Sirada Boonmarn 6609655353

Clustering analysis - K-mean

This clustering analysis assignment using this data below for finding K-means using python library including pandas, seaborn, matplotlib and sklearn

Table 8.3. Principal component regression for the pitprop data: coefficients, variances, regression coefficients and t-statistics for each component.

	Principal component												
	1	2	3	4	5	6	7	8	9	10	11	12	13
x_1	−0.40	0.22	−0.21	−0.09	−0.08	0.12	−0.11	0.14	0.33	−0.31	0.00	0.39	−0.57
x_2	−0.41	0.19	−0.24	−0.10	−0.11	0.16	−0.08	0.02	0.32	−0.27	−0.05	−0.41	0.58
x_3	−0.12	0.54	0.14	0.08	0.35	−0.28	−0.02	0.00	−0.08	0.06	0.12	0.53	0.41
x_4	−0.17	0.46	0.35	0.05	0.36	−0.05	0.08	−0.02	−0.01	0.10	−0.02	−0.59	−0.38
x_5	−0.06	−0.17	0.48	0.05	0.18	0.63	0.42	−0.01	0.28	−0.00	0.01	0.20	0.12
x_6	−0.28	−0.01	0.48	−0.06	−0.32	0.05	−0.30	0.15	−0.41	−0.10	−0.54	0.08	0.06
x_7	−0.40	−0.19	0.25	−0.07	−0.22	0.00	−0.23	0.01	−0.13	0.19	0.76	−0.04	0.00
x_8	−0.29	−0.19	−0.24	0.29	0.19	−0.06	0.40	0.64	−0.35	−0.08	0.03	−0.05	0.02
x_9	−0.36	0.02	−0.21	0.10	−0.10	0.03	0.40	−0.70	−0.38	−0.06	−0.05	0.05	−0.06
x_{10}	−0.38	−0.25	−0.12	−0.21	0.16	−0.17	0.00	−0.01	0.27	0.71	−0.32	0.06	0.00
x_{11}	0.01	0.21	−0.07	0.80	−0.34	0.18	−0.14	0.01	0.15	0.34	−0.05	0.00	−0.01
x_{12}	0.12	0.34	0.09	−0.30	−0.60	−0.17	0.54	0.21	0.08	0.19	0.05	0.00	0.00
x_{13}	0.11	0.31	−0.33	−0.30	0.08	0.63	−0.16	0.11	−0.38	0.33	0.04	0.01	−0.01
Variance	4.22	2.38	1.88	1.11	0.91	0.82	0.58	0.44	0.35	0.19	0.05	0.04	0.04
% of total variance	32.5	18.3	14.4	8.5	7.0	6.3	4.4	3.4	2.7	1.5	0.4	0.3	0.3
Regression coefficient γ_k	0.13	−0.37	0.13	−0.05	−0.39	0.27	−0.24	−0.17	0.03	0.00	−0.12	−1.05	0.00
t-value	68.6	14.39	4.38	1.26	9.23	6.19	4.50	2.81	0.46	0.00	0.64	5.26	0.01

