MAHATMA EDUCATION SOCIETY'S

PILLAI COLLEGE OF ARTS, COMMERCE & SCIENCE (Autonomous)

NEW PANVEL

PROJECT REPORT ON

'Sentiment Analysis of Amazon Alexa Customer Reviews'

IN PARTIAL FULFILLMENT OF

MASTER OF SCIENCE DATA ANALYTICS (PART

II) SEMESTER III– 2024-25

PROJECT GUIDE

Prof. Dipti Khiste

SUBMITTED BY: SHREYA BHATTACHARJEE

ROLL NO: 6861

PILLAI COLLEGE OF ARTS,COMMERCE & SCIENCE (Autonomous)

Re-accredited "A" Grade by NAAC (3rd Cycle)

Project Completion Certificate

This is to certify that **Shreya Bhattacharjee** of **M.Sc. Data Analytics Part – 2** has completed the project titled '**Sentiment Analysis of Amazon Alexa Customer Reviews**' of subject **Sentiment Analysis** under our guidance and supervision **Prof. Dipti Khiste** during the academic year 2024-25 in the department of Computer Science.

INDEX

INTRODUCTION

The "Sentiment Analysis of Amazon Alexa Customer Reviews" project aims to analyze customer feedback to understand the overall sentiment towards Amazon Alexa products. By leveraging natural language processing (NLP) techniques, the project identifies and classifies the sentiment expressed in customer reviews as positive, negative, or neutral. This analysis provides valuable insights into customer satisfaction and areas for improvement. The project utilizes various machine learning algorithms and data visualization tools to present findings in an easily interpretable format, ultimately helping Amazon to enhance product features and customer experience. By understanding the sentiment trends, Amazon can prioritize product updates and address common customer concerns more effectively. Furthermore, this analysis can assist in marketing strategies by highlighting the strengths appreciated by users, thereby fostering a stronger connection with the customer base.

PROJECT SUMMARY

PROJECT OVERVIEW:

The "Sentiment Analysis of Amazon Alexa Customer Reviews" project focuses on developing a system to analyze customer feedback and understand the overall sentiment towards Amazon Alexa products. The primary goal is to classify customer reviews into positive, negative, or neutral categories and provide actionable insights for product improvement and customer satisfaction enhancement.

OBJECTIVE:

The goal of this project is to develop a system for analyzing the sentiment of Amazon Alexa customer reviews. The system aims to determine whether customer feedback has a positive, negative, or neutral sentiment and to identify key areas for improvement based on this sentiment analysis.

KEY COMPONENTS:

1. Data Collection:

Gather a dataset of Amazon Alexa customer reviews. This data can be sourced from Amazon's website, Kaggle, or other relevant datasets.

2. Sentiment Analysis:

Text Processing: Use natural language processing (NLP) techniques to pre-process customer reviews (e.g., tokenization, stemming, and removal of stop words).

Sentiment Classification: Apply sentiment analysis models to classify the sentiment of each review as positive, negative, or neutral.

This can be achieved using machine learning models such as Logistic Regression, Multinomial Naive Bayes, Support Vector Machine.

3. Correlation Analysis:

Analyze the relationship between the sentiment scores of customer reviews and product ratings. This involves examining how positive or negative feedback correlates with overall product ratings and identifying any patterns or trends.

4. Model Evaluation:

Evaluate the performance of sentiment analysis models using metrics such as accuracy, precision, recall, and F1 score.

5. Application:

Customer Insights: Provide actionable insights for Amazon based on the sentiment analysis, helping them make informed decisions about product improvements and customer service enhancements.

Product Development: Use sentiment insights to prioritize product updates and feature enhancements based on customer feedback.

FINDINGS:

Sentiment Impact: Positive reviews generally correlate with higher product ratings, while negative reviews are associated with lower ratings. Neutral reviews have a less predictable impact.

Predictive Insights: The sentiment analysis provided valuable predictive insights into customer satisfaction and areas for potential product improvement.

CHALLENGES:

Data Quality: Ensuring the accuracy and relevance of customer reviews and product rating data.

Sentiment Complexity: Handling nuances in language and context to improve sentiment classification accuracy.

Volume of Data: Managing and analyzing a large volume of customer feedback efficiently.

FUTURE DIRECTIONS:

Model Improvement: Enhance sentiment analysis models by integrating more sophisticated NLP techniques and expanding training datasets.

Real-Time Analysis: Develop systems for real-time sentiment analysis to provide immediate insights and impact assessments.

Holistic Approach: Incorporate additional data sources, such as social media sentiment and competitor product reviews, for a more comprehensive analysis of customer feedback.

OUTCOME:

The successful implementation of this project will result in a robust tool for analysing and interpreting the sentiment of Amazon Alexa customer reviews. This can offer valuable insights for Amazon to enhance product features, improve customer satisfaction, and make informed decisions about product development and marketing strategies.

