Loan Approval Prediction

December 28, 2023

1 Loan Approval Prediction

The aim of this project is to predict whether the loan would be approved by the bank, by analyzing the applicant's information which includes loan amount, tenure, cibil score, education, assets and many other variables. Through this project, we can analyze the factors that affect the loan approval status for a new applicant. Morever, this will help in providing priority services to the customers who are more likely to get their loan approved.

1.1 About the Dataset

The loan approval dataset is a collection of financial records and associated information used to determine the eligibility of individuals or organizations for obtaining loans from a lending institution. It includes various factors such as cibil score, income, employment status, loan term, loan amount, assets value, and loan status. This dataset is commonly used in machine learning and data analysis to develop models and algorithms that predict the likelihood of loan approval based on the given features.

1.2 Data Dictionary

Variable	Description
loan_id	Unique loan ID
no_of_dependents	Number of dependents of the applicant
education	Education level of the applicant
self_employed	If the applicant is self-employed or not
income_annum	Annual income of the applicant
loan_amount	Loan amount requested by the applicant
loan_tenure	Tenure of the loan requested by the applicant (in Years)
cibil_score	CIBIL score of the applicant
residential_asset_value	Value of the residential asset of the applicant
commercial_asset_value	Value of the commercial asset of the applicant
luxury_asset_value	Value of the luxury asset of the applicant
bank_assets_value	Value of the bank asset of the applicant
loan_status	Status of the loan (Approved/Rejected)

[1]:
```
# Importing the libraries
```

```python
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings('ignore')
```

[2]:
```python
# Loading the dataset

df = pd.read_csv('loan_approval_dataset.csv')
df.head()
```

[2]:
```
   loan_id  no_of_dependents      education self_employed  income_annum  \
0        1                 2       Graduate            No       9600000
1        2                 0   Not Graduate           Yes       4100000
2        3                 3       Graduate            No       9100000
3        4                 3       Graduate            No       8200000
4        5                 5   Not Graduate           Yes       9800000

   loan_amount  loan_term  cibil_score  residential_assets_value  \
0     29900000         12          778                   2400000
1     12200000          8          417                   2700000
2     29700000         20          506                   7100000
3     30700000          8          467                  18200000
4     24200000         20          382                  12400000

   commercial_assets_value  luxury_assets_value  bank_asset_value  \
0                 17600000             22700000           8000000
1                  2200000              8800000           3300000
2                  4500000             33300000          12800000
3                  3300000             23300000           7900000
4                  8200000             29400000           5000000

  loan_status
0    Approved
1    Rejected
2    Rejected
3    Rejected
4    Rejected
```

1.3 Data Preprocessing

[3]:
```python
# Checking the shape of the dataset

df.shape
```

[3]: (4269, 13)

Removing the unnecessary loan_id as it is an identifier column

[4]:
```
df.drop(columns = 'loan_id', inplace = True)
```

[5]:
```
# Checking for null/missing values

df.isnull().sum()
```

[5]:
```
no_of_dependents            0
education                   0
self_employed               0
income_annum                0
loan_amount                 0
loan_term                   0
cibil_score                 0
residential_assets_value    0
commercial_assets_value     0
luxury_assets_value         0
bank_asset_value            0
loan_status                 0
dtype: int64
```

[6]:
```
# Checking the data types of the columns

df.dtypes
```

[6]:
```
no_of_dependents             int64
education                   object
self_employed               object
income_annum                 int64
loan_amount                  int64
loan_term                    int64
cibil_score                  int64
residential_assets_value     int64
commercial_assets_value      int64
luxury_assets_value          int64
bank_asset_value             int64
loan_status                 object
dtype: object
```

The dataset has 4 kinds of assets that are - Residential, Commercial, Luxury and Bank. I am categorizing these assets into two category i.e. Movable and Immovable assets. The Residential and Commercial asset would be added to the Immovable assets and Luxury and Bank assets would be added to the Movable assets.

[7]:
```
# Movable Assets

df['Movable_assets'] = df[' bank_asset_value'] + df[' luxury_assets_value']
```

[8]:
```
# Immovable Assets

df['Immovable_assets'] = df[' residential_assets_value'] + df['␣
↪commercial_assets_value']
```

[9]:
```
# Drop columns

df.drop(columns = [' residential_assets_value', ' commercial_assets_value', '␣
↪luxury_assets_value', ' bank_asset_value'],
        inplace = True)
```

Descriptive Statistics

[10]:
```
df.describe()
```

[10]:
```
        no_of_dependents   income_annum   loan_amount    loan_term  \
count        4269.000000   4.269000e+03  4.269000e+03  4269.000000
mean            2.498712   5.059124e+06  1.513345e+07    10.900445
std             1.695910   2.806840e+06  9.043363e+06     5.709187
min             0.000000   2.000000e+05  3.000000e+05     2.000000
25%             1.000000   2.700000e+06  7.700000e+06     6.000000
50%             3.000000   5.100000e+06  1.450000e+07    10.000000
75%             4.000000   7.500000e+06  2.150000e+07    16.000000
max             5.000000   9.900000e+06  3.950000e+07    20.000000

        cibil_score  Movable_assets  Immovable_assets
count   4269.000000    4.269000e+03      4.269000e+03
mean     599.936051    2.010300e+07      1.244577e+07
std      172.430401    1.183658e+07      9.232541e+06
min      300.000000    3.000000e+05     -1.000000e+05
25%      453.000000    1.000000e+07      4.900000e+06
50%      600.000000    1.960000e+07      1.060000e+07
75%      748.000000    2.910000e+07      1.820000e+07
max      900.000000    5.380000e+07      4.660000e+07
```

[11]:
```
df.head()
```

[11]:
```
    no_of_dependents      education  self_employed   income_annum  \
0                  2       Graduate             No        9600000
1                  0   Not Graduate            Yes        4100000
2                  3       Graduate             No        9100000
3                  3       Graduate             No        8200000
4                  5   Not Graduate            Yes        9800000
```

```
   loan_amount  loan_term  cibil_score loan_status  Movable_assets  \
0     29900000         12          778    Approved        30700000
1     12200000          8          417    Rejected        12100000
2     29700000         20          506    Rejected        46100000
3     30700000          8          467    Rejected        31200000
4     24200000         20          382    Rejected        34400000

   Immovable_assets
0          20000000
1           4900000
2          11600000
3          21500000
4          20600000
```

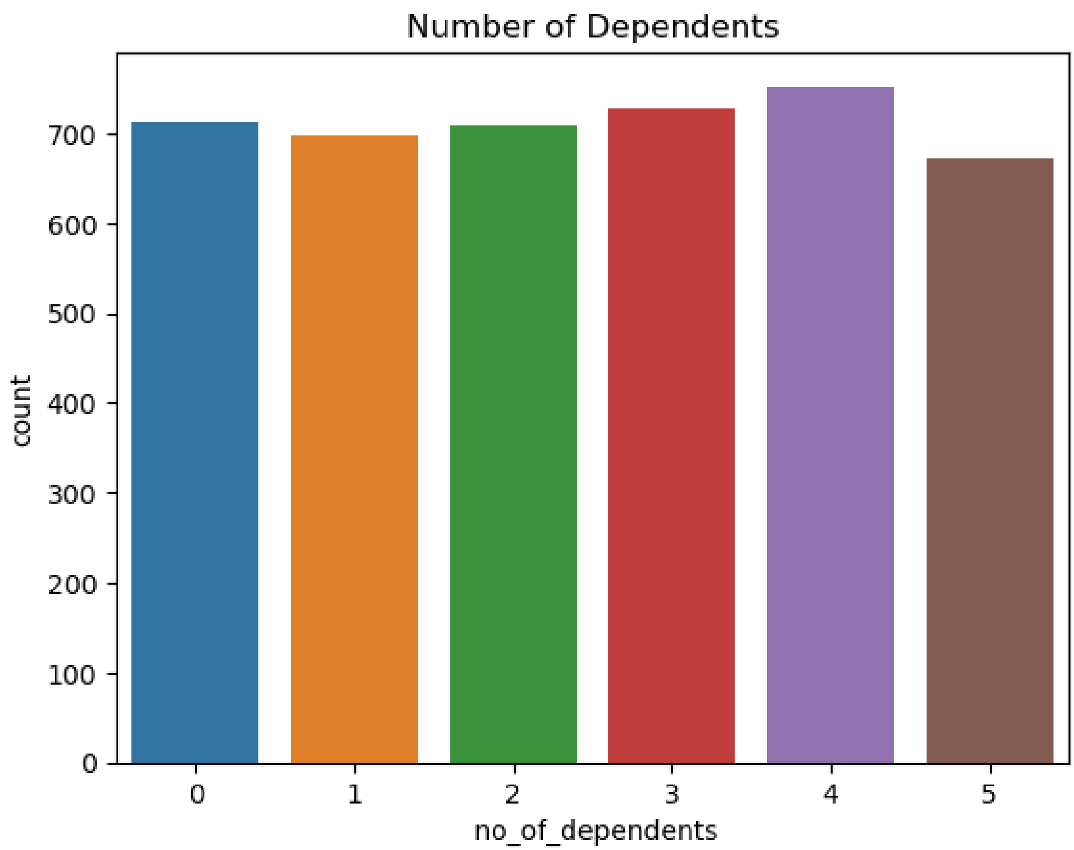
1.4 Exploratory Data Analysis

In the exploratory data analysis, i will be looking at the distribution of the data across the variables, followed by relationship between the independent and target variable and the correlation among the variables. Through the visualization, Iwill be able to understand the possible trends and patterns in the data and come to know about the hidden insights of the data.

