In [1]:
```python
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
import matplotlib
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

In [4]:
```python
df = pd.read_csv("F:/ml/loan_prediction/trainloan.csv")
df.head()
```

Out[4]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0

In [5]:
```python
df.describe()
```

Out[5]:

	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

In [6]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
Loan_ID              614 non-null object
Gender               601 non-null object
Married              611 non-null object
Dependents           599 non-null object
Education            614 non-null object
Self_Employed        582 non-null object
ApplicantIncome      614 non-null int64
CoapplicantIncome    614 non-null float64
LoanAmount           592 non-null float64
Loan_Amount_Term     600 non-null float64
Credit_History       564 non-null float64
Property_Area        614 non-null object
Loan_Status          614 non-null object
dtypes: float64(4), int64(1), object(8)
memory usage: 62.4+ KB
```

In [7]:
```python
# find the null values
df.isnull().sum()
```

Out[7]:
```
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
dtype: int64
```

In [8]:
```python
df['LoanAmount'] = df['LoanAmount'].fillna(df['LoanAmount'].mean())
df['Loan_Amount_Term'] = df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mean())
df['Credit_History'] = df['Credit_History'].fillna(df['Credit_History'].mean())
```

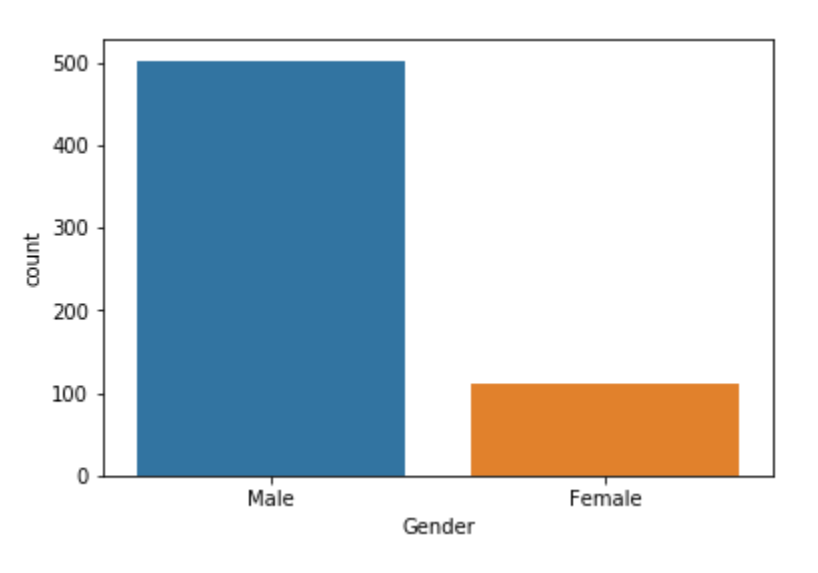
In [9]:
```python
# categorical terms - mode
df['Gender'] = df["Gender"].fillna(df['Gender'].mode()[0])
df['Married'] = df["Married"].fillna(df['Married'].mode()[0])
df['Dependents'] = df["Dependents"].fillna(df['Dependents'].mode()[0])
df['Self_Employed'] = df["Self_Employed"].fillna(df['Self_Employed'].mode()[0])
```

In [10]:
```python
df.isnull().sum()
```

Out[10]:
```
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
dtype: int64
```

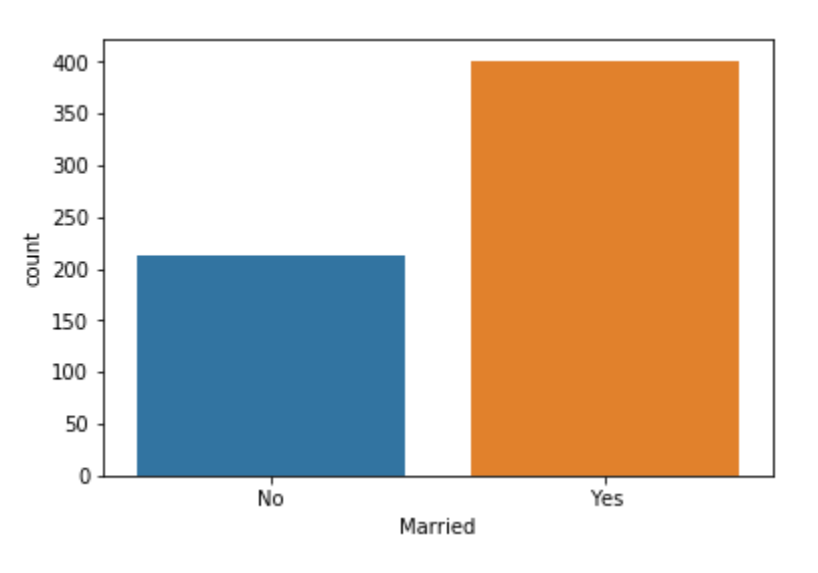
In [11]:
```python
sns.countplot(df['Gender'])
```

Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0x178361bd518>



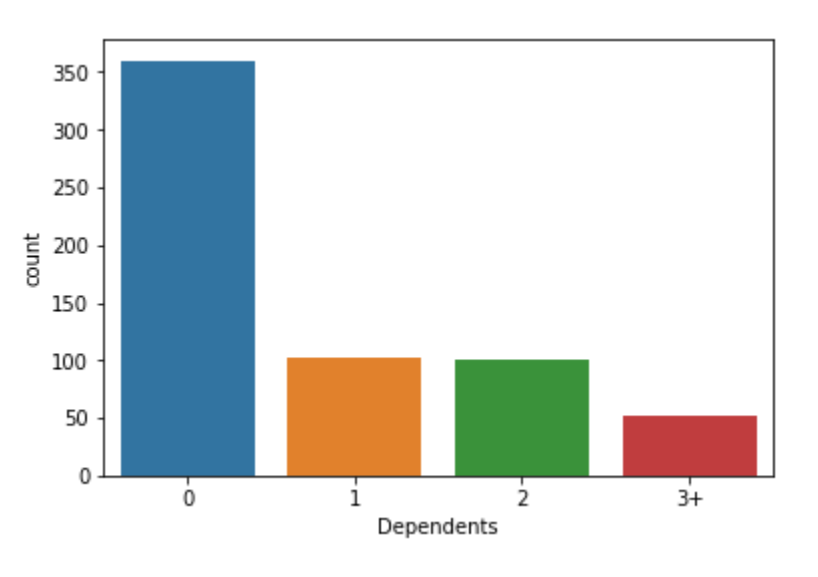
In [12]:
```python
sns.countplot(df['Married'])
```

Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x17835b6acf8>



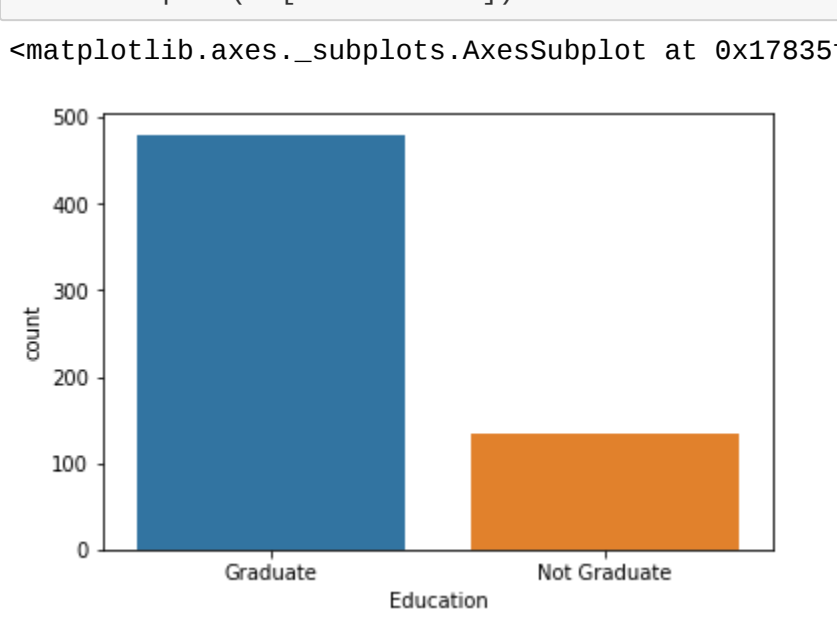
In [13]:
```python
sns.countplot(df['Dependents'])
```

Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x17835d90438>



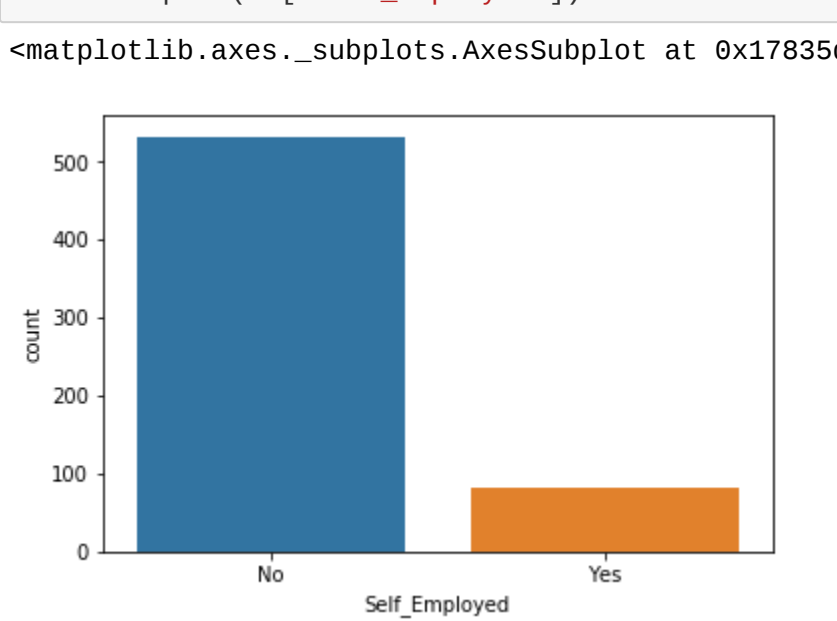
In [14]:
```python
sns.countplot(df['Education'])
```

Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x17835fe8d68>



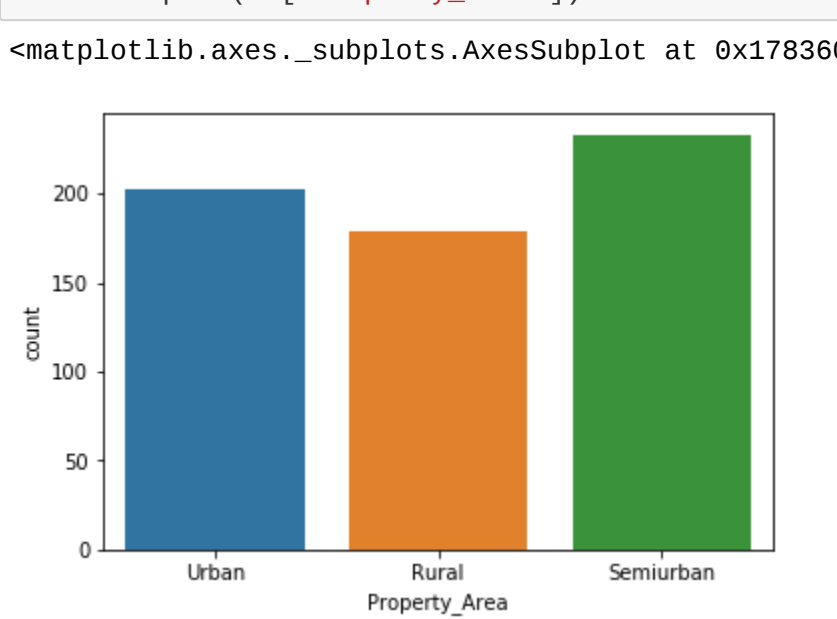
In [15]:
```python
sns.countplot(df['Self_Employed'])
```

Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x17835d90080>



In [16]:
```python
sns.countplot(df['Property_Area'])
```

Out[16]: <matplotlib.axes._subplots.AxesSubplot at 0x178360c27b8>

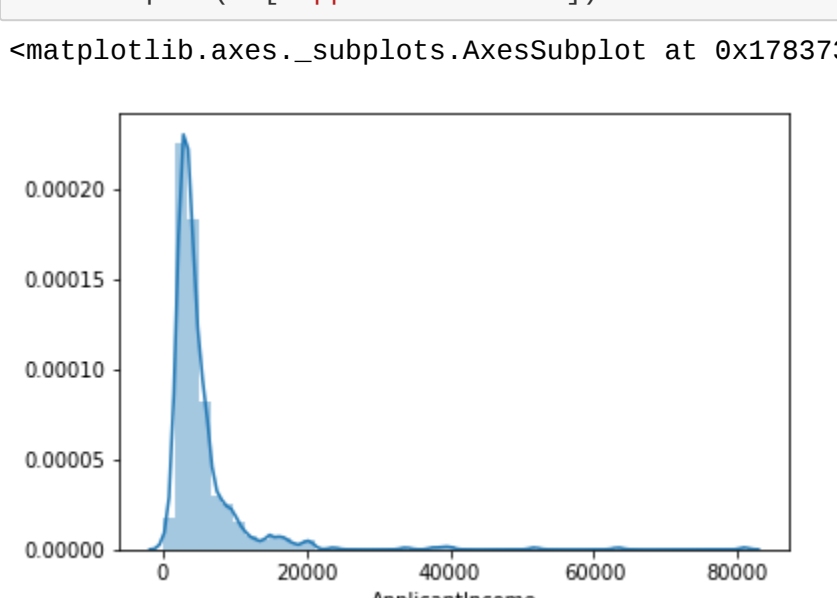


In [17]:
```python
sns.countplot(df['Loan_Status'])
```

Out[17]: <matplotlib.axes._subplots.AxesSubplot at 0x1783614ed68>

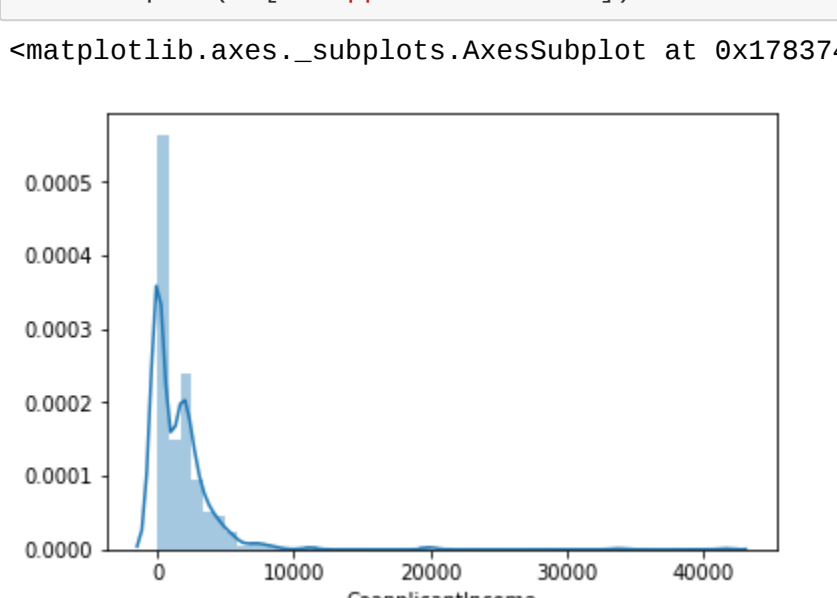
In [18]:
```python
sns.distplot(df["ApplicantIncome"])
```

Out[18]: <matplotlib.axes._subplots.AxesSubplot at 0x178373faf98>



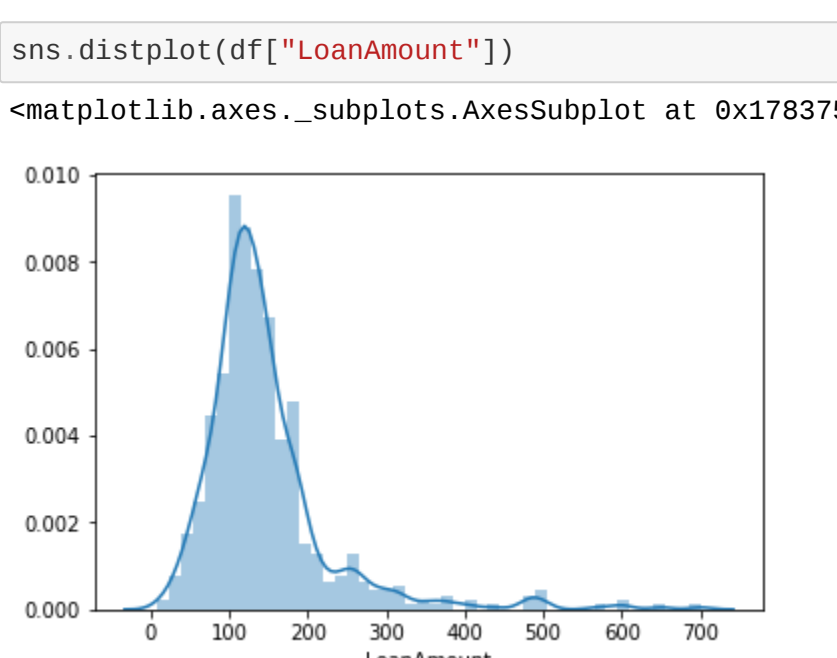
In [19]:
```python
sns.distplot(df["CoapplicantIncome"])
```

