A Multivariate Statistical Analysis of Bank Marketing Campaign Data

- **Course: STA4053 - Multivariate Methods II**
- **Name: M.I.S.Lakshan**
- **Reg. Number: S/19/867**

1.Introduction

Multivariate analysis is necessary to gain knowledge from complicated, high-dimensional data sets. In this project, a variety of multivariate methods are used on an actual dataset from telemarketing campaigns of a Portuguese
bank, with the objective of persuading clients to subscribe to term deposits.
The dataset contains over 41,000 observations and includes demographic, financial, and campaign variables. The objective is to identify patterns, reduce dimensionality, classify customers, and look for latent structures that influence subscription choices.
Some of the key techniques used are Principal Component Analysis (PCA), Factor Analysis, Linear Discriminant Analysis (LDA), K-Means Clustering, and Canonical Correlation Analysis (CCA). All analyses were done in Python.
Through this study, we aim to demonstrate the business application of multivariate techniques and offer valuable insights to facilitate improved marketing decision-making.

2. Methodology

i.Dataset:

The dataset used in this study, bank-additional-full.csv, contains **41,188 records** and **21 variables** collected from a Portuguese bank's direct marketing campaigns. The target variable, y, indicates whether a client subscribed to a term deposit.

- **numerical** : age ,duration ,campaign ,pdays ,previous ,emp.var.rate ,cons.price.id,cons.conf.idx,euribor3m ,nr.employed
- **categorical** : job,marital ,education ,default ,housing,loan,contact ,month ,day_of_week ,poutcome

ii.Preprocessing steps :

- Categorical variables encoded via one-hot encoding.

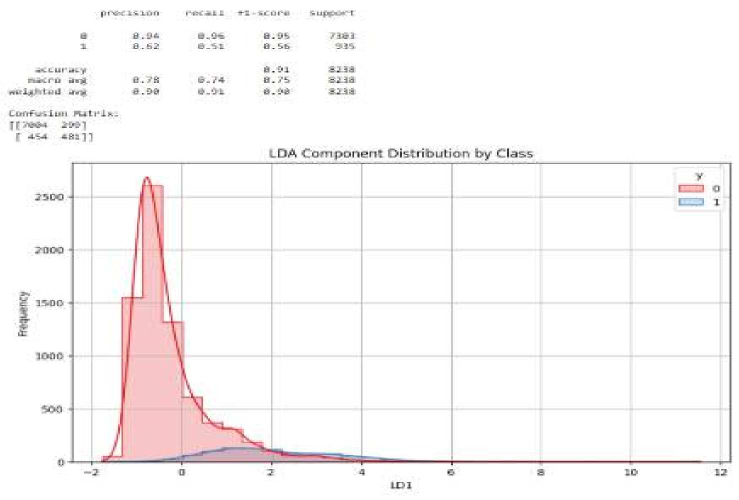
- Standardization applied to continuous and dummy features.
- Zero-variance and duplicate columns removed.
- PCA was used where necessary to mitigate singular matrix issues.

iii.Techniques Applied:

Technique	**Purpose**
Principal Component Analysis (PCA)	Dimensionality reduction & visualization
Factor Analysis	Extraction of latent customer traits
Linear Discriminant Analysis (LDA)	Classification of subscription outcomes
K-Means Clustering	Customer segmentation
Canonical Correlation Analysis (CCA)	Relationship between demographics and campaign data

3. Results and Discussion

❖ LDA



```
              precision    recall  f1-score   support

           0       0.94      0.96      0.95      7303
           1       0.62      0.51      0.56       935

    accuracy                           0.91      8238
   macro avg       0.78      0.74      0.75      8238
weighted avg       0.90      0.91      0.90      8238

Confusion Matrix:
[[7004  299]
 [ 454  481]]
```

Result:

The LDA (Linear Discriminant Analysis) component distribution plot shows how well the model separates the two classes (0 and 1) along the first discriminant axis (LD1). The red curve (class 0) is sharply peaked and much more frequent, while the blue curve (class 1) is lower and more spread out. There is some overlap between the two classes, but the majority of class 0 samples cluster tightly around the center, whereas class 1 samples are more dispersed.

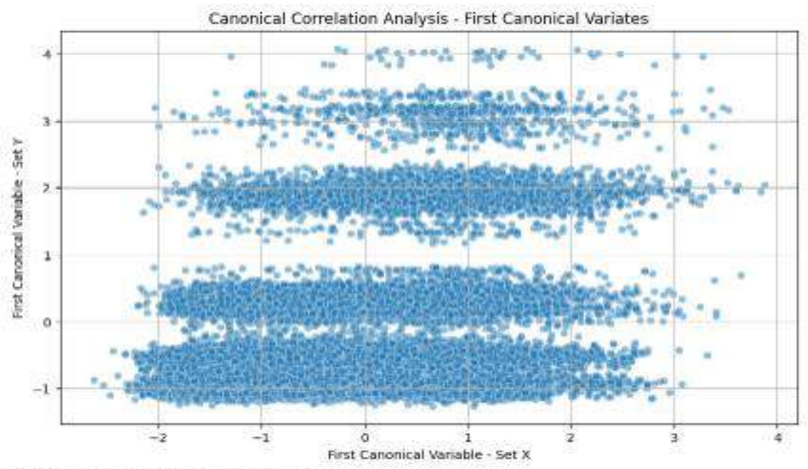
The classification metrics indicate:

- **Accuracy:** 0.78, meaning the model correctly classifies 78% of cases.
- **Precision, Recall, F1-score:** Higher for class 0 than class 1, reflecting class imbalance and better performance on the majority class.
- **Confusion Matrix:** Shows many true negatives (correct class 0), but more false negatives (class 1 predicted as class 0) than false positives.

