International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 10 | Oct 2020 www.irjet.net p-ISSN: 2395-0072

PHISHING WEBSITE DETECTION USING DIFFERENT MACHINE LEARNING TECHNIQUES

Challa Sai Bhanu Teja[1], Tanikella Sai Siva Sasank[2], Yakasiri Jeevan Sreeram Reddy[3]

[1]B. Tech Student, Computer Science, Koneru Lakshmaiah Education Foundation, Andhra Pradesh, India
[2,3]B. Tech Student, Electronics and Communication, Vel Tech Rangarajan Dr.Sagunthala R&D Institute of Science and Technology, Tamil Nadu, India

---

Abstract - *Phishing websites are proliferating day by day. Internet users are under perilous risk by these phishing websites. How can anyone differentiate a website as a phishing website or a bona fide website? A website identification as a phishing website depends on different factors like URL length, shortening service, having a special case letter '@', double slash redirecting, having subdomains, etc. Although the above-mentioned factors are present on a website, no one can assert that website as a phishing website, it could be a bona fide website too. To solve this conundrum, machine learning algorithms are used, as they provide efficient results. In this research paper, various machine learning and deep learning algorithms are tested, and the respective algorithm accuracies are benchmarked.*

Key Words: **Phishing website detection, ANN, classification, Decision tree classifier, contrast of machine learning models, Random Forest Classifier.**

1. INTRODUCTION

What is phishing? Phishing has been recognized as a mendacious attempt to acquire personal information or data, such as usernames, passwords, home addresses, Aadhaar card details and credit card details, by disguising oneself as a trustworthy company in an electronic communication. Typically carried out by email spoofing, instant messaging, and text messaging, internet users are directed by phishing to enter personal information at a fake website that matches the look and feel of the legitimate site.

Internet users are swindled into entering their personal information. Users impulsively trust social websites which are actually phishing websites. This is a technique called social engineering. Various factors with phishing incidents include training the user, public and social awareness, and security measures. If anyone receives an email (or instant message) from someone they don't know directing, you to sign in to a website then they may have received a phishing email with links to a phishing website. Generally, fraudsters try to deceive users into providing their information so that they can gain access to an online account. Once they gain access to your intel, they use your personal information to commit identity theft, charge your credit cards, empty your bank accounts, read your email, and lock you out of your online account. The problem of detecting phishing websites has been addressed many times using various methods not from unconventional classifiers but from conventional classifiers to more complex hybrid methods.

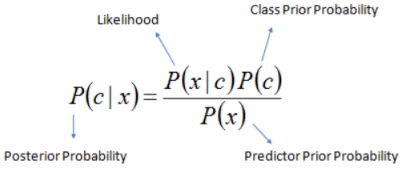
Machine learning techniques have proven themselves as the best techniques in solving complex classification problems. As this is a classification problem, we are testing the accuracy of different machine learning algorithms to solve this conundrum by using "Phishing Website Dataset" from "Kaggle". Machine learning algorithms that are tested in this paper are Decision tree classifier, K-nearest neighbors, logistic regression, naive Bayes, support vector machine (SVM), Random forest, Gradient boosting tree classifier, and artificial neural network.

2. CLASSIFICATION MACHINE LEARNING ALGORITHMS

These are the classification algorithms in machine learning.

2.1 Naive Bayes Classifier (Generative Learning Model)

A classification technique that is based on Bayes' Theorem with the presumption of independence among predictors. Naive Bayes is a way used to predict the class of the dataset. Using this, one can perform a multi-class prediction. If the assumption of independence is valid, then Naive Bayes is much more capable than the other algorithms like logistic regression. Furthermore, less training data is required for the classification. Naive Bayes classifier works efficiently in real-world situations such as document classification and spam filtering. Although, it is merely recognized as a bad estimator. It is an easy and a quick technique



$$P(c|X) = P(x_1|c)\times P(x_2|c)\times\cdots\times P(x_n|c)\times P(c)$$

P(c/x) is the posterior probability of class (target) given predictor (attribute).

