**Ex No: 1**

**UNIVARIATE, BIVARIATE, MULTIVARIATE PROFILING – PYTHON**

**AIM**:

To analyse the dataset, get statistical description and to visualize it in python.

**Dataset Description:**

The dataset contains information on over 2,000 mobile phones from different brands. It includes details such as the storage capacity, RAM, screen size, camera specifications, battery capacity, and price of each device.

The dataset is structured as a CSV file with 7 columns:

* Brand: The brand name of the mobile phone.
* Model: The model name of the mobile phone.
* Storage: The amount of storage space available on the mobile phone in GB.
* RAM: The amount of random access memory available on the mobile phone in GB.
* Screen Size: The size of the mobile phone's screen in inches.
* Camera: The quality of the mobile phone's cameras, measured in megapixels.
* Battery Capacity: The amount of battery life the mobile phone has in mAh.
* Price: The price of the mobile phone in USD.

**Problem Statement**

The mobile phone price prediction problem is to develop a model that can predict the price of a mobile phone given a set of features. The target variable is the price of the mobile phone in USD. The goal of the problem is to develop a model that can accurately predict the price of a mobile phone given its features. This model can be used by a variety of stakeholders, including:

Mobile phone manufacturers: Manufacturers can use the model to develop a pricing strategy for their products. They can also use the model to identify the features that are most important to consumers and to determine how much they should charge for their phones based on those features.

Retailers: Retailers can use the model to set prices for mobile phones in their store. They can also use the model to compare the prices of different phones from different manufacturers and to ensure that they are charging a competitive price.

Consumers: Consumers can use the model to make informed decisions about which mobile phone to buy. They can use the model to compare the prices of different phones with different features and to find the best value for their money.

**PROGRAMS WITH OUTPUT:**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

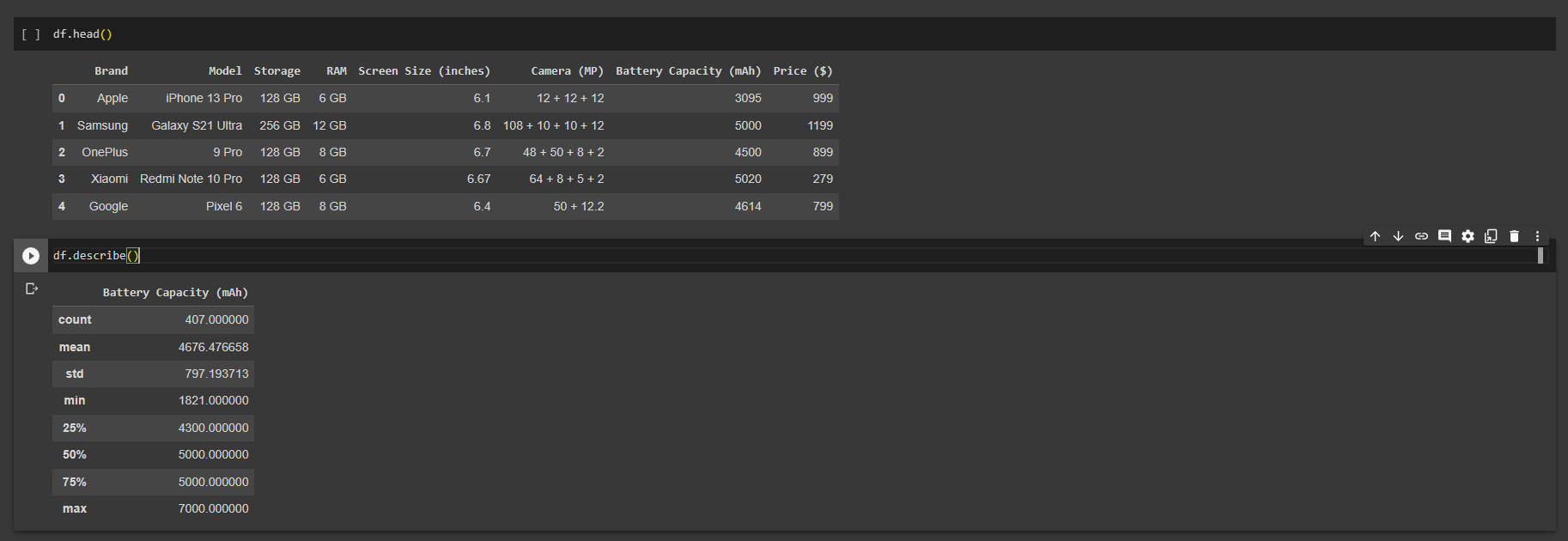
import seaborn as sns

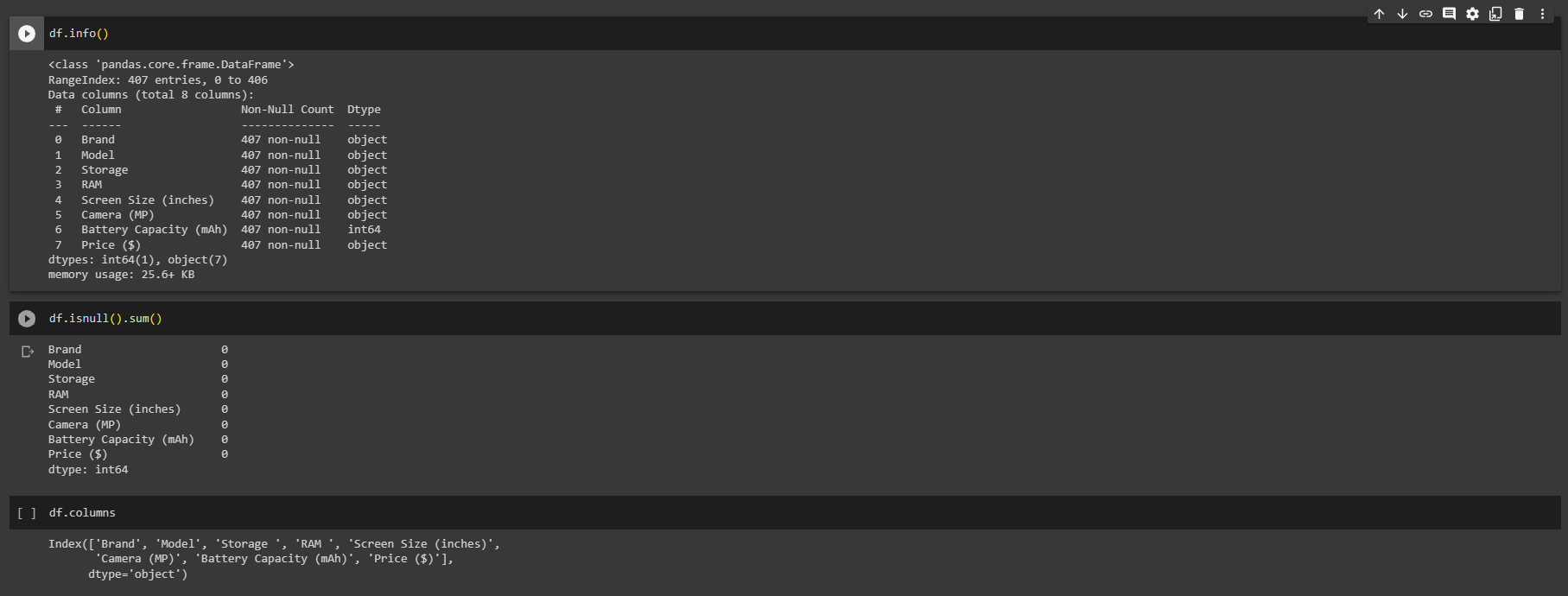
from google.colab import drive

drive.mount('/content/drive')

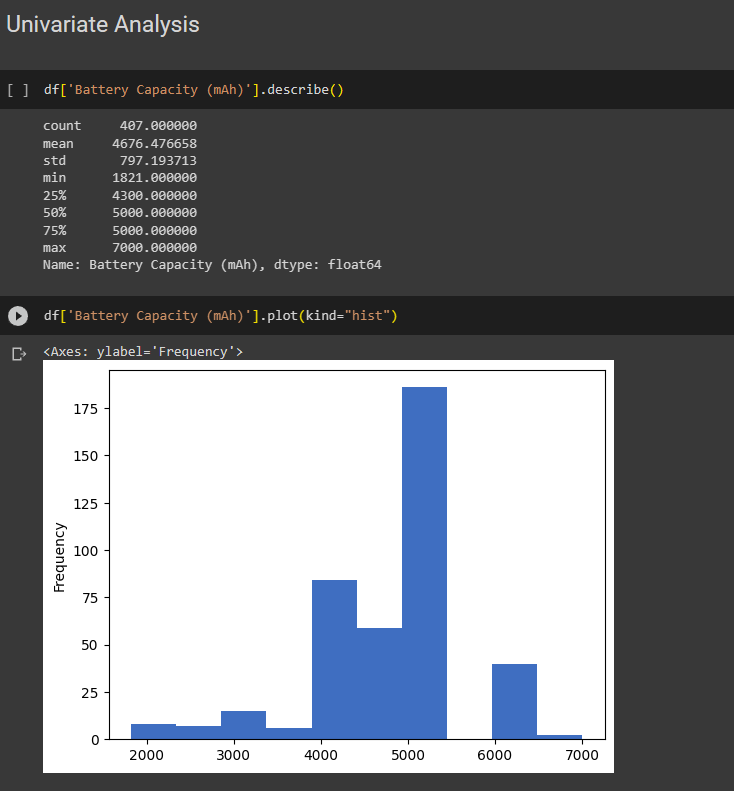
p1 = '/content/drive/MyDrive/Colab Notebooks/MVT/Mobile.csv'

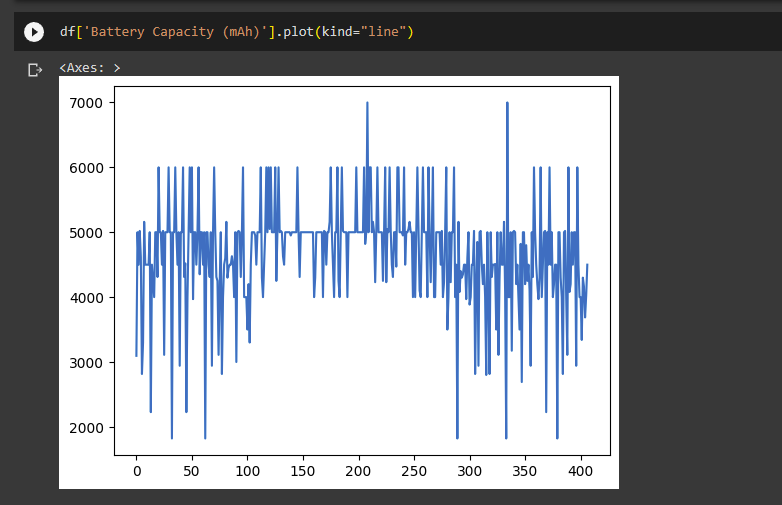
df = pd.read\_csv(p1)

