Malware Classification

TEAM-EMMA :
Prajay Shetty
Hiten Nirmal
Maulik Shah

Data Preprocessing

Cleaning: Removing extra 00s, CCs and other spaces.

Line vs File wise processing

N-Grams

Hashing TF

Naive-Bayes

Pipelining in Spark (File wise)

Issues faced:

Feature indexing problem in HashingTF

- Use of pipeline.

Category Indexing: Naive-Bayes starts its labels from 0.

- StringIndexer and IndexToString

Random Forest with variety-200 Random Cluster

Creating a Data Frame- File wise

DataFrame-[Text, Label]

Passing DataFrame into Pipeline

Feature Extraction- tokenizing, 2-gram, Hashing TF ,

Re-Partitioning the data to boost the boost the performance

Algorithm- Line Based MLP-Sudo Code -1/2

1.Read all the urls [f1 ...fn] from the master file list

2.Read the data present in the url location into the RDD

3.Split each line in the file as (index,line,label) .

4.Randomly select k lines for each class label (base sampling) (Embedding)

5.Apply Hashing TF, request 200 feature vectors against 16 features present in the training set for each line.

6. Apply one hot encoding to prepare labels for one vs all modelling.

7.Training C models , but while training randomly select m lines from each class and send it to model for training at each training sequence (we have used just 60,000 samples for each class training).

Algorithm 2/2

8.For each line present in the test set , perform prediction for all the lines. (key,[c1...cn])

9.Sum all the prediction for that particular index

10.Select the class as label for that particular file which has received maximum prediction.

Models Used

First 3 classes are using SVM and rest 6 classes are using Logistic Regression in Spark

We also experimented with MLP network , but it didn't given best performance compared to above classes.

We also faced class imbalancing issue, we solved the problem with applying sample first and then using limit

We have received an F1 score of 0.87+ for at least 6 classes compared to 9 classes for random-validation

Performance Tuning

Setting spark properties to use the maximum of the clusters

Updating settings to use memory and disk both for storing data-frames.

Repartitioning

Setting a high-end master node cluster. - **Trained 60% of training data in 30 minutes for Naive Bayes.**

Q&A

Thank You.