Cyber Analytics Visual Analytics Team 1 Project 2 Report

Zicheng Liu
Department of Computer Science
University of Delaware
zcliu@udel.edu

Ezeanaka Kingsley. U
Department of Computer Science
University of Delaware
ekingsly@udel.edu

Wanxin Li
Department of Computer Science
University of Delaware
wanxinli@udel.edu

ABSTRACT

This report recorded the second phase of a course project done by graduate students from Applications of Advanced Analytics to Cybersecurity Course lectured by John Cavazos from University of Delaware, which implemented the whole Class Signature process to analyze a testset of 1100 malware to distinguish discriminative features of good predicted and bad predicted malware clusters. The process in second phase mainly include adding filter module, generating ROC curve / confusion matrix, developing a web-page for display and optimization from the first phase.

1. INTRODUCTION

This report extended the part 1 of this project. In part 2, we used all the datasets (except for one for testing) given to us by Tristan to train a decision tree model used as a filter for the test dataset. We set a high bound and a low bound as thresholds for good and bad enough predictions of the test dataset. Thus, we filtered the results of moderate prediction of test dataset and kept good and bad enough results of prediction, that way we got a filtered test dataset. After that we used the filtered test dataset as the input of Clustering Module to generate 66 new sub-clusters. Then we applied the pipeline of project 1 on these new sub-clusters. Besides, for each of 11 malware families we also generated corresponding ROC curves and confusion matrices to evaluate the goodness and badness of predictions.

Fig.1 *- Finished Target Sample*

The functionalities we added in project 2 would help us enhance the discriminative nature of the top features of each sub-cluster of 11 malware families. We used Scikit Learn Python API to help create visualizations. Thus, we completed the extension to Class

Signature technique and finished our expected target sample shown as fig. 1 or check it online at this address https://class-signature-zichengl.c9users.io/index.html

1.1 Overview of Filtering Module

In the first phase of this project, we started Class Signature process directly from clustering module. As a result, the feature ranking and T-SNE modules displayed the visual analytic results of 66 sub-clusters simply generated from the 1100 original malware data via K-Means clustering algorithm.

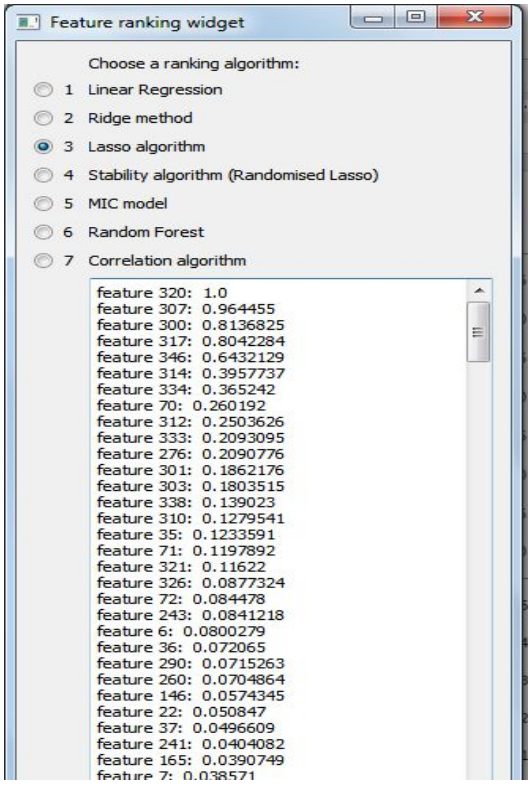
However, the whole Class Signature process should be implemented based on a prediction model at the beginning. Because the experts who created Class Signature process used this technique as a tool to analyze prediction model. In the second phase of this project, we developed filter module, which consists of a decision-tree model and thresholds. This filter module made Class Signature process become meaningful in visual analytics of malware.

