Predicting Loan Default Risk

Data Analytics Lab – G1

Project Group 16

Team Members

MAYUR MILIND KAMATKAR

ZHANG XINGJIAN

KRUTI CHANDRASHEKAR

Project description

Super Lender

Leading the charge in the digital lending revolution, Super Lender has carved out a special place for itself by skillfully evaluating borrower repayment capacity using advanced credit risk models. To differentiate between clients who are creditworthy and those who might default, this strategy is essential. With the goal of improving loan repayment projections and streamlining its lending process, the organization is now concentrating on utilizing data science and machine learning. Super Lender aims to reduce financial risks and increase company efficiency and profitability in the fiercely competitive digital lending sector by using this innovation.

Need for Enhanced Repayment Prediction

Super Lender is enhancing its loan repayment assessments by embracing data science and machine learning, outpacing traditional models based on manual analysis. This shift towards sophisticated risk assessment models promises greater precision in predicting loan defaults, enabling the company to fine-tune its lending practices and financial offerings. The goal is to bolster Super Lender's credit risk system, reducing financial risks and driving profitability in the competitive arena of digital lending.

Objectives

The project's decision to concentrate on the Loan Default Prediction Challenge shows a strong interest in applying data science to solve actual financial issues. The understanding of the crucial role predictive analytics plays in the financial industry, especially in digital lending, is what drove this choice. The intricacies and subtleties of loan default prediction provide the team with a singular chance to explore an area where data-driven insights may profoundly impact company results. The effort is in line with a new trend in the financial sector, where more sophisticated, data-centric approaches are quickly replacing old ways. In today's tech-driven financial market, this trend towards using sophisticated analytical tools to anticipate loan payback probability is not only very important but also intellectually engaging.

The project is also motivated by its potential applications and business value to Super Lender, a digital lending company. The prospect of using precise loan default predictions to support better customer relationship strategies, increased profitability, and improved risk management excites your team. Leveraging extensive datasets encompassing past and current loans, along with demographic information, this project is set to sharpen data analysis skills through both descriptive and predictive analytics. The objective is to craft a refined predictive model that assesses credit risk and categorizes loans into 'Good' for likely repayment and 'Bad' for high default risk, thereby enhancing financial decision-making with data-driven insights.

Provide Super Lender with data-driven insights so that they may minimize bad loans and maximize revenue while making lending decisions.

Depending on the predictive risk model, Super Lender may choose whether or not to lend to a potential borrower as well as the amount, interest rate, and length of the loan offer.

Dataset Description

Three essential datasets are being used in our project, which will help us create a reliable predictive model for estimating the probability of loan default. The project utilizes three key datasets: 'Demographic Data' detailing consumer profiles, 'Loan Performance Data' with current loan specifics, and 'Previous Loans Data' for historical loan behaviors, offering a comprehensive view of customers' socioeconomic backgrounds, recent loan activities, and past repayment patterns.

Our objective is to create a prediction model that can accurately identify between "Good" loan outcomes where the debt is likely to be repaid and "Bad" loan outcomes where there is a chance that the loan may default using these datasets."

Analytic Approach

Data Preprocessing and Integration:

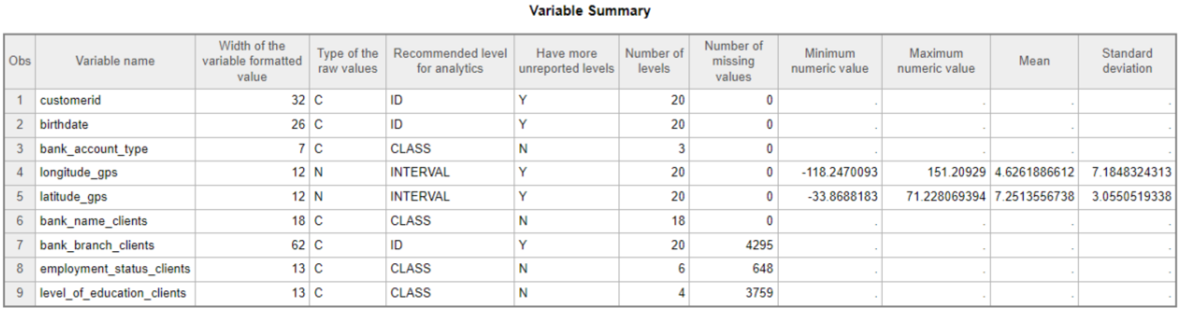
Let's begin by taking a closer look at these datasets to see what kind of information they include. This will support the planning of the procedures for data preparation and analysis. We'll examine each dataset separately:

trainperf.csv: This file probably includes historical loan performance information.

trainprevloans.csv: This file includes information on prior loans that the clients have taken out.

The file traindemographics.csv is anticipated to include consumer demographic data.

Summary statistics of the datasets



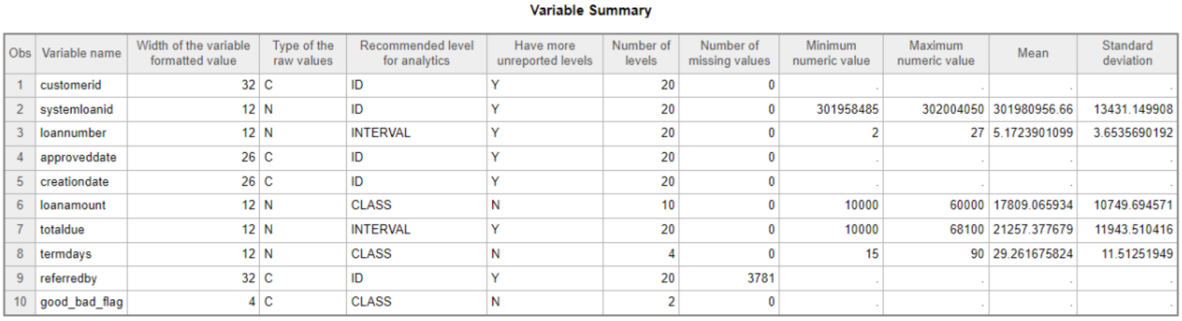
Variable Summary

Obs	Variable name	Width of the variable formatted value	Type of the raw values	Recommended level for analytics	Have more unreported levels	Number of levels	Number of missing values	Minimum numeric value	Maximum numeric value	Mean	Standard deviation
1	customerid	32	C	ID	Y	20	0	.	.	.	.
2	birthdate	26	C	ID	Y	20	0	.	.	.	.
3	bank_account_type	7	C	CLASS	N	3	0	.	.	.	.
4	longitude_gps	12	N	INTERVAL	Y	20	0	-118.2470093	151.20929	4.6261886612	7.1848324313
5	latitude_gps	12	N	INTERVAL	Y	20	0	-33.8688183	71.228069394	7.2513556738	3.0550519338
6	bank_name_clients	18	C	CLASS	N	18	0	.	.	.	.
7	bank_branch_clients	62	C	ID	Y	20	4295	.	.	.	.
8	employment_status_clients	13	C	CLASS	N	6	648	.	.	.	.
9	level_of_education_clients	13	C	CLASS	N	4	3759	.	.	.	.

Figure 1: Summary Statistics of Demographics table

Missing Values: The bank branch clients (4,295 missing) and level_of_education_clients (3,759 missing) variables have a large number of missing values, indicating that numerous customers did not provide this information.

