Text

Description automatically generated

**Activity based**

**Project Report on**

**Artificial Intelligence**

**Project Phase - III**

**Submitted to Vishwakarma University, Pune**

**Under the Initiative of**

**Contemporary Curriculum, Pedagogy, and Practice (C2P2)**

**By**

**Imaad Imran Hajwane**

**SRN No: 202101132**

**Roll No: 23**

**Div: A**

**Third Year Engineering**

**Faculty In charge: - Prof. Tarapore Sir**

**Date Of Project Phase 2: - 23rd April 2024**

**Department of Computer Engineering**

**Faculty of Science and Technology**

**Academic Year**

**2023-2024 Term-II**

**AI: Phase III**

**Project Name: Product Review Sentiment Analysis for Restaurant Review with Food Images**

1. Problem Statement and Objectives:

Clearly define the problem statement including the implementation plan for the given algorithm.

Implement the Sentiment Analysis system in a programming language, incorporating the designed data processing pipeline and sentiment classification model. Develop modules for aspect-based sentiment analysis, feature extraction, and machine learning algorithms for sentiment classification. Utilize a labeled dataset for training and validating the model. Ensure the system accurately analyzes product reviews, considering both overall sentiment and sentiments related to specific aspects of the product.

Plan:

1. Understanding Requirements

* Review the provided statement to understand the requirements thoroughly.
* Clarify any ambiguities or uncertainties with stakeholders.

2. Data Acquisition

* Identify and obtain a labeled dataset suitable for sentiment analysis.
* Ensure the dataset covers various aspects of products and includes both positive and negative sentiments.
* Preprocess the dataset to remove noise, handle missing values, and standardize text formats.

3. Designing Data Processing Pipeline

* Develop a pipeline to preprocess raw text data before feeding it into the sentiment analysis model.
* Include steps such as tokenization, lowercasing, stop word removal, stemming or lemmatization, and handling special characters.

4. Aspect-Based Sentiment Analysis Module

* Design a module to extract aspects or features mentioned in the reviews.
* Associate sentiment polarity with each aspect mentioned in the reviews.

5. Machine Learning Algorithms for Sentiment Classification

* Choose appropriate machine learning algorithms for sentiment classification, such as:
* Logistic Regression
* Support Vector Machines (SVM)
* Random Forest
* Split the dataset into training, validation, and testing sets.
* Train multiple models using different algorithms and hyperparameters.
* Evaluate models using appropriate metrics like accuracy, precision, recall, and F1-score.

6. Model Evaluation and Selection

* Compare the performance of different models using validation data.
* Select the best-performing model based on evaluation metrics.
* Fine-tune the selected model if necessary.

7. Testing and Validation

* Conduct thorough testing to ensure the system functions correctly under various scenarios.
* Validate the accuracy and effectiveness of sentiment analysis, both overall and for specific product aspects, using test datasets and real-world reviews.

1. **Methodology details:**

* Identify dataset

The dataset used for this machine learning project is a text-based dataset for restaurant reviews

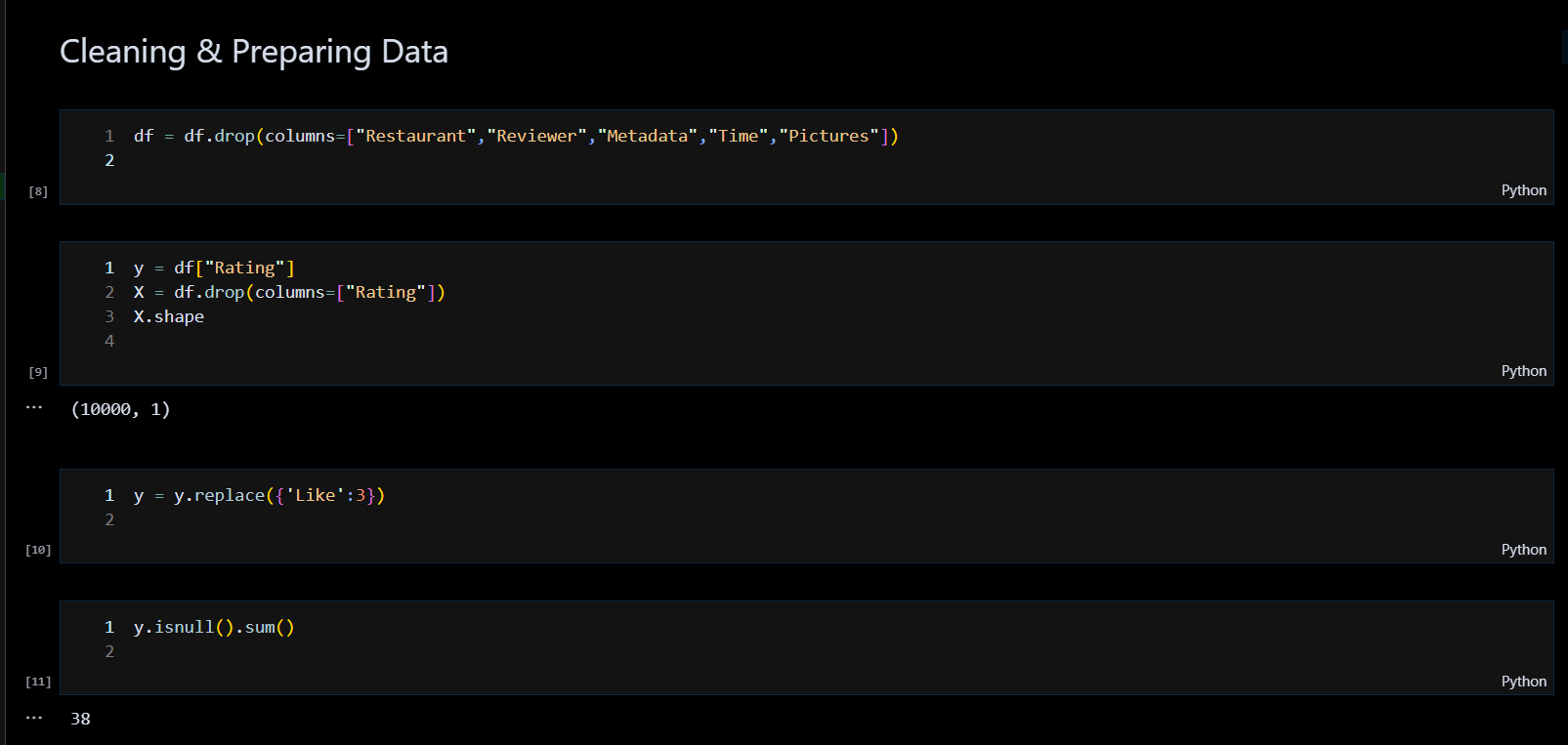
It contains 7 distinct columns in the dataset.

