**ASSIGNMENT OF MACHINE LEARNING WITH PYTHON**

**NAME: Ramandeep Kaur**

DEGREE**: B.TECH- BIOTECHNOLOGY**

**AFFILIATION:**

**PUNJAB AGRICULTURAL UNIVERSITY, LUDHIANA**

**EMAIL- ID:** [**kaurr9435@gmail.com**](mailto:kaurr9435@gmail.com)

**CATEGORY: GENERAL**

**PROBLEM 1**

Create a dataset having following features- experience of the candidate, written score, interview score and salary. Based on three input features, HR decide the salary of the selected candidates. Using this data, build a machine learning model for HR department that can help them decide salaries of the candidates. Using the built model, predict the salaries for the following candidates:

(a) 5 Yrs experience, 8 written test score, 10 interview score

(b) 8 Yrs experience, 7 written test score, 6 interview score

**SOLUTION:**

**DATA:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | experience | test\_score | interview\_score | Salary |
| 0 | 6 | 8 | 8 | 50000 |
| 1 | 6 | 5 | 4 | 22000 |
| 2 | 3 | 6 | 5 | 30000 |
| 3 | 5 | 9 | 9 | 55000 |
| 4 | 6 | 3 | 5 | 13000 |
| 5 | 6 | 2 | 1 | 9000 |
| 6 | 10 | 8 | 6 | 48000 |
| 7 | 1 | 1 | 2 | 500 |
| 8 | 15 | 9 | 9 | 60000 |
| 9 | 13 | 7.5 | 7.5 | 45000 |
| 10 | 10 | 5 | 5 | 25000 |
| 11 | 6 | 1 | 1 | 500 |
| 12 | 1 | 3 | 6 | 25000 |
| 13 | 4 | 5 | 4 | 36000 |
| 14 | 6 | 5 | 6 | 8000 |
| 15 | 11 | 9 | 8 | 55000 |
| 16 | 12 | 9 | 9 | 58000 |
| 17 | 6 | 2 | 6 | 12000 |
| 18 | 3 | 5 | 5 | 20000 |
| 19 | 5 | 6 | 7 | 35000 |

The data is taken from Kaggle.com.

The dataset is been classified into 4 columns which represent the input the output features the prediction model will be based on the first three columns i.e. experience, test score, interview score and the output feature Salary.

The data is already in integer so it doesn’t require any pre-processing. After defining input and output features as X and Y, we apply the split on train and test data using sklearn train and test library.

The model is defined by sklearn’s Linear Regression.

The model is then fitted on the train data of input variable x\_train and output variable y\_train.

The efficiency of the model is calculated using the r\_score method. The efficiency is 96.425%

The query data is inputed in the model by the read\_csv method and predicted.

**CODE:**

import pandas as pd

import numpy as np

from sklearn.linear\_model import LinearRegression

data=pd.read\_csv('C:\\Users\\Karan\\Downloads\\\_salary\_predict\_dataset1.csv')

print(data)

data2=data.drop('Salary',axis=1)

print(data2)

x=data.iloc[:,:-1]

y=data.iloc[:,-1:]

print(x)

print(y)

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

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2,random\_state=1)

print(x\_train)

print(y\_train)

model=LinearRegression()

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

print(model.coef\_)

print(model.intercept\_)

pred= model.predict(x\_test)

print(pred)

from sklearn.metrics import r2\_score

score = r2\_score(y\_test,pred)

print(score)

#salary of candidate with 5 yrs experience,8 test score and 10 interview score

x=model.predict([[5,8,10]])

print(x)

#salary of candidate with 8 yrs experience,7 test score and 6 interview score

y=model.predict([[8,7,6]])

print(y)

**PROBLEM 2**

Create a dataset having following features- Graduations percentage, experience of the candidate, written score, interview score and selection. Selection feature is binary in nature and contains the status of the candidate. Also store at least 25 records in this dataset. Using this data, build a Logistic Regression model for HR department that can help them to decide whether the candidate will be selected or not. Take 80% data as training data and remaining a testing data randomly. Using the built model, predict the status for the following unseen data:

(a) 90 %, 5 Yrs experience, 8 written test score, 10 interview score

(b) 75%, 8 Yrs experience, 7 written test score, 6 interview score

Also calculate the possible classification metrics for the above cases and save these values in the .CSV file by executing the model at least 5 times.

