D**ATA SCIENCE & MACHINE LEARNING**

**LAB CYCLE 4**

**1. Using the iris data set implement the KNN algorithm. Take different values for Test and training data set .Also use different values for k. Also find the accuracy level.**

**Program**

**import** numpy **as** np

**import** matplotlib.pyplot **as** plt

**import** pandas **as** pd

dataset **=** pd**.**read\_csv("iris.csv")

*# print(dataset)*

*# dataset.info()*

x **=** dataset**.**iloc[:, :**-**1]**.**values

y **=** dataset**.**iloc[:,4]**.**values

*# 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**=**0.20)

*# print(x\_test)*

*# from sklearn.preprocessing import StandardScaler*

*# scaler = StandardScaler()*

*# scaler.fit(x\_train)*

*# x\_train = scaler.transform(x\_train)*

*# x\_test = scaler.transform(x\_test)*

*# print(x\_train)*

**from** sklearn.neighbors **import** KNeighborsClassifier

classifier **=** KNeighborsClassifier(n\_neighbors**=**5)

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

KNeighborsClassifier()

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

**from** sklearn.metrics **import** accuracy\_score

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

print("Accuracy:",accuracy)

**for** (i,j) **in** zip(y\_pred,y\_test):

**if** i**!=**j:

print("Actual value:",i,"Predicted value:",j)

print("Number of mislabeled points from test data set:",(y\_test **!=** y\_pred)**.**sum())

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

**from** sklearn.metrics **import** classification\_report, confusion\_matrix

print(confusion\_matrix(y\_test, y\_pred))

print(classification\_report(y\_test, y\_pred))

**Output**

Accuracy: 0.9333333333333333

Actual value: Versicolor Predicted value: Virginica

Actual value: Versicolor Predicted value: Virginica

Number of mislabeled points from test data set: 2

[[ 8 0 0]

[ 0 13 0]

[ 0 2 7]]

precision recall f1-score support

Setosa 1.00 1.00 1.00 8

Versicolor 0.87 1.00 0.93 13

Virginica 1.00 0.78 0.88 9

accuracy 0.93 30

macro avg 0.96 0.93 0.93 30

weighted avg 0.94 0.93 0.93 30

**2. *Download another data set suitable for the KNN and implement the KNN algorithm. Take different values for Test and training data set .Also use different values for k.***

**Program**

**import** numpy **as** np

**import** matplotlib.pyplot **as** plt

**import** pandas **as** pd

dataset **=** pd**.**read\_csv("cancer.csv")

*# print(dataset)*

*# dataset.info()*

*# dataset.head()*

x **=** dataset**.**iloc[:, 2:35]**.**values

y **=** dataset**.**iloc[:,1]**.**values

*# 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**=**0.20)

**from** sklearn.neighbors **import** KNeighborsClassifier

classifier **=** KNeighborsClassifier(n\_neighbors**=**5)

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

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

**from** sklearn.metrics **import** classification\_report, confusion\_matrix

print(classification\_report(y\_test, y\_pred))

**from** sklearn.metrics **import** accuracy\_score

print ("Accuracy : ", accuracy\_score(y\_test, y\_pred))

df **=** pd**.**DataFrame({'Real Values':y\_test, 'Predicted Values':y\_pred})

**Output**

precision recall f1-score support

N 0.87 0.97 0.92 34

R 0.50 0.17 0.25 6

accuracy 0.85 40

macro avg 0.68 0.57 0.58 40

weighted avg 0.81 0.85 0.82 40

Accuracy : 0.85

**3. Using iris data set, implement naive bayes classification for different naive Bayes classification algorithms.( (i) gaussian (ii) bernoulli etc)**

** Find out the accuracy level w.r.t to each algorithm**

** Display the no:of mislabeled classification from test data set**

** List out the class labels of the mismatching records**

**Program**

**(i) gaussian**

**import** numpy **as** np

**import** matplotlib.pyplot **as** plt

**import** pandas **as** pd

dataset **=** pd**.**read\_csv('iris.csv')

X **=** dataset**.**iloc[:,:4]**.**values

y **=** dataset['variety']**.**values

dataset**.**head(5)