Perf train table



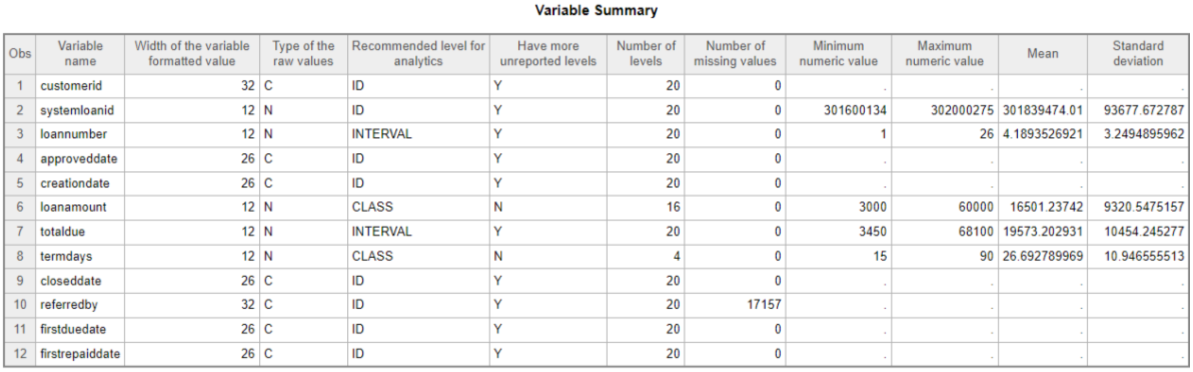
Variable Summary

Obs	Variable name	Width of the variable formatted value	Type of the raw values	Recommended level for analytics	Have more unreported levels	Number of levels	Number of missing values	Minimum numeric value	Maximum numeric value	Mean	Standard deviation
1	customerid	32	C	ID	Y	20	0	.	.	.	.
2	systemloanid	12	N	ID	Y	20	0	301958485	302004050	301980956.66	13431.149908
3	loannumber	12	N	INTERVAL	Y	20	0	2	27	5.1723901099	3.6535690192
4	approveddate	26	C	ID	Y	20	0	.	.	.	.
5	creationdate	26	C	ID	Y	20	0	.	.	.	.
6	loanamount	12	N	CLASS	N	10	0	10000	60000	17809.065934	10749.694571
7	totaldue	12	N	INTERVAL	Y	20	0	10000	68100	21257.377679	11943.510416
8	termdays	12	N	CLASS	N	4	0	15	90	29.261675824	11.51251949
9	referredby	32	C	ID	Y	20	3781	.	.	.	.
10	good_bad_flag	4	C	CLASS	N	2	0	.	.	.	.

Figure 2:Summary Statistics of Perf table

With 3,781 rows left blank, the perf table's summary statistics demonstrate that the referredby variable has many missing values. This suggests that a sizable percentage of the clients withheld referral information. If customers who come through referrals may have different characteristics or behavior patterns from those who come independently, the referredby feature may be indicative of specific behaviors or loan repayment probabilities.

Trainprevloans table



Variable Summary

Obs	Variable name	Width of the variable formatted value	Type of the raw values	Recommended level for analytics	Have more unreported levels	Number of levels	Number of missing values	Minimum numeric value	Maximum numeric value	Mean	Standard deviation
1	customerid	32	C	ID	Y	20	0	.	.	.	.
2	systemloanid	12	N	ID	Y	20	0	301600134	302000275	301839474.01	93677.672787
3	loannumber	12	N	INTERVAL	Y	20	0	1	26	4.1893526921	3.2494895962
4	approveddate	26	C	ID	Y	20	0	.	.	.	.
5	creationdate	26	C	ID	Y	20	0	.	.	.	.
6	loanamount	12	N	CLASS	N	16	0	3000	60000	16501.23742	9320.5475157
7	totaldue	12	N	INTERVAL	Y	20	0	3450	68100	19573.202931	10454.245277
8	termdays	12	N	CLASS	N	4	0	15	90	26.692789969	10.946555513
9	closeddate	26	C	ID	Y	20	0	.	.	.	.
10	referredby	32	C	ID	Y	20	17157	.	.	.	.
11	firstduedate	26	C	ID	Y	20	0	.	.	.	.
12	firstrepaiddate	26	C	ID	Y	20	0	.	.	.	.

Figure 3:Summary statistics for trainprevloans csv file

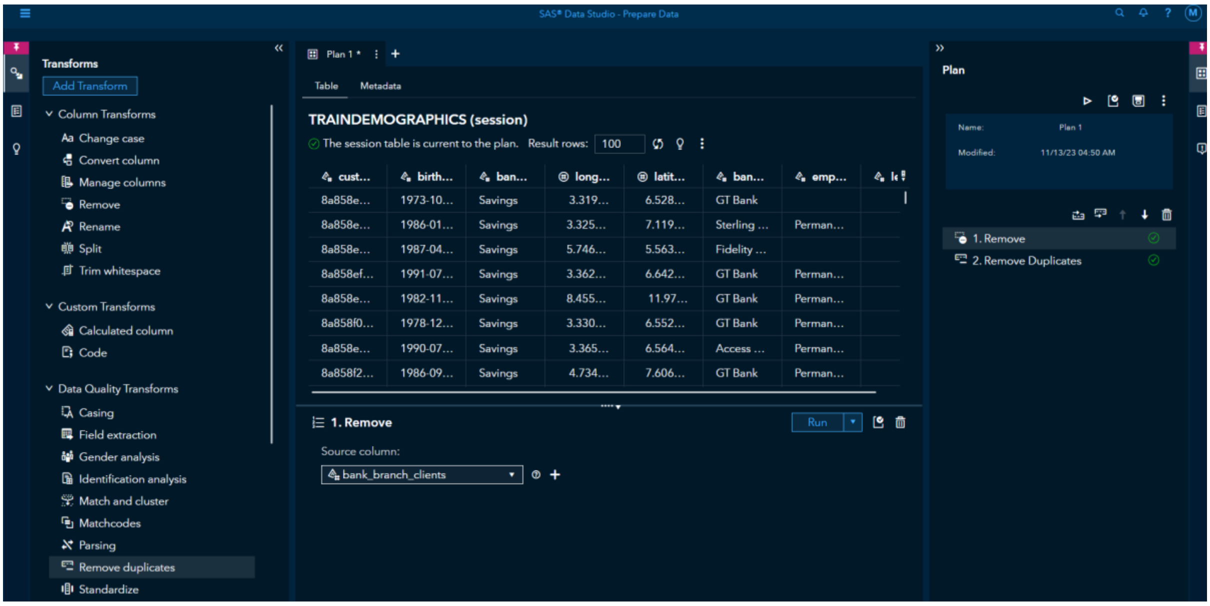
The referredby column, which has 17,157 missing entries overall, stands out in the trainprevloans table's summary data. This implies that a large number of the clients in this dataset did not enter through a referral program or did not have a documented referral source. This pattern may suggest that clients in this dataset do not frequently use referrals.

The lack of data in referredby might restrict the use of this variable in predictive modeling unless the missingness itself can be useful or can be well imputed. It may also have an impact on specific types of analysis, such as referral influence on loan repayment.

Removing duplicates and missing values

At first, there were 4,346 entries in the demographics table, each with nine attributes. There were 12 duplicate entries among the 4,334 unique customer records that resulted from eliminating duplicates using the customerid column.

The bank_branch_clients column was also eliminated from the dataset. Due to the sparsity of the data, this column was removed since it had a high number of missing values, which would have complicated the analysis and modeling process without adding much value.



TRAINDEMOGRAPHICS (session)

The session table is current to the plan. Result rows: 100

cust...	birth...	ban...	long...	latit...	ban...	emp...	le
8a858e...	1973-10...	Savings	3.319...	6.528...	GT Bank		
8a858e...	1986-01...	Savings	3.325...	7.119...	Sterling ...	Perman...	
8a858e...	1987-04...	Savings	5.746...	5.563...	Fidelity ...		
8a858ef...	1991-07...	Savings	3.362...	6.642...	GT Bank	Perman...	
8a858e...	1982-11...	Savings	8.455...	11.97...	GT Bank	Perman...	
8a858f0...	1978-12...	Savings	3.330...	6.552...	GT Bank	Perman...	
8a858e...	1990-07...	Savings	3.365...	6.564...	Access ...	Perman...	
8a858f2...	1986-09...	Savings	4.734...	7.606...	GT Bank	Perman...	

