Project Name – Sarcasm Detection

By T SAI KISHORE

Problem Statement –

This case requires trainees to develop a text classification model to label a news headline as sarcastic or not. This News Headlines dataset for Sarcasm Detection is collected from two news website. We collect real (and non-sarcastic) news headlines from different Post.

1. Introduction

Sentiment analysis is the field of study that analyses people's sentiments, attitudes, and emotions from text. It is one of the most active research areas widely studied in data mining, Web mining, and text mining. Sarcasm is a special kind of sentiment that comprise of words which mean the opposite of what you really want to say, especially in order to wit someone or to be funny.

In the given problem, the dataset comprised of 3 attributes :

1. is sarcastic: 1 if the record is sarcastic otherwise 0
2. Headline: the headline of the news article
3. article_link: link to the original news article. Useful in collecting supplementary data

The task was to develop a text classification model to label a news headline as sarcastic or not. The given problem falls under structured data classification.

Structured Data Classification

Classification can be performed on structured or unstructured data. Classification is a technique where we categorize data into a given number of classes. The main goal of a classification problem is to identify the category/class to which a new data will fall under.

Few of the terminologies encountered in machine learning – classification:

- **Classifier:** An algorithm that maps the input data to a specific category.
- **Classification model:** A classification model tries to draw some conclusion from the input values given for training. It will predict the class labels/categories for the new data.
- **Feature:** A feature is an individual measurable property of a phenomenon being observed.
- **Binary Classification:** Classification task with two possible outcomes. Eg: Gender classification (Male / Female)
- **Multi-class classification:** Classification with more than two classes. In multi class classification each sample is assigned to one and only one target label. Eg: An animal can be cat or dog but not both at the same time
- **Multi-label classification:** Classification task where each sample is mapped to a set of target labels (more than one class). Eg: A news article can be about sports, a person, and location at the same time.

2. Exploratory Data Analysis (EDA)

Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations. During the EDA process we can check for null count, mean, median, mode, datatypes of each columns and also plot graphs for analysing various parameters in the dataset.

Data visualization involves exploring data through visual representations. It is closely associated with data analysis, which uses code to explore the patterns and connections in a data set. A data set can be made up of a small list of numbers that fits in one line of code or it can be many gigabytes of data. One of the most popular tools is Matplotlib, a mathematical plotting library.

In the data set EDA was performed on is_sarcastic column to find the distribution of sarcastic and non sarcastic headline

