Spent Nuclear Fuel Attribution using Statistical Methods: Impacts of Information Reduction on Prediction Performance

by

Arrielle C. Opotowsky

A dissertation submitted in partial fulfillment of

the requirements for the degree of

Doctor of Philosophy

(Nuclear Engineering & Engineering Physics)

at the

UNIVERSITY OF WISCONSIN–MADISON

2021

Date of final oral examination: 16 August 2021

The dissertation is approved by the following members of the Final Oral Committee:
Sunil S. Chirayath, Associate Professor, Nuclear Engineering, Texas A&M University
Douglass L. Henderson, Spangler Professor, Nuclear Engineering & Engineering Physics
Benjamin A. Lindley, Assistant Professor, Nuclear Engineering & Engineering Physics
Robert D. Nowak, Nosbusch Professor, Electrical & Computer Engineering
Paul P.H. Wilson, Grainger Professor, Nuclear Engineering & Engineering Physics

© Copyright by Arrielle C. Opotowsky 2021

All Rights Reserved

For onkwehshón:'a (the People)

For iokhih'nisténha ohóntsia (Mother Earth)

For ohneka'shón:'a (the Waters)

For kentsionshón:'a (the Fish)

For tsi shonkwaienthó:wi (the Plants)

For kaien'thóhshera (the Food Plants)

For ononkwa'shón:'a (the Medicinal Herbs)

For kontírio (the Animals)

For okwire'shón:'a (the Trees)

For otsi'ten'okón:'a (the Birds)

For owera'shón:'a (the Four Winds)

For ratiwé:ras (the Thunder Beings)

For kionhkehnéhkha karáhkwa (Brother Sun)

For ionkhih'sótha ahsonthenhnéhkha karákwa (Grandmother Moon)

For otsistanohkwa'shón:'a (the Stars)

For kaié:ri niionkè:take (the Four Beings)

For shonkwaia'tíson (the Creator)

éhtho niiohtónha'k ne onkwa'nikón:ra (and now our minds are one)

ACKNOWLEDGMENTS

This material is based upon work funded by the National Science Foundation Graduate Fellowship Program, as well as the United States (US) Department of Homeland Security (DHS) under Grant Award Number 2012-DN-130-NF0001.[1]

This dissertation would not be possible with the wisdom and patience of my advisor, Paul Wilson, and the amazing community he fostered, CNERG. Kelly Burton and Max Lagally both convinced me to remain in graduate school path—more than once. And Steven "if you're gonna be dumb you gotta be tough" Harrell also kept me in school—more than once, and even after cancer took him too soon.

Niá:wen'kówa to the people who taught me who I am and taught me how to heal.

My GERS friends have been with me since the beginning of this journey, especially Robert, Richard, Chandler, Dinh, Nhi. Extra love to Kalin and Chelsea, who were there in some dark times and some well-lit ones too. For tirelessly encouraging my academic and personal pursuits, I'm appreciative of my California family, Mel, Bonnie, Joelle, and Jamie. Thanks to Didier for the housing while I wrote my dissertation in a pandemic while living nomadically. Lou, my seestre! Olivia, my bean sister; thank you for the nitrogen these last few months. Heather, thanks for being so supportive of me, for years and years; I'll come get my mail, I promise. Robin, you have been such a light in my life for nearly 15 years, and I know I wouldn't be where I am today without your friendship and beautiful way of seeing the world. And my last thank you is to Kreezy, who gracefully tolerate my existence in their apartment even as I write these acknowledgments, and whose friendship I am so blessed to have. Konorónkwa.

Ohtá akwé:kon wátons!

[1]The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the US DHS.

CONTENTS

Contents iii

List of Tables vii

List of Figures viii

Acronyms xi

Abstract xiv

1 Let Them Eat Steak: A Chapter for the Non-Scientist 1

2 Introduction 20

2.1 Motivation . . . 22

2.1.1 Needs in Nuclear Forensics . . . 24

2.1.2 Contribution of Statistical Methods . . . 26

2.2 Methodology . . . 27

2.3 Goals . . . 31

3 Background and Literature Review 34

3.1 Nuclear Forensics . . . 34

3.1.1 Types of Nuclear Forensics Investigations . . . 35

3.1.1.1 Post-Detonation . . . 35

3.1.1.2 Pre-Detonation . . . 36

3.1.2 Nuclear Forensics as an Inverse Problem . . . 37

3.1.3 Nuclide Signatures for Nuclear Forensics of Spent Fuel . . . 39

3.2 Machine Learning . . . 42

3.2.1 Algorithms for Statistical Learning . . . 44

3.2.1.1 Nearest Neighbors . . . 45
3.2.1.2 Decision Trees . . . 46
3.2.1.3 Maximum Log-Likelihood Calculations . . . 50
3.2.2 Algorithm Performance . . . 51
3.2.2.1 Testing Error . . . 51
3.2.2.2 Model Complexity . . . 55
3.3 Applications of Statistical Methods to Nuclear Forensics Analysis . . . 58
3.3.1 Factor Analysis Work . . . 59
3.3.2 Other Classification Work . . . 61
3.3.3 Regression Work . . . 62
3.3.4 Maximum Log-Likelihood Calculations . . . 63
3.3.5 Summary . . . 65

4 Reactor Parameter Prediction Using Nuclide Masses 68
4.1 Training Data Simulation . . . 70
4.1.1 Simulation Fidelity . . . 70
4.1.2 Training Set Labels . . . 74
4.1.3 Training Set Features . . . 76
4.2 Information Reduction . . . 77
4.2.1 Scikit Algorithms . . . 78
4.2.2 Maximum Log-Likelihood Calculations . . . 78
4.3 Statistical Learning Implementation . . . 79
4.3.1 Scikit-learn Algorithms . . . 79
4.3.2 Maximum Log-Likelihood Calculations . . . 83
4.4 Prediction Performance Evaluation . . . 83
4.4.1 Random Error Impacts on Prediction . . . 84
4.4.1.1 Reactor Type Classification . . . 84

4.4.1.2 Regression Results 88
4.4.1.3 Model Generalization 91
4.4.1.4 Reactor Type Prior Knowledge 94
4.4.2 SFCOMPO Test Set 97
4.4.2.1 Reactor Type Classification 99
4.4.2.2 Regression Cases 103
4.5 Summary 111

5 Reactor Parameter Prediction Using Processed Gamma Spectra 115
5.1 Training Data Simulation 116
5.2 Information Reduction 118
5.3 Statistical Learning Implementation 125
5.4 Performance Evaluation 128
5.4.1 Reactor Type Classification 130
5.4.2 Regression Results 138
5.4.2.1 Burnup Regression 139
5.4.2.2 ^{235}U Enrichment Regression 143
5.4.2.3 Time Since Irradiation Regression 147
5.5 Summary 152

6 Conclusions and Future Work 155
6.1 Summary 155
6.2 Conclusions 156
6.3 Future Work 161
6.3.1 Training Set Features 161
6.3.2 Statistical Method Optimization 164
6.3.3 Serial Prediction 164
6.3.4 SFCOMPO 166

References 168

LIST OF TABLES

3.1 Example of results from INDEPTH 39

4.1 Example of SCALE simulation benchmarking results 72
4.2 Training set database design for ORIGEN input files. 74
4.3 Set of nuclide features for first experiment 77
4.4 Optimized algorithm hyperparameters for the 29 nuclide training set . . . 81
4.5 Comparison of different CV implementation errors 82
4.6 MAPEs for three regression cases comparing known versus unknown reactor type prior knowledge 95
4.7 Number of assays each nuclide is measured for in SFCOMPO 99
4.8 Performance of reactor type classification of SFCOMPO entries 100
4.9 Performance of burnup regression of SFCOMPO entries 104
4.10 Performance of enrichment regression of SFCOMPO entries 108

5.1 Set of nuclide features for second experiment 117
5.2 Details of detector setups 120
5.3 Energy window sizes and list lengths for processing gamma spectra 122
5.4 Nuclides that are represented by the short and long energy windows lists . 123
5.5 Optimized algorithm hyperparameters for all training sets in second experiment 127

LIST OF FIGURES

2.1 Interdicted nuclear materials for trafficking or malicious use 23
2.2 Example nuclear forensics investigation workflow 25
2.3 Comparison of nuclear forensics workflow against research approaches . . . 28
2.4 Experimental methodology flowchart 30

3.1 Schematic of supervised machine learning process 43
3.2 Schematic of k-nearest neighbors regression 46
3.3 Example of a decision tree 48
3.4 Example of a confusion matrix 53
3.5 Illustration of cross validation 54
3.6 Schematic of the bias-variance trade-off 56
3.7 Diagram of model performance with respect to model complexity 57
3.8 Example of factor analysis for SNF discrimination 60
3.9 Example of dimensionality reduction for visualization 62
3.10 Burnup performance with respect to training set error 63
3.11 Example of likelihood maximum predicting burnup and time since irradiation 64
3.12 Sensitivity of likelihood to training set uncertainty 65

4.1 Experimental methodology in Chapter 4 68
4.2 First portion of the flowchart from Figure 4.1 70
4.3 Visualization of training set labels distributions 75
4.4 Second portion of the flowchart from Figure 4.1 77
4.5 Third portion of the flowchart from Figure 4.1 79
4.6 Fourth portion of the flowchart from Figure 4.1 83
4.7 Prediction performance of reactor type classification with increasing training set error 85

4.8 Confusion matrices of reactor type classification 86
4.9 Prediction performance of burnup regression with increasing training set error 89
4.10 Prediction performance of enrichment regression with increasing training set error 90
4.11 Prediction performance of time since irradiation regression with increasing training set error 90
4.12 Learning curves for all four prediction parameters 92
4.13 Heatmaps for three regression cases comparing known versus unknown reactor type prior knowledge 96
4.14 Scatter plot of distribution of SFCOMPO testing set labels 98
4.15 Confusion matrices of reactor type classification of SFCOMPO entries . . . 101
4.16 Box plots of burnup regression of SFCOMPO entries 105
4.17 True versus predicted burnup of SFCOMPO test cases 106
4.18 Box plots of enrichment regression of SFCOMPO entries 109
4.19 True versus predicted enrichment of SFCOMPO test cases 110

5.1 Experimental methodology in Chapter 5 115
5.2 First portion of the flowchart from Figure 5.1 116
5.3 Second portion of the flowchart from Figure 5.1 118
5.4 Portion of gamma spectrum with windows, which shows summation regions for spectra processing 124
5.5 Third portion of the flowchart from Figure 5.1 125
5.6 Fourth portion of the flowchart from Figure 5.1 128
5.7 Demonstration of plot being used to evaluate the results 128
5.8 Prediction performance of reactor type classification with decreasing detector energy resolution 130
5.9 Confusion matrices of reactor type classification for non-detector training sets 133

5.10 Confusion matrices for auto energy window list training sets. 134
5.11 Confusion matrices for short energy window list training sets. 135
5.12 Confusion matrices for long energy window list training sets. 136
5.13 Prediction performance of burnup regression with decreasing detector energy resolution 140
5.14 Box plots (without outliers) of burnup regression for six detectors 141
5.15 Box plots (with outliers) of burnup regression for six detectors 142
5.16 Prediction performance of ^{235}U regression with decreasing detector energy resolution 144
5.17 Box plots (without outliers) of ^{235}U enrichment regression for six detectors 145
5.18 Box plots (with outliers) of ^{235}U enrichment regression for six detectors . . 146
5.19 Prediction performance of time since irradiation regression with decreasing detector energy resolution 148
5.20 Box plots (without outliers) of time since irradiation regression for six detectors 149
5.21 Box plots (with outliers) of time since irradiation regression for six detectors 150

6.1 Example of an ROC curve 166

ACRONYMS

$LaBr_3$ lanthanum bromide. 120, 121, 126

SrI_2 strontium iodide. 120, 126, 131, 132, 137, 138, 153, 158

^{235}U uranium-235. x, xiv, 7, 10, 16, 21, 27, 39–41, 69, 71, 74, 78, 84, 89, 90, 92, 96, 103, 107, 110, 111, 117, 118, 128, 144–146, 155, 163

AI artificial intelligence. 42

BWR boiling water reactor. 6, 21, 22, 49, 50, 53, 61, 69, 76, 81, 85, 87, 88, 93–95, 97, 98, 100–102, 112, 114, 117, 132, 137, 138, 153, 155, 158, 165, 166

CHTC Center for High Throughput Computing. 82, 83, 156

CV cross-validation. vii, 54, 81, 82, 156, 160

CWMD Countering Weapons of Mass Destruction. 20

CZT cadmium zinc telluride. 120, 126, 151

DHS Department of Homeland Security. ii, 20

DNDO Domestic Nuclear Detection Office. 20

DOE Department of Energy. 20

DRF detector response function. 117–119, 152, 155

FWHM full width at half maximum. 120

GADRAS GAmma Detector Response and Analysis Software. 116–120, 152, 155, 160, 162, 164

HPGe high-purity germanium. 120, 124, 126, 131, 132, 137, 138, 140, 143, 144, 147, 151, 153, 158, 163

i.i.d. independent and identically distributed. 58, 75

IAEA International Atomic Energy Agency. 22, 23

INDEPTH INverse DEPletion THeory. vii, 26, 38, 39

ITDB Incident and Trafficking Database. 22, 23

LWR light water reactor. 40, 41, 73, 87, 88

MAE mean absolute error. 53, 54, 82, 88–91, 95, 139, 140, 143, 147, 151, 158, 159

MAPE mean absolute percentage error. vii, 53, 54, 88–90, 92–95, 138–140, 143, 144, 147, 148, 158

MedAE median absolute error. 53, 54, 88, 89, 91, 139, 143, 147, 151, 158

ML machine learning. 27, 29, 31, 32, 42, 44, 51, 55, 58, 75, 78, 79, 112, 118, 156

MLL maximum log-likelihood. 10, 30, 31, 44, 50, 63, 65, 66, 78, 79, 82–85, 87–89, 91, 93, 94, 96, 97, 99, 100, 102–109, 111–114, 124, 125, 128, 130–132, 137, 140, 143, 144, 147, 148, 151–153, 156–160, 164, 165, 167

NaI sodium iodide. 120, 121, 126, 151

NNSA National Nuclear Security Administration. 20

NTNFC National Technical Nuclear Forensics Center. 20

ORIGEN Oak Ridge Isotope GENeration. vii, 29, 59, 60, 68, 70, 71, 74, 77, 112, 116, 119, 155

ORIGEN-ARP ORIGEN-Automatic Rapid Processing. 38, 71, 73, 76, 161, 162

OSG Open Science Grid. 83, 156

PHWR pressurized heavy water reactor. 6, 21, 22, 49, 53, 69, 73, 76, 81, 85, 87, 88, 93–98, 100–102, 104, 108, 112–114, 117, 132, 137, 138, 153, 155, 157, 158, 164, 166, 167

PWR pressurized water reactor. 6, 21, 22, 50, 53, 61, 69, 72, 76, 81, 85, 87, 88, 93–103, 112, 114, 117, 132, 137, 138, 153, 155, 157, 158, 164, 166

RMSE root-mean-squared error. 53

ROC receiver operating characteristic. x, 165, 166

SCALE Standardized Computer Analyses for Licensing Evaluation. vii, 29, 71–73, 162

SFCOMPO Spent Fuel isotopic COMPOsition. vii, ix, 60, 65–67, 72–74, 76, 77, 97–100, 103–106, 108, 110, 112, 114, 157–160, 162, 164, 166, 167

SNF spent nuclear fuel. viii, xiv, 21, 22, 24–29, 32, 35–38, 40, 44, 59–61, 65, 66, 68–71, 73, 76, 78, 97, 112, 115–121, 152, 153, 155, 157, 165

SNM special nuclear material. 24, 26, 35, 36

UOC uranium ore concentrate. 22, 24, 36, 59, 66

US United States. ii, 20, 25, 35

UW University of Wisconsin. 82, 83, 156

ABSTRACT

Nuclear forensics is a nuclear security capability that is broadly defined as material attribution in the event of a nuclear incident. Improvement and research is needed for technical components of this process. One such area is the provenance of non-detonated special nuclear material; studied here is spent nuclear fuel (SNF), which is applicable in a scenario involving the unlawful use of commercial byproducts from nuclear power reactors. The experimental process involves measuring known forensics signatures to ascertain the reactor parameters that produced the material, assisting in locating its source. This work proposes the use of statistical methods to determine these quantities instead of empirical relationships.

The purpose of this work is to probe the feasibility of this method with a focus on field-deployable detection. Thus, two experiments are conducted, the first informing the second by providing a baseline of performance. Both experiments use simulated nuclide measurements as observations and reactor operation parameters as the prediction goals. First, machine learning algorithms are employed with full-knowledge training data, i.e., nuclide vectors from simulations that mimic lab-based mass spectrometry. The error in the mass measurements is artificially increased to probe the prediction performance with respect to information reduction. Second, this machine learning workflow is performed on training data analogous to a field-deployed gamma detector that can only measure radionuclides. The detector configuration is varied so that the information reduction now represents decreasing detector energy resolution. The results are evaluated using the error of the reactor parameter predictions.

The reactor parameters of interest are the reactor type and three quantities that can attribute SNF: burnup, initial ^{235}U enrichment, and time since irradiation. The algorithms used to predict these quantities are k-nearest neighbors, decision trees, and maximum log-likelihood calculations. The first experiment predicts all of these quantities

well using the three algorithms, except for k-nearest neighbors predicting time since irradiation. For the second experiment, most of the detector configurations predict burnup well, none of them predict enrichment well, and the time since irradiation results perform on or near the baseline. This approach is an exploratory study; the results are promising and warrant further study.

1 LET THEM EAT STEAK: A CHAPTER FOR THE NON-SCIENTIST

This chapter was written to convey my PhD work to the general public and was supported by the Wisconsin Initiative for Science Literacy (WISL). I have much gratitude to WISL and Prof. Bassam Shakhashiri for the editing assistance and the opportunity.

Writing this chapter is a also result of me keeping a promise to myself, and so despite its cheesy approach to telling a tale of science, it is a beautiful and important moment for me. I have a lot of people to credit for helping bring this story from a parallel universe into reality: Anna Stephenson for the illustrations and helping me convert my graphics from sterile science to adorable art; Robin Kinchen Cenac (and Reya!) and Louise Opotowsky for overall creative guidance and for suffering through highly technical explanations of my work to prepare me for writing this chapter; Prof. Paul P.H. Wilson, Almost-Dr. Kalin Keisling, Dr. Dinh Truong, and Dr. Richard Rojas Delgado for feedback and suggestions on my fake country names; and last, never least, but always the littlest, Ninjita Binjita, for the all-important role of lap warmer.

Narrator:
Welcome, curious companions! Our good friend has got a tale to tell. But they cannot tell this story on their own, so they asked me to give you some background and science along the way.

Be warned: the country names are drawn from a parallel universe with different nations and international relations. Any similarities to countries that exist in this universe are purely coincidental. Additionally, there are fantastical details throughout the tale, and the capability of our curious companions to decipher between fantasy and science is presumed. This parallel universe also doesn't have an Earth with the same climate crisis, so the steak in this analogy is definitely from a happy cow on a regenerative farm.

Background & Introduction

Many underappreciated jobs keep a civilization functioning. For example, excluding

New Orleanians and other People of the Pothole (yes, New Orleans exists in the parallel universe), you probably don't think about how you hold the expectation that your roads are drivable. There are those responsible for moving your garbage out of sight and mind, there are also people who clean up roadkill, and there are those who clear the shards of a car accident with fascinating speed. In fact, when any civic role functions well, it isn't noticed. It is an odd result of a well-functioning society that the most essential components remain unseen until they no longer function. Jobs like this exist at the federal level, generally unseen, because they are so crucial they regularly get bipartisan support. This is a story about *those* people.

Now to our friend…

> Imagine the scene: they were sitting in their backyard in perfect weather, breeze blowing, flowers flowering, and chipmunks chirping, eyes closed as the sun warms their skin. Suddenly they felt a chill, and opened their eyes to a dark sky.
>
> Except it wasn't a dark sky, it was a drone hovering over them with a package for delivery! C'est mystérieux! They hadn't ordered anything. What could it be?
>
> Why, it was a package of nuclear material *(Narrator: well-packaged, because we are not irradiating our friend)* delivered anonymously. Turns out, they unknowingly intercepted the attempted smuggling of nuclear material to construct a weapon inside the borders of United Fissions of Uranium (the UFU). And now our friend is officially in the middle of an international drama. What to do? Who to tell?

Narrator:
I actually don't know who they should have told; federal jurisdictional decisions for nuclear incidents is not the drama being told today. But the authorities quickly found out and figured that out for themselves.

Our friend's day is quite ruined. Illustration by Anna Stephenson

This turns out to be a misdelivered package, because nuclear terrorists are people too…that sometimes make typos. The UFU authorities believe that there are many more packages on their way to different locations, but having no intel on where to intercept them, they need to know where this material came from to locate the terrorist group responsible. They need nuclear security experts, and FAST.

Narrator:

Enter: nuclear security. This is not to be confused with nuclear safety, that is, making sure nuclear power reactors behave and do not have accidents that harm the environment

and the beings in it—a more-than-worthy effort, but not the one being discussed here. The nuclear security enterprise instead focuses on preventing or mitigating undesirable outcomes of a different variety, like nuclear terrorism. Nuclear security's goal is keeping all of the nuclear material in the world inside a regulatory pipeline, so none of it gets into the hands of people who want to do others harm.

In this universe:
A high profile example of nuclear security at work, at least on a diplomatic level, is the Joint Comprehensive Plan of Action[1], better known as the Iran nuclear deal. Personal opinions (if you have them) and recent news (if you've seen it) aside, its purpose is to keep a closer eye on the country to be sure they aren't developing the capabilities necessary to make weapons. Another part of the nuclear security effort is a strong nuclear forensics capability. Nuclear forensics begins **after** a nuclear incident occurs, which sadly, happens. This incident can be some intact material drone-delivered to a friend by mistake, or it could be something even worse, like the detonation of a nuclear weapon. Just in 2019, the International Atomic Energy Agency confirmed malicious intent for six incidents of trafficked nuclear material[2].

Most might think of forensics as catching a murderer, but this is more like catching the nuclear smuggler. Given some nuclear material (a body) and composition of the material/how it was encased and transported (the clues around it like blood and fingerprints), how/from where was the nuclear material obtained and/or smuggled (what conclusions can be drawn about the murder)? In both situations, forensics work ideally leads to blaming, with court-admissible proof, someone for the illegal act. Fingerprints of humans are important to a murder investigation, and likewise, there are fingerprints of nuclear materials that can provide their point of origin and/or where they were processed.

> Slight correction: They need **nuclear forensics** experts, and FAST.
>
> These UFU authorities are in luck, since our friend happens to be a hobby

[1]For more information on the JCPOA, see this fact sheet
[2]Here is the 2020 document that contains this information

> nuclear forensics scientist! As a citizen scientist, they cannot use actual nuclear materials or well-equipped laboratories to test their methods and ideas. Our friend instead uses their software development and simulation skills to study their favorite topic of attributing mysterious nuclear materials to their point of origin. This is now their chance to unveil a research method to the authorities and see if they can help prevent a nuclear weapon from being detonated in the UFU in time.
>
> But the authorities are not sure. Some experimental method developed by a ~~grad student~~ hobby scientist surely wouldn't work? Also, it's not validated, so it wouldn't hold up in court. But the race to save lives is on. "What," they ask, "do ya got cookin'?"

Narrator:
I'll tell you all about what our friend has cookin': some steak. But hold on, I'll get there in a minute.

From a visual inspection, the nuclear material in question has been determined to be nuclear fuel after it's been loaded into, used in, and removed from a nuclear reactor. By performing some to-be-discussed nuclear forensics approaches on this material, we can figure out all of the details of this fuel related to its creation, time in the reactor, and how long it has been out of the reactor.

First, I need to define some terms for you. There are four main concepts that are covered: *reactor type*, *burnup*, *enrichment*, and *time since irradiation*. Ideally, the process of determining these parameters can pinpoint a sample of nuclear fuel to the exact reactor it came out of!

Let's consider the nuclear fuel as food, specifically, steak. We can think of the reactor as the type of pan our steak was cooked in. If it's cast iron, it'll make a different steak than a $10 nonstick pan that's only nonstick for 3 uses (the parallel universe shares some similar woes). The same is true for nuclear fuel; it looks quite different depending on which reactor type it spent time in. Our friend focuses on three main types of nuclear

If you imagine nuclear fuel as steak, you might be able to figure out the reactor that made it! Illustration by Anna Stephenson

reactors, called pressurized water reactors (PWRs), boiling water reactors (BWRs), and pressurized heavy water reactors (PHWRs)[3]; different countries use one or a mix of these three main technologies. (More than these three exist, but these are the ones our friend wants to focus on.)

There's also a measurement called burnup. In steak-talk, this is how well-done it is (more accurately, it is how much energy your steak produces, but it is more "well-done" as it cooks longer and produces more energy), and would be measured in energy produced

[3]We won't cover any details about these reactors here, but if you're curious about different types of nuclear reactors, here is a great summary.

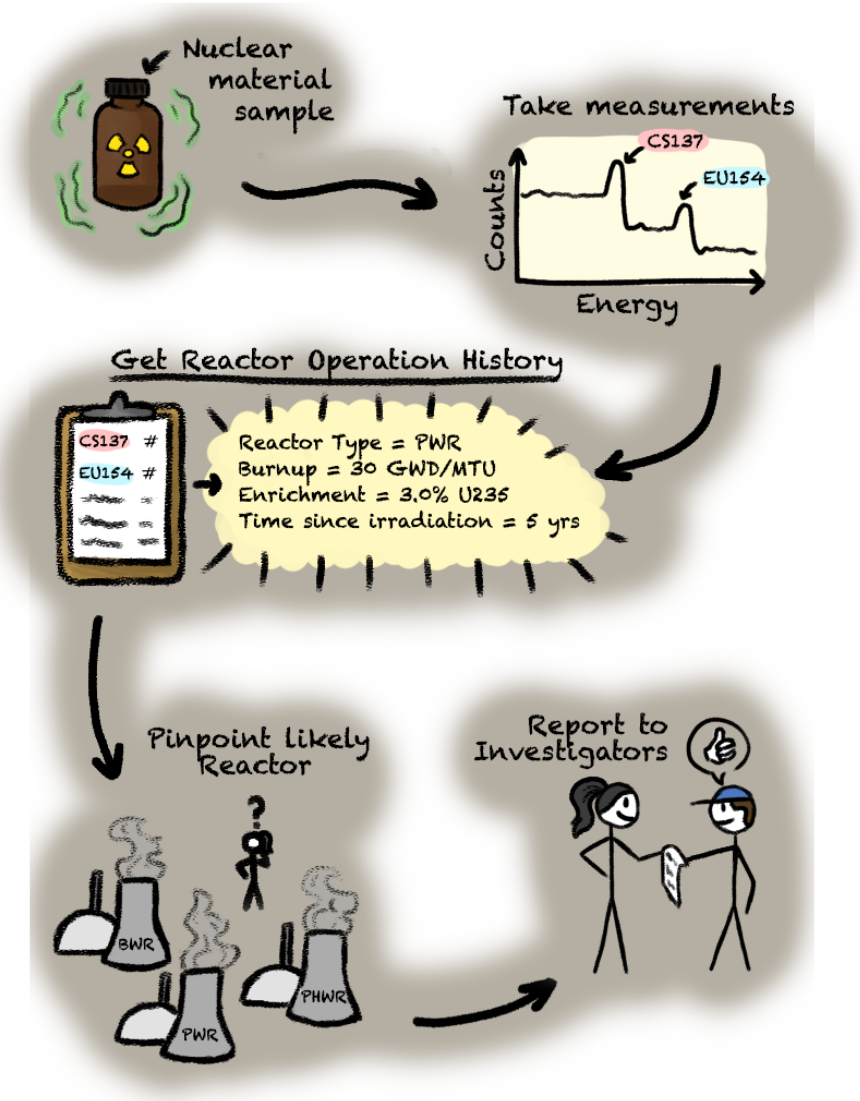
per unit of raw steak. In nuclear-talk, it's measured in energy (mega- or gigawatt-days) per metric ton of initial uranium (MWd/MTU or GWd/MTU).

Next, the enrichment, meaning % uranium-235 (^{235}U) enrichment, which refers to how much of this type of uranium is in the nuclear fuel when it's freshly made. A lot of the time, nuclear engineers refer to a specific element from the periodic table with a mass number attached, like ^{235}U, as nuclides, because the concept of nuclides emphasizes nuclear properties, which can differ drastically even though they are the same element on the periodic table. "U" is shorthand for uranium, and the mass number 235 refers to the number of protons (92) plus neutrons (143) in the nucleus of the atom. The protons have a positive charge, and the neutrons have no charge; the protons are balanced by the negative charge of an equal number of electrons, but we aren't worried about those right now. ^{235}U is a special nuclide that nuclear engineers call *fissile*[4]: when it absorbs an extra neutron, it splits into two atoms and releases some energy. When this energy is harnessed into our electrical grid, it's great, but that energy can also be harnessed into a weapon, which is not great. This is like the calorie content or fat content of your steak. The more fat, the more calories, and so the more energy it can supply. In nuclear fuel, more fissile material in the form of a higher ^{235}U enrichment means that the fuel can provide more energy than a fuel of lower enrichment. Uranium naturally has 0.7% ^{235}U in it, but commercial nuclear fuel is commonly enriched up to 5%.

Last, the time since irradiation measures how long the nuclear fuel, or steak, has been cooling after it leaves the reactor, or pan. Nuclear fuel is intensely radioactive when it leaves the reactor, which produces a lot of heat, so it needs to cool off for a few years to be able to be stored longer term without heat dissipating measures (which is submerging the nuclear fuel in water; think of the fuel as taking a several year vacation in a swimming pool). This is just like our steak needing to rest a little before it's consumed. And that's about as far as this metaphor can go, because aside from some recent Godzilla movies, I don't think any of us are eating nuclear waste (in this universe, at least).

[4]For more information on nuclides and what fissile means, check out this link.

If a nuclear material spent time in a nuclear reactor, these four parameters, part of what we will call the reactor operation history, are important to identifying where it came from. Next, we will talk about how identifying where it came from can happen in an investigation.



It's as simple as this ↑. Illustration by Anna Stephenson

After some of-unknown-origin nuclear material believed to be from a reactor enters

our consciousness, some measurements will be taken by technicians working with the government. For example, this could be something called *gamma spectroscopy*, which is pictured in the process below. This detector measures a type of radioactivity called gamma rays[5], and gamma rays have different energies. So the material is just sitting there spitting out gamma rays left and right and up and down and the detector is just sitting there measuring the ones that hit it. It collects counts of gamma rays associated with an energy; this is called a gamma spectrum. Gamma rays of certain energies are known to come from certain nuclides.

Knowing how much of certain radioactive nuclides is in a material can tell us about the reactor operation history: the reactor type (pan), burnup (doneness), enrichment (fat), and time since irradiation (rest time). After determining these parameters, a specialist can pinpoint a specific reactor somewhere in the world (via access to a reactor history database) that created the material and investigators can use that to move forward with their work.

Methodology

In the time it took to get through the lesson above, a foe had enough time to arrive on the scene: an official government scientist. This scientist has a different priority: precision over speed. Our friend's research is driven by "how fast can I get an answer?", whereas the scientist is driven by "what's the most correct answer?" These two priorities in this situation are at odds, but both equally important. The authorities need an answer, and fast, but it needs to be the right one because otherwise many UFU lives are at risk.

Our friend was in the middle of telling the authorities about their fast nuclear fuel-identifying machine learning approach when this scientist arrived, so they got to listen in:

"Machine learning is a field under the umbrella of artificial intelligence, which

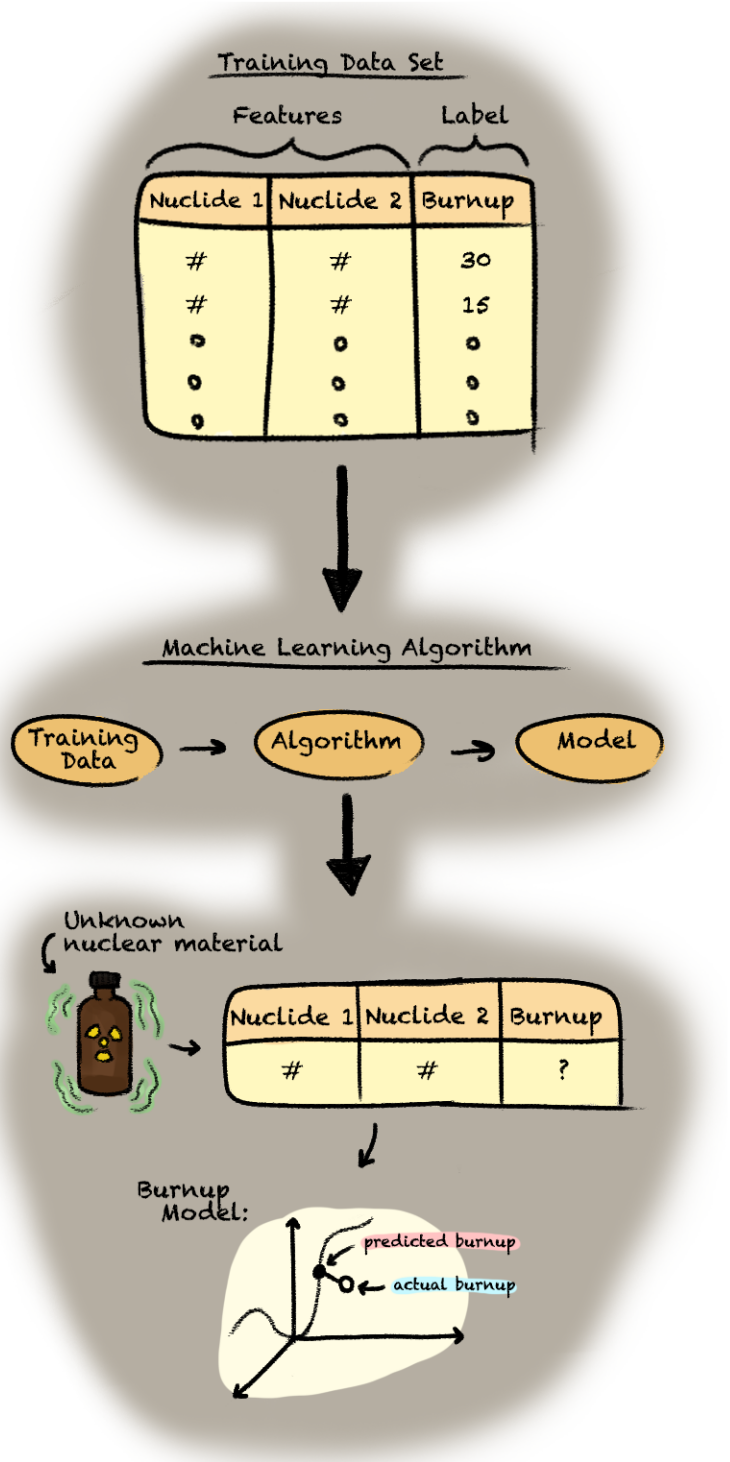
[5]Gamma rays are really cool, and if you read this article, you'll think so too!

allows computers to imitate human behavior. Scientists are using machine learning in many fields to solve complex problems, and so I wanted to see if it could be useful in my favorite area of study: nuclear forensics."

Pointing to the top of the diagram, they say, "So, I first simulated hundreds of thousands of examples of nuclear fuel scenarios: different reactor types, many levels of burnup, different levels of ^{235}U enrichment, and times since irradiation spanning up to 16 years. Each simulation gives me lists of nuclides and their measurements that are important to determining those four parameters. Machine learning professionals call these lists of nuclides the *features*, and the parameters are *labels*. All together, it's called a *training data set*."

Pointing to the middle, "Next, this training data is put into a *machine learning algorithm*[6], which is how people teach computers to teach themselves with some software method. Using the training data set, the algorithm creates a model, which is usually a model we can't see or understand as humans. They are quite secretive creatures, don't you think? Anyway, there are many different types of algorithms, and I have tested out some simple ones to see if this approach is even remotely feasible. These also happen to be the less-secretive type of algorithms so we can understand what the models are doing. One seems to work really well, called maximum log-likelihood (MLL) calculations[7], and I think it's good enough to use to save UFU."

Last, they point to the bottom of the diagram. "So now we have the model. If we take the same measurements that exist in the training database features, then we can use the model to give us a predicted label, in this case burnup. But because I'm doing this experimentally, I know the actual label because I simulated this unknown nuclear material. So in this way, I can measure the prediction errors and refine my method."



Again, it's as simple as this ↑. Illustration by Anna Stephenson

The UFU authorities' eyes glazed over, but the scientist was excited. They were thinking, "My oh my, we could use this! I have a database just like this of the most perfect simulated nuclide measurements that I can use back at the lab! I never really knew what to do with it, so I took a screenshot of a few entries and used it as my desktop background; databases are beautiful."

But our friend didn't think the scientist could take the proper measurements in time. Our friend uses this method with a different kind of training set, one that is created by simulating detectors that can take measurements in minutes *(Narrator: remember the gamma spectroscopy from above?)*, with the expectation that this would help in a real world scenario like this one. The gamma detectors measure the radioactivity of the sample, which is more difficult to get a direct answer from than the scientist's method in the lab, but our friend is all about speed. The measurements the scientist needs to take to match the sample with their training database involve dissolving the material and making many different measurements of the nuclides using a technique called mass spectrometry[8]. It gives super accurate results that will do well with the machine learning method, but the measurements take weeks.

They fought about this for about an hour, which was silly because our friend could have taken the gamma measurements in a fraction of that time and been off to use their machine learning model. But, tensions were high, egos were flaring, and everyone wanted to save lives.

The UFU authorities deglazed their eyes and looked at each other, then at our friend, then at the scientist. After some telepathic decision making, they said, "We choose……"

[6]For a better introduction to machine learning, read this!

[7]If you have institutional access to journals, here's the method's first paper.

[8]I tried to find a non-company-affiliated source that explained this simply, but failed. This is a good explanation, though, if you're curious about mass spectrometry.

Narrator:

Now, you, curious companion, must choose your own adventure. Do we use our friend's speedy strategy or do we trust the scientist's careful course of action? Remember, we want to be fast, which our friend can most likely do, but we also want to be right, which the scientist can most likely do.

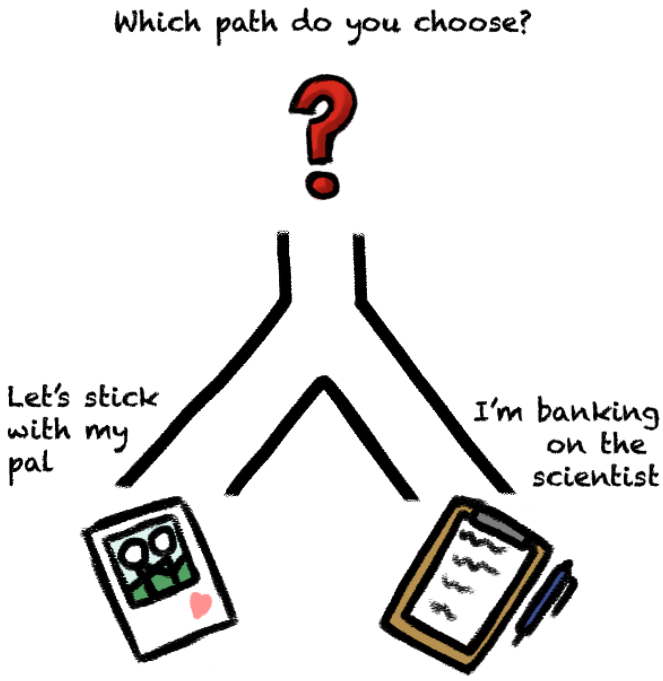
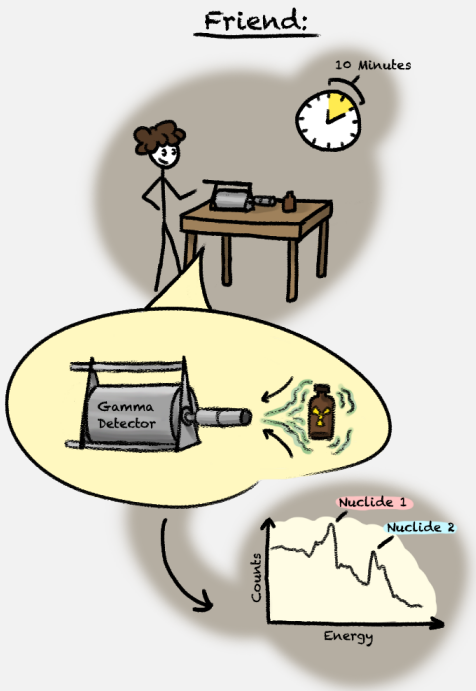


Illustration by Anna Stephenson

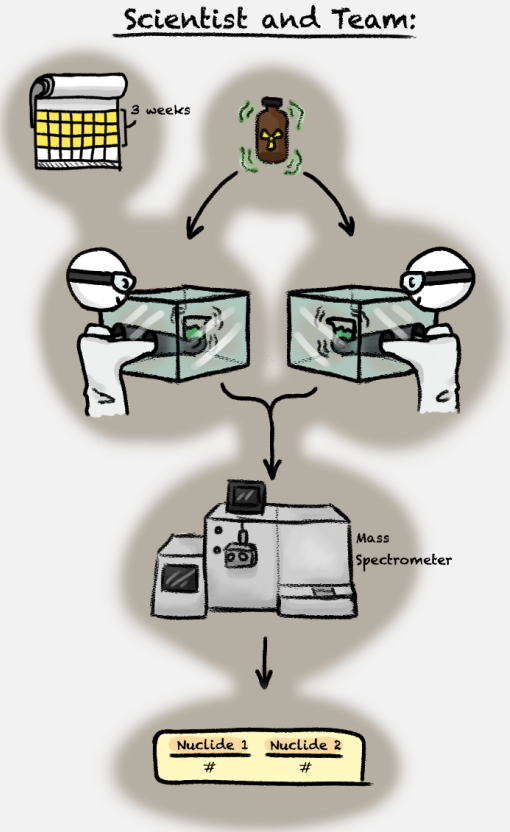
"...our friend!" Now the race is on to measure the material with a gamma detector and predict the fuel's reactor operation history. For a hobbyist, our friend has a pretty great gamma detector: a portable high-purity germanium detector. It can detect the gamma rays very precisely, and our friend always wants to use the best detector they can get their hands on.

"...the scientist!" Now the scientist takes the nuclear fuel to start making measurements. Back in their fancy state-of-the-art lab with all the mass spectrometry equipment a radiochemist could ever dream of, the scientist and their team get started. One week goes by, two weeks go by. And by the third week the scientist and their team had measured 29 nuclides in the nuclear fuel sample to high precision!



This process doesn't require advanced training. Illustration by Anna Stephenson

Using the technical assistance of the UFU authorities, our friend was able to protect themself against radiation and take the sample out of its packaging to get the best measurements possible. They let the detector measure the sample for 10 minutes, et voilà: a gamma spectrum of the sample. Our friend then took the gamma spectrum and compared it against their machine-learned model that was created using a training set composed of 450,000 simulated gamma spectra of different types of nuclear fuel. And out popped an answer:



This process is complicated, but here are some snapshots.
Illustration by Anna Stephenson

Now they were ready to borrow our friend's machine learning method to predict the parameters of the reactor operation history. The scientist then took the list of 29 nuclides and their measurements and compared that information against their machine-learned model that was created using a training set composed of 450,000 of the exact same 29 simulated measurements of different types of nuclear fuel. And out popped an answer:

Results

Reactor Type	BWR
Burnup	44.02 GWd/MTU
Enrichment	2.04 % ^{235}U
Time Since Irrad	5.34 $years$

"Ok! We got it!" said our friend. Given these values, some of the UFU authorities specializing in worldwide reactor operational history databases were able to determine this came from a reactor in the Democratic People's Republic of Thoria (DPRT).

This made sense to everyone because the DPRT had been a threat for some time. Everyone knew their missiles couldn't get to the UFU, so they must have concocted a different plan.

It was a matter of hours before the UFU had hundreds of DPRT conspirators in custody. With the culprits contained, the drone-delivered material didn't make it to the bomb assembly location, and the day was saved!

…Except, three weeks later, the capital city of Curiumville was bombed.

Reactor Type	BWR
Burnup	44.02 GWd/MTU
Enrichment	4.11 % ^{235}U
Time Since Irrad	4.65 $years$

"Ok! We got it!" said the scientist. Given these values, some of the UFU authorities specializing in worldwide reactor operational history databases were able to determine this came from a reactor in …GASP!

The Commonwealth of Puerto Plutonio, a Territory of the UFU?! This made no sense! We thought they liked being colonized!

It was now a rush to track down the conspirators since there wasn't much intelligence data on them. The UFU was scrambling.

And in the middle of the scramble, the capital city of Curiumville was bombed.

Narrator:
Now, curious companion, you are both permitted and encouraged to read the other adventure.

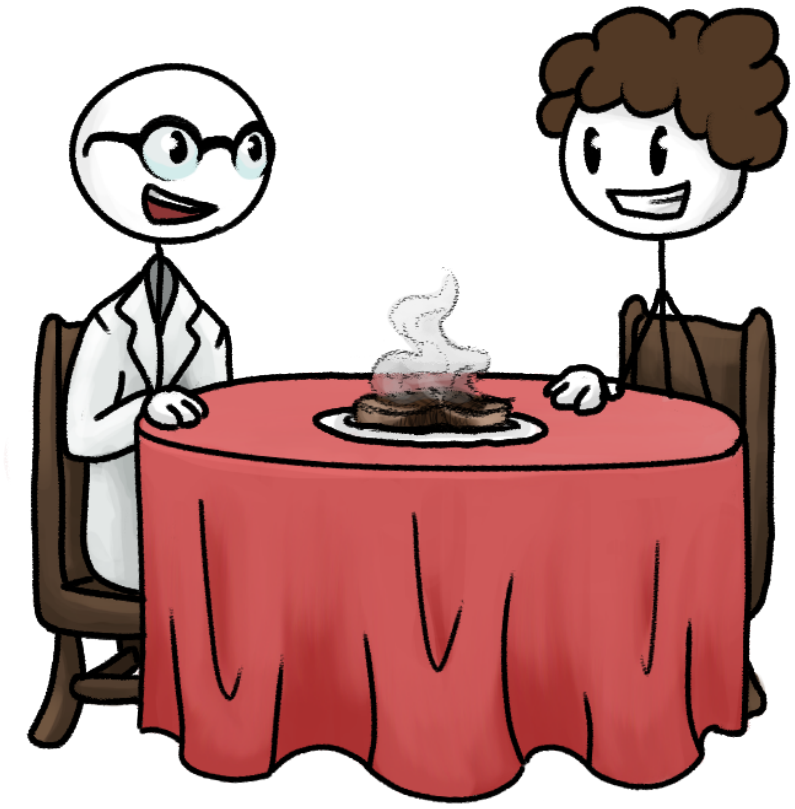
After weeks went by the UFU authorities with the help of the scientist were able to confirm the actual parameters:

Reactor Type	BWR
Burnup	44.02 GWd/MTU
Enrichment	4.11 % ^{235}U
Time Since Irradiation	4.86 $years$

Our friend's experimental machine learning method isn't so bad for a method developed with little resources! Their gamma spectroscopy-based approach predicted the correct reactor type and burnup. Most significantly, though, their method did not predict the ^{235}U enrichment well, and this is what led to the false blame on the DPRT. (The time since irradiation was also 6 months too long, but didn't heavily impact the false attribution like the enrichment did.) The scientist's mass spectrometry-based approach was clearly more accurate for all four parameters. The reactor type, burnup, and enrichment were correctly predicted. Although the time since irradiation was off by 2-3 months, this error didn't result in any false blame being allocated.

In both versions, luckily, the bomb didn't detonate and no one died. It was too rushed of a job, and with the nuclear test ban treaty, no one actually knows their nuclear weapons WILL work. You didn't think our friend's tale was going that dark, did you?

> "Hey," the scientist said to our friend, "I'm famished." Our friend said, "Oh goodness, me too!" They looked at each other, and after some telepathic decision making, agreed on a nice steak.



