# ANALYSIS AND DETECTION OF MALWARE IN ANDROID APPLICATIONS USING MACHINE LEARNING

[1]M.Manasa,[2] B.Nikitha, [3]K.Sravan kumar, [4]K.Vamshi, [5]Dr.Neelakandapillai Subash
[1,2,3,4] U.G.Scholor, Department of ECE, Sri Indu College Of Engineering & Technology, Ibrahimpatnam, Hyderabad.
[5]Professor, Department of ECE, Sri Indu College Of Engineering & Technology, Ibrahimpatnam, Hyderabad.

## ABSTRACT

Android platform due to open source characteristic and Google backing has the largest global market share. Being the world's most popular operating system, it has drawn the attention of cyber criminals operating particularly through wide distribution of malicious applications. This paper proposesan effectual machine-learning based approach forAndroidMalwareDetectionmakinguseof evolutionary Genetic algorithm for discriminatory feature selection. Selected features from Genetic algorithm are used to train machine learning classifiers and their capability in identification of Malware before and after feature selection is compared. The experimentation results validate that Genetic algorithm givesmostoptimizedfeaturesubset helping in reduction of feature dimension to less than half of the original feature-set. Classification accuracy of more than 94% is maintained post feature selection for the machine learning based classifiers, while working on much reduced feature dimension, thereby, having a positive impact on computational complexity of learning classifiers.

## INTRODUCTION

Android Apps are freely available on Google Playstore, the official Android app store as well as third-party app stores for users to download. Due to its open source nature and popularity, malware writers are increasingly focusingondevelopingmaliciousapplications for Android operating system. In spite of various attempts by Google Playstore to protect against malicious apps, they still find their way to mass market and cause harm to users by misusing personal informationrelated to their phone book, mail accounts, GPS location information and others for misuse by third parties or else take control of the phones remotely. Therefore, there is need to perform malware analysis or reverse- engineering of such malicious applications which pose serious threat to Android platforms. Broadly speaking, Android Malware analysis is of two types: Static Analysis and Dynamic Analysis. Static analysis basicallyinvolves analyzing the code structure without executing it while dynamic analysis is examination of the runtime behavior of Android Apps in constrained environment. Given in to the ever-increasing variantsofAndroidMalwareposingzero-day threats, an efficient mechanism for detection of Android malwares is required. In contrastto signature-based approach which requires regular update of signature database.

**Motivation:**

In this paper author is using two machine learning algorithms such as SVM (Support Vector Machine) and NN (Neural Networks). App will contains more than 100 features and machine learning will take more time to build model so we need to optimized (reducedataset columns size) features, to optimized features author is using genetic algorithm. Genetic algorithm will choose important features from dataset to train model and remove un-important features. Due to this process dataset size will be reduced and trainingmodelwillbegeneratedfaster. Inthis paper comparison we are losing some accuracy after applying genetic algorithm but we are able to reduce model trainingexecution time.

**Objective:**

Android is an open source free operating system and it has support from Google to publish android application on its Play Store. Anybody can developed an android app and publish on play store free of cost. Thisandroid feature attract cyber-criminals to developed and publish malware app on play store. If anybody install such malware app then it will steal information from phone and transfer to cyber-criminals or can give total phone control to criminal’s hand. To protect users from such app in this paper author is using machine learning algorithm to detect malwarefrommobileapp.Todetectmalware from appweneedto extract all codefrom app using reverse engineering and then check whetherappisdoinganymischievousactivity such as sending SMS or copying contact details without having proper permissions. If suchactivitygivenincodethenwewilldetect that app as malicious app. In a single appthere could be more than 100 permissions (examples of permissions are transact, API call signature, on Service Connected, APIcall signature, bind Service, API call signature, attach Interface, API call signature, Service Connection, API call signature, android. os. Binder, API call signature, SEND_SMS, Manifest Permission, Ljava. lang.Class. Get Canonical Name, API call signature etc.) which we need to extract from code and then generate a features dataset, if app has proper permission then we will put value 1 in the features data and if not then we will value 0. Based on those features dataset app will be mark as malware or good ware.

## LITERATURE SURVEY

### Android Malware Detection UsingMachine Learning on Image Patterns

Inthispaper,amalwareclassificationmodel hasbeenproposedfordetectingmalware samplesintheAndroidenvironment.The proposedmodelisbasedonconvertingsome files from the source of the Android applicationsintograyscaleimages.Some image-based local features and global features,includingfourdifferenttypesof localfeaturesandthreedifferenttypesof globalfeatures,havebeenextractedfromthe constructedgrayscaleimage datasetsandused fortrainingtheproposedmodel.Tothebest of ourknowledge,thistypeoffeaturesisused forthefirsttimeintheAndroidmalware detectiondomain.Moreover,thebagofvisual wordsalgorithmhasbeenusedtoconstruct onefeaturevectorfromthedescriptorsofthe localfeatureextractedfromeachimage.The extractedlocalandglobalfeatureshavebeen usedfortrainingmultiplemachinelearning classifiersincludingRandomforest,k-nearest neighbors,DecisionTree,Bagging,AdaBoost andGradientBoost.Theproposedmethod obtainedaveryhighclassificationaccuracy reached98.75%withatypicalcomputational timedoesnotexceed0.018sforeachsample. The results of the proposed model outperformedtheresultsofallcompared state-of-art models in term of both classification accuracy and computational time.

