Android Malware Detection Techniques

Software Security Project-1 Part-3

Snigdha Sri Nemani
Computer Science
Arizona State University
Tempe Arizona USA
snemani2@asu.edu

Saideepthi Korupolu
Computer Science
Arizona State University
Tempe Arizona USA
skorupo1@asu.edu

Venkat Ratnam Sabbavarapu
Computer Science
Arizona State University
Tempe Arizona USA
vsabbava@asu.edu

ABSTRACT

Android has recently overtaken iOS as the most popular smartphone operating system because of the popularity of free apps, which prompted hackers to create malware-infected apps that can steal sensitive data from these devices. Finding malware-infected apps and keeping them out of the Google Play store is the most pressing issue. The Android app's underlying permission model is where the vulnerability exists. As a result, it is now the responsibility of app developers to properly identify the rights that their apps will request during the installation and execution of the apps.

This study paper's experimental effort comprises the creation of a powerful malware detection system that aids in determining and examining the detective effects of a number of well-known and widely utilized sets of features for malware detection. Three different feature selection procedures are used to choose the best features from the data set namely Chi Square Test, Mutual Information Gain, and PCA (Principal Component Analysis). Further, we developed a malware detection model by utilizing ML classifiers like Logistic Regression, SVM (Support Vector Machine), and Decision Tree out of which the decision tree has been concluded as the best approach as the F-1 score is high. Apart from the three feature reduction techniques, the whole dataset is also used and it has given better results as the model is trained on all the attributes, thus it has more information while training.

KEYWORDS

Machine learning, Support Vector Machine, Malware detection, Principal Component Analysis.

1 INTRODUCTION

Today's smartphones are more than just cell phones; they can also connect with an operating system that functions somewhat like a computer and can carry out a variety of tasks thanks to apps. In the year 2000, Symbian became the first cutting-edge mobile operating system for smartphones. Following their lead, a small number of mobile phone manufacturers, including Nokia, Microsoft, Apple, and Google, released their own mobile operating systems on the market. Among them, the Android operating system, introduced by Google in 2008, is quite well-liked because it is open source, freely downloadable, and offers a large selection of free apps in its play store [1]. As mobile intelligent terminals quickly evolve, Android overtakes other smartphone operating systems in terms of usage. Android is a target for malware assaults due to its widespread distribution and open-source nature, which allows users to get applications from sources other than the official Android Market.

Every program on Android's privilege-separated operating system has a unique system identification, such as a Group-ID and Linux user-ID. The system may automatically grant permissions or may ask users to approve or reject permission requests depending on how sensitive the requested rights are. Cybercriminals want to invade users' privacy by utilizing these rights. According to Footnote 5, G-Data Security experts identified over 7,50,000 new malware apps by the end of 2019. They had detected 3,246,284 malware apps as of the end of the year 2018 and more than 7,50,000 as of the end of 2019.[1]

Android apps operate under the permission-model theory [2]. Additionally, it offers four levels of protection, classifying permissions as "signature," "signature or system," "normal," and "hazardous." Because they are system granted, "signature" and "signature or system" are not considered in our analysis. We only consider rights that the user has explicitly authorized as "normal" and "hazardous." Normal permissions pose no threat to the privacy of the user. If the permission is stated in its manifest, the system will automatically grant it. On the other side, risky permissions provide access to the user's private information. However, the decision to grant or deny access to a permission or group of permissions rests entirely with the user. Google's security check is simple to go over, allowing dangerous Android apps to enter the Google Play store and ultimately end up on consumers' smartphones. Cybercriminals regularly create malicious apps using these permissions, then invite victims to install them. The market has more than two billion active Android devices.

The effectiveness of malware detection depends on choosing the proper combination of features. The performance of malware detection is significantly impacted by the attributes that are chosen

as an input. In this work, we employ various feature selection algorithms to choose the right features.

2 BACKGROUND

This section discusses the types of malware and the current malware detection techniques as well as briefly introducing the malware itself.

