N-SLOPE: A ONE-CLASS CLASSIFICATION ENSEMBLE FOR NUCLEAR FORENSICS

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Computer Science

by

Justin Kehl

June 2018

COMMITTEE MEMBERSHIP

TITLE:	N-SLOPE: A One-Class Classification Ensemble For Nuclear Forensics
AUTHOR:	Justin Kehl
DATE SUBMITTED:	June 2018
COMMITTEE CHAIR:	Lubomir Stanchev, Ph.D. Professor of Computer Science
COMMITTEE MEMBER:	Franz Kurfess, Ph.D. Professor of Computer Science
COMMITTEE MEMBER:	Dennis Sun, Ph.D. Assistant Professor of Statistics Assistant Professor of Computer Science by Courtesy

ABSTRACT

N-SLOPE: A One-Class Classification Ensemble For Nuclear Forensics

Justin Kehl

One-class classification is a specialized form of classification from the field of machine learning. Traditional classification attempts to assign unknowns to known classes, but cannot handle novel unknowns that do not belong to any of the known classes. One-class classification seeks to identify these outliers, while still correctly assigning unknowns to classes appropriately. One-class classification is applied here to the field of nuclear forensics, which is the study and analysis of nuclear material for the purpose of nuclear incident investigations. Nuclear forensics data poses an interesting challenge because false positive identification can prove costly and data is often small, high-dimensional, and sparse, which is problematic for most machine learning approaches.

A web application is built using the R programming language and the shiny framework that incorporates N-SLOPE: a machine learning ensemble. N-SLOPE combines five existing one-class classifiers with a novel one-class classifier introduced here and uses ensemble learning techniques to combine output. N-SLOPE is validated on three distinct data sets: Iris, Obsidian, and Galaxy Serpent 3, which is an enhanced version of a recent international nuclear forensics exercise. N-SLOPE achieves high classification accuracy on each data set of 100%, 83.33%, and 83.33%, respectively, while minimizing false positive detection rate to 0% across the board and correctly detecting every single novel unknown from each data set. N-SLOPE is shown to be a useful and powerful tool to aid in nuclear forensic investigations.

ACKNOWLEDGMENTS

Thanks to:

- Martin Robel of Lawrence Livermore National Laboratory, for mentoring and supporting my research.
- Naomi Marks of Lawrence Livermore National Laboratory, for providing validation data.
- My family and friends, for their love and encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

Classification is a well-known problem in the field of machine learning that has been studied extensively. Traditional classification algorithms learn from a training set of knowns and attempt to place unknowns into one of the known classes. An interesting problem arises when an unknown is encountered that has never been seen before and thus does not belong to any of the known classes. This problem is addressed by one-class classification.

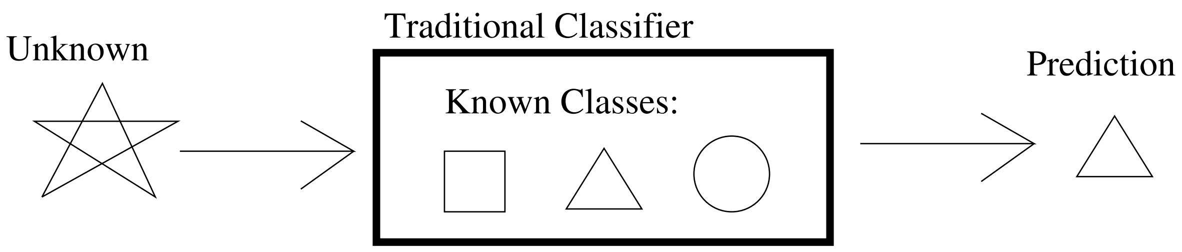


Figure 1.1: Incorrect behavior of a traditional classifier when presented with a novel unknown

1.1 One-class Classification

One-class classification was first described in 1996 by Moya and Hush [30]. Moya and Hush defined generalization as the "ability to classify arbitrary patterns correctly after training is complete" and described three types of generalization: *within-class*, *between-class*, and *out-of-class* generalization. *Within-class* generalization is the ability to recognize patterns as belonging to a known class, whereas distinguishing between known classes and recognizing when a pattern belongs to none of the known classes describes *between-class* and *out-of-class* generalization, respectively.

According to Moya and Hush, “Only those classifiers that exhibit all three types of generalization can function as one-class classifiers.” Traditional classifiers act as discriminators exhibiting only within-class and between-class generalization, whereas true one-class classifiers act as detectors and exhibit all three types of generalization [30]. In simpler terms, a one-class classifier is able to determine whether an unknown belongs to any of the known classes or the unknown belongs to a separate class altogether.

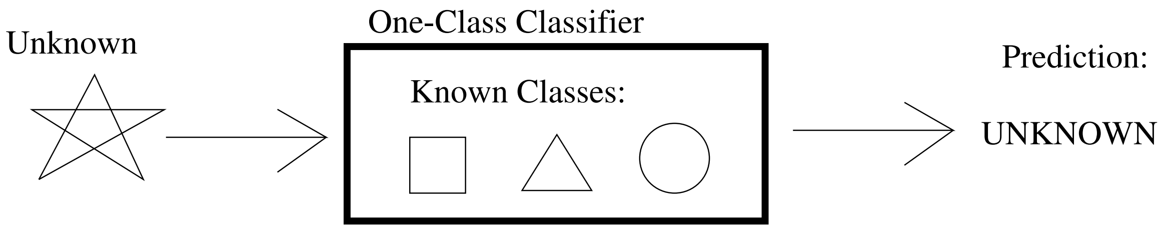


Figure 1.2: Correct behavior of a one-class classifier when presented with a novel unknown

1.2 Nuclear Forensics

One-class classification has many applications and here is applied to the field of nuclear forensics, which is the “science, techniques, and analyses of nuclear materials and their near environments for information pertinent to nuclear incident investigations by law enforcement and intelligence agencies” [29]. Nuclear forensics is a particularly challenging field for classification because data is often high-dimensional, limited, sparse, and restricted. Furthermore, it is unlikely that any training set will be complete, meaning novel unknowns are likely to occur, which presents a significant problem for traditional classification. Misclassification, especially false-positive identification, of nuclear material can have serious consequences, making accuracy a prime concern.

1.3 Approach

A machine learning toolbox is developed in the R programming language [33] to aid in nuclear forensics investigations. Several different classifiers are employed including Partial Least Squares with Discriminant Analysis (PLS-DA), Soft Independent Modeling of Class Analogies (SIMCA), One-Class Support Vector Machine (OC-SVM), Local Outlier Factor (LOF), Extreme Learning Machine (ELM), and a novel approach combining ideas from Principal Component Analysis (PCA), LOF, Nearest Centroid, and SIMCA. PLS-DA and SIMCA are chosen both for performance and domain-specific application [24]. OC-SVM, LOF, and ELM are chosen for their exceptional performance [2, 22, 15] as one-class classifiers. The toolbox analyzes the output from each of these classifiers and uses ensemble learning techniques to determine the final class of a given unknown. Combining classifiers increases robustness, generalizability, and accuracy at the potential cost of efficiency [17].

1.4 Novel Algorithm

The novel algorithm presented here and included in N-SLOPE assumes that data is roughly spherical with no strong shape, as is often the case with high-dimensional data. PCA is performed first to reduce dimensionality, then statistics are calculated for each class. Drawing from concepts in LOF and Nearest Centroid, these statistics include the mean and standard deviation of distances between each point and its class centroid. Once summary statistics have been calculated, unknowns are plotted in the same space and the distance to each class centroid is computed. The algorithm acts as a soft classifier, like SIMCA, by assigning class membership to each unknown that falls within an acceptance threshold based on distance to centroid. In this way, unknowns may be assigned to zero, one, or more of the known classes.

1.5 Contribution Summary

The primary work product is a web application that utilizes machine learning for classification. The major contributions are as follows:

- N-SLOPE toolbox for robust one-class classification;
- Research and implementation of five existing one-class classifiers;
- Design and implementation of one novel one-class classifier;
- Ensemble learning for automated prediction results;
- Model cross-validation for predicted classification accuracy.

1.6 Overview

Background information on the project domain is listed in Chapter 2. Research associated with the machine learning algorithms described above is found in Chapter 3. Chapter 4 describes the web application and overall system design. Algorithm and ensemble learning implementation details are outlined in Chapter 5. Chapter 6 presents the methodology and results of validation on three data sets including a nuclear forensics data set, while a summary of the thesis and suggestions for future improvements are provided in Chapters 7 and 8, respectively.

Chapter 2

BACKGROUND

2.1 Chemometrics

Chemometrics combines mathematical and statistical techniques to analyze and increase understanding of chemical data. Traditionally, this involves performing calculations and building models or simulations of various chemical phenomena based on empirical measurements. Modern approaches incorporate machine learning techniques such as SIMCA and PLS-DA that can accommodate the difficult high-dimensionality and low sample size nature of the data that is so prevalent in chemical sciences [24].

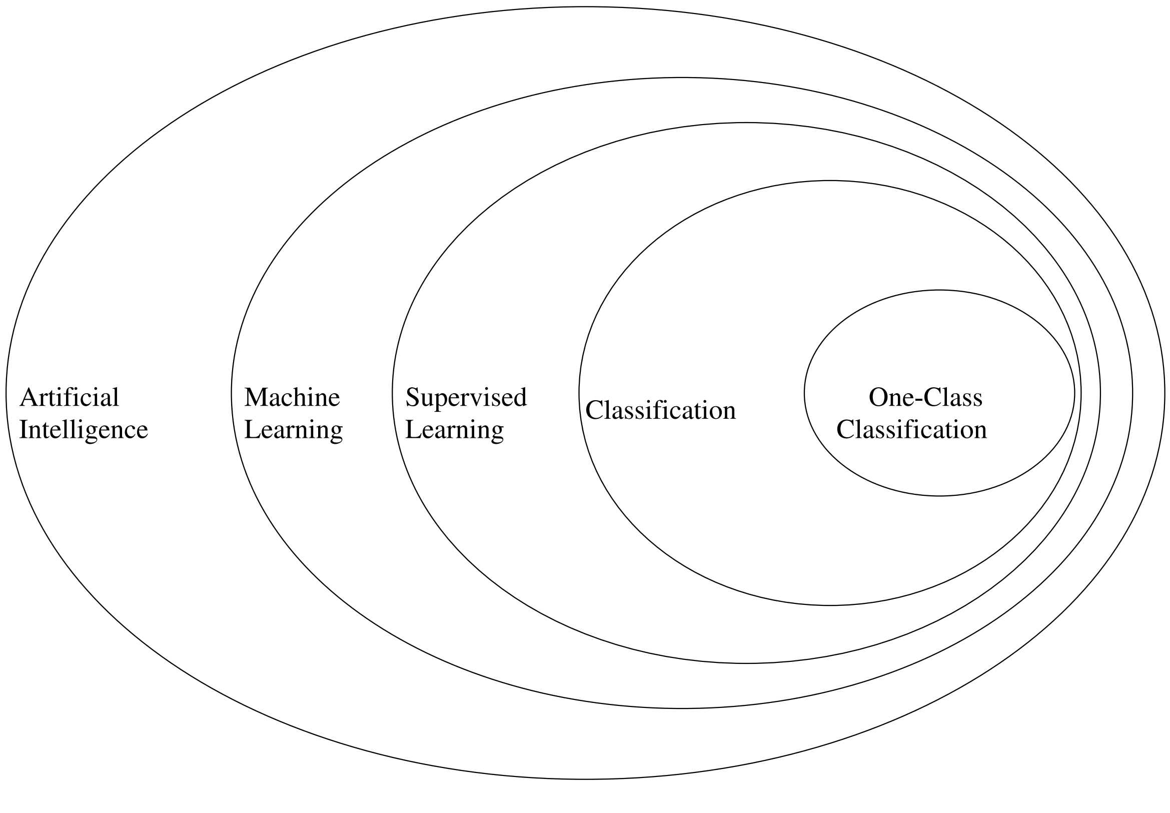


Figure 2.1: One-Class Classification in the Artificial Intelligence hierarchy

2.2 Artificial Intelligence

Artificial intelligence is a broad concept that involves machines thinking, learning, or moving intelligently towards a goal. There are many different categories and subsets of problems within the field that all involve machines solving problems that require some degree of intelligence.

2.3 Machine Learning

Machine learning is a subset of artificial intelligence that involves machines gathering knowledge and drawing conclusions about a problem without explicit programmatic instructions. Machine learning typically involves building models to recognize latent patterns present in large amounts of data and using these models to make predictions. Machine learning is often able to identify trends in data that are not obvious to human examination.

2.4 Supervised Learning

Supervised learning is a class of machine learning in which models are built and trained on known, labeled data. Labeled training data provides a baseline of knowledge for the models to work from and allows for more accurate predictions. Classification and one-class classification are specific problems within supervised learning.

2.5 Classification

Classification involves learning to recognize distinct categories, known as classes, from a set of labeled training data where the labels indicate class membership. When new data is encountered, classification assigns the new data to whichever known class it most resembles. In this way, classification aims to match and categorize data.

2.6 One-Class Classification

One-class classification has the same goal as classification, but with one important difference: it aims to assign unknowns to the known class they most resemble, or to none of the known classes if the unknown differs significantly from the known classes. This is a more challenging problem than traditional classification, but it is also more powerful in that novel unknowns, which do not belong to any of the known classes, may be detected rather than incorrectly assigned to one of the known classes.