P(c) is the prior probability of class.

P(x/c) is the likelihood which is the probability of predictor given class.

P(x) is the prior probability of predictor.

2.2 K-Nearest Neighbor

The k-nearest-neighbors algorithm, which is a supervised classification technique that uses juxtaposition for 'sameness'. The algorithm takes a bunch of triggered points and uses them to learn how to trigger other points. These KNNs are used in real-life situations where non-parametric and non-statistical algorithms are required. These algorithms do not make any presumptions about how the data distribution is done. When we are given primary data, the KNN classifies the coordinates that are identified by a specific attribute. Although, the only disadvantage is to determine the value of K and the computation cost is high as it needs to compute the distance of each instance to all the training samples.

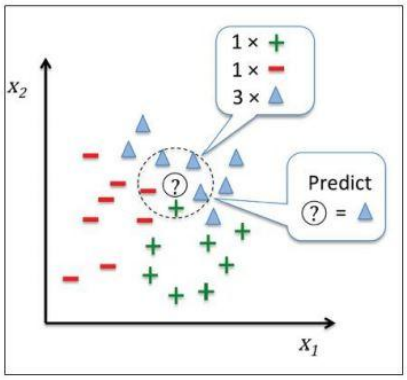


Fig-1: Finding neighbors and Voting for labels

2.3 Logistic Regression (Predictive Learning Model)

It is a method for analyzing a data set in which there are one or more than one independent variables that are to determine an outcome. The outcome is contrasted with a dichotomous variable (in which there are only two possible outcomes). In this algorithm, the probability describing the suitable outcomes of a single trial are shaped using a logistic function. Logistic regression is designed for this purpose, and is the most useful technique for understanding the influence of independent variables on a single outcome variable. Although, it works only when the predicted variable is binary, assumes all predictors are independent of each other and assumes data is free of missing values.

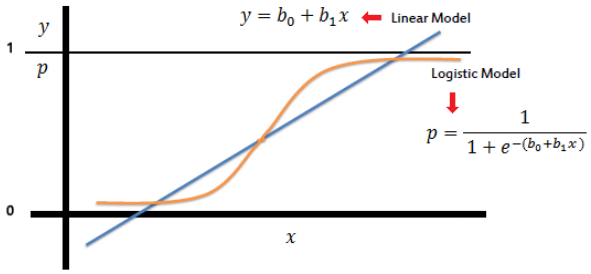


Fig-2: Contrast plot between linear and logistic models

2.4 Decision Trees

Decision tree develops or builds classification or regression models like a tree structure. It divides a data set into shards of subsets while an associated decision tree is developed at the same time. The final result consists of a tree with decision and leaf nodes. A decision node that has two or more branches and a leaf node, indicates a classification or decision. Decision tree could create complicated trees that are not generalized well, and decision trees could be unstable because of little variations in the data that might result in a completely different tree that is being generated. Overall, Decision Tree is simple to understand and visualize, requires little data preparation, and can handle both numerical and categorical data.

$$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

Using the frequency table of one attribute

$$E(T, X) = \sum_{c \in X} P(c) E(c)$$

Using the frequency table of two attributes

2.5 Random Forest

Random forests or random decision forests are group learning method for classification, and regression tasks which functions by forming a lot of decision trees at training time and resulting in the class which is the mode of the classes or mean prediction of the respective trees. Reduction in over-fitting is more accurate than decision trees in many cases. The only disadvantage is slow real time prediction, difficult to implement, and complex algorithm.

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (fi - yi)^2$$

2.6 SUPPORT VECTOR MACHINE

Support Vector Machine is a type of supervised machine learning algorithm that provides data analysis for classification and regression analysis. SVM is mostly used for classification. The value of each feature is equal to the value of the specified coordinate. Then, we detect the ideal hyperplane that differentiates between the two classes. Support vector machine is a representation as points in space contrasted into categories by a gap that is as wide as possible of the training data. It is effectual and efficient in high dimensional spaces and uses a subset of training points in the decision function, hence, it is also known for its memory efficiency. The algorithm indirectly provides probability estimations; these are calculated using five-fold cross-validation.