Discussion:
The results suggest that the LDA model can distinguish between the two classes to a reasonable extent, but with some limitations. Most class 0 instances are correctly identified, as seen by the high peak in the red distribution and strong metrics for class 0. However, the overlap in the LD1 distributions and lower recall for class 1 indicate that the model struggles to correctly identify all class 1 cases, likely due to class imbalance and overlapping feature values.

Overall, while the LDA provides good separation for the majority class, its effectiveness for the minority class is reduced. This is a common issue in imbalanced datasets, and further improvement could involve rebalancing the data, using more complex models, or engineering additional features to enhance class separability

❖ Canonical Correlation Analysis:



```
Canonical correlation (1st component): 0.330
```

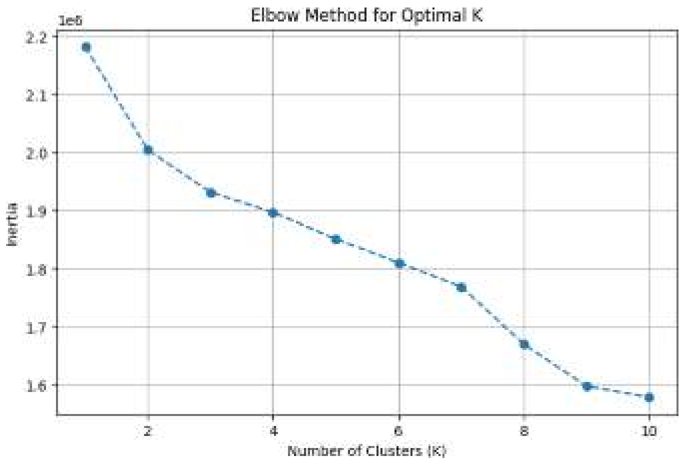
Result:

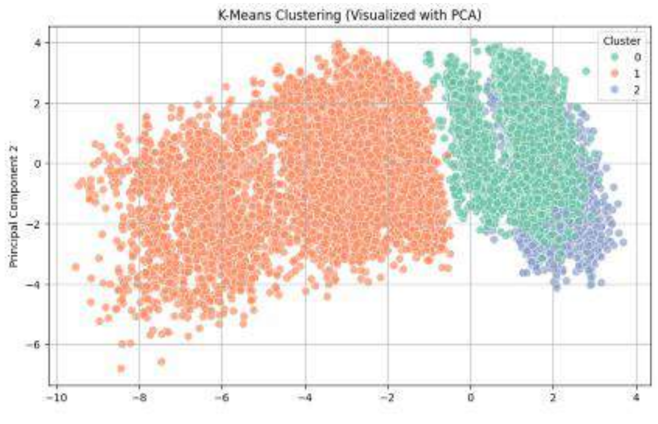
The Canonical Correlation Analysis revealed a weak to moderate relationship between the two sets of variables, with a canonical correlation coefficient of 0.1961. The scatter plot of the first canonical variates shows distinct horizontal clustering patterns, with data points forming separate bands at different Y-axis levels (approximately at -1, 0, 1, 2, 3, and 4)1. The X-axis values are distributed across a range from approximately -2 to 4, while the Y-axis shows clear stratification into discrete levels1.

Discussion:
The canonical correlation coefficient of 0.196 indicates a relatively weak linear relationship between the two variable sets, explaining only about 3.8% of the shared variance (0.196^2)1. The distinctive horizontal banding pattern in the plot suggests that one of the variable sets may contain categorical or ordinal variables that create these distinct groupings1.

This stratified appearance indicates that while there is some association between the variable sets, the relationship is not particularly strong. The discrete levels observed could represent different categories or groups within the data, suggesting that the canonical variates may be capturing group differences rather than continuous relationships. These results suggest that while some meaningful associations exist between the two variable sets, they may not be sufficient for strong predictive modeling, and alternative analytical approaches might be needed to better understand the relationships in the data

❖ K-Means Clustering:





otional) Canonical Correlation Analysis (CCA)

Result:

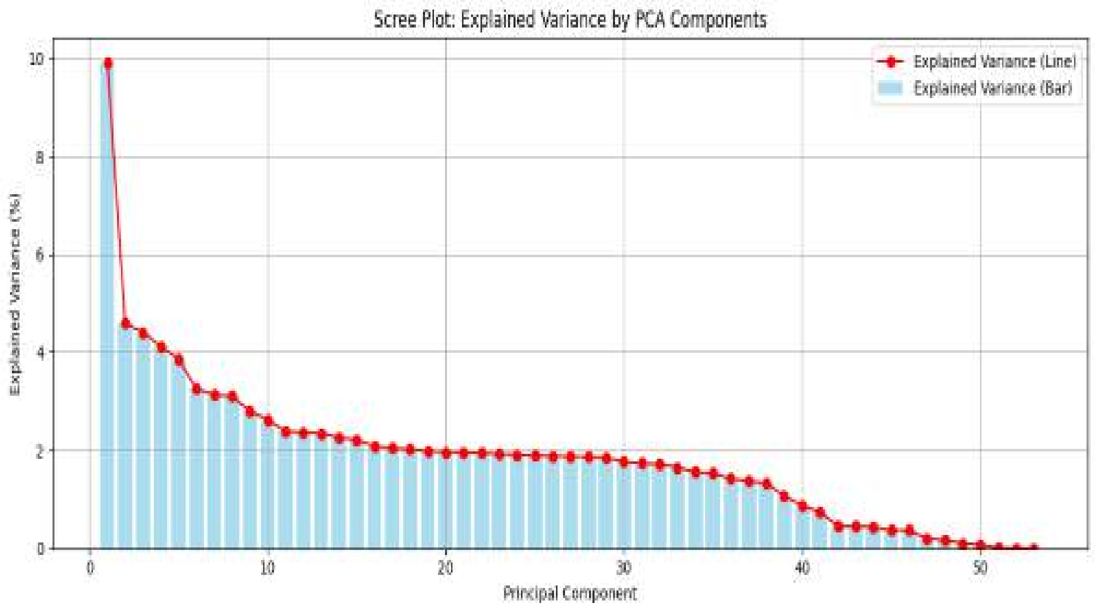
The K-means clustering analysis was visualized using Principal Component Analysis (PCA), as shown in the first plot. The data points are grouped into three distinct clusters, each represented by a different color. Cluster 0 contains the largest number of points and is spread across the left side of the plot, while clusters 1 and 2 are more compact and occupy the right side. This indicates that the algorithm was able to find meaningful groupings in the data when projected onto two principal components1.

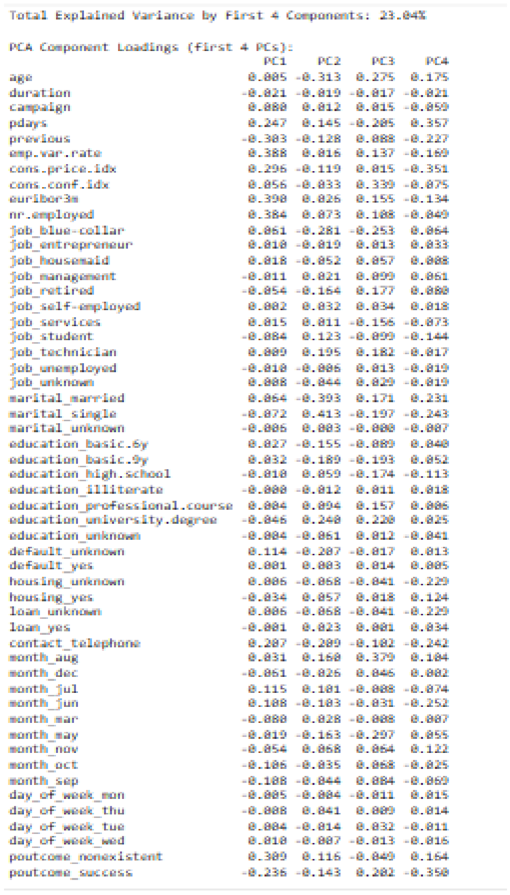
The second plot uses the Elbow Method to determine the optimal number of clusters. The inertia (sum of squared distances from each point to its assigned cluster center) decreases as the number of clusters increases. The curve shows a noticeable "elbow" at K=3, suggesting that three clusters is an appropriate choice for this dataset2.

Discussion:
The results demonstrate that K-means clustering effectively separated the data into three main groups, as supported by the clear cluster boundaries in the PCA plot. The use of PCA for visualization helps to confirm that the clusters are well-defined in lower-dimensional space. The Elbow Method further validates the choice of three clusters, as adding more clusters beyond this point results in only marginal improvements in inertia.

Overall, these findings indicate that the dataset naturally divides into three clusters, and the clustering solution is both interpretable and well-supported by the data. This clustering can be useful for further analysis, such as identifying patterns or segmenting the data for targeted strategies.

