News Article Dataset

Introduction:-

The News Article dataset has 2015 to 2021 news data with two column dates and headlines.It contains data for different company basically the headlines of the news

In this data we are going to perform sentiment analysis and get the result that whether the stock price will increase or decrease by just headlines of the news.

Text preprocessing for sentiment analysis:

To perform sentiment analysis on news headlines, the data must be prepped in advance. The data is loaded and viewed, then, we prepped the data, by converting everything into desired datatypes say dates into date format from string. After this step, we got our dates and respective news headlines as a table. Then Punctuations are removed.
Then contractions were handled, contractions such as i'll ,i'd etcetera were converted to proper english words i will, i would.
After this, the text is converted into lower case, so it will be easy for us to proceed further.Then stopwords were removed. Then Lemmatization and Stemming has been performed on the data. Lemmatization removes the grammar tense and transforms each word into its original form. Another way of converting words to its original form is called stemming. While stemming takes the linguistic root of a word, lemmatization is taking a word into its original lemma.
Atlast, the data was tokenized, then a word cloud was made, to display the words with high frequency.
Summarizing text preprocessing:

1. Imported , viewed the data.
2. Converting everything to desired datatypes.
3. Removal of punctuations.
4. Contractions were handled.
5. Lowercase conversion.
6. Stopwords removal.
7. Lemmatization and stemming.
8. Tokenization.
9. Wordcloud

Sentiment Analysis:-

Sentiment Analysis or Opinion Mining is a way of finding out the polarity or strength of the opinion (positive or negative) that is expressed in written text, in the case of this project – stock news articles. According to the principle of document level sentiment analysis, each individual document is tagged with its respective polarity. This is generally done by finding the polarities of each individual word/phrase and sentences and combining them to predict the polarity of the whole document.

Modelling

- We have applied Machine Learning techniques such as logistic regression,Naïve bayes model to predict the target variable.

* We have quantified the sentiment of news headlines with a positive or negative value, called **polarity**.For sentiment analysis calculation of overall polarity of headline of each date is required ,for which sentiment function of textblob is used. The overall sentiment is inferred/labelled as positive, neutral or negative based on the sign of the polarity score. * The **compound score** is the sum of positive, negative & neutral scores which is then normalized between -1(most extreme negative) and +1 (most extreme positive).The more Compound score closer to **+1**, the **higher** the positivity of the text. * **VADER** is used to quantify how much of ***positive or negative emotion*** the text has and also the intensity of emotion. Vader SentimentIntensityAnalyzer is used in this program to calculate the news headline compound value for a given day.

Step 1) Defining the model

In this step, we generate our model-fitting our dataset in the MultinomialNB. We will use one of the Naive Bayes (NB) classifier for defining the model. Specifically, we will use MultinomialNB classifier. We used Naïve Bayes model because it works particularly well with text classification and spam filtering. Advantages of working with NB algorithm are:

- Requires a small amount of training data to learn the parameters
- Can be trained relatively fast compared to sophisticated models

Given the dependent feature vector ***(x1, …, xn)*** and the class ***Ck***. Bayes' theorem is stated mathematically as the following relationship:

$$P(C_k \mid x_1, \ldots, x_n) = \frac{P(C_k)P(x_1, \ldots, x_n \mid C_k)}{P(x_1, \ldots, x_n)}$$

According to the "naive" conditional independence assumptions, for the given class ***Ck*** each feature of vector ***xi*** is conditionally independent of every other feature ***xj*** for ***i≠j***..

$$P(x_i \mid C_k, x_1, \ldots, x_n) = P(x_i \mid C_k)$$

Thus, the relation can be simplified to

$$P(C_k \mid x_1, \ldots, x_n) = \frac{P(C_k)\prod_{i=1}^{n} P(x_i \mid C_k)}{P(x_1, \ldots, x_n)}$$

Since $P(x_1, \ldots, x_n)$ is constant, if the values of the feature variables are known, the following classification rule can be used:

$$P(C_k \mid x_1, \ldots, x_n) \propto P(C_k) \prod_{i=1}^{n} P(x_i \mid C_k)$$

$$\Downarrow$$

$$\hat{y} = \arg\max_k P(C_k) \prod_{i=1}^{n} P(x_i \mid C_k)$$

To avoid underflow, log probabilities can be used.

$$\hat{y} = \arg\max_k \left(\ln P(C_k) + \sum_{i=1}^{n} \ln P(x_i \mid C_k) \right)$$

The variety of naive Bayes classifiers primarily differs between each other by the assumptions they make regarding the distribution of $P(x_i \mid C_k)$, while $P(C_k)$ is usually defined as the relative frequency of class C_k in the training dataset.

Step 2) Splitting the Dataset

We do the train/test split before the CountVectorizer to properly simulate the real world where our future data contains words we have not seen before.Split then vectorize is considered the correct way.

Step 3) Applying tf vectorizer (count vectorizer)

If we want to use text in machine learning algorithms, we'll have to convert them to a numerical representation. One of the methods is called bag-of-words approach. The bag of words model ignores grammar and order of words. Here, we use

CountVectorizer (another term of TfVectorizer).it will basically convert these sentences into vectors. That is what bag of words means.

Now we have the training and testing data. We should start the analysis.

Step 4) Applying Naive Bayes

Training Naive Bayes classifier with train data.Since we are using sklearn's modules and classes we just need to import the precompiled classes.we generate our model-fitting our dataset in the MultinomialNB.Naive Bayes using bag of words (unigrams of words) and character level n-grams. N-gram is a tokenization process to separate words based on the type of token used. In this study mainly bigram was used. Bigram is a n-word solution in a review sentence with n = 2. ngram_range=(2,2)

Step 5) Evaluation

The performance evaluation of the Naive Bayes algorithm in this study was carried out by calculating the value of Precision, Recall, Accuracy and Error Rate by using confusion matrix. Here we are applying the classification report, confusion matrix, and accuracy score to check the accuracy of our model.Cross-validation is a statistical method used to estimate the skill of machine learning models.

Step 5) Improving the model and handling unbalanced data

As is known to us, Naive Bayes algorithm is a simple and efficient categorization algorithm. However, the assumption of conditional independence in this algorithm does not conform to objective reality which affects its categorization performance to some extent. In order to improve the categorization performance of Naive Bayes algorithm in text categorization, a Naive Bayes text categorization algorithm based on TF-IDF attribute weighting is used. Let's use TF-IDF, which takes in account the product of term frequency and inverse document frequency. TF-IDF shows the rarity of a word in the corpus. If a word is rare then probably its a signature word for a particular sentiment/information.

RESULT:-

After calculation of accuracy and generation of classification report for sentiment analysis, from test data from Naive Bayes classifier model ; We got an accuracy of 91.54% .