

HOTEL REVIEW OF SENTIMENT ANALYSIS

Submitted by,

Jiya Jagadeesh

1.INTRODUCTION

Reviews have become a prominent factor that affects people's bookings. people spend hours and even days sifting through dozens if not hundreds of options. The number of things to consider and the variety of reviews from previous guests is mind-blowing. guest reviews clearly influence people's booking decisions, which means, one should pay attention to what people are saying about their hotel. Not only should hoteliers strive for good reviews, but also implement it in a way that can help them learn the most about their customers. Reviews can tell the management if they are keeping up with the customers' expectations, which is crucial for developing marketing strategies based on the personas of the customers. Reviews are important and hotel owners, need to start leveraging it. Sentiment analysis is a classification problem which will be implemented using Naïve Bayes classification and logistic regression algorithm. Flask integration and IBM deployment will also be done.

1.1 OVERVIEW

Sentiment analysis is used to analyze raw text to drive objective quantitative results using natural language processing, machine learning, and other data analytics techniques. It is used to detect positive or negative sentiment in text, and often businesses use it to gauge branded reputation among their customers.

1.2 PURPOSE

One of the primary benefits of using sentiment analysis is knowing your hotel's reputation. Most of the reputation management software uses a combination of NLP, NPS, and machine-learning to evaluate a hotel's reputation. The reviews from all the booking platforms you are listed on are collaborated to form a dataset. online reviews are an extremely important part of any hotel business. It's the first thing someone sees when they search your hotel name, after all. They'll help increase traffic and bookings, as well as make guests feel more confident about their decision to stay at your establishment. Sentiment analysis (or opinion mining) is a natural language processing (NLP) technique used to determine whether data is positive, negative or neutral. Sentiment analysis is often performed on textual data to help businesses monitor brand and product sentiment in customer feedback, and understand customer needs.

2.LITERATURE SURVEY

2.1 EXISTING SYSTEM

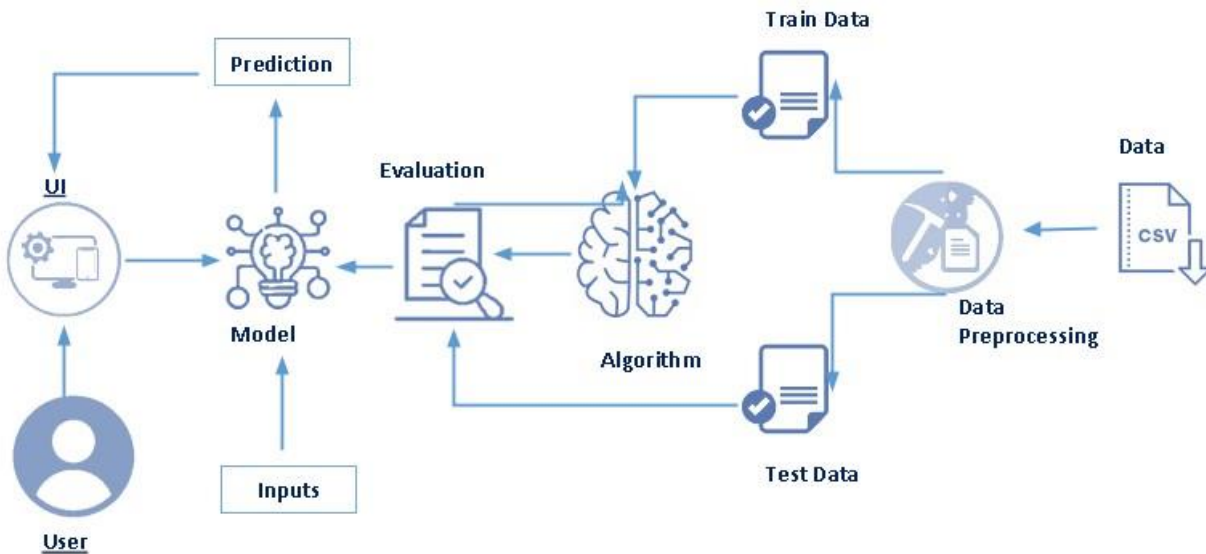
Sentiment analysis can be grouped into three main categories: knowledge-based techniques, statistical methods, and hybrid approaches. Knowledge-based techniques classify text by affect categories based on the presence of unambiguous affect words such as happy, sad, afraid, and bored.

