UNIVERSITY of CALIFORNIA · IRVINE

Malware Detection Using Ensemble Methods

Jungin Hong, Rikako Yamamoto, and Steven Seader
{junginh, rikakoy, sseader} @ uci.edu

Background

Cybersecurity risk is rated a top three global risk in 2019. Malware attacks are one of the fastest-growing areas of cyber risks. Malware detection is an extremely important ability of any business seeking to provide personal safety and privacy.
Our project seeks to predict whether a computer would become infected based on the device's configuration using various ensemble methods.

Dataset

The dataset we used was extracted from Kaggle Microsoft Malware Prediction [1]. In our project, we used 500,000 individual machines information. Each machine in the dataset has 83 features (though there are some machines that do not have data for a given feature), most of which are categorical features, the rest being numerical.

Our data was randomly split 80/20 for training/testing respectively:

- Training: 400,000 machines
- Test: 100,000 machines

Preprocessing

- First, we removed features that consisted of more than 99% missing data or were highly skewed.
- Second, features that had over 10% missing values were manually replaced with placeholder values of our own.
- Third, we used the correlation matrix to find the pairs of features with a high correlation coefficient and remove the feature with less unique values.
- Lastly, we changed all categorical to numerical values.
- Final features reduced to 65 features.

Models

All methods we used were based on a decision tree algorithm.

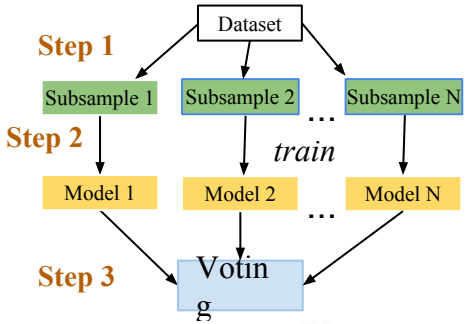
1. Decision Tree

We used entropy to decide the criteria.

$$Entropy = \sum_{c \in class} -P(c)\log_2 P(c)$$

2. Bagging (bootstrap aggregation)

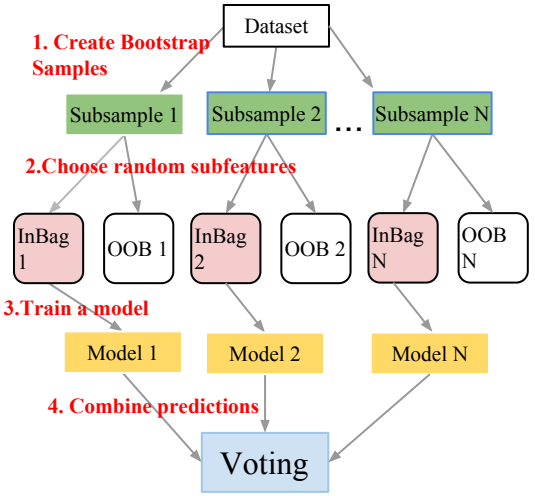
1.This method uses Bootstrapping (a resampling technique to create a random sample of the original data with replacement) to obtain a set of samples.
2. Train a model on each sample to get prediction.
3. Compute the average of the predictions



inference $\hat{f} = \frac{1}{N}\sum_{n=1}^{N} \hat{f}_n$

3. Random Forest

Random Forest is an ensemble of decision trees, trained with bagging method. Unlike bagging, this method uses a random subset of features to make a splitting decision for each level of the tree.



4. AdaBoost (Adaptive Boosting)

Like Bagging, Adaboost is an ensemble method (in our project, of Decision Trees). This method iteratively learns weak classifiers with respect to a distribution of adding them to final strong classifier.

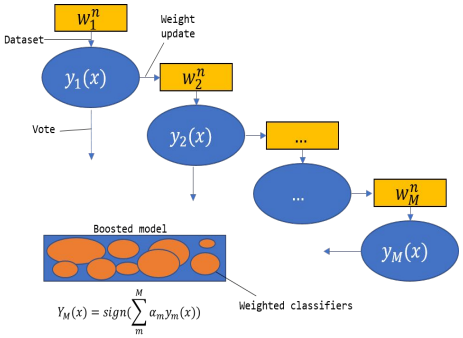


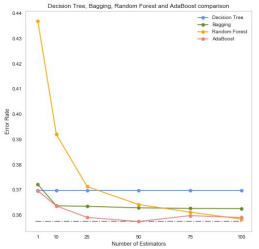
Fig 1. [2]: Boosting updates weight of each data and feedback to next classifier

References

[1] Dataset link: https://www.kaggle.com/c/microsoft-malware-prediction/data
[2] Boosting_visual_approach [Digital image]. (n.d.). https://www.python-course.eu/images/Boosting_visual_approach.png
[3] C. M. Bishop, Pattern Recognition and Machine Learning. New York, NY: Springer New York, 2016.

Results

Model	Hyperparameters	Test Accuracy
Decision Tree	Max_depth = 10	0.6303
Bagging	n_estimators = 100	0.6374
Random Forest	n_estimators = 100	0.6416
AdaBoost	n_estimators = 50, learning_rate= 0.06	0.6425



Logistic Regression with Top 10 AdaBoost/Random Forest feature selection

Model	Accuracy
AdaBoost	0.5407
Random Forest	0.5131

AdaBoost can get the best performance with smaller number of trees while bagging and random forest requires larger number of trees. Thus, Adaboost is the best learning model in terms of both speed and accuracy.
Also, Adaboost is useful for feature reduction (figure to the right).

Future Work

There are a number of tweaks and alternate routes this project could take in future work. Perhaps the most substantial addition to our project would be to attempt many of the same techniques, but instead of only using decision trees, we use an assortment of classifiers. This would add several new angles and potential branches to the work done here. Another potential path for future work could lie in finding exact measures of what features would betray the presence of malware; obviously not all malware use the same exploits, nor are all potential exploits necessarily utilized, but further research into exactly what aspects of the machines in question allow them to be compromised (as opposed to the current project of which features are simply present) may provide valuable insight to further heighten success rates.