# Multinomial Regression

**Instructions:**

Please share your answers filled in-line in the word document. Submit code separately wherever applicable.

Please ensure you update all the details:

**Name: Vishvash C Batch ID:** 23012024

**Topic: Multinomial Regression.**

**Guidelines:**

**1. An assignment submission is considered complete only when correct and executable code(s) are submitted along with the documentation explaining the method and results. Failing to submit either of those will be considered an invalid submission and will not be considered as correct submission.**

**2. Ensure that you submit your assignments correctly and in full. Resubmission is not allowed.**

**3. Post the submission you can evaluate your work by referring to keys provided. (will be available only post the submission).**

**Hints:**

1. **Business Problem**
   1. **What is the business objective?**
   2. **Are there any constraints?**
2. **Work on each feature of the dataset to create a data dictionary as displayed in the below image:**

**Make a table as shown above and provide information about the features such as its data type and its relevance to the model building. And if not relevant, provide reasons and a description of the feature.**

**Using Python codes perform:**

1. **Data Pre-processing**

**3.1 Data Cleaning, Feature Engineering, etc.**

**3.2 Outlier treatment.**

1. **Exploratory Data Analysis (EDA):**
   1. **Summary.**
   2. **Univariate analysis.**
   3. **Bivariate analysis.**
2. **Model Building**
   1. **Build the model on the scaled data (try multiple options).**
   2. **Build a Multinomial Regression model.**
   3. **Train and test the model and compare accuracies by confusion matrix, ROC & AUC curves.**
   4. **Briefly explain the model output in the documentation.**
3. **Write about the benefits/impact of the solution - in what way does the business (client) benefit from the solution provided?**

**Problem Statement:**

1. You work for a consumer finance company that specializes in lending loans to urban customers. When the company receives a loan application, the company has to make a decision for loan approval based on the applicant’s profile. Two types of risks are associated with the bank’s decision:

* If the applicant is likely to repay the loan, then not approving the loan results in a loss of business to the company
* If the applicant is not likely to repay the loan, i.e. he/she is likely to default, then approving the loan may lead to a financial loss for the company

The data given below contains information about past loan applicants and whether they ‘defaulted’4 or not. The aim is to identify patterns that indicate if a person is likely to default, which may be used for taking actions such as denying the loan, reducing the amount of the loan, lending (for risky applicants) at a higher interest rate, etc.

In this case study, you will use EDA to understand how consumer attributes and loan attributes influence the tendency of default.

When a person applies for a loan, there are two types of decisions that could be taken by the company:

1. Loan accepted: If the company approves the loan, there are 3 possible scenarios described below:

* Fully paid: Applicant has fully paid the loan (the principal and the interest rate)
* Current: Applicant is in the process of paying the installments, i.e., the tenure of the loan is not yet completed. These candidates are not labeled as 'defaulted'.
* Charged-off: Applicant has not paid the installments in due time for a long period of time, i.e. he/she has defaulted on the loan

2. Loan rejected: The company had rejected the loan (because the candidate does not meet their requirements etc.). Since the loan was rejected, there is no transactional history of those applicants with the company and so this data is not available with the company (and thus in this dataset)

Like most other lending companies, lending loans to ‘risky’ applicants is the largest source of financial loss (called credit loss). Credit loss is the amount of money lost by the lender when the borrower refuses to pay or runs away with the money owed. In other words, borrowers who default cause the largest amount of loss to the lenders. In this case, the customers labeled as 'charged-off' are the 'defaulters'.

If one can identify these risky loan applicants, then such loans can be reduced thereby cutting down the amount of credit loss.

In other words, the company wants to understand the driving factors (or driver variables) behind loan default, i.e. the variables which are strong indicators of default. The company can utilize this knowledge for its portfolio and risk assessment.

Perform Multinomial regression on the dataset in which loan\_status is the output (Y) variable and it has three levels in it.