Out[19]: <matplotlib.axes._subplots.AxesSubplot at 0x178374874e0>



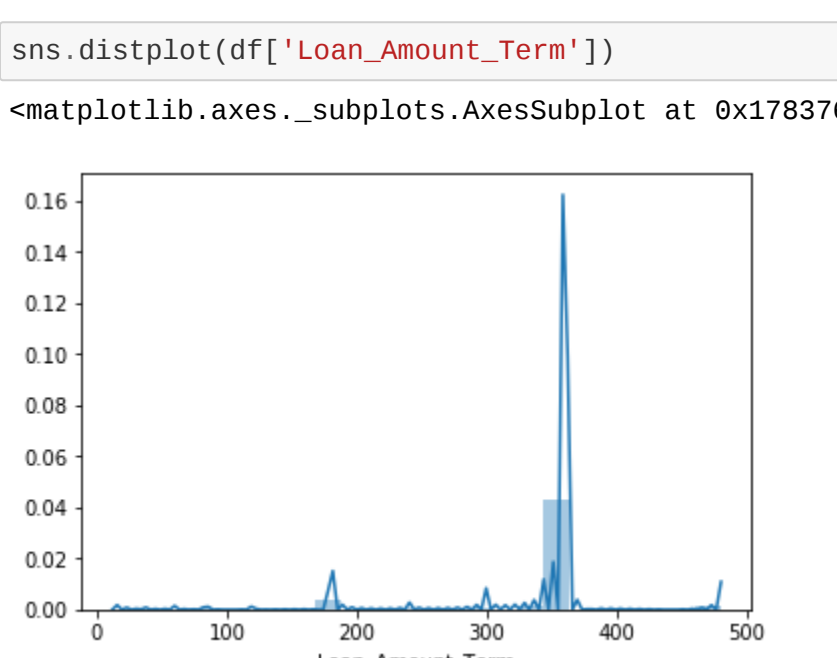
In [20]:
```python
sns.distplot(df["LoanAmount"])
```

Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0x17837587ef0>



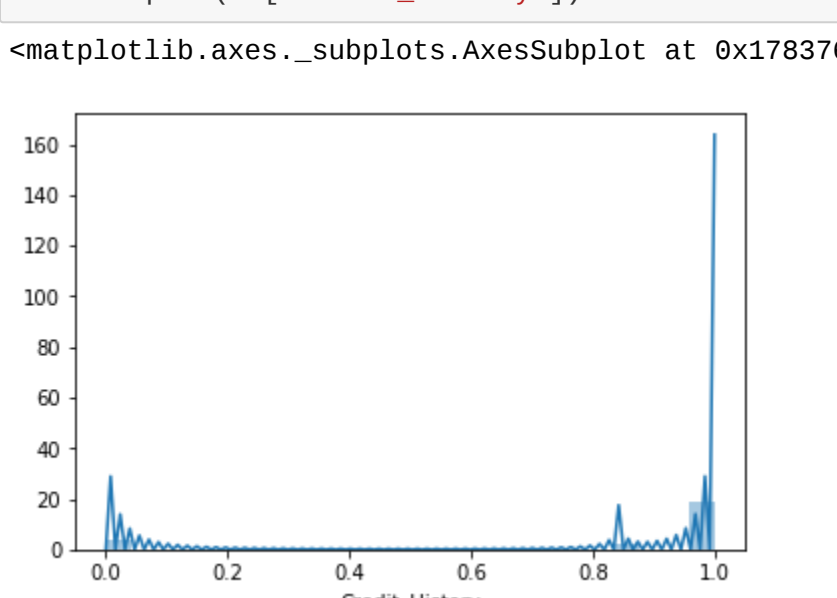
In [21]:
```python
sns.distplot(df['Loan_Amount_Term'])
```

