
CUSTOMER
FEEDBACK

OUTLINE:

01. Introduction
02. GOAL
03. Questions that related to the goal.
04. Correlation heatmap.
05. Data Preprocessing
06. The Algorithm & model we used
07. Analysis & results
08. Conclusion

INTRODUCTION

- This Dataset contains of reviews covering various services and products, such as movies, food, and websites. These reviews are categorized into positive and negative sentiments.

G O A L

- we has been selected so as to make it easier for decision makers to know the customer's opinions by analyzing their responses and developing the product to satisfy the customer, The data we used is from Kaggel.

WE WILL TRY TO ANSWER THE QUESTION

- Were most comments positive or negative?
- Is there a comment consisting of a negative and positive part at the same time?
- does the customers feedback get affected by their location/source/day of the week?

ANALYSIS AND RESULTS:

THE TOOLS AND LIBRARIES WE USED:

```
"""

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
from nltk.corpus import stopwords
stop=set(stopwords.words('english'))
from nltk.util import ngrams
from nltk.tokenize import word_tokenize
from sklearn.feature_extraction.text import CountVectorizer
from collections import defaultdict
from collections import Counter
plt.style.use('ggplot')
import re
import string
import warnings
warnings.filterwarnings('ignore')
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

import seaborn as sns
import matplotlib.pyplot as plt
```

PRE-PROCESSING

DATA CLEANING:

```
#_____________________________________________________________________________
#loading the data set
data=pd.read_csv('sentiment-analysis.csv')
data.head()
#separating the data into columns
data = data['Text, Sentiment, Source, Date/Time, User ID, Location, Confidence Score'].str.split(', ', expand=True)
data.columns = ['Text', 'Sentiment', 'Source', 'Date/Time', 'User ID', 'Location', 'Confidence Score']
print('There are {} rows and {} columns in data.'.format(data.shape[0], data.shape[1]))

data.head(10)
```

```
#Data Cleaning
data.isnull().sum()
data.dropna(inplace=True)
data.head(10)

#there are some leading and trailing spaces in 'Date/Time' column, so let's trim them
data['Date/Time'] = data['Date/Time'].str.strip()

data[['Date', 'Time']] = data['Date/Time'].str.split(' ', expand=True)

data.drop(columns=['Date/Time'], inplace=True)
data['Date'] = pd.to_datetime(data['Date'])
data['Time'] = pd.to_datetime(data['Time'], format='%H:%M:%S').dt.time
```

PRE-PROCESSING

DATA CLEANING:

data - DataFrame

Index	timent, Source, Date/Time, User ID, Location, Confiden
0	"I love this product!", Positive, Twitter, 202…
1	"The service was terrible.", Negative, Yelp Re…
2	"This movie is amazing!", Positive, IMDb, 2023…
3	"I'm so disappointed with their customer suppo…
4	"Just had the best meal of my life!", Positive…
5	"The quality of this product is subpar.", Nega…
6	"I can't stop listening to this song. It's inc…
7	"Their website is so user-friendly. Love it!",…
8	"I loved the movie! It was fantastic!", Positi…
9	"The customer service was terrible.", Negative…
10	"This book made me feel inspired. Highly recom…
11	"I'm extremely disappointed with their product…
12	"Just had the most amazing vacation! I can't w…
13	"The food at this restaurant was awful. Never …

