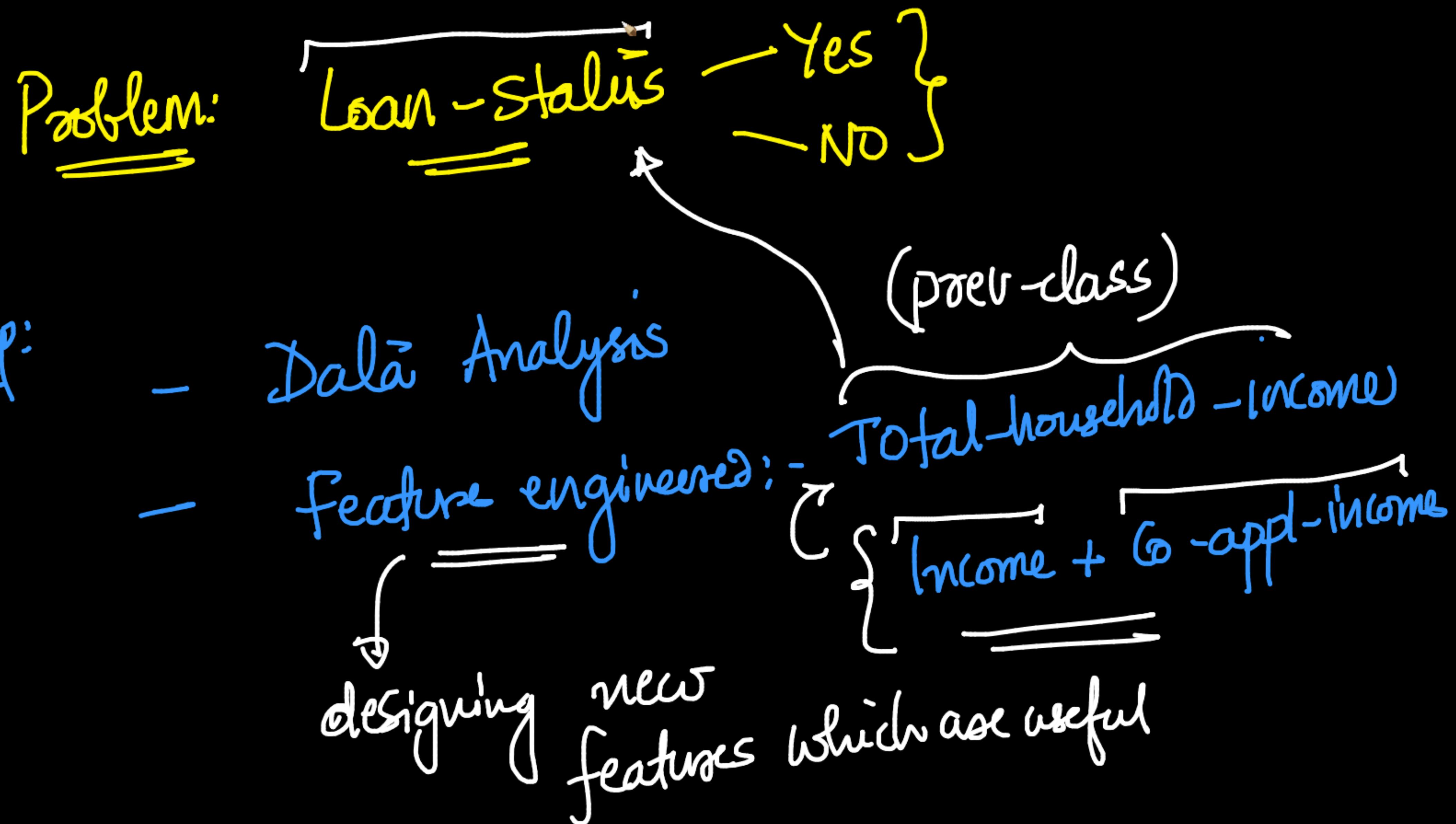




→ No class on Saturday ←

Next week: MWF → SQL  
TTS



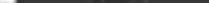
EDA\_FE.ipynb - Colaboratory

Code + Text

✓ RAM Disk

## ► Incomes

[ 1 ↣ 11 cells hidden]



#### ▼ Loan Amount and Loan Term

✓ [232] data[ 'Loan Amount Term' ].value\_counts()

✓	360.0	✓	512	→	30 yr
	180.0		44	→	15 yr
	480.0		15		
→	300.0		13		
	240.0		4		
	84.0		4		
	120.0		3		
	60.0		2		
	36.0		2		
	12.0		1	✓	

Name: Loan\_Amount\_Term, dtype: int

```
✓ [233] data['Loan_Amount_Term'] = (data['Loan_Amount_Term']/12).astype('float')
```

 EDA\_FE.ipynb - Colaboratory X

```
Code + Text
```

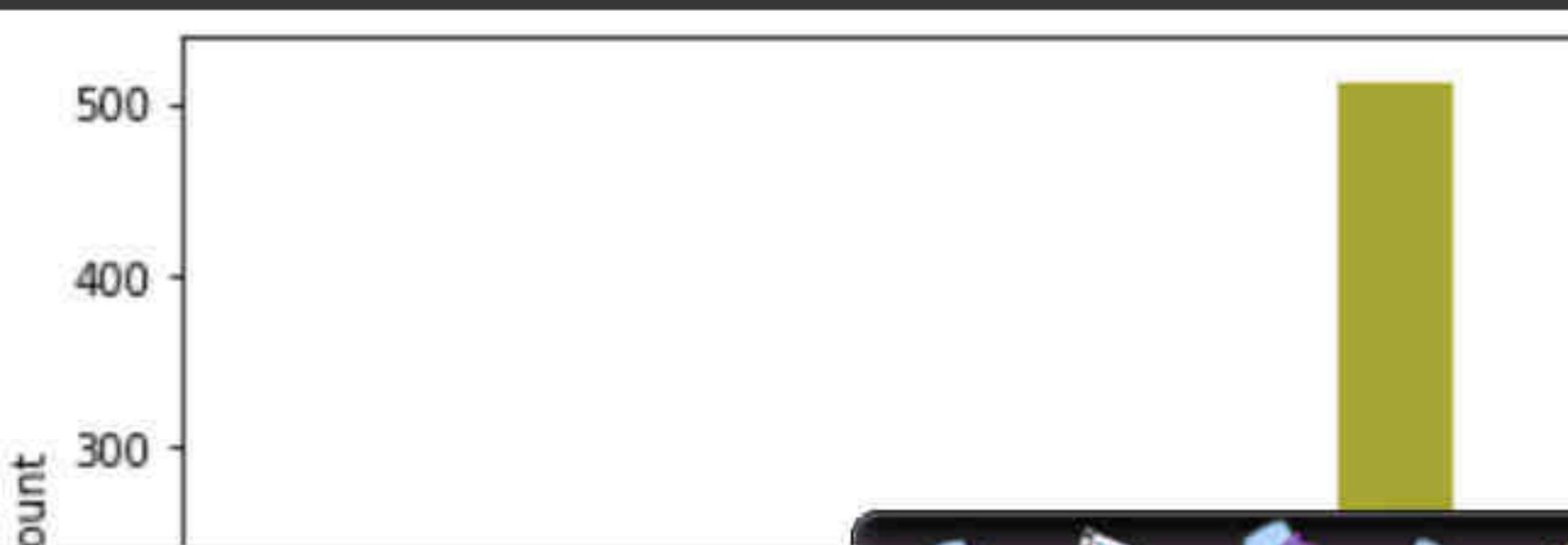
---

300.0	13
240.0	4
84.0	4
120.0	3
60.0	2
36.0	2
12.0	1

```
Name: Loan_Amount_Term, dtype: int
```

```
[233] data['Loan_Amount_Term'] = (data['Loan_Amount_Term']/12).astype('float')
```

```
sns.countplot(x='Loan_Amount_Term', data=data)
plt.xlabel("Term in years")
plt.show()
# Observation: We can clearly see that more than 90% of the loans were applied for 30 years.
```