2.7 Ensemble Learning

Ensemble learning aims to better solve problems by combining multiple different approaches to solving the same problem. For classification, this involves training separate classifiers on the same problem and then analyzing and combining their results to form one unified decision. Combining results improves overall accuracy and can be done in a variety of ways [18]. Ensemble learning is a powerful tool for improving classification performance [17] and is most effective when the classifiers reveal unique and complementary information about the problem [1].

2.8 *k*-Fold Cross-Validation

Cross-validation is used to benchmark the performance of an algorithm or model by measuring accuracy on training data alone. With k-fold cross-validation, the training set is divided into k similarly-sized chunks known as folds. One of these folds is used as artificial testing data (where the true class is known) and the remaining folds are used as training data. A model is trained using the generated training set before predicting the artifical testing set and then the predictions are compared against the true classes to produce an accuracy measurement. This is repeated with each fold being used once as the artifical testing set and the accuracy is averaged across folds. This technique cannot be used to validate a model's out-of-class generalization ability as it operates soley on training data and therefore cannot contain novel unknowns to predict.

2.9 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a statistical technique for reducing the dimensionality of a dataset while preserving the characteristics and variability of the data. PCA operates by transforming existing attributes into a different vector space and extracting new attributes, known as principal components, that explain the variation in the data. PCA cannot produce more principal components than the original dimensionality of the data or one less than the number of records in the dataset, whichever is lower. In general, only the first several principal components are retained for further calculation as these provide the most accurate representation of the data with the lowest dimensionality possible [12]. Machine learning algorithms typically struggle with high-dimensional data, which leads to over-fitting, poor computational performance, and decreased accuracy. This makes PCA an effective solution for high-dimensional machine learning [12].

2.10 *k*-Nearest Neighbors (KNN)

The k-Nearest Neighbors (KNN) algorithm is a very simple, yet surprisingly effective, machine learning technique. KNN acts as a traditional classifier by plotting all knowns and unknowns in the same space. For each unknown, the algorithm examines the k known points closest to the unknown and uses a simple majority vote to assign a class to the unknown. The optimal value of k depends on the data set with the goal being to classify unknowns as the class they most resemble. KNN is a density-based approach and as such struggles with small datasets, high dimensionality, outliers, and noisy data.

2.11 Nearest Centroid

Nearest Centroid is a simple classifier that assigns a given unknown to the class of the closest known centroid, which is the center or mean of a known class.

2.12 Support Vector Machine (SVM)

A Support Vector Machine (SVM) is a binary classifier that attempts to distinguish between two known classes. This is done by mapping the training data to a sample space and attempting to create a hyperplane boundary of maximal margin between the two known classes. Unknowns are then mapped to this same space and classified based on which side of the boundary they appear. Traditionally, SVM creates a linear boundary which does not perform well on non-linearly separable data. To combat this, SVM can utilize a kernel function that translates the original sample space into a higher dimensional feature space [4]. SVM then attempts to create a linear boundary in this new feature space which appears as a nonlinear boundary in the original sample space.

2.13 Neural Network (NN)

A Neural Network (NN) is a complex system of interconnected nodes that is modeled after synaptic communications between neurons in the human brain. The network is organized into layers of nodes: one input layer, one output layer, and some number of hidden layers between the input and output layers. Each layer is composed of some number of nodes running activation functions and each node has weighted connections to nodes in other layers. For classification purposes, the input layer usually consists of one node for each attribute and the output layer usually consists of one node for each possible classification. The network is trained by passing individual data points to the input layer, allowing values to propogate through the network, observing results in the output layer, and modifying connection weights and activation function hyperparameters to improve overall classification accuracy. There are many varieties of NN that incorporate different learning styles and network characteristics which allow them to model arbitrarily complex relationships between attributes. NN are often viewed as "black-boxes" due to the complex, dynamic, internal structure that makes it difficult to trace and verify results.

Chapter 3

RELATED RESEARCH

3.1 Partial Least Squares with Discriminant Analysis (PLS-DA)

Partial Least Squares with Discriminant Analysis (PLS-DA) is a specific case of Partial Least Squares (PLS) where the response variable is categorical. Madden and Howley described PLS as "a two-step multivariate regression method, which first reduces the data using PCA (using concentration information to extract the PC [Principal Component] scores) and then performs linear regression on the PC [Principal Component] scores" [24]. PLS-DA differs by additionally creating class boundaries associated with a confidence threshold after performing the linear regression. PLS-DA is commonly used for chemometric analysis [24, 44] because it excels with high-dimensional, low record count data, particularly when there exist linear relationships between attributes. SVM and NN have been shown to outperform PLS-DA, particularly on data sets with many records and nonlinear relationships between attributes [44, 43]. Despite this, PLS-DA is a proven chemometric standard and requires minimal tuning compared to its opponents, which justifies its inclusion in this application.

3.2 Soft Independent Modeling of Class Analogies (SIMCA)

Soft Independent Modeling of Class Analogies (SIMCA) is a statistical technique for multivariate classification. SIMCA operates by first performing PCA on each class in the training set independently, then constructing confidence threshold boundaries for each class based on the residual of standard deviation and distance, and then finally mapping unknowns into the component space for classification. SIMCA operates as a "soft" classifier in that it can map unknowns to zero, one, or more distinct classes [7]. Because of this, SIMCA is often outperformed by PLS-DA classification [7] which itself can be outperformed by more traditional machine learning techniques [44, 43]. However, SIMCA excels with small, high-dimensional data sets [37] and acts as a particularly strong within-class classifier by allowing unknowns to be assigned to multiple known classes [7, 37]. SIMCA is included in this application for its unique within-class classification abilities, aptitude for dealing with the difficult data domain present in chemometrics, and because it is one of the most common techniques used for chemical spectral data analysis [37].

3.3 One-Class Support Vector Machine (OC-SVM)

One-Class Support Vector Machine (OC-SVM), also known as Support Vector Data Description or Support Vector Domain Description, is a natural modification of the traditional SVM. Instead of creating a decision boundary between known classes as in traditional SVM, OC-SVM creates either a closed hypersphere of minimal volume around the entire training set [39] or a hyperplane between the origin and training set [35]. Unknowns are mapped to this space and considered novel if they fall outside the decision boundary. OC-SVM is flexible with many optimizations, such as training with novel unknowns (if examples exist) that must fall outside of the decision boundary, kernel functions to alter the behavior, shape, and dimensionality of the decision boundary to better describe different data characteristics, and ignoring outliers in the training set when building the model [39]. Like most machine learning techniques, OC-SVM struggles with outliers and smaller, high-dimensional data sets, although optimizations can be made to improve results under these conditions [2, 39]. Overall, OC-SVM has been shown to have excellent performance [2, 15, 39, 38, 26] and is often seen as the default one-class classifier in a category of its own [17]. These qualifications make OC-SVM an exceptional candidate for this application.

3.4 Local Outlier Factor (LOF)

Local Outlier Factor (LOF) is an algorithmic technique for quantifying how strongly a given unknown resembles an outlier. LOF resembles KNN initially, but differs in that the distance between an unknown and each of its k-nearest neighbors is compared to the average distance between each neighbor's k-nearest neighbors and this ratio is averaged to produce an outlier factor for each unknown [6]. Unlike other outlier detection methods which attempt to determine whether or not an unknown is an outlier in a binary fashion, LOF assigns a floating-point value, starting at one, to each unknown with higher values indicating a greater degree of outlying characteristics [6]. This non-binary approach can be useful for determining the difference between a weak, possible outlier and a strong, definite outlier, although a final classification is not performed. Because LOF is density-based, it struggles with small, high-dimensional, loosely-clustered data sets [15], but its local-density approach excels with between-class generalization and is able to identify outliers often missed by other approaches [6]. Interestingly, LOF has also been shown to perform well compared to OC-SVM [2] and in some cases a relationship may exists where LOF performs well on data sets where OC-SVM struggles and vice versa [15]. LOF is included in this application for its between-class generalization and complementary potential with OC-SVM, but challenges from the difficult data domain are expected.

3.5 Extreme Learning Machine (ELM)

Extreme Learning Machine (ELM) as applied to one-class classification is a very simple feed-forward NN without back-propagation that contains one hidden layer and a single node in the output layer [22]. ELM attempts to minimize both error and the norm of output weights during training to optimize performance [22]. ELM requires minimal tuning, outperforms the former NN-based one-class classifier known as autoencoder [25] in both computational speed and classification accuracy, and has been shown to outperform OC-SVM in some cases [22]. The ability of ELM to utilize nonlinear kernels also gives it a distinct advantage over linear approaches like PLS-DA and SIMCA when working with nonlinear data [43]. ELM is included in this application as a complementary NN approach in the ensemble and because of its high speed and performance potential. However, the minimal tuning, simplicity of the network, and black-box nature of NN in general suggest caution and verification of results.

Chapter 4

WEB APPLICATION

4.1 Overview

The primary work product of this thesis is a web application for comprehensive data analysis of nuclear forensics data. The application is written in the R programming language [33] and runs using the shiny framework [8]. The application is divided into three feature groups with functionality centered around data manipulation, visualization, and classification, respecitvely. Features of interest are detailed below. N-SLOPE is the core classification tool and is described here only as it relates to the application. This application has been developed for, and remains the property of, Lawrence Livermore National Laboratory and the United States Department of Energy.

This application is designed to aid investigations by combining commonly used visualization and machine learning tools into a single, web-based, interactive product. Additional data manipulation features and advanced classifiers provide automated results, which reduce the amount of time spent manually examining and processing data. Furthermore, this application allows users with minimal domain-specific knowledge to aid in nuclear forensics investigations by presenting output and results in an understandable form.

4.2 Implementation

4.2.1 R

R [33] is a programming language designed for statistical computing and graphical output with a large number of packages centered around data analytics. It was chosen for this project due to public package support and maintainability in the form of conciseness and familiarity. Existing tools for nuclear forensics may be proprietary, which makes modification and modernization difficult and expensive. Using public packages in R provides flexibility and allows the application to be modified easily and updated with new analytics as needed. Additionally, R is a high-level language that is quick to learn and consists of short, powerful syntax which produces a smaller codebase.

4.2.2 Shiny

Shiny [8] is an R package and framework that allows developers to write solely in R with Shiny automatically generating the HTML, CSS, and JS necessary to make an interactive web app. Shiny apps are divided into two main sections: the front-end UI and the back-end server. The UI is where developers specify which input widgets and output graphs, tables, and text appear and how the interface should be organized. Shiny provides a large number of standard input and output widgets, which makes front-end development quick. The server is where the main computations occur and where developers specify what to display in the various UI output widgets. Shiny connects UI widgets with server output using reactivity, which essentially links items through dependencies such that changing a value will automatically signal anything that depends on that value to react and recompute. By default, reactions are lazy meaning that dependents are not recomputed immediately when a dependency changes, but instead when the dependent becomes visible to the user. Lazy loading makes the app feel a bit more responsive at the cost of delays when requesting more computationally-intensive content.

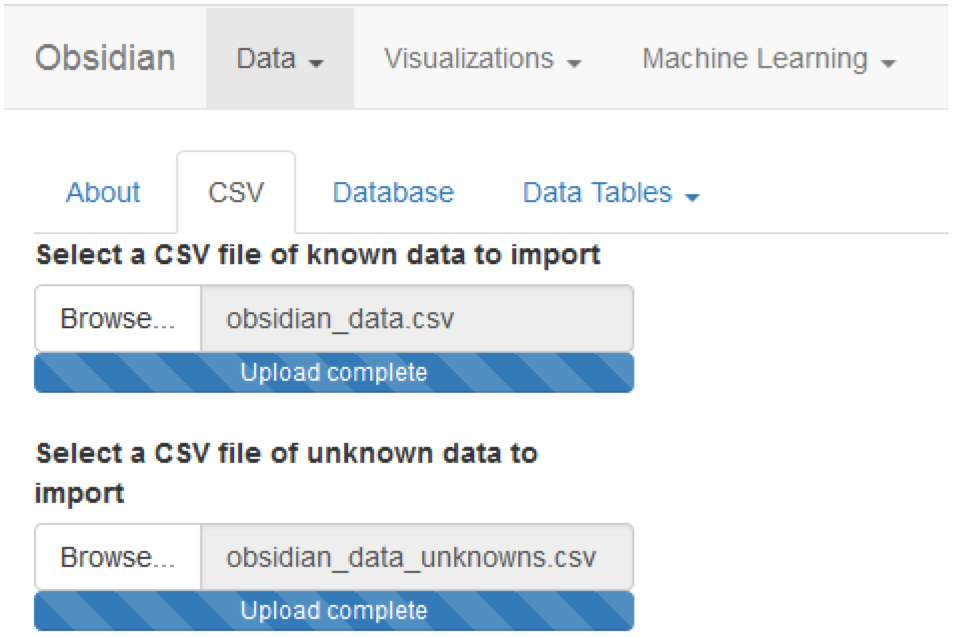
4.3 Data Features

4.3.1 Imputation

It is common for chemometric data to contain missing values for certain attributes due to the fact that most of these values are the result of various chemical tests or experiments and it is not always possible to perform the same tests on every sample. This makes chemometric data particularly difficult to work with because it is not unusual to be missing 10-30 percent or more of the possible data values. Despite the missing values, samples cannot be discarded due to the typically small size of these data sets. Instead, this problem is addressed with imputation, which performs statistical analysis on the row(s) and column(s) containing the missing value(s) and attempts to calculate a plausible substitute value(s) that will not alter the characteristics of the data set. This application uses the `mice package` [41] to automatically impute missing values as needed for the machine learning tools.