1.2 Feature ranking (Best features)

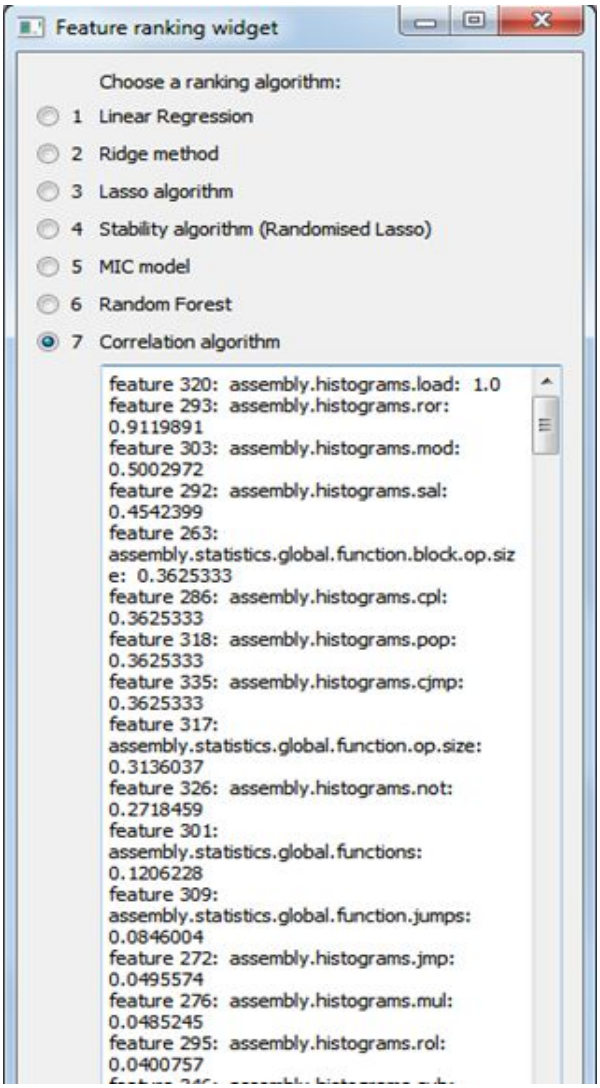
The big question in this area of machine learning is "Which features should you use to create a predictive model?"**.** This is arguably a difficult question that may require a deep knowledge of the problem domain.

It is actually possible to automatically select the very features in your data that prove to be the most useful or relevant for the given problem that you are working on. This is a process called feature selection.[1] One important attribute of feature selection is that it does not seek to modify the features in the original dataset but rather it selects the relevant features based on some scoring benchmark. The process of scoring the features and grouping them according to their scores is feature ranking[2].

In order to a make a simple, yet accurate predictive model, we need to select only the best features of that dataset. In part one of the project we already talked briefly about the different machine learning algorithms that are used to train a predictive model and generate set of scores for each of the features. In this part we will extract those best features, together with their their respective scores for our model construction.



(a)



(b)

Fig.2 *- Feature ranking scores from the highest ranked down to the lowest ranked:*

(a) In part one project phase without including the names of the features,

(b) In part two project phase with the corresponding feature names included

Figure 1 above displays the feature ranking interface at the two phases of our project. The first figure which comes from the first part of the project shows the list of the features numbered serially but sorted in descending order of their scores. In the second figure, we see an important modification in the display; In addition to the serially named features, we see the actual names of the features that own those scores. And of course, we also see that while the first output is produced by Lasso training algorithm, the second feature ranking scores were generated by the Correlation algorithm.

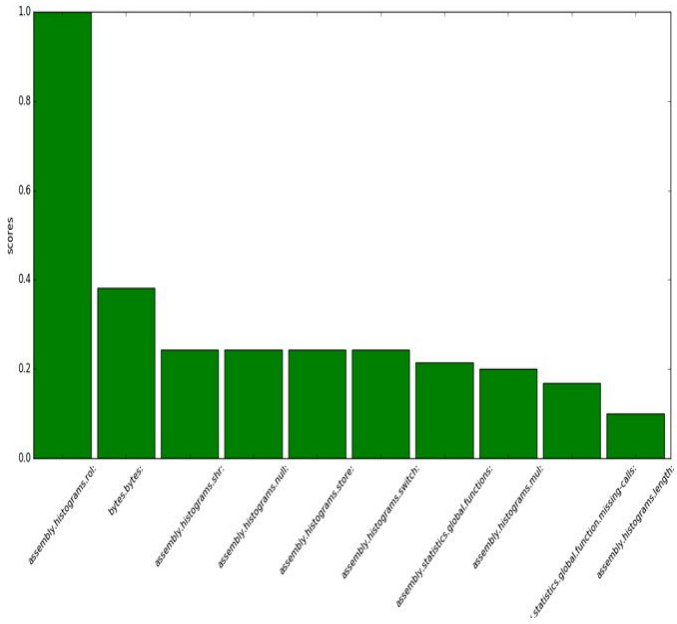


Fig.3 *- The top ten features for the 16-th cluster of our clustered malware dataset.*

Figure 3 above illustrates how the best features that got selected from one of the clusters compare against each other in terms of their relative importance scores. We see the highest ranked feature as “assembly-histograms-rol” and the least ranked ranked from our list of best features as “assembly-histograms-length”.

1.3 ROC and Confusion Matrix

In statistics, a receiver operation characteristic curve, i.e. ROC curve, is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied[3].

