EXPLORATORY DATA ANALYSIS THROUGH PRINCIPAL COMPONENT ANALYSIS

MATRIX COMPUTATIONS FOR DATA ENGINEERING

DS502

AY 2023-24

Center for Applied Research and Date Sciences
Indian Institute of Technology Ropar

Prepared By -

Kavya Agrawal (2023DSS1017)

Harsha Harod (2023DSS1015)

Presented To –

Dr. G S Raju

What is Principal Component Analysis (PCA)?

Principal Component Analysis (PCA) was introduced by mathematician Karl Pearson in 1901. It serves as a technique to decrease the dimensionality of a dataset by identifying a new set of variables, smaller than the original set, while preserving the majority of the sample's information. The fundamental principle underlying PCA is the maximization of data variance in a lower-dimensional space when mapping data from a higher-dimensional space.

PCA addresses several challenges in data analysis:

1. **The Curse of Dimensionality:**
    - With an increasing number of features or dimensions in a dataset, the amount of data required for statistically significant results grows exponentially. This leads to issues such as overfitting, longer computation times, and reduced accuracy in machine learning models.
2. **Computational Complexity:**
    - As the number of dimensions rises, the potential combinations of features increase exponentially, making it computationally challenging to obtain a representative sample of the data. This complexity hampers tasks like clustering or classification, becoming costly and resource-intensive.
3. **Sensitivity of Machine Learning Algorithms:**
    - Some machine learning algorithms are sensitive to the number of dimensions, demanding more data to achieve comparable accuracy to lower-dimensional data.

To mitigate the curse of dimensionality, various feature engineering techniques, including feature selection and feature extraction, are employed. Dimensionality reduction, a form of feature extraction, aims to decrease the number of input features while preserving as much original information as possible.

"Primary purpose of PCA is to reduce the number of features in a dataset while preserving as much of the variance or information as possible"

Some important use cases are in:

- Simplifying data visualisation
- Speed up ML algorithm
- Remove noise or redundancy in data

In this report we will be focusing how we can do Exploratory Data Analysis (EDA) with the help of Principal Component Analysis (PCA). EDA is the first most important step in any data science project.

Geometric Intuition of PCA

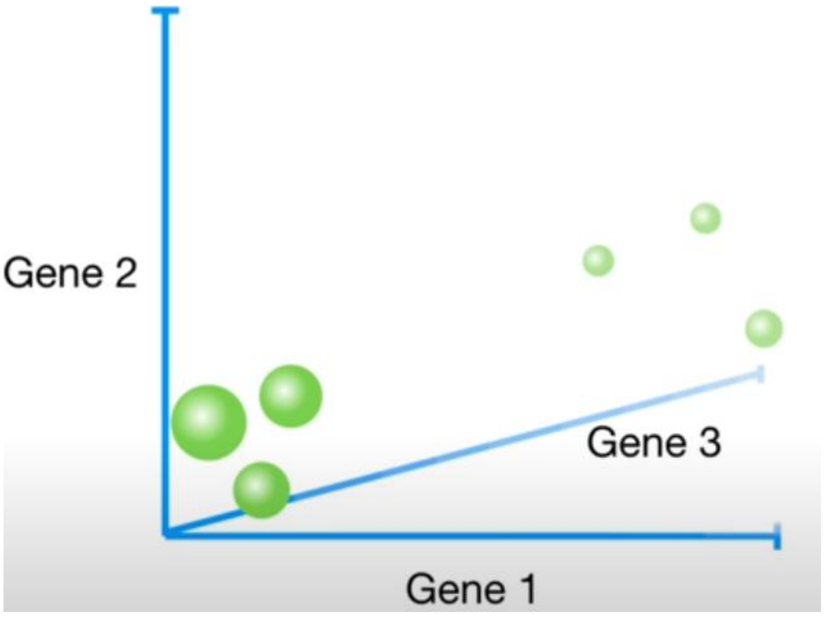
Here we will explain all the steps to perform Principal component analysis to find the principal components and construct a PC plot to better visualize the data.

Let us take a random dataset for the explanation:

	S1	**S2**	**S3**	**S4**	**S5**	**S6**
Gene 1	10	11	8	3	2	1
Gene 2	6	4	5	3	2.8	1
Gene 3	12	9	10	2.5	1.3	2

How can we visualize this data?

We can plot the graph for the above dataset with three axis:



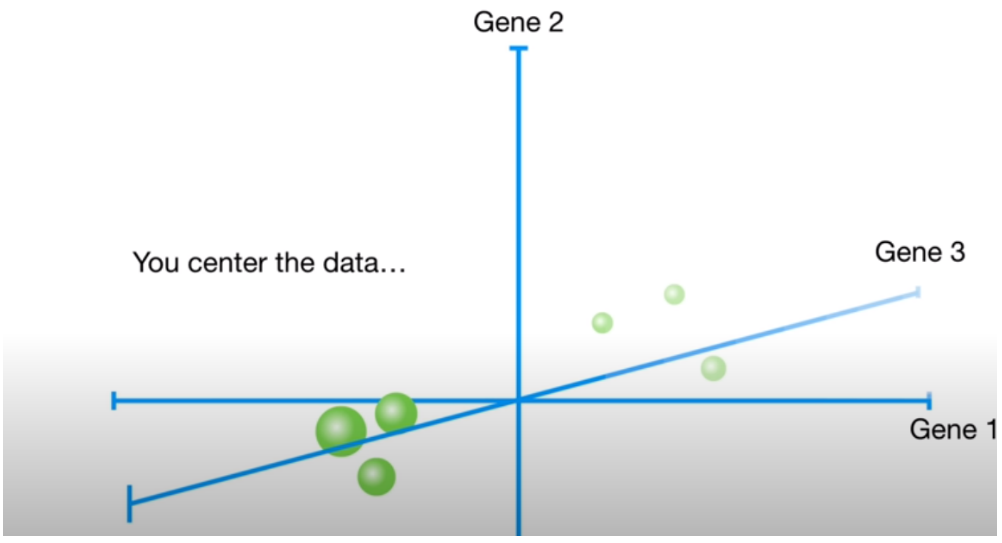
Step 1: Scaling and centring the data

Making sure that the variables are on the same scale. Plotting directly can give us the incorrect picture of our data. The standard practice is to dividing by its standard deviation.

Centring the data by subtracting the mean from each value so that the mean is zero and standard deviation is 1. We will observe that the centre of the data shifts towards the origin.

So, by centring the data, you ensure that PCA focuses on the essential patterns and variations in the data, not how they differ from the mean. This makes the analysis more meaningful and interpretable

The graph after centring and scaling:

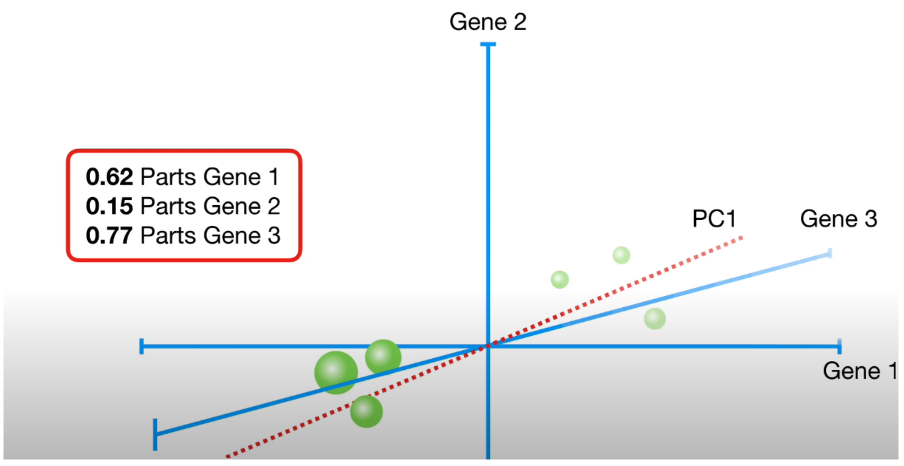


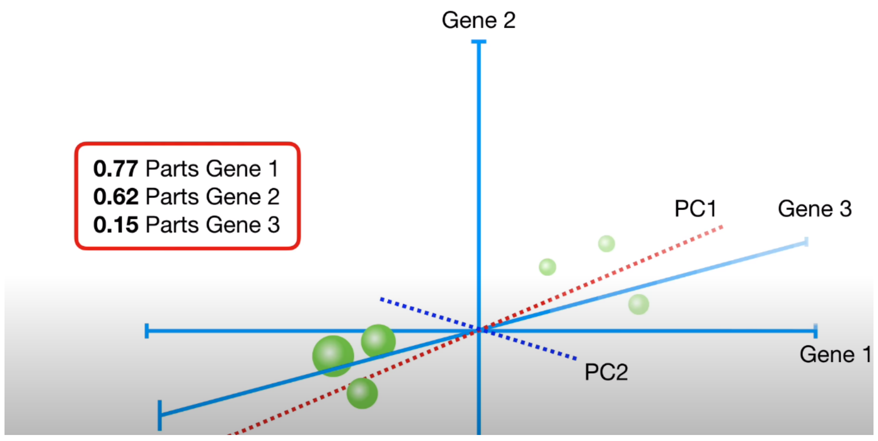
Step 2: Finding the covariance matrix

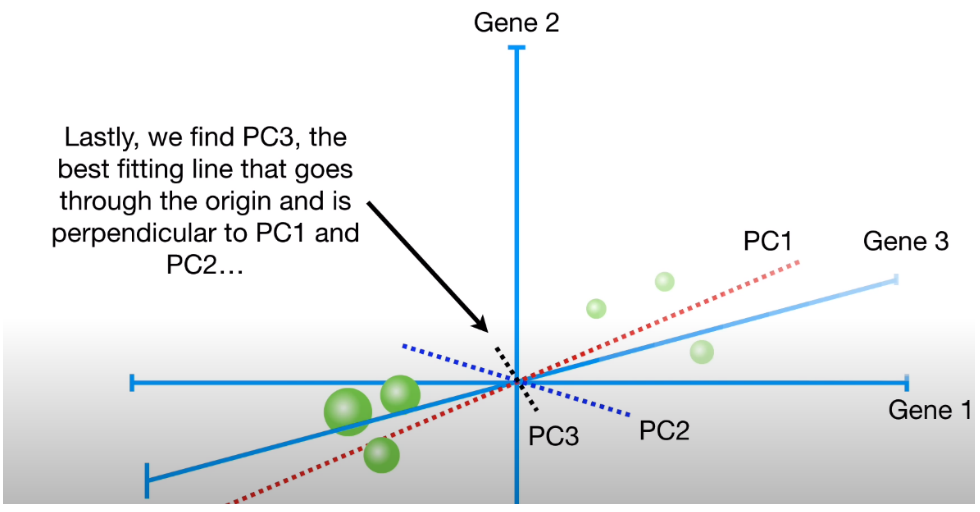
The objective of this step is to examine the variations of variables within the input dataset concerning the mean in relation to one another. In simpler terms, it aims to determine whether there exists any correlation between the variables. This exploration is crucial because variables can be strongly interrelated, potentially containing duplicative information. To unveil these relationships, the covariance matrix is calculated, helping identify correlations and patterns in the dataset.

$$\begin{bmatrix} Cov(x,x) & Cov(x,y) & Cov(x,z) \\ Cov(y,x) & Cov(y,y) & Cov(y,z) \\ Cov(z,x) & Cov(z,y) & Cov(z,z) \end{bmatrix}$$

Step 3: Finding the eigen values and eigen vectors of the covariance matrix

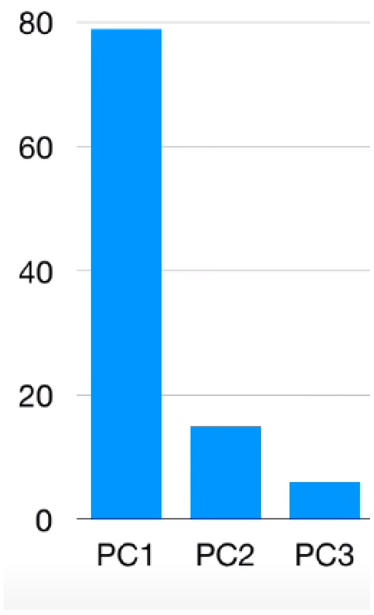


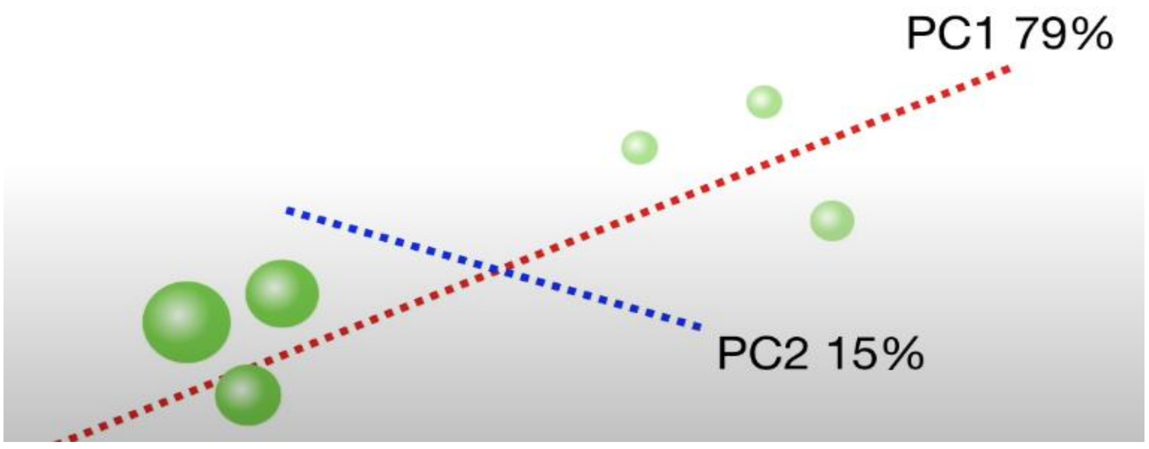



Gene 2
Lastly, we find PC3, the best fitting line that goes through the origin and is perpendicular to PC1 and PC2…
PC1
Gene 3
Gene 1
PC3
PC2

Step 4: Finding the screen plot

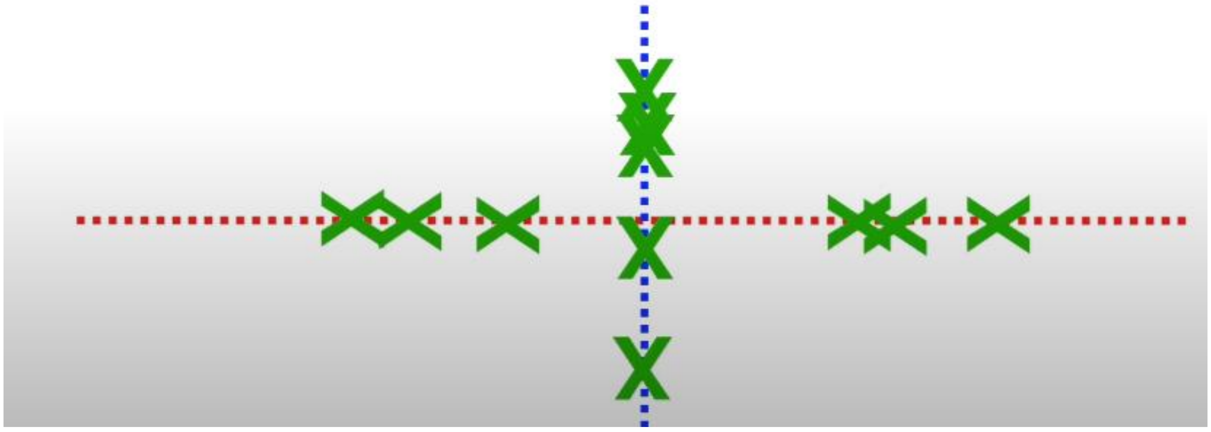
A screen plot is a graphical representation used in (PCA) to help determine the number of principal components to retain. It displays the eigenvalues of the principal components in decreasing order. The scree plot allows analysts to visually inspect the point at which the eigenvalues "level off," helping them decide how many principal components to keep in the analysis.

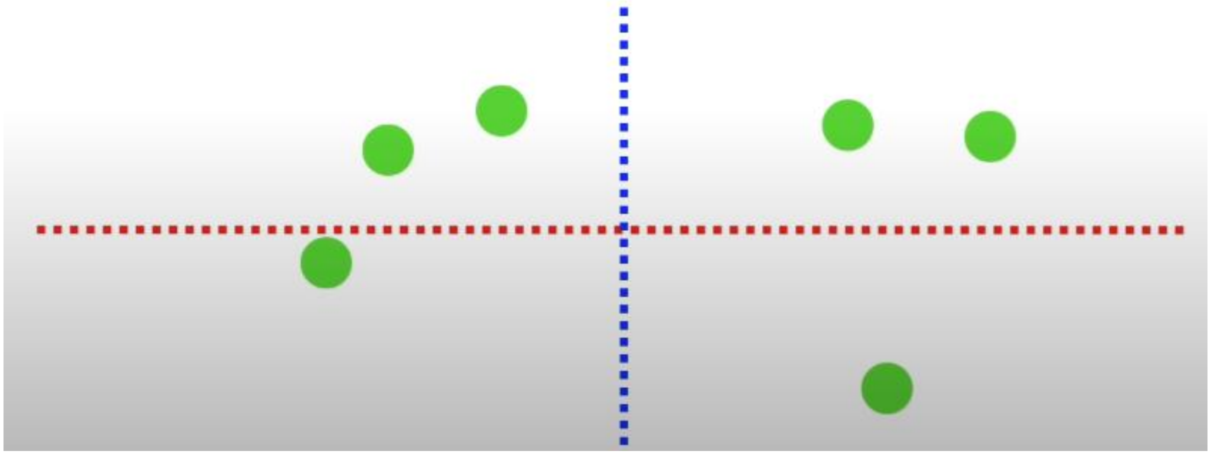




Step 5: Projecting the original data on to the PC components

If X is your centred data matrix, and L is the matrix of loading scores for the principal components you want to keep, then the projected data X_proj =X·L





Step 6: Computing the loading score and plotting loading scores

In PCA, when you calculate the eigenvectors of the covariance matrix, these eigenvectors are the loading vectors or loading scores.

The loading scores represent the weights or coefficients assigned to each original variable in the construction of the principal components.

PC loadings let us know:

- Which variables are influential.
- How variables are correlated.

How to make inferences in the context of EDA?

Principal Component Analysis (PCA) can be a valuable tool for Exploratory Data Analysis (EDA) in various ways, particularly in understanding and visualizing high-dimensional datasets. Here are some application ideas of PCA in the context of EDA:

1. ***Feature Importance:*** Assess the importance of individual features by examining their contribution to the principal components. Identifying which features have the most influence on the principal components can guide feature selection and engineering decisions.
2. ***Anomaly Detection:*** Detect anomalies or outliers in the data by examining data points that are far from the mean or that exhibit unusual patterns in the principal component space. Outliers in a lower-dimensional space may be more apparent than in the original high-dimensional space.
3. ***Grouping and Clustering:*** Use PCA to preprocess data before applying clustering algorithms, such as k-means or hierarchical clustering. Clustering is a valuable EDA technique for identifying patterns and groups within the data.
4. ***Data Separation:*** Explore the separation or overlap of classes or categories in your dataset. PCA can help visualize how well different groups can be distinguished in the reduced-dimensional space.
5. ***Visualization of Trends:*** Use PCA to visualize trends and patterns in time-series data or multivariate datasets. This can help in identifying cyclical patterns or shifts in data over time.
6. ***Identifying Redundant Features:*** PCA can reveal redundancies and correlations among features. Removing or consolidating redundant features can simplify the dataset and improve the interpretability of the EDA results.
7. ***Understanding Multicollinearity:*** PCA can help identify and quantify multicollinearity (high correlation between variables) in the dataset, which is essential when preparing data for regression analysis.
8. ***Principal Component Loadings:*** Examine the loadings of each original feature on the principal components to understand which variables contribute most to the principal components and how they are related.

9. ***Cross-Variable Relationships:*** Analyse how different variables relate to each other by exploring their patterns in the principal component space. This can uncover complex relationships that may not be immediately apparent in the original feature space.

Exploratory Data Analysis through PCA

EDA is the practice of summarizing, visualizing, and understanding the main characteristics of a dataset to discover patterns, trends, anomalies, and insights. It is often the first step in a data analysis project and helps data scientists or analysts get a better grasp of the data they are working with.

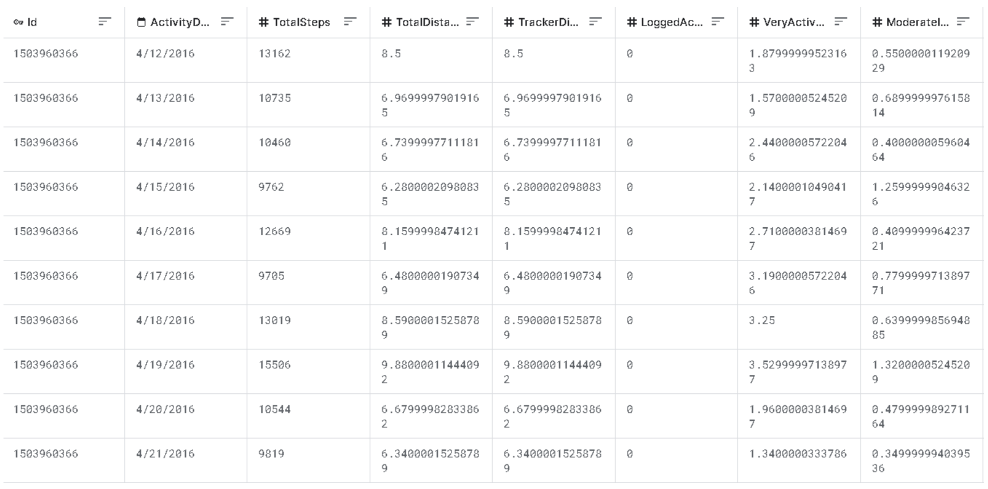
Problem Statement: Analysing the Fitbit fitness tracker dataset. To know which features are impacting the most variation in the dataset. Using Principal components to capture the variation in the dataset. Learn if any pattern is noticed.

Dataset:

Data is sourced from the public domain; dataset was made available through Mobius Data. Fitbit Fitness Tracker Data (CC0: Public Domain, dataset made available through Mobius) This dataset was generated by respondents to a distributed survey via Amazon Mechanical Turk between 03.12.2016–05.12.2016.

It contains personal fitness tracker from thirty Fitbit users who consented to the submission of personal tracker data, including minute-level output for physical activity, heart rate, and sleep monitoring. It includes information about daily activity, steps, and heart rate that can be used to explore users' habits.

Preview of dataset:



Id	ActivityD...	# TotalSteps	# TotalDista...	# TrackerDi...	# LoggedAc...	# VeryActiv...	# Moderatel...
1503960366	4/12/2016	13162	8.5	8.5	0	1.8799999952316 3	0.5500000119209 29
1503960366	4/13/2016	10735	6.9699997901916 5	6.9699997901916 5	0	1.5700000524520 9	0.6899999976158 14
1503960366	4/14/2016	10460	6.7399997711181 6	6.7399997711181 6	0	2.4400000572204 6	0.4000000059604 64
1503960366	4/15/2016	9762	6.2800002098083 5	6.2800002098083 5	0	2.1400001049041 7	1.2599999904632 6
1503960366	4/16/2016	12669	8.1599998474121 1	8.1599998474121 1	0	2.7100000381469 7	0.4099999964237 21
1503960366	4/17/2016	9705	6.4800000190734 9	6.4800000190734 9	0	3.1900000572204 6	0.7799999713897 71
1503960366	4/18/2016	13019	8.5900001525878 9	8.5900001525878 9	0	3.25	0.6399999856948 85
1503960366	4/19/2016	15506	9.8800001144409 2	9.8800001144409 2	0	3.5299999713897 7	1.3200000524520 9
1503960366	4/20/2016	10544	6.6799998283386 2	6.6799998283386 2	0	1.9600000381469 7	0.4799999892711 64
1503960366	4/21/2016	9819	6.3400001525878 9	6.3400001525878 9	0	1.3400000333786	0.3499999940395 36

Program code:

***Link for google colab notebook:* EDA_PCA**

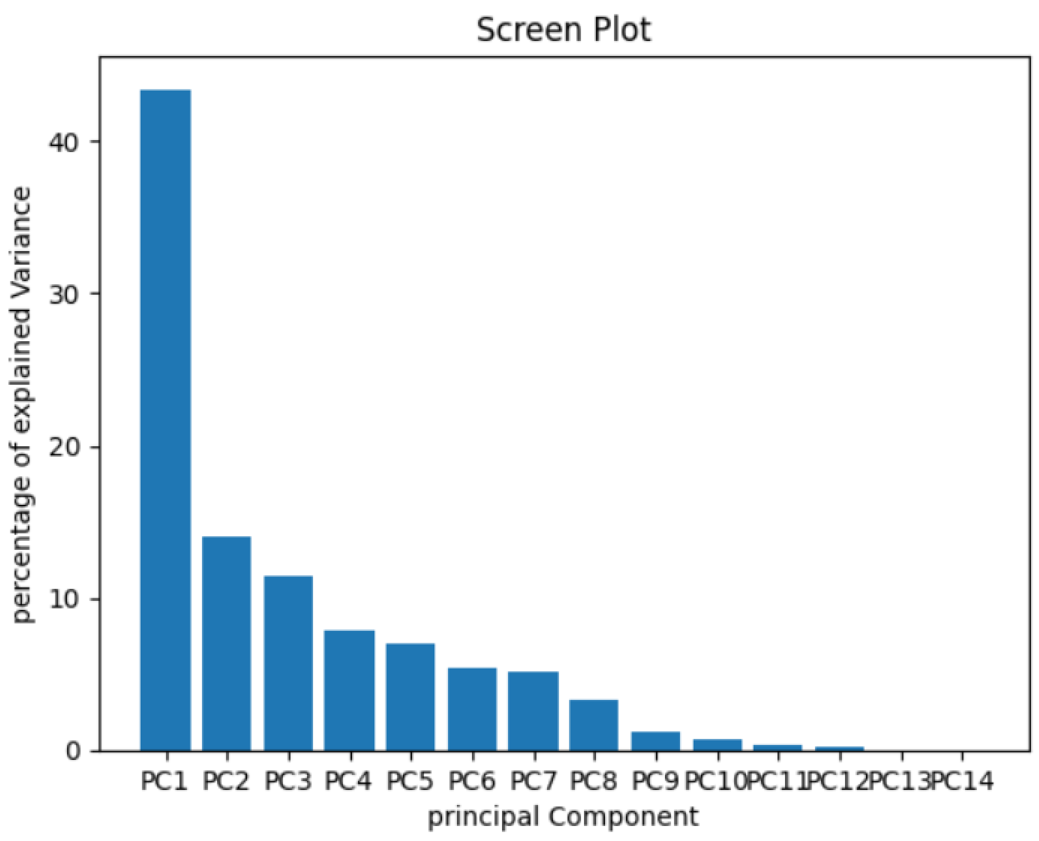
```
import pandas as pd
import numpy as np
from sklearn.decomposition import PCA
from sklearn import preprocessing
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler


# creating dataframe
df=pd.read_csv('dailyActivity_merged.csv')
data=pd.DataFrame(df)
print(data.head())

#Data preparation,to check if any null values are present

null_values = df.isnull()
null_counts = df.isnull().sum()

# Display the DataFrame with null values highlighted
print("Original DataFrame:")
print(df)

print("\nIdentified Null Values:")
print(null_values)

print("\nNull Value Counts:")
print(null_counts)

#Handling date data type

df['ActivityDate'] = pd.to_datetime(df['ActivityDate'])

# Separate the date column and numerical features
dates = df['ActivityDate']
numerical_features = df.drop('ActivityDate', axis=1)
```

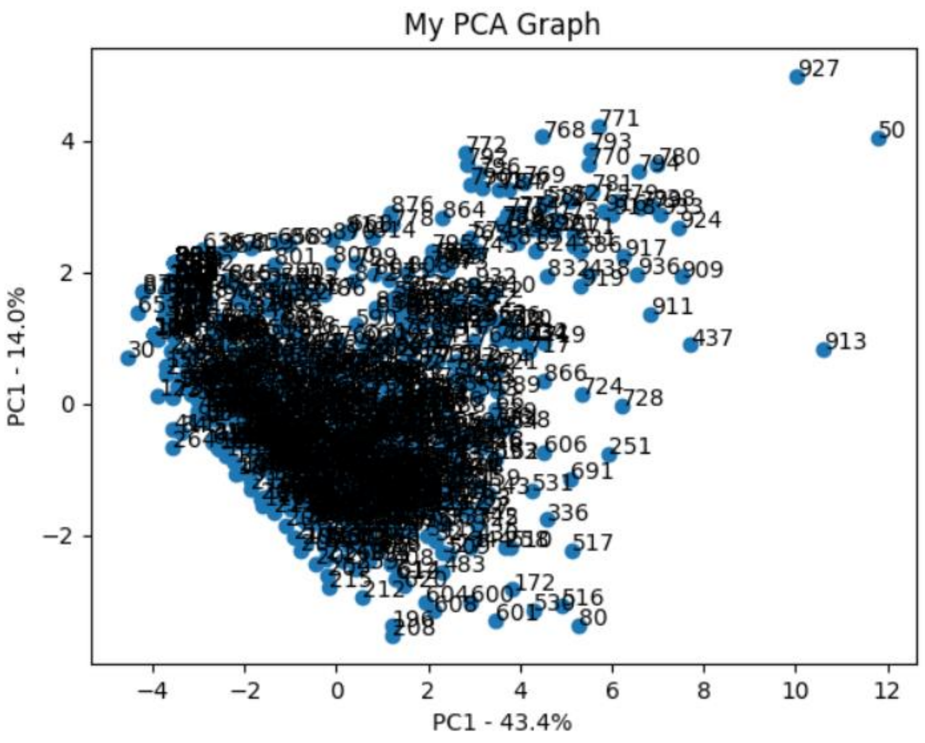
```
scaled_data=preprocessing.scale(numerical_features)

#Applying PCA analysis

pca=PCA()
pca.fit(scaled_data)
pca_data=pca.transform(scaled_data)

per_var=np.round(pca.explained_variance_ratio_*100,decimals=1)
labels=['PC'+str(x) for x in range(1,len(per_var)+1)]

plt.bar(x=range(1,len(per_var)+1),height=per_var,tick_label=labels)
plt.ylabel('percentage of explained Variance')
plt.xlabel('principal Component')
plt.title('Screen Plot')
print(plt.show())
#data.shape

pca_df=pd.DataFrame(pca_data, index=range(940),columns=labels)
plt.scatter(pca_df.PC1, pca_df.PC2)
plt.title('My PCA Graph')
plt.xlabel('PC1 - {0}%'.format(per_var[0]))
plt.ylabel('PC1 - {0}%'.format(per_var[1]))

for sample in pca_df.index:
  plt.annotate(sample, (pca_df.PC1.loc[sample], pca_df.PC2.loc[sample]))
print(plt.show())

#Loading score of each feature to know which feature have impacted the most variance

loading_scores = pca.components_

# Create a DataFrame with loading scores for each feature
loading_scores_df = pd.DataFrame(loading_scores, columns=numerical_features.columns)

# Side-by-side bar plot of loading scores for PC1 and PC2
bar_width = 0.35
```

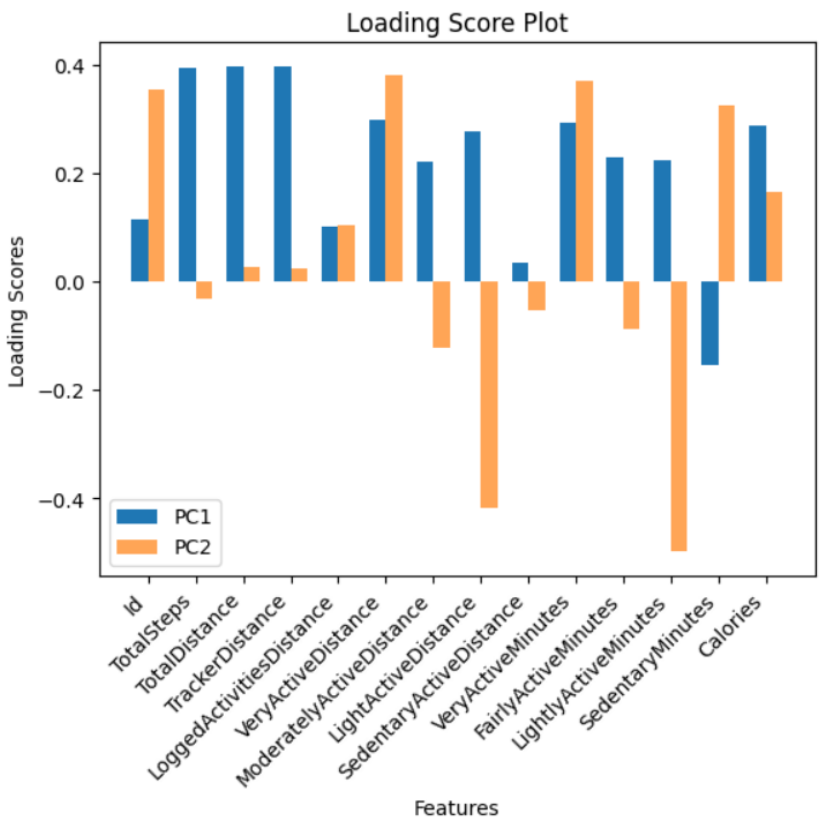
```
index = np.arange(len(numerical_features.columns))

plt.bar(index, loading_scores_df.iloc[0, :], width=bar_width, label='PC1')
plt.bar(index + bar_width, loading_scores_df.iloc[1, :], width=bar_width, label='PC2', alpha=0.7)

plt.xlabel('Features')
plt.ylabel('Loading Scores')
plt.title('Loading Score Plot')
plt.xticks(index + bar_width / 2, numerical_features.columns, rotation=45, ha='right')  # Rotate x-axis labels
plt.legend()
plt.show()

#to print scatter plot

variables = numerical_features.columns  # Replace with your actual variable names

# Scatter plot of loading scores for PC1 and PC2
plt.scatter(loading_scores[0], loading_scores[1])
plt.xlabel("Loading Scores PC1")
plt.ylabel("Loading Scores PC2")
plt.title("Loading Scores Scatter Plot")
# Annotate the points with variable names
label_offset = 0.01  # Adjust this value to control the distance between points and labels
for i, variable in enumerate(variables):
    plt.text(loading_scores[0, i] + label_offset, loading_scores[1, i] + label_offset, variable)

# Add grid for better readability
plt.grid(True)

# Draw lines to divide the plot into four quadrants
plt.axhline(0, color='black',linewidth=0.5)
plt.axvline(0, color='black',linewidth=0.5)

plt.tight_layout()

# Show the plot
plt.show()
print(loading_scores_df)
```

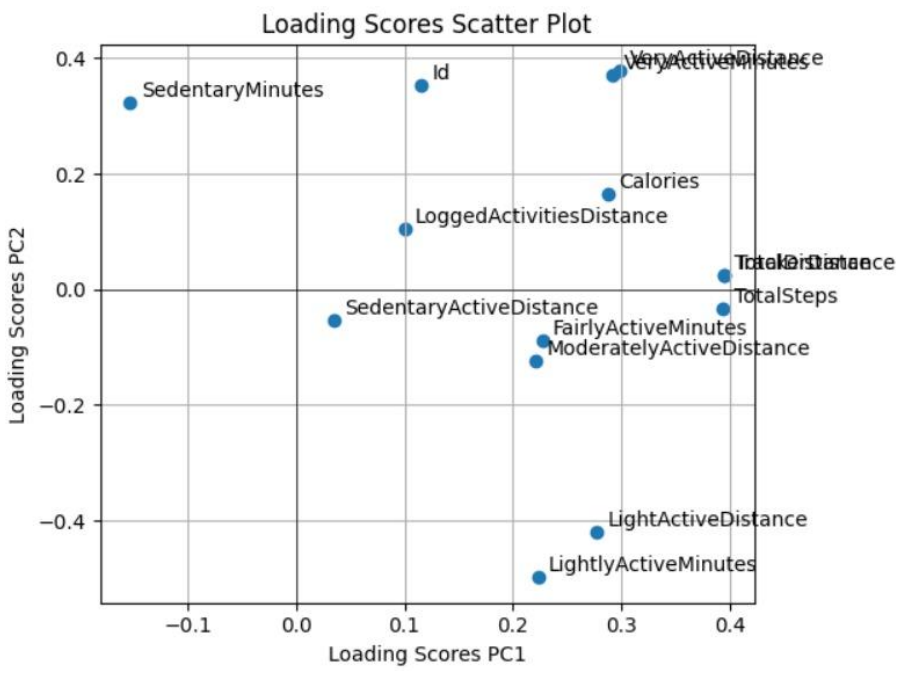
Output:

Graphs:




My PCA Graph
PC1 - 14.0%
PC1 - 43.4%
927
50
771
768
793
770
780
913
437
911
909
924
917
936
866
724
728
606
251
691
531
336
517
172
516
80
601
212
213
196
208
30


Loading Score Plot
Loading Scores
Features
PC1
PC2
Id
TotalSteps
TotalDistance
TrackerDistance
LoggedActivitiesDistance
VeryActiveDistance
ModeratelyActiveDistance
LightActiveDistance
SedentaryActiveDistance
VeryActiveMinutes
FairlyActiveMinutes
LightlyActiveMinutes
SedentaryMinutes
Calories



Inference:

- From the scatter plot, the data which is grouped together shows positive correlation and the data points which is are in the opposite directions are negatively correlated.
- Here, [Fairly active minutes, moderately active distance], [total Distance, Total Steps], [Light active Distance, Lightly Active Minutes] are positively correlated. We can also notice that [calories, very Active Distance] though look is not grouped are vertically aligned and we know that PC2<PC1 in terms of capturing variation, therefore they are close vertically than horizontal data points.
- Similarly, [Sedentary Minutes, (Light Active Distance, Lightly Active Minutes)] are negatively correlated.
- Here, features which are farther away from origin can be extracted as features which will create most impact in our model.

Conclusion:

In conclusion, the Principal Component Analysis (PCA) project has proven to be a valuable tool for understanding and extracting essential insights from our dataset. Through the systematic reduction of dimensionality, we have successfully transformed a high-dimensional set of features into a more manageable and interpretable form.

The key findings and outcomes of the PCA project include:

1. **Dimensionality Reduction:**
    - PCA effectively reduced the dimensionality of the dataset by identifying and retaining the most significant principal components. This not only simplified the data but also facilitated a clearer understanding of the underlying patterns.
2. **Variance Explanation:**
    - The principal components selected in our analysis accounted for a substantial portion of the variance in the original data. This implies that the reduced set of features captures the essential information present in the dataset.
3. **Feature Importance:**
    - Through the examination of loading scores, we identified the contribution of each original feature to the principal components. This information is crucial for discerning the features that play a pivotal role in shaping the dataset's structure.
4. **Interpretability:**
    - The reduced-dimensional representation provided by PCA enhances interpretability. Patterns and relationships between features have become more apparent, enabling a more intuitive grasp of the dataset's intrinsic characteristics.
5. **Optimization Opportunities:**
    - By identifying redundant or less informative features, PCA opens avenues for optimizing subsequent analyses and models. This optimization can lead to improved computational efficiency and enhanced model performance.

****