In [1]:

```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
sns.set_style('whitegrid')
%matplotlib inline
import numpy as np
```

In [2]:

```
# Data analysis Project.
# Step1- Read the Loan Prediction Dataset.csv
# https://github.com/sameerCoder/DATA_ANALYST_DATASETS/blob/main/Loan%20Prediction%20Da
taset.csv
url="https://raw.githubusercontent.com/sameerCoder/DATA_ANALYST_DATASETS/main/Loan%20Pr
ediction%20Dataset.csv"
df=pd.read_csv(url)
df
```

Out[2]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	(
0	LP001002	Male	No	0	Graduate	No	5849	
1	LP001003	Male	Yes	1	Graduate	No	4583	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	
4	LP001008	Male	No	0	Graduate	No	6000	
609	LP002978	Female	No	0	Graduate	No	2900	
610	LP002979	Male	Yes	3+	Graduate	No	4106	
611	LP002983	Male	Yes	1	Graduate	No	8072	
612	LP002984	Male	Yes	2	Graduate	No	7583	
613	LP002990	Female	No	0	Graduate	Yes	4583	

614 rows × 13 columns

In [3]:

```
# Write code to print the first 10 rows of data.
df1=df.head(10)
df1
```

Out[3]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coa
0	LP001002	Male	No	0	Graduate	No	5849	
1	LP001003	Male	Yes	1	Graduate	No	4583	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	
4	LP001008	Male	No	0	Graduate	No	6000	
5	LP001011	Male	Yes	2	Graduate	Yes	5417	
6	LP001013	Male	Yes	0	Not Graduate	No	2333	
7	LP001014	Male	Yes	3+	Graduate	No	3036	
8	LP001018	Male	Yes	2	Graduate	No	4006	
9	LP001020	Male	Yes	1	Graduate	No	12841	

In [4]:

Write code to print count, mean, std, 25% , 50%, 75% and max df2=df1.describe() df2

Out[4]:

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	10.000000	10.000000	9.000000	10.0	10.000000
mean	4964.800000	2457.600000	165.777778	360.0	0.900000
std	3079.278047	3270.009147	88.729614	0.0	0.316228
min	2333.000000	0.000000	66.000000	360.0	0.000000
25%	3009.000000	377.000000	120.000000	360.0	1.000000
50%	4294.500000	1521.000000	141.000000	360.0	1.000000
75%	5741.000000	2467.500000	168.000000	360.0	1.000000
max	12841.000000	10968.000000	349.000000	360.0	1.000000
4					•

In [5]:

Print all the columns name with there datatype
df1.dtypes

Out[5]:

Loan_ID object Gender object Married object Dependents object Education object Self_Employed object ApplicantIncome int64 CoapplicantIncome float64 LoanAmount float64 float64 Loan_Amount_Term Credit_History float64 object Property_Area Loan_Status object dtype: object

In [6]:

print all columns and number of rows having nan value.
df3 = df[df.isna().any(axis=1)]
df3

Out[6]:

		Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	(
	0	LP001002	Male	No	0	Graduate	No	5849	
	11	LP001027	Male	Yes	2	Graduate	NaN	2500	
	16	LP001034	Male	No	1	Not Graduate	No	3596	
	19	LP001041	Male	Yes	0	Graduate	NaN	2600	
	23	LP001050	NaN	Yes	2	Not Graduate	No	3365	
5	92	LP002933	NaN	No	3+	Graduate	Yes	9357	
5	97	LP002943	Male	No	NaN	Graduate	No	2987	
6	00	LP002949	Female	No	3+	Graduate	NaN	416	
6	01	LP002950	Male	Yes	0	Not Graduate	NaN	2894	
6	05	LP002960	Male	Yes	0	Not Graduate	No	2400	

134 rows × 13 columns

In [7]:

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

Out[7]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	(
0	LP001002	Male	No	0	Graduate	No	5849	
1	LP001003	Male	Yes	1	Graduate	No	4583	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	
4	LP001008	Male	No	0	Graduate	No	6000	
609	LP002978	Female	No	0	Graduate	No	2900	
610	LP002979	Male	Yes	3+	Graduate	No	4106	
611	LP002983	Male	Yes	1	Graduate	No	8072	
612	LP002984	Male	Yes	2	Graduate	No	7583	
613	LP002990	Female	No	0	Graduate	Yes	4583	
	ows × 14 c	olumns						
4								

In [8]:

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

E:\SOFTWARES\ANACONDA\lib\site-packages\ipykernel_launcher.py:2: SettingWi
thCopyWarning:

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#returning-a-view-versus-a-copy

Out[8]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	(
0	LP001002	Male	No	0	Graduate	No	5849	
1	LP001003	Male	Yes	1	Graduate	No	4583	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	
4	LP001008	Male	No	0	Graduate	No	6000	
609	LP002978	Female	No	0	Graduate	No	2900	
610	LP002979	Male	Yes	3+	Graduate	No	4106	
611	LP002983	Male	Yes	1	Graduate	No	8072	
612	LP002984	Male	Yes	2	Graduate	No	7583	
613	LP002990	Female	No	0	Graduate	Yes	4583	

614 rows × 14 columns

```
In [9]:
```

```
df['NEWCOL1'].head(20)
Out[9]:
```

0 LP001002 1 LP001003 2 LP001005 3 LP001006 4 LP001008 5 LP001011 6 LP001013 7 LP001014 8 LP001018 9 LP001020 10 NaN 11 NaN 12 NaN NaN 13 14 NaN 15 NaN 16 NaN 17 NaN 18 NaN 19 NaN

Name: NEWCOL1, dtype: object

In [10]:

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

Out[10]:

Loan_ID 0 13 Gender Married 3 15 Dependents Education 0 32 Self_Employed ApplicantIncome 0 0 CoapplicantIncome 22 LoanAmount Loan_Amount_Term 14 50 Credit_History Property_Area 0 Loan_Status 0 NEWCOL1 41 dtype: int64

In [11]:

```
df1=df
```

In [12]:

```
# Fill the nan value of numerical column with there mean.
df['ApplicantIncome'].fillna(df['ApplicantIncome'].mean(),inplace=True)
df['CoapplicantIncome'].fillna(df['CoapplicantIncome'].mean(),inplace=True)
df['LoanAmount'].fillna(df['LoanAmount'].mean(),inplace=True)
df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mean(),inplace=True)
df['Credit_History'].fillna(df['Credit_History'].mean(),inplace=True)
df
```

Out[12]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	(
0	LP001002	Male	No	0	Graduate	No	5849	
1	LP001003	Male	Yes	1	Graduate	No	4583	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	
4	LP001008	Male	No	0	Graduate	No	6000	
609	LP002978	Female	No	0	Graduate	No	2900	
610	LP002979	Male	Yes	3+	Graduate	No	4106	
611	LP002983	Male	Yes	1	Graduate	No	8072	
612	LP002984	Male	Yes	2	Graduate	No	7583	
613	LP002990	Female	No	0	Graduate	Yes	4583	

614 rows × 14 columns

4

In [13]:

```
# Fill the nan value of non-numerical column with string "DATA MISSING" OR MODE.
df["Gender"].fillna("Data missing", inplace = True)
df["Married"].fillna("Data missing", inplace = True)
df["Dependents"].fillna("Data missing", inplace = True)
df["Self_Employed"].fillna("Data missing", inplace = True)
df["NEWCOL1"].fillna("Data missing", inplace = True)
df
```

Out[13]:

_oan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	(
P001002	Male	No	0	Graduate	No	5849	
P001003	Male	Yes	1	Graduate	No	4583	
P001005	Male	Yes	0	Graduate	Yes	3000	
P001006	Male	Yes	0	Not Graduate	No	2583	
P001008	Male	No	0	Graduate	No	6000	
P002978	Female	No	0	Graduate	No	2900	
P002979	Male	Yes	3+	Graduate	No	4106	
P002983	Male	Yes	1	Graduate	No	8072	
P002984	Male	Yes	2	Graduate	No	7583	
P002990	Female	No	0	Graduate	Yes	4583	
	2001002 2001003 2001005 2001006 2001008 2002978 2002979 2002983 2002984	2001002 Male 2001003 Male 2001005 Male 2001006 Male 2001008 Male 2002978 Female 2002979 Male 2002983 Male 2002984 Male	2001002 Male No 2001003 Male Yes 2001005 Male Yes 2001006 Male Yes 2001008 Male No 2002978 Female No 2002979 Male Yes 2002983 Male Yes 2002984 Male Yes	2001002 Male No 0 2001003 Male Yes 1 2001005 Male Yes 0 2001006 Male Yes 0 2001008 Male No 0 2002978 Female No 0 2002979 Male Yes 3+ 2002983 Male Yes 1 2002984 Male Yes 2	2001002 Male No 0 Graduate 2001003 Male Yes 1 Graduate 2001005 Male Yes 0 Graduate 2001006 Male Yes 0 Not Graduate 2001008 Male No 0 Graduate 2002978 Female No 0 Graduate 2002979 Male Yes 3+ Graduate 2002983 Male Yes 1 Graduate 2002984 Male Yes 2 Graduate	2001002 Male No 0 Graduate No 2001003 Male Yes 1 Graduate No 2001005 Male Yes 0 Graduate Yes 2001006 Male Yes 0 Not Graduate No 2001008 Male No 0 Graduate No 2002978 Female No 0 Graduate No 2002979 Male Yes 3+ Graduate No 2002983 Male Yes 1 Graduate No 2002984 Male Yes 2 Graduate No	0001002 Male No 0 Graduate No 5849 001003 Male Yes 1 Graduate No 4583 001005 Male Yes 0 Graduate Yes 3000 001006 Male Yes 0 Not Graduate No 2583 001008 Male No 0 Graduate No 6000 0002978 Female No 0 Graduate No 2900 0002979 Male Yes 3+ Graduate No 4106 0002983 Male Yes 1 Graduate No 7583 0002984 Male Yes 2 Graduate No 7583

614 rows × 14 columns

→

In [14]:

```
# Check all nan values has been removed or not
df1.isnull().sum()
```

Out[14]:

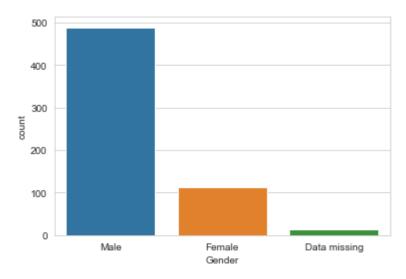
Loan_ID	0
Gender	0
Married	0
Dependents	0
Education	0
Self_Employed	0
ApplicantIncome	0
CoapplicantIncome	0
LoanAmount	0
Loan_Amount_Term	0
Credit_History	0
Property_Area	0
Loan_Status	0
NEWCOL1	0
dtype: int64	

In [15]:

```
# plot countplot of Gender column
sns.countplot('Gender',data=df)
```

Out[15]:

<matplotlib.axes._subplots.AxesSubplot at 0x20c1a9e75c8>

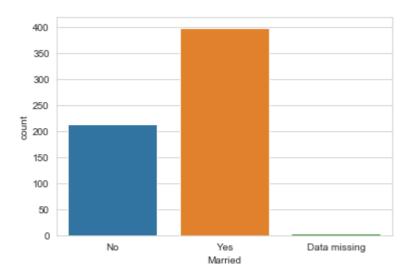


In [16]:

```
# plot - do countplot of Married column
# plot - do countplot of Dependent column
# plot - do countplot of Education column
# plot - do countplot of selfemployed column
# plot - do countplot of Loanstatus column
# plot - do distplot of ApplicantIncome column
sns.countplot('Married',data=df)
```

Out[16]:

<matplotlib.axes._subplots.AxesSubplot at 0x20c1b187f88>

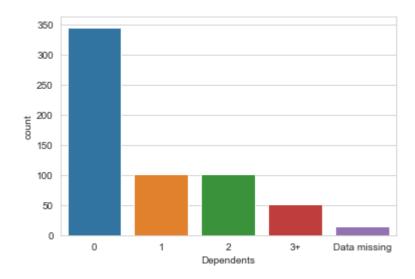


In [17]:

sns.countplot('Dependents',data=df)