PYTHON CODE AND OUTPUT:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
from matplotlib import style
style.use("ggplot")
import seaborn as sns
import re
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer
from nltk.corpus import stopwords
stop_words = set(stopwords.words('english'))
from wordcloud import WordCloud
from sklearn.feature_extraction.text import
CountVectorizer
from sklearn.model_selection import train_test_split
```

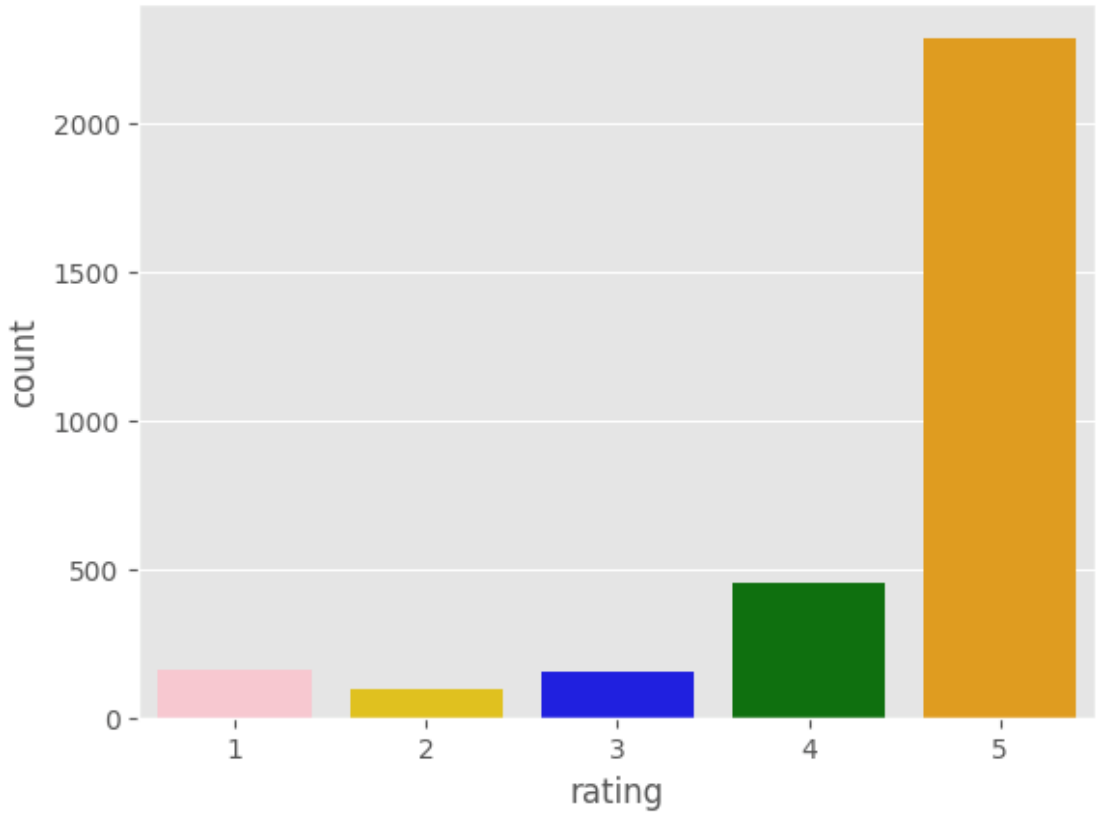
```
df = pd.read_csv("/content/amazon_alexa.tsv" , sep =
'\t')
df.head()
```

	rating	date	variation	verified_reviews	feedback
0	5	31-Jul-18	Charcoal Fabric	Love my Echo!	1
1	5	31-Jul-18	Charcoal Fabric	Loved it!	1
2	4	31-Jul-18	Walnut Finish	Sometimes while playing a game, you can answer...	1
3	5	31-Jul-18	Charcoal Fabric	I have had a lot of fun with this thing. My 4 ...	1
4	5	31-Jul-18	Charcoal Fabric	Music	1

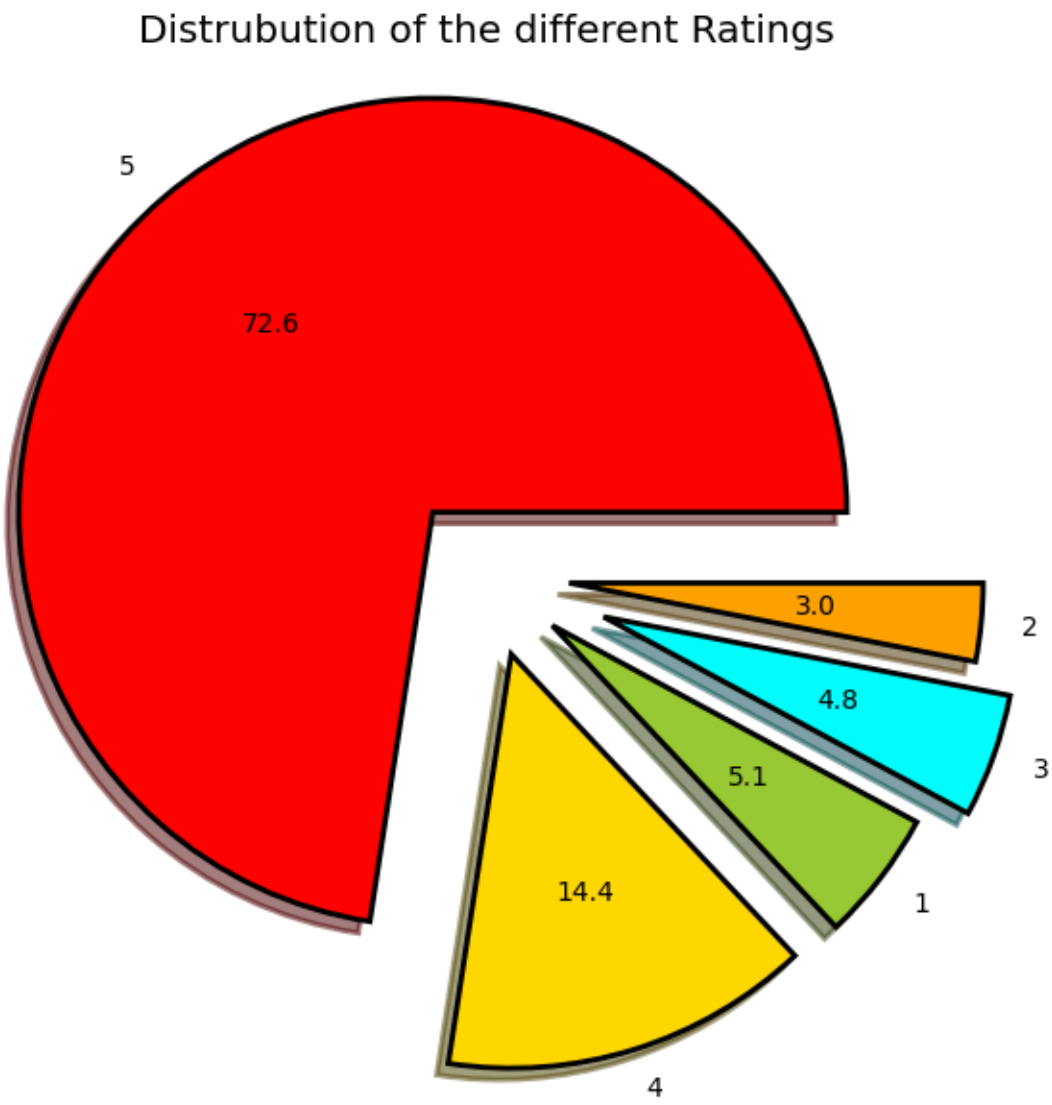
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3150 entries, 0 to 3149
Data columns (total 5 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   rating            3150 non-null   int64 
 1   date              3150 non-null   object
 2   variation         3150 non-null   object
 3   verified_reviews  3149 non-null   object
 4   feedback          3150 non-null   int64 
dtypes: int64(2), object(3)
memory usage: 123.2+ KB
```

```
sns.countplot(x='rating', data=df, palette=colors)
plt.show()
```

