Bankind_EDA_Project

July 11, 2025

[5]:
```
pip install mysql-connector-python
```

```
Requirement already satisfied: mysql-connector-python in
c:\users\hpmag\anaconda3\lib\site-packages (9.3.0)
Note: you may need to restart the kernel to use updated packages.


[notice] A new release of pip is available: 25.0.1 -> 25.1.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

[6]:
```
import mysql.connector
import pandas as pd

# connect to server
cnx = mysql.connector.connect(
    host = "127.0.0.1",
    port = 3306,
    user ="root",
    password = "SAP@123#suj")

print("Connected successfully!")
```

```
Connected successfully!
```

[7]:
```
query = "select * from banking_case.customer"
```

[8]:
```
df = pd.read_sql(query, cnx)
```

```
C:\Users\hpmag\AppData\Local\Temp\ipykernel_10444\1600954950.py:1: UserWarning:
pandas only supports SQLAlchemy connectable (engine/connection) or database
string URI or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested.
Please consider using SQLAlchemy.
  df = pd.read_sql(query, cnx)
```

[9]:
```
cnx.close()
```

[10]:
```
print(df)
```

```
    ï»¿Client ID               Name  Age  Location ID Joined Bank  \
0        IND81288      Raymond Mills   24        34324  06-05-2019
1        IND65833      Julia Spencer   23        42205  10-12-2001
2        IND47499     Stephen Murray   27         7314  25-01-2010
3        IND72498     Virginia Garza   40        34594  28-03-2019
4        IND60181    Melissa Sanders   46        41269  20-07-2012
…             …                  …  …          …         …
2995     IND66827          Earl Hall   82         8760  09-10-2014
2996     IND40556   Billy Williamson   44        32837  05-02-2009
2997     IND72414       Victor Black   70        36088  29-12-2009
2998     IND46652        Andrew Ford   56        24871  13-02-2006
2999     IND40216         Amy Nguyen   79        38518  08-12-2005

         Banking Contact Nationality                     Occupation  \
0          Anthony Torres    American          Safety Technician IV
1        Jonathan Hawkins     African           Software Consultant
2           Anthony Berry    European           Help Desk Operator
3              Steve Diaz    American                  Geologist II
4              Shawn Long    American           Assistant Professor
…                   …          …                            …
2995       Joshua Bennett    American      Accounting Assistant III
2996          Dennis Ruiz    European                     Paralegal
2997          Joshua Ryan    American               Statistician IV
2998  Nicholas Cunningham    European  Human Resources Assistant III
2999           Joe Hanson    American            Biostatistician III

    Fee Structure Loyalty Classification  …  Bank Deposits  \
0            High                   Jade  …     1485828.64
1            High                   Jade  …      641482.79
2            High                   Gold  …     1033401.59
3             Mid                 Silver  …     1048157.49
4             Mid               Platinum  …      487782.53
…           …                      …  …            …
2995         High                   Gold  …     1089957.03
2996          Mid                   Gold  …      136891.32
2997          Low                   Jade  …      214860.89
2998          Mid                   Jade  …      742630.22
2999         High                   Jade  …       65617.66

     Checking Accounts  Saving Accounts  Foreign Currency Account  \
0            603617.88        607332.46                  12249.96
1            229521.37        344635.16                  61162.31
2            652674.69        203054.35                  79071.78
3           1048157.49        234685.02                  57513.65
4            446644.25        128351.45                  30012.14
…                  …              …                       …
2995         532867.88        657849.62                  12947.31
2996          56581.74         93195.61                  23205.69
```

```
2997          158726.06         35539.15                     30291.81
2998          404638.26         56411.33                      6413.14
2999           77769.08         32371.38                      8992.36

      Business Lending  Properties Owned  Risk Weighting  BRId  GenderId  IAId
0           1134475.30                 1               2     1         1     1
1           2000526.10                 1               3     2         1     2
2            548137.58                 1               3     3         2     3
3           1148402.29                 0               4     4         1     4
4           1674412.12                 0               3     1         2     5
…                  …               …             …   …       …   …
2995        1238859.91                 1               3     3         2     4
2996         277171.07                 1               2     3         2     5
2997         502947.22                 2               2     3         2     6
2998        1538368.60                 3               1     3         2     7
2999         329412.55                 1               1     3         2     8

[3000 rows x 25 columns]
```

[11]:
```
df.head(5)
```

[11]:
```
  ï»¿Client ID             Name  Age  Location ID Joined Bank  \
0     IND81288    Raymond Mills   24        34324  06-05-2019
1     IND65833    Julia Spencer   23        42205  10-12-2001
2     IND47499   Stephen Murray   27         7314  25-01-2010
3     IND72498   Virginia Garza   40        34594  28-03-2019
4     IND60181  Melissa Sanders   46        41269  20-07-2012

     Banking Contact Nationality            Occupation Fee Structure  \
0     Anthony Torres    American  Safety Technician IV          High
1   Jonathan Hawkins     African   Software Consultant          High
2      Anthony Berry    European    Help Desk Operator          High
3         Steve Diaz    American          Geologist II           Mid
4         Shawn Long    American   Assistant Professor           Mid

  Loyalty Classification  …  Bank Deposits  Checking Accounts  \
0                   Jade  …     1485828.64          603617.88
1                   Jade  …      641482.79          229521.37
2                   Gold  …     1033401.59          652674.69
3                 Silver  …     1048157.49         1048157.49
4               Platinum  …      487782.53          446644.25

   Saving Accounts  Foreign Currency Account  Business Lending  \
0        607332.46                  12249.96        1134475.30
1        344635.16                  61162.31        2000526.10
2        203054.35                  79071.78         548137.58
3        234685.02                  57513.65        1148402.29
```

```
4        128351.45                  30012.14        1674412.12

   Properties Owned  Risk Weighting  BRId  GenderId  IAId
0                 1               2     1         1     1
1                 1               3     2         1     2
2                 1               3     3         2     3
3                 0               4     4         1     4
4                 0               3     1         2     5

[5 rows x 25 columns]
```

[12]:
```
df.describe()
```

[12]:
```
                Age   Location ID  Estimated Income  Superannuation Savings  \
count  3000.000000   3000.000000       3000.000000             3000.000000
mean     51.039667  21563.323000     171305.034263            25531.599673
std      19.854760  12462.273017     111935.808209            16259.950770
min      17.000000     12.000000      15919.480000             1482.030000
25%      34.000000  10803.500000      82906.595000            12513.775000
50%      51.000000  21129.500000     142313.480000            22357.355000
75%      69.000000  32054.500000     242290.305000            35464.740000
max      85.000000  43369.000000     522330.260000            75963.900000

       Amount of Credit Cards  Credit Card Balance    Bank Loans  \
count             3000.000000          3000.000000  3.000000e+03
mean                 1.463667          3176.206943  5.913862e+05
std                  0.676387          2497.094709  4.575570e+05
min                  1.000000             1.170000  0.000000e+00
25%                  1.000000          1236.630000  2.396281e+05
50%                  1.000000          2560.805000  4.797934e+05
75%                  2.000000          4522.632500  8.258130e+05
max                  3.000000         13991.990000  2.667557e+06

       Bank Deposits  Checking Accounts  Saving Accounts  \
count   3.000000e+03       3.000000e+03     3.000000e+03
mean    6.715602e+05       3.210929e+05     2.329084e+05
std     6.457169e+05       2.820796e+05     2.300078e+05
min     0.000000e+00       0.000000e+00     0.000000e+00
25%     2.044004e+05       1.199475e+05     7.479440e+04
50%     4.633165e+05       2.428157e+05     1.640866e+05
75%     9.427546e+05       4.348749e+05     3.155750e+05
max     3.890598e+06       1.969923e+06     1.724118e+06

       Foreign Currency Account  Business Lending  Properties Owned  \
count               3000.000000      3.000000e+03       3000.000000
mean               29883.529993      8.667598e+05          1.518667
std                23109.924010      6.412303e+05          1.102145
```

```
min                45.000000     0.000000e+00          0.000000
25%             11916.542500     3.748251e+05          1.000000
50%             24341.190000     7.113147e+05          2.000000
75%             41966.392500     1.185110e+06          2.000000
max            124704.870000     3.825962e+06          3.000000

       Risk Weighting         BRId     GenderId          IAId
count     3000.000000  3000.000000  3000.000000   3000.000000
mean         2.249333     2.559333     1.504000     10.425333
std          1.131191     1.007713     0.500067      5.988242
min          1.000000     1.000000     1.000000      1.000000
25%          1.000000     2.000000     1.000000      5.000000
50%          2.000000     3.000000     2.000000     10.000000
75%          3.000000     3.000000     2.000000     15.000000
max          5.000000     4.000000     2.000000     22.000000
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3000 entries, 0 to 2999
Data columns (total 25 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   ï»¿Client ID              3000 non-null   object
 1   Name                      3000 non-null   object
 2   Age                       3000 non-null   int64
 3   Location ID               3000 non-null   int64
 4   Joined Bank               3000 non-null   object
 5   Banking Contact           3000 non-null   object
 6   Nationality               3000 non-null   object
 7   Occupation                3000 non-null   object
 8   Fee Structure             3000 non-null   object
 9   Loyalty Classification    3000 non-null   object
 10  Estimated Income          3000 non-null   float64
 11  Superannuation Savings    3000 non-null   float64
 12  Amount of Credit Cards    3000 non-null   int64
 13  Credit Card Balance       3000 non-null   float64
 14  Bank Loans                3000 non-null   float64
 15  Bank Deposits             3000 non-null   float64
 16  Checking Accounts         3000 non-null   float64
 17  Saving Accounts           3000 non-null   float64
 18  Foreign Currency Account  3000 non-null   float64
 19  Business Lending          3000 non-null   float64
 20  Properties Owned          3000 non-null   int64
 21  Risk Weighting            3000 non-null   int64
 22  BRId                      3000 non-null   int64
 23  GenderId                  3000 non-null   int64
 24  IAId                      3000 non-null   int64
```

