Одеса-Черкаси, 24-26 травня 2017 р. – Черкаси: Видавець Ольга Вовчок, 2017. – С. 255-258.

MICROSOFT MALWARE PREDICTION COMPETITION

Heorhii Kalaichev
Maksym Shpakovych
Maxim Sidorov, PhD, associate professor
Kharkiv National University of Radio Electronics

The main goal of this work is to show the ways of preparation the amount of data, building a classification model on the huge dataset and evaluating resulting model on test data. Initial problem which was solved in this work was taken from Microsoft Malware Prediction Competition from Kaggle site. This task is an appropriate for our goal since training dataset contains different types of features for preprocessing and 9 million of rows.

I. INTRODUCTION.

We build our research with using of CRISP-DM methodology which contains the following steps: business understanding, data understanding, data preparation, modeling, evaluation and deployment. We use each of these steps to create a structure of current work and follow it until have a success.

II. BUSINESS AND DATA UNDERSTANDING.

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 [1].

III. DATA PREPARATION.

The dataset contains 81 columns (without unique identifier and target column with class's labels) which should be used to predict will machine be attacked soon or not. Before talking about column's type, we want to tell about tools for working with data. We used Python language to process data: pandas for working with CSV tables, sklearn.preprocessing module to prepare data before model training.

Columns have different type and can not be used to build a classifier without preparation step. The basic and common used approach to work with nominal or categorical type of features is to use one-hot encoding approach. Then, some of categorical features had obviously too many unique values. It was decided to remove nominal features which have more than 15 unique values.

Then, we had to do something with missing data. First of all, we calculated how many NaN data were in each column and than removed columns with more than 60% of NaN values. Other NaN values were left since we use tree-based model which is resistant to missing data.

After all these steps we got a sparse table with 1701 columns. Since data is sparse, it is useful to store this matrix as a sparse matrix. However, entire dataset contains 9 millions of rows which produces the problem with storing this data into memory. For fast training of the further models sparse matrix was stored on disk by chunks of size 100 000 rows and was loaded on demand.

One more important thing is that before training models, people do shuffle of data to create more robust model. But it is not trivial task how to shuffle 9 millions of rows. To do such task we decided to use tempfile package from standard python libraries which allows to store information on disk and have an access to it in efficient way.

After all of these steps, we obtained shuffled dataset with encoded features, filtered by NaN count and stored by parts on disk in sparse format. The same technique was applied to test.csv table, except shuffling.

IV. MODELING AND EVALUATION.

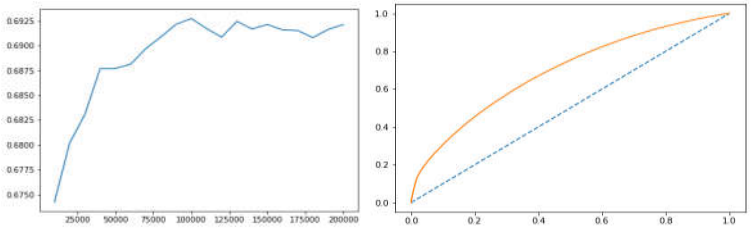
The best kind of classifiers for such high dimensional data with missing values are tree-based classifiers. In our research we used one of the state-of-art tree-based ensemble classifier XGBoost.

The main problems during modeling are selecting model type and finding optimal hyper-parameters. To solve first step we relied on our experience and chose one of the state-of-art classification algorithm – xgboost. What about the second step, the most popular approach is a Grid search with Cross validation. We applied this

technique and found following optimal hyper-parameters for XGBClassifier estimator.

Since we have amount of data, the important thing that we should do is to select the volume of data which is enough to train robust model. This can be done in simple way: training model on the different sizes of data (increasing sequence of size), plot scores and detect optimal count of rows for training.

On Figure 1a we can see that 100 000 rows of shuffled data are enough to build the model. Test set for scoring was 1 million rows. We used Area Under Curve score for evaluating the model since this is a basic metric for classifiers selection and also it was a rule in Kaggle competition. On Figure 1b we can see the ROC curve for the best classifier which we got during our work.



(a) ROC AUC score for different train data size

(b) ROC curve for the best model

Figure 1. ROC AUC scores and curve for best model

Results. During this work, classifier which can predict if machine will be attacked soon by malware or not, was created. The best score which we achieved on the test set was 0.6927 by ROC AUC score. Also, since tree-based model can be well tractable, we defined that the most important feature for classification was SmartScreen_Blocked, which means that if SmartScreen option was blocked by user then there is a high probability that machine will be attacked soon.

REFERENCES

1. Microsoft Malware Prediction Competition, https://www.kaggle.com/c/ microsoft-malwareprediction/data.

2. Tianqi Chen, Carlos Guestrin. XGBoost: A Scalable Tree Boosting System, 2016.