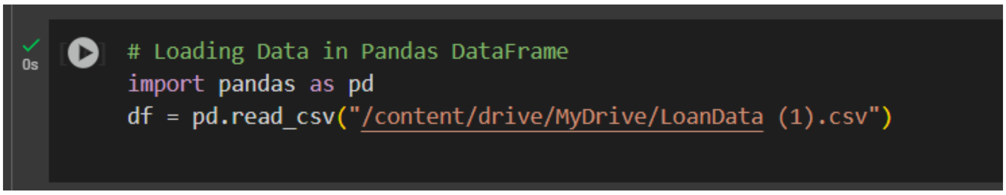
CASE STUDY – 1 (Python)

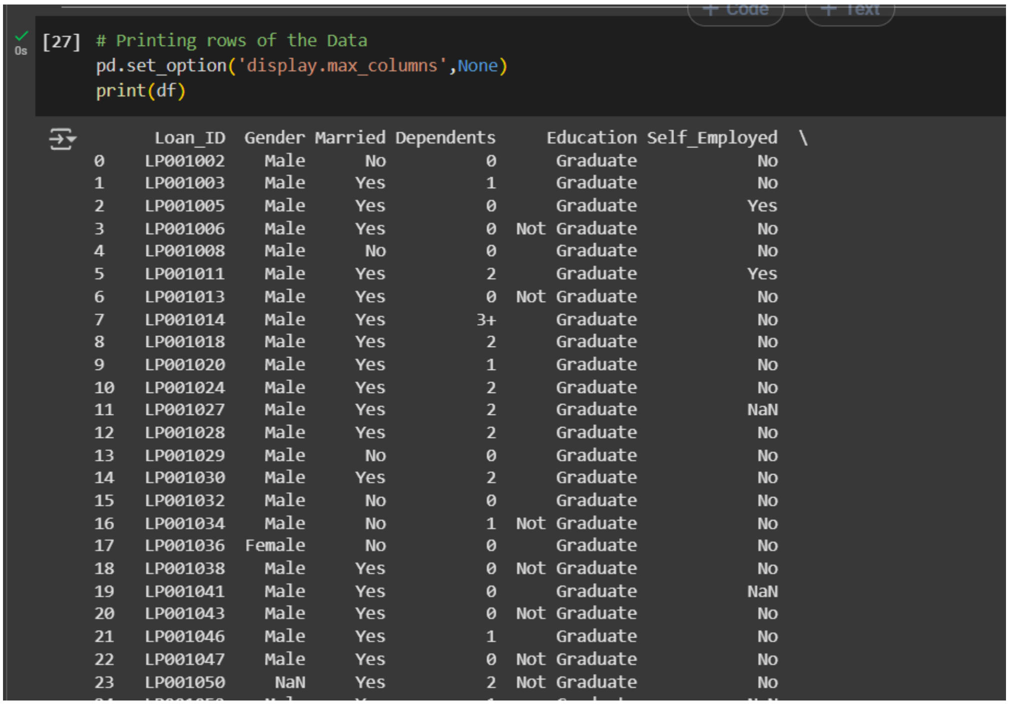
TRAINEE NAME: Swathi Baskaran

1. Loading data in Pandas Dataframe



```
# Loading Data in Pandas DataFrame
import pandas as pd
df = pd.read_csv("/content/drive/MyDrive/LoanData (1).csv")
```

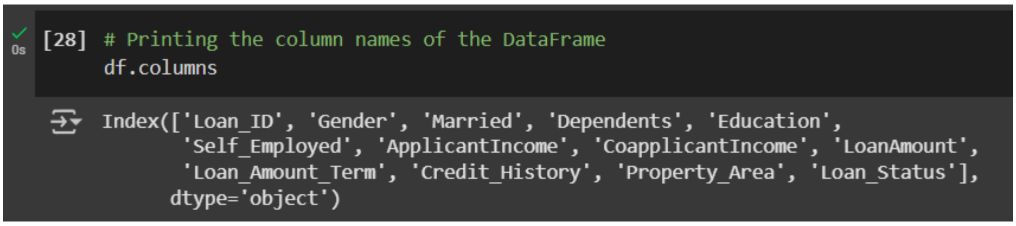
2. Printing rows of the Data



```
[27] # Printing rows of the Data
     pd.set_option('display.max_columns',None)
     print(df)
```

```
      Loan_ID  Gender Married Dependents     Education Self_Employed  \
0    LP001002    Male      No          0      Graduate            No
1    LP001003    Male     Yes          1      Graduate            No
2    LP001005    Male     Yes          0      Graduate           Yes
3    LP001006    Male     Yes          0  Not Graduate            No
4    LP001008    Male      No          0      Graduate            No
5    LP001011    Male     Yes          2      Graduate           Yes
6    LP001013    Male     Yes          0  Not Graduate            No
7    LP001014    Male     Yes         3+      Graduate            No
8    LP001018    Male     Yes          2      Graduate            No
9    LP001020    Male     Yes          1      Graduate            No
10   LP001024    Male     Yes          2      Graduate            No
11   LP001027    Male     Yes          2      Graduate           NaN
12   LP001028    Male     Yes          2      Graduate            No
13   LP001029    Male      No          0      Graduate            No
14   LP001030    Male     Yes          2      Graduate            No
15   LP001032    Male      No          0      Graduate            No
16   LP001034    Male      No          1  Not Graduate            No
17   LP001036  Female      No          0      Graduate            No
18   LP001038    Male     Yes          0  Not Graduate            No
19   LP001041    Male     Yes          0      Graduate           NaN
20   LP001043    Male     Yes          0  Not Graduate            No
21   LP001046    Male     Yes          1      Graduate            No
22   LP001047    Male     Yes          0  Not Graduate            No
23   LP001050     NaN     Yes          2  Not Graduate            No
```

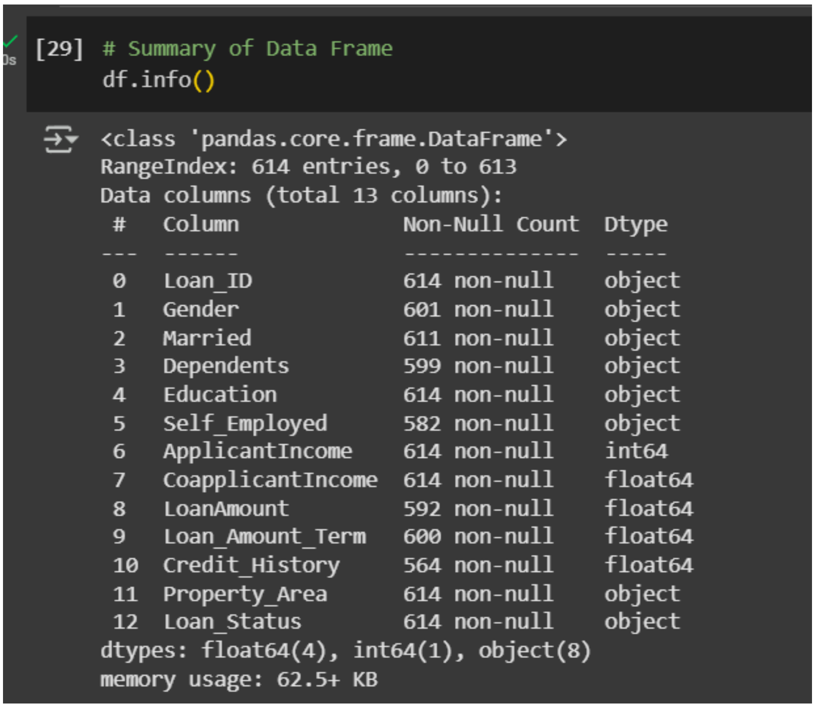
3. Printing the column names of the DataFrame



```
[28] # Printing the column names of the DataFrame
     df.columns
```