2.2 PROPOSED SYSTEM

Sentiment analysis is a technique that allows us to analysis the emotion and opinions from text data. The text data can be in the form of tweets, product reviews, status, posts etc. This technique allows us to find out the sentiment behind a text data and do further analysis.

3.THEORITICAL ANALYSIS

3.1 BLOCK DIAGRAM



3.2 HARDWARE / SOFTWARE DESIGNING

The hardware required for the development of this project is:

Processor : Intel Corei3 7th Gen
Processor speed : 2.3GHz
RAM Size : 4 GB DDR
System Type : X64-based processor

SOFTWARE DESIGNING:

The software required for the development of this project is:

Desktop GUI	: Anaconda Navigator
Operating system	: Windows 10
Front end	: HTML, CSS, JAVASCRIPT
Programming	: PYTHON
Cloud Computing Service	: IBM Cloud Services

4.EXPERIMENTAL INVESTIGATION

IMPORTING AND READING THE DATASET

Importing the Libraries

First step is usually importing the libraries that will be needed in the program.

Pandas: It is a python library mainly used for data manipulation.

NumPy: This python library is used for numerical analysis.

Matplotlib and Seaborn: Both are the data visualization library used for plotting graph which will help us for understanding the data.

csr_matrix() :A dense matrix stored in a NumPy array can be converted into a sparse matrix using the CSR representation by calling the `csr_matrix()` function.

Train_test_split: used for splitting data arrays into training data and for testing data.

Pickle: to serialize your machine learning algorithms and save the serialized format to a file.

Reading the Dataset

For this project, we make use of datasets (hotel_reviews.csv). We will be selecting the important features from these datasets that will help us in recommending the best results.

The next step is to read the dataset into a data structure that's compatible with pandas.

Let's load a .csv data file into pandas. There is a function for it, called **read_csv()**. We will need to locate the directory of the CSV file at first (it's more efficient to keep the dataset in the same directory as your program). If the dataset in same directory of your program, you can directly read it, without any path. After the next Steps we made following bellow:

- 1.Data visualization
- 2.Collabrative and filtering
- 3.Creating the Model
- 4.Test and save the model
- 5.Buil Python Code
- 6.Build HTML Code
- 7.Run the Application

