# MuhammadHassanShah_P200025_C_AILab_07

April 4, 2023

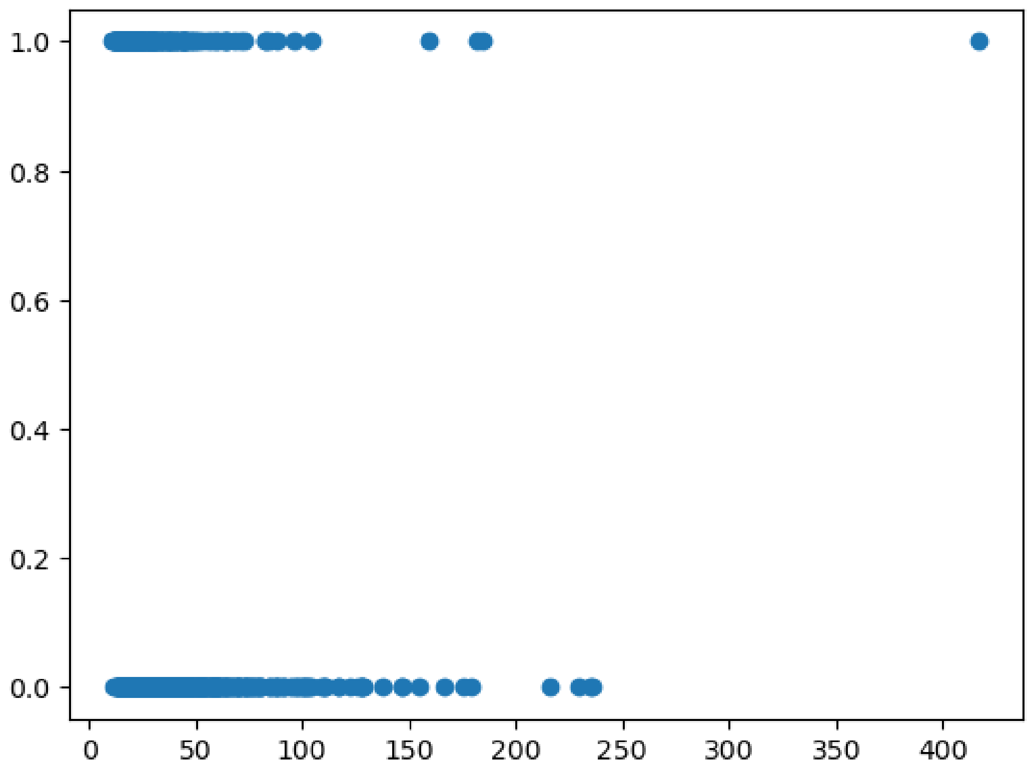
[24]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
import seaborn as sns
from sklearn.metrics import accuracy_score, ConfusionMatrixDisplay,␣
 ↪confusion_matrix, classification_report
```

[2]:
```
df = pd.read_csv('Cust_Segmentation.csv')

df.head()
```

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

   Defaulted Address  DebtIncomeRatio
0        0.0  NBA001              6.3
1        0.0  NBA021             12.8
2        1.0  NBA013             20.9
3        0.0  NBA009              6.3
4        0.0  NBA008              7.2
```

[3]:
```
x = df['Income'] - (df['Card Debt'] + df['Other Debt'])

y = df['Defaulted']
plt.scatter(x,y)
```

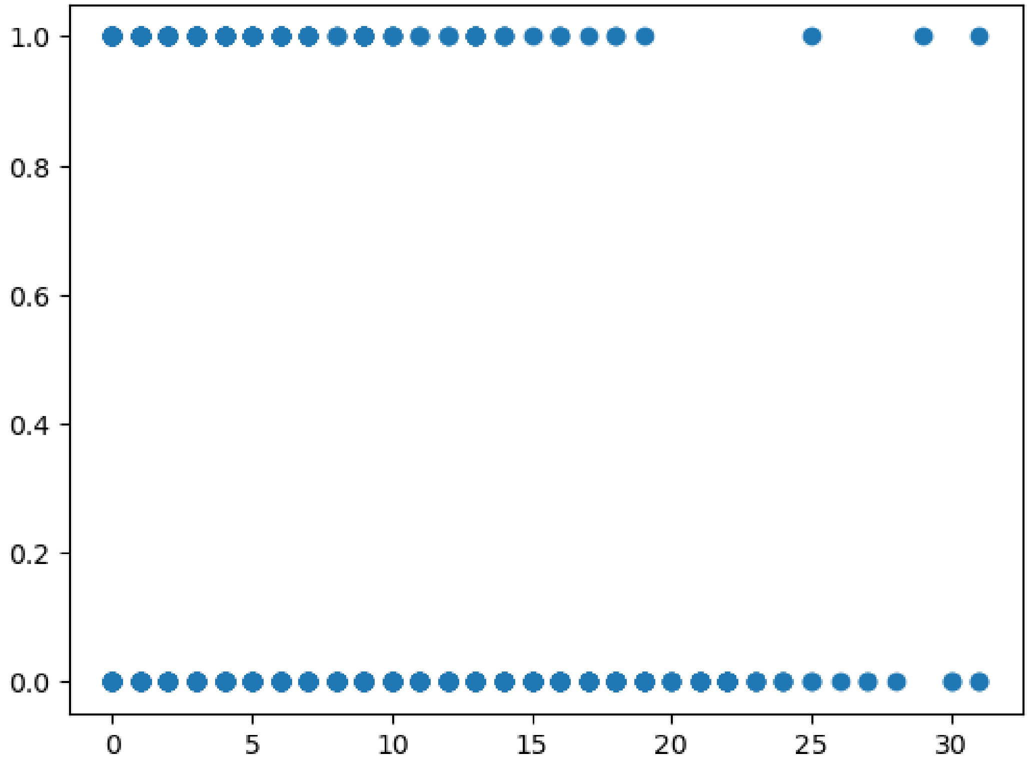
[3]:
```
<matplotlib.collections.PathCollection at 0x7f7cd2292550>
```



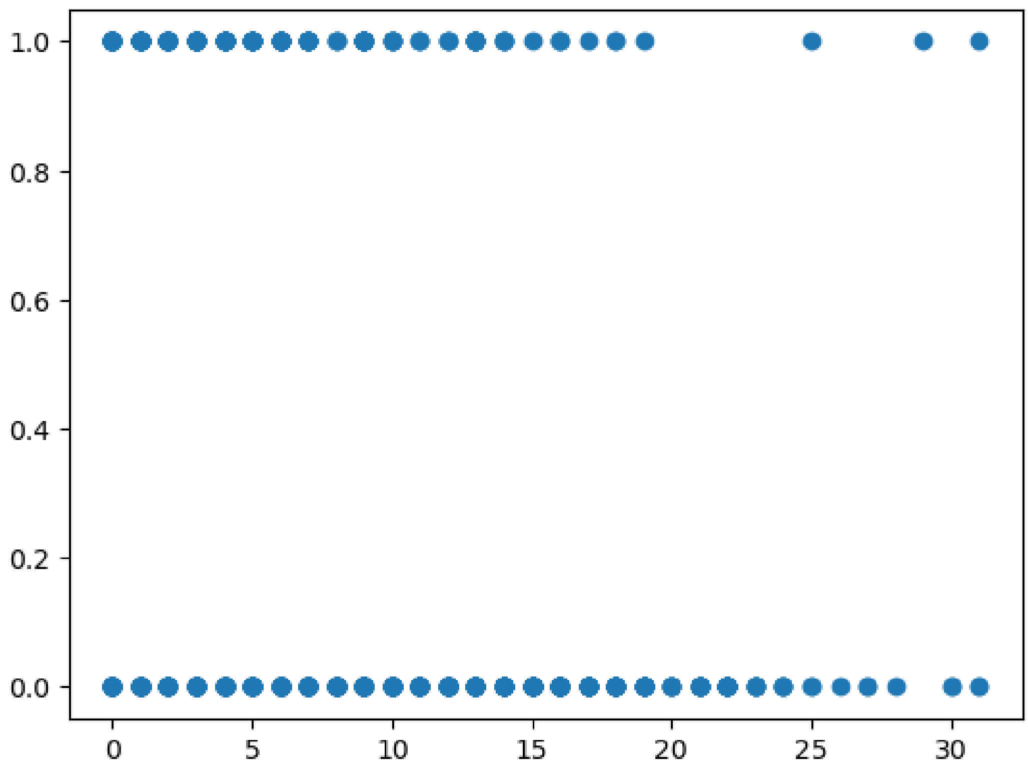

[4]:
```
df['Defaulted'].mean()
```

[4]: 0.26142857142857145

[5]:
```
plt.scatter(df['Years Employed'],df['Defaulted'])
plt.show()
```




[6]:
```
plt.scatter(df['Years Employed'],df['Defaulted'])
plt.show()
```




[7]:
```
# nothing that we can decide how to fill defaulted null values because no␣
↪pattern is being found,
#so just replace with mod
```

[9]:
```
df['Defaulted'].mode()
```

[9]:
```
0    0.0
Name: Defaulted, dtype: float64
```

[11]:
```
df.replace(np.nan,0, inplace = True)
```

[12]:
```
target = df['Defaulted']
df = df.drop(['Customer Id', 'Address', 'Defaulted'], axis=1) # removing␣
↪useless features and splitting
```

[13]:
```
df.head()
```

[13]:

| | Age | Edu | Years Employed | Income | Card Debt | Other Debt | DebtIncomeRatio |
|---|---|---|---|---|---|---|---|
| 0 | 41 | 2 | 6 | 19 | 0.124 | 1.073 | 6.3 |
| 1 | 47 | 1 | 26 | 100 | 4.582 | 8.218 | 12.8 |
| 2 | 33 | 2 | 10 | 57 | 6.111 | 5.802 | 20.9 |
| 3 | 29 | 2 | 4 | 19 | 0.681 | 0.516 | 6.3 |

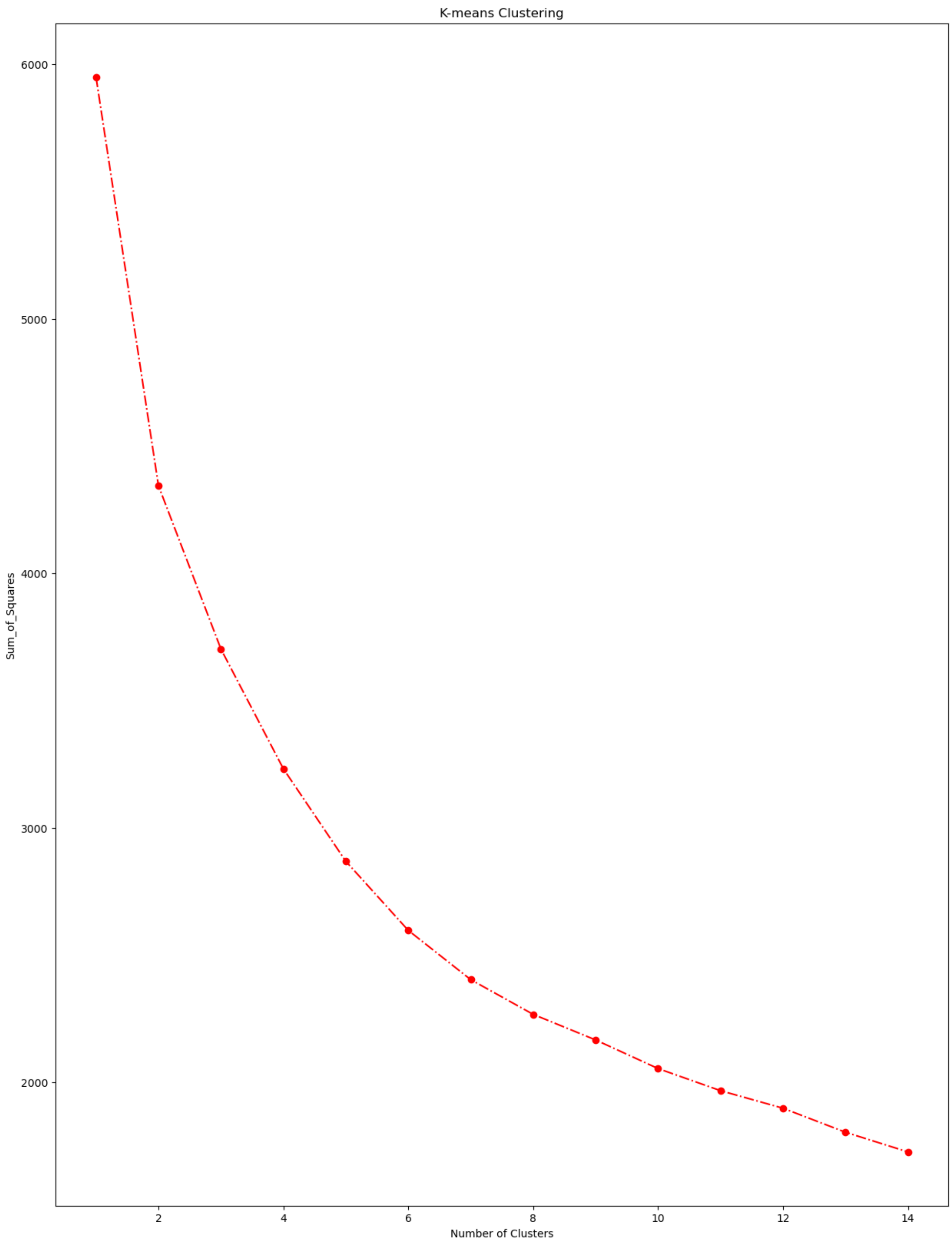
```
   4   47    1                 31      253       9.308        8.908                 7.2
```

[14]:
```python
scaler = StandardScaler()
df[['Age','Edu','Years Employed','Income','Card Debt','Other␣
 ↪Debt','DebtIncomeRatio']] = scaler.fit_transform(df)

df.head()
```

[14]:
```
        Age       Edu  Years Employed    Income  Card Debt  Other Debt  \
0  0.742915  0.312122       -0.378790 -0.718459  -0.683811   -0.590489
1  1.489490 -0.766349        2.573721  1.384325   1.414474    1.512962
2 -0.252518  0.312122        0.211712  0.268032   2.134141    0.801704
3 -0.750235  0.312122       -0.674041 -0.718459  -0.421643   -0.754467
4  1.489490 -0.766349        3.311849  5.356249   3.638900    1.716094

   DebtIncomeRatio
0        -0.576525
1         0.391387
2         1.597554
3        -0.576525
4        -0.442507
```

[15]:
```python
Sum_of_Squares = []
for i in range(1,15):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
    kmeans.fit(df)
    Sum_of_Squares.append(kmeans.inertia_)
```

[16]:
```python
plt.figure(figsize = (15,20))
plt.plot(range(1, 15), Sum_of_Squares, marker = 'o', linestyle = '-.
 ↪',color='red')
plt.xlabel('Number of Clusters')
plt.ylabel('Sum_of_Squares')
plt.title('K-means Clustering')
plt.show()
```


K-means Clustering
6000
5000
4000
3000
2000
Sum_of_Squares
2
4
6
8
10
12
14
Number of Clusters

# 1 k = 4

[17]:
```python
kmeans = KMeans(n_clusters = 4, init = 'k-means++', random_state = 42)
```

[18]:
```python
kmeans.fit(df)
y_mean = kmeans.fit_predict(df)
y_mean
```

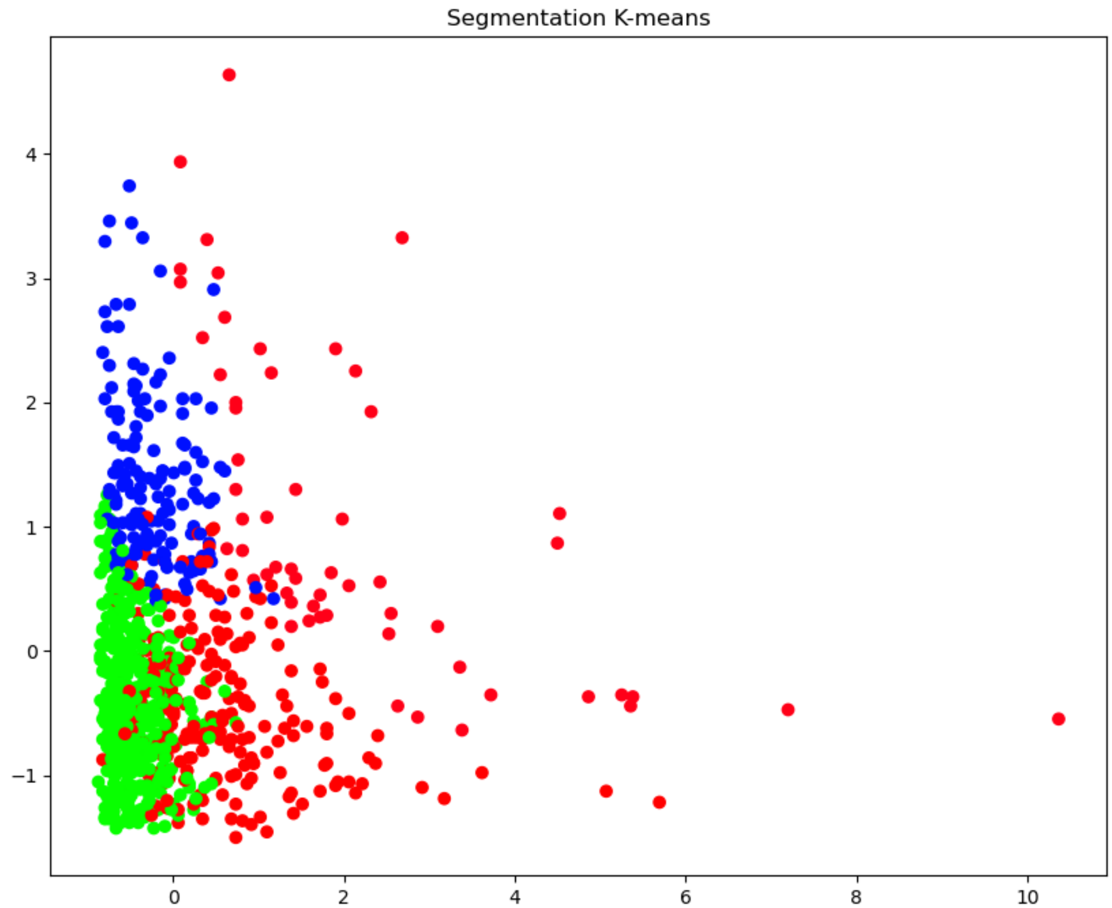
[18]:
```
array([1, 3, 2, 1, 3, 0, 1, 1, 1, 0, 0, 1, 1, 2, 1, 1, 1, 1, 0, 0, 1, 2,
       2, 0, 3, 0, 1, 0, 0, 1, 0, 2, 1, 1, 1, 1, 1, 1, 1, 0, 1, 3, 2, 3,
       2, 0, 1, 1, 1, 1, 0, 2, 2, 0, 2, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1,
       0, 1, 1, 1, 1, 0, 0, 2, 1, 1, 0, 1, 3, 0, 0, 2, 3, 1, 0, 1, 1, 1,
       1, 2, 0, 1, 1, 2, 1, 0, 1, 1, 2, 1, 2, 3, 0, 0, 1, 3, 1, 1, 1, 0,
       0, 2, 1, 1, 1, 1, 1, 0, 2, 1, 1, 1, 1, 0, 1, 0, 1, 1, 2, 1, 0, 0,
       1, 1, 1, 1, 0, 1, 0, 2, 2, 1, 1, 1, 3, 2, 0, 2, 1, 1, 1, 2, 1, 2,
       1, 2, 0, 0, 2, 2, 1, 2, 0, 1, 1, 1, 1, 1, 1, 0, 3, 1, 1, 1, 1, 2,
       0, 1, 1, 0, 2, 1, 1, 3, 2, 2, 1, 0, 0, 0, 2, 2, 0, 1, 0, 1, 1, 0,
       3, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 2, 2, 0, 0,
       1, 0, 1, 1, 1, 1, 3, 1, 2, 0, 1, 2, 2, 1, 3, 1, 0, 1, 2, 1, 1, 0,
       1, 1, 1, 0, 0, 0, 1, 1, 1, 2, 1, 2, 0, 2, 0, 1, 1, 1, 1, 2, 2, 1,
       0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 2, 2, 1, 1, 1, 3, 0, 1, 0, 1,
       0, 1, 0, 0, 1, 0, 1, 2, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 2, 1, 2, 2,
       2, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 3, 1, 0, 1, 1, 0, 2, 1, 0, 0,
       0, 3, 1, 0, 0, 2, 1, 0, 2, 0, 2, 0, 1, 2, 3, 0, 1, 1, 1, 1, 3, 1,
       0, 1, 1, 0, 3, 1, 1, 1, 0, 2, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0,
       2, 2, 2, 0, 1, 1, 1, 1, 1, 0, 3, 0, 0, 2, 1, 1, 1, 2, 0, 2, 1, 1,
       1, 0, 1, 1, 2, 1, 1, 1, 1, 2, 0, 2, 0, 0, 1, 0, 2, 2, 1, 0, 0, 2,
       2, 1, 0, 2, 1, 1, 3, 0, 1, 0, 2, 1, 2, 2, 1, 1, 3, 1, 1, 1, 3, 0,
       0, 0, 1, 3, 0, 2, 1, 1, 0, 1, 3, 1, 1, 1, 3, 1, 2, 1, 1, 2, 1, 3,
       2, 2, 1, 1, 0, 1, 0, 2, 0, 2, 0, 0, 0, 1, 0, 1, 1, 2, 2, 2, 0, 1,
       1, 1, 2, 3, 1, 1, 2, 3, 0, 0, 0, 1, 1, 3, 0, 1, 1, 1, 3, 0, 1, 1,
       0, 2, 0, 2, 1, 1, 1, 3, 0, 0, 1, 1, 1, 2, 2, 1, 1, 1, 0, 1, 1, 0,
       2, 2, 1, 2, 3, 3, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1,
       2, 0, 1, 0, 1, 0, 3, 0, 2, 1, 0, 0, 2, 2, 1, 0, 0, 1, 3, 1, 1, 3,
       1, 1, 1, 2, 2, 1, 1, 0, 3, 1, 0, 0, 0, 2, 2, 1, 1, 2, 1, 0, 0, 2,
       2, 1, 1, 1, 0, 0, 1, 0, 1, 3, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1, 2, 2,
       2, 1, 1, 1, 3, 1, 2, 2, 1, 2, 1, 1, 0, 3, 0, 1, 0, 0, 3, 1, 2, 0,
       1, 0, 1, 1, 3, 1, 0, 2, 0, 1, 1, 1, 1, 1, 3, 0, 1, 1, 1, 3, 0, 1,
       1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 3, 1, 1, 0, 1, 0, 2, 1, 2, 1,
       2, 0, 1, 0, 2, 0, 1, 1, 1, 1, 1, 0, 1, 1, 2, 0, 1, 1, 1, 2, 1, 1,
       1, 0, 2, 1, 0, 2, 3, 1, 2, 3, 1, 2, 1, 0, 0, 3, 1, 1, 1, 2, 2, 3,
       1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 2, 0, 2, 0, 1, 2, 1, 0, 0, 2,
       1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 0, 2, 3, 2, 1, 0, 1, 1, 1, 1, 1, 0,
       1, 1, 1, 0, 2, 0, 3, 1, 0, 2, 1, 1, 1, 0, 3, 1, 0, 1, 1, 1, 0, 3,
       0, 2, 1, 1, 1, 1, 2, 1, 1, 0, 1, 2, 1, 1, 0, 1, 0, 2, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 2, 0, 3, 2, 1, 1, 2, 1, 2, 2, 1, 2,
       1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 2, 1, 0], dtype=int32)
```

# 2 Making a new feature which will show which cluster 1 row belongs to

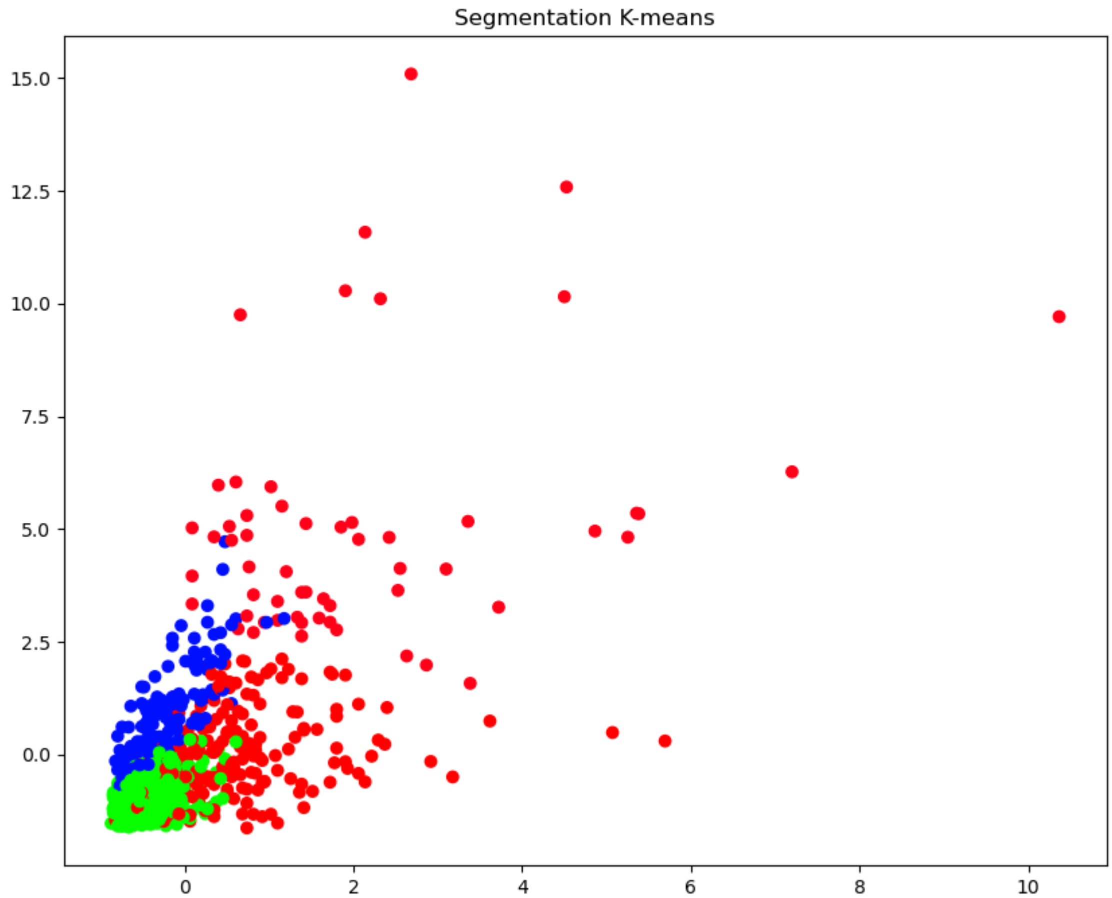
[19]:
```
df['K'] = kmeans.labels_
df.head()
```

[19]:
```
        Age       Edu  Years Employed    Income  Card Debt  Other Debt  \
0  0.742915  0.312122       -0.378790 -0.718459  -0.683811   -0.590489
1  1.489490 -0.766349        2.573721  1.384325   1.414474    1.512962
2 -0.252518  0.312122        0.211712  0.268032   2.134141    0.801704
3 -0.750235  0.312122       -0.674041 -0.718459  -0.421643   -0.754467
4  1.489490 -0.766349        3.311849  5.356249   3.638900    1.716094

   DebtIncomeRatio  K
0        -0.576525  1
1         0.391387  3
2         1.597554  2
3        -0.576525  1
4        -0.442507  3
```

