```
'''
This Model will help us to find out wheather we sud provide loan or not.

'''
```

```
     '\nThis Model will help us to find out wheather we sud provide loan or not.\n\n'
```

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

```
# Data analysis Project.
# Step1- Read the Loan Prediction Dataset.csv
# https://github.com/sameerCoder/DATA_ANALYST_DATASETS/blob/main/Loan%20Prediction%20Dataset.
```

```
df = pd.read_csv('https://raw.githubusercontent.com/sameerCoder/DATA_ANALYST_DATASETS/main/Lo
```

```
'''
Reading the file using python code.
Data Cleaning
Data Manupulation
Data Visulization
'''
```

```
'\nReading the file using python code.\nData Cleaning\nData Manupulation\nData Visuliza
tion\n'
```

```
# Write code to print the first 10 rows of data.
df.head(10)
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Co
0	LP001002	Male	No	0	Graduate	No	5849	

```
# Write code to print count, mean, std, 25% , 50%, 75% and max
df.describe()
```

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	614.000000	614.000000	592.000000	600.00000	564.000000
mean	5403.459283	1621.245798	146.412162	342.00000	0.842199
std	6109.041673	2926.248369	85.587325	65.12041	0.364878
min	150.000000	0.000000	9.000000	12.00000	0.000000
25%	2877.500000	0.000000	100.000000	360.00000	1.000000
50%	3812.500000	1188.500000	128.000000	360.00000	1.000000
75%	5795.000000	2297.250000	168.000000	360.00000	1.000000
max	81000.000000	41667.000000	700.000000	480.00000	1.000000

```
# Print all the columns name with there datatype.
df.columns
```

```
Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education',
       'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
       'Loan_Amount_Term', 'Credit_History', 'Property_Area', 'Loan_Status',
       'NEWCOL1'],
      dtype='object')
```

```
# PreProcessing The Data
```

```
# print all columns and number of rows having nan value.
df.isna().sum()
```

```
Loan_ID              0
Gender              13
Married              3
Dependents          15
Education            0
Self_Employed       32
ApplicantIncome      0
CoapplicantIncome    0
LoanAmount          22
Loan_Amount_Term    14
Credit_History      50
Property_Area        0
Loan_Status          0
```

```
NEWCOL1              0
dtype: int64
```

```
# create a new column name - NEWCOL1 having data copy of first column of dataset file.
df['NEWCOL1'] = df[df.columns[0]]
```

```
# IN NEWCOL1 replace the row number 10 to 50 with nan value.
df['NEWCOL1'][10:51] = np.nan
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: 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_
```

```
# Fill the nan value of numerical column with there mean.
df.fillna(df.mean(), inplace=True)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: FutureWarning: Dropping
```

```
# Fill the nan value of non-numerical column with string "DATA MISSING" OR MODE.
df.fillna('DATA MISSING', inplace=True)
```

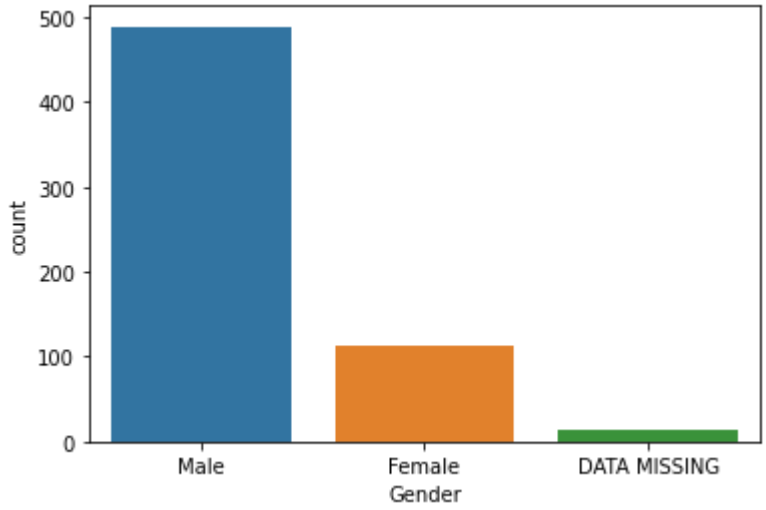
```
# Check all nan values has been removed or not
df.isna().sum()
```

