Prediction Model on a Revolving Credit Line

BY : SAKET NANDAN

Business Problem:

- Revolving Credit means you're borrowing against a Line of Credit.

- Users can borrow the amount of credit allowed to use each month known as *“Credit line or Credit Limit”*.

- Similar to a **Credit card** with the only difference being *“Lesser Interest Rates and Secured Business Assets”*.

- At the end of each statement period, a Bill gets generated.
  If not paid fully, the balance is carried over, or revolved over to the next month along with Interest incurred on the remaining balance.

- *As you pay down the balance, more of your credit line becomes available.*

Objective:

“To predict the revolving balance maintained by each customer to

Exploratory Data Analysis (EDA)

EXCELR
Raising Excellence

Numerical Type

mths_since_last_record	0.845553
mths_since_last_major_derog	0.750160
mths_since_last_delinq	0.511971
tot_curr_bal	0.079195
tot_colle_amt	0.079195
collections_12_mths_ex_med	0.000163
inq_last_6mths	0.000033
acc_now_delinq	0.000033
delinq_2yrs	0.000033
total_credits	0.000033
pub_rec	0.000033
numb_credit	0.000033
annual_inc	0.000005

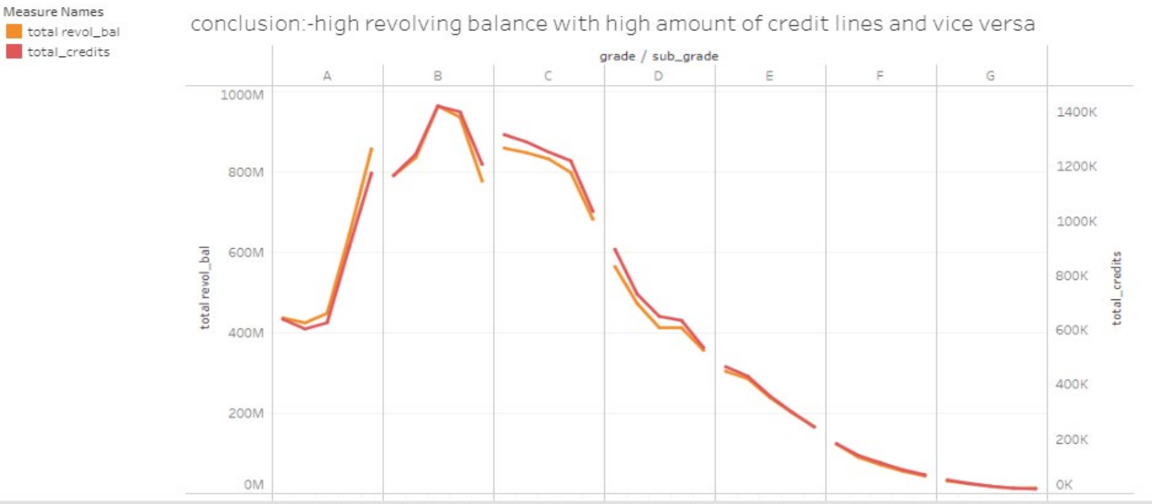
MISSING VALUES???