```
dtypes: float64(9), int64(8), object(8)
memory usage: 586.1+ KB
```

[14]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

[15]:
```python
df.shape
```

[15]:
```
(3000, 25)
```

[16]:
```python
df['Estimated Income'].value_counts()
```

[16]:
```
Estimated Income
75384.77     1
341878.19    1
325675.36    1
144361.23    1
127999.76    1
            ..
96970.50     1
267003.90    1
126128.01    1
41843.49     1
56826.53     1
Name: count, Length: 3000, dtype: int64
```

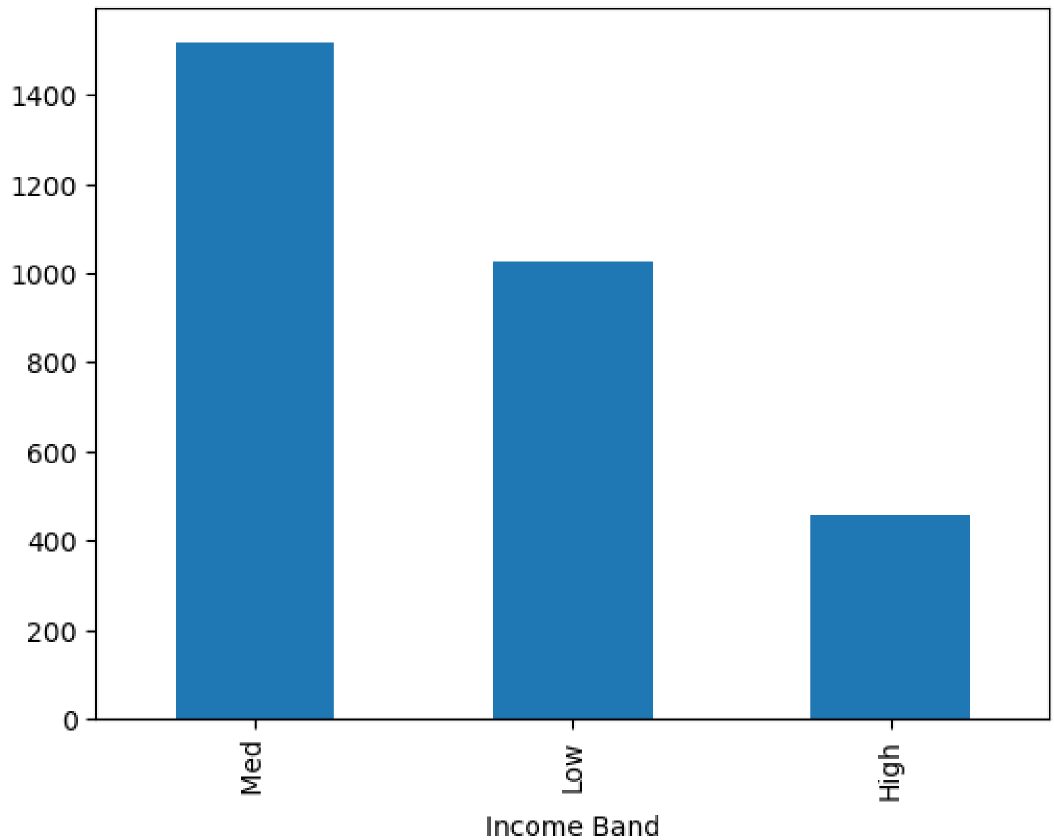
[17]:
```python
df['Estimated Income'].min()
```

[17]:
```
15919.48
```

[18]:
```python
bins =[0,100000,300000,float('inf')]
labels = ['Low',' Med', 'High']

df['Income Band'] =  pd.cut(df['Estimated Income'],bins = bins, labels =␣
↪labels, right= False)
```

[19]:
```python
df['Income Band'].value_counts().plot(kind = 'bar')
```

[19]:
```
<Axes: xlabel='Income Band'>
```



[20]:
```
df.columns
```

[20]:
```
Index(['ï»¿Client ID', 'Name', 'Age', 'Location ID', 'Joined Bank',
       'Banking Contact', 'Nationality', 'Occupation', 'Fee Structure',
       'Loyalty Classification', 'Estimated Income', 'Superannuation Savings',
       'Amount of Credit Cards', 'Credit Card Balance', 'Bank Loans',
       'Bank Deposits', 'Checking Accounts', 'Saving Accounts',
       'Foreign Currency Account', 'Business Lending', 'Properties Owned',
       'Risk Weighting', 'BRId', 'GenderId', 'IAId', 'Income Band'],
      dtype='object')
```

[61]:
```
# Examine the distribution of unique categories in categorical columns
categorical_cols = df[['BRId','GenderId','IAId','Amount of Credit␣
↪Cards','Nationality', 'Occupation', 'Fee Structure',
       'Loyalty Classification','Properties Owned',
       'Risk Weighting','Income Band']].columns

for col in categorical_cols:
    print(f"value counts for '{col}':")
    display(df[col].value_counts())
```

```python
for i, predictor in enumerate(df[['BRId','GenderId','IAId','Amount of Credit␣
↪Cards','Nationality', 'Occupation', 'Fee Structure',
       'Loyalty Classification','Properties Owned',
       'Risk Weighting','Income Band']].columns):
        plt.figure(i)
        sns.countplot(data =df, x = predictor, hue = 'GenderId' )
```

```
value counts for 'BRId':

BRId
3    1352
1     660
2     495
4     493
Name: count, dtype: int64

value counts for 'GenderId':

GenderId
2    1512
1    1488
Name: count, dtype: int64

value counts for 'IAId':

IAId
1     177
3     177
4     177
8     177
2     177
11    176
15    176
14    176
13    176
12    176
10    176
9     176
7      89
6      89
5      89
16     88
17     88
18     88
19     88
20     88
21     88
22     88
Name: count, dtype: int64
```

```
value counts for 'Amount of Credit Cards':

Amount of Credit Cards
1    1922
2     765
3     313
Name: count, dtype: int64

value counts for 'Nationality':

Nationality
European      1309
Asian          754
American       507
Australian     254
African        176
Name: count, dtype: int64

value counts for 'Occupation':

Occupation
Structural Analysis Engineer    28
Associate Professor             28
Recruiter                       25
Human Resources Manager         24
Account Coordinator             24
                                ..
Office Assistant IV              8
Automation Specialist I          7
Computer Systems Analyst I       6
Developer III                    5
Senior Sales Associate           4
Name: count, Length: 195, dtype: int64

value counts for 'Fee Structure':

Fee Structure
High    1476
Mid      962
Low      562
Name: count, dtype: int64

value counts for 'Loyalty Classification':

Loyalty Classification
Jade        1331
Silver       767
Gold         585
Platinum     317
Name: count, dtype: int64

value counts for 'Properties Owned':
```

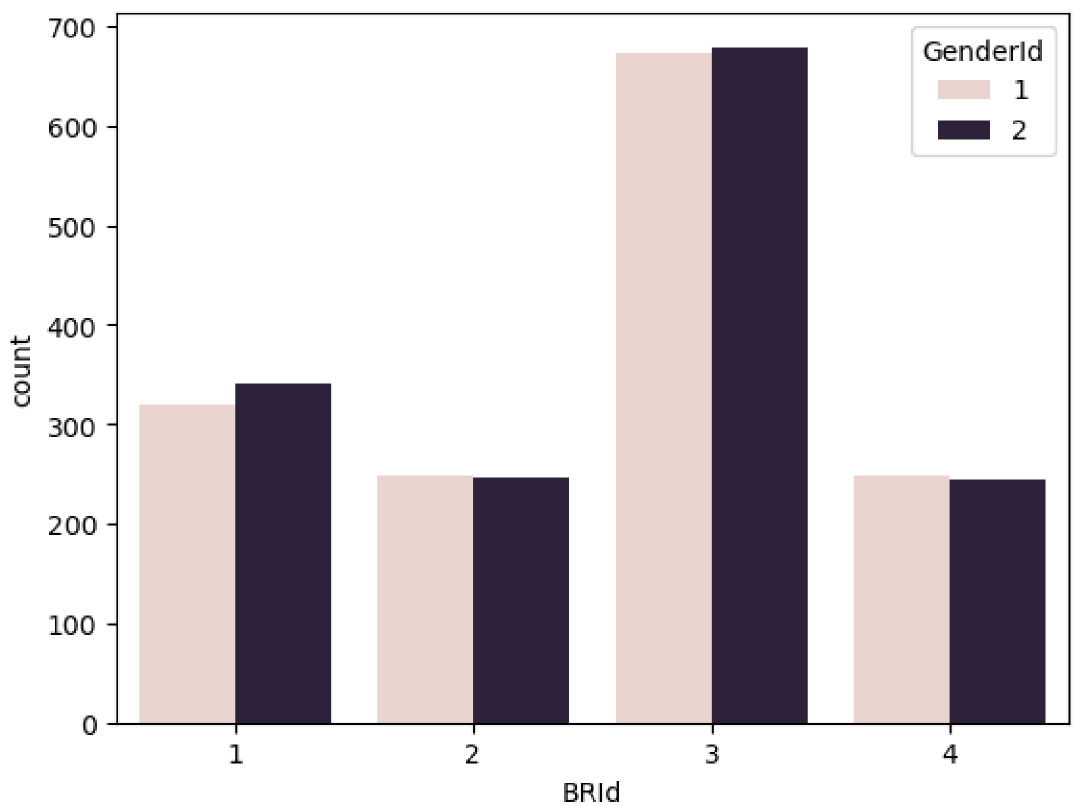
```
Properties Owned
2    777
1    776
3    742
0    705
Name: count, dtype: int64
```

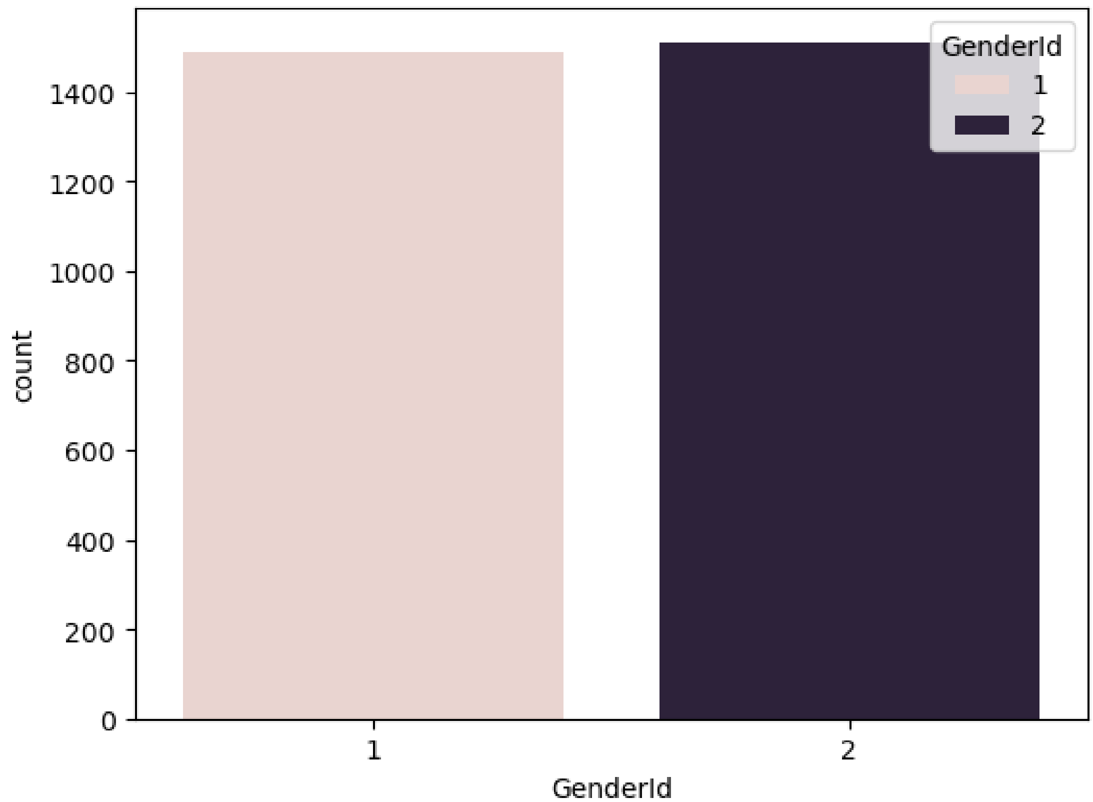
```
value counts for 'Risk Weighting':
```

```
Risk Weighting
2    1222
1     836
3     460
4     322
5     160
Name: count, dtype: int64
```

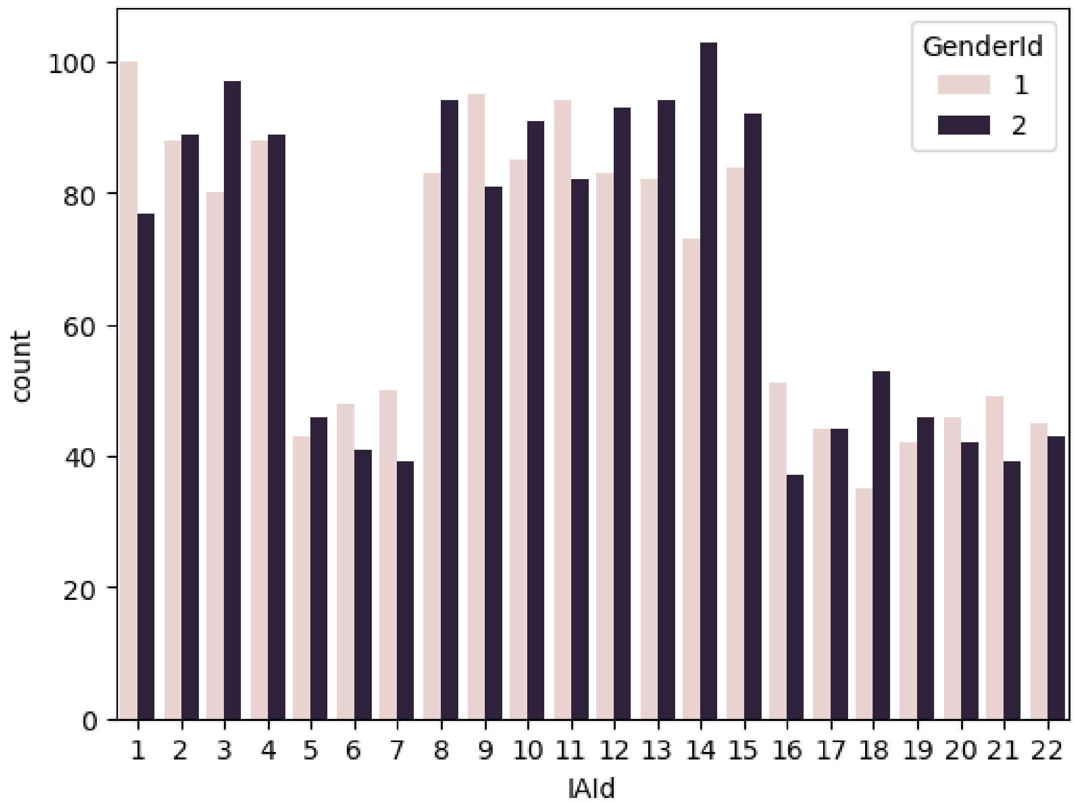
```
value counts for 'Income Band':
```

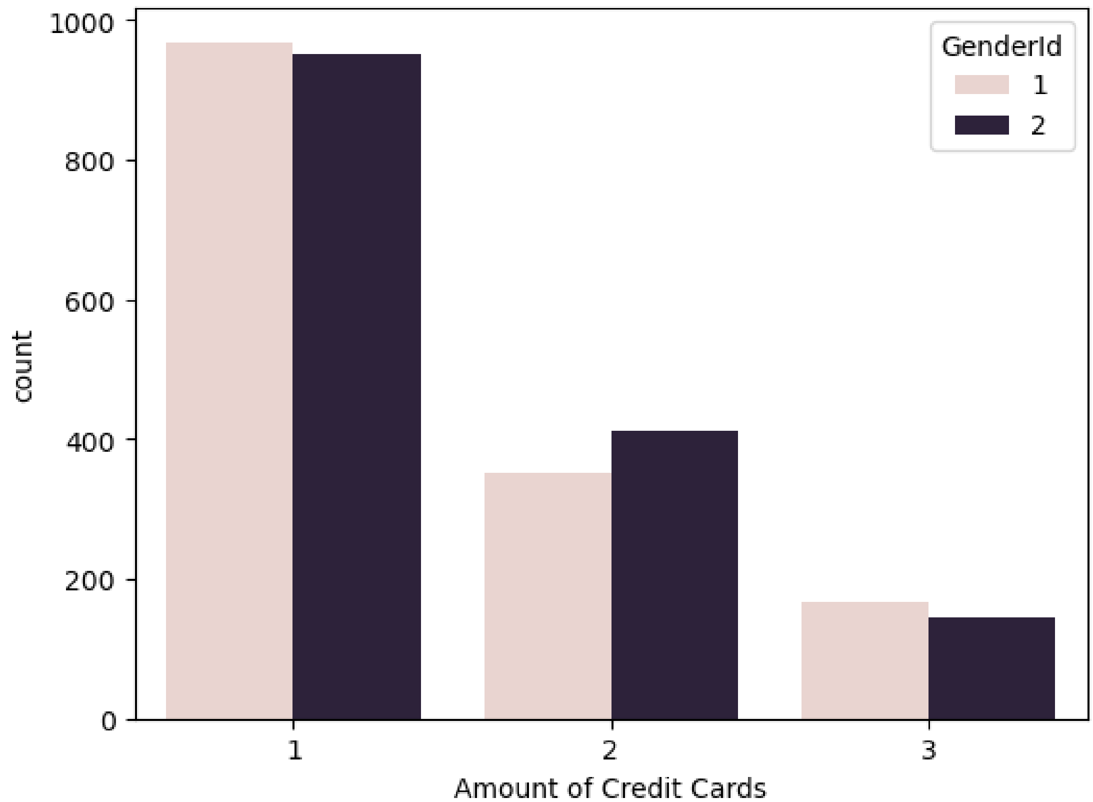
```
Income Band
 Med    1517
Low     1027
High     456
Name: count, dtype: int64
```

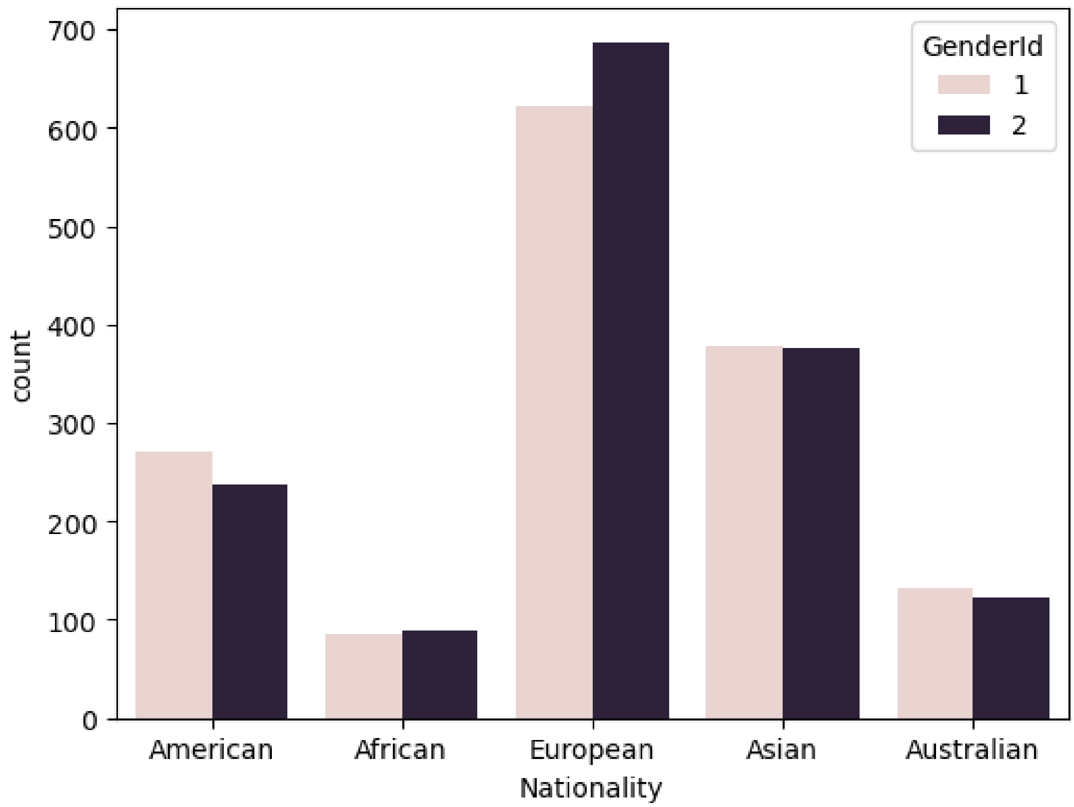


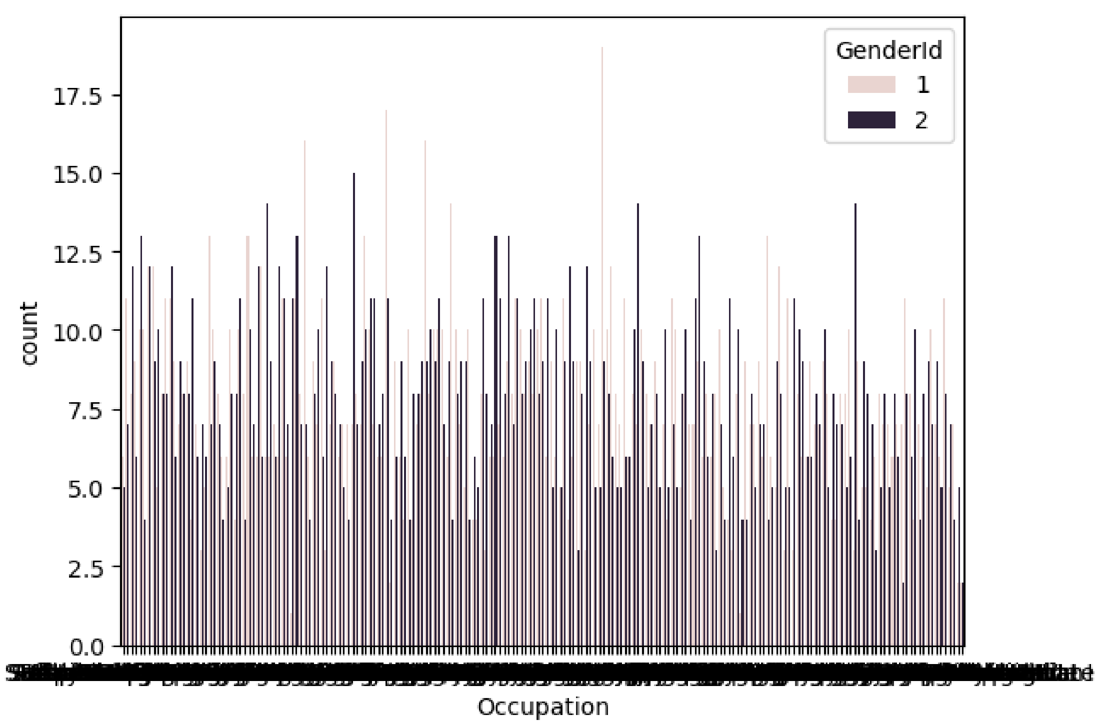


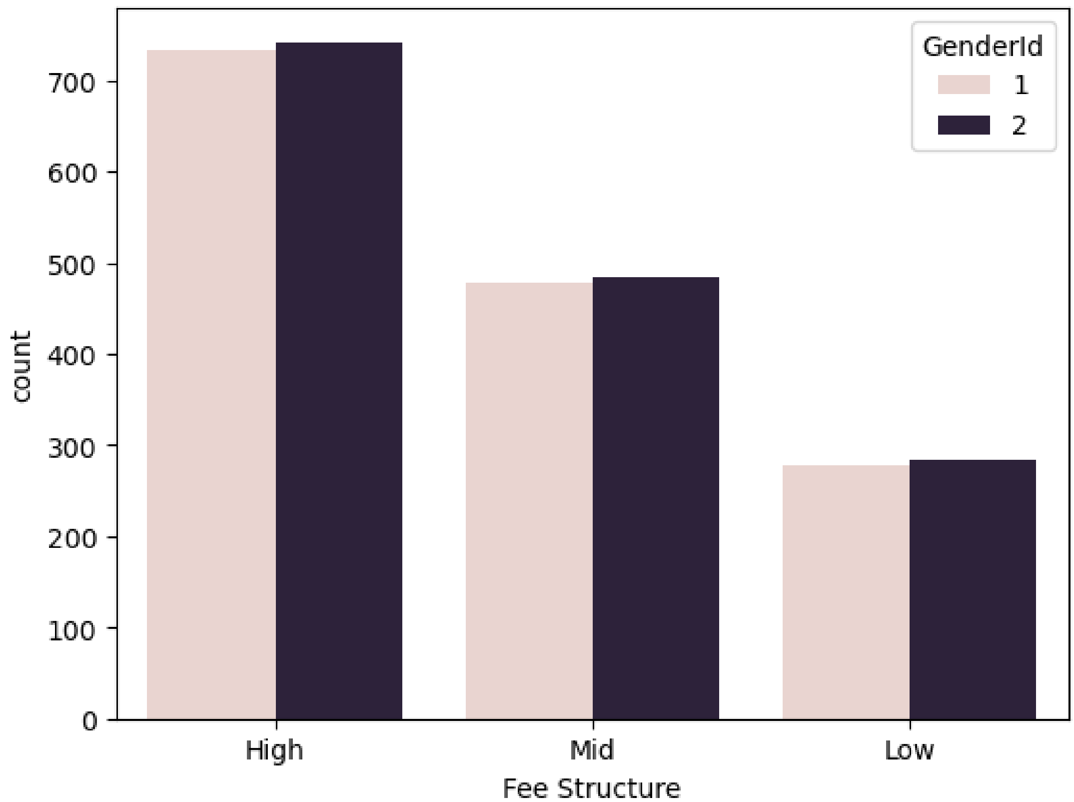
GenderId
1
2
1400
1200
1000
800
600
400
200
0
count
1
2
GenderId

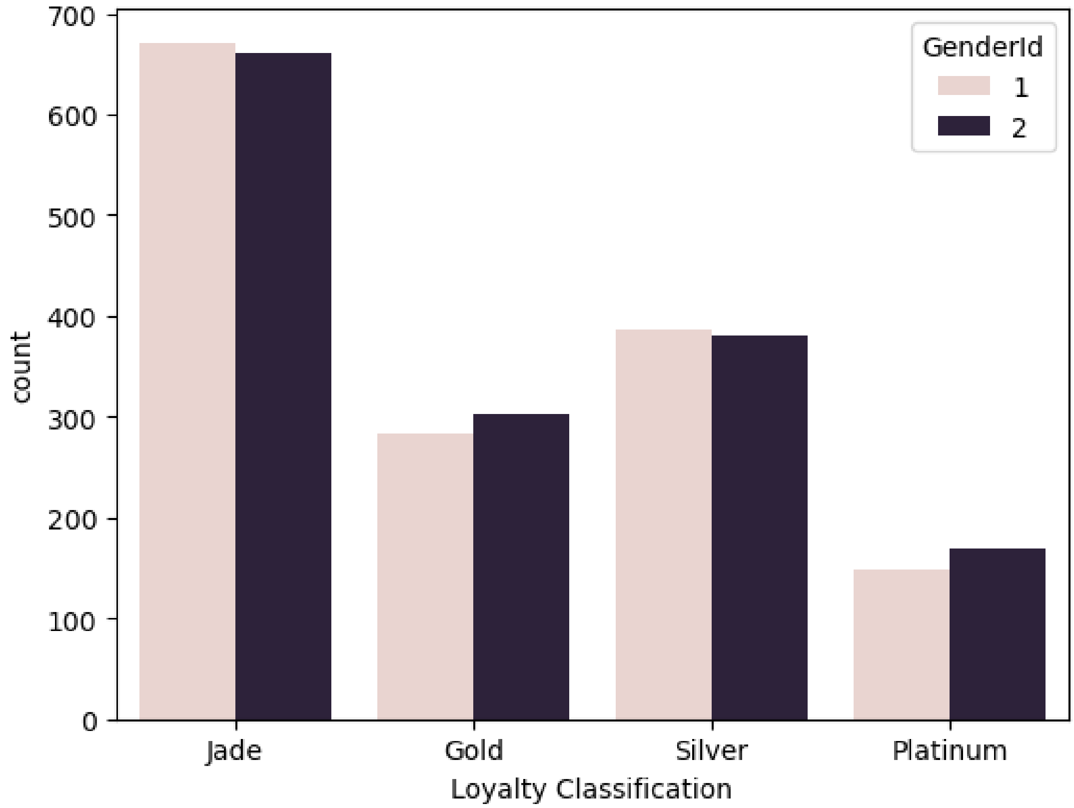

