Virus Share

Fossemò Daniele, Lozzi Daniele, Rea Gianluca

Università degli Studi Di L'Aquila

AA 2021/2022

Description of the dataset - general info and labels

The dataset contains the 482 features of 107888 executables, collected by VirusShare and evaluated by Cuckoo.
The samples are labeled (column 0 of the dataset) with a continuous value [0, 1] representing the probability that the sample is a malware. We decided to make classification experiments on the dataset, so we have rounded the labels:

- 0 $\rightarrow$ Not a malware (29843 samples).
- 1 $\rightarrow$ Malware (78013 samples)

paper font (cited by UCI repository): `https://link.springer.com/chapter/10.1007/978-3-319-67786-6_2`

Description of the dataset - Features

- **API call Category** (1:353) $\rightarrow$ the invoking frequencies by the sample of the API provided by the operating system.
- **Registry category** (354:357) $\rightarrow$ the number of registries of the OS being written, opened, read and deleted by the sample.
- **File system category** (358:369) $\rightarrow$ the number of files (and directories) being opened, written, moved, read, deleted, failed and copied
- **Miscellaneous category** (370:482): $\rightarrow$ the binary features of whether the community signatures of Cuckoo are triggered or not.

So the variables in the firsts three category are discrete numeric ratio variables, while the ones in the last are categorical dichotomous variables.

Data Cleaning

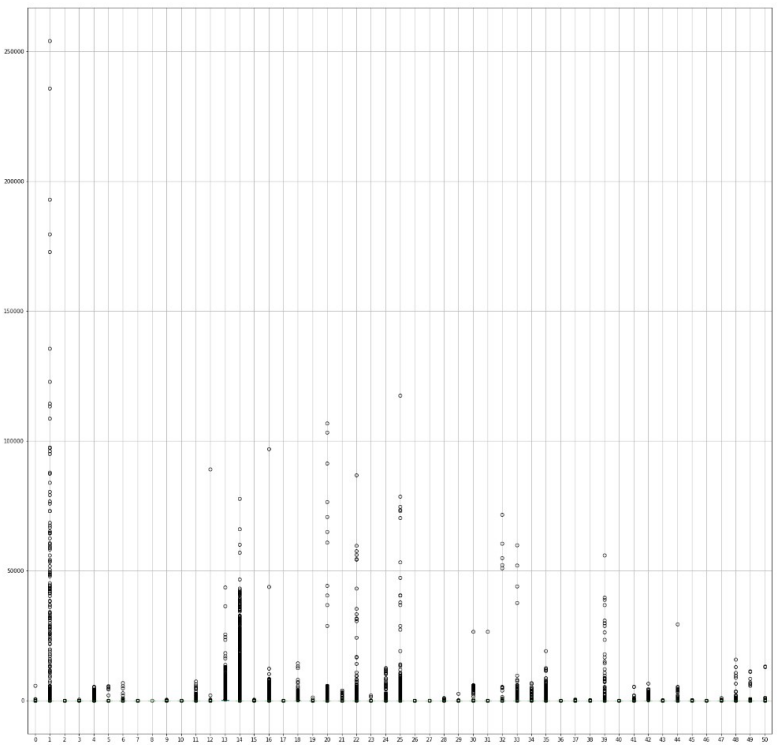
The dataset is loaded from the UCI repository in which each file contains observations grouped by years and month and each row was organized like a dictionary $\rightarrow$ KEY:VALUE. To reduce the dimension of the dataset, we perform two actions:

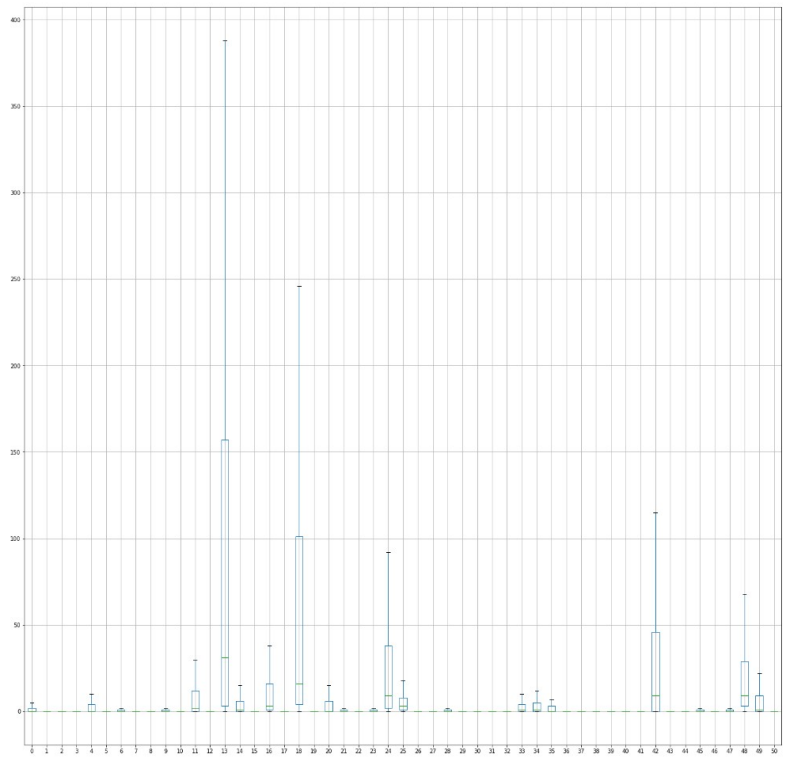
- Checking and removing duplicate observations
- Checking and removing features with all zeros

At the end of feature reduction, we have a dataset with 80248 observation and 265 columns. Starting from 482 features, the reduction was 54.97%. Also the dataset after the cleaning is more balanced respect to labels of samples.

Exploratory analysis - Boxplot

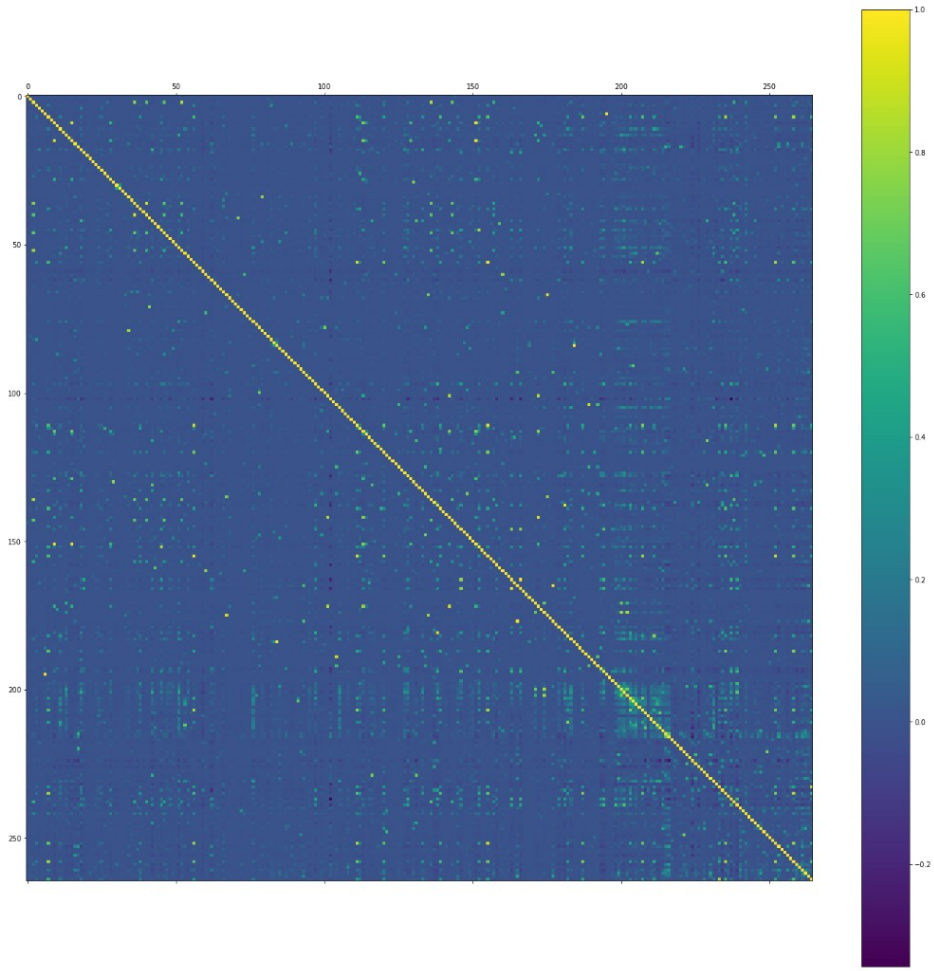
As one can see from the boxplot of the firsts 50 feature (after the data cleaning) the data are not well distributed (on the right the same boxplot without outliers to make visible boxes and whiskers).





