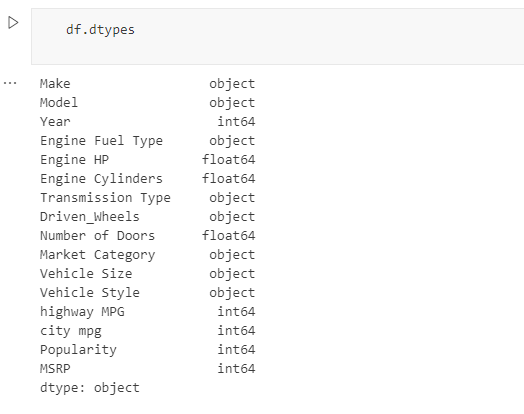


**2. Write a program to perform Exploratory Data Analysis (EDA) for Classification usingPandas and Matplotlib**

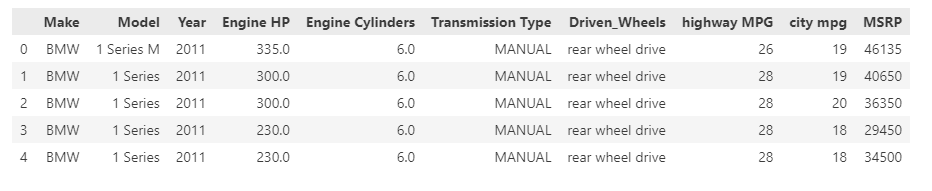
|  |
| --- |
| import pandas as pd  import numpy as np  import seaborn as sns  import matplotlib.pyplot as plt  sns.set(color\_codes=True) |

|  |
| --- |
| df = pd.read\_csv("car\_prices.csv")  df.head() |

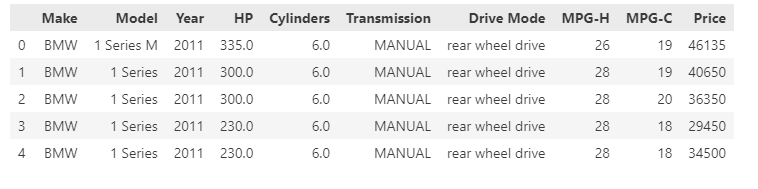
|  |
| --- |
|  |



|  |
| --- |
| df = df.drop(['Engine Fuel Type','Market Category','Vehicle Style','Popularity','Number of Doors', 'Vehicle Size'], axis=1)  df.head(5) |



|  |
| --- |
| df = df.rename(columns={"Engine HP": "HP", "Engine Cylinders": "Cylinders", "Transmission Type": "Transmission", "Driven\_Wheels": "Drive Mode","highway MPG": "MPG-H", "city mpg": "MPG-C", "MSRP": "Price" })  df.head(5) |



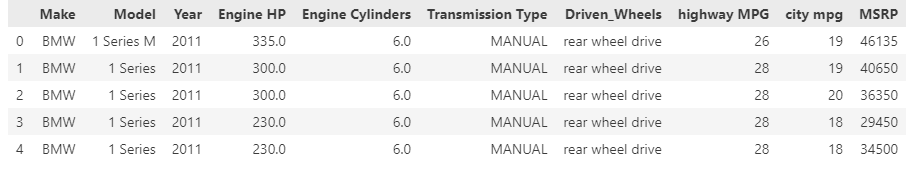
|  |
| --- |
| df.shape |



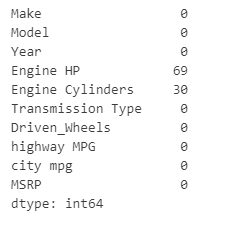
|  |
| --- |
| duplicate\_rows\_df = df [df.duplicated()]  print("Number of duplicate rows: ", duplicate\_rows\_df.shape) |



|  |
| --- |
| df = df.drop\_duplicates()  df.head (5) |



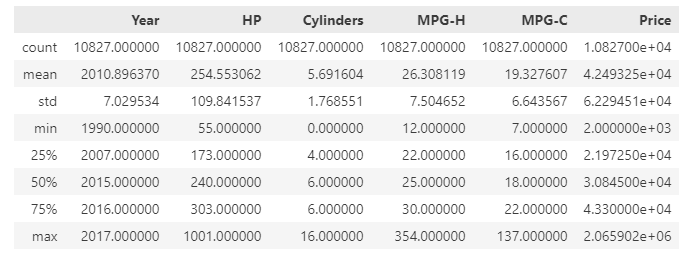
|  |
| --- |
| print(df.isnull().sum()) |