4.3.2 Import

Data must be imported and must consist of a labeled training set and a testing set with matching column headers. The first and second column must be a unique positive integer identifying the sample and a positive integer or string labeling the class, respectively. Data can be imported as Comma Separated Value (CSV) files or directly from MySQL databases using the `RMySQL package` [32]. If the training data contains any missing values, imputation is performed and statistically uninteresting data is inserted. Once data has been imported, it can be viewed directly in the app.



Select a CSV file of known data to import

obsidian_data.csv

Select a CSV file of unknown data to import

obsidian_data_unknowns.csv

Figure 4.1: Import functionality with Obsidian data

4.3.3 Filtering

Training data with many classes or highly overlapping classes can be difficult to classify accurately, particularly for algorithms like PLS-DA. Pruning classes from the training set allows the algorithm to re-identify latent variables, recompute class boundaries, and in some cases separate previously overlapping classes which boosts classification accuracy. The application contains a tool for selecting classes from the training set to filter out for visualization and machine learning purposes. Filtering is not performed automatically because algorithms like OC-SVM that require larger data sets may perform poorly. Instead, filtering is a manual option available to the user to aid in investigations.

4.4 Visualization Features

The application includes several visualization tools that display training and testing data in interesting ways. Foremost among these tools is the radar plot, also known as a spider plot. A radar plot is a two-dimensional graph that displays multiple attributes at once. In this case, each class is averaged and displayed on the plot along with a single unknown from the test set. Each class average can be toggled on or off to facilitate viewing and a single unknown can be selected at a time to compare against the class averages. This plot provides quick feedback on how similar a given unknown appears compared to the known classes, with an emphasis being placed on the overall shape or relative values of attributes rather than particular numbers. The radar plot is good for initial estimates and continued investigation as it can be used to predict class membership, identify obvious outliers, and verify or dispute machine learning predictions. This application uses the `fmsb package` [31] to generate the radar plot.

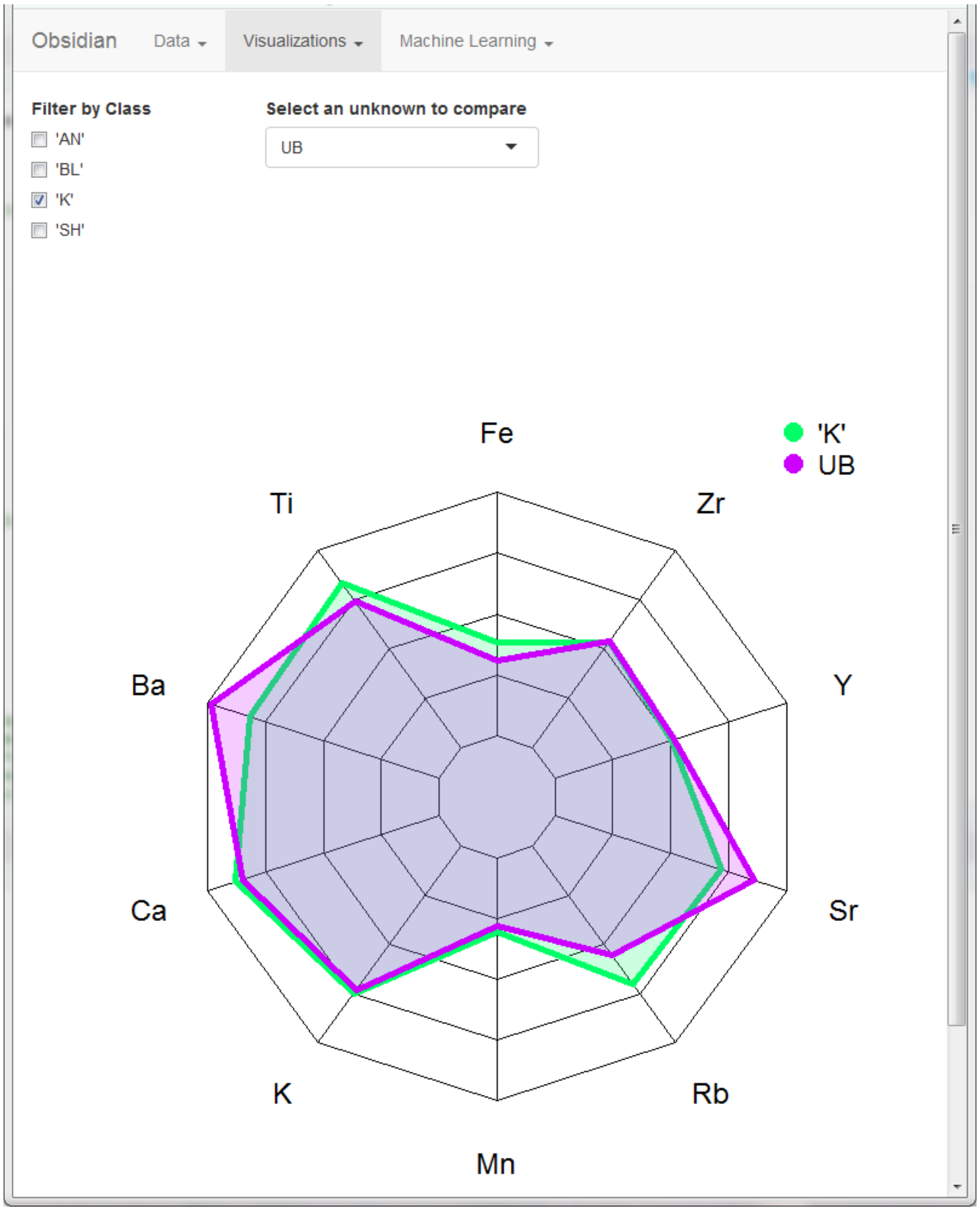


Figure 4.2: Radar plot of Obsidian Unknown 2 vs. true class K

4.5 N-SLOPE Classification Features

This section describes the external layout of N-SLOPE and the internal handling of models, validation, and outputs without going into specific implementation or algorithmic details. The internal workings of N-SLOPE are explained in greater detail in Chapter 5.

4.5.1 Layout

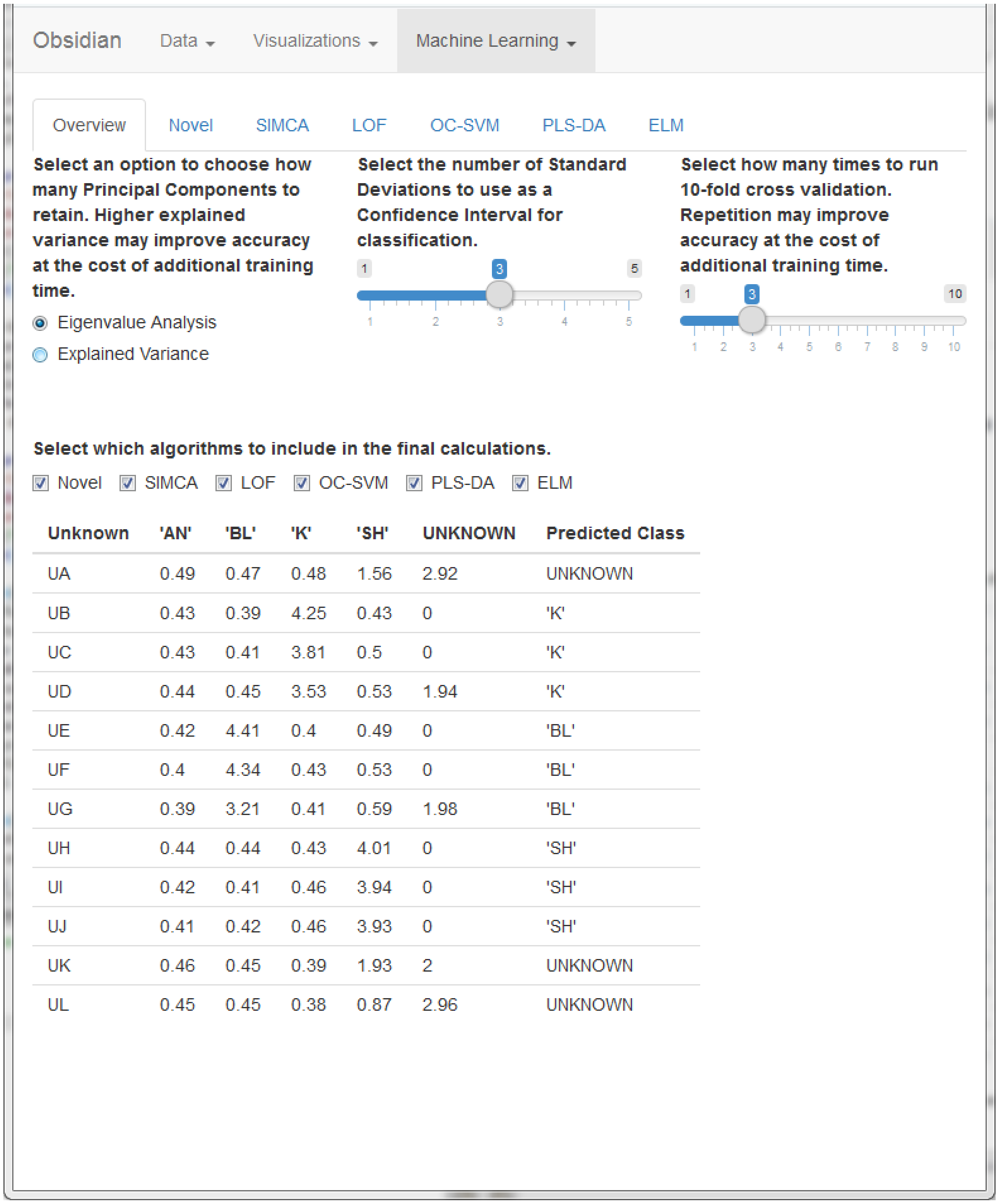
The UI conforms to the rest of the application and is divided into seven separate tabs including one overview tab and one tab for each algorithm in N-SLOPE. The overview tab provides input controls (described in the following subsection) to modify general algorithm behavior and displays the final results of running N-SLOPE. Each algorithm tab runs the respective algorithm and displays meaningful characteristics and classification results. Any action that involves non-trivial computation is tracked with progress bars.

4.5.2 Inputs

N-SLOPE contains several input controls including principal component retention methods, confidence interval classification range, 10-fold cross validation repititions, and selectors for which algorithms to include in the final results. Each input can be summarized as follows.

Principal Components

The number of principal components to retain for the novel, LOF, and PLS-DA algorithms can be determined using *eigenvalue analysis* or *explained variance.* The *eigenvalue analysis* method keeps components with eigenvalues greater than one and



Select an option to choose how many Principal Components to retain. Higher explained variance may improve accuracy at the cost of additional training time.

[✓ Eigenvalue Analysis] Explained Variance

Select the number of Standard Deviations to use as a Confidence Interval for classification.

1 — 3 — 5

Select how many times to run 10-fold cross validation. Repetition may improve accuracy at the cost of additional training time.

1 — 3 — 10

Select which algorithms to include in the final calculations.

[✓ Novel] [✓ SIMCA] [✓ LOF] [✓ OC-SVM] [✓ PLS-DA] [✓ ELM]

Unknown	'AN'	'BL'	'K'	'SH'	UNKNOWN	Predicted Class
UA	0.49	0.47	0.48	1.56	2.92	UNKNOWN
UB	0.43	0.39	4.25	0.43	0	'K'
UC	0.43	0.41	3.81	0.5	0	'K'
UD	0.44	0.45	3.53	0.53	1.94	'K'
UE	0.42	4.41	0.4	0.49	0	'BL'
UF	0.4	4.34	0.43	0.53	0	'BL'
UG	0.39	3.21	0.41	0.59	1.98	'BL'
UH	0.44	0.44	0.43	4.01	0	'SH'
UI	0.42	0.41	0.46	3.94	0	'SH'
UJ	0.41	0.42	0.46	3.93	0	'SH'
UK	0.46	0.45	0.39	1.93	2	UNKNOWN
UL	0.45	0.45	0.38	0.87	2.96	UNKNOWN

Figure 4.3: N-SLOPE overview tab using Obsidian data

discards remaining components that explain only a small amount of variance in the data set. The *explained variance* method keeps as many components as necessary to reach a cumulative explained variance threshold set by the user with higher thresholds usually retaining more components than the *eigenvalue analysis* method. Both methods use the `FactoMineR package` [21] to determine how many components to retain.

Confidence Interval

The novel algorithm and LOF use confidence intervals to produce final classification results. Computed values are compared against statistics from the training set and final classifications are made using a number of standard deviations from training set means. The user is able to select how many standard deviations to include with lower numbers being more sensitive to outlier detection.