Number of Dependents

[12]:
```
sns.countplot(x = ' no_of_dependents', data = df).set_title('Number of␣
↪Dependents')
```

[12]:
```
Text(0.5, 1.0, 'Number of Dependents')
```

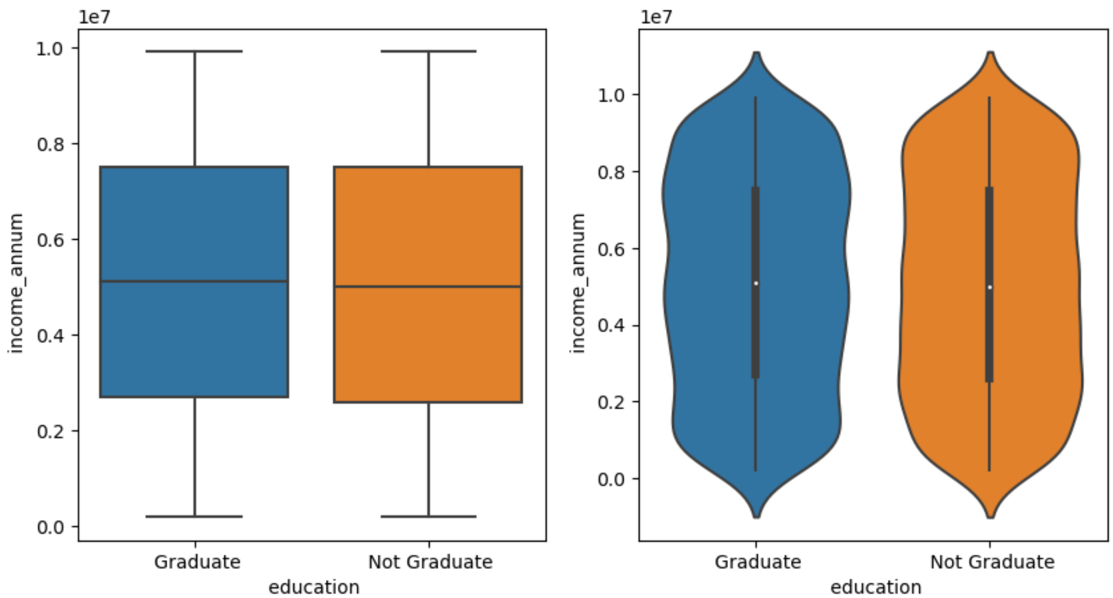


This graph shows the number of dependent indivduals on the loan applicant. There is not much difference in the number of dependents, however, there are more applicants with 4 and 3 dependents than the other categories. Since the number of dependents increases the disposable income of the applicant decreases. So I assume that that the number of applicants with 0 or 1 dependent will have higher chances of loan approval.

Education and Income

[13]:
```
fig, ax = plt.subplots(1,2,figsize = (10,5))
sns.boxplot(x = ' education', y = ' income_annum', data = df, ax = ax[0])
sns.violinplot(x = ' education', y = ' income_annum', data = df, ax = ax[1])
```

[13]: <Axes: xlabel=' education', ylabel=' income_annum'>

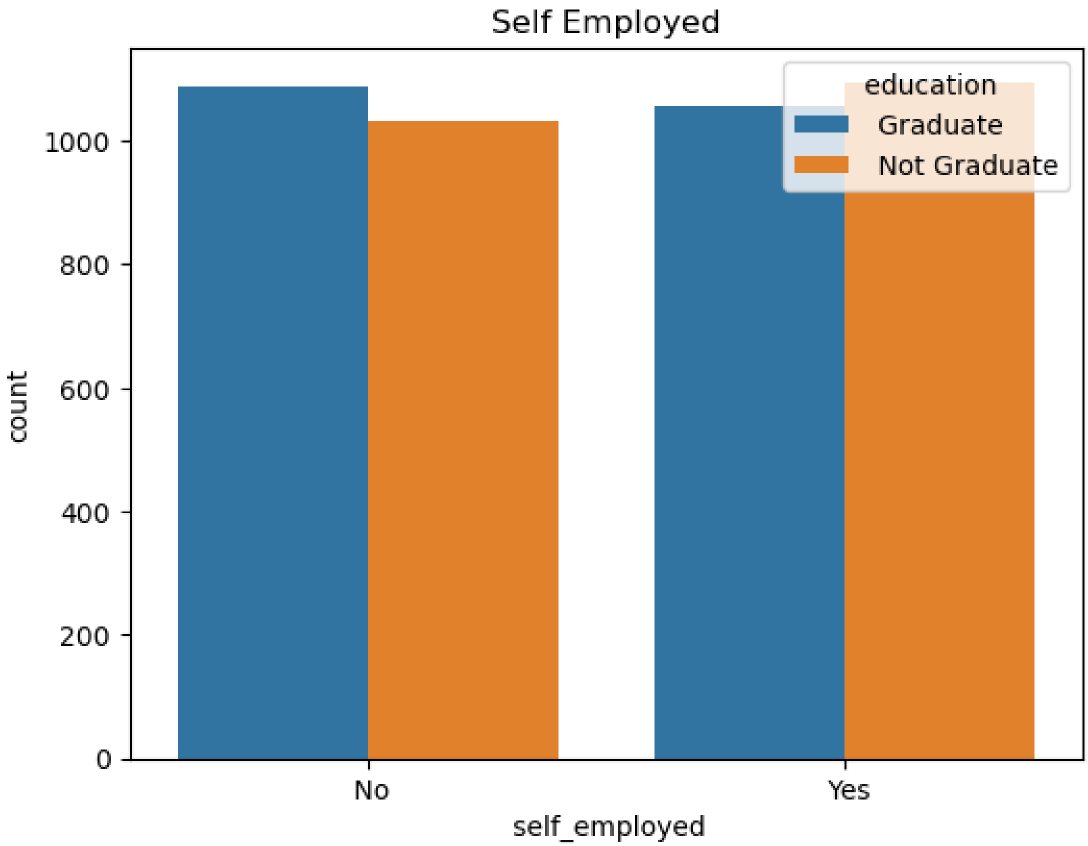


These two graphs - boxplot and violinplot visualizes the education of applicants along with their annual income. The boxplot shows some interesting fact that both the graduates and non-graduates have nearly same median income with very small increase in income of graduates. Moreover the violinplot shows the distribution of income among the graduates and non graduate applicants, where we can see that non graduate applicants have a even distribution between income 2000000 and 8000000, whereas there is a uneven distribution among the graduates with more applicants having income between 6000000 and 8000000. Since there is not much change in annual income of graduates and non graduates, I assume that education does not play a major role in the approval of loan.

Employment Status and Education

[14]:
```
sns.countplot(x = ' self_employed', data = df, hue = ' education').
 ↪set_title('Self Employed')
```

[14]: Text(0.5, 1.0, 'Self Employed')

