Linear Regression - Jaswanth Galle

Predict Interest Rates From Loan Data

In this I tried to show few Visualizations to get Insights that can be done on Numerical ,Categorical Data

Handled Missing Values

Handled Outliers

Chisq Test for Categorical Variables

Build Linear Regression Model

Checking Assumptions of Linear Regression

In [673…

```
import pandas as pd
import numpy as np
```

In [674…

```
train_file='loan_data_train.csv'
test_file='loan_data_test.csv'

train_data=pd.read_csv(train_file)
test_data=pd.read_csv(test_file)
```

In [675…
```
train_data.head()
```

Out[675…

	ID	Amount.Requested	Amount.Funded.By.Investors	Interest.Rate	Loan.Length	Loan.Purpose	Debt.To.Income.Ratio	State
0	79542.0	25000	25000	18.49%	60 months	debt_consolidation	27.56%	VA
1	75473.0	19750	19750	17.27%	60 months	debt_consolidation	13.39%	NY
2	67265.0	2100	2100	14.33%	36 months	major_purchase	3.50%	LA
3	80167.0	28000	28000	16.29%	36 months	credit_card	19.62%	NV
4	17240.0	24250	17431.82	12.23%	60 months	credit_card	23.79%	OH

In [676…
```
# COMBINE BOTH TRAIN AND TEST DATA TO PERFORM FEATURE ENGINEERING

test_data['Interest.Rate']=np.nan
train_data['data']='train'
test_data['data']='test'
test_data=test_data[train_data.columns]
Combined_data=pd.concat([train_data,test_data],axis=0)
```

In [677…
```
Combined_data.dtypes
```

Out[677…
```
ID                              float64
Amount.Requested                 object
Amount.Funded.By.Investors       object
Interest.Rate                    object
Loan.Length                      object
Loan.Purpose                     object
Debt.To.Income.Ratio             object
State                            object
Home.Ownership                   object
Monthly.Income                  float64
FICO.Range                       object
Open.CREDIT.Lines                object
Revolving.CREDIT.Balance         object
Inquiries.in.the.Last.6.Months  float64
Employment.Length                object
data                             object
dtype: object
```

In [678…
```
# Dropping Unwanted variables
```

```
Combined_data.drop(['ID','Amount.Funded.By.Investors'],axis=1,inplace=True)
```

In [679…

```
# Removing % character in these variables 'Interest.Rate','Debt.To.Income.Ratio' to convert them to numeric

for col in ['Interest.Rate','Debt.To.Income.Ratio']:
    Combined_data[col]=Combined_data[col].str.replace("%","")
```

In [680…

```
# Converting to Numeric

for col in ['Amount.Requested', 'Interest.Rate','Debt.To.Income.Ratio',
            'Open.CREDIT.Lines','Revolving.CREDIT.Balance']:
    Combined_data[col]=pd.to_numeric(Combined_data[col],errors='coerce')
```

In [681…

```
# Removing '-' character in FICO score range and taking mean of the extreme values

credit_score=Combined_data['FICO.Range'].str.split("-",expand=True).astype(float)

Combined_data['fico_mean']=(credit_score[0]+credit_score[1])*(1/2)

del Combined_data['FICO.Range']
```

In [682…

```
Combined_data['Employment.Length'].value_counts()
```

Out[682…

```
10+ years    653
< 1 year     249
2 years      243
3 years      235
5 years      202
4 years      191
1 year       177
6 years      163
7 years      127
8 years      108
9 years       72
.              2
Name: Employment.Length, dtype: int64
```

In [683…

```
def remove_special_char(df,var):
    # we can write many rules to remove the special characters ,
    #this function helps to remove special characters
    df[var] = df[var].str.replace('years',"")
    df[var] = df[var].str.replace('year',"")
    df[var] = np.where(df[var].str[:2]=="10",10,df[var])
    df[var] = np.where(df[var].str[:2]=="<",0,df[var])
    df[var] = df[var].str.replace('>',"")
    df[var] = df[var].str.replace('-',"")

    return df
```

In [684…
```
Combined_data = remove_special_char(Combined_data,'Employment.Length')
```

In [685…
```
Combined_data ['Employment.Length']=pd.to_numeric(Combined_data ['Employment.Length'],errors='coerce')
```

DATA VISUALIZATION

In [686…
```
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

In [687…
```
Combined_data.reset_index(inplace=True)
```

In [688…
```
Combined_data.drop(['index'],axis=1,inplace=True)
```

In [689…
```
Combined_data.head()
```

Out[689…

	Amount.Requested	Interest.Rate	Loan.Length	Loan.Purpose	Debt.To.Income.Ratio	State	Home.Ownership	Monthly.Income	C
0	25000.0	18.49	60 months	debt_consolidation	27.56	VA	MORTGAGE	8606.56	
1	19750.0	17.27	60 months	debt_consolidation	13.39	NY	MORTGAGE	6737.50	
2	2100.0	14.33	36 months	major_purchase	3.50	LA	OWN	1000.00	

	Amount.Requested	Interest.Rate	Loan.Length	Loan.Purpose	Debt.To.Income.Ratio	State	Home.Ownership	Monthly.Income	C
3	28000.0	16.29	36 months	credit_card	19.62	NV	MORTGAGE	7083.33	
4	24250.0	12.23	60 months	credit_card	23.79	OH	MORTGAGE	5833.33	

Missing Data

In [690…

```
Combined_data_na = (Combined_data.isnull().sum() / len(Combined_data)) * 100
null_data = pd.DataFrame({'Missing Values' :Combined_data_na})
null_data.head(30)
```

Out[690…

	Missing Values
Amount.Requested	0.20
Interest.Rate	12.00
Loan.Length	0.04
Loan.Purpose	0.04
Debt.To.Income.Ratio	0.04
State	0.04
Home.Ownership	0.04
Monthly.Income	0.12
Open.CREDIT.Lines	0.36
Revolving.CREDIT.Balance	0.20
Inquiries.in.the.Last.6.Months	0.12
Employment.Length	39.28
data	0.00
fico_mean	0.00

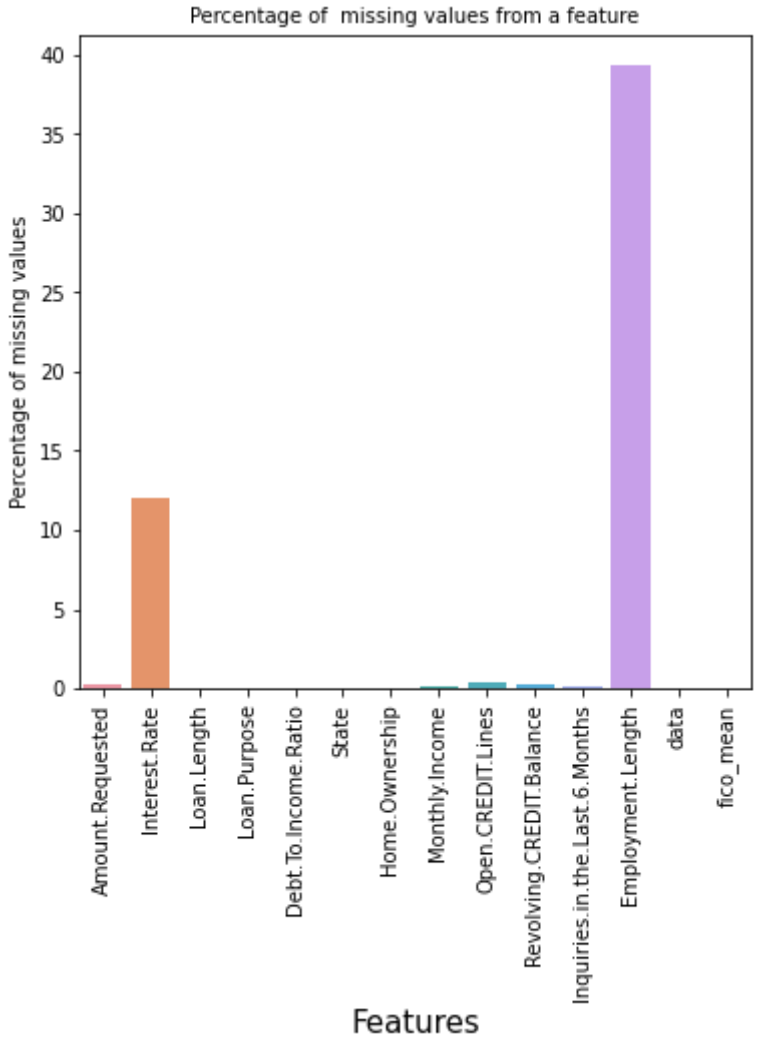
In [691…

```
null_data.index.values.tolist()
null_df = Combined_data[null_data.index.values.tolist()]
null_numerical = null_df.select_dtypes(include=np.number).columns.tolist()
```

```
f, ax = plt.subplots(figsize=(6, 6))
plt.xticks(rotation='90')
sns.barplot(x=Combined_data_na.index, y=Combined_data_na)
plt.xlabel('Features', fontsize=15)
plt.ylabel('Percentage of missing values', fontsize=10)
plt.title('Percentage of  missing values from a feature', fontsize=10)
```

Out[691…

```
Text(0.5, 1.0, 'Percentage of  missing values from a feature')
```



In [692…

```
#there are high number of missing values is Interest rate as we merged test data to train
```

Probability density plots

In [693…

```
numeric_columns = Combined_data.select_dtypes(include=['float64']).columns
```

In [694…