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

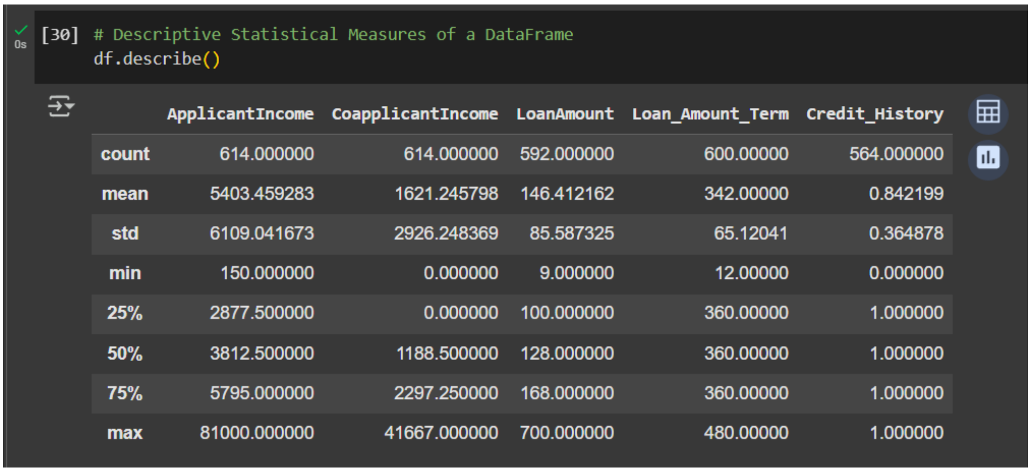
4. Summary of Data Frame



```
[29] # Summary of Data Frame
     df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Loan_ID            614 non-null    object
 1   Gender             601 non-null    object
 2   Married            611 non-null    object
 3   Dependents         599 non-null    object
 4   Education          614 non-null    object
 5   Self_Employed      582 non-null    object
 6   ApplicantIncome    614 non-null    int64
 7   CoapplicantIncome  614 non-null    float64
 8   LoanAmount         592 non-null    float64
 9   Loan_Amount_Term   600 non-null    float64
 10  Credit_History     564 non-null    float64
 11  Property_Area      614 non-null    object
 12  Loan_Status        614 non-null    object
dtypes: float64(4), int64(1), object(8)
memory usage: 62.5+ KB
```

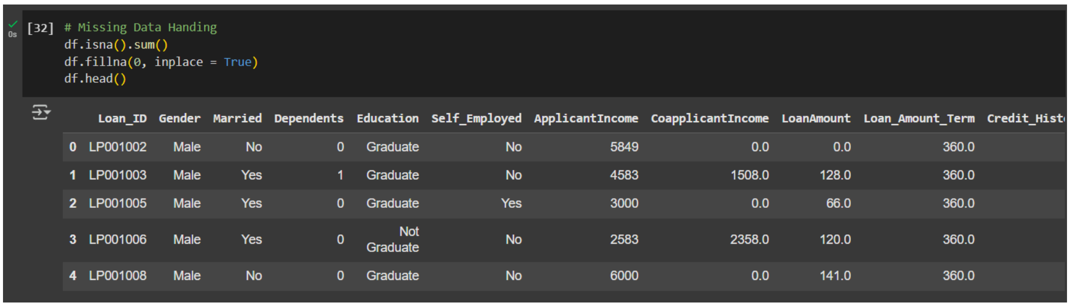
5. Descriptive Statistical Measures of a DataFrame



```
[30] # Descriptive Statistical Measures of a DataFrame
     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

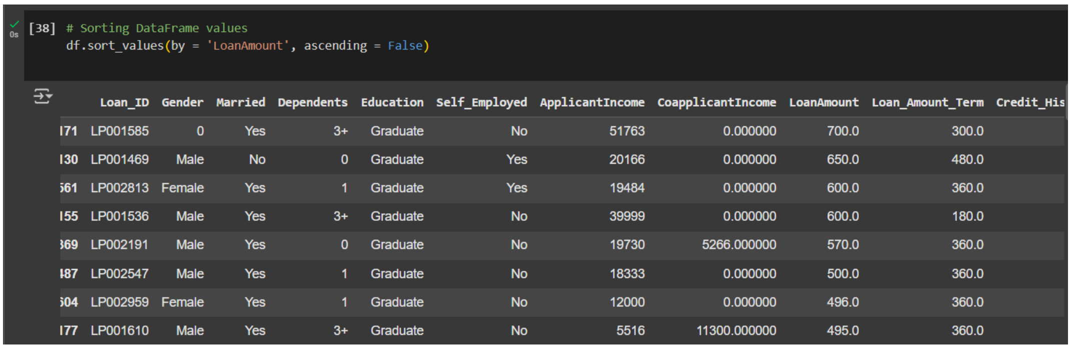
6. Missing Data Handing



```
[32] # Missing Data Handing
     df.isna().sum()
     df.fillna(0, inplace = True)
     df.head()
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_Hist
0	LP001002	Male	No	0	Graduate	No	5849	0.0	0.0	360.0	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	

