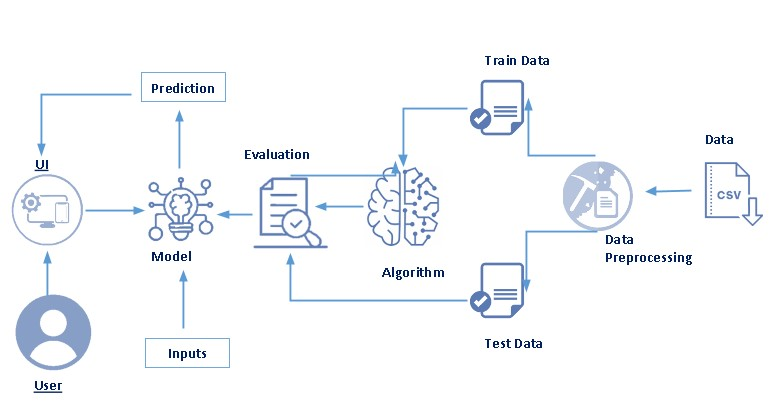
Sentiment Analysis of Hotel review

# Project Description

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 decision, 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.

**Architecture**



**Pre-requisites**

To complete this project, knowledge of the following software, concepts and packages is required.

* **Anaconda navigator** 
  + Refer the link below to download anaconda navigator
  + Link: <https://www.youtube.com/watch?v=5mDYijMfSzs>
* **Python packages**
  + Open anaconda prompt as administrator
  + Type “pip install pandas’” and click enter.
  + Type “pip install scikit-learn” and click enter.

**Prior Knowledge**

You must have prior knowledge of following topics to complete this project.

* **ML Concepts**
  + Supervised learning: <https://www.javatpoint.com/supervised-machine-learning>
  + Unsupervised learning: <https://www.javatpoint.com/unsupervised-machine-learning>
  + Regression and classification: <https://www.javatpoint.com/regression-vs-classification-in-machine-learning>
  + Naïve Bayes Classifier: [https://www.geeksforgeeks.org/naive-bayes-classifiers/#:~:text=Naive%20Bayes%20classifiers%20are%20a,is%20independent%20of%20each%20other](https://www.geeksforgeeks.org/naive-bayes-classifiers/%23:~:text=Naive%20Bayes%20classifiers%20are%20a,is%20independent%20of%20each%20other.%20)
  + Logistic Regression: [https://www.tutorialspoint.com/machine\_learning\_with\_python/classification\_algorithms\_logistic\_regression.htm#:~:text=Logistic%20regression%20is%20a%20supervised,be%20only%20two%20possible%20classes](https://www.tutorialspoint.com/machine_learning_with_python/classification_algorithms_logistic_regression.htm%23:~:text=Logistic%20regression%20is%20a%20supervised,be%20only%20two%20possible%20classes)
  + Evaluation metrics (Precision, recall, fbeta): [https://machinelearningmastery.com/fbeta-measure-for-machine-learning/](https://machinelearningmastery.com/fbeta-measure-for-machine-learning/%20)

**Project Objectives**

By the end of this project, you will:

* Know fundamental concepts and techniques used for machine learning.
* Gain a broad understanding about data.
* Have knowledge on pre-processing the textual data and classification algorithms.

# Project Flow

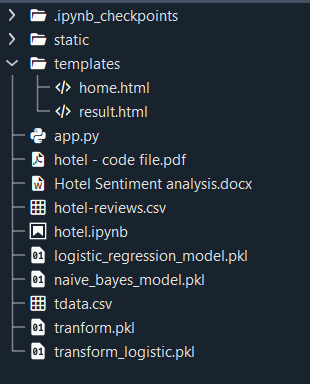
* The data is loaded and cleaning and feature extraction is performed on the data.
* Model is trained using machine learning models.
* Once model is trained, it is tested and evaluation is performed.

To accomplish this, we have to complete all the activities listed below,

* Data collection
  + Collect the dataset or create the dataset
* Data pre-processing
  + Removal of stop words
  + Creating count vectors
  + Splitting data into train and test
* Model building
  + Import the model building libraries
  + Initializing the model
  + Training and testing the model
  + Evaluating performance of model
  + Save the model

# Project Structure

Create the Project folder which contains files as shown below



* We are building a flask application which needs HTML pages stored in the templates folder and a python script app.py for scripting.
* Our saved models are logistic\_regression\_model.pkl and naïve\_bayes\_model.pkl. These models will be used for flask integration. Also, transform.pkl and transform\_logistic.pkl are the saved models for count vectorization for each model and will also be used for flask integration.
* Dataset is the hotel-reviews.csv file, downloaded from Kaggle.
* The tdata.csv file is the preprocessed file. (Since preprocessing the data is time consuming, this file has been saved.

# Data Collection

ML depends heavily on data. It is most crucial aspect that makes algorithm training possible. So this section allows you to download the required dataset.

**Download the dataset**

There are many popular open sources for collecting the data. Eg: kaggle.com, UCI repository, etc.

In this project, tripadvisor\_hotel\_reviews.csv is downloaded from kaggle.com (Name of the dataset has been changed after downloading to hotel-reviews,csv for ease in using). Please refer the link given below to download the dataset.

