KMeans_Clstering_Credit_ (1)

April 14, 2025

0.0.1 Problem Statement:

A credit card company aims to optimize its marketing strategy for new credit card products in Uganda. To achieve this, the company seeks to segment its potential customer base into distinct groups based on financial behavior, demographics, and risk profiles. This project will employ unsupervised learning techniques, specifically K-Means and hierarchical clustering, to identify these segments. The analysis will:

- Characterize each segment through descriptive statistics and visualizations of key customer attributes.
- Develop customer personas that go beyond numerical descriptions, providing actionable insights into the lifestyles and financial needs of each segment.
- Generate targeted marketing and product recommendations for each segment, enabling the credit card company to tailor its offerings and communication for maximum effectiveness and profitability.

0.0.2 Summary Conclusions and Recommendations

Cluster 1:

- **Target Audience**: This cluster, representing the majority (63.9%) of the customer base, appears to be composed primarily of **younger individuals** who are likely in the **early stages of their careers** indicated by their lowest reported income and fewest years of employment.
- **Financial Behavior**: Despite their lower income, they exhibit fairly good financial behavior, demonstrated by their lowest levels of both credit card debt and other forms of debt. Their moderate Debt-to-Income Ratio (DTI) suggests they are managing their existing financial obligations responsibly relative to their income.
- **Segment Description** This segment likely includes recent graduates, entry-level professionals, or individuals in the initial phase of building their financial stability.
- **Recommendation**:
- Given their growth potential, the credit card company should focus on nurturing long-term relationships with this segment by offering cards that scale with career or business progression.
- This could include reward programs that focus on career milestones (e.g., bonuses for first-time cardholders, promotions, home or car or students loans).
- As they grow in income, the company could gradually offer premium cards with more benefits, positioning itself as a financial partner throughout their career trajectory.

Cluster 0 (Green):

- **Target Audience**: This cluster represents a segment of **older**, more **established individuals** characterized by the **highest reported income** and a **significant number of years employed**. Holds the second-largest share with 20.6% of the data points.
- **Financial Behavior**: While they carry moderate levels of both credit card debt and other debt, their Debt-to-Income Ratio (DTI) is the lowest among all clusters. This indicates a strong ability to manage their financial obligations relative to their high income. Notably, this group also exhibits the lowest propensity for defaulting on their payments.
- **Segment Description**: This segment likely comprises seasoned professionals, established in their careers, potentially nearing or in retirement.
- **Recommendations**:
- Higher credit limits: Reflecting their high income and low risk of default.
- Offer dedicated customer service or relationship managers to provide a higher level of personalized attention. Easy since they are the least group.
- Introduce investment-linked credit card features or financial planning tools, such as integration with wealth management services, retirement savings support, or cashback on investment-related spending.
- Roll out exclusive, invite-only events or networking experiences that tap into their professional status and interests like investment summits, golf tournaments, or business roundtables.

Cluster 2 (Red):

- **Target Audience**: This cluster represents the **smallest segment** of the customer base, accounting for 15.6% of the total. Individuals in this group are characterized by being in the **middle age** range and having a **moderate income**.
- **Financial Behavior**: They carry the **highest levels** of both credit card debt and other forms of debt, resulting in the highest Debt-to-Income Ratio (DTI) among all clusters. They have a **moderate number of years employed**, suggesting they are likely established in their careers but may be facing significant financial burdens.
- **Alarmingly**, the number of individuals who have defaulted on payments is greater than those who have not within this cluster, indicating a high-risk segment.
- **Recommendations**:
- Lower Credit Limits: For existing cardholders in this segment, consider gradually lowering credit limits to reduce losses.
- Stricter Monitoring: Implement more frequent monitoring of their payment behavior and credit utilization.
- Cautious Approach to New Credit: Be extremely selective and conservative when considering offering new credit products to individuals with profiles similar to this cluster. Thorough risk assessment is crucial.
- Avoid Aggressive Marketing of New Debt: Refrain from marketing new credit cards or increased credit limits to this segment. Communication should focus on providing solutions

and support rather than encouraging further borrowing.

- Secured Loans: Explore the possibility of offering secured loans.

[1]:
```
# Importing Libraries
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import OneHotEncoder, LabelEncoder, \
OrdinalEncoder, StandardScaler, MinMaxScaler
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.metrics import classification_report, confusion_matrix, \
accuracy_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
from sklearn.preprocessing import RobustScaler
from scipy.cluster.hierarchy import dendrogram, linkage, fcluster
from sklearn.model_selection import GridSearchCV
from sklearn.tree import plot_tree
from sklearn.linear_model import LogisticRegression
```

[2]:
```
# Reading the customer seg data
from google.colab import auth
auth.authenticate_user()
import gspread
from google.auth import default
creds, _ = default()
gc = gspread.authorize(creds)
worksheet = gc.open('Copy of cust_seg[Task 5]').sheet1
# get_all_values gives a list of rows.
rows = worksheet.get_all_values()
# Convert to a DataFrame and render.
import pandas as pd
seg = pd.DataFrame.from_records(rows)
```