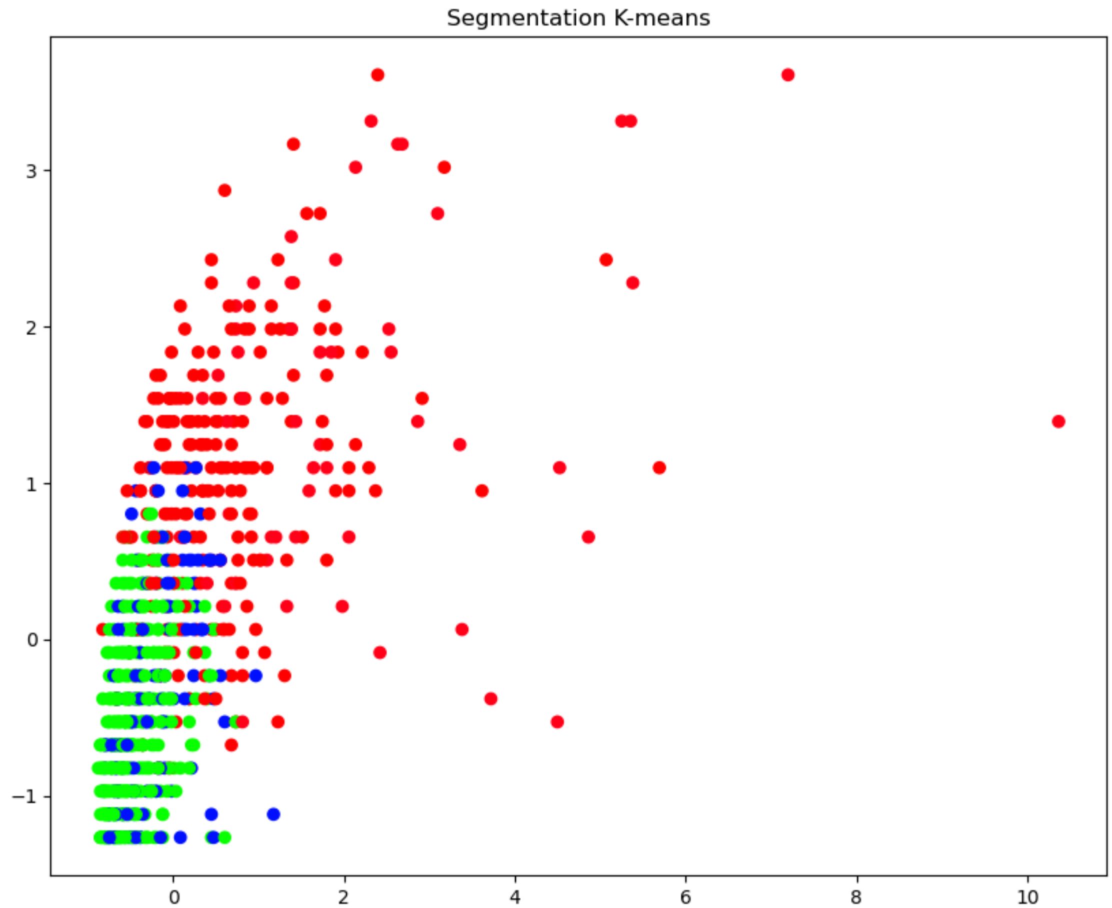
[31]:
```
x_axis = df['Income']
y_axis = df['DebtIncomeRatio']
plt.figure(figsize = (10, 8))
color_map = plt.cm.get_cmap('hsv', len(np.unique(df['K'])))
plt.scatter(x_axis, y_axis, c=df['K'],cmap=color_map)
plt.title('Segmentation K-means')
plt.show()
```



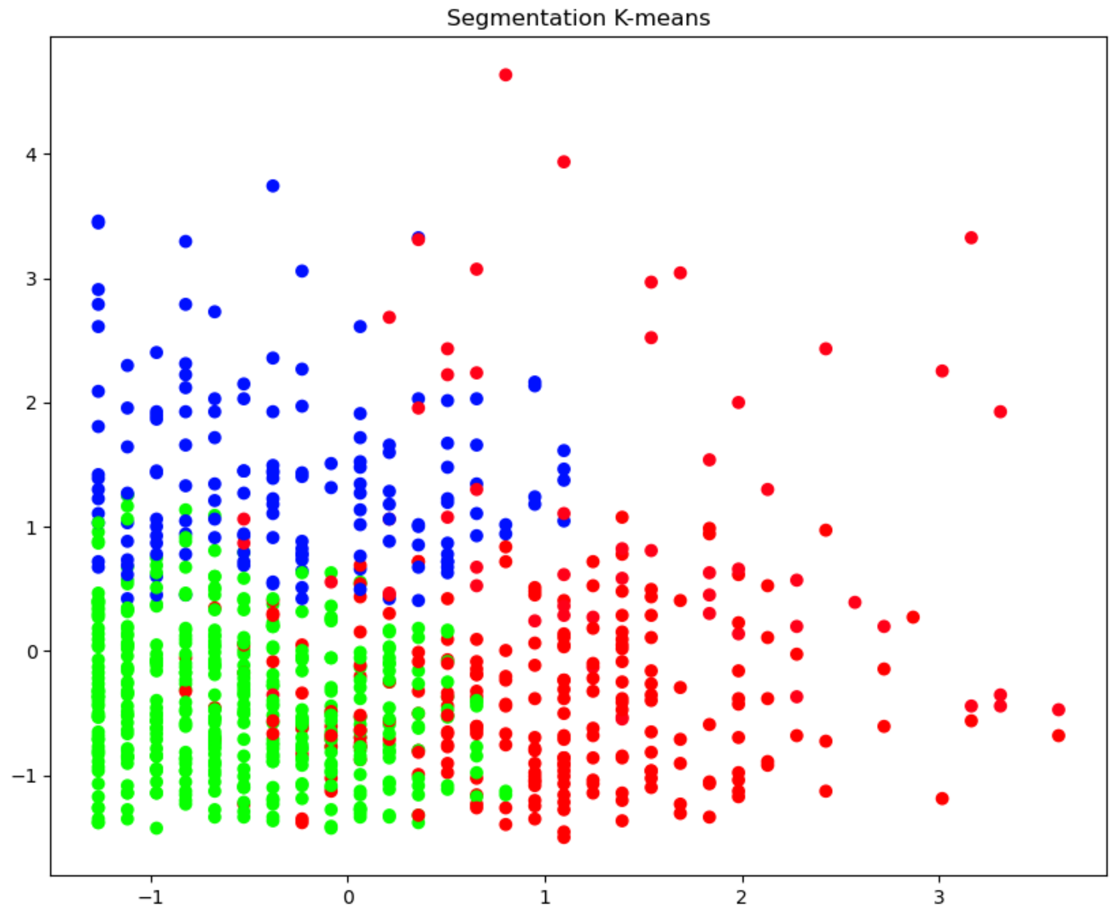

[32]:
```
x_axis = df['Income']
y_axis = df['Card Debt'] + df ['Other Debt']
plt.figure(figsize = (10, 8))
color_map = plt.cm.get_cmap('hsv', len(np.unique(df['K'])))
plt.scatter(x_axis, y_axis, c=df['K'],cmap=color_map)
plt.title('Segmentation K-means')
plt.show()
```



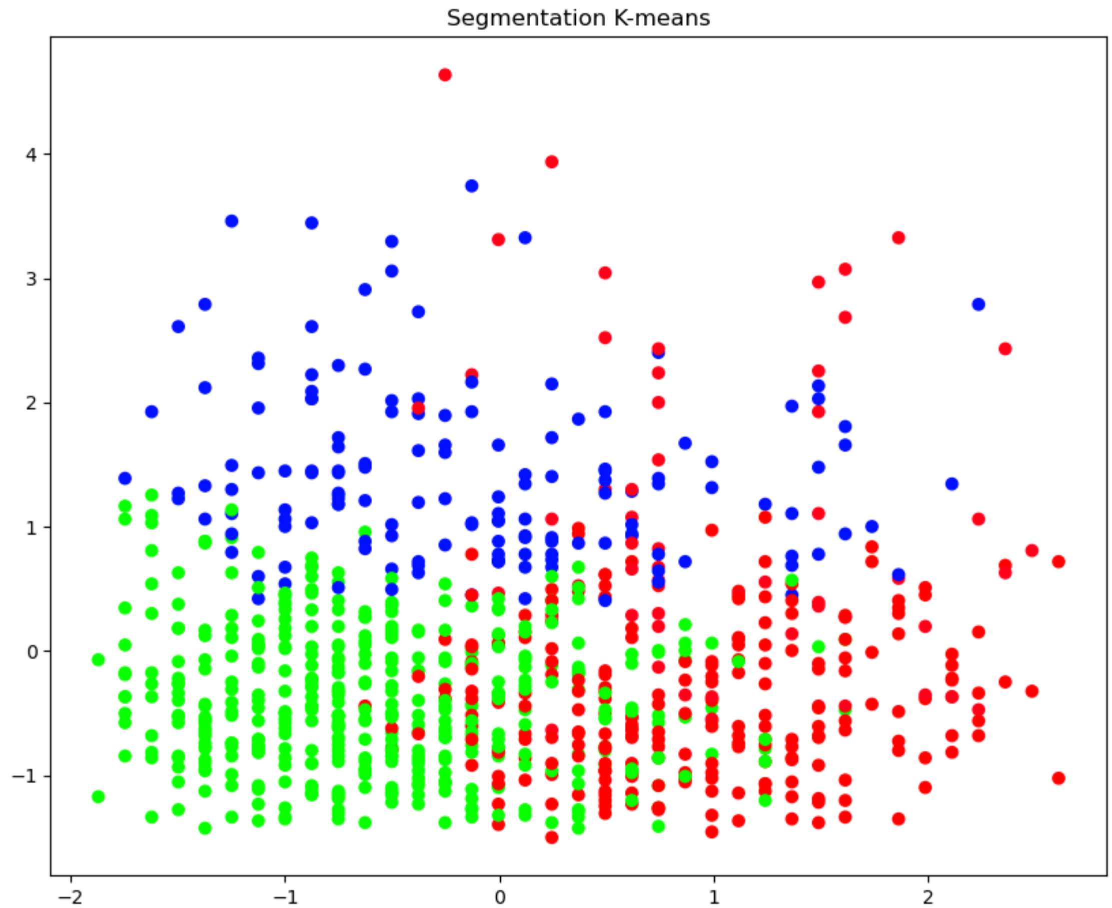

[34]:
```
x_axis = df['Income']
y_axis = df['Years Employed']
plt.figure(figsize = (10, 8))
color_map = plt.cm.get_cmap('hsv', len(np.unique(df['K'])))
plt.scatter(x_axis, y_axis, c=df['K'],cmap=color_map)
plt.title('Segmentation K-means')
plt.show()
```



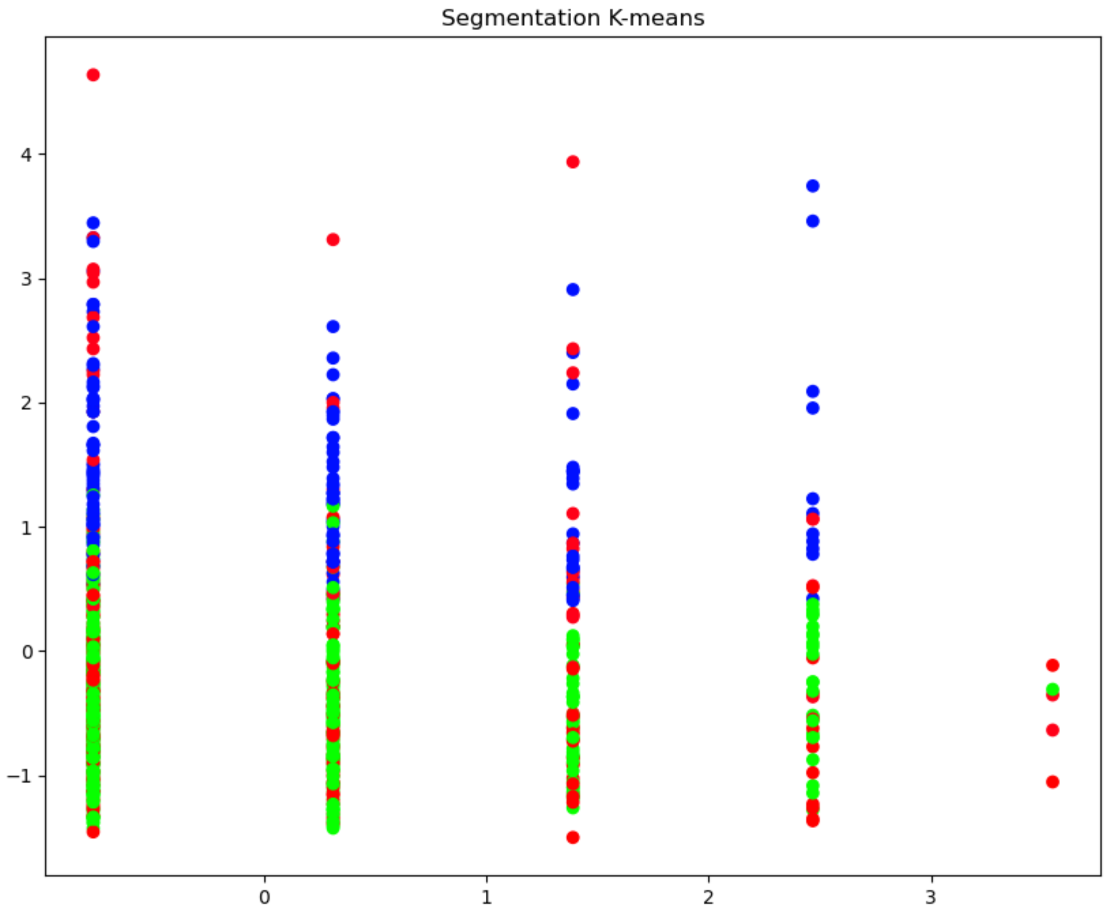

[36]:
```
x_axis = df['Years Employed']
y_axis = df['DebtIncomeRatio']
plt.figure(figsize = (10, 8))
color_map = plt.cm.get_cmap('hsv', len(np.unique(df['K'])))
plt.scatter(x_axis, y_axis, c=df['K'],cmap=color_map)
plt.title('Segmentation K-means')
plt.show()
```



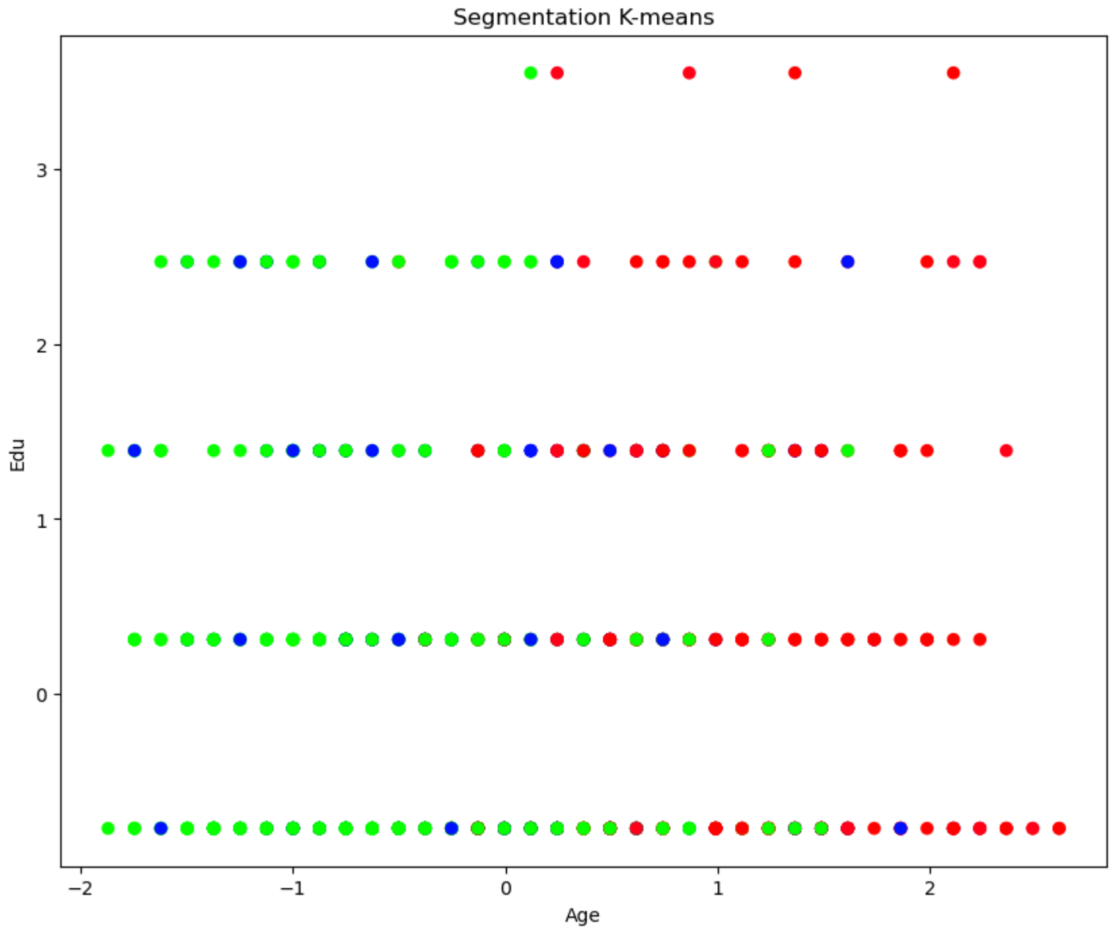

[37]:
```
x_axis = df['Age']
y_axis = df['DebtIncomeRatio']
plt.figure(figsize = (10, 8))
color_map = plt.cm.get_cmap('hsv', len(np.unique(df['K'])))
plt.scatter(x_axis, y_axis, c=df['K'],cmap=color_map)
plt.title('Segmentation K-means')
plt.show()
```

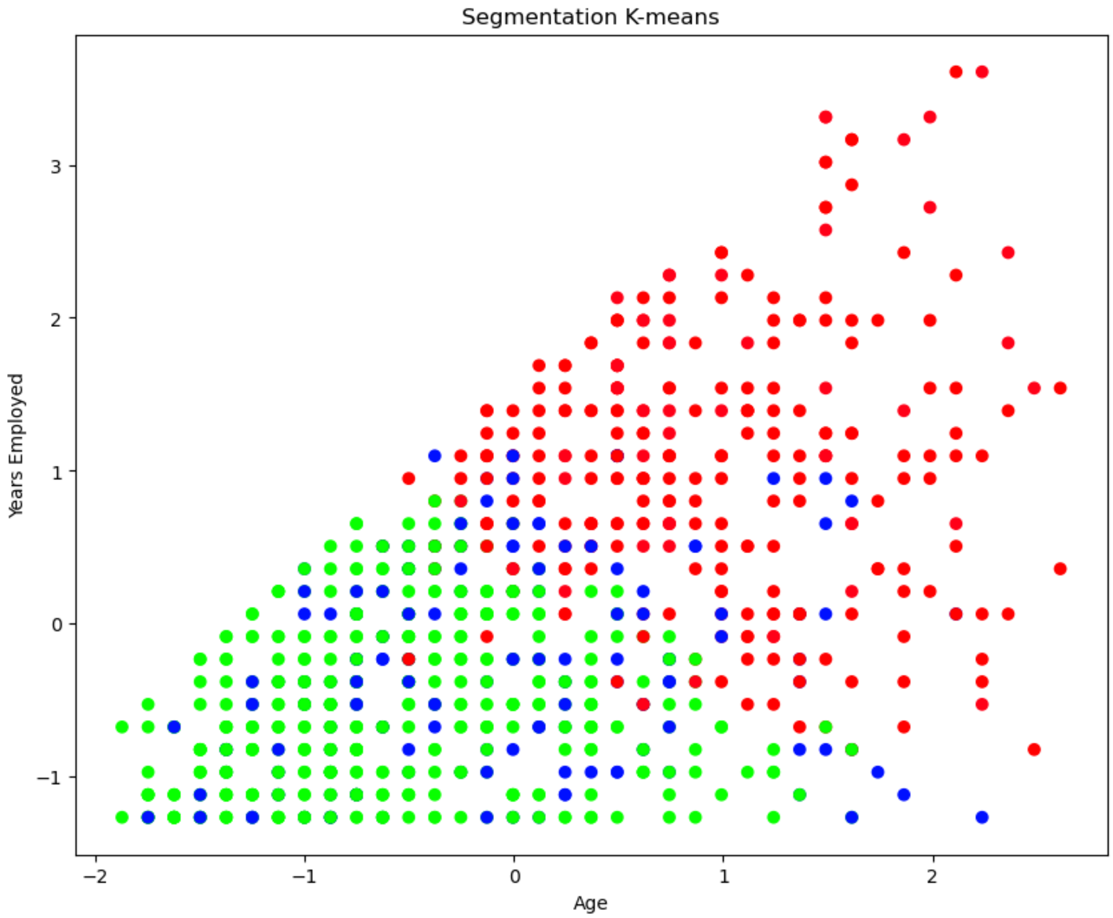



[38]:
```
x_axis = df['Edu']
y_axis = df['DebtIncomeRatio']
plt.figure(figsize = (10, 8))
color_map = plt.cm.get_cmap('hsv', len(np.unique(df['K'])))
plt.scatter(x_axis, y_axis, c=df['K'],cmap=color_map)
plt.title('Segmentation K-means')
plt.show()
```

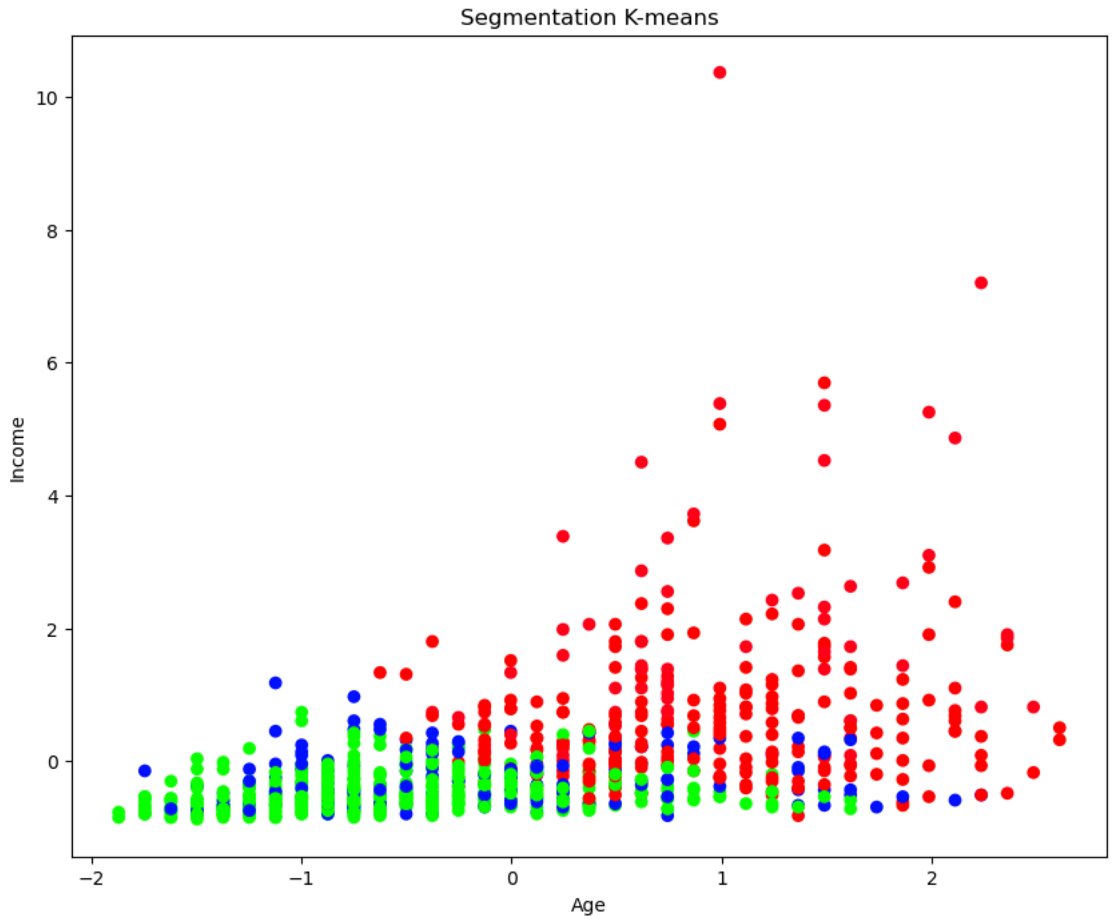


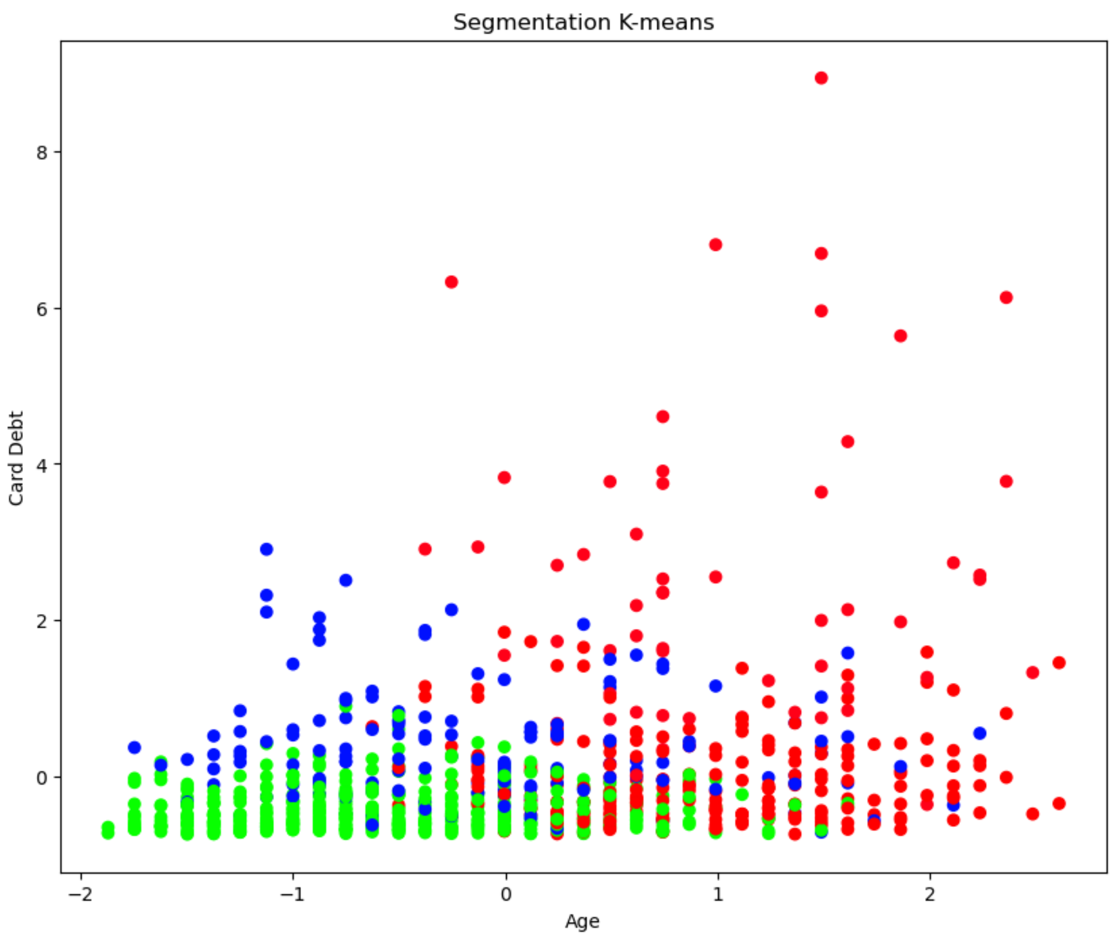

[41]:
```python
for i in df:
    for j in df:
        if i == j:
            continue
        x_axis = df[i]
        y_axis = df[j]
        plt.figure(figsize = (10, 8))
        color_map = plt.cm.get_cmap('hsv', len(np.unique(df['K'])))
        plt.xlabel(df[i].name)
        plt.ylabel(df[j].name)
        plt.scatter(x_axis, y_axis, c=df['K'],cmap=color_map)
        plt.title('Segmentation K-means')
        plt.show()
```