The end. Illustration by Anna Stephenson

Discussion & Conclusions

This machine learning-based research protocol is designed to answer the question: *How does the ability to determine forensic-relevant spent nuclear fuel attributes using machine-learning techniques degrade as less information is available?* The dissertation written after this chapter answers that in much more detail than the two scenarios presented here, but I hope to have communicated the basics of what I'm doing to a general audience. I actually don't use any real-measured samples by which to compare the different types of training sets (the samples in this story are a part of my work, but simulated), but I do use a real-world set of test cases with the 29 nuclide mass training set. There are many challenges with doing this yet to be resolved, so it is not presented here in a research snapshot.

The lack of a feel-good resolution in this tale is not meant to reduce confidence in our national nuclear forensics capability or my research project, but rather to show how science does not necessarily result in clear-cut answers to questions. Much of the time, asking a question and answering it using the scientific method creates more questions than answers. For example, there are questions about why the gamma spectra approach gave such a wrong enrichment prediction (something echoed in my results, which are aggregate statistics of $450,000$ cases versus the one case presented here). Another question might be whether a 3-month or 6-month time since irradiation prediction error is too large of an error, or an acceptable error.

Author Commentary

Last, I wanted to make a short statement about my work on an even broader scale.

Every scientist should take note of the ethical and political implications of their work. Yes, I said it: science is political[9]! Although the morality of preventing or mitigating a nuclear disaster is not necessarily in question, the nuclear field (both commercial power and defense/the nuclear weapon program) is far from blameless when it comes to destroying human bodies and the environment. The mining industry caused uranium contamination and early death for many Diné in Navajo Nation and payouts/cleanup only began recently[10], plutonium production during World War II has resulted in the displacement and illness of US citizens around Hanford, WA, and government-sanctioned human radiation experiments were conducted on unwitting people and children. None of this (and so much more that's not mentioned here) comes as a surprise knowing the entire nuclear enterprise sits on a foundation of well-documented racism[11]. Last, the obvious must be stated: the US is the only country to ever deploy a nuclear weapon against another country, where the human toll was undeniably brutal[12]. This and much more is documented in a list of resources curated by Kalin Kiesling with help from the

[9] Everyone knows it!
[10] For more information about the 500 abandoned mines and the cleanup efforts, read more here.
[11] Here's a good article on nuclear's racist roots.
[12] And Black American journalists exposed the government's lies about it

nuclear community[13].

None of this means that nuclear power as an energy source is inherently evil, but the industry and our government must acknowledge and take responsibility for abusing both the land and the beings on it. It is hard to hold this knowledge and still want to participate in nuclear science, but if more people in nuclear science and industry also hold this knowledge, then maybe taking life and land for granted can be less of a norm. Of course, better policy is always a stronger influence. Why am I saying all of this inside a(n unofficial) chapter in a nuclear engineering dissertation? Science is political, and so I must be too.

[13]The resources are curated in this Google document.

2 INTRODUCTION

The realm of nuclear security involves parallel efforts in nonproliferation (verification of treaty compliance, monitoring for smuggling, proper storage and transportation of nuclear materials), cyber security, minimizing stocks of weaponizable materials, disaster response training, and nuclear forensics. Nuclear forensics is the process by which nuclear material out of regulatory control is analyzed to provide attribution. All of these efforts have been continually improving, but there was an unaddressed gap regarding the ability of the United States (US) to coordinate and respond to a nuclear incident, especially with the technical portion of nuclear forensics: characterization and analysis. After all, the first textbook on the topic was published in 2005 [1]. In 2006, the US Department of Homeland Security (DHS) founded the National Technical Nuclear Forensics Center (NTNFC) within the Domestic Nuclear Detection Office (DNDO), with the mission to establish a robust nuclear forensics capability by attributing radioactive materials with demonstrable proof. This endeavor was and is highly dependent on inter-agency cooperation. In 2017 - 2018, the DNDO was absorbed into the newer Countering Weapons of Mass Destruction (CWMD) office, and in 2019, much of the nuclear forensics research operation was moved to multiple offices within the Department of Energy (DOE) National Nuclear Security Administration (NNSA).

There is much overlap between nuclear nonproliferation and safeguards and nuclear forensics, although they are distinct nuclear security disciplines. Both fields are concerned with radiological and isotopic measurements of nuclear materials. So, areas like material collection techniques and detector technology are broadly beneficial to both fields. However, much of the time, safeguards must probe a nuclear material or process in a non-destructive manner or in a way that does not reveal proprietary technology; hence, safeguards has limitations that nuclear forensics does not. Therefore, radiochemical separations (a destructive process for material characterization) and identifying forensic

signatures are areas important to nuclear forensics. These approaches within nuclear forensics can vary based on whether the material being collected is pre-detonation (e.g., spent nuclear fuel (SNF)) or post-detonation (e.g., bomb debris).

As mentioned, nuclear forensics is deployed as a response to a nuclear incident. Given that the incident could vary from a weapon detonation to an unknown type of nuclear material being found outside of regulatory control, there are many approaches that a nuclear forensics investigation could take. The incident type that this work addresses is that of interception of intact SNF, perhaps being transported or smuggled into a country. The design of the study reflects this scenario, where there is a small amount of spent fuel from a common commercial reactor. This could be on its way to being reprocessed to extract the plutonium (in certain cases where there is enough fissile material), or it could be intended for a "dirty bomb", or radiological dispersal device.

This project uses statistical methods to model the production history of a nuclear material using measurements of nuclides present in SNF. The SNF is identified in this work by focusing on four characteristics in particular that can potentially provide material attribution: reactor type, burnup, initial uranium-235 (^{235}U) enrichment, and time since fuel irradiation.

1. The **reactor type** is classified as one of the three most common types of commercial power reactors: pressurized water reactor (PWR), boiling water reactor (BWR), or pressurized heavy water reactor (PHWR).

2. The **burnup** describes how much energy was produced by the fuel, taking on the units megawatt-days (or gigawatt-days) per metric ton of heavy metal (or initial uranium); it is referred to in this work as MWd/MTU, and sometimes GWd/MTU.

3. The **enrichment** is the percentage of ^{235}U with respect to the entire amount of uranium in the fuel, and is reported as $\%\ ^{235}$U.

4. Lastly, the **time since irradiation** is defined as how long the fuel has been out of the reactor core, and is measured usually in *days*, but sometimes *years*. It may also be referred to as cooling time.

While the methodology in this exploratory work is limited to the scenario where the SNF in question is from three chosen types of common commercial power reactors (PWRs, BWRs, PHWRs), it could be expanded or applied to other scenarios. In the pre-detonation realm with SNF, there are several commercial reactor types not considered, there are small modular reactors in development that would be of interest, and there is also the exclusion of research reactors. Moving beyond SNF, other nuclear materials are sometimes of interest, e.g., uranium ore concentrate (UOC) or highly enriched non-irradiated uranium. This approach could also be applied to post-detonation as well, perhaps using measured swipe samples from a weapons detonation to answer questions about the where the nuclear material in the weapon may have come from.

2.1 Motivation

Preventing nuclear terrorism is important work that has relevance along the entire lifetime of nuclear material, from the beginning (e.g., enrichment facility inspections) to the end (e.g., SNF management). The International Atomic Energy Agency (IAEA) tracks reports of radioactive material out of regulatory control via the Incident and Trafficking Database (ITDB) [2], and the incidents that are confirmed or likely to be related to trafficking or malicious use are summarized in Figure 2.1. These are the circumstances in which a robust nuclear forensics capability is beneficial, since a combination of nuclear forensics and intelligence is required to investigate such incidents.

Nuclear forensics is an important aspect of deterring nuclear terrorism even though it is not, at first glance, obvious preventative nuclear security. The most common defense

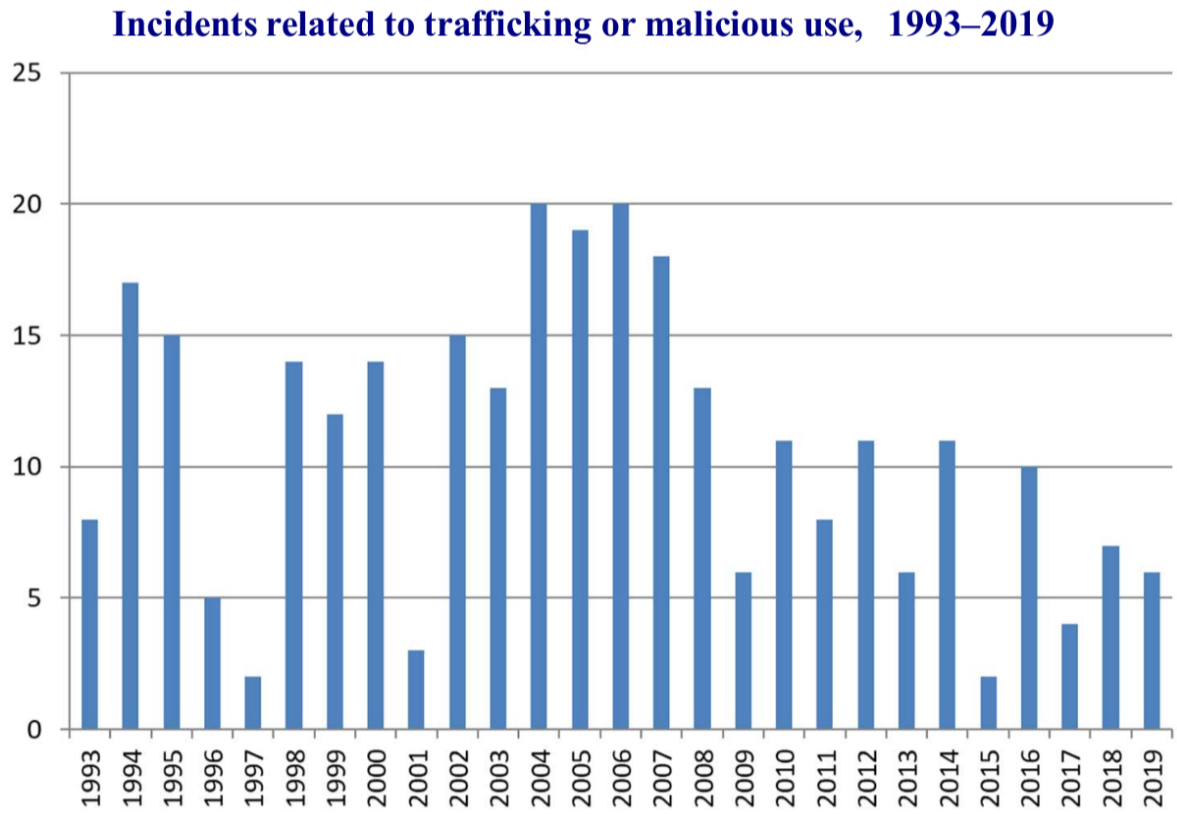


Figure 2.1: Incidents reported to the IAEA ITDB related to malicious use. Included are highly enriched uranium (12), plutonium (2), plutonium-beryllium neutron sources (5) [2]

of the field is that nuclear forensics deters state actors, not terrorist organizations. While it is true that a strong capability encourages governments to be more active in prevention of nuclear terrorism, it can also deter the terrorist organizations as well by increasing their chances of failure. Small destructive successes tend to be more valued than high-risk mass destruction. Nuclear forensics can also assist in cutting off certain suppliers of nuclear materials or technologies (e.g., nuclear specialists that are only involved for financial reasons, access to state suppliers), building a concrete barrier to nuclear terrorism. Therefore, nuclear forensics is considered to impede nuclear terrorism in both tangible and abstract ways [3].

Following the prevention value of nuclear forensics, it is important to understand

the process of the technical portion of the investigation and how that can be improved. In the event of a nuclear incident, such as the retrieval of stolen special nuclear material (SNM) or the detonation of a dirty bomb, it is necessary to learn as much as possible about the source of the materials in a timely manner. In the case of non-detonated SNM, knowing the processes that produced it is crucial to determine the chain of custody of the interdicted material. Section 2.1.1 covers the specific needs of the nuclear forensics community for SNF provenance, and Section 2.1.2 discusses how computational approaches are useful, with a focus on why statistical methods in particular are being pursued.

2.1.1 Needs in Nuclear Forensics

The process of technical nuclear forensics includes the analysis and interpretation of nuclear material to determine its history, whether that be intercepted SNF, UOC, or the debris from an exploded nuclear device. After the technical portion is complete, intelligence data is used to aid in material attribution; this is the overall goal of nuclear forensics.

After a nuclear incident, the material or debris is sampled and evaluated through many techniques that provide the following information: material structure, chemical and elemental compositions, and radioisotopic compositions and/or ratios. These measurements or ratios comprise the forensic signatures of the sample in question. These signatures can be analyzed with specific domain knowledge; for example, UOC will have trace elements depending on where it was mined from. They can also be analyzed against a materials reference database in the case of SNF.

Measurement needs and techniques vary vastly depending on the material, as does the type of signature. This study focuses on non-detonated materials, specifically, SNF. Tracing the source to an entity or state depends on determining if some intercepted

SNF is from an undisclosed reactor or a commercial fuel cycle. This can be done by following the steps shown in Figure 2.2. First, select chemical and elemental signatures and isotopic ratios are obtained, and these measurements are compared to those in an existing forensics database of reference SNF. The signatures for SNF correlate to characteristics that can, in a best case scenario with a reactor history database, point to the exact reactor from which the fuel was intercepted. The reactor parameters of interest are the reactor type, fuel type, enrichment at beginning of irradiation, cooling time, and burnup [4, 5, 6].

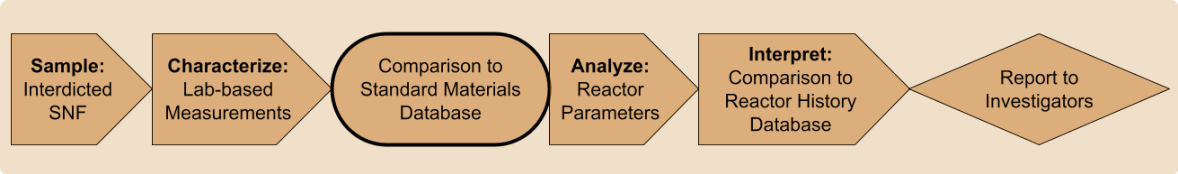


Figure 2.2: An example workflow for a nuclear forensics investigation.

The current and future work of this study are designed based on two primary needs to bolster the US nuclear forensics capability: post-incident rapid characterization, and forensics database challenges and imperfection.

First, our best measurement techniques cannot be available in an emergency scenario, and fast measurements typically yield inaccurate results. Currently, both radiological measurements and mass spectrometry are used in nuclear forensics exercises. Because these techniques have a multitude of variants within each category, there are differing levels of measurement uncertainties. A lofty goal would be to develop methods that provide instantaneous information, reliable enough to guide an investigation (e.g., within 24 hours). In the case of SNF, it takes weeks in a lab to measure isotopes via advanced (cooled detector) gamma spectroscopy and mass spectrometry equipment. A handheld detector that measures gamma spectra could provide the fast measurements to calculate isotopic ratios for the above-mentioned fuel parameters of interest. However, while this nondestructive analysis is rapid, it is also difficult to evaluate because of the

presence of overlapping peaks and the fact that uncertainties differ significantly because of the detector response, environment, storage, electronics, etc. Broadly speaking, there is a time/cost and information tradeoff. On one hand, gamma spectra give less information at a higher uncertainty than the near-perfect results of the destructive mass spectrometry techniques used for characterization, such as inductively coupled plasma mass spectrometry [7]. On the other hand, gamma detection is both faster and cheaper than mass spectrometry techniques.

Second, forensics databases present challenges; this is three-fold. Because these databases are kept by individual countries and the reactor operation history information is well-guarded, it may be difficult to study SNM from a country that has a different fuel cycle. It is proposed that using simulated SNF may be able to combat this issue. Next, because of the values needed for material provenance, forensics databases are highly multidimensional. Additionally, because of the number of measurement types, the forensics databases have inconsistent uncertainties or missing data entries [8]. Thus, direct comparison between measurement results and a database therefore may not yield accurate parameter predictions. Using statistical models may be able to overcome these two challenges, and is introduced next in Section 2.1.2.

2.1.2 Contribution of Statistical Methods

As previously mentioned, there are two main issues that are being addressed for forensics of SNF: database issues and speed of characterization. Many have begun considering computational techniques developed by nuclear engineers to calculate the parameters relevant to nuclear forensics analysis. One example is the INverse DEPletion THeory (INDEPTH) tool [4, 5, 6, 9, 10, 11]. INDEPTH uses an iterative optimization method involving many forward simulations to obtain reactor parameters of interest given some initial guesses.

Another approach utilizes statistical methods to solve nuclear forensics problems, such as implementing searching algorithms for the database comparison step [12] or machine learning (ML) algorithms for determining reactor parameters from SNF characteristics [13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25]. A variety of ML tools have been used to characterize SNF by predicting *categories* (e.g., reactor type, fuel type) as well as predicting *values* (e.g., burnup, initial ^{235}U enrichment, time since fuel irradiation). The former uses classification algorithms and the latter uses regression algorithms, many of which can be altered to perform both classification and regression.

Statistical methods have the uniqueness of requiring minimal domain knowledge via methods or ML algorithms that predict the characteristics or values of interest [13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25]. In the case of most ML algorithms, they first create a black-box (meaning, unable to be human-interpreted) statistical model using the entries of measurements in a database. From that, they can predict the reactor parameters of an unknown sample based on that model. Having an ML model based on a large number of simulations may also overcome the challenges of missing data, irregular uncertainty, or lack of information on other fuel cycles. This logic also follows for other computational methods using a large number of simulations. Although they may not involve a reusable model, they also could overcome missing data, irregular uncertainties, or ignorance of different or non-commercial fuel cycles. Also, statistical methods can be employed to reduce the dimensions in the forensics databases (i.e., reducing the number of forensics signatures required to be measured), which is another valuable characteristic.

2.2 Methodology

The main goal of the technical portion of a forensics investigation is to provide a means of nuclear material attribution by taking and analyzing measurements of the unknown

material. The methodology of this work is based on this process. The top panel in Figure 2.3 shows an example technical nuclear forensics workflow as it could occur in the real world for a pre-detonation scenario. After a sample is obtained, characterization begins. With a focus on SNF, elemental, chemical, and radiological measurements are taken. They are compared to databases filled with previously measured standard materials with known reactor parameters, and/or the reactor parameters are calculated from empirical relationships. These steps might be performed iteratively in a real investigation, first using non-destructive measurements (e.g., gamma spectroscopy), and then destructive measurements (e.g., mass spectrometry). The reactor parameters could then allow the lookup of reactor history information, if available, and these results would be provided to investigators.

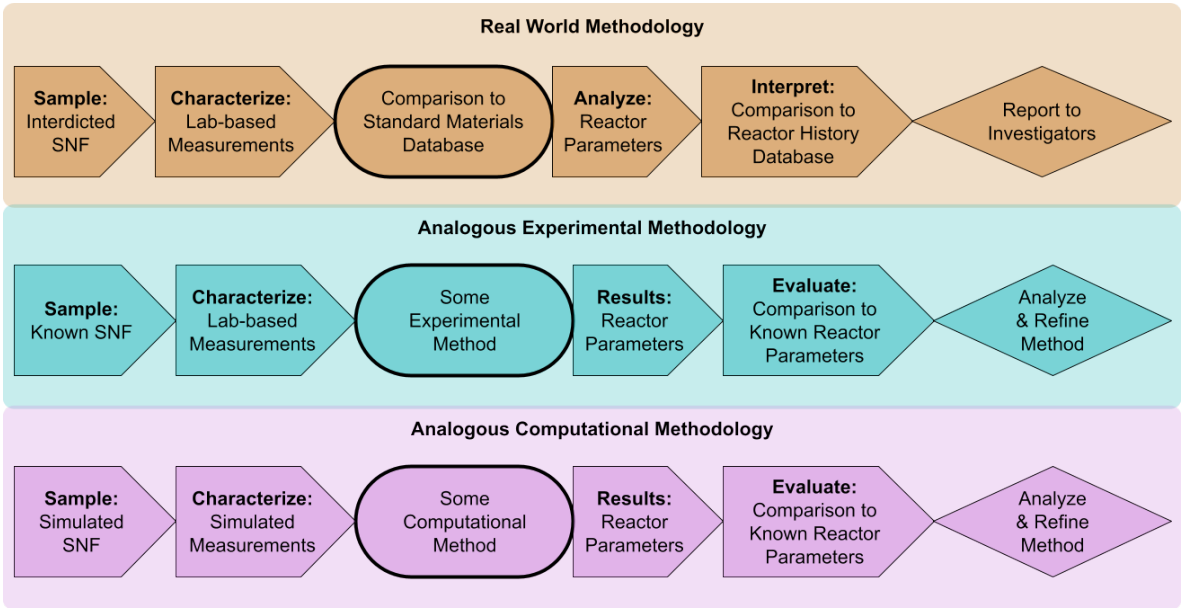


Figure 2.3: Schematic comparing nuclear forensics investigative workflow in the top panel with research approaches for physical and computational experiments.

Next, the investigation process can be translated to an experimental workflow. The middle and bottom panels in Figure 2.3 are analogous physical and computational experiments, respectively. Both of these experimental scenarios would have validated measurements of SNF; the middle panel shows this being done in the laboratory and the

bottom panel shows that these are values from a simulation. The goal of an experimental laboratory study is to test or develop empirical relationships between forensics signatures and the desired reactor parameters. The goal of computational studies can be this, finding new empirical relationships, or performing forensics workflows prior to the implementation of new reactor technologies. For studying alternative measurement techniques or a slight difference in the overall approach, a researcher would iterate through multiple studies using known materials to probe sensitivities or other weaknesses in the procedure.

As mentioned, these processes have informed the experimental design of this work, which follows the bottom panel in Figure 2.3. A test sample of simulated SNF will undergo a measurement that is computed using techniques that mimic destructive or non-destructive measurements of nuclides present in the sample. Using a statistical model, the measurements are compared to a database of SNF entries and their reactor parameters, finding a closest match or prediction for the test sample. Since the test sample has previously known parameters, the error in the prediction can be measured and the method can be tuned.

In addition to the steps of an investigation informing the experimental protocol, there are other considerations to take into account. In the simulation and statistical learning paradigm, it is important to determine how much information to what quality is needed to train an ML model. Because creating databases from real measurements to represent SNF from reactor technologies from around the world is not within the scope of this project, the database in this study will be created from simulations via the Standardized Computer Analyses for Licensing Evaluation (SCALE) [26] system using Oak Ridge Isotope GENeration (ORIGEN) [27, 28]. This is the first process shown in Figure 2.4, referred to as training data because this is what input information is called in ML. It is covered in Section 4.1.

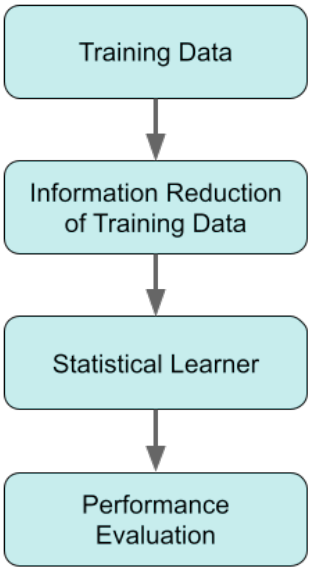


Figure 2.4: Flowchart of the steps the experimental methodology for this work.

The second process shown in Figure 2.4 is information reduction, and is detailed in Sections 4.2 and 5.2. This refers to measurement error and/or the measurement type that provides the nuclide information, and allows the extension of this workflow to better mimic constraints seen in a real-world setting. For example, the primary example here is the reduction of information quality via gamma ray detector, which provide less exact nuclide information than, e.g., mass spectrometry. The radionuclide concentrations from the simulations can be converted into gamma energies, which then undergo a detector response calculation to represent real as-measured gamma spectra as closely as possible. If an algorithm could overcome the limitations of gamma detection and still provide useful results, this would warrant further studies and perhaps be field-applicable.

The third step is to choose a method that performs model creation via statistical learning. Statistical learners have varied strengths and weaknesses based on what is being predicted and how they implement optimization. Introduced in Section 3.2.1 and implemented in Section 4.3 are three methods: k-nearest neighbors, decision trees, and maximum log-likelihood (MLL) calculations. The first two are implemented via

a python ML library, scikit-learn [29]. The MLL method was originally developed for similar attribution work [23, 24, 25] and was implemented in python.

After the training is complete, the results of each models' predictions must be evaluated according to their prediction performance, denoted as the fourth process in Figure 2.4. The machine-learned model predicts the parameters of a previously unseen test set. The difference between the model predictions and the actual simulated parameters is known as the testing error or prediction error. This is outlined in Section 3.2.2.1, and the results are in Sections 4.4 and 5.4.

Thus, ultimately, this research protocol is designed to answer the question *How does the ability to determine forensic-relevant spent nuclear fuel attributes using machine-learning techniques degrade as less information is available?*.

2.3 Goals

The main purpose of this work is to evaluate the utility of statistical methods as an approach to determine nuclear forensics-relevant quantities as less information is available. ML algorithms (k-nearest neighbors, decision trees, and MLL calculations) are used to train models to provide these categories (reactor type) and values (burnup, enrichment, and time since irradiation) from the available information. The training data is simulated, which provides an array of nuclide measurements as the features ($\vec{X}$). The prediction parameters of interest ($\vec{y}$) are provided from the simulation inputs. Information reduction on $\vec{X}$ is carried out using artificially injected random error or computationally generated gamma spectra. The prediction errors ($\vec{y}_{true}$ versus $\vec{y}_{pred}$) will be studied to draw conclusions about the capability of statistical methods to inform nuclear material attribution with less precise detection techniques.

The necessary background is covered in Chapter 3. First, an introduction to the broader field of nuclear forensics is in Section 3.1 to place this work in the context of the

technical mission areas. After that, a short discussion of the field of ML, the algorithms used, and model performance considerations are in Section 3.2. Lastly, a review of statistical methods being used in studies of forensics analysis is covered in Section 3.3.

After the existing work is discussed, the first experimental procedure is outlined in Chapter 4: Reactor Parameter Prediction Using Nuclide Masses. This experiment is done as a demonstration of the methodology with the "perfect knowledge" of nuclide masses. These measurements also undergo information reduction by way of randomly applied uniform error. The performance of test cases drawn from the training data is presented and discussed, as well as the performance of real external test cases of nuclide concentration measurements. The methodology and implementation of the experimental components are introduced in four subsections: the simulated training data is in Section 4.1, the information reduction is in Section 4.2, the details for training models are in Section 4.3, and the performance evaluation is in Section 4.4. Within the performance evaluation, the random error injection results are in Section 4.4.1 and the performance using a real world set of test cases in Section 4.4.2.

Next, the second experimental procedure is explored in Chapter 5: Reactor Parameter Prediction Using Processed Gamma Spectra. This experiment's purpose is to probe the usefulness of field-deployable detectors for giving rapid information about presumed SNF. The information reduction is achieved by using computational gamma spectra of various detectors with decreasing detector energy resolution. The performance of the prediction of reactor parameters is measured by using test cases drawn from the training set, where there is a training set for each detector in this study. The methodology follows the same workflow as the first experiment, but updates to the components will be covered: the new training set features are in Section 5.1, the information reduction from full nuclide knowledge to processed gamma spectra is in Section 5.2, the changes to the training models are in Section 5.3, and the performance evaluation is in Section 5.4.

Lastly, the concluding remarks are discussed in Chapter 6. After a summary in Section 6.1, the main conclusions drawn from the results in Sections 4.4 and 5.4 are covered in Section 6.2. There are also many avenues that this work does not explore and several ways to extend this work; these are outlined in Section 6.3.

3 BACKGROUND AND LITERATURE REVIEW

This chapter provides background and a literature review of the necessary components for this project. Section 3.1 outlines the broader field of technical nuclear forensics, with a focus on the area that motivates this project. Section 3.2 introduces the field of machine learning for an uninitiated audience, covers the relevant algorithms, and presents the methods field practitioners use for validation. Finally, the marriage of Sections 3.1 and 3.2 is presented in Section 3.3, which is a review of previous work applying statistical methods to the nuclear forensics analysis of pre-detonated nuclear materials.

3.1 Nuclear Forensics

Nuclear forensics comprises a large part of an investigation into a nuclear incident, such as interdicted nuclear material or the detonation of a weapon containing radioactive components. The forensics portion of the investigation encompasses both the analysis of nuclear material and/or related paraphernalia as well as the interpretation of these results to establish nuclear material provenance. The former has many technical aspects, relying on a range of nuclear science and chemistry. The latter involves intelligence and political considerations of the material analyses for attribution. This review will only consider the technical portion of the nuclear forensics workflow.

First discussed are the types of forensic investigations in Section 3.1.1, followed by an introduction to inverse problem theory in Section 3.1.2 as a way to frame the forensics problem.

3.1.1 Types of Nuclear Forensics Investigations

The technical programs researching improvements to the US's nuclear forensics capabilities are split between the type of material being investigated. The analysis of irradiated debris from a weapon has different collection and measurement requirements than a mass of special nuclear material (SNM). This separates the field into post-detonation and pre-detonation nuclear forensics. While both are discussed below in Sections 3.1.1.1 and 3.1.1.2, respectively, there is more focus on pre-detonation topics since this work is based on spent nuclear fuel (SNF).

3.1.1.1 Post-Detonation

Post-detonation nuclear forensics requires a diverse set of measurements to obtain the following information: identification of nuclear material, reconstruction of the weapon device design, and reactor parameters for nuclear material provenance. This could apply to an improvised nuclear device or a nuclear bomb. In conjunction with the measurements and characterization are a large array of logistical concerns, including recovery efforts, personnel safety, and material collection cataloging and transportation.

In the case of a full explosion using fissile material, the collection of materials and debris occurs as quickly as possible. It can be in the crater created by the explosion, further away from the center in the fallout, and in the atmosphere above or downwind from the detonation. These are collected by finding glass-like material near the epicenter, debris swipes in the fallout region, and advanced particle collection in the atmosphere via an airplane, respectively. While the epicenter cannot be reached for some time, the debris and atmosphere measurements of radioactive material can provide the yield of the weapon and whether it was made using uranium or plutonium. This along with other physical and chemical measurement allow device reconstruction to begin. Attribution begins to narrow to specific countries or organizations based on this information. [3]

The research needs for post-detonation focus on material collection and analysis as well as nuclear device modeling for reconstruction purposes. Ideally, most material sample collection would be done using automatic instrumentation. Additionally, bolstering the existing device modeling code for reverse engineering is needed. And, as with pre-detonation, a database of standard materials must be both strengthened and centralized. [3]

3.1.1.2 Pre-Detonation

Pre-detonation nuclear forensics investigations occur for every scenario in which non-detonated nuclear material has been found or intercepted. Although this could technically be an intact bomb, it is more likely that SNM intended for a weapon would be the target of an investigation since attempts at materials smuggling are much more common. The range of intact materials for measurement could be as small as a gram-sized plutonium sample or as large as a shipment of uranium ore concentrate (UOC). The goal is to determine the provenance of the SNM, which in the case of SNF is generally done by reconstructing the irradiation process that created the material.

For SNF, where the material was obtained is the first step of the investigation. This would be gleaned from the reactor parameters and storage history (e.g., reactor type, cooling time, burnup), which requires first measuring and calculating certain values: isotopic ratios, concentration of chemical compounds, or existence of trace elements. Both radiological methods (e.g., gamma spectroscopy) and ionization methods (e.g., mass spectrometry) measure these quantities.

Although this is less of a humanitarian emergency than a post-detonation investigation, it is still important to have rapid characterization capabilities via on-site non-destructive analyses. There could be weapons construction already in progress that would need to be curtailed as quickly as possible, or other related malicious intent that a faster investigation could potentially derail. As previously discussed in Section 2.1,

however, the faster measurements result in poor measurement quality. Also, there is a need for research to combat the database issues, as an insufficient forensics database can reduce the accuracy and/or certainty of a reconstructed set of reactor parameters. Another area of research is deeper study of known forensics signatures or discovering new signatures with modeling, simulation, or statistical methods.

3.1.2 Nuclear Forensics as an Inverse Problem

Nuclear forensics is a traditional inverse problem, which has been well documented mathematically and applied to a range of scientific disciplines. Understanding inverse problem theory can help systematically define the limitations of certain solution methods. This section provides an introduction to the topic as well as its application to nuclear forensics.

As outlined in a textbook on the formal approach to inverse problem theory [30], the study of a typical physical system encompasses three areas:

1. *Model parameterization*
2. *Forward problem:* predict measurement values given model parameters
3. *Inverse problem:* predict model parameters given measurement values

First, this shows that it is important to consider the parameters that comprise a model; this is denoted as the *model space*. This is not every measurable quantity; domain knowledge is necessary to determine the model space. In the nuclear forensics context for SNF, this would consist of the reactor operation history parameters. For example, this could be the time since irradiation because the SNF decays and material measurements are different depending on when the measurement is taken.

Second, understanding the physical system also requires an understanding of the forward problem. Predicting how a certain set of values of model parameters will affect the resulting measurements is a problem with a unique solution. The breadth of these

end measurements provides the *data space*, which are all the conceivable results of a given forward problem. So for SNF this would be, e.g., the range of nuclide measurements typical of a commercial reactor.

Lastly, the inverse problem is predicting the model parameters (like time since irradiation) given a solution (like an assay of nuclide measurements). It is statistical in nature; there is a probability that the measured nuclides are caused by some value of a model parameter. Thus, the problem is *ill-posed* because a prediction is not guaranteed to be unique [31].

This can be done by directly inverting the model for a solution, or it can be done from first principles or empirical relationships, which typically requires iterative forward modeling that converges upon a solution. These two approaches (direct inversion and forward modeling) are compared in some nuclear forensics work in Reference [32], where the authors are discussing the discrimination capability of their approaches as it applies to plutonium powder from reprocessing. For forward modeling, they included two different linear models to obtain the parameters of interest: a frequentist linear model and a Bayesian linear model. For direct inversion, they employed principal components regression and partial least squares regression. While the data set in the paper did not provide enough measurements to effectively discriminate, it is interesting to compare both inverse problem approaches in the same work. The authors point out that finding agreement using several methods can be important to proving the predictions are robust.

Another example of iterative forward modeling being used in a nuclear forensics context is in a suite of work using INverse DEPletion THeory (INDEPTH), a tool developed at Oak Ridge National Laboratory [4, 5, 6, 9, 10, 11]. An initial report on the methodology discusses the approach INDEPTH uses [4] and a later report discusses INDEPTH with more detailed examples [11].

Through a nonlinear least-squares regression algorithm, repeated runs of ORIGEN-

Sample No.	Initial Enrichment (%)			Cooling Time (days)			Burnup (GWd/MTU)		
	Declared	Calculated	% Error	Declared	Calculated	% Error	Declared	Calculated	% Error
1	4.013	4.206	+4.80	6515	6,688.5	+2.66	45.9	45.41	−1.06
2	4.013	4.181	+4.18	6515	6,632.1	+1.80	55.0	53.88	−2.03
3	4.013	4.225	+5.27	6515	6,553.7	+0.59	53.5	53.22	−0.53
4	4.013	4.252	+5.95	6515	6,530.4	+0.24	52.7	52.32	−0.72
5	4.013	4.371	+8.93	6515	6,567.5	+0.81	52.4	52.25	−0.28
Average	—	—	+5.83	—	—	+1.14	—	—	−0.92

Table 3.1: Example set of results from INDEPTH solving the inverse problem being described in this work in Reference [11].

Automatic Rapid Processing (ORIGEN-ARP) are carried out given an initial guess. The squared error residual is calculated from comparing the computed nuclide measurements against a set of known nuclide measurements, and the repetition terminates when the sum of the squared error is at some minimum. [4] This approach was also tested with fission product measurements [5] and later with gamma spectra [6]. Table 3.1 shows an example of the results from a set of samples that have the same uranium-235 (^{235}U) enrichment and time since irradiation but different burnups. The average error for the enrichment is 5.83%, but the other two parameters are both predicted within approximately 1% error.

The iterative forward modeling approach has much merit, but another approach is to use statistical methods to determine relationships between nuclide measurements and reactor operation parameters. According to Reference [32] in which forward modeling was compared against direct inversion methods, the statistical approach in this work is also considered direct inversion. The background for this is introduced next in Section 3.2 and is later discussed within the context of nuclear forensics in Section 3.3.

3.1.3 Nuclide Signatures for Nuclear Forensics of Spent Fuel

This work focuses on the nuclides created in the fuel (or pre-existing ones that remain in the fuel) from its time in a nuclear reactor as the forensic signatures of interest. Thus, what follows is a discussion on how some nuclides can provide information about reactor

type, burnup, ^{235}U enrichment, and time since irradiation. Since the reactors considered here are common commercial reactors that use uranium-oxide fuel, this configuration is the focus and not, e.g., mixed-oxide fuel from reprocessing that could be used in a light water reactor (LWR).

The groups of nuclides tracked fall into two categories: actinides and fission products. There are isotopes of both actinides and fission products tracked for SNF attribution in this work. Few actinides are naturally occurring (e.g., thorium and uranium), and the rest are created in the reactor core through neutron captures of other actinides that do not lead to fission. Fission products are the fragments of fissile isotopes (they can also be created by neutron capture of the fragments, or radioactive decay of either of these), which is usually ^{235}U, but could be ^{239}Pu or ^{241}Pu (for uranium-oxide fuel). The non-uranium and non-plutonium actinides are present in much lower quantities than the fission products.

In SNF, the buildup of actinides is dependent on neutron capture (or the lack thereof) of (initially) uranium isotopes and the buildup of fission products is dependent on initially ^{235}U fissions, and later ^{239}Pu fissions. Thus, the neutron energy spectrum in the reactor affects the levels of both actinides and fission products. This allows the distinguishing of different reactor types. Their creation is of course also linked to the initial enrichment and initial amounts of various uranium isotopes via the number of fissions that can occur. The burnup of the fuel also impacts the amount of these nuclides, since more are created as the fuel remains in the reactor core longer. The radionuclides with long half lives also can contribute to the time since irradiation determination. While some nuclides might be a strong contributor to the knowledge of one of the parameters, it is more common that they contribute information about multiple.

The buildup of ^{239}Pu is an example, where it is formed by the neutron capture of ^{238}U. This is dependent on the neutron energy, so gives some indication of reactor type,

but it is also dependent on the amount of ^{238}U in the fuel, which gives some indication of initial ^{235}U enrichment. It also will capture neutrons and make higher mass plutonium isotopes or fission the longer the fuel is in the reactor core. Therefore, it is also linked to the burnup.

As mentioned, many of the nuclides of interest are linked to multiple reactor operation parameters of interest, so it can be difficult to use a small number of these nuclide signatures to effectively determine all four parameters. Reference [31] covers the degeneracy of the solution space of burnup, enrichment, and time since irradiation when only one nuclide is measured. Tested were long-lived nuclides such as ^{137}Cs, ^{154}Eu, ^{244}Cm and short-lived nuclides such as ^{134}Cs, ^{144}Ce, ^{106}Ru. The degeneracy of the solution space decreases as more nuclides are considered.

It can also be useful to use ratios of nuclides to address the issue of decoupling the dependence from one parameter so that the ratio can be used as a signature for another parameter. Reference [23] has a list of 10 ratios used with exactly this purpose. For example, ^{137}Cs could be a useful signature for time since irradiation, but it is also heavily linked to burnup. Since ^{133}Cs is also a burnup indicator but is stable, taking their ratio removes the dependence on burnup and allows that quantity to be used as a time since irradiation signature. Reactor type determination (in this reference, distinguishing LWRs from fast breeder reactors) is similarly done with the ratio ^{150}Sm/^{149}Sm. The fission product ^{149}Sm is a neutron poison, meaning it easily captures many of the thermal neutrons (but not fast neutrons) that are capable of producing fissions in LWRs. Thus, the ratio is dependent on the burnup but with a strong dependency on the neutron energy spectrum, allowing easy distinction between reactor types that depend on fast versus thermal neutrons.

3.2 Machine Learning

Machine learning (ML) is a sub-field of artificial intelligence (AI) within the broad category of computer science. The goal of AI is to create computer systems that respond to their environment according to some set of criteria or goal [33]. For example, self-driving vehicles have computers on board that learn to avoid curbs and humans. While its use has been increasing in the commercial sector, there is also much anecdotal evidence to support the existence of a rapid increase of AI use in academic research across many disciplines beyond robotics. AI systems have been used in detection (e.g., fraud or spam), medical diagnostics, user analysis (e.g., Netflix ratings), and a host of scientific disciplines that have increasing amounts of multivariate data.

ML research focuses on the underlying algorithms using mathematical optimization, methods for pattern recognition, and computational statistics. Much of the recent advances to the field of AI have occurred in the statistical realm, which forgoes domain knowledge in favor of large data sets. In this work, there is no distinction made between the terminology of machine learning and statistics. Additionally, this study is not concerned with computational time, but rather the ability to correctly predict values and categories relevant to the nuclear forensics mission. This restricts the relevancy of the algorithms to the underlying theory and its impact on the resulting model's accuracy.

ML algorithms can be separated into two main categories: unsupervised and supervised learning. The former groups or interprets a set of input data, predicting patterns or structures. The latter includes both the input and output data, enabling the ML model to predict future outputs. Broadly speaking, the unsupervised learning algorithms are designed for clustering data sets or dimensionality reduction (i.e., determining some subset or linear combination of features most relevant to the input data) of data sets. Supervised learning algorithms predict both discrete and continuous values via classi-

fication and regression, respectively. Some algorithms can perform both classification and regression, and neural networks can even be modified to perform either supervised or unsupervised learning.[34]

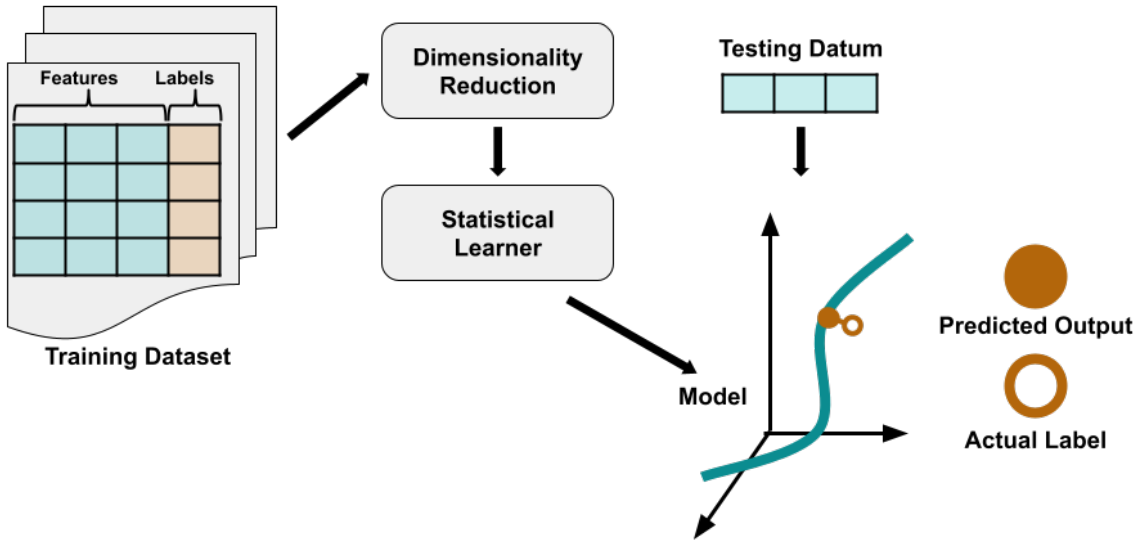


Figure 3.1: Diagram of a supervised machine learning process: a model created from a training data set and a statistical learner (with an optional dimensionality reduction step) allows a prediction when introduced with a test sample. If the actual label is known, an error in model performance can be calculated.

As shown in Figure 3.1, a typical (supervised) machine learning workflow begins with a training data set, which has a number of *instances*, or rows of *samples*, commonly referred to as *entries* later in this work. Each entry has some *attributes*, also referred to as *features*. It also has a *label*, which can be a categorical label or discrete/continuous values.

The training data are then inserted into a statistical learner, where the learner is capable of predicting labels given a set of features. The algorithm for the statistical learner calculates some objective, minimizes or maximizes that objective, and provides some model. This model can be evaluated using a testing set that has the same set of features and labels (but different observations). The comparison of what the model predicts and the actual label gives the *testing error*. Depending on the performance

and application, the model may need improvement from more training and/or some changes in the algorithm parameters. Once the model is performing well enough and validated, it is finalized; then a user can provide a single test sample and a value can be predicted from that.

This study performs regression tasks using supervised learning algorithms. Differences among the structure underlying mathematics of the algorithms impact the ML models. Therefore, the algorithms used in this study will be discussed in Section 3.2.1. Next, a discussion on model performance and prediction errors takes place in Section 3.2.2.

3.2.1 Algorithms for Statistical Learning

For relevant nuclear forensics predictions, both classification and regression algorithms must be used. For example, one may want to predict the reactor type label given some measurement-based features of SNF of an unknown source. This would require a classification algorithm. Or perhaps the input fuel composition is relevant to an investigation on weapons intent, so a regression algorithm would be used.

To limit the scope of this work, it cannot be an exhaustive study of ML methods. Therefore, three algorithms are presented in this section: k-nearest neighbors, decision trees, and maximum log-likelihood (MLL) calculations. They were chosen based on their simplicity; this work has yet to be benchmarked using simple algorithms so a more complex treatment of the training sets in this work would be premature. Additionally, in part because of their simplicity, they are all "white box" methods. This is unique in the ML universe, since most algorithms create a black box model that is unable to be analyzed by a human. The decision trees method provides an output model that can be used to discern behavior and understand predictions, and k-nearest neighbors and MLL calculations do not create a model at all. Individual predictions can still be analyzed, however, since the procedures are so simple.

3.2.1.1 Nearest Neighbors

Nearest neighbors classification and regression are unique algorithms in that they are instance-based; they do not actually generalize, but instead track the observations in the training set. The main metric for this algorithm is distance (or dissimilarity) between the test sample and the closest training sample(s) in the vicinity. During prediction, the algorithm will calculate a value based on the instance that is closest to the current test sample. Thus, there is not any learning, but instead a direct comparison between an unknown sample and the space that the training set populates. The predictions from nearest neighbors can be quite accurate, but are highly unstable to perturbations [34].

The process of prediction with k-nearest neighbors is as follows. First, the distances between the test sample and each of the training set instances are calculated. Most commonly the Euclidean distance is used, but this walk-through uses the Manhattan distance:

$$d_i = \sum_{j=1}^{N_{feats}} |x_{j,train} - x_{j,test}| \tag{3.1}$$

where i is each training set instance, and j refers to each feature in the training set. The lowest k d_i are chosen. For k-nearest neighbors regression, the value, y is predicted using the following equation.

$$y(\boldsymbol{x}) = \frac{1}{\sum_i w_i} \sum_{i=1}^{k} w_i \cdot y_i \tag{3.2}$$

where w_i is either uniform and takes on a value of 1 or is distance-based and takes on a value of $1/d_i$ and $\boldsymbol{x}$ is the full set of features. The regression equation averages the closest k neighbors for an estimate of the unknown sample. In k-nearest neighbors classification, the class label y is predicted using the mode of the nearest neighbors selected using the k smallest d_i, or when w_i is $1/d_i$, the weighted mode is used to choose the predicted label.

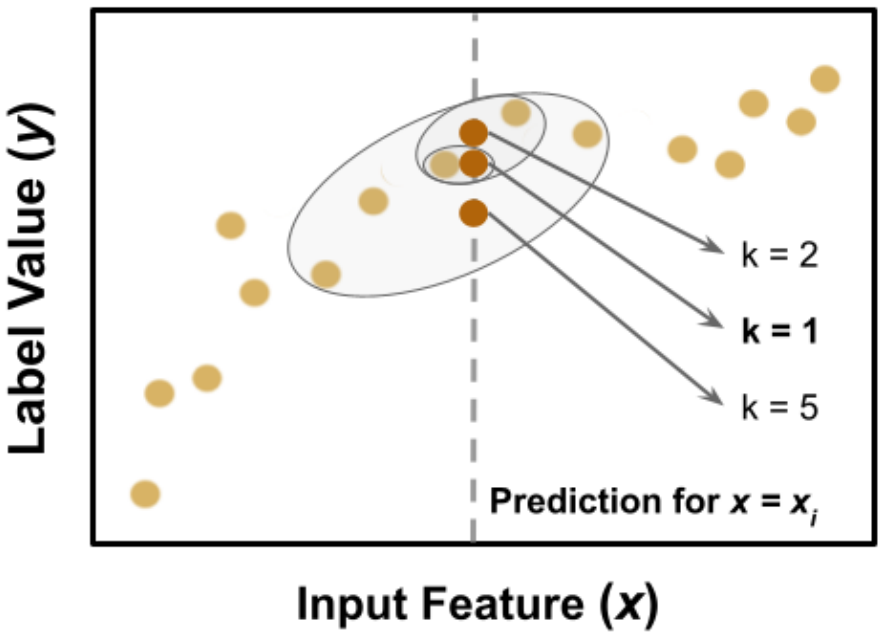


Figure 3.2: Schematic of k-nearest neighbors regression, showing how changing k alters the predicted label value y.