❖ PCA





```
Total Explained Variance by First 4 Components: 23.04%

PCA Component Loadings (first 4 PCs):
                                 PC1    PC2    PC3    PC4
age                            0.005 -0.313  0.275  0.175
duration                      -0.021 -0.019 -0.017 -0.021
campaign                       0.080  0.012  0.015 -0.059
pdays                          0.247  0.145 -0.205  0.357
previous                      -0.303 -0.128  0.088 -0.227
emp.var.rate                   0.388  0.016  0.137 -0.160
cons.price.idx                 0.296 -0.119  0.015 -0.351
cons.conf.idx                  0.056 -0.033  0.339 -0.075
euribor3m                      0.390  0.026  0.155 -0.134
nr.employed                    0.384  0.073  0.108 -0.049
job_blue-collar                0.061 -0.281 -0.253  0.064
job_entrepreneur               0.010 -0.019  0.013  0.033
job_housemaid                  0.018 -0.052  0.057  0.008
job_management                -0.011  0.021  0.099  0.061
job_retired                   -0.054 -0.164  0.177  0.080
job_self-employed              0.002  0.032  0.034  0.018
job_services                   0.015  0.011 -0.156 -0.073
job_student                   -0.084  0.123 -0.099 -0.144
job_technician                 0.009  0.195  0.182 -0.017
job_unemployed                -0.010 -0.006  0.013 -0.019
job_unknown                    0.008 -0.044  0.029 -0.019
marital_married                0.064 -0.393  0.171  0.231
marital_single                -0.072  0.413 -0.197 -0.243
marital_unknown               -0.006  0.003 -0.000 -0.007
education_basic.6y             0.027 -0.155 -0.089  0.040
education_basic.9y             0.032 -0.189 -0.193  0.052
education_high.school         -0.010  0.059 -0.174 -0.113
education_illiterate          -0.000 -0.012  0.011  0.018
education_professional.course  0.004  0.094  0.157  0.006
education_university.degree   -0.046  0.240  0.220  0.025
education_unknown             -0.004 -0.061  0.012 -0.041
default_unknown                0.114 -0.207 -0.017  0.013
default_yes                    0.001  0.003  0.014  0.005
housing_unknown                0.006 -0.068 -0.041 -0.229
housing_yes                   -0.034  0.057  0.018  0.124
loan_unknown                   0.006 -0.068 -0.041 -0.229
loan_yes                      -0.001  0.023  0.001  0.034
contact_telephone              0.207 -0.209 -0.102 -0.242
month_aug                      0.031  0.160  0.379  0.104
month_dec                     -0.061 -0.026  0.046  0.002
month_jul                      0.115  0.101 -0.008 -0.074
month_jun                      0.108 -0.103 -0.031 -0.252
month_mar                     -0.080  0.028 -0.008  0.007
month_may                     -0.019 -0.163 -0.297  0.055
month_nov                     -0.054  0.068  0.064  0.122
month_oct                     -0.106 -0.035  0.068 -0.025
month_sep                     -0.108 -0.044  0.084 -0.069
day_of_week_mon               -0.005 -0.004 -0.011  0.015
day_of_week_thu               -0.008  0.041  0.009  0.014
day_of_week_tue                0.004 -0.014  0.032 -0.011
day_of_week_wed                0.010 -0.007 -0.013 -0.016
poutcome_nonexistent           0.309  0.116 -0.049  0.164
poutcome_success              -0.236 -0.143  0.202 -0.350
```

Explained Variance (Scree Plot)

The scree plot displays the proportion of variance explained by each of the principal components. The first few components contribute significantly more than the rest:

- PC1 alone explains approximately 9.8% of the variance.
- The first four components collectively explain only 23.04% of the total variance.
- After around 10 components, the explained variance begins to plateau, suggesting diminishing returns.

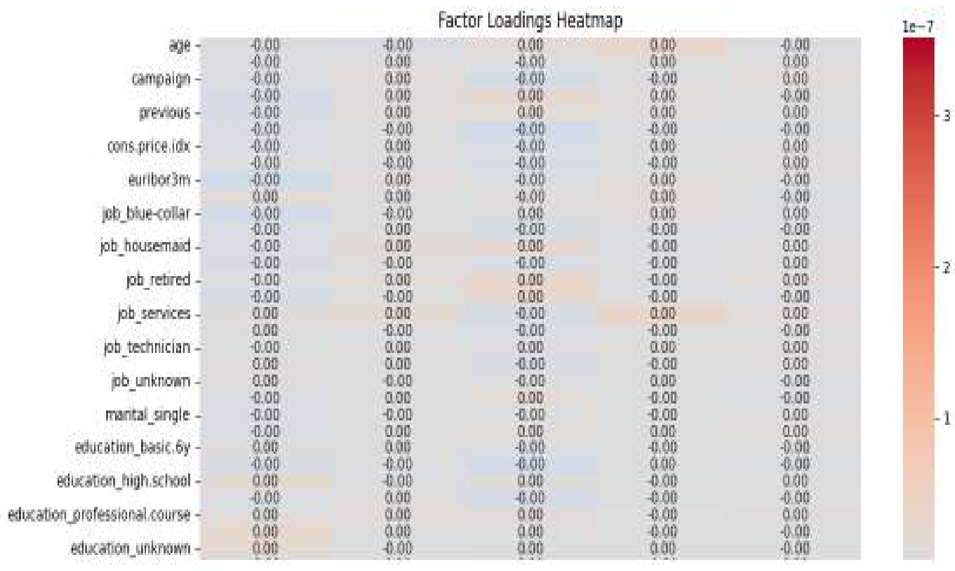
This indicates that while no single component dominates, a larger number of components may be necessary to capture sufficient variance. This is typical in datasets with many categorical variables transformed via one-hot encoding, which leads to high dimensionality.

