Classification Project

Loans Eligibility Prediction

Evgenia Samsonov and Idan Leibovitch

Mission statement

Dream Housing Finance company deals in all home loans. They have a presence across all urban, semi-urban, and rural areas. Customer-first applies for a home loan and then the company validates the customer eligibility for a loan.

Goal :
Create automated tool for loans qualification process by the given data.

Overview

- EDA with Pandas and Seaborn
- Preprocessing the data, convert categorical to numerical, handling missing values
- Prepare for models (fit models, normalize, oversampling)
- Apply classifications models for comparing results of scores
- Model Comparison and choosing best one of scores for business inquiries
- Conclusions and insights

EDA and Preprocessing the data

Loan_ID----------> Unique Loan ID
Gender ----------> Male/ Female
Married ----------> Applicant married (Y/N)
Dependents ----------> Number of dependents
Education ----------> Applicant Education (Graduate/ Under Graduate)
Self_Employed ---------> Self-employed (Y/N)
ApplicantIncome -------> Applicant income
CoapplicantIncome -----> Coapplicant income
LoanAmount ----------> Loan amount in thousands
Loan_Amount_Term ------> Term of a loan in months
Credit_History --------> Credit history meets guidelines
Property_Area ---------> Urban/ Semi-Urban/ Rural
Loan_Status ----------> Loan approved (Y/N)

There are 614 entries
There are total 13 features
There are three types of datatype : float64(4), int64(1), object(8)
There are three many missing values (146 NaN).

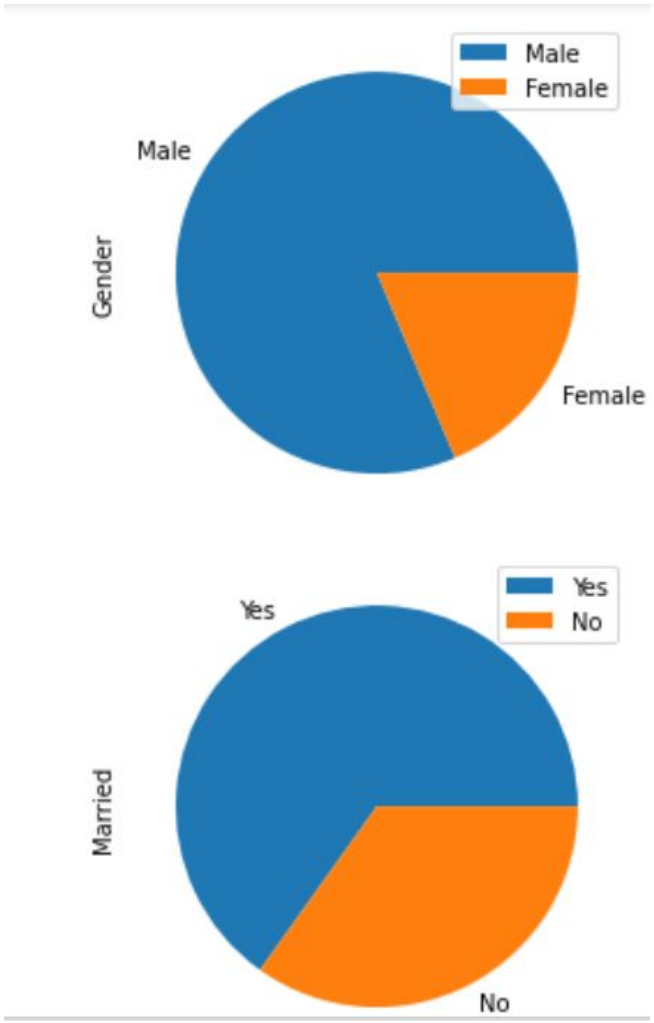
Loan_Status feature Boolean values, So we replace Y values with 1 and N values with 0 and same for other Boolean types of columns, "Male": 1, "Female" : 0 After replacing we will use median, mean and mode with NaN values.