data - DataFrame

Index	Text	Sentiment	Source	User ID	Location	fidence S	Date	Time
0	"I love this product!"	Positive	Twitter	@user123	New York	0.85	2023-06-15 00:00:00	09:23:14
1	"The service was terrible."	Negative	Yelp Reviews	user456	Los Angeles	0.65	2023-06-15 00:00:00	11:45:32
2	"This movie is amazing!"	Positive	IMDb	moviefan789	London	0.92	2023-06-15 00:00:00	14:10:22
3	"I'm so disappointed with their customer suppo…	Negative	Online Forum	forumuser1	Toronto	0.78	2023-06-15 00:00:00	17:35:11
4	"Just had the best meal of my life!"	Positive	TripAdvisor	foodie22	Paris	0.88	2023-06-16 00:00:00	08:50:59
5	"The quality of this product is subpar."	Negative	Amazon Reviews	shopper123	San Francisco	0.72	2023-06-16 00:00:00	10:15:27
6	"I can't stop listening to this song. It's inc…	Positive	Spotify	musiclover456	Berlin	0.91	2023-06-16 00:00:00	13:40:18
7	"Their website is so user-friendly. Love it!"	Positive	Website Testimonial	testimonialuser1	Sydney	0.87	2023-06-16 00:00:00	16:05:36
8	"I loved the movie! It was fantastic!"	Positive	IMDb	user123	New York	0.92	2023-07-02 00:00:00	09:12:34
9	"The customer service was terrible."	Negative	Yelp Reviews	user456	Los Angeles	0.65	2023-07-02 00:00:00	10:45:21
10	"This book made me feel inspired. Highly recom…	Positive	Goodreads	bookworm789	London	0.88	2023-07-02 00:00:00	12:34:56
11	"I'm extremely disappointed with their product…	Negative	Online Store	shopper789	San Francisco	0.72	2023-07-02 00:00:00	15:21:43
12	"Just had the most amazing vacation! I can't w…	Positive	TripAdvisor	travelenthusiast1	Sydney	0.93	2023-07-02 00:00:00	18:01:23
13	"The food at this restaurant was awful. Never …	Negative	Zomato	foodlover123	Mumbai	0.55	2023-07-02 00:00:00	20:45:37
14	"I can't stop listening to this song. It's my …	Positive	Spotify	musiclover789	Berlin	0.91	2023-07-03 00:00:00	09:17:52
15	"Their website is so confusing and poorly desi…	Negative	Website Review	user789	Toronto	0.68	2023-07-03 00:00:00	11:59:18
16	"I had an incredible experience at the theme p…	Positive	Trip Report	thrillseeker1	Orlando	0.89	2023-07-03 00:00:00	14:40:05
17	"The product arrived damaged. Very disappointe…	Negative	Online Store	buyer123	Chicago	0.76	2023-07-03 00:00:00	17:25:09
18	"The concert was absolutely breathtaking. Best…	Positive	Event Review	musicfan456	Paris	0.95	2023-07-03 00:00:00	20:15:33
19	"I had a terrible experience with their custom…	Negative	Online Chat	user1234	Sydney	0.61	2023-07-04 00:00:00	08:32:41
20	"This movie is a masterpiece! I was blown away…	Positive	IMDb	cinephile789	Los Angeles	0.93	2023-07-04 00:00:00	14:05:27
21	"The customer service at this store is top-not…	Positive	Yelp Reviews	shopper456	New York	0.82	2023-07-04 00:00:00	16:35:19
22	"I'm disappointed with the ending of this book…	Negative	Goodreads	booklover123	London	0.68	2023-07-04 00:00:00	19:18:53
23	"The product I received was damaged. Unaccepta…	Negative	Online Store	buyer789	San Francisco	0.75	2023-07-04 00:00:00	21:52:41
24	"Just had the worst flight experience. Delayed…	Negative	Airline Review	traveler456	Sydney	0.57	2023-07-05 00:00:00	09:10:15
25	"The hotel stay was absolutely amazing! Luxury…	Positive	TripAdvisor	traveladdict1	Paris	0.95	2023-07-05 00:00:00	12:30:08

BEFORE

AFTER

WE RANDOMLY USED 20% OF THE DATA AS THE TEST SET:

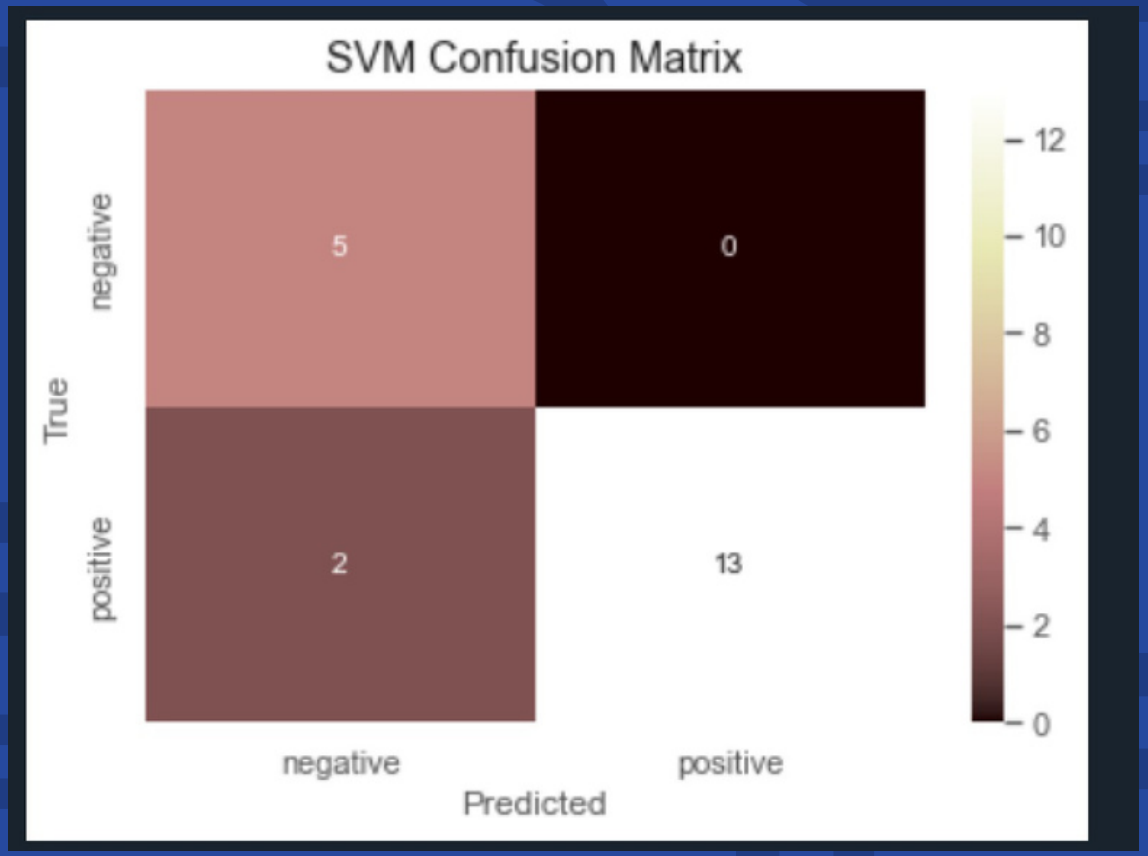
```
# Preparing the data
X = data['Text']
y = data['Sentiment']

# Splitting 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)

# Feature extraction using TF-IDF vectorizer
tfidf_vectorizer = TfidfVectorizer(max_features=5000)
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)
```

WHAT IS THE MODEL USED?