```
numeric_columns
```

Out[694…

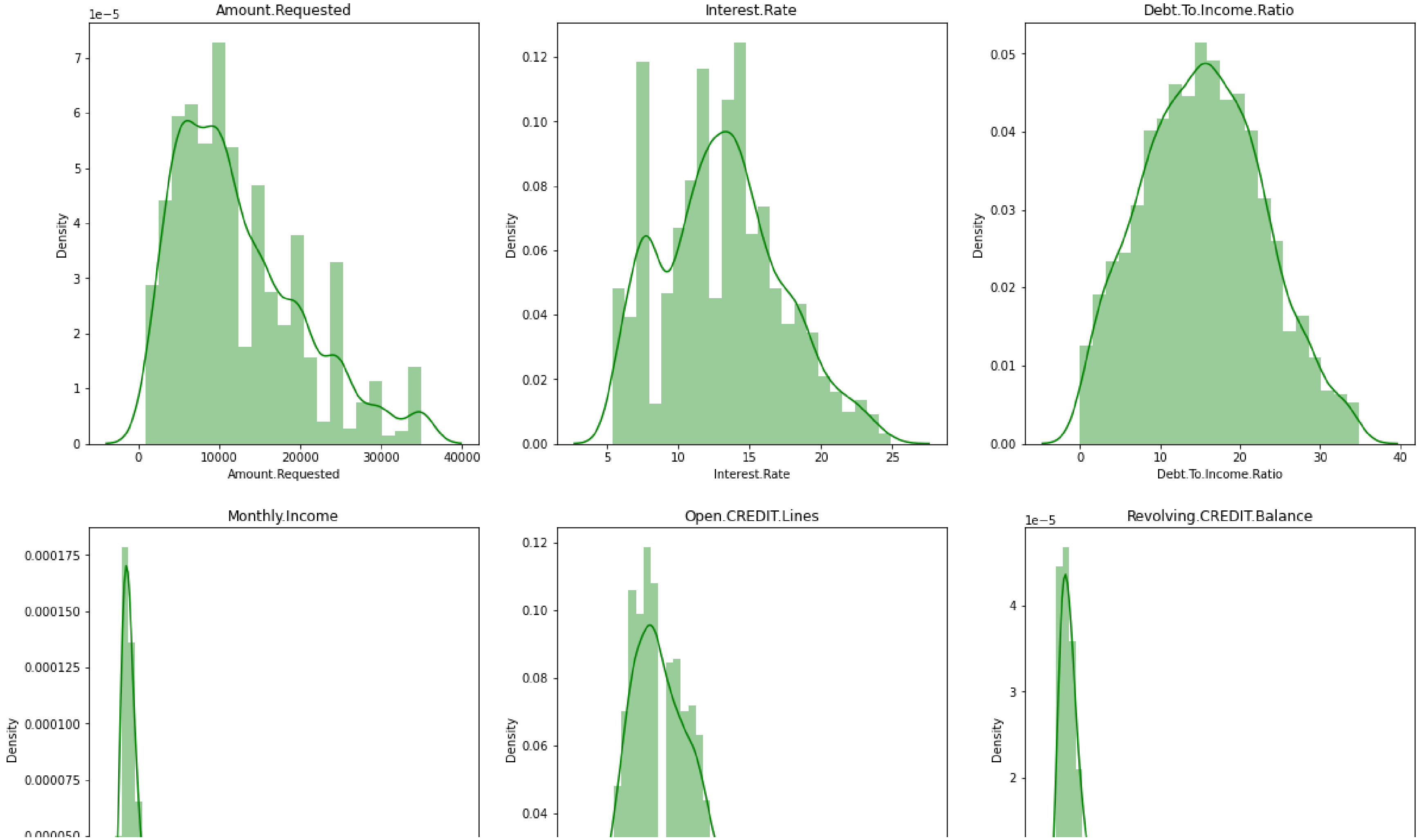
```
Index(['Amount.Requested', 'Interest.Rate', 'Debt.To.Income.Ratio',
       'Monthly.Income', 'Open.CREDIT.Lines', 'Revolving.CREDIT.Balance',
       'Inquiries.in.the.Last.6.Months', 'Employment.Length', 'fico_mean'],
      dtype='object')
```

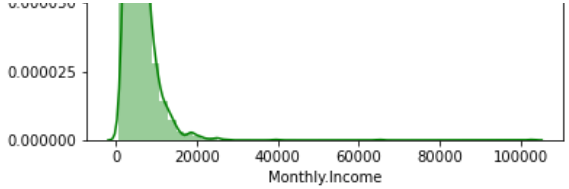
In [695…

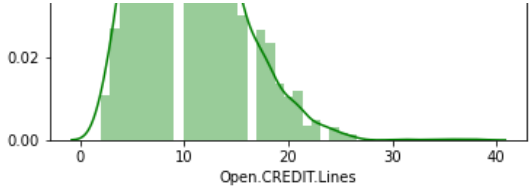
```
fig, axes = plt.subplots(nrows = 4, ncols = 3)    # axes is 2d array (3x3)
axes = axes.flatten()         # Convert axes to 1d array of length 9
fig.set_size_inches(20, 30)

for ax, col in zip(axes, Combined_data[numeric_columns].columns):
    sns.distplot(Combined_data[numeric_columns][col], ax = ax, color='green')
    ax.set_title(col)
```

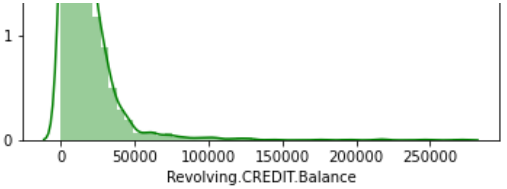
```
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `dis
plot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `dis
plot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `dis
plot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `dis
plot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `dis
plot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `dis
```

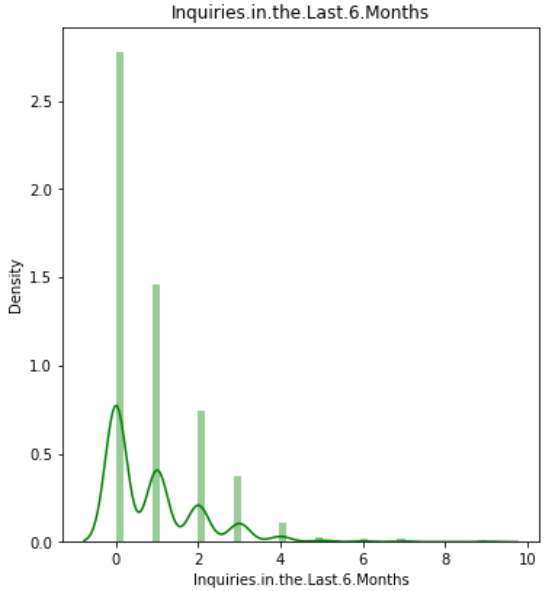
```
plot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `dis
plot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `dis
plot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `dis
plot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

