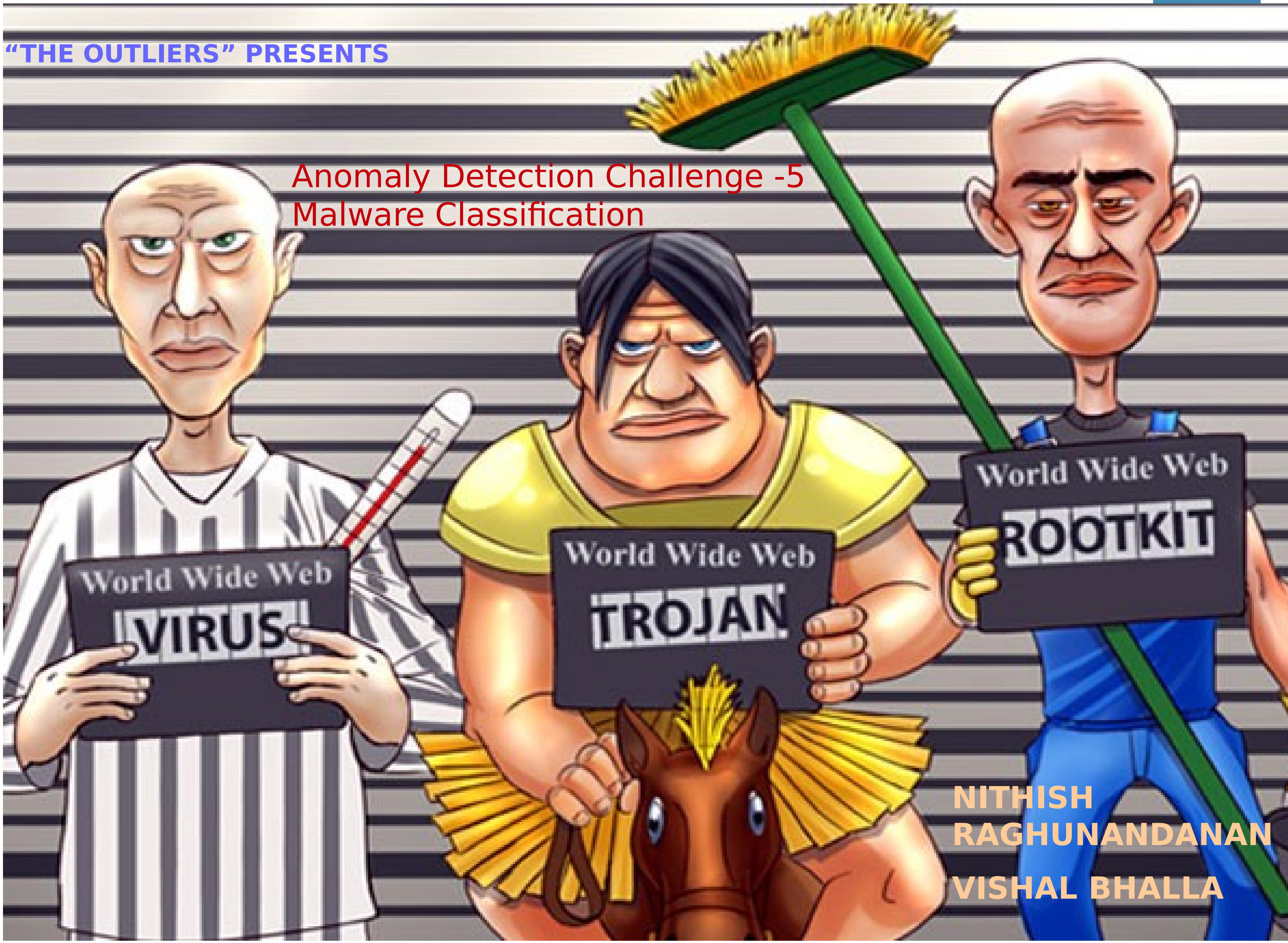
“THE OUTLIERS” PRESENTS

Anomaly Detection Challenge -5
Malware Classification

NITHISH RAGHUNANDANAN

VISHAL BHALLA



Picture courtesy Inquirer.net

Agenda

Introduction

- Aim: Classify Malware into predefined families (0…9) or Outlier (10).
- Problem Type: Multi-class Classification
- Samples
  - Training Set: 2042 (257 Empty Training Data)
  - Test Set: 706 (30 Missing Test Data)
- Number of Features: Behavioral Data(Function Call Stacks)
- Classification
  - There are 11 decision classes: 0 to 10.

