capstone-project-1

October 8, 2023

Capstone Project1:

Utilize machine learning approaches to predict credit card approval based on customer information.

A bank's credit card department is one of the top adopters of data science. A top focus for the bank has always been acquiring new credit card customers. Giving out credit cards without doing proper research or evaluating applicants' creditworthiness is quite risky. The credit card department has been using a data-driven system for credit assessment called Credit Scoring for many years, and the model is known as an application scorecard. A credit card application's cutoff value is determined using the application scorecard, which also aids in estimating the applicant's level of risk. This decision is made based on strategic priority at a given time.

Customers must fill out a form, either physically or online, to apply for a credit card. The application data is used to evaluate the applicant's creditworthiness. The decision is made using the application data in addition to the Credit Bureau Score, such as the FICO Score in the US or the CIBIL Score in India, and other internal information on the applicants. Additionally, the banks are rapidly taking a lot of outside data into account to enhance the caliber of credit judgements.

Features name: (Credit_Card.csv)

Ind_ID: Client ID

Gender: Gender information

Car_owner: Having car or not

Propert_owner: Having property or not

Children: Count of children

Annual_income: Annual income

Type_Income: Income type

Education: Education level

Marital_status: Marital_status

Housing_type: Living style

Birthday_count: Use backward count from current day (0), -1 means yesterday.

Employed_days: Start date of employment. Use backward count from current day (0). Positive value means, individual is currently unemployed.

Mobile_phone: Any mobile phone

Work_phone: Any work phone

Phone: Any phone number

EMAIL_ID: Any email ID

Type_Occupation: Occupation

Family_Members: Family size

Another data set (Credit_card_label.csv) contains two key pieces of information

ID: The joining key between application data and credit status data, same is Ind_ID

Label: 0 is application approved and 1 is application rejected.

Instructions Project proposal to predict credit card approval

Questions

Hypothesis

Approach

You will prepare a project proposal detailing the questions we are wanting to answer. The initial hypotheses about the data relationships and the approach you will take to get your answers.

Proposal is just a plan.

End goal is important

Section 1: Questions to Answer

What questions do you want to answer? 2-5

1Q)Why is your proposal important in today's world? How predicting a good client is worthy for a bank?

In today's dynamic financial landscape, our proposal holds immense significance. The ability to predict credit card approval outcomes has far-reaching implications for both individuals and financial institutions. Here's why our proposal is essential:

Importance in Today's World:

Data-Driven Insights: In the digital age, vast amounts of data are generated daily. Our proposal leverages advanced analytics to transform this data into actionable insights, enabling banks to make informed credit decisions rapidly.

Risk Management: With global economic uncertainty, banks face increased risk. Predicting good clients enhances risk management by identifying individuals with a higher likelihood of repaying debts, mitigating financial losses.

Operational Efficiency: Traditional credit assessment processes are time-consuming and costly. Our proposal streamlines this process, reducing the time and resources required for manual evaluations.

Fair and Inclusive Lending: Machine learning models can eliminate human bias, promoting fair lending practices and expanding access to credit for underrepresented segments of the population.

Customer Experience: Quick credit decisions lead to improved customer satisfaction. Our proposal's efficiency enables banks to offer seamless experiences, fostering long-term customer relationships.

Predicting a Good Client's Worth for a Bank:

Risk Mitigation: Banks operate on the principle of managing risk. Identifying good clients significantly reduces the probability of defaults, thereby safeguarding the bank's financial stability.

Profitability: Lending to clients who are likely to repay enhances the bank's revenue through interest income and reduces provisioning for bad debts, boosting profitability.

Capital Efficiency: Efficient risk assessment enables optimal allocation of capital, optimizing the bank's return on investment and capital utilization.

Regulatory Compliance: Regulatory authorities mandate responsible lending practices. Our proposal aligns with these requirements, ensuring the bank's compliance with financial regulations.

Long-Term Relationships: Predicting good clients helps banks cultivate lasting relationships by enabling clients to access credit, establish credit histories, and achieve financial goals.

2Q) How is it going to impact the banking sector?

The impact of our proposal on the banking sector is transformational:

Enhanced Decision-Making: Accurate credit predictions empower banks to make faster, more informed credit decisions, improving overall efficiency and customer experience.

Risk Reduction: By identifying high-risk applicants, banks can significantly reduce default rates and associated financial losses, bolstering the industry's financial stability.

Innovation: As data-driven technology becomes essential, embracing predictive analytics positions banks at the forefront of innovation and competitiveness.

Inclusive Lending: Our proposal promotes fair lending practices, contributing to financial inclusion by enabling access to credit for a wider range of individuals.

Operational Efficiency: Streamlining credit assessments minimizes manual work, leading to cost savings and resource optimization.

3Q.) If any, what is the gap in knowledge or how can your proposed method be helpful if required in the future for any bank in India?

The knowledge gap exists in the practical application of advanced analytics for credit card approval. While traditional methods exist, integrating modern machine learning techniques to predict creditworthiness is relatively unexplored. Our proposed method offers the following advantages:

Personalized Risk Assessment: Our approach adapts to the unique credit landscape in India, considering local socioeconomic factors and customer behaviors to provide tailored risk evaluations.

Real-time Insights: Our model can be updated in real-time to incorporate emerging trends, ensuring relevancy and accuracy in rapidly changing economic environments.

Adaptability: Our method is not limited to a specific bank or demographic. It can be customized and scaled to suit any bank's requirements, accommodating variations in customer behaviors and lending practices.

Long-Term Viability: The foundation of our method lies in its sustainable application. As data availability increases and machine learning matures, our proposal will remain relevant, offering a consistent advantage to banks.

In conclusion, our project proposal addresses critical questions about the significance of credit card approval prediction in today's financial landscape. The ability to predict good clients has profound implications for banks, influencing risk management, profitability, operational efficiency, and customer relationships. Moreover, our proposed method's adaptability and potential to bridge knowledge gaps position it as a valuable asset for the future of banking in India.

Section 2: Initial Hypotheses

1Q)Here you have to make some assumptions based on the questions you want to address based on the DA track or ML track.

1a)If DA track please aim to identify patterns in the data and important features that may impact a ML model.

Data Analysis (DA) Track Assumptions:

Assumption 1: There exists a correlation between applicants' credit scores and their likelihood of credit card approval. Higher credit scores are likely to lead to higher approval rates.

Assumption 2: Monthly income is positively correlated with creditworthiness. Individuals with higher incomes are more likely to be approved for credit cards.

Assumption 3: The presence of a stable employment history positively influences credit card approval. Applicants with consistent employment records are more favorable candidates.

Assumption 4: Debt-to-income ratio may play a role in credit card approval. A lower ratio suggests better financial stability and may lead to higher approval chances.

1b)If ML track please perform part 'i' as well as multiple machine learning models, perform all required steps to check if there is any assumption and justify your model. Why is your model better than any other possible model? Please justify it by relevant cost functions and if possible by any graph.

Machine Learning (ML) Track Assumptions and Hypotheses:

Assumption 1: Based on the DA track findings, we hypothesize that a machine learning model, particularly an ensemble-based model like Random Forest or Gradient Boosting, will outperform single algorithms in predicting credit card approval.

Assumption 2: Interaction terms among relevant features might enhance model performance. For instance, combining credit score and income could provide stronger predictive power.

Assumption 3: The model's performance can be optimized through feature engineering, such as transforming skewed distributions, handling missing values, and standardizing features.

Assumption 4: Hyperparameter tuning will further improve the model's predictive accuracy. Grid or random search can help identify optimal hyperparameters.

Assumption 5: The model's performance should be evaluated using appropriate evaluation metrics, such as accuracy, precision, recall, and F1-score, considering the imbalanced nature of the data.

2Q)From step 1, you may see some relationship that you want to explore and will develop a belief about data.

Exploration of Relationships:

Based on the initial analysis and hypotheses, we anticipate exploring the following relationships in the data:

Credit Score vs. Approval Probability: We expect to find a clear trend between higher credit scores and higher approval probabilities. Visualization may reveal a threshold credit score above which approval chances increase significantly.

Income vs. Approval Odds: A scatter plot of income against approval outcomes could reveal whether there's a point beyond which income no longer significantly impacts approval chances.

Employment History Impact: We will examine the distribution of employment lengths for approved and rejected applicants to determine if a stable employment history indeed influences approval rates.

Debt-to-Income Ratio: A comparison of debt-to-income ratios for approved and rejected applicants will help verify whether a lower ratio is associated with higher approval likelihood.

Feature Importance: For ML models, feature importance plots can show which variables contribute most to prediction. This can validate or challenge the initial hypotheses.

In summary, our initial hypotheses guide our data analysis and modeling efforts. We aim to identify important patterns, relationships, and features in the data, ultimately contributing to the creation of a robust predictive model for credit card approval.

Section 3: Data analysis approach

1Q)What approach are you going to take in order to prove or disprove your hypothesis?

Approach to Prove or Disprove Hypotheses:

Our approach involves a combination of Exploratory Data Analysis (EDA), feature engineering, and machine learning to address our hypotheses.

For the DA track, we will conduct thorough EDA to visualize relationships between features and the target variable (credit card approval). This will help us validate or disprove initial assumptions.

For the ML track, we will apply various machine learning algorithms, starting with ensemble models like Random Forest and Gradient Boosting. We will iteratively evaluate model performance, make necessary adjustments, and compare against baseline results.

2Q)What feature engineering techniques will be relevant to your project?

Relevant Feature Engineering Techniques:

Feature engineering plays a crucial role in enhancing model performance. Techniques relevant to our project include:

One-Hot Encoding: Converting categorical variables into binary representations to make them suitable for machine learning algorithms. Feature Scaling: Standardizing numerical features to ensure their scales do not influence model predictions disproportionately. Missing Value Imputation: Handling missing values appropriately, either through mean, median, or more advanced imputation techniques. Polynomial Features: Generating polynomial combinations of features to capture non-linear relationships. Interaction Terms: Creating interaction features to capture potential synergies among variables. Log Transform: Applying logarithmic transformations to features with skewed distributions.

3Q)Please justify your data analysis approach.

Justification of Data Analysis Approach:

Our chosen approach combines the strengths of both exploratory analysis and machine learning to provide a comprehensive understanding of credit card approval prediction.

EDA Importance: Exploratory Data Analysis allows us to visually identify patterns, correlations, and outliers in the data. It helps in confirming initial hypotheses and uncovering unexpected relationships.

Machine Learning Utilization: The ML track complements EDA by enabling a quantitative assessment of our hypotheses. Various algorithms and models will be tested to identify the best-performing one.

Iterative Process: The iterative nature of our approach ensures that we continuously refine our analysis based on new insights. We can adjust our feature engineering strategies and model selection as we gain deeper understanding.

4Q)Identify important patterns in your data using the EDA approach to justify your findings.

Identifying Important Patterns through EDA:

In our EDA approach, we'll:

Create bar plots for categorical features against approval status to identify patterns related to approval rates. Generate histograms and density plots for numerical features to explore their distributions. Develop correlation matrices and heatmaps to visualize relationships among variables and their influence on approval. As an example, we might discover from EDA that higher credit scores are indeed associated with higher approval rates. We could also find that certain income ranges have higher approval probabilities. Moreover, we'll examine whether stable employment history and lower debt-to-income ratios align with higher approval odds.

In conclusion, our data analysis approach employs a balance between exploratory analysis and machine learning. This approach not only helps us validate our initial hypotheses but also provides a solid foundation for building predictive models that can accurately predict credit card approval outcomes.

Section 4: Machine learning approach

1Q)What method will you use for machine learning based predictions for credit card approval?

We will employ various machine learning algorithms for credit card approval prediction. The initial focus will be on ensemble methods due to their ability to handle complex relationships and potential interactions among features. Specifically, we will utilize Random Forest, Gradient Boosting, and Logistic Regression as benchmarks.

2Q)Please justify the most appropriate model.

We'll start with ensemble methods like Random Forest and Gradient Boosting due to their strengths:

Random Forest: Handles feature interactions well, resistant to overfitting, provides feature importance scores for better understanding. Gradient Boosting: Builds on the weaknesses of the previous model's mistakes, gradually improving predictive performance. Given that credit card approval prediction likely involves both linear and non-linear relationships, using a combination of ensemble methods and logistic regression will provide a comprehensive view of how different algorithms handle the task. Logistic regression serves as a baseline to gauge the complexity required in the model.

3Q)Please perform necessary steps required to improve the accuracy of your model.

To improve model accuracy, we will undertake the following steps:

Feature Engineering: Implementing one-hot encoding, scaling, and transformation techniques to ensure optimal feature representation. Handling Imbalanced Data: Addressing class imbalance by using techniques such as oversampling, undersampling, or the Synthetic Minority Over-sampling Technique (SMOTE). Hyperparameter Tuning: Utilizing techniques like grid search or random search to identify optimal hyperparameters for each model. Model Stacking: Experimenting with model stacking, combining predictions from multiple models to create a more robust final prediction.

4Q)Please compare all models (at least 4 models).

We will compare the performance of at least four models:

Random Forest: Ensemble model known for its robustness and interpretability.

Gradient Boosting: Another ensemble model, suitable for capturing complex relationships.

Logistic Regression: A simple linear model that serves as a baseline for comparison.

Support Vector Machine (SVM): A non-linear classification model that can capture complex decision boundaries. Comparison will be based on:

Accuracy: Overall classification accuracy on the test dataset. Precision, Recall, F1-Score: Evaluation metrics that consider false positives and false negatives. Receiver Operating Characteristic (ROC) Curve: Visualizing the trade-off between true positive rate and false positive rate. Area Under the ROC Curve (AUC-ROC): Providing a summarized view of the model's performance across various thresholds. The goal is to select the model that strikes the best balance between precision and recall while considering computational efficiency. We will assess both the numeric scores and visualizations to make an informed decision about which model to proceed with.

In summary, our machine learning approach employs a range of algorithms, leverages feature engineering, and aims to optimize model performance. Through careful comparison, we will identify the best-performing model for predicting credit card approval.

#Importing All Necessary Libraries:

[ ]:
```
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
```

```
from sklearn.metrics import accuracy_score, confusion_matrix,␣
↪classification_report
```

#Importing Credit_card dataset:

```
[ ]: credit_details = pd.read_csv('Credit_card.csv')
credit_details.head()
```

```
[ ]:     Ind_ID GENDER Car_Owner Property_Owner  CHILDREN  Annual_income  \
0  5008827      M         Y              Y         0       180000.0
1  5009744      F         Y              N         0       315000.0
2  5009746      F         Y              N         0       315000.0
3  5009749      F         Y              N         0            NaN
4  5009752      F         Y              N         0       315000.0

            Type_Income         EDUCATION Marital_status       Housing_type  \
0             Pensioner  Higher education        Married  House / apartment
1  Commercial associate  Higher education        Married  House / apartment
2  Commercial associate  Higher education        Married  House / apartment
3  Commercial associate  Higher education        Married  House / apartment
4  Commercial associate  Higher education        Married  House / apartment

   Birthday_count  Employed_days  Mobile_phone  Work_Phone  Phone  EMAIL_ID  \
0        -18772.0         365243             1           0      0         0
1        -13557.0           -586             1           1      1         0
2             NaN           -586             1           1      1         0
3        -13557.0           -586             1           1      1         0
4        -13557.0           -586             1           1      1         0

  Type_Occupation  Family_Members
0             NaN               2
1             NaN               2
2             NaN               2
3             NaN               2
4             NaN               2
```

#Checking null values present in the Credit_card dataset columnwise:

```
[ ]: credit_details.isnull().sum()
```

```
[ ]: Ind_ID              0
GENDER              7
Car_Owner           0
Property_Owner      0
CHILDREN            0
Annual_income      23
Type_Income         0
EDUCATION           0
```

```
Marital_status       0
Housing_type         0
Birthday_count      22
Employed_days        0
Mobile_phone         0
Work_Phone           0
Phone                0
EMAIL_ID             0
Type_Occupation    488
Family_Members       0
dtype: int64
```

#Checking Non-Null Count and Datatype of each column present in the Credit_card dataset:

[ ]:
```
credit_details.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1548 entries, 0 to 1547
Data columns (total 18 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Ind_ID           1548 non-null   int64
 1   GENDER           1541 non-null   object
 2   Car_Owner        1548 non-null   object
 3   Property_Owner   1548 non-null   object
 4   CHILDREN         1548 non-null   int64
 5   Annual_income    1525 non-null   float64
 6   Type_Income      1548 non-null   object
 7   EDUCATION        1548 non-null   object
 8   Marital_status   1548 non-null   object
 9   Housing_type     1548 non-null   object
 10  Birthday_count   1526 non-null   float64
 11  Employed_days    1548 non-null   int64
 12  Mobile_phone     1548 non-null   int64
 13  Work_Phone       1548 non-null   int64
 14  Phone            1548 non-null   int64
 15  EMAIL_ID         1548 non-null   int64
 16  Type_Occupation  1060 non-null   object
 17  Family_Members   1548 non-null   int64
dtypes: float64(2), int64(8), object(8)
memory usage: 217.8+ KB
```

#Importing Credit_card_label dataset:

[ ]:
```
creditcard_label = pd.read_csv('Credit_card_label.csv')
creditcard_label
```

```
[ ]:       Ind_ID  label
     0     5008827      1
     1     5009744      1
     2     5009746      1
     3     5009749      1
     4     5009752      1
     …         …      …
     1543  5028645      0
     1544  5023655      0
     1545  5115992      0
     1546  5118219      0
     1547  5053790      0

     [1548 rows x 2 columns]
```

#Checking null values present in the creditcard_label dataset columnwise:

```
[ ]: creditcard_label.isnull().sum()
```

```
[ ]: Ind_ID    0
     label     0
     dtype: int64
```

#Checking Non-Null Count and Datatype of each column present in the creditcard_label dataset:

```
[ ]: creditcard_label.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1548 entries, 0 to 1547
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Ind_ID  1548 non-null   int64
 1   label   1548 non-null   int64
dtypes: int64(2)
memory usage: 24.3 KB
```

#Merging both 'credit_details' and 'creditcard_label' Tables into 'creditcard_complete_details' table:

```
[ ]: creditcard_complete_details = pd.merge(credit_details, creditcard_label,
      ↪on='Ind_ID', how='inner')
     creditcard_complete_details
```

```
[ ]:       Ind_ID GENDER Car_Owner Property_Owner  CHILDREN  Annual_income  \
     0     5008827      M         Y              Y         0       180000.0
     1     5009744      F         Y              N         0       315000.0
     2     5009746      F         Y              N         0       315000.0
     3     5009749      F         Y              N         0            NaN
```

4	5009752	F	Y	N	0	315000.0
…	…	…	…	…	…	…
1543	5028645	F	N	Y	0	NaN
1544	5023655	F	N	N	0	225000.0
1545	5115992	M	Y	Y	2	180000.0
1546	5118219	M	Y	N	0	270000.0
1547	5053790	F	Y	Y	0	225000.0

	Type_Income	EDUCATION \
0	Pensioner	Higher education
1	Commercial associate	Higher education
2	Commercial associate	Higher education
3	Commercial associate	Higher education
4	Commercial associate	Higher education
…	…	…
1543	Commercial associate	Higher education
1544	Commercial associate	Incomplete higher
1545	Working	Higher education
1546	Working	Secondary / secondary special
1547	Working	Higher education

	Marital_status	Housing_type	Birthday_count	Employed_days \
0	Married	House / apartment	-18772.0	365243
1	Married	House / apartment	-13557.0	-586
2	Married	House / apartment	NaN	-586
3	Married	House / apartment	-13557.0	-586
4	Married	House / apartment	-13557.0	-586
…	…	…	…	…
1543	Married	House / apartment	-11957.0	-2182
1544	Single / not married	House / apartment	-10229.0	-1209
1545	Married	House / apartment	-13174.0	-2477
1546	Civil marriage	House / apartment	-15292.0	-645
1547	Married	House / apartment	-16601.0	-2859

	Mobile_phone	Work_Phone	Phone	EMAIL_ID	Type_Occupation \
0	1	0	0	0	NaN
1	1	1	1	0	NaN
2	1	1	1	0	NaN
3	1	1	1	0	NaN
4	1	1	1	0	NaN
…	…	…	…	…	…
1543	1	0	0	0	Managers
1544	1	0	0	0	Accountants
1545	1	0	0	0	Managers
1546	1	1	1	0	Drivers
1547	1	0	0	0	NaN

```
      Family_Members  label
0                  2      1
1                  2      1
2                  2      1
3                  2      1
4                  2      1
…                …     …
1543               2      0
1544               1      0
1545               4      0
1546               2      0
1547               2      0

[1548 rows x 19 columns]
```

#Checking null values again after merging both tables:

[ ]:
```
creditcard_complete_details.isnull().sum()
```

[ ]:
```
Ind_ID               0
GENDER               7
Car_Owner            0
Property_Owner       0
CHILDREN             0
Annual_income       23
Type_Income          0
EDUCATION            0
Marital_status       0
Housing_type         0
Birthday_count      22
Employed_days        0
Mobile_phone         0
Work_Phone           0
Phone                0
EMAIL_ID             0
Type_Occupation    488
Family_Members       0
label                0
dtype: int64
```

#Checking Type of data present in each column:

[ ]:
```
creditcard_complete_details.Ind_ID.unique()
```

[ ]:
```
array([5008827, 5009744, 5009746, …, 5115992, 5118219, 5053790])
```

[ ]:
```
creditcard_complete_details.GENDER.unique()
```

[ ]: array(['M', 'F', nan], dtype=object)

[ ]:
```
creditcard_complete_details.Car_Owner.unique()
```

[ ]: array(['Y', 'N'], dtype=object)

[ ]:
```
creditcard_complete_details.Property_Owner.unique()
```

[ ]: array(['Y', 'N'], dtype=object)

[ ]:
```
creditcard_complete_details.CHILDREN.unique()
```

[ ]: array([0, 1, 2, 4, 3, 14])

[ ]:
```
creditcard_complete_details.Annual_income.unique()
```

```
[ ]: array([ 180000. ,  315000. ,      nan,  450000. ,   90000. ,  472500. ,
         270000. ,  126000. ,  202500. ,  157500. ,  112500. ,  540000. ,
         292500. ,  135000. ,   76500. ,  215100. ,  225000. ,   67500. ,
         171000. ,  103500. ,   99000. ,  391500. ,   65250. ,   72900. ,
         360000. ,  256500. ,  675000. ,  247500. ,   85500. ,  121500. ,
         130500. ,  211500. ,   81000. ,   72000. ,  148500. ,  162000. ,
         195750. ,  585000. ,  216000. ,  306000. ,  108000. ,   63000. ,
          45000. ,  337500. ,  131400. ,  117000. ,  445500. ,  234000. ,
        1575000. ,  144000. ,   67050. ,   73350. ,  193500. ,  900000. ,
          94500. ,  198000. ,   54000. ,  166500. ,  167400. ,  153000. ,
         423000. ,  243000. ,  283500. ,  252000. ,  495000. ,  612000. ,
          36000. ,  139500. ,  133650. ,  427500. ,  261000. ,  231750. ,
          90900. ,   45900. ,  119250. ,   58500. ,  328500. ,  787500. ,
         594000. ,  119700. ,   69372. ,   37800. ,  387000. ,  207000. ,
         189000. ,  333000. ,  105750. ,  382500. ,  141750. ,   40500. ,
         405000. ,   44550. ,  301500. ,  351000. ,  175500. ,  121900.5,
         238500. ,   33750. ,  116100. ,  297000. ,  630000. ,  418500. ,
          83250. ,  173250. ,  274500. ,  115200. ,   56250. ,   95850. ,
         185400. ,  810000. ,  184500. ,  165600. ,  114750. ,   47250. ,
          49500. ,   69750. ])
```

[ ]:
```
creditcard_complete_details.Type_Income.unique()
```

[ ]: array(['Pensioner', 'Commercial associate', 'Working', 'State servant'],
      dtype=object)