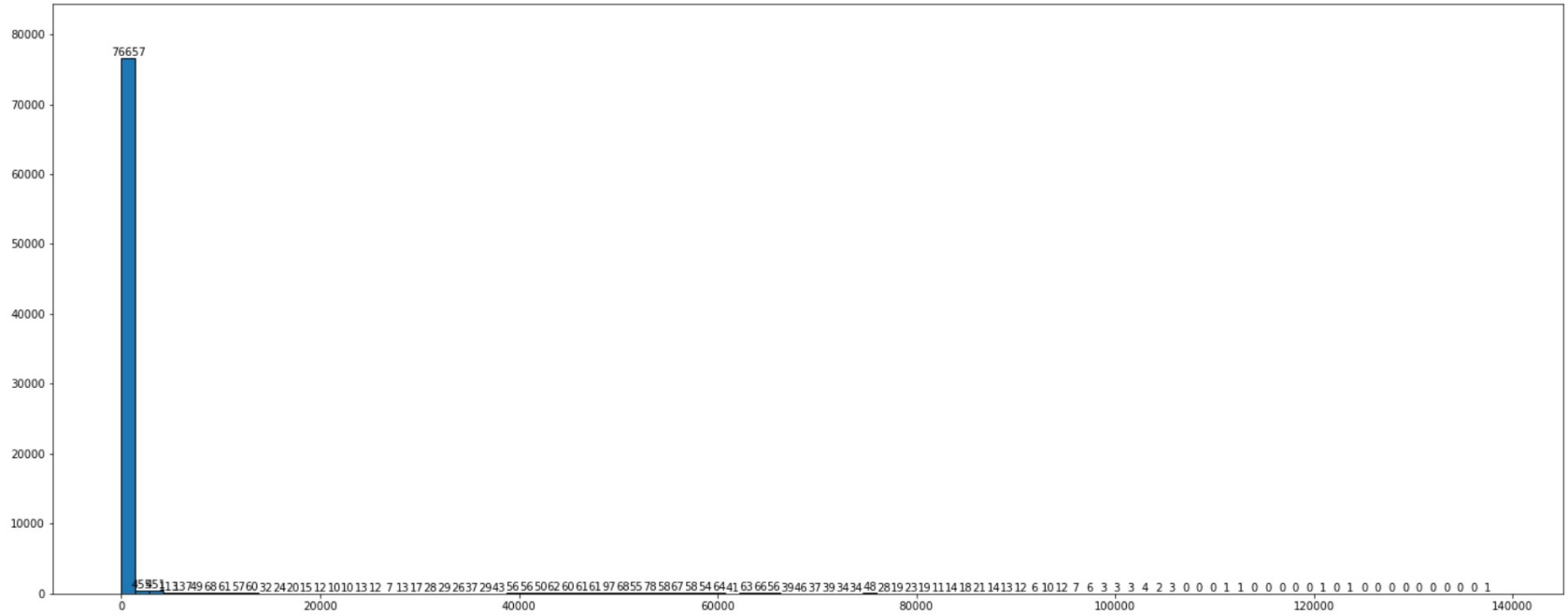
Exploratory analysis - Correlation Matrix

As one can see from the correlation matrix, in general there isn't correlation, except for data among feature about from 200 and 215 in which there is a bit of correlation



Exploratory analysis - Standard deviation

In general dataset has high spread. 66 feature has std over 100 (some arrives over 1000) while 84 has std under 1. This difference of spread arise probably because of numeric discrete feature on one side (which in many cases have an high concentration at the starting values but then have sparse very high values) and categorical dichotomous feature on the other side. Under is reported the distribution of feature 81, which is the one with the higher std.



