Minor Project Report on

Fake-News Detection

Using Machine Learning Algorithm

Report submitted in fulfillment of the requirement for the degree of Bachelor of Technology in Computer Science and Communication Engineering

SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

DEEMED TO BE UNIVERSITY, BHUBANESWAR

Under the Guidance of:

Dr. Himansu Das

Submitted By:

Avirup Chattopadhyay (1929209)

Adrita Chatterjee (1929133)

Harsh Raj(1929216)

Prayash Mohapatra (1929232)

CERTIFICATE

This is to certify that the project report entitled "FAKE-NEWS DETECTION" was submitted by:-

ADRITA CHATTERJEE (1929133)

AVIRUP CHATTOPADHYAY (1929209)

HARSH RAJ (1929216)

PRAYASH MOHAPATRA (1929232)

In partial fulfillment of the requirement for the degree of Bachelor of Technology in Computer Science and Communication Engineering is a bona fide record of the work carried out under the guidance and supervision at the School of Computer Science Engineering, KIIT (Deemed to be University).

Signature of Supervisor

Dr. Himansu Das

School of Computer Science Engineering

KIIT (Deemed to be University)

ABSTRACT

Fake News contains misleading information that could be checked. This maintains lie about a certain statistic in a country or exaggerated cost of certain services for a country, which may arise unrest for some countries like in Arabic spring. There are organizations, like the House of Commons and the Crosscheck project, trying to deal with issues as confirming authors are accountable. However, their scope is so limited because they depend on human manual detection, in a globe with millions of articles either removed or being published every minute, this cannot be accountable or feasible manually. A solution could be, by the development of a system to provide a credible automated index scoring, or rating for credibility of different publishers, and news context.

This paper proposes a methodology to create a model that will detect if an article is authentic or fake based on its words, phrases, sources and titles, by applying supervised machine learning algorithms on an annotated (labeled) dataset, that are manually classified and guaranteed. Then, feature selection methods are applied to experiment and choose the best fit features to obtain the highest precision, according to confusion matrix results. We propose to create the model using different classification algorithms. The product model will test the unseen data, the results will be plotted, and accordingly, the product will be a model that detects and classifies fake articles and can be used and integrated with any system for future use.

INTRODUCTION

Machine learning (ML) is an application of artificial intelligence (AI) that allows systems to automatically learn and improve from their experiences without the need to be explicitly programmed in advance. It is the study of computer algorithms that may improve themselves autonomously as a result of experience and the collection and analysis of data. For the purpose of making predictions or making judgments, machine learning algorithms construct a model based on sample data, which is referred to as training data.

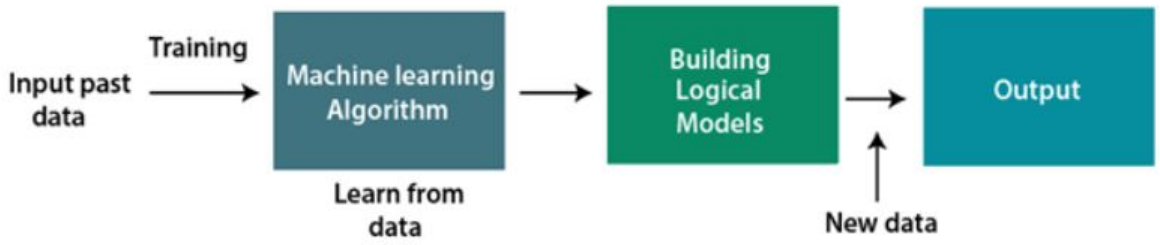
Computer vision, medicine, email filtering, and speech recognition are just a few of the fields where machine learning algorithms are being utilized to solve problems where it is difficult or impossible to create traditional algorithms to accomplish the required tasks.We begin the process of learning with observations or data (such as examples, first hand experience, or instruction), with the goal of identifying patterns in data and making better judgments in the future based on the examples we provide. The fundamental goal is to enable computers to learn on their own, without the need for human interaction or support, and to change their behavior accordingly.

In many cases, machine learning and data mining are used in conjunction with one another and have significant overlap; however, while machine learning focuses on prediction based on known properties learned from the training data, data mining focuses on the discovery of (previously unknown) properties in data; this is the analysis step of knowledge discovery in databases. When it comes to data mining, there are various machine learning methods to choose from, each with its own set of aims. On the other hand, machine learning uses data mining approaches as "unsupervised learning" or as a preprocessing step to increase learner accuracy.

Computational learning theory is an area of theoretical computer science that studies the computational analysis of machine learning algorithms and their performance. Given the limited number of training sets available and the uncertainty surrounding the future, learning theory does not always provide guarantees about the performance of algorithms. Instead, probabilistic bounds on the performance are fairly popular in computer science applications. The bias–variance decomposition is one method of estimating generalization error.