**Android mobile security by detecting and classification of malware based on permissions using machine learning algorithms**

Android occupies a major share in the mobile application market. Android mobiles have become an easy target for the attackers. The main reason is the user ignorance in the process of installing and usage of the apps. Android malware can be detected based onthe permissions it requests from the user. Several machine learning algorithms arebeing used in the detection of android malware based on the list of permissions enabled for each app. This paper makes an attempt to study the performance of some of the machine learning algorithms, viz., naïve Bayes, J48, Random Forest, Multi-class classifier and Multi-layer perceptron. Google playstore2015and2016appdataareused for normal apps and standard malware data sets are used in the evaluation. Multi-class classifier was found to be outperforming the other algorithms in terms of classification accuracy. Naïve Bayes classifier has outperformed as far as model construction time is concerned.

**An Android Behaviour Based Malware Detection Method using Machine Learning**

In this paper, we propose An Android Behavior-Based Malware Detection Method using Machine Learning. We improve an Android application sandbox, Droidbox, by inserting a view-identification automatic trigger program which can click mobile applications in the meaningful order. Taking advantage of Droidbox result, we collect the behavior such as network activities, file read/write and permission as the feature data and use different machine learning algorithms to classify malware and evaluate the performance. We use a large number of malware and normal application samples to prove that our method has high accuracy.

## SYSTEMANALYSIS

## EXISTINGSYSTEM

The main contribution of the work isreduction of feature dimension to less than half of original feature-set using Genetic Algorithm such that it can be fed as input to machine learning classifiers for training with reduced complexity while maintaining their accuracyinmalwareclassification.Incontrast to exhaustive method of feature selection which requires testing for 2N different combinations,whereNisthenumberof eatures, Genetic Algorithm, a heuristic searching approach based on fitness function has been used for feature selection. The optimized feature set obtained using Genetic algorithm is used to train two machine learning algorithms: Support Vector Machine and Neural Network. It is observed that a decent classification accuracy of more than 94% is maintained while working on a much lower feature dimension, thereby, reducingthe training time complexity of classifiers.

## PROPOSEDSYSTEM

- Two set of Android Apps or APKs: Malware/Good wareis reverse engineered toextractfeaturessuchaspermissionsand count of App Components such as Activity, Services, Content Providers, etc. These features are used as feature vector with class labels as Malware and Good ware represented by 0 and 1 respectively in CSV format.

- To reduce dimensionality of feature-set, the CSV is fed to Genetic Algorithm to select the most optimized set of features. The optimized set of features obtained is used for training two machine learning classifiers: Support Vector Machine and Neural Network.

- In the proposed methodology, static features are obtained from AndroidManifest.xml which contains all the important information needed by any Android platform about the Apps. Androguard tool has been used for disassemblingoftheAPKsandgettingthe static features.

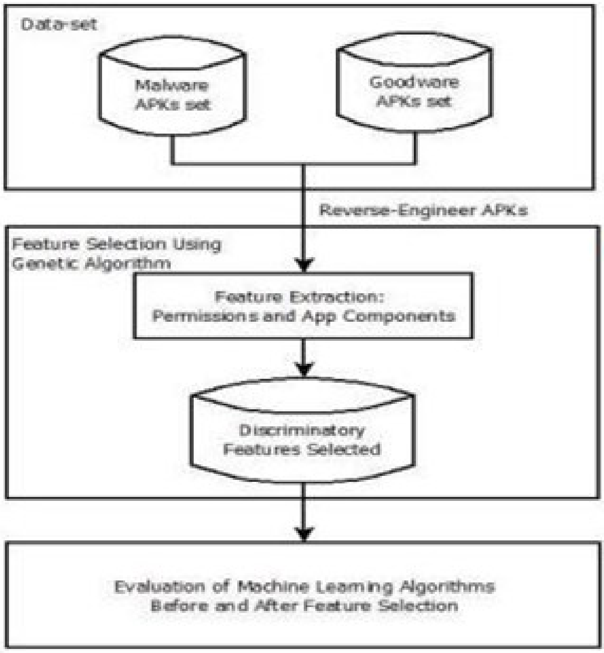

Data-set
Malware APKs set
Goodware APKs set
Reverse-Engineer APKs
Feature Selection Using Genetic Algorithm
Feature Extraction: Permissions and App Components
Discriminatory Features Selected
Evaluation of Machine Learning Algorithms Before and After Feature Selection

Fig. 1. Proposed Methodology

**Advantagesofproposedsystem:**

- Security
- Proposed a novel and efficient algorithm for feature selection to improve overall detection accuracy.
- Machine-learningbasedapproachin combination with static anddynamic analysis can be used to detect new variants of Android Malware posing zero-day threats.