Unsupervised learning - Clustering

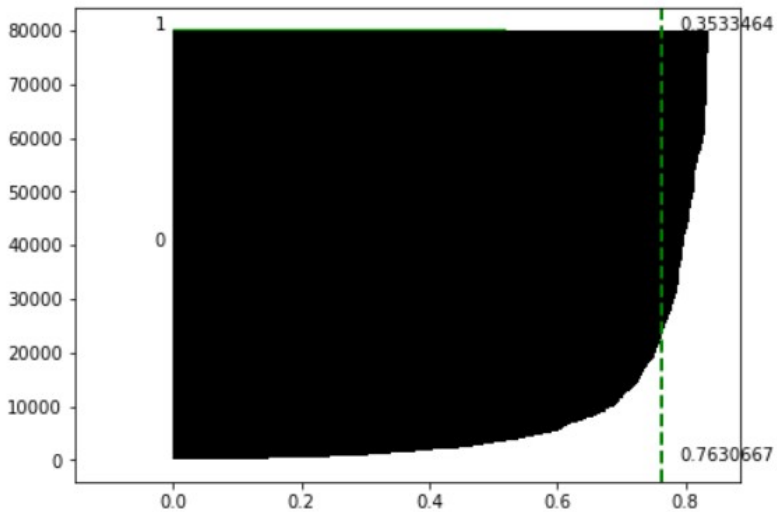
We performed various k-mean clustering methods for k = (2,3,4,5) on the data after the PCA. Down below we report the results for each k-mean:

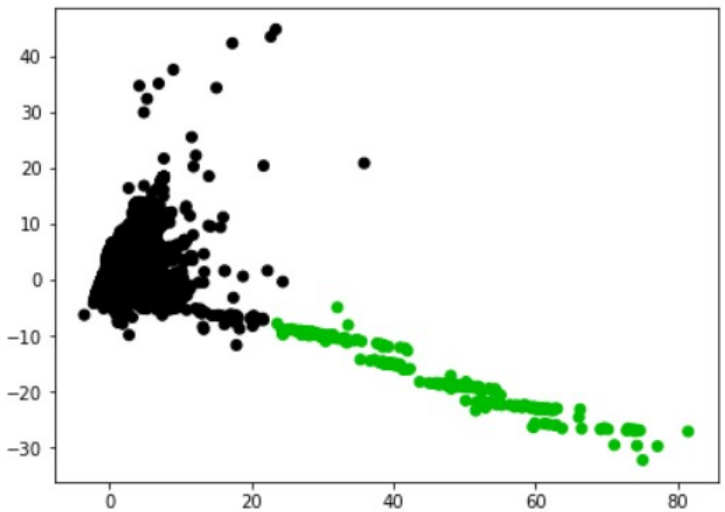
- k = 2 , overall clustering silhouette = 0,7616
- k = 3 , overall clustering silhouette = 0.6506
- k = 4 , overall clustering silhouette = 0.2123
- k = 5 , overall clustering silhouette = 0.1867

We can see that the best result is given by the k = 2 with an overall clustering silhouette equal to 0.7616. Visually this can be seen by the stripe in green.

Unsupervised learning - Clustering

The data are principally grouped in a main group (the dark one) and a very little part of data in another group (the green line of data in the bottom right part).





Unsupervised learning - PCA

In this case we have used PCA more to decreasing the dimensionality of the dataset than to exploratory analysis, but in any case it seems to give us some information. To perform PCA, we have before scaled the dataset, and we choose to mantein only the first 134 principal components that explain 80.1% of the variance.

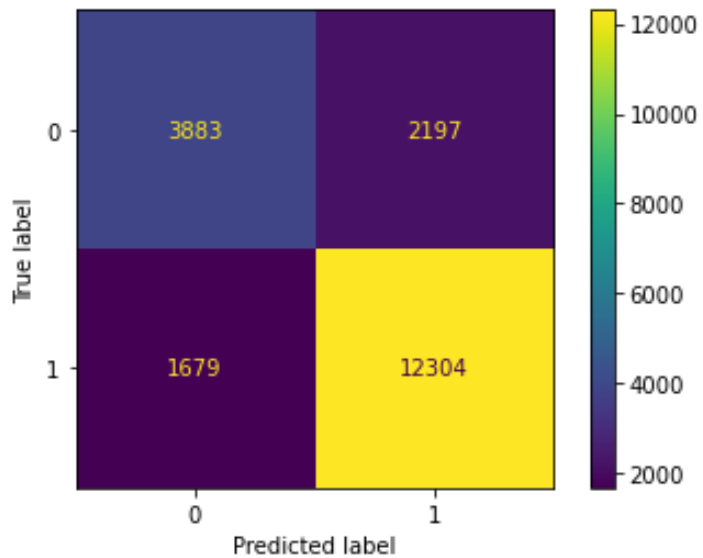
Supervised learning

To performe predction, we built three machine learning models:

- Multilayer Perceptron Classifier (with Grid Search for MLP Classifier)
- Logistic Regression
- Linear Support Vector Machine

Multilayer Perceptron Classifier

MLP is a method used to find nonlinear relationship using deep neural network. First, we build simple MLP Classifier choosing simple dense network composed by 3 hidden layer with 32, 16 and 8 neurons, this in order to classify if the observation is or not a Malware (0 is not Malware, 1 is Malwere) based on the labeling computed before. We used sklearn package and left default hyperparameters except for activation function of output layer, choosing Sigmoid. Under is reported the relative confusion matrix.