[3]:
```
# first row as columns
seg.columns = seg.iloc[0]
# resetting and dropping old index
seg = seg.iloc[1:].reset_index(drop=True)
```

```
seg.head()
```

```
[3]: 0    Customer Id Age Edu Years Employed Income Card Debt Other Debt Defaulted  \
     0  0            1  41   2              6     19     0.124      1.073         0
     1  1            2  47   1             26    100     4.582      8.218         0
     2  2            3  33   2             10     57     6.111      5.802         1
     3  3            4  29   2              4     19     0.681      0.516         0
     4  4            5  47   1             31    253     9.308      8.908         0

     0 DebtIncomeRatio
     0             6.3
     1            12.8
     2            20.9
     3             6.3
     4             7.2
```

```
[4]: """
     Eliminating the first 2 columns becasue they are just unique row identifiers
     and do not have any relevant information
     """
     seg_data = seg.drop(columns = ['', 'Customer Id'])
     seg_data.head()
```

```
[4]: 0 Age Edu Years Employed Income Card Debt Other Debt Defaulted DebtIncomeRatio
     0  41   2              6     19     0.124      1.073         0             6.3
     1  47   1             26    100     4.582      8.218         0            12.8
     2  33   2             10     57     6.111      5.802         1            20.9
     3  29   2              4     19     0.681      0.516         0             6.3
     4  47   1             31    253     9.308      8.908         0             7.2
```

```
[5]: data = seg_data.copy()
```

```
[6]: # missing values in seg data
     seg_data.isna().sum()
```

```
[6]: 0
     Age                0
     Edu                0
     Years Employed     0
     Income             0
     Card Debt          0
     Other Debt         0
     Defaulted          0
     DebtIncomeRatio    0
     dtype: int64
```

[7]:
```
# '' in seg data
for col in seg_data.columns:
  print(col, (seg_data[col] == '').sum())
```

```
Age 0
Edu 0
Years Employed 0
Income 0
Card Debt 0
Other Debt 0
Defaulted 150
DebtIncomeRatio 0
```

[8]:
```
seg_data.shape
```

[8]: (850, 8)

[9]:
```
# Removing the rows with empty strings
seg_data = seg_data[seg_data['Defaulted'] != '']
```

Transforming the data

[10]:
```
seg_data.head(2)
```

[10]:

0	Age	Edu	Years Employed	Income	Card Debt	Other Debt	Defaulted	DebtIncomeRatio
0	41	2	6	19	0.124	1.073	0	6.3
1	47	1	26	100	4.582	8.218	0	12.8

[11]:
```
# Appling ordinal encoder on ordinal columns
ordinal_encoder = OrdinalEncoder()
seg_data['Edu'] = ordinal_encoder.fit_transform(seg_data[['Edu']])
```

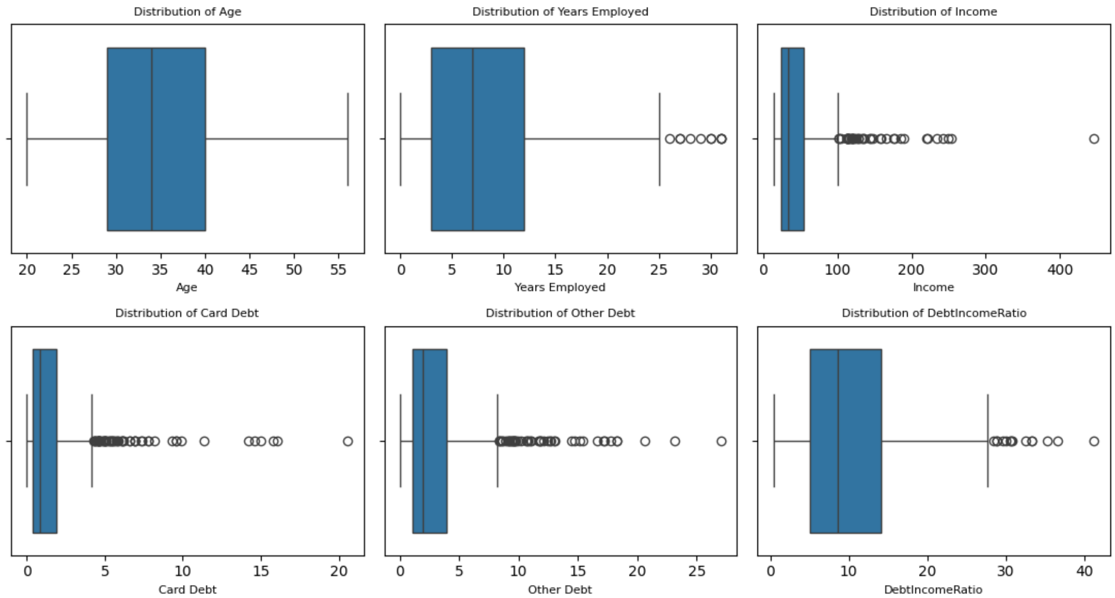
[12]:
```
# Appling lebale encoder on catgeorical columns
le = LabelEncoder()
seg_data['Defaulted'] = le.fit_transform(seg_data['Defaulted'])
```