Figure 3.2 provides a pictorial explanation of Equation 3.2 for a prediction where there is one feature. In this figure, there is a test sample with a feature, valued at x_i, indicated with the grey dotted line. The three circles represent the neighborhood given by the value of k, and the darker dots on the line represent the reported prediction y for each choice of k. In this illustration, $k = 1$ or $k = 2$ provide a more accurate prediction according to a visual inspection of the trend, but higher values of k can be useful, and will be discussed in Section 3.2.2.2.

3.2.1.2 Decision Trees

Decision trees are a common choice because they are simple to implement and provide an interpretable model. However, the predictions from decision trees, similar to k-nearest neighbors, are unstable to perturbations. What follows is a highly simplified explanation of the Classification and Regression Trees algorithm for growing decision trees, showing only the equations for splitting criteria. A more complete treatment can be found in

Reference [34] or in the User Guide in Reference [29].

At their core, decision trees algorithms split the feature space into different regions. Decision trees are constructed by iteratively finding places in the feature space at which to split the data to best predict a label. Some measure of information gain (more accurately the opposite, impurity, denoted here as H is used to select a splitting criterion at each split, which maximizes differentiation between average label values in regression or groups similar labels together in classification. This process continues until some externally set stopping requirement is met, or no information gain can be made by continuing to create splits.

Each split creates two new nodes on the tree, where the node has to find a new splitting criterion. In the math that follows, there are nodes given by m, and a number of samples in each node given by $N_{\text{samples},m}$. The individual node samples are counted by i, and so the sample labels are y_i. The impurity at the node is denoted as $H(m)$. In classification, the node impurity $H(m)$ can be measured by the Gini index, where $p_{m,k}$ is the proportion of class k observations at the node:

$$\begin{aligned} p_{m,k} &= \frac{1}{N_{\text{samples},m}} \sum_{i=1}^{N_{\text{samples},m}} \mathbb{I}(y_i = k) \\ H(m) &= \sum_k p_{m,k}(1 - p_{m,k}) \end{aligned} \tag{3.3}$$

Note that $\mathbb{I}()$ here is being used to refer to the indicator function, where when $y = k$ it takes a value of 1 and when $y \neq k$ it takes a value of 0. And in regression, the node impurity $H(m)$ can be measured by the mean squared error, where $\bar{y}_m$ is the average value of the samples in the node.

$$\begin{aligned} \bar{y}_m &= \frac{1}{N_{\text{samples},m}} \sum_{i=1}^{N_{\text{samples},m}} y_{i,m} \\ H(m) &= \frac{1}{N_{\text{samples},m}} \sum_{i=1}^{N_{\text{samples},m}} (y_i - \bar{y}_{i,m})^2 \end{aligned} \tag{3.4}$$

The splitting criterion with the lowest impurity is the one that is chosen to make the split. This will partition the feature space and the splitting process will continue until a pre-defined tree size or number of samples per node. Without a pre-defined stopping point the tree will grow until there is one sample per node. This process can be understood more intuitively by studying Figure 3.3. Note that this tree was created using a maximum tree depth of 3 for visualization purposes in order to explain the process, so is not indicative of a real decision tree.

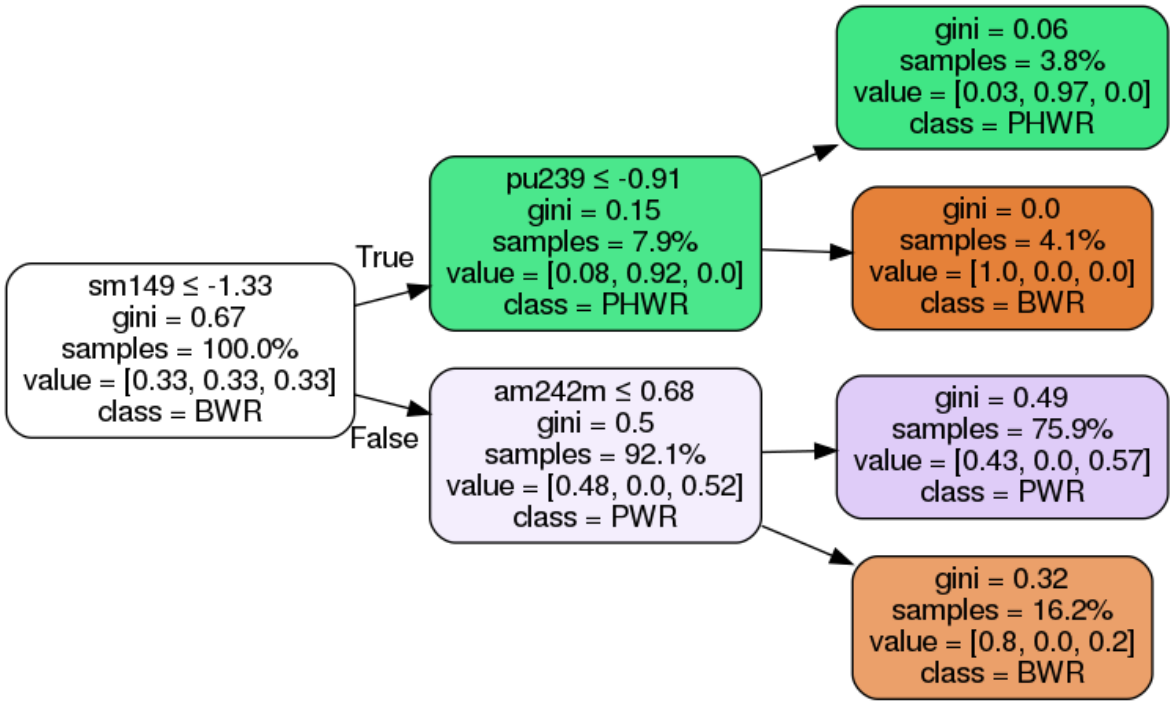


Figure 3.3: Example of a decision tree process, where maximum tree depth is limited to 3.

In Figure 3.3, each node has the following information: the splitting criterion, Gini impurity value, percentage of samples in the training set in the node, a value list (explained below), and the majority class present in the node. In the visualization, the shading of the colors in the tree are bolder for there being a higher fraction of a single class. As a method that produces a model that can be understood by humans, the reasoning for the splits is also discussed.

The first split is determined to occur at the feature ^{149}Sm on whether the nuclide measurement is above or below a value of -1.33. Note that this value is negative because of the scaling process the training set is put through, described in Section 4.3. This is an unsurprising split because the creation of ^{149}Sm is heavily dependent on the neutron energy spectrum and thus helps distinguish reactor types. The majority class at this node is boiling water reactor (BWR) which is expected since the training set is 72% BWR. The values list indicates the fraction of each class in the node, which is alphabetically ordered by `[BWR, PHWR, PWR]`. It is even among the three because the class weights are balanced. This splitting criterion provides a Gini impurity score of 0.67, which represents the minimum Gini impurity of all the candidate splits, but also indicates there are multiple classes represented in this node (again, expected). It would be 0 if there were only one class in the node.

The next level of the decision tree in Figure 3.3 has two nodes. The top node has a splitting criterion of whether ^{239}Pu is above or below a value of -0.91. This is also an unsurprising choice for the decision tree because its creation and destruction is heavily dependent on the enrichment, since pressurized heavy water reactors (PHWRs) have more ^{238}U available for neutron capture. This node contains a majority of the PHWR class, but also a fraction of 0.08 of the BWR class. Even though the PHWR class is only 1.5% of the training set (see Section 4.1), this node has 7.9% of the training set samples. The next level split for the top node has fractions of 0.97 PHWR and 1.0 BWR in the next two nodes, with Gini impurities of 0.06 and 0.0, respectively. This is a rapid approach to zero considering the size of the training set and number of features.

The lower node has a splitting criterion of whether ^{242m}Am is above or below 0.68. This one is less obvious of a choice, but is in the decay chain of ^{241}Pu (if it does not fission), which is created by a series of neutron captures. This links ^{242m}Am to a dependence on the neutron energy spectrum and therefore makes it somewhat capable

of distinguishing reactor type. The remaining 92.1% of the training set is in this node, and there is only a slight majority of the (balanced) pressurized water reactor (PWR) class, which is why the shading is nearly white. The Gini impurity for the top node is 0.15. This is much lower than the Gini impurity of 0.5 for the bottom node, which is more evenly split between two classes. The next level split for the bottom node made modest improvements over the Gini impurity of 0.5, and would need to go much deeper to properly classify PWR versus BWR. This suggests that ^{242m}Am is only a moderate reactor type indicator.

3.2.1.3 Maximum Log-Likelihood Calculations

The MLL calculations approach applied here is based on a method developed to do similar work [23, 24, 25]. That work involved matching nuclear material samples based on some select measurements to entries in a database of containing those measurements (see Section 3.3). Each database entry also has a similar list of labels to the labels being predicted in this work: reactor type, burnup, and time since irradiation.

Interestingly, the MLL calculations method works exactly like k-nearest neighbors with $k = 1$, where there is no model but a prediction according to the closest match database entry. There is one detail that differs, however: the selection criterion. Whereas k-nearest neighbors minimizes distance/dissimilarity, this approach instead maximizes similarity via a likelihood function. An "unknown" test sample is compared against the training set using the likelihood calculation between that sample and the training set entries. The higher the likelihood, the higher the probability that the database entry represents the sample. The likelihood is in Equation 3.5, whereas the log-likelihood is used more often in practice, shown in Equation 3.6 [23].

$$L(M|x_{test}) = \prod_i \frac{1}{\sigma_{i,train}\sqrt{2\pi}} \exp \frac{-(x_{i,test} - x_{i,train})^2}{2\sigma_{i,train}^2} \tag{3.5}$$

$$ln(L(M|x_{test})) = \sum_i ln(\frac{1}{\sigma_{i,train}\sqrt{2\pi}}) - \frac{(x_{i,test} - x_{i,train})^2}{2\sigma_{i,train}^2} \tag{3.6}$$

Although these equations were borrowed directly from Reference [23], some of the variables have been changed for better cohesion with the terminology in this work. The likelihood is a measure of the probability that a model M produced the measurements seen in the test sample, given by $L(M|x_{test})$. In both Equations 3.5 and 3.6, x refers to the set of features, and $x_{i,test}$ and $x_{i,train}$ are the individual features for the test sample and the training set entries, respectively. The uncertainty of the measurement associated with each feature is represented by $\sigma_{i,train}$.

3.2.2 Algorithm Performance

After a model is trained, its performance must be evaluated. The following discusses the considerations taken for the evaluation of the prediction performance for the algorithms used in this work.

ML algorithms are heavily dependent on the training inputs and algorithm parameters given to them, such as training set sizes, regularization (defined below in Section 3.2.2.2), number of features in the training set, algorithm hyperparameters, etc. To obtain reliable models, one must both choose or create a training set carefully and study the impact of various algorithm parameters on the error. Various error metrics are first covered in Section 3.2.2.1 before the causes of error are discussed in Section 3.2.2.2.

3.2.2.1 Testing Error

The creation of an ML model is (usually) a hidden process. Although the model emerges from a black box, there are ways to evaluate its generalization (i.e., prediction) capability. This is done by removing a small portion of the database for use as a testing set. The rest of the data set is known as the training set and is used to train a model. After training, the test set is used to calculate the model's error to unseen test samples. This

error is typically referred to as the *testing error*, as it is measuring the ability of the model to predict future cases that were not introduced in the training phase. Next, the various metrics used to evaluate classification and regression are covered.

Reactor Type Classification

For the classification of reactor type, it is typical to use an accuracy score for classification, where the total number of correct predictions is taken as a fraction of the entire sample set.

$$accuracy = \frac{1}{N_{\text{samples}}} \sum_{i=1}^{N_{\text{samples}}} \mathbb{I}(y_{pred,i} = y_{true,i}) \tag{3.7}$$

Note that $\mathbb{I}(x)$ here is being used to refer to the indicator function, where when x is true it takes a value of 1 and when x is false it takes a value of 0.

But training sets can have an uneven number of classes represented. A more fair scoring system for imbalanced data sets is instead balanced accuracy, which averages the accuracy of each class according to its class frequency. This still provides a range from 0 to 1, but the meaning of the score has changed from the accuracy score. This is done by first defining a sample weight based on its class frequency w_i [29]:

$$w_i = \frac{1}{\sum_j \mathbb{I}(y_j = y_i)} \tag{3.8}$$

for j classes and i samples. Thus, the balanced accuracy is the following.

$$balanced\text{-}accuracy = \frac{1}{\sum_i w_i} \sum_{i=1}^{N_{\text{samples}}} w_i \cdot \mathbb{I}(y_{pred,i} = y_{true,i}) \tag{3.9}$$

In this work, however, the balanced accuracy is used with `adjusted=True` in scikit-learn. After computing the balanced accuracy, this rescales the range to $-\frac{1}{1-N_{\text{classes}}}$ to 1, where 0 is considered random scoring.

In addition to the metrics which combine the knowledge of true positive and true

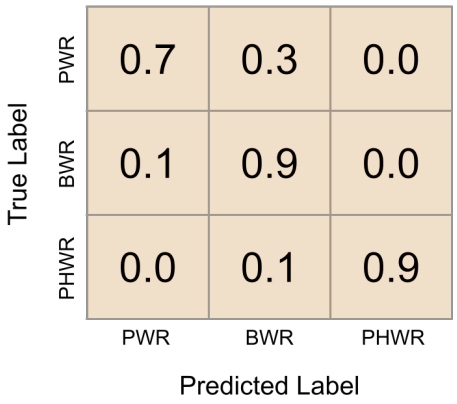


Figure 3.4: Example of a confusion matrix for the three reactor types

negative predictions, it is also important to study the misclassifications. Confusion matrices show both the true positive predictions and the false positive predictions, which can be useful information in understanding an imperfect accuracy or balanced accuracy score. For example, Figure 3.4 pictures an example confusion matrix that could result from this data set, where a fraction of PHWRs are correctly predicted (0.9), but there are some misclassified as BWR (0.1). While there are no PWRs or BWRs misclassified as PHWR, they do get misclassified as each other: 0.3 PWRs and 0.1 BWRs get misclassified as the other. Note that each row must add up to 1, since those are the true class labels. The columns do not generally add up to 1 (unless there are no misclassifications or the confusion matrix is symmetric).

Regression Mean Error Calculations

For the three regression cases, there are a few metrics that are being calculated to measure prediction performance: the mean absolute error (MAE), median absolute error (MedAE), and mean absolute percentage error (MAPE). The most common metrics to use for comparing regression errors are root-mean-squared error (RMSE) or MAE, but these mean errors do not provide the full picture. The MedAE can provide interesting insight into a middle-ground or majority behavior. Also, the relative error via MAPE can provide insight especially when there is a large span of values for a regression case;

a large absolute error could be a small relative error. Therefore, they are all tracked. The MAE, MedAE, and MAPE are calculated as follows, respectively, for each label j (which is suppressed in the math below for clarity).

$$MAE = \frac{1}{N_{\text{samples}}} \sum_{i=1}^{N_{\text{samples}}} |y_{true,i} - y_{pred,i}| \tag{3.10}$$

$$MedAE = \text{median}(|\ y_{true,1} - y_{pred,1}\ |, \ldots, |\ y_{true,n} - y_{pred,n}\ |) \tag{3.11}$$

$$MAPE = \frac{100}{N_{\text{samples}}} \cdot \sum_{i=1}^{N_{\text{samples}}} \frac{|y_{true,i} - y_{pred,i}|}{y_{true,i}} \tag{3.12}$$

Cross Validation

A testing set that would be used during training to give feedback, a *cross-validation (CV) set*, can provide a faster convergence to a satisfactory model. As shown in Figure 3.5, this can be done by splitting the data set into three groups: a large training set, a small CV set, and a small testing set.

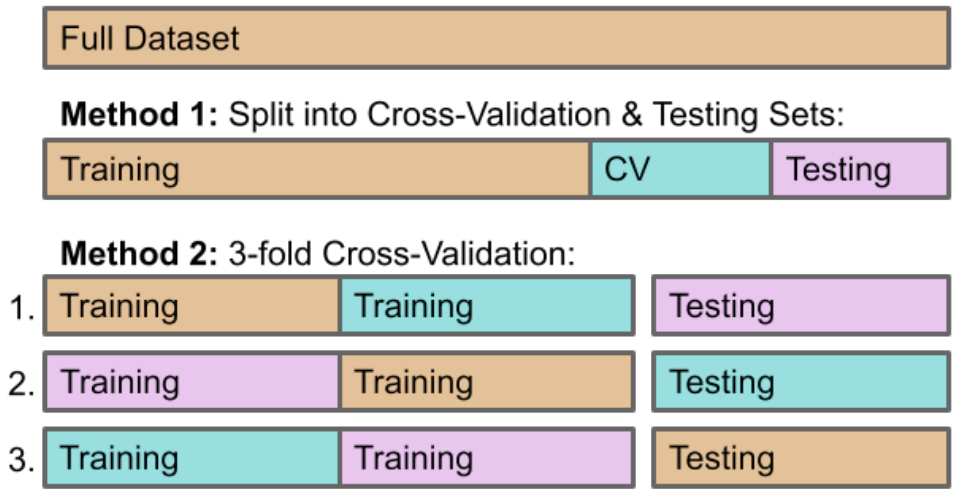


Figure 3.5: Illustration of two ways of performing CV: a one-time split, or k-fold CV.

However, in practice, multiple rounds of CV steps are used provide a better estimate of model performance by increasing the number of testing sets. This is referred to as *k-fold cross-validation*. An example where $k = 3$ is illustrated in Figure 3.5. One

partition of the training set is designated as the testing set, and a model is trained with the rest. This returns the testing error for that first testing partition. Following the first training phase, another begins, this time with a different subset as the testing set. In total, this process is performed three times, giving three models. Since each partition becomes a testing set at one point, all entries in the training set are tested, which reduces the chance that the model is being tested with a misrepresentative testing set. In most applications, the testing error results from each partition (which are an average of all test cases in that partition) are averaged to provide a picture of the model performance. However, this work instead focuses on the aggregate statistics of all the *individual* test case errors taken together, regardless of which partition they were in.

3.2.2.2 Model Complexity

In statistical learning, there are two sources of error that need to be simultaneously minimized: bias and variance. Bias is caused by simplifications in the model, so the error is caused by missed relationships in the data; high bias is an indication of an underfit model. Variance is caused by including random noise in the model, so the error is caused by oversensitivity to that noise; high variance is an indication of an overfit model. What follows is a discussion on error considerations that all reduce to one concept: how *complex should a model be* to best predict a previously unseen test sample?

Figure 3.6 shows the tradeoff between the bias and variance. The shape of the total error curve has a minimum that we seek to achieve with our model. Some bias is desired in order to generalize to future unknown data. But, some variance is positive for the model because it captures the relationships in the data that the bias counteracts.

Regularization refers to introducing a term into the ML model to prevent overfitting; it is used in many ML algorithms to reduce the model complexity and therefore the resulting variance. This would be represented by increasing k in k-nearest neighbors,

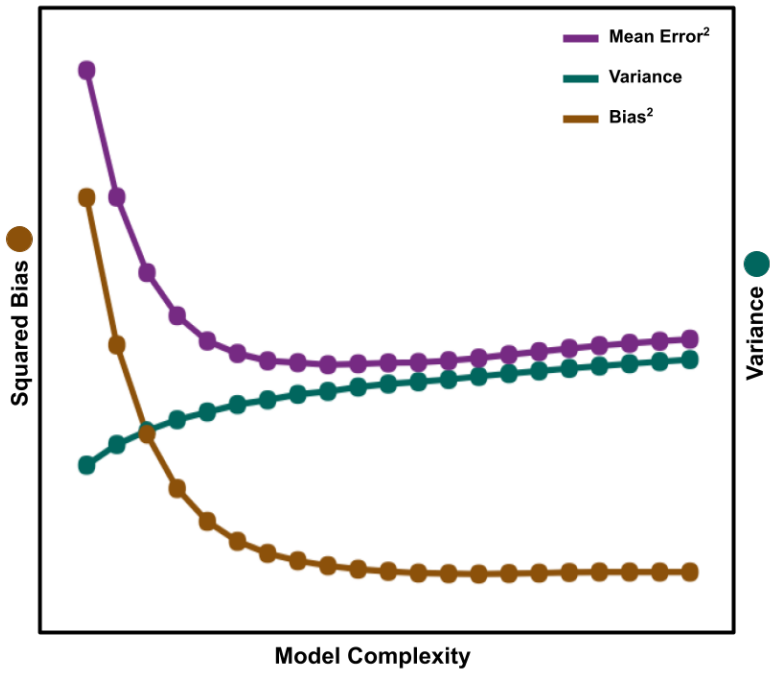


Figure 3.6: Schematic showing the sources of error, bias and variance, and how they behave with respect to model complexity in the bias-variance trade-off.

or reducing the maximum features a decision trees implementation could consider for splitting. The top three windows in Figure 3.7 show the effects of regularization on a simple linear regression model. With heavy regularization comes high bias, represented by the left-most window. The right-most window shows a low regularization scenario, where most individual points are tracked by the model, but generalizing beyond that might be problematic. The middle plot represents an approximately well-fit model.

Diagnostic plots show the testing errors with respect to some variable on the horizontal axis. Typically this variable is related in some way to the model complexity. This provides insight into the model's fitness, and whether small tweaks can be made to increase bias or variance to improve testing performance. Put another way, these approaches can evaluate under- or over-fitting. When the horizontal axis is related to regularization parameters, often referred to as algorithm hyperparameters in this work, it is known as a *validation curve*. A full treatment of validation curves is not considered

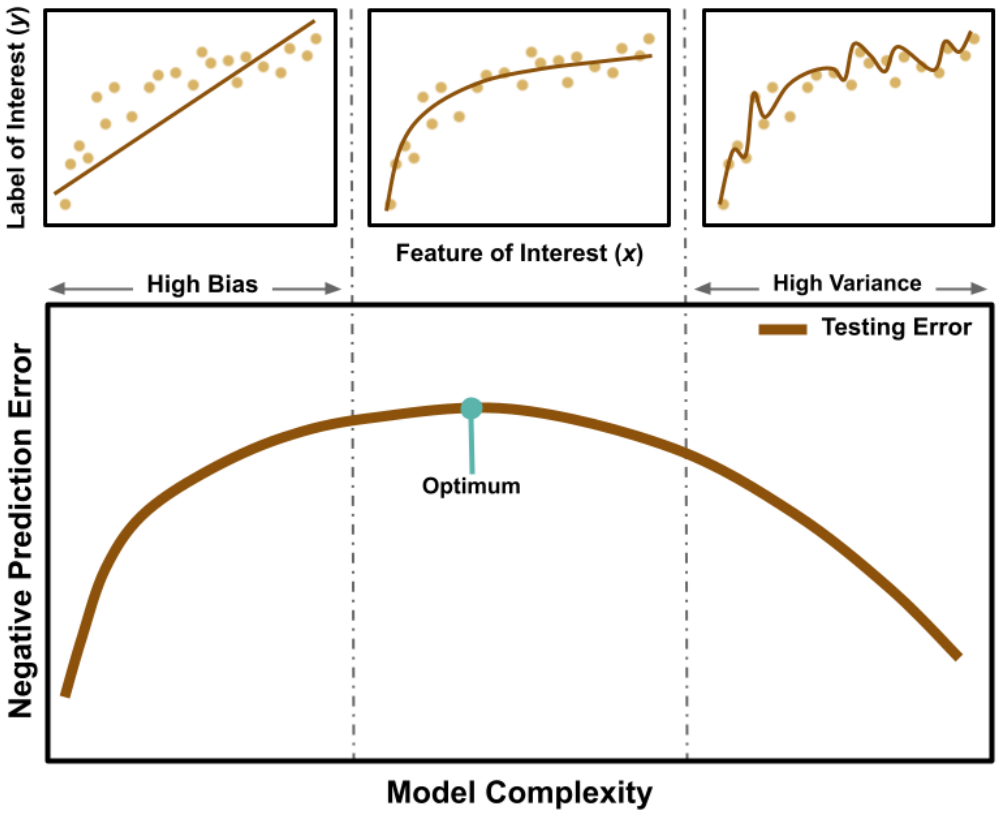


Figure 3.7: Diagram showing effect of model complexity/regularization at three different levels on model performance.

here, but Figure 3.7 shows the portion relevant to the discussion in the bottom plot. The negative prediction error is plotted on the vertical axis so that the orientation of higher is better is maintained. A parameter influencing model complexity is on the horizontal axis. The testing error is typically low for the high bias and high variance models, but there usually exists an optimum parameter for model complexity for most training data sets.

Another parameter indirectly influencing model complexity is the training set size. When the training set size is plotted on the horizontal axis as either the percentage of the training set used or the absolute number of observations, it is called a *learning curve*. These allow for the user to evaluate the optimum number of training set observations to include in the training phase. This is relevant in a scenario like this work where the

training set is large (large being a relative term).

The training set size must be large and diverse enough to be considered independent and identically distributed (i.i.d.) because most ML algorithms are developed upon this assumption. Sometimes this is not possible, and the training data are skewed, i.e., a portion of the data is over-represented. This must be handled explicitly, but since each algorithm handles skewed data differently, it is currently beyond the scope of this work. Instead, attempts were made to best create an i.i.d. training set, which is covered in Section 4.1.2.

Another area worthy of investigation is the other dimension of the training set: the number of features included for model training. This is not usually a value that would be plotted on the horizontal axis of a diagnostic plot (unless one's data is predisposed to this kind of study), but is considered in this work. The feature set selection is discussed in Sections 4.1.3, 5.1, and 5.2.

In practice, plotting learning and validation curves can be iterative. But too many optimizations will result in a poorly performing model when exposed to data outside of the training set, so there is a risk associated with better prediction after using optimization tools. This increase in performance from over-optimization could be linked to the training set performance and might not generalize outside of the specific type of input data used. A workaround for this scenario is to obtain more data for the set or to obtain a completely different data set altogether.

3.3 Applications of Statistical Methods to Nuclear Forensics Analysis

Although the body of literature in the area of proposed research is not expansive, there have been a number of relevant studies. These, in some way, are related to the the

prediction of forensics-applicable categories or quantities of nuclear materials using statistical methods. With regards to broader forensics capabilities, materials from different steps of the nuclear fuel cycle are being studied. Even though each material has its own forensics signatures, the process of applying statistical methods to the analysis of material provenance is similar for each.

For example, on the front end of the fuel cycle, an entity may have obtained UOC if they have enrichment capabilities. One study performed statistical analyses on UOC from 21 sources (throughout seven countries) using 30 concentration measurements of various elements, isotopes, and compounds, e.g., sodium, magnesium, thorium, uranium-234, or halide compounds [19]. The goal of classifying the source and the country was reached 60% and 85% of the time, respectively, with the unique method developed in the paper (an iterative partial least squares discrimination analysis approach), which outperformed decision trees and k-nearest neighbors.

On the back end, an organization might have interest in SNF if they have reprocessing capabilities. Or, perhaps already separated plutonium from SNF has been intercepted and needs to be traced. Another study addresses this by performing factor analysis on theoretical separated plutonium from various sources of Oak Ridge Isotope GENeration (ORIGEN)-simulated SNF based on their composition at the end of irradiation [16]. Since in this study all materials are the same age, five plutonium isotopes ($A = 238-242$) correctly predicted a test sample. However, taking different times since irradiation and reprocessing into account requires more isotopic measurements.

3.3.1 Factor Analysis Work

In addition to the immediately aforementioned work, there is a suite of work on performing classification using factor analysis or isotopic ratios in combination with visual distinction. Although factor analysis explicitly requires the input of domain knowledge,

it is a valuable first step towards understanding how statistical methods can provide insightful models to classify materials. This series of work seeks to accomplish classification of a reactor history (reactor type, enrichment, burnup) using the distinguishability between similar sets of samples, as pictured in Figure 3.8.

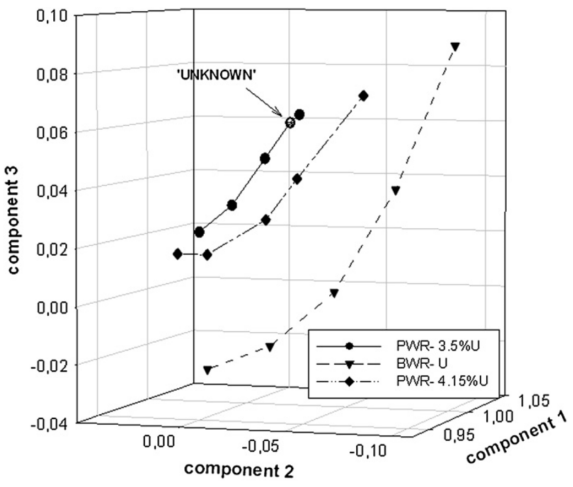


Figure 3.8: Example from Reference [16] on using factor analysis to show similarities between classes of SNF for visual identification.

The chronologically earliest study in this grouping of publications [14] uses ORIGEN-simulated uranium and plutonium (^{234}U, ^{235}U, ^{236}U, ^{238}U, ^{238}Pu, ^{239}Pu, ^{240}Pu, ^{241}Pu and ^{242}Pu) of various SNF designations. These simulated measurements are subject to factor analysis and the results usually plotted. The unknown samples in this work are in visual alignment with the factor-analysis-determined groupings. Reference [17] extends that study (same statistical method, same list of isotopes) to real measured samples from the Spent Fuel isotopic COMPOsition (SFCOMPO) database [35, 36]. The goal here was to determine whether the factor analysis approach could distinguish the chosen SFCOMPO entries well, and the results confirm the goal was achieved. Reference [18] uses three plutonium ratios (^{242}Pu/^{240}Pu, ^{238}Pu/Pu$_{\text{Total}}$, and ^{239}Pu/^{240}Pu) to accomplish the same goal with simulated SNF without factor analysis, and the results are similar.

This work mostly relies on actinides for identifying SNF [14, 16, 17, 18], but the use

of fission products is also promising [15]. This particular publication was interesting because the author chose to use one set of fission products to represent a typical mass spectrometry assay (^{133}Cs, ^{140}Ce, ^{150}Sm, ^{152}Sm, ^{144}Nd, ^{145}Nd, ^{146}Nd, ^{148}Nd, ^{150}Nd), and another set to represent what could be determined from a gamma detector (^{95}Zr, ^{95}Nb, ^{106}Ru, ^{134}Cs, ^{137}Cs, ^{144}Ce). This approach is successful when the simulation uncertainty is below 3% [15].

3.3.2 Other Classification Work

There are other papers on statistical methods that focus on the classification of the reactor type for unknown samples. One study simulated and tracked 34 nuclides of a set of typical commercial nuclear power reactors and their operation parameters, but first used statistical dimensionality reduction (via Laplacian eigenmaps) before subjecting the training data to reactor type classification, comparing linear discriminant analysis, quadratic discriminant analysis, random forests, and Parzen window classifiers [21, 22]. Condensing the 34 features into three has not only computational and potentially discriminatory benefits, there are visualization benefits as shown in Figure 3.9. This plot and this work also highlights and addresses a known problem: reliable discrimination between SNF from PWRs and BWRs.

Another paper compared principal components analysis and partial least squares discriminant analysis to classify reactor type [20]. As with the above, a set of SNF from typical commercial power reactors from around the world were simulated and used as test samples for this attribution step, but unlike the above they chose to focus on uranium and plutonium isotopes to address the challenge of identifying chemically separated uranium or plutonium. This work uses a qualitative visual distinguishability discussion to choose the better method of partial least squares discriminant analysis.

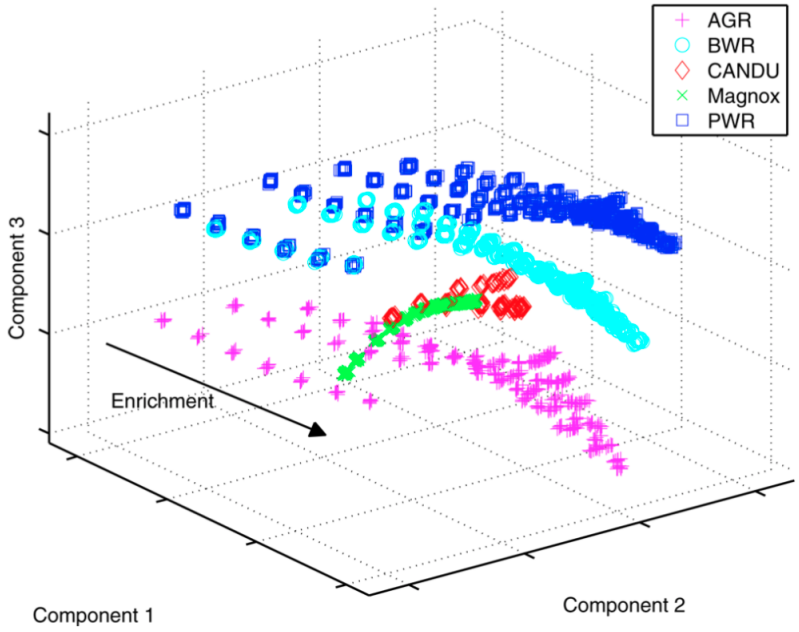


Figure 3.9: Results of leveraging a pre-processing step of dimensionality reduction for visualization purposes, from Reference [22].

3.3.3 Regression Work

Switching the focus to regression, one work combines the use of statistical methods with an investigation of those methods when faced with information reduction via random nuclide measurement errors in the training data set [13]. Additionally, feature reduction was investigated by using various nuclide compositions: the top 200 nuclides by concentration in each vector, fission products only, and a principal components analysis-derived shortened nuclide list.

Using these various feature sets, three methods were also compared. First and second, the nearest neighbor algorithm with two different distance metrics was used: Manhattan distance (L_1 norm, or sum of absolute differences of Cartesian coordinates) and Euclidean distance (L_2 norm, or square root of the sum of squared differences of Cartesian coordinates). The nearest neighbor approaches classified reactor type and predicted burnup. Third, ridge regression with an L_2 norm for regularization was only applied to burnup prediction. In both classification and regression cases, using the

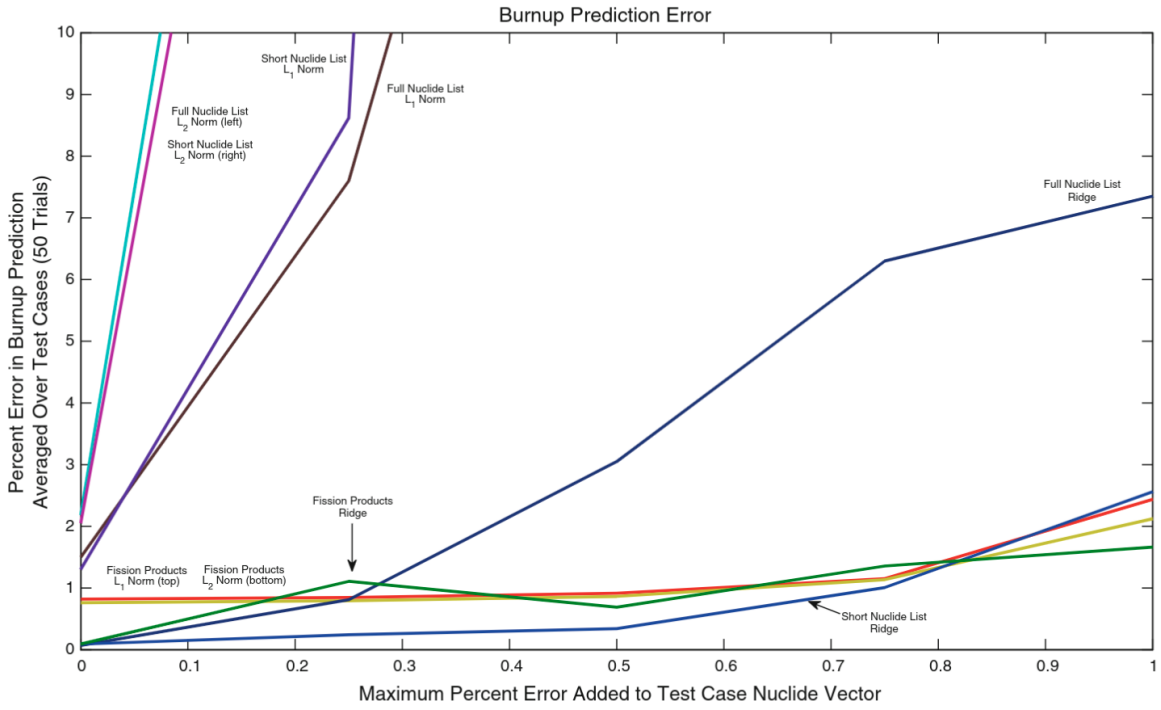


Figure 3.10: Plot from Reference [13] that shows the degradation of burnup prediction performance with respect to increasing training set error for three algorithm implementations and three feature sets.

fission products nuclide list with both nearest neighbor methods performed the best. All other nuclide lists quickly devolved to random guesses with an increase in nuclide error in the case of reactor prediction, and more than 100% error in the case of burnup prediction. Figure 3.10 shows the behavior of burnup prediction at low training set error (under 1%) for the various methods with feature set combinations.

3.3.4 Maximum Log-Likelihood Calculations

Another set of publications focuses on a novel methodology for attributing separated plutonium with different reactor histories. Although many commonly used statistical methods have been previously discussed in this section, this approach is unique: a MLL calculation approach for determining the similarity of a test sample to a database of simulated samples [23].

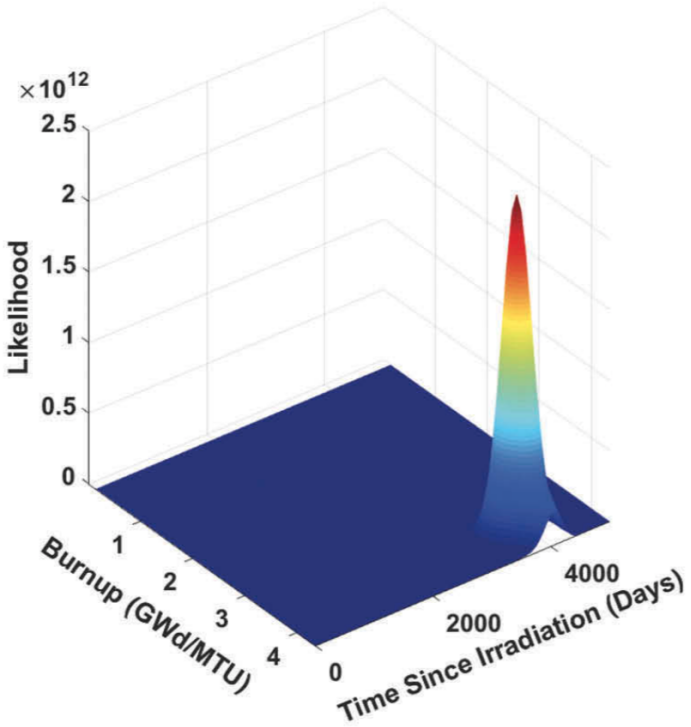


Figure 3.11: Plot from Reference [23] that shows a sample being assigned burnup and time since irradiation values based on a likelihood maximum.

Since this approach was developed to attribute weapons-grade plutonium, the training database is simulated with low burnup values, and there are 5000 one-day time steps. The reactor type, burnup, and time since irradiation comprise the labels, and the features are a set of 10 carefully selected isotope ratios.

Figure 3.11 shows one of the results, where a simulated sample of 4.39 GWd/MTU burnup and 3652 *days* time since irradiation is tested against the training database via likelihood calculations of the feature set of 10 isotope ratios; the maximum is visibly close to the ground truths. This method was later validated with experimental samples, in Reference [25].

After experimental validation, sensitivity studies were conducted in Reference [24]. Figure 3.12 shows a set of results from the increasing of uncertainty for the same sample being predicted in Figure 3.11. It is interesting that the likelihood decrease

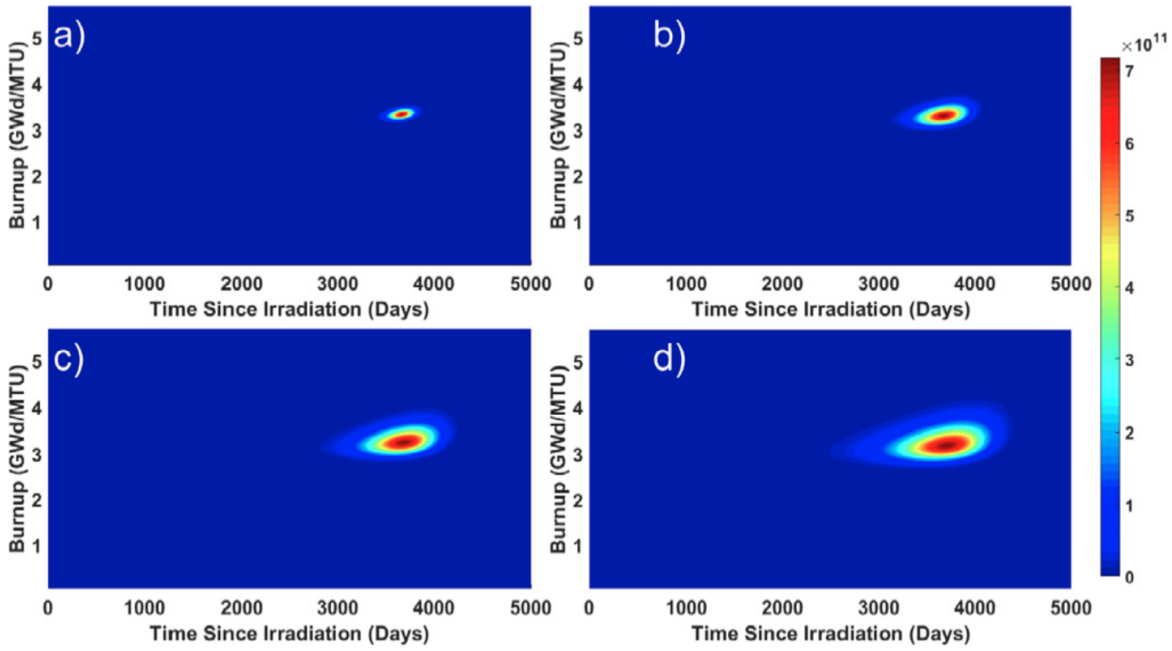


Figure 3.12: Results from a sensitivity study on the MLL method where uncertainty of the training set was increased. The levels shown here are (a) 7%, (b) 14%, (c) 21%, and (d) 28% [24].

happens faster on the time since irradiation axis than on the burnup axis. The main result from the sensitivity studies is that while a different sample was robust to the increased uncertainty and was predicted at the 99% confidence level even at the highest uncertainty (28%), the sample shown here was predicted at only a 68% confidence level at the highest uncertainty.

3.3.5 Summary

The factor analysis works [14, 15, 16, 17, 18] and other visualization- and classification-based works [20, 21, 22] helped provide a foundation by which to understand the scope of problem of attributing SNF. Additionally, some of these use the SFCOMPO database in their study design [17, 21]. While Reference [17] uses the factor analysis method directly on some subset of SFCOMPO database entries, Reference [21] uses nine samples

from the database as test cases for both reactor type classification and burnup/^{235}U enrichment. This work instead tests all available SFCOMPO entries after some filtering steps are carried out to remove cases that are not approximated by the training set. Reference [15] chooses a set of nuclides based on a comparison of mass spectrometry to gamma spectroscopy, which also happens in this work. However, there is no indication that there is a gamma detector-based treatment of the chosen fission products. It is presumed they are using mass- or activity-based values to represent the features in their training set.

Reference [19] performs prediction of UOC provenance, comparing a newly developed method against two simple algorithms; this informed the method of choosing of algorithms in this work. It is important to first create a benchmark a simple approach before scaling up to more complex treatments of the data. The approach in this reference used k-nearest neighbors, decision trees, and a unique iterative method. It is actually a coincidence that the first two methods match the ones chosen for this work, because other methods were tested before deciding to use both k-nearest neighbors and decision trees. Still, this work chooses a different third method to implement and shifts the focus to SNF.

This third method is the MLL calculations developed in References [23, 24, 25]. The mathematical framework for MLL calculations is in Section 3.2.1. There are a few differences in implementation. This work focuses on SNF instead of a material source presumed to be separated plutonium. Additionally, the time steps in the references are much smaller than what exists in the training set in this work, and these publications focus on a set of 10 isotope ratios as the feature set. Also, instead of studying two well-characterized samples (that were at first simulated but then irradiated and real measurements were taken), this work instead is focused on the aggregate statistics of many predictions. Despite these differences, this approach is still applicable to and an

asset for this work.

A paper that greatly influenced the development of this study [13] seeded the decision to evaluate the effect of information reduction on the predictive capabilities of the statistical methods used. While the first experiment in Chapter 4 uses a similar application of random error, the experiment in Chapter 5 instead uses detectors with decreasing energy resolution to accomplish information reduction.

In summary, there is limited treatment of statistical methods predicting any nuclear material's attributes when facing information reduction in the literature. There is also limited use of the SFCOMPO database as a testing set. The first experiment in Chapter 4 addresses these thin areas of previous work. There is no work to the author's knowledge using a gamma detector-based treatment for the measurement of nuclides for use in statistical approaches, and the second experiment in Chapter 5 concentrates on this.

4 REACTOR PARAMETER PREDICTION USING NUCLIDE MASSES

This chapter covers the parameter prediction workflow using nuclide masses as the input features. The methodology is introduced by detailing each experimental component and its implementation. This is split into four sections, which correspond to the four steps summarized in Figure 4.1.

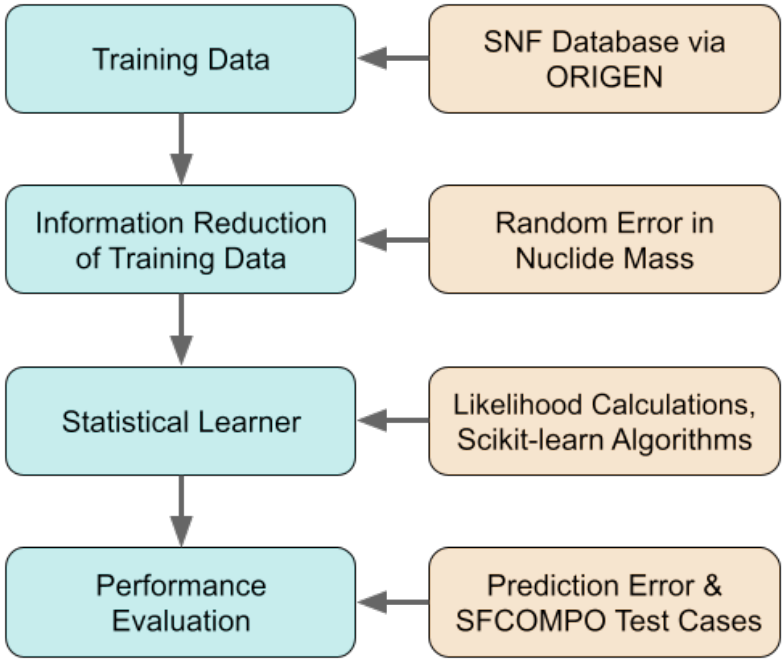


Figure 4.1: Flowchart of the experimental methodology and the way each step is being implemented.

Section 4.1 discusses how the training data set is obtained through simulations and computational means. The initial training data simulated via Oak Ridge Isotope GENeration (ORIGEN) is detailed in Section 4.1 This provides a set of spent nuclear fuel (SNF) observations with known reactor operation parameters, i.e., labels that are to be predicted. The four labels being predicted are as follows:

1. The classification of the **reactor type** is one of the three most common commercial power reactors: pressurized water reactor (PWR), boiling water reactor (BWR), or pressurized heavy water reactor (PHWR).

2. The **burnup** describes how much energy was produced by the fuel and has the units: MWd/MTU (or GWd/MTU), mega (or giga) watt-days per metric ton of initial uranium.

3. The **enrichment** is the percentage of uranium-235 (^{235}U) with respect to the entire amount of uranium in the fuel: $\%\,^{235}$U.