[ ]:
```
creditcard_complete_details.EDUCATION.unique()
```

[ ]: array(['Higher education', 'Secondary / secondary special',
       'Lower secondary', 'Incomplete higher', 'Academic degree'],
      dtype=object)

```
[ ]: creditcard_complete_details.Marital_status.unique()
```

```
[ ]: array(['Married', 'Single / not married', 'Civil marriage', 'Separated',
            'Widow'], dtype=object)
```

```
[ ]: creditcard_complete_details.Housing_type.unique()
```

```
[ ]: array(['House / apartment', 'With parents', 'Rented apartment',
            'Municipal apartment', 'Co-op apartment', 'Office apartment'],
           dtype=object)
```

```
[ ]: creditcard_complete_details.Birthday_count.unique()
```

```
[ ]: array([-18772., -13557.,     nan, …, -10229., -15292., -16601.])
```

```
[ ]: creditcard_complete_details.Employed_days.unique()
```

```
[ ]: array([365243,   -586,   -678,  -1002,   -913,   -248,  -2470,  -1644,
             -4327,  -1674,  -1086,   -925,   -854,   -185,  -3350,   -691,
             -4770,  -2394,   -384,  -3647,  -1546,   -808,  -1285,   -855,
             -7369,  -2269,  -4114,  -1161,  -7288,   -166,   -866,   -564,
             -3366,  -2150,  -1458,  -8369,   -508,  -1567,   -137,  -3974,
             -1570,  -2878,   -140,  -2576,  -1905,  -1724,  -1328,  -4987,
             -3496,  -1394,  -1265,  -2531,  -2105,   -860,   -217,   -535,
             -6866,   -584,  -2227,  -8161,   -613,   -606,  -6944,   -346,
             -1808,  -3420,   -863,  -7413,  -7553,  -3931,  -1039,   -134,
              -622,  -1595,  -1626,  -1868,   -555,  -1935,   -931,   -900,
             -4305,   -499,  -2418,   -189,  -1132,  -1770,   -919,  -1081,
              -502,  -2420,  -1325,  -6367,  -2484,   -341,  -9422,  -3054,
             -2987,  -2128,   -820,   -141,  -1692,  -4686,   -693,   -567,
              -885,  -2769,  -1547,  -3179,  -5204,  -3072,   -320,  -2469,
              -583,   -834,  -1085,  -1399,  -7310,  -1748,  -2479,   -875,
             -2805,  -2477,  -1477,  -1782,   -431,  -1416,   -263,  -2349,
              -412,  -7068,  -1787,  -1345,  -3717,  -1048,  -2667,  -2606,
             -2135,  -1534,  -2311,  -1323,  -5061,  -2213,  -2152,  -4509,
             -1552,  -1569,  -1679,  -8684,  -6337,    -97,  -1222,   -531,
             -7591,  -5639,  -1776,  -5498,  -5880,   -460,  -4532,  -3309,
             -3873,   -344,  -1923,   -604,  -1922,  -1496,   -708, -12332,
             -5209,  -6273,  -2722,  -9363,   -746,  -1322,  -3458,   -158,
             -2457,  -2811,  -7018,  -2026,   -188,  -2967,  -3166,  -5107,
             -1649,  -3694,  -3697,  -4596,  -5674,  -1682,   -196,   -530,
             -1696,  -2168, -13382,  -1509,  -1347,  -1405,   -227,  -9975,
             -1505,  -5084,  -2905,   -356,  -1719,  -3680,   -962,  -3000,
             -4275,   -110,  -2078,  -8377,  -4392,   -126,  -7640,  -3153,
              -139,  -4023,  -4888, -14887,  -3319,  -1966,   -200,  -6123,
              -117,  -2625,  -1763,  -3689,  -3414,  -2910,  -3412,  -1107,
             -8036,  -3608,   -101,  -3231,   -203,  -1589,  -3577,   -735,
```

```
  -8737,  -1617,  -1591,   -979,   -422,   -655,  -3653,  -1261,
  -7346,  -5545,  -3810,  -1190,  -3246,  -7676,  -1353,  -3715,
  -1113,   -992,  -2289, -12621,  -3643,   -869,  -2174,  -5336,
  -3535,  -4407,   -453,   -960,  -4711,  -1965,  -2543,  -1393,
  -2234,  -1631,  -1619,  -5981,  -2682,  -3443,  -2330,   -413,
  -6645,   -505,   -201,  -3273,  -1462,  -2609,   -521,  -1321,
  -2467,   -716,  -1632,   -614,  -1904,   -351,  -7260,   -626,
  -6853,   -458,  -1628,  -6094,   -734,   -657,  -2700,   -880,
   -157,   -135,  -1497,  -1266,   -515,  -1023,  -2129,   -575,
  -3262,   -430,  -2207,  -1866,  -4568,   -309,   -578,  -6908,
  -1931,  -8033,  -2257,  -3776,  -4583,  -4404,   -727,  -3805,
  -2316,  -1455,  -5422,   -581,   -893,  -1860,  -6317,  -1000,
  -4766,  -2259,  -1344,  -4690,  -8290,  -2052,  -5004,   -923,
  -1193,   -793,  -5453,    -88,  -5495,   -593,  -4816,  -1586,
  -2104,  -1032,  -9698,  -1131,  -1175,  -8375,  -2199,  -1912,
   -926,  -2499,  -2654,  -1174,   -230,  -2276,  -2197,  -2468,
 -11448,   -932,  -2956,   -619,  -3787,  -1253,  -3242,   -298,
   -124,  -4082,  -1431,  -5330,  -1017,  -1128,  -3088,  -1466,
  -1689,  -3003,  -4039,   -588,   -282,  -4491,  -1155,  -1220,
  -1203,   -712,  -2844,  -4031,  -3322,  -6693,  -2760,  -2532,
  -1646, -11542,  -5408,  -1374,  -3092,  -2993,  -2379,  -1281,
  -7804,  -3002,   -825,  -2057,  -1611,   -652,  -2537,  -2331,
  -4710,  -1191,  -1162,  -3900,   -219,   -223,  -3034,  -3329,
   -235,  -3503,  -1324,  -3112,  -2959,   -234,   -719,  -2616,
  -4410,  -1587,  -1537,  -1430,  -1928,  -2753,   -867,  -2134,
  -3232,    -95,  -3690,  -9870,  -4979,  -3396,   -461,  -5206,
  -3173,   -732,   -432,  -2102,  -6583,   -156,  -5460,  -2506,
  -3397,  -3792,  -3230,  -2464,  -3334,  -3110,  -2966,  -2728,
  -4161,  -7494,  -4280,  -5891,  -1350,   -609,  -2381,  -1613,
  -3533,  -6218,  -2874,  -1654,  -2433,  -4534,   -522,   -798,
  -2620,   -145,   -663,  -2299,  -1812,  -2589,   -130,  -5056,
   -796,  -7341,  -1130,  -5436,  -2368,  -1889,  -1315,  -1678,
  -2939,  -1616,  -8405,  -3476,  -3453,  -3769,  -1359,  -2990,
  -2321,  -4267,  -1386,   -645,   -886,  -1823,  -4965,  -3716,
  -1557,  -3761,  -1981,   -102,   -288,  -2147,  -3929,   -424,
  -1069,  -2323,  -3332,   -553,   -375,  -6352,  -1056,  -1172,
  -4663,  -5587,  -1406,  -8966,  -6860,  -1463,   -561,  -3661,
  -6764, -10600,  -3999,  -3960,  -2065,  -1205,  -1135,  -5190,
  -3720,  -2633,  -3272,  -2500,  -6033,   -170,   -278,  -3062,
  -8157,  -2653,  -1953,  -5457,  -1997,  -3435,  -1504,  -2696,
 -13735,   -132,  -6621,  -2481,  -2818, -11906,  -1077,  -9258,
  -2277,  -2422,   -826,  -2969,  -2086,  -1070,  -8172,   -701,
  -1384,  -9447,  -1160,  -1952,  -1354,   -469,   -563,  -1526,
  -2092,  -1749,   -168,  -4029,  -4967,  -3282,   -339,  -7979,
  -9683,   -577,  -2622,   -976,  -1994,  -1563,  -4523,   -340,
  -6614,  -1639,  -3392,  -3261,  -3928,  -5082,  -3630,  -3591,
  -1199,  -4304,  -2514,  -2187,  -7140,    -96,  -5176,  -6761,
```

```
       -2356,  -2908,  -3333,  -3596,  -4606,  -4030,  -5246,  -2906,
       -1915,  -1267,  -1256,  -1721,  -2317,  -1518,   -864,  -7364,
       -1101,  -2231,  -1327,   -355,  -1934,  -5179,  -2326,   -980,
       -1224,  -7048,  -2266,    -73,  -2661,   -286,  -8044,   -523,
        -978,   -795,  -2351,  -1114,  -3760,  -5145,  -4119,  -3641,
        -481,  -4603,   -888,   -393,  -1007, -10688,  -2228,  -8479,
       -2180,  -1362,  -1876,  -2265, -11389,  -4146,  -5500,  -5521,
        -658,  -3695,   -901,  -1751,  -3663,  -3149,  -1375,   -183,
        -500,   -489,  -2450,  -1539,  -1556,  -7095,   -674,  -2605,
       -1999,  -4599,  -6488,  -1054,   -745,  -3505,  -1585,  -6904,
       -2530,  -1420,   -314,  -2441, -11884,  -4520,  -3289,  -1495,
       -3504,  -3093,  -2417,  -2554,   -818,  -9359,  -1508, -10364,
       -4331,  -2296,  -2142,  -3163,  -3961,  -2896,  -1185,  -1670,
       -3670,  -1202,  -4936,   -596,  -6648,   -747,  -1741,  -1615,
       -1444,  -2089,   -397,  -2592,  -1618,  -2822,  -1600,  -2558,
        -991,  -2758,  -2988,  -2904,  -3377,  -3902,  -3202,   -671,
        -848,   -552,  -2559,  -2361,  -3854,  -1182,  -2384,  -8230,
       -2493,  -2362,  -4034,  -2143,  -2393,  -2565,   -450,  -5496,
       -2319,   -191,  -3506,  -2865,  -2339,  -2072,  -7013,  -3008,
       -2932,   -565,  -1282,  -3154,  -1225,  -2333,  -1869, -11940,
       -3515,  -3291,   -407,   -444,   -226,   -279,  -1661,   -896,
        -670,  -2533,  -7400,  -3339,  -1259,   -265,  -1870,  -2540,
       -2978,   -187,   -477,  -1436,  -1978,  -1133,  -2985, -10758,
       -4288,    -89,  -6930,  -1807,  -3143,  -3850,  -4560,   -785,
       -3570,   -777, -11451,   -742,  -5950,   -464,   -385,  -2475,
       -2051,  -3607,  -2454,  -7614,  -2796,   -642,  -4004,  -4917,
       -5622,   -313,  -1753,  -5830,  -1677,  -3078,  -3061,   -625,
       -1381,   -308,  -5437,   -218,  -3195,  -1198,  -1473,  -5483,
        -704, -12870,  -5373,  -7128,  -2255,  -5345,  -3574,   -164,
       -6226,  -6678,  -2024,   -890,  -6075,  -1774,   -686,  -3229,
       -3455,  -5454, -13010,  -5507,  -6640,  -5488,   -221,   -127,
       -8760,  -7718,  -3123,  -1652,  -6879,  -2673,   -420,  -1072,
       -5760,  -1871,  -1707,  -1962,  -1598,  -1583,  -7264,   -536,
       -1313,  -2512,  -5673,  -3380,  -3944,  -3654,   -171,  -1801,
       -1138,  -1897,  -2980,  -1581,  -6558,  -4434,   -210,  -3032,
       -2389,  -1467,   -377, -10762,  -2788,  -3668,  -2340,  -9925,
        -918,  -4219,   -194,  -3010,   -370,   -232,   -285,  -1830,
       -3853,  -2866,  -1022,  -2423,  -2545,  -6230,   -104,  -2631,
        -367,   -857,  -4887,   -433,   -803,  -3361,  -1520,  -5155,
       -5006,  -2191,  -1064,   -828,  -4874,  -1217,  -7557,  -5334,
        -611,  -5215,   -438,   -958,  -3095,  -2827,  -3693,  -2262,
        -195,   -109,  -5862,  -1200,  -1891,  -4662,  -2924,  -1648,
       -3536,  -2182,  -1209,  -2859])
```

[ ]:
```
creditcard_complete_details.Mobile_phone.unique()
```

[ ]:
```
array([1])
```

[ ]:
```
creditcard_complete_details.Work_Phone.unique()
```

[ ]:
```
array([0, 1])
```

[ ]:
```
creditcard_complete_details.Phone.unique()
```

[ ]:
```
array([0, 1])
```

[ ]:
```
creditcard_complete_details.EMAIL_ID.unique()
```

[ ]:
```
array([0, 1])
```

[ ]:
```
creditcard_complete_details.Type_Occupation.unique()
```

[ ]:
```
array([nan, 'Core staff', 'Cooking staff', 'Laborers', 'Sales staff',
       'Accountants', 'High skill tech staff', 'Managers',
       'Cleaning staff', 'Drivers', 'Low-skill Laborers', 'IT staff',
       'Waiters/barmen staff', 'Security staff', 'Medicine staff',
       'Private service staff', 'HR staff', 'Secretaries',
       'Realty agents'], dtype=object)
```

[ ]:
```
creditcard_complete_details.Family_Members.unique()
```

[ ]:
```
array([ 2,  3,  1,  4,  6,  5, 15])
```

##Checking null values again after merging both tables:

[ ]:
```
creditcard_complete_details.isnull().sum()
```

[ ]:
```
Ind_ID               0
GENDER               7
Car_Owner            0
Property_Owner       0
CHILDREN             0
Annual_income       23
Type_Income          0
EDUCATION            0
Marital_status       0
Housing_type         0
Birthday_count      22
Employed_days        0
Mobile_phone         0
Work_Phone           0
Phone                0
EMAIL_ID             0
Type_Occupation    488
Family_Members       0
```

```
label                0
dtype: int64
```

#Filling null values by using median() and mode() operation:

```
[ ]: creditcard_complete_details['GENDER'].
     ↪fillna(creditcard_complete_details['GENDER'].mode()[0], inplace=True)
```

```
[ ]: creditcard_complete_details['Annual_income'].
     ↪fillna(creditcard_complete_details['Annual_income'].median(), inplace=True)
```

```
[ ]: creditcard_complete_details['Birthday_count'].
     ↪fillna(creditcard_complete_details['Birthday_count'].mode()[0], inplace=True)
```

```
[ ]: creditcard_complete_details['Type_Occupation'].
     ↪fillna(creditcard_complete_details['Type_Occupation'].mode()[0],␣
     ↪inplace=True)
```

```
[ ]: creditcard_complete_details
```

```
[ ]:       Ind_ID GENDER Car_Owner Property_Owner  CHILDREN  Annual_income  \
     0     5008827      M         Y              Y         0       180000.0
     1     5009744      F         Y              N         0       315000.0
     2     5009746      F         Y              N         0       315000.0
     3     5009749      F         Y              N         0       166500.0
     4     5009752      F         Y              N         0       315000.0
     …         …    …       …            …        …             …
     1543  5028645      F         N              Y         0       166500.0
     1544  5023655      F         N              N         0       225000.0
     1545  5115992      M         Y              Y         2       180000.0
     1546  5118219      M         Y              N         0       270000.0
     1547  5053790      F         Y              Y         0       225000.0

                     Type_Income                      EDUCATION  \
     0                 Pensioner               Higher education
     1      Commercial associate               Higher education
     2      Commercial associate               Higher education
     3      Commercial associate               Higher education
     4      Commercial associate               Higher education
     …                       …                              …
     1543   Commercial associate               Higher education
     1544   Commercial associate              Incomplete higher
     1545                Working               Higher education
     1546                Working  Secondary / secondary special
     1547                Working               Higher education

                Marital_status       Housing_type  Birthday_count  Employed_days  \
```

```
0                  Married  House / apartment       -18772.0         365243
1                  Married  House / apartment       -13557.0           -586
2                  Married  House / apartment       -22655.0           -586
3                  Married  House / apartment       -13557.0           -586
4                  Married  House / apartment       -13557.0           -586
…                     …                …              …             …
1543               Married  House / apartment       -11957.0          -2182
1544  Single / not married  House / apartment       -10229.0          -1209
1545               Married  House / apartment       -13174.0          -2477
1546        Civil marriage  House / apartment       -15292.0           -645
1547               Married  House / apartment       -16601.0          -2859

      Mobile_phone  Work_Phone  Phone  EMAIL_ID Type_Occupation  \
0                1           0      0         0        Laborers
1                1           1      1         0        Laborers
2                1           1      1         0        Laborers
3                1           1      1         0        Laborers
4                1           1      1         0        Laborers
…            …         …    …       …              …
1543             1           0      0         0        Managers
1544             1           0      0         0     Accountants
1545             1           0      0         0        Managers
1546             1           1      1         0         Drivers
1547             1           0      0         0        Laborers

      Family_Members  label
0                  2      1
1                  2      1
2                  2      1
3                  2      1
4                  2      1
…               …     …
1543               2      0
1544               1      0
1545               4      0
1546               2      0
1547               2      0

[1548 rows x 19 columns]
```

#After performing fillna() operation again checking is there any null values present in columns:

[ ]:
```
creditcard_complete_details.isnull().sum()
```

[ ]:
```
Ind_ID            0
GENDER            0
Car_Owner         0
```

```
Property_Owner     0
CHILDREN           0
Annual_income      0
Type_Income        0
EDUCATION          0
Marital_status     0
Housing_type       0
Birthday_count     0
Employed_days      0
Mobile_phone       0
Work_Phone         0
Phone              0
EMAIL_ID           0
Type_Occupation    0
Family_Members     0
label              0
dtype: int64
```

#Selecting Indepenent variables:

```
[ ]: X =␣
      ↪creditcard_complete_details[['GENDER','Car_Owner','Property_Owner','CHILDREN','Annual_incom
                                              ␣
      ↪'Marital_status','Housing_type','Birthday_count','Employed_days','Mobile_phone','Type_Occup
     X
```

```
[ ]:      GENDER Car_Owner Property_Owner  CHILDREN  Annual_income  \
     0         M         Y              Y         0       180000.0
     1         F         Y              N         0       315000.0
     2         F         Y              N         0       315000.0
     3         F         Y              N         0       166500.0
     4         F         Y              N         0       315000.0
     …     …       …             …        …        …
     1543      F         N              Y         0       166500.0
     1544      F         N              N         0       225000.0
     1545      M         Y              Y         2       180000.0
     1546      M         Y              N         0       270000.0
     1547      F         Y              Y         0       225000.0

                    Type_Income                  EDUCATION  \
     0                Pensioner           Higher education
     1     Commercial associate           Higher education
     2     Commercial associate           Higher education
     3     Commercial associate           Higher education
     4     Commercial associate           Higher education
     …                        …                         …
     1543  Commercial associate           Higher education
```

```
1544  Commercial associate             Incomplete higher
1545               Working              Higher education
1546               Working  Secondary / secondary special
1547               Working              Higher education

            Marital_status       Housing_type  Birthday_count  Employed_days  \
0                  Married  House / apartment        -18772.0         365243
1                  Married  House / apartment        -13557.0           -586
2                  Married  House / apartment        -22655.0           -586
3                  Married  House / apartment        -13557.0           -586
4                  Married  House / apartment        -13557.0           -586
…                    …                  …              …              …
1543               Married  House / apartment        -11957.0          -2182
1544  Single / not married  House / apartment        -10229.0          -1209
1545               Married  House / apartment        -13174.0          -2477
1546        Civil marriage  House / apartment        -15292.0           -645
1547               Married  House / apartment        -16601.0          -2859

      Mobile_phone Type_Occupation  Family_Members
0                1        Laborers               2
1                1        Laborers               2
2                1        Laborers               2
3                1        Laborers               2
4                1        Laborers               2
…            …              …              …
1543             1        Managers               2
1544             1     Accountants               1
1545             1        Managers               4
1546             1         Drivers               2
1547             1        Laborers               2

[1548 rows x 14 columns]
```

#Selecting Target Variable:

[ ]:
```
y = creditcard_complete_details[['label']]
y
```

[ ]:
```
      label
0         1
1         1
2         1
3         1
4         1
…       …
1543      0
1544      0
```

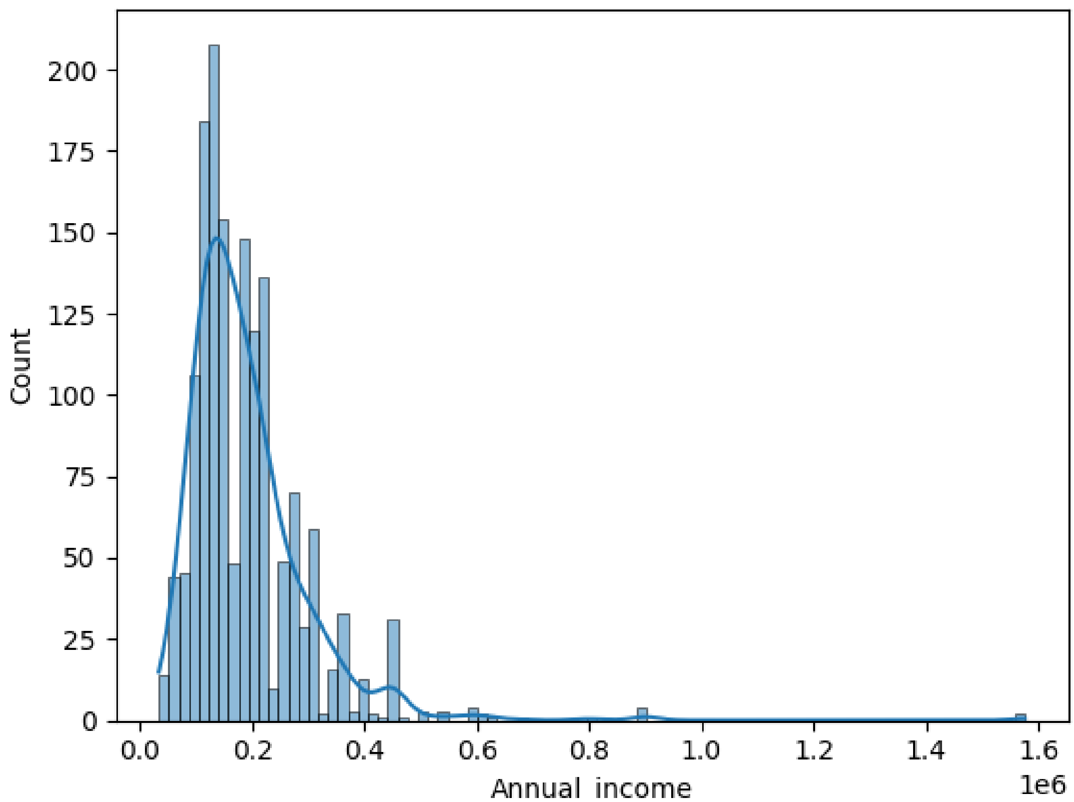
```
1545       0
1546       0
1547       0

[1548 rows x 1 columns]
```

#Checking is outliers present in Annual_income column?:

[ ]:
```
sns.histplot(x = X['Annual_income'],kde = True)
```

[ ]:
```
<Axes: xlabel='Annual_income', ylabel='Count'>
```



#performing Log Transformation for 'Annual_income' column to get the Normalization of Skewed Data:

[ ]:
```
X['Annual_income'] = np.log(X['Annual_income'])
X
```

```
<ipython-input-39-ba63689f76b6>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  X['Annual_income'] = np.log(X['Annual_income'])
```

[ ]:

```
     GENDER Car_Owner Property_Owner  CHILDREN  Annual_income  \
0         M         Y              Y         0      12.100712
1         F         Y              N         0      12.660328
2         F         Y              N         0      12.660328
3         F         Y              N         0      12.022751
4         F         Y              N         0      12.660328
…       …         …              …       …            …
1543      F         N              Y         0      12.022751
1544      F         N              N         0      12.323856
1545      M         Y              Y         2      12.100712
1546      M         Y              N         0      12.506177
1547      F         Y              Y         0      12.323856

               Type_Income                      EDUCATION  \
0                Pensioner               Higher education
1     Commercial associate               Higher education
2     Commercial associate               Higher education
3     Commercial associate               Higher education
4     Commercial associate               Higher education
…                      …                             …
1543  Commercial associate               Higher education
1544  Commercial associate              Incomplete higher
1545               Working               Higher education
1546               Working  Secondary / secondary special
1547               Working               Higher education

            Marital_status       Housing_type  Birthday_count  Employed_days  \
0                  Married  House / apartment        -18772.0         365243
1                  Married  House / apartment        -13557.0           -586
2                  Married  House / apartment        -22655.0           -586
3                  Married  House / apartment        -13557.0           -586
4                  Married  House / apartment        -13557.0           -586
…                      …                …              …             …
1543               Married  House / apartment        -11957.0          -2182
1544  Single / not married  House / apartment        -10229.0          -1209
1545               Married  House / apartment        -13174.0          -2477
1546        Civil marriage  House / apartment        -15292.0           -645
1547               Married  House / apartment        -16601.0          -2859

      Mobile_phone Type_Occupation  Family_Members
0                1        Laborers               2
1                1        Laborers               2
2                1        Laborers               2
```

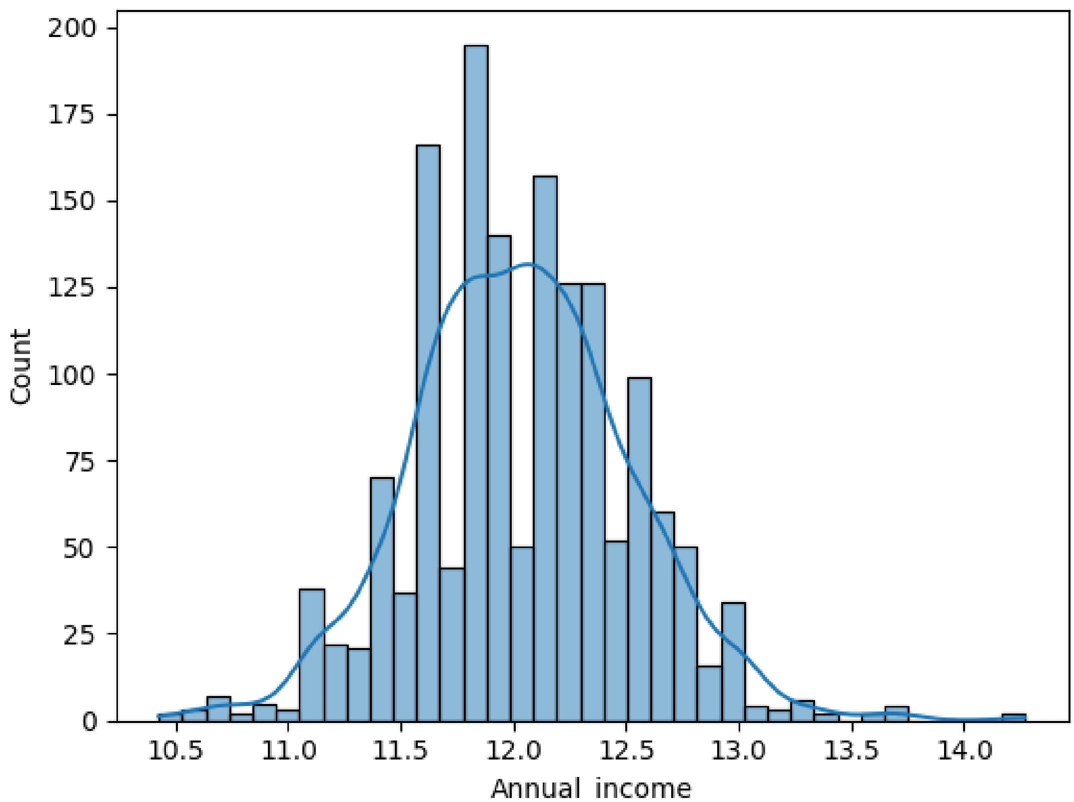
```
3                 1        Laborers               2
4                 1        Laborers               2
…             …             …             …
1543              1        Managers               2
1544              1     Accountants               1
1545              1        Managers               4
1546              1         Drivers               2
1547              1        Laborers               2

[1548 rows x 14 columns]
```

#After performing Log Transformation again checking 'Annual_income' column:

[ ]:
```
sns.histplot(x = X['Annual_income'],kde = True)
```

[ ]:
```
<Axes: xlabel='Annual_income', ylabel='Count'>
```



#Performing Dummy Encoding for Categorical data columns:

[ ]:
```
X = pd.
 ↪get_dummies(X,columns=['GENDER','Car_Owner','Property_Owner','Type_Income','EDUCATION','Mar
```

```
X
```

[ ]:

```
      CHILDREN  Annual_income  Birthday_count  Employed_days  Mobile_phone  \
0            0      12.100712        -18772.0         365243             1
1            0      12.660328        -13557.0           -586             1
2            0      12.660328        -22655.0           -586             1
3            0      12.022751        -13557.0           -586             1
4            0      12.660328        -13557.0           -586             1
…          …            …              …              …            …
1543         0      12.022751        -11957.0          -2182             1
1544         0      12.323856        -10229.0          -1209             1
1545         2      12.100712        -13174.0          -2477             1
1546         0      12.506177        -15292.0           -645             1
1547         0      12.323856        -16601.0          -2859             1

      Family_Members  GENDER_F  GENDER_M  Car_Owner_N  Car_Owner_Y  …  \
0                  2         0         1            0            1  …
1                  2         1         0            0            1  …
2                  2         1         0            0            1  …
3                  2         1         0            0            1  …
4                  2         1         0            0            1  …
…                …       …       …          …          …  …
1543               2         1         0            1            0  …
1544               1         1         0            1            0  …
1545               4         0         1            0            1  …
1546               2         0         1            0            1  …
1547               2         1         0            0            1  …

      Type_Occupation_Laborers  Type_Occupation_Low-skill Laborers  \
0                            1                                   0
1                            1                                   0
2                            1                                   0
3                            1                                   0
4                            1                                   0
…                          …                                 …
1543                         0                                   0
1544                         0                                   0
1545                         0                                   0
1546                         0                                   0
1547                         1                                   0

      Type_Occupation_Managers  Type_Occupation_Medicine staff  \
0                            0                               0
1                            0                               0
2                            0                               0
3                            0                               0
4                            0                               0
```

```
…                        …                              …
1543                     1                              0
1544                     0                              0
1545                     1                              0
1546                     0                              0
1547                     0                              0

      Type_Occupation_Private service staff  Type_Occupation_Realty agents  \
0                                         0                              0
1                                         0                              0
2                                         0                              0
3                                         0                              0
4                                         0                              0
…                                     …                            …
1543                                      0                              0
1544                                      0                              0
1545                                      0                              0
1546                                      0                              0
1547                                      0                              0

      Type_Occupation_Sales staff  Type_Occupation_Secretaries  \
0                               0                            0
1                               0                            0
2                               0                            0
3                               0                            0
4                               0                            0
…                           …                            …
1543                            0                            0
1544                            0                            0
1545                            0                            0
1546                            0                            0
1547                            0                            0

      Type_Occupation_Security staff  Type_Occupation_Waiters/barmen staff
0                                  0                                     0
1                                  0                                     0
2                                  0                                     0
3                                  0                                     0
4                                  0                                     0
…                              …                                    …
1543                               0                                     0
1544                               0                                     0
1545                               0                                     0
1546                               0                                     0
1547                               0                                     0

[1548 rows x 50 columns]
```

#Split the dataset into X_train, X_test, y_train, y_test:

```
[ ]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,␣
      ↪random_state=0)
```

```
[ ]: X_train
```

```
[ ]:       CHILDREN  Annual_income  Birthday_count  Employed_days  Mobile_phone  \
     757          3      12.218495        -13350.0          -1981             1
     29           0      13.199324        -19996.0           -691             1
     270          0      12.729321        -13188.0           -584             1
     35           0      11.630709        -20333.0          -1546             1
     1450         0      11.119883        -22981.0         365243             1
     …          …            …               …             …            …
     763          0      12.218495        -11998.0          -1497             1
     835          2      11.547327        -22655.0          -6621             1
     1216         0      11.119883         -9963.0          -1876             1
     559          0      11.967181        -20867.0          -1266             1
     684          0      13.017003        -16169.0           -855             1

           Family_Members  GENDER_F  GENDER_M  Car_Owner_N  Car_Owner_Y  …  \
     757                5         0         1            0            1  …
     29                 2         0         1            1            0  …
     270                1         0         1            0            1  …
     35                 2         0         1            0            1  …
     1450               2         0         1            0            1  …
     …                …         …       …          …          …  …
     763                1         1         0            0            1  …
     835                4         1         0            1            0  …
     1216               2         1         0            1            0  …
     559                2         0         1            1            0  …
     684                1         0         1            0            1  …

           Type_Occupation_Laborers  Type_Occupation_Low-skill Laborers  \
     757                          1                                   0
     29                           0                                   0
     270                          1                                   0
     35                           0                                   0
     1450                         1                                   0
     …                        …                                 …
     763                          0                                   0
     835                          0                                   0
     1216                         0                                   0
     559                          1                                   0
     684                          1                                   0

           Type_Occupation_Managers  Type_Occupation_Medicine staff  \
```

```
757                          0                              0
29                           1                              0
270                          0                              0
35                           0                              0
1450                         0                              0
…                          …                              …
763                          1                              0
835                          0                              1
1216                         0                              0
559                          0                              0
684                          0                              0

      Type_Occupation_Private service staff  Type_Occupation_Realty agents  \
757                                       0                              0
29                                        0                              0
270                                       0                              0
35                                        0                              0
1450                                      0                              0
…                                       …                            …
763                                       0                              0
835                                       0                              0
1216                                      0                              0
559                                       0                              0
684                                       0                              0

      Type_Occupation_Sales staff  Type_Occupation_Secretaries  \
757                             0                            0
29                              0                            0
270                             0                            0
35                              0                            0
1450                            0                            0
…                             …                          …
763                             0                            0
835                             0                            0
1216                            1                            0
559                             0                            0
684                             0                            0

      Type_Occupation_Security staff  Type_Occupation_Waiters/barmen staff
757                                0                                     0
29                                 0                                     0
270                                0                                     0
35                                 0                                     0
1450                               0                                     0
…                                …                                   …
763                                0                                     0
835                                0                                     0
```

```
1216                                   0                                       0
559                                    0                                       0
684                                    0                                       0

[1161 rows x 50 columns]
```

[ ]:
```
X_test
```

[ ]:
```
      CHILDREN  Annual_income  Birthday_count  Employed_days  Mobile_phone  \
53           1      11.813030         -9594.0           -866             1
1166         2      11.630709        -15766.0          -1086             1
1464         0      12.100712        -21554.0         365243             1
66           0      11.502875        -24611.0         365243             1
191          0      12.100712        -14756.0           -567             1
…          …              …               …             …            …
602          0      11.707670         -8947.0          -1611             1
1490         0      12.100712         -9810.0          -2993             1
482          0      11.547327        -11855.0          -4583             1
781          0      11.407565        -20125.0         365243             1
984          0      12.100712        -17386.0           -888             1

      Family_Members  GENDER_F  GENDER_M  Car_Owner_N  Car_Owner_Y  …  \
53                 3         0         1            0            1  …
1166               4         1         0            1            0  …
1464               2         1         0            1            0  …
66                 1         0         1            1            0  …
191                1         1         0            1            0  …
…                …         …         …          …          …  …
602                1         0         1            1            0  …
1490               2         0         1            0            1  …
482                2         1         0            1            0  …
781                1         1         0            1            0  …
984                2         1         0            1            0  …

      Type_Occupation_Laborers  Type_Occupation_Low-skill Laborers  \
53                           1                                   0
1166                         1                                   0
1464                         1                                   0
66                           1                                   0
191                          1                                   0
…                          …                                 …
602                          1                                   0
1490                         1                                   0
482                          1                                   0
781                          1                                   0
984                          1                                   0
```

	Type_Occupation_Managers	Type_Occupation_Medicine staff	\
53	0	0	
1166	0	0	
1464	0	0	
66	0	0	
191	0	0	
…	…	…	
602	0	0	
1490	0	0	
482	0	0	
781	0	0	
984	0	0	

	Type_Occupation_Private service staff	Type_Occupation_Realty agents	\
53	0	0	
1166	0	0	
1464	0	0	
66	0	0	
191	0	0	
…	…	…	
602	0	0	
1490	0	0	
482	0	0	
781	0	0	
984	0	0	

	Type_Occupation_Sales staff	Type_Occupation_Secretaries	\
53	0	0	
1166	0	0	
1464	0	0	
66	0	0	
191	0	0	
…	…	…	
602	0	0	
1490	0	0	
482	0	0	
781	0	0	
984	0	0	

	Type_Occupation_Security staff	Type_Occupation_Waiters/barmen staff
53	0	0
1166	0	0
1464	0	0
66	0	0
191	0	0
…	…	…
602	0	0

```
1490                                  0                                       0
482                                   0                                       0
781                                   0                                       0
984                                   0                                       0

[387 rows x 50 columns]
```

[ ]:
```
y_train
```

[ ]:
```
      label
757       0
29        1
270       0
35        1
1450      0
…     …
763       0
835       0
1216      0
559       0
684       0

[1161 rows x 1 columns]
```

[ ]:
```
y_test
```

[ ]:
```
      label
53        1
1166      0
1464      0
66        1
191       0
…     …
602       0
1490      0
482       0
781       0
984       0

[387 rows x 1 columns]
```

#Performing StandardScaling:

[ ]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
columns_to_scale =␣
 ↪['CHILDREN','Annual_income','Birthday_count','Employed_days','Family_Members']
```

```
# Use the fit_transform method to scale the selected columns
X_train[columns_to_scale] = scaler.fit_transform(X_train[columns_to_scale])
X_test[columns_to_scale] = scaler.transform(X_test[columns_to_scale])
```

[ ]:
```
X_train
```

[ ]:
```
      CHILDREN  Annual_income  Birthday_count  Employed_days  Mobile_phone  \
757   3.168546       0.422711        0.664546      -0.461176             1
29   -0.521205       2.492148       -0.889745      -0.452015             1
270  -0.521205       1.500495        0.702433      -0.451255             1
35   -0.521205      -0.817451       -0.968559      -0.458087             1
1450 -0.521205      -1.895234       -1.587843       2.146539             1
…          …            …              …             …            …
763  -0.521205       0.422711        0.980736      -0.457739             1
835   1.938629      -0.993377       -1.511602      -0.494125             1
1216 -0.521205      -1.895234        1.456659      -0.460430             1
559  -0.521205      -0.107533       -1.093445      -0.456098             1
684  -0.521205       2.107471        0.005270      -0.453180             1

      Family_Members  GENDER_F  GENDER_M  Car_Owner_N  Car_Owner_Y  …  \
757         2.891688         0         1            0            1  …
29         -0.175227         0         1            1            0  …
270        -1.197532         0         1            0            1  …
35         -0.175227         0         1            0            1  …
1450       -0.175227         0         1            0            1  …
…                …         …         …           …           …  …
763        -1.197532         1         0            0            1  …
835         1.869383         1         0            1            0  …
1216       -0.175227         1         0            1            0  …
559        -0.175227         0         1            1            0  …
684        -1.197532         0         1            0            1  …

      Type_Occupation_Laborers  Type_Occupation_Low-skill Laborers  \
757                          1                                   0
29                           0                                   0
270                          1                                   0
35                           0                                   0
1450                         1                                   0
…                          …                                   …
763                          0                                   0
835                          0                                   0
1216                         0                                   0
559                          1                                   0
684                          1                                   0

      Type_Occupation_Managers  Type_Occupation_Medicine staff  \
```

757	0	0
29	1	0
270	0	0
35	0	0
1450	0	0
…	…	…
763	1	0
835	0	1
1216	0	0
559	0	0
684	0	0

	Type_Occupation_Private service staff	Type_Occupation_Realty agents \
757	0	0
29	0	0
270	0	0
35	0	0
1450	0	0
…	…	…
763	0	0
835	0	0
1216	0	0
559	0	0
684	0	0

	Type_Occupation_Sales staff	Type_Occupation_Secretaries \
757	0	0
29	0	0
270	0	0
35	0	0
1450	0	0
…	…	…
763	0	0
835	0	0
1216	1	0
559	0	0
684	0	0

	Type_Occupation_Security staff	Type_Occupation_Waiters/barmen staff
757	0	0
29	0	0
270	0	0
35	0	0
1450	0	0
…	…	…
763	0	0
835	0	0

```
1216                                    0                                       0
559                                     0                                       0
684                                     0                                       0

[1161 rows x 50 columns]
```

[ ]:
```
X_test
```

[ ]:
```
      CHILDREN  Annual_income  Birthday_count  Employed_days  Mobile_phone  \
53    0.708712      -0.432774        1.542957      -0.453258             1
1166  1.938629      -0.817451        0.099519      -0.454820             1
1464 -0.521205       0.174203       -1.254113       2.146539             1
66   -0.521205      -1.087165       -1.969049       2.146539             1
191  -0.521205       0.174203        0.335727      -0.451135             1
…          …              …               …              …             …
602  -0.521205      -0.655072        1.694270      -0.458548             1
1490 -0.521205       0.174203        1.492441      -0.468362             1
482  -0.521205      -0.993377        1.014180      -0.479653             1
781  -0.521205      -1.288258       -0.919914       2.146539             1
984  -0.521205       0.174203       -0.279348      -0.453414             1

      Family_Members  GENDER_F  GENDER_M  Car_Owner_N  Car_Owner_Y  …  \
53          0.847078         0         1            0            1  …
1166        1.869383         1         0            1            0  …
1464       -0.175227         1         0            1            0  …
66         -1.197532         0         1            1            0  …
191        -1.197532         1         0            1            0  …
…                …       …         …           …           …  …
602        -1.197532         0         1            1            0  …
1490       -0.175227         0         1            0            1  …
482        -0.175227         1         0            1            0  …
781        -1.197532         1         0            1            0  …
984        -0.175227         1         0            1            0  …

      Type_Occupation_Laborers  Type_Occupation_Low-skill Laborers  \
53                           1                                   0
1166                         1                                   0
1464                         1                                   0
66                           1                                   0
191                          1                                   0
…                          …                                 …
602                          1                                   0
1490                         1                                   0
482                          1                                   0
781                          1                                   0
984                          1                                   0
```

```
      Type_Occupation_Managers  Type_Occupation_Medicine staff  \
53                           0                               0
1166                         0                               0
1464                         0                               0
66                           0                               0
191                          0                               0
…                          …                             …
602                          0                               0
1490                         0                               0
482                          0                               0
781                          0                               0
984                          0                               0

      Type_Occupation_Private service staff  Type_Occupation_Realty agents  \
53                                        0                              0
1166                                      0                              0
1464                                      0                              0
66                                        0                              0
191                                       0                              0
…                                       …                            …
602                                       0                              0
1490                                      0                              0
482                                       0                              0
781                                       0                              0
984                                       0                              0

      Type_Occupation_Sales staff  Type_Occupation_Secretaries  \
53                              0                            0
1166                            0                            0
1464                            0                            0
66                              0                            0
191                             0                            0
…                             …                          …
602                             0                            0
1490                            0                            0
482                             0                            0
781                             0                            0
984                             0                            0

      Type_Occupation_Security staff  Type_Occupation_Waiters/barmen staff
53                                 0                                     0
1166                               0                                     0
1464                               0                                     0
66                                 0                                     0
191                                0                                     0
…                                …                                   …
602                                0                                     0
```

```
1490                                  0                                      0
482                                   0                                      0
781                                   0                                      0
984                                   0                                      0

[387 rows x 50 columns]
```

[ ]:
```
y_train
```

[ ]:
```
      label
757       0
29        1
270       0
35        1
1450      0
…       …
763       0
835       0
1216      0
559       0
684       0

[1161 rows x 1 columns]
```

[ ]:
```
y_test
```

[ ]:
```
      label
53        1
1166      0
1464      0
66        1
191       0
…       …
602       0
1490      0
482       0
781       0
984       0

[387 rows x 1 columns]
```

#1)Model1:LogisticRegression

[ ]:
```
# Create a logistic regression model instance
model = LogisticRegression()

#Train the model using the training sets
```

```
model.fit(X_train, y_train)

#Predict the response for test dataset
y_pred = model.predict(X_test)

#accuracy_score,confusion_matrix,classification_report
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)

print("Accuracy:", accuracy)
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", class_report)
```

```
Accuracy: 0.9018087855297158
Confusion Matrix:
 [[346   0]
 [ 38   3]]
Classification Report:
               precision    recall  f1-score   support

           0       0.90      1.00      0.95       346
           1       1.00      0.07      0.14        41

    accuracy                           0.90       387
   macro avg       0.95      0.54      0.54       387
weighted avg       0.91      0.90      0.86       387


/usr/local/lib/python3.10/dist-packages/sklearn/utils/validation.py:1143:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example using
ravel().
  y = column_or_1d(y, warn=True)
```

#2)MODEL2:Support Vector Machine

- List item
- List item

[ ]:
```
#Importing SVM model
from sklearn import svm

#Creating a svm Classifier
clf = svm.SVC(kernel='linear') # Linear Kernel

#Train the model using the training sets
clf.fit(X_train, y_train)
```

```
#Predict the response for test dataset
y_pred = clf.predict(X_test)

#accuracy_score,confusion_matrix,classification_report
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)

print("Accuracy:", accuracy)
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", class_report)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/utils/validation.py:1143:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example using
ravel().
  y = column_or_1d(y, warn=True)
```

```
Accuracy: 0.8992248062015504
Confusion Matrix:
 [[345   1]
 [ 38   3]]
Classification Report:
               precision    recall  f1-score   support

           0       0.90      1.00      0.95       346
           1       0.75      0.07      0.13        41

    accuracy                           0.90       387
   macro avg       0.83      0.54      0.54       387
weighted avg       0.88      0.90      0.86       387
```

#3)MODEL3:RandomForest

[ ]:
```
#Fitting Decision Tree classifier to the training set
from sklearn.ensemble import RandomForestClassifier
classifier= RandomForestClassifier(n_estimators= 9, criterion="entropy")

#Train the model using the training sets
classifier.fit(X_train, y_train)

#Predicting the test set result
y_pred= classifier.predict(X_test)

#accuracy_score,confusion_matrix,classification_report
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
```

```
class_report = classification_report(y_test, y_pred)

print("Accuracy:", accuracy)
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", class_report)
```

```
Accuracy: 0.917312661498708
Confusion Matrix:
 [[340   6]
 [ 26  15]]
Classification Report:
               precision    recall  f1-score   support

           0       0.93      0.98      0.96       346
           1       0.71      0.37      0.48        41

    accuracy                           0.92       387
   macro avg       0.82      0.67      0.72       387
weighted avg       0.91      0.92      0.91       387


<ipython-input-54-26c6efd8267f>:6: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().
  classifier.fit(X_train, y_train)
```

#4)MODEL4:GRADIANT BOOSTING

[ ]:
```
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# Create a Gradient Boosting Classifier instance with 100 trees and a fixed
 ↪random seed
model = GradientBoostingClassifier(n_estimators=100, random_state=42)

#Train the model using the training sets
model.fit(X_train, y_train)

#Predicting the test set result
y_pred = model.predict(X_test)

#accuracy_score,confusion_matrix,classification_report
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)

print("Accuracy:", accuracy)
```

```
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", class_report)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/ensemble/_gb.py:437:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example using
ravel().
  y = column_or_1d(y, warn=True)
```

```
Accuracy: 0.9018087855297158
Confusion Matrix:
 [[342   4]
 [ 34   7]]
Classification Report:
               precision    recall  f1-score   support

           0       0.91      0.99      0.95       346
           1       0.64      0.17      0.27        41

    accuracy                           0.90       387
   macro avg       0.77      0.58      0.61       387
weighted avg       0.88      0.90      0.88       387
```

#download the cleaned 'creditcard_complete_details' file for performing SQL operation:

[ ]:
```
creditcard_complete_details.to_csv('creditcard_complete_details.csv',␣
 ↪index=False)
# importing file from a local folder
from google.colab import files
files.download('creditcard_complete_details.csv')
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.Javascript object>
```