[13]:
```
# Appling standard scaler on the numerical features
num = ['Age', 'Years Employed', 'Income', 'Card Debt', 'Other Debt',␣
↪'DebtIncomeRatio']
```

[14]:
```
# changing data types in num to float
seg_data[num] = seg_data[num].astype(float)
```

[15]:
```
# Distribution of the num features
fig, axes = plt.subplots(nrows=2, ncols=3, figsize=(11, 6))
# Flatten axes for easy iteration
axes = axes.flatten()
```

```
for i, col in enumerate(num):
    sns.boxplot(data=seg_data, x=col, ax=axes[i])
    axes[i].set_title(f'Distribution of {col}', fontsize=8)
    axes[i].set_xlabel(col, fontsize = 8)
plt.tight_layout()
plt.show()
```



Insights

1. Age - Relatively left skewed with a median around 34. Indicates that most of the customers are of the middle age group.
2. Years Employed - Highly left skewed. Indicating that most of the customers have.
3. Income - Highly left skewed. Most individuals having lower incomes while a few have extremely high incomes.
4. Card Debt - Highly left skewd, with the majority having very low card debt and a few with significantly higher debt.
5. Other Debt - Similar to the distribution of card debt.
6. Debt Income Ration - Left Skewed with a few ouliers with high DIR. These customers are risky.

Due to the presence of skewness in most of the numerical features, standard scaler or Robust Scaler is favorable for scaling the num feature unlike MinMax Scaler which can be affected by the presence of outliers.

[16]:
```
# Appling scaling to the num features
scaler = StandardScaler()
seg_data[num] = scaler.fit_transform(seg_data[num])
```

[17]:
```
# Displaying seg_data
seg_data.head()
```

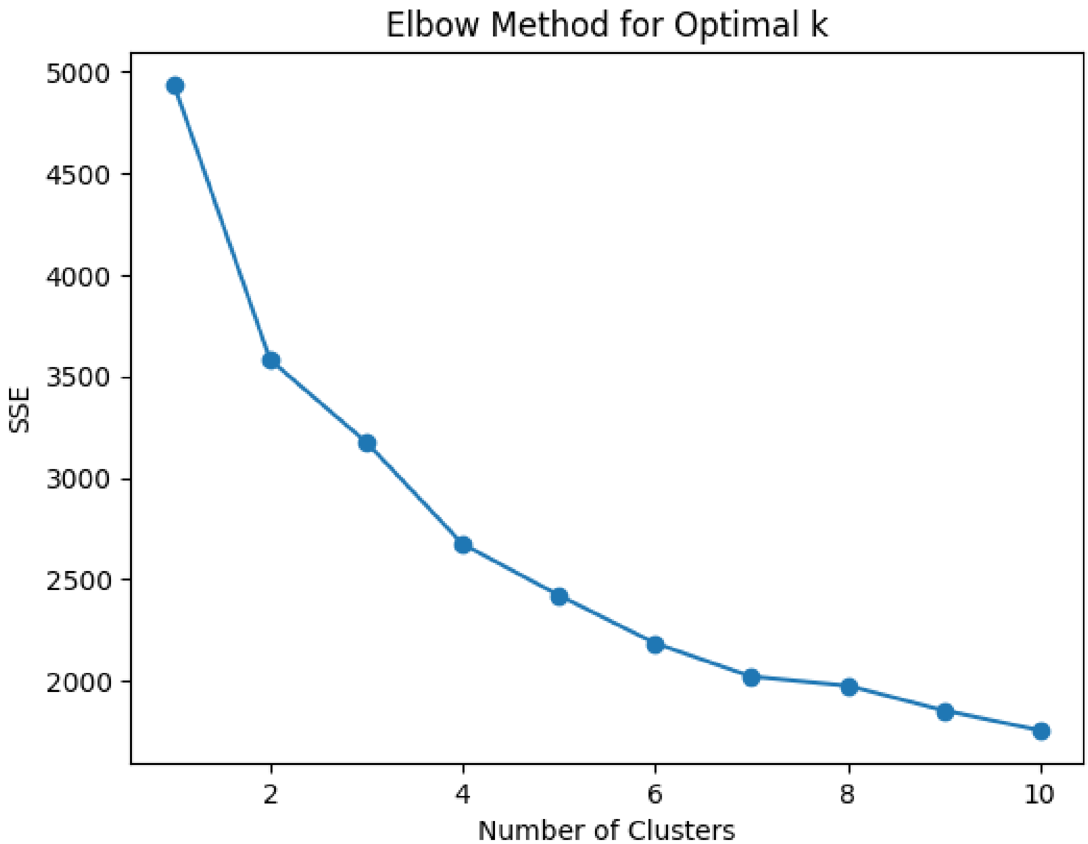
[17]:
```
0        Age  Edu  Years Employed    Income  Card Debt  Other Debt  Defaulted  \
0  0.768304  1.0       -0.359007 -0.723102  -0.675699   -0.604284          0
1  1.519090  0.0        2.647029  1.478707   1.431421    1.570620          0
2 -0.232744  1.0        0.242201  0.309845   2.154119    0.835201          1
3 -0.733267  1.0       -0.659610 -0.723102  -0.412427   -0.773833          0
4  1.519090  0.0        3.398538  5.637681   3.665215    1.780653          0

0  DebtIncomeRatio
0        -0.580528
1         0.372222
2         1.559495
3        -0.580528
4        -0.448609
```

