|  |  |  |  |
| --- | --- | --- | --- |
| A picture containing drawing, stop, room  Description automatically generated | Research in Computing  Practical # 1 | | |
|  |  |  |  |
|  |  |  |  |
| **Name** | Ninad Karlekar | **Roll Number** | 22306A1012 |
| **Subject/Course:** | Research in Computing Practical | **Class** | M.Sc. IT – Sem I |
| **Topic** | descriptive statistics of data | **Batch** | 1 |
| 1. **Write a program for obtaining descriptive statistics of data.** | | | |
| #Practical 1A: Write a python program on descriptive statistics analysis.  ################################################################  import pandas as pd  #Create a Dictionary of series  d = {'Age':pd.Series([25,26,25,23,30,29,23,34,40,30,51,46]),  'Rating':pd.Series([4.23,3.24,3.98,2.56,3.20,4.6,3.8,3.78,2.98,4.80,4.10,3.65])}  #Create a DataFrame  df = pd.DataFrame(d)  print(df)  print('############ Sum ########## ')  print (df.sum())  print('############ Mean ########## ')  print (df.mean())  print('############ Standard Deviation ########## ')  print (df.std())  print('\nNinad Karlekar 22306A1012') | | | |
|  | | | |
| **Steps (EXCEL):**   1. Open Excel file 2. Go to File -> Options -> Add-ins -> Click on Analysis Toolpack -> click on Go 3. Tick mark Analysis toolpack -> click on OK -> The Data Analysis option will be added in Data tab 4. Now click on Data analysis -> Descriptive Statistiscs -> click on OK 5. Click on input range -> select **Age** column in input column -> and select any blank coloumn in output range -> Tick Mark on **Summery statistics, confidence level for mean(95%), kth Largest(1), kth smallest(1)**. -> click on Ok   **Output:** | | | |
| 1. **Import data from different data sources (from Excel, csv, mysql, sql server, oracle to R/Python/Excel)** | | | |
| **From csv** | | | |
| **Code:**  import sqlite3 as sq  import pandas as pd  Base='C:/VKHCG'  sDatabaseName=Base + '/01-Vermeulen/00-RawData/SQLite/vermeulen.db'  conn = sq.connect(sDatabaseName)  sFileName='C:/VKHCG/01-Vermeulen/01-Retrieve/01-EDS/02-Python/Retrieve\_IP\_DATA.csv'  print('Loading :',sFileName)  IP\_DATA\_ALL\_FIX=pd.read\_csv(sFileName,header=0,low\_memory=False)  IP\_DATA\_ALL\_FIX.index.names = ['RowIDCSV']  sTable='IP\_DATA\_ALL'  print('Storing :',sDatabaseName,' Table:',sTable)  IP\_DATA\_ALL\_FIX.to\_sql(sTable, conn, if\_exists="replace")  print('Loading :',sDatabaseName,' Table:',sTable)  TestData=pd.read\_sql\_query("select \* from IP\_DATA\_ALL;", conn)  print('## Data Values')  print(TestData)  print('################')  print('## Data Profile')  print('################')  print('Rows :',TestData.shape[0])  print('Columns :',TestData.shape[1])  print("Ninad Karlekar 22306A1012")  print('### Done!! ############################################') | | | |
| **Output:** | | | |
| **From Excel** | | | |
| import os  import pandas as pd  Base='F:/tmp/practical-data-science/VKHCG'  sFileDir=Base + '/01-Vermeulen/01-Retrieve/01-EDS/02-Python'  CurrencyRawData = pd.read\_excel('F:/tmp/practical-data-science/VKHCG/01-Vermeulen/00-RawData/Country\_Currency.xlsx')  sColumns = ['Country or territory', 'Currency', 'ISO-4217']  CurrencyData = CurrencyRawData[sColumns]  CurrencyData.rename(columns={'Country or territory': 'Country', 'ISO-4217':  'CurrencyCode'}, inplace=True)  CurrencyData.dropna(subset=['Currency'],inplace=True)  CurrencyData['Country'] = CurrencyData['Country'].map(lambda x: x.strip())  CurrencyData['Currency'] = CurrencyData['Currency'].map(lambda x:  x.strip())  CurrencyData['CurrencyCode'] = CurrencyData['CurrencyCode'].map(lambda x:  x.strip())  print(CurrencyData)  print('~~~~~~ Data from Excel Sheet Retrived Successfully ~~~~~~~ ')  print("\nNinad Karlekar 22306A1012") | | | |
|  | | | |

|  |  |
