CSI4900 Honours Project

Microsoft Malware Prediction

Student: Xiaoxin Zhou
Student number: 7957115
Supervisor: Verena Kantere

Introduction

The cybersecurity industries grow rapidly in the last decade, and will likely continue to increase in the future, as more individuals and businesses rely on computers frequently (Zacks, 2018). Microsoft collected enormous data and posted a cybersecurity-related challenge on Kaggle, a data science community, let everyone participates in and generates technologies to predict the malware occurrences. Thus, this project is investigating different machine learning algorithms on this dataset. The motivation for choosing this challenge is exploring the performance of supervised machine learning algorithm on malware prediction. This paper will provide an insight of the project in terms of three phases, preprocessing, modelling and evaluation.

Preprocessing

Firstly, this section discusses the details of the environment was applied in the project. Using a cloud computer with high performance is a key improvement when working on a large dataset. Therefore, the option of using the Amazon Web Service (AWS) EC instance was considered. However, building a project on a cloud instance has few major drawbacks including overhead connection and memory limits issues. In the end, this project decided to uses a local computer to preprocess the datasets with more optimized methods. For example, encoding a string column to integer would take O(n) run time by a linear algorithm (n is the number of the rows in the dataset), though, a vectorized (row-wise batch processing) algorithm would take O(1) run-time. In

addition, Jupyter notebook provides an interactive user interface and individual blocks for code execution, which helps users programming in a flexible way.

Secondly, getting to know the data. Since the source file is in CSV format, converting it to a readable DataFrame in Pandas. The dataset consists of 84 columns and 2,243,048 rows. In the aspect of meanings, each column, except the last one, represents a feature, or attribute. Each row stands for an object, and each object is a machine. In the aspect of data types, the features are textual categorical data and numeric categorical data. The label is an integer: 0 or 1. Moreover, because missing values cause errors in modelling and result in a wrong classification, the columns have missing values were detected. There are 65 columns have null values.

Thirdly, unifying the spelling. Since the data is collected from real machines, some values are inconsistent and noisy. There are two cases in this project. In the first case, multiple words have the same meaning but in different representations, such as *OFF, Off,* and *off*. In the future modelling step, the model would evaluate these three values are three distinct vectors. The second case has a mixture of two data types. For example, one column called *ChassisTypeName* has both string and integer values such as *Notebook,* and *30*. In fact, based on the cheatsheet given online, a string description has its corresponding numeric identifier. Thus, *30* also means *Notebook*. Then identifiers are converted into its name based on the cheatsheet.

Following, cleaning the dataset. The instruction indicates that the dataset contains not-null labels, but after checking the missing values on the label column, the last row was detected has absent label. While removing the last row, the number of columns that have missing values was decreased to 43. The columns that have high missing percentage have also been dropped. The percentage is obtained by dividing the number of rows, which have null values in that column, by the total number of rows. If the percentage greater than 50%, the column is dropped. The columns and their missing rates are illustrated in Figure 1.

Name	Portion*
PuaMode	99.97%
Census_ProcessorClass	99.59%
DefaultBrowsersIdentifier	95.14%
Census_IsFlightingInternal	83.03%
Census_InternaBatteryType	71.03%
Census_ThresholdOptIn	63.50%
Census_IsWIMBootEnabled	63.41%

Figure 1

There are four additional strategies for handling the missing values. The first strategy is filling the missing cell with the most frequent value in that feature only when the number of rows of the most popular value is far away greater than the second popular one. For example in Figure 2, the first column indicates the feature has 6 categorical values and the second column shows the number of rows that have that

value. In this case, the missing values were filled with *7.0* because *7.0* occurs about 45 times than *0.0*.

Values	Number of rows
7.0	2,175,227
0.0	47,864
8.0	5,596
5.0	5,138
3.0	749
1.0	409

Figure 2

The second strategy is also filling a popular value, but depend on other columns. For example, the *Country* feature and *City* feature are correlated. The *Country* column doesn't have any missing values but the *City* column does. Calculating the most frequent city for each country from the rows that have non-null values in both columns. Then when filling the missing cell for *City* feature, used its country and filled the cell with the most frequent city of the country.

The fourth strategy is distribution. Since attributes are categorical, the probabilities of exist values are able to be calculated. For example, v1 occurs 85%, v2 occurs 10% and v3 occurs 5%. Then randomly filled the missing cells based on the probabilities, thus, there is 85% chance to select v1 to fill the cell. This method leads to less bias than previous strategies.

The last method is a machine learning algorithm called K-nearest neighbours (KNN). The idea of KNN can be illustrated in Figure 3, which is retrieved from Tavish's blog (2018).