```
Loan_ID              0
Gender               0
Married              0
Dependents           0
Education            0
Self_Employed        0
ApplicantIncome      0
CoapplicantIncome    0
LoanAmount           0
Loan_Amount_Term     0
Credit_History       0
Property_Area        0
Loan_Status          0
NEWCOL1              0
dtype: int64
```

```
# DATA Visulization
```

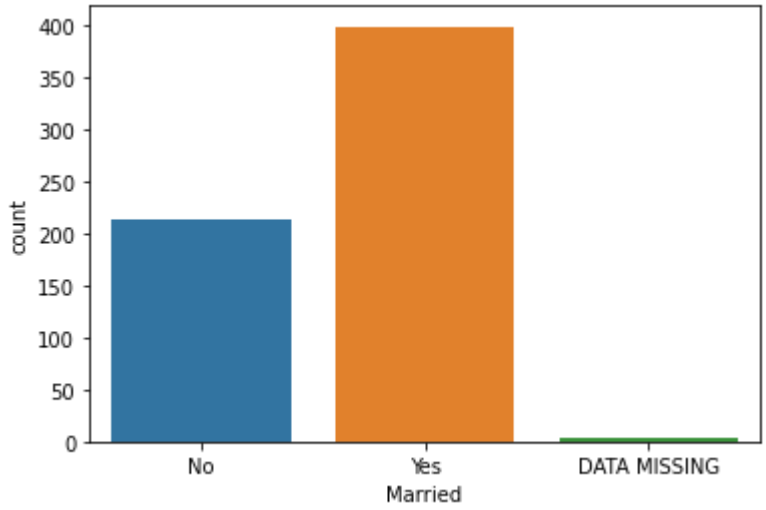
```
# plot countplot of Gender column
sns.countplot(df['Gender'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f7e43a8c6d0>
```




```
# plot - do countplot of Married column
sns.countplot(df['Married'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f7e43a1f610>
```

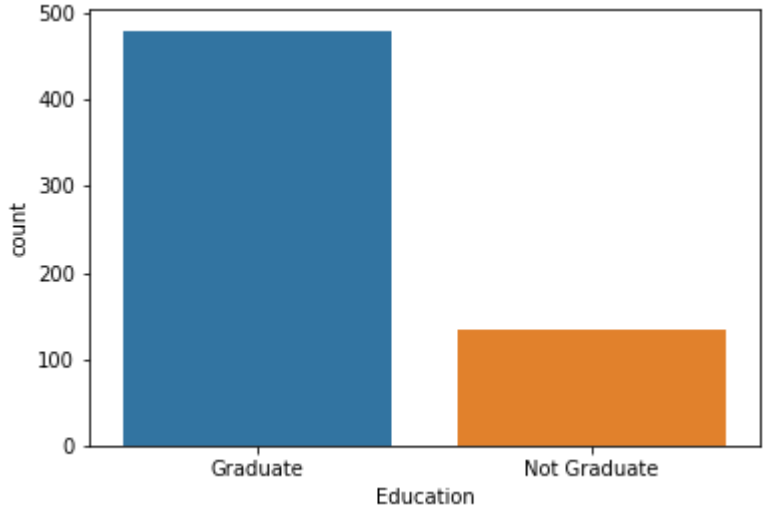



```
# plot - do countplot of Dependent column
sns.countplot(df['Dependents'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f7e43c36e50>
```


