Configuration Manual

MSc Research Project
Data Analytics

Vaibhav Sonia
Student ID: 22136860

School of Computing
National College of Ireland

Supervisor: Taimur Hafeez

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Vaibhav Sonia
Student ID:	22136860
Programme:	Data Analytics
Year:	2018
Module:	MSc Research Project
Supervisor:	Taimur Hafeez
Submission Due Date:	20/12/2018
Project Title:	Configuration Manual
Word Count:	591
Page Count:	9

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	14th December 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Vaibhav Sonia
22136860

1 Introduction

The following configuration manual illustrates the study of machine learning models and finding the best model that is fit for recommendations by using sentiment analysis and machine learning. Further, a manual will explain the software and hardware requirements that were used for the successful implementation of the project.

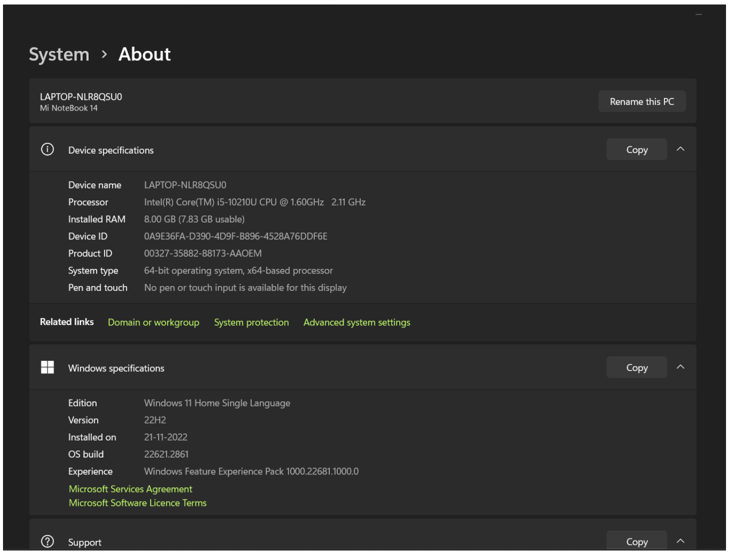
2 System Configuration

Below are the hardware and software attributes that were used for successful implementation of the project.

2.0.1 Hardware requirement

Table 1: Laptop Hardware Configurations

System	LAPTOP: NLR8QSU0
Operating System	Windows 11 Home Single Language
RAM	8 GB
Hard Disk	476.94 GB
Graphics Card	Intel(R) UHD Graphics
Processor	Intel(R) Core i5-10210U



System › About

LAPTOP-NLR8QSU0
Mi NoteBook 14

Device specifications

Device name	LAPTOP-NLR8QSU0
Processor	Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz 2.11 GHz
Installed RAM	8.00 GB (7.83 GB usable)
Device ID	0A9E36FA-D390-4D9F-B896-4528A76DDF6E
Product ID	00327-35882-88173-AAOEM
System type	64-bit operating system, x64-based processor
Pen and touch	No pen or touch input is available for this display

Windows specifications

Edition	Windows 11 Home Single Language
Version	22H2
Installed on	21-11-2022
OS build	22621.2861
Experience	Windows Feature Experience Pack 1000.22681.1000.0

Figure 1: Operating System Configurations

2.0.2 Software requirement

The software configurations used for implementations are as follows:
Software : Version Python : 3.9.13 (64 bits)

2.1 Project Implementation

2.1.1 Data Summary

The list below contains the data column summary and the data description of cosmetic brand Sephora.