1. Remove

Source column: bank_branch_clients

Name: Plan 1

Modified: 11/13/23 04:50 AM

1. Remove

2. Remove Duplicates

Figure 4: Removing bank_branch_clients column and duplicates

After deleting the column and duplicates, the data has been saved in the new table called as traindemographics_1

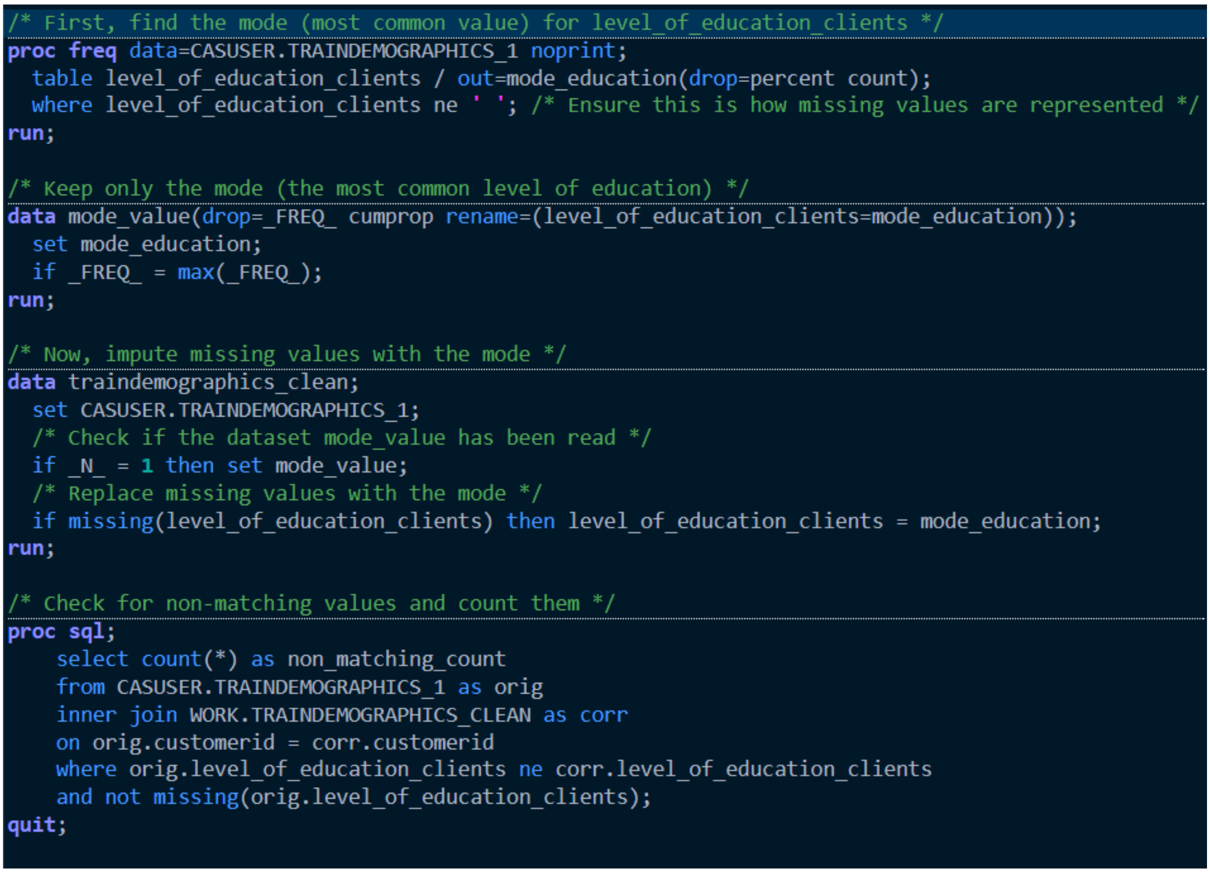
There are several choices for columns like level_of_education_clients that have many missing values:

Like bank branch clients, a column can be removed if it is not essential to the analysis or predictive modeling.

Depending on the nature of the data and the objectives of the research, we might try to impute the missing values using statistical or machine learning approaches if it's possibly relevant.

Although the employment_status_clients column similarly lacks data, it may be more important to keep it as employment status is a reliable indicator of a borrower's capacity to repay a loan. To figure out the optimal course of action, the method for this column may entail more advanced imputation techniques or pattern analysis of the missing data.

Code

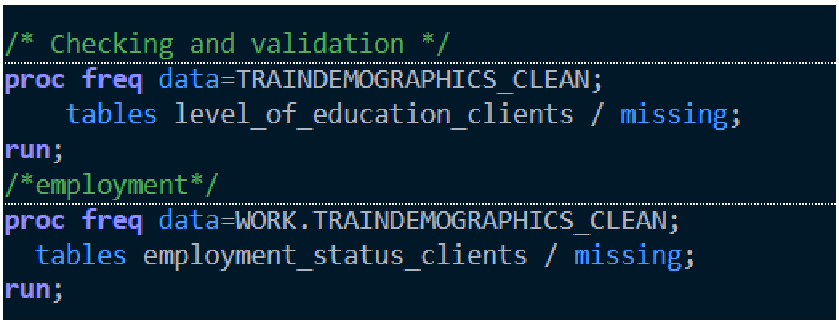


```
/* First, find the mode (most common value) for level_of_education_clients */
proc freq data=CASUSER.TRAINDEMOGRAPHICS_1 noprint;
  table level_of_education_clients / out=mode_education(drop=percent count);
  where level_of_education_clients ne ' '; /* Ensure this is how missing values are represented */
run;

/* Keep only the mode (the most common level of education) */
data mode_value(drop=_FREQ_ cumprop rename=(level_of_education_clients=mode_education));
  set mode_education;
  if _FREQ_ = max(_FREQ_);
run;

/* Now, impute missing values with the mode */
data traindemographics_clean;
  set CASUSER.TRAINDEMOGRAPHICS_1;
  /* Check if the dataset mode_value has been read */
  if _N_ = 1 then set mode_value;
  /* Replace missing values with the mode */
  if missing(level_of_education_clients) then level_of_education_clients = mode_education;
run;

/* Check for non-matching values and count them */
proc sql;
    select count(*) as non_matching_count
    from CASUSER.TRAINDEMOGRAPHICS_1 as orig
    inner join WORK.TRAINDEMOGRAPHICS_CLEAN as corr
    on orig.customerid = corr.customerid
    where orig.level_of_education_clients ne corr.level_of_education_clients
    and not missing(orig.level_of_education_clients);
quit;
```

This SAS code may be used to manage the missing values in the level_of_education_clients variable by considering the missing data as a new category 'Unknown' or by imputing the most common education level (mode).

- The first **proc freq** step finds the most common **level_of_education_clients** and stores it in the **mode_education** dataset.
- The second data step, **data mode_value**, isolates this mode value into a variable called **mode_education**.
- The final data step, **data traindemographics_clean**, imputes the missing values. The mode value from **mode_value** is applied to all missing entries of **level_of_education_clients** in the **traindemographics_clean** dataset.