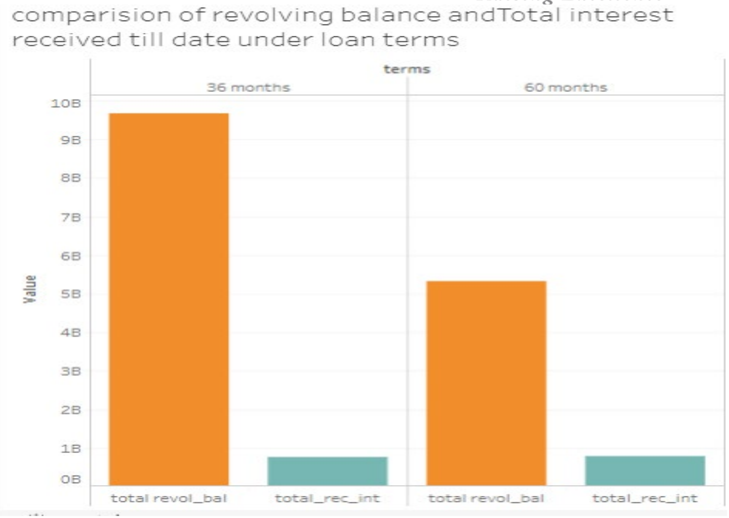
Categorical Type

verification_status_joint	0.999424
Emp_designation	0.057993
Experience	0.050514

- **36 months** in *'terms'* column has **70%** data compared to **60 months**.
- **33%** of the customers have over **10 years of Experience**.
- **50%** of *'home_ownership'* have **MORTGAGED** while **40%** of them are on **RENT**.
- **Almost 60%** of loans taken cater the *purpose* of *'debt_consolidation'*.
- **15% (MAX)** of customers are based out of **California State** while only **12 users(MIN)** are from **Idaho state**.
- **80%** of the customers **DON'T HAVE** *delinquent accounts for a span of 2 years* while **56%** customers **DON'T HAVE** *delinquent accounts since 6 months*.
- **Almost 80%** of *'tot_colle_amt '(total collection amount ever owed)* is **ZERO** dollars.
- *'mths_since_last_record'* ,*'mths_since_last_major_derog'* and *'verification_status_joint'* have **LARGEST** amount of *missing values* – **84%, 75% and 99.9%** respectively.

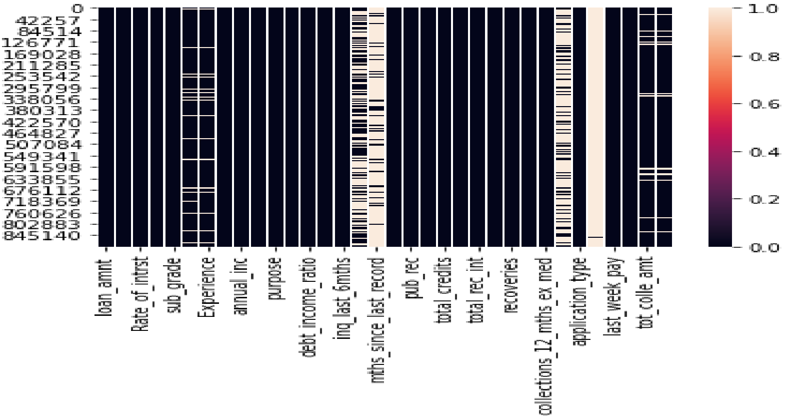
EDA(VISUALIZATION)



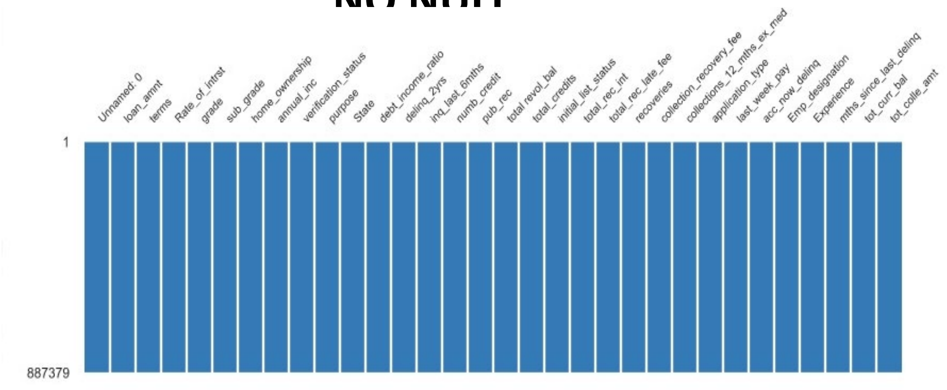


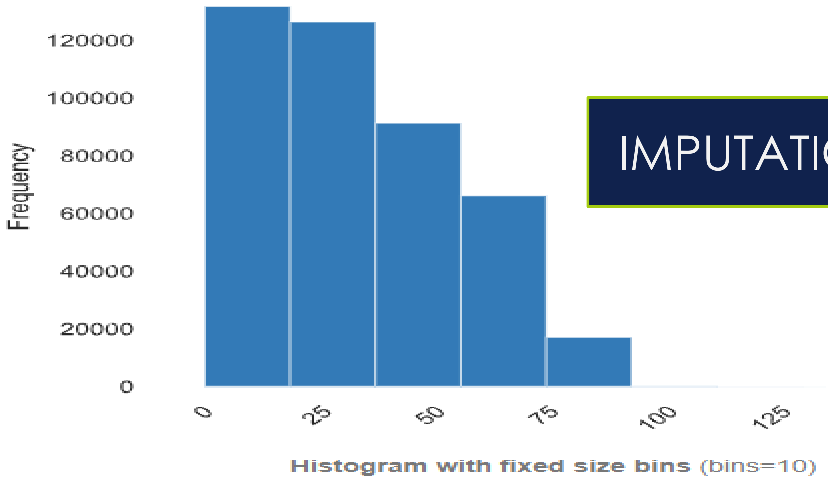
EDA(VISUALIZATION) Contd..