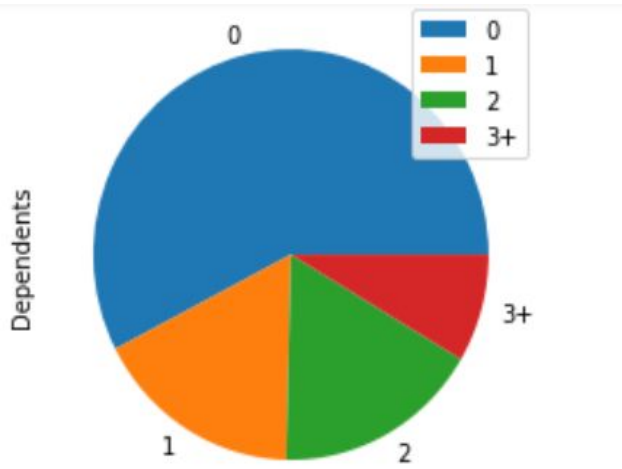
```
loans.Loan_Status = loans.Loan_Status.replace({"Y": 1, "N" : 0})
```

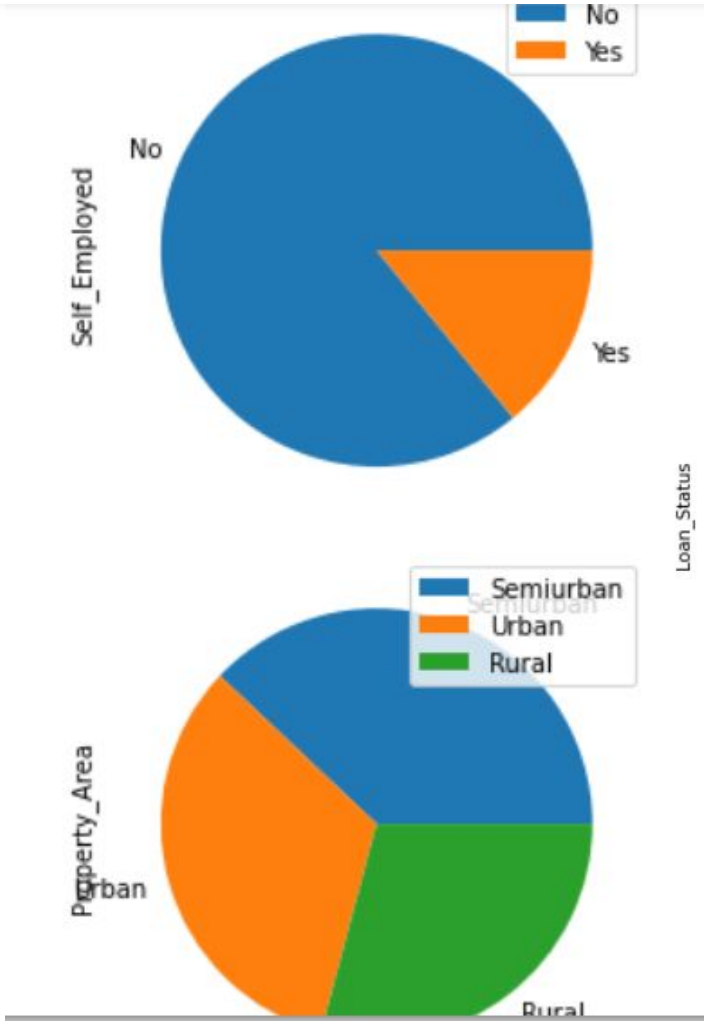
```
loans['Gender'].fillna(loans['Gender'].mode()[0.0], inplace=True)
```

