

```

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

df = pd.read_csv('/banking-clients.csv')
df.head(5)

{"type": "dataframe", "variable_name": "df"}

df.shape

(3000, 25)

df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3000 entries, 0 to 2999
Data columns (total 25 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   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

# Genrate discriptive statistics for the dataframe
df.describe()

```

```
{
  "summary": {
    "name": "df",
    "rows": 8,
    "fields": [
      {
        "column": "Age",
        "properties": {
          "dtype": "number",
          "std": 1044.4070732954572,
          "min": 17.0,
          "max": 3000.0,
          "num_unique_values": 8,
          "samples": [
            51.03966666666667,
            51.0,
            3000.0
          ],
          "semantic_type": "",
          "description": ""
        },
        "column": "Location ID",
        "properties": {
          "dtype": "number",
          "std": 14612.18148735417,
          "min": 12.0,
          "max": 43369.0,
          "num_unique_values": 8,
          "samples": [
            21563.323,
            21129.5,
            3000.0
          ],
          "semantic_type": "",
          "description": ""
        },
        "column": "Estimated Income",
        "properties": {
          "dtype": "number",
          "std": 165675.79415446558,
          "min": 3000.0,
          "max": 522330.26,
          "num_unique_values": 8,
          "samples": [
            171305.03426333333,
            142313.47999999998,
            3000.0
          ],
          "semantic_type": "",
          "description": ""
        },
        "column": "Superannuation Savings",
        "properties": {
          "dtype": "number",
          "std": 23834.521347506627,
          "min": 1482.03,
          "max": 75963.9,
          "num_unique_values": 8,
          "samples": [
            25531.59967333333,
            22357.355000000003,
            3000.0
          ],
          "semantic_type": "",
          "description": ""
        },
        "column": "Amount of Credit Cards",
        "properties": {
          "dtype": "number",
          "std": 1060.1482852588065,
          "min": 0.6763867645368546,
          "max": 3000.0,
          "num_unique_values": 6,
          "samples": [
            3000.0,
            1.4636666666666667,
            3.0
          ],
          "semantic_type": "",
          "description": ""
        },
        "column": "Credit Card Balance",
        "properties": {
          "dtype": "number",
          "std": 4303.156900678545,
          "min": 1.17,
          "max": 13991.99,
          "num_unique_values": 8,
          "samples": [
            3176.2069433333336,
            2560.8050000000003,
            3000.0
          ],
          "semantic_type": "",
          "description": ""
        },
        "column": "Bank Loans",
        "properties": {
          "dtype": "number",
          "std": 860222.1898574389,
          "min": 0.0,
          "max": 2667556.66,
          "num_unique_values": 8,
          "samples": [
            591386.1554866667,
            479793.4,
            3000.0
          ],
          "semantic_type": "",
          "description": ""
        },
        "column": "Bank Deposits",
        "properties": {
          "dtype": "number",
          "std": 1272432.2911375419,
          "min": 0.0,
          "max": 3890598.08,
          "num_unique_values": 8,
          "samples": [
            671560.1939233334,
            671560.1939233334,
            671560.1939233334
          ],
          "semantic_type": "",
          "description": ""
        }
      ]
    }
  }
}
```

```

463316.46,\n          3000.0\n          ],\n          \"semantic_type\":\n          \"\",\n          \"description\": \"\"\n          },\n          {\n          \"column\": \"Checking Accounts\",\n          \"properties\": {\n          \"dtype\": \"number\",\n          \"std\": 643980.7752101668,\n          \"min\": 0.0,\n          \"max\": 1969923.08,\n          \"num_unique_values\": 8,\n          \"samples\": [\n          321092.94912666664,\n          242815.655,\n          3000.0\n          ],\n          \"semantic_type\": \"\",\n          \"description\": \"\"\n          },\n          {\n          \"column\": \"Saving Accounts\",\n          \"properties\": {\n          \"dtype\": \"number\",\n          \"std\": 569501.1225021764,\n          \"min\": 0.0,\n          \"max\": 1724118.36,\n          \"num_unique_values\": 8,\n          \"samples\": [\n          164086.555,\n          3000.0\n          ],\n          \"semantic_type\": \"\",\n          \"description\": \"\"\n          },\n          {\n          \"column\": \"Foreign Currency Account\",\n          \"properties\": {\n          \"dtype\": \"number\",\n          \"std\": 39821.13354767674,\n          \"min\": 45.0,\n          \"max\": 124704.87,\n          \"num_unique_values\": 8,\n          \"samples\": [\n          29883.529993333334,\n          24341.190000000002,\n          3000.0\n          ],\n          \"semantic_type\": \"\",\n          \"description\": \"\"\n          },\n          {\n          \"column\": \"Business Lending\",\n          \"properties\": {\n          \"dtype\": \"number\",\n          \"std\": 1231479.8807215113,\n          \"min\": 0.0,\n          \"max\": 3825961.94,\n          \"num_unique_values\": 8,\n          \"samples\": [\n          866759.8084066667,\n          711314.6599999999,\n          3000.0\n          ],\n          \"semantic_type\": \"\",\n          \"description\": \"\"\n          },\n          {\n          \"column\": \"Properties Owned\",\n          \"properties\": {\n          \"dtype\": \"number\",\n          \"std\": 1060.1241040744355,\n          \"min\": 0.0,\n          \"max\": 3000.0,\n          \"num_unique_values\": 7,\n          \"samples\": [\n          3000.0,\n          1.5186666666666666,\n          2.0\n          ],\n          \"semantic_type\": \"\",\n          \"description\": \"\"\n          },\n          {\n          \"column\": \"Risk Weighting\",\n          \"properties\": {\n          \"dtype\": \"number\",\n          \"std\": 1059.8841834225843,\n          \"min\": 1.0,\n          \"max\": 3000.0,\n          \"num_unique_values\": 7,\n          \"samples\": [\n          3000.0,\n          2.2493333333333334,\n          3.0\n          ],\n          \"semantic_type\": \"\",\n          \"description\": \"\"\n          },\n          {\n          \"column\": \"BRId\",\n          \"properties\": {\n          \"dtype\": \"number\",\n          \"std\": 1059.8239053751968,\n          \"min\": 1.0,\n          \"max\": 3000.0,\n          \"num_unique_values\": 7,\n          \"samples\": [\n          3000.0,\n          2.5593333333333335,\n          3.0\n          ],\n          \"semantic_type\": \"\",\n          \"description\": \"\"\n          },\n          {\n          \"column\": \"GenderId\",\n          \"properties\": {\n          \"dtype\": \"number\",\n          \"std\": 1060.1550392950937,\n          \"min\": 0.5000673512490724,\n          \"max\": 3000.0,\n          \"num_unique_values\": 5,\n          \"samples\": [\n          1.504,\n          2.0,\n          ]

```

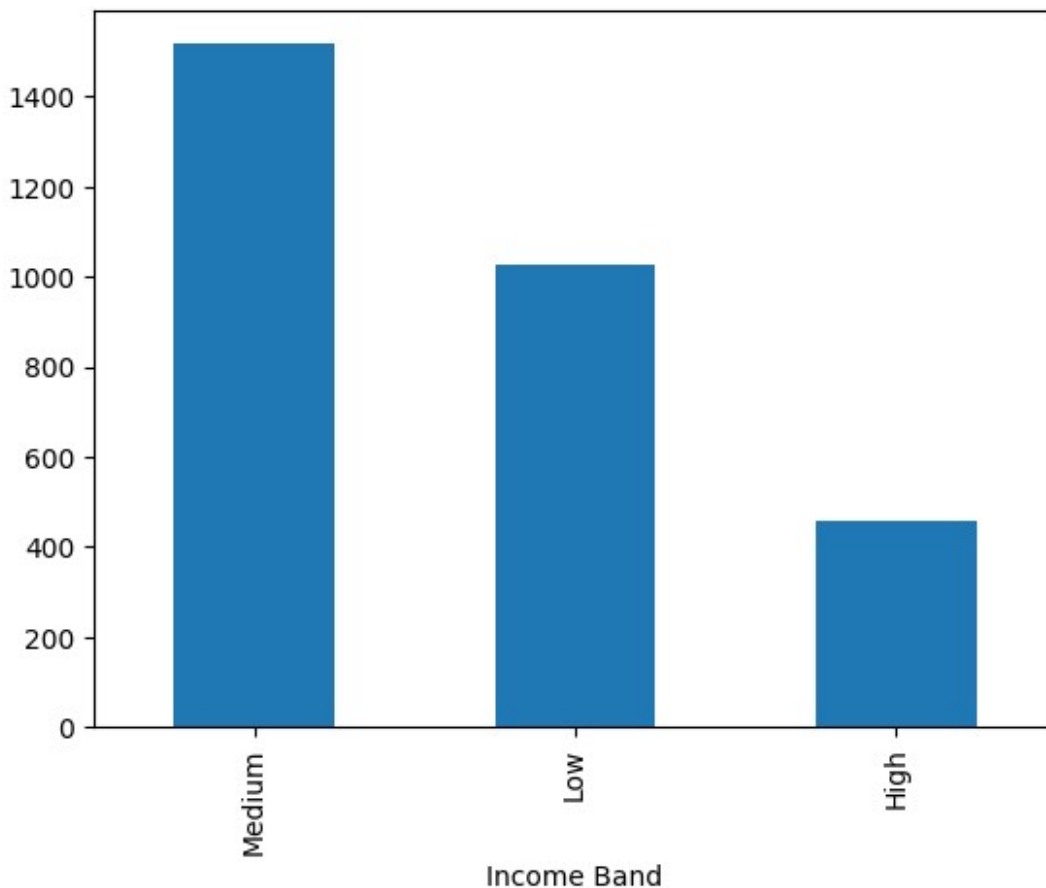
```
0.5000673512490724\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n    },\n    {\n        \"column\":\n        \"IAId\",\n        \"properties\": {\n            \"dtype\": \"number\",\n            \"std\": 1057.1739338184325,\n            \"min\": 1.0,\n            \"max\": 3000.0,\n            \"num_unique_values\": 8,\n            \"samples\": [\n                10.425333333333333,\n                10.0,\n                3000.0\n            ],\n            \"semantic_type\": \"\",\n            \"description\": \"\"\n        }\n    }\n]\n}","type":"dataframe"}
```

```
bins = [0,100000,300000,float('inf')]\nlabels = ['Low','Medium','High']
```