2.1 Malware

Malware is software created with the purpose of interfering with, harming, or allowing unauthorized access to a computer system. Viruses, worms, trojan horses, spyware, botnets, ransomware, adware, rootkits, keyloggers, and backdoors are only a few examples of the many different types of malware [3]. These malware variants are not mutually exclusive; therefore they may simultaneously introduce the traits of several variants. In order to determine the damage that malware can cause to the targeted system and, if at all feasible, learn more precise information about the attacker, lker kara [4] examines the malware's potential. He suggested an approach that includes behavior analysis, memory analysis, and code analysis in order to evaluate malware utilizing digital data and an actual malware attack. It was discovered that malware may be tracked using the server's Who is information to which it is linked, allowing for research to be done based on the behavior of malware.

2.2 Malware Detection Techniques

The three primary kinds of malware detection techniques are signature-based, heuristic-based, and specification-based. These methods locate and recognize malware and take action against it to protect computer systems from potential data and resource loss.

2.1.1 Signature-based malware detection

When malware is created, a string of bits commonly referred to as a signature is inserted into the code, which can later be used to determine which malware family it is a member of. Most antivirus products employ the signature-based detection method. The antivirus tool analyzes the infected file's code and looks for patterns that are exclusive to a particular malware family [5]. Malware signatures are kept in a database and used for comparison later on in the detection process. String or pattern scanning or matching are other names for this type of detection method. It may also be static, dynamic, or hybrid.

2.1.2 Heuristic-Based malware detection

Heuristic-based detection picks up on or distinguishes between a system's normal and anomalous behavior in order to finally find and stop known and unidentified malware threats. Two steps make up the heuristic-based detection method. The system's behavior is first observed when an attack is not present, and crucial data is recorded so it may be checked and validated in the event of an attack. In the second stage, this distinction is scrutinized to find malware belonging to a specific family.

The three main parts of a behavior detector employed in a heuristic-based technique are as follows.

Data collection: As its name suggests, this part is concerned with gathering either static or dynamic data.

Data interpretation: This step transforms the data into an intermediate format after being interpreted by the data gathering component.

Algorithm for matching: In the interpretation component, the converted information is compared to the behavior signature using this component.

Although heuristic-based detection is an effective technique, there are drawbacks to the technique, including a higher level of false positives and a higher resource requirement. This approach is often referred to as a proactive strategy, behavior analysis, or anomaly identification. Before using heuristic-based detection, many types of analysis, including file-based, weight-based, rule-based, and generic signature analysis, are conducted.

2.1.3 Specification-Based malware detection

Applications are monitored in accordance with their specifications in specification-based detection techniques, which look for both normal and anomalous behavior. This technique is derived from heuristic-based techniques, but there is a key distinction between the two: while heuristic-based detection techniques used machine learning and AI methods to detect legitimate and fraudulent activity of a legitimate program, specification-based detection techniques are based on an analysis of the behavior that is specified in the system specification [6]. This technique involves manually comparing some system's typical operations. By reducing the level of false positives and raising the level of false negatives, it gets around the drawback of heuristic-based approaches.

3 APPROACH AND STUDY METHODOLOGY

3.1 Data Collection and Analysis

In our project, the permissions are generated by using a regex operation which matches the strings and the number of permissions have been generated by using the count. In the similar manner, the number of intent filters are generated using the same regex pattern. Also, the app name and the Boolean value for the app presence are also generated. All this data has been saved into a CSV file which is used as a dataset in this project.

After the dataset analysis, we have analyzed that there are 500 benign attacks and 500 malware attacks which will make the dataset a balanced one. In the dataset, there are 194 columns in

which 1 attribute is the app name, 188 attributes are the permissions of the apps, 1 attribute is number of permissions of the app, 1 attribute is number of intent filters of the app, 1 attribute is the file size, 1 attribute is whether the app has manifest file or not, and 1 attribute is whether the app has malware or benign attacks.

3.2 Feature Selection Techniques

3.2.1 Chi Square Test

The significance of a statistic's relation to the class is used in our study to forecast the ranking of characteristics. This test is used to analyze the self-determination between two events [7]. Higher computed values imply the rejection of outliers, making these aspects more relevant for identifying malware in Android apps.

Take into consideration a random variable Y that has the following N-degrees-of-freedom chi-squared distribution:

$$Y = \sum_{i=1}^{N} X_i^2 \qquad (1)$$

where the independent random variables X1, X2,..., XN are distributed using the conventional normal distribution.

