

The data set consists of 2000 samples from each of two categories. Five variables are

1. Income
2. Age
3. Loan
4. Loan to Income (engineered feature)
5. Default

```
import pandas as pd
```

```
default = pd.read_csv('https://github.com/ybifoundation/Dataset/raw/main/Credit%20Default.csv')
```

```
default.head()
```

	Income	Age	Loan	Loan to Income	Default
0	66155.92510	59.017015	8106.532131	0.122537	0
1	34415.15397	48.117153	6564.745018	0.190752	0
2	57317.17006	63.108049	8020.953296	0.139940	0
3	42709.53420	45.751972	6103.642260	0.142911	0
4	66952.68885	18.584336	8770.099235	0.130990	1

```
default.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 5 columns):
#   Column          Non-Null Count  Dtype  
---  -
0   Income          2000 non-null  float64
1   Age             2000 non-null  float64
2   Loan            2000 non-null  float64
3   Loan to Income  2000 non-null  float64
4   Default         2000 non-null  int64  
dtypes: float64(4), int64(1)
memory usage: 78.2 KB
```

```
default.describe()
```



	Income	Age	Loan	Loan to Income	Default
count	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000
mean	45331.600018	40.927143	4444.369695	0.098403	0.141500
std	14326.327119	13.262450	3045.410024	0.057620	0.348624
min	20014.489470	18.055189	1.377630	0.000049	0.000000
25%	32796.459720	29.062492	1939.708847	0.047903	0.000000
50%	45789.117310	41.382673	3974.719418	0.099437	0.000000

```
default['Default'].value_counts()
```

```
0    1717
1     283
Name: Default, dtype: int64
```

```
default.columns
```

```
Index(['Income', 'Age', 'Loan', 'Loan to Income', 'Default'], dtype='object')
```

```
default.columns
```

```
Index(['Income', 'Age', 'Loan', 'Loan to Income', 'Default'], dtype='object')
```

```
y = default['Default']
```

```
X = default.drop(['Default'],axis=1)
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y, train_size=0.7, random_state=2529)
```

```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

```
((1400, 4), (600, 4), (1400,), (600,))
```

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
```

```
model.fit(X_train,y_train)
```

```
▼ LogisticRegression
LogisticRegression()
```


0.95

```
print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
0	0.97	0.97	0.97	519
1	0.83	0.79	0.81	81
accuracy			0.95	600
macro avg	0.90	0.88	0.89	600
weighted avg	0.95	0.95	0.95	600

[Colab paid products](#) - [Cancel contracts here](#)

✓ 0s completed at 6:33 PM