Training Repetitions

Every algorithm in N-SLOPE is validated on the training set using 10-fold cross-validation repeated a settable number of times with training accuracy listed on each algorithm's tab. Reported training accuracy reflects only the algorithm's ability to correctly classify the training set with classification error existing as misclassifications within the known set or incorrect outlier detection. In this way, a training accuracy of 100% may indicate some degree of overfitting, as no points in the training set were considered outliers.

Algorithms

The user is able to manually select which N-SLOPE algorithms to include in the final result calculations, but the selected algorithms must be run before they can be included.

All of these controls allow the user to modify the behavior of N-SLOPE for improved performance on specific data sets based on external factors or intrinsic knowledge of the data. For example, the user may choose to retain fewer principal components to get results more quickly or opt to repeat cross validation multiple times in an attempt to improve results at the cost of training time. With some prior knowledge of the data, the user may select more standard deviations to cope with noisy data or if true outliers are unlikely to occur, or the user may choose to exclude linear algorithms like PLS-DA if the data is expected to be nonlinear.

4.5.3 Models

Each N-SLOPE algorithm consists of one or more models which are used to produce output for display in the application. Internally, each model is stored as a `reactiveValue` that updates when a user requests output that depends on the underlying model (e.g. clicking one of the algorithm tabs). Each algorithm is tracked by its own model except for LOF, which requires a KNN model, LOF model, and summary model, and OC-SVM, which requires a parameter model in addition to its base model. The overall design of N-SLOPE and its integration with shiny mimic the classic Model-View-Controller (MVC) design pattern where interactions with the controller (inputs on overview tab) alter the model (underlying algorithm models) that the view (output on algorithm tabs) observes to update appropriately. This modular design makes it easy to add additional views to display more information or model characteristics without major changes to the algorithms or their models.

4.5.4 Validation

Every algorithm in N-SLOPE is validated on the training set using 10-fold cross-validation. SIMCA, PLS-DA, ELM, and the novel algorithm undergo manual cross-validation purely to calculate a training accuracy that represents the percentage of correct classifications performed on the training set. Except for ELM, these algorithms rely solely on hyperparameters and do not need to train additional parameters as part of a learning process. ELM modifies weights between nodes in the network as it trains, but this process is entirely automated with no external tuning necessary. Manual cross-validation is run all at once and independently of the algorithms themselves, meaning that changing N-SLOPE inputs will automatically re-validate these four algorithms, but each algorithm will still have to be run to update the underlying model(s). Manual cross-validation is performed on all four algorithms at once rather than idependently for simplicity and to reduce code reuse.

LOF and OC-SVM perform indepentent cross-validation because both algorithms train parameters as part of the learning process. LOF trains a KNN model to determine an optimal k neighbors to consider when calculated local density and OC-SVM trains *gamma* and *nu*, which represent the bias/variance tradeoff and minimal percentage of support vectors to include in the model, respectively. These two algorithms are validated and run sequentially due to their specialized training procedures, unlike the other four algorithms in N-SLOPE.

4.5.5 Outputs

N-SLOPE outputs are designed similarly to models with all outputs being tracked by `reactiveValues` that update when an algorithm is validated or run. These values separately track both the classification results (including probabilities if applicable) and the training accuracy of each algorithm. Outputs are combined using the sum rule [18] which involves summing each algorithm's prediction output and choosing the maximal combined sum to produce unified classification results for the entire N-SLOPE ensemble.

Chapter 5

N-SLOPE

5.1 Implementation Overview

Each algorithm in N-SLOPE has specific strengths and weaknesses. Combining these algorithms with ensemble learning attempts to minimize their weaknesses and generate more stable, accurate results across data sets. Some properties of each algorithm are displayed in Table 5.1. Implementation details of each algorithm are given below and a running example from the standard Iris data set [3, 9] is used to detail functionality.

Table 5.1: Summary and characteristics of N-SLOPE algorithms

Algorithm	Package	Linearity	Overfitting	High Dimension	Small Data	Runtime
Novel		Linear		✓		Fast
SIMCA	`rrcovHD` [40]	Linear		✓	✓	Fast
LOF	`Rlof` [13]	Linear		✓		Slow
OC-SVM	`e1071` [28]	Nonlinear	✓	✓		Slow
PLS-DA	`caret` [20]	Linear		✓	✓	Medium
ELM	`elmNN` [11]	Nonlinear	✓	✓		Medium

5.2 Ensemble Learning

N-SLOPE consists of several different algorithms whose results must be combined to generate a singular, cohesive output. This is done using the sum rule because it has been shown to outperform other ensemble learning techniques [18]. The basic principle is that each algorithm makes predictions which are summed, and the prediction with the greatest value is selected as the ensemble's decision. This requires that each algorithm in the ensemble produce output that can be summed in a meaningful way.

$$\begin{aligned} \text{N-SLOPE}_{pred} &= \sum\nolimits_{\text{Alg}\in\text{N-SLOPE}} \text{Alg}_{cont} \\ \text{Alg}_{cont} &= \text{Alg}_{acc} \times \text{Alg}_{pred} \end{aligned} \tag{5.1}$$

N-SLOPE implements the sum rule as follows. First, each algorithm in N-SLOPE undergoes 10-fold cross-validation to calculate a training accuracy, which represents the percentage of correct classifications made on the training set. Each algorithm is then run on the testing set and predictions are made. Each algorithm produces slightly different output that must be coerced to a common state before being combined. This process is unique to each algorithm and described in detail in the following sections. Once the predictions have been standardized, they are multiplied by each algorithm's respective training accuracy before being summed. Equation 5.1 shows the general form of these calculations. The final results are tabulated and displayed on the N-SLOPE overview tab and the greatest decision sum of each unknown is selected as the final N-SLOPE prediction.

5.3 Running Example

The standard Iris data set [3, 9] is used as a simple example to illustrate how each algorithm's output is structured and how outputs are combined to produce a final classification result. The Iris data set [3, 9] contains 150 rows and five columns. One column tracks the species (class) of Iris flower and the other columns contain floating point measurements of the flowers. There are three possible classes (Setosa, Versicolor, and Virginica) and 50 records per class. Setosa is linearly separable from the other two classes, so a highly overlapping Virginica sample (Sample 139) has been chosen as an example. This sample will be treated as testing data with the remaining samples treated as training data for the purpose of this example.

5.4 Existing Algorithms

5.4.1 SIMCA

SIMCA operates by independently performing PCA on each class in the training set and then projecting the knowns and unknowns to the same space and comparing against a threshold to determine class membership. SIMCA in N-SLOPE is provided by the `rrcovHD package` [40] which implements a robust SIMCA approach, or RSIMCA, and returns Orthogonal Distance Scores (ODSC) and Score Distance Scores (SDSC) for each unknown. ODSC represent the Euclidean distance between a point and the center of each PCA subspace, whereas SDSC represent the Mahalanobis distance, which is the distance along each principal component between a point and the PCA subspace [5]. These scores are used to compute the classification rules `R1` and `R2` proposed by Branden and Hubert [5]. The minimum of the two values is kept for each class with membership being associated with any value less than one. In this way, RSIMCA can assign an unknown to zero, one, or more classes with lower values indicating a stronger association. An unknown with computed `R1` and `R2` greater than one is not assigned to any known class and thus labeled an outlier.

Table 5.2: SIMCA results on running example

Setosa	Versicolor	Virginica	Predicted Class
2.77	0.33	0.13	Virginica

Using the running example, RSIMCA produces the output shown in Table 5.2.

This indicates the unknown belongs to both the Versicolor and Virginica classes, but not the Setosa class. Since the computed classification value is lower for Virginica, the unknown is assigned to that class. RSIMCA's contribution to the ensemble is computed as

$$\text{SIMCA}_{cont} = \text{SIMCA}_{acc} \times (1 - \text{SIMCA}_{pred}) \tag{5.2}$$

where SIMCA_{acc} is the training accuracy and SIMCA_{pred} is the computed classification values less than one. Continuing with the running example where RSIMCA has a training accuracy of 94.89%, the contribution is calculated as follows.

$$\begin{aligned}\text{SIMCA}_{cont} &= 0.9489 \times \left(1 - \begin{bmatrix}1 & 0.33 & 0.13 & 1\end{bmatrix}\right)\\ &= 0.9489 \times \begin{bmatrix}0 & 0.67 & 0.87 & 0\end{bmatrix}\\ &= \begin{bmatrix}0 & 0.6358 & 0.8255 & 0\end{bmatrix}\end{aligned} \tag{5.3}$$

The ensemble contains the results from SIMCA shown in Table 5.3 that correctly predict the unknown belongs to Virginica, although there is some confusion with possible Versicolor class membership.

Table 5.3: Ensemble contribution of SIMCA on running example

Setosa	Versicolor	Virginica	UNKNOWN
0	0.6358	0.8255	0

5.4.2 LOF

LOF operates by comparing the local density of each unknown to the local density of neighboring classes with values of approximately one indicating membership. The LOF implementation in N-SLOPE is provided by the `Rlof package` [13] and requires a number of neigbors to examine when computing densities. This number is determined by training a KNN classification model courtesy of the `caret package` [20] and using the same optimal k (determined to be $k = 11$ for the running example). Since LOF produces a score rather than a fixed classification, statistics including the mean and standard deviation are calculated for each sample in the training set by class. Table 5.4 shows these values for the running example.

Table 5.4: LOF statistical summary on running example

Class	Mean	Standard Deviation
Setosa	1.15	0.34
Versicolor	1.08	0.16
Virginica	1.23	0.5

Each unknown's LOF is compared against the LOF statistics of its nearest neighboring class and class membership is determined by whether or not the LOF falls within a user-settable threshold of standard deviations from that class's mean. Output from LOF on the running example is shown in Table 5.5 and the unknown is incorrectly classified as Versicolor.

Table 5.5: LOF results on running example

LOF	Nearest Neighbors	Predicted Class
0.98	Versicolor	Versicolor

Since LOF does not produce any associated class membership probabilities, the final contribution to the ensemble is simply the LOF training accuracy added to the predicted class for each unknown. LOF contribution for the running example, where the training accuracy is 92.73%, is calculated in Equation 5.4 and shown in Table 5.6.

$$\begin{aligned} \mathrm{LOF}_{cont} &= \mathrm{LOF}_{acc} \times \mathrm{LOF}_{pred} \\ &= 0.9273 \times \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0.9273 & 0 & 0 \end{bmatrix} \end{aligned} \tag{5.4}$$

Table 5.6: Ensemble contribution of LOF on running example

Setosa	Versicolor	Virginica	UNKNOWN
0	0.9273	0	0

5.4.3 OC-SVM

OC-SVM operates by creating a hypersphere decision boundary around the training data and assigning class membership to unknowns based on whether or not they fall within this boundary. OC-SVM in N-SLOPE is provided by the `e1071 package` [28] and has two tunable parameters: *gamma* and *nu*. *Gamma* controls the bias/variance tradeoff with small *gamma* leading to high variance and overfitting and large *gamma* leading to high bias and underfitting. *Nu* controls the minimal percentage of support vectors used when building the model, as well as the maximal percentage of training data allowed to be misclassified. *Gamma* and *nu* are tuned during training and vary between data sets. This OC-SVM implementation supports linear, polynomial, radial basis, and sigmoid kernel functions. Since OC-SVM is included in N-SLOPE to add support for complex, nonlinear relationships, radial basis or sigmoid are both attractive options. Since N-SLOPE includes two nonlinear solutions, OC-SVM uses the radial basis kernel function and ELM, the other nonlinear solution, uses sigmoid.

$$\begin{aligned}\text{OC-SVM}_{cont} &= \text{OC-SVM}_{acc} \times \text{OC-SVM}_{pred} \\ &= 0.9566 \times \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 & 0.9566 & 0 \end{bmatrix}\end{aligned} \tag{5.5}$$

Table 5.7: Ensemble contribution of OC-SVM on running example

Setosa	Versicolor	Virginica	UNKNOWN
0	0	0.9566	0

OC-SVM only produces a true/false value for each unknown indicating whether or not the unknown belongs to any of the known classes. A traditional SVM is created using the same *gamma* and *nu* values to complete classification for unknowns that belong to one of the known classes. The final output of OC-SVM is simply a classification: one of the known classes if it is predicted to belong, otherwise it is marked as an outlier. OC-SVM output is factored into the ensemble by adding the OC-SVM training accuracy to the associated class for each unknown. OC-SVM has a training accuracy of 95.66% on the running example and correctly predicted that the unknown belongs to Virginica. The final contribution to the ensemble is calculated in Equation 5.5 and shown in Table 5.7.

5.4.4 PLS-DA

PLS-DA operates by performing PCA on the entire training set to reduce dimensionality and then running linear regression and creating class decision boundaries based on confidence intervals. PLS-DA in N-SLOPE is provided by the `caret package` [20] and supports two different classification decision algorithms: Softmax and Bayes. Since PLS-DA does not act as a true one-class classifier and always attempts to place unknowns into one of the known classes, two models are trained with the same number of principal components using different decision algorithms. The final output for each unknown is a probability of belonging to each of the possible known classes and a classification as either one of the known classes (if both models agree) or the unknown is marked as an outlier if they do not agree. In this way, this implementation of PLS-DA marks unknowns as outliers more aggressively than other algorithms in N-SLOPE because true outliers and confusion between known classes are treated the same. This flaw is factored into the PLS-DA contribution to the ensemble by including class probabilities whether or not the unknown is marked as an outlier. Table 5.8 shows output from the running example that highlights this behaviour.

Table 5.8: PLS-DA results on running example

Setosa	Versicolor	Virginica	Predicted Class
0.26	0.34	0.4	UNKNOWN

The final contribution to the ensemble from PLS-DA is calculated by mulitplying the training accuracy with each unknown's assocaited class probability. If PLS-DA predicts the unknown is an outlier, then the full training accuracy is applied, but the remaining probabilities are still included. PLS-DA has a training accuracy of 76.64% on the running example, which creates the final contribution calculated in Equation 5.6 and shown in Table 5.9.

$$\begin{aligned} \text{PLS-DA}_{cont} &= \text{PLS-DA}_{acc} \times \text{PLS-DA}_{pred} \\ &= 0.7664 \times \begin{bmatrix} 0.26 & 0.34 & 0.4 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 0.1993 & 0.2606 & 0.3066 & 0.7664 \end{bmatrix} \end{aligned} \tag{5.6}$$

PLS-DA incorrectly predicts the unknown is an outlier, likely due to confusion between Versicolor and Virginica since the two classes are non-linearly separable.

However, PLS-DA does contribute its second strongest prediction to the correct class: Virginica.

Table 5.9: Ensemble contribution of PLS-DA on running example

Setosa	Versicolor	Virginica	UNKNOWN
0.1993	0.2606	0.3066	0.7664

5.4.5 ELM

ELM operates by building a NN with one input node for each dimension, a single hidden layer, and a single output node that predicts whether or not an unknown belongs to any of the known classes. ELM in N-SLOPE is provided by the `elmNN package` [11] and starts with randomly initialized weights between nodes that are updated during a single training phase. This implementation of ELM allows for several different kernel functions and setting the number of nodes in the hidden layer. The sigmoid function is chosen to provide a nonlinear approach and to complement the radial basis kernel function used by OC-SVM. Selecting an optimal number of hidden layer nodes for generalized performance is a difficult problem, but there is some suggestion that single-layer, feed-forward networks like ELM are more reliant on updating connection weights than the number of nodes for general performance [14]. Through trial and error, the hidden layer has been chosen to contain 5000 nodes.

$$\begin{aligned}\mathrm{ELM}_{cont} &= \mathrm{ELM}_{acc} \times \mathrm{ELM}_{pred} \\ &= 1 \times \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 \end{bmatrix} \\ &= \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 \end{bmatrix}\end{aligned} \tag{5.7}$$

Since ELM output only indicates whether or not an unknown is an outlier, its contribution to the ensemble is fairly straightforward. If the unknown is determined

Table 5.10: Ensemble contribution of ELM on running example

Setosa	Versicolor	Virginica	UNKNOWN
0.33	0.33	0.33	0

to belong to one of the known classes, the ELM training accuracy is evenly divided among all known classes. If the unknown is determined to be an outlier, the entire training accuracy supports this classification. ELM has a training accuracy of 100% on the running example, predicting that every known belongs to one of the known classes. ELM correctly determined that the unknown belongs to one of the known classes and its contribution for the running example is calculated in Equation 5.7 and shown in Table 5.10. ELM operates as more of a "black box" than the other algorithms in N-SLOPE and there is some concern around overfitting, but it does identify certain outliers when working with larger, more complex data sets.

5.5 Novel Algorithm

5.5.1 Overview

The novel algorithm included in N-SLOPE is inspired by PCA, LOF, Nearest Centroid, and SIMCA. PCA is performed for dimensionality reduction and summary statistics are computed for each class, similar to the decision method used by the LOF implementation in N-SLOPE. Ideas from Nearest Centroid and SIMCA are incorporated through scaling the final decision scores into a soft classifier based on distances between unknowns and class centroids. A complete description of the novel algorithm's operation and contribution to the ensemble is provided in the following subsections.

5.5.2 Design

Algorithm 1: N-SLOPE Novel Algorithm

```
Input  : trainingData, testingData
Output: Calculated values for each testing sample and known class that
        provide one-class classification with class assignment on values (0,1)
1  pcaData ← PCA(trainingData, testingData)
2  train ← pcaData[trainingData]
3  test ← pcaData[testingData]
4  foreach class in train do
5      classCenter[class] ← Mean(train[class])
6  end
7  foreach sample in train do
8      distToCenter[sample] ← Euclidean_Distance(sample,
         classCenter[sample.class])
9  end
10 foreach class in train do
11     avgDist[class] ← Mean(distToCenter[sample.class == class])
12     stdDevDist[class] ← Std_Dev(distToCenter[sample.class == class])
13     threshold[class] ← avgDist[class] ± stdDevDist[class]
14     foreach unknown in test do
15         unknownDist[unknown, class] ← Euclidean_Distance(unknown,
             classCenter[class])
16         pred[unknown, class] ← $\frac{(unknownDist[unknown, class] - threshold[class].lower)}{(threshold[class].upper - threshold[class].lower)}$
17     end
18 end
19 return pred
```

5.5.3 Limitations

This novel algorithm is somewhat naïve and assumes data is roughly spherical, causing it to perform poorly on strongly shaped data. Shaped data is less likely to exist in the high-dimensional data common to this domain, but even moderate shaping will negatively impact this algorithm's classification accuracy. Another limitation of this algorithm is that it is distance-based and will thus struggle with sparse data. Furthermore, since the algorithm uses summary statistics to make classification decisions, small data sets may also pose a problem. These last two considerations are common problems for most machine learning techniques and are one of the interesting challenges posed by the chemometric domain.

5.5.4 Strengths

This novel algorithm runs significantly faster than the other N-SLOPE algorithms, particularly on larger, high-dimensional data sets. It is also able to clearly detect outliers in the case where data is roughly doughnut-shaped with knowns in a spherical shape surrounding an empty or hollow core. This is a specialized and uncommon case, but not unheard of for high-dimensional, sparse data sets.

5.5.5 Implementation

The novel algorithm implementation is described here with parenthetical references to line numbers shown above in Algorithm 1. The novel algorithm begins by combining the training and testing data into a single data frame, performing PCA on the data courtesy of the `FactoMineR package` [21], and retaining the number of components selected through *eigenvalue analysis* or *explained variance* as specified by the user (Line 1). The data is then split back into the original training (Line 2) and testing

(Line 3) sets.

The center point of each known class is calculated (Line 5) by averaging every column in each class. Next, the Euclidian distance between each known and its respective class center is calculated (Line 8). These distances are used to calculate an average distance-to-center (Line 11) as well as a standard deviation of distance-to-center measurements (Line 12) for each class, similar to how LOF computes class summary statistics. These values are used to generate loose class boundaries in the form of confidence intervals by calculating upper and lower acceptance thresholds (Line 13) as

$$\begin{aligned} \text{Threshold}_{lower} &= \text{Dist}_{avg} - (CI \times \text{Dist}_{sd}) \\ \text{Threshold}_{upper} &= \text{Dist}_{avg} + (CI \times \text{Dist}_{sd}) \end{aligned} \tag{5.8}$$

for each class where CI is the user-settable number of standard deviations to include for N-SLOPE decision boundaries. The algorithm then computes the distance between each unknown and the center of each known class (Line 15) and compares this distance with the computed acceptance thresholds of each class. This comparison is normalized into the range $(0, 1)$ for each unknown and class by calculating the following (Line 16).

$$\frac{(\text{Dist}_{toCenter} - \text{Threshold}_{lower})}{(\text{Threshold}_{upper} - \text{Threshold}_{lower})} \tag{5.9}$$

Class membership is assigned to any computed values between zero and one. Negative computed values indicate the unknown is uncharacteristically close to the mathematical center of the class, while values greater than one indicate the unknown lies uncharacteristically far from the mathematical center of the class. Since this algorithm acts as a soft classifier like SIMCA, it is possible to assign a single unknown to multiple classes. Unlike SIMCA, the computed value does not necessarily indicate a strength of belonging, meaning that any value between zero and one is equally indicative of class membership. The novel algorithm's contribution to the ensemble reflects this fact. If a single decision is required, the algorithm will draw on ideas from Nearest Centroid and select the minimal computed value between zero and one as the final class prediction because this indicates the unknown is statistically closest to that class centroid.

5.5.6 Contribution to Ensemble

Once the novel algorithm has made class predictions for each unknown, contributing its results to the ensemble is fairly straightforward. The novel algorithm's training accuracy is added to each class where membership is predicted, or to the outlier class if the unknown is predicted to not belong to any of the known classes.

Table 5.11: Novel algorithm results on running example

Setosa	Versicolor	Virginica
1.03	0.26	0.27

Table 5.11 shows the novel algorithm's output on the running example, where the training accuracy is 84.82% and the unknown is classified as both Versicolor and Virginica. The final contribution to the ensemble is calculated in Equation 5.10 and shown in Table 5.12.

$$\begin{aligned} \text{Novel}_{cont} &= \text{Novel}_{acc} \times \text{Novel}_{pred} \\ &= 0.8482 \times \begin{bmatrix} 0 & 1 & 1 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0.8482 & 0.8482 & 0 \end{bmatrix} \end{aligned} \tag{5.10}$$

Table 5.12: Ensemble contribution of novel algorithm on running example

Setosa	Versicolor	Virginica	UNKNOWN
0	0.8482	0.8482	0

5.6 Running Example Summary

The output of each algorithm in N-SLOPE is combined using the sum rule [18] to produce a single classification for each unknown. A running example has been used to highlight the output and ensemble contributions of each unknown. Tables 5.3, 5.6, 5.7, 5.9, 5.10, and 5.12 display the contributions of each algorithm on the running example. These values are summed as shown in Equation 5.11 to produce the final classification results shown in Table 5.13 that correctly assign the unknown to Virginica.

$$\begin{aligned}
\text{N-SLOPE}_{pred} &= \sum\nolimits_{\text{Alg}\in\text{N-SLOPE}} \text{Alg}_{cont} \\
&= \begin{bmatrix} 0 & 0.8482 & 0.8482 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0.6358 & 0.8255 & 0 \end{bmatrix} + \\
&\quad \begin{bmatrix} 0 & 0.9273 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0.9566 & 0 \end{bmatrix} + \\
&\quad \begin{bmatrix} 0.1993 & 0.2606 & 0.3066 & 0.7664 \end{bmatrix} + \begin{bmatrix} 0.33 & 0.33 & 0.33 & 0 \end{bmatrix} \\
&= \begin{bmatrix} 0.5293 & 3.0019 & 3.2669 & 0.7664 \end{bmatrix}
\end{aligned} \tag{5.11}$$

N-SLOPE correctly classified the unknown despite SIMCA and the novel algorithm assigning it to both Versicolor and Virginica, and LOF and PLS-DA misclassifying it entirely as Versicolor and an outlier, respectively. This running example serves to illustrate how N-SLOPE operates and highlight the potential power of ensemble learning.

Table 5.13: Final N-SLOPE output on running example

Setosa	Versicolor	Virginica	UNKNOWN	Predicted Class
0.5293	3.0019	3.2669	0.7664	Virginica

Chapter 6

VALIDATION

6.1 Data Sets

6.1.1 Iris

The standard Iris data set [3, 9] contains 150 rows and four columns of floating point measurements of Iris flowers. There are three possible classes (Setosa, Versicolor, and Virginica) representing three different species of Iris and 50 records per class. Setosa is linearly separable from the other two classes, which are overlapping.

For validation purposes, this data set was divided into a training and testing set. The testing set contains 13 samples: four Setosa, four Versicolor, and five Virginica. The training set contains the remaining 137 samples. This simple, standard data set is included as a basic benchmark for N-SLOPE and to conclude the running example from the previous section that dealt with this same data set.

6.1.2 Obsidian

This data set is a subset of data first collected and analyzed as archaeological artifacts [19, 36] then compiled and labeled by Eigenvector Research, Inc [42]. The data describes 10 chemical concentrations of various elemental impurities present in Obsidian, which is a dark-colored, volcanic glass. The training set contains 63 samples belonging to one of four classes (AN, BL, K, SH) that represent the location where the Obsidian fragment was collected. Each class is linearly separable from the others. The testing set contains 12 unknowns whose true identities are unfortunately not known. However, analysis has been performed on these samples and predicted classes recorded for each. These predicted classes will be treated as true classes for the purpose of validating N-SLOPE.

This data set is a good analog for nuclear forensics because Obsidian naturally absorbs impurities from the environment in which it forms and these impurities can be used to trace Obsidian to its formation site. Furthermore, this small, high-dimensional data set (63 rows by 10 columns with one class containing just nine total samples) presents an interesting machine learning challenge appropriate for this difficult data domain.

6.1.3 Galaxy Serpent 3

Galaxy Serpent 3 (GS3) is a tabletop exercise run by the Nuclear Forensics International Technical Working Group (ITWG) to support the development of a National Nuclear Forensics Library (NNFL), which aims to facilitate nuclear forensic investigations. GS3 uses simulated data that attempts to mimic the properties or characteristics of real uranium ore concentrate. Simulated data is used due to the sensitive and proprietary nature of such information. This data set comes courtesy of Naomi Marks from Lawrence Livermore National Laboratory and accurately simulates chemometric data used in nuclear forensic investigations.