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

```
df['Income Band'].value_counts().plot(kind="bar")
```

```
<Axes: xlabel='Income Band'>
```



```
# Examine the distribution of unique categories in categorical columns
categorical_cols = df[["BRIId", "GenderId", "IAId", "Amount of Credit\nCards", "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())
```

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  
2       177  
3       177  
4       177  
8       177  
9       176  
13      176  
12      176  
10      176  
11      176  
14      176  
15      176  
6        89  
5        89  
7        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

Associate Professor	28
Structural Analysis Engineer	28
Recruiter	25
Account Coordinator	24
Human Resources Manager	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
Medium      1517
Low         1027
High         456
Name: count, dtype: int64

```

Univariate Analysis

```

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)

```

```

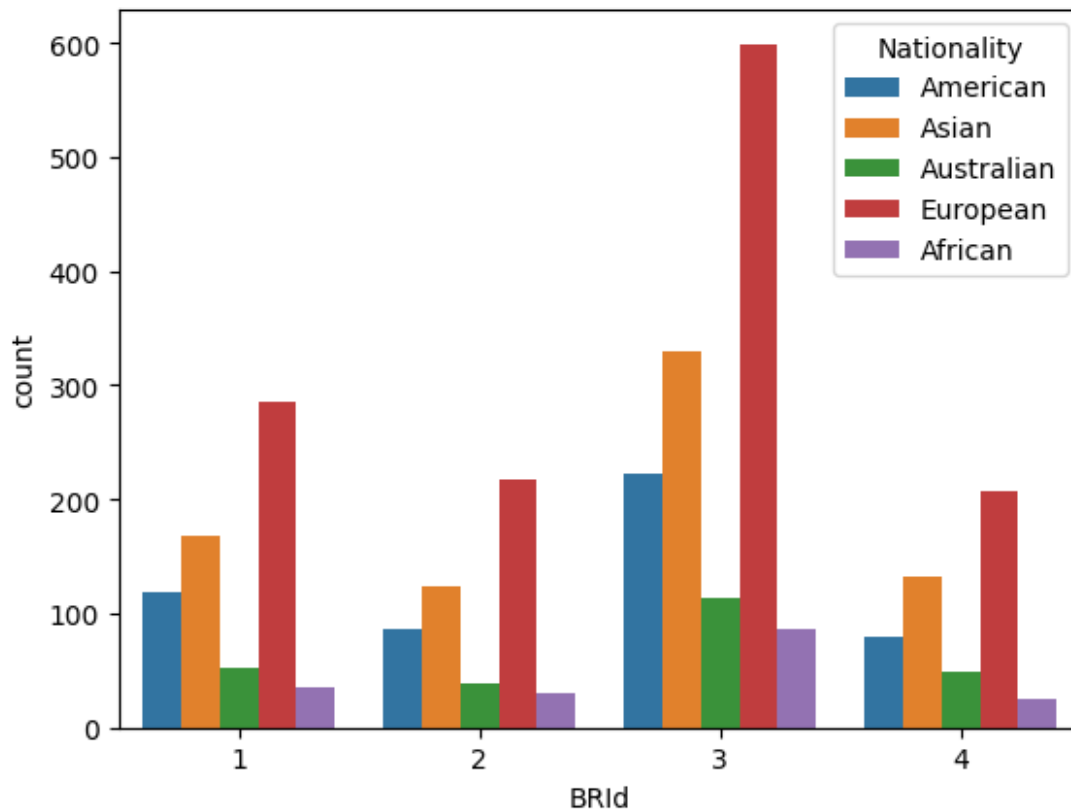
-----
-----
NameError                                Traceback (most recent call
last)
<ipython-input-1-533aad80979d> in <cell line: 0>()
----> 1 for i, predictor in enumerate(df[["BRId", "GenderId", "IAId",
"Amount of Credit Cards", "Nationality", "Occupation", "Fee Structure",
"Loyalty Classification", "Properties Owned", "Risk Weighting", "Income
Band"]].columns):
      2     plt.figure(i)
      3     sns.countplot(data = df, x = predictor)

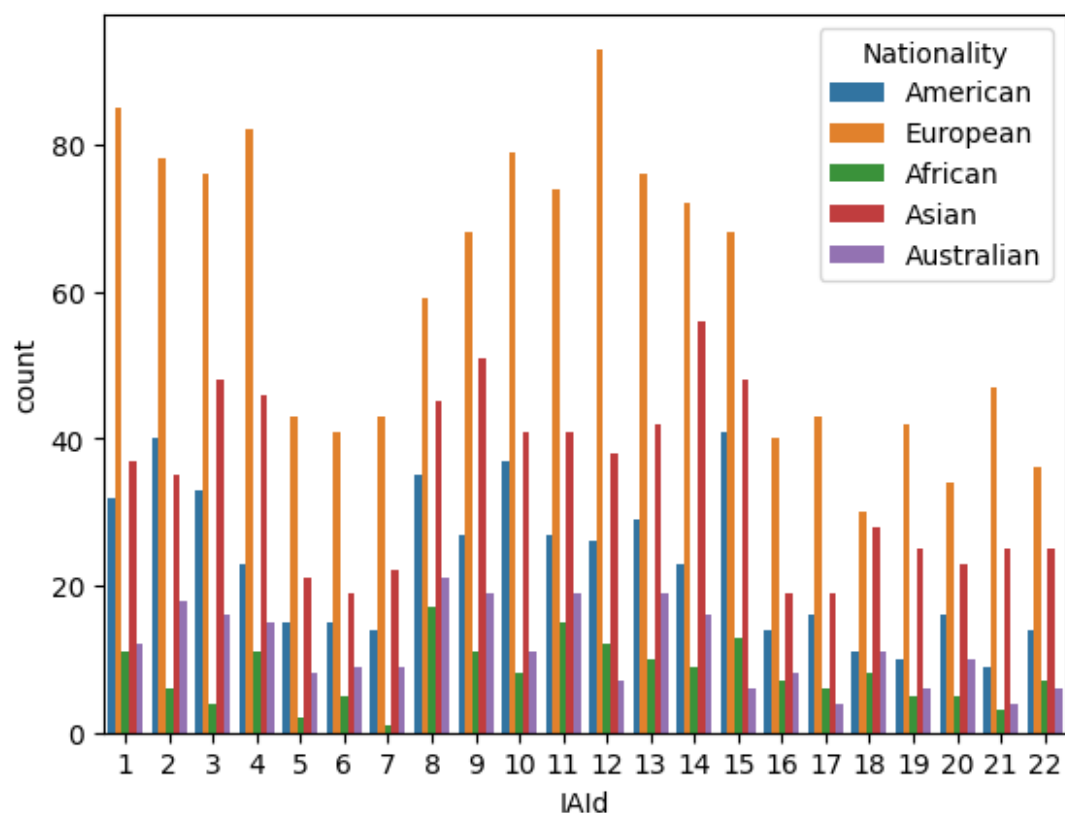
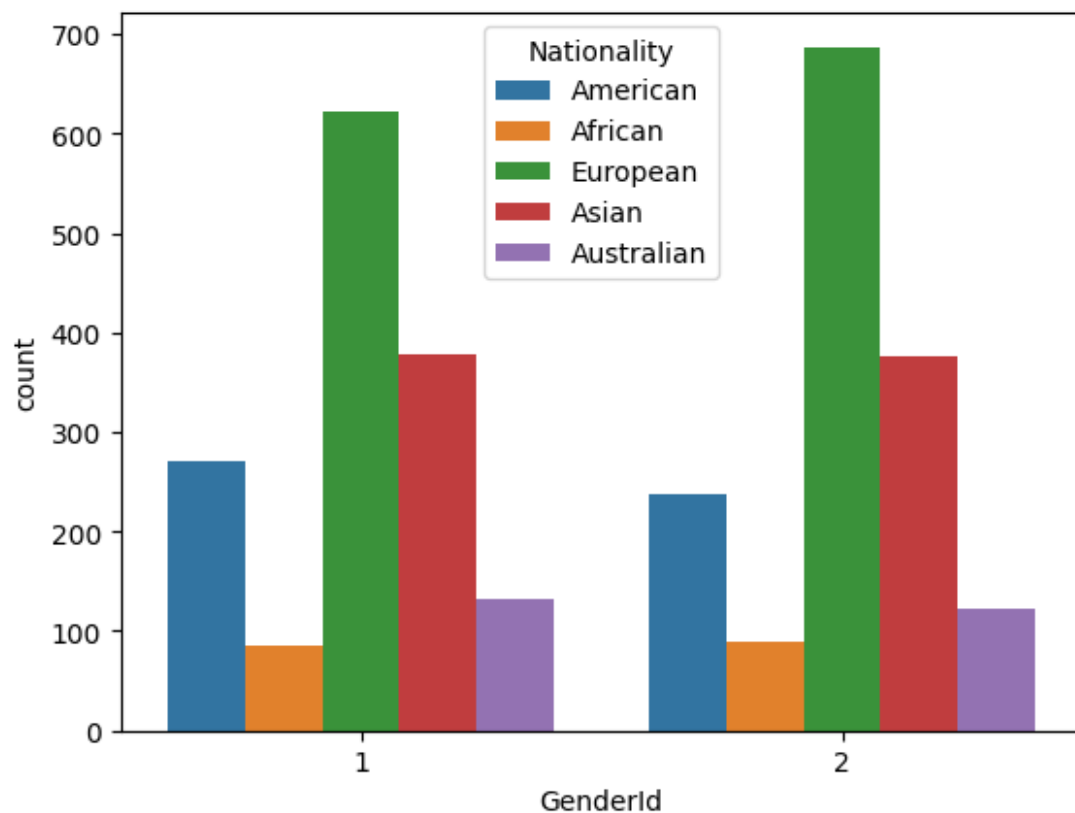
NameError: name 'df' is not defined