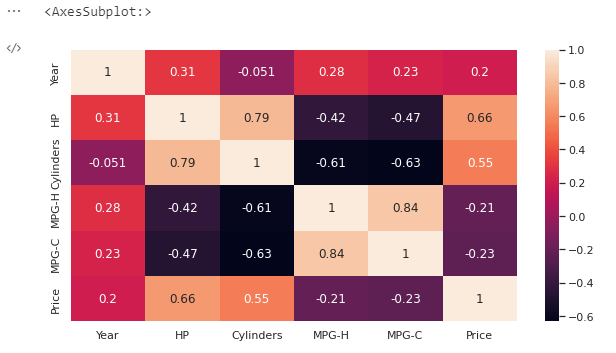
|  |
| --- |
| df=df.dropna()  df.shape |



|  |
| --- |
| df.describe() |

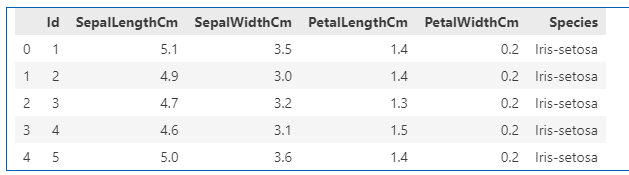


|  |
| --- |
| plt.figure(figsize=(10,5))  sns.heatmap(df.corr(),annot=True) |



**3. Reading data from text files, Excel and the web and exploring various commands for doing descriptive analytics on the Iris data set.**

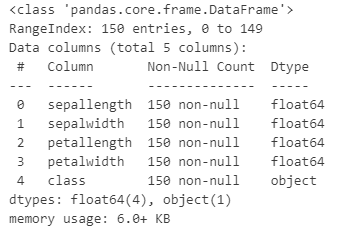
|  |
| --- |
| import pandas as pd  # Reading the CSV file  df = pd.read\_csv("Iris.csv")  # Printing top 5 rows  df.head() |



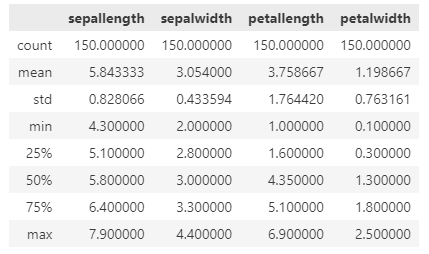
|  |
| --- |
| df.shape |



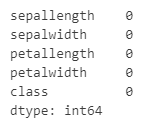
|  |
| --- |
| df.info() |



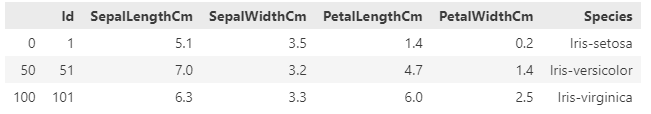
|  |
| --- |
| df.describe() |



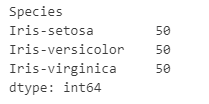
|  |
| --- |
| df.isnull().sum() |



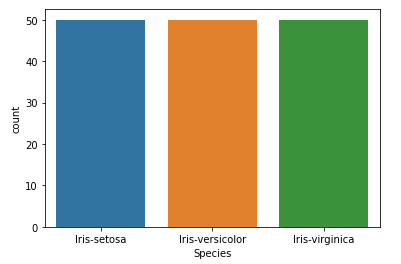
|  |
| --- |
| data = df.drop\_duplicates(subset ="Species",)  data |