```
# plot - do countplot of Education column
sns.countplot(df['Education'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f7e43497f90>
```

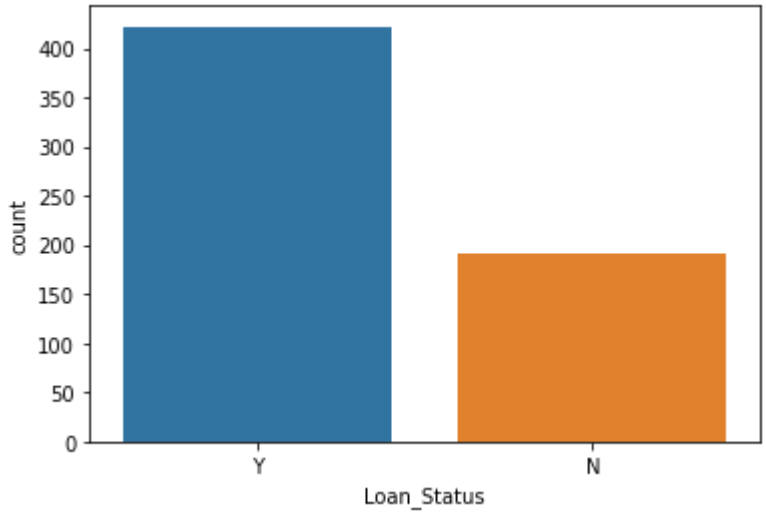



```
# plot - do countplot of selfemployed column
sns.countplot(df['Self_Employed'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f7e43439f10>
```

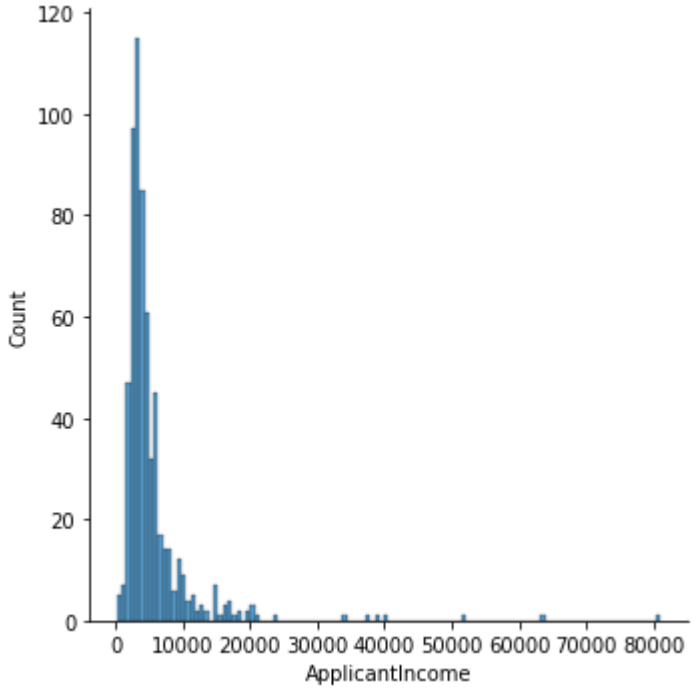

```
# plot - do countplot of Loanstatus column
sns.countplot(df['Loan_Status'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f7e433a0890>
```




```
# plot - do distplot of ApplicantIncome column
sns.displot(df['ApplicantIncome'])
```

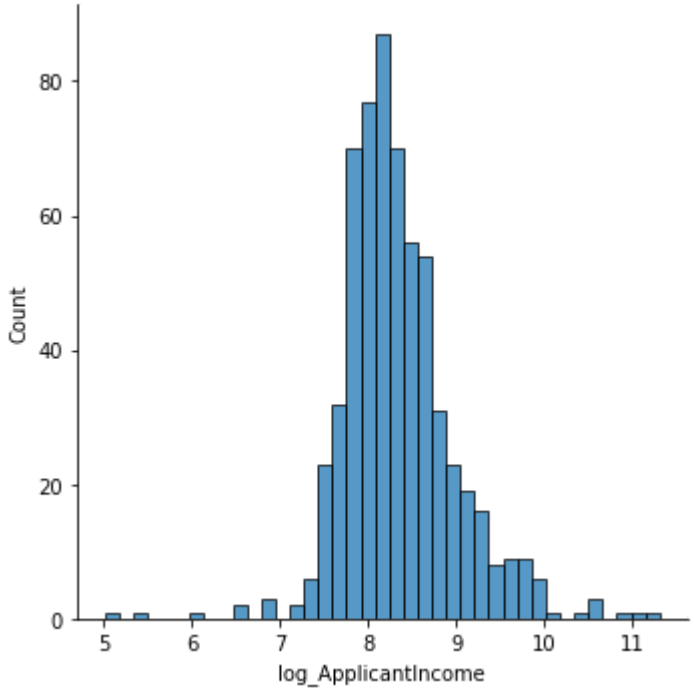
```
<seaborn.axisgrid.FacetGrid at 0x7f7e435523d0>
```



```
# ApplicantIncome column is not appearing good so
# we will do log normalization
df['log_ApplicantIncome'] = np.log(df['ApplicantIncome'])
# write code to do the log of ApplicantIncome column.