## IMPLEMENTATION

## MODULES:

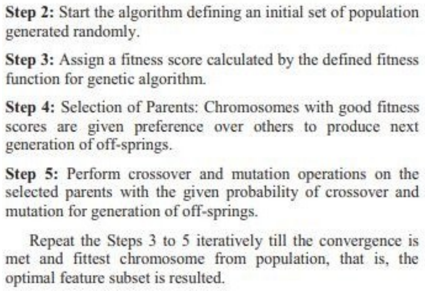
Featureselectionisanimportantpartin machine learning to reduce data dimensionality and extensive research carried outforareliablefeatureselectionmethod.For feature selection filter method and wrapper method have been used. In filter method, features are selected on the basis of their scores in various statistical tests that measure the relevance of features by their correlation withdependentvariableoroutcomevariable.

Wrapper method finds a subset of features by measuring the usefulness of a subset offeature with the dependent variable. Hence filtermethodsareindependentofanymachine learning algorithm whereas in wrappermethod the best feature subset selected depends on the machine learning algorithm used to train the model. In wrapper method a subset evaluator uses all possible subsets and then uses a classification algorithm to convince classifiers from the features in each subset. The classifier considers the subset of feature with which the classificationalgorithm performs the best. To find the subset, the evaluator uses different search techniques like depth first search, random search, breadth first search or hybrid search. The filter method uses an attribute evaluator along with a ranker to rank all the features in the dataset. Here one feature is omitted at a time that has lower ranks and then sees the predictive accuracy of the classification algorithm. Weights or rank put by the ranker algorithms are different than those by the classification algorithm. Wrapper method is useful for machine learning test whereas filter method is suitable for data mining testbecausedatamininghasthousandsofmillions of features.

- UploadAndroiddataset
- GenerateTrain&testmodel
- Pre-processing
- RunSVM&Neuralnetwork alg

**Algorithmsusedinthisproject:-**

The steps involved in feature selection using Genetic Algorithm can be summarized as below:



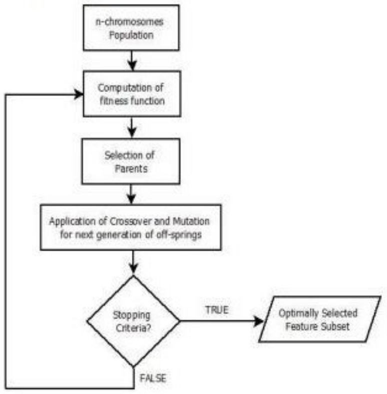
**Step 2:** Start the algorithm defining an initial set of population generated randomly.

**Step 3:** Assign a fitness score calculated by the defined fitness function for genetic algorithm.

**Step 4:** Selection of Parents: Chromosomes with good fitness scores are given preference over others to produce next generation of off-springs.

**Step 5:** Perform crossover and mutation operations on the selected parents with the given probability of crossover and mutation for generation of off-springs.

Repeat the Steps 3 to 5 iteratively till the convergence is met and fittest chromosome from population, that is, the optimal feature subset is resulted.


n-chromosomes Population
Computation of fitness function
Selection of Parents
Application of Crossover and Mutation for next generation of off-springs
Stopping Criteria?
TRUE
Optimally Selected Feature Subset
FALSE

