Applied Machine Learning

Assignment 2

Names:

Sema Abdelnasser Mosaad

Nada Mohammed Zakaria

Dina Ibrahim mohammady

Part 1

1)

1) Calculate the prior probabilities:
    - P(Yes)=6/14
    - P(No)=8/14
2) Calculate the likelihood probabilities:

Color	P(Yes)	P(No)
Red	3/6	4/8
Yellow	2/6	2/8
Blue	1/6	2/8

Type	P(Yes)	P(No)
Sports	4/6	3/8
SUV	2/6	5/8

Origin	P(Yes)	P(No)
Domestic	2/6	5/8
Imported	4/6	3/8

3) Calculate the posterior probabilities:
    - P(Yes\Blue, SUV, Domestic)= 0.0079/P(Blue, SUV, Domestic)
    - P(No\Blue, SUV, Domestic)= 0.0558/ P(Blue, SUV, Domestic)

P(Blue, SUV, Domestic)= 0.0079+0.0558=0.0637

P(Yes\Blue, SUV, Domestic)=0.124

P(No\Blue, SUV, Domestic)=0.876

Given the fact P(Yes\Blue, SUV, Domestic)< P(No\Blue, SUV, Domestic) , we classify the new instance as "NO"

2)

$R(a1\backslash x) = 0*P(class1\backslash x) + 6*P(class2\backslash x) = 6(1-P(class1\backslash x))$

$R(a2\backslash x) = 3*P(class1\backslash x) + 0*P(class2\backslash x) = 3P(class1\backslash x)$

We choose a1 if:

$R(a1\backslash x) < 2$

$6(1-P(class1\backslash x)) < 2$

$P(class1\backslash x)>2/3$

We choose a2 if:

$R(a2\backslash x) <2$

$3P(class1\backslash x) <2$

$P(class1\backslash x) <2/3$

So, we reject if:

$2/3 <P(class1\backslash x)< 2/3$

Part 2

Part (a):

Splitting the dataset into training and test data

```
[23] # Calculate the index at which to split the dataset
     split_index = int(0.8 * len(data))

     # Split the dataset into training and test data
     training_data = data[:split_index]
     test_data = data[split_index:]
```

```
[24] # Splitting training_data into x_train and y_train
     x_training = training_data.iloc[:, :-1]  # Select all columns except the last one as input features
     y_training = training_data.iloc[:, -1]   # Select the last column as the label

     # Splitting test_data into x_test and y_test
     x_testing = test_data.iloc[:, :-1]    # Select all columns except the last one as input features
     y_testing = test_data.iloc[:, -1]     # Select the last column as the label
```

Gaussian Naive Bayes Classifier

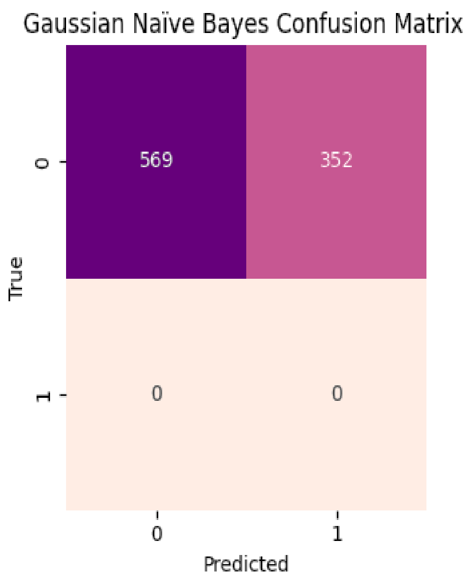
The trained classifiers are then used to make predictions on the test data. The accuracy of each classifier is calculated by comparing the predicted labels with the true labels of the test data.

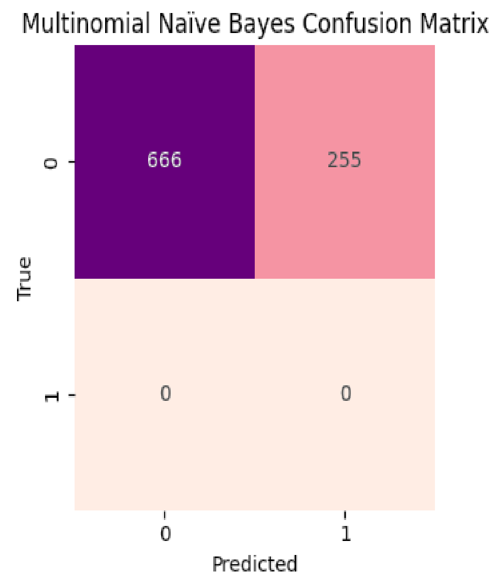
The accuracy scores of the three classifiers on the test data are as follows:

Gaussian Naive Bayes Accuracy: `0.6178067318132465`

Multinomial Naive Bayes Accuracy: `0.7231270358306189`

Confusion Matrices





Part (b):

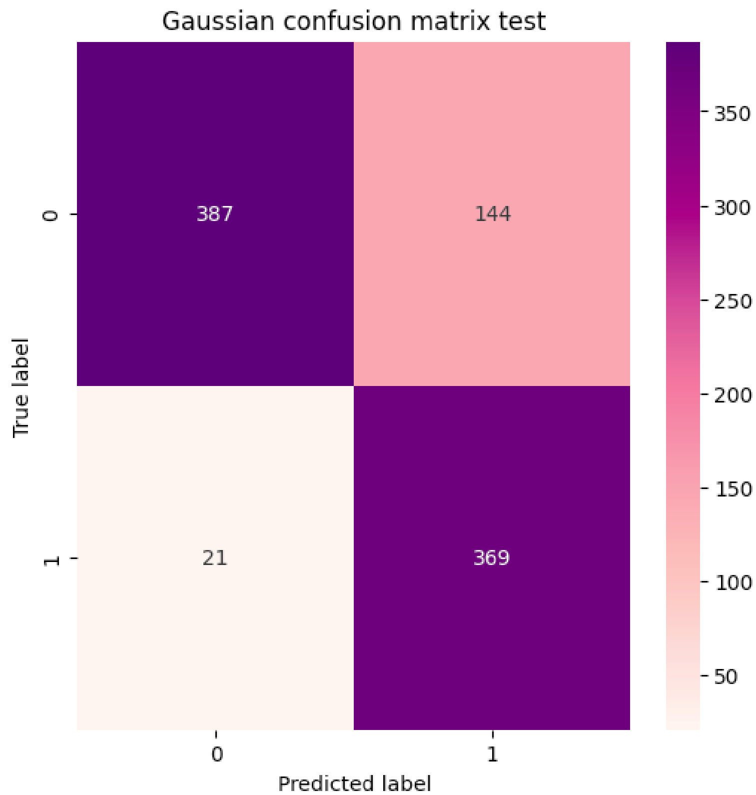
Train-Test Split on Full Dataset

In this part, the entire dataset is split into training and test data using the **train_test_split** function from the **sklearn.model_selection** module. The data is split in an 80:20 ratio, with 80% of the samples used for training and 20% for testing

```
[27]
    y = data[57]
    x = data.drop(columns=[57])
    X_train, X_test, y_train, y_test = train_test_split(x,y, test_size=0.2,random_state=42)
```

Gaussian Naive Bayes Classifier

Confusion Matrices



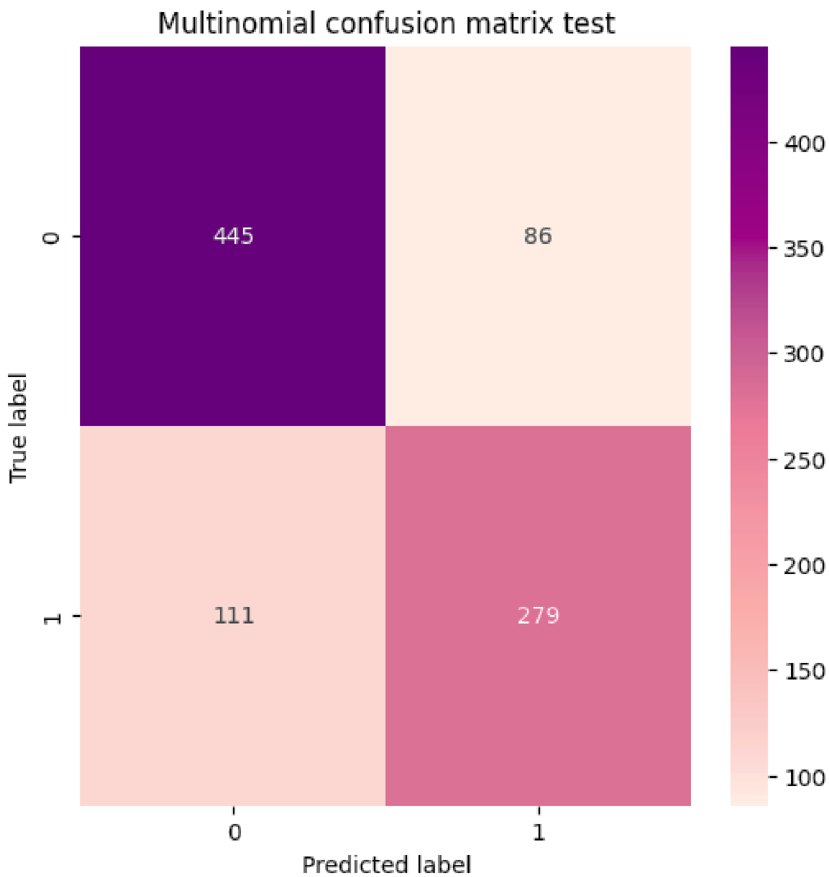
Classification Report

```
              precision    recall  f1-score   support

           0       0.95      0.73      0.82       531
           1       0.72      0.95      0.82       390

    accuracy                           0.82       921
   macro avg       0.83      0.84      0.82       921
weighted avg       0.85      0.82      0.82       921
```

```
Gaussian Naive Bayes Classifiers test acc_sco
 0.8208469055374593
precision of Gaussian Naive Bayes Classifiers
 0.7192982456140351
recall of Gaussian Naive Bayes Classifiers
 0.9461538461538461
f1_score of Gaussian Naive Bayes Classifiers
 0.8172757475083057
```