NULL VALUES

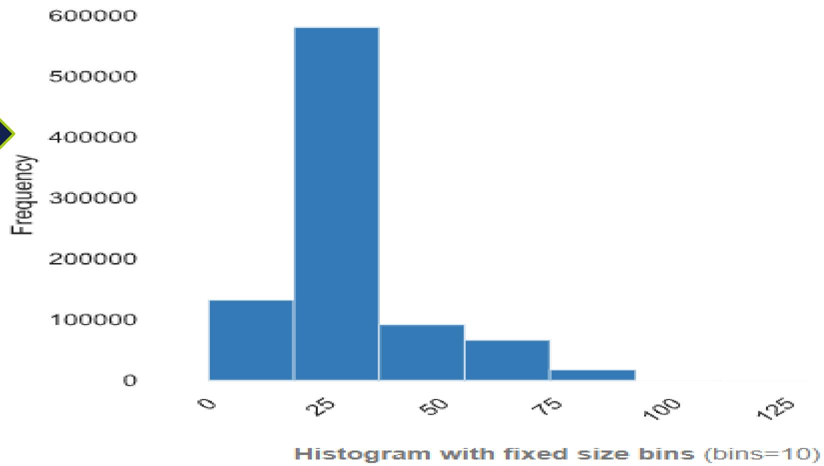


NO NULL

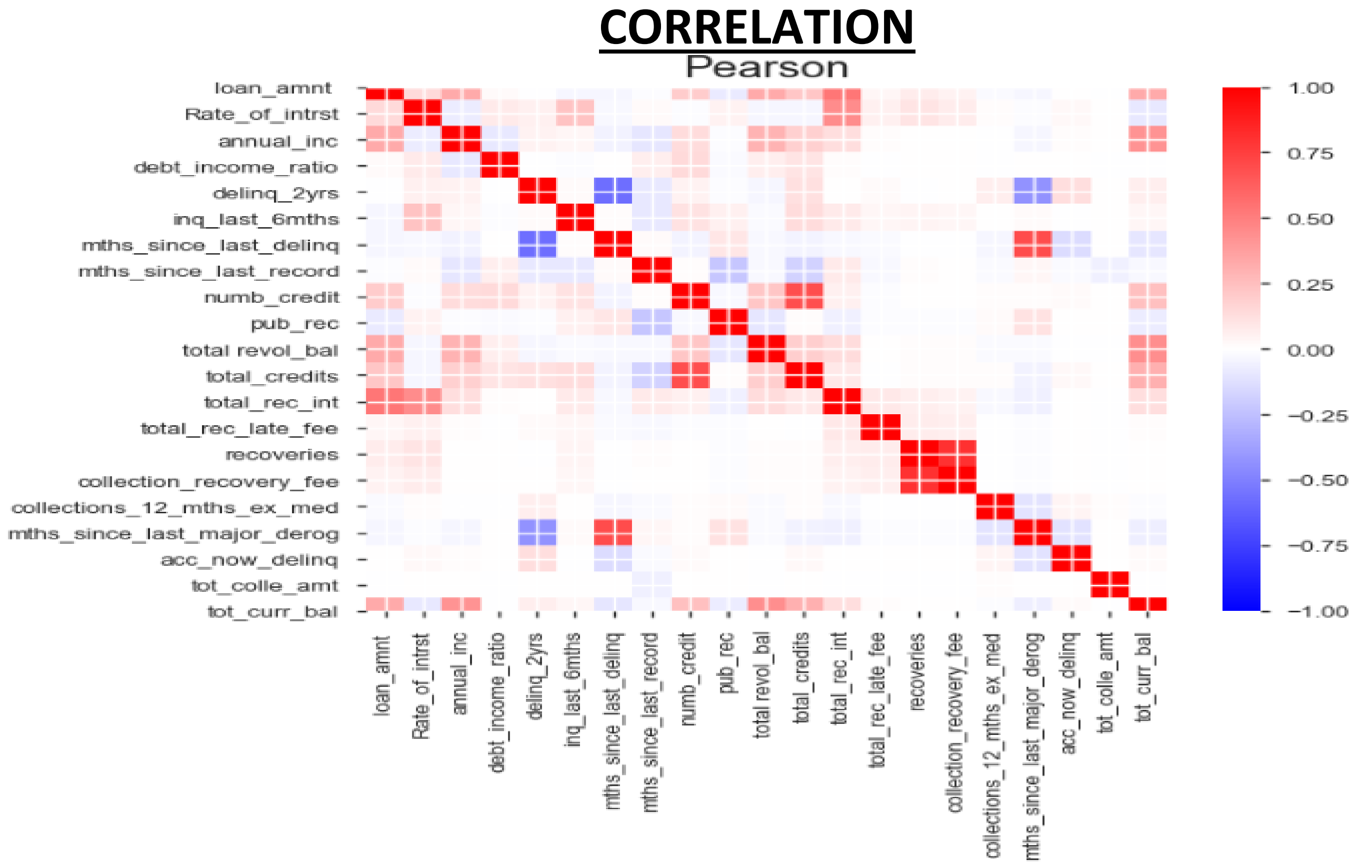




IMPUTATION



EDA(VISUALIZATION) Contd..