On the first step we're preparing data for google colab by writing a script to write a csv file in to colab's storage for avoiding data losses when runtime get reset

```
csv_data = """Variable,PC1,PC2,PC3,PC4,PC5,PC6,PC7,PC8,PC9,PC10,PC11,PC12,PC13
x1,-0.40,0.22,-0.21,-0.09,-0.08,0.12,-0.11,0.14,0.33,-0.31,0.00,0.39,-0.57
x2,-0.41,0.19,-0.24,-0.10,-0.11,0.16,-0.08,0.02,0.32,-0.27,-0.05,-0.41,0.58
x3,-0.12,0.54,0.14,0.08,0.35,-0.28,-0.02,0.00,-0.08,0.06,0.12,0.53,0.41
x4,-0.17,0.46,0.35,0.05,0.36,-0.05,0.08,-0.02,-0.01,0.10,-0.02,-0.59,-0.38
x5,-0.06,-0.17,0.48,0.05,0.18,0.63,0.42,-0.01,0.28,-0.00,0.01,0.20,0.12
x6,-0.28,-0.01,0.48,-0.06,-0.32,0.05,-0.30,0.15,-0.41,-0.10,-0.54,0.08,0.06
x7,-0.40,-0.19,0.25,-0.07,-0.22,0.00,-0.23,0.01,-0.13,0.19,0.76,-0.04,0.00
x8,-0.29,-0.19,-0.24,0.29,0.19,-0.06,0.40,0.64,-0.35,-0.08,0.03,-0.05,0.02
x9,-0.36,0.02,-0.21,0.10,-0.10,0.03,0.40,-0.70,-0.38,-0.06,-0.05,0.05,-0.06
x10,-0.38,-0.25,-0.12,-0.21,0.16,-0.17,0.00,-0.01,0.27,0.71,-0.32,0.06,0.00
x11,0.01,0.21,-0.07,0.80,-0.34,0.18,-0.14,0.01,0.15,0.34,-0.05,0.00,-0.01
x12,0.12,0.34,0.09,-0.30,-0.60,-0.17,0.54,0.21,0.08,0.19,0.05,0.00,0.00
x13,0.11,0.31,-0.33,-0.30,0.08,0.63,-0.16,0.11,-0.38,0.33,0.04,0.01,-0.01"""

# Write the CSV data to a file
with open("data.csv", "w") as file:
    file.write(csv_data)
```

After that we start to import and analyze data before performing data calculation

```
# Import libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler

# Load the data
data = pd.read_csv("data.csv")

# Remove the 'Variable' column
X = data.drop(columns=['Variable'])

print(data.describe())
print(data.info())
```

Data calculations

- **Pair Plot**

```
#Pair Plot
sns.pairplot(data)
plt.show()
```

- visualizes pairwise relationships between all numerical columns in the dataset.
- Helps to identify patterns, trends, and correlations that may influence clustering results.

- **K-means Clustering**

```
# Scale the data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

inertia = []
k_range = range(1, 10)  # Test cluster sizes from 1 to 9
for k in k_range:
    kmeans = KMeans(n_clusters=k, random_state=42, n_init=10)
    kmeans.fit(X_scaled)
    inertia.append(kmeans.inertia_)
    cluster = kmeans.fit_predict(X_scaled)
    plt.figure(figsize=(8, 6))
    plt.scatter(X_scaled[:, 0], X_scaled[:, 1], c=cluster, cmap='viridis', s=50)
    plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s=200, c='red', marker='X')
    plt.title('K-Means Clustering (PC1 vs PC2)' + ' k = ' + str(k))
    plt.xlabel('PC1')
    plt.ylabel('PC2')
    plt.show()
```

1. Data Scaling

- Standardization ensures all variables have the same scale (mean = 0, standard deviation = 1).
- K-Means is sensitive to the scale of data; therefore, scaling is a necessary preprocessing step.