A screenshot of a cell phone

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **Name of Feature** | **Description** | **Type** | **Relevance** |
| id | A unique identifier for the loan | Nominal | Irrelevant |
| member\_id | A unique identifier for the borrower | Nominal | Irrelevant |
| loan\_amnt | The listed amount of the loan applied for | Quantitative | Relevant |
| funded\_amnt | The total amount committed to that loan | Quantitative | Relevant |
| funded\_amnt\_inv | The total amount committed by investors | Quantitative | Relevant |
| term | The number of payments on the loan | Nominal | Relevant |
| int\_rate | Interest rate on the loan | Quantitative | Relevant |
| installment | Monthly payment owed by the borrower | Quantitative | Relevant |
| grade | LC assigned loan grade | Nominal | Irrelevant |
| sub\_grade | LC assigned loan subgrade | Nominal | Irrelevant |
| emp\_title | The job title supplied by the Borrower | Nominal | Irrelevant |
| emp\_length | Employment length in years | Quantitative | Relevant |
| home\_ownership | The home ownership status of the borrower | Nominal | Irrelevant |
| annual\_inc | The self-reported annual income provided by the borrower during registration | Quantitative | Relevant |
| verification\_status | Indicates if income was verified by LC | Nominal | Irrelevant |
| issue\_d | The month the loan was funded | Nominal | Irrelevant |
| loan\_status | Current status of the loan | Nominal | Relevant |
| pymnt\_plan | Indicates if a payment plan has been put in place | Nominal | Irrelevant |
| url | URL for the LC page with listing data | Nominal | Irrelevant |
| desc | Loan description provided by the borrower | Nominal | Irrelevant |
| purpose | A category provided by the borrower for the loan request | Nominal | Irrelevant |
| title | The loan title provided by the borrower | Nominal | Irrelevant |
| zip\_code | The first 3 numbers of the zip code provided by the borrower | Nominal | Irrelevant |
| addr\_state | The state provided by the borrower in the loan application | Nominal | Irrelevant |
| dti | A ratio calculated using the borrower’s total monthly debt payments on the total debt obligations, excluding mortgage and the requested LC loan, divided by the borrower’s self-reported monthly income | Quantitative | Relevant |
| delinq\_2yrs | The number of 30+ days past-due incidences of delinquency in the borrower's credit file for the past 2 years | Quantitative | Relevant |
| earliest\_cr\_line | The month the borrower's earliest reported credit line was opened | Nominal | Irrelevant |
| inq\_last\_6mths | The number of inquiries in the past 6 months | Quantitative | Relevant |
| mths\_since\_last\_delinq | The number of months since the borrower's last delinquency | Quantitative | Relevant |
| mths\_since\_last\_record | The number of months since the last public record | Quantitative | Relevant |
| open\_acc | The number of open credit lines in the borrower's credit file | Quantitative | Relevant |
| pub\_rec | Number of derogatory public records | Quantitative | Relevant |
| revol\_bal | Total credit revolving balance | Quantitative | Relevant |
| revol\_util | Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit | Quantitative | Relevant |
| total\_acc | The total number of credit lines currently in the borrower's credit file | Quantitative | Relevant |
| initial\_list\_status | The initial listing status of the loan | Nominal | Irrelevant |
| out\_prncp | Remaining outstanding principal for total amount funded | Quantitative | Relevant |
| out\_prncp\_inv | Remaining outstanding principal for portion of total amount funded by investors | Quantitative | Relevant |
| total\_pymnt | Payments received to date for total amount funded | Quantitative | Relevant |
| total\_pymnt\_inv | Payments received to date for portion of total amount funded by investors | Quantitative | Relevant |
| total\_rec\_prncp | Principal received to date | Quantitative | Relevant |
| total\_rec\_int | Interest received to date | Quantitative | Relevant |
| total\_rec\_late\_fee | Late fees received to date | Quantitative | Relevant |
| recoveries | Post charge-off gross recovery | Quantitative | Relevant |
| collection\_recovery\_fee | Post charge-off collection fee | Quantitative | Relevant |
| last\_pymnt\_d | Last month payment was received | Nominal | Irrelevant |
| last\_pymnt\_amnt | Last total payment amount received | Quantitative | Relevant |
| next\_pymnt\_d | Next scheduled payment date | Nominal | Irrelevant |
| last\_credit\_pull\_d | The most recent month LC pulled credit for this loan | Nominal | Irrelevant |
| collections\_12\_mths\_ex\_med | Number of collections in 12 months excluding medical collections | Quantitative | Irrelevant |
| mths\_since\_last\_major\_derog | Months since most recent 90-day or worse rating | Quantitative | Irrelevant |
| policy\_code | Publicly available policy\_code=1 new products not publicly available policy\_code=2 | Nominal | Irrelevant |
| application\_type | Indicates whether the loan is an individual application or a joint application | Nominal | Irrelevant |