Model Building

Model Building

Linear Regression

Dataset Type	Test Size	Random State	R-squared	RMSE
NULL	0.3	0	0.261	18947.63
NOT NULL	0.3	0	0.261	18851.73

Model Building

Random Forest

Data Type	Model Type	Test Size	Random State	R-squared (Train)	R-squared (Test)	RMSE (Train)	RMSE (Test)
NOT NULL	Only Numerical	0.3	0	0.66	0.33	13296.07	17496.04
NOT NULL	Only Numerical (Feature Importance)	0.3	0	0.804	0.33	10092.98	17565.39
NOT NULL	Both Numerical& Categorical	0.3	0	0.805	0.33	10061.32	17574.13
NOT NULL	Both Numerical& Categorical (Feature Importance)	0.3	0	0.814	0.326	9842.85	17629.99

Model Building

XGBoost

Dataset Type	Encoding	Min. Child Weight	Max Depth	Learning Rate	Gamma	R-squared (Train)	R-squared (Test)	RMSE (Train)	RMSE (Test)
NOT NULL	Label Encoding	3	6	0.15	0.4	0.55	0.35	15162.58	17242.74
NOT NULL	Frequency Encoding	5	3	0.08	0.1	0.43	0.35	17121.70	17260.07

Model Selection

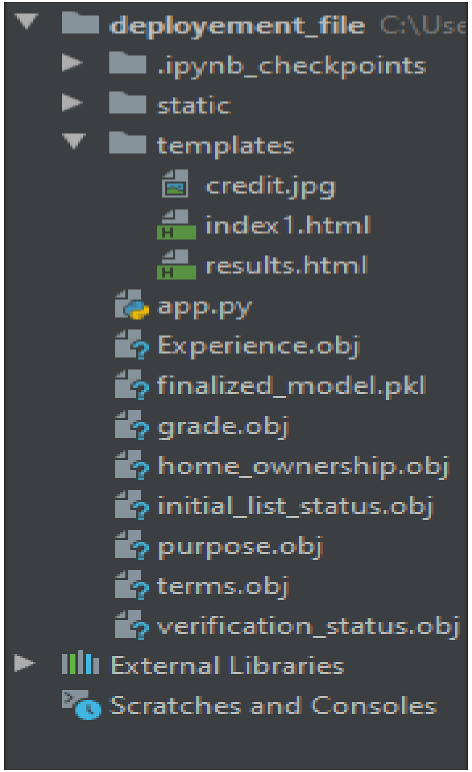
VARIABLES	**PREDICTORS**	'loan_amnt ', 'terms', 'Rate_of_intrst', 'grade', 'home_ownership','annual_inc', 'verification_status', 'purpose', 'debt_income_ratio','delinq_2yrs', 'inq_last_6mths', 'numb_credit', 'pub_rec', 'total_credits', 'initial_list_status','total_rec_int', 'total_rec_late_fee', 'recoveries','Experience','mths_since_last_delinq', 'tot_curr_bal', 'tot_colle_amt']]
	TARGET	['total_revol_bal']
TRAIN/TEST	Test Size	0.2
	Random State	2

XGBoost Model

Dataset Type	Encoding	Min. Child Weight	Max Depth	Learning Rate	Gamma	R-squared (Train)	R-squared (Test)	RMSE (Train)	RMSE (Test)
NOT NULL	Frequency Encoding	5	8	0.05	0.2	0.51	0.44	15644.92	17016.71

Model Deployment using Flask

Folder structure of our deployment folder



Codes on the pycharm where our app will start from the @app.route('/')

```
import pickle
import numpy as np
from flask import Flask, request, render_template
import xgboost as xgb

app = Flask(__name__)

terms = pickle.load(open('terms.obj','rb'))
grade = pickle.load(open('grade.obj','rb'))
home_ownership = pickle.load(open('home_ownership.obj','rb'))
verification_status = pickle.load(open('verification_status.obj','rb'))
purpose = pickle.load(open('purpose.obj','rb'))
initial_list_status = pickle.load(open('initial_list_status.obj','rb'))
Experience = pickle.load(open('Experience.obj','rb'))
model = pickle.load(open('finalized_model.pkl','rb'))


@app.route('/')
def home():
    return render_template('index1.html')


@app.route('/predict',methods=['POST'])


def predict():
```

Here , whatever input we r taking from the user interface we are storing in varaibles and storing in array and passing for prediction.