Multinomial Naive Bayes Classifier

Confusion Matrices



Classification Report

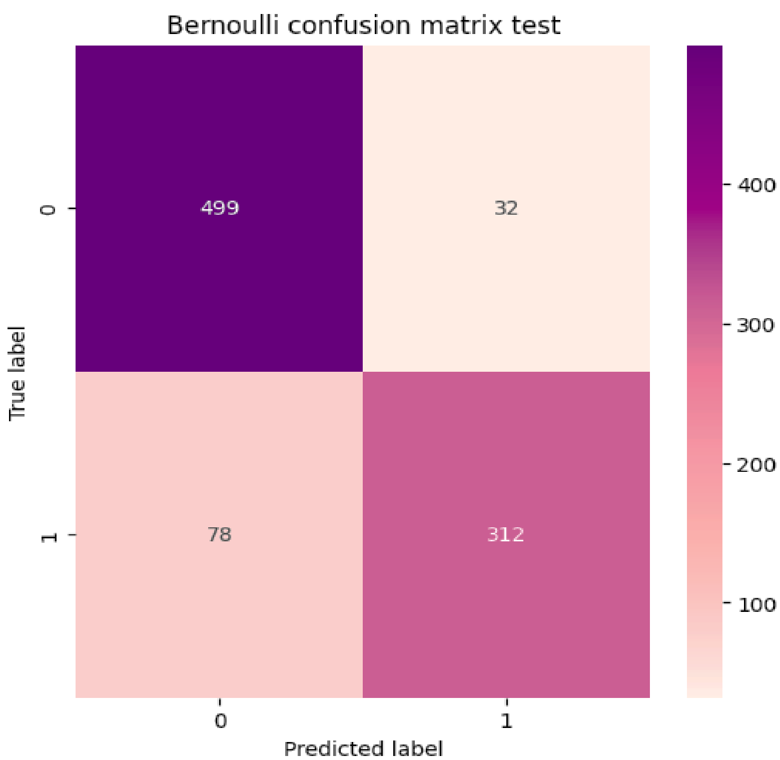
	precision	recall	f1-score	support
0	0.80	0.84	0.82	531
1	0.76	0.72	0.74	390
accuracy			0.79	921
macro avg	0.78	0.78	0.78	921
weighted avg	0.79	0.79	0.79	921

```
Multinomial Naive Bayes Classifiers test acc_sco:
 0.7861020629750272
precision of Multinomial Naive Bayes Classifiers
 0.7643835616438356
recall of Multinomial Naive Bayes Classifiers
 0.7153846153846154
f1_score of Multinomial Naive Bayes Classifiers
 0.7390728476821192
```

Part (c): Bernoulli Naive Bayes Classifier

```
#Bernoulli Naive Bayes Classifiers
#The selected model is Bernoulli Naive Bayes Classifiers
ber=BernoulliNB()
ber.fit(X_train,y_train)

#test confusion matrix
yb_test_Pred=ber.predict(X_test)
testb_confusion_matrix = confusion_matrix(y_test,yb_test_Pred)

#test accuracy
testb_score=accuracy_score(y_test,yb_test_Pred)
```