## SYSTEMDESIGN

### SystemArchitecture:

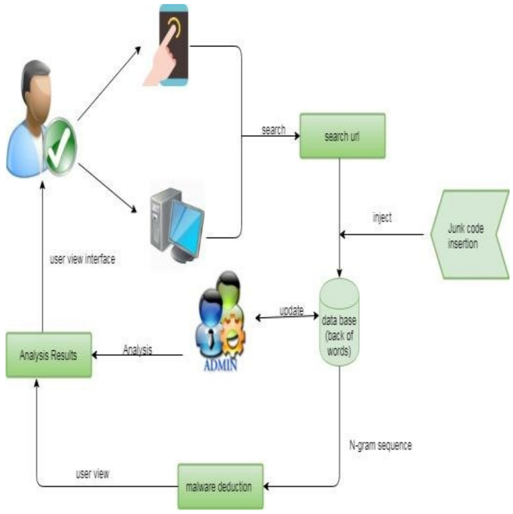
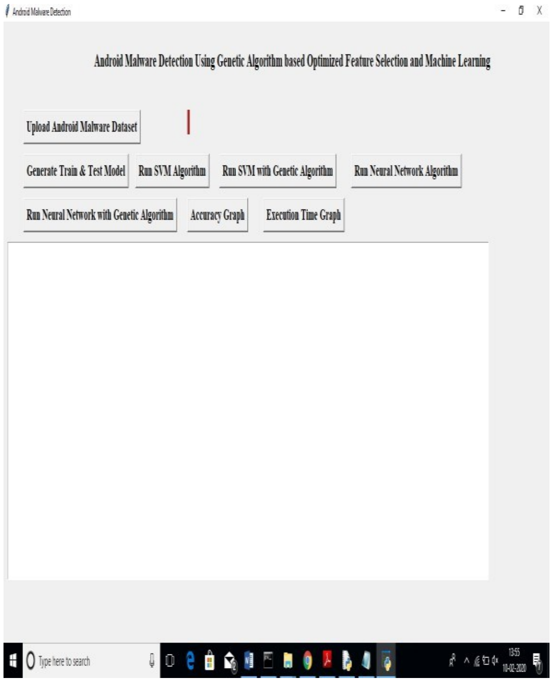
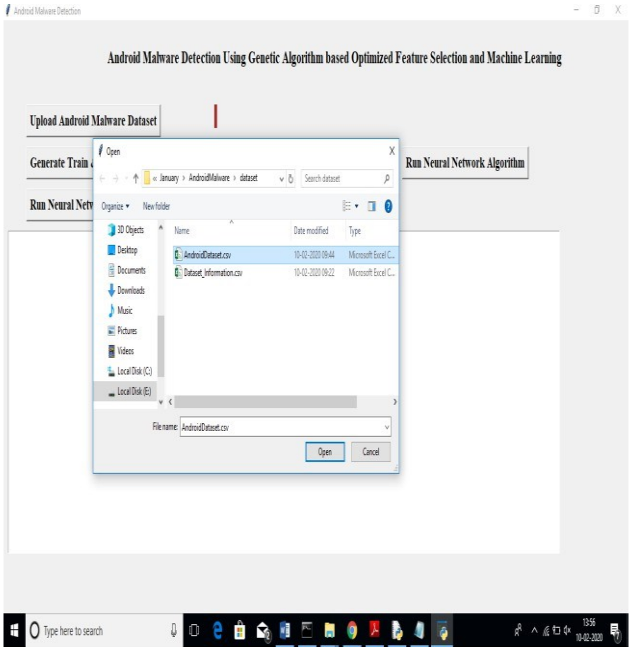


**Fig.System Architecture**

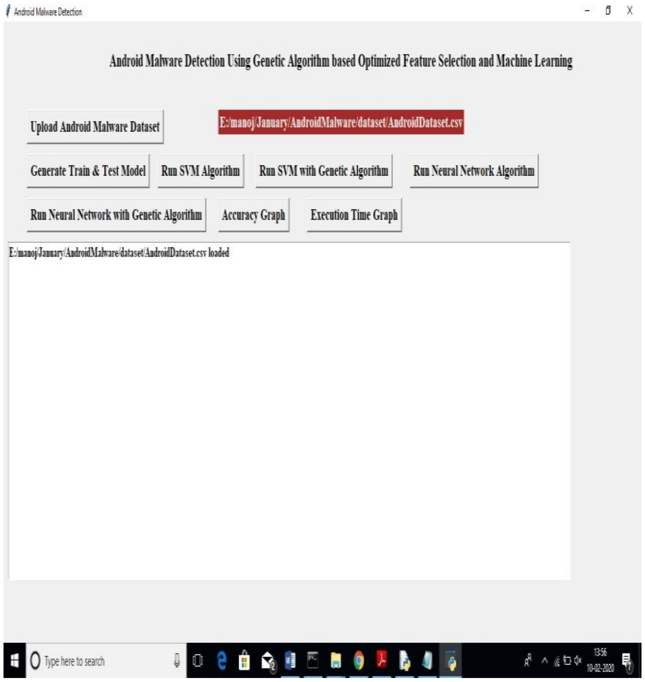
## Results


Android Malware Detection
Android Malware Detection Using Genetic Algorithm based Optimized Feature Selection and Machine Learning
Upload Android Malware Dataset
Generate Train & Test Model
Run SVM Algorithm
Run SVM with Genetic Algorithm
Run Neural Network Algorithm
Run Neural Network with Genetic Algorithm
Accuracy Graph
Execution Time Graph
Type here to search

Inabovescreenclickon‘UploadAndroid Malware Dataset’ button and upload dataset.

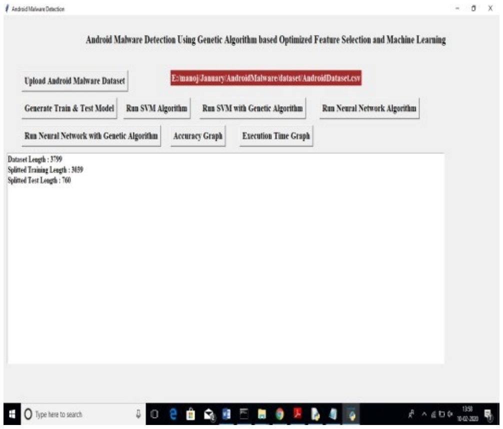


In above screen I am uploading ‘AndroidDataset.csv’ file and after uploadwill get below screen