Measurement errors are typically assumed to have a normal distribution with a zero mean and known variance when power system state estimation problems are formulated. A function f(x) can be defined as given in (2) under the same premise, where f(x) has a chi-squared distribution with a maximum of (m- n) degrees of freedom (m being the number of measurements and n being the number of the states). Because a solution requires at least n observations, it should be noted that in a power system with m measurements and n system states, at most (m- n) errors can be linearly independent. The degrees of freedom will therefore be at most (m-n).

$$f(x) = \sum_{i=1}^{m} R_{ii}^{-1} e_i^2 = \sum_{i=1}^{m} \left(\frac{e_i}{\sqrt{R_{ii}}} \right)^2 = \sum_{i=1}^{m} \left(e_i^N \right)^2 \qquad (2)$$

In (2), R is the diagonal error covariance matrix, and ei is the measurement error with normal distribution. Rii is the variance of the ith measurement error. The normalized error, abbreviated eN, has a typical normal distribution.

$$P\{X \geq x_t\} = \int_{x_t}^{\infty} \chi^2(u) du \qquad (3)$$

The likelihood that X will be greater than xt is represented by equation (3). As xt rises, this probability falls as the distribution's tail decays. According to Fig. 1, the dotted line for the selected probability of 0.05 indicates that xt is 25.

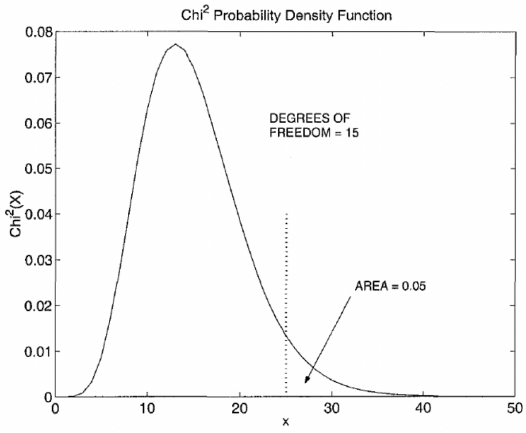

Figure 1: Chi-Squared Probability Density Function [8]

The maximum value that won't be labeled as a flawed measurement is represented by xt. The existence of a flawed measurement will be suspected if the measured value exceeds the threshold.

Most commercial state estimators that use the WLS estimate approach employ the following metric to identify faulty data:

$$J(\hat{x}) = \sum_{i=1}^{m} \frac{(z_i - h_i(\hat{x}))^2}{\sigma_i^2} = \sum_{i=1}^{m} \frac{(r_i)^2}{\sigma_i^2} \qquad (4)$$

where m is the total number of measurements, x is the (nx1) estimated state vector, hi (x), zi, and ri are the estimated, measured, and residual values, respectively, for the ith measurement, and 2 is the corresponding measurement I variance, which is the same as Rii. If the computed metric J (x) is more than 2 and the bad data suspicion (mn),p threshold value according to a chi-squared distribution for a given probability p and degrees of freedom, the conventional chi-squares test will suspect the existence of poor data (m-n).

3.2.2 Mutual Information Gain

The reduction in entropy or surprise that results from changing a dataset in any way is calculated as information gain. By assessing the information gain for each variable and choosing the one that maximizes information gain, which reduces entropy and best divides the dataset into groups for effective classification, it is frequently employed in the creation of decision trees from a training dataset. By assessing each variable's gain in relation to the target variable, information gain can be utilized for feature selection. The calculation is referred to as mutual information between the two random variables.

We can develop a metric representing more information about Y provided by X that represents the amount by which the entropy of

Y lowers given the entropy as a criterion of impurity in a training set S. IG is the name of this metric. It is provided by

$$IG=H(Y)-H(Y/X)=H(X)-H(X/Y) \qquad (5)$$

In terms of symmetry, IG is a measure. The knowledge acquired about Y following observation of X is equivalent to the knowledge acquired about X following observation of Y. The IG criterion has the drawback of favoring characteristics with higher values even when those features are not more informative.

3.2.3 PCA (Principal Component Analysis)

PCA is used to reduce the number of attributes in our data collection once it has been collected. A high-dimensional data space can be converted into a low-dimensional data space with the aid of PCA. The ability to detect malware relies heavily on features that are present in low dimension. Since there is high correlation among many features, PCA is used to reposition the features that are not highly connected. Principal component domain features are the term given to the obtained features. Furthermore, a low main component value is all that is required to detect meaningful patterns in the data.