System Pipeline

Data Visualization → Feature Selection → Evaluation of Models

- Data Visualization
  - Tabulate & analyse the structure of the data.
- Feature Selection
  - Encoding selected parts of the data as features.
- Evaluation of Models
  - Classification accuracy using K Fold Stratified Cross Validation.

Data Analysis

- Function Call Stacks
  - ✓ Variable number of function calls per sample
    - – 0 to 9353
  - ✓ Analysis of functions
    - – 5729 different functions

Data Preprocessing

- Handling of Empty Training Samples
  - No function call stacks for 257 samples of the training set
  - Encoded as null feature vector.
- Handling of Missing Test Samples
  - Checked Malwr & VirusTotal for missing samples' data.
  - Checked Challenge-3 test & training sets.
    - Test set contained 24/30 samples.
  - Took the best classification output for the 24 samples from Challenge 3(Accuracy of 96.778%).

Feature Engineering

- Bag of Words
  - Sparse vector of occurrence of functions.
  - Loses order of occurrence of functions.
  - Dimensions: 5729
  - Example: A->C: [1, 0, 1]
- Term Frequency-Inverse Document Frequency
  - Frequency of functions normalised by their occurrence in different documents.
  - Loses order of occurrence of functions.
  - Dimensions: 5729
  - Example: A->C: [1/Occ(A), 0, 1/Occ(C)] where Occ(x) denotes the occurences of x across documents.

Feature Engineering(2)

- State Transitions
  - Each function is encoded a state from 1.
  - Each sample is represented by a vector of the largest vector size in the sample space.
  - Empty transitions are represented by 0.
  - Dimension: 9353.
  - Example: A->C in a vector of size 4= [1, 3, 0, 0]
- Stacking
  - TF-IDF + State Transitions
  - Bag of Words + State Transitions

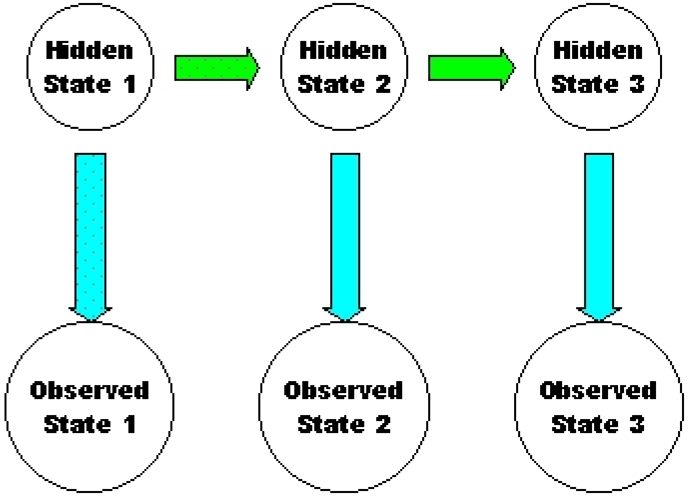
Feature Engineering(3)

- N Gram
  - A contiguous sequence of n items from a given sequence of text – here, function calls
  - ngram_range=(1, 3) – Mono, Bi, Tri Grams
  - Total words in the vocabulary - 78310
  - It didn't improve the results, and gave a memory error when the n-grams were increased.

Approaches

Sequential Data Models [dietterich2002machine]

- State Hidden Markov Models (HMM)
  - ✓ Observation – Function Calls
  - ✓ Finding the model parameters
  - ✓ Hidden state, optimal sequence?
  - ✓ Training Labels - Observation or State?



