WOLF: Automated Machine Learning Workflow Management for Various Applications

Authors:
- Sohaib Kiani
- Sana Awan
- Jun Huan
- Fengjun Li
- Bo Luo

Authors

- Sohaib Kiani
- Sana Awan
- Jun Huan
- Fengjun Li
- Bo Luo

Contents

Motivation

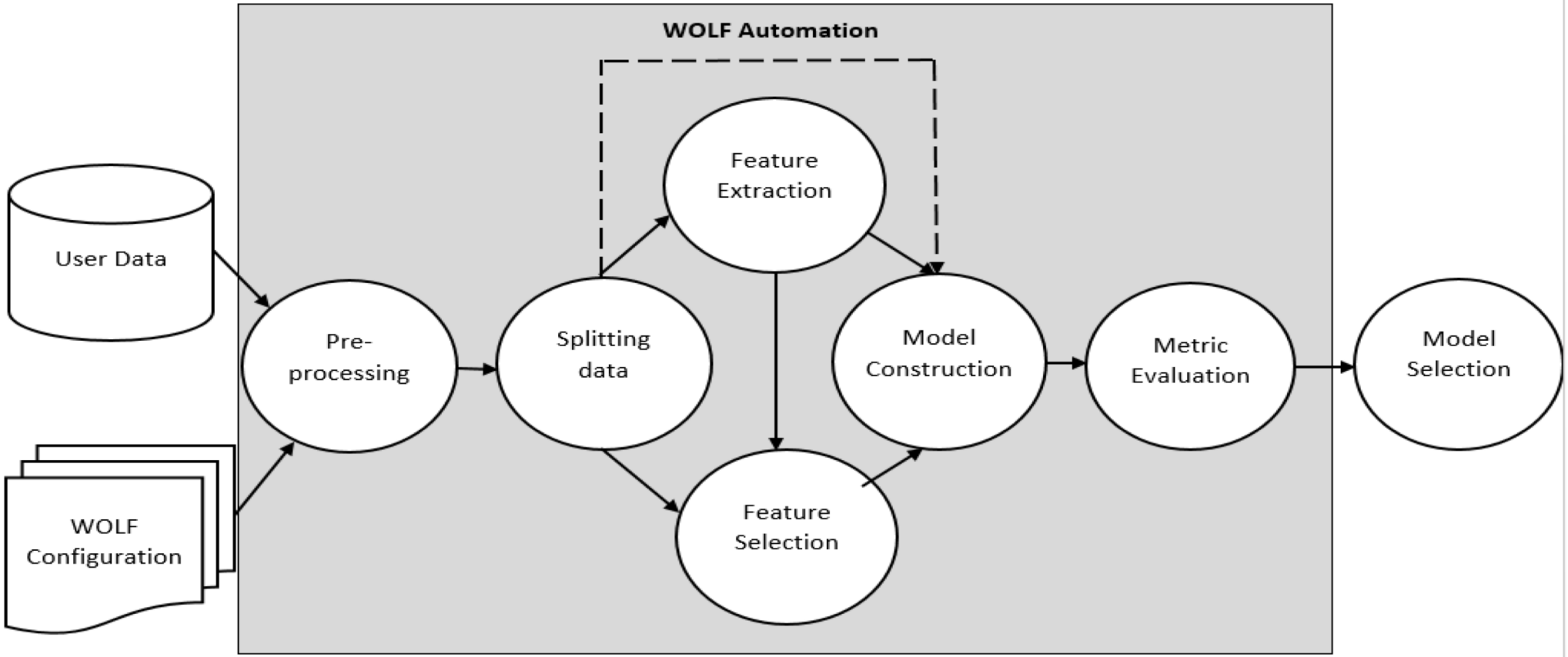
- Help novices attaining solutions without any prior knowledge of ML.
- Selecting different algorithms and their hyperparameters is a tedious task.
- Keeping track of performance with different configurations is also vital.
- Effectively use inhouse resources.

