CS F415: Data Mining Bonus Assignment
REPORT

Name: Arkil Patel
ID: 2016A7PS0665G

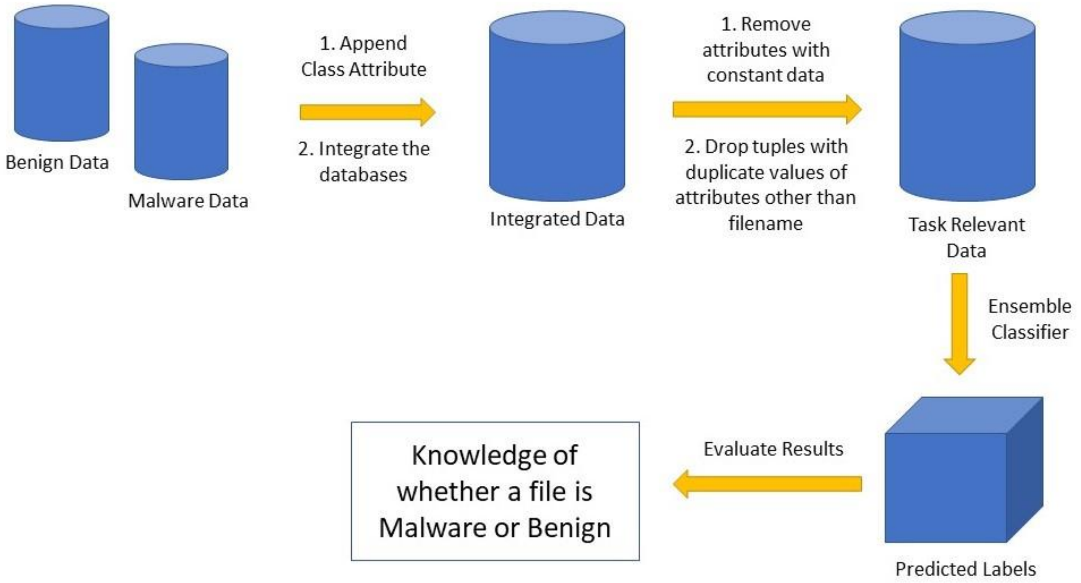
PROPOSED MODEL FOR CLASSIFICATION

I propose a unique model for classification which is a **combination of KNN, Random Forest and Neural Network**. Basically, it is just an ensemble model consisting of these three classifiers. Each classifier will vote either 0 or 1 and the majority is considered. The calculation method is shown at IN [112]. I was getting good results individually with these three classifiers so I combined them to yield an even better result. As expected, it yields an accuracy which is better than any of the models alone. This is because if for any given example, if one particular classifier makes a mistake due to its own fault or working mechanism which the other classifiers are not subjected to, the other classifiers can vote in majority to nullify the faults of the first classifier.

Result:
Accuracy: 0.99
Precision: 0.99
Recall: 0.99

Detailed discussion of performance and evaluation of individual models is given in the last section. Data preparation and pre-processing is also discussed there.

KDD PROCESS DIAGRAM


Benign Data
Malware Data
1. Append Class Attribute
2. Integrate the databases
Integrated Data
1. Remove attributes with constant data
2. Drop tuples with duplicate values of attributes other than filename
Task Relevant Data
Ensemble Classifier
Predicted Labels
Evaluate Results
Knowledge of whether a file is Malware or Benign

INSIGHTS ABOUT DATA AND MODELS

Initial Observations and Steps

- I am given a labelled dataset for classification. The classification is binary, i.e. two classes – benign and malware. I shall **denote benign as ‘0’** and **malware as ‘1’**.
- The Files are given separately and without classes. So, my first step is to **add the ‘Class’ column** with respect to above mentioned values and **concatenate the two datasets** into one for ease of use. Also, the

rows are shuffled so that any Classifier learns about both the Classes uniformly and not one after the other.