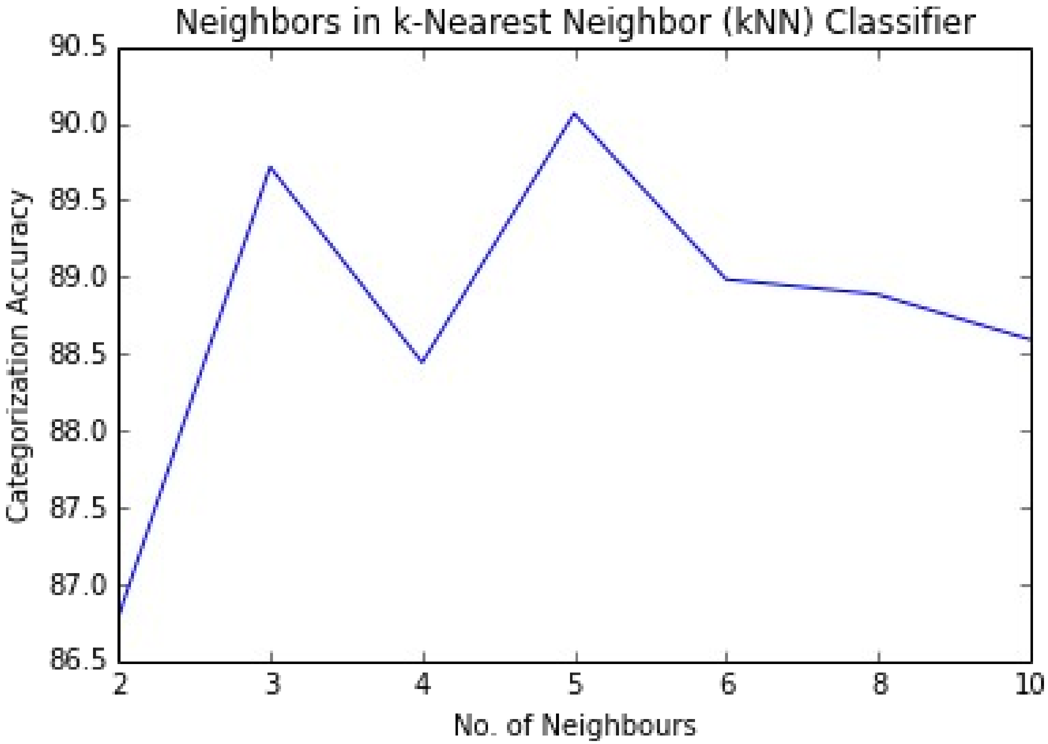
- Conditional Random Fields (CRF)
- Sliding window
- Recurrent Networks

Model

- We tried different types of Multi Class Classification Models to fit our data
  - Random Forests
  - Multi Class SVM with Linear Kernel
  - K Nearest Neighbors with 5 Neighbors
  - Ada Boost
  - Neural Networks
  - NLTK – Naive Bayes, Max Entropy, Decision Trees