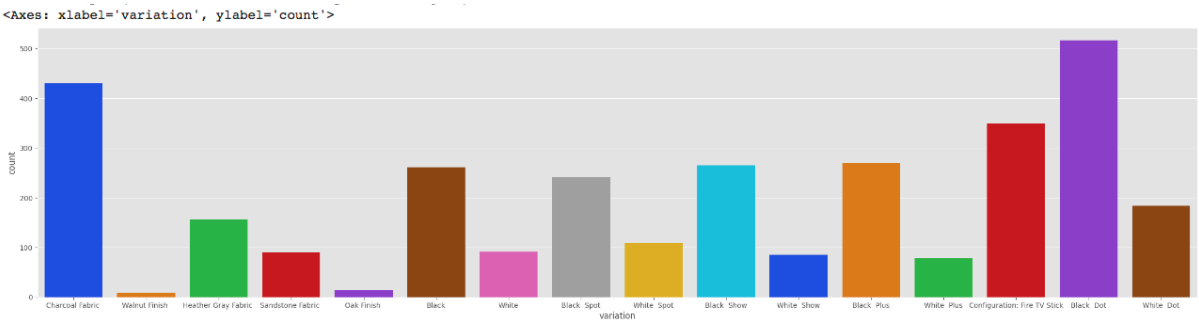



```
fig = plt.figure(figsize = (7,7))
colors = ('red', 'gold','yellowgreen','cyan','orange')
wp = {'linewidth': 2, 'edgecolor':'black'}
tags = df['rating'].value_counts()
explode = (0.2,0.2,0.2,0.3,0.2)
tags.plot(kind ='pie', autopct = '%1.1f',colors =
colors, shadow = True,
          startangle =0, wedgeprops =wp, explode =
explode, label ='')
plt.title('Distrubution of the different Ratings')
plt.show()
```




```
fig = plt.figure(figsize = (30,7))
sns.countplot(x='variation', data = df,
palette='bright')
```

<Axes: xlabel='variation', ylabel='count'>

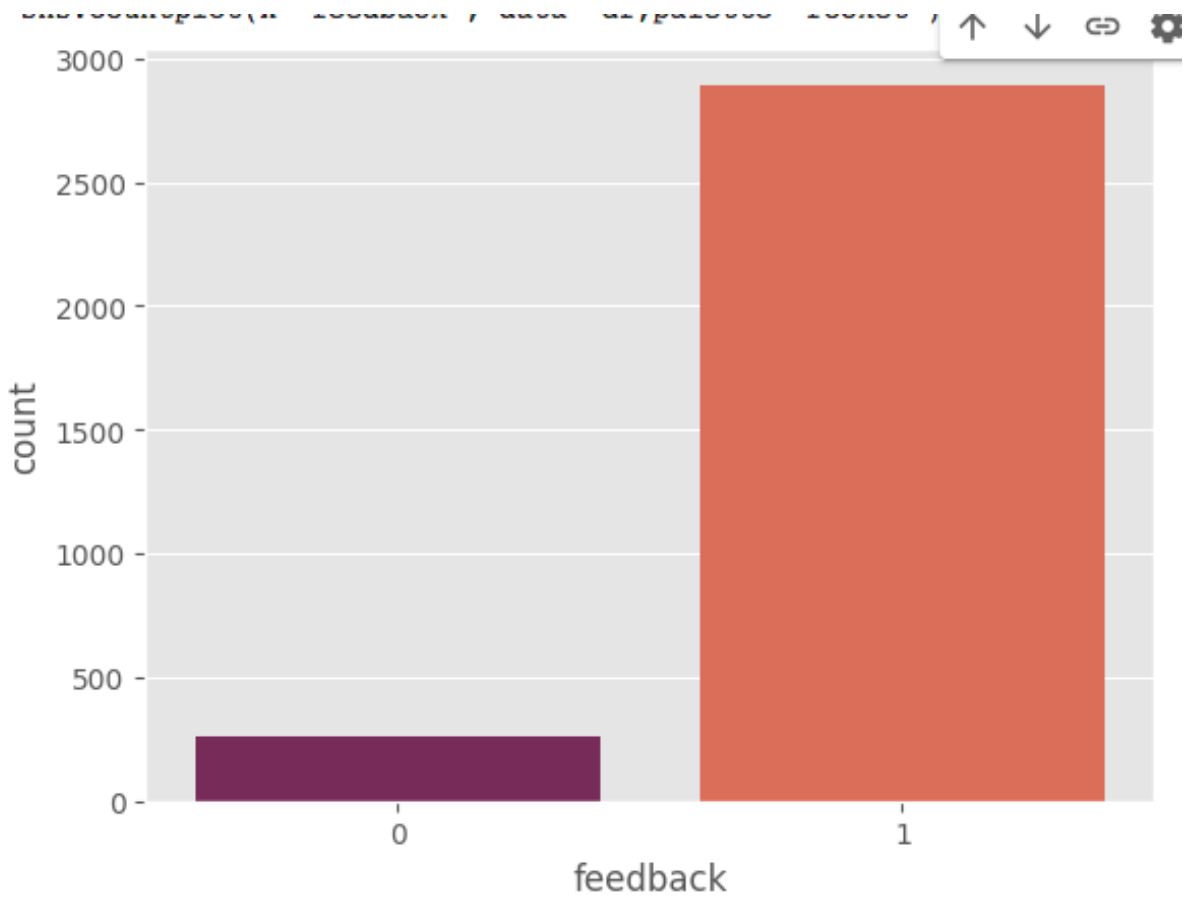



```
df['variation'].value_counts()
```

variation	count
Black Dot	516
Charcoal Fabric	430
Configuration: Fire TV Stick	350
Black Plus	270
Black Show	265
Black	261
Black Spot	241
White Dot	184
Heather Gray Fabric	157
White Spot	109
White	91
Sandstone Fabric	90
White Show	85
White Plus	78
Oak Finish	14
Walnut Finish	9

