Malware Detection
Team : Globetrotters

Nitesh Jain
MT2020118
IIIT Bangalore
nitesh.jain@iiitb.org

Mohd Asad Ansari
MT2020147
IIIT Bangalore
mohammadasad.ansari@iiitb.org

Abhishek Garg
MT2020021
IIIT Bangalore
abhishek.garg@iiitb.org

***Abstract*—The problem given in the competition is a Machine Learning problem to predict the probability whether a Windows machine is infected by the Malware or not. This is checked on the basis of different properties/characteristics of the machine.**

***Index Terms*—Exploratory data analysis,Data preprocessing,Feature engineering,Model Ensembling,Stacking.**

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.

We are provided with certain properties of the Window's machine using which we need to detect if the machine is infected with the Malware.

Dataset

The data set given in the competition contain multiple feature properties and each row in this dataset corresponds to a machine, uniquely identified by a MachineIdentifier.There are 83 columns in the dataset, amongst which there are some row values marked with "NA"

While the dataset provided here has been roughly split by time, the complications and sampling requirements mentioned above may mean you may see imperfect agreement between your cross validation, public, and private scores! Additionally, this dataset is not representative of Microsoft customers' machines in the wild; it has been sampled to include a much larger proportion of malware machines.

Uniquely identified MachineIdentifier in each row corresponds to HasDetections indicating that Malware was detected on the machine.Using the information and labels in train.csv, you must predict the value for HasDetections for each machine in test.csv.

I. Introduction

Malware detection is crucial with malware's prevalence on the Internet because it functions as an early warning system for the computer secure regarding malware and cyber attacks. It keeps hackers out of the computer and prevents the information from getting compromised.As someone who works in computers, you try your best to ensure that malware doesn't affect your system.

The malware can be detected in the system using some features. These features can help us to detect if the system is affected by the malware. We are given with such features using which our model can predict the same.

II. Data Visualization

The dataset consists of features as columns. These are different kinds of data types that is int 64 or float64 while other are categorical data. In the dataset, there are certain features that have very high number of missing values(approx 99 percent).There is a high amount of correlation (higher than 0.6) amongst the numerical features. In the object data types, there is large number of categories. In the dataset, we can also observe that the dataset is highly unbalanced.

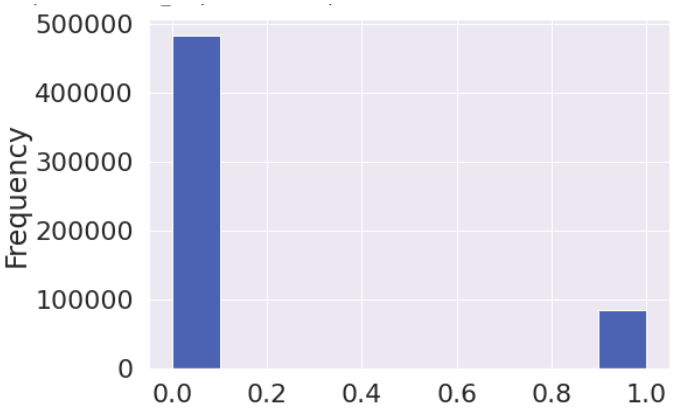

Fig. 1. Target Analysis

III. Data Pre-processing

There are total of 83 features in the train dataset while in test dataset there are 82 features. It is observed that hasDetection feature is missing. It is also observed from the dataset that the data is highly skewed and contains a lots of missing values. In the dataset, there are both numerical as well as categorical features.

A. Handling Missing Data

For the missing values, we have replaced the NaN with 0 to put the whole dataset to same ground. Later, we selected the values with more than 1000 observations and then dropped

those rows. At the end, 0 will be used in place of dropped values. Also at the end, we drop unbalanced values.

