PERSONAL LOAN MODELING FOR THERA BANK

GROUP 05

SANUJI DEVASURENDRA - S14810
SAHAN MADHUSHANKA - S14835
ISINI ASALYA - S15023

Abstract

For every bank with an expanding customer base, increasing its asset customers is both beneficial and challenging, as we should consider multiple factors when promoting our products, especially personal loans. Hence, the analysis of personal loan modeling is specifically carried out for Thera Bank, based in the USA, which has a growing customer base. The dataset is sourced from the Kaggle website, and both exploratory and advanced analyses were carried out to identify key factors that we should consider when promoting a personal loan to the borrowers. Among the significant factors, average monthly spending on credit cards, annual income, age, professional experience, and education level of the customer plays a pivotal role. The best model was identified as the XGBoosting classifier, and to explain the relationship of key factors with the likelihood of a customer purchasing a personal loan, partial dependency plots were utilized.

Table of Contents

List of Figures

List of Tables

1. Introduction

In the ever-competitive financial environment of today, institutions are continually striving for inventive methods to improve their services and boost revenue. A pivotal approach adopted by banks and credit institutions involves the promotion and provision of personal loans. These loans provide clients with the monetary flexibility necessary to realize their objectives, be it acquiring a vehicle, renovating their residences, or consolidating debts.

Thera Bank, based in the USA, has a growing customer base, with most of its customers being depositors as they are interested in depositing money with varying sizes of deposits. However, the bank has fewer customers who borrow money, which is an asset to the bank. According to the Department of Banking in Connecticut, USA, banks primarily make money through interest rates. Banks usually benefit by paying low interest to depositors but charging higher interest to borrowers (asset customers). Hence, Thera Bank aims to increase its borrower customer base to provide more loans and earn more through interest rates.

Therefore, Thera Bank management plans to run a campaign aimed at improving targeted marketing. The goal is to increase the success rate with a minimal budget. This decision is influenced by the positive outcome of last year's campaign, which achieved a successful conversion rate of over 9% for liability customers.

2. Description of the Question

To address the challenge of identifying the right customer who may turn into a personal loan customer, the application of classification models in predictive analytics has become increasingly important. These models enable financial institutions to assess the likelihood that a customer will buy a personal loan based on various factors, such as their financial status, demographics, and past interactions with the institution.

Therefore, our study of interest focuses on "**forecasting whether a customer is inclined to accept or decline a personal loan offer**," as it is much more beneficial to the banking organization to develop campaigns with better target marketing to increase the success rate with a minimal budget. Given the relatively small number of loan borrowers, Thera Bank's management is keen on encouraging personal loans by transitioning their liability customers into personal loan customers while retaining their depositor status. Consequently, to enhance their personal loan campaign and reduce campaign costs, the retail marketing team has set forth the following sub-objectives:

1. Identify the factors directly associated with a customer's decision to borrow a personal loan from Thera Bank.
2. Construct a predictive model capable of determining whether a customer is likely to become a loan borrower or not.

3. Description of the Dataset

The “Thera-Bank Dataset” is a Kaggle-sourced dataset that contains 5000 observations and aims to predict whether a liability customer who visits Thera Bank would borrow a personal loan or not. The dataset is a collection of 14 variables, where the response is a binary that identifies whether the customer is a personal loan borrower or not. The description of the dataset is as follows:

No.	Variable Name	Description
1	ID	Customer ID
2	Age	Customer's age in completed years
3	Experience	Number of years of professional experience
4	Income	Annual income of the customer ($000)
5	ZIPCode	Home Address ZIP code
6	Family	Family size of the customer
7	CCAvg	Avg. spending on credit cards per month ($000)
8	Education	Education Level. 1: Undergrad; 2: Graduate; 3: Advanced/Professional
9	Mortgage	Value of the house mortgage, if any. ($000)
10	Personal Loan	Did this customer accept the personal loan offered in the last campaign?
11	Securities Account	Does the customer have a securities account with the bank?
12	CD Account	Does the customer have a certificate of deposit (CD) account with the bank?
13	Online	Does the customer use internet banking facilities?
14	CreditCard	Does the customer use a credit card issued by UniversalBank?

Table 3.1 Description of Dataset

3.1. Data Pre-processing

The dataset contained no missing values or duplicate entries. Since the dataset carries both nominal and ordinal variables, a dummy encoder and label encoder were used respectively to transform variables. Afterwards, since the variable “Experience” carried negative values, they were converted to positive using the absolute conversion. Variable “ID” was removed as it has no impact on the study, and variable "ZIPCode" was removed as it carried incorrect ZIP codes in the USA. Furthermore, after performing an outlier analysis using the interquartile range technique, two observations were removed as they contained extreme values for the income.

4. Important Results from the Descriptive Analysis

As for the objective of developing an accurate model to predict the likelihood of a customer becoming a loan borrower, it is essential to understand the relationship between the response variable and the predictor variables, which is considered in this exploratory analysis.

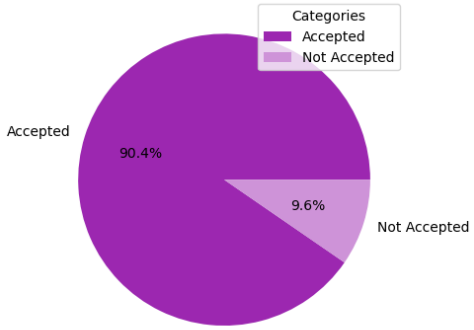

Figure 4.1: Pie Chart of Personal Loans

Hence, the binary response variable "Personal Loan" was initially focused.

Figure 4.1. reveals that after last year's campaign, only 9.6% of liability customers tend to purchase a personal loan, and still the majority of the customers have not considered applying for a loan. Hence, even though the percentage conversion of customers is fairly good, to have a better target this year, a retail marketing group has to employ changes in their strategy.

Also, since the positive class of the response is the minor class with a very low proportion (9.6%), it can be concluded that a class imbalance is present.