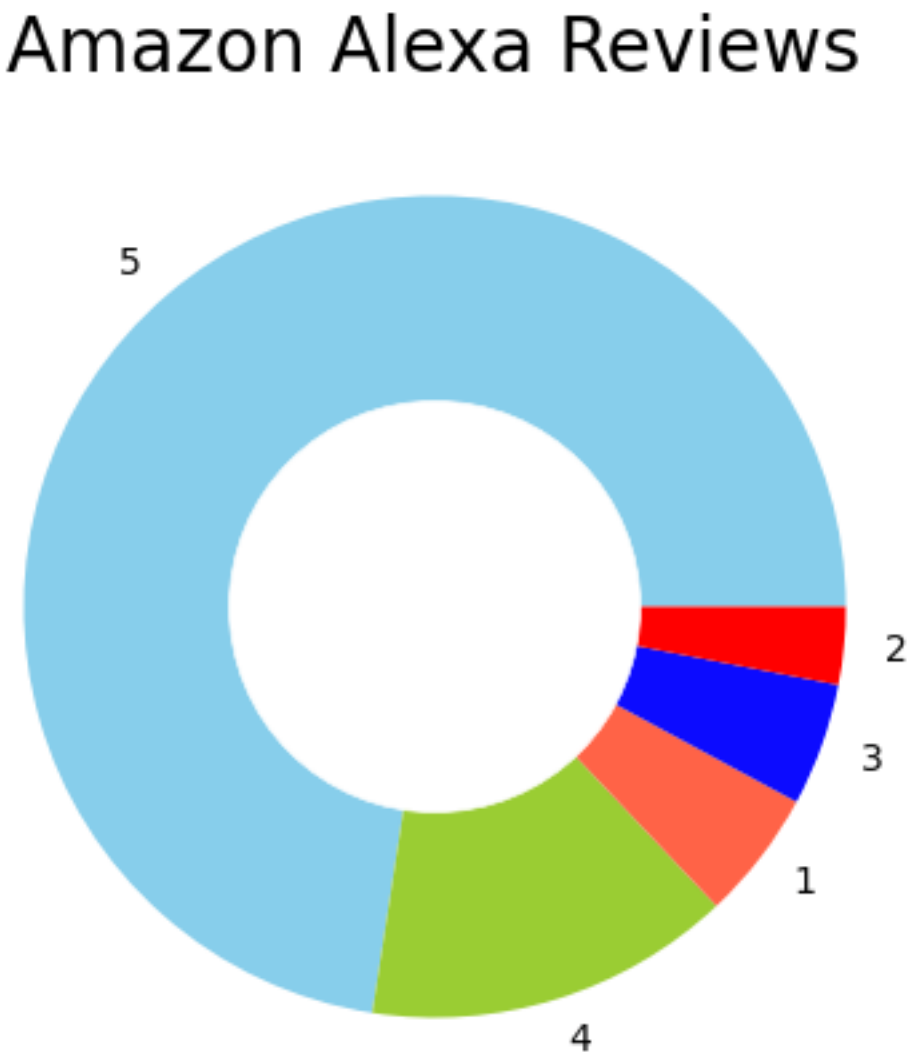
dtype: int64

```
sns.countplot(x='feedback', data =df,palette='rocket')
# Replace rocket with a string that represents a valid
palette name.
plt.show()
```




```
ratings = df["rating"].value_counts()
numbers = ratings.index
quantity = ratings.values

custom_colors = ["skyblue", "yellowgreen", 'tomato',
"blue", "red"]
plt.figure(figsize=(5, 5))
plt.pie(quantity, labels=numbers, colors=custom_colors)
central_circle = plt.Circle((0, 0), 0.5, color='white')
fig = plt.gcf()
fig.gca().add_artist(central_circle)
plt.rc('font', size=12)
plt.title("Amazon Alexa Reviews", fontsize=20)
```

```
plt.show()
```




```
import nltk
nltk.download('vader_lexicon') # Download lexicon for
sentiment analysis

from nltk.sentiment.vader import
SentimentIntensityAnalyzer # Import the class

sentiments = SentimentIntensityAnalyzer()
df["Positive"] = [sentiments.polarity_scores(i)["pos"]
for i in df["verified_reviews"]]
```

```
df["Negative"] = [sentiments.polarity_scores(i)["neg"]
for i in df["verified_reviews"]]
df["Neutral"] = [sentiments.polarity_scores(i)["neu"]
for i in df["verified_reviews"]]
print(df.head())
```

```
[nltk_data] Downloading package vader_lexicon to /root/nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
   rating       date         variation  \
0       5  31-Jul-18  Charcoal Fabric
1       5  31-Jul-18  Charcoal Fabric
2       4  31-Jul-18    Walnut Finish
3       5  31-Jul-18  Charcoal Fabric
4       5  31-Jul-18  Charcoal Fabric

                                   verified_reviews  feedback  Positive  \
0                                         love echo         1     0.808
1                                             loved         1     1.000
2  sometimes playing game answer question correct...         1     0.201
3  lot fun thing 4 yr old learns dinosaurs contro...         1     0.554
4                                             music         1     0.000

   Negative  Neutral
0     0.000    0.192
1     0.000    0.000
2     0.145    0.654
3     0.000    0.446
4     0.000    1.000
```

```
x = sum(df["Positive"])
y = sum(df["Negative"])
z = sum(df["Neutral"])

def sentiment_score(a, b, c):
    if (a>b) and (a>c):
        print("Positive 😊 ")
    elif (b>a) and (b>c):
        print("Negative 😠 ")
    else:
        print("Neutral 🙂 ")
sentiment_score(x, y, z)
```

```
Neutral 🙂
```

```
print("Positive: ", x)
print("Negative: ", y)
print("Neutral: ", z)
```

```
Positive:  1410.0230000000001
Negative:  126.29399999999997
Neutral:  1525.6899999999957
```

So we can see that Positive and Neutral are above 1000 where Negative is below 100. So this means that most of the customers of Amazon Alexa are satisfied with its services.

```
for i in range(5):
    print(df['verified_reviews'].iloc[i],"\n")
```

```
Love my Echo!

Loved it!

Sometimes while playing a game, you can answer a question correctly but Alexa says you got it wrong and answers the same as you.  I like being able to t

I have had a lot of fun with this thing. My 4 yr old learns about dinosaurs, i control the lights and play games like categories. Has nice sound when pl

Music
```

```
import nltk
nltk.download('punkt')

def data_processing(text):
    if isinstance(text, str):  # Check if text is a
string
        text = text.lower()
        text = re.sub(r"http\S+www\S+|https\S+", '',
text, flags=re.MULTILINE)
        text = re.sub(r'[^\w\s]', '', text)
        text_tokens = word_tokenize(text)
        filtered_text = [w for w in text_tokens if not
w in stop_words]
        return " ".join(filtered_text)
    else:
```

```
        return ""  # Return empty string for non-string
values

df.verified_reviews =
df['verified_reviews'].apply(data_processing)
```

```
pos_reviews = df[df.feedback == 1]
pos_reviews.head()
```

	rating	date	variation	verified_reviews	feedback
0	5	31-Jul-18	Charcoal Fabric	love echo	1
1	5	31-Jul-18	Charcoal Fabric	loved	1
2	4	31-Jul-18	Walnut Finish	sometimes playing game answer question correct...	1
3	5	31-Jul-18	Charcoal Fabric	lot fun thing 4 yr old learns dinosaurs contro...	1
4	5	31-Jul-18	Charcoal Fabric	music	1

```
text = ' '.join([word for word in
pos_reviews['verified_reviews']])
plt.figure(figsize=(20,15), facecolor='None')
wordcloud = WordCloud(max_words=500, width=1600,
height=800, colormap='hot').generate(text)
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('Most frequent words in positive reviews',
fontsize=19)
plt.show()
```