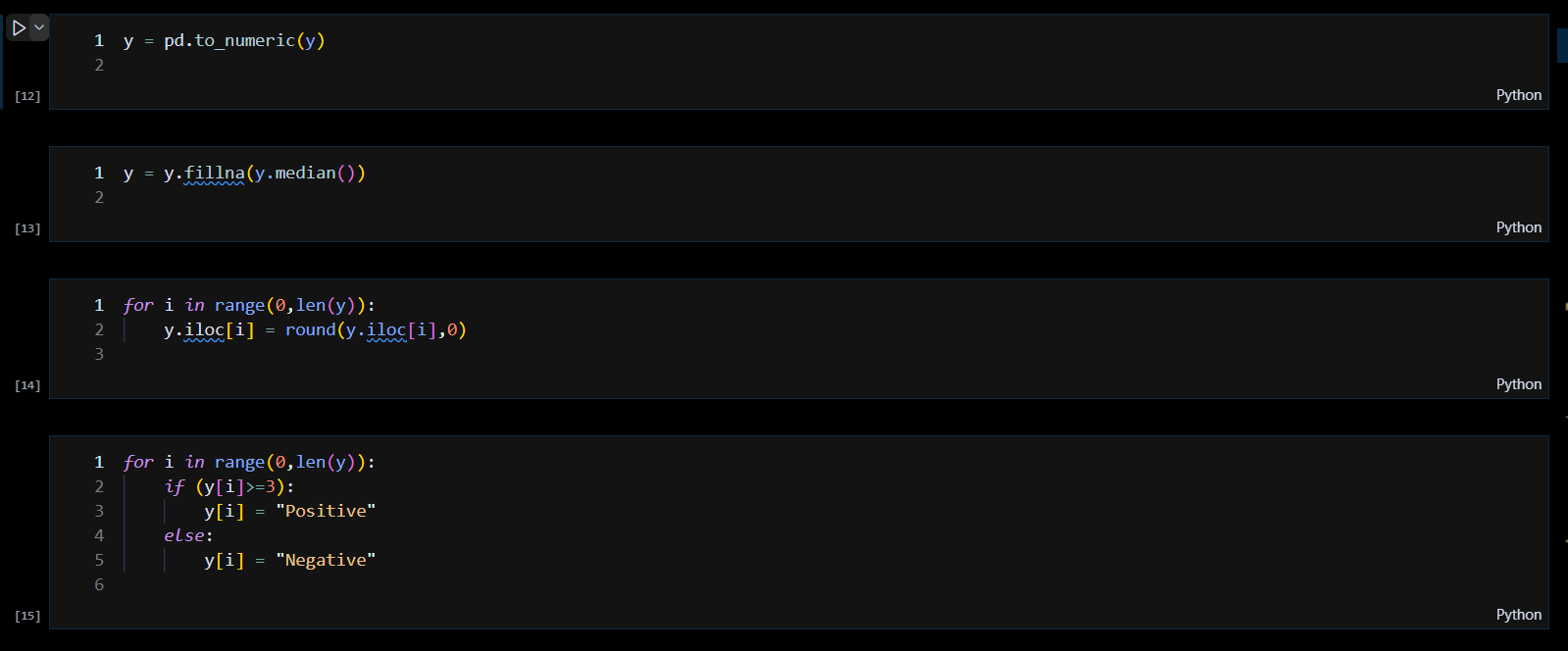
The columns are:

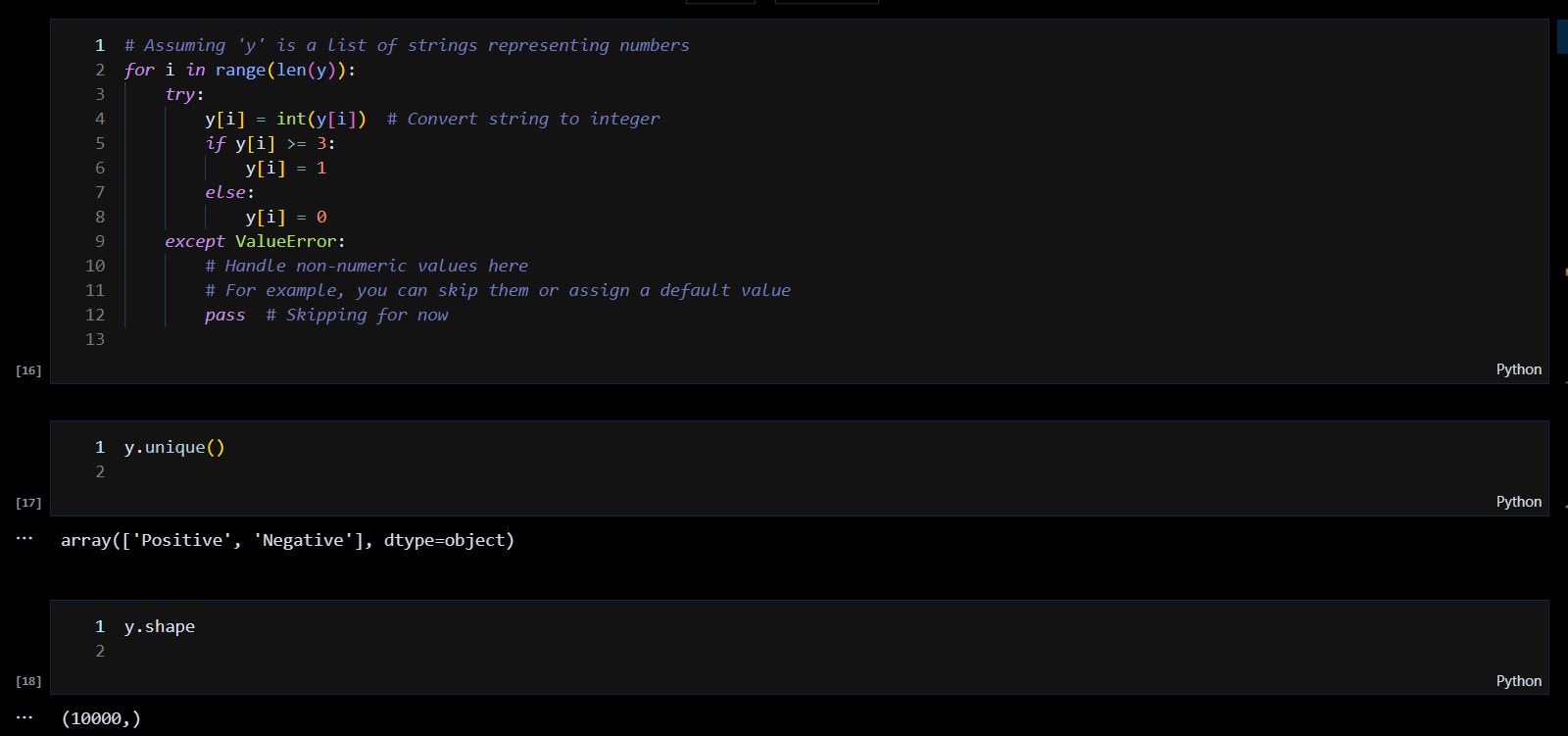
|  |  |
| --- | --- |
| Columns Names | *Descriptions:* |
| Restaurant | Name of the restaurant |
| Reviewer | Name of the reviewer |
| Review | Text of the review |
| Rating | Rating given by the reviewer |
| Metadata | Additional metadata related to the review (if any) |
| Time | Time of the review |
| Pictures | Number of pictures attached to the review (if any) |

These are the columns present in the dataset, which are used to create this machine learning project, using these columns and their values we can achieve the processing result for various machine learning algorithms.