Android Malware Detection
Android Malware Detection Using Genetic Algorithm based Optimized Feature Selection and Machine Learning
Upload Android Malware Dataset
E:/manoj/January/AndroidMalware/dataset/AndroidDataset.csv
Generate Train & Test Model
Run SVM Algorithm
Run SVM with Genetic Algorithm
Run Neural Network Algorithm
Run Neural Network with Genetic Algorithm
Accuracy Graph
Execution Time Graph
Type here to search

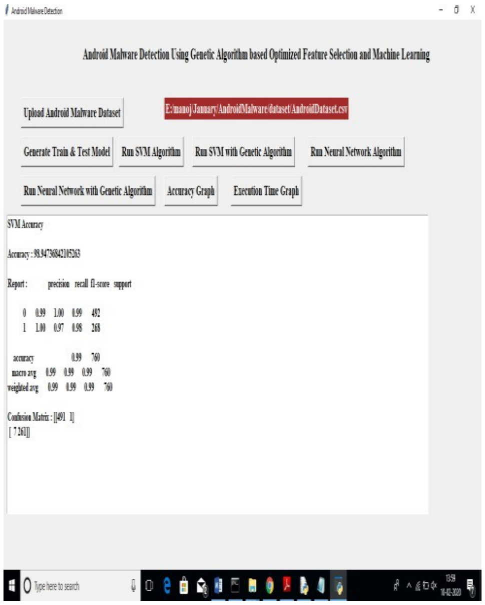
```
E:/manoj/January/AndroidMalware/dataset/AndroidDataset.csv loaded
```

Now click on ‘Generate Train & Test Model’ button to split dataset into train and test part. All machine learning algorithms will take80% dataset for training and 20% dataset to test accuracy of trained model. After clicking that button will get train and test model


Android Malware Detection
Android Malware Detection Using Genetic Algorithm based Optimized Feature Selection and Machine Learning
Upload Android Malware Dataset
E:/manoj/January/AndroidMalware/dataset/AndroidDataset.csv
Generate Train & Test Model
Run SVM Algorithm
Run SVM with Genetic Algorithm
Run Neural Network Algorithm
Run Neural Network with Genetic Algorithm
Accuracy Graph
Execution Time Graph
Type here to search

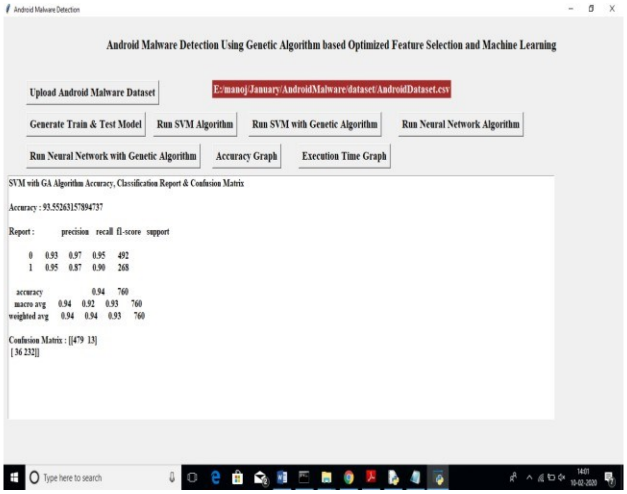
```
Dataset Length : 3799
Splitted Training Length : 3039
Splitted Test Length : 760
```

In above screen we can see there are total 3799androidapprecordsarethereand application using 3039 records for trainingand 760 records for testing. Now we haveboth train and test model and now click on 'Run SVM Algorithm' button to generate SVM model on train and test and get its accuracy


Android Malware Detection
Android Malware Detection Using Genetic Algorithm based Optimized Feature Selection and Machine Learning
Upload Android Malware Dataset
E:/manoj/January/AndroidMalware/dataset/AndroidDataset.csv
Generate Train & Test Model
Run SVM Algorithm
Run SVM with Genetic Algorithm
Run Neural Network Algorithm
Run Neural Network with Genetic Algorithm
Accuracy Graph
Execution Time Graph
Type here to search
13:59
10-02-2020

```
SVM Accuracy

Accuracy : 98.94736842105263

Report :        precision  recall  f1-score  support

      0     0.99    1.00    0.99    492
      1     1.00    0.97    0.98    268

  accuracy                  0.99    760
 macro avg     0.99    0.99    0.99    760
weighted avg   0.99    0.99    0.99    760

Confusion Matrix : [[491  1]
 [ 7 261]]
```

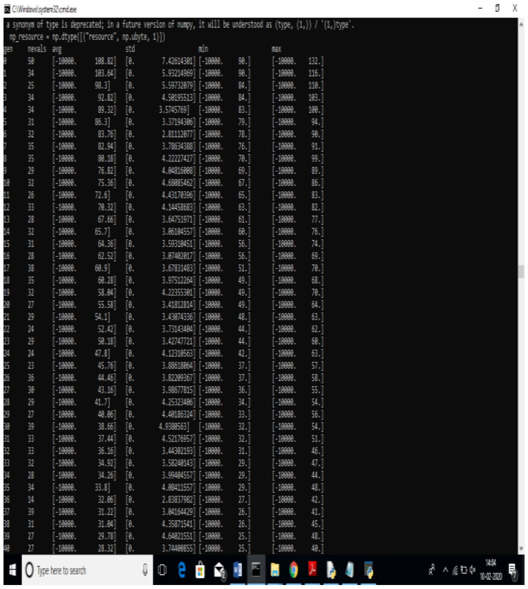
In above screen we got 98% accuracy for SVM and now click on 'Run SVM with GeneticAlgorithm'buttonto chooseoptimize features and then run SVM on optimize features to get accuracy