Most frequent words in positive reviews

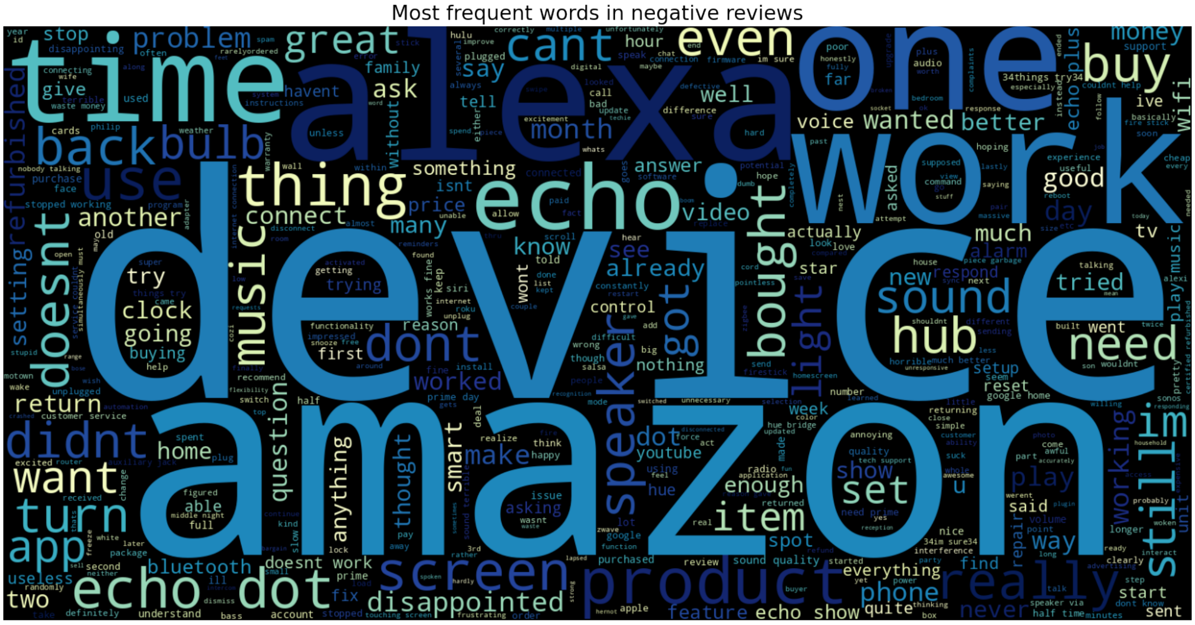

```
neg_reviews = df[df.feedback==0]
neg_reviews.head()
```

	rating	date	variation	verified_reviews	feedback
46	2	30-Jul-18	Charcoal Fabric	like siri fact siri answers accurately alexa d...	0
111	2	30-Jul-18	Charcoal Fabric	sound terrible u want good music get bose	0
141	1	30-Jul-18	Charcoal Fabric	much features	0
162	1	30-Jul-18	Sandstone Fabric	stopped working 2 weeks didnt follow commands ...	0
176	2	30-Jul-18	Heather Gray Fabric	sad joke worthless	0

```
text = ' '.join([word for word in
neg_reviews['verified_reviews']])
plt.figure(figsize=(20,15), facecolor='None')
wordcloud = WordCloud(max_words=500, width=1600,
height=800, colormap='YlGnBu').generate(text)
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('Most frequent words in negative reviews',
fontsize=19)
```

```
plt.show()
```

Most frequent words in negative reviews




```
X = df['verified_reviews']
Y = df['feedback']
```

```
cv = CountVectorizer()
X = cv.fit_transform(df['verified_reviews'])
```

```
x_train, x_test, y_train, y_test =
train_test_split(X,Y, test_size=0.2, random_state=42)
```

```
print("Size of x_train: ",(x_train.shape))
print("Size of y_train: ",(y_train.shape))
print("Size of x_test: ",(x_test.shape))
print("Size of y_test: ",(y_test.shape))
```

```
Size of x_train:  (2520, 4364)
Size of y_train:  (2520,)
Size of x_test:  (630, 4364)
Size of y_test:  (630,)
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score,
classification_report, confusion_matrix

logreg = LogisticRegression()
logreg.fit(x_train, y_train)
logreg_pred = logreg.predict(x_test)
logreg_acc = accuracy_score(logreg_pred, y_test)
print("Test accuracy: {:.2f}%".format(logreg_acc*100))
```

```
Test accuracy: 93.65%
```

```
print(confusion_matrix(y_test, logreg_pred))
print("\n")
print(classification_report(y_test, logreg_pred))
```

```
[[ 21  37]
 [  3 569]]
```

	precision	recall	f1-score	support
0	0.88	0.36	0.51	58
1	0.94	0.99	0.97	572
accuracy			0.94	630
macro avg	0.91	0.68	0.74	630
weighted avg	0.93	0.94	0.92	630

```
mnb = MultinomialNB()
mnb.fit(x_train, y_train)
mnb_pred = mnb.predict(x_test)
mnb_acc = accuracy_score(mnb_pred, y_test)
print("Test accuracy: {:.2f}%".format(mnb_acc*100))

print(confusion_matrix(y_test, mnb_pred))
print("\n")
print(classification_report(y_test, mnb_pred))
```

```
Test accuracy: 92.22%
[[ 15  43]
 [  6 566]]
```

	precision	recall	f1-score	support
0	0.71	0.26	0.38	58
1	0.93	0.99	0.96	572
accuracy			0.92	630
macro avg	0.82	0.62	0.67	630
weighted avg	0.91	0.92	0.91	630

```
from sklearn.svm import SVC
svm = SVC()
svm.fit(x_train, y_train)
svm_pred = svm.predict(x_test)
svm_acc = accuracy_score(svm_pred, y_test)
print("Test accuracy: {:.2f}%".format(svm_acc*100))

print(confusion_matrix(y_test, svm_pred))
print("\n")
print(classification_report(y_test, svm_pred))
```

```
Test accuracy: 91.75%
[[  6  52]
 [  0 572]]
```

	precision	recall	f1-score	support
0	1.00	0.10	0.19	58
1	0.92	1.00	0.96	572
accuracy			0.92	630
macro avg	0.96	0.55	0.57	630
weighted avg	0.92	0.92	0.89	630

```python
from sklearn.model_selection import GridSearchCV

# Hyperparameter tuning for Logistic Regression
param_grid = {'C': [0.1, 1, 10]}
grid_search = GridSearchCV(logreg, param_grid, cv=5)
grid_search.fit(x_train, y_train)

print("Best parameters for Logistic Regression:",
grid_search.best_params_)
print("Best cross-validation score for Logistic
Regression:", grid_search.best_score_)

# Hyperparameter tuning for Multinomial Naive Bayes
param_grid = {'alpha': [0.1, 1, 10]}
grid_search = GridSearchCV(mnb, param_grid, cv=5)
grid_search.fit(x_train, y_train)

print("Best parameters for Multinomial Naive Bayes:",
grid_search.best_params_)
```