K-Means Clustering

[18]:
```
# Finding optimal k using elblow method
sse = []
for k in range(1,11):
  kmeans = KMeans(n_clusters=k, random_state = 42)
  kmeans.fit(seg_data)
  sse.append(kmeans.inertia_)


# Elbow Plot
plt.plot(range(1,11), sse, marker='o')
plt.title('Elbow Method for Optimal k')
plt.xlabel('Number of Clusters')
plt.ylabel('SSE')
plt.show()
```



Explanantionf the Elbow plot

- The Elbow Method graph plots the Number of Clusters on the x-axis against the SSE (Within-Cluster Sum of Squared Errors) on the y-axis, showing how the error decreases as more clusters are formed.
- The optimal number of clusters is suggested to be 3, as this is where the significant decrease in SSE begins to plateau, forming an "elbow". This indicates adding more clusters yields diminishing returns in terms of reducing within-cluster variance.

[19]:
```
km = KMeans(n_clusters=3, random_state=42)

# Training the data
seg_trained = km.fit(seg_data)

# Predicting
seg_pred = seg_trained.predict(seg_data)
```

[20]:
```
# Adding seg_pred to seg_transformed
seg_data['Cluster'] = seg_pred
seg_data.head()
```

[20]:
```
0       Age  Edu  Years Employed    Income  Card Debt  Other Debt  Defaulted  \
0  0.768304  1.0       -0.359007 -0.723102  -0.675699   -0.604284          0
1  1.519090  0.0        2.647029  1.478707   1.431421    1.570620          0
2 -0.232744  1.0        0.242201  0.309845   2.154119    0.835201          1
3 -0.733267  1.0       -0.659610 -0.723102  -0.412427   -0.773833          0
4  1.519090  0.0        3.398538  5.637681   3.665215    1.780653          0

0  DebtIncomeRatio  Cluster
0        -0.580528        1
1         0.372222        0
2         1.559495        2
3        -0.580528        1
4        -0.448609        0
```

[21]:
```
# cluster centers
centers = km.cluster_centers_
centers
```

[21]:
```
array([[ 1.05158663,  0.73611111,  1.31518822,  1.1368455 ,  0.3321607 ,
         0.39135902,  0.06944444, -0.34671788],
       [-0.40154439,  0.6689038 , -0.46559636, -0.42871275, -0.39006644,
        -0.4439892 ,  0.24384787, -0.28980113],
       [ 0.25744833,  0.9266055 ,  0.17187589,  0.25622797,  1.16081246,
         1.30373829,  0.58715596,  1.6464998 ]])
```

[22]:
```
seg_data.dtypes
```

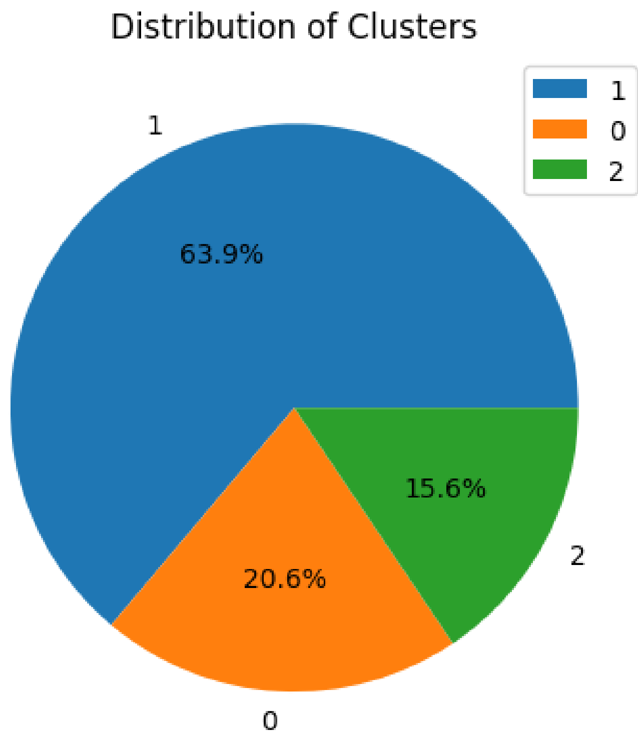
[22]:
```
0
Age                  float64
Edu                  float64
Years Employed       float64
Income               float64
Card Debt            float64
Other Debt           float64
Defaulted              int64
DebtIncomeRatio      float64
Cluster                int32
dtype: object
```

Creating analytical visualizations that explore statistics for each feature for each cluster. Distribution of Clusters

[37]:
```
# counts of each cluster
seg_data['Cluster'].value_counts()

# pie charch sowing districution of clusters
```

```
plt.pie(seg_data['Cluster'].value_counts(), labels=seg_data['Cluster'].
    ↪value_counts().index, autopct='%1.1f%%')
plt.title('Distribution of Clusters')
plt.legend()
plt.show()
```



Insights

The provided pie chart illustrates the distribution of data points across the three distinct clusters, labeled 0, 1, and 2. - Cluster 1 represents the largest proportion of the data, accounting for 63.9% of the total. - Cluster 0 holds the second-largest share with 20.6% of the data points. - Finally, Cluster 2 contains the smallest proportion of the data, making up 15.6% of the total distribution. - This visualization clearly shows that Cluster 1 is the dominant group within the dataset, while Clusters 0 and 2 represent smaller subgroups.