Table 2: Product Data Content

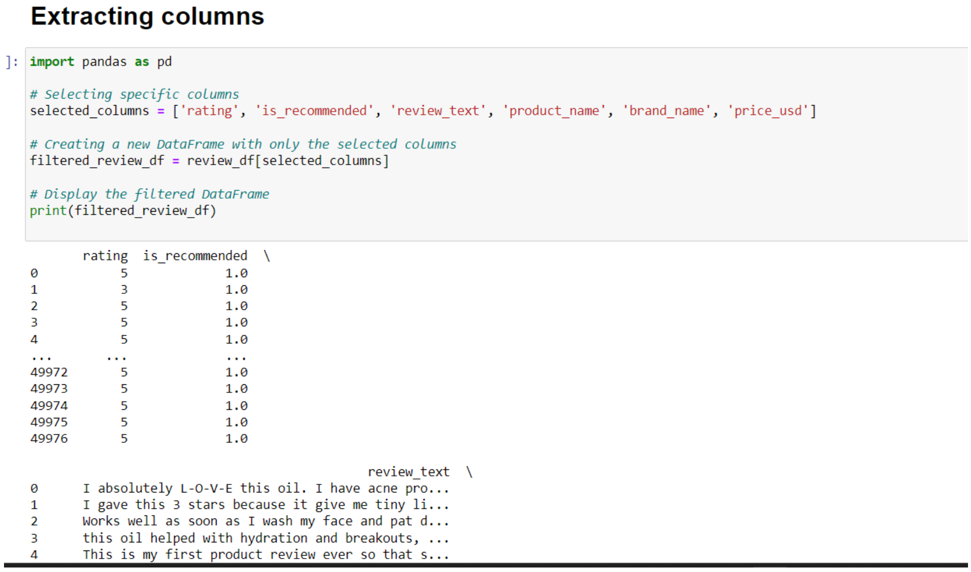
Column	**Description**
`product_id`	Unique identifier for the product from the site
`product_name`	Full name of the product
`brand_id`	Unique identifier for the product brand from the site
`brand_name`	Full name of the product brand
`rating`	Average rating of the product based on user reviews
`reviews`	Number of user reviews for the product
`price_usd`	Price of the product in US dollars

2.1.2 Data Preparation

After loading the csv files, the data had many unwanted elements. Figure 2 illustrates the extraction.

Table 3: Reviews Window

Column	Description
author_id	Unique identifier for the author of the review on the website
rating	Rating given by the author for the product on a scale of 1 to 5
is_recommended	Indicates if the author recommends the product or not (1-true, 0-false)
review_text	Main text of the review written by the author
review_title	Title of the review written by the author
skin_tone	Author's skin tone (e.g., fair, tan, etc.)
eye_color	Author's eye color (e.g., brown, green, etc.)
skin_type	Author's skin type (e.g., combination, oily, etc.)
hair_color	Author's hair color (e.g., brown, auburn, etc.)
product_id	Unique identifier for the product on the website



Extracting columns

```
]: import pandas as pd

   # Selecting specific columns
   selected_columns = ['rating', 'is_recommended', 'review_text', 'product_name', 'brand_name', 'price_usd']

   # Creating a new DataFrame with only the selected columns
   filtered_review_df = review_df[selected_columns]

   # Display the filtered DataFrame
   print(filtered_review_df)
```

```
       rating  is_recommended  \
0           5             1.0
1           3             1.0
2           5             1.0
3           5             1.0
4           5             1.0
...       ...             ...
49972       5             1.0
49973       5             1.0
49974       5             1.0
49975       5             1.0
49976       5             1.0

                                             review_text  \
0      I absolutely L-O-V-E this oil. I have acne pro...
1      I gave this 3 stars because it give me tiny li...
2      Works well as soon as I wash my face and pat d...
3      this oil helped with hydration and breakouts, ...
4      This is my first product review ever so that s...
```

Figure 2: Extracting the required columns

2.1.3 Data Pre-processing

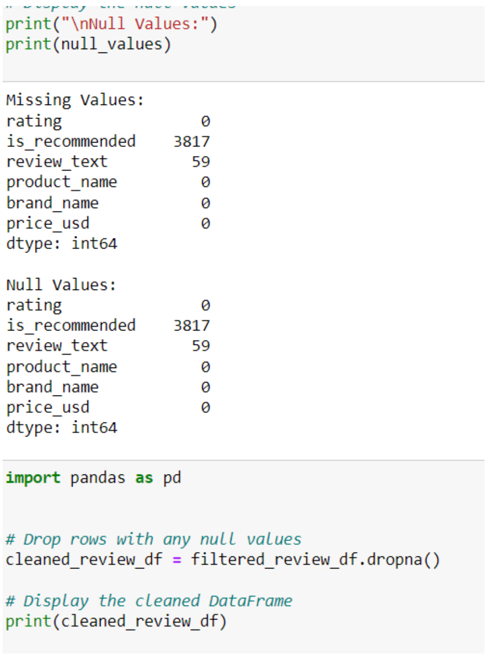
The dataset has many unwanted entries; hence, they are removed. Figure 3 illustrates the pre processing part of removing unwanted entries

The data has values which had different dataframes.They were corrected to perform further computations. Figure 4 illustrates the dataframe arrangement.

