Github: https://github.com/ShrJatin/Machine-Learning-Project-22

Machine Learning Project

Classification of Malicious executable files

:- Srijan Arora
:- Kumar Ankit
:- Mrinal Anand
:- Jatin Sharma

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY **DELHI**

Motivation :

Malware detection using signature/heuristic-based methods are becoming more and more difficult since all current malware applications tend to use polymorphic layers or use side mechanisms to update themselves to a newer version in a short period. Using the Microsoft malware dataset, we would like to explore the utility of different machine learning techniques for classification of different executables to identify which class a malware belongs to.

Literature Review :

- Paper 1: Malware Detection Using Machine Learning
  - The study extracts features from binary files and converts them into a binary vector with 308 features, the number of benign files being much larger than the number of malicious files.
  - It explains the use of various perceptron algorithms like One-Sided Perceptron, Kernelized One-Sided Perceptron and Cascade Classification to obtain accuracies upto 96%
  - It applies K-fold with the above algorithm for different number of folds and compares the results based on F1 and F2 scores, with an emphasis on minimizing false positives.
  - The paper also discusses a few optimizations for above algorithms in order to tackle large datasets in a comparatively shorter time.

Literature Review

- Paper 2: Malware detection using Linear SVMs

  The paper discusses the use of Linear SVMs to classify files into various types of malware/non malwares for use in anti-virus softwares.

  The technique used is to create bag-of-words and TFIDF vectors for each file to obtain vectors, which are then classified using Linear SVMS. The method gives 75-83% accuracy for classification of new files.

Dataset description :

About dataset:

- Dataset from Microsoft Malware Classification
- 10868 samples each of .byte file
    - Hexadecimal representation of the original executables.
- 10868 samples each of .asm file
    - Assembly dump of obtained from IDA tool.
- 9 classes, a malware could belong to.

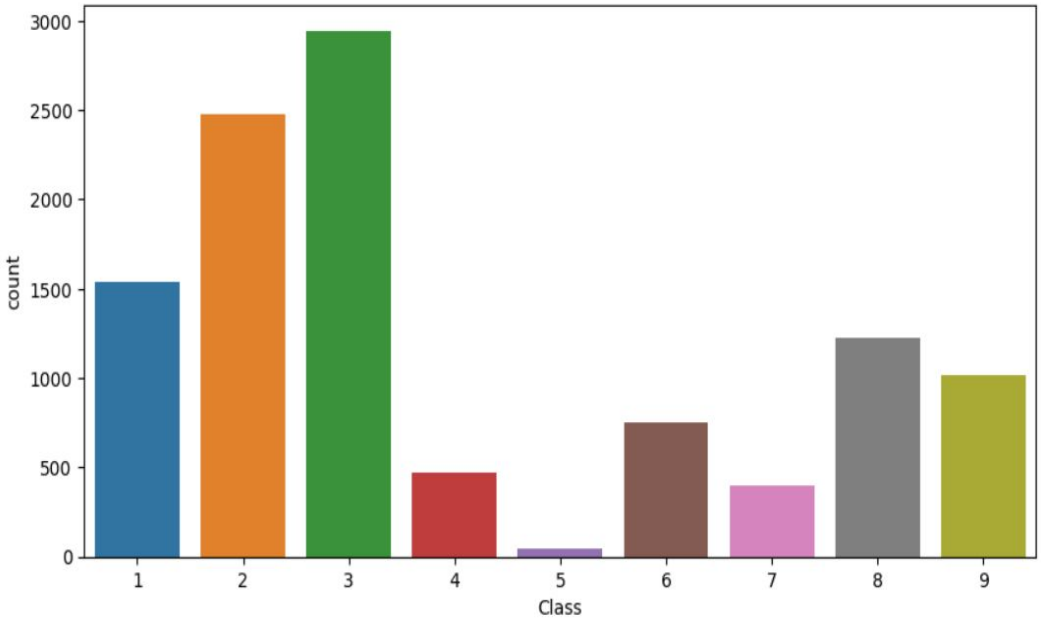
Preprocessing:

Obtaining Feature vectors:

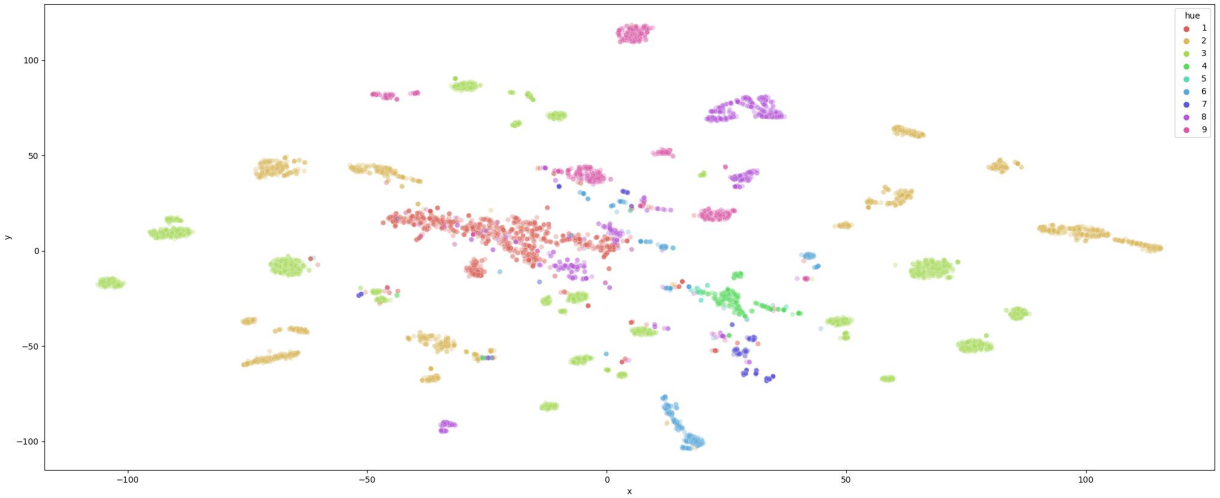
- 3 type of different feature vectors
  - 1 from byte files, 1 from asm file, 1 combined dataset of both asm and byte features.
- Used bag of words to get the feature vectors for both type of files.
- Used size, frequency of hexadecimals from 00 to ff as feature vectors for byte file.
- Used size, Header, Prefixes, Registers, Opcode as feature vectors for asm file.
- Combine both the feature vectors obtained above and used as feature vectors.
- Class balancing of data to remove the skewness of data

Visualization :

- Class distribution of the Data.
  - Skewed data(for class 4, 5, 7)
- Class balancing could be done.

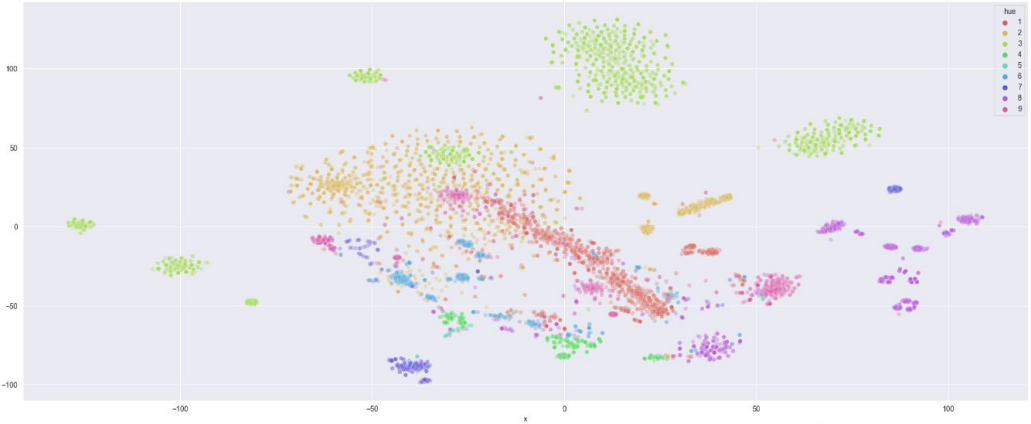


Visualization- TSNE



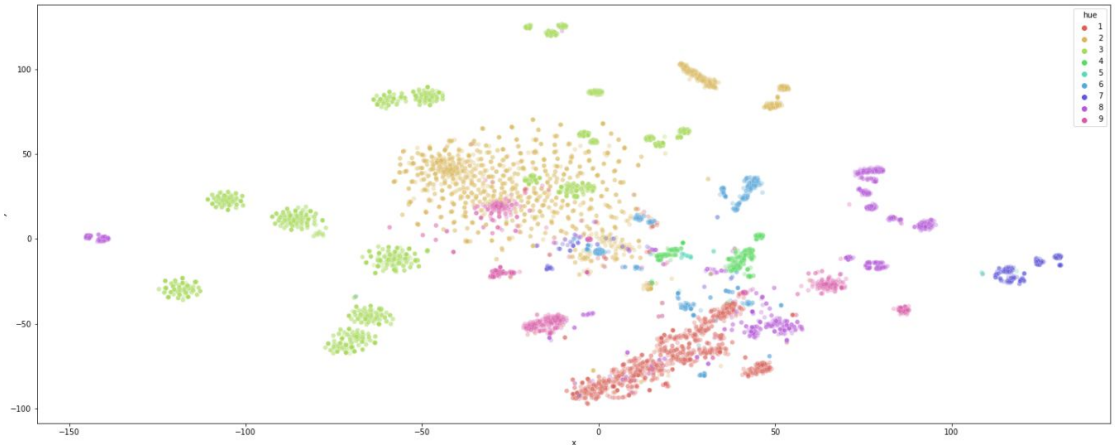
The TSNE plot for asm data. As can be seen, the data is not linearly separable and is all over the place.

Visualization- TSNE



The plot on the left is of the bytefile which has 258 features.

The plot on the right is of the combined dataset (Asm + ByteFile) which has 307 features.



Methodology :

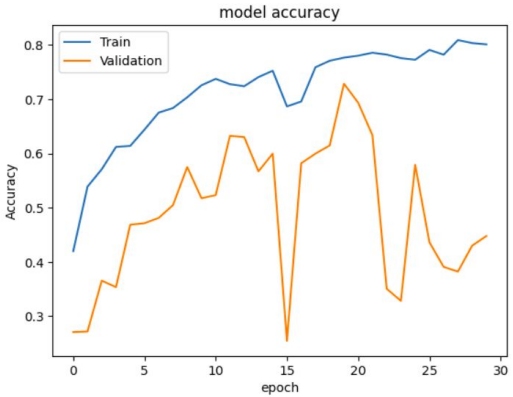
- For each of the 3 datasets, we also created oversampled versions of the data to reduce the class imbalance and compared results over the 6 resultant datasets.
- The same methodology is followed to train 5 different models on all the datasets:-
  - Logistic Regression (LR)
  - Naive Bayes (NB)
  - Random Forests (RF)
  - Support Vector Machines(SVM)
  - Artificial Neural Networks(ANN)

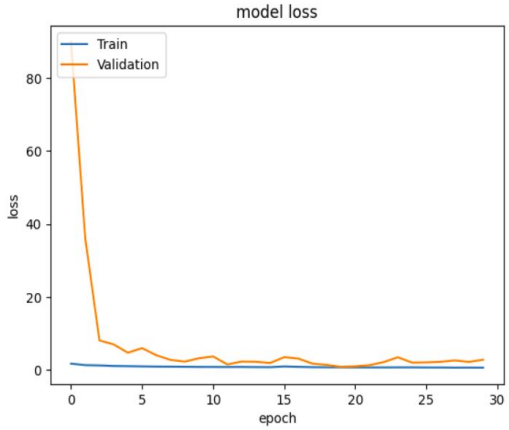
Methodology

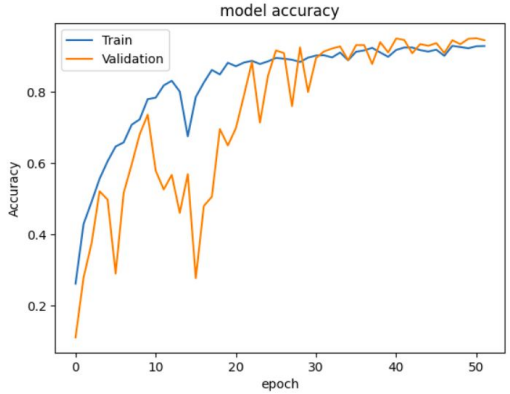
- For Logistic Regression and Random Forests, we applied Random Search and Grid Search algorithms to obtain the optimal hyperparameters.
- For Artificial Neural Networks, we used “leaky relu” as the activation function and used Grid Search to identify the best fit.
- For SVM, we tried using different kernels, with the RBF kernel giving the best results

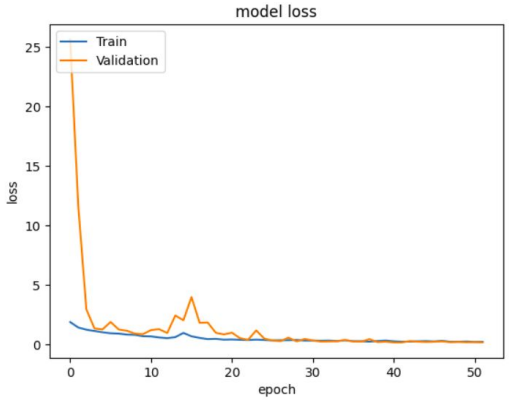
ANN Results:

Original Data





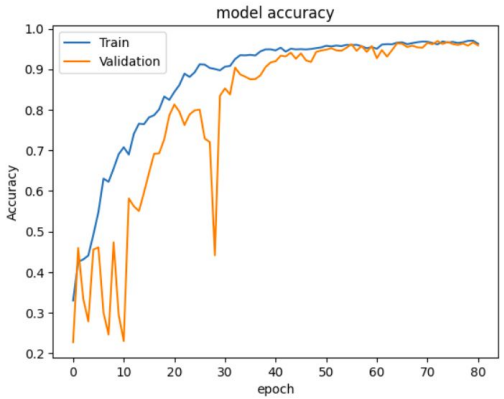


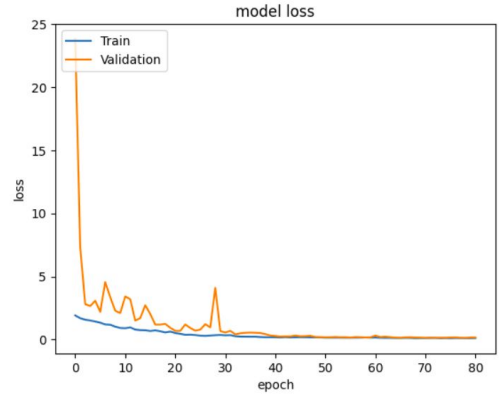


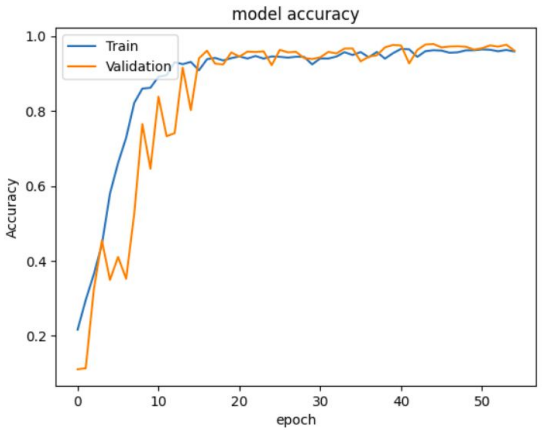
Oversampled Data

ANN Results

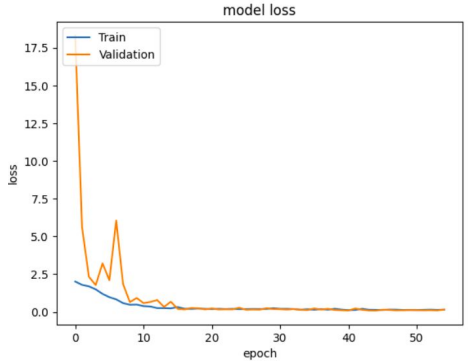
Original Data





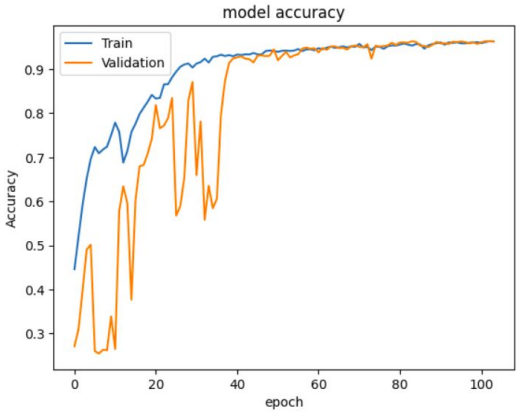


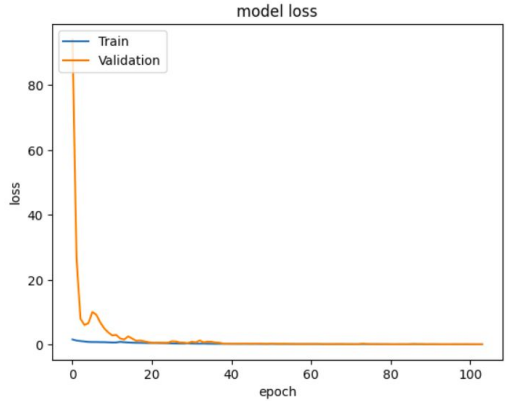
Oversampled Data

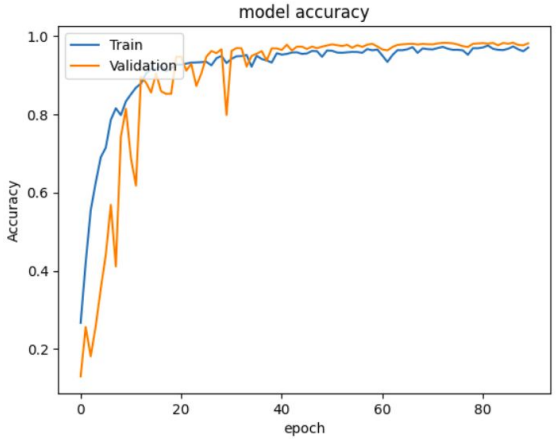


ANN Results

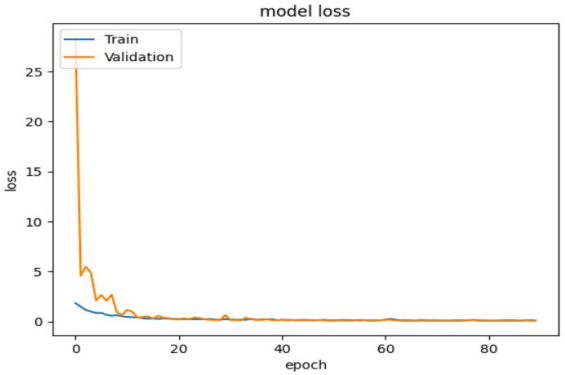
Original Data







Oversampled Data



Results and Analysis:

- Performance trend follows LR<NB<SVM<ANN<RF for accuracy across all 3 types of datasets (with one exception)
- RF always enjoys better performance when class balancing is carried out.
- Combined dataset always outperforms the individual 2 datasets.
- High variance detected in random forest when using combined dataset, which is mitigated by doing feature selection
- Performance of RandomForestCV is usually worse than GridSearchCV. However, performance drop in RF is negligible, making it usable for quick testing and deployment.
- ANN with Bytes dataset was the worst performing model, with other datasets giving good performance.

Model	Original Data				Oversampled Data			
	Accuracy	Train Loss	CV loss	Test Loss	Accuracy	Train Loss	CV loss	Test Loss
ByteFile Features	0.6476	1.0937	1.1366	1.1521	0.64558	1.3227	1.3367	1.263
Asm Features	0.6872	1.0115	1.0127	0.9982	0.6297	1.2601	1.2767	1.2368
ByteFile + Asm	0.6987	0.9272	0.9234	0.9598	0.6334	1.206	1.2093	1.207

Table 1. Results for Naive Bayes

Model	Original Data				Oversampled Data			
	Accuracy	Train Loss	CV loss	Test Loss	Accuracy	Train Loss	CV loss	Test Loss
ByteFile Features								
Logistic Regression	0.5896	1.1504	1.1482	1.1694	0.586	1.5779	1.5703	1.3901
Random Forest	0.9825	0.026	0.0641	0.078	0.9967	0.0191	0.0344	0.0267
Asm Features								
Logistic Regression	0.4144	1.683	1.6966	1.6846	0.2528	2.001	1.9925	1.9918
Random Forest	0.9898	0.026	0.0599	0.0551	0.9947	0.0094	0.021	0.0234
ByteFile + Asm								
Logistic Regression	0.6057	1.1765	1.1677	1.1704	0.6849	1.5337	1.5414	1.357
Random Forest	0.9912	0.0172	0.035	0.0413	0.9964	0.0082	0.0211	0.0199

Table 2. Results for Random Search

Model		Original Data				Oversampled Data		
	Accuracy	Train Loss	CV loss	Test Loss	Accuracy	Train Loss	CV loss	Test Loss
ByteFile Features								
Logistic Regression	0.6007	1.1484	1.1634	1.1612	0.5188	1.1572	1.5629	1.4004
Random Forest	0.9806	0.0288	0.0783	0.082	0.994	0.0189	0.0428	0.0285
Asm Features								
Logistic Regression	0.6076	1.1364	1.1663	1.11	0.5381	1.5842	1.5922	1.3745
Random Forest	0.9912	0.0155	0.0487	0.0418	0.9945	0.0094	0.021	0.0233
ByteFile + Asm								
Logistic Regression	0.6251	1.1689	1.1672	1.1588	0.6881	1.5232	1.5397	1.3551
Random Forest	0.9908	0.0167	0.048	0.0426	0.9949	0.0088	0.0241	0.0218

Table 3. Results for Grid Search

Model	Original Data				Oversampled Data			
	Accuracy	Train Loss	CV loss	Test Loss	Accuracy	Train Loss	CV loss	Test Loss
ByteFile Features	0.94576	0.227	0.2811	0.2899	-	-	-	-
Asm Features	0.9577	0.277	0.3508	0.3733	0.972	0.1754	0.1978	0.2434
ByteFile + Asm	0.973	0.1543	0.2554	0.2071	0.979	0.0975	0.1354	0.1366

Table 4. Results for SVM

Model	Original Data			Oversampled Data		
	Train Acc	Val Acc	Test Acc	Train Acc	Val Acc	Test Acc
ByteFile Features	0.5169	0.5095	0.5092	0.9501	0.9452	0.9489
Asm Features	0.9733	0.9586	0.95998	0.9615	0.9608	0.9603
ByteFile + Asm	0.9760	0.9638	0.97148	0.9843	0.9809	0.9769

Table 5. Results for ANN

Timeline :

Due to the splitting of initial work between 2 groups, the timeline given initially was not followed. The models trained using the combined dataset was done in the last week leading up to the deadline.

For the 2nd part of the project, the following timeline was followed:

1/11/22-14/11/22- generating features (both pairs work on respective datasets)

14/11/22-21/11/22-running models on new datasets

21/11/22-27/11/22-models on combined datasets

27-11/22-4/12/22-comparing models, report writing

Contribution :

The work was split in 2 in order to parallely work with bytes and asm files. The portion of the bytes file was handled by Srijan and Mrinal, while the portion of the asm files was handled by Jatin and Ankit.

For bytes file:-
Data extraction, data visualisation and EDA - Srijan
Model training, Cross validation - Mrinal

For asm file:-
Data extraction, data visualisation and EDA - Ankit
Model training, Cross validation - Jatin

For combined file:-
Data combination, data visualisation and EDA - Srijan,Ankit
Model training, Cross validation - Mrinal, Jatin

Thank You!