Malware Detection

Abhinav Singh
MT20202001

Labdhi Kapasi
MT20202066

Shivam Jain
MT2020002

***Abstract*—This is a detailed report on our work on Malware Prediction Problem, with the aim to predict if a machine will soon be hit with malware or not.**

***Index Terms*—EDA, Feature Engineering, Preprocessing, Random Search CV, Thresholding, Decision Tree Classifier, Random Forest classifier, LightGBM Classifier, Stacking Classifier.**

PROBLEM STATEMENT

The malware industry continues to be a well-organized, well-funded market dedicated to evading traditional security measures. Once a computer is infected by malware, criminals can hurt consumers and enterprises in many ways.

The goal of this competition is to predict a Windows machine’s probability of getting infected by various families of malware, based on different properties of that machine. The telemetry data containing these properties and the machine infections was generated by combining heartbeat and threat reports collected by Microsoft's endpoint protection solution, Windows Defender.

Can you help protect more than one billion machines from damage before it happens?

DATASET

Each row in this dataset corresponds to a machine, uniquely identified by a `MachineIdentifier`. `HasDetections` is the ground truth and indicates whether Malware was detected on the machine.

The dataset contains two files train.csv and test.csv ,

`train.csv` contains 567730 rows and 83 columns

`test.csv` contains 243313 rows and 82 columns.

Dataset contains 53 numerical features and 30 categorical features.

In Dataset some of the features contains NULL values.

Few important columns are -

- **`HasDetections` -** Ground truth
- **`MachineIdentifier`** - Unique Machine ID
- **`SmartScreen`** - This is the SmartScreen enabled string value from registry files.
- **`AvSigVersion`** – Windows Defender state information
- **`Wdtf_IsGamer`** - Indicates whether the device is a gamer device
- **`Firewall`** -This attribute is true (1) for Windows 8.1 and above if windows firewall is enabled, as reported by the service.
- **`SMode` -** Whether the device is known to be in S Mode where only Microsoft Store apps can be installed
- **`IsProtected` -** whether there is at least one active and up-to-date antivirus product running on this machine.

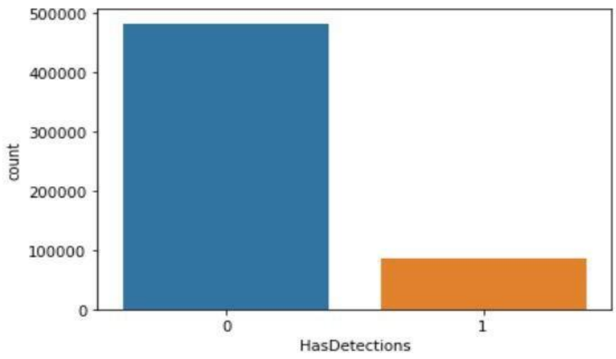
‘

I. INTRODUCTION

Malicious software is abundant in a world of innumerable computer users, who are constantly faced with these threats from various sources like the internet, local networks and portable drives. Malware is potentially low to high risk and can cause systems to function incorrectly, steal data and even crash. Malware may be executable or system library files in the form of viruses, worms, Trojans, all aimed at breaching the security of the system and compromising user privacy. Typically, anti-virus software is based on a signature definition system which keeps updating from the internet and thus keeping track of known viruses. While this may be sufficient for home-users, a security risk from a new virus could threaten an entire enterprise network.

The primary goal of this competition is to predict a Windows machine’s probability of getting infected by various families of malware, based on different properties of that machine. Hence, the data required for malware prediction can be any information about the state of a computer which is hit by a malware attack. As there are various types of malware attacks, machines may behave differently when attacked. Therefore, it is useful to collect a large amount of data about computers that are attacked. Most of the data comes from the system behavior of the machine and the type of the machine. For our case Microsoft has done all the process of capturing the information from a windows defender through a long course of time.

II. DATA PREPROCESSING

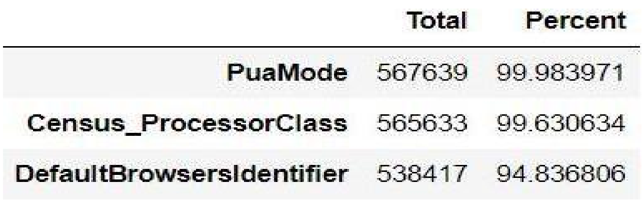


```
From total data  15.0 %  Malware Detected
From total data  85.0 % Malware Not Detected
```

The raw data is very high dimensional and has many missing values. The size of data is also comparatively large. Each row in this dataset corresponds to a machine, uniquely identified by a MachineIdentifier. HasDetections is the ground truth and indicates that Malware was detected on the machine. As we can see from above image that HasDetections is highly imbalanced, so while training the model we need to select the model's which could handle imbalanced data and evaluate the predictions using the right performance metrices otherwise results may be misleading.

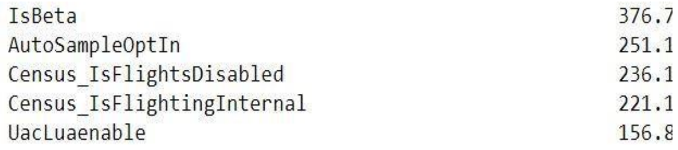
As we all know the biggest challenge before training the model is selecting the relevant features. Second, preparing the data so that it can properly fit the algorithm. Hence in our project here as well, we tried to preprocess the data. Hence reducing the size of the data and removing non relevant or unnecessary columns was very crucial. Here we have provided the data cleaning and preprocessing steps with description -

- **Mostly-missing Feature**- In the given data set there are columns that have more than 99% of missing values removing them will not affect accuracy of model. Below are the features having NULL value percentage > 90%



	Total	Percent
PuaMode	567639	99.983971
Census_ProcessorClass	565633	99.630634
DefaultBrowsersIdentifier	538417	94.836806

- **Too-skewed features** - There are the columns in dataset which are highly skewed means majority of occurrences are covered by one type of categories so these columns will not contain any useful information hence we can remove them. Below are the mentioned highly skewed features with their skewness value.



```
IsBeta                                                    376.7
AutoSampleOptIn                                           251.1
Census_IsFlightsDisabled                                  236.1
Census_IsFlightingInternal                                221.1
UacLuaenable                                              156.8
```

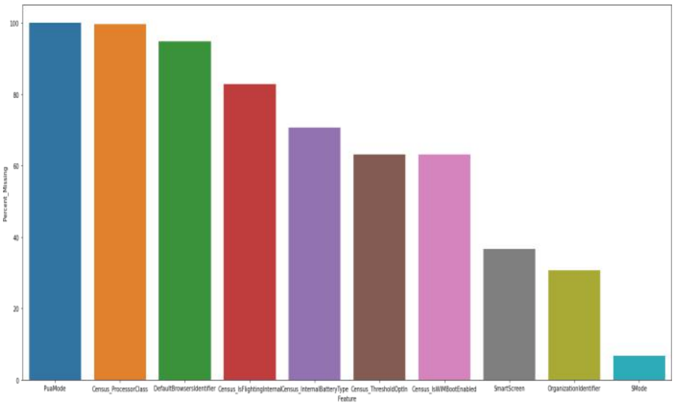
- **Highly-correlated features**- Correlations between columns show how related the two columns are, we have used heatmap and VIF score to check highly correlated features and removed highly correlated features.
  1. Census_OSBuildNumber,
  2. Census_OSInstallLanguageIdentifier,
  3. Census_InternalPrimaryDisplayResolutionVertical

These are the features which can be dropped since they can be derived from another features. So, in total we have removed 11 columns without affecting the performance of model.

In Dataset there are some categorical features so we need to encode them before giving it to model we have used one hot encoding for features having less than 5 categories followed by label encoding to encode remaining features.

III. Data Visualization

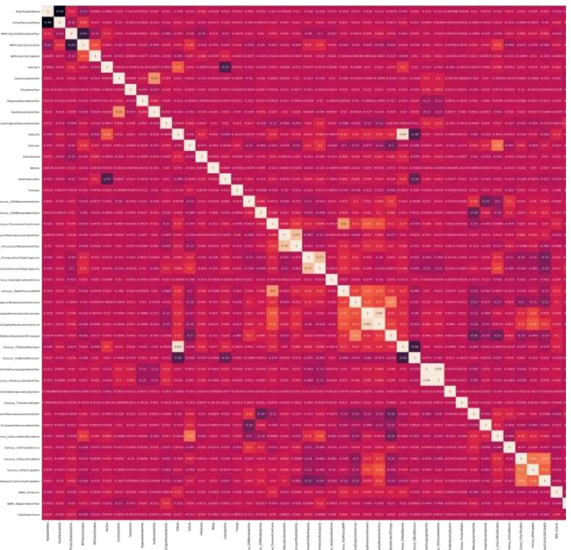
We have seen output to be predicted is binary and highly imbalanced we are provided with 82 features based on which we need to predict whether Malware will be detected or not and some of these features are categorical.
While analyzing these features we have seen some of them are highly skewed and some of them contain high null value percentage.



This bar plot displays the features which are having high NULL value percentage

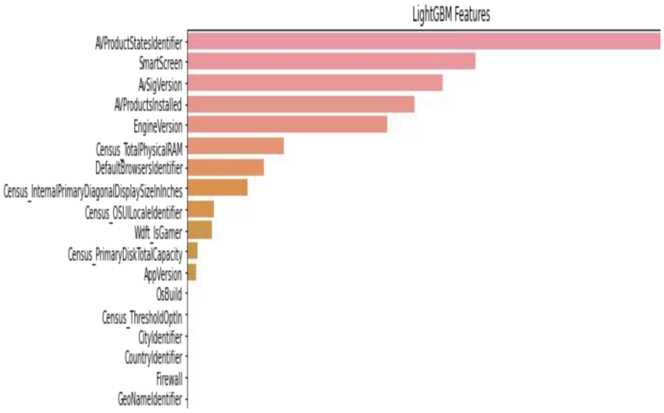
We also tried to plot heatmap to analyze correlation between features.

From the heatmap we observed that there are some features which have high correlation.

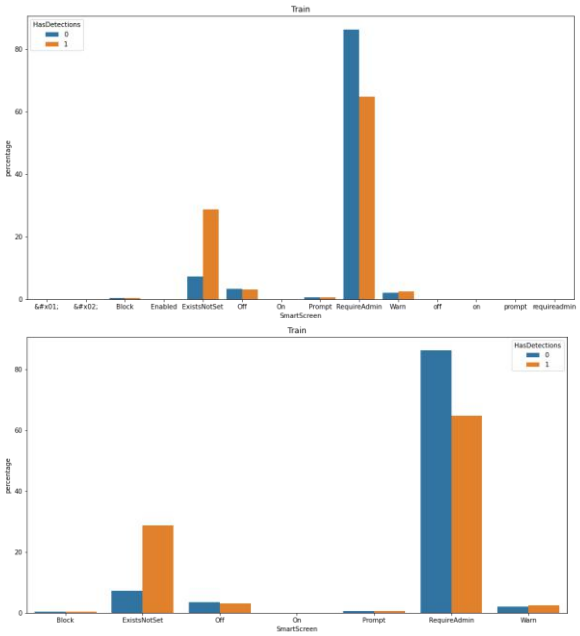


- Census_OSInstallLanguageIdentifier and Census_OSUIlLocaleIdentifier are highly correlated
- Census_OSBuildNumber and OSBuld are highly correlated.

IV. Feature Engineering



Here to determine the important features we used LGBM classifier on preprocessed data. The results we found after plotting the feature importance graph are as shown above. We then carefully analyzed each of these features again. In feature SmartScreen we found many categories representing the same value but with different categories so we combined them. For feature AVProductsInstalled ,which represents the number of antivirus products installed in the system, we filled the NA values with –1 to represent the null value category. For the rest of the important numerical features, we filled the null values with the mode to maintain their distributions.



Percentage plot of Smart Screen before and after combining categories

V. Training and Results

After performing data preprocessing where we removed less relevant features and performed encoding of categorical features. We have now total 82 features in our dataset on which we need to train the model, while training the model we calculated the best hyperparameters for each model using Randomized search CV. After this we calculated the threshold value using k cross validation at which model will give best Auc Score. We then feed the best parameters returned by Randomized search CV to model and fit our model on the training data accordingly. Finally, we made our predictions for the test data using the best hyperparameters.

We trained our data on the following algorithms -

- Decision Tree Classifier
- Random Forest Classifier
- LGBM Classifier
- Stacking Classifier

The AUC SCORE on these models were the following

AUC SCORE ON VARIOUS ALGORITHMS

Classifier	AUC SCORE
Decision Tree Classifier	0.6696
Random Forest classifier	0.7114
LGBM Classifier	0.7224
Stacking classifier	0.7176

VI. Conclusion

There is always a tradeoff when we choose different machine learning algorithms to carry out our training and testing. It is obvious that there are challenges associated with machine learning algorithms. The main challenges or the so-called concern are the time it takes to run and the accuracy that the gives. While working on different model's we also witnessed that some model takes large run time and large amount of memory like Random forest Classifier that is why we tried LGBM which is a gradient boosting framework that uses tree-based learning algorithms and takes very less time and less memory to execute and is highly efficient. After training and analyzing various model's we found that Lgbm classifier yield's the best AUC score and fastest runtime, along with lowest training time.

VII. Acknowlegement

We would like to thank our teaching assistant Saurabh Jain for helping us out on numerous occasions, Special thanks to Raghavan sir. His highly detailed lectures on several topics helped us understand concepts very well.

VIII. Refrences

- https://www.kaggle.com/youhanlee/my-eda-i-want-to-see-all
- https://www.kaggle.com/airbourne/data-dictionary
- https://www.kaggle.com/harmeggels/random-forest-feature-importances
- https://neptune.ai/blog/lightgbm-parameters-guide
- https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5
- https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html