# now again do distplot of ApplicatnIncome column.
sns.displot(df['log_ApplicantIncome'])
# write the commment now how the plot appear after log
```

```
<seaborn.axisgrid.FacetGrid at 0x7f7e43378110>
```

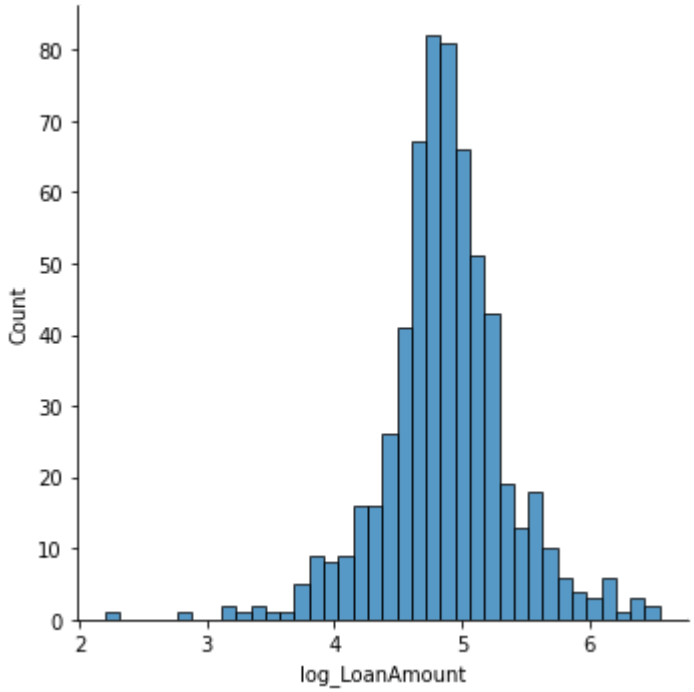



```
# Distplot loanAmount
sns.displot(df['LoanAmount'])
# log the loanAmount
# Distplot loanAmount
```

```
<seaborn.axisgrid.FacetGrid at 0x7f7e4357f2d0>
```


```
df['log_LoanAmount'] = np.log(df['LoanAmount'])
sns.displot(df['log_LoanAmount'])
```

```
<seaborn.axisgrid.FacetGrid at 0x7f7e40995c10>
```

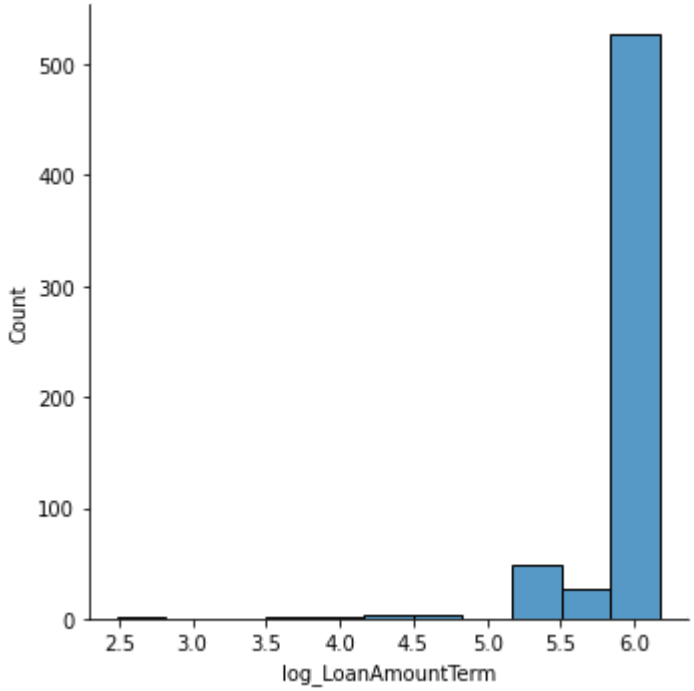