Out[21]: <matplotlib.axes._subplots.AxesSubplot at 0x17837649a58>



In [22]:
```python
sns.distplot(df['Credit_History'])
```

Out[22]: <matplotlib.axes._subplots.AxesSubplot at 0x17837677748>



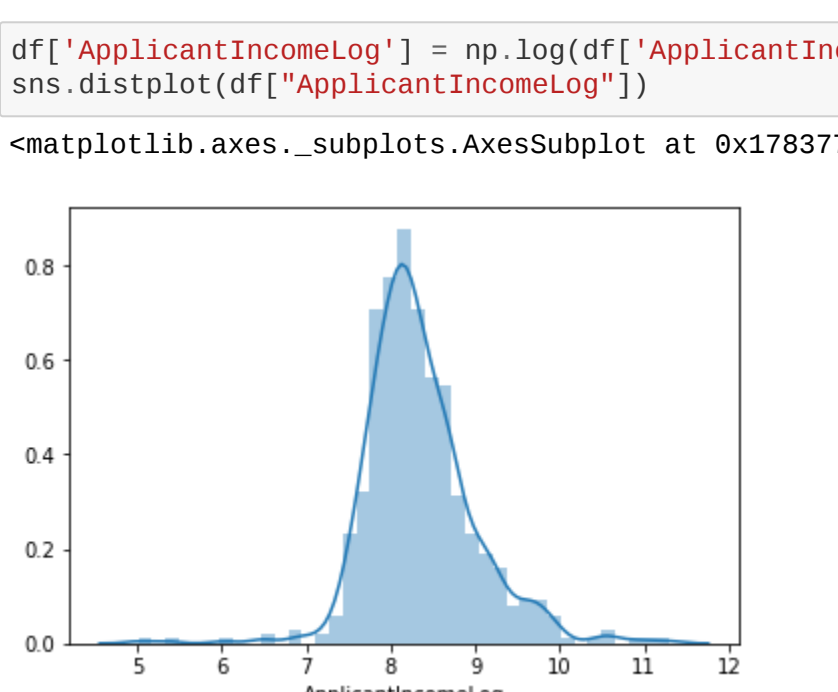
In [23]:
```python
df['Total_Income'] = df['ApplicantIncome'] + df['CoapplicantIncome']
df.head()
```

Out[23]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount
0	LP001002	Male	No	0	Graduate	No	5849	0.0	146.412162
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.000000
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.000000
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.000000
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.000000

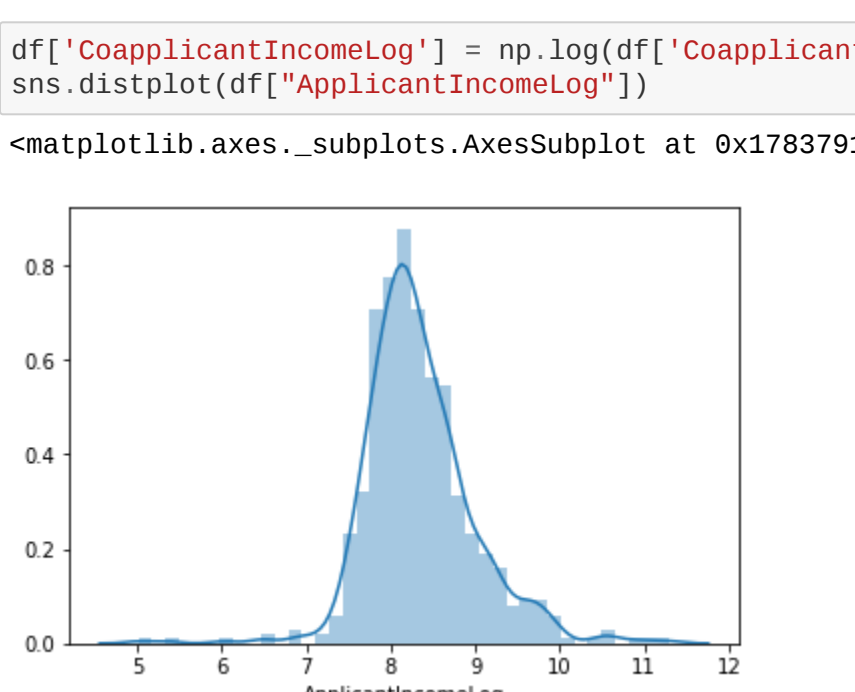
In [25]:
```python
df['ApplicantIncomeLog'] = np.log(df['ApplicantIncome'])
sns.distplot(df["ApplicantIncomeLog"])
```