According to Cadence Bank, USA resources, employment history, and income play a pivotal role in pushing customers to purchase loans.

According to Figure 4.2, it can be observed that those who are interested in purchasing a loan are the customers who have a relatively high annual income when compared to those who are not interested.

The probable reason that Forbes explains is that in the USA, commercial banks consider a debt-to-income (DTI) ratio by comparing the monthly debt pay to gross monthly income.

Figure 4.2: Box Plot of Personal Loan Status by Income

Since the USA is a country with a high cost of living, the lower the salary, the higher the DTI ratio. As a result of that, customers with lower salaries might not get the opportunity to purchase loans. However, there are a certain set of high-end salary customers who are not interested in purchasing a loan, and this could be probable as they earn a fair amount.

Figure 4.3: Boxplot of Income Vs. Family Size by Personal Loan Status

As discussed above, when the number of members in a household increase, expenses increase. Then there is either a chance that people are interested in buying personal loans or disregarding loans since it might tighten their expenses further.

However, according to Figure 4.3, there is no visible variation in how customers tend to buy personal loans as the number of family members increases.

Security accounts are a type of account that holds financial assets, such as securities, on behalf of customers, such as investors, brokers, or custodians.

According to Figure 4.4, it can be observed that not only the security account holders are relatively low, but also the majority of the customers who have security accounts are not willing to purchase personal loans.

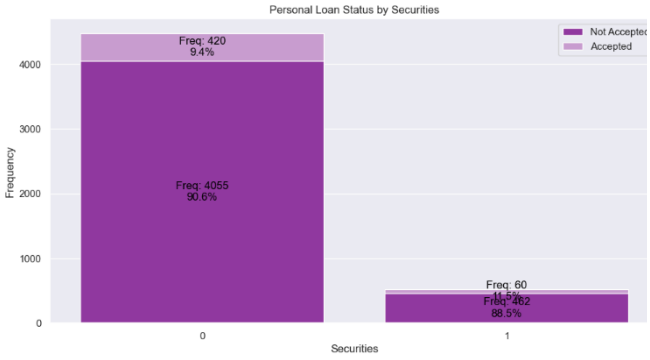

Figure 4.4: Stacked Bar Plot of Securities by Personal Loan Status

From one side, this could be probable due to the fact that the majority of the customers that visit the bank are not probably investors who maintain securities accounts, as the maximum income of the customers who visit Thera Bank is not more than $200,000. Afterwards, the majority of security account holders are not interested, as personal loans will be another liability for them.

Investopedia mentions that credit card debts are considered an asset-backed security. Hence, understanding the relationship between average spending on credit cards as a part of securities and the purchase of a personal loan could lead to meaningful insights.

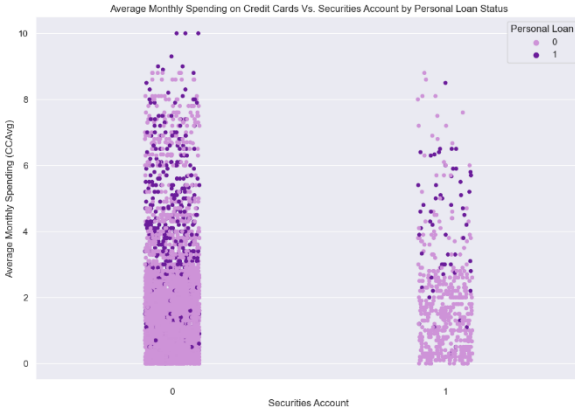

Figure 4.5: Strip Plot of AvgCC Vs. Securities Account by Personal Loan Status

Hence, it could be observed that the majority of the customers who acquired a personal loan in the last year's campaign seem to have a higher average spending on credit cards. The movement is possible as Investopedia elaborates that credit cards are quite expensive in the USA as the interest charge is quite high for them. Hence, it seems like people tend to go with relatively cheaper options like personal loans since they are charged lower rates.

Although it is considered that sometimes higher education will be an advantage for the borrower because it is thought that an educated individual will typically handle credit obligations responsibly, which translates into timely loan repayment, a higher education level is also considered to have a higher income potential.

However, pragmatically speaking, most of the time this will not be the case, and there will be no issue with your loan request being accepted as long as you are considered to be a trustworthy person who will be able to pay off the loan. According to Figure 4.6, there is no clear pattern to confirm that the loan approval rate increases with the education level.

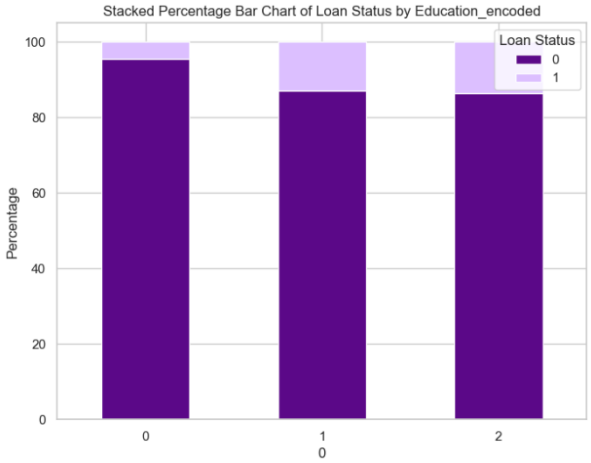

Figure 4.6: Stacked Bar Plot of Loan

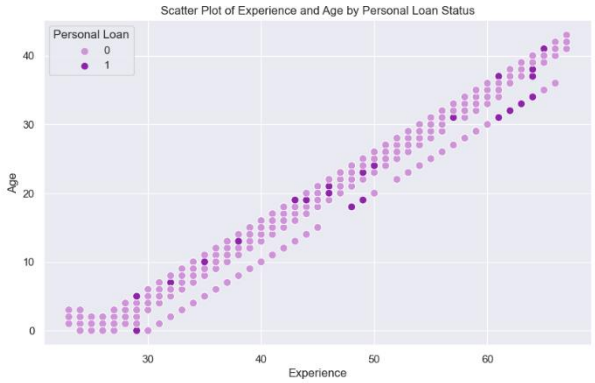

Figure 4.7: Scatter Plot of Experience and Age by Personal Loan Status

According to the Economic Times, the impact of age on the interest rate of personal loans is somewhat indirect. Meaning that the banks usually tend to offer loans to salaried applicants between 30 and 50 years old. This is mainly because of a stable source of income and professional work experience.