Android Malware Detection
Android Malware Detection Using Genetic Algorithm based Optimized Feature Selection and Machine Learning
Upload Android Malware Dataset
E:/manoj/January/AndroidMalware/dataset/AndroidDataset.csv
Generate Train & Test Model
Run SVM Algorithm
Run SVM with Genetic Algorithm
Run Neural Network Algorithm
Run Neural Network with Genetic Algorithm
Accuracy Graph
Execution Time Graph
Type here to search
14:01
10-02-2020

```
SVM with GA Algorithm Accuracy, Classification Report & Confusion Matrix

Accuracy : 93.55263157894737

Report :        precision   recall  f1-score   support

      0     0.93    0.97    0.95    492
      1     0.95    0.87    0.90    268

  accuracy                  0.94    760
 macro avg     0.94    0.92    0.93    760
weighted avg   0.94    0.94    0.93    760

Confusion Matrix : [[479  13]
 [36 232]]
```

In above screen SVM with Genetic algorithm got 93% accuracy. Genetic with SVM accuracy is less but its execution time will be less which we can see at the time of comparison graph.

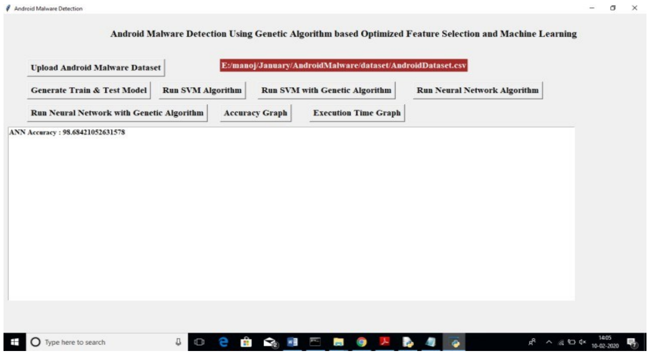
(Note: when u run genetic then 4 empty windows will open u just close all those 4 windows and let main window to run)



```
a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
gen     nevals  avg                     std                     min                     max
0       50      [-10000.    108.82]     [0.         7.42614301] [-10000.    90.]        [-10000.   132.]
1       34      [-10000.    103.64]     [0.         5.93214969] [-10000.    90.]        [-10000.   116.]
2       25      [-10000.    98.3]       [0.         5.59732079] [-10000.    84.]        [-10000.   110.]
3       34      [-10000.     92.82]     [0.         4.50195513] [-10000.    84.]        [-10000.   103.]
4       34      [-10000.     89.32]     [0.        3.5745769]   [-10000.    83.]        [-10000.   100.]
5       31      [-10000.    86.3]       [0.         3.37194306] [-10000.    79.]        [-10000.    94.]
6       32      [-10000.     83.76]     [0.         2.81112077] [-10000.    78.]        [-10000.    90.]
7       35      [-10000.     82.94]     [0.         3.78634388] [-10000.    76.]        [-10000.    91.]
8       35      [-10000.     80.18]     [0.         4.22227427] [-10000.    70.]        [-10000.    99.]
9       29      [-10000.     76.82]     [0.         4.04816008] [-10000.    69.]        [-10000.    89.]
10      32      [-10000.     75.36]     [0.         4.60085462] [-10000.    67.]        [-10000.    86.]
11      26      [-10000.    72.6]       [0.         4.43170396] [-10000.    65.]        [-10000.    83.]
12      33      [-10000.     70.32]     [0.         4.14458683] [-10000.    63.]        [-10000.    82.]
13      28      [-10000.     67.66]     [0.         3.64751971] [-10000.    61.]        [-10000.    77.]
14      32      [-10000.    65.7]       [0.         3.06104557] [-10000.    60.]        [-10000.    76.]
15      31      [-10000.     64.36]     [0.         3.59310451] [-10000.    56.]        [-10000.    74.]
16      28      [-10000.     62.52]     [0.         3.07402017] [-10000.    56.]        [-10000.    69.]
17      38      [-10000.    60.9]       [0.         3.67831483] [-10000.    51.]        [-10000.    70.]
18      35      [-10000.     60.28]     [0.         3.97512264] [-10000.    49.]        [-10000.    68.]
19      32      [-10000.     58.04]     [0.         4.22355301] [-10000.    49.]        [-10000.    70.]
20      27      [-10000.     55.58]     [0.         3.41812814] [-10000.    49.]        [-10000.    64.]
21      29      [-10000.    54.1]       [0.         3.43074336] [-10000.    48.]        [-10000.    63.]
22      24      [-10000.     52.42]     [0.         3.73143404] [-10000.    44.]        [-10000.    62.]
23      29      [-10000.     50.18]     [0.         3.42747721] [-10000.    44.]        [-10000.    60.]
24      24      [-10000.    47.8]       [0.         4.12310563] [-10000.    42.]        [-10000.    63.]
25      23      [-10000.     45.76]     [0.         3.88610064] [-10000.    37.]        [-10000.    57.]
26      36      [-10000.     44.46]     [0.         3.82209367] [-10000.    37.]        [-10000.    58.]
27      30      [-10000.     43.16]     [0.         3.98677815] [-10000.    36.]        [-10000.    55.]
28      29      [-10000.    41.7]       [0.         4.25323406] [-10000.    34.]        [-10000.    54.]
29      27      [-10000.     40.06]     [0.         4.40186324] [-10000.    33.]        [-10000.    56.]
30      39      [-10000.     38.66]     [0.        4.9300563]   [-10000.    32.]        [-10000.    54.]
31      33      [-10000.     37.44]     [0.         4.52176957] [-10000.    32.]        [-10000.    51.]
32      33      [-10000.     36.16]     [0.         3.44302193] [-10000.    31.]        [-10000.    46.]
33      32      [-10000.     34.92]     [0.         3.58240143] [-10000.    29.]        [-10000.    47.]
34      28      [-10000.     34.26]     [0.         3.99404557] [-10000.    29.]        [-10000.    44.]
35      34      [-10000.    33.8]       [0.         4.08411557] [-10000.    29.]        [-10000.    48.]
36      14      [-10000.     32.06]     [0.         2.83837982] [-10000.    27.]        [-10000.    42.]
37      39      [-10000.     31.22]     [0.         3.04164429] [-10000.    26.]        [-10000.    41.]
38      31      [-10000.     31.04]     [0.         4.35871541] [-10000.    26.]        [-10000.    45.]
39      27      [-10000.     29.78]     [0.         4.64021551] [-10000.    25.]        [-10000.    48.]
40      27      [-10000.     28.32]     [0.         3.74400855] [-10000.    25.]        [-10000.    40.]
```

