Feature Selection and Scaling for Random Forest Powered Malware Detection System

Ashutosh Tripathi
Birla Institute of Applied Sciences
Naman Bhoj (namanbhoj99@gmail.com)
Birla Institute of Applied Sciences
Mayank Khari
Birla Institute of Applied Sciences
Bishwajeet Pandey
Birla Institute of Applied Sciences

Research Article

Keywords: malware detection, machine learning, random forest, feature selection, feature scaling

Posted Date: August 4th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-778333/v1

Abstract

With the rise of internet usage malwares pose a great threat to user security and privacy. Therefore, to mitigate the problem it is essential to develop an efficient malware detection framework. In our research we experimented with various machine learning and feature scaling algorithms. Chi-Square was used as the feature selection technique which selected a set of 48 features from a feature space of 128 features.

The empirical results provide us with conclusive evidence that Random Forest is the best algorithm for detection of malware achieving an accuracy of 91.300% on our dataset followed by Gradient Boosting and Support Vector Machines.

Introduction

Recent survey reported that there are 7.8 billion people in the world, out of which 57.17% of the population i.e 4.66 billions are actively using the internet on a daily basis and the number is only growing with 87,5000 new users being introduced to the internet everyday [1]. This overwhelming increase is due to the advancement in network technology, businesses and government offering their services on the internet for faster and easier reachability to the audience. As everything from shopping to managing finances, learning and healthcare can be easily accessed on various websites. This consequently has also increased the number of threats posed to users security and data privacy on these platforms due to a number of online attacks such as malware, phishing and DDos. Therefore, it has become really essential for businesses, government to mitigate this problem by enhancing their security system to provide a safe and secure transaction based platform to users.

A report by Purplesec [2] suggests that there are 230,000 new malwares generated everyday and there were 812.67 million malware attacks in 2018, which is a 65.86 fold increase in malware attacks compared to 12.4 million attacks in 2012. 90% of the financial institutions have reported to suffer from malware attacks in 2018 where around 18 millions websites were infected by malwares in any given week [2].

To mitigate the problem of malware attacks in this paper we aim to develop a safe and secure framework for identification of malware attacks. The high-level architecture of the malware detection system is shown in Figure 1.

The user sends a request to the website and the response data from the website is fed to a malware detection system which analyses the data and notifies the user if there is a potential malware breach. If there is no such attack identified then the request is processed and data is downloaded or shown to the user otherwise a warning is generated and the site is blocked.

The main contribution of this paper is as follow:

- Investigate various machine learning techniques and use feature selection to identify high value features for malware detection.
- Investigate and compare the performance of feature scaling technique on the feature selected by feature selection technique.
- Derive a final framework for malware detection based on empirical results of the experiments.

The remainder of the paper is organized as follows: Section 2 comprises the Literature Review. Section 3 describes the Research Methodology. Section 4 Analyses the Result and Section 5 Concludes the paper.

Literature Review

Developing malware detection systems has become a challenging task as cybercriminals are developing innovative techniques to bypass the current detectors. Various approaches have been employed for detection of malware in the past which are discussed in this section.

Hossein Sayadi [3] used J48 and JRip for malware detection in embedded devices. The minimum hardware overhead was found in using J48 and JRip. Their research reported an accuracy of 81% using this technique.

Pedro Silva [4] used icons as a new feature in their dataset. Their research focused on detection of PE malware files. They reported the highest accuracy of 84.4%.

Khulood Al Messabi [5] research used a set of 8 unique features for malware detection. Their proposed work reported the highest accuracy of 77.52%.

Rima Masri [6] in their research aimed to detect malicious advertisements on the internet which may contain javascript code that redirects the user to malicious websites or installs the malware on the system. They deployed three different malware detection systems and attained the highest accuracy of 73%.

Bojan Kolosnjaji [7] in their research performs hierarchical feature extraction which combines n-grams and convolution with sequence modelling. Using this approach they achieve the highest accuracy of 85.6%.