| annual\_inc\_joint | The combined self-reported annual income provided by the co-borrowers during registration | Quantitative | Irrelevant |
| dti\_joint | A ratio calculated using the co-borrowers' total monthly payments on the total debt obligations, excluding mortgages and the requested LC loan, divided by the co-borrowers' combined self-reported monthly income | Quantitative | Irrelevant |
| verification\_status\_joint | Indicates if the co-borrowers' joint income was verified by LC | Nominal | Irrelevant |
| acc\_now\_delinq | The number of accounts on which the borrower is now delinquent | Quantitative | Irrelevant |
| tot\_coll\_amt | Total collection amounts ever owed | Quantitative | Irrelevant |
| tot\_cur\_bal | Total current balance of all accounts | Quantitative | Irrelevant |
| open\_acc\_6m | Number of open trades in last 6 months | Quantitative | Irrelevant |
| open\_il\_6m | Number of currently active installment trades | Quantitative | Irrelevant |
| open\_il\_12m | Number of installment accounts opened in past 12 months | Quantitative | Irrelevant |
| open\_il\_24m | Number of installment accounts opened in past 24 months | Quantitative | Irrelevant |
| mths\_since\_rcnt\_il | Months since most recent installment accounts opened | Quantitative | Irrelevant |
| total\_bal\_il | Total current balance of all installment accounts | Quantitative | Irrelevant |
| il\_util | Ratio of total current balance to high credit/credit limit on all install acct | Quantitative | Irrelevant |
| open\_rv\_12m | Number of revolving trades opened in past 12 months | Quantitative | Irrelevant |
| open\_rv\_24m | Number of revolving trades opened in past 24 months | Quantitative | Irrelevant |
| max\_bal\_bc | Maximum current balance owed on all revolving accounts | Quantitative | Irrelevant |
| all\_util | Balance to credit limit on all trades | Quantitative | Irrelevant |
| total\_rev\_hi\_lim | Total revolving high credit/credit limit | Quantitative | Irrelevant |
| inq\_fi | Number of personal finance inquiries | Quantitative | Irrelevant |
| total\_cu\_tl | Number of finance trades | Quantitative | Irrelevant |
| inq\_last\_12m | Number of credit inquiries in past 12 months | Quantitative | Irrelevant |
| acc\_open\_past\_24mths | Number of trades opened in past 24 months | Quantitative | Irrelevant |
| avg\_cur\_bal | Average current balance of all accounts | Quantitative | Irrelevant |
| bc\_open\_to\_buy | Total open to buy on revolving bankcards or lines | Quantitative | Irrelevant |
| bc\_util | Ratio of total current balance to high credit/credit limit for all bankcard accounts | Quantitative | Irrelevant |
| chargeoff\_within\_12\_mths | Number of charge-offs within 12 months | Quantitative | Irrelevant |
| delinq\_amnt | The past-due amount owed for the accounts on which the borrower is now delinquent | Quantitative | Irrelevant |
| mo\_sin\_old\_il\_acct | Months since oldest bank installment account opened | Quantitative | Irrelevant |
| mo\_sin\_old\_rev\_tl\_op | Months since oldest revolving account opened | Quantitative | Irrelevant |
| mo\_sin\_rcnt\_rev\_tl\_op | Months since most recent revolving account opened | Quantitative | Irrelevant |
| mo\_sin\_rcnt\_tl | Months since most recent account opened | Quantitative | Irrelevant |
| mort\_acc | Number of mortgage accounts | Quantitative | Irrelevant |
| mths\_since\_recent\_bc | Months since most recent bankcard account opened | Quantitative | Irrelevant |
| mths\_since\_recent\_bc\_dlq | Months since most recent bankcard delinquency | Quantitative | Irrelevant |
| mths\_since\_recent\_inq | Months since most recent inquiry | Quantitative | Irrelevant |
| mths\_since\_recent\_revol\_delinq | Months since most recent revolving delinquency | Quantitative | Irrelevant |
| num\_accts\_ever\_120\_pd | Number of accounts ever 120 or more days past due | Quantitative | Irrelevant |
| num\_tl\_120dpd\_2m | Number of accounts currently 120 days past due (updated in past 2 months) | Quantitative | Irrelevant |
| num\_tl\_30dpd | Number of accounts currently 30 days past due | Quantitative | Irrelevant |
| num\_tl\_90g\_dpd\_24m | Number of accounts 90 or more days past due in last 24 months | Quantitative | Irrelevant |
| num\_tl\_op\_past\_12m | Number of accounts opened in past 12 months | Quantitative | Irrelevant |
| pct\_tl\_nvr\_dlq | Percent of trades never delinquent | Quantitative | Irrelevant |
| percent\_bc\_gt\_75 | Percentage of all bankcard accounts > 75% of limit | Quantitative | Irrelevant |
| pub\_rec\_bankruptcies | Number of public record bankruptcies | Quantitative | Irrelevant |
| tax\_liens | Number of tax liens | Quantitative | Irrelevant |
| tot\_hi\_cred\_lim | Total high credit/credit limit | Quantitative | Irrelevant |
| total\_bal\_ex\_mort | Total credit balance excluding mortgage | Quantitative | Irrelevant |
| total\_bc\_limit | Total bankcard high credit/credit limit | Quantitative | Irrelevant |
| total\_il\_high\_credit\_limit | Total installment high credit/credit limit | Quantitative | Irrelevant |