Out[17]:

<matplotlib.axes._subplots.AxesSubplot at 0x20c1b1efa48>

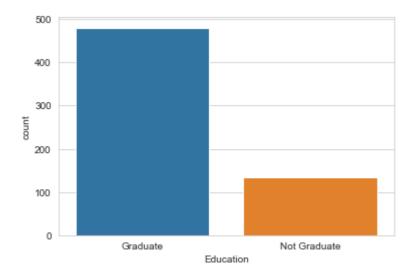


In [18]:

sns.countplot('Education',data=df)

Out[18]:

<matplotlib.axes._subplots.AxesSubplot at 0x20c1b2627c8>

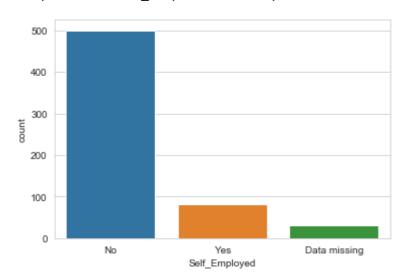


In [19]:

sns.countplot('Self_Employed',data=df)

Out[19]:

<matplotlib.axes._subplots.AxesSubplot at 0x20c1b2b9b08>

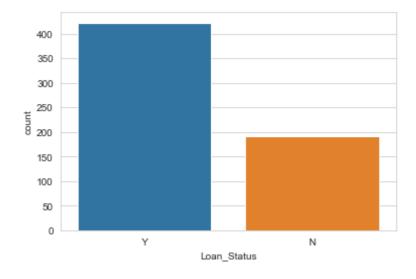


In [20]:

sns.countplot('Loan_Status',data=df)

Out[20]:

<matplotlib.axes._subplots.AxesSubplot at 0x20c1b315248>

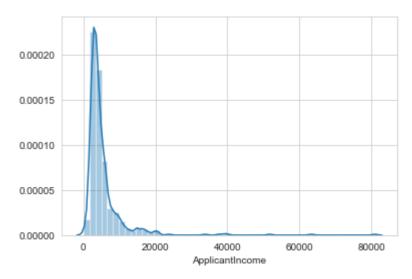


In [21]:

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

Out[21]:

<matplotlib.axes._subplots.AxesSubplot at 0x20c1b375448>



In [22]:

```
# ApplicantIncome column is not appearing good so
# we will do log normalization
# write code to do the log of ApplicantIncome column.

df['lognorm_ApplicantIncome'] = np.log(df['ApplicantIncome'])
df
```

Out[22]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	(
0	LP001002	Male	No	0	Graduate	No	5849	
1	LP001003	Male	Yes	1	Graduate	No	4583	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	
4	LP001008	Male	No	0	Graduate	No	6000	
609	LP002978	Female	No	0	Graduate	No	2900	
610	LP002979	Male	Yes	3+	Graduate	No	4106	
611	LP002983	Male	Yes	1	Graduate	No	8072	
612	LP002984	Male	Yes	2	Graduate	No	7583	
613	LP002990	Female	No	0	Graduate	Yes	4583	

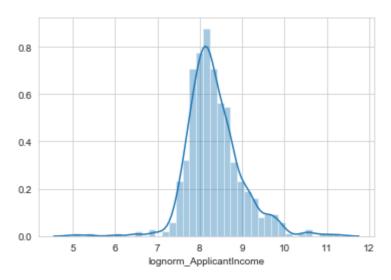
614 rows × 15 columns

In [23]:

```
# now again do distplot of ApplicatnIncome column.
sns.distplot(df['lognorm_ApplicantIncome'])
#By normalising the scale has been changed
```

Out[23]:

<matplotlib.axes._subplots.AxesSubplot at 0x20c1b3b0148>

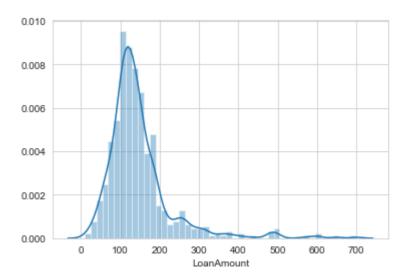


In [24]:

```
# Distplot LoanAmount
sns.distplot(df['LoanAmount'])
```

Out[24]:

<matplotlib.axes._subplots.AxesSubplot at 0x20c1b4c5788>



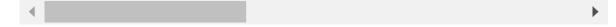
In [25]:

```
# log the loanAmount
df['lognorm_LoanAmount'] = np.log(df['LoanAmount'])
df
```

Out[25]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	(
0	LP001002	Male	No	0	Graduate	No	5849	_
1	LP001003	Male	Yes	1	Graduate	No	4583	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	
4	LP001008	Male	No	0	Graduate	No	6000	
609	LP002978	Female	No	0	Graduate	No	2900	
610	LP002979	Male	Yes	3+	Graduate	No	4106	
611	LP002983	Male	Yes	1	Graduate	No	8072	
612	LP002984	Male	Yes	2	Graduate	No	7583	
613	LP002990	Female	No	0	Graduate	Yes	4583	

614 rows × 16 columns

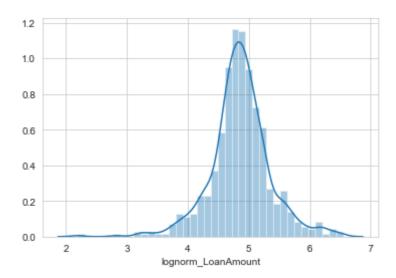


In [26]:

```
# Distplot LoanAmount
sns.distplot(df['lognorm_LoanAmount'])
```

Out[26]:

<matplotlib.axes._subplots.AxesSubplot at 0x20c1b548d88>

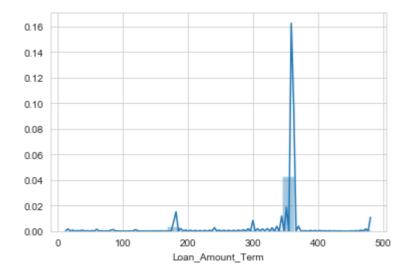


In [27]:

```
# Distplot Loan_Amount_Term
sns.distplot(df['Loan_Amount_Term'], kde_kws={'bw': 0})
```

Out[27]:

<matplotlib.axes._subplots.AxesSubplot at 0x20c1b6e3a08>



In [28]:

```
# Log the Loan_Amount_Term

df['lognorm_Loan_Amount_Term'] = np.log(df['Loan_Amount_Term'])
df
```

Out[28]:

Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	(
LP001002	Male	No	0	Graduate	No	5849	
LP001003	Male	Yes	1	Graduate	No	4583	
LP001005	Male	Yes	0	Graduate	Yes	3000	
LP001006	Male	Yes	0	Not Graduate	No	2583	
LP001008	Male	No	0	Graduate	No	6000	
LP002978	Female	No	0	Graduate	No	2900	
LP002979	Male	Yes	3+	Graduate	No	4106	
LP002983	Male	Yes	1	Graduate	No	8072	
LP002984	Male	Yes	2	Graduate	No	7583	
LP002990	Female	No	0	Graduate	Yes	4583	
	LP001002 LP001005 LP001006 LP001008 LP002978 LP002979 LP002983 LP002984	LP001002 Male LP001003 Male LP001005 Male LP001006 Male LP001008 Male LP002978 Female LP002979 Male LP002983 Male LP002984 Male	LP001002 Male No LP001003 Male Yes LP001005 Male Yes LP001006 Male Yes LP001008 Male No LP002978 Female No LP002979 Male Yes LP002983 Male Yes LP002984 Male Yes	LP001002 Male No 0 LP001003 Male Yes 1 LP001005 Male Yes 0 LP001006 Male Yes 0 LP001008 Male No 0 LP002978 Female No 0 LP002979 Male Yes 3+ LP002983 Male Yes 1 LP002984 Male Yes 2	LP001002 Male No 0 Graduate LP001003 Male Yes 1 Graduate LP001005 Male Yes 0 Graduate LP001006 Male Yes 0 Not Graduate LP001008 Male No 0 Graduate LP002978 Female No 0 Graduate LP002979 Male Yes 3+ Graduate LP002983 Male Yes 1 Graduate LP002984 Male Yes 2 Graduate	LP001002 Male No 0 Graduate No LP001003 Male Yes 1 Graduate No LP001005 Male Yes 0 Graduate Yes LP001006 Male Yes 0 Graduate No LP001008 Male No 0 Graduate No LP002978 Female No 0 Graduate No LP002979 Male Yes 3+ Graduate No LP002983 Male Yes 1 Graduate No LP002984 Male Yes 2 Graduate No	LP001002 Male No 0 Graduate No 5849 LP001003 Male Yes 1 Graduate No 4583 LP001005 Male Yes 0 Graduate Yes 3000 LP001006 Male Yes 0 Not Graduate No 2583 LP001008 Male No 0 Graduate No 6000 LP002978 Female No 0 Graduate No 2900 LP002979 Male Yes 3+ Graduate No 4106 LP002983 Male Yes 1 Graduate No 8072 LP002984 Male Yes 2 Graduate No 7583