2. K-Means Clustering for Different k values

- Visualize each k values to a graph

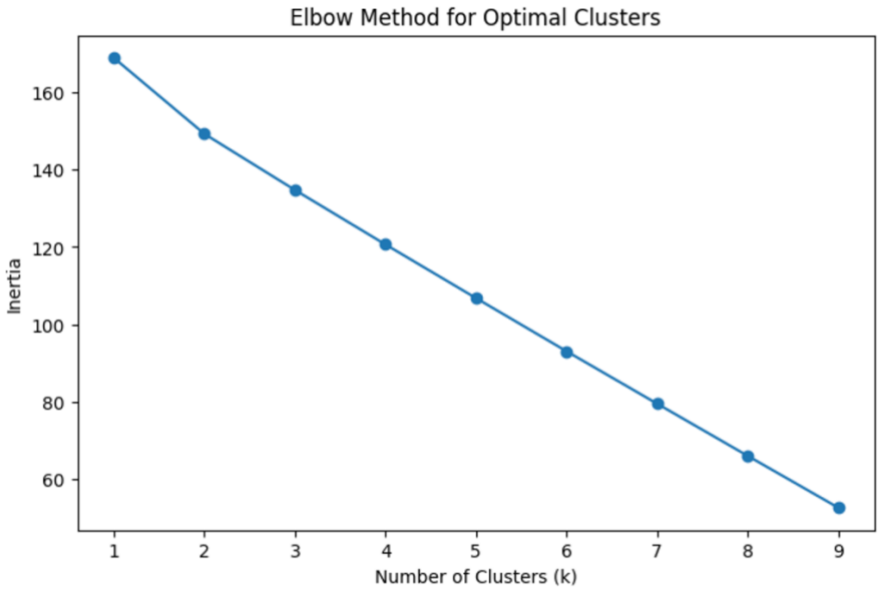
- **Elbow Method to Determine Optimal k**

```
# Plot the Elbow Curve
plt.figure(figsize=(8, 5))
plt.plot(k_range, inertia, marker='o')
plt.xlabel('Number of Clusters (k)')
plt.ylabel('Inertia')
plt.title('Elbow Method for Optimal Clusters')
plt.show()
```

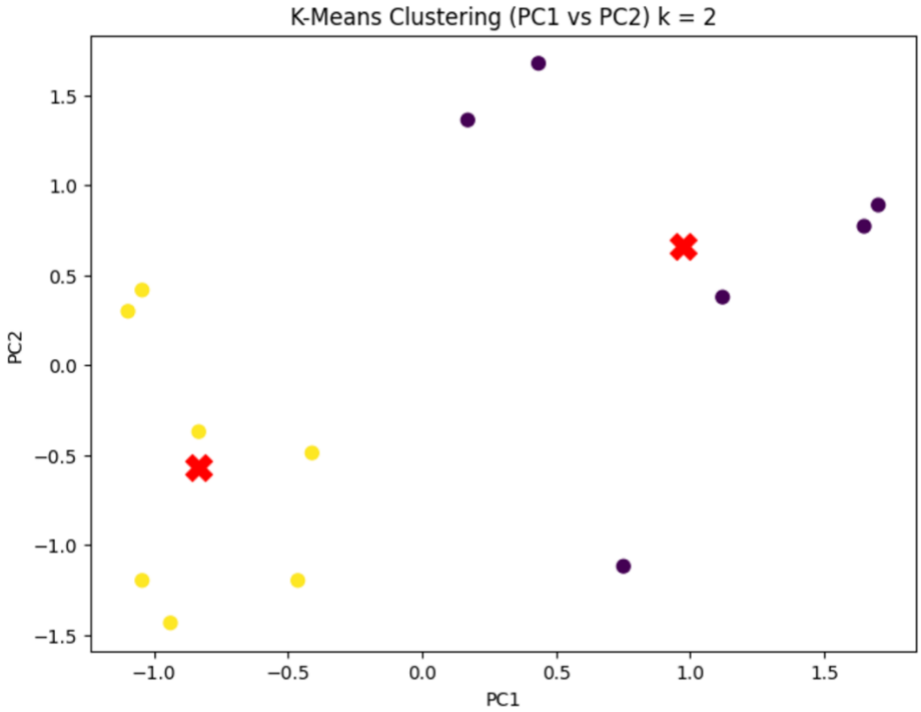
- The Elbow Method helps identify the optimal number of clusters.
- Inertia is plotted against k values:
  - Initially, inertia decreases rapidly as k increases.
  - At a certain point (the “elbow”), the decrease slows down, indicating the optimal number of clusters.
- The “elbow” point is the value of k where adding more clusters does not significantly reduce inertia.

Results

Elbow Curve



From this elbow curve result as “At a certain point, the decrease slows down, indicating the optimal number of clusters”. We can see that at k = 2 is where the decrease slow down. So, we choose 2 as an optimal k value



Graph of K = 2 K-means clustering result