**Code:**

'''

Business objective: Minimize the defaulters

Business constraints: Identify the repayabel amount by defaulters

Business success criteria: Reduce the proportion of loan defaults to less than 5%

ML success criteria: Achieve the accurace of at least 70 percent

Economic success criteria: Increaset the return on investment atleast by 10%

'''

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split # train and test

from sklearn.compose import ColumnTransformer

from sklearn.pipeline import Pipeline

from feature\_engine.outliers import Winsorizer

from sklearn.impute import SimpleImputer

from sklearn.preprocessing import MinMaxScaler

from sklearn.metrics import accuracy\_score # confusion\_matrix

import pickle, joblib

from sklearn.linear\_model import LogisticRegression

from sklearn.model\_selection import GridSearchCV

mode = pd.read\_csv(r"C:/Users/Lenovo/Downloads/Study material/Data Science/Multinomial Regression/Assignments/Multinomial Regression Assignment/loan\_refined.csv").convert\_dtypes()

mode = mode.head(1000)

mode.head(10)

mode.info()

mode.describe()

mode.loan\_status.value\_counts()

mode.drop(columns = 'collection\_recovery\_fee', inplace = True) #due to high outliers

mode.boxplot(by = "loan\_status" , sharey = False, figsize = (15, 8))

plt.subplots\_adjust(wspace=0.5, hspace=0.5)

# Scatter plot between each possible pair of independent variable and also histogram for each independent variable

# sns.pairplot(mode, hue = "loan\_status") # With showing the category of each car loan\_status in the scatter plot

# sns.pairplot(mode)

# Correlation values between each independent features

a = mode.iloc[:,1:].corr()

##########################

## Auto EDA ##

import dtale

import pandas as pd

d = dtale.show(mode)

d.open\_browser()

##########################

mode.info()

# Predictors

X = mode.iloc[:,1:]

# Target

Y = mode.iloc[:,[0]]

X.info()

# Numeric input features

numeric\_features = X.select\_dtypes(exclude = ['object']).columns

num\_pipeline1 = Pipeline(steps = [('impute1', SimpleImputer(strategy = 'mean'))])

# Imputation Transformer

preprocessor = ColumnTransformer([('mean', num\_pipeline1, numeric\_features)])

print(preprocessor)

# Fit the data to pipeline

impute\_data = preprocessor.fit(X)

# Save the pipeline

joblib.dump(impute\_data, 'impute')

# Transform the data

X1 = pd.DataFrame(impute\_data.transform(X), columns = X.columns)

X1.isna().sum()

# Outlier Analysis

# Multiple boxplots in a single visualization.

# pandas plot() function with parameters kind = 'box' and subplots = True

X1.plot(kind = 'box', subplots = True, sharey = False, figsize = (15, 8))

'''sharey True or 'all': x- or y-axis will be shared among all subplots.

False or 'none': each subplot x- or y-axis will be independent.'''

# increase spacing between subplots

plt.subplots\_adjust(wspace = 0.75) # ws is the width of the padding between subplots, as a fraction of the average Axes width.

plt.show()

# from feature\_engine.outliers import Winsorizer

winsor = Winsorizer(capping\_method = 'gaussian', # choose IQR rule boundaries or gaussian for mean and std

tail = 'both', # cap left, right or both tails

fold = 1,

variables = list(X1.columns))

outlier\_pipeline = Pipeline(steps=[('winsor', winsor)])

preprocessor1 = ColumnTransformer(transformers = [('wins', outlier\_pipeline, numeric\_features)], remainder = 'passthrough')

print(preprocessor1)

# Train the pipeline

winz\_data = preprocessor1.fit(X1)

# Save the pipeline

joblib.dump(winz\_data, 'winzor')

# X2 = X1

# X1 = X2

X1 = pd.DataFrame(winz\_data.transform(X1), columns=X1.columns)

# X1=winz\_data.transform(X1)

X1.info()

X1.describe()