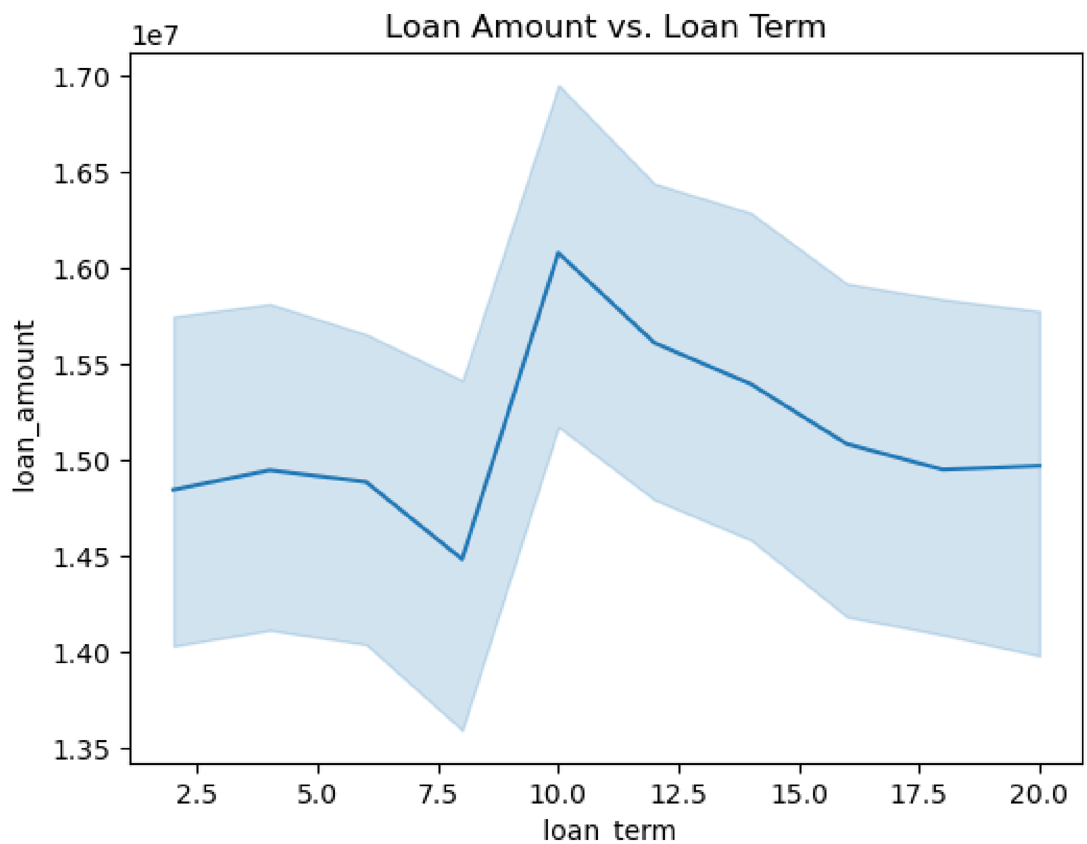


This graph shows the number of self employed applicants along with their education. From the educational prepespective the majority of the graducate applicants are not self employed wheareas majority of the non-graduates are self employed. This means that graduates applicants are more likely to be salaried employees and non-graduates are more likely to be self employed. This could be a determining factor in loan approval because salaried employees are more likely to have a stable income and hence are more likely to pay back the loan as compared to self employed applicants whose income may not be stable. But this could also be possible that the self employed applicants are earning more than the salaried employees and hence are more likely to pay back the loan. This is a very important factor to consider while predicting the loan approval.

Loan Amount and Tenure

[15]:
```
sns.lineplot(x = ' loan_term', y = ' loan_amount', data = df).set_title('Loan␣
↪Amount vs. Loan Term')
```

[15]:
```
Text(0.5, 1.0, 'Loan Amount vs. Loan Term')
```

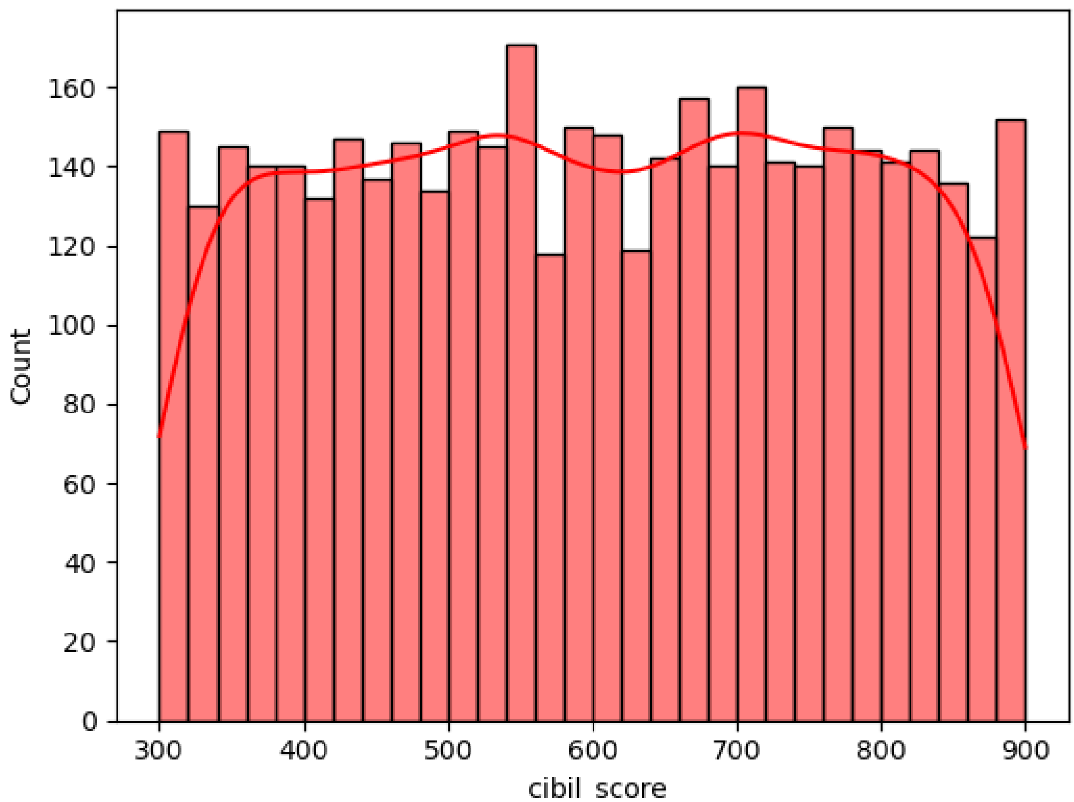


This line plot shows the trend between the loan amount and the loan tenure. Between the loan tenure of 2.5 - 7.5 years the loan amount is between 1400000 - 15500000. However the loan amount is significantly higher for the loan tenure of 10 years.

CIBIL Score Distribution

[16]:
```
sns.histplot(df[' cibil_score'], bins = 30, kde = True, color = 'red')
```

[16]: `<Axes: xlabel=' cibil_score', ylabel='Count'>`



Before looking at the cibil score, lets have a look at the cibil score ranges and their meaning.

Cibil Score	Meaning
300-549	Poor
550-649	Fair
650-749	Good
750-799	Very Good
800-900	Excellent

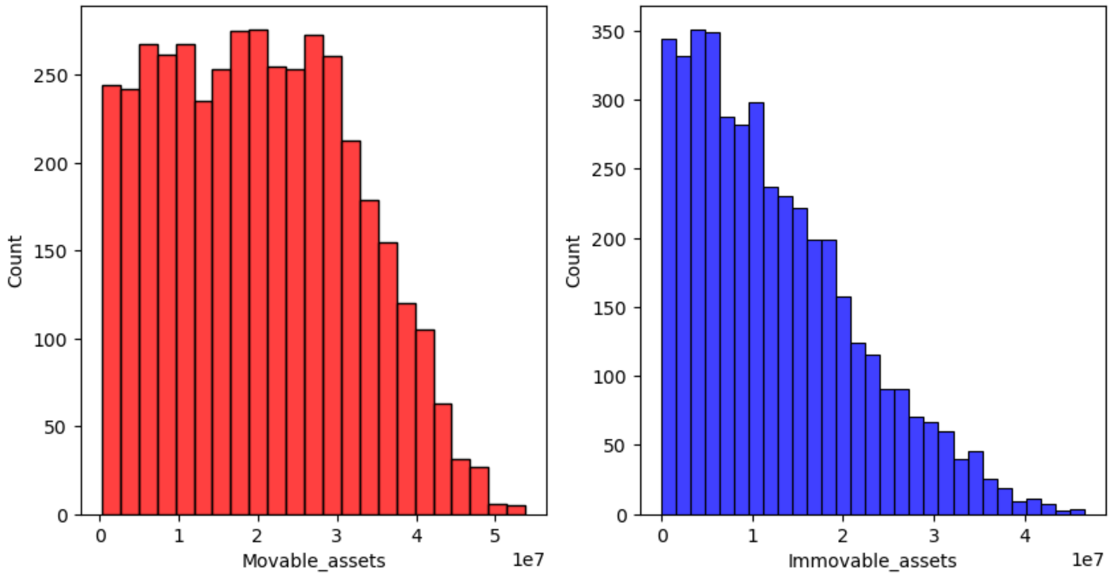
Source: godigit.com

Taking the above table as a reference for the cibil score quality, majority of the customers have cibil score below 649, which affects their loan application. However there are many applicants with cibil score above 649, which is a good sign for the bank. The bank can target these customers and provide them with priority services. The bank can also provide them with special offers and discounts to attract them to take loans from the bank. From this, I build a hypothesis that the customers with cibil score above 649 are more likely to get their loan approved.

Asset Distribution

[17]:
```
fig, ax = plt.subplots(1, 2, figsize = (10,5))
sns.histplot(df['Movable_assets'], ax = ax[0], color = 'red')
sns.histplot(df['Immovable_assets'], ax = ax[1], color = 'blue')
```

[17]: <Axes: xlabel='Immovable_assets', ylabel='Count'>



Assets play a major role in loan application. They provides a security to the bank that the person will repay the loan. Looking at the assets, as eralier mentionedI have categorized them in movable and immovable assets. The above graphs shows the distribution of movable and immovable assets in the dataset.

