Practical Number: 3A

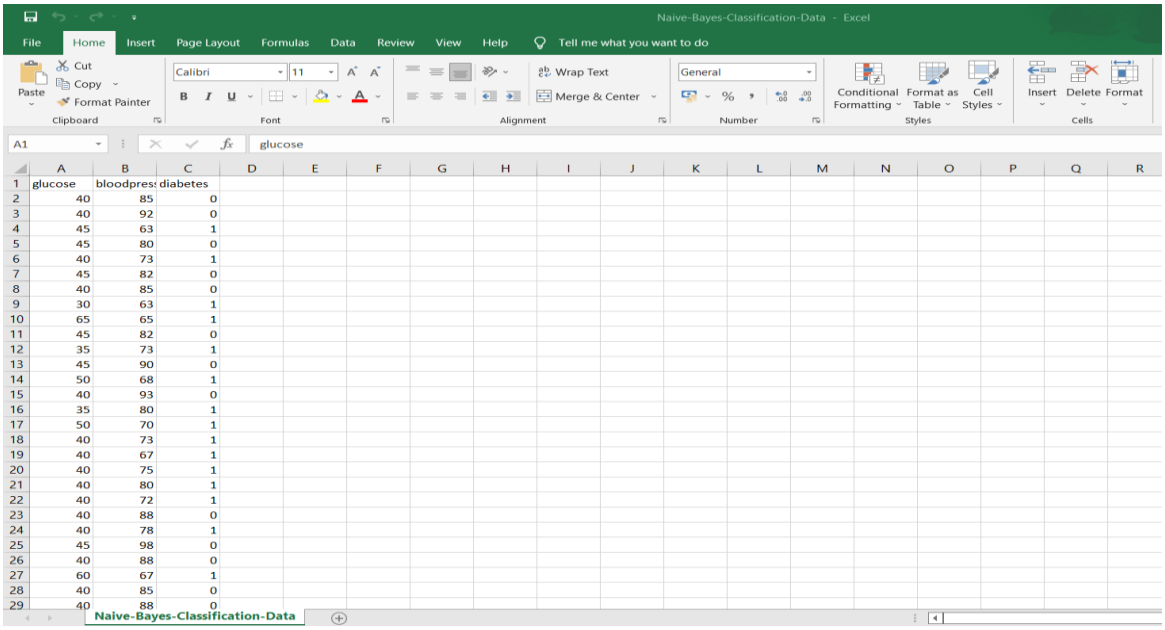
Aim: Write a program to implement the naïve Bayesian classifier for a sample training data set stored as a .CSV file. Compute the accuracy of the classifier, considering few test data sets.

Code:

```
import numpy as np

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.naive_bayes import GaussianNB

from sklearn.metrics import accuracy_score

import matplotlib.pyplot as plt


df = pd.read_csv('Naive-Bayes-Classification-Data.csv')


#Data pre-processing step

#Here, we'll create the x and y variables by taking them from the dataset

#and using the train_test_split function of scikit-learn to split the data into training and test sets.

#Note that the test size of 0.25 indicates we've used 25% of the data for testing.

#random_state ensures reproducibility. For the output of train_test_split, we get x_train, x_test, y_train, and y_test values.


x=df.drop('diabetes',axis=1)

y=df['diabetes']

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=42)


#Train the model

#We're going to use x_train and y_train, obtained above, to train our naive Bayes classifier model.

#We're using the fit method and passing the parameters.
```

```
model=GaussianNB()
model.fit(x_train,y_train)


#Prediction
#Once the model is trained, it's ready to make predictions.
#We can use the predict method on the model and pass x_test as a parameter to get the output as y_pred.
#Notice that the prediction output is an array of real numbers corresponding to the input array.


y_pred=model.predict(x_test)
print(y_pred)


#Model Evaluation
#Finally, we need to check to see how well our model is performing on the test data.
#We evaluate our model by finding the accuracy score produced by the model.


accuracy=accuracy_score(y_test,y_pred)*100
print(accuracy)
```

Naive-Bayes-Classification-Data.csv file:

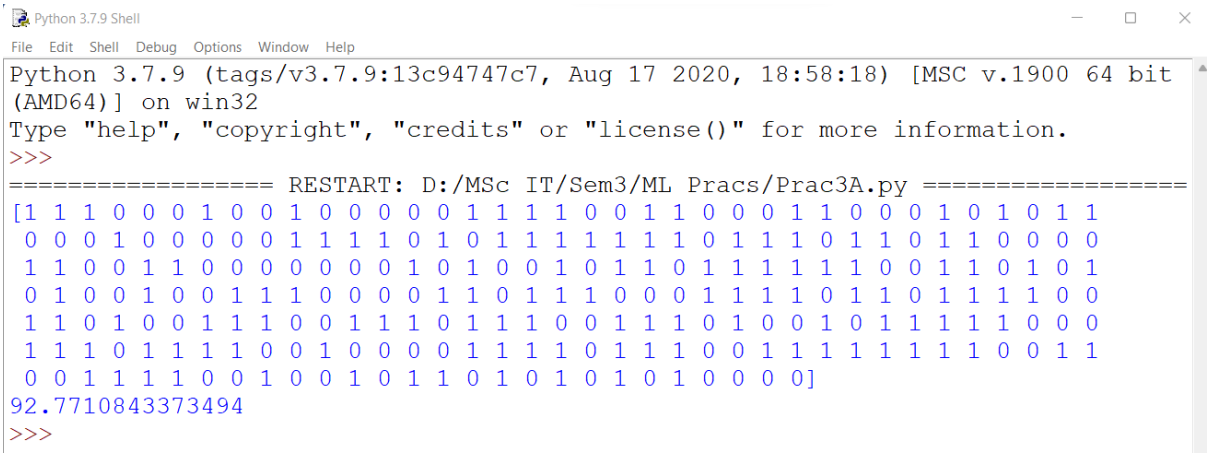


Naive-Bayes-Classification-Data - Excel

	A	B	C
1	glucose	bloodpres	diabetes
2	40	85	0
3	40	92	0
4	45	63	1
5	45	80	0
6	40	73	1
7	45	82	0
8	40	85	0
9	30	63	1
10	65	65	1
11	45	82	0
12	35	73	1
13	45	90	0
14	50	68	1
15	40	93	0
16	35	80	1
17	50	70	1
18	40	73	1
19	40	67	1
20	40	75	1
21	40	80	1
22	40	72	1
23	40	88	0
24	40	78	1
25	45	98	0
26	40	88	0
27	60	67	1
28	40	85	0
29	40	88	0

Naive-Bayes-Classification-Data

Output:



```
Python 3.7.9 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.7.9 (tags/v3.7.9:13c94747c7, Aug 17 2020, 18:58:18) [MSC v.1900 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: D:/MSc IT/Sem3/ML Pracs/Prac3A.py ==========
[1 1 1 0 0 0 1 0 0 1 0 0 0 0 0 1 1 1 1 0 0 1 1 0 0 0 1 1 0 0 0 1 0 1 0 1 1
 0 0 0 1 0 0 0 0 0 1 1 1 1 0 1 0 1 1 1 1 1 1 1 0 1 1 1 0 1 1 0 1 1 0 0 0 0
 1 1 0 0 1 1 0 0 0 0 0 0 0 1 0 1 0 0 1 0 1 1 0 1 1 1 1 1 1 0 0 1 1 0 1 0 1
 0 1 0 0 1 0 0 1 1 1 0 0 0 0 1 1 0 1 1 1 0 0 0 1 1 1 1 0 1 1 0 1 1 1 1 1 0 0
 1 1 0 1 0 0 1 1 1 0 0 1 1 1 0 1 1 1 0 0 1 1 1 0 1 0 0 1 0 1 1 1 1 1 0 0 0
 1 1 1 0 1 1 1 1 0 0 1 0 0 0 0 1 1 1 1 0 1 1 1 0 0 1 1 1 1 1 1 1 1 0 0 1 1
 0 0 1 1 1 1 0 0 1 0 0 1 0 1 1 0 1 0 1 0 1 0 1 0 0 0 0]
92.7710843373494
>>>
```

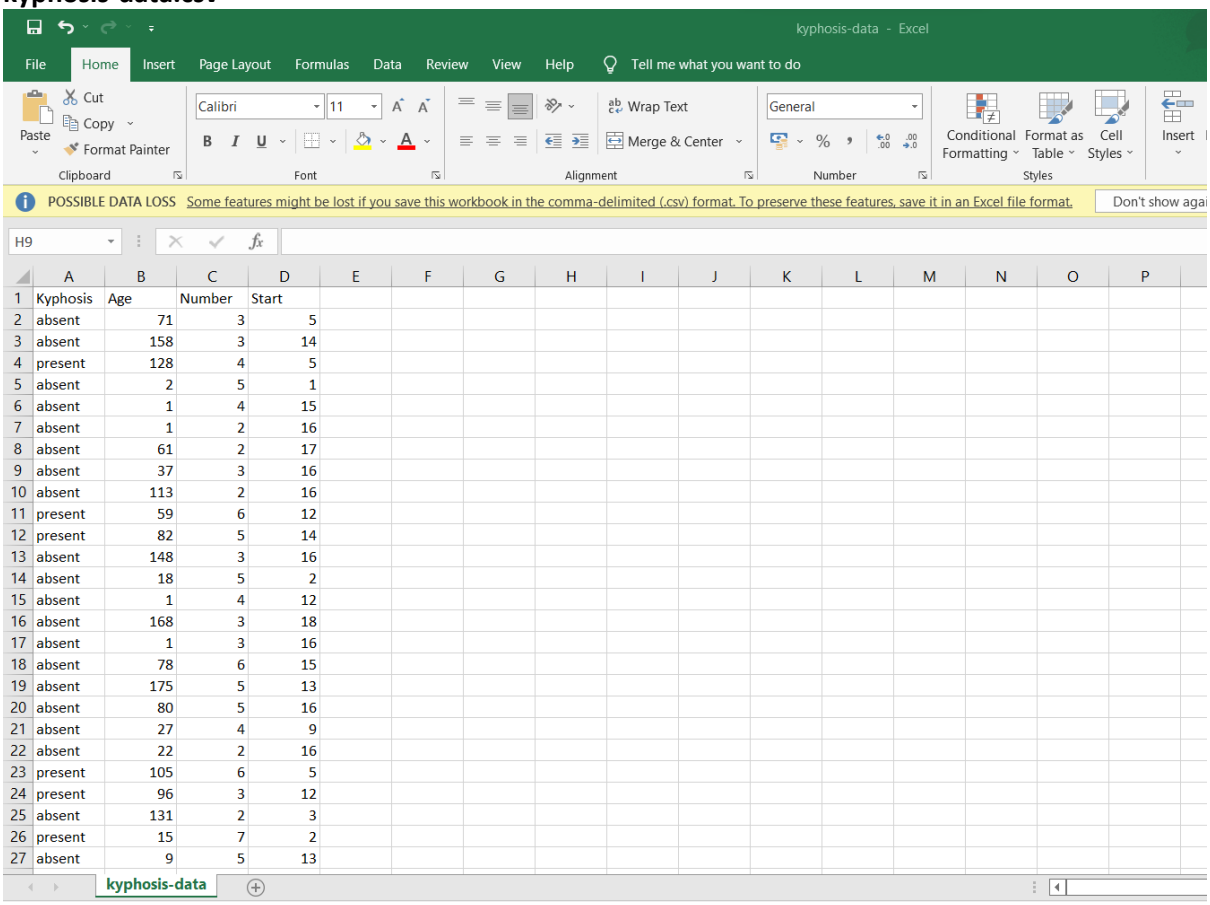
PRACTICAL NUMBER: 3B

Aim: Write a program to implement Decision Tree and Random forest with Prediction, Test Score and Confusion Matrix**.**

Code:

```
#Numerical computing libraries
import pandas as pd
import numpy as np
#Visalization libraries
import matplotlib.pyplot as plt
import seaborn as sns

raw_data = pd.read_csv('kyphosis-data.csv')
raw_data.columns

#Exploratory data analysis
raw_data.info()
sns.pairplot(raw_data, hue = 'Kyphosis')
plt.show()
#Split the data set into training data and test data
from sklearn.model_selection import train_test_split
x = raw_data.drop('Kyphosis', axis = 1)
y = raw_data['Kyphosis']
x_training_data, x_test_data, y_training_data, y_test_data = train_test_split(x, y, test_size =
0.3)
#Train the decision tree model
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier()
model.fit(x_training_data, y_training_data)
predictions = model.predict(x_test_data)
#Measure the performance of the decision tree model
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix

print('Performance of decision tree model:')
print(classification_report(y_test_data, predictions))
print('Confusion matrix of decision tree model:')
print(confusion_matrix(y_test_data, predictions))

#Train the random forests model
from sklearn.ensemble import RandomForestClassifier
random_forest_model = RandomForestClassifier()
random_forest_model.fit(x_training_data, y_training_data)
random_forest_predictions = random_forest_model.predict(x_test_data)

#Measure the performance of the random forest model
print('Performance of random forest model:')
print(classification_report(y_test_data, random_forest_predictions))
print('Confusion matrix of random forest model:')
print(confusion_matrix(y_test_data, random_forest_predictions))
```

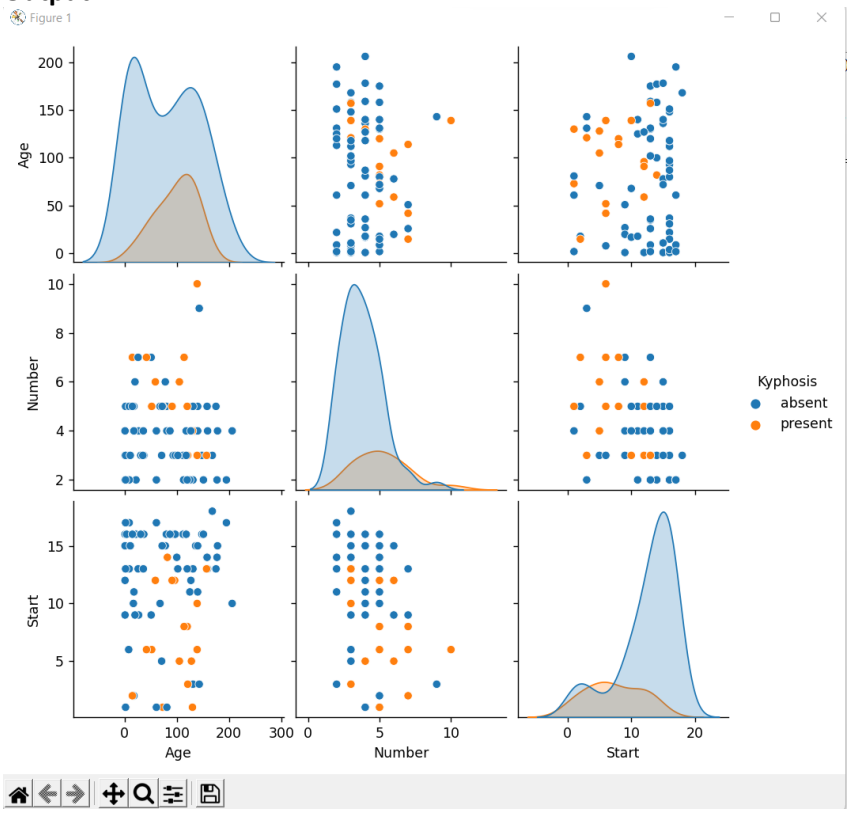
kyphosis-data.csv

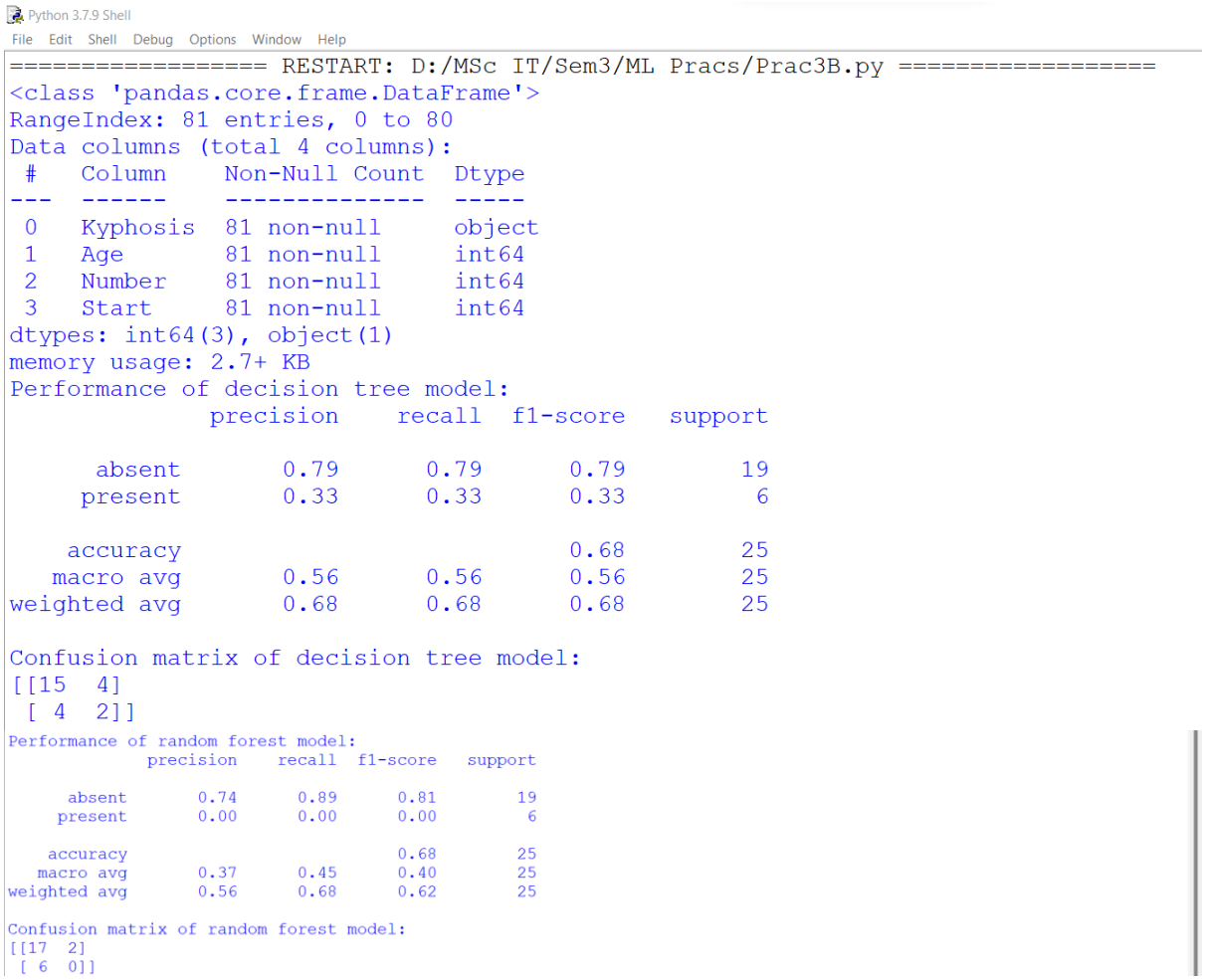


kyphosis-data - Excel

	A	B	C	D
1	Kyphosis	Age	Number	Start
2	absent	71	3	5
3	absent	158	3	14
4	present	128	4	5
5	absent	2	5	1
6	absent	1	4	15
7	absent	1	2	16
8	absent	61	2	17
9	absent	37	3	16
10	absent	113	2	16
11	present	59	6	12
12	present	82	5	14
13	absent	148	3	16
14	absent	18	5	2
15	absent	1	4	12
16	absent	168	3	18
17	absent	1	3	16
18	absent	78	6	15
19	absent	175	5	13
20	absent	80	5	16
21	absent	27	4	9
22	absent	22	2	16
23	present	105	6	5
24	present	96	3	12
25	absent	131	2	3
26	present	15	7	2
27	absent	9	5	13

Output:





```
Python 3.7.9 Shell
File  Edit  Shell  Debug  Options  Window  Help
========== RESTART: D:/MSc IT/Sem3/ML Pracs/Prac3B.py ==========
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 81 entries, 0 to 80
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Kyphosis  81 non-null     object
 1   Age       81 non-null     int64
 2   Number    81 non-null     int64
 3   Start     81 non-null     int64
dtypes: int64(3), object(1)
memory usage: 2.7+ KB
Performance of decision tree model:
              precision    recall  f1-score   support

      absent       0.79      0.79      0.79        19
     present       0.33      0.33      0.33         6

    accuracy                           0.68        25
   macro avg       0.56      0.56      0.56        25
weighted avg       0.68      0.68      0.68        25

Confusion matrix of decision tree model:
[[15  4]
 [ 4  2]]
Performance of random forest model:
              precision    recall  f1-score   support

      absent       0.74      0.89      0.81        19
     present       0.00      0.00      0.00         6

    accuracy                           0.68        25
   macro avg       0.37      0.45      0.40        25
weighted avg       0.56      0.68      0.62        25

Confusion matrix of random forest model:
[[17  2]
 [ 6  0]]
```

1. **Explain Bayesian Algorithm .**

2. **Compute the accuracy of classifier with one example**