Supervised learning - Multilayer Perceptron Classifier

The result of this model was very good compared to the next two models:

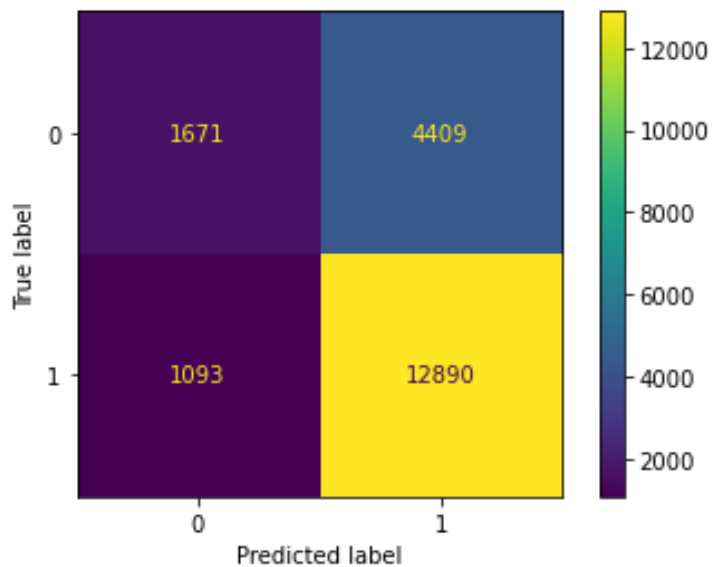
- Accuracy: 80.68%
- Balanced Accuracy: 75.92%
- Precision: 84.84%
- Recall: 87.99%

We also performed GridSearch to find the best combination of Hyperparameters on MLP Classifier using sklearn and get the highest accuracy 81.44% so our model was near to the best one.

Supervised learning - Logistic Regression

Logistic Regression is a type of regression where given an input value, it returns the probability of belonging to the output class. The shape of this function is called Sigmoid becase the *y-limit* goes between 0 and 1.
The result of our model is worst than the previous model:

- Score: 72.57%
- Balanced Accuracy: 59.83%
- Precision: 74.51%
- Recall: 92.18%

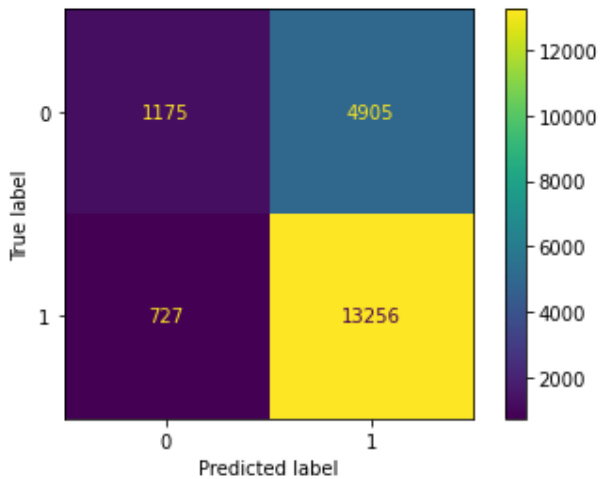


Supervised learning - Linear Support Vector Machine

Support Vector Machine is a Machine Learning model whose goal is to find the best hyperplane that allows to separate classes on the feature spaces.

The result of our model is worst than the previous model:

- Score: 71.88%
- Balanced Accuracy: 57.55%
- Precision: 73.26%
- Recall: 93.92%



Note: we don't used polynomial transformation because of the huge amount of data and consequential amount of time execution.

Conclusion

In conclusion, we have observed that the dataset it's not well distributed and there isn't correlation among the data, that said we get good results with the method used. Particularly the best model is Multilayer Perceptron Classifier, with overall better metrics and an accuracy of 81.44% with GRID search.