```
# Distplot Loan_Amount_Term
sns.displot(df['Loan_Amount_Term'])
# log the Loan_Amount_Term
# Distplot Loan_Amount_Term
```

```
<seaborn.axisgrid.FacetGrid at 0x7f7e43440790>
```


```
df['log_LoanAmountTerm'] = np.log(df['Loan_Amount_Term'])
sns.displot(df['log_LoanAmountTerm'])
```

```
<seaborn.axisgrid.FacetGrid at 0x7f7e4351d450>
```




```
# Create new Total_Income column= ApplicantIncome , CoapplicantIncome
df['TotalIncome'] = df['ApplicantIncome'] + df['CoapplicantIncome']


# Create ApplicantIncomeLog column = value log of ApplicantIncome
df['ApplicantIncomeLog'] = np.log(df['ApplicantIncome'])


# Create CoaplicantIncomeLog column = value log of CoaplicantIncome
df['CoaplicantIncomeLog'] = np.log(df['CoapplicantIncome'])
```

```
/usr/local/lib/python3.7/dist-packages/pandas/core/arraylike.py:364: RuntimeWarning: div
  result = getattr(ufunc, method)(*inputs, **kwargs)
```

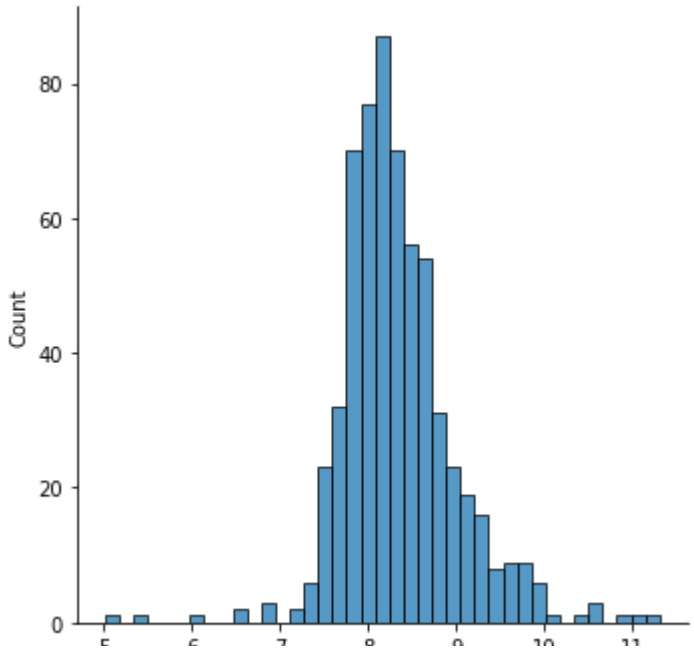
```
# plot displot of ApplicationIncomeLog
# plot displot of CoapplicantIncomeLog
sns.displot(df['ApplicantIncomeLog'])
sns.distplot(df['CoaplicantIncomeLog'])
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `di
  warnings.warn(msg, FutureWarning)
/usr/local/lib/python3.7/dist-packages/numpy/lib/function_base.py:4009: RuntimeWarning:
  diff_b_a = subtract(b, a)
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-54-ba4407444787> in <module>()
      2 # plot displot of CoapplicantIncomeLog
      3 sns.displot(df['ApplicantIncomeLog'])
----> 4 sns.distplot(df['CoaplicantIncomeLog'])
      5 plt.show()

 1 frames
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py in
_freedman_diaconis_bins(a)
   2463         return int(np.sqrt(a.size))
   2464     else:
-> 2465         return int(np.ceil((a.max() - a.min()) / h))
   2466
   2467

ValueError: cannot convert float NaN to integer
```

SEARCH STACK OVERFLOW



✓ 0s completed at 12:03 PM