The hypothesis's complexity should be proportional to the complexity of the function underlying the data in order to achieve the greatest performance in the

context of generalization.. Similarly, if the hypothesis is less complex than the function, the model will have under-fitted the observed data. The training error lowers if the complexity of the model is increased in response to the response variable.



Classification of Machine Learning

1. Supervised Learning
2. Unsupervised Learning
3. Reinforcement Learning

1) Supervised Learning

Supervised learning is a type of machine learning method in which we provide sample labeled data to the machine learning system in order to train it, and on that basis, it predicts the output.

The system creates a model using labeled data to understand the datasets and learn about each data, once the training and processing are done then we test the model by providing a sample data to check whether it is predicting the exact output or not.

The goal of supervised learning is to map input data with the output data. The supervised learning is based on supervision, and it is the same as when a student learns things in the supervision of the teacher. The example of supervised learning is **spam filtering**.

Supervised learning can be grouped further in two categories of algorithms:

- **Classification**
- **Regression**

2) Unsupervised Learning

Unsupervised learning is a learning method in which a machine learns without any supervision.

The training is provided to the machine with the set of data that has not been labeled, classified, or categorized, and the algorithm needs to act on that data without any supervision. The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns.

In unsupervised learning, we don't have a predetermined result. The machine tries to find useful insights from the huge amount of data. It can be further classifieds into two categories of algorithms:

- **Clustering**
- **Association**

3) Reinforcement Learning

Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action. The agent learns automatically with these feedbacks and improves its performance. In reinforcement learning, the agent interacts with the environment and explores it. The goal of an agent is to get the most reward points, and hence, it improves its performance.

The robotic dog, which automatically learns the movement of his arms, is an example of Reinforcement learning.

Modules used in the Project

- **NumPy**:-It is a general purpose array processing package.it provides a high performance multidimensional array object, and tools for working with these arrays. It's also an efficient multidimensional container of generic data.
- **Pandas**:- It is the most popular python library that is used for data analysis.it provides highly optimized performance with back-end source code is purely written is **C** or **python .**subdivided into two parts it is :- **series and dataframe.**
- **Matplotlib**:-It is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environment across platforms .it tries to make easy things easy and hard things possible. you can

generate plots, histograms, power spectra, bar charts, error charts, scatterplots, etc.,with just a few lines of code.

- **Seaborn**:-It is a python data visualization library based on matplotlib.it provides a high level interface for drawing attractive and informative statistical graphics.it is closely integrated with pandas datastructures.

- **Scikit-learn**:-it is a free machine-learning library for python programming language.it features various classification ,regression and clustering algorithms including support vector machines ,random forests, gradient boosting ,k-means, DBSCAN and is designed to incorporate with the python numerical and scientific libraries numpy and scypy.

Classification algorithms:

Here is the list of commonly used machine learning algorithms. These algorithms can be applied to almost any data problem:

1. Logistic Regression
2. SVM (Support Vector Machine)
3. KNN (K-Nearest Neighbours)
4. Decision Trees
5. Naive Bayes
6. ANN (Artificial Neural Network)

1. Logistic Regression

It is a classification not a regression algorithm. It is used to estimate discrete values (Binary values like 0/1, yes/no, true/false) based on given set of independent variables. In simple words, it predicts the probability of occurrence of an event by fitting data to a logit function. Hence, it is also known as **logit regression**. Since, it predicts the probability, its output values lies between 0 and 1 (as expected).

odds= p/ (1-p) = probability of event occurrence / probability of not event occurrences

$$\ln(odds) = \ln(p/(1-p))$$

$$logit(p) = \ln(p/(1-p)) = b0+b1X1+b2X2+b3X3....+bkXk$$

Above, p is the probability of presence of the characteristic of interest. It chooses parameters that maximize the likelihood of observing the sample values rather than that minimize the sum of squared errors (like in ordinary regression).

Algorithm for Logistic Regression:

Step 1: Data Preprocessing of dataset Fake.csv and true.csv

Step 2: Import the LogisticRegression class of the sklearn library.

Step 3: Create a classifier object and use it to fit the model to the logistic regression.

Step 4: Fitting Logistic Regression to the Training set

Step 5: Predicting the test result

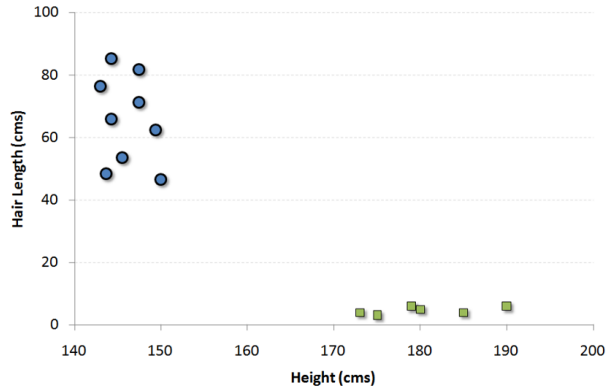
Step 6: Test accuracy of the result

Step 7: Visualizing the test set result.

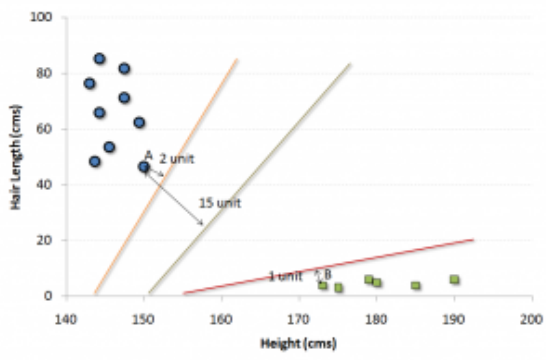
2. SVM (Support Vector Machine)

It is a classification method. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate.

For example, if we only had two features like Height and Hair length of an individual, we'd first plot these two variables in two-dimensional space where each point has two co-ordinates (these co-ordinates are known as **Support Vectors**)



Now, we will find some *line* that splits the data between the two differently classified groups of data. This will be the line such that the distances from the closest point in each of the two groups will be farthest away.



In the example shown above, the line which splits the data into two differently classified groups is the *black* line, since the two closest points are the farthest apart from the line. This line is our classifier. Then, depending on where the testing data lands on either side of the line, that's what class we can classify the new data as.

Algorithm for SVM:

Step 1: Data Preprocessing of dataset Fake.csv and true.csv

Step 2: Import the LinearSVC class of the sklearn library.

Step 3: Create a classifier object and use it to fit the model to the logistic regression.

Step 4: Fitting LinearSVC to the Testing set

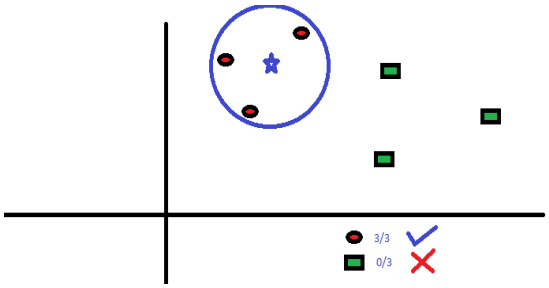
Step 5: Predicting the test results

Step 6: Test accuracy of the result using confusion matrix

Step 7: Visualizing the test set result using classification report

3. KNN (k- Nearest Neighbors)

It can be used for both classification and regression problems. However, it is more widely used in classification problems in the industry. K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases by a majority vote of its k neighbors. The case being assigned to the class is most common amongst its K nearest neighbors measured by a distance function. These distance functions can be Euclidean, Manhattan, Minkowski and Hamming distance. First three functions are used for continuous function and fourth one (Hamming) for categorical variables. If K = 1, then the case is simply assigned to the class of its nearest neighbor. At times, choosing K turns out to be a challenge while performing kNNmodelling.



KNN can easily be mapped to our real lives. If you want to learn about a person, of whom you have no information, you might like to find out about his close friends and the circles he moves in and gain access to his/her information!

Things to consider before selecting KNN:

- KNN is computationally expensive
- Variables should be normalized else higher range variables can bias it
- Works on pre-processing stage more before going for kNN like an outlier,noise removal.

Algorithm for KNN:

Step 1: Data Preprocessing of dataset Fake.csv and true.csv

Step 2: Import the KNeighborsClassifier class of the sklearn library.

Step 3: Create a classifier object and use it to fit the model to the logistic regression.

Step 4: Fitting KNeighborsClassifier to the Training set

Step 5: Predicting the test result

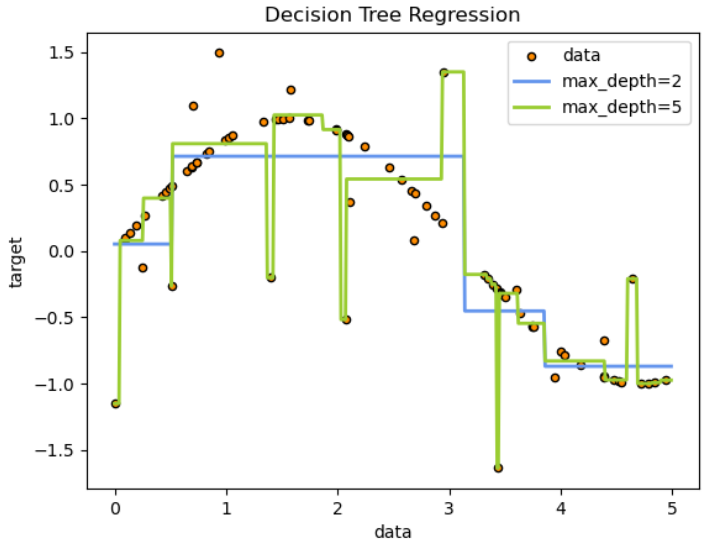
Step 6: Test accuracy of the result using confusion matrix

Step 7: Visualizing the test set result using classification report

4. Decision Trees

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation.

For instance, in the example below, decision trees learn from data to approximate a sine curve with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.



> Information Gain= Entropy(S)- [(Weighted Avg) *Entropy(each feature)

Entropy: Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

`Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)`

Where,

- **S= Total number of samples**
- **P(yes)= probability of yes**
- **P(no)= probability of no**

> Gini Index= $1- \sum_j P_j^2$

Algorithm for Decision Trees:

Step 1: Data Preprocessing of dataset Fake.csv and true.csv

Step 2: Import the DecisionTreeClassifier class of the sklearn library.

Step 3: Create a classifier object and use it to fit the model to the logistic regression.

Step 4: Fitting DecisionTreeClassifier to the Training set

Step 5: Predicting the test result

Step 6: Test accuracy of the result using confusion matrix.

Step 7: Visualizing the test set result using classification report.

5. Naive Bayes (Bernoulli)

Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms where all of

them share a common principle, i.e. every pair of features being classified is independent of each other.

In spite of their apparently over-simplified assumptions, naive Bayes classifiers have worked quite well in many real-world situations, famously document classification and spam filtering. They require a small amount of training data to estimate the necessary parameters. (For theoretical reasons why naive Bayes works well, and on which types of data it does, see the references below.)

Naive Bayes learners and classifiers can be extremely fast compared to more sophisticated methods. The decoupling of the class conditional feature distributions means that each distribution can be independently estimated as a one dimensional distribution. This in turn helps to alleviate problems stemming from the curse of dimensionality.

Bernoulli Naive Bayes: The multivariate Bernoulli event model, features are independent booleans (binary variables) describing inputs. Like the multinomial model, this model is popular for document classification tasks, where binary term occurrence(i.e. a word occurs in a document or not) features are used rather than term frequencies(i.e. frequency of a word in the document).

The decision rule for Bernoulli naive Bayes is based on

$$P(x_i \mid y) = P(i \mid y)x_i + (1 - P(i \mid y))(1 - x_i)$$

<u>Algorithm for Naive Bayes (Bernoulli):</u>

Step 1: Data Preprocessing of dataset Fake.csv and true.csv

Step 2: Import the BernoulliNB class of the sklearn library.

Step 3: Create a classifier object and use it to fit the model to the logistic regression.

Step 4: Fitting BernoulliNB to the Training set

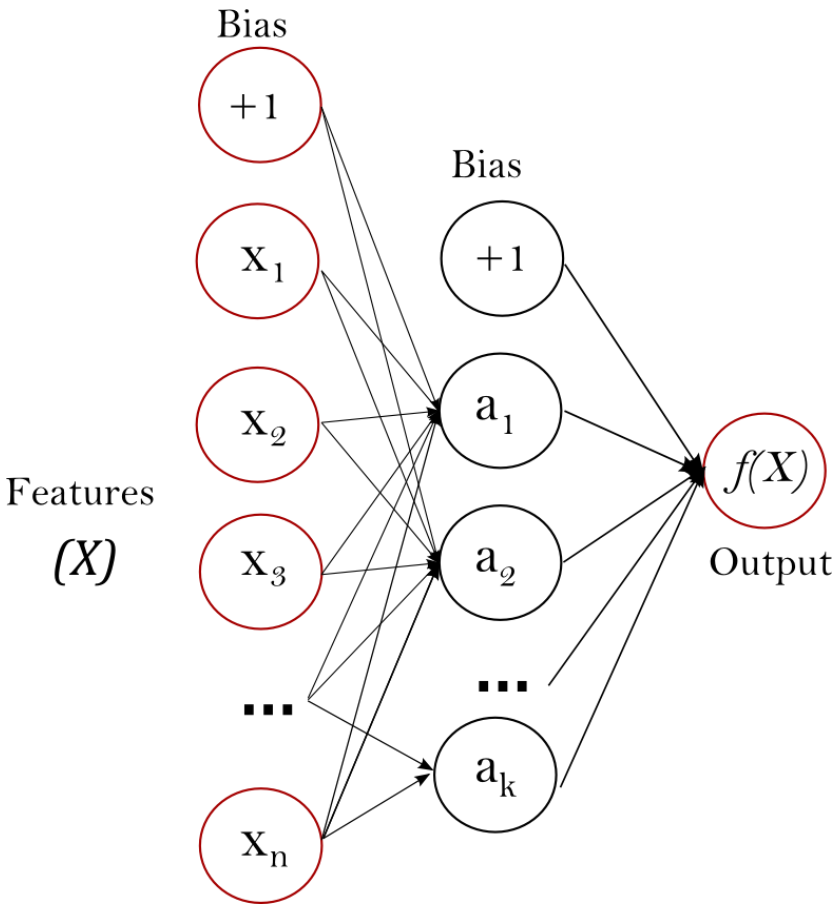
Step 5: Predicting the test result

Step 6: Test accuracy of the result

Step 7: Visualizing the test set result.