Out[25]: <matplotlib.axes._subplots.AxesSubplot at 0x17837758c18>



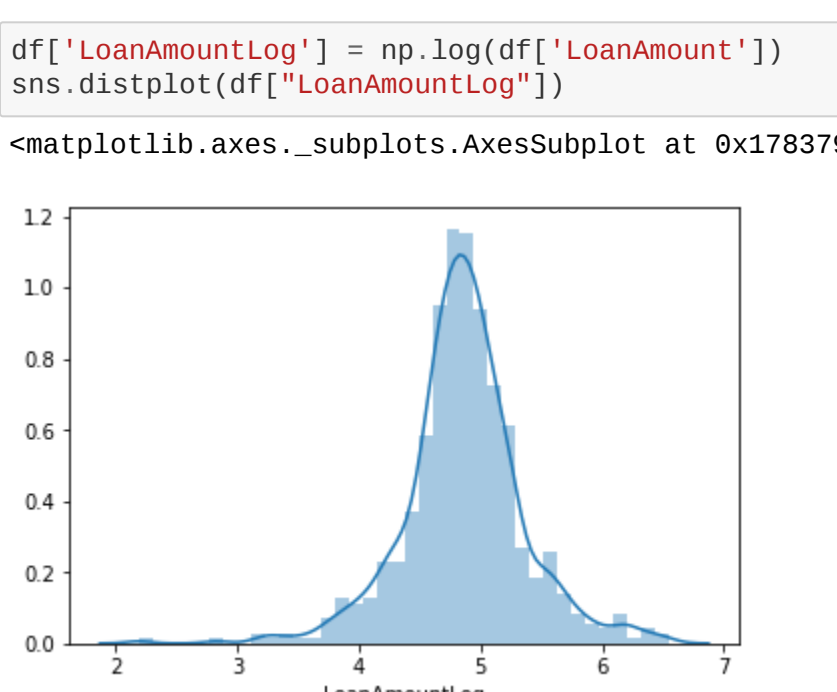
In [26]:
```python
df['CoapplicantIncomeLog'] = np.log(df['CoapplicantIncome'])
sns.distplot(df["ApplicantIncomeLog"])
```

Out[26]: <matplotlib.axes._subplots.AxesSubplot at 0x1783791ca58>



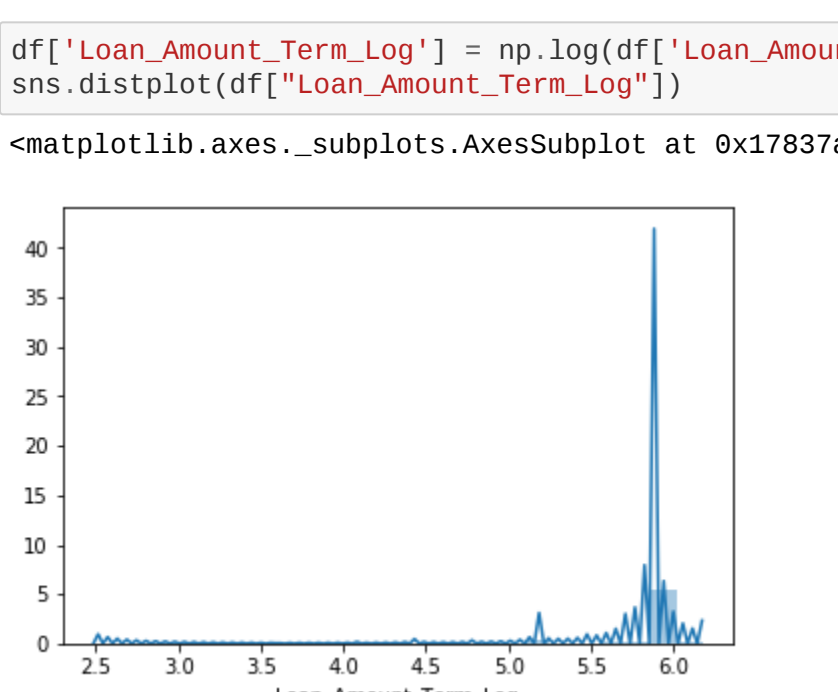
In [27]:
```python
df['LoanAmountLog'] = np.log(df['LoanAmount'])
sns.distplot(df["LoanAmountLog"])
```

