Improving Learning Outcomes by Using Clustering Validity Analysis to Reduce Label Uncertainty

by

Man Chon U

(Under the direction of Khaled Rasheed)

Abstract

When people make critical decisions, they often consider the opinions of multiple experts from different domains rather than committing themselves to a single expert or relying on their own judgment. The same principle applies in machine learning for the construction of robust predictive models. However, the outcome of each expert/model might include a certain degree of uncertainty, which affects the robustness of the combined predictive model.

In this dissertation, we present two effective machine learning frameworks (ROMP and VAMO) to tackle challenging problems in the domains of cancer prediction and malware clustering. Our frameworks successfully improve the learning outcomes from multiple sources by using cluster validity analysis to reduce label uncertainty.

In ROMP, we introduce novel features for identifying cancer-causing mutations in the cancer kinome, and utilize various feature selection methods to evaluate our proposed features. We combine multiple classifiers for the prediction of rare oncogenic mutations, followed by using the Expectation Maximization (EM) clustering algorithm and our self-invented cluster validity metrics to improve the learning outcomes from the ensemble classifier as well as to identify suspicious mutations in the dataset. Our framework successfully discovered rare causative mutations that were later confirmed by lab experiments.

Another framework – VAMO, provides a fully automated assessment of the quality of malware clustering results. VAMO does not require a manual mapping between malware family labels output by different AV scanners. Furthermore, VAMO does not discard malware samples for which a majority voting-based consensus cannot be reached. Instead, VAMO explicitly deals with the inconsistencies typical of multiple AV labels to build a more representative reference set. Our evaluation, which includes extensive experiments in a controlled setting and a real-world application, show that VAMO performs better than majority voting-based approaches, and provides a way for malware analysts to automatically assess the quality of their malware clustering results.

INDEX WORDS: Machine Learning, Classification, Clustering, Model Ensemble Cluster Validity Analysis, Protein Kinase, Malware

Improving Learning Outcomes by Using Clustering Validity Analysis to Reduce Label Uncertainty

by

Man Chon U

B.Eng., Shanghai University, Shanghai, China, 2006

A Dissertation Submitted to the Graduate Faculty
of The University of Georgia in Partial Fulfillment
of the
Requirements for the Degree

Doctor of Philosophy

Athens, Georgia

2013

Improving Learning Outcomes by Using Clustering Validity Analysis to Reduce Label Uncertainty

by

Man Chon U

Improving Learning Outcomes by Using Clustering Validity Analysis to Reduce Label Uncertainty

Man Chon U

July 25, 2013

Acknowledgments

First and foremost, I would like to express my deepest appreciation to Dr. Khaled Rasheed, my major advisor who gave me invaluable help in the past four years. Khaled has always allowed me complete freedom to define and explore my own directions in research. I am most fortunate to have him as my advisor. His excellent guidance and humble character encouraged me through the end of my study, and inspired me to explore the unknown world.

Furthermore, many thanks to Dr. Natarajan Kannan, Dr. Roberto Perdisci, and Dr. Hamid R. Arabnia for being the members of my committee, for all their insightful advice and support.

I also want to thank Dr. Keith Harris, head of Department of Pathology (College of Veterinary Medicine) for providing me with much appreciated financial support during my degree.

I thank Dr. Eric Talevich for his excellent work in the ROMP project. Eric is a great partner with excellent technical background. I also thank Akul Dewan for his contribution for the ROMP project.

Finally, I would like to thank my parents, they have provided unconditional support and encouragement through both the highs and lows of my life. Whatever I have accomplished, they deserve to share the credit!!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

We generate a huge amount of data every minute in our daily lives, from everyday activities such as searching, online shopping, phone calls, web browsing, and accessing social media sites, etc. With the tremendous data nowadays, lots of researchers have been trying various methods to extract useful information from different sources of data. For this reason, machine learning (both supervised and unsupervised learning) has become one of the most active research areas in not only computer science community, but also in math, statistics, and even linguistics. The major reason of that is due to its diverse applicability; you can find machine learning in everywhere such as gene discovery, cancer prediction, spam detection, malware clustering, computer vision, natural language processing, and medical diagnosis.

However, clean and well structured data seldom exists. Instead, most of the available data somehow contain different levels of noise. Sometimes the given data of a machine learning task is labeled, but labeled data in some domains are either too difficult or expensive to obtain. It turns out that some problems that require machine learning techniques to get involved do not have clearly labeled data available, instead, the labels might be partially missing, no label available at all, or the available data was labeled with certain degrees of uncertainty.

Although many methods such as feature selection, feature extraction, and sophisticated learning algorithms were introduced in the past couple of decades, research has shown that no single machine learning approach is superior across all different domains. In fact, predictions made by a single learning algorithm may implicitly include bias to a certain extent. Therefore, combining multiple learning outcomes and/or multiple learning approaches can avoid such bias and reduce the possible label uncertainty to finally lead to a more robust prediction result.

In this dissertation, we introduce two machine learning frameworks (ROMP and VAMO) that combine multiple learning outcomes as well as improve the learning outcomes by using clustering validity analysis to reduce label uncertainty. We present these two frameworks in two different problem domains, namely cancer prediction and malware clustering. The reason for choosing these two problem domains is that data available in both of these domains contain some degrees of uncertainty; confirmed cancer-causing mutations (correct labels) are limited, and mutations labeled as cancer-causing in some public databases (i.e. somatic mutations observed in clinical samples) are under certain assumptions; labels of a same malware sample that generated by different anti-virus scanners are greatly different, and the mapping between these labels are usually unavailable. Furthermore, the problem formulation in these two domains can be apply to many other research that have similar data structure, or research that have similar problem to solve.

The goal for ROMP is to identify rare oncogenic mutations as well as mutations with suspicious labels, while the goal for VAMO is to provide a fully automated quantitative analysis of the validity of malware clustering results.

1.2 Dissertation Outline

Considering the different reading preference between audiences from different backgrounds – biologists care more about the results than the computational methodologies, and computer scientists prefer to read about the methodologies before the results – the contents in Chapters 3 and 4 are organized slightly differently.

Chapter 2 defines the terms and gives an overview of concepts of supervised and unsupervised learning. It also describes certain popular ensemble methods including Voting, Stacking, and Grading, that we use in this dissertation. Furthermore, the difference between the two main categories of feature selection methods – Wrapper Methods and Filter Methods – is illustrated. Finally, it provides the definition of the common measurement indexes used in supervised and unsupervised learning.

Chapter 3 begins by introducing the definition of cancer and the rationale for utilizing machine learning techniques to predict causative mutations. Followed by defining the problem domain that our research focus on as well as the difference from previous studies. Section 3.5 describes the insights of our data source and the partition of data for later experiments. The details of feature preprocessing and feature selection are given in Section 3.6. Section 3.7 introduces the details of our supervised learning module, including the learning methods, evaluation of individual classifiers, prioritization of putative EGFR mutations with ensemble classifier, and the lab experiment results based on our prioritization. Section 3.8 introduces the details of our unsupervised learning module, including the learning methods, our self-invented metric for the measurement of level of oncogenicity, and the results on the most updated dataset. Section 3.9 describes the way of how we combine the supervised and the unsupervised learning module which leads to our effective framework, followed by providing the prediction results of the putative mutations as well as mutations with suspicious labels on the most updated dataset.

Chapter 4 first explains the application of malware clustering and its importance, followed by stating the existing problems in the field of evaluating malware clustering results. Afterwards, it describes how our framework (VAMO) can help to solve the problem by constructing a system that enables an automatic quantitative analysis of the validity of malware clustering results. Section 4.2 describes the previous studies in cluster validity analysis and malware clustering, as well as introduces how our framework is different from the previous research. Section 4.2 provides quantitative information regarding the inconsistency typical of multiple AV labels, followed by the discussion of the background concepts that VAMO use to perform automated clustering validity analysis. The detailed system design and the definition of terms are given in Section 4.3. Afterwards, Section 4.4 provides more details on how VAMO builds the reference clustering by leveraging multiple AV labels, and how the clustering validity indexes are computed to compare third-party malware clustering results to VAMO's reference clustering. The evaluation of VAMO versus majority voting approach is given in Secion 4.5. Finally, discussion and conclusion are provided in Section 4.7.

Finally, Chapter 5 presents conclusions and suggests future work.

Chapter 2

Background

2.1 Machine Learning

Machine Learning is "the study of computer algorithms that improve automatically through experience." [2]. Machine Learning techniques have been applied successfully on a myriad of problems ranging from autonomous vehicle driving [3, 4] to stock market prediction [5], to malware clustering [6]. The main idea behind machine learning is to construct a model from existing data, so as to better predict (classify, regression, cluster) properties of the unseen data.

In general, machine learning problems are divided into two big categories: "supervised" and "unsupervised" learning. The difference between them is that in supervised learning, the examples (instances) are predetermined (labeled), a learner learns from the data and corresponding labels to construct and evaluate a predictive model. In unsupervised learning is the opposite, the examples are unlabeled, an unsupervised learner does not have any predetermined fact about the given data, it groups similar data into the same cluster by some means.

More details about supervised and unsupervised learning, as well as some major concepts of machine learning are introduced in the following subsections.

2.2 Supervised Learning

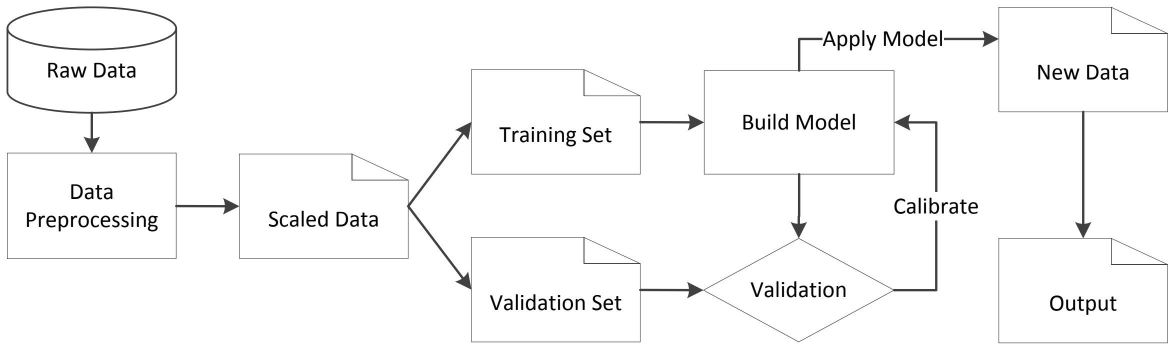


Figure 2.1: Supervised Learning Workflow

Figure 2.1 illustrates the workflow of supervised learning. Supervised learning is a task of constructing a predictive model from labeled data. In supervised learning, each instance (data sample, usually represented as a vector) is paired with a predetermined label, the labels are called classes that are usually a finite set. Supervised learning algorithms generalize from the labeled data, search for patterns and construct predictive models that can be used for correctly classifying unseen instances. Supervised learning can be divided into two broad categories:

- **Classification.** Mapping the input vector into unordered categorical output values. Examples of classification problems include: a gene mutation is causative or non-causative; the weather tomorrow will be sunny, rainy, or cloudy; the risk of approving a loan is high, medium, or low.

- **Regression.** Mapping the input vector into either continuous or discrete numerical output values. Examples of regression problems include: the possibility of a gene mutation to be causative, or non-causative; the probability that the weather tomorrow is sunny (or rainy, or cloudy); a numerical prediction to the risk assessment of approving a loan.

2.3 Unsupervised Learning

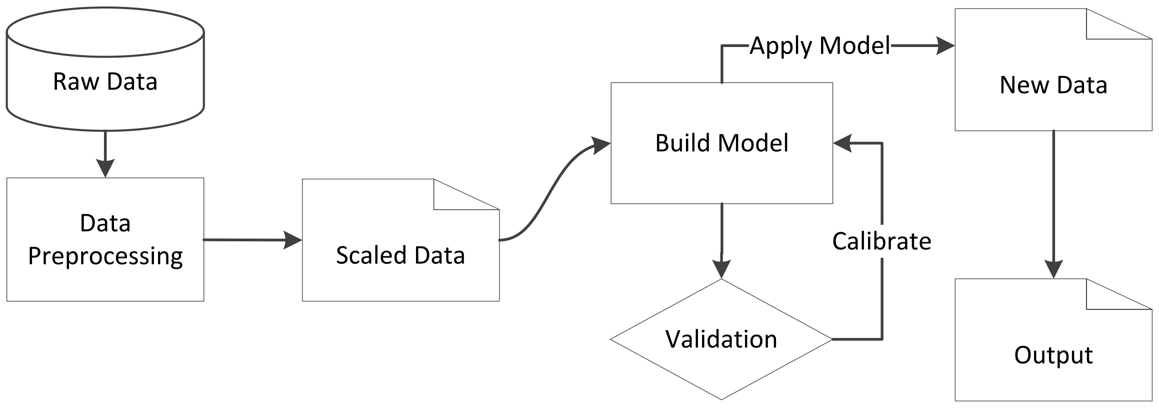


Figure 2.2: Unsupervised Learning Workflow

Figure 2.1 illustrates the workflow of unsupervised learning. Unsupervised learning is a machine learning task of finding hidden structure (patterns) in unlabeled input data. In unsupervised learning, there is no supervised target output or reward from its environment associated with each instance. This distinguishes unsupervised learning from supervised learning and reinforcement learning. It may seem somewhat mysterious to imagine what the learning algorithm could possibly learn given that it does not get any feedback from its environment, but unsupervised learning algorithms can build representations of the input data that can be used for decision making, predicting future inputs, segmenting data into groups, etc. Unsupervised learning can be divided into two broad categories:

- **Clustering:** Organizing data into clusters so that there is high intra-cluster similarity and low inter-cluster similarity. Examples of clustering problems include: identifying malware samples that belong to the same family; discovering distinct customer groups in the market; grouping documents with the same/similar topic.

- **Dimensionality Reduction:** Reducing the number of random variables (attributes) of the data. Algorithms for dimensionality Reduction include but are not limited to: principal component analysis (PCA); independent component analysis (ICA); singular value decomposition (SVD).

2.4 Ensemble Learning

Overview of Model Ensemble

Methods for the combination of multiple predictive models are referred to as ensemble methods in the machine learning community [7]. The goal of ensemble learning is, by combining multiple predictive models, to obtain better predicative performance than any of the ensemble members. In contrast to common machine learning approaches which try to learn one predictive model from available data, ensemble methods try to construct a set of predictive models from the date and then combine them through some way for further use. A necessary prerequisite for an ensemble classifier to outperform its best performing member is for the mistakes made by the ensemble members to be uncorrelated. The underlying logic of the member classifiers should be diverse as much as possible, and the majority of the member classifiers should be accurate, because a few good votes will be easily overturned by many bad votes. There are many ways to combine multiple classifiers. The most popular methods to combine individual classifiers of the same type are Bagging [8] and Boosting [9], while the methods for combining individual classifiers from different types include Majority Voting, Stacking [10], and Grading [11]. Here we do not consider the methods that combine multiple classifiers of the same type. We will describe the basic idea of how Majority Voting, Stacking, and Grading work in general.

Table 2.1 shows a dataset for a learning problem with m instances (examples), each of which has n attributes and a class label. Our example is a binary classification learning task. Thus the number of possible values for the class label are only two (T and F in this case). We then assume that there are l individual classifiers. Table 2.2 is a prediction matrix that shows the prediction results of each instance generated by each classifier. We assume that the results are produced by some cross-validation scheme.

Table 2.1: Sample Training Data

Attributes				Class
x_{11}	x_{12}	$\cdots$	x_{1n}	T
x_{21}	x_{22}	$\cdots$	x_{2n}	F
x_{31}	x_{32}	$\cdots$	x_{2n}	F
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
x_{m1}	x_{m2}	$\cdots$	x_{mn}	T

Table 2.2: Sample Output of Individual Classifiers

Classifier$_1$	Classifier$_2$	$\cdots$	Classifier$_l$
T	F	$\cdots$	T
F	T	$\cdots$	T
F	F	$\cdots$	T
$\cdots$	$\cdots$	$\cdots$	$\cdots$
F	T	$\cdots$	F

Majority Voting

One way to combine the outputs of multiple individual learners is by voting, either through an unweighted or a weighted approach. A voting mechanism is easy to implement, and the performance is easy to track. However, a prerequisite for unweighted voting (simple majority voting) is that all member learners should perform comparably well. If the majority of the learners make incorrect prediction, the ensemble learner will also make its prediction incorrectly. Thus an enhanced weighted voting mechanism is introduced to solve this issue. In a weighted voting mechanism, each of the trained classifiers casts one vote indicating the prediction of a given instance. Each vote is weighted by a predefined mechanism: $Prediction = \sum_{i=1}^{l}(w_i \times prediction_i)$, in where $\sum_{i=1}^{l} w_i = 1$. In Chapter 3, we use an analogous method – Weighted Average – to compute the final probability of each instance being classified as a specific class (details are given in Section 3.7).

Stacking

Stacking is usually used to combine models of different type. Stacking uses the metalearner concept to replace the voting step in the majority voting approach. There are two levels of learners in Stacking, base learner and meta learner. The prediction outputs of the base models (a.k.a. level-0 models) are the inputs to the metamodel (a.k.a. level-1 model). The basic idea of Stacking is to use the predictions of the individual classifiers as attributes in a new training set that keeps the original class labels. Instances are first input into the level-0 classifiers (base-classifiers), and each of the learners generates its own prediction on the entire dataset. All those predictions are then fed into the level-1 classifier (meta-classifier) as input to generate the final prediction, with the original class labels as the target attribute. In our example, the training data for a meta-classifier is described in Table 2.3. Since level-0 learners have already performed most of the work, it is reasonable to choose a relatively simple algorithm as the level-1 classifier.

Table 2.3: Sample Output of Base-Classifiers in Stacking (Input for Meta-Classifier)

$\texttt{Classifier}_1$	$\texttt{Classifier}_2$	$\cdots$	$\texttt{Classifier}_l$	Actual Class
T	F	$\cdots$	T	T
F	T	$\cdots$	T	F
F	F	$\cdots$	T	F
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
F	T	$\cdots$	F	T

Grading

Assume that there are l individual classifiers; the basic idea of the Grading method is to evaluate the predictions generated by each of the l base classifiers, then use the graded evaluation result of each instance as the target attribute and keep the original attributes for training a set of meta classifiers that learn to predict when the base classifier is correct. Table 2.4 shows the graded evaluation of the prediction outputs of the base classifiers, correct predictions are graded as "+" and incorrect predictions are graded as "−". Each base classifier generates a two-class training set for the metaclassifier. We therefore have l two-class training sets ("+" and "−"). We then train l metaclassifiers (Table 2.5), each of which gets exactly one of those training sets. Therefore, each meta-classifier tries to predict when its base classifier makes a correct/incorrect prediction. During classification, each base classifier makes a prediction for the input instance. Those base classifiers that are predicted to be correct by the meta-classification schemes make the final prediction, by voting or by making use of the confidence estimates of the base classifiers.

Table 2.4: Sample Output of Individual Classifiers in Grading

Classifier$_1$	Classifier$_2$	$\cdots$	Classifier$_l$
$+$	$-$	$\cdots$	$+$
$+$	$-$	$\cdots$	$-$
$+$	$+$	$\cdots$	$-$
$\cdots$	$\cdots$	$\cdots$	$\cdots$
$-$	$+$	$\cdots$	$-$

Table 2.5: Sample Training Sets for Grading

Attributes				Class		Attributes				Class
x_{11}	x_{12}	$\cdots$	x_{1n}	$+$		x_{11}	x_{12}	$\cdots$	x_{1n}	$+$
x_{21}	x_{22}	$\cdots$	x_{2n}	$+$	$\cdots$	x_{21}	x_{22}	$\cdots$	x_{2n}	$-$
x_{31}	x_{32}	$\cdots$	x_{3n}	$+$		x_{31}	x_{32}	$\cdots$	x_{3n}	$-$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$		$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
x_{m1}	x_{m2}	$\cdots$	x_{mn}	$-$		x_{m1}	x_{m2}	$\cdots$	x_{mn}	$-$

2.5 Feature Selection

The quality of the input dataset is the most important factor, out of many, that affects the success of a machine learning task. People who are not familiar with Machine Learning usually think that having more features would better describe the data, and therefore make the learning algorithms perform better. However, practical experiments showed that is not always the case as some features might be redundant. Therefore, the need for feature subset selection becomes essential.

Feature subset selection aims at identifying and removing as many irrelevant (or redundant) features as possible [12]. Feature subset selection algorithms fall into two broad approaches: One approach is to make an independent assessment of the feature subset based on the general characteristics of the data, and the other is to assess the subset using the machine learning algorithm that will be used on the data. The former approach is called the filter approach [13, 14] because the irrelevant attributes get filtered out and the filtered data is then presented to the machine learning algorithm. The second approach is called the wrapper approach [13, 14], because the machine learning algorithm is involved and wrapped in the feature selection process. Figure 2.3 and Figure 2.4 depict the way in which wrapper based and filter based approaches work. Since the underlying mechanisms between filter and wrapper approaches are different, the feature subsets that are selected by both approaches are expected to vary to some extent.

Wrapper Approach

Wrapper methods for feature selection use a prediction model to assign scores to different feature subsets. Each feature subset is used to train a prediction model, which is tested on a hold-out set. Computing the error rate of the model gives the score for the corresponding feature subset. Wrapper methods use a search algorithm to search through the space of possible features and evaluate each feature subset by applying a prediction model on the feature subset. Since wrapper methods require to train a new prediction model for each feature subset, they are very computationally intensive. Although they usually provide the best performing feature set for that particular type of model, wrapper methods can have a risk of over fitting to the prediction model. Figure 2.3 shows an overview of how wrapper methods work.

Filter Approach

Filters are similar to Wrappers in the search approach, but instead of evaluating using a model, a simpler filter is evaluated. Filter methods for feature selection use heuristics based on general characteristics of the data instead of the error rate of a learned model to score a feature subset. The measurement used in filter methods is chosen to be fast to

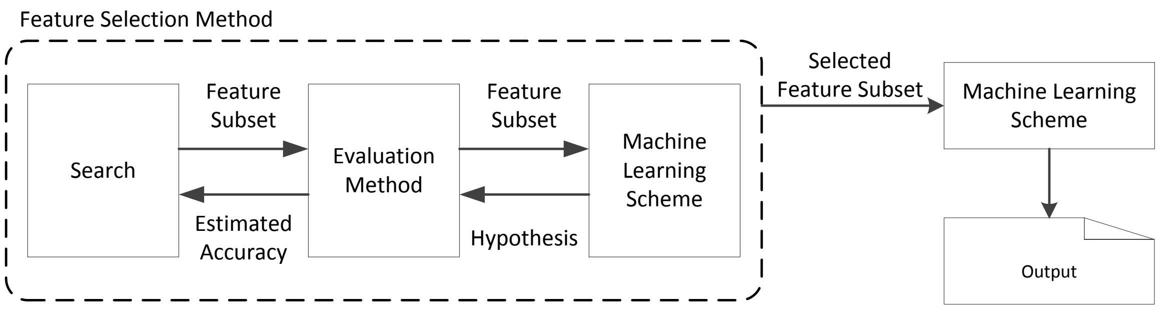


Figure 2.3: Wrapper Feature Selection

compute, whilst still capturing the usefulness of the feature subset. Mutual Information, Pearson product-moment correlation coefficient, and the inter/intra class distance are some of the commonly used measurements. Filters are usually less computationally intensive than wrappers, but they produce a feature set which is not tuned to a specific type of predictive model. Many filters provide a feature ranking rather than an explicit best feature subset, and the cut off point in the ranking is chosen via cross-validation. Figure 2.4 shows an overview of how filter methods work.

2.6 Performance Metrics

Performance metrics are measurement values that allow us to evaluate the performance of the learning models. Metrics that are used to measure the performance of supervised learning models are different from those used to measure the performance of unsupervised learning models. We describe some common performance metrics that are used in this dissertation next.

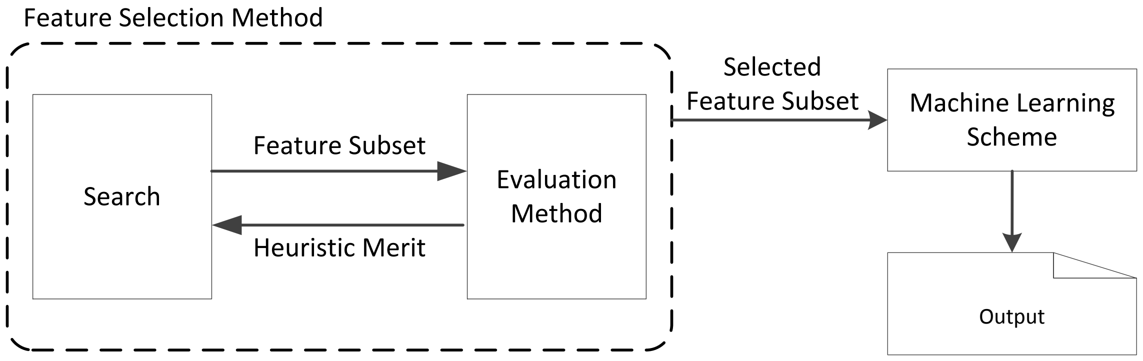


Figure 2.4: Filter Feature Selection

Measurement Indexes Used in Supervised Learning

In supervised learning, performance metrics are calculated from the predictions of the classifiers. Most of those performance metrics are usually calculated from a confusion matrix. We use a binary classification (yes and no) as an example to explain certain basic measurement indexes commonly used in supervised learning.

In a binary classification case with classes: yes or no, go or not go, causative or non-causative, a classifier can produce four different outcomes as Table 2.6. The true positives (TP) and true negatives (TN) are correct classifications, meaning that an instance is correctly predicted as yes (or positive) when it is actually yes, or an instance is correctly predicted as no (or negative) when it is actually no. The false positive (FP) and false negative (FN) are incorrect classifications. They occur when an instance is incorrectly predicted as yes when it is actually no (FP), or an instance is incorrectly predicted as no when it is actually yes (FN).

- **Accuracy**. The accuracy is the percentage of correctly classified instances; $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$. This is the most common performance measurement. However, the

Table 2.6: Confusion Matrix

		Predicted Class	
		Yes	No
Actual Class	Yes	True Positive (TP)	False Negative (FN)
	No	False Positive (TP)	True Negation (TN)

problem with accuracy is that it assumes equal cost for both kinds of errors. A 99% accuracy can be excellent, but it can also be poor if the dataset is highly imbalanced and the class of interest is the minority class.

- **TP Rate**. The true positive rate is the TP divided by the total number of positives; $TP\_Rate = \frac{TP}{TP+FN}$
- **FP Rate**. The false positive rate is FP divided by the total number of negatives; $FP\_Rate = \frac{FP}{FP+TN}$
- **Recall**. The recall is the same as the TP rate, which is the percentage of true positive instances from all the instances that are actually positive; $Recall = \frac{TP}{TP+FN}$
- **Precision**. The precision is the percentage of true positive instances from all the instances classified as positive by the classifier; $Precision = \frac{TP}{TP+FP}$
- **F-Measure**. The F-Measure is the harmonic mean (or weighted average) which considers both the precision and the recall of the test; $FMeasure = \frac{2 \times Recall \times Precision}{Recall+Precision} = \frac{2 \times TP}{2 \times TP+FP+FN}$

For all indexes above, which take values in $[0, 1]$, higher values indicate a better performance of the classifier.

Measurement Indexes Used in Unsupervised Learning

Unlike in supervised learning settings, analyzing the performance of the clustering results is intrinsically hard. Three main *cluster validity* approaches are described in [15]: (1) *external criteria* evaluate the clustering results by comparing them to a pre-specified structure, or *reference clustering*; (2) *internal criteria* rely solely on quantities derived from the data vectors in the clustered dataset (e.g., using a proximity matrix, and computing quantities such as inter- and intra-cluster distances); (3) *relative criteria* compare clustering results obtained using the same clustering algorithm with different parameter settings, to identify the best parameter configuration.

Based on the dataset we used in this dissertation, we mainly focus on the external validity indexes defined as follows. Assuming a reference clustering (ground truth) is available, let $\mathcal{M}$ be our dataset, $\mathbf{Rc} = \{Rc_1, .., Rc_s\}$ be the set of s *reference clusters*, and $\mathbf{C} = \{C_1, .., C_n\}$ be our clustering results over $\mathcal{M}$. Given a pair of data samples (m_1, m_2), with $m_1, m_2 \in \mathcal{M}$, we can compute the following quantities:

- a is the number of pairs (m_1, m_2) for which if both samples belong to the same reference cluster Rc_i, they also belong to the same cluster C_j.

- b is the number of pairs (m_1, m_2) for which both samples belong to the same reference cluster Rc_i, but are assigned to two different clusters C_k and C_h.

- c is the number of pairs (m_1, m_2) for which both samples belong to the same cluster C_i, but are assigned to two different reference clusters Rc_k and Rc_h.

- d is the number of pairs (m_1, m_2) for which if the samples belong to two different reference clusters Rc_i and Rc_j, they also belong to different clusters C_l and C_m.

Based on the above definitions, we can compute the following external cluster validity indexes [15]:

- **Rand Statistic**. $RS = \frac{a+d}{a+b+c+d} = \frac{a+d}{|\mathcal{M}|}$
- **Jaccard Coefficient**. $JC = \frac{a}{a+b+c}$
- **Folkes and Mallows Index**. $FM = \frac{a}{\sqrt{(a+b)(a+c)}}$
- **Precision**. $Prec = 1/n \cdot \sum_{j=1}^{n} \max_{k=1,..,s}(|C_j \cap Rc_k|)$
- **Recall**. $Rec = 1/s \cdot \sum_{k=1}^{s} \max_{j=1,..,n}(|C_j \cap Rc_k|)$
- **F1 Index**. $F1 = 2\frac{Prec \cdot Rrec}{Prec + Rrec}$

For all three indexes above, which take values in $[0, 1]$, higher values indicate a closer similarity between the clustering **C** and the reference clustering **Rc**.

Chapter 3

ROMP: Prioritization of Rare Oncogenic Mutations in Cancer Kinome

Given a labeled dataset, supervised learning approaches can be used to analyze the data and to produce an inferred function for mapping new samples. However, given a dataset which is labeled with certain degrees of uncertainty due to the constraints of the problem domain, supervised learning schemes alone might not be sufficient. In this chapter, we present ROMP[1], an effective framework that first constructs a high performance ensemble classifier using the labeled dataset containing certain degrees of uncertainty, followed by utilizing clustering algorithm and our self-invented cluster validity metrics to improve the learning outcomes from the ensemble classifier to reduce label uncertainty in the dataset. The prediction that generated by ROMP is a probability score, characterizing the prediction of a given sample to be a specific class.

ROMP can be used in any problem domain with labeled dataset, and it will be a great

[1]Rare Oncogenic Mutation Predictor.

fit to domains that satisfy (or partially satisfy) the following criteria: (1) Labels in the dataset contain some degree of uncertainty; (2) Prefers to consider the outputs from multiple experts (learning algorithms); (3) Requires a probability score rather than just a class label as prediction. In this chapter, we introduce ROMP and apply it to the problem domain of genetic mutations in cancer, where many of the datasets currently available contain substantial uncertainty.

Cancer is a genetic disease which develops through a series of somatic mutations, a subset of which drive cancer progression. Although cancer genome sequencing studies are beginning to reveal the mutational patterns of oncogenes in various cancers, identifying the small subset of "causative" mutations from the large subset of "non-causative" mutations, which accumulate as a consequence of the disease, is a challenge. We present ROMP, an effective framework for identifying causative mutations in human protein kinases, a class of signaling proteins known to be frequently mutated in human cancers.

Our input dataset consists of gene mutations labeled as causative or non-causative with varying degrees of uncertainty. Given an additional set of putative mutations, the task is to output a reliable ranked list of the mutations according to the predicted probability of a mutation being causative, and to identify mutations with suspicious labels. This application is relevant to biological research in prioritizing the selection of rare oncogenic mutations for further analysis in the laboratory.

We evaluate the performance of 11 well known supervised learners and show that a multiple classifier approach, which combines the outputs of 11 individual learners, significantly improves the classification of known causative mutations. The classifiers are trained and compared on several different datasets. We introduce several novel features related specifically to structural and functional characteristics of protein kinases and find that the level of conservation of the mutated residue at specific evolutionary depths is an important predictor of oncogenic effect. We also consolidate the novel features and the ensemble classifier to prioritize and experimentally test a set of rare unconfirmed mutations in the epidermal growth factor receptor tyrosine kinase (EGFR). Our studies predict T725M and L861R as rare causative mutations in EGFR inasmuch as these mutations display increased autophosphorylation activity in the absence of the activating EGF ligand.

Afterwards, we apply the unsupervised learning module to the most promising dataset, COSMIC-FG1 v57. We first use the expectation maximization (EM) algorithm to cluster the mutations, then we use our self-invented cluster validity metrics to measure the level of oncogenicity of each of the instances (mutations) based on similarity between the mutations, on top of the clustering outputs. This approach can help to reduce the label uncertainty by identifying suspicious mutations in the given dataset.

Finally, we effectively combine the supervised and unsupervised learning outputs to generate a prioritization list of a set of rare oncogenic mutations in the cancer kinome. We also generate a list of mutations with suspicious labels. The combination process is user tunable, based on a user's preference on supervised and unsupervised learning methods. This procedure to flag suspicious data points provides a mechanism to improve the learning outcomes by reducing the uncertainty in the dataset, an important tool to help Biologists refine potentially messy datasets.

3.1 Introduction

Since the beginning of the last century, various types of cancers have risen in prominence as a major cause of human mortality, yet no single treatment has been found to effectively cure all cancers.

Cancer is a complex disease in which healthy cells undergo a series of genetic changes, eventually becoming cancerous, growing uncontrollably and spreading throughout the body [16]. Identification of the specific genetic changes that promote cancer traits within a cell can yield clues into potential treatments for the disease. Large-scale cancer genome sequencing studies have thus been initiated in order to catalogue the mutations observed in human cancers [17–20].

Not all mutations have equal influence on the disease state of a cell, however. Certain mutations, called "drivers," are known to have a causative effect, driving the transformation of a cell from healthy to cancerous, often by promoting cell growth or inhibiting apoptosis (programmed cell death) [16]. In contrast, the majority of the mutations do not significantly affect the cancer characteristics of a cell, and can be considered relatively benign "passengers" in a tumor cell [21]. Mutated driver genes are worthwhile targets for drug discovery, because counteracting the mutation's effects can potentially slow or reverse cancer progression in individual patients [22,23]. To fully realize this potential, however, there is a need to develop computational approaches that can (i) distinguish causative from non-causative mutations, and (ii) identify key causative mutations for experimental studies and clinical targeting.

In this research, we consider a subset of the problem: non-synonymous single nucleotide variants (nsSNVs), or point mutations, in protein kinase genes. These mutations are reflected in the amino-acid sequence of protein kinases, a class of enzymes frequently implicated as oncogenes and for which several targeted drug therapies have already been successfully introduced [40]. In general, a protein kinase acts as a "molecular switch" in the cell, responding to a stimulus and relaying the signal through the cell. Changes in the amino acid sequence of a protein kinase may alter the structure in critical ways - for example, certain point mutations in epidermal growth factor receptor (EGFR) can lock the protein into an active state, generating signals for the cell to grow even in absence of the stimulus, epidermal growth factor (EGF). This uncontrolled growth can lead to cancer.

Indeed, several previous studies have proposed computational methods to identify causative mutations in cancer genomes. These methods fall into three major categories: (i) statistically based methods, (ii) structure-based methods and (iii) machine learning methods. Statistically based approaches are based on the assumption that mutations that occur in multiple patient samples are likely to be causative [24–26]. Although such assumptions are valid for some recurring mutations such as the L858R mutation in EGFR [27], there is emerging evidence that rare mutations can be drivers [21,28–30]. Moreover, a comprehensive analysis of several breast and colorectal cancer genomes revealed that the genomic landscapes of those cancers are dominated by a large number of rare gene mutations rather than recurrent oncogene mutations [31].

Structure-based methods offer a powerful way of predicting the impact of mutations by taking into account the three-dimensional context of the mutated residues [32–35]. However, such approaches are not applicable on a genome-wide scale because of the lack of crystal structure information for several oncogenes.

In the past decade, machine learning has been applied to various biological problems [36–40]. The nature of machine learning fits many problems in the biological domain, especially in where there are large amounts of multi-dimensional information available which are infeasible to analyze manually because a wide variety of factors play into the oncogenic effect of any given mutation. The large amount of clinical information now available in public databases allows us to apply a machine learning approach to this problem to efficiently prioritize candidate mutations for their likely impact on disease. Machine learning approaches have become a method of choice to predict causative mutations based on a variety of contextual information [32,41–49]. In general, supervised learning approaches learn from the features of known causative and non-causative mutations to classify unknown mutations, while unsupervised learning approaches cluster mutations into different groups based on the proposed features. Although several machine learning-based methods have been proposed previously, there still remains a need to improve the sensitivity and efficacy of existing methods [50]. First, most existing approaches use standard features of mutated residues for training the classifier and do not take into account gene- or family-specific features that can improve prediction accuracy [47]. Second, existing approaches use one or two common machine learning algorithms and do not consider the biases introduced by these algorithms. Third, most existing approaches provide a binary "yes" or "no" classification of causativeness, which does not solve the problem of prioritizing candidate mutations for follow-up experimental studies. Fourth, unsupervised learning approaches have never been utilized in previous studies, in where we believe unsupervised learning algorithms can strengthen the confidence of the prediction.

In this research, we apply a combined supervised and unsupervised machine learning approach to predict and prioritize causative mutations in protein kinases, a class of proto-oncogenes frequently mutated in human cancers. Our approach differs in some major ways from previous approaches, the summary of novel contributions of this part of the research are as follows:

- We introduce new kinase-specific features, beyond those used in previous methods [47, 51, 52], to improve prediction accuracy.

- We combine supervised and unsupervised learning to provide a reliable scheme for the prioritization of a set of "unconfirmed" causative mutations. The supervised component combines multiple supervised learning algorithms (individual classifiers), overcomes the biases introduced by each method. The unsupervised component strengthens the confidence of our prioritization.

- We use our machine learning framework to produce a numerical ranking of causative mutations in EGFR and test the impact of predicted mutations on EGFR kinase activity using cell-based assays.

- Our studies identify T725M as a rare causative mutation inasmuch as the T725M mutation increases EGFR autophosphorylation and displays catalytic activity in the absence of the activating EGF ligand.

3.2 Background

Before the past decade, our knowledge of human gene variation remained rudimentary. In 1999, Michele Cargill et al [53] conducted a systematic survey of SNPs in the coding regions of human genes, this survey provided a fundamental description of sequence variation in the coding regions of human genes. This study provided thorough insight on SNPs characterization. Based on this study, a lot of computational models have been introduced to accelerate the research of SNPs classification and the prediction of causative SNPs.

Early Probabilistic Models In Cancer Research

In the very beginning of the past decade, many studies considered how to identify pathogenic and tumor-derived mutations in genes, and some of them specifically focused on how to distinguish deleterious and neutral nsSNPs. In turn, several corresponding probabilistic models have been built. Some of the significant works are introduced in the following. Wang and Moult [54] provided a set of empirical rules to predict deleterious SNPs, and showed that most of the detrimental nsSNPs affect protein function indirectly through effects on protein structural stability (i.e. disruption to the protein hydrophobic core). Based on this work, some other studies [42, 55–58] have proposed the importance of considering protein structure as a useful feature for their prediction models, which depend on mapping SNPs to positions in homologous 3D protein structures, as well as using information from multiple sequence alignments.

These studies are the earliest computational models that are heavily referenced by researchers in this domain. Although machine learning methods have been heavily applied into the research of identification of cancer-causing mutations lately, studies based on statistical/probabilistic models are still being introduced.

Early Machine Models In Cancer Research

Although some previous studies used simple machine learning techniques (rule based systems and simple classification models) to predict the impact of SNPs, it seems that [59] was the first to actually apply solid machine learning algorithms onto the SNP prediction problem. Based on the idea that automated learning from training data is an attractive alternative to manual tuning of empirical rules, Krishana and Westhead [59] applied two machine learning methods, namely Decision Trees [60, 61] and Support Vector Machines (SVM) [62, 63], to predict the functional effects of non-synonymous SNPs in protein coding regions of the genome. The authors also tried to optimize the set of features by gradually adding the above features into their feature set, thus forming a manual wrapper approach for feature selection. Since the study of [59] only predicts the effects of SNPs on protein function, but did not have it tested on human mutations or SNPs (deleterious or neutral), Cai et al. [64] purposed a Bayesian approach [65] to discover pathogenic SNPs in conserved protein domains. Cai claims that their study is the first to use a Bayesian approach to predict pathogenic SNPs and test it on SNPs in human genes in relation to human diseases. The reason why they used a Bayesian approach was because they thought that our knowledge (at that time) of human SNPs in relation to diseases was not yet to the point at which large-scale supervised learning can be directly applied.

Prevalence of Machine Learning Models

Since 2005, the number of publications utilizing machine learning techniques to study the problems in the domain of cancer prediction (specifically for nsSNPs classification) has been extensively increasing. It is obvious that researchers began to realize how powerful the machine learning techniques could help to accelerate their research.

Karchin et al. developed the very first version of CanPredict [66], namely LS-SNP, which is a genomic scale software pipeline to annotate nsSNPs. This system combines a rule-based approach to identify putatively destabilizing nsSNPs and a SVM classifier to identify nsSNPs likely to have an impact on human health. The authors used information gain, and the 1R algorithm [67] (with 10-fold cross-validation and bucket size 14) to rank all attributes, and both methods showed that the degree of sequence conservation (PSIC) at the nsSNP position is the most important attribute. The optimized subset of attributes for each attribute set at each level of imbalance is presented with the wrapper-based feature selection method. Moreover, because of the big difference between the size of the neutral dataset and the diseased dataset, the authors also assessed the effect of resampling and weighting on the prediction performance, and their results show that the best prediction comes from the balanced data set that included all attributes.

The use of the Support Vector Machine (SVM) algorithm has also been raised along with its prevalence in other research domains. Yue and Moult [32] proposed the SNPs3D, a system that utilizes SVM to identify deleterious human SNPs. Torkamani et al [46] used support vector machine (SVM) to predict the disease association (causative or non-causatie) of nsSNPs in the human kinase gene family. They also explored two other classifiers namely neural network (Multilayer Perceptron), and Decision Table, but they were ultimately discarded in favor of the SVM. However, the authors did not explain or provide clear evidence to support their choice of selecting SVM as the classifier. Second, the authors failed to prove the robustness of their feature set as there was no feature selection method implemented; the only explanation that can be inferred from this is that the features they used were inherited from other previous studies that also have no feature selection methods applied. In a follow-up study [47], Torkamani et al. applied the same method to protein kinases and not only announced that their method could identify known drivers with high accuracy but also claimed that their predicted driver mutations are under positive selection with very high confidence. Later, Torkamani et al. [48] tried to identify the rare cancer driver mutations by network reconstruction.

Karchin et al. developed an upgraded version of CanPredict [45]. The CanPredict trains a Random Forest [68] to discriminate between mutations from the COSMIC cancer somatic mutation database (variants in those genes most likely to be involved in oncogenesis) [69] and nsSNPs from dbSNP database [70] with high MAFs of ¿ 20% (Minor Allele Frequencies; Likely passenger) [71].

Having seen various studies of using SVM as a tool for the prediction of deleterious nsSNPs, some researchers proposed to use some other different machine learning algorithms to tackle this problem afterwards. Bromberg and Rost [72] proposed SNAP, a standard feed-forward neural network-based model [2] for the prediction of the functional effects (effect or no effect) of nsSNPs. Their model needs only sequence information as input, but the authors also state that functional and structural annotations would help to improve the performance.

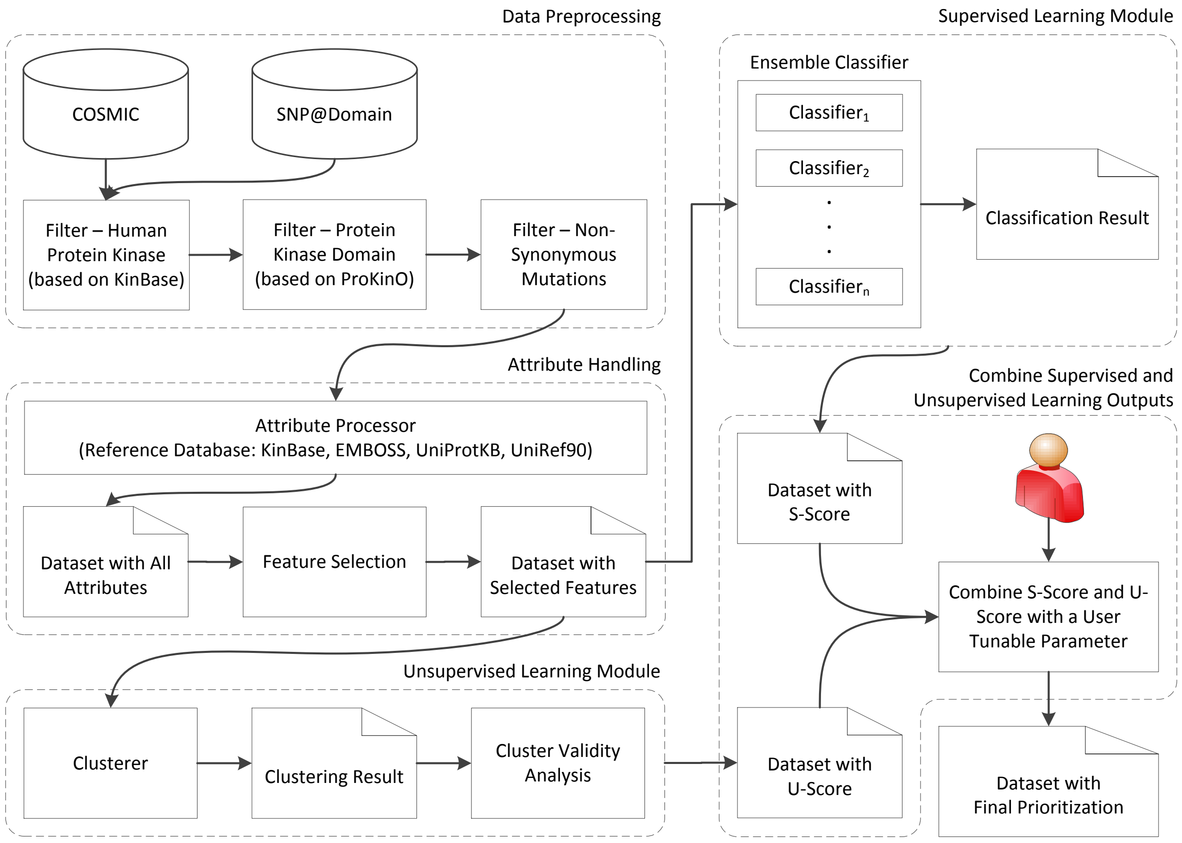
Carter et al. [73] developed a computational method called Cancer-specific High-throughput Annotation of Somatic Mutations (CHASM), to identify and prioritize those missense mutations most likely to generate functional changes that enhance tumor cell proliferation. They compared the performance between Random Forest and SVM, and showed that the random forest method proved superior. They also showed that their random forest based model outperformed the models obtained by other methods, including PolyPhen [42], SIFT [57], CanPredict [45], and KinaseSVM [46]. It is good to see that the authors did not surrender to the prevalence of SVM but provide solid evidence to support the choice of using Random Forest.

Adzhubei et al. introduced PolyPhen-2 [74], which is an upgrade to PolyPhen [42]. PolyPhen-2 is different from PolyPhen in the set of predictive features, the alignment pipeline and the method of classification. The authors used wrapper methods with both forward and backward elimination to evaluate all 32 features, and then 11 features were selected by both methods. Finally, a Naive Bayes classifier was utilized to perform prediction. The authors claimed that PolyPhen-2 was consistently superior compared to PolyPhen, and it was also competitive to three other popular prediction tools, namely SIFT, SNAP, and SNPs3D.

Based on the success of previous studies (i.e. CHASM, CanPredict, SIFT, PFAM, etc.), some researchers recently tried to analyze if there is still any room for the improvement in this domain by utilizing different approaches such as proposing some novel features, as well as utilizing machine learning techniques with the output of previous computational models.

Masso and Vaisamn [75] developed a predictive model with Random Forest algorithm to classify neutral and disease-associated nsSNPs. But their model actually achieved 76% cross-validation accuracy, which is average. Shi and Moult [33] used SNPs3D to analyse the structural and functional impact of driver mutations. Capriotti and Altman [76] proposed another SVM approach to identify the cancer-causing missense variants. Their method achieved 93% overall accuracy, but caution should be taken as they use 100% balanced dataset in where the number of driver mutations is the same as the number of passenger mutations. This would cause the improvement of the accuracy, which has been showed by Dobson et al. [77].

Previous studies that applied machine learning techniques to classify causative and non-causative mutations (or distinguish deleterious and neutral nsSNPs) share a common drawback: They did not provide clear evidence or comparison for the performance between different machine learning methods or different feature selection methods, they applied a couple of famous (at that time) machine learning algorithms and sometimes a single feature selection approach to build the models. Therefore, we proposed ROMP to address all these issues. ROMP combines the results of 5 feature selection methods to select the best feature subset, constructs an ensemble classifier with 11 individual classifiers to predict oncogenic mutations, uses EM clustering algorithm and our self-invented clustering validity metrics to reduce the label uncertainty, finally combines the supervised and unsupervised components



Data Preprocessing
COSMIC
SNP@Domain
Filter – Human Protein Kinase (based on KinBase)
Filter – Protein Kinase Domain (based on ProKinO)
Filter – Non-Synonymous Mutations
Supervised Learning Module
Ensemble Classifier
Classifier$_1$
Classifier$_2$
Classifier$_n$
Classification Result
Attribute Handling
Attribute Processor (Reference Database: KinBase, EMBOSS, UniProtKB, UniRef90)
Dataset with All Attributes
Feature Selection
Dataset with Selected Features
Combine Supervised and Unsupervised Learning Outputs
Dataset with S-Score
Combine S-Score and U-Score with a User Tunable Parameter
Unsupervised Learning Module
Clusterer
Clustering Result
Cluster Validity Analysis
Dataset with U-Score
Dataset with Final Prioritization

Figure 3.1: ROMP System Overview

to deliver a robust prediction result as well as to identify instances with suspicious labels in the dataset.

3.3 System Overview

Figure 3.1 provides a high-level overview of ROMP. We now provide a brief description of ROMP's components shown in Figure 3.1, and more details will be given in subsequent sections.

Data Preprocessing Given a large dataset $\mathcal{A}$ which contains oncogenic mutations (causative) as well as healthy mutations (non-causative), we first filter out the useless samples for this research by referencing certain published databases. After filtering, this step produces a dataset $\mathcal{M}$ which includes causative mutations, non-causative mutations, and putative mutations as the input for our system.

Attribute Processor With dataset $\mathcal{M}$, we first extract the attributes that were used in previous literature, and the novel attributes proposed by our collaborator (Biologist). Followed by utilizing multiple feature selection methods to identify the useful features. This step produces a training dataset $\mathcal{MSF}_{train}$ and a testing dataset $\mathcal{MSF}_{test}$. The combination of these two datasets are basically identical to $\mathcal{M}$ but have the useless attributes removed.

Supervised Learning Module ROMP uses 11 supervised learning algorithms (classifiers) to learning and predict the oncogenicity of each mutations in $\mathcal{MSF}_{train}$ and $\mathcal{MSF}_{test}$. All 11 classifiers are trained individually, and then we effectively combine them together to construct a more robust ensemble classifier. This step assigns each mutation an $S - Score$, which is the possibility of a mutation to be causative.

Unsupervised Learning Module In the domain of Biology, there is nothing 100% certain. Independent from the supervised learning module, we use unsupervised learning algorithm and our self-invented cluster validity metrics to reduce the possible uncertainty. Each mutation in $\mathcal{MSF}_{train}$ and $\mathcal{MSF}_{test}$ is clustered by a clustering algorithm, and then our self-invented metrics will measure the level of oncogenicity of each mutation. This step assigns each mutation a $U - Score$, which is the possibility of a mutation to be causative.

Combining Supervised and Unsupervised Learning Outputs Finally, ROMP combines the $S - Score$ generated by the supervised learning module and the $U - Score$ generated by the unsupervised learning module to assign each mutation a final prioritization score. The combination rate of these two scores is user tunable. This step considers both supervised and unsupervised learning outputs, and produces a prioritization list of all putative mutations with high confidence.

3.4 Evaluation

Before introducing the details of each component shown in Figure 3.1, we first present the performance evaluation of our framework with the real-world datasets.

Datasets for Evaluation

To evaluate our framework, we conducted experiments on benchmark datasets from the UCI Machine Learning Repository [78]. We selected 4 datasets namely Tic-Tac-Toe Dataset [79], Wisconsin Breast Cancer (Original) Dataset [80], Wisconsin Breast Cancer (Diagnostic) Dataset [81], and Blood Transfusion Service Center Dataset [82]. Table 3.1 shows a brief description of the datasets used for our evaluation. The associated tasks for all three selected datasets are binary classification, but actually there is no limitation for ROMP to be used in problems with multi-class datasets. The types of attributes are different across datasets; Tic-Tac-Toe dataset has categorical attributes; WBC-Original has integer attributes; and WBC-Diagnostic and Transfusion have real valued attributes. Moreover, Some attribute values are missing in the WBC-Original dataset. It is worth mentioning that we have renamed the classes in all these datasets to "Positive" (P) and "Negative" (N) for clarity.

Table 3.1: Summary of the UCI Datasets Used in Evaluation

Dataset	Tic-Tac-Toe	WBC-Original	WBC-Diagnostic	Transfusion
Number of Instances	958	699	569	748
Number of Attributes	9	10	32	5
Dataset Characteristics	Multivariate	Multivariate	Multivariate	Multivariate
Attribute Characteristics	Categorical	Integer	Real	Real
Associated Tasks	Classification	Classification	Classification	Classification
Missing Values	No	Yes	No	No

Performance of ROMP

Table 3.2: Performance of ROMP with UCI Datasets

Dataset	TP Rate	FP Rate	Accuracy(%)	F-Measure	S-P	S-N
Tic-Tac-Toe	1	0	100	1	0	0
WBC-Original	0.9959	0.0209	98.33	0.9677	2	19
WBC-Diagnostic	0.9764	0.0112	98.42	0.9787	8	11
Transfusion	0.4663	0.0807	76.20	0.5407	151	285

Table 3.2 illustrates the overall performance of ROMP with the selected UCI datasets. As the table shows, ROMP performed fairly well on Tic-Tac-Toe, WBC-Original, and WBC-Diagnostic datasets, reaching at least 0.9764 TP rate and at most 0.0209 FP rate. It even performs perfectly on the Tic-Tac-Toe dataset by classifying all instances with 100% accuracy. For the messy dataset – Transfusion, ROMP performs competitive to the best result that has been reported in literature [83]. ROMP is not only a robust predictor that effectively combines supervised and unsupervised learning methods, it is also able to identify suspicious instances (outliers) through the learning process. The last two columns of the table describe the number of "Suspicious Positive" and "Suspicious Negative" labeled instances identified by ROMP in the corresponding dataset (see Section 3.9 for details).

Several algorithms have been suggested in literature which claim high accuracy on these benchmark datasets. We now present the comparison between ROMP and the reported top results of the datasets in the literature.

Tic-Tac-Toe Dataset

This dataset shows complete set of all possible end of game board configurations of a tic-tac-toe game. Table 3.3 shows the comparison of ROMP with 3 other algorithms that reported with top accuracies. Here we compared our accuracy with some of the latest methods suggested in the literature. We compared our results with " [84], [85] and [86]. As the table shows, ROMP provides the best result among all, in terms of reaching 100% accuracy.

Table 3.3: Performance of ROMP with Tic-Tac-Toe Dataset

Reference	Method	Accuracy (%)	F-Measure
[84]	IB3-CI	99.1	n/a
[85]	CI3	98.4	n/a
[86]	Cluster Based Classification	99.2	n/a
[87]	ROMP	**100**	**1**

Wisconsin Breast Cancer (Original) Dataset

WBC-Original dataset consists of features computed from a digitalized image of a fine needle aspirate (FNA) of a breast mass. These features describe the properties of the cell nuclei present in the image.

Table 3.4 illustrates the comparison of the Fuzzy c-means clustering method with ROMP. Muhic [88] suggested the Fuzzy c-means clustering method for learning WBC-Original dataset. The method showed 100% TP rate, and 13% FP rate. Evidently the Fuzzy c-means had better TP rate but worse FP rate compared to ROMP (TP rate = 0.9959, and FP rate = 0.0209). ROMP performs 6.91% better than the Fuzzy c-means clustering method in terms of accuracy, and 7.84% better in F-Measure.

Table 3.4: Performance of ROMP with WBC-Original Dataset

Reference	Method	Accuracy (%)	F-Measure
[88]	Fuzzy c-means clustering	91.42	0.8893
[87]	ROMP	**98.33**	**0.9677**

Wisconsin Breast Cancer (Diagnostic) Dataset

WBC-Diagnostic dataset consists of features computed from a digitalized image of a fine needle aspirate (FNA) of a breast mass. These features describe the properties of the cell nuclei present in the image. Each cell nucleus is defined using 10 traits namely - radius, texture, perimeter, area, smoothness, compactness, concavity, concave-point, symmetry and fractal dimension. Furthermore, for each trait mean, standard error and worst value is computed. For our processing we removed the unique ID value given to row in dataset.

Table 3.5 shows the comparison between ROMP and 2 other studies that reported with top accuracies. We compared ROMP with the methods suggested by [89] and [90]. As the table shows, ROMP outperforms the results of the method suggested by [89] by 1.25% in accuracy and 2.53% in F-Measure. The method suggested by [90] produced the best result, 1.4% and 1.65% better than ROMP in accuracy and F-Measure respectively. However, the results reported by [90] is with feature selection, but ROMP includes the raw dataset with all the features.

Table 3.5: Performance of ROMP with WBC-Diagnostic Dataset

Reference	Method	Accuracy (%)	F-Measure
[89]	Fuzzy k-nearest neighbor	97.17	0.9534
[90]	Random Forest with feature selection	**99.82**	**0.9952**
[87]	ROMP	98.42	0.9787

Blood Transfusion Service Center Dataset

This is a donor database of Blood Transfusion Service Center from Hsin-Chu City in Taiwan. It was developed to demonstrate the RFMTC marketing model (a modified version of RFM). To gather blood donated, the service center passed their blood transfusion service bus to one university in Hsin-Chu City in about every three months. 748 donors were selected at random from the donor database.

Table 3.6 shows the comparison between ROMP and the best result reported in the latest literature. [83] suggests an improved method of over-sampling dataset SMOTE (originally suggested by [91]). SMOTE is a minority protecting oversampling method. As the table shows, ROMP outperforms the results of the method suggested by [83] by 0.6% in accuracy and 3.98% in F-Measure.

Table 3.6: Performance of ROMP with WBC-Diagnostic Dataset

Reference	Method	Accuracy (%)	F-Measure
[91]	SVM	80.48	0.4710
[91]	SVM with oversampling method	65.11	0.5010
[87]	ROMP	**81.15**	**0.5407**

Summary

We compared ROMP with 4 real-world datasets obtained from UCI Machine Learning Repository. ROMP outperformed all other reported top results in the Tic-Tac-Toe Dataset, the Wisconsin Breast Cancer (Original) Dataset, and the Blood Transfusion Service Center Dataset, but performed slightly wrose than the method reported by Nguyen et al. [90] in the Wisconsin Breast Cancer (Diagnostic) Dataset. However, the results reported by [90] are with feature selection, but ROMP used the raw dataset with all the features. Furthermore, the difference is subtle. Based on above results, we proved that ROMP is a high performance predictor with fairly good performance on different datasets. It effectively combines supervised and unsupervised learning methods to generate robust prediction. Moreover, it is also able to identify suspicious instances (outlier) through the learning process, such information can provide useful information to the domain experts for further analysis, such as to refine the data samples, to identify useful features, and to perform detailed study on the suspicious objects. More details about ROMP and its application on a complex domain – cancer research – are presented in the following sections.

3.5 Data Preprocessing

Data Sources

The data sources used for training and evaluating the classifiers consist of a “positive” set of known-causative mutations, a “negative” set of known-benign mutations, and several attributes of proteins and amino acids, which we drew from several databases.

Positive mutation set: Oncogenic Mutations

As the basis for our “positive” set we used the Catalogue of Somatic Mutations in Cancer (COSMIC), a database of somatic mutations observed in cancer samples gathered from published literature sources and provided by the Wellcome Trust Sanger Institute’s Cancer Genome Project (CGP) [17, 92]. We use two versions of the COSMIC (versions 50 and 57) in this research as the version 57 only became available at the last stage of our research, as an updated dataset. The mutations from the COSMIC database (versions 50 and 57) were each filtered by gene name for human protein kinases as identified in KinBase [93]. We then filtered these mutations for those occurring in the protein kinase domain, as identified by ProKinO [94], yielding 922 distinct point mutations in 152 protein kinase genes (v50) and 1451 distinct point mutations in 225 protein kinase genes (v57). Each kinase gene was therefore associated with several mutations; EGFR kinase, for example, has 278 mutations in the protein kinase domain (v57).

As COSMIC is a catalogue of somatic mutations observed in clinical samples, it potentially contains both causative and non-causative mutations. We selected a subset of mutations which have been observed more than once in COSMIC, as an initial list of 67 (v50) and 226 (v57) likely causative mutations to use as the "positive" (disease) set in our classification.

Negative mutation set: Benign Polymophisms

The "negative" (benign, non-disease) set consisted of the 331 non-synonymous mutations in protein kinase genes obtained from SNP@Domain, an online database of naturally occurring single nucleotide polymorphisms (SNPs) within protein domain structures and sequences [95], which was itself originally derived from dbSNP [70]. We consider commonly-occurring polymorphisms to be effectively benign because the polymorphisms were originally sampled from healthy individuals, an approach previously used by others (e.g. [45]).

Protein Features

Features of the mutations used for training the classifiers were retrieved from several sources. The hierarchical classification of protein kinases into major groups and families is according to KinBase [93]. Amino acid properties, such as hydrophobicity and molecular weight, were taken as listed in the data files included with EMBOSS [96]. Protein sequence annotations were retrieved from the UniProtKB database [97]. We also used protein sequences from the UniRef90 database [98] in our own calculations of amino acid conservation, position of the kinase domain, and subdomain locations (refer to Feature Preprocessing).

Out of the 518 protein kinases in the human genome, several are "atypical" kinases which lack sequence homology to the other kinases. Thus, certain features could not be calculated for the atypical kinases, leaving 503 distinct protein kinase genes for which we successfully extracted features for machine learning.

Data Partition and Definitions

Since COSMIC is a compilation of many studies, we assume that mutations which appear more than once in the dataset are more likely to be "drivers" (causative) than those which appear only once in the table. Furthermore, objects that have been observed less frequently might possess weaker characteristics than those observed more frequently. Thus we consider the COSMIC entries that have only been reported once as unconfirmed drivers, or potential passengers, providing a useful application for our method in prioritizing the unconfirmed drivers for further study.

Our benign set (non-disease set) is composed by the non-synonymous mutations obtained from SNP@Domain. For the positive set (disease set), we separated the COSMIC data set into two subsets, with the assumption that some COSMIC mutations observed only once might be false positives. Certain definitions of our dataset that appear later in this dissertation are described below.

- **COSMIC-FG1** (Exp. I.2 and III.2): The causative set of mutations in COSMIC that are observed in more than one distinct sample (**F**requency **G**reater than **1**). The non-causative set is the non-synonymous mutations obtained from SNP@Domain.
- **COSMIC-ALL** (Exp. I.1 and III.1): The causative set consists of all nonsynonymous COSMIC mutations that appear in the protein kinase domain, i.e. omitting the filter applied in COSMIC FG1. The non-causative set is the non-synonymous mutations obtained from SNP@Domain.
- **COSMIC-FE1**: The "unconfirmed" set of mutations in COSMIC that are observed only once. (**F**requency **E**qual to **1**).
- **Experiment I, II, III**: Designations for specific input training sets and filters used to train the classifiers and perform computational experiments.

Experimental settings, inputs and associated statistics are described for COSMIC v.50 in Tables 3.8 and 3.9, and for COSMIC v.57 in Tables 3.19, 3.20 and 3.18. More experimental settings for both COSMIC v.50 and v.57 correspond to Experiment III.

According to Table 3.9 , it is clear that the dataset of COSMIC FG1 (Exp. I.2) is highly imbalanced: the number of instance of the non-disease set is almost 5 times more than the disease set. Since highly imbalanced datasets often lead to inferior classification results [15], the handling of these imbalanced datasets becomes a critical issue before performing any classification. Re-sampling, a common approach to handle the problem of imbalanced data, would not work in our case - whether oversampling the minority class or under-sampling the majority - since every single instance in our dataset is one unique mutation. For this reason, on top of the classifiers we utilized, we apply the Cost Sensitive Classifier for defining the cost matrix according to the imbalance ratio of the dataset. Since the minority classes in our experiments are the positive set (disease set), we define the False Negative cost in the cost matrix as the ratio between the majority class (benign set) and minority class (disease set). However, the Cost Sensitive Classifier is no longer necessary for COSMIC v57 as the positive set and negative set became more balanced, as Table 3.18 illustrates.

3.6 Attribute Handeling

Attribute Preprocessing

Mutations are uniquely identified by the gene name, protein sequence position, wild-type amino acid, and mutant amino acid type. We extract features related to biochemical, structural, functional and evolutionary properties, which in the end generated 29 features in total, as follows.

Of these 29 features, 25 were previously explored by others, including amino acid biochemical properties, sequence conservation, and kinase subdomain [44, 46, 77, 99]. Our novel features are the protein kinase classification terms (group and family) and the conservation levels within kinase group and family.

Comparative

The protein kinase superfamily constitutes a diverse range of enzymes which can be classified hierarchically into 8 major groups (plus the "other" and "atypical" categories) and multiple families within each group, each with distinct mechanisms and interacting partners [100]. To account for these potential differences, we tracked the names of each kinase's group and family, per KinBase. Additionally, we calculated the evolutionary conservation of the wild-type or consensus amino acid type among diverse eukaryotic species at each position within the protein kinase domain. This approach considers the evolutionary process as "nature's laboratory": over the course of over 2 billion years since the divergence of eukaryotes, each type of mutation at each position in the protein kinase gene has occurred many times, simply by chance as part of a random mutational process. The mutations that do not alter the overall functioning of the protein are tolerated, and will be observed in extant species, while more drastic or deleterious mutations will be quickly eliminated from the gene pool of surviving organisms.

Since the association between evolutionary conservation and disease-causing mutations in protein kinases is not yet fully understood, we calculated conservation at three evolutionary depths: within the same PK family (close evolutionary relatives), within the same PK major group (greater evolutionary distance), and among all eukaryotic protein kinases (ancient divergence, preserving only the overall fold of the protein structure). While conservation of amino acid types has been considered in previous studies, our distinction between depths of divergence at the family and group levels is novel. In addition, we calculate conservation relative to both the wild-type residue and the "consensus" residue type in an alignment. The conservation value at a given alignment site is the frequency of the wild-type or consensus-type residue in the alignment column. The calculations are performed on large multiple-sequence alignments using sequences from UniRef90 and aligned using the MAPGAPS program [101] and HMMer 3.0 [102].

Amino Acid Properties

These features describe the properties of individual amino acids in the sequence of both the healthy, normally occurring "wild-type" protein and the mutant, as well as the difference between the two where this can be quantified. The properties are the hydropathy index (hydrophobicity), charge, polarity, van der Waals volume, molecular mass, and naturally occurring residue frequency. We also include the BLOSUM62 amino acid substitution matrix values [103] as a measure of the overall similarity between the wild-type and mutant residues.

Structural and Functional

We introduced features that captured the structural and functional location of mutations. The eukaryotic protein kinases can be identified by a set of 11 structural regions, known as subdomains, which are conserved across the entire superfamily [104]. We identified the sequence locations of these subdomains in each of the 503 typical protein kinases in the human genome, using a Gibbs motif sampler with curated motif models for each subdomain, similar to the method described in [47]. Mapping the position of each mutation onto these regions allowed us to identify which sub-domain the mutation belongs to, if any. Two additional features describe functional roles of amino acids: whether a position is a binding site (e.g. for ATP or another protein), and the post-translational modification of the residue, if any (e.g. phosphorylation).

Feature Selection

Feature selection serves two purposes: to choose a smaller, more computationally tractable subset of meaningful features which can be used to effectively predict the target attribute (causative or non-causative), and to understand the relative usefulness or contribution of each feature toward predicting the target [105]. With emphasis on the latter, we independently applied five feature selection algorithms to evaluate our attributes, as following:

- OneR algorithm [106], with a minimum bucket size of 14. This algorithm evaluates the contribution of an attribute by using the minimum-error attribute for prediction. It requires discretization if the values of an attribute are numeric. It generates a single level decision tree (a set of rules) which tests one particular attribute each time.

- Relief-based selection [107], with 10 nearest neighbors for attribute estimation. This algorithm is an instance-based algorithm which evaluates the worth of an attribute by repeatedly sampling N instances and considering the value of the N nearest instance of the same and different classes.

- Chi-Square selection. This algorithm evaluates the worth of an attribute by computing the value of the chi-squared statistic with respect to the class.

- Gain-ratio-based filter approach [67], using the Gain Ratio Attribute Evaluator and Spread Subsample method. Gain Ratio evaluates the worth of an attribute by measuring the gain ratio with respect to the class.

- Correlation-based selection [12], with Greedy (forward) searching algorithm. This approach performs a greedy forward search through the space of attribute subsets. Starting with no attributes in the space, and stoping when the addition of any remaining attributes results in a decrease in evaluation. The main idea of correlation-based selection algorithm is to find a feature subset which includes features that are highly correlated to the target attribute and have very low correlation to other features.

Selection Methodology

All five feature selection methods were each applied separately on both the COSMIC-All and COSMIC-FG1 datasets, attributes were ranked in terms of effectiveness as a predictor according to each selection method. We then combined the selected features through a majority voting approach in where each of the selection algorithms casts one vote indicating whether a given attribute is selected or not.

We selected the attributes selected by at least 3 out of 5 (60%) of the selection methods, yielding a ranked list of 17 selected features (from the original 29) as our final feature subset which we used as input for the training process in subsequent analysis. We also determined a ranking indicating the usefulness of each selected feature by taking the arithmetic mean of the feature's ordinal ranking across all 5 selection algorithms. This "average rank" enables a complete ranking of the selected features. The results of feature selection on both these datasets with 10-fold cross-validation are shown side-by-side in Table3.7.

Selection Results

According to Table3.7, about 88% (15 out of 17) of the selected features are identical in both the COSMIC-All and COSMIC-FG1 datasets, with "average rank" from 1.4 to 13 with the COSMIC-All dataset, and from 1.2 to 16.25 with the COSMIC-FG1 dataset. Finally, we decided to use the feature subset that based on COSMIC-All dataset as our final feature subset for the following experiments. The justification is that this evaluation was performed on the combined positive and negative data sets, with the positive set also including COSMIC mutations that occur only once, this ensure that the feature selection process is performed on a larger data set to retain the generality.

Table 3.7: Selected features based on v.57 COSMIC All and FG1 datasets.

Selected Feature	Votes		Avg Rank	
	All	FG1	All	FG1
Protein kinase family	5	5	1.40	1.20
Protein kinase group	5	4	1.80	2.25
Amino acid type, WT	5	5	8.00	7.80
BLOSUM62 pairwise score	5	4	8.20	8.25
Side-chain polarity, mutant	5	3	11.00	11.67
Conservation of wild type in all kinases	5	4	11.60	6.00
Conservation of consensus type in kinase group	5	4	11.60	10.50
Conservation of consensus type in all kinases	5	4	13.00	13.00
Conservation of consensus type in kinase family	4	5	5.75	4.60
Kinase subdomain	4	5	6.00	3.60
Average mass of amino acid, WT	4	5	7.50	7.80
Is a binding site?	4	—	8.25	—
Van der Waals volume, WT	4	5	8.75	9.80
Site modification type (if any)	4	3	9.25	10.67
Amino acid type, mutant	4	4	10.75	10.00
Side-chain polarity, WT	4	4	11.50	13.25
Is in protein kinase domain?	3	4	11.67	16.25
*Isoelectric point, WT	—	4	—	11.00

* = Not included in our final feature subset. The "Votes" column indicates how many feature selection algorithms cast a vote for that particular feature during the 10-fold cross-validation selecting procedure; the "Avg Rank" column describes the averaged rank of a particular feature within the selected algorithms. The feature "binding site" was selected with COSMIC-All but not COSMIC-FG1, and "Isoelectric point, WT" was selected with COSMIC-FG1 but not COSMIC-All.

We first sought to determine a subset of features that show high predictive value in distinguishing causative from non-causative mutations, and to evaluate the contribution of the kinase-specific features we introduced in this study, namely the hierarchical protein kinase classification levels (group, family) and the conservation levels at each evolutionary depth (all kinases, group and family).

Table3.7 shows that all 5 selection algorithms selected the features "Protein kinase family" and “Protein kinase group”, and each individual algorithm ranked these two features at the top. The features “amino acid type (WT)”, “BLOSUM62 score”, and “side chain polarity (Mutant)” were also selected by all 5 algorithms and ranked highly by individual algorithms.

Conservation scores of the wild-type residue among all kinases, and of the alignment consensus type among all kinases, among the major groups and among major families, were also ranked highly, indicating that they extensively contribute to the prediction of the target attribute, a result that supports the importance of our novel proposed features.

Furthermore, other features that have been identified as important characteristics of deleterious mutations in previous studies, namely “Sub domain”, “Avg Mass”, “binding site”, “Van der Waals Volume Wild”, “modification”, “Mutant Type Amino Acid”, and “Side Chain Polarity Wild”, were also selected for inclusion. Finally, the binary feature “is PK domain” is the last feature that has been included into our final feature subset. The detailed feature selection records of the 5 selection methods with 10-fold cross-validation is given in Appendix A

3.7 Supervised Learning Module

In this section, we first introduce the supervised learning algorithms used in this research, followed by presenting the evaluation of individual learning algorithms based on COSMIC v50 dataset. Then we combine multiple classifiers to prioritize putative EGFR mutations in the same dataset. Afterwards, we update the results with the latest COSMIC dataset – v57.

Learning Methods

In order to enable the comprehensive comparative evaluation of classifiers found lacking in previous studies, we applied a variety of supervised learning algorithms to classify point mutations in human protein kinase sequences as either causative or non-causative.

Since a necessary prerequisite for an ensemble classifier to be more accurate than any of its individual members is for the classifiers to be accurate and diverse [108], based on the selected features, we applied 11 machine learning methods to our dataset, spanning all the basic schools of inductive learning including J48 (Tree) [60, 61], Random Forest [68], NB Tree [109], Functional Tree [110], Decision Table [111], DTNB [112], LWL (J48+KNN) [113], Bayes Net [65], Naive Bayes [114], SVM [62, 63], and Neural Network [2]. We chose these algorithms to cover the major categories of classifier algorithms: Trees, Rules, Instance-based, Functions, and Bayes. Due to the highly imbalanced dataset in certain experiments, a meta-learner, Cost Sensitive Classifier, is used to define the confusion matrix to optimize the learning process [115].

Cross-validation

To account for the possibility of over-fitting or bias resulting from small sample sizes, we applied 10-fold cross-validation [116, 117] to train all of the models presented in this research. Cross-validation is a robust approach to validate the training methods as it repeatedly tests different trained models on "unseen" data [116]. In 10-fold cross-validation, the training set is randomly split into 10 equal subsets (folds) and the algorithm is repeatedly trained on nine subsets and tested on the remaining subset, which is a set of "unseen" data. This ensures each instance is included into the testing set once. The final results are the average of the results of 10 independent training models.

Evaluation of Individual Classifiers Using COSMIC v50

Cost Sensitive Classifier

The settings for Experiment I are described in Table 3.8, and corresponding statistics of the dataset is illustrated in Table 3.9.

Table 3.8: Experiment Setting I - Based on COSMIC v50 Dataset.

Exp.#	Training Set	Testing Set (Prediction)
I.1	Mutations (Kinase domain) in COSMIC v50 dataset, excludes 177 EGFR mutations in the COSMIC v50 dataset whose frequency equal to 1	177 EGFR mutations in the COSMIC v50 dataset whose frequency equal to 1
I.2	Mutations (Kinase domain) in COSMIC v50 dataset whose frequency greater than 1	177 EGFR mutations in the COSMIC v50 dataset whose frequency equal to 1

Table 3.9: Training Dataset based on COSMIC v50.

Exp.#	Dataset	Causative	Non-Causative	Total
I.1	COSMIC-All	625	331	956
I.2	COSMIC-FG1	67	331	398

According to Table 3.9, it is clear that the dataset of COSMIC-FG1 (Exp. I.2) is highly imbalanced: the number of instance of the non-causative set is almost 5 times more than the causative set (67 vs. 331). Since highly imbalanced datasets can lead to inferior classification results [77], the handling of these imbalanced datasets becomes a critical issue before performing any classification. Re-sampling, a common approach to handle the problem of imbalanced data, would not work in our case — whether oversampling the minority class or under-sampling the majority — since every single instance in our dataset is one unique mutation. For this reason, on top of the classifiers we utilized, we applied the Cost Sensitive Classifier for defining the cost matrix according to the imbalance ratio of the dataset. We defined the False Negative cost in the cost matrix as the ratio between the majority class (in this case, the negative set, non-causative mutations) and the minority class (the positive set, causative mutations).

Comparison of COSMIC-All and COSMIC-FG1 Dataset

Table 3.10 shows the 10-fold cross-validation accuracy of the 11 individual classifiers trained and tested on the two datasets, COSMIC-FG1 and COSMIC-All, for models that were trained with all features and those trained with only selected features.

Table 3.10: Accuracy of 11 Trained Models – Experiment Setting I (COSMIC v50).

Algorithm	All Features		Selected Features	
	I.1	I.2	I.1	I.2
J48 (Tree)	84.4142	96.2312	84.6234	96.2312
Random Forest	83.1590	94.7236	84.6234	96.4824
NB Tree	82.3222	93.2161	84.1004	93.4673
Functional Tree	83.5774	96.4824	84.8326	96.4824
Decision Table	86.1925	88.1910	86.1925	88.191
DTNB	87.5523	94.4724	87.5523	93.9698
LWL(J48+KNN)	83.5774	94.4724	85.1464	95.4774
Bayes Net	84.8326	95.7286	84.9372	95.7286
Naive Bayes	83.5774	92.4623	83.5774	94.9749
SVM	87.3431	96.7337	87.1339	96.7337
Neural Network	83.4728	94.9749	83.7866	95.9799

In general, 10 out of 11 classifiers were more accurate when trained only with the selected features, with an average 0.59% improvement in experiment I.1 and 0.55% improvement in Experiment I.2. These results support the effectiveness of our selected features. In addition, the differences in accuracy between the 11 classifiers are very subtle within each dataset, though DTNB and SVM perform slightly better than others in Experiment I.1, while SVM and Functional Tree performs slightly better than others in Experiment I.2.

The performance of all these classifiers is lower on COSMIC-All than on COSMIC-FG1 dataset. One possible explanation for this is that COSMIC-All may contain some non-causative mutations in the positive set, blurring the distinction between the positive and negative sets during training.

On another note, although the accuracies of the models trained with the COSMIC-

FG1 dataset are about 10% higher than those with the COSMIC-All dataset, this result should be taken with some caution because the dataset COSMIC-FG1 is much smaller than COSMIC-All, and therefore the trained models of COSMIC-FG1 may be less generalizable than those of COSMIC-All. We therefore considered the possibility that the trained models in COSMIC-FG1 might not outperform the models of COSMIC-All when applied to new, unseen data. The procedure of 10-fold cross-validation is intended to address this concern by repeatedly reserving an "unseen" test dataset during the evaluation process. We also evaluated our assumption on the most recent COSMIC v57 dataset, which includes mutations that were not available in COSMIC v50 (see Section B).

Detailed Analysis with Alternative Measurement Indexes

We provide Table 3.11 and Table 3.12 to present the experimental results in terms of confusion matrix and several other major measurement indexes to better explain the insight of the performance of individual classifiers. These alternative measurements of classifier performance are more informative than simple accuracy, especially when classes in a dataset are highly imbalanced (i.e. instance counts in the positive and negative sets are far from equal, then the classifier may classifies all instances of the minority class into the class with majority instances).

We individually consider the classification results of COSMIC-FG1 dataset. Both table show that all 11 classifiers perform fairly well, with all TP (True Positive) rates ranging from 0.866 to 0.985 and FP (False Positive) rates ranging from 0.021 to 0.13. TP means a causative instance is actually classified as causative (+), TN means a non-causative instance is actually classified as non-causative (-). Table 3.12 shows that J48, Decision Table, and DTNB perform excellently in classifying the positive set (high TP and low FN), while SVM, Functional Tree, and Random Forest are the best in classifying the benign set (high TN and low FP). This information is not available if we only refer to the accuracy table (Table 3.10).

Table 3.11: Confusion Matrix – Classifiers Trained with Selected Features on COSMIC-FG1 v50 Dataset.

Algorithms	TP	FN	TN	FP
J48 (Tree)	66	1	317	14
Random Forest	62	5	322	9
NB Tree	60	7	312	19
Functional Tree	61	6	323	8
Decision Table	63	4	288	43
DTNB	63	4	311	20
LWL(J48+KNN)	62	5	318	13
Bayes Net	61	6	320	11
Naive Bayes	61	6	317	14
SVM	61	6	324	7
Neural Network	58	9	324	7

All *in silico* experiments were evaluated with 10-fold cross-validation. TP means an instance in the positive set (COSMIC) was correctly classified as causative, TN means an instance in the negative set (dbSNP) was correctly classified as non-causative.

Table 3.12: Detail Measurement – Classifiers Trained with Selected Features on COSMIC-FG1 v50 Dataset.

Algorithms	TP Rate	FP Rate	Precision	Recall	F-Measure
J48 (Tree)	**0.985**	0.042	0.825	**0.985**	0.898
Random Forest	0.925	0.027	0.873	0.925	0.899
NB Tree	0.896	0.057	0.759	0.896	0.822
Functional Tree	0.91	0.024	0.884	0.91	0.897
Decision Table	0.94	0.13	0.594	0.94	0.728
DTNB	0.94	0.06	0.759	0.94	0.84
LWL(J48+KNN)	0.925	0.039	0.827	0.925	0.873
Bayes Net	0.91	0.033	0.847	0.91	0.878
Naive Bayes	0.91	0.042	0.813	0.91	0.859
SVM	0.91	**0.021**	**0.897**	0.91	**0.904**
Neural Network	0.866	**0.021**	0.892	0.866	0.879

Moreover, the "Recall" column in Table 3.12 provides another measure of true positives. This indicates that SVM may not have been the best choice of classifier for identify the causative

(causative mutation) where it was either used or claimed to be the best in several previous studies (refer to Introduction). However, if we consider the the averaged predictability, SVM is the best among 11 classifiers. Furthermore, the competitive performance according to multiple measures among the 11 classifiers suggests that the predictions of all these classifiers can provide value in further applications.

Evaluation of Ensemble Classifiers using COSMIC v50

In practice, predictions made by a single expert (classifier) may implicitly include bias to a certain extent, and this sort of biases may also lead to misclassification of the objects. Therefore, combining the outputs of multiple sophisticated classifiers would produce more robust classification results (prediction) [118].

Having evaluated the 11 trained models that are described in the previous section, we selected the models trained on the combined well-performing positive sets — mutations that appear more than once in the COSMIC dataset (COSMIC-FG1) — for further application.

In this section, we present the performance of three ensemble method – Weighted Average, Stacking, and Grading – on the COSMIC-FG1 v50 dataset, with comparison to the performance of individual classifiers introduced in Section 3.7.

Weighted Average Approach

Simple majority ranking approach places equal weight on the predictions made by each of the 11 classifiers. To account for the differences in predictive ability between individual classifiers, we produced an alternative ranking in which each classifier's "vote" is weighted by its accuracy (as determined previously by 10-fold cross-validation), as following:

$$ProbabilityRankScore_i = \frac{\sum_i \left(Pr_i^{j,D} \times Accuracy^j \right)}{\sum_j Accuracy^j}$$

Each of the 11 trained classifiers casts one vote indicating whether a given mutation is causative. The vote here is the sum of the probability of instance *i* being classified as causative, weighted by the accuracy of corresponding classifier. The sum is then normalized by dividing by the sum of the accuracies of the 11 classifiers The result is a decimal score between 0 and 1, where values above our pre-defined threshold, 0.5, indicate a positive prediction.

Combining Multiple Classifiers Improves the Prediction of Causative Mutations

We provide Table 3.13 and Table 3.14 to present the *in silico* experimental results in terms of confusion matrix and several other measurement indexes which quantify the performance of the three ensemble method – Weighted Average, Stacking, and Grading.

Table 3.13: Confusion Matrix – Ensemble Classifiers Trained with Selected Features on COSMIC-FG1 v50 Dataset.

Algorithms	TP	FN	TN	FP
Weighted Average	**67**	**0**	**328**	**3**
Stacking	62	5	293	38
Grading	61	6	324	7

The threshold for Weighted Average is set to 0.5. The meta-classifier is J48, base-classifiers are the other 10 individual classifiers listed in the table.

Table 3.14: Detail Measurement – Ensemble Classifiers Trained with Selected Features on COSMIC-FG1 v50 Dataset.

Algorithms	TP Rate	FP Rate	Precision	Recall	F-Measure
Weighted Average	**1**	**0.009**	**0.957**	**1**	**0.978**
Stacking	0.925	0.115	0.62	0.925	0.743
Grading	0.91	0.021	0.897	0.91	0.904

The threshold for Weighted Average is set to 0.5.
The meta-classifier is J48, base-classifiers are the other 10 individual classifiers listed in the table.

All 3 ensemble classifiers performed fairly well, with TP Rates at least 91% and FP rates at most at most 11.5%. Of the individual classifiers (Table 3.12), J48 performed the best in classifying the causative instances (“+”), while SVM and Neural Network performed the best in classifying the non-causative instances (“−”). However, the weighted average ensemble classifier outperforms all individual classifiers as well as Stacking and Grading, reaching 100% for classifying the causative instances and 99.1% for classifying he non-causative instances in accuracy.

An alternative metric is the “F-Measure”, a harmonic mean of precision and recall, on which all single classifiers achieved scores from 0.822 to 0.904 (Table 3.12), a result consistent with previous studies [46]. The high averaged F-measure score of 0.978 for the weighted average ensemble classifier also vouches for the stability of our feature set on the relatively small training dataset.

In [11], the author of Grading present an empirical evaluation to compare the performance of grading, stacking, voting, and selection by cross-validation on 26 dataset from the UCI Machine Learning Repository [78]. The results show that out of 26 dataset, Grading outperforms others in 5 dataset, cross-validation in 5, Stacking in 8, and Voting in 8. However, Grading approach outperforms all others by averaging the accuracies of the 26 datasets each approach perform, from 0.45% (compares to cross-validation) to 0.16% (compares to Voting). The differences in performance between these ensemble methods are very subtle, therefore there is enough evidence that we should follow the Occam's razor: ”Prefer the simplest hypothesis that fits the data.”, to use Weighted Average Approach.

Application of Ensemble Classifier to Predict Rare Variants in EGFR with COSMIC v50

We combined the 11 classifiers through weighted average approach to effectively identify and prioritize rare EGFR mutations for experimental studies. The rare EGFR mutations are those that appear only once in the COSMIC v50 dataset, under the assumption that since these mutations have not been replicated in other tumor samples, it is possible that some of them may be non-causative. Having withheld these "unconfirmed" samples from the training set, we used the trained ensemble classifier to sort the "unconfirmed" EGFR mutations by their likelihood to be causative.

For each of the "unconfirmed" non-synonymous point mutations in the kinase domain of EGFR, we calculated a numerical score, namely S-Score, for the likelihood to be causative using a weighted average approach in which each classifier's "vote" is weighted by its accuracy as previously estimated by 10-fold cross-validation.

Detailed distribution of the ranked mutations that generated by the weighted average approach is given at Figure 3.2. Based on this ranking approach, if we posit 50% as a threshold to differentiate "Causative" and "Non-Causative", 175 mutations were given more than 50% probability to be "causative" from all 11 classifiers while other 2 with less than 50%. Specifically, 64 mutations ranked between 90% and 100%, 41 ranked 80% to 89.99%, 27 ranked 70% to 79.99%, 31 ranked 60% to 69.99%, 12 ranked 50% to 59.99% - by a applying 50% as threshold, these would be considered likely causative. Of the other mutations, which are likely non-causative, 2 ranked between 40% and 49.99%.

It is clear that the weighted approach gave us a informative prediction result, in terms of indicating the possible probability of each mutation that it is likely to be a causative or a non-causative. The complete list of ranking of these 177 putative EGFR mutations is given in Appendix A.

The top 30 ranked "unconfirmed" mutations in EGFR using the weighted average ensemble classifier is listed in Table 3.15.

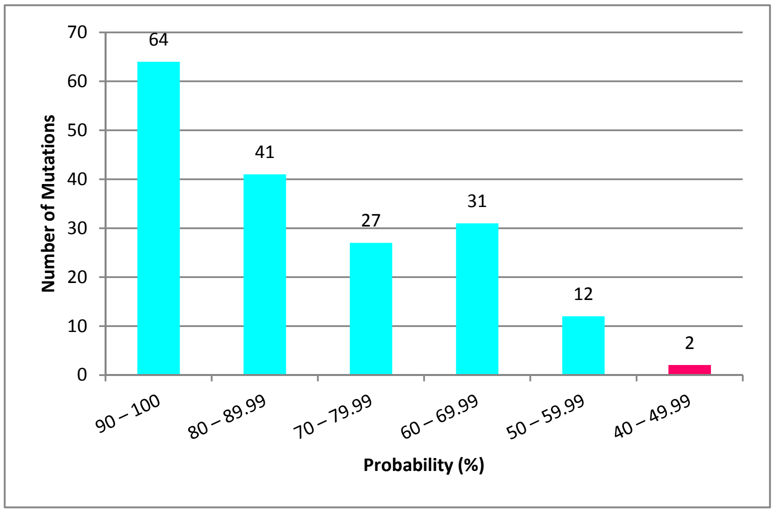


Figure 3.2: Distribution of Predicted 177 Putative EGFR Mutations in COSMIC v50. Prioritized with the weighted average ensemble classifier trained with Experiment Setting I.2.

Selection of Mutations for *in vitro* Experiments

Based on our prioritization as ranked by the weighted average approach, we selected four mutations from the top 30 ranked mutations for further investigation. The selected mutations are L861R and G724S (ranked #1 and #2), and two slightly lower-ranked sites, T725M (ranked #21) and L858Q (ranked #25). Additionally, we selected a fifth mutation that was not strongly predicted to be causative or non-causative, E746K (ranked #161). We chose the lower-ranked mutations because the potential impact of these mutations is not obvious from crystal structure analysis (see Discussion).

Rare variants identified as potentially causative

We looked at the features associated with each of the highly ranked mutations to understand how the combined classifier predicted these mutations to be causatives (Table 3.16). The

Table 3.15: Top Predicted Unconfirmed EGFR Mutations in COSMIC v50 Dataset.

Rank	S-Score	Position	WT	Mutant
1*	0.97699	861	L	R
2*	0.97649	724	G	S
3	0.97644	721	G	S
4	0.97577	858	L	K
5	0.97566	721	G	D
6	0.97559	861	L	P
7	0.97558	862	L	P
8	0.97509	719	G	A
9	0.97507	721	G	A
10	0.97483	729	G	R
11	0.97369	857	G	E
12	0.97365	719	G	V
13	0.97185	854	T	A
14	0.97110	735	G	S
15	0.97023	856	F	S
16	0.96854	856	F	L
17	0.96507	729	G	E
18	0.96462	855	D	G
19	0.96399	779	G	S
20	0.96291	858	L	A
21*	0.96238	725	T	M
22	0.96210	858	L	W
23	0.96034	779	G	C
24	0.95998	723	F	S
25*	0.95649	858	L	Q
26	0.95400	858	L	M
27	0.95381	731	W	R
28	0.95333	799	L	R
29	0.95268	720	S	P
30	0.95253	838	L	P
...				
161*	0.61788	746	E	K

Probability scores and rankings of the top predicted mutations. Scores were calculated with the weighted average ensemble classifier trained on COSMIC v50 data. Asterisks indicate the five mutations selected for cell-based assays.

predicted mutation sites were visualized on a solved crystal structure of EGFR [PDB:1JIU] using PyMOL [1] to view the structural context of each mutation (Figure 3.3).

Table 3.16: Feature Values of Selected Mutations for Cell-based Assays.

`Mutation`	`G724S`	`T725M`	`L858Q`	`E746K`	`L861R`
Protein Family	EGFR	EGFR	EGFR	EGFR	EGFR
Protein Group	TK	TK	TK	TK	TK
Wildtype amino acid	G	T	L	E	L
Blosum62	0	-1	-2	1	-2
Side Chain Polarity Mut	1	0	1	1	1
Conservation AllKinase Wild	0.5453	0.1229	0.4099	0.0493	0.1236
Conservation Group	0.9490	0.3804	0.8039	0.3922	0.2157
Conservation AllKinase	0.7168	0.1469	0.5103	0.1860	0.0700
Conservation Family	1	0.9091	0.8182	0.4545	0.8182
Sub domain	I	I	VIb	II	VIb
Avg Mass	105.0934	149.2078	146.1459	146.1893	174.2027
binding site	NA	NA	NA	NA	NA
Van der Waals Volume Wild	48	93	124	109	124
modification	NA	Phosphorylation	NA	NA	NA
snp amino acid	S	M	Q	K	R
Side Chain Polarity Wild	0	1	0	1	0
Is Pk Domain	0	0	0	0	0

The two top-scoring rare EGFR mutations are L861R and G724S. L861R (rank #1) occurs at the same site as another known causative, L861Q [71], in the activation segment of the kinase domain. L861Q is a frequently occurring lung cancer mutation in EGFR and this mutation is known to activate EGFR [119]; however, the functional impact of the rare L861R is not well understood.

G724S (rank #2) alters a conserved GxGxxG motif in the glycine-rich loop. However, it is the least conserved among the three glycines and does not directly participate in phosphoryl transfer [CITE PKA paper on the G-loop mutational analysis]. We note that there are other naturally occurring, active kinases which also have a serine at this position, such as 3-phosphoinositide dependent protein kinase-1 (PDPK1), calcium/calmodulin-dependent protein kinases (CAMK1a, CAMK2b), casein kinase II (CK2a, CK2b), mitogen-activated

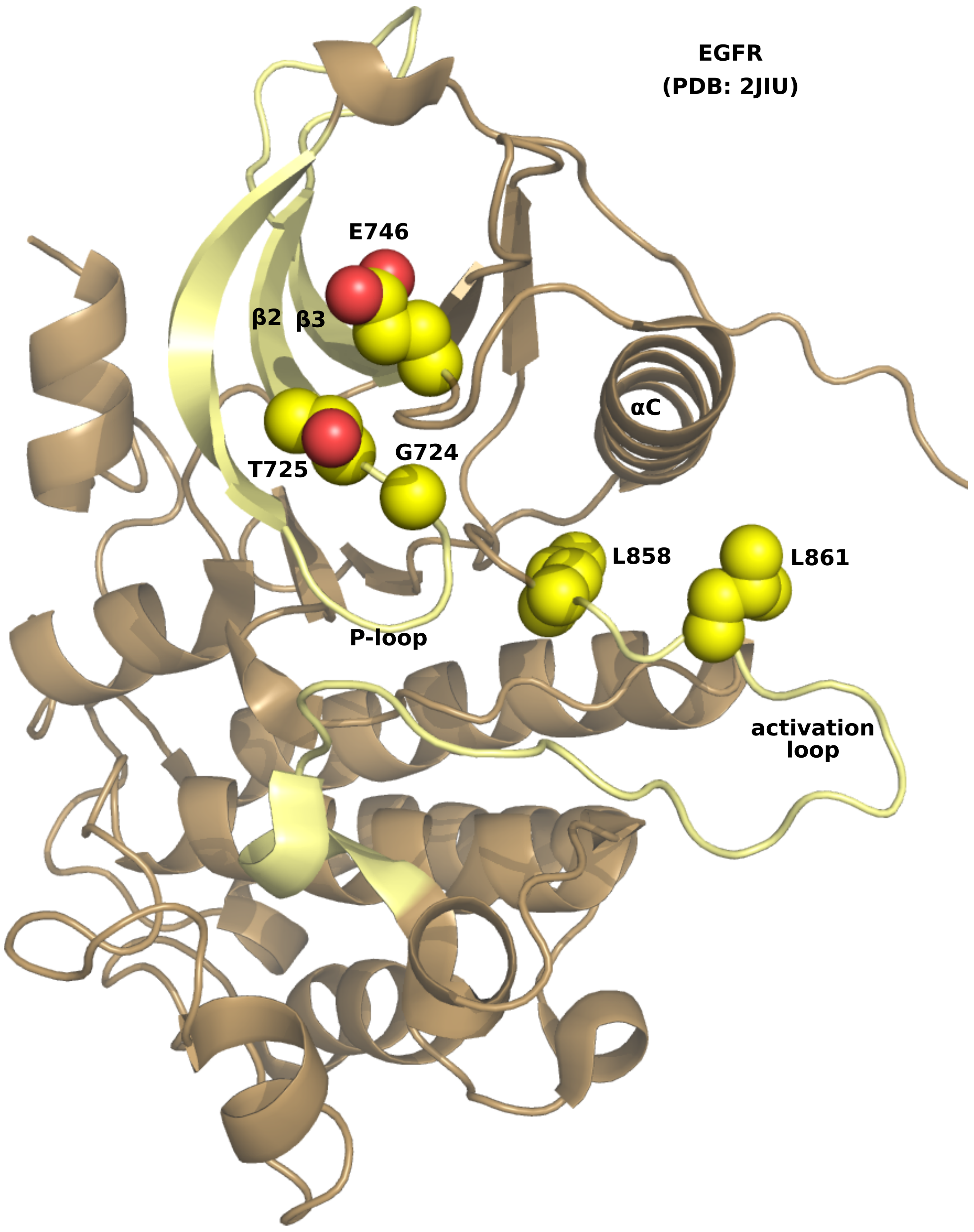


Figure 3.3: Structural location of selected EGFR mutation sites. Protein crystal structure [PDB:2JIU] shown as cartoon, with sites G724, T725, L858 and L861 shown as spheres. Structural regions highlighted in yellow are kinase subdomain I and the activation loop. Generated with PyMOL [1].

protein kinase kinase kinase 1 (MAP3K1), dual-specificity testis-specific protein kinase (TESK1, TESK2) and Tousled kinase 1 (TLK1) and 2 (TLK2) .

In addition to the two high-scoring mutations we also chose additional mutations for experimental analysis (T725M, E746K and L858Q). Specifically, we chose mutations whose impact was not obvious from crystal structure analysis.

T725M (rank #21) is another mutation in kinase subdomain I, in the $\beta2$ strand adjacent to the glycine-rich loop. From a structural perspective this prediction is surprising, given that the residue is solvent-exposed, oriented away from the kinase ATP-binding pocket and does not occur in the active site (Figure 3.3). The mutation type is not particularly drastic (BLOSUM62 score -1), and the conservation of this site is not strong outside across the protein kinase superfamily. It is unclear how mutation of this threonine to a methionine could impact kinase activity. However, the site is strongly conserved within the EGFR family (91%), and is a potential phosphorylation site [120, 121]. The trained classifiers appear to have used these features to assign a high probability of this mutation to be causative.

L858Q (rank #25) occurs in the kinase activation loop, like L861R (above), and is the site of another known frequently occurring lung cancer mutation, L858R [27].

E746K (rank #161) occurs in subdomain II but is spatially close to the site T725, C-terminal to the kinase-conserved VAIK motif, and also solvent-exposed in the EGFR structure (Figure 3.3). Although the ranking is low among other EGFR mutations, the priority score is 0.6178, indicating that it is predicted as more likely than not to be causative.

T725M, E746K and L861R are activating EGFR mutations

EGFR point mutations T725M, E746K and L861R showed increased auto-phosphorylation compared to WT as shown in Figure 3.4. T725M and L861R resulted in hyper-phosphorylation at almost all the sites examined, i.e. Y1086, Y1045, Y845, Y1173 and Y1068. E746K, however, showed enhanced phosphorylation at Y1068, Y845, Y1173 and Y1068. The mutations G724S and L858Q, ranked 2 and 25 respectively (Table 3.15), did not show significant difference in C-terminal tail autophosphorylation compared to wild-type EGFR (Figure 3.4; Figure 3.5). This will be discussed in the discussion section 3.10.

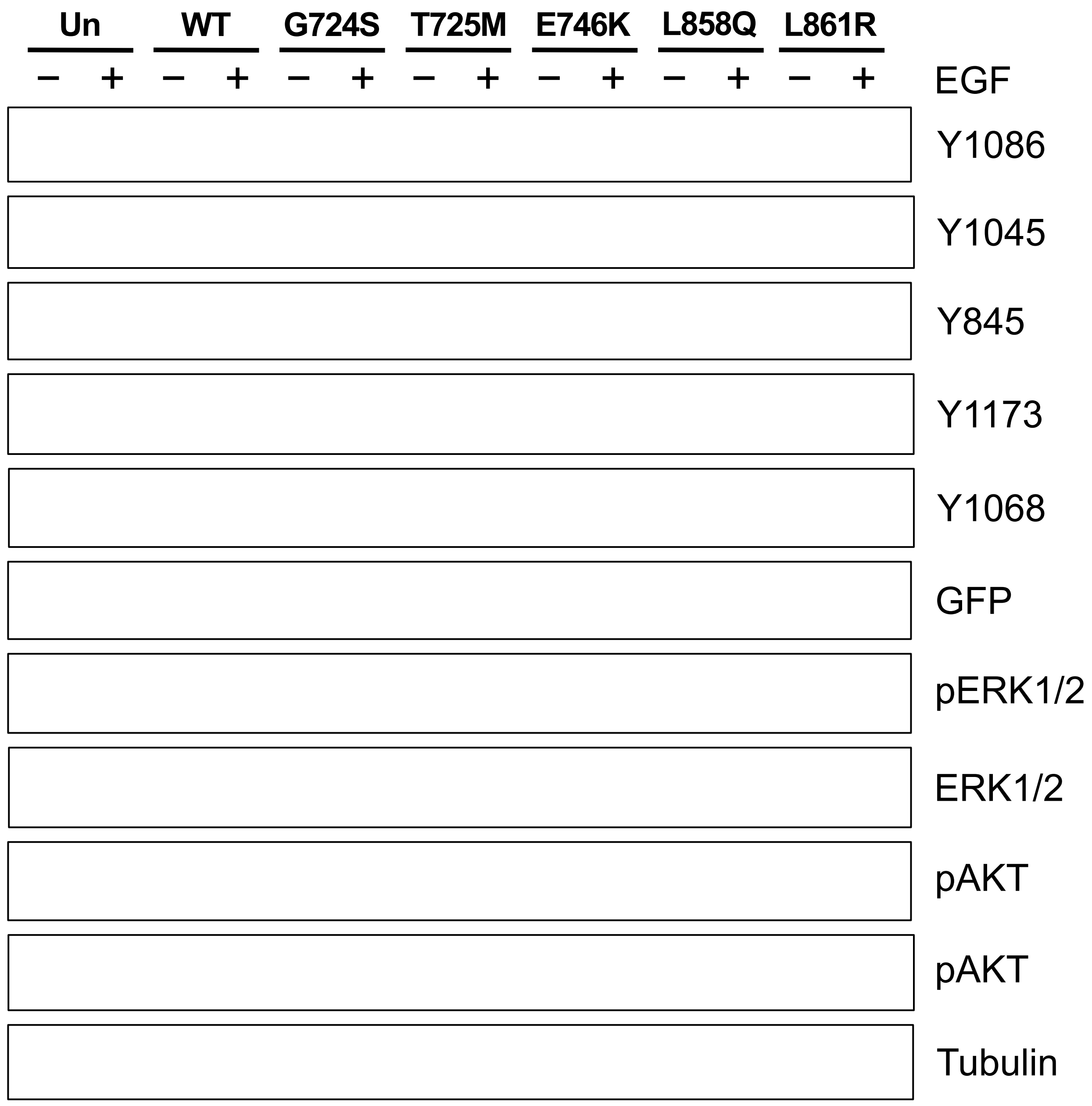


Figure 3.4: Auto-phosphorylation of wild-type and mutant-type EGFR.

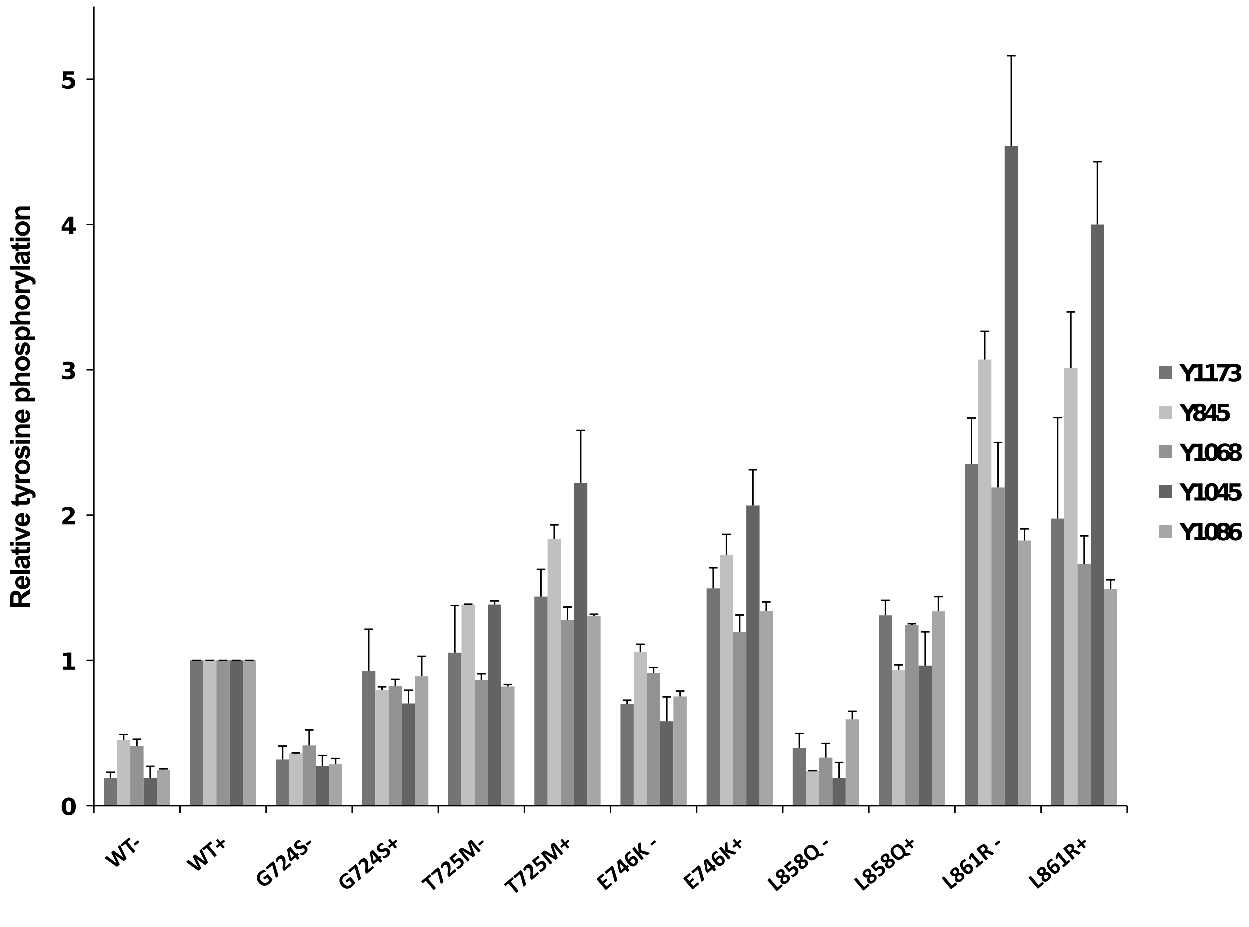


Figure 3.5: Quantified tyrosine auto-phosphorylation levels of wild-type and mutant-type EGFR.

Comparison of COSMIC v50 and v57 : Justification of Using COSMIC-FG1 as Positive Set

An updated dataset – COSMIC v57 – became available after we finished the in-vitro experiment and analysis on the COSMIC v50 dataset. The updated dataset includes more observations on gene mutations. We then curious on how well can our previously trained model (with COSMIC v50 FG1 as positive set) predicted the 71 EGFR mutations that appear only once in COSMIC v50 but appear more than once in COSMIC v57 (Table 3.17 - Experiment Settings II), because this can verify our assumption on using COSMIC-FG1 as positive set for training the predictive models.

Table 3.17: Experiment Setting II. – Based on COSMIC v50 and v57 Dataset.

`Exp. #`	`Training Set`	`Testing Set (Prediction)`
II.1	Mutations (Kinase domain) in COSMIC v50 dataset whose frequency greater than 1	71 EGFR mutations that appear only once in COSMIC v50 but more than once in COSMIC v57

Methodology

To provide further evidence to support the rationale on our selection of the models trained on the well-performing positive sets (COSMIC mutations observed in more than one sample, as described in the main text) for EGFR prioritization, we present a more detailed analysis in this section with the updated COSMIC v57 dataset.

The statistics of the new dataset are presented in Table 3.18. Between COSMIC versions 50 and 57, the number of “causative” instances has increased in the COSMIC-All table by nearly double and the in COSMIC-FG1 table by threefold. As the dataset based on COSMIC v57 is relatively well balanced between the positive and the negative instances, the Cost Sensitive Classifier is not needed here. The number of instance in the updated COSMIC dataset is increased, but biologists still cannot identify which mutation has higher probability to be a causative. This again brings up the urgent request on a stable and highly reliable computational method for this domain.

Table 3.18: Training Dataset based on COSMIC v57.

Dataset	Causative	Non-Causative	Total
COSMIC-All	1084	331	1580
COSMIC-FG1	226	331	557

In the COSMIC v50, there are 177 EGFR mutations that were observed in only one unique sample, while the most updated COSMIC dataset (v57) includes 165 such single-observation mutations. 106 of these mutations are the same among these two versions of COSMIC datasets. Furthermore, 71 EGFR mutations that appear only once in COSMIC v50 appear more than once in COSMIC v57, and there are 59 new EGFR mutations that appear only once in v57 are not included in v50.

The changes between COSMIC v50 and v57 are summarized as following:

- 177 single-observation EGFR mutations in v50
- 165 single-observation EGFR mutations in v57
- 106 single-observation EGFR mutations shared between v50 and v57
- 71 EGFR mutations observed once in v50 but more than once in v57
- 59 EGFR mutations in v57 are new (not in v50)

Results

The result of the comparison is presented in Figure 3.6. Using our previously trained model (with COSMIC v50 FG1 as positive set), of the 71 EGFR mutations whose frequency is equal to 1 in COSMIC v50 but greater than 1 in COSMIC v57, 69 were predicted as causative and the remaining 2 (V786M and R831H) were predicted non-causative. This result further supports our assumption that mutations appear more than once in the COSMIC dataset are more likely to be causative than those appearing only once. The complete list of these 71 mutations is given in Appendix A.

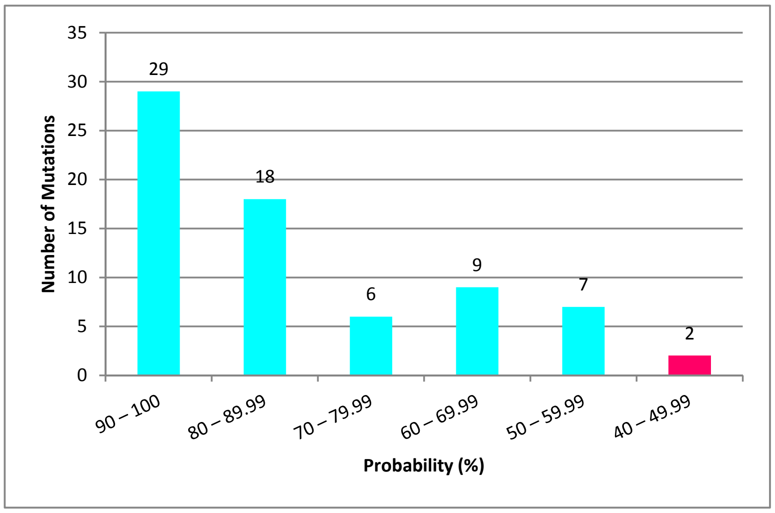


Figure 3.6: Prediction probabilities 71 EGFR mutations observed once in COSMIC v50 but more than once in COSMIC v57.

Evaluation of Individual Classifiers Using COSMIC v57

Based on the experiments presented in previous sections, we designed more *in silico* experiments with the most updated COSMIC dataset, version 57. These updated evaluations provide clear guidance to researchers for the usage of our prediction results for further analysis. The detailed settings of the experiments are summarized in Table 3.19 The statistics of the dataset for experiment III is presented in Table 3.20.

The positive dataset for training in experiment III.1 includes all non-synonymous mutations

Table 3.19: Experiment Setting III - Based on COSMIC v57 Dataset.

Experiment	Training Set	Testing Set (Prediction)
III.1	Mutations in COSMIC v57 dataset, exclude 165 EGFR mutations in the COSMIC v57 dataset whose frequency equal to 1	165 EGFR mutations in the COSMIC v57 dataset whose frequency equal to 1
III.2	Frequency greater than 1	165 EGFR mutations in the COSMIC v57 dataset whose frequency equal to 1

Table 3.20: Training Dataset based on COSMIC v57 – Experiment Setting III.

Exp. #	Dataset	Causative	Non-Causative	Total
III.1	COSMIC-All	1084	331	1415
III.2	COSMIC-FG1	226	331	557

Training set sizes obtained from COSMIC v57 and SNP@Domain under several criteria.

(occuring in the kinase domain) in COSMIC v57, by excluding 165 EGFR mutations whose frequency equal to 1 for testing. Experiment III.2 is designed to use the mutations (Kinase domain) in COSMIC v57 dataset whose frequency greater than 1 as positive set, to predict 165 EGFR mutations in the COSMIC v57 dataset whose frequency equal to 1. Experiment III.1 and IIII.2 are the updates of Experiment I.1 and I.2 with the new version dataset – COSMIC v57.

Removal of Cost Sensitive Classifier

The COSMIC v57 update increased the sized of the positive set, reducing the imbalance relative to the size of the negative set (226 vs. 331, see Table 3.18). As this imbalance is less severe, the Cost Sensitive Classifier was not needed for the *in silico* experiments based on that dataset.

Performance of Individual Classifiers with Experiment Setting III.

Table 3.21 shows the accuracy of the 11 trained classifiers with 10-fold cross-validation of Experiments III. Similarly, the differences of performance between the 11 classifiers are inconspicuous within each experiment, though Decision Table and SVM perform slightly better in experiment III.1 and III.2 respectively.

Table 3.21: Accuracy of 11 Trained Models – Experiment Setting III (COSMIC v57).

Algorithm	Experiment III.1	Experiment III.2
J48 (Tree)	86.3604	96.7684
Random Forest	86.5018	96.2298
NB Tree	85.7951	94.7935
Functional Tree	85.1590	96.9479
Decision Table	**87.9859**	92.9982
DTNB	86.5018	96.9479
LWL(J48+KNN)	86.9965	96.2298
Bayes Net	83.9576	95.8707
Naive Bayes	83.3922	94.6140
SVM	87.8445	**97.3070**
Neural Network	83.0389	96.7684

Evaluation of Individual and Ensemble Classifiers on COSMIC-FG1 v57.

Here we provide the analysis of the performance of the individual classifiers (Table 3.22 and Table 3.23) and ensemble classifiers (Table 3.24 and Table 3.25) using COSMIC-FG1 v57 (Training model of Experiment Setting III.2 and III.3). to present the *in silico* experimental results in terms of confusion matrix and several other measurement indexes.

All 11 individual classifiers and 3 ensemble classifiers performed fairly well, with TP Rates at least 96% and FP rates at most at most 10.6%. Of the individual classifiers (Table 3.11), Decision Table performed the best in classifying the causative instances ("+"), while SVM and Functional Tree performed the best in classifying the non-causative instances ("−").

Table 3.22: Confusion Matrix – Classifiers Trained with Selected Features on COSMIC-FG1 v57 Dataset.

Algorithms	TP	FN	TN	FP
J48 (Tree)	221	5	318	13
Random Forest	216	10	320	11
NB Tree	217	9	311	20
Functional Tree	217	9	**323**	8
Decision Table	**222**	4	296	35
DTNB	219	7	321	10
LWL(J48+KNN)	220	6	316	15
Bayes Net	221	5	313	18
Naive Bayes	218	8	309	22
SVM	219	7	**323**	8
Neural Network	218	8	321	10

All *in silico* experiments were evaluated with 10-fold cross-validation. TP means an instance in the positive set (COSMIC) was correctly classified as causative, TN means an instance in the negative set (dbSNP) was correctly classified as non-causative.

Table 3.23: Detail Measurement - Classifiers Trained with Selected Features on COSMIC-FG1 v57 Dataset.

Algorithms	TP Rate	FP Rate	Precision	Recall	F-Measure
J48 (Tree)	0.978	0.039	0.944	0.978	0.961
Random Forest	0.956	0.033	0.952	0.956	0.954
NB Tree	0.96	0.06	0.916	0.96	0.937
Functional Tree	0.96	0.024	0.964	0.96	0.962
Decision Table	0.982	0.106	0.864	0.982	0.919
DTNB	0.969	0.03	0.956	0.969	0.963
LWL(J48+KNN)	0.973	0.045	0.936	0.973	0.954
Bayes Net	0.978	0.054	0.925	0.978	0.951
Naive Bayes	0.965	0.066	0.908	0.965	0.936
SVM	0.969	0.024	0.965	0.969	0.967
Neural Network	0.965	0.03	0.956	0.965	0.960

However, the weighted average ensemble classifier again outperforms all individual classifiers as well as Stacking and Grading, reaching 98.7% for classifying the causative instances ("+") and 99.1% for classifying he non-causative instances ("–") in accuracy.

Table 3.24: Confusion Matrix – Ensemble Classifiers Trained with Selected Features on COSMIC-FG1 v50 Dataset.

Algorithms	TP	FN	TN	FP
Weighted Average	**223**	**3**	**328**	**3**
Stacking	217	9	323	8
Grading	220	6	323	8

The threshold for Weighted Average is set to 0.5. The meta-classifier is J48, base-classifiers are the other 10 individual classifiers listed in the table.

Table 3.25: Detail Measurement – Ensemble Classifiers Trained with Selected Features on COSMIC-FG1 v57 Dataset.

Algorithms	TP Rate	FP Rate	Precision	Recall	F-Measure
Weighted Average	**0.987**	**0.009**	**0.987**	**0.987**	**0.987**
Stacking	0.96	0.024	0.964	0.96	0.962
Grading	0.973	0.024	0.965	0.973	0.969

The threshold for Weighted Average is set to 0.5. The meta-classifier is J48, base-classifiers are the other 10 individual classifiers listed in the table.

By comparing the averaged F-measure of the training models between experiment III.2 and experiment I.2, we can see that there is on average 0.9% improvement for individual classifiers (Table 3.23), 0.8% for Grading, and 6.9% for Stacking. Although weighted average decreased by 0.4%, it remains at a very high level by reaching 98.9% (Table 3.25. The underlying rationale of these improvements is due to the increment in the size of our dataset and the balance between the number of instances of the causative set and non-causative set, as well as how we select our training data (COSMIC-FG1). In the very beginning, we made the assumption that mutations which appear more than once in the COSMIC dataset have a higher possibility to be causative than those appearing only once, and then we proved the correctness (from a computational point of view) of this assumption by tracking how those mutations that appear only once in COSMIC v50 but more than once in COSMIC v57. Therefore, by comparing experiment III.2 to I.2, it provides further evidence on why the balance of the dataset is important in general, as well as the rationale for selecting COSMIC-FG1 as training dataset could better improve the capability of generalization of our training models.

Application of Ensemble Classifier to Predict Rare Variants in EGFR with COSMIC v57

By applying experiment setting III.2 to the updated COSMIC v57 dataset, the weighted average ensemble classifier predicts all 165 single-observed instance non-synonymous point mutations in the kinase domain of EGFR are likely causatives. Specifically, 146 mutations ranked between 90% and 100%, 18 ranked 80% to 89.99%, 1 ranked 70% to 79.99%. Detailed distribution of the ranked mutations is illustrated in Figure 3.7. The top 30 ranked "unconfirmed" mutations in EGFR using the weighted average ensemble classifier are listed in Table 3.26. The complete list of ranking of these 165 putative EGFR mutations is given in Appendix A.

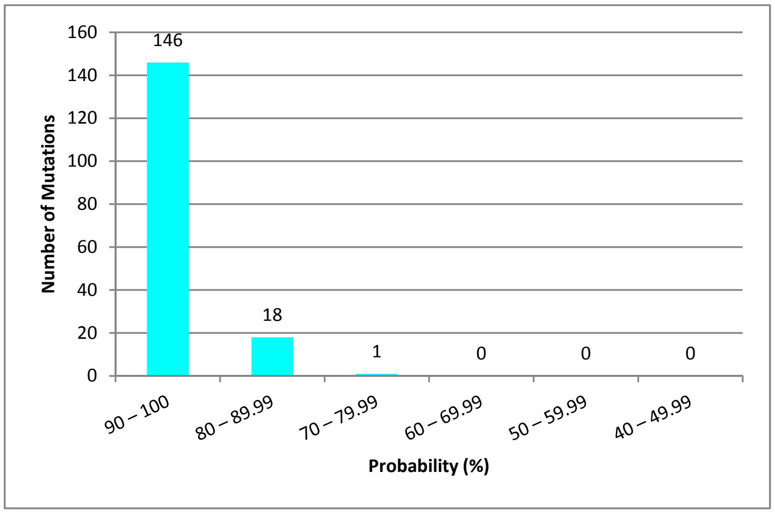


Figure 3.7: Visualization of the probability distribution of predicted 165 EGFR mutations in COSMIC v57 – Weighted Average Approach.

Table 3.26: Top Predicted Unconfirmed EGFR Mutations in COSMIC v57 Dataset.

Rank	S-Score	Position	WT	Mutant
1	0.98234	858	L	A
2	0.98224	858	L	P
3	0.98176	858	L	G
4	0.98137	861	L	P
5	0.98100	721	G	D
6	0.98078	862	L	P
7	0.98018	858	L	K
8	0.97885	779	G	C
9	0.97864	802	V	A
10	0.97769	844	L	P
11	0.97731	851	V	A
12	0.97701	768	S	C
13	0.97679	779	G	D
14	0.97677	729	G	R
15	0.97620	723	F	S
16	0.97557	796	G	D
17	0.97534	854	T	I
18	0.97525	884	E	K
19	0.97485	841	R	K
20	0.97466	856	F	S
21	0.97461	836	R	S
22	0.97404	862	L	Q
23	0.97377	784	S	Y
24	0.97370	869	Y	C
25	0.97366	784	S	P
26	0.97307	855	D	G
27	0.97245	858	L	W
28	0.97209	725	T	A
29	0.97135	866	E	K
30	0.97067	850	H	R

Probability scores and rankings of the top predicted mutations. Scores were calculated with the weighted average classifier trained on COSMIC v57 data.

3.8 Unsupervised Learning Module

In this section, we first introduce the unsupervised learning methods, followed by presenting the details of our self-invented cluster validity metrics. Finally, present the module evaluation and prediction of "unconfirmed" mutations using COSMIC v57 dataset.

Learning Methods

In order to add another level of confidence about the relationship between different mutations in our dataset thereby reducing the label uncertainty, we first use the Expectation-Maximization (EM) algorithm to cluster the mutations, followed by using our self-invented cluster validity metrics to measure the level of oncogenicity of the mutations.

Expectation-Maximization (EM) Algorithm

Expectation-Maximization (EM) [122, 123] is a popular algorithm in data mining. It is an iterative method for finding maximum likelihood or maximum a posteriori (MAP) estimates of parameters in statistical models, where the model depends on unobserved latent variables. In short, EM finds clusters by determining a mixture of Gaussians that fit a given dataset.

The EM [124] algorithm works in two alternating steps: The expectation (E) step computes the expectation of the loglikelihood evaluated using the current estimate for the parameters, it calculates the probability that each datum is a member of each cluster; The maximization (M) step computes parameters maximizing the expected log-likelihood found in the E step, it alters the parameters of each cluster to maximize those probabilities. These parameter-estimates are then used to determine the distribution of the latent variables in the next E step. The E step and M step works iteratively until converges, but there is no guarantee for perfect correctness.

Methodology

As described in Algorithm 15, we first filter the labeled (COSMIC-FG1 v57) and unlabeled (COSMIC-FE1) dataset with the selected features presented in Table 3.7, then we randomize the labeled dataset, followed by randomly splitting the labeled dataset into 90% and 10%. We then combine the 90% split labeled data with the unlabeled dataset, apply EM algorithm onto this combined dataset to perform clustering. Afterwards, we use the trained EM model to cluster the split 10% labeled data. With the clustering outputs, we integrate the unlabeled dataset, and the labeled dataset (both the 90% and 10%) together for validity analysis. For each instance in this combined dataset, we first find the cluster it belongs to, then we compute the metrics introduced in Section 3.8 for this instance.

Repeats the above steps for 10 times, then we generated the averaged result for every single instance in both the labeled and unlabeled dataset, namely "U-Score". Finally, we extract the measurement indexes (i.e. TP, FP, F-Measure, etc.) of the labeled dataset describes in Section Section 3.8 to measure the performance of our method.

Algorithm 1: Algorithm to Extract U-Score

```
input  : Labeled dataset $l\_data$ $(m \times n)$, unlabled dataset $ul\_data$ $(k \times n)$
output : A set of files with a set of measurement metric attached to the input files

1  for $j \leftarrow 0$ to $9$ do
2      $l\_data_f \leftarrow$ FilterFeatures($l\_data$);
3      $ul\_data_f \leftarrow$ FilterFeatures($ul\_data$);
4      $l\_data_{f,r} \leftarrow$ Randomize($l\_data_f$);
       // Randomly split data into 90% and 10%
5      $(l\_data_{f,r,90}, l\_data_{f,r,10}) \leftarrow$ SplitData($l\_data_{f,r}$);
       // Combine unlabeled dataset with the 90% labeled data
6      $ul\_data_f\_and\_l\_data_{f,r,90} \leftarrow$ Combine($ul\_data_f, l\_data_{f,r,90}$);
       // Apply EM algorithm onto the combined dataset
7      $(cluster\_rs1, trained\_em) \leftarrow$ Cluster($em$, $ul\_data_f\_and\_l\_data_{f,r,90}$);
       // Apply the trained EM model onto $l\_data_{f,r,10}$
8      $(cluster\_rs2, trained\_em) \leftarrow$ Cluster($trained\_em$, $l\_data_{f,r,10}$);
9      $cluster\_rs\_all \leftarrow$ Combine($cluster\_rs1, cluster\_rs2$);
10     foreach $instances\ i\ in\ cluster\_rs\_all$ do
11         FindCluster($i$);
12         ComputeDistanceMetric($i$);
13     $rs\_fold\_j \leftarrow$ ChachResult($cluster\_rs\_all$);
   // Average metric value so as to get 10-fold x-validation result
14 $rs\_10\text{–}fold\_X\text{–}validation \leftarrow$ Avg($\sum_{j=0}^{9} rs\_fold\_j$);
   // Compute Measurement Indexes (i.e.  TP, FP, F-Measure, etc.)
15 $output \leftarrow$ ExtractMeasurementIndexes($rs\_10fold\_xvalidation$);
```

Measurement Metrics

In this section, we introduce the definition of our self-invented clustering validity metrics which use to measure the level of oncogenicity of given mutations.

Metric Definition

Definition 1 - ***Class Ratio***.

Calculates the ratio of the causative instances and the ratio of the non-causative instances in the same cluster. **i** *is an instance in the dataset,* **c** *is a* **cluster** *that formed by the unsupervised learning algorithm,* $\mathbf{c_{pos}}$ *and* $\mathbf{c_{neg}}$ *stand for the causative and non-causative instances in cluster* **c** *respectively. To compute the class ratio of cluster* **c** *which instance* **i** *belongs to, we have the following:*

$$R\_Pos_c^i = \frac{\sum_{j \in c_{pos}} j}{\sum_{j \in c_{pos}} j + \sum_{k \in c_{neg}} k} \tag{3.1}$$

$$R\_Neg_c^i = \frac{\sum_{k \in c_{neg}} k}{\sum_{j \in c_{pos}} j + \sum_{k \in c_{neg}} k} \tag{3.2}$$

Definition 2 - ***Intra Distance***.

Calculates the Euclidean Distance of instances in a same cluster **c**. $distance(i, j)$ *is the distance between instance* **i** *and instance* **j** *in cluster* **c**. $\mathbf{c_{pos}}$ *and* $\mathbf{c_{neg}}$ *stand for the causative and non-causative instances in cluster* **c** *respectively. Distances between an instance* **i** *to instances of a class (i.e. Positive or Negative) is infinity (∞) if there is no instance in* **c** *belongs to that particular class.*

$\mathbf{Distance_{avg}}$. *Calculates the average distance between instance* **i** *and other instances, causative ($D\_Pos\_Avg_c^i$) and non-causative ($D\_Neg\_Avg_c^i$), in cluster* **c**. *Details are as following:*

$$D\_Pos\_Avg_c^i = \frac{\sum_{j \in c_{pos}} distance(i,j)}{\sum_{j \in c_{pos}} j} \tag{3.3}$$

$$D\_Neg\_Avg_c^i = \frac{\sum_{k \in c_{neg}} distance(i,k)}{\sum_{k \in c_{neg}} k} \tag{3.4}$$

Distance$_{\mathbf{med}}$. *Calculates the median distance between instance* **i** *and other instances, causative ($D\_Pos\_Med_c^i$) and non-causative ($D\_Neg\_Med_c^i$), in cluster* **c**. *Details are as following:*

$$D\_Pos\_Med_c^i = \arg\operatorname*{med}_{j \in c_{pos}} \{distance(i,j)\} \tag{3.5}$$

$$D\_Neg\_Med_c^i = \arg\operatorname*{med}_{k \in c_{neg}} \{distance(i,k)\} \tag{3.6}$$

Distance$_{\mathbf{min}}$. *Calculates the minimum distance between instance* **i** *and other instances, causative ($D\_Pos\_Min_c^i$) and non-causative ($D\_Neg\_Min_c^i$), in cluster* **c**. *Details are as following:*

$$D\_Pos\_Min_c^i = \arg\min_{j \in c_{pos}} \{distance(i,j)\} \tag{3.7}$$

$$D\_Neg\_Min_c^i = \arg\min_{k \in c_{neg}} \{distance(i,k)\} \tag{3.8}$$

Definition 3 - ***Raw Score***.

Calculates the level of oncogenicity of a instance **i** *in a specific cluster* **c**. *All parameters are pre-defined in definition 1 and definition 2.*

PurityScore$_{\mathbf{avg}}$. *Calculates the Purity Score of instance* **i** *in cluster* **c** *with the pre-*

computed average distance parameters: $D\_Pos\_Avg_c^i$ and $D\_Neg\_Avg_c^i$. Φ is an optional cost ratio that a user can set if the number of instances between different classes in the input dataset are highly imbalanced. ϵ is a very small value (10^{-32}) to avoid divided by zero error. Details are as following:

$$Pos\_Score\_Avg_c^i = \sqrt{\Phi \times R\_Pos_c^i \times (1 - \frac{(D\_Pos\_Avg_c^i)^2}{(D\_Pos\_Avg_c^i)^2 + (D\_Neg\_Avg_c^i)^2 + \epsilon} + \frac{1}{\exp^{R\_Pos_c^i}})} \tag{3.9}$$

$$Neg\_Score\_Avg_c^i = \sqrt{\Phi \times R\_Neg_c^i \times (1 - \frac{(D\_Neg\_Avg_c^i)^2}{(D\_Pos\_Avg_c^i)^2 + (D\_Neg\_Avg_c^i)^2 + \epsilon} + \frac{1}{\exp^{R\_Neg_c^i}})} \tag{3.10}$$

PurityScore$_{\mathbf{med}}$. *Calculates the Purity Score of instance* **i** *in cluster* **c** *with the pre-computed median distance parameters: $D\_Pos\_Med_c^i$ and $D\_Neg\_Med_c^i$. Details are as following:*

$$Pos\_Score\_Med_c^i = \sqrt{\Phi \times R\_Pos_c^i \times (1 - \frac{(D\_Pos\_Med_c^i)^2}{(D\_Pos\_Med_c^i)^2 + (D\_Neg\_Med_c^i)^2 + \epsilon} + \frac{1}{\exp^{R\_Pos_c^i}})} \tag{3.11}$$

$$Neg\_Score\_Med_c^i = \sqrt{\Phi \times R\_Neg_c^i \times (1 - \frac{(D\_Neg\_Med_c^i)^2}{(D\_Pos\_Med_c^i)^2 + (D\_Neg\_Med_c^i)^2 + \epsilon} + \frac{1}{\exp^{R\_Neg_c^i}})} \tag{3.12}$$

PurityScore$_{\mathbf{min}}$. *Calculates the Purity Score of instance* **i** *in cluster* **c** *with the pre-computed minimum distance parameters: $D\_Pos\_Min_c^i$ and $D\_Neg\_Min_c^i$. Details are as* *following:*

$$Pos\_Score\_Min_c^i = \sqrt{\Phi \times R\_Pos_c^i \times (1 - \frac{(D\_Pos\_Min_c^i)^2}{(D\_Pos\_Min_c^i)^2 + (D\_Neg\_Min_c^i)^2 + \epsilon} + \frac{1}{\exp^{R\_Pos_c^i}})} \tag{3.13}$$

$$Neg\_Score\_Min_c^i = \sqrt{\Phi \times R\_Neg_c^i \times (1 - \frac{(D\_Neg\_Min_c^i)^2}{(D\_Pos\_Min_c^i)^2 + (D\_Neg\_Min_c^i)^2 + \epsilon} + \frac{1}{\exp^{R\_Neg_c^i}})} \tag{3.14}$$

Definition 4 - ***Weighted U-Score***.

Calculates the level of oncogenicity of a instance **i** *in a specific cluster* **c**, *based on the average, median, and min pre-calculated Purity Scores. All parameters are pre-defined in definition 3.*

$$Weights = \begin{cases} \alpha = 0.5 \\ \beta = 0.4 \\ \gamma = 0.1 \end{cases} \tag{3.15}$$

$$Pos\_Score\_Weighted_c^i = \alpha \times Pos\_Score\_Min_c^i + \beta \times Pos\_Score\_Med_c^i + \gamma \times Pos\_Score\_Avg_c^i \tag{3.16}$$

$$Neg\_Score\_Weighted_c^i = \alpha \times Neg\_Score\_Min_c^i + \beta \times Neg\_Score\_Med_c^i + \gamma \times Neg\_Score\_Avg_c^i \tag{3.17}$$

Definition 5 - ***Normalized U-Score***.

Calculates the level of oncogenicity of a instance **i** *in a specific cluster* **c**, *with normalization. All parameters are pre-defined in definition 4.*

$$Pos\_Score\_Norm_c^i = \frac{Pos\_Score\_Weighted_c^i}{Pos\_Score\_Weighted_c^i + Neg\_Score\_Weighted_c^i} \tag{3.18}$$

$$Neg\_Score\_Norm_c^i = \frac{Neg\_Score\_Weighted_c^i}{Pos\_Score\_Weighted_c^i + Neg\_Score\_Weighted_c^i} \tag{3.19}$$

Metric Statistics

For the three type of purity scores in definition 3 (min, med, avg) and the final normalized purity scores in definition 5, we canculate different statistical metric for all of them as following.

- $TP = \{\sum j \mid \forall j \in pos \bigcap \forall j \in Pos\_Score_j > Neg\_Score_j\}$
- $FN = \{\sum x \mid \forall x \in pos \bigcap \forall x \in Neg\_Score_x > Pos\_Score_x\}$
- $TN = \{\sum k \mid \forall k \in neg \bigcap \forall k \in Neg\_Score_k > Pos\_Score_k\}$
- $FP = \{\sum y \mid \forall y \in neg \bigcap \forall y \in Pos\_Score_y > Neg\_Score_y\}$
- $Recall = TP/(TP + FN)$
- $Precision = TP/(TP + FP)$
- $FMeasure = (2 \times Recall \times Precision)/(Recall + Precision)$
- $Accuracy = (TP + TN)/(TP + FP + TN + FN)$

- $Min\_Pos = \arg\min\{Pos\_Score_j \mid \forall j \in pos\}$
- $Med\_Pos = \arg\mathrm{med}\{Pos\_Score_j \mid \forall j \in pos\}$
- $Max\_Pos = \arg\max\{Pos\_Score_j \mid \forall j \in pos\}$
- $Min\_Neg = \arg\min\{Neg\_Score_k \mid \forall k \in neg\}$
- $Med\_Neg = \arg\mathrm{med}\{Neg\_Score_k \mid \forall k \in neg\}$
- $Max\_Neg = \arg\max\{Neg\_Score_k \mid \forall k \in neg\}$

We also calculates the difference (magnitude) between the Pos_Score and Neg_Score of causative instances and non-causative instances:

- $Diff\_Pos_j = Pos\_Score_j - Neg\_Score_j \mid \forall j \in pos$
- $Diff\_Neg_k = Neg\_Score_k - Pos\_Score_k \mid \forall k \in neg$

Evaluation of Module Using COSMIC v57

We first use EM clustering algorithm to build the clustering model with COSMIC-FG1 v57 dataset, and then quantify each mutation with our self-invented cluster validity metrics to measure the possibility of being causative mutation. Based on the assumption that instances in COSMIC-FG1 are correctly labeled, we then use the labels as ground truth to measure the performance of the clustering approach and our metrics. As Table 3.27 illustrates, more than 98% causative instances were measured as causative rather than neutral, and about 77% of non-causative instances were measured as non-causative. Furthermore, both Accuracy and FMeasure reached around 0.85, a fairly good performance compares to some previous studies (see Section 3.2probability distribution of predicted 1). The visualization of the corresponding U-Score distribution of the "Causative" and "Non-Causative" labeled instances is shown in Figure 3.8.

The result does not perform as good as the supervised learning module, but this actually indicates that uncertainty does exist in the labels. The supervised learning module learns from the labeled dataset to determine the corresponding parameters for the predictive models, but clustering group instances based on similarity (or distance) between instances without knowing the labels. Therefore, by quantifying the clustering results, we know that uncertainty does exist in the labels, so that we can identify the suspicious mutations whose S-Score and U-Score are very different. Biologists can then target these suspicious mutations to conduct in-vitro experiments for detailed analysis.

Table 3.27: Performance of the Unsupervised Module with COSMIC-FG1 v57 Dataset.

TP	FN	TN	FP	TP Rate	FP Rate	Recall	Precision	FMeasure	Accuracy
222	4	255	76	0.98230	0.22961	0.98230	0.74497	0.84733	0.85637

Application of Unsupervised Learning Module to Predict Rare Variants in EGFR with COSMIC v57

Having built the clustering model, we then applied the built model onto the single-observed instance non-synonymous point mutations in the kinase domain of EGFR to identify and prioritize rare oncogenic mutations.

Detailed distribution of the 165 single-observed EGFR mutations is presented in Figure 3.9. If we keep 50% as a threshold to differentiate "Causative" and "Non-Causative", 161 mutations were given more than 50% Probability to be "causative" while the other 4 less than 50%. Specifically, 12 mutations ranked between 80% to 89.99%, 38 ranked 70% to 79.99%, 36 ranked 60% to 69.99%, 75 ranked 50% to 59.99% – by a applying 50% as threshold, these would be considered likely causative. Of the other mutations, which are possibly non-causative, 4 ranked between 40% to 49.99%. The visualization of the corresponding

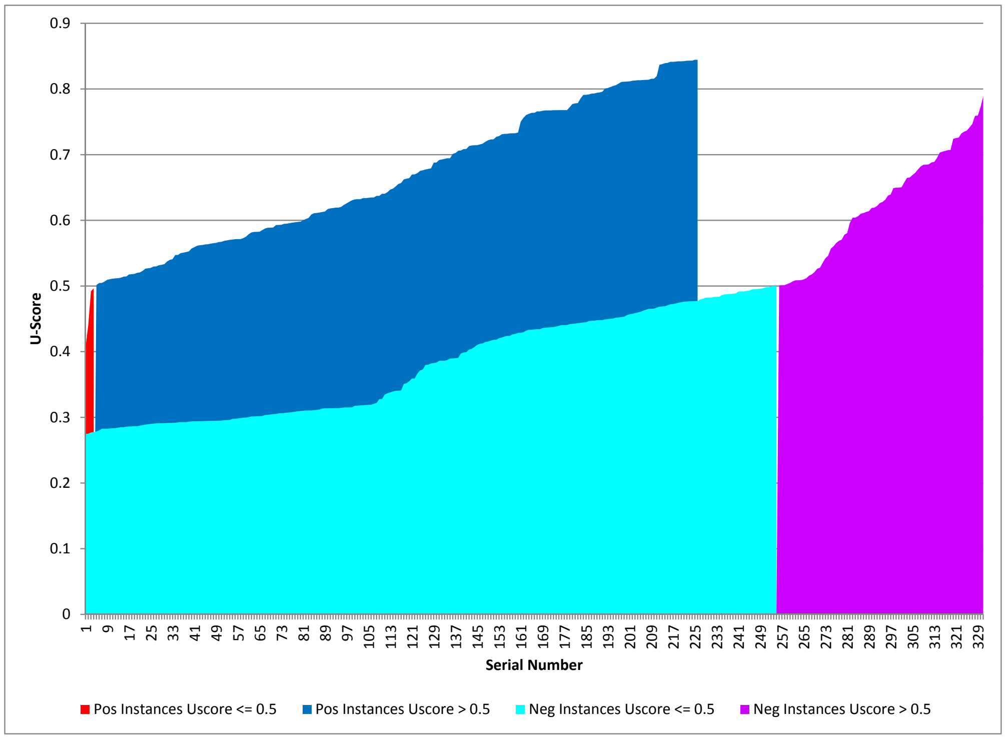


Figure 3.8: Visualization of the U-Score distribution of COSMIC-FG1 v57 dataset.