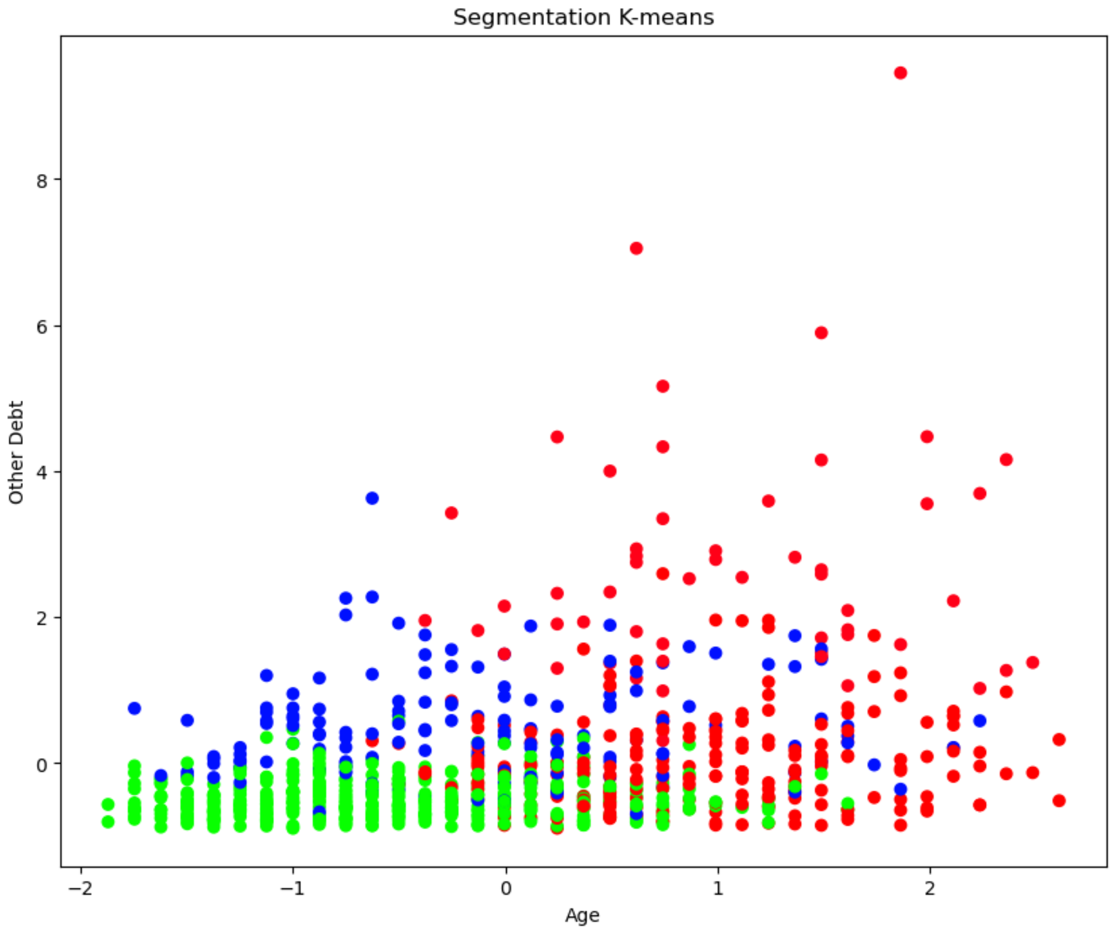

Segmentation K-means
Edu
Age
3
2
1
0
−2
−1
0
1
2



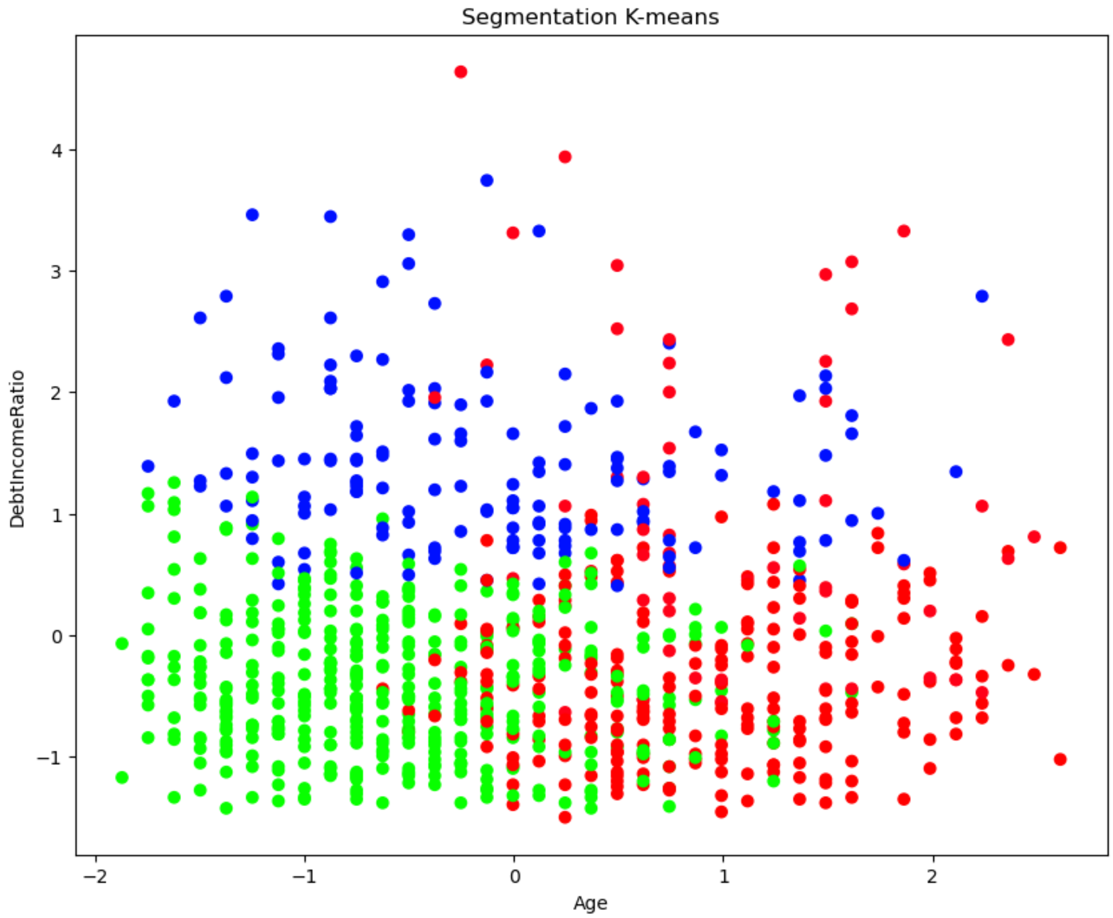
Segmentation K-means
Years Employed
Age
3
2
1
0
−1
−2
−1
0
1
2

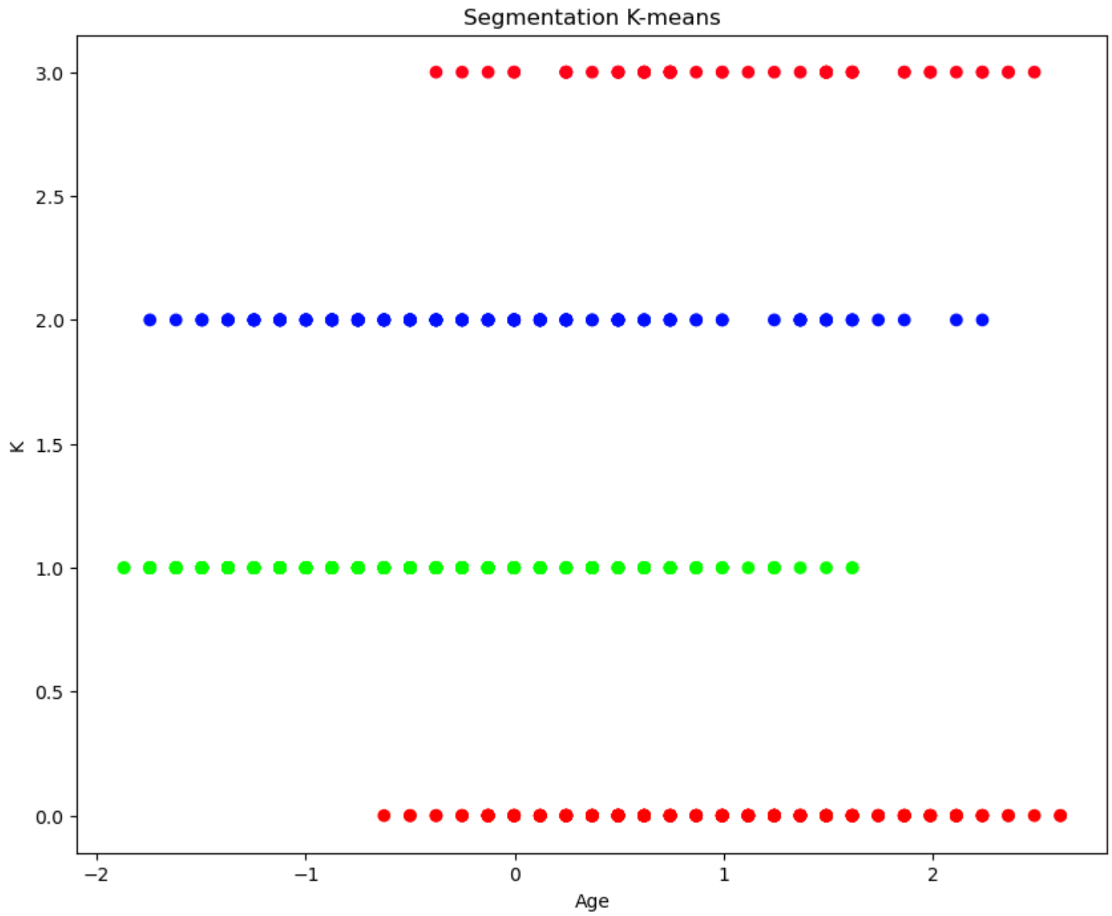

Segmentation K-means
Income
10
8
6
4
2
0
−2
−1
0
1
2
Age

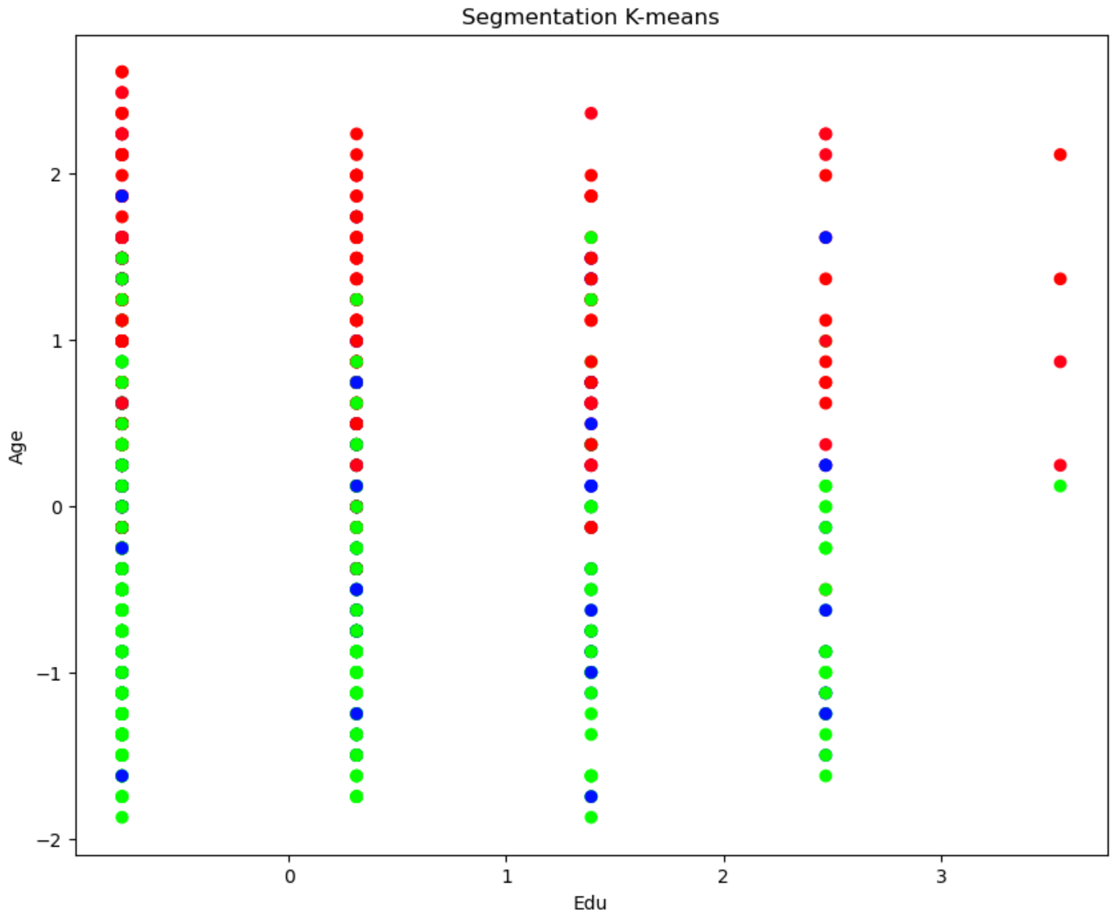

Segmentation K-means
Card Debt
Age
8
6
4
2
0
−2
−1
0
1
2

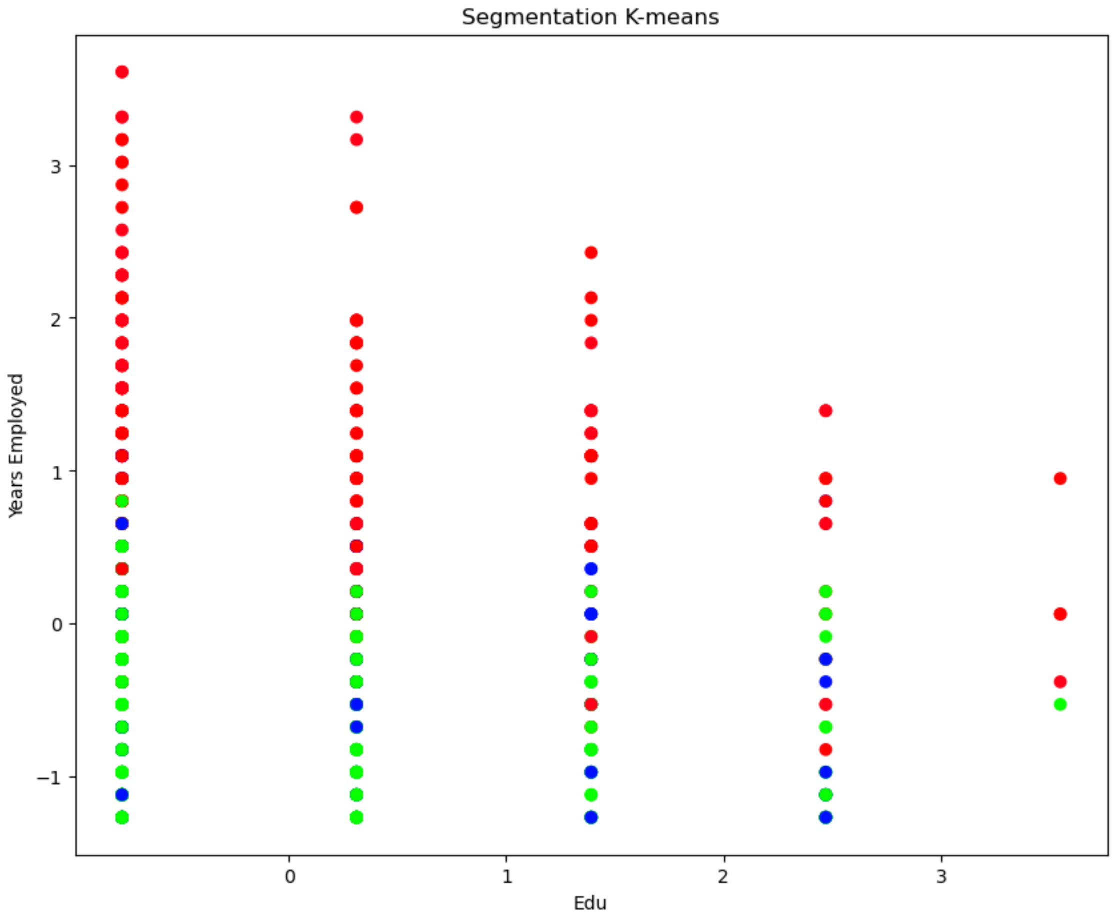


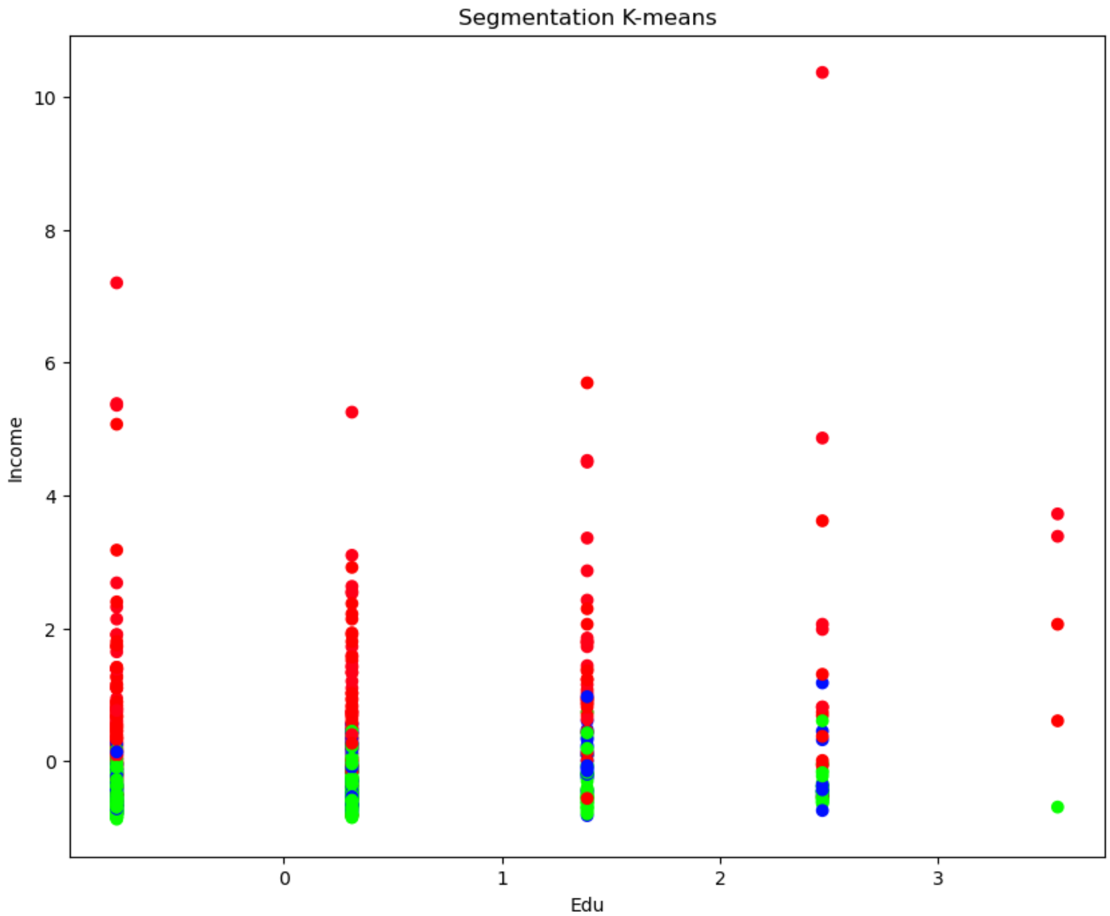
Segmentation K-means
Other Debt
Age
8
6
4
2
0
−2
−1
0
1
2