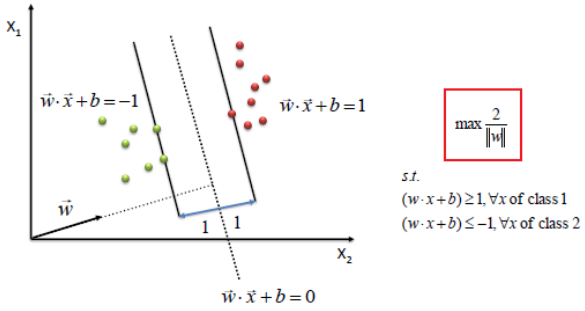


Fig-3: Two support vectors with hyperplane

2.7 GRADIENT BOOSTING TREE CLASSIFIER

Gradient boosting classifiers are a cluster of machine learning algorithms that amalgamates feeble learning models together to engender a strong predictive model. Decision trees are usually used in gradient boosting. Gradient boosting models are becoming popular as they are effective in classifying complex datasets. Gradient boosting is an algorithm that can over-fit a training dataset quickly. It benefits from regularization methods that penalize various parts of the algorithm and by reducing overfitting, it ameliorates the performance of the algorithm.

2.8 ARTIFICIAL NEURAL NETWORK

A neural network consists of neurons, arranged in layers that convert an input vector into output vector. Each neuron takes an input, and applies a non-linear function to it, and passes the output on to the next layer. An artificial neural network (ANN) is the shred of a computing system designed the simulation of human brain analyses and processes information. It solves problems that would prove complex by humans or statistical standards. ANN has self-learning capabilities that are enabled to produce the best results as more data becomes available.

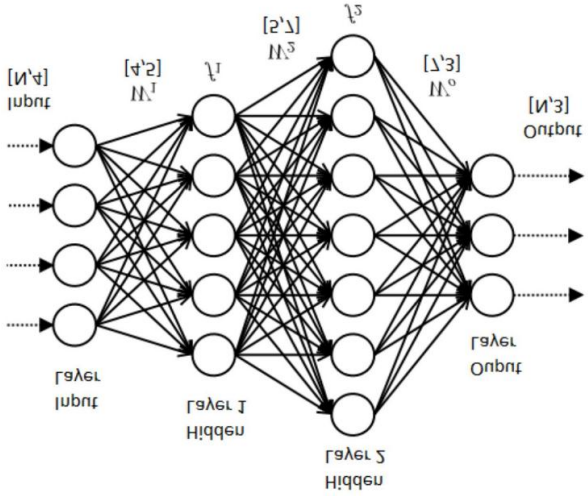


Fig-4: Architecture of Artificial Neural Network

3. IMPLEMENTATION

The implementation process consists of multiple parts. They are:

3.1 DATA PRE-PROCESSING

In any Machine Learning project, data pre-processing is that process in which data gets transformed, or encoded, to bring it to such a state that now the machine can easily understand it. In other words, the features of the data can now easily be decoded by the algorithm. Similarly, in this project, the dataset should be pre-processed before using any machine learning algorithm. Fortunately, our dataset from Kaggle has no null values hence, the null values need not be filtered. Therefore, we can analyze all the factors or fields that are responsible for a website to become a phishing website. A lot of factors or fields are present in the Kaggle dataset. Each factor or field consists of only three values (-1 or 0 or 1). '-1' represents that the website is a phishing website, '0' represents that the website is a suspicious (maybe a phishing website or maybe not) website, '1' represents that the website is not a phishing website, and it is a Legitimate website.