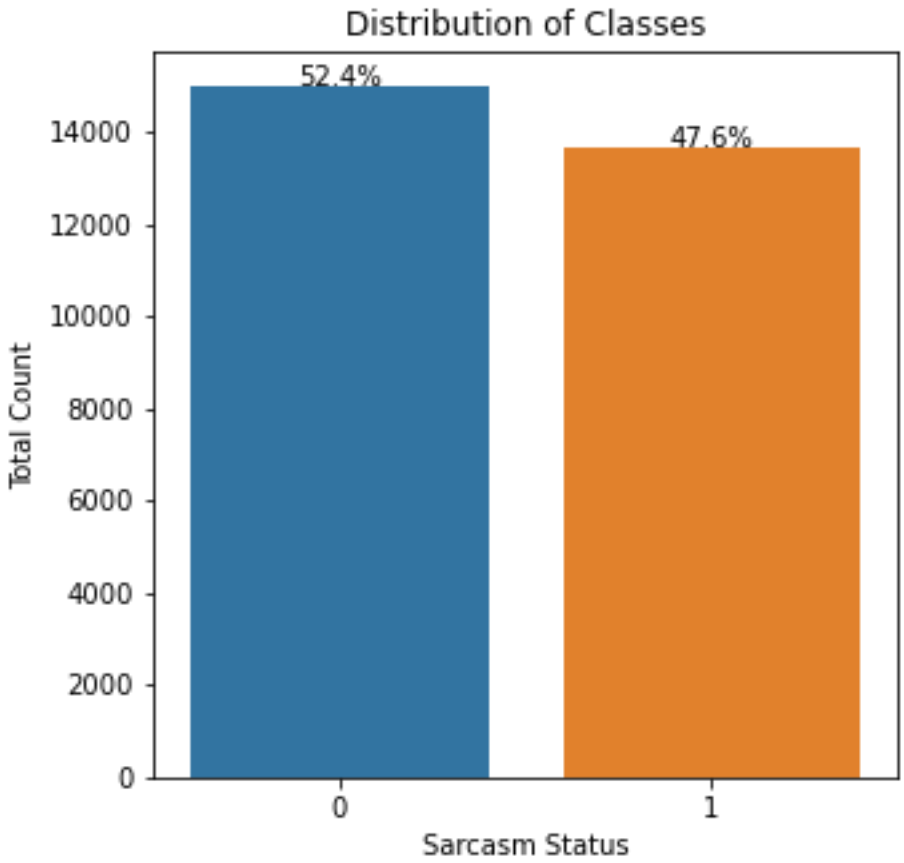


Fig 1

In the dataset, EDA was performed on the article_link column to check the number of posts of the news news headline.

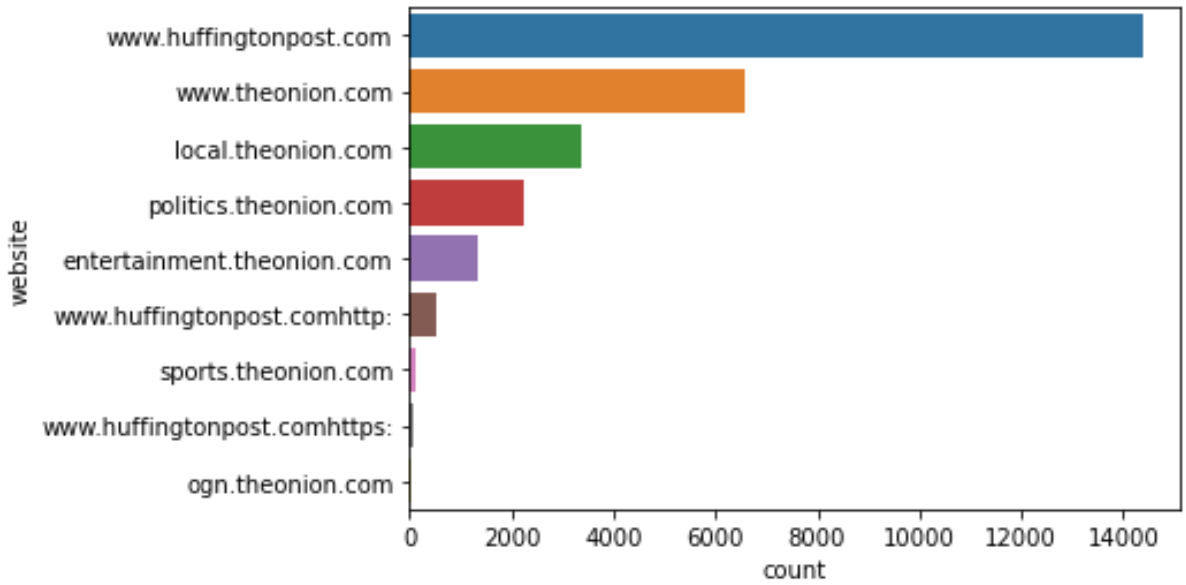


Fig 2

From the above horizontal bar graph(Fig 2) we can see the number of articles that were taken from different posts .

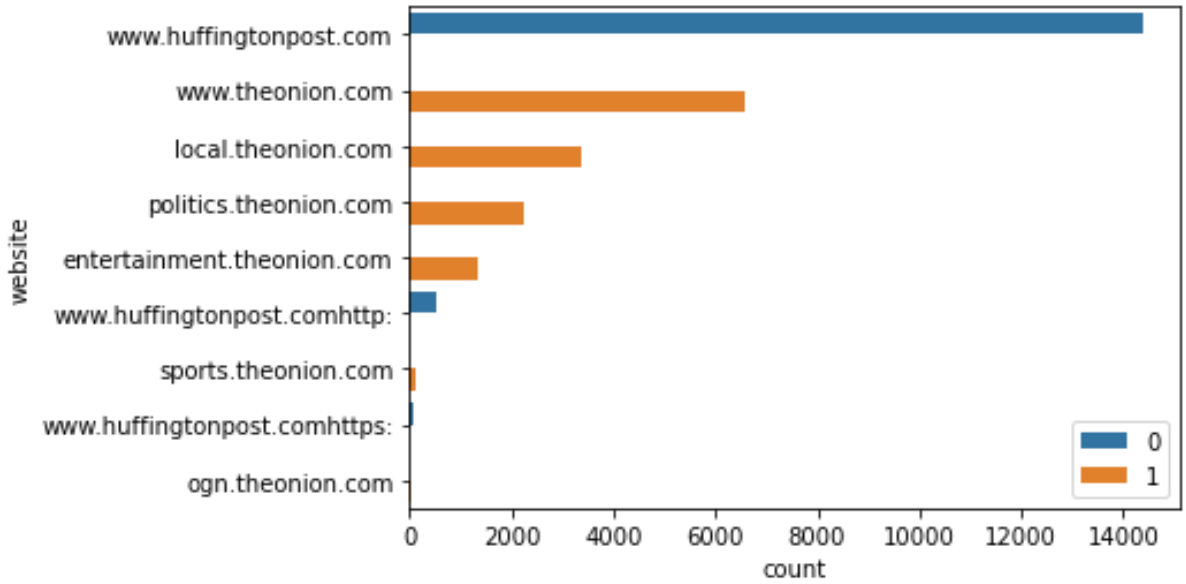


Fig 3

The above graph(Fig 3) represents the number of sarcastic and non sarcastic headlines belonging to a category of articles. It is evident that huffingtonpost has most number of non sarcastic news headline.

2.1 Wordcloud

Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance. Significant textual data points can be highlighted using a word cloud.

Fig 4

The above wordcloud (Fig 4) represents the most frequent words occurring in headline column for both sarcastic and non sarcastic headline.

N Grams

An N-Gram is a sequence of N tokens of words. Given a sequence of words, an N-gram model predicts the most probable word that might follow this sequence. It's a probabilistic model that's trained on a corpus of text. Such a model is useful in many NLP applications including speech recognition, machine translation and predictive text input. An N-gram model is built by counting how often word sequences occur in corpus text and then estimating the probabilities. Since a simple N-gram model has limitations, improvements are often made via smoothing, interpolation and back off. An N-gram model is one type of a **Language Model (LM)**, which is about finding the probability distribution over word sequences. An n gram can be unigram, bigram, trigram or n-gram. A model that simply relies on how often a word occurs without looking at previous words is called **unigram**. If a model considers only the previous word to predict the current word, then it's called **bigram**. If two previous words are considered, then it's a **trigram** model. N-gram models are usually at word level. It's also been used at character level to do stemming, that is, separate the root word from the suffix. In general, many NLP applications benefit from N-gram models including part-of-speech tagging, natural language generation, word similarity, sentiment extraction and predictive text input.

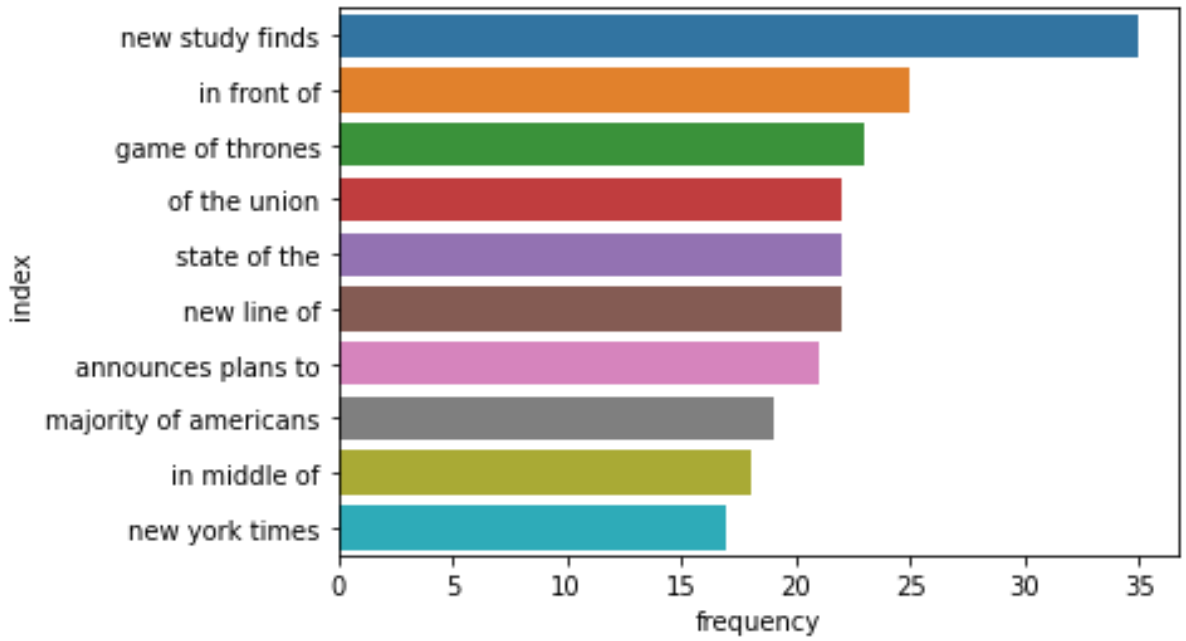


Fig 5

The above bar graph(Fig 5) represents the graphical representation(frequency of most occurring trigram words) being implemented on the headline column.

3. Tokenization Lemmatization Stemming

NLP is a field of computer science that focuses on the interaction between computers and humans. NLP techniques are used to analyze text, providing a way for computers to understand human language. A few examples of NLP applications include automatic summarization, topic segmentation, and sentiment analysis. For this nltk library is used. **NLTK** stands for **Natural Language Toolkit**. This is a suite of libraries and programs for symbolic and statistical NLP for English. Using nltk library, we can tokenize, stem, lemmatize the text.

Tokenization is splitting a phrase, sentence, paragraph, or an entire text document into smaller units, such as individual words or terms. The tokens could be words, numbers or punctuation marks. This is important because the meaning of the text could easily be interpreted by analysing the words present in the text.

Stemming is a technique used to extract the base form of the words by removing affixes from them. It is just like cutting down the branches of a tree to its stems. For example, the stem of the words eating, eats, eaten is eat.
There are various stemming algorithms:

1. **Porter's stemmer algorithm**: It is based on the idea that the suffixes in the English language are made up of a combination of smaller and simpler suffixes. This stemmer is known for its speed and simplicity. The main applications of Porter Stemmer include data mining and Information retrieval. However, its applications are only limited to English words. Also, the group of stems is mapped on to the same stem and the output stem is not necessarily a meaningful word.
2. **Snowball stemmer**: When compared to the Porter Stemmer, the Snowball Stemmer can map non-English words too. Since it supports other languages the Snowball Stemmers can be called a multi-lingual stemmer. The Snowball stemmers is imported from the nltk package. This stemmer is based on a programming language called 'Snowball' that processes small strings and is the most widely used stemmer. The Snowball stemmer is way more aggressive than Porter Stemmer and is also referred to as Porter2 Stemmer. Because of the improvements added when compared to the Porter Stemmer, the Snowball stemmer is having greater computational speed.
3. **Lancaster stemmer**: The Lancaster stemmers are more aggressive and dynamic compared to the other two stemmers. The stemmer is really faster, but the algorithm is really confusing when dealing with small words. But they are not as efficient as Snowball Stemmers. The Lancaster stemmers save the rules externally and basically uses an iterative algorithm.

4. **Regexp stemmer (Regular expression)**: It takes a single regular expression and removes any prefix or suffix that matches the expression

Lemmatization is the process of grouping together the different inflected forms of a word so they can be analysed as a single item. Lemmatization is similar to stemming but it brings context to the words. So it links words with similar meaning to one word.

Eg : rocks : rock better : good corpora : corpus

3.1 Finding most frequent Sarcastic and non Sarcastic words

For finding the most frequent sarcastic and non_sarcastic words in the headline column, the sarcastic and non sarcastic headlines need to be cleaned. They are first loaded into different dataframes on whether they are sarcastic or not and then tokenized, loaded into a single data frame and the stopwords are removed from the list. For this nltk library is used.

```
Length of original Sarcasm list: 140546 words
Length of Sarcasm list after stopwords removal:
105627 words
==========================================================
==========================================
Length of original non sarcastic list: 147128 words
Length of non sarcastic list after stopwords removal:
103525 words
```

This is the output of the number of words in the list before and after removal of stopwords

4.Finding The Topic Of The News Article Headline

In the given dataset there was no attribute that mention of the topic that the article belonged to. In order to predict to which topic a headline was referring to, Latent Dirichlet Allocation (LDA) model is implemented. It is important to note that topic modelling is not the same as topic classification. Topic classification is a supervised learning approach in which a model is trained using manually annotated data with predefined topics. After training, the model accurately classifies unseen texts according to their topics. On the other hand, topic modelling is an unsupervised learning approach in which the model identifies the topics by detecting the patterns such as words clusters and frequencies.

LDA assumes that documents are composed of words that help determine the topics and maps documents to a list of topics by assigning each word in the document to different topics.

LDA has three hyper parameters;
1) document-topic density factor 'α'
2) topic-word density factor 'β'
3) the number of topics 'K' to be considered.

The **'α' hyper parameter** controls the number of topics expected in the document. Low value of 'α' is used to imply that fewer number of topics in the mix is expected and a higher value implies that one would expect the documents to have higher number topics in the mix.

The **'β' hyper parameter** controls the distribution of words per topic. At lower values of 'β', the topics will likely have fewer words and at higher values topics will likely have more words.

The **'K' hyper parameter** specifies the number of topics expected in the corpus of documents. Choosing a value for K is generally based on domain knowledge. An alternate way is to train different LDA models with different numbers of K values and compute the Coherence Score. Choose the value of K for which the coherence score is highest. Topic coherence score is a measure of how good a topic model is in generating coherent topics. A coherent topic should be semantically interpretable and not of statistical inference. A higher coherence score indicates a better topic model

4.1 Data pre-processing for LDA

The typical pre-processing steps before performing LDA are
1) tokenization,
2) punctuation and special character removal,
3) stop word removal and
4) lemmatized.

Different topics that can be identified are

```
THE TOP 15 WORDS FOR TOPIC #1
['business', 'apple', 'christmas', 'best', 'bernie', 'stop', 'says', 'c
ollege', 'life', 'sanders', 'women', 'love', 'boy', 'really', 'book', '
sex', 'high', 'home', 'teen', 'year', 'million', 'report', 'school', '1
0', 'new']
```

From topic 1 we can relate this topic to students and college life, Christmas

```
THE TOP 15 WORDS FOR TOPIC #2
['study', 'friend', 'work', 'real', 'going', 'mom', 'new', 'video', 're
port', 'friends', 'good', 'life', 'nation', 'really', 'years', 'office'
, 'thing', 'guy', 'like', 'way', 'time', 'woman', 'just', 'area', 'man'
]
```

From topic 2 we can relate this topic to wishing new year to friends and family through video

```
THE TOP 15 WORDS FOR TOPIC #3
['black', 'political', 'clinton', 'end', 'says', 'court', 'people', 'am
erica', 'rights', 'john', 'campaign', 'donald', 'time', 'ryan', 'care',
'gop', 'bush', 'white', 'gay', 'american', 'paul', 'house', 'health', '
women', 'trump']
```

From topic 3 we can relate this topic to politics

```
THE TOP 15 WORDS FOR TOPIC #5
['isis', 'attack', 'cruz', 'korea', 'violence', 'like', 'war', 'gop', '
white', 'says', 'gun', 'americans', 'world', 'watch', 'pope', 'presiden
t', 'anti', 'north', 'state', 'donald', 'hillary', 'obama', 'clinton',
'new', 'trump']
```

From topic 5 we can relate this topic to violence

5.Building A Machine Learning Model

The most important phase in model building in a text classification model is tokenization stemming and removing stop words form the list for better accuracy and prediction. In this model Punkt tokenizer. Punkt Sentence Tokenizer divides a text into a list of sentences by using an unsupervised algorithm to build a model for abbreviation words, collocations, and words that start sentences. It must be trained on a large collection of plaintext in the target language before it can be used. The NLTK data package includes a pre-trained Punkt tokenizer for English.

During the process we have loaded TfidfVectorizer. It Converts a collection of raw documents to a matrix of TF-IDF features. 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.

Term Frequency (TF)
This measures the frequency of a word in a document. This highly depends on the length of the document and the generality of word
Document Frequency
This measures the importance of document in whole set of corpus, this is very similar to TF. The only difference is that TF is frequency counter for a term t in document d, whereas DF is the count of occurrences of term t in the document set N. In other words, DF is the number of documents in which the word is present. We consider one occurrence if the term consists in the document at least once, we do not need to know the number of times the term is present. To keep this also in a range, we normalize by dividing with the total number of documents.
Inverse Document Frequency
IDF is the inverse of the document frequency which measures the informativeness of term t. When we calculate IDF, it will be very low for the most occurring words such as stop words

The next step is vectorization. In this technique arrays are implemented instead of loops which helps in minimizing the running time and execute code effetely.

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer("english")

features = vectorizer.fit_transform(text_feature)
```

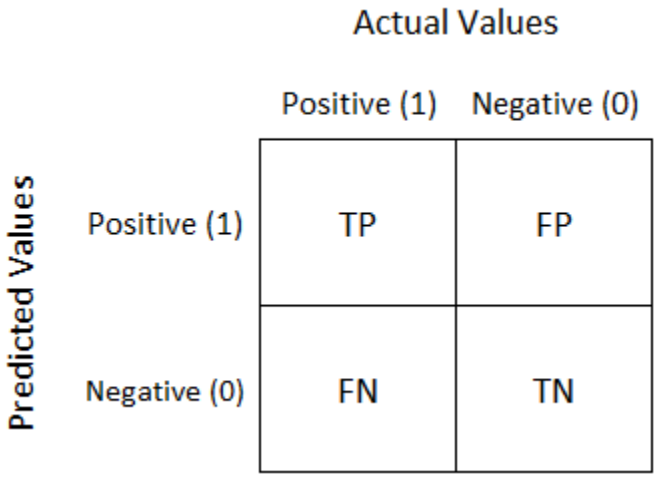
once the data is vectorised we have to split the data into test and train datasets for model training testing and evaluation . once the training and testing data are ready we can start implementing different machine learning algorithms.

6.Model Evaluation and Analysis

In this process we first predict the accuracy and then construct the confusion matrix and classification report.

A **confusion matrix** is a technique for summarizing the performance of a classification algorithm. Classification accuracy alone can be misleading if you have an unequal number of observations in each class or if you have more than two classes in your dataset. Calculating a confusion matrix can give you a better idea of what your classification model is getting right and what types of errors it is making.
There are four ways to check if the predictions are right or wrong:

1. **TN / True Negative**: the case was negative and predicted negative
2. **TP / True Positive**: the case was positive and predicted positive
3. **FN / False Negative**: the case was positive but predicted negative
4. **FP / False Positive**: the case was negative but predicted positive



The **classification report** visualizer displays the precision, recall, F1, and support scores for the model.

Precision — Precision is the ability of a classifier not to label an instance positive that is actually negative. For each class, it is defined as the ratio of true positives to the sum of a true positive and false positive.
Precision: - Accuracy of positive predictions.
Precision = TP/ (TP + FP

Recall is the ability of a classifier to find all positive instances. For each class it is defined as the ratio of true positives to the sum of true positives and false negatives.
Recall: - Fraction of positives that were correctly identified.
Recall = TP/ (TP+FN)

The **F1 score** is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0. F1 scores are lower than accuracy measures as they embed precision and recall into their computation. As a rule of thumb, the weighted average of F1 should be used to compare classifier models, not global accuracy.
F1 Score = 2*(Recall * Precision) / (Recall + Precision)

Support is the number of actual occurrences of the class in the specified dataset. Imbalanced support in the training data may indicate structural weaknesses in the reported scores of the classifier and could indicate the need for stratified sampling or rebalancing. Support doesn't change between models but instead diagnoses the evaluation process.

AUC ROC

The **Receiver Operator Characteristic (ROC)** curve is an evaluation metric for binary classification problems. It is a probability curve that plots the **TPR** against **FPR** at various threshold values and essentially **separates the 'signal' from the 'noise'**. The **Area Under the Curve (AUC)** is the measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve. The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes.

- When AUC = 1, then the classifier is able to perfectly distinguish between all the Positive and the Negative class points correctly. If, however, the AUC had been 0, then the classifier would be predicting all Negatives as Positives, and all Positives as Negatives.
- When 0.5<AUC<1, there is a high chance that the classifier will be able to distinguish the positive class values from the negative class values. This is so because the classifier is able to detect more numbers of True positives and True negatives than False negatives and False positives.
- When AUC=0.5, then the classifier is not able to distinguish between Positive and Negative class points. Meaning either the classifier is predicting random class or constant class for all the data points.

In a ROC curve, a higher X-axis value indicates a higher number of False positives than True negatives. While a higher Y-axis value indicates a higher number of True positives than False negatives. So, the choice of the threshold depends on the ability to balance between False positives and False negatives.

7.Different Machine Learning Algorithms Implementation

7.1 LinearSVC

Linear Support Vector Classifier(Linear SVC). Linear SVM is the fast machine learning algorithm for solving classification problems for large data sets that implements an original proprietary version of a cutting plane algorithm for designing a linear support vector machine.
Upon implementing this model, it is found that this model has an accuracy score of 79%.

```
Test accuracy is 0.790356394129979
```

Confusion matrix :

```
[[3028  683]
 [ 817 2627]]
```

Classification Report :

	precision	recall	f1-score	support
0	0.79	0.82	0.80	3711
1	0.79	0.76	0.78	3444
accuracy			0.79	7155
macro avg	0.79	0.79	0.79	7155
weighted avg	0.79	0.79	0.79	7155

```
AUC score of LinearSV :  0.7934
```

7.2 LogisticRegression

Logistic regression is a fundamental classification technique. It belongs to the group of linear classifiers and is similar to polynomial and linear regression.
Logistic regression is fast and relatively uncomplicated, and it's convenient to interpret the results. Although it's essentially a method for binary classification, it can also be applied to multiclass problems.
Upon implementing this model the accuracy score is 78.71%

```
Test accuracy is 0.787141858839972
```

Confusion matrix :

```
[[3144  567]
 [ 956 2488]]
```

Classification Report :

	precision	recall	f1-score	support
0	0.77	0.85	0.81	3711
1	0.81	0.72	0.77	3444
accuracy			0.79	7155
macro avg	0.79	0.78	0.79	7155
weighted avg	0.79	0.79	0.79	7155

AUC score of Logistic Regression : 0.7849

7.3 MultinomialNB (Naive Bayes classifier for multinomial models)

The multinomial Naive Bayes classifier is suitable for classification with discrete features (e.g., word counts for text classification). The multinomial distribution normally requires integer feature counts. However, in practice, fractional counts such as tf-idf may also work.

By implementing MultinomialNB the accuracy score is 78.36%

Test accuracy is 0.7836477987421384
Confusion matrix :
[[3036 675]
[873 2571]]
Classification Report :

	precision	recall	f1-score	support
0	0.78	0.82	0.80	3711
1	0.79	0.75	0.77	3444
accuracy			0.78	7155
macro avg	0.78	0.78	0.78	7155
weighted avg	0.78	0.78	0.78	7155

AUC score of MultinomialNB : 0.7889

7.4 RandomForestClassifier

Random forests is a supervised learning algorithm. It can be used both for classification and regression. It is also the most flexible and easy to use algorithm. A forest is comprised of trees. It is said that the more trees it has, the more robust a forest is. Random forests creates decision trees on randomly selected data samples, gets prediction from each tree and selects the best solution by means of voting. It also provides a pretty good indicator of the feature importance.

Upon implementing this algorithm the accuracy score is 75.54%

```
accuracy is 0.7554
```

Confusion matrix :

```
[[3187  524]
 [1226 2218]]
```

Classification Report :

	precision	recall	f1-score	support
0	0.72	0.86	0.78	3711
1	0.81	0.64	0.72	3444
accuracy			0.76	7155
macro avg	0.77	0.75	0.75	7155
weighted avg	0.76	0.76	0.75	7155

```
AUC score of Random Forest Classifier :  0.7525
```

7.5 SGDClassifier

Stochastic Gradient Descent (SGD) is a simple yet efficient optimization algorithm used to find the values of parameters/coefficients of functions that minimize a cost function. In other words, it is used for discriminative learning of linear classifiers under convex loss functions such as SVM and Logistic regression. It can be successfully applied to large-scale datasets because the update to the coefficients is performed for each training instance, rather than at the end of instances. Stochastic Gradient Descent (SGD) classifier basically implements a plain SGD learning routine supporting various loss functions and penalties for classification.

Upon implementing the algorithm the accuracy is 78.95%

```
Test accuracy is 0.7895178197064989
```

Confusion matrix :