Pertaining to past research conducted in this area, we explain the undertaken approach in our paper in the next section

Research Methodology

Dataset

We used the malicious domain dataset for conducting research in this paper [8]. It had 12 independent features, where 1 feature was text based, 3 were multi-category and the rest of the features were

numerical. The final goal using these independent features was to predict the categorical dependent feature (malware/not malware). The dataset consisted of 10000 instances of these features and was divided into 80-20 ratio where 80% of data was used for training the models and 20% of data was used for testing purposes. This train-test distribution ensures us a class balanced data, where the number of examples of both the classes were equal [9].

Models

Support Vector Classifier (SVC) - Support Vector Classifier [10] is a supervised machine learning algorithm. SVC is used for both classification and regression problems. SVC aims to predict the best decision boundary between two class labels in N-dimensional space, Where N is the number of features. All the data points in the space are placed according to given features, to classify them into given class SVC draws separating planes to make a boundary line of each class. The best possible separating plane is known as Hyperplane. The data points which lie at the nearest distance from the hyperplane are called support vectors and Planes passing through support vectors and parallel to the hyperplane are known as marginal planes and the hyperplane is equidistant from both marginal planes, and the distance between these is called marginal distance.

Many decision boundaries can be drawn for the same data to classify into the same classes but the most accurate separation can be obtained by the hyperplanes having larger marginal distances.

The equation for the hyperplane is:

$$wT\Phi(z)+b=0$$

Where b is a constant.Now let z0 be a data point vector in the space. Distance of z0 from hyperplane can be calculated as:

$$d(\Phi(z0))= | wT\Phi(z0)+b|/||w||$$

Where b is constant and ||w|| is the euclidean norm of w.

Since support vectors are the nearest data point vectors from the hyperplane, therefore the distance of the support vector can be obtained by minimizing the distance.

$$dmin= minm(| wT\Phi(zm)+b|/||w||)$$

Dmin is the distance from the hyperplane to the marginal plane i.e. marginal distance. For the optimal classification, marginal distance should be maximum.

$$w*= max(minm(| wT\Phi(zm)+b|/||w||))$$

w=max(dmin)*

Mathematically we can understand maximizing marginal distance helps to create accurate separating boundaries.

Random Forest (RF) - Random forest [11] is an ensemble bagging technique. It is used for both regression and classification tasks. Random forest uses decision trees as base learners. In a random forest multiple decision trees perform classification individually on random input data samples. Inputs to the decision trees may be repeated but repetition of inputs may limit the performance of the model to achieve better accuracy inputs should be selected randomly. Each decision tree predicts the result independently. To get final results, random forests find the majority of the predictions given by the multiple decision trees. If we use a single decision tree for the final prediction, and a small part of the dataset is changed then the decision tree may change its prediction value. In the random forest if a small part of the dataset is altered then it doesn't leave any major impact on the final prediction. The accuracy of the model is also dependent on the number of decision trees; a large number of decision trees leads to better accuracy. Random forest is used when features are weak. Random forests easily process data and perform better with noisy data and if a part of data is missing.

For optimal classification of a dataset on a node of a decision tree Gini impurity and entropy are calculated for splitting of data into given classes. Gini impurity is calculated to get the cost of splitting, which helps in optimal classification at a node with the given features. Gini impurity is the probability of misclassification and its lower value leads to better splitting. Entropy is also calculated to ensure optimal splitting of sample data on a node. Its value lies between 0 and 1. Entropy determines the boundary of classification.

Gradient Boosting Classifier (GBC) - Gradient boosting is a supervised machine learning algorithm[12] that uses ensemble bagging techniques to perform classification and regression tasks. Gradient boosting prediction is done by combining multiple decision trees in an additive manner. Multiple decision trees are added in a series with changed target values for each tree. In gradient boosting weak learners make predictions. Weak learners are used in the direction of gradient descent to reduce prediction error in a minimum amount of time. Gradient boosting combines the results of multiple weak learners to achieve better accuracy. Dependent and independent features are used as input to the model. Independent features remain unchanged for all models while dependent feature changes for each model. For better prediction accuracy, the decision tree should have more height.