EDA\_FE.ipynb - Colaboratory

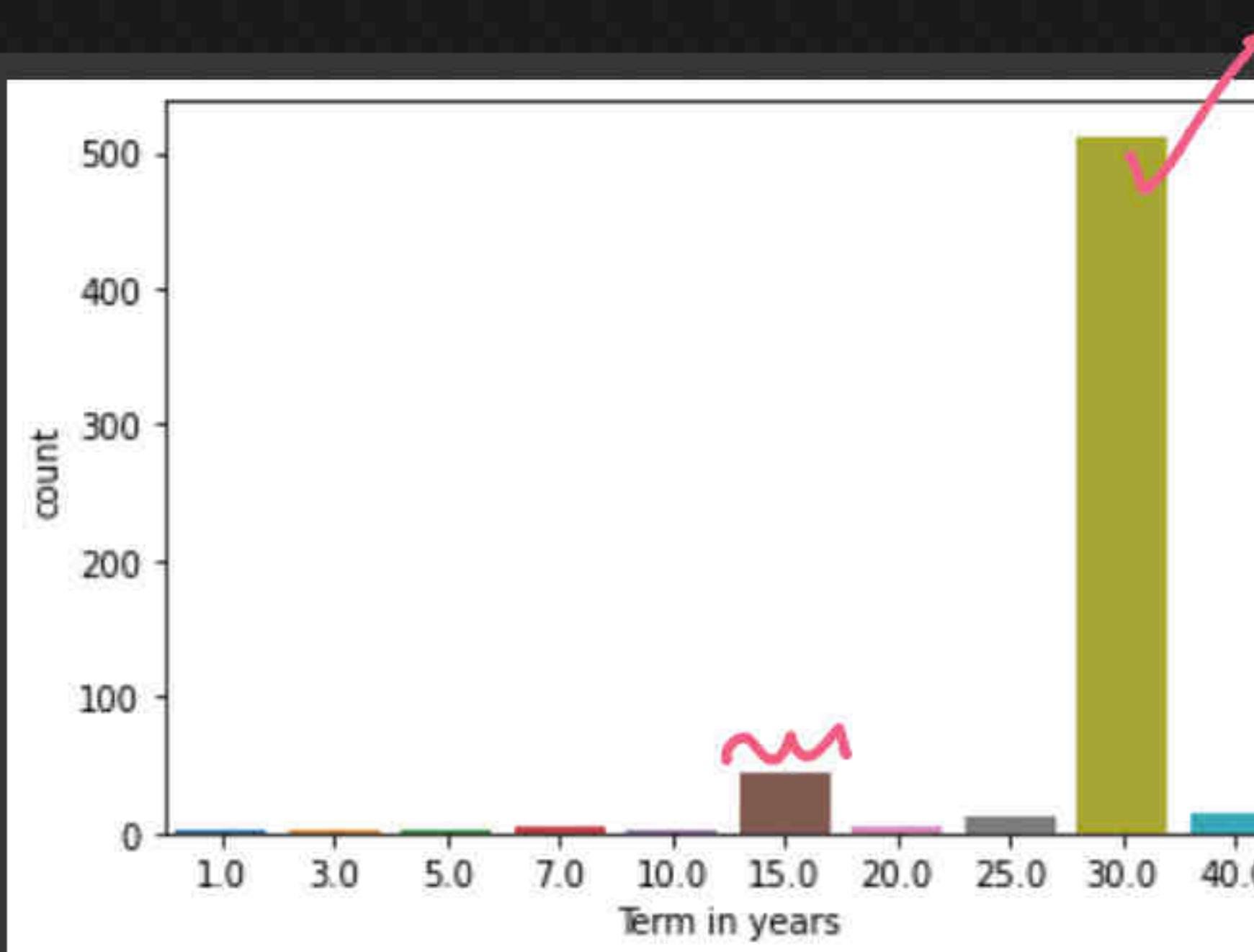
colab.research.google.com/drive/1\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=f2aROZfGMz9t

+ Code + Text

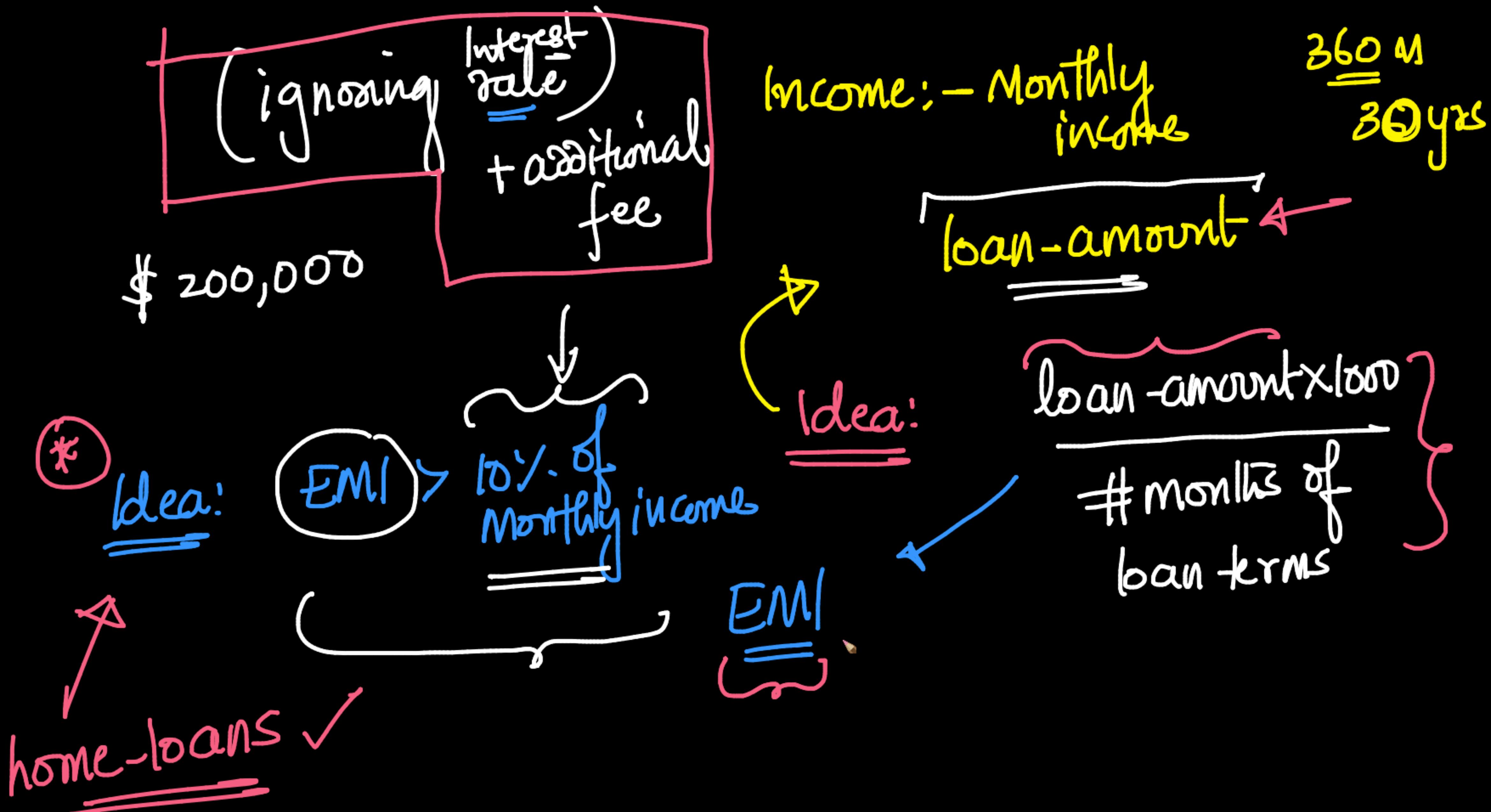
✓ RAM Disk



```
[234] sns.countplot(x='Loan_Amount_Term', data=data)
      plt.xlabel("Term in years")
      plt.show()
# Observation: We can clearly see that more than 90% of the loans were applied for 30 years.
```



[235] ### Distribution of "LoanAmount" variable :



