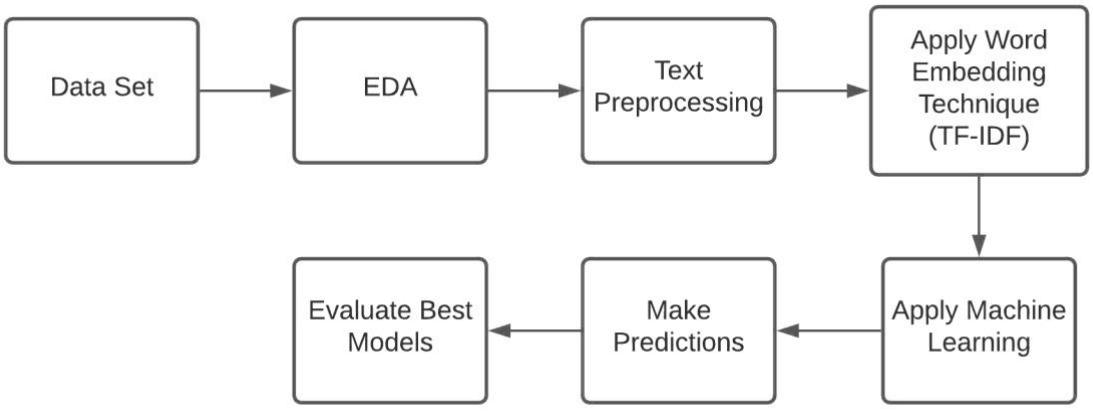
SMS Spam Classification

Introduction:

The number of mobile phone (smartphone) users increases from 1 billion to 3.8 billion in five years. The top three countries using more mobiles are China, India, US. Short Message Service or SMS is a text messaging service available for the last several years. SMS service can be availed without internet also. So, SMS service is available in smartphones and basic mobiles also. Although smart phones bring several apps like WhatsApp for text messaging, this service can be availed with the help of the internet only. But SMS can be availed at any time. So, the traffic for SMS service increasing day by day. A spammer is a person/company which is responsible for unsolicited messages. For their organization benefits or personal benefits, spammers sending a vast number of messages to the users. These messages are called spam messages. Although there are various SMS spam filtering techniques available, still there is a need to handle this problem with advanced techniques. Mobile users may get annoyed because of spam messages. Spam messages can be two types, SMS spam or email spam. The purpose of email spam or SMS spam is the same. Generally, these spam messages are spent by spammers for the promotion of their utilities or business. Sometimes, the users may also undergo financial loss due to these spam messages. Machine Learning is a technology, where machines learn from previous data and made a prediction on future data. Nowadays, machine learning and deep learning can be applied to solve most of the real-world problems in all sectors like health, security, market analysis, etc. There are various techniques available in machine learning like supervised learning, unsupervised, semi supervised learning, etc. In supervised learning, the dataset is having output labels, whereas unsupervised learning deals with datasets with no labels.

Methodology:

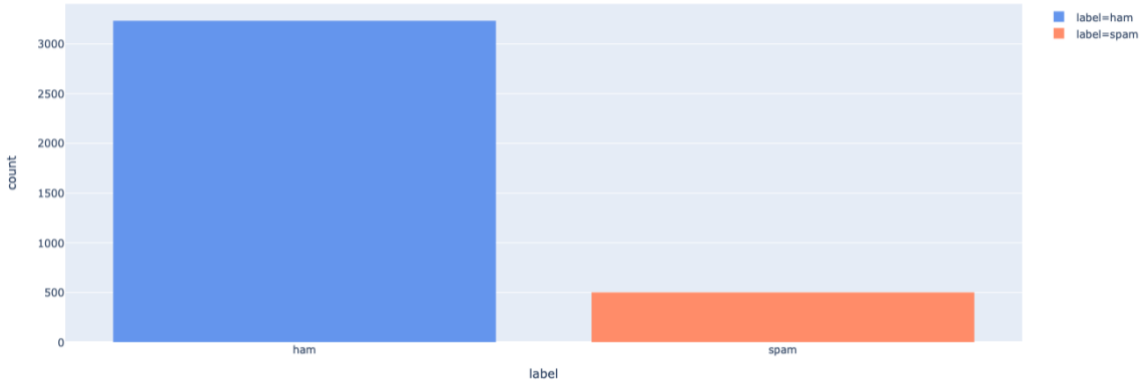


Dataset:

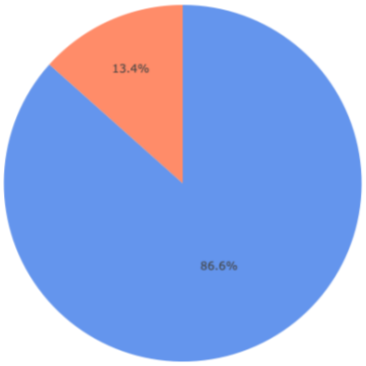
The dataset contains **3733** messages which are labelled as spam and ham, where spam represent the spam message and ham represent the not spam message.

Exploratory Data Analysis:

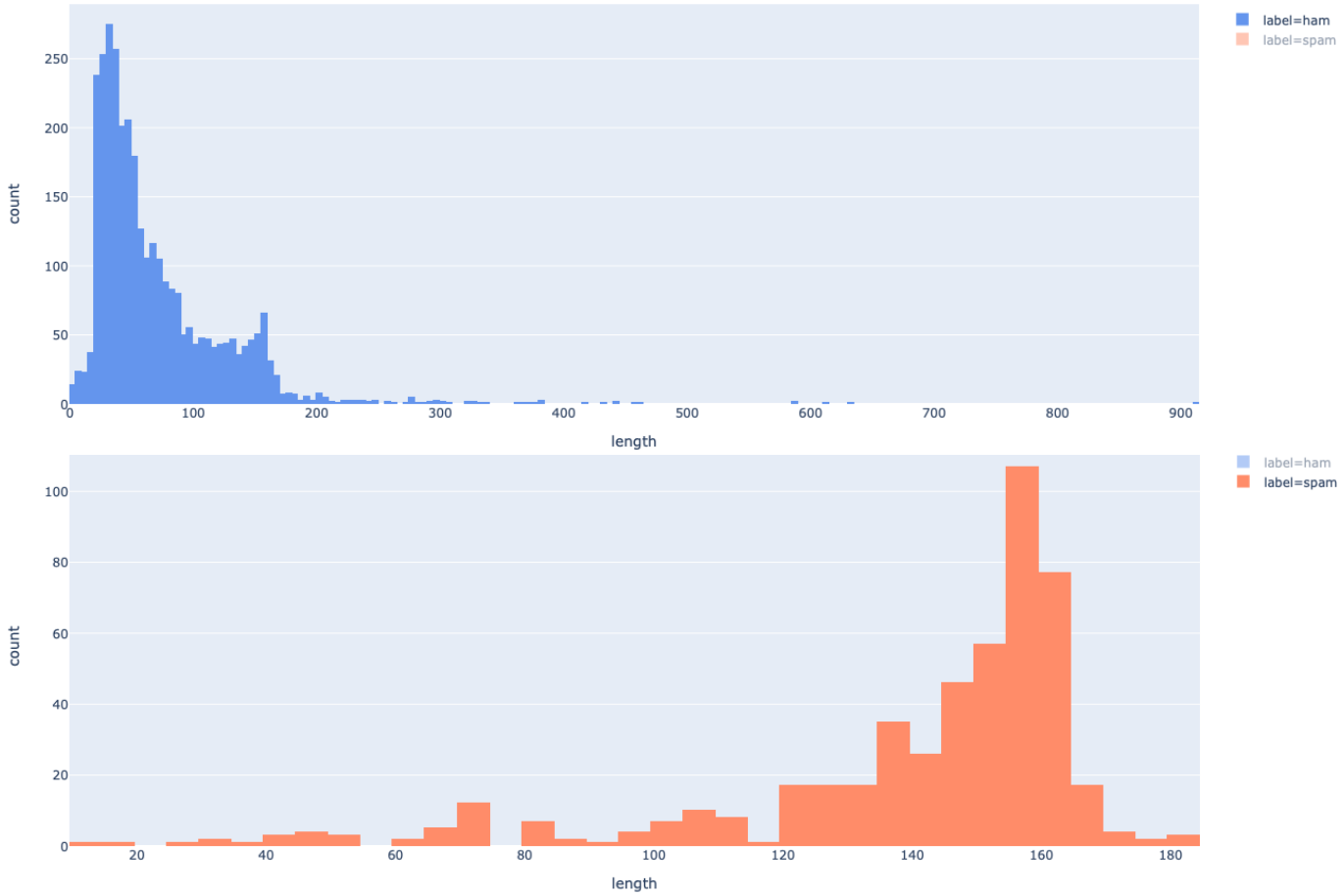
The given data set is being loaded into a python pandas data frame and the distribution of the messages as per the categorized labels are observed using a histogram.