The GS3 data set contains both a training set of labeled knowns and testing set of unknowns. The training set has 821 rows, 45 columns of numerical data, and 4 described classes: IAB, MORB, OIB, and ZCRFB. IAB is linearly separable from the other classes, which are overlapping. The testing set contains 60 unknowns whose true identities are excluded from the original exercise, but have been revealed here for the purpose of validation. The testing set contains a number of samples from each known class, as well as a number of samples from at least four distinct groups that do not belong to any of the known classes.

The training set, by design, is missing about one third of all possible measurements. This further increases the difficulty of the problem, but accurately reflects the reality where certain samples undergo certain tests and other samples are subject to different scientific measurements. Missing data in the training set is substituted with imputation, while columns containing missing data are dropped from the testing set, as imputation would prove inaccurate. Columns are then synchronized between the training and testing set to ensure each set includes measurements of the same metrics.

6.2 Runtime

Each algorithm was run and timed on each data set using N-SLOPE default settings of *eigenvalue analysis* for PCA component retention, two standard deviations for classification thresholds, and one run of 10-fold cross validation. Timing results vary greatly between algorithms and are displayed in Table 6.1.

Table 6.1: N-SLOPE algorithm runtimes on each data set

	Novel	SIMCA	LOF	OC-SVM	PLS-DA	ELM
Iris	55.8ms	31.2ms	4.6s	6.5s	145.4ms	302.8ms
Obsidian	46.8ms	152.8ms	6.5s	4.7s	222.8ms	87.0ms
GS3	383.8ms	874.3ms	13.9s	178.1s	4.5s	9.6s

The novel algorithm and SIMCA run fastest overall, with the novel algorithm beating SIMCA on higher dimensional data sets, while OC-SVM runs slowest overall. The runtimes of the novel algorithm, OC-SVM, and ELM are more dependent on the number of samples than dimensionality. Conversely, the runtimes of SIMCA, LOF, and PLS-DA are more effected by dimensionality than the number of samples. As both the number of samples and dimensionality increase, the novel algorithm performs faster than the other N-SLOPE algorithms, taking just over a third of a second to run on the GS3 data set, and OC-SVM performs significantly slower than the other algorithms, taking almost three minutes to run on the GS3 data set.

6.3 Results

The classification results of running N-SLOPE on each data set described above are provided here in the following format. Each algorithm's predictions are listed in separate columns followed by the final N-SLOPE ensemble predictions with the true class of each unknown listed in the last column. Each of the six algorithms has an associated training accuracy displayed below it in parenthesis. A prediction from any algorithm in N-SLOPE will either assign an unknown to one or more of the known classes or classify the unknown as an outlier and mark it as UNKNOWN. The one exception is ELM, which marks unknowns as either known (assigned equally to all classes) or UNKNOWN (novel unknown that does not belong to any known classes).

The overall classification accuracies for each algorithm in the ensemble are displayed separately. These tables include Classification Accuracy (CA - percentage of unknowns correctly classified), False Positive Rate (FPR - percentage of novel unknowns incorrectly classified as belonging to one of the known classes), and False Negative Rate (FNR - percentage of inlying unknowns incorrectly classified as outliers). FPR is omitted when the testing set lacks any novel unknowns.

Ideal classification performance should yield a high CA, low FPR, and low FNR. FPR and FNR are typically inversely coorelated such that one must be prioritized over the other. Since this application is designed to aid nuclear forensic investigations, minimizing FPR is the priority as falsely assigning a novel unknown to a known class is more costly than failing to identify an inlying unknown.

6.3.1 Iris

Analysis was performed using the default N-SLOPE settings: the number of principal components to retain was determined using *eigenvalue analysis*, two standard deviations were used as a confidence interval for classification, and 10-fold cross-validation was performed a single time.

Table 6.2: N-SLOPE classification results on Iris test data

	Novel	SIMCA	LOF	OC-SVM	PLS-DA	ELM	N-SLOPE
CA	84.62%	100%	92.31%	100%	69.23%	100%	100%
FPR							
FNR	0%	0%	0%	0%	30.77%	0%	0%

Tables 6.2 and 6.3 show the accuracies and predictions, respectively, of N-SLOPE on this data set. FPR is omitted from Table 6.2 because the testing set does not have any novel unknowns. The novel algorithm and PLS-DA in particular struggled a bit on this data set because the Versicolor and Virginica classes are overlapping and many of the unknowns in the training set fall in this overlap. Despite slight confusion on Unknowns 8 and 9, N-SLOPE was able to correctly classify every unknown in the training set and perform very well as a whole.

Table 6.3: N-SLOPE predictions on Iris test data with possible classes SE (Setosa), VE (Versicolor), VI (Virginica), KN (Known), and UN (UNKNOWN)

	Novel (81.02%)	SIMCA (96.35%)	LOF (93.61%)	OC-SVM (96.37%)	PLS-DA (76.64%)	ELM (100%)	N-SLOPE	True Class
1	SE	SE	SE	SE	SE	KN	**SE**	**SE**
2	SE	SE	SE	SE	SE	KN	**SE**	**SE**
3	SE	SE	SE	SE	SE	KN	**SE**	**SE**
4	SE	SE	SE	SE	SE	KN	**SE**	**SE**
5	VE	VE	VE	VE	UN	KN	**VE**	**VE**
6	VE	VE	VE	VE	VE	KN	**VE**	**VE**
7	VE	VE	VE	VE	VE	KN	**VE**	**VE**
8	VI	VE	VE	VE	UN	KN	**VE**	**VE**
9	VE	VI	VE	VI	UN	KN	**VI**	**VI**
10	VI	VI	VI	VI	VI	KN	**VI**	**VI**
11	VI	VI	VI	VI	VI	KN	**VI**	**VI**
12	VI	VI	VI	VI	UN	KN	**VI**	**VI**
13	VI	VI	VI	VI	VI	KN	**VI**	**VI**

6.3.2 Obsidian

N-SLOPE settings were modified for this extremely small data set. The number of principal components to retain was selected using *eigenvalue analysis* as normal, but three standard deviations were used to construct classification confidence intervals (rather than the default two) and 10-fold cross-validation was performed a total of three times (rather than once by default). These changes allow for a greater variability within known classes and allow algorithms additional training time.

The accuracies in Table 6.4 and predictions in Table 6.6 highlight the unique chal-

Table 6.4: N-SLOPE classification results on Obsidian test data

	Novel	SIMCA	LOF	OC-SVM	PLS-DA	ELM	N-SLOPE
CA	91.67%	91.67%	75.0%	75.0%	75.0%	91.67%	83.33%
FPR	0%	0%	0%	8.33%	8.33%	8.33%	0%
FNR	8.33%	8.33%	25.0%	16.67%	16.67%	0%	16.67%

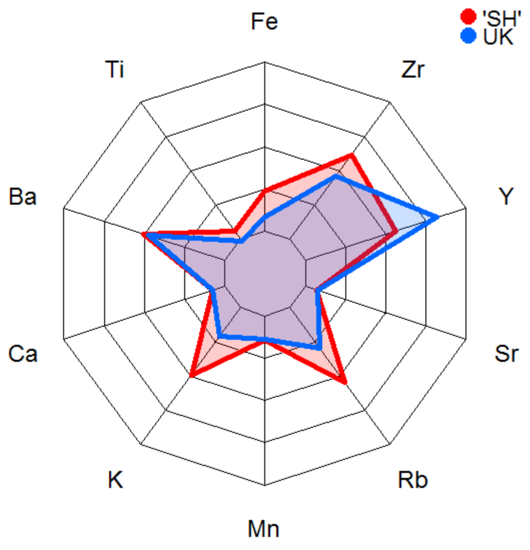


Figure 6.1: Radar plot of Obsidian Unknown 11 vs. true class SH

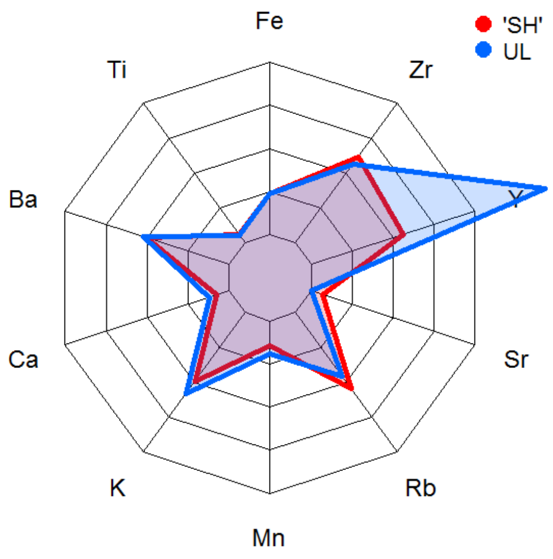


Figure 6.2: Radar plot of Obsidian Unknown 12 vs. true class SH

lenges this data set presents. In particular, algorithms struggled with Unknowns 1, 4, 7, 11, and 12. The N-SLOPE ensemble was able to correctly classify Unknowns 1, 4, and 7 by sizeable margins as shown in Table 6.5, but failed to identify Unknowns 11 and 12 by a narrow and wide margin, respectively. Further examination of these two unknowns using the radar plot tool included in the web application reveals the source of misclassification. Figures 6.1 and 6.2 show the relation between the true class SH and Unknowns 11 and 12, respectively. Unknown 11 does resemble SH; not quite as closely as other unknowns like Unknown 10 shown in Figure 6.3, which explains the narrow margin of misclassification, but close enough for additional manual investigation to conclude Unknown 11 likely comes from SH. Unknown 12 resembles SH very closely except for its concentration of Y, which is about 17.5% higher than

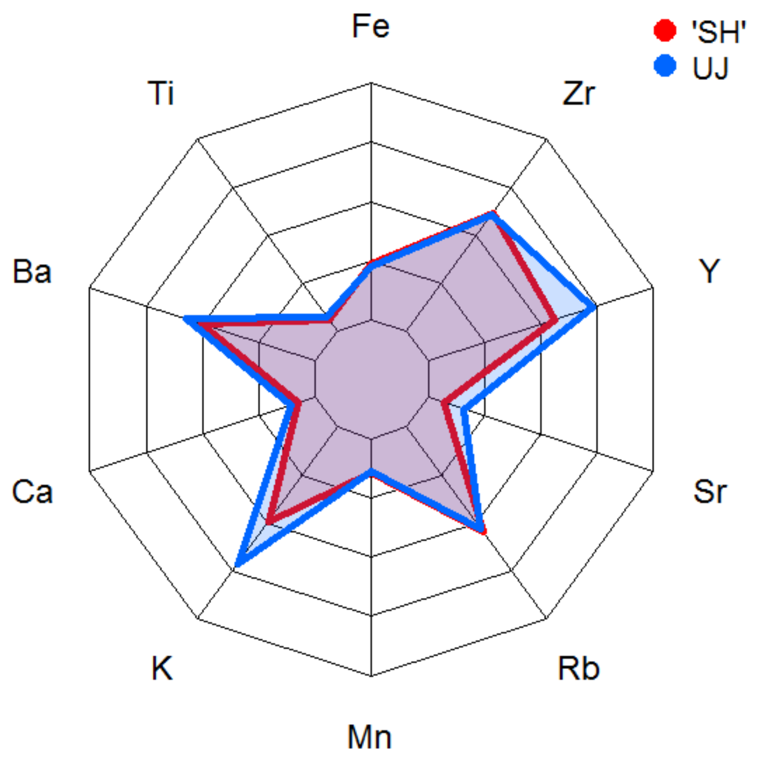


Figure 6.3: Radar plot of Obsidian Unknown 10 vs. true class SH

any other sample in the data set and 23.5% higher than any other sample from SH in the data set. This is enough for N-SLOPE to flag the unknown as an outlier, which encourages further manual examination.

Table 6.5: Raw N-SLOPE output on difficult Unknowns 1, 4, 7, 11, and 12 from the Obsidian data set

	AN	BL	K	SH	UNKNOWN	Predicted Class
1	0.49	0.47	0.48	1.56	2.93	UNKNOWN
4	0.44	0.45	3.53	0.53	1.95	K
7	0.39	3.23	0.41	0.59	1.98	BL
11	0.46	0.45	0.39	**1.94**	**2**	UNKNOWN
12	0.45	0.45	0.38	0.87	2.96	UNKNOWN

The novel algorithm and SIMCA performed best overall, misclassifying just one distinct unknown in the testing set each. ELM also misclassified only a single unknown, but it failed to detect the novel unknown. The remaining algorithms each misclassified three unknowns, with OC-SVM and PLS-DA failing to identify the novel unknown and producing a false positive identificiation. N-SLOPE performed fairly well on this difficult data set as a whole, misclassifying just two unknowns, both incorrectly as outliers, while correctly identifying the one supposed novel unknown in the set. This behavior is desirable as false positives are more costly than false negatives.

Table 6.6: N-SLOPE predictions on Obsidian test data

	Novel (96.83%)	SIMCA (94.44%)	LOF (100%)	OC-SVM (100%)	PLS-DA (97.62%)	ELM (100%)	N-SLOPE	True Class
1	UN	UN	UN	SH	SH	KN	**UN**	**UN**
2	K	K	K	K	K	KN	**K**	**K**
3	K	K	K	K	K	KN	**K**	**K**
4	K	UN	K	K	UN	KN	**K**	**K**
5	BL	BL	BL	BL	BL	KN	**BL**	**BL**
6	BL	BL	BL	BL	BL	KN	**BL**	**BL**
7	BL	BL	UN	BL	UN	KN	**BL**	**BL**
8	SH	SH	SH	SH	SH	KN	**SH**	**SH**
9	SH	SH	SH	SH	SH	KN	**SH**	**SH**
10	SH	SH	SH	SH	SH	KN	**SH**	**SH**
11	SH	SH	UN	UN	SH	KN	**UN**	**SH**
12	UN	SH	UN	UN	SH	KN	**UN**	**SH**

6.3.3 Galaxy Serpent 3

Analysis was performed using the default N-SLOPE settings: the number of principal components to retain was determined using *eigenvalue analysis*, two standard deviations were used as a confidence interval for classification, and 10-fold cross-validation was performed a single time. Overall accuracy is displayed in Table 6.7, which includes the additional metric True Negative Rate (TNR) recording the percentage of novel unknowns N-SLOPE correctly identified as novel unknowns. This metric is included here to highlight the fact that N-SLOPE correctly identified every novel unknown in the testing set, which is the primary motivator behind using one-class classification techniques over traditional classification.

Table 6.7: N-SLOPE classification results on GS3 test data

	Novel	SIMCA	LOF	OC-SVM	PLS-DA	ELM	N-SLOPE
CA	80.0%	85.0%	78.33%	73.33%	58.33%	88.33%	83.33%
TNR	100%	100%	68.75%	100%	37.5%	62.5%	100%
FPR	0%	0%	8.33%	0%	16.67%	10.0%	0%
FNR	10.0%	6.67%	1.67%	23.33%	25.0%	3.33%	6.67%

N-SLOPE predictions on the GS3 test data are shown in Tables 6.8, 6.9, and 6.10 and highlight the difficulty of this data set. N-SLOPE performed perfectly when it came to classifying MORB, ZCRFB, and recognizing novel unknowns, but struggled with classifying IAB and OIB. Internally, certain algorithms appear better suited for identifying specific classes in this data set. SIMCA was the only algorithm that correctly identified every unknown belonging to IAB, but SIMCA struggled with MORB (unlike the other algorithms) and OIB (like the other algorithms). LOF was the only algorithm that correctly identified every unknown belonging to OIB, but LOF struggled with IAB (like the other algorithms) and detecting novel unknowns (unlike the other algorithms).

Each algorithm in N-SLOPE has strengths and weaknesses that are somewhat mitigated by combining them in an ensemble. SIMCA performed best overall, misclassifying just nine samples from the testing set, correctly identifying every novel unknown in the testing set, and producing no false positive classifications. The novel algorithm came in second misclassifying 12 samples, but correctly identifying every novel unknown and producing no false positives. PLS-DA performed worst overall, misclassifying 25 samples and failing to identify almost two-thirds of the novel unknowns. This can be somewhat explained by the fact that PLS-DA is not a true one-class classifier and has been modified here to identify ambiguous classifications as outliers, not necessarily to detect outliers in and of themselves.

N-SLOPE misclassified 10 samples in the testing set, while successfully identifying every novel unknown in the testing set and producing no false positive classifications. SIMCA was able to correctly classify one more sample than the ensemble as a whole, but N-SLOPE outperformed every other algorithm in the ensemble.

Table 6.8: N-SLOPE predictions on GS3 test data 1-25

	Novel (87.39%)	SIMCA (90.26%)	LOF (96.80%)	OC-SVM (98.30%)	PLS-DA (92.98%)	ELM (96.43%)	N-SLOPE	True Class
1	IAB	IAB	IAB	IAB	IAB	KN	**IAB**	**IAB**
2	UN	IAB	UN	UN	IAB	KN	**UN**	**IAB**
3	UN	UN	UN	UN	UN	KN	**UN**	**UN**
4	IAB	IAB	IAB	IAB	IAB	KN	**IAB**	**IAB**
5	IAB	IAB	IAB	IAB	IAB	KN	**IAB**	**IAB**
6	IAB	IAB	IAB	IAB	IAB	KN	**IAB**	**IAB**
7	IAB	IAB	IAB	IAB	IAB	KN	**IAB**	**IAB**
8	IAB	IAB	IAB	IAB	IAB	KN	**IAB**	**IAB**
9	IAB	IAB	IAB	IAB	IAB	KN	**IAB**	**IAB**
10	IAB	IAB	IAB	IAB	IAB	KN	**IAB**	**IAB**
11	IAB	IAB	IAB	IAB	IAB	KN	**IAB**	**IAB**
12	UN	IAB	IAB	UN	IAB	KN	**IAB**	**IAB**
13	UN	IAB	IAB	UN	IAB	KN	**IAB**	**IAB**
14	UN	IAB	IAB	UN	IAB	KN	**IAB**	**IAB**
15	MORB	IAB	MORB	UN	UN	KN	**MORB**	**IAB**
16	MORB	IAB	MORB	UN	UN	KN	**MORB**	**IAB**
17	MORB	IAB	MORB	UN	UN	KN	**MORB**	**IAB**
18	ZCRFB	IAB	MORB	UN	UN	KN	**MORB**	**IAB**
19	IAB	IAB	MORB	UN	UN	KN	**IAB**	**IAB**
20	IAB	IAB	ZCRFB	ZCRFB	UN	KN	**ZCRFB**	**IAB**
21	IAB	IAB	ZCRFB	ZCRFB	UN	KN	**ZCRFB**	**IAB**
22	MORB	MORB	MORB	MORB	MORB	KN	**MORB**	**MORB**
23	MORB	MORB	MORB	MORB	MORB	KN	**MORB**	**MORB**
24	OIB	OIB	MORB	MORB	MORB	KN	**MORB**	**MORB**
25	MORB	OIB	MORB	MORB	MORB	KN	**MORB**	**MORB**

Table 6.9: N-SLOPE predictions on GS3 test data 26-50

	Novel (87.39%)	SIMCA (90.26%)	LOF (96.80%)	OC-SVM (98.30%)	PLS-DA (92.98%)	ELM (96.43%)	N-SLOPE	True Class
26	MORB	OIB	MORB	MORB	MORB	KN	**MORB**	**MORB**
27	MORB	OIB	MORB	MORB	MORB	KN	**MORB**	**MORB**
28	MORB	OIB	MORB	MORB	UN	KN	**MORB**	**MORB**
29	MORB	MORB	MORB	MORB	MORB	KN	**MORB**	**MORB**
30	MORB	MORB	MORB	MORB	MORB	KN	**MORB**	**MORB**
31	OIB	OIB	OIB	OIB	OIB	KN	**OIB**	**OIB**
32	UN	UN	OIB	UN	OIB	UN	**UN**	**OIB**
33	OIB	UN	OIB	UN	UN	UN	**UN**	**OIB**
34	OIB	UN	OIB	UN	OIB	KN	**OIB**	**OIB**
35	OIB	OIB	OIB	UN	OIB	KN	**OIB**	**OIB**
36	UN	UN	OIB	UN	UN	KN	**UN**	**OIB**
37	MORB	ZCRFB	ZCRFB	ZCRFB	UN	KN	**ZCRFB**	**ZCRFB**
38	ZCRFB	ZCRFB	ZCRFB	ZCRFB	UN	KN	**ZCRFB**	**ZCRFB**
39	ZCRFB	ZCRFB	ZCRFB	ZCRFB	UN	KN	**ZCRFB**	**ZCRFB**
40	ZCRFB	ZCRFB	ZCRFB	ZCRFB	UN	KN	**ZCRFB**	**ZCRFB**
41	ZCRFB	ZCRFB	ZCRFB	ZCRFB	UN	KN	**ZCRFB**	**ZCRFB**
42	ZCRFB	ZCRFB	ZCRFB	ZCRFB	ZCRFB	KN	**ZCRFB**	**ZCRFB**
43	ZCRFB	ZCRFB	ZCRFB	ZCRFB	ZCRFB	KN	**ZCRFB**	**ZCRFB**
44	ZCRFB	ZCRFB	ZCRFB	ZCRFB	ZCRFB	KN	**ZCRFB**	**ZCRFB**
45	ZCRFB	ZCRFB	ZCRFB	ZCRFB	ZCRFB	KN	**ZCRFB**	**ZCRFB**
46	UN	UN	UN	UN	OIB	UN	**UN**	**UN**
47	UN	UN	UN	UN	OIB	KN	**UN**	**UN**
48	UN	UN	UN	UN	MORB	UN	**UN**	**UN**
49	UN	UN	UN	UN	UN	UN	**UN**	**UN**
50	UN	UN	UN	UN	UN	UN	**UN**	**UN**

Table 6.10: N-SLOPE predictions on GS3 test data 51-60

	Novel (87.39%)	SIMCA (90.26%)	LOF (96.80%)	OC-SVM (98.30%)	PLS-DA (92.98%)	ELM (96.43%)	N-SLOPE	True Class
51	UN	UN	UN	UN	OIB	KN	**UN**	**UN**
52	UN	UN	UN	UN	UN	KN	**UN**	**UN**
53	UN	UN	UN	UN	UN	KN	**UN**	**UN**
54	UN	UN	UN	UN	UN	UN	**UN**	**UN**
55	UN	UN	IAB	UN	MORB	UN	**UN**	**UN**
56	UN	UN	IAB	UN	MORB	UN	**UN**	**UN**
57	UN	UN	IAB	UN	MORB	UN	**UN**	**UN**
58	UN	UN	IAB	UN	MORB	UN	**UN**	**UN**
59	UN	UN	IAB	UN	MORB	UN	**UN**	**UN**
60	UN	UN	UN	UN	ZCRFB	KN	**UN**	**UN**

6.4 Summary

N-SLOPE has been validated on three distinct data sets with unique properties. The 150x4 three-class Iris data serves as a baseline metric on a common small data set with a mix of linearly and non-linearly separable classes. The 63x10 four-class Obsidian data set is used to measure N-SLOPE's performance on extremely small, comparatively high-dimensional data relevant to the chemometric domain. Lastly, the 821x45 four-class GS3 data acts as a realistic simulation of nuclear forensics data as a small, high-dimensional data set missing a large portion of available measurements.

N-SLOPE performed perfectly on the Iris data, correctly classifying every unknown in the testing set. SIMCA, OC-SVM, and ELM performed best overall followed by LOF, the novel algorithm, and PLS-DA, respectively. N-SLOPE's strong performance on this data is somewhat expected due to the simplistic nature of the

Iris data set.

N-SLOPE performed fairly well on the Obsidian data with training accuracies in the mid 90's and above, but failed to correctly classify two of the unknowns in the testing set. Both unknowns were incorrectly marked as outliers with one very close decision and one more certain. Upon further examination of the data, it seems reasonable to consider the second incorrectly classified unknown an outlier, due to an uncharacteristically high elemental concentration as described in the previous section. The novel algorithm and SIMCA performed best on this data followed by ELM then LOF, OC-SVM, and PLS-DA. N-SLOPE correctly identified the novel unknown in the testing set and performed well overall given the difficult nature of the Obsidian data.

N-SLOPE performed very well on the GS3 data given the unique challenges posed by this data set. N-SLOPE correctly classified 83.33% of testing samples and correctly identified each of the 16 novel unknowns as not belonging to any of the known classes. This second metric is particularly important, and impressive, as correctly identifying novel unknowns is the main focus of one-class classification and the novel unknowns included in this set were specifically chosen from a variety of excluded classes with the expectation that certain novel unknowns would be difficult to distinguish from the known classes.

Table 6.11: Summary of N-SLOPE classification accuracies on each validation data set

	Novel	SIMCA	LOF	OC-SVM	PLS-DA	ELM	N-SLOPE
Iris	84.62%	100%	92.31%	100%	69.23%	100%	100%
Obsidian	91.67%	91.67%	75.0%	75.0%	75.0%	91.67%	83.33%
GS3	80.0%	85.0%	78.33%	73.33%	58.33%	88.33%	83.33%

As predicted, each individual algorithm within N-SLOPE has strengths and weaknesses that effect its performance on different data sets. SIMCA performed best overall on all three data sets with classification accuracies of 100%, 91.67%, and 85.0%, respectively. The novel algorithm performed near the bottom on Iris, tied for best on Obsidian, and second best on GS3 with classification accuracies of 84.62%, 91.67%, and 80.0%, respectively. PLS-DA performed worst overall on all three data sets with classification accuracies of 69.23%, 75.0%, and 58.33%, respectively. A full summary of classification accuracies can be found in Table 6.11.

6.5 Discussion

N-SLOPE achieved higher classification accuracy and lower false positive rate than each individual algorithm, except for SIMCA, on at least one of the validation data sets. Surprisingly, SIMCA performed best overall on each data set, even slightly outperforming the N-SLOPE ensemble on the Obsidian and GS3 data sets by one additional correctly-classified unknown each. While this result is interesting, it is very unlikely that SIMCA is inherently superior to all the other algorithms. Instead, these results emphasize the fact that SIMCA excels at modeling classes with high within-class variance, as is particularly apparent on the IAB class in the GS3 data set. IAB samples are highly variable with "typical" and "atypical" samples represented and an overall percent deviation of 41.6% higher than any of the other known classes. Furthermore, IAB is linearly separable from the other known classes. These data characteristics produce an optimal environment for SIMCA to perform well on IAB, which is exactly what is shown with SIMCA correctly classifying every single IAB unknown, where other algorithms struggle. Since IAB unknowns are disproportionally represented over unknowns from other classes, making up one third of the total unknowns, SIMCA classification accuracy is artificially inflated. SIMCA struggles with overlapping classes, as seen in the confusion between MORB and OIB in

Tables 6.8 and 6.9, and would likely perform poorly compared to other N-SLOPE algorithms on a data set with highly overlapping classes, minimal within-class variation, or nonlinear data relationships. In these cases, algorithms like LOF, the novel algorithm, and OC-SVM, respectively, would likely perform well. Since prediction results from each algorithm are incorporated into N-SLOPE through ensemble learning, N-SLOPE has the potential to achieve high classification accuracy regardless of specific data set characteristics.

Chapter 7

CONCLUSION

One-class classification is a difficult combination of traditional classification and outlier detection that is here applied to the field of nuclear forensics. Small, high-dimensional, chemometric data from this field exacerbates the problem by creating a challenging domain for traditional machine learning algorithms. A web application has been developed that incorporates N-SLOPE to aid in nuclear forensic investigations with one-class classification support. N-SLOPE introduces a novel one-class classifier and combines five distinct algorithms into a single package that uses ensemble learning to generate classification predictions.

N-SLOPE has been validated on three very different data sets including one realistic set previously used in an official international nuclear forensics exercise. N-SLOPE performed well on each data set with classification accuracies of 100%, 83.33%, and 83.33% on Iris, Obsidian, and GS3, respectively, which is impressive given the difficult characteristics of each set. Furthermore, N-SLOPE correctly identified every novel unknown in each data set proving its worth as a robust one-class classifier. In cases where N-SLOPE produces ambiguous results, such as Unknown 11 from the Obsidian data set, additional investigatative tools included in the web application can be used to refine predictions. N-SLOPE is automated, efficient, and accurate, but uncertain classifications benefit from additional analysis.

N-SLOPE demonstrates strong performance as a generalized one-class classification ensemble. Combining algorithms using ensemble learning bolsters classification accuracy and outlier detection and affords N-SLOPE a high degree of accuracy regardless of data set characteristics. Of the algorithms within N-SLOPE, SIMCA performed best overall, the novel algorithm fell near the top of the pack, and PLS-

DA performed worst overall. The web application with N-SLOPE presented here is a powerful tool to aid in nuclear forensic investigations.

Chapter 8

FUTURE WORK

8.1 Cross-Validation

Each algorithm in N-SLOPE currently undergoes 10-fold cross-validation to determine a training accuracy that represents each algorithm's ability to correctly classify samples in the training set. This is a useful metric and an important part of determining each algorithm's contribution to the N-SLOPE ensemble.

Instead of simply calculating the training accuracy as the percentage of correct classifications, it would be beneficial to expand this metric and separate incorrect classifications as either between-class or out-of-class failures. Between-class error would indicate the algorithm incorrectly assigned the sample to one of the known classes, whereas out-of-class error would indicate the algorithm incorrectly identified the sample as a novel unknown. This distinction would help identify when algorithms are struggling with the data itself or possibly when models are overfitting or underfitting the data.

Breaking these metrics down further by class could highlight particularly difficult inseparable classes and inform better ensemble contribution calculations. Furthermore, adding these cross-validation metrics to the N-SLOPE ensemble as a whole would yield an interesting and informative review of N-SLOPE's overall performance.

8.2 Ensemble Learning

N-SLOPE currently uses an implementation of the sum rule to combine algorithm output and create a final classification decision for each unknown. The sum rule was chosen due to its supposed strong performance and generalizability [18], but the rule itself does not have specific implementation details. N-SLOPE currently combines algorithm output as described in Chapter 5.

While this combination scheme seems to perform well on the validation data as described in Chapter 6, it may be useful to implement additional versions of the sum rule that emphasize algorithms differently. Currently OC-SVM, LOF, and the novel algorithm are somewhat emphasized over the other algorithms due to the nature of the contribution calculations. Implementing a sum rule that slightly emphasized SIMCA and PLS-DA, for example, would be useful when working with linear data. Automating this process may prove difficult because individual algorithm accuracy on one data set does not necessarily predict accuracy on other data sets, as has been shown in Chapter 6.

It would also be interesting to explore other ensemble learning techniques like the median rule because while the sum rule performs best in theory [18], it is possible that N-SLOPE may be more compatible with other rules in implementation.

The addition of new ensemble learning techniques would pair well with updated cross-validation as described above and could allow N-SLOPE to automatically select different combination rules based on advanced training metrics for optimal, data-independent classification.

8.3 Novel Algorithm

The novel algorithm presented here and incorporated into N-SLOPE acts as a one-class classifier with certain strengths, such as computational speed and internal outlier detection (ability to detect outliers uncharacteristically close to the centroid of a known class). Conversely, the novel algorithm performs poorly on small, sparse data and is unable to model complex, nonlinear relationships. Depending on the data set, the novel algorithm falls near the upper middle of the ensemble when it comes to classification accuracy.

Since the novel algorithm already provides some internal outlier detection, it may be useful to extend this functionality and enhance the algorithm's between-class outlier detection capabilities to produce an algorithm more tailored to detecting outliers within a set of known data. This could be done by incorporating the notion of local density from LOF into the set of metrics that are caluclated for each class. Rather than determining class membership simply from distance-to-center, density could be employed to detect unknowns that are close to the class center, but far enough from other points to be labelled outliers. Adding a concept of local density would also help the novel algorithm handle strongly shaped data, although strongly shaped data is unlikely to occur in the high-dimensional spaces in which this algorithm is designed to operate.

One additionally useful modification would be to refactor the algorithm to make it compatible with the `caret package` [20]. This would allow the algorithm to take advantage of the common `preProcess`, `train`, and `predict` functions provided in `caret` [20] and make it significantly easier to develop new features and maintain the algorithm. Furthermore, new developers would be able to work with the algorithm more easily, as its behaviour would be standardized, and the novel algorithm presented here could potentially be published and incorporated into the public `caret package` [20].

BIBLIOGRAPHY

[1] K. Ali. On the link between error correlation and error reduction in decision tree ensembles. *Technical Report from the Department of Information and Computer Science of the University of California, Irvine*, 1995.

[2] M. Amer, M. Goldstein, and S. Abdennadher. Enhancing one-class support vector machines for unsupervised anomaly detection. In *ODD '13 Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description*, pages 8–15, 2013.

[3] E. Anderson. The species problem in iris. *Annals of the Missouri Botanical Garden*, 23:457–509, 1936.

[4] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

[5] K. V. Branden and M. Hubert. Robust classification in high dimensions based on the SIMCA method. *Chemometrics and Intelligent Laboratory Systems*, 79(1-2):10–21, 2005.

[6] M. M. Breunig, H. Kriegel, R. T. Ng, and J. Sander. LOF: identifying density-based local outliers. In *ACM sigmod record*, volume 29, pages 93–104. ACM, 2000.

[7] M. Bylesjö, M. Rantalainen, O. Cloarec, J. K. Nicholson, E. Holmes, and J. Trygg. OPLS discriminant analysis: combining the strengths of PLS-DA and SIMCA classification. *Journal of Chemometrics*, 20(8-10):341–351, 2006.

[8] W. Chang, J. Cheng, J. Allaire, Y. Xie, and J. McPherson. *shiny: Web Application Framework for R*, 2017. R package version 1.0.5.

[9] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Human Genetics*, 7(2):179–188, 1936.

[10] L. Friedland, A. Gentzel, and D. Jensen. Classier-adjusted density estimation for anomaly detection and one-class classication. *SIAM*, 2014.

[11] A. Gosso. *elmNN: Implementation of ELM (Extreme Learning Machine) algorithm for SLFN (Single Hidden Layer Feedforward Neural Networks)*, 2012. R package version 1.0.

[12] T. Howley, M. G. Madden, M. O'Connell, and A. G. Ryder. The effect of principal component analysis on machine learning accuracy with high-dimensional spectral data. *Knowledge-Based Systems*, 19(5):363–370, 2006.

[13] Y. Hu, W. Murray, Y. Shan, and Australia. *Rlof: R Parallel Implementation of Local Outlier Factor(LOF)*, 2015. R package version 1.1.1.

[14] G. Huang, L. Chen, and C. K. Siew. Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Trans. Neural Networks*, 17(4):879–892, 2006.

[15] J. H. Janssens, I. Flesch, and E. O. Postma. Outlier detection with one-class classifiers from ML and KDD. *International Conference on Machine Learning and Applications*, 2009.

[16] A. Karatzoglou, D. Meyer, and K. Hornik. Support vector machines in R. *Journal of Statistical Software*, 15(9), 2006.

[17] S. S. Khan and M. G. Madden. One-class classification: Taxonomy of study and review of techniques. *CoRR*, abs/1312.0049, 2013.

[18] J. Kittler, M. Hatef, R. P. Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

[19] B. Kowalski, T. Schatzki, and F. Stross. Classification of archaeological artifacts by applying pattern recognition to trace element data. *Analytical Chemistry*, 44(13):2176–2180, 1972.

[20] M. Kuhn, J. Wing, S. Weston, A. Williams, C. Keefer, A. Engelhardt, T. Cooper, Z. Mayer, B. Kenkel, the R Core Team, M. Benesty, R. Lescarbeau, A. Ziem, L. Scrucca, Y. Tang, C. Candan, and T. Hunt. *caret: Classification and Regression Training*, 2018. R package version 6.0-79.

[21] S. Lê, J. Josse, and F. Husson. FactoMineR: A package for multivariate analysis. *Journal of Statistical Software*, 25(1):1–18, 2008.

[22] Q. Leng, H. Qi, J. Miao, W. Zhu, and G. Su. One-class classification with extreme learning machine. *Mathematical Problems in Engineering*, 2015, 2015.

[23] W. Liu, G. Hua, and J. R. Smith. Unsupervised one-class learning for automatic outlier removal. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[24] M. G. Madden and T. Howley. A machine learning application for classification of chemical spectra. *Applications and Innovations in Intelligent Systems XVI*, pages 77–90, 2009.

[25] L. M. Manevitz and M. Yousef. Document classification on neural networks using only positive examples (poster session). In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 304–306. ACM, 2000.

[26] L. M. Manevitz and M. Yousef. One-class SVMs for document classification. *Journal of Machine Learning Research*, 2001.

[27] B. Mertens, M. Thompson, and T. Fearn. Principal component outlier detection and SIMCA: A synthesis. *Analyst*, page 27772784, 1994.

[28] D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, and F. Leisch. *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*, 2017. R package version 1.6-8.

[29] K. J. Moody, P. M. Grant, and I. D. Hutcheon. *Nuclear Forensic Analysis.* CRC Press, 2005.

[30] M. M. Moya and D. R. Hush. Network constraints and multi-objective optimization for one-class classification. *Neural Networks*, 9(3):463–474, 1996.

[31] M. Nakazawa. *fmsb: Functions for Medical Statistics Book with some Demographic Data*, 2018. R package version 0.6.3.

[32] J. Ooms, D. James, S. DebRoy, H. Wickham, and J. Horner. *RMySQL: Database Interface and 'MySQL' Driver for R*, 2018. R package version 0.10.14.

[33] R Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2018.

[34] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):14431471, 2001.

[35] B. Schölkopf, R. C. Williamson, A. Smola, and J. Shawe-Taylor. SV estimation of a distribution's support. In *Advances in Neural Information Processing Systems*, 1999.

[36] D. Stevenson, F. Stross, and R. Heizer. An evaluation of X-ray fluorescence analysis as a method for correlating obsidian artifacts with source location. *Archaeometry*, 13(1):17–25, 1971.

[37] B. Stumpe, T. Engel, B. Steinweg, and B. Marschner. Application of PCA and SIMCA statistical analysis of FT-IR spectra for the classification and identification of different slag types with environmental origin. *Environmental Science & Technology*, 46(7):3964–3972, 2012.

[38] D. M. Tax and R. P. Duin. Support vector domain description. *Pattern Recognition Letters*, 20:1191–1199, 1999.

[39] D. M. Tax and R. P. Duin. Support vector data description. *Machine Learning*, 54(1):4566, 2004.

[40] V. Todorov. *rrcovHD: Robust Multivariate Methods for High Dimensional Data*, 2016. R package version 0.2-5.

[41] S. Van Buuren and K. Groothuis-Oudshoorn. mice: Multivariate Imputation by Chained Equations in R. *Journal of Statistical Software*, 45(3):1–67, 2011.

[42] B. M. Wise, N. B. Gallagher, R. Bro, J. M. Shaver, W. Windig, and R. S. Koch. *PLS_Toolbox 4.0*. Eigenvector Research, Inc, 3905 West Eaglerock Drive, Wenatchee, WA 98801 USA, 2006.

[43] H. Yang, P. R. Griffiths, and J. Tate. Comparison of partial least squares regression and multi-layer neural networks for quantification of nonlinear systems and application to gas phase Fourier transform infrared spectra. *Analytica Chimica Acta*, 489(2):125 – 136, 2003.

[44] T. Zou, Y. Dou, H. Mi, J. Zou, and Y. Ren. Support vector regression for

determination of component of compound oxytetracycline powder on near-infrared spectroscopy. *Analytical Biochemistry*, 355(1):1 – 7, 2006.