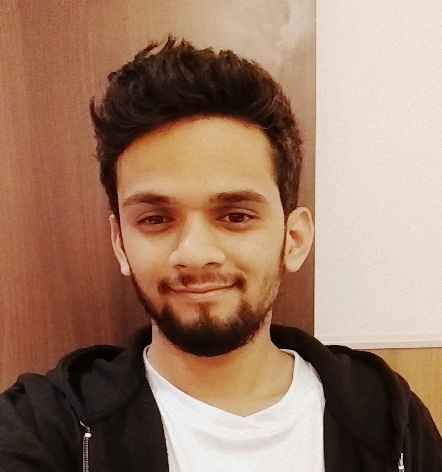


|  |
| --- |
| Lending Club Loan Data |
|  |
| MARCH 6  PRDXN - Data Engineer Code Test  Authored by: Abhishek Verma |



# Part 1: Data Exploration and Evaluation

|  |
| --- |
| Data Exploration and Evaluation This Exploratory Data Analysis of Lending Club Loan Data focuses on exploring and finding possible driving factors which leads to a Bad Loan.  This involves stepwise processes involved in data cleaning and data preparation processes such as handling missing values, removing redundant and irrelevant variables, normalization of features, outlier detection and removal. |
| *“Good data science is more about the questions you pose of the data rather than data munging and analysis”*  *– Riley Newman* |
| *Approach –* To get started with data exploration and pre-processing of any data science problem, I follow a certain set of procedure which serves me good insights all the time. This can be done in below steps –   1. Variable Identification 2. Univariate Analysis 3. Bi-Variate Analysis 4. Missing Values treatment 5. Outlier treatment 6. Variable transformation 7. Variable creation   But before jumping to solution first we will look at the problem itself more closely to check what is the loan status and counts of Good and Bad Loan. Below graphs shows the status of loan in each category and the count (0 = good loan and 1 = bad loan) -  *A screenshot of a cell phone  Description automatically generatedA picture containing screenshot  Description automatically generated*  UNIVARIATE ANALYSIS   1. Term: This shows number of installments to be paid for loan. Around 80% of the loans are given on 36 months terms.   *A screenshot of a cell phone  Description automatically generated* |

1. Grade: These grades are assigned by Lending Club from (A-G). Grade B and C contributes more than 50% of the loans

A screenshot of a cell phone

Description automatically generated

1. Employment Length: This is the duration of employment for employees.

A screenshot of a cell phone

Description automatically generated

1. Home Ownership: Most of the Home ownership status is either rent or mortgage.

A screenshot of a cell phone

Description automatically generated

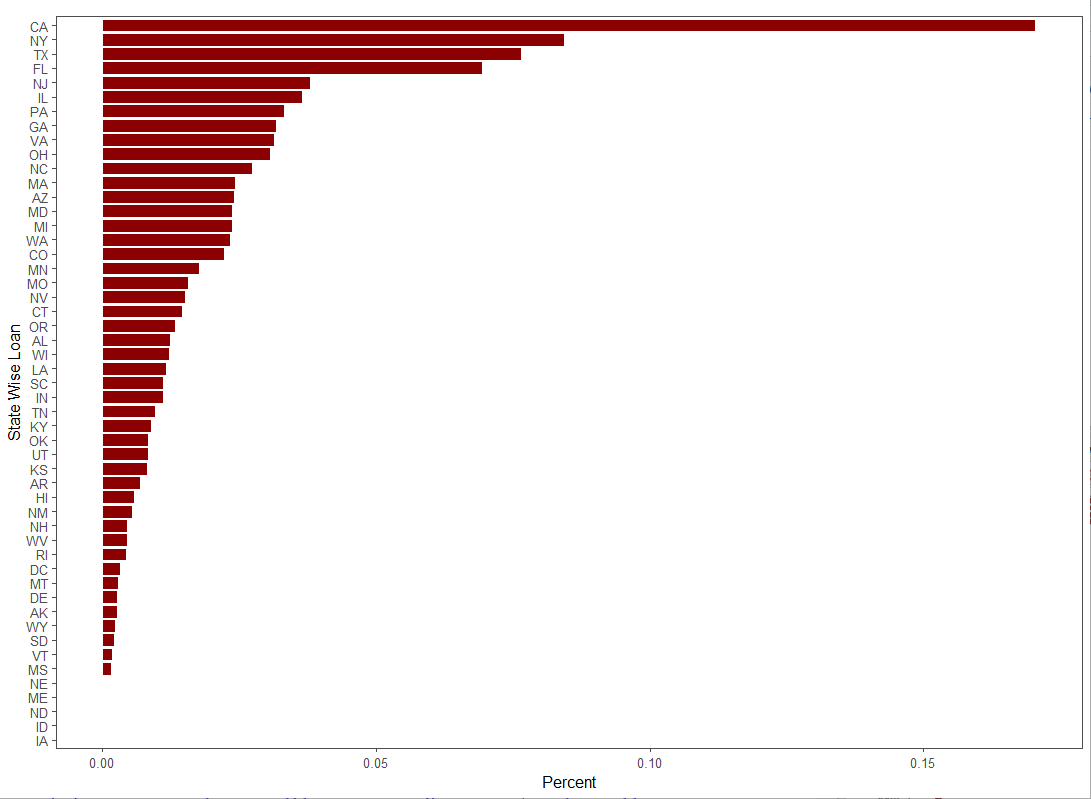
1. Verification Status: More than 30% of the loans are not verified.A screenshot of a cell phone

   Description automatically generated
2. Purpose: consolidation of debt and credit card bill are the two major reasons for which loan was given.

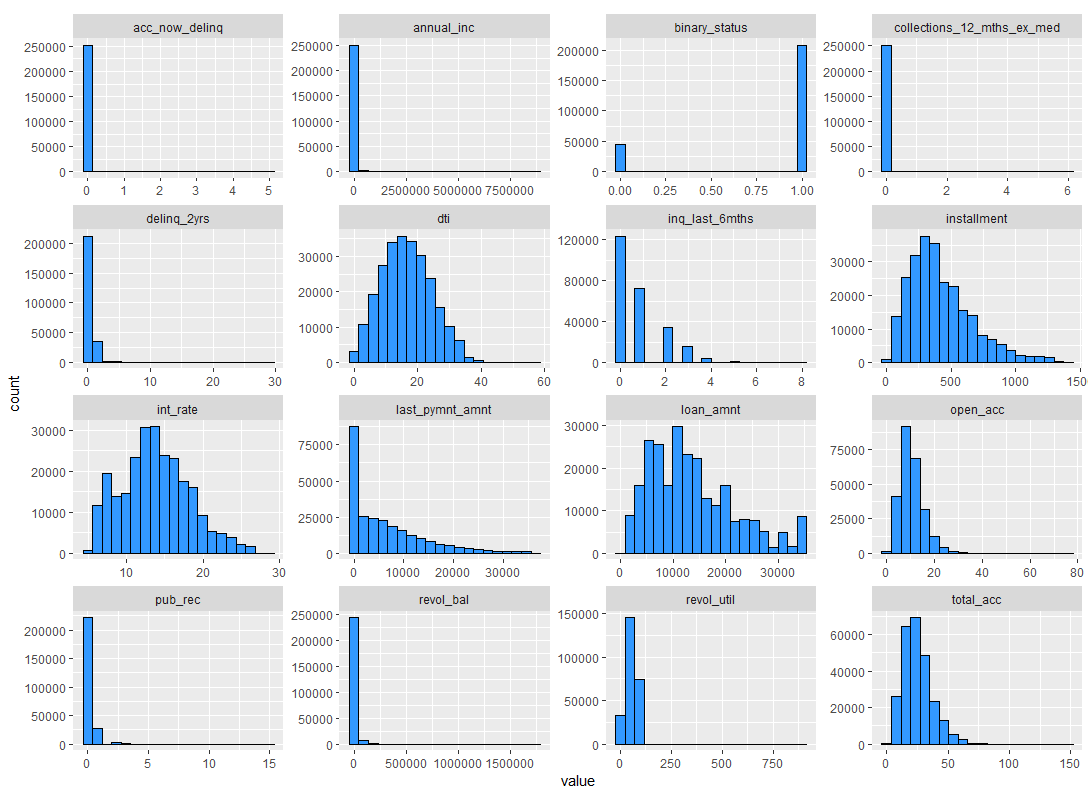
A screenshot of a cell phone

Description automatically generated

1. State: More than 15% of the loans are applied in California. New York and Texas share a percentage of (8-9%) and Florida shares 6%.



Analysis of distribution of continuous variables



SEGMENTED UNIVARIATE ANALYSIS

1. Term and Loan Status: The percentage of Bad Loans in case of loans with 60 months term is higher as compared to the loans with 36 months term.

A screenshot of a cell phone

Description automatically generated

1. Grade and Loan Status: Bad Loans increases with increase in Grade from A to G, which indicates that a loan with grade A means lowest risk of loan default and G means higher risk of loan default.

A screenshot of a cell phone

Description automatically generated

1. Employee length and Loan Status: Number of bad loans are almost equal among all employment length, means this variable has less significance in deciding Good and Bad Loan.

A screenshot of a cell phone

Description automatically generated

1. Home Ownership and Loan Status: Home ownership status with ‘other’ has the greatest number of Bad Loans.

A screenshot of a cell phone

Description automatically generated

1. Verification and Loan Status: Percentage of Bad Loans in verified category is slightly more than other two.

A screenshot of a cell phone

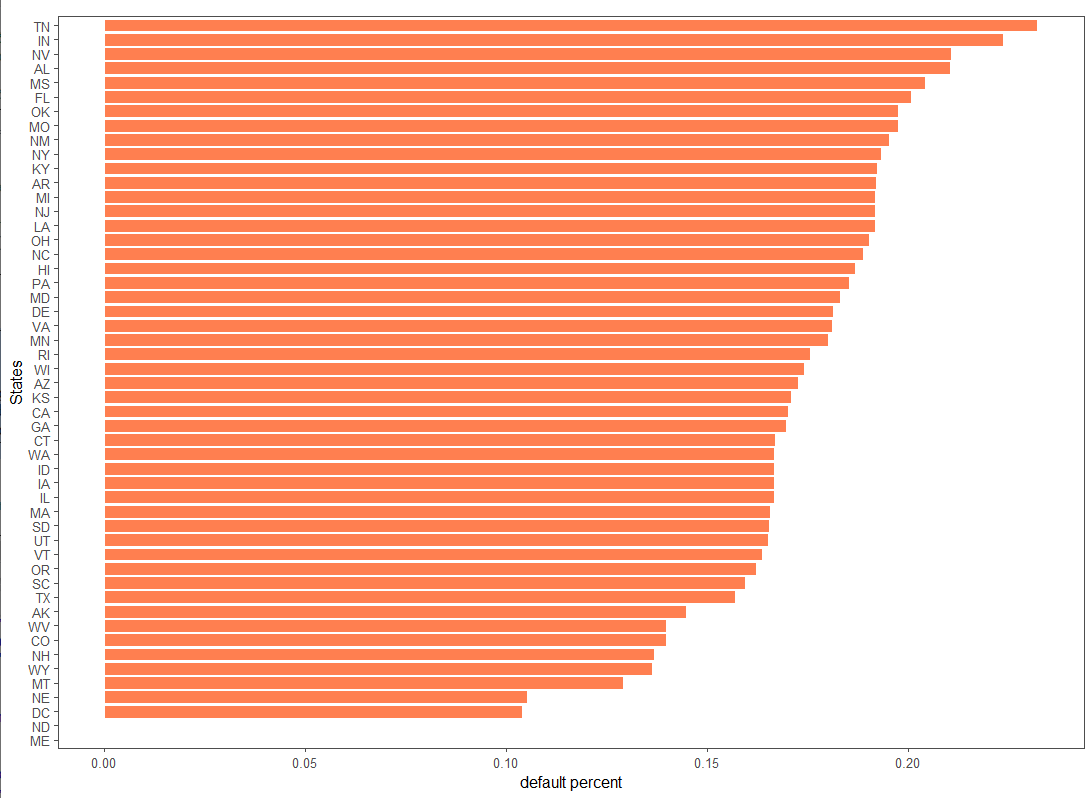
Description automatically generated

1. Purpose of Loan: The default rate in small business is higher than other categories.

A screenshot of a cell phone

Description automatically generated

1. State and Loan Status: The default rate is highest in state of TENESSE.



SEGMENTED UNIVARIATE ANALYSIS OF CONTINUOUS VARIABLE

1. Loan Amount and Loan Status: The below distribution of good and bad loan over loan amount looks uniform.

A screenshot of a cell phone

Description automatically generated

1. Interest rate and Loan Status: In general, the number of bad loans increases with increase in interest rate with a slight exception. These exceptions can be treated as outliers as shown in the boxplot.

A screenshot of a cell phone

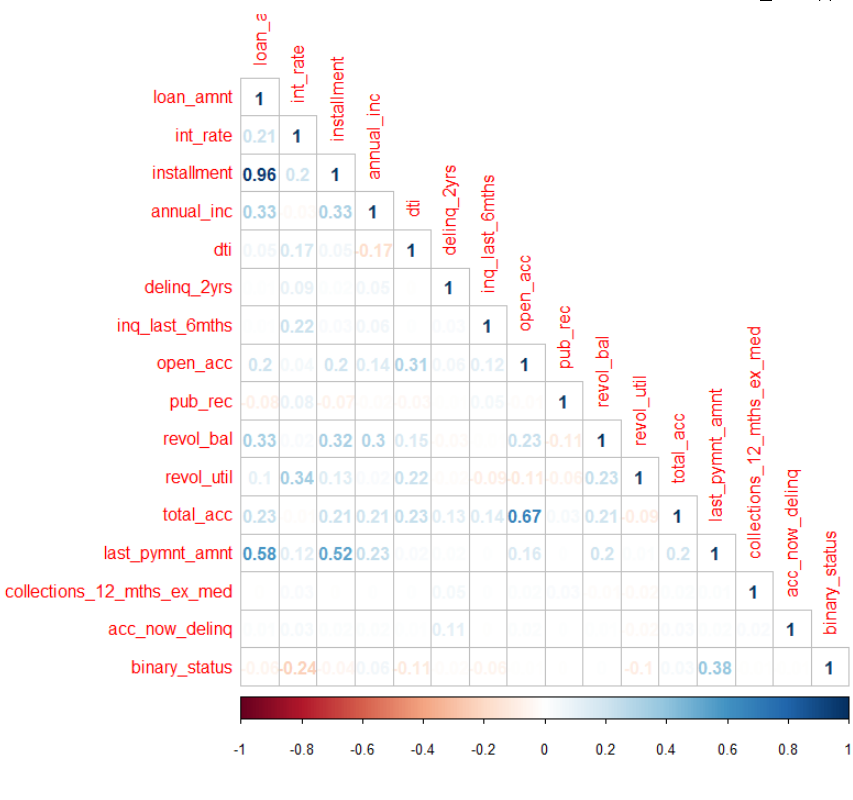
Description automatically generated

A screenshot of a cell phone

Description automatically generated

BIVARIATE ANALYSIS

1. Correlation Check: Positive correlation seems to exist among installment, amount of loan, and interest rate.



Findings:

1. Term - Number of Bad Loans are higher on 60 months term.
2. Interest rate - Number of Bad Loans increases with increase in interest rate.
3. Loan amount - Since, Loan amount and interest rate has positive correlation we can infer that high loan amount also increases the chances of bad loan due to high interest rate.
4. Grade – ‘A’ assigned grade has lower chances of bad loan while ‘G’ assigned grade has higher chances of bad loan. It is increasing linearly from A to G.
5. Even though the main purpose of getting loan is debt consolidation and credit card but the number of bad loans is higher in case of small business category.
6. The state of TENESSE has highest number of bad loans.
7. Number of bad loans are higher when home ownership is “other”.

# Part 2: Engineering

|  |
| --- |
| Data Pipeline and Modelling To create a typical data science pipeline, I follow the approach “Data Science is Awesome” or OSEMN.  Below is the explanation of what it is - |
| *“Data Science is OSEMN”* |
| **OSEMN Pipeline**   * O – Obtaining the data * S – Scrubbing/Cleaning the data * E – Exploring/Visualizing the data will allow us to find patterns and trends * M – Modelling the data will give us our predictive power as a wizard * N – Next Steps to interpret and validate the data.   Below is the detailed explanation of each of the above terms to create a typical data science pipeline from framing the business problem to creating actionable insights.  **Business Question**  A typical pipeline largely depends on the type of problem so before even creating the pipeline, the most crucial and important step is to understand the problem and to know how our solution is going to help business when implemented in production.  **Obtain the Data**  This includes identifying the data source which can be Cloud, Server etc. Here I am sharing the typical process to get data in production based on my experience.  If the data source is server then it can be loaded into HDFS and Hive with the help of Sqoop jobs but we have to create schema to store data in the form of tables in Hive and if the data source is DB2 or Oracle database then the table can be directly loaded into Hive without defining schema.  **Scrubbing/ Cleaning the Data**  This phase of pipeline requires the most time and effort because it involves all the pre-processing steps such as handling missing values, outlier treatment, removing redundant and irrelevent variables.  **Exploring (Exploratory Data Analysis)**  During this phase we try to understand the patterns. It involves different types of visualization and statistical testings. And it is required to drive hidden meanings behind our data through various graphs and analysis.  **Modeling (Machine Learning)**  Here comes the most interesting part. Once the data is cleaned, the data type conversion needs to be done to feed the numbers to machine and with the help of algorithms predictions are done.  **Interpreting (Data Storytelling)**  The most important step in the pipeline is to explain the findings through communication so that it can be well understood by business users. This includes dashboards in data visualiztion tool such as Tableau or Power BI.  **Updating the Model**  Now the model is in production, its important to update the model periodically, depending on how often we receive new data. Models often fails to catch trend or seasonal change hence, to make sure the model is serving its purpose as expected. This check is necessary.  **Code for creating data pipeline –**   1. Pulling data from server/linux machine –   Step 1: Create HDFS directory –  Delete the directory if already exists –  Hadoop fs -rm -r /apps/database\_name/loandata  Create new directory –  Hadoop fs -mkdir /apps/database\_name/loandata  Step 2: Copy CSV to HDFS –  Hadoop fs -put /home/default/loandata.csv hdfs://apps/database\_name/loandata   1. Pulling data from DB2 database into Hive and HDFS –   Sqoop job for hive: /usr/bin/sqoop import  - -connect “jdbc:db2://server\_name.com:userid”  - -password “\*\*\*\*\*\*\*\*”  - -username “USERNAME”  - - table “DB2\_tablename”  - - hive-import  - -hive-database DATABASE\_NAME  - -hive-table HIVE\_TABLENAME  - - target-dir – “hdfs://path\_of\_target\_directory” -m 1  Sqoop job for HDFS: /usr/bin/sqoop import - -connect “jdbc:db2://server\_name.com:userid”  - -password “\*\*\*\*\*\*\*\*”  - -username “USERNAME”  - - table “DB2\_tablename”  - - target-dir – “hdfs://path\_of\_target\_directory” -m 1  Commands for connecting with hive –   * Hive * Show databases; * Use database\_name; * Show tables;   The above command will show the list of tables in hive database and there would be a table in this list which we have created with the sqoop job in above step. Now since we got the table we can simply run any SQL command to fetch the data as per the requirement. This will work when we load data from DB2/Oracle. For creating table manually, we need to define schema as below –  DROP TABLE loandata;  CREATE SCHEMA IF NOT EXISTS database\_name;  CREATE EXTERNAL TABLE IF NOT EXISTS database.loandata  (term string, loan\_amount int……and so on)  ROW FORMAT DELIMITED  FIELDS TERMINATED BY ‘,’  STORED AS TEXTFILE  LOCATION ‘hdfs://target\_directory\_path’;  Connect with Spark:   * Pyspark * Hive\_context=HiveContext(sc) * loandata = hive\_context.table(“database\_name.loandata”) * loandata.show()   Now we have data frame created in spark and we are ready to jump into machine learning part –  Do initial imports:  import os  import json  import numpy as np  import pandas as pd  import tensor flow as tf  from sklearn import tree  from sklearn import metrics  from sklearn.metrics import precision\_recall\_fscore\_support  from sklearn.externals import joblib  from sklearn.model\_selection import train\_test\_split, GridSearchCV  from sklearn.base import BaseEstimator, TransformerMixin  from sklearn.ensemble import RandomForestClassifier  from sklearn.pipeline import make\_pipeline  import warnings  warnings.filterwarnings("ignore")  Data pre-processing:  # Column Names  list(loandata.columns)  # Size of data  loandata.shape  # Check records in dataframe  loandata.head()  # Summary statistics  loandata.describe()  # Null values  loandata.isnull().sum().value\_counts()  # Variable selection  pred\_var = [‘loan\_status’ , ‘loan\_amnt’ , ‘int\_rate’ , ‘grade’ , ‘emp\_length’ , ‘home\_ownership’ , ‘annual\_inc’ , ‘term’]  # create custom pre-processing estimator so that it can be used in pipeline  Class PreProcessing(BaseEstimator, TransformerMixin):  Def \_ \_init\_ \_(self): pass  Def transform(self, loandata):  pred\_var = [‘loan\_status’ , ‘loan\_amnt’ , ‘int\_rate’ , ‘grade’ , ‘emp\_length’ , ‘home\_ownership’ , ‘annual\_inc’ , ‘term’]  loandata = loandata[pred\_var]  loan\_status.replace({‘Fully Paid’ : Good Loan, ‘Charged Off’ : Bad Loan, ‘Current’ : Good Loan, ‘Default’ : Bad Loan, ‘Late (31-120 days)’ : Bad Loan, ‘In Grace Period’ : Bad Loan, ‘Late (16-30 days)’ : Bad Loan, ‘NMCP Fully Paid’ : Bad Loan, ‘NMCP Charged Off’ : Bad Loan, ‘Issued’ : Good Loan}, inplace=True)  return loandata.as\_matrix()  return self()  # Creating training and test datasets  X\_train, X\_test, y\_train, y\_test = train\_test\_split(data[pred\_var], data[‘loan\_status’], test\_size=0.25)  # Convert train and test set to numpy array:  y\_train = y\_train.replace({'Good Loan':1, 'Bad Loan':0}).as\_matrix()  y\_test = y\_test.replace({'Good Loan':1, 'Bad Loan':0}).as\_matrix()  # Create Random Forest object  randomforestclassifier = RandomForestClassifier(bootstrap=True, criterion='gini',max\_depth=8, max\_features='auto', max\_leaf\_nodes=20,n\_estimators=30, n\_jobs=1)  # Create pipeline  pipe = make\_pipeline(PreProcessing(),RandomForestClassifier())  # Fitting model  Pipe.fit(X\_train, y\_train)  # Predict Output  Pipe.predict(X\_test)  Note: The above code doesn’t have all data exploratory and all data cleaning steps required for the problem. I,ve added some important steps just to show that it can be done in python as well.  **Data Modelling**   * Language/ Modelling Engine: R/ R Studio * Model Used: Logistic regression * Performance measure criteria used: Confusion Matrix, ROC (Reciever Operating Characteristics) curve * Prediction accuracy = 79.3% , Area under Curve (ROC) = 69.3% * Other models used: Decision Tree, Random Forest   Random Forest Model Confusion Matrix:  Confusion Matrix and Statistics  Reference  Prediction 0 1  0 191247 15569  1 2935 875  Accuracy : 0.9121  A screenshot of a cell phone  Description automatically generated  Note : Most of the pre-processing is covered in Part 1 already, Please refer code file to get more idea on feature selection. |

**Conclusion**

*A logistic regression model was used to predict the loan status. Different cut off’s were used to decide if the loan should be granted or not. Cut off of 30% gave a good accuracy of 79.3%. The decision to set a cut off is arbitrary and higher levels of threshold increases the risk. The Area Under Curve also gives a measure of accuracy, which came out to be 69.3%.*

|  |
| --- |
| Part 3: Business AnalysisSQL Queries and Resultant Data In Part 2, I’ve already covered the process of getting data in HDFS and Hive.  Once the data is loaded in hive database in the form of tables, we can query the data using HiveQL which is like SQL.  Below are the SQL queries which are required (table name: loandata) –   1. Assuming the loans with status that are ‘Current’, ‘Issued’ and ‘Fully Paid’ as “Good Loans”, what is the percentage of good loans across each the 36- and 60-month terms.     **Query** –  SELECT GOOD\_LOAN\_36\*100/TOTAL\_CNT\_36 AS GOOD\_LOAN\_PERC\_36  ,GOOD\_LOAN\_60\*100/TOTAL\_CNT\_60 AS GOOD\_LOAN\_PERC\_60  FROM(  SELECT  COUNT(case when term = "36 months" AND (loan\_status = “Current” or loan\_status = “Issued” or loan\_status = “Fully Paid”) then “Good Loans” END) as GOOD\_LOAN\_36,  COUNT(CASE WHEN term = "36 months" THEN 1 END) AS TOTAL\_CNT\_36,  COUNT(case when term = "60 months" AND (loan\_status = “Current” or loan\_status = “Issued” or loan\_status = “Fully Paid”) then “Good Loans” END) as GOOD\_LOAN\_60,  COUNT(CASE WHEN term = "60 months" THEN 1 END) AS TOTAL\_CNT\_60  FROM LOANDATA  )LOANDATA   1. What are the title(s) of employee(s) who took the most loans and least number of loans?   **Query** --FOR MAX LOAN TITLE  SELECT  EMP\_TITLE  FROM  (  SELECT EMP\_TITLE, COUNT(EMP\_TITLE) AS TITLE\_CNT  GROUP BY EMP\_TITLE  )LOANDATA  QUALIFY DENS\_RANK() OVER(PARTITION BY '' ORDER BY TITLE\_CNT DESC) = 1  **Query** --FOR MIN LOAN TITLE  SELECT  EMP\_TITLE  FROM  (  SELECT EMP\_TITLE, COUNT(EMP\_TITLE) AS TITLE\_CNT  GROUP BY EMP\_TITLE  )LOANDATA  QUALIFY DENS\_RANK() OVER(PARTITION BY '' ORDER BY TITLE\_CNT) = 1   1. What is the most common purpose of the loans that are considered “Bad Loans” (please use definition mentioned for “Good Loans” in #1 above).   **Query –**  SELECT  PUPOSE  FROM(  SELECT  PURPOSE  ,COUNT(case when loan\_status <> “Current” AND loan\_status <> “Issued” AND loan\_status <> “Fully Paid” then “BAD Loans” END) as BAD\_LOAN  FROM LOANDATA  GROUP BY PURPOSE  )LOANDATA  QUALIFY DENS\_RANK() OVER(PARTITION BY '' ORDER BY BAD\_LOAN DESC) = 1  Note: Since, I don’t have access to any data warehouse to load personal data I have provided only the SQL queries and not the result from these queries. |
|  |
|  |