1. Support vector machine (SVM)
2. Random forest

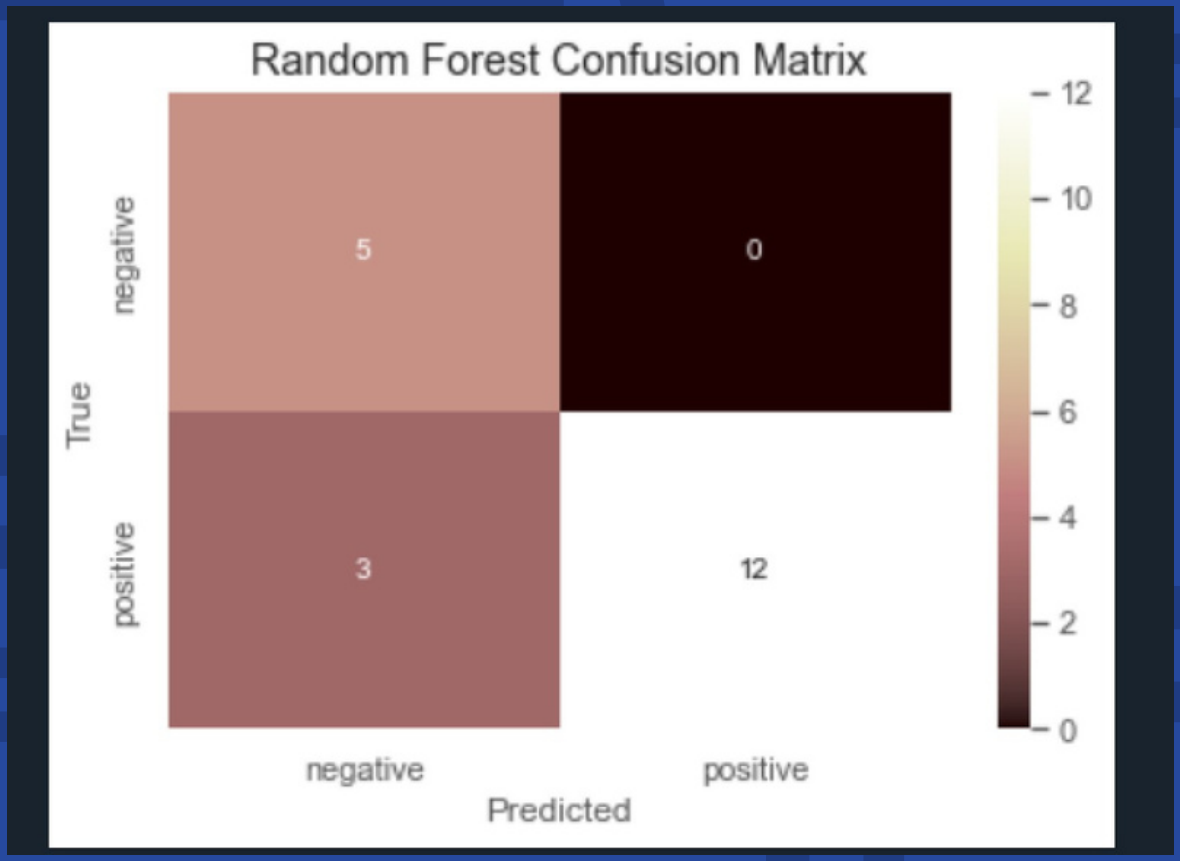
S V M :

```
There are 98 rows and 7 columns in data.
SVM Accuracy: 0.9

Classification Report for SVM:
              precision    recall  f1-score   support

    Negative       0.71      1.00      0.83         5
    Positive       1.00      0.87      0.93        15

    accuracy                           0.90        20
   macro avg       0.86      0.93      0.88        20
weighted avg       0.93      0.90      0.90        20

Confusion Matrix for SVM:
 [[ 5  0]
 [ 2 13]]
```