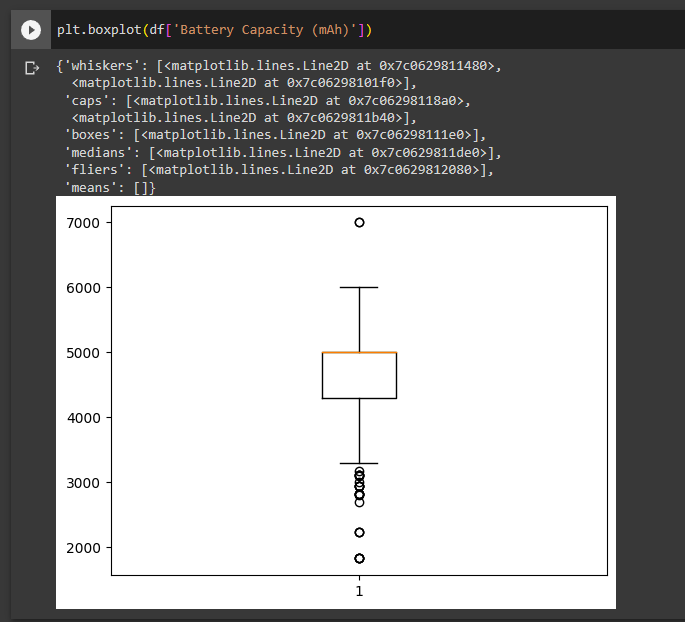


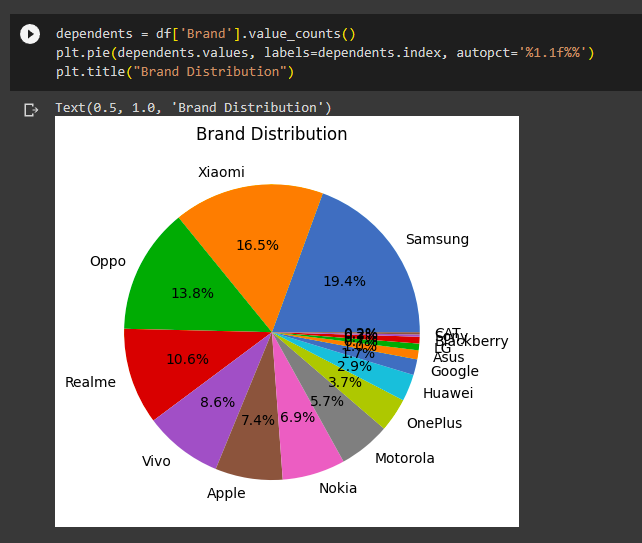


**Univariate Analysis**

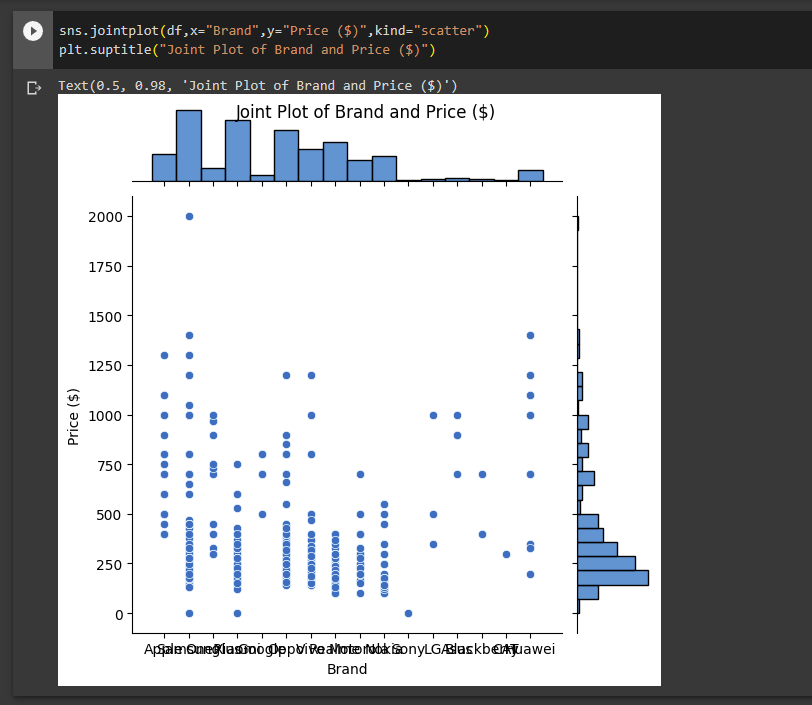


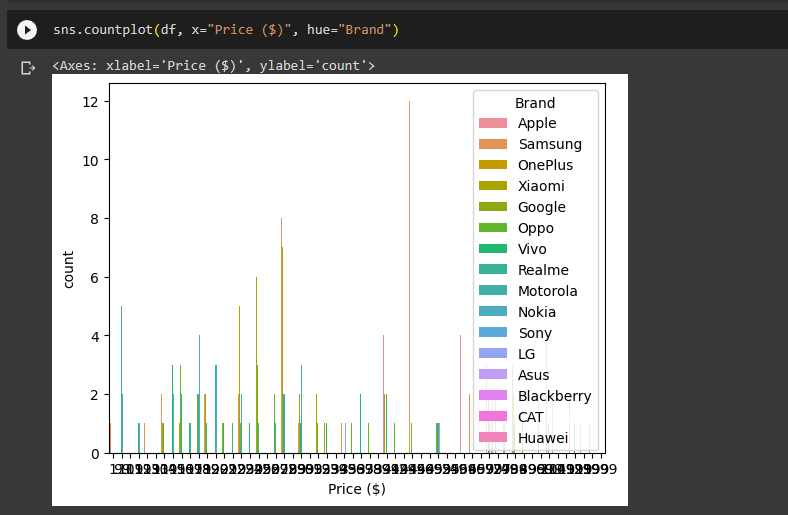


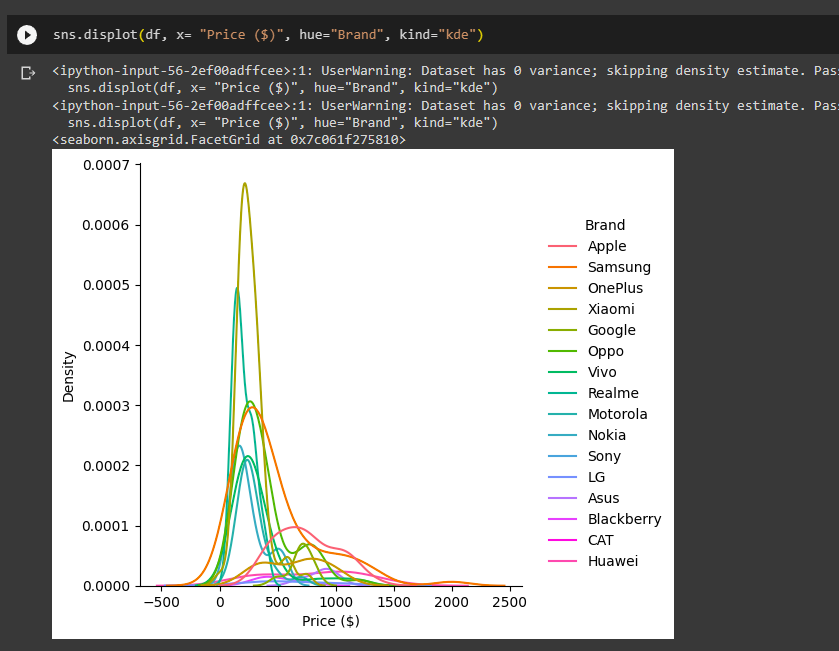


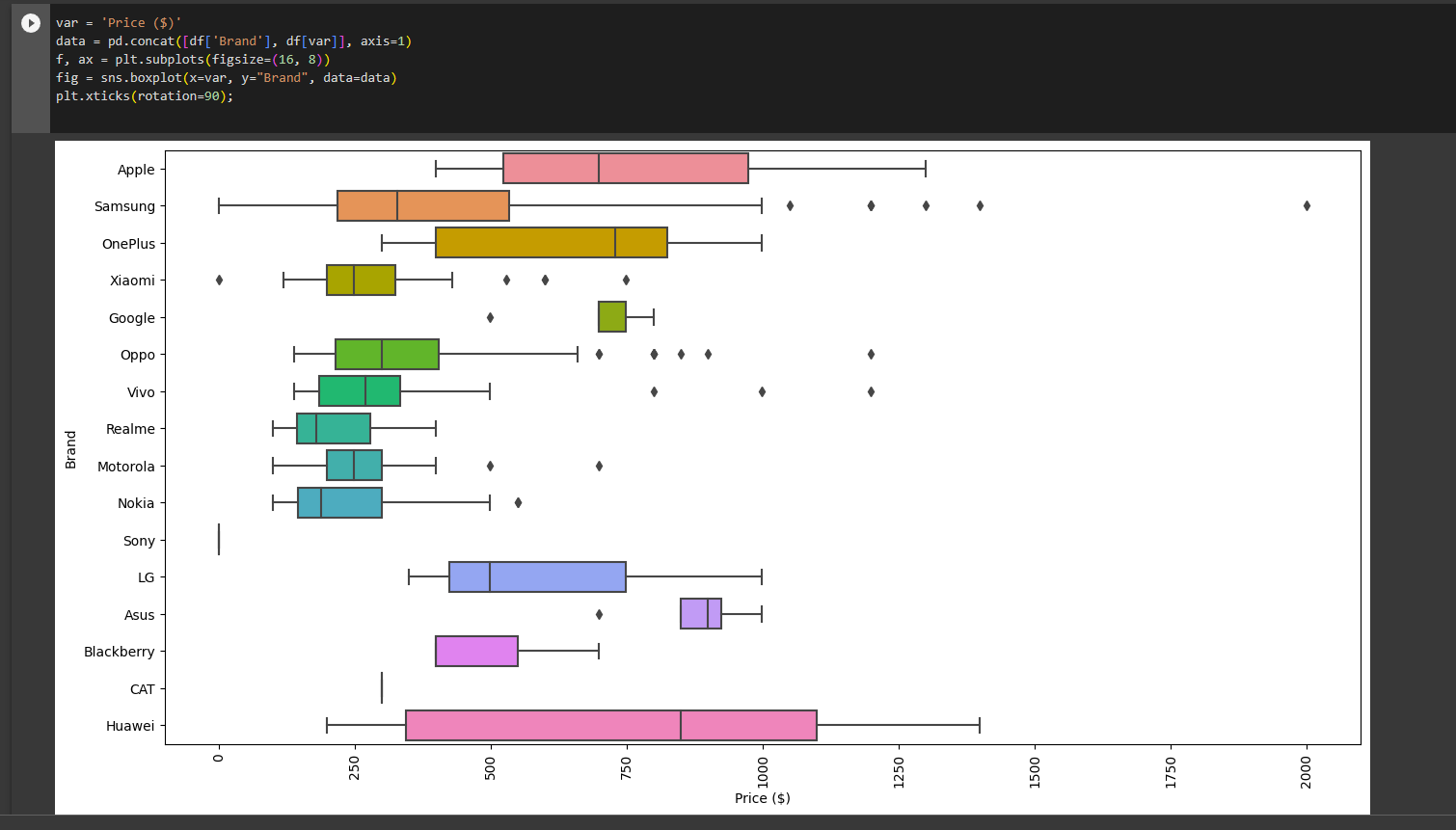


**BIVARIATE ANALYSIS**



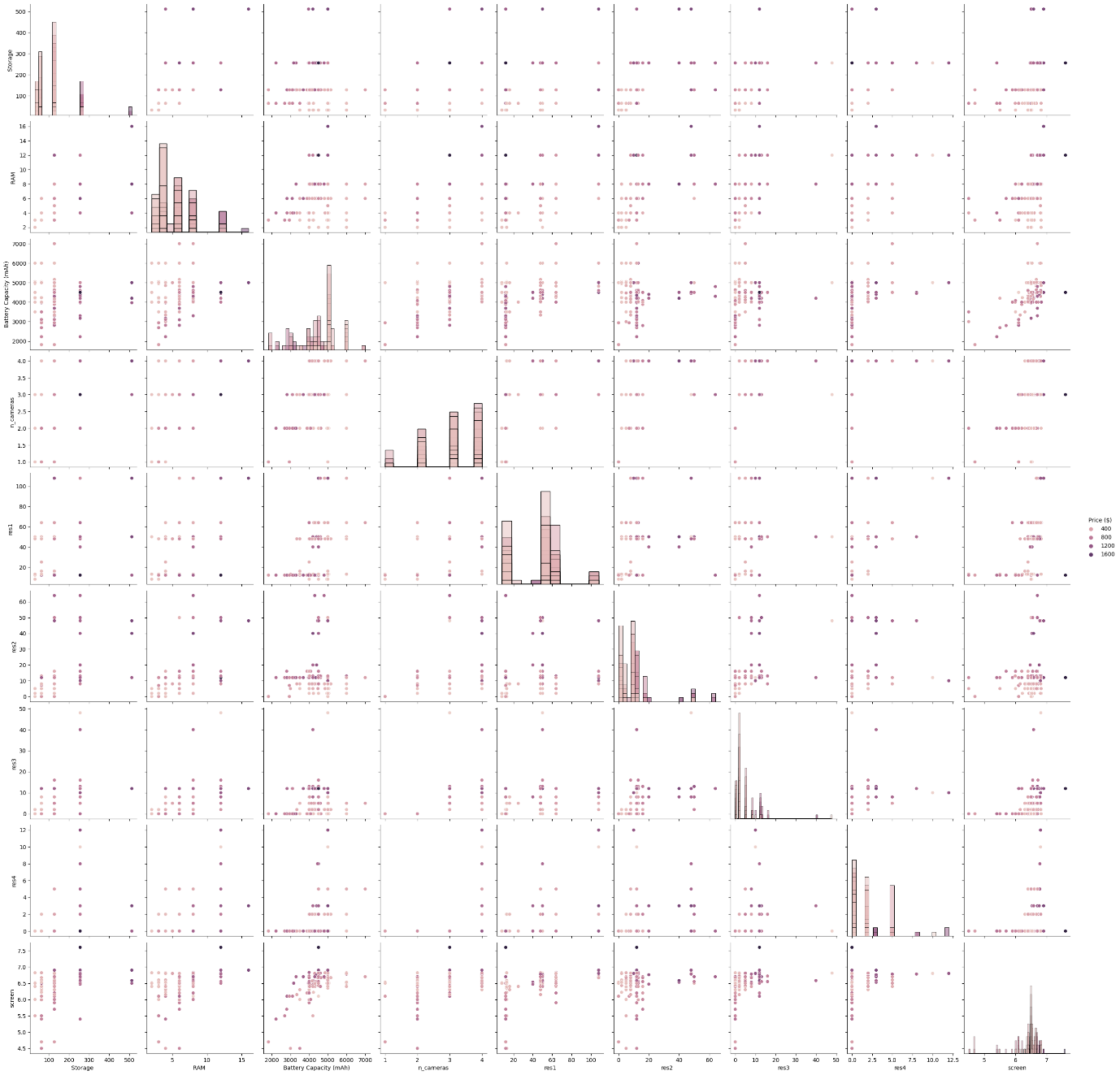






MULTI VARIATE ANALYSIS

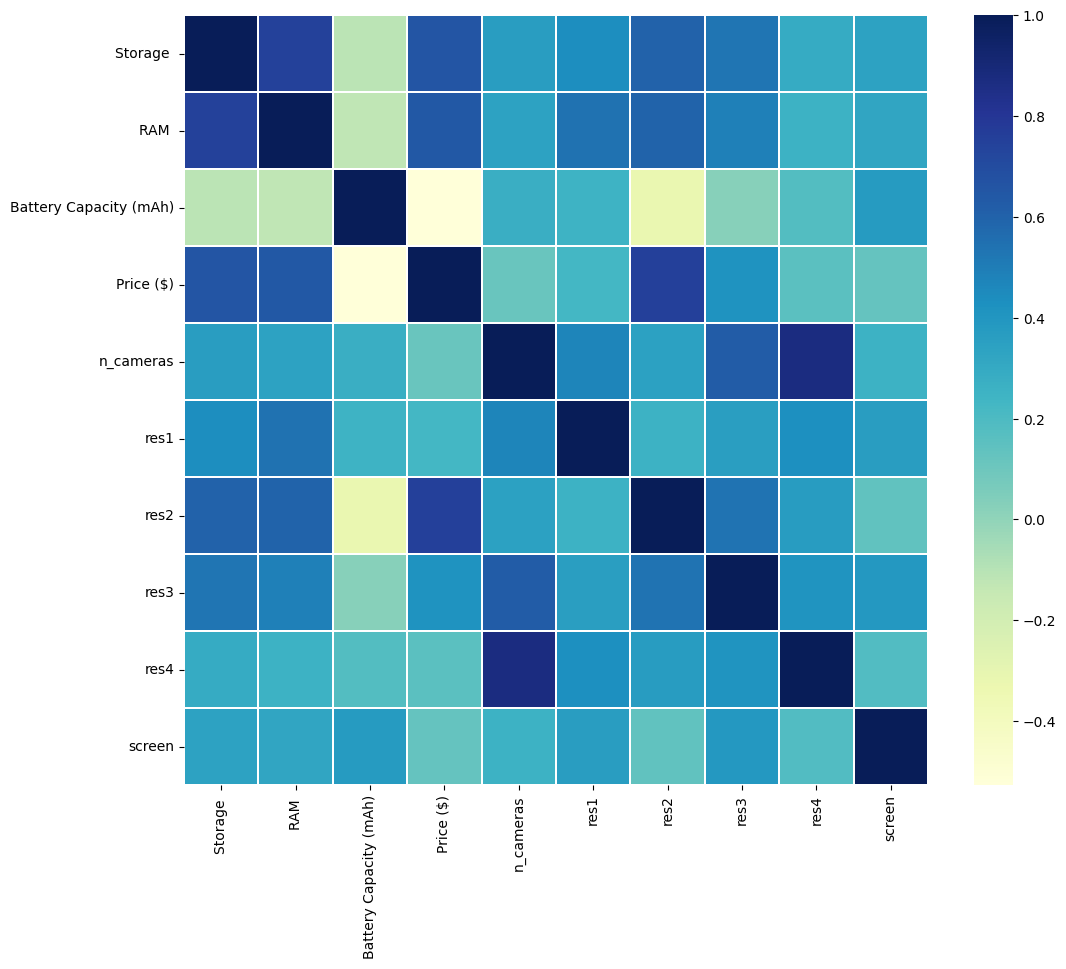
s=sns.pairplot(df,hue='Price ($)',size=3,diag\_kind='hist')



corrmat = df.corr(method='spearman')

f, ax = plt.subplots(figsize=(12, 10))

sns.heatmap(corrmat, ax=ax, cmap="YlGnBu", linewidths=0.1)



**CONCLUSION:**

The univariate, bivariate and multivariate data analysis has been done using the given dataset and the results have been analyzed using the above visualizations.

**Ex No: 2**

**Exploratory Data Analysis** (EDA)

**AIM**:

To perform exploratory data analysis for the dataset and obtain Measures of Central Tendency, Measure of Dispersion, Descriptive Statistics, Skewness and Kurtosis, and correlation using python.

**Dataset Description:**

The dataset contains information on over 2,000 mobile phones from different brands. It includes details such as the storage capacity, RAM, screen size, camera specifications, battery capacity, and price of each device.

The dataset is structured as a CSV file with 7 columns:

* Brand: The brand name of the mobile phone.
* Model: The model name of the mobile phone.
* Storage: The amount of storage space available on the mobile phone in GB.
* RAM: The amount of random access memory available on the mobile phone in GB.
* Screen Size: The size of the mobile phone's screen in inches.
* Camera: The quality of the mobile phone's cameras, measured in megapixels.
* Battery Capacity: The amount of battery life the mobile phone has in mAh.
* Price: The price of the mobile phone in USD.

**Problem Statement**

The mobile phone price prediction problem is to develop a model that can predict the price of a mobile phone given a set of features. The target variable is the price of the mobile phone in USD. The goal of the problem is to develop a model that can accurately predict the price of a mobile phone given its features. This model can be used by a variety of stakeholders, including:

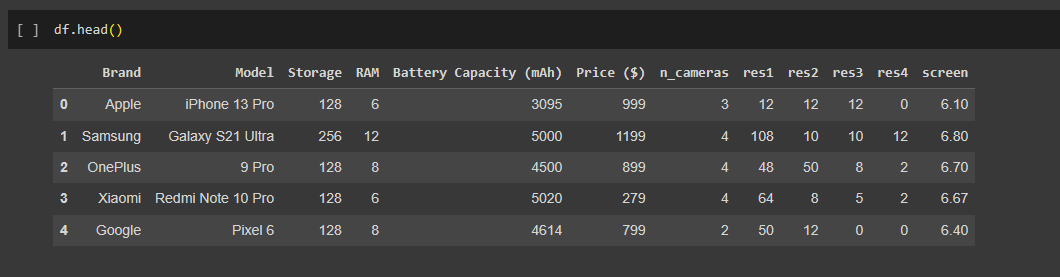
Mobile phone manufacturers: Manufacturers can use the model to develop a pricing strategy for their products. They can also use the model to identify the features that are most important to consumers and to determine how much they should charge for their phones based on those features.

Retailers: Retailers can use the model to set prices for mobile phones in their store. They can also use the model to compare the prices of different phones from different manufacturers and to ensure that they are charging a competitive price.

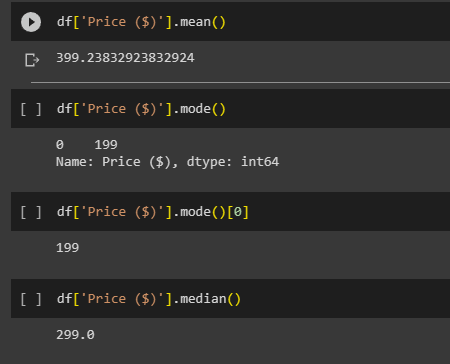
Consumers: Consumers can use the model to make informed decisions about which mobile phone to buy. They can use the model to compare the prices of different phones with different features and to find the best value for their money.

**PROGRAMS WITH OUTPUT:**

df.head()

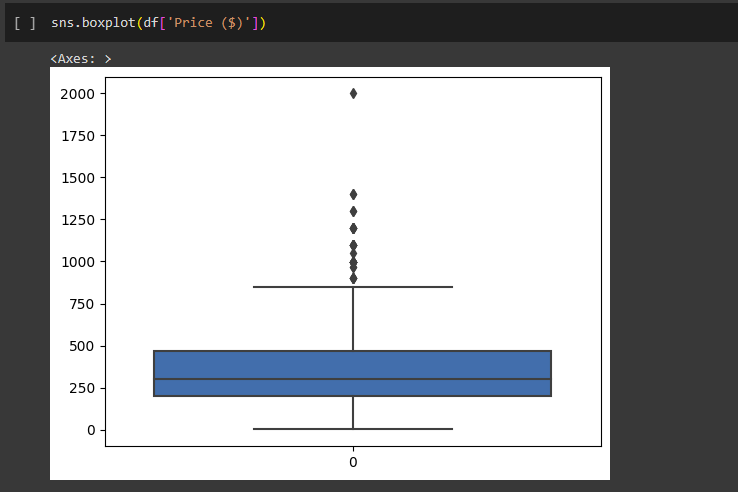
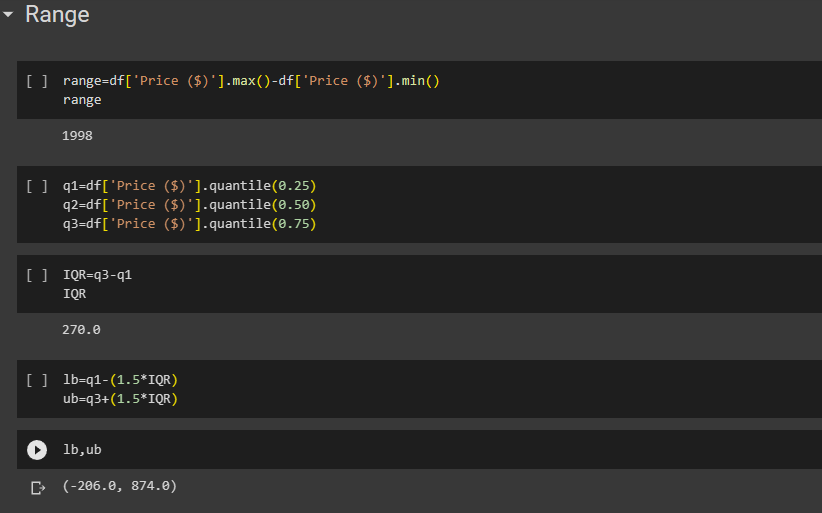
****

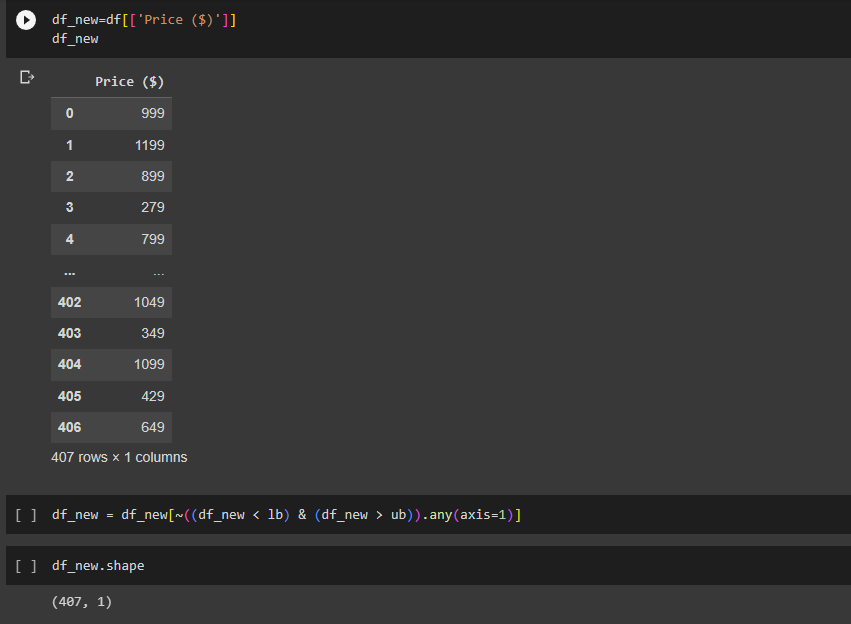
**Measures of Central Tendency**



**Measure of Dispersion**

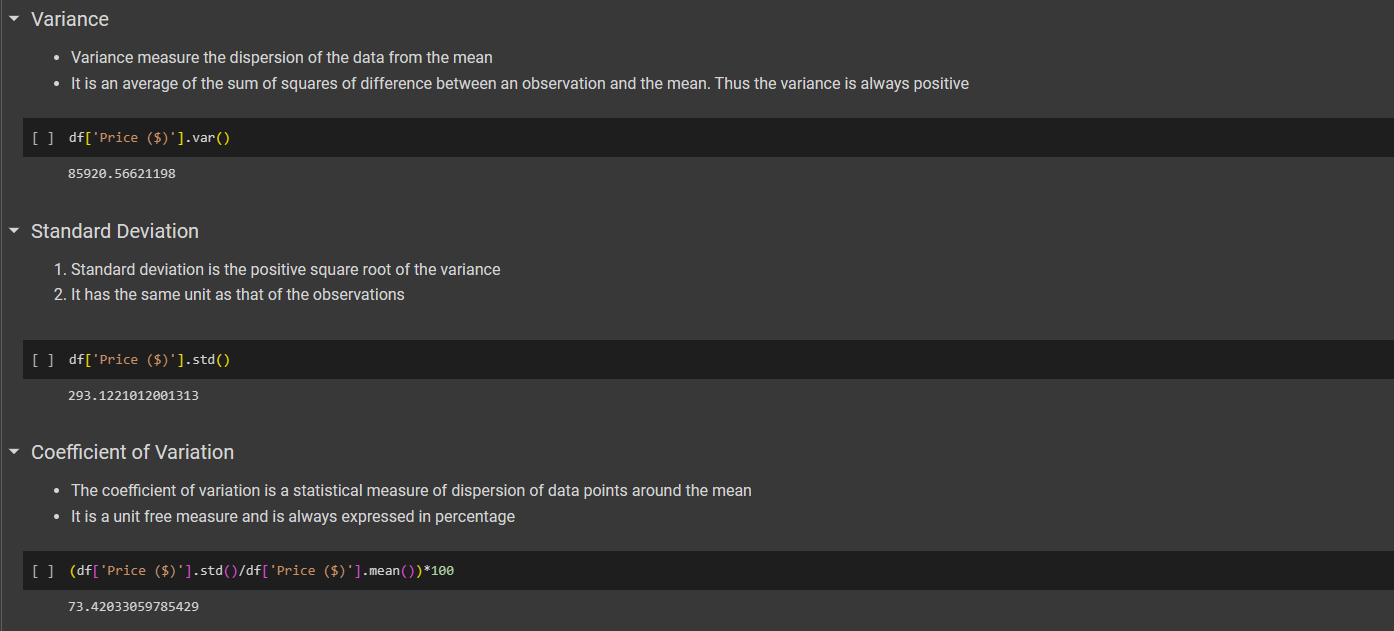
1. **Range**

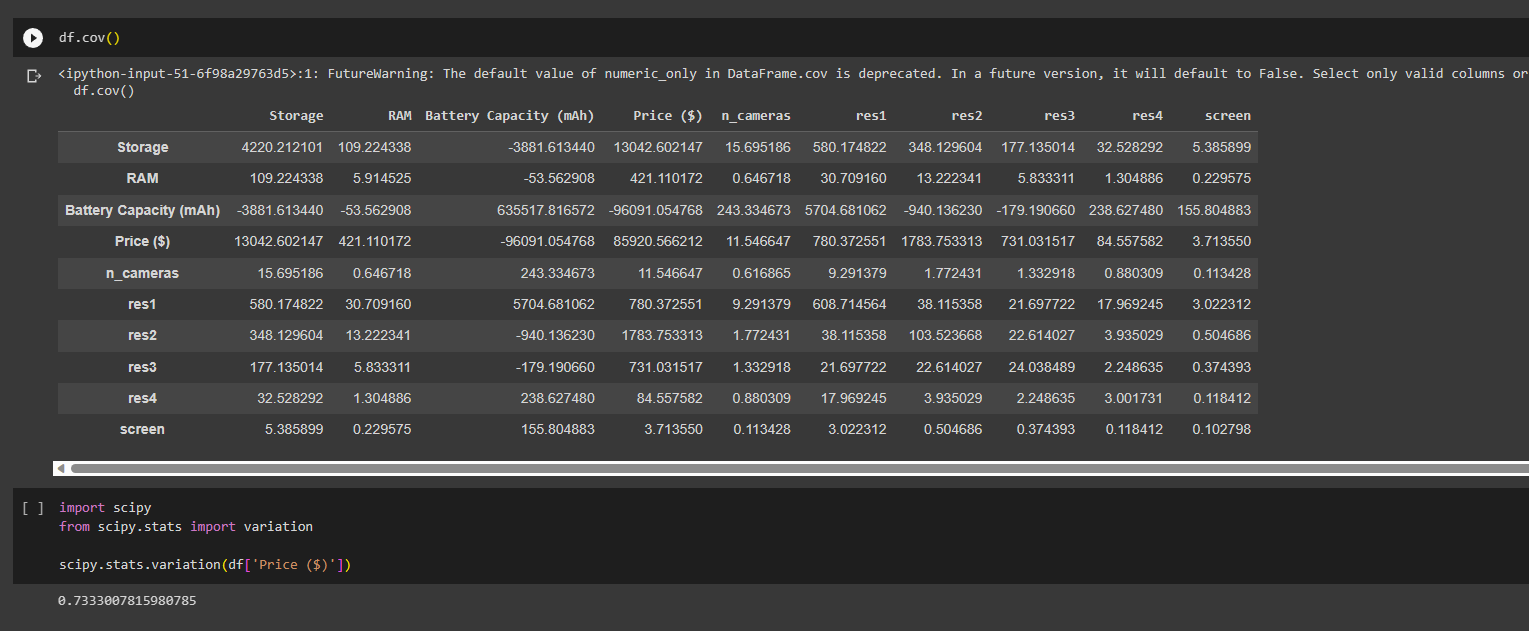




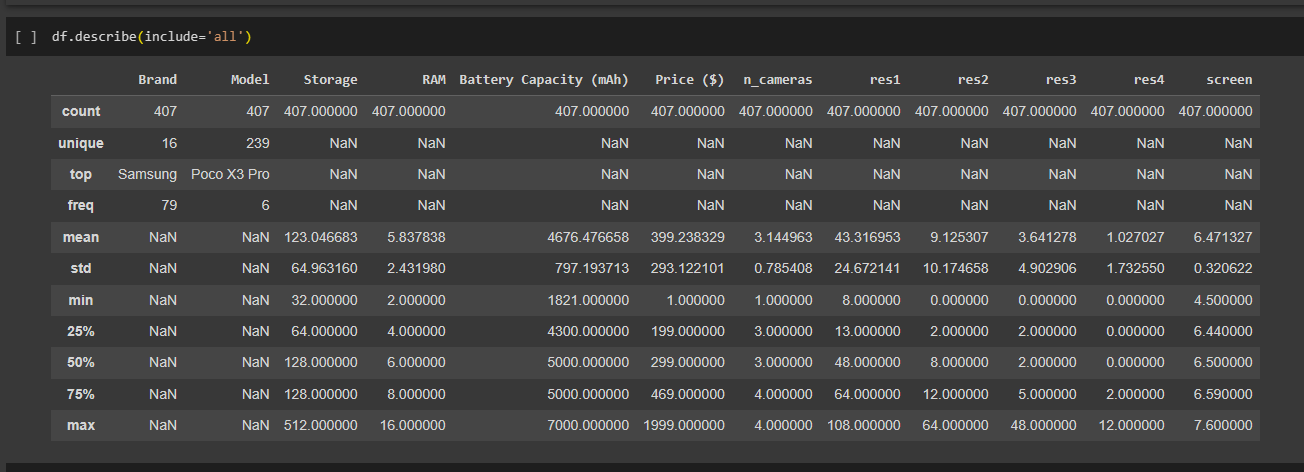


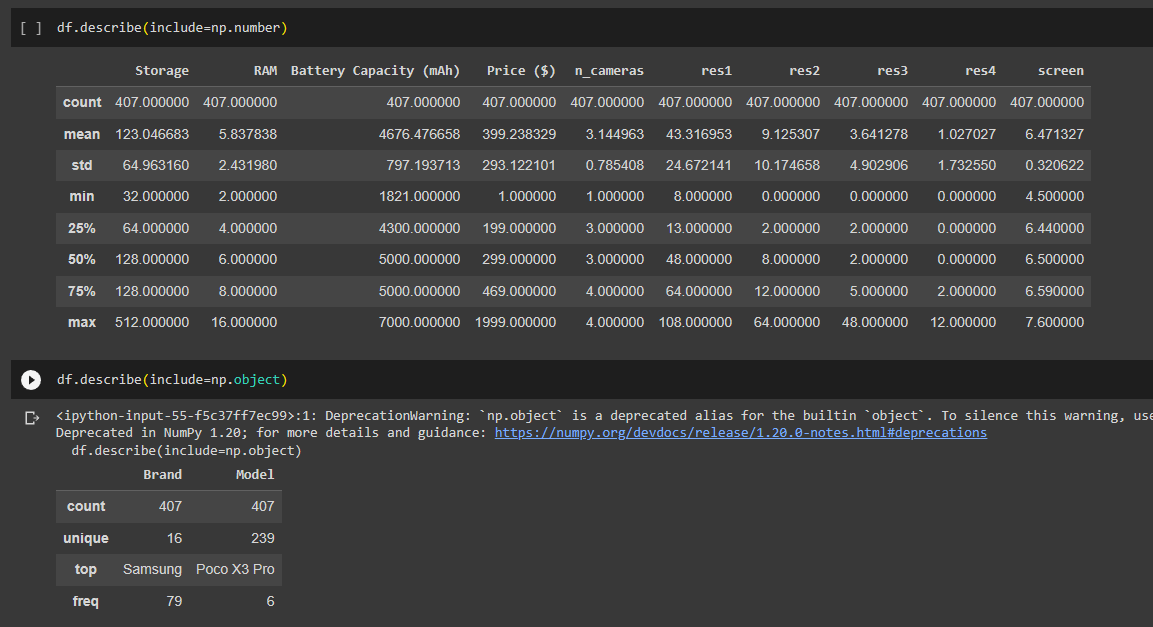
1. **Variance And Standard Deviation**



****

**Descriptive Statistics**

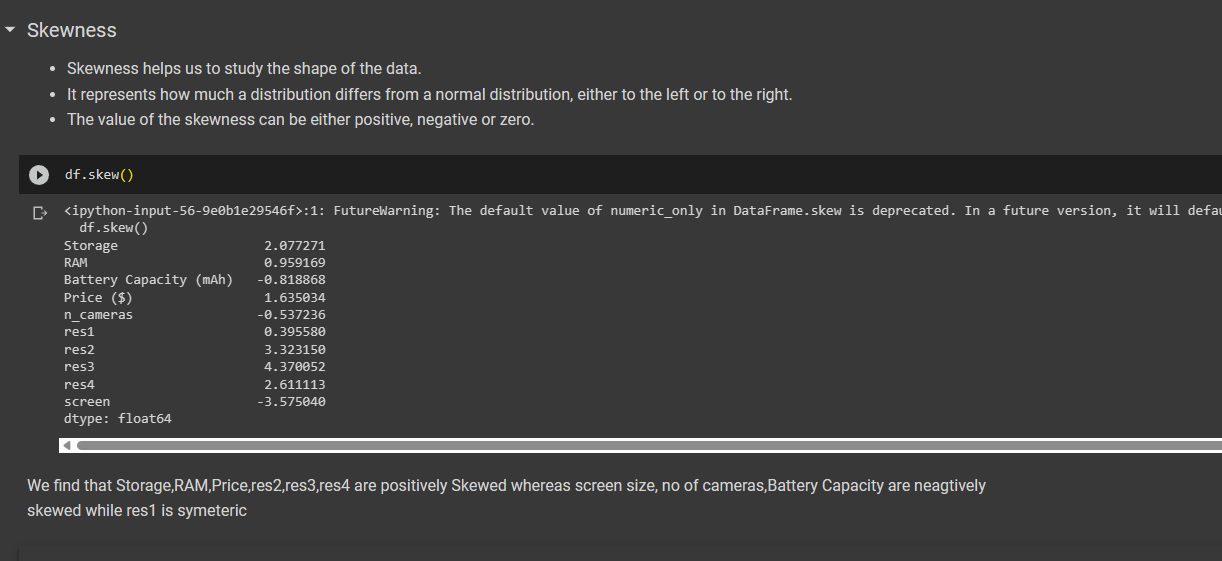
****

****

**Shape of the data**

**1.Skewness**

* Skewness helps us to study the shape of the data.
* It represents how much a distribution differs from a normal distribution, either to the left or to the right.
* The value of the skewness can be either positive, negative or zero.

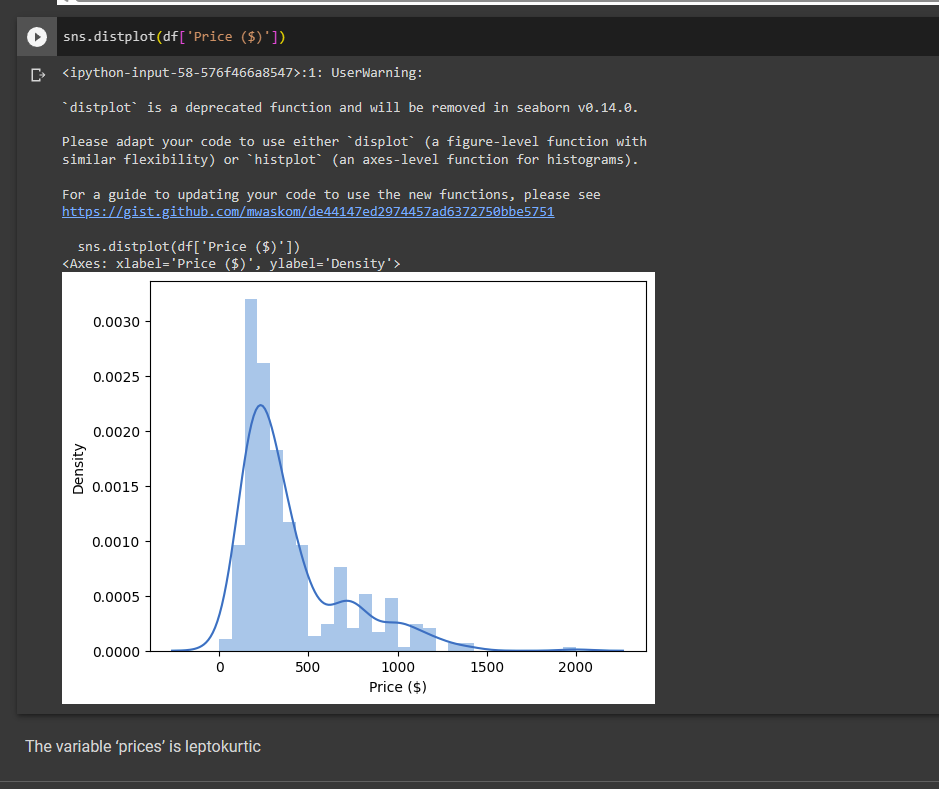
****

We find that Storage,RAM,Price,res2,res3,res4 are positively Skewed whereas screen size, no of cameras,Battery Capacity are neagtively skewed while res1 is symeteric

**2.Kurtosis**

* Kurtosis measures the peakedness of the distribution
* In other words, kurtosis is a statistical measure that defines how the tails of the distribution differ from the normal distribution
* Kurtosis identifies whether the tails of a given distribution contain extreme values

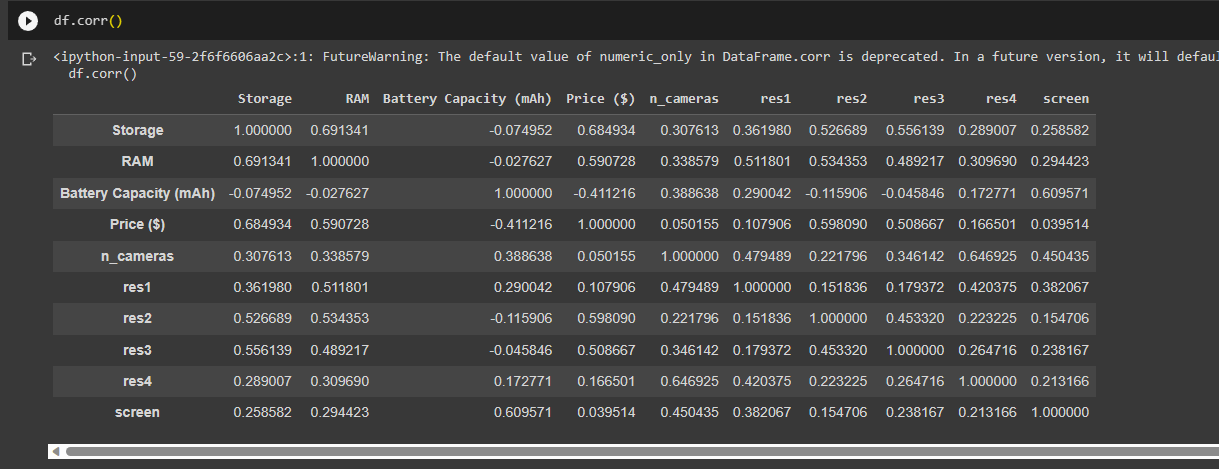


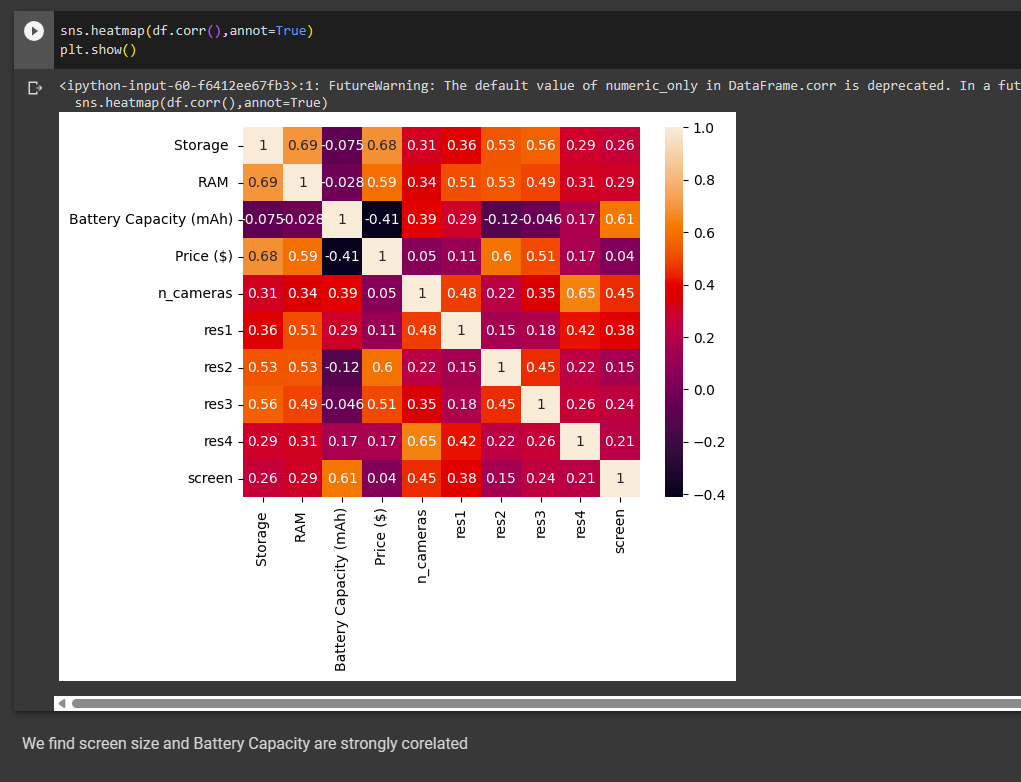


The variable ‘prices’ is leptokurtic

**Correlation**

* It shows whether pairs of variables are related to each other
* If there is correlation, it shows how strong the correlation is
* Correlation takes values between -1 to +1, where values close to +1 represents strong positive correlation while values close to -1 represents strong negative correlation

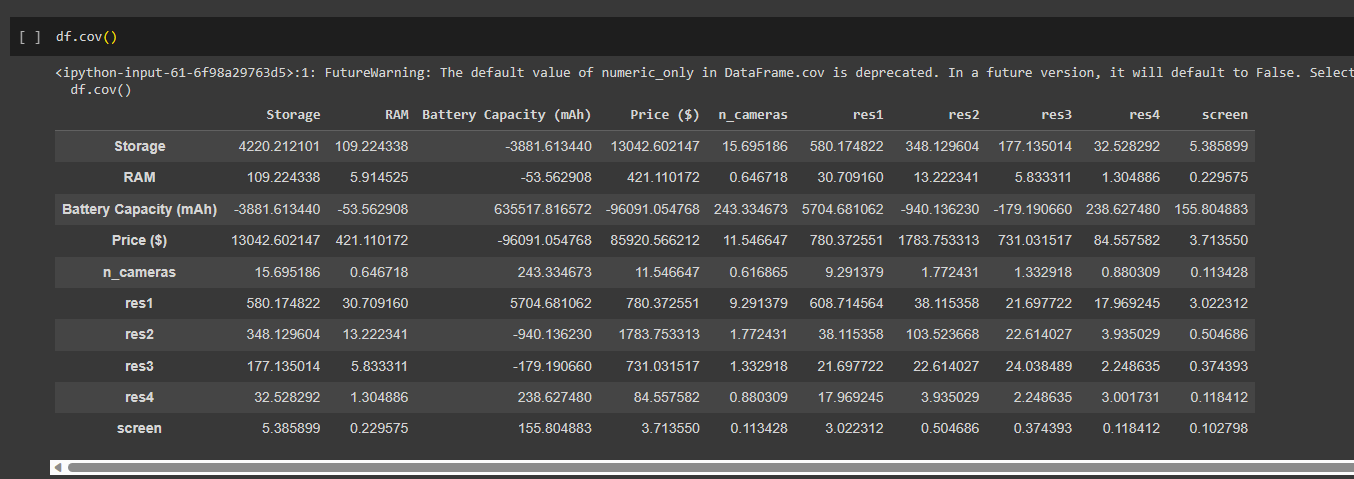


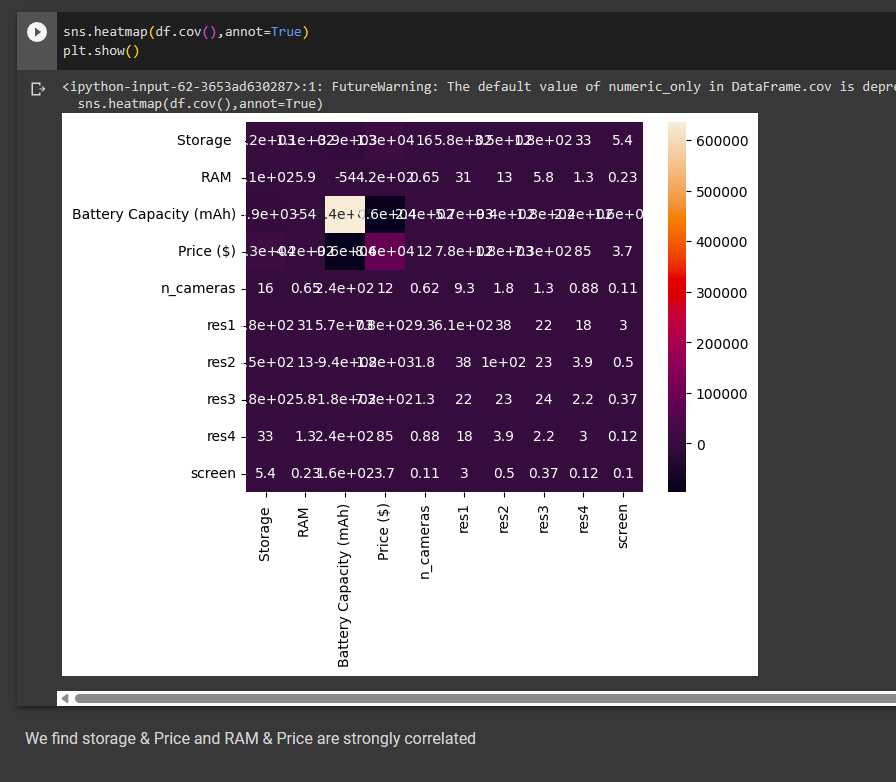


We find screen size and Battery Capacity are strongly corelated

**Covariance**

* It is the relationship between a pair of random variables where change in one variable causes change in another variable
* It can take any value between -infinity to +infinity, where the negative value represents the negative relationship whereas a positive value represents the positive relationship





**CONCLUSION:**

The exploratory data analysis has been done using the given dataset and the results have been analysed using the above visualizations.

**Ex No: 3**

**LINEAR REGRESSION**

**AIM**:

To perform prediction with Linear regression using random linear regression dataset.

**Dataset Description:**

The dataset contains information on over 2,000 mobile phones from different brands. It includes details such as the storage capacity, RAM, screen size, camera specifications, battery capacity, and price of each device.

The dataset is structured as a CSV file with 7 columns:

* Brand: The brand name of the mobile phone.
* Model: The model name of the mobile phone.
* Storage: The amount of storage space available on the mobile phone in GB.
* RAM: The amount of random access memory available on the mobile phone in GB.
* Screen Size: The size of the mobile phone's screen in inches.
* Camera: The quality of the mobile phone's cameras, measured in megapixels.
* Battery Capacity: The amount of battery life the mobile phone has in mAh.
* Price: The price of the mobile phone in USD.

**Problem Statement**

The mobile phone price prediction problem is to develop a model that can predict the price of a mobile phone given a set of features. The target variable is the price of the mobile phone in USD. The goal of the problem is to develop a model that can accurately predict the price of a mobile phone given its features. This model can be used by a variety of stakeholders, including:

Mobile phone manufacturers: Manufacturers can use the model to develop a pricing strategy for their products. They can also use the model to identify the features that are most important to consumers and to determine how much they should charge for their phones based on those features.

Retailers: Retailers can use the model to set prices for mobile phones in their store. They can also use the model to compare the prices of different phones from different manufacturers and to ensure that they are charging a competitive price.

Consumers: Consumers can use the model to make informed decisions about which mobile phone to buy. They can use the model to compare the prices of different phones with different features and to find the best value for their money.

**PROGRAMS WITH OUTPUT:**

Linear Regression Using numpy

*def mean(values):*

*return sum(values)/float(len(values))*

*def variance(values,mean):*

*return sum([(x-mean)\*\*2 for x in values])*

*def covariance(x,mean\_x,y,mean\_y):*

*covar=0.0*

*for i in range(len(x)):*

*covar+=(x[i]-mean\_x) \* (y[i]-mean\_y)*

*return covar*

*def coefficients(dataset):*

*b1=covariance(x,mean\_x,y,mean\_y)/variance(x,mean\_x)*

*b0=mean\_y-b1\*mean\_x*

*return[b0,b1]*

*def simple\_linear\_regression(train,test):*

*for row in test:*

*ytest = b0 + b1 \* row[0]*

*return ytest*

*dataset=[[50,28],[60,40],[48,45],[70,50],[55,50],[60,38],[45,20]]*

*x=[row[0] for row in dataset]*

*y=[row[1] for row in dataset]*

*mean\_x=mean(x)*

*mean\_y=mean(y)*

*variance\_x=variance(x,mean\_x)*

*variance\_y=variance(y,mean\_y)*

*print('x stats:mean=%.3f variance=%.3f' % (mean\_x,variance\_x))*

*print('y stats:mean=%.3f variance=%.3f' % (mean\_y,variance\_y))*

*covar = covariance(x,mean\_x,y,mean\_y)*

*print('covariance: %.3f' % (covar))*

*b0,b1 = coefficients(dataset)*

*print('coefficients:b0=%.3f,b1=%.3f' % (b0,b1))*

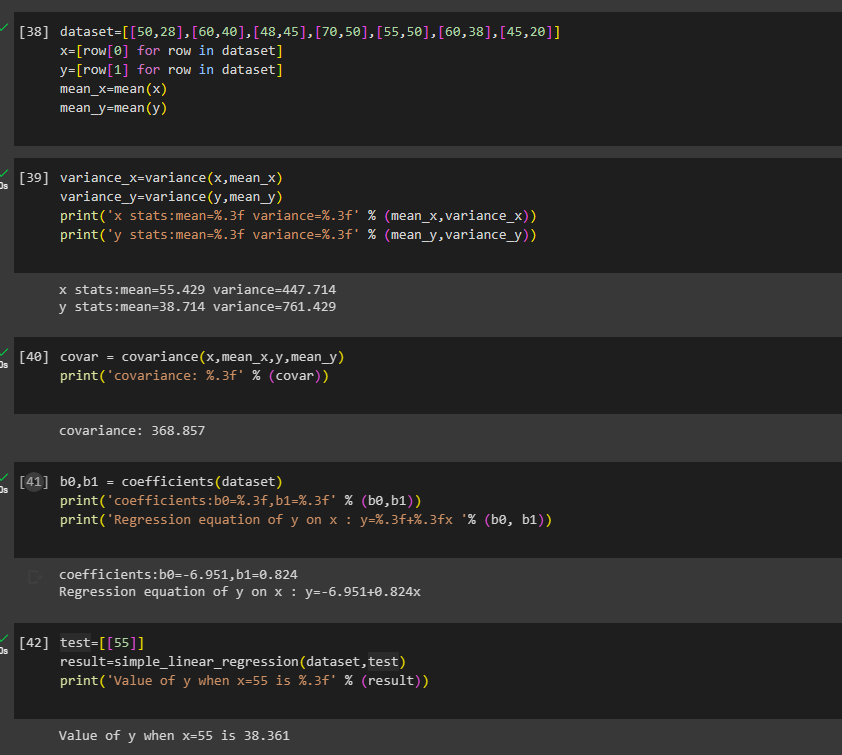
*print('Regression equation of y on x : y=%.3f+%.3fx '% (b0, b1))*

*test=[[55]]*

*result=simple\_linear\_regression(dataset,test)*

*print('Value of y when x=55 is %.3f' % (result))*





Linear Regression Using Sklearn

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

*Xtrain,Xtest,ytrain,ytest=train\_test\_split(X,y,test\_size=0.3,random\_state=1)*

*from sklearn.preprocessing import StandardScaler*

*scaler=StandardScaler()*

*Xtrain=scaler.fit\_transform(Xtrain)*

*Xtest=scaler.transform(Xtest)*

*from sklearn.linear\_model import LinearRegression*

*model = LinearRegression()*

*model.fit(Xtrain, ytrain)*

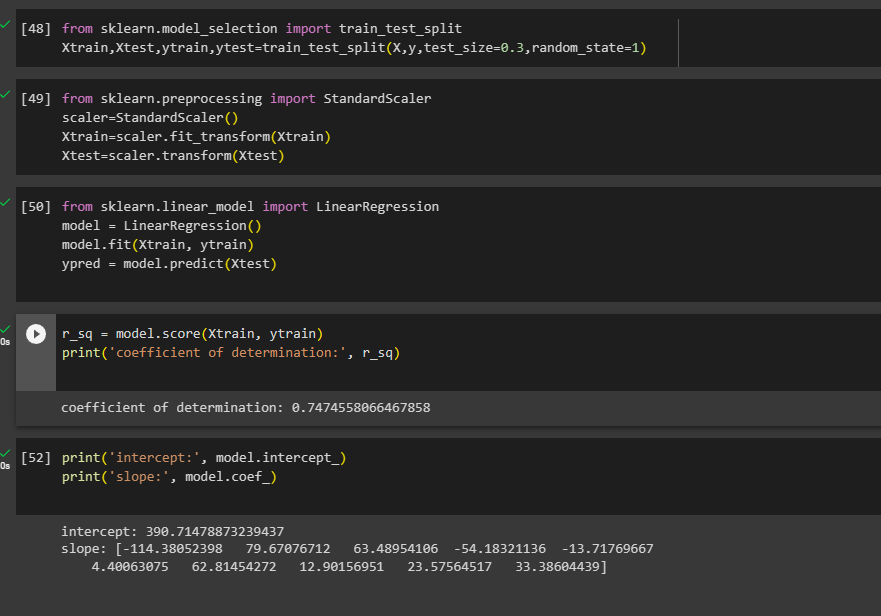
*ypred = model.predict(Xtest)*

*r\_sq = model.score(Xtrain, ytrain)*

*print('coefficient of determination:', r\_sq)*

*print('intercept:', model.intercept\_)*

*print('slope:', model.coef\_)*



**CONCLUSION:**

The predicted output is displayed using the linear regression model trained with the given dataset and results are verified.

**Ex No: 4**

**PRINCIPAL COMPONENT ANALYSIS**

**AIM**:

To perform Principal component analysis using Mobile dataset.

**Dataset Description:**

The dataset contains information on over 2,000 mobile phones from different brands. It includes details such as the storage capacity, RAM, screen size, camera specifications, battery capacity, and price of each device.

The dataset is structured as a CSV file with 7 columns:

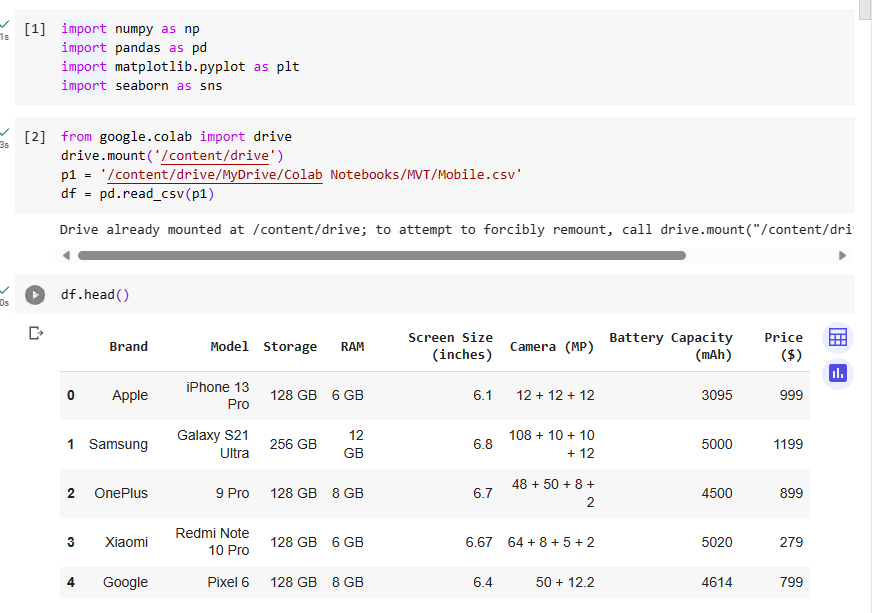
* Brand: The brand name of the mobile phone.
* Model: The model name of the mobile phone.
* Storage: The amount of storage space available on the mobile phone in GB.
* RAM: The amount of random access memory available on the mobile phone in GB.
* Screen Size: The size of the mobile phone's screen in inches.
* Camera: The quality of the mobile phone's cameras, measured in megapixels.
* Battery Capacity: The amount of battery life the mobile phone has in mAh.
* Price: The price of the mobile phone in USD.

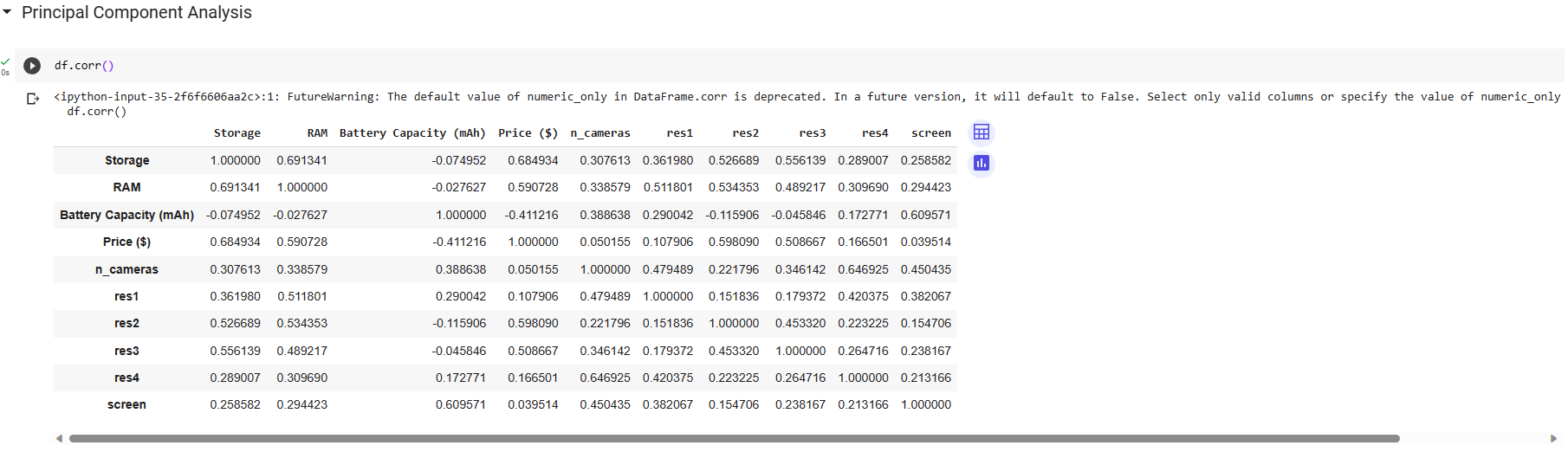
**PROCEDURE:**

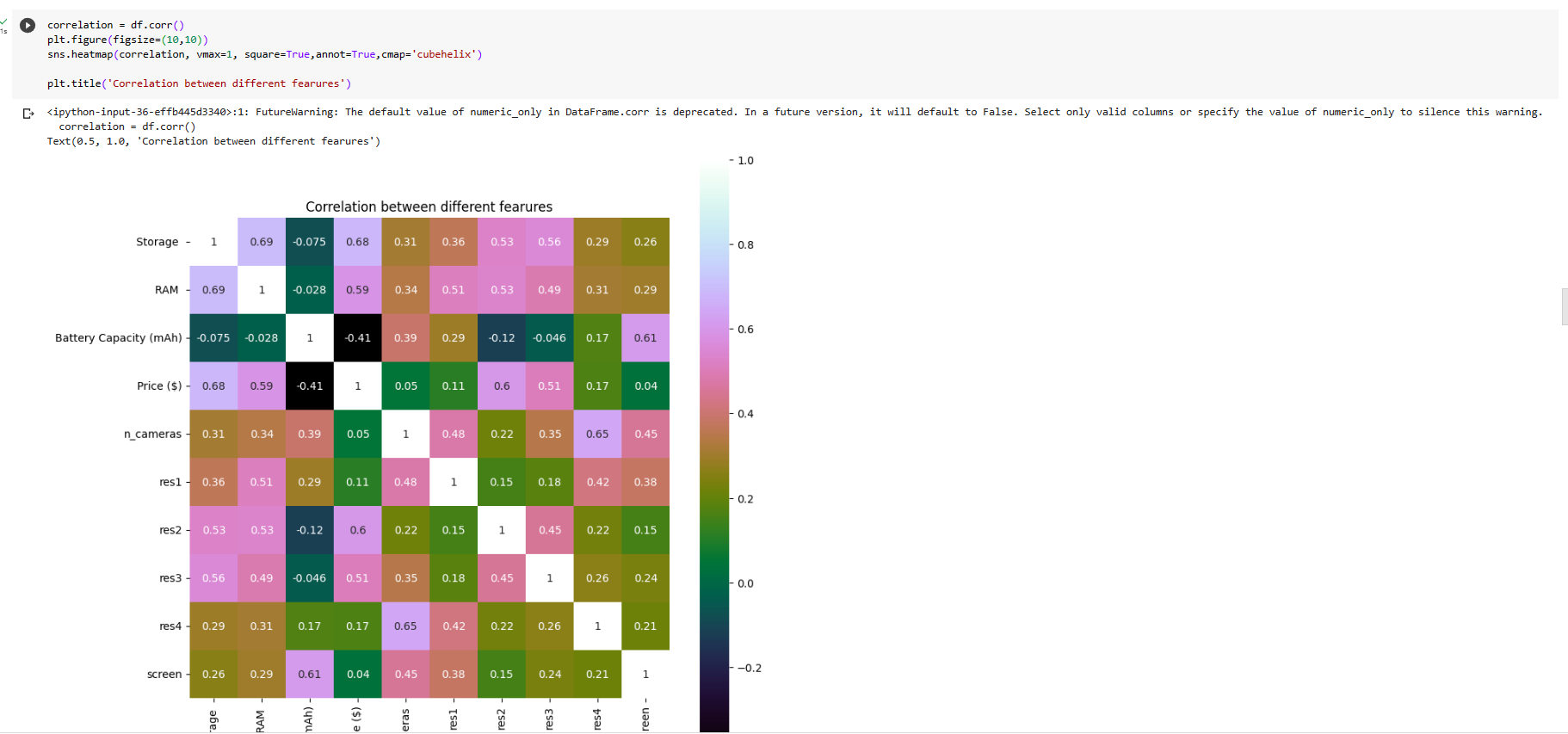
1. Import the necessary library functions.
2. Load the required dataset into the dataframe.
3. Print the head and shape of the dataset to find the dimensions of the given data.
4. Load the training dataset and fit the data into the PCA model.
5. Display the heatmap for the correlation.
6. Load the test dataset and predict the value using the model
7. Display the results of the output predicted from the model.

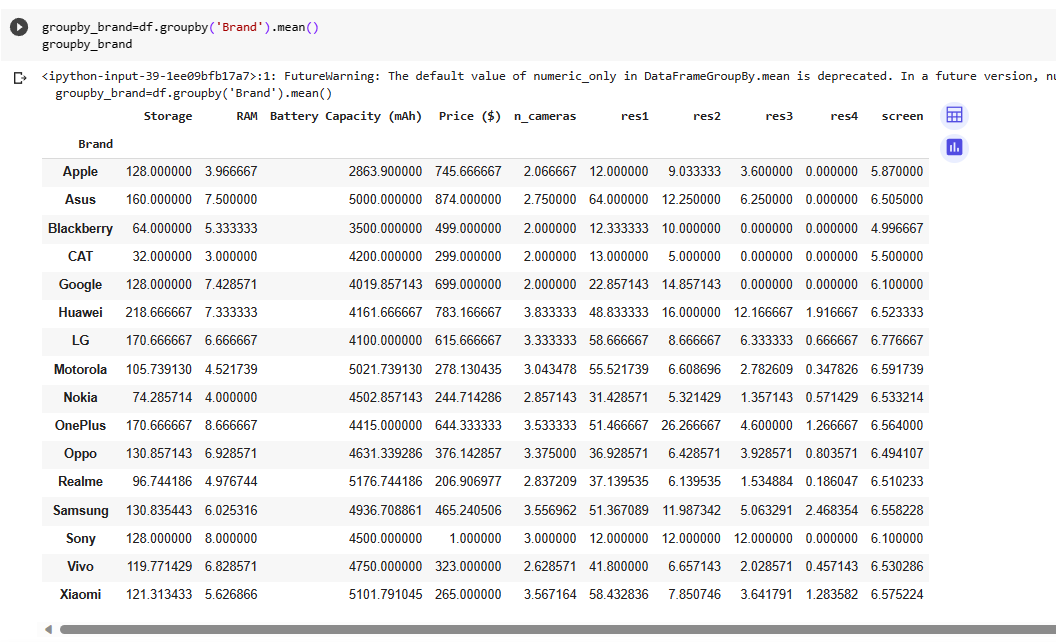
**PROGRAM AND OUTPUT:**

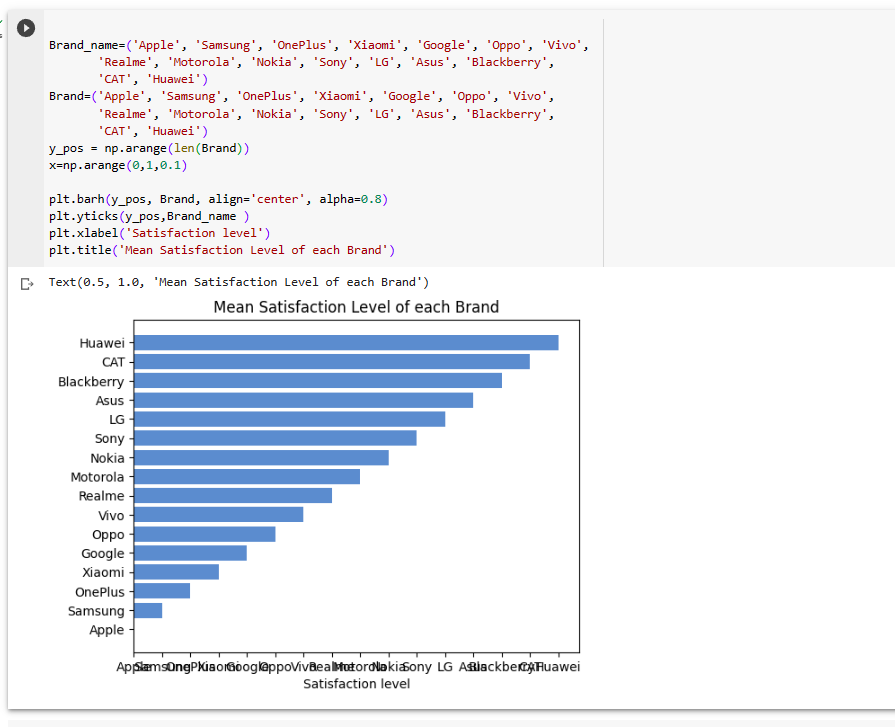
Importing All required libraries and loading the dataset.

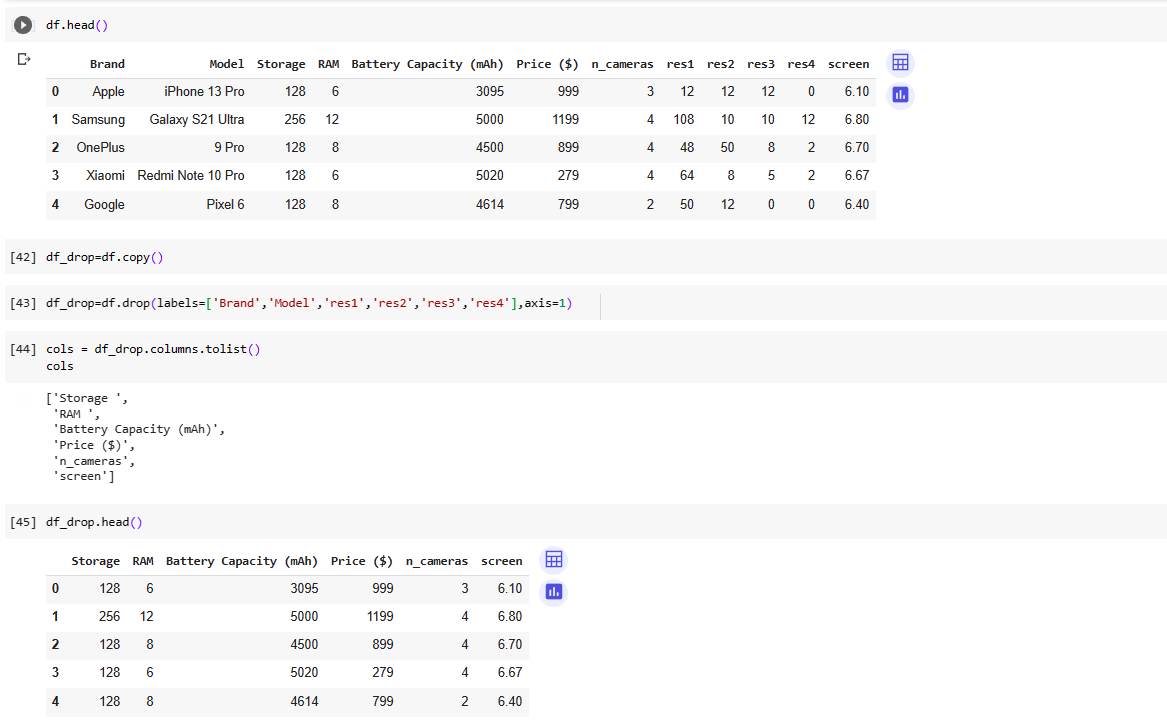
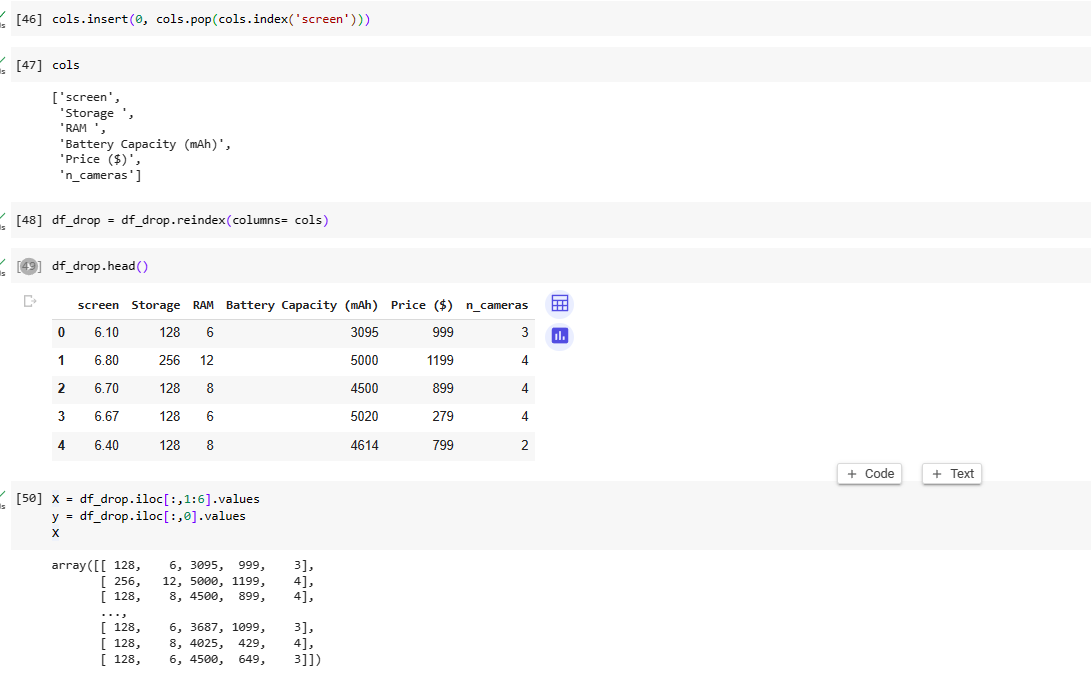
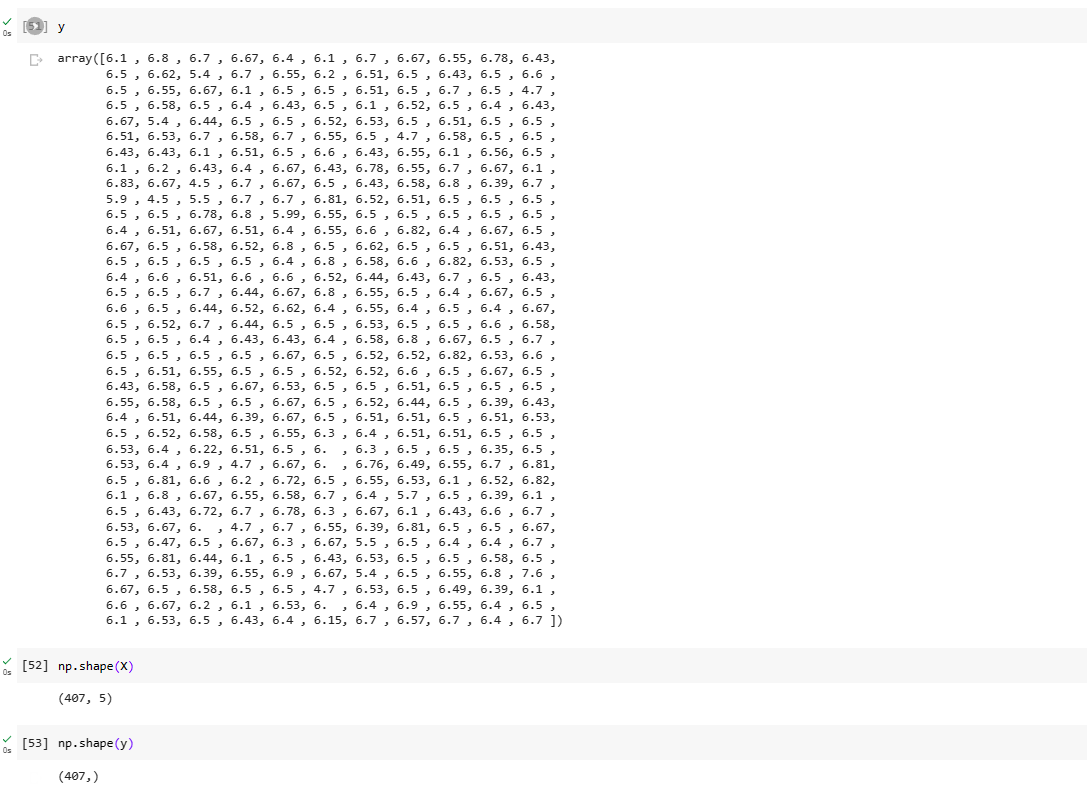
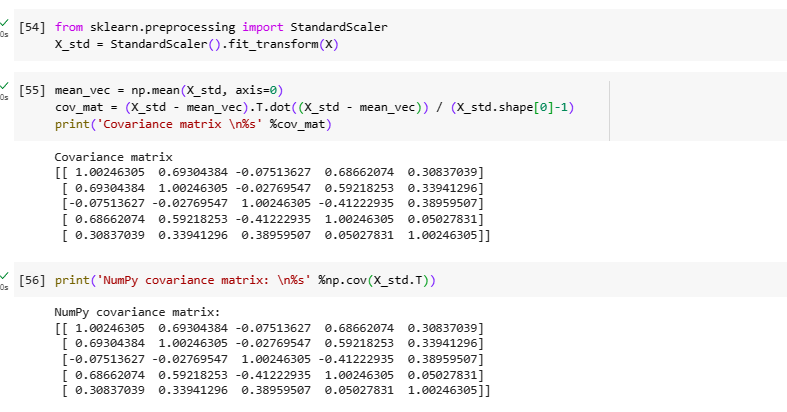
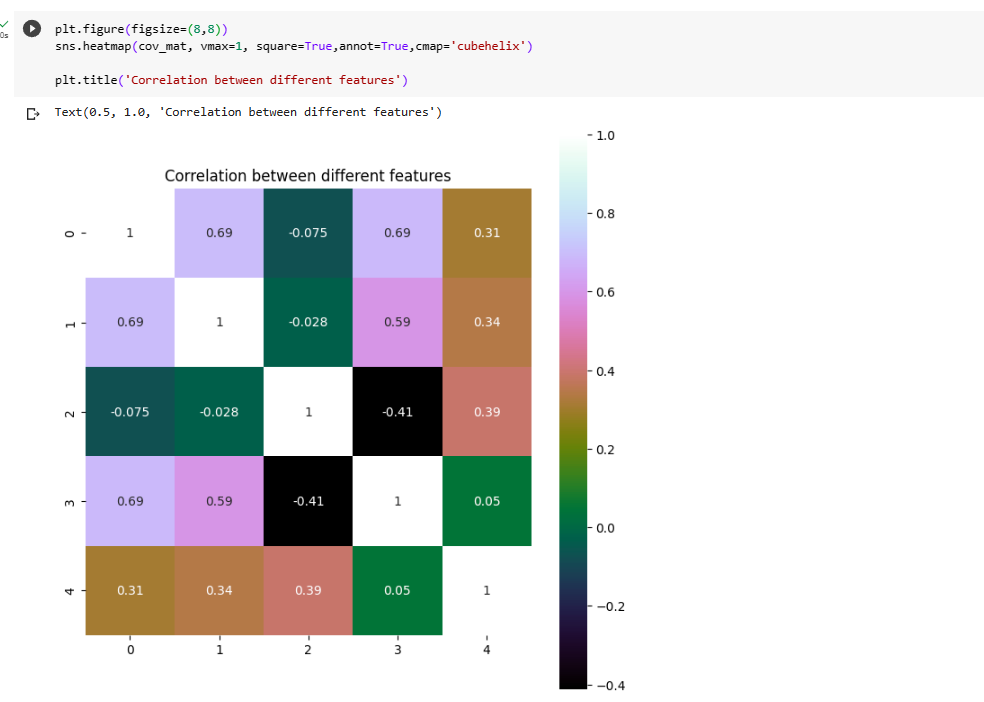


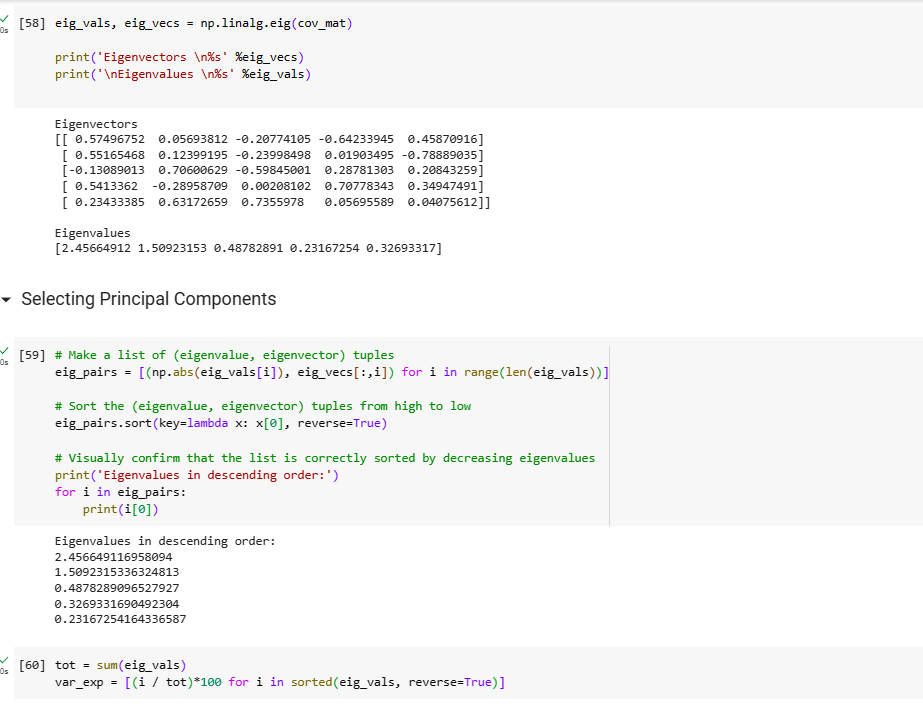
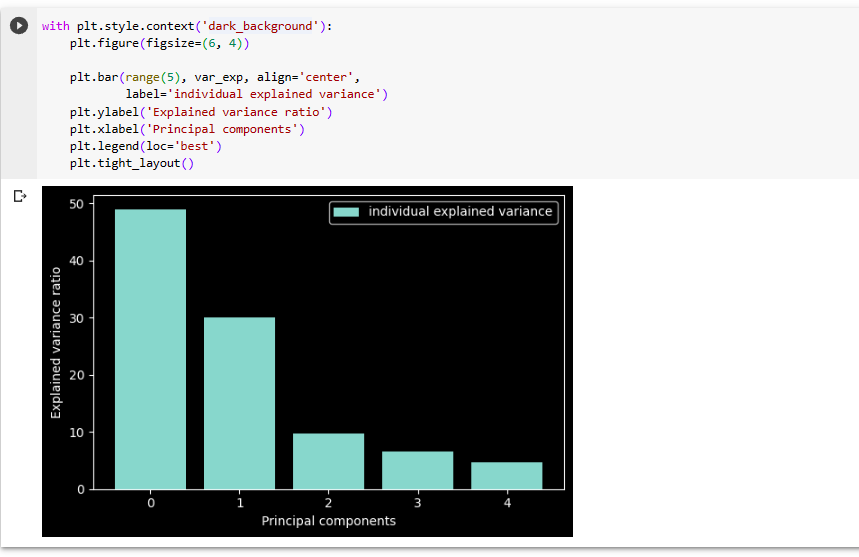
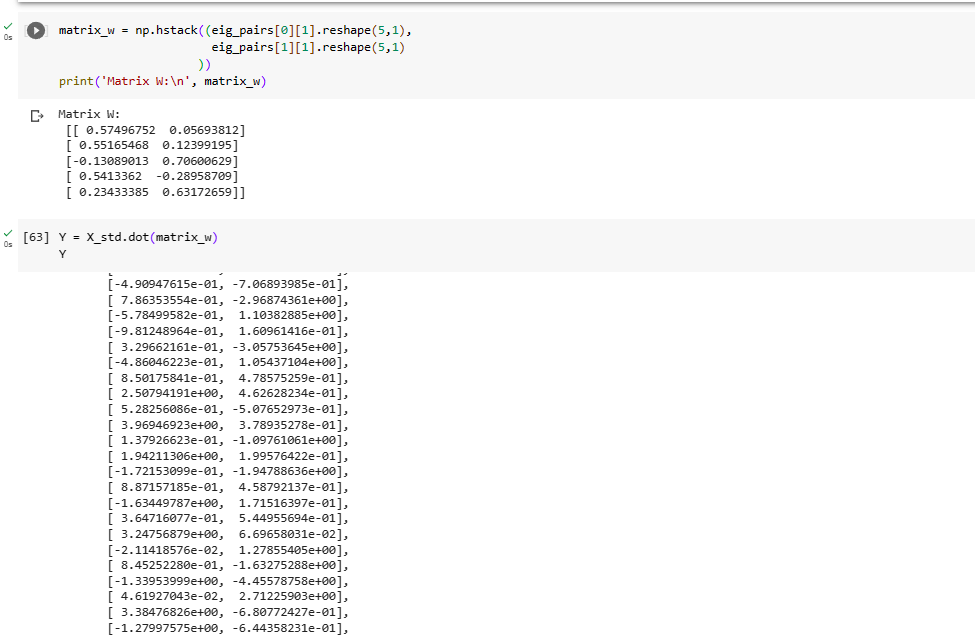


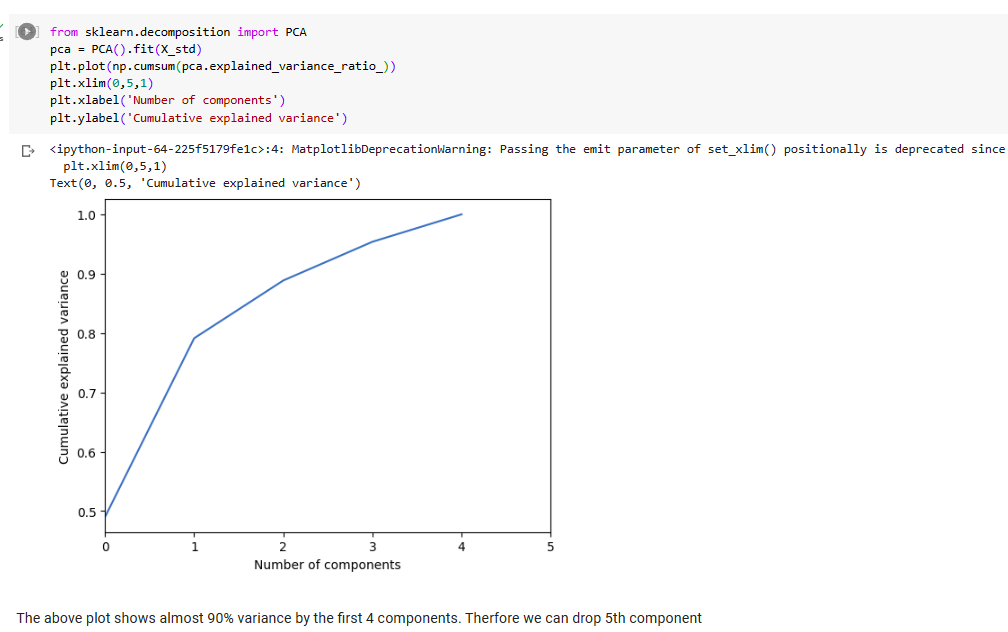


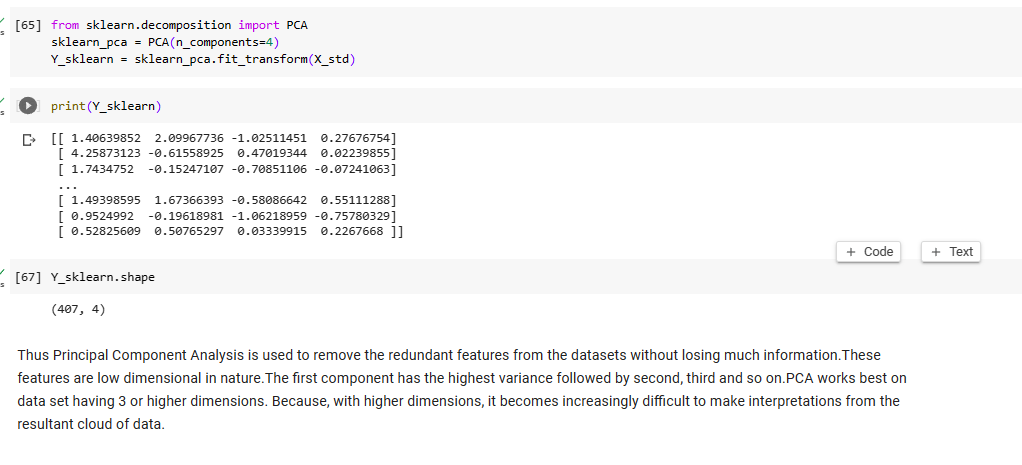








**CONCLUSION:**

The predicted output is displayed using the Principal Component Analysis model trained with the given dataset and results are verified. Thus the PCA is used to reduce the dimension of the dataset.

**Ex No: 5**

**FACTOR ANALYSIS**

**AIM**:

To perform Factor analysis using Mobile dataset.

**Dataset Description:**

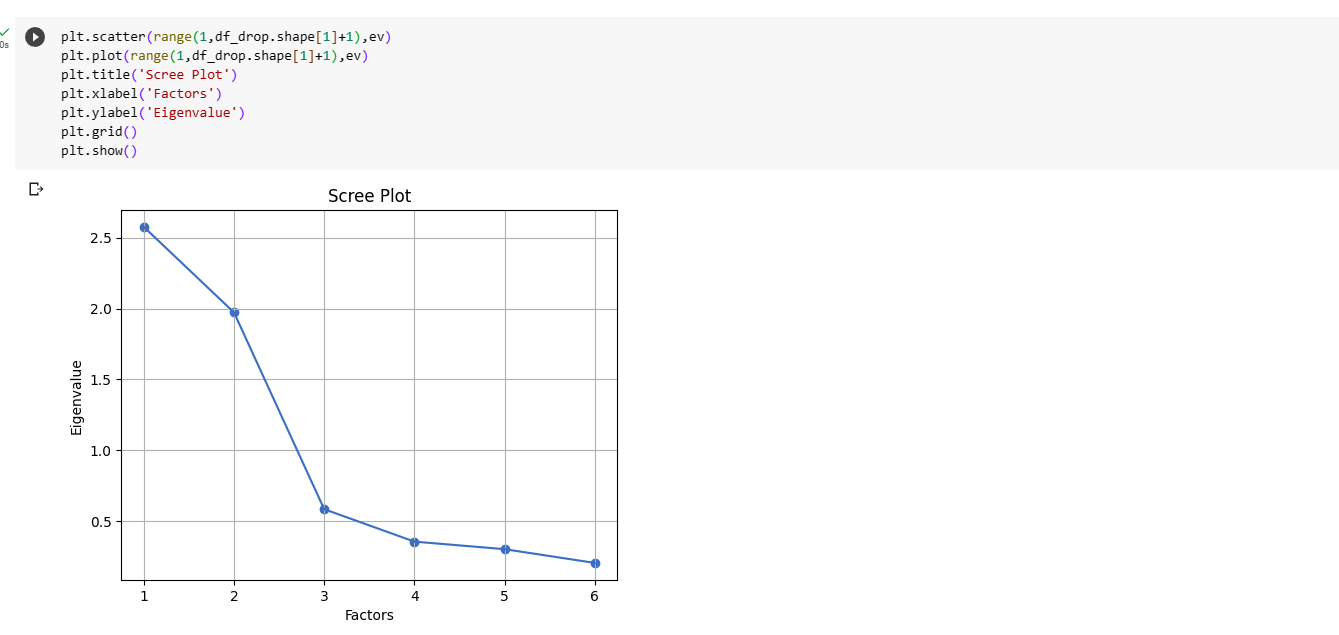
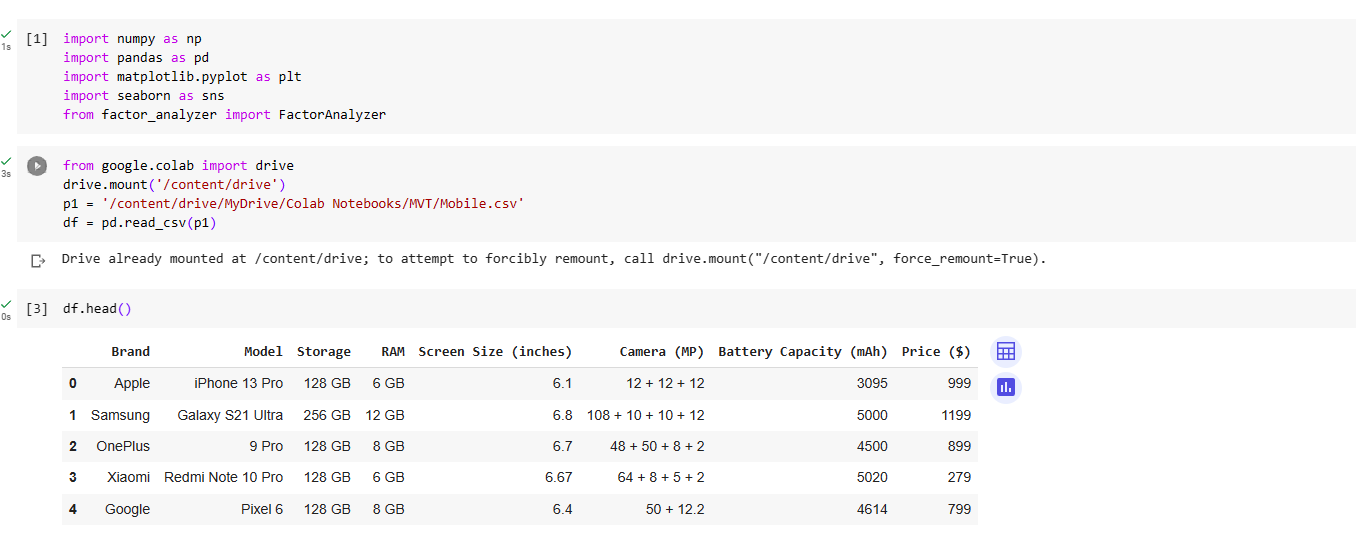
The dataset contains information on over 2,000 mobile phones from different brands. It includes details such as the storage capacity, RAM, screen size, camera specifications, battery capacity, and price of each device.

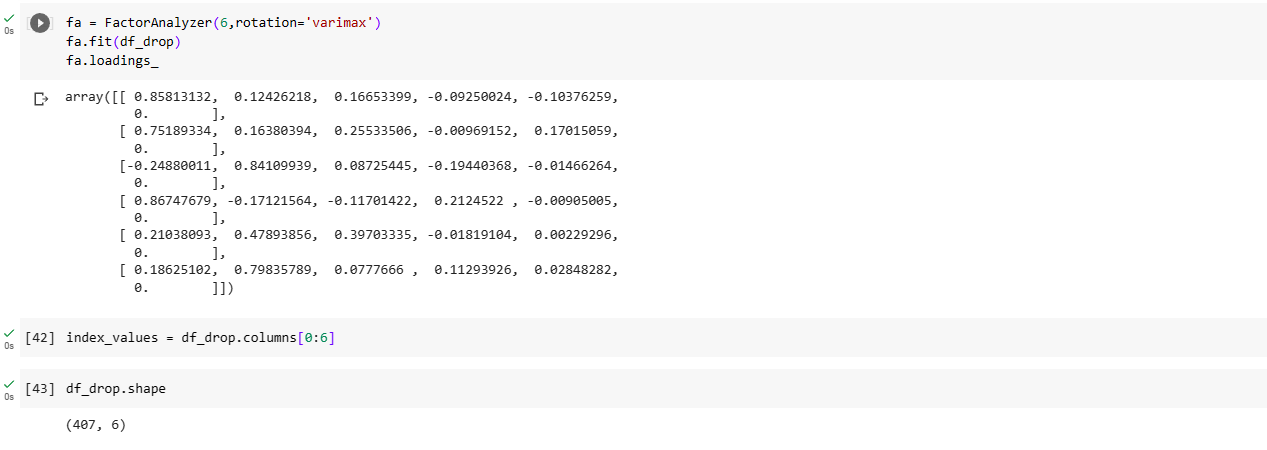
The dataset is structured as a CSV file with 7 columns:

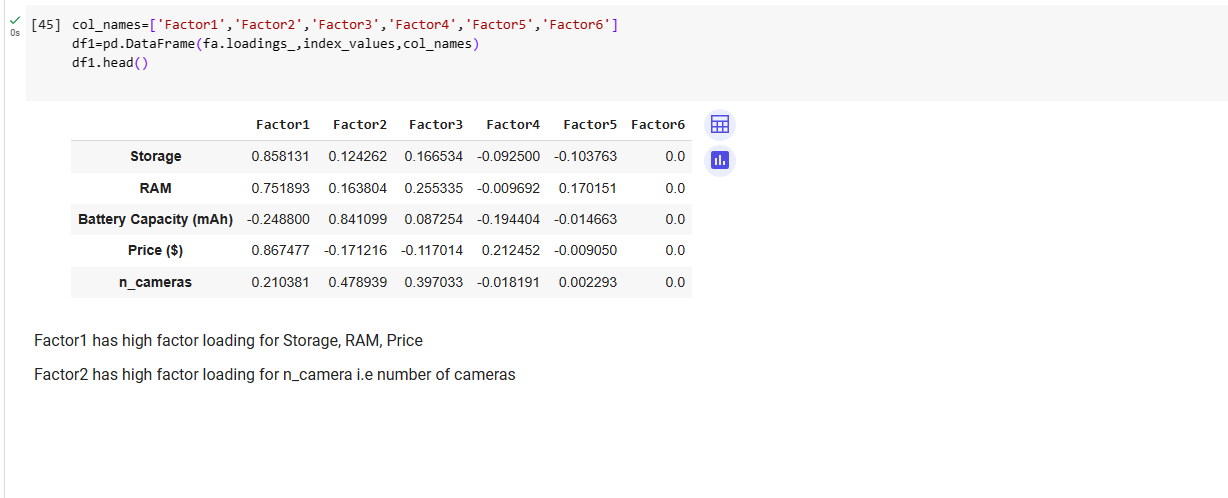
* Brand: The brand name of the mobile phone.
* Model: The model name of the mobile phone.
* Storage: The amount of storage space available on the mobile phone in GB.
* RAM: The amount of random access memory available on the mobile phone in GB.
* Screen Size: The size of the mobile phone's screen in inches.
* Camera: The quality of the mobile phone's cameras, measured in megapixels.
* Battery Capacity: The amount of battery life the mobile phone has in mAh.
* Price: The price of the mobile phone in USD.

**PROCEDURE:**

1. Import the necessary library functions.
2. Load the required dataset into the dataframe. (Dataset used : Personality data)
3. Print the head and shape of the dataset to find the dimensions of the given data.
4. Load the training dataset and fit the data into the Factor analysis model.
5. Display the scatterplot for the eigenvalues.
6. Load the test dataset and predict the value using the model
7. Display the results of the output predicted from the model.

**PROGRAM AND OUTPUT:**





**CONCLUSION:**

The predicted output is displayed using the Factor Analysis model trained with the given dataset and results are verified. Thus the Factor Analysis is used to reduce the dimension of  the dataset by selecting the most important factors in the dataset based on the latent features.

**Ex No: 6**

**LINEAR PROGRAMMING**

**AIM:**

To perform Linear programming in python for the given equations with the constraints and get the optimized values.

**PROCEDURE:**

1. Import the necessary library functions.
2. If pulp is not available use pip install method and install pulp library and import the entire package
3. Give the required constraints and maximization function to the model
4. View the model constraints and verify it.
5. Solve the equations using PULP\_CBC\_CMD ()
6. View the status of the model
7. Print the results which are calculated by the model
8. Get the optimized values of the given equation and constraints.

**PROGRAM:**

!pip install pulp

from pulp import \*

import pandas as pd

import numpy as np

# Create a LP Maximization problem

# LpProblem - Function

# LpMaximize - Objective function is to Maximize

model = LpProblem("Problem", LpMaximize)

# Create problem Variables

x = pulp.LpVariable("x", lowBound = 0) # Create a variable x >= 0

y = pulp.LpVariable("y", lowBound = 0) # Create a variable y >= 0

# Objective Function

model += 2 \* x + y

# Constraints:

model += (3 \* x + 2 \* y <= 12,"Constraint 1")

model += (x + 2.3 \* y <= 6.9,"Constraint 2")

model += (x + 1.4 \* y <= 4.9,"Constraint 3")

# Display the problem

print(model)

# Model.solve()

model.solve(PULP\_CBC\_CMD())

status = LpStatus[model.status]

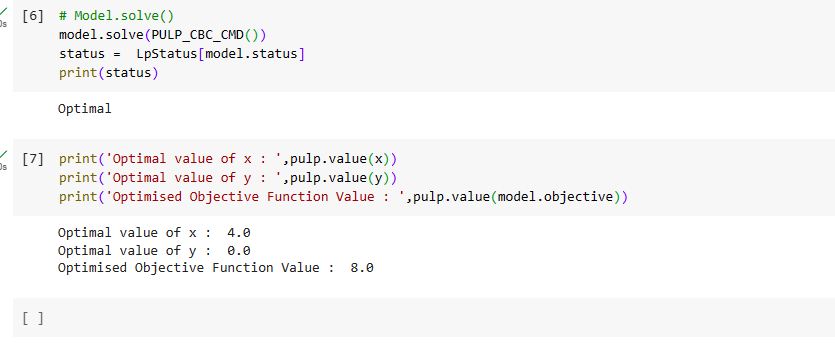
print(status)

print('Optimal value of x : ',pulp.value(x))

print('Optimal value of y : ',pulp.value(y))

print('Optimised Objective Function Value : ',pulp.value(model.objective))

**OUTPUT**



**CONCLUSION:**

Thus the Linear programming method using python was implemented and the results of various equations and optimized values was verified successfully.

**Ex No: 07**

**TRANSPORTATION PROBLEM**

**AIM:**

To perform Transportation problem in python for the given equations with the constraints and get the optimized values.

**PROCEDURE:**

1. Import the necessary library functions.
2. If pulp is not available use pip install method and install pulp library and import the entire package
3. Give the required constraints and minimization function to the model
4. View the model constraints and verify it.
5. Solve the equations using PULP\_CBC\_CMD()
6. View the status of the model
7. Print the results which are calculated by the model
8. Get the optimized values of the given equation and constraints.

**PROGRAM:**

!pip install pulp

from pulp import \*

# Creates a list of all the supply nodes

supply\_nodes = ["S1","S2","S3"]

# Creates a dictionary for the number of units of supply for each supply node

supply = {"S1": 11,

      "S2": 13,

      "S3": 19}

# Creates a list of all demand nodes

demand\_nodes = ["D1","D2","D3" ,"D4"]

# Creates a dictionary for the number of units of demand for each demand node

demand = {"D1": 6,

      "D2": 10,

      "D3": 12,

      "D4":15}

# Creates a list of costs of each transportation path

costs = [# Demand

     #D1 D2 D3 D4

     [21,16,25,13], #S1

     [17,18,14,23], #S2  Supply

     [32,27,18,41]  #S3

     ]

costs = makeDict((supply\_nodes, demand\_nodes),costs)

print(costs)

# Creates the prob variable to contain the problem data

prob = LpProblem("Product Distribution Problem",LpMinimize)

# Creates a list of tuples containing all the possible routes for transport

Routes = [(s,d) for s in supply\_nodes for d in demand\_nodes]

# A dictionary called route\_vars is created to contain the referenced variables (the routes)

route\_vars = LpVariable.dicts("Route",(supply\_nodes,demand\_nodes),0,None,LpInteger)

# The objective function is added to prob first

prob += lpSum([route\_vars[s][d]\*costs[s][d] for (s,d) in Routes]), "Sum of Transporting Costs"

# The supply maximum constraints are added to prob for each supply node (warehouse)

for s in supply\_nodes:

    prob += lpSum([route\_vars[s][d] for d in demand\_nodes]) <= supply[s], "Sum of Products out of supply %s"%s

# The demand minimum constraints are added to prob for each demand node (bar)

for d in demand\_nodes:

    prob += lpSum([route\_vars[s][d] for s in supply\_nodes]) >= demand[d], "Sum of Products into demand %s"%d

prob.solve()

print("Status:",LpStatus[prob.status] )

total=0

for v in prob.variables():

    if(v.varValue != None):

        if (v.varValue > 0):

            print(v.name, "=", v.varValue)

            total+=v.varValue

total=0

for v in prob.variables():

    if(v.varValue != None):

        if (v.varValue > 0):

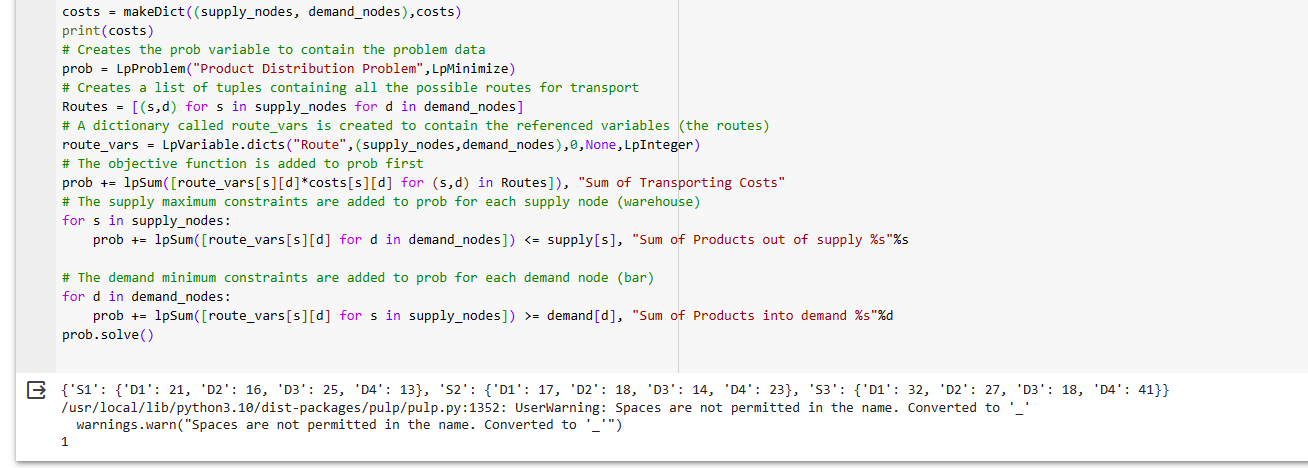
            print(v.name, "=", v.varValue)

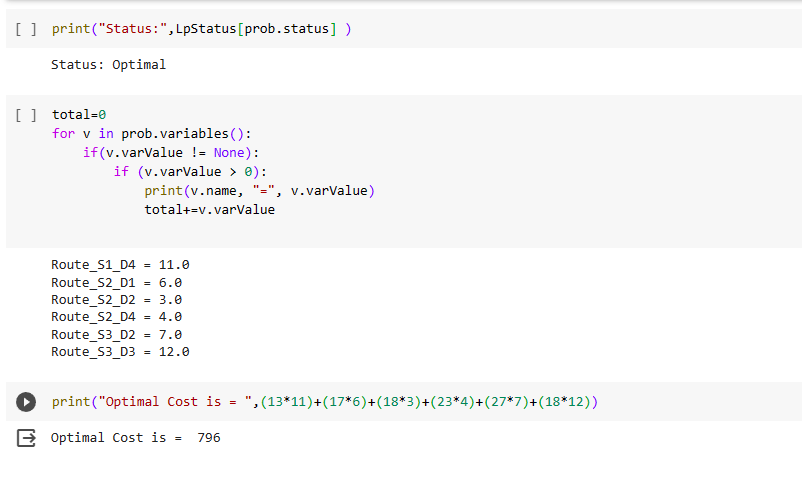
            total+=v.varValue

print("Optimal Cost is = ",(13\*11)+(17\*6)+(18\*3)+(23\*4)+(27\*7)+(18\*12))

**OUTPUT:**







**CONCLUSION:**

Thus the transportation problem method using python was implemented and the results of various equations and optimized values was verified successfully.

**Ex No: 08**

**ASSIGNMENT PROBLEM**

**AIM:**

To perform assignment problem in python for the given cost matrix and get the optimized cost.

**PROCEDURE:**

1. Import the necessary library functions.
2. If scipy is not available use pip install method and install scipy library and import the entire package
3. Create a cost matrix and pass it to function to solve it
4. Solve the equations using linear\_sum\_assignment ()
5. Extract the optimal assignment values
6. Print the optimal assignment values
7. Calculate the total cost

**PROGRAM:**

import numpy as np

from scipy.optimize import linear\_sum\_assignment

# Create a cost matrix

cost\_matrix = np.array([

[10, 11, 4, 2, 8],

[ 7, 11, 10, 14, 12],

[ 5, 6, 9, 12, 14],

[ 13, 15, 11, 10, 7]

])

# Solve the assignment problem

row\_indices, col\_indices = linear\_sum\_assignment(cost\_matrix)

# Extract the optimal assignment

assignment = [(row, col) for row, col in zip(row\_indices, col\_indices)]

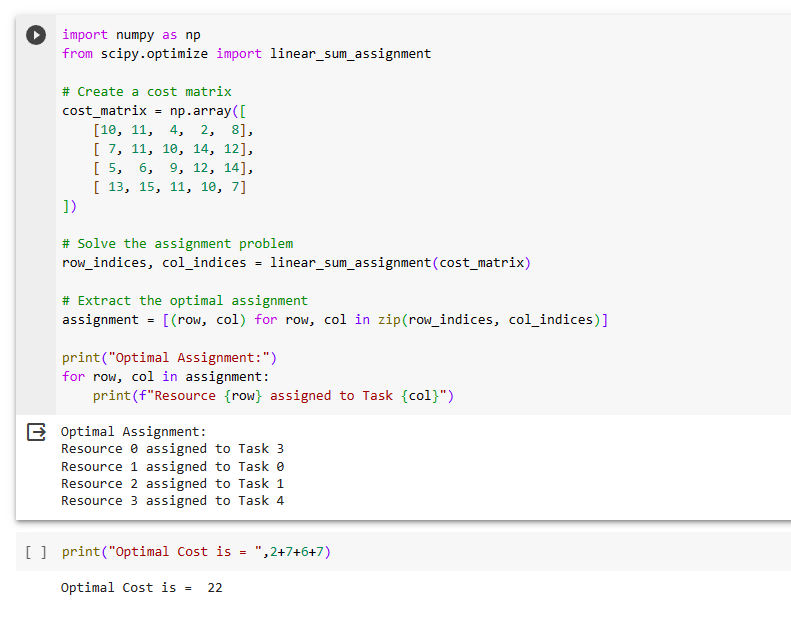
print("Optimal Assignment:")

for row, col in assignment:

print(f"Resource {row} assigned to Task {col}")

print("Optimal Cost is = ",2+7+6+7)

**OUTPUT:**



**CONCLUSION:**

Thus the assignment problem method using python was implemented and the results of cost matrix was verified successfully.

**Ex No: 9**

**HIERARCHICAL CLUSTERING**

**AIM**:

To perform Hierarchical Clustering using agglomerative clustering with four types of linkage methods ward ,single ,average ,complete using sklearn make blobs dataset.

**Dataset Description:**

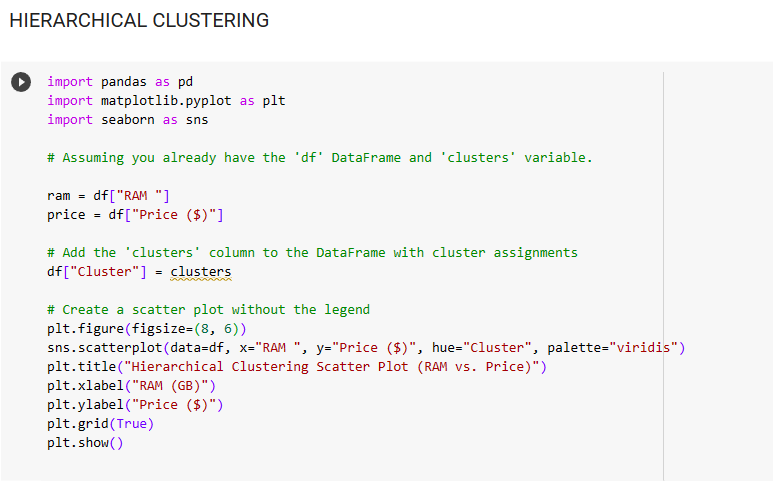
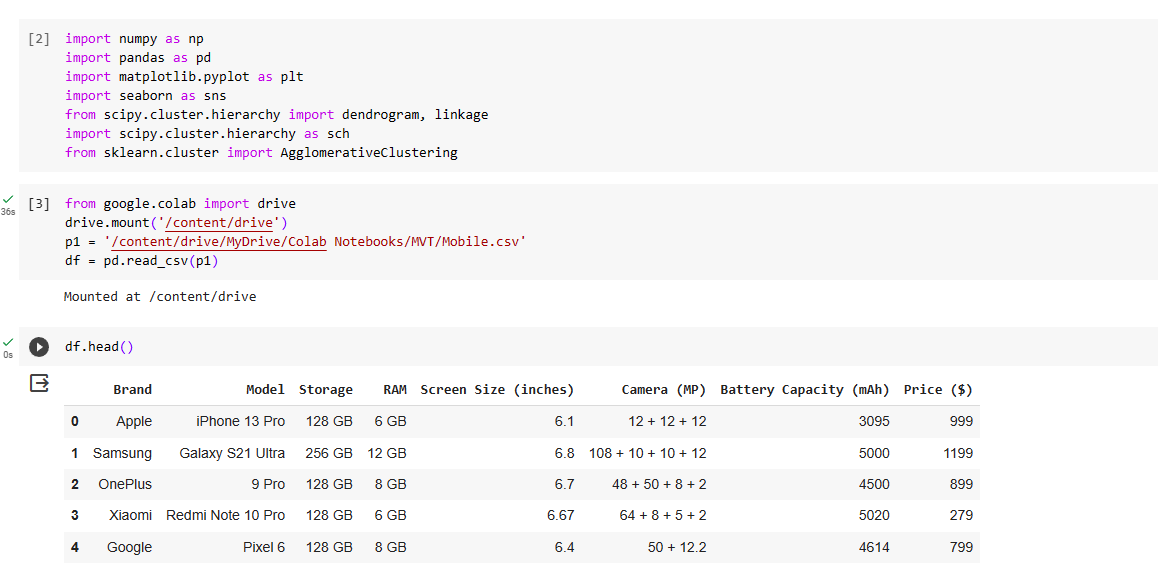
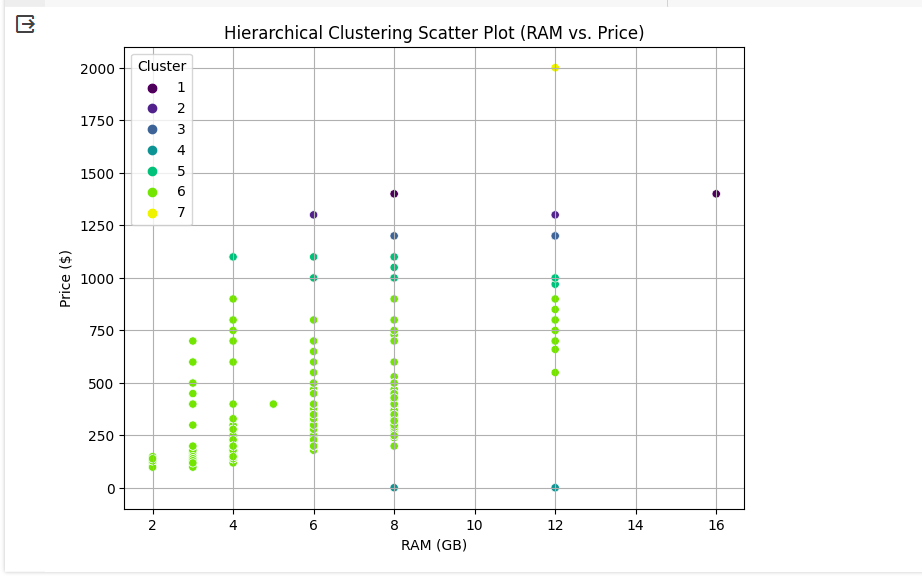
The dataset contains information on over 2,000 mobile phones from different brands. It includes details such as the storage capacity, RAM, screen size, camera specifications, battery capacity, and price of each device.

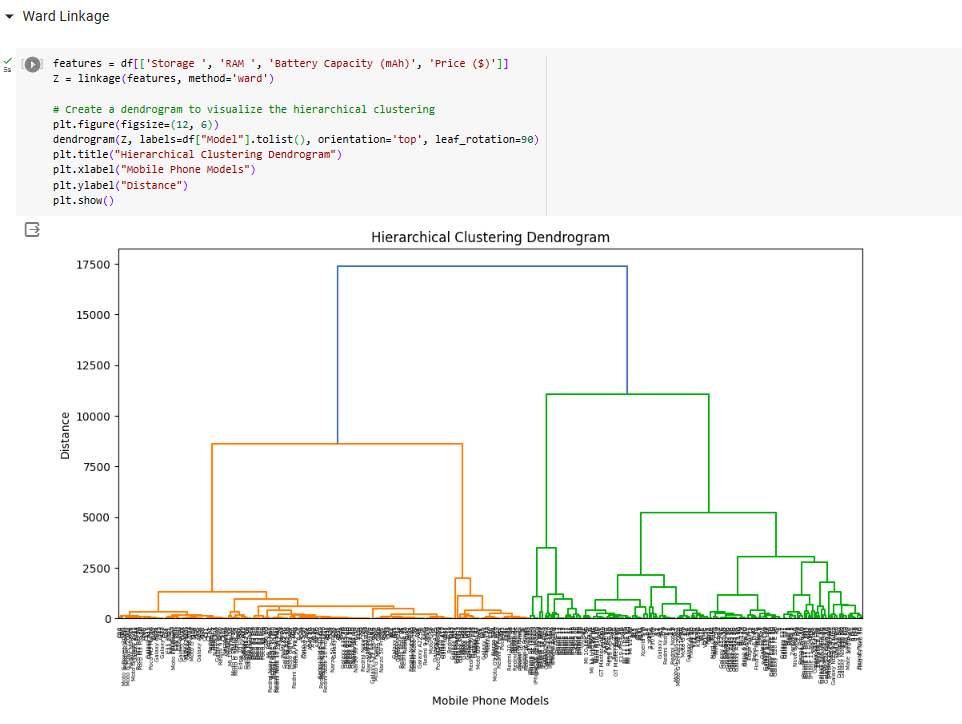
The dataset is structured as a CSV file with 7 columns:

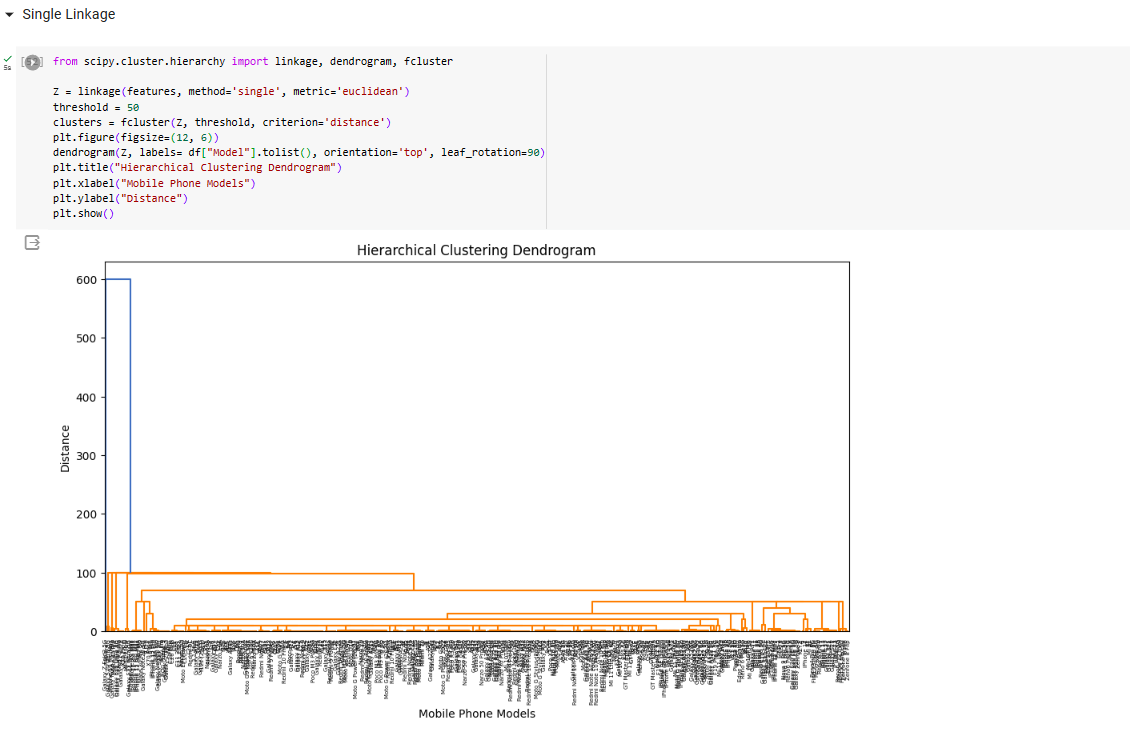
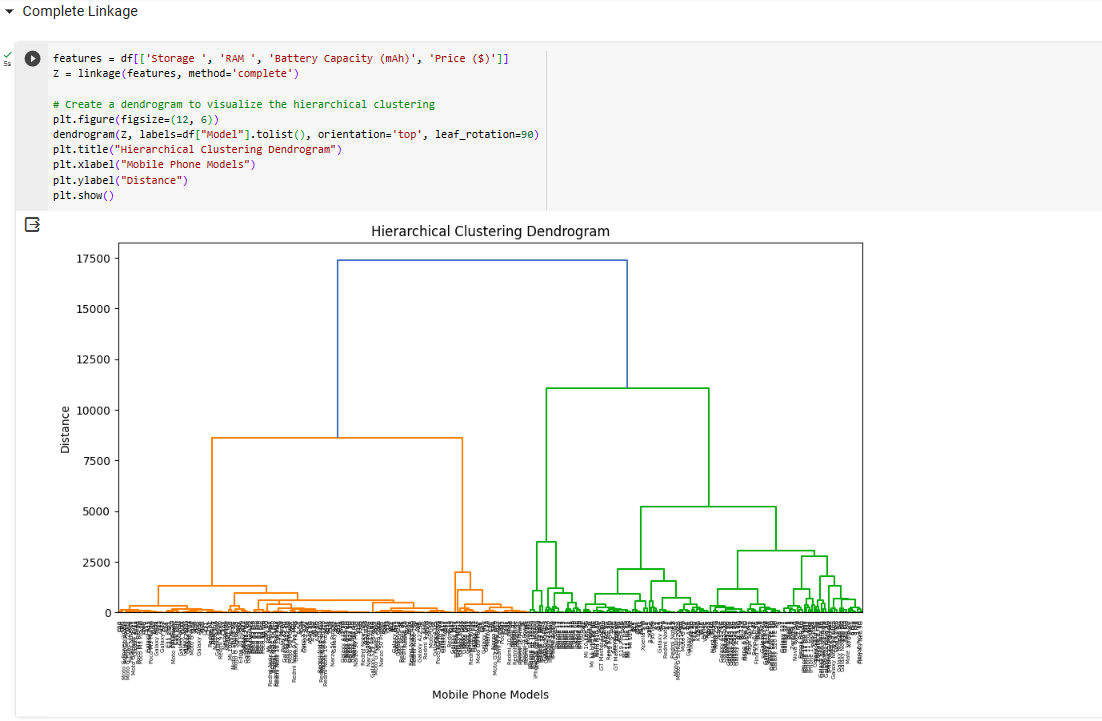
* Brand: The brand name of the mobile phone.
* Model: The model name of the mobile phone.
* Storage: The amount of storage space available on the mobile phone in GB.
* RAM: The amount of random access memory available on the mobile phone in GB.
* Screen Size: The size of the mobile phone's screen in inches.
* Camera: The quality of the mobile phone's cameras, measured in megapixels.
* Battery Capacity: The amount of battery life the mobile phone has in mAh.
* Price: The price of the mobile phone in USD.

**PROCEDURE:**

1. Import the necessary library functions.
2. Load the required dataset into the dataframe. (Dataset used : sklearn inbuilt makeblob dataset)
3. Load the training dataset and fit the data into the hierarchical clustering ,agglomerative clustering model.
4. Display the scatterplot for the two columns.
5. Display the dendogram using agglomerative model.
6. Fit and predict the point in agglomerative clustering model.
7. Display the scatter plot of the clusters.

**PROGRAM AND OUTPUT;** 







**CONCLUSION:**

Thus the given dataset is clustered using the hierarchical clustering with agglomerative clustering method with 4 different types of linkage methods called as wards, single, complete ,average linkage methods and results were verified.

**Ex No: 10**

**K-MEANS CLUSTERING**

**AIM**:

To perform Non-hierarchical clustering using K-Means algorithm using given dataset.

**Dataset Description:**

The dataset contains information on over 2,000 mobile phones from different brands. It includes details such as the storage capacity, RAM, screen size, camera specifications, battery capacity, and price of each device.

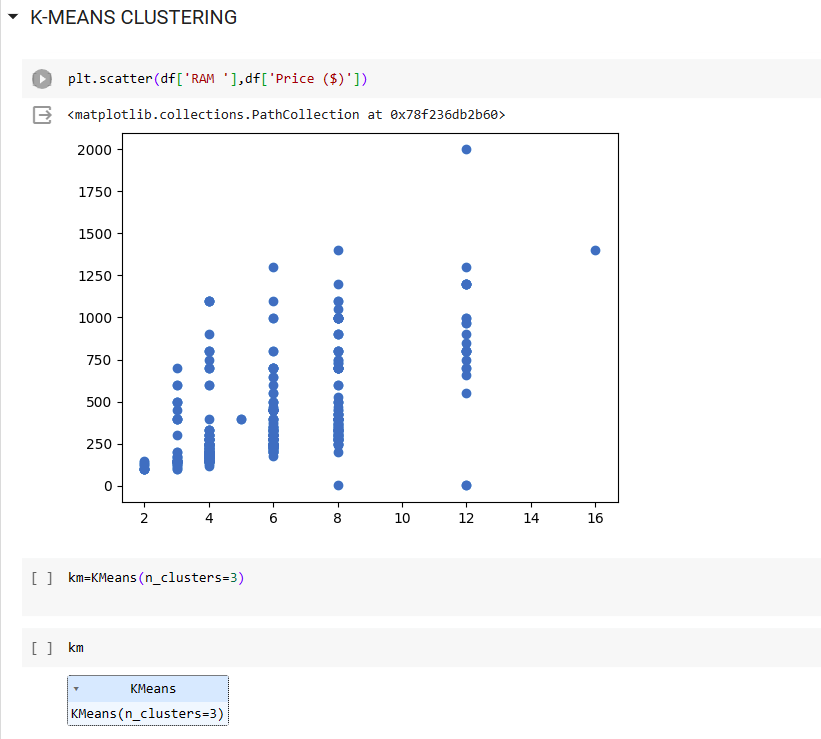
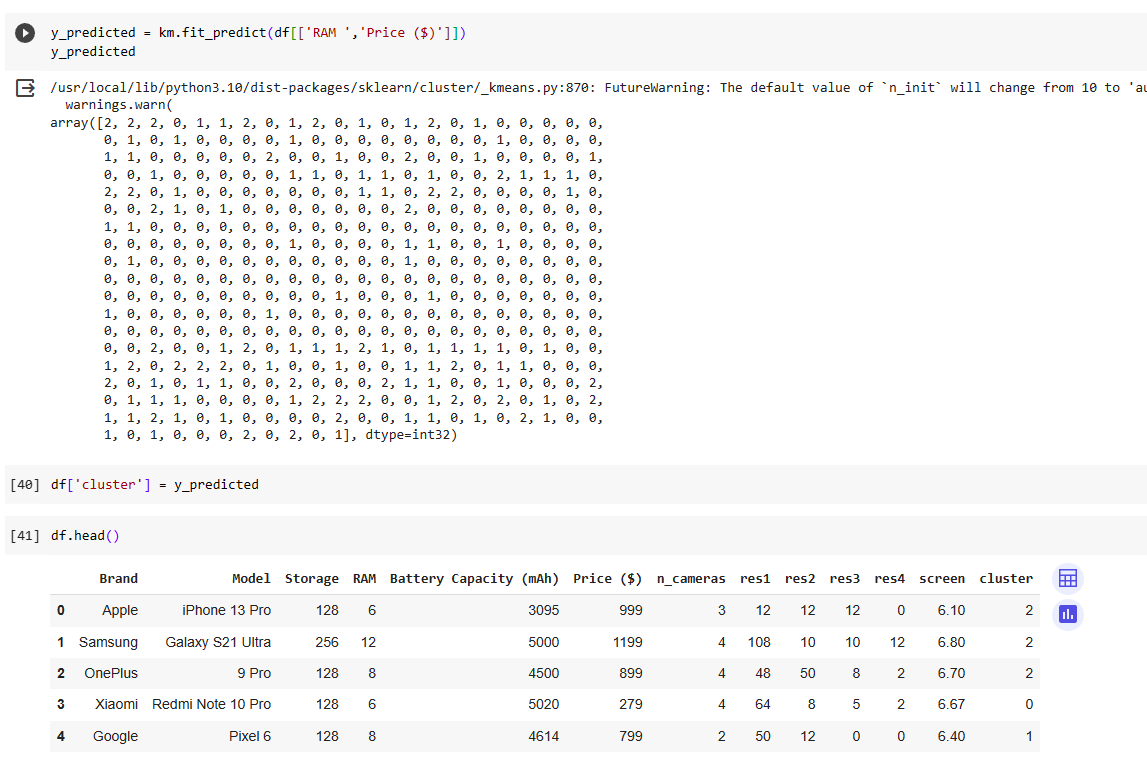
The dataset is structured as a CSV file with 7 columns:

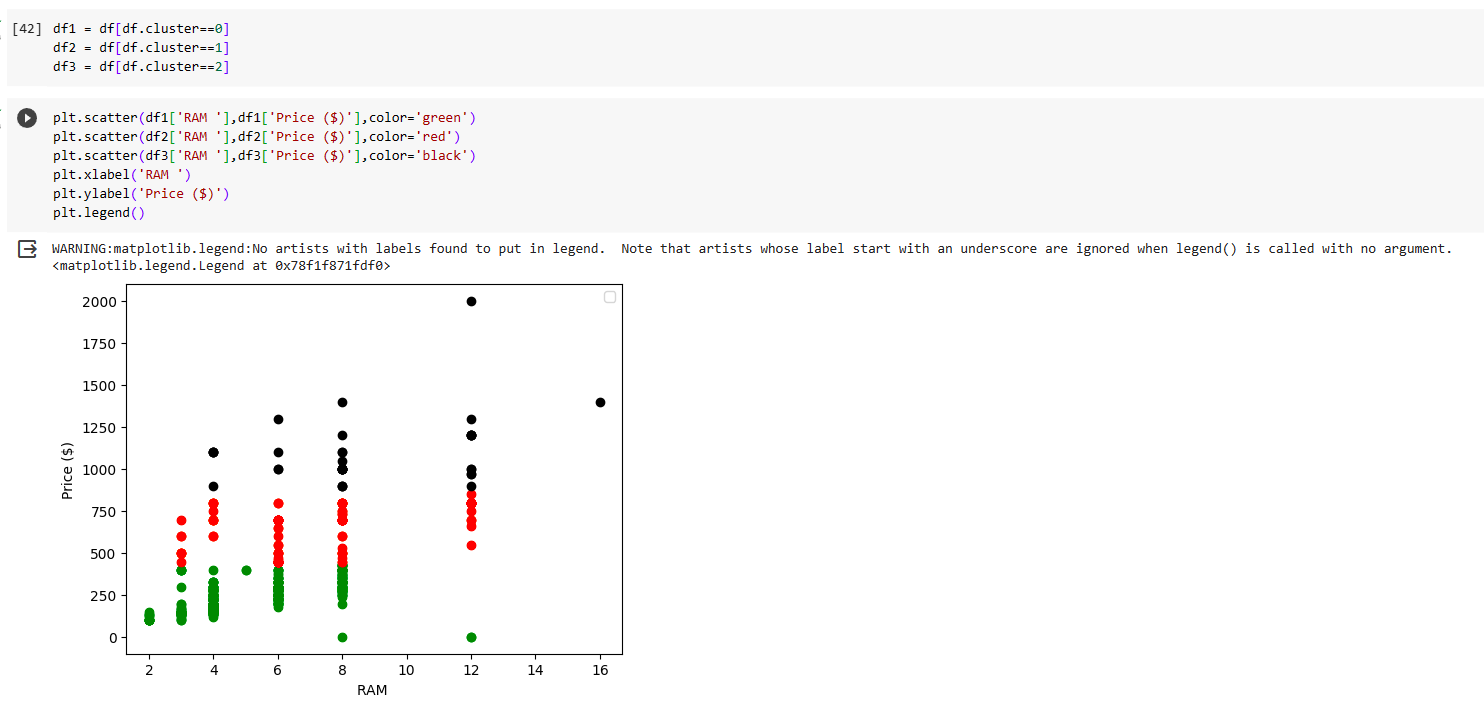
* Brand: The brand name of the mobile phone.
* Model: The model name of the mobile phone.
* Storage: The amount of storage space available on the mobile phone in GB.
* RAM: The amount of random access memory available on the mobile phone in GB.
* Screen Size: The size of the mobile phone's screen in inches.
* Camera: The quality of the mobile phone's cameras, measured in megapixels.
* Battery Capacity: The amount of battery life the mobile phone has in mAh.
* Price: The price of the mobile phone in USD.

**PROCEDURE:**

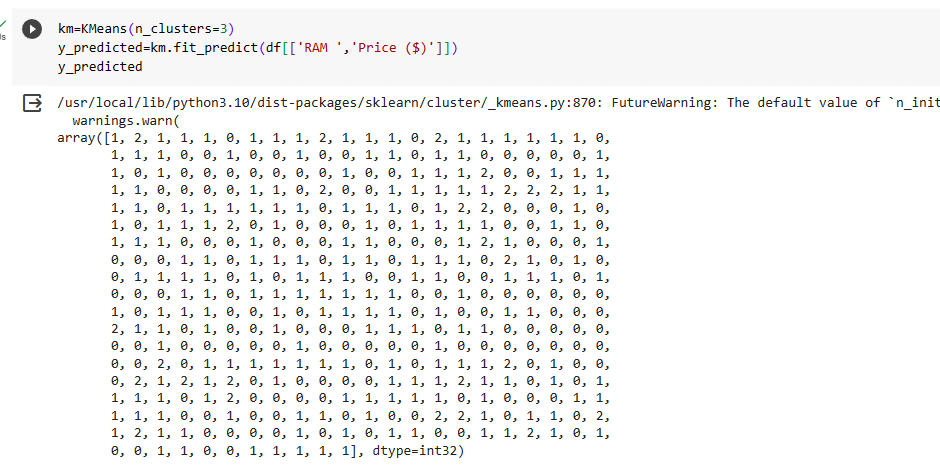
1. Import the necessary library functions.
2. Load the required dataset into the dataframe. (Dataset used : Iris dataset)
3. Print the head and shape of the dataset to find the dimensions of the given data.
4. Load the training dataset and fit the data into the K-Means clustering model.
5. Display the scatterplot for the two columns.
6. Using min-maxscaler find the number of cluster required and plot the graph.
7. With the help of the elbow diagram , find the number of clusters needed and do the k-means clustering.

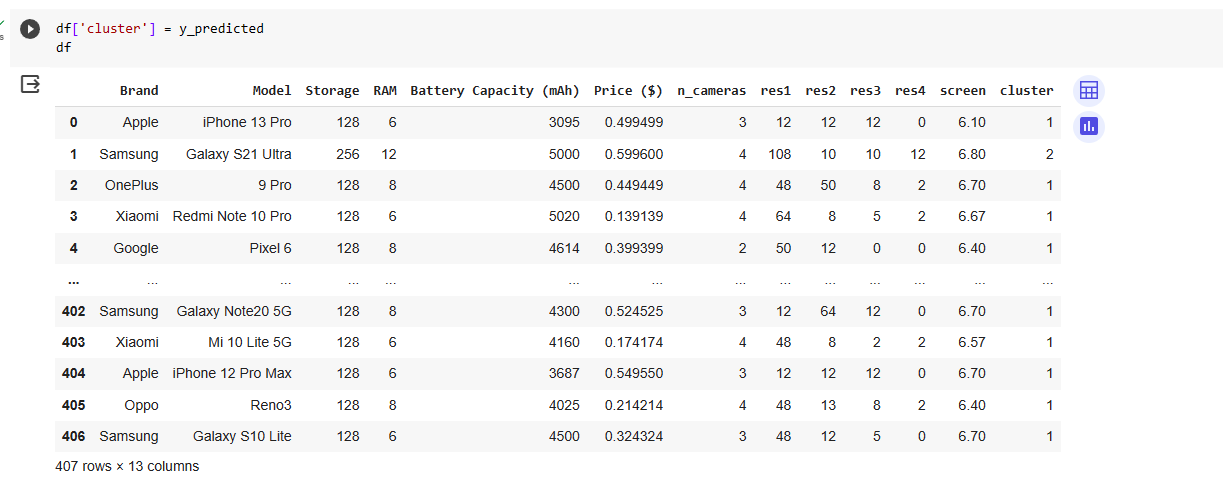
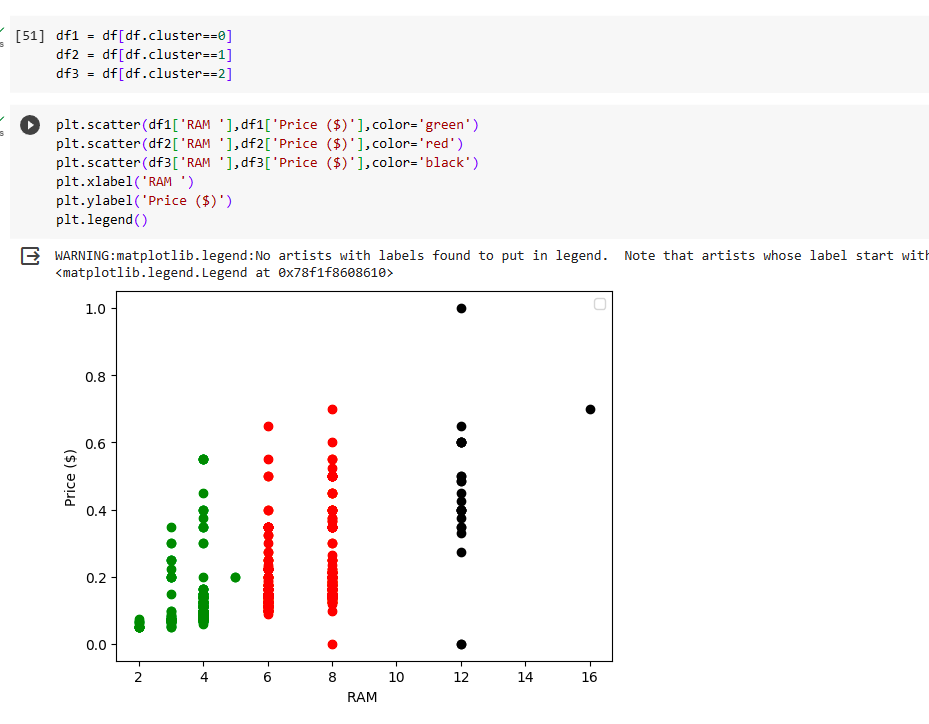
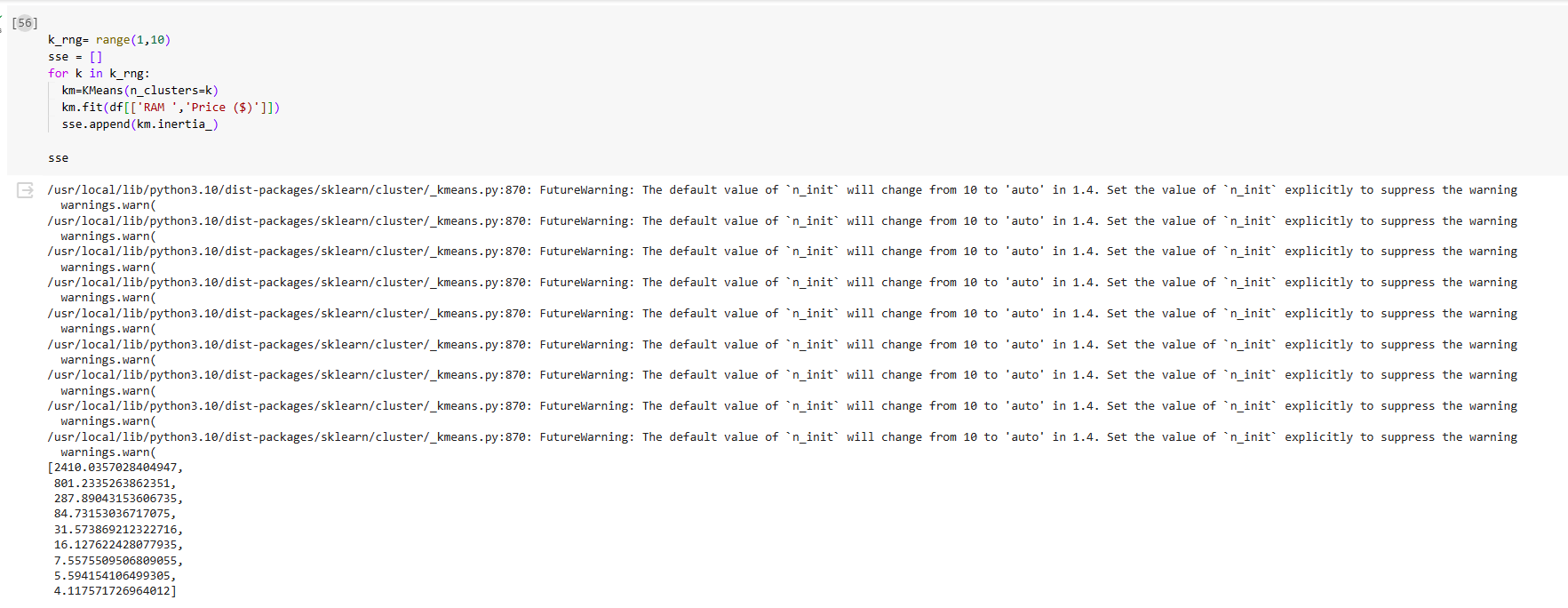
**PROGRAM AND OUTPUT:**

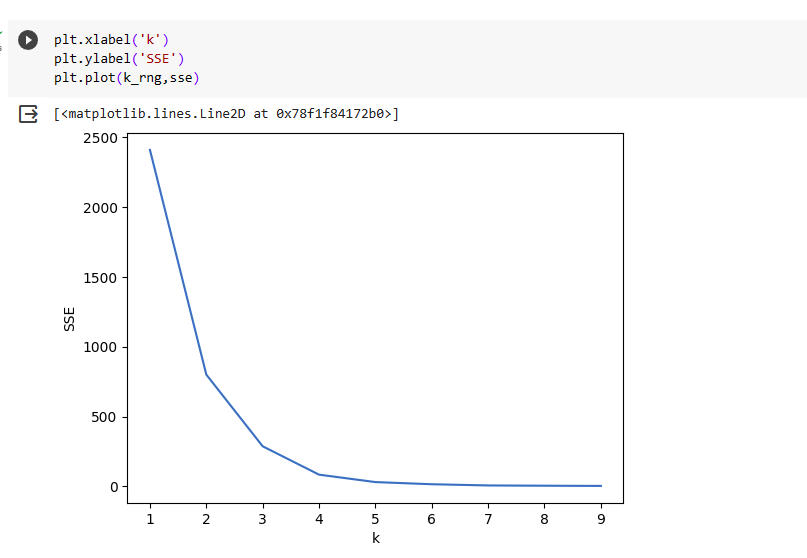










**CONCLUSION:**

Thus the given dataset is clustered using k-means clustering algorithm and 4 clusters has been grouped