**Q6. Take a regression dataset from Kaggle and implement linear regression.**

**AIM**

To implement linear regression model on housing data from Kaggle.

**SOFTWARE USED**

Jupyter Platform - Python Programming Language

**PROGRAM CODE**

|  |
| --- |
| *import numpy as np*  *import pandas as pd*  *import matplotlib.pyplot as plt*  *import seaborn as sns*  *import mpl\_toolkits*  *%matplotlib inline*  *data = pd.read\_csv("kc\_house\_data.csv")*  *data.head()*  *data.describe()*  *train1 = data.drop(['id', 'price'],axis=1)*  *train1.head()*  *from sklearn.linear\_model import LinearRegression*  *reg = LinearRegression()*  *labels = data['price']*  *conv\_dates = [1 if values == 2014 else 0 for values in data.date ]*  *data['date'] = conv\_dates*  *train1 = data.drop(['id', 'price'],axis=1)*  *from sklearn.model\_selection import train\_test\_split*  *x\_train , x\_test , y\_train , y\_test = train\_test\_split(train1 , labels , test\_size = 0.10,random\_state =2)*  *reg.fit(x\_train,y\_train)*  *reg.score(x\_test,y\_test)* |

**OUTPUT**

Graphical user interface, text

Description automatically generated with medium confidence

Graphical user interface, application, table, Excel

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

**DISCUSSION and CONCLUSION**

The linear regression model has been applied and executed successfully on housing data.

**Q7. Take a classification dataset from Kaggle and classify the data into output classes.**

**Also evaluate the classifier efficiency using various evaluation measures.**

**AIM**

Implement a classification problem on a Kaggle dataset using logistic regression.

**SOFTWARE USED**

Jupyter Platform - Python Programming Language

**PROGRAM CODE**

|  |
| --- |
| *import numpy as np # linear algebra*  *import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)*  *import matplotlib.pyplot as plt # data visualization*  *import seaborn as sns # statistical data visualization*  *%matplotlib inline*  *import os*  *for dirname, \_, filenames in os.walk('/kaggle/input'):*  *for filename in filenames:*  *print(os.path.join(dirname, filename))*  *data = '/kaggle/input/weather-dataset-rattle-package/weatherAUS.csv'*  *df = pd.read\_csv(data)*  *# preview the dataset*  *df.head()*  *col\_names = df.columns*  *col\_names*  *X = df.drop(['RainTomorrow'], axis=1)*  *y = df['RainTomorrow']*  *# split X and y into training and testing sets*  *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 = 0)*  *# check the shape of X\_train and X\_test*  *X\_train.shape, X\_test.shape*  *from sklearn.preprocessing import MinMaxScaler*  *scaler = MinMaxScaler()*  *X\_train = scaler.fit\_transform(X\_train)*  *X\_test = scaler.transform(X\_test)*  *X\_train = pd.DataFrame(X\_train, columns=[cols])*  *X\_test = pd.DataFrame(X\_test, columns=[cols])*  *X\_train.describe()*  *# train a logistic regression model on the training set*  *from sklearn.linear\_model import LogisticRegression*  *# instantiate the model*  *logreg = LogisticRegression(solver='liblinear', random\_state=0)*  *# fit the model*  *logreg.fit(X\_train, y\_train)*  *y\_pred\_test = logreg.predict(X\_test)*  *y\_pred\_test*  *# probability of getting output as 0 - no rain*  *logreg.predict\_proba(X\_test)[:,0]*  *# probability of getting output as 1 - rain*  *logreg.predict\_proba(X\_test)[:,1]*  *from sklearn.metrics import accuracy\_score*  *print('Model accuracy score: {0:0.4f}'. format(accuracy\_score(y\_test, y\_pred\_test)))*  *y\_pred\_train = logreg.predict(X\_train)*  *y\_pred\_train*  *print('Training-set accuracy score: {0:0.4f}'. format(accuracy\_score(y\_train, y\_pred\_train)))*  *# Print the Confusion Matrix and slice it into four pieces*  *from sklearn.metrics import confusion\_matrix*  *cm = confusion\_matrix(y\_test, y\_pred\_test)*  *print('Confusion matrix\n\n', cm)*  *print('\nTrue Positives(TP) = ', cm[0,0])*  *print('\nTrue Negatives(TN) = ', cm[1,1])*  *print('\nFalse Positives(FP) = ', cm[0,1])*  *print('\nFalse Negatives(FN) = ', cm[1,0])*  *from sklearn.metrics import classification\_report*  *print(classification\_report(y\_test, y\_pred\_test))*  *# print classification accuracy*  *classification\_accuracy = (TP + TN) / float(TP + TN + FP + FN)*  *print('Classification accuracy : {0:0.4f}'.format(classification\_accuracy))*  *# print classification error*  *classification\_error = (FP + FN) / float(TP + TN + FP + FN)*  *print('Classification error : {0:0.4f}'.format(classification\_error))*  *# print precision score*  *precision = TP / float(TP + FP)*  *print('Precision : {0:0.4f}'.format(precision))*  *recall = TP / float(TP + FN)*  *print('Recall or Sensitivity : {0:0.4f}'.format(recall))*  *true\_positive\_rate = TP / float(TP + FN)*  *print('True Positive Rate : {0:0.4f}'.format(true\_positive\_rate))*  *false\_positive\_rate = FP / float(FP + TN)*  *print('False Positive Rate : {0:0.4f}'.format(false\_positive\_rate))*  *specificity = TN / (TN + FP)*  *print('Specificity : {0:0.4f}'.format(specificity))* |

**OUTPUT**

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text

Description automatically generated with medium confidence

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

**DISCUSSION and CONCLUSION**

The logistic regression model has been applied and executed successfully on the classification problem over the weather dataset of Australia from Kaggle.