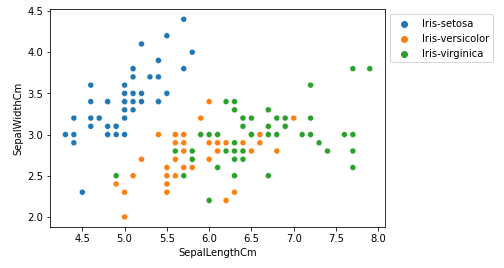
|  |
| --- |
| df.value\_counts("Species") |



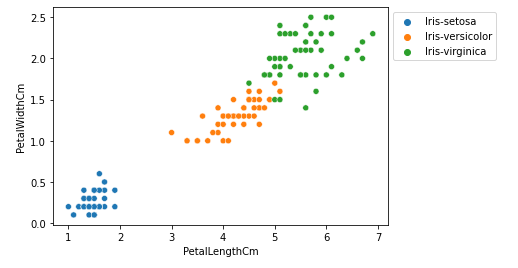
|  |
| --- |
| # importing packages  import seaborn as sns  import matplotlib.pyplot as plt  sns.countplot(x='Species', data=df, )  plt.show() |



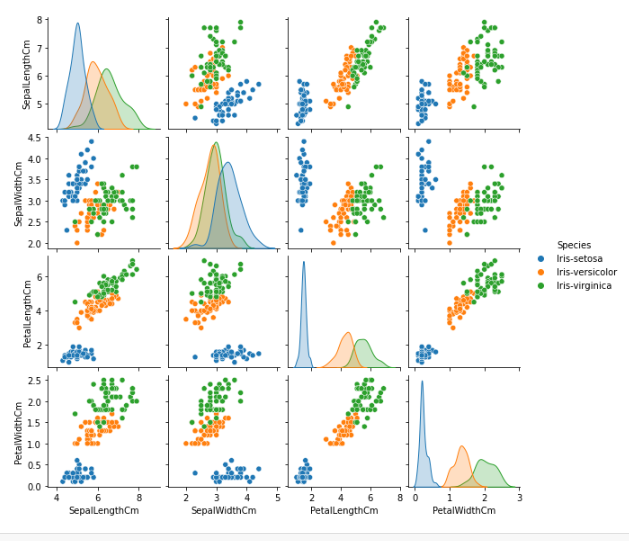
|  |
| --- |
| # importing packages  import seaborn as sns  import matplotlib.pyplot as plt  sns.scatterplot(x='SepalLengthCm', y='SepalWidthCm',  hue='Species', data=df, )  # Placing Legend outside the Figure  plt.legend(bbox\_to\_anchor=(1, 1), loc=2)  plt.show() |



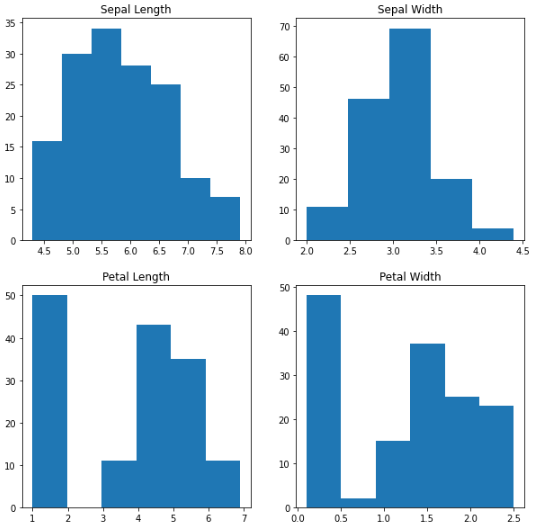
|  |
| --- |
| # importing packages  import seaborn as sns  import matplotlib.pyplot as plt  sns.scatterplot(x='PetalLengthCm', y='PetalWidthCm',  hue='Species', data=df, )  # Placing Legend outside the Figure  plt.legend(bbox\_to\_anchor=(1, 1), loc=2)  plt.show() |



|  |
| --- |
| # importing packages  import seaborn as sns  import matplotlib.pyplot as plt  sns.pairplot(df.drop(['Id'], axis = 1),  hue='Species', height=2) |