U-Score distribution is shown in Figure 3.10. The top 30 ranked "unconfirmed" mutations in EGFR using the EM clustering algorithm and our self-invented cluster validity metrics is listed in Table 3.28.

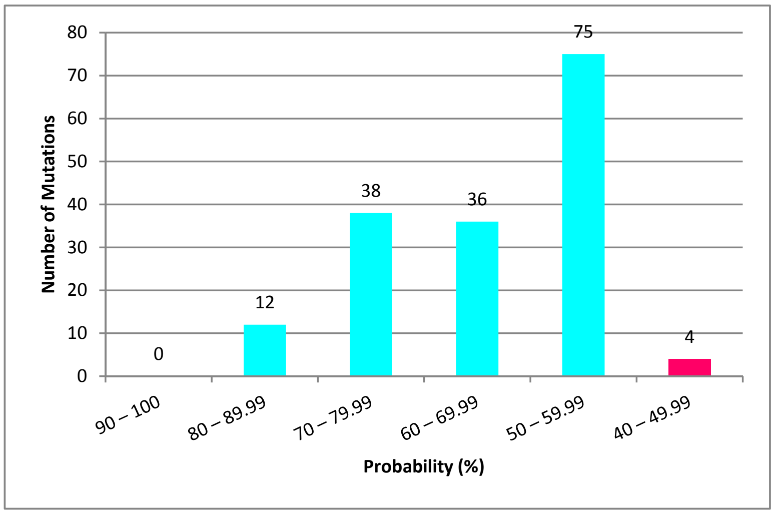


Figure 3.9: Probability distribution of predicted 165 EGFR mutations in COSMIC v57.

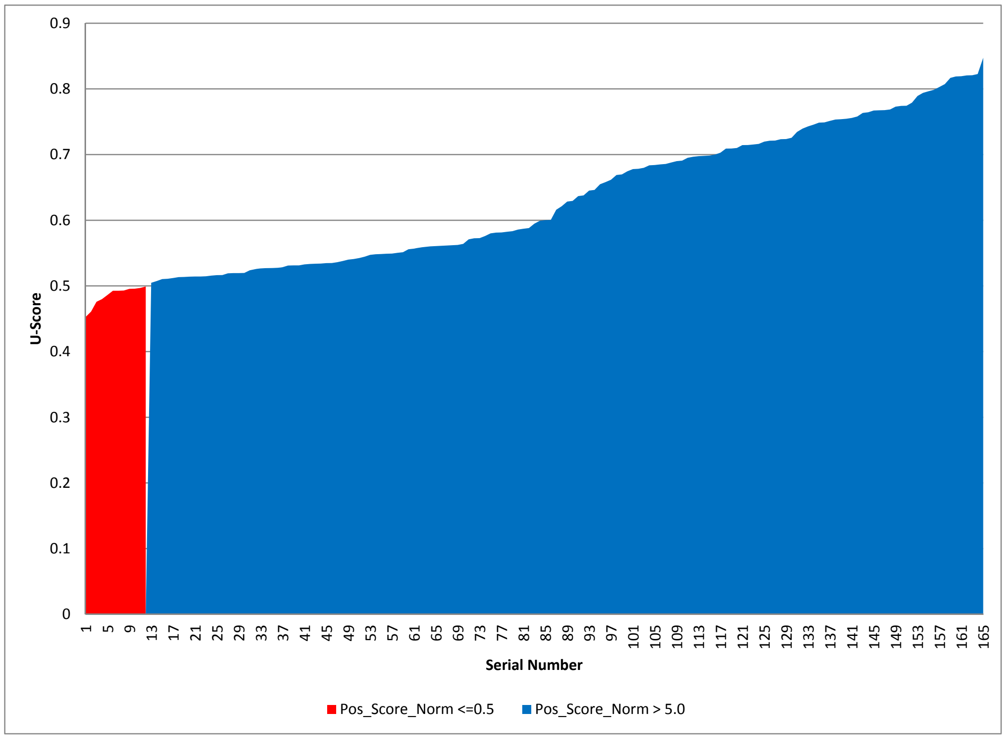


Figure 3.10: Visualization of the U-Score distribution of predicted 165 EGFR mutations in COSMIC v57 dataset.

Table 3.28: Top Predicted Unconfirmed EGFR Mutations in COSMIC v57 Dataset.

Rank	U-Score	Position	WT	Mutant
1	0.84395	721	G	D
2	0.82487	844	L	P
3	0.81762	719	G	V
4	0.81752	796	G	D
5	0.81741	779	G	C
6	0.81549	796	G	V
7	0.81424	858	L	K
8	0.81367	729	G	R
9	0.81341	857	G	E
10	0.81012	777	L	Q
11	0.80950	856	F	Y
12	0.80380	743	A	T
13	0.79642	856	F	S
14	0.79497	884	E	K
15	0.79432	723	F	S
16	0.79324	844	L	V
17	0.79222	743	A	P
18	0.79141	855	D	G
19	0.79119	858	L	P
20	0.79054	798	L	H
21	0.79005	743	A	S
22	0.78435	838	L	V
23	0.78294	828	L	M
24	0.78095	858	L	A
25	0.78008	858	L	W
26	0.77950	726	V	M
27	0.77916	837	D	G
28	0.77664	798	L	F
29	0.77523	858	L	G
30	0.77295	779	G	D

3.9 Combining Supervised and Unsupervised Learning Outputs to Predict Rare Variants in EGFR and to Identify Suspicious Mutations with COSMIC v57

There are two scores in our system – namely the S-Score and the U-Score – that are generated by the supervised and unsupervised learning modules respectively to measure the oncogenicity of a mutation. Both scores are scaled within 0 to 1, where a higher score stands for a higher probability for the corresponding mutation being causative. The S-Score is a probability score that generated through combining 11 well established classifiers (see Section 3.7), with 10-fold cross-validation accuracy as weight. Based on the clustering results generated in the unsupervised learning module, the U-Score is a number that generated by our self-invented metrics which measure the similarity of a mutation to the causative and non-causative mutations in the same cluster it locates (see Section 3.8).

Since there exists a certain level of uncertainty in the labels ("Causative" and "Non-Causative") of our dataset, the predictive model that is trained by supervised learning module might be biased. Therefore we introduced the unsupervised learning module to help reduce the label uncertainty as it performs clustering without considering the labels, and the labels are only used for the computation of the U-Score to measures the oncogenicity based on similarity. We have already seen that both supervised and unsupervised learning modules perform fairly well individually, in this section, we introduce the discovery on our dataset by comparing both S-Score and U-Score. Comparing S-Score and U-Score can help us to identify some suspected mutations that might be labeled incorrectly.

Methodology

We first combine both scores linearly with 9 different ratios, from 10% S-Score and 90% U-Score to 90% S-Score and 10% U-Score combinations with step size 10%, to show how these 9 different ratios between S-Score and U-Score would affect the performance (based on the labels) and what we can discover from that. Afterwards, we select a well performed ratio for further analysis in order to identify the suspicious mutations for Biologists' followup study.

Evaluation of ROMP Using COSMIC v57

Table 3.29: Confusion Matrix – ROMP with Different Ratio between S-Score and U-Score

	Ratio between S-Score and U-Score								
	1S/9U	2S/8U	3S/7U	4S/6U	5S/5U	6S/4U	7S/3U	8S/2U	9S/1U
TP	223	225	224	224	223	223	223	223	223
FN	3	1	2	2	3	3	3	3	3
TN	272	284	309	321	324	326	328	328	328
FP	59	47	22	10	7	5	3	3	3

Table 3.30: Detailed Measurement – ROMP with Different Ratio between S-Score and U-Score

	Ratio between S-Score and U-Score								
	1S/9U	2S/8U	3S/7U	4S/6U	5S/5U	6S/4U	7S/3U	8S/2U	9S/1U
TP Rate	0.9867	0.9956	0.9912	0.9912	0.9867	0.9867	0.9867	0.9867	0.9867
FP Rate	0.1782	0.1420	0.0665	0.0302	0.0211	0.0151	0.0091	0.0091	0.0091
Receall	0.9867	0.9956	0.9912	0.9912	0.9867	0.9867	0.9867	0.9867	0.9867
Precision	0.7908	0.8272	0.9106	0.9573	0.9696	0.9781	0.9867	0.9867	0.9867
FMeasure	0.8780	0.9036	0.9492	0.9739	0.9781	0.9824	0.9867	0.9867	0.9867

With the assumption that instances in COSMIC-FG1 are correctly labeled (instances that observed more than once in the COSMIC table are labeled as causative and common variants that obtained from SNP@Domain are labeled as non-causative), we then use the labels as ground truth to measure the performance of the clustering approach by quantifying the clustering outputs with our self-invented metrics (see Section 3.8).

Table 3.29 and Table 3.30 illustrate the evaluation of the outputs of our metrics based on the clustering results, "XS/YU" stand for $X\%$ S-Score and $Y\%$ U-Score, in where $X + Y = 100\%$. As the tables illustrate, all 9 combinations of S-Score and U-Score perform fairly well, with TP Rates at least 98.67% and FP rates at most 17.82%. The ratio between S-Score and U-Score is a tunable parameter that is set by a user's preference, because we learnt that some Biologists put more trust into supervised learning while others prefer clustering. However, as we mentioned previously in Section 3.8, the target of our unsupervised module is to help reduce the uncertainty of the labels by identifying the suspicious mutations. Considering S-Score or U-Score only does not fit our goal. Therefore, we select "5S/5U", which consider 50% of the S-Score and 50% of the U-Score , for detailed analysis in the following sections. The "5S/5U" split reaches 98.67% TP rate and 2.11% FP rate, a competitive performance compare to the best results states in the tables. The reason we use an arithmetic average to compute the combined score is that S-Score and U-Score are independent events. If one mutation happens to have a low S-Score, the same mutation's chances of getting a low (or high) U-Score is not affected. In other words, S-Score and U-Score are independent from each other.

Figure 3.11 visualizes the S-Score and U-Score of all mutations in COSMIC-FG1 and COSMIC-FE1 v57 dataset. Each number on the X axis is a combined score "5S/5U", an arithmetic mean between S-Score and U-Score of corresponding mutation, scaled from 0 to 1. The Y axis shows the absolute values of S-Score or U-Score, scaled from 0 to 1.

As Figure 3.11 illustrates, some mutations that have high S-Scores are having low U-Scores, and vice versa. This situation happens obviously in a decent amount of mutations in the COSMIC-FG1 dataset, which indicates that uncertainty does exit in the labels. However, the S-Score and U-Score of the majority of the cmutations in COSMIC-FE1 dataset are

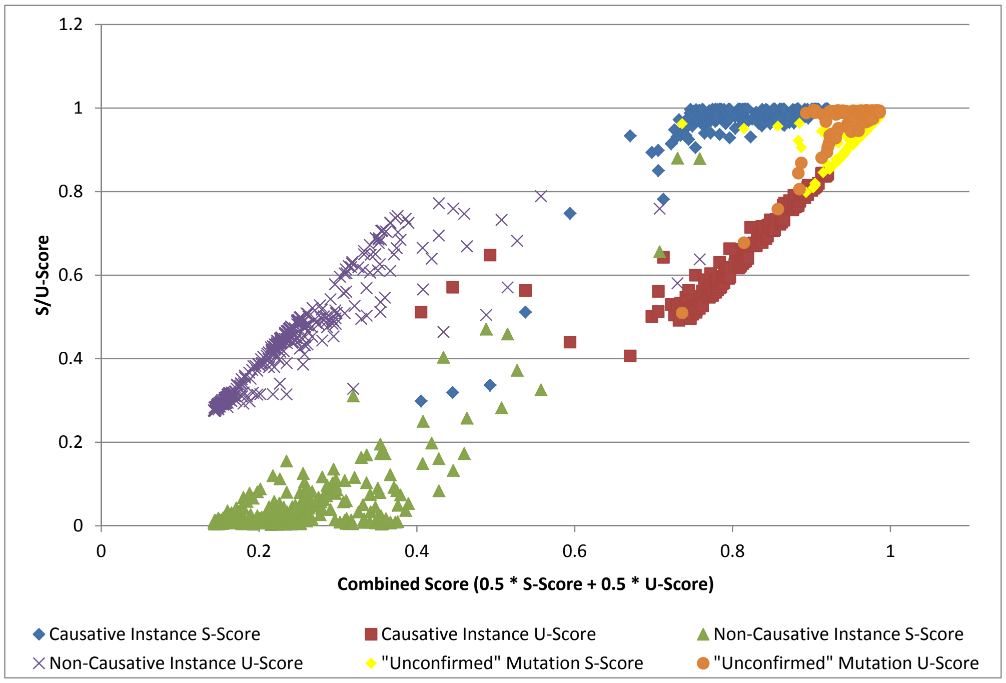


Figure 3.11: Visualization of the Combined Score of mutations in COSMIC-FG1 v57 and COSMIC-FE1 v57 dataset.

having very high coherence, most of them are predicted as causative mutation with high possibility in both S-Score and U-Score. To identify the suspicious mutations, we present the detailed analysis of COSMIC-FG1 v57 and COSMIC-FE1 v57 individually in the following sections.

Application of ROMP to Identify Suspicious Mutations in COSMIC-FG1 v57

Table 3.31 shows the possible S-Score and U-Score combinations for causative and non-causative instances with 0.5 as the threshold. ✓and ✗stands for expected and suspicious output respectively. The instances that labeled as causative in our COSMI-FG1 dataset are based on the assumption that mutations appear more than once in the COSMIC dataset are more likely to be causative than those appear only once, therefore instances that were labeled as causative are expected to have both high S-Score and high U-Score, while the commonly-occurring polymorphisms were labeled as non-causative because the polymorphisms were originally sampled from healthy individuals, and they are expected to have both low S-Score and high U-Score. Any other situations that marked as ✗in Table 3.31 are suspicious and requires further analysis.

Table 3.31: Possible S-Score and U-Score Combinations for Causative and Non-Causative Instances

	Causative Label		Non-Causative Label	
	S-Score $>$ 0.5	S-Score $\leq$ 0.5	S-Score $>$ 0.5	S-Score $\leq$ 0.5
U-Score $>$ 0.5	✓	✗	✗	✗
U-Score $\leq$ 0.5	✗	✗	✗	✓

✓= Expected, ✗= Suspicious

Figure 3.12 visualizes the S-Score and U-Score of all mutations in COSMIC-FG1 v57 dataset. Each number on the X axis is a combined score "5S/5U", an arithmetic mean between S-Score and U-Score of corresponding mutation, scaled from 0 to 1. The Y axis shows the absolute values of S-Score or U-Score, scaled from 0 to 1. As Figure 3.12 shows, the majority of causative labeled instances and the majority of non-causative labeled instances fall into the "Expected" category states in Table 3.31. Specifically, 219 out of 226 instances ($\approx$ 97%) that labeled as causative fall into the "Expected" category, and 255 out of 331 instances ($\approx$ 77%) that labeled as non-causative fall into the "Expected" category. This is a fairly good results to support our assumption that mutations appear more than once in the COSMIC dataset are more likely to be causative than those appear only once. Furthermore, Biologists are in general more interested in the causative mutations (TP) than the non-causative (TN) mutations.

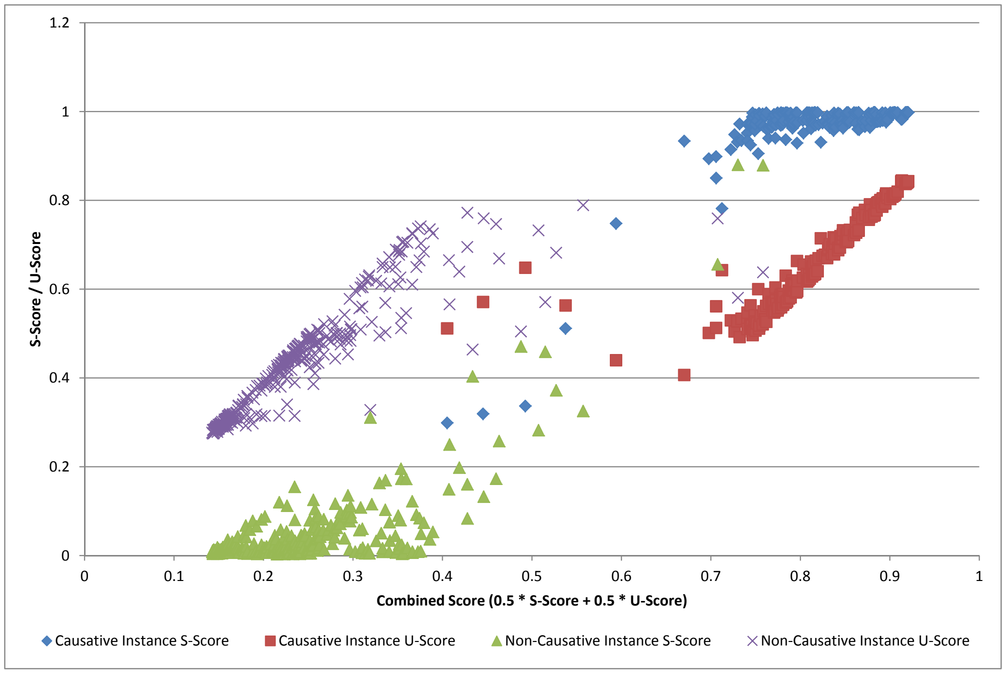


Figure 3.12: Visualization of the Combined Score distribution of 557 mutations in COSMIC-FG1 v57 dataset.

Details of the Suspicious Mutations in COSMIC-FG1 v57

Figure 3.13 illustrates the S-Score and U-Score distribution of the suspicious instances, 7 with causative label and 76 with non-causative label. Table 3.32 shows the detailed information of the 7 suspicious instances with causative labels, and Table 3.33 describes the top 30 suspicious instances with non-causative label.

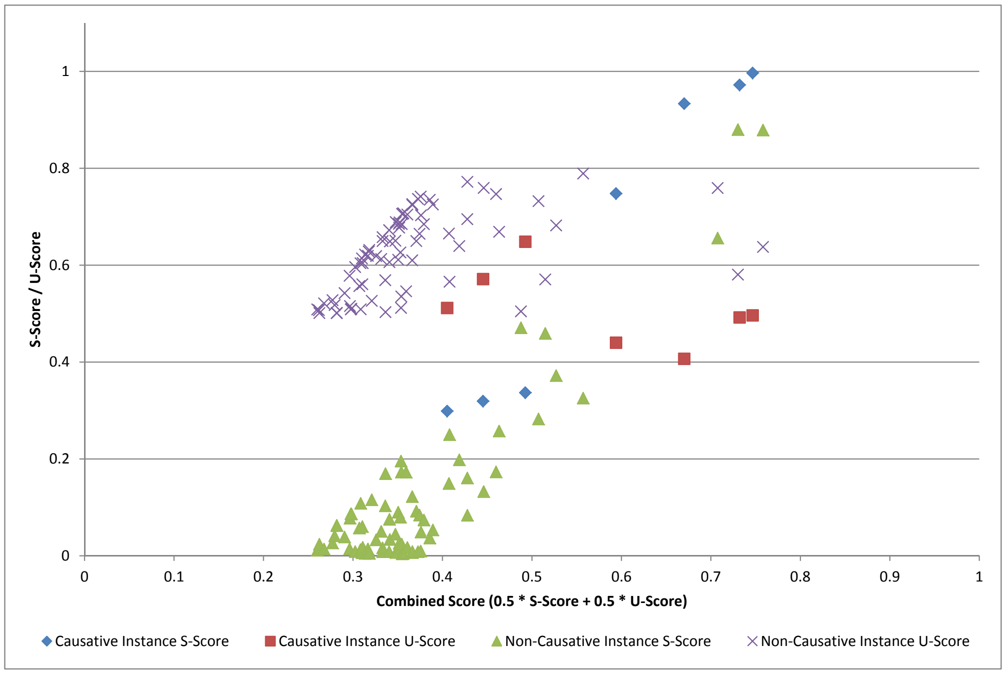


Figure 3.13: Visualization of the Combined Score distribution of 557 mutations in COSMIC-FG1 v57 dataset.

Table 3.32: Suspicious Mutations with Causative Labels in COSMIC-FG1 v57

Rank	Gene	S-Score	U-Score	5S/5U	Position	WT	Mutant
1	BRAF	0.99683	0.49640	0.74662	605	S	N
2	EGFR	0.97213	0.49205	0.73209	717	V	A
3	ERBB2	0.93356	0.40680	0.67018	769	D	H
4	ALK	0.74793	0.44001	0.59397	1275	R	Q
5	MAP2K4	0.33680	0.64836	0.49258	154	R	W
6	MAP2K4	0.31930	0.57128	0.44529	184	S	L
7	FLT1	0.29884	0.51167	0.40526	943	E	K

Table 3.33: Top 30 Suspicious Mutations with Non-Causative Labels in COSMIC-FG1 v57

Rank	Gene	S-Score	U-Score	5S/5U	Position	WT	Mutant
1	ERBB3	0.87885	0.63776	0.75831	758	D	H
2	EGFR	0.88000	0.58050	0.73025	848	V	E
3	EGFR	0.65595	0.75921	0.70758	835	K	N
4	KDR	0.32547	0.78906	0.55726	848	V	E
5	EGFR	0.37216	0.68196	0.52706	962	R	G
6	EGFR	0.45902	0.57067	0.51485	952	V	I
7	ZAK	0.28244	0.73228	0.50736	115	G	S
8	EGFR	0.47082	0.50471	0.48777	890	H	Q
9	CSK	0.25772	0.66898	0.46335	357	S	G
10	PIM2	0.17315	0.74675	0.45995	165	K	R
11	AKT1	0.13271	0.75945	0.44608	319	E	G
12	DDR1	0.08366	0.77202	0.42784	835	R	W
13	ROR1	0.16061	0.69492	0.42776	566	T	M
14	TGFBR2	0.19817	0.63936	0.41876	316	E	V
15	TEC	0.25002	0.56572	0.40787	387	V	A
16	RET	0.14931	0.66529	0.40730	746	E	G
17	PAK4	0.05331	0.72528	0.38930	442	K	N
18	MOS	0.03661	0.73500	0.38581	221	V	A
19	TLK1	0.07387	0.68459	0.37923	646	D	V
20	NEK4	0.04907	0.70305	0.37606	64	N	D
21	MOS	0.00978	0.74150	0.37564	242	T	P
22	SCYL1	0.08409	0.66472	0.37440	59	Q	L
23	CDK4	0.00862	0.73671	0.37267	123	H	Q
24	RIPK3	0.09196	0.64979	0.37087	260	E	V
25	CDK5	0.00882	0.72408	0.36645	171	L	I
26	MAK	0.00663	0.72604	0.36634	269	L	F
27	IRAK1	0.12232	0.61005	0.36618	315	R	G
28	NLK	0.01709	0.70455	0.36082	208	I	T
29	MAP2K7	0.17266	0.54606	0.35936	259	L	F
30	MOS	0.00579	0.70674	0.35627	114	V	L

Application of ROMP to Predict Rare Variants in EGFR with COSMIC-FE1 v57

Figure 3.14 visualizes the S-Score and U-Score of the 165 single-observed EGFR mutations in COSMIC-FE1 v57 dataset. Each number on the X axis is a combined score "5S/5U", an arithmetic mean between S-Score and U-Score of corresponding mutation, scaled from 0 to 1. The Y axis shows the absolute values of S-Score or U-Score, scaled from 0 to 1. As Figure 3.14 shows, all instances fall into the "Expected" causative category states in Table 3.31. A category that both S-Score and U-Score are greater than the pre-defined threshold (0.5). Detailed distributions of the instances according to the combined score "5S/5U" is described in Figure 3.15. Specifically, 1 mutations ranked between 90% and 100%, 67 ranked 80% to 89.99%, 91 ranked 70% to 79.99%, 6 ranked 60% to 69.99% – by a applying 50% as threshold, these would be considered likely causative. The top 30 "unconfirmed" mutations in EGFR ranked by ROMP is listed in Table 3.34.

Application of ROMP to Identify Suspicious Rare Variants in EGFR with COSMIC-FE1 v57

COSMIC-FE1 v57 dataset includes only the EGFR mutations that appear once in COSMIC v57, therefore the suspicious mutations would be those whose S-Score and U-Score do not fall into the same category – both greater than 0.5 or both less than 0.5 (see Table 3.31).

There are 12 out of 165 ($\approx 7.2\%$) suspicious instances in COSMIC-FG1 v57. Figure 3.16 illustrates the S-Score and U-Score distribution of the 12 suspicious instances in COSMIC-FG1 v57. Table 3.35 shows the detailed information of the 12 suspicious instances.

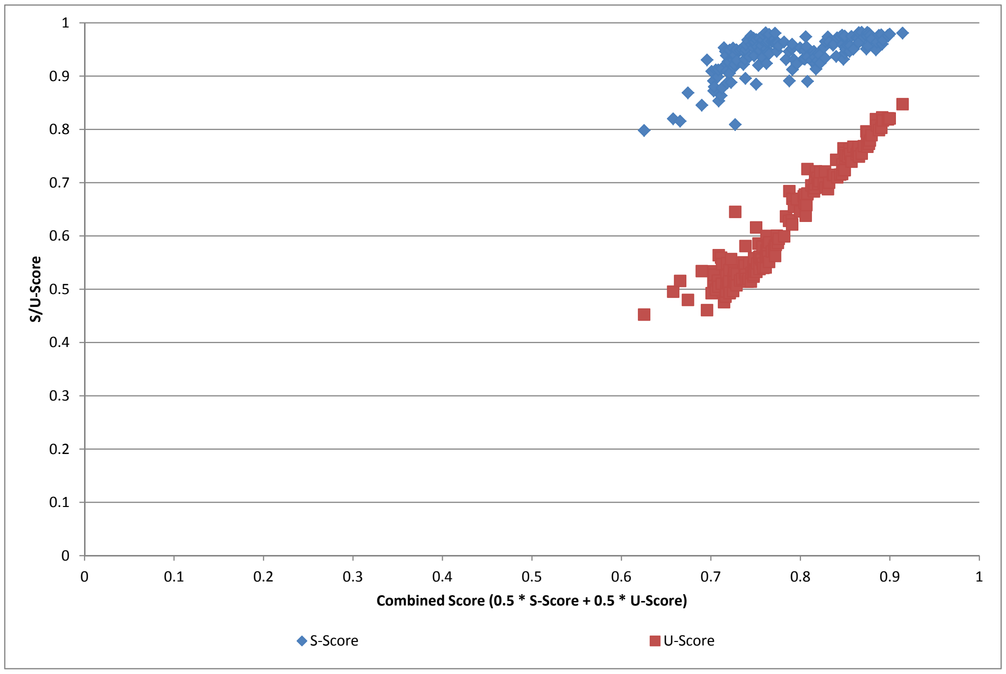


Figure 3.14: Visualization of the Combined Score distribution of predicted 165 EGFR mutations in COSMIC v57 dataset.

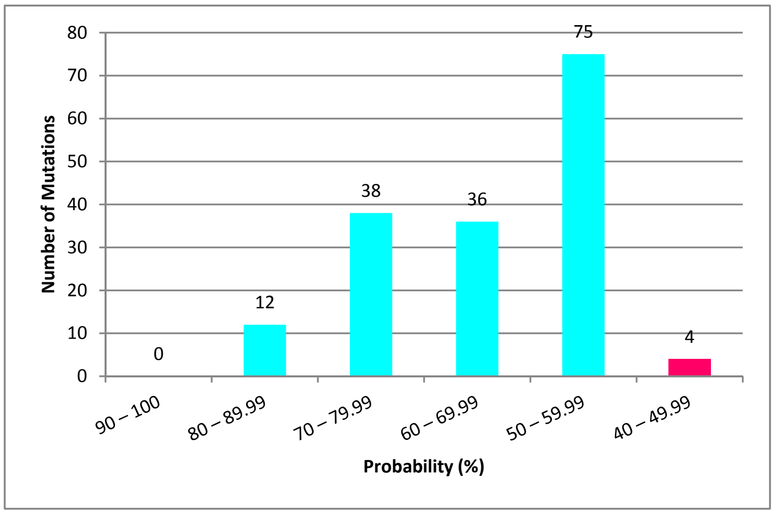


Figure 3.15: Probability distribution of predicted 165 EGFR mutations in COSMIC v57.

Table 3.34: Top Predicted Unconfirmed EGFR Mutations in COSMIC v57 Dataset – ROMP.

Rank	S-Score	U-Score	5S/5U	Position	WT	Mutant
1	0.98100	0.84742	0.91421	721	G	D
2	0.97885	0.82060	0.89972	779	G	C
3	0.97557	0.82074	0.89815	796	G	D
4	0.97677	0.81891	0.89784	729	G	R
5	0.96758	0.81679	0.89218	857	G	E
6	0.96031	0.82264	0.89147	719	G	V
7	0.97769	0.80296	0.89033	844	L	P
8	0.97679	0.79866	0.88773	779	G	D
9	0.96359	0.80726	0.88543	743	A	T
10	0.94949	0.81927	0.88438	796	G	V
11	0.96961	0.78925	0.87943	856	F	Y
12	0.97620	0.77909	0.87764	723	F	S
13	0.98018	0.77298	0.87658	858	L	K
14	0.95653	0.79392	0.87522	743	A	S
15	0.98224	0.76748	0.87486	858	L	P
16	0.97525	0.77406	0.87465	884	E	K
17	0.97466	0.77436	0.87451	856	F	S
18	0.95105	0.79638	0.87372	743	A	P
19	0.97307	0.76868	0.87088	855	D	G
20	0.98234	0.75449	0.86842	858	L	A
21	0.98176	0.74900	0.86538	858	L	G
22	0.96269	0.76707	0.86488	844	L	V
23	0.97055	0.75802	0.86428	838	L	V
24	0.97245	0.75335	0.86290	858	L	W
25	0.95073	0.76763	0.85918	777	L	Q
26	0.97461	0.73956	0.85708	836	R	S
27	0.95719	0.75384	0.85552	837	D	G
28	0.94656	0.76342	0.85499	742	V	A
29	0.95716	0.74865	0.85291	798	L	F
30	0.95002	0.75133	0.85067	726	V	M

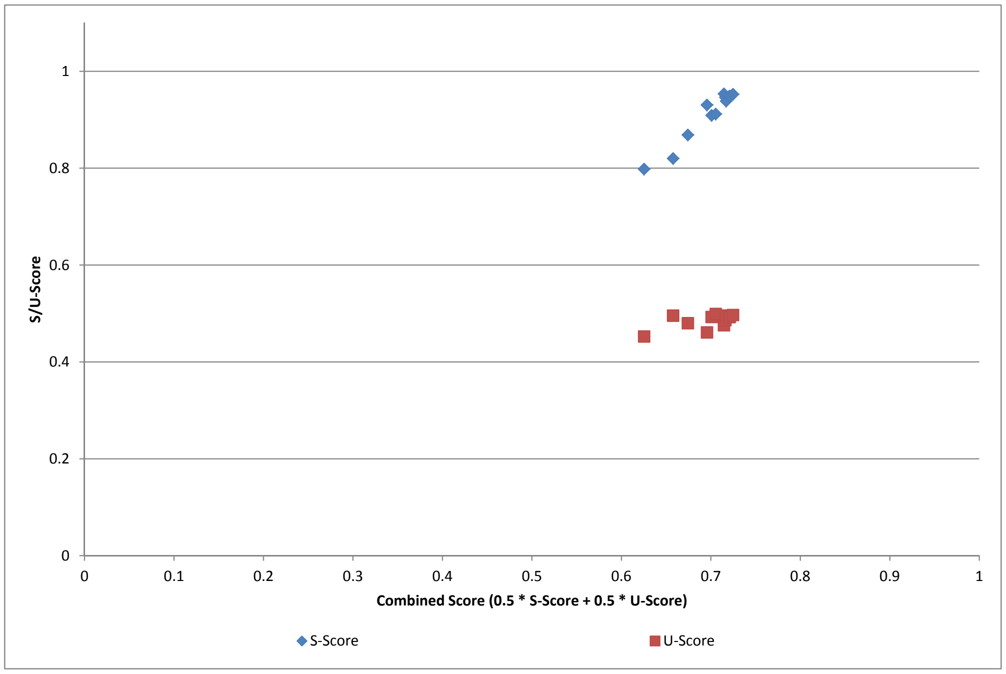


Figure 3.16: Visualization of the Combined Score distribution of 165 mutations in COSMIC-FE1 v57 dataset.

Table 3.35: Suspicious Mutations in COSMIC-FE1 v57

Rank	Gene	S-Score	U-Score	5S/5U	Position	WT	Mutant
1	EGFR	0.95266	0.49687	0.72476	868	E	D
2	EGFR	0.94909	0.49298	0.72103	728	K	R
3	EGFR	0.94836	0.49264	0.72050	716	K	R
4	EGFR	0.93849	0.49574	0.71712	768	S	N
5	EGFR	0.94630	0.48618	0.71624	730	L	F
6	EGFR	0.95343	0.47584	0.71463	720	S	T
7	EGFR	0.91175	0.49907	0.70541	785	T	N
8	EGFR	0.90907	0.49265	0.70086	860	K	E
9	EGFR	0.93045	0.46086	0.69565	758	E	D
10	EGFR	0.86865	0.47990	0.67428	829	E	Q
11	EGFR	0.82004	0.49555	0.65779	757	K	N
12	EGFR	0.79813	0.45250	0.62531	714	K	N

3.10 Discussion

Mutational activation of EGFR is implicated in many cancers including lung, head and neck cancer, and clinical and cancer genome sequencing studies have identified hundreds of mutations in the protein kinase domain. However, much of the focus thus far has been on a handful of frequently occurring mutations such as L858R and L861Q, while relatively little is known about the many rare mutations in EGFR such as T725M. The scoring scheme and multiple classifier approach we have introduced here helps identify key rare mutations for follow-up experimental studies.

In particular, our studies suggest T725M as a likely causative mutation because it increases EGFR auto-phosphorylation activity in comparison to wild-type and other activating mutations such as L858R. The impact of the T725M mutation cannot be predicted by existing structural or functional information alone, and clinical samples do not currently highlight this as a recurrent mutation. However, the multiple classifier approach has helped identify T725M as a likely causative mutation.

The frequently occurring L861Q is known to activate EGFR, but the impact of the rare L861R is not known. Here we show that L861R is likely to be causative insasmuch as it activates EGFR in a manner analogous to L861Q.

The discrepancy between predictions and experiments for two other mutations (L858Q and G724S) suggests that there are additional nuances in the oncogenic mechanisms that remain to be incorporated into our models. This does not necessarily mean that G724S and L858Q are not oncogenic, as these mutations might instead lead to the activation of other phosphorylation sites that we have not analyzed — not to mention that activation of downstream pathways including ERK1/2, AKT and JAK/STAT are controlled by specific phosphorylation sites in EGFR [125–130]. A higher level of EGFR phosphorylation is often linked to malfunctioning of downstream pathways, which in turn leads to abnormal cell growth and proliferation.