Age and Income

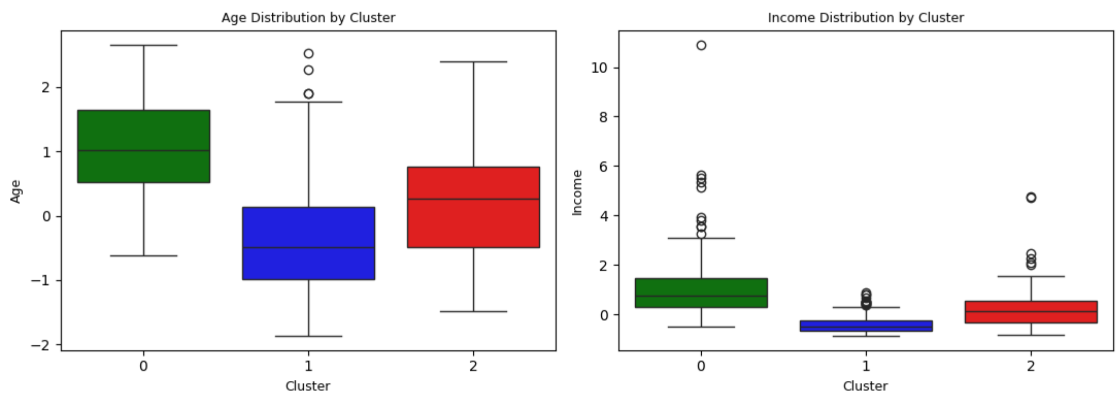
[23]:
```
# Distinct colors for clusters
cluster_colors = ['green', 'blue', 'red', 'yellow', 'black']

# Create subplot
fig, axes = plt.subplots(1, 2, figsize=(11, 4))
```

```python
# Boxplot for Age
sns.boxplot(data=seg_data, x='Cluster', y='Age', palette=cluster_colors,
    ax=axes[0])
axes[0].set_title('Age Distribution by Cluster', fontsize = 9)
axes[0].set_xlabel('Cluster', fontsize = 9)
axes[0].set_ylabel('Age', fontsize = 9)

# Boxplot for Income
sns.boxplot(data=seg_data, x='Cluster', y='Income', palette=cluster_colors,
    ax=axes[1])
axes[1].set_title('Income Distribution by Cluster', fontsize = 9)
axes[1].set_xlabel('Cluster', fontsize = 9)
axes[1].set_ylabel('Income', fontsize = 9)

plt.tight_layout()
plt.show()
```



Insights Age

1. Cluster 1 (Blue): This segment clearly consists of the youngest customers among the three clusters. The median age is significantly lower, and the entire distribution is shifted towards younger individuals with fewer older outliers.
2. Cluster 0 (Green): This segment represents the oldest customer group. The median age is the highest.
3. Cluster 2 (Red): This segment represents customers with a relatively wide range of ages, centered around a slightly younger median age. There are some notably older outliers in this group. This is the middle age group as the median is between the median of the lowest age group and that of the the highest.

Income

1. Cluster 0 (Green): This is the high-income group. It has the highest median income and a wider income range compared to the other clusters. There are several high-income outliers, more spread out, indicating a broader range of higher earnings.

2. Cluster 1 (Blue): This group represents the low-income customers. The median income is the lowest, and the range is concentrated at the lower end. Even the outliers are closely packed, showing less variation and a generally low-income distribution.
3. Cluster 2 (Red): This is the moderate-income group. The median income is in between the other two clusters. It includes a few individuals with slightly higher incomes than those in cluster 1.

Card Debt and Other Debt

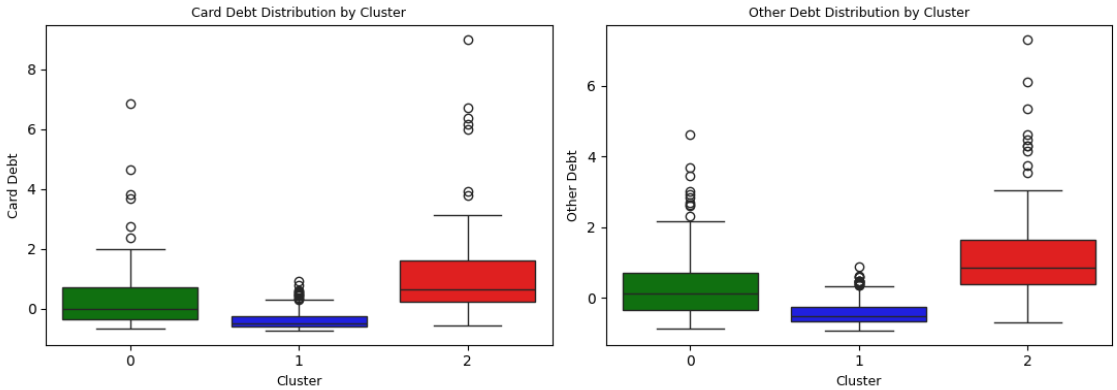
[24]:
```
# Distinct colors for clusters
cluster_colors = ['green', 'blue', 'red', 'yellow', 'black']

# Create subplot
fig, axes = plt.subplots(1, 2, figsize=(11, 4))

# Boxplot for Age
sns.boxplot(data=seg_data, x='Cluster', y='Card Debt', palette=cluster_colors,␣
↪ax=axes[0])
axes[0].set_title('Card Debt Distribution by Cluster', fontsize = 9)
axes[0].set_xlabel('Cluster', fontsize = 9)
axes[0].set_ylabel('Card Debt', fontsize = 9)

# Boxplot for Income
sns.boxplot(data=seg_data, x='Cluster', y='Other Debt', palette=cluster_colors,␣
↪ax=axes[1])
axes[1].set_title('Other Debt Distribution by Cluster', fontsize = 9)
axes[1].set_xlabel('Cluster', fontsize = 9)
axes[1].set_ylabel('Other Debt', fontsize = 9)

plt.tight_layout()
plt.show()
```



Insights Card Debt

1. Cluster 1 (Blue): Minimal reliance on credit cards and lower risk. This segment exhibits the lowest levels of card debt. The box plot is highyleft skewed, and the overall range is small, with only a few very low outliers campared to the other figures.
2. Cluster 0 (Green): Moderate Risk group. This segment shows slightly higher card debt compared to Cluster 1, but still relatively low overall. The median is still low, but the upper quartile extends a bit further, and there are a few more noticeable outliers with higher card debt.
3. Cluster 2 (Red): High Risk group. This segment has significantly higher levels of card debt compared to Clusters 0 and 1. The median card debt is the highest, and the distribution has a much wider range, with several high outliers.

Other Debt

1. Cluster 1 (Blue): This segment has the lowest levels of "other debt". The median is low, and the range is relatively small with a few low outliers.
2. Cluster 0 (Green): Similar to card debt, this segment shows slightly higher levels of other debt compared to Cluster 0, but still generally low. The median is a bit higher, and there are with a few tightly clustered outliers with higher other debt.
3. Cluster 2 (Red): High Risk Group. This segment also exhibits the highest levels of "other debt" among the three clusters. The median is significantly higher, and the distribution has a wide range with several high spread out outliers unlike that of cluster 1 and 0. These dispersed outliers suggest a diverse set of individuals with extreme debt levels, signaling increased financial risk and the need for targeted intervention or risk mitigation strategies.

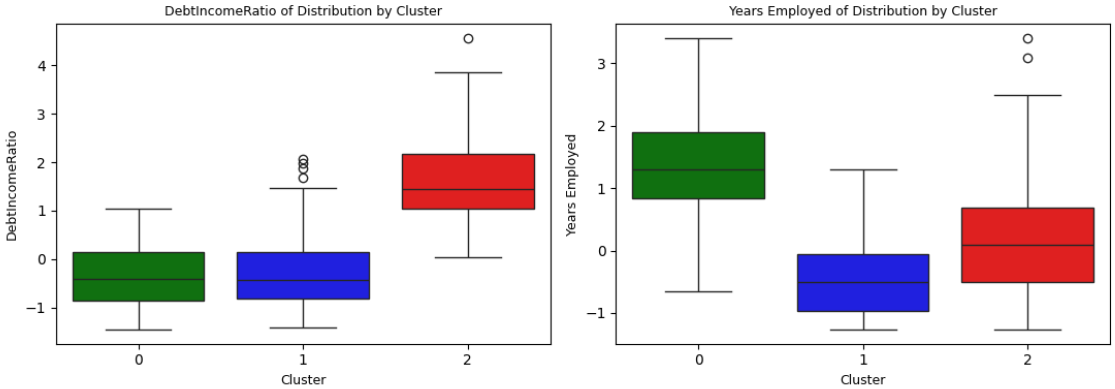
DebtIncomeRatio and Years Employed

[25]:
```python
# Create subplot
fig, axes = plt.subplots(1, 2, figsize=(11, 4))

# Boxplot for Age
sns.boxplot(data=seg_data, x='Cluster', y='DebtIncomeRatio',␣
 ↪palette=cluster_colors, ax=axes[0])
axes[0].set_title('DebtIncomeRatio of Distribution by Cluster', fontsize = 9)
axes[0].set_xlabel('Cluster', fontsize = 9)
axes[0].set_ylabel('DebtIncomeRatio', fontsize = 9)

# Boxplot for Income
sns.boxplot(data=seg_data, x='Cluster', y='Years Employed',␣
 ↪palette=cluster_colors, ax=axes[1])
axes[1].set_title('Years Employed of Distribution by Cluster', fontsize = 9)
axes[1].set_xlabel('Cluster', fontsize = 9)
axes[1].set_ylabel('Years Employed', fontsize = 9)

plt.tight_layout()
plt.show()
```



Insights Debt Income Ration

1. Cluster 0 (Green): This segment exhibits the lowest Debt-to-Income Ratio (DIR). The median value is the least and the overall range is also the lowest.
2. Cluster 2 (Red): This segment has the highest Debt-to-Income Ratio. The median DIR the highest among the three clusters, with a significant spread and several high outliers, indicating a group with a higher proportion of their income going towards debt payments.
3. Cluster 1 (Blue): This segment shows a moderate Debt-to-Income Ratio. The meadian DIR is almost similar to that of cluster 2 expect that it has a few tightly clustered outliers, individuals with higher DIR which are not in cluster 0.