The sentiment value of review texts were extracted in order to understand the sentiment approach of dataset. Figure 5 represents assigning of sentiment value.

As it can be observed , there were entries in negative which is not the best fit for further computation or modeling. Hence the entries were scaled from 1-10 and drawn for computation. Figure 6 shows the scaling of entries.

After cleaning and scaling, the dataframe was ready for modeling. Feature selection enabled the modeling to perform with high accuracy and computing. The features such as 'scaled_sentiment_score' ,'rating' and 'recommended'. Figure 7 shows entries that are observed to be computed.

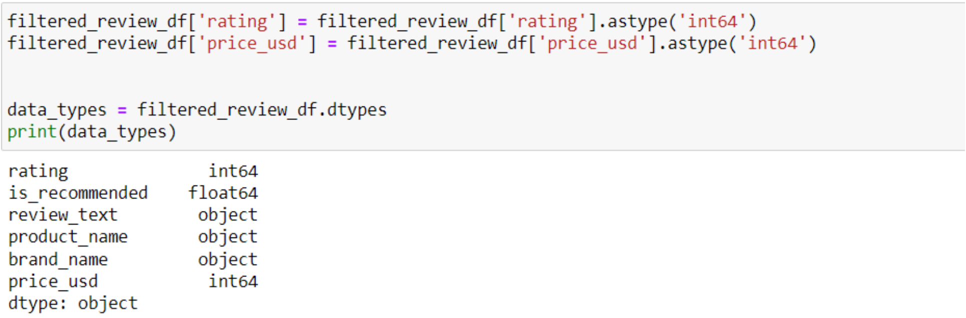


```
# Display the null values
print("\nNull Values:")
print(null_values)
```

```
Missing Values:
rating              0
is_recommended   3817
review_text        59
product_name        0
brand_name          0
price_usd           0
dtype: int64

Null Values:
rating              0
is_recommended   3817
review_text        59
product_name        0
brand_name          0
price_usd           0
dtype: int64
```

```
import pandas as pd


# Drop rows with any null values
cleaned_review_df = filtered_review_df.dropna()

# Display the cleaned DataFrame
print(cleaned_review_df)
```

Figure 3: Eliminating null values



```
filtered_review_df['rating'] = filtered_review_df['rating'].astype('int64')
filtered_review_df['price_usd'] = filtered_review_df['price_usd'].astype('int64')


data_types = filtered_review_df.dtypes
print(data_types)
```

```
rating              int64
is_recommended    float64
review_text        object
product_name       object
brand_name         object
price_usd           int64
dtype: object
```

Figure 4: Enlisting the dataframe as per requirement

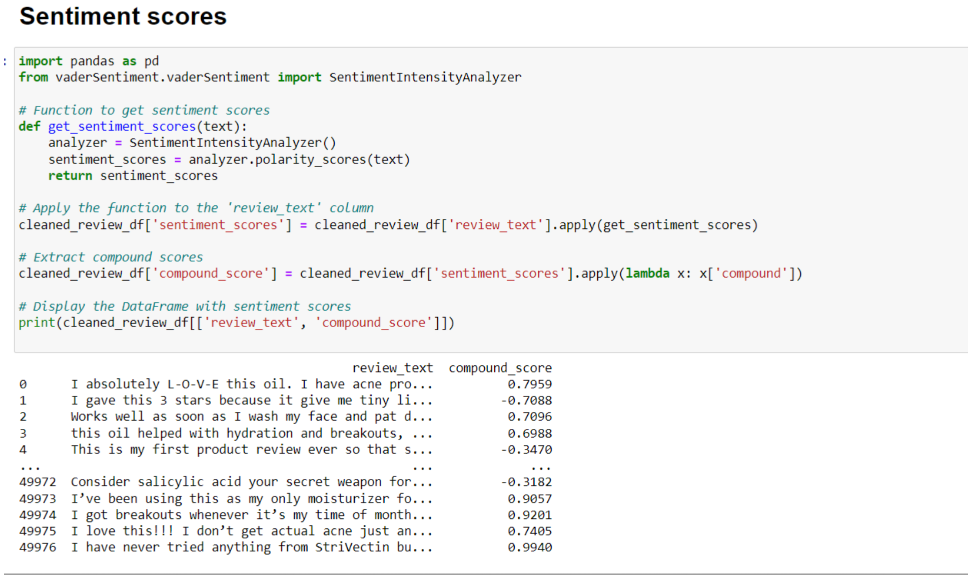
2.2 Model Building

2.2.1 Design specification

The design specification is drawn and shown in figure 8.

