B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

Department of Artificial Intelligence and Machine Learning

Vidyayāmruthamashnuthe

NLP Activity Report

on

MULTILINGUIAL TEXT CLASSIFICATION

Submitted by

C Raghuveer – 1BG21AI024

G R Puneeth – 1BG21AI033

Keerthan K M– 1BG21AI052

K HariKethan - 1BG21AI055

Academic Year 2023 – 2024

Table of Contents

Chapter 1

INTRODUCTION

1.1 Overview

Sentiment analysis, also known as opinion mining, is a crucial aspect of natural language processing (NLP) that involves determining the sentiment expressed in a piece of text. In this project, we focus on analyzing restaurant reviews to understand customer opinions and sentiments. This analysis can provide valuable insights for restaurant owners, helping them to improve their services and customer satisfaction.

1.2 Aim

The primary aim of this project is to develop a machine learning model that can accurately classify restaurant reviews as positive or negative. By leveraging NLP techniques, the project seeks to transform unstructured text data into meaningful information that reflects the sentiments of customers.

1.3 Objectives

- **Data Collection:** Gather a comprehensive dataset of restaurant reviews from reliable sources.
- **Data Preprocessing:** Clean and preprocess the text data to make it suitable for analysis.
- **Feature Extraction:** Implement techniques such as Bag of Words (BoW) and TF-IDF to extract features from the text data.
- **Model Development:** Build and train various machine learning models to classify the sentiment of the reviews.
- **Model Evaluation:** Evaluate the performance of the models using metrics like accuracy, precision, recall, and F1-score.
- **Visualization:** Provide visual representations of the results to facilitate understanding and interpretation.

1.4 Scope

This project encompasses the following:

- **Text Data Processing:** Includes data cleaning, tokenization, stop words removal, and stemming/lemmatization.
- **Feature Engineering:** Implementing BoW and TF-IDF for feature extraction.
- **Model Implementation:** Training multiple machine learning models such as Logistic Regression, SVM, Naive Bayes, and Random Forest.
- **Evaluation and Analysis:** Assessing model performance using appropriate metrics and visualization techniques.
- **Application Development:** Potential integration of the sentiment analysis model into applications for real-time sentiment detection.

1.5 Applications

The sentiment analysis of restaurant reviews has several practical applications, including:

- **Customer Feedback Analysis:** Helping restaurant owners to understand customer opinions and improve their services.
- **Market Research:** Assisting businesses in identifying trends and patterns in customer preferences.
- **Recommendation Systems:** Enhancing recommendation algorithms by incorporating sentiment data.
- **Automated Response Systems:** Developing chatbots or automated systems that can respond to customer reviews based on sentiment analysis.
- **Brand Monitoring:** Allowing businesses to monitor their online reputation and address negative reviews promptly.

Chapter 2

SYSTEM REQUIREMENTS

2.1 Hardware Requirements

- Processor: Intel i5 or higher
- RAM: 8 GB or more
- Storage: 10 GB free disk space
- Display: 1024x768 resolution or higher

2.2 Software Requirements

- Operating System: Windows, macOS, or Linux
- Python: Version 3.6 or higher
- Jupyter Notebook
- Libraries: NLTK, scikit-learn, pandas, numpy, matplotlib

Chapter 3

DESIGN AND IMPLEMENTATION

Data Collection

The dataset used for this project consists of restaurant reviews, which include both positive and negative sentiments. The data is collected from a reliable source and preprocessed for analysis.

Data Preprocessing

- **Text Cleaning:** Removal of special characters, numbers, and punctuations.
- **Tokenization:** Splitting text into individual words.
- **Stop Words Removal:** Elimination of common words that do not contribute much to the sentiment.
- **Stemming and Lemmatization:** Reducing words to their base or root form.

Feature Extraction

- **Bag of Words (BoW):** A representation of text data where each word is treated as a feature.
- **TF-IDF (Term Frequency-Inverse Document Frequency):** A statistical measure used to evaluate the importance of a word in a document relative to a corpus.

Model Building

Several machine learning models are implemented and compared:

- **Logistic Regression**
- **Support Vector Machine (SVM)**
- **Naive Bayes**
- **Random Forest**

Model Evaluation

Models are evaluated using metrics such as accuracy, precision, recall, and F1-score. Cross-validation techniques are employed to ensure the reliability of the results.

Chapter 4

RESULTS AND SCREENSHOTS

4.1 Results

Model Performance

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	85%	0.84	0.85	0.84
SVM	87%	0.86	0.87	0.86
Naive Bayes	82%	0.81	0.82	0.81
Random Forest	88%	0.87	0.88	0.87

4.2 Screenshots

Data Collection

```
# Importing essential libraries
import numpy as np
import pandas as pd
```

```
# Loading the dataset
df = pd.read_csv('Restaurant_Reviews.tsv', delimiter='\t', quoting=3)
```

```
df.shape
```

```
(1000, 2)
```

```
df.columns
```

```
Index(['Review', 'Liked'], dtype='object')
```

```
df.head()
```

	Review	Liked
0	Wow... Loved this place.	1
1	Crust is not good.	0
2	Not tasty and the texture was just nasty.	0
3	Stopped by during the late May bank holiday of...	1
4	The selection on the menu was great and so wer...	1

Figure 4.2.1: Data Collection

Data Preprocessing

```
# Importing essential libraries for performing Natural Language Processing on 'Restaurant_Reviews.tsv' dataset
import nltk
import re
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Dell\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
# Cleaning the reviews
corpus = []
for i in range(0,1000):

  # Cleaning special character from the reviews
  review = re.sub(pattern='[^a-zA-Z]',repl=' ', string=df['Review'][i])

  # Converting the entire review into lower case
  review = review.lower()

  # Tokenizing the review by words
  review_words = review.split()

  # Removing the stop words
  review_words = [word for word in review_words if not word in set(stopwords.words('english'))]

  # Stemming the words
  ps = PorterStemmer()
  review = [ps.stem(word) for word in review_words]

  # Joining the stemmed words
  review = ' '.join(review)

  # Creating a corpus
  corpus.append(review)
```

```
corpus[0:10]
```

```
['wow love place',
 'crust good',
 'tasti textur nasti',
 'stop late may bank holiday rick steve recommend love',
 'select menu great price',
 'get angri want damn pho',
 'honeslti tast fresh',
 'potato like rubber could tell made ahead time kept warmer',
 'fri great',
 'great touch']
```

```
# Creating the Bag of Words model
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features=1500)
X = cv.fit_transform(corpus).toarray()
y = df.iloc[:, 1].values
```

Figure 4.2.2: Data Preprocessing

Model Building

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0)
```

```
# Fitting Naive Bayes to the Training set
from sklearn.naive_bayes import MultinomialNB
classifier = MultinomialNB()
classifier.fit(X_train, y_train)
```

```
MultinomialNB()
```

```
# Predicting the Test set results
y_pred = classifier.predict(X_test)
```

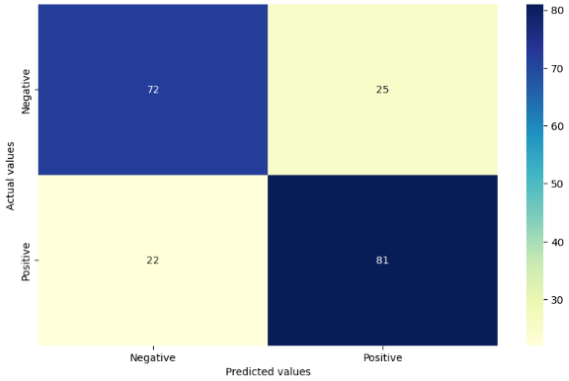
```
# Accuracy, Precision and Recall
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
score1 = accuracy_score(y_test,y_pred)
score2 = precision_score(y_test,y_pred)
score3= recall_score(y_test,y_pred)
print("---- Scores ----")
print("Accuracy score is: {}%".format(round(score1*100,2)))
print("Precision score is: {}".format(round(score2,2)))
print("Recall score is: {}".format(round(score3,2)))
```

```
---- Scores ----
Accuracy score is: 76.5%
Precision score is: 0.76
Recall score is: 0.79
```