Years Employed

1. Cluster 2 (Red): This segment has a moderate range of years employed, centered around a relatively low median. This might indicate a mix of early to mid-career professionals.
2. Cluster 1 (Blue): This segment shows the lowest number of years employed, with the lowest median. This could be a group of relatively new entrants to the workforce or those with unstable employment history.
3. Cluster 0 (Green): This segment exhibits the highest number of years employed, with a higher median and a wider range, including individuals with many years of employment. This likely represents more established professionals.

Education Level and Default Rate

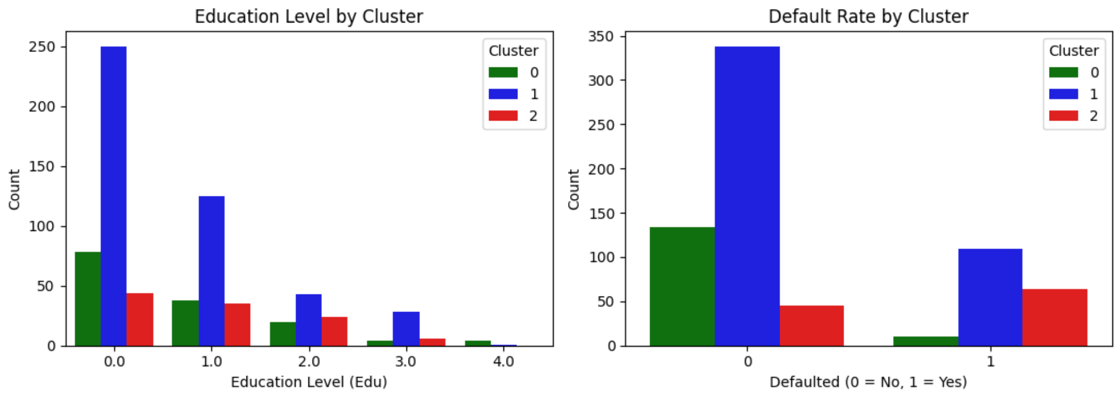
[26]:
```
# Create subplot
fig, axes = plt.subplots(1, 2, figsize=(11, 4))

# Count plot for Education Level
sns.countplot(data=seg_data, x='Edu', hue='Cluster', palette=cluster_colors,␣
 ↪ax=axes[0])
axes[0].set_title('Education Level by Cluster')
axes[0].set_xlabel('Education Level (Edu)')
axes[0].set_ylabel('Count')

# Count plot for Defaulted
```

```
sns.countplot(data=seg_data, x='Defaulted', hue='Cluster',␣
↪palette=cluster_colors, ax=axes[1])
axes[1].set_title('Default Rate by Cluster')
axes[1].set_xlabel('Defaulted (0 = No, 1 = Yes)')
axes[1].set_ylabel('Count')

# Adjust layout
plt.tight_layout()
plt.show()
```



Insights Education Level

1. Cluster 0 (Green): Has a significant number of individuals at education level 0.0, a smaller number at 1.0, and even fewer at higher levels.
2. Cluster 1 (Blue): Is heavily concentrated at education level 0.0. Its concentration reduces are the the education level goes higher.
3. Cluster 2 (Red): While it does have individuals at education levels 2.0 and 3.0, the most significant proportion of individuals in Cluster 2 is actually at the lowest education level (0.0). The counts at levels 2.0 and 3.0 are present but are smaller than the count at level 0.0. **Overall Insight:**
   Most customers across all clusters fall under **lower education levels**, with **education level 0.0 being the most dominant**.

Defaulted

1. Cluster 1 (Blue): Has a higher number of non-defaulters compared to defaulters.
2. Cluster 1 (Blue): Shows a significantly higher number of non-defaulters compared to defaulters. The number of non-defaulters is the highest among the three clusters.
3. Cluster 2 (Red): Has a higher number of defaulters compared to non-defaulters. **Overall Insight:**
   Most of the customers across all clusters have not defaulted.

Recommendations to Stakeholders

1. Cluster 1 (Younger Segment): This segment likely includes students and early-career individuals. In addition to beginner credit cards, they may benefit from targeted credit products such as tuition support, laptop financing, or short-term education loans. Offers should focus on tools that support personal growth and mobility. Marketing can leverage youth-driven platforms like TikTok, X, Instagram, and mobile apps, with messaging that highlights access, empowerment, and future-building.
2. Cluster 2 (older): For older consumers with good credit history, recommend credit cards that offer high credit limits, personalized rewards tailored to their lifestyle, and user-friendly technology. Focus on easy-to-use apps or websites with clear navigation, along with enhanced security features like fraud protection as the older people tend to be more caustious. Offer premium perks such as Health and travel insurance.