4. Lastly, the **time since irradiation**, or cooling time, is defined as how long the fuel has been out of the reactor core: $days$ (or $years$).

Next, the information reduction step is covered in Section 4.2, where uniform error is randomly applied. After this, the less-precise training data sets will be input to a statistical learner for the next step: training models.

Section 4.3 details the implementation of the algorithms introduced in 3.2.1. They use the features and labels in the training data sets to formulate a model.

After this, the algorithms must be evaluated for their prediction performance when given test samples (i.e., a new SNF measurement that has no labels according the to algorithm). The approach is shown in Section 4.4. First presented is the prediction performance of samples that are taken out of the training data set to be used as test cases, which is shown in Section 4.4.1. Second, an external test set with nuclide concentrations is used to show how this approach performs with real world measurements. This is discussed in Section 4.4.2.

4.1 Training Data Simulation

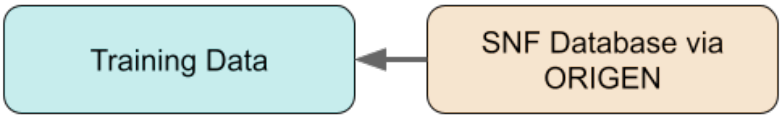


Figure 4.2: First portion of the flowchart from Figure 4.1 being described in this section.

Of interest to an entity trying to create a weapon is partially irradiated fuel if they have plutonium separations capabilities or any radioactive substance in the case of a dirty bomb. Thus, this work focuses on SNF from commercial power reactors. Ideally, a large enough database of SNF nuclide assays would be able to be used for this work. As a sufficiently large enough database does not exist, the database will be simulated via ORIGEN [27, 28].

4.1.1 Simulation Fidelity

Nuclear fuel cycle studies involve tracking the material flow of nuclear fuel. This can be anywhere from mining to waste management, or focus on a process step in between. Fuel cycle studies are not necessarily nuclear-specific. For example, they can be used to evaluate economic predictions, environmental impact, transportation planning, etc. In order to draw conclusions from these studies, it is common to use a nuclear fuel cycle simulator that tracks the quantities of interest. These allow the comparison of different fuel types, reactor technologies, material processing steps, etc.

There are simplifications researchers need to make in order to experiment in a controlled way. Fuel cycle simulators, built for a specific needs, must remove complicating factors that are less relevant to the study. For example, one tool might be suited well to large-scale systems analysis with little nuclear physics included in the models, and another might focus on detailed isotopics within a system to track plutonium.

Because a large portion of a nuclear forensics investigation relies on measuring isotopics, this work used ORIGEN [27], which is a part of the Standardized Computer Analyses for Licensing Evaluation (SCALE) 6.2 modeling and simulation suite of computational tools developed for nuclear design and safety [26]. ORIGEN was chosen for its physically detailed models of activation, depletion, and decay. Specifically, the ARP module of the code was used: ORIGEN-Automatic Rapid Processing (ORIGEN-ARP) [28].

ORIGEN calculates time-dependent nuclide concentrations (or quantities derived from these) that result from activation and depletion calculations. The physics (i.e., neutron transport and decay) calculations are carried out in other SCALE modules that solve the depletion equations. This generates libraries for ORIGEN that include the probabilities of reaction (i.e., cross sections) for the system.

To obtain an SNF recipe from a reactor simulation, ORIGEN uses the desired input power generation with the cross section library to calculate a flux, the resulting depletion, and the end composition (i.e., isotopic recipes or nuclide vectors). Another output is decay; the composition is computed using decay equations with nuclear data [37]. These compositions provide source terms for other calculations, such as decay emission spectra from neutrons, alpha particles, beta particles, and gamma rays. Other derived quantities like activity, decay heat, or radiological hazard factors are also an option.

ORIGEN-ARP allows users to access a wider range of simulations by interpolating between the pre-calculated libraries instead of creating new libraries. The libraries contained in ORIGEN-ARP for various reactor technologies and fuel assemblies are optimized to sets of ^{235}U enrichments for the relevant coolant and moderator densities. They also contain optimized burnup steps, which informed the burnup steps used in this work [28]. The SNF simulated in this work comes from a homogenized reactor core, and so does not take axial variations of burnup into account. Through ORIGEN, given

Sample ID	NT3G24.SF97-1		NT3G24.SF97-2		NT3G24.SF97-3	
Enrichment (%)	4.11		4.11		4.11	
Burnup (GWd/MTU)	17.39		30.48		42.10	
Nuclide ID	E/C [a]	E/C-1 (%)	E/C	E/C-1 (%)	E/C	E/C-1 (%)
U-234	0.936	-6.42	0.909	-9.15	0.935	-6.52
U-235	0.959	-4.11	0.983	-1.69	0.979	-2.09
U-236	1.002	0.19	1.007	0.73	1.004	0.39
U-238	1.003	0.27	1.001	0.09	1.001	0.08
Pu-238	0.793	-20.72	1.082	8.21	1.088	8.84
Pu-239	0.759	-24.10	0.952	-4.81	0.947	-5.33
Pu-240	0.886	-11.36	0.938	-6.16	0.925	-7.49
Pu-241	0.786	-21.41	1.033	3.26	1.022	2.25
Pu-242	0.850	-14.96	1.011	1.12	1.017	1.73
Np-237	0.822	-17.80	0.991	-0.86	0.963	-3.74
Am-241	0.587	-41.31	0.777	-22.28	0.778	-22.19
Am-242m	0.488	-51.19	0.800	-19.96	0.845	-15.51
Am-243	0.604	-39.61	0.894	-10.60	0.887	-11.25
Cm-242	0.875	-12.48	1.065	6.47	0.983	-1.67
Cm-243	0.822	-17.81	1.263	26.33	1.219	21.86
Cm-244	0.586	-41.40	1.068	6.85	1.056	5.55
Cm-245	0.621	-37.90	1.473	47.34	1.456	45.62
Cm-246	0.771	-22.87	15.748	1,474.80	1.551	55.13
Cm-247	–	–	1.581	58.13	1.513	51.33
Ru-106	1.094	9.45	1.060	5.99	1.022	2.18
Sb-125	0.776	-22.39	0.824	-17.57	0.543	-45.73
Cs-134	1.095	9.46	1.268	26.76	1.218	21.77
Cs-137	1.039	3.89	1.032	3.21	1.029	2.89
Ce-144	1.234	23.43	1.122	12.22	1.028	2.77
Nd-142	–	–	–	–	–	–
Nd-143	1.011	1.07	1.003	0.32	0.998	-0.22
Nd-144	0.987	-1.34	0.998	-0.24	1.026	2.57
Nd-145	1.012	1.17	0.994	-0.61	0.993	-0.73
Nd-146	0.994	-0.61	0.995	-0.54	0.994	-0.62
Nd-148	1.000	-0.01	1.000	-0.05	1.000	-0.04
Nd-150	0.976	-2.43	0.990	-1.00	0.986	-1.36
Sm-147	1.036	3.62	0.998	-0.21	1.017	1.74
Sm-148	0.874	-12.63	1.058	5.76	1.109	10.91
Sm-149	0.939	-6.08	1.049	4.85	0.981	-1.87
Sm-150	0.974	-2.62	0.976	-2.36	0.974	-2.58
Sm-151	0.664	-33.62	0.772	-22.77	0.749	-25.11
Sm-152	0.949	-5.05	0.820	-17.95	0.779	-22.07
Sm-154	0.986	-1.41	1.008	0.83	0.999	-0.09
Eu-154	0.857	-14.27	1.050	4.98	0.996	-0.37

Table 4.1: Example set of results from work that uses SFCOMPO test cases for PWR simulation benchmarking of SCALE in Reference [38].

an initial material composition, some reactor operation parameters, and a reactor type, one can quickly perform many different nuclear reactor simulations and obtain SNF recipes. Since there are nearly 500k simulations being used in this work, ORIGEN-ARP is able to provide these simulations with relative computational ease.

Of course, there may be some losses in simulation fidelity by taking this route. ORIGEN-ARP is well-validated for light water reactor (LWR) SNF [39]. Additionally, recycled SNF in the form of mixed oxide fuel has been benchmarked for the relevant reactors [40]. Still, there are some nuclides for which simulations are not well-matched to the experimental measurements. The systematic overprediction of Eu154 is documented in Reference [41]. Some details of computed versus experimental nuclide compositions for PHWRs are covered in Reference [42]. Table 4.1 shows a small slice of the results in Reference [38]. This work uses several SCALE modules to simulate nuclide measurements, and using a former release of Spent Fuel isotopic COMPOsition (SFCOMPO) [35, 36], compares these to the experimental measurements. Although ORIGEN-ARP was not used, some reactor core simplifications were made, where the target fuel rod had individual mixtures specified and the remainder of the core was homogenized. The $\frac{E}{C}$ in the table refers to the experimental to calculated ratio of each measurement, for which there are three sets of comparisons (labeled at the top by "Sample ID"). The highlighting is for nuclides that are in one or both of the experiments in this work, and tend to have larger errors. The first sample is a lower burnup and has the largest errors.

In summary, it is important to note that that the simulated training set as ground truth is not perfect truth, with the level of inaccuracy varying for some nuclides more than others. The prediction performance will be impacted by the poorly simulated nuclides, but a detailed study on which nuclide measurements to exclude is outside of the scope of this work.

4.1.2 Training Set Labels

The design of the training set is dependent on a number of factors. First, it must have a sufficient number of burnup sets and time since irradiation steps to provide robust prediction. This is chosen by maximizing the steps for both parameters, while balancing the computational limitations of a large training set. Through previous experience, an approximate limit would be around 10^6 database entries for the specific calculations in this work and employing reasonable computational limitations.

PWR	**BWR**	**PHWR**
CE14x14	GE7x7-0	CANDU19
W17x17	Abb8x8-1	CANDU28
S18x18	Atrium10x10-9	CANDU37
BW15x15	SVEA64-1	
VVER440		
VVER1000		

(a) ORIGEN designations for reactor technologies and fuel assembly design.

	PWR	**BWR**	**PHWR**
Power Density [MW/MTU]	25, 35, 41	10, 22	2.2, 18, 22
Burnup [GWd/MTU]	2–68	1–68	0.45–12.6
Moderator Density [g/cc]	0.71	{0.1, 0.3, 0.5, 0.7}	0.84
Enrichment [% ^{235}U]	{0.5, 1.5, 2, 3, 4, 5}	{0.5, 1.5, 2, 3, 4, 5}	0.711
Cooling Time [days]	{0–6000} in 100-day steps		

(b) Simulation parameters for ORIGEN input files.

Table 4.2: Training set database design and parameters for ORIGEN input files.

Secondly, the training set must represent what exists in the real world. This was accomplished by studying the spread of parameters in the SFCOMPO database [35, 36]. To ensure this, a variety of reactor types and assembly designs were included, listed in Table 4.2a. Table 4.2b lists the rest of the simulation inputs. These include not only the labels of prediction interest, ^{235}U enrichment, burnup, and time since irradiation,

but also other important simulation input parameters such as the reactor power density and the moderator density. (Water is both the moderator and coolant in all simulated reactor types.)

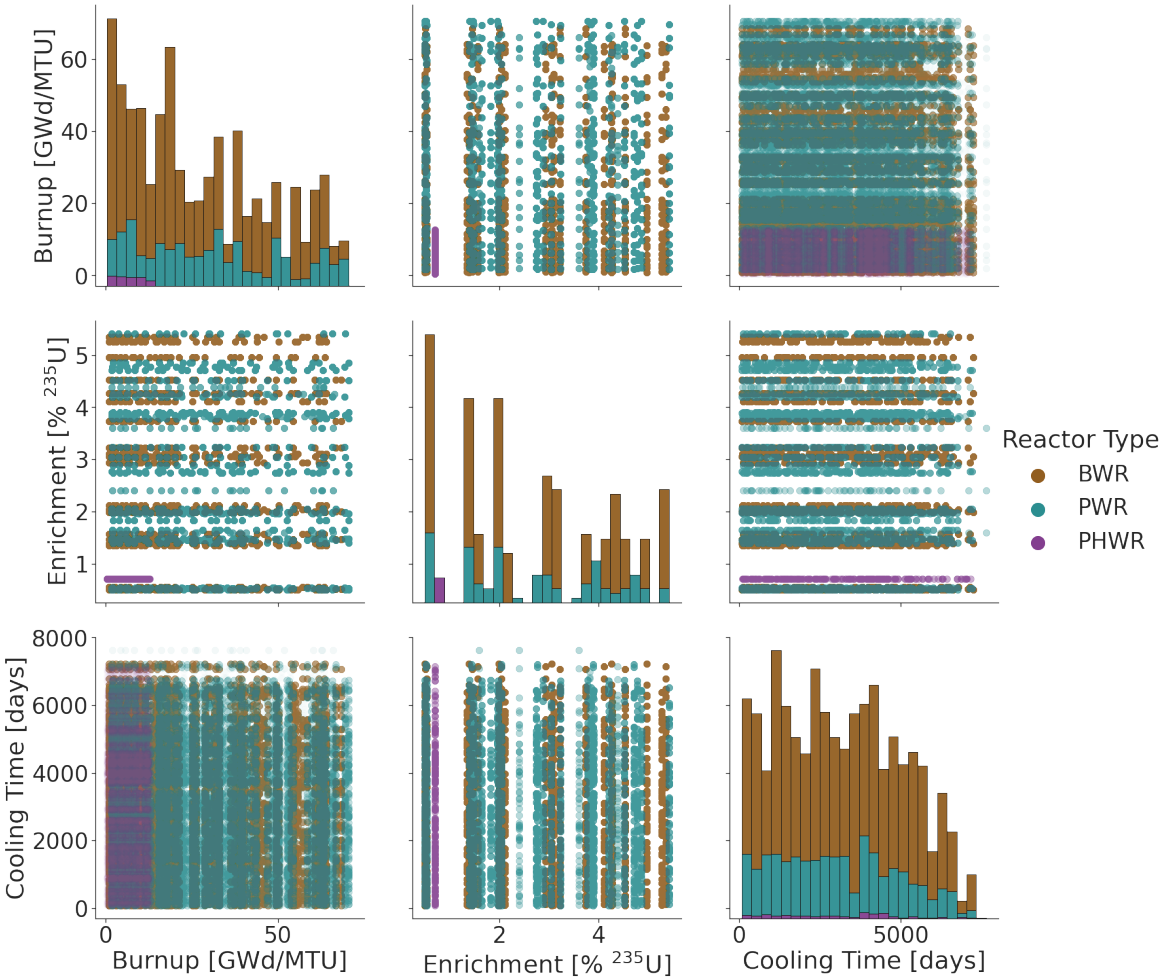


Figure 4.3: A combination of histograms and scatter plots to visualize the distribution of prediction labels in the training set.

The third factor influencing database design is machine learning (ML) algorithm performance. As mentioned in Section 3.2.2, many algorithms are developed with the assumption that the training set will be independent and identically distributed (i.i.d.). This is important so that the model does not overvalue or overfit a certain area in the training space. With the training set design, there are predetermined values for enrichment, burnup, and time since irradiation. While there are $21 - 28$ burnup steps

(depending on the reactor type) and 61 cooling time steps, there are only 6 values for enrichment. This creates the risk that the algorithm will end up being unable to generalize outside of those discrete values. Therefore, the burnup steps and time steps are perturbed randomly in a range that is ±10% and ±30% from the originally defined values, respectively. The enrichment also gets perturbed by ±10%, and not more because the cross-section libraries in ORIGEN-ARP are pre-calculated for those enrichment values, so deviating too far from them would result in inaccurate SNF simulations. The power densities and moderator densities were kept at the values defined in Table 4.2b. Additionally, natural variations in ^{234}U and ^{236}U were not considered. The percentage of the two uranium isotopes in fresh fuel were kept the same for all SNF simulations: 0.0356% for ^{234}U and 0.0184% for ^{236}U. The 0-valued burnup and cooling time entries were then filtered out so that the future calculations of relative error would be possible for all predictions.

The resulting training set is 450240 (or 4.5×10^5) entries. Figure 4.3 visualizes the somewhat even distribution of the burnup and cooling time parameters, and shows the lack of even distribution of the enrichment parameter through a combination of scatter plots and histograms. Note that there are many more BWRs present in the histograms because of the multiple moderator densities simulated (see Table 4.2b). The number of PWR, BWR, and PHWR entries in the training set is 120960 (26.8%), 322560 (71.6%), and 6720 (1.5%), respectively.

4.1.3 Training Set Features

The other design decision regarding the training set is related to which nuclides to track, i.e., the features. For this experiment, nuclide masses are necessary, and the most common measurements in SFCOMPO guide the list of nuclides tracked.

The set of training features of 29 nuclide masses listed in Table 4.3 was designed

with the following reasons in mind. First, the training set feature measurement units are chosen to be convertable to those present in the external, real-world test set: the SFCOMPO database. The ORIGEN simulations output the nuclide masses in grams, g, and they are converted to the units of milligrams per gram of initial uranium, mg/gU_i, when the models are externally tested against SFCOMPO. Second, the 29 nuclides chosen were based on the presence of measurements in SFCOMPO, where these were present in at least 100 of the samples in the database. The external test set is described in more detail in Section 4.4.2. These choices are made so that the training set entries mimic a full assay being done via mass spectrometry techniques, and thus represents what this work refers to as "perfect knowledge" scenario.

^{241}Am	^{242m}Am	**^{243}Am**	^{242}Cm	**^{244}Cm**	**^{134}Cs**	**^{137}Cs**	**^{154}Eu**
^{143}Nd	^{144}Nd	^{145}Nd	^{146}Nd	^{148}Nd	^{150}Nd	**^{237}Np**	**^{238}Pu**
^{239}Pu	**^{240}Pu**	^{241}Pu	^{242}Pu	^{147}Sm	^{149}Sm	^{150}Sm	^{151}Sm
^{152}Sm	**^{234}U**	**^{235}U**	^{236}U	^{238}U			

Table 4.3: Set of features saved for the first experiment, nuclide masses measured in *grams*. The bold nuclide masses overlap with the nuclides in Table 5.1.

4.2 Information Reduction

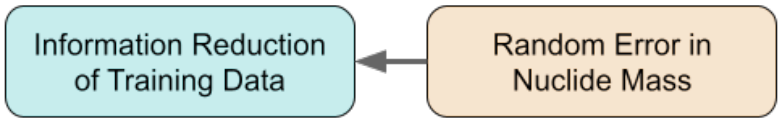


Figure 4.4: Second portion of the flowchart from Figure 4.1 being described in this section.

In this section, the information quality reduction is implemented as increasing the error of the nuclide mass measurements in the training database. After this, the statistical

models will be evaluated as to how they perform under increasing error in the nuclide measurements.

The training database for the first experiment is meant to be a proof of principle with the developed methodology, and mimic a scenario where there is "perfect knowledge" of a set of nuclides of interest. But the overall goal of this project is to determine how much information to what quality is needed to train an ML model that can provide SNF attribution by correctly predicting the reactor type, burnup, ^{235}U enrichment, and time since irradiation. Therefore error and uncertainty were injected into the nuclide mass measurements in the training database for the machine learning algorithms and maximum log-likelihood (MLL) calculations, respectively.

4.2.1 Scikit Algorithms

For the k-nearest neighbor and decision tree algorithms, a uniform error is applied randomly to each nuclide mass as follows. For a maximum error percentage of E_{max}, each nuclide mass is perturbed by a random value in the range: $[100 - E_{max}, 100 + E_{max}]$. This occurs for 10 error levels between $0\% < E_{max} < 20\%$: 0, 1, 2, 5, 8, 10, 12, 15, 18, 20. Therefore the 0% error case represents full knowledge of nuclide masses, and that knowledge slowly decreases up to 20%.

4.2.2 Maximum Log-Likelihood Calculations

For the MLL calculations, a uniform uncertainty was introduced to each nuclide mass. The uncertainty is calculated from error propagation of Equation 3.6 [23, 24]. Thus, each nuclide is given an uncertainty of 1, 5, 10, 15, and 20% via:

$$\sigma^2_{LogL} = \sum_i \left(\frac{x_{i,test} - x_{i,train}}{\sigma^2_{i,train}} \right)^2 (\sigma^2_{i,train} + \sigma^2_{i,test}) \tag{4.1}$$

where $x_{i,train}$ and $x_{i,test}$ are the nuclide measurements for the simulated/training set samples and the test samples, respectively, and $\sigma_{i,train}$ and $\sigma_{i,test}$ are their respective standard deviations calculated from the four uncertainty levels listed above.

4.3 Statistical Learning Implementation

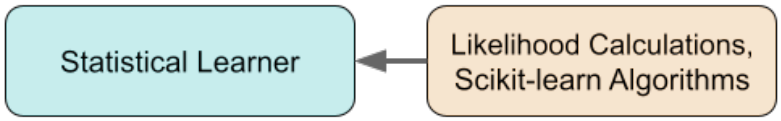


Figure 4.5: Third portion of the flowchart from Figure 4.1 being described in this section.

Since this work has yet to be benchmarked with simple algorithms, two straightforward algorithms were chosen, k-nearest neighbors and decision trees, and were introduced in Section 3.2.1. Also covered in that section is the mathematical framework of the MLL calculation method. The implementation details of these three approaches are covered here.

4.3.1 Scikit-learn Algorithms

The machine learning toolkit chosen for this work is scikit-learn [29], a package in python. Virtually all modern ML toolkits will have acceptably fast and reliable algorithms, but the use of python provides a platform for seamless integration of all the tools in the workflow. This section walks through the implementation of the two scikit algorithms.

The first step is splitting the training set into the features ($\vec{X}$, with 29 nuclide mass measurements), and a single set of labels ($\vec{y}$, e.g., burnup). After this, the set of features $\vec{X}$ is scaled so that there is a zero mean and unit variance; this is done because the magnitude of the features vary widely and it can be either helpful or required for

many scikit-learn algorithms. Next, the two algorithms are initialized. The labels that require regression analysis (burnup, enrichment, cooling time) are predicted using the `KNeighborsRegressor` and `DecisionTreeRegressor` in scikit-learn. The reactor type label uses classifiers: `KNeighborsClassifier` and `DecisionTreeClassifier`.

Each algorithm has a list of hyperparameters that impact the model complexity and prediction performance, as discussed in Section 3.2.2.2. Since this work is a demonstrative investigation, many of the default choices are retained. However, some of these were tuned using hyperparameter optimization, shown in Table 4.4. Hyperparameter optimization involves completing the training, predicting, and error evaluation process for a list of parameters that are of a continuous nature. This process is carried out with k for k-nearest neighbors, choosing the value that provides the best results. For decision trees, the maximum features was chosen to be 29 because the performance was highly variable otherwise, but this is not necessarily ideal for the best performance. Instead, only the maximum depth was tuned via optimization. There are also other parameters governing algorithm behavior that have a discrete set of choices. The selections made for each of these algorithms is summarized below (note this is just a subset of all available hyperparameters [29]):

1. `KNeighborsClassifier` & `KNeighborsRegressor`
    - Number of nearest neighbors (k) to include changes for each prediction category and is in Table 4.4.
    - Distance metric is "Manhattan" (aka L_1 loss, or absolute differences): $d_i = \sum_{j=1}^{N_{feats}} |x_{j,train} - x_{j,test}|$.
    - Sample weighting is with respect to distance and not uniform : $w_i = 1/d_i$.
2. `DecisionTreeClassifier` & `DecisionTreeRegressor`
    - Maximum number of features included is the length of the feature set: 29.

- Maximum depth of decision tree changes for each prediction category and is in Table 4.4.
- For both the classifier and regressor, the default splitting criterion (function for measuring the quality of a split) is kept. This is the Gini impurity for the former (see Equation 3.3), and mean squared error (aka L_2 loss, see Equation 3.4) for the latter.
- For the classifier, the `class_weight` is "balanced", meaning that the weights are inversely proportional to the frequency of each class (PWR, BWR, PHWR) in the training set.

Prediction Parameter	***k* (N neighbors)**	**Max Depth**	**Max Features**
Reactor Type	4	56	29
Burnup	1	77	29
Enrichment	2	45	29
Cooling Time	1	73	29

Table 4.4: Optimized algorithm hyperparameters for the 29 nuclide mass training set.

After the algorithms are initialized, the training set iteratively undergoes error injection according to the method in Section 4.2. Next, the (information reduced) data set is preprocessed by scaling and normalization because the nuclide concentrations vary by many orders of magnitude. After this, the training data features have a zero mean and unit variance. The initialized algorithm is then used to fit the training set and is then ready for prediction of "unseen" test samples.

Introduced in Section 3.2.2.1, these algorithms provide predictions based on the k-fold cross-validation (CV) approach (not to be confused with the k in k-nearest neighbors). Tested were 5, 10, and 15 CV folds and they all performed similarly because the training set is large. The results shown in Table 4.5 show the percent change in the

	% Difference, k=5 to k=10		**% Difference, k=5 to k=15**	
	kNN	DTree	kNN	DTree
Reactor Type	0.4	0.2	0.4	0.4
Burnup	-4.0	-2.1	-5.2	-2.6
Enrichment	-5.1	-6.1	-6.6	-6.7
Time Since Irradiation	-1.8	-3.1	-2.3	-3.8

Table 4.5: Comparison of prediction errors between different k-fold CV implementations.

accuracy score or mean absolute error (MAE) relative to when $k = 5$. This means that the accuracy score is under 0.5% higher for reactor type predictions for both k-nearest neighbors and decision trees changing k folds from 5 to 10 or 15. And the MAEs are slightly lower (under 7% difference) when changing k folds to 10 or 15. Given the close performance, the number of CV folds was chosen to be 5 based on computational expediency. The `KFold` cross-validator with observation shuffling (i.e., the data set was shuffled prior to splitting) in the scikit package carried out this task for the regression prediction cases. For reactor type classification, the `StratifiedKFold` cross-validator was used since there is an imbalance of classes in the training database.

Because the performance needs to be compared to the MLL predictions, the scikit method `cross_val_predict` was used, as it returns the predictions of each CV fold as it becomes the test set. Thus, the entire training set becomes a test case at some point, and the predictions return equal the number of entries in the training set.

The scripts written to run the scikit-learn algorithms were deployed using University of Wisconsin (UW)-Madison's Center for High Throughput Computing (CHTC) resources and the UW campus grid.

4.3.2 Maximum Log-Likelihood Calculations

The MLL calculation method is implemented in python using the SciPy statistics toolkit and NumPy functionality [43, 44]. In this method, one test sample is removed from the training set at a time, and the log-likelihood calculation in Equation 3.6 is performed between the test sample and all the rows of the training set. The observation with the largest log-likelihood is returned as the predicted labels. Just like with the scikit-learn algorithm-based predictions, the entire training set becomes a test sample at some point, so the predictions returned equal the number of entries in the training set.

A key difference in how the MLL method provides predictions is that it returns all the labels together, so it is not doing any generalizing on the continuous variable labels (burnup, enrichment, and cooling time). It is analogous to k-nearest neighbors where $k = 1$, because there is no model being created, it just matches upon the call for prediction. Because of this, the MLL method performing well is highly dependent on the training set being sufficiently large.

The scripts written to run the MLL calculations were deployed using UW–Madison's CHTC resources, the UW campus grid, and the Open Science Grid (OSG) [45, 46].

4.4 Prediction Performance Evaluation

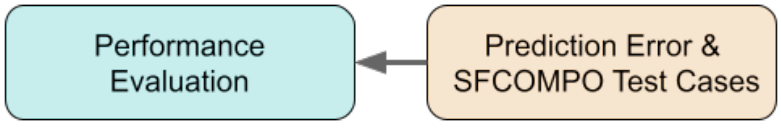


Figure 4.6: Fourth portion of the flowchart from Figure 4.1 being described in this section.

As previously introduced in Section 3.2.2.1, the prediction performance is measured by evaluating the accuracy of the reactor type classification or the error of the regression

cases (burnup, ^{235}U enrichment, cooling time). These performance metrics for all four prediction types are compared across the three algorithms used: k-nearest neighbors (denoted in plots as *kNN*), decision trees (denoted in plots as *Dec Tree* or *DTree*), and MLL calculations.

4.4.1 Random Error Impacts on Prediction

To judge the degradation of predictions of the algorithms with increasing nuclide mass measurement error (i.e., reduced information quality, detailed in Section 4.2), several plots are made with the introduced error on the horizontal axis and a prediction performance metric on the vertical axis. The vertical axis is always oriented so that lower is poorer performance and higher is better performance. This is why Figures 4.9–4.11 present a negative error on the vertical axis. Additionally, the data points on all the plots have a small horizontal separation to show error bars that are otherwise impossible to see.

In all of the results in this section, the statistics being reported is on all entries in the training set. This means that for the performance metrics introduced in Section 3.2.2.1, the calculations will all include $N_{\text{samples}} = 4.5 \times 10^5$.

4.4.1.1 Reactor Type Classification

Figure 4.7 shows the balanced accuracy of reactor type classification, where a score of 1 is perfect prediction and a score of 0 is random classification. The error bars reflect a 99% confidence interval. While the two scikit-learn algorithms follow a very similar path of decreased accuracy as the error increases, the MLL calculation approach appears to be more robust to the nuclide mass measurement error. Another interesting result is that the MLL calculation performs slightly worse for low errors. If the expected measurement errors of nuclide masses in a training database or in a test sample can

be guaranteed to be better than 2%, the MLL calculation is no longer the obvious preferred choice for reactor type prediction.

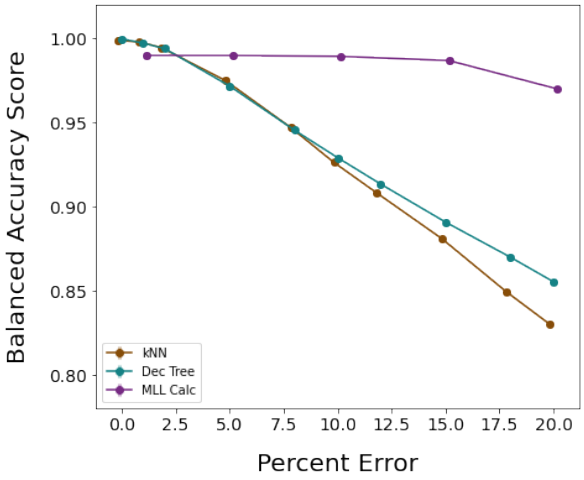


Figure 4.7: Prediction performance of reactor type as measured by balanced accuracy with respect to uniform/random error applied to the nuclide mass measurements in the training set.

Although the balanced accuracy score provides more information about classification performance for an imbalanced data set (the training set is 26.8% PWR, 71.6% BWR, and 1.5% PHWR), it still does not provide much detail about what is being misclassified. To probe this further, Figure 4.8 shows three sets of confusion matrices, originally introduced in Section 3.2.2.1. The off-diagonal squares are the fraction of false positives for each reactor type, where the predicted label (horizontal axis) is something other than the true label (vertical axis). The false positive fractions are normalized to the number of true labels. Below the fractions inside the pixels are the raw numbers of false positives. The diagonal squares have three numbers in them. The top numbers are the fraction of true positives for each reactor type (normalized to true labels), where the predicted label (horizontal axis) is equal to the true label (vertical axis). The middle number in parentheses is the true positive fraction subtracted by one: TP − 1. Again, the bottom number is the raw number of true positives.

A coloring is introduced to quickly identify departures from the ideal solution.

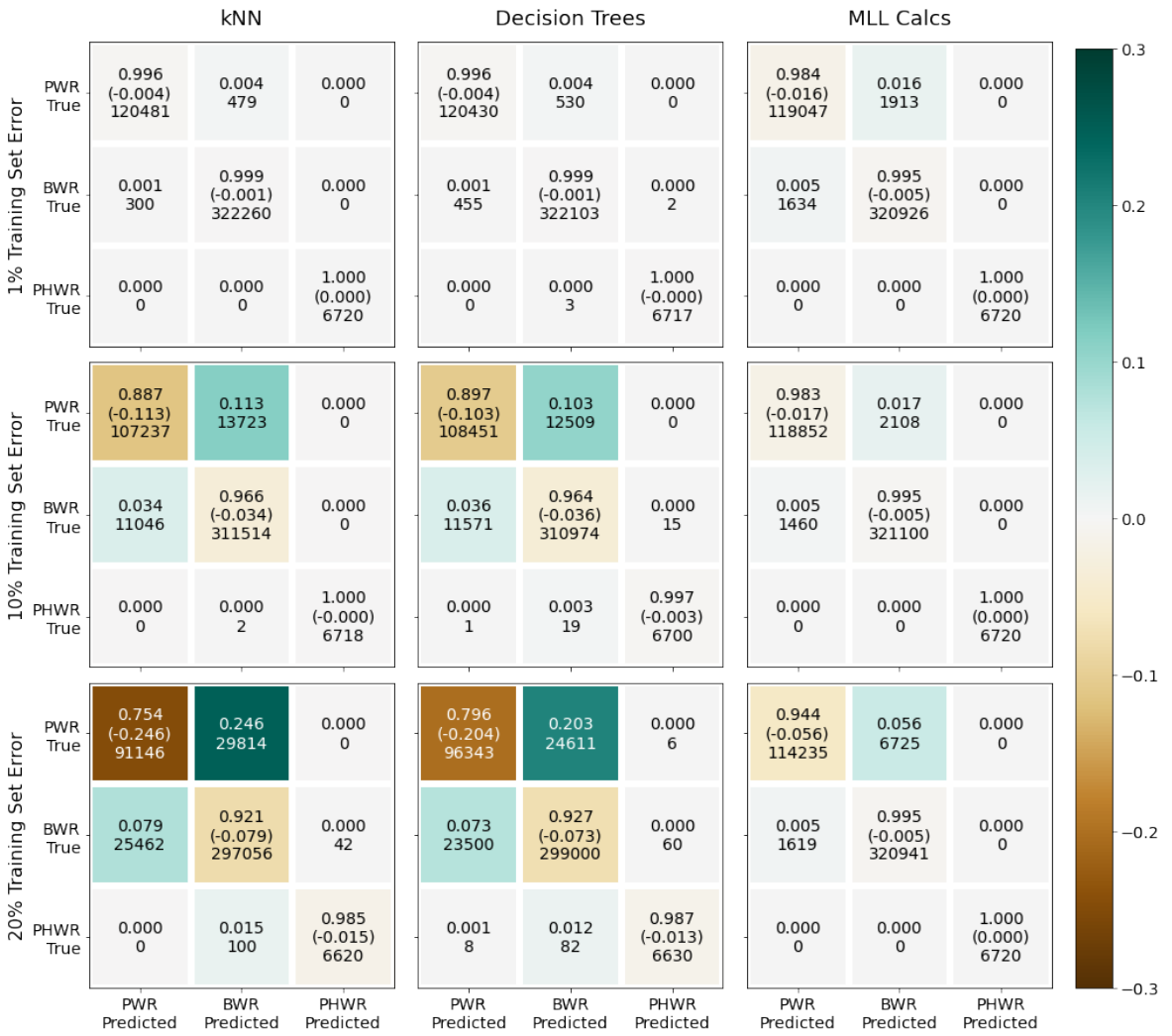


Figure 4.8: Confusion matrices of reactor type prediction for each algorithm at three training set error levels: 1%, 10%, and 20%, in the top, middle, and bottom panels, respectively.

However, if the coloring is based on the true positive fraction, the diagonal has an ideal of 1 and the off-diagonal has an ideal of 0. By relying on the $TP - 1$ for the diagonal elements, the ideal is 0 for all pixels, with departures from ideal being negative on the diagonal and positive on the off-diagonals. To allow the fractions to be rapidly perceived, a colorbar provides perceptually uniform shading for these pixels that deviate from 0. The deviation from 0 to the negative in the diagonal pixels is thus a different color (brown) than the deviation from 0 to the positive in the off-diagonal pixels (teal).

Perfect performance is represented by the middle of the colorbar, white.

In the top panel of Figure 4.8, the three algorithms are presented for the 1% random error case. In Figure 4.7, one can see these three data points on the plot clustered near the top showing almost-perfect performance. (Recall that the true positive fractions in the confusion matrices do not map directly to the balanced accuracy score, which puts more weight on the underrepresented classes.) The confusion matrices give more dimension to this near-perfect reactor type classification performance. The majority of the misclassification is in PWRs being classified as BWRs: 0.4% for k-nearest neighbors and decision trees, and 1.6% for MLL calculations. However, there are also some BWRs that are misclassified as PWRs: 0.1% for k-nearest neighbors and decision trees, and 0.5% for MLL calculations. There are zero misclassified PHWR cases and zero LWR cases misclassified as PHWR; the value of 0.000 to three decimals fraction here represents a real zero-count, but this is not necessarily the case for the other sets of confusion matrices. The PWR/BWR distinction is known to be a difficult problem[20], so the correct PHWR classifications are not particularly notable for this discussion.

The middle panel of Figure 4.8 shows confusion matrices for the three algorithms for the 10% random error case. In Figure 4.7, one can see these three data points on the plot, where the MLL point is near a balanced accuracy score of 1, and the scikit-learn algorithms both have score of around 0.93. As with the 1% error case, the majority of the misclassification is in PWRs being classified as BWRs: 11.3% for k-nearest neighbors, 10.3% for decision trees, and 1.7% for MLL calculations. The BWRs are being misclassified as PWRs at the following percentages: 3.4% for k-nearest neighbors, 3.6% for decision trees, and 0.5% for MLL calculations. Note how the performance of the MLL calculations are nearly the same for both error levels, which is shown by the MLL line in Figure 4.7. Because of the normalization, the LWRs that are misclassified as PHWRs appear to be zero. However, this does happen, just rarely: 15 BWRs are

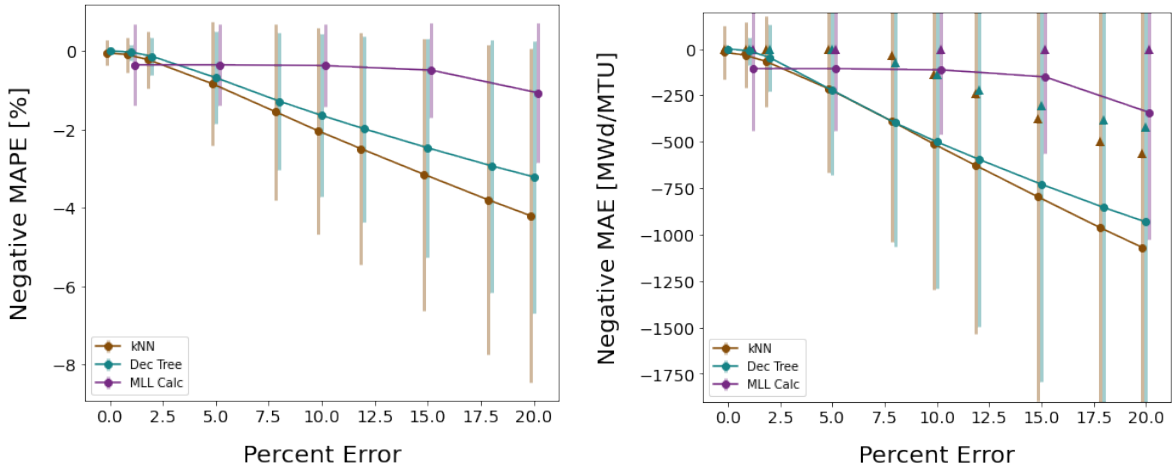
classified as PHWR using decision trees. Also, k-nearest neighbors and decision trees misclassified PHWR as an LWR 2 times using the former and 20 times using the latter (no PHWR misclassifications happened using MLL).

The bottom panel of Figure 4.8 shows confusion matrices for the three algorithms for the 20% random error case. In Figure 4.7, one can see these three data points on the plot, where the MLL point is near a balanced accuracy score of 0.97, k-nearest neighbors is around 0.83, and decision trees is around 0.86. As with the previous two error cases, the majority of the misclassification is in PWRs being classified as BWRs: 24.6% for k-nearest neighbors, 20.3% for decision trees, and 5.6% for MLL calculations. The BWRs are being misclassified as PWRs at the following percentages: 7.9% for k-nearest neighbors, 7.3% for decision trees, and 0.5% for MLL calculations. PHWRs are misclassified as an LWR 100 times for k-nearest neighbors, 82 times for decision trees, and 0 times for MLL calculations. LWRs were misclassified as PHWR 42 times for k-nearest neighbors, 66 times for decision trees, and 0 times for MLL calculations.

4.4.1.2 Regression Results

Each set of plots for a given prediction parameter in this section shows both the relative error (mean absolute percentage error (MAPE)) and the absolute error (MAE). In addition to the MAE on the second plot for each regression case, the median absolute error (MedAE) is represented as a triangle on the plot as well. These three errors taken together provide more detailed information about the performance of each algorithm when faced with training set noise. As previously mentioned, the vertical axis has negative errors, so that higher is always better.

Figure 4.9 demonstrates the burnup prediction performance, with the MAPE in Figure 4.9a and the MAE and MedAE in Figure 4.9b. In these figures, the error bars reflect one standard deviation of the percentage errors or the absolute errors, respectively. As with the reactor type prediction in Figure 4.7, the MLL method is robust to training



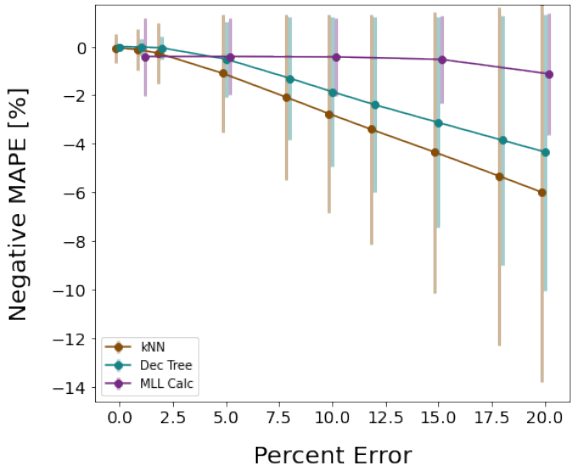
(a) Negative MAPE of burnup regression with respect to random error.

(b) Negative MAE of burnup regression with respect to random error.

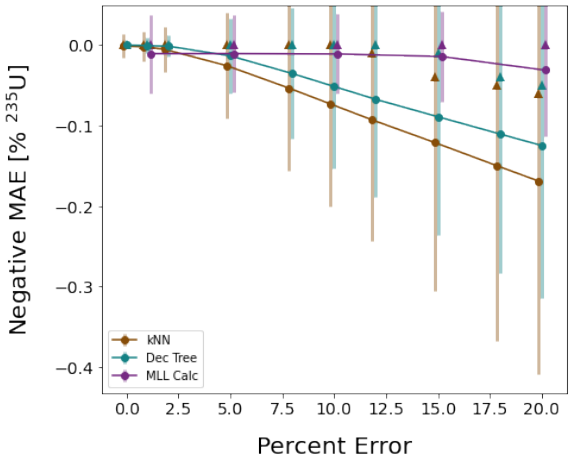
Figure 4.9: Prediction performance of burnup as measured by relative and absolute errors with respect to uniform/random error applied to the nuclide mass measurements in the training set.

set error but performs slightly worse at low error values. All three methods calculate burnup with a maximum error of -5% or $-1000\,MWd/MTU$ at 20% error in the training set. The MedAEs show a more encouraging picture of the performance as compared to the MAEs. It is interesting that the scikit-learn algorithms and the MLL calculations diverge at 5% training set error for MAPE and MAE, but at 10% training set error for the MedAE.

Figure 4.10 demonstrates the ^{235}U enrichment prediction performance, with the MAPE in Figure 4.10a and the MAE and MedAE in Figure 4.10b. In these figures, the error bars reflect one standard deviation of the percentage errors or the absolute errors, respectively. Again, the MLL method is robust to training set error but performs slightly worse at low error values. All three methods calculate enrichment with a maximum error of -6% or -0.17 % ^{235}U at 20% error in the training set. Again, the MedAEs show a more encouraging picture of the performance as compared to the MAEs. The scikit-learn algorithms and the MLL calculations diverge at 10% training set error for MAPE and MAE, but at 15% training set error for the MedAE.

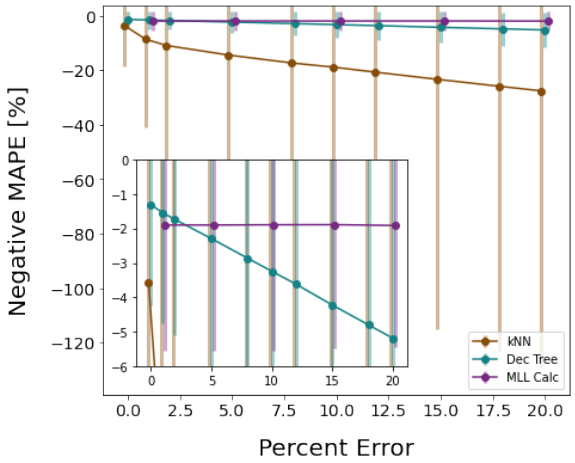


(a) Negative MAPE of ^{235}U enrichment regression with respect to random error.

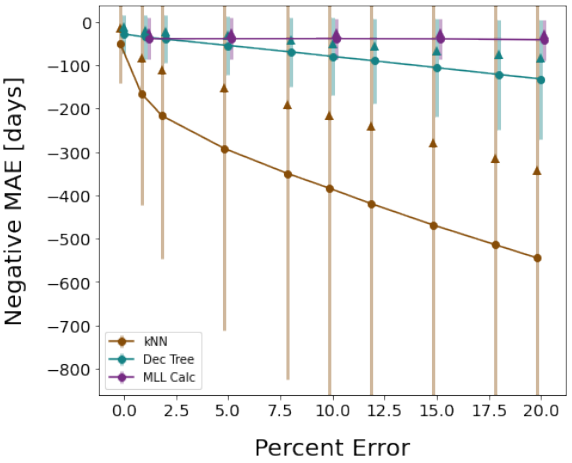


(b) Negative MAE of ^{235}U enrichment regression with respect to random error.

Figure 4.10: Prediction performance of enrichment as measured by relative and absolute errors with respect to uniform/random error applied to the nuclide mass measurements in the training set.



(a) Negative MAPE of time since irradiation regression with respect to random error.



(b) Negative MAE of time since irradiation regression with respect to random error.

Figure 4.11: Prediction performance of time since irradiation as measured by relative and absolute errors with respect to uniform/random error applied to the nuclide mass measurements in the training set.

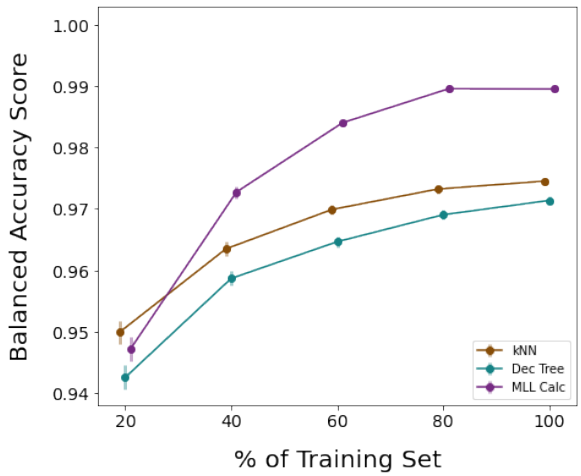
Last, the time since irradiation prediction performance for the three algorithms with respect to increasing nuclide mass error is pictured in Figure 4.11. The MAPE is shown in Figure 4.11a, including an inset to show more detail about the behavior of

the decision trees and MLL calculations curves, and the MAE and MedAE are shown in Figure 4.11b. The error bars reflect one standard deviation of the percentage errors or the absolute errors, respectively. The MLL method is most robust to training set error, but for this prediction parameter, the behavior of k-nearest neighbors is unique here versus the previous two regression categories. Time since irradiation is predicted with a maximum error of -30% or $-550\,days$ at 20% error in the training set using the k-nearest neighbors algorithm. If one ignores the clearly anomalous k-nearest neighbors performance by using the inset in Figure 4.11a and focuses instead on decision trees, those values are -6% and $-120\,days$. The MLL calculations remain nearly constant at approximately 2% prediction error for all training set error levels. The MedAE, as usual, shows a more encouraging picture of the performance as compared to the MAE.