Related Work

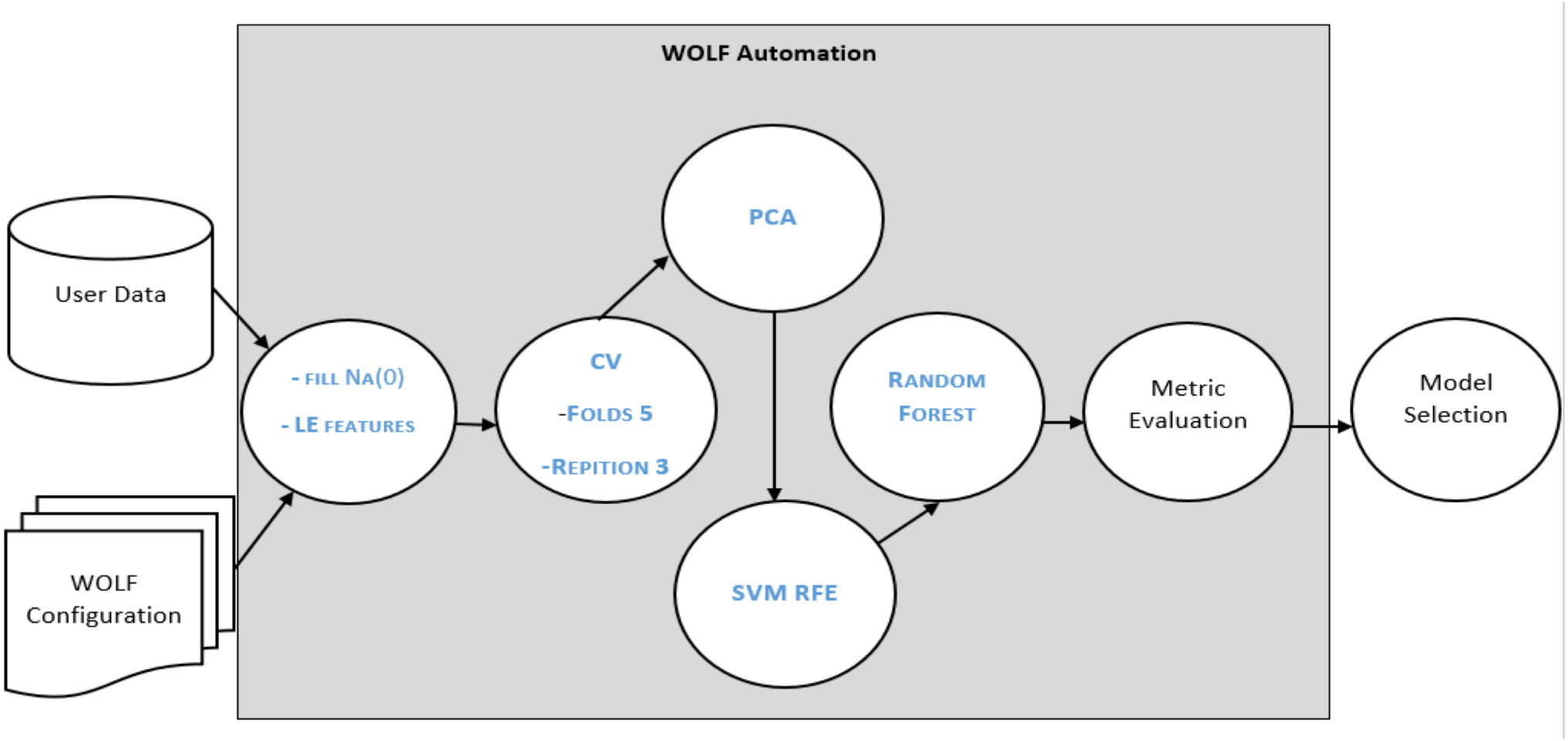
- Auto-Weka
  - Supports classification algorithm.
- TPot
  - Only takes user data file as input
  - Find ideal workflow pipeline
  - Used to work only for numerical data.

- Eibe Frank, Mark A. Hall, and Ian H. Witten (2016). The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques", Morgan Kaufmann, Fourth Edition, 2016.
- Randal S. Olson, Ryan J. Urbanowicz, Peter C. Andrews, Nicole A. Lavender, La Creis Kidd, and Jason H. Moore (2016). Automating biomedical data science through tree-based pipeline optimization. *Applications of Evolutionary Computation*, pages 123-137.