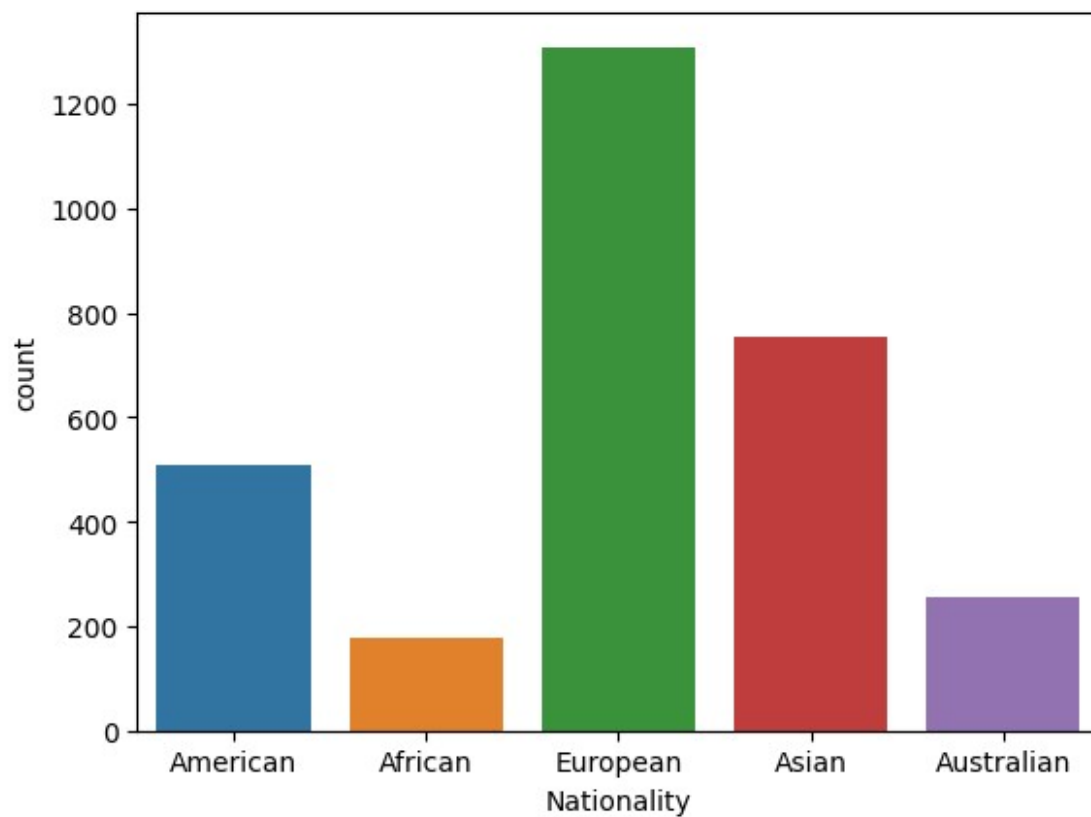
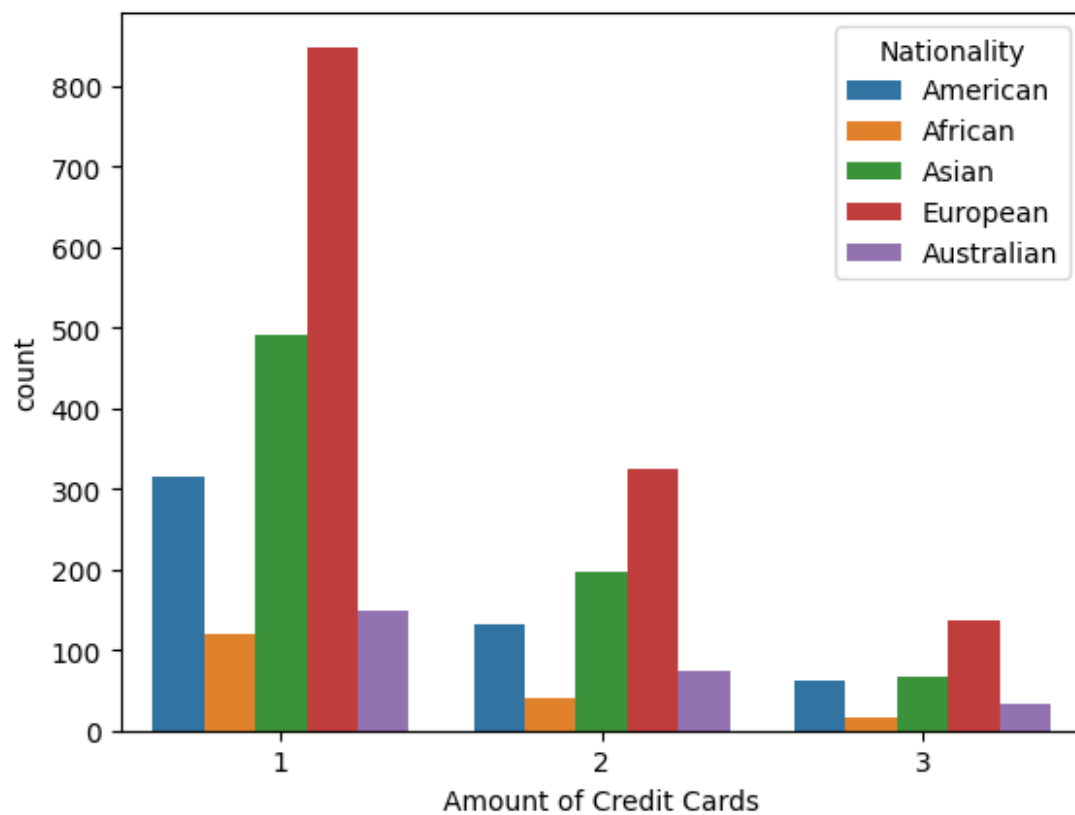
```

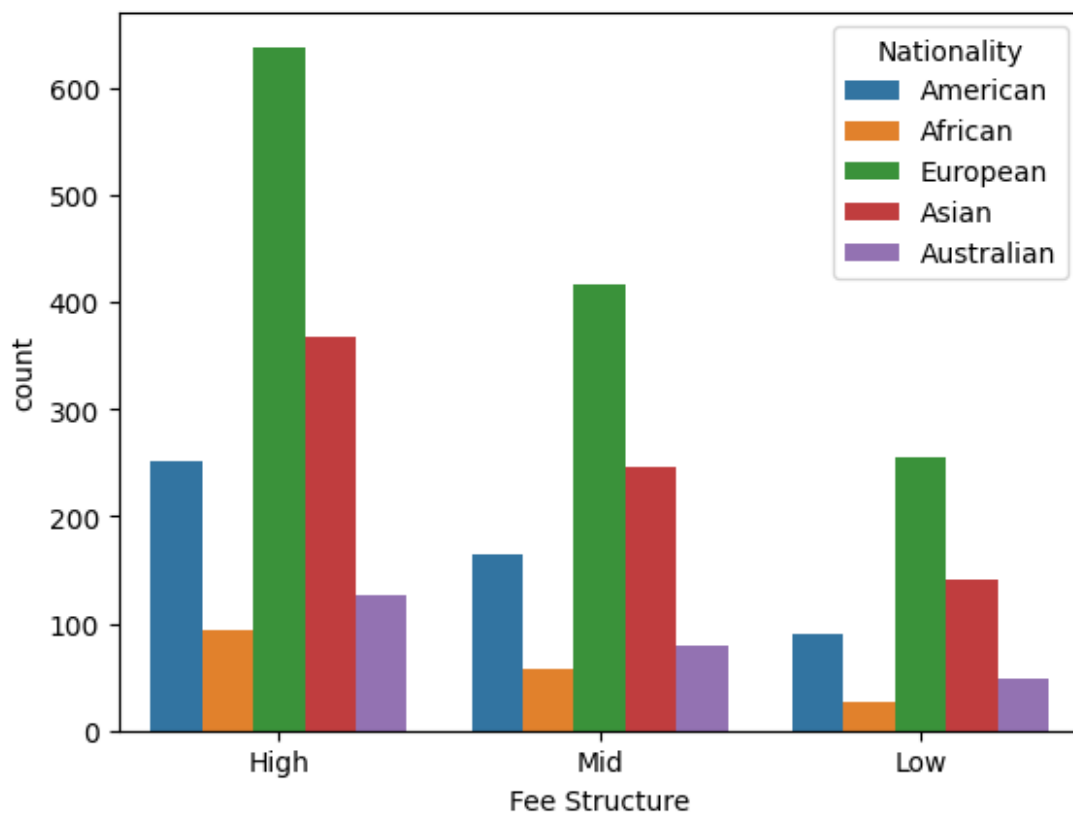
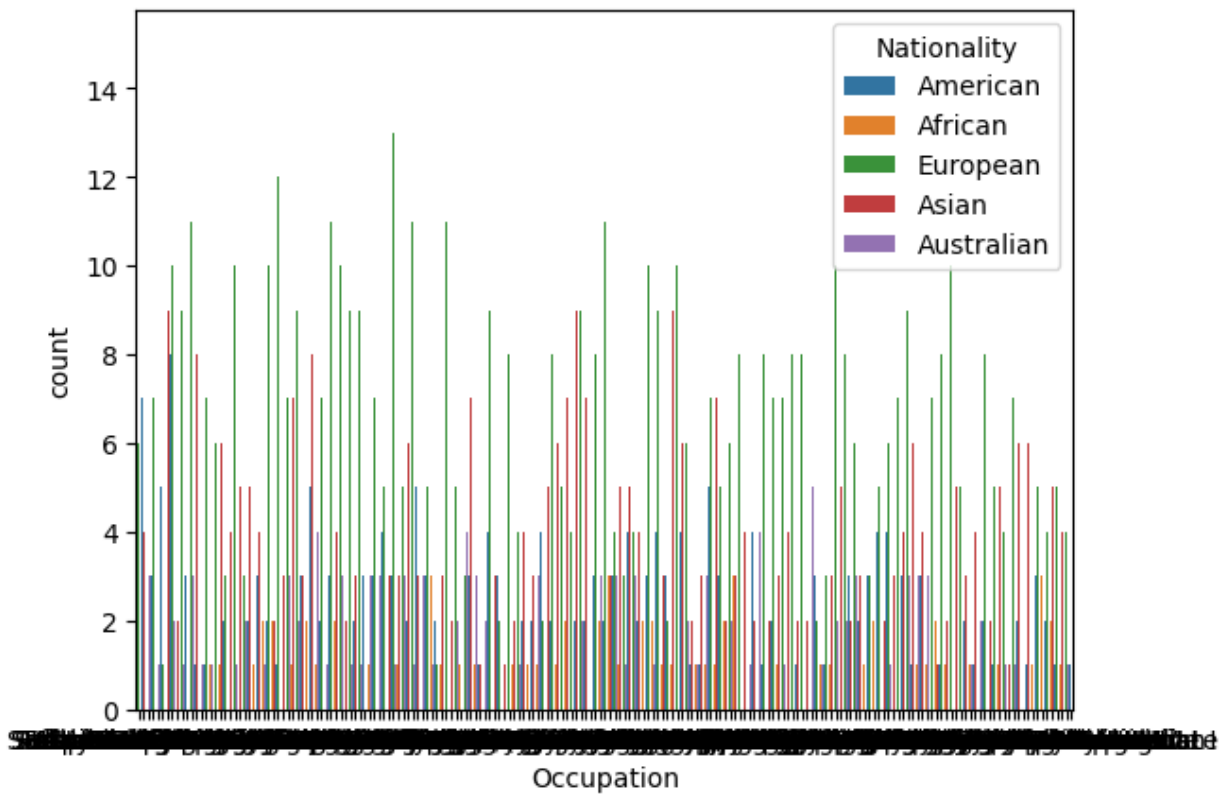
Bivariate Analysis

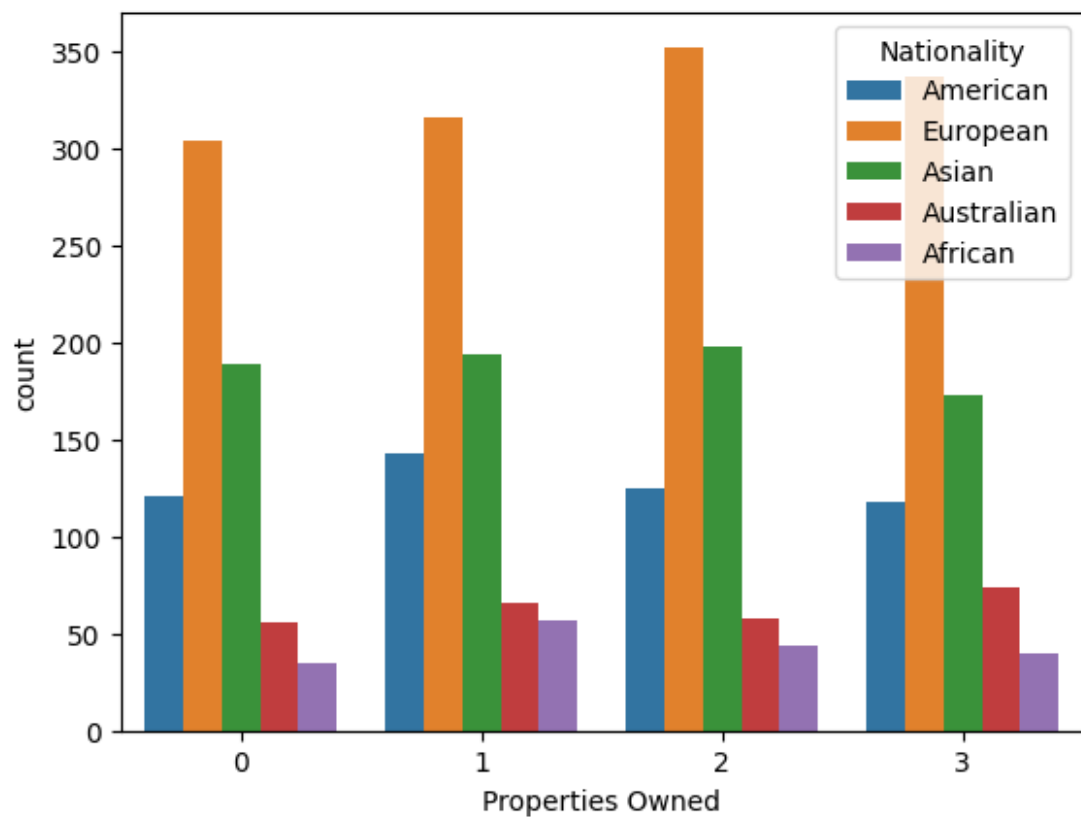
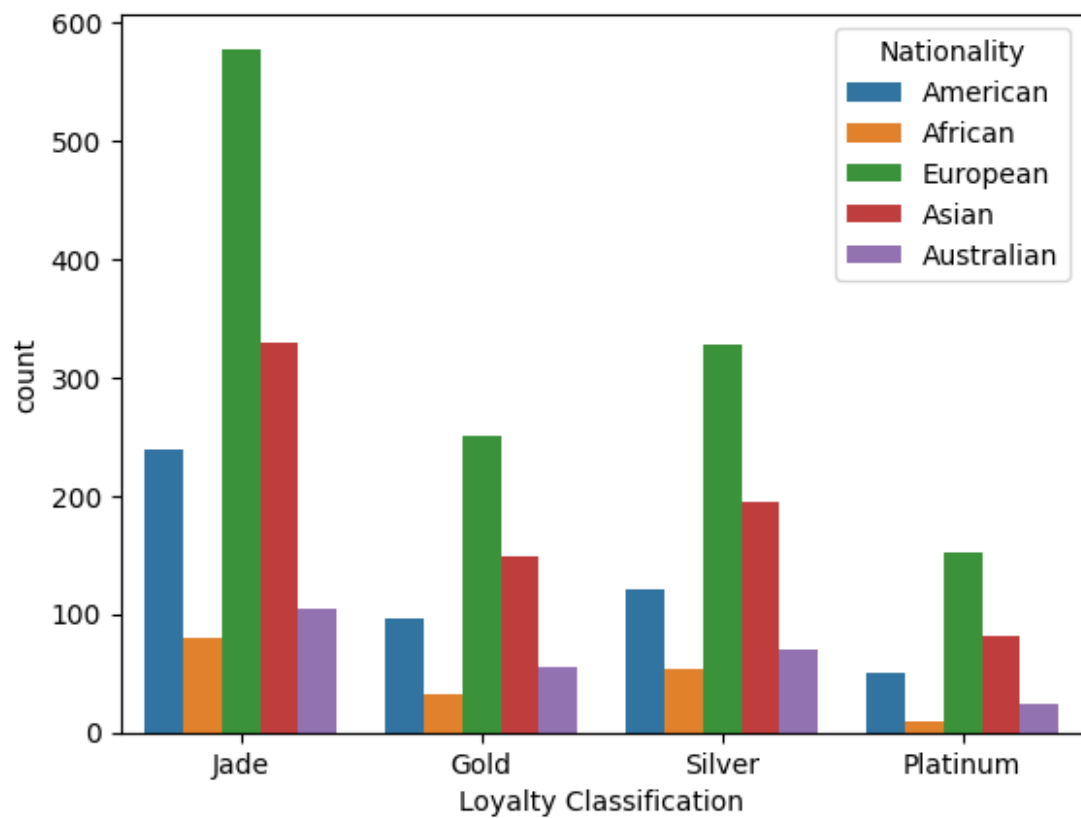
```
for i, predictor in enumerate(df[["BRId", "GenderId", "IAId", "Amount of  
Credit Cards", "Nationality", "Occupation", "Fee Structure", "Loyalty  
Classification", "Properties Owned", "Risk Weighting", "Income  
Band"]].columns):  
    plt.figure(i)  
    sns.countplot(data = df, x = predictor, hue='Nationality')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

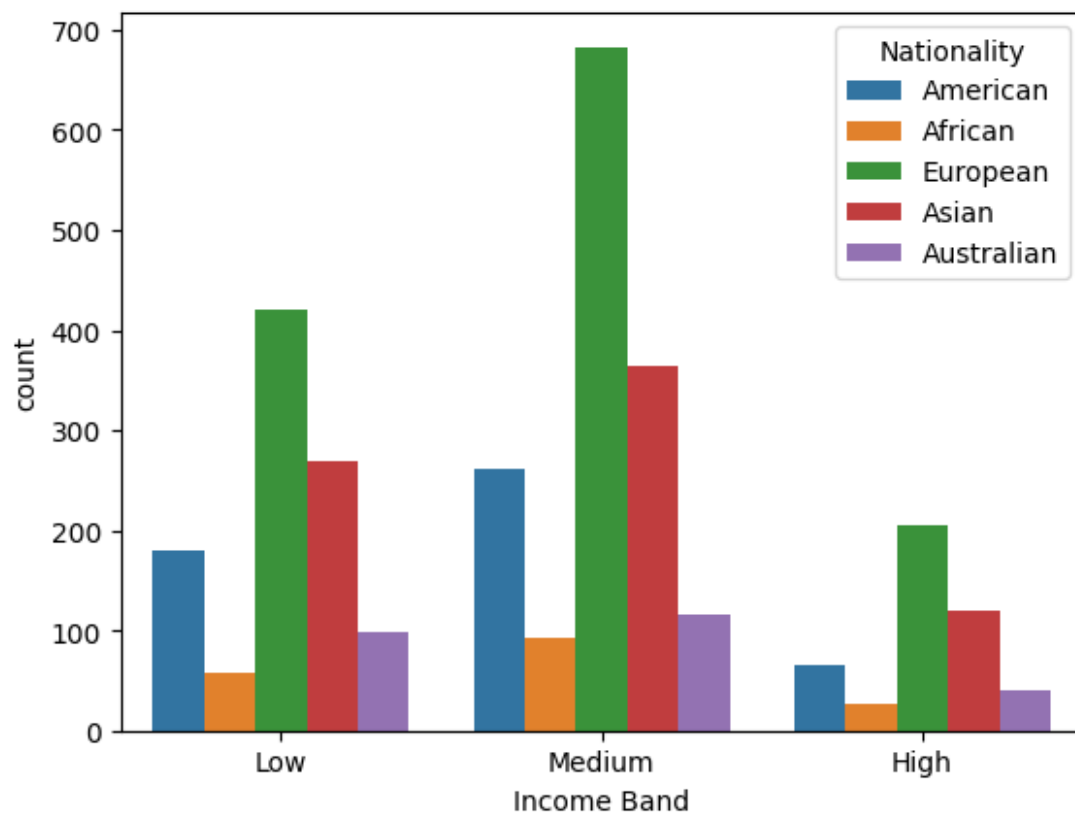
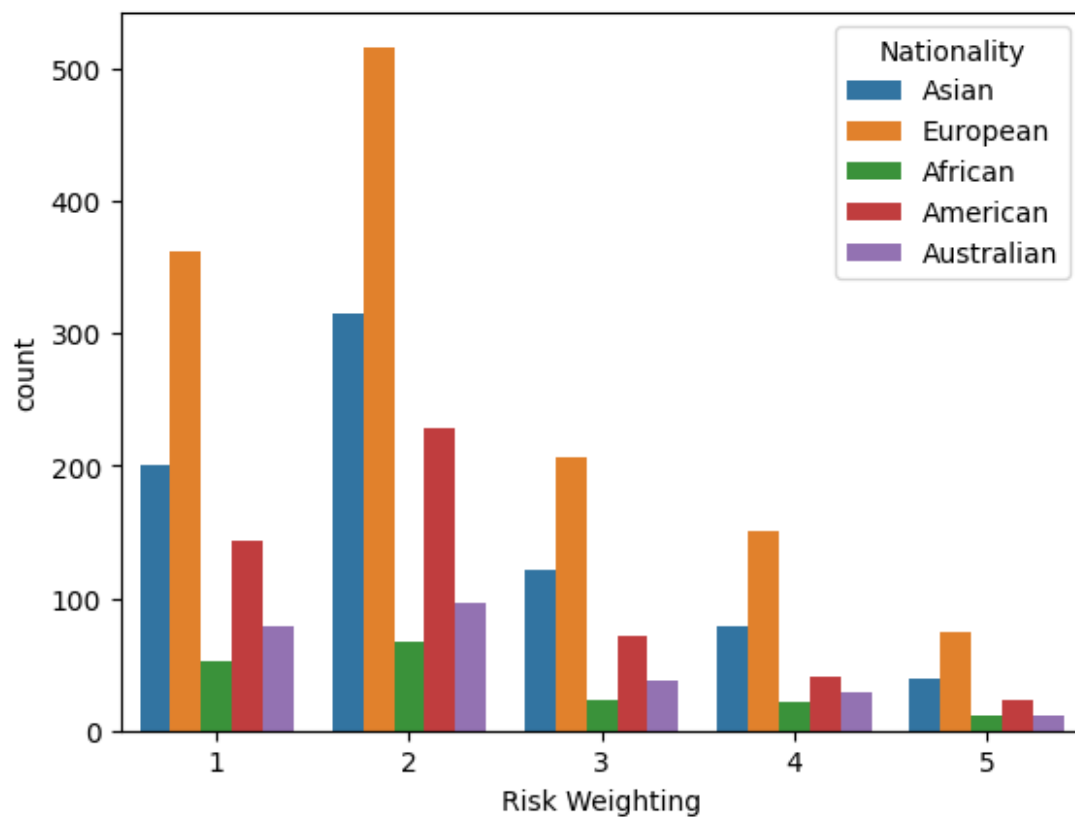






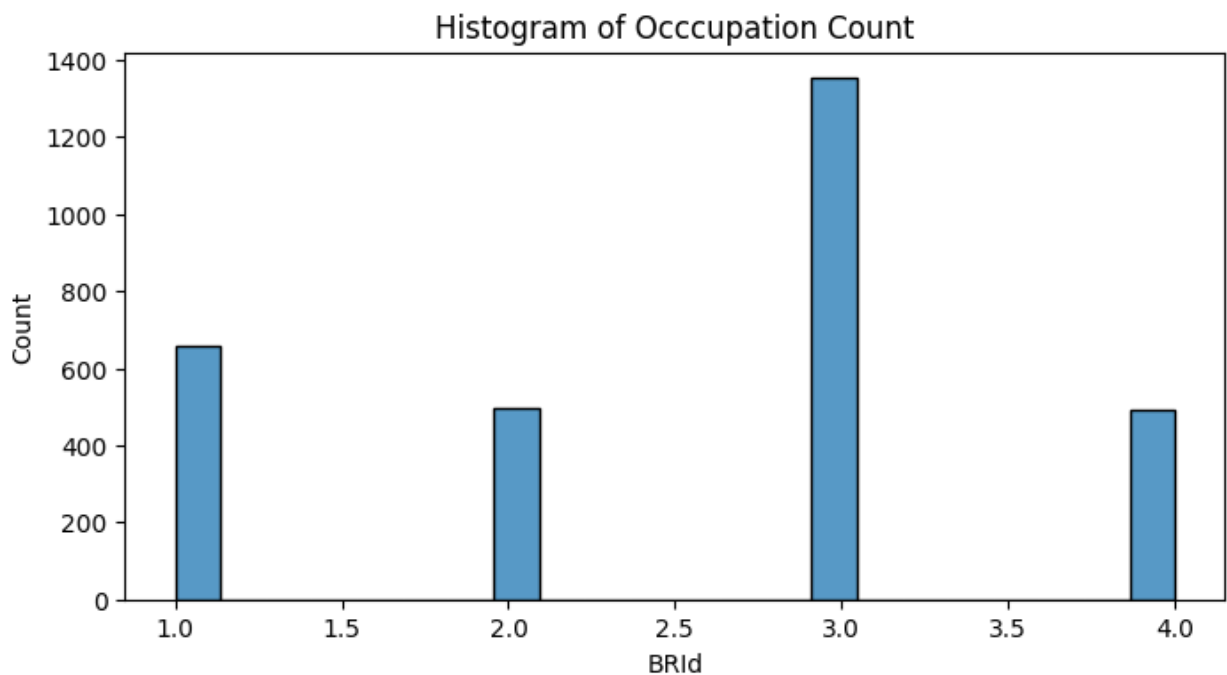


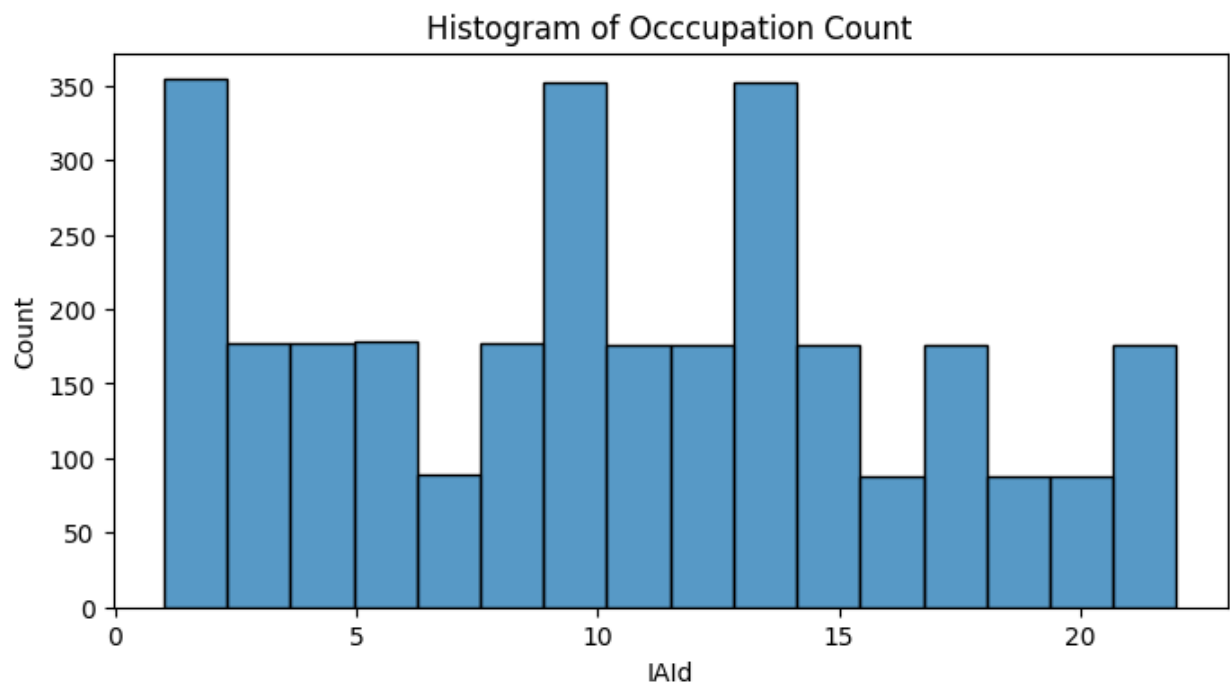
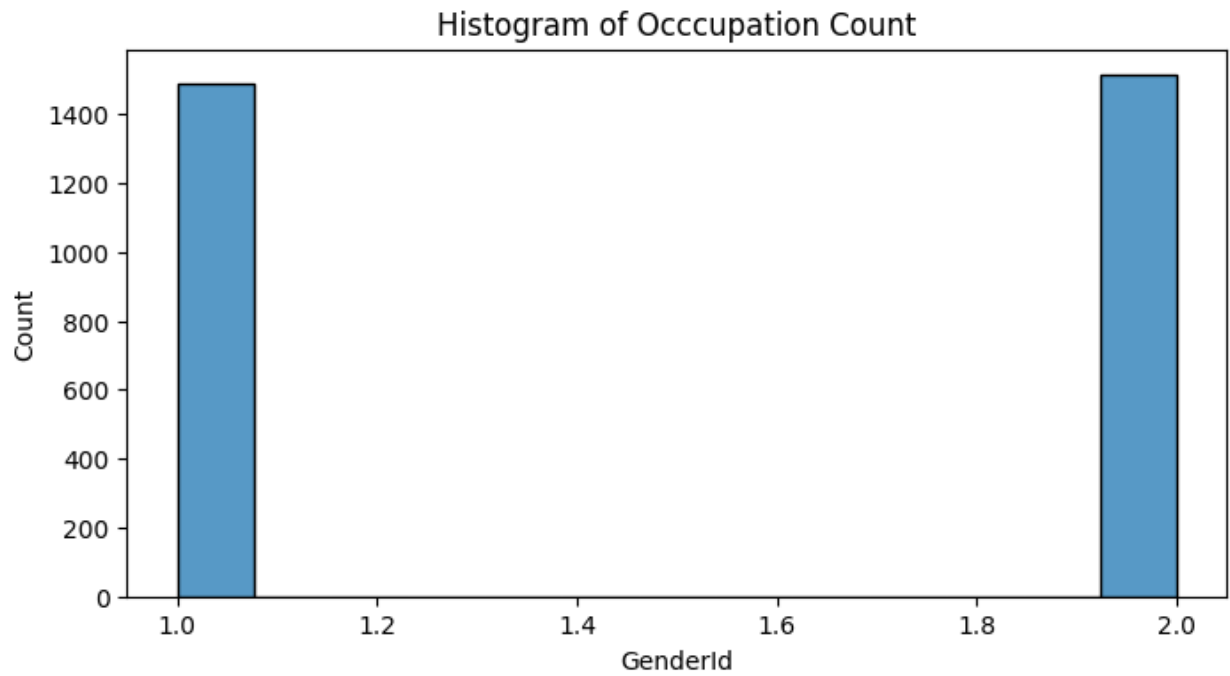


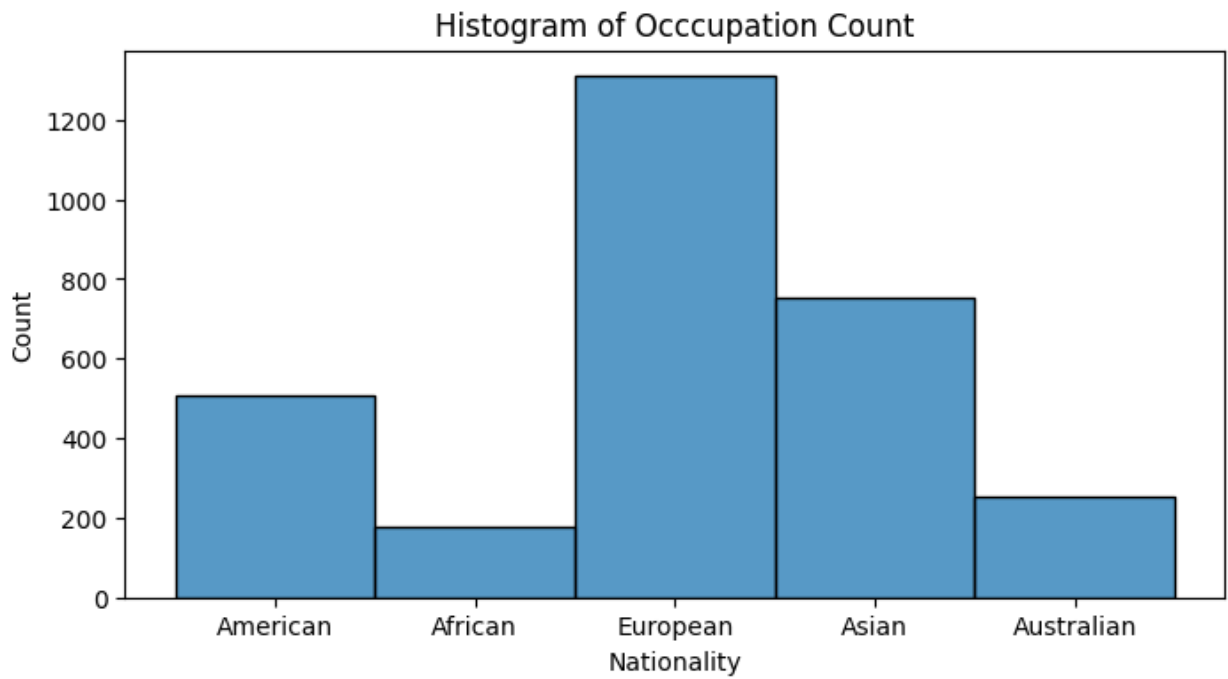
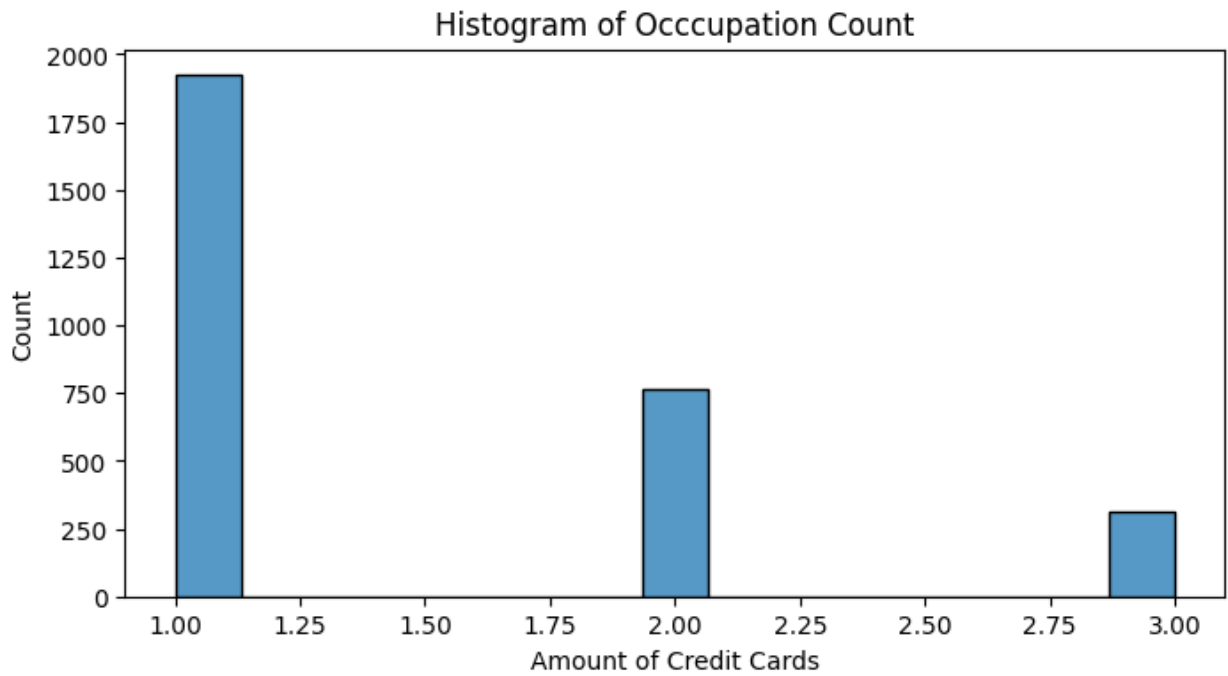


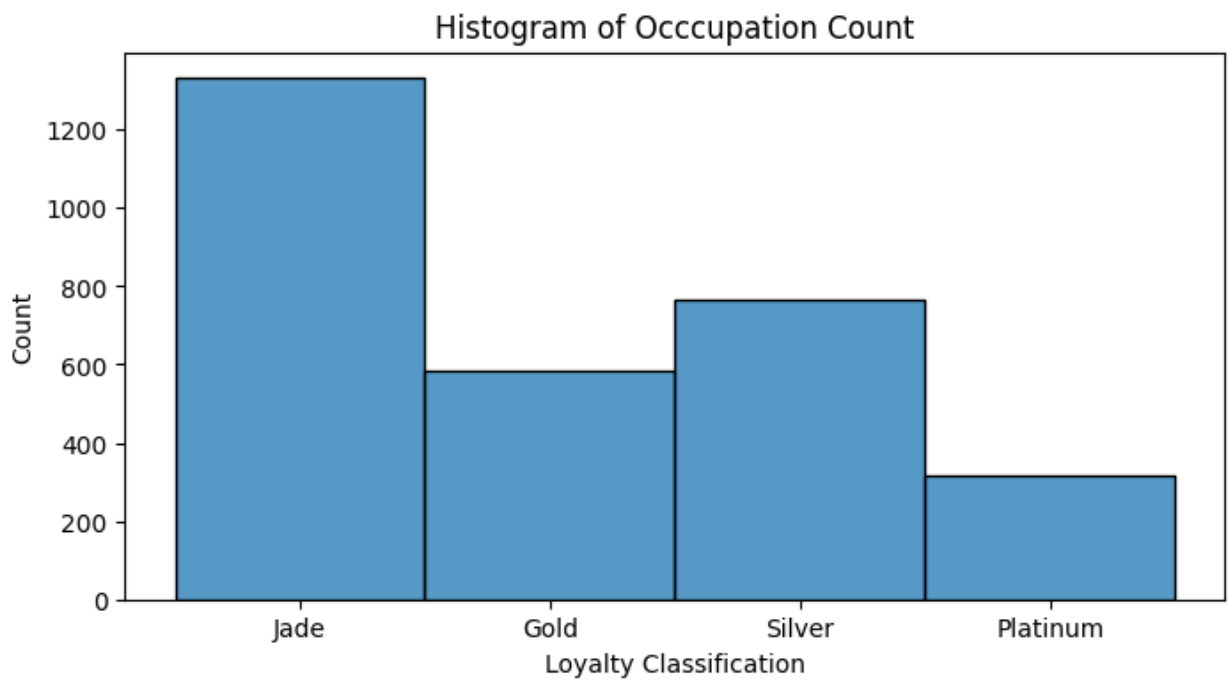
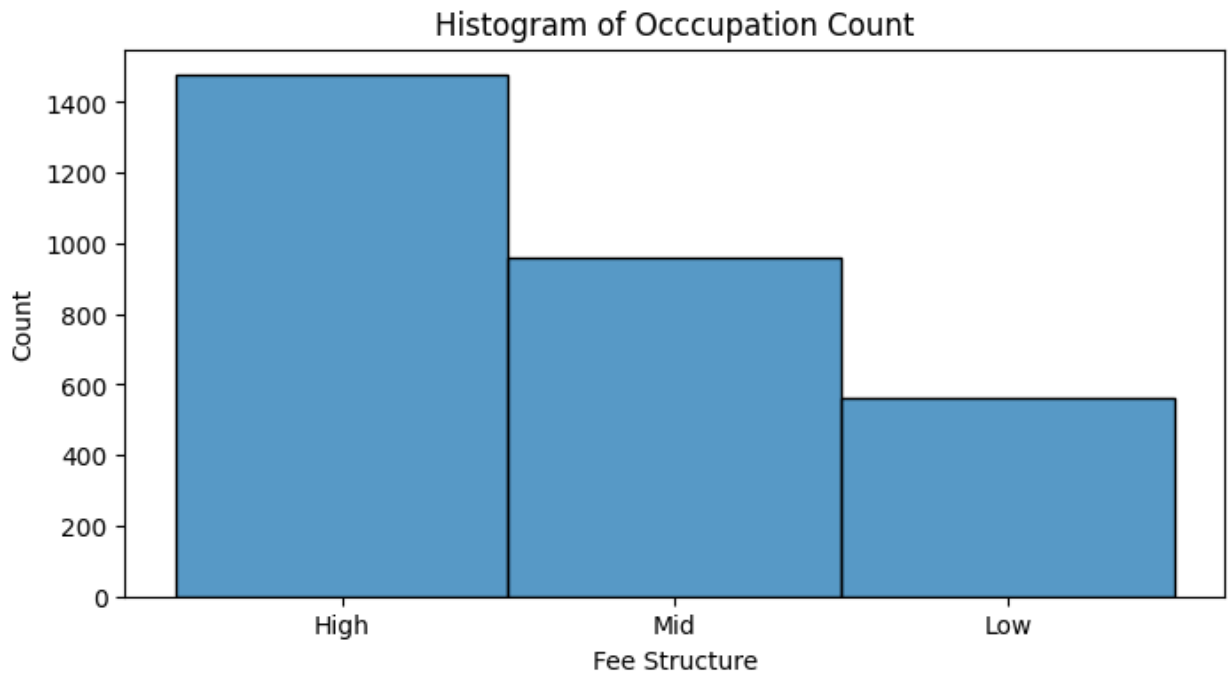
```
# Hisplot of Value Counts for different Occupations
```

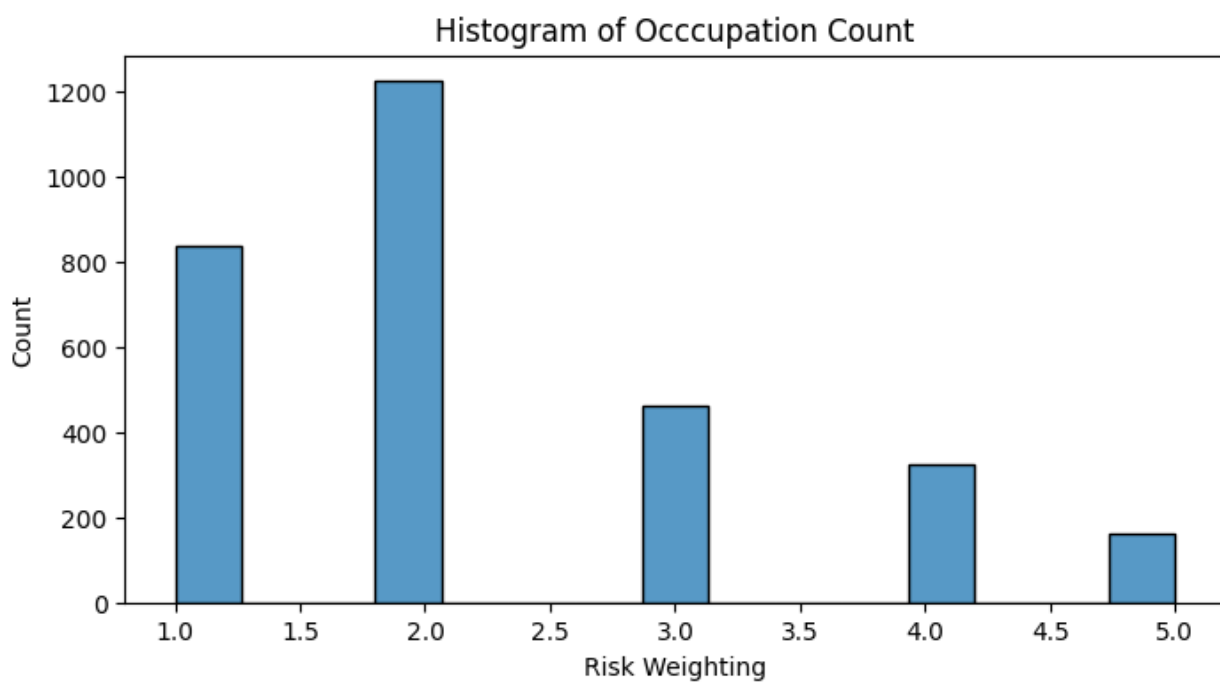
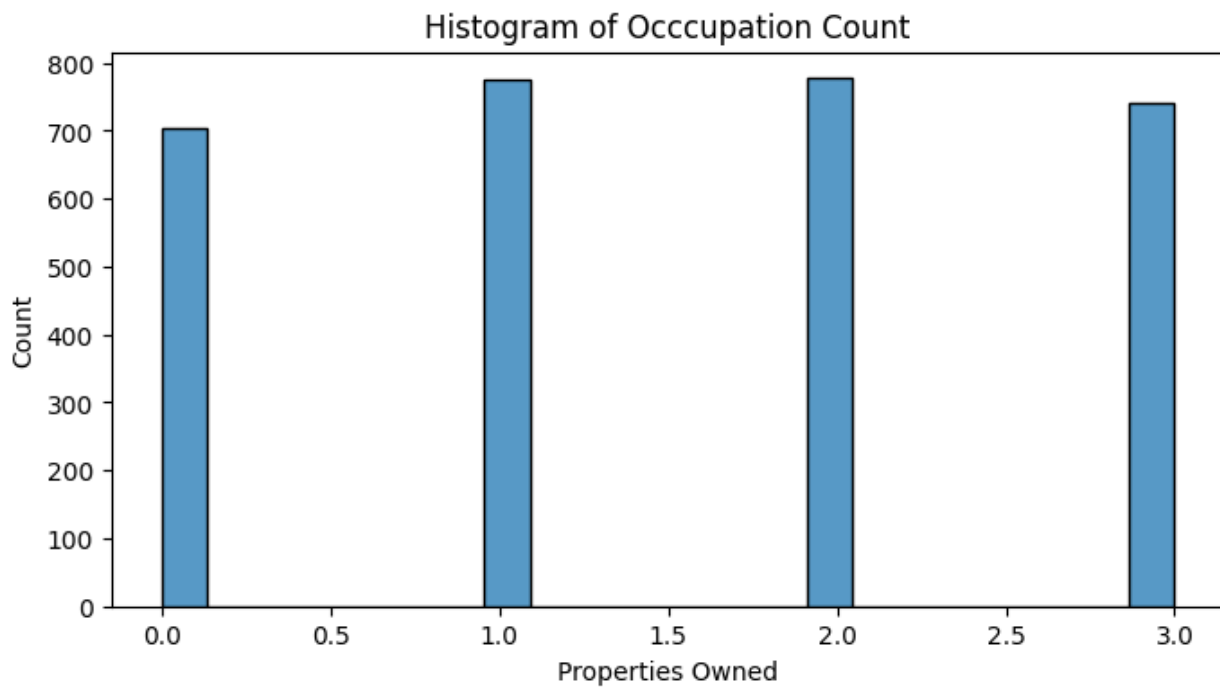
```
for col in categorical_cols:  
    if col == "Occupation":  
        continue  
    plt.figure(figsize=(8,4))  
    sns.histplot(df[col])  
    plt.title('Histogram of Occcupation Count')  
    plt.xlabel(col)  
    plt.ylabel("Count")  
    plt.show()
```