# Boxplot

X1.plot(kind = 'box', subplots = True, sharey = False, figsize = (15, 8))

# increase spacing between subplots

plt.subplots\_adjust(wspace = 0.75) # ws is the width of the padding between subplots, as a fraction of the average Axes width.

plt.show()

# Minmax Scaler

scale\_pipeline = Pipeline(steps = [('scale', MinMaxScaler())])

preprocessor2 = ColumnTransformer(transformers = [('scale', scale\_pipeline, numeric\_features)], remainder = 'passthrough')

print(preprocessor2)

scale = preprocessor2.fit(X1)

# Save the pipeline

joblib.dump(scale, 'scale')

X2 = pd.DataFrame(scale.transform(X1), columns = X1.columns)

X2

# Data Partitioning

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X2, Y, test\_size = 0.2,

random\_state = 0,

stratify = Y)

# ‘multinomial’ option is supported only by the ‘lbfgs’ and ‘newton-cg’ solvers

logmodel = LogisticRegression(multi\_class = "multinomial", solver = "newton-cg")

# Train the model

model = logmodel.fit(X\_train, Y\_train)

# Train accuracy

train\_predict = model.predict(X\_train) # Train predictions

accuracy\_score(Y\_train, train\_predict)

# Predict the results for Test Data

test\_predict = model.predict(X\_test) # Test predictions

# Test accuracy

accuracy\_score(Y\_test, test\_predict)

# Hyperparameter Optimization

logmodel1 = LogisticRegression(multi\_class = "multinomial")

param\_grid = [

{'penalty' : ['l1', 'l2', 'elasticnet', 'none'],

'C' : np.logspace(-4, 4, 20),

'solver' : ['lbfgs','newton-cg','liblinear','sag','saga'],

'max\_iter' : [100, 1000,2500, 5000]

}

]

# from sklearn.model\_selection import GridSearchCV

clf = GridSearchCV(logmodel1, param\_grid = param\_grid, cv = 3, verbose=True, n\_jobs=-1)

best\_clf = clf.fit(X\_train, Y\_train)

# Best estimator

best\_clf.best\_estimator\_

print (f'Accuracy - : {best\_clf.score(X\_train, Y\_train):.3f}')

print (f'Accuracy - : {best\_clf.score(X\_test, Y\_test):.3f}')

# Y1 = np.ravel(Y)

# Fitting on Full data

best\_clf1 = clf.fit(X2, Y)

best\_clf1.best\_estimator\_

print (f'Accuracy - : {best\_clf1.score(X2, Y):.3f}')

print (f'Accuracy - : {best\_clf1.score(X\_test, Y\_test):.3f}')

# Save the best Model

pickle.dump(best\_clf1, open('multinomial.pkl', 'wb'))

############################

# Predictions on New Data

model = pickle.load(open('multinomial.pkl', 'rb'))

impute = joblib.load('impute')

winzor = joblib.load('winzor')

minmax = joblib.load('scale')

data = pd.read\_excel(r"C:/Users/Lenovo/Downloads/Study material/Data Science/Multinomial Regression/Assignments/Multinomial Regression Assignment/loan\_test.xlsx")

data.drop(columns = 'collection\_recovery\_fee', inplace = True)

clean = pd.DataFrame(impute.transform(data), columns = data.columns)

clean1 = pd.DataFrame(winzor.transform(clean), columns = data.columns)

clean3 = pd.DataFrame(minmax.transform(clean1), columns = data.columns)

prediction = pd.DataFrame(model.predict(clean3), columns = ['loan\_status'])

prediction

**Output:**

mode.head(10)

Out[102]:

loan\_amnt funded\_amnt ... collection\_recovery\_fee last\_pymnt\_amnt

0 5000 5000 ... 0.00 171.62

1 2500 2500 ... 1.11 119.66

2 2400 2400 ... 0.00 649.91

3 10000 10000 ... 0.00 357.48

4 3000 3000 ... 0.00 67.79

5 5000 5000 ... 0.00 161.03

6 7000 7000 ... 0.00 1313.76

7 3000 3000 ... 0.00 111.34

8 5600 5600 ... 2.09 152.39

9 5375 5375 ... 2.52 121.45

[10 rows x 29 columns]

mode.loan\_status.value\_counts()

Out[103]:

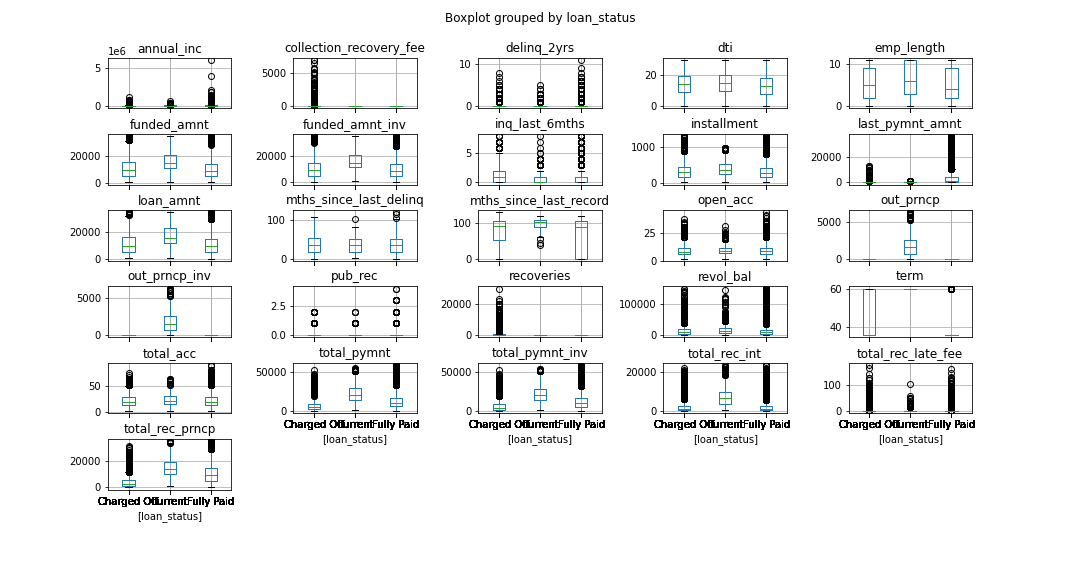
loan\_status

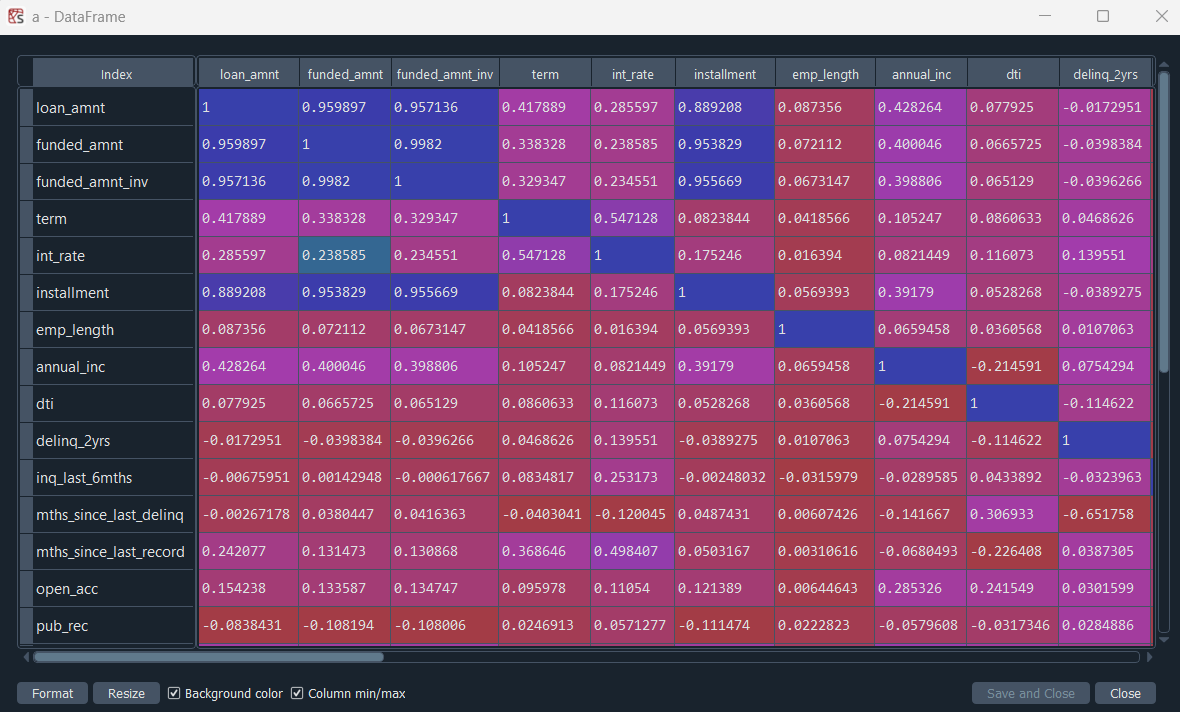
Fully Paid 32950

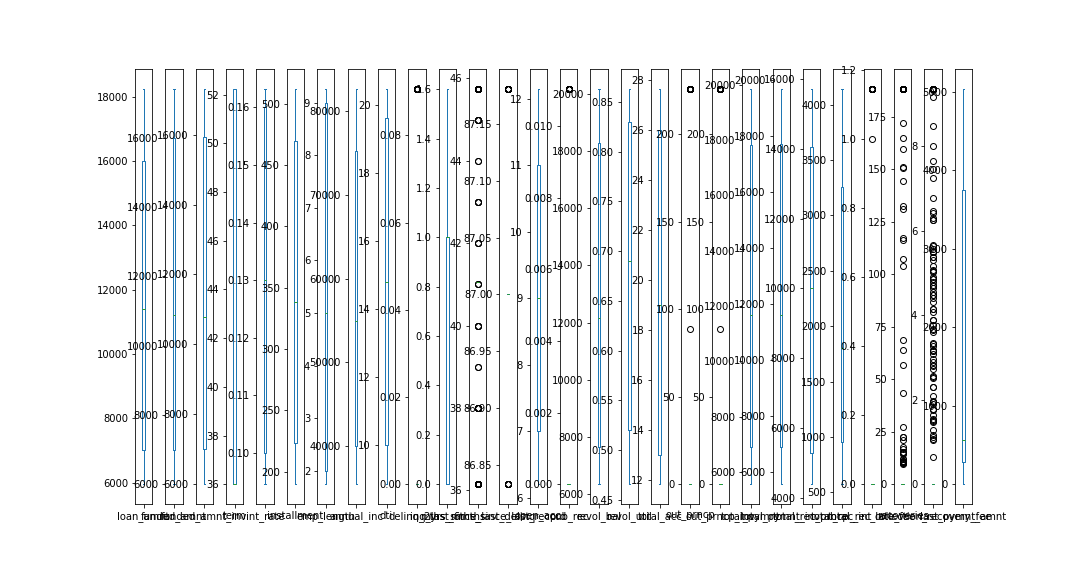
Charged Off 5627

Current 1140

Name: count, dtype: int64







Removed “collection\_recovery\_fee” column due to high outliers

X2

Out[168]:

loan\_amnt funded\_amnt ... recoveries last\_pymnt\_amnt

0 0.000000 0.000000 ... 0.000000 0.034079

1 0.000000 0.000000 ... 0.622223 0.023761

2 0.000000 0.000000 ... 0.000000 0.129055

3 0.329552 0.350186 ... 0.000000 0.070986

4 0.000000 0.000000 ... 0.000000 0.013461

.. ... ... ... ... ...

995 0.522665 0.558414 ... 0.000000 1.000000

996 1.000000 1.000000 ... 0.000000 1.000000

997 0.654794 0.700886 ... 0.000000 1.000000

998 0.654794 0.700886 ... 0.000000 0.942477

999 0.000000 0.000000 ... 0.000000 0.642700

[1000 rows x 27 columns]

accuracy\_score(Y\_train, train\_predict)

**Out[175]: 0.9825**

# Predict the results for Test Data

test\_predict = model.predict(X\_test) # Test predictions

# Test accuracy

accuracy\_score(Y\_test, test\_predict)

**Out[179]: 0.965**

best\_clf.best\_estimator\_

Out[184]:

LogisticRegression(C=4.281332398719396, max\_iter=1000,

multi\_class='multinomial', penalty='l1', solver='saga')

print (f'Accuracy - : {best\_clf.score(X\_train, Y\_train):.3f}')

**Accuracy - : 0.991**

print (f'Accuracy - : {best\_clf.score(X\_test, Y\_test):.3f}')

**Accuracy - : 0.980**

best\_clf1.best\_estimator\_

Out[189]:

LogisticRegression(C=78.47599703514607, max\_iter=5000,

multi\_class='multinomial', penalty='l1', solver='saga')

