1.write a program to implement the Naïve Bayes classifier using GaussianNB

import pandas as pd

import numpy as np

df = pd.read\_csv('dataset.csv')

print('dataset : \n',df)

x = df.iloc[:,0:-1].values

y = df.iloc[:,-1].values

print('features : \n',x)

print('label : \n',y)

from sklearn.model\_selection import train\_test\_split

xtrain,xtest,ytrain,ytest = train\_test\_split(x,y,test\_size=0.30,random\_state=1)

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

xtrain = sc.fit\_transform(xtrain)

xtest = sc.transform(xtest)

from sklearn.naive\_bayes import GaussianNB

clf = GaussianNB()

clf.fit(xtrain,ytrain)

ypred = clf.predict(xtest)

print(ypred)

print(clf.score(xtest,ytest))

from sklearn.metrics import confusion\_matrix

cm = confusion\_matrix(ytest,ypred)

print(cm)

from sklearn.metrics import classification\_report as cr

cr = cr(ytest,ypred)

print(cr)

import matplotlib.pyplot as plt

import seaborn as sns

sns.heatmap(cm,annot=True)

plt.show()

#dataset for Naïve Bayes :

dataset.csv

col,label

1,1

2,1

3,1

4,1

5,1

6,1

7,1

8,1

9,1

10,1

2.Write a program to implement KNearestNeighbors KNN

import pandas as pd

import numpy as np

df = pd.read\_csv('dataset.csv')

print('dataset : \n',df)

x = df.iloc[:,0:-1].values

y = df.iloc[:,-1].values

print('features : \n',x)

print('label : \n',y)

from sklearn.model\_selection import train\_test\_split

xtrain,xtest,ytrain,ytest = train\_test\_split(x,y,test\_size=0.30,random\_state=1)

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

xtrain = sc.fit\_transform(xtrain)

xtest = sc.transform(xtest)

from sklearn.neighbors import KNeighborsClassifier

clf = KNeighborsClassifier()

clf.fit(xtrain,ytrain)

ypred = clf.predict(xtest)

print(ypred)

print(clf.score(xtest,ytest))

from sklearn.metrics import confusion\_matrix

cm = confusion\_matrix(ytest,ypred)

print(cm)

from sklearn.metrics import classification\_report as cr

cr = cr(ytest,ypred)

print(cr)

import matplotlib.pyplot as plt

import seaborn as sns

sns.heatmap(cm,annot=True)

plt.show()

# data set for KNN

dataset.csv

col,label

1,1

2,1

3,1

4,1

5,1

6,1

7,1

8,1

9,1

10,1

3.write a program to implement SVM

# Support Vector Machin svm Classifier SVC

from sklearn.svm import SVC

# sample dataset

from sklearn.datasets import load\_iris

df = load\_iris()

x = df.data

y = df.target

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test = train\_test\_split(x,y,test\_size=0.3,random\_state=1)

model = SVC(kernel='linear')

model.fit(x\_train,y\_train)

ypred = model.predict(x\_test)

print(model.score(x\_test,y\_test))

import sklearn.metrics as m

print (m.accuracy\_score(y\_test,ypred))

cm = m.confusion\_matrix(y\_test,ypred)

print (m.classification\_report(y\_test,ypred))

import seaborn as sns

import matplotlib.pyplot as plt

sns.heatmap(cm,annot=True)

plt.show()

# SVM dataset given in datasets library

4.write a program to implement Decision Tree

import pandas as pd

import numpy as np

df = pd.read\_csv('dataset.csv')

print('dataset : \n',df)

x = df.iloc[:,0:-1].values

y = df.iloc[:,-1].values

print('features : \n',x)

print('label : \n',y)

from sklearn.model\_selection import train\_test\_split

xtrain,xtest,ytrain,ytest = train\_test\_split(x,y,test\_size=0.30,random\_state=1)

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

xtrain = sc.fit\_transform(xtrain)

xtest = sc.transform(xtest)

from sklearn.tree import DecisionTreeClassifier

clf = DecisionTreeClassifier()

clf.fit(xtrain,ytrain)

ypred = clf.predict(xtest)

print(ypred)

print(clf.score(xtest,ytest))

from sklearn.metrics import confusion\_matrix

cm = confusion\_matrix(ytest,ypred)

print(cm)

from sklearn.metrics import classification\_report as cr

cr = cr(ytest,ypred)

print(cr)

import matplotlib.pyplot as plt

import seaborn as sns

sns.heatmap(cm,annot=True)

plt.show()

5.Logistic Regression

'''

Logistic Regression

sigmoid function = 1/1+e\*\*(-x)

'''

import numpy as np

import pandas as pd

df = pd.read\_csv('user.csv')

f = ['Age','EstimatedSalary']

x = df[f]

y=df.Purchased

print(df.describe())

print(df.shape)

from sklearn.model\_selection import train\_test\_split

xtrain,xtest,ytrain,ytest = train\_test\_split(x,y,test\_size=0.30,random\_state=1)

from sklearn.preprocessing import StandardScaler

ss = StandardScaler()

xtrain = ss.fit\_transform(xtrain)

xtest = ss.transform(xtest)

from sklearn.linear\_model import LogisticRegression

clf = LogisticRegression(random\_state=0)

clf.fit(xtrain,ytrain)

score = clf.score(xtrain,ytrain)

ypred = clf.predict(xtest)

print('score : ',int(score\*100),'%')

from sklearn import metrics as m

print('Accuracy : \n',m.accuracy\_score(ytest,ypred))

print('confusion\_matrix : \n',m.confusion\_matrix(ytest,ypred))

print('classification\_report : \n',m.classification\_report(ytest,ypred))

print('mean\_abosolute\_error : \n',m.mean\_absolute\_error(ytest,ypred))

print('mean\_squared\_error : \n',m.mean\_squared\_error(ytest,ypred))

print(

'sqrt mean\_squared\_error : \n',np.sqrt(m.mean\_squared\_error(ytest,ypred)))

Logistic Regression DataSet : user.csv

ID,Gender,Age,EstimatedSalary,Purchased

1,m,19,1999,0

2,m,29,2300,1

3,m,21,1999,1

4,m,22,1939,1

5,f,25,1399,1

6,f,18,1629,1

7,f,15,1320,1

8,f,24,1999,0

9,f,21,1992,0

10,m,19,1999,0

6.Multiple Linear Regression

7.Single Linear Regression

# mixture

import pandas as pd

import numpy as np

df = pd.read\_csv('dataset.csv')

print('dataset : \n',df)

x = df.iloc[:,0:-1].values

y = df.iloc[:,-1].values

print('features : \n',x)

print('label : \n',y)

from sklearn.model\_selection import train\_test\_split

xtrain,xtest,ytrain,ytest = train\_test\_split(x,y,test\_size=0.30,random\_state=1)

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

xtrain = sc.fit\_transform(xtrain)

xtest = sc.transform(xtest)

from sklearn.naive\_bayes import GaussianNB

clf = GaussianNB()

clf.fit(xtrain,ytrain)

from sklearn.neighbors import KNeighborsClassifier

clf = KNeighborsClassifier()

clf.fit(xtrain,ytrain)

from sklearn.tree import DecisionTreeClassifier

clf = DecisionTreeClassifier()

clf.fit(xtrain,ytrain)

ypred = clf.predict(xtest)

print(ypred)

print(clf.score(xtest,ytest))

from sklearn.metrics import confusion\_matrix

cm = confusion\_matrix(ytest,ypred)

print(cm)

from sklearn.metrics import classification\_report as cr

cr = cr(ytest,ypred)

print(cr)

import matplotlib.pyplot as plt

import seaborn as sns

sns.heatmap(cm,annot=True)

plt.show()