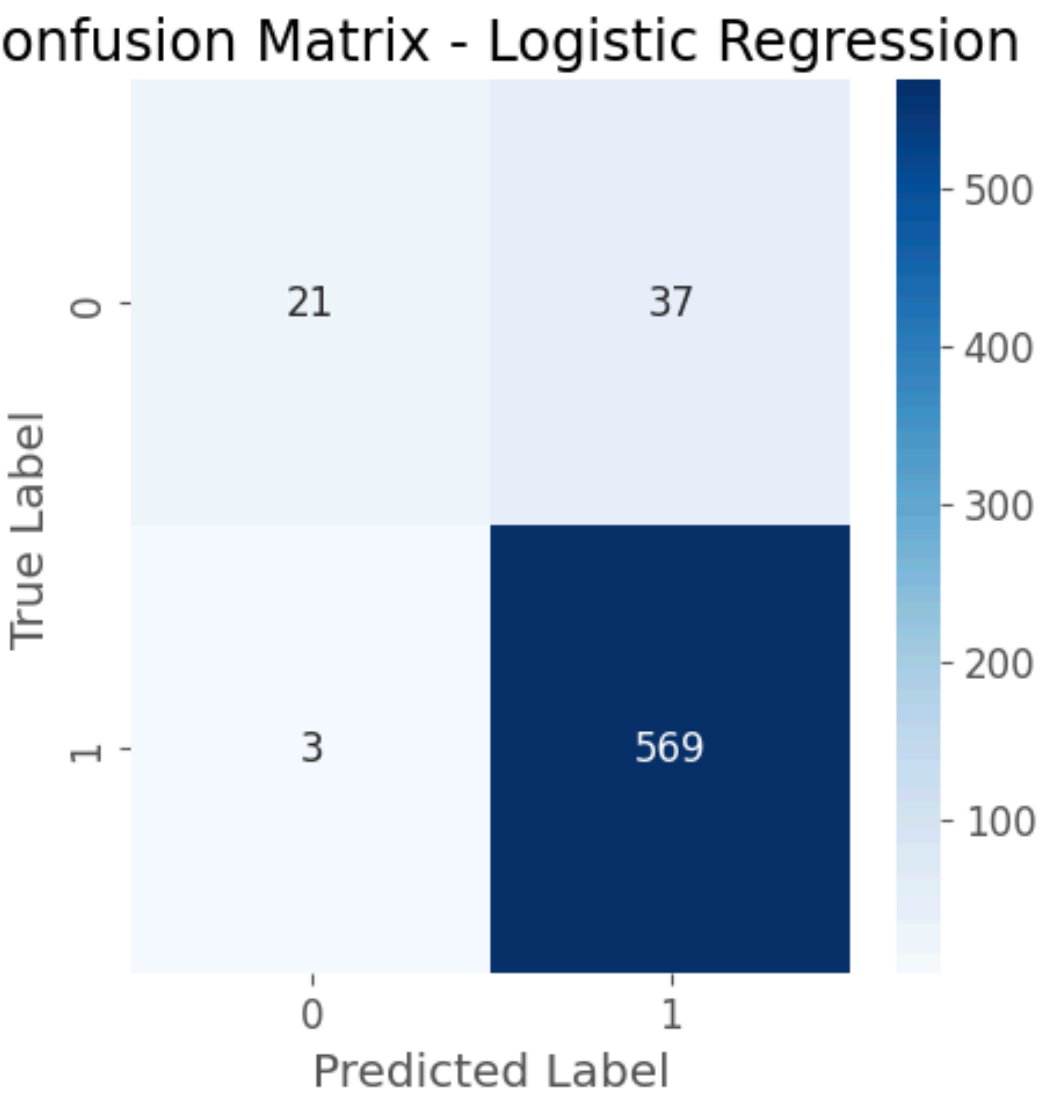
```
print("Best cross-validation score for Multinomial
Naive Bayes:", grid_search.best_score_)

# Hyperparameter tuning for Support Vector Machine
param_grid = {'C': [0.1, 1, 10], 'kernel': ['linear',
'rbf']}
grid_search = GridSearchCV(svm, param_grid, cv=5)
grid_search.fit(x_train, y_train)

print("Best parameters for Support Vector Machine:",
grid_search.best_params_)
print("Best cross-validation score for Support Vector
Machine:", grid_search.best_score_)
```

```
Best parameters for Logistic Regression: {'C': 1}
Best cross-validation score for Logistic Regression: 0.9365079365079365
Best parameters for Multinomial Naive Bayes: {'alpha': 1}
Best cross-validation score for Multinomial Naive Bayes: 0.932936507936508
Best parameters for Support Vector Machine: {'C': 0.1, 'kernel': 'linear'}
Best cross-validation score for Support Vector Machine: 0.934126984126984
```

```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import accuracy_score,
confusion_matrix, classification_report

# Assuming you have already defined logreg, mnb, svm,
and their predictions
models = [
    ('Logistic Regression', logreg, logreg_pred),
    ('Multinomial Naive Bayes', mnb, mnb_pred),
    ('Support Vector Machine', svm, svm_pred)
]

for name, model, pred in models:
    print(f"--- {name} ---")
    print("Accuracy:
{:.2f}%".format(accuracy_score(y_test, pred) * 100))
```