L858R is a frequently occurring lung cancer mutation which activates EGFR and also impacts drug binding [27]. This information perhaps contributed to the classification of L858Q as causative. However, our mutational experiments revealed that the L858Q mutant behaves like the wild-type protein and does not increase activity as predicted. The context of this mutation suggests that the activation loop of EGFR is a frequent site of activating mutations; however, the L858Q mutation in particular may not cause the mechanistic changes necessary to increase phosphorylation, at least at the EGFR C-tail sites we tested.

3.11 Conclusion

We have introduced certain novel features for identifying cancer-causing mutations in cancer kinome, and we have also utilized various feature selection methods to evaluate our proposed features. Our experiments provided insight into the relationship between different levels of evolutionary conservation and the functional effect of mutations, a consequence of the novel features introduced.

Our weighted average ensemble classifier, and our self-invented clustering validity metrics based on the EM clustering outputs performed well according to a variety of metrics. On the basis of this result, we combined these two modules to produce a prioritized list of putative driver mutations in EGFR. These predictions provide useful guidance for further experimental follow-up - in particular, an assay of the kinase activity of a few of the mutant forms predicted with the highest confidence could confirm these predictions and indicate the most promising targets for pharmaceutical inhibition. Furthermore, we also identified the suspicious mutations in the COSMIC dataset for further analysis. These lists of suspicious mutations provides a good insight to Biologists to conduct detailed research to discover more hidden information of the mutations.

As our catalog of known causative mutations improves we can further improve our prediction system, to build a more sophisticated model to differentiate between causative and non-causative mutations in cancer. Moreover, our work could be extended to a prediction tool with clinical value, as well as providing a basis for further investigation into the relationship between protein evolution and disease.

Chapter 4

VAMO: Towards a Fully Automated Malware Clustering Validity Analysis

In Chapter 3, we present an machine learning framework (ROMP) which combines multiple supervised learning algorithms, an unsupervised learning algorithm, as well as our self-invent cluster validity metrics to effectively predict the oncogenic mutations in the cancer kinome. The unsupervised learning module, especially our self-invented clustering validity metrics in ROMP helps to improve the outputs from the ensemble classifiers. Specifically, it helps to identify a set of suspicious mutations for Biologists' followup study. It is worth to mention that, there is no missing labels in the COSMIC dataset, the amount of different labels in the dataset are easy to manage (only causative and non-causative), and labels that generated by different experts (individual learning algorithms) are commonly understood by each other. Furthermore, we also have good understanding of how each expert performs on the given dataset (10-fold cross-validation accuracy). All these make our framework become

Perdisci R, U M (2012) VAMO: Towards a Fully Automated Malware Clustering Validity Analysis. *Proceedings of the 28th Annual Computer Security Applications Conference.* New York, NY, USA: ACM, ACSAC'12, pp. 329-338.

measurable and combining multiple experts' outputs appear to be not highly complicated.

However, in some circumstances experts do not have enough knowledge to label all instances in the dataset which may make the dataset have missing or incorrect labels. To make the situation worse, labels that generated by different experts are partially understood (or even can't be understood) by each other, and there is no prior information available about how accurate each expert performs. Then the task of combing the outputs of multiple experts to generate robust outputs would become difficult. Even though such "robust" outputs can be generated, it might be under certain restrictions. For instance, by applying the majority voting approach, samples that cannot be reached the consensus by majority experts need to be removed. Therefore, we introduce a framework – VAMO – in this chapter which is capable to address all above issues. Our framework generates reliable outputs by automatically reducing the label uncertainty, it can be used by other experts to assess the quality of their learning results. We introduce VAMO by applying it onto malware clustering as there is lots of uncertainty in the domain that we mentioned above.

Malware clustering is commonly applied by malware analysts to cope with the increasingly growing number of distinct malware variants collected every day from the Internet. The main objective of malware clustering systems is to group malware samples into *families*, whereby samples that are similar to each other can be considered as variants of the same malware family. Intuitively, malware clustering results can be useful in several ways. For example, new malware samples that are clustered with known malware variants of a given family may be also categorized as belonging to that same family. These newly discovered variants may be used to derive more generic malware detection signatures that have a better chance to match *future variants* of the same family. While malware clustering systems can be useful for a variety of applications, assessing the quality of their results is intrinsically hard. In fact, clustering can be viewed as an *unsupervised learning* process over a dataset for which the complete ground truth is usually not available. Previous studies propose to evaluate malware clustering results by leveraging the labels assigned to the malware samples by multiple anti-virus scanners (AVs). However, the methods proposed thus far require a (semi-)manual adjustment and mapping between labels generated by different AVs, and are limited to selecting a *reference sub-set* of samples for which an agreement regarding their labels can be reached across a majority of AVs. This approach may bias the reference set towards “easy to cluster” malware samples, thus potentially resulting in an overoptimistic estimate of the accuracy of the malware clustering results.

In this chapter we propose VAMO, a system that provides a *fully automated* quantitative analysis of the validity of malware clustering results. Unlike previous work, VAMO *does not seek a majority voting-based consensus* across different AV labels, and does not discard the malware samples for which such a consensus cannot be reached. Rather, VAMO explicitly deals with the inconsistencies typical of multiple AV labels to build a more representative reference set, compared to majority voting-based approaches. Furthermore, VAMO avoids the need of a (semi-)manual mapping between AV labels from different scanners that was required in previous work. Through an extensive evaluation in a controlled setting and a real-world application, we show that VAMO outperforms majority voting-based approaches, and provides a better way for malware analysts to automatically assess the quality of their malware clustering results.

4.1 Introduction

Due to the extensive use of *packing* and other code obfuscation techniques [131], the number of new malware samples collected by anti-virus (AV) vendors has grown enormously in recent years, reaching tens or even hundreds of thousand of new malware samples collected per

Validity Analysis of Malware-clustering Outputs.

While “anti-malware” is probably a more appropriate term, we use “anti-virus” because that is the way in which many vendors of malware scanners and defense solutions still advertise their products.

day (e.g., in 2010 Symantec collected 286 million distinct malware variants [132]). To cope with this increasingly growing number of malware samples and boost the scalability and effectiveness of current malware analysis infrastructures, a number of *malware clustering* and automatic malware categorization systems have been recently proposed [133–139].

The main objective of malware clustering systems is to group malware samples into *families*, whereby samples that are similar to each other can be considered as variants of the same malware family. Intuitively, malware clustering results can be useful in several ways. For example, new malware samples that are clustered with known malware variants of a given family f may be also categorized as belonging to f. In turn, these newly discovered variants may be used to derive more generic malware detection signatures that have a better chance to match *future variants* of the same family [138]. In addition, malware clustering results may make it easier to identify new, previously unknown malware families [136], or may be used to perform malware triage [139], thus allowing malware experts to select only a small number of variants of a given malware family for manual analysis.

To take full advantage of the above mentioned benefits, malware clustering systems clearly need to be accurate. Unfortunately, it is very challenging to quantitatively assess the accuracy of malware clustering results, because of the lack of reliable ground truth. A common approach to validating the quality of malware clustering results is to compare them to a *reference clustering* obtained by leveraging family labels assigned to the samples by multiple AV scanners [136, 139]. To compensate for inconsistencies in the AV labels, both [135, 136] and [139] use a majority voting approach to select the samples for which an agreement regarding their *AV family* label can be reached. Therefore, a cluster in the reference clustering will include all samples belonging to the same AV family. However, while this approach may appear as a natural choice in absence of complete ground truth, Li et al. [140] have suggested that it may result in an overoptimistic estimate of the malware clustering accuracy. In particular, limiting the reference clustering to samples for which a majority voting-based consensus on the family label can be reached, and discarding the remaining ones, may reduce the reference clusters to only include "easy to cluster" malware samples (i.e., clear-cut cases of malware samples that are very similar to each other) [140], thus potentially causing the accuracy of the malware clustering results to be largely overestimated. In fact, the experiments reported in [136] state that among 14,212 malware samples, a majority voting-based consensus could be reached only for 2,658 cases. That is, more than 80% of the samples in the clustering results had to be excluded from the cluster validity analysis.

In this chapter we propose VAMO, a system that enables an automatic quantitative analysis of the validity of malware clustering results. Like previous work, VAMO leverages the labels assigned to malware samples by multiple AV scanners to construct a reference clustering. However, unlike previous work, VAMO *does not seek a majority voting-based consensus*, and does not discard the samples for which such a consensus cannot be reached. Rather, VAMO explicitly deals with (and aims to mitigate the effect of) the inconsistencies typical of the AV labels to build a more representative reference clustering. Furthermore, VAMO avoids the need of a (semi) manual mapping between AV labels from different scanners that was required in previous work (notice that while some efforts exist to standardize the "language" used to assign the AV labels (e.g., http://maec.mitre.org), so far they have not been successful). Also, we would like to emphasize that while AV labels suffer from some limitations, as we discuss more in detail in Section 4.6, they are used as a reference by many researchers because it is hard to obtain a more accurate ground truth for datasets containing tens of thousands of malware samples.

VAMO leverages historic malware archives and the related multiple AV labels to *learn an AV Label Graph* (see Figure 4.1). An *AV Label Graph* (see Section 4.4) is defined as an undirected weighted graph, which aims to: (1) automatically learn the mapping between malware family names assigned by different AVs, thus avoiding the need to manually build or adjust such mappings; (2) identify cases in which one (or more) AV scanners tend to inconsistently use several family names to label samples that belong to the same family according to other competitors' scanners; (3) learn the *level of similarity* between AV labels assigned by different AV scanners, by looking at the number of times that certain malware family labels are jointly assigned to the same samples. While the concept of *AV Label Graph* was first introduced in [138], here we refine its definition and use it in the context of our novel VAMO system. Also, it is worth noting that the *AV Label Graph* is only one component of the entire VAMO system.

Learning the *AV Label Graph* enables us to measure the similarity between malware samples in a dataset based purely on their AV labels (see Section 4.4 for details). As shown in Figure 4.1, given a malware dataset $\mathbf{M}$ and the related multiple AV labels assigned to its malware samples, we can (a) apply a third-party malware clustering algorithm (e.g., [135, 136, 138, 139]) on $\mathbf{M}$ to partition it in a number of malware clusters, (b) use VAMO to build a reference clustering for $\mathbf{M}$ using similarities among its samples measured according to their AV labels, and (c) compute the *level of agreement* between VAMO's reference clustering and the third-party malware clustering results, thus quantitatively assessing their quality.

In summary, this paper makes the following contributions:

- We propose a novel system, called VAMO, that enables a *fully automated* malware clustering validity analysis.

- We perform an extensive evaluation of how different types of AV label inconsistencies may negatively impact a validity analysis performed via majority voting-based approaches, and show the advantages that VAMO brings over previous work.

- We perform experiments with real-world malware archives, and demonstrate how VAMO can be applied in practice to assess the quality of malware clustering results over large malware datasets.

4.2 Background

In this Section, we first provide quantitative information regarding the inconsistency typical of multiple AV labels. Then, we discuss the background concepts that we will use to perform automated clustering validity analysis.

Related Work

Cluster Validity Analysis Besides the clustering validity indexes reported in Halkidi et al.'s survey [15], which we summarize in Section 4.2, a number of alternative validity indexes have been proposed. In [141], Rendon et al. present a comparison of internal and external clustering validity indexes, while Meilă [142] and Pfitzner et al. [143] introduce a number of new metrics to compare two different clusterings. In [144], Fowlkes and Mallows introduce a measure of similarity between two hierarchical clusterings obtained by cutting the two dendrograms at heights h_1 and h_2, respectively, which yield the same number of clusters k. Than, for each value of k, the number of matching entries from the two different clusterings are counted to obtain a measure of comparison. Our approach to cluster validity analysis (Section 4.4) is inspired by [144]. However, our method does not focus on comparing different hierarchical clusterings. Rather, VAMO leverages hierarchical clustering to generate a *reference clustering dendrogram*, and compares third-party clustering results to this dendrogram by finding the cut height h that yields the maximum agreement between the third-party results and VAMO's reference clustering.

Malware Clustering Bailey et al. [133] presented one of the first studies on behavior-based malware clustering. Furthermore, in [133] the authors presented a quantitative analysis of the inconsistency in the labels assigned by different AVs. Bayer et al. [136] introduced a much more scalable way to perform behavior-based malware clustering. In addition, they proposed to validate their clustering results by comparing them against a clustering obtained using a majority voting-based approach over multiple malware family labels assigned to the samples by six different AVs [136]. A similar validation approach was used in [135]. In [137], Hu et al. perform malware clustering using static analysis, instead of behavior-based features, by leveraging function-call graphs, while [139] introduces a system called BitShred that aims to improve scalability in malware clustering systems.

In [140], Li et al. discuss a number of challenges related to the evaluation of results generated by malware clustering systems. In particular, by using plagiarism detection algorithms to measure the similarity between malware samples, they show that a factor contributing to the strong results reported in [136] might be that the 2,658 validation instances selected via majority voting on multiple AVs are simply easy to classify. However, no complete solution is offered on how to perform a better malware clustering validity analysis. Our work is a step forward towards such a solution.

While most malware clustering systems are based on system-level behavior or static-analysis-based features, [138] proposed a malware clustering system that focuses on the network behavior of malware and introduced the concept of AV Label Graph, which we refine and use in this paper in the context of VAMO. It is worth noting that the use of AV Label Graphs in [138] is significantly different from this paper. Previous work did not present a comprehensive malware clustering validity analysis system, and the cohesion and separation validity indexes used in [138] were mainly *internal* validity indexes that required a significant amount of interpretation through manual analysis. On the other hand, VAMO introduces a comprehensive, fully automated malware clustering validity analysis process that can more readily be used to select the parameters of a malware clustering system, or to compare results obtained using different clustering algorithms.

Measuring Inconsistency in AV Labels

In this Section, we aim to quantify the "inconsistency" typical of multiple AV labels that has been *qualitatively* discussed in previous work [135, 136, 138], and analyzed more in details in [133, 145]. Our main goal is to suggest that (semi-) manually creating a mapping between malware family labels and correct the inconsistent (or erroneous) labels, which was required in previous work to perform malware cluster validity analysis (e.g., in [136]), is in fact a fairly difficult task. In addition, we show that in a large number of cases no majority voting-based consensus can be reached. Our results confirm previous findings [133] by using a more recent and much larger malware dataset.

To this end, we performed a number of measurements over a large dataset of AV labels assigned by four different major AV vendors (namely, Symantec, McAffee, Avira, and Trend Micro) to a set of 1,108,289 distinct malware samples. These malware samples were collected from different sources over the course of one entire year, from 2011-01-01 to 2011-12-31 (it is worth noting that we only consider malware samples that were detected as such by at least one out of the four AV scanners). The AVs used to scan the samples were updated daily, and each malware sample was scanned with each AV once a day for 30 days, starting from the day in which the sample was collected. In the following, we will refer to the four AV scanners, in no particular order, as `AV1`, `AV2`, `AV3`, and `AV4`. We intentionally mask the specific AV vendor names, when reporting the results, to avoid controversy (the results we report may be seen as damaging to one or more vendors, due to their low detection rate). After all, we do not intend to establish what vendor performs the best over our malware dataset. Rather, we focus on the inconsistencies in the malware labels, both within a given AV vendor as well as across vendors.

This dataset was kindly provided by a well-known security company.

If an AV scanner `AVi` detected a sample m and assigned it a label on day $d < 30$, the data collector would stop scanning m with `AVi` for the remaining days, but continued scanning the sample with the other AVs until they also assigned a label or $d > 30$.

Table 4.1: AV labels for a dataset of 1,108,289 distinct malware samples.

	AV1	AV2	AV3	AV4
Detected samples	590,341	825,766	702,124	1,030,354
Detection rate (%)	53.3%	74.5%	63.4%	93.0%
Distinct AV labels	20,217	15,138	2,208	175,333
Distinct family labels	3,330	4,729	1,710	3,520
Distinct first variants	20,217	13,851	2,199	51,732

Overview

Table 4.1 summarizes our AV label dataset. As we can see, the detection rate, number of distinct (complete) labels, and the number of distinct malware family labels varies greatly across the different AV scanners. For example, `AV3` assigned a label to 702,124 (63.4%) malware samples, but the number of distinct labels was only 2,208. This means that, in average, the same label was assigned to 317 different samples. This behavior is very different from the other AVs, and in particular from `AV4` for which in average the same label was assigned to (approximately) six samples. In addition, among the 1,108,289 distinct malware samples, only 420,920 (38%) were labeled (i.e., detected) by more than two different AVs. This suggests that because a majority voting approach would require three out of four AVs to agree on the labels (two out of four would only represent a tie), in our example scenario no majority voting-based consensus can be reached on the correct malware family label for at least 38% of the samples. This problem is exacerbated by the fact that even in the cases in which three or more labels are available, the AVs may not agree on the family those samples belong to, as we discuss in Section 4.2

Family Labels

We now focus on malware family names, rather than considering full AV labels. We will consider the malware cluster shown in Table 4.2 as an example, to explain how we derive the malware family names. This malware cluster was obtained using [138]. In Table 4.2, each row represents a malware sample (indexed by the last four bytes of its `MD5` sum), and reports the labels assigned to the sample by three different AVs, namely McAfee (`M`), Avira (`A`), and Trend Micro (`T`).

Table 4.2: Malware Cluster Example with Inconsistent AV Labels.

Short-MD5	McAfee	Avira	Trend Micro
b1b6da81	M=W32/Virut.gen	A=TR/Drop.VB.DU.1	T=PE_VIRUT.XO-1
ec34ca31	M=W32/Virut.gen	A=TR/Drop.VB.DU.1	T=PE_VIRUT.XO-1
c2276216	M=W32/Virut.gen	A=W32/Virut.E.dam	T=PE_VIRUT.NS-4
089ae4f5	M=W32/Virut.gen	A=W32/Virut.AX	T=PE_VIRUT.D-1
8ba552c9	M=W32/Virut.gen	A=TR/Drop.VB.DU.1	T=PE_VIRUT.XO-1
8cb0ab6c	M=W32/Virut.gen	A=WORM/Korgo.U	T=PE_VIRUT.D-4
b0b75f70	M=W32/Virut.gen	A=W32/Virut.X	T=PE_VIRUT.XO-1
a306b4e7	M=W32/Virut.gen	A=W32/Virut.Gen	T=PE_VIRUT.D-1
337a2cf4	M=W32/Virut.gen	A=W32/Virut.Gen	T=PE_VIRUT.D-1
62d18c7e	M=W32/Virut.gen	A=W32/Virut.Gen	T=PE_VIRUT.D-1
8dbca633	M=W32/Virut.gen	A=TR/Drop.VB.DU.1	T=PE_VIRUT.XO-1
ac433383	M=W32/Virut.n	A=W32/Virut.Gen	T=PE_VIRUX.A-3
cae61d9e	M=W32/Virut.gen	A=W32/Virut.X	T=PE_VIRUT.XO-2
7cc795f1	M=W32/Virut.gen	A=W32/Virut.Gen	T=PE_VIRUT.D-1
8de5214b	M=W32/Virut.gen.a	A=W32/Virut.AM	T=PE_VIRUT.XY
4d26cb0a	M=W32/Virut.gen	A=W32/Virut.Gen	T=PE_VIRUT.D-1
9fb75631	M=W32/Virut.n	A=W32/Virut.Gen	T=PE_VIRUX.A-3
229004b9	M=W32/Virut.gen	A=W32/Virut.X	T=PE_VIRUT.XO-1
28a85d8a	M=W32/Virut.gen	A=TR/Drop.VB.DU.1	T=PE_VIRUT.XO-1
663c5f6c	M=W32/Virut.d	A=W32/Virut.Z	T=PE_VIRUT.GEN-
de6f1e00	M=	A=W32/Virut.Gen	T=PE_VIRUT.D-4
1ff43bca	M=	A=W32/Virut.X	T=PE_VIRUT.XO-4
ea580f6d	M=W32/Virut.n	A=W32/Virut.Gen	T=PE_VIRUX.A-3
a844eeff	M=W32/Virut.gen	A=TR/Drop.VB.DU.1	T=PE_VIRUT.XO-1
4f8613fd	M=W32/Virut.gen	A=TR/Drop.VB.DU.1	T=PE_VIRUT.XO-1

To derive the malware family name, we split each label into substrings divided by the '.' symbol, and we extract the first substring. For example, `W32/Virut.gen` becomes `W32/Virut` (Symantec uses a slightly different notation, compared to the other AV vendors. To extract the family label from Symantec's labels, we consider the first two substrings obtained by splitting the labels by the '.' symbol. For example, `W32.Sality.AE` would become `W32.Sality`).

As we can see from Table 4.2, in this case both McAfee and Trend Micro are very consistent, because they label the vast majority of the samples as belonging to the *Virut* malware family, with the exception of two samples that were missed by McAfee and three samples that are labeled as `PE_VIRUX` (rather than `PE_VIRUT`) by Trend Micro. On the other hand, Avira is much less consistent, because it assigned three different family names to the samples (i.e., `TR/Drop`, `W32/Virut`, `WORM/Korgo`).

Table 4.1 reports the total number, per AV, of distinct family names obtained from all labels in our datasets. Also, Table 4.1 reports the total number, per AV, of distinct "first variant" labels, i.e., labels obtained by combining the first two label substrings (the first three, in case of Symantec). Again, there is a relatively large difference between the numbers obtained from different AVs.

To measure the number of common family names per sample across different AVs, we further normalized the family names, for example by cutting the label prefix (e.g., `W32/`, `PE_`, etc.) and reducing all labels to lower case. For example, the first sample in Table 4.2 would be labeled as {`virut`, `drop`, `virut`}. This was done to maximize the number of common family names we could find for a given sample across different AVs. Even after this normalization, we could find a common family label across at least three out of four AVs for only 2.4% of the samples, and a common label across at least two out of four AVs for only 5.6% of the samples. Performing a manual mapping between the labels to mitigate the effect of different "terminology" used by different AVs may improve on these results. However, even after such manual mapping a majority voting-based consensus between the AVs cannot be reached for the vast majority of the samples. This findings are consistent with the experiments conducted in [136], in which a majority voting-based consensus could be reached only for less than 20% of the samples. Therefore, a reference clustering generated via majority voting may miss to represent a large portion of the malware dataset, causing a potential overestimate of the clustering quality, as also suggested in [140].

Validity Indexes

Clustering can be viewed as an *unsupervised learning* process over a dataset for which the complete ground truth is usually not available. Therefore, unlike in supervised learning settings, analyzing the *validity* of the clustering results is intrinsically hard. The assessment of the quality of clustering results often involves the use of subjective criteria of optimality [146], which are typically application specific, and commonly involves extensive manual analysis by domain experts. To aid the clustering validation process, a number of methods and quality indexes have been proposed [15, 147]. Halkidi et al. [15] provide a survey of cluster validity analysis techniques, which aim to evaluate the clustering results to *find the partitioning that best fits the underlying data.*

Three main *cluster validity* approaches are described [15]: (1) *external criteria* evaluate the clustering results by comparing them to a pre-specified structure, or *reference clustering*; (2) *internal criteria* rely solely on quantities derived from the data vectors in the clustered dataset (e.g., using a proximity matrix, and computing quantities such as inter- and intra-cluster distances); (3) *relative criteria* compare clustering results obtained using the same clustering algorithm with different parameter settings, to identify the best parameter configuration.

External validation criteria are particular attractive, because they offer a quantitative way to measure the *level of agreement* between the obtained clustering results and a reference clustering that is considered to be the ground truth [15, 143]. However, the main problem is exactly how to construct the reference clustering in the first place. This is one of the problems we address in this paper: building a reference clustering that can be used for validating the results of malware clustering systems.

Assuming a reference clustering is available, different external validity indexes can be used for measuring the quality of the clustering results. We briefly describe some of them below. Let $\mathcal{M}$ be our dataset, $\mathbf{Rc} = \{Rc_1, .., Rc_s\}$ be the set of s *reference clusters*, and $\mathbf{C} = \{C_1, .., C_n\}$ be our clustering results over $\mathcal{M}$. Given a pair of data samples (m_1, m_2), with $m_1, m_2 \in \mathcal{M}$, we can compute the following quantities:

- a is the number of pairs (m_1, m_2) for which if both samples belong to the same reference cluster Rc_i, they also belong to the same cluster C_j.
- b is the number of pairs (m_1, m_2) for which both samples belong to the same reference cluster Rc_i, but are assigned to two different clusters C_k and C_h.
- c is the number of pairs (m_1, m_2) for which both samples belong to the same cluster C_i, but are assigned to two different reference clusters Rc_k and Rc_h.
- d is the number of pairs (m_1, m_2) for which if the samples belong to two different reference clusters Rc_i and Rc_j, they also belong to different clusters C_l and C_m.

Based on the above definitions, we can compute the following external cluster validity indexes [15]:

- **Rand Statistic**. $RS = \frac{a+d}{a+b+c+d} = \frac{a+d}{|\mathcal{M}|}$
- **Jaccard Coefficient**. $JC = \frac{a}{a+b+c}$
- **Folkes and Mallows Index**. $FM = \frac{a}{\sqrt{(a+b)(a+c)}}$

For all three indexes above, which take values in $[0, 1]$, higher values indicate a closer similarity between the clustering $\mathbf{C}$ and the reference clustering $\mathbf{Rc}$.

The authors of [135, 136], proposed to use different indexes, based on *precision* and *recall*, to measure the level of agreement between behavior-based malware clustering results **C** and a (semi-)manually generated reference clustering **Rc** derived by using majority voting over multiple AV labels. In this setting, precision and recall, and the related $F1$ index, are defined as follows:

- **Precision**. $Prec = 1/n \cdot \sum_{j=1}^{n} \max_{k=1,..,s}(|C_j \cap Rc_k|)$
- **Recall**. $Rec = 1/s \cdot \sum_{k=1}^{s} \max_{j=1,..,n}(|C_j \cap Rc_k|)$
- **F1 Index**. $F1 = 2\frac{Prec \cdot Rrec}{Prec + Rrec}$

In the remainder of this chapter, we will often refer to the external validity indexes defined above.

4.3 System Overview

Figure 4.1 provides a high-level overview of VAMO. We assume that a third-party has employed a malware clustering system, for example one of the systems proposed in [136, 138, 139], to partition a malware dataset **M** into a set of clusters $\mathbf{C} = \{C_1, C_2, .., C_x\}$, with $\bigcup_{i=1}^{x} C_i = \mathbf{M}$. VAMO's objective is to validate the quality of **C** (i.e., the malware clustering results). We now provide a description of VAMO's components shown in Figure 4.1.

AV Label Dataset Given a large *historic archive dataset* of malware samples $\mathcal{A}$ (which is different from **M**), we first collect the set of family labels assigned by multiple AV scanners to each of the malware samples $m_k \in \mathcal{A}$. The resulting AV labels dataset can be represented as a set of tuples $\mathcal{L} = \{(l_{k,1}, l_{k,2}, .., l_{k,\nu})\}_{k=1..n}$, where $l_{k,i}$ is the malware family label assigned by the i-th of ν AV scanners to malware sample m_k, with $k = 1, .., n$, and $n = |\mathcal{A}|$. If an AV scanner misses to detect a malware sample, the related label in the set $\mathcal{L}$ will be assigned a *unique* placeholder "unknown" family label. It is worth noting that the malware dataset $\mathcal{A}$

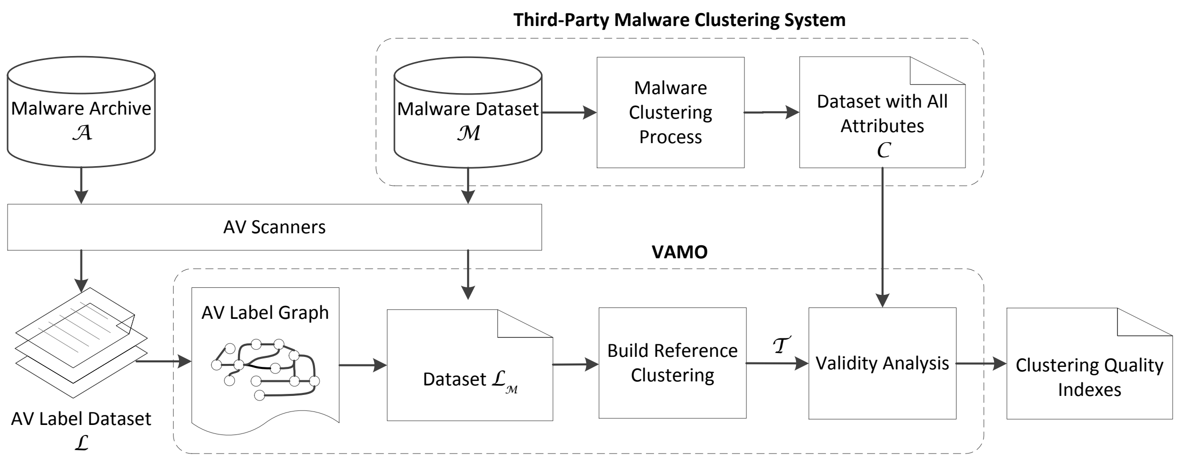

Third-Party Malware Clustering System
Malware Archive $\mathcal{A}$
Malware Dataset $\mathcal{M}$
Malware Clustering Process
Dataset with All Attributes C
AV Scanners
VAMO
AV Label Graph
Dataset $\mathcal{L}_{\mathcal{M}}$
Build Reference Clustering
$\mathcal{T}$
Validity Analysis
Clustering Quality Indexes
AV Label Dataset $\mathcal{L}$

Figure 4.1: VAMO System Overview

need not contain actual executable malware samples. In fact, $\mathcal{A}$ may simply contain a list of hashes (e.g., `md5` or `sha1`) computed by a third party (e.g., the owner of a large malware dataset who cannot share the malware itself) over known malware samples. In this case, the label dataset $\mathcal{L}$ may be obtained by querying a service such as virustotal.com to obtain, for each hash, the related malware family labels from multiple AV scanners.

AV Label Graph VAMO uses the label dataset $\mathcal{L}$ to *learn* an *AV Label Graph* (defined formally in Section 4.4.). Basically, a node in the graph represents a malware family name attributed by a certain AV scanner (or AV, for short) to one or more malware samples in $\mathcal{A}$. For example, assuming the i-th AV assigned the label `family_x` to at least one malware sample, the *AV Label Graph* will contain a node called `AVi_family_x`. Two nodes, say `AVi_family_x` and `AVj_family_y`, will be connected by an undirected edge if there exists at least one malware sample $m_k \in \mathcal{A}$ that has been assigned label `family_x` by the i-th AV, and `family_y` by the j-th AV, respectively. Each edge is assigned a weight that depends on the number of times that the connected nodes (i.e., the connected labels) were assigned to a same malware sample. Notice that if the i-th AV missed to detect a given malware sample, the related missing label will be replace by a label such as `AVi_unknown_U`, where `U` is a unique identifier.

Reference Clustering Similarly to what we did with $\mathcal{A}$, given the malware dataset $\mathbf{M}$ (i.e., the input to the third-party clustering system), we first collect the set of labels assigned by ν different AVs to each of the malware samples $m_k \in \mathbf{M}$, thus obtaining a dataset $\mathcal{L}_M$ consisting of a tuple (or vector) of family labels $L_k = (l_{k,1}, l_{k,2}, .., l_{k,\nu})$ per each sample m_k. At this point, we leverage the previously learned *AV Label Graph* to measure the *dissimilarity* (or distance) between samples in $\mathbf{M}$ according to their malware family labels. Specifically, we measure the distance between two malware samples $m_i, m_j \in \mathbf{M}$ by measuring the distance between their respective label vectors L_i and L_j in the graph. We give a formal definition of label-based distance between malware samples in Section 4.4. At a high level, we compute the distance between two samples m_i, m_j by computing the median among the shortest paths in the *AV Label Graph* between all paris of labels l_k, l_h, with $l_k \in L_i$ and $l_h \in L_j$. This allows us to compute an $r \times r$ distance matrix $\mathbf{D}$, where $r = |\mathbf{M}|$ and element $\mathbf{D}[i, j]$ is the distance between samples m_i, m_j. The final reference clustering is obtained by applying average-linkage hierarchical clustering [146, 147] on the distance matrix $\mathbf{D}$. The result is not an actual partitioning of the malware dataset $\mathbf{M}$. Rather, the reference clustering is represented by a *dendrogram* [147], i.e., a tree-like data structure that expresses the "relationship" between malware samples. Cutting this dendrogram at any particular height would produce a partitioning of $\mathbf{M}$ according to the AV label-based distances (see Section 4.4 for details).

Validity Analysis Let $\mathcal{T}$ be the reference clustering dendrogram output by the previous step. The *Validity Analysis* module takes in input $\mathcal{T}$ and the set of malware clusters $\mathbf{C}$ output by the third-party malware clustering system. At this point, VAMO applies the *external* validity indexes introduced in Section 4.2 to compute the maximum level of agreement between **C** and all possible reference clusterings obtained by cutting $\mathcal{T}$ at different heights. For example, we can compute the maximum Jaccard coefficient $\hat{J}$ between all possible reference clusterings and **C**. The higher $\hat{J}$, the stronger the agreement between **C** and the AV label-based reference clustering.

Effectively, VAMO compares the third-party clustering results **C** to a reference clustering obtained by partitioning the dataset **M** according to the relationships among multiple AV labels learned from the archive malware dataset $\mathcal{A}$. It is worth noting that this process has some similarities with the majority voting-based approach used in previous work. In fact, the effect of the majority voting approach is to group a subset of the malware in **M** according to the labels assigned by multiple AVs to the samples in the very same **M** dataset. VAMO is different because (a) it automatically learns the relationships among malware family labels assigned by different AVs, and does not require any manual (or semi-manual) mapping between them; (b) it introduces a measure of label-based distance between malware samples that is not limited to the cases in which a majority voting-based consensus can be achieved; (c) it enables the computation of well known external validity indexes over the entirety of malware clustering results, rather than focusing only on "easy-to-cluster" subset of the malware dataset. In Section 4.5 we empirically show that building a reference clustering based on the *AV Label Graph* and applying the validity analysis process outlined above outperforms the majority voting-based cluster validation approach proposed in previous work.

4.4 Validity Analysis

In this Section, we provide more details on how VAMO builds the reference clustering by leveraging multiple AV labels, and how the clustering validity indexes are computed to compare third-party malware clustering results to VAMO's reference clustering.

Building a Reference Clustering

As mentioned in Section 4.3, the first step to obtaining the reference clustering is to build an *AV Labels Graph*. This graph expresses the "relationships" between different AV labels, and *automatically learns* the likelihood that different labels from different AVs will be assigned to the same malware sample, based on historic observations.

Assume $\mathbf{M}$ is the malware dataset used as input to a third-party (e.g., behavior-based) malware clustering system, as shown in Figure 4.1. Also, let $\mathcal{A}$ be a large historic malware archive containing, for example, malware samples collected during the past several months, and that $\mathbf{M} \subset \mathcal{A}$ (i.e., $\mathcal{A}$ contains all the "current" samples collected in $\mathbf{M}$, plus a large set of malware samples collected in the past). We define an *AV Label Graph* learned from $\mathcal{A}$ as follows.