5.FLOWCHART

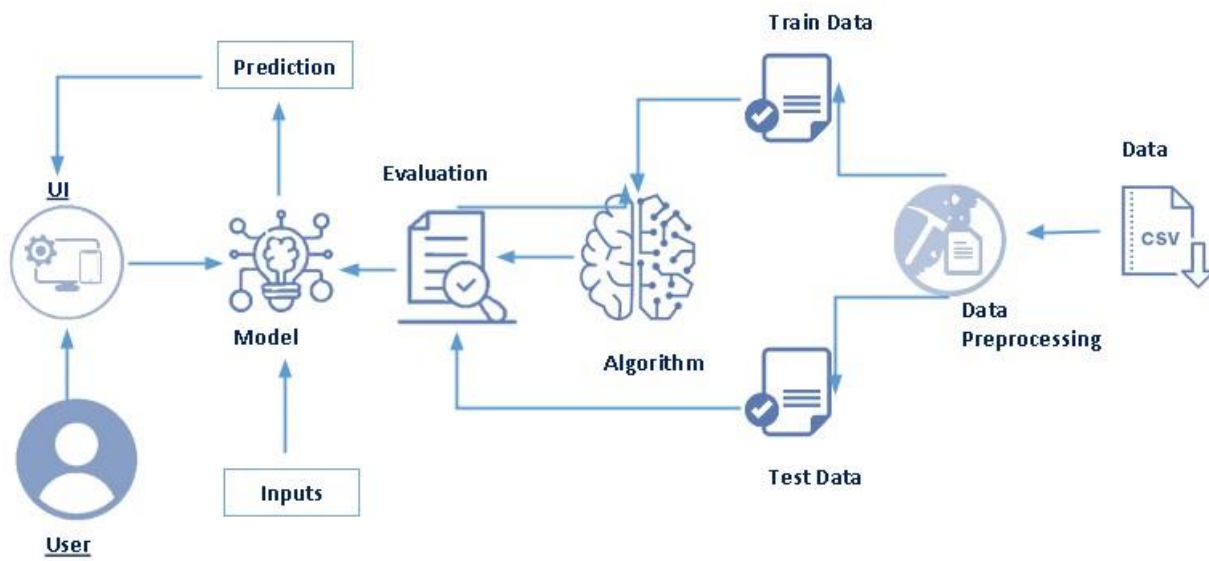


Fig 5.1 Flowchart of the project

Project Flow:

1. User interacts with the UI (User Interface) to upload the input features.
2. Uploaded features/input is analyzed by the model which is integrated.

Once a model analyses the uploaded inputs, the prediction is showcased on the UI.

1. Data Collection.

- Collect the dataset or Create the dataset

2. Data Pre- processing.

- Import the Libraries.
- Importing the dataset.
- Exploratory Data Analysis
- Data Visualization

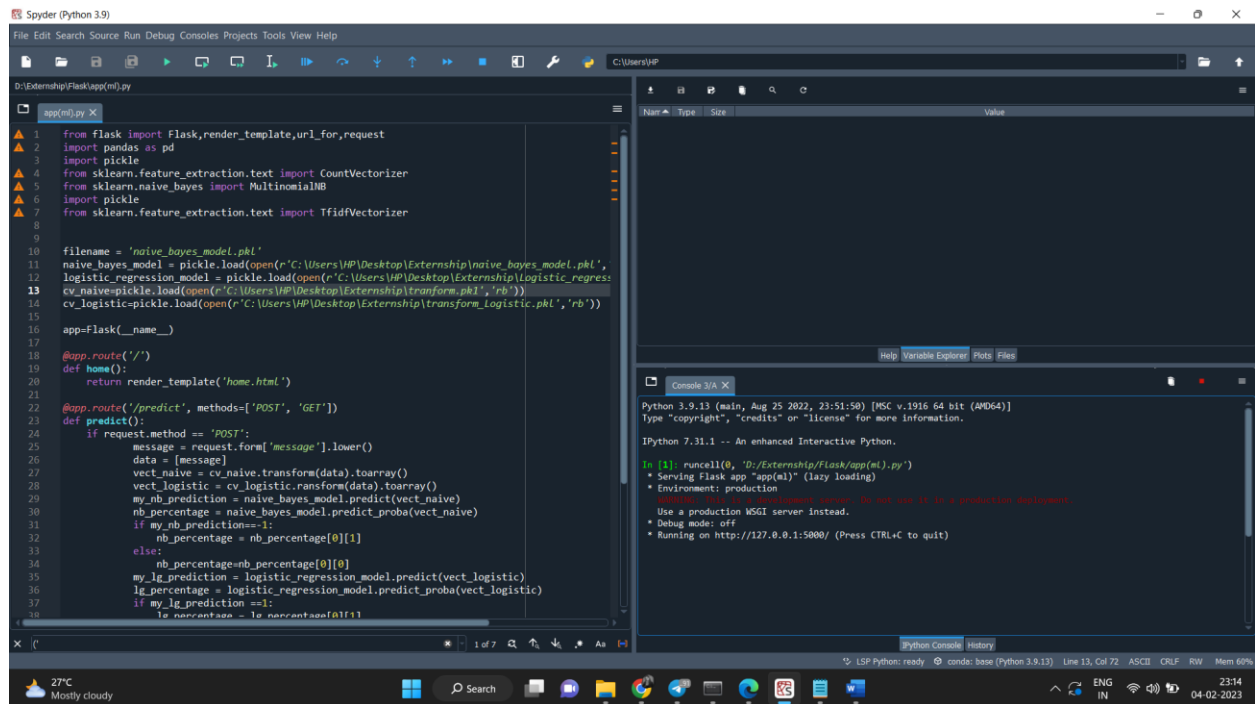
3. Collaborating Filtering

- Merging datasets
- Creating the Model
- Predicting the results
- Saving our model and dataset

