In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
df=pd.read_csv(r"C:\Users\user\Downloads\C8_loan-train - C8_loan-train.csv")
df
```

Out[2]:

| | Loan_ID | Gender | Married | Dependents | Education | Self_Employed | ApplicantIncome | Coappli |
|---|---|---|---|---|---|---|---|---|
| **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]:
```
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
```

In [4]:
```
data=df.fillna(0)
data
```

Out[4]:

| | Loan_ID | Gender | Married | Dependents | Education | Self_Employed | ApplicantIncome | Coappli |
|---|---|---|---|---|---|---|---|---|
| **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 [5]:
```
from sklearn.linear_model import LogisticRegression
```

In [6]:
```
df1=data[['ApplicantIncome','CoapplicantIncome','LoanAmount','Loan_Amount_Term',
df1
```

Out[6]:

| | ApplicantIncome | CoapplicantIncome | LoanAmount | Loan_Amount_Term | Credit_History |
|---|---|---|---|---|---|
| **0** | 5849 | 0.0 | 0.0 | 360.0 | 1.0 |
| **1** | 4583 | 1508.0 | 128.0 | 360.0 | 1.0 |
| **2** | 3000 | 0.0 | 66.0 | 360.0 | 1.0 |
| **3** | 2583 | 2358.0 | 120.0 | 360.0 | 1.0 |
| **4** | 6000 | 0.0 | 141.0 | 360.0 | 1.0 |
| **...** | ... | ... | ... | ... | ... |
| **609** | 2900 | 0.0 | 71.0 | 360.0 | 1.0 |
| **610** | 4106 | 0.0 | 40.0 | 180.0 | 1.0 |
| **611** | 8072 | 240.0 | 253.0 | 360.0 | 1.0 |
| **612** | 7583 | 0.0 | 187.0 | 360.0 | 1.0 |
| **613** | 4583 | 0.0 | 133.0 | 360.0 | 0.0 |

614 rows × 5 columns

In [13]:
```
feature_matrix=df1.iloc[:,0:4]
target_vector=df1.iloc[:,-1]
```

In [14]:
```
feature_matrix.shape
```

Out[14]: (614, 4)

In [15]:
```
target_vector.shape
```

Out[15]: (614,)

In [16]:
```
from sklearn.preprocessing import StandardScaler
```

In [17]:
```
fs=StandardScaler().fit_transform(feature_matrix)
```

In [18]:
```
logr = LogisticRegression()
logr.fit(fs,target_vector)
```

Out[18]: LogisticRegression()

In [ ]:
```
observation=[[5,7,9,8,6]]
```

In [ ]:
```
prediction=logr.predict(observation)
print(prediction)
```

In [ ]:
```
logr.classes_
```

In [ ]:
```
logr.predict_proba(observation)[0][0]
```

In [ ]:
```
logr.predict_proba(observation)[0][0]
```

In [ ]:
```
import re
from sklearn.datasets import load_digits
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
```

In [ ]:
```
digits = load_digits()
digits
```

In [ ]:
```
plt.figure(figsize=(20,4))
for index,(image,label) in enumerate(zip(digits.data[0:5],digits.target[0:8])):
    plt.subplot(1,5,index+1)
    plt.imshow(np.reshape(image,(8,8)))
    plt.title("number\n"%label,fontsize=15)
```

In [ ]:
```
x_train,x_test,y_train,y_test=train_test_split(digits.data,digits.target,test_siz
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

In [ ]:
```
logre=LogisticRegression()
logre.fit(x_train,y_train)
```

In [ ]:
```
print(logre.predict(x_test))
```

In [ ]:
```
print(logre.score(x_test,y_test))
```