In the first step, the base model predicts the result for every row in the data sample, and then After getting the predicted value, residuals/ error is calculated-

Error = (x-x')

where x' is the predicted value and x is an actual value. In the next step, models are trained in an additive approach by targeting error as an input feature. to reduce error, a loss function is generated. For

regression problems generally, the least square function is considered for loss calculation.

Loss function=$(x-x')^2$

To get optimal classification we have to optimize the loss function.

Data Analysis

In this subsection we discuss the data analysis steps undertaken while conducting the research. The research workflow is shown in Figure 2.

After loading our dataset, we first checked our dataset for null values and found our dataset containing no null values. Then we started to extract top level domain (com) and domain (netflix) from our existing "domain name (www.netflix.com)" feature. These two extracted features were treated as categorical features and after extraction "domain name" feature was dropped from our dataset. In the next step we removed "." from the "IP" feature and treated them as numerical features.

As a final step we extracted top 20 occurrences from top level domain, domain, country code and owner and added them to the dataset as new categorical features. After this step the initial "top level domain" , "domain", "country code" and "owner" features were dropped. These preprocessing steps made our dataset containing 128 distinct features and also added some noise to our data. The features in our dataset were also from varying scales and hence making it important to use feature selection and feature scaling techniques to help our machine learning algorithm converge faster and make accurate predictions.

We used the Support Vector Machine, Random Forest and Gradient Boosting algorithm for creating our models.

Then we used two different feature scaling techniques with these models. The techniques are listed below:

- **Min-Max Scaler:** Rescales the feature between a custom range. We scaled our features in the range [0,1]. Scikit-Learn Library was used for implementing Min-Max Scaler [13].
- **Robust Scaler:** The shortcoming of Min-Max scaling is that it is sensitive to outliers. Therefore, to mitigate the possibility of this happening we used Robust Scaler. It scales data using Interquartile range and is less sensitive to the outliers [14].

For selecting the best subset of features from our dataset, we used the Chi-Square filter-based feature selection technique [15] which selected 48 features in total. Using 3 different machine learning

algorithms, 2 different scaling techniques and 1 feature selection technique made it possible to make a total of 6 models for the problem.

The comprehensive results are discussed in the next section i.e. Result and Analysis. The related data analysis code can be found in our github repository.

Results And Analysis

In this section we comprehensively discuss the results of the experiments performed in this paper.

For evaluating the performance of the models, we use Accuracy and F1-Score as the evaluation metrics:

- Accuracy: For a binary classification problem it is defined as the ratio of sum of True Negative and Positive to total number of instances.
- F1-Score: It can simply be defined as the harmonic mean of Precision and Recall.

Table 1: Accuracy and F1-Score for MinMax-Chi Square

Accuracy and F1-Score using MinMax-Chi Square			
Algorithm	*Accuracy*	*F1-Score*	*Class Instances*
SVC	88.050%	88.044	1000
RF	**91.300%**	**91.297**	1000
GBC	90.550	90.548	1000

It is evident from Table that for the class balanced data classification, the Random Forest model outperformed the other two models. It bested the Support Vector Classifier model with an absolute accuracy and f1-score difference of 3.25 and 3.253, whereas the difference for Gradient Boosting Classifier model was 0.75 and 0.7517. This is due to the fact that with increase in the number of trees in Random Forest it is robust to noise and less prone to overfitting.

TABLE . Accuracy and F1-Score for Robust-Chi Square

Accuracy and F1-Score using Robust-Chi Square			
Algorithm	*Accuracy*	*F1-Score*	*Class Instances*
SVC	89.100%	89.097	1000
RF	**91.200%**	**91.197**	1000
GBC	90.950%	89.949	1000

From Table it can be seen that, again, the Random Forest model outperforms the other two models by a considerable margin for balanced class classification, where Support Vector Classifier was bested by an absolute margin of 2.1 for both accuracy and f1-score. Gradient Boosting Classifier was bested the least with a margin of 0.25 and 1.248 in terms of accuracy and f1-score respectively.

Though it is evident from Table and Table that the Random Forest model outperformed all the other models. To analyze the effectiveness of feature scaling techniques results are aggregated in Figure 3, Figure 4 and Table respectively.