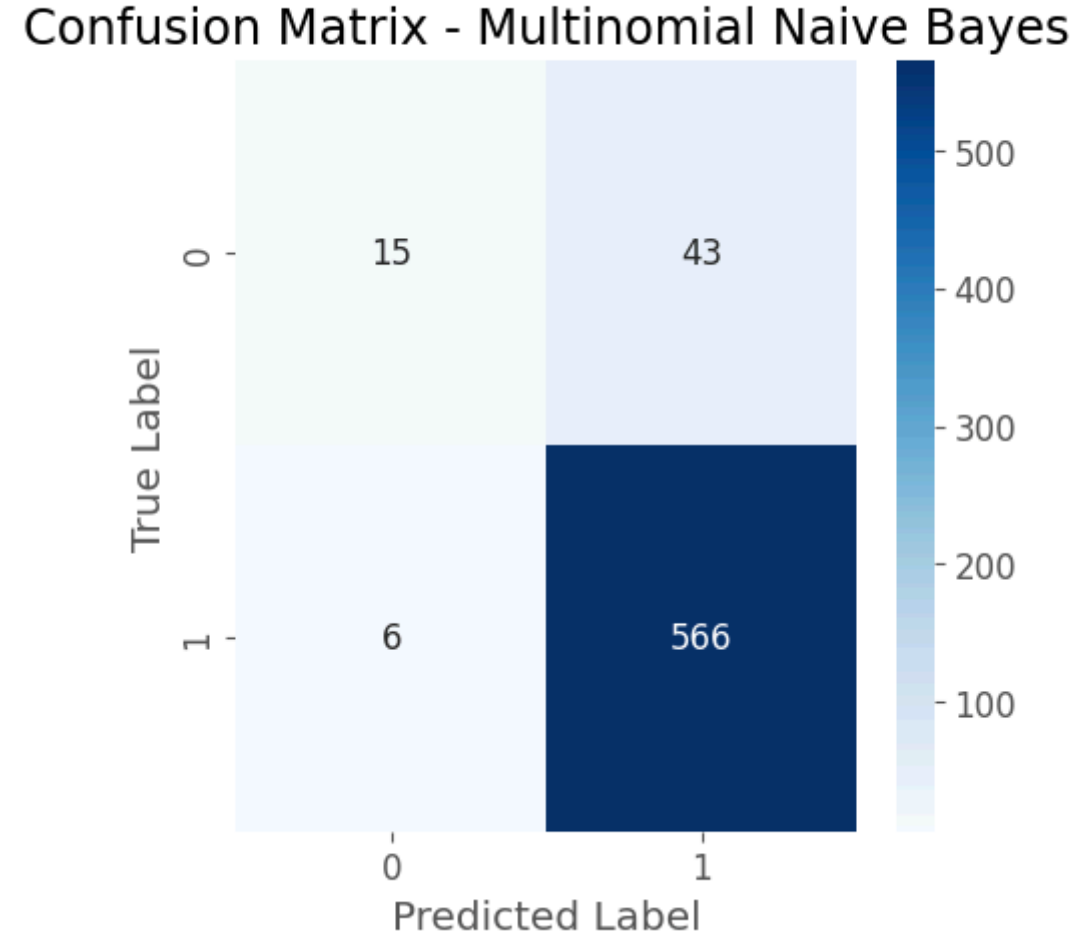
```
    # Confusion Matrix Visualization
    cm = confusion_matrix(y_test, pred)
    plt.figure(figsize=(5, 5))
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
    plt.title(f'Confusion Matrix - {name}')
    plt.xlabel('Predicted Label')
    plt.ylabel('True Label')
    plt.show()

    # Classification Report Visualization (Bar Chart)
    report = classification_report(y_test, pred,
output_dict=True)
    metrics = ['precision', 'recall', 'f1-score']
    classes = list(report.keys())[:-3]  # Exclude
'accuracy', 'macro avg', 'weighted avg'

    plt.figure(figsize=(8, 6))
    for metric in metrics:
        scores = [report[class_][metric] for class_ in
classes]
        plt.bar(classes, scores, label=metric)

    plt.xlabel('Class')
    plt.ylabel('Score')
    plt.title(f'Classification Report - {name}')
    plt.legend()
    plt.show()

    print("\n")
```

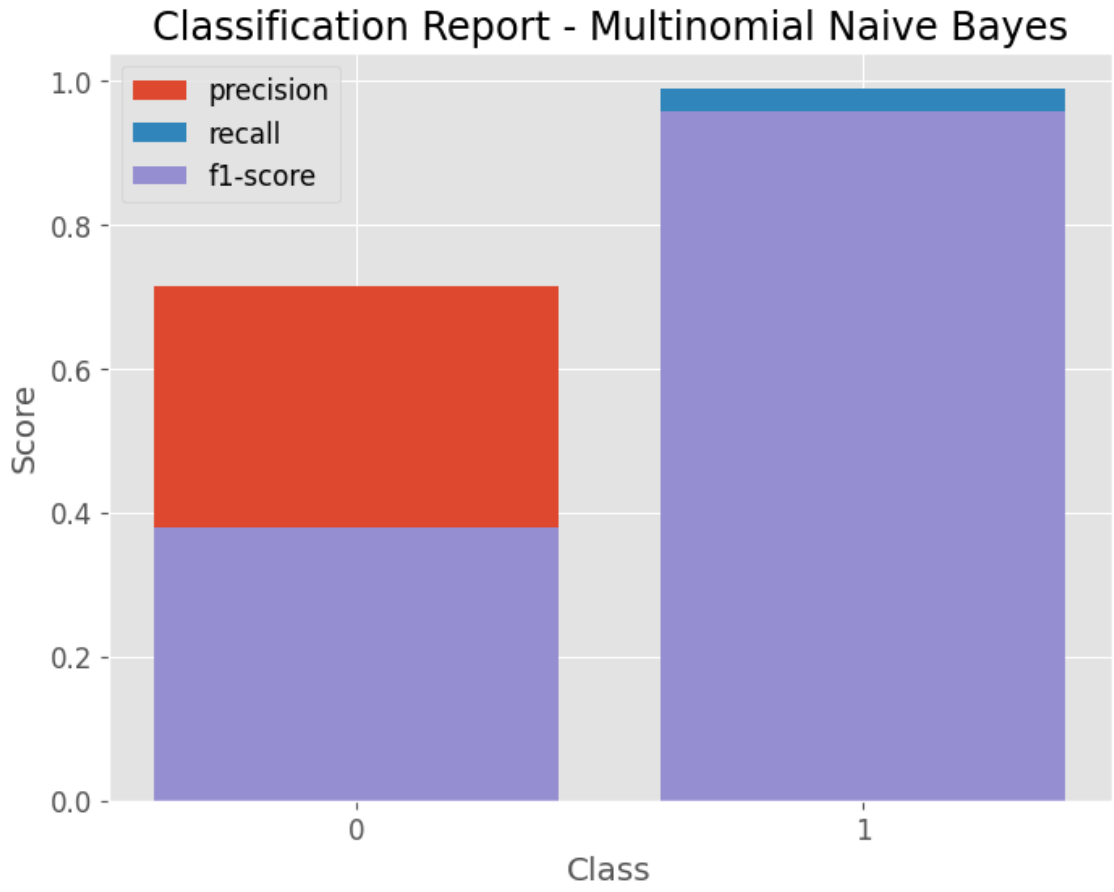
```
--- Logistic Regression ---
Accuracy: 93.65%
```



