Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

A.Y. 2022-2023

DATA MINING AND WAREHOUSE

AYUSH JAIN

COMPUTER ENGINEERING | TE – B2 | 60004200132

EXPERIMENT – 3

AIM: Implementation of Classification algorithm Using

1. Decision Tree ID3 and
2. Naïve Bayes algorithm

THEORY:

A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems. It is mainly used in text classification that includes a high dimensional training dataset.

Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.

It is a probabilistic classifier, which means it predicts on the basis of the probability of an object. Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

CODE:

```
import io
import pandas as pd
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.naive_bayes import GaussianNB
from sklearn import tree
from sklearn.datasets import load_iris
from sklearn.datasets import load_digits
from sklearn.datasets import load_winequality-red
from sklearn.datasets import load_breast_cancer
from sklearn.datasets import fetch_olivetti_faces
from sklearn.model_selection import train_test_split

#split original DataFrame into training and testing sets
X, y = fetch_olivetti_faces(return_X_y=True)
features, testfeatures, target, testtarget = train_test_split(X, y,
test_size=0.2, random_state=0)

# train, test = train_test_split(df, test_size=0.2, random_state=0)
algo = GaussianNB()
algo = tree.DecisionTreeClassifier()
algo.fit(features, target)

# print(algo.score(features, target))
print(algo.predict(testfeatures))

# print(testtarget)
print(f'The {algo} Has Achieved %.2f Percent
Accuracy'%(algo.score(features, target)))

# confusion matrix
print(confusion_matrix(algo.predict(testfeatures),testtarget))

import matplotlib.pyplot as plt
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import RocCurveDisplay
from sklearn.datasets import load_wine
from sklearn.model_selection import train_test_split
```

```
X, y = fetch_olivetti_faces(return_X_y=True)
y = y == 2
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42)
svc = SVC(random_state=42)
svc.fit(X_train, y_train)
svc_disp = RocCurveDisplay.from_estimator(svc, X_test, y_test)
plt.show()
from sklearn.model_selection import train_test_split, KFold
k_folds = KFold(n_splits = 6, shuffle = True, random_state = 42)
sum = 0
for train_index, val_index in k_folds.split(features):

# Splitting the training set from the validation set for this specific
fold
X_train, X_val = features[train_index, :], features[val_index, :]
y_train, y_val = target[train_index], target[val_index]
rfc_model = GaussianNB()
rfc_model = tree.DecisionTreeClassifier()

# Fitting the X_train and y_train datasets to the
RandomForestClassifier model
rfc_model.fit(X_train, y_train)

# Getting inferential predictions for the validation dataset
val_preds = rfc_model.predict(X_val)

# Generating validation metrics by comparing the inferential
predictions (val_preds) to the actuals (y_val)
val_accuracy = accuracy_score(y_val, val_preds)
val_confusion_matrix = confusion_matrix(y_val, val_preds)
sum = sum + val_accuracy

# Printing out the validation metrics
print(f'Accuracy Score: {val_accuracy}')
print(f'Confusion Matrix: \n{val_confusion_matrix}')
print(f'\n\nAverage Accuracy Score: {sum/6}')
```

Part A:

Program using inbuilt functions.

Predict class of unseen samples.

Results should display

1. Confusion matrix
2. Classifier accuracy

Naive Bayes Algorithm Confusion matrix and Classifier:

- Iris-dataset

```
0.95
The GaussianNB() Has Achieved 0.95 Percent Accuracy
[[11  0  0]
 [ 0 13  1]
 [ 0  0  5]]
```

- Digits

```
0.8559498956158664
The GaussianNB() Has Achieved 0.86 Percent Accuracy
[[27  0  0  0  0  0  0  0  0  0]
 [ 0 31  7  0  1  1  0  0  5  2]
 [ 0  0 17  1  0  0  0  0  0  0]
 [ 0  0  0 24  0  0  0  0  0  3]
 [ 0  0  0  0 22  0  0  0  0  0]
 [ 0  0  0  0  0 35  0  0  1  0]
 [ 0  0  0  0  0  0 44  0  0  0]
 [ 0  0  0  0  7  3  0 39  1  3]
 [ 0  4 12  4  0  0  0  0 32  7]
 [ 0  0  0  0  0  1  0  0  0 26]]
```

- Breast Cancer Dataset

```
[0 1 1 1 1 1 1 1 1 0 1 1 0 0 0 1 0 0 0 0 0 1 1 0 1 1 0 1 0 1 0 1 0 1 0 1
 0 1 0 1 1 0 1 0 0 1 1 1 0 0 0 0 1 1 1 1 1 1 0 0 0 1 1 0 1 0 0 0 1 1 0 1 1
 0 1 1 1 1 1 0 0 0 1 0 1 1 1 0 0 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 0 1 0 1 1 0 1
 0 0 1]
The GaussianNB() Has Achieved 0.95 Percent Accuracy
[[43  4]
 [ 4 63]]
```

- Fetch_olivetti_faces

```
[13 23 27 19 24 31 23 26 14 21 26 13 22 16  1  5  9 14 39 19 15 27 12 34
  0 31  7  1 28 10 22  1 33 22 35  7  9 12 22  0 31 32  0 14 29  5 37  4
  3 36  0 14  9  9 28 31  5 14  8  4  7 27 25 35 19 37 14  7 26 31 35 13
 35 13 10 29 36 30 36 32]
The DecisionTreeClassifier() Has Achieved 1.00 Percent Accuracy
[[1 0 0 ... 0 0 1]
 [0 2 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 2 0 0]
 [0 0 0 ... 0 2 0]
 [0 0 0 ... 0 0 0]]
```

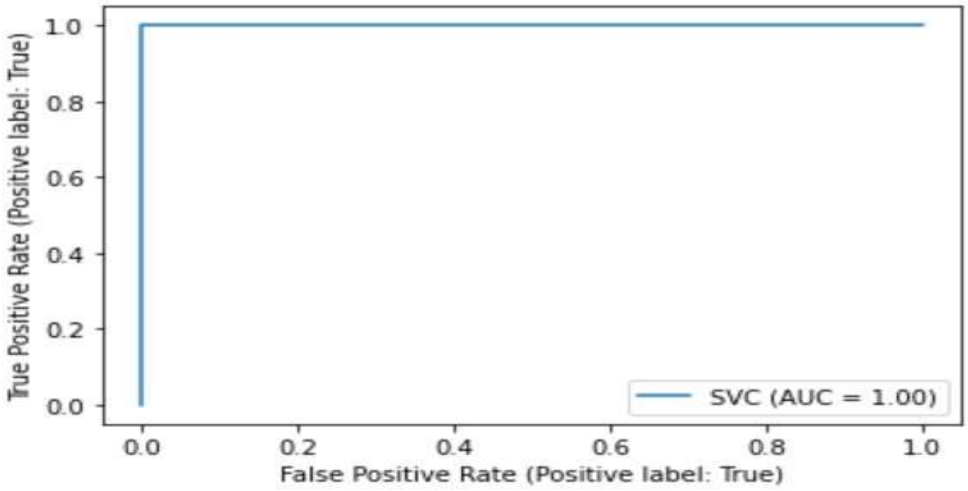
- Winequality-red

```
0.547302580140735
The Gaussian Model Has Achieved 0.55 Percent Accuracy
array([[ 0,  0,  2,  0,  1,  0],
       [ 0,  0,  5,  3,  0,  0],
       [ 1,  7, 86, 33,  0,  0],
       [ 1,  3, 34, 73, 11,  0],
       [ 0,  0,  8, 29, 13,  3],
       [ 0,  1,  0,  4,  2,  0]])
```

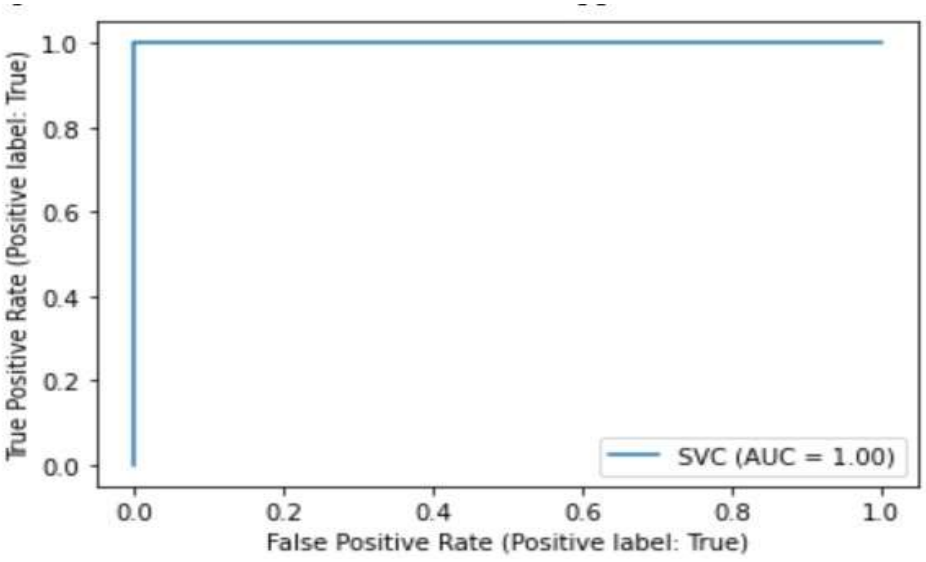
Part B:

1. Compare results of DT and ND for 5 datasets.
2. Plot AUROC
3. Plot comparison graphs using the results of DT and NB

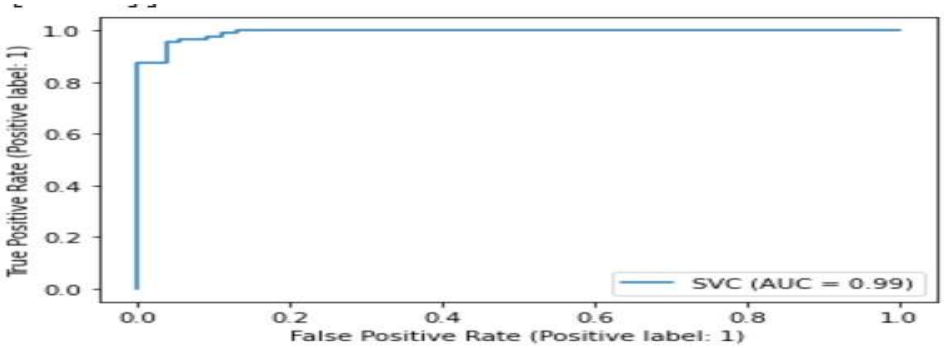
- Iris-dataset



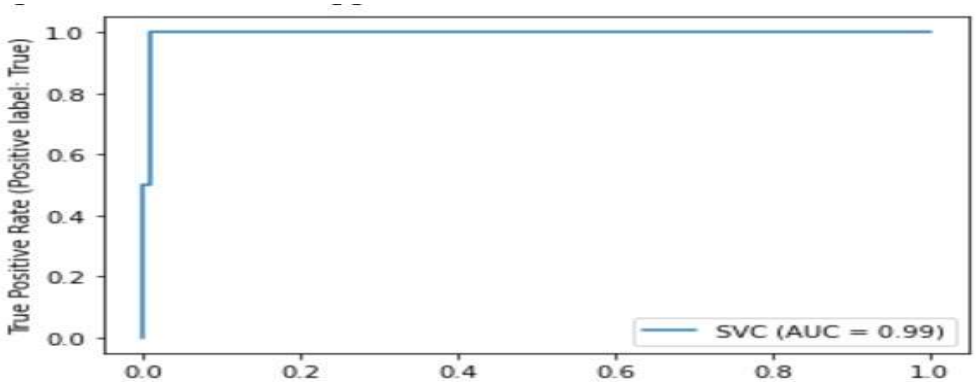
- Digits



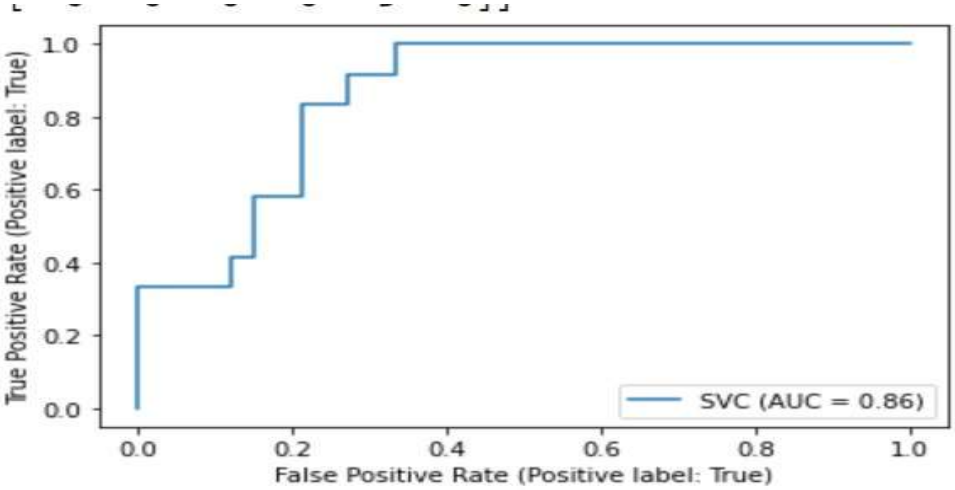
- Breast Cancer Dataset



- Fetch_olivetti_faces



- Winequality-red



Part C:

Modify DT/NB to use k-fold cross validation and ensemble models

Naive Bayes Algorithm K-folds

- Iris-dataset

```
Accuracy Score: 0.85
Confusion Matrix:
[[ 3  0  0]
 [ 0  3  1]
 [ 0  2 11]]
Accuracy Score: 1.0
Confusion Matrix:
[[6 0 0]
 [0 7 0]
 [0 0 7]]
Accuracy Score: 0.95
Confusion Matrix:
[[ 7  0  0]
 [ 0 10  0]
 [ 0  1  2]]
Accuracy Score: 0.95
Confusion Matrix:
[[ 7  0  0]
 [ 0  2  1]
 [ 0  0 10]]
Accuracy Score: 1.0
Confusion Matrix:
[[9 0 0]
 [0 5 0]
 [0 0 6]]
Accuracy Score: 0.95
Confusion Matrix:
[[7 0 0]
 [0 7 1]
 [0 0 5]]


Average Accuracy Score: 0.9500000000000001
```

- Digits

```
Confusion Matrix:
[[28  0  0  0  1  0  0  0  0  0]
 [ 0 17  0  0  0  0  0  0  2  3]
 [ 0  2 15  0  0  0  1  0  4  2]
 [ 0  0  0 18  0  1  0  3  3  0]
 [ 0  1  0  0 19  0  0  5  1  0]
 [ 0  0  0  0  0 21  0  1  1  0]
 [ 0  1  0  0  0  0 20  0  1  0]
 [ 0  1  0  0  0  0  0 21  0  0]
 [ 0  4  1  0  0  0  0  1 21  0]
 [ 0  1  0  0  0  0  0  1  3 15]]
Accuracy Score: 0.8791666666666667
Confusion Matrix:
[[23  0  0  0  0  0  0  0  0  1]
 [ 0 30  0  0  0  0  0  0  1  0]
 [ 0  1 17  0  0  0  0  0  9  0]
 [ 0  1  0 25  0  0  0  1  0  0]
 [ 0  2  0  0 22  0  0  1  0  0]
 [ 0  0  0  1  0 26  0  0  0  0]
 [ 0  0  0  0  0  0 25  0  0  0]
 [ 0  0  0  0  0  0  0 19  0  0]
 [ 0  5  0  0  0  1  0  0 16  0]
 [ 0  2  0  0  1  0  0  1  1  8]]
```

```
Confusion Matrix:
[[21  0  0  0  0  0  0  0  0  0]
 [ 0 19  1  0  0  0  0  1  0  1]
 [ 0  1 15  0  1  0  0  0  2  0]
 [ 0  0  1 22  0  0  0  0  1  1]
 [ 0  0  0  0 20  0  1  3  0  0]
 [ 0  0  0  1  0 14  0  1  0  0]
 [ 0  0  0  0  0  1 21  0  0  0]
 [ 0  0  1  0  0  1  0 32  0  0]
 [ 0  2  0  1  0  0  0  0 21  0]
 [ 0  1  0  3  1  1  1  4  3 18]]
Accuracy Score: 0.8493723849372385
Confusion Matrix:
[[27  0  0  0  0  0  0  0  0  0]
 [ 0 27  0  0  0  0  0  0  2  0]
 [ 0  3 15  0  0  0  0  0  7  0]
 [ 0  0  0 17  0  1  0  1  5  0]
 [ 0  0  0  0 22  2  0  3  0  0]
 [ 0  1  0  0  0 23  1  1  0  0]
 [ 0  0  0  0  0  0 20  0  0  0]
 [ 0  0  0  0  0  0  0 22  0  0]
 [ 0  1  0  0  0  0  0  0 16  0]
 [ 0  2  0  0  0  1  0  2  3 14]]
```

```
Average Accuracy Score: 0.8476202928870293
```

- Breast Cancer Dataset

```
Accuracy Score: 0.9605263157894737
Confusion Matrix:
[[30  3]
 [ 0 43]]
Accuracy Score: 0.9078947368421053
Confusion Matrix:
[[21  5]
 [ 2 48]]
Accuracy Score: 0.9605263157894737
Confusion Matrix:
[[19  2]
 [ 1 54]]
Accuracy Score: 0.881578947368421
Confusion Matrix:
[[31  5]
 [ 4 36]]
Accuracy Score: 0.9342105263157895
Confusion Matrix:
[[24  2]
 [ 3 47]]
Accuracy Score: 0.9733333333333334
Confusion Matrix:
[[21  2]
 [ 0 52]]


Average Accuracy Score: 0.936345029239766
```

- Fetch_olivetti_faces

```
Accuracy Score: 0.46296296296296297
Confusion Matrix:
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 2 ... 0 0 0]
 ...
 [0 0 0 ... 2 0 0]
 [0 0 0 ... 0 3 0]
 [0 0 0 ... 0 0 0]]
Accuracy Score: 0.46296296296296297
Confusion Matrix:
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 2 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 2]]
Accuracy Score: 0.6037735849056604
Confusion Matrix:
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 1 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 2 0]
 [0 0 0 ... 0 0 1]]
```

```
Accuracy Score: 0.4716981132075472
Confusion Matrix:
[[0 0 0 ... 0 0 0]
 [0 1 0 ... 0 0 0]
 [0 0 1 ... 1 0 0]
 ...
 [0 0 0 ... 1 0 0]
 [0 0 0 ... 0 1 0]
 [0 0 0 ... 0 0 1]]
Accuracy Score: 0.5094339622641509
Confusion Matrix:
[[0 0 0 ... 1 0 0]
 [0 1 0 ... 0 0 0]
 [0 0 1 ... 0 0 1]
 ...
 [0 0 0 ... 1 0 0]
 [0 0 0 ... 0 2 0]
 [0 0 0 ... 0 0 0]]
Accuracy Score: 0.41509433962264153
Confusion Matrix:
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 1]
 [0 0 1 ... 0 0 0]
 ...
 [0 0 0 ... 1 0 0]
 [0 0 0 ... 0 0 0]
 [1 0 0 ... 0 0 1]]
```

Average Accuracy Score: 0.4876543209876543

- Winequality-red

```
Accuracy Score: 0.5747663551401869
Confusion Matrix:
[[ 0  0  1  0  0  0]
 [ 0  0  5  0  0  0]
 [ 0  6 50 20  8  0]
 [ 0  4 21 50 12  0]
 [ 0  1  1 11 22  0]
 [ 0  0  0  1  0  1]]
Accuracy Score: 0.6244131455399061
Confusion Matrix:
[[ 0  1  1  2  0  0]
 [ 0  1  1  5  1  0]
 [ 0  3 72 19  4  0]
 [ 0  1 15 45  7  1]
 [ 0  0  2 13 15  2]
 [ 0  0  0  1  1  0]]
Accuracy Score: 0.6338028169014085
Confusion Matrix:
[[ 0  0  1  0  0  0]
 [ 0  0  6  4  0  0]
 [ 0  3 72 20  1  0]
 [ 0  3 17 46  9  1]
 [ 0  0  0  9 16  0]
 [ 0  0  0  3  1  1]]
Accuracy Score: 0.5727699530516432
```

```
Confusion Matrix:
[[ 0  0  1  0  0  0]
 [ 0  0  1  1  1  0]
 [ 0  1 61 28  4  0]
 [ 0  1 21 53 11  2]
 [ 0  0  3 13  7  2]
 [ 0  0  1  0  0  1]]
Accuracy Score: 0.6150234741784038
Confusion Matrix:
[[ 0  0  0  0  0  0]
 [ 0  2  4  4  0  0]
 [ 2  2 64 18  2  0]
 [ 0  1 25 48  8  1]
 [ 0  0  3  9 17  0]
 [ 0  0  0  2  1  0]]
Accuracy Score: 0.6291079812206573
Confusion Matrix:
[[ 0  0  1  0  0  0]
 [ 0  1  4  0  1  0]
 [ 0  1 65 17  3  0]
 [ 0  0 29 55  9  0]
 [ 0  0  7  6 13  0]
 [ 0  0  0  0  1  0]]


Average Accuracy Score: 0.608313954338701
```

CONCLUSION: Hence, we implemented Classifier algorithms Naive Bayes, ID3 in Python and verified the results.