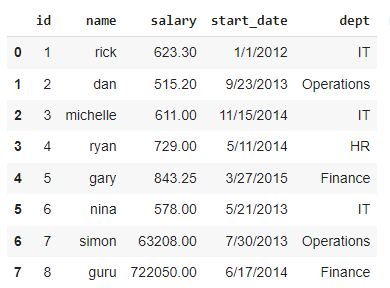
|  |
| --- |
| # importing packages  import seaborn as sns  import matplotlib.pyplot as plt  fig, axes = plt.subplots(2, 2, figsize=(10,10))  axes[0,0].set\_title("Sepal Length")  axes[0,0].hist(df['SepalLengthCm'], bins=7)  axes[0,1].set\_title("Sepal Width")  axes[0,1].hist(df['SepalWidthCm'], bins=5);  axes[1,0].set\_title("Petal Length")  axes[1,0].hist(df['PetalLengthCm'], bins=6);  axes[1,1].set\_title("Petal Width")  axes[1,1].hist(df['PetalWidthCm'], bins=6); |



**4.**

**a. Reading different types of data sets (.txt, .csv) from Web and disk and writing in file in specific disk location.**

|  |
| --- |
| import pandas as pd  data = pd.read\_csv("input.csv")  data |



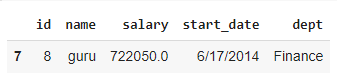
|  |
| --- |
| import pandas as pd  data = pd.read\_csv("input.csv")  print(isinstance(data, pd.DataFrame))  print(data.shape[1])  print(data.shape[0]) |



|  |
| --- |
| import pandas as pd  data = pd.read\_csv("input.csv")  sal = data['salary'].max()  sal |



|  |
| --- |
| import pandas as pd  data = pd.read\_csv("input.csv")  sal = data['salary'].max()  retval = data[data['salary'] == data['salary'].max()]  retval |



|  |
| --- |
| import pandas as pd  data = pd.read\_csv("input.csv")  retval = data[data['dept'] == "IT"]  retval |

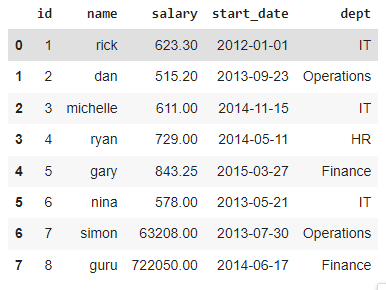


|  |
| --- |
| retval.to\_csv("output.csv", index=False)  newdata = pd.read\_csv("output.csv")  newdata |



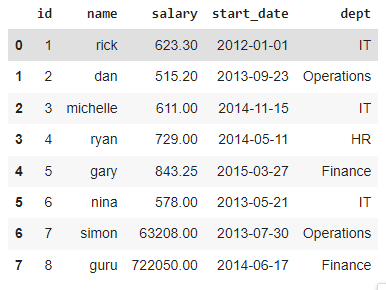
**b. Reading Excel data sheet in Python.**

|  |
| --- |
| !pip install openpyxl  import pandas as pd  data = pd.read\_excel("input.xlsx", sheet\_name=0)  data |



**c. Reading XML dataset in Python(shows error cant do anything but write this one )**

|  |
| --- |
| !pip install openpyxl  import pandas as pd  data = pd.read\_excel("input.XLM", sheet\_name=0)  data |

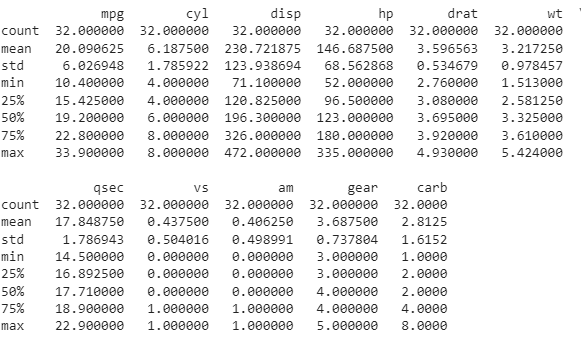


**5. Write a python program to find basic descriptive statistics using summary, str, quartile function on mtcars& cars datasets.**

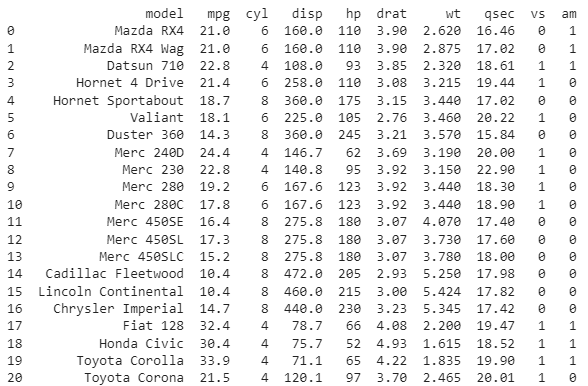
|  |
| --- |
| import pandas as pd  mtcars = pd.read\_csv('mtcars.csv')  mtcars |



|  |
| --- |
| import pandas as pd  mtcars = pd.read\_csv('mtcars.csv')  summary = mtcars.describe()  print(summary) |



|  |
| --- |
| import pandas as pd  mtcars = pd.read\_csv('mtcars.csv')  print(mtcars) |

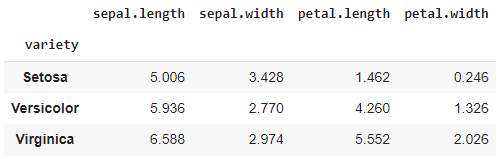


|  |
| --- |
| import numpy as np  np.quantile(mtcars['mpg'], q=0.5) |

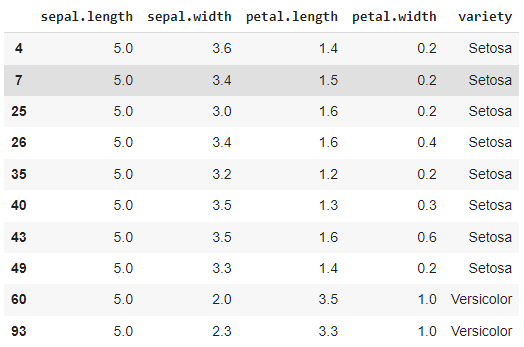


**6. Write an python program to find subset of dataset by using subset (), aggregate () functions on iris dataset**

|  |
| --- |
| import pandas as pd  iris = pd.read\_csv('iris.csv')  # Assuming iris.csv is the dataset file  iris.groupby('variety').mean() |



|  |
| --- |
| import pandas as pd  iris = pd.read\_csv('iris.csv')  # Assuming iris data is stored in a CSV file named 'iris.csv'  subset\_iris = iris[iris['sepal.length'] == 5.0]  subset\_iris |



**7. Use the diabetes,UCI and Pima Indians Diabetes data set for performing the following:**

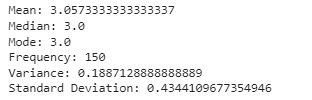
**a. Univariate analysis: Frequency, Mean, Median, Mode, Variance, Standard Deviation, Skewness and Kurtosis**.

|  |
| --- |
| import numpy as np  import statistics as st  from sklearn.datasets import load\_iris |

|  |
| --- |
| X, y = load\_iris(return\_X\_y=True) |

|  |
| --- |
| arr = X[:,1] |

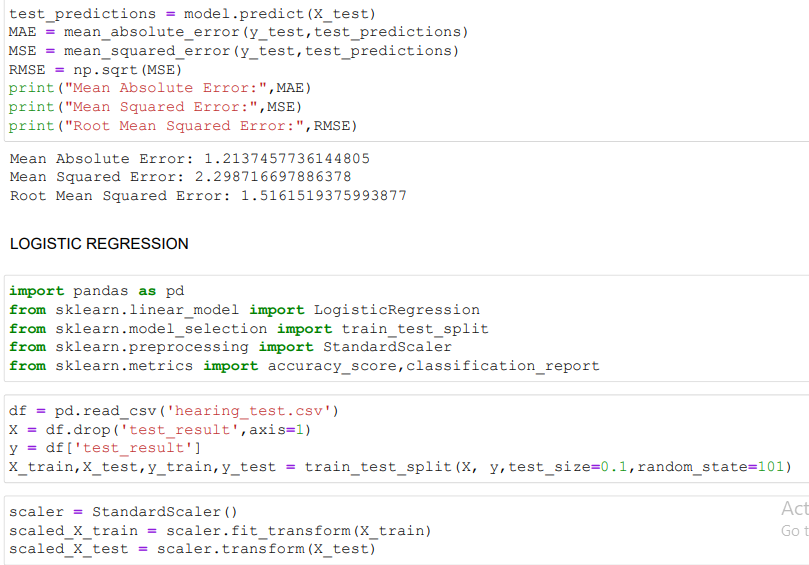
|  |
| --- |
| print('Mean:', np.mean(arr))  print('Median:', np.median(arr))  print('Mode:', st.mode(arr))  print('Frequency:', len(arr))  print('Variance:', np.var(arr))  print('Standard Deviation:', np.std(arr)) |



**b. Bivariate analysis: Linear and logistic regression modeling(same as 10th problem)**

**10. Develop python program to Build linear regression model and logistic regression model, check the model on a test data and predict the numerical quantities.**

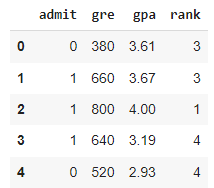






**11. Import data from web storage. Name the dataset and now do Logistic Regression to find out relation between variables that are affecting the admission of a student in a institute based on his or her GRE score, GPA obtained and rank of the student. Also check the model is fit or not. Require (foreign), require (MASS).**

|  |
| --- |
| import pandas as pd  mydata = pd.read\_csv("gredataset.csv")  mydata.head() |



**12. Write a python program to implement Support Vector Machines.Compute the accuracy of  the classifier, considering few test data sets.**

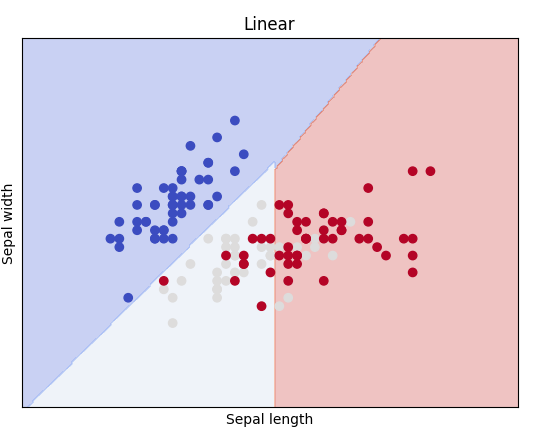
|  |
| --- |
| from sklearn import svm, datasets  import matplotlib.pyplot as plt  import numpy as np  from sklearn.metrics import accuracy\_score  from sklearn.model\_selection import train\_test\_split |

|  |
| --- |
| iris = datasets.load\_iris()  X = iris.data[:, :2] # we only take the first two features  y = iris.target  x\_train, x\_test, y\_train, y\_test = train\_test\_split(X, y, random\_state = 0, test\_size = 0.25)  clf = svm. SVC (kernel='linear', C=1).fit(x\_train, y\_train) |

|  |
| --- |
| classifier\_predictions = clf.predict(x\_test)  print (accuracy\_score (y\_test, classifier\_predictions)\*100) |



|  |
| --- |
| h = 0.02  x\_min, x\_max = X[:, 0].min() - 1, X[:, 0].max() + 1  y\_min, y\_max = X[:, 1].min() - 1, X[:, 1].max() + 1  xx, yy = np.meshgrid(np.arange(x\_min, x\_max, h),                       np.arange(y\_min, y\_max, h))  xx.shape  Z = clf.predict(np.c\_[xx.ravel(), yy.ravel()])  Z = Z.reshape(xx.shape)  plt.contourf(xx, yy, Z, cmap=plt.cm.coolwarm, alpha=0.3)  plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.coolwarm)  plt.xlabel('Sepal length')  plt.ylabel('Sepal width')  plt.xlim(xx.min(), xx.max())  plt.ylim(yy.min(), yy.max())  plt.xticks(())  plt.yticks(())  plt.title("Linear")  plt.show() |

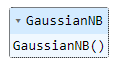


**13. Write a python program to implement the naïve Bayesian classifier for a sample training dataset stored as a .CSV file. Compute the accuracy of the classifier, considering few testdata sets.**

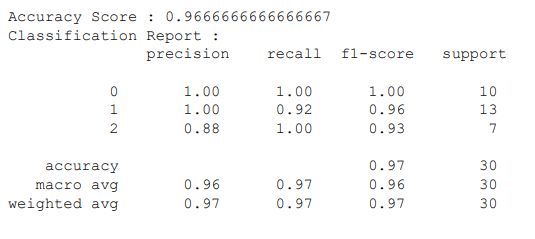
|  |
| --- |
| from sklearn.naive\_bayes import GaussianNB  import pandas as pd  from sklearn.model\_selection import train\_test\_split  from sklearn.metrics import accuracy\_score, classification\_report  from sklearn.datasets import load\_iris |

|  |
| --- |
| X, y = load\_iris(return\_X\_y=True)  X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.2,random\_state=101) |

|  |
| --- |
| model = GaussianNB()  model.fit(X\_train, y\_train) |



|  |
| --- |
| preds = model.predict(X\_test)  print('Accuracy Score :',accuracy\_score(preds,y\_test))  print('Classification Report :\n',classification\_report(preds,y\_test)) |



**14.a. Find the data distributions using box and scatter plot.**

# Import libraries

import matplotlib.pyplot as plt

import numpy as np

# Creating dataset

np.random.seed(10)

data = np.random.normal(100, 20, 200)

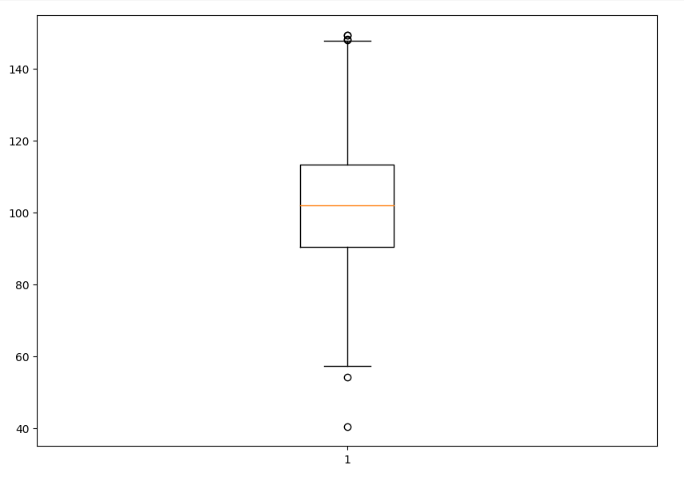
fig = plt.figure(figsize =(10, 7))

# Creating plot

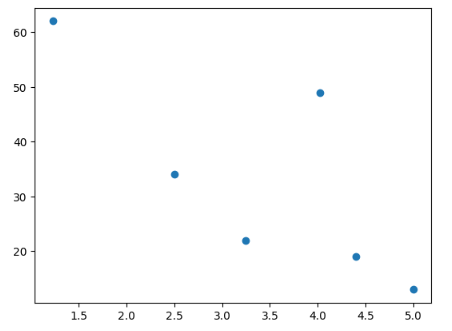
plt.boxplot(data)

# show plot

plt.show()