6. ANN (Artificial Neural Network)

Artificial Neural Networks are a special type of machine learning algorithms that are modeled after the human brain. That is, just like how the neurons in our nervous system are able to learn from the past data, similarly, the ANN is able to learn from the data and provide responses in the form of predictions or classifications.



ANNs are nonlinear statistical models which display a complex relationship between the inputs and outputs to discover a new pattern. A variety of tasks such as

image recognition, speech recognition, machine translation as well as medical diagnosis makes use of these artificial neural networks.

A neural network may contain the following 3 layers:

- Input layer – The activity of the input units represents the raw information that can feed into the network.
- Hidden layer – To determine the activity of each hidden unit. The activities of the input units and the weights on the connections between the input and the hidden units. There may be one or more hidden layers.
- Output layer – The behavior of the output units depends on the activity of the hidden units and the weights between the hidden and output units.

Algorithm for ANN:

Step 1: Data Preprocessing of dataset Fake.csv and true.csv

Step 2: Import the MLPClassifier class of the sklearn library**.**

Step 3: Create a classifier object and use it to fit the model to the logistic regression.

Step 4: Fitting MLPClassifier to the Training set

Step 5: Predicting the test result

Step 6: Test accuracy of the result

Step 7: Visualizing the test set result.

RESULT ANALYSIS

Decision Tree:

```
In [106]:
1 from sklearn.tree import DecisionTreeClassifier

In [107]:
1 DT = DecisionTreeClassifier()
2 DT.fit(xv_train, y_train)

Out[107]: DecisionTreeClassifier()

In [108]:
1 pred_dt = DT.predict(xv_test)

In [109]:
1 DT.score(xv_test, y_test)

Out[109]: 0.9956922162804516

In [110]:
1 print(classification_report(y_test, pred_dt))

              precision    recall  f1-score   support

           0       1.00      1.00      1.00      7083
           1       1.00      0.99      1.00      6381

    accuracy                           1.00     13464
   macro avg       1.00      1.00      1.00     13464
weighted avg       1.00      1.00      1.00     13464
```

Decision Tree						
	Precision	Recall	F1-Score	Accuracy	Confusion Matrix	Error (root mean)
Fake News Data set -1	1	1	1	1	[[7039 21] [24 6380]]	0.06333
Fake News Data set -2	0.98	0.98	0.98	0.99	[[1858 42] [40 6386]]	0.09924
Fake News Data set -3	0.68	0.58	0.63	0.94	[[363 264] [167 6260]]	0.25536
Fake News Data set -4	1	0.99	0.99	0.99	[[6942 41] [34 6453]]	0.07461

K Nearest Neighbour:

```
In [122]:
1 from sklearn.neighbors import KNeighborsClassifier

In [123]:
1 ng = KNeighborsClassifier()
2 ng.fit(xv_test, y_test)

Out[123]: KNeighborsClassifier()

In [124]:
1 pred_knn = ng.predict(xv_test)

In [125]:
1 ng.score(xv_test,y_test)

Out[125]: 0.672979797979798

In [126]:
1 print(classification_report(y_test, pred_knn))      # Precision , Recall , F-measure , Accuracy

              precision    recall  f1-score   support

           0       0.62      0.99      0.76      7083
           1       0.96      0.32      0.48      6381

    accuracy                           0.67     13464
   macro avg       0.79      0.66      0.62     13464
weighted avg       0.78      0.67      0.63     13464
```

K Nearest Neighbor						
	Precision	Recall	F1-Score	Accuracy	Confusion Matrix	Error
Fake News Data set -1	0.62	0.99	0.76	0.67	[[6989 94] [4309 2072]]	0.57354
Fake News Data set -2	0.31	0.98	0.47	0.5	[[1862 38] [4103 2323]]	0.70523
Fake News Data set -3	0.59	1	0.74	0.94	[[624 3] [437 5990]]	0.25144
Fake News Data set -4	0.61	0.99	0.75	0.67	[[6005 78] [4418 2009]]	0.57773

Support Vector Machine:

```
In [129]:  1 from sklearn.svm import LinearSVC

In [130]:  1 ls = LinearSVC()

In [131]:  1 ls.fit(xv_test, y_test)

Out[131]: LinearSVC()

In [132]:  1 pred_ls = ls.predict(xv_test)

In [133]:  1 ls.score(xv_test,y_test)

Out[133]: 0.9994800950683304

In [134]:  1 print(classification_report(y_test, pred_ls))      # Precision , Recall , F-measure , Accuracy
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      7083
           1       1.00      1.00      1.00      6381

    accuracy                           1.00     13464
   macro avg       1.00      1.00      1.00     13464
weighted avg       1.00      1.00      1.00     13464
```