Checking and validation for missing values of level of education

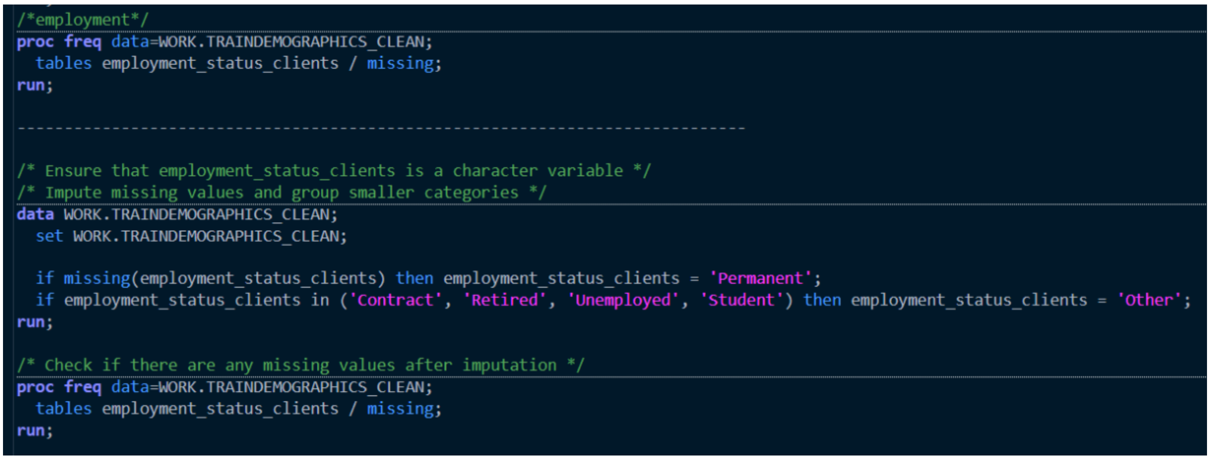


```
/* Checking and validation */
proc freq data=TRAINDEMOGRAPHICS_CLEAN;
    tables level_of_education_clients / missing;
run;
/*employment*/
proc freq data=WORK.TRAINDEMOGRAPHICS_CLEAN;
  tables employment_status_clients / missing;
run;
```

Dataset shows that there are no longer any missing entries in the level_of_education_clients field. Level of education customers' Number of Missing Values is displayed as 0, indicating that the imputation or category assignment for missing data has been effective.

Employment_status_clients

- Impute Missing Values or Create 'Unknown' Category
- Grouping Small Categories



```
/*employment*/
proc freq data=WORK.TRAINDEMOGRAPHICS_CLEAN;
  tables employment_status_clients / missing;
run;

----------------------------------------------------------------------------

/* Ensure that employment_status_clients is a character variable */
/* Impute missing values and group smaller categories */
data WORK.TRAINDEMOGRAPHICS_CLEAN;
  set WORK.TRAINDEMOGRAPHICS_CLEAN;

  if missing(employment_status_clients) then employment_status_clients = 'Permanent';
  if employment_status_clients in ('Contract', 'Retired', 'Unemployed', 'Student') then employment_status_clients = 'Other';
run;

/* Check if there are any missing values after imputation */
proc freq data=WORK.TRAINDEMOGRAPHICS_CLEAN;
  tables employment_status_clients / missing;
run;
```

Given this distribution, here's what you can do in SAS:

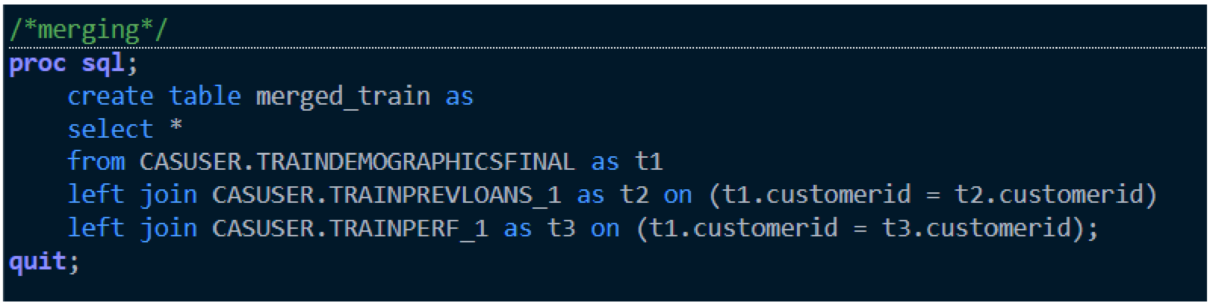
1. **Impute Missing Values**: Since 'Permanent' is the overwhelmingly common category, you might consider imputing the missing values with 'Permanent'.
2. **Grouping Small Categories**: Categories like 'Contract', 'Retired', and potentially 'Unemployed' and 'Student' could be grouped into a 'Other' category if you believe that the specifics of these categories do not have different impacts on the loan default rate.

3. **Create an 'Unknown' Category**: The missing values can be assigned to an 'Unknown' category. This preserves the data without making assumptions about the missing information.

In the same way, removing the duplicates and removing the column called “referred_by” from both the tables and saving the updated data in trainprevloans_1 and trainperf_1 respectively.

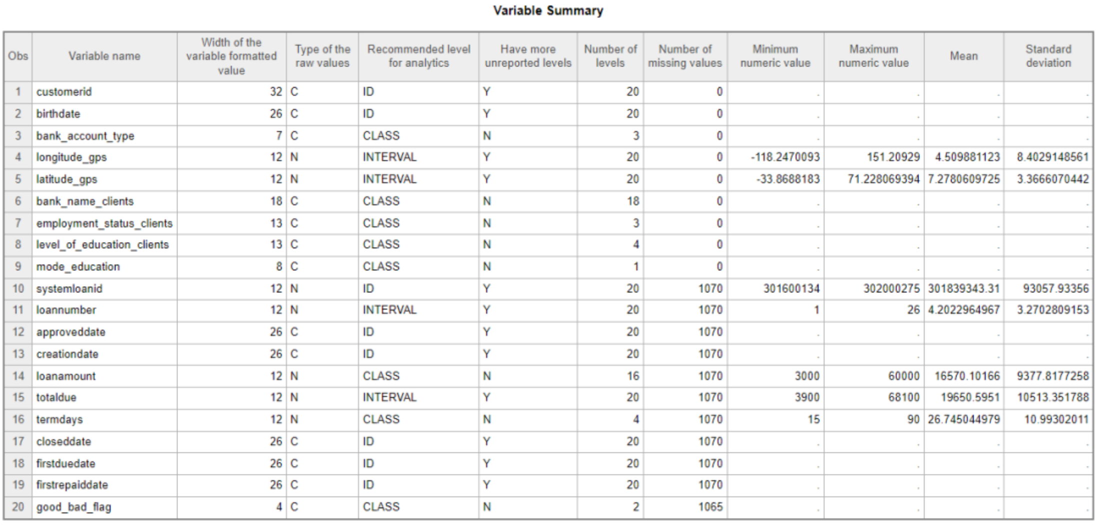
Now once the missing values and duplicates are been removed, its time to merge the three datasets into the new table merge_train table.

Merging the tables



```
/*merging*/
proc sql;
    create table merged_train as
    select *
    from CASUSER.TRAINDEMOGRAPHICSFINAL as t1
    left join CASUSER.TRAINPREVLOANS_1 as t2 on (t1.customerid = t2.customerid)
    left join CASUSER.TRAINPERF_1 as t3 on (t1.customerid = t3.customerid);
quit;
```

After merging the tables, have found that the many values are missing in the columns. There are 1070 missing values in several columns, including



Variable Summary

Obs	Variable name	Width of the variable formatted value	Type of the raw values	Recommended level for analytics	Have more unreported levels	Number of levels	Number of missing values	Minimum numeric value	Maximum numeric value	Mean	Standard deviation
1	customerid	32	C	ID	Y	20	0	.	.	.	.
2	birthdate	26	C	ID	Y	20	0	.	.	.	.
3	bank_account_type	7	C	CLASS	N	3	0	.	.	.	.
4	longitude_gps	12	N	INTERVAL	Y	20	0	-118.2470093	151.20929	4.509881123	8.4029148561
5	latitude_gps	12	N	INTERVAL	Y	20	0	-33.8688183	71.228069394	7.2780609725	3.3666070442
6	bank_name_clients	18	C	CLASS	N	18	0	.	.	.	.
7	employment_status_clients	13	C	CLASS	N	3	0	.	.	.	.
8	level_of_education_clients	13	C	CLASS	N	4	0	.	.	.	.
9	mode_education	8	C	CLASS	N	1	0	.	.	.	.
10	systemloanid	12	N	ID	Y	20	1070	301600134	302000275	301839343.31	93057.93356
11	loannumber	12	N	INTERVAL	Y	20	1070	1	26	4.2022964967	3.2702809153
12	approveddate	26	C	ID	Y	20	1070	.	.	.	.
13	creationdate	26	C	ID	Y	20	1070	.	.	.	.
14	loanamount	12	N	CLASS	N	16	1070	3000	60000	16570.10166	9377.8177258
15	totaldue	12	N	INTERVAL	Y	20	1070	3900	68100	19650.5951	10513.351788
16	termdays	12	N	CLASS	N	4	1070	15	90	26.745044979	10.99302011
17	closeddate	26	C	ID	Y	20	1070	.	.	.	.
18	firstduedate	26	C	ID	Y	20	1070	.	.	.	.
19	firstrepaiddate	26	C	ID	Y	20	1070	.	.	.	.
20	good_bad_flag	4	C	CLASS	N	2	1065	.	.	.	.

systemloanid, **loannumber**, **approveddate**, **creationdate**, **loanamount**, **totaldue**, **termdays**, **closeddate**, **firstduedate**, and **firstrepaiddate**. Additionally, there are 1065 missing values for your target variable **good_bad_flag**. Before building your predictive model, you'll need to decide how to handle these missing values.

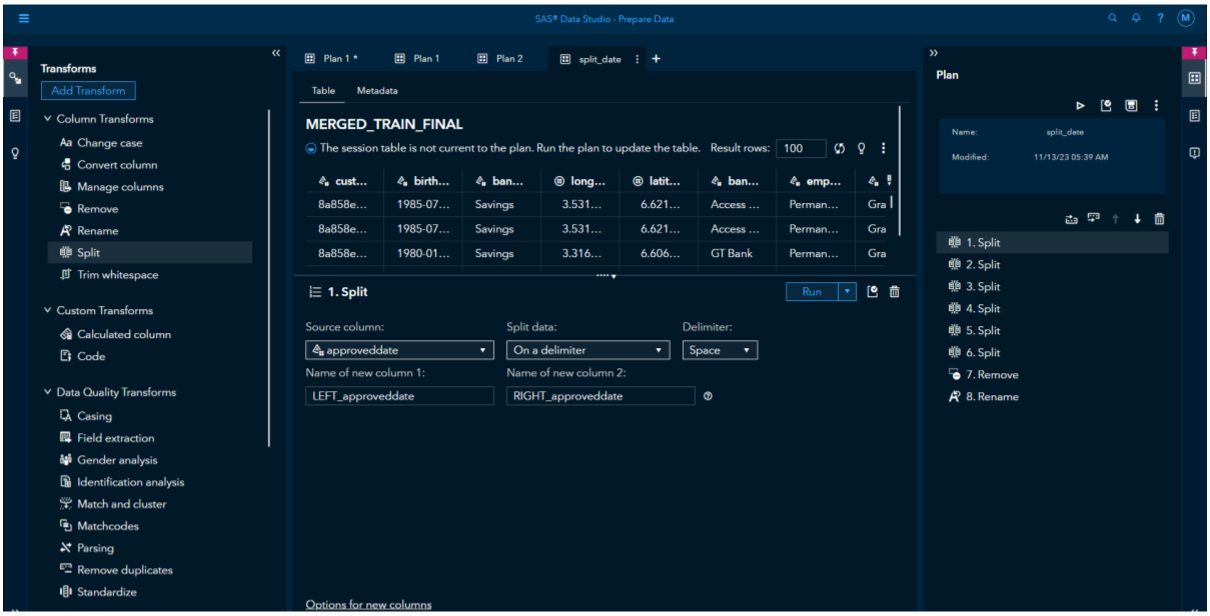
Yes, for the purpose of building a predictive model, especially when your target variable (**good_bad_flag**) has missing values, it is generally a good idea to remove those rows. The target variable is essential for supervised learning, and if it's missing, the model has no basis for learning in those particular cases.

Missing Target Values: A row cannot be utilized for training if it has a missing target value (good_bad_flag), as this information does not aid in learning. Typically, these rows should be deleted.

Missing Predictive Features: If the missingness is not significant and does not induce bias, you may want to attempt imputation for additional columns that have missing values. If a sizable piece of a predictive trait is absent, or if the absence is informative in and of itself, you may require a more involved strategy than straightforward deletion.

So, have decided to delete the rows for the missing columns such that a cleaned data will be created.

After deleting the rows, to get the date, month, year from the dataset have to split the function with the delimeter as ‘ ‘ (space).



MERGED_TRAIN_FINAL

The session table is not current to the plan. Run the plan to update the table. Result rows: 100

cust...	birth...	ban...	long...	latit...	ban...	emp...	
8a858e...	1985-07...	Savings	3.531...	6.621...	Access ...	Perman...	Gra
8a858e...	1985-07...	Savings	3.531...	6.621...	Access ...	Perman...	Gra
8a858e...	1980-01...	Savings	3.316...	6.606...	GT Bank	Perman...	Gra

1. Split

Source column: approveddate

Split data: On a delimiter

Delimiter: Space

Name of new column 1: LEFT_approveddate

Name of new column 2: RIGHT_approveddate

Name: split_date

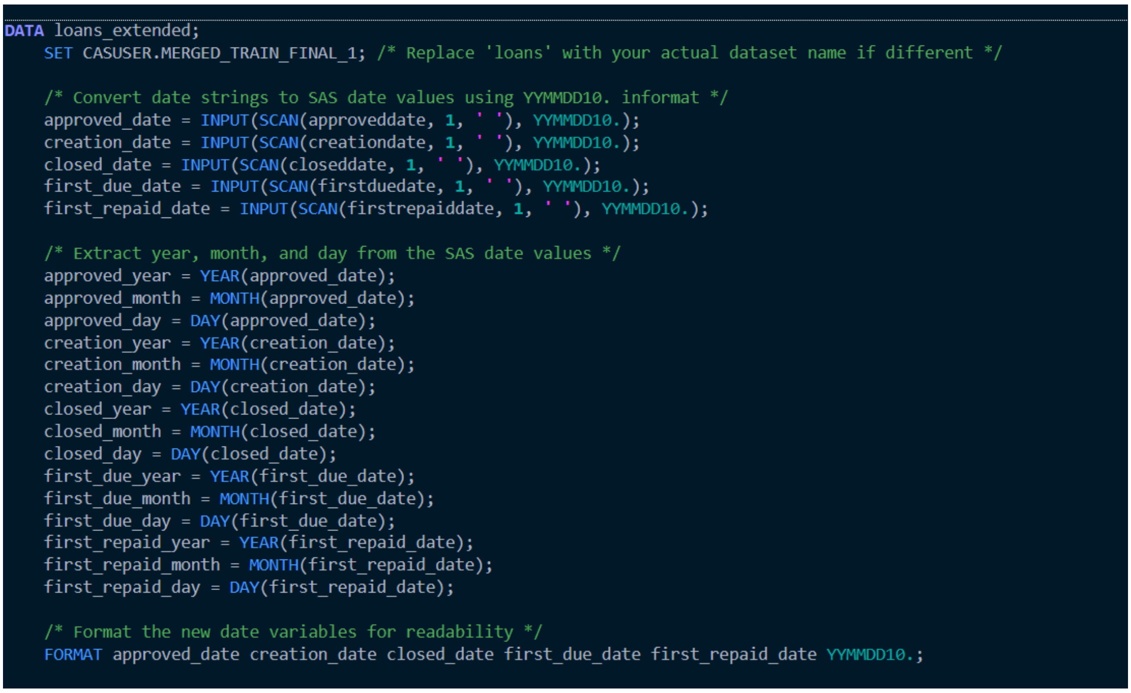
Modified: 11/13/23 05:39 AM

1. Split
2. Split
3. Split
4. Split
5. Split
6. Split
7. Remove
8. Rename

Then deleted the unwanted rows from the table and saved the table name as merged_train_final_1.

Feature Engineering

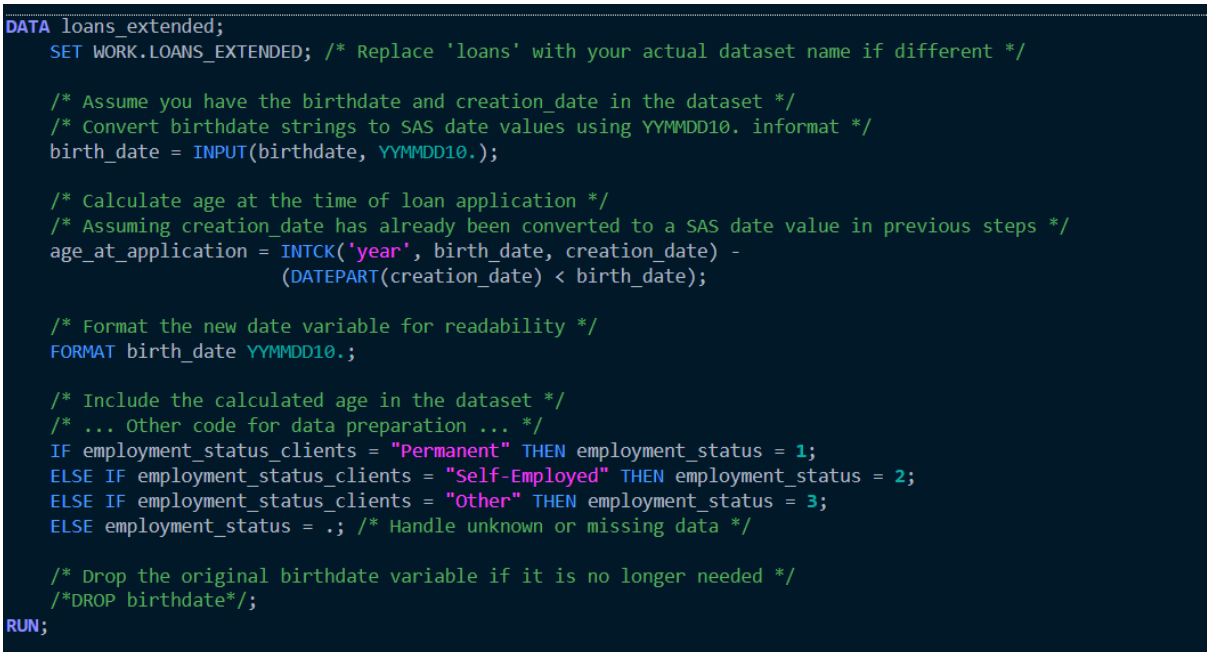
1. **Time of Day for Approval and Creation**:
    - Use the **HOUR** function to extract the hour part of the **approveddate** and **creationdate** timestamps.
2. **Day of the Week**:
    - Convert the timestamp to a SAS date value using **INPUT** and **ANYDTDTE.** informat, and then use the **WEEKDAY** function to get the day of the week.
3. **Duration Between Approval and Creation**:
    - Calculate the difference in times using the **DHMS** function to convert date and time parts into a SAS datetime value, and then take the difference.
4. **Duration Between Due Date and Repayment Date**:
    - Similar to the duration between approval and creation, you can use **DHMS** to create datetime values and then calculate the difference.



```
DATA loans_extended;
    SET CASUSER.MERGED_TRAIN_FINAL_1; /* Replace 'loans' with your actual dataset name if different */

    /* Convert date strings to SAS date values using YYMMDD10. informat */
    approved_date = INPUT(SCAN(approveddate, 1, ' '), YYMMDD10.);
    creation_date = INPUT(SCAN(creationdate, 1, ' '), YYMMDD10.);
    closed_date = INPUT(SCAN(closeddate, 1, ' '), YYMMDD10.);
    first_due_date = INPUT(SCAN(firstduedate, 1, ' '), YYMMDD10.);
    first_repaid_date = INPUT(SCAN(firstrepaiddate, 1, ' '), YYMMDD10.);

    /* Extract year, month, and day from the SAS date values */
    approved_year = YEAR(approved_date);
    approved_month = MONTH(approved_date);
    approved_day = DAY(approved_date);
    creation_year = YEAR(creation_date);
    creation_month = MONTH(creation_date);
    creation_day = DAY(creation_date);
    closed_year = YEAR(closed_date);
    closed_month = MONTH(closed_date);
    closed_day = DAY(closed_date);
    first_due_year = YEAR(first_due_date);
    first_due_month = MONTH(first_due_date);
    first_due_day = DAY(first_due_date);
    first_repaid_year = YEAR(first_repaid_date);
    first_repaid_month = MONTH(first_repaid_date);
    first_repaid_day = DAY(first_repaid_date);

    /* Format the new date variables for readability */
    FORMAT approved_date creation_date closed_date first_due_date first_repaid_date YYMMDD10.;
```

The dates appear to be in the ISO 8601 format, which is a widely used international standard for date and time representations. In SAS, you can use the **ANYDTDTE.** informat to read dates in this format.

In SAS, you can calculate the age of a customer at the time of the loan application by subtracting the birth date from the application date. If you have the birth dates in **YYYY-MM-DD** format and the application date (which could be the **creation_date** you previously used), you can use the **intck** function in SAS to calculate the difference in years between the two dates.



```
DATA loans_extended;
    SET WORK.LOANS_EXTENDED; /* Replace 'loans' with your actual dataset name if different */

    /* Assume you have the birthdate and creation_date in the dataset */
    /* Convert birthdate strings to SAS date values using YYMMDD10. informat */
    birth_date = INPUT(birthdate, YYMMDD10.);

    /* Calculate age at the time of loan application */
    /* Assuming creation_date has already been converted to a SAS date value in previous steps */
    age_at_application = INTCK('year', birth_date, creation_date) -
                         (DATEPART(creation_date) < birth_date);

    /* Format the new date variable for readability */
    FORMAT birth_date YYMMDD10.;

    /* Include the calculated age in the dataset */
    /* ... Other code for data preparation ... */
    IF employment_status_clients = "Permanent" THEN employment_status = 1;
    ELSE IF employment_status_clients = "Self-Employed" THEN employment_status = 2;
    ELSE IF employment_status_clients = "Other" THEN employment_status = 3;
    ELSE employment_status = .; /* Handle unknown or missing data */

    /* Drop the original birthdate variable if it is no longer needed */
    /*DROP birthdate*/;
RUN;
```

Once I derived all the data from the parent tables, have deleted the tables which are not necessary.

Model evaluation

Because of the unique nature of the work and the prior study that conducted, model assessment is especially crucial for the project on loan default prediction.

Impact of Decision: Every choice made in the context of loan default prediction, based on the model's forecast, may have far-reaching financial ramifications. Evaluating the model's performance guarantees that the lending institution may depend on it to make choices that will reduce risk and loss of money.

Imbalanced Classes: 'Good' loans probably outweigh 'Bad' loans in the unbalanced dataset that the application works with. Model evaluation makes ensuring that the model is actually recognizing the features of each class and not just predicting the majority class.

To assess the model's performance on validation data, use measures appropriate for binary classification, such as accuracy, precision, recall, F1-score, and AUC-ROC curve.

Selection of model for testing

The selection of Decision Tree, Random Forest, and Logistic Regression models for the loan default prediction study is in line with the goal for a number of reasons, all of which stem from the advantages and traits of each modeling technique:

Decision Tree:

Interpretability: Decision trees give stakeholders an easy-to-understand visual representation of the decision-making process.

Relevance of Features: By highlighting the most important factors for default prediction, they automatically choose features.

Random Forest:

Accuracy: By averaging several decision trees to reduce overfitting, Random Forest, as an ensemble of decision trees, often produces a greater accuracy and better generalization.

Robustness: It successfully manages non-linear data and outliers, which are frequent in financial datasets.

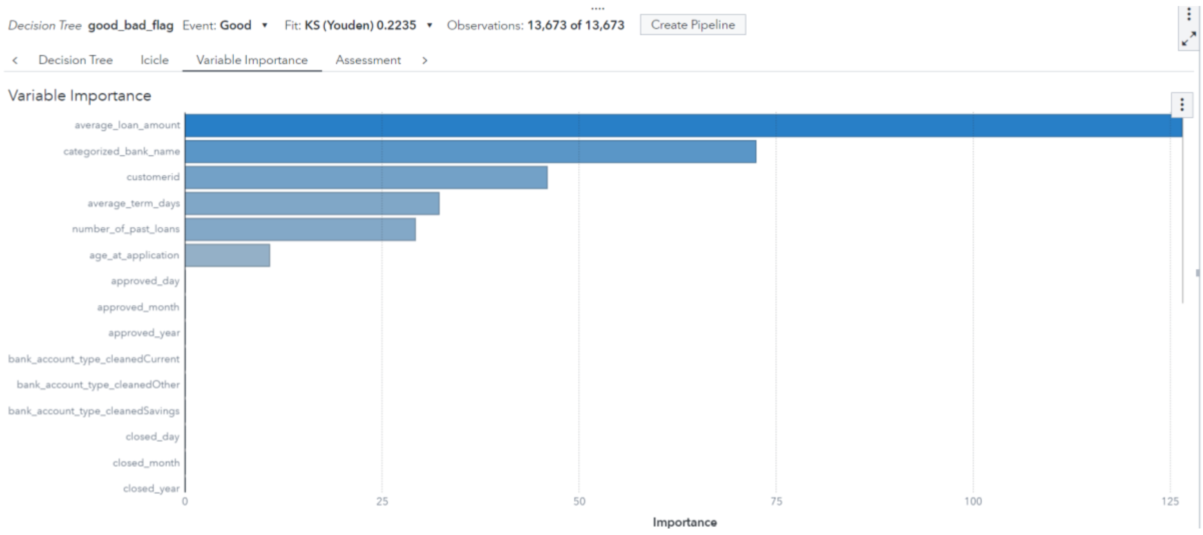
Logistic Regression:

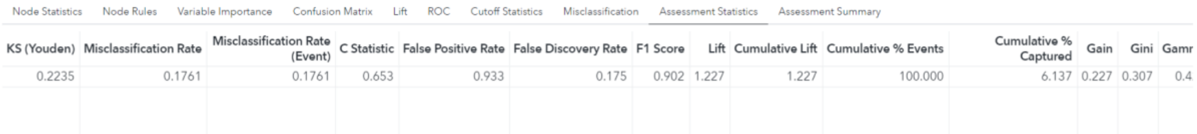
Speed: The computational efficiency of logistic regression models enables rapid iteration and retraining with fresh data.

Probabilistic Interpretation: They give default probabilities, making risk assessment simple.

Test Deployment and Predictions:

Decision tree


Decision Tree **good_bad_flag** Event: **Good** Fit: **KS (Youden) 0.2235** Observations: **13,673 of 13,673**



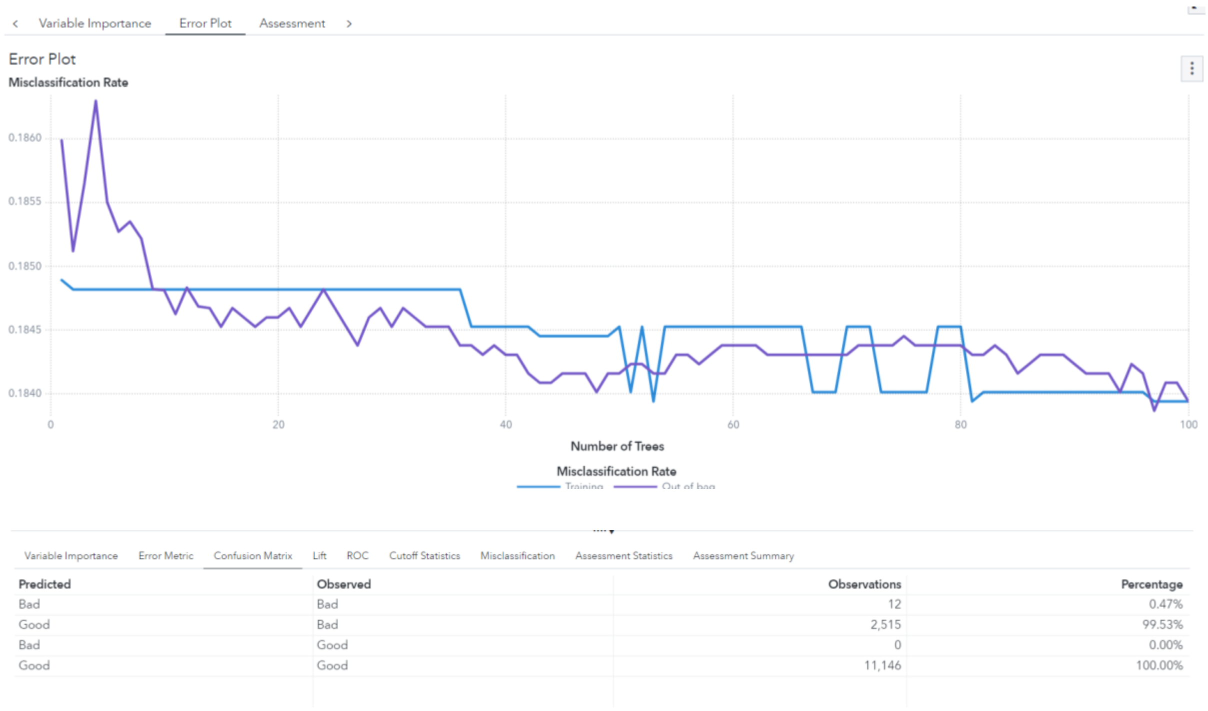
KS (Youden)	Misclassification Rate	Misclassification Rate (Event)	C Statistic	False Positive Rate	False Discovery Rate	F1 Score	Lift	Cumulative Lift	Cumulative % Events	Cumulative % Captured	Gain	Gini	Gamr
0.2235	0.1761	0.1761	0.653	0.933	0.175	0.902	1.227	1.227	100.000	6.137	0.227	0.307	0.4

Findings

The loan default prediction project's Decision Tree model study produced some insightful results. Based on performance indicators, such as a higher Kolmogorov-Smirnov (KS) statistic that shows a strong capacity to distinguish between "good" and "bad" loans, the Decision Tree has been named the best model. One of the model's main advantages is its interpretability, as the decision rules clearly define the parameters for assessing default risk. These rules demonstrated the importance of several factors in loan result prediction, including customerid and categorized_bank_name. Understanding the variables that affect default risk is helpful because it may direct risk-reduction tactics.

The Decision Tree also showed a high degree of classification accuracy, as evidenced by its F1 score, which shows a performance that strikes a balance between recall and precision. As per the cross-validation report, the model demonstrated exceptional competence in accurately classifying loans as "Good," with a 100% success rate in capturing this category. It is noteworthy, though, that the false positive rate was high, suggesting that although the model is good at detecting "Good" loans, it could be unduly conservative, which could result in more loans being rejected than required. Overall, the Decision Tree model is a useful tool for lenders looking to reduce risk while guaranteeing credit availability to qualified applicants because of its transparency, use, and efficacy in identifying loan defaults.

Random Forcest

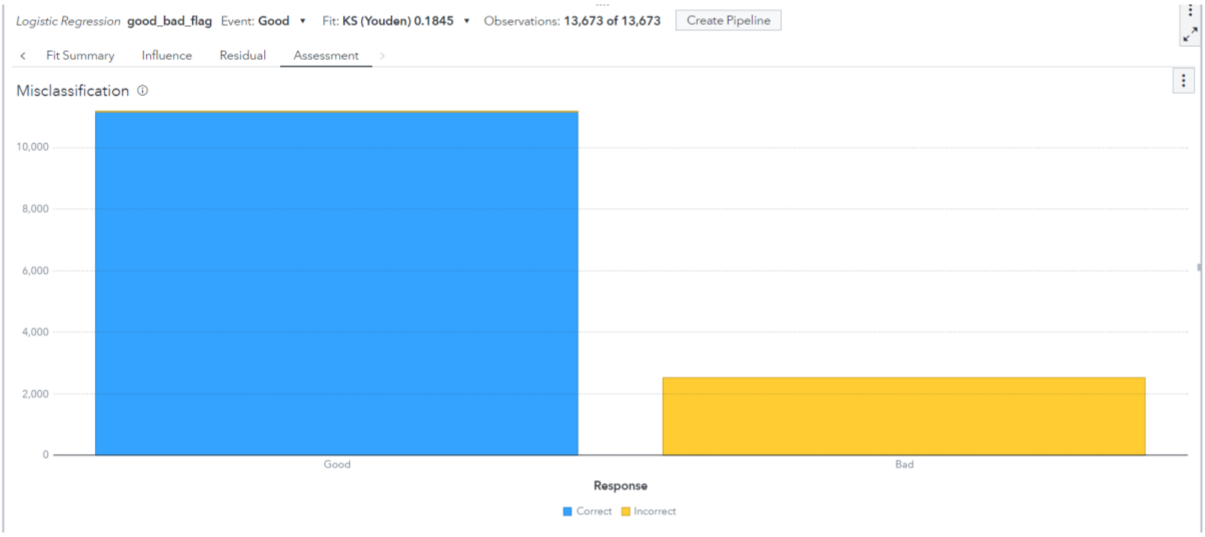


Predicted	Observed	Observations	Percentage
Bad	Bad	12	0.47%
Good	Bad	2,515	99.53%
Bad	Good	0	0.00%
Good	Good	11,146	100.00%

Findings

For the purpose of predicting loan default, an ensemble of decision trees known as the Random Forest model was assessed. In our investigation, the Random Forest model fared worse than the Decision Tree, despite having a reputation for excellent accuracy and resilience to overfitting. The model's ROC AUC score was lower than the Decision Tree model's, indicating that it performed less well in differentiating between "good" and "bad" loans. Additionally, the Random Forest model showed a comparatively high misclassification rate, and the KS statistic suggested a lower capacity for discriminating. These results imply that, for this specific dataset and goal, the Random Forest's complexity did not result in a corresponding improvement in prediction accuracy, perhaps because of the subtleties of the dataset.

Logistic regression

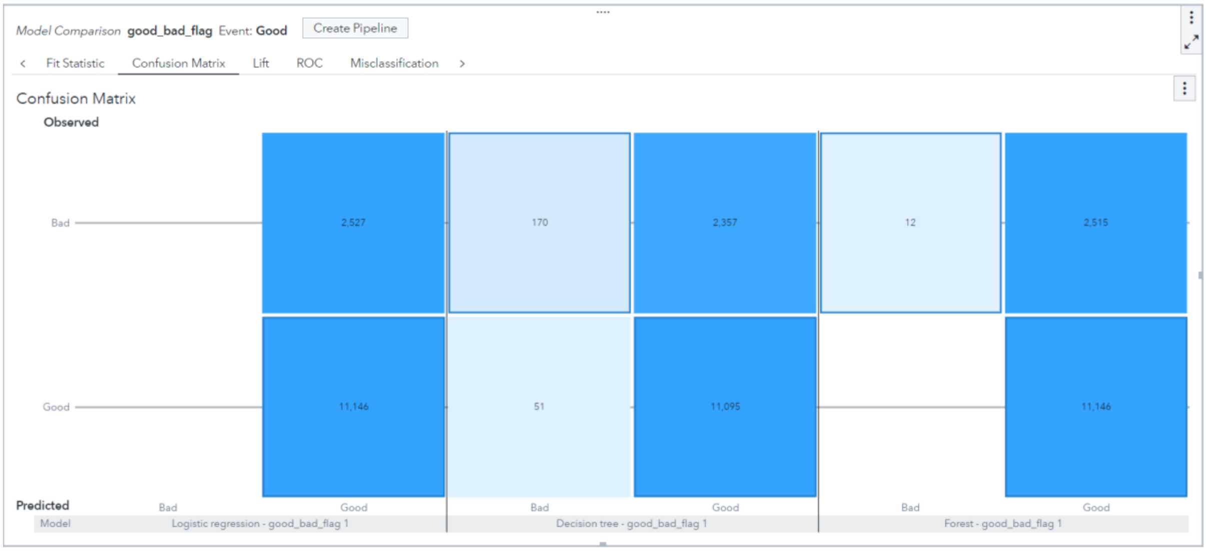


Logistic Regression **good_bad_flag** Event: **Good** Fit: **KS (Youden) 0.1845** Observations: **13,673 of 13,673**

Findings

The loan default prediction project's performance using the Logistic Regression model provided insightful information. Logistic Regression, a statistical model renowned for its ease of use and interpretability, supplied a performance baseline with a ROC AUC value that was passably high but lower than that of the Decision Tree model. The power of the model is in its capacity to yield probabilities for default likelihood, which are readily comprehensible as risk levels. When it comes to financial firms trying to estimate the likelihood of loan defaults, this characteristic is very helpful. But because logistic regression is linear, it might not have been able to fully represent the more intricate patterns in the data that would have been better captured by non-linear techniques like decision trees.

Model Comparison



Confusion Matrix

Observed	Logistic regression - good_bad_flag 1: Bad	Logistic regression - good_bad_flag 1: Good	Decision tree - good_bad_flag 1: Bad	Decision tree - good_bad_flag 1: Good	Forest - good_bad_flag 1: Bad	Forest - good_bad_flag 1: Good
Bad		2,527	170	2,357	12	2,515
Good		11,146	51	11,095		11,146

Predicted

The following are the main conclusions drawn by contrasting the outcomes of the three models—Logistic Regression, Decision Tree, and Random Forest—with the aim of forecasting loan defaults:

Logistic Regression:

With its quick and easily understandable results, this model provided a baseline. It's especially helpful in evaluating the likelihood of default, giving a clear probabilistic result that may be used to establish cutoff points for judgment calls. As seen by its ROC AUC score, it did not perform as well as the other models that were evaluated.

Decision Tree:

The model that performed best in terms of interpretability and performance was the Decision Tree. Out of the three, it had the highest KS statistic, demonstrating its greater discriminative ability to differentiate between "good" and "bad" loans. It is simple for stakeholders to comprehend and have faith in the model's forecasts because of its clear guidelines for decision-making. It also got the largest cumulative lift, indicating that it can efficiently rate people according to their likelihood of defaulting.

Random Forest:

The Random Forest model did not perform better than the Decision Tree in this instance, despite being able to decrease overfitting and making it typically a high performer in many modeling tasks. Compared to the Decision Tree, it had a worse ROC AUC and KS statistic, indicating that the extra complexity was not beneficial in terms of predicting power for this dataset.

Conclusion

Our project's goal was to precisely forecast loan defaults, which is an essential duty for every financial organization. After testing a few models, we discovered that the Decision Tree offered the best combination of prediction performance and interpretability. Among the models evaluated, it had the greatest Kolmogorov-Smirnov statistic, indicating improved discriminating between 'good' and 'bad' loans, a crucial attribute for our goal. Its capacity to give transparent criteria for decision-making and to capture all "good" loans fits in nicely with the demand for concise, useful insights in loan approval procedures.

References

Hosmer Jr, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression. John Wiley & Sons.

Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.