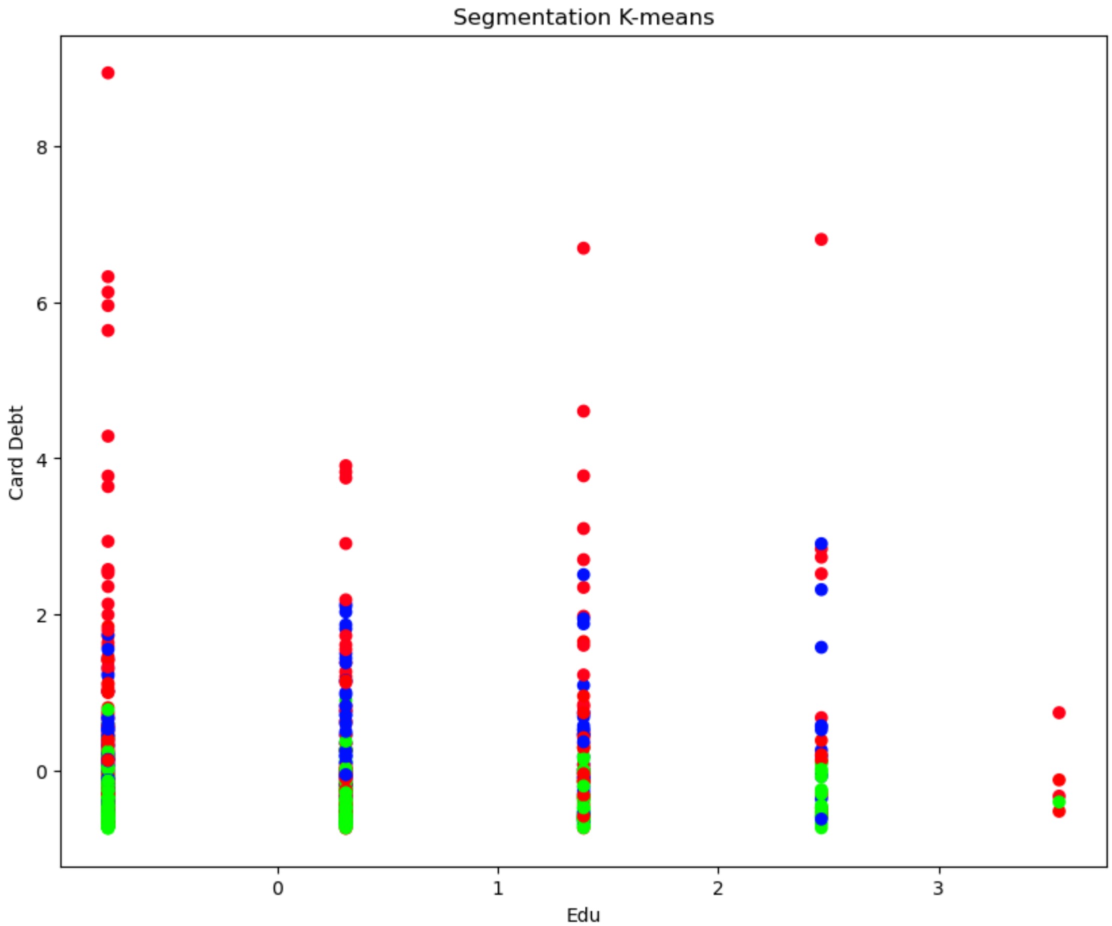

Segmentation K-means
DebtIncomeRatio
Age
4
3
2
1
0
−1
−2
−1
0
1
2

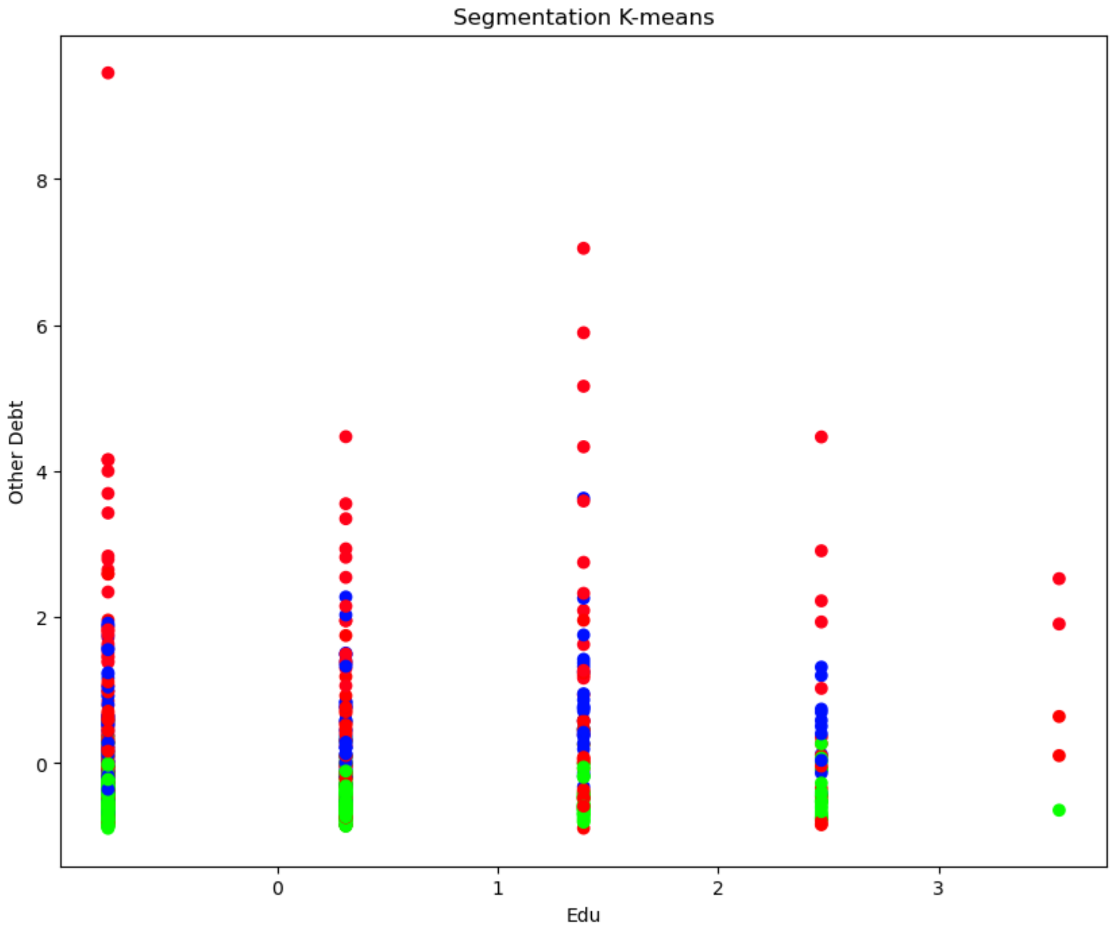

Segmentation K-means
3.0
2.5
2.0
1.5
1.0
0.5
0.0
K
−2
−1
0
1
2
Age


Segmentation K-means
Age
Edu
2
1
0
−1
−2
0
1
2
3

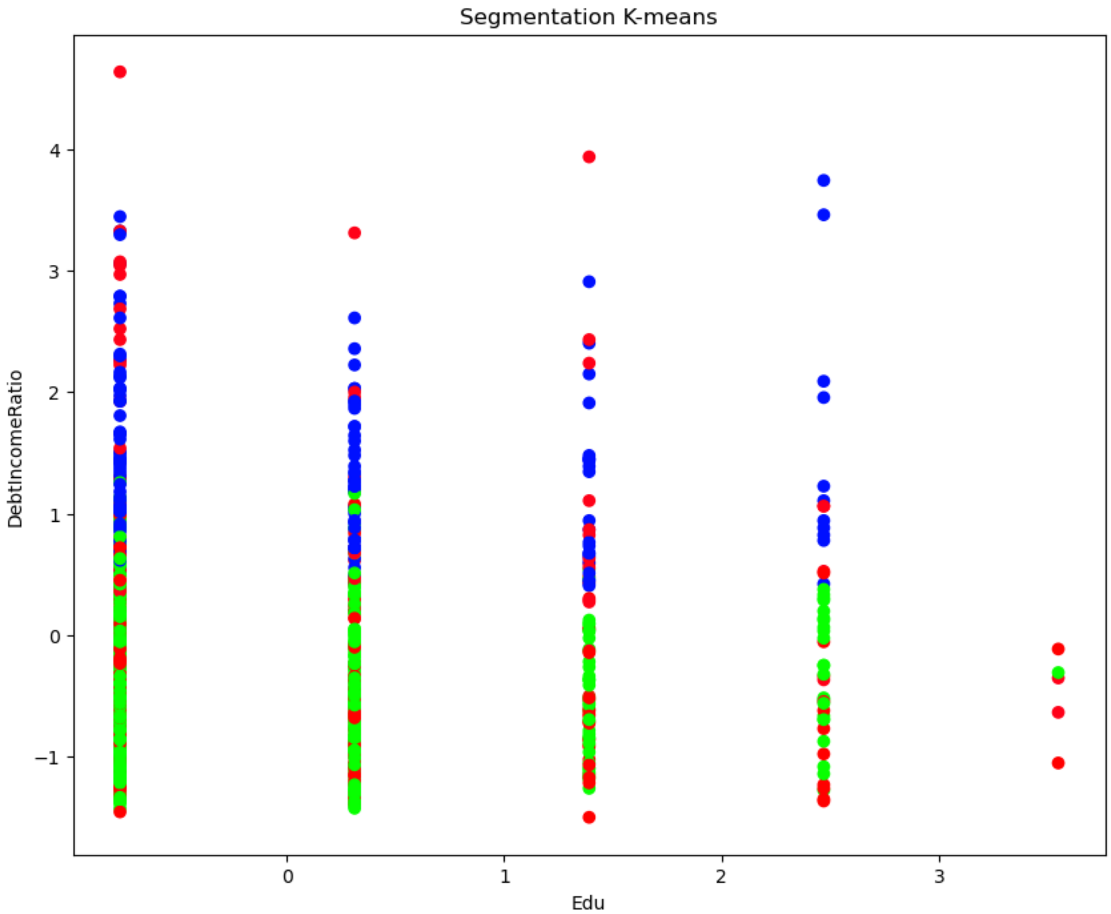

Segmentation K-means
Years Employed
Edu
3
2
1
0
−1
0
1
2
3

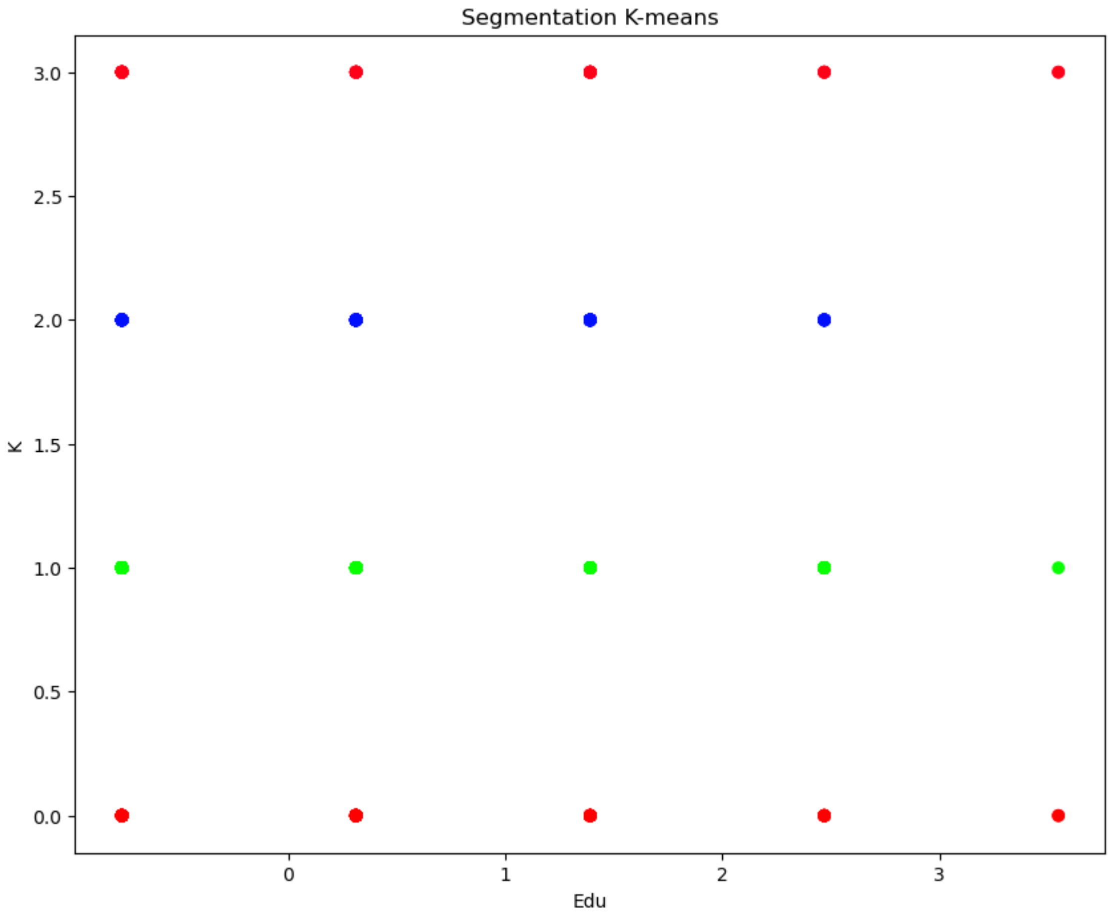

Segmentation K-means
Income
Edu
10
8
6
4
2
0
0
1
2
3

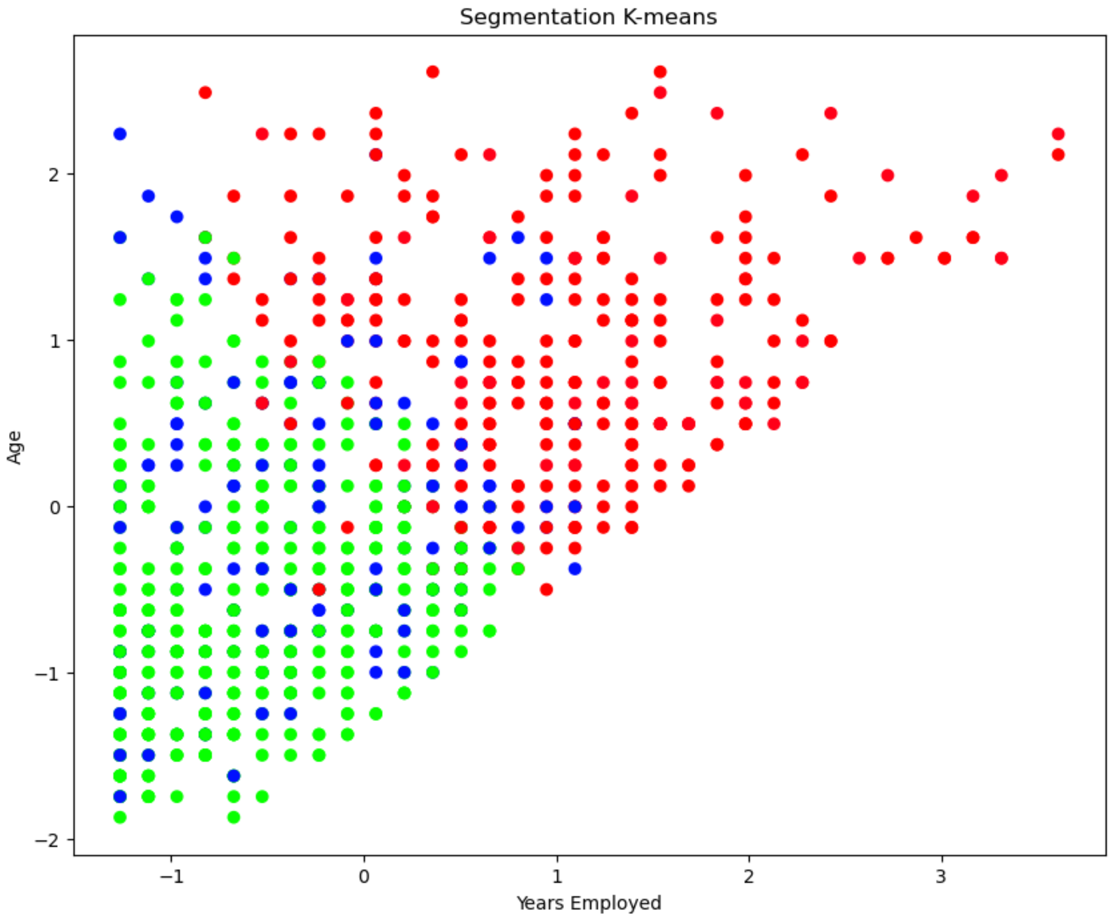

Segmentation K-means
Card Debt
Edu
8
6
4
2
0
0
1
2
3

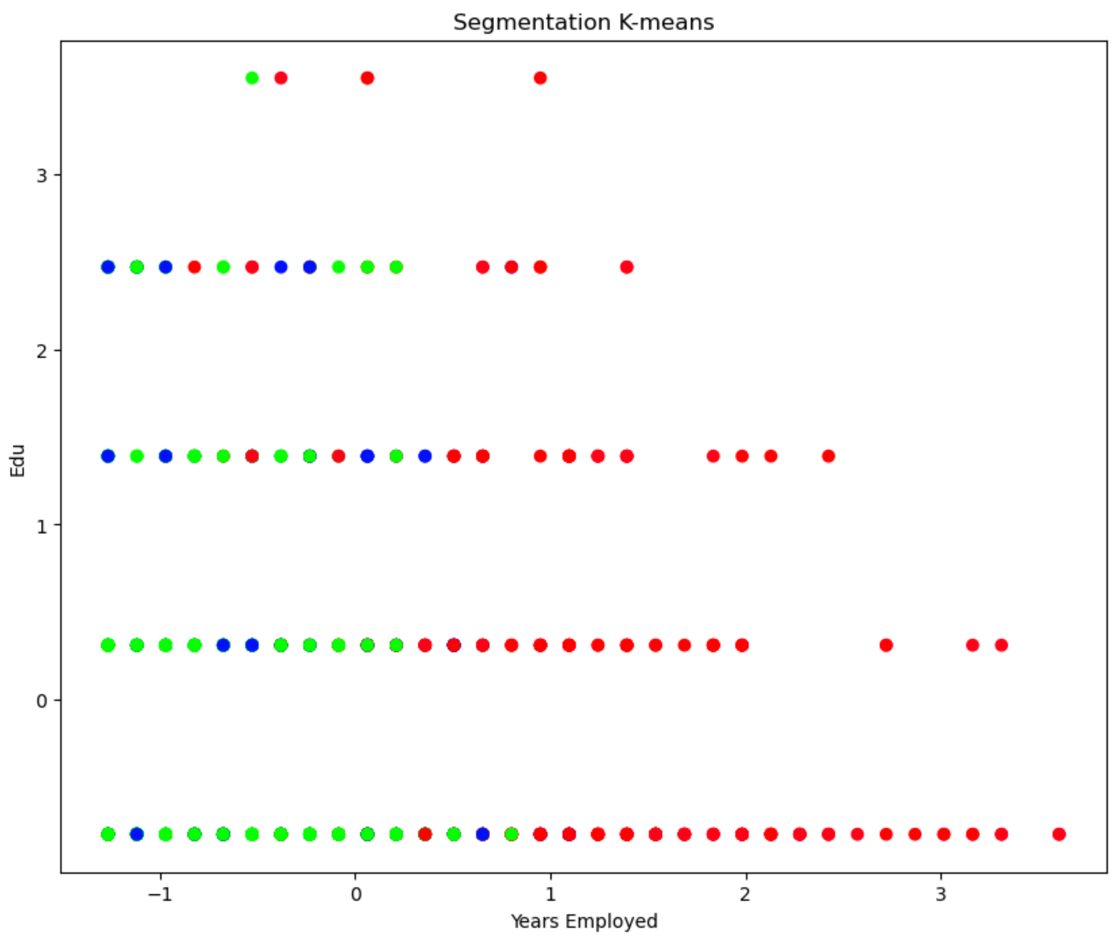


Segmentation K-means
Other Debt
Edu
0
2
4
6
8
0
1
2
3

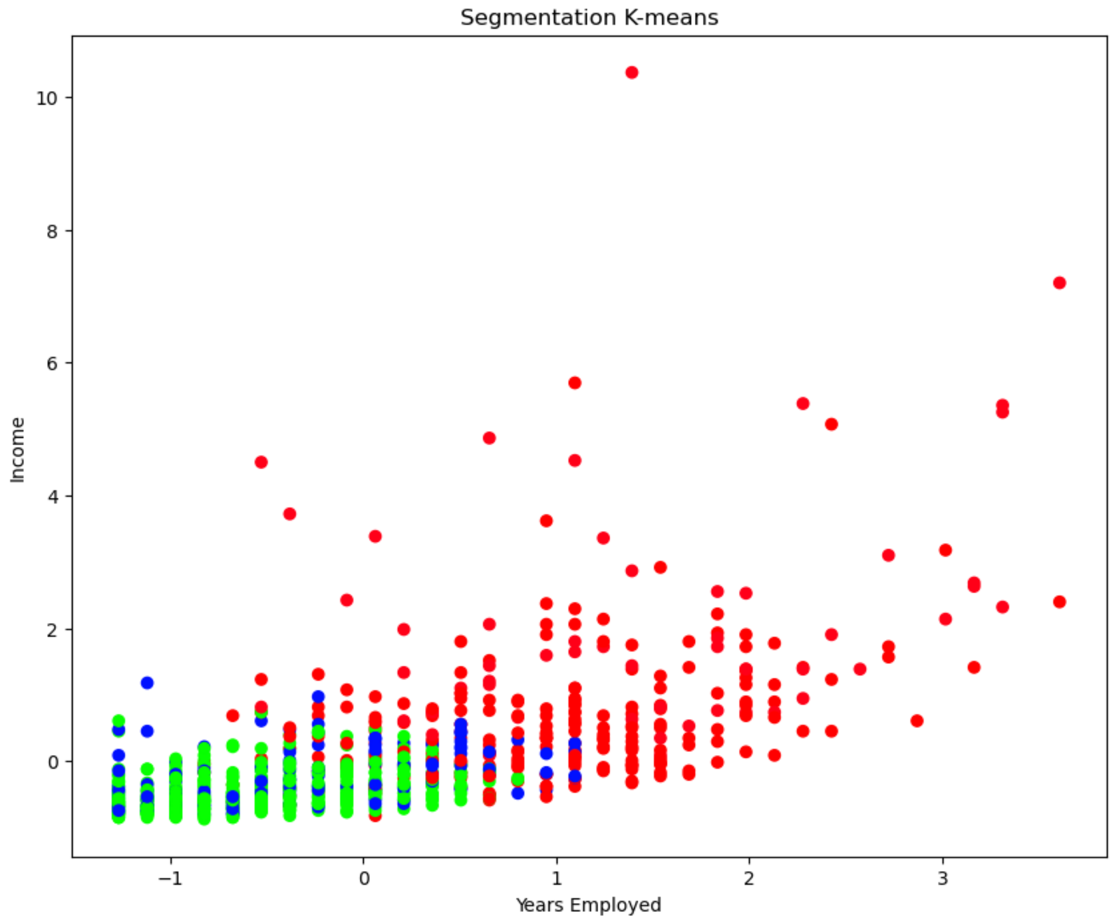

Segmentation K-means
DebtIncomeRatio
Edu
4
3
2
1
0
−1
0
1
2
3


Segmentation K-means
3.0
2.5
2.0
1.5
1.0
0.5
0.0
K
0
1
2
3
Edu

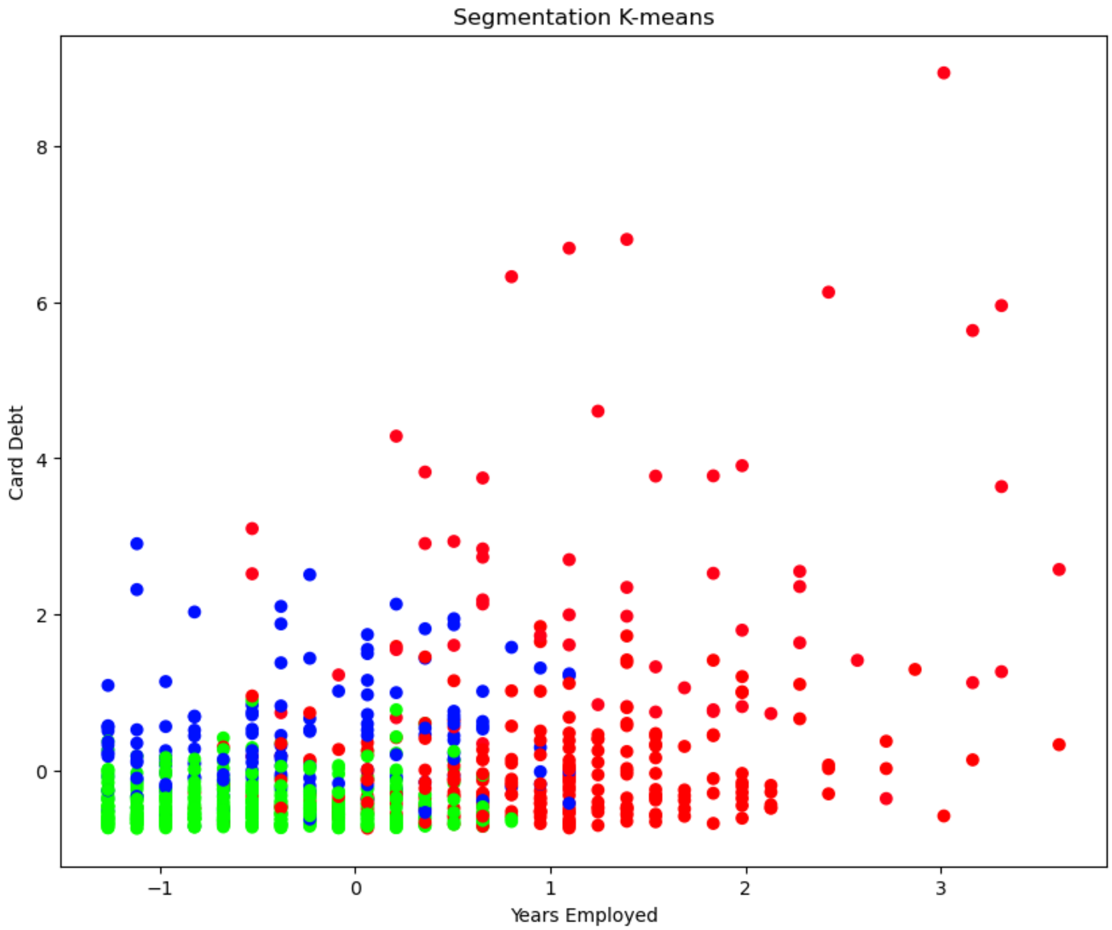

Segmentation K-means
Age
Years Employed
2
1
0
−1
−2
−1
0
1
2
3

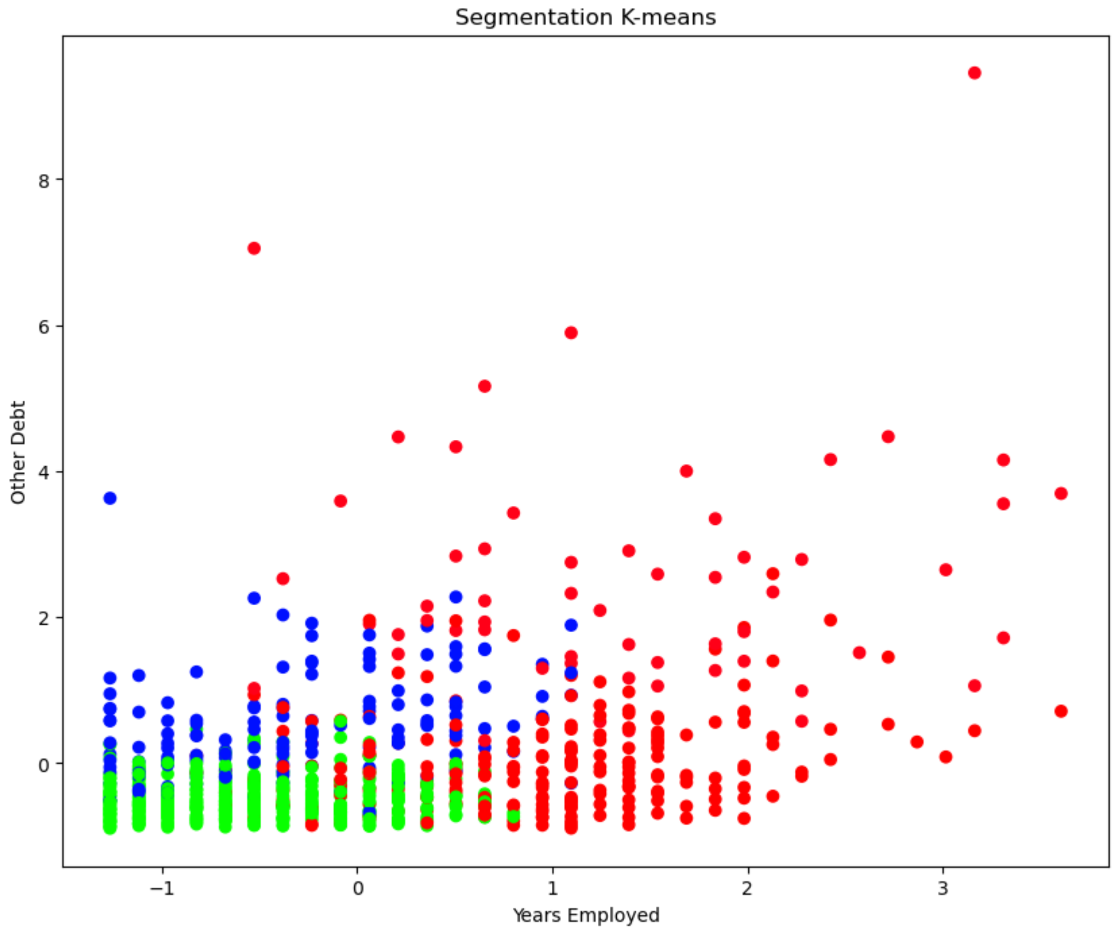


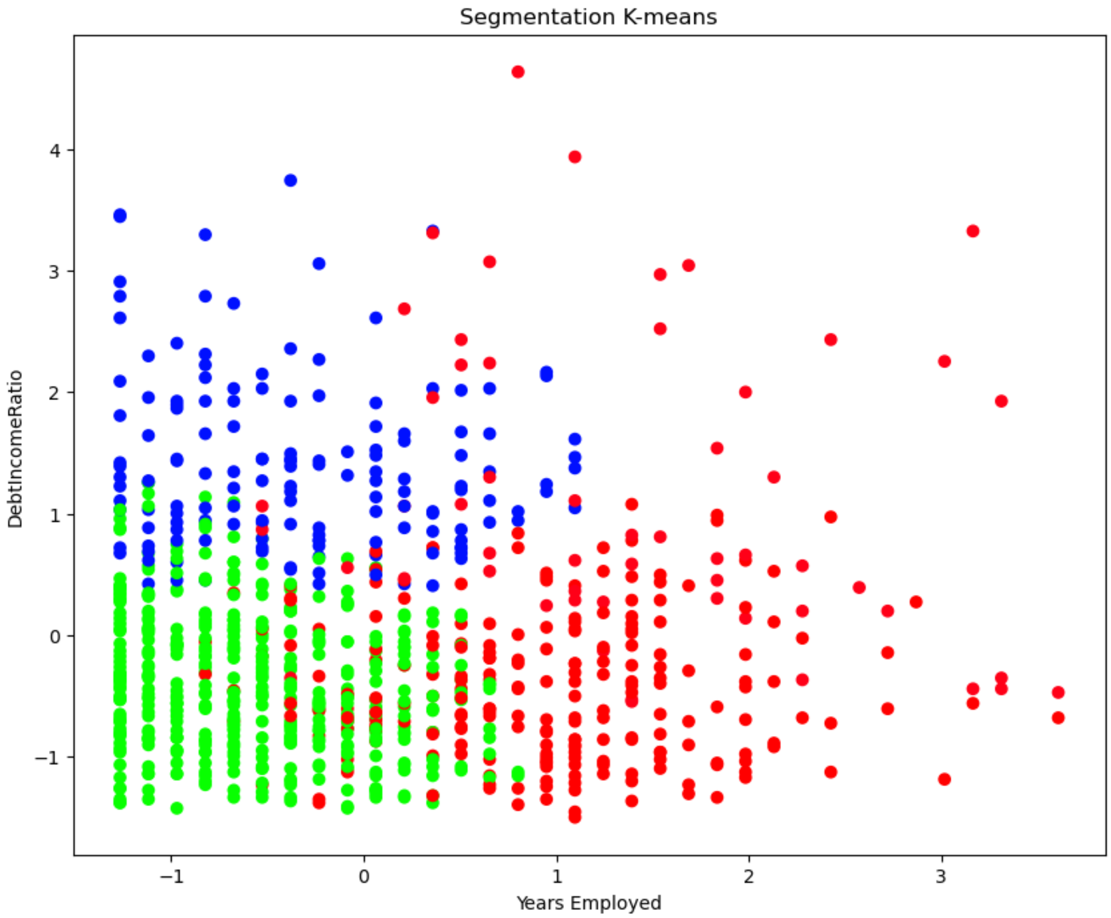
Segmentation K-means
Edu
Years Employed
3
2
1
0
−1
0
1
2
3

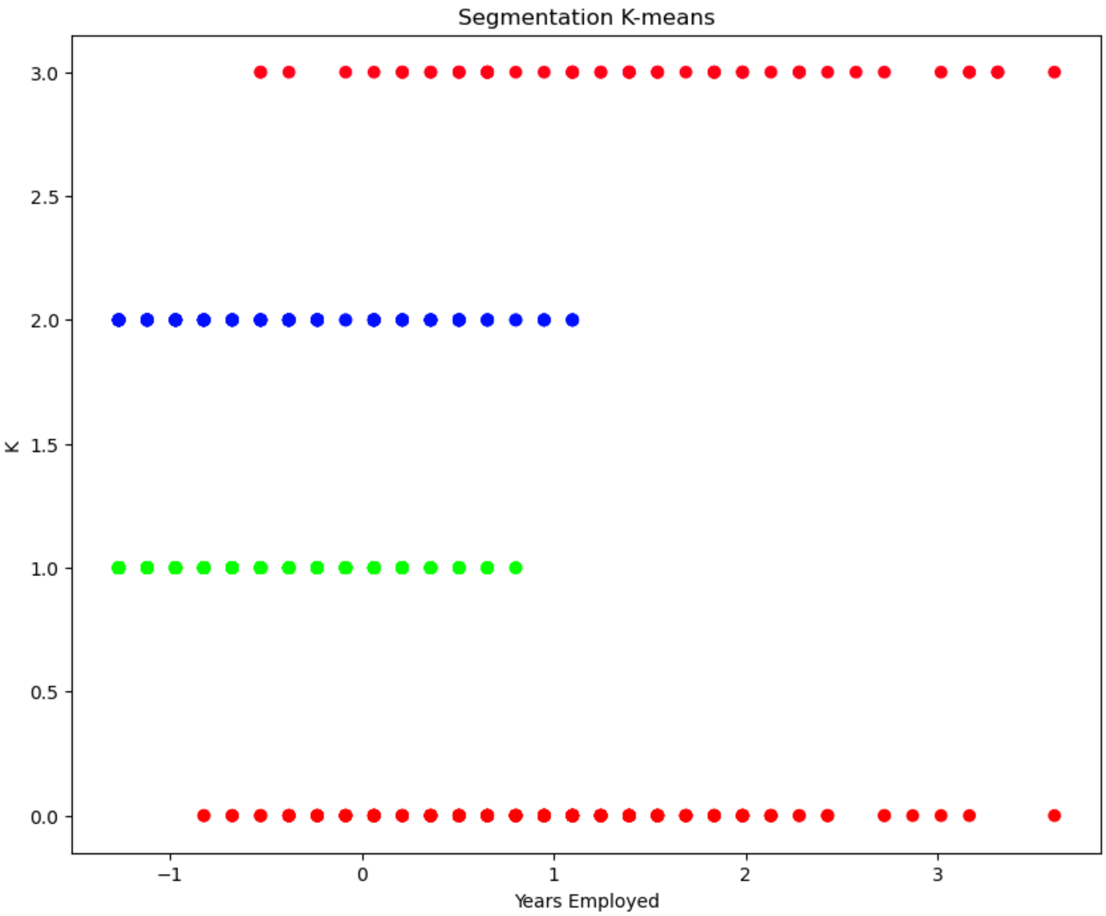


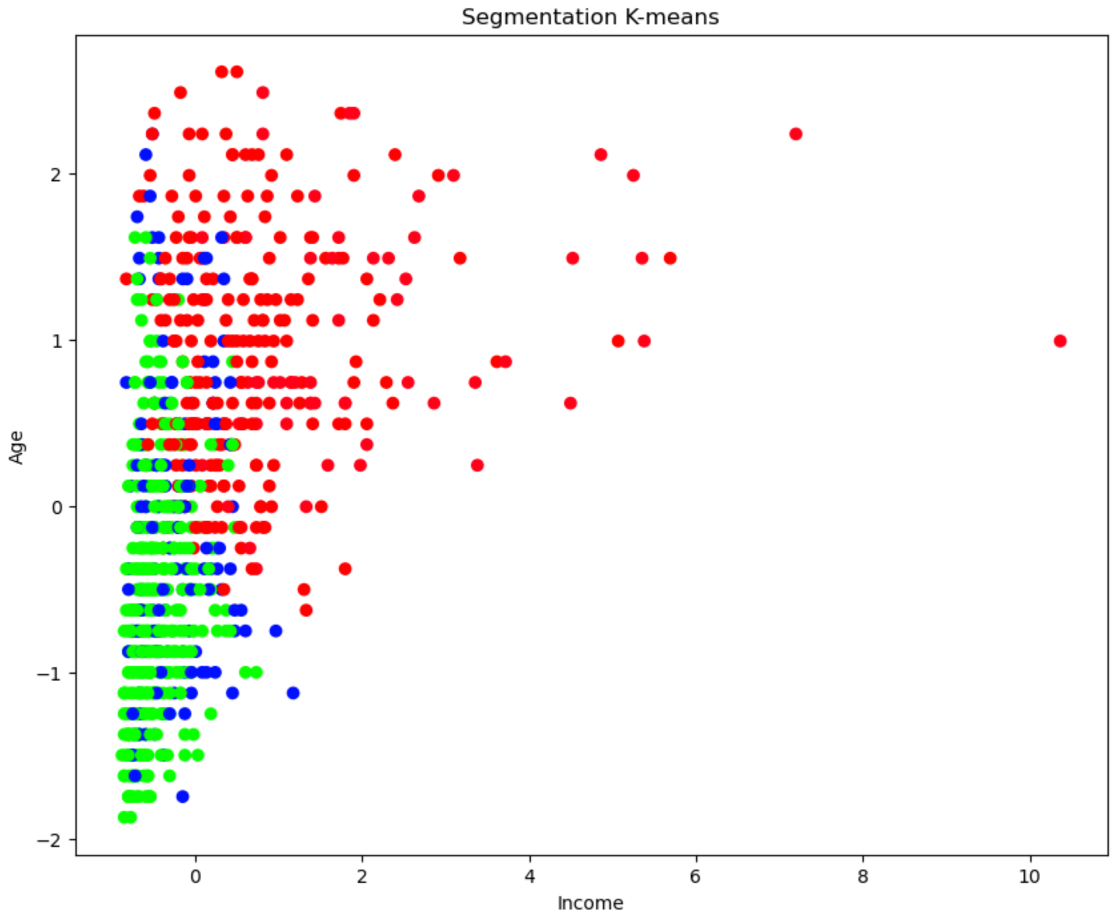
Segmentation K-means
10
8
6
4
2
0
Income
−1
0
1
2
3
Years Employed