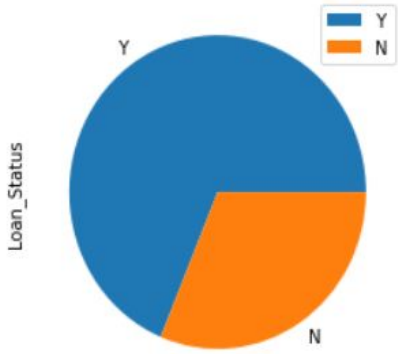
EDA

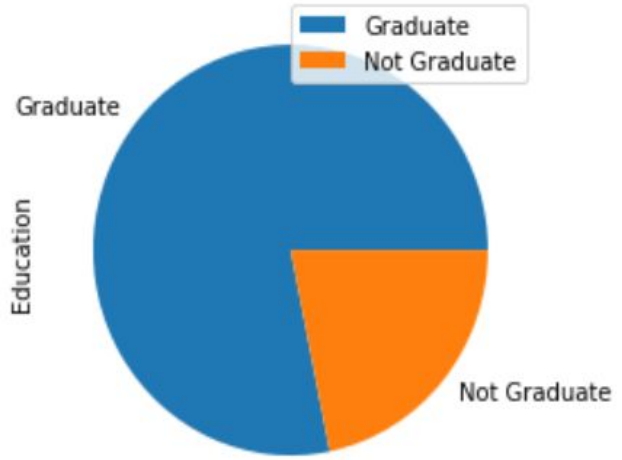
Explore object type with value_counts





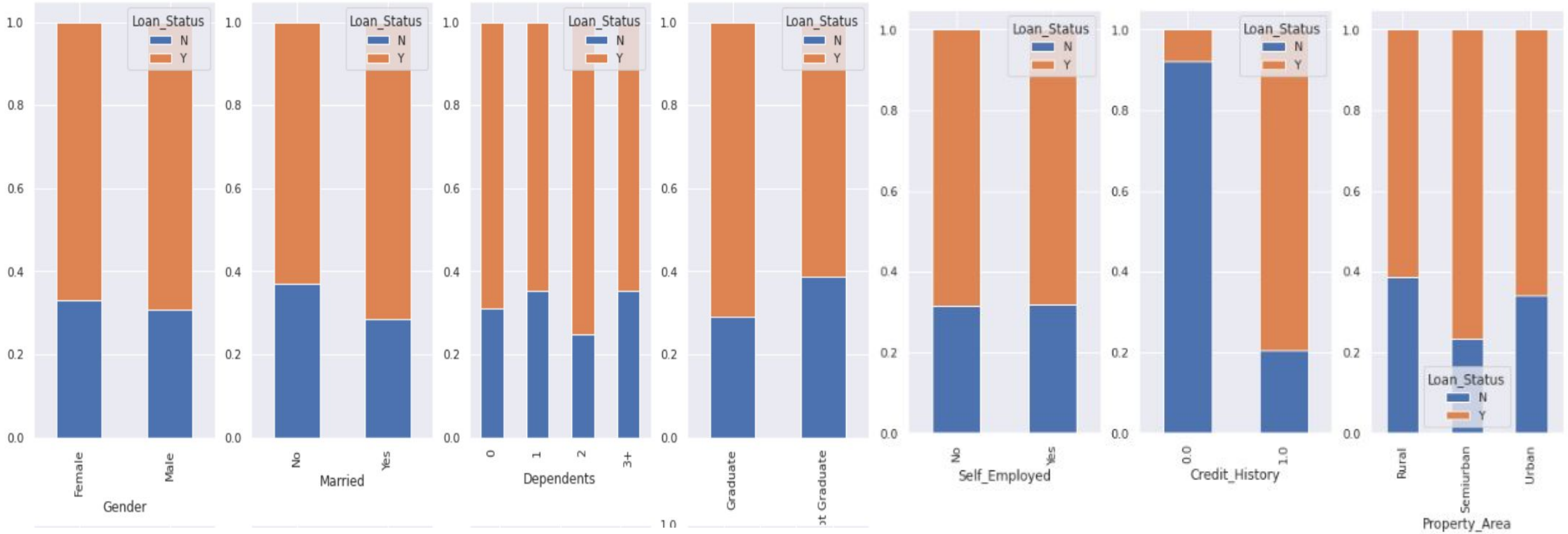




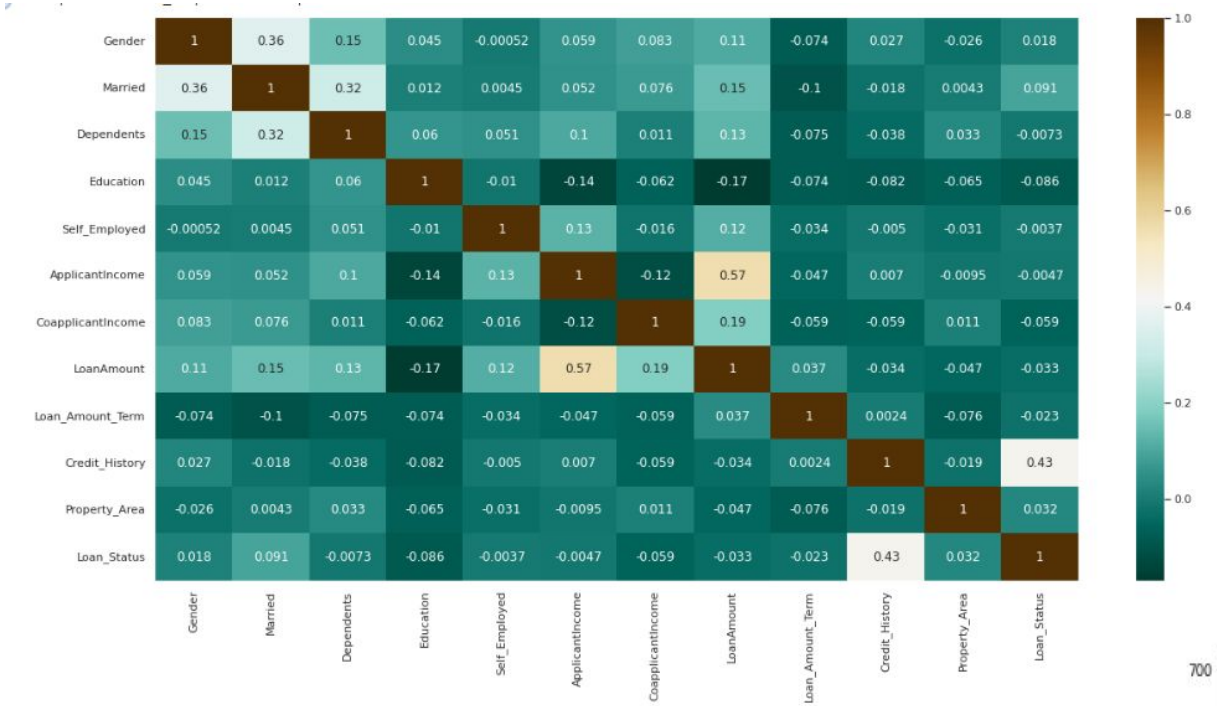


EDA

Explore Loan_Status by each feature

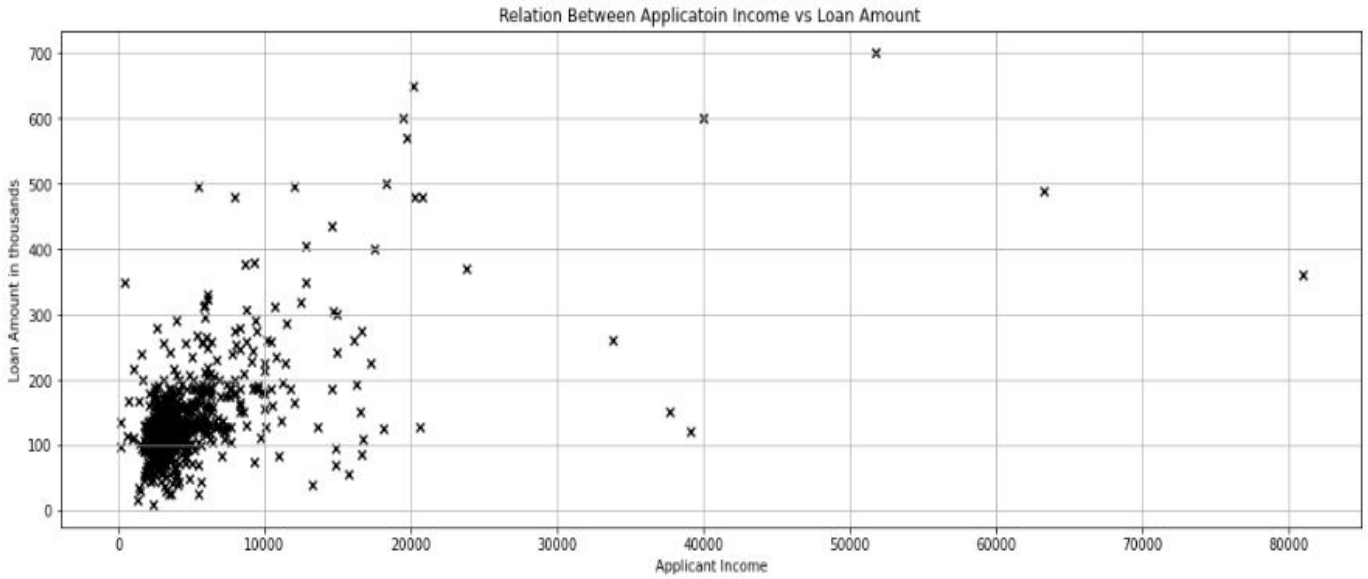


