12/2/2022

Sentiment Analysis

On

Restaurant Reviews

Chandrol, Harshit

Masters of Science in Business Analytics

**Course Project**

MIST 6160 P 1 201 Advanced Data Mining 2022 Fall

**Table of Contents**

* [Overview](#overview)
* [Data Overview](#dataoverview)
* [Problem Statement](#problemstatement)
* [Business Objective](#businessobjective)
* [Exploratory Data Analysis](#EDA)
* [Modeling](#models)
* [Conclusion](#conclusion)

**Overview**

I have taken this task as a business case and will take certain assumptions before starting on the exercise.

Let's suppose you are the owner of a tech startup named “Previsit”. Previsit is a third-party consulting service that provides rating and reviews analysis to all e-commerce-based advisory companies and more in-depth customer sentiments for their products.

And as per this task, we are providing sentiment analysis to the restaurant chains present in New York.

We have been provided with all the customer unstructured ratings from their social media pages and they have asked us to build a solution for their customer support team to automatically classify the customer unstructured comments into a positive or negative class. This result helps them to target the customer with negative feedback and other business objectives. This report will demonstrate the stepwise process to understand how we perform sentiment analysis on unstructured text ratings.

**Problem Statement**

As mentioned above, analyzing unstructured customer text rating sentiment and providing a well-working solution to the client can be defined as the problem statement for this business case.

**Business Objective**

The sentiment analysis will help the client to target the customers which are not happy with the services they provide and have their ratings classified so they can connect with them to resolve their issues and improve customer experience.

**Data Overview**

Rating data with labels have been provided having 408369 customer reviews labeled as positive and negative which we will use to learn the model. Secondly, we have 102087 fresh reviews which are not labeled which we will use to predict from our trained model.

Let us start the process following the below steps.

1. First, we will load the data from our CSV files into the pandas data frame.

A picture containing graphical user interface

Description automatically generated

Code

1. Checking the null values in our datasets.

Graphical user interface, application, Word

Description automatically generated

Code

1. Removing the null values which we discovered in our previous step.

Graphical user interface, text, application, email

Description automatically generated

Code

**Exploratory Data Analysis**

1. Looking for positive sentiments in our data

Text

Description automatically generated

Code

* Importing nltk – nltk is a toolkit built for working with Natural language processing in python. It provides us with various text-processing libraries. A variety of tasks can be performed using nltk NLTK such as tokenizing, parse tree visualization etc.
* Removing stopwords – the stop words in nltk are the most common words in our data.
* Word cloud – using a word cloud to see what words appear the most.

Text

Description automatically generated

Figure

we can see that the most frequent word in out positive wordclous are "place","food","excellent","service","staff","delicious". However, how the words are used in the sentence also matters. this is not the final sentiment to be concluded.

1. Looking for negative sentiments in our train data

Graphical user interface, text, application

Description automatically generated

Code

Text

Description automatically generated

Figure

The negative sentiment word cloud was filled with mostly negative words, such as "restaurant", "service", "place", "average" etc. the words are not entirely negative but could be used negatively int he sentences. for example, the service was not good in XYZ restaurant.

1. Finally, we can take a look at the distribution of reviews with sentiment across the dataset:

Text

Description automatically generated

Code

Chart, histogram

Description automatically generated

Figure

1. Let us remove the stopwords from our train data and also stem the words to their original phase.

Graphical user interface, text, application, email

Description automatically generated

Code

* PorterStemmer – the porter stemming algorithm is a process for removing the commoner morphological and flexional endings from words in English.

1. We will also remove special characters from our reviews using regular expression and also convert the reviews into lower case and append all of the reviews into a corpus.

Text

Description automatically generated

Code

1. Converting the data in our corpus to a count vector which is required for the algorithms.

Graphical user interface, text

Description automatically generated

Code

1. Checking a few features.

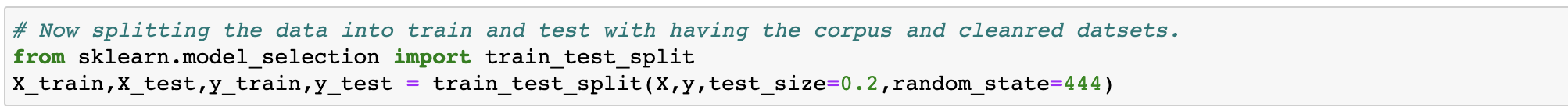
Text

Description automatically generated

Code

**Modeling**

1. Now, splitting the data into train and test using train\_test\_split()



Code

1. Implementing cross-validation and logistic regression using our data-rich corpus.

Graphical user interface, text

Description automatically generated

Code

We get the cross validation accuracy score of ~ 92%. Which is a decent score!

1. Performing hyperparameter tuning.

Graphical user interface, text

Description automatically generated

Code

We got the best parameter where C = 10 however, it didn’t increased our validation score of ~92%.

1. Let us predict our test data using our best estimator from gridsearchcv.

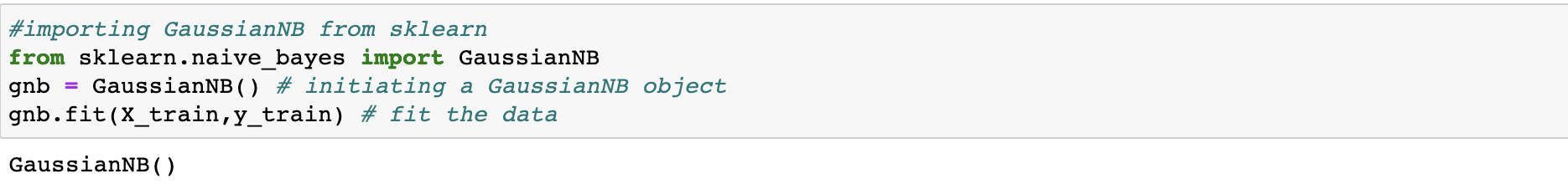
Graphical user interface, text, application

Description automatically generated

Code

* The accuracy score here also comes out to be ~92%. Which is similar for out train data.

1. Performing Naïve Bayes on the same datasets.



Code

* Predicting on test data with naïve bayes.

Graphical user interface, text, application, email

Description automatically generated

Code

* the accuracy score with GaussainNB came out to be 0.8197 or ~82% which is way less then logistic regression.

1. Checking N-gram bag of words. This can be done by performing tfidf

* Convert a collection of raw documents to a matrix of TF-IDF features.
* Here we will create a pipeline that will first vectorize our data using TF-IDF vectorizer and then perform a logistic regression algorithm with gridSearchCV of 5 folds.

Text

Description automatically generated

Code

1. Executing the above code will provide us the best\_score across cross validation and best parameters.

Graphical user interface, text

Description automatically generated

Code

* our best score is 0.94 or ~94% our best parameter acrooss all the values assigned is {'logisticregression\_\_C': 10, 'tfidfvectorizer\_\_ngram\_range': (1, 3)}

1. converting results to pandas data frame

Text

Description automatically generated

Code

1. checking the values across c and n\_gram range

Text

Description automatically generated

Code

Chart

Description automatically generated

Figure

* we can see from the results, we improved performance by a bit more than a percent by adding bigram and trigram features. We can visualize the cross-validation accuracy as a function of the ngram\_range and C parameter as a heat map

1. Extract feature names and coefficients.

Text

Description automatically generated

Code

Chart

Description automatically generated

Figure

* "disappointing" is indicative of a negative review, while "very not disappointing" and "be

disappointed" are indicative of a positive review. This is a prime example of context influencing the meaning of the word “disappoint.” Next, we’ll visualize only trigrams, to provide further insight into why these features are helpful. Many of the useful bigrams and trigrams consist of common words that would not be informative on their own, as in the phrases "none of the", "the only good", "on and on", "this is one", "of the most", and so on. However, the impact of these features is quite limited compared to the importance of the unigram features, as shown in coefficient visualization.

1. Finding 3-gram features.

Text

Description automatically generated

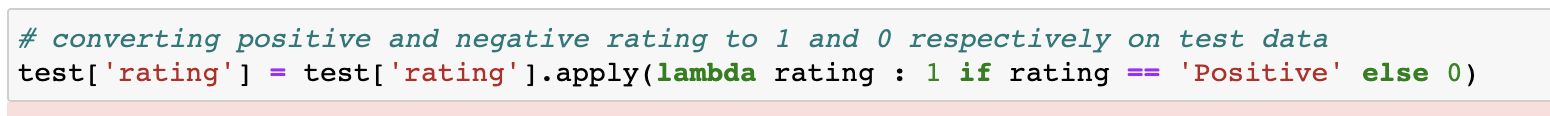
Code

A picture containing diagram

Description automatically generated

Figure

1. Performing validation on holdout data



Code

1. Feature engineering on holdout data.

Graphical user interface, text, application

Description automatically generated

Code

1. Converting the array, we have as a corpus to a list object.

Graphical user interface

Description automatically generated with medium confidence

Code

1. Adding the predicted labels to the fresh reviews and passing them to the customer support team to work further and connect with negative class customers for reaching business objectives.

**Conclusion**

Using the Tf-idf vectorizer we achieved an accuracy score of ~94%, we can deploy this algorithm as a system solution for the business case. It will help the customer classify the new reviews into positive and negative with ~94% accuracy.