From Figure 3 and Figure 4, it can be clearly seen that the Random Forest classifier achieves the highest accuracy and f1-score. We can also see that Min-Max feature scaling combined with Chi-Square feature selection provides the highest accuracy and precision value for Random Forest and Gradient Boosting Classifier. Therefore, conclusive inferences can be drawn that Min-Max feature scaling is better suited for feature selection tasks for our problem.

From Table conclusive inferences can be drawn that the best model i.e Random Forest attains the highest accuracy and f1-score using the Min-Max Scaling. The absolute difference of .100 for both accuracy and f1-score.

TABLE . Comparison of Scaling Technique

Scaling Technique	Highest Accuracy	Highest F1-Score	Difference in Accuracy	Difference in F1-Score
Robust (RF)	91.200%	91.197	.100	.100
Min-Max (RF)	**91.300%**	**91.297**		

TABLE . Comparison with existing research

Existing Work	Accuracy Comparison	Proposed Work
H. Sayadi [3]	81%	**91.30%**
P. Silva [4]	84.4%	**91.30%**
K. Al Messabi,[5]	71.52%	**91.30%**
Rima Masri [6]	73%	**91.30%**
Bojan [7]	85.6%	**91.30%**

From Table it is evident that our research significantly contributed to the domain of knowledge and outperformed the majority of existing work reported in our paper.

Conclusion

The research aims to present an effective malware detection framework by experimenting with various machine learning algorithms, two different feature scaling techniques and feature selection. The results of the experiment presented conclusive evidence that the Random Forest technique is best suited for classification of malware detection attaining the highest accuracy and f1-score of 91.300% and 91.297. It was found that Min-Max scaling was more effective as a feature scaling technique for our problem which reduced the feature space from 128 features to 48 features, therefore providing us with a computationally inexpensive model for fast computation.

The performance and accuracy of the model can further be improved in future by using Evolutionary Algorithms for feature selection and training the model using Deep Learning models.

References

[1] "Global digital overview – global digital insights", DataReportal – Global Digital Insights, 2021.

[2] “2019 cyber SECURITY Statistics trends & data,” 24-Mar-2021.

[3] H. Sayadi, H. M. Makrani,O. Randive,S. Manoj P.D., S. Rafatirad and H. Homayoun, “Customized machine learning-based hardware-assisted malware detection in embedded devices,”, 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), pp. 1685-1688, Sept 2008

[4] P. Silva, S. A. Masouleh , and L. Li, “Improving malware detection accuracy by extracting icon information.”,IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), pp. 408-411, June 2018.

[5] K. Al Messabi, M. Aldwairi, A. Al Youshif, A.Thoban and F. Belqasmi, “Malware detection using dns records and domain name features.”, Proceedings of the 2nd International Conference on Future Networks and Distributed Systems., pp. 1-7, June 2018.

[6] R. Masri, and M. Aldwairi. "Automated malicious advertisement detection using virustotal, urlvoid, and trendmicro." 2017 8th International Conference on Information and Communication Systems (ICICS). IEEE, 2017.

[7] B. Kolosnjaji, A. Zarras, G. Webster, and C. Eckert "Deep learning for classification of malware system call sequences." Australasian Joint Conference on Artificial Intelligence. Springer, Cham, 2016.

[8] T. A. Y. Al, “Malicious Domain Name , 12 Features Datasets 10000,” Kaggle, 29-Sep-2020.

[9] N. BHOJ, et al. “Comparative Analysis of Feature Selection Techniques for Malicious Website Detection in SMOTE Balanced Data.” RS Open Journal on Innovative Communication Technologies, vol. 2, issue 3, pp. 1-10, 2021, doi:10.46470/03d8ffbd.993cf635.

[10] V. Vapnik, “The nature of statistical learning theory.” Springer science & business media, 2013.

[11] L. Breiman, "Random forests," Machine learning 45.1 pp. 5-32, 2001.

[12] J. H. Friedman,"Greedy function approximation: a gradient boosting machine," Annals of statistics (2001), pp. 1189-1232, 2001.

[13] “sklearn.preprocessing.MinMaxScaler” scikit.

[14] “sklearn.preprocessing.RobustScaler¶,” scikit.

[15] N. Rachburee, and W. Punlumjeak, "A comparison of feature selection approach between greedy, IG-ratio, Chi-square, and mRMR in educational mining." 2015 7th International Conference on Information Technology and Electrical Engineering (ICITEE). IEEE, 2015.

Competing Interests

The authors declare no competing interests.

Figures

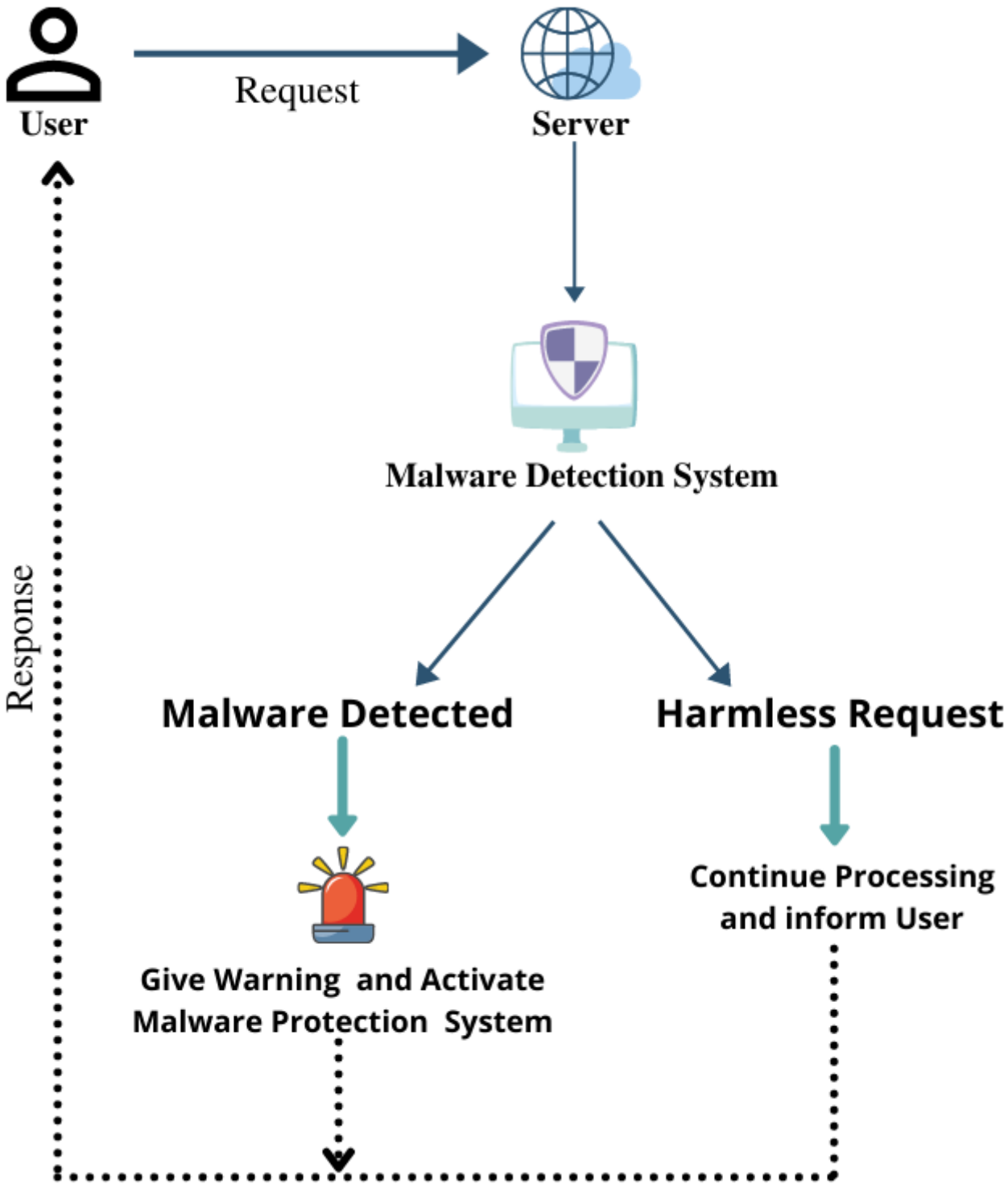

Figure 1

Malware Detection System

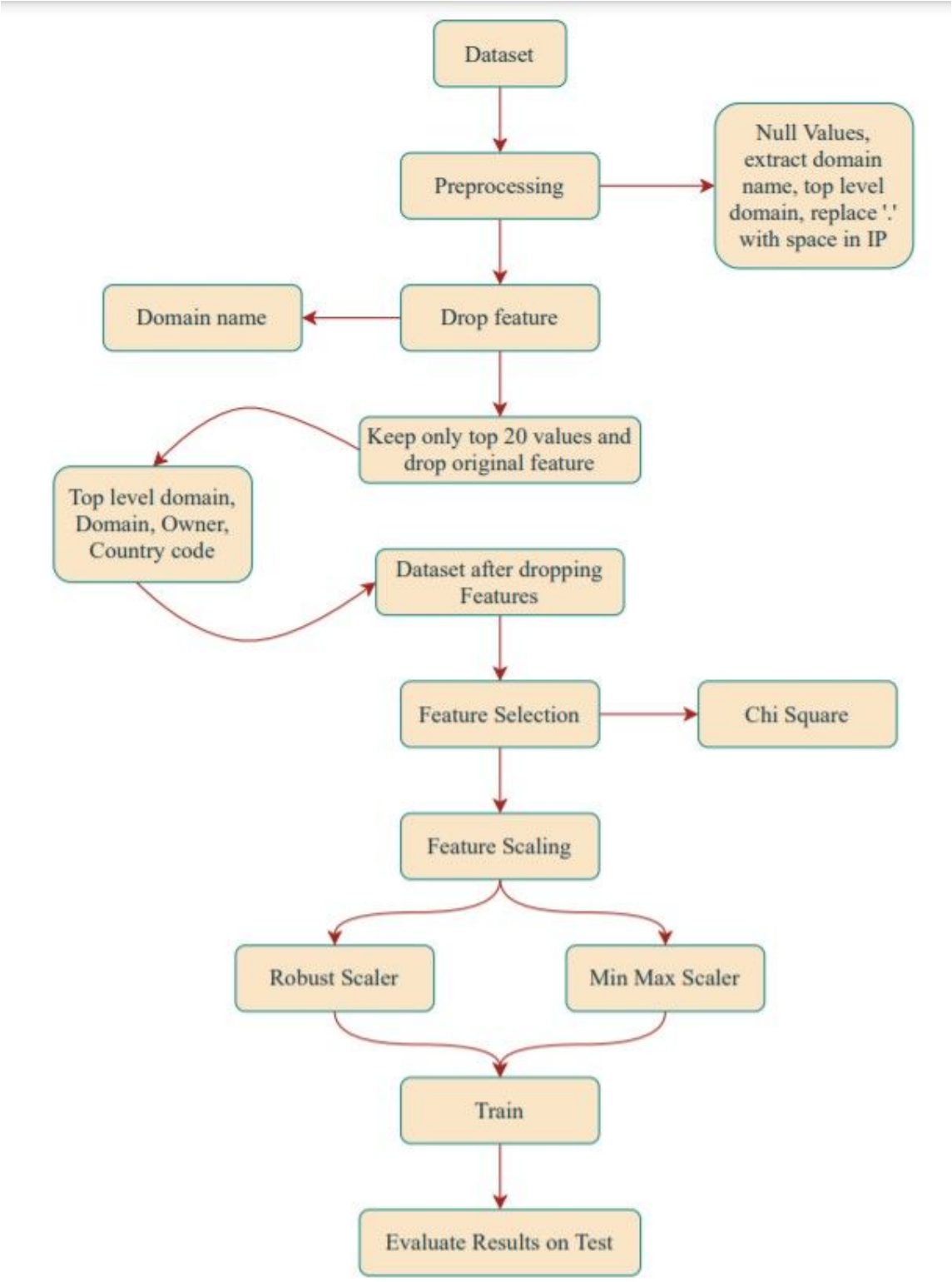

Figure 2

Research Workflow