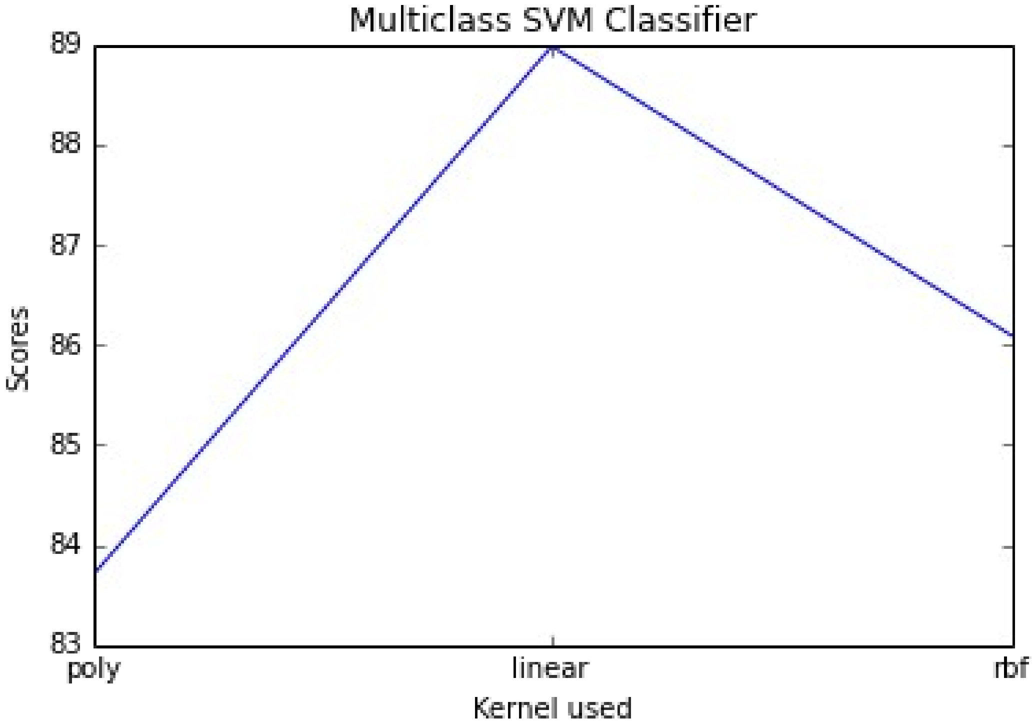
Tuning Model Parameters

K Nearest Neighbors (kNN)



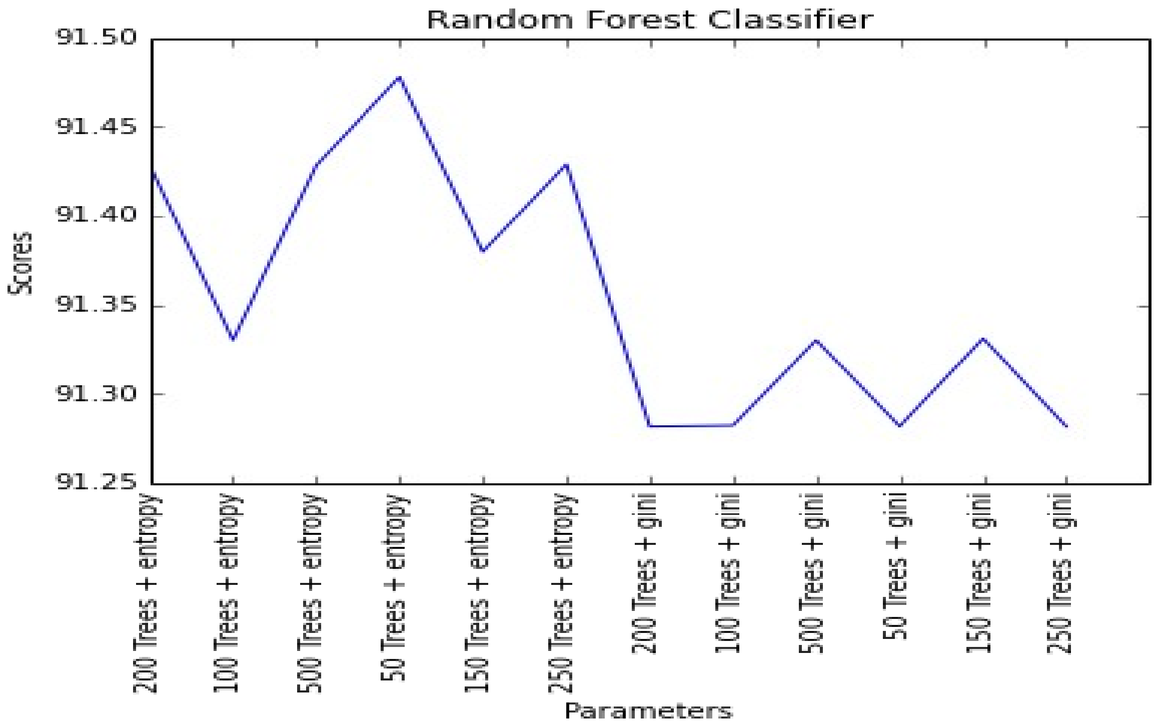
Tuning Model Parameters(2)

Multi Class Support Vector Machines (SVM)