Out[27]: <matplotlib.axes._subplots.AxesSubplot at 0x178379d4dd8>



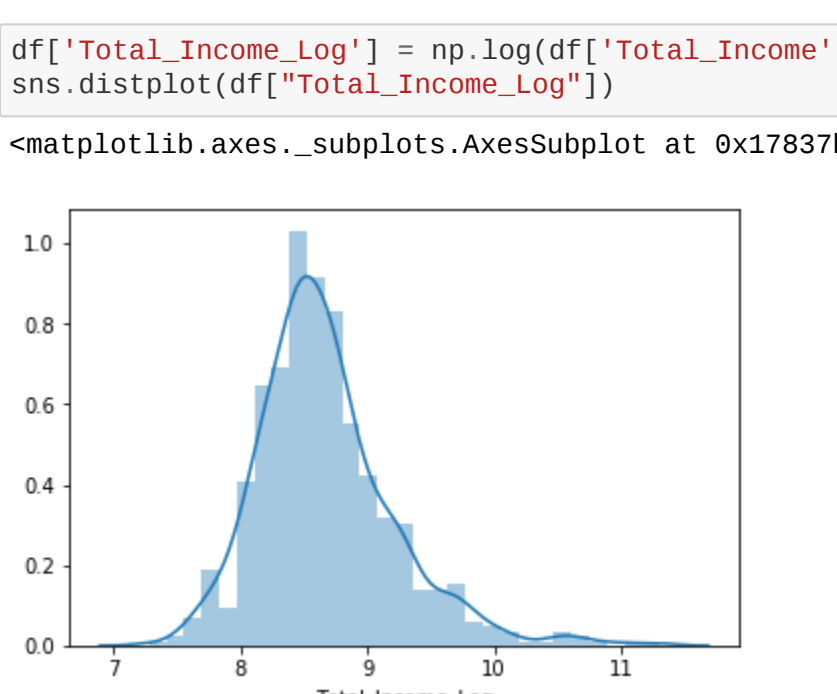
In [29]:
```python
df['Loan_Amount_Term_Log'] = np.log(df['Loan_Amount_Term'])
sns.distplot(df["Loan_Amount_Term_Log"])
```

Out[29]: <matplotlib.axes._subplots.AxesSubplot at 0x17837ae66a0>



In [30]:
```python
df['Total_Income_Log'] = np.log(df['Total_Income'])
sns.distplot(df["Total_Income_Log"])
```

Out[30]: <matplotlib.axes._subplots.AxesSubplot at 0x17837bcd2b0>



In [31]:
```python
df.head()
```

Out[31]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount
0	LP001002	Male	No	0	Graduate	No	5849	0.0	146.412162
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.000000
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.000000
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.000000
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.000000

In [32]:
```python
cols = ['ApplicantIncome', 'CoapplicantIncome', "LoanAmount", "Loan_Amount_Term", "Total_Inc
ome", 'Loan_ID', 'CoapplicantIncomeLog']
df = df.drop(columns=cols, axis=1)
df.head()
```

Out[32]:

	Gender	Married	Dependents	Education	Self_Employed	Credit_History	Property_Area	Loan_Status	ApplicantIncomeLog
0	Male	No	0	Graduate	No	1.0	Urban	Y	8.674026
1	Male	Yes	1	Graduate	No	1.0	Rural	N	8.430109
2	Male	Yes	0	Graduate	Yes	1.0	Urban	Y	8.006368
3	Male	Yes	0	Not Graduate	No	1.0	Urban	Y	7.856707
4	Male	No	0	Graduate	No	1.0	Urban	Y	8.699515

In [33]:
```python
from sklearn.preprocessing import LabelEncoder
cols = ['Gender',"Married","Education",'Self_Employed',"Property_Area","Loan_Status","Depend
ents"]
le = LabelEncoder()
for col in cols:
```

In [34]:
```python
df.head()
```

Out[34]:

	Gender	Married	Dependents	Education	Self_Employed	Credit_History	Property_Area	Loan_Status	ApplicantIncomeLog
0	1	0	0	0	0	1.0	2	1	8.674026
1	1	1	1	0	0	1.0	0	0	8.430109
2	1	1	0	0	1	1.0	2	1	8.006368
3	1	1	0	1	0	1.0	2	1	7.856707
4	1	0	0	0	0	1.0	2	1	8.699515

In [35]:
```python
X = df.drop(columns=['Loan_Status'], axis=1)
y = df['Loan_Status']
```

In [36]:
```python
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)
```

In [38]:
```python
from sklearn.model_selection import cross_val_score
def classify(model, x, y):
    x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=4
2)
    model.fit(x_train, y_train)
    print("Accuracy is", model.score(x_test, y_test)*100)
    score = cross_val_score(model, x, y, cv=5)
```

In [39]:
```python
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
classify(model, X, y)
```

```
Accuracy is 77.27272727272727
Cross validation is 80.79587519830778
```

In [ ]:

In [ ]: