Finance Project Report: Predicting Personal Loan Acceptance

1) Project Overview
2) Data Overview
3) Data Pre-processing
    3.1 Data Cleaning
    3.2 Data Transformation
    3.3 Feature Engineering
4) Correlation Analysis
5) Exploratory Data Analysis (EDA)
    4.1 Distribution of Key Variables
    4.2 Visual Insights

6) Feature Selection.

7) Data Flow

Project Overview: The objective of this project is to help Thera Bank increase the success rate of its personal loan campaigns by leveraging demographic, financial, and behavioral data of its customers. Specifically, the goal is to predict whether a customer will accept a personal loan offer based on their attributes.

Data Overview: The dataset, "Bank_Personal_Loan_Modelling.xlsx," contains data for 5000 customers, with the following key variables:

- **Demographic Information**: Age, Experience, Income, Family, Education, etc.
- **Bank Relationship**: Mortgage, Credit Card, Securities Account, CD Account, etc.
- **Target Variable**: Personal Loan (whether the customer accepted the loan or not).

Only 9.6% (480) of customers accepted the personal loan offer in the past campaign, which highlights the challenge of predicting loan acceptance.

```
In [2]: df= pd.read_excel("Bank_Personal_Loan_Modelling.xlsx", 1)
        df.head(5)
```

Out[2]:

	ID	Age	Experience	Income	ZIP Code	Family	CCAvg	Education	Mortgage	Personal Loan	Securities Account	CD Account	Online	CreditCard
0	1	25	1	49	91107	4	1.6	1	0	0	1	0	0	0
1	2	45	19	34	90089	3	1.5	1	0	0	1	0	0	0
2	3	39	15	11	94720	1	1.0	1	0	0	0	0	0	0
3	4	35	9	100	94112	1	2.7	2	0	0	0	0	0	0
4	5	35	8	45	91330	4	1.0	2	0	0	0	0	0	1

```
In [6]: df.describe()
```

Out[6]:

	ID	Age	Experience	Income	ZIP Code	Family	CCAvg	Education	Mortgage	Personal Loan	Securities Account	Ac
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.
mean	2500.500000	45.338400	20.104600	73.774200	93152.503000	2.396400	1.937913	1.881000	56.498800	0.096000	0.104400	0.
std	1443.520003	11.463166	11.467954	46.033729	2121.852197	1.147663	1.747666	0.839869	101.713802	0.294621	0.305809	0.
min	1.000000	23.000000	-3.000000	8.000000	9307.000000	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.
25%	1250.750000	35.000000	10.000000	39.000000	91911.000000	1.000000	0.700000	1.000000	0.000000	0.000000	0.000000	0.
50%	2500.500000	45.000000	20.000000	64.000000	93437.000000	2.000000	1.500000	2.000000	0.000000	0.000000	0.000000	0.
75%	3750.250000	55.000000	30.000000	98.000000	94608.000000	3.000000	2.500000	3.000000	101.000000	0.000000	0.000000	0.
max	5000.000000	67.000000	43.000000	224.000000	96651.000000	4.000000	10.000000	3.000000	635.000000	1.000000	1.000000	1.

Data Pre-processing:

1. **Data Cleaning:**
   - The columns 'ID' and 'ZIP Code' were removed as they were not relevant for predictive analysis.
   - Missing values: No missing values were found in the dataset.
   - Outliers: The 'Experience' column contained negative values, which were replaced with the mean of the column (20.10 years).

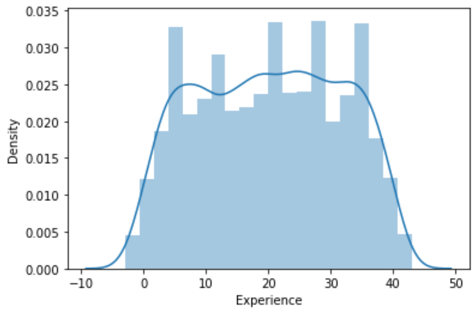
```
In [6]: df.drop(['ID', 'ZIP Code'], axis = 1, inplace = True)

In [7]: df.columns

Out[7]: Index(['Age', 'Experience', 'Income', 'Family', 'CCAvg', 'Education',
               'Mortgage', 'Personal Loan', 'Securities Account', 'CD Account',
               'Online', 'CreditCard'],
              dtype='object')
```

```
In [13]: import seaborn as sns
```

```
In [14]: sns.distplot(df['Experience'])
```