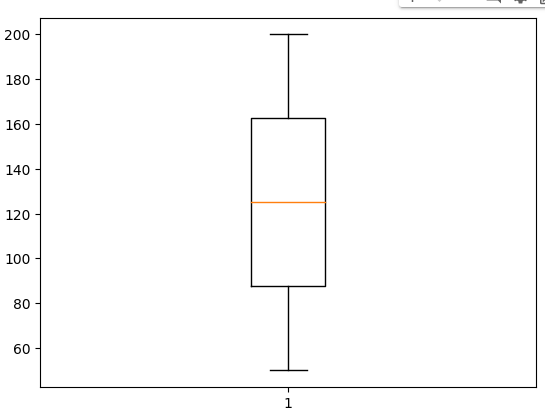


|  |
| --- |
| import matplotlib.pyplot as plt  price = [2.50, 1.23, 4.02, 3.25, 5.00, 4.40]  sales\_per\_day = [34, 62, 49, 22, 13, 19]  plt.scatter(price, sales\_per\_day)  plt.show() |



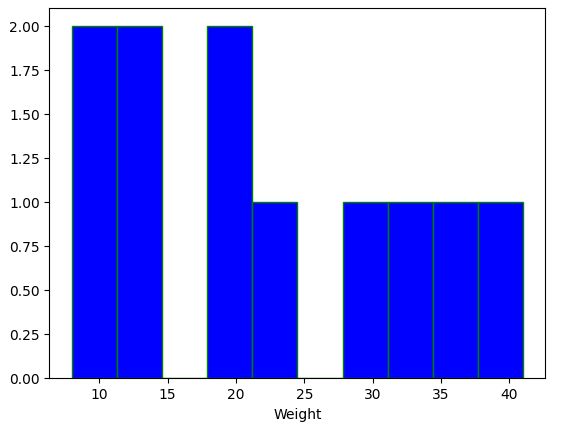
**b. Find the outliers using plot.**

|  |
| --- |
| import matplotlib.pyplot as plt  v = [50, 75, 100, 125, 150, 175, 200]  plt.boxplot(v)  plt.show() |

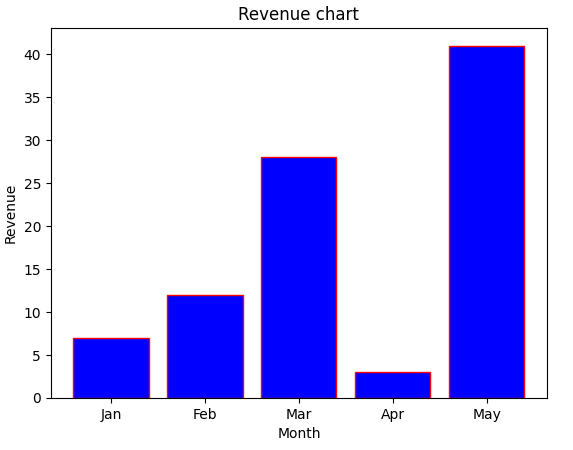


 c. Plot the histogram, bar chart and pie chart on sample data

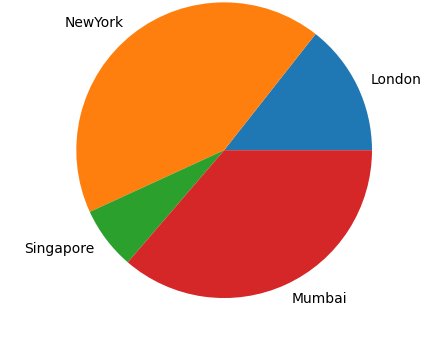
|  |
| --- |
| import matplotlib.pyplot as plt  v = [9, 13, 21, 8, 36, 22, 12, 41, 31, 33, 19]  plt.hist(v, color="blue", edgecolor="green")  plt.xlabel("Weight")  plt.show |



|  |
| --- |
| import matplotlib.pyplot as plt  H = [7, 12, 28, 3, 41]  M = ["Jan", "Feb", "Mar", "Apr", "May"]  plt.bar(M, H, color="blue", edgecolor="red")  plt.xlabel("Month")  plt.ylabel("Revenue")  plt.title("Revenue chart")  plt.show |



|  |
| --- |
| import matplotlib.pyplot as plt  x = [21, 62, 10, 53]  labels = ["London", "NewYork", "Singapore", "Mumbai"]  plt.pie(x, labels=labels) |



**15.visualizing geo graphical data with basemap**