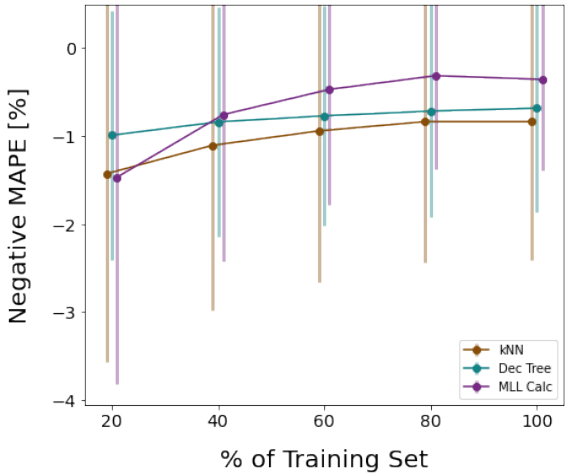
Overall, the robustness to training set error that is present in the MLL calculations applies to all of the prediction parameters. The decision trees performance is second best, and k-nearest neighbors performs the worst. In the case of time since irradiation, k-nearest neighbors degrades incredibly fast with introduced training set error. While it is difficult to draw a baseline for minimum acceptable behavior on these plots, these performances can serve as a benchmark for the work presented in Chapter 5.

4.4.1.3 Model Generalization

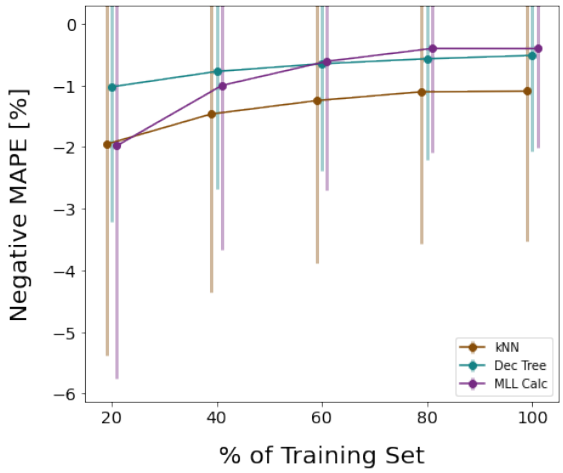
Although a key takeaway from Section 4.4.1 is that the MLL calculations are the most resilient to introduced error in the training set features for all four prediction categories, there is another aspect of the algorithm performance not explicitly shown in those plots: generalization. MLL does not generalize to unseen data; it provides predictions based on finding the closest training set entry to the test sample. It (usually) outperforms the scikit-learn methods in part because the training set is so large. This is also true for the k-nearest neighbor implementations where $k = 1$ (burnup and cooling time, as seen in Table 4.4).



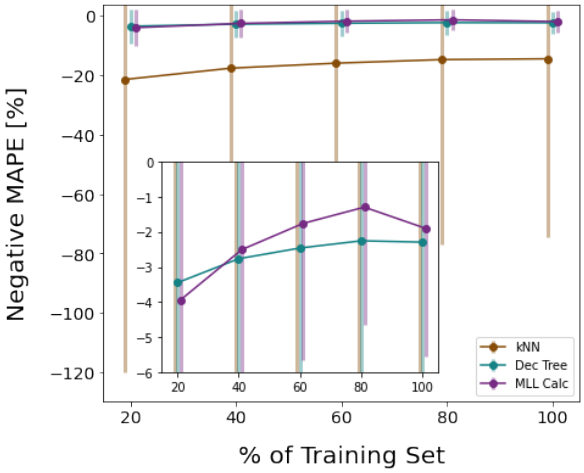
(a) Balanced accuracy of reactor type classification with respect to training set size.



(b) Negative MAPE of burnup regression with respect to to training set size.



(c) Negative MAPE of ^{235}U enrichment regression with respect to training set size.



(d) Negative MAPE of time since irradiation regression with respect to training set size.

Figure 4.12: Learning curves for reactor type, burnup, enrichment, and time since irradiation with respect to increasing fraction of the training set, for 5% training set random error.

One way to show that an algorithm is generalizing well in comparison to others is to view the shape of its learning curve (introduced in Section 3.2.2.2): the prediction performance with respect to training set size. It is crucial to have the training sets be identical for each algorithm, so they were created in advance and the learning curves are constructed manually rather than relying on the scikit-learn method. Smaller training

sets were created from the original one by taking 80%, 60%, 40%, and 20% of the entries. The training sets are all stratified so that the original fractions of BWR, PWR, and PHWR are retained. They are also built on top of one another, so the 20%-size training set is contained within the 40%-size set, and so forth.

Learning curves were constructed for all four prediction categories, demonstrated in Figure 4.12. As in Figures 4.7–4.11 , the vertical axis is always oriented so that lower is poorer performance and higher is better performance; also, the error bars reflect a 99% confidence interval for Figure 4.12a, and one standard deviation of the average percentage errors for Figures 4.12b–4.12d. These learning curves represent the 5% random error case in Figures 4.7–4.11, so the scores/errors in these figures are the data points at the 100% training set level in Figure 4.12. Therefore, the leftmost data in Figure 4.12 will show the MLL point being slightly above the scikit-learn points for the reactor type, burnup, and enrichment predictions, and the k-nearest neighbor point is below the MLL and decision trees points for time since irradiation.

Figure 4.12a shows that the balanced accuracy score of reactor type classification for the MLL calculations decreases more at lower training set size than for the scikit-learn algorithms. Here, the curve crosses below the k-nearest neighbors curve at the lowest training set size of 20%. For the burnup MAPE in Figure 4.12b and the enrichment MAPE in Figure 4.12c, the MLL curve crosses below both of the scikit-learn algorithm curves. This happens between 20% and 40% training set size for burnup, and between 40% and 60% training set size for enrichment. Lastly, Figure 4.12d shows a different arrangement, which is to be expected from the results shown in Figure 4.11, where the k-nearest neighbors performance is significantly worse than the other two algorithms. Because the k-nearest neighbors curve and error bars are so large, there is an inset showing a closeup of the other two curves above -6%. The decision trees and MLL calculations curves now appear to follow the trend in the burnup and enrichment cases,

and the MLL curve crosses under the decision trees curve between 20% and 40% training set size.

There is a dependence on training set size for all three algorithms in Figure 4.12. For the most part, the k-nearest neighbors and decision trees curves follow an approximately parallel path, whereas the MLL method shows an increased rate of degradation at low training set sizes. Since this training set is large enough, i.e., the prediction parameters were included in small enough steps, that MLL has consistent performance at the larger sizes, there is not a concern in this work about its inability to generalize. It must be noted, however, that the MLL approach requires a fine grid of simulation parameters in a training database to perform better than the simple scikit-learn algorithms.

4.4.1.4 Reactor Type Prior Knowledge

There is similar work being done to this work that focuses on similar prediction categories but in a serial manner, i.e., first determining the reactor type before moving forward with other predictions [47]. This work predicts reactor operation parameters while blind to the reactor type, but it makes sense intuitively that having previous knowledge of the reactor type would allow more accurate regression of these parameters. Therefore, the change in regression performance from the reactor type-blind predictions to having prior knowledge of the reactor type is discussed.

The workflow was repeated for the three regression cases where they were trained on reactor type-specific training sets. A 5% random error was applied to these training sets, and the 5% random error full training set was used as comparison. The errors for each algorithm (k-nearest neighbors, decision trees, and MLL calculations) were tallied for each regression category (burnup, enrichment, and time since irradiation) and within that, for each reactor type (PWR, BWR, PHWR). Two sets of error were tracked: whether the reactor type was *known* or *unknown* prior to prediction.

Table 4.6 shows the MAPEs for each regression category and within that, for each

reactor type (PWR, BWR, PHWR). The columns are separated first by the algorithms and second by whether the reactor type was known or unknown prior to prediction, denoted as K and U, respectively. Most of these relative errors are quite low, and around or under 2%. So, e.g., despite burnup prediction from PHWRs improving, it was by 0.61%, a precision of which may not be of concern. Still, these performance differences can be looked at in more detail.

		kNN		Dec Trees		MLL Calcs	
Prediction Parameter	Reactor Type	K	U	K	U	K	U
Burnup % $[MWd/MTU]$	PWR	0.60	0.66	0.54	0.75	0.24	0.25
	BWR	0.88	0.90	0.60	0.66	0.40	0.40
	PHWR	0.66	1.27	0.14	0.54	0.28	0.28
Enrichment % [% ^{235}U]	PWR	0.85	0.99	0.36	0.48	0.26	0.29
	BWR	1.14	1.16	0.51	0.54	0.45	0.46
	PHWR	0.00	0.00	0.00	0.02	0.00	0.00
Time Since Irradiation % $[days]$	PWR	11.44	10.48	2.35	2.19	1.55	1.46
	BWR	15.39	15.48	2.27	2.28	2.06	2.05
	PHWR	19.32	34.41	4.96	4.52	2.30	2.30

Table 4.6: MAPEs for the three prediction cases for each algorithm at 5% training set error. K refers to *known* reactor type and U refers to *unknown* reactor type prior to regression.

To better see these performance differences, the percent change in prediction MAE for each algorithm and reactor type between the reactor type being known versus unknown prior to prediction was calculated as: $100 \cdot \frac{MAE_{unknown} - MAE_{known}}{MAE_{unknown}}$. This was chosen to be relative to the unknown error since that is the benchmark in this case. Figure 4.13 is three heatmaps that show this percent change for each prediction category, algorithm, and reactor type. This value is reflected by a diverging color bar as well as a positive or negative percentage in each square. The positive percentages indicate the error decreased/improved from the unknown reactor type case to the known reactor

type case. The negative percentages indicate the error increased/worsened from the unknown to the known case.

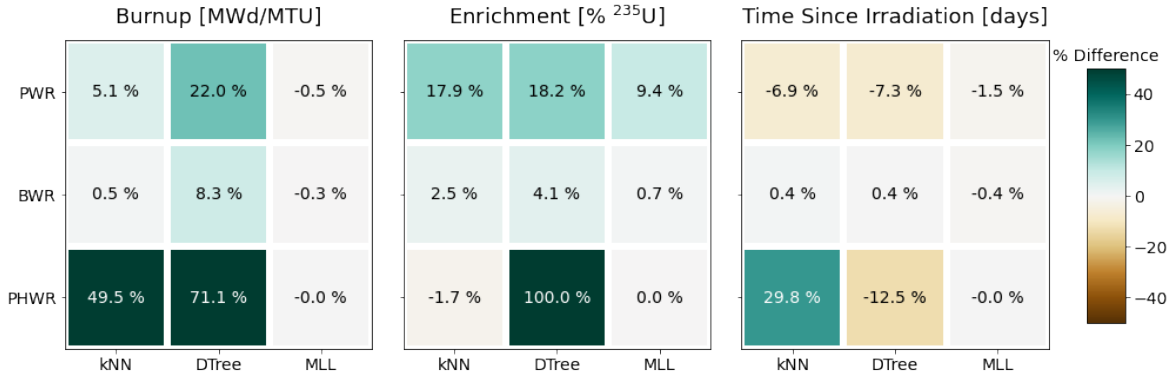


Figure 4.13: Heatmaps for the three regression cases showing the percent difference in prediction error between a known reactor type and unknown reactor type using a 5% training set error.

For burnup prediction, most differences are within ±10% except for three scenarios. The decision tree algorithm has improved burnup prediction for PWRs by 22.0% and for PHWRs by 71.1% given a known reactor type. The k-nearest neighbors algorithm has 49.5% improved burnup prediction for the PHWR. For ^{235}U enrichment, the PWR predictions improve by approximately 18% for the scikit-learn algorithms. Even though the value is within ±10%, this is the only scenario where there is an appreciable difference in MLL performance. The decision tree enrichment prediction of PHWRs also has a sizeable improvement of 100.0%. The time since irradiation predictions for the most part do not show improvement outside of ±10%. Of note is some volatile behavior for the PHWR case with the scikit-learn algorithms. While k-nearest neighbors improves by 29.8%, the decision tree predictions were worse by 12.5%. Since the main concern here is showing how prediction performance improves with prior reactor type knowledge, this reduction in performance is odd but not worthy of further investigation.

The improvements in the PHWR predictions are not surprising since the generalization of the scikit-learn algorithms could lead to the unique PHWR cases being ignored,

since they are after all only 1.5% of the training set. Another interesting result is that the BWR predictions experience no large changes, which makes sense given that they comprise 72% of the training set. Also, the MLL predictions are approximately the same, which is expected because this algorithm does not generalize, and the prediction comes as a set of labels and is therefore already linked to the reactor type. Overall, it is important to be aware that the regression labels coming from a PHWR will be unlikely to be optimal results (except for those from MLL calculations).

4.4.2 SFCOMPO Test Set

The testing described in Section 4.4.1 describes the process of evaluating the methodology with test cases drawn from the training database. It is also helpful to test the methodology against real assays of SNF. The SFCOMPO database was created to allow access to these sorts of measurements linked to the reactor operation parameters being predicted in this work [35, 36]. The only parameter not part of the SFCOMPO database is the time since irradiation, so that is not predicted here.

The database used in this work is a filtered version of all the entries in the original database. First, only the nuclide concentration measurements are kept so that the training set measurements could be converted to the units in SFCOMPO. The assays in SFCOMPO are presented as nuclide concentrations with the units milligrams per grams of initial uranium, or mg/gU_i. The training set of nuclide measurements in *grams* is converted to these concentration units prior to prediction is converted to these concentration units prior to prediction. Second, only the PWR, BWR, and PHWR entries are retained and all other reactor types are excluded. Third, uranium-gadolinium fuels are not simulated in the training set and therefore are also removed from the testing set. Last, duplicate entries for some measurements exist and the first entry is kept in these cases.

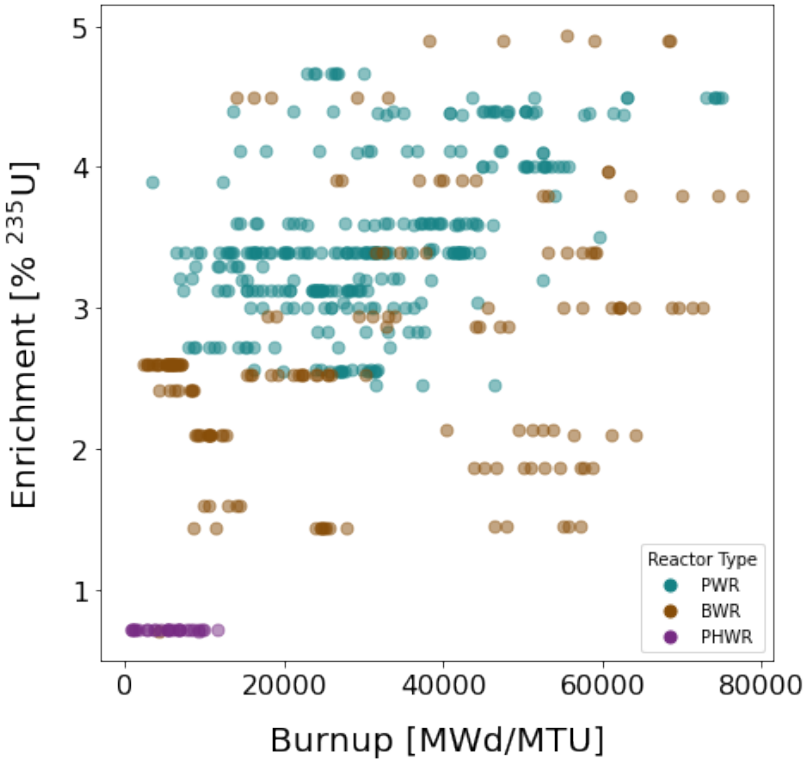


Figure 4.14: Scatter plot showing the range of reactor operation parameters in the SFCOMPO testing set that are being predicted.

In all, there are 505 test cases that are able to compare against the training database. The number of each reactor type is as follows: 312 PWRs, 165 BWRs, and 28 PHWRs. The space of enrichment and burnup values is visualized in Figure 4.14. These are sufficiently represented in the training set design, as pictured in Figure 4.3, although the proportions of PWR and BWR are approximately opposite to the training set.

There is one main issue with using SFCOMPO as a testing set: missing nuclide measurements. The feature set of 29 nuclides in Table 4.3 was chosen based on the frequency of these measurements being present in the database at an arbitrary level of 100 measurements. This happened before filtering the uranium-gadolinium fuel, so there are some nuclide measurements present at under 100 counts. Each nuclide's frequency in SFCOMPO is listed in Table 4.7. While every assay contains several plutonium measurements and most contain uranium measurements as well, the remaining nuclides

are present at a much lower rate.

^{241}Am	237	^{145}Nd	162	^{147}Sm	97
^{242m}Am	110	^{146}Nd	139	^{149}Sm	97
^{243}Am	203	^{148}Nd	275	^{150}Sm	97
^{242}Cm	214	^{150}Nd	121	^{151}Sm	97
^{244}Cm	269	^{237}Np	155	^{152}Sm	97
^{134}Cs	113	^{238}Pu	369	^{234}U	355
^{137}Cs	185	^{239}Pu	505	^{235}U	479
^{154}Eu	100	^{240}Pu	505	^{236}U	462
^{143}Nd	162	^{241}Pu	504	^{238}U	433
^{144}Nd	113	^{242}Pu	505		

Table 4.7: Number of assays each nuclide is measured for in the SFCOMPO database.

Although some algorithms in theory can handle null values in the testing stage, scikit-learn does not currently include this capability. The MLL method is designed to handle null values, however. This is done by converting them to zero and filtering out all zero-valued nuclides during the likelihood calculations. But there is a technique more commonly applied than imputing missing values with zero. This involves taking the mean or median of the existing feature measurements in the testing set and applying that value to the assays in which it is missing. The remainder of this section discusses using the three algorithms to predict the SFCOMPO test cases where the nulls are both imputed using zero and the mean.

4.4.2.1 Reactor Type Classification

Table 4.8 presents two metrics for the two missing value techniques: the accuracy and balanced accuracy scores. The accuracy scores for both the mean-imputed nulls and zero-imputed nulls test sets are mostly under 0.62, which is the fraction of PWR entries. Therefore, a classifier could predict PWR every time and do better than these accuracy scores. For the zero-imputed nulls test set predictions using MLL, however, the accuracy

of 0.72 does exceed the "majority guess" accuracy of 0.62. Since MLL calculations filter out null values, it is expected that the scores will be higher for all prediction categories where MLL is being used with the zero-imputed nulls test set. This expected MLL performance also holds true when looking at the balanced accuracy score. A balanced accuracy score of 0 denotes random guessing, but it can also be negative if the classifications are worse than random guessing. The balanced accuracy of 0.63 for the zero-imputed nulls case is a promising result. The balanced accuracies of k-nearest neighbors and decision trees are all quite low. Also, the higher accuracies correspond to lower balanced accuracies, and vice versa. Therefore, further investigation is necessary.

	Accuracy Scores			Balanced Accuracy Scores		
Null Handling	kNN	DTree	MLL	kNN	DTree	MLL
Mean Imputation	0.52	0.60	0.39	0.09	0.12	-0.01
Zero Imputation	0.45	0.42	0.72	0.21	0.30	0.63

Table 4.8: Accuracy and balanced accuracy scores for reactor type prediction of the SFCOMPO test cases.

Figure 4.15 allows for a deeper look into what is happening with the reactor type predictions for both of the SFCOMPO test sets by studying the confusion matrices, which are introduced in Sections 3.2.2.1 and 4.4.1.1. The matrices from using mean-imputed null values are in the top panel and the matrices from using zero-imputed null values are in the bottom panel. The scikit-learn algorithms have a higher accuracy for the mean-imputed test set than their zero-imputed counterparts, but the balanced accuracies follow the opposite direction. For k-nearest neighbors, mean-imputed nulls cause more than half of the PWRs and all of the PHWRs to be misclassified as BWRs. Additionally, 28.5% BWRs are misclassified as PWRs. For the zero-imputed nulls test set, there is a much larger correct BWR classification percentage, but also a much larger PWR misclassification percentage. The PHWR true positive percentage increases from

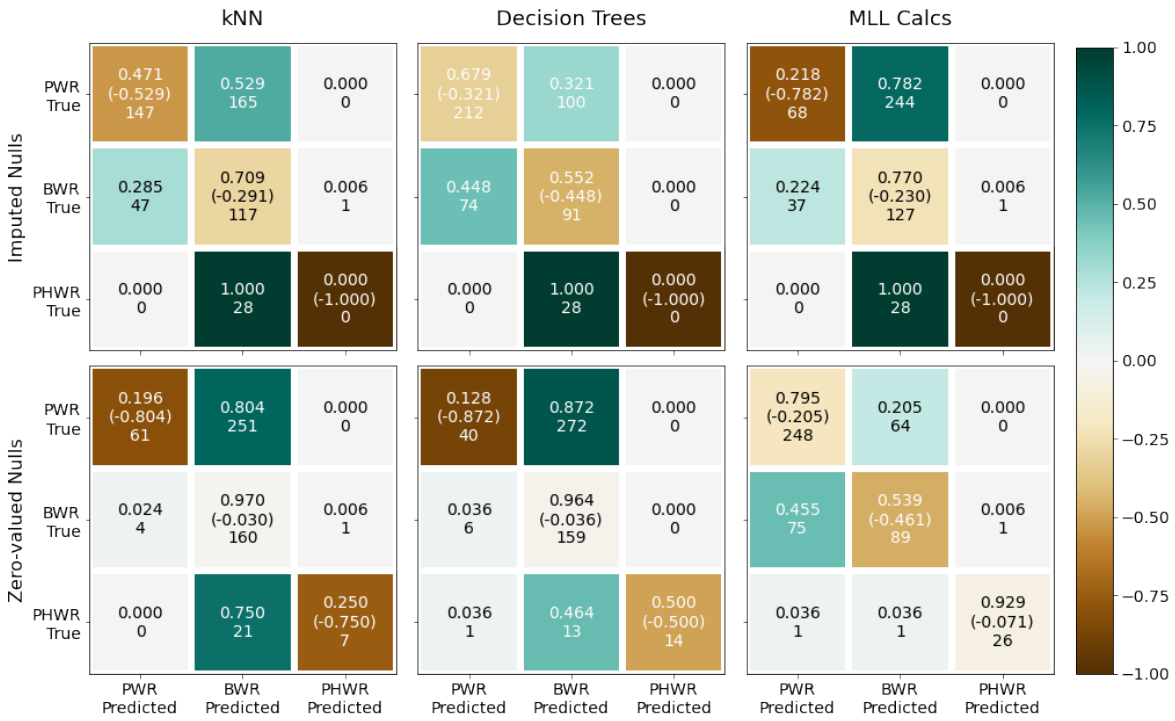


Figure 4.15: Confusion matrices of reactor type prediction for each algorithm using two missing entry techniques: imputation with mean values (top panel) and with zero (bottom panel).

0 to 25%. BWR (32% of test cases) and PHWR (5.5% of test cases) are the two minority classes in the database. Because they both have higher true positive fractions but there were overall fewer correct predictions (from a much higher PWR misclassification) from the mean-imputed nulls to the zero-imputed nulls, the accuracy decreased but the balanced accuracy increased.

Decision trees follows a similar pattern the the k-nearest neighbors example. The PWR misclassification increases from the mean-imputed nulls to the zero-imputed nulls test set in a similar manner, although the original PWR true positive fraction is higher (leading to a higher accuracy for the mean-imputed nulls case). As before, the BWR correct classification increases drastically from the mean-imputed nulls to zero-imputed nulls case. Additionally, the PHWR classification improvement from mean-imputed nulls to zero-imputed nulls is better for decision trees than for k-nearest neighbors.

Therefore, again, the accuracy decreased and the balanced accuracy increased. The larger improvement for the minority classes has led to the larger balanced accuracy improvement for decision trees.

In Figure 4.15, the confusion matrices for the MLL calculations tell a very different story than those for the scikit-learn algorithms. The only similarity is that using mean-imputed nulls causes all PHWRs to be classified as BWRs. PWRs are misclassified as BWRs at 78.2%, and BWRs are misclassified as PWRs at 22.4%. For the zero-imputed nulls test set, PHWRs and PWRs true positive percentages sharply improve to 92.9% and 79.5%, respectively. The true positive rate for BWRs, however, decreases from 77.0% to 53.9%. This is the opposite trend from both of the scikit-learn algorithms. Despite the misclassification increase for BWRs, both the accuracy score and balanced accuracy score increase for MLL calculations when moving from using mean-imputed null values to zero-imputed null values in the test set. The improvement in MLL classification is likely because the mean-imputed nulls test set hides information rather than removing it from consideration, which is what the zero-imputed nulls test set does.

All three algorithms using both test sets tend towards misclassifying PHWRs as BWRs (except for MLL calculations using zero-imputed null missing values). This is likely because BWRs comprise the majority of the training set (72%), and no matter how the missing measurements are handled there may be too little information to predict these well with most algorithms. For the two scikit-learn algorithms, the zero-imputed nulls test set predicts BWRs the majority of the time (despite there being 50% correct PHWR prediction for decision trees). This also is likely from BWR being the training set majority class, so with many nuclides measuring at zero, there are likely to be few good matches, and the majority class becomes the most likely prediction. This explanation is possibly applicable to the mean-imputed nulls test set as well, but the behavior pattern is less clear because the mean-imputed nulls give more information

for these algorithms than zero-imputed nulls (since the zero-imputed values cannot be removed from consideration), so a larger proportion of PWRs are being predicted properly.

4.4.2.2 Regression Cases

Next, the prediction of SFCOMPO test samples for the regression cases will be discussed. There is no time since irradiation value in the database, so only burnup and ^{235}U enrichment are discussed here. While the mean and median errors for burnup and enrichment prediction are listed in Tables 4.9 and 4.10, respectively, there are also box plots included for both absolute and relative errors in Figures 4.16 and 4.18, respectively. Box plots were chosen since they can provide a larger amount of information than just a mean or median value. The white triangles represent the mean error, and the white line in notched box is the median error. The box itself is the 25% (Q1) and 75% (Q3) quartiles at the bottom and top, respectively. The error bars or whiskers are meant to represent the spread of all errors minus the outliers. The bottom whisker reaches to $Q1 - 1.5(Q3 - Q1)$ and the top whisker reaches to $Q3 + 1.5(Q3 - Q1)$. Any values outside of the total whisker range, $4(Q3 - Q1)$, are considered outliers. [48] Lastly, there are plots of the true value versus the predicted value for both burnup and enrichment in Figures 4.17 and 4.19, respectively.

Burnup

The expected results that MLL will perform better with the zero-imputed nulls test set also holds true for the regression cases, as shown in Table 4.9. While the scikit-learn algorithms have a moderate increase in both the mean and median burnup errors from the mean-imputed nulls to the zero-imputed nulls, the MLL calculations have an order of magnitude decrease in error when moving in that same direction. Seeing this trend between Figures 4.16a and 4.16c is a little difficult due to the different ranges on the vertical axes, but the drastic improvement in MLL burnup error from the mean-imputed

nulls in Figure 4.16a to the zero-imputed nulls in Figure 4.16c is still visually clear. In the latter figure, there is one outlier for k-nearest neighbors and 39 for MLL calculations.

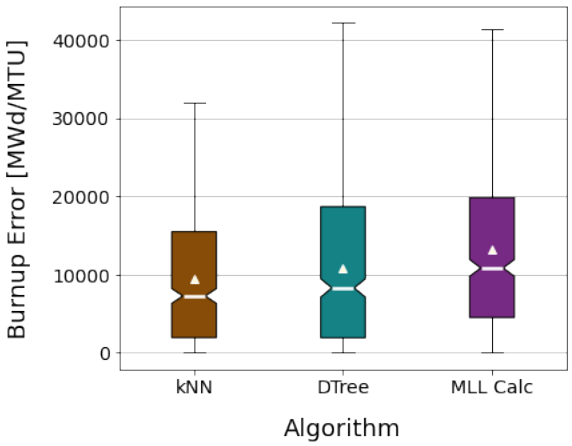
	Mean Errors $[GWd/MTU]$			Median Errors $[GWd/MTU]$		
Null Handling	kNN	DTree	MLL	kNN	DTree	MLL
Mean Imputation	9.43	10.89	13.17	7.26	8.28	10.84
Zero Imputation	14.88	15.18	3.53	11.47	8.79	1.70

Table 4.9: Mean and median errors for burnup prediction of the SFCOMPO test cases.

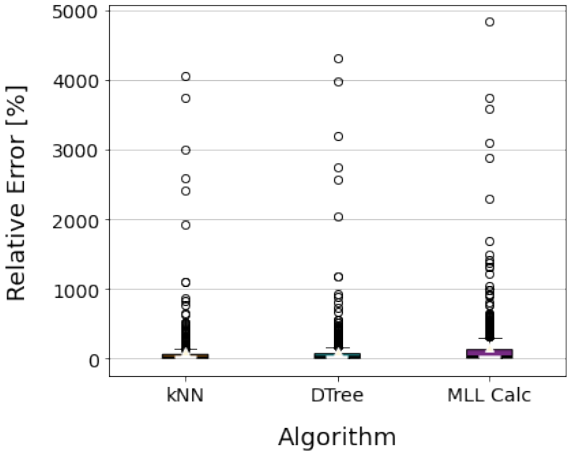
Although the mean and median errors are contained in the range of $1-15 GWd/MTU$, the large spread in burnup errors in Figures 4.16a and 4.16c for all three algorithms was broad enough to warrant an investigation into the range of relative errors, expressed as percent errors in Figures 4.16b and 4.16d. In Figure 4.16b there are 75, 72, and 73 outliers (all around 15% of the test database) for the k-nearest neighbors, decision trees, and MLL calculations, respectively. In Figure 4.16d there are 45 outliers for the MLL calculations.

The large ranges seen for the mean-imputed nulls test set in Figure 4.16b are because of large overpredictions of low burnups ($< 10\, GWd/MTU$), since a small number in the denominator will yield a high percentage error. A large subset of the low burnup cases are PHWRs. Their burnups are also unlikely to be predicted well because of the inability of PHWR reactors to be represented accurately with this methodology. Removing the PHWR reactors from the results removes all outliers with percentage errors larger than 1750%. That is still a very high relative error, but there are other low burnup cases in the database. Removing PHWRs does not significantly alter the zero-imputed nulls results in Figure 4.16d.

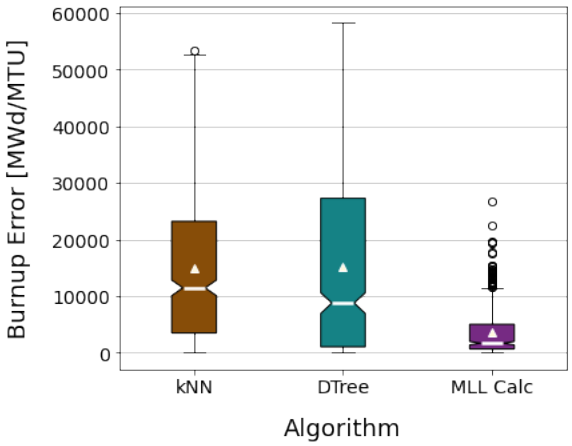
The range of percentage errors for the zero-imputed nulls test set in Figure 4.16d tells a different story. There is only one case (an MLL outlier) that is above 100% error. While the absolute errors for the scikit-learn algorithms in Figure 4.16c span a



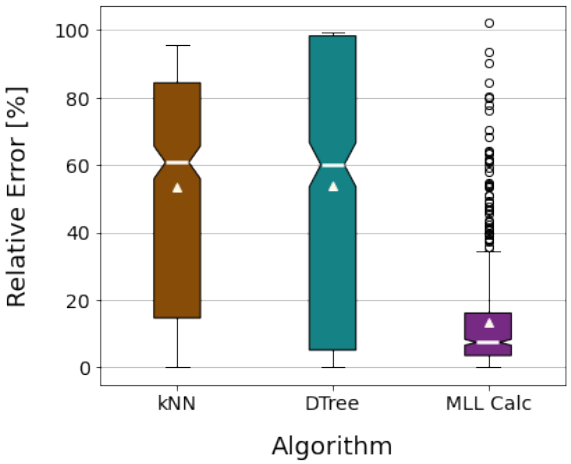
(a) Box plots of burnup errors using mean-imputed null values.



(b) Box plots of burnup percentage errors using mean-imputed null values.



(c) Box plots of burnup errors using zero-imputed null values.

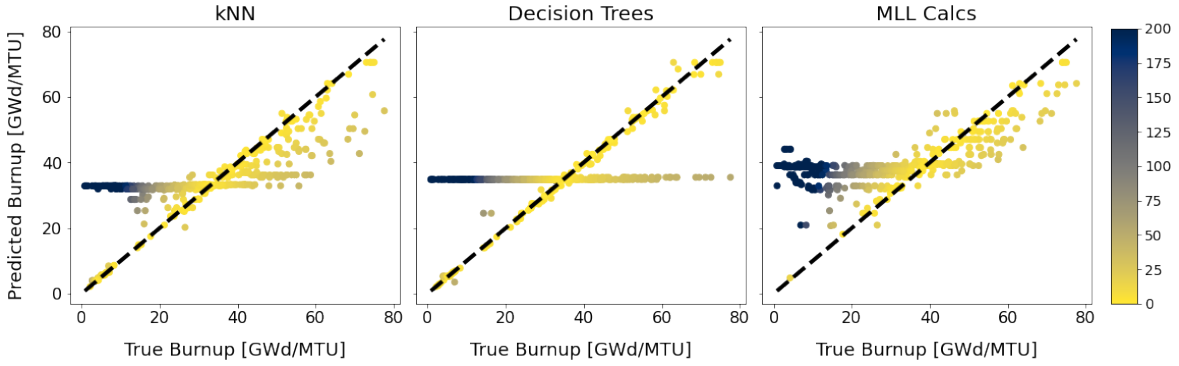


(d) Box plots of burnup percentage errors using zero-imputed null values.

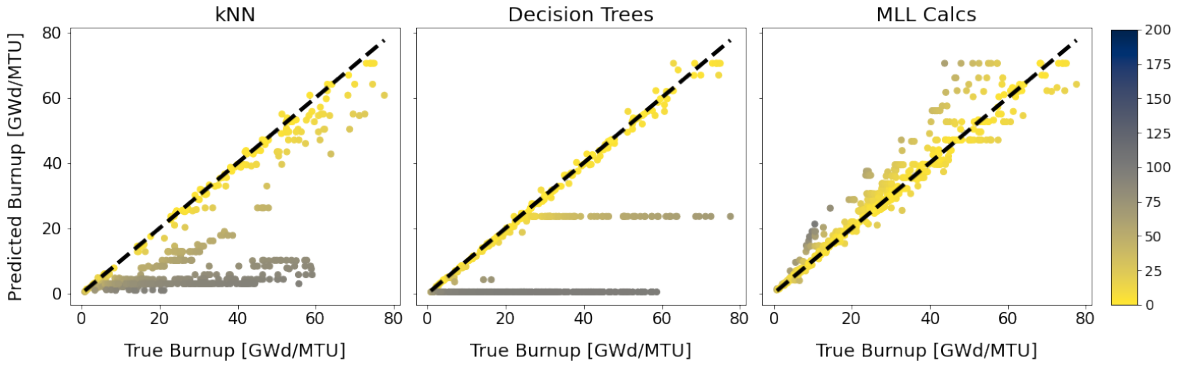
Figure 4.16: Box plots of burnup prediction errors and percentage errors for each algorithm using two missing entry techniques: imputation with mean values and with zero.

larger range than their counterparts in Figure 4.16a, their relative errors remain within $0 - 100\%$. The only case that predicts the burnup well is the MLL method with the zero-imputed missing values treatment of the SFCOMPO test set, but about 8% of the test cases are outliers. If the best-case median error of $1.7\,GWd/MTU$ in Table 4.9 were to also correspond to a lower relative error, then that would be an acceptable result.

However, while the MLL calculations have a percentage error below 20% at the 75% quartile, the non-outlier data reaches almost 40% and the 8% of the data reaches 100%.



(a) True versus predicted burnup using mean-imputed null values.



(b) True versus predicted burnup using zero-imputed null values.

Figure 4.17: True versus predicted burnup for each algorithm using two missing entry techniques for SFCOMPO: imputation with mean values and with zero.

To better understand the high errors in Figure 4.16 and especially the relative error outliers, Figure 4.17 presents the plots of the true burnup versus the predicted burnup for each test sample in SFCOMPO for each algorithm and both imputation techniques. The colorbar is the percentage error and was chosen with the range up to 200%. This is in order to show the difference between the large errors below the diagonal line generally having a maximum error of 100%, whereas the mispredicted low-burnup cases have errors exceeding 200%. First, Figure 4.17a shows that all of the large errors are centered around a certain predicted burnup range, $30 - 40\,GWd/MTU$.

This is happening because mean imputation technique is being applied to a data set where the measurements likely only exist for a small range of burnup values, and each test sample is likely to have many missing values. The very large relative errors from Figure 4.16b ($200\% - 5000\%$) are clustered in the same place for all three algorithms. For Figure 4.17b, the large-error predictions are shown clustered towards the bottom. This makes sense for the scikit-learn algorithms since the zero measurements are easily interpreted as low burnup. Of course, this is not the case for the MLL calculations since the zero values are filtered.

Overall, the absolute errors in Table 4.9 tell a much more encouraging story than the box plots in Figure 4.16, so investigating beyond the mean and median absolute errors was necessary to show the real picture of this unique testing scenario. The additional details provided by directly plotting the true versus predicted burnup in Figure 4.17 is crucial in understanding how the null-value handling methods impacted the results.

^{235}U Enrichment

For both reactor type classification and burnup regression, the zero-imputed nulls test set predicted by MLL calculations far outperform all other algorithm/test set scenarios. However, the enrichment regression results break this trend. Table 4.10 shows that decision trees outperform the other methods, and furthermore, there isn't a large difference in performance between the two test sets, especially seeing that the median absolute error is the same for both test sets. The other two algorithms follow their previous behavior: moving from mean-imputed nulls to zero-imputed nulls, k-nearest neighbors has worse performance and MLL calculations has better performance.

The mean and median absolute errors are also visible with more statistical information in the box plots in Figure 4.18. The outliers for decision trees and MLL calculations are 30 and 16 for the mean-imputed nulls in Figure 4.18a, respectively. So although decision trees provides a typically low absolute error, the outliers reach nearly as far as the

	Mean Errors [% ^{235}U]			Median Errors [% ^{235}U]		
Null Handling	kNN	DTree	MLL	kNN	DTree	MLL
Mean Imputation	0.72	0.31	1.25	0.50	0.22	1.13
Zero Imputation	1.67	0.36	0.49	2.02	0.22	0.35

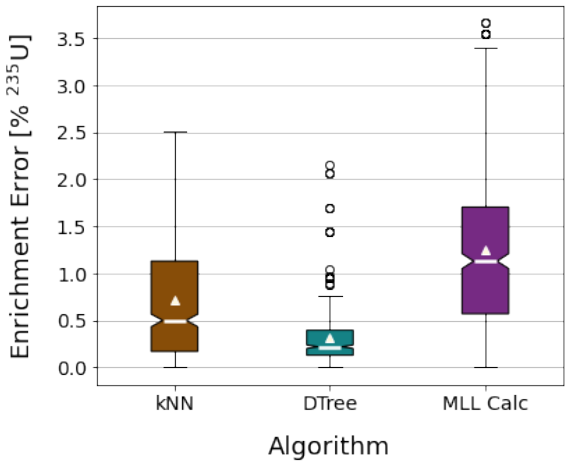
Table 4.10: Mean and median errors for enrichment prediction of the SFCOMPO test cases.

spread of k-nearest neighbors. The number of outliers for the zero-imputed nulls results in Figure 4.18c are 45 and 16 for decision trees and MLL calculations, respectively. The spread of the outliers for these algorithms is similar to the previous figure, but k-nearest neighbors has a larger spread of absolute error.

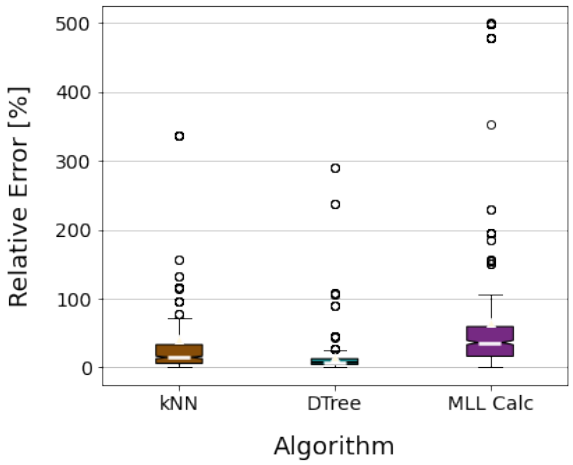
Again, a look at the relative errors gives a different sense of these results. Figures 4.18b and 4.18d present the percent error statistics for the mean-imputed nulls and zero-imputed nulls test sets, respectively. In Figure 4.18b there are 57, 39, and 49 outliers for the k-nearest neighbors, decision trees, and MLL calculations, respectively. In Figure 4.18d there are 44 and 23 outliers for the decision trees and MLL calculations, respectively.

As with the burnup regression, the high percentage errors are caused by a large overprediction of enrichments that are of low value. All of the PHWRs fit into this category, and removing them from the results removes the outliers with the highest errors from the mean-imputed nulls results in Figure 4.18b, leaving a maximum of 250% enrichment error. This is still a high maximum error because there are other low enrichments not belonging to the PHWR class that remain overpredicted. As with the burnup, removing PHWRs does not significantly alter the zero-imputed nulls results in Figure 4.18d.

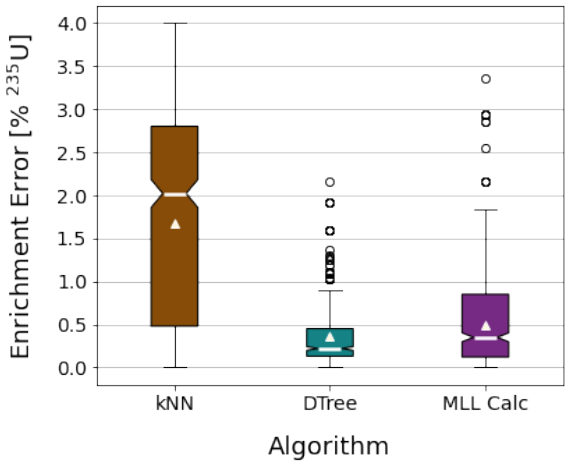
Unlike the case with burnup, the decision trees algorithm gives a better performance that is null handling-independent than the other algorithm/test set scenarios. Although



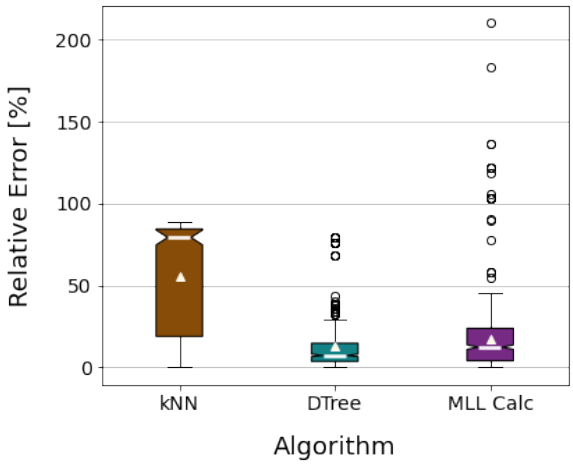
(a) Box plots of enrichment errors using mean-imputed null values.



(b) Box plots of enrichment percentage errors using mean-imputed null values.



(c) Box plots of enrichment errors using zero-imputed null values.

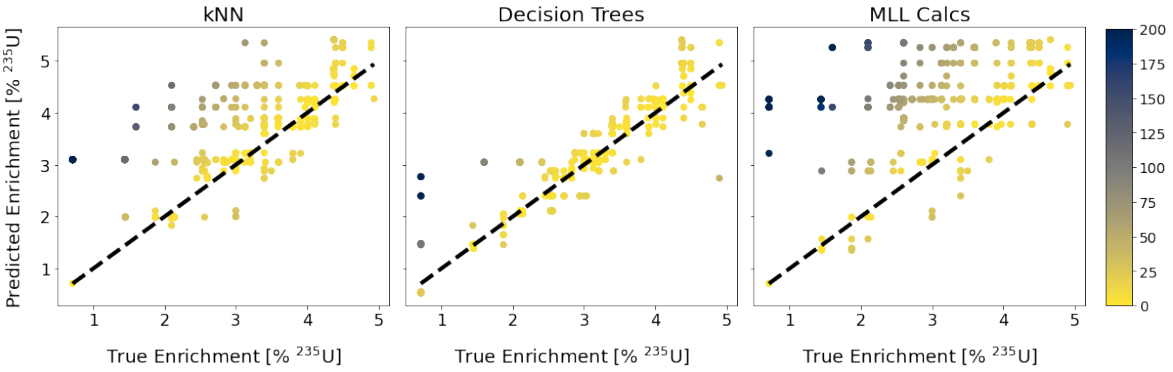


(d) Box plots of enrichment percentage errors using zero-imputed null values.

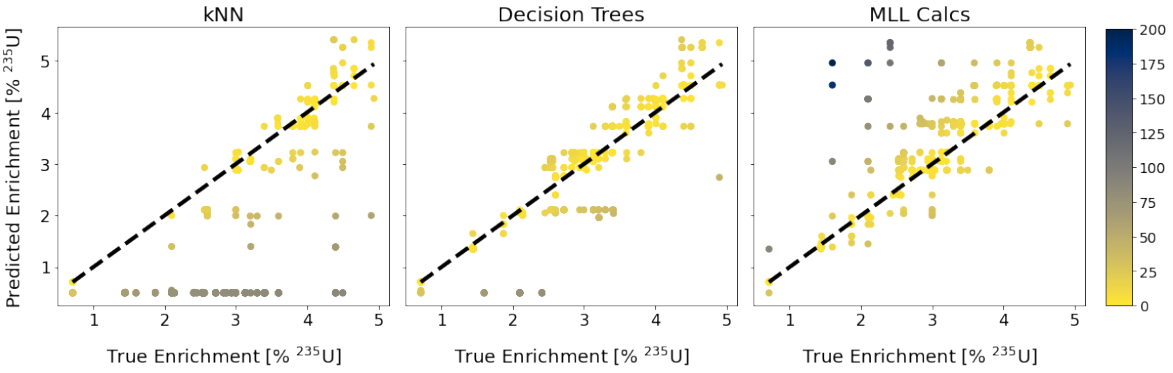
Figure 4.18: Box plots of enrichment prediction errors and percentage errors for each algorithm using two missing entry techniques: imputation with mean values and with zero.

there was a slightly worse mean absolute error for the zero-imputed nulls test set (the median absolute error was the same), the relative error remains below 100% for all outliers, as seen in Figure 4.18d. This makes the decision trees algorithm with the zero-imputed nulls test set the best performing case for enrichment regression. It is an unexpected result to have either one of the scikit algorithms outperform MLL

calculations for the zero-imputed nulls test set, since they tended to have lower errors using the mean-imputed nulls test set. But decision trees are capable of ignoring certain features if they do not lower the node impurity (recall Equations 3.3 and 3.4), so it is not an unexpected result without an explanation.



(a) True versus predicted enrichment using mean-imputed null values.



(b) True versus predicted enrichment using zero-imputed null values.

Figure 4.19: True versus predicted enrichment for each algorithm using two missing entry techniques for SFCOMPO: imputation with mean values and with zero.

To better understand the high errors in Figure 4.18 and especially the relative error outliers, Figure 4.19 presents the plots of the true ^{235}U enrichment versus the predicted ^{235}U enrichment for each test sample in SFCOMPO for each algorithm and both imputation techniques. The colorbar is the percentage error and was chosen with the range up to 200%. This is in order to show the difference between the large errors below the diagonal line generally having a maximum error of 100%, whereas the errors

above the diagonal line can exceed 200%. First, Figure 4.19a does not follow the same pattern as its burnup counterpart (Figure 4.17a) where the predictions are clustered at a certain level due to the mean imputation. There is still a general trend of lower levels of enrichment being over-predicted, which are the large relative error cases. This is not happening because the missing values are not strong enrichment indicators, because there is a clear effect when they are imputed with zero, as shown in Figure 4.19b. Here, the large-error predictions are shown clustered towards the bottom for the scikit-learn algorithms. As with burnup (Figure 4.17b), this is not the case for the MLL calculations since the zero values are filtered.

Overall, as with the case with burnup, the mean and median absolute errors in Table 4.10 tell a much more encouraging story than the box plots in Figure 4.18. Looking at the spread of absolute errors and the large relative errors gives the sense that the enrichment predictions are quite poor as a whole. This does not mean it is hopeless; the decision tree approach here warrants more investigation since the spread of errors in Figure 4.18c is under 1.0% ^{235}U. The additional details provided by directly plotting the true versus predicted enrichment in Figure 4.19 is helpful in understanding how the null-value handling methods impacted the results.