The feature data set is compiled as a m n matrix, which includes m extracted features out of n data samples. The feature data set is normalized using equation:

$$\mu_j = \frac{1}{n}\sum_{i=1}^{n} x_i^j$$

Next, we use the Matlab environment to compute the eigen value and eigen vector. Next, we used the following procedures to choose the first k principal components from the covariance matrix.

$$\sum_{i=1}^{m} \lambda_{ii}$$

While cumvar stands for (cumulative variance) and indicates eigen values arranged in descending order, 99% of the variance is kept if (cumvar >=0.99)or (1 – cumvar <= 0.01) return k. Reduced feature sets are chosen for training after this is evaluated.

3.3 ML Modeling Techniques

3.3.1 Logistic Regression

Univariate Logistic Regression (ULR) analysis is being taken into consideration for feature ranking in order to confirm the level of significance for each feature set [9]. In the current study, we take into account two LR model benchmarks: determining the significance of each feature and ranking individual feature sets. The following variables are used in logistic regression analysis:

1. Regression coefficient value: The coefficient measure of features reveals the strength of each feature set's link with malware.

2. P-value: P-value, or level of significance, reveals the significance of an association.

Classification is the primary use of logistic regression. The fact that its data points are not grouped in line rows is the main distinction between it and linear regression. Each pile represents a category and each type of data piece has a corresponding category name. It might be a bunch here. For logistic regression, the classification boundary line, which is represented by the regression formula, should be found, as illustrated in Figure 2. To determine the optimal regression coefficient in the regression formula, the training classifier employs an optimization technique. [10] [11].

Given any set of inputs, the output of a function that uses logistic regression-based classification is the categorization of the input data. For instance, while classifying, the function output 0 or 1 in the two categories symbolizes two classes to facilitate processing. The range of the aforementioned function argument is from positive infinity to negative infinity in accordance with the real demands and the analysis above. Range of the dependent variable is 0 or 1. There are numerous functions that meet the aforementioned requirements. The 0-1 step function is the most understandable. The step point, however, does not allow for steering of the step function, which is not advantageous for processing mathematically. The Sigmoid function is therefore now often utilized, and its image is displayed in Figure 3.

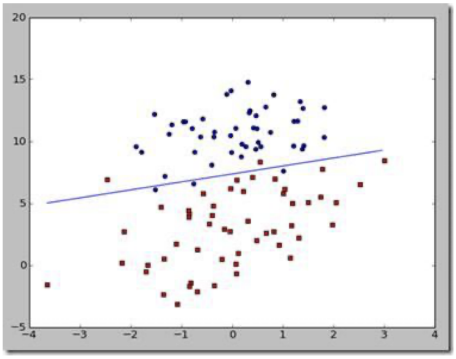

Figure 2: Logistic Regression

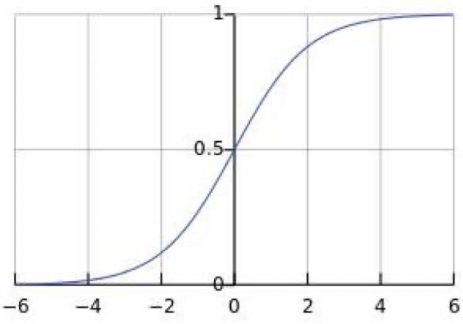

Figure 3: Sigmoid Function

The figure demonstrates that (0)=0.5. Z>0 causes the function value to approach 1 and transition into class 1 as z rises.

The function value approaches 0 and enters the 0 class when z 0, fulfilling the criteria for the aforementioned classification function. Following are some examples of how logistic regression classifies data: It is presumptively possible to describe the characteristics of the input data as (x0, x1, x2,..., xn), and each feature is multiplied by a regression coefficient (w0, w1,..., wn), after which the input is added as the sigmoid function:

$$z = w_0x_0 + w_1x_1 + w_2x_2 + \cdots + w_nx_n$$

Which is:

$$z = w^Tx$$

In the above equation, w is the row vector, x is the column vector. The output is a number between 0 and 1, with data categorized as class 1 if the output is larger than 0.5 and class 0 if the output is less than 0.5. The best regression coefficient (w0, w1, w2,..., wn) for this classifier is now identified.