	Total	Percent
PuaMode	567626	99.981681
Census_ProcessorClass	565537	99.613725
DefaultBrowsersIdentifier	538189	94.796646
Census_IsFlightingInternal	469456	82.690011
Census_InternalBatteryType	400058	70.466243
Census_ThresholdOptIn	358394	63.127543
Census_IsWIMBootEnabled	357842	63.030314
SmartScreen	207184	36.493404
OrganizationIdentifier	176175	31.031476
SMode	38414	6.766245
CityIdentifier	20711	3.648037
Wdft_IsGamer	18985	3.344019
Wdft_RegionIdentifier	18985	3.344019
Census_InternalBatteryNumberOfCharges	16042	2.825639
Census_FirmwareManufacturerIdentifier	12987	2.287531
Census_FirmwareVersionIdentifier	11310	1.992144
Census_IsFlightsDisabled	9983	1.758406
Census_OEMModelIdentifier	6851	1.206736
Census_OEMNameIdentifier	6387	1.125007
Firewall	5781	1.018266
Census_TotalPhysicalRAM	5364	0.944815
Census_IsAlwaysOnAlwaysConnectedCapable	4413	0.777306

Fig. 2. Missing Values %

B. Feature Selection

In the numerical data, we observed a very high correlation. To remove that we need to remove highly correlated coloumns i.e., coloumns with correlation values greater than 0.6 in order to avoid multidimentionality in the dataset.

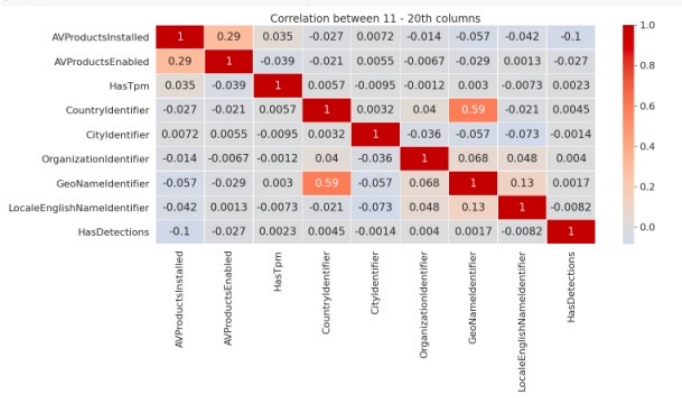

Fig. 3. HeatMap-Correlation Matrix

C. Encoding

Most of the categorical data is of string data type. So we used Label Encoder to to convert the same into numerical ones.That's primarily the reason we need to convert categorical columns to numerical columns so that a machine learning algorithm understands it.

D. Imputation

To handle missing data amongst the numeric features we employ mean strategy of imputation. For categorical features Label Encoder automatically assigns a numeric category to missing (NaN) values.

IV. Model Ensembling

Ensemble modeling is a process where multiple diverse models are created to predict an outcome, either by using many different modeling algorithms or using different training data sets. The ensemble model then aggregates the prediction of each base model and results in once final prediction for the unseen data. The motivation for using ensemble models is to reduce the generalization error of the prediction.[1]

A. Logistic Regression

The logistic classification model (or logit model) is a binary classification model in which the conditional probability of one of the two possible realizations of the output variable is assumed to be equal to a linear combination of the input variables, transformed by the logistic function.

B. Light GBM

Light GBM is a fast, distributed, high-performance gradient boosting framework based on decision tree algorithm, used for ranking, classification and many other machine learning tasks.Since it is based on decision tree algorithms, it splits the tree leaf wise with the best fit whereas other boosting algorithms split the tree depth wise or level wise rather than leaf-wise.

C. XGBoost

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting that solve problems in a fast and accurate way[2].

D. Stratified K-Fold

Approach used for model evaluation is the train/test split and the k-fold cross-validation procedure. Both approaches can be very effective in general, although they can result in misleading results and potentially fail when used on classification problems with a severe class imbalance. Instead, the techniques must be modified to stratify the sampling by the class label, called stratified train-test split or stratified k-fold cross-validation.Each fold tries to depicts or duplicates the original data[3].

V. Training the model

After feature engineering, our number of useful features reduced.Reducing the features also decreased the complexity of the model as well.After modifying the model parameters we could improve the model accuracy.Also we tried different supervised learning models to improve our accuracy.We trained our data on the following algorithms:

- Logistic Regressor(LR)
- LightGBM(LGBM)
- XGBoost(XGB)
- Stratified KFold(SKFold)