In the dataset, there are 32 columns and 11055 rows. The first column is the index number of the respective row. As the output does not depend on the index number of the row, we can remove this column or field from our dataset. Our output depends on the fields which are from the second column to the thirty-first column so these columns or fields should be present in our dataset. The thirty-second column is our required output. We need to divide the dataset into two parts (input columns and output columns). In our new dataset, the input is thirty columns, and the output is the thirty-first column. We need to divide the input data frame into two test and train data frames, and the same goes for the output data frame. We should have 80% of the rows

in train datasets and 20% of the rows in test dataset. This can be easily done using the "train_test_split" method from 'sklearn' library.

```
import pandas as pd
import numpy as np
```

```
df = pd.read_csv("dataset.csv")

x = df.iloc[:,1:-1].values
y = df.iloc[:,-1].values

y1 = []

for i in y:
    if(i==-1):
        y1.append(0)
    else:
        y1.append(1)
y = np.array(y1)

from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.2, random_state=0)
```

Fig-5: Pre-processing implementation code

3.2 CREATION OF MACHINE LEARNING MODEL

In this research paper, we have created several machine learning models and have trained them. The models are imported from the sklearn library. Different machine learning models are present in the sklearn library and the choice of which one to use generally depends on the problem at hand. No machine learning model works best for all kinds of problems. So, our job in this research paper is to test multiple models and to find out which one yields the best accuracy. As this is a classification problem, we have tested only those machine learning algorithms which are used for solving classification problems. The machine learning algorithms which we used in this research paper are,

1. Naive Bayes Classifier

2. K-Nearest Neighbor

3. Logistic Regression

4. Decision Tree

5. Random Forest

6. SUPPORT VECTOR MACHINE

7. GRADIENT BOOSTING TREE CLASSIFIER

8. ARTIFICIAL NEURAL NETWORK

The above-mentioned machine learning models except for ANN are in-built machine learning models and they are imported from the sklearn library. For KNN, we have assigned three neighbors in our model. In the Random Forest classifier, the number of estimators are set to 100. In Logistic regression, we have selected “liblinear” as our solver. In SVM, we have chosen the kernel as linear kernel.

We have created a custom ANN using the 'Keras' library. There are a total of 9 layers including the input and the output layers. The dimension of the input layer is 30. We have used Rectified Linear Unit (ReLU) for all the layers except for the final layer. In every layer except for the last layer, there are fifteen neurons but for the last layer there is only one neuron. The input layer consists of thirty neurons. For the final layer, we have used the sigmoid activation function which is best for classification problems. For the loss function, we have used binary_crossentropy, which works best for binary classification problems. The only problem with the binary_crossentropy loss function is, it gives the output as 0 or 1 whereas we require the output as -1 or 1 hence, we had to change the output -1 to 0. For the loss function, we have used binary_crossentropy, which works best for binary classification problems. For optimizer, we have chosen the 'Adam optimizer'. The neural network that is used in this problem is a sequential model with accuracy as a metric.

```
from keras.models import Sequential
from keras.layers import Dense
model = Sequential()
model.add(Dense(15, input_dim=30, activation='relu'))
model.add(Dense(15, activation='relu'))
model.add(Dense(15, activation='relu'))
model.add(Dense(15, activation='relu'))
model.add(Dense(15, activation='relu'))
model.add(Dense(15, activation='relu'))
model.add(Dense(15, activation='relu'))
model.add(Dense(15, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

```
Using TensorFlow backend.
```

```
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

Fig-6: Custom ANN implementation

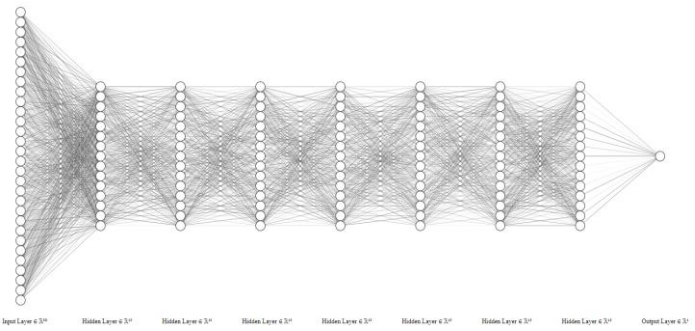


Fig-7: Custom ANN architecture

3.3 TRAINING THE MODEL

Training a model simply means learning (determining) good values for all the weights and the bias from the input dataset. A machine learning algorithm constructs a model by verifying multiple examples and