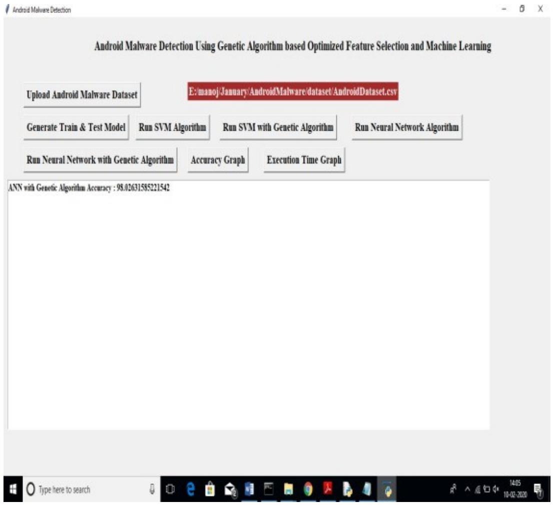
In above console we can see geneticalgorithm chooses 40features from all dataset features.

Now click on 'Run Neural Network Algorithm' button to test neural network accuracy.


Android Malware Detection
Android Malware Detection Using Genetic Algorithm based Optimized Feature Selection and Machine Learning
Upload Android Malware Dataset
E:/manoj/January/AndroidMalware/dataset/AndroidDataset.csv
Generate Train & Test Model
Run SVM Algorithm
Run SVM with Genetic Algorithm
Run Neural Network Algorithm
Run Neural Network with Genetic Algorithm
Accuracy Graph
Execution Time Graph
Type here to search
14:05
10-02-2020

```
ANN Accuracy : 98.68421052631578
```

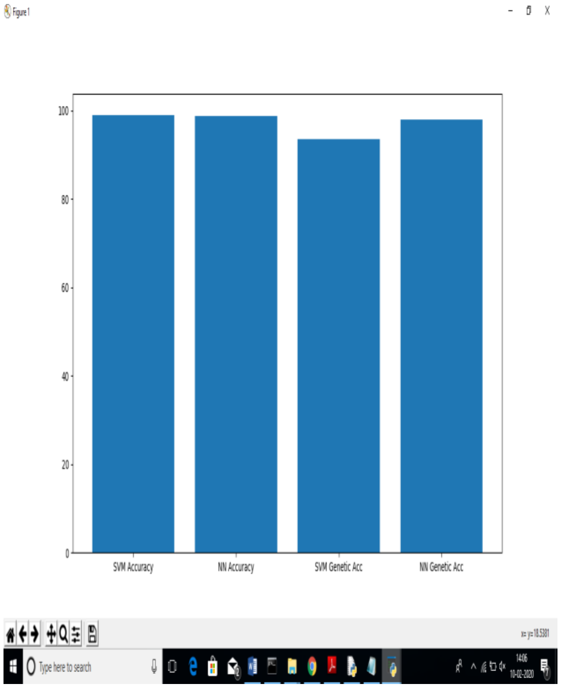
In above screen neural network also gave 98.64% accuracy. Now click on 'Run Neural Network with Genetic Algorithm' button to get NN accuracy with genetic algorithm