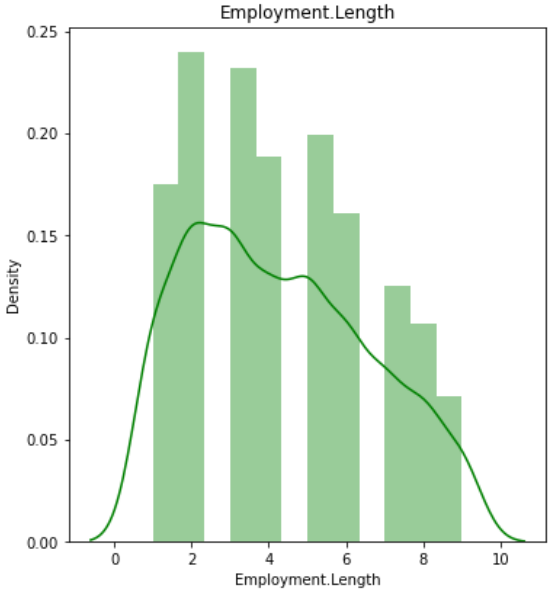


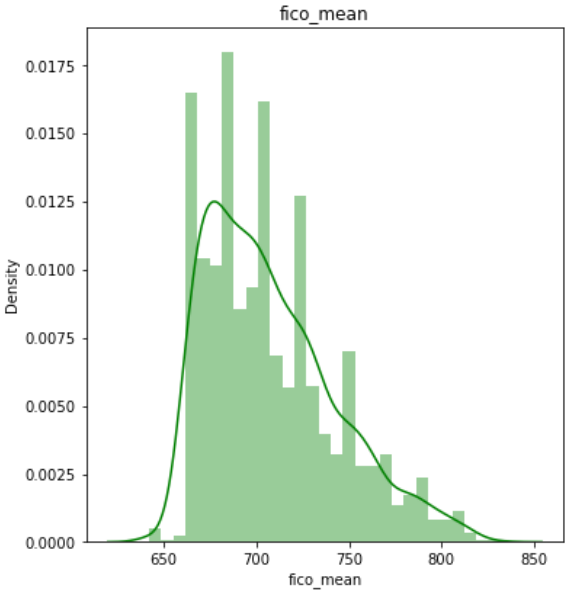

0.000025
0.000000
0
20000
40000
60000
80000
100000
Monthly.Income

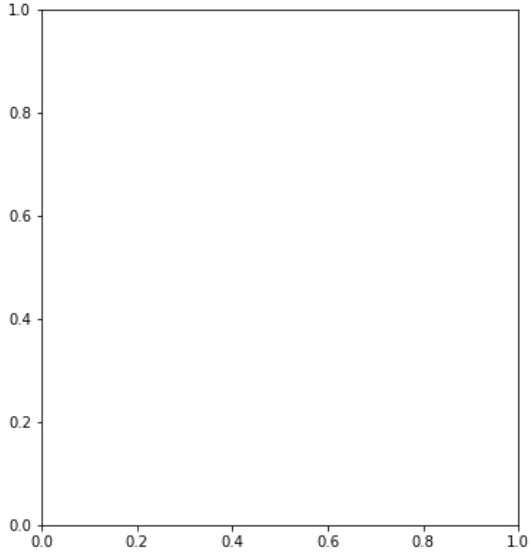

0.02
0.00
0
10
20
30
40
Open.CREDIT.Lines

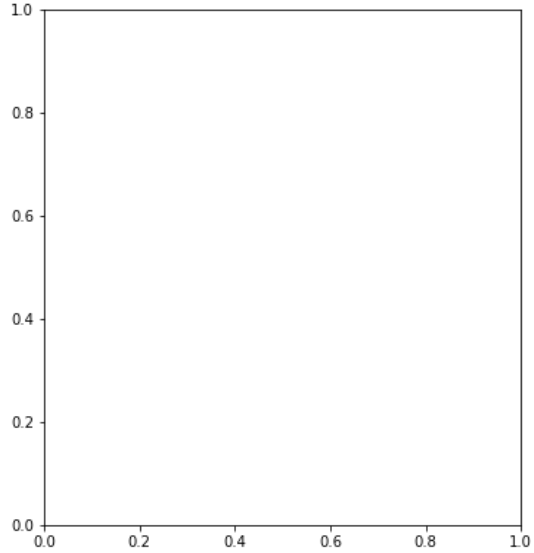

1
0
0
50000
100000
150000
200000
250000
Revolving.CREDIT.Balance

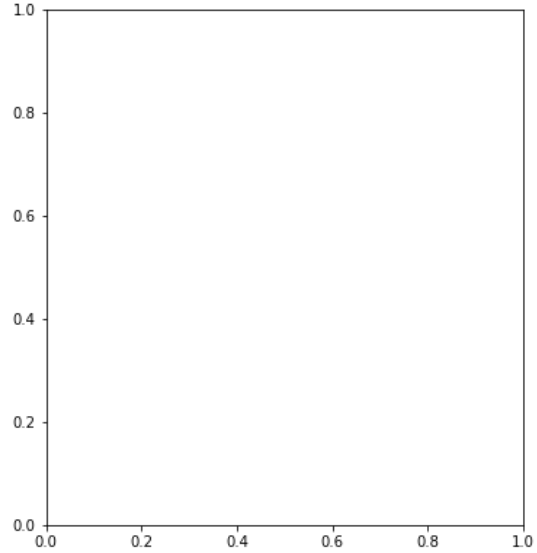

Inquiries.in.the.Last.6.Months
Density
2.5
2.0
1.5
1.0
0.5
0.0
0
2
4
6
8
10
Inquiries.in.the.Last.6.Months


Employment.Length
Density
0.25
0.20
0.15
0.10
0.05
0.00
0
2
4
6
8
10
Employment.Length


fico_mean
Density
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
650
700
750
800
850
fico_mean


1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

In [696…

```
# we can observe there is positive skewness in FICO variable and the data points in Employmentlength
#variable is widely spread, few columns like Deb to income ratio , monthly income are normally distributed
```

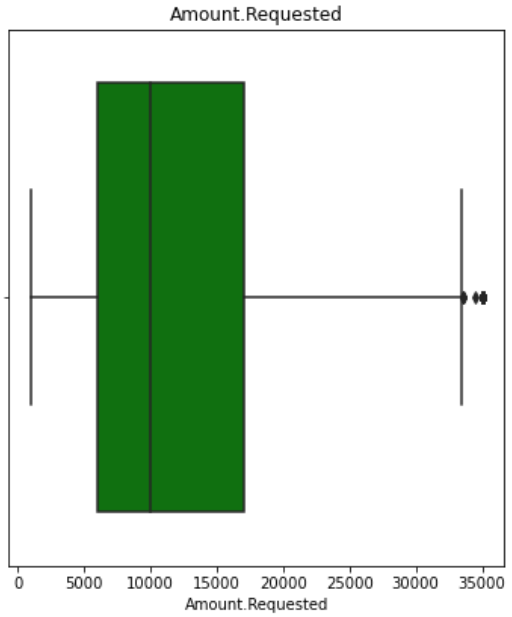
Let's See BOXPLOTS For These Variables

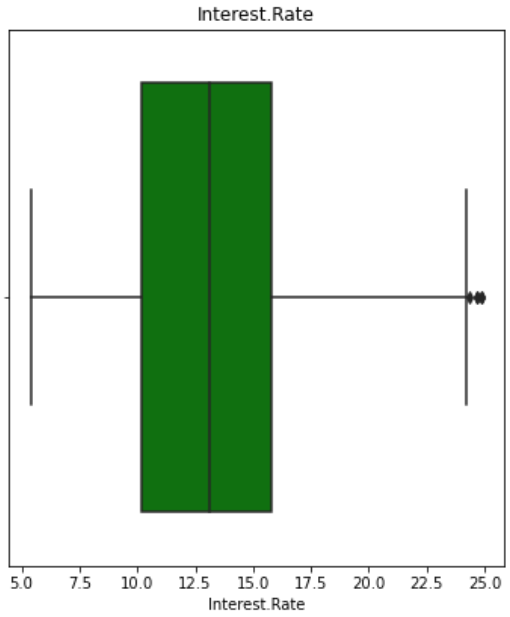
In [697…

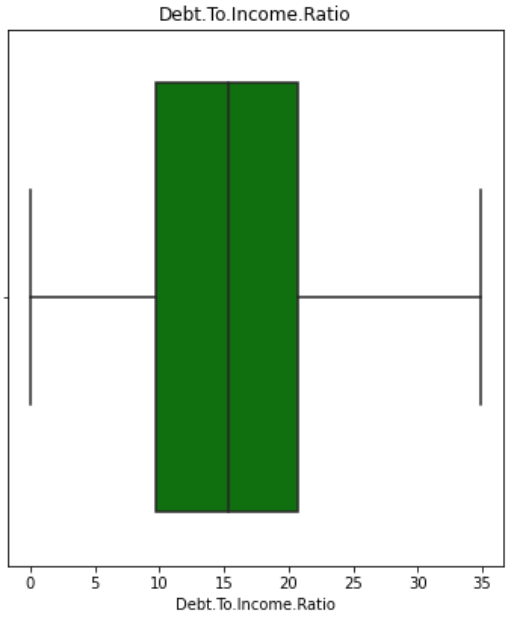
```
fig, axes = plt.subplots(nrows = 4, ncols = 3)    # axes is 2d array (3x3)
axes = axes.flatten()         # Convert axes to 1d array of length 9
fig.set_size_inches(20, 30)

for ax, col in zip(axes, Combined_data[numeric_columns].columns):
    sns.boxplot(Combined_data[numeric_columns][col], ax = ax, color='green')
    ax.set_title(col)
```

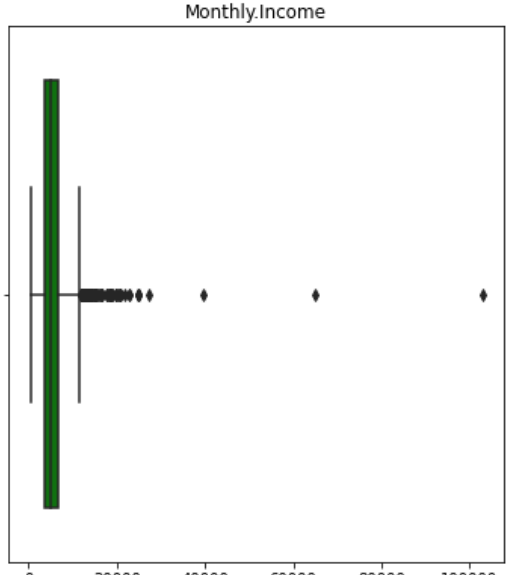
```
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

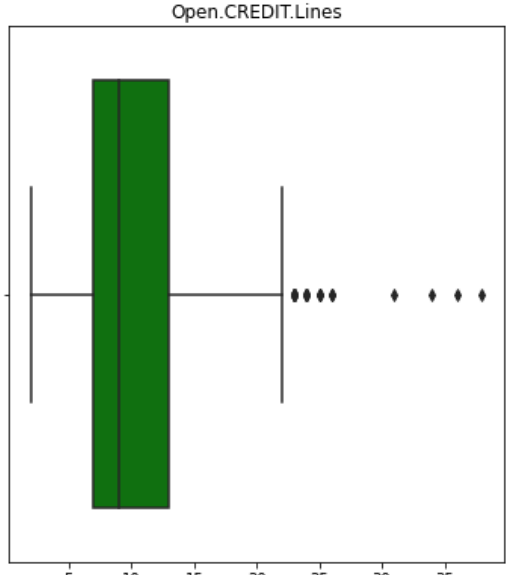
```
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the fo
llowing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

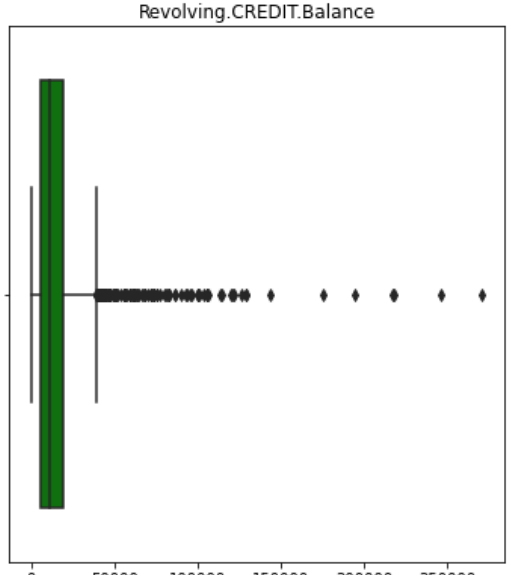


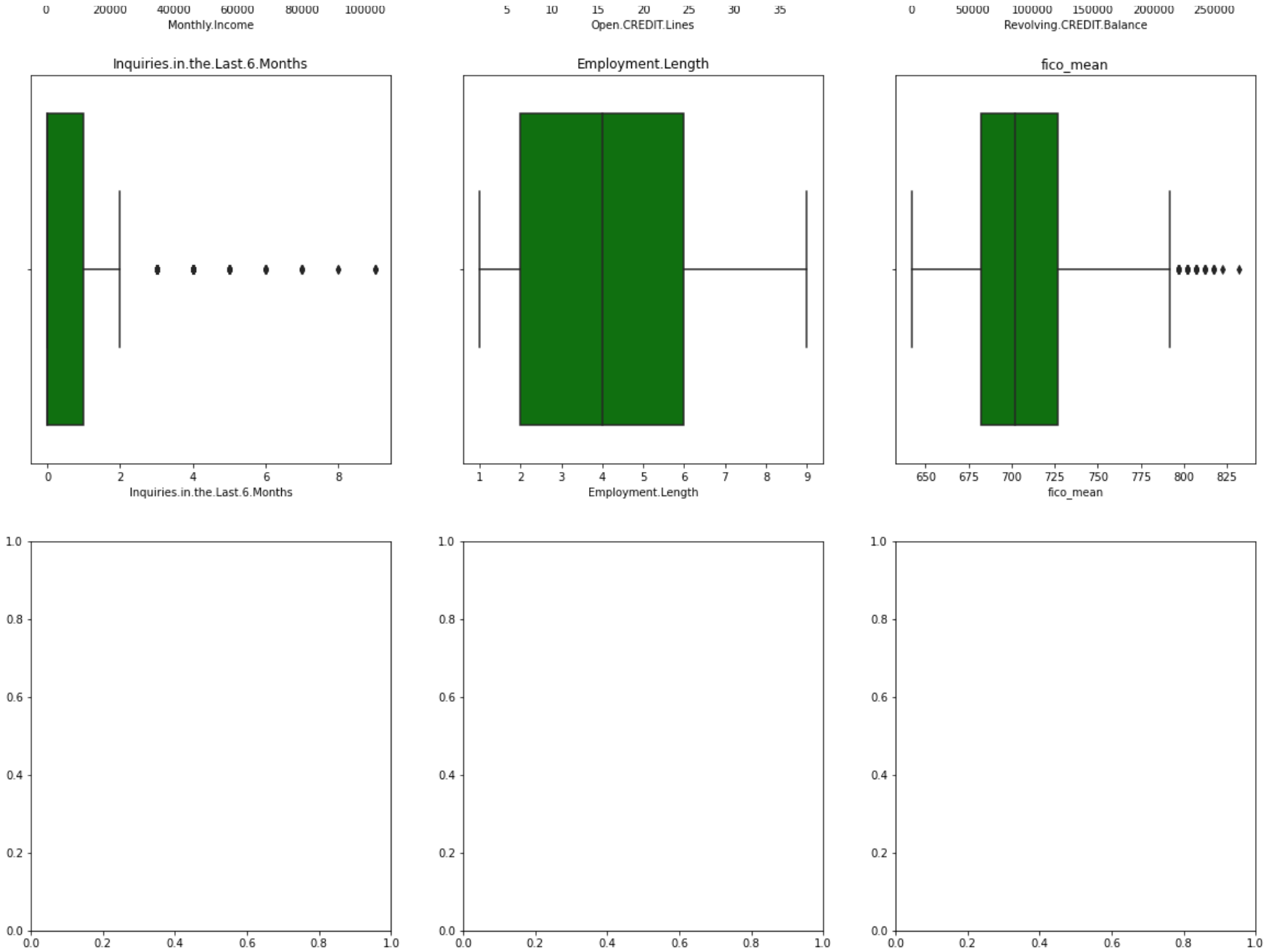












In [698…

```
#we can observe there are high number of outliers  in Revolving.CREDIT.Balance,Open Credit Lines ,
#Montly Income, Fico variables
```

In [699…
```
Combined_data[numeric_columns].head()
```

Out[699…

	Amount.Requested	Interest.Rate	Debt.To.Income.Ratio	Monthly.Income	Open.CREDIT.Lines	Revolving.CREDIT.Balance	Inquiries.in.t
0	25000.0	18.49	27.56	8606.56	11.0	15210.0	
1	19750.0	17.27	13.39	6737.50	14.0	19070.0	
2	2100.0	14.33	3.50	1000.00	13.0	893.0	
3	28000.0	16.29	19.62	7083.33	12.0	38194.0	
4	24250.0	12.23	23.79	5833.33	6.0	31061.0	

In [700…
```
Combined_data.columns
```

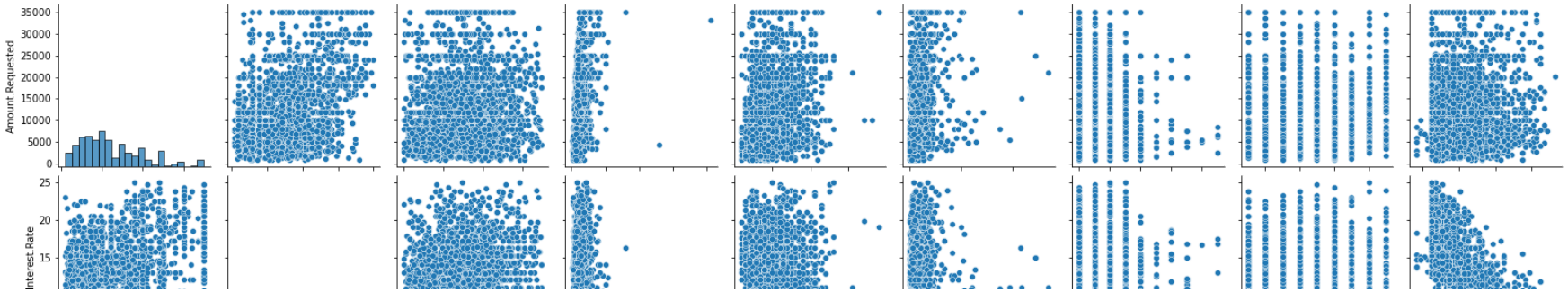
Out[700…
```
Index(['Amount.Requested', 'Interest.Rate', 'Loan.Length', 'Loan.Purpose',
       'Debt.To.Income.Ratio', 'State', 'Home.Ownership', 'Monthly.Income',
       'Open.CREDIT.Lines', 'Revolving.CREDIT.Balance',
       'Inquiries.in.the.Last.6.Months', 'Employment.Length', 'data',
       'fico_mean'],
      dtype='object')
```

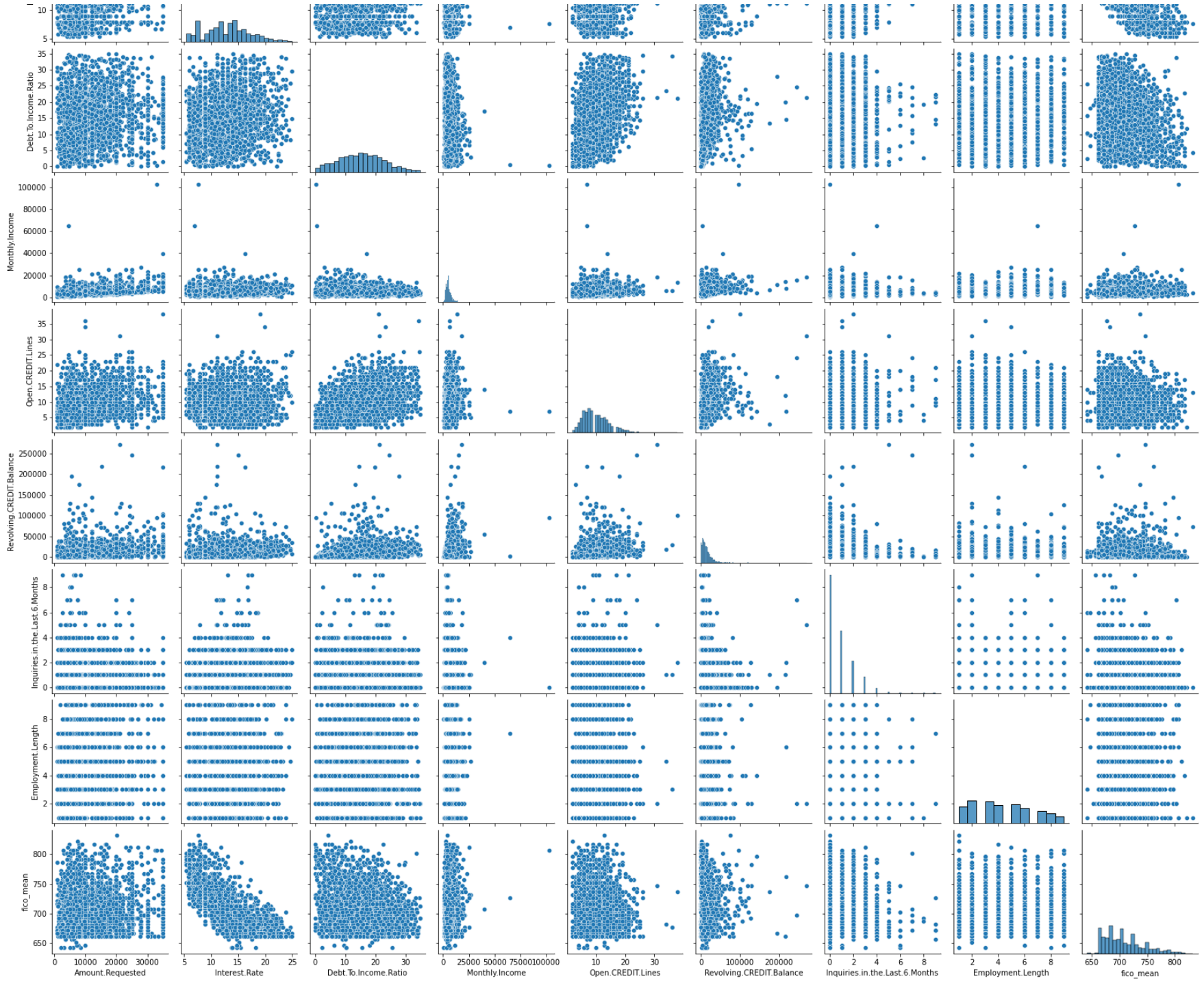
In [701…
```
numeric_columns
```

Out[701…
```
Index(['Amount.Requested', 'Interest.Rate', 'Debt.To.Income.Ratio',
       'Monthly.Income', 'Open.CREDIT.Lines', 'Revolving.CREDIT.Balance',
       'Inquiries.in.the.Last.6.Months', 'Employment.Length', 'fico_mean'],
      dtype='object')
```

In [702…
```
sns.pairplot(Combined_data[numeric_columns])
```

Out[702…
```
<seaborn.axisgrid.PairGrid at 0x7fe1d8c37280>
```




Debt.To.Income.Ratio
Monthly.Income
Open.CREDIT.Lines
Revolving.CREDIT.Balance
Inquiries.in.the.Last.6.Months
Employment.Length
fico_mean
Amount.Requested
Interest.Rate
Debt.To.Income.Ratio
Monthly.Income
Open.CREDIT.Lines
Revolving.CREDIT.Balance
Inquiries.in.the.Last.6.Months
Employment.Length
fico_mean

From the above pair plots we can see that there are few columns ,

which follow similar kind of pattern , 'Amount.Requested','Debt.To.Income.Ratio' are following same kind of pattern

We can observe few columns which are following similar kind of pattern w.r.t to the target variable

In [ ]:

In [ ]:

Removing Outliers

In [703…
```
#replace outliers with mean value
def replace_outliers(df,var):
    df_var = df[var]
    q1=df_var.quantile(0.25)
    q2 = df_var.quantile(0.5)
    q3=df_var.quantile(0.75)

    IQR=q3-q1
    df[var]= df[var].apply(lambda x : q2 if x<(q1-1.5*IQR) or (x> (q1+1.5*IQR)) else x )
    return df
```

In [704…
```
Combined_data.columns
```

Out[704…
```
Index(['Amount.Requested', 'Interest.Rate', 'Loan.Length', 'Loan.Purpose',
       'Debt.To.Income.Ratio', 'State', 'Home.Ownership', 'Monthly.Income',
       'Open.CREDIT.Lines', 'Revolving.CREDIT.Balance',
```

```
       'Inquiries.in.the.Last.6.Months', 'Employment.Length', 'data',
       'fico_mean'],
      dtype='object')
```

In [705…

```
Combined_data=replace_outliers(Combined_data,'Revolving.CREDIT.Balance')
Combined_data=replace_outliers(Combined_data,'fico_mean')
Combined_data=replace_outliers(Combined_data,'Inquiries.in.the.Last.6.Months')
Combined_data=replace_outliers(Combined_data,'Monthly.Income')
Combined_data=replace_outliers(Combined_data,'Open.CREDIT.Lines')
```

Correlation

In [706…

```
#we can observe How Interest rate feature is Correlated with other independent features
```

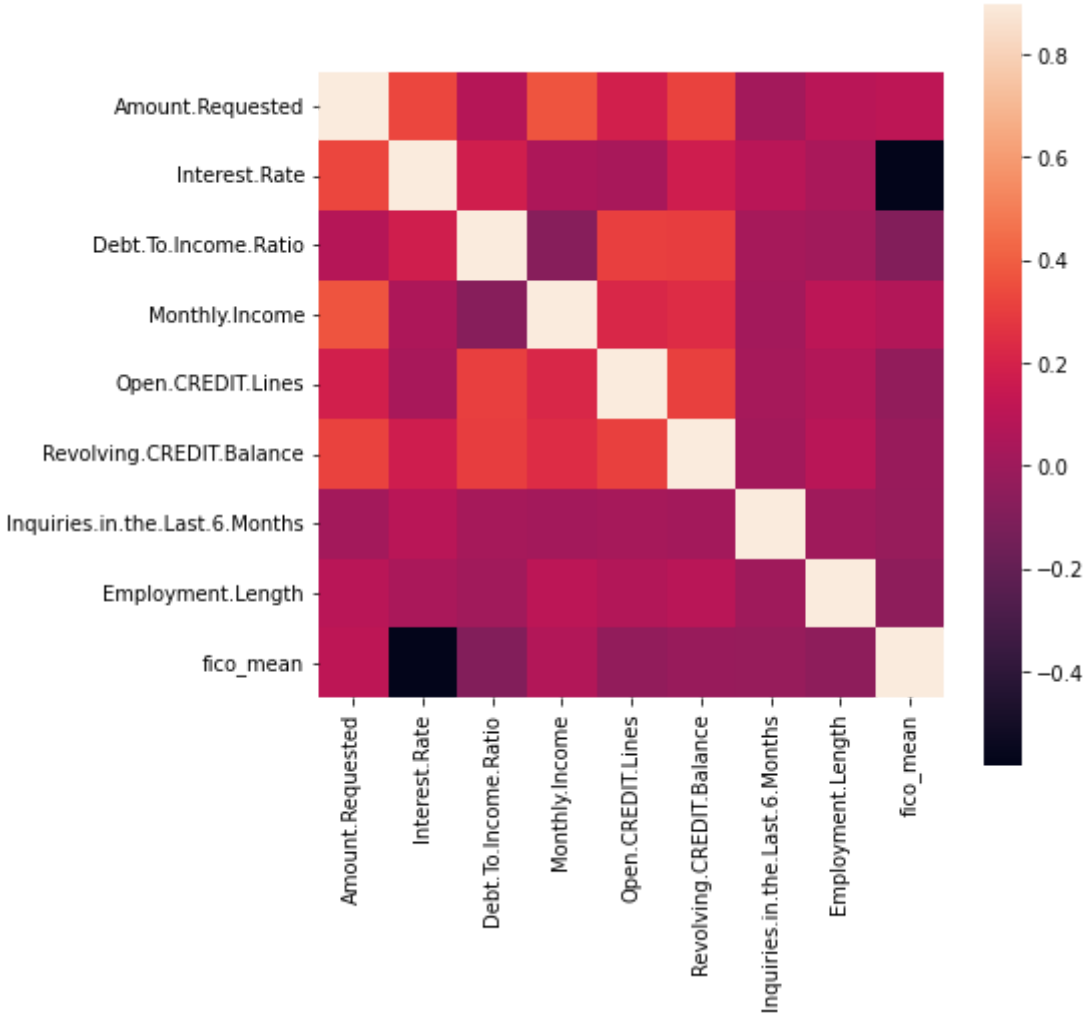
In [707…

```
# Insigths from Correlation : ,
  # Interest rate is more dependent on Amount Requested
  # Interest rate is Least dependent on Fico score( credit score)
  # Interest rate is moderatley dependent on montly income
```

In [708…

```
corrmat = Combined_data.corr(method='pearson')
plt.subplots(figsize=(7,7))
sns.heatmap(corrmat, vmax=0.9, square=True)
```

Out[708… <AxesSubplot:>



Visualization on How Categorical variables are spreaded ,checking whether categorical variables are dependent on other features or not

The below graph shows How monthly income is different for various Homeownerships and also we can - find the LOAN purpose

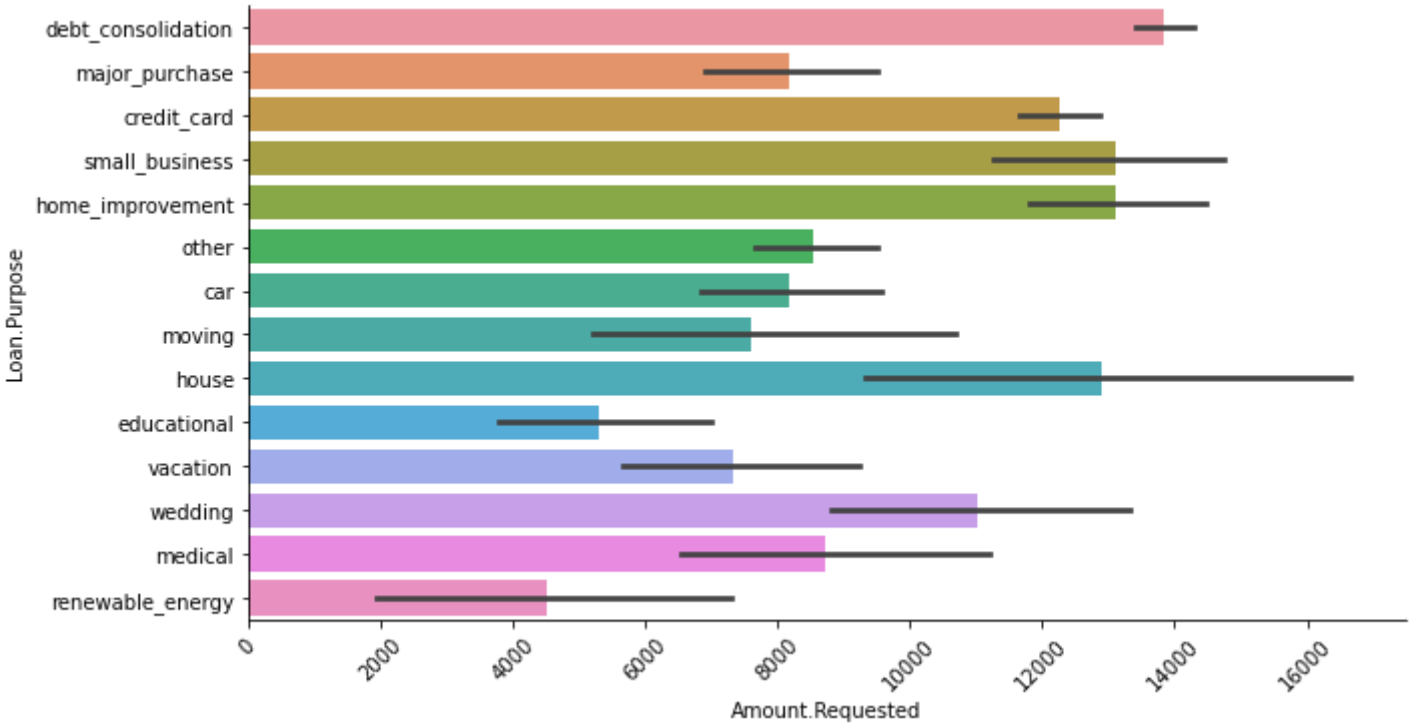
with respect to these features

In [709…

```
sns.catplot(data=Combined_data, x='Amount.Requested', y="Loan.Purpose", kind="bar",aspect=2,)
plt.xticks(rotation='45')
```

Out[709…

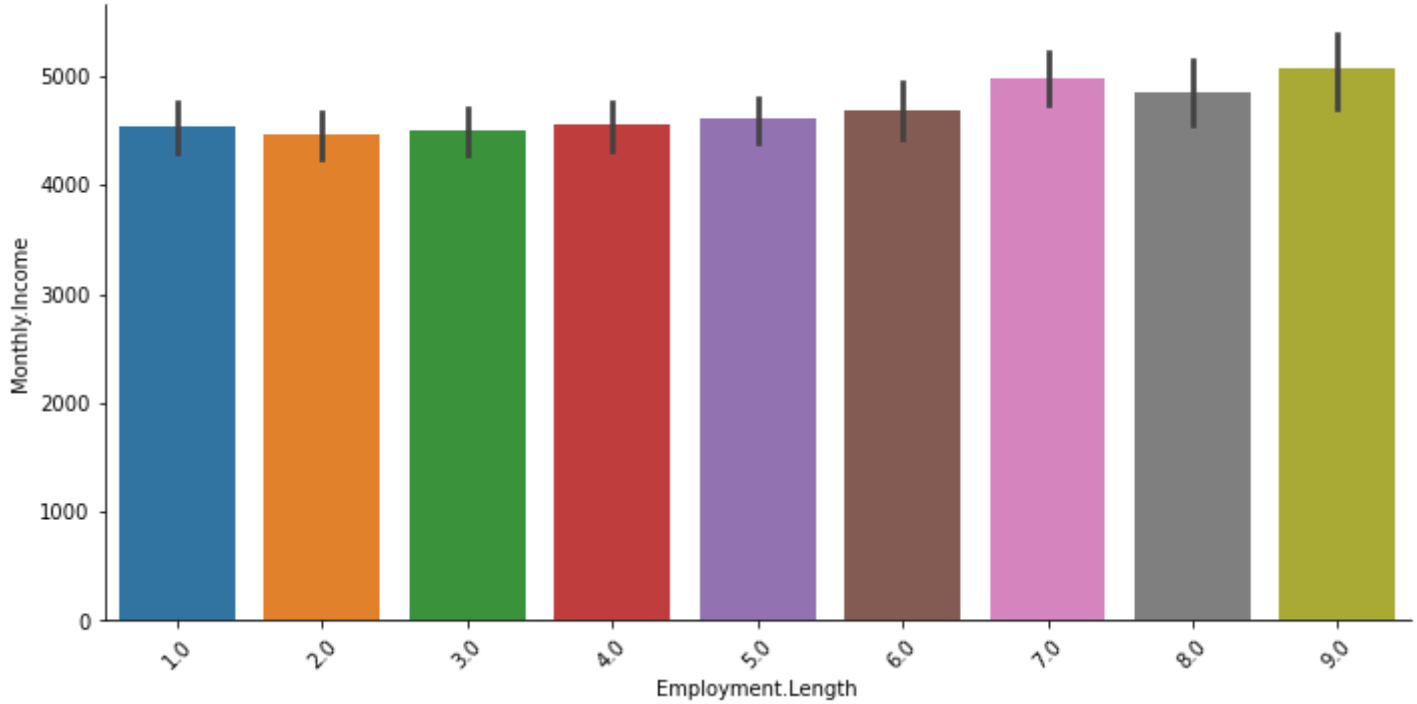
```
(array([    0.,  2000.,  4000.,  6000.,  8000., 10000., 12000., 14000.,
        16000., 18000.]),
 [Text(0.0, 0, '0'),
  Text(2000.0, 0, '2000'),
  Text(4000.0, 0, '4000'),
  Text(6000.0, 0, '6000'),
  Text(8000.0, 0, '8000'),
  Text(10000.0, 0, '10000'),
  Text(12000.0, 0, '12000'),
  Text(14000.0, 0, '14000'),
  Text(16000.0, 0, '16000'),
  Text(18000.0, 0, '18000')])
```



In [710…

```
sns.catplot(data=Combined_data, x='Employment.Length', y="Monthly.Income", kind="bar",aspect=2,)
plt.xticks(rotation='45')
```

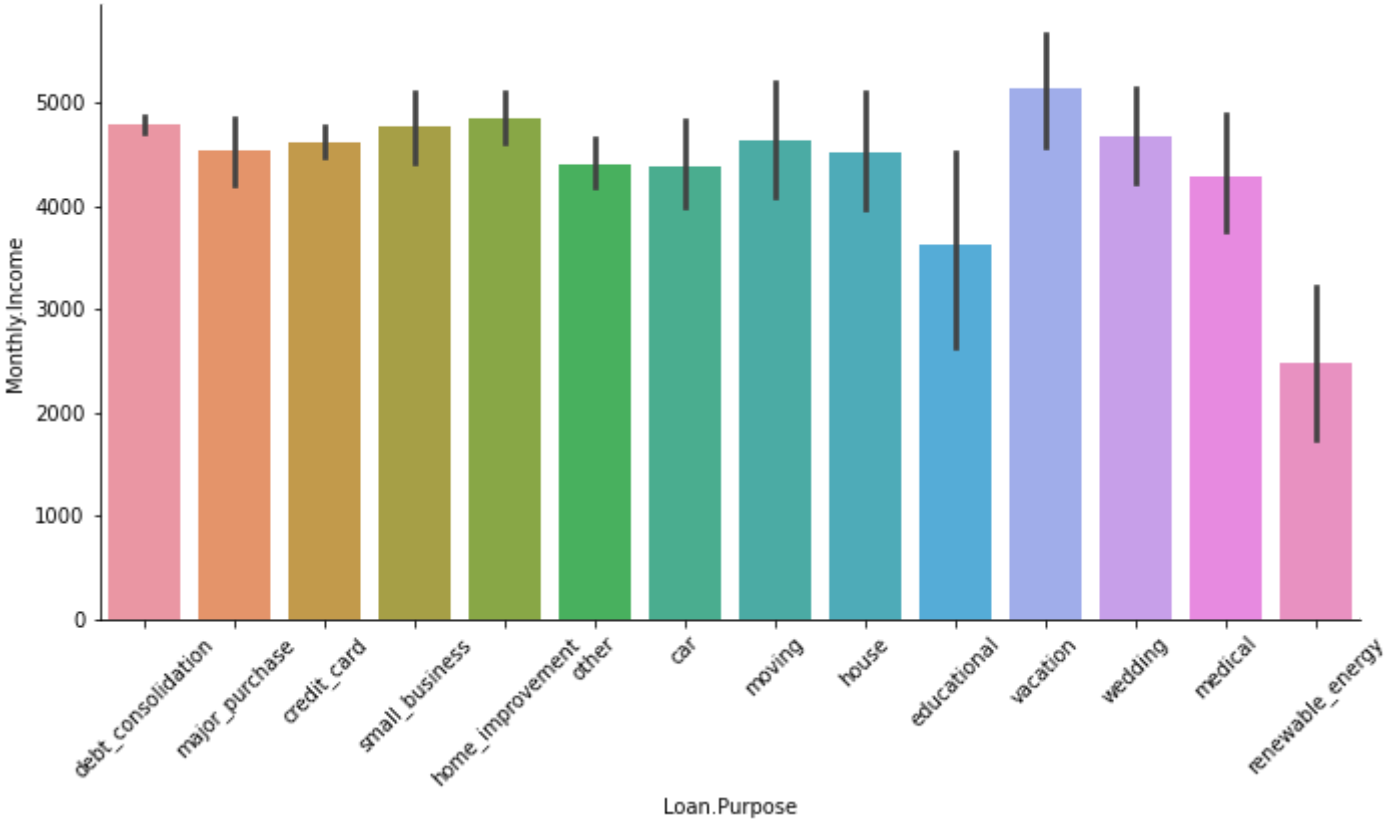
Out[710…
```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8]),
 [Text(0, 0, '1.0'),
  Text(1, 0, '2.0'),
  Text(2, 0, '3.0'),
  Text(3, 0, '4.0'),
  Text(4, 0, '5.0'),
  Text(5, 0, '6.0'),
  Text(6, 0, '7.0'),
  Text(7, 0, '8.0'),
  Text(8, 0, '9.0')])
```



In [711…
```
sns.catplot(data=Combined_data, x='Loan.Purpose', y="Monthly.Income", kind="bar",aspect=2,)
plt.xticks(rotation='45')
```

Out[711…
```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13]),
 [Text(0, 0, 'debt_consolidation'),
  Text(1, 0, 'major_purchase'),
  Text(2, 0, 'credit_card'),
  Text(3, 0, 'small_business'),
  Text(4, 0, 'home_improvement'),
  Text(5, 0, 'other'),
```

```
Text(6, 0, 'car'),
Text(7, 0, 'moving'),
Text(8, 0, 'house'),
Text(9, 0, 'educational'),
Text(10, 0, 'vacation'),
Text(11, 0, 'wedding'),
Text(12, 0, 'medical'),
Text(13, 0, 'renewable_energy')])
```



Insights from above visualizations:

1. Most of the amount requested for Housing

2. Least amount requested is for renewal energy

3. Most of the employee are having same income though they have difference in experience

4. People who requested Loan for vacation have highest income

The below graph shows how monthly income is different for various Homeownerships and LOAN purpose

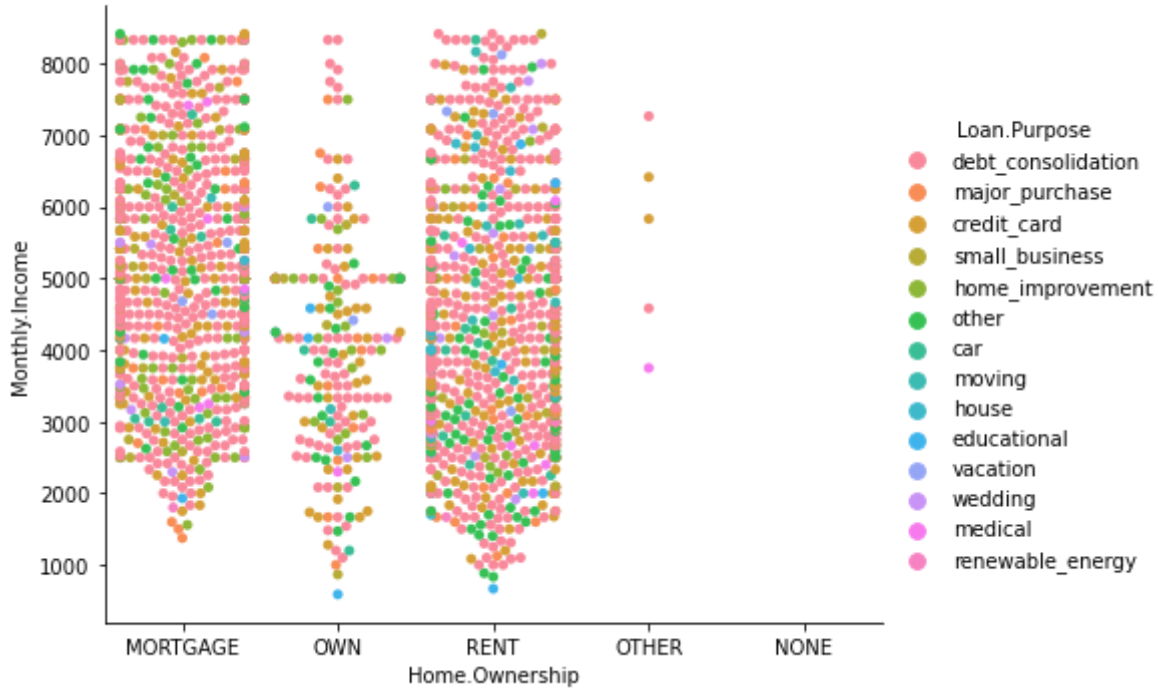
In [712…

```
sns.catplot(data=Combined_data, x='Home.Ownership', y="Monthly.Income", hue='Loan.Purpose', kind="swarm",aspect
```

```
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/categorical.py:1296: UserWarning: 59.9% of th
e points cannot be placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/categorical.py:1296: UserWarning: 12.0% of th
e points cannot be placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/categorical.py:1296: UserWarning: 55.4% of th
e points cannot be placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```

Out[712…

```
<seaborn.axisgrid.FacetGrid at 0x7fe1cb2d14c0>
```



In [713…
```
#Visualisation of above graph with respect to variable Loan Length
```

In [714…
```
Combined_data['Loan.Length'].value_counts()
```

Out[714…
```
36 months    1950
60 months     548
.               1
Name: Loan.Length, dtype: int64
```

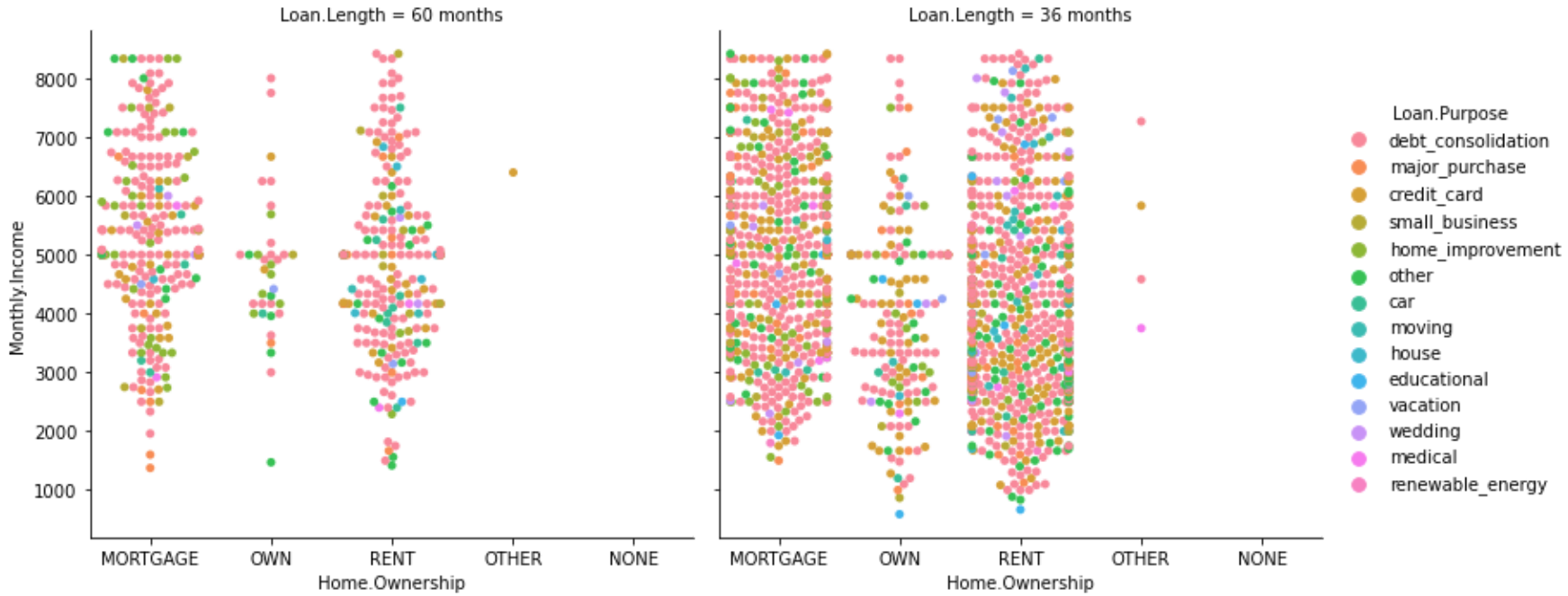
In [715…
```
Combined_data=Combined_data[Combined_data['Loan.Length'] != '.']
```

In [716…
```
sns.catplot(
    data=Combined_data, x='Home.Ownership', y="Monthly.Income", hue='Loan.Purpose',
    kind="swarm", col="Loan.Length",aspect=1.1
)
```

```
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/categorical.py:1296: UserWarning: 27.6% of th
e points cannot be placed; you may want to decrease the size of the markers or use stripplot.
```

```
  warnings.warn(msg, UserWarning)
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/categorical.py:1296: UserWarning: 11.5% of th
e points cannot be placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/categorical.py:1296: UserWarning: 54.7% of th
e points cannot be placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/categorical.py:1296: UserWarning: 9.7% of the
points cannot be placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/categorical.py:1296: UserWarning: 53.9% of th
e points cannot be placed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
<seaborn.axisgrid.FacetGrid at 0x7fe1cb36e220>
```

Out [716…



Performing Chisq Test on categorical variables

In [717…

```
import scipy.stats
from scipy.stats import chi2
```

In [718…
```
Count_table=pd.crosstab(Combined_data["Loan.Purpose"],Combined_data["Loan.Length"])
Count_table_2=pd.crosstab(Combined_data["Loan.Purpose"],Combined_data["Home.Ownership"])
```

In [719…
```
print(Count_table)
print('\n')
print(Count_table_2)
```

```
Loan.Length          36 months  60 months
Loan.Purpose
car                         33         17
credit_card                382         62
debt_consolidation         986        319
educational                 14          1
home_improvement           106         46
house                       14          6
major_purchase              83         18
medical                     26          4
moving                      24          5
other                      164         36
renewable_energy             4          0
small_business              62         25
vacation                    17          4
wedding                     34          5


Home.Ownership      MORTGAGE  NONE  OTHER  OWN  RENT
Loan.Purpose
car                       18     0      0    5    27
credit_card              189     0      2   38   214
debt_consolidation       620     0      2   87   597
educational                5     0      0    3     7
home_improvement         117     0      0   20    15
house                      5     0      0    2    13
major_purchase            43     0      0   14    44
medical                   11     0      1    1    17
moving                     4     0      0    1    24
other                     72     1      0   18   109
renewable_energy           1     0      0    0     3
small_business            40     0      0    6    41
vacation                   7     0      0    3    11
wedding                   14     0      0    2    23
```

In [ ]:

In [720…
```
def check_chisqtest(table,v1,v2):
    alpha=0.05
    chi2_stat, p_value, deg_freedom, expected = scipy.stats.chi2_contingency(Count_table_2)
    if p_value <= alpha:
        print(p_value)
        print('WE can REJECT the Null Hypothesis ,Stating that two variables' )
        print('{} , {}  are DEPENDENT to each other'.format(v1,v2))
    else:
        print(p_value)
        print('WE can Accept the Null Hypothesis ,Stating that two variables {} ,{} ')
        print('{} , {}  are DEPENDENT to each other'.format(v1,v2))
```

In [721…
```
# Checking dependency for Categorical Variables Loan.Length', 'Loan.Purpose' 'Home.Ownership'
```

In [722…
```
check_chisqtest(Count_table,'Loan.Length', 'Loan.Purpose')
```

```
6.021672834622822e-13
WE can REJECT the Null Hypothesis ,Stating that two variables
Loan.Length , Loan.Purpose  are DEPENDENT to each other
```

In [723…
```
check_chisqtest(Count_table_2,'Loan.Purpose' ,'Home.Ownership')
```

```
6.021672834622822e-13
WE can REJECT the Null Hypothesis ,Stating that two variables
Loan.Purpose , Home.Ownership  are DEPENDENT to each other
```

As there is dependency between these variables , and it is violating the,

assumption of Linear regression let drop 2 categorical variables

In [724…
```
Combined_data.columns
```

Out[724…
```
Index(['Amount.Requested', 'Interest.Rate', 'Loan.Length', 'Loan.Purpose',
       'Debt.To.Income.Ratio', 'State', 'Home.Ownership', 'Monthly.Income',
```

```
       'Open.CREDIT.Lines', 'Revolving.CREDIT.Balance',
       'Inquiries.in.the.Last.6.Months', 'Employment.Length', 'data',
       'fico_mean'],
      dtype='object')
```

In [725…
```
Combined_data.drop([ 'Home.Ownership'],axis=1,inplace=True)
```

```
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/pandas/core/frame.py:4906: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  return super().drop(
```

In [726…
```
# Categorical Variables
cat_cols=Combined_data.select_dtypes(['object']).columns
```

In [727…
```
cat_cols
```

Out[727…
```
Index(['Loan.Length', 'Loan.Purpose', 'State', 'data'], dtype='object')
```

In [728…
```
cat_cols=cat_cols[:-1]
```

In [729…
```
cat_cols
```

Out[729…
```
Index(['Loan.Length', 'Loan.Purpose', 'State'], dtype='object')
```

In [730…
```
# Creating dummy variables for categorical Variables

for col in cat_cols:
    dummy=pd.get_dummies(Combined_data[col],drop_first=True,prefix=col)
    Combined_data=pd.concat([Combined_data,dummy],axis=1)
    del Combined_data[col]
    print("Created Dummy Variables for column {}".format(str(col)))
```

```
Created Dummy Variables for column Loan.Length
Created Dummy Variables for column Loan.Purpose
Created Dummy Variables for column State
```

In [731…
```
Combined_data.shape
```

Out[731…
```
(2499, 70)
```

In [732…
```
Combined_data.isnull().sum()
```

Out[732…
```
Amount.Requested          5
Interest.Rate           300
Debt.To.Income.Ratio      1
Monthly.Income            3
Open.CREDIT.Lines         9
                       ...
State_VT                  0
State_WA                  0
State_WI                  0
State_WV                  0
State_WY                  0
Length: 70, dtype: int64
```

In [733…
```
# Replacing missing values of Input features with mean

for col in Combined_data.columns:
    if (col not in ['Interest.Rate','data'])& (Combined_data[col].isnull().sum()>0):
        Combined_data.loc[Combined_data[col].isnull(),col]=Combined_data.loc[Combined_data['data']=='train',col
```

In [734…
```
Combined_data_train=Combined_data[Combined_data['data']=='train']
del Combined_data_train['data']
Combined_data_test=Combined_data[Combined_data['data']=='test']
Combined_data_test.drop(['Interest.Rate','data'],axis=1,inplace=True)
```

In [735…
```
from sklearn.model_selection import train_test_split
```

In [736…
```
Combined_data_train1,Combined_data_train2=train_test_split(Combined_data_train,test_size=0.2,random_state=2)
```

In [737…
```
# Notice that only train data is used for imputing missing values in both train and test

x_train_data1=Combined_data_train1.drop('Interest.Rate',axis=1)
y_train_data1=Combined_data_train1['Interest.Rate']
```

In [738…
```
from sklearn.linear_model import LinearRegression
```

In [739…
```
linear_model=LinearRegression()
```

In [740…
```
linear_model.fit(x_train_data1,y_train_data1)
```

Out[740…
```
LinearRegression()
```

In [741…
```
x_train_data1.shape
```

Out[741…
```
(1759, 68)
```

In [742…
```
linear_model.intercept_
```

Out[742…
```
89.75662738873308
```

In [743…
```
list(zip(x_train_data1.columns,linear_model.coef_))[:7]
```

Out[743…
```
[('Amount.Requested', 0.00013855621541699963),
 ('Debt.To.Income.Ratio', 0.048331956864370584),
 ('Monthly.Income', -8.512466722593288e-05),
 ('Open.CREDIT.Lines', -0.10809315898188815),
 ('Revolving.CREDIT.Balance', 2.821700204447275e-05),
 ('Inquiries.in.the.Last.6.Months', 0.6616592886017565),
 ('Employment.Length', -0.01403722860060097)]
```

In [744…
```
linear_model.score(x_train_data1,y_train_data1)
```

Out[744…
```
0.6360059950041335
```

In [745…
```
x_train2=Combined_data_train2.drop('Interest.Rate',axis=1)
```

In [746…
```
predicted_ir=linear_model.predict(x_train2)
```

In [747…
```
from sklearn.metrics import mean_absolute_error ,r2_score
```

In [748…
```
mean_absolute_error(Combined_data_train2['Interest.Rate'],predicted_ir)
```

Out[748… 1.9673094751598321

In [749…
```
r2_score(Combined_data_train2['Interest.Rate'],predicted_ir)
```

Out[749… 0.6314940370014244

In [ ]:

In [ ]:

Checking Assumptions of Linear Regression:

1. Homoscedasticity (Error should be constant)

2. Normality of errors (errors should be normally distributed)

3. There should be no multicollinearity between independent variables

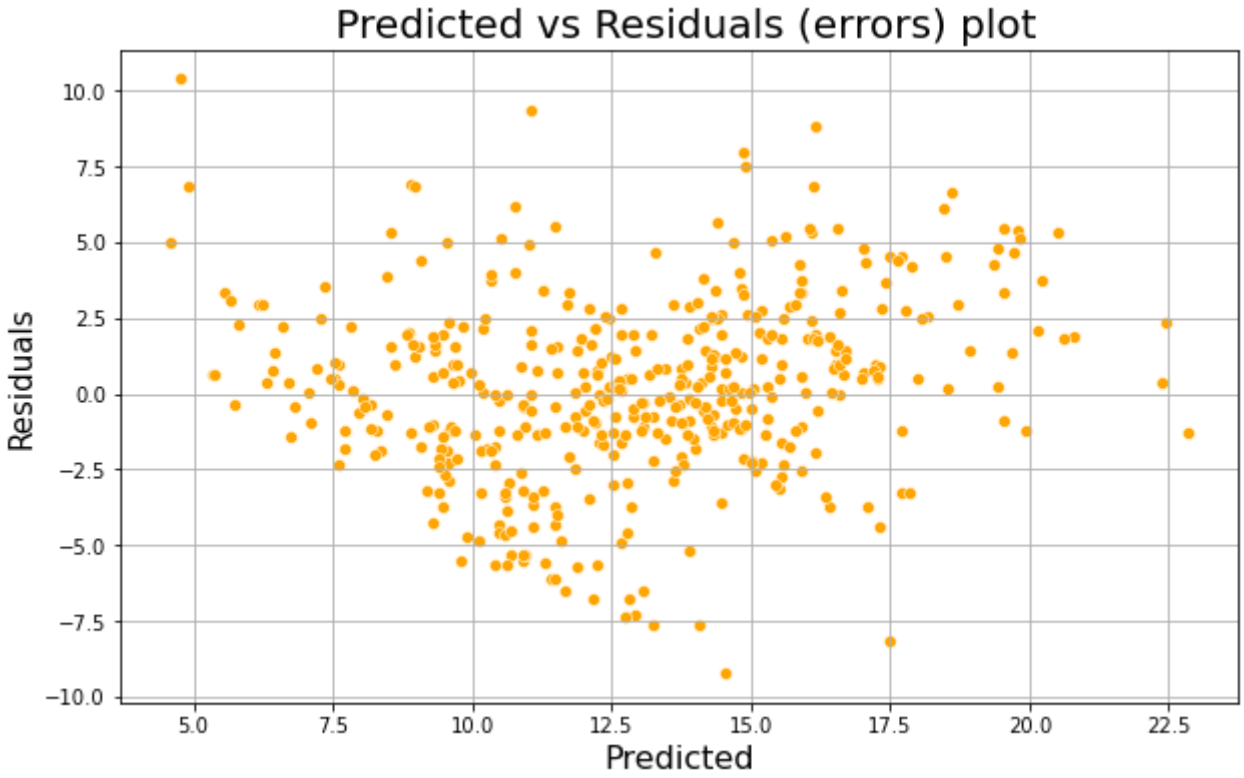
In [ ]:

In [ ]:

1. Homoscedasticity (Error should be constant)

In [754…

```
plt.figure(figsize=(10,6))
p=sns.scatterplot(x=predicted_ir,y=residuals,color='orange')
xmin=min(predicted_ir)
xmax = max(predicted_ir)
plt.xlabel("Predicted",fontsize=16)
plt.ylabel("Residuals",fontsize=15)
plt.title(" Predicted vs Residuals (errors) plot",fontsize=20)
plt.grid(True)
plt.show()
```



In [ ]:

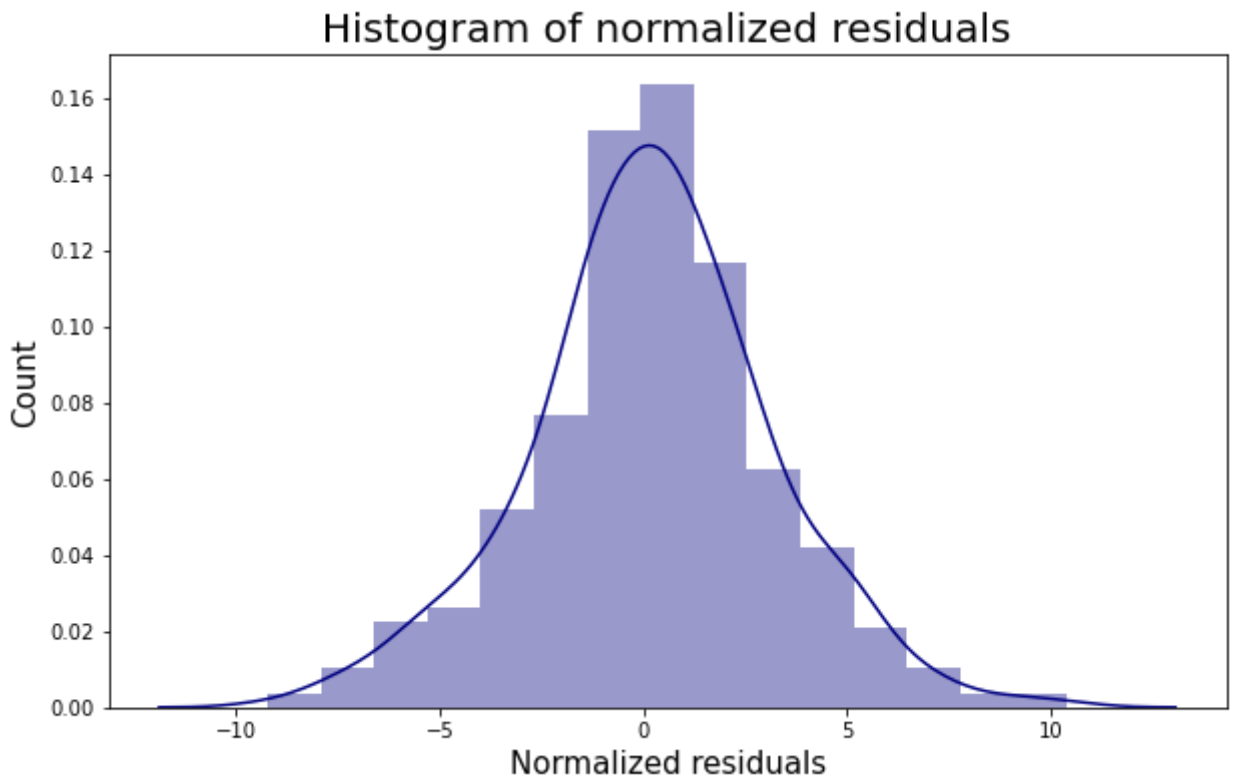
2. Normality of errors (errors should be normally distributed)

In [ ]:

In [752…

```
plt.figure(figsize=(10,6))
sns.distplot(residuals,bins=15,color='navy')
plt.ylabel('Count',fontsize=15)
plt.xlabel('Normalized residuals',fontsize=15)
plt.title("Histogram of normalized residuals",fontsize=20)
plt.show()
```

```
/Users/jaswanth/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distpl
ot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `dis
plot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```


Histogram of normalized residuals
Count
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
−10
−5
0
5
10
Normalized residuals