PROGRAM 6 : NAIVE BAYES GAUSSIAN NB

[1]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.naive_bayes import GaussianNB
```

[2]:
```
df=pd.read_csv("iris_data.csv")
```

[3]:
```
y=df['class']
x=df.drop('class',axis=1)
```

[4]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
print(len(x_train))
print(len(x_test))
```

```
120
30
```

[5]:
```
sc=StandardScaler()
x_train=sc.fit_transform(x_train)
x_test=sc.fit_transform(x_test)
```

[6]:
```
nb=GaussianNB()
nb.fit(x_train,y_train)
```

[6]:
```
GaussianNB
GaussianNB()
```

[7]:
```
y_pred=nb.predict(x_test)
print(y_pred)
```

```
['Iris-versicolor' 'Iris-setosa' 'Iris-virginica' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-setosa' 'Iris-versicolor' 'Iris-virginica'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica' 'Iris-setosa'
 'Iris-setosa' 'Iris-setosa' 'Iris-setosa' 'Iris-versicolor'
 'Iris-virginica' 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica'
 'Iris-setosa' 'Iris-versicolor' 'Iris-setosa' 'Iris-virginica'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-setosa' 'Iris-setosa']
```

[8]:
```
cm=confusion_matrix(y_test,y_pred)
print(cm)
accuracy=(cm[0,0]+cm[1,1]+cm[2,2])/np.sum(cm)
print(accuracy)
```

```
[[10  0  0]
 [ 0  9  0]
 [ 0  1 10]]
0.9666666666666667
```

PROGRAM 7 : K_MEANS CLUSTERING

[*]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
```

[*]:
```
df=pd.read_csv("wine-clustering.csv")
df.head()
```

[*]:
```
df.describe().T
df.info()
```

[*]:
```
sns.pairplot(df)
```

```
selected_features=df[['OD280','Alcohol']] ...
```

[*]:
```
centers=kmeans_obj.cluster_centers_
print(centers)
```

[*]:
```
sns.scatterplot(x=selected_features['OD280'],
                y=selected_features['Alcohol'],
                hue=kmeans_obj.labels_)
plt.scatter(kmeans_obj.cluster_centers_[:,0],
            kmeans_obj.cluster_centers_[:,1],
            s=200,c='red')
```

PROGRAM 8 : KNN ALGORITHM

[18]:
```
import pandas as pd
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
from sklearn.metrics import accuracy_score,confusion_matrix
import numpy as np
import seaborn as sns
from sklearn.neighbors import KNeighborsClassifier
```

[19]:
```
y=data['diagnosis']
x=data.drop(['diagnosis'],axis=1)
x
```

[20]:
```
data=pd.read_csv('breast-cancer-wisconsin-data_data (1).csv')
data.head()
```

[21]:
```
data = data.drop(["id", "Unnamed: 32"], axis=1)
data.head()
```

[22]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)

knn_cfr = KNeighborsClassifier(n_neighbors=3)
knn_cfr.fit(x_train, y_train)

y_pred = knn_cfr.predict(x_test)
```

[23]:
```
acc=accuracy_score(y_test,y_pred)
acc
```

[23]:
```
0.9298245614035088
```

PROGRAM 9 : PCA

[2]:
```
import pandas as pd
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
```

[3]:
```
df=pd.read_csv("diabetes.csv")
row,col=df.shape
print("The dimension of the dataset is: ",col)
```

```
The dimension of the dataset is:  9
```

[4]:
```
pca=PCA(n_components=5)
reduced_dim=pca.fit_transform(df)
reduced_dim=pd.DataFrame(reduced_dim,columns=['pc1','pc2','pc3','pc4','pc5'])
```

[6]:
```
row,col=reduced_dim.shape
print("The reduced dimension of the dataset is: ",col)
```

```
The reduced dimension of the dataset is:  5
```

[7]:
```
plt.figure(figsize=(8,3))
plt.scatter(reduced_dim['pc1'],reduced_dim['pc2'])
plt.show()
```

PROGRAM 10 : ENSEMBLE BASED LEARNING

[8]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
```

[9]:
```
names = [
    "K-Nearest Neighbors",
    "Linear SVM",
    "Decision Tree",
    "Multilayer Perceptron",
    "Gaussian Naive Bayes",
    "Random Forest"
]
```

[10]:
```
classifiers = [
    KNeighborsClassifier(3),
    SVC(kernel="linear", C=0.025),
    DecisionTreeClassifier(max_depth=5),
    MLPClassifier(alpha=1, max_iter=1000),
    GaussianNB(),
    RandomForestClassifier(n_estimators=100, max_depth=5, random_state=42)
]
```

[11]:
```
df = pd.read_csv("Iris_data.csv")
df.head()
```

[12]:
```
X = df.iloc[:, :-1]
X.head()
```

[13]:
```
y = df.iloc[:, -1]
y.head()
```

[14]:
```
y = LabelEncoder().fit_transform(y)
y = pd.DataFrame(y)
y.head()
```

[15]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)
```

[16]:
```
for name, clf in zip(names, classifiers):
    clf.fit(X_train, y_train.values.ravel())
    score = clf.score(X_test, y_test)
    print("Classifier Name:", name, "Score:", score)
```