4. Application Building

- Create an HTML file
- Build a Python Code

6.RESULT



The image shows the Spyder Python IDE interface. The main editor window displays a Python file named `app(ml).py` with the following code:

```
1 from flask import Flask,render_template,url_for,request
2 import pandas as pd
3 import pickle
4 from sklearn.feature_extraction.text import CountVectorizer
5 from sklearn.naive_bayes import MultinomialNB
6 import pickle
7 from sklearn.feature_extraction.text import TfidfVectorizer
8
9
10 filename = 'naive_bayes_model.pkl'
11 naive_bayes_model = pickle.load(open(r"C:\Users\HP\Desktop\Externship\naive_bayes_model.pkl",
12 logistic_regression_model = pickle.load(open(r"C:\Users\HP\Desktop\Externship\logistic_regres
13 cv_naive=pickle.load(open(r"C:\Users\HP\Desktop\Externship\transform.pkl", 'rb'))
14 cv_logistic=pickle.load(open(r"C:\Users\HP\Desktop\Externship\transform_logistic.pkl", 'rb'))
15
16 app=Flask(__name__)
17
18 @app.route('/')
19 def home():
20     return render_template('home.html')
21
22 @app.route('/predict', methods=['POST', 'GET'])
23 def predict():
24     if request.method == 'POST':
25         message = request.form['message'].lower()
26         data = [message]
27         vect_naive = cv_naive.transform(data).toarray()
28         vect_logistic = cv_logistic.ransform(data).toarray()
29         my_nb_prediction = naive_bayes_model.predict(vect_naive)
30         nb_percentage = naive_bayes_model.predict_proba(vect_naive)
31         if my_nb_prediction==1:
32             nb_percentage = nb_percentage[0][1]
33         else:
34             nb_percentage=nb_percentage[0][0]
35         my_lg_prediction = logistic_regression_model.predict(vect_logistic)
36         lg_percentage = logistic_regression_model.predict_proba(vect_logistic)
37         if my_lg_prediction ==1:
38             lg_percentage = lg_percentage[0][1]
39
40
```