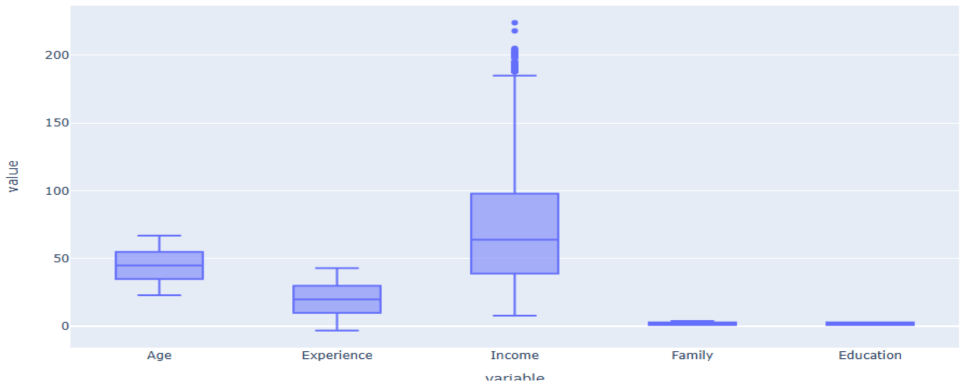
```
C:\Application\anaconda\lib\site-packages\seaborn\distributions.py:2551: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a
figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
```

```
Out[14]: <AxesSubplot:xlabel='Experience', ylabel='Density'>
```




```
In [8]: # 5 NUmber summay
        import plotly.express as ps
```

```
In [9]: fig = ps.box(df, y = ['Age', 'Experience', 'Income', 'Family', 'Education'])
        fig.show()
```

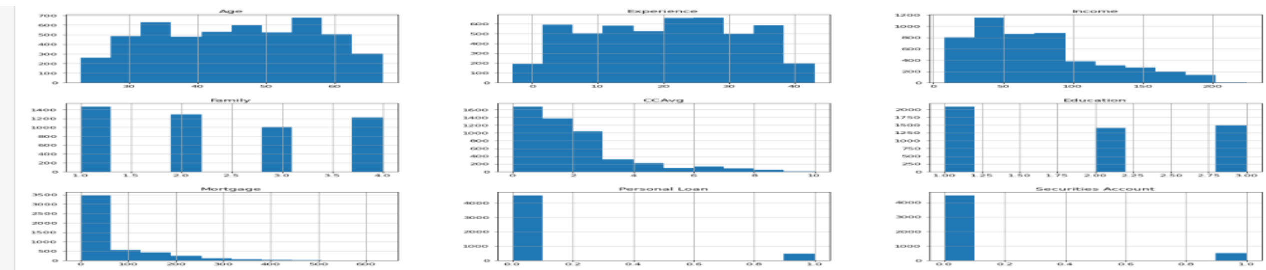


Data Transformation:

- **Log Transformation**: A log transformation was applied to 'Income' and 'CCAvg' (Credit Card Average) to reduce skewness and normalize the distribution.
- **Power Transformation**: Applied the Yeo-Johnson method to 'Income' for normalization.
- **Categorical Encoding**: The 'Education' column was converted into a categorical variable (Undergraduate, Graduate, Professional Person).

```
In [11]: df.skew()
```

```
Out[11]: Age                   -0.029341
         Experience            -0.026325
         Income                 0.841339
         Family                 0.155221
         CCAvg                  1.598457
         Education              0.227093
         Mortgage               2.104002
         Personal Loan          2.743607
         Securities Account     2.588268
         CD Account             3.691714
         Online                -0.394785
         CreditCard             0.904589
         dtype: float64
```



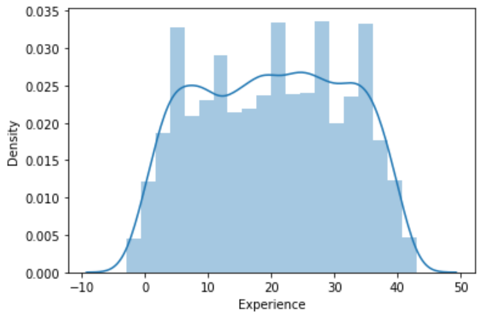
Feature Engineering:

- **Account Holder Category**: A new feature was created by combining 'Securities Account' and 'CD Account' to classify customers into four categories:
    - Holds Securities & Deposit
    - Holds only Securities Account
    - Holds only Deposit Account
    - Does not hold Securities & Deposit Account.

In [14]:
```
sns.distplot(df['Experience'])
```

```
C:\Application\anaconda\lib\site-packages\seaborn\distributions.py:2551: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a
figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
```

Out[14]: <AxesSubplot:xlabel='Experience', ylabel='Density'>



In [15]:
```
df['Experience'].mean()
```

Out[15]: 20.1046

In [16]:
```
negative_exp = df[df['Experience']<0]
negative_exp.head()
```

Out[16]:

	Age	Experience	Income	Family	CCAvg	Education	Mortgage	Personal Loan	Securities Account	CD Account	Online	CreditCard
89	25	-1	113	4	2.30	3	0	0	0	0	0	1
226	24	-1	39	2	1.70	2	0	0	0	0	0	0
315	24	-2	51	3	0.30	3	0	0	0	0	1	0
451	28	-2	48	2	1.75	3	89	0	0	0	1	0
524	24	-1	75	4	0.20	1	0	0	0	0	1	0

Correlation Analysis:

- A correlation matrix was plotted to analyze the relationships between variables. Features such as 'Income' and 'Credit Card Average' showed stronger correlations with the target variable, 'Personal Loan'.

In [26]:
```
import numpy as np
```

In [27]:
```
data['Experience'] = np.where(data['Experience'] < 0,
                              data['Experience'].mean(),
                              data['Experience'])
```

In [28]:
```
data[data['Experience']<0]
```

Out[28]:

Age	Experience	Income	Family	CCAvg	Education	Mortgage	Personal Loan	Securities Account	CD Account	Online	CreditCard

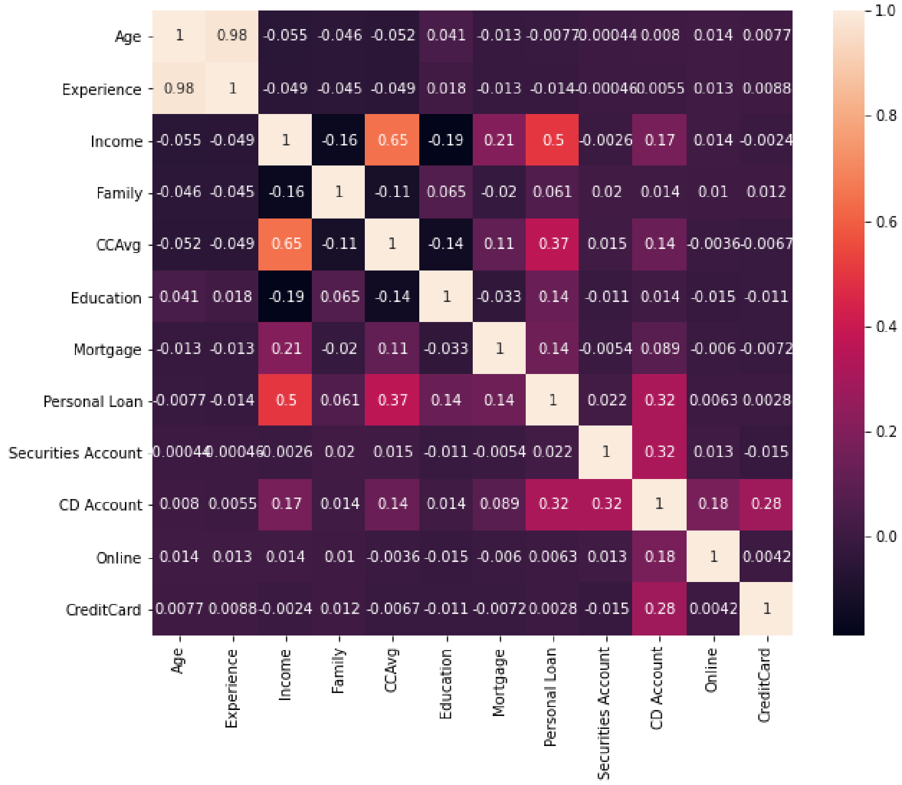
In [30]:
```
data.corr()
```

Out[30]:

	Age	Experience	Income	Family	CCAvg	Education	Mortgage	Personal Loan	Securities Account	CD Account	Online	CreditCard
Age	1.000000	0.977008	-0.055269	-0.046418	-0.052030	0.041334	-0.012539	-0.007726	-0.000436	0.008043	0.013702	0.007681
Experience	0.977008	1.000000	-0.049054	-0.045488	-0.048719	0.018097	-0.013378	-0.014045	-0.000462	0.005502	0.013455	0.008833
Income	-0.055269	-0.049054	1.000000	-0.157501	0.645993	-0.187524	0.206806	0.502462	-0.002616	0.169738	0.014206	-0.002385
Family	-0.046418	-0.045488	-0.157501	1.000000	-0.109285	0.064929	-0.020445	0.061367	0.019994	0.014110	0.010354	0.011588
CCAvg	-0.052030	-0.048719	0.645993	-0.109285	1.000000	-0.136138	0.109909	0.366891	0.015087	0.136537	-0.003620	-0.006686
Education	0.041334	0.018097	-0.187524	0.064929	-0.136138	1.000000	-0.033327	0.136722	-0.010812	0.013934	-0.015004	-0.011014
Mortgage	-0.012539	-0.013378	0.206806	-0.020445	0.109909	-0.033327	1.000000	0.142095	-0.005411	0.089311	-0.005995	-0.007231
Personal Loan	-0.007726	-0.014045	0.502462	0.061367	0.366891	0.136722	0.142095	1.000000	0.021954	0.316355	0.006278	0.002802
Securities Account	-0.000436	-0.000462	-0.002616	0.019994	0.015087	-0.010812	-0.005411	0.021954	1.000000	0.317034	0.012627	-0.015028
CD Account	0.008043	0.005502	0.169738	0.014110	0.136537	0.013934	0.089311	0.316355	0.317034	1.000000	0.175880	0.278644
Online	0.013702	0.013455	0.014206	0.010354	-0.003620	-0.015004	-0.005995	0.006278	0.012627	0.175880	1.000000	0.004210
CreditCard	0.007681	0.008833	-0.002385	0.011588	-0.006686	-0.011014	-0.007231	0.002802	-0.015028	0.278644	0.004210	1.000000

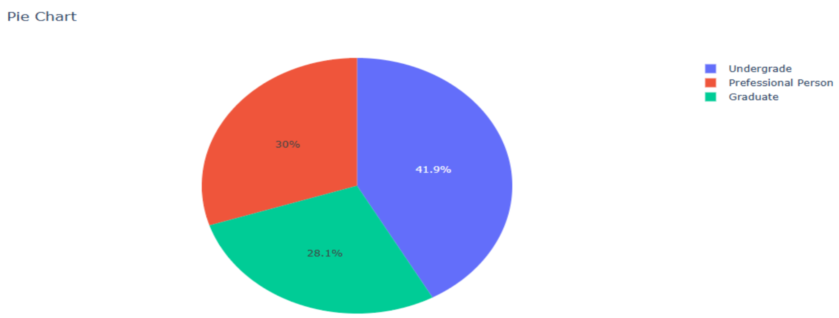
```
In [34]: plt.figure(figsize = (10,8))
         sns.heatmap(data.corr(), annot = True)
```

```
Out[34]: <AxesSubplot:>
```



Exploratory Data Analysis (EDA):

1. **Distribution of Key Variables**:
    - Age, Income, and Credit Card Average (CCAvg) showed significant variation, with skewness observed in the distribution of Income and CCAvg.
    - A comparison of the income distribution for customers who accepted and did not accept the loan revealed that customers who accepted the loan generally had higher incomes.
2. **Visual Insights**:
    - **Box Plots**: Used to identify outliers and trends across categories like Education, Mortgage, and Income.
    - **Pie Charts**: Used to show the distribution of educational backgrounds and account holder categories.
    - **Count Plots**: Illustrated the relationship between customer segments (e.g., Online Banking, Credit Card Usage) and loan acceptance.



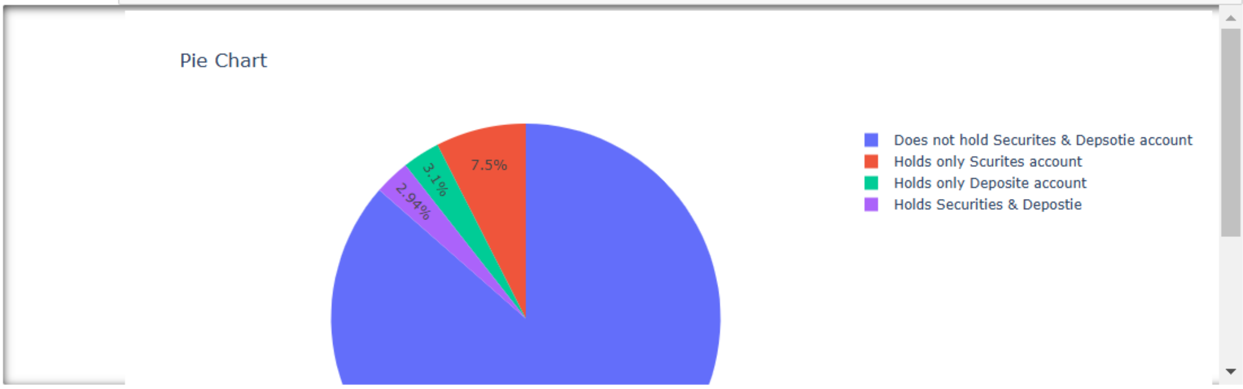
Feature Selection:

The most relevant features identified through EDA and correlation analysis include:

- **Age**
- **Income**
- **CCAvg (Credit Card Average)**
- **Education (Converted to 'EDU')**
- **Account Holder Category**
- **Mortgage**
- **Online Banking Status**

These features were selected to build predictive models.

```
In [61]: fig = ps.pie(data, values =values, names =values.index, title = 'Pie Chart')
         fig.show()
```

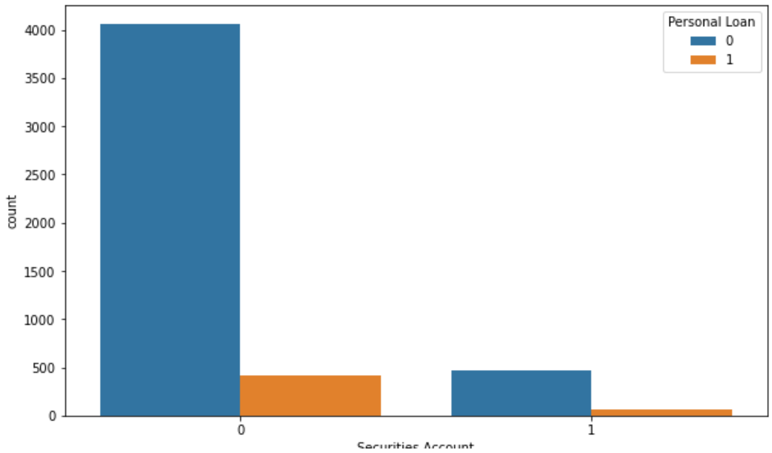



```
In [62]: data.columns
```

```
Out[62]: Index(['Age', 'Income', 'Family', 'CCAvg', 'Education', 'Mortgage',
                'Personal Loan', 'Securities Account', 'CD Account', 'Online',
                'CreditCard', 'EDU', 'Account_holder_category'],
               dtype='object')
```

```
]: col = ['Securities Account',
          'Online',
          'Account_holder_category',
          'CreditCard']
```

```
]: for i in col:
       plt.figure(figsize = (10,6))
       sns.countplot(x = i, data = data, hue = 'Personal Loan')
```



3]:
```
# Log Noranl Transform

data_1 = data[['Income', 'CCAvg']]
data_1 = np.log(data_1 + 1)
data_1
```

3]:

	Income	CCAvg
0	3.912023	0.955511
1	3.555348	0.916291
2	2.484907	0.693147
3	4.615121	1.308333
4	3.828641	0.693147
...	...	...
4995	3.713572	1.064711
4996	2.772589	0.336472
4997	3.218876	0.262364
4998	3.912023	0.405465
4999	4.430817	0.587787