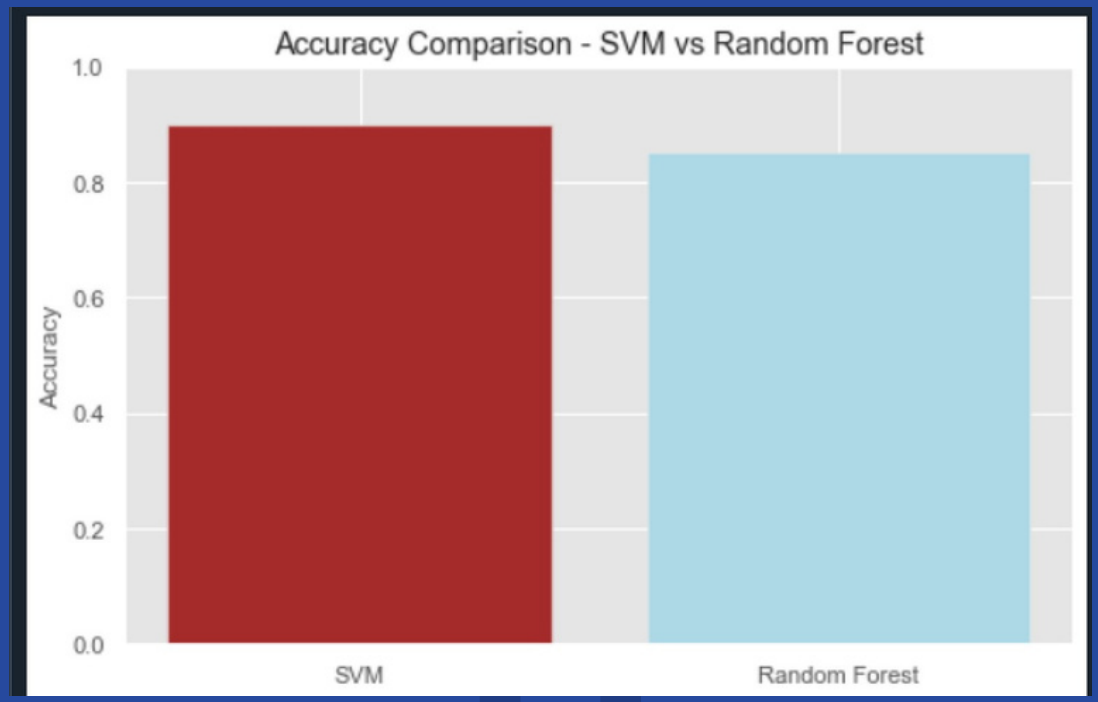


R F C :

```
Random Forest Accuracy: 0.85

Classification Report for Random Forest:
              precision    recall  f1-score   support

    Negative       0.62      1.00      0.77         5
    Positive       1.00      0.80      0.89        15

    accuracy                           0.85        20
   macro avg       0.81      0.90      0.83        20
weighted avg       0.91      0.85      0.86        20

Confusion Matrix for Random Forest:
 [[ 5  0]
 [ 3 12]]
```

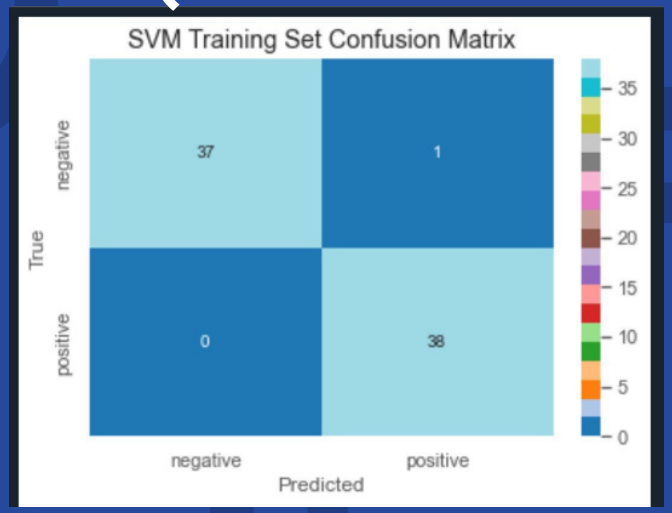


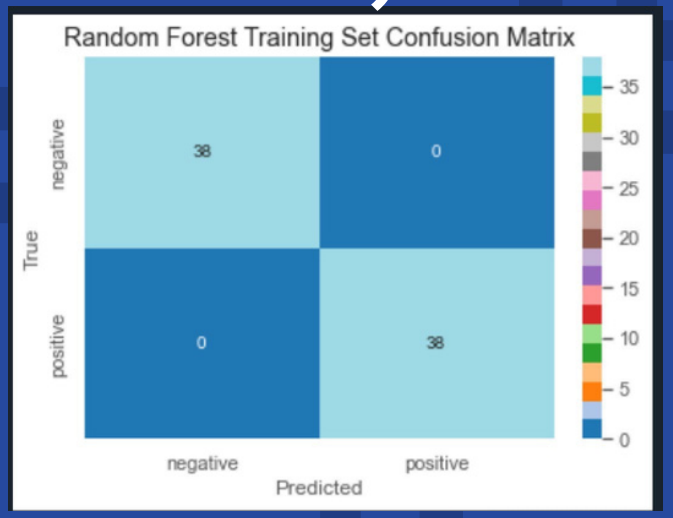
COMPARE MODELS (VISUALIZING):

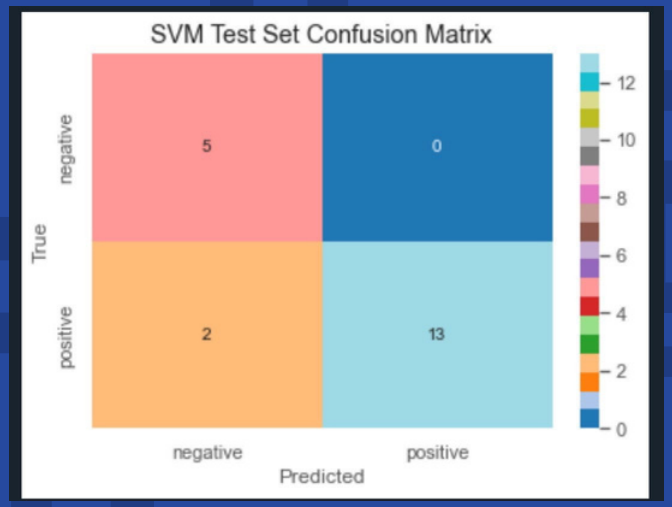


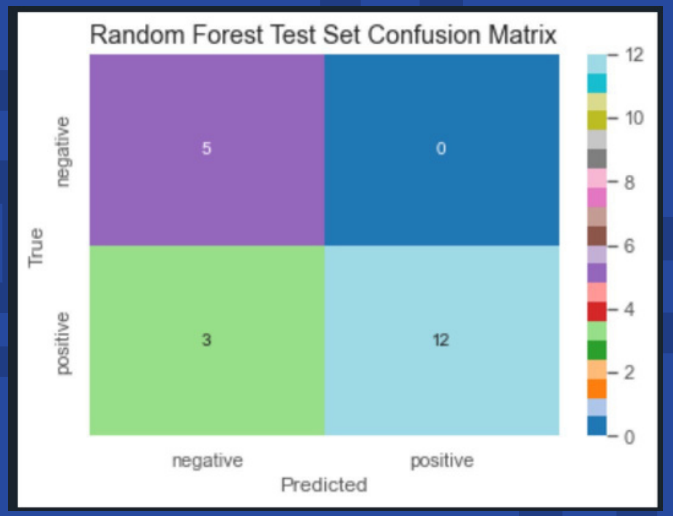
We can see on the left the SVM Accuracy is 0.9 (90%)
on the right for Random Forest the accuracy is 0.85 (85%)

COMPARE TEST AND TRAINING DATA (CONFUSION MATRIX):





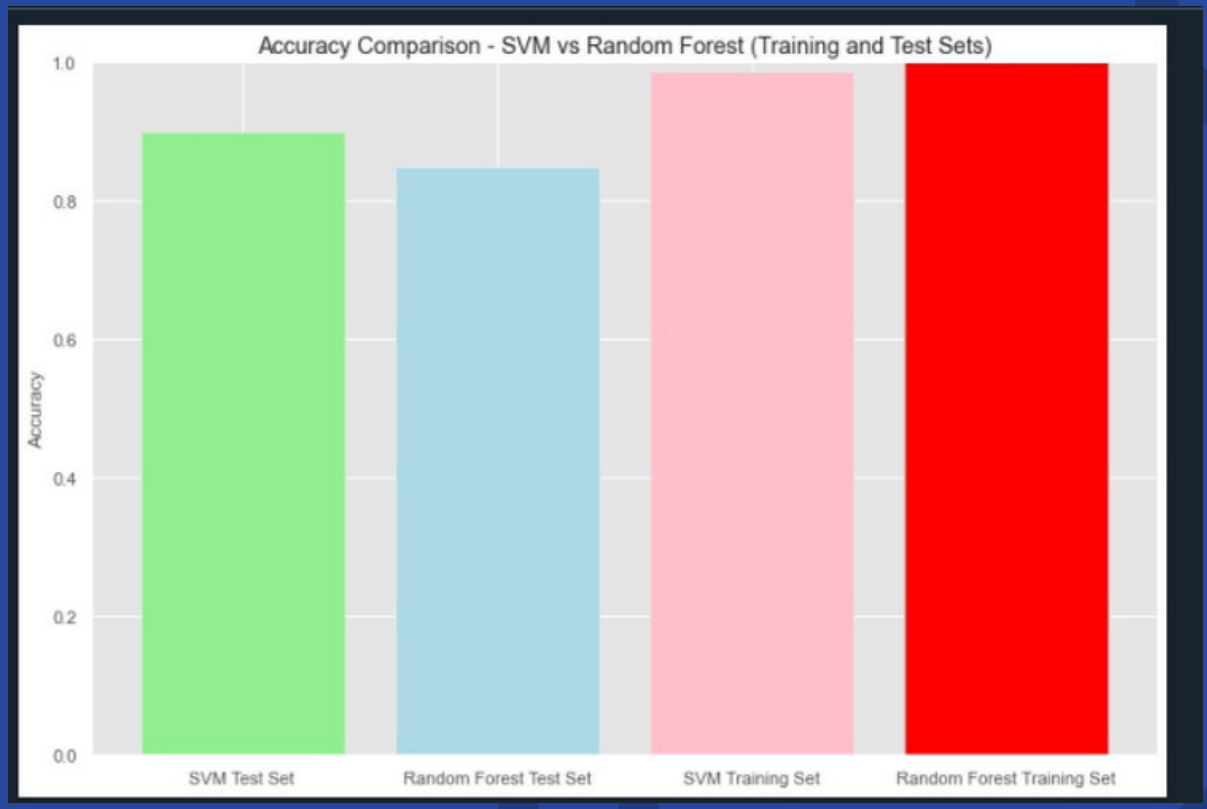




COMPARE TEST AND TRAINING DATA:

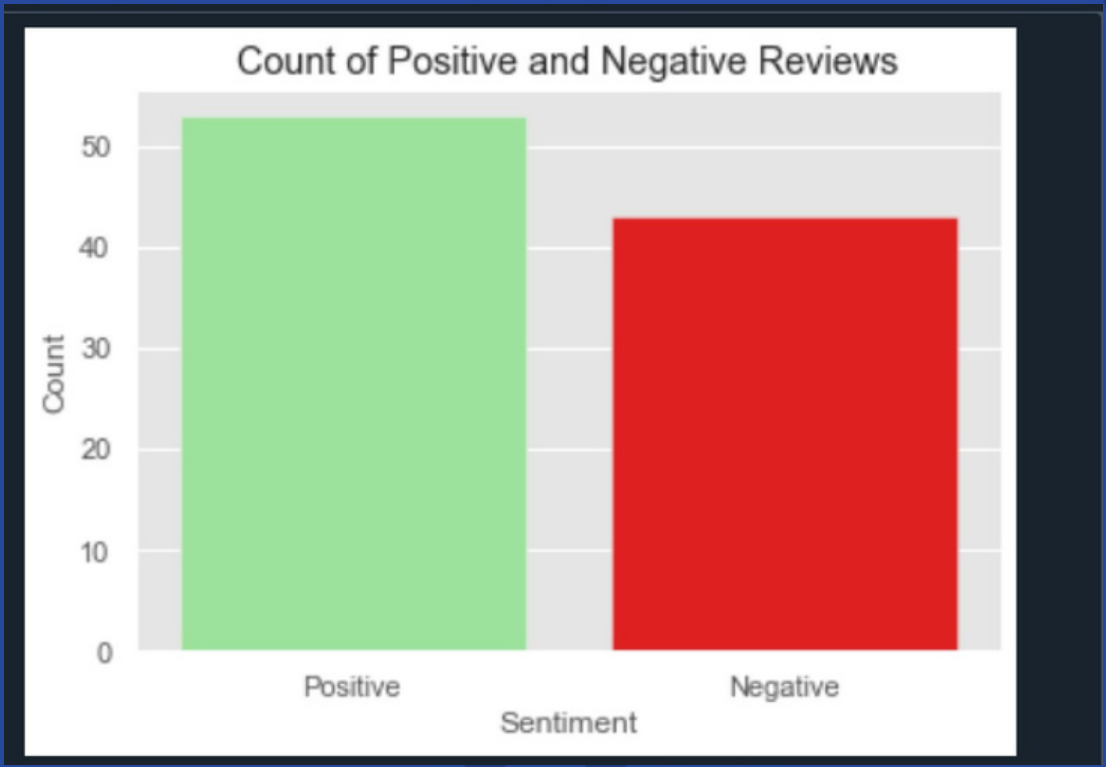
```python
# Function to plot confusion matrix
def plot_confusion_matrix(y_true, y_pred, title, labels):
    cm = confusion_matrix(y_true, y_pred)
    plt.figure(figsize=(6, 4))
    sns.heatmap(cm, annot=True, fmt='d', cmap='tab20', xticklabels=labels, yticklabels=labels)
    plt.title(title)
    plt.xlabel('Predicted')
    plt.ylabel('True')
    plt.show()

# Plotting Confusion Matrix for SVM (Test Set)
plot_confusion_matrix(y_test, svm_predictions, 'SVM Test Set Confusion Matrix', ['negative', 'positive'])

# Plotting Confusion Matrix for Random Forest (Test Set)
plot_confusion_matrix(y_test, rf_predictions, 'Random Forest Test Set Confusion Matrix', ['negative', 'positive'])

# Plotting Confusion Matrix for SVM (Training Set)
svm_train_predictions = svm_classifier.predict(X_train_tfidf)
plot_confusion_matrix(y_train, svm_train_predictions, 'SVM Training Set Confusion Matrix', ['negative', 'positive'])

# Plotting Confusion Matrix for Random Forest (Training Set)
rf_train_predictions = rf_classifier.predict(X_train_tfidf)
plot_confusion_matrix(y_train, rf_train_predictions, 'Random Forest Training Set Confusion Matrix', ['negative', 'positive'])

# Plotting Accuracy Comparison
models = ['SVM Test Set', 'Random Forest Test Set', 'SVM Training Set', 'Random Forest Training Set']
accuracies = [accuracy_score(y_test, svm_predictions), accuracy_score(y_test, rf_predictions),
              accuracy_score(y_train, svm_train_predictions), accuracy_score(y_train, rf_train_predictions)]
```

COMPARE ALL USED ALGORITHMS:




```
plt.figure(figsize=(12, 8))
plt.bar(models, accuracies, color=['lightgreen', 'lightblue', 'pink', 'red'])
plt.ylabel('Accuracy')
plt.title('Accuracy Comparison - SVM vs Random Forest (Training and Test Sets)')
plt.ylim(0, 1)
plt.show()
```

COMPARE POSTIVE VS NEGATIVE:




```
#Data Distribution
sns.countplot(x='Sentiment', data=data, palette= ['lightgreen', 'red'])
plt.xlabel('Sentiment')
plt.ylabel('Count')
plt.title('Count of Positive and Negative Reviews')
plt.show()
data['Sentiment'] = data['Sentiment'].str.strip()
data['Sentiment'] = data['Sentiment'].str.lower()
fig, axes = plt.subplots(2, 2, figsize=(12, 8))
```

COUNT USER, NUM OF REVIEW, POS&NEG?

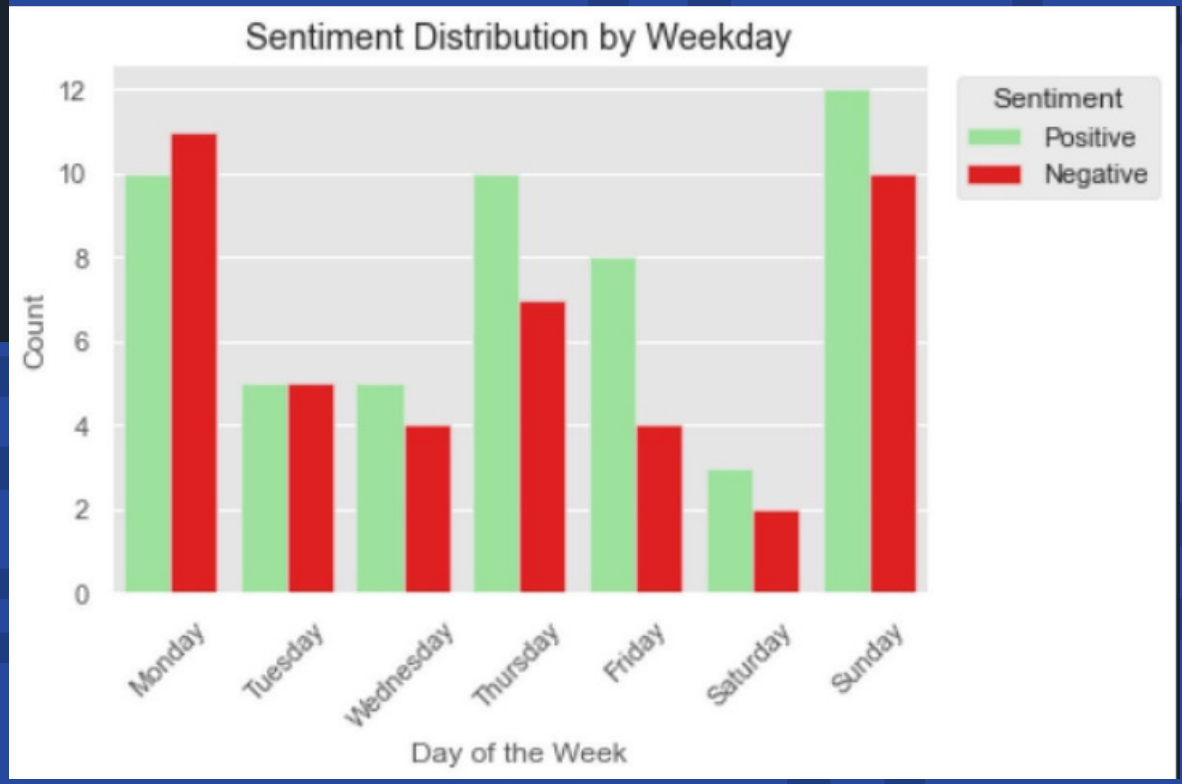
```python
#User Distribution
data.describe()

data['User ID'] = data['User ID'].str.strip()

user_reviews = data.groupby('User ID').agg(
    ReviewCount=('Text', 'count'),
    SentimentDistribution=('Sentiment', lambda x: dict(x.value_counts()))
).reset_index()

user_reviews = user_reviews[user_reviews['ReviewCount'] >= 2]

sentiment_data = pd.DataFrame(user_reviews['SentimentDistribution'].to_list())
sentiment_data.index = user_reviews['User ID']
sentiment_data
data[data['User ID'] == 'user456'][['Text', 'Source']]
```

user_reviews - DataFrame

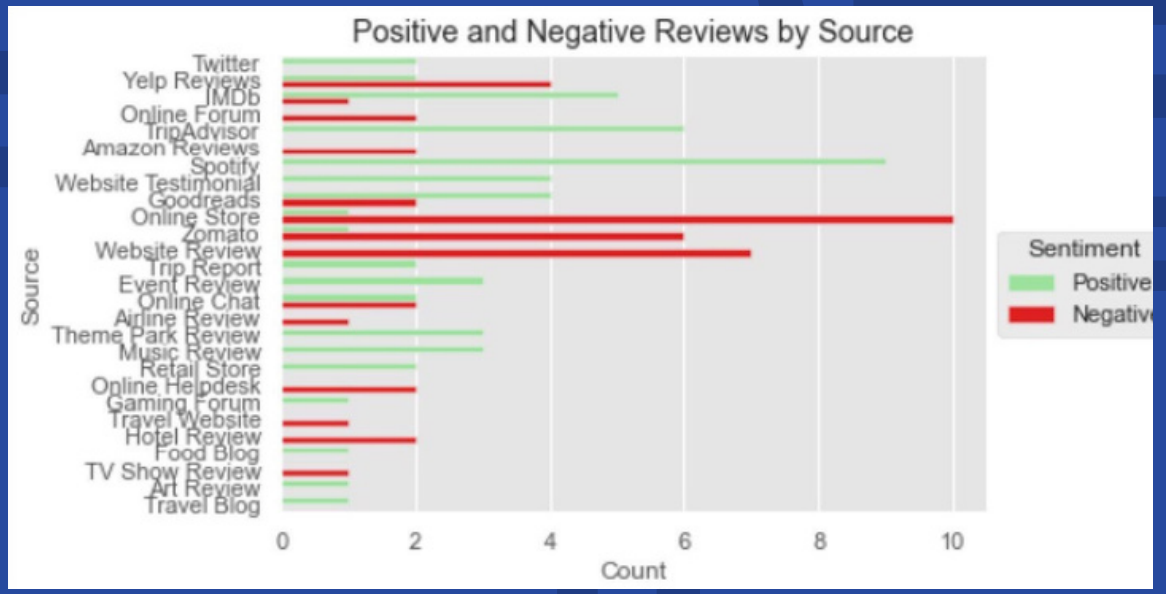
Index	User ID	viewCou	ntimentDistributi
0	@user123	2	{'positive': 2}
6	bookworm789	2	{'positive': 2}
17	foodie22	2	{'positive': 2}
20	foodlover123	2	{'negative': 2}
22	foodlover2468	2	{'negative': 2}
23	forumuser1	2	{'negative': 2}
28	moviefan789	2	{'positive': 2}
34	musiclover456	2	{'positive': 2}
35	musiclover789	2	{'positive': 2}
38	nostalgiacat123	2	{'positive': 2}
41	shopper123	2	{'negative': 2}
46	shopper789	2	{'negative': 2}
48	testimonialuser1	2	{'positive': 2}
50	thrillseeker1	2	{'positive': 2}

COUNT USER, NUM OF REVIEW, POS&NEG?

```python
#Data Distribution
sns.countplot(x='Sentiment', data=data, palette= ['lightgreen', 'red'])
plt.xlabel('Sentiment')
plt.ylabel('Count')
plt.title('Count of Positive and Negative Reviews')
plt.show()
data['Sentiment'] = data['Sentiment'].str.strip()
data['Sentiment'] = data['Sentiment'].str.lower()
data['DayOfWeek'] = data['Date'].dt.day_name()
custom_order = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']
data['DayOfWeek'] = pd.Categorical(data['DayOfWeek'], categories=custom_order, ordered=True)

sns.countplot(x='DayOfWeek', hue='Sentiment', data=data, palette= ['lightgreen', 'red'])

plt.xlabel('Day of the Week')
plt.ylabel('Count')
plt.title('Sentiment Distribution by Weekday')
plt.legend(title='Sentiment', labels=['Positive', 'Negative'], bbox_to_anchor=(1.02, 1), loc='upper left')
plt.xticks(rotation=45)
plt.show()
sns.countplot(y='Source', hue='Sentiment', data=data, palette=['lightgreen', 'red'])
plt.xlabel('Count')
plt.ylabel('Source')
plt.title('Positive and Negative Reviews by Source')
plt.legend(title='Sentiment', loc='center left', bbox_to_anchor=(1, 0.5), labels=['Positive', 'Negative'])
plt.show()

sns.countplot(y='Location', hue='Sentiment', data=data, palette=['lightgreen', 'red'])
plt.xlabel('Count')
plt.ylabel('Source')
plt.title('Positive and Negative Reviews by Location')
plt.legend(title='Sentiment', loc='center left', bbox_to_anchor=(1, 0.5), labels=['Positive', 'Negative'])
plt.show()
```



VIS COUNT USER, NUM OF REVIEW, POS&NEG?



CONCLUSION:

- The accuracy using SVM model was 0.9 (90%) and using Random Forest gave us 0.85 (85%). Thus, we can conclude that SVM is the better option for answering our questions because it’s higher.

- Analysis of customer feedback using predictive models like SVM and Random Forest can aid society and institutions by enhancing product/service quality and understanding consumer needs for better decision-making and improved offerings.

ETHICAL RESPONSIBILITIES:

- we handled customers data by make sure to applied privacy policy & accuracy by using Privacy Policy.

- applied inductive bias & fairness by using random split

TEAM MEMBERS:

01.	Salma Mohammed	2110956
02.	Shatha Al-qarni	2111884
03.	Dania Abdelmonem	2112241
04.	Nrdeen Ali	2110419

THANK YOU!