CS 5350/6350: Android Malware Detection

1 Introduction

Each example in this classification task is an Android application. The goal is to predict whether the application is malicious or not. This data was recently published at a workshop on security and privacy analytics [1]. Each application was fingerprinted by observing the system calls it made during execution. Each feature in the feature vector corresponds to the number of times a particular system call was made by it. (The paper discusses two feature sets, we provide only the first one here.)

2 Data

The data-splits directory contains the following three data files (one training set and two test sets):

1. `data-splits/data.train`: This is the training set, in the usual lib-SVM format. There are 7597 training examples.

2. `data-splits/data.test`: This is the set of examples on which you will report results in your final report. There are 2531 test examples.

3. `data-splits/data.eval.anon`: These 2532 examples are all labeled positive in the provided data set. You should use your models to make predictions on each example and upload them to Kaggle. See below for the format of the upload. Half of these examples are used to produce the public leader board. The other half will be used to evaluate your results.

[1] Dimjašević, M., Atzeni, S., Ugrina, I. and Rakamaric, Z., 2016, March. Evaluation of Android Malware Detection Based on System Calls. In Proceedings of the 2016 ACM on International Workshop on Security And Privacy Analytics (pp. 1-8). ACM.

In addition, the directory also contains a file called `data-splits/eval.ids`. This file has as many rows as the data.eval.anon file. Each line consists of an example id, that uniquely identifies the evaluation example. The ids from this file will be used to match your uploaded predictions on Kaggle.

In all, there are 360 features. Note that as part of your project, you are welcome to try feature space expansions, kernels and other non-linear methods.

3 Evaluation

The data is imbalanced – there are many more negative examples than positives. We will use the F1 [2] score to evaluate the classifiers. The examples are all split randomly among the three files. So we expect that the cross-validation performance on the training set and the F1 scores on the test set and the public and private splits of the evaluation set will be similar.

4 Submission format

Kaggle accepts a csv file with your predictions on the examples in the evaluation data. There should be a header line containing `example_id,label`. Each subsequent line should consist of two entries: The example id (from the file `data.eval.ids`) and the prediction (0 or 1).

We have provided two sample solutions for your reference:

1. `sample-solutions/sample-solutions.all.positive.csv`: Where all examples are labeled as positive
2. `sample-solutions/sample-solutions.half-neg.csv`: Where the first half of examples are labeled false and the second half are labeled true

[2] `https://en.wikipedia.org/wiki/F1_score`