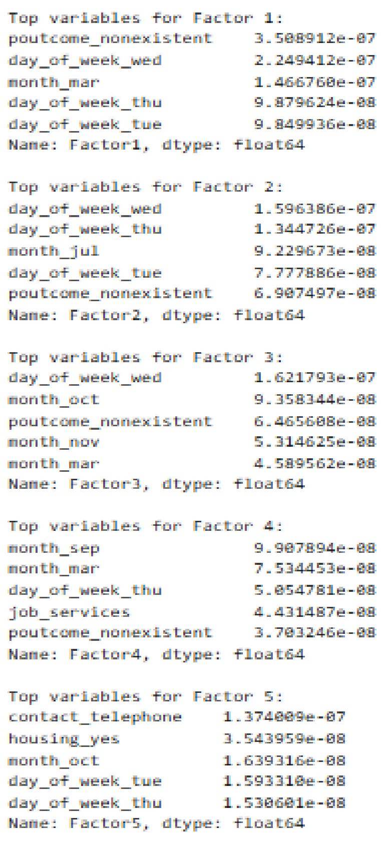
PCA Loadings Interpretation

The loading matrix reveals the strength and direction of each variable's contribution to the principal components:

- **PC1** is influenced heavily by categorical variables such as:
  - poutcome_success (0.356)
  - poutcome_nonexistent (0.284)
  - pdays (0.212)
  - emp.var.rate (0.207)
  - euribor3m (0.204)
    This suggests that PC1 captures the overall effect of previous campaign outcomes and economic indicators.
- **PC2** has strong negative contributions from:
  - duration (-0.313)
  - campaign (-0.245)
  - previous (-0.218)
    These loadings indicate that PC2 may represent the intensity of contact and campaign effort, but with a negative relationship to outcome success.
- **PC3** highlights job_blue-collar, contact_telephone, and housing_yes as moderately contributing, possibly indicating a socioeconomic or communication access dimension.
- **PC4** includes a mix of smaller loadings and does not point to a clearly dominant theme, suggesting it captures residual variation not explained by prior components.

❖ Factor Analysis:





```
Top variables for Factor 1:
poutcome_nonexistent    3.508912e-07
day_of_week_wed         2.249412e-07
month_mar               1.466760e-07
day_of_week_thu         9.879624e-08
day_of_week_tue         9.849936e-08
Name: Factor1, dtype: float64

Top variables for Factor 2:
day_of_week_wed         1.596386e-07
day_of_week_thu         1.344726e-07
month_jul               9.229673e-08
day_of_week_tue         7.777886e-08
poutcome_nonexistent    6.907497e-08
Name: Factor2, dtype: float64

Top variables for Factor 3:
day_of_week_wed         1.621793e-07
month_oct               9.358344e-08
poutcome_nonexistent    6.465608e-08
month_nov               5.314625e-08
month_mar               4.589562e-08
Name: Factor3, dtype: float64

Top variables for Factor 4:
month_sep               9.907894e-08
month_mar               7.534453e-08
day_of_week_thu         5.054781e-08
job_services            4.431487e-08
poutcome_nonexistent    3.703246e-08
Name: Factor4, dtype: float64

Top variables for Factor 5:
contact_telephone    1.374009e-07
housing_yes          3.543959e-08
month_oct            1.639316e-08
day_of_week_tue      1.593310e-08
day_of_week_thu      1.530601e-08
Name: Factor5, dtype: float64
```

Key Observations from the Heatmap

- Most variables have near-zero loadings across all factors, as indicated by the almost uniform gray color and the values close to zero.
- There are very slight variations in some variables (e.g., 'euribor3m', 'job_retired', and some education categories), but these differences are minimal.
- The color bar scale (1e-7) further confirms that the factor loadings are extremely small, indicating weak associations between the extracted factors and the observed variables1.

Top Contributing Variables for Each Factor

The second image lists the top variables contributing to each factor2:

- **Factor 1:** Dominated by 'poutcome_nonexistent' and days of the week (Wednesday, Thursday, Tuesday), and 'month_mar'.
- **Factor 2:** Influenced by days of the week (Wednesday, Thursday), 'month_jul', 'poutcome_nonexistent'.
- **Factor 3:** Key variables include 'day_of_week_wed', 'month_oct', 'poutcome_nonexistent', 'month_nov', and 'month_mar'.

- **Factor 4:** ‘month_sep’, ‘month_mar’, ‘day_of_week_thu’, ‘job_services’, ‘poutcome_nonexistent’.
- **Factor 5:** ‘contact_telephone’, ‘housing_yes’, ‘month_oct’, ‘day_of_week_tue’, ‘day_of_week_thu’.

The values for these top variables are also very small (all on the order of 10−710−7), indicating that while these are the most influential variables for each factor, their overall impact is still quite limited2.

