LOAN INTEREST RATE PREDICTION

Feature	Definition
Loan_ID	A unique id for the loan.
Loan_Amount_Requested	The listed amount of the loan applied for by the borrower.
Length_Employed	Employment length in years
Home_Owner	The home ownership status provided by the borrower during registration. Values are: Rent, Own, Mortgage, Other.
Annual_Income	The annual income provided by the borrower during registration.
Income_Verified	Indicates if income was verified, not verified, or if the income source was verified
Purpose_Of_Loan	A category provided by the borrower for the loan request.
Debt_To_Income	A ratio calculated using the borrower's total monthly debt payments on the total debt obligations, excluding mortgage and the requested loan, divided by the borrower's self-reported monthly income.
Inquiries_Last_6Mo	The number of inquiries by creditors during the past 6 months.
Months_Since_Deliquency	The number of months since the borrower's last delinquency.
Number_Open_Accounts	The number of open credit lines in the borrower's credit file.
Total_Accounts	The total number of credit lines currently in the borrower's credit file
Gender	Gender
Target	**Definition**
Interest_Rate	Interest Rate category (1/2/3) of the loan application

Import Package

In [145]:
```
import pandas as pd
import numpy as np
import seaborn as sn
import matplotlib.pyplot as plt

from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, RidgeCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.metrics import mean_squared_error

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

Menggunakan beberapa package pengolahan data, matematika/kalkulasi, visualisasi data, sklearn(membantu pemrosesan data, skoring data dan memodelkan data) dan warnings (menghilangakan alert/output danger).

Dataset yang disediakan :

1. Train.csv (Data Train)
2. Test.csv (Data Test)

Data Exploration

```
In [148]: train.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 164309 entries, 0 to 164308
Data columns (total 14 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   Loan_ID                 164309 non-null  int64
 1   Loan_Amount_Requested   164309 non-null  object
 2   Length_Employed         156938 non-null  object
 3   Home_Owner              138960 non-null  object
 4   Annual_Income           139207 non-null  float64
 5   Income_Verified         164309 non-null  object
 6   Purpose_Of_Loan         164309 non-null  object
 7   Debt_To_Income          164309 non-null  float64
 8   Inquiries_Last_6Mo      164309 non-null  int64
 9   Months_Since_Deliquency 75930 non-null   float64
 10  Number_Open_Accounts    164309 non-null  int64
 11  Total_Accounts          164309 non-null  int64
 12  Gender                  164309 non-null  object
 13  Interest_Rate           164309 non-null  int64
dtypes: float64(3), int64(5), object(6)
memory usage: 17.6+ MB
```

Dari informasi diatas kita bisa simpulkan tipe data masing-masing column dan ada beberapa column yang memiliki data yang bernilai null. Berikut column-column yang harus kita proses :

1. Length_Employed
2. Home_Owner
3. Annual_Income
4. Months_Since_Deliquency

Column yang memiliki data kategorikal :

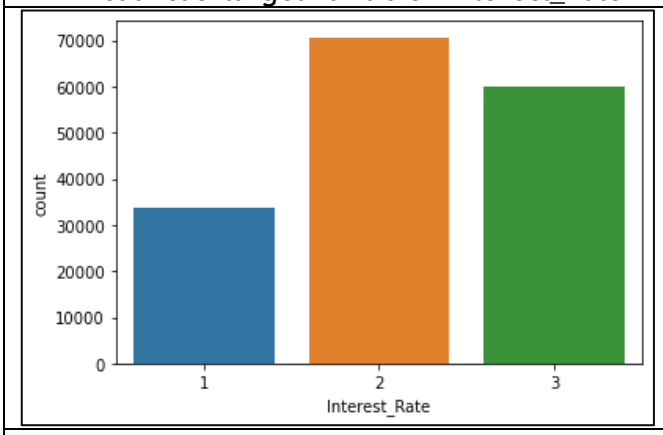
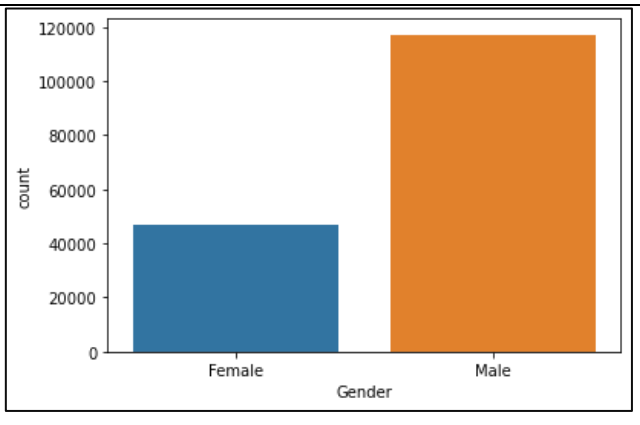
```
In [152]: train['Income_Verified'].value_counts()

Out[152]: VERIFIED - income           59421
          VERIFIED - income source    53015
          not verified                51873
          Name: Income_Verified, dtype: int64
```

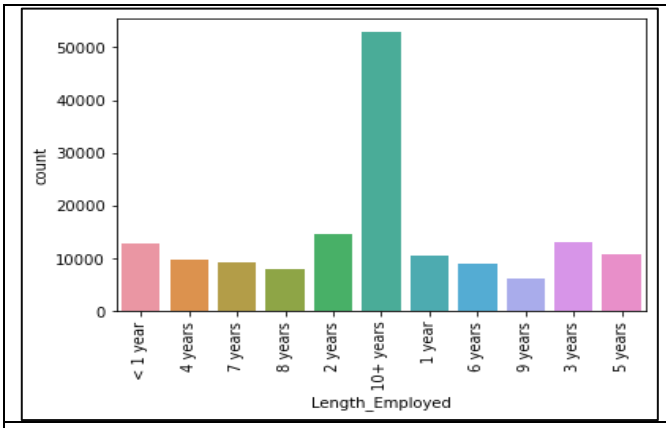
```
In [153]: train['Home_Owner'].value_counts()
Out[153]: Mortgage    70345
          Rent        56031
          Own         12525
          Other          49
          None           10
          Name: Home_Owner, dtype: int64
```

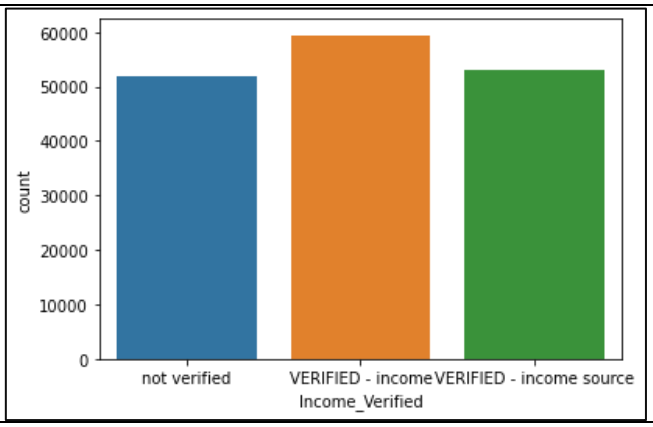
```
In [154]: train['Purpose_Of_Loan'].value_counts()
Out[154]: debt_consolidation    97101
          credit_card           36684
          home_improvement       9269
          other                  8346
          major_purchase         3435
          small_business         2392
          car                    1885
          medical                1541
          moving                  974
          vacation                837
          wedding                 828
          house                   773
          renewable_energy        123
          educational             121
          Name: Purpose_Of_Loan, dtype: int64
```

```
In [155]: train['Gender'].value_counts()
Out[155]: Male      117176
          Female     47133
          Name: Gender, dtype: int64
```

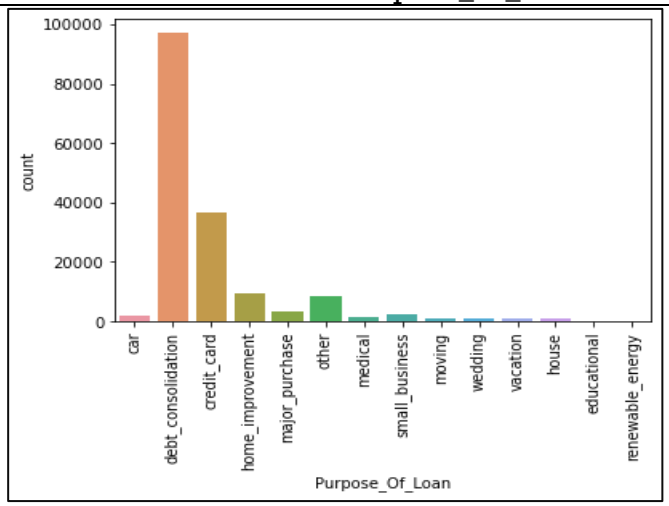
Hasil visualisasi column-column yang memiliki data kategorikal

Visualisasi target variable : Interest_Rate	Visualisasi Data Feature : Gender
	
Visualisasi Data Feature : Length_Employed	Visualisasi Data Feature : Income_Verified

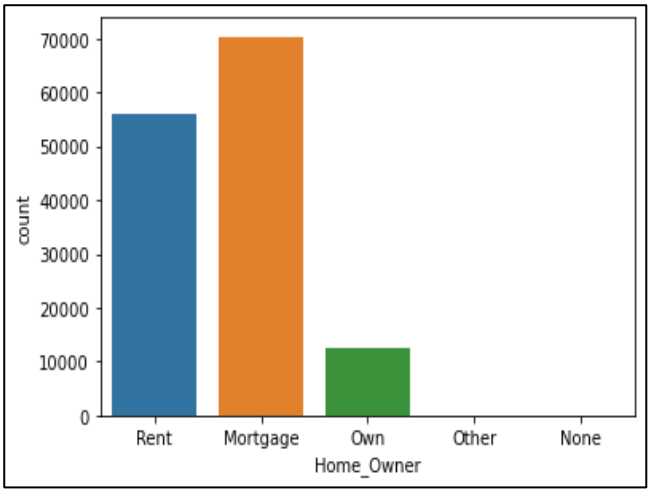




Visualisasi Data Feature : Purpose_Of_Loan



Visualisasi Data Feature : Home_Owner



Data Preprocessing

NO	FEATURE	NOTE
1	Loan_ID	Loan_ID didrop karena tidak meberikan nilai prediksi terhadapa target, column ini akan kita gunakan saat nilai prediktif dari column Interest_Rate telah dicapai.
2	Loan_Amount_Requested	Konversi tipe data menjadi float
3	Length_Employed	Konversi value column berdasarkan kondisi tertentu dan mengubah ke bentuk tipe data float
4	Home Owner	Mengisi data yang kosong sebagai “Own” karena yang paling mendekati missing value
5	Annual_Income	Secara langsung mengganti nilai yang kosong Annual_Income dengan median/mean/modus, tetapi berasumsi bahwa Length_Employed yang berbanding lurus dengan Annual_Income, karena pendapatan untuk 10+ tahun karyawan tidak akan sebanding dengan karyawan 1 tahun (tentu saja ada outlier) sehingga kita lebih memilih median daripada mean
6	Months_Since_Deliquency	Mengisi data yang kosong dengan median dari column Months_Since_Deliquency, karena data min-max data / beberapa data sangat jauh dari nilai mean.

Melakukan Model Selection

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=42,stratify=Train_dt_full['Interest_Rate'])
```

Train Size = 70 ; Test Size = 30;
Random State = 42 ;
Stratify = Interest_Rate

Data Modeling

```
RF_CLF=RandomForestClassifier(n_estimators=100,random_state=42)
RF_CLF.fit(X_train,y_train)
y_pred=RF_CLF.predict(X_test)
```

```
print('Accuracy Score for 100 estimators: ',accuracy_score(y_test,y_pred))
print(pd.crosstab(y_test,y_pred,rownames=['Actual Loan Category'],
colnames=['Predicted Loan Category']))
```

```
Accuracy Score for 100 estimators:  0.7776560566409023
Predicted Loan Category     1      2      3
Actual Loan Category
1                        7302   2173    667
2                         899  16978   3297
3                         431   3493  14053
```

Dari beberapa percobaan parameter n_estimator dan random state, maka pada saat n_estimator = 100 dan random_state = 42 memiliki accuracy score di **77%.**