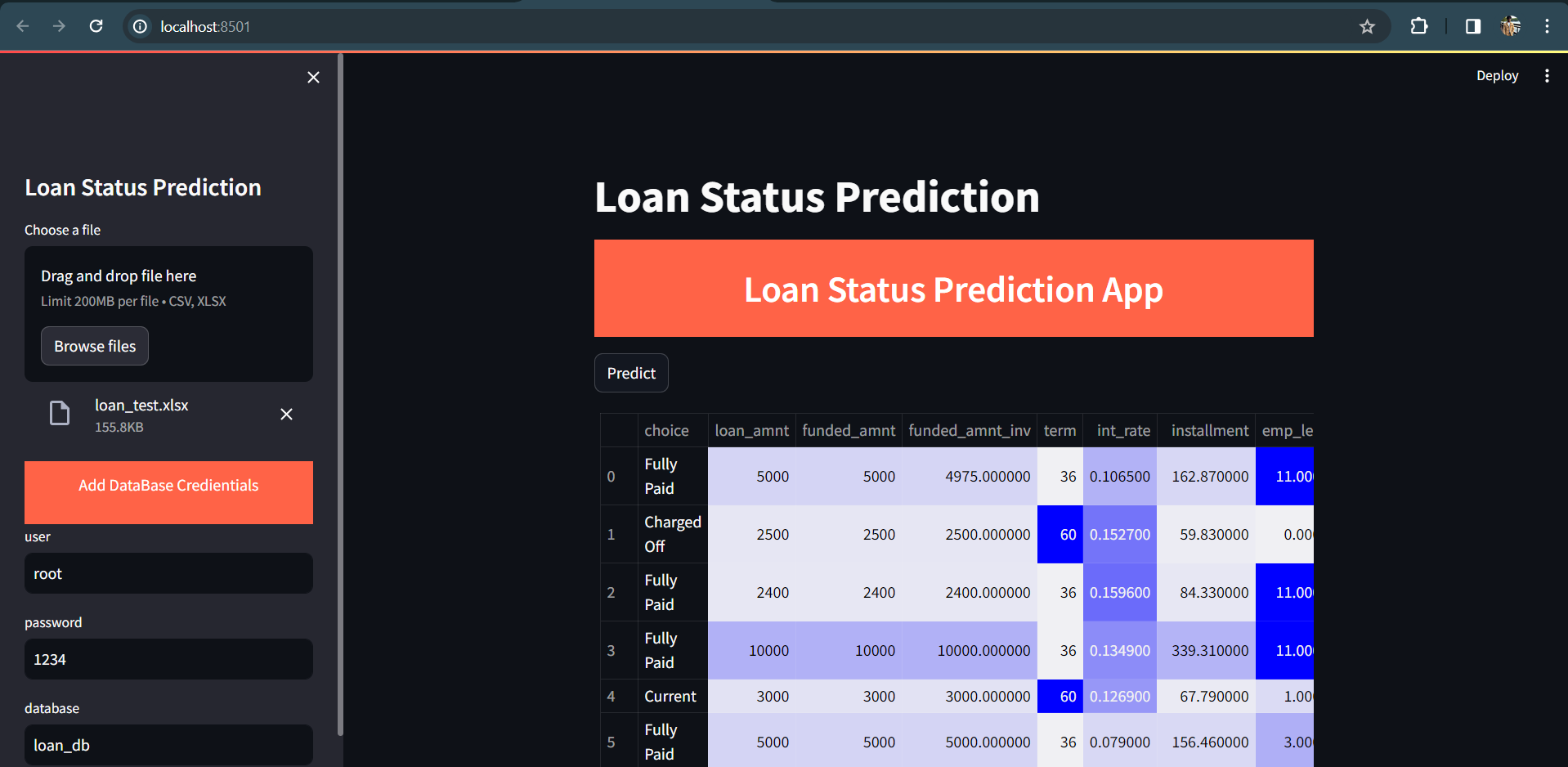
print (f'Accuracy - : {best\_clf1.score(X2, Y):.3f}')

**Accuracy - : 0.993**

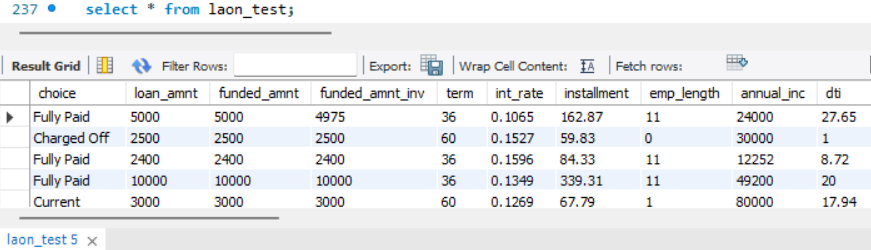
print (f'Accuracy - : {best\_clf1.score(X\_test, Y\_test):.3f}')

**Accuracy - : 0.990**

**Deployment of Multinomial regression problem using streamlit**



**Saving the predicted values in MySQL for monitoring**

****