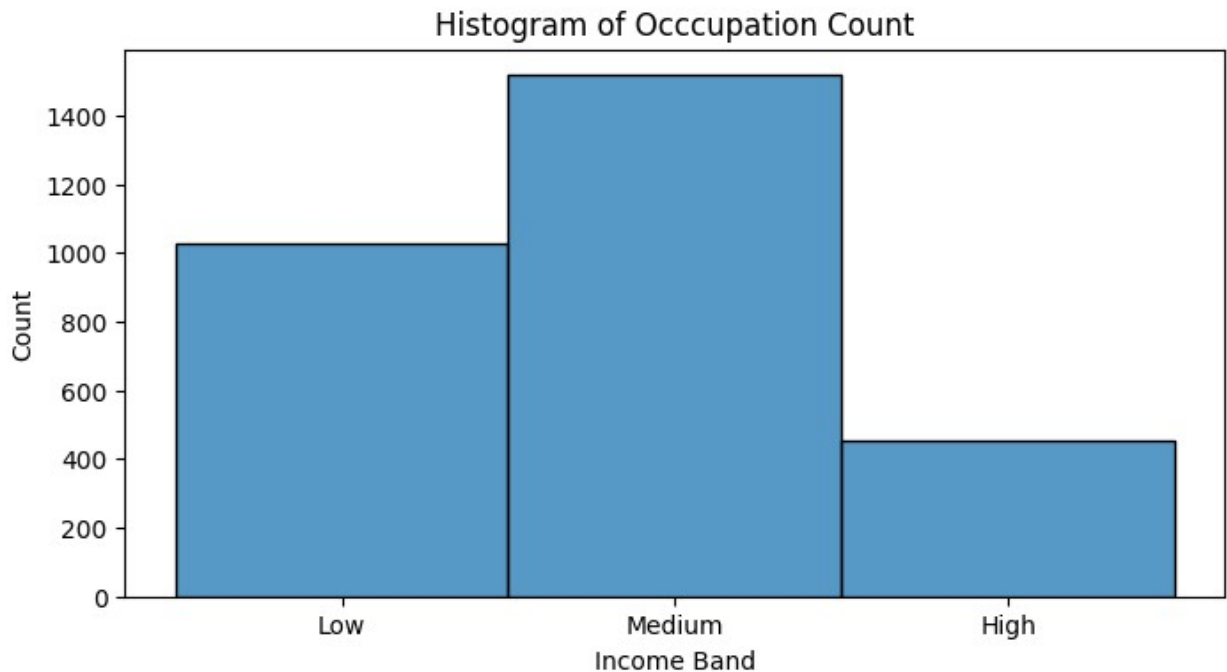










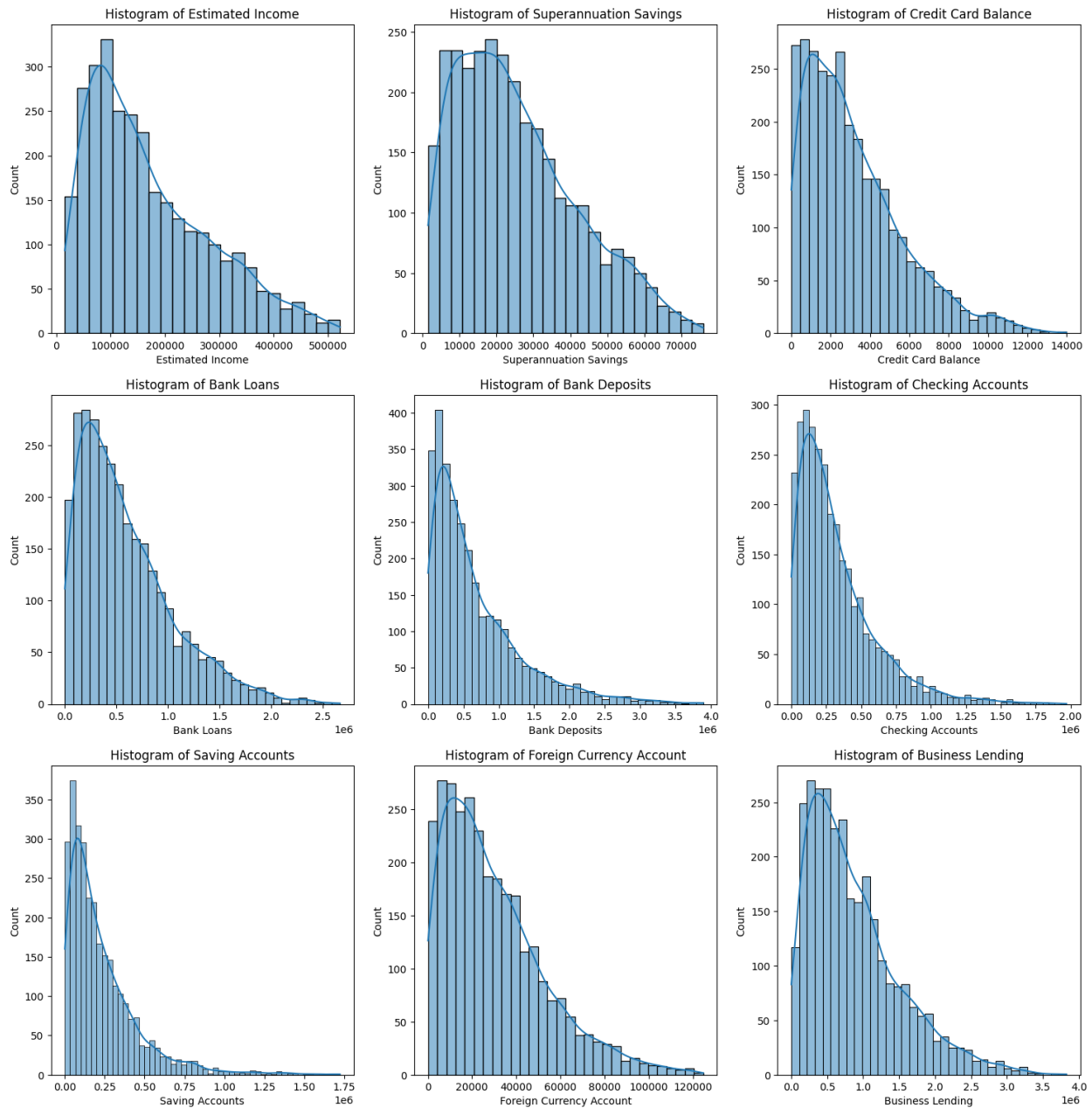


Numerical Analysis

```
numerical_col = ['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=(18,25))  
for i,col in enumerate(numerical_col):  
    plt.subplot(4,3,i+1)  
    sns.histplot(df[col],kde=True)  
    plt.title(f'Histogram of {col}')  
plt.show()
```



Heatmaps

```
numerical_col=['Estimated Income', 'Superannuation Savings','Credit Card Balance', 'Bank Loans','Bank Deposits','Checking Accounts','Saving Accounts', 'Foreign Currency Account', 'Business Lending']
```

```
correlation_matrix = df[numerical_col].corr()
```

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



Insights of EDA:

1. The strongest positive correlation occur among "Bank Deposits" with "Creating Accounts", "Saving Accounts" and "Foregin Currency Account" indicating that customers who maintain high balance in one account type often hold substantial amount/funds across other accounts as well.