Segmentation K-means
Card Debt
Years Employed
8
6
4
2
0
−1
0
1
2
3

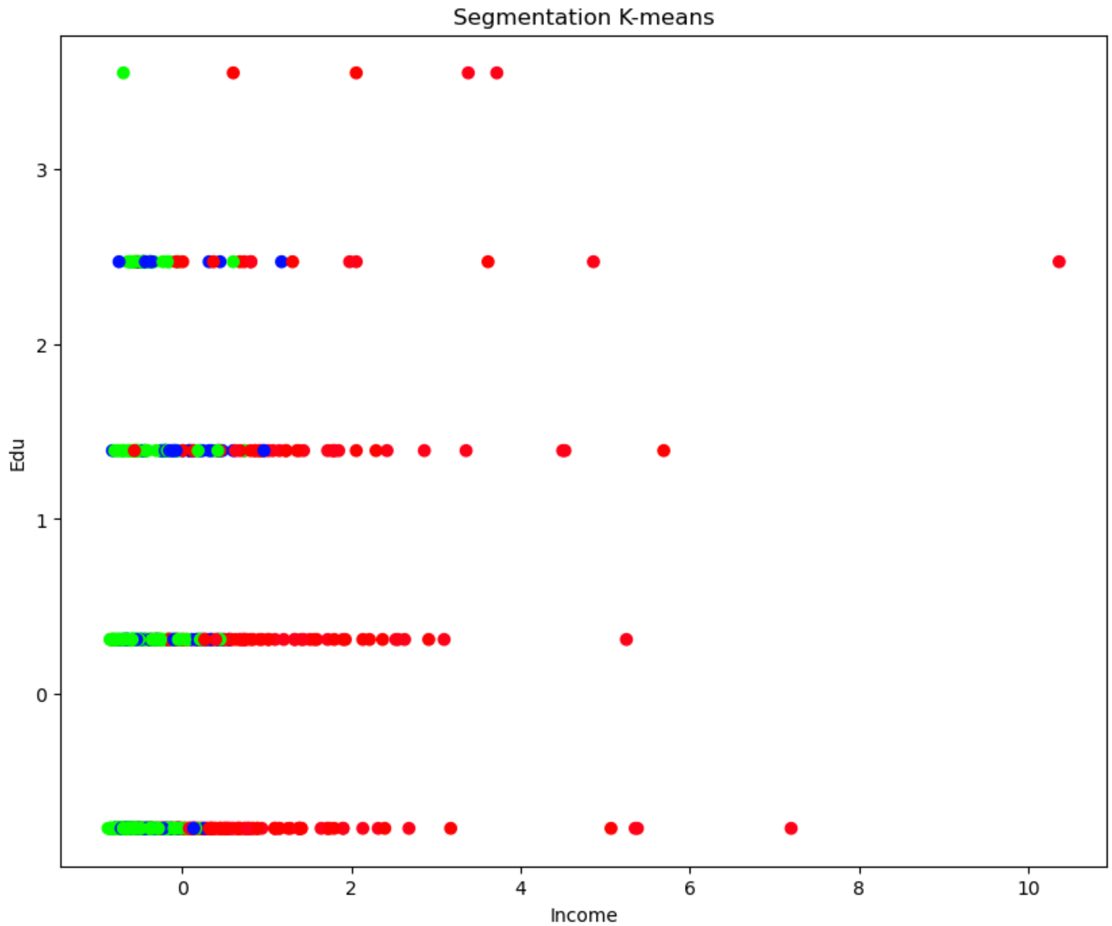

Segmentation K-means
Other Debt
Years Employed
8
6
4
2
0
−1
0
1
2
3

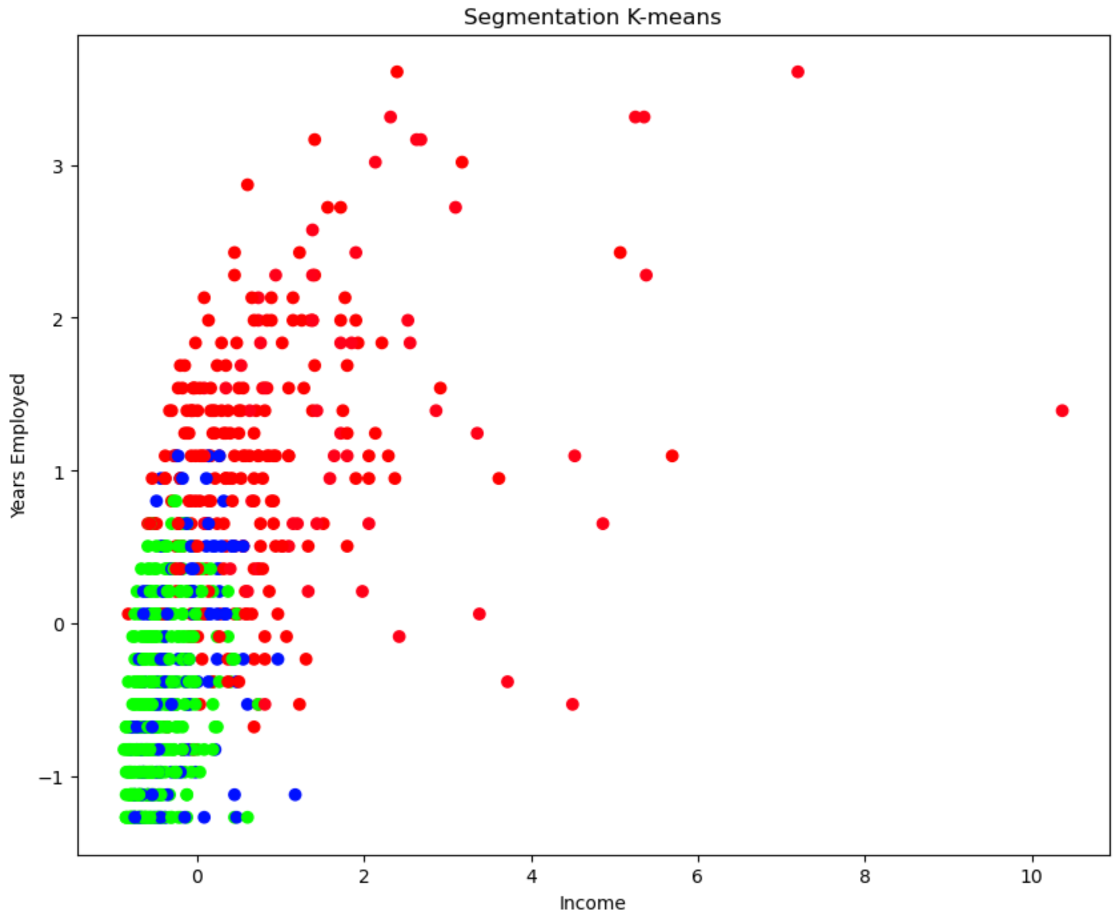

Segmentation K-means
DebtIncomeRatio
Years Employed
4
3
2
1
0
−1
−1
0
1
2
3

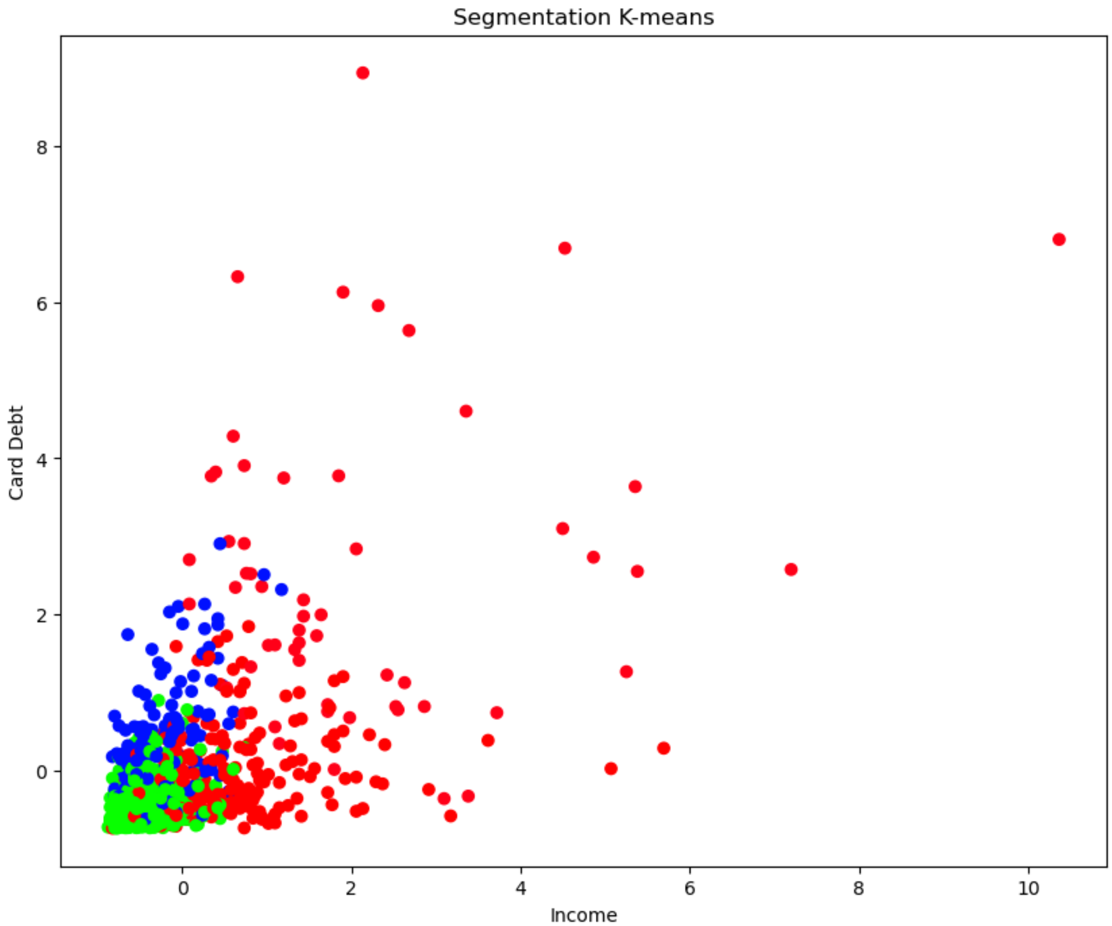

Segmentation K-means
3.0
2.5
2.0
1.5
1.0
0.5
0.0
K
−1
0
1
2
3
Years Employed

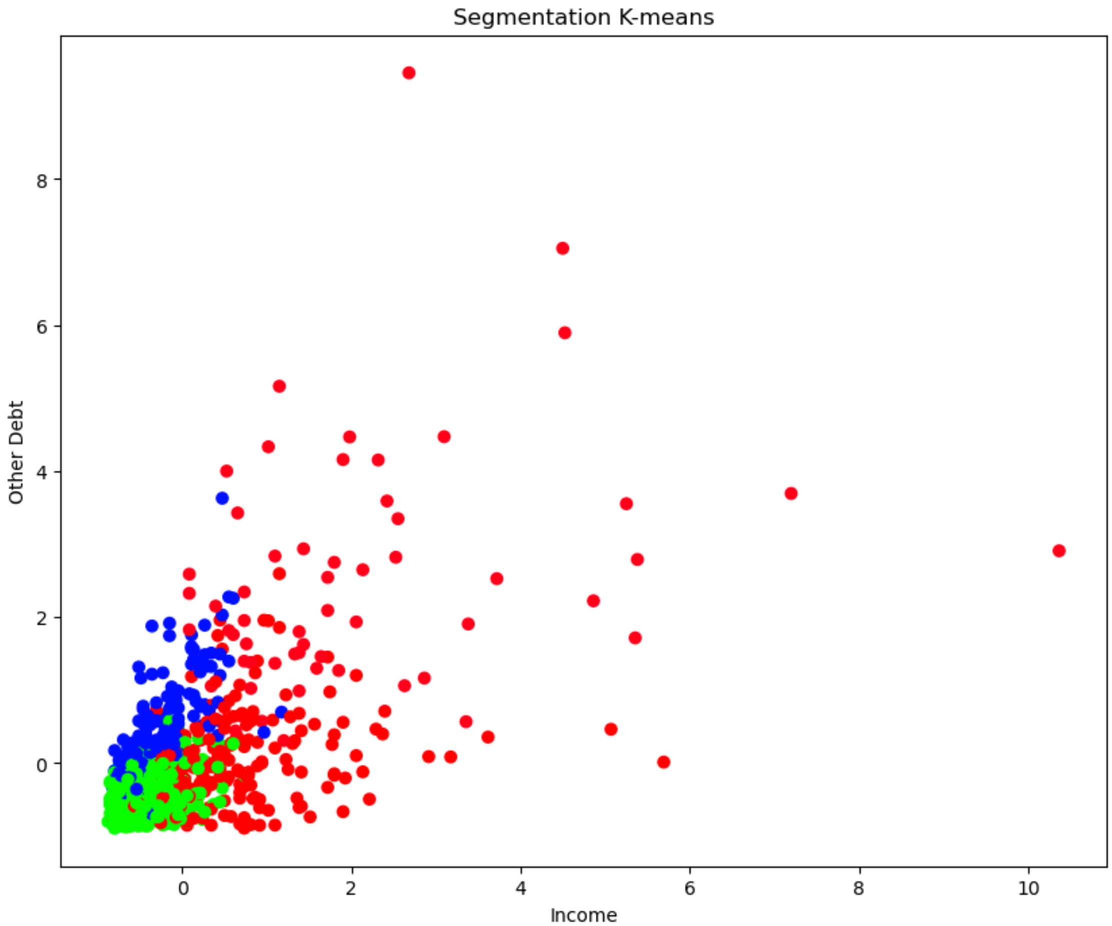


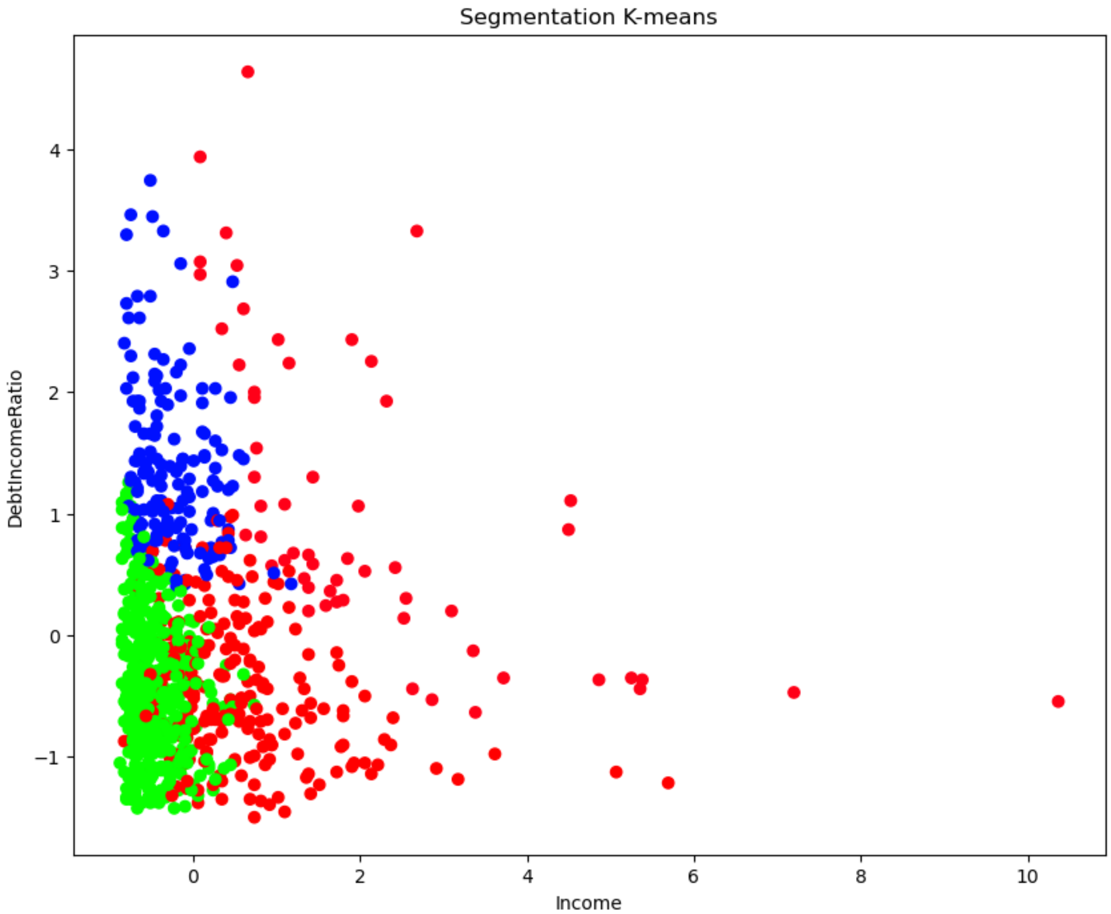
Segmentation K-means
Age
Income
2
1
0
−1
−2
0
2
4
6
8
10