| --- | --- |
| A picture containing drawing, stop, room  Description automatically generated | Research in Computing  Practical # 2 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
| **Name** | Ninad Karlekar | **Roll Number** | 22306A1012 |
| **Subject/Course:** | Research in Computing Practical | **Class** | M.Sc. IT – Sem I |
| **Topic** | Case study | **Batch** | 1 |
| 1. **Design a survey form for a given case study, collect the primary data and analyse it** | | | |
| <https://forms.gle/UBZHCYfUpAvv2hfRA> | | | |
|  | | | |
| 1. **Perform analysis of given secondary data.** | | | |
| **Steps:**   1. Open World\_population 2010 excel file. 2. Find the sum of Male ani Female Column. 3. Create and find total of Male and Female coloumn (=B4+C4) 4. Find Sum of all Total column values. 5. Find Percentage of Male (= -1\*100\*male column value/ sum of all total values) (=-1\*100\*B4/$D$22) 6. Find Percentage of Male (= 100\*Female column value/ sum of all total values) (=100\*C4/$D$22) 7. Find sum of both male% and female% 8. Select Male% and Female% -> insert -> clustered Bar 9. Put the tip of your mouse arrow on the Y-axis (vertical axis) so it says “Category Axis”, right click and chose Format Axis 10. Choose Axis options tab and set the major and minor tick mark type to None, Axis labels to Low, and click OK 11. Click on any of the bars in your pyramid, click right and select “format data series”. Set the **Overlap to 100** and **Gap Width to 0**. Click OK. | | | |
| OUTPUT: | | | |

|  |  |
| --- | --- |
| A picture containing drawing, stop, room  Description automatically generated | Research in Computing  Practical # 3 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
| **Name** | Ninad Karlekar | **Roll Number** | 22306A1012 |
| **Subject/Course:** | Research in Computing Practical | **Class** | M.Sc. IT – Sem I |
| **Topic** |  | **Batch** | 1 |
| 1. **Perform testing of hypothesis using one sample t-test.** | | | |
| from scipy.stats import ttest\_1samp  import numpy as np  ages = np.genfromtxt('/content/ages.csv')  print(ages)  ages\_mean = np.mean(ages)  print(ages\_mean)  tset, pval = ttest\_1samp(ages, 30)  print('p-values - ',pval)  if pval< 0.05: # alpha value is 0.05  print(" we are rejecting null hypothesis")  else:  print("we are accepting null hypothesis")  print("\nNinad Karlekar 22306A1012") | | | |
|  | | | |
| 1. **Write a program for t-test comparing two means for independent samples.** | | | |
| **Steps(Excel):-**   1. Open Excel file 2. Find the average(mean) of both Experimental and comparison columns 3. Find the Standard deviation of both Experimental and comparison columns 4. Go to Data analysis -> Select t-test: Paired Two Sample for Means -> OK 5. For Variable 1 range(Experimental)= A1 to A21 For Variable 2 range(Comparison)= B1 to B21 For Output Range= D5 to F17 6. Write 2 Hypothesis H0 - Difference in gain score is not likely the result of experiment. H1 - Difference in gain score is likely the result of experimental treatment and not the result of change variation 7. To calculate the T-Test square value go to cell E20 and type   =(A22-B22)/SQRT((A23\*A23)/COUNT(A2:A21)+(B23\*B23)/COUNT(A2:A21)) Formula=(Mean A-Mean B)/SQRT((STDEV A\*STDEV B)/COUNT(of A) + (STDEV\*STDEV)/COUNT(of A))   1. Now go to cell E21 and type =IF(E20<E12,"H0 is Accepted", "H0 is Rejected and H1 is Accepted") | | | |
| **OUTPUT:** | | | |
| **PYTHON:CODE:**  import numpy as np  from scipy import stats  from numpy.random import randn  N = 20  a = 5 \* randn(100) + 50  b = 5 \* randn(100) + 51  var\_a = a.var(ddof=1)  var\_b = b.var(ddof=1)  s = np.sqrt((var\_a + var\_b)/2)  t = (a.mean() - b.mean())/(s\*np.sqrt(2/N))  df = 2\*N - 2  #p-value after comparison with the t  p = 1 - stats.t.cdf(t,df=df)  print("t = " + str(t))  print("p = " + str(2\*p))  if t> p :  print('Mean of two distribution are differnt and significant')  else:  print('Mean of two distribution are same and not significant')  print('\nNinad Karlekar 22306A1012') | | | |
|  | | | |
| 1. **Perform testing of hypothesis using paired t-test.** | | | |
| The paired sample t-test is also called dependent sample t-test. It’s an univariate test that tests for a significant difference between 2 related variables. An example of this is if you where to collect the blood pressure for an individual before and after some treatment, condition, or time point. The data set contains blood pressure readings before and after an intervention. These are variables “bp\_before” and “bp\_after”.  The hypothesis being test is:  • H0 - The mean difference between sample 1 and sample 2 is equal to 0.  • H0 - The mean difference between sample 1 and sample 2 is not equal to 0.  **Code & Output:**  from scipy import stats  import matplotlib.pyplot as plt  import pandas as pd  df = pd.read\_csv("blood\_pressure.csv")  print(df[['bp\_before','bp\_after']].describe())  #First let’s check for any significant outliers in  #each of the variables.  df[['bp\_before', 'bp\_after']].plot(kind='box')  # This saves the plot as a png file  plt.savefig('boxplot\_outliers.png')  #################\*\*#############################  # make a histogram to differences between the two scores.  df['bp\_difference'] = df['bp\_before'] - df['bp\_after']  df['bp\_difference'].plot(kind='hist', title= 'Blood Pressure Difference Histogram')  #Again, this saves the plot as a png file  #################\*\*#############################  plt.savefig('blood pressure difference histogram.png')  stats.probplot(df['bp\_difference'], plot= plt)  plt.title('Blood pressure Difference Q-Q Plot')  plt.savefig('blood pressure difference qq plot.png')  stats.shapiro(df['bp\_difference'])  stats.ttest\_rel(df['bp\_before'], df['bp\_after']) | | | |
| OUTPUT: | | | |

|  |  |
| --- | --- |
| A picture containing drawing, stop, room  Description automatically generated | Research in Computing  Practical # 4 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
| **Name** | Ninad Karlekar | **Roll Number** | 22306A1012 |
| **Subject/Course:** | Research in Computing Practical | **Class** | M.Sc. IT – Sem I |
| **Topic** | chi-squared | **Batch** | 1 |
| 1. **Perform testing of hypothesis using chi-squared goodness-of-fit test.** | | | |
| STEPS(EXCEL):-   1. Find total of both columns 2. Formula 3. Find the sum of all | | | |
| OUTPUT: | | | |
| 1. **Perform testing of hypothesis using chi-squared test of independence.** | | | |
| **Steps:**   1. Find the total for all columns and rows 2. To calculate the expected value Ei Go to Cell N9 and type =N8/2 Go to Cell O9 and type =O8/2 Go to Cell P9 and type =P8/2 Go to Cell Q9 and type =Q8/2 Go to Cell R9 and type =R8/2 3. Now Calculate Go to cell **T6** and type   =SUM((N6-$N$9)^2/$N$9,(O6-$O$9)^2/$O$9,(P6-$P$9)^2/$P$9,(Q6-Q$9)^2/$Q$9, (R6-$R$9)^2/$R$9)  Go to cell **T7** and type  =SUM((N7-$N$9)^2/$N$9,(O7-$O$9)^2/$O$9,(P7-$P$9)^2/$P$9,(Q7-Q$9)^2/$Q$9, (R7-$R$9)^2/$R$9)  To get the table value go to cell T11 and type =CHIINV(0.05,4)  Go to cell O13 and type =IF(T8>=T11," H0 is Accepted", "H0 is Rejected") | | | |
|  | | | |
| **Using Python** | | | |
| Code:  import numpy as np  import pandas as pd  import scipy.stats as stats  np.random.seed(10)  stud\_grade = np.random.choice(a=["O","A","B","C","D"],  p=[0.20, 0.20 ,0.20, 0.20, 0.20], size=100)  stud\_gen = np.random.choice(a=["Male","Female"], p=[0.5, 0.5], size=100)  mscpart1 = pd.DataFrame({"Grades":stud\_grade, "Gender":stud\_gen})  print(mscpart1)  stud\_tab = pd.crosstab(mscpart1.Grades, mscpart1.Gender, margins=True)  stud\_tab.columns = ["Male", "Female", "row\_totals"]  stud\_tab.index = ["O", "A", "B", "C", "D", "col\_totals"]  observed = stud\_tab.iloc[0:5, 0:2 ]  print(observed)  expected = np.outer(stud\_tab["row\_totals"][0:5], stud\_tab.loc["col\_totals"][0:2]) / 100  print(expected)  chi\_squared\_stat = (((observed-expected)\*\*2)/expected).sum().sum()  print('Calculated : ',chi\_squared\_stat)  crit = stats.chi2.ppf(q=0.95, df=4)  print('Table Value : ',crit)  if chi\_squared\_stat>= crit:  print('H0 is Accepted ')  else:  print('H0 is Rejected ')  print("\nNinad Karlekar 22306A1012") | | | |
|  | | | |

|  |  |
| --- | --- |
| A picture containing drawing, stop, room  Description automatically generated | Research in Computing  Practical # 5 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | Ninad Karlekar | **Roll Number** | 22306A1012 |
| **Subject/Course:** | Research in Computing Practical | **Class** | M.Sc. IT – Sem I |
| **Topic** | Perform testing of hypothesis using Z-test. | **Batch** | 1 |
| 1. **Perform testing of hypothesis using Z-test.** | | | |
| Description:-  Define Hypothesis:  Hypothesis is a strong, short statement that forms the basis of your research.  The purpose of Hypothesis is to predict the findings, conclusions and data. It is a educated guess based on your observation and Environment around you.  Define Null hypothesis:  Null hypothesis is a type of hypothesis that is presumed to be true until it is invalidated by testing.  What is hypothesis Testing?  Hypothesis testing provides a way to verify whether the results of an experiment are valid. It is a type of tool  What is Z Test?  Z test is a statistical test that is conducted on data that approximately follows a normal distribution. The z test can be performed on one sample, two samples, or on proportions for hypothesis testing. It checks if the means of two large samples are different or not when the population variance is known.  Procedure:-  Use a Z test if:  • Your sample size is greater than 30. Otherwise, use a t test.  • Data points should be independent from each other. In other words, one data point isn’t related or doesn’t affect another data point.  • Your data should be normally distributed. However, for large sample sizes (over 30) this doesn’t always matter.  • Your data should be randomly selected from a population, where each item has an equal chance of being selected.  • Sample sizes should be equal if at all possible.  H0 - Blood pressure has a mean of 156 units  Dataset- blood\_pressure.csv  **Code:-**  from statsmodels.stats import weightstats as stests  import pandas as pd  from scipy import stats  df = pd.read\_csv("blood\_pressure.csv")  df[['bp\_before','bp\_after']].describe()  print(df)  ztest ,pval = stests.ztest(df['bp\_before'], x2=None, value=156)  print(float(pval))  if pval<0.05:  print("reject null hypothesis")  else:  print("accept null hypothesis")  print("\nNinad Karlekar 22306A1012") | | | |
|  | | | |
| 1. **Two-Sample Z test** | | | |
| Code:-  import pandas as pd  from statsmodels.stats import weightstats as stests  df = pd.read\_csv("blood\_pressure.csv")  df[['bp\_before','bp\_after']].describe()  print(df)  ztest ,pval = stests.ztest(df['bp\_before'], x2=df['bp\_after'], value=0,alternative='two-sided')  print(float(pval))  if pval<0.05:  print("reject null hypothesis")  else:  print("accept null hypothesis")  print("\nNinad Karlekar 22306A1012") | | | |
|  | | | |

|  |  |
| --- | --- |
| A picture containing drawing, stop, room  Description automatically generated | Research in Computing  Practical # 6 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
| **Name** | Ninad Karlekar | **Roll Number** | 22306A1012 |
| **Subject/Course:** | Research in Computing Practical | **Class** | M.Sc. IT – Sem I |
| **Topic** | Hypothesis using ANOVA | **Batch** | 1 |
| 1. **Perform testing of hypothesis using one-way ANOVA** | | | |
| Steps(EXCEL):   1. Open scores.csv file 2. Go to Data analysis -> Anova single factor -> ok 3. Select input range as all values from [Average Score (SAT Math), Average Score (SAT Reading), Average Score (SAT Writing)] columns 4. OUTPUT | | | |
| 1. **Perform testing of hypothesis using two-way ANOVA.** | | | |
| Description:  ANOVA (Analysis of Variance) is a statistical test used to analyses the difference between the means of more than two groups.  A two-way ANOVA is used to estimate how the mean of a quantitative variable changes according to the levels of two categorical variables. Use a two-way ANOVA when you want to know how two independent variables, in combination, affect a dependent variable.  Steps   1. Open ToothGrowth.csv file 2. Go to Data analysis -> Anova two factor with replication-> ok 3. Select all cell in input range , Rows per sample=30 Alpha=0.05 | | | |
| **Output:** | | | |
| 1. **Perform testing of hypothesis using multivariate ANOVA (MANOVA)** | | | |
| **Description:**  The Multivariate analysis of variance (MANOVA) procedure provides regression analysis and analysis of variance for multiple dependent variables by one or more factor variables or covariates. The factor variables divide the population into groups. Using this general linear model procedure, you can test null hypotheses about the effects of factor variables on the means of various groupings of a joint distribution of dependent variables. You can investigate interactions between factors as well as the effects of individual factors. In addition, the effects of covariates and covariate interactions with factors can be included. For regression analysis, the independent (predictor) variables are specified as covariates.  **PYTHON CODE:**  import pandas as pd  from statsmodels.multivariate.manova import MANOVA  df = pd.read\_csv('Iris.csv', index\_col=0)  df.columns = df.columns.str.replace(".", "\_")  df.head()  print('~~~~~~~~ Data Set ~~~~~~~~')  print(df)  maov = MANOVA.from\_formula('SepalLengthCm + SepalWidthCm + \  PetalLengthCm + PetalWidthCm ~ Species', data=df)  print('~~~~~~~~ MANOVA Test Result ~~~~~~~~')  print(maov.mv\_test()) | | | |
| **OUTPUT:** | | | |

|  |  |
| --- | --- |
| A picture containing drawing, stop, room  Description automatically generated | Research in Computing  Practical # 7 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
| **Name** | Ninad Karlekar | **Roll Number** | 22306A1012 |
| **Subject/Course:** | Research in Computing Practical | **Class** | M.Sc. IT – Sem I |
| **Topic** | Perform the Random sampling | Perform the Stratified sampling | **Batch** | 1 |
| 1. **Perform the Random sampling for the given data and analyse it.** | | | |
| Example 1: From a population of 10 women and 10 men as given in the table in Figure 1 on the left below, create a random sample of 6 people for Group 1 and a periodic sample consisting of every 3rd woman for Group 2.  You need to run the sampling data analysis tool twice, once to create Group 1 and again to create Group 2. For Group 1 you select all 20 population cells as the Input Range and Random as the Sampling Method with 6 for the Random Number of Samples. For Group 2 you select the 10 cells in the Women column as Input Range and Periodic with Period 3. | | | |
| 1. Open existing excel sheet with population data Sample Sheet looks as given below: 2. Set Cell O1 = Male and Cell O2 = Female 3. To generate a random sample for male students from given population go to Cell O1 and type =INDEX(E$2:E$62,RANK(B2,B$2:B$62)) Drag the formula to the desired no of cell to select random sample. 4. Now, to generate a random sample for female students go to cell P1 and type =INDEX(K$2:K$40,RANK(H2,H$2:H$40)) Drag the formula to the desired no of cell to select random sample. | | | |
| **OUTPUT:** | | | |
| 1. **Perform the Stratified sampling for the given data and analyse it.** | | | |
| we are to carry out a hypothetical housing quality survey across Lagos state, Nigeria. And we looking at a total of 5000 houses (hypothetically). We don’t just go to one local government and select 5000 houses, rather we ensure that the 5000 houses are a representative of the whole 20 local government areas Lagos state is comprised of. This is called stratified sampling. The population is divided into homogenous strata and the right number of instances is sampled from each stratum to guarantee that the test-set (which in this case is the 5000 houses) is a representative of the overall population. If we used random sampling, there would be a significant chance of having bias in the survey results. | | | |
| **Program Code:**  import pandas as pd  import numpy as np  import matplotlib  import matplotlib.pyplot as plt  plt.rcParams['axes.labelsize'] = 14  plt.rcParams['xtick.labelsize'] = 12  plt.rcParams['ytick.labelsize'] = 12  import seaborn as sns  color = sns.color\_palette()  sns.set\_style('darkgrid')  import sklearn  from sklearn.model\_selection import train\_test\_split  housing =pd.read\_csv('housing.csv')  print(housing.head())  print(housing.info())  #creating a heatmap of the attributes in the dataset  correlation\_matrix = housing.corr()  plt.subplots(figsize=(8,6))  sns.heatmap(correlation\_matrix, center=0, annot=True, linewidths=.3)  corr =housing.corr()  print(corr['median\_house\_value'].sort\_values(ascending=False))  sns.distplot(housing.median\_income)  plt.show() | | | |
| **output:** | | | |

|  |  |
| --- | --- |
| A picture containing drawing, stop, room  Description automatically generated | Research in Computing  Practical # 8 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
| **Name** | Ninad Karlekar | **Roll Number** | 22306A1012 |
| **Subject/Course:** | Research in Computing Practical | **Class** | M.Sc. IT – Sem I |
| **Topic** | Compute different types of correlation. | **Batch** | 1 |
| **Write a program for computing different correlation** | | | |
| 1. **Positive Correlation:** | | | |
| Positive Correlation:  Let’s take a look at a positive correlation. Numpy implements a corrcoef() function that returns a matrix of correlations of x with x, x with y, y with x and y with y. We’re interested in the values of correlation of x with y (so position (1, 0) or (0, 1)). | | | |
| **Code:**  import numpy as np  import matplotlib.pyplot as plt  np.random.seed(1)  # 1000 random integers between 0 and 50  x = np.random.randint(0, 50, 1000)  # Positive Correlation with some noise  y = x + np.random.normal(0, 10, 1000)  np.corrcoef(x, y)  matplotlib.style.use('ggplot')  plt.scatter(x, y)  plt.show()  print("\nNinad Karlekar 22306A1012") | | | |
| **Output:** | | | |
| 1. **Negative Correlation:** | | | |
| **CODE:**  import numpy as np  import matplotlib.pyplot as plt  np.random.seed(1)  # 1000 random integers between 0 and 50  x = np.random.randint(0, 50, 1000)  # Negative Correlation with some noise  y = 100 - x + np.random.normal(0, 5, 1000)  np.corrcoef(x, y)  plt.scatter(x, y)  plt.show()  print("\nNinad Karlekar 22306A1012")  print("Practical 8-B") | | | |
| **OUTPUT:** | | | |
| 1. **No/Weak Correlation:** | | | |
| **CODE:**  import numpy as np  import matplotlib.pyplot as plt  np.random.seed(1)  x = np.random.randint(0, 50, 1000)  y = np.random.randint(0, 50, 1000)  np.corrcoef(x, y)  plt.scatter(x, y)  plt.show()  print("\nNinad Karlekar 22306A1012")  print("Practical 8-C") | | | |
| **OUTPUT:** | | | |

|  |  |
| --- | --- |
| A picture containing drawing, stop, room  Description automatically generated | Research in Computing  Practical # 9 |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
| **Name** | Ninad Karlekar | **Roll Number** | 22306A1012 |
| **Subject/Course:** | Research in Computing Practical | **Class** | M.Sc. IT – Sem I |
| **Topic** | Linear regression for prediction. | Polynomial regression for prediction. | **Batch** | 1 |
| 1. **Write a program to Perform linear regression for prediction.** | | | |
| **CODE:**  #PRAC 9A #Jupyter  import quandl, math  import numpy as np  import pandas as pd  from sklearn.model\_selection import train\_test\_split  from sklearn import svm  from sklearn.linear\_model import LinearRegression  import matplotlib.pyplot as plt  from matplotlib import style  import datetime  style.use('ggplot')  df = quandl.get("WIKI/GOOGL")  df = df[['Adj. Open', 'Adj. High', 'Adj. Low', 'Adj. Close', 'Adj. Volume']]  df['HL\_PCT'] = (df['Adj. High'] - df['Adj. Low']) / df['Adj. Close'] \* 100.0  df['PCT\_change'] = (df['Adj. Close'] - df['Adj. Open']) / df['Adj. Open'] \* 100.0  df = df[['Adj. Close', 'HL\_PCT', 'PCT\_change', 'Adj. Volume']]  forecast\_col = 'Adj. Close'  df.fillna(value=-99999, inplace=True)  forecast\_out = int(math.ceil(0.01 \* len(df)))  df['label'] = df[forecast\_col].shift(-forecast\_out)  X = np.array(df.drop(['label'], 1))  X = preprocessing.scale(X)  X\_lately = X[-forecast\_out:]  X = X[:-forecast\_out]  df.dropna(inplace=True)  y = np.array(df['label'])  X\_train, X\_test, y\_train, y\_test = sklearn.model\_selection.train\_test\_split(X, y, test\_size=0.2)  clf = LinearRegression(n\_jobs=-1)  clf.fit(X\_train, y\_train)  confidence = clf.score(X\_test, y\_test)  forecast\_set = clf.predict(X\_lately)  df['Forecast'] = np.nan  last\_date = df.iloc[-1].name  last\_unix = last\_date.timestamp()  one\_day = 86400  next\_unix = last\_unix + one\_day  for i in forecast\_set:  next\_date = datetime.datetime.fromtimestamp(next\_unix)  next\_unix += 86400  df.loc[next\_date] = [np.nan for \_ in range(len(df.columns)-1)]+[i]  df['Adj. Close'].plot()  df['Forecast'].plot()  plt.legend(loc=4)  plt.xlabel('Date')  plt.ylabel('Price')  plt.show()  print("\nNinad Karlekar 22306A1012")  print("Practical 9-A") | | | |
| **OUTPUT:** | | | |
| 1. **Perform polynomial regression for prediction.** | | | |
| **CODE:**  import numpy as np  import matplotlib.pyplot as plt  def estimate\_coef(x, y):  # number of observations/points  n = np.size(x)  # mean of x and y vector  m\_x, m\_y = np.mean(x), np.mean(y)  # calculating cross-deviation and deviation about x  SS\_xy = np.sum(y\*x) - n\*m\_y\*m\_x  SS\_xx = np.sum(x\*x) - n\*m\_x\*m\_x  # calculating regression coefficients  b\_1 = SS\_xy / SS\_xx  b\_0 = m\_y - b\_1\*m\_x  return(b\_0, b\_1)  def plot\_regression\_line(x, y, b):  # plotting the actual points as scatter plot  plt.scatter(x, y, color = "m",marker = "o", s = 30)  # predicted response vector  y\_pred = b[0] + b[1]\*x  # plotting the regression line  plt.plot(x, y\_pred, color = "g")  # putting labels  plt.xlabel('x')  plt.ylabel('y')  # function to show plot  plt.show()  def main():  # observations  x = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])  y = np.array([1, 3, 2, 5, 7, 8, 8, 9, 10, 12])  # estimating coefficients  b = estimate\_coef(x, y)  print("Estimated coefficients:\nb\_0 = {} b\_1 = {}".format(b[0], b[1]))  # plotting regression line  plot\_regression\_line(x, y, b)  if \_\_name\_\_ == "\_\_main\_\_":  main()  print("\nNinad Karlekar 22306A1012")  print("Practical 9-B") | | | |
| **OUTPUT**: | | | |