Hence, Figure 4.7 is able to well explain the fact that most of the personal loan borrowers tend to be older and have more professional experience as well.

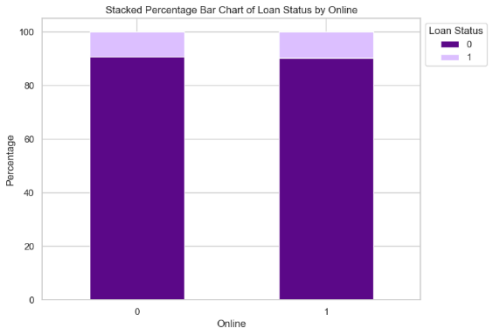

Figure 4.8: Stacked Bar Plot of Online Users by Loan Status

Figure 4.9 reveals that customers who have cash deposit accounts at Thera Bank are more likely to purchase a personal loan than

According to Figure 4.8, it could be observed that both online users and non-users have an approximately equal chance of purchasing a personal loan. Hence, the retail marketing team could provide or enhance the facilities for purchasing online loans, so customers do not need to visit the bank in person.

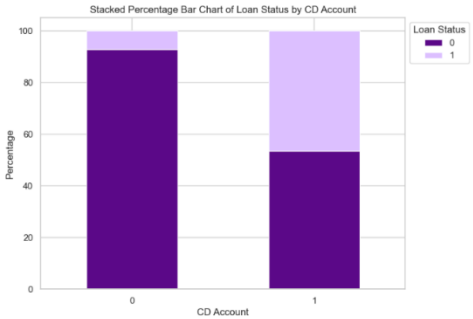

Figure 4.9: Stacked Bar Plot of CD Account by Loan Status

those who do not have a CD account with the bank. This is actually the objective of Thera Bank, where they want to convert their liability customers (depositors) into asset customers by promoting personal loans to the customers.

5. Suggestions for Advanced Analysis

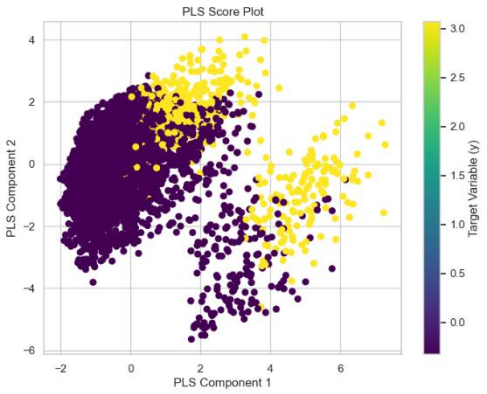

Figure 5.1: PLS Score Plot

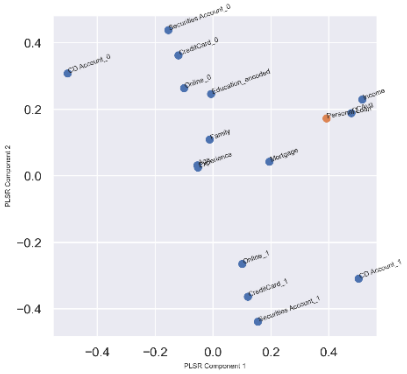

Figure 5.2: Loadings Plot of PLS

According to Figure 5.1, it can be observed that there are no well separated observational clusters, but approximately two clusters can be observed. where the loadings plot in Figure 5.2 suggests that multiple variables have a significant correlation with the purchasing status of a personal loan. However, when we consider the variation explained by the partial least squares around 0.25, since it fails to explain the variation fairly, to observe any observational clusters, an adequate clustering technique can be utilized.

According to Figures 5.3 and 5.4, there are high correlations between the predictor variables. Therefore, there is multicollinearity in the dataset, implying that the models that are robust to multicollinearity should be used in the advanced analysis of the dataset.

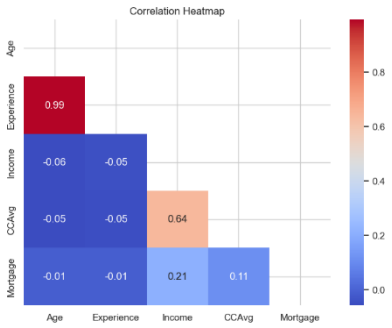

Figure 5.3: Pearson's correlation heatmap for numerical variables

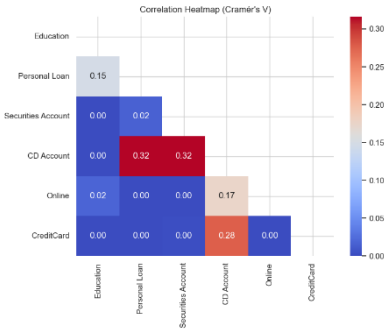

Figure 5.4: Cramer's V correlation heatmap for categorical variables

5.1. Cluster analysis

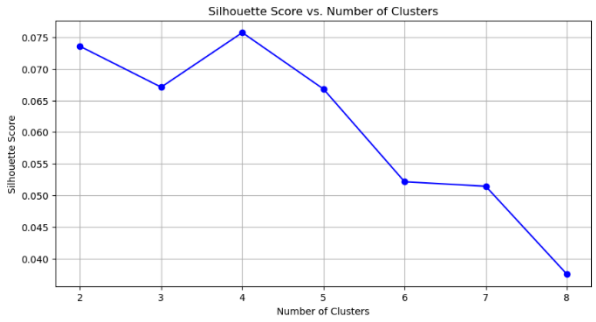

Figure 5.5: Silhouette score Vs. No. of Clusters in the cluster analysis

In the PLS Score plot in Figure 5.1, we can assume to have two approximate clusters. However, the variation explained by the PLS is quite low, to have a precise conclusion, cluster analysis was performed, and the K-prototype technique was used.

In this case it could be observed any Silhoutte score across clusters ranges from 2 to 8, not even close to 0.5, which can be considered a benchmark to perform clusters as it exceeds.

6. Important Results of the Advanced Analysis

Our study is in the interest of identifying potential customers who are likely to purchase a personal loan during the campaign that will be held this year based on past year's information. However, since Thera Bank still has a growing customer base, there are a few customers with respect to liability customers. Hence, the class imbalance is present in the dataset, and to address the problem, necessary practices should be employed.

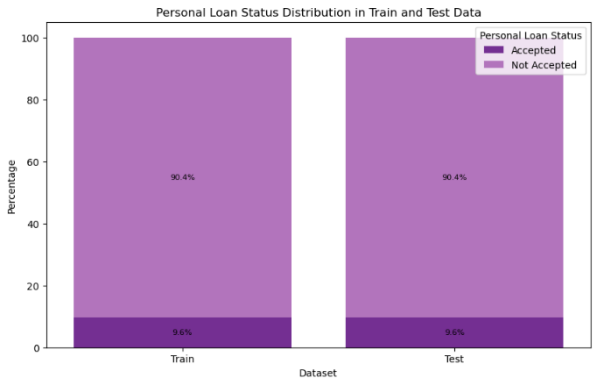

Figure 6.1: Personal Loan Distribution across Train and Test Datasets

In this case, stratified shuffle splitting was utilized to keep the percentage of each class constant in situations where some classes are underrepresented, as you can observe from Figure 6.1.

Afterwards, appropriate statistical models were utilized to develop a predictive model for this year's campaign to identify the target market.

6.1. Logistic Ridge Classifier

According to Figures 5.3 and 5.4, it could be noted that a slightly high level of multicollinearity is present in the data. Hence, to overcome the problem, one of the shrinkage techniques was utilized; Logistic Ridge Classifier.\

With no resampling technique			With SMOTE		
Accuracy	F1 Score		Accuracy	F1 Score	
	Class 0	Class 1		Class 0	Class 1
0.95	0.97	0.73	0.94	0.96	0.71

Table 6.1: Table of overall accuracy and class wise F1 Score results of Logistic Ridge Classifier

6.2. Random Forest Classifier

Random forest is an ensemble learning method that leverages the collective strength of multiple decision trees to enhance prediction accuracy and resilience. It is known for its lower likelihood of being overfitted and for being relatively easy to use and interpret. According to Table 6.1, it seems like SMOTE decreased in correctly identifying the customers who are likely to purchase a loan.

With no resampling technique			With SMOTE		
Accuracy	F1 Score		Accuracy	F1 Score	
	Class 0	Class 1		Class 0	Class 1
0.99	1.00	0.95	0.98	0.99	0.91

Table 6.2: Table of overall accuracy and class wise F1 Score results of Random Forest Classifier

6.3. Gradient Boosting Classifier

Gradient boosting is another ensemble learning method. Typically, decision trees are used by weak learners, and generally, this model is better than a random forest. In Table 6.3, we can see that oversampling caused the model's accuracy to decrease.

With no resampling technique			With SMOTE		
Accuracy	F1 Score		Accuracy	F1 Score	
	Class 0	Class 1		Class 0	Class 1
0.99	0.99	0.95	0.98	0.99	0.91

Table 6.3: Table of overall accuracy and class wise F1 Score results of Gradient Boosting Classifier

6.4. XG Boosting Classifier

XGBoost (Extreme Gradient Boosting) is a powerful machine learning algorithm renowned for its effectiveness and precision in forecasting. It is a member of the gradient boosting family that builds an ensemble of weak decision trees sequentially, optimizing for errors at each step. The XGBoost model is known for its ability to handle complex relationships in data and avoid overfitting. According to Table 6.4, the model is best fitted for the original dataset rather than the oversampled dataset.

With no resampling technique			With SMOTE		
Accuracy	F1 Score		Accuracy	F1 Score	
	Class 0	Class 1		Class 0	Class 1
0.99	1.00	0.96	0.99	0.99	0.93

Table 6.4: Table of overall accuracy and class wise F1 Score results of XGBoost Classifier

6.5. Support Vector Machine Classifier

Support vector machines are one of the most robust prediction models.Table 6.5 gives accuracy and f1 scores for support vector machines for the linear kernel. The linear kernel gives the highest accuracy compared to the Radial Basis Function (rbf), poly and sigmoid.Here also, we can see that the model for the oversampled dataset has less accuracy than the model for the original data.

With no resampling technique			With SMOTE		
Accuracy	F1 Score		Accuracy	F1 Score	
	Class 0	Class 1		Class 0	Class 1
0.95	0.98	0.74	0.93	0.96	0.69

Table 6.5: Table of overall accuracy and class wise F1 Score results of Support Vector Machine Classifier

7. Discussions and Conclusions

7.1. Train-Test Set Comparison

Model	Test Set			Train Set		
	Accuracy	F1 Score		Accuracy	F1Score	
		Class 0	Class 1		Class 0	Class 1
Logistic Ridge Classifier	0.95	0.97	0.73	0.93	0.96	0.69
Random Forest Classifier	0.99	1.00	0.95	1.00	1.00	1.00
XGBoost CLassifier	0.99	1.00	0.96	1.00	1.00	1.00

Gradient Boosting Classifier	0.99	0.99	0.95	0.99	1.00	0.96
Support Vector Machine	0.95	0.98	0.74	0.95	0.97	0.71

Table 6.6: Table of train set and test set accuracy metrics

7.2. The Best Model

According to Section 6 and Section 7.1, it could be observed that the XGBoost classifier with no resampling technique can be considered the best model as it has a higher chance of precisely identifying the customers who turned out to purchase a personal loan after the campaign held last year.

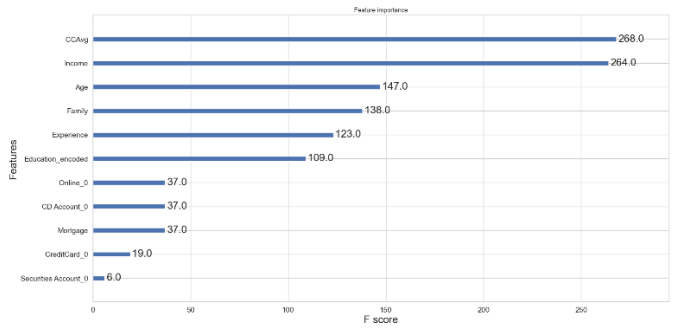

Figure 7.2.1: Variable Importance Plot of XGBoosting

Further, when we consider the feature importance plot through XGBoosting, as shown in Figure 7.2.1, it could be considered that the following factors play a pivotal role in turning a liability customer into an asset.

1. Average monthly credit card spending
2. Annual Income
3. Age
4. Family Size
5. Professional experience in years
6. Educational level

7.3. Key Considerations when promoting personal loans to Customers of Thera Bank

7.4.1 Average monthly credit card spending of the Applicant.

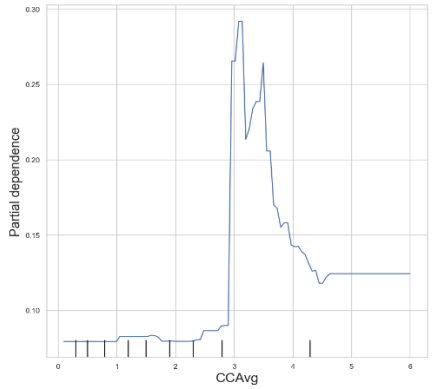

Figure 7.4.1: PD plot of Monthly Credit Card Expenditure in ($ '000')

According to Figure 7.4.1, it could be observed that for the middle range of $3000 to $5000, there is a potential chance of having a personal loan. This could be explained by the fact that there is a high interest charge on credit cards when compared to personal loans in the USA, and people with the mentioned range could have a fair chance to switch from credit cards to personal loans. However, customers with extreme expenditures might not be interested, as they can manage the cost of credit card interest.

7.4.2 Annual Income of the Applicant

It is clearly visible that customers with higher incomes have a high chance of purchasing a personal loan, which was suggested by the descriptive study as well, as they can get approval easily as they earn well. Also, from another point of view, once banks reach customers with higher incomes, they will be interested in them as they earn a higher income. According to CNBC, in the USA, earning a salary above $85,000 will allow people to live a comfortable life in highly expensive cities like San Francisco. Hence, having a salary above $120,000 will make customers interested if the campaign is well organized.

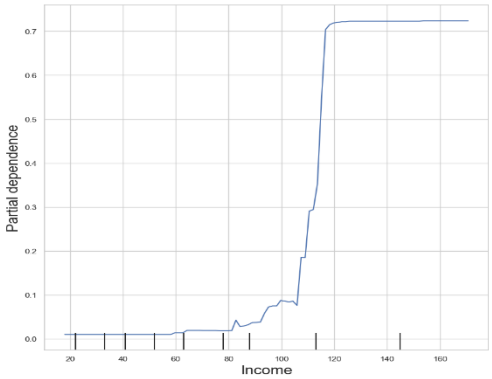

Figure 7.4.2: PD Plot of Annual Income in ($ '000')

7.4.3. Family Size of the Applicant

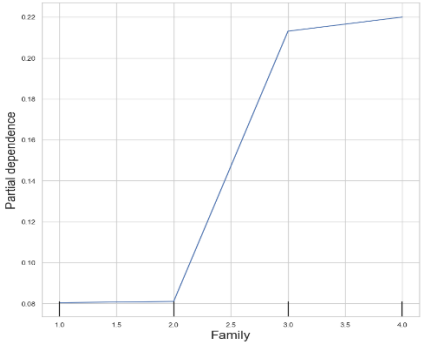

Figure 7.4.3: PD Plot for Family Size

Figure 7.4.3 well explains the fact that as the number of members in the family increases up to three and four, there is an increased chance of customers going for a personal loan. The reason is clear; as we know, when the number of household members increases, family expenditure increases. Hence, in most of such cases, customers will tend to buy a loan, at least for a temporary period of time.

8. Issues encountered and proposed solutions

- SOMTE was utilized to address the class imbalance problem. However, all the models fitted with no resampling technique provided better results than the results with SMOTE.
- Variable ZIPCode had to be removed as it contained invlaid ZIPCodes and removing them would cause a loss of information.

Next, to provide insights on what sort of liability customers Thera Bank should focus on in the campaign this year in order to convert them into assets, as follows:

- Customers with average monthly spending in the range of $3,000 to $4,000
- Customers with salaries approximately above $120,000
- Customers who belong to large families may be members of more than three.
- Customers with a good educational background

Addressing the above target market will help the retail marketing department of Thera Bank complete the campaign successfully with a minimum cost or budget.

Appendix

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import chi2_contingency
from scipy.stats import f_oneway
import numpy as np
from sklearn.cross_decomposition import PLSRegression
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import scale
from sklearn import model_selection
from sklearn.model_selection import RepeatedKFold
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
df = pd.read_csv("Bank_Personal_Loan_Modelling.csv")
df.head(5)
df.isna().sum()
df.duplicated().sum()
i='Income'
Q1 = df[i].quantile(0.25)
Q3 = df[i].quantile(0.75)
IQR = Q3 - Q1
outliers =df[(df[i] < (Q1 - 1.5 * IQR)) | (df[i] > (Q3 + 1.5 * IQR))]
print(i,'outliers Count:',outliers.shape[0])
#After analyzing for outliers, these 2 observations are deleted (In Income Var)
df.drop(labels=[3896, 4993],axis=0,inplace=True)
dummy_col = ['Securities Account',
      'CD Account', 'Online', 'CreditCard']
df2 =pd.get_dummies(data = df, columns = dummy_col)
df['Education'].unique()
df2['Experience'].unique()
df2.query('Experience < 0')
#Convert Negative Values in Experience into Positive
df2['Experience']=df2['Experience'].apply(lambda x :abs(x))
label_encoder = LabelEncoder()
df2['Education_encoded'] = label_encoder.fit_transform(df2['Education'])
df2 = df2.drop('Education', axis = 1)
df2.head(5)sns.set(style="darkgrid")
custom_colors = ["#7752FE","#B0578D"]
sns.countplot(x='Personal Loan', data=df2 , palette=custom_colors)
plt.legend(title='Categories', labels=['Not Accepted','Accepted'])
plt.show()
counts = df2['Personal Loan'].value_counts()
explode = (0.1, 0)
plt.pie(counts, labels=['Not Accepted','Accepted'], colors=custom_colors,
autopct='%1.1f%%',explode=explode, shadow=True)
plt.legend(title='Categories', labels=['Not Accepted','Accepted'])
plt.show()
df2['Personal Loan'].value_counts()
num_var = ['Age', 'Experience', 'Income', 'CCAvg', 'Mortgage']
colors = ['skyblue', 'salmon', 'green', 'orange', 'purple']
for i, color in zip(num_var, colors):
    plt.hist(df[i], bins=20, color=color, edgecolor='black')
    plt.xlabel(i)
    plt.title(f'Histogram of {i}')
    plt.show()
sns.set(style="whitegrid")
plt.figure(figsize=(8, 6))
sns.countplot(x='Family', data=df2 , palette=colors)
plt.xlabel("Family Members")
plt.ylabel("Count")
plt.title("Countplot of Family Members")
plt.show()
sns.set(style="whitegrid")
plt.figure(figsize=(10, 6))
sns.stripplot(x='Family', y='Income', hue='Personal Loan', data=df2, jitter=True, palette={1: 'green', 0: 'red'})
plt.xlabel("Family Size")
plt.ylabel("Income")
plt.title("Strip Plot of Income by Family Size with Loan Approval")
plt.legend(title='Loan Approval')
plt.show()
for i in num_var:
    plt.boxplot(df[i])
    plt.xlabel(i)
    plt.title(f'Box Plot of {i}')
    plt.show()
colors = ['#5B0888', '#D0A2F7']
for i in num_var:
    sns.boxplot(data=df, x='Personal Loan', y=i, palette = colors)
    # Add labels and title
    plt.xlabel('Loan status')
    plt.ylabel(i)
    plt.title(f'Box Plot of {i} by Loan Status')
    # Show the plot
    plt.show()
import pandas as pd
import matplotlib.pyplot as plt
binary_var = ['Securities Account', 'CD Account', 'Online', 'CreditCard']
col = ['#5B0888', '#DCBFFF']
contingency_tables = {}
for i in range(len(binary_var)):
    var1 = binary_var[i]
    # Create a contingency table using the crosstab function
    contingency_table = pd.crosstab(df[var1], df['Personal Loan'])
    # Store the contingency table in the dictionary
    contingency_tables[f'{var1}_vs_Personal Loan'] = contingency_table
    print(f'Contingency Table for {var1}_vs_Personal Loan: \n{contingency_table}\n')
    # Plotting with colors
    ax = contingency_table.plot(kind='bar', rot=0, color=col, figsize=(8, 6))
    plt.xlabel(var1)
    plt.ylabel('Count')
    plt.title(f'Grouped Bar Chart of Loan Status by {var1}')
    plt.legend(title='Loan Status')
    plt.show()
binary_var = ['Securities Account', 'CD Account', 'Online', 'CreditCard']
colors = ['#5B0888', '#DCBFFF']
contingency_tables = {}
for i in range(len(binary_var)):
    var1 = binary_var[i]
    # Create a contingency table using the crosstab function
    contingency_table = pd.crosstab(df[var1], df['Personal Loan'])
    # Store the contingency table in the dictionary
    contingency_tables[f'{var1}_vs_Personal Loan'] = contingency_table
    print(f'Contingency Table for {var1}_vs_Personal Loan: \n{contingency_table}\n')
    # Convert counts to percentages
    contingency_table_percentage = contingency_table.div(contingency_table.sum(axis=1), axis=0) * 100
    # Plot the stacked percentage bar chart
    ax = contingency_table_percentage.plot(kind='bar', rot=0, figsize=(8, 6), color=colors, stacked=True)
    plt.xlabel(var1)
    plt.ylabel('Percentage')
    plt.title(f'Stacked Percentage Bar Chart of Loan Status by {var1}')
plt.legend(title='Loan Status', bbox_to_anchor=(1, 1), loc='upper left')
    plt.show()
sns.set(style="whitegrid")
plt.figure(figsize=(8, 6))
sns.countplot(x='Education_encoded', data=df2 , palette="viridis")
plt.xlabel("Education Levels")
plt.ylabel("Count")
plt.title("Countplot of Education Levels")
# Show the plot
plt.show()
bin_var = ['Education_encoded']
colors = ['#5B0888', '#D0A2F7']
contingency_tables = {}
for i in range(len(bin_var)):
      var1 = bin_var[i]
      # Create a contingency table using the crosstab function
      contingency_table = pd.crosstab(df2[var1], df['Personal Loan'])
      contingency_tables[f'{var1}_vs_Personal Loan'] = contingency_table
      print(f'Contingency Table for {var1}_vs_Personal Loan: \n{contingency_table}\n')
      ax = contingency_table.plot(kind='bar', rot=0,
figsize=(8, 6), color=colors)
      plt.xlabel(i)
      plt.ylabel('Count')
      plt.title(f'Grouped Bar Chart of Loan Status by {var1}')
      plt.legend(title='Loan Status')
      plt.show()
df2['Age'].describe()
corr_matrix = df[num_var].corr()
# Create a heatmap
mask = np.triu(np.ones_like(corr_matrix, dtype=bool))
plt.figure(figsize=(8, 6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm',
fmt=".2f", mask=mask)
plt.title('Correlation Heatmap')
plt.show()
categorical_var = ['Education','Personal Loan', 'Securities Account',
      'CD Account', 'Online', 'CreditCard']
cat_df = df[categorical_var]
def cramers_v(x, y):
    confusion_matrix = pd.crosstab(x, y)
    chi2 = chi2_contingency(confusion_matrix)[0]
    n = confusion_matrix.sum().sum()
    phi2 = chi2 / n
    r, k = confusion_matrix.shape
    phi2corr = max(0, phi2 - ((k-1)*(r-1)) / (n-1))
    rcorr = r - ((r-1)**2) / (n-1)
    kcorr = k - ((k-1)**2) / (n-1)
    return np.sqrt(phi2corr / min((kcorr-1), (rcorr-1)))
corr_matrix = pd.DataFrame(index=cat_df.columns, columns=cat_df.columns, dtype=float)
for i in cat_df.columns:
    for j in cat_df.columns:
        corr_matrix.loc[i, j] = cramers_v(cat_df[i], cat_df[j])
# Create a heatmap
mask = np.triu(np.ones_like(corr_matrix, dtype=bool))
plt.figure(figsize=(8, 6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt=".2f" , mask=mask)
plt.title('Correlation Heatmap (Cramér\'s V)')
plt.show()
# Assumption(H0) is that FuelType and CarPrices are NOT correlated
for i in num_var:
    CategoryGroupLists=df.groupby('Personal Loan')[i].apply(list)
    AnovaResults = f_oneway(*CategoryGroupLists)
    print(f'P-Value for Anova between Loan status and {i} is: ', AnovaResults[1]
PSLR
X = df_standardized.drop('Personal Loan', axis 1)
y = df_standardized[['Personal Loan']]
y.head(5)
cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=1)
mse = []
n = len(X)
```

```
score = -1*model_selection.cross_val_score(
PLSRegression(n_components=1),
          np.ones((n,1)), y, cv=cv,
scoring='neg_mean_squared_error').mean()
mse.append(score)
for i in np.arange(1, 6):
    pls = PLSRegression(n_components=i)
    score = -
1*model_selection.cross_val_score(pls, scale(X),y
, cv=cv,
  scoring='neg_mean_squared_error').mean()
    mse.append(score)
plt.plot(mse)
plt.xlabel('Number of PLS Components')
plt.ylabel('MSE')
plt.title('hp')
X_train,X_test,y_train,y_test =
train_test_split(X,y,test_size=0.3,random_state=1)
 pls = PLSRegression(n_components=2)
pls.fit(X_train, y_train)
np.sqrt(mean_squared_error(y_test,
pls.predict(X_test)))
x=pls.x_scores_[:,0]
y=pls.x_scores_[:,1]
scores = pd. DataFrame(pls.x_scores_,
columns=['x', 'y'])
z=y_train
df_scores=pd.concat([scores.reset_index(drop=Tr
ue),
z.reset_index(drop=True)], axis=1)
sns.set(style='whitegrid')
fmri = sns.load_dataset("fmri")
chart=sns.scatterplot(x="x",
      y="y, hue="Personal
Loan",palette='plasma', data=df_scores)
chart.set_title('PLSR Scores Plot', fontdict={'size':
17, 'weight': 'bold'})
chart.set_xlabel('PLSR Component 1',
fontdict={'size': 15})
chart.set_ylabel('PLSR Component 2',
fontdict={'size': 15})
plt.savefig('scores.png',dpi=300)
plt.show()
loadings = pls.x_loadings_
loadings1= pls.y_loadings_
plt.figure(figsize=(5,5),dpi=300)
plt.rcParams.update({'font.size': 6})
plt.scatter(loadings[:, 0], loadings[:, 1])
plt.scatter(loadings1[:, 0], loadings1[:, 1])
for i, feature_name in
enumerate(X_train.columns):
    plt.annotate(feature_name, (loadings[i, 0],
loadings[i, 1]),rotation=20)
for i, feature_name in
enumerate(y_train.columns):
    plt.annotate(feature_name, (loadings1[i, 0],
loadings1[i, 1]),rotation=10)
plt.xlabel('PLSR Component 1')
plt.ylabel('PLSR Component 2')
plt.savefig('loadings.png',dpi=300,bbox_inches='ti
ght')
plt.show()
scores = pls.x_scores_[:,:2]
total_variance = np.var(X_train, axis=0).sum()
scores_variance = np.var(scores, axis=0).sum()
scores_variance
components
variance_explained = scores_variance/
total_variance
variance_explained*100
import matplotlib.pyplot as plt
import numpy as np
from sklearn.cross_decomposition import
PLSRegression
from sklearn.metrics import r2_score
def pls_explained_variance(pls, X, Y_true,
do_plot=True):
    r2 = np.zeros(pls.n_components)
    x_transformed = pls.transform(X)
for i in range(0, pls.n_components):
       Y_pred = (np.dot(x_transformed[:, i][:,
np.newaxis],
pls.y_loadings_[:, i][:, np.newaxis].T) *
pls._y_std
           + pls._y_mean)
       r2[i] = r2_score(Y_true, Y_pred)
       overall_r2 = r2_score(Y_true,
pls.predict(X))  # Use all components together
    if do_plot:
        component = np.arange(pls.n_components) + 1
        plt.plot(component, r2, '.-')
        plt.xticks(component)
        plt.xlabel('PLS Component #'), plt.ylabel('r2')
 plt.title(f'Summed individual r2: {np.sum(r2):.3f}, '
              f'Overall r2: {overall_r2:.3f}')
        plt.show()
    return r2, overall_r2
pls = PLSRegression(n_components=3).fit(X_train, y_train)
pls_explained_variance(pls, X_test, y_test)
X = df2.drop('Personal Loan', axis = 1)
y = df2[['Personal Loan']]
ADVANCED ANALYSIS
from sklearn.model_selection import StratifiedShuffleSplit
from imblearn.over_sampling import SMOTE
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from xgboost import XGBClassifier, plot_importance
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score,
classification_report, confusion_matrix, f1_score
from sklearn.inspection import PartialDependenceDisplay
#Stratified shufflesplit
n_splits = 1
stratified_splitter = StratifiedShuffleSplit(n_splits=n_splits,
test_size=0.2, random_state=42
for train_index, test_index in stratified_splitter.split(X, y):
    trainx, testx = X.iloc[train_index], X.iloc[test_index]
    trainy, testy = y.iloc[train_index], y.iloc[test_index]
counts = testy['Personal Loan'].value_counts()
custom_colors = ["#FF9999", "#66B3FF"]
plt.pie(counts, labels=['Not Accepted','Accepted'],
colors=custom_colors, autopct='%1.1f%%')
plt.legend(title='Categories', labels=['Not
Accepted','Accepted'])
plt.show()
testy['Personal Loan'].value_counts()
counts = trainy['Personal Loan'].value_counts()
custom_colors = ["#FF9999", "#66B3FF"]
plt.pie(counts, labels=['Not Accepted','Accepted'],
colors=custom_colors, autopct='%1.1f%%')
plt.legend(title='Categories', labels=['Not
Accepted','Accepted'])
plt.show()
trainy['Personal Loan'].value_counts()
oversample = SMOTE(random_state=0)
smote_x , smote_y = oversample.fit_resample(trainx , trainy)
print(smote_y.value_counts())
lridge_model = LogisticRegression(penalty='l2', C=1.0,
solver='liblinear')
lridge_model.fit(trainx, trainy)
lridge_pred = lridge_model.predict(testx)
print("Classification Report (test set):")
print(classification_report(testy, lridge_pred))
label_encoder = LabelEncoder()
testy_encoded = label_encoder.fit_transform(testy)
trainy_encoded = label_encoder.fit_transform(trainy)
xgb_model = XGBClassifier()
xgb_model.fit(trainx, trainy_encoded)
xgb_pred = xgb_model.predict(testx)
print("Classification Report:")
print(classification_report(testy_encoded, xgb_pred))
plt.figure(figsize=(20, 10))
plot_importance(xgb_model)
plt.xticks(fontsize=14)
plt.yticks(fontsize=14)
plt.show()
features =
['CCAvg','Income','Age','Family','Experience','Education_enco
ded']
features = ['CCAvg']
fig, ax = plt.subplots(figsize=(10,10))
plt.rc('font', size=20)
one=PartialDependenceDisplay.from_estimator(xgb_model,
trainx, features = features,ax=ax)
plt.savefig('0.png',dpi=300)
kernel = "linear"
y_pred = svm_classifier.predict(testx)
accuracy = accuracy_score(testy, y_pred)
conf_matrix = confusion_matrix(testy, y_pred)
classification_rep = classification_report(testy, y_pred
# Print the results
print(f'Accuracy: {accuracy}')
print(f'Confusion Matrix:\n{conf_matrix}')
print(f'Classification Report:\n{classification_rep}')
from sklearn.ensemble import
GradientBoostingClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.datasets import load_digits
from sklearn.metrics import accuracy_score,
classification_report, confusion_matrix
# Create a Gradient Boosting Classifier
gb_classifier = GradientBoostingClassifier(
random_state=42)
# Train the model
gb_classifier.fit(trainx, trainy)
# Make predictions on the test set
y_pred = gb_classifier.predict(testx)
# Evaluate the model
accuracy = accuracy_score(testy, y_pred)
conf_matrix = confusion_matrix(testy, y_pred)
classification_rep = classification_report(testy,
y_pred)
print(f'Accuracy: {accuracy}')
print(f'Confusion Matrix:\n{conf_matrix}')
print(f'Classification Report:\n{classification_rep}')
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import
RandomForestClassifier
from sklearn.metrics import accuracy_score,
classification_report, confusion_matrix
# Create a Random Forest Classifier
rf_classifier =
RandomForestClassifier(n_estimators=100,
random_state=42)
rf_classifier.fit(trainx, trainy)
y_pred = rf_classifier.predict(testx)
accuracy = accuracy_score(testy, y_pred)
conf_matrix = confusion_matrix(testy, y_pred)
classification_rep = classification_report(testy,
y_pred)
print(f'Accuracy: {accuracy}')
print(f'Confusion Matrix:\n{conf_matrix}')
print(f'Classification Report:\n{classification_rep}')
#Cluster Analysis
from kmodes.kprototypes import KPrototypes
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import silhouette_score
numerical_columns =
['Age','Experience','Income','Family','CCAvg','Mortga
ge']
categorical_columns = ['Education_encoded',
'Securities Account_0', 'Securities Account_1',
      'CD Account_0', 'CD Account_1', 'Online_0',
'Online_1', 'Credit Card_0',
      'Credit Card_1']
x_copy = trainx
scaler = StandardScaler()
x_copy[numerical_columns] =
scaler.fit_transform(x_copy[numerical_columns])
X = x_copy.values #Combining data into single
array
#Define the range for clusters
min_clusters = 2
max_clusters = 8
silhouette_scores = []
#perform K-prototype
for n_clusters in range(min_clusters, max_clusters
+ 1):
    kproto = KPrototypes(n_clusters=n_clusters,
verbose=2) #max_iter=100)
    clusters = kproto.fit_predict(X,
categorical=list(range(len(categorical_columns))))
    silhouette_avg = silhouette_score(X, clusters)
    silhouette_scores.append(silhouette_avg
    print(f"Number of Clusters: {n_clusters},
Silhouette Score: {silhouette_avg}")
plt.figure(figsize=(10, 5))
plt.plot(range(min_clusters, max_clusters + 1),
silhouette_scores, marker='o', linestyle='-', color='b')
plt.title('Silhouette Score vs. Number of Clusters')
plt.xlabel('Number of Clusters')
plt.ylabel('Silhouette Score')
plt.grid()
plt.show()
```