In this project, we have 11 target classes. Since ROC curve is applied in the binary classifier system, we converted 11 classes into 11 different corresponding combinations of binary classes with the specific class being positive. Thus, we generated 11 ROC curve along with corresponding positive class.

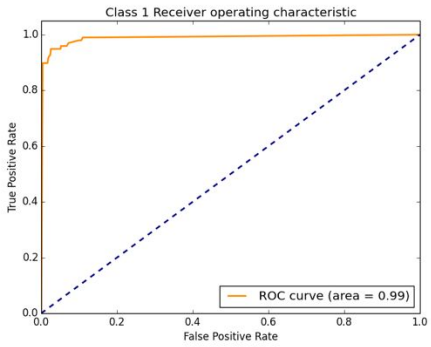


Fig.4 *- ROC Curve of Class 1*

In the field of machine learning and specifically the problem of statistical classification, a confusion matrix, also known as an error matrix, is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one (in unsupervised learning it is usually called a matching matrix). Each column of the matrix represents the instances in a predicted class while each row represents the instances in an actual class (or vice versa). The name stems from the fact that it makes it easy to see if the system is confusing two classes (i.e. commonly mislabelling one as another) [4].

In this project, we used scikit learn's confusion matrix API to generate 11 confusion matrices for each of combinations of binary classes. The degree of color's darkness represents the number of samples.

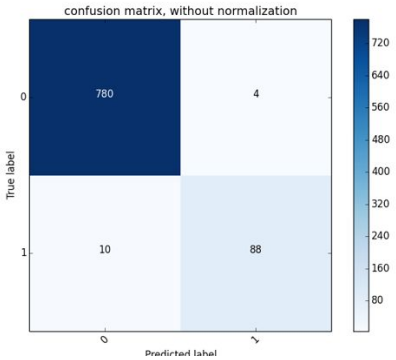


Fig.5 *- Confusion Matrix of Class 1*

1.4 Overview of the Web-Page

In the first phase of this project, we generated the feature ranking lists and T-SNE pictures for 66 malware sub-clusters. In the second phase of this project, we optimized feature ranking lists as histogram and optimize T-SNE as same shape with different colors for 66 malware sub-clusters. Besides feature ranking and T-SNE, we also generated ROC curve and confusion matrix for 11 malware families. However, all of these visual results were saved separately in folder.

Thus, we designed and developed a friendly web-page to display all of the visual results together in grid style via HTML framework.

2. PROGRAM PIPELINES

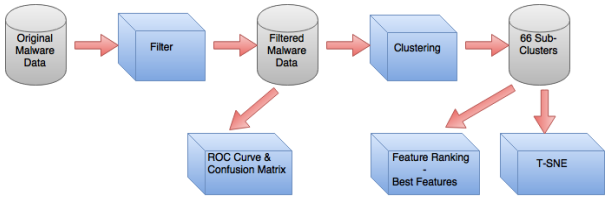


Fig.6 *- Project Workflow*

2.1 Filtering Module

In filter module, we initially imported 9,900 malware samples to train a decision-tree model using sklearn open source library. Then, we imported the 1,100 malware test set to this model, which is the same original dataset in project phase one. After that, we can get the family label prediction probabilities of these 1,100 malwares.

In the second step, we set the thresholds (<=0.2 or >=0.8) to filter 1,110 malware test set based on prediction probabilities. The "<=0.2" threshold means the prediction probability of target family label is less than or equal 20%, which can represent the very bad predictions. The ">=0.8" threshold means the prediction probability of target family label is bigger than or equal 80%, which can represent the very good predictions. Through this way, we can filter out median predictions and only keep the very good predictions and the very bad predictions. Since our prediction model is very good, the majority predictions are very good. As a result, we still got 882 filtered malware dataset.

Finally, we use three CVS files to save all the information of these 882 filtered malware dataset. The first CVS file called "filtered_matrix.csv" saves all the feature information of filtered malware dataset. The second CVS file called "filtered_target.csv" saves all the ground truth of family labels of filtered malware dataset. And the third CSV file called "filtered_prediction.csv" saves all the prediction probabilities of filtered malware dataset. There are 11 family labels in total, so there are 11 columns of prediction probabilities in this CSV file. The first two CSV files serve as the input for clustering module. The rest process are same with project phase one, which are generating 66-sub clusters with feature ranking and T-SNE.

The third CSV file serves as the input for ROC curve, which is a new feature in project phase two.