**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)

**from** sklearn.naive\_bayes **import** GaussianNB

classifier **=** GaussianNB()

classifier**.**fit(X\_train, y\_train)

y\_pred **=** classifier**.**predict(X\_test)

print("Gaussian")

**from** sklearn.metrics **import** accuracy\_score

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

print("Accuracy of Gaussion:",accuracy)

**for** (i,j) **in** zip(y\_pred,y\_test):

**if** i**!=**j:

print("Actual value:",i,"Predicted value:",j)

print("Number of mislabeled points from test data set:",(y\_test **!=** y\_pred)**.**sum())

print("")

**Output**

Gaussian

Accuracy of Gaussion: 0.9666666666666667

Actual value: Versicolor Predicted value: Virginica

Number of mislabeled points from test data set: 1

**(ii) bernoulli**

**Program**

print("Bernoulli")

**from** sklearn.naive\_bayes **import** BernoulliNB

classifier2 **=** BernoulliNB()

classifier2**.**fit(X\_train, y\_train)

y\_pred **=** classifier2**.**predict(X\_test)

**from** sklearn.metrics **import** accuracy\_score

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

print("Accuracy of Bernoulli:",accuracy)

**for** (i,j) **in** zip(y\_pred,y\_test):

**if** i**!=**j:

print("Actual value:",i,"Predicted value:",j)

print("Number of mislabeled points from test data set:",(y\_test **!=** y\_pred)**.**sum())

print("")

**Output**

Bernoulli

Accuracy of Bernoulli: 0.23333333333333334

Actual value: Virginica Predicted value: Versicolor

Actual value: Virginica Predicted value: Versicolor

Actual value: Virginica Predicted value: Setosa

Actual value: Virginica Predicted value: Setosa

Actual value: Virginica Predicted value: Setosa

Actual value: Virginica Predicted value: Setosa

Actual value: Virginica Predicted value: Setosa

Actual value: Virginica Predicted value: Versicolor

Actual value: Virginica Predicted value: Setosa

Actual value: Virginica Predicted value: Setosa

Actual value: Virginica Predicted value: Setosa

Actual value: Virginica Predicted value: Setosa

Actual value: Virginica Predicted value: Setosa

Actual value: Virginica Predicted value: Setosa

Actual value: Virginica Predicted value: Versicolor

Actual value: Virginica Predicted value: Setosa

Actual value: Virginica Predicted value: Versicolor

Actual value: Virginica Predicted value: Versicolor

Actual value: Virginica Predicted value: Setosa

Actual value: Virginica Predicted value: Versicolor

Actual value: Virginica Predicted value: Versicolor

Actual value: Virginica Predicted value: Setosa

Actual value: Virginica Predicted value: Versicolor

Number of mislabeled points from test data set: 23

**4. Use car details CSV file and implement decision tree algorithm**

** Find out the accuracy level.**

** Display the no:of mislabeled classification from test data set**

** List out the class labels of the mismatching records**

**Program**

**import** os

**import** numpy **as** np

**import** pandas **as** pd

**import** numpy **as** np**,** pandas **as** pd

**import** matplotlib.pyplot **as** plt

**from** sklearn **import** tree, metrics, model\_selection

data**=**pd**.**read\_csv('car.csv',names**=**['buying','maint','doors','persons','lug\_boot','safety','class'])

data**.**head()

data**.**info()

data['class'],class\_names**=**pd**.**factorize(data['class'])

print(class\_names)

print(data['class']**.**unique())

data['buying'],\_ **=** pd**.**factorize(data['buying'])

data['maint'],\_ **=** pd**.**factorize(data['maint'])

data['doors'],\_ **=** pd**.**factorize(data['doors'])

data['persons'],\_ **=** pd**.**factorize(data['persons'])

data['lug\_boot'],\_ **=** pd**.**factorize(data['lug\_boot'])

data['safety'],\_ **=** pd**.**factorize(data['safety'])

data**.**head()

data**.**info()

X **=** data**.**iloc[:,:**-**1]