2.2.2 Linear regression

Figure 9 illustrates linear regression.



Sentiment scores

```
import pandas as pd
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

# Function to get sentiment scores
def get_sentiment_scores(text):
    analyzer = SentimentIntensityAnalyzer()
    sentiment_scores = analyzer.polarity_scores(text)
    return sentiment_scores

# Apply the function to the 'review_text' column
cleaned_review_df['sentiment_scores'] = cleaned_review_df['review_text'].apply(get_sentiment_scores)

# Extract compound scores
cleaned_review_df['compound_score'] = cleaned_review_df['sentiment_scores'].apply(lambda x: x['compound'])

# Display the DataFrame with sentiment scores
print(cleaned_review_df[['review_text', 'compound_score']])
```

```
                                             review_text  compound_score
0      I absolutely L-O-V-E this oil. I have acne pro...          0.7959
1      I gave this 3 stars because it give me tiny li...         -0.7088
2      Works well as soon as I wash my face and pat d...          0.7096
3      this oil helped with hydration and breakouts, ...          0.6988
4      This is my first product review ever so that s...         -0.3470
...                                                  ...             ...
49972  Consider salicylic acid your secret weapon for...         -0.3182
49973  I've been using this as my only moisturizer fo...          0.9057
49974  I got breakouts whenever it's my time of month...          0.9201
49975  I love this!!! I don't get actual acne just an...          0.7405
49976  I have never tried anything from StriVectin bu...          0.9940
```

Figure 5: Assigning sentiment value

2.2.3 Logistic Regression

Figure 10 illustrates the deployment of logistic regression.

2.2.4 Decision Trees

Figure 11 displays applying of decision trees.

2.2.5 Support Vector Classification

Figure 12 shows Support vector classification.

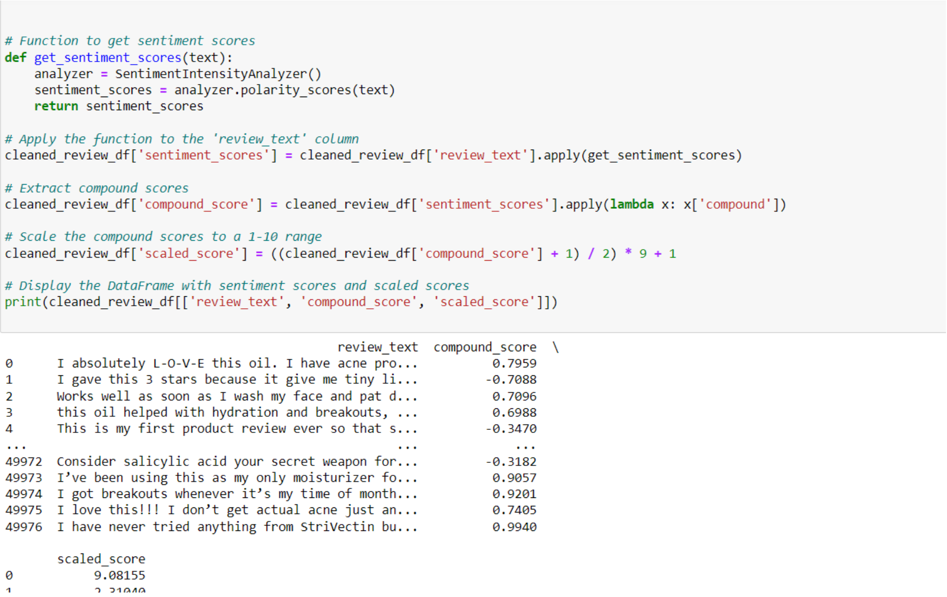
2.2.6 Naive Bayes

Figure 13 shows the execution of Naive Bayes.

3 Comparative Analysis

The performance of algorithm was observed and the models performed will be evaluated by accuracy, precision , recall and F1- score.