* Preprocess dataset







* Implement algorithm

For this ML Project, we have used:

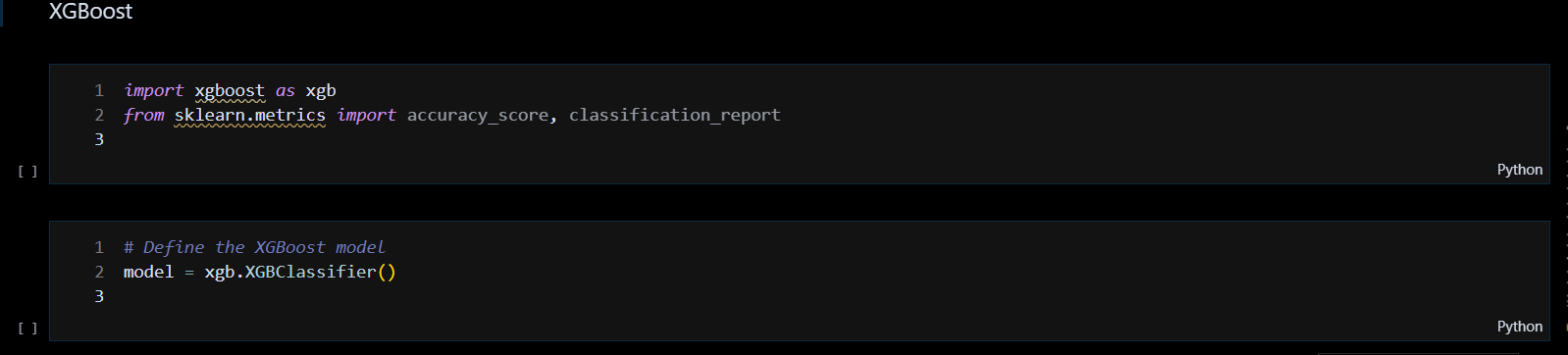
* LSTM
* MultinomialNB
* Random Forest
* SVM
* KNN
* LR
* XGBoost

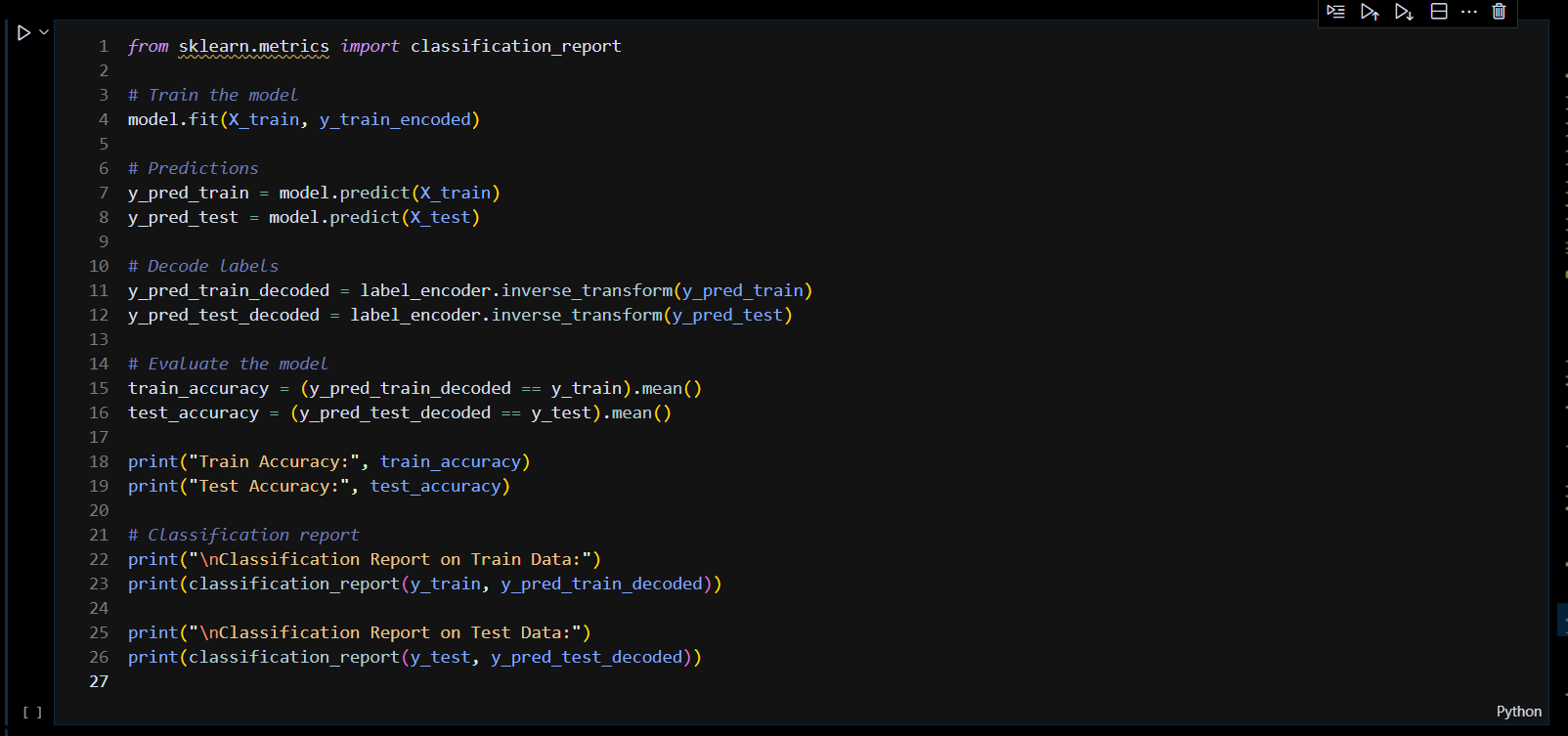
After using all these algorithms for the dataset, we have the results as follows:

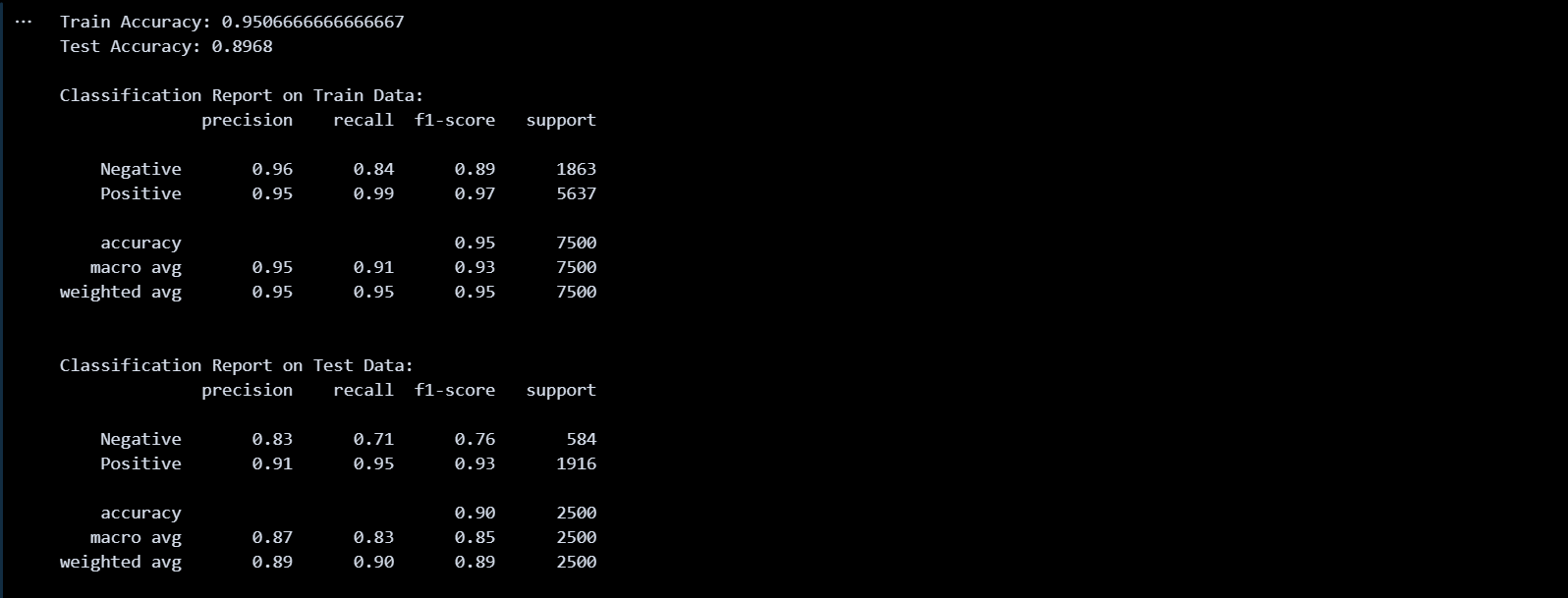
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Models Name | Accuracy | Recall | Precision | F-Beta-Score |
| MultinomialNB | 0.9084 | 0.856 | 0.7684 | 0.8865 |
| Random Forest | 0.8924 | 0.855 | 0.8321 | 0.8064 |
| SVM | 0.891 | 0.80 | 0.785 | 0.777 |
| KNN | 0.8156 | 0.70 | 0.689 | 0.80 |
| LR | 0.77 | 0.50 | 0.38 | 0.43 |
| XGBoost | 0.90 | 0.83 | 0.87 | 0.85 |

From all these algorithms we came to a conclusion that the XGBoost algorithm is the best suitable as per the defined dataset.

* Verify output with expected output based on domain knowledge







* Validation and testing

# *Importing Libraries*

*import* pandas *as* pd

*import* numpy *as* np

*from* sklearn.metrics *import* classification\_report,confusion\_matrix,accuracy\_score

*from* nltk.corpus *import* stopwords

*from* nltk.stem.porter *import* PorterStemmer

*import* re

*import* nltk

# *Importing Dataset*

df=pd.read\_csv('Restaurant reviews.csv', encoding = "ISO-8859-1")

df = df.drop(columns=["Restaurant","Reviewer","Metadata","Time","Pictures"])

# *Transforming & Cleaning Data*

y = df["Rating"]

X = df.drop(columns=["Rating"])

y = y.replace({'Like':3})

y = y.fillna(y.median())

y = pd.to\_numeric(y)

*for* i *in* range(0,len(y)):

    y.iloc[i] = round(y.iloc[i],0)

*for* i *in* range(0,len(y)):

*if* (y[i]>=3):

        y[i] = "Positive"

*else*:

        y[i] = "Negative"

# *Applying Stemming with excluding StopWords*

ps = PorterStemmer()

corpus = []

*for* i *in* range(0, len(X)):

    review = re.sub('[^a-zA-Z]',' ', str(X['Review'][i]))

    review = review.lower()

    review = review.split()

    review = [ps.stem(word) *for* word *in* review *if* not word in stopwords.words('english')]

    review = ' '.join(review)

    corpus.append(review)

# *Creating Matrix of CountVectorizer*

*from* sklearn.feature\_extraction.text *import* CountVectorizer

cv = CountVectorizer(max\_features=9000)

X = cv.fit\_transform(corpus).toarray()

# *Train-Test Split*

*from* sklearn.model\_selection *import* train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.25, random\_state=0)

# *Applying MultinomialNB*

*from* sklearn.naive\_bayes *import* MultinomialNB

classifier = MultinomialNB().fit(X\_train, y\_train)

# *Making Predictions*

y\_pred = classifier.predict(X\_test)

# *Creating Consusion Matrix*

*from* sklearn.metrics *import* confusion\_matrix

confusion\_m = confusion\_matrix(y\_test, y\_pred)

print(confusion\_m)

# *Getting the Accuracy*

*from* sklearn.metrics *import* accuracy\_score

accuracy = accuracy\_score(y\_test, y\_pred)

print(accuracy)