attempting to find a model that reduces loss and this process is called empirical risk minimization. Loss is the penalty for a bad prediction. The loss is a number indicating how bad the model's prediction is on a single example. If the model's prediction is perfect, the loss is zero otherwise, the loss is greater than zero. The primary objective of training a model is to find weights and biases that have *low* loss across all the examples in a dataset. The process of training a Machine Learning model involves an algorithm with training data to learn. The term ML model refers to the model antiquity that is engendered by the training process. The training data must have the correct answer, which is known as a target. The learning algorithm determines patterns in the training data that map the input data attributes to the target, and it outputs an ML model that got accustomed to these patterns. In this problem, we have used 'fit' method to train all the machine learning models, by passing train input and train output datasets as parameters. We have trained the custom ANN for 1000 epochs with batch size as 250.

3.4 EVALUATING THE MODEL

The main metrics used to evaluate a classification model are accuracy, recall, precision, specificity, F1 score and confusion matrix.

3.4.1 ACCURACY

Accuracy is defined as the percentage of correct predictions which are predicted from the test data. It is calculated by dividing the number of correct predictions from the total number of predictions. The most commonly used metric to valuate a model and it is actually not a clear indicator of the performance. The worse happens when classes are imbalanced.

$$Accuracy = \frac{Number\ of\ Features\ classified\ correctly}{Number\ of\ Features}$$

3.4.2 RECALL

Recall is defined as the shard of outcomes which were predicted to belong to a class with respect to all of the examples that truly belong in the class. Therefore, denominator is the actual number of positive occurences present in the dataset.

$$\frac{TP}{TP + FN}$$

3.4.3 PRECISION

Precision is defined as the percentage of positive instances out of total predicted positive instances. Here, denominator is the model extrapolation done as positive from the whole dataset.

$$\frac{TP}{TP + FP}$$

3.4.4 SPECIFICITY

Specificity is defined as the percentage of negative instances out of total actual negative instances. Therefore, denominator is the actual number of negative instances present in the dataset. It is as similar as the recall, but the shift is on the negative instances. It is a measure to see how separate the classes are.

$$\frac{TN}{TN + FP}$$

3.4.5 F1 score

It is defined as the harmonic mean of both precision and recall. This takes the contribution of both precision and recall. Hence, if the F1 score is higher, then the result is better. A model does well in F1 score, if the positive predicted instances are actually positives (precision) and doesn't miss out on positive instances and predicts them negative (recall). One drawback is that both precision and recall are given equal importance according to our application. We may need one higher than the other and F1 score is not viable to be the exact metric for it.

$$\frac{2}{\frac{1}{precision} + \frac{1}{recall}} = \frac{2 * precision * recall}{precision + recall}$$

3.4.6 CONFUSION MATRIX

A Confusion matrix is a matrix used for evaluating the performances of classification models, the rows and columns present are the number of target classes. The matrix contrasts the actual target values with those predicted values by the machine learning model. This provides us a holistic view of how our classification model is performing and what types of errors it is making.

		Actual Values: Positive (1)	Actual Values: Negative (0)
Predicted Values	Positive (1)	TP	FP
Predicted Values	Negative (0)	FN	TN

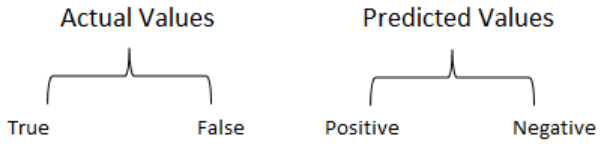


Fig-8: General confusion matrix

A confusion matrix is a method used for summarizing the performances of classification algorithms. Classification accuracies alone can be misleading if we have an unequal number of observations in every class or if we have more than two classes in our dataset. For our research paper, we have collected various confusion matrices and accuracies of various machine learning models that are discussed earlier in the paper. The contrast between the models will be discussed in the results.