Link: <https://www.kaggle.com/datasets/andrewmvd/trip-advisor-hotel-reviews>

# Visualizing the data

As the dataset is downloaded. Let us read and understand the data properly with the help of some visualization techniques and some analysing techniques.

**There are numerous techniques for understanding the data. But here some of them are used. In addition, other and multiple techniques can also be used.**

**Importing the libraries**

**Import the necessary libraries as shown in the image.**



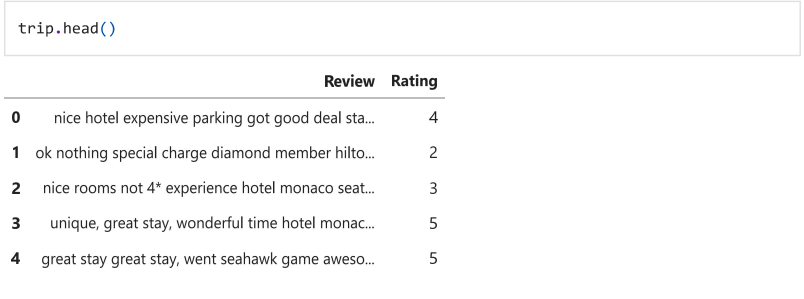
**Read the Dataset**

The dataset format might be in .csv, excel files, .txt, .json, etc. So, the dataset can be read with the help of pandas.

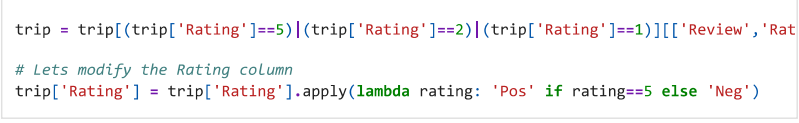
In pandas we have a function called read\_csv() to read the dataset. As a parameter we have to give the directory of csv file.

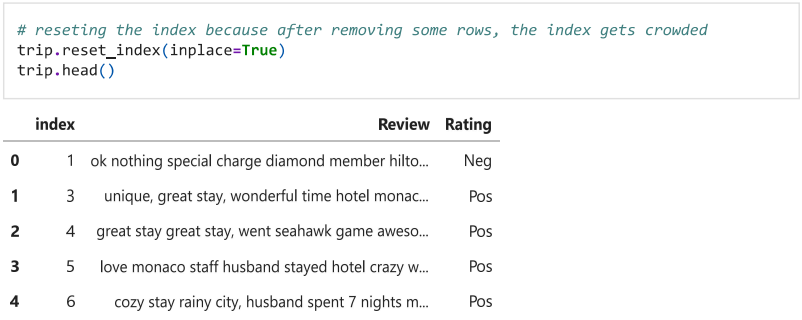


After reading the datasets, we will be viewing them by using the .head() function which will by default display first 5 rows of the dataset.

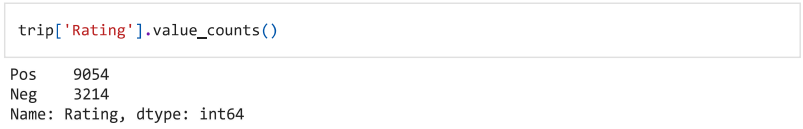


The dataset contains dependent features which have a 5 categorical rating from 1 to 5, due to which it is imbalanced. Due to its imbalanced nature, it is modified by converting the rating of 5 into positive and rating of 1 or 2 into negative.





After the modification, the count of positive and negative vales is fetched.



# Data Pre-processing

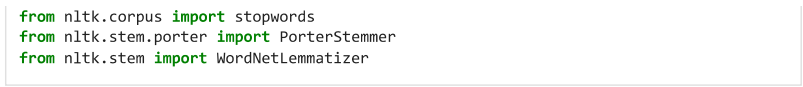
As we seen and understood the description of the data, lets pre-process the collected data.

For sentiment analysis, the downloaded data set is not suitable for training in the raw format. The text data set contains complete sentences, commas, period, and sometimes special characters. So, the dataset has to be cleaned properly in order to fetch good results. This activity includes the following steps.

* Regular expression : Removing all the symbols and numbers except the letters.
* lowercase : Lowering all the letters.
* Tokenization : Tokenization of each sentence into words for further processing.
* Lemmatization : Performing lemmatization operation on each words.
* Stopwords : Removing the words which do not put any weight into our features.

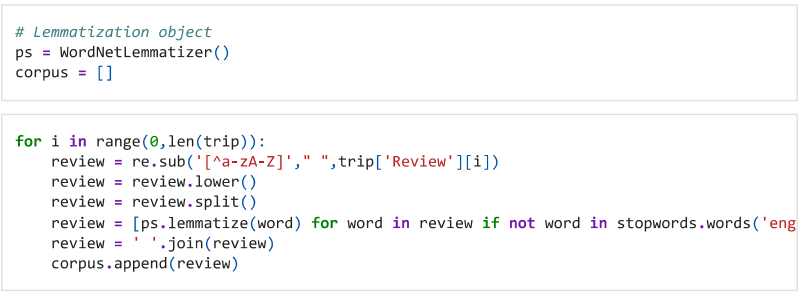
When implementing this for the first time, we need to install stopwords and wordnet from nltk.







