

**COMP-8920 - Summer 2019**

**A Machine Learning Based Malware Detector**

**MINI PROJECT 02**

**DONE BY:**

**SAMEER AKHTAR SYED** **( Student ID: 105152677)**

**Feature Extraction:**

* We tried Principal Component Analysis (PCA) to extract 50 features from the input data. we also tried directly taking the given input as 77 features. The latter gave better or equal train and test accuracies, for all the models that we used. This trend did not change by changing the number of components for PCA. Hence we decided not to use PCA.
* The reason why PCA reduces accuracy may be related to the fact that it does not look at y-labels for preprocessing. Hence a feature which is varying a lot but not important towards deciding the y-label can make way into the 50 features selected by PCA. This may in turn lead to more important features being left out and hence a decrease in accuracy.
* We also tried other feature extraction techniques like normalization, but made the same observation with them. So we ended up taking the given inputs directly as features.
* As for the y-labels, we directly took the convention adopted in input, ie 1 for malware and 0 for safe.

**Model design, evaluation and debugging:**

* First, we used a decision tree model. However it had a train accuracy of 99.97%. We realized that such a high train accuracy is a result of overfitting and hence not a good metric for model performance.
* A better way to access the model would be to split it into train and validation and then measure the accuracy on validation for error calculation. However this may introduce selection bias. Hence the best way to go is k-cross validation, where we divide the train data into k parts, and alternatively use each part as validation and the rest for training. This overcomes the selection bias.
* The mean of accuracies obtained from each part serves as a proxy for test accuracy, which is not measurable due to absence of y-labels in test data.
* However this does not solve the overfitting problem of decision trees. To solve it we used the classic ensembling technique and used a Random Forest model. This reduces overfitting and hence we can expect an improvement in test accuracy. While test accuracy is not measurable, we did see an increase in the k-cross validation accuracy as described above.
* It should be noted here that while classification accuracy may be a desirable criteria for this assignment, recall score is more desirable in real life. It is much worse to miss an actual malware than to wrongly classify a harmless program as malware.

* The Decision Tree model surpasses the Random Forest model in this matter. This might be due to decision trees overfitting to *y=1* (Malware) points which are present in excess in the input data.

**Novelty evaluation:**

* The classes predicted by Decision Tree (dt\_clf) and Random Forest (rf\_clf) models are stored in “predictions.csv” along with the names of the program. The classes predicted by first performing PCA on the data are also available for the sake of inclusion with “\_pca” suffix.
* Note that we used the same principal axes in the test data as the one calculated from train data. This is so that any differences in the distribution of test and train data does not lead to vastly different principal axes.