3.3.2 SVM (Support Vector Machine)

SVM, or Support Vector Machine, is a method for classifying data. Training and test datasets are the primary ingredients in the data classification process. Each dataset piece includes a number of characteristics and classification attributes. The basic idea behind SVM is to build a model for classification prediction based on the properties of the current test dataset element. Although the SVM algorithm can use a variety of kernels, the linear, polynomial, RBF, and sigmoid are often the most popular.

The transformation matrix W is essential for feature extraction, and if SVM is being used as a classifier, the penalty parameter and kernel parameter need to be tuned. The combination of the feature extraction algorithm and optimized SVM parameters is taken into consideration because there is mutual influence and restriction between obtaining the transformation matrix and optimizing the SVM parameters. This allows for the realization of adaptive feature extraction and the generation of SVM with high generalization ability.

The feature extraction issue can be categorized as a combined optimization issue including the transformation matrix W, the SVM penalty parameter C, and the kernel parameter (using the Radial basis function). The accuracy of SVM classification is the ideal goal function. The following optimization problem represents the feature extraction algorithm.

$$\max_{W,C,\gamma}(A_{vc}(W,C,\gamma))$$

A_{vc} represents the classification accuracy of SVM.

Algorithm Steps:

1) The necessary parameters of SVM are defined, together with the size and optimization range of the transformation matrix W, the optimization range of the penalty parameter C, and the kernel parameter of the SVM.
2) We run the algorithm to find the best optimal solution.
3) The total number of support vectors is determined once the SVM has been trained on the entire set of training samples. The ultimate solution is determined by the solution with the lowest number.
4) Test samples are used to evaluate feature extraction and SVM parameter optimization.

When there is a small amount of data, SVM has good accuracy, but it requires a lot of training data samples. Also, this algorithm can only recognize straight line characters, but not the characters that are tilted to the left or right side. Due to this, SVM will take more time for the calculations for the curvelet transforms.

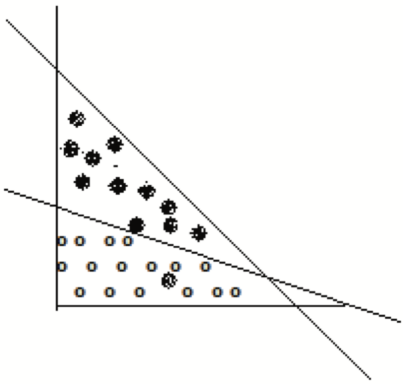

Figure 4: SVM Line Graph for states

3.3.3 Decision Tree Analysis

Decision trees (DT) are strong and well-liked tools for prediction and categorization. The fact that DT signifies rules is what makes it so alluring. The decision tree (DT) classifier has nodes that are either leaves or decision nodes in the form of a tree structure. The latter provides a test to be performed on a single attribute-value, with one branch and sub-tree for each conceivable test outcome, while the former refers to a class of instances. Starting at the tree's root and progressing through it until you reach a leaf node—which offers the classification of the instance—is how a DT can be used to categorize an example.

The method used to choose a test as the tree's root has a significant impact on the decision tree's structure. Quinlan's information theory is the standard for choosing the tree's root (information gain)

Building a decision tree is essentially a divide and conquer operation. An array T of training data has k classes (c1, c2,.....ck). T will be a leaf if T solely consists of cases from a single class. T is a leaf and its associated class will be assigned to the major class of its parent node if T does not contain any cases. In the event that T contains instances of mixed classes, a test based on an attribute ai of the training data will be run and divided into n subsets (T1, T2,..., Tn), where n is the total number of results of the test over attribute ai. Until every subset belongs to a single class, the same

decision tree construction method is applied recursively to each Ti, where j is a positive integer between 1 and n.

The decision tree algorithm for the feature selection works as follows:

Step-1: T<- SSV decision tree built for X, Y
Step-2: G(N)<- classification error reduction of Non-leaf node N.
Step-3: F<- set of all features of the input space.
Step-4: i<-0
Step-5: while F = null do:
(a) For each feature where f belongs to F not used by T define its rank R(f)<-i. Remove these features from F.
(b) Delete all the final splits of nodes N for which G(N) is minimal
(c) Delete all the final splits of nodes N for which G(N)=0.
(d) i <-i+1
Step-6: Return the list of features.