Lemmatization is the process of grouping together the different inflected forms of a word so they can be analysed as a single item. Lemmatization links words with similar meanings to one word.



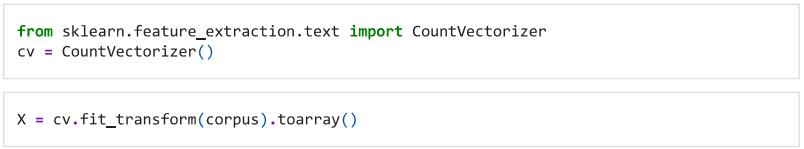
The above statements will perform the following on the ‘review’ column:

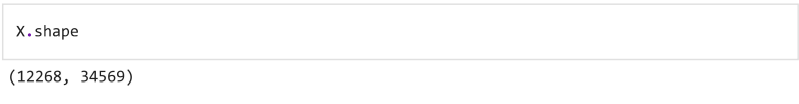
* Removal of special characters except the alphabet, i.e., a – z,
* Returns the string in all lowercase,
* Returns an array of substrings for each string,
* Performs lemmatization for all words except the stopwords,
* Returns the elements of a sequence separated by a space between them
* A corpus is defined as a collection of text documents. The results of the above statements is appended to the document. i.e., the dataset column ‘review’.

Note: This pre-processing is time consuming, so if one wants, they can save the pre-processed file as follows and use it for model building.



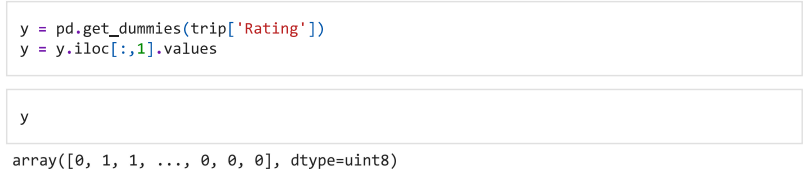
**CountVectorizer**is provided by the scikit-learn library in python. It is used to transform a given text into a vector on the basis of the frequency (count) of each word that occurs in the entire text. This is helpful when we have multiple such texts, and we wish to convert each word in each text into vector, and return it in a form of array by using .toarray() function.





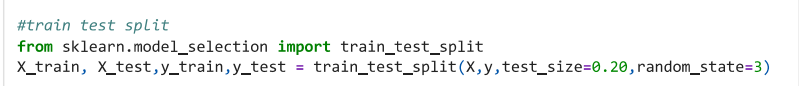
The .shape function returns the dimensions of the array.

The ‘review’ column has indicators as Pos and Neg. To convert them to indicator variable (1 and 0), the get\_dummies() function is used.



**Splitting data into train and test**

For splitting the dataset, X and y variables are created. For splitting training and testing data, train\_test\_split() function is used from sklearn. As parameters, X, y, *test\_size*, *random\_state*.

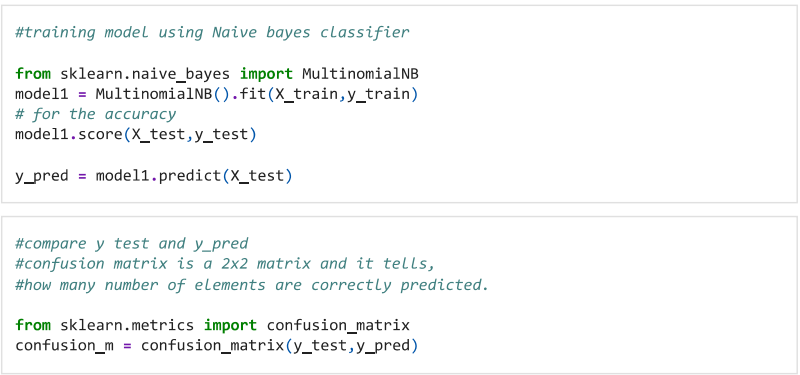


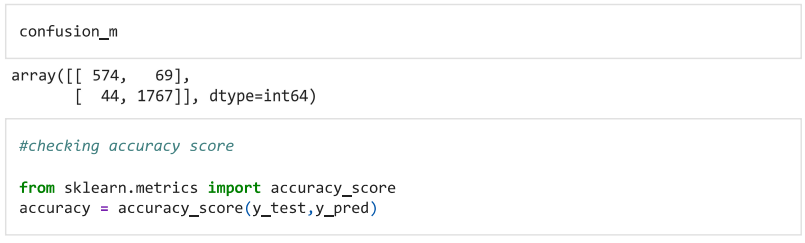
# Model Building

Now that the data is cleaned, it’s time to build the model. We can train our data on different algorithms. For this project we will be using the Naïve Bayes classifier and Logistic Regression classifier.