EDA



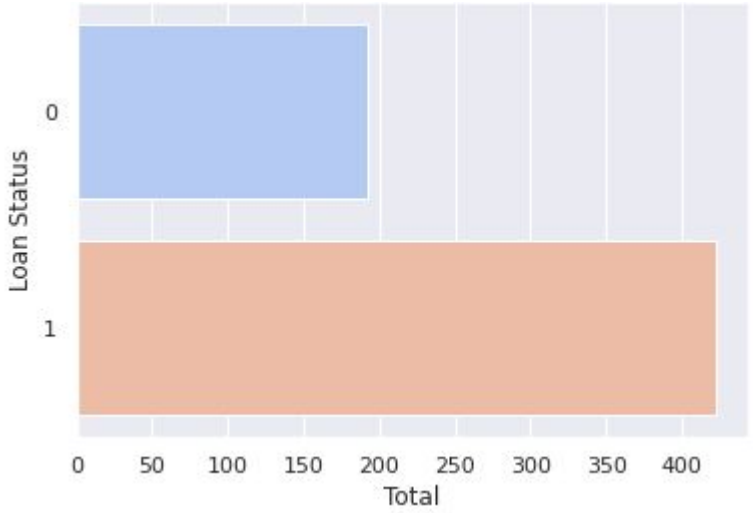
Heatmap reflects the strong relations between loan_status and credit_history, as well for Application_income and loan_amount

Plot shows relation between Applicant Income vs requested Loan Amount.



Preparing for models