Android Malware Detection
Android Malware Detection Using Genetic Algorithm based Optimized Feature Selection and Machine Learning
Upload Android Malware Dataset
Generate Train & Test Model
Run SVM Algorithm
Run SVM with Genetic Algorithm
Run Neural Network Algorithm
Run Neural Network with Genetic Algorithm
Accuracy Graph
Execution Time Graph
Type here to search

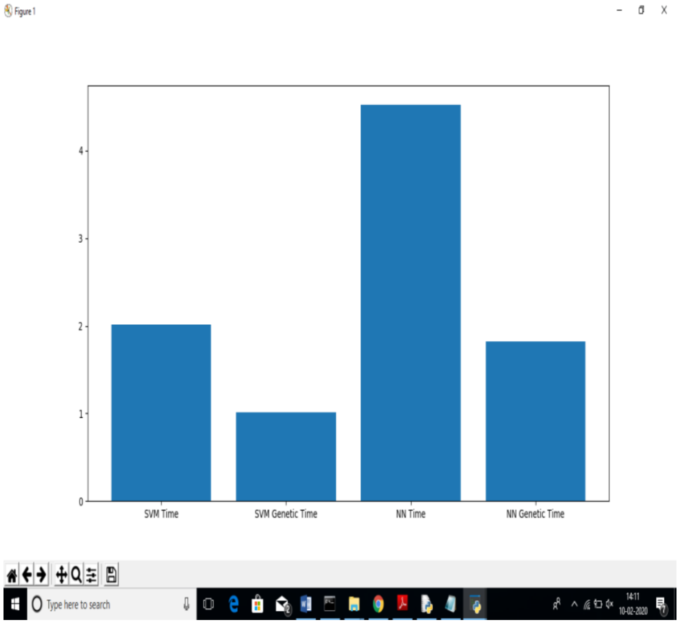
E:/manoj/January/AndroidMalware/dataset/AndroidDataset.csv

```
ANN with Genetic Algorithm Accuracy : 98.02631585221542
```

In above screen NN with genetic got 98.02% accuracy. Now click on 'Accuracy Graph' button to see all algorithms accuracy in graph



In above graph x-axis represents algorithm nameandy-axisrepresentsaccuracyandin all SVM got high accuracy. Now click on 'Execution Time Graph' button to get execution time of all algorithm



In above graph x-axis represents algorithm name and y-axis represents execution time. From above graph we can conclude that with genetic algorithm machine learningalgorithms taking less time to build model.

## CONCLUSION

As the number of threats posed to Android platforms is increasing day to day, spreading mainly through malicious applications or malwares, therefore it is very important to design a framework which can detect such malwares with accurate results. Where signature-based approach fails to detect new variants of malware posing zero-day threats, machine learning based approaches are being used. The proposed methodology attempts to make use of evolutionary Genetic Algorithm togetmostoptimizedfeaturesubsetwhich can be used to train machine learning algorithms in most efficient way.

## FutureEnhancements

From experimentations, it can be seen that a decent classification accuracy of more than 94% is maintained using Support Vector Machine and Neural Network classifiers while working on lowerdimensionfeature-set,therebyreducingthe training complexity of the classifiers Further work can be enhanced using larger datasets for improved results and analyzing the effect onother machine learning algorithms when used in conjunction with Genetic Algorithm.

## REFERENCES

[1] D. Arp, M. Spreitzenbarth, M. Hübner, H. Gascon, and K. Rieck, "Drebin: Effective and Explainable Detection of Android Malware in Your Pocket," in Proceedings 2014 Networkand Distributed System Security Symposium, 2014.

[2] N.Milosevic,A.Dehghantanha,andK.K. R. Choo, "Machine learning aided Android malware classification," Comput.Electr.Eng., vol. 61, pp. 266–274, 2017.

[3] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-An, and H. Ye, "Significant PermissionIdentification for Machine-Learning-Based Android Malware Detection," IEEE Trans. Ind. Informatics,vol.14,no.7,pp.3216–3225, 2018.

[4] A. Saracino, D. Sgandurra, G. Dini, and F. Martinelli, "MADAM: Effective and Efficient Behavior-basedAndroidMalwareDetectionand Prevention,"IEEETrans.DependableSecur. Comput., vol. 15, no. 1,pp. 83–97, 2018.

[5] S. Arshad, M. A. Shah, A. Wahid, A. Mehmood, H. Song, and H. Yu, "SAMADroid: A Novel 3-Level Hybrid Malware Detection Model for Android Operating System," IEEE Access, vol. 6, pp. 4321–4339, 2018.