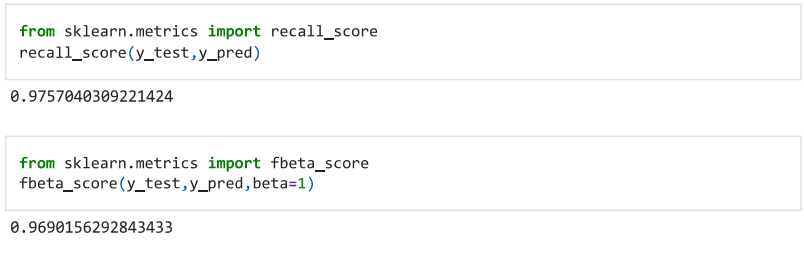
**Naïve Bayes Classifier**

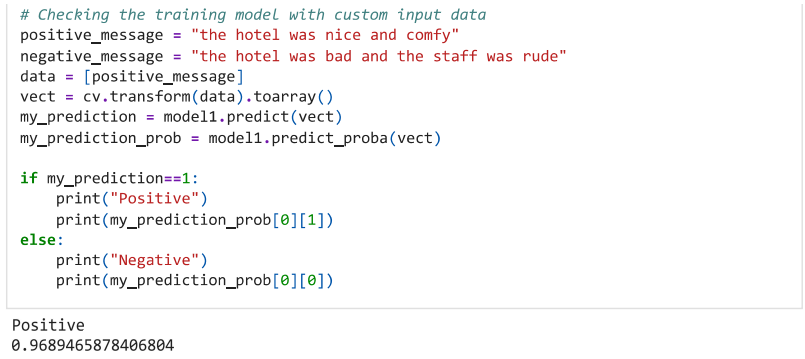
Naive Bayes classifier is used for classification for text classification. The multinomial distribution normally requires integer feature counts. It is called using the MultinomialNB from sklearn. The training set is passed for training using the .fit() function and the prediction is done using .predict() function. The confusion matrix and the evaluation metrics accuracy, precision, recall and fbeta score are also displayed.









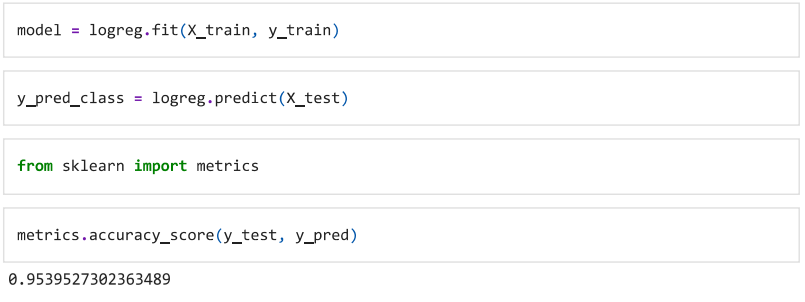


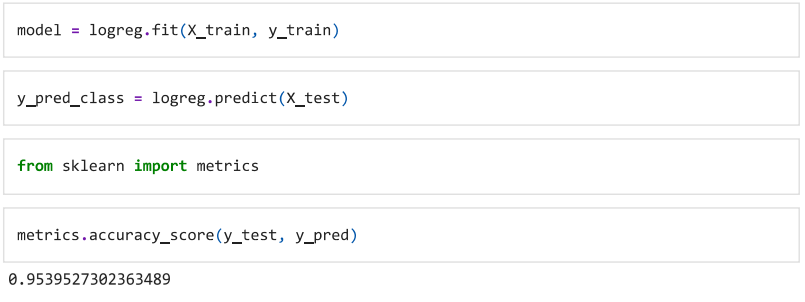


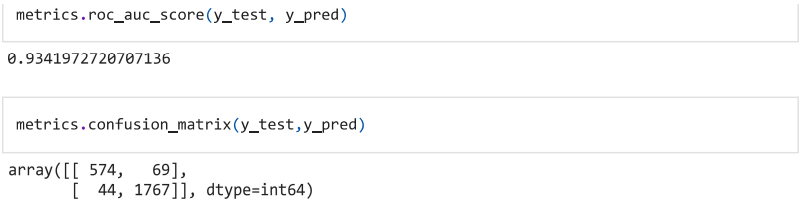
**Logistic Regression Classifier**

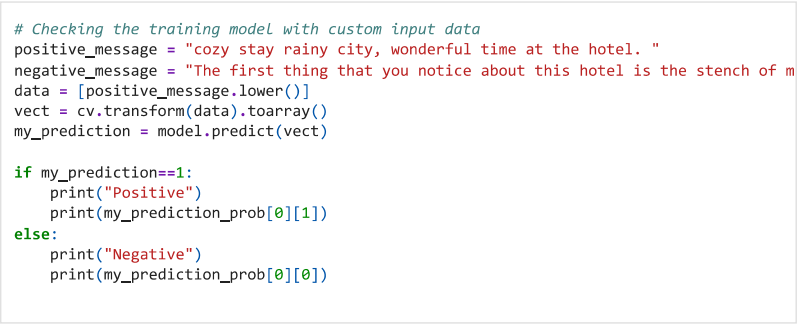
LogisticRegression function is called from sklearn, is nitialized and training data is passed to the model with .fit() function. The classification algorithm, the confusion matrix is displayed and, the accuracy and roc-auc score is displayed.