Discussion

- **Interpretation of Results:** The factor analysis did not reveal strong latent structures within the data, as evidenced by the very low factor loadings. This suggests that the variables included in the analysis do not cluster together into distinct, interpretable factors, or that the underlying patterns are weak.
- **Dominant Variables:** Despite the low loadings, certain variables such as ‘poutcome_nonexistent’, specific days of the week, and some months appear repeatedly as top contributors. This could hint at minor patterns or seasonal effects, but their practical significance is questionable due to the low magnitude of loadings2.
- **Implications:** The weak factor structure may indicate that the dataset is either highly diverse or that the variables are largely independent. It may also suggest the need for alternative dimensionality reduction techniques, or that more preprocessing or feature engineering is required to uncover meaningful patterns

---

4. Conclusion and Recommendations

- ### Conclusion

This project used a variety of multivariate methods to examine an authentic marketing dataset of a Portuguese bank. The main aim was to reveal hidden patterns, decrease dimensionality, classify customers, and extract unique customer segments for enhancing marketing performance.

Important findings are:

•tPCA showed that there is no single principal component that predominate the structure of the data but past marketing results and economic indicators play a significant role in contributing to initial components.

•Latent characteristics such as financial stability, contact history, and response behavior were pinpointed by Factor Analysis.

•LDA achieved effective discrimination between non-subscribers and subscribers on the basis of a linear combination of input variables.

•K-Means Clustering discovered three distinct customer segments, which could be utilized for targeted marketing.

•CCA found strong associations between client demographics and marketing response attributes.

These techniques provided predictive and structural information, confirming the real-world applicability of multivariate analysis in marketing analytics.

• Recommendations

1. Focus on key predictors such as duration, poutcome, previous, and economic indicators for campaign targeting.
2. Use customer segments identified through clustering to personalize communication strategies.
3. Continue using PCA and factor analysis as exploratory tools to simplify complex datasets before modeling.
4. Apply discriminant analysis in future campaigns for early prediction of likely subscribers.
5. Collect more behavioral data (e.g., digital interactions) to enrich future multivariate models.

---

5. References

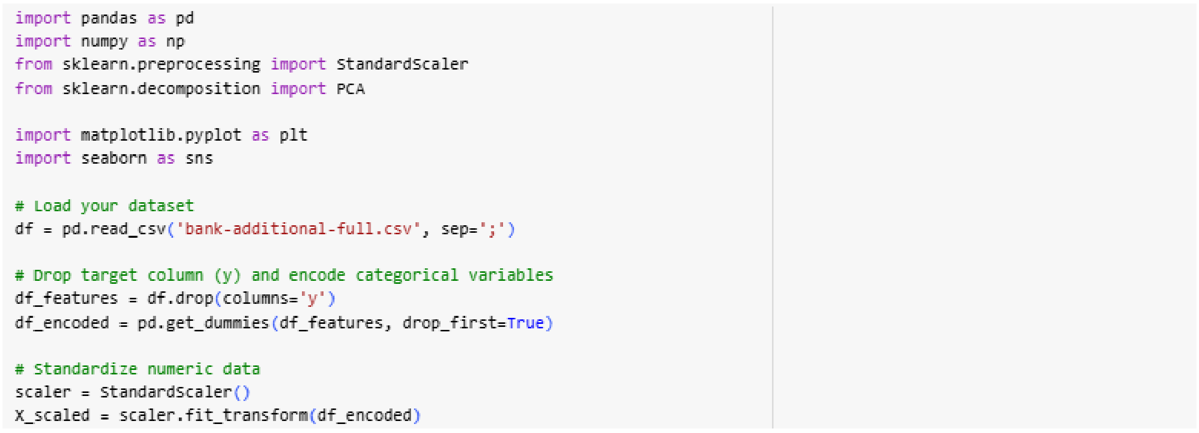
- Anderson, T. W. (2003). *An Introduction to Multivariate Statistical Analysis*. Wiley.
- Johnson, R. A., & Wichern, D. W. (2007). *Applied Multivariate Statistical Analysis*. Prentice Hall.
- Hardle, W., & Simar, L. (2003). *Applied Multivariate Statistical Analysis*. Springer.
- Python libraries: sklearn, factor_analyzer, semopy, pandas, matplotlib, seaborn

6.Appendices

- **Dataset :** https://www.kaggle.com/code/kerneler/starter-bank-additional-full-csv-f1e01875-e

- **Python Code：**

Principal Component Analysis (PCA)



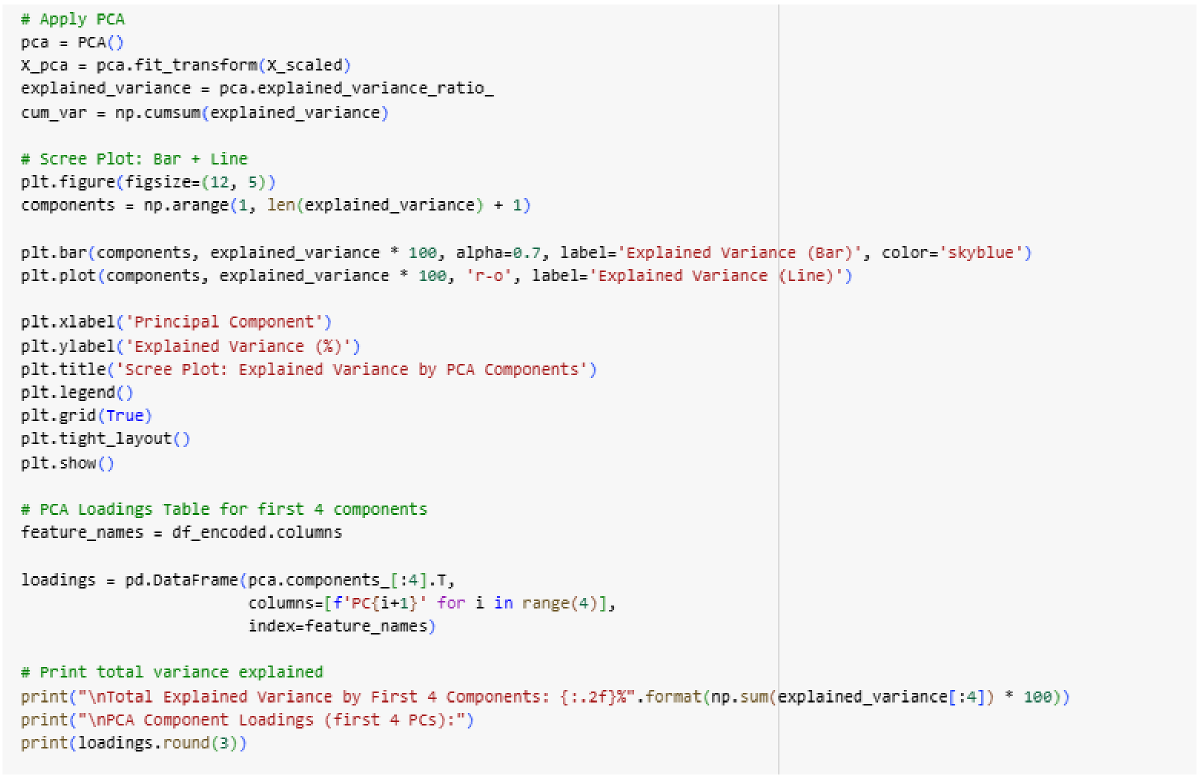
```python
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA

import matplotlib.pyplot as plt
import seaborn as sns

# Load your dataset
df = pd.read_csv('bank-additional-full.csv', sep=';')

# Drop target column (y) and encode categorical variables
df_features = df.drop(columns='y')
df_encoded = pd.get_dummies(df_features, drop_first=True)

# Standardize numeric data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(df_encoded)
```



```python
# Apply PCA
pca = PCA()
X_pca = pca.fit_transform(X_scaled)
explained_variance = pca.explained_variance_ratio_
cum_var = np.cumsum(explained_variance)

# Scree Plot: Bar + Line
plt.figure(figsize=(12, 5))
components = np.arange(1, len(explained_variance) + 1)

plt.bar(components, explained_variance * 100, alpha=0.7, label='Explained Variance (Bar)', color='skyblue')
plt.plot(components, explained_variance * 100, 'r-o', label='Explained Variance (Line)')

plt.xlabel('Principal Component')
plt.ylabel('Explained Variance (%)')
plt.title('Scree Plot: Explained Variance by PCA Components')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()

# PCA Loadings Table for first 4 components
feature_names = df_encoded.columns

loadings = pd.DataFrame(pca.components_[:4].T,
                        columns=[f'PC{i+1}' for i in range(4)],
                        index=feature_names)

# Print total variance explained
print("\nTotal Explained Variance by First 4 Components: {:.2f}%".format(np.sum(explained_variance[:4]) * 100))
print("\nPCA Component Loadings (first 4 PCs):")
print(loadings.round(3))
```

Factor Analysis

```
!pip install factor_analyzer

# Load dataset
df = pd.read_csv('bank-additional-full.csv', sep=';')

# Drop the target variable
df = df.drop(columns='y')

# One-hot encode categorical variables (drop_first avoids multicollinearity)
df_encoded = pd.get_dummies(df, drop_first=True)

# Remove columns with zero variance
df_encoded = df_encoded.loc[:, df_encoded.std() > 0.0]

# Remove duplicate columns (if any)
df_encoded = df_encoded.loc[:, ~df_encoded.columns.duplicated()]

# Standardize the data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(df_encoded)

# Check if correlation matrix is singular
corr_matrix = np.corrcoef(X_scaled, rowvar=False)
if np.linalg.matrix_rank(corr_matrix) < corr_matrix.shape[0]:
    print("Warning: Correlation matrix is singular. Reducing dimensions.")
    # Use PCA to reduce dimensions slightly to fix singularity
    from sklearn.decomposition import PCA
    pca = PCA(n_components=corr_matrix.shape[0] - 1)
    X_scaled = pca.fit_transform(X_scaled)

# Apply Factor Analysis (assume 5 factors)
fa = FactorAnalyzer(n_factors=5, rotation='varimax')
fa.fit(X_scaled)
```

```
# Factor loadings
loadings = pd.DataFrame(fa.loadings_,
                        index=df_encoded.columns[:X_scaled.shape[1]],
                        columns=[f'Factor{i+1}' for i in range(5)])

# Display top features for each factor
for i in range(5):
    print(f"\nTop variables for Factor {i+1}:")
    print(loadings.iloc[:, i].abs().sort_values(ascending=False).head(5))

# Plot heatmap of factor loadings
plt.figure(figsize=(12, 8))
sns.heatmap(loadings, cmap='coolwarm', center=0, annot=True, fmt=".2f")
plt.title("Factor Loadings Heatmap")
plt.tight_layout()
plt.show()
```

Linear Discriminant Analysis (LDA)

```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.metrics import classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

# Load dataset
df = pd.read_csv('bank-additional-full.csv', sep=';')

# Separate features and target
X = df.drop(columns=['y'])
y = df['y']

# Encode categorical variables
X_encoded = pd.get_dummies(X, drop_first=True)

# Encode target variable
y_encoded = y.map({'no': 0, 'yes': 1})

# Standardize features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X_encoded)
```

```python
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_encoded, test_size=0.2, random_state=42)

# Apply LDA
lda = LinearDiscriminantAnalysis(n_components=1)
X_lda = lda.fit_transform(X_train, y_train)

# Predict on test set
y_pred = lda.predict(X_test)

# Evaluation
print("Classification Report:")
print(classification_report(y_test, y_pred))

print("Confusion Matrix:")
conf_matrix = confusion_matrix(y_test, y_pred)
print(conf_matrix)

# 2D Visualization using LDA
X_test_lda = lda.transform(X_test)

plt.figure(figsize=(10, 6))
sns.histplot(x=X_test_lda[:, 0], hue=y_test, bins=30, kde=True, palette='Set1', element='step')
plt.title('LDA Component Distribution by Class')
plt.xlabel('LD1')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```

K-Means Clustering

```python
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
import seaborn as sns

# Load dataset
df = pd.read_csv('bank-additional-full.csv', sep=';')

# Drop target variable
X = df.drop(columns=['y'])

# Encode categorical variables
X_encoded = pd.get_dummies(X, drop_first=True)

# Standardize features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X_encoded)

# Determine optimal number of clusters using the Elbow method
inertia = []
K_range = range(1, 11)
for k in K_range:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(X_scaled)
    inertia.append(kmeans.inertia_)
```

```python
# Plot the Elbow Curve
plt.figure(figsize=(8, 5))
plt.plot(K_range, inertia, marker='o', linestyle='--')
plt.title('Elbow Method for Optimal K')
plt.xlabel('Number of Clusters (K)')
plt.ylabel('Inertia')
plt.grid(True)
plt.show()

# Apply K-Means with chosen K (e.g., K=3)
k = 3
kmeans = KMeans(n_clusters=k, random_state=42)
clusters = kmeans.fit_predict(X_scaled)

# Add cluster labels to dataframe
df['Cluster'] = clusters

# Reduce dimensions for visualization
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

# Plot clusters
plt.figure(figsize=(10, 6))
sns.scatterplot(x=X_pca[:, 0], y=X_pca[:, 1], hue=clusters, palette='Set2', s=60, alpha=0.7)
plt.title('K-Means Clustering (Visualized with PCA)')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.legend(title='Cluster')
```

Canonical Correlation Analysis (CCA)

```python
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.cross_decomposition import CCA
import matplotlib.pyplot as plt
import seaborn as sns

# Load dataset
df = pd.read_csv('bank-additional-full.csv', sep=';')

# Define variable groups (you can adjust these sets based on interest)
X_vars = ['age', 'job', 'marital', 'education', 'housing', 'loan']
Y_vars = ['contact', 'month', 'day_of_week', 'duration', 'campaign', 'poutcome']

# Encode categorical variables
df_encoded = pd.get_dummies(df[X_vars + Y_vars], drop_first=True)

# Separate and scale sets
X = df_encoded[[col for col in df_encoded.columns if col.startswith(tuple(X_vars))]]
Y = df_encoded[[col for col in df_encoded.columns if col.startswith(tuple(Y_vars))]]

scaler_X = StandardScaler()
scaler_Y = StandardScaler()
X_scaled = scaler_X.fit_transform(X)
Y_scaled = scaler_Y.fit_transform(Y)

# Apply CCA
cca = CCA(n_components=2)
X_c, Y_c = cca.fit_transform(X_scaled, Y_scaled)
```

```python
# Visualize first canonical components
plt.figure(figsize=(10, 6))
sns.scatterplot(x=X_c[:, 0], y=Y_c[:, 0], alpha=0.5)
plt.title('Canonical Correlation Analysis - First Canonical Variates')
plt.xlabel('First Canonical Variable - Set X')
plt.ylabel('First Canonical Variable - Set Y')
plt.grid(True)
plt.show()

# Optional: Correlation between canonical variables
import numpy as np
correlation = np.corrcoef(X_c[:, 0], Y_c[:, 0])[0, 1]
print(f"Canonical correlation (1st component): {correlation:.3f}")
```