EDA\_FE.ipynb - Colaboratory +

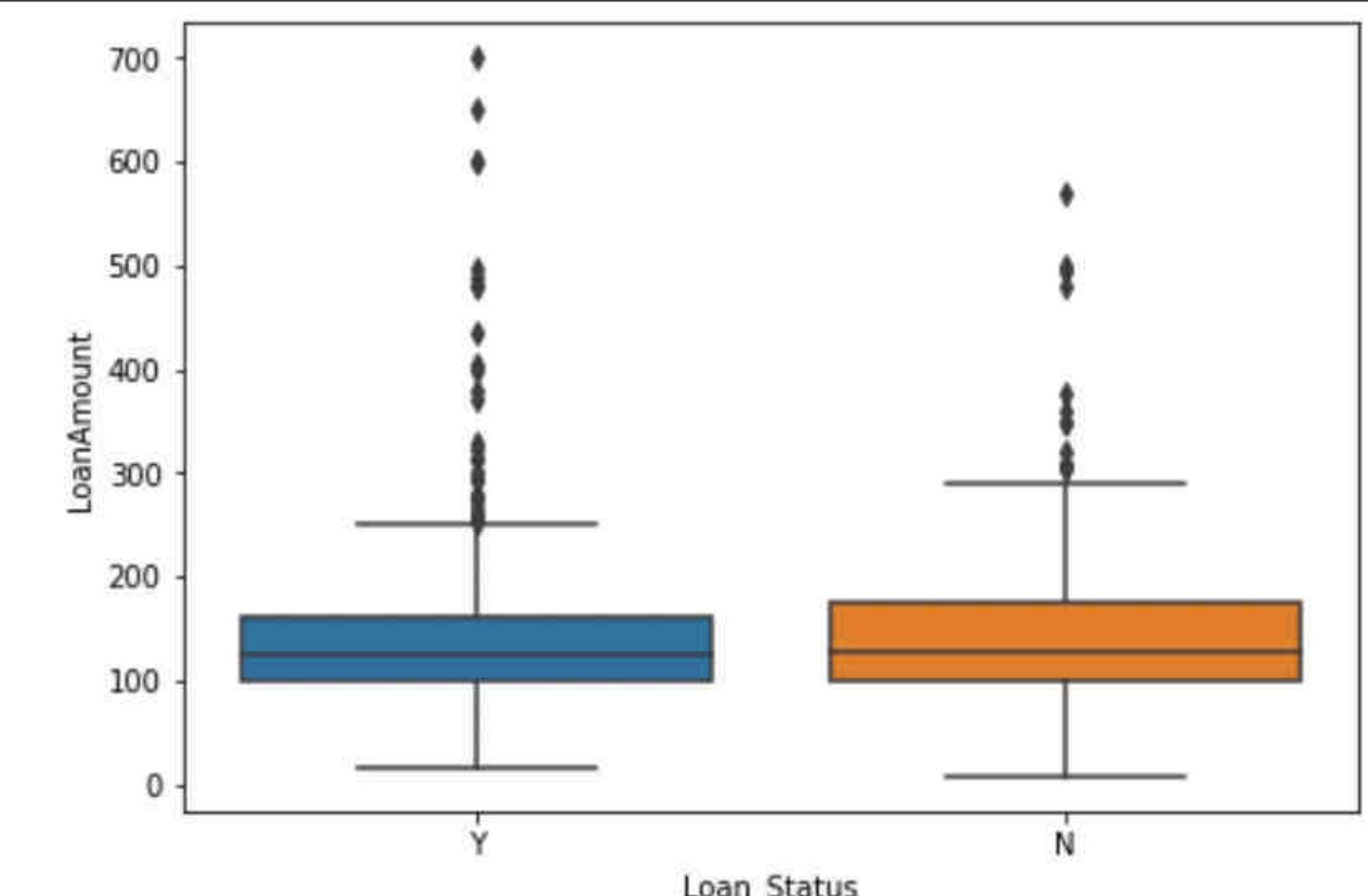
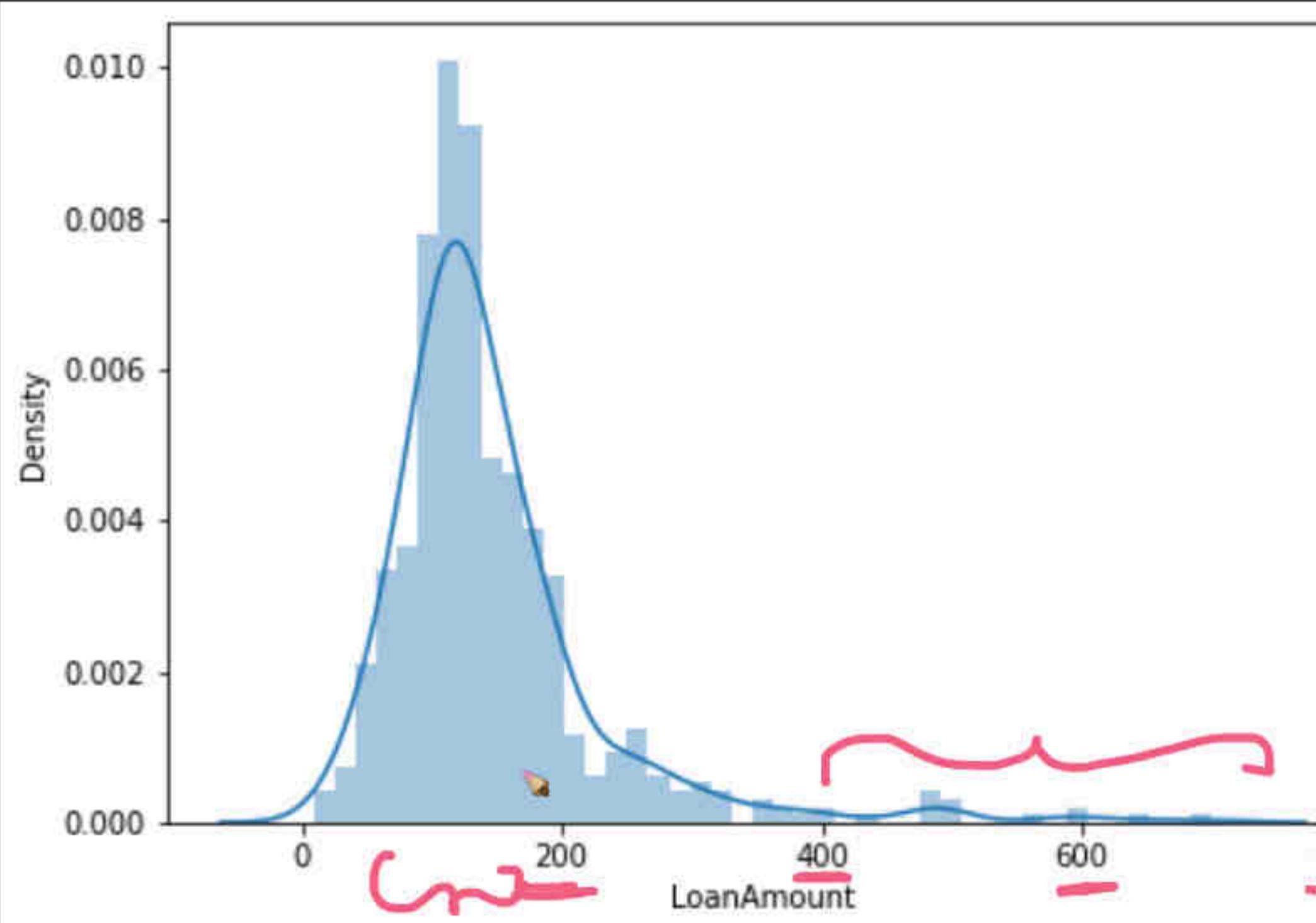
colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=K4uMgnkQNE39

RAM Disk

+ Code + Text

```
plt.subplot(122)
sns.boxplot(data=data, x='Loan_Status', y = 'LoanAmount')

plt.show()
```



[236] # Approximate calc: ignoring interest rates as we dont know that.

data['Loan\_Amount\_per\_year'] = data['LoanAmount']/data['Loan\_Amount\_Term']