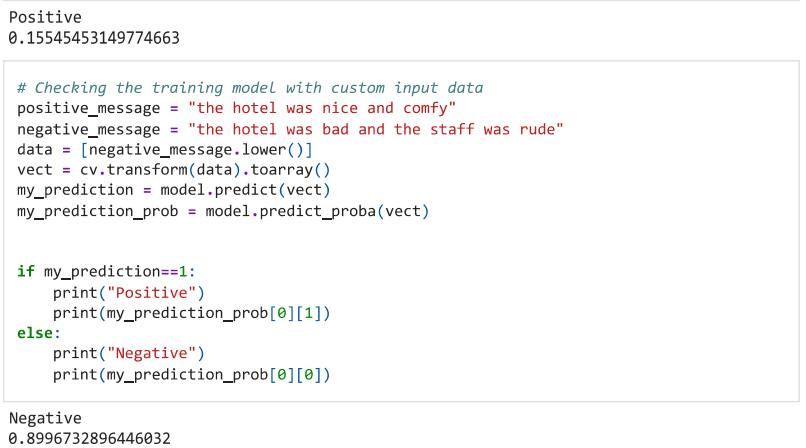




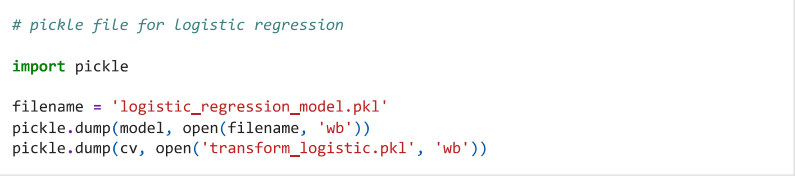




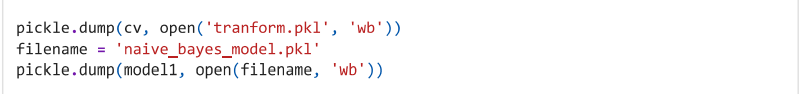




The model is saved as







* **Application Building**

In this section, we will be building a web application that is integrated to the model we built. A UI is provided for the uses where they have to enter the values for predictions. The entered values are given to the saved model and prediction is showcased on the UI.