Support Vector Machine						
	Precision	Recall	F1-Score	Accuracy	Confusion Matrix	Error
Fake News Data set -1	1	1	1	1	[[7079 4] [3 6378]]	0.02111
Fake News Data set -2	1	1	1	1	[[1897 3] [3 6423]]	0.026844
Fake News Data set -3	1	0.97	0.99	1	[[611 16] [2 6425]]	0.05324
Fake News Data set -4	1	1	1	1	[[6980 3] [1 6486]]	0.01723

Naive Bayes

```
In [114]:  1 from sklearn.naive_bayes import BernoulliNB

In [115]:  1 BNB = BernoulliNB()

In [116]:  1 BNB.fit(xv_train, y_train)
Out[116]: BernoulliNB()

In [117]:  1 pred_bnb = BNB.predict(xv_test)

In [118]:  1 BNB.score(xv_test,y_test)
Out[118]: 0.942364824717766

In [119]:  1 print(classification_report(y_test, pred_bnb))     #Precision, Recall, F-Measure, Accuracy
              precision    recall  f1-score   support

           0       0.96      0.93      0.94      7083
           1       0.92      0.96      0.94      6381

    accuracy                           0.94     13464
   macro avg       0.94      0.94      0.94     13464
weighted avg       0.94      0.94      0.94     13464
```

Naive Bayes						
	Precision	Recall	F1-Score	Accuracy	Confusion Matrix	Error
Fake News Data set -1	0.96	0.92	0.94	0.94	[[6486 574] [262 6142]]	0.23364
Fake News Data set -2	0.68	0.64	0.66	0.85	[[1208 692] [576 5850]]	0.39024
Fake News Data set -3	0.97	0.37	0.54	0.94	[[234 393] [7 6420]]	0.24342
Fake News Data set -4	0.96	0.93	0.94	0.94	[[6479 584] [268 6227]]	0.2381

Artificial Neural Network

```
In [137]:  1 from sklearn.neural_network import MLPClassifier

In [138]:  1 ann = MLPClassifier()

In [139]:  1 ann.fit(xv_test, y_test)

Out[139]: MLPClassifier()

In [140]:  1 pred_ann = ann.predict(xv_test)

In [141]:  1 ann.score(xv_test,y_test)

Out[141]: 1.0

In [142]:  1 print(classification_report(y_test, pred_ann))      # Precision , Recall , F-measure , Accuracy
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      7083
           1       1.00      1.00      1.00      6381

    accuracy                           1.00     13464
   macro avg       1.00      1.00      1.00     13464
weighted avg       1.00      1.00      1.00     13464
```

Artificial Neural Network						
	Precision	Recall	F1-Score	Accuracy	Confusion Matrix	Error
Fake News Data set -1	1	1	1	1	[[7083 0] [0 6381]]	0.00861
Fake News Data set -2	1	1	1	1	[[1900 0] [1 6425]]	0.01095
Fake News Data set -3	1	1	1	1	[[627 0] [0 6427]]	0
Fake News Data set -4	1	1	1	1	[[6983 0] [0 6487]]	0

Area Under Curve-:

AUC - ROC curve is a performance measurement for the classification problems at various threshold settings. ROC is a probability curve and AUC represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. Higher the AUC, the better the model is at predicting 0 classes as 0 and 1 classes as 1. By analogy, the Higher the AUC, the better the model is at distinguishing between patients with the disease and no disease. The ROC curve is plotted with TPR against the FPR where TPR is on the y-axis and FPR is on the x-axis.

1.Confusion Matrix -:

The confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data. It can only be determined if the true values for test data are known. The matrix itself can be easily understood, but the related terminologies may be confusing. Since it shows the errors in the model performance in the form of a matrix, hence also known as an error matrix.

[TP FP][FN TN]

(TP - Trues Positive , FP - False Positive , TN - True Negative , FN - False Negative)

2.Sensitivity (TPR) = TP/TP+FN

Sensitivity tells us what proportion of the positive class got correctly classified. A simple example would be to determine what proportion of the actual sick people were correctly detected by the model.

3.Specificity = TN/TN+FP Specificity tells us what proportion of the negative class got correctly classified.Taking the same example as in Sensitivity, Specificity would mean determining the proportion of healthy people who

were correctly identified by the model.

4. FPR (1-Specificity) = FP/TN+FP

4.1. FPR- False Positive Rate

FPR tells us what proportion of the negative class got incorrectly classified by the classifier. A higher TNR and a lower FPR is desirable since we want to correctly classify the negative class. Out of these metrics, Sensitivity and Specificity are perhaps the most important and we will see later on how these are used to build an evaluation metric. But before that, let's understand why the probability of prediction is better than predicting the target class directly.