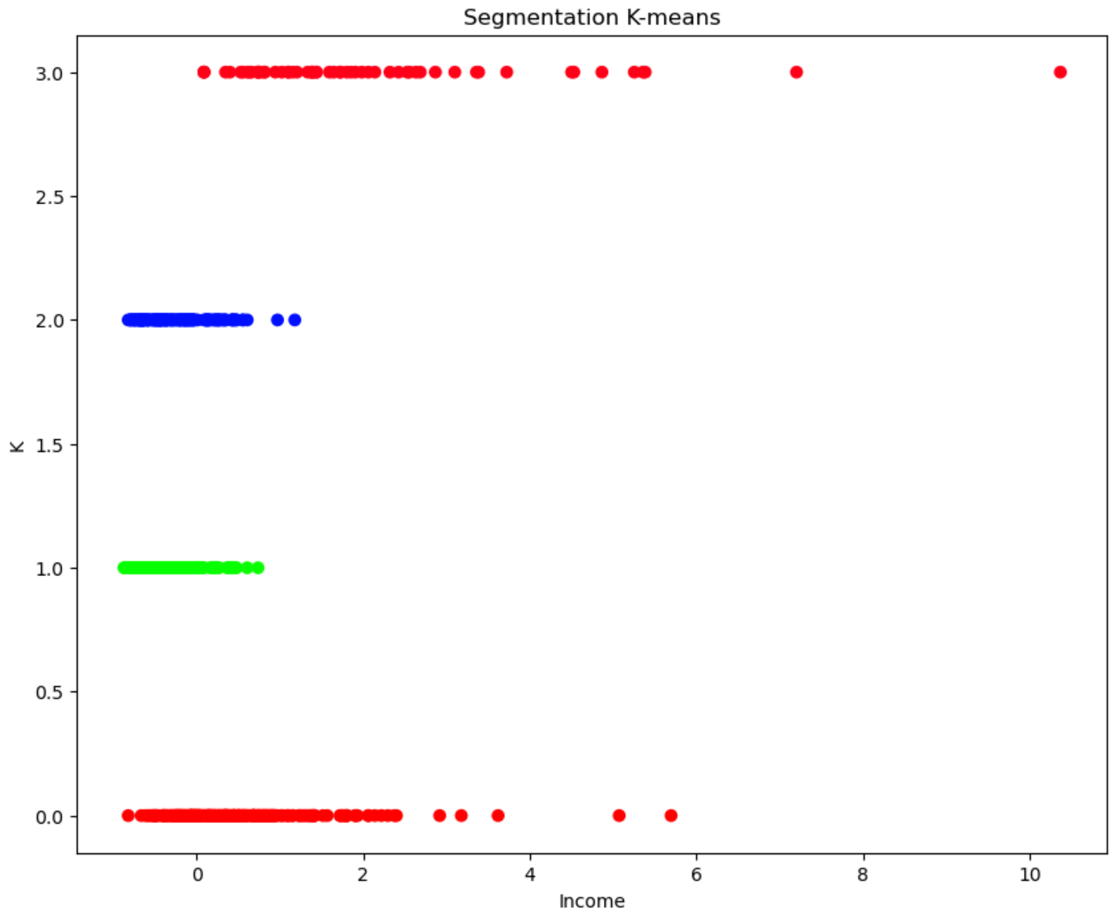

Segmentation K-means
Edu
Income
3
2
1
0
0
2
4
6
8
10

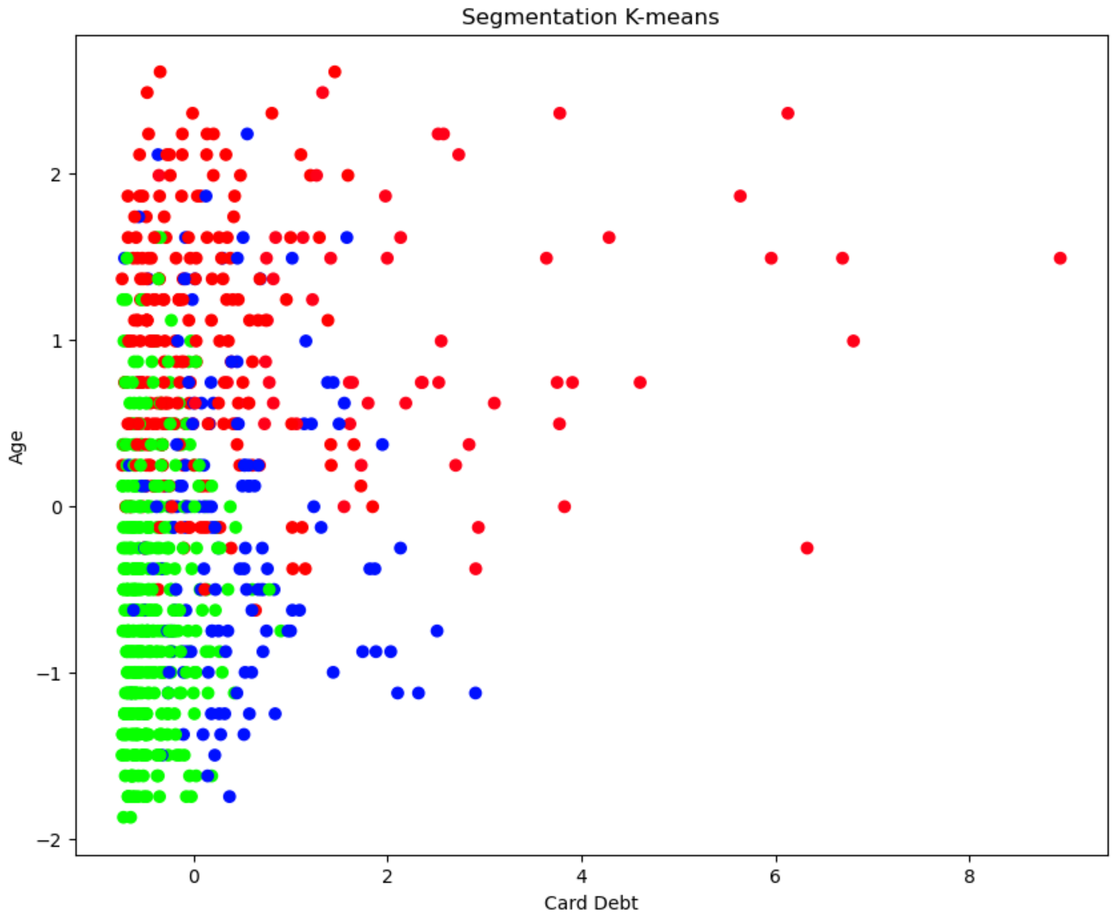

Segmentation K-means
Years Employed
Income
3
2
1
0
−1
0
2
4
6
8
10

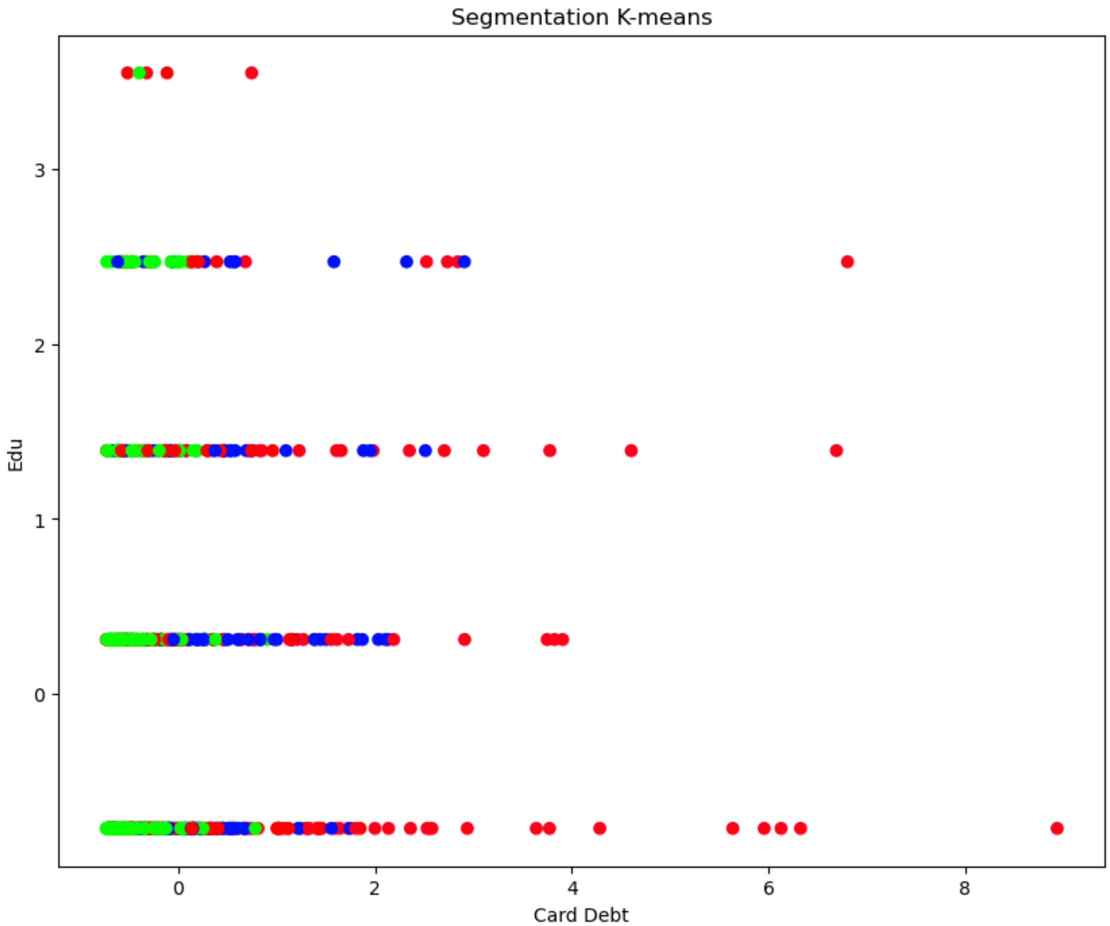

Segmentation K-means
Card Debt
Income
8
6
4
2
0
0
2
4
6
8
10

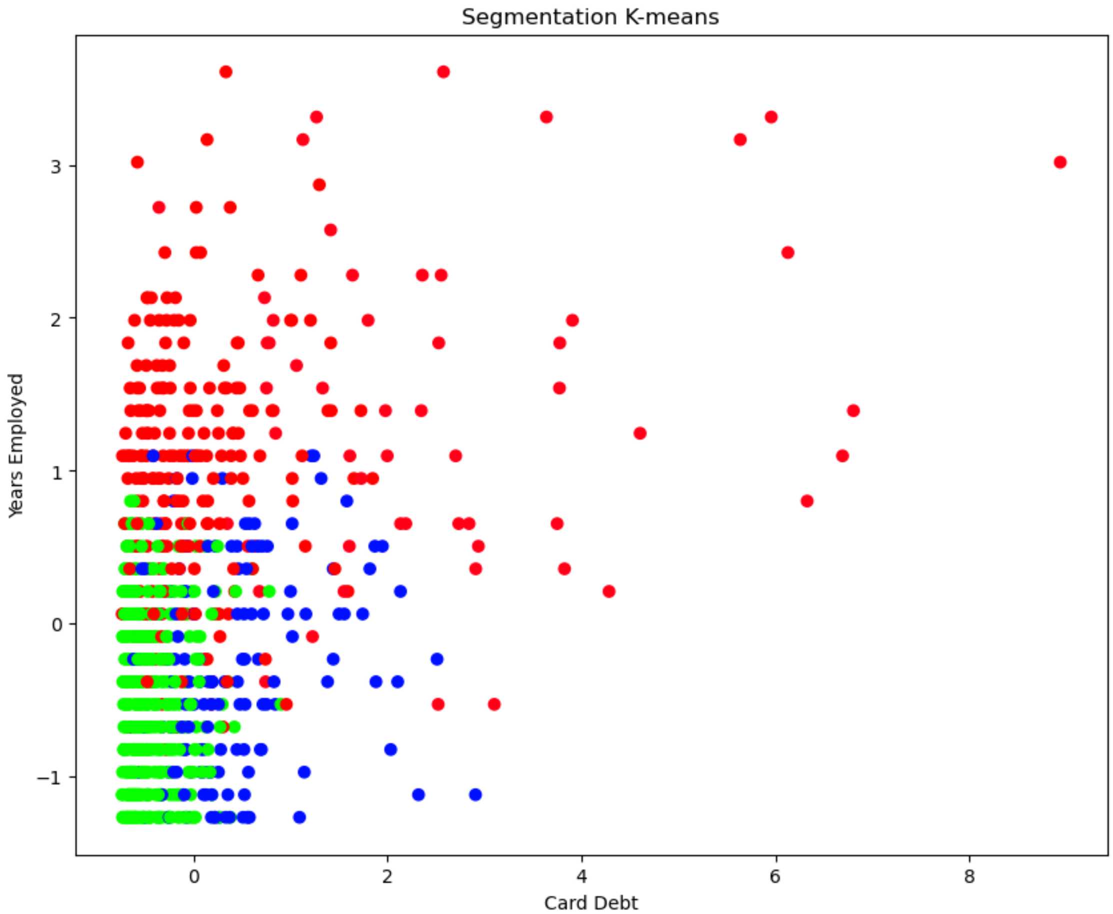

Segmentation K-means
Other Debt
Income
0
2
4
6
8
10


Segmentation K-means
DebtIncomeRatio
Income
4
3
2
1
0
−1
0
2
4
6
8
10

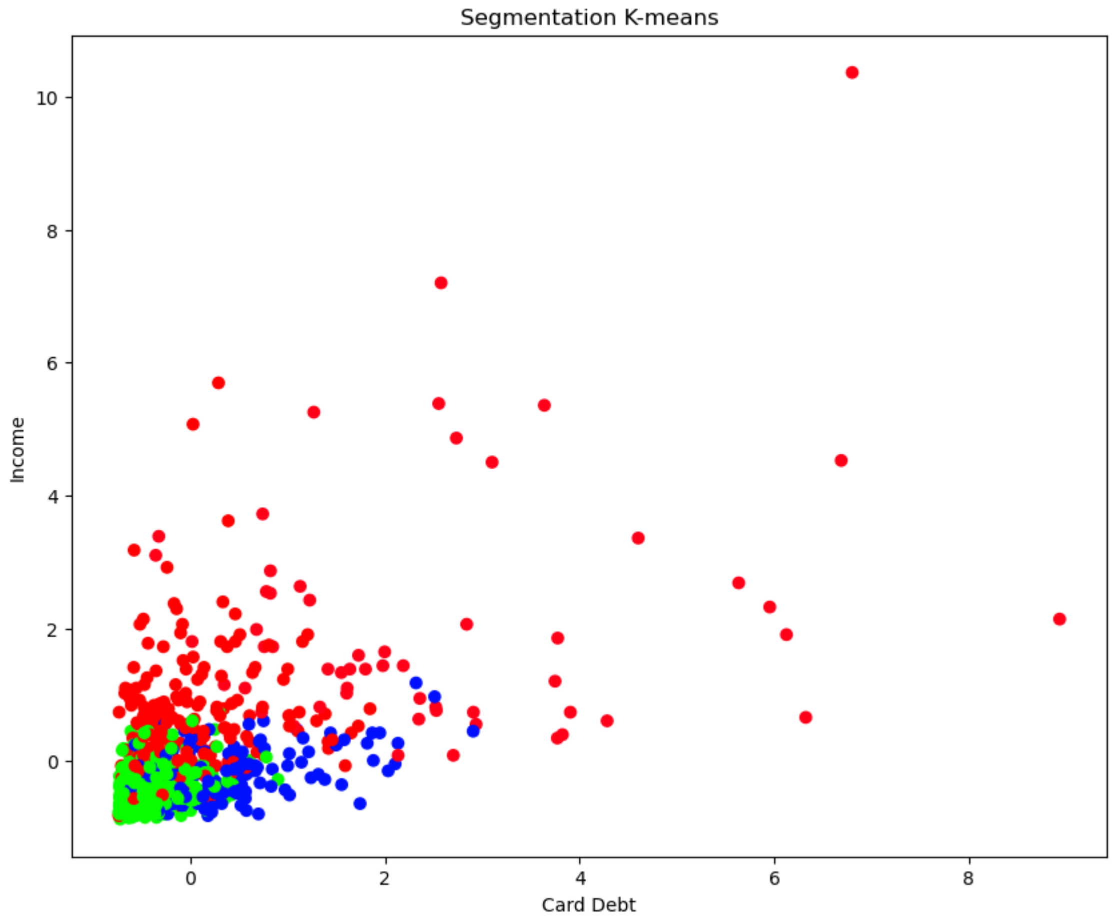

Segmentation K-means
3.0
2.5
2.0
1.5
1.0
0.5
0.0
K
0
2
4
6
8
10
Income

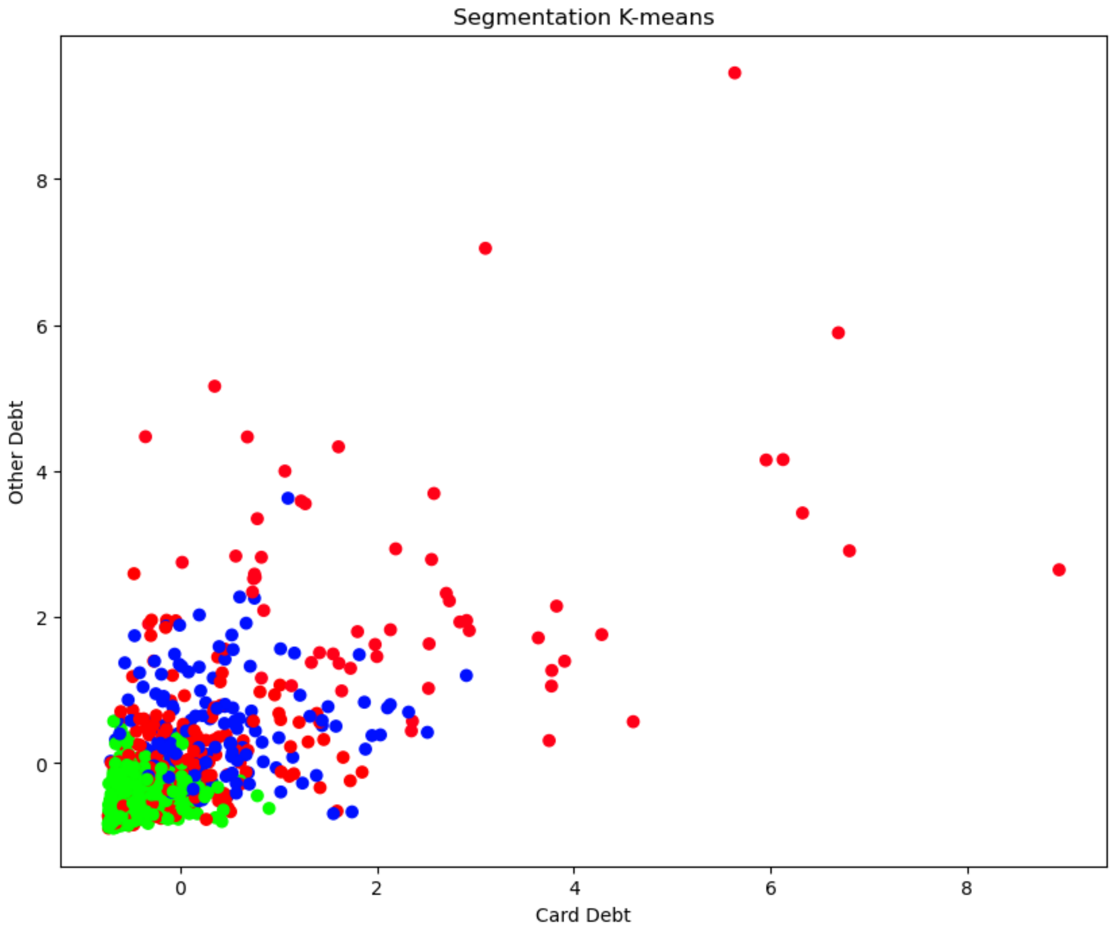

Segmentation K-means
Age
Card Debt
2
1
0
−1
−2
0
2
4
6
8


Segmentation K-means
Edu
Card Debt
0
1
2
3
0
2
4
6
8

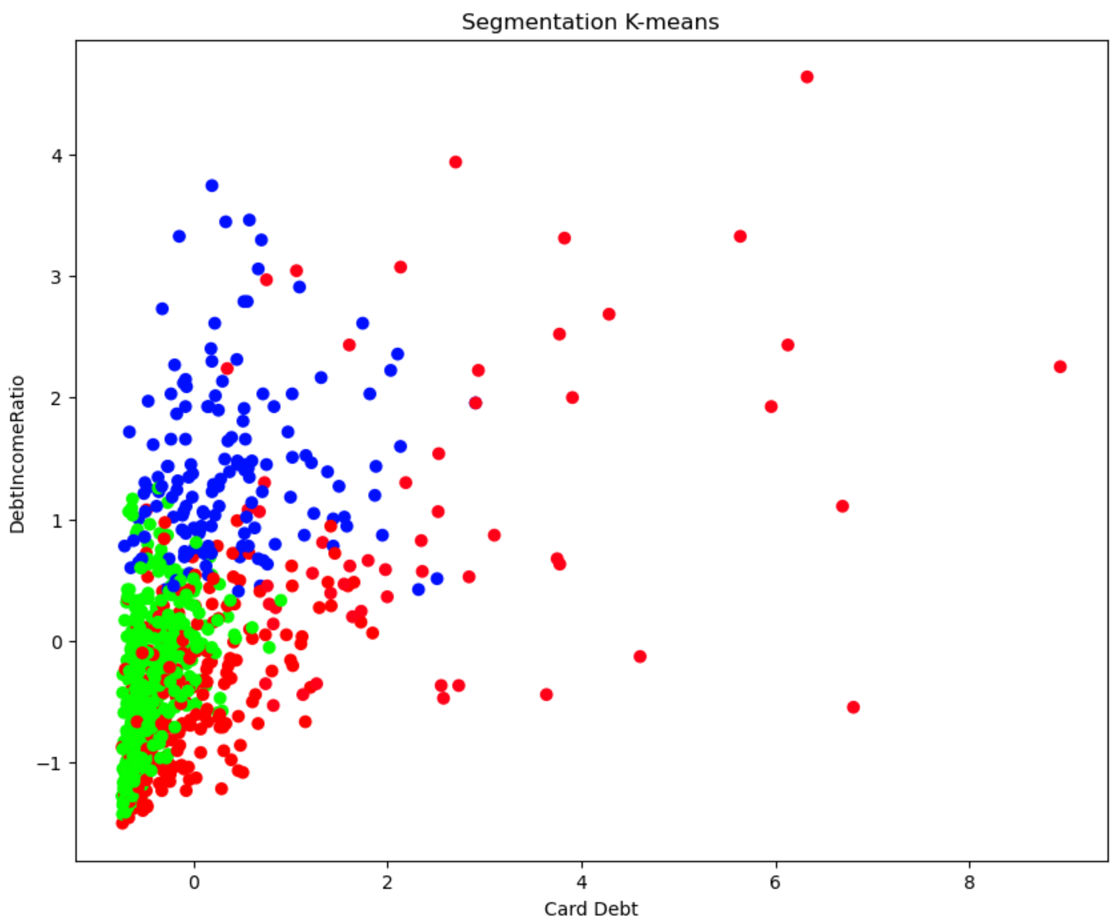


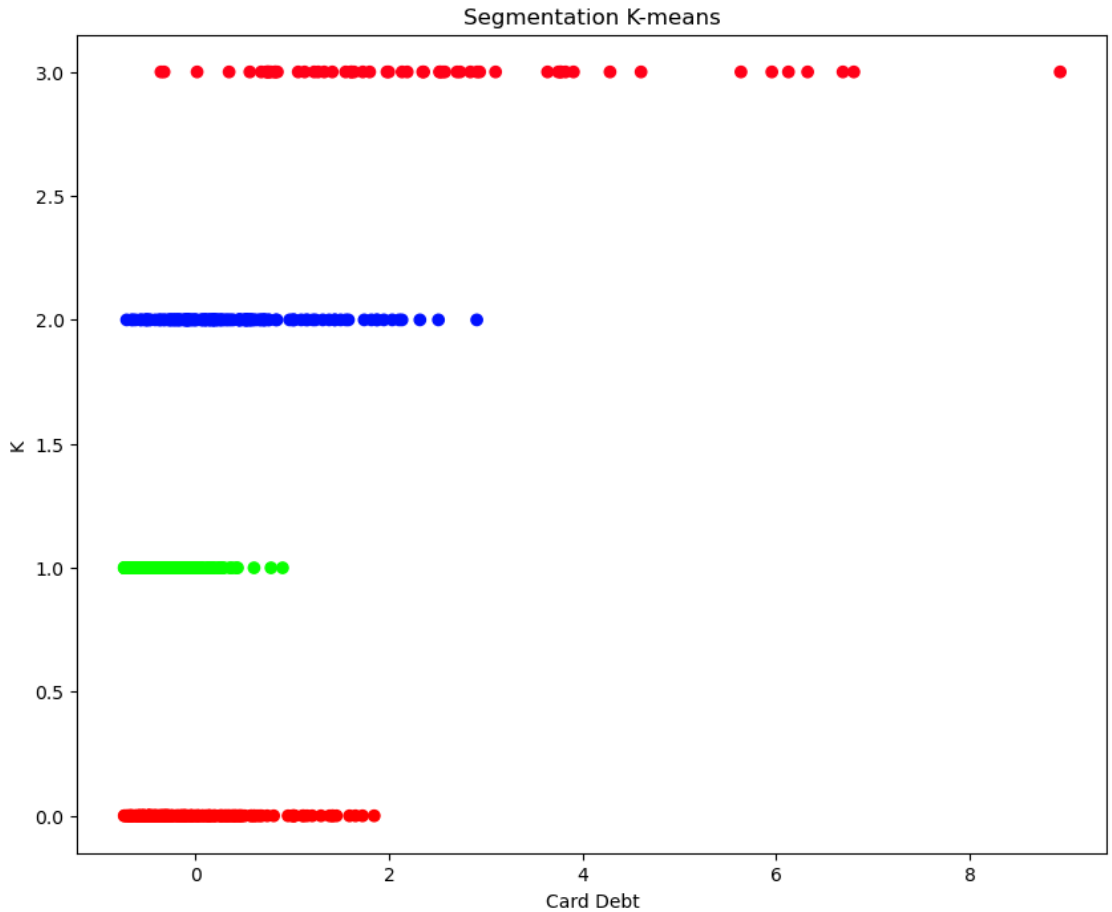
Segmentation K-means
Years Employed
Card Debt
3
2
1
0
−1
0
2
4
6
8

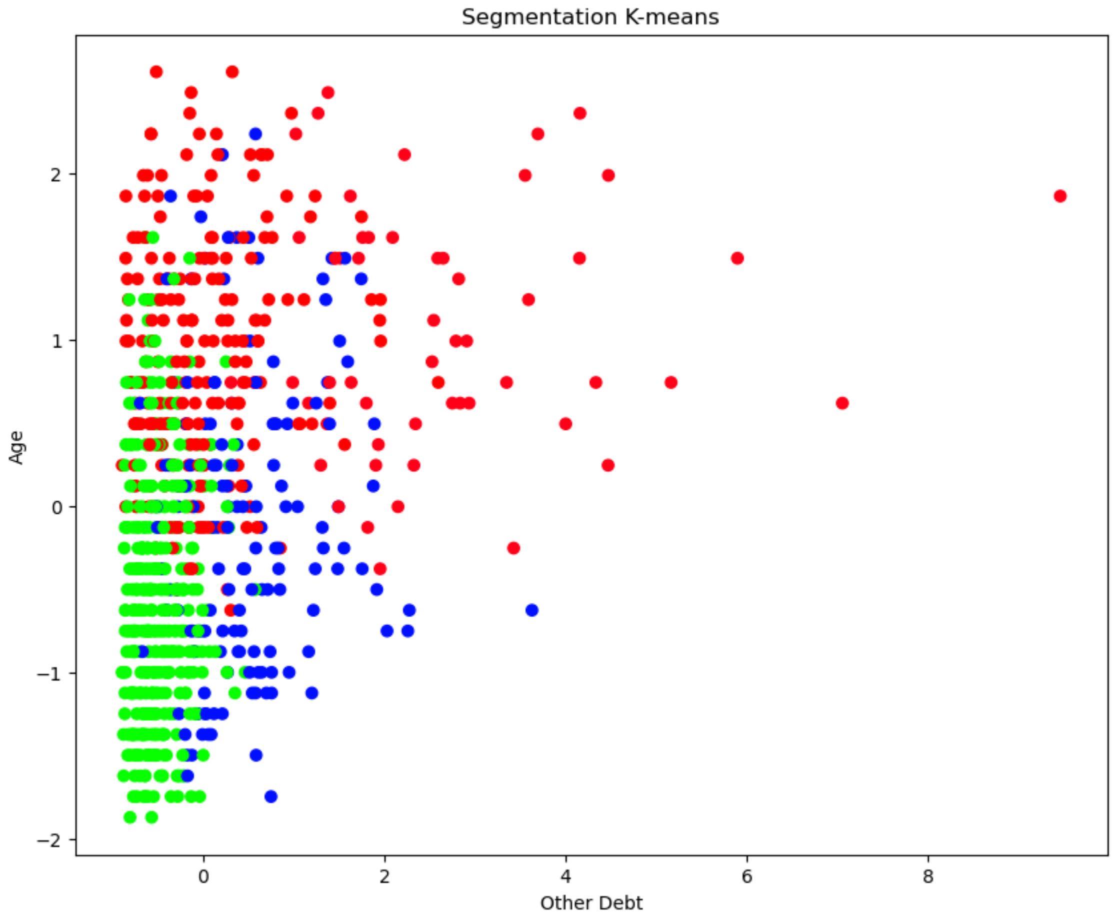

Segmentation K-means
Income
Card Debt
10
8
6
4
2
0
0
2
4
6
8

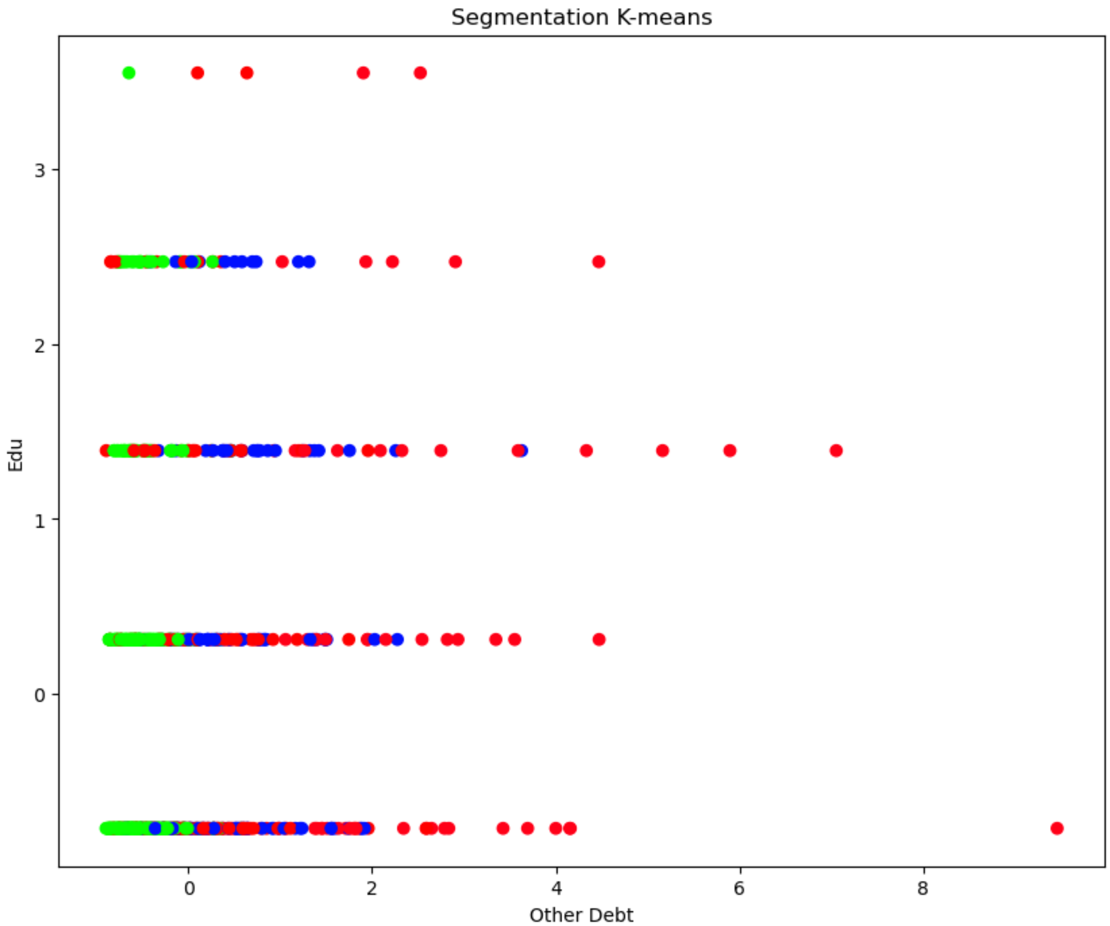


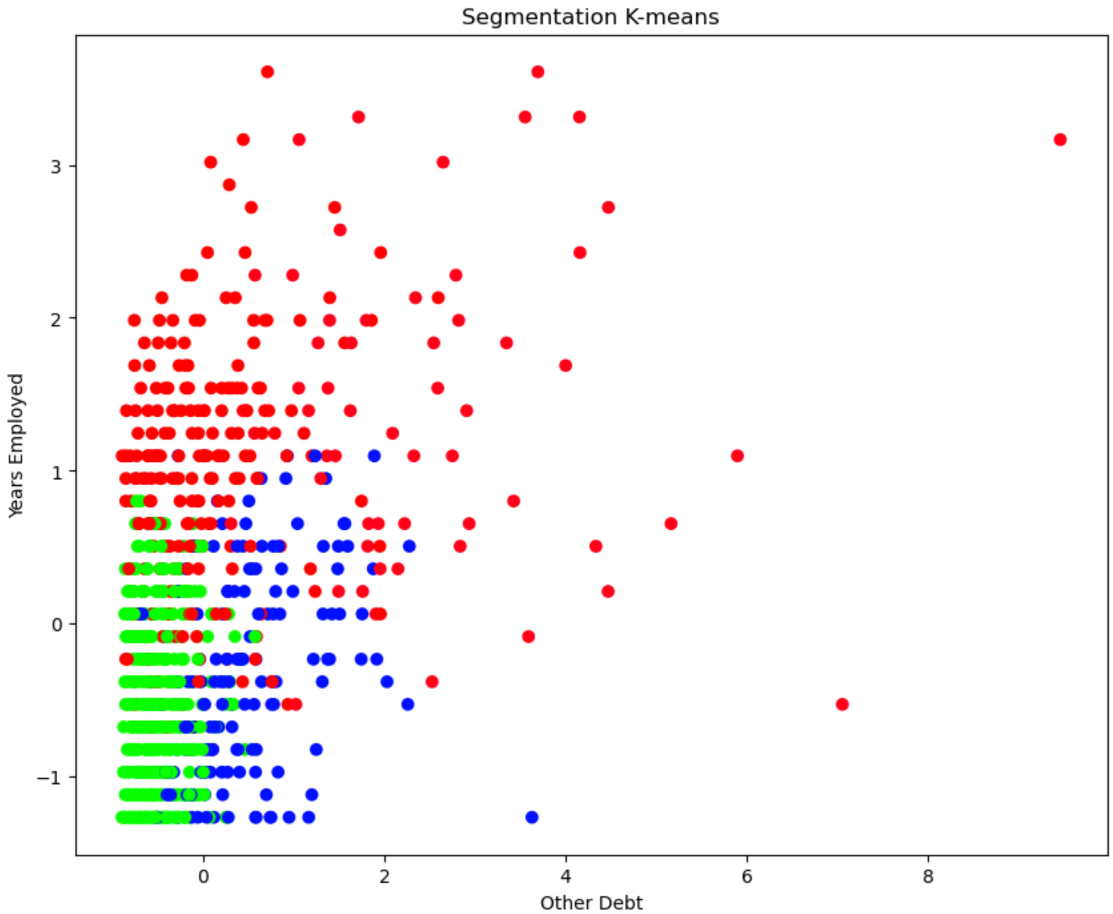
Segmentation K-means
Other Debt
Card Debt
0
2
4
6
8
0
2
4
6
8

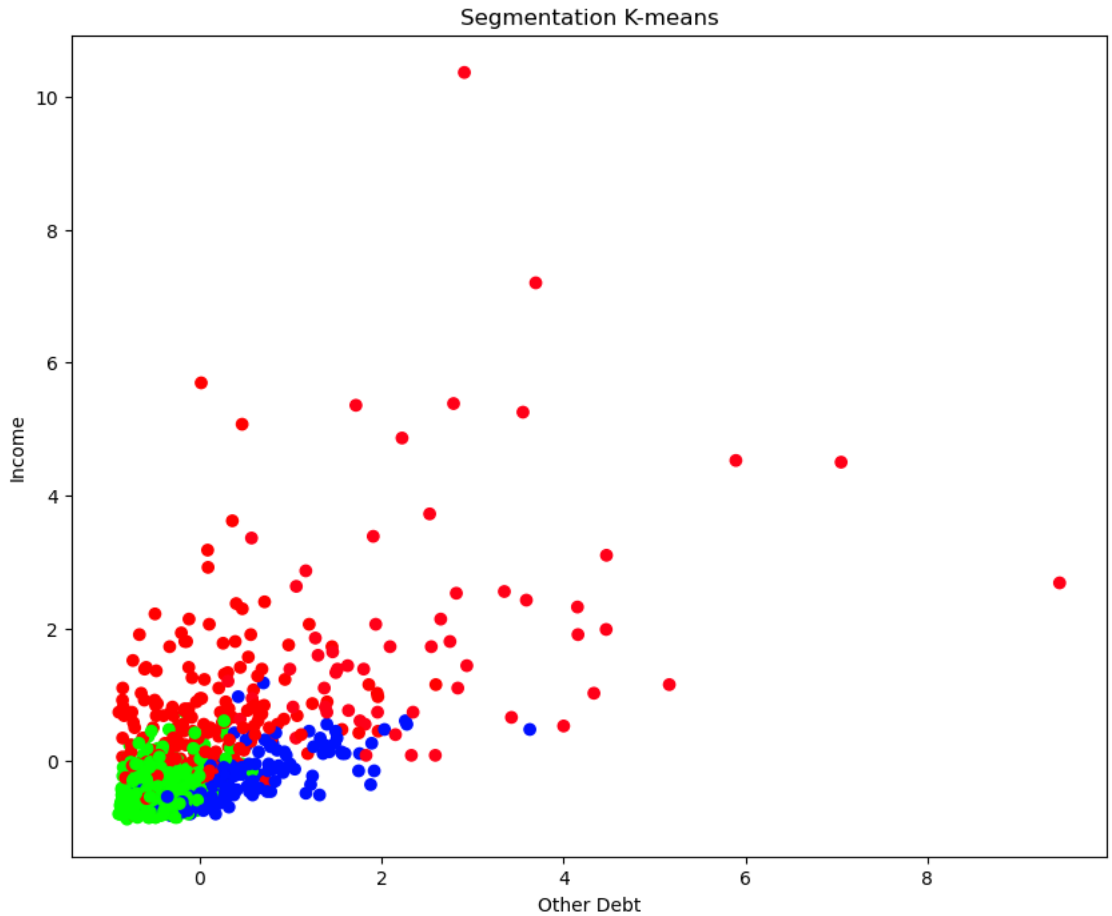

Segmentation K-means
DebtIncomeRatio
Card Debt
4
3
2
1
0
−1
0
2
4
6
8

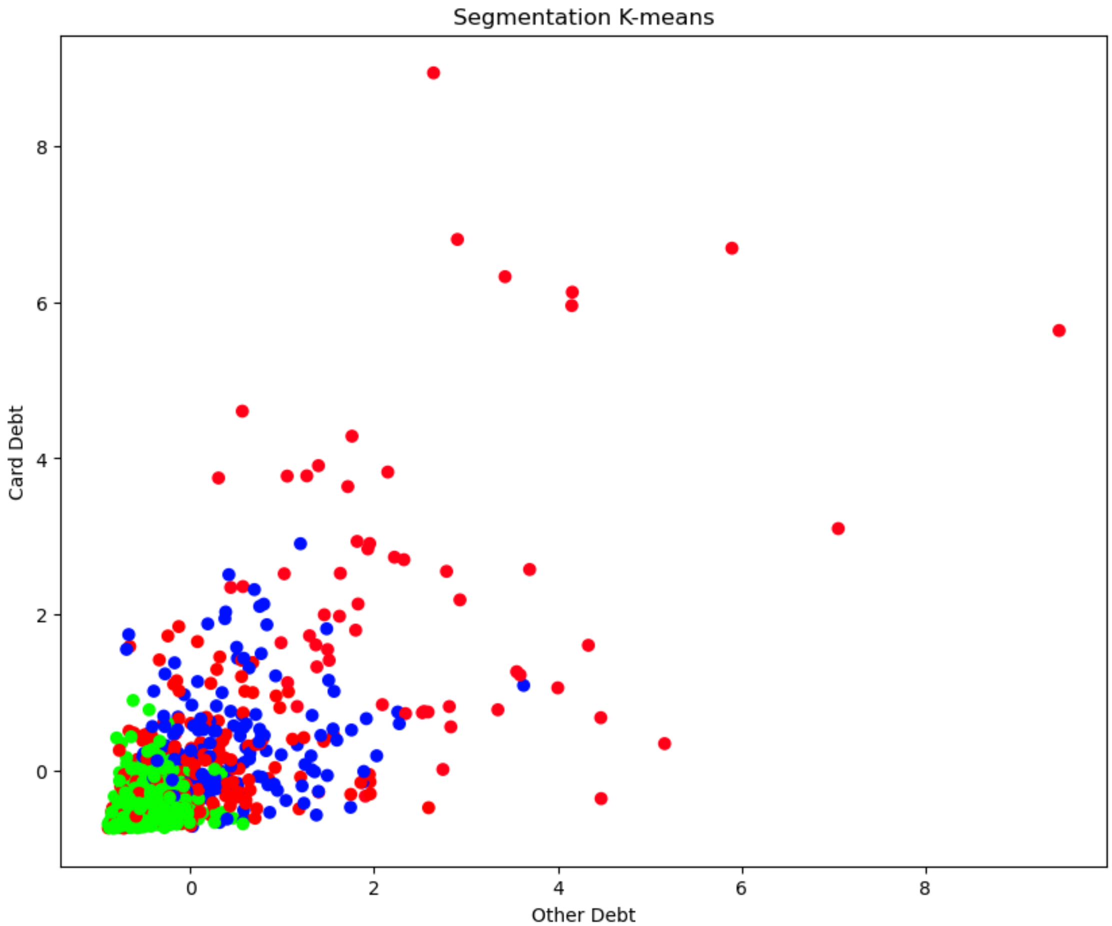

Segmentation K-means
3.0
2.5
2.0
1.5
1.0
0.5
0.0
K
0
2
4
6
8
Card Debt

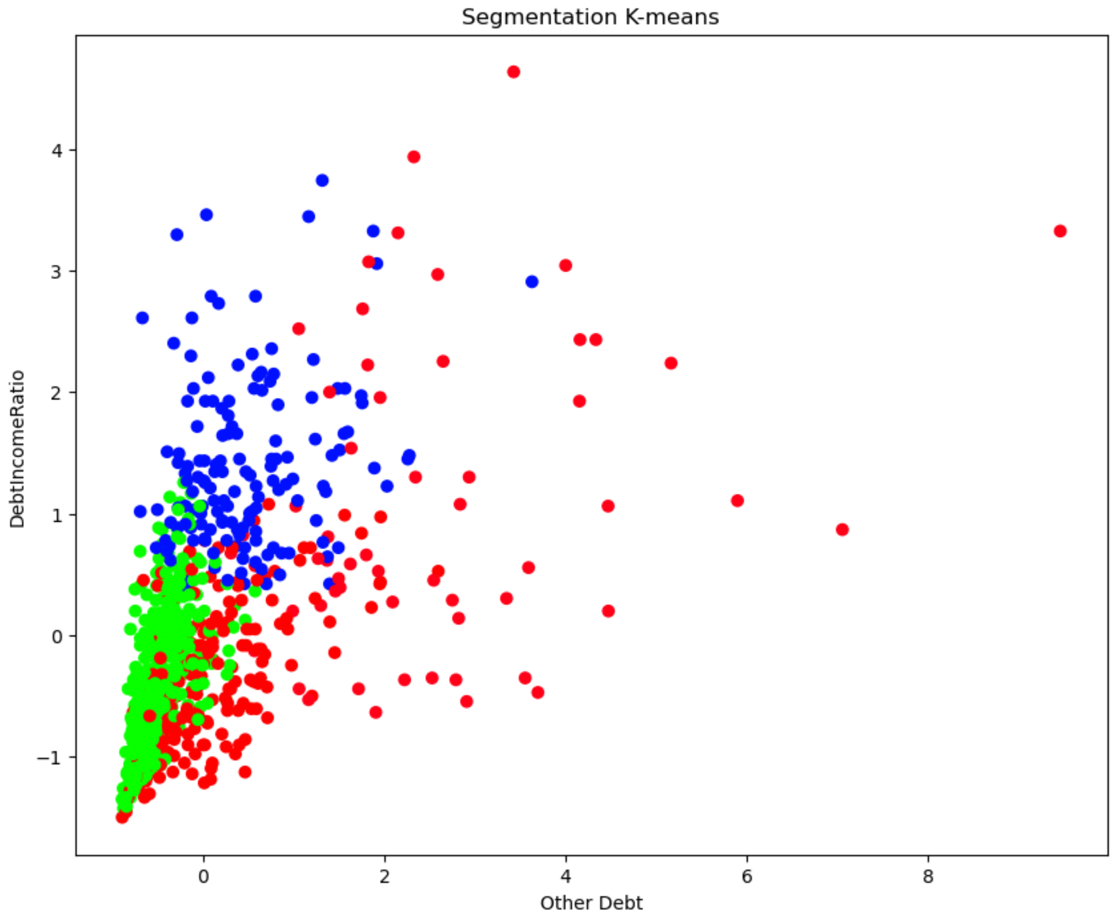

Segmentation K-means
Age
Other Debt
2
1
0
−1
−2
0
2
4
6
8


Segmentation K-means
Edu
Other Debt
3
2
1
0
0
2
4
6
8

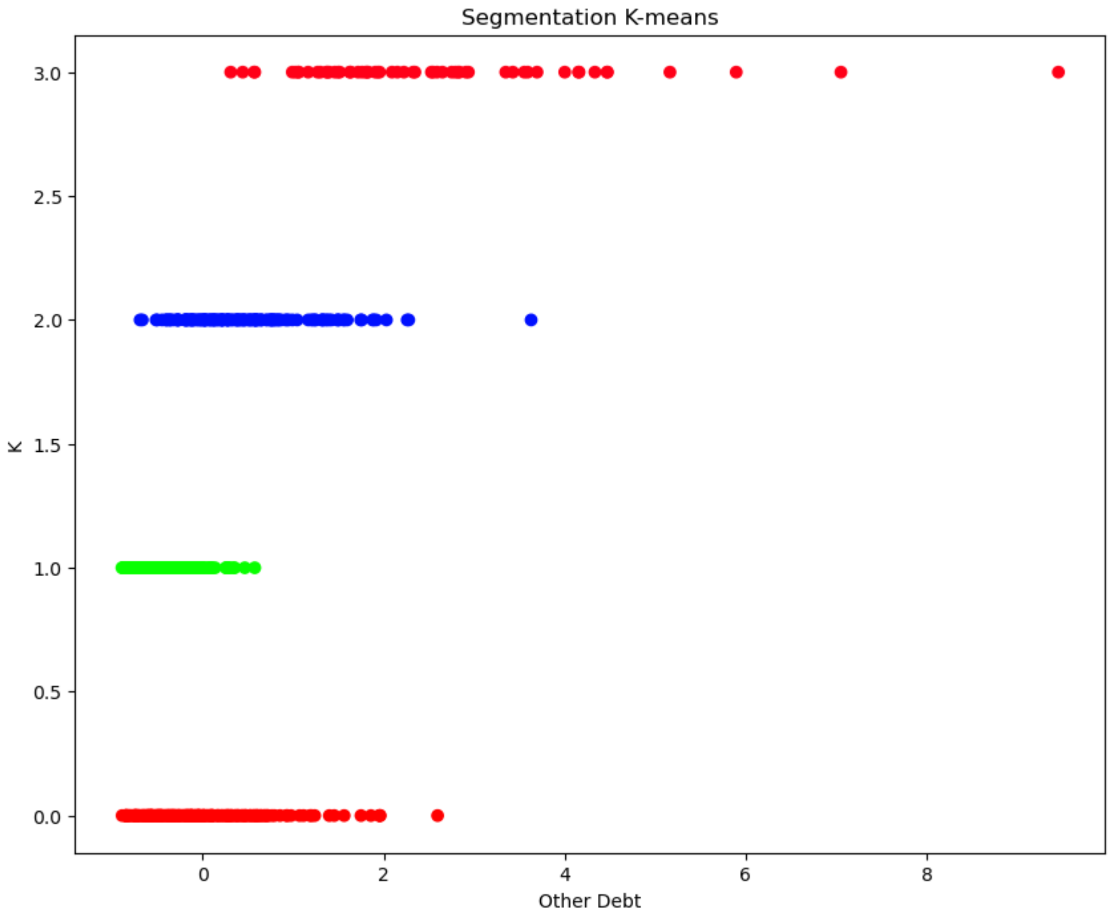

Segmentation K-means
Years Employed
Other Debt
3
2
1
0
−1
0
2
4
6
8

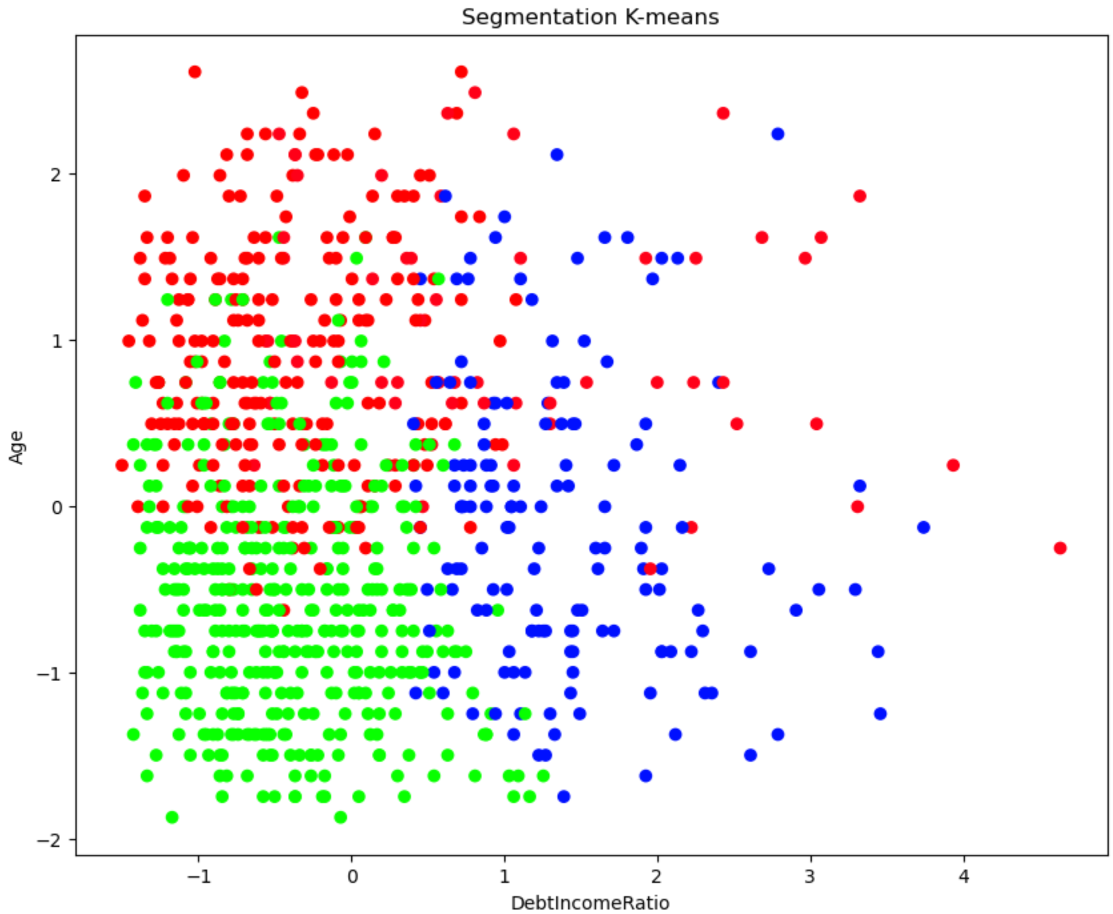

Segmentation K-means
Income
Other Debt
10
8
6
4
2
0
0
2
4
6
8

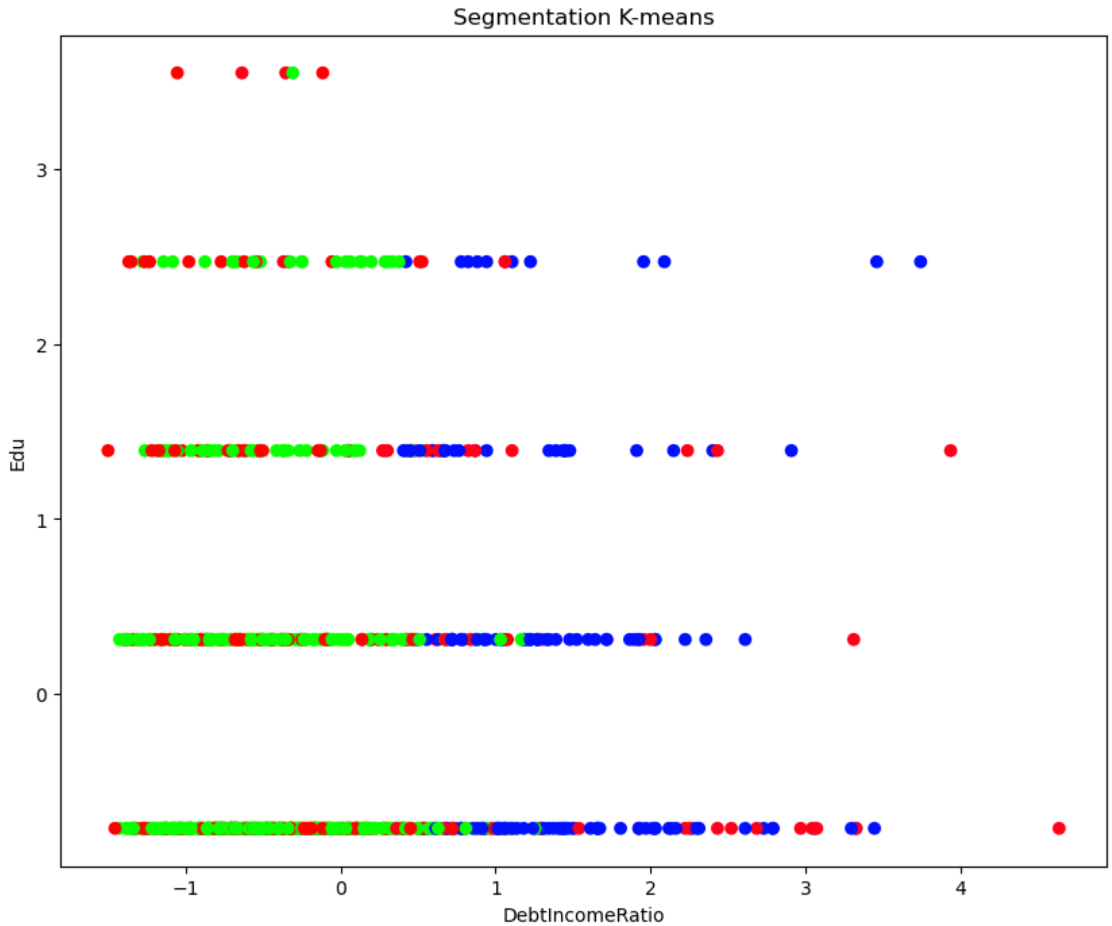

Segmentation K-means
Card Debt
Other Debt
0
2
4
6
8
0
2
4
6
8

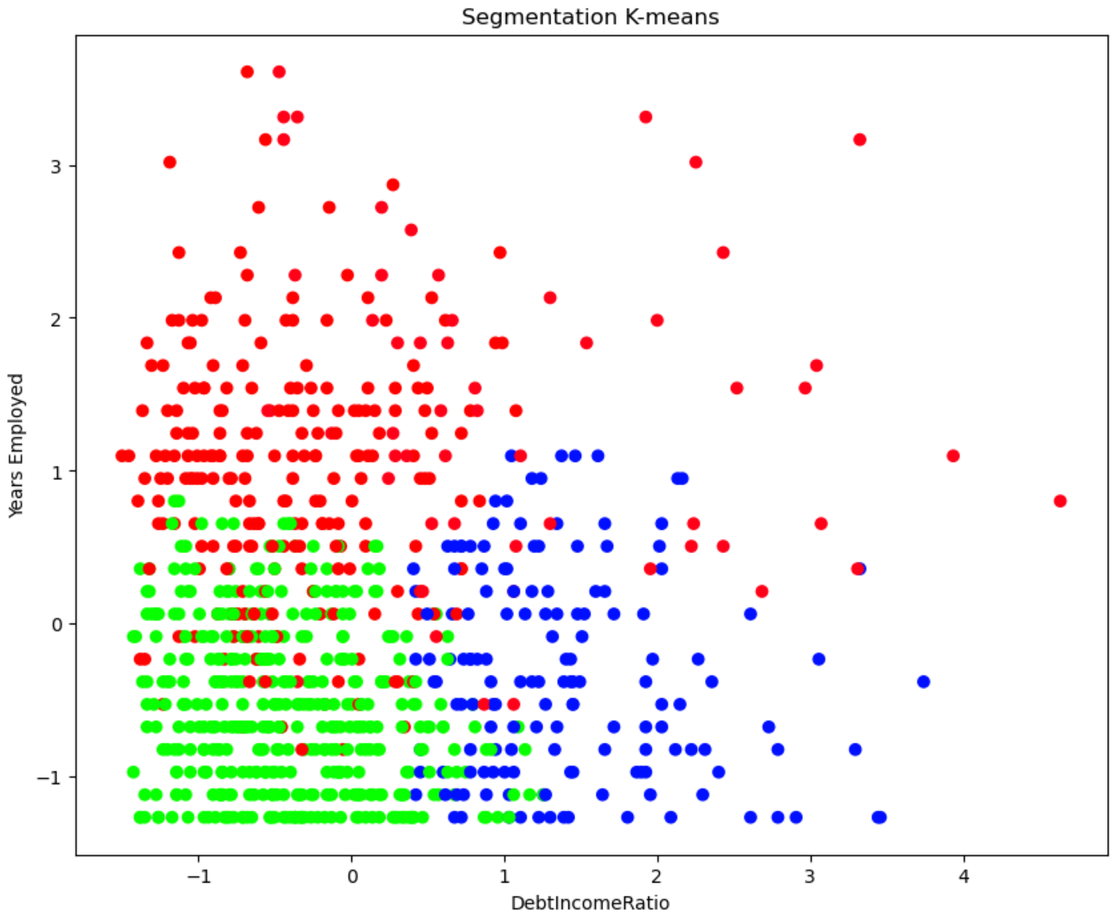


Segmentation K-means
DebtIncomeRatio
Other Debt

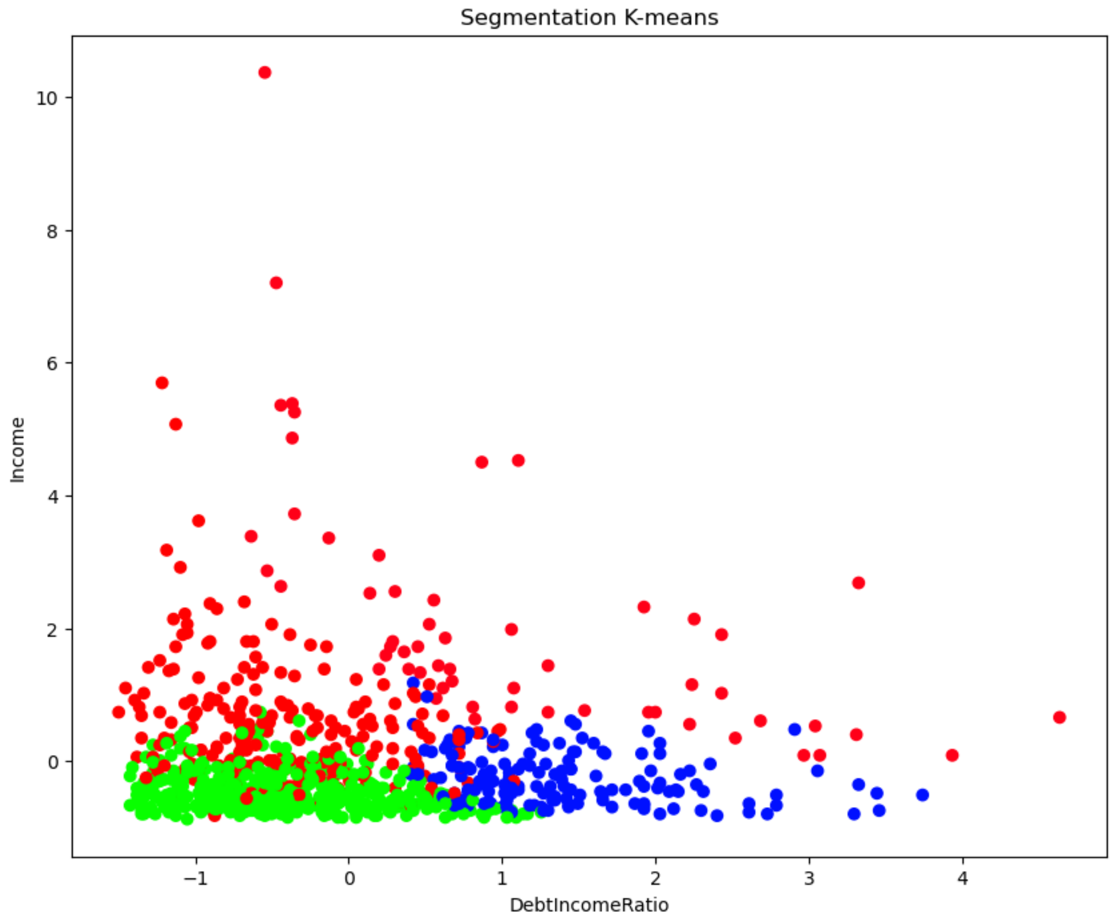

Segmentation K-means
3.0
2.5
2.0
1.5
1.0
0.5
0.0
K
0
2
4
6
8
Other Debt


Segmentation K-means
Age
DebtIncomeRatio
2
1
0
−1
−2
−1
0
1
2
3
4

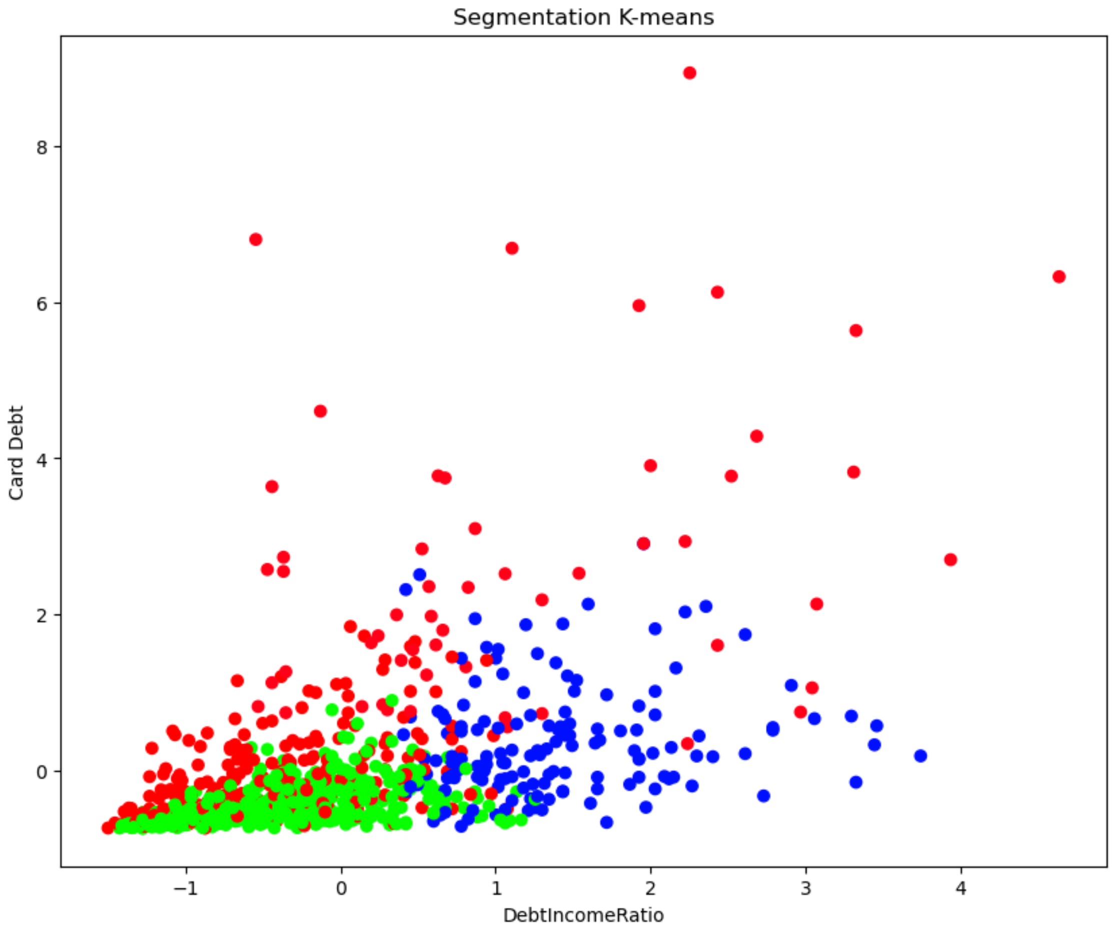

Segmentation K-means
Edu
DebtIncomeRatio
3
2
1
0
−1
0
1
2
3
4



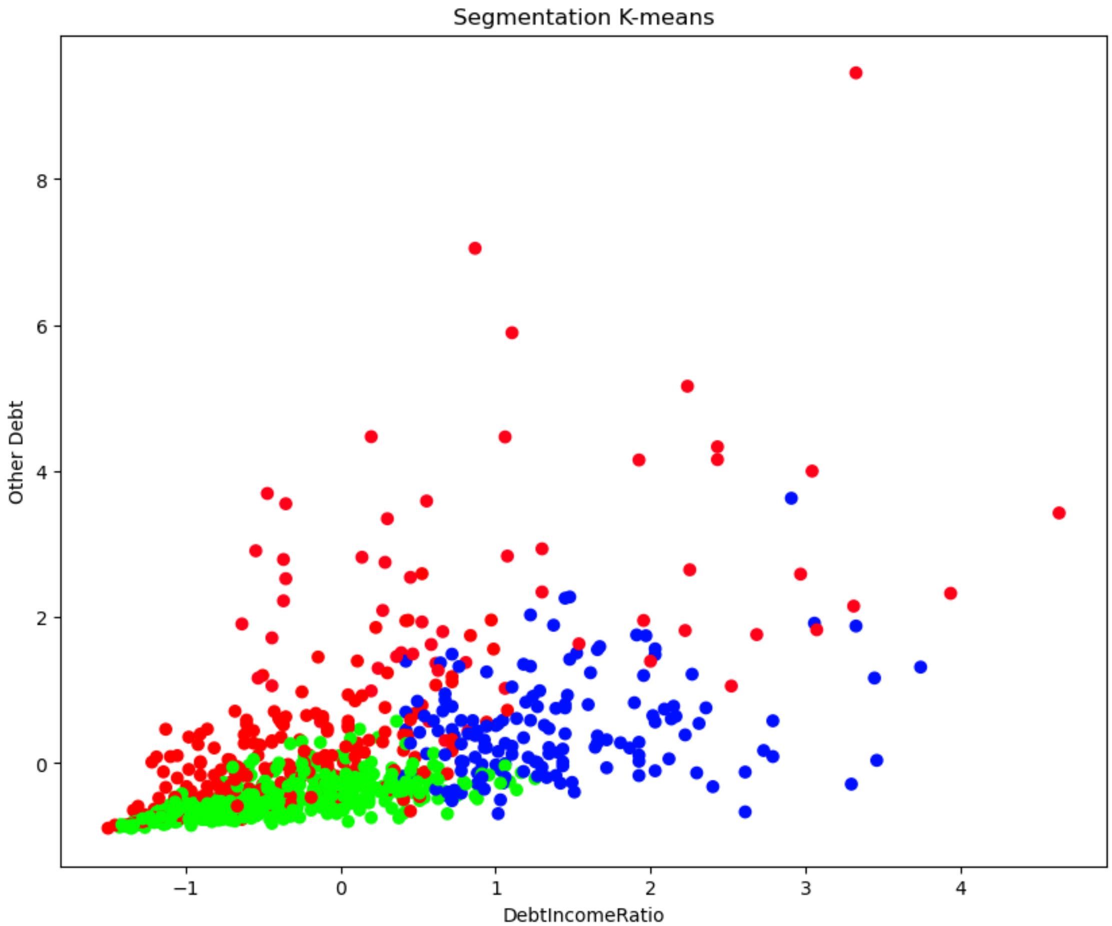
Segmentation K-means
Years Employed
DebtIncomeRatio
−1
0
1
2
3
4