EDA\_FE.ipynb - Colaboratory +

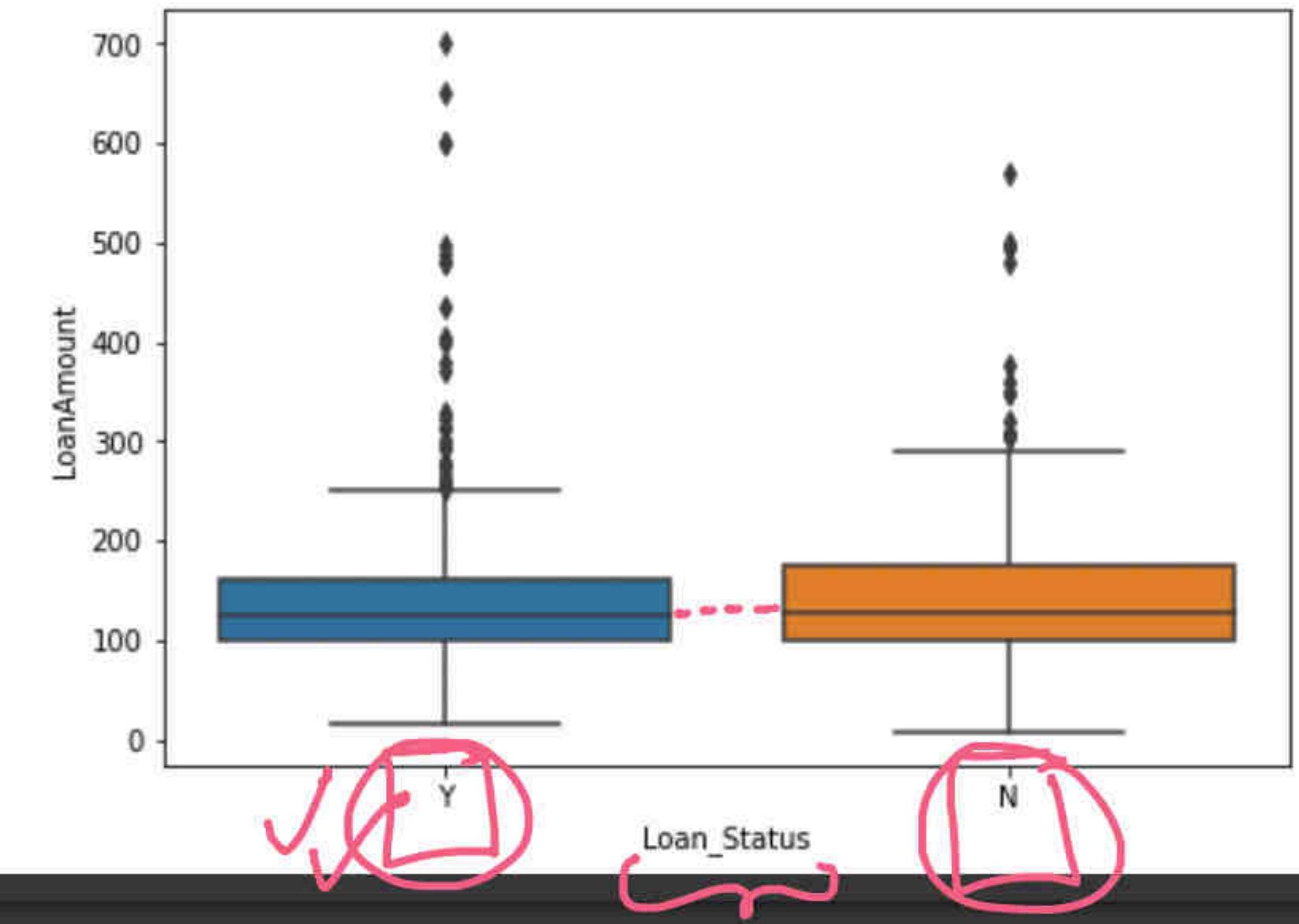
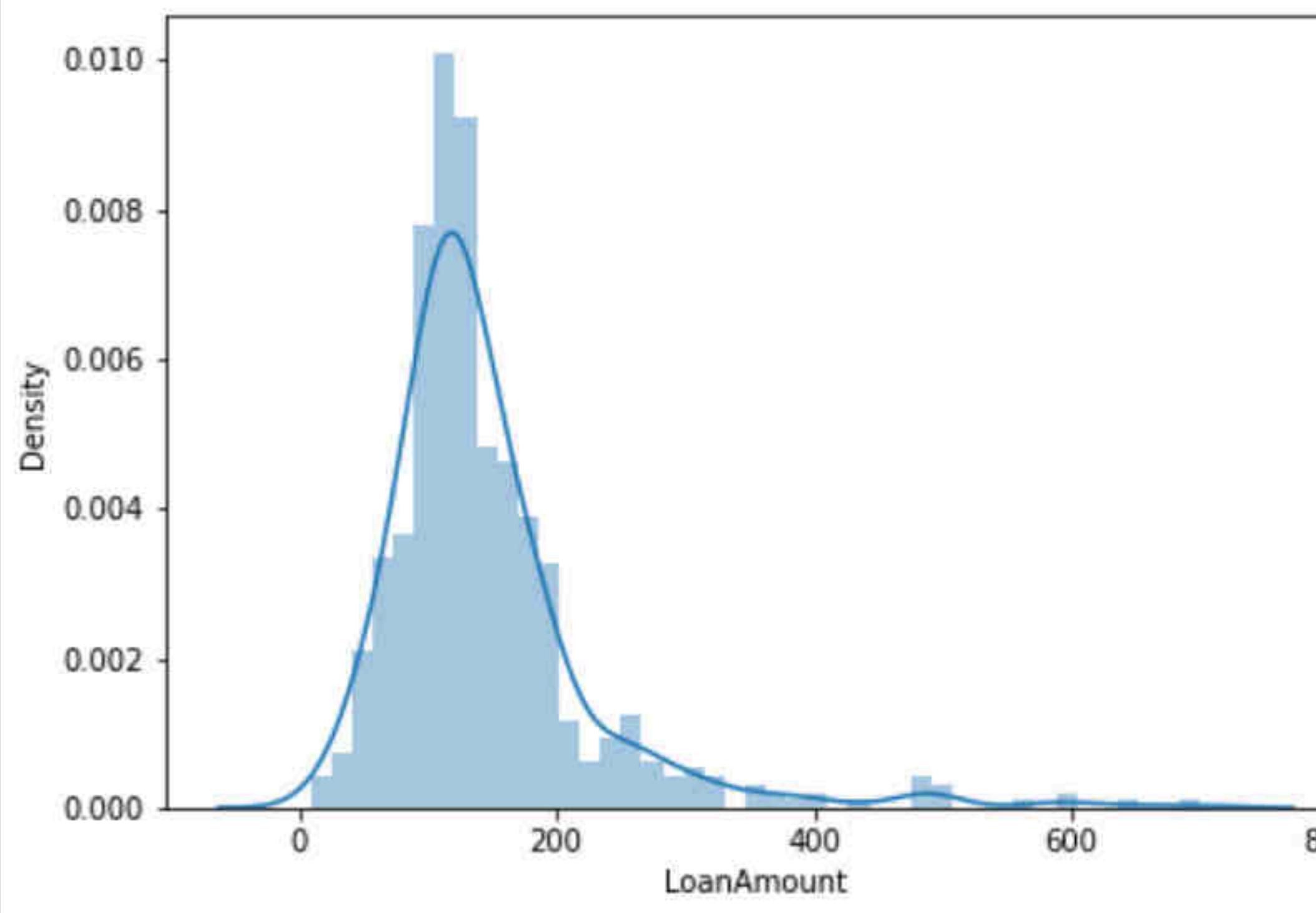
colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=K4uMgnkQNE39

+ Code + Text

```
plt.subplot(122)
sns.boxplot(data=data, x='Loan_Status', y = 'LoanAmount')

plt.show()
```