5.TPR-: True Positive Rate **:**

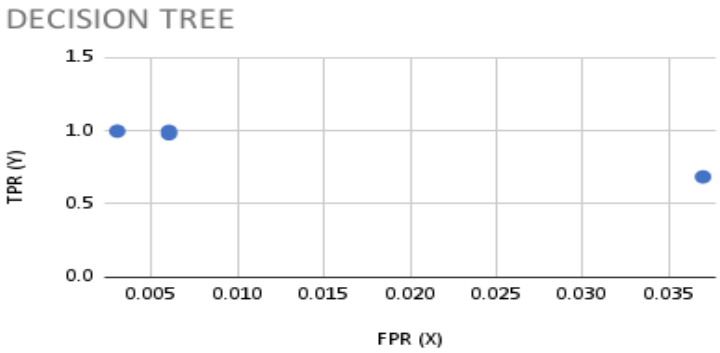
The predicted value as well as the actual value both are positive, i.e., the model correctly predicts the class label to be positive.

6. F1-Score: It is used to measure test accuracy. It is a weighted average of the precision and recall. When F1 score is 1 it's best and on 0 it's worst.

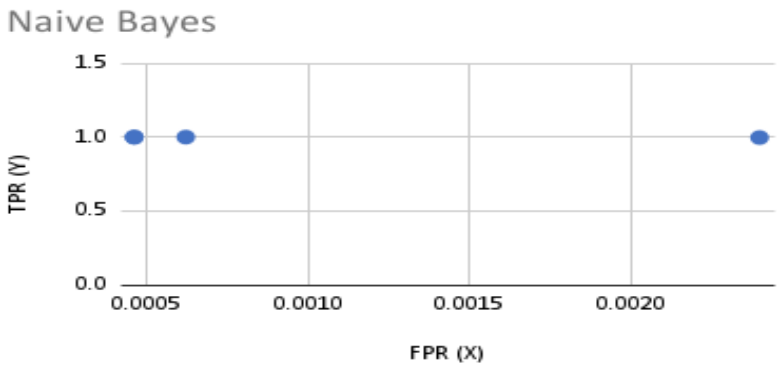
F1 = 2 * (precision * recall) / (precision + recall)

Precision and Recall should always be high.**The relation between Sensitivity, Specificity, FPR, and Threshold.**When we decrease the threshold, we get more positive values thus it increases the sensitivity and decreases the specificity. Similarly, when we increase the threshold, we get more negative values thus we get higher specificity and lower sensitivity.

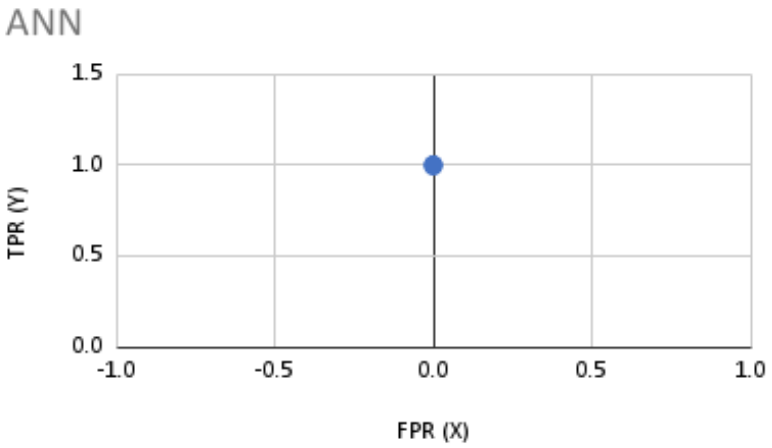
Decision Tree				
	D-1	**D-2**	**D-3**	**D-4**
FPR (X)	0.003	0.006	0.037	0.006
TPR (Y)	0.996	0.978	0.684	0.995



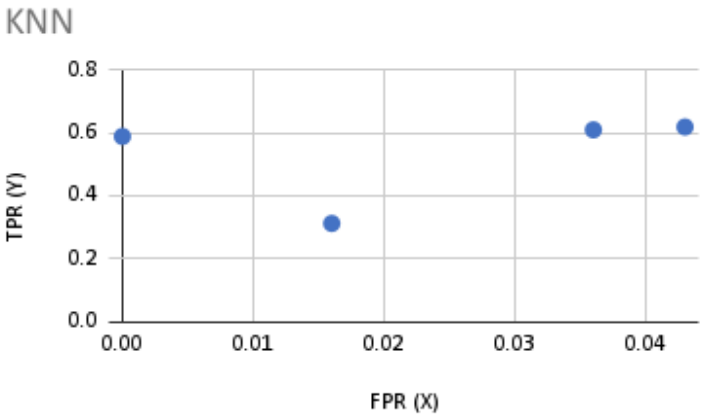
Naive Bayes				
	D-1	**D-2**	**D-3**	**D-4**
FPR (X)	0.085	0.105	0.057	0.075
TPR (Y)	0.961	0.677	0.97	0.96