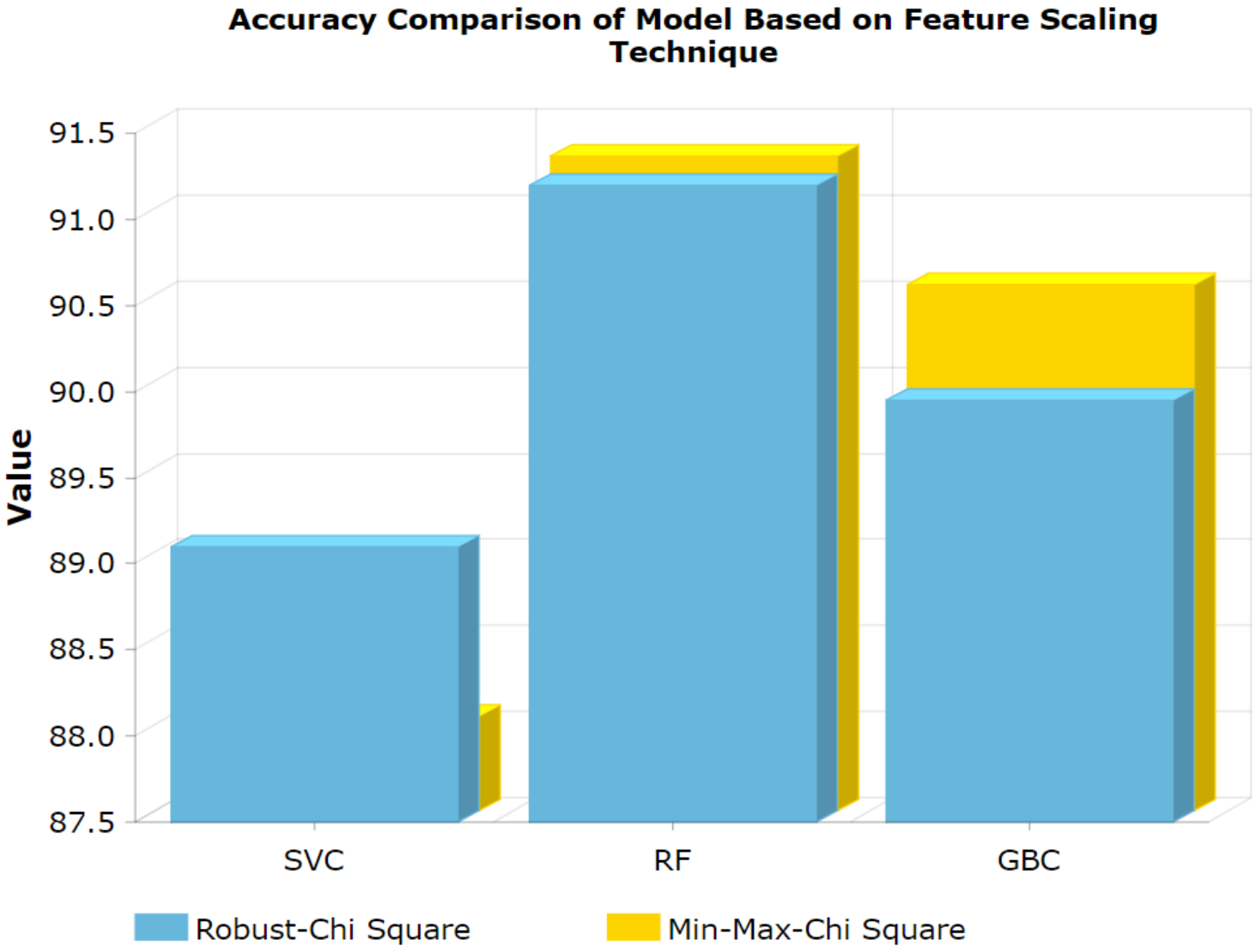

Figure 3

Accuracy Comparison based on feature scaling technique

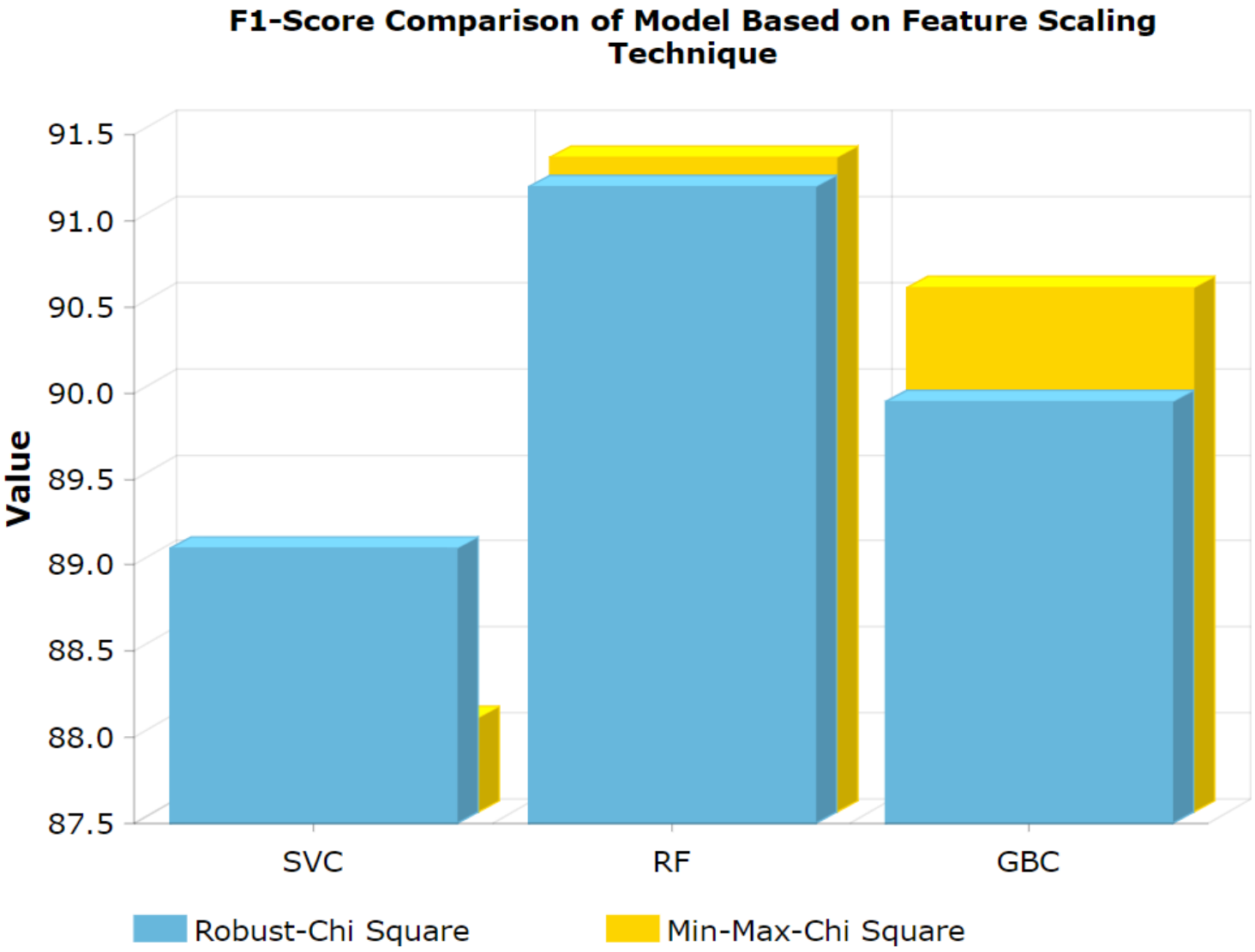

Figure 4

F1-Score comparison based on feature scaling technique