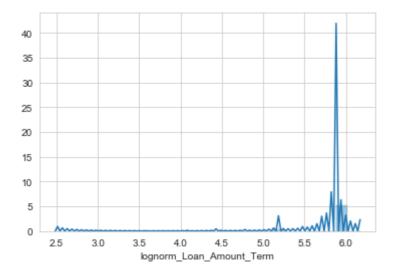
614 rows × 17 columns

In [29]:

```
# Distplot Loan_Amount_Term
sns.distplot(df['lognorm_Loan_Amount_Term'],kde_kws={'bw':0})
```

Out[29]:

<matplotlib.axes._subplots.AxesSubplot at 0x20c1b78ae08>



In [35]:

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

Out[35]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	(
0	LP001002	Male	No	0	Graduate	No	5849	
1	LP001003	Male	Yes	1	Graduate	No	4583	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	
4	LP001008	Male	No	0	Graduate	No	6000	
609	LP002978	Female	No	0	Graduate	No	2900	
610	LP002979	Male	Yes	3+	Graduate	No	4106	
611	LP002983	Male	Yes	1	Graduate	No	8072	
612	LP002984	Male	Yes	2	Graduate	No	7583	
613	LP002990	Female	No	0	Graduate	Yes	4583	
044	40							

614 rows × 18 columns

In [41]:

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

E:\SOFTWARES\ANACONDA\lib\site-packages\pandas\core\series.py:679: Runtime
Warning: divide by zero encountered in log
 result = getattr(ufunc, method)(*inputs, **kwargs)

Out[41]:

		Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	(
	0	LP001002	Male	No	0	Graduate	No	5849	
	1	LP001003	Male	Yes	1	Graduate	No	4583	
	2	LP001005	Male	Yes	0	Graduate	Yes	3000	
	3	LP001006	Male	Yes	0	Not Graduate	No	2583	
	4	LP001008	Male	No	0	Graduate	No	6000	
6	609	LP002978	Female	No	0	Graduate	No	2900	
6	310	LP002979	Male	Yes	3+	Graduate	No	4106	
(611	LP002983	Male	Yes	1	Graduate	No	8072	
6	312	LP002984	Male	Yes	2	Graduate	No	7583	
6	313	LP002990	Female	No	0	Graduate	Yes	4583	

614 rows × 21 columns

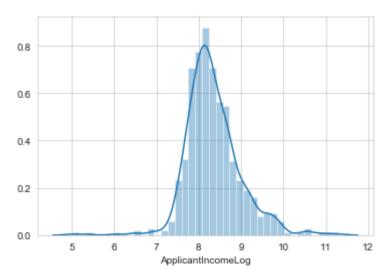
4

In [38]:

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

Out[38]:

<matplotlib.axes._subplots.AxesSubplot at 0x20c1baf99c8>



In []:

In []: