SENTIMENT ANALYSIS

Machine Learning October Major Project

Done by: T. Kavya Persis

GITHUB LINK: https://github.com/kavyapersis-joshi/Verzeo-major-project-kavya-persis

1. Introduction

Automate detection of different sentiments from textual comments and feedback, A machine learning model is created to understand the sentiments of the restaurant reviews. The problem is that the review is in a textual form and the model should understand the sentiment of the review and automate a result.

The main motive behind this project is to classify whether the given feedback or review in textual context is positive or negative. Reviews can be given to the model and it classifies the review as a negative review or a positive. This shows the satisfaction of the customer or the experience the customer has experienced.

The basic approach was trying a different machine learning model and look for the one who is performing better on that data set. The restaurant reviews are very related to the project topic as reviews are made on websites and we can apply this model on such data sets to get the sentiments.

2. Problem Definition and Algorithm

2.1 Task Definition

To develop a machine learning model to detect different types of sentiments contained in a collection of English sentences or a large paragraph.

I have chosen Restaurant reviews as my topic. Thus, the objective of the model is to correctly identify the sentiments of the users by reviews which is an English paragraph and the result will be in positive or negative only.

For example,

If the review given by the user is:

" We had lunch here a few times while on the island visiting family and friends. The servers here are just wonderful and have great memories it seems. We sat on the oceanfront patio and enjoyed the view with our delicious wine and lunch. Must try! "

Then the model should detect that this is a positive review. Thus the output for this text will be Positive.

2.2 Algorithm Definition

The data set which I chose for this problem is available on Kaggle. The sentiment analysis is a classification because the output should be either positive or negative. That is why I tried 3 of the classification algorithms on this data set.

Multinomial Naive Bayes

Bernoulli Naive Bayes

Logistic Regression

i) Multinomial Naive Bayes:

Naive Bayes Classifier Algorithm is a family of probabilistic algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of a feature.

Bayes theorem calculates probability P(c|x) where c is the class of the possible outcomes and x is the given instance which has to be classified, representing some certain features.

> $P(c|x) = P(x|c) * P(c) / P(x)$

Naive Bayes is mostly used in natural language processing (NLP) problems. Naive Bayes predict the tag of a text. They calculate the probability of each tag for a given text and then output the tag with the highest one.

ii) Bernoulli Naive Bayes

BernoulliNB implements the naive Bayes training and classification algorithms for data that is distributed according to multivariate Bernoulli distributions; i.e., there may be multiple features but each one is assumed to be a binary-valued (Bernoulli, boolean) variable. Therefore, this class requires samples to be represented as binary-valued feature vectors; if handed any other kind of data, a BernoulliNB instance may binarize its input (depending on the binarize parameter).

> The decision rule for Bernoulli naive Bayes is based on:- $P(x_i \mid y) = P(i \mid y)\, x_i + (1 - P(i \mid y))(1 - x_i)$

which differs from multinomial NB's rule in that it explicitly penalizes the non-occurrence of a feature that is an indicator for class, where the multinomial variant would simply ignore a non-occurring feature.

In the case of text classification, word occurrence vectors (rather than word count vectors) may be used to train and use this classifier. BernoulliNB might perform better on some datasets, especially those with shorter documents. It is advisable to evaluate both models if time permits.

iii) Logistic Regression

Logistic regression is a supervised classification algorithm. In a classification problem, the target variable(or output), y, can take only discrete values for the given set of features(or inputs), X.

Contrary to popular belief, logistic regression is a regression model. The model builds a regression model to predict the probability that a given data entry belongs to the category numbered as "1". Just like Linear regression assumes that the data follows a linear function, Logistic regression models the data using the sigmoid function.

3. Experimental Evaluation

3.1 Methodology

All the models were judged based on a few criteria. These criteria are also recommended by the scikit-learn website itself for the classification algorithms.

The criteria are:

Accuracy score:

Classification Accuracy is what we usually mean when we use the term accuracy. It is the ratio of the number of correct predictions to the total number of input samples.

Confusion Matrix:

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known.

i) There are two possible predicted classes: "yes" and "no". If we were predicting the presence of a disease, for example, "yes" would mean they have the disease, and "no" would mean they don't have the disease.

ii) The classifier made a total of 165 predictions (e.g., 165 patients were being tested for the presence of that disease).

iii) Out of those 165 cases, the classifier predicted "yes" 110 times, and "no" 55 times.

iv) In reality, 105 patients in the sample have the disease, and 60 patients do not.

true positives (TP): These are cases in which we predicted yes (they have the disease), and they do have the disease.

true negatives (TN): We predicted no, and they don't have the disease.

false positives (FP): We predicted yes, but they don't have the disease. (Also known as a "Type I error.")

false negatives (FN): We predicted no, but they do have the disease. (Also known as a "Type II error.")

F1 score

F1 Score is the Harmonic Mean between precision and recall. The range for F1 Score is [0, 1]. It tells you how precise your classifier is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances).

High precision but lower recall, gives you an extremely accurate, but it then misses a large number of instances that are difficult to classify. The greater the F1 Score, the better is the performance of our model. Mathematically, it can be expressed as :

F1 Score tries to find the balance between precision and recall.

Precision: It is the number of correct positive results divided by the number of positive results predicted by the classifier.

Recall: It is the number of correct positive results divided by the number of all relevant samples (all samples that should have been identified as positive).

3.2 Result

All of the 3 mentioned machine learning models are very measured on the above-mentioned metrics. The result of the evaluation of the metrics is mentioned below:

i) Multinomial Naive Bayes:

Confusion Matrix:

[[119, 33],

[34, 114]]

Accuracy, Precision and Recall

Accuracy is 77.67 %

Precision is 0.78

Recall is 0.77

ii) Bernoulli Naive Bayes

Confusion Matrix:

[[115, 37],

[32, 116]]

Accuracy, Precision and Recall

Accuracy is 77.0 %

Precision is 0.76

Recall is 0.78

iii) Logistic Regression

Confusion matrix:

[[125, 27],

[43, 105]]

Accuracy, Precision and Recall

Accuracy is 76.67 %

Precision is 0.8

Recall is 0.71

The above results are very clear. That is why I chose Multinomial Naive Bayes and tried to tweak it by tuning the model for better results. For which I have iterated with different parameters and found the best-suited parameter with the highest accuracy.

4. Work

The approach was straight forward. I have selected a few classifiers algorithms for my project. I chose restaurant reviews as my project title. Firstly I understood the working of the algorithm and read about them.

After gathering the data set from Kaggle. The first step was to process the data. In data processing, I used NLTK (Natural Language Toolkit) and cleared the unwanted words in my vector. I accepted only alphabets and converted it into lower case and split it in a list.

Using the PorterStemmer method stem I shorten the lookup and Normalized the sentences.

Then stored those words which are not a stopword or any English punctuation.

Secondly, I used CountVectorizer for vectorization. Also used fit and transform to fit and transform the model. The maximum features were 1500.

The next step was Training and Classification. Using train_test_split 30% of data was used for testing and remaining was used for training. The data were trained on all the 3 algorithms mentioned above.

Later metrics like Confusion matrix, Accuracy, Precision, Recall were used to calculate the performance of the model.

The best model was tuned to get a better result.

Lastly, we checked the model with real reviews and found the model is detecting the sentiments of the customer reviews properly.

Google colab Link:
https://colab.research.google.com/drive/1X4T6SLzLkZ5kBIr4dI7MjuTBX0pjSDHp?usp=sharing

5. Heroku Deployment

Done using youtube references

6. Conclusion

The motive of the model is to correctly detect the sentiments of the textual reviews or feedback. The developed model has an accuracy of 77.67% and successfully detects the sentiments of the textual reviews or feedback.

The model has been tested with few of the online reviews and was found that it detects the sentiments correctly.

Thus, can conclude that the motive was successful and the model can be used to detect the sentiments of the reviews and feedback.