WOLF Architecture



WOLF Pipeline Sample



Configuration File

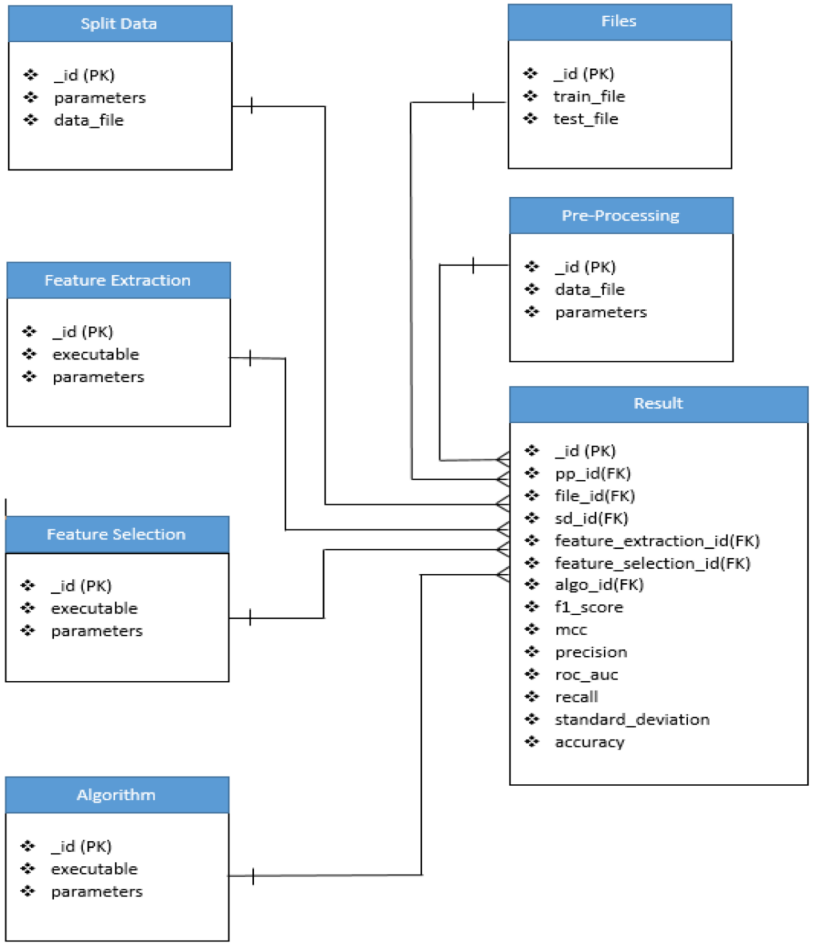
- Restricted Keywords

- Three Categories:
  - Transaction Name
  - Transaction variable
  - Parameter values
    - Single
    - Collection

```
preProcessing:
    executable: "PreProcessing.py"
    missing_value: 0
    lbel_encoding: True

datasplit:
    executable: "Splitingdata.py"
    no_of_files: 1
    data_file: "diabetes.arff"
    output_folder: "output"
    parameters:
    - ["collection","list",-k,[3,4,5]]
    - ["single",-r,2]
    - ["single",-l,"class"]

feature_extraction:
    algo1:
        executable: "PCA.py"
        parameters:
            - ["collection","list",-n,[8,10]]
            - ["single",-c,True]
            - ["single",-w,False]
    no_of_files: 1

feature_selection:
    algo1:
        executable: "SVM_RFE.py"
    no_of_files: 1

algorithm:
    algo1:
        executable: "RandomForest.py"
        parameters:
            - ["single",-d,5]
            - ["single",-t,50]
    algo2:
        executable: "LinearSupportVectorClassification.py"
    algo3:
        executable: "LogisticRegression.py"
    no_of_files: 1

metric_calculation:
    no_of_files: 1
    executable: "MetricCollection.py"
```

Database Management

- Used Non-relational Database.
- Integrated with all the transactions.
- To keep track of different configurations.