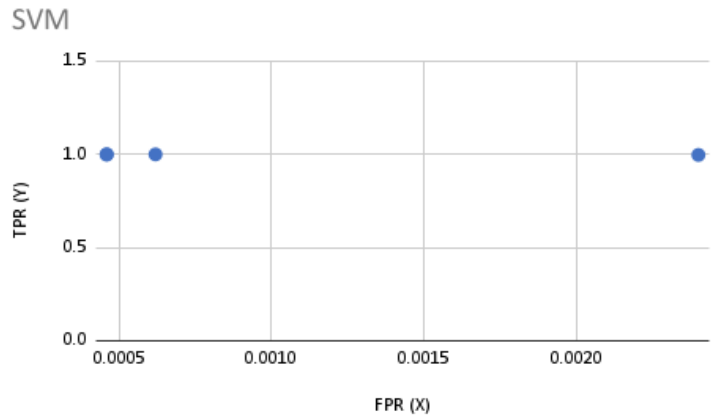
ANN				
	D-1	**D-2**	**D-3**	**D-4**
FPR (X)	0	0	0	0
TPR (Y)	1	0.99	1	1



KNN				
	D-1	**D-2**	**D-3**	**D-4**
FPR (X)	0.043	0.016	0	0.036
TPR (Y)	0.618	0.312	0.588	0.609



SVM				
	D-1	**D-2**	**D-3**	**D-4**
FPR (X)	0.00062	0.00046	0.0024	0.00046
TPR (Y)	0.999	0.998	0.996	0.999



Result Analysis (Using Feature Section)-:

Decision Tree:

Decision Tree						
	Precision	Recall	F1-Score	Accuracy	Confusion Matrix	**Error (root mean)**
Fake News Data set -1	1	1	1	1	[[7029 0] [0 6441]]	0
Fake News Data set -2	0.7	0.66	0.68	0.86	[[1255 637] [531 5903]]	0.3745
Fake News Data set -3	0.59	0.62	0.6	0.93	[[390 237] [273 6154]]	0.2688
Fake News Data set -4	1	1	1	1	[[7023 0] [0 6447]]	0

K Nearest Neighbor:

K Nearest Neighbor						
	Precision	Recall	F1-Score	Accuracy	Confusion Matrix	**Error**
Fake News Data set -1	0.97	0.93	0.95	0.95	[[6533 496] [220 6221]]	**0.2305**
Fake News Data set -2	0.99	0.93	0.96	0.98	[[1756 136] [26 6408]]	**0.1394**
Fake News Data set -3	0.09	1	0.17	0.14	[[626 1] [6078 349]]	**0.92832**
Fake News Data set -4	0.96	0.93	0.95	0.95	[[6550 473] [252 6195]]	**0.2319**

Support Vector Machine:

Support Vector Machine						
	Precision	Recall	F1-Score	Accuracy	Confusion Matrix	**Error**
Fake News Data set -1	1	1	1	1	[[7029 0] [0 6441]]	**0**
Fake News Data set -2	0.98	0.93	0.95	0.98	[[1814 145] [38 6329]]	**0.1482**
Fake News Data set -3	0.99	0.95	0.97	0.99	[[594 33] [4 6423]]	**0.07242**
Fake News Data set -4	1	1	1	1	[[7023 0] [0 6447]]	**0**

Naive Bayes:

Naive Bayes						
	Precision	Recall	F1-Score	Accuracy	Confusion Matrix	Error
Fake News Data set -1	1	0.99	1	1	[[6983 46] [4 6437]]	0.0609
Fake News Data set -2	0.77	0.85	0.81	0.91	[[1604 288] [470 5964]]	0.30172
Fake News Data set -3	0.9	0.33	0.49	0.94	[[210 417 [24 6403	0.25003
Fake News Data set -4	1	0.99	1	1	[[6986 37] [1 6446]]	0.05311

Artificial Neural Network:

Artificial Neural Network						
	Precision	Recall	F1-Score	Accuracy	Confusion Matrix	Error
Fake News Data set -1	1	1	1	1	[[7029 0] [0 6441]]	0
Fake News Data set -2	1	1	1	1	[[1891 1] [0 6434]]	0
Fake News Data set -3	1	1	1	1	[[627 0] [0 6427]]	0
Fake News Data set -4	1	1	1	1	[[7023 0] [0 6447]]	0

Conclusion

Looking at all the results derived after implementing the following classifiers : 1. Decision Trees 2. Naive Bayes 3. KNN 4. ANN 5. SVM on the 4 FakeNews Datasets we have arrived at a conclusion that ANN is the most efficient algorithm as it gives the best precision and accuracy and least error.

SOFTWARE REQUIREMENTS

- os: windows or linux
- python IDE:python 2.7.x and above
- jupyter notebook
- setup tools and pin to be installed for 3.6 and above
- language python

HARDWARE REQUIREMENTS

- RAM:16 GB
- Processor :intel i5 8th Gen and above