The study made many inferences. The execution of linear regression did not meet any expectations since the low mean square error of 0.0035 was not significant for a conclusion. Logistic regression performed with an accuracy of 96.46%, indicating a good threshold. The performance was supported by precision and recall scores, respectively. The decision



```
# Function to get sentiment scores
def get_sentiment_scores(text):
    analyzer = SentimentIntensityAnalyzer()
    sentiment_scores = analyzer.polarity_scores(text)
    return sentiment_scores

# Apply the function to the 'review_text' column
cleaned_review_df['sentiment_scores'] = cleaned_review_df['review_text'].apply(get_sentiment_scores)

# Extract compound scores
cleaned_review_df['compound_score'] = cleaned_review_df['sentiment_scores'].apply(lambda x: x['compound'])

# Scale the compound scores to a 1-10 range
cleaned_review_df['scaled_score'] = ((cleaned_review_df['compound_score'] + 1) / 2) * 9 + 1

# Display the DataFrame with sentiment scores and scaled scores
print(cleaned_review_df[['review_text', 'compound_score', 'scaled_score']])
```

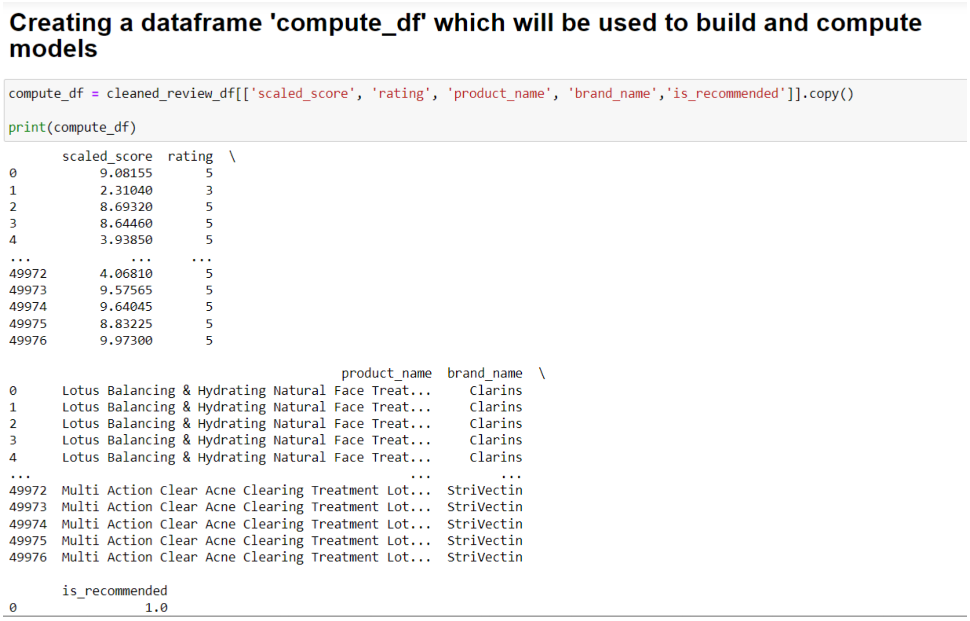
```
                                             review_text  compound_score  \
0      I absolutely L-O-V-E this oil. I have acne pro...          0.7959   
1      I gave this 3 stars because it give me tiny li...         -0.7088   
2      Works well as soon as I wash my face and pat d...          0.7096   
3      this oil helped with hydration and breakouts, ...          0.6988   
4      This is my first product review ever so that s...         -0.3470   
...                                                  ...             ...   
49972  Consider salicylic acid your secret weapon for...         -0.3182   
49973  I’ve been using this as my only moisturizer fo...          0.9057   
49974  I got breakouts whenever it’s my time of month...          0.9201   
49975  I love this!!! I don’t get actual acne just an...          0.7405   
49976  I have never tried anything from StriVectin bu...          0.9940   

       scaled_score  
0           9.08155  
1           2.31040  
```

Figure 6: Scaling sentiment value

Algorithm	Accuracy (%)	Precision	Recall	F1-score
Logistic Regression	96.46	0.99	0.96	0.98
Decision Tree	95.33	0.95	0.95	0.95
Support Vector Classification	96.46	0.97	0.96	0.97
Naive Bayes	80.93	0.66	0.81	0.72

Table 4: Performance Metrics of Classification Algorithms

tree performed with 95.33% accuracy, which is a good interpretation for prediction, and considering the ability to handle complex data and relationships, the decision tree performed well. Support vector classification and Naive Bayes have performed well with accuracy of 96.46%. Overall, the models performed well in predicting recommendations, which have business as well as machine learning implications, with business implications helping with revenue and machine learning implications helping as a guiding tool.