Binary Classification

DataSets

Dataset Name	No. of Attributes	No. of samples	Pos. / Neg. Samples	Percent Positive Samples
Bank note authentication	5	1372	610 / 762	0.445
Blood Transfusion Service Center	5	748	178 / 570	0.237967914
Climate Model Simulation Crashes	19	540	494 / 46	0.914814815
Connectionist (Sonar, Mines vs. Rocks)	61	208	111 / 97	0.533653846
default of credit card clients	24	30000	6636 / 23364	0.2212
Fertility	10	100	12 / 88	0.12
LSVT Voice Rehabilitation	311	126	42 / 84	0.333333333
Pima Indians Diabetes	8	768	268 / 500	0.348958333
Spambase	58	4601	1813 / 2788	0.394044773
Vertebral Column	7	310	210 / 100	0.677419355
Wholesale customers	8	440	142 / 298	0.322727273

Accuracy Scores

Dataset Name	RF	Linear SVM	DNN	LR	Bernoulli Naive Bayes	LDA	Ada Boost	DT
Bank note auth.	0.9923	0.9889	0.9998	0.9896	0.8419	0.9786	0.996	0.9810
Blood Transfusion	0.6279	0.5323	0.5003	0.5495	0.4993	0.5419	0.6181	0.5852
Climate Sim. Crashes	0.5501	0.7941	0.6581	0.5773	0.5	0.7158	0.7535	0.6527
Sonar, Mines/Rocks	0.8167	0.7692	0.61	0.7507	0.5051	0.7358	0.7954	0.6919
Default of credit card	0.6540	0.5217	0.5	0.4999	0.6731	0.6127	0.6388	0.6081
Fertility	0.5531	0.4960	0.5223	0.4988	0.5	0.4878	0.5375	0.4964
Voice Rehabilitation	0.7831	0.5058	0.6344	0.5529	0.6854	0.7256	0.7916	0.7351
Pima Indians Diabetes	0.7216	0.5597	0.6864	0.7151	0.5035	0.7253	0.7131	0.6412
Spambase	0.9323	0.8252	0.6706	0.9215	0.8736	0.8699	0.9345	0.9025
Vertebral Column	0.8058	0.7261	0.8101	0.8095	0.6438	0.8017	0.7917	0.7592
Wholesale customers	0.9053	0.6939	0.5	0.8765	0.5	0.7748	0.8800	0.8520

Other Performance Metrics for Spambase

Algorithm Name	Accuracy	f1 score	mcc	Precision	Recall	roc auc
Random Forest	0.940491943	0.922081025	0.875282549	0.952318939	0.893881254	0.932342067
Decision Tree	0.905932828	0.881324856	0.803675827	0.87639301	0.88665373	0.902562047
AdaBoost Classifier	0.938579417	0.921548713	0.871293486	0.927807641	0.915611007	0.934563288
Deep Neural Network	0.723321129	0.538341427	0.405704617	0.776242927	0.421798091	0.670599051
Bernouli Naive Bayes	0.885960569	0.849226568	0.759761205	0.886503696	0.815392448	0.873621528
Linear Support Vector	0.832480662	0.776085134	0.673030914	0.821482253	0.791323227	0.825286975
Logistic Regression	0.927605137	0.906666538	0.847999572	0.921425047	0.892614492	0.921487069
Linear Discriminant Analysis	0.887568983	0.846374519	0.764242867	0.916240821	0.78682252	0.869953218
Gaussian Naive Bayes	0.821346197	0.808553048	0.676682579	0.700090522	0.957035295	0.845072219