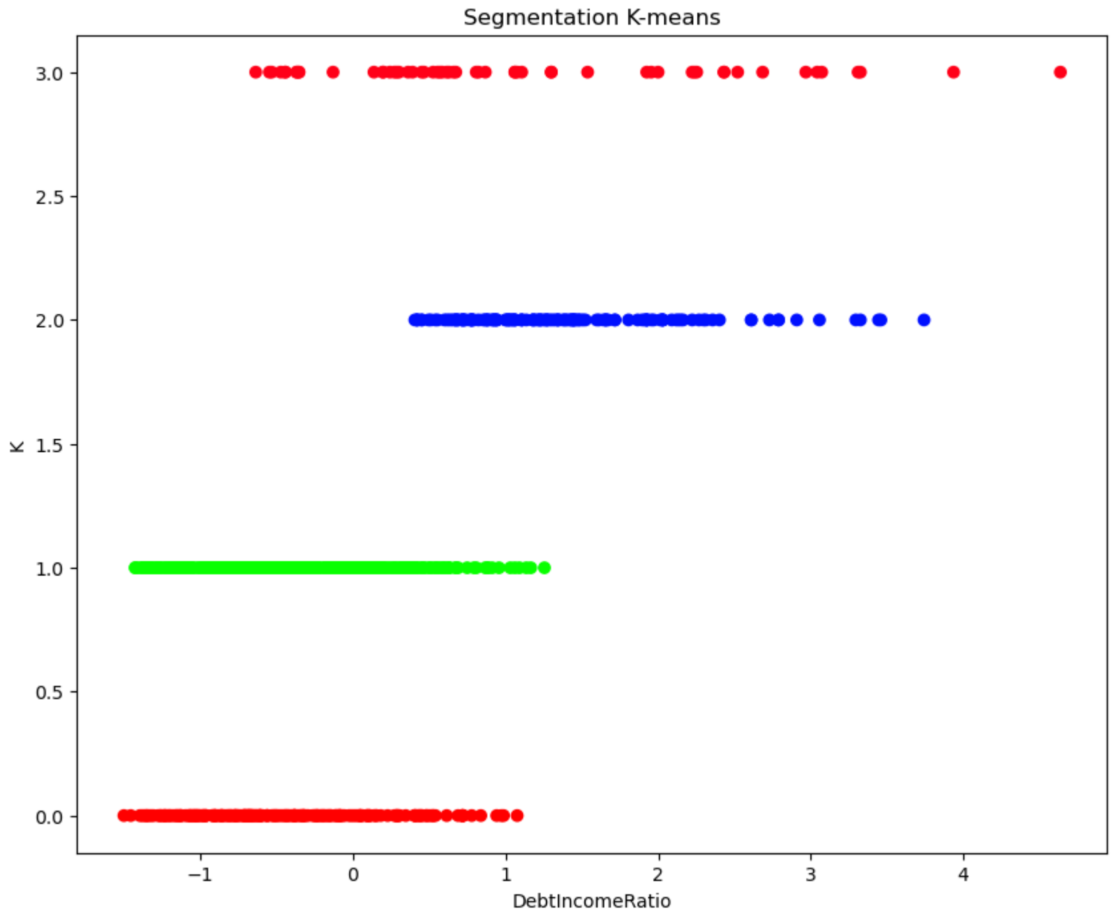


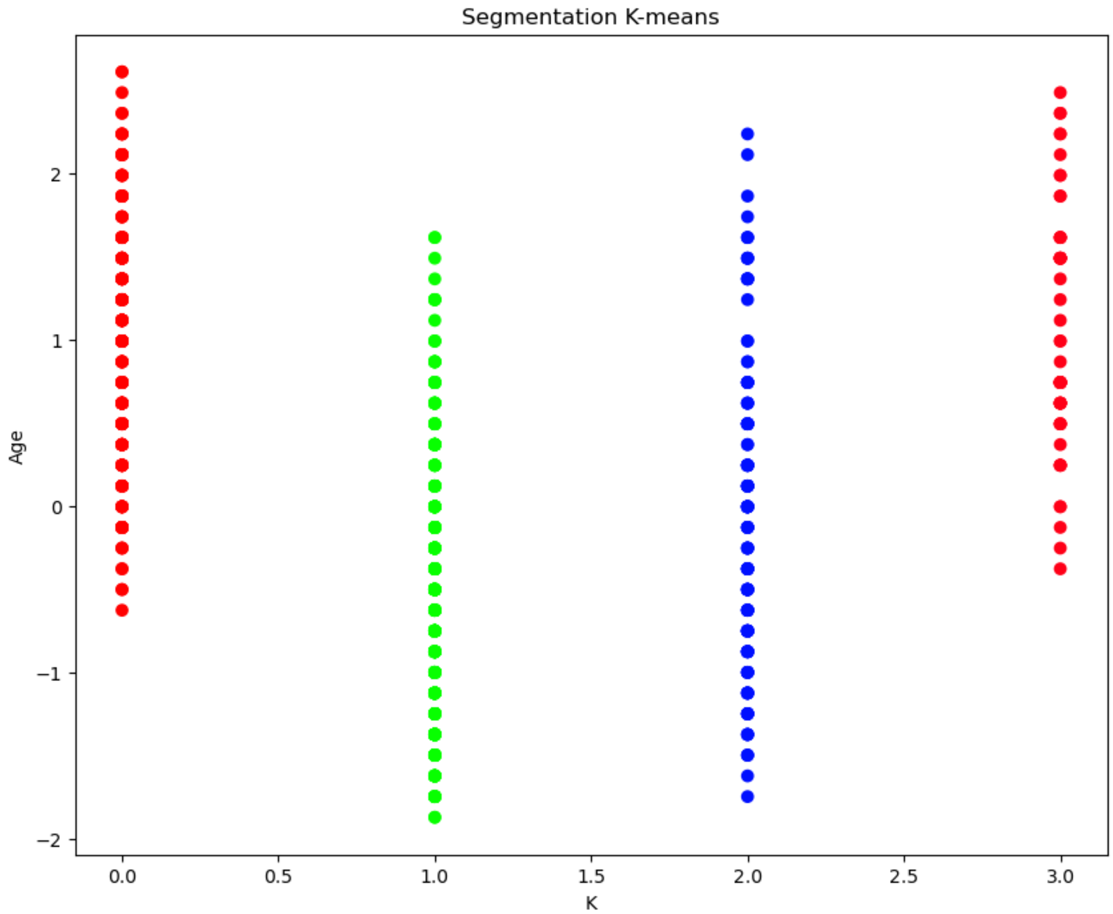
Segmentation K-means
Income
DebtIncomeRatio
10
8
6
4
2
0
−1
0
1
2
3
4

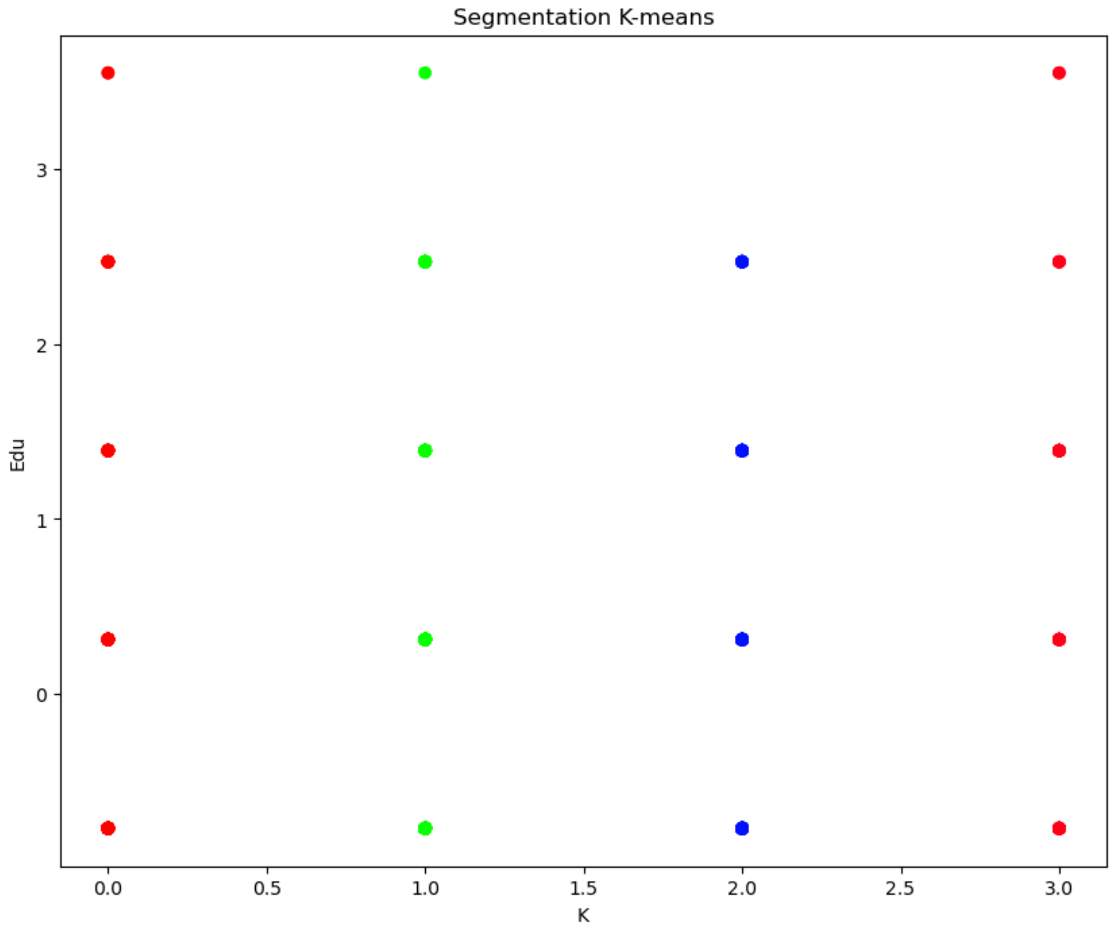


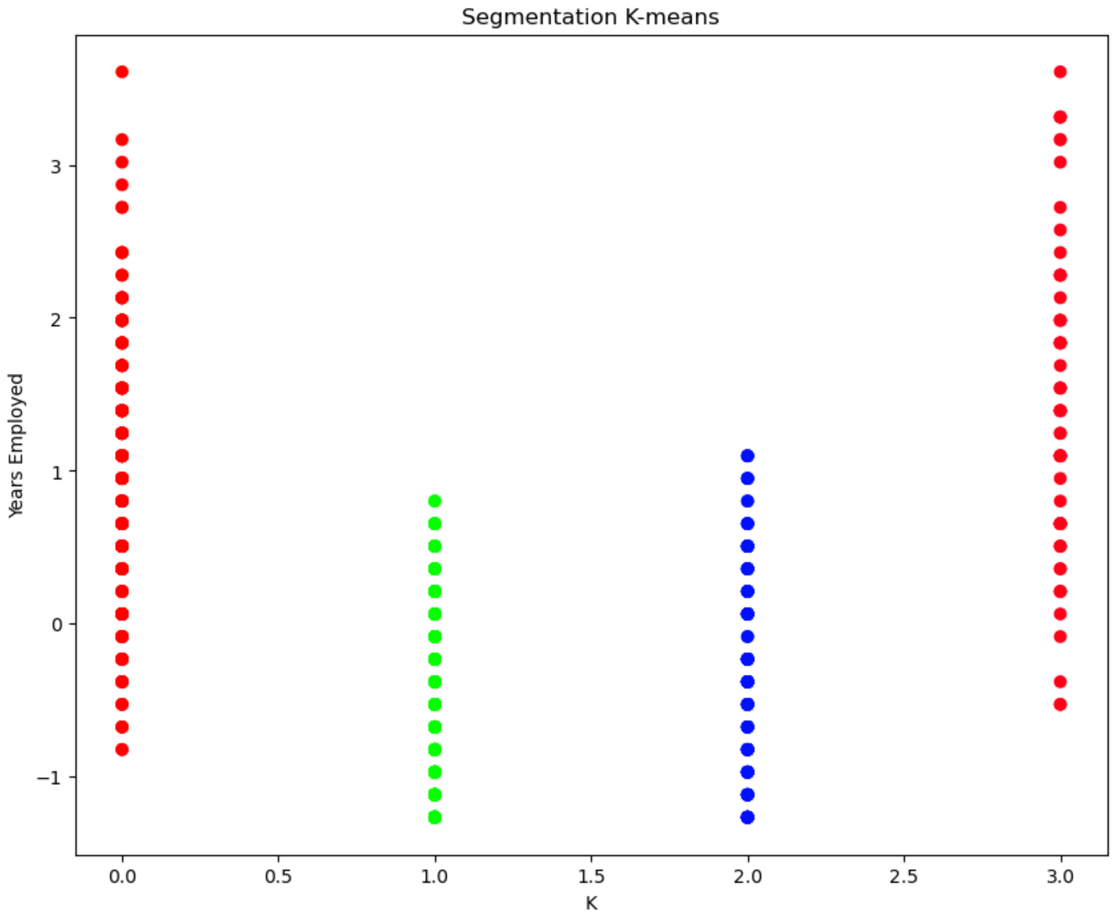
Segmentation K-means
Card Debt
DebtIncomeRatio
8
6
4
2
0
−1
0
1
2
3
4

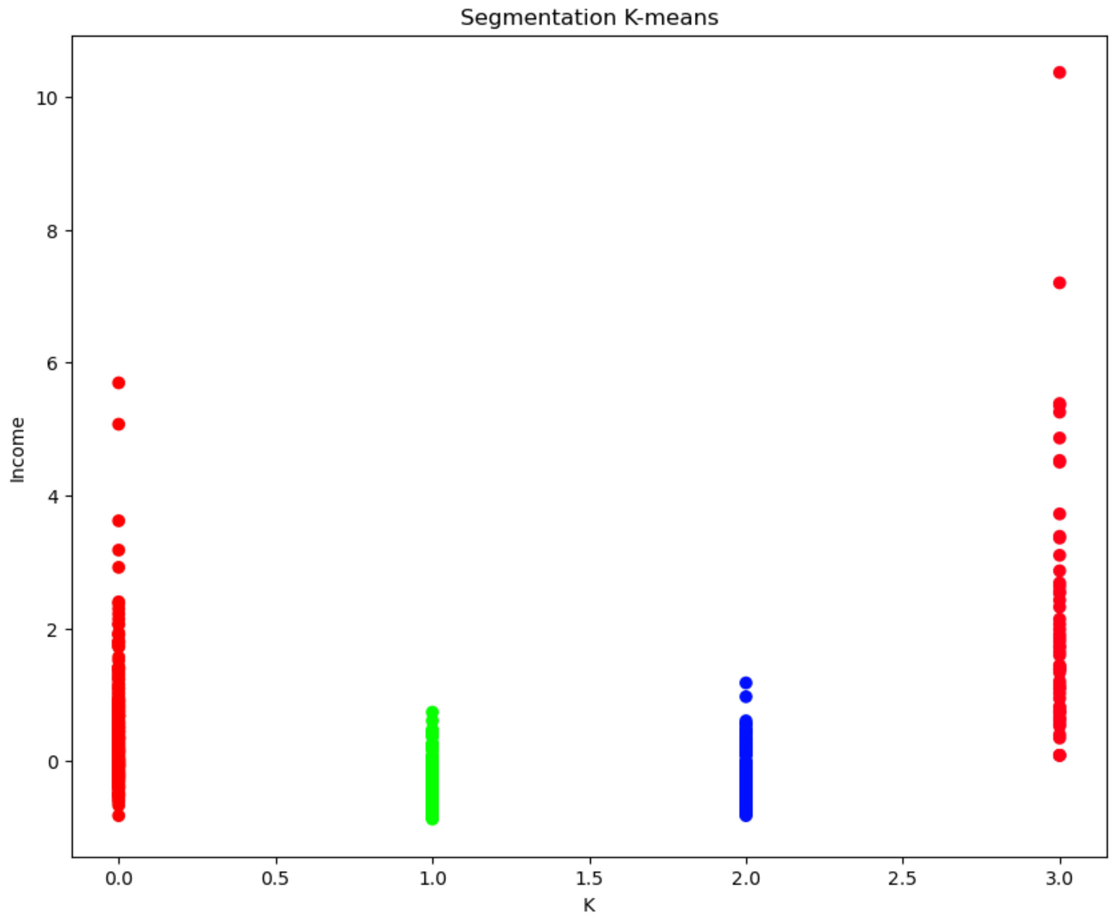

Segmentation K-means
Other Debt
DebtIncomeRatio
8
6
4
2
0
−1
0
1
2
3
4

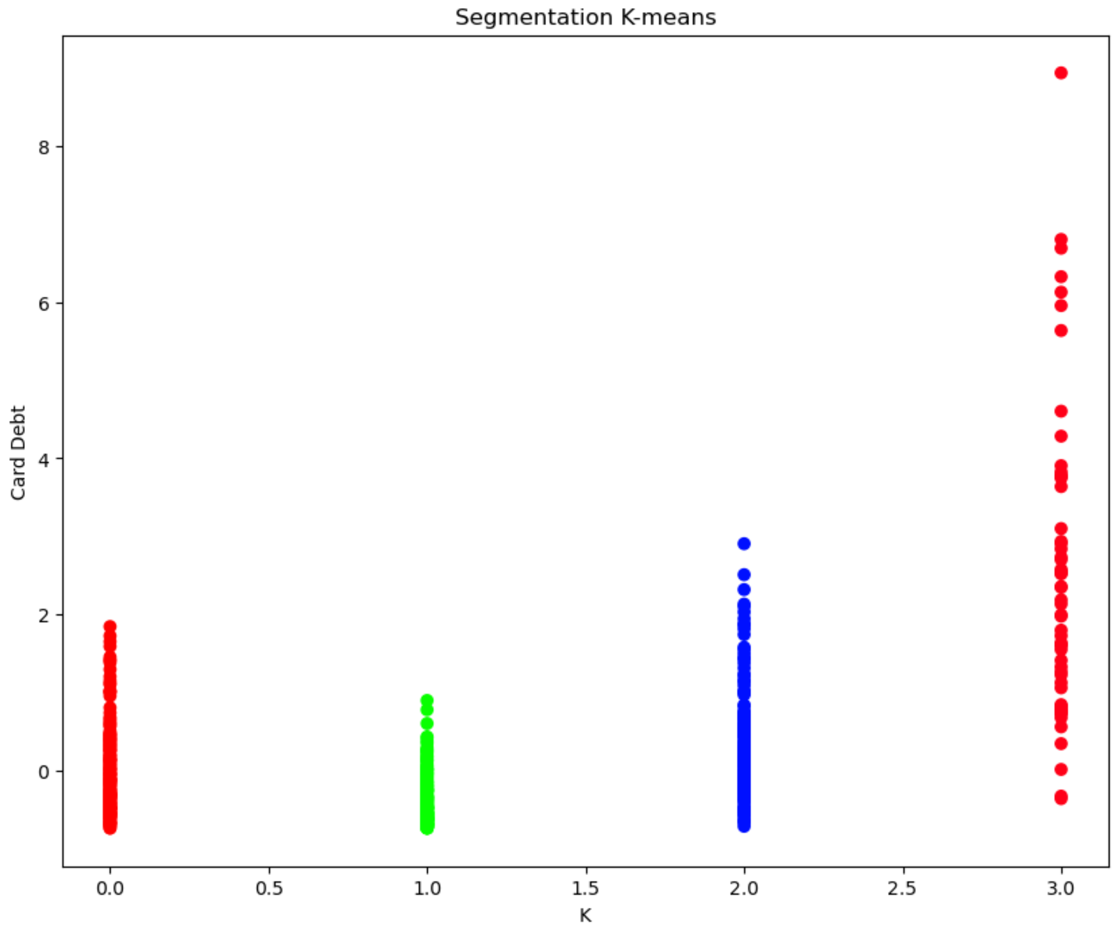

Segmentation K-means
3.0
2.5
2.0
1.5
1.0
0.5
0.0
K
−1
0
1
2
3
4
DebtIncomeRatio

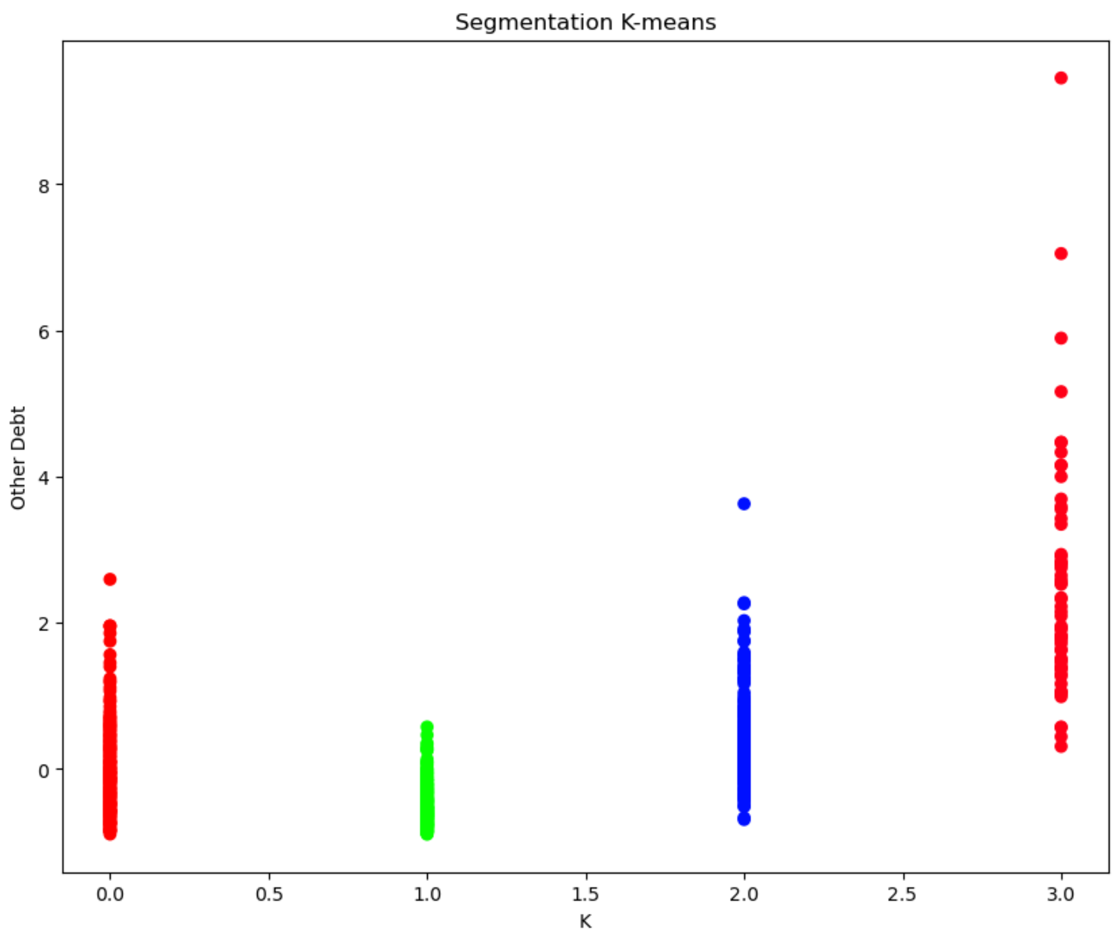

Segmentation K-means
Age
K
2
1
0
−1
−2
0.0
0.5
1.0
1.5
2.0
2.5
3.0

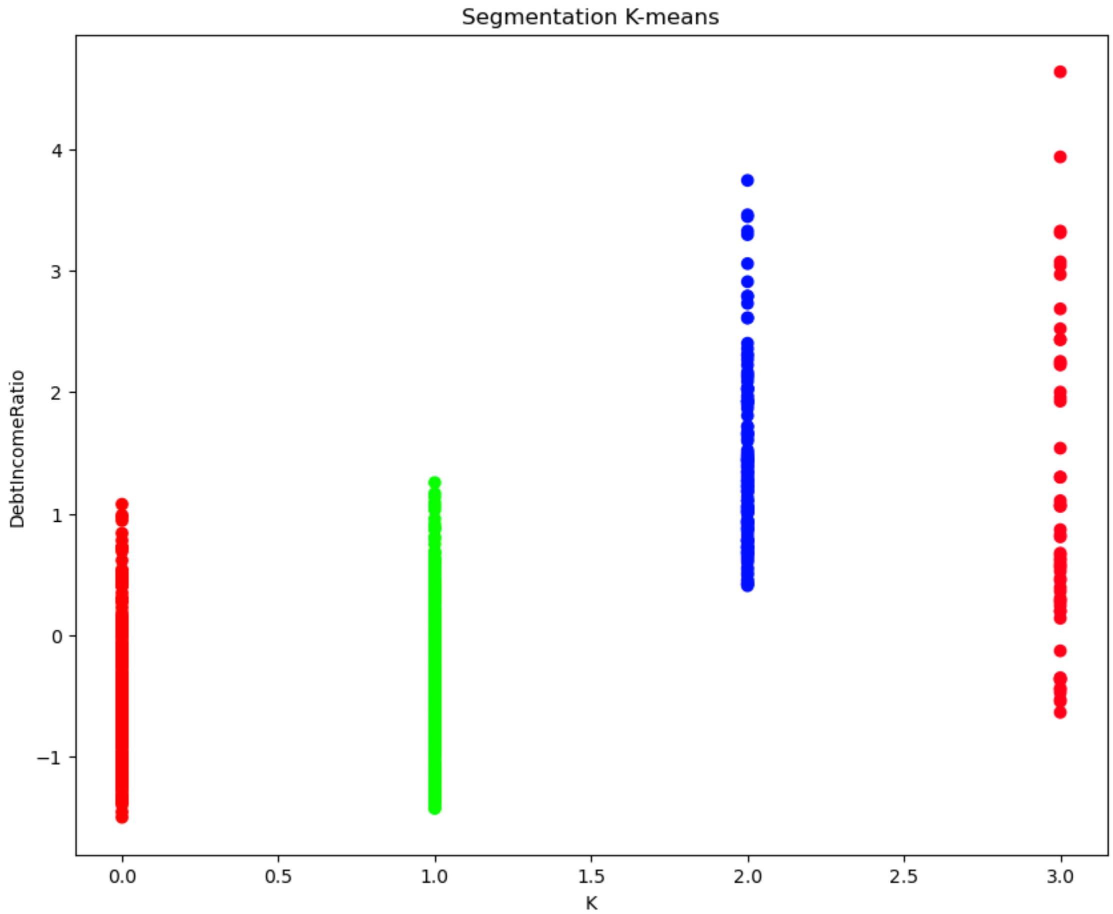

Segmentation K-means
Edu
K
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0
1
2
3


Segmentation K-means
Years Employed
K
3
2
1
0
−1
0.0
0.5
1.0
1.5
2.0
2.5
3.0


Segmentation K-means
Income
K
10
8
6
4
2
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0


Segmentation K-means
Card Debt
K
8
6
4
2
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0


Segmentation K-means
Other Debt
K
0
2
4
6
8
0.0
0.5
1.0
1.5
2.0
2.5
3.0




# 3 Q1. When should we split the data into training and testing sets when using K-means clustering, and why?

If you are using K-means clustering as a pre-processing step for a supervised learning task, such as classification or regression, then you may need to split the data into training and testing sets. In this case, you would typically apply K-means clustering to the training data only and then use the resulting cluster assignments as features in your supervised learning model. You would then evaluate the performance of your model on the testing data to ensure that it generalizes well to new, unseen data.

# 4 Q2. Why do we need to scale the features before performing K-means clustering?

K-means clustering is a distance-based algorithm, and the distances between data points are sensitive to the scale of the features. When the features are on different scales, K-means clustering may be biased towards features with larger scales, which can lead to incorrect cluster assignments and poor clustering performance.

```
[ ]:
```