```
[[3154  557]
 [ 949 2495]]
```

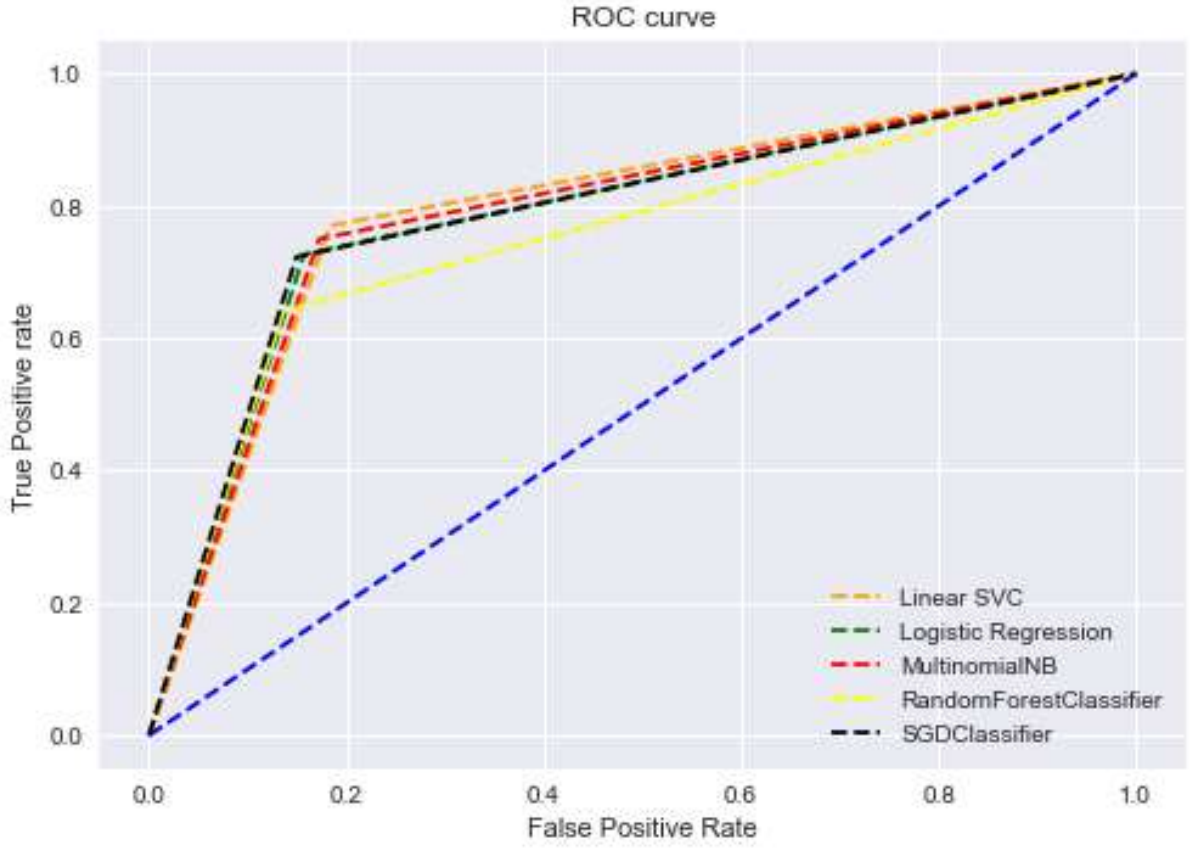
Classification Report:

	precision	recall	f1-score	support
0	0.77	0.85	0.81	3711
1	0.82	0.72	0.77	3444
accuracy			0.79	7155
macro avg	0.79	0.79	0.79	7155
weighted avg	0.79	0.79	0.79	7155

```
AUC score of SGDClassifier :  0.7858
```

Accuracy scores of each algorithm implemented

Algorithm	Accuracy	AUC
LinearSVC	79.05%	79.34%
Logistic Regression	78.71%	78.49%
MultinomialNB	78.36%	78.89%
Random Forest Classifier	75.54%	75.25%
SGDClassifier	78.95%	78.58%



The above graph depicts the ROC curve for each algorithm implemented.

Based on the accuracy score, and analysing confusion matrix, classification report and AUC-ROC report, it is found that Linear SVC algorithm performs better than other algorithms.

Instructions to run a python code

The following code is saved as sarcasm_headline.ipynb file which the file can run directly through jupyter notebook.

1. When we open the anaconda navigator we get the jupyter notebook application.
2. We need to open the application. It is a web interface application which can open in chrome .Now select the required .ipynb file from the stored location.
3. Once we open the required file, the set of codes get displayed on the jupyter notebook through windows.
4. Run the file and check the output.

We can also run the file through Dos prompt.

1. Open the cmd prompt
2. Choose the location where the file is stored and direct it through the cmd prompt
3. Once the location (where the file is located in the folder) is set, type Python *filename.*py

While running the file in cmd promt, if it shows to load the specific libraries we can load them through pip install command.