✓ ↗  
Loan-amount ✓

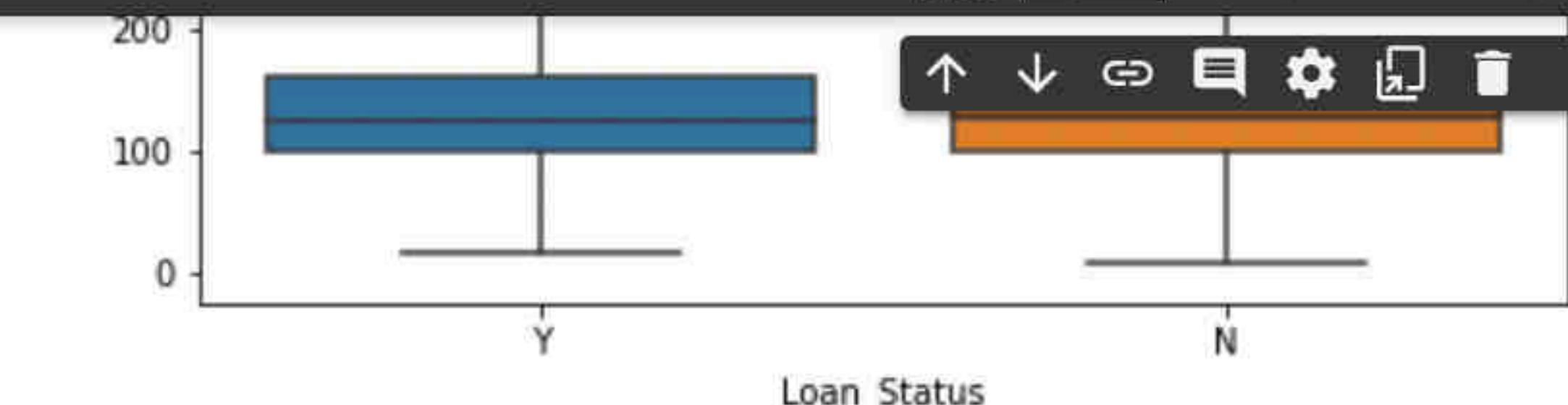
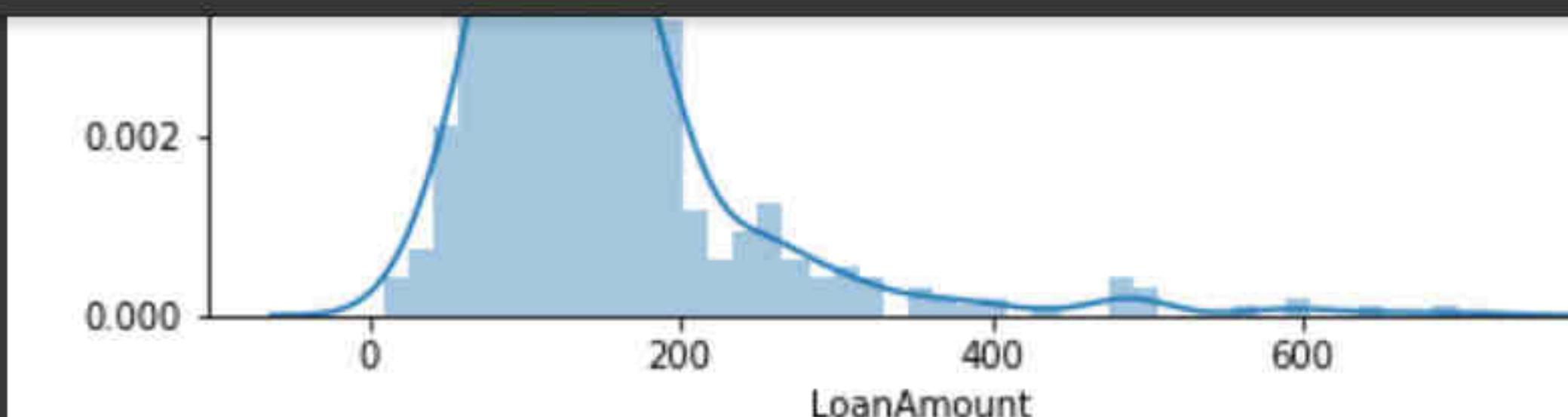


[236] # Approximate calc: ignoring interest rates as we dont know that.

```
data['Loan_Amount_per_year'] = data['LoanAmount']/data['Loan_Amount_Term']
```

EDA\_FE.ipynb - Colaboratory

+ Code + Tex



✓ [236] # Approximate calc: ignoring interest rates as we dont know them

```
data['Loan_Amount_per_year'] = data['LoanAmount']/data['Loan_Amount_Term']
```

لـ

```
plt.figure(figsize=(16,5)
plt.subplot(121)
sns.distplot(data['Loan_Amount_Term'])
```

```
plt.subplot(122)
sns.boxplot(data=data, x='Loan_Status', y = 'Loan_Amount_per_year')
```

```
plt.show()
```



EDA\_FE.ipynb - Colaboratory

+ Code

+ Text

RAM  
Disk

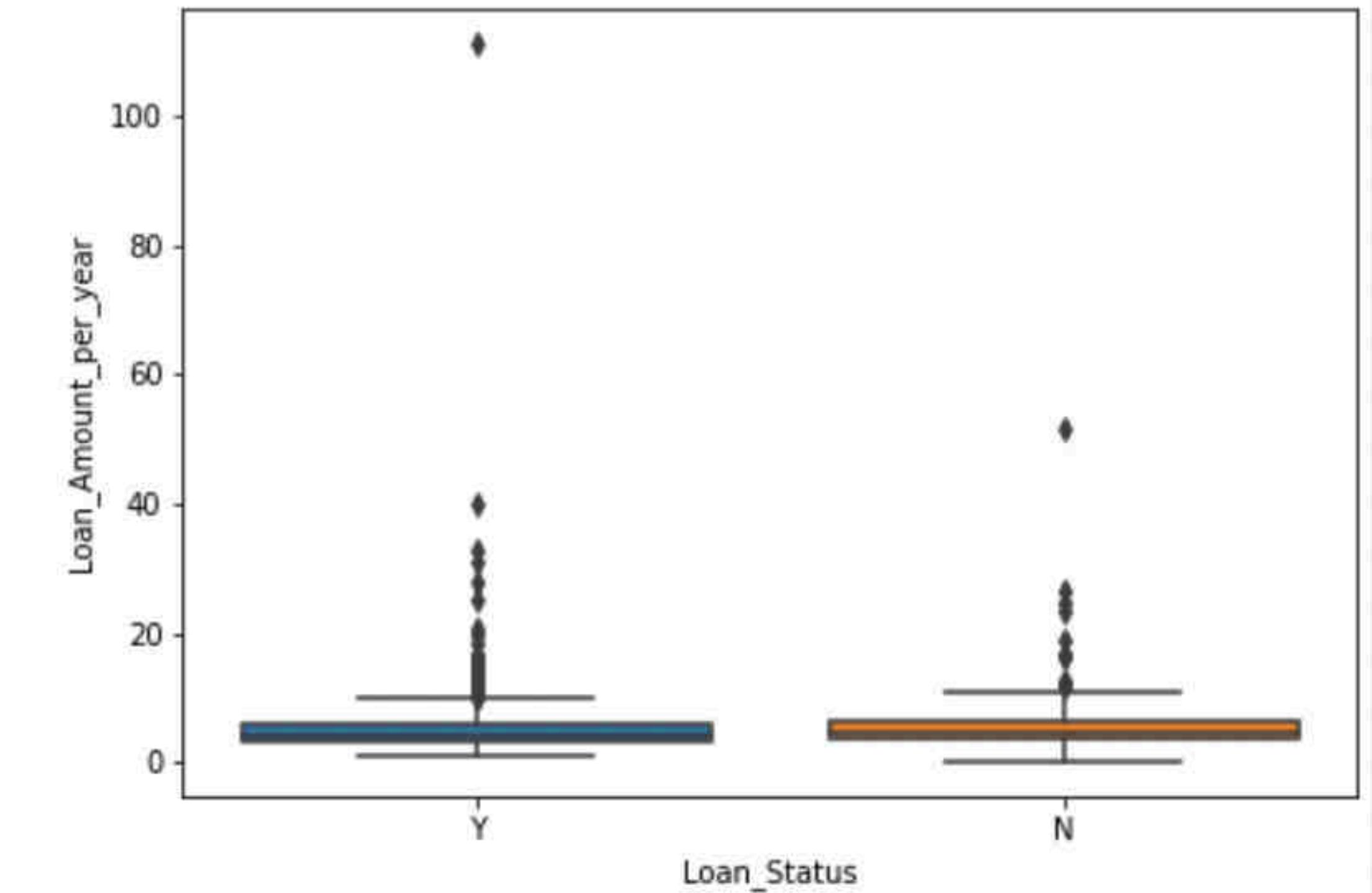
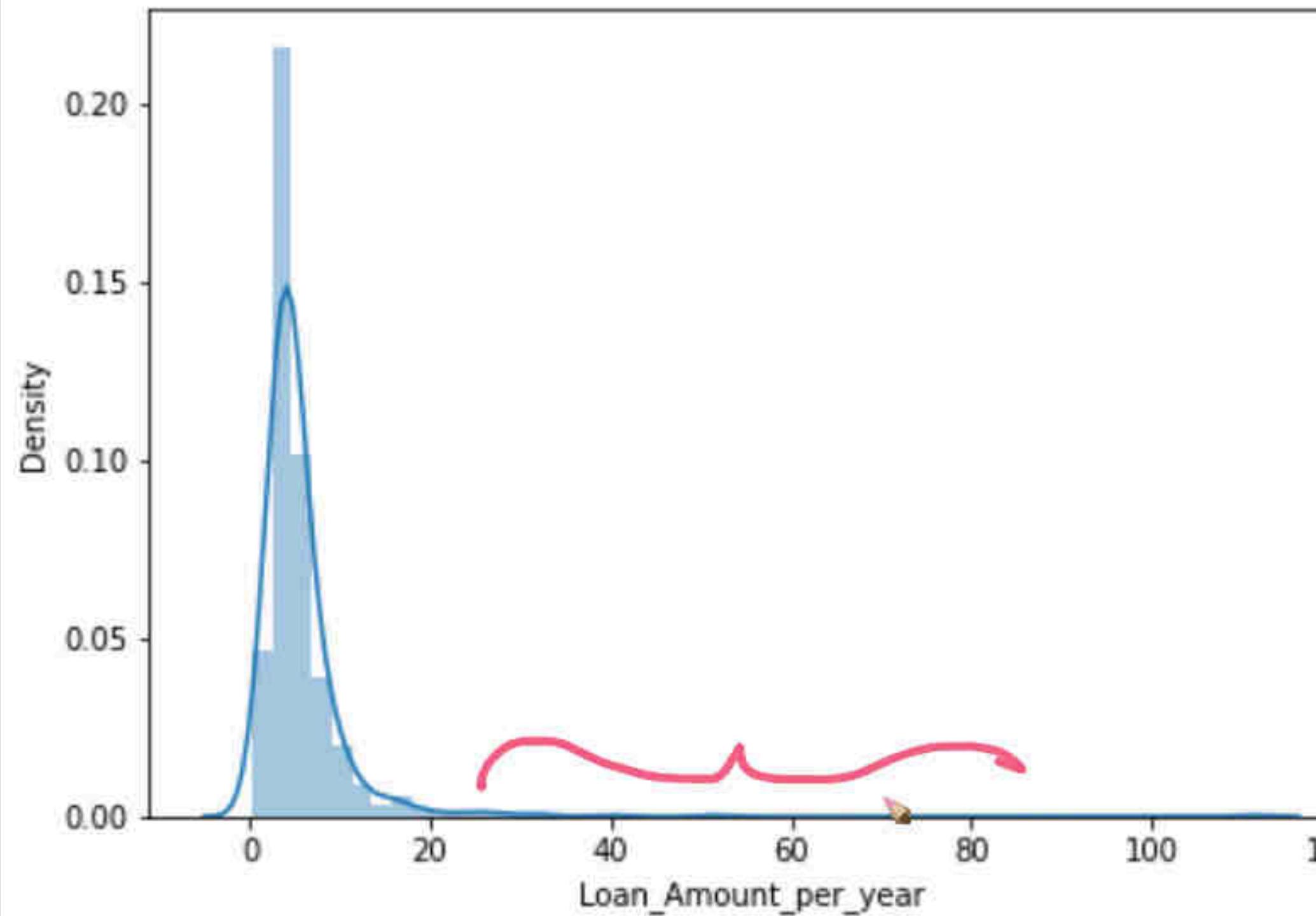
User Settings

Up Down Refresh Settings Print Delete More

```
plt.subplot(121)
sns.distplot(data['Loan_Amount_per_year']);

plt.subplot(122)
sns.boxplot(data=data, x='Loan_Status', y = 'Loan_Amount_per_year')

plt.show()
```

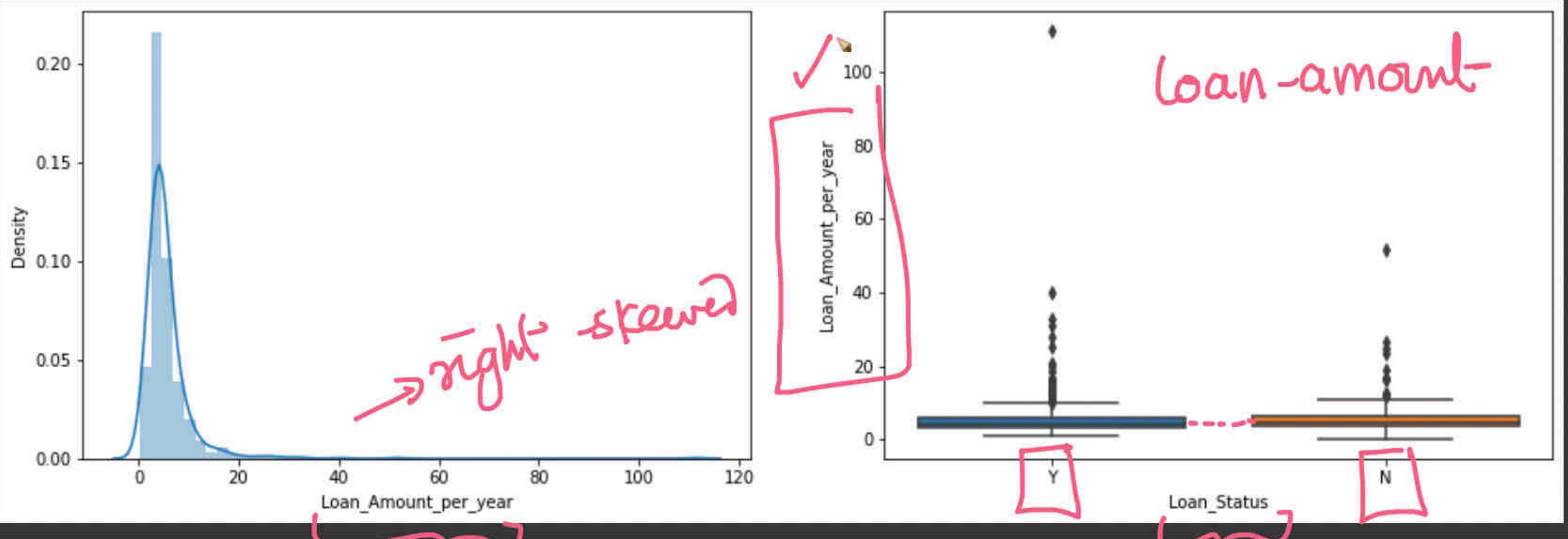


EDA\_FE.ipynb - Colaboratory +

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=GnRzeqOPNduE

+ Code + Text

```
sns.distplot(data['Loan_Status'], x='Loan_Status', y='Loan_Amount_per_year')  
plt.show()
```