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

*# split data randomly into 70% training and 30% test*

X\_train, X\_test, y\_train, y\_test **=** model\_selection**.**train\_test\_split(X, y, test\_size**=**0.3, random\_state**=**0)

# train the decision tree

dtree **=** tree**.**DecisionTreeClassifier(criterion**=**'entropy', max\_depth**=**3, random\_state**=**0)

dtree**.**fit(X\_train, y\_train)

*# use the model to make predictions with the test data*

y\_pred **=** dtree**.**predict(X\_test)

*# how did our model perform?*

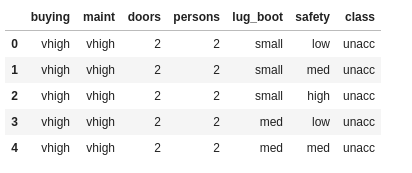
accuracy **=** metrics**.**accuracy\_score(y\_test, y\_pred)

print('Accuracy: {:.2f}'**.**format(accuracy))\

count\_misclassified = (y\_test != y\_pred).sum()

print('Misclassifiedsamples{}'**.**format(count\_misclassified))

**Output**



<class 'pandas.core.frame.DataFrame'>

RangeIndex: 1728 entries, 0 to 1727

Data columns (total 7 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 buying 1728 non-null object

1 maint 1728 non-null object

2 doors 1728 non-null object

3 persons 1728 non-null object

4 lug\_boot 1728 non-null object

5 safety 1728 non-null object

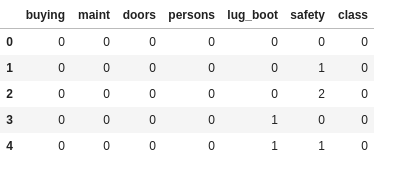
6 class 1728 non-null object

dtypes: object(7)

memory usage: 94.6+ KB

Index(['unacc', 'acc', 'vgood','good'],dtype='object')

[0 1 2 3]



<class 'pandas.core.frame.DataFrame'>

RangeIndex: 1728 entries, 0 to 1727

Data columns (total 7 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 buying 1728 non-null int64

1 maint 1728 non-null int64

2 doors 1728 non-null int64

3 persons 1728 non-null int64

4 lug\_boot 1728 non-null int64

5 safety 1728 non-null int64

6 class 1728 non-null int64

dtypes: int64(7)

memory usage: 94.6 KB

DecisionTreeClassifier(criterion='entropy', max\_depth=3, random\_state=0)

Accuracy: 0.82

Misclassified samples: 96

**5. Implement Simple and multiple linear regression for the data sets ‘student\_score.csv’ and ‘company\_data .csv’ respectively**

**Program**

**import** numpy **as** np

**import** pandas **as** pd

**import** matplotlib.pyplot **as** plt

*#data set contains details of no.of hours spend by students for studt and their marks*

student **=** pd**.**read\_csv('student\_scores.csv')

student**.**head()

student**.**describe()

student**.**info()

**import** matplotlib.pyplot **as** plt

Xax**=**student**.**iloc[:,0]

Yax**=**student**.**iloc[:,1]

plt**.**scatter(Xax,Yax)

plt**.**xlabel("No.of hours")

plt**.**ylabel("Score")

plt**.**title("Student scores")

plt**.**show()

*#Perform the simple linear regression model*

*#Equation: Y=w0+w1.x*

*#Here Y(marks)=w0+w1.x*

*#Create x as hours and Y as marks*

X **=** student**.**iloc[:, :**-**1]

y **=** student**.**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)

print(X\_train)

**from** sklearn.linear\_model **import** LinearRegression

regressor **=** LinearRegression()

regressor**.**fit(X\_train, y\_train)

print(regressor**.**intercept\_)

print(regressor**.**coef\_)

y\_pred **=** regressor**.**predict(X\_test)

**for**(i,j) **in** zip(y\_test,y\_pred):

**if** i**!=**j:

print("Actual value :",i,"Predicted value :",j)

print("Number of mislabeled points from test data set :", (y\_test **!=** y\_pred)**.**sum())