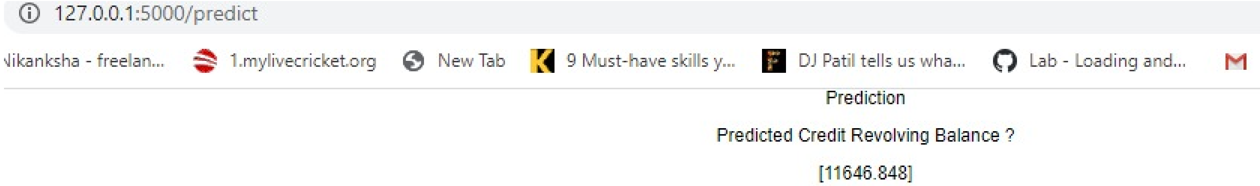
```
def predict():

    features=[]
    a = (float(request.form["loan_amnt"]))
    features.append(a)
    b=request.form["terms"]
    features.append(terms['terms'][b])
    c=(float(request.form["Rate_of_intrst"]))
    features.append(c)
    d=request.form["grade"]
    features.append(grade['grade'][d])
    e=request.form["home_ownership"]
    features.append(home_ownership['home_ownership'][e])
    f=(float(request.form["annual_inc"]))
    features.append(f)
    g=request.form["verification_status"]
    features.append(verification_status['verification_status'][g])
    h=request.form["purpose"]
    features.append(purpose['purpose'][h])
    i=(float(request.form["debt_income_ratio"]))
    features.append(i)
    j=(float(request.form["delinq_2yrs"]))
    features.append(j)
    k=(float(request.form["inq_last_6mths"]))
    features.append(k)
    l=(float(request.form["numb_credit"]))
```

Finaly we r calling here model for predicton and passing our data taken from client in array for prediction. After that catching the predicted value and passing it to the result html page

```python
    features.append(initial_list_status['initial_list_status'][o])
    p=(float(request.form["total_rec_int"]))
    features.append(p)
    q=(float(request.form["total_rec_late_fee"]))
    features.append(q)
    r=(float(request.form["recoveries"]))
    features.append(r)
    s=request.form["Experience"]
    features.append(Experience['Experience'][s])
    t=(float(request.form["mths_since_last_delinq"]))
    features.append(t)
    u=(float(request.form["tot_curr_bal"]))
    features.append(u)
    v=(float(request.form["tot_colle_amt"]))
    features.append(v)

    final_features = np.array(features).reshape(-1,22)
    prediction = model.predict(final_features)
    return render_template('results.html',prediction=prediction)

    #return render_template('index1.html', prediction_text='Predicted Credit Revolving Balance : {}'.format(prediction))

if __name__ == "__main__":
    app.run(debug=True)
```

Model Deployment

Model Deployment - Prediction



Summary

- Out of the 34 columns provided, we deployed a model <u>using 22 columns</u> on the basis of **HIGH Feature Importance and RELEVANCE to the prediction**.

- **Clustering** was introduced to improve the functionality between the input variables.

- **“replace” function** was used to merge similar kind of variables to reduce the dimensionality.

- **Frequency Encoding** was used to convert categorical values into numeric.

- To achieve **LOW RMSE and Balanced R-squared values, XGBoost was chosen over Linear Regression, Random Forest and Neural Network models**.

- Using **Flask**, we have deployed a web-application which predicts the revolving balance of a user.

Challenges faced?

- Complexity on the volume of data?
- Relationship between the variables present in the data?
- How to cluster and encode, and not invite the curse of dimensionality?
- Achieve a reasonable RMSE and R-squared without over- fitting/under-fitting data?
- Deployment of selected model as a web-based application?

How did you overcome?

- Complexity on the volume of data? ***– Understand the terms and exclude variables of less importance.***
- Relationship between the variables present in the data? ***- Data Analysis and Visualization techniques helped figure out relationship and enhance prediction.***
- How to cluster and encode, and not invite the curse of dimensionality? ***– Used 'replace' function to merge variables and then Frequency Encoded to maintain dimensionality.***
- Achieve a reasonable RMSE and R-squared without over- fitting/under-fitting data? ***– Employed various models like Linear Regressor, Random Forest Regressor, XGBoost and Neural Networks to optimize the model selection and its deployment.***
- Deployment of selected model as a web-based application? – ***Peer Learning within the***

Thank you