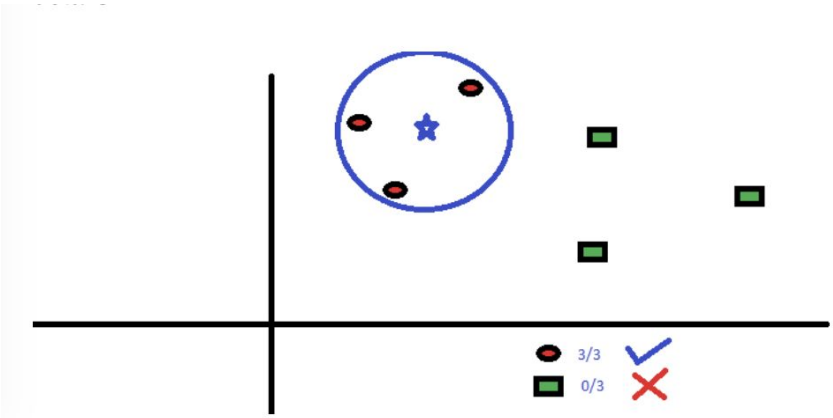

Figure 3

There are two values in the space: red circles and green rectangles, the goal is to find the class of the blue star. In this example, the k-value is defined as 3. The approach makes the blue star as the circle centre and lets the circle is just big enough to enclose three closest points. Thus, 3/3 is greater than 0/3, the star is classified as red circles. In the real dataset, the choice of k is a critical part. A small k denotes that the noise data has an impact on the result, and a large k causes costly computing. In the project, k is six, and these six attributes initially don't have missing values. Bias easily occurs because the target column may have no correlation with these six training columns.

Last but not least, converting non-numeric data types into integers. Once the data types are consistent, splitting the dataset into training and testing datasets. The approach is selecting the rows with index 0, 3, 6, … 3n into a testing dataset, the other rows into a training dataset.

Modelling

In supervised learning, there are two types of algorithms: classification and regression. The main difference is the outcome type. In classification, the label is categorical or discrete; however, in regression, the label is numerical or continuous. The label in this project is a binary, thus it is a classification problem. Classification models include several types, such as tree-based, rule-based etc. The models used here are tree-based: the decision tree structure and the random forest structure. In the decision tree, the internal nodes indicate the value tests, the branches denote the different outcomes of the test and the leaves denote the class of that branch. In random forest, each classifier is a decision tree classifier and is generated using a random selection of attributes at each node to determine the split. The performance of data preprocessing and modelling can be presented in the evaluation phase.

Evaluation

In the evaluation phase, the evaluation metrics compare the prediction results with the actual labels. Two methods are implemented: mean absolute error (MAE) and F1 score. The performance is summarized in Figure 4.

	Decision tree	Random forest
MAE	0.440	0.443
F-Score	0.560	0.596

Figure 4

The best performance is the random forest structure with F1 Score and the worst performance is the decision tree with MAE. Comparing the results vertically, random forest performs better than the decision tree. To figure out the reasons behind it, need to know the limitation of the decision tree. There is a concept named overfitting, which means a model that modelling the training data so well. Jason(2016) states that it happens when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data. A decision tree is prune to overfitting especially the tree is deep because the tree wants to cover all the data. Random forest overcomes this problem since it randomly selects trees and averages the result. Comparing the results horizontally, F1 score performs better than MAE because of the data type. By definition on Wikipedia, MAE is a measure of the difference between two continuous variables. Unlike the categorical data, continuous data can be measured, such as weight. If the predicted weight is 110lbs and 120lbs, while the actual data is 90lbs and 100lbs, the MAE = [abs(110-90) + abs(120-100)]/2 = 20. In this project, the label is discrete, which only has value 0 or 1. The differences

between any pairs of these two values are always 0 or 1. Hence, MAE will tells more information for continuous data rather than discrete data. Besides, the F1 score is the harmonic mean of precision and recall. Precision refers to the percentage of the results are relevant, and the recall refers to the percentage of total relevant results correctly classified by the model. Compared with MAE, F1 score is more suitable for evaluating classification algorithms. Therefore, it is essential to choose proper evaluation methods for different models.

Future work

The evaluation results show there is a large room for improvement. In fact, the quality of data determines the ceiling of prediction accuracy, meanwhile, the modelling algorithms are trying to achieve this ceiling. For improving the quality of data, feature selection and missing value handling are critical. Because not all features have a correlation with the label, the unnecessary features disorder the modelling. Thus, these features need to be removed by assumption or statistics. Moreover, there is no general way to fill the missing values, try multiple methods and compare them might help. In addition, learning from other people is also helpful. For instance, go to the discussion board on Kaggle, and learn from other approaches.

References:

Microsoft Malware Detection. Kaggle. Retrieved from: https://www.kaggle.com/c/microsoft-malware-prediction/data

Zacks, A. (2018, October 28). *Malware Statistics, Trends and Facts in 2019*. Retrieved from https://www.safetydetective.com/blog/malware-statistics/

Srivastava, T. (2018, March 26). *Introduction to K-Nearest Neighbors.* Retrieved from: https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/

Brownlee, J. (2016, March 21). *Overfitting and Underfitting with Machine Learning Algorithms*. Retrieved from: https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/