Definition 6 - ***AV Label Graph***. *An AV Label Graph is an undirected weighted graph. Given an archive of n malware samples $\mathcal{A} = \{m_i\}_{i=1..n}$, let $\mathcal{L} = \{L_1 = (l_1, .., l_\nu)_1, .., L_n = (l_1, .., l_\nu)_n\}$ be a set of label vectors, where a label vector $L_h = (l_1, .., l_\nu)_h$ is an ordered set of malware family labels assigned by ν different AV scanners to malware $m_h \in \mathcal{A}$. The AV Label Graph $\mathcal{G} = \{V_k, E_{k_1,k_2}\}_{k=1..l}$ is constructed by adding a node V_k for each distinct label $l_k \in \mathcal{L}$. Two nodes V_{k_1} and V_{k_2} are connected by a weighted edge E_{k_1,k_2} if the labels l_{k_1} and l_{k_2} related to the two nodes appear at least once in the same label vector $L_h \in \mathcal{L}$ (that is, if they are both assigned to a malware sample m_h). Each edge E_{k_1,k_2} is assigned a weight $w = 1 - \frac{m}{\max(n_1, n_2)}$, where n_1 is the number of label vectors $L_h \in \mathcal{L}$ that contain l_{k_1}, n_2 is the number of vectors that contain l_{k_2}, and m is equal to the number of vectors containing both l_{k_1} and l_{k_2}.*

For example, assume $\mathcal{A}$ contains all (and only) the samples shown in Table 4.2 (shown in Section 4.2). In this case, the related *AV Label Graph* is shown in Figure 4.2. Notice that in reality $\mathcal{A}$ will typically contain thousands of samples, and that the graph in Figure 4.2 is

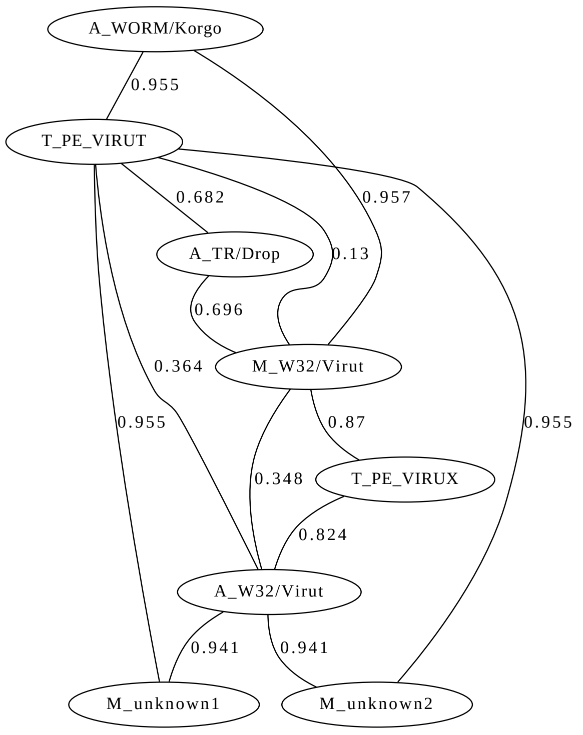


Figure 4.2: AV Label Graph for Clust.1 (see Section 4.2)

reported simply to provide an example of how the *AV Label Graph* is computed. Also, notice that the missing labels were replaced with *unique* "unknown" identifiers.

Once the *AV Label Graph* is computed, we build a reference clustering *dendrogram* as follows (notice that a dendrogram is a tree-like data structure generated by hierarchical clustering [147]). Given any two samples $m_i, m_j \in \mathbf{M}$ we first "map" each sample onto the graph, and then compute the distance $d_{i,j}$ between m_i, m_j on the graph, thus obtaining a distance matrix $\mathbf{D}$ in which $\mathbf{D}[i, j] = d_{i,j}$. A more formal definition of graph-based distance between malware samples is given below.

Definition 7 - ***Graph-based Distance***. *Let $m_i \in \mathbf{M}$ be a malware sample, and $L_i = (l_1, .., l_\nu)_i$ be its label vector. By definition, each label $l_{h,i} \in L_i$ corresponds to a node $V_{h,i}$ in the AV Label Graph, with $h = 1, .., \nu$. Therefore, sample m_i can be mapped to a list $\mathbf{V}_i = (V_{1,i}, .., V_{\nu,i})$ of ν nodes in the graph. Now, let $\mathbf{V}_i$ and $\mathbf{V}_j$ be the lists of nodes related to m_i and m_j, respectively. To compute the distance $d_{i,j}$ between m_i, m_j, we first compute the length of the shortest path p_k among a pair of nodes ($V_{k,i}$,$V_{k,j}$), for each $k = 1, .., \nu$. Then, we compute $d_{i,j}$ as the median among all p_k, with $k = 1, .., \nu$.*

After computing the distance matrix $\mathbf{D}$, we apply average-linkage hierarchical clustering, which outputs a *dedrogram* $\mathcal{T}$ that expresses the "relationship" between the malware samples in $\mathbf{M}$ according to their AV labels. Section 4.2 explains in details how the reference clustering dendrogram $\mathcal{T}$ can be used to validate third-party clustering results.

Computing the Validity Indexes

As mentioned above, by cutting the reference clustering dendrogram $\mathcal{T}$ at a given height h, we obtain an actual partitioning of the dataset $\mathbf{M}$ into a set of reference clusters $\mathbf{Rc} = \{Rc_1, .., Rc_w\}$. Then, the *level of agreement* between $\mathbf{Rc}$ and the third-party clustering results $\mathbf{C} = \{C_1, C_2, .., C_x\}$ can be computed using the external validity indexes introduced in Section 4.2. Naturally, different values of h will produce a different set of reference clusters, and therefore the values of these validity indexes will also differ. Therefore, to decide where exactly to cut the dendrogram $\mathcal{T}$ we proceed as follows. Let $\mathbf{Rc}(h)$ be the set of reference clusters obtained by cutting $\mathcal{T}$ at height h. Also, assume $I(\mathbf{Rc}(h), \mathbf{C})$ is an external validity index computed over the clusterings $\mathbf{Rc}(h)$ and $\mathbf{C}$ (e.g., $I(\cdot)$ could be equal to the Jaccard index, or one of the other indexes outlined in Section 4.2). We then cut $\mathcal{T}$ at height

$h^* = argmax_h\{I(\mathbf{Rc}(h), \mathbf{C})\}$, so that h^* is the cut at which the level of agreement between $\mathbf{C}$ and the VAMO's reference clustering is maximum.

In summary, we perform hierarchical clustering of the malware samples in $\mathbf{M}$ according to similarities in their AV labels by leveraging the previously learned *AV Label Graph*, and then we find the set of reference clusters $\mathbf{Rc}(h^*)$ that *best explains* (or *agrees* with) the third-party clustering results $\mathbf{C}$. This is useful because given two different third-party results $\mathbf{C_1}$ and $\mathbf{C_2}$ (e.g., given by the same behavior-based malware clustering systems configured with different parameter values, or given by different malware clustering systems), VAMO allows us to establish which of them has the *highest level of agreement with the underlying multiple AV labels.*

4.5 Evaluation

VAMO v.s. Majority Voting

In this Section, we present a set of experiments performed in a controlled setting. Our objective is to show that, when faced with noisy AV labels, VAMO outperforms majority voting-based approaches. Namely, in the vast majority of cases VAMO produces an AV label-based reference clustering that better explains (or agrees with) the *true* malware clusters. To this end, we use the following high-level approach. We simulate a controlled dataset of malware samples for which we know exactly what samples should belong to what malware cluster, and first assume that all samples are perfectly (i.e., correctly) labeled by multiple AVs. Then, we gradually introduce more and more noise into the AV labels, thus simulating the inconsistent labeling typical of real-world AVs (see Section 4.2). For each noise increase, we apply both VAMO and a majority voting-based approach to obtain an AV label-based reference clustering, and the obtained results show that VAMO's reference clustering yields validity indexes that offer a higher level of agreement with the *true* malware clusters, compared to using majority voting.

Controlled Datasets

We create a synthetic dataset to simulate a scenario in which we have a historic archive $\mathcal{A}$ consisting of 3,000 distinct malware samples and the related dataset $\mathcal{L}$ of labels assigned by three different AV scanners to each of these 3,000 samples. Furthermore, we create a dataset $\mathbf{M}$ containing 300 distinct samples, with $\mathbf{M} \subset \mathcal{A}$ (i.e., $\mathbf{M}$ is a proper subset of $\mathcal{A}$). Therefore, the label dataset $\mathcal{L}_M$ containing the AV labels for the malware samples in $\mathbf{M}$ can be directly obtained from $\mathcal{L}$ (since $\mathbf{M} \subset \mathcal{A}$, then $\mathcal{L}_M \subset \mathcal{L}$). It is worth noting that we named these datasets following the same terminology that we used in Section 4.3 and in Figure 4.1.

At first, we assume to have perfect knowledge (i.e., perfect *ground truth*) regarding the malware family each sample belongs to. Specifically, we construct the datasets so that the samples in $\mathcal{A}$ (and the related malware labels in $\mathcal{L}$) belong to 15 different malware families, with 200 samples per family, and that each of the three AVs consistently assigns the correct malware family name to the samples in $\mathcal{A}$, and therefore also to the samples in $\mathbf{M}$. In practice, to obtain $\mathbf{M}$ we simply randomly (uniformly) select 300 samples from $\mathcal{A}$. Also, since we know exactly what malware belong to what family, we can precisely partition the dataset $\mathbf{M}$ into a set of 15 malware clusters $\mathbf{C} = \{C_1, C_2, ..., C_s\}$, with $s = 15$.

It is worth noting that in this idealized scenario we also assume the AVs use the very same family names for the malware family labels. In other words, we assume the AVs all agree on using the same terminology or notation. This means that no manual mapping between family names assigned by different AVs is needed, and a majority voting-based approach can be applied directly. This typically does not hold in practice, in which case we would need to obtain the name mapping before being able to apply majority voting. On the other hand, VAMO is agnostic to differences in the terminology that the AVs use to assign malware family names, because VAMO will automatically learn the relationships between different malware family names through the *AV Label Graph* construction, as discussed in Section 4.3 and Section 4.4.

Simulating Inconsistency in the AV Labels

To simulate inconsistency in the AV labels, we proceed as follows. We start from the label dataset $\mathcal{L}$ described above, and we progressively inject more and more noise into the labels. Specifically we inject the following two types of noise:

- **Label Flips** Given a malware $m_k \in \mathcal{A}$, and its label vector $L_k = (l_{1,k}, l_{2,k}, l_{3,k}) \in \mathcal{L}$, with probability p'_f we replace label $l_{\nu,k}$ with a different label $l'_{\nu,k}$ chosen among the 14 other possible malware family labels, where the probability p'_f is a preset "probability of flip".
- **Missing Labels** Similarly, given a malware $m_k \in \mathcal{A}$, and its label vector $L_k = (l_{1,k}, l_{2,k}, l_{3,k}) \in \mathcal{L}$, with probability p'_m we drop label $l_{\nu,k}$ to simulate the case in which the ν-th AV missed to detect m_k, where the probability p'_m is a preset "probability of missed detection".

These two types of noise can affect, with different preselected probabilities, either one, two, or three AVs. To better explain this, let $\mathsf{n} = [p_f, p_m; p_1, p_2, p_3]$ be a "noise vector" whose elements express the following probabilities: p_f is the overall probability that a malware sample m will be affected by a label flip, while p_m is the overall probability that a sample will be affected by a missing label; on the other hand, p_x (with x =1,2, or 3) represents the probability that the noise (through label flips and/or missing labels) will affect exactly x out of the three AVs, for a given malware sample. Notice that $p_1 + p_2 + p_3 = 1$, and $p_f + p_m \leq 1$. Namely, with probability $1 - (p_f + p_m)$ a sample will not be affected by any noise (i.e., the sample remains perfectly labeled).

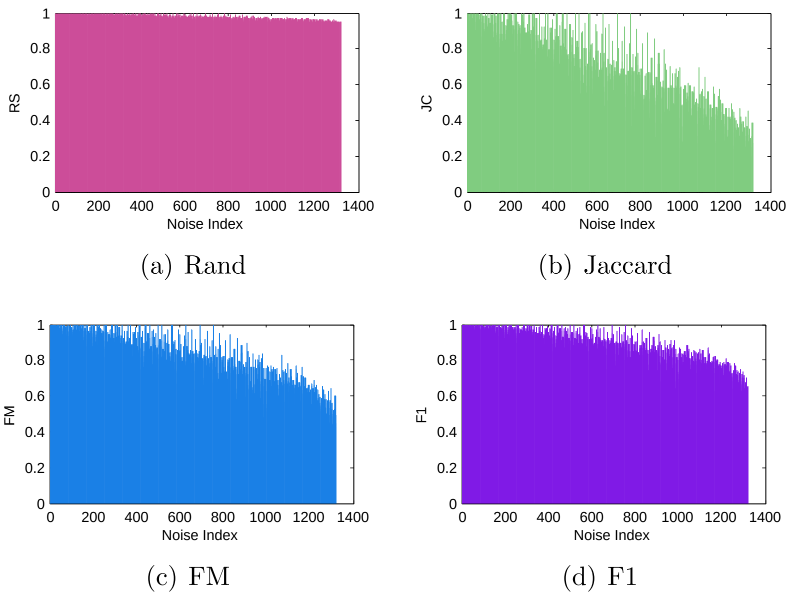
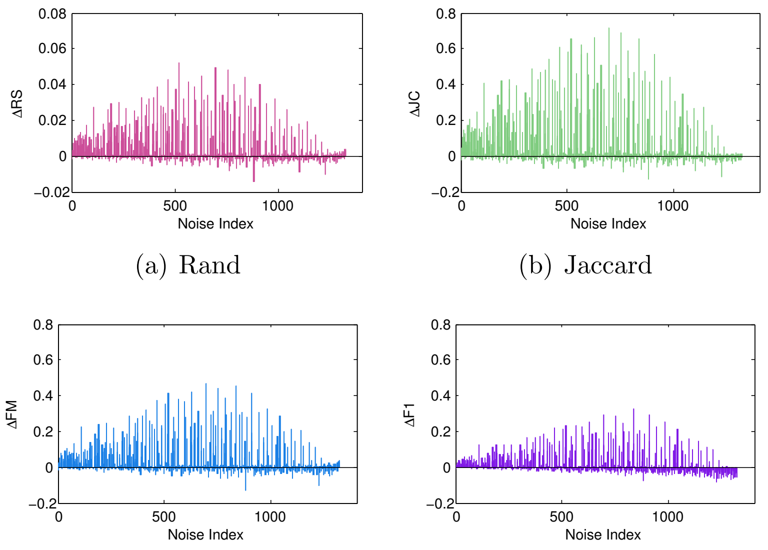


Figure 4.3: VAMO, absolute values of cluster validity indexes.

Building a Reference Clustering via Majority Voting

Intuitively, the majority voting-based approach to construct a reference clustering works as follows. Given a malware sample $m_i \in \mathbf{M}$, and the label vector $L_i = (l_{1,i}, l_{2,i}, l_{3,i}) \in \mathcal{L}_M$ containing the malware family labels assigned to m_i by the three AVs, m_i is assigned to malware cluster R_j if the majority of labels in L_i indicate that m_i belongs to family f_j. If no majority-based consensus can be reached (i.e., the majority of AVs disagree on the family name attributed to m_i), then the sample m_i is assigned to a *singleton cluster*, namely a cluster that contains only m_i. Following this approach, we can partition the dataset $\mathbf{M}$ into a set of majority voting-based reference clusters $\mathbf{Rc}^{MV} = \{Rc_1^{MV}, R_2^{MV}, ..., R_q^{MV}\}$.

Then, given $\mathbf{Rc}^{MV}$ and the *ground truth* clusters $\mathbf{C} = \{C_1, ..., C_s\}$ (which are derived before injecting the noise into the AV labels), we can compute the four external validity indexes described in Section 4.2.



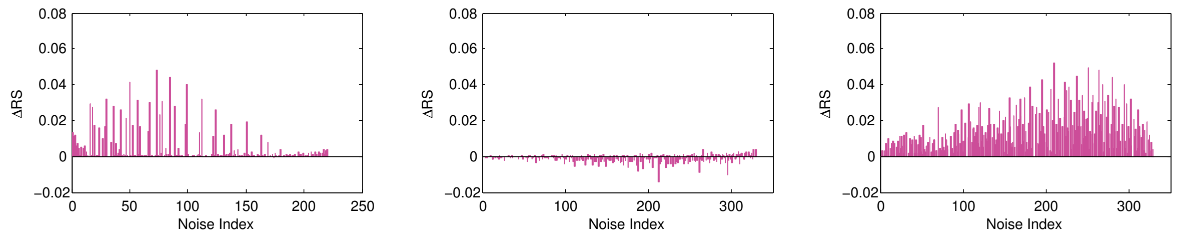
(a) Rand (b) Jaccard (c) FM (d) F1

Figure 4.4: VAMO vs. Majority Voting (index "deltas").

Computing the Validity Indexes

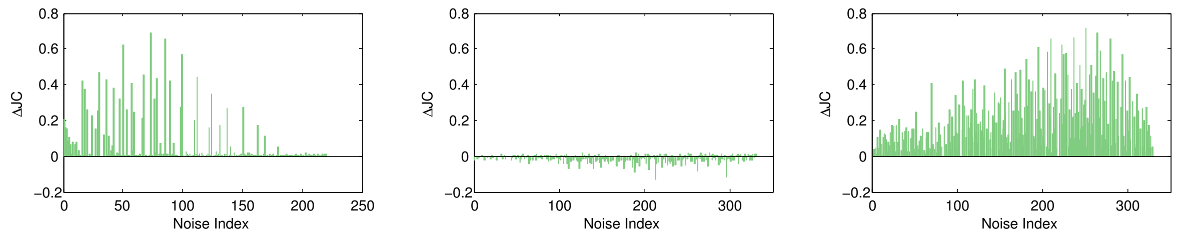
Let n be a particular noise vector, with a given combination of values for the probabilities p_f, p_m, p_1, p_2, and p_3. Applying the noise injection approach described above results in a noisy label dataset $\mathcal{L}(\mathsf{n})$. In turn, if from $\mathcal{L}(\mathsf{n})$ we only consider the labels related to the malware samples in $\mathbf{M}$, we can obtain a (noisy) label dataset $\mathcal{L}_M(\mathsf{n})$ (notice that because $\mathbf{M} \subset \mathcal{A}$, then $\mathcal{L}_M(\mathsf{n}) \subset \mathcal{L}(\mathsf{n})$).

Given $\mathcal{L}(\mathsf{n})$ and $\mathcal{L}_M(\mathsf{n})$, we apply VAMO to compute four validity indexes (see Figure 4.1), thus essentially measuring the *level of agreement* (see Section 4.3) between the reference clustering derived from the *AV Label Graph* learned from $\mathcal{L}(\mathsf{n})$, and the *ground truth* clusters $\mathbf{C} = \{C_1, C_2, ..., C_s\}$ in which $\mathbf{M}$ was originally partitioned (i.e., before any noise was applied). Let $RS^{VAMO}(\mathsf{n})$ be the resulting Rand statistic, $JC^{VAMO}(\mathsf{n})$ be the Jaccard coefficient, $FM^{VAMO}(\mathsf{n})$ be the Folkes-Mallows index, and $F1^{VAMO}(\mathsf{n})$ be the F1 index that combines precision and recall (see Section 4.2).



(a) F & M (> 1 AV) (b) Only Flips (c) Only Missing

Figure 4.5: VAMO vs. Maj. Voting: Rand Statistic.



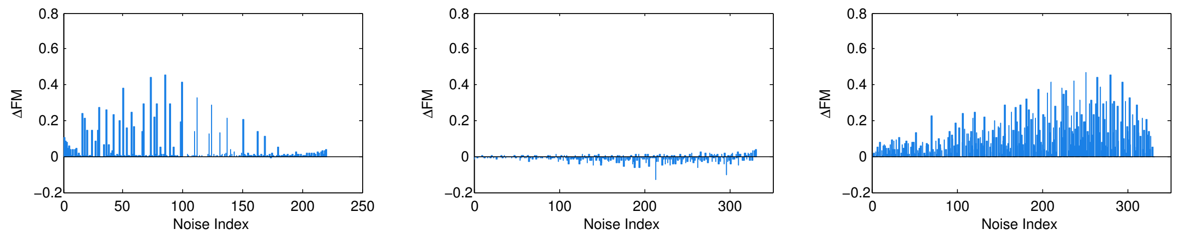
(a) F & M (> 1 AV) (b) Only Flips (c) Only Missing

Figure 4.6: VAMO vs. Maj. Voting: Jaccard Coefficient.

We similarly compute these four external cluster validity indexes by first applying the majority voting-based approach described in Section 4.5 over $\mathbf{M}(\mathsf{n})$ to obtain a reference clustering $\mathbf{Rc}^{MV}(\mathsf{n})$, and then comparing this reference clustering to $\mathbf{C}$. Let $RS^{MV}(\mathsf{n})$ be the resulting Rand statistic, $JC^{MV}(\mathsf{n})$ be the Jaccard coefficient, $FM^{MV}(\mathsf{n})$ be the Folkes-Mallows index, and $F1^{MV}(\mathsf{n})$ be the F1 index. Now, for each value of n we compute the difference between the validity indexes obtained using VAMO and the ones based on the majority voting approach. For example, we compute $\Delta RS(\mathsf{n}) = RS^{VAMO}(\mathsf{n}) - RS^{MV}(\mathsf{n})$, and in a similar way we also compute $\Delta JC(\mathsf{n})$, $\Delta FM(\mathsf{n})$, and $\Delta F1(\mathsf{n})$.

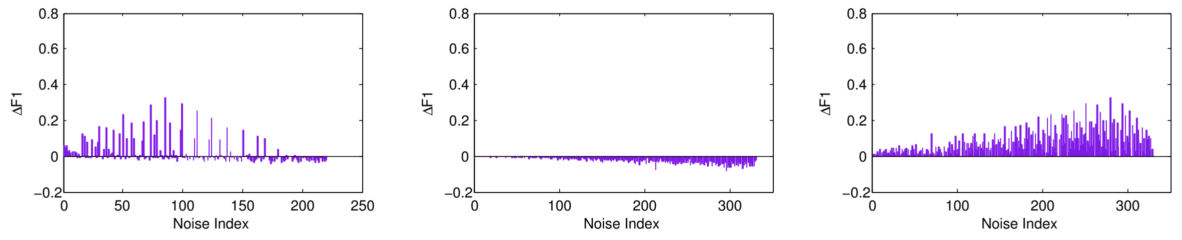
Results

Figure 4.3 reports the absolute values of the cluster validty indexes obtained using VAMO, while Figure 4.4 plots the difference between the four external validity indexes produced by



(a) F & M (> 1 AV) (b) Only Flips (c) Only Missing

Figure 4.7: VAMO vs. Maj. Voting: Folkes-Mallows.



(a) F & M (> 1 AV) (b) Only Flips (c) Only Missing

Figure 4.8: VAMO vs. Maj. Voting: F1 Index.

the comparison between VAMO's results and the majority voting approach, as explained above. In Figure 4.3, the y axis reports the absolute value of the indexes, while in Figure 4.4 it reports the "deltas". In both cases, the x axis is simply the index of the experiment round, with the noise increasing per each experiment . Specifically, we use 1,320 different noise configurations (i.e., different values of the elements of the noise vector n), with the only constraint that $p_f + p_m \leq 0.5$, i.e., at most 50% of the malware samples will be affected by some noise in their AV labels. It is also worth noting that the y axis for ΔRS varies in $[-0.02, 0.08]$, while all other "deltas" graphs have values on the y axis in $[-0.2, 0.8]$.

Figure 4.4(a) shows that the difference between RS^{VAMO} and RS^{MV} are relatively small, and ΔRS varies between -0.02 and 0.06. However, the three remaining validity indexes

The experiment rounds are ordered according to a summary *noise level* computed as $nl = (0.6p_f + 0.4p_m) \cdot (0.1p_1 + 0.3p_2 + 0.6p_3)$.

(Figure 4.4(b) through Figure 4.4(d)) clearly show that VAMO's reference clustering *agrees more closely* with the underlying true clustering **C**, compared to the majority voting-based reference clustering. In fact, in all four indexes the "deltas" are positive for the vast majority of the noise combinations, meaning that the quality indexes obtained by VAMO show a better agreement with the true clustering, compared to the quality indexes obtained via majority voting.

To better analyze the effect of the noisy AV labels, Figure 4.5 through Figure 4.8 present the validity index "deltas" considering all noise vectors n for which: (a) at least two AVs are affected by noise, i.e., $p_1 = 0$; (b) the only type of noise affecting the labels is the "label flips", i.e., $p_m = 0$ (no missing labels, which means that all the AVs assign a malware family label to all samples); (c) the only type of noise is "missing labels", i.e., $p_f = 0$ (no label flips). As we can see, whenever the label noise (or inconsistencies) affects a majority of AVs (case (a)), or when any AV misses to detect some malware samples (case (c)), VAMO clearly outperforms the majority voting-based approach, because VAMO's reference clustering more closely agrees with the true malware clusters. While the "label flips" (i.e., case (b), which simulates the scenario in which AVs assign the incorrect malware family name) have a more negative effect on VAMO because they more heavily affect the edges (and their weights) learned through the *AV Label Graph*, VAMO performs comparably to majority voting, as shown by the very small negative "deltas".

Real-World Application

In this Section, we discuss how VAMO can be applied in practice to assess the quality of the results produced by malware clustering systems. Specifically, we apply VAMO to the results that the behavior-based malware clustering system presented in [136] produced over a real-world malware dataset **M** containing 2,026 distinct malware samples collected in February 2009. To obtained the behavior-based clustering we proceeded as follows. We

l	*clusters*	**Rand Index**	**Jaccard Coeff.**	**Folkes-Mallows**	**F1 Index**
0.10	674	0.8767	0.2086	0.4494	0.7100
0.20	451	0.9172	0.5438	0.7308	0.7918
0.30	313	0.9205	0.5777	0.7482	0.7948
0.31	**301**	**0.9792**	**0.8924**	**0.9434**	*0.8436*
0.32	291	0.9790	0.8916	0.9430	0.8431
0.33	288	0.9759	0.8782	0.9357	**0.8501**
0.34	286	0.9759	0.8782	0.9357	0.8496
0.35	280	0.9758	0.8775	0.9353	0.8479
0.36	274	0.9757	0.8772	0.9352	0.8467
0.37	261	0.9721	0.8614	0.9265	0.8433
0.38	255	0.9721	0.8613	0.9265	0.8424
0.39	248	0.9722	0.8623	0.9270	0.8421
0.40	241	0.9721	0.8617	0.9268	0.8401
0.50	187	0.9585	0.8081	0.8971	0.7937
0.60	142	0.9260	0.7070	0.8366	0.7489
0.70	113	0.8527	0.5614	0.7354	0.7260
0.80	85	0.7789	0.4659	0.6656	0.7124

Table 4.3: Application of VAMO to behavior-based malware clustering results.

provided all malware samples in $\mathbf{M}$ to the authors of [136], who kindly agreed to analyze them and provide us a distance matrix $\mathbf{D}$ containing the pair-wise distances between the samples computed based on their system-level behavioral features. Given, $\mathbf{D}$, we applied *precise* average-linkage hierarchical clustering (this step is slightly different from [136], in which the authors applied an *approximate* hierarchical clustering algorithm), and obtained a dendrogram, which we will refer to as $\mathcal{Y}$ in the following. As usual, the dendrogram $\mathcal{Y}$ can be cut at a given height to obtain a partitioning of dataset $\mathbf{M}$ into a number of malware clusters (see discussion below).

To generate VAMO's *AV Label Graph*, we used a dataset $\mathcal{A}$ consisting of 998,104 real-world distinct malware samples collected between August 2008 and August 2009. All of these 998,104 samples were scanned using four different popular AVs, in a way analogous to the malware dataset we discussed in Section 4.2, to obtain the label dataset $\mathcal{L}$. Each sample in this dataset was assigned at least one AV label. Also, $\mathcal{L}$ contained the labels for most of the 2,026 samples in $\mathbf{M}$. Specifically, $\mathcal{L}$ included at least one label for 1,985 samples in $\mathbf{M}$, while the remaining 41 samples were not represented in $\mathcal{L}$, and therefore remained *unlabeled*.

Taking the labeled dataset $\mathcal{A}$ and the labels for the samples in dataset $\mathbf{M}$ (including the placeholder "unknown" labels for the 41 samples that remained undetected) as input, we applied VAMO to produce a reference clustering, following the procedure outlined in Section 4.3 and Section 4.4. Then, given the dendrogram $\mathcal{Y}$ obtained from the third-party malware clustering system [136], we cut $\mathcal{Y}$ at different heights $l_1, l_2, .., l_n$, thus obtaining a sequence of different clusterings $\mathbf{C}(l_1), \mathbf{C}(l_2), .., \mathbf{C}(l_n)$. For each of these clustering results, we used VAMO to compute a set of validity indexes (see Section 4.4). Table 4.3 summarizes our results. The first column in Table 4.3 reports the value of the hight l at which $\mathcal{Y}$ is cut, while the second column reports the related number of clusters that was obtained from $\mathbf{M}$. For example, by cutting $\mathcal{Y}$ at height $l = 0.5$, $\mathbf{M}$ is partitioned into 187 clusters. The remaining columns represent the values of five different external cluster validity indexes (Section 4.4) measured by comparing the obtained malware clusters to VAMO's reference clustering, as explained in Section 4.4. We varied $l \in [0, 1]$ at steps equal to 0.01 (in practice, we excluded the extreme values $l = 0$ and $l > 0.8$, because they result either into one malware per cluster or into artificially large clusters, respectively). In the interest of space, because the maximum value of the validity indexes is located between $l = 0.3$ and $l = 0.4$, we report the results at steps of 0.01 only within that range.

As we can see from Table 4.3, the best value of the cut l is equal to 0.31, because that is the cut hight at which three out of four external validity indexes express the fact that there is maximum agreement between the behavior-based clusters and the AV labels generated by four different AV scanners. Put another way, VAMO's results indicate that the AV labels provide the best explanation of the underlying malware dataset $\mathbf{M}$ when $\mathbf{M}$ is partitioned into 301 clusters by cutting $\mathcal{Y}$ at $l = 0.31$.

It is worth noting that the F1 index is the only external validity index that is not maximum at $l = 0.31$. However, the value of 0.8436 obtained at $l = 0.31$ is quite close to the maximum value of 0.8502 reached at $l = 0.33$. This result suggests that to find the best configuration parameters for the third-party malware clustering system, it may be better to consider multiple validity indexes, rather than focusing only on analyzing precision and recall (and the related F1 index), as proposed in previous work [136, 139].

4.6 Discussion

Using AV labels to build a reference clustering has some potential limitations, even though the label inconsistencies can be mitigated using VAMO. First, we need to take into account that the features used by the AVs to characterize malware samples and assign them to a given malware family may be different from the features used by a third-party malware clustering system to measure the similarity among samples. For example, AV vendors often base their malware categorization process on features extracted from reverse engineering the malware binaries. On the other hand, behavior-based malware clustering systems leverage features related to the malware's system [136] or network activities [138], for example. Naturally, different features may highlight different types of similarities in the samples. Therefore, while the AV labels clearly represent a valuable point of reference, especially in absence of a more perfect ground truth, the comparison between behavior-based malware clustering results and AV family labels should be taken with a grain of salt. A similar argument is made in [148], in which the authors outline the potential pitfalls of using labeled datasets meant for training and testing of supervised learning algorithms for evaluating the effectiveness of (unsupervised) clustering algorithms. Nonetheless, AV label-based cluster validity analysis, especially when fully automated such as in VAMO, is certainly a valuable tool that can assist malware analysts in the analysis of their malaware clustering results.

Another factor to consider is the fact that AV labels *evolve* in time. That is, a malware sample m assigned by an AV to family f_i at time t_0, may be "renamed" by the same AV as belonging to a different family f_j at a future time $t_1 > t_0$. This is due to the fact that AV signatures are sometimes refined by the AV vendors to reduce possible false positives and more specifically characterize the malware samples (e.g., by assigning a sample previously labeled as "generic" to a more specific malware family). To take this into account, the historic archive of malware labels used by VAMO should be kept updated. This may be done by either periodically re-scanning the malware dataset, or by querying online services such as virustotal.com.

4.7 Conclusion

In this chapter, we presented a novel framework, called VAMO, that provides a fully automated assessment of the quality of malware clustering results. Previous studies propose to evaluate malware clustering results by leveraging the labels assigned to the malware samples by multiple AVs. However, they require a manual mapping between labels assigned by different AV vendors, and are limited to selecting a *reference sub-set* of samples for which an agreement regarding their labels can be reached across a majority of AVs.

Unlike previous work, VAMO does not require a manual mapping between malware family labels output by different AV scanners. Furthermore, VAMO does not discard malware samples for which a majority voting-based consensus cannot be reached. Instead, VAMO explicitly deals with the inconsistencies typical of multiple AV labels to build a more representative reference set. Our evaluation, which includes extensive experiments in a controlled setting and a real-world application, show that VAMO performs better then majority voting-based approaches, and provides a way for malware analysts to automatically assess the quality of their malware clustering results.

Chapter 5

Conclusions and Future Work

In this dissertation, we present two effective machine learning frameworks to tackle challenging problems in the domains of cancer prediction and malware clustering. Our frameworks successfully improve the learning outcomes by using cluster validity analysis to reduce label uncertainty.

5.1 ROMP

We have introduced certain novel features for identifying cancer-causing mutations in the cancer kinome, and we have also utilized various feature selection methods to evaluate our proposed features. Our experiments provided insight into the relationship between different levels of evolutionary conservation and the functional effect of mutations, a consequence of the novel features introduced.

Given the COSMIC dataset with certain degrees of uncertainty, we use multiple classifiers to construct a high performance ensemble classifier to identify the rare oncogenic mutations. We also use EM clustering and our self-invented cluster validity metrics to improve the learning outcomes from the ensemble classifier as well as to identify suspicious mutations in the dataset. Our framework not only successfully identified rare oncogenic mutations, but also provided useful guidance for further experimental follow-up - in particular, an assay of the kinase activity of a few of the mutant forms predicted with the highest confidence could confirm these predictions and indicate the most promising targets for pharmaceutical inhibition.

As our catalog of known causative mutations improves we can further improve our prediction system, to build a more sophisticated model to differentiate between causative and non-causative mutations in cancer. Moreover, our work could be extended to a prediction tool with clinical value, as well as providing a basis for further investigation into the relationship between protein evolution and disease.

5.2 VAMO

We presented a novel framework – VAMO, that provides a fully automated assessment of the quality of malware clustering results. Unlike previous work, VAMO does not require a manual mapping between malware family labels output by different AV scanners. Furthermore, VAMO does not discard malware samples for which a majority voting-based consensus cannot be reached. Instead, VAMO explicitly deals with the inconsistencies typical of multiple AV labels to build a more representative reference set. Our evaluation, which includes extensive experiments in a controlled setting and a real-world application, shows that VAMO performs better than majority voting-based approaches, and provides a way for malware analysts to automatically assess the quality of their malware clustering results.

Using AV labels to build a reference clustering has some potential limitations, even though the label inconsistencies can be mitigated using VAMO. For example, we need to take into account that the features used by the AVs to characterize malware samples and assign them to a given malware family may be different from the features used by a third-party malware clustering system to measure the similarity among samples. However, AV label-based cluster validity analysis, especially when fully automated such as in VAMO, is certainly a valuable tool that can assist malware analysts in the analysis of their malaware clustering results. On another note, the historic archive of malware labels used by VAMO should be kept updated. This may be done by either periodically re-scanning the malware dataset, or by querying online services such as virustotal.com.

VAMO computes the distance between two malware samples by computing the median among the shortest paths in the AV Label Graph between all pairs of labels assigned to the two samples by different AV scanners. It would be interesting if we can apply sophisticated graph theory and/or complex mathematical models to effectively calculate this distance. This may result in better performance of the framework.

Appendix A

ROMP – Supplemental Materials

Supplemental materials for Chapter 3 is compressed as a zip file available at our ECML website. The detailed breakdown of the appendix file is described as following.