Creating a dataframe 'compute_df' which will be used to build and compute models

```
compute_df = cleaned_review_df[['scaled_score', 'rating', 'product_name', 'brand_name','is_recommended']].copy()

print(compute_df)
```

```
       scaled_score  rating  \
0           9.08155       5
1           2.31040       3
2           8.69320       5
3           8.64460       5
4           3.93850       5
...             ...     ...
49972       4.06810       5
49973       9.57565       5
49974       9.64045       5
49975       8.83225       5
49976       9.97300       5

                                            product_name  brand_name  \
0      Lotus Balancing & Hydrating Natural Face Treat...     Clarins
1      Lotus Balancing & Hydrating Natural Face Treat...     Clarins
2      Lotus Balancing & Hydrating Natural Face Treat...     Clarins
3      Lotus Balancing & Hydrating Natural Face Treat...     Clarins
4      Lotus Balancing & Hydrating Natural Face Treat...     Clarins
...                                                  ...         ...
49972  Multi Action Clear Acne Clearing Treatment Lot...  StriVectin
49973  Multi Action Clear Acne Clearing Treatment Lot...  StriVectin
49974  Multi Action Clear Acne Clearing Treatment Lot...  StriVectin
49975  Multi Action Clear Acne Clearing Treatment Lot...  StriVectin
49976  Multi Action Clear Acne Clearing Treatment Lot...  StriVectin

       is_recommended
0                 1.0
```

Figure 7: Feature selection and displaying

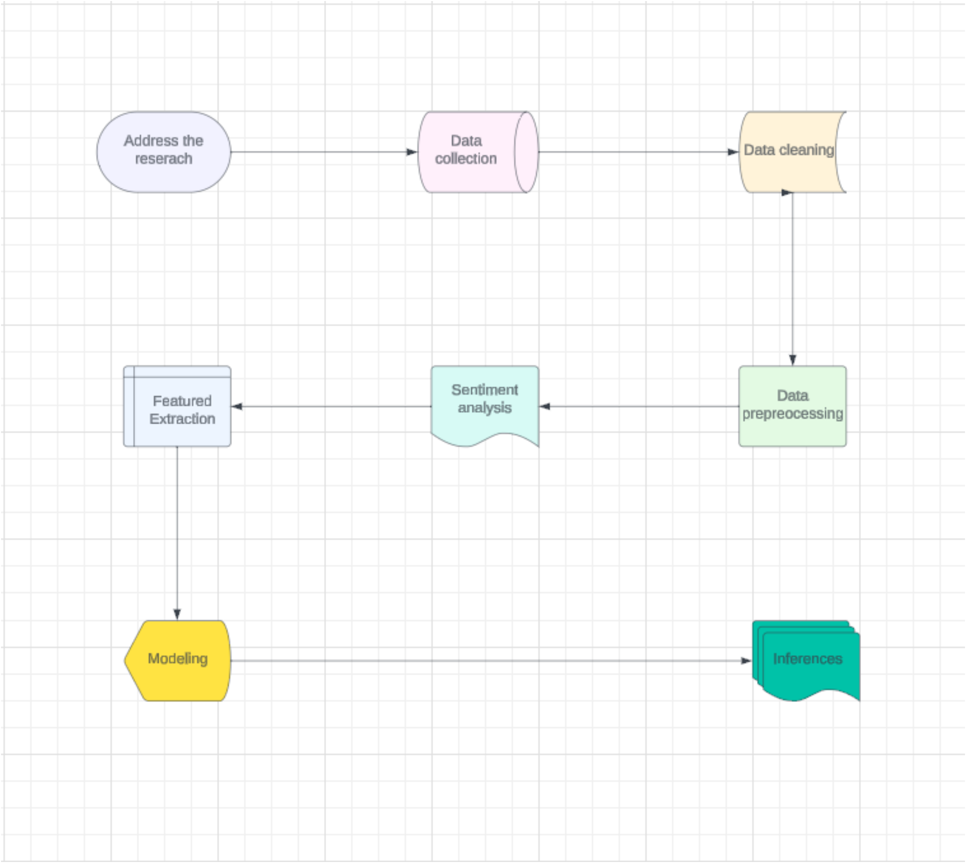
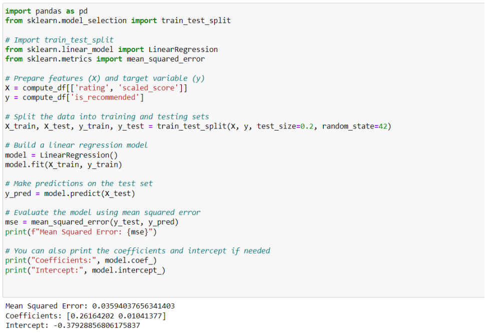


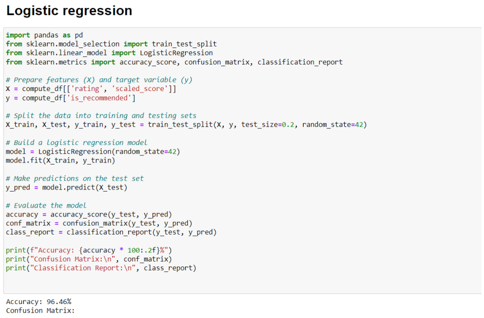
Figure 8: Work flowchart



```
import pandas as pd
from sklearn.model_selection import train_test_split

# Import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

# Prepare features (X) and target variable (y)
X = compute_df[['rating', 'scaled_score']]
y = compute_df['is_recommended']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Build a linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the model using mean squared error
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")

# You can also print the coefficients and intercept if needed
print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)
```

```
Mean Squared Error: 0.03594037656341403
Coefficients: [0.26164202 0.01041377]
Intercept: -0.37928856806175837
```

Figure 9: Linear regression

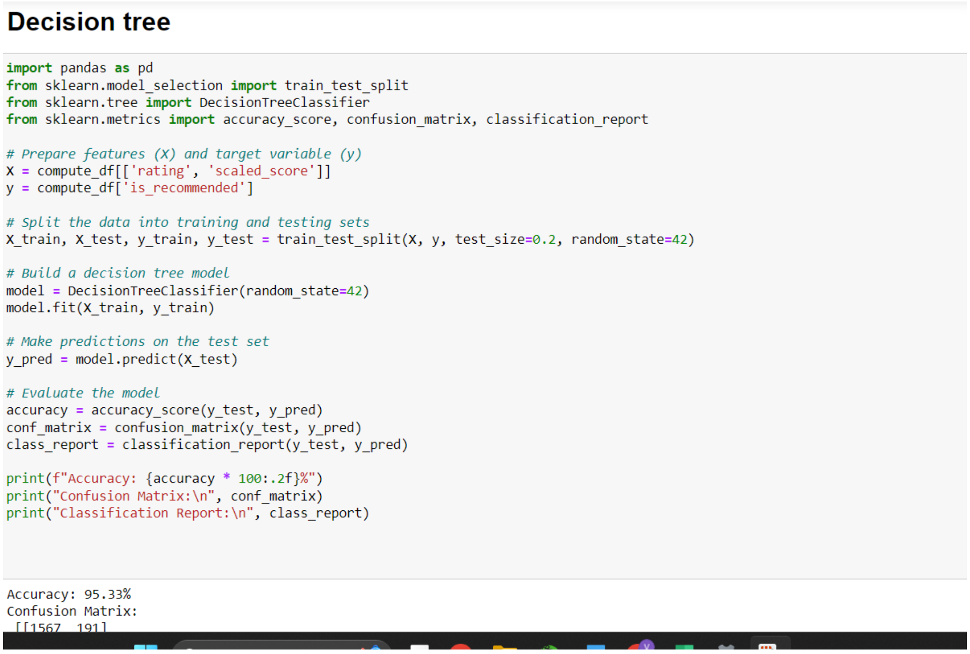


Logistic regression

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

# Prepare features (X) and target variable (y)
X = compute_df[['rating', 'scaled_score']]
y = compute_df['is_recommended']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Build a logistic regression model
model = LogisticRegression(random_state=42)
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)

print(f"Accuracy: {accuracy * 100:.2f}%")
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", class_report)
```