4. RESULT

Machine Learning Algorithm	Accuracy	Precision	Recall	Specificity	F1 score
Random Forest Classifier	0.970149254	0.984662577	0.949704	0.987469	0.966867
Artificial Neural Network	0.968340099	0.765553869	0.995069	0.741855	0.865352
Decision Tree Classifier	0.961555857	0.968718466	0.946746	0.974102	0.957606
K-Nearest Neighbors	0.949344188	0.956477733	0.931953	0.964077	0.944056
Gradient Boosting Tree Classifier	0.9461782	0.950654582	0.930966	0.959064	0.940708
Support Vector Machine	0.91813659	0.927179487	0.891732	0.940685	0.90911
Logistic Regression	0.917232022	0.923547401	0.893491	0.937343	0.908271
Naïve Bayes	0.615106287	0.543699732	1	0.289056	0.704411

Table-1: Evaluation of metrics of all machine learning algorithms

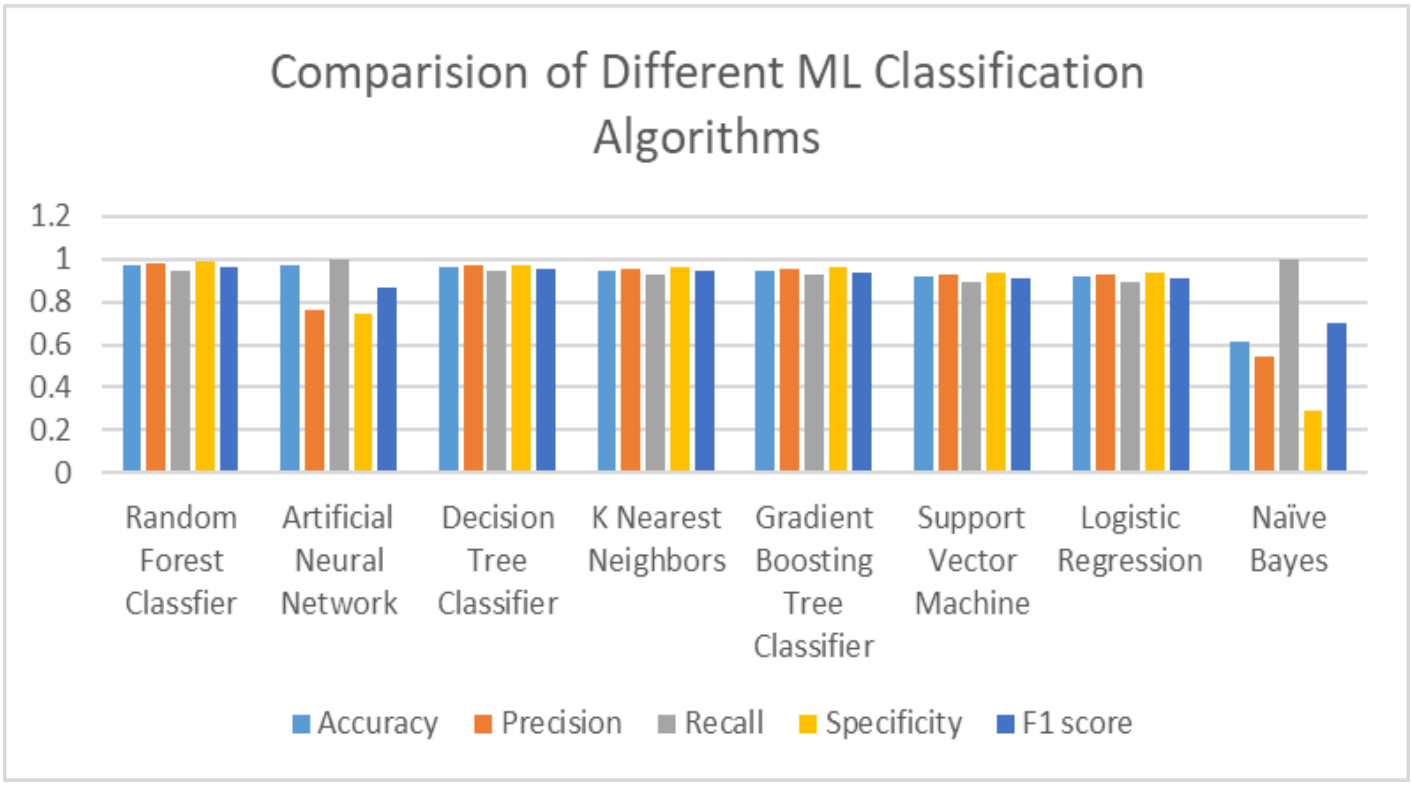


Fig-9: Bar plot of metrics of all machine learning algorithms

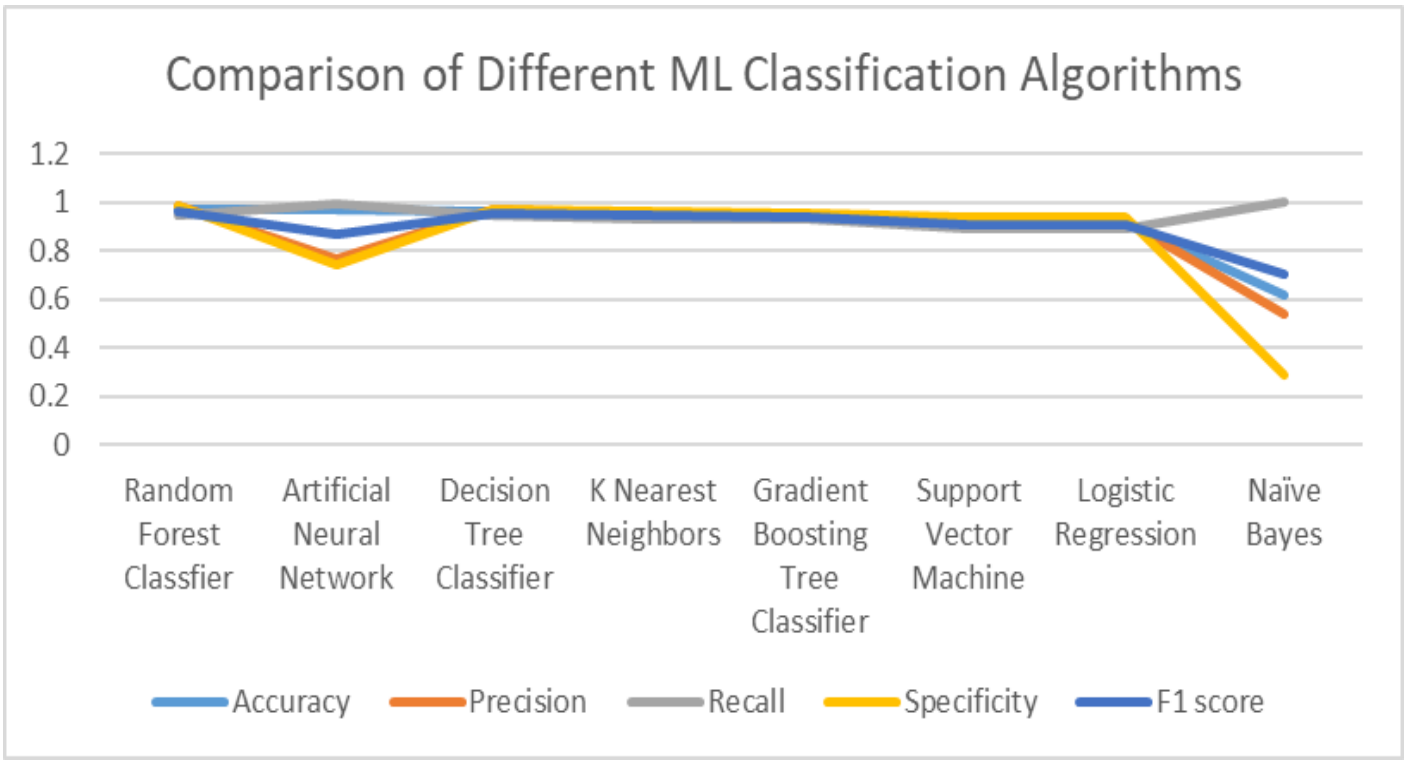


Fig-10: Contrast between metrics of all algorithms

5. CONCLUSION

We can conclude from the above results and plots that Random Forest Classifier is the best algorithm for classification problems and the algorithm with lowest metrics is Naïve Bayes, as it is the least effective algorithm for solving classification problems. Artificial Neural Network is almost as effective as Random Forest Classifier. If anyone can architect an ANN by using hyper parameter tuning, then it can be more effective than Random Forest Classifier. Random Forest Classifier is the best algorithm to use for better results with less time and work.

REFERENCES