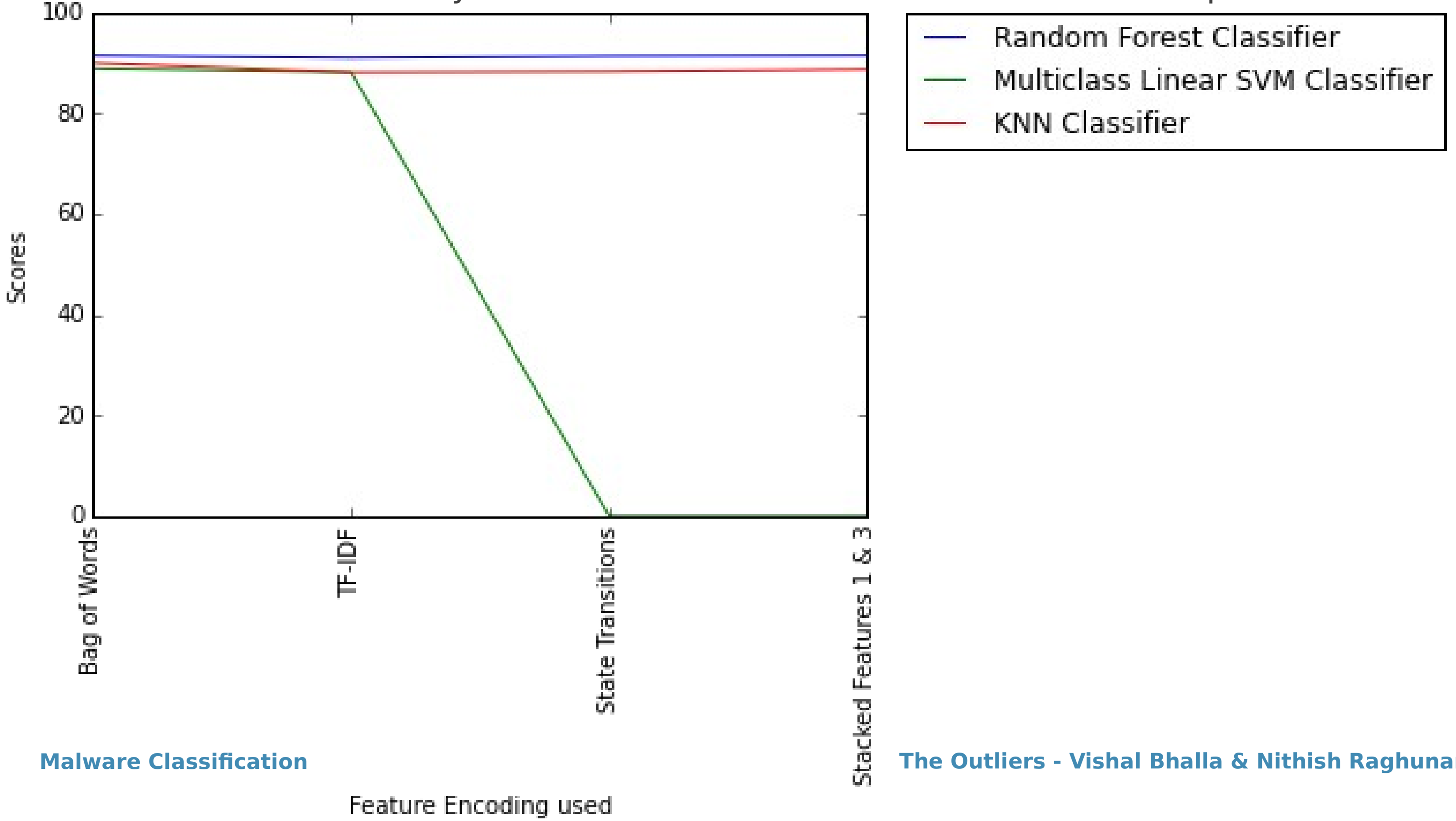
Tuning Model Parameters(3)

Random Forests (RF)