Generating another set of segments using hierarchical clustering In agglomerative hierarchical clustering, we start with each individual data point as its own cluster. Then we gradually merge the closest clusters step by step, until all the data points are in one big cluster — that's the full population.

[27]:
```
data.head(2)
```

[27]:

0	Age	Edu	Years Employed	Income	Card Debt	Other Debt	Defaulted	DebtIncomeRatio
0	41	2	6	19	0.124	1.073	0	6.3
1	47	1	26	100	4.582	8.218	0	12.8

[28]:
```
num_features = ['Age', 'Years Employed', 'Income', 'Card Debt', 'Other Debt',␣
↪'DebtIncomeRatio']
```

[29]:
```
data[num_features].dtypes
```

[29]:
```
0
Age                 object
Years Employed      object
Income              object
Card Debt           object
Other Debt          object
DebtIncomeRatio     object
dtype: object
```

[30]:
```
# num_features to float
data[num_features] = data[num_features].astype(float)
```

[31]:
```
# creating a new data frame
X = data[num_features]
X.head()
```

[31]:

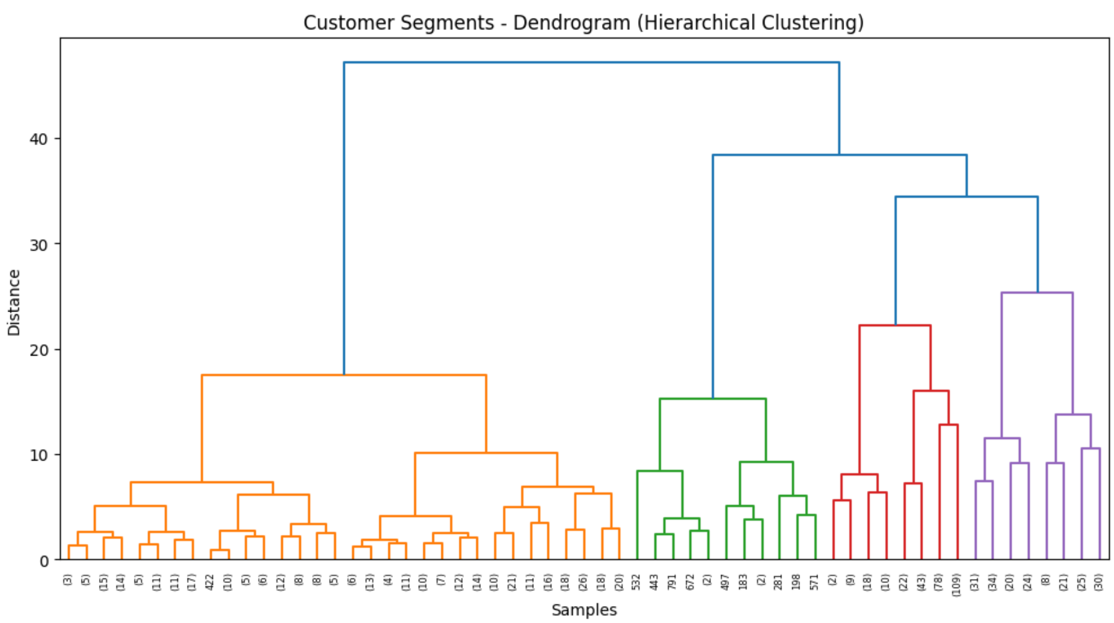
0	Age	Years Employed	Income	Card Debt	Other Debt	DebtIncomeRatio
0	41.0	6.0	19.0	0.124	1.073	6.3
1	47.0	26.0	100.0	4.582	8.218	12.8

2	33.0	10.0	57.0	6.111	5.802	20.9
3	29.0	4.0	19.0	0.681	0.516	6.3
4	47.0	31.0	253.0	9.308	8.908	7.2

[32]:
```
# Scaling X
x_scaler = StandardScaler()
X_scaled = x_scaler.fit_transform(X)
```

[33]:
```
# Use Ward's method for linkage
# This computes the linkage matrix, which contains the distances between␣
↪clusters at each merge step.
linked = linkage(X_scaled, method='ward')
```

[34]:
```
plt.figure(figsize=(12, 6))
dendrogram(linked, truncate_mode='level', p=5)
plt.title('Customer Segments - Dendrogram (Hierarchical Clustering)')
plt.xlabel('Samples')
plt.ylabel('Distance')
plt.show()
```



Grapgh Interpretation

1. **X-axis (Samples)**

- Each label on the x-axis represents a customer (or data point).
- These are merged into clusters as you move up the tree.

2. **Y-axis (Distance)**

- This shows the distance at which clusters were joined.

3. **Horizontal Lines = Merges**

- Every horizontal line connects two clusters (or points) that were joined.
- The height of the line tells you the distance at which they were merged.
- For the tallest horizontal blue line, that was the last merge, combining the two biggest groups at the highest distance (i.e., they were the least similar).
- Shorter distance = more similarity
- Longer distance = less similarity