1.Target variable-Loan_Status

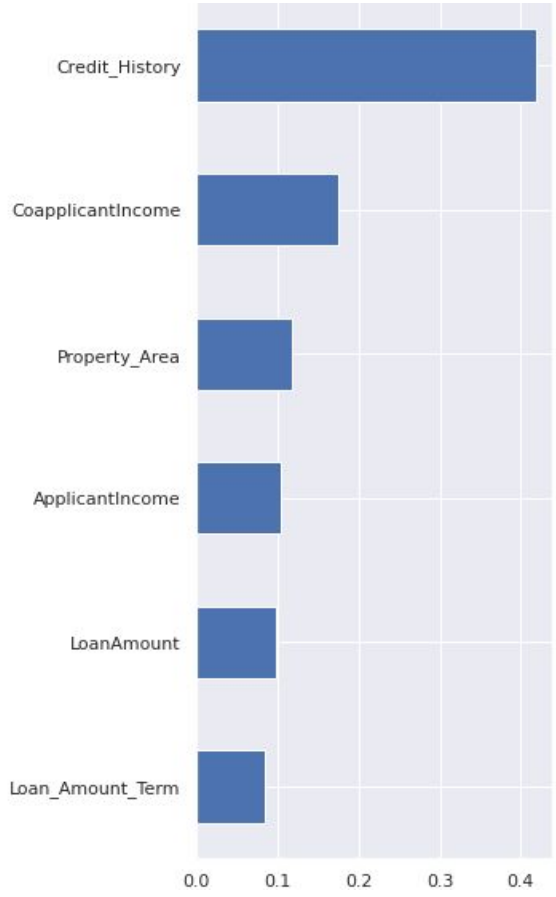


2. Over sampling by SMOTE (Imbalanced data)

```
# Preparing strategy
oversample = SMOTE()

## Over sampling
X_over, y_over = oversample.fit_resample(X_train, y_train)
print(f"after under sampeling:  counter = {Counter(y_over)}")
```

