Lab MCA: Multiple Correspondence Analysis (MCA) using prince on a dataset with categorical features

Lab Duration: 2-3 Hours
Prerequisites: Basic understanding of Python, Pandas, Matplotlib, and Multiple Correspondence Analysis

---

Lab Objectives

By the end of this lab, students will be able to:

1. Perform MCA using the prince library.
2. Show and interpret the eigenvalues, explained variance and cumulative explained variance.
3. Get the new coordinates for the rows (Individuals) and columns (Modalities) and show the data in the new reduced space.
4. Get and interpret the column/row contributions to the total explained variance of each dimension.
5. Know the quality of representation of each point in the factor map.

Lab Outline

Part 1: Loading Data

- Loading the **Balloons** dataset.

Part 2: Performing MCA using prince

- Initializing the prince MCA model.
- Fitting the model to the data.
- Getting the explained variance
- Transforming the data and getting the new columns and rows' coordinates

Part 3: Visualization & Interpretation

- Visualizing the data in the new dimensions (a 2-D space).
- Display rows and columns contributions.
- Display the quality of representation of each data point on the factor maps (cosine similarities)
- Discussing the insights obtained from the plots.

Part 4: Conclusion and Q&A

- Summarize key points.

The Ballons Dataset

- The dataset is related to a classification task where the goal is to predict whether a given combination of features about balloons leads to a "happy" or "unhappy" outcome.
- The dataset consists of categorical features, each describing a characteristic related to the balloons:
    - Color: The color of the balloon (e.g., Yellow, Purple).
    - Size: The size of the balloon (e.g., Small, Large).
    - Act: Describes the action or state (e.g., Stretch, Dip).
    - Age: The age group involved (e.g., Adult, Child).
- Class (Target):
    - The target variable is "Inflated", indicating whether the balloon's state is "True" (happy) or "False" (unhappy).
- Dataset Size: The dataset is quite small, containing only 16 instances. Each instance corresponds to a unique combination of the categorical features.

Part 1: Loading the Data

Step 1: Install Required Libraries (if not yet installed)

Step 2: Load the Dataset and display its first 5 rows

```
import pandas as pd

dataset = pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/balloons/adult+stretch.data')
dataset.columns = ['Color', 'Size', 'Action', 'Age', 'Inflated']
dataset.head()
```

Question: what do you think about the format of “dataset” ? Is it one-hot encoded?

Part 2: Performing MCA using prince

Step 1: Initialize the MCA Model

- Initialize MCA with 2 components:

```python
import prince

mca = prince.MCA(
    n_components=2,
    n_iter=3,
    copy=True,
    check_input=True,
    engine='sklearn',
    random_state=42
)
```

Q. Explain the main parameters ?

Step 2: Fit the MCA Model

If your data is not yet one-hot encoded (is not an indicator matrix):

- Fit the MCA model to the dataset:

```python
mca = mca.fit(dataset) #MCA one-hot encodes the data.
```

If your data is already one-hot encoded, use the following:

```python
##If your dataset is already one-hot-encoded, use the following instead:
#one_hot = pd.get_dummies(dataset)
#mca_no_one_hot = prince.MCA(one_hot=False)
#mca_no_one_hot = mca_no_one_hot.fit(one_hot)
```

- Print eigenvalues, explained variance and total inertia of mca

```python
#Get the eignevalues

mca.eigenvalues_summary
```

Q. How is the percentage of explained variance calculated for each component?

Step 3: Transforming the data and getting the new columns and rows' coordinates

Get the rows (individuals) and columns (modalities) coordinates.

Note that there is no "mca.transform()" as mca.row_coordinates() and mca.column_coordinates() already do the transformation.

```
# Column (variable) Coordinates
mca.column_coordinates(dataset)
```

```
#Row (individual) Coordinates
mca.row_coordinates(dataset).head()
```

Part 3: Visualization & Interpretation

We can use the plot() function of prince library or use matplotlib (as we did in the PCA lab):

- **1st visualization: Sow the row (individuals) and column (modalities) on the factor maps:**

```
#Visualization
mca.plot(
    dataset,
    x_component=0,
    y_component=1,
    show_column_markers=True,
    show_row_markers=True,
    show_column_labels=False,
    show_row_labels=False
)
```

Q1. Explain the different parameters?

Q2. What do these factor maps show?

- **2nd visualization:** Change the above code to only show the modalities on the factor map ?

- **Displaying the contributions of the modalities (columns) to the 1st dimension in a barplot:**

```
import matplotlib.pyplot as plt
import seaborn as sns

column_contributions = mca.column_contributions_

# Convert the column contributions to a DataFrame for easy plotting
contrib_df = column_contributions.iloc[:, 0]  # Use first dimension (0-based index)

# Plot the bar chart
plt.figure(figsize=(10, 6))
sns.barplot(x=contrib_df.index, y=contrib_df.values)
plt.title('Column Contributions to the 1st Dimension')
plt.xlabel('Columns (Modalities)')
plt.ylabel('Contribution')
plt.xticks(rotation=45, ha='right')  # Rotate labels for readability
plt.tight_layout()
plt.show()
```

- **Displaying the contributions of the individuals (rows) to the 1st dimension in a barplot.**
  Change the above code.

- **Getting the quality representation of each individual in the 1st dimension:**
  - **Cosine similarities of rows :**

```
# Get the quality of the representation of each point in the reduced space. Higher values indicate that a point is well-represented on
#the selected dimensions.
mca.row_cosine_similarities(dataset)
```

  - **Visualize the cosine similarities of rows in a barplot:**

```python
import matplotlib.pyplot as plt
import seaborn as sns

col_cos = mca.row_cosine_similarities(dataset)


col_cos_df = col_cos.iloc[:, 0]  # Use first dimension (0-based index)

# Plot the bar chart
plt.figure(figsize=(10, 6))
sns.barplot(x=col_cos_df.index, y=col_cos_df.values)
plt.title('Row Cosine similarities - First dimension')
plt.xlabel('Rows (Individuals)')
plt.ylabel('Cosine similarities')
plt.xticks(rotation=45, ha='right')  # Rotate labels for readability
plt.tight_layout()
plt.show()
```

- **Do the same for the quality representation of each modality in the 1st dimension.**

Part 4: Conclusions & QA

- Discuss how much variance is explained by the first two dimensions and the importance of each dimension.
- Discuss how columns (modalities) and rows (individuals) contribute to the factors and what the plot tells us about the relationship about the different variables.