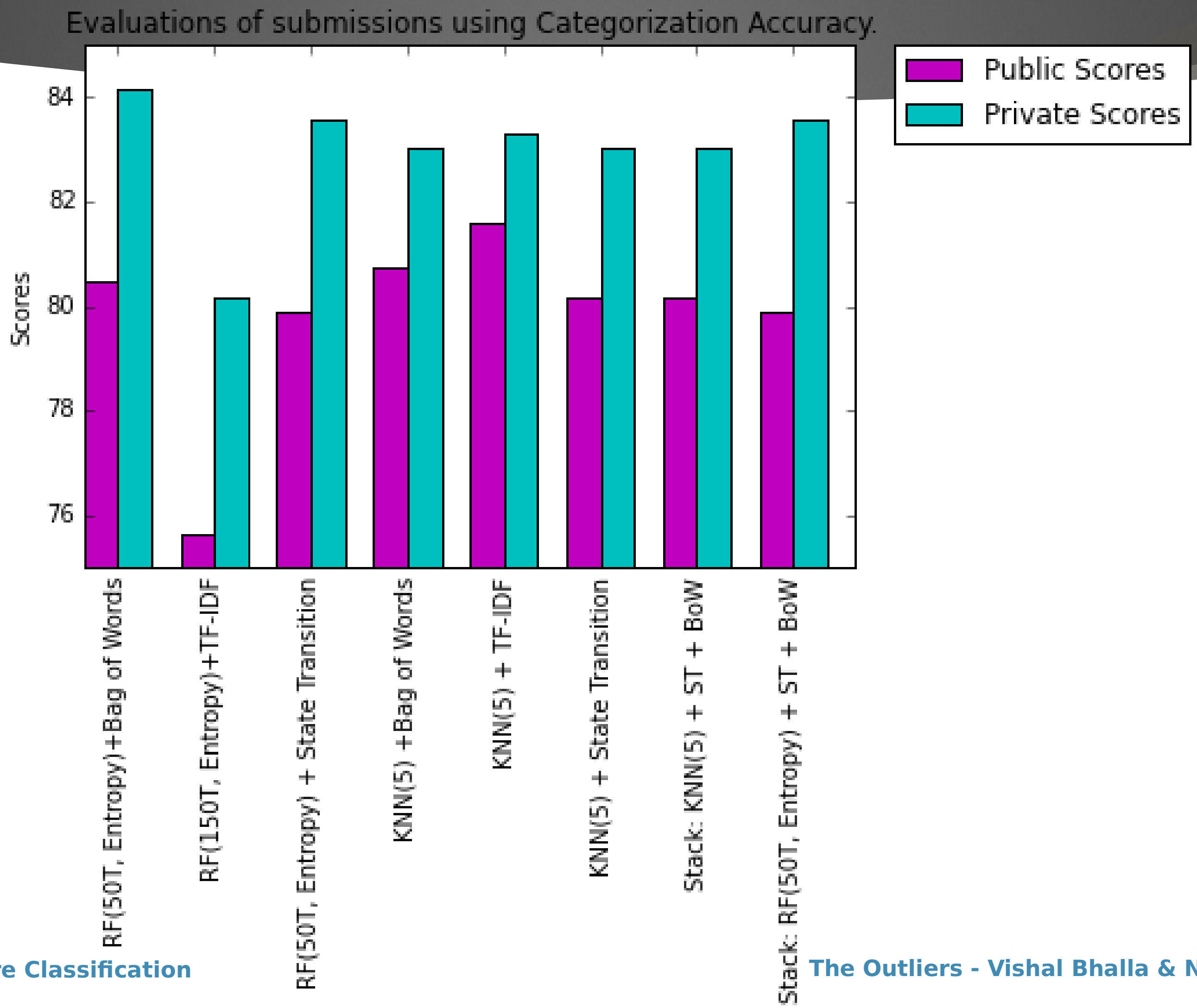


Evaluation of the Models

Classification Accuracy on Stratified K-Fold Cross Validation with a split of 4:1.



Kaggle Results



Conclusion

- The best results were observed for K Nearest Neighbors (with K=5) on Text Frequency-Inverse Document Frequency. This could be due to the fact that similar call stacks are observed for similar Malware classes.
- The stacking of the state transitions with bag of words or TF-IDF gives almost the same results.
- Random Forests & Multi Class SVM worked better on bag of words.

Key Takeaways

- TF-IDF & Bag of words for feature engineering of categorical features. The order of words are lost though
- NGrams are memory intensive
- Inherent structure of our data consisted of mainly categorical features. Many of the patterns repeated in many samples. These patterns were recognized by KNN
- Adding more features could result in not improving the classification accuracy
- Sequential data models could have helped

References

[*dietterich2002machine*]

Dietterich, Thomas G. "Machine learning for sequential data: A review." Structural, syntactic, and statistical pattern recognition. Springer Berlin Heidelberg, 2002. 15-30.

Learnings from this Course!

- Good knowhow about the entire ML Pipeline
  - Especially the importance of Feature Engineering
- Applied & in the process learnt about different models
  - Used state of the art 3rd party toolkits & libraries
  - Overfitting! → Slipped in 3 Challenges
- Smooth & steady transition:
  - Novices in IPython → Competing in Kaggle challenges
- Suggestion → More domain knowledge would have helped in Challenge 3 & 5
- Overall Worth Recommending !

Questions?

Thank You !