**from** sklearn **import** metrics

print("Mean Absolute error :", metrics**.**mean\_absolute\_error(y\_test,y\_pred))

print("Mean Squared error :", metrics**.**mean\_squared\_error(y\_test,y\_pred))

print("Root Mean Squared error :", np**.**sqrt(metrics**.**mean\_squared\_error(y\_test,y\_pred)))

**import** matplotlib.pyplot **as** plt

c**=**X\_test['Hours']**.**count()

xax**=**np**.**arange(c)

print(xax)

X\_axis **=** np**.**arange(len(xax))

plt**.**bar(X\_axis**-**0.2, y\_test, 0.6, label**=**'Actual')

plt**.**bar(X\_axis**+**0.2, y\_pred, 0.6, label**=**'Predicted')

plt**.**xlabel("Test Records")

plt**.**ylabel("Marks")

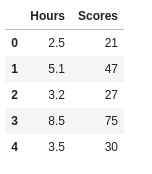
plt**.**title("Student Score prediction")

plt**.**legend()

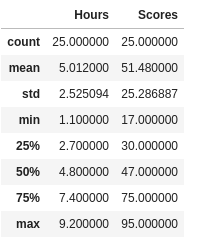
plt**.**show()

**Output**

**student.head()**



**student.describe()**

****

**student.info()**

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 25 entries, 0 to 24

Data columns (total 2 columns):

# Column Non-Null Count Dtype

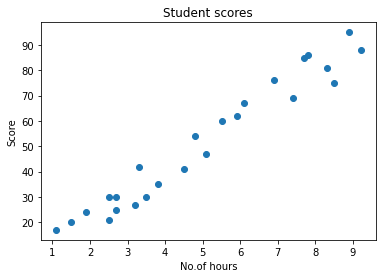
--- ------ -------------- -----

0 Hours 25 non-null float64

1 Scores 25 non-null int64

dtypes: float64(1), int64(1)

memory usage: 528.0 bytes



**print(X)**

Hours

0 2.5

1 5.1

2 3.2

3 8.5

4 3.5

5 1.5

6 9.2

7 5.5

8 8.3

9 2.7

10 7.7

11 5.9

12 4.5

13 3.3

14 1.1

15 8.9

16 2.5

17 1.9

18 6.1

19 7.4

20 2.7

21 4.8

22 3.8

23 6.9

24 7.8

**print(y)**

0 21

1 47

2 27

3 75

4 30

5 20

6 88

7 60

8 81

9 25

10 85

11 62

12 41

13 42

14 17

15 95

16 30

17 24

18 67

19 69

20 30

21 54

22 35

23 76

24 86

Name: Scores, dtype: int64

**print(X\_train)**

Hours

13 3.3

11 5.9

18 6.1

22 3.8

8 8.3

16 2.5

24 7.8

3 8.5

12 4.5

14 1.1

0 2.5

10 7.7

15 8.9

4 3.5

7 5.5

5 1.5

2 3.2

19 7.4

17 1.9

23 6.9

**print(regressor.intercept\_)**

2.339955743698276

**print(regressor.coef\_)**

[9.88294529]

Actual value : 47 Predicted value : 52.74297671733691

Actual value : 88 Predicted value : 93.2630524020268

Actual value : 54 Predicted value : 49.77809313065229

Actual value : 30 Predicted value : 29.02390802385991

Actual value : 25 Predicted value : 29.02390802385991

Number of mislabeled points from test data set : 5

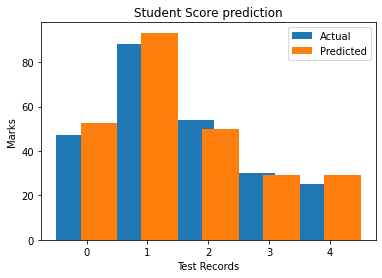
Mean Absolute error : 4.045587197742284

Mean Squared error : 19.130118221233708

Root Mean Squared error : 4.373799060454619

**print(xax)**

[0 1 2 3 4]



**multiple linear regression for the data sets ‘student\_score.csv’ and ‘company\_data .csv’ respectively**

**6.**

**Program**

**Output**