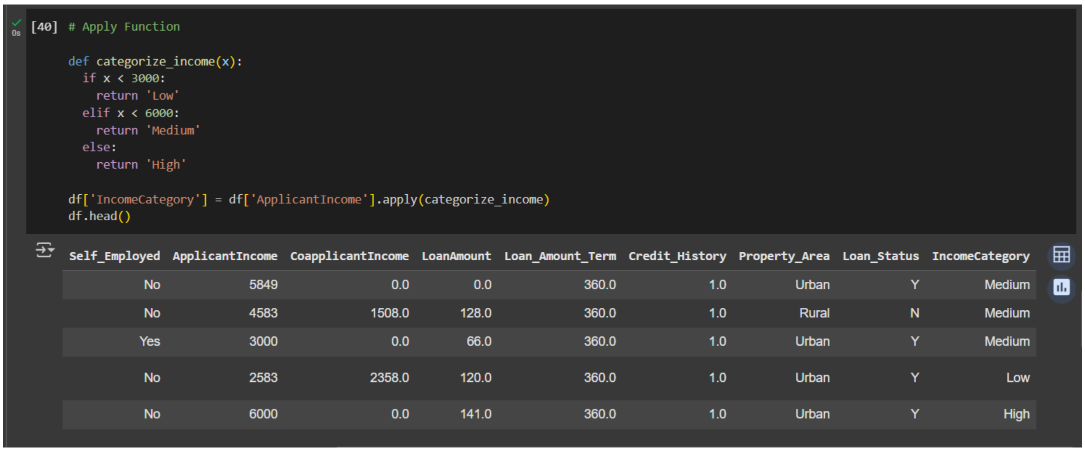
7. Sorting DataFrame values



```
[38] # Sorting DataFrame values
     df.sort_values(by = 'LoanAmount', ascending = False)
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_His
171	LP001585	0	Yes	3+	Graduate	No	51763	0.000000	700.0	300.0	
130	LP001469	Male	No	0	Graduate	Yes	20166	0.000000	650.0	480.0	
561	LP002813	Female	Yes	1	Graduate	Yes	19484	0.000000	600.0	360.0	
155	LP001536	Male	Yes	3+	Graduate	No	39999	0.000000	600.0	180.0	
369	LP002191	Male	Yes	0	Graduate	No	19730	5266.000000	570.0	360.0	
487	LP002547	Male	Yes	1	Graduate	No	18333	0.000000	500.0	360.0	
504	LP002959	Female	Yes	1	Graduate	No	12000	0.000000	496.0	360.0	
177	LP001610	Male	Yes	3+	Graduate	No	5516	11300.000000	495.0	360.0	

8. Apply Function



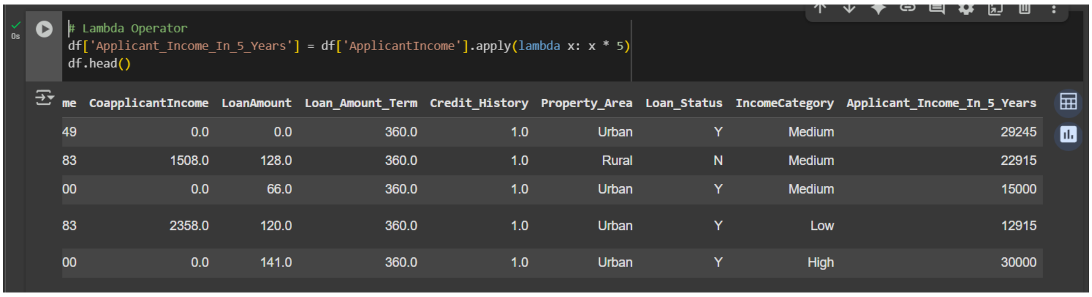
```
[40] # Apply Function

     def categorize_income(x):
       if x < 3000:
         return 'Low'
       elif x < 6000:
         return 'Medium'
       else:
         return 'High'

     df['IncomeCategory'] = df['ApplicantIncome'].apply(categorize_income)
     df.head()
```

Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status	IncomeCategory
No	5849	0.0	0.0	360.0	1.0	Urban	Y	Medium
No	4583	1508.0	128.0	360.0	1.0	Rural	N	Medium
Yes	3000	0.0	66.0	360.0	1.0	Urban	Y	Medium
No	2583	2358.0	120.0	360.0	1.0	Urban	Y	Low
No	6000	0.0	141.0	360.0	1.0	Urban	Y	High

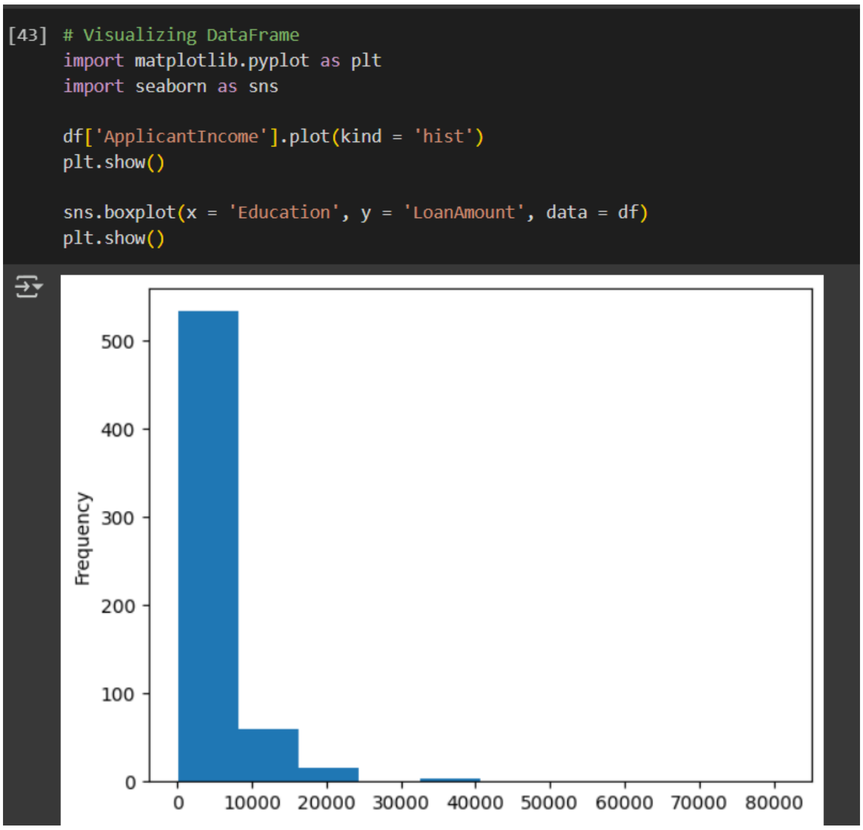
9. Lambda operator



```
# Lambda Operator
df['Applicant_Income_In_5_Years'] = df['ApplicantIncome'].apply(lambda x: x * 5)
df.head()
```

me	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status	IncomeCategory	Applicant_Income_In_5_Years
49	0.0	0.0	360.0	1.0	Urban	Y	Medium	29245
83	1508.0	128.0	360.0	1.0	Rural	N	Medium	22915
00	0.0	66.0	360.0	1.0	Urban	Y	Medium	15000
83	2358.0	120.0	360.0	1.0	Urban	Y	Low	12915
00	0.0	141.0	360.0	1.0	Urban	Y	High	30000

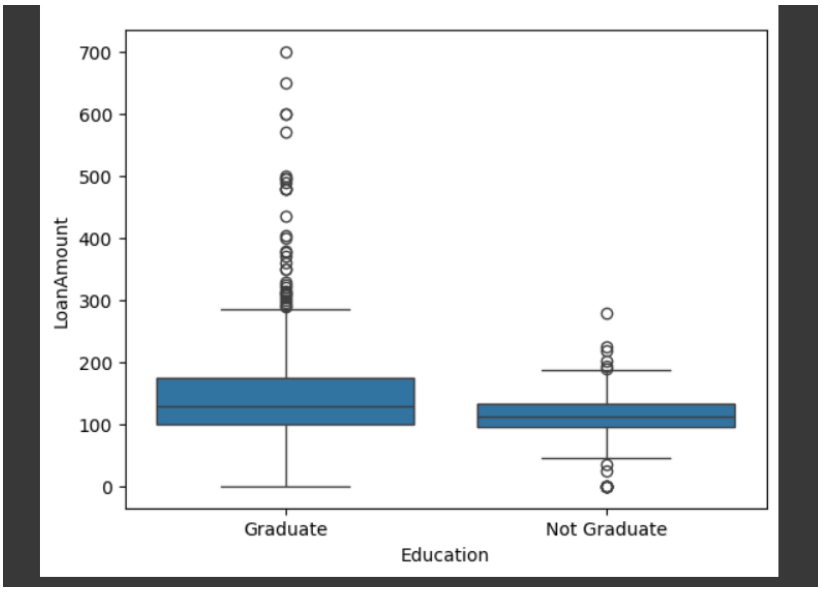
10.Visualizing DataFrame



```
[43] # Visualizing DataFrame
     import matplotlib.pyplot as plt
     import seaborn as sns

     df['ApplicantIncome'].plot(kind = 'hist')
     plt.show()

     sns.boxplot(x = 'Education', y = 'LoanAmount', data = df)
     plt.show()
```



LoanAmount
Graduate
Not Graduate
Education