- While simply observing the data, I noticed that there were some columns which hold constant values throughout the dataset. These columns will not affect classification since they hold the same values irrespective of an example being malware or benign. Hence **all columns with constant values are dropped**.
- There are no NULL values.
- After removing the filename column, **duplicate values are found**. These are then removed.
- The **data is normalized** so that all column values lie in same range and no column gets undue importance. However, all classifiers except KNN work better without normalizing data as per observation.
- Past experience with classification including Assignment 1 tells me that dedicated Classification algorithms work better than Clustering algorithms for Classification. Hence, I shall focus only on Classification Algorithms.
- I shall evaluate and compare the following Classification algorithms – (1) KNN, (2) Logistic Regression, (3) Random Forest, (4) Naïve Bayes, (5) Neural Network. Note that I will not talk about **Decision Trees** separately in this report because they are **almost always outperformed by Random Forest** for Classification because a Random forest aggregates results of multiple Decision Trees and hence is not prone to overfit meanwhile also keeping the error low. This can also be seen from my code.

I. K Nearest Neighbours Model

- Since there is no intuition for deciding k, I plotted the error against k (see Fig2).
- From Fig 2**, K=3** was found to be the best choice. Any lesser value of K would lead to overfitting and higher value leads to poor results.
  Result:
  Accuracy: 0.98
  Precision: 0.98
  Recall: 0.98
- This model works well as can be seen from the Result.
- KNN is a lazy learner, so given the fact that there are enough training examples, it gives good accuracy.
- The classifier is prone to overfitting because of a low value of k.
- KNN works better on Normalized Data. This is because it uses a Euclidean distance as measure and hence we cannot give undue weight to any particular attribute just because its values are high.
- Note that there are more False Negatives as compared to False Positives. This is NOT desirable because this would mean that the classifier is considering actually malware file as benign a greater number of times.

II. Logistic Regression Model

- There are no major hyperparameters to be tuned for Logistic Regression.
- Logistic Regression is an Eager learner. It is observed to give poorer results than KNN.
  Result:
  Accuracy: 0.97
  Precision: 0.97
  Recall: 0.97
- Since Logistic Regression can be imagined as a Neural Network with a single neuron, it performs slightly worse than a Neural Network with multiple neurons. This is because multiple neurons can generate a better hypothesis for classification.
- Note that there are more False Negatives as compared to False Positives just like in KNN.

III. Random Forest Model

- Number of Decision Trees in the Random Forest was a hyperparameter which required to be tuned. In order to find this, I plotted the score against the number of trees (see Fig4).

- As can be seen from Fig4, the model starts to overfit after **3 trees**.
- This is the best Classifier of all the classifiers that I have used.

  Result:
  Accuracy: 0.98
  Precision: 0.98
  Recall: 0.98
- The reason that Random Forest performs so well is that it is actually an ensemble method which classifies on the basis of majority of votes of its constituent decision trees.
- Both Decision Trees and Random Forests were found to work better without normalizing the data.
- Note that there are more False Negatives as compared to False Positives just like in KNN and Logistic Regression. However, Random Forest significantly reduces the number of False Negatives when compared to KNN and Logistic Regression.

IV. Naïve Bayes Model

- There are no major hyperparameters to be tuned for Naïve Bayes Classifier.
- This is the worst Classifier of all the Classifiers that I have used.

  Result:
  Accuracy: 0.92
  Precision: 0.93
  Recall: 0.92
- The basic assumption for Naïve Bayes is that the effect of any attribute's value on the class is independent of the effect of other attributes' values on class. However, in our case, the attributes represent the number of opcodes which may sometimes have a combined effect on the class. I believe that this is the reason Naïve Bayes performs so poorly as compared to other classifiers.
- Note that there are more False Negatives as compared to False Positives just like in KNN, Logistic Regression and Random Forest.

V. Neural Network Model

- I wanted to experiment with a Neural Network to see if it performs better than a standalone Logistic Regression Unit.
- Since multiple layers would make the model unnecessarily complex and may lead to overfitting, I decided to use just one layer.
- The major hyperparameter that needed to be tuned was Number of neurons/units. For this, I plotted the score against the number of neurons (see Fig5).
- As can be seen from Fig5, the model becomes unstable and starts to overfit after more than **5 neurons**.
- The number of epochs and minibatch size also required some tuning. For epochs greater than 20, the model starts to overfit since it has learned too much.

  Result:
  Accuracy: 0.98
  Precision: 0.98
  Recall: 0.98
- Neural Network is observed to perform slightly better than Logistic Regression. It performs nearly as good as KNN but is poorer than a Random Forest.
- Note that there are more False Negatives as compared to False Positives. Hence this is a common pattern in all the classifiers.