Confusion Matrices



Classification Report

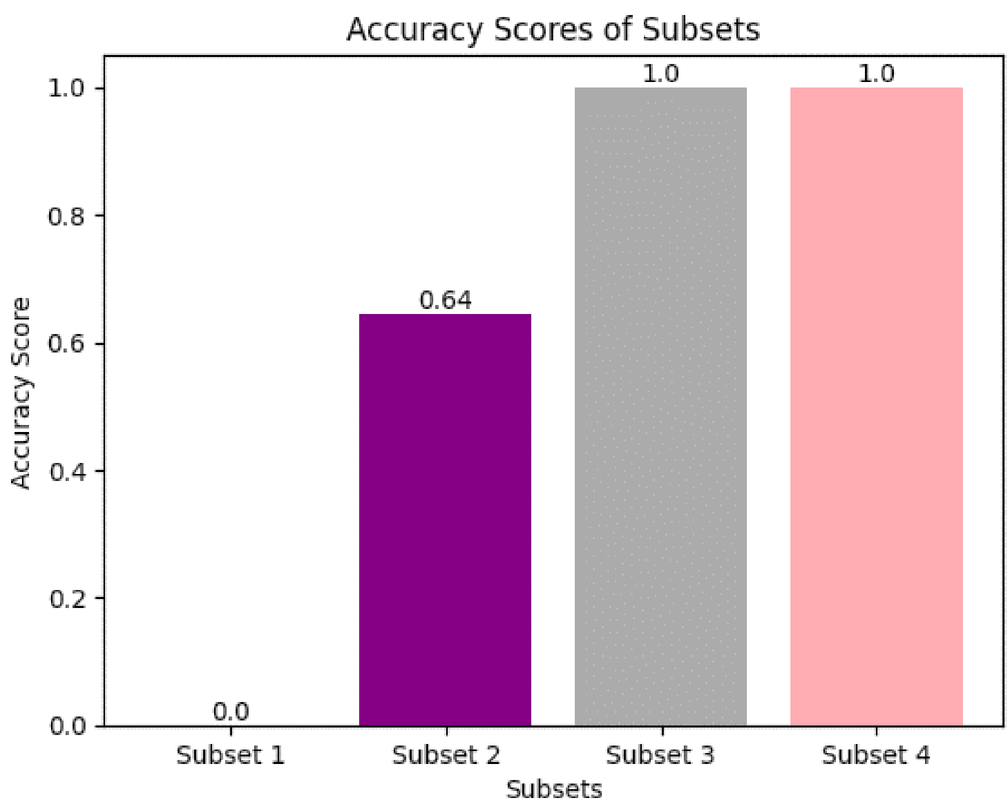
	precision	recall	f1-score	support
0	0.86	0.94	0.90	531
1	0.91	0.80	0.85	390
accuracy			0.88	921
macro avg	0.89	0.87	0.88	921
weighted avg	0.88	0.88	0.88	921

```
Bernoulli Naive Bayes Classifiers test acc_sco
 0.8805646036916395
precision of Bernoulli Naive Bayes Classifiers
 0.9069767441860465
recall of Bernoulli Naive Bayes Classifiers
 0.8
f1_score of Bernoulli Naive Bayes Classifiers
 0.8501362397820164
```

- The Bernoulli Naive Bayes classifier achieved the highest test accuracy score, followed by the Gaussian Naive Bayes classifier and then the Multinomial Naive Bayes classifier. This suggests that the data may be better represented as binary features, and the presence or absence of certain features may have more discriminatory power for classification than considering their frequencies or assuming a Gaussian distribution

Part (d): Subset Evaluation

```
Subset 1 Accuracy Score: 0.0
Subset 2 Accuracy Score: 0.6438653637350705
Subset 3 Accuracy Score: 1.0
Subset 4 Accuracy Score: 1.0
```



Subsets 3 and 4 appear to contain more representative and discriminative characteristics, which improves classification accuracy. In order to boost Subset 1's classification performance, more research and perhaps even more characteristics may be required.

Summary:

The report demonstrates the application of Naive Bayes classifiers on the spam base dataset, analyzing their performance metrics and presenting confusion matrices. It also explores the performance of Bernoulli Naive Bayes classifiers on subsets of the training data, highlighting the potential for improved classification.