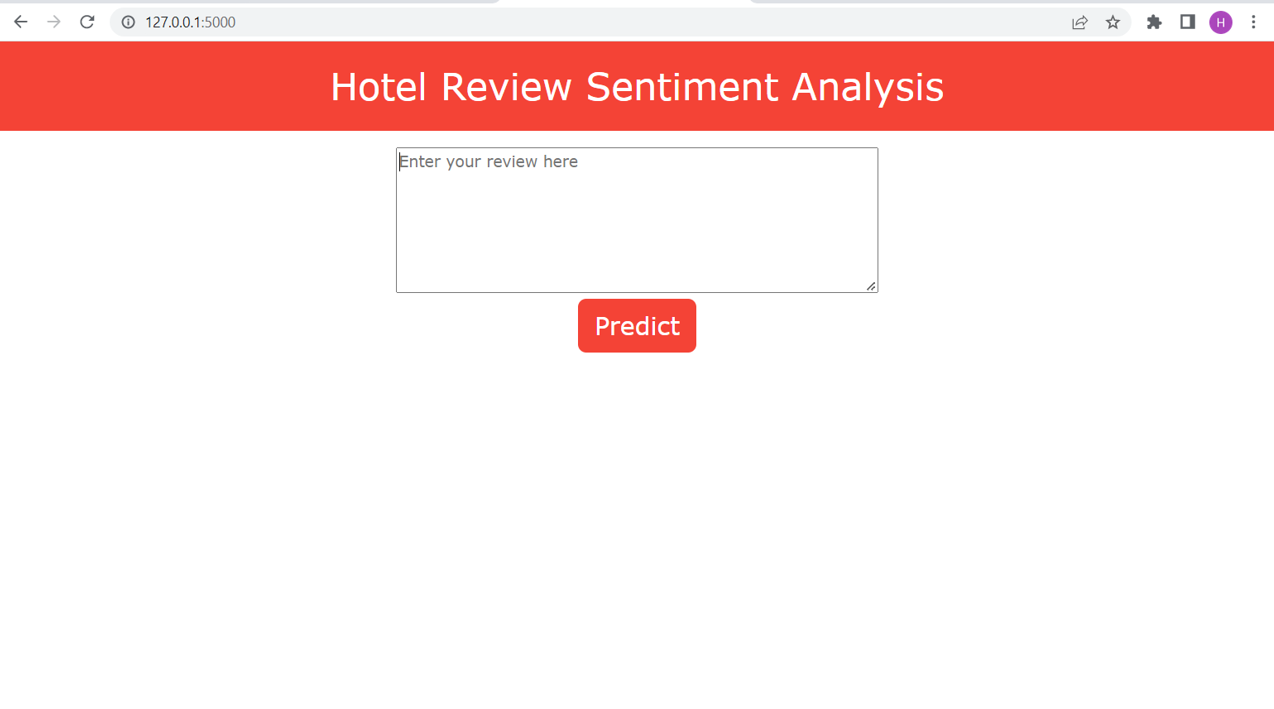
This section has the following tasks

* Building HTML Pages
* Building server-side script

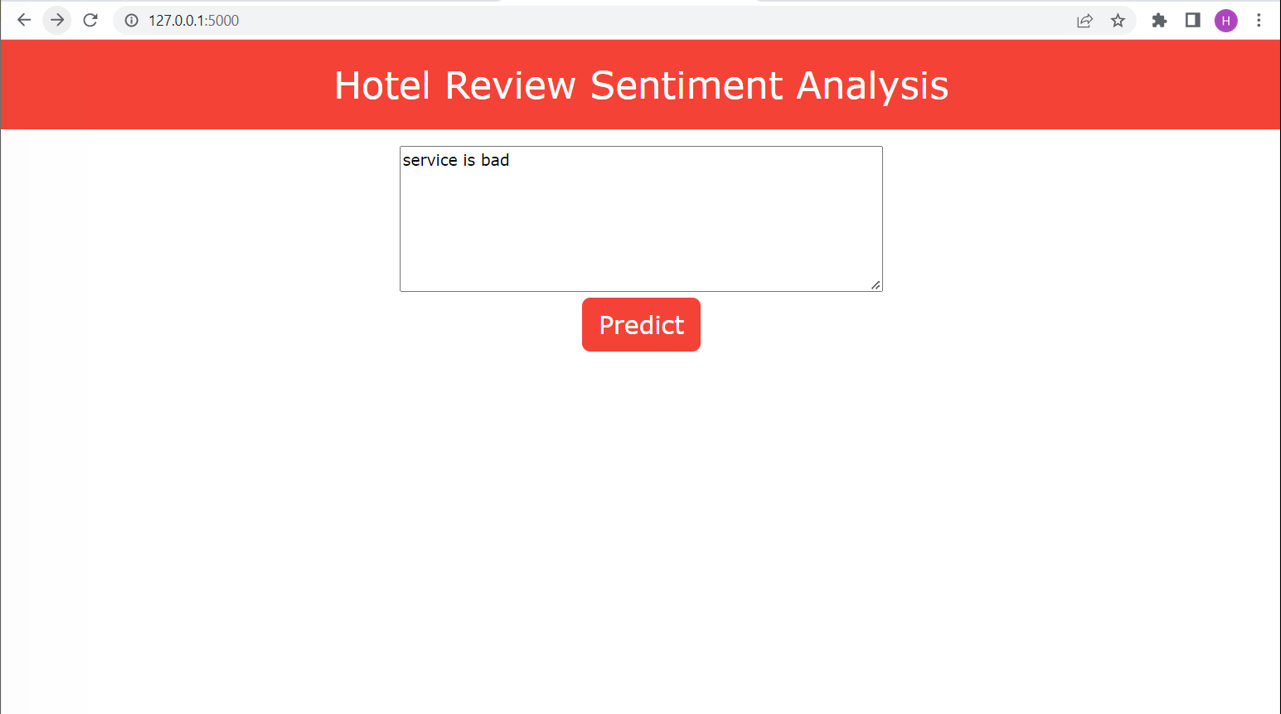
**Building HTML Pages:**

For this project, two HTML files called ‘home.html’ and ‘result.html’ are created and saved in templates folder.

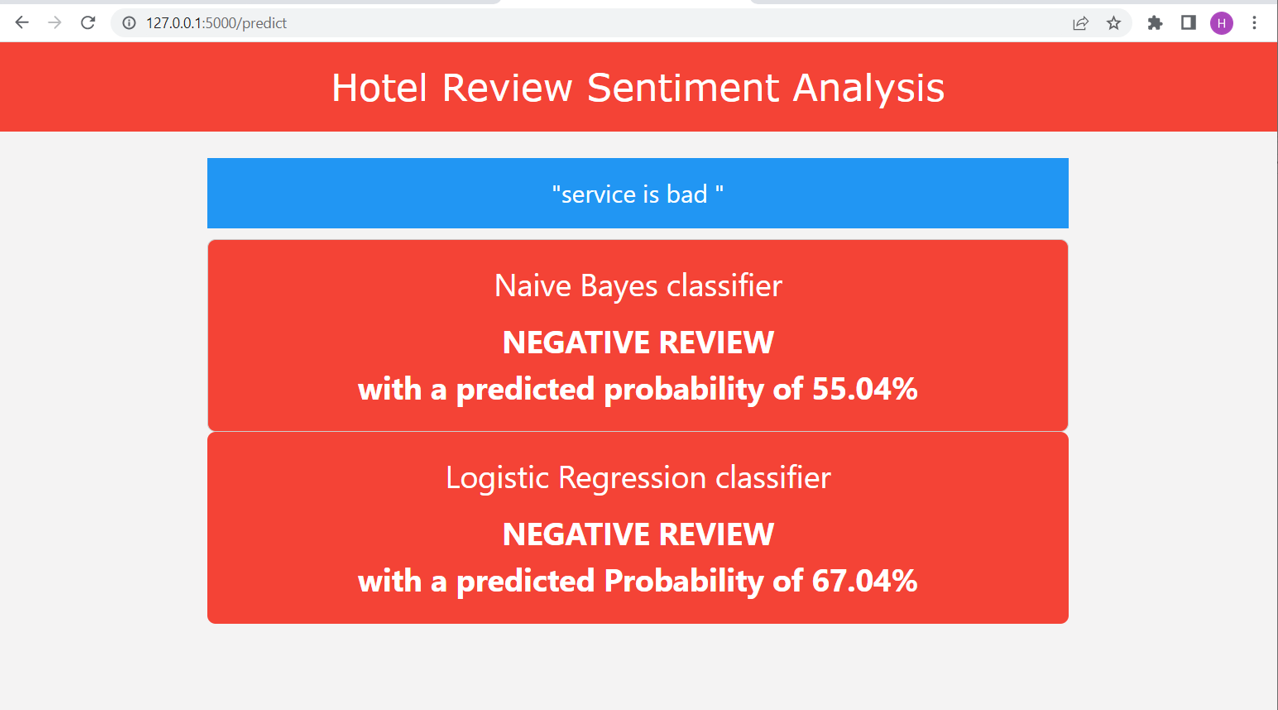
The home.html page looks like:

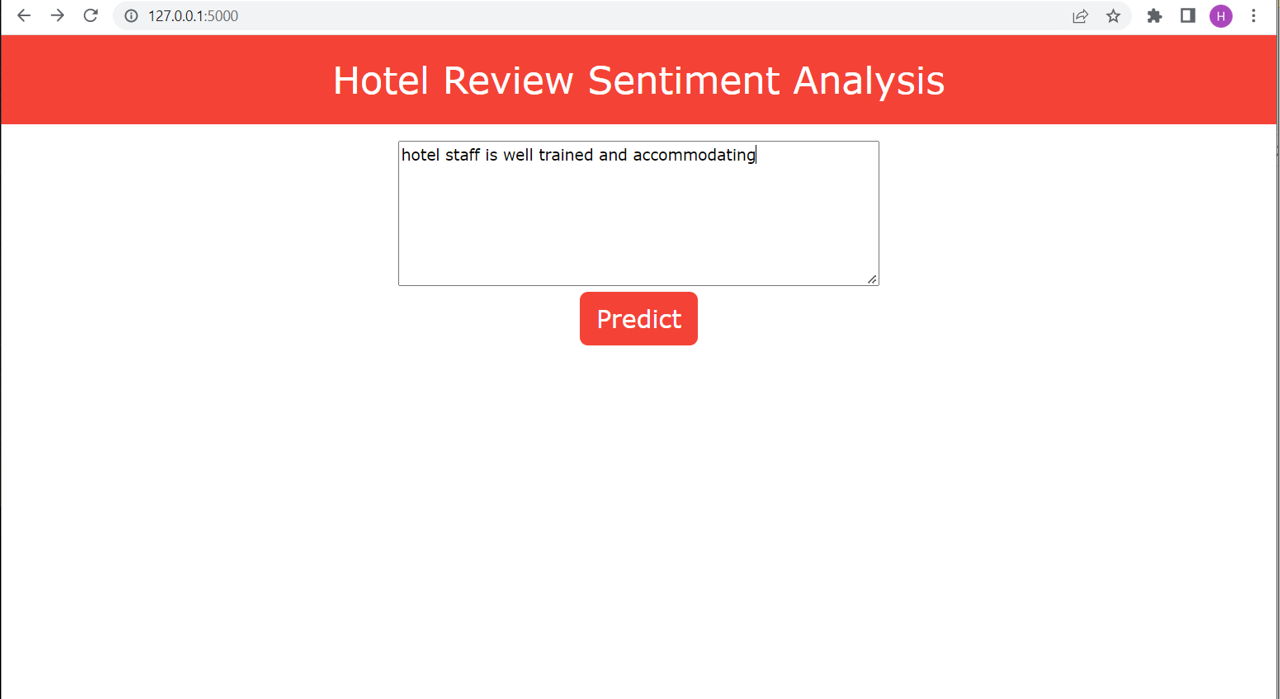


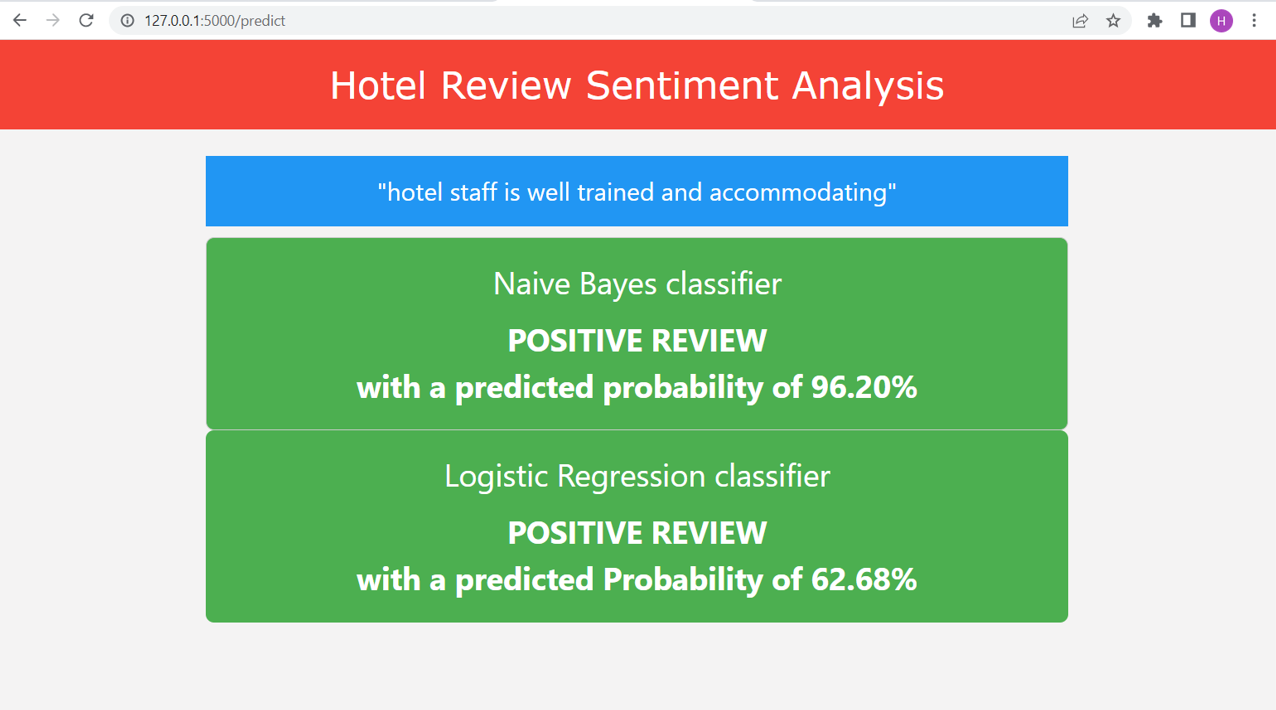
The home page is displayed and the user can enter their review.