4.5 Summary

This chapter covers the details of the methodology in four main sections: training set simulations, information reduction, statistical methods implementation, and performance evaluation. This approach mimics the situation where there is a full set of well-measured nuclides, some of which require destructive techniques to measure.

First, in Section 4.1, the training data is simulated. The features are an array of nuclide measurements, and the prediction parameters are the simulation inputs: reactor type, burnup, ^{235}U enrichment, and time since irradiation. Given the chosen inputs, the

makeup of 4.5×10^5 SNF entries are simulated using ORIGEN [26, 27, 28].

Second, in Section 4.2, information reduction on the training set is carried out using randomly injected uniform error. The random error is used to study the robustness of the methodology to artificial noise in the feature set, which is comprised of 29 nuclide masses. The introduced training set error is increased up to 20%.

Third, in Section 4.3, three algorithms, k-nearest neighbors, decision trees, and MLL calculations, are used to train models to predict the four reactor parameters of interest. The first two are algorithms implemented using the scikit-learn python ML toolkit, and MLL is implemented in python leveraging SciPy and NumPy for fast likelihood calculations [29, 43, 44]. For the scikit algorithms, the hyperparameters governing model complexity were optimized to minimize the prediction errors.

Fourth, in Section 4.4, the prediction errors are evaluated to draw conclusions about the capability of the chosen statistical methods to inform SNF attribution with increasingly less precise material measurements. For Section 4.4.1, the prediction performance is measured by using the training set to provide testing samples. The entire training set is tested at some point for all three algorithms. Sections 4.4.1.1 and 4.4.1.2 show the results of information reduction by injecting noise into the training set (randomly applied uniform error). Next, the impacts of training set size are evaluated to understand the effects of model generalization in Section 4.4.1.3. After this, the impacts of having prior knowledge of the reactor type on the quality of prediction of the regression cases are studied in Section 4.4.1.4. Last, the external testing set, the SFCOMPO database, is used to test this methodology in Section 4.4.2.

Sections 4.4.1.1 and 4.4.1.2 cover the impacts of increasing training set error on the four prediction parameters. The reactor type classification shows a tendency for PWRs to be misclassified as BWR, and less so vice versa. The few misclassified PHWRs also tend towards BWR. Neither of these findings are surprising because BWR is the

majority class. A more balanced training set and/or better simulation fidelity would be required to improve these misclassifications. Burnup, ^{235}U enrichment, and time since irradiation relative prediction errors are all above -6% (if the anomalous k-nearest neighbors results for time since irradiation are excluded), even at 20% training set error.

For all four prediction categories, the MLL calculations method is the most resilient to introduced error, followed by decision trees then k-nearest neighbors. Interestingly, at the 1% training set error level, MLL calculations tend to do slightly worse than the scikit-learn algorithms. The behaviors of each algorithm's performance degradation for reactor type, burnup, and enrichment all behave in a similar fashion. But for time since irradiation, the k-nearest neighbors prediction performance has a drastic drop that starts with even 1% introduced training set error. *While it is difficult to draw a line and say which algorithm's performance is acceptable or "good enough" (especially considering the relative errors are all under 6%), the prediction performances at 20% training set error are used as a minimum baseline for future work.*

In Section 4.4.1.3, the impacts of fewer training set entries are implemented to investigate how the algorithms each generalize to unseen samples. Using the training set that has 5% error, it was reduced in four steps from its full size, the lowest size being 20% of the full training set. Although the MLL calculations perform above the scikit-learn algorithms at most sizes, the data points at 20% training set size show MLL below one or both of the scikit-learn algorithms. *The takeaway is that while this training set is more than large enough to achieve good performance from a high variance approach, the performance may not hold with a different training set design.*

Section 4.4.1.4 shows whether the burnup, enrichment, and time since irradiation predictions benefit from having the prior knowledge of reactor type. The largest improvement is for the PHWR (only 1.5% of training set) regression cases for the scikit-learn algorithms, whereas there is no improvement for MLL calculations. Because the

BWR class dominates the training set, there is modest improvement in PWR regression cases as well for the scikit-learn algorithms. *These results indicate regression cases linked to PHWRs may not perform well.*

Last in this chapter is Section 4.4.2, where the performance of this methodology using real world test cases of nuclide concentration measurements via the SFCOMPO database is demonstrated. While the training set design spans the label space that exists in SFCOMPO, there are many missing measurements from the features in the training set, which is comprised of 29 nuclide masses. Because of an imbalanced training set and the method used to handle the null values, the reactor type classification results are mostly poor (except for the MLL calculations with the zero-imputed nulls testing set with a balanced accuracy score of 0.63), which extends to the regression cases. It is again true that investigating relative error statistics alongside absolute error statistics provides a fuller picture of the regression of the SFCOMPO database entries. The MLL calculations perform the best for burnup and the decision trees perform the best for ^{235}U enrichment (both with the zero-imputed nulls testing set), but many of the prediction errors are still large. An investigation into the true versus predicted values (especially for burnup) shows that *the mean-imputed nulls database is clearly not worth further investigation, but much improvement could be made to the zero-imputed nulls method.*

It is possible that because most of the 505 entries contain many of the of-interest uranium and plutonium isotopes, that this testing set should be used only in a study limited to those isotopes. They are known to provide good discrimination on their own [14, 15, 16, 17, 18, 20]. There are two areas for improvement: new approaches to testing set features (using only plutonium and uranium isotopes, different approaches to null value imputation) and improving the simulation fidelity of the training set. *The SFCOMPO database has challenges to being used as an external testing set with this methodology, but could be improved with more consideration.*

5 REACTOR PARAMETER PREDICTION USING PROCESSED GAMMA SPECTRA

This chapter covers the parameter prediction workflow using processed spectra as the input features. The methodology from Chapter 4 is reapplied with new implementations surrounding the training set moving beyond full knowledge of nuclide masses to degraded knowledge of processed gamma spectra. This is described in four sections that correspond to the four steps summarized in Figure 5.1.

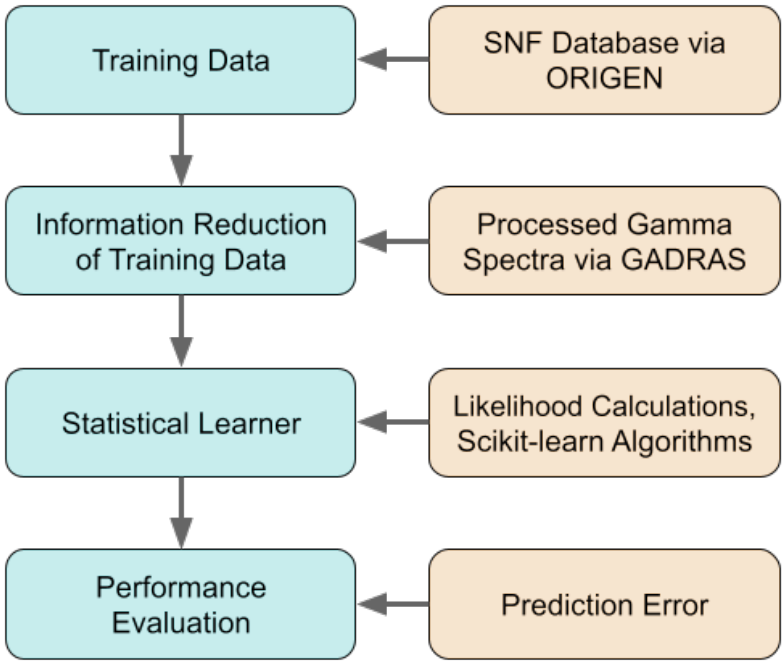


Figure 5.1: Flowchart of the experimental methodology and the way each step is being implemented.

The full description of the simulations and training labels can be found in Section 4.1. Section 5.1 provides the updates on the features used in the training set. This results in a set of spent nuclear fuel (SNF) observations with the same known reactor operation parameters, i.e., labels that are to be predicted, but new nuclide features.

Next, the information reduction step is covered in Section 5.2. Here, computational

gamma spectra are created via the GAmma Detector Response and Analysis Software (GADRAS) tool; six detectors with decreasing energy resolution were chosen. The approach for processing the spectra from these detectors is outlined here as well.

The increasingly less-precise training data sets are input to a statistical learner for the next step: training models. These use the features (from processed spectra) and labels (reactor parameters/simulation inputs) in the training data sets to formulate a model. This is introduced in Section 4.3, and the updates for this experiment are in Section 5.3.

Lastly, the algorithms must be evaluated for their prediction performance when given test samples (i.e., a new SNF measurement that has no labels according the to algorithm). This follows much of what was introduced in Section 4.4, and the updates to the approach is shown in Section 5.4.

5.1 Training Data Simulation

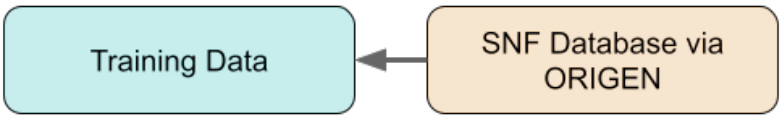


Figure 5.2: First portion of the flowchart from Figure 5.1 being described in this section.

In the second experiment, activities of radionuclides are necessary to calculate gamma spectra from these values. The Oak Ridge Isotope GENeration (ORIGEN) simulations and their inputs are the same as Section 4.1, but the outputs tracked are a different set of nuclides, measured in Ci, or $Curies$. In this experiment, the full knowledge scenario is the set of 32 nuclide activities found in Table 5.1. The simulation parameters that are being predicted are the same:

1. The classification of the **reactor type**: pressurized water reactor (PWR), boiling water reactor (BWR), or pressurized heavy water reactor (PHWR).

2. The **burnup**: MWd/MTU (or GWd/MTU), mega (or giga) watt-days per metric ton of initial uranium.

3. The uranium-235 (^{235}U) **enrichment**: % ^{235}U.

4. The **time since irradiation**, or cooling time: *days* (or *years*).

^{227}Ac	**^{241}Am**	**^{243}Am**	^{133}Ba	^{249}Cf	^{252}Cf	^{243}Cm	**^{244}Cm**
^{245}Cm	**^{134}Cs**	**^{137}Cs**	^{152}Eu	**^{154}Eu**	^{166m}Ho	^{85}Kr	^{94}Nb
^{236}Np	**^{237}Np**	^{231}Pa	^{146}Pm	^{236}Pu	**^{238}Pu**	**^{239}Pu**	**^{240}Pu**
^{226}Ra	^{125}Sb	^{228}Th	^{229}Th	^{232}U	^{233}U	**^{234}U**	**^{235}U**

Table 5.1: Set of features saved for the second experiment, nuclide activities measured in Ci. The bold nuclide activities overlap with the nuclides in Table 4.3.

The second feature set with 32 nuclide activities listed in Table 5.1 was designed with the following reasons in mind. First, nuclide activities are the most straightforward units to use for application to the detector response function (DRF) in the GADRAS tool for the second experiment. This process is used to obtain gamma spectra for each SNF entry in the database, which is detailed in 5.2. Second, these specific nuclides were chosen because they remained after four steps of filtering:

1. They exist in the 196-long radionuclide list in GADRAS.

2. They have an activity above $1 \times 10^{-7}\,Ci$ (cutoff chosen to filter out nuclides that are unlikely to produce gamma energy peaks).

3. They have a half-life longer than 1 *year* (cutoff chosen based on maximum time since irradiation of 16 *years*).

4. They have at least one gamma energy line above 200 keV (cutoff chosen based on low-energy gamma energy peaks being difficult to discern in some detectors).

This one initial training set will undergo several transformations to become six training sets (one for each detector) for each spectra-processing approach (three for each set of detectors). This is described next in Section 5.2.

5.2 Information Reduction

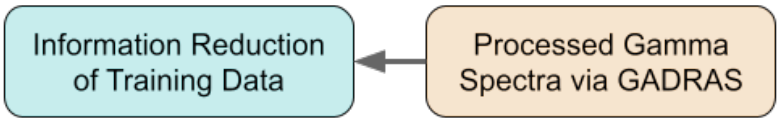


Figure 5.3: Second portion of the flowchart from Figure 5.1 being described in this section.

The overall goal of this project is to determine how much information to what quality is needed to train a machine learning (ML) model that can provide SNF attribution by correctly predicting the reactor type, burnup, ^{235}U enrichment, and time since irradiation. In this section, the information quality is treated as the energy resolution of gamma spectra from several detectors. This is because field-deployable detectors are of interest.

This process is outlined here for the second experiment, in which a gamma spectrum is computed for each sample in the database from the nuclide activities in Section 5.1. The GADRAS code [49] developed at Sandia National Laboratories will provide computational gamma spectra. There are three steps to this process: inputting the radionuclide activities into a DRF to compute gamma spectra, processing the gamma spectra into a smaller feature set, and applying a statistical counting error to those features.

Step 1: Computational Gamma Detection

The first step is obtaining a gamma spectrum for every SNF entry in the database; this is done using GADRAS. The GADRAS tool includes many capabilities related to radiation detection and spectra analysis; its predominant use is related to the simulation of the detection of gamma rays and neutrons from user-defined sources and detector configurations. A combined first principles and empirical modeling approach based on interaction cross sections and radiation scattering inform the DRF code to calculate typical gamma detector spectrum features, e.g., photopeaks and the Compton continuum. Although new detectors can be created and calibrated within the software, this work employs the use of pre-calibrated detectors. Thus, given a source, which is a list of radionuclides and their activities in this case, GADRAS applies a DRF from a given detector configuration to the gamma energy lines from these radionuclides and models the spectrum. [49]

The nuclide activity data requires some processing to be used in this way. The activities that come from the ORIGEN simulations are based on there being 1 MT of initial uranium-based fuel. Not only is this quantity an unlikely amount to be smuggled, it would overwhelm a detector at the calibrated source-detector distances in GADRAS. Therefore, the material (and resulting nuclide activities) are scaled to be 1 g of SNF.

For the GADRAS calculations, the primary input is a nuclide activity vector as the source, and the output is an array of energy bins (measured in keV) and the counts per energy bin as the spectrum. The sources are provided without any background; this is because any spectrum would undergo background subtraction before further analysis. Additionally, the nuclides are pre-decayed in ORIGEN to correspond to various cooling times, but a source age must be provided to GADRAS as a non-zero value (otherwise, important decay transitions are missed). Various source times (10 sec, 30 sec, 1 min, 5 min, 10 min, 20 min, 30 min, 1 hr, 2 hr) were tested for two samples, and both a

qualitative visual analysis of the major photopeaks plus a study of the total counts leveling off was used to choose a nonzero age for the material. At a source age of 20 *minutes*, the expected peaks are visible and the total counts had reached a value similar to the longer ages tested.

The other input information is related to the detector configuration. User-chosen variables are the source-detector distance [*cm*], height of source-detector setup from nearest surface [*cm*], the live time [*s*], and the number of channels for the detector. The detector configuration file in GADRAS contains much more information, however. In it, there are a number of parameters from calibration results, detector geometry, information about scattering, and shielding information (shielding is not considered in this work).

Detector	**% FWHM @ 661 keV**	**Distance (cm)**	**Height (cm)**	**Live Time (s)**	**Num Channels**
In-Lab HPGe	0.21	100.0	84.0	600	8192
Portable HPGe	0.29	100.0	100.0	600	8192
CZT	1.20	100.0	100.0	600	1024
SrI_2	2.94	100.0	100.0	600	1024
$LaBr_3$	3.63	213.0	84.5	2400	1024
NaI	7.74	213.0	85.4	2400	1024

Table 5.2: Select details of 6 detector setups used to obtain gamma spectra-based training databases.

Training databases were created for the six detectors outlined in Table 5.2. They were chosen to compare the highest energy resolution detector, a lab-based high-purity germanium (HPGe), against the rest, in order of decreasing energy resolution: portable HPGe, cadmium zinc telluride (CZT), strontium iodide (SrI_2), lanthanum bromide ($LaBr_3$), and sodium iodide (NaI) detectors. This is displayed in the table by including the full width at half maximum (FWHM) of the 661 keV peak for ^{137}Cs. GADRAS is used to create a spectrum for every SNF entry in the 32 nuclide activity training

database using these six detector configurations. At this point, there are six versions of the original database, one for each detector. The database entries are each a full gamma spectrum of a given SNF sample for the detector setup in that training set.

Step 2: Processing Gamma Spectra

The second step covers the processing of the gamma spectrum generated for each SNF entry into a training set. It is not computationally prudent to use full gamma spectra for training and testing, since the spectra returned have 1024 or 8192 bins, and machine learning algorithms are not designed to handle thousands of features. Thus, these spectra are processed into fewer features.

There is much ongoing work on the topic of attributing SNF with gamma detection using targeted or advanced measurement techniques [50, 51, 52] and using innovative spectra evaluation and radioisotope identification methods [53, 54, 55, 56, 57, 58, 59, 60]. Since this is a topic of active research and the approaches are heavily detector-dependent, a simple processing approach is developed here. Since all entries in a given training set are background-subtracted spectra using the same detector setup and calibration for the same measurement duration, each photopeak can be directly compared to other photopeaks in the training set. This is accomplished by comparing them via an area under the curve: placing an energy window on a peak, and summing the counts of the bins within that energy window.

There are two main design choices here: the width of the energy windows and the number of energy windows to include. First, the energy window width is a value that is fixed for each detector prior to processing. The different energy window widths are listed in Table 5.3. They are chosen manually; the spectra were plotted and different values were tested and visually analyzed to be sure the windows were encompassing the peaks. This was preferable to some linear function based on the detector energy resolution because, e.g., the $LaBr_3$ and NaI detectors have the same human-chosen

window but the energy resolutions are different (see Table 5.2).

Detector	**Energy Window Size [keV]**	**# of Energy Windows**		
		Auto	Short	Long
In-Lab HPGe	2	206	42	151
Portable HPGe	3	120	42	151
CZT	8	30	42	151
SrI_2	10	17	42	151
$LaBr_3$	12	19	42	151
NaI	12	9	42	151

Table 5.3: Energy window sizes and list lengths for 6 detector setups used to process the gamma spectra-based training databases.

The second design decision is the number of energy windows to include. Table 5.3 lists three energy window list length columns, *Auto*, *Short*, and *Long*. These correspond to different processed training sets that have a different number of energy windows included. There are two approaches taken: a nuclear physics-based method that generates an energy window list based on the gamma energies expected to be detected (short and long), and an automatic peak search of a manually chosen gamma spectrum in the full gamma spectra training database (auto).

The first energy window list method provides the short and long lists in Table 5.3; the length of these lists are the same for all detectors because they are based on the gamma energies most likely to be detected, which is independent of the detector quality. To obtain these lists, the expected number of decays of each gamma energy is calculated based on the activities of the 32 tracked nuclides using the Python for Nuclear Engineering toolkit [61] from a reference sample in the training set. This reference sample emerged as the sample of choice because it contains the superset of gamma energies of all tested samples, of which there were nine (three for each reactor type).

After the expected number of decays for each gamma energy for all 32 nuclides are

calculated, an arbitrary minimum number of decays is selected to filter out the gamma energies that are unlikely to produce counts high enough to be detected. The first arbitrary minimum number of decays is set at 5×10^8 decays, and the long list of 151 gamma energies that remain above this cutoff is created. A list of length 151 is likely to contain many features that do not contribute discrimination to the models, and thus they may add noise to the training set, so a shorter list is needed. So, a higher arbitrary minimum of 5×10^{10} decays is chosen; this threshold creates the short list of 42 gamma energies.

The short and long lists of gamma energies correspond to the nuclides in Table 5.4. The 12 nuclides listed come from the long list, and the subset of seven bold nuclides come from the short list. To determine a "full knowledge" scenario for the detector-based training sets, two training sets are also created with the 7- and 12-nuclide activity lists. It should be noted that several (three for each reactor type) training set entries were selected based on sampling evenly throughout the training set parameters, with the intention that there would be a set of gamma energies comprised from multiple entries. However, one sample emerged as a superset of the others. This sample is thus chosen for the second method, discussed next.

^{241}Am	**^{243}Am**	^{243}Cm
^{244}Cm	^{245}Cm	**^{134}Cs**
^{137}Cs	^{152}Eu	**^{154}Eu**
^{85}Kr	^{238}Pu	**^{125}Sb**

Table 5.4: Nuclides that are represented by the gamma energy lines in the energy lists. The entire set of 12 nuclides belongs to the long list, and the 7 bold nuclides belong to the short list.

The column denoted as *Auto* in Table 5.3 is obtained by a physics-free approach. It is based on a peak search of a spectrum in the full gamma spectra training database. The previously mentioned sample is selected for all six detectors, and a peak searching

algorithm implemented in python using the SciPy toolkit [43] is applied. Using the peak search on the six different spectra for the same sample, the resulting number of energy windows for each detector is in Table 5.3.

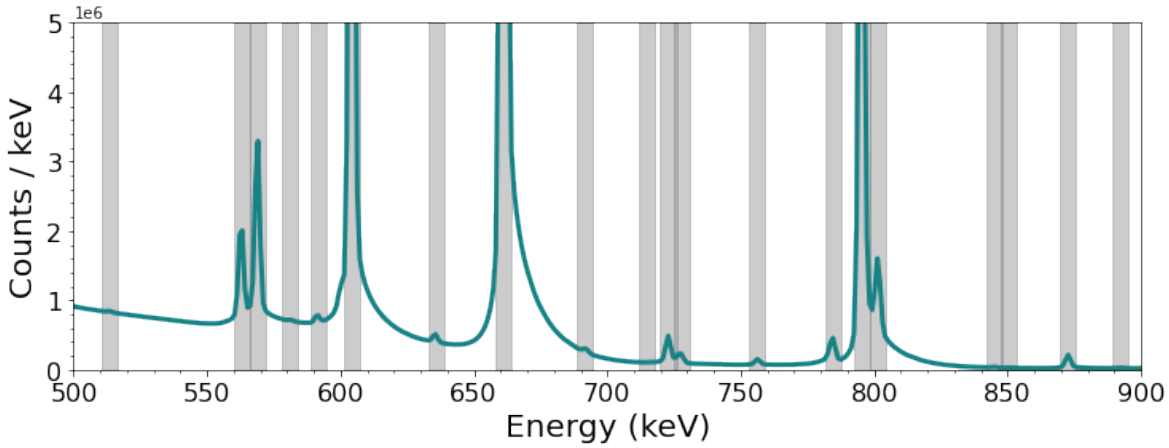


Figure 5.4: Slice of an example gamma spectrum in one of the training databases showing the windows over the gamma energy peaks. This is the portable HPGe with the auto energy window list and the reference sample's spectrum.

After the three energy window lists are created, the full gamma spectra database of a given detector are processed into three training sets, one for each list. Next, as previously mentioned, the energy window width for each detector is used to sum the binned counts for each energy window list entry. This is visualized in Figure 5.4, where a portion of a portable HPGe spectrum is shown with the $\pm 3\, keV$ windows from the auto energy window list. Three training sets are created for each detector, resulting in 18 detector-based processed gamma spectra training sets.

Step 3: Apply Statistical Counting Error

The third step involves the inclusion of the counting error for the summed energy windows. This is quite simple mathematically speaking, as statistical counting error of n counts is $\sqrt{n}$. As in Section 4.2, this error gets applied in the same way for the scikit-learn algorithms, where the uniform error is applied randomly within the range $[x_i - \sqrt{x_i}, x_i + \sqrt{x_i}]$ for each summed energy window x_i. For the maximum log-likelihood

(MLL) calculations, Equation 4.1 is used, where $\sigma_i = \sqrt{x_i}$.

In summary, there are three steps taken to arrive at training databases based on gamma spectra from six chosen detectors. Unlike the simple increase in training set error applied in Chapter 4, each step of the process adds an additional layer of reduced information quality, which are listed here:

1. Instead of "perfect" radionuclide activity knowledge, they are being measured by a gamma detector.

2. The processing of the gamma spectra can be highly variable.

3. The $\sqrt{n}$ error of counts-based detection is included.

5.3 Statistical Learning Implementation

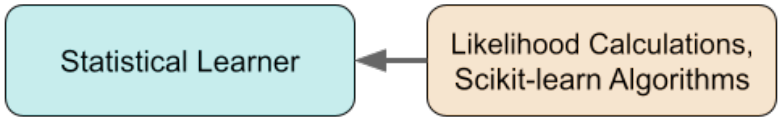


Figure 5.5: Third portion of the flowchart from Figure 5.1 being described in this section.

The chosen algorithms (k-nearest neighbors, decision trees, and MLL calculations) are introduced in Section 3.2.1 and their implementation details are in Section 4.3. This section will therefore only cover the implementation differences from the previous work. The MLL calculations are implemented identically to Chapter 4. However, the scikit-learn algorithms in this experiment did undergo a new round of hyperparameter optimization.

The full list of 22 training sets that undergo training and prediction are as follows:

- 1 set of nuclide masses (29 nuclide masses from Chapter 4)

- 3 sets of nuclide activities
  - 32 nuclide activities (full knowledge scenario for all nuclides, whether or not they are present in a quantity able to be detected)
  - 12 nuclide activities (full knowledge for long energy windows list)
  - 7 nuclide activities (full knowledge for short energy windows list)
- 18 detector-based processed spectra sets
  - Auto energy windows lists applied to six detectors (lab-based HPGe, portable HPGe, CZT, SrI_2, $LaBr_3$, NaI)
  - Short energy windows lists applied to six detectors above
  - Long energy windows lists applied to six detectors above

Table 5.5 lists the hyperparameter optimization results for the 22 training sets. Because the 29 nuclide mass training set is included in this chapter for comparison, its optimization results are also listed here. The number of features for decision trees are not limited because the test runs for optimization provided highly variable results. The only case where this is not true is with the auto energy windows list for the lab-based HPGe with a length of 206; the maximum features for this one case are limited to 150. Instead, optimization was carried out only on the maximum depth for decision trees with keeping the full length of features. This is an area that could undergo deeper exploration than what occurs in this work, since the training sets with large feature sets can become overfit.

The optimization took place in two rounds, where the first round had a coarser grid of k for k-nearest neighbors and maximum depth for decision trees and the second round had a finer grid of parameters. The 7 & 12 nuclide activity training sets were optimized separately but contain averages of the two results for the maximum depth,

Training Set Description	**Prediction Parameter**	***k* (N neighbors)**	**Max Depth**	**Max Features**
29 Nuclide Masses	Reactor Type	4	56	29
	Burnup	1	77	29
	Enrichment	1	73	29
	Cooling Time	2	45	29
32 Nuclide Activities	Reactor Type	1	41	32
	Burnup	1	49	32
	Enrichment	1	67	32
	Cooling Time	7	56	32
7 or 12 Nuclide Activities	Reactor Type	1	67	7 or 12
	Burnup	1	78	7 or 12
	Enrichment	1	60	7 or 12
	Cooling Time	4	68	7 or 12
Energy Windows: Short	Reactor Type	1	62	42
	Burnup	1	62	42
	Enrichment	4	64	42
	Cooling Time	2	54	42
Energy Windows: Long	Reactor Type	4	62	151
	Burnup	1	51	151
	Enrichment	5	73	151
	Cooling Time	2	64	151
Energy Windows: Auto	Reactor Type	2	61	None or 150
	Burnup	1	52	None or 150
	Enrichment	4	67	None or 150
	Cooling Time	2	58	None or 150

Table 5.5: Optimized algorithm hyperparameters; the energy lists took all detectors into account.

and the higher value of k when the two did not match. The k and maximum depths were averaged across all six detectors for each energy window list length (short, long, and auto). There were fairly consistent results from the short and long lists, but the variable length of the auto-generated energy windows lists (in Table 5.3) gave a wider range of ideal hyperparameters.

5.4 Performance Evaluation

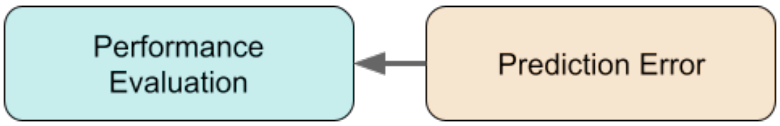


Figure 5.6: Fourth portion of the flowchart from Figure 5.1 being described in this section.

The prediction performance is measured by the balanced accuracy of the reactor type classification or the absolute and/or relative error of the regression cases (burnup, ^{235}U enrichment, cooling time), which were introduced in Section 3.2.2.1. These performance metrics for all four prediction types and all six detectors are compared across the three algorithms used: k-nearest neighbors (denoted in plots as kNN), decision trees (denoted in plots as *Dec Tree* or *DTree*), and MLL calculations. In all of the results in this section, the statistics being reported is on all 4.5×10^5 entries in the training set.

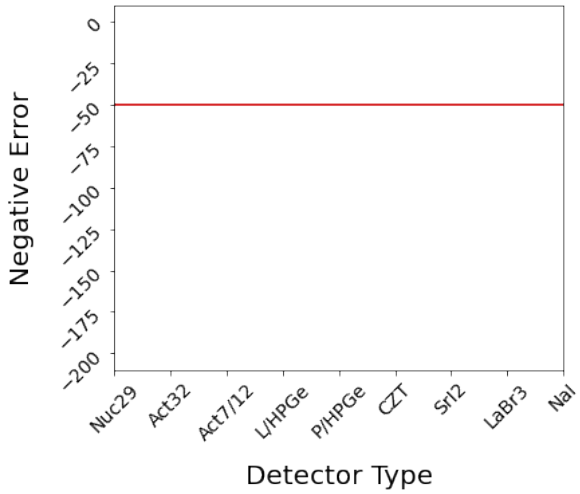


Figure 5.7: Demonstration of plot format being used to evaluate the results, shown in order to explain the axes and baseline.

In order to evaluate the prediction performance degradation with decreasing information quality via detector energy resolution, a basic plot format is presented, pictured

in Figure 5.7. There are three plots for each prediction parameter, corresponding to the three energy window lists: auto, short, and long. The vertical axis is always oriented so that higher levels indicate better performance, so is plotted as accuracy score or negative error.

The horizontal axis is oriented so that reduced information travels rightward. First, the 29 nuclide mass training set is included for comparison, followed by the 32 nuclide activity training set, which is intended to represent a full knowledge scenario of nuclide activities before gamma detection. These are both the same for all three energy window lists. The 7 & 12 nuclide activities are next, and they represent the full knowledge scenario for the post-gamma detection short and long energy window lists, respectively. So if a window is presenting the short energy window list results, it will include the data point from the 7 nuclide activity training set; the long energy window list results present the 12 nuclide activities results. The 12 nuclide activity training set is used for the auto energy window list plots, but it is not possible to know which set best represents the auto results since the peak search approach does not take physics into account and it varies across detectors. All of the nuclide mass/nuclide activity training sets are predicted with a 1% training set error applied, so that there is a non-zero but low error estimate mimicking near-perfect information. Last, the six detectors are in order of decreasing detector energy resolution. Thus, for each prediction parameter, there are three plots for each energy window list with these nine horizontal axis categories.

The last component of Figure 5.7 is the red horizontal line. It represents a minimum acceptable performance, interpreted from the results in Section 4.4. It is drawn at the level of the worst performing algorithm at 20% training set error. This is therefore somewhat arbitrary, but if the detectors cannot predict above this level, it means that a detector-based training set with only counting error cannot reach the level of performance of 20% error in a training set based on an assay of 29 nuclide masses.

5.4.1 Reactor Type Classification

To judge the degradation of algorithm predictions with decreasing information quality (detailed in Section 5.2), a plot based on Figure 5.7 is presented for each energy window list (auto, short, and long). Figure 5.8 shows the balanced accuracy of reactor type classification for the previously described horizontal axis, where a score of 1 is perfect prediction and a score of 0 is random classification. The error bars (which are not visible past the marker size on the plots) reflect a 99% confidence interval. The red baseline that indicates a minimum acceptable performance is at 0.84 balanced accuracy, which is based on the lowest performance of all three algorithms at 20% training set error for the 29 nuclide mass training set in Figure 4.7.

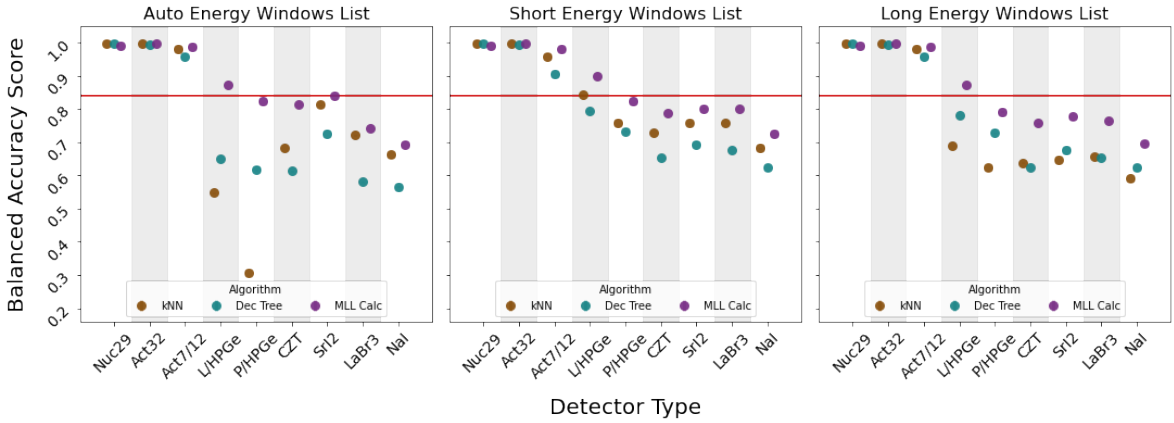


Figure 5.8: Prediction performance of reactor type as measured by balanced accuracy with respect to decreasing detector energy resolution for three types of processed gamma spectra.

The previous results in Figure 4.7 show the k-nearest neighbors line emerging with the lowest performance with decreasing information, and it is interesting that this pattern did not emerge with the detectors' performances in Figure 5.8. Instead, for the auto and short energy windows lists, the k-nearest neighbors performance is below MLL calculations and above decision trees (except in the two unique cases in the auto energy windows list). This broader trend does not hold for some specific cases, however.

For the lab-based and portable HPGes, k-nearest neighbors has the worst performance among all three algorithms for the auto and long lists, and drastically so for the two unique auto list cases of extremely poor performance. In terms of the baseline, there is one instance of k-nearest neighbors being above it: the lab-based HPGe with the short energy windows list. This is the only instance of a scikit-learn algorithm out-performing the baseline for this entire set of results.

There are no detector-based training sets at all for which decision trees performs above the baseline. For the last four detectors on the horizontal axis and all three energy windows lists, decision trees almost consistently performs the worst.

The MLL calculations are the best performing algorithm on on three plots. For the auto energy windows list, the lab-based HPGe and SrI_2 detector-based training sets perform above the baseline. For the short and long energy windows lists, only the lab-based HPGe is above the baseline.

Only the first three horizontal axis categories exceed the baseline for all three algorithms, which is expected for the various levels of full-knowledge represented. The baseline may or may not be drawn at a reasonable level with which to distinguish "acceptable" performance, but given its location, only the lab-based HPGe detector with MLL calculations performs at an acceptable level.

Next, the comparison of the gamma spectra processing will be discussed, neglecting the baseline. The short energy windows list performs the best across the three methods used to process the gamma spectra when considering all three algorithms. But for this set of results, the MLL calculations using the auto energy windows list performs the best. The auto energy windows lists even has one of the scintillator detectors (SrI_2) outperform the baseline for MLL, but the scikit-learn algorithms for the solid state detectors perform erratically. The variable behavior of the auto energy windows list would require further study, but the fact that the SrI_2 detector outperforms all of the

solid state detectors speaks to how an automatic peak search should not be discarded as a method. The auto energy windows list set of points for SrI_2 outperforms even the portable HPGe on the short energy windows list. The slightly worse outcomes for the long energy windows list are to be expected from the algorithms having to deal with a lot of non-useable features, especially for the last four detectors, since their spectra will not contain measurable peaks from many of the gamma energies in that list.

As in Section 4.4.1, the reactor type classification performance discussion would not be complete without the added detail that confusion matrices supply (see Sections 3.2.2.1 and 4.4.1.1 for an introduction to confusion matrices). First, the four "full knowledge" training sets are presented in Figure 5.9. Note that the color bar extends from -0.1 to 0.1, which is a very small range. The results in all four panels are all quite accurate, so a small color bar range is necessary to make a visual distinction of performance relative to the other full knowledge cases. The 29 nuclide mass training set (top panel) is repeated from the 1% training set error case in Figure 4.8. For the 32 nuclide activity training set (second panel), the numbers are approximately the same, but interestingly, they improve slightly for k-nearest neighbors and MLL calculations. A visible increase in misclassification is clear with the 12 and 7 nuclide activity training sets, especially for decision trees. The starkest increase in misclassification is with PWRs being classified as BWRs. This increases from about 5% to about 10% from the 12 nuclides set to the 7 nuclides set. For the 7 nuclides set decision trees classification, the misclassification of PHWRs as BWRs reaches about 5%. By contrast, the same misclassification is 3% for k-nearest neighbors and 1% for MLL calculations.

Next, Figures 5.10, 5.11, and 5.12 show the confusion matrices for all six detectors (in the same order top-to-bottom as the horizontal axis in Figure 5.8) for the auto, short, and long energy windows list training sets, respectively. For these figures, the color bar range is much wider (-0.65 to 0.65) than with the more accurate results in Figure 5.9.

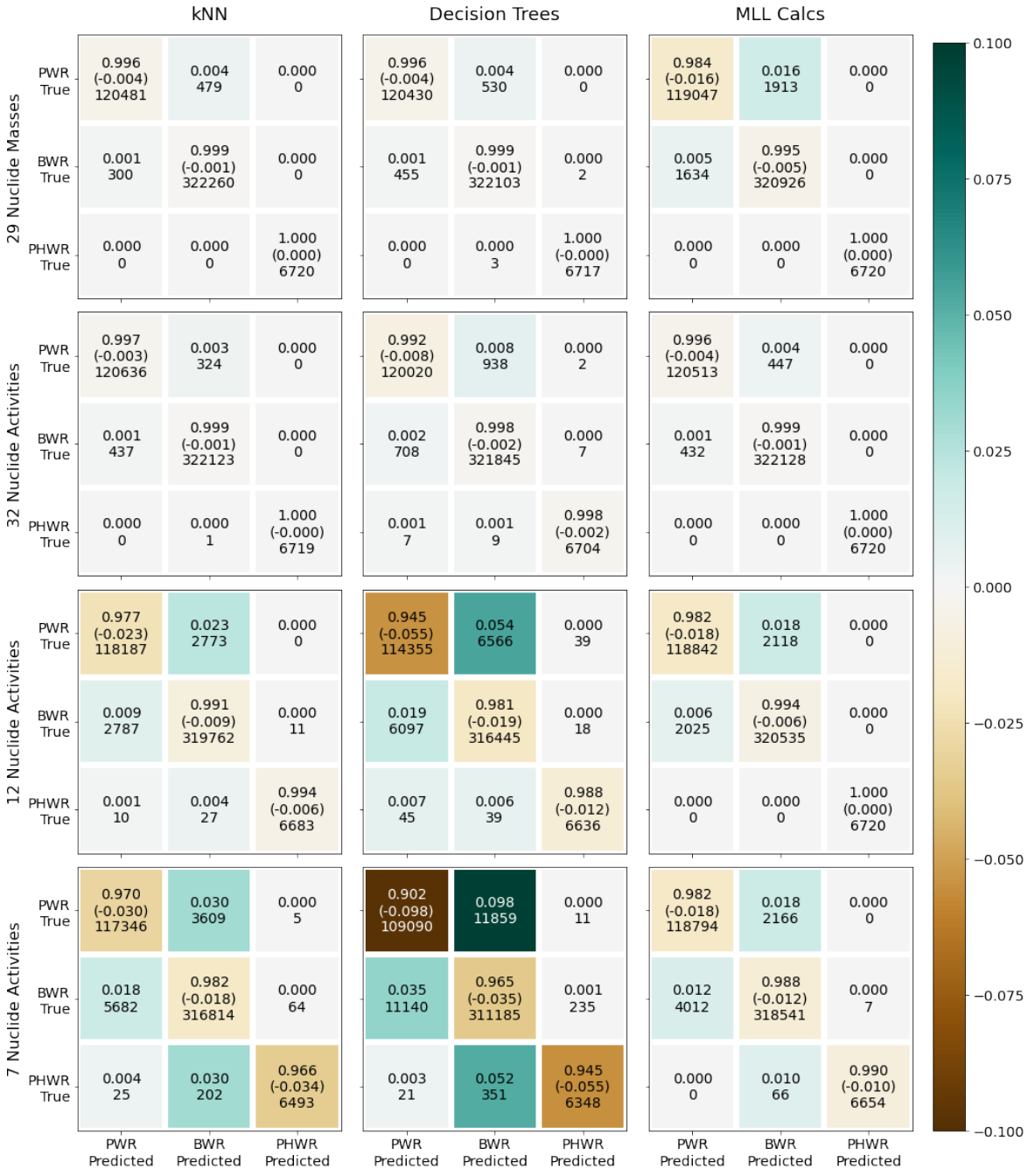


Figure 5.9: Confusion matrices of reactor type prediction for each algorithm for different training sets (all at a 1% error level): 29 nuclide masses, 32 nuclide activities, 12 nuclide activities, and 7 nuclide activities.

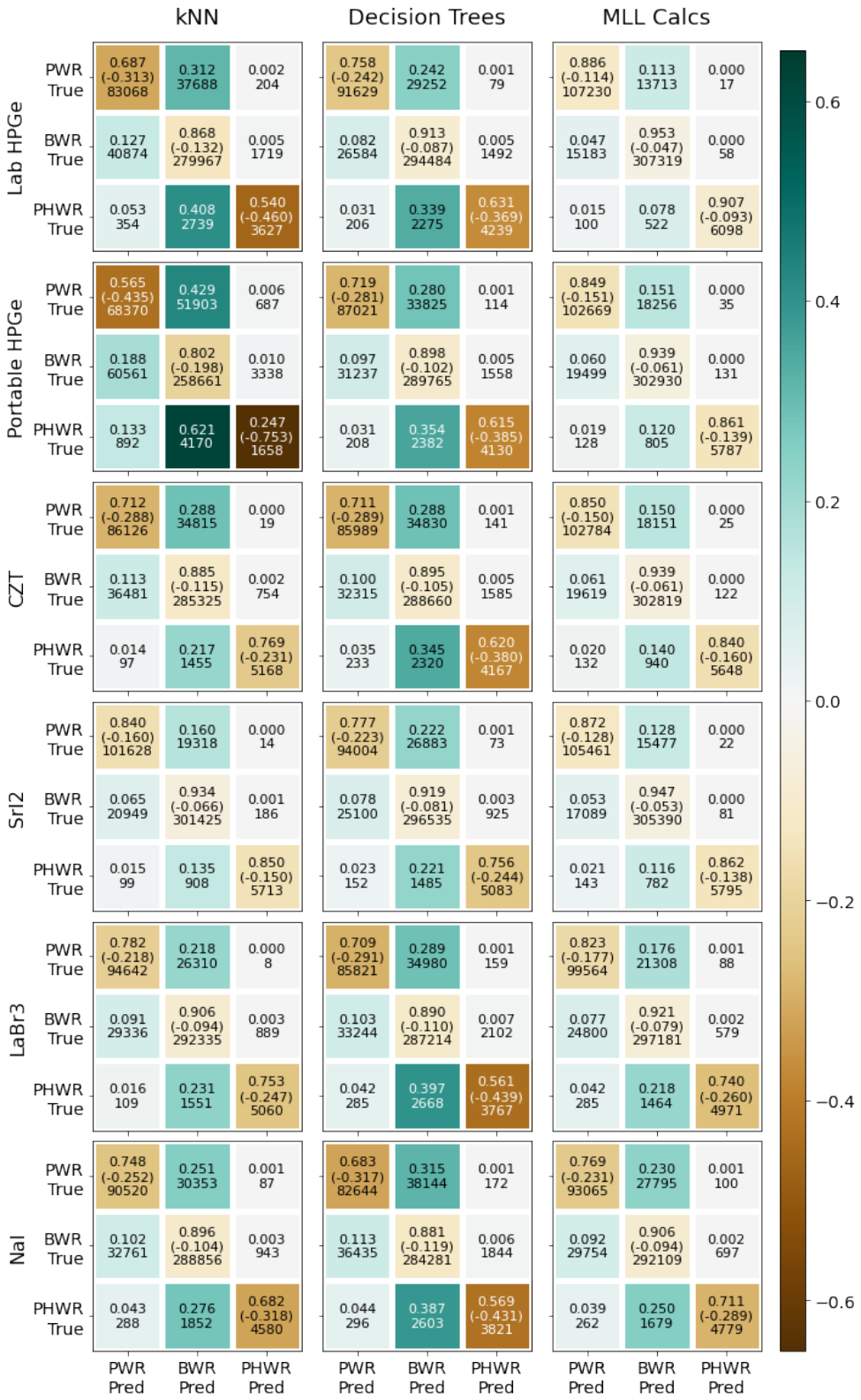


Figure 5.10: Confusion matrices for auto energy window list training sets.

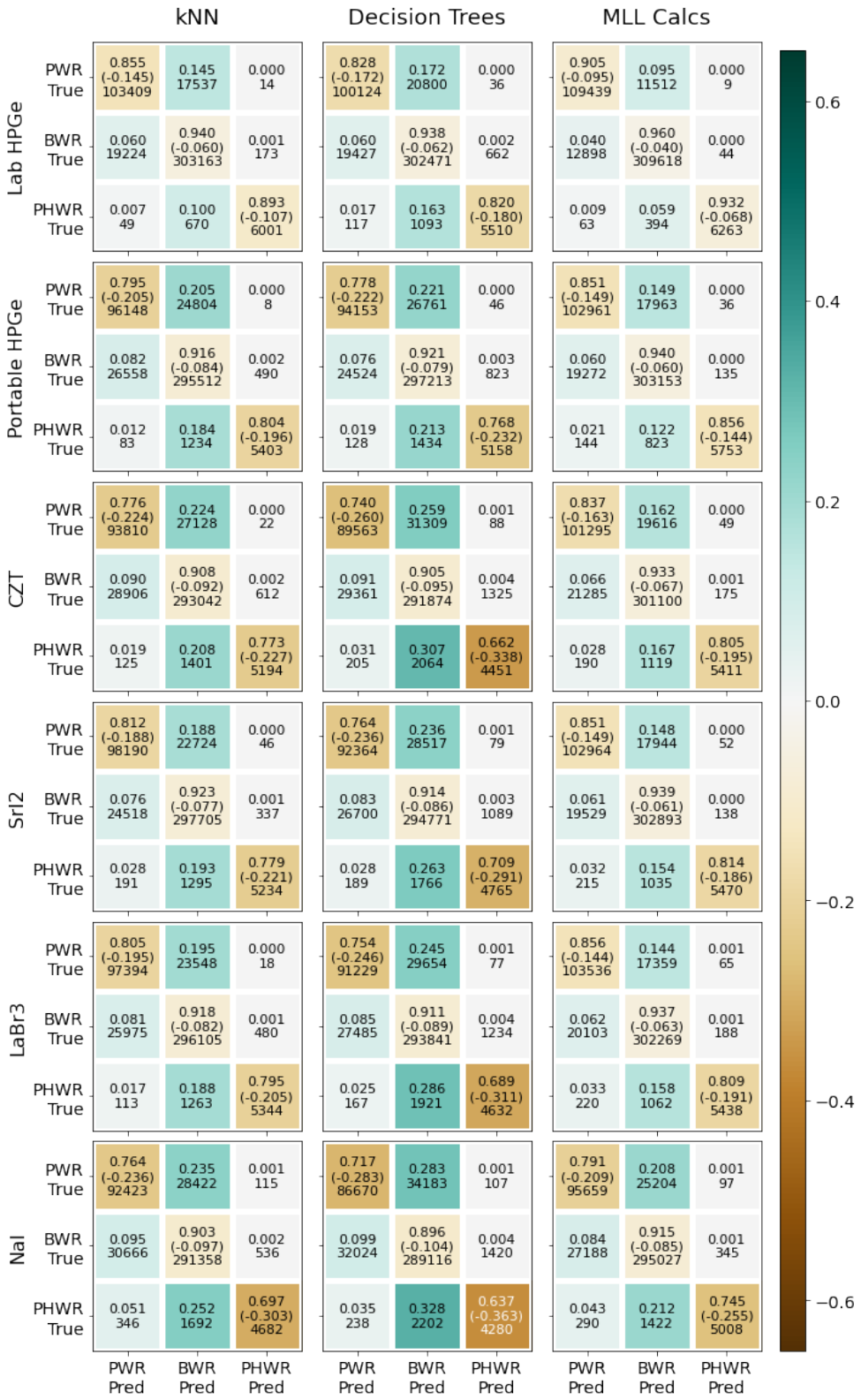


Figure 5.11: Confusion matrices for short energy window list training sets.