# *Dumping Models*

*import* pickle

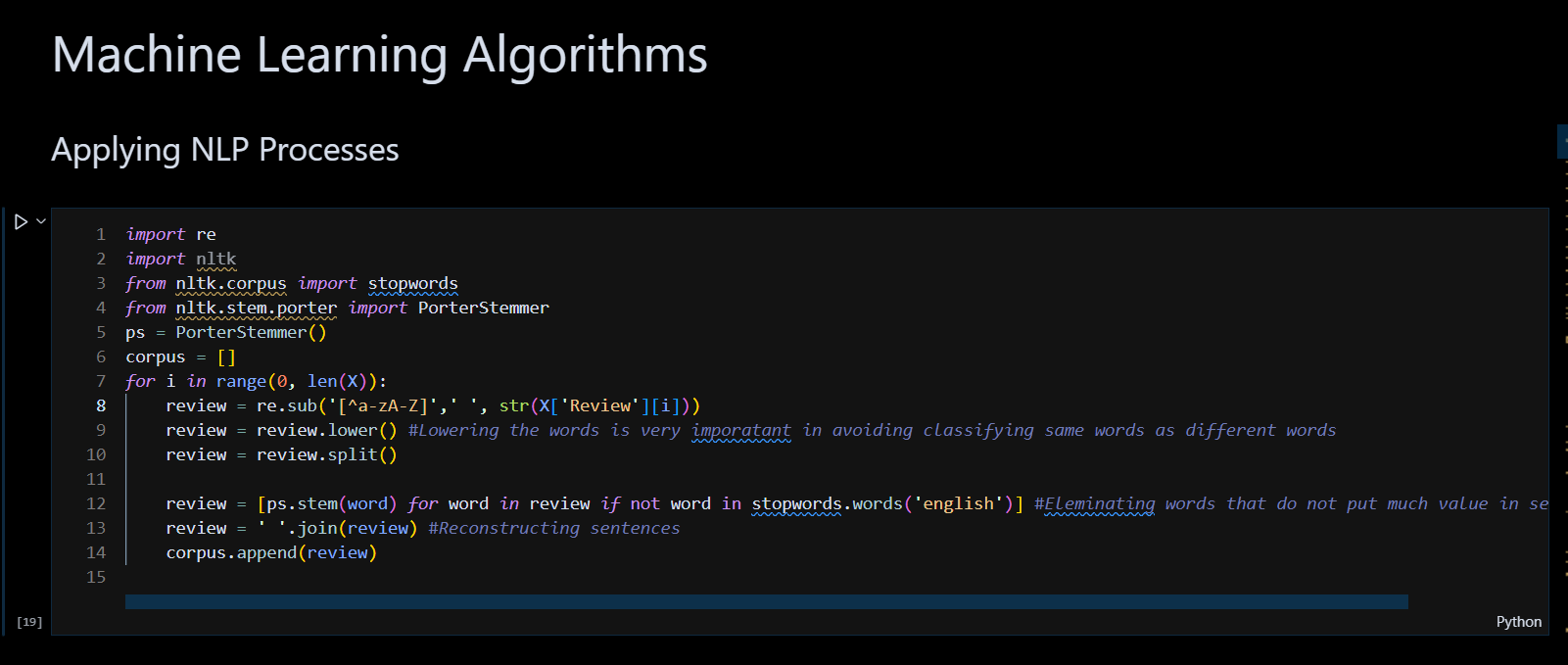
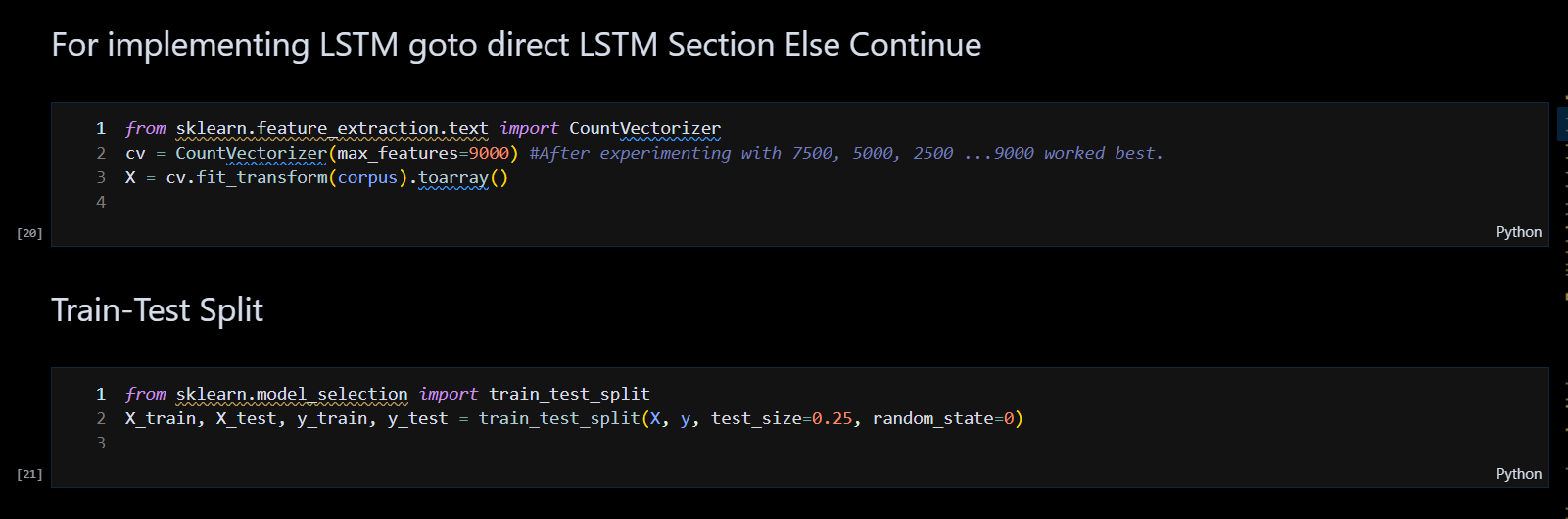
pickle.dump(classifier,open('model.pkl','wb'))

pickle.dump(cv,open('cv-model.pkl','wb'))

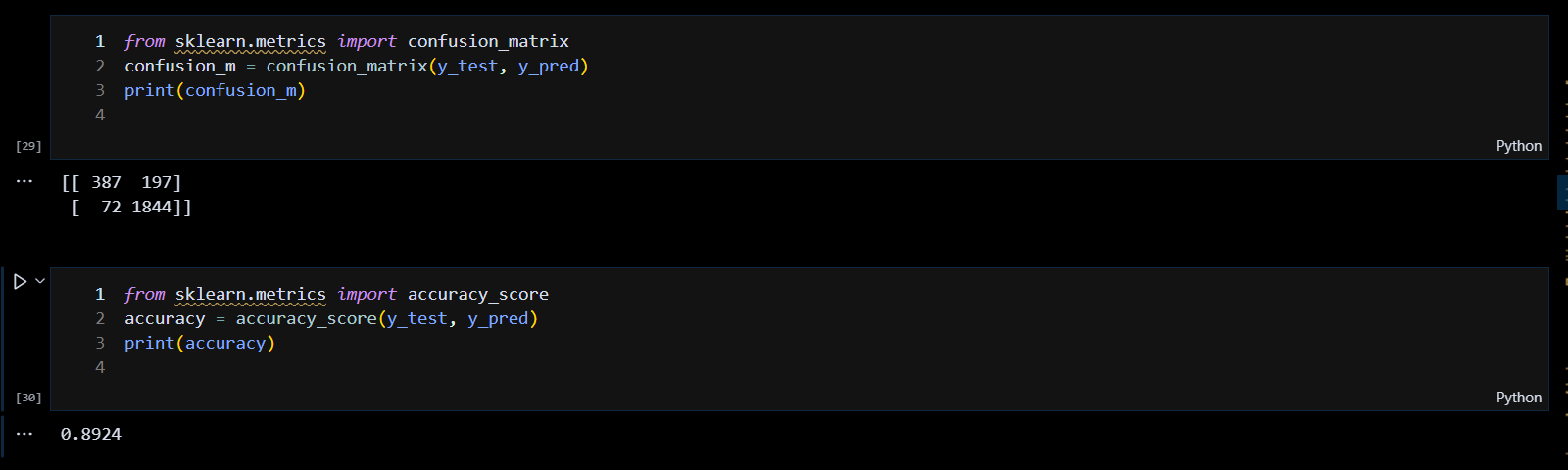
The output of this code is the pickle file, which is the used to make the model or we can say, it helps us to create an interface model which is used to integrate it in the frontend of the project

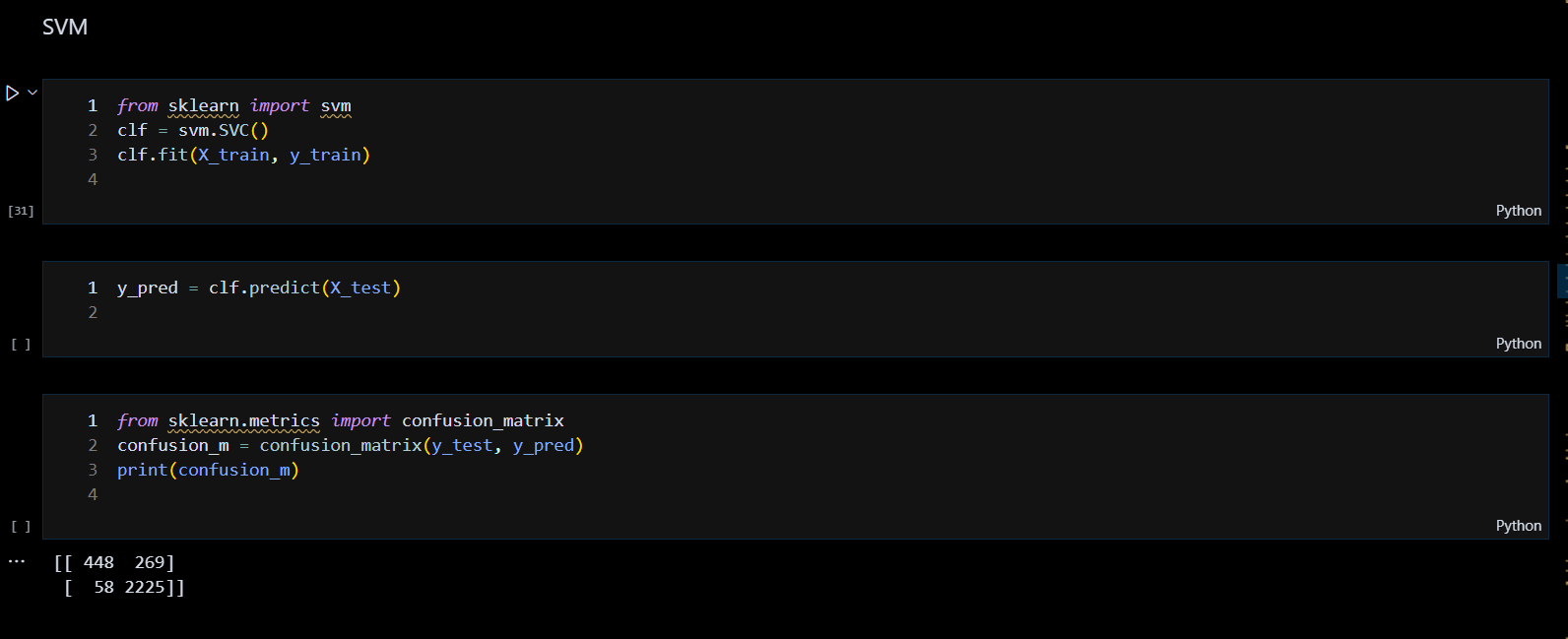


1. **Source code:**

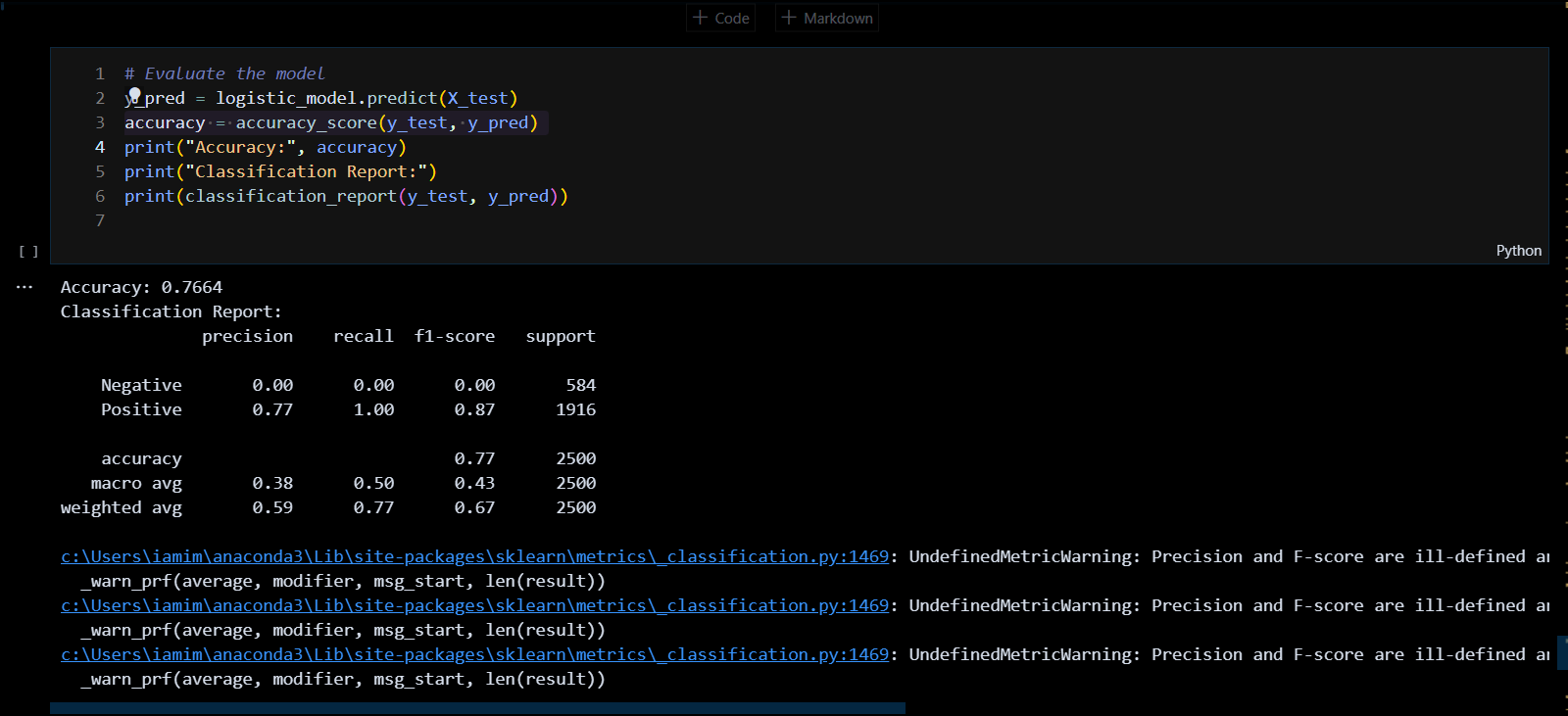




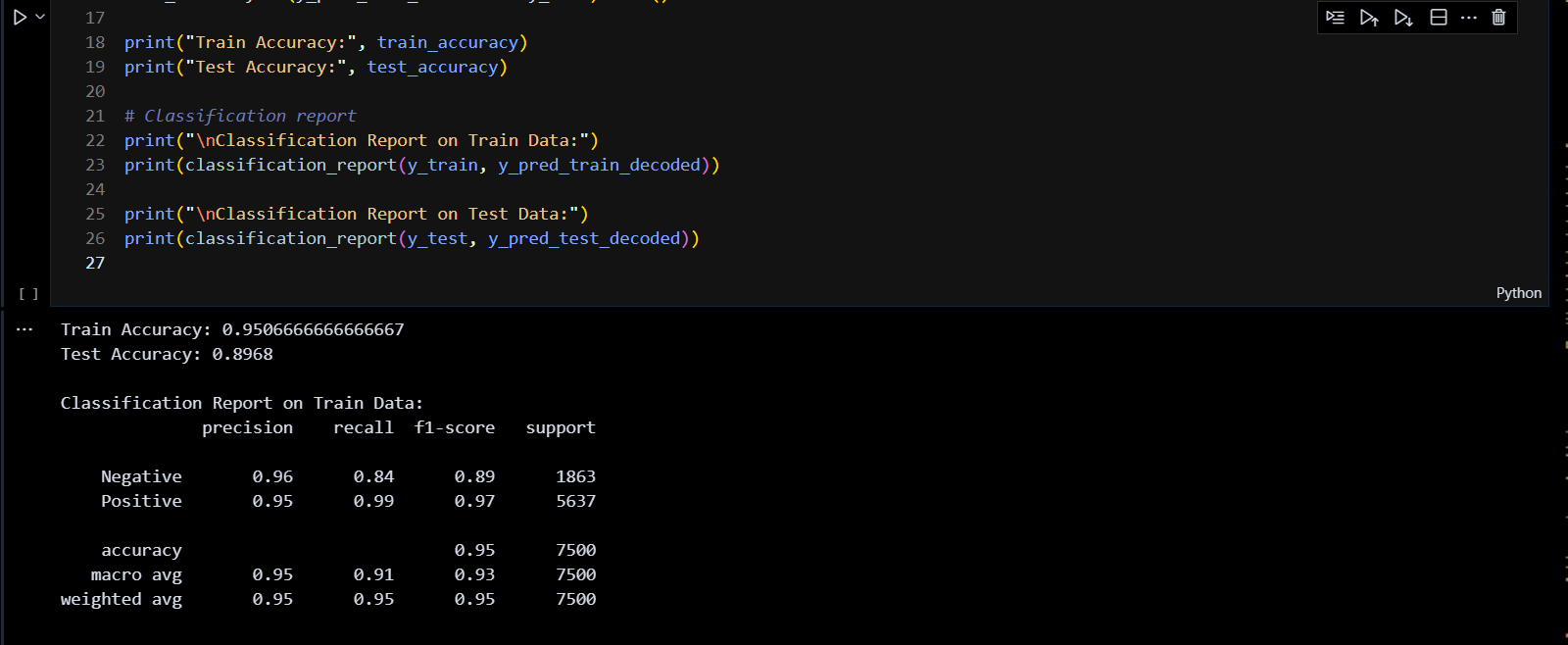


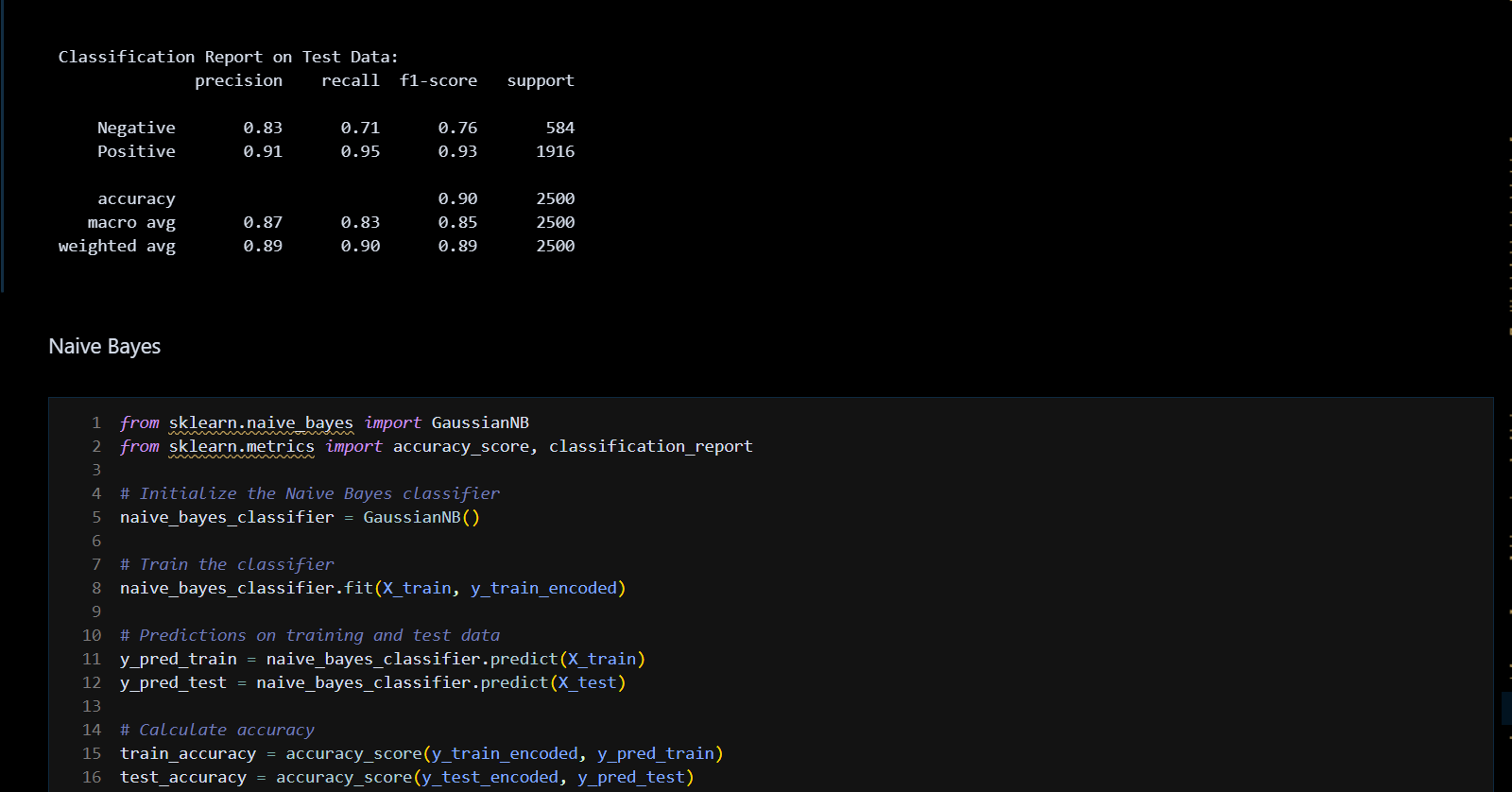


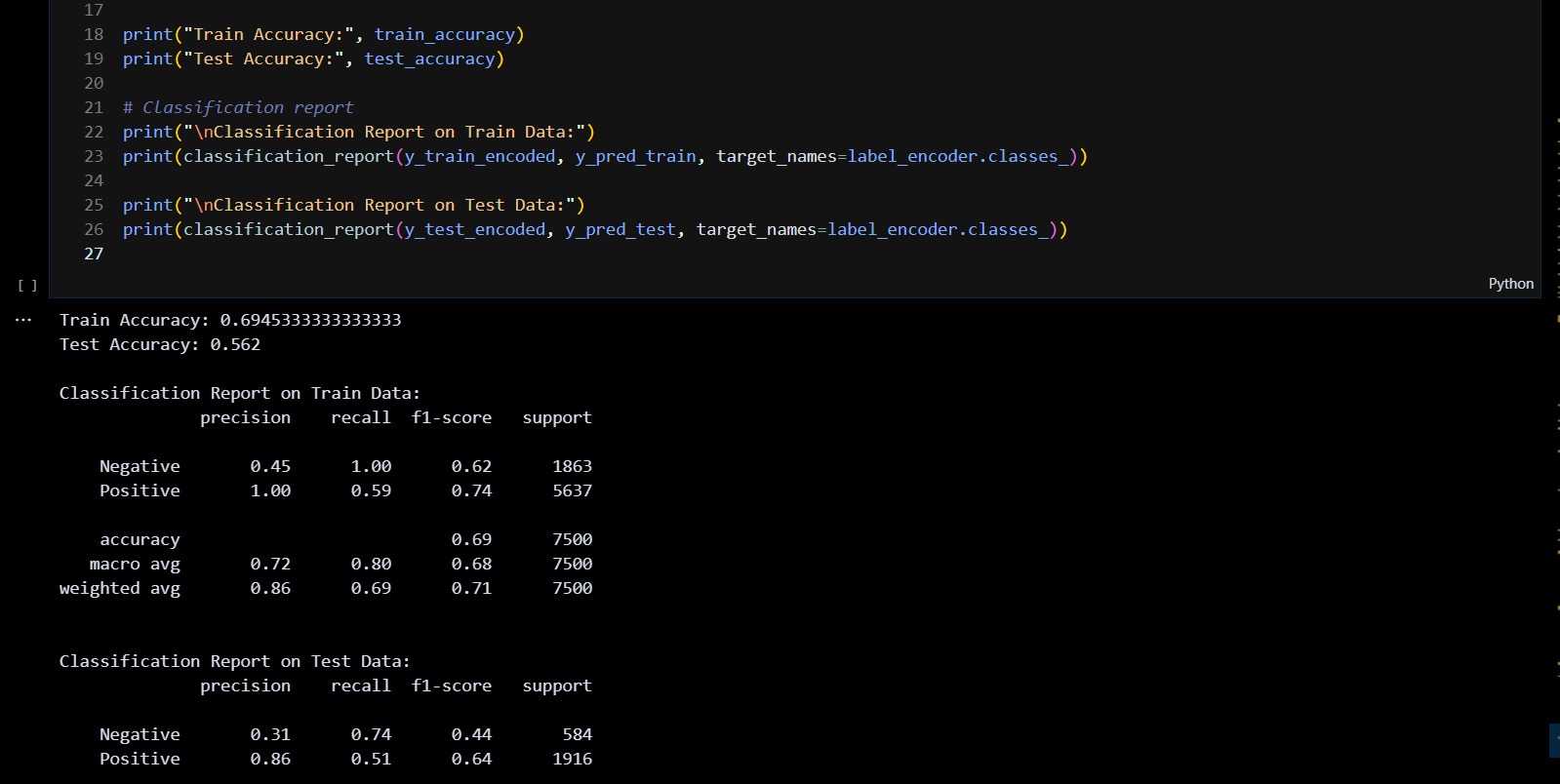












1. Model Evaluation and Model Testing

Confusion matrix:

*from* sklearn.preprocessing *import* LabelEncoder

# *Initialize LabelEncoder*

label\_encoder = LabelEncoder()

# *Fit LabelEncoder to your target variable and transform it*

y\_train\_encoded = label\_encoder.fit\_transform(y\_train)

y\_test\_encoded = label\_encoder.transform(y\_test)

# *Now, initialize and train the XGBoost model*

model = XGBClassifier()

model.fit(X\_train, y\_train\_encoded)

# *Now make predictions on the test set*

predicted\_labels = model.predict(X\_test)

# *Generate confusion matrix*

cm = confusion\_matrix(y\_test\_encoded, predicted\_labels)

# *Define classes (assuming binary classification)*

classes = label\_encoder.classes\_

# *Plot confusion matrix*

plt.figure(figsize=(8, 6))

sns.set(font\_scale=1.2)  # *for label size*

sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",

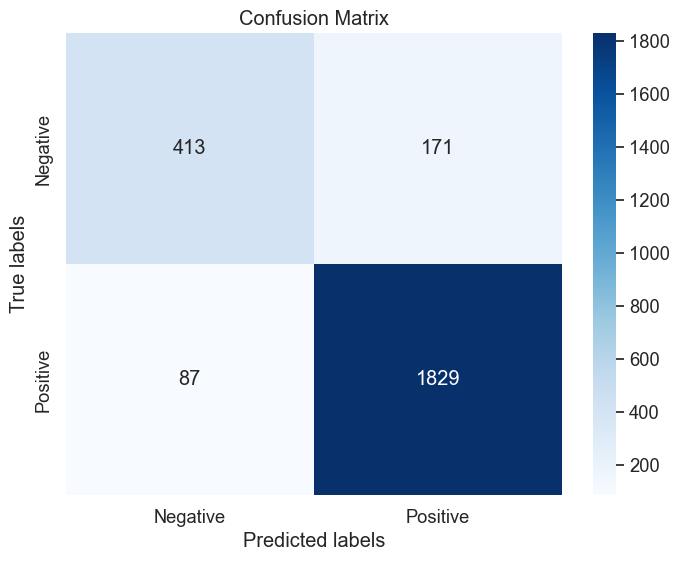
            xticklabels=classes, yticklabels=classes)

plt.xlabel('Predicted labels')

plt.ylabel('True labels')

plt.title('Confusion Matrix')

plt.show()



Roc Curve:

*from* sklearn.metrics *import* roc\_curve, roc\_auc\_score

# *Get predicted probabilities for positive class*

y\_prob = model.predict\_proba(X\_test)[:, 1]

# *Calculate ROC curve*

fpr, tpr, thresholds = roc\_curve(y\_test\_encoded, y\_prob)

# *Calculate AUC*

auc\_value = roc\_auc\_score(y\_test\_encoded, y\_prob)

# *Plot ROC curve*

plt.figure(figsize=(8, 6))

plt.plot(fpr, tpr, color='blue', lw=2,

         label='ROC curve (AUC = %0.2f)' % auc\_value)

plt.plot([0, 1], [0, 1], color='gray', linestyle='--')

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.05])

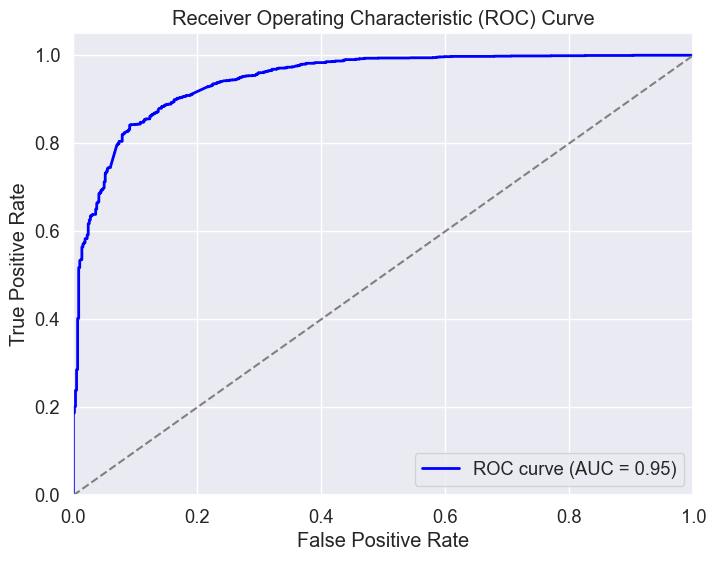
plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic (ROC) Curve')

plt.legend(loc='lower right')

plt.show()



An ROC (Receiver Operating Characteristic) curve graphically illustrates the

performance of a binary classifier. A 0.95 AUC (Area Under the Curve)

indicates strong discriminatory ability, where the model distinguishes between

classes with high accuracy. The curve plots true positive rate against false

positive rate, with higher AUC values indicating better classifier performance.

This metric is crucial in evaluating and comparing the effectiveness of various

classification models.

1. Conclusion

In this third phase of project to develop sentiment analysis model for restaurant

reviews a machine learning model was trained and evaluated based on pre processed

data.