PROJECT REPORT

Sentiment Analysis Using Bag of words Model

Rup Kiran Boyana

Contents

[Introduction – 3](#_Toc140704942)

[Problem Statement – 3](#_Toc140704943)

[Datasets used – 3](#_Toc140704944)

[Approach 1 3](#_Toc140704945)

[Output of the Approach1 - 7](#_Toc140704946)

[Approach 2 8](#_Toc140704947)

[Output of the Approach2 – 10](#_Toc140704948)

[Conclusion – 11](#_Toc140704949)

# Introduction –

The sentiment expressed in a piece of text can be determined computationally using sentiment analysis and the BOW model. The bag of words module is frequently used in text mining and natural language processing. In this project, the bag of words technique is used to perform sentiment analysis, with the findings of the analysis being determined by a Naive based classifier.

# Problem Statement –

The bag of words model has some drawbacks, including the inability to take word order or semantics into account when expressing text as a collection of individual words. This restriction can be important since the meaning and context of the text can be affected by the order of words in a sentence as the words around them might change or reverse the meaning. The sentiment of each word or token is ascertained by sentiment analysis. Because of the constraints of the bag of words model, performing sentiment analysis using this method will change how words are perceived. Here, we will use the bag of words technique to analyze the sentiment of various social media comments. Based on the results We will be able to determine whether the bag of words model could be employed for sentiment analysis and whether it produces reliable results given its constraints based on the analysis's findings.

# Datasets used –

I used two separate sets of data. To generate one of the data sets, I collected both positive and negative reviews of the Star Wars films. The second dataset, which I acquired from Kaggle, is about the imdb reviews. Type, reviews, and label columns are present in every dataset. The column labels identify reviews as good or negative, the column reviews contain individual reviews of the people, and the column type categorizes the data into train and test data.

# Approach 1

Step1 –

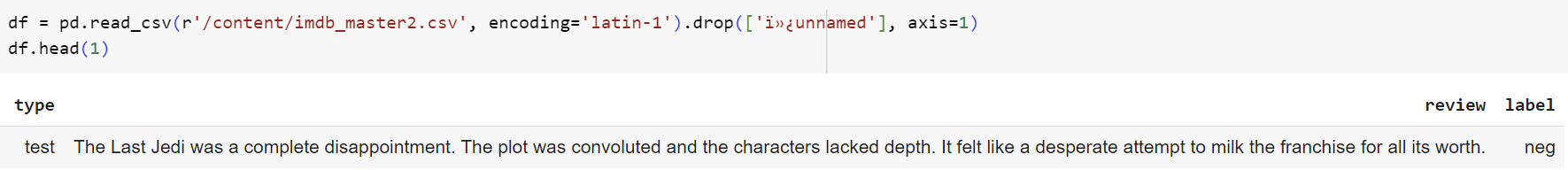
Import all the required libraries to your notebook.

A white background with black and white clouds

Description automatically generated

Step2 –

Covert the excel file to .csv format to your notebook, read the csv file and assign it to a data frame object



Step3 –

In this step I have assigned value 0 to negative reviews and value 1 to positive reviews from the label column and checked what are the value counts of the required columns.

A computer code on a white background

Description automatically generated

Step4 –

In this step I have converted all the text data into lower case.

A close up of a screen

Description automatically generated

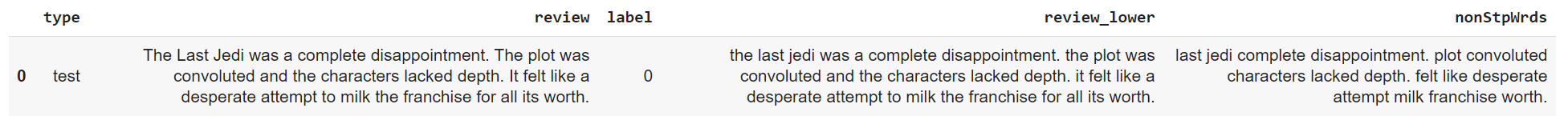
Step5 –

In this step I am eliminating the English stop words from the text data. Here the split method splits the text into individual words and checks if it is not in the English stop words and later the join method combined the words.

A screenshot of a computer code

Description automatically generated

Output –



Step6 –

In this step we are performing lemmatization on the nonStpwrds and are stored in a new column named lemmatizedWrds.

A screen shot of a computer

Description automatically generated

Output –

A close up of a text

Description automatically generated

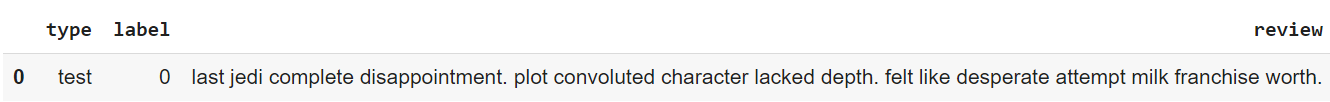
Step7 –

In this step we are dropping columns review, review\_lower and nonStpWrds, renaming the lemmatizedWrds column as reviews and stored in the data frame filtered.

A screenshot of a computer code

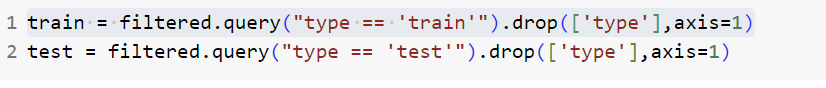
Description automatically generated

Output –



Step8 –

In this step we are selecting rows where the value in the type column is train and test and then dropping the type column from the data frame, later assigning the data to the newly created train and test data frames.



Step9 –

In this step I have created x\_train, y\_train, x\_test, y\_test variables and selecting review and label columns from the data frames train and test and then storing there values in the newly created variables.

A close-up of a computer code

Description automatically generated

Step10 –

In this step I have created instance of CountVectorizer and called fit and transform methods on the x\_train variable to fetch the words out of x\_train and making features and later converting it to the numeric representation.

A screenshot of a computer code

Description automatically generated

Step11 –

In this step I have created instance of MultinomialNB, later using the fit method and there by passing the trained\_vector and y\_train details to train the model.

A screenshot of a computer

Description automatically generated

Step12 –

In this step transform method is used to converting the extracted data into the numeric representation and later stored in the test\_vector variable. In the second line of code the model.predict() function takes this test data as input and returns the predicted labels for each data point in the test set.

A screenshot of a computer code

Description automatically generated

Step13 –

In this step I have used classification\_report() method from the sklearn library to generate an comprehensive report based on true labels and the predicted labels.

A close up of words

Description automatically generated

# Output of the Approach1 -

Dataset -1

A number of numbers on a white background

Description automatically generated

Dataset – 2

A number of numbers in a row

Description automatically generated

# Approach 2

After step 7 from approach1 I have worked on identifying parts of speech of every word from the reviews. For this I have created a column tokens and using nltk.word\_tokenize I have tokenized the sentences into words. Later created a new column posTags and using nltk.pos\_tag determined the parts of speech for every single word.

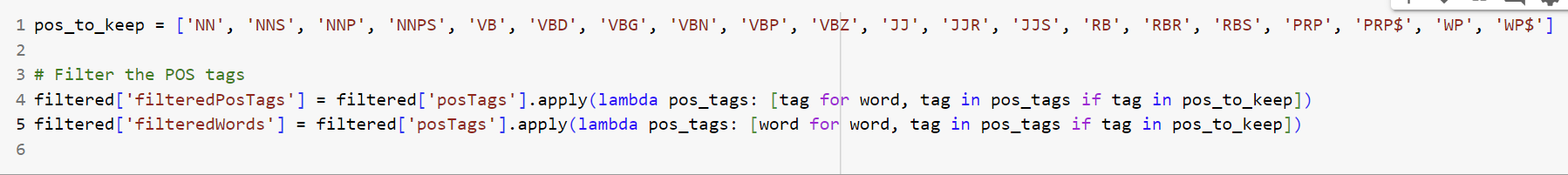
A screenshot of a computer program

Description automatically generated

A close up of a text

Description automatically generated

In this step I have created a pos\_to\_keep variable and stored them as a list by only selecting pos taggings that are used for sentiment analysis. The below used pos tags are related to Nouns, Verbs, Adjectives, Adverbs and Pronouns. Later I have created columns filteredPosTags and filteredWords and using lambda function determined each list in this column includes only the POS tags that are present in the pos\_to\_keep list and each list in this column includes only the words whose POS tags are present in the pos\_to\_keep list.



A screenshot of a computer screen

Description automatically generated

In this step using drop method I have eliminated unnecessary columns and renamed filteredWords column as review column.

A screenshot of a computer

Description automatically generated

In step10 and step12 from approach1 I have created a combined\_text variable and later using join method this list comprehension takes each text in x\_train and x\_test which is a list of sentences or documents, and joins the words in each text using a space as a separator and stored in the combined\_text.

A computer code with text

Description automatically generated

In step12 from approach1 have added the below code

A close-up of text

Description automatically generated

# Output of the Approach2 –

Dataset -1

A screenshot of a computer screen

Description automatically generated

Dataset -2

A number of numbers in a row

Description automatically generated

# 

# Conclusion –

By identifying and avoiding the model's limitations through case conversion of the text, eliminating stop words, lemmatization of the words, and finding accuracy using naive based model, the methods and techniques used in this paper have improved the accuracy of the Bag of Words model to determine sentiment analysis. The accuracy of the results was also improved by using specific parts of speech. If we compare the output results using the first approach and second approach, approach ones 1's precision, recall, f-1 score, and accuracy are significantly lower. This is made possible by incorporating specific speech patterns into our approach two, which increases the model's overall accuracy.