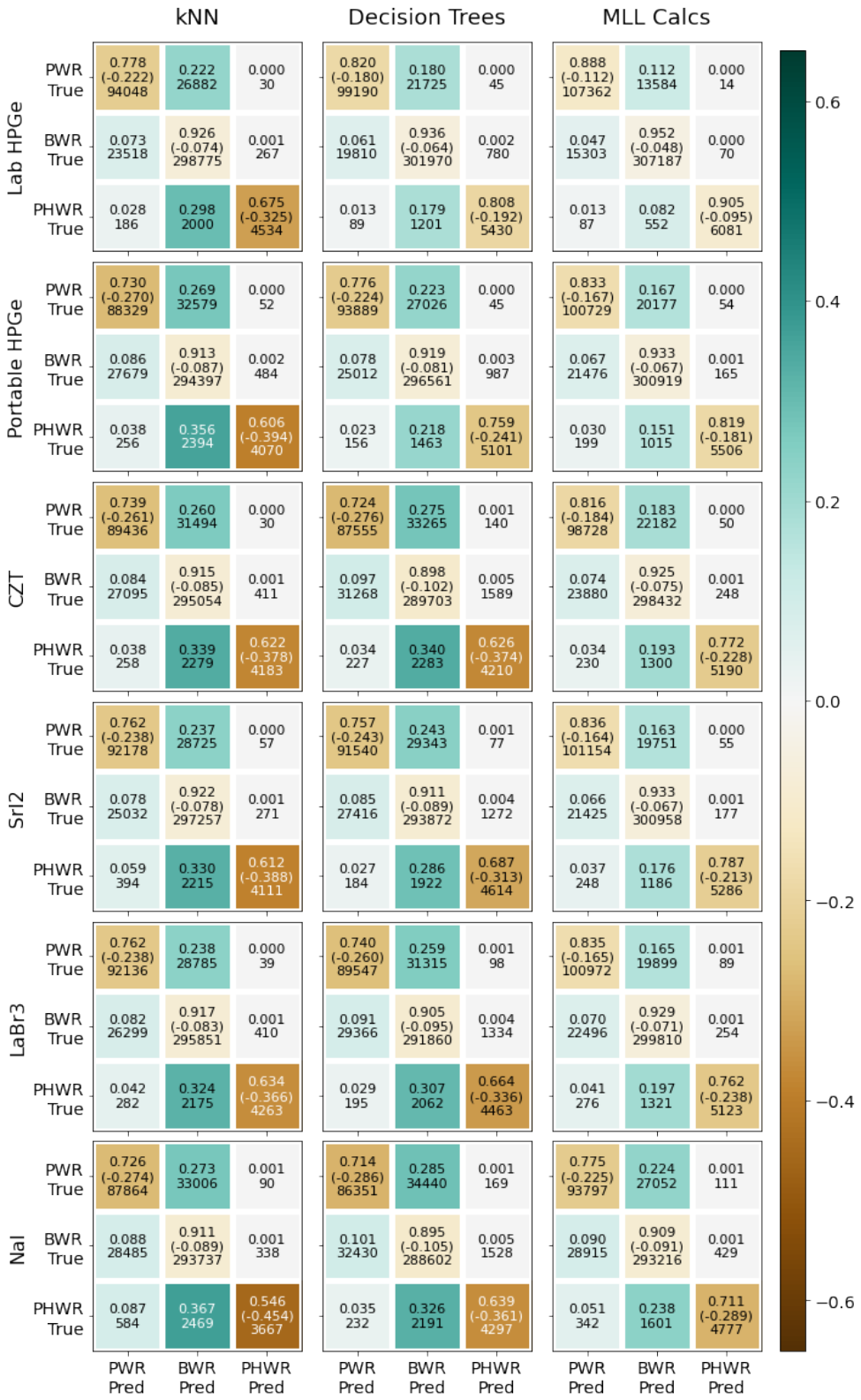


Figure 5.12: Confusion matrices for long energy window list training sets.

In Figure 5.10, the two poorly performing k-nearest neighbors HPGe cases are the most noticeable confusion matrices visually. The k-nearest neighbors performances for the two HPGes were notably poor in the plot, and the confusion matrices for these two cases explains why: not only are 40.8% and 62.1% of PHWRs being misclassified as BWRs for the lab-based and portable HPGes, respectively, 31.2% and 42.9% of the PWRs are misclassified as BWRs for the same two detectors, respectively.

By contrast, the anomalous high performance of the SrI_2 detector training set, for all three algorithms, can be seen in its panel that is paler in color than its surroundings. The confusion matrix (in the fourth panel) for those data points show no more than 16% of BWR-misclassified PWRs and PHWRs for k-nearest neighbors, no more than 23% for decision trees and no more than 13% for MLL calculations.

Aside from the two poorly performing k-nearest neighbors cases, the decision trees column has the boldest colors/poorest performance. The lowest BWR misclassification value is 22%. Additionally, it can be visually perceived at a glance that the MLL column has better performance than the other two algorithms. This is expected from the higher balanced accuracy scores in Figure 5.8. Throughout all six panels, the highest misclassification value is 25%.

Next, Figure 5.11 is on the following page, providing more detail about the reactor type classification than is seen in the middle plot in Figure 5.8. The better classification performance across all algorithms can be seen instantly, since the color bar spans the same range as with Figure 5.10. Again, the relatively poor performance of decision trees can be seen with the bolder colors compared to the two other algorithms. For decision trees, the misclassification of PHWRs as BWRs reaches 32.8%, whereas those values are 25.2% for k-nearest neighbors and 21.2$ for MLL calculations.

Last, Figure 5.12 provides more classification detail on the third plot in Figure 5.8. The MLL performance takes a similar shape at a slightly lower balanced accuracy

score compared to the short auto energy windows list, seen in Figure 5.8 and in the comparison of Figures 5.11 and 5.12. What makes the long energy windows list results unique is that k-nearest neighbors and decision trees have very close balanced accuracy scores for the bottom four detectors. By looking at their confusion matrices, one can see that the misclassification patterns are also similar. The first two detectors, both of the HPGes, have notably lower classification performance for k-nearest neighbors, however. This may or may not be related to the reason the two training sets from these detectors perform significantly worse for the auto energy windows list.

The two extreme k-nearest neighbors cases for the two HPGes is a behavior that will repeat for not-yet-discussed prediction parameters. The most likely explanation is that the automatic peak search step (as opposed to predefining the selection of peaks based on expected gamma energy photopeaks) finds peaks that are adding noise and not information to the energy windows lists. Since these algorithms all suffer from high variance, it is possible that this peak search approach is causing these two detectors (which have a high energy resolution) to succumb to the additional noise. This should be contrasted with the case of the SrI_2 detector, where it is one of the only detector-based training sets to exceed the baseline defined level.

Overall, the sets of confusion matrices presented in Figures 5.9-5.12 show various degrees of the same misclassification pattern. With less information, more PWRs and PHWRs get classified as BWRs. Since about 72% of the training set is an entry on BWR, this pattern is expected.

5.4.2 Regression Results

In this section, the results of the regression cases (burnup, enrichment, and time since irradiation) are presented. For each prediction parameter, the plot format described in Figure 5.7 is used to show the mean absolute percentage error (MAPE) for each of

the three energy window lists. However, while seeing the performance decrease with all three algorithms on the same plot is helpful for getting a bigger picture of the results, a more detailed visual is also helpful.

Introduced in Section 4.4.2.2, box plots provide increased statistical detail and a direct comparison of mean and median error for each data point, which each taken alone can paint a drastically different picture of the results. These box plots have a small variation from the original description, however. The mean absolute error (MAE) is now represented by black triangles, since many are above the 75% quartile level (i.e., outside of the box). The median absolute error (MedAE) remains represented by a white line, which may be hard to see in some plots because many median errors are near zero.

There is one set of boxes for each algorithm for a given energy window list-based set of detector training sets. Because these are box plots, they are not oriented to the "higher is better" standard of the vertical axis of the original prediction performance plots. There are the same red baselines, but they are now at a positive value since the vertical axis is no longer negative. In order to see the spread of the data and compare the mean and median errors, the outliers were suppressed from the box plots. There are many outliers in these results, and the values can be quite large. Each set of box plots is therefore repeated with a set of box plots with the outliers included.

Lastly, the full knowledge cases (the 29 nuclide masses, and the 32, 12, and 7 nuclide activities sets) are not able to be represented on the same scale as the detector training set results, so they are excluded from the box plot figures.

5.4.2.1 Burnup Regression

The baseline performance in Figure 5.13 is at -4% MAPE for burnup, chosen by the lowest performance of the three algorithms at the reference point of 20% training set error for the 29 nuclide mass training set in Figure 4.9a. In Figures 5.14 and 5.15, the red baseline is at 1000 MWd/MTU, from the reference point in Figure 4.9b. In these

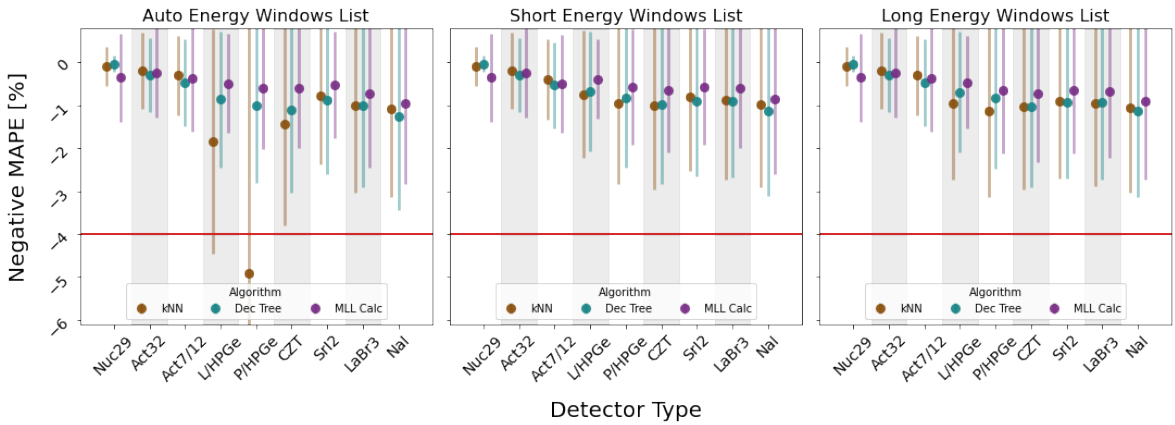


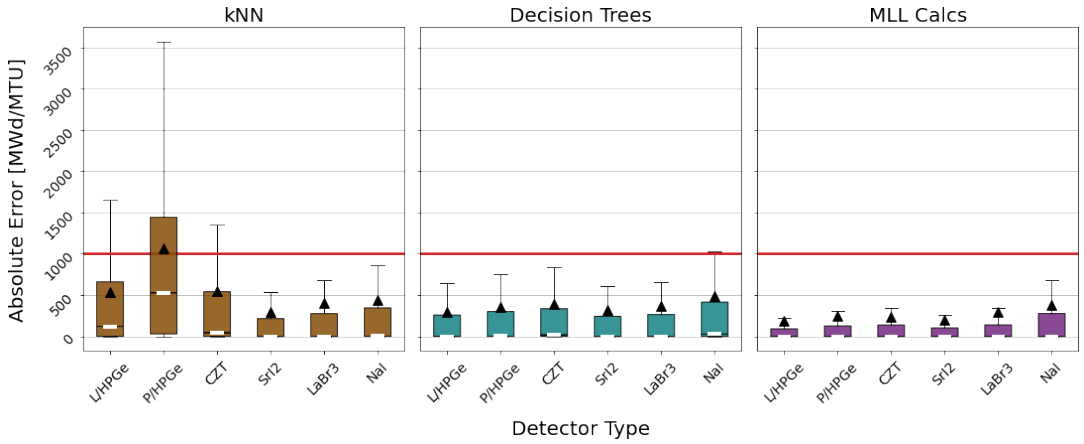
Figure 5.13: Prediction performance of burnup measured by MAPE with respect to decreasing detector energy resolution for three types of processed gamma spectra.

two sets of plots, the baseline is at a positive value because the box plots do not have a negative vertical axis.

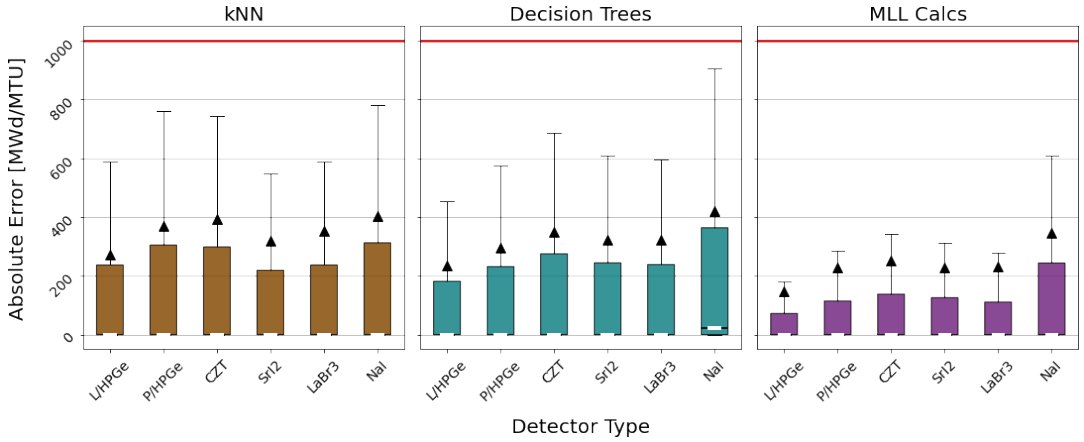
Figure 5.13 shows encouraging results about the burnup prediction for all three algorithms and all three energy windows lists used to process the gamma spectra. There is overall a gradual decrease from perfect knowledge starting at close to 0% error to the lowest energy resolution detector at about -1.5%. There are two anomalous cases, as with the reactor type results in Figure 5.8: the predictions using k-nearest neighbors for the two HPGe detectors processed with the auto energy windows list. The reason for this behavior is discussed at the end of Section 5.4.1.

Next, in Figures 5.14 and 5.15 the vertical axis orientation flips to positive absolute error, so that higher values are worse and the goal is to have the mean and median errors be below the red line. Most of the burnup errors in Figures 5.14a, 5.14b, and 5.14c are below the red line, except for the MAE from the portable HPGe. This set of errors from the portable HPGe also encompass a range than reaches up to $> 3500\ MWd/MTU$, which is about 4× higher than the other box plots in all three subfigures.

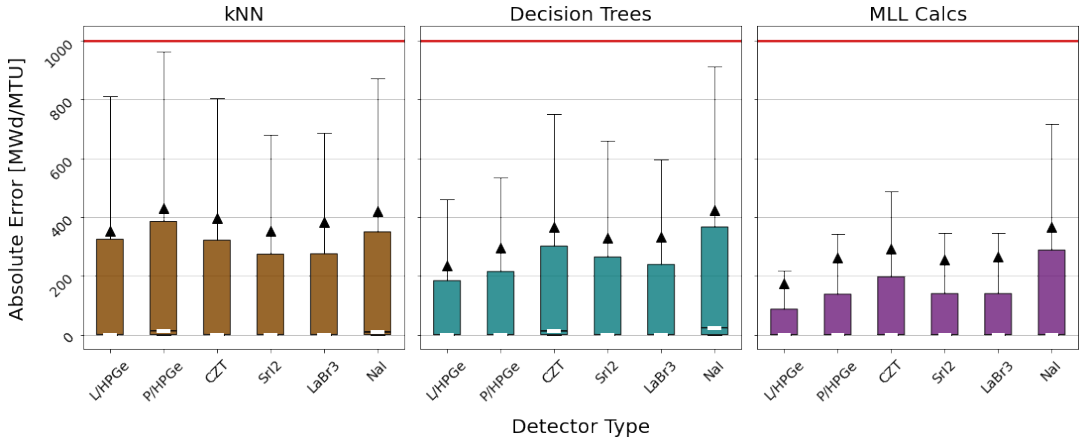
The burnup predictions for the most part perform better than the red line, for all algorithms and gamma spectra-processing methods. Of course, the MLL calculations



(a) Burnup prediction error box plots for auto energy windows list.

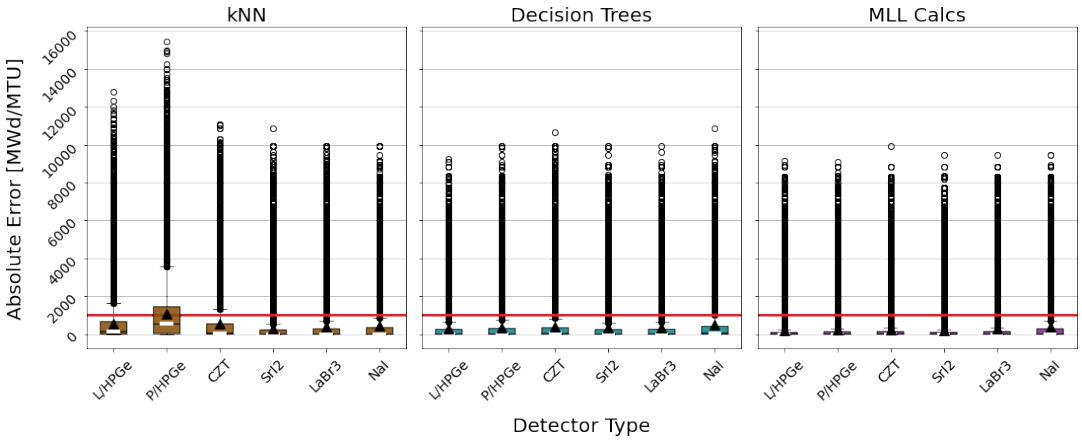


(b) Burnup prediction error box plots for short energy windows list.

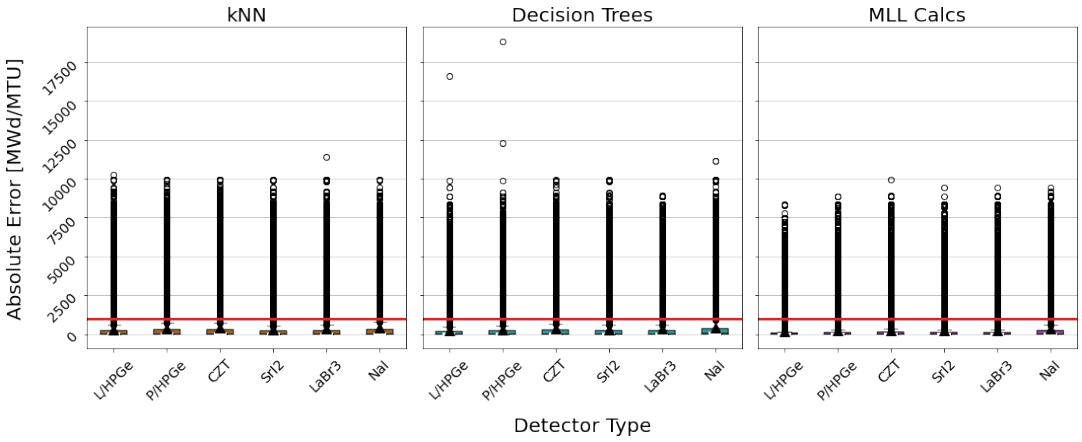


(c) Burnup prediction error box plots for long energy windows list.

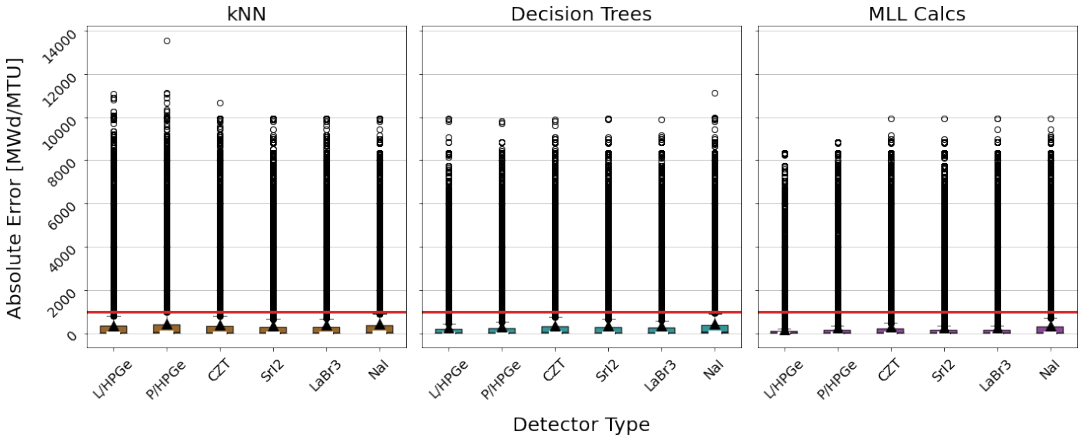
Figure 5.14: Prediction performance of burnup for six detectors as shown by box plots *without* outliers shown.



(a) Burnup prediction error box plots for auto energy windows list.



(b) Burnup prediction error box plots for short energy windows list.



(c) Burnup prediction error box plots for long energy windows list.

Figure 5.15: Prediction performance of burnup for six detectors as shown by box plots *with* outliers shown.

consistently have the lowest errors. In all but the two HPGe cases with the auto energy windows list, the MedAEs are close to zero. This indicates that there are very good predictions for at least half of the training set. However, the MAEs are all above the 75% quartile, which suggests the errors that do exist are quite large relative to the interquartile range.

Furthermore, looking at the sets of box plots with outliers present in Figure 5.15 shows that there are many errors far outside of the full range of burnup errors that reach up to almost 20000 MWd/MTU (for decision trees with the short energy windows list in Figure 5.15b), an error that has a magnitude of about 30% of the largest burnup value in the training set. Figure 5.13 shows that these large MAEs are actually are of low relative error with the data points all being above -1.5%. The number of outliers ranges from an average between $53k$ and $65k$ per box for the two scikit-learn algorithms, and about $76k$ per box for the MLL calculations. This means that $12 - 17\%$ of the training set samples are outliers. The high magnitude errors along with the large number of outliers contributes to the standard deviations in Figure 5.13, which are so large that it is difficult to conclusively define a trend.

The overall flatness of the boxplots and better-than-baseline performance indicates that the features needed to determine burnup are easily measured by all detector configurations.

5.4.2.2 ^{235}U Enrichment Regression

The baseline performance for the enrichment predictions in Figure 5.16 is at -6% MAPE, chosen by the lowest performance of the three algorithms at the reference point of 20% training set error for the 29 nuclide mass training set in in Figure 4.10a. In Figures 5.17 and 5.18, the red baseline is at 0.17 % ^{235}U, from the reference point in Figure 4.10b. In these two sets of plots, the baseline is at a positive value because the box plots do not have a negative vertical axis.

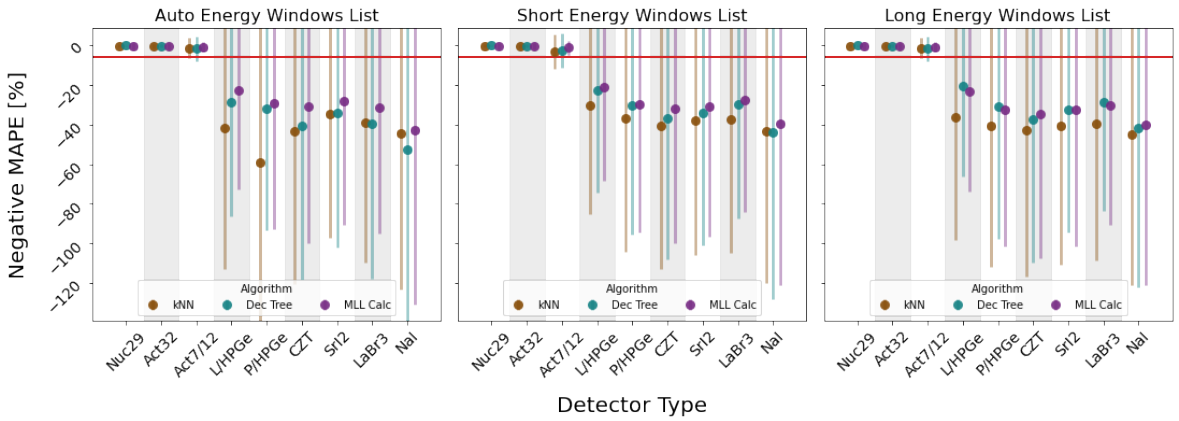
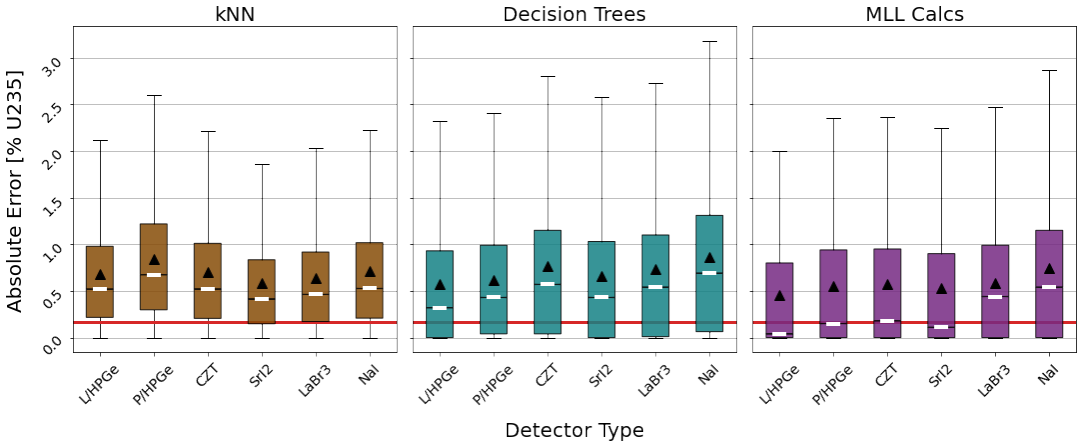


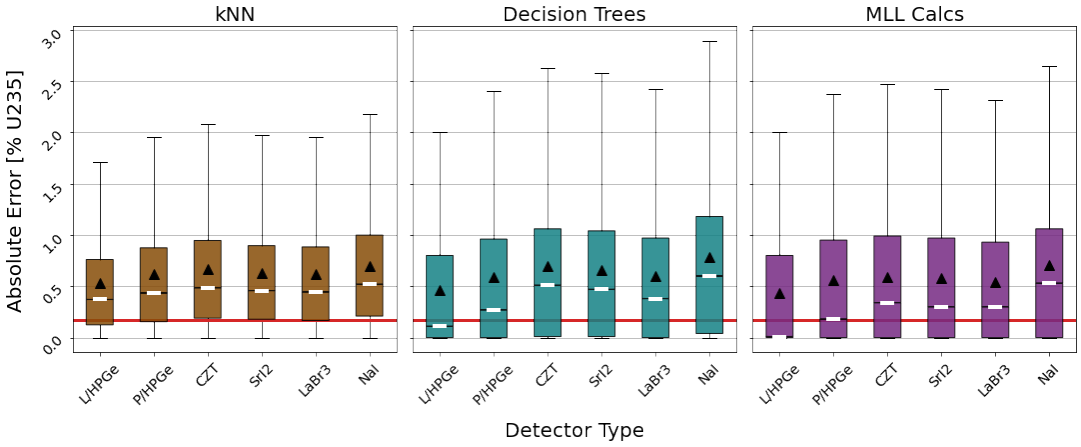
Figure 5.16: Prediction performance of ^{235}U enrichment measured by MAPE with respect to decreasing detector energy resolution for three types of processed gamma spectra.

Figure 5.16 shows discouraging results about the ^{235}U enrichment prediction for all three algorithms and all three energy windows lists used to process the gamma spectra. Taken as a whole, the data points in Figure 5.16 have a distinct shape, similar to the behavior in Figure 5.8. The three full knowledge cases have near-perfect enrichment prediction and the six detectors for all three energy windows lists are nearly flat, where most data points are in the -50% to -30% range. This is a large range but also a high relative error. As with the other previously discussed prediction types, there are two anomalous cases: the prediction using k-nearest neighbors for the two HPGe detectors processed with the auto energy windows list. The reason for this behavior is discussed at the end of Section 5.4.1. In this set of results, k-nearest neighbors usually performs the worst or, when it does not, about equal to decision trees. The MLL calculations are the best performing for the auto energy windows list, but the results in the short and long energy windows lists shows decision trees performing close to and sometimes outperforming MLL.

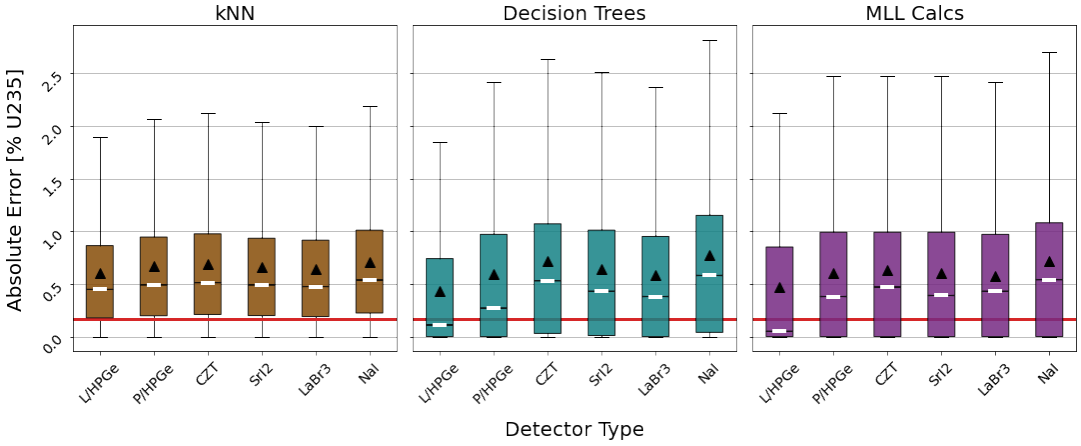
Next, in Figures 5.17 and 5.18 the vertical axis orientation flips to positive absolute error, so that higher values are worse and the goal is to have the mean and median errors be below the red line. Unlike with the burnup errors being mostly below the red



(a) ^{235}U enrichment prediction error box plots for auto energy windows list.

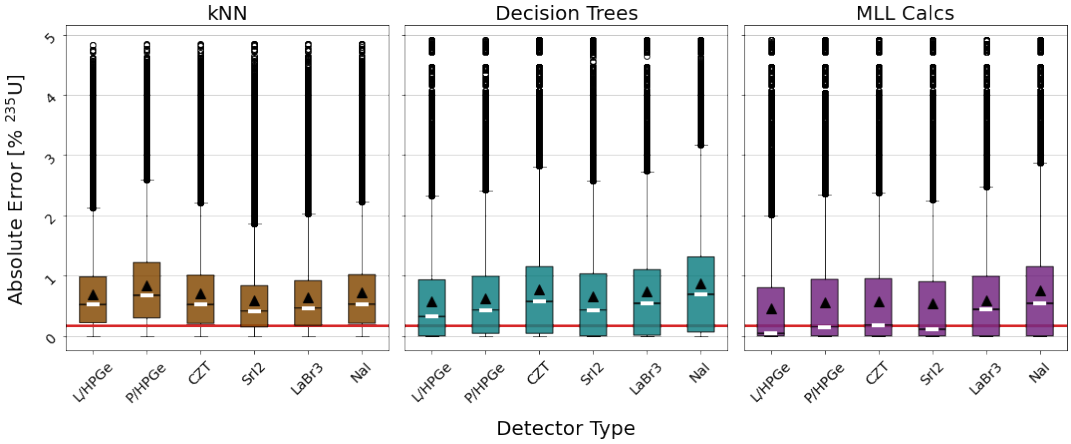


(b) ^{235}U enrichment prediction error box plots for short energy windows list.

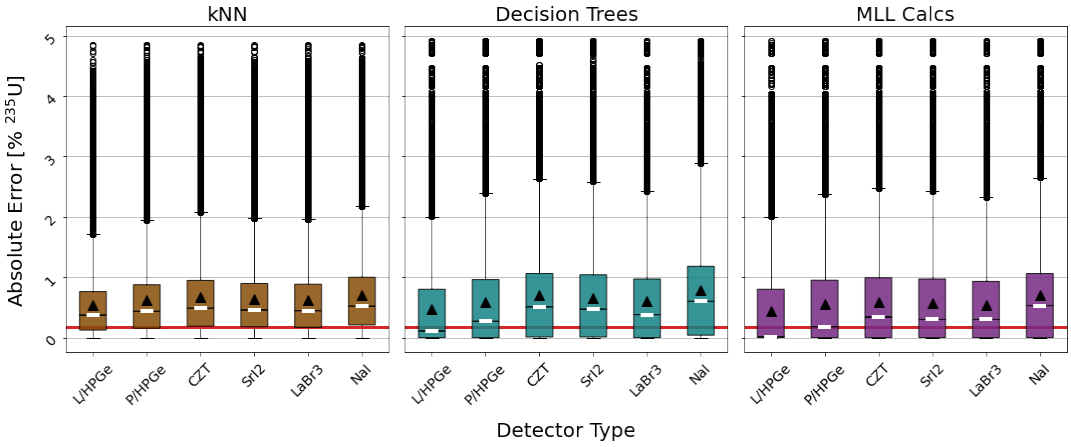


(c) ^{235}U enrichment prediction error box plots for long energy windows list.

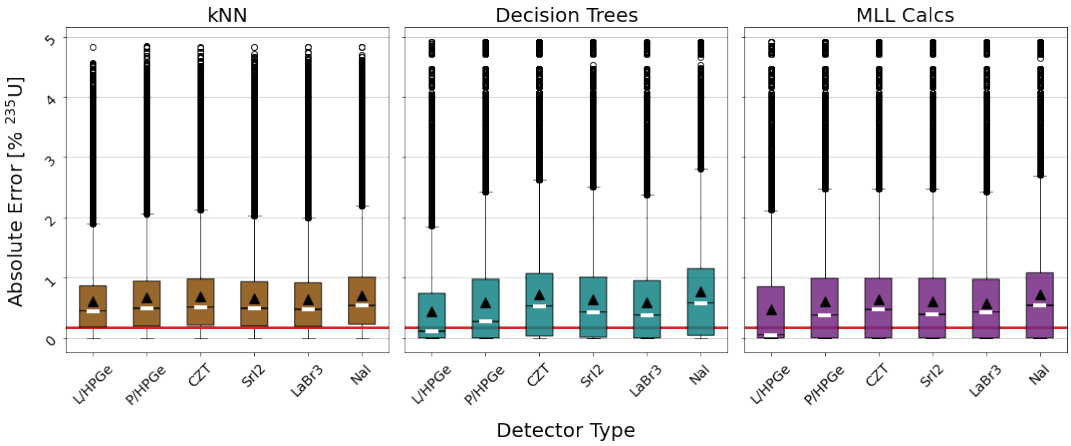
Figure 5.17: Prediction performance of ^{235}U enrichment for six detectors as shown by box plots *without* outliers shown.



(a) ^{235}U enrichment prediction error box plots for auto energy windows list.



(b) ^{235}U enrichment prediction error box plots for short energy windows list.



(c) ^{235}U enrichment prediction error box plots for long energy windows list.

Figure 5.18: Prediction performance of ^{235}U enrichment for six detectors as shown by box plots *with* outliers shown.

line in Figure 5.14, this is not the case for enrichment. None of the MAEs are below the red line, and the majority of the MedAEs are also above it. There are a few box plots where the MedAE is below the red line: the first four detectors processed with the auto energy windows list as predicted using MLL, and the lab-based HPGe processed with both the short and long lists for decision trees and MLL calculations. Interestingly, the red line appears to run across the bottom quartile for the k-nearest neighbor predictions for all three energy windows lists, which indicates about 25% of those predictions are below the standard defined using mass-based information.

The range of errors reaches to about 3 % ^{235}U without outliers, which is a large magnitude since the maximum enrichment is about 5 % ^{235}U. All nine combinations of algorithm and energy windows list in Figure 5.18 have outliers that reach the full 5 % ^{235}U, indicating that some of the 0.5% enriched fuels are being mispredicted at a 5% level. Furthermore, an error that large is of the magnitude of the highest enrichment level in the training set. The number of outliers ranges from an average of about $18k$ for the k-nearest neighbors boxes, and about $14k$ per box for the decision trees and MLL calculations, meaning about 3-4% of the training set samples are outliers. Although there are much fewer outliers than with the burnup predictions, the wide range of MAE magnitudes and the large MAPEs contribute to the large standard deviations in Figure 5.16. Again, the standard deviations are so large that it is difficult to conclusively define a trend.

The overall flatness of the boxplots and far-below-baseline performance indicates that the features needed to determine enrichment are not well-measured by any of the detector configurations.

5.4.2.3 Time Since Irradiation Regression

The original baseline performance for the time since irradiation predictions in Figure 5.19 were at -30% MAPE, chosen by the lowest performance of the three algorithms at

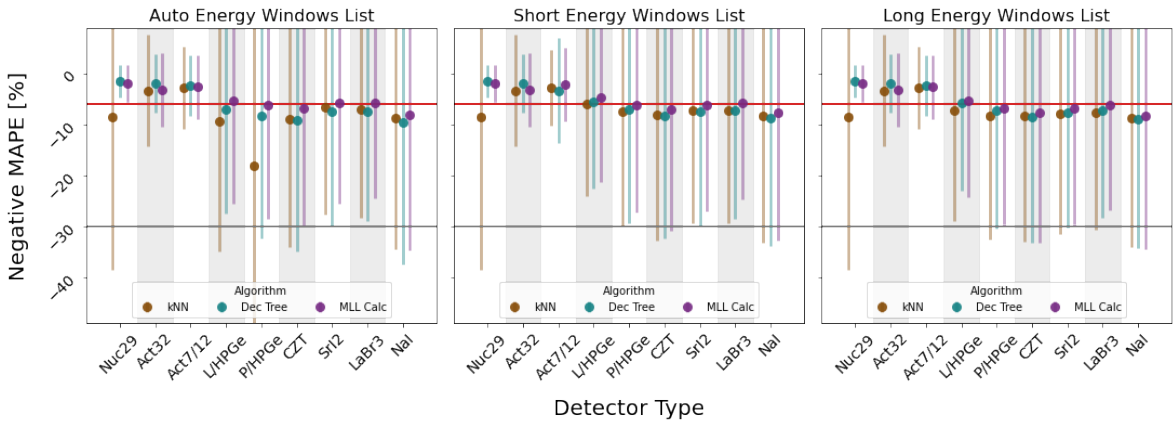
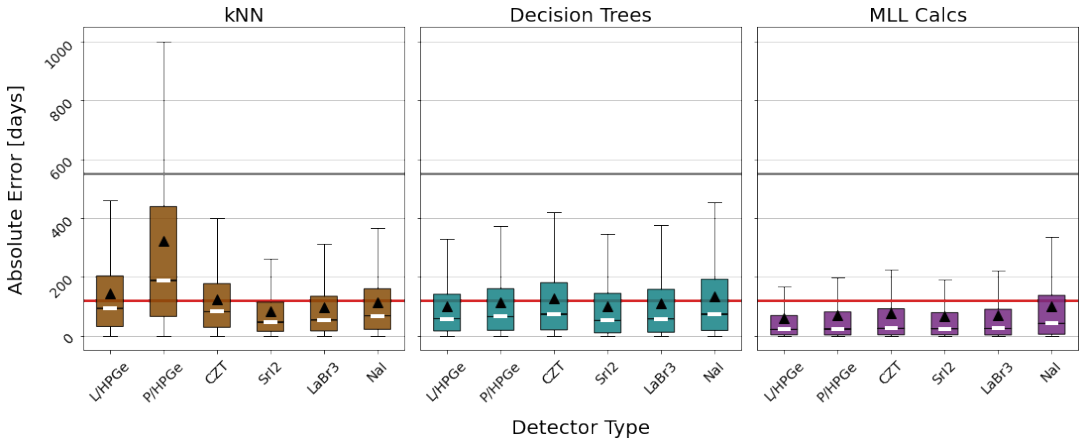


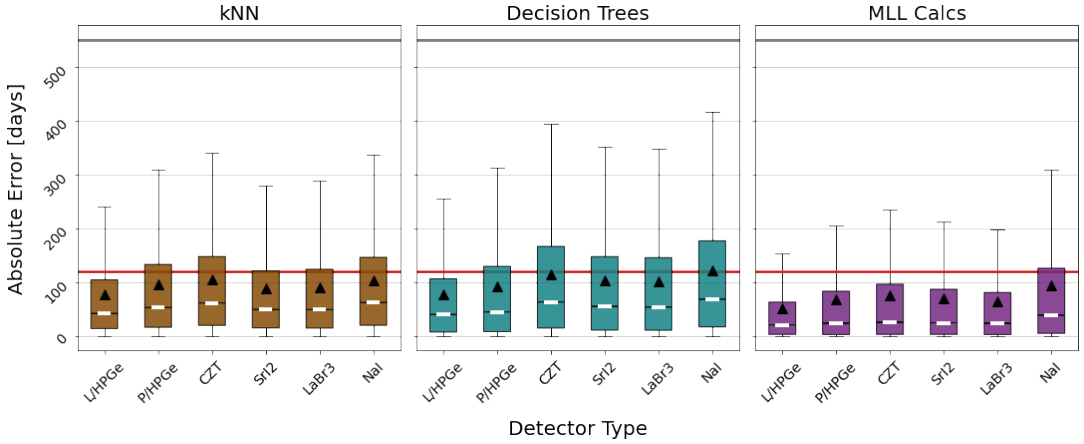
Figure 5.19: Prediction performance of time since irradiation measured by MAPE with respect to decreasing detector energy resolution for three types of processed gamma spectra.

the reference point of 20% training set error for the 29 nuclide mass training set in in Figure 4.11a. However, the k-nearest neighbors algorithm in that plot had exceptionally poor performance compared to the other two algorithms, so all of the results from the detector-based training sets (including those predicted using k-nearest neighbors, oddly enough) far outperform this baseline. Therefore, a new one was chosen based on the decision trees performance at the reference point in Figure 4.11a, at a MAPE of -6%. Similarly, in Figures 5.20 and 5.21, the original baseline is at 550 *days* and the updated red baseline is at 120 *days*, from the k-nearest neighbors and decision trees performances, respectively, at the reference point in Figure 4.11b. In these two sets of plots, the baseline is at a positive value because the box plots do not have a negative vertical axis.

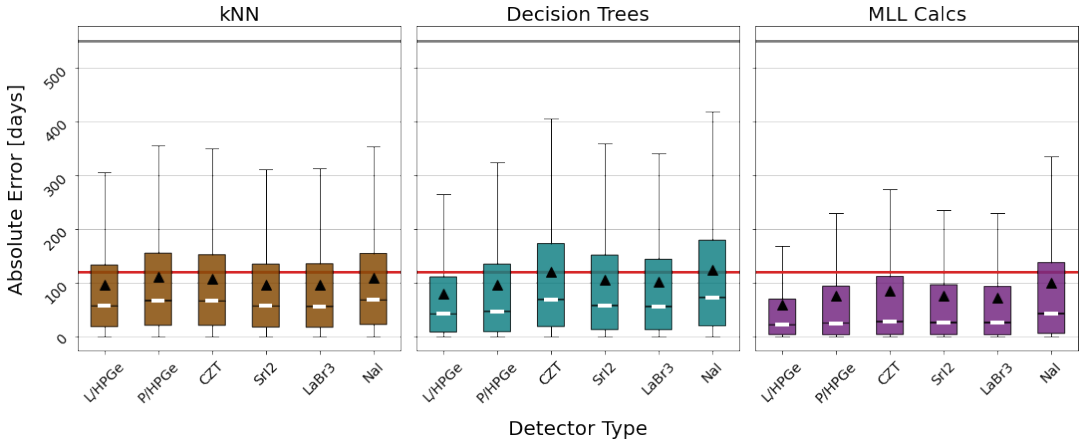
Figure 5.19 shows the relative error results from the time since irradiation prediction for all three algorithms and all three energy windows lists used to process the gamma spectra. There is a gradual decrease from perfect knowledge, in contrast to enrichment but similar to burnup, starting at close to -2% error to the lowest energy resolution detector at about -10%. The red line in these plots makes for an interesting delineation of the results, since there are 2 or 3 MLL data points in each plot that are on or exceed



(a) Time since irradiation prediction performance box plots for auto energy windows list.

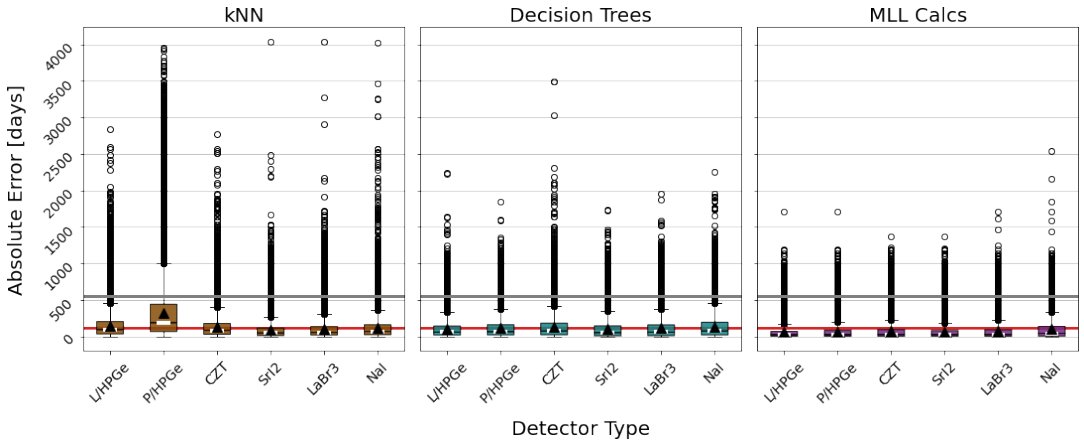


(b) Time since irradiation prediction performance box plots for short energy windows list.

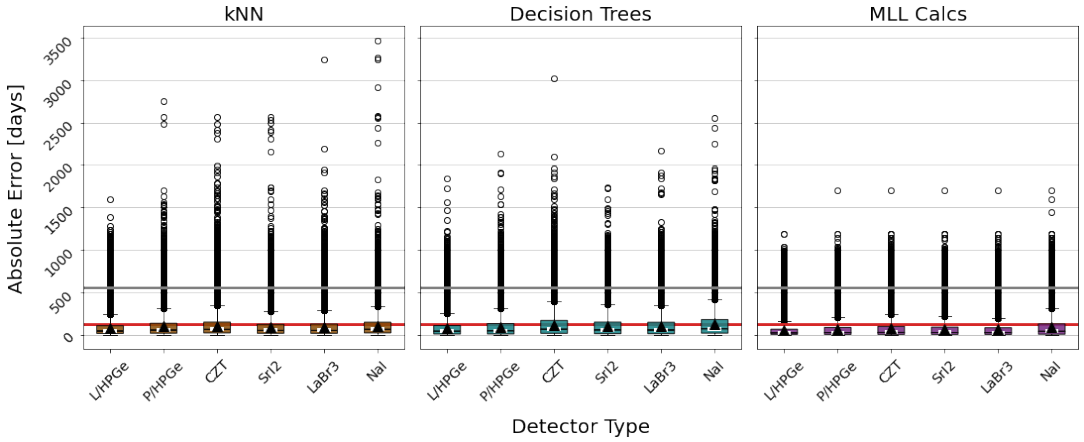


(c) Time since irradiation prediction performance box plots for long energy windows list.

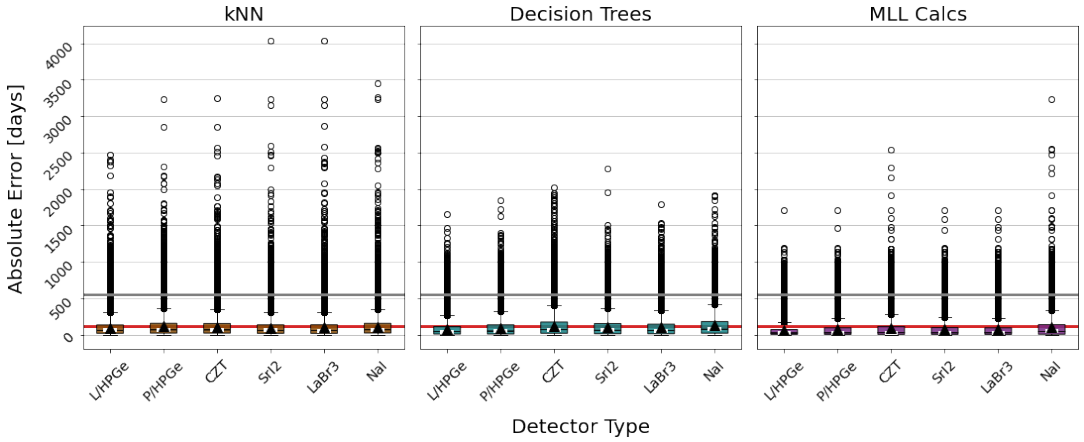
Figure 5.20: Prediction performance of time since irradiation for six detectors as shown by box plots *without* outliers shown.



(a) Time since irradiation prediction performance box plots for auto energy windows list.



(b) Time since irradiation prediction performance box plots for short energy windows list.



(c) Time since irradiation prediction performance box plots for long energy windows list.

Figure 5.21: Prediction performance of time since irradiation for six detectors as shown by box plots *with* outliers shown.

the baseline. Additionally, the decision trees data points for the lab-based HPGe for the short and long energy windows lists also exceed the baseline. The remainder of the detector-based training sets are below the baseline. Notably, the k-nearest neighbors case with the 29 nuclide masses training set is below the line at about -10%. This is from the previously mentioned issues with this algorithm's performance with this particular prediction parameter. For reactor type, burnup, and enrichment, the predictions using k-nearest neighbors for the two HPGe detectors processed with the auto energy windows list performed drastically worse than the rest of the scenarios. For time since irradiation, however, this is only true for the portable HPGe. The reason for this behavior is discussed at the end of Section 5.4.1.

Next, in Figures 5.20 and 5.21 the vertical axis orientation flips to positive absolute error, so that higher values are worse and the goal is to have the mean and median errors be below the red line. The auto energy windows list results in Figure 5.20a have many of the MAEs and all of the MedAEs below the red line. The MAEs of the two HPGes for k-nearest neighbors are above it, and the CZT and NaI detectors for decision trees are above it as well. Nearly all of the cooling time errors in Figures 5.20b, and 5.20c are on or below the red line, except for the MAE of the decision trees results for the NaI detector. Other than the portable HPGe box which encompasses a range reaching 1000 $days$, the range of absolute errors rarely exceeds 400 $days$, or a little over a year. Taking an aggregate view of the three plots, many of the MAEs are around 100 $days$.

Although the range of absolute errors typically reaches to around 400 $days$, the outliers in Figure 5.21 reach to about $10x$ that level to approximately 4000 $days$. This error magnitude is about 70% of the largest time since irradiation level in the training set. The number of outliers have an average of about $26k$ per box for the two scikit-learn algorithms, and about $41k$ per box for the MLL calculations. This means that $6 - 9\%$

of the training set samples are outliers. The high magnitude errors along with the large number of outliers contributes to the standard deviations in Figure 5.19, which are so large that it is difficult to conclusively define a trend.

The overall flatness of the boxplots and close-to-baseline performance indicates that the features needed to determine time since irradiation are close to adequately measured by most of the detector configurations.

5.5 Summary

This chapter covers the details of the methodology in four main sections: training set simulations, information reduction, statistical methods implementation, and performance evaluation. This approach focuses on the situation where there are only nuclides present in the training sets that are measured in a non-destructive manner, i.e., via gamma detectors.

First, in Section 5.1, the training data is simulated using the same inputs as in Section 4.1. For this training set, however, the features tracked are 32 radionuclides.

Second, in Section 5.2, information reduction on the training set is carried out using computationally generated gamma spectra. The training set is initially comprised of nuclide activities, and GADRAS computes the gamma spectra via DRFs for six detectors [49] using the inputs of the nuclide activity measurements for each SNF entry. Each detector-based training set undergoes processing by summing the counts of the bins of each energy window from three different lists: the auto, short, and long energy windows lists. These training sets are intended to answer the question of whether field-deployable detectors can give enough information about radionuclides to successfully attribute SNF.

Third, in Section 5.3, the updates to the algorithm implementation from Section 4.3 are covered. The three algorithms, k-nearest neighbors, decision trees, and MLL

calculations, are used to train models to predict the four reactor parameters of interest. The new training sets also first underwent hyperparameter optimization.

Fourth, in Section 5.4, the prediction errors are presented to evaluate the ability of this methodology to attribute SNF with non-destructive measurements of radionuclides. The reactor type classification is discussed in Section 5.4.1 and the regression cases are discussed in Section 5.4.2.3. The performance of the prediction of reactor parameters is measured by using test cases drawn from the training set.

The reactor type classification results are presented in Section 5.4.1. The MLL calculations consistently perform the best across the algorithms, and the short energy windows list has the best performance across the energy windows lists. Most of the misclassifications are PWR and PHWR being labeled as BWR. The automatic peak searching results in erratic behavior for the scikit-learn algorithms because the high energy resolution detectors perform poorly but one of the lower energy resolution detectors performed very well. Most of the algorithm-detector combinations do not exceed the baseline, and the only cases that do are the MLL calculations for the lab-based HPGe with all three energy windows lists and the MLL calculations for the SrI_2 detector with the auto energy windows list. *Since most of the detector-based data points do not exceed the baseline for balanced accuracy, reactor type classification only meets the standard defined using mass-based information for the four aforementioned cases and the three full-knowledge training sets.*

Next, the regression cases are shown in Section 5.4.2.3. The burnup predictions for all the algorithms and energy windows lists outperform the baseline, except in one case of k-nearest neighbors being used with the auto energy windows list. The enrichment predictions for the set of six detectors all falls below the baseline, however, and the time since irradiation predictions all are very close to the baseline, with some points right above it and some right below it. The spread of outliers encompasses $3 - 17\%$ of the

training set depending on the case, and in most cases the magnitude of the outlier errors is of significant concern, although most of the median errors paint a better picture of the performance. *Based on the performances relative to the baseline for each regression case, burnup performs above the standard, enrichment performs far under the standard, and time since irradiation has several cases that perform on or near the baseline and thus are very close to the minimum standard.*

For all three regression cases, there is little contrast among the three energy windows lists. The auto list has more erratic behavior, and the short list has a slight but not significant average performance over the long list. *This indicates that the performance of the methodology in this work is largely independent of the gamma spectra processing approaches.*

6 CONCLUSIONS AND FUTURE WORK

This chapter focuses on concluding remarks and future work. First, a summary is outlined in Section 6.1, Next, in Section 6.2, the main experimental results from each section are brought together to discuss the bigger picture of this work. Additionally, there are areas of the methodology that are simplified and several avenues this work could pursue, so this is discussed in Section 6.3.

6.1 Summary

The main research question that this work addresses is as follows: *How does the ability to determine forensic-relevant spent nuclear fuel attributes using machine-learning techniques degrade as less information is available?*. The workflow to examine this question takes place in four steps.

First, the training data is simulated, which provides an array of nuclide measurements as the features. The prediction parameters are the simulation inputs: reactor type, burnup, ^{235}U enrichment, and time since irradiation. The reactor types are one of three common commercial reactors: PWR, BWR, or PHWR. The burnup is measured in MWd/MTU , the enrichment is measured in $\%^{235}$U, and the time since irradiation is measured in *days*. The size of the training set is 4.5×10^5 SNF entries, which were simulated using ORIGEN [26, 27, 28]. These steps are covered in Sections 4.1 and 5.1.

Second, information reduction on the training set is carried out using randomly injected uniform error or computationally generated gamma spectra. For the former, the training set is comprised of nuclide masses and the random error is used to study the robustness of the methodology to artificial noise in the feature set. For the latter, the training set becomes lists of summed energy windows of the gamma spectra, which were computed from nuclide activity measurements using GADRAS DRFs for six detectors

[49]. Each detector-based training set undergoes three different methods of processing the gamma spectra, denoted as auto, short, and long energy windows lists. These steps are covered in Sections 4.2 and 5.2.

Third, three algorithms are used to predict the four reactor operation parameters in this work: k-nearest neighbors, decision trees, from the scikit-learn python ML toolkit, and MLL calculations, implemented in python using SciPy and NumPy [29, 43, 44]. For the scikit algorithms, the hyperparameters governing model complexity were optimized to minimize the prediction errors. All of the code was run using University of Wisconsin (UW)–Madison's Center for High Throughput Computing (CHTC) resources, the UW campus grid, and the Open Science Grid (OSG) [45, 46]. These steps are covered in Sections 4.3 and 5.3.

Fourth, the prediction errors are evaluated to understand the increase in error as the information supplied is less precise. This impacts the conclusions that can be formed about the capability of simple statistical methods to inform nuclear material attribution. The testing samples come directly from the training set, and the prediction errors from these testing sample are what is studied for the performance. When a test sample is used, it is first removed from the training set so that there is no exact replica of it in the training stage. This testing scheme allows the entire training set to be tested eventually. It is accomplished via 5-fold cross-validation (CV) for the scikit-learn algorithms and by testing one training set entry at a time for the MLL calculations. These results are presented in Sections 4.4 and 5.4, and are summarized as follows.

6.2 Conclusions

Reactor Parameter Prediction Using Nuclide Masses

Sections 4.4.1.1 and 4.4.1.2 demonstrate the impacts of injected training set noise on the prediction parameters. Aside from the k-nearest neighbors prediction of time since

irradiation, all of the algorithms follow a similar pattern for each prediction scenario. The MLL calculations are the most resilient to introduced error, followed by decision trees then k-nearest neighbors. *It is difficult to state whether or not these results are "good enough" on their own, but they do serve as a useful baseline for the results in Chapter 5.*

Next, the model generalization is studied in Section 4.4.1.3 by creating learning curves using $20 - 100\%$ of the training set. It shows that the training set is large enough for the MLL calculations to remain the best performing method until the 20% training set size, when those data points dip below one or both of the scikit-learn algorithms depending on the prediction parameter. *This performance is highly dependent on the training set design because these are qualitatively high variance methods (compared to common algorithms not shown in this work) that do not generalize well.*

Section 4.4.1.4 investigates the effect of having reactor type prior knowledge on burnup, enrichment, and time since irradiation prediction. There is modest and significant improvement for PWR and PHWR cases, respectively, since they are both the minority classes in the training set. These improvements only exist for the scikit-learn algorithms, since the MLL calculations are not affected by reactor type knowledge. *The key takeaway is that regression cases linked to PHWRs are not expected to perform well.*

The last study in this chapter is in Section 4.4.2, where an external test set is used that is comprised of measurements from real commercial power SNF from around the world: the Spent Fuel isotopic COMPOsition (SFCOMPO) database. The main challenge with this database are that many entries only contain measurements for uranium and plutonium isotopes, and the missing measurements are inconsistent. Two methods were implemented: imputing the null values with the mean of a given feature, and imputing the null values with zero. The reactor type classification results are very poor except for the MLL predictions using zero-valued nulls. This combination also best

predicts the burnup, but oddly, the decision trees for both methods of null handling best predicts the enrichment. *Overall, the errors are still quite large, so the SFCOMPO database presents challenges to being used with this methodology. The techniques for improving the performance are discussed next in the future work suggestions, Section 6.3.*

Reactor Parameter Prediction Using Processed Gamma Spectra

The reactor type classification results are first covered in Section 5.4.1. Unsurprisingly, the MLL calculations consistently perform the best across the algorithms. There is a slightly better overall performance by the short energy windows list as compared to the others, although the large variation by the auto energy windows lists has some data points performing better. The confusion matrices show the details of every data point, but most of the misclassifications are PWR and PHWR being labeled as BWR. The only algorithm-detector combinations that exceed the baseline are the MLL calculations for the lab-based HPGe with all three energy windows lists and the MLL calculations for the SrI_2 detector with the auto energy windows list. *Given the choice in baseline, reactor type classification is only acceptable for the four cases exceeding it and the three full-knowledge training sets.*

Next, the regression cases are shown in Section 5.4.2.3. Regardless of the energy windows list, the detector-based training sets all do the following for the MAPE and MAE: the burnup tends to far outperform the baseline, the enrichment consistently underperforms with respect to the baseline, and the time since irradiation has values along the baseline, where many are below line but close to it. The spread of absolute errors including the outliers shows that there is a significant portion of the training set that is being predicted with very large errors; the spread of outliers encompassed $3 - 17\%$ of the training set depending on the case. Studying these results with respect to the MedAE, the burnup and time since irradiation predictions do well with respect

to their baselines. *Relative to the baseline, the burnup prediction performs well over an acceptable level, enrichment prediction performs well under an acceptable level, and time since irradiation prediction has several cases that perform at or near the acceptable level.* Additionally, there is not a large difference in performance among the three energy windows lists. *This suggests that the performance is independent of the gamma spectra processing approaches.*

Post-Conclusions Discussion

Despite many of the nuclides that are included in the short and long energy windows lists having known poor simulation quality [38, 42], the burnup prediction clearly had enough information to do well. The poor enrichment prediction is likely because the detected radionuclides are not enrichment indicators, although ^{152}Eu is and should be included in the long energy windows list (although it is not guaranteed to be a well-resolved feature). Additionally, it is likely that gamma detection alone cannot resolve the enrichment, and that passive neutron detection would be required, e.g. via measuring ^{244}Cm [31]. The time steps in the training set being about 100 *days* was a limiting factor in its quality of prediction; many of the MAE values were around this level in the second experiment.

There is a clear leader throughout all of the results in terms of being a consistent best performer: the MLL calculations. The reason it can perform well, though, is because of the large number of simulations. If there were to be a training set of much higher fidelity but larger steps between the training set labels, it is possible MLL would not be a top performer.

The two scikit-learn algorithms still had some interesting variation in their prediction capabilities despite usually under-performing. Decision trees best predicted the enrichment for the external testing set scenario with SFCOMPO. For all of the 0% training set error cases in the first experiment, the scikit-learn algorithms performed

better. The MLL robustness to error is likely from the different manners in which the error was applied. The scikit-learn approach was to change the training set feature itself to a new value within some range based on the error, but the MLL approach was to include this error instead as uncertainty. In this way, it is not affecting the features directly, just the resulting uncertainty of the log-likelihood calculation. If the same approach was used, or some error existed in the measurements, MLL again might not be the best choice.

Another way in which the implementation differed was the 5-fold CV being used as the testing protocol for the scikit-learn algorithms. For 5 rounds, 20% of the training set is removed as a testing set, where eventually all 5 folds are tested. By contrast, the MLL test samples get removed a single entry at a time. If the training set were to be more sparsely populated with the same range of each label, this would be of larger concern. From the learning curves in Figure 4.12, it appears that reducing the training set by 20% would in theory not impact the MLL performance, but it is still a major difference in implementation.

It should be noted that the injection of error in the training set in the same manner as the test samples is not necessarily realistic. In a real-world scenario, the training database would be quite accurate and the testing samples might have variable error.

The nuclide-based feature sets (i.e., 29 nuclide masses and 32 nuclide activities) in both experiments were determined by external factors, i.e., the SFCOMPO database measurements and the radionuclides available in GADRAS. Since each training set is designed to be able to predict all four labels, not all features are relevant to a given label. This could add noise for a given label prediction if there are only a handful of relevant features.

With respect to the number of features affecting the prediction performance for the detector-based training sets, the overall similarity of performance among the three energy

windows lists might mean that the feature numbers are not impacting the algorithms much. Another explanation is that they all include too many features (outside of the scintillator detectors with the auto energy windows list). It is possible that even the short list with 42 features has too high of a dimension for these methods to perform well. The statement from above also applies, that if only a few features are relevant, the remaining ones only add noise.

The automatic peak searching approach results in erratic behavior for the scikit-learn algorithms. Noting the poor performance of the high energy resolution detectors, this is presumably from the added noise of including all peaks instead of just targeted peaks like the other two lists. It is still a promising approach for the lower energy resolution detectors, and possibly even the high energy resolution detectors since in theory one could filter the noise from the detector peak searches.

6.3 Future Work

When designing any study, simplifications and assumptions must be made to take a first-order look at the problem. Additionally, any good exploration is likely to generate more questions than answers. This final section covers a list of future work based on the the simplifications made and questions created.

6.3.1 Training Set Features

Simulation Fidelity

ORIGEN-Automatic Rapid Processing (ORIGEN-ARP) not only offers computational expediency, but the training set creation was able to be fully scripted and automatically carried out. The 4.5×10^5 entries in the training set were simulated in a matter of hours on a personal computer.

Section 4.1.1 discusses the fidelity of the simulations used to create the training database, and how the ground truth in this work is therefore not perfect truth. The first improvement to be made with the training set is filtering out problematic-to-simulate nuclides, such as ^{125}Sb. An exercise was carried out to compare ORIGEN-ARP simulations to an entry in SFCOMPO, and there were some nuclide discrepancies that reached 40%. It would be better to remove these from consideration if they are systematically poorly simulated.

One level beyond this would be to forego the convenience that ORIGEN-ARP provides and use higher fidelity simulations in Standardized Computer Analyses for Licensing Evaluation (SCALE). Maximizing the accuracy of the training set is worthwhile, since it is a static entity that is only simulated once.

Feature Set Study

The lists of training set features in this work were not chosen based on knowledge of nuclear reactor operation and the best signatures for the various labels being predicted (reactor type, burnup, ^{235}U enrichment, time since irradiation). Instead, they were chosen based on the nuclide availability in the SFCOMPO database and the computational gamma spectra tool, GADRAS, which gave 29 and 32 nuclides, respectively.

For the 29 nuclide mass training set, a small feature importance exercise was carried out to determine if these lists could be reduced further. This was carried out by using `SelectKBest(score_func=f_regression, k=29).fit(X,y)` (or `f_classif` for reactor type classification), sorting the scores of the features, choosing the top N features for each label, and taking the set of features together for all labels. The resulting set of best-scoring features for all the labels included most or all of the original list depending on the arbitrary decision of N. This could be further refined by combining domain knowledge of reactor physics and a more structured statistical feature importance approach. Furthermore, because mass spectrometry provides ratios of nuclides, studying

how various ratios perform versus nuclide masses or nuclide concentrations with respect to initial uranium mass would be prudent.

Gamma Spectra Processing

For the 32 nuclide activity training set, the most detectable gamma energies emerged from the short and long energy windows lists to include from 7 and 12 nuclides, respectively, shown in Table 5.4. So in this way, nuclear physics performed the feature selection. It is clear from the results that the 7- and 12-nuclide lists performed quite well, and exceeded the baseline for all four prediction scenarios. However, there were many high energy photopeaks excluded from these lists that might provide more information. These were captured via the auto peak search method, but the resulting energy windows list did not provide better results (and in some cases with k-nearest neighbors, provided worse performance).

There is much work on the topic of automatic radioisotope identification [53, 54, 55, 56, 57, 58, 59, 60] as well as the topic of elucidating typically undetectable nuclides using gamma spectroscopy with advanced detection techniques [50, 51]. So, this simple approach should be refined with a study on the latest gamma spectra processing tools and which are suitable to process a training set of 4.5×10^5 spectra within a reasonable amount of time.

In addition to the gamma spectra processing itself, there are also more advanced detection techniques being studied that could clarify peaks from nuclides that typically are not detectable. A Compton suppression measurement technique allows low-intensity peaks to be more detectable [51]. An Ultra-High Rate HPGe system operates such that peaks at the high energy end of the spectrum can be detected at large enough counts [50]. It is possible these approaches could be simulated and studied at a statistical level.

Further, adding simulated neutron detection to be able to resolve the ^{235}U enrichment would be necessary if limiting the approach to passive radioactivity detection. This is

possible in GADRAS.

6.3.2 Statistical Method Optimization

The methods in this work all have high variance, and so it would have seemed prudent to apply stronger regularization. However, hyperparameter optimization did not shift the high variance defaults much towards more bias. The resulting inability to generalize likely was a contributing factor to the large errors in the results when using the SFCOMPO database as a testing set. A closer look at optimization is warranted in tandem with the above work on the feature sets for each label.

Since the MLL approach is essentially k-nearest neighbors with $k = 1$ and a different loss metric, the MLL calculations could be expanded to include the option to have k samples averaged for a response, which would increase the bias. Another option would be to use the likelihood calculations as custom sample weights within the k-nearest neighbors algorithm. The implementation of k-nearest neighbors in this work used Manhattan distance d_i with samples weighted by $1/d_i$. This would change the sample weights to $L(M|x_{test})$, the likelihood as calculated in Equation 3.5.

It is entirely possible that optimizing the methods in this work can only improve the prediction performance by a minimal amount, and that more complex methods could be beneficial. Because of this possibility, a broader survey of statistical methods with this training set is another possible next step.

6.3.3 Serial Prediction

Figure 4.13 shows that for some combinations of prediction label and algorithm, there is an improvement in the regression performance for the PHWR class, and less so for some PWR cases. (There was no significant improvement seen for any reactor type/regression case combination for the MLL calculations.) It is important to attribute PHWR fuels,

since plutonium builds up faster in these reactors and so they pose a larger risk than other typical commercial SNF. Given the majority of the database is comprised of BWR entries, the improvement is not surprising; it is the lack of improvement in the other cases that is unexpected. Either way, it makes sense to first determine the reactor type before performing the other predictions if not using the MLL approach. One example of this being done is in Reference [47].

Performing regression after first classifying the reactor type means much care should be taken to have a robust reactor type classification algorithm. One way to ensure better classification is by implementing a study that leverages receiver operating characteristic (ROC) curves. ROC curves plot the true positive rate (vertical axis, $\frac{\text{True Positives}}{\text{True Positives + False Negatives}}$) with respect to the false positive rate (horizontal axis, $\frac{\text{False Positives}}{\text{False Positives + True Negatives}}$) for various *decision thresholds*, as shown in Figure 6.1 [62].

During prediction of a test sample, the scikit-learn classifiers calculate a probability that the sample belongs to a given class. In binary prediction, the default threshold is 0.5. A probability above this threshold means the sample will be predicted to belong to the positive class, and below that the sample will belong to the negative class. In multi-class prediction, the highest probability is chosen as the predicted class.

An ideal classifier would have values along the vertical axis and then straight across the top, meaning beyond some threshold value there are zero false positives. On the other end of the spectrum, if the classifier being used in Figure 6.1 were only choosing classes randomly, it would follow an $x = y$ diagonal line. The one shown here performs somewhere in between these two extremes.

If the algorithm returns the probabilities of class belonging instead of the predicted class, the threshold can be tuned after the fact to complete predictions. With this work, e.g., the decision threshold could be manually tuned to require a higher threshold for BWR classification. If that were the case, there could be fewer misclassifications of

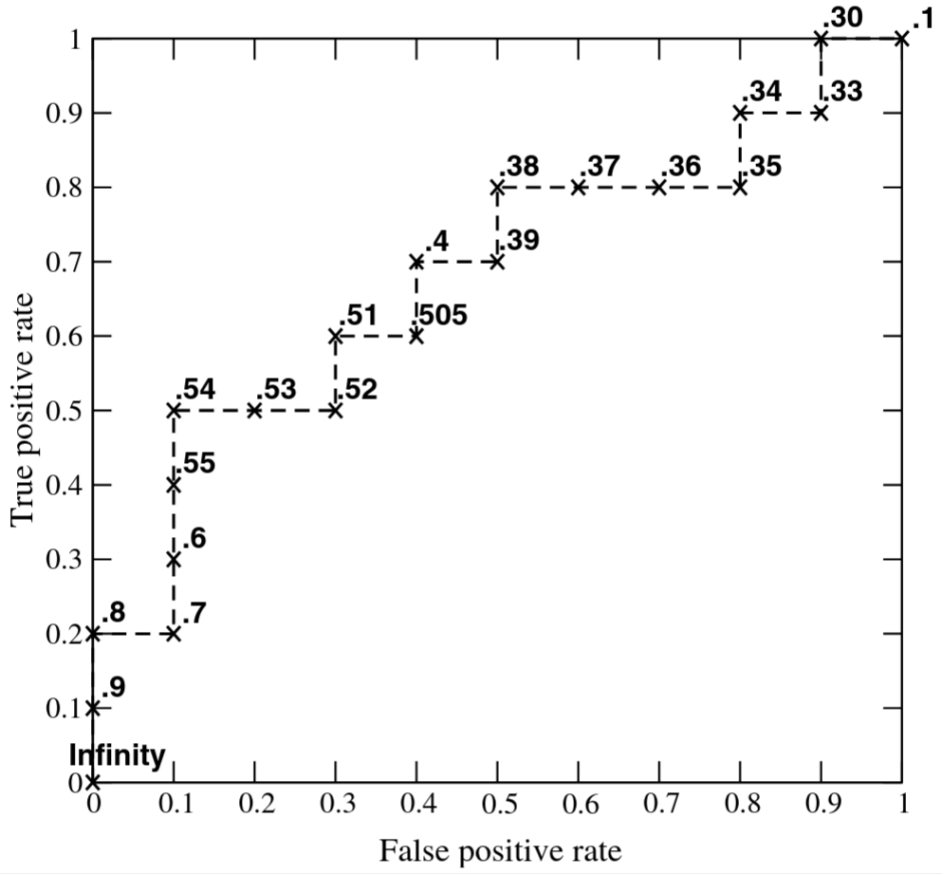


Figure 6.1: Example of an ROC curve showing the decision threshold (probability of class belonging) changing from ∞ to 0.1; this is borrowed from Reference [62].

PHWRs and PWRs as BWRs. The issues with regression of PHWRs could then be resolved if they were accurately classified as such beforehand.

6.3.4 SFCOMPO

The SFCOMPO database being used as an external test set within the framework of this methodology performed poorly in the sense that some relative errors were extremely large. Section 4.4.2 shows the results using two different methods for handling the missing measurement values: imputation with the mean value of a given nuclide, and replacing the null values with zero.

One way to improve the results is using the previously discussed serial prediction

approach, where the reactor type is first determined before the regression cases are predicted using training sets that only include entries from the predicted reactor type. This was partially touched on in Section 4.4.2 by discussing the largest relative errors disappearing when PHWRs were removed from consideration. Another way that might improve the results is with better simulation fidelity, also previously discussed.

A third approach is revisiting the method by which the missing measurements are handled. One option is to side-step the missing nuclide measurements all together. Reducing the training set to only consider uranium and plutonium isotopes accomplishes this; the majority of measurements in the training set would exist and there would be very few missing measurements. This is essentially what the zero-valued nulls with MLL calculations does, and likely why it performed the best for the reactor type and burnup predictions (and a close second for the ^{235}U enrichment prediction).

Although the zero-null values approach is likely to always perform the best with MLL calculations, there are better approaches than mean (or median) null value imputation. For example, one could take a k-nearest neighbors approach to imputation where the mean, median, or mode of the k-closest samples are used for imputation (rather than the mean, median, or mode of the entire column in that testing set). Another approach for handling missing values is outlined in Reference [8], where a novel imputation approach for the SFCOMPO database, called Monte Carlo Bayesian Database Generation, is discussed.

REFERENCES

[1] Moody, K.J., P.M. Grant, and I.D. Hutcheon. 2005. *Nuclear Forensic Analysis*. 1st ed. Boca Raton, Florida, USA: CRC Press. https://books.google.com/books?id=Q9mgDnWoPLYC.

[2] Division of Nuclear Security. 2020. IAEA Incident and Trafficking Database: Incidents of nuclear and other radioactive material out of regulatory control. Tech. Rep. 2020 Factsheet, International Atomic Energy Agency, Vienna, Austria. https://www.iaea.org/sites/default/files/20/02/itdb-factsheet-2020.pdf.

[3] May, Michael, Reza Abedin-Zadeh, Donald Barr, Albert Carnesale, Philip E. Coyle, Jay Davis, William Dorland, William Dunlop, Steve Fetter, Alexander Glaser, Ian D. Hutcheon, Francis Slakey, and Benn Tannenbaum. 2007. Nuclear Forensics: Role, State of the Art, and Program Needs. Tech. Rep., Joint Working Group of the American Physical Society and the American Association for the Advancement of Science. https://www.aaas.org/report/nuclear-forensics-role-state-art-program-needs.

[4] Weber, Chuck F, and Bryan L Broadhead. 2006. Inverse depletion/decay analysis using the scale code system. In *Transactions of the American Nuclear Society Winter Meeting*, vol. 95, 248–249. Albuquerque, NM, USA. Track 4: Nuclear and Criticality Safety Technologies.

[5] Broadhead, Bryan L, and Charles F Weber. 2010. Validation of inverse methods applied to forensic analysis of spent fuel. In *Proceedings of the Institute of Nuclear Materials Management 51st Annual Meeting*. Baltimore, MD, USA. https://www.osti.gov/scitech/biblio/1001291.

[6] Weber, Charles F, Vladimir A Protopopescu, Michael H Ehinger, Alexander A Solodov, and Catherine E Romano. 2011. Inverse solutions in spectroscopic analysis with applications to problems in global safeguards. In *Proceedings of the Institute of Nuclear Materials Management 52nd Annual Meeting*. Palm Desert, CA, USA. https://www.osti.gov/scitech/biblio/1031530.

[7] Kristo, M.J., D.K. Smith, S. Niemeyer, G.D. Dudder, and R. Abedin-Zadeh. 2006. Nuclear Forensics Support: Technical Guidance Reference Manual. Tech. Rep., IAEA, Vienna, Austria. http://www-pub.iaea.org/MTCD/Publications/PDF/Pub1241_web.pdf.

[8] Langan, Roisin T., Richard K. Archibald, and Vincent E. Lamberti. 2016. Nuclear forensics analysis with missing data. *Journal of Radioanalytical and Nuclear Chemistry* 308:687. http://link.springer.com/article/10.1007/s10967-015-4458-x.

[9] Grogan, Brandon R., and Scott Richards. 2017. Verifying Safeguards Declarations with INDEPTH: A Sensitivity Study. In *International Conference on Mathematics & Computational Methods Applied to Nuclear Science and Engineering*. Jeju, South Korea: Oak Ridge National Lab. (ORNL), Oak Ridge, TN (United States). https://www.osti.gov/biblio/1354659.

[10] Richards, Scott M., and Brandon R. Grogan. 2017. Sensitivity Study of INDEPTH for Verification of Facility Spent Nuclear Fuel Declarations. In *International High-Level Radioactive Waste Management (IHLRWM) Conference*. Charlotte, NC, USA: Oak Ridge National Lab. (ORNL), Oak Ridge, TN (United States). https://www.osti.gov/biblio/1364306.

[11] Grogan, Brandon R. (ORCID:000000018462271X), Murray (ORCID:0000000301452036) Purves, and Jordan P. (ORCID:0000000262573328)

Lefebvre. 2018. Reconstructing Reactor Operating Histories Using the INDEPTH Code. In *59th Annual Meeting of the Institute for Nuclear Materials Management*. Baltimore, Maryland, USA: Oak Ridge National Lab. (ORNL), Oak Ridge, TN (United States). https://www.osti.gov/biblio/1468182.

[12] Gey, Frederic, Chloe Reynolds, Ray Larson, and Electra Sutton. 2012. Nuclear forensics: A scientific search problem. In *Proceedings of the Lernen, Wissen, Adaption (Learning, Knowledge, Adaptation) Conference*. Dortmund, Germany. http://metadata.berkeley.edu/nuclear-forensics/Paper_9-12-12_lwa-2012-nuclear-forensics-scientific-search-problem_v7.pdf.

[13] Dayman, Kenneth, and Steven Biegalski. 2013. Feasibility of fuel cycle characterization using multiple nuclide signatures. *Journal of Radioanalytical and Nuclear Chemistry* 296:195–201. http://link.springer.com/article/10.1007%2Fs10967-012-1987-4.

[14] Nicolaou, G. 2006. Determination of the origin of unknown irradiated nuclear fuel. *Journal of Environmental Radioactivity* 86:313–318. http://nuclear.ee.duth.gr/upload/A13%20%20%20identification.pdf.

[15] ———. 2009. Identification of unknown irradiated nuclear fuel through its fission product content. *Journal of Radioanalytical and Nuclear Chemistry* 279(2):503–508. http://link.springer.com./article/10.1007%2Fs10967-007-7300-x.

[16] ———. 2008. Provenance of unknown plutonium material. *Journal of Environmental Radioactivity* 99(10):1708–1710. http://www.sciencedirect.com/science/article/pii/S0265931X08000969.

[17] ———. 2014. Discrimination of spent nuclear fuels in nuclear forensics through isotopic fingerprinting. *Annals of Nuclear Energy* 72:130–133. Technical

Note, http://www.sciencedirect.com.ezproxy.library.wisc.edu/science/article/pii/S0306454914002308.

[18] Lantzos, I., C. Kouvalaki, and G. Nicolaou. 2015. Plutonium fingerprinting in nuclear forensics of spent nuclear fuel. *Progress in Nuclear Energy* 85(Supplement C):333–336. http://www.sciencedirect.com/science/article/pii/S0149197015300329.

[19] Robel, Martin, Michael J. Kristo, and Martin A. Heller. 2009. Nuclear forensic inferences using iterative multidimensional statistics. In *Proceedings of the Institute of Nuclear Materials Management 50th Annual Meeting*. Tuscon, AZ, USA: Institute of Nuclear Materials Management. LLNL-CONF-414001, https://e-reports-ext.llnl.gov/pdf/374432.pdf.

[20] Robel, Martin, and Michael J. Kristo. 2008. Discrimination of source reactor type by multivariate statistical analysis of uranium and plutonium isotopic concentrations in unknown irradiated nuclear fuel material. *Journal of Environmental Radioactivity* 99(11):1789–1797. http://www.sciencedirect.com/science/article/pii/S0265931X08001203.

[21] Jones, Andrew, Phillip Turner, Colin Zimmerman, and J.Y. Goulermas. 2014. Machine learning for classification and visualisation of radioactive substances for nuclear forensics. In *Techniques and Methods for Safeguards, Nonproliferation and Arms Control Verification Workshop*. Portland, Oregon. https://www.researchgate.net/publication/264352908_Machine_Learning_for_Classification_and_Visualisation_of_Radioactive_Substances_for_Nuclear_Forensics.

[22] Jones, Andrew E., Phillip Turner, Colin Zimmerman, and John Y. Goulermas. 2014. Classification of spent reactor fuel for nuclear forensics. *Analytical Chemistry* 86:5399–5405. http://pubs.acs.org/doi/ipdf/10.1021/ac5004757.

[23] Osborn, Jeremy M., Evans D. Kitcher, Jonathan D. Burns, Charles M. Folden III, and Sunil S. Chirayath. 2018. Nuclear Forensics Methodology for Reactor-Type Attribution of Chemically Separated Plutonium. *Nuclear Technology* 201(1):1–10. https://doi.org/10.1080/00295450.2017.1401442.

[24] Kitcher, Evans D., Jeremy M. Osborn, and Sunil S. Chirayath. 2019. Sensitivity studies on a novel nuclear forensics methodology for source reactor-type discrimination of separated weapons grade plutonium. *Nuclear Engineering and Technology*. http://www.sciencedirect.com/science/article/pii/S1738573318309434.

[25] Osborn, Jeremy M., Kevin J. Glennon, Evans D. Kitcher, Jonathan D. Burns, Charles M. Folden, and Sunil S. Chirayath. 2019. Experimental validation of a nuclear forensics methodology for source reactor-type discrimination of chemically separated plutonium. *Nuclear Engineering and Technology* 51(2):384–393. http://www.sciencedirect.com/science/article/pii/S1738573318303036.

[26] Oak Ridge National Laboratory. 2018. SCALE: A Comprehensive Modeling and Simulation Suite for Nuclear Safety Analysis and Design. Code Suite, Oak Ridge National Laboratory, Oak Ridge, Tennessee, USA. Version 6.2.3, ORNL/TM-2005/39, Available from Radiation Safety Information Computational Center as CCC-834, http://scale.ornl.gov.

[27] Rearden, B.T., and M.A. Jessee. 2018. Ch. 5 Depletion, Activation, and Spent Fuel Source Terms. In *SCALE Code System: User Documentation*, 5.1–5.263. Oak Ridge, Tennessee, USA: Oak Ridge National Laboratory. Version 6.2.3; ORNL/TM-2005/39, https://www.ornl.gov/sites/default/files/SCALE%20Code%20System.pdf.

[28] Bowman, Stephen M, and Ian C Gauld. 2010. OrigenArp Primer: How to Perform Isotopic Depletion and Decay Calculations with SCALE/ORIGEN. Tech. Rep.

ORNL/TM-2010/43, 986788, Oak Ridge National Laboratory. http://www.osti.gov/servlets/purl/986788/.

[29] Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12(85):2825–2830. http://jmlr.org/papers/v12/pedregosa11a.html.

[30] Tarantola, Albert. 2005. *Inverse Problem Theory and Methods for Model Parameter Estimation*, chap. 1. The General Discrete Inverse Problem, 1–40. Philadelphia, Pennsylvania, USA: Society for Industrial and Applied Mathematics. http://epubs.siam.org/doi/pdf/10.1137/1.9780898717921.ch1.

[31] Skutnik, Steven E., and David R. Davis. 2016. Characterization of the non-uniqueness of used nuclear fuel burnup signatures through a Mesh-Adaptive Direct Search. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 817:7–18. http://www.sciencedirect.com/science/article/pii/S0168900216001571?via%3Dihub.

[32] Lewis, John R., Adah Zhang, and Christine M. Anderson-Cook. 2018. Comparing multiple statistical methods for inverse prediction in nuclear forensics applications. *Chemometrics and Intelligent Laboratory Systems* 175:116–129. https://www.sciencedirect.com/science/article/abs/pii/S0169743917304240.

[33] Langley, Pat. 2011. The changing science of machine learning. *Machine Learning* 82:275–279. https://link.springer.com/article/10.1007%2Fs10994-011-5242-y.

[34] Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2001. *The Elements of Statistical Learning*. Springer Series in Statistics, New York, NY, USA: Springer New York Inc.

[35] Michel-Sendis, F., I. Gauld, J. S. Martinez, C. Alejano, M. Bossant, D. Boulanger, O. Cabellos, V. Chrapciak, J. Conde, I. Fast, M. Gren, K. Govers, M. Gysemans, V. Hannstein, F. Havlůj, M. Hennebach, G. Hordosy, G. Ilas, R. Kilger, R. Mills, D. Mountford, P. Ortego, G. Radulescu, M. Rahimi, A. Ranta-Aho, K. Rantamäki, B. Ruprecht, N. Soppera, M. Stuke, K. Suyama, S. Tittelbach, C. Tore, S. Van Winckel, A. Vasiliev, T. Watanabe, Toru Yamamoto, and Toshihisa Yamamoto. 2017. SFCOMPO-2.0: An OECD NEA database of spent nuclear fuel isotopic assays, reactor design specifications, and operating data. *Annals of Nuclear Energy* 110(Supplement C):779–788. `http://www.sciencedirect.com/science/article/pii/S0306454917302104`.

[36] Gauld, Ian C., Mark L. Williams, Franco Michel-Sendis, and Jesus S. Martinez. 2017. Integral nuclear data validation using experimental spent nuclear fuel compositions. *Nuclear Engineering and Technology* 49(6):1226–1233. `https://linkinghub.elsevier.com/retrieve/pii/S1738573317303054`.

[37] Chadwick, M.B., M. Herman, P. Obložinský, M.E. Dunn, Y. Danon, A.C. Kahler, D.L. Smith, B. Pritychenko, G. Arbanas, R. Arcilla, R. Brewer, D.A. Brown, R. Capote, A.D. Carlson, Y.S. Cho, H. Derrien, K. Guber, G.M. Hale, S. Hoblit, S. Holloway, T.D. Johnson, T. Kawano, B.C. Kiedrowski, H. Kim, S. Kunieda, N.M. Larson, L. Leal, J.P. Lestone, R.C. Little, E.A. McCutchan, R.E. MacFarlane, M. MacInnes, C.M. Mattoon, R.D. McKnight, S.F. Mughabghab, G.P.A. Nobre, G. Palmiotti, A. Palumbo, M.T. Pigni, V.G. Pronyaev, R.O. Sayer, A.A. Sonzogni, N.C. Summers, P. Talou, I.J. Thompson, A. Trkov, R.L. Vogt, S.C. van der Marck, A. Wallner, M.C. White, D. Wiarda, and P.G. Young. 2011. Endf/b-

vii.1 nuclear data for science and technology: Cross sections, covariances, fission product yields and decay data. *Nuclear Data Sheets* 112(12):2887–2996. http://www.sciencedirect.com/science/article/pii/S009037521100113X.

[38] Radulescu, Georgeta, Ian C Gauld, and Germina Ilas. 2010. SCALE 5.1 Predictions of PWR Spent Nuclear Fuel Isotopic Compositions. Tech. Rep. ORNL/TM-2010/44, 983556. http://www.osti.gov/servlets/purl/983556/.

[39] Leal, L. C., O. W. Hermann, S. M. Bowman, and C. V. Parks. 1998. Arp: Automatic rapid process for the generation of problem-dependent sas2h/origen-s cross-section libraries. Tech. Rep., Lockheed Martin Energy Research Corporation and Oak Ridge National Laboratory. ORNL/TM-13584, https://digital.library.unt.edu/ark:/67531/metadc684948/.

[40] Gauld, I. C. 2003. Mox cross-section libraries for origen-arp. Tech. Rep., UT-Battelle, LLC and Oak Ridge National Laboratory. ORNL/TM-2003/2, https://www.nrc.gov/docs/ML0410/ML041040058.pdf.

[41] Skutnik, Steven E. 2017. Proposed re-evaluation of the 154Eu thermal (n, γ) capture cross-section based on spent fuel benchmarking studies. *Annals of Nuclear Energy* 99:80–107. https://www-sciencedirect-com/science/article/pii/S0306454916305710.

[42] ———. 2017. ORIGEN-based Nuclear Fuel Inventory Module for Fuel Cycle Assessment: Final Project Report. Tech. Rep. DOE/NEUP–13-5415, 1364128. http://www.osti.gov/servlets/purl/1364128/.

[43] Virtanen, Pauli, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman,

Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17:261–272. `https://rdcu.be/b08Wh`.

[44] Harris, Charles R., K. Jarrod Millman, Stéfan J van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. 2020. Array programming with NumPy. *Nature* 585:357–362. `https://www.nature.com/articles/s41586-020-2649-2`.

[45] Pordes, Ruth, Don Petravick, Bill Kramer, Doug Olson, Miron Livny, Alain Roy, Paul Avery, Kent Blackburn, Torre Wenaus, Frank Würthwein, Ian Foster, Rob Gardner, Mike Wilde, Alan Blatecky, John McGee, and Rob Quick. 2007. The open science grid. In *J. phys. conf. ser.*, vol. 78 of *78*, 012057.

[46] Sfiligoi, Igor, Daniel C Bradley, Burt Holzman, Parag Mhashilkar, Sanjay Padhi, and Frank Wurthwein. 2009. The pilot way to grid resources using glideinwms. In *2009 wri world congress on computer science and information engineering*, vol. 2 of *2*, 428–432.

[47] O'Neal, Patrick J., and Sunil S. Chirayath. 2021. Separated Plutonium Attribution using Machine Learning Techniques. In *Technology and Policy Advancements in Nuclear Nonproliferation, Transactions of the American Nuclear Society*, vol. 124,

428–431. American Nuclear Society. https://epubs.ans.org/?a=49623&_ga=2.26421434.644585875.1626725786-1803871342.1626725786.

[48] Hunter, John D. 2007. Matplotlib: A 2d graphics environment. *Computing in Science Engineering* 9(3):90–95. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4160265.

[49] Horne, Steven M., Gregory G Thoreson, Lisa A. Theisen, Dean J. Mitchell, Lee Harding, and Sean O'Brien. 2018. Gamma Detector Response and Analysis Software. GADRAS Version 18 User's Manual SAND2018-12436, Sandia National Laboratories, Albuquerque, New Mexico, USA. API provided from developers using Version 18.8.0; Official Use Only.

[50] Fast, James E., Jeffrey W. Chenault, Brian D. Glasgow, Douglas C. Rodriguez, Brent A. VanDevender, and Lynn S. Wood. 2014. Spent Nuclear Fuel Measurements. Tech. Rep. PNNL-23561, Pacific Northwest National Lab. (PNNL), Richland, WA (United States). https://www.osti.gov/biblio/1159319-spent-nuclear-fuel-measurements.

[51] S. Bender, B. Heidrich, and K. Ünlü. 2015. Compton suppressed LaBr3 detection system for use in nondestructive spent fuel assay. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 784:474–481. https://www.sciencedirect.com/science/article/abs/pii/S0168900214014661.

[52] Smith, Susan K. (ORCID:0000000263845743), Andrew D. (ORCID:0000000153035424) Nicholson, Stephen Croft, and Greg Nutter. 2018. High-Resolution Gamma Spectrometry Data Set for High Burnup Pressurized Water Reactor Used Fuel. Tech. Rep., Oak Ridge National Lab. (ORNL), Oak Ridge, TN (United States). https://www.osti.gov/biblio/1468177.

[53] Burr, Tom, and Michael Hamada. 2009. Radio-Isotope Identification Algorithms for NaI γ Spectra. *Algorithms* 2(1):339–360. http://www.mdpi.com/1999-4893/2/1/339.

[54] He, Jianping, Xiaobin Tang, Pin Gong, Peng Wang, Liangsheng Wen, Xi Huang, Zhenyang Han, Wen Yan, and Le Gao. 2018. Rapid radionuclide identification algorithm based on the discrete cosine transform and BP neural network. *Annals of Nuclear Energy* 112:1–8. https://www.sciencedirect.com/science/article/pii/S0306454917303146.

[55] Sullivan, Clair J., Scott E. Garner, Krastan B. Blagoev, and Doug L. Weiss. 2007. Generation of customized wavelets for the analysis of γ-ray spectra. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 579(1):275–278. https://www.sciencedirect.com/science/article/abs/pii/S0168900207006286.

[56] Sullivan, C.J., and J. Stinnett. 2015. Validation of a Bayesian-based isotope identification algorithm. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 784:298–305. https://www.sciencedirect.com/science/article/abs/pii/S0168900214014272.

[57] Kamuda, M., J. Stinnett, and C. J. Sullivan. 2017. Automated Isotope Identification Algorithm Using Artificial Neural Networks. *IEEE Transactions on Nuclear Science* 64(7):1858–1864. https://ieeexplore.ieee.org/document/7898423.

[58] Stinnett, J., C. J. Sullivan, and H. Xiong. 2017. Uncertainty Analysis of Wavelet-Based Feature Extraction for Isotope Identification on NaI Gamma-Ray Spectra. *IEEE Transactions on Nuclear Science* 64(7):1670–1676. https://ieeexplore.ieee.org/document/7866851.

[59] Dayman, Kenneth, Steven Biegalski, Derek Haas, Amanda Prinke, and Sean Stave. 2016. Evaluation of independent and cumulative fission product yields with gamma spectrometry. *Journal of Radioanalytical and Nuclear Chemistry* 307(3):2239–2245. https://doi.org/10.1007/s10967-015-4491-9.

[60] Dayman, Kenneth, and Steven Biegalski. 2016. Automatic identification and quantification of radionuclides in gamma spectra using numerical optimization. *Journal of Radioanalytical and Nuclear Chemistry* 307(3):2247–2252. https://doi.org/10.1007/s10967-015-4492-8.

[61] Bates, Cameron R., Elliott Biondo, Kathryn Huff, Kalin Kiesling, Anthony Scopatz, Robert Carlsen, Andrew Davis, Matthew Gidden, Tim Haines, Joshua Howland, Blake Huff, Kevin Manalo, Arielle Opotowsky, Rachel Slaybaugh, Eric Relson, Paul Romano, Patrick Shriwise, John D. Xia, Paul Wilson, and Julie Zachman. 2014. PyNE Progress Report. *Transactions of the American Nuclear Society* 111: 1165–1168. 2014 ANS Winter Meeting and Nuclear Technology Expo ; Conference date: 09-11-2014 Through 13-11-2014.

[62] Fawcett, Tom. 2006. An introduction to ROC analysis. *Pattern Recognition Letters* 27(8):861–874. https://www.sciencedirect.com/science/article/abs/pii/S016786550500303X.