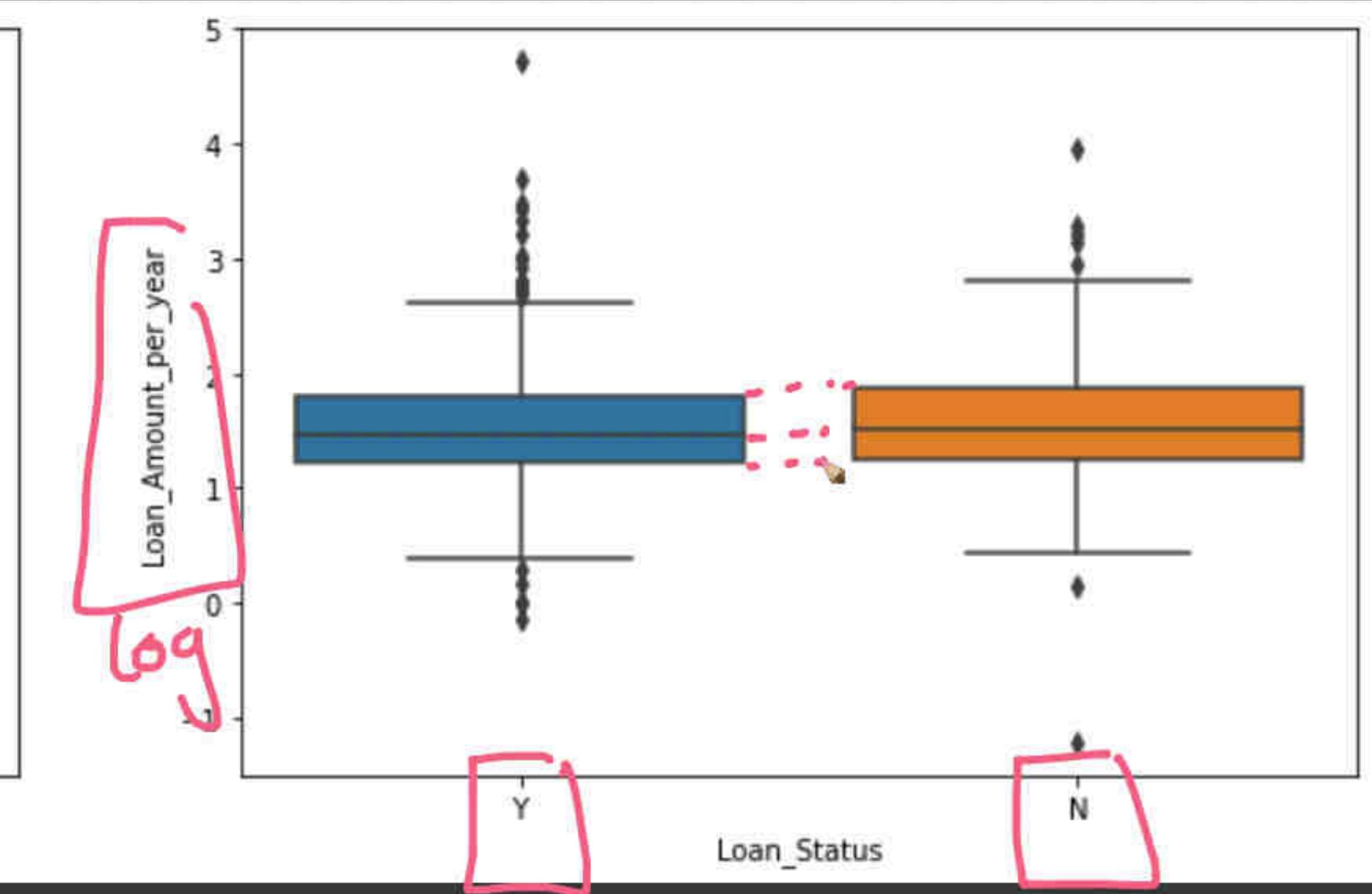
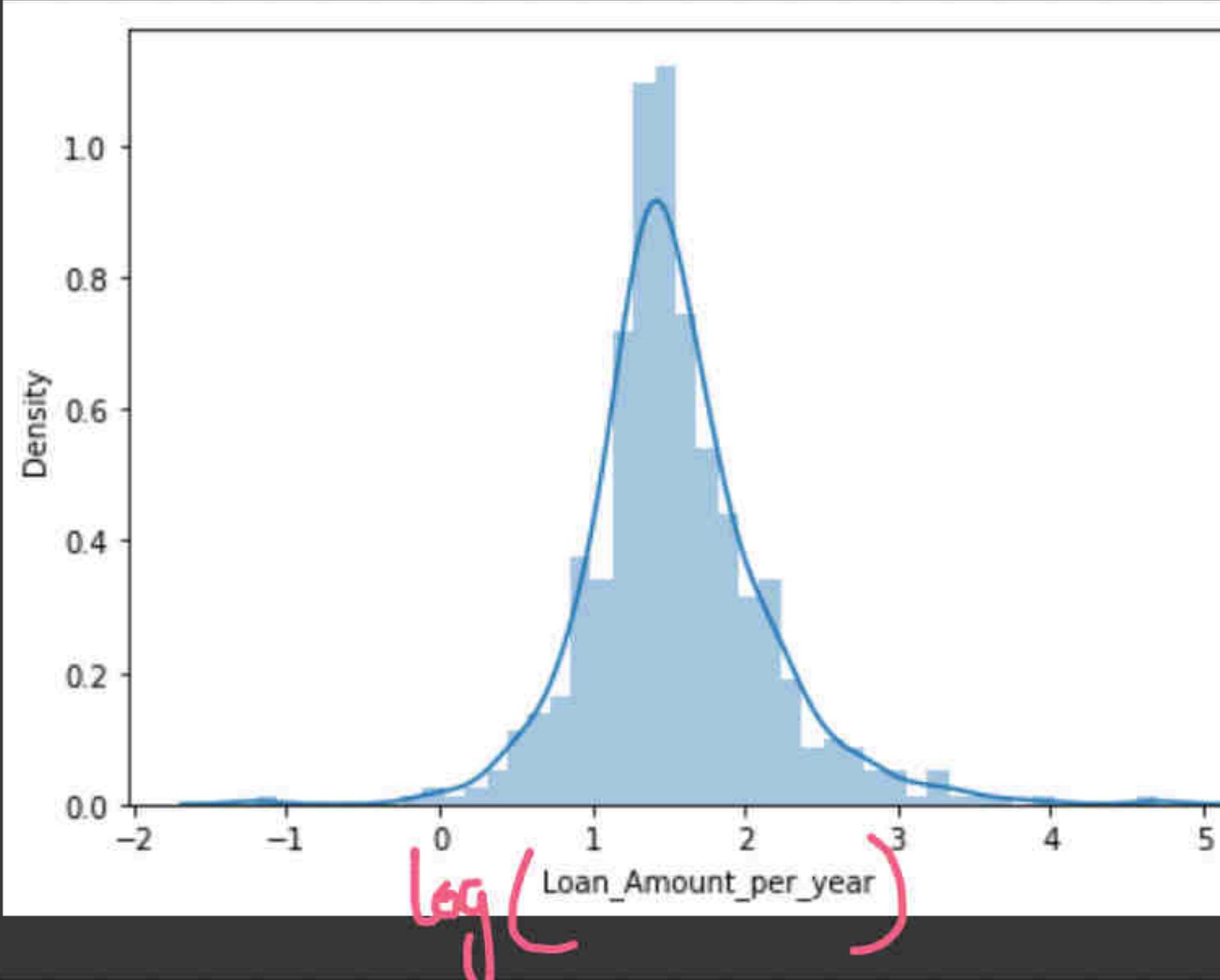


```
[238] # log transform  
plt.figure(figsize=(16,5))  
plt.subplot(121)  
log_loanAmount = np.log(data['Loan Amount per year'])
```

```
+ Code + Text
```

```
plt.subplot(122)
sns.boxplot(data=data, x='Loan_Status', y = log_loanAmount)

plt.show()
```