[1] Baik S, Bala J (2004) A decision tree algorithm for distributed data mining: towards network intrusion detec1038 tion. Lecture notes in computer science, vol. 3046, pp 206–212

[2] Gehrke J, Ramakrishnan R, Ganti V (2000) RainForest—a framework for fast decision tree construction of large datasets. Data Min Knowl Disc 4(2–3):127–162R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[3] X. Qi and B.D. Davidson. Web page classification: Features and algorithms, ACM Computing Surveys (CSUR), 2009, vol. 41, issue 2, pp. 1–31

[4] T. Joachims. Transductive inference for text classification using support vector machines. In Proceedings of the International Conference on Machine Learning (ICML'99), 1999, pp. 200–209

[5] I. Ben-Gal, "Bayesian Networks", In Ruggeri, Fabrizio; Kennett, Ron S.; Faltin, Frederick W. Encyclopedia of Statistics in Quality and Reliability," John Wiley &Sons, ISBN 978-0-470-01861-3, 2007

[6] Alexander J. Stimpson and Mary L. Cummings in "Accessing Intervention Timing in Computer-Based Education using Machine Learning Algorithms".

[7] A Deep Learning Tutorial: From Perceptrons To Deep Networks

[8] Quinlan, J. R. (1986). Induction of decision trees. Machine Learning, 1, 81-106.

[9] Rivest, R. L. (1987). Learning decision lists. Machine Learning, 2, 229-246.

[10] Breiman, L., Random Forests, Machine Learning 45(1), 5-32, 2001.

[11] Introduction to Decision Trees and Random Forests, Ned Horning; American Museum of Natural History's