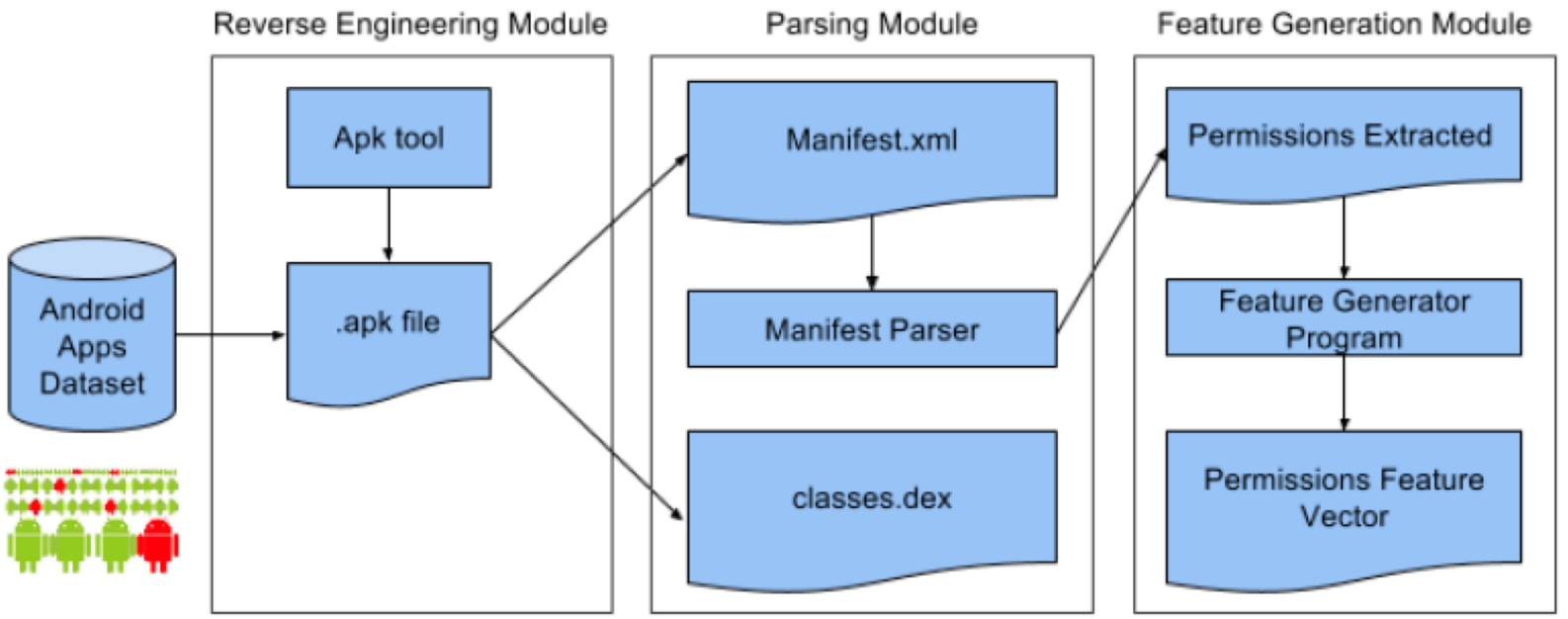
Android Malware Detection

- Objective: To find what kind of features are more effective for Malware detection

- Dataset:
  - Android applications.
  - Contains 40,000 benign and 3000 malware applications.