From the above figure, we can see that there is a significant domination of ham messages in the given data set when compared against spam messages.

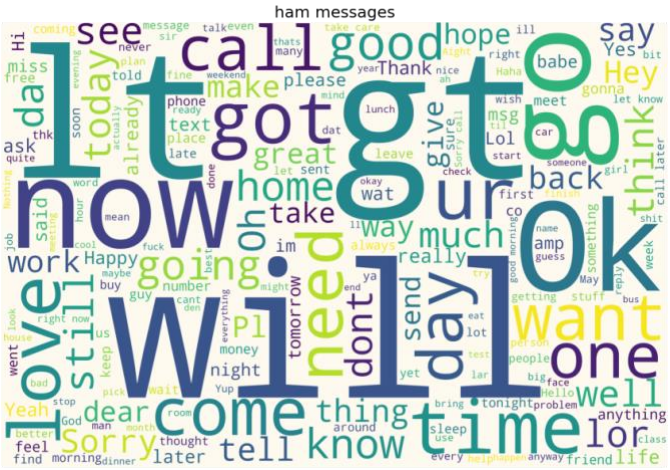


We can see that, there are 3233 ham messages where as there are only 500 spam messages. This contributes to only 13.4% of the total messages in the data set whereas ham messages contribute to 86.6% of the total messages. This clearly shows that the data set is imbalanced.



It can be seen that ham messages are shorter than spam messages as the distribution of ham and spam message lengths are cantered around 30-40 and 155-160 characters, respectively.

Having a view of the most common words used in spams and hams will help us understand the dataset better. A word cloud can give us an idea of what kind of words are dominant in each class.



Text Pre-processing & Word Embedding:

Text pre-processing is a method to clean the text data and make it ready to feed data to the model. Text data contains noise in various forms like emotions, punctuation, text in a different case. When we talk about Human Language then, there are different ways to say the same thing, And this is only the main problem we have to deal with because machines will not understand words, they need numbers so we need to convert text to numbers in an efficient manner. Text data can be represented in vectorized format. Text pre-processing can be done by various NLP (Natural Language Processing) techniques. Machine Learning algorithms work with numbers only. So, there is a need to encode text data into the numeric format. Tokenization is a process of dividing text data into different parts. In-Text pre-processing stop words are removed. stop words are the words that are not useful for analysing the text data. For example, the words like is, was, that are stop words. After removing, stemming can also be applied. Stemming is a process of reducing the word to its stem. For example, the word “playing” can be changed as “play”. After that word embeddings can be done, where the words are changed as vectors of real values. Here TF-IDF Vectorizer technique has been used. **TF-IDF Vectorizer -** TF-IDF is an abbreviation for Term Frequency Inverse Document Frequency. This is very common algorithm to transform text into a meaningful representation of numbers which is used to fit machine algorithm for prediction. TF is number of times the sentence appears. IDF is inverse document frequencies.

TF-IDF = TF(t, d) * IDF(t)

TF(t, d) = Number of times term t appears in a document d

IDF(t) = Inverse document frequency (log (1+n/1 + df(d, t)) + 1)

where n is the number of documents.

Machine Learning techniques:

After converting text into real valued vectors, various machine learning classifiers like naive bayes, random forest decision tree etc have been applied once the data has been split into training and testing sets. Since the data at hand is imbalanced, imblearn's Synthetic Minority Oversampling Technique has been used to up sample the minority data class.

LightGBM: It is a gradient boosting framework that uses tree based learning algorithms. It has the following benefits: Faster training speed and higher efficiency, Lower memory usage, Better accuracy, Support of parallel and GPU learning, Capable of handling large-scale data. Apart from this mode, few other models have also been used for getting to know the best model in terms of evaluation metrics. For this, a custom function has been built to take care of running and outputting the performance evaluation metrics in one go.

Naive Bayes Classification: Naive Bayes classification algorithm is based on bayes theorem. This theorem is based on probability theory.

Logistic Regression: Logistic Regression uses logit function and sigmoid function for classification tasks. The output variable is predicted based on s-shaped curve.

K-Nearest Neighbours: K-NN is a simple but efficient machine learning classifier, which is based on distance calculation. It identifies the k-nearest neighbours and based on the count of neighbour's class; the new data point is classified.

Decision Tree Classification: Decision Tree classifier builds a tree based on which classification can be done. The tree is built recursively until a fixed number of minimum nodes.

Random Forest classification: Random Forest classifier takes the opinion of several decision trees to decide the class of a new datapoints. It is an ensemble approach.

AdaBoost classifier: An AdaBoost classifier is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

Bagging classifier: A Bagging classifier is an ensemble meta-estimator that fits base classifiers each on random subsets of the original dataset and then aggregate their individual predictions (either by voting or by averaging) to form a final prediction.

Extra-trees classifier: This class implements a meta estimator that fits a number of randomized decision trees (a.k.a. extra-trees) on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

Gradient Boosting for classification: GB builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions.

XGBClassifier: XGBoost provides a wrapper class to allow models to be treated like classifiers or regressors in the scikit-learn framework. This means we can use the full scikit-learn library with XGBoost models. The XGBoost model for classification is called XGBClassifier.

Evaluation Metrics:

Precision, Recall, Accuracy are the three measures used for comparing performance evaluation of classifiers.

Accuracy measures how often the classifier makes the correct prediction. It's the ratio of the number of correct predictions to the total number of predictions (the number of test data points).

Precision tells us what proportion of messages we classified as spam, actually were spam. It is a ratio of true positives(words classified as spam, and which are actually spam) to all positives(all words classified as spam, irrespective of whether that was the correct classification).

Recall(sensitivity) tells us what proportion of messages that actually were spam were classified by us as spam. It is a ratio of true positives(words classified as spam, and which are actually spam) to all the words that were actually spam.

Results of Experiments:

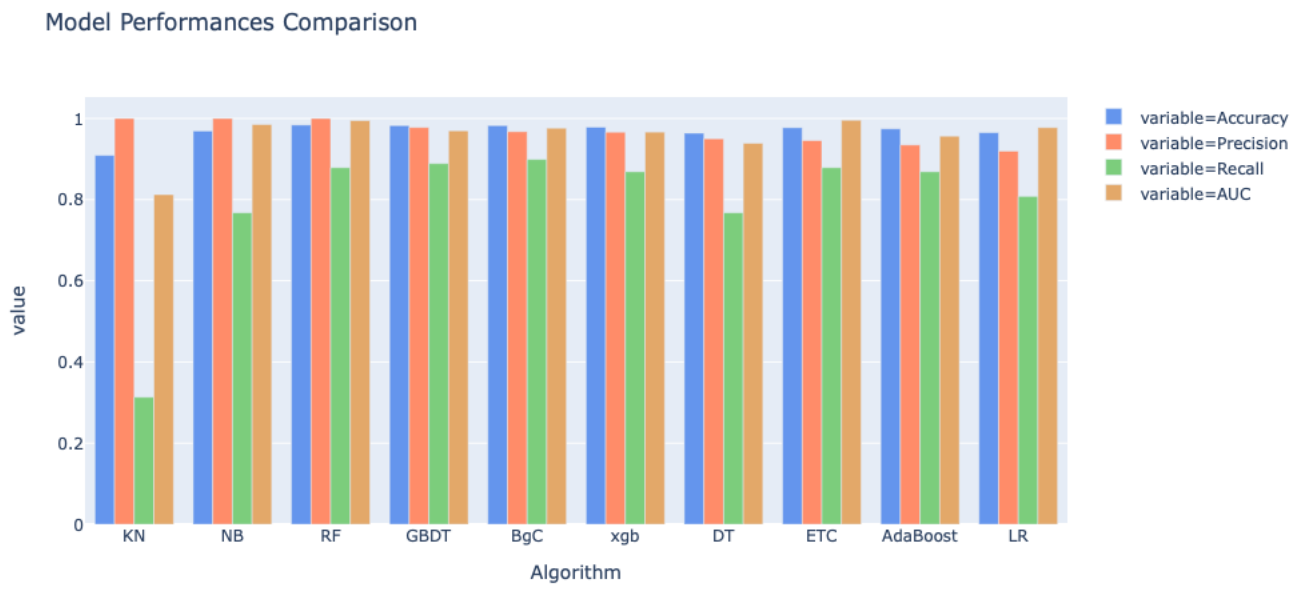
Algorithm	Accuracy	Precision	Recall	AUC
KN	0.908969	1.000000	0.313131	0.812445
NB	0.969210	1.000000	0.767677	0.985269
RF	0.983936	1.000000	0.878788	0.994879
GBDT	0.982597	0.977778	0.888889	0.969713
BgC	0.982597	0.967391	0.898990	0.976197
xgb	0.978581	0.966292	0.868687	0.966556
DT	0.963855	0.950000	0.767677	0.938739
ETC	0.977242	0.945652	0.878788	0.995783
AdaBoost	0.974565	0.934783	0.868687	0.956478
LR	0.965194	0.919540	0.808081	0.977444