GenderId
1
2
count
IAId
0
20
40
60
80
100
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22

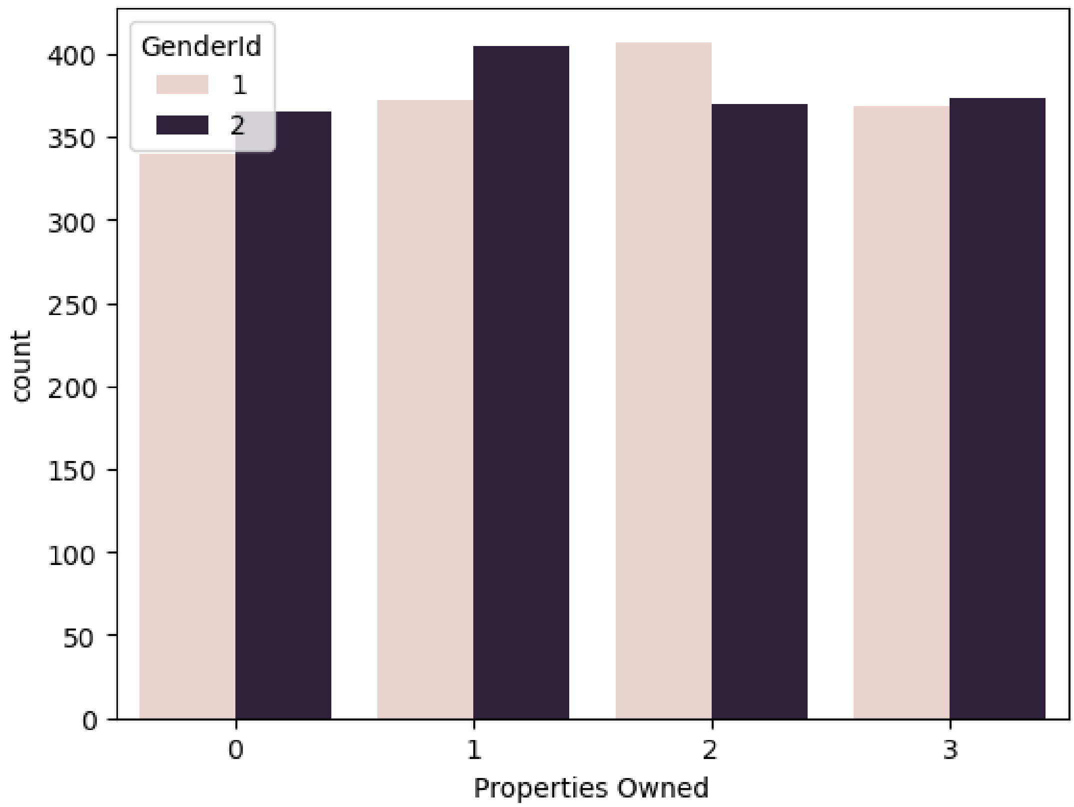

GenderId
1
2
count
1000
800
600
400
200
0
1
2
3
Amount of Credit Cards

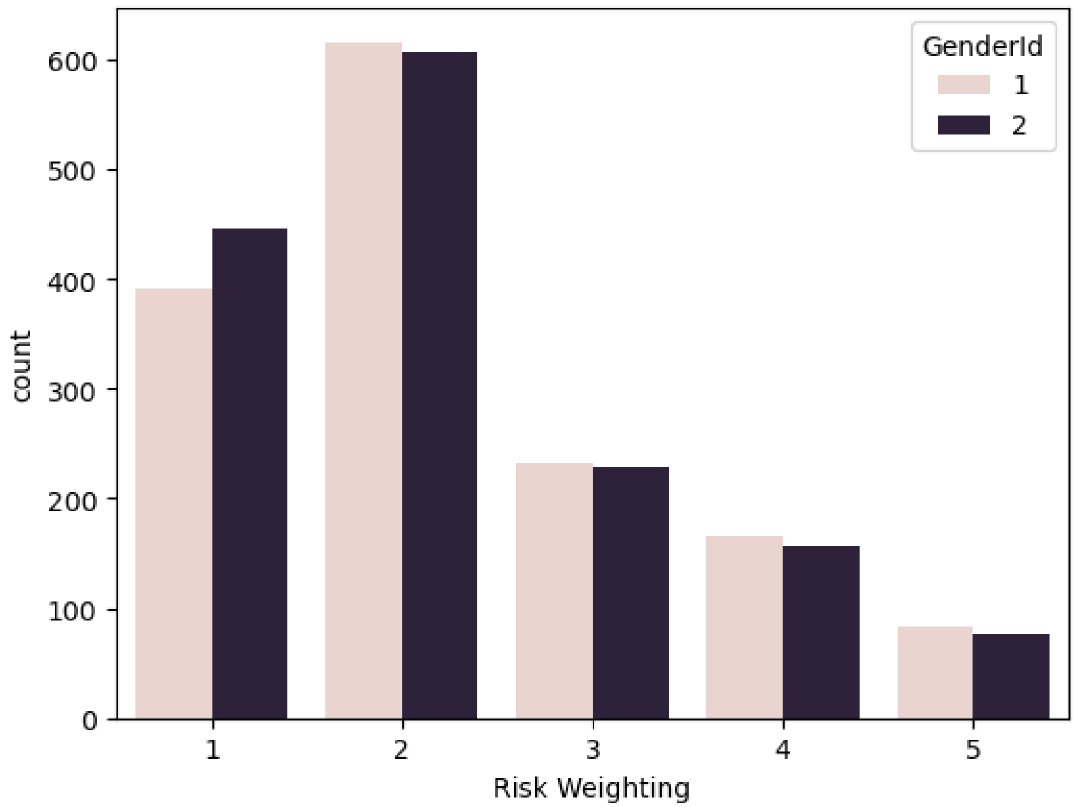

GenderId
1
2
count
700
600
500
400
300
200
100
0
American
African
European
Asian
Australian
Nationality

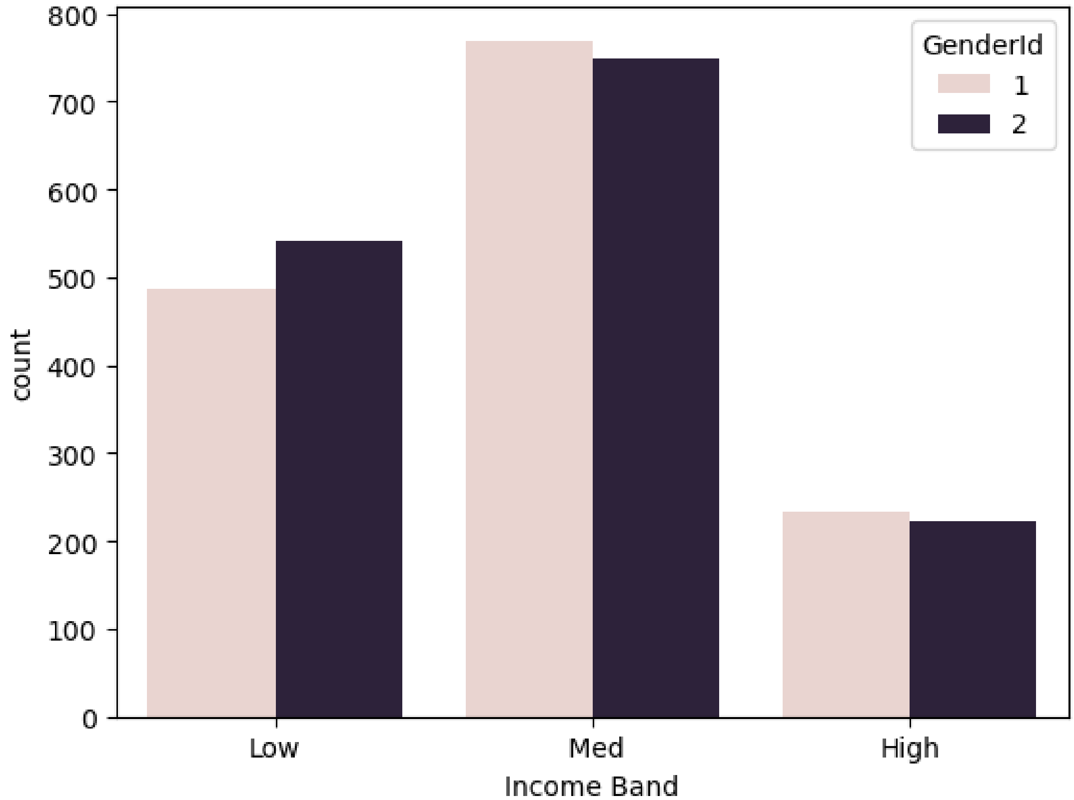

GenderId
1
2
count
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
Occupation


GenderId
1
2
700
600
500
400