S.No.	Algorithm	roc_auc score
1.	Logistic+RandomForest	0.66235
2.	LGBM+XGB+CatBoost	0.71481
3.	Stratified KFold LGBM	0.72111
4.	Stratified KFold XGBoost	0.72197

```
Fold 1
[0]     validation_0-auc:0.672933
Will train until validation_0-auc hasn't improved in 500 rounds.
[500]   validation_0-auc:0.718705
[1000]  validation_0-auc:0.720848
[1500]  validation_0-auc:0.721101
[2000]  validation_0-auc:0.720611
Stopping. Best iteration:
[1506]  validation_0-auc:0.72113


Fold 2
[0]     validation_0-auc:0.668894
Will train until validation_0-auc hasn't improved in 500 rounds.
[500]   validation_0-auc:0.713902
[1000]  validation_0-auc:0.716383
[1500]  validation_0-auc:0.716566
Stopping. Best iteration:
[1365]  validation_0-auc:0.716649


Fold 3
[0]     validation_0-auc:0.669975
Will train until validation_0-auc hasn't improved in 500 rounds.
[500]   validation_0-auc:0.7176
[1000]  validation_0-auc:0.720056
[1500]  validation_0-auc:0.720607
[2000]  validation_0-auc:0.720359
Stopping. Best iteration:
[1697]  validation_0-auc:0.720781
```

Fig. 4. Execution of various KFold for XGB model

The k value must be chosen carefully for your data sample.A poorly chosen value for k may result in a mis-representative idea of the skill of the model, such as a score with a high variance,or a high bias, (such as an overestimate of the skill of the model).

VI. Conclusion

Using the stratified Light GBM and Stratified XGB,our model was able to predict whether the malware was detected by the system or not with an accuracy of 72.197% .Also we could predict which are the more dominant features that helps in better prediction thereby increasing the model accuracy.

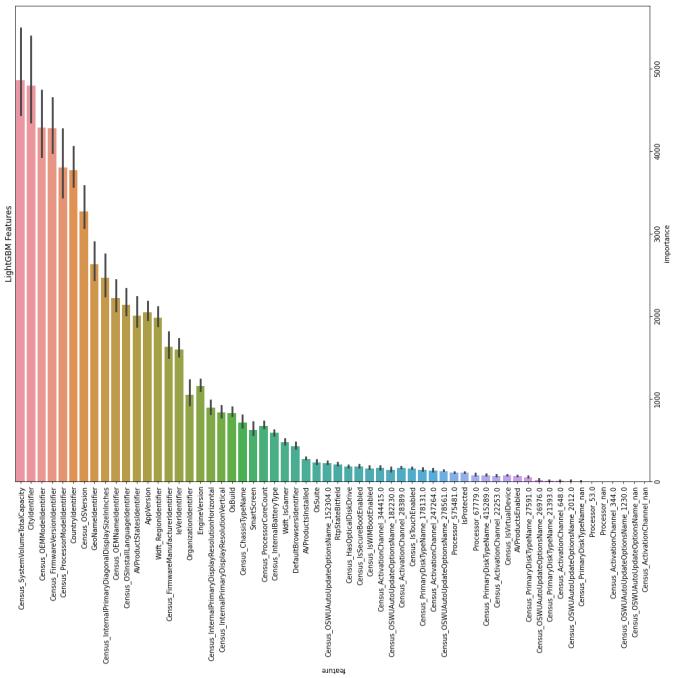

Fig. 5. Feature Relevance & Selection

VII. Acknowledgement

We would like to thank Professor G. Srinivas Raghavan and our Machine Learning teaching assistant Shreyas Gupta,for giving us the opportunity to work on the project and helping us out in the initial stages in numerous occasions. His highly detailed lectures on various topics helped us understand what we were actually doing.

Leader-board was great motivation to work on the project and the competition forced us to read up various articles and papers which gave us ideas and enthusiasm for the project.

VIII. PROJECT FILE LINK

Our project folder including the python code,script file and submission csv file,is available at the following GDrive link: http://bit.do/Malware-Detection-GATeam

IX. REFERENCES

[1] Jovan Sardinha. An introduction to model ensembling. [Online]. Available: https://medium.com/weightsandbiases/an-introduction-to-model-ensembling-63effc2ca4b3

[2] Philip Hyunsu Cho, Nan Zhu et.al., XGBoost. [Version: 1.2.0]. [Online]. Available: https://xgboost.readthedocs.io/

[3] Jérémie du Boisberranger, Joris Van den Bossche et.al., sckit-learn: Open source scientific tools for Machine Learning. [Latest] [Online]. Available: https://scikit-learn.org/

[4] Microsoft Corporation Revision 9597326e. LightGBM. [Latest]. [Online]. Available: https://lightgbm.readthedocs.io/

[5] Andrei (Andrey) Khropov, annaveronika et.al., catboost.ai. [version: 0.24.3]. [Online] https://github.com/catboost/catboost