SSV stands for Separability of Split Value

As the decision tree building algorithm chooses the splits locally, i.e. in relation to the splits chosen in earlier stages, the decision tree's features complement one another. In other instances, only a tiny portion of the features are used by the complete categorization decision trees. The greatest number of characteristics that can be chosen is the number employed by the tree, as the method provides no information regarding the ranking of the remaining features. This sort of rigid ranking criteria also damages certain beneficial concealed aspects.

4 RESULTS AND ANALYSIS

In this section, we will analyze the results of 3 feature selection techniques with the 3 proposed ML model classifiers and compare the results. At the end, we will conclude which feature selection technique and which ML model is the best based on the results obtained.

4.1 Analysis of Chi Square Test with ML models

Following are the features that are obtained as per of chi square test feature selection technique:

1)BILLING
2) SEND_SMS
3) READ_LOGS
4) number_of_permissions
5) number_of_intent_filters
6) file_size

Now, with these features, we will try three different models and check the predictions whether the app is benign or malware.

4.1.1 Logistic Regression Results

The precision, recall, f1, and support values are as follows:

	precision	recall	f1-score	support
0	0.69	0.93	0.79	94
1	0.91	0.63	0.74	106
accuracy			0.77	200
macro avg	0.80	0.78	0.77	200
weighted avg	0.80	0.77	0.77	200

Figure 5: Result of Logistic Regression model with chi square test technique

4.1.2 SVM Results

	precision	recall	f1-score	support
0	0.76	0.73	0.75	94
1	0.77	0.79	0.78	106
accuracy			0.77	200
macro avg	0.76	0.76	0.76	200
weighted avg	0.76	0.77	0.76	200

Figure 6: Result of SVM model with chi square test technique

4.1.3 Decision Tree Results

	precision	recall	f1-score	support
0	0.83	0.81	0.82	94
1	0.83	0.85	0.84	106
accuracy			0.83	200
macro avg	0.83	0.83	0.83	200
weighted avg	0.83	0.83	0.83	200

Figure 7: Result of Decision Tree model with chi square test technique

Analysis:

From the above results, we can analyze that the F-1 score of the decision tree is 84% which is more than the other two models. Thus, we can conclude that out of logistic regression, SVM, and Decision Tree, Decision Tree is the best model and works better with chi square feature reduction technique.

4.2 Analysis of Mutual Information Gain Technique with ML models

Following are the features that are obtained as per of Mutual Information Gain feature selection technique:

1)BILLING
2) SEND_SMS
3) READ_PHONE_STATE
4) number_of_permissions
5) number_of_intent_filters
6) file_size

Now, with these features, we will try three different models and check the predictions whether the app is benign or malware.

4.2.1 Logistic Regression Results

The following are the results of the Mutual Information Gain selection technique when the modeling is performed with Logistic Regression.

	precision	recall	f1-score	support
0	0.70	0.93	0.80	94
1	0.91	0.65	0.76	106
accuracy			0.78	200
macro avg	0.80	0.79	0.78	200
weighted avg	0.81	0.78	0.78	200

Figure 8: Result of Logistic Regression model with Mutual Information Gain technique

4.2.2 SVM Results

The following are the results of the Mutual Information Gain selection technique when the modeling is performed with Support Vector Machine.

	precision	recall	f1-score	support
0	0.76	0.73	0.75	94
1	0.77	0.79	0.78	106
accuracy			0.77	200
macro avg	0.76	0.76	0.76	200
weighted avg	0.76	0.77	0.76	200

Figure 9: Result of SVM model with Mutual Information Gain technique

4.1.3 Decision Tree Results

The following are the results of the Mutual Information Gain selection technique when the modeling is performed with Decision Tree.

	precision	recall	f1-score	support
0	0.86	0.89	0.88	94
1	0.90	0.87	0.88	106
accuracy			0.88	200
macro avg	0.88	0.88	0.88	200
weighted avg	0.88	0.88	0.88	200

Figure 10: Result of Decision Tree model with Mutual Information Gain technique

Analysis:

From the above results, we can conclude that since the F-1 score of Decision Tree is higher than the other two models, decision tree is considered as the best model in the comparison and thus Mutual Information Gain technique works better when the modeling is performed with Decision Tree.

4.3 Analysis of PCA with ML models

With the PCA approach, the dataset is reduced into 3 features which can be attributed to the dimensionality reduction performed by the PCA.

Now, with these 3 features, we will try three different models and check the predictions whether the app is benign or malware.

4.3.1 Logistic Regression Results

The following are the results of the PCA feature selection technique when the modeling is performed with Logistic Regression.

	precision	recall	f1-score	support
0	0.90	0.85	0.87	104
1	0.84	0.90	0.87	96
accuracy			0.87	200
macro avg	0.87	0.87	0.87	200
weighted avg	0.87	0.87	0.87	200

Figure 11: Result of Logistic Regression model with PCA

4.3.2 SVM Results

The following are the results of the PCA feature selection technique when the modeling is performed with Support Vector Machine.

	precision	recall	f1-score	support
0	0.90	0.86	0.88	104
1	0.85	0.90	0.87	96
accuracy			0.88	200
macro avg	0.88	0.88	0.87	200
weighted avg	0.88	0.88	0.88	200

Figure 12: Result of SVM model with PCA

4.3.3 Decision Tree Results

The following are the results of the PCA feature selection technique when the modeling is performed with Decision Tree.

	precision	recall	f1-score	support
0	0.91	0.83	0.86	104
1	0.83	0.91	0.87	96
accuracy			0.86	200
macro avg	0.87	0.87	0.86	200
weighted avg	0.87	0.86	0.86	200

Figure 13: Result of Decision Tree model with PCA

Analysis:
From the above results, the F-1 scores of Logistic Regression, SVM, and Decision Tree are 86.8%, 87.3%, and 86.5% respectively. PCA gave better results on all the three models.Thus, we can conclude that PCA feature selection method gives better results than other feature selection techniques like Chi Square Test and Mutual Information Gain.

4.4 Analysis of the Whole dataset with ML models

Now, the whole dataset without any feature selection technique is being modeled with Logistic Regression, SVM, and Decision Tree.

4.4.1 Logistic Regression Results

The following are the results when the whole dataset is being modeled with logistic regression.

	precision	recall	f1-score	support
0	0.79	0.85	0.82	104
1	0.82	0.76	0.79	96
accuracy			0.81	200
macro avg	0.81	0.80	0.80	200
weighted avg	0.81	0.81	0.80	200

Figure 14: Result of Logistic Regression model with the whole dataset

4.4.2 SVM Results

The following are the results when the whole dataset is being modeled with SVM.

	precision	recall	f1-score	support
0	0.83	0.74	0.78	104
1	0.75	0.83	0.79	96
accuracy			0.79	200
macro avg	0.79	0.79	0.78	200
weighted avg	0.79	0.79	0.78	200

Figure 15: Result of SVM model with the whole dataset

4.4.3 Decision Tree Results

The following are the results when the whole dataset is being modeled with a decision tree.

	precision	recall	f1-score	support
0	0.95	0.88	0.92	104
1	0.88	0.95	0.91	96
accuracy			0.92	200
macro avg	0.92	0.92	0.91	200
weighted avg	0.92	0.92	0.92	200

Figure 16: Result of Decision Tree model with the whole dataset

Analysis:
The F-1 scores of Logistic Regression, SVM, and Decision Tree when modeled on the whole dataset are 86.7%, 87.3%, and 91.4% respectively. Thus, since the F-1 score of the Decision tree is higher than the other two models, we can conclude that when the model is trained on all the features it gives better results as the model has more information while training .

4.5 Final Result:

Feature Selection Technique	Features Extracted	Logistic Regression F1-score	SVM F1-score	Decision Tree F1-score
Chi-Square Test	READ_LOGS,SMS,BILLING Number_of_permissions Number_of_intent_filters file_size	74.4	78.1	84.1
Mutual Information Gain	READ_LOGS,SMS,READ_ PHONE_STATE Number_of_permissions Number_of_intent_filters file_size	75.8	78.2	84.4
PCA	N_components = 3	86.8	87.3	86.5
Whole Data	All permissions Manifest file Number_of_permissions Number_of_intent_filters file_size	86.7	88.8	91.4

Table 1: Result of 3 Feature Selection Techniques with three ML models

The comparison of the best model is performed on the basis ofF-1 score and the best feature selection technique is chosen based on the features selected and reduced. From Table 1, we can analyze that PCA is the best feature selection technique as it performed the dimensionality reduction and reduced the dataset to 3 important features. Out of the 3 ML classifiers, Decision Tree is the best model as the F-1 score of decision tree is higher than the other two models. Also, another analysis regarding the whole dataset is that it gave better results too as the model is trained on all attributes, thus it has more information while training.

4.6 TP, TN, FP, and FN

True Positive (TP): A true positive is an outcome where the model correctly predicts the positive class.

True Negative (TN): A true negative is an outcome where the model correctly predicts the negative class.

False Positive (FP): A false positive is an outcome where the model incorrectly predicts the positive class.

False Negative (FN): A false negative is an outcome where the model incorrectly predicts the negative class.

Based on the above 4 values, we can calculate the F-1, Precision, and Recall values as follows:

F-1 score= 2 TP/(2TP+FP+FN)
Precision = TP/(TP+FP)
Recall = TP/(TP+FN)

5 RELATED WORK

Arvind Mahindru, and A.L. Sangal focused on creating a malware detection framework utilizing a chosen collection of features that allows to determine if an Android app belongs to the malicious class or the benign class. Thirty different kinds of Android apps were used to aid in the execution procedure. According to their suggested detection methodology, a model created using LSSVM and an RBF kernel can identify 98.75% of unknown malware in real-world programs. When compared to commercial anti-virus software, their proposed framework is able to detect 98.8% of malware-infected Android apps. Additionally, it achieved a 3% higher detection rate when compared to other frameworks or techniques.[1]

6 CONCLUSION

In this project, we have used three feature selection techniques (Chi Square Test, Mutual Information Gain, and PCA). These 3 techniques are modeled with three ML classifiers namely Logistic Regression, Support Vector Machine, and PCA. Also, even the whole dataset is modeled with the three ML classifiers. Out of the feature selection techniques, PCA reduced the dataset and found three important features, while the other techniques found 6 features, thus PCA has been declared as the best feature selection technique. After the comparison of the F-1 scores, Decision Tree has the highest F-1 score, thus decision tree has been declared as the best ML classifier than Logistic Regression and SVM based on the F-1 scores obtained. For future work, We could increase the dataset's features and test how different machine learning classifiers perform. Additionally, we might think about comparing machine learning techniques with various APKs' system calls or Android intent.

REFERENCES

[1] Mahindru, A., Sangal, A. FSDroid:- A feature selection technique to detect malware from Android using Machine Learning Techniques. *Multimed Tools Appl* **80**, 13271–13323 (2021). https://doi.org/10.1007/s11042-020-10367-w

[2] Birendra C (2016) Android permission model. arXiv:160704256

[3] Qamar, A., Karim, A., Chang, V.: Mobile malware attacks: review, taxonomy & future directions. Fut. Gener. Comput. Syst. **97**, 887–909 (2019)

[4] Arp D, Spreitzenbarth M, Hubner M, Gascon H, Rieck K, Siemens C (2014) Drebin: Effective and explainable detection of android malware in your pocket. In: Ndss, vol 14, pp 23–26

[5] Landage, Jyoti, and M. P. Wankhade. "Malware and malware detection techniques: A survey." International Journal of Engineering Research and Technology (IJERT) 2.12 (2013): 2278-0181.

[6]. Robiah, Y., et al. "A new generic taxonomy on hybrid malware detection technique." arXiv preprint arXiv: 0909.4860 (2009).

[7] Plackett RL (1983) Karl pearson and the chi-squared test. In: International statistical review/Revue International de Statistique, pp 59–72

[8] A. Abur and A. Gomez-Exposito, "Power System State Estimation: Theory and Implementation", book, Marcel Dekker, 2004.

[9] Cruz AEC, Ochimizu K (2009) Towards logistic regression models for predicting fault-prone code across software projects. In: 2009 3rd International Symposium on Empirical Software Engineering and Measurement, IEEE, pp 460–463

[10] Z. Qian, Meng. Deyu, Xu. Zongben, "L_(1/2) Regularized Logistic Regression," Pattern Recognition and Artificial Intelligence, vol. 25(05), pp. 721-728, 2012.

[11] Bowes, X. Yu, Statistical methods for classification data analysis, CA: Social Sciences Academic Press. 2009