After entering the review, we can perform the prediction by clicking on the predict button.



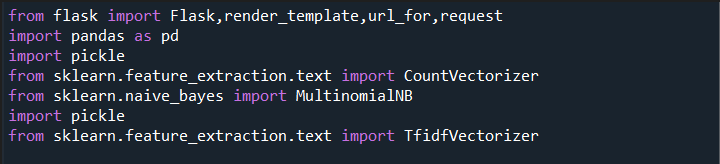




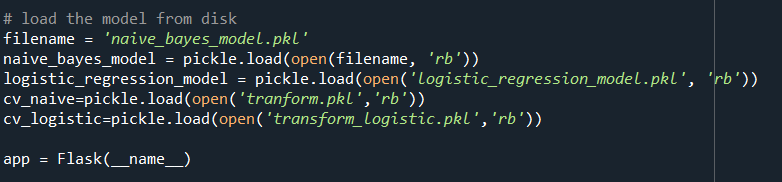
After the prediction is done, the predicted sales are displayed.

**Building the python code in flask:**

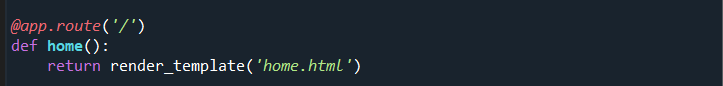
Import the libraries



Then load the saved model. Importing flask module in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of the current module (\_\_name\_\_) as argument.



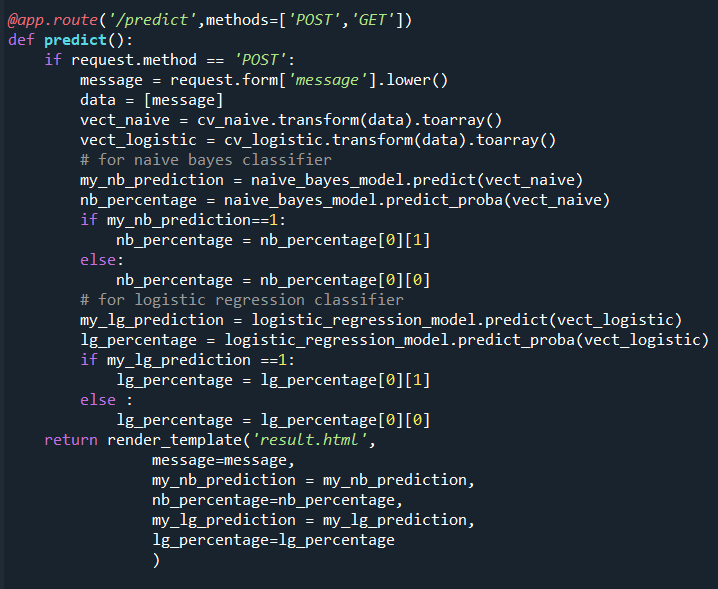
Render the HTML page:



Here we will be using declared constructor to route to the HTML page which we have created earlier and stores in the templates folder.

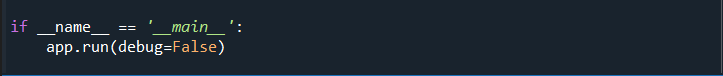
In the above example, ‘/’ URL is bound with index.html function. Hence, when the home page of the web server is opened in browser, the html page will be rendered. Whenever you enter the values from the html page the values can be retrieved using POST Method.

Retrieves the value from UI:



Here we are routing our app to predict() function. This function retrieves all the values from the HTML page using Post request. That is stored in an array. This array is passed to the prediction functions. This function returns the prediction. And this prediction value will rendered to the text that we have mentioned in the result.html page.

**Main Function:**



**Run the application**

* Open anaconda prompt from the start menu
* Navigate to the folder where your python script is.
* Now type “python app.py” command
* Navigate to the localhost where you can view your web page.
* Click on the predict button from the top right corner, enter the inputs, click on the submit button, and see the result/prediction on the web.