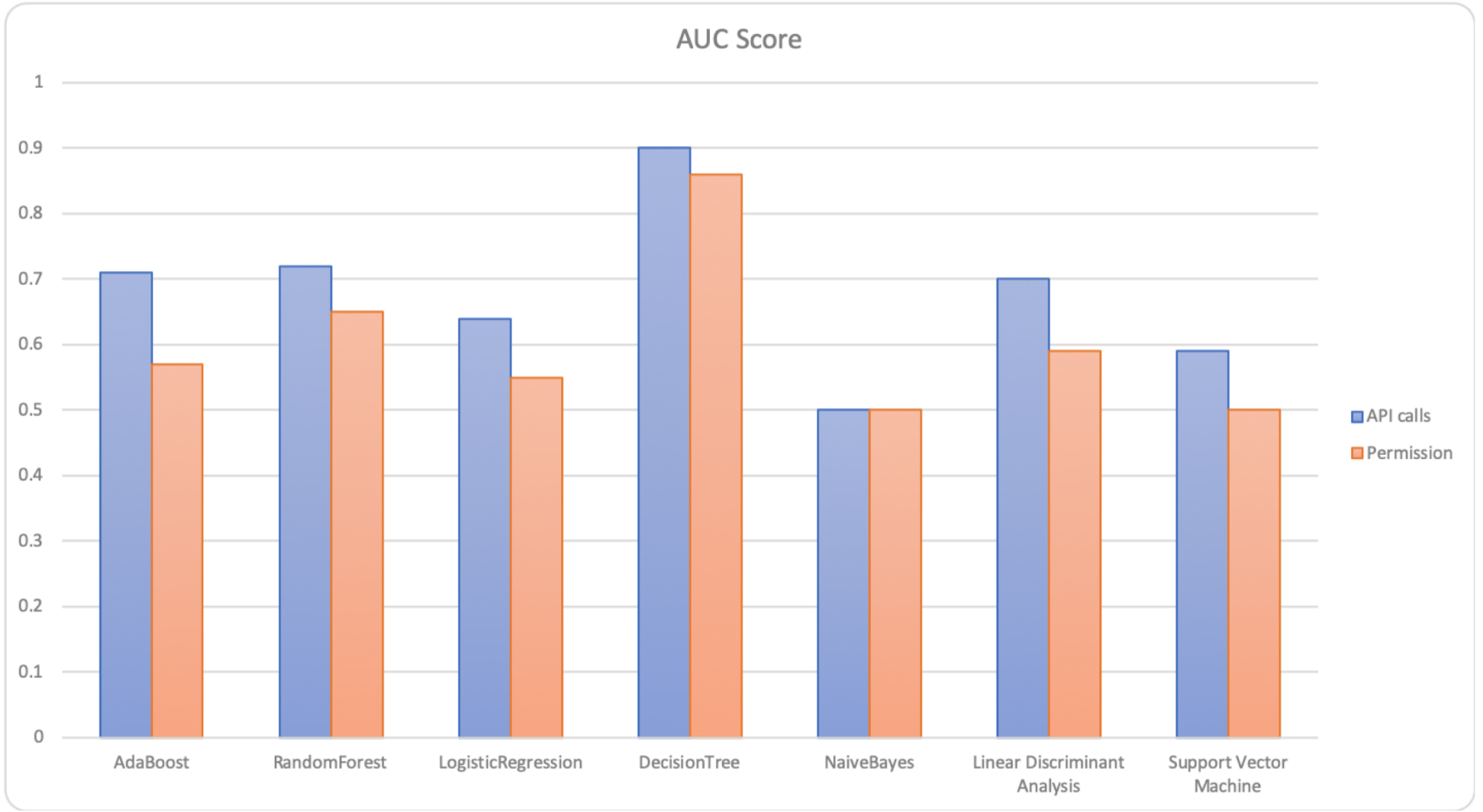
- Two sets of features are ideal for malware detection:
  - Permission requests
  - System calls

Permission requests Feature extraction



D. Arp, M. Spreitzenbarth, M. Huebner, H. Gascon, and K. Rieck. 2014. ***Drebin: Efficient and Explainable Detection of Android Malware in Your Pocket***. In 21st Annual Network and Distributed System Security Symposium (NDSS).

Performance



Multiclass Classification

Multiclass Classification

Dataset Name	No. of Attributes	No. of samples	Classes
Dermatology	35	358	6
Iris	5	150	3
Leaf	16	340	30
Page Blocks Classification	11	5473	5
Wine Quality	12	4898	11

Multiclass Classification Accuracy

Algorithm Name	Dermatology	Iris	Leaf	Page Blocks Classification	Wine Quality
Random Forest	0.97	0.929	0.745	0.852	0.412
Linear SVM	0.956	0.934	0.578	0.667	0.0801
DNN	0.613	0.571	0.585	0.119	0.050
Gaussian Naive Bayes	0.843	0.933	0.713	0.495	0.219
LR	0.97	0.93	0.385	0.732	0.210
LDA	0.956	0.964	0.7937	0.669	0.253
AdaBoost	0.48	0.905	0.0853	0.554	0.114
Decision Tree	0.97	0.929	0.745	0.852	0.412

Summary

- Fine-tuning of hyperparameters is essential.
- No Standard ML algorithms for all applications.
- Allow user to spend more time on Feature engineering.
- Framework design suitable for inhouse development.
- Easy to modify to cater needs of different users.

For Questions:

sohaib.kiani@ku.edu