**SOLUTION:**

**DATA:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Experience | Test Score | Interview Score | Graduation percentage | Selection |
| 6 | 8 | 8 | 9 | 1 |
| 6 | 5 | 4 | 7.5 | 0 |
| 3 | 6 | 5 | 7.8 | 0 |
| 5 | 9 | 9 | 9.6 | 1 |
| 6 | 3 | 5 | 6.7 | 0 |
| 6 | 2 | 1 | 5.3 | 0 |
| 10 | 8 | 6 | 7.9 | 1 |
| 1 | 1 | 2 | 4.7 | 0 |
| 15 | 9 | 9 | 9.2 | 1 |
| 13 | 7.5 | 7.5 | 8.5 | 1 |
| 10 | 5 | 5 | 7.6 | 1 |
| 6 | 1 | 1 | 6 | 0 |
| 1 | 3 | 6 | 7.1 | 0 |
| 4 | 5 | 4 | 5.9 | 0 |
| 6 | 5 | 6 | 8.6 | 1 |
| 11 | 9 | 8 | 8.9 | 0 |
| 12 | 9 | 9 | 9.5 | 1 |
| 6 | 2 | 6 | 7.7 | 0 |
| 3 | 5 | 5 | 7 | 0 |
| 5 | 6 | 7 | 7.3 | 1 |
| 7 | 6 | 5 | 8.1 | 1 |
| 6 | 8 | 4 | 8.8 | 1 |
| 12 | 4 | 7 | 6.9 | 1 |
| 5 | 9 | 9 | 9.4 | 1 |
| 2 | 5 | 2 | 7.6 | 0 |
| 9 | 7 | 5 | 7.9 | 1 |

The data is taken from Kaggle.com. Also Altered

The dataset is been classified into 5 columns which represent the input the output features the prediction model will be based on the first three columns i.e. experience, test score, interview score, graduation score and the output feature Selection.

The data has string output so in order to convert this into integer.

After defining input and output features as X and Y, we apply the split on train and test data using sklearn train and test library.

The model is defined by sklearn’s Linear Regression

The model is then fitted on the train data of input variable x\_train and output variable y\_train.

The efficiency of the model is calculated using the r\_score method. The efficiency is 100%.

The query data is inputed in the model by the read\_csv method and predicted.

**CODE:**

#predicting selection of the student

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.linear\_model import LinearRegression

data=pd.read\_csv('C:\\Users\\Karan\\Downloads\\0\_Student\_selection\_dataset.csv')

print(data)

x=data.iloc[:,:-1]

y=data.iloc[:,-1:]

print(x)

print(y)

model = LinearRegression()

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

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2,random\_state=1)

print(x\_train)

print(y\_train)

from sklearn.preprocessing import StandardScaler

sc\_x= StandardScaler()

x\_train =sc\_x.fit\_transform(x\_train)

x\_test=sc\_x.transform(x\_test)

print(x\_train)

print(x\_test)

from sklearn.linear\_model import LogisticRegression

model = LogisticRegression()

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

pred = model.predict(x\_test)

print(pred)

print(model.intercept\_)

print(model.coef\_)

from sklearn.linear\_model import LogisticRegression

classifier=LogisticRegression()

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

y\_pred=classifier.predict(x\_test)

print(y\_pred)

#computing confusion matrix

from sklearn.metrics import confusion\_matrix

c=confusion\_matrix(y\_test,y\_pred)

print('confusion\_matrix: \n',c)

p= classifier.predict\_proba(x\_test)[:,:]

print(p)

#efficiency of data

from sklearn.metrics import r2\_score

score= r2\_score(y\_test,y\_pred)

print(score)

#predicting slection with 90%, 5 yrs experience, 8 test score and 10 interview score

x=model.predict([[5,8,10,90]])

print(x)

#predicting slection with 75%, 8 yrs experience, 7 test score and 6 interview score

y=model.predict([[8,7,6,75]])

print(y)

**PROBLEM 4**

Write a python code to Implement the SVM classifier on Breast Cancer Dataset using scikit-learn. Also check the accuracy of the model.

**SOLUTION:**

**DATA:**

DATA is to big to be pasted here !!!!!

**CODE:**

#creating svm classifier model

import pandas as pd

import numpy as np

data=pd.read\_csv('C:\\Users\\Karan\\Downloads\\breast\_cancer\_data.csv')

print(data)

print(data.shape)

d1=pd.get\_dummies(data['diagnosis'],drop\_first=True)

print(d1)

data1=pd.concat([d1,data],axis=1)

print(data1)

data2=data1.drop('diagnosis',axis=1)

print(data2)

x=data1.iloc[:,2:32].values

y=data1.iloc[:,0:1].values

print(x.shape)

print(y.shape)

data1.isnull().sum()#checking null value

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

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2,random\_state=1)

print(len(x\_train))

print(len(x\_test))

#feature scaling

from sklearn.preprocessing import StandardScaler

sc=StandardScaler()

x\_train=sc.fit\_transform(x\_train)

x\_test=sc.transform(x\_test)

print(x\_train)

print(x\_test)

#fit the svm model to traing dataset

from sklearn.svm import SVC

svm\_model=SVC(kernel='rbf')

svm\_model.fit(x\_train,y\_train)

#prediction on test dataset

y\_pred=svm\_model.predict(x\_test)

print(y\_pred)

#confusion matrix

from sklearn.metrics import confusion\_matrix

cm=confusion\_matrix(y\_test,y\_pred)

print(cm)

#checking accuracy of the model

from sklearn.metrics import accuracy\_score

acc=accuracy\_score(y\_test,y\_pred)

print(acc)