The console window on the right shows the output of the application:

```
Python 3.9.13 (main, Aug 25 2022, 23:51:58) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license()" for more information.

IPython 7.31.1 -- An enhanced Interactive Python.
>>> In [1]: runcell(0, 'D:/Externship/Flask/app(ml).py')
* Serving Flask app "app(ml)" (lazy loading)
* Environment: production
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Fig 6.1 Flask code on Spyder

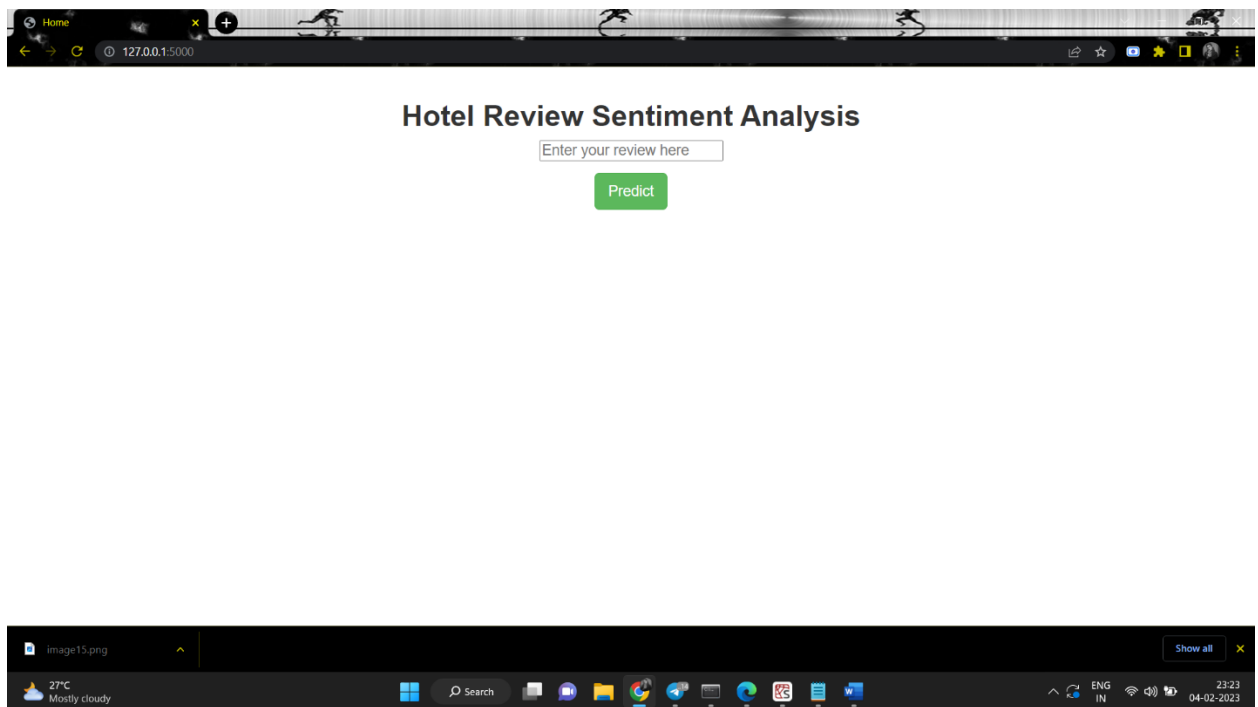


Fig 6.2 Home page for Hotel Review of sentiment analysis

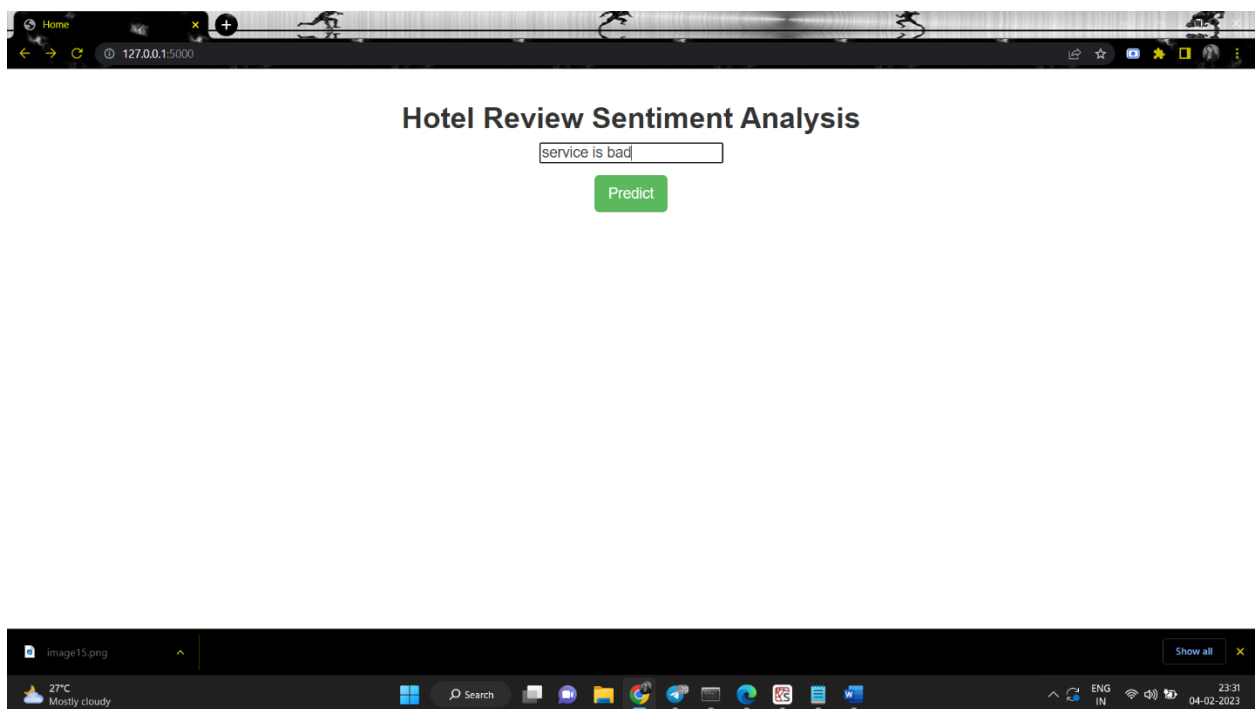


Fig 6.3 Predicting page of Hotel Review of sentiment analysis

7.ADVANTAGES AND DISADVANTAGES

ADVANTAGES

Sentiment analysis helps companies communicate better with customers and develop more relevant messages. By identifying the users' emotions, you can get a better idea of their experience and provide better customer service, which eventually leads to a decrease in customer churn. Sentiment analysis also means you'll be able to detect changes in the overall opinion towards your brand. Because it provides insight into the way your customers are feeling when they approach you, you can monitor trends and see if overall opinion towards your company drops or rises.

DISADVANTAGES

8.APPLICATIONS

9.CONCLUSION

10.FUTURESCOPE

11.BIBILOGRAPHY

- Hastie, Friedman, and Tibshirani, *The Elements of Statistical Learning*, 2001
- Bishop, *Pattern Recognition and Machine Learning*, 2006
- Ripley, *Pattern Recognition and Neural Networks*, 1996
- Duda, Hart, and Stork, *Pattern Classification*, 2nd Ed., 2002
- Tan, Steinbach, and Kumar, [Introduction to Data Mining](#), Addison-Wesley, 2005.
- Scholkopf and Smola, *Learning with Kernels*, 2002

APPENDIX

A Source Code of Flask:

```
from flask import Flask,render_template,url_for,request
import pandas as pd
import pickle
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
import pickle
from sklearn.feature_extraction.text import TfidfVectorizer

filename = 'naive_bayes_model.pkl'

naive_bayes_model =
pickle.load(open(r'C:/Users/HP/Desktop/Externship/naive_bayes_model.pkl','rb'))

logistic_regression_model =
pickle.load(open(r'C:/Users/HP/Desktop/Externship/Logistic_regression_model.pkl', 'rb'))

cv_naive=pickle.load(open(r'C:/Users/HP/Desktop/Externship/tranform.pkl','rb'))
cv_logistic=pickle.load(open(r'C:/Users/HP/Desktop/Externship/transform_Logistic.pkl','rb'))

app=Flask(__name__)

@app.route('/')
def home():
```

```
return render_template('home.html')
```

```
@app.route('/predict', methods=['POST', 'GET'])
```

```
def predict():
```

```
    if request.method == 'POST':
```

```
        message = request.form['message'].lower()
```

```
        data = [message]
```

```
        vect_naive = cv_naive.transform(data).toarray()
```

```
        vect_logistic = cv_logistic.transform(data).toarray()
```

```
        my_nb_prediction = naive_bayes_model.predict(vect_naive)
```

```
        nb_percentage = naive_bayes_model.predict_proba(vect_naive)
```

```
        if my_nb_prediction == -1:
```

```
            nb_percentage = nb_percentage[0][1]
```

```
        else:
```

```
            nb_percentage = nb_percentage[0][0]
```

```
        my_lg_prediction = logistic_regression_model.predict(vect_logistic)
```

```
        lg_percentage = logistic_regression_model.predict_proba(vect_logistic)
```

```
        if my_lg_prediction == 1:
```

```
            lg_percentage = lg_percentage[0][1]
```

```
        else :
```

```
            lg_percentage = lg_percentage[0][0]
```

```
        return render_template('result.html' ,
```

```
            message=message,
```

```
            my_nb_prediction = my_nb_prediction,
```

```
            nb_percentage=nb_percentage,
```

```
            my_lg_prediction = my_lg_prediction,
```

```
            lg_percentage=lg_percentage
```

```
        )
```

```
if __name__ == '__main__':
```

```
    app.run(debug=False)
```