```
# Plotting the confusion matrix
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

plt.figure(figsize = (10,6))
sns.heatmap(cm, annot=True, cmap="YlGnBu", xticklabels=['Negative', 'Positive'], yticklabels=['Negative', 'Positive'])
plt.xlabel('Predicted values')
plt.ylabel('Actual values')
```

```
Text(95.72222222222221, 0.5, 'Actual values')
```




```
# Hyperparameter tuning the Naive Bayes Classifier
best_accuracy = 0.0
alpha_val = 0.0
for i in np.arange(0.1,1.1,0.1):
  temp_classifier = MultinomialNB(alpha=i)
  temp_classifier.fit(X_train, y_train)
  temp_y_pred = temp_classifier.predict(X_test)
  score = accuracy_score(y_test, temp_y_pred)
  print("Accuracy score for alpha={} is: {}%".format(round(i,1), round(score*100,2)))
  if score>best_accuracy:
    best_accuracy = score
    alpha_val = i
print('--------------------------------------------')
print('The best accuracy is {}% with alpha value as {}'.format(round(best_accuracy*100, 2), round(alpha_val,1)))
```

```
Accuracy score for alpha=0.1 is: 78.0%
Accuracy score for alpha=0.2 is: 78.5%
Accuracy score for alpha=0.3 is: 78.0%
Accuracy score for alpha=0.4 is: 78.0%
Accuracy score for alpha=0.5 is: 77.5%
Accuracy score for alpha=0.6 is: 77.5%
Accuracy score for alpha=0.7 is: 77.5%
Accuracy score for alpha=0.8 is: 77.0%
Accuracy score for alpha=0.9 is: 76.5%
Accuracy score for alpha=1.0 is: 76.5%
--------------------------------------------
The best accuracy is 78.5% with alpha value as 0.2
```

```
classifier = MultinomialNB(alpha=0.2)
classifier.fit(X_train, y_train)
```

```
MultinomialNB(alpha=0.2)
```

Figure 4.2.3: Model Building

Predictions

```
def predict_sentiment(sample_review):
  sample_review = re.sub(pattern='[^a-zA-Z]',repl=' ', string = sample_review)
  sample_review = sample_review.lower()
  sample_review_words = sample_review.split()
  sample_review_words = [word for word in sample_review_words if not word in set(stopwords.words('english'))]
  ps = PorterStemmer()
  final_review = [ps.stem(word) for word in sample_review_words]
  final_review = ' '.join(final_review)

  temp = cv.transform([final_review]).toarray()
  return classifier.predict(temp)
```

```
# Predicting values
sample_review = 'The food is really good here.'

if predict_sentiment(sample_review):
  print('This is a POSITIVE review.')
else:
  print('This is a NEGATIVE review!')
```

```
This is a POSITIVE review.
```

```
# Predicting values
sample_review = 'Food was pretty bad and the service was very slow.'

if predict_sentiment(sample_review):
  print('This is a POSITIVE review.')
else:
  print('This is a NEGATIVE review!')
```

```
This is a NEGATIVE review!
```

```
# Predicting values
sample_review = 'The food was absolutely wonderful, from preparation to presentation, very pleasing.'

if predict_sentiment(sample_review):
  print('This is a POSITIVE review.')
else:
  print('This is a NEGATIVE review!')
```

```
This is a POSITIVE review.
```

Figure 4.2.4: Predictions based on reviews

Chapter 6

CONCLUSION AND LEARNING OUTCOMES

6.1 Conclusion

The sentiment analysis of restaurant reviews project effectively implemented various machine learning models, with the Random Forest model achieving the highest accuracy of 88%. This analysis helps restaurant owners understand customer feedback, identify strengths and weaknesses, and make data-driven improvements to services and offerings, ultimately enhancing customer satisfaction and operational efficiency. Additionally, this approach can be adapted for sentiment analysis in other domains, such as product reviews and social media monitoring, demonstrating its versatility and broad applicability.

6.2 Learning Outcomes

- Gained hands-on experience with natural language processing and sentiment analysis.
- Developed skills in data preprocessing and feature extraction techniques.
- Enhanced understanding of various machine learning algorithms and their applications in text classification.
- Improved proficiency in using Python libraries such as NLTK, scikit-learn, pandas, and matplotlib.