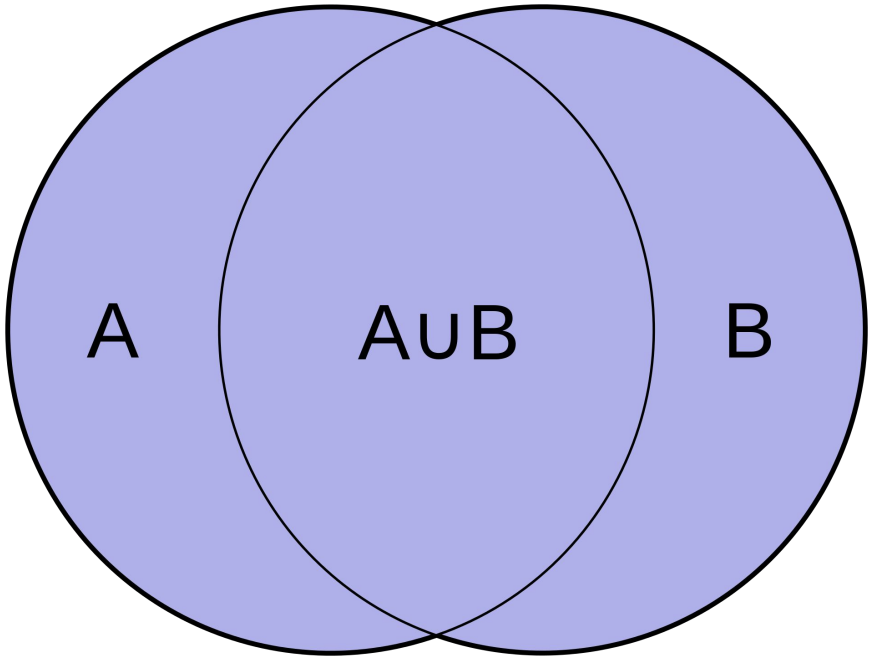
```
Before over under sampeling : counter = Counter({1: 291, 0: 120})
after under sampeling:  counter = Counter({1: 291, 0: 291})
```

3. Feature importance: Credit_History, CoapplicantIncome, Property_Area, ApplicantIncome, LoanAmount, Loan_Amount_Term



Classification models

- Gradient Boosting
- Ada boosting on Logistic Regression and Decision Tree
- Logistic Regression
- KNN
- Decision Tree
- Random Forest



Running the models with:

1.Clean (regular) data

2.Normalized data

3.Balanced target

4.Features important

Models with clean (regular) data

	Model	Training Accuracy %	Testing Accuracy %	Training precision %	Testing precision %	Training recall %	Testing recall %
0	Logistic Regression	76.156	78.818	78.299	78.205	91.753	93.130
1	KNN	71.290	65.025	72.123	65.625	96.907	96.183
2	RandomForest	100.000	70.443	100.000	72.611	100.000	87.023
3	Decision Tree	80.535	73.892	87.000	78.676	85.567	81.679
4	Gradiant Boosting	90.268	75.862	88.146	76.623	99.656	90.076
5	ADA Boosting	87.835	66.502	92.281	72.340	90.378	77.863

Models with normalized data

	Model	Training Accuracy %	Testing Accuracy %	Training precision %	Testing precision %	Training recall %	Testing recall %
0	Logistic Regression - Norm	77.129	76.847	79.758	77.632	90.722	90.076
1	KNN - Norm	76.156	73.892	77.493	74.074	93.471	91.603
2	RandomForest- Norm	100.000	72.414	100.000	73.885	100.000	88.550
3	Decision Tree- Norm	80.535	73.892	86.760	78.676	85.567	81.679
4	Gradiant Boosting - Norm	90.268	75.862	88.146	76.623	99.656	90.076
5	ADA Boosting- Norm	87.835	66.502	92.281	72.340	90.378	77.863

Models with balanced target

	Model	Training Accuracy %	Testing Accuracy %	Training precision %	Testing precision %	Training recall %	Testing recall %
0	Logistic Regression - SMOTE	77.320	74.427	75.563	72.535	80.756	78.626
1	KNN - SMOTE	65.979	48.092	67.286	48.201	62.199	51.145
2	RandomForest- SMOTE	100.000	74.046	100.000	71.724	100.000	79.389
3	Decision Tree- SMOTE	82.818	78.244	78.852	74.026	89.691	87.023
4	Gradiant Boosting - SMOTE	92.440	77.099	89.968	72.327	95.533	87.786
5	ADA Boosting- SMOTE	90.378	75.954	85.285	72.667	97.595	83.206

Models with features important

	Model	Training Accuracy %	Testing Accuracy %	Training precision %	Testing precision %	Training recall %	Testing recall %
0	Logistic Regression - Features	75.669	77.833	78.006	76.875	91.409	93.893
1	KNN - Features	71.290	65.025	72.123	65.625	96.907	96.183
2	RandomForest- Features	100.000	73.399	100.000	75.497	100.000	87.023
3	Decision Tree- Features	80.779	73.892	86.806	78.676	85.911	81.679
4	Gradiant Boosting - Features	88.078	69.458	86.667	71.166	98.282	88.550
5	ADA Boosting- Features	85.888	62.562	92.364	69.784	87.285	74.046

Grid search

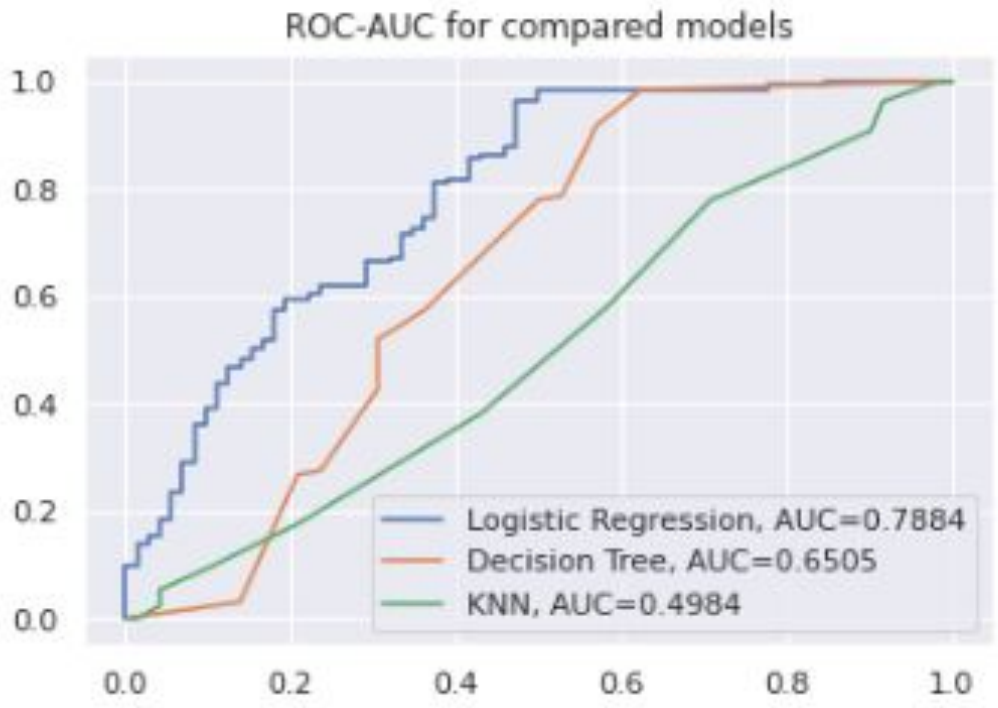
scoring='accuracy'

```
----- model = LogisticRegression

train =  0.7712895377128953
test =  0.7684729064039408


----- model = DecisionTreeClassifier

train =  0.959
test =  0.709


----- model = KNeighborsClassifier

train =  0.715
test =  0.635
```

scoring='recall'

```
----- model = LogisticRegression

train =  1.0
test =  1.0


----- model = DecisionTreeClassifier(

train =  1.0
test =  0.7404580152671756


----- model = KNeighborsClassifier

train =  0.993
test =  1.0
```

ROC-AUC



Probability by specific data line

	Model	Probablity for X	Probablity for X_train	Probabilty for X_test
0	Logistic Regression Prob	69.54 %	75.7 %	81.55 %
1	Decision Tree Prob	62.26 %	96.35 %	66.2 %
2	Random Forest Prob	87.0 %	95.0 %	81.0 %
3	ADA Boost Prob	68.07 %	94.17 %	65.66 %
4	Gradiant Boost Prob	82.15 %	91.56 %	79.76 %

Insights and Conclusions

- Business' problem/question is affecting on the model and metric need to be chosen:
  - For general accuracy if to approve/disapprove the loan we will use the 'accuracy' metric and the score is 77%.
  - For only approved loans with low risk -we'll use the recall metric which has high score of 100% actual loans approved from the predicted .
- For recall and precision metrics our score for "0" (unapproved loans) is very low. This must been taken in consideration if proceeding with those metrics.
- Future plans :
  - Find the best models for unapproved loans, getting the complete business solution.
  - Testing more models for getting higher accuracy score and AUC closer to 1.
  - Examining the threshold for each model

Thank You for your time