```
Accuracy: 96.46%
Confusion Matrix:
```

Figure 10: Logistic Regression



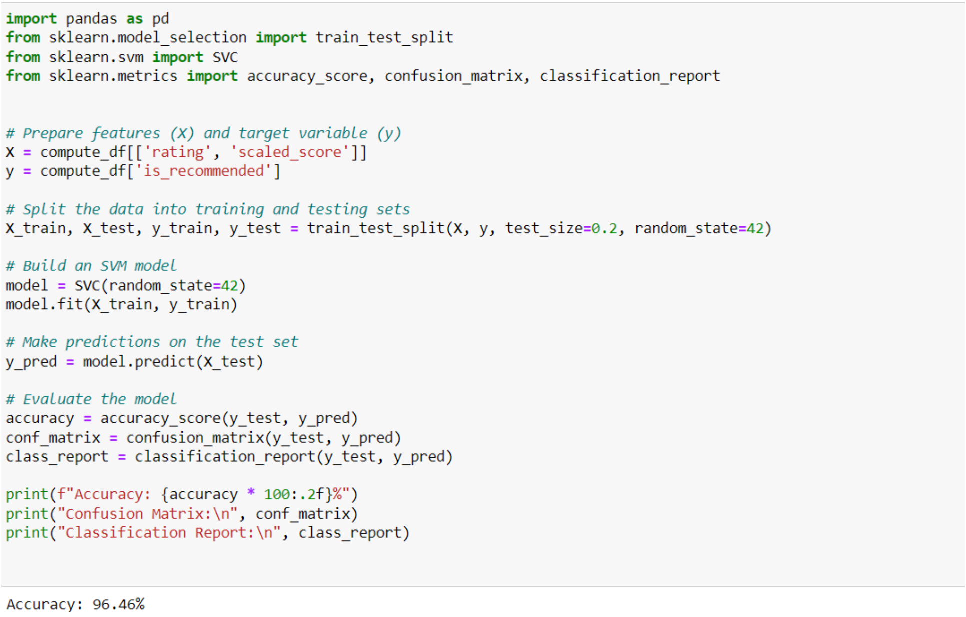
Decision tree

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

# Prepare features (X) and target variable (y)
X = compute_df[['rating', 'scaled_score']]
y = compute_df['is_recommended']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Build a decision tree model
model = DecisionTreeClassifier(random_state=42)
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)

print(f"Accuracy: {accuracy * 100:.2f}%")
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", class_report)
```

```
Accuracy: 95.33%
Confusion Matrix:
 [[1567  191]
```

Figure 11: Decision Trees

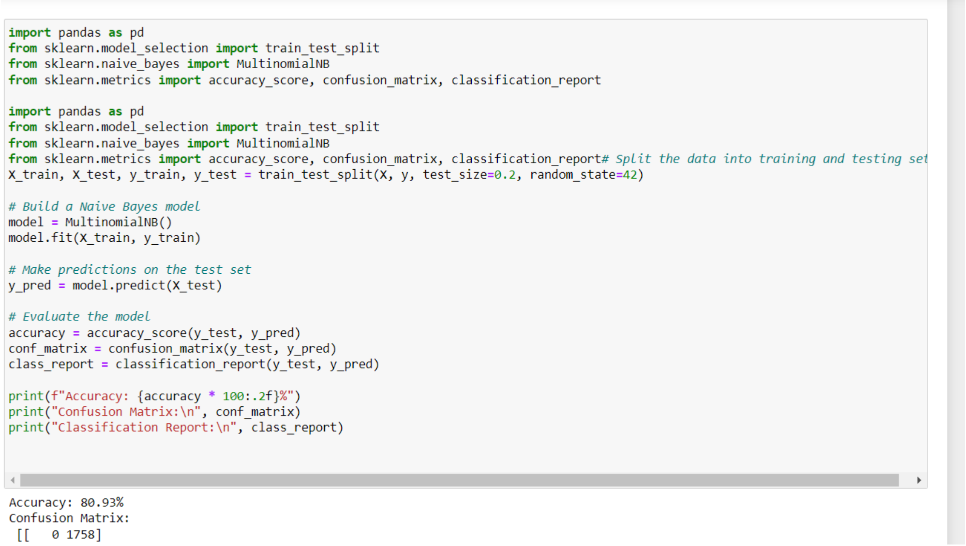
Support vector classification



```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report


# Prepare features (X) and target variable (y)
X = compute_df[['rating', 'scaled_score']]
y = compute_df['is_recommended']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Build an SVM model
model = SVC(random_state=42)
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)

print(f"Accuracy: {accuracy * 100:.2f}%")
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", class_report)
```

```
Accuracy: 96.46%
```

Figure 12: Support Vector Classification



```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report# Split the data into training and testing set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Build a Naive Bayes model
model = MultinomialNB()
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)

print(f"Accuracy: {accuracy * 100:.2f}%")
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", class_report)
```

```
Accuracy: 80.93%
Confusion Matrix:
 [[    0 1758]
```

Figure 13: Naive Bayes