- A0. Experiment Design and Corresponding Records. Descriptions of the underlying datasets for each computational experiment, including the folder name of corresponding results in A5.
- A1. Feature Selection Results. The detailed feature selection records of the 5 selection methods with 10-fold cross-validation.
- A2. Ranking of 177 Single-Observation EGFR Mutations in COSMIC v50.
- A3. Ranking of 165 Single-Observation EGFR Mutations in COSMIC v57.
- A4. Ranking of 71 Single-Observation EGFR Mutations in COSMIC v50 that were observed more than once in COSMIC v57.
- A5. Detailed Experiment Records and Results. Detailed experiment results shown in Chapter 3, detailed mappings to experiment settings is described in A0.

http://ecml.uga.edu/dissertation/manchonu/appendix.zip

Appendix B

ROMP – Alternative Ranking and Analyses of EGFR Mutations

Experiments presented in this section are based on the most updated COSMIC dataset, version 57 (partially referencing version 50 for filtering purpose).

These experiments are composed with different combination of training and testing datasets to thoroughly analysis the robustness our combined multiple classifiers method, as well as to provide clearer guidance for the usage of our prediction results for your further analysis.

The detailed settings of the experiments are summarized in Table B.1 The statistics of the dataset for experiment IV is presented in Table 3.20. The detailed records of experiments introduced in this section are given in Supplement A5.

Experiment IV.1 are designed to use the mutations (Kinase domain) in COSMIC v57 dataset whose frequency greater than 1 as positive set, to predict the 106 EGFR mutations that commonly appears in both the COSMIC v50 and v57 dataset with frequency equal to 1.

The positive dataset for training of experiment IV.2, IV.3, and IV.4 includes mutations (Kinase domain) in COSMIC v57 dataset whose frequency greater than 1, by excluding the 71

Table B.1: Experiment Setting IV – Based on COSMIC v57 Dataset

Experiment	Training Set	Testing Set (Prediction)
IV.1	Frequency greater than 1	106 EGFR mutations that appears in both the COSMIC v50 and v57 dataset whose frequency equal to 1
IV.2	Frequency greater than 1, exclude the 71 EGFR mutations whose freq were 1 in v50 but >1 in v 57	71 EGFR mutations that appear only once in COSMIC v50 turn into appear more than once in COSMIC v57
IV.3	Frequency greater than 1, exclude the 71 EGFR mutations whose freq were 1 in v50 but >1 in v 57	177 EGFR mutations in the COSMIC v50 dataset whose frequency equal to 1
IV.4	Frequency greater than 1, exclude the 71 EGFR mutations whose freq were 1 in v50 but >1 in v 57	165 EGFR mutations in the COSMIC v57 dataset whose frequency equal to 1
IV.5	Frequency greater than 1, exclude all EGFR mutations	165 EGFR mutations in the COSMIC v57 dataset whose frequency equal to 1
IV.6	Frequency greater than 1, exclude all EGFR mutations	All (253) EGFR mutations in the COSMIC v57 dataset (includes those in COSMIC v50)

Table B.2: Training Dataset based on COSMIC v57 – Experiment Setting IV.

Experiment	Causative	Non-Causative	Total
IV.1	226	331	557
IV.2/3/4	155	331	486
IV.5/6	138	326	464

Training set sizes obtained from COSMIC v57 and SNP@Domain under several criteria.

EGFR mutations whose frequency was 1 in COSMIC v50 but later on increased to more than 1 in the v57 dataset. Experiment IV.2 is very interesting because testing dataset shows how those 71 EGFR mutations that appear only once in COSMIC v50 turn into appear more than once in COSMIC v57 are classified. This is also a further confirmation to Experiment II.1 (see Section 3.7), in where we use COSMIC v50 FG1 as training dataset. Experiment IV.5 and IV.6 should be given some attention as they help to clear the suspicion of the training set of our previous experiments. The training set of previous experiments include EGFR mutations whose frequency are greater than one, while all the EGFR mutations are completely excluded from the training set in these experiments, and the testing set (prediction) includes 165 EGFR mutations whose frequency equals to one (IV.5) and all EGFR mutations regardless of their frequencies (IV.6). In sum, IV.5 is a subset of IV.6.

B.1 Performance of Individual Classifiers with Experiment Setting IV

Table B.3 shows the F-measure of the 11 trained classifiers with 10-fold cross-validation of Experiments IV. Similarly, the differences of performance between the 11 classifiers are inconspicuous within each experiment, though SVM and tree classifiers perform slightly better in most cases.

Table B.3: Accuracy of 11 Trained Models – Experiment Setting IV (COSMIC v50 and v57 Mixed).

Algorithm	Experiment IV #		
	1	2/3/4	5/6
J48 (Tree)	0.968	**0.971**	**0.981**
Random Forest	0.962	0.963	0.952
NB Tree	0.948	0.949	0.939
Functional Tree	0.969	0.969	0.978
Decision Table	0.931	0.919	0.918
DTNB	0.970	0.967	0.978
LWL(J48+KNN)	0.962	0.965	0.978
Bayes Net	0.959	0.959	0.974
Naive Bayes	0.946	0.947	0.946
SVM	**0.973**	**0.971**	0.976
Neural Network	0.968	0.959	0.961

B.2 Analysis of Single-Occurance Mutations in COSMIC v50 Replicated in COSMIC v57

Experiment IV.2 is another interesting experiment which re-prioritize the five mutations we picked in Section 3.7 with the most updated dataset – COSMIC v57.

As Table B.4 shown, the S-Score of G724S, L861R, and L858Q are all greater than 0.9; the S-Score of T725M and E746K are 0.8170 and 0.7145 respectively. All the S-Score indicates a relatively high probability of being a causative mutation. Although the U-Score of E746K and L861R drops to 0.5960 and 0.5760, they are still being counted as causative by the unsupervised learning module, but not having as high probability as the supervised learning module predicted. However, the interesting observation goes to G724S and L858Q. They both have very high S-Score and U-Score, but they were not classified as causative in the cell-based assays. This indicates that there are additional nuances in the oncogenic mechanisms that remain to be incorporated into our models. This does not necessarily mean that G724S and L858Q are not oncogenic, as these mutations might instead lead to the activation of other phosphorylation sites that we have not analyzed — not to mention that activation of downstream pathways including ERK1/2, AKT and JAK/STAT are controlled by specific phosphorylation sites in EGFR [125–130]. A higher level of EGFR phosphorylation is often linked to malfunctioning of downstream pathways, which in turn leads to abnormal cell growth and proliferation.

Table B.4: Ranks of the 5 Selected Mutations with Experiment Setting IV.2.

Gene	S-Score	U-Score	5S/5U	Position	WT	Mutant
EGFR	0.9295	0.8436	0.8865	724	G	S
EGFR	0.8170	0.7221	0.7696	725	T	M
EGFR	0.7145	0.5760	0.6452	746	E	K
EGFR	0.9085	0.8096	0.8591	858	L	Q
EGFR	0.9137	0.5960	0.7549	861	L	R

B.3 All COSMIC v57 EGFR Mutations Withheld as A Test Set

In experiment IV.5 (see Table 3.19) we wonder if all mutations of a single gene (EGFR in this case) are withheld from the training set, then how would the classifiers trained on mutations in other genes classify the mutations in the missing gene. The testing set of experiment IV.5 is the 165 EGFR mutations in the COSMIC v57 dataset whose frequency equal to 1, therefore it is intuitive that we should select experiment III.2 for comparison as we have already proved the robustness of the training models in III.2. In the 165 single-observation EGFR mutations, we gradually selected the top 50% to 5% (with step size of 5%) ranked mutations from the predictions of experiment III.2 and IV.5, then count the overlaps between them.

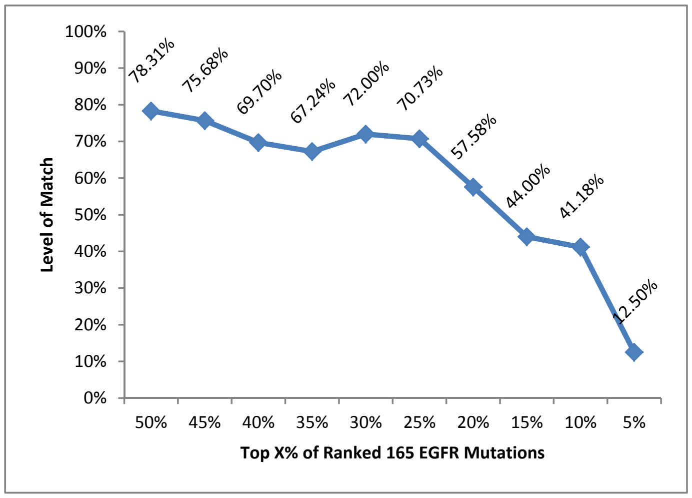


Figure B.1: Level of match between Experiment III.2 and IV.5.

As illustrated in Figure B.1, excluding all mutations that belong to a single gene from the training dataset does not result in a robust classifier. Ideally, the predictions of these two experiments should match fairly closely, but here we see the closest matching occurs at the top 50% threshold, at which point less than 80% of the ranked mutations between two prediction sets are match, and the proportion of matched predictions even drops to 12.5% at the top 5% threshold.

Bibliography

[1] Delano W (2011). The PyMOL Molecular Graphics System. URL http://www.pymol.org/.

[2] Mitchell T (1997) Machine Learning. McGraw-Hill Series in Computer Science. McGraw Hill. URL http://www.cs.cmu.edu/~tom/mlbook.html.

[3] Thrun S, Montemerlo M, Aron A (2006) Probabilistic terrain analysis for high-speed desert driving. In: Proceedings of Robotics: Science and Systems. Philadelphia, USA.

[4] Urmson C, Ragusa C, Ray D, Anhalt J, Bartz D, et al. (2006) A robust approach to high-speed navigation for unrehearsed desert terrain. Journal of Field Robotics 23: 467–508.

[5] U M, Rasheed K (2010) A relative tendency based stock market prediction system. Machine Learning and Applications, Fourth International Conference on 0: 949-953.

[6] Perdisci R, U M (2012) Vamo: towards a fully automated malware clustering validity analysis. In: Proceedings of the 28th Annual Computer Security Applications Conference. New York, NY, USA: ACM, ACSAC '12, pp. 329–338. doi: 10.1145/2420950.2420999. URL http://doi.acm.org/10.1145/2420950.2420999.

[7] Dietterich TG (1997) Machine-learning research – four current directions. AI MAGAZINE 18: 97–136.

[8] Breiman L (1996) Bagging predictors. Mach Learn 24: 123–140.

[9] Schapire RE (1990) The strength of weak learnability. Mach Learn 5: 197–227.

[10] Wolpert DH (1992) Stacked generalization. Neural Networks 5: 241-259.

[11] Seewald A, Fuernkranz J (2001) An evaluation of grading classifiers. In: et al FH, editor, Advances in Intelligent Data Analysis: 4th International Conference. Berlin/Heidelberg/New York/Tokyo: Springer, pp. 115-124.

[12] Hall MA (1999) Correlation-based feature selection for machine learning. Technical report, Departmnet of Computer Science, The University of Waikato.

[13] Kohavi R, Shoham Y, Friedman J, Nilsson N (1995). Wrappers for performance enhancement and oblivious decision graphs.

[14] Kohavi R, John GH (1997) Wrappers for feature subset selection. Artif Intell 97: 273–324.

[15] Halkidi M, Batistakis Y, Vazirgiannis M (2001) On clustering validation techniques. J Intell Inf Syst 17: 107–145.

[16] Hanahan D, Weinberg RA (2000) The Hallmarks of Cancer. Cell 100: 57–70.

[17] Bamford S, Dawson E, Forbes S, Clements J, Pettett R, et al. (2004) The COSMIC (Catalogue of Somatic Mutations in Cancer) database and website. British journal of cancer 91: 355–8.

[18] Futreal PA, Coin L, Marshall M, Down T, Hubbard T, et al. (2004) A census of human cancer genes. Nature reviews Cancer 4: 177–83.

[19] Network TCGAR (2008) Comprehensive genomic characterization defines human glioblastoma genes and core pathways. Nature 455: 1061–8.

[20] Hudson TJ, Anderson W, Artez A, Barker AD, Bell C, et al. (2010) International network of cancer genome projects. Nature 464: 993–8.

[21] Greenman C, Stephens P, Smith R, Dalgliesh GL, Hunter C, et al. (2007) Patterns of somatic mutation in human cancer genomes. Nature 446: 153–8.

[22] Chin L, Gray JW (2008) Translating insights from the cancer genome into clinical practice. Nature 452: 553–63.

[23] Chin L, Andersen JN, Futreal PA (2011) Cancer genomics: from discovery science to personalized medicine. Nature medicine 17: 297–303.

[24] Davies H, Bignell GR, Cox C, Stephens P, Edkins S, et al. (2002) Mutations of the BRAF gene in human cancer. Nature 417: 949–54.

[25] Brose MS, Volpe P, Feldman M, Kumar M, Rishi I, et al. (2002) BRAF and RAS mutations in human lung cancer and melanoma. Cancer research 62: 6997–7000.

[26] Puente XS, Pinyol M, Quesada V, Conde L, Ordóñez GR, et al. (2011) Whole-genome sequencing identifies recurrent mutations in chronic lymphocytic leukaemia. Nature 475: 101–5.

[27] Paez JG, Jänne Pa, Lee JC, Tracy S, Greulich H, et al. (2004) EGFR mutations in lung cancer: correlation with clinical response to gefitinib therapy. Science (New York, NY) 304: 1497–500.

[28] Shigematsu H, Gazdar AF (2006) Somatic mutations of epidermal growth factor receptor signaling pathway in lung cancers. International journal of cancer Journal international du cancer 118: 257–62.

[29] Stephens P, Edkins S, Davies H, Greenman C, Cox C, et al. (2005) A screen of the complete protein kinase gene family identifies diverse patterns of somatic mutations in human breast cancer. Nature genetics 37: 590–2.

[30] Stratton MR, Campbell PJ, Futreal PA (2009) The cancer genome. Nature 458: 719–24.

[31] Wood LD, Parsons DW, Jones S, Lin J, Sjöblom T, et al. (2007) The genomic landscapes of human breast and colorectal cancers. Science (New York, NY) 318: 1108–13.

[32] Yue P, Melamud E, Moult J (2006) SNPs3D: Candidate gene and SNP selection for association studies. BMC Bioinformatics 7: 166.

[33] Shi Z, Moult J (2011) Structural and functional impact of cancer-related missense somatic mutations. Journal of molecular biology 413: 495–512.

[34] Hashimoto K, Rogozin IB, Panchenko AR (2012) Oncogenic potential is related to activating effect of cancer single and double somatic mutations in receptor tyrosine kinases. Human mutation 33: 1566–75.

[35] Zhang Z, Witham S, Petukh M, Moroy G, Miteva M, et al. (2013) A rational free energy-based approach to understanding and targeting disease-causing missense mutations. Journal of the American Medical Informatics Association : JAMIA .

[36] Liu KL, Wong TT, Xie G, Hengartner NW (2009) Improving Naive Bayesian Classifier for Metagenomic Reads Assignment. In: International Conference on Bioinformatics & Computational Biology (BIOCOMP 2009). pp. 259–264.

[37] Hoff KJ, Lingner T, Meinicke P, Tech M (2009) Orphelia: predicting genes in metagenomic sequencing reads. Nucleic Acids Research 37: W101–W105.

[38] Yooseph S, Li W, Sutton G (2008) Gene identification and protein classification in microbial metagenomic sequence data via incremental clustering. BMC Bioinformatics 9: 182.

[39] Mahamuda V, U M, Rasheed K (2010) Application of Machine Learning Algorithms for Binning Metagenomic Data. In: Arabnia HR, Tran QN, Chang R, He M, Marsh A, et al., editors, The 2010 International Conference on Bioinformatics & Computational Biology (BIOCOMP'10). Las Vegas, USA: CSREA Press (ISBN:1-60132-132-5), pp. 68–74. URL http://www.world-academy-of-science.org/worldcomp10/ws/conferences/biocomp10.

[40] U M, Mahamuda V, Rasheed K (2010) On the Scalability of Supervised Learners in Metagenomics. In: 2010 Ninth International Conference on Machine Learning and Applications. Washington, D.C., USA: IEEE (ISBN: 978-0-7695-4300-0), pp. 803–807. URL http://www.computer.org/portal/web/csdl/doi/10.1109/ICMLA.2010.123.

[41] Ng PC, Henikoff S (2002) Accounting for Human Polymorphisms Predicted to Affect Protein Function. Genome Research 12: 436–446.

[42] Ramensky V, Bork P, Sunyaev S (2002) Human non-synonymous SNPs: server and survey. Nucleic acids research 30: 3894–900.

[43] Thomas PD, Kejariwal A (2004) Coding single-nucleotide polymorphisms associated with complex vs. Mendelian disease: Evolutionary evidence for differences in molecular effects. Proceedings of the National Academy of Sciences of the United States of America 101: 15398–15403.

[44] Ferrer-Costa C, Gelpí JL, Zamakola L, Parraga I, de la Cruz X, et al. (2005) PMUT: a web-based tool for the annotation of pathological mutations on proteins. Bioinformatics (Oxford, England) 21: 3176–8.

[45] Kaminker JS, Zhang Y, Waugh A, Haverty PM, Peters B, et al. (2007) Distinguishing cancer-associated missense mutations from common polymorphisms. Cancer research 67: 465–73.

[46] Torkamani A, Schork NJ (2007) Accurate prediction of deleterious protein kinase polymorphisms. Bioinformatics (Oxford, England) 23: 2918–25.

[47] Torkamani A, Schork NJ (2008) Prediction of cancer driver mutations in protein kinases. Cancer research 68: 1675–82.

[48] Torkamani A, Schork NJ (2009) Identification of rare cancer driver mutations by network reconstruction. Genome research 19: 1570–8.

[49] Carter H, Chen S, Isik L, Tyekucheva S, Velculescu VE, et al. (2009) Cancer-specific high-throughput annotation of somatic mutations: computational prediction of driver missense mutations. Cancer research 69: 6660–7.

[50] Torkamani A, Kannan N, Taylor SS, Schork NJ (2008) Congenital disease SNPs target lineage specific structural elements in protein kinases. Proceedings of the National Academy of Sciences of the United States of America 105: 9011–6.

[51] Izarzugaza JMG, Hopcroft LEM, Baresic A, Orengo Ca, Martin ACR, et al. (2011) Characterization of pathogenic germline mutations in human protein kinases. BMC bioinformatics 12 Suppl 4: S1.

[52] Izarzugaza JM, Del Pozo A, Vazquez M, Valencia A (2012) Prioritization of pathogenic mutations in the protein kinase superfamily. BMC genomics 13 Suppl 4: S3.

[53] Cargill M, Altshuler D, Ireland J, Sklar P, Ardlie K, et al. (1999) Characterization of single-nucleotide polymorphisms in coding regions of human genes. Nature genetics 22: 231–8.

[54] Wang Z, Moult J (2001) SNPs, protein structure, and disease. Human Mutation 17: 263–270.

[55] Chasman D, Adams RM (2001) Predicting the functional consequences of non-synonymous single nucleotide polymorphisms: structure-based assessment of amino acid variation. Journal of molecular biology 307: 683–706.

[56] Saunders C (2002) Evaluation of Structural and Evolutionary Contributions to Deleterious Mutation Prediction. Journal of Molecular Biology 322: 891–901.

[57] Ng PC, Henikoff S (2001) Predicting Deleterious Amino Acid Substitutions. Genome Research : 863–874.

[58] Sunyaev S, Ramensky V, Koch I, Lathe W, Kondrashov aS, et al. (2001) Prediction of deleterious human alleles. Human molecular genetics 10: 591–7.

[59] Krishnan V, Westhead D (2003) A comparative study of machine-learning methods to predict the effects of single nucleotide polymorphisms on protein function. Bioinformatics 19: 2199–2209.

[60] Quinlan JR (1986) Induction of decision trees. Machine Learning 1: 81–106.

[61] Quinlan JR (1993) C4.5: Programs for Machine Learning. Morgan Kaufmann series in {M}achine {L}earning. Morgan Kaufmann, 302 pp. URL http://books.google.com/books?hl=en\&lr=\&id=HExncpjbYroC\&oi=fnd\&pg=PR7\&dq=c+4.5+quinlan\&ots=nJscbRt4Xm\&sig=SCckYx-NOoLnCWMTTLjwek5YFXM.

[62] Platt JC (1999) Fast training of support vector machines using sequential minimal optimization. Advances in Kernel Methods 12: 185–208.

[63] Keerthi SS, Shevade SK, Bhattacharyya C, Murthy KRK (2001) Improvements to Platt's SMO Algorithm for SVM Classifier Design. Neural Computation 13: 637–649.

[64] Cai Z, Tsung EF, Marinescu VD, Ramoni MF, Riva A, et al. (2004) Bayesian approach to discovering pathogenic SNPs in conserved protein domains. Human mutation 24: 178–84.

[65] Jensen FV (1996) An Introduction to Bayesian Networks, volume 39. UCL Press, 178 pp. doi:10.2307/1271143. URL http://www.jstor.org/stable/1271143?origin=crossref.

[66] Karchin R, Diekhans M, Kelly L, Thomas DJ, Pieper U, et al. (2005) LS-SNP: large-scale annotation of coding non-synonymous SNPs based on multiple information sources. Bioinformatics (Oxford, England) 21: 2814–20.

[67] Witten IH, Frank E, Hall MA (2011) Data Mining: Practical Machine Learning Tools and Techniques. Amsterdam: Morgan Kaufmann, 3 edition. URL http://www.sciencedirect.com/science/book/9780123748560.

[68] Breiman L (2001) Random Forests. Machine Learning 45: 5–32.

[69] Bonetta L (2005) Going on a cancer gene hunt. Cell 123: 735–737.

[70] Sherry ST, Ward MH, Kholodov M, Baker J, Phan L, et al. (2001) dbSNP: the NCBI database of genetic variation. Nucleic acids research 29: 308–11.

[71] Lynch TJ, Bell DW, Sordella R, Gurubhagavatula S, Okimoto RA, et al. (2004) Activating mutations in the epidermal growth factor receptor underlying responsiveness of non-small-cell lung cancer to gefitinib. The New England journal of medicine 350: 2129–39.

[72] Bromberg Y, Yachdav G, Rost B (2008) SNAP predicts effect of mutations on protein function. Bioinformatics 24: 2397–2398.

[73] Carter H, Samayoa J, Hruban RH, Karchin R (2010) Prioritization of driver mutations in pancreatic cancer using cancer-specific high-throughput annotation of somatic mutations (CHASM). Cancer Biology & Therapy 10: 582–587.

[74] Adzhubei Ia, Schmidt S, Peshkin L, Ramensky VE, Gerasimova A, et al. (2010) A method and server for predicting damaging missense mutations. Nature methods 7: 248–9.

[75] Masso M, Vaisman II (2010) Knowledge-based computational mutagenesis for predicting the disease potential of human non-synonymous single nucleotide polymorphisms. Journal of theoretical biology 266: 560–8.

[76] Capriotti E, Altman RB (2011) A new disease-specific machine learning approach for the prediction of cancer-causing missense variants. Genomics 98: 310–317.

[77] Dobson RJ, Munroe PB, Caulfield MJ, Saqi MA (2006) Predicting deleterious nsSNPs: an analysis of sequence and structural attributes. BMC bioinformatics 7: 217.

[78] A Asuncion DN (2007). UCI machine learning repository. URL http://www.ics.uci.edu/$\sim$mlearn/{MLR}epository.html.

[79] Aha DW (1991). UCI machine learning repository - tic-tac-toe dataset. URL http://archive.ics.uci.edu/ml/datasets/Tic-Tac-Toe+Endgame.

[80] Mangasarian OL, Wolberg WH (1990) Cancer diagnosis via linear programming 23: 1–18.

[81] Wolberg DWH (1995). UCI machine learning repository - wisconsin breast (diagnostic) cancer dataset. URL http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic).

[82] Yeh IC, Yang KJ, Ting TM (2009) Knowledge discovery on RFM model using Bernoulli sequence. Expert Systems with Applications 36: 5866–5871.

[83] Dang XT, Hirose O, Saethang T, Tran VA, Nguyen LAT, et al. (2013) A novel over-sampling method and its application to mirna prediction .

[84] Aha DW, Clark P, Salzberg S, Blix G, et al. (1991) Incremental constructive induction: An instance-based approach .

[85] Zheng Z (1996) Constructing new attributes for decision tree learning. Ph.D. thesis, Citeseer.

[86] Mehrotra KG, Ozgencil NE, McCracken N (2007) Squeezing the last drop: Cluster-based classification algorithm. Statistics & probability letters 77: 1288–1299.

[87] Ph.D. thesis.

[88] Muhic I (2013) Fuzzy analysis of breast cancer disease using fuzzy c-means and pattern recognition. SouthEast Europe Journal of Soft Computing 2.

[89] Ghazavi SN, Liao TW (2008) Medical data mining by fuzzy modeling with selected features. Artificial Intelligence in Medicine 43: 195–206.

[90] Nguyen C, Wang Y, Nguyen HN (2013) Random forest classifier combined with feature selection for breast cancer diagnosis and prognostic .

[91] Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2011) Smote: synthetic minority over-sampling technique. arXiv preprint arXiv:11061813 .

[92] Forbes Sa, Tang G, Bindal N, Bamford S, Dawson E, et al. (2010) COSMIC (the Catalogue of Somatic Mutations in Cancer): a resource to investigate acquired mutations in human cancer. Nucleic acids research 38: D652–7.

[93] Manning G, Plowman GD, Hunter T, Sudarsanam S (2002) Evolution of protein kinase signaling from yeast to man. Trends in biochemical sciences 27: 514–20.

[94] Gosal G, Kochut KJ, Kannan N (2011) ProKinO: An Ontology for Integrative Analysis of Protein Kinases in Cancer. PLoS ONE 6: e28782.

[95] Han A, Kang HJ, Cho Y, Lee S, Kim YJ, et al. (2006) SNP@Domain: a web resource of single nucleotide polymorphisms (SNPs) within protein domain structures and sequences. Nucleic acids research 34: W642–4.

[96] Rice P, Longden I, Bleasby A (2000) EMBOSS: The European Molecular Biology Open Software Suite. Trends in Genetics 16: 276–277.

[97] Magrane M, Consortium U (2011) UniProt Knowledgebase: a hub of integrated protein data. Database : the journal of biological databases and curation 2011: bar009.

[98] Suzek BE, Huang H, McGarvey P, Mazumder R, Wu CH (2007) UniRef: comprehensive and non-redundant UniProt reference clusters. Bioinformatics (Oxford, England) 23: 1282–8.

[99] Stehr H, Jang SHJ, Duarte JM, Wierling C, Lehrach H, et al. (2011) The structural impact of cancer-associated missense mutations in oncogenes and tumor suppressors. Molecular cancer 10: 54.

[100] Manning G, Whyte DB, Martinez R, Hunter T, Sudarsanam S (2002) The protein kinase complement of the human genome. Science 298: 1912–34.

[101] Neuwald AF (2009) Rapid detection, classification and accurate alignment of up to a million or more related protein sequences. Bioinformatics 25: 1869–1875.

[102] Eddy SR (2009) A new generation of homology search tools based on probabilistic inference. International Conference on Genome Informatics 23: 205–211.

[103] Henikoff S, Henikoff JG (1992) Amino acid substitution matrices from protein blocks. Proceedings of the National Academy of Sciences of the United States of America 89: 10915–10919.

[104] Hanks SK, Hunter T (1995) Protein kinases 6. The eukaryotic protein kinase superfamily: kinase (catalytic) domain structure and classification. FASEB journal : official publication of the Federation of American Societies for Experimental Biology 9: 576–96.

Key: Hanks1995

Annotation: Describes - 12 subdomains, w/ alignment - AGC, CAMK, CMGC and TK groups; STE is explicitly classified as "Other" - Names families, representative genes from vertebrates, fly, nematode, yeasts, dicty, sea hare - Mentions eukaryotic-like protein kinases (ELKs), e.g. pknA - Name "HRDLKxxN", "DFG", "APE" motifs

[105] Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. The Journal of Machine Learning Research 3: 1157–1182.

[106] Holte RC (1993) Very Simple Classification Rules Perform Well on Most Commonly Used Datasets. Machine Learning 11: 63–91.

[107] Kira K, Rendell LA (1992) A Practical Approach to Feature Selection. In: International Conference on Machine Learning. pp. 249–256.

[108] Hansen LK, Salamon P (1990) Neural network ensembles. IEEE Trans Pattern Anal Mach Intell 12: 993–1001.

[109] Kohavi R (1996) Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid. In: Han ES, W J, editors, Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. Menlo Park, USA: AAAI Press, volume 7, pp. 202–207. URL http://www.aaai.org/Library/KDD/1996/kdd96-033.php.

[110] Gama J (2004) Functional Trees. Machine Learning 55: 219–250.

[111] Kohavi R (1995) The Power of Decision Tables. In: Lavrač N, Wrobel S, editors, Lecture Notes in Computer Science. Springer-Verlag, volume 912 of *Lecture Notes in Artificial Intelligence*, pp. 174–189. doi:10.1.1.49.4576. URL http://www.springerlink.com/index/p5n736u105315054.pdf.

[112] Hall M, Frank E (2008) Combining Naive Bayes and Decision Tables. Intelligence : 2–3.

[113] Atkeson CG, Moore AW, Schaal S (1997) Locally Weighted Learning. Artificial Intelligence Review 11: 11–73.

[114] John GH, Langley P (1995) Estimating continuous distributions in Bayesian classifiers. Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence 1: 338–345.

[115] Landgrebe T, Pacl P, Tax DMJ, Verzakov S, Duin RPW (2004) Cost-Based Classifier Evaluation for Imbalanced Problems. Lecture Notes in Computer Science 3138: 762–770.

[116] Kohavi R (1995) A study of cross-validation and bootstrap for accuracy estimation and model selection. International Joint Conference on Artificial Intelligence 14: 1137–1143.

[117] Crowther P, Cox R (2005) A method for optimal division of data sets for use in neural networks. In: Knowledge-Based Intelligent Information and Engineering Systems. Springer, volume 20, pp. 1–7. URL http://www.springerlink.com/index/7UDXVWY47528GUA4.pdf.

[118] Dietterich TG (2000) Ensemble Methods in Machine Learning. In: Proceedings of the First International Workshop on Multiple Classifier Systems. London, UK, UK: Springer-Verlag, MCS '00, pp. 1–15. URL http://dl.acm.org/citation.cfm?id=648054.743935.

[119] Chen YR, Fu YN, Lin CH, Yang ST, Hu SF, et al. (2006) Distinctive activation patterns in constitutively active and gefitinib-sensitive EGFR mutants. Oncogene 25: 1205–15.

[120] Daub H, Olsen JV, Bairlein M, Gnad F, Oppermann FS, et al. (2008) Kinase-selective enrichment enables quantitative phosphoproteomics of the kinome across the cell cycle. Molecular cell 31: 438–48.

[121] Zhang G, Fang B, Liu RZ, Lin H, Kinose F, et al. (2011) Mass spectrometry mapping of epidermal growth factor receptor phosphorylation related to oncogenic mutations and tyrosine kinase inhibitor sensitivity. Journal of proteome research 10: 305–19.

[122] Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the em algorithm. JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B 39: 1–38.

[123] Do CB, Batzoglou S (2008) What is the expectation maximization algorithm? Nature biotechnology 26: 897–9.

[124] Celeux G, Govaert G (1992) A classification em algorithm for clustering and two stochastic versions. Comput Stat Data Anal 14: 315–332.

[125] Lowenstein EJ, Daly RJ, Batzer aG, Li W, Margolis B, et al. (1992) The SH2 and SH3 domain-containing protein GRB2 links receptor tyrosine kinases to ras signaling. Cell 70: 431–42.

[126] Kloth MT, Catling AD, Silva CM (2002) Novel activation of STAT5b in response to epidermal growth factor. The Journal of Biological Chemistry 277: 8693–701.

[127] Andl CD, Mizushima T, Oyama K, Bowser M, Nakagawa H, et al. (2004) EGFR-induced cell migration is mediated predominantly by the JAK-STAT pathway in primary esophageal keratinocytes. American journal of physiology Gastrointestinal and liver physiology 287: G1227–37.

[128] Johnson GL, Dohlman HG, Graves LM (2005) MAPK kinase kinases (MKKKs) as a target class for small-molecule inhibition to modulate signaling networks and gene expression. Current opinion in chemical biology 9: 325–31.

[129] Singh AB, Harris RC (2005) Autocrine, paracrine and juxtacrine signaling by EGFR ligands. Cellular signalling 17: 1183–93.

[130] Huang Y, Chang Y (2011) Epidermal Growth Factor Receptor (EGFR) Phosphorylation, Signaling and Trafficking in Prostate Cancer. In: Spiess PE, editor, Prostate Cancer - From Bench to Bedside, InTech, volume 4, chapter 8. URL http://www.intechopen.com/books/prostate-cancer-from-bench-to-bedside/epidermal-growth-factor-receptor-egfr-phosphorylation-signaling-and-trafficking-in-prostate-cancer.

[131] Guo F, Ferrie P, Chiueh T (2008) A study of the packer problem and its solutions. In: Recent Advances in Intrusion Detection.

[132] Symantec (2011). Symantec internet security threat report, trends for 2010.

[133] Bailey M, Oberheide J, Andersen J, Mao ZM, Jahanian F, et al. (2007) Automated classification and analysis of internet malware. In: Recent Advances in Intrusion Detection.

[134] Christodorescu M, Jha S, Kruegel C (2007) Mining specifications of malicious behavior. In: ACM SIGSOFT symposium on the foundations of software engineering. ESEC-FSE '07.

[135] Rieck K, Trinius P, Willems C, Holz T (2011) Automatic analysis of malware behavior using machine learning. J Comput Secur 19: 639–668.

[136] Bayer U, Milani Comparetti P, Hlauschek C, Kruegel C, Kirda E (2009) Scalable, behavior-based malware clustering. In: Network and Distributed System Security Symposium.

[137] Hu X, Chiueh Tc, Shin KG (2009) Large-scale malware indexing using function-call graphs. In: Proceedings of the 16th ACM conference on Computer and communications security. CCS '09.

[138] Perdisci R, Lee W, Feamster N (2010) Behavioral clustering of http-based malware and signature generation using malicious network traces. In: Proceedings of the 7th USENIX Symposium on Networked Systems Design and Implementation. NSDI'10.

[139] Jang J, Brumley D, Venkataraman S (2011) Bitshred: feature hashing malware for scalable triage and semantic analysis. In: Proceedings of the 18th ACM conference on Computer and communications security. CCS '11.

[140] Li P, Liu L, Gao D, Reiter MK (2010) On challenges in evaluating malware clustering. In: Proceedings of the 13th international conference on Recent advances in intrusion detection. RAID'10.

[141] Rendn E, Abundez I, Arizmendi A, Quiroz EM (2011) Internal versus external cluster validation indexes. universitypressorguk 5.

[142] Meilă M (2007) Comparing clusterings—an information based distance. J Multivar Anal 98: 873–895.

[143] Pfitzner D, Leibbrandt R, Powers D (2009) Characterization and evaluation of similarity measures for pairs of clusterings. Knowl Inf Syst 19: 361–394.

[144] Fowlkes EB, Mallows CL (1983) A method for comparing two hierarchical clusterings. Journal of the American Statistical Association 78: 553–569.

[145] Maggi F, Bellini A, Salvaneschi G, Zanero S (2011) Finding non-trivial malware naming inconsistencies. In: International Conference on Information Systems Security. ICISS'11.

[146] Jain AK, Murty MN, Flynn PJ (1999) Data clustering: a review. ACM Comput Surv 31: 264–323.

[147] Jain AK, Dubes RC (1988) Algorithms for clustering data. Prentice-Hall, Inc.

[148] Färber I, Günnemann S, Kriegel H, Kröger P, Müller E, et al. (2010) On using class-labels in evaluation of clusterings. In: MultiClust: 1st International Workshop on Discovering, Summarizing and Using Multiple Clusterings Held in Conjunction with KDD.