EXPRIMENT :- 6

To perform dimensionality reduction operation using PCA for Houses Data Set

Solution:

```
from sklearn.datasets import fetch_california_housing
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
import pandas as pd

# Load California Housing Dataset
housing = fetch_california_housing(as_frame=True)
data = housing.data

# Standardize the data
scaled_data = StandardScaler().fit_transform(data)

# Apply PCA to retain 95% variance
pca = PCA(n_components=0.95)
principal_components = pca.fit_transform(scaled_data)

# Explained variance ratio
print("Explained Variance Ratio:", pca.explained_variance_ratio_)
print("Number of Components:", pca.n_components_)
```

Output:

```
Explained Variance Ratio: [0.25336868 0.23516245 0.15888635 0.12887971 0.12538195 0.0824225 ]
Number of Components: 6
```

EXPRIMENT :- 7

To perform Simple Linear Regression with R.

Solution:

```r
# Load dataset
data(mtcars)  # Built-in dataset in R

# View the first few rows
head(mtcars)

# Perform Simple Linear Regression
# Predicting mpg (miles per gallon) based on wt (weight of the car)
model <- lm(mpg ~ wt, data = mtcars)

# Summary of the model
summary(model)

# Plot the data and regression line
plot(mtcars$wt, mtcars$mpg,
     main = "Simple Linear Regression: MPG vs Weight",
     xlab = "Car Weight (1000 lbs)",
     ylab = "Miles Per Gallon",
     pch = 19, col = "blue")

# Add regression line
abline(model, col = "red", lwd = 2)
```

Output:

```
                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

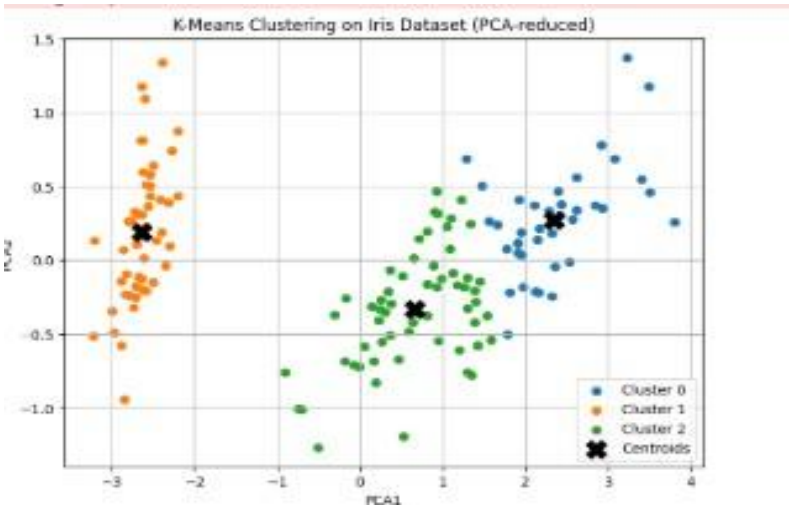
EXPRIMENT :- 8:

To perform K-Means clustering operation and visualize for iris data set

Solution:

```
import pandas as pd
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA

# Load the Iris dataset
iris = load_iris()
data = pd.DataFrame(iris.data, columns=iris.feature_names)

# Apply K-Means clustering
kmeans = KMeans(n_clusters=3, random_state=42)
data['Cluster'] = kmeans.fit_predict(data)

# Reduce dimensions for visualization using PCA
pca = PCA(n_components=2)
reduced_data = pca.fit_transform(data.iloc[:, :-1])

# Add PCA results to the dataframe
data['PCA1'] = reduced_data[:, 0]
data['PCA2'] = reduced_data[:, 1]

# Visualize clusters
plt.figure(figsize=(8, 6))
for cluster in range(3):
    cluster_data = data[data['Cluster'] == cluster]
    plt.scatter(cluster_data['PCA1'], cluster_data['PCA2'], label=f'Cluster {cluster}')

# Add cluster centers (PCA-reduced)
centers = pca.transform(kmeans.cluster_centers_)
plt.scatter(centers[:, 0], centers[:, 1], s=200, c='black', marker='X', label='Centroids')

plt.title('K-Means Clustering on Iris Dataset (PCA-reduced)')
plt.xlabel('PCA1')
plt.ylabel('PCA2')
plt.legend()
plt.grid()
plt.show()
```

Output:



EXPRIMENT :- 9:

Learn how to collect data via web-scraping, APIs and data connectors from suitable sources as specified by the instructor

Solution:

1)-Web Scrapping:

```
import requests
from bs4 import BeautifulSoup

# URL of the website to scrape
url = 'https://example.com'

# Send a GET request to fetch the HTML content
response = requests.get(url)

# Parse the HTML content
soup = BeautifulSoup(response.content, 'html.parser')

# Extract specific data (e.g., all paragraph texts)
paragraphs = soup.find_all('p')
for p in paragraphs:
    print(p.text)
```

2)-APIs:

```
import requests

# TMDB API Key
api_key = 'your_api_key'
url = f'https://api.themoviedb.org/3/movie/popular?api_key={api_key}'

# Send a GET request to fetch the data
response = requests.get(url)
data = response.json()

# Print the movie titles
for movie in data['results']:
    print(movie['title'])
```

3)-Data Connectors:

```
import mysql.connector

# Connect to the database
connection = mysql.connector.connect(
    host='localhost',
    user='your_username',
    password='your_password',
    database='your_database'
)

# Query the data
cursor = connection.cursor()
cursor.execute('SELECT * FROM your_table')

# Fetch and display results
for row in cursor.fetchall():
    print(row)

# Close connection
cursor.close()
connection.close()
```