2.2 Getting the best features

The program builds on what we have for the first part of the project. We recall that the feature ranking widget reads a dataset of multiple features, numbers the features serially and trains the dataset using one of several machine learning algorithms to generate importance scores for all the features in the given dataset, and produces an output containing the scores of the features all listed in descending order of their importance together with their corresponding featureID's. In this final part of the project, the program extracts the names associated with the features in the process of training the dataset and prints the names alongside the featureID's of the features.

In the second phase of the part 2 of the project, the feature ranking scores generated are each copied to files for subsequent use in this phase. Hence the scores generated for the features associated with the label01

cluster will be copied and stored in the scores01.txt file. The score files are then used as inputs for extracting the best features which we decide, will be the top ten features in the feature rankings.

These top ten features together with their names and respective scores are then used plot a bar graph that visually shows how their relative importance of the features compare with one another.

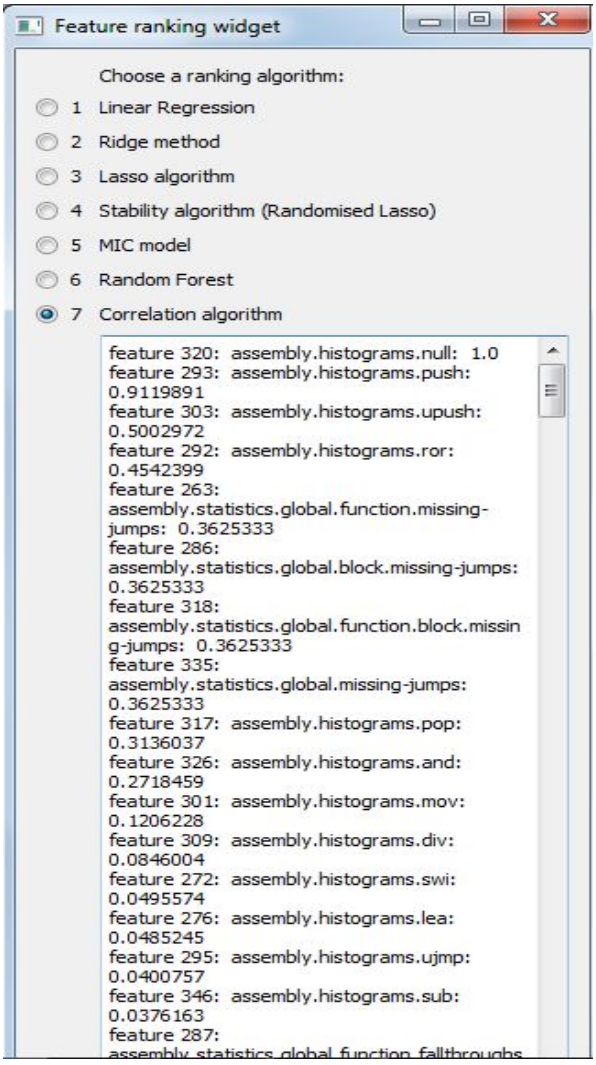


Fig.7 *- Screenshot of the widget after executing the correlation algorithm on the 66th cluster.*

Figure 7 above illustrates what we have been talking about.

As we can see, after completing the sorting of the importance scores, using the correlation algorithm, the highest ranked feature is "assembly.histogram.null". This feature, together with the next 9 top ranked features are used in making the bar chart seen in Figure 8 below.

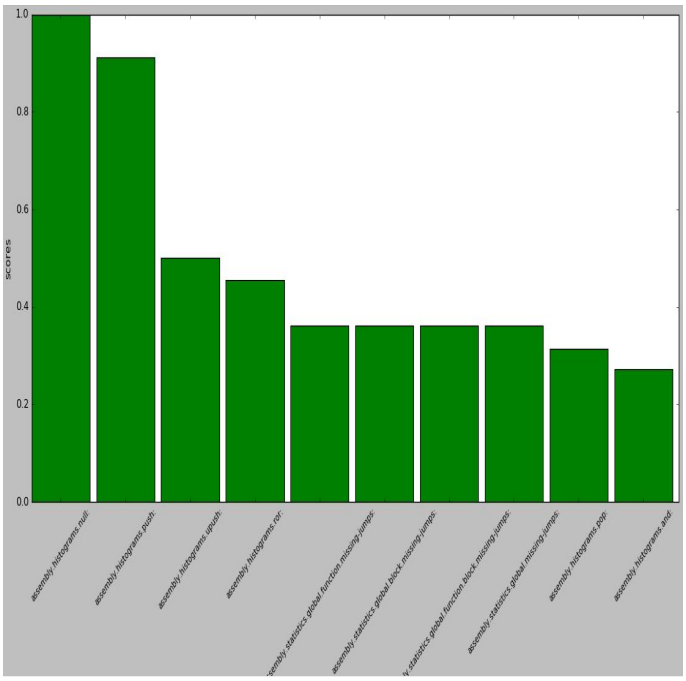


Fig. 8 *- Top ten ranked features of the 66th cluster*

2.3 ROC & Confusion Matrix

After we got the filtered data set, we converted 11 classes into 11 different corresponding combinations of binary classes with the specific class being positive. That is, for example, in the binary class of Andromeda, we set all samples labeled with 'Andromeda' positive and all samples with other labels negative. Then with the

combination of file named 'filtered_pridiction.csv' which is used to generate positive labels' scores, we created 11 inputs for ROC functions. That way we generated 11 ROC curves for each of Malware Class.

As for the binary class files, not only did we generate real labeled binary classes, but also the predicted binary classes according the decision tree model. That way we were able to create confusion matrices for each of Malware Class.

2.4 Web-Page

To design and develop a web-page, we use Bootstrap as framework. "Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web."[5] We displayed all of our visual results in grid on the web-page with the title of "Malware Class Signature". There are 11 rows on the web-page, which represent for 11 families based on labels. For each row, there are 7 columns. The first column contains the ROC curve and confusion matrix for its family. The rest 6 columns contains 6 sets of T-SNE graphs and feature ranking histograms for 6 sub-clusters generated by clustering module.

3. LIMITATIONS

3.1 Feature ranking limitations

Some of the limitations and challenges encountered in phase 1 remained in this phase, albeit, with lesser intensity. My knowledge of the python programming language has gradually progressed over the course of the project development but some difficulties associated with beginners of any programming languages persist.

Some of the algorithms for training the dataset are more effective than others in distinguishing features. The random forest algorithm, for example, was only able to distinctly give scores above 0 for only about a dozen features out of the vast number of features in the input dataset for cluster 66.

The graph also struggles to show the complete names of some of the features with unusually long names. Hence if you have a look at the feature ranking charts of some of the generated features, you would see that some of the feature names were not completely displayed.

3.2 Non-manually Set Thresholds of ROC

Since we used scikit learn as our external tool to help create ROC curves, the built-in functions do not provide the interfaces with setting the number of thresholds when drawing a ROC curve. It was automatically set according the result of predictions. Thus, we could not generate a ROC curve that was a great finish after our experiments with setting different number of thresholds.

4. CONCLUSIONS & FUTURE WORK

In project phase two, we finished the whole process of implementing Class Signature technique on malware dataset. Comparing with project phase one, we add new filter

module, ROC curve/confusion matrix, a web-page and optimization for feature ranking from phase one.

The filter module consists of a decision-tree prediction model and thresholds. After filtering, we got the dataset of very good and very bad predictions. The filtered dataset serve as the input for clustering module, which is same with the process in phase one. The prediction probabilities can serve as the input for ROC curve for each malware family based on label name, which is a new feature in phase two. Besides that, we can also get the confusion matrix for each malware family. In feature ranking, we generate the histograms of top ten features for each sub-cluster.

In order to display all the visual results on a friendly GUI, we design and implement a web-page via html framework. There are 11 rows which represent 11 malware families. For each row, there are 7 columns. The first column contains the ROC curve and confusion matrix. The remaining 6 columns consist of 6 sets of T-SNE graphs and feature ranking histograms.

The plan we will do in the future regarding this project is about its wrap-up process. That is to create a shell that can automate the pipeline of this project with user-friendly interface. We are also going to extend the functionality of this software to make it adaptive, saying that we can process any dataset that provides us with feature vectors, predictions and classes and generate the target Class Signature visualization to help apply visual analysis on the dataset.

In the feature ranking part, the plan for the future would be automating the process of reading the cluster files rather than having them done manually one file at a time.

5. REFERENCES

[1] Bermingham, Mairead L.; Pong-Wong, Ricardo; Spiliopoulou, Athina; Hayward, Caroline; Rudan, Igor; Campbell, Harry; Wright, Alan F.; Wilson, James F.; Agakov, Felix; Navarro, Pau; Haley, Chris S. (2015). *"Application of high-dimensional feature selection: evaluation for genomic prediction in man"*. *Sci. Rep.* .

[2] Gareth James; Daniela Witten; Trevor Hastie; Robert Tibshirani (2013). *An Introduction to Statistical Learning*. Springer. p. 204.

[3] Wikipedia. (2017, May 17). *Receiver operating* characteristic. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Receiver_operating_characteristi

[4] Wikipedia. (2017, May 14). *Confusion matrix*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Confusion_matrix

[5] Bootstrap. http://getbootstrap.com