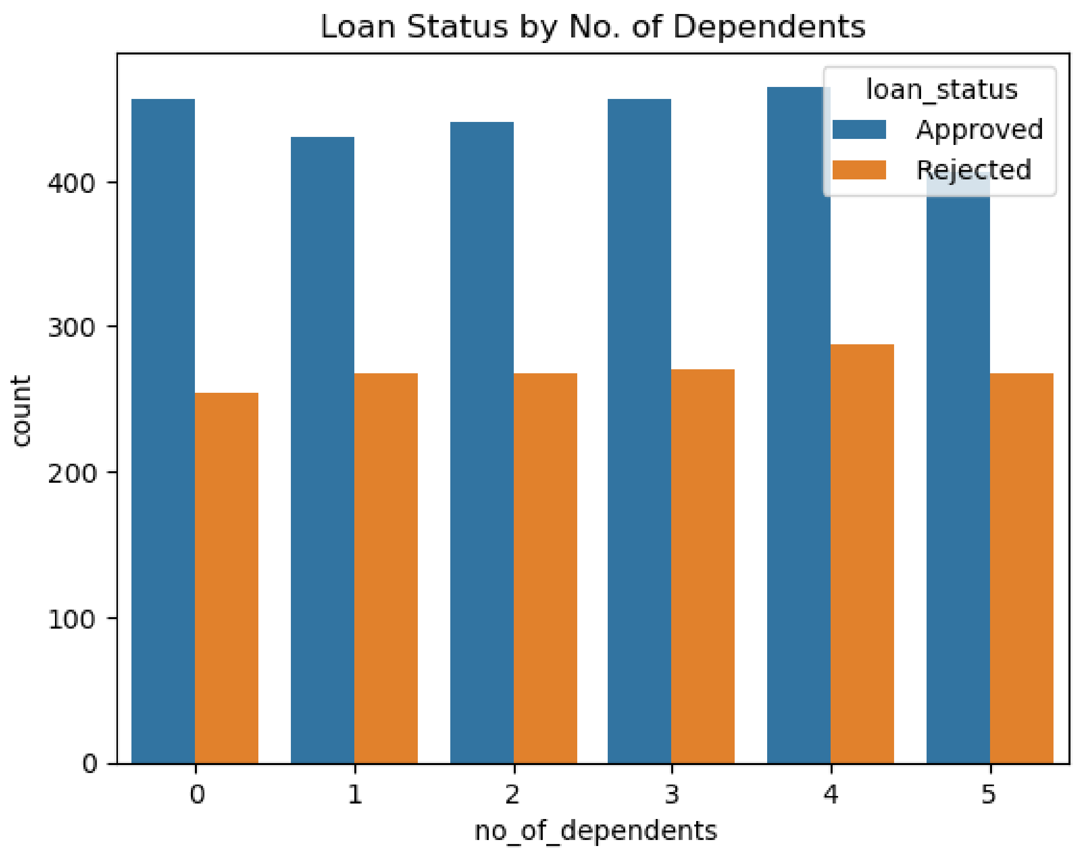
Looking at the movable assets which include bank assets and luxury assets, majority of the applicants have less than 30 million and there is a slight trend of decreasing number of applicants as the movable assets increases. Coming to the immovable assets, which include residential assets and commercial assets, majority of the applicants have less than 15 million of immovable assets and there is a strong trend of decreasing number of applicants as the immovable assets increases after 20 million.

Till now in the EDA, I have explored the distribution of data across the various features as well as relationship between the some of the variables as well and made some assumptions and hypothesis. Now, in order to prove my assumptions and hypothesis I will be looking at the visualization of the relation between the independent variables and the target variable.

Number of Dependents vs. Loan Status

[18]:
```
sns.countplot(x = ' no_of_dependents', data = df, hue = ' loan_status').
 ↪set_title("Loan Status by No. of Dependents")
```

[18]: Text(0.5, 1.0, 'Loan Status by No. of Dependents')

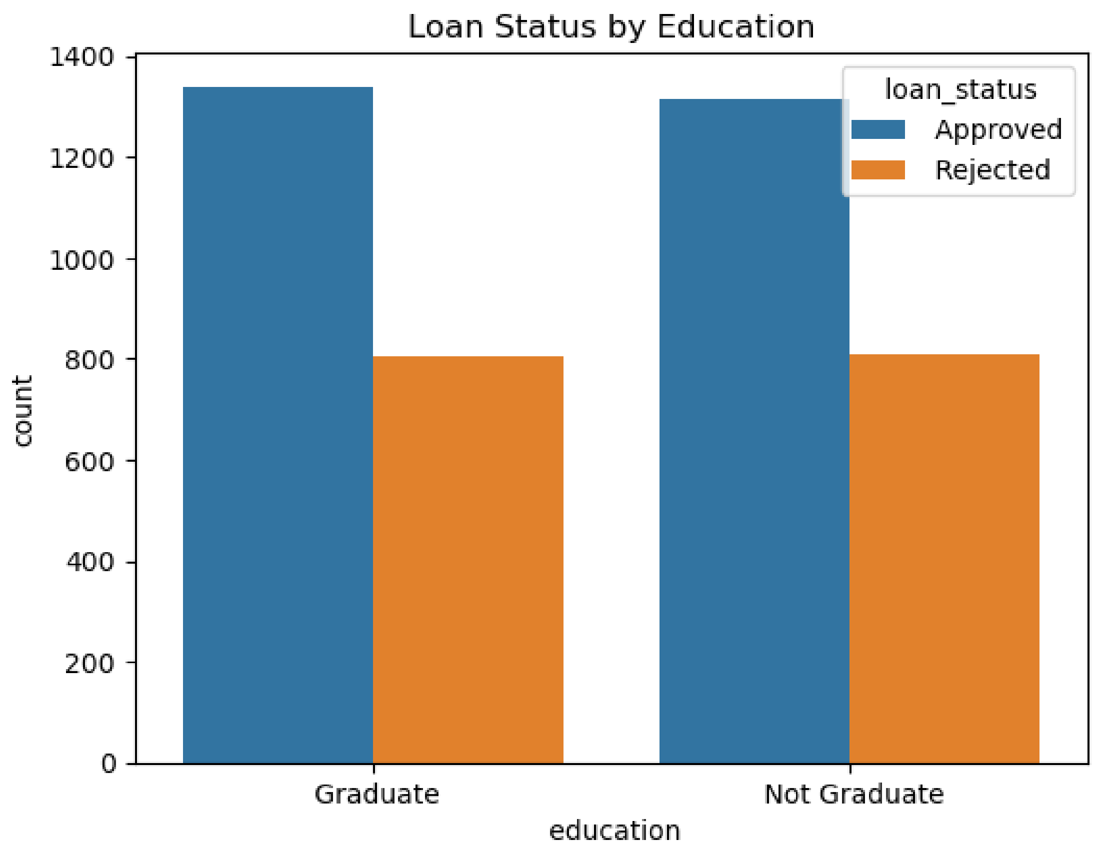


My hypothesis regarding the loan approval based on number of dependents has mixed results. First the hypothesis was somewhat true regarding the rejection chances, the number of loan rejection increases with increase in number of dependents. But the hypothesis was not true regarding the approval chances, the number of loan approval decreases with increase in number of dependents as per my hypothesis. But according to this graph, there has been no major change in the loan approval count with increase in number of dependents. So, my hypothesis regarding the loan approval based on number of dependents is not true.

Education vs. Loan Status

[19]:
```
sns.countplot(x = ' education', hue = ' loan_status', data = df).
↪set_title('Loan Status by Education')
```

[19]:
```
Text(0.5, 1.0, 'Loan Status by Education')
```

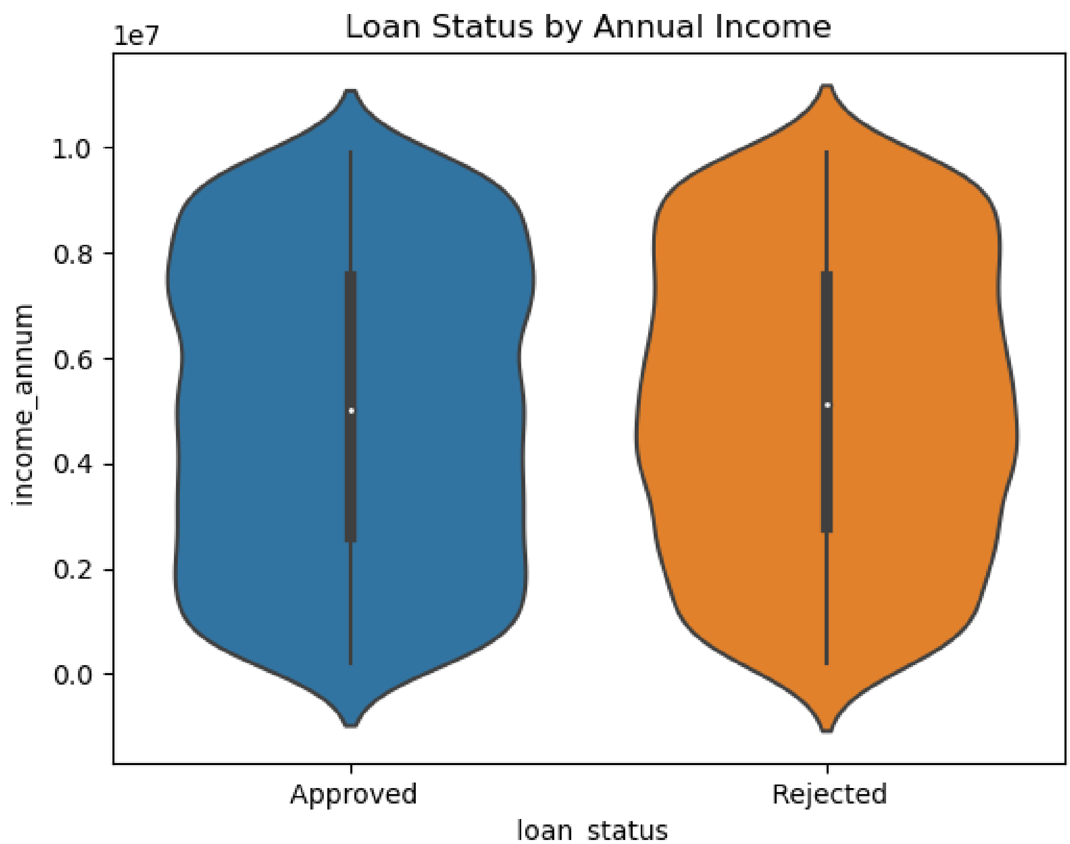


My hypothesis regarding the education not being factor in loan approval was right. The graph shows very minor difference between loan approval and rejection count for the graduate and non graduate applicants. The difference is not significant enough.

Anual Income vs. Loan Status

[20]:
```
sns.violinplot(x = ' loan_status', y = ' income_annum', data = df).
↪set_title('Loan Status by Annual Income')
```

[20]:
```
Text(0.5, 1.0, 'Loan Status by Annual Income')
```

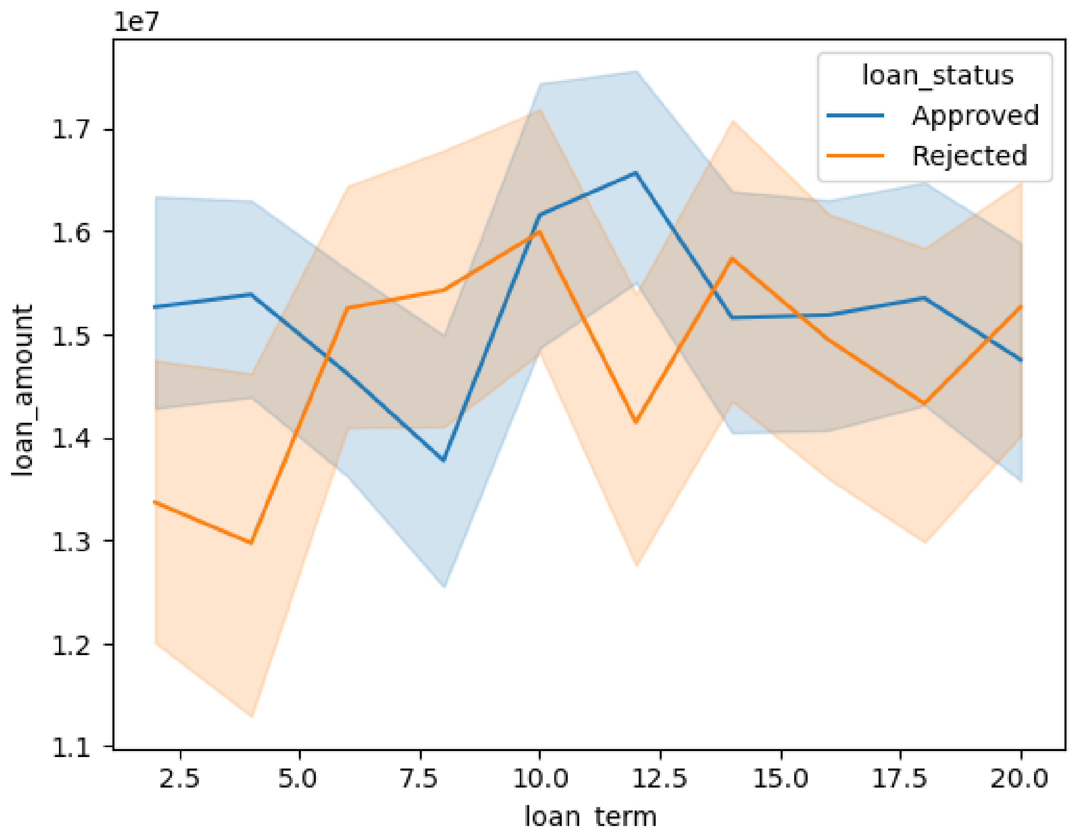


On the whole, there has been no major difference between the annual incomes of the applicant with approved or rejected loan. But still, the approved loan applicants tend to have a higher annual income than the rejected loan applicants which is visible from the violin plot where the approved loan applicants have a higher density in the annual income near 8 miilion annual income.

Loan Amount & Tenure vs. Loan Status

[21]:
```
sns.lineplot(x = ' loan_term', y = ' loan_amount', data = df, hue = '␣
↪loan_status')
```

[21]:
```
<Axes: xlabel=' loan_term', ylabel=' loan_amount'>
```

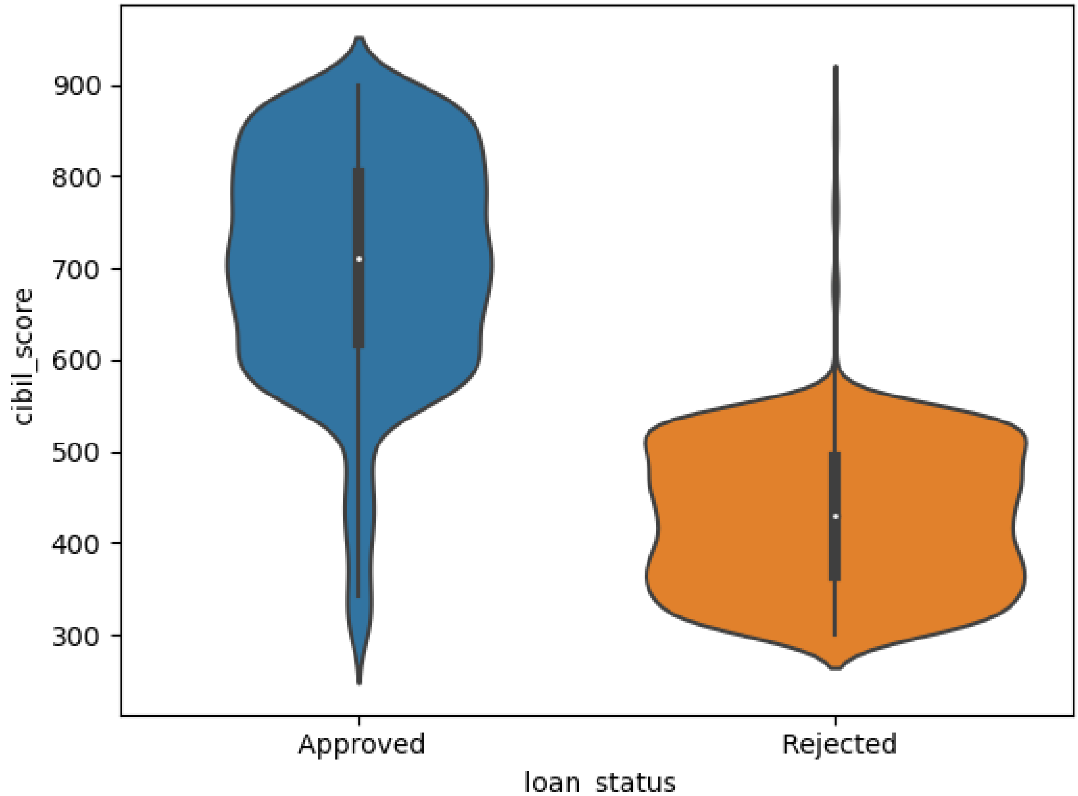


This graph shows the relation between loan amount, loan tenure and loan status. Generally, the approved loans tend have higher amount and shorter repayment tenure. The rejected loans tend to have lower amount and longer repayment tenure. This could be a result of the bank's policy to reject loans with longer repayment tenure. The bank may also reject loans with lower amount as they may not be profitable for the bank.

CIBIL Score vs. Loan Status

[22]:
```
sns.violinplot(x = ' loan_status', y = ' cibil_score', data = df)
```

[22]:
```
<Axes: xlabel=' loan_status', ylabel=' cibil_score'>
```

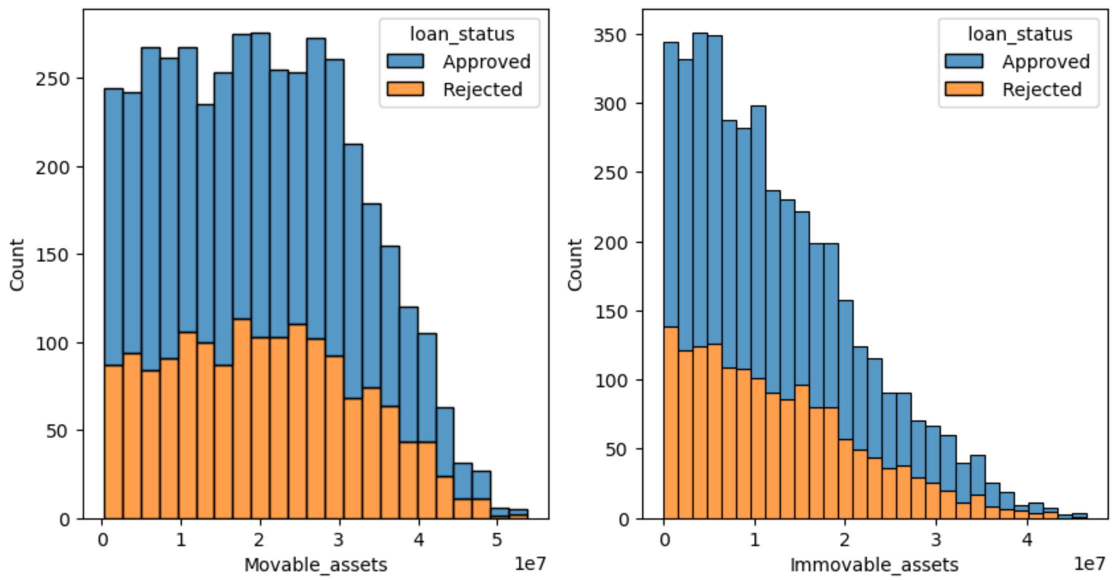


My hypothesis regarding the cibil score and loan approval is absolutely correct. It is evident through the violinplot, where the there is a high distribution above 600 cibil score from the loan approved category. The distribution of the loan not approved category is more spread out and has cibil score less than 550. This also proves my assumption that majority of the applicants have a poor/fair cibil score which affects their loan approval. Hence, having a high cibil score particularly grater than 600 would definitely increase the chances of loan approval.

Assets vs. Loan Status

[23]:
```
fig, ax = plt.subplots(1,2,figsize = (10,5))
sns.histplot(x = 'Movable_assets', data = df, ax = ax[0], hue = ' loan_status',␣
 ↪multiple = 'stack')
sns.histplot(x = 'Immovable_assets', data = df, ax = ax[1], hue = '␣
 ↪loan_status', multiple = 'stack')
```

[23]: <Axes: xlabel='Immovable_assets', ylabel='Count'>



Assets provide security to the bank against which the loan is issued. These two graph visualizes the relation between the movable and immovable assets along with the loan status. The both graph shows that, with increase in the assets the chances of loan approval increases and rejection decreases. The graph also shows that, the movable assets are more than the immovable assets.

Data Preprocessing 2

Label Encoding the Categorical Variables

[24]:
```
# Label Encoding

df[' education'] = df[' education'].map({' Not Graduate':0, ' Graduate':1})
df[' self_employed'] = df[' self_employed'].map({' No':0, ' Yes':1})
df[' loan_status'] = df[' loan_status'].map({' Rejected':0, ' Approved':1})
```

[25]:
```
df.head()
```

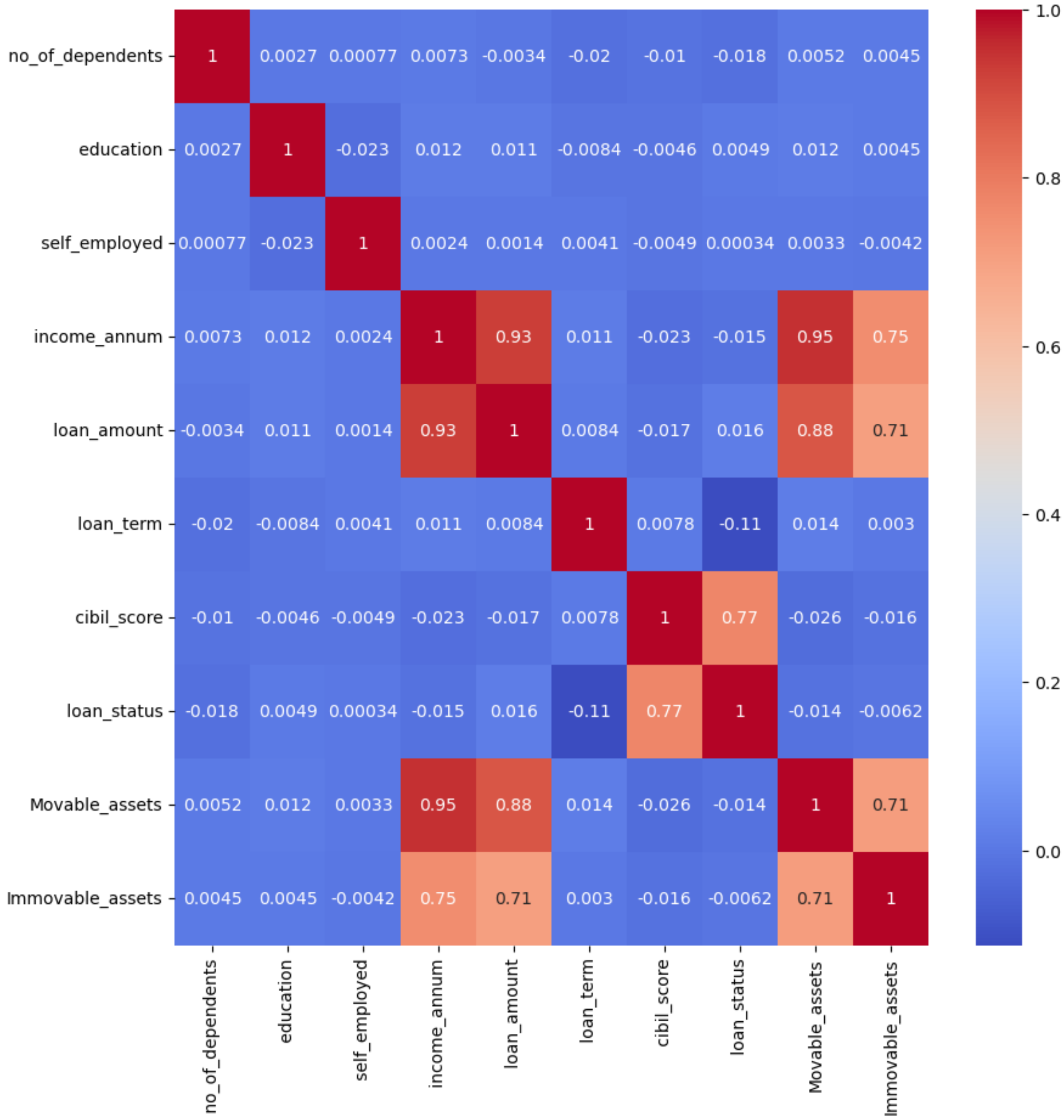
[25]:
```
   no_of_dependents  education  self_employed  income_annum  loan_amount  \
0                 2          1              0       9600000     29900000
1                 0          0              1       4100000     12200000
2                 3          1              0       9100000     29700000
3                 3          1              0       8200000     30700000
4                 5          0              1       9800000     24200000

   loan_term  cibil_score  loan_status  Movable_assets  Immovable_assets
0         12          778            1        30700000          20000000
1          8          417            0        12100000           4900000
2         20          506            0        46100000          11600000
3          8          467            0        31200000          21500000
```

4	20	382	0	34400000	20600000

Correlation Matrix Heatmap

[26]:
```
plt.figure(figsize = (10,10))
sns.heatmap(df.corr(), annot = True, cmap = 'coolwarm')
```

[26]: <Axes: >



This coorelation matrix heatmap has the folowing strong correlations: 1. Movable Assets and Immovable Assets 2. Income and Movable Assets 3. Income and Immovable Assets 4. Movable Assets and Loan Amount 5. Immovable Assets and Loan Amount 6. Loan Status and Cibil Score

7. Loan Amount and Income

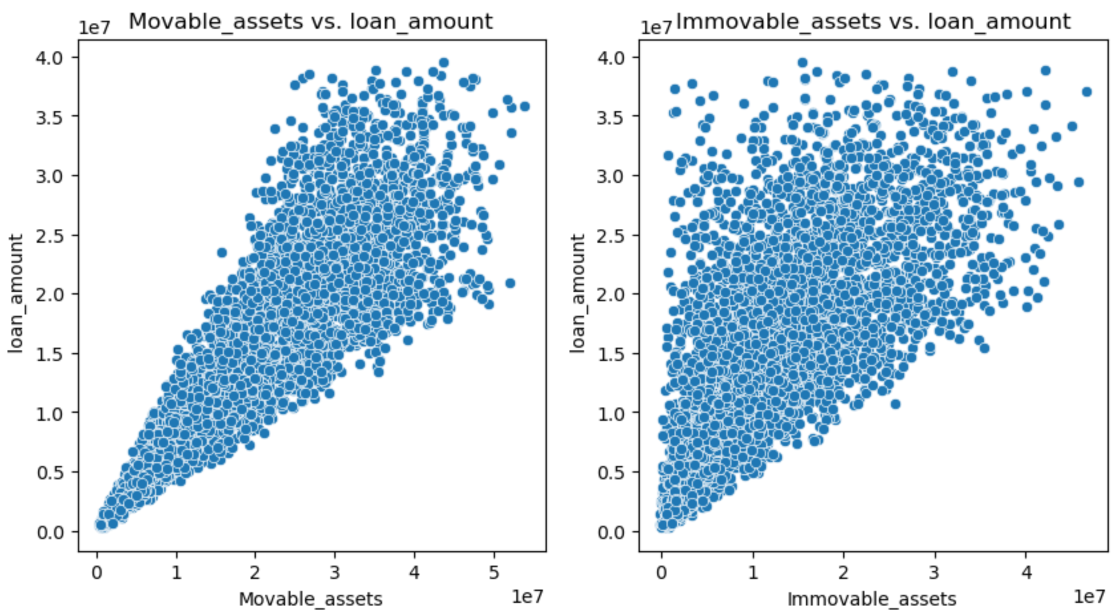
The coorelation between the movable and immovable assets is justified because both come under assets and its obvious that person with more movable assets will have more immovable assets and vice versa. Same is with Income and Movables and Immovale assets. The person with greater income will have greater assets.

Now, I will be exploring the coorleation between Assets and Loan Amount, and also between Income and Loan Amount. The relation between the loan status and cibil score is already explored in the previous section.

Assets vs. Loan Amount

[27]:
```
fig, ax = plt.subplots(1,2,figsize = (10,5))
sns.scatterplot(x = 'Movable_assets', y = ' loan_amount', data= df, ax = ax[0]).
 ↪set_title('Movable_assets vs. loan_amount')
sns.scatterplot(x = 'Immovable_assets', y = ' loan_amount', data= df, ax =␣
 ↪ax[1]).set_title('Immovable_assets vs. loan_amount')
```

[27]:
```
Text(0.5, 1.0, 'Immovable_assets vs. loan_amount')
```

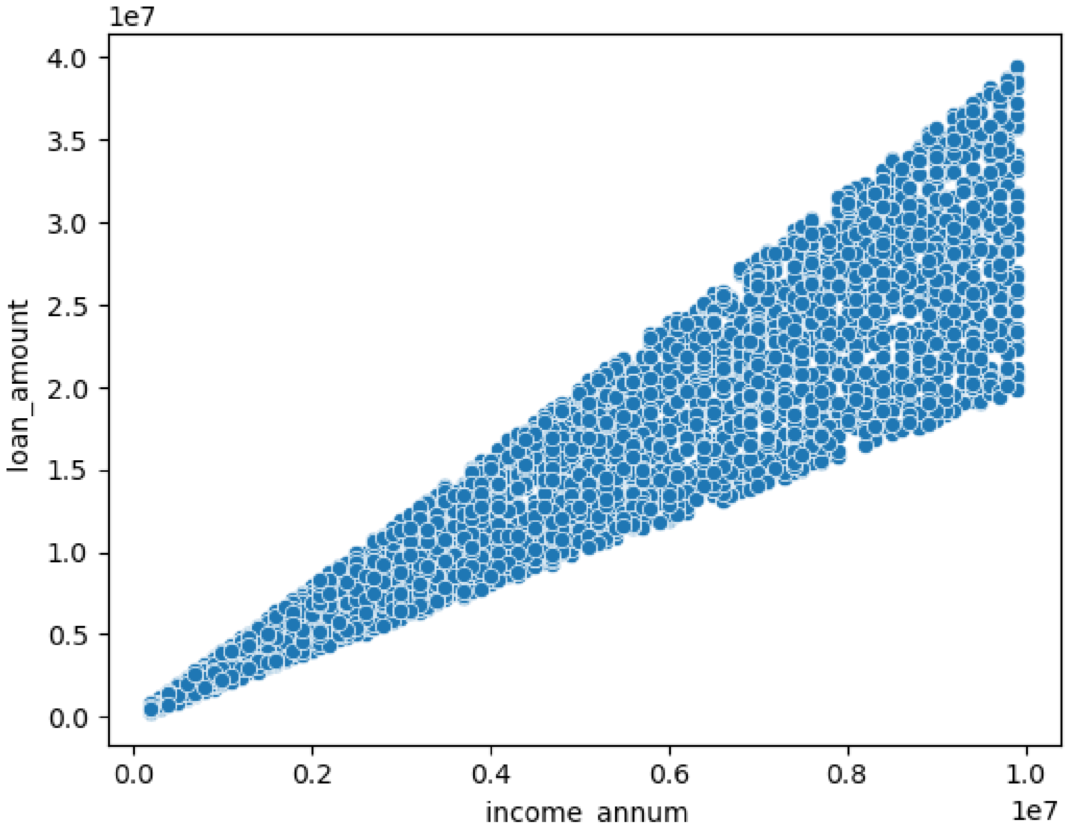


The loan amount has positive relation with movable and immovable assets. The more the assets, the more the loan amount issued by the bank.

Loan Amount vs. Income

[28]:
```
sns.scatterplot(x = ' income_annum', y = ' loan_amount', data=df)
```

[28]:
```
<Axes: xlabel=' income_annum', ylabel=' loan_amount'>
```



The loan amount and applicant's annual income have a very direct relation between them. The higher the income, the higher the loan amount. This is because the applicant's income is the main factor in deciding the how much loan needed.

Train Test Split

[29]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(df.drop(' loan_status',␣
↪axis = 1), df[' loan_status'], test_size = 0.2, random_state = 42)
```

Model Building

I will be using the following machine learning models to predcit the loan approval status: 1. Decision Tree Classifier 2. Random Forest Classifier

1. Decision Tree Classifier

[30]:
```
from sklearn.tree import DecisionTreeClassifier

# Create Decision Tree Object

d_tree = DecisionTreeClassifier()
```

[31]:
```python
# Training the model using the training data

d_tree.fit(x_train, y_train)
```

[31]: DecisionTreeClassifier()

[32]:
```python
# Training Accuracy

d_tree.score(x_train, y_train)
```

[32]: 1.0

[33]:
```python
# Predicting the Loan Approval Status

d_tree_pred = d_tree.predict(x_test)
```

2. Random Forest Classifier

[34]:
```python
from sklearn.ensemble import RandomForestClassifier

# Create a Random Forest Classifier

rfc = RandomForestClassifier()
```

[35]:
```python
# Training the model using the training data

rfc.fit(x_train, y_train)
```

[35]: RandomForestClassifier()

[36]:
```python
# Training Accuracy

rfc.score(x_train, y_train)
```

[36]: 1.0

[37]:
```python
# Predicting the Loan Approval Status

rfc_pred = rfc.predict(x_test)
```

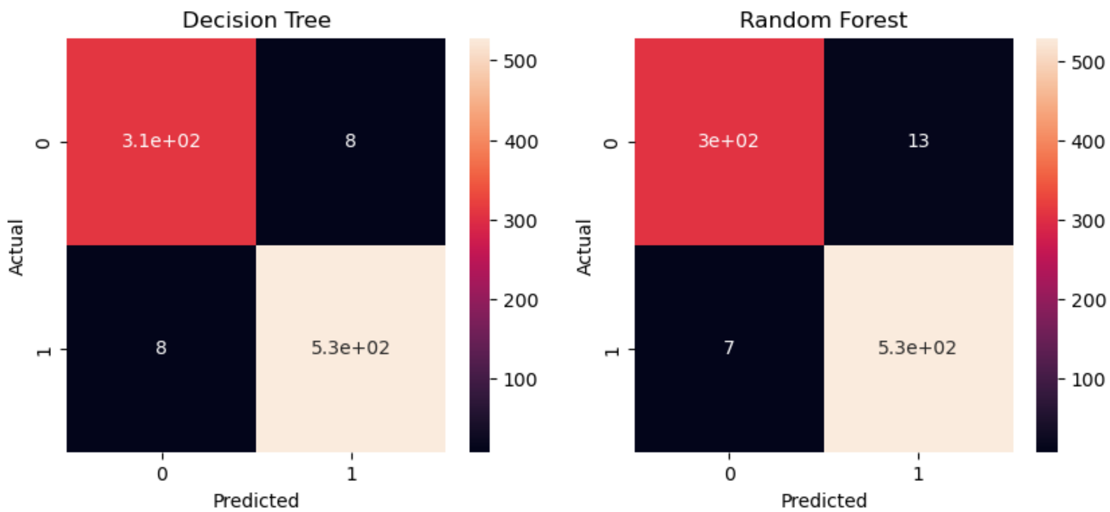
Model Evaluation

Confusion Matrix

[38]:
```python
from sklearn.metrics import confusion_matrix

fig, ax = plt.subplots(1,2, figsize = (10,4))
sns.heatmap(confusion_matrix(y_test, d_tree_pred), annot = True, ax = ax[0]).
 ↪set_title('Decision Tree')
```

```
ax[0].set_xlabel('Predicted')
ax[0].set_ylabel('Actual')

sns.heatmap(confusion_matrix(y_test, rfc_pred), annot = True, ax = ax[1]).
 ↪set_title('Random Forest')
ax[1].set_xlabel('Predicted')
ax[1].set_ylabel('Actual')
```

[38]: Text(518.4494949494949, 0.5, 'Actual')

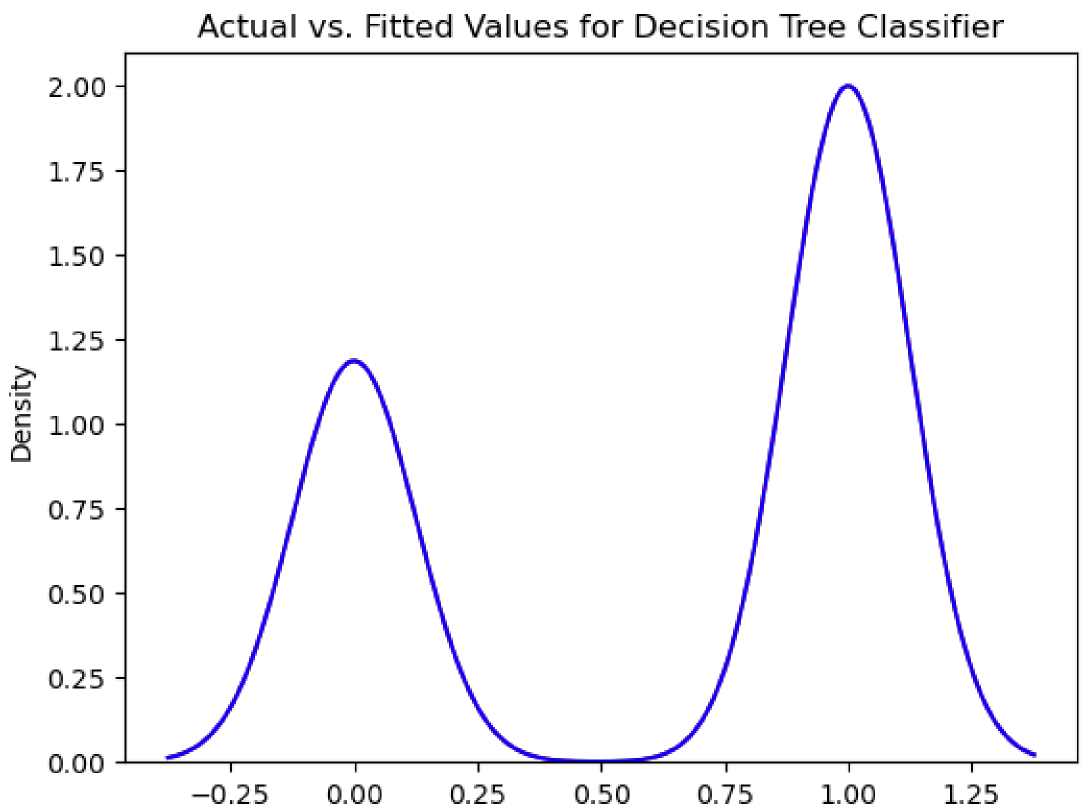


The above confusion matrix heatmap visualizes the the true positive and true negative value counts in both the machine learning models. The decision tree classfier has only 17 false positve and negative valyes where has random forest classifier has 21 false postive and negative values. The decision tree classifier has a better accuracy compared to random forest classifier.

Distribution Plot

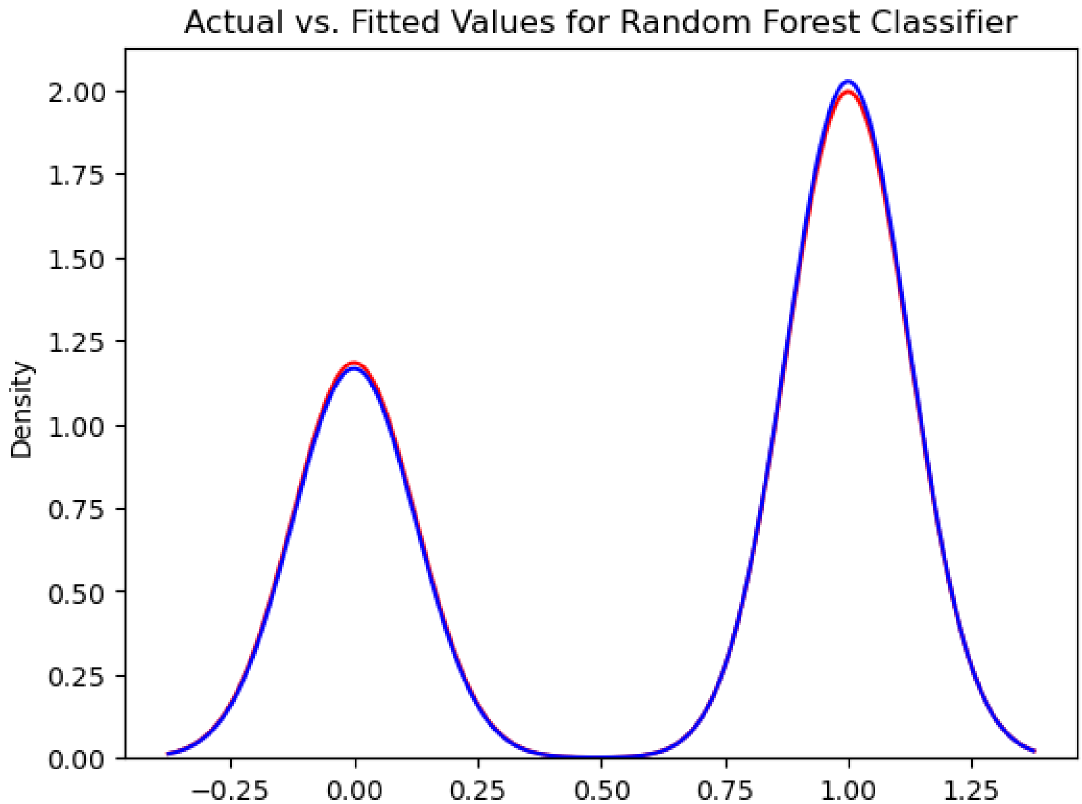
[39]:
```
ax = sns.distplot(x = y_test, hist = False, color = 'r', label = "Actual␣
 ↪Values")
sns.distplot(x = d_tree_pred, hist = False, color = 'b', label = 'Fitted␣
 ↪Values', ax = ax)
plt.title('Actual vs. Fitted Values for Decision Tree Classifier')
```

[39]: Text(0.5, 1.0, 'Actual vs. Fitted Values for Decision Tree Classifier')



[40]:
```
ax = sns.distplot(x = y_test, hist = False, color = 'r', label = "Actual␣
↪Values")
sns.distplot(x = rfc_pred, hist = False, color = 'b', label = 'Fitted Values',␣
↪ax = ax)
plt.title('Actual vs. Fitted Values for Random Forest Classifier')
```

[40]:
```
Text(0.5, 1.0, 'Actual vs. Fitted Values for Random Forest Classifier')
```



The distribution plot of both the models are almost same. There is very minute difference in the distribution density of the predicted and actual values in the random forest classifer. However, in case of decision tree classifier, the distribution density of the predicted values clearly overlaps with the actual values. Hence, we can say that the decision tree classifier is a better model than the random forest classifier for this dataset.

1.4.1 Classification Report

[41]:
```python
from sklearn.metrics import classification_report

print(classification_report(y_test, d_tree_pred))
print(classification_report(y_test, rfc_pred))
```

```
              precision    recall  f1-score   support

           0       0.97      0.97      0.97       318
           1       0.99      0.99      0.99       536

    accuracy                           0.98       854
   macro avg       0.98      0.98      0.98       854
weighted avg       0.98      0.98      0.98       854
```

```
              precision    recall  f1-score   support

           0       0.98      0.96      0.97       318
           1       0.98      0.99      0.98       536

    accuracy                           0.98       854
   macro avg       0.98      0.97      0.97       854
weighted avg       0.98      0.98      0.98       854
```

[42]:
```python
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error

# Decision Tree Classifier
print('R2 Score: ', r2_score(y_test, d_tree_pred))
print('Mean Squared Error: ', mean_squared_error(y_test, d_tree_pred))
print('Mean Absolute Error: ', mean_absolute_error(y_test, d_tree_pred))
print('\n')


# Random Forest Classifier
print('R2 Score: ', r2_score(y_test, rfc_pred))
print('Mean Squared Error: ', mean_squared_error(y_test, rfc_pred))
print('Mean Absolute Error: ', mean_absolute_error(y_test, rfc_pred))
```

```
R2 Score:  0.9198347883225382
Mean Squared Error:  0.01873536299765808
Mean Absolute Error:  0.01873536299765808


R2 Score:  0.8997934854031728
Mean Squared Error:  0.0234192037470726
Mean Absolute Error:  0.0234192037470726
```

From all the above metrics, graphs and reports, I conclude that descision tree classifier is a better machine learning model to predict the loan approval status of a person.

1.5 Conclusion

From the exploratory data analysis, we can conclude that the following factors are important for the approval of loan: - CIBIL Score: People with higher CIBIL score have higher chances of loan approval - Number of Dependents: People with more number of dependents have less chances of loan approval - Assets: People with more assets (including movable and immovable) have higher chances of loan approval - Loan Amount and Tenure: People with higher loan amount and lower tenure have more chances of loan approval

Coming to the machine learning models, I have used Decision Tree Classifier and Random Forest Classifier. Both the models have given excellent results having accuracies - 91.4 % and 89.4 % repectively. But the decision tree classifier has yielded better results than the random forest classifier.

```
[ ]:
```