300
200
100
0
count
High
Mid
Low
Fee Structure


GenderId
1
2
count
700
600
500
400
300
200
100
0
Jade
Gold
Silver
Platinum
Loyalty Classification

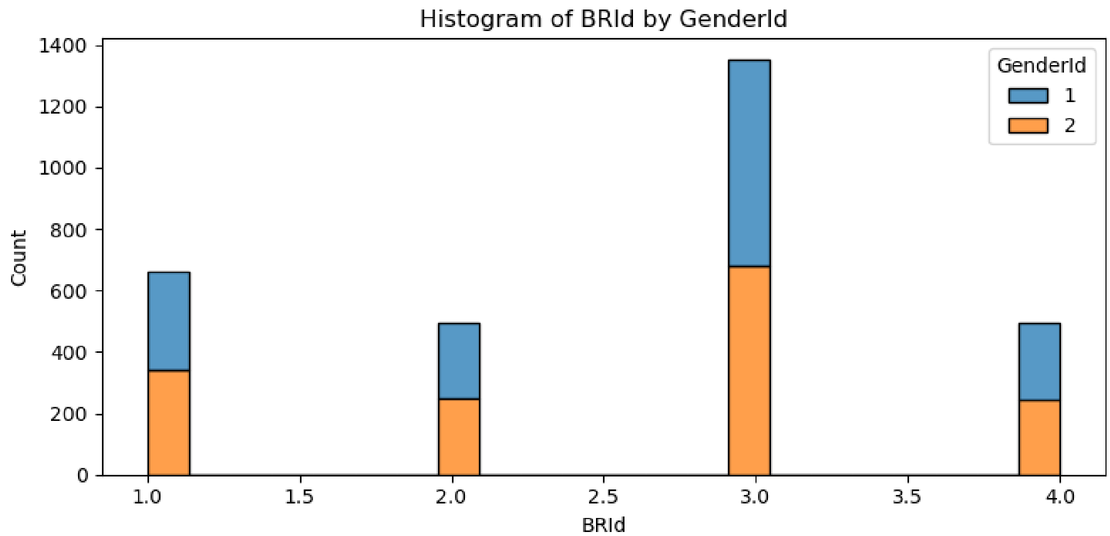

GenderId
1
2
count
400
350
300
250
200
150
100
50
0
0
1
2
3
Properties Owned


GenderId
1
2
count
600
500
400
300
200
100
0
1
2
3
4
5
Risk Weighting

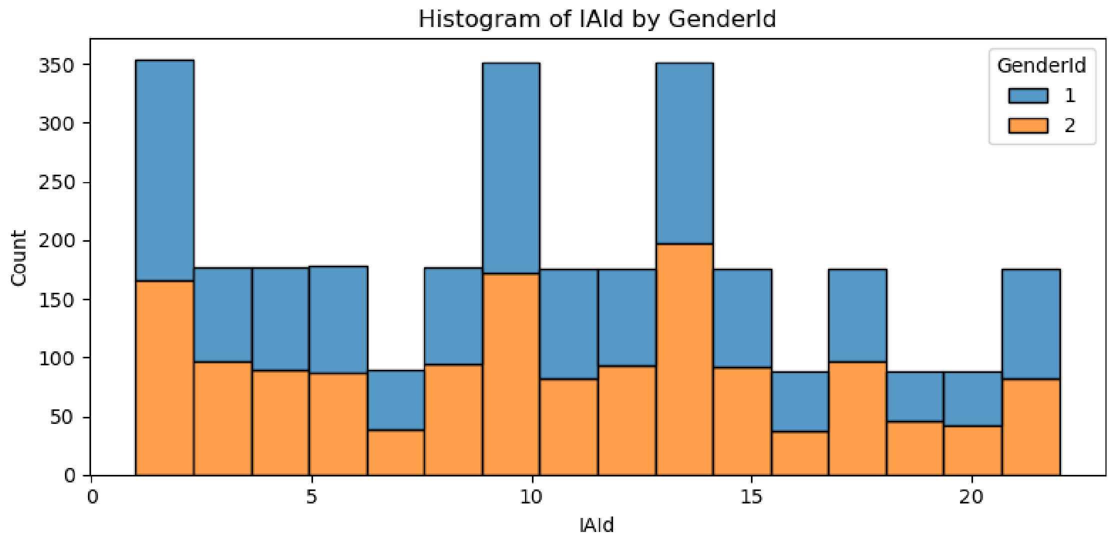


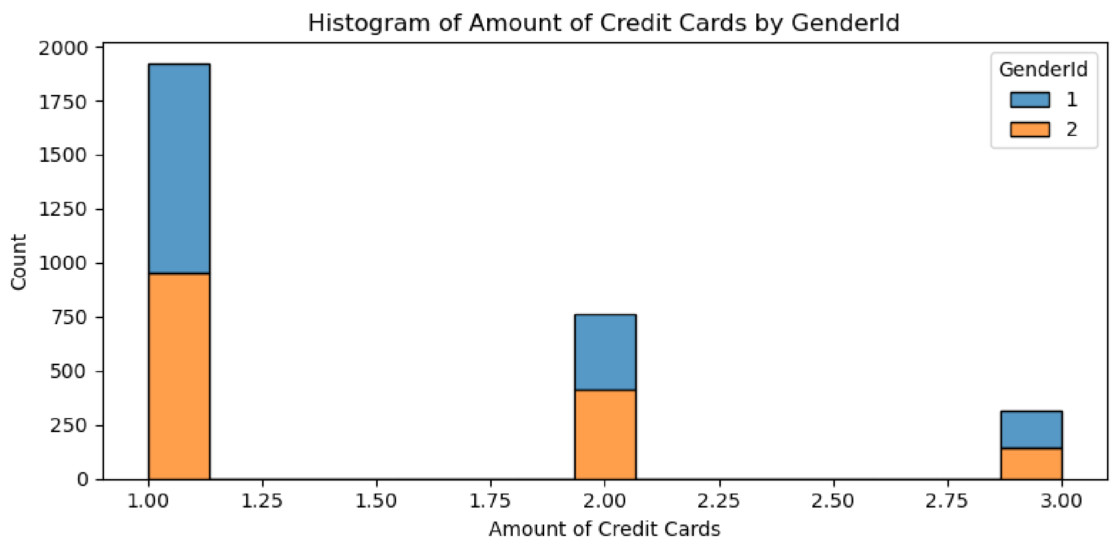
[63]:

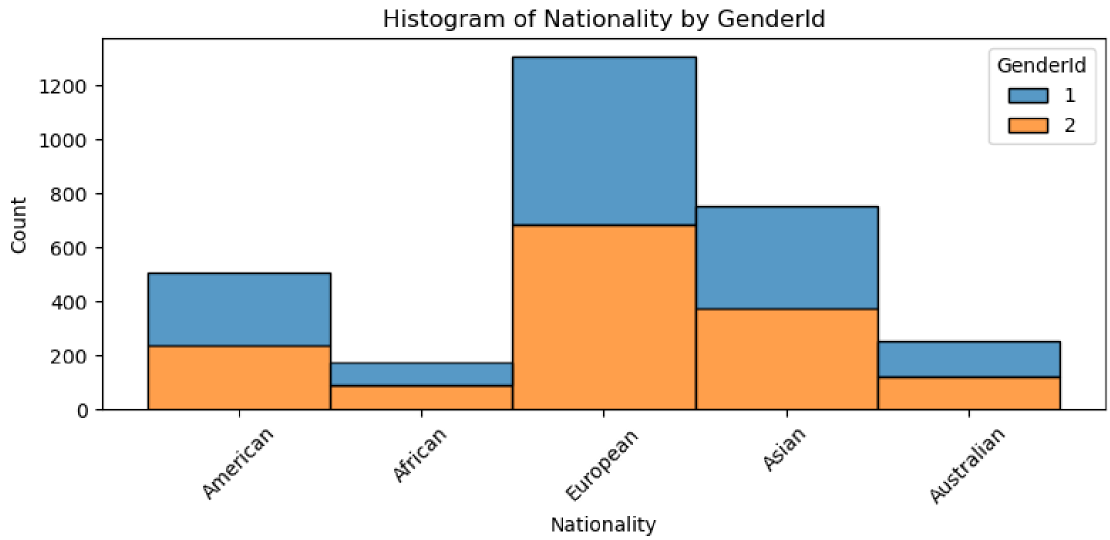
```
# Histplot of values counts for different occupatioin

for col in categorical_cols:
    if col == "Occupation":
        continue
    plt.figure(figsize=(8, 4))
    sns.histplot(data=df, x=col, hue='GenderId', multiple='stack',␣
 ↪palette='tab10', edgecolor='black')
    plt.title(f"Histogram of {col} by GenderId")
    plt.xlabel(col)
    plt.ylabel("Count")
    plt.xticks(rotation=45 if col in ['Nationality'] else 0)
    plt.tight_layout()
    plt.show()
```

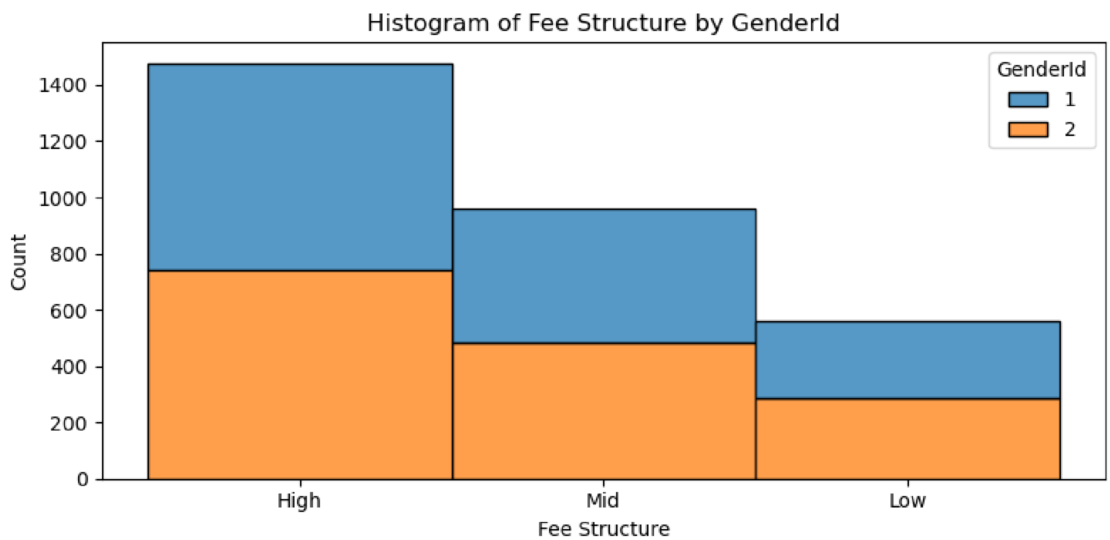

Histogram of BRId by GenderId
GenderId
1
2
Count
0
200
400
600
800
1000
1200
1400
1.0
1.5
2.0
2.5
3.0
3.5
4.0
BRId

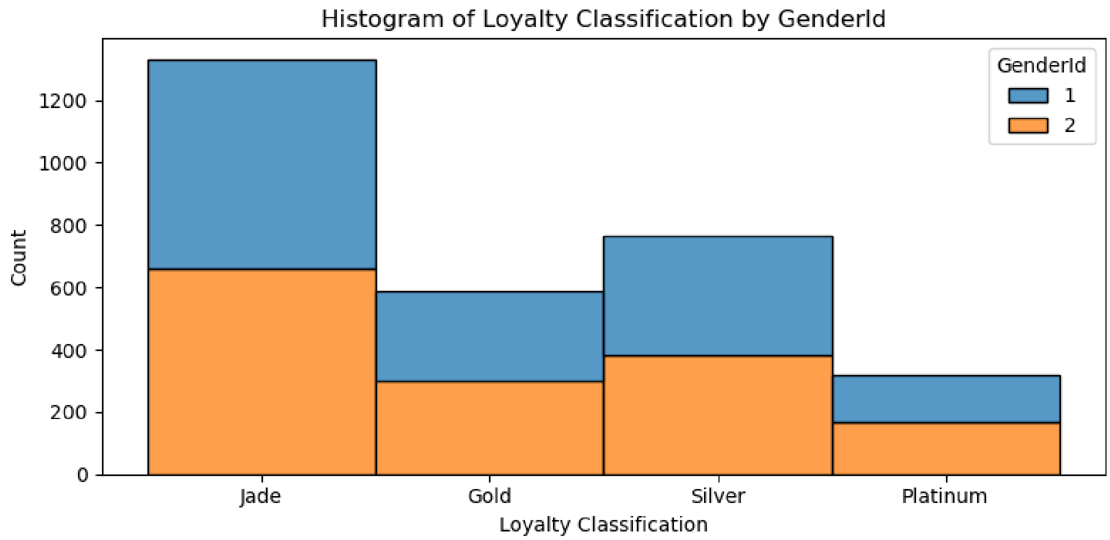

Histogram of GenderId by GenderId
GenderId
1
2
Count
0
200
400
600
800
1000
1200
1400
1.0
1.2
1.4
1.6
1.8
2.0
GenderId

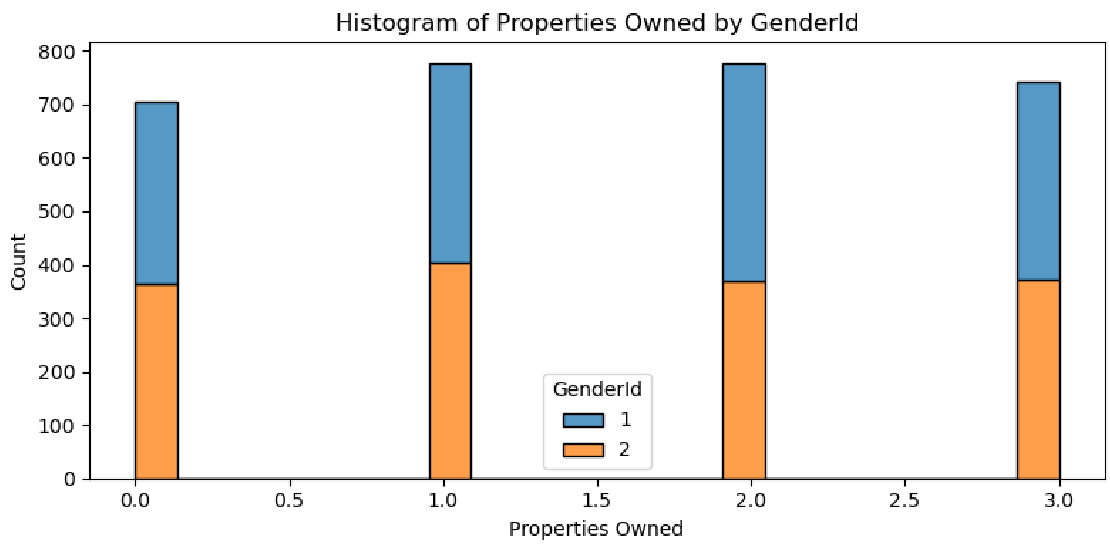

Histogram of IAId by GenderId
GenderId
1
2
Count
IAId
0
50
100
150
200
250
300
350
0
5
10
15
20

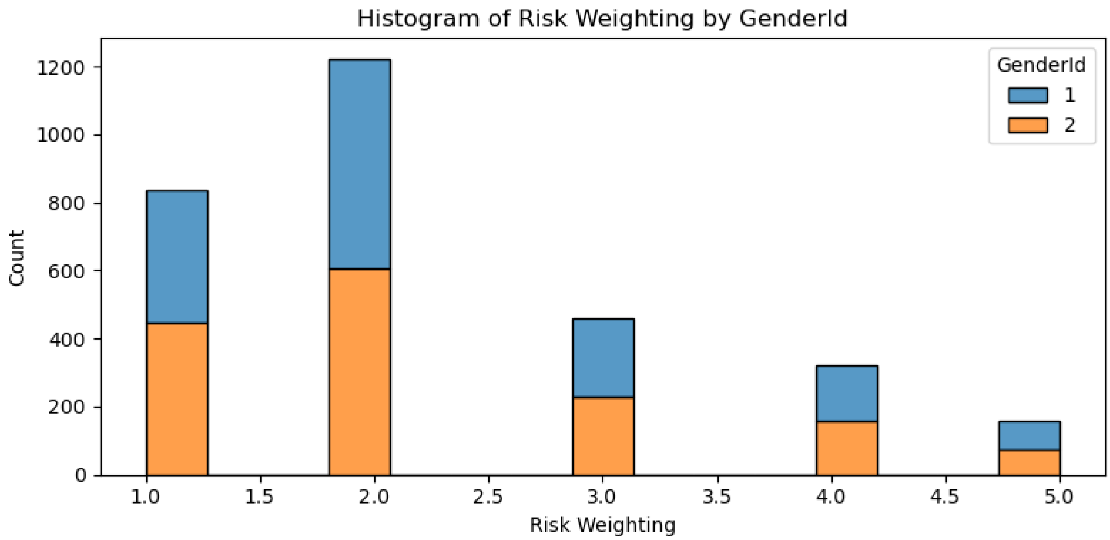

Histogram of Amount of Credit Cards by GenderId
GenderId
1
2
Count
Amount of Credit Cards
0
250
500
750
1000
1250
1500
1750
2000
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00

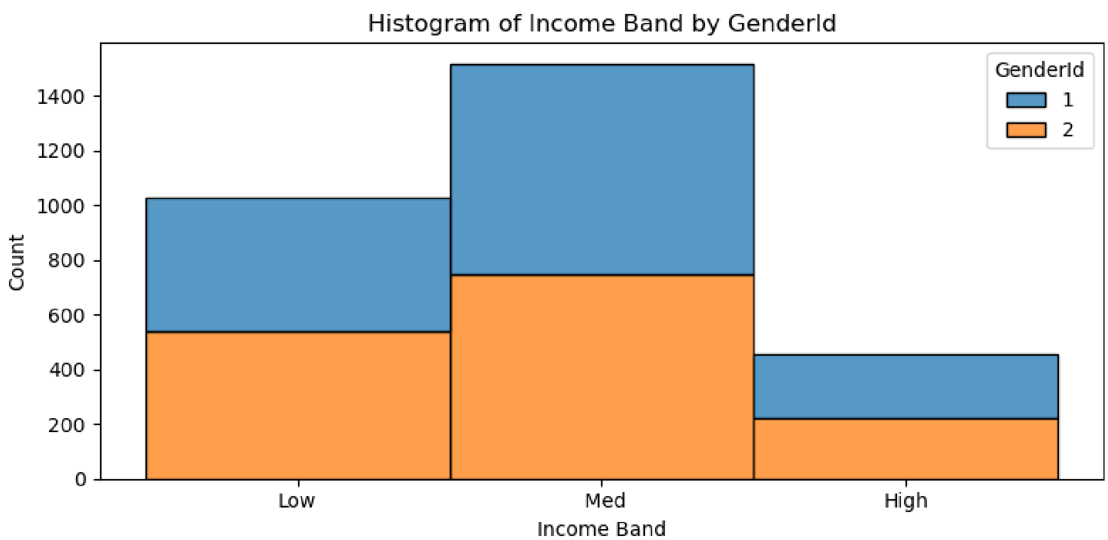

Histogram of Nationality by GenderId
GenderId
1
2
Count
0
200
400
600
800
1000
1200
American
African
European
Asian
Australian
Nationality


Histogram of Fee Structure by GenderId
GenderId
1
2
Count
0
200
400
600
800
1000
1200
1400
High
Mid
Low
Fee Structure


Histogram of Loyalty Classification by GenderId
GenderId
1
2
Count
0
200
400
600
800
1000
1200
Jade
Gold
Silver
Platinum
Loyalty Classification


Histogram of Properties Owned by GenderId
GenderId
1
2
Count
0
100
200
300
400
500
600
700
800
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Properties Owned

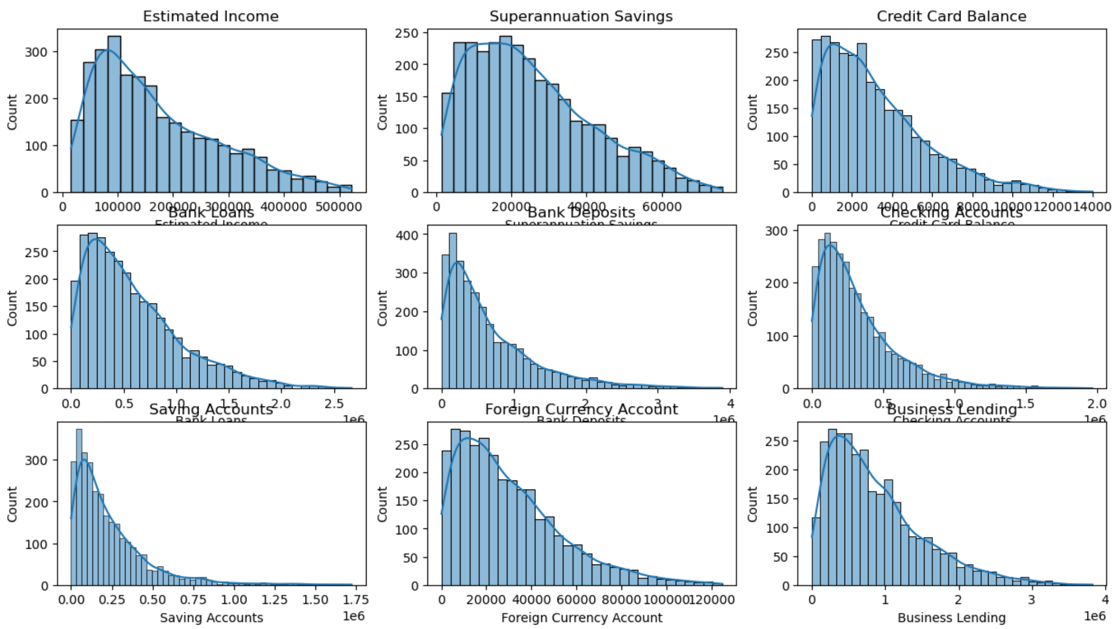




0.1 numerical analysis

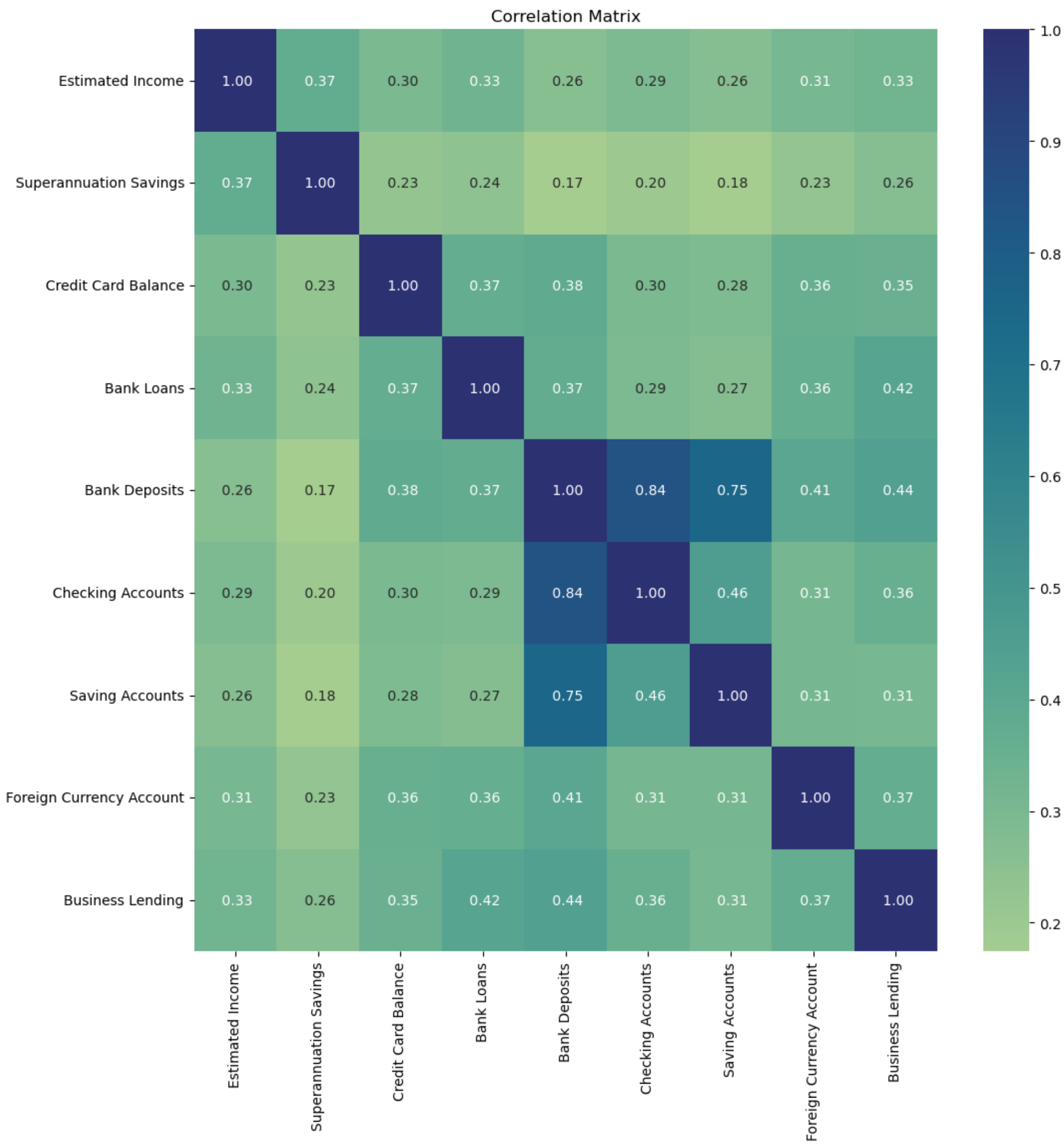
[24]:
```
numerical_cols = ["Estimated Income","Superannuation Savings",'Credit Card␣
↪Balance','Bank Loans', 'Bank Deposits', 'Checking Accounts', 'Saving␣
↪Accounts',
       'Foreign Currency Account','Business Lending']

# Univariate analysis and visualization
plt.figure(figsize = (15,8))
for i,col in enumerate(numerical_cols):
```

```
    plt.subplot(3,3,i+1)
    sns.histplot(df[col], kde = True)
    plt.title(col)
plt.show()
```

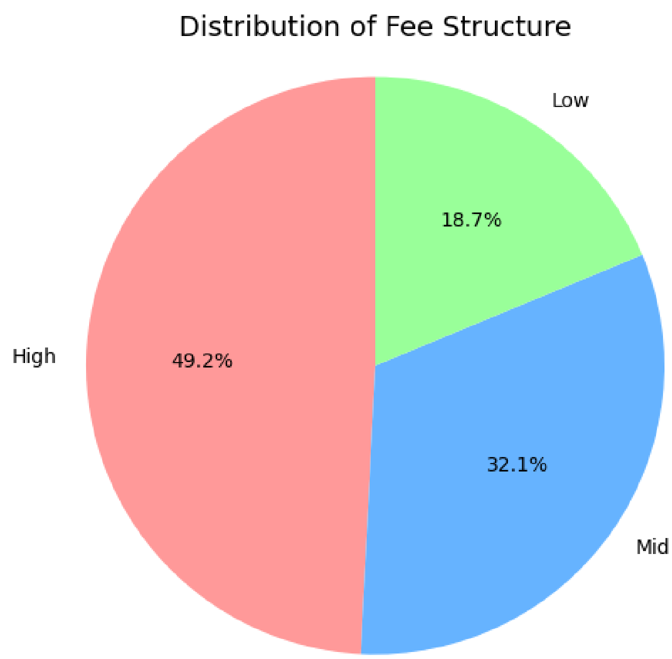


0.2 Heatmap

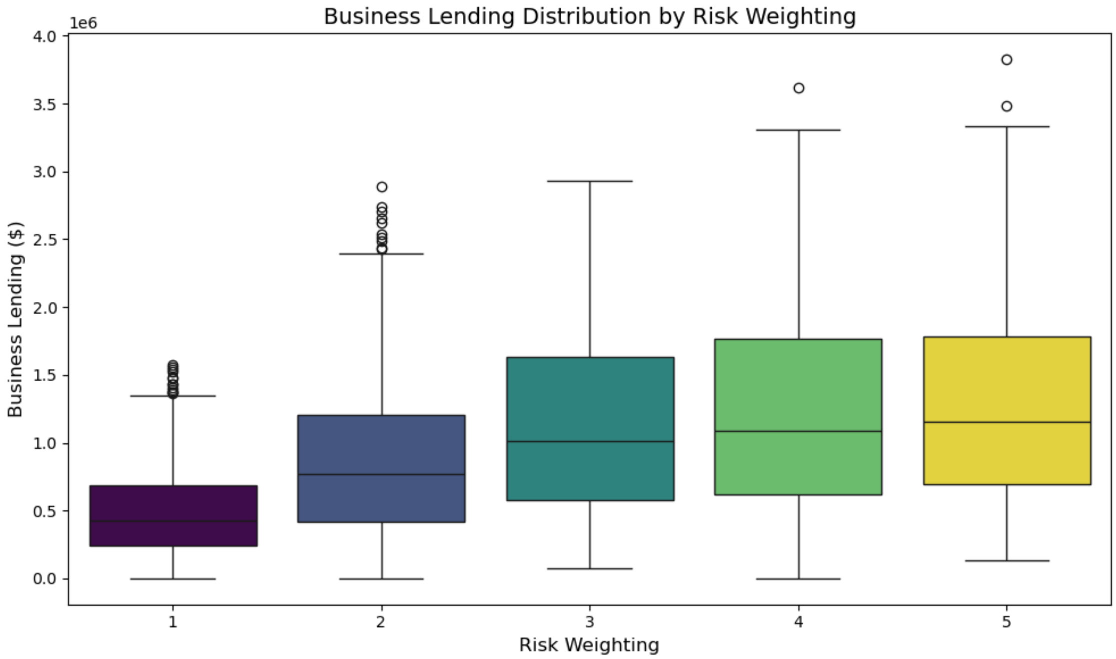
[26]:
```
numerical_cols = ["Estimated Income","Superannuation Savings",'Credit Card Balance','Bank Loans', 'Bank Deposits', 'Checking Accounts', 'Saving Accounts',
       'Foreign Currency Account','Business Lending']
correlation_matrix = df[numerical_cols].corr()

plt.figure(figsize = (12,12))
sns.heatmap(correlation_matrix, annot = True, cmap = "crest", fmt =" .2f")
plt.title("Correlation Matrix")
plt.show()
```



[51]:
```python
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(8, 5))
fee_counts = df['Fee Structure'].value_counts()
plt.pie(fee_counts, labels=fee_counts.index, autopct='%1.1f%%', startangle=90,
    colors=['#ff9999', '#66b3ff', '#99ff99'])
plt.title('Distribution of Fee Structure', fontsize=14)
plt.axis('equal')
plt.tight_layout()
plt.show()
```



[55]:
```
plt.figure(figsize=(10, 6))
sns.boxplot(x='Risk Weighting', y='Business Lending', data=df, hue='Risk␣
↪Weighting', palette='viridis', legend=False)
plt.title('Business Lending Distribution by Risk Weighting', fontsize=14)
plt.xlabel('Risk Weighting', fontsize=12)
plt.ylabel('Business Lending ($)', fontsize=12)
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```



[28]:
```
#### Insight of EDA

# 1. the strongest correlation occur among "Bank Deposits" with "Checking␣
↪Accounts", "Saving Accounts" and "Foreign Currency Account" indicating taht␣
↪customers who maintain high balance in one account type often hold␣
↪substantial amount/funds across other accounts as well.
```