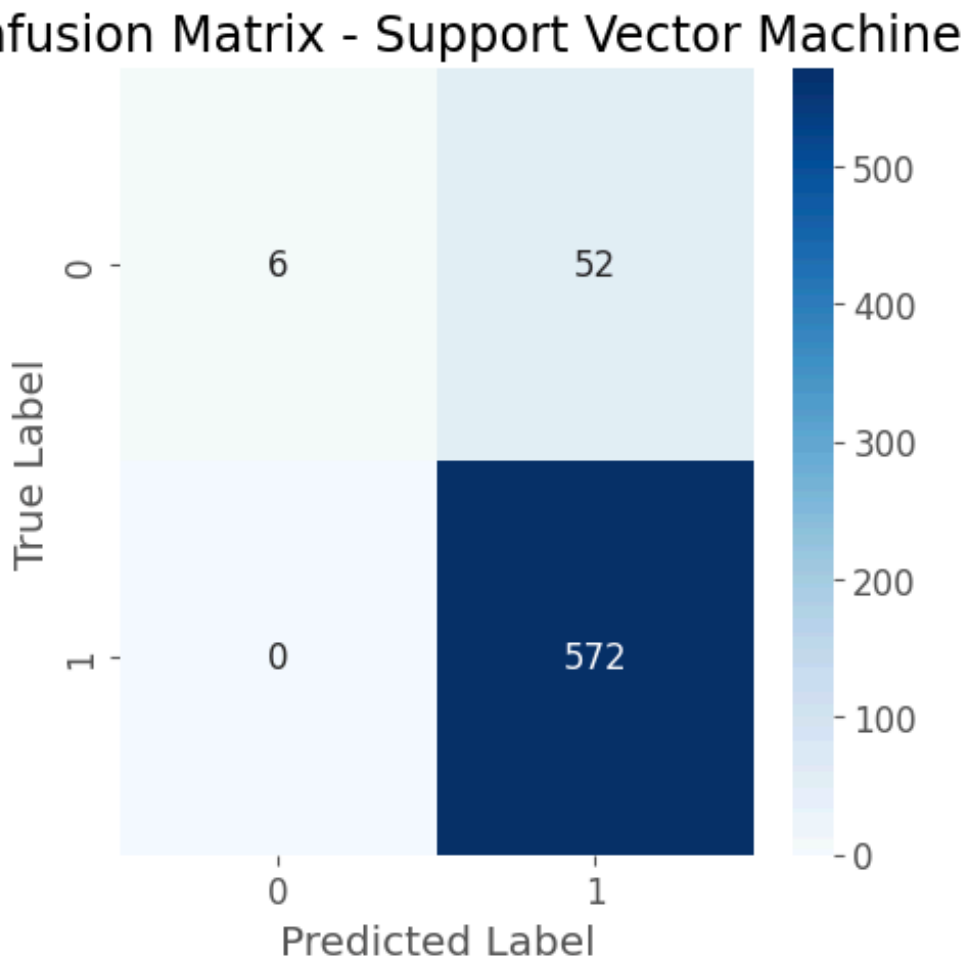

Classification Report - Logistic Regression
precision
recall
f1-score
1.0
0.8
0.6
0.4
0.2
0.0
Score
0
1
Class

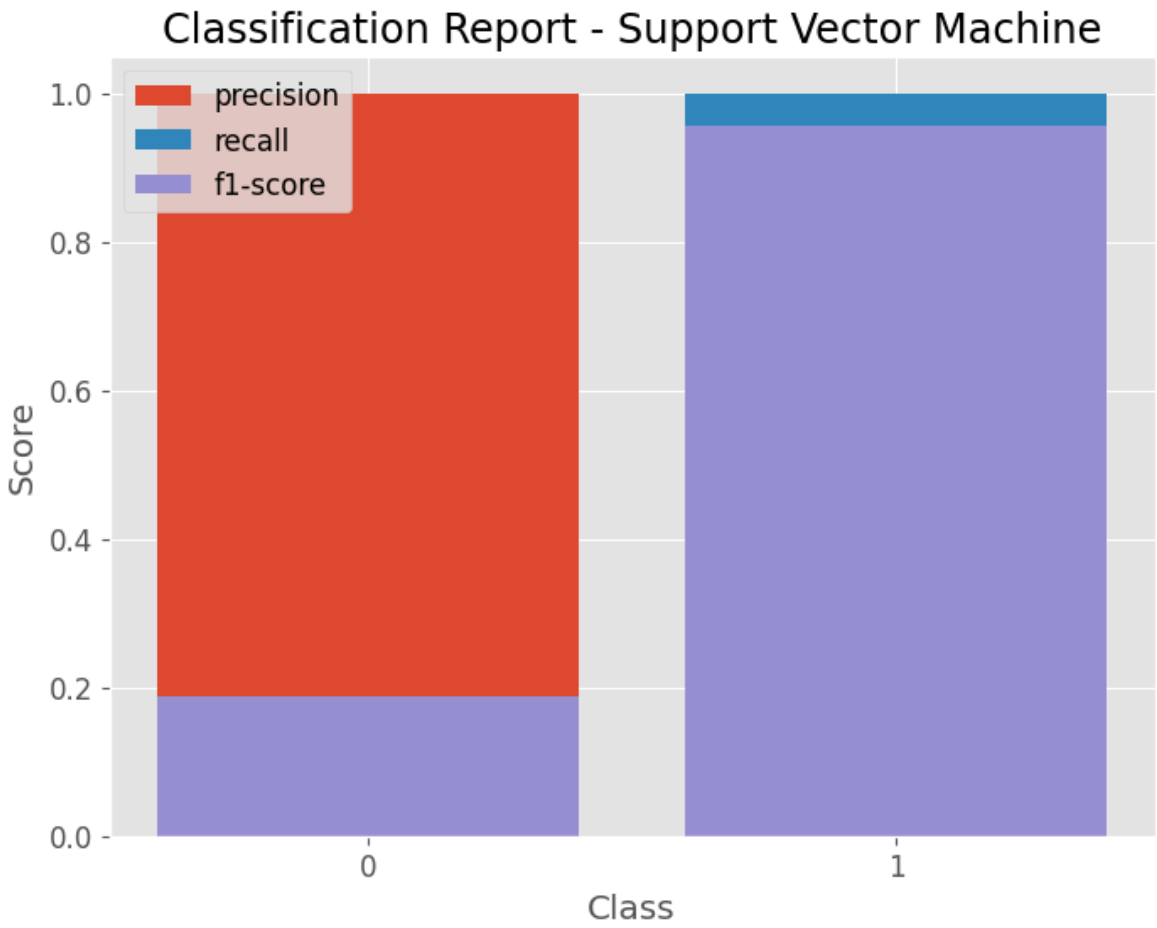
```
--- Multinomial Naive Bayes ---
Accuracy: 92.22%
```




Classification Report - Multinomial Naive Bayes
precision
recall
f1-score
1.0
0.8
0.6
0.4
0.2
0.0
Score
0
1
Class

```
--- Support Vector Machine ---
Accuracy: 91.75%
```





CONCLUSION

In conclusion, this project demonstrates the potential of sentiment analysis in transforming customer feedback into actionable insights, ultimately helping Amazon to enhance its products and strengthen its relationship with customers. Further advancements in NLP techniques and the integration of additional data sources will continue to improve the accuracy and impact of sentiment analysis in the future.