[12] Breiman, L.: Random Forests. Machine. Learning. 45, 5–32 (2001). DOI 10.1023/A:1010933404324] Breiman, L.: Random Forests. Machine. Learning. 45, 5–32 (2001). DOI 10.1023/A:1010933404324

[13]] E. E. Tripoliti, D. I. Fotiadis, and G. Manis, "Modifications of the construction and voting mechanisms of the Random Forests Algorithm," Data and Knowledge Engineering, vol. 87, pp. 41–65, 2013.

[14] H. Bostrom, "Estimating class probabilities in random forests," Sixth International Conference on Machine Learning and Applications, pp. 211–216, 2007.

[15] T. G. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization," Machine learning, vol. 40, no. 2, pp. 139–157, 2000

[16] Y. Freund and R. Shapire. Experiments with a new boosting algorithm. In L. Saitta, editor, Machine Learning: Proceedings of the 13th International Conference, pages 148–156, San Francisco, 1996. Morgan Kaufmann.

[17] Y. Lin and Y. Jeon. Random forests and adaptive nearest neighbors. Journal of the American Statistical Association, 101:578–590, 2006.

BIOGRAPHIES

Challa Sai Bhanu Teja
B. Tech Student, Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Andhra Pradesh, India.

Tanikella Sai Siva Sasank
B. Tech Student, Department of Electronics and Communication Engineering, Vel Tech Rangarajan Dr.Sagunthala R&D Institute of Science and Technology, Tamil Nadu, India.

Yakasiri Jeevan Sreeram Reddy
B. Tech Student, Department of Electronics and Communication Engineering, Vel Tech Rangarajan Dr.Sagunthala R&D Institute of Science and Technology, Tamil Nadu, India.