Before Up sampling

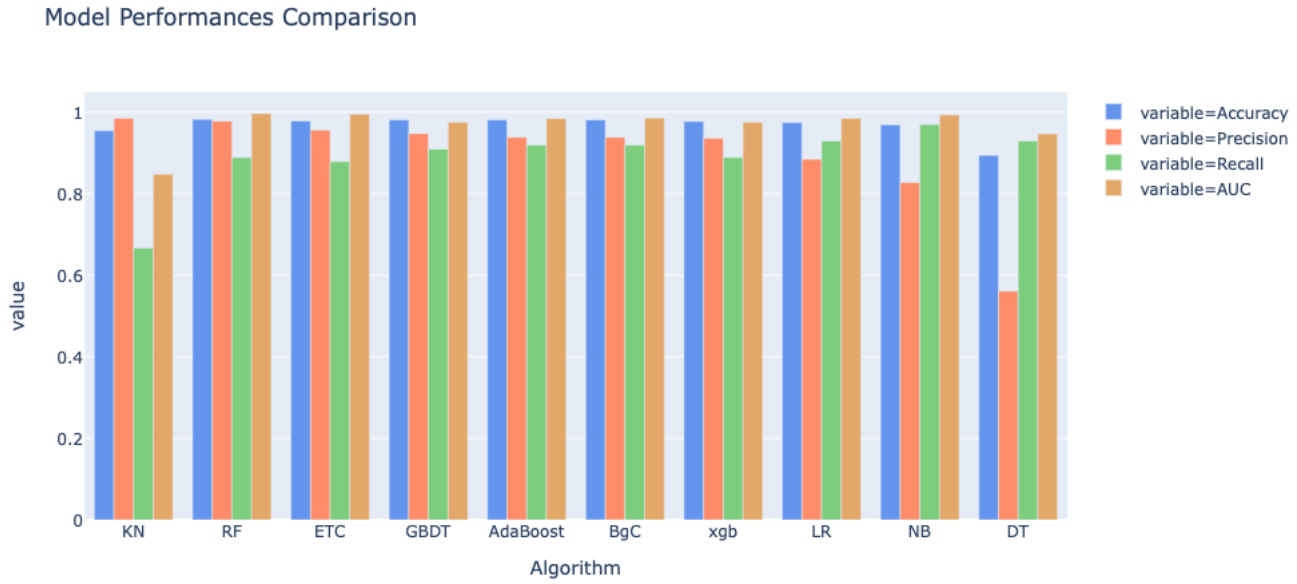
Algorithm	Accuracy	Precision	Recall	AUC
KN	0.954485	0.985075	0.666667	0.847893
RF	0.982597	0.977778	0.888889	0.996586
ETC	0.978581	0.956044	0.878788	0.995511
GBDT	0.981258	0.947368	0.909091	0.975309
AdaBoost	0.981258	0.938144	0.919192	0.984116
BgC	0.981258	0.938144	0.919192	0.985901
xgb	0.977242	0.936170	0.888889	0.975480
LR	0.974565	0.884615	0.929293	0.984661
NB	0.969210	0.827586	0.969697	0.993437
DT	0.894244	0.560976	0.929293	0.946876

After Up sampling

Before Up sampled results:



After Up sampled results:



<u>Final Suggestions:</u>

From the above graphs and metrics, we can see that **Random Forest classifier** gives the best results after taking into the consideration of all the four performance metrics as it gives highest values for all the four metrics.