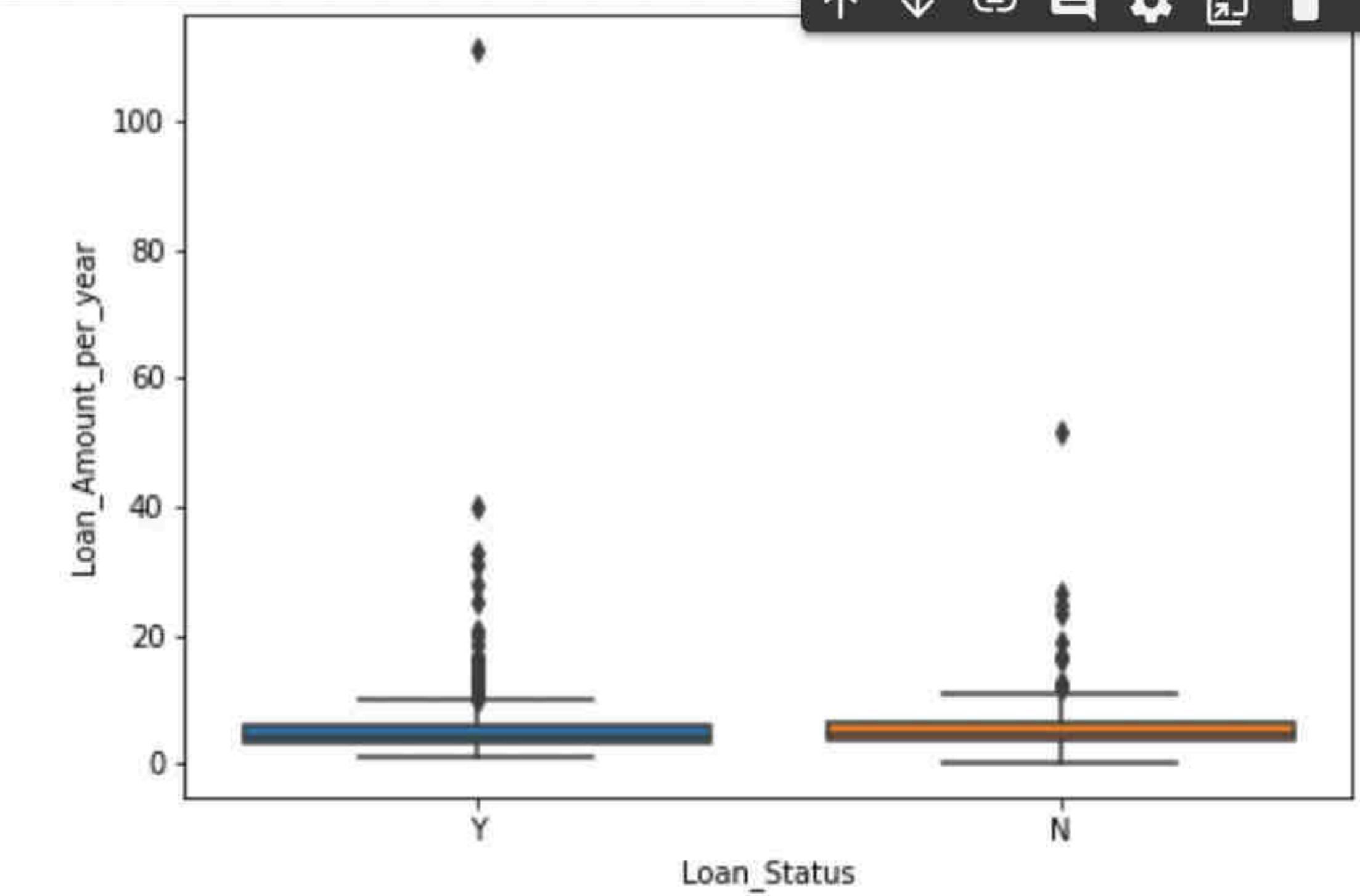
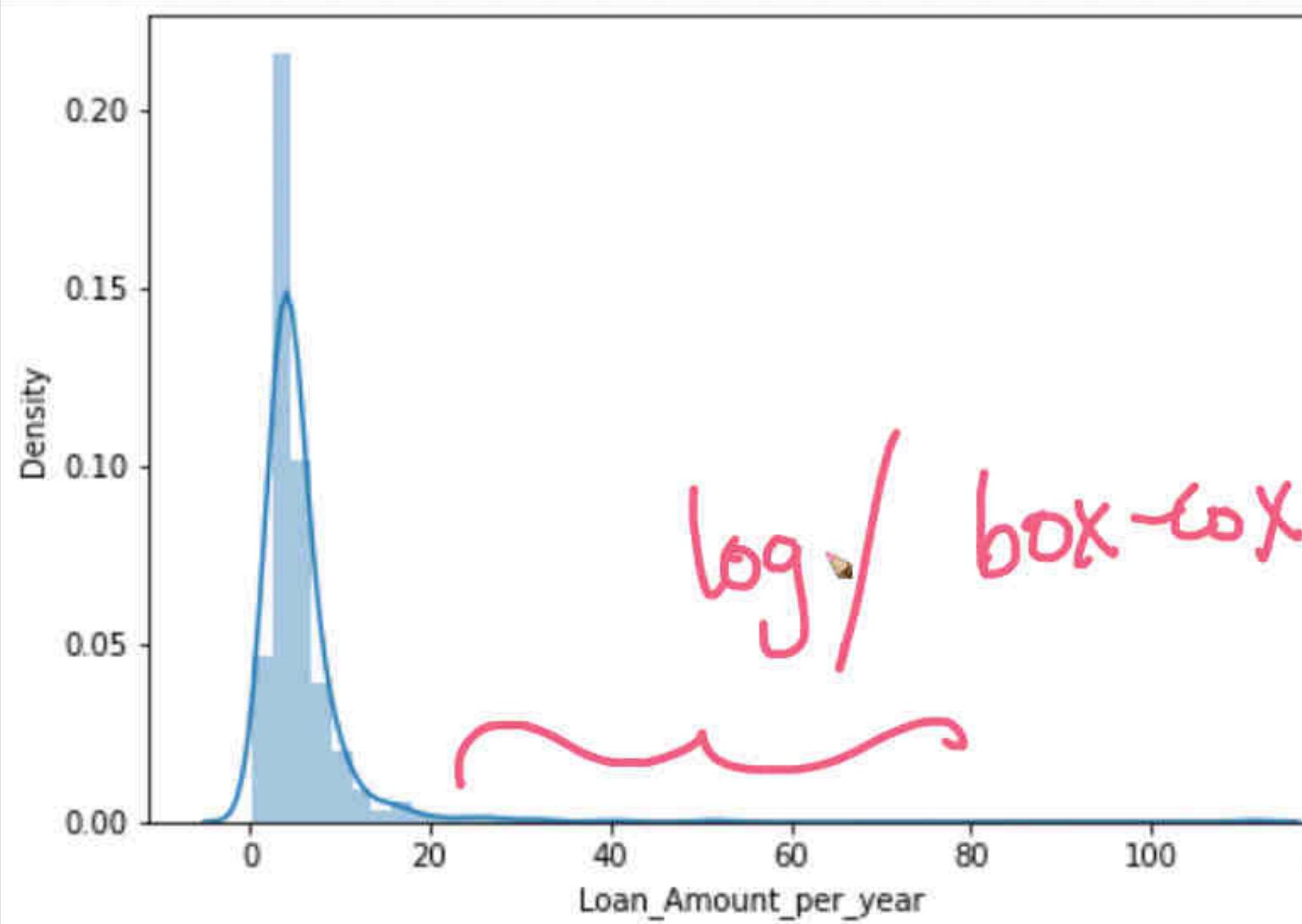
```
[239] # Feature : Calculate the EMI based on the Loan Amount Per year  
data[ 'EMI' ] = data[ 'Loan_Amount_per_year' ]*1000/12
```

EDA\_FE.ipynb - Colaboratory

colab.research.google.com/drive/1\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=GnRzeqOPNduE

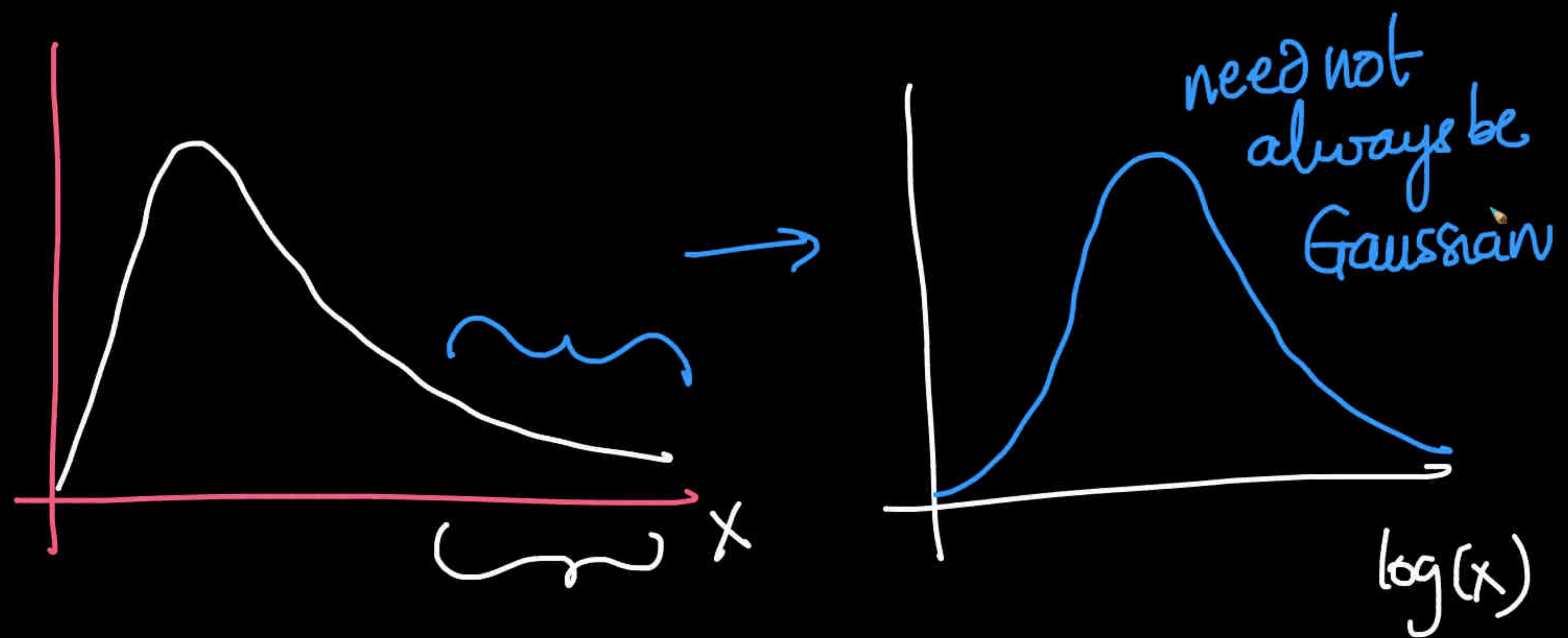
+ Code + Text

RAM Disk



```
[238] # log transform
plt.figure(figsize=(16,5))
plt.subplot(121)
log_loanAmount = np.log(data[ 'Loan_Amount_per_year'])
sns.distplot(log_loanAmount)

plt.subplot(122)
```



EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x +

google.com/search?q=plot+log(x)&rlz=1C5CHFA\_enIN958IN958&oq=plot+log(x)&aqs=chrome.0.69i59j07i30j0i30j0i5i7i30j0i390l2.3449j0j7&sourceid=chrome&ie=UTF-8

Google plot log(x)

All Images News Videos Shopping More Tools

About 4,86,00,00,000 results (0.49 seconds)

Graph for  $\log(x)$

x: 7.48736174 y: 0.874328816

More info

<https://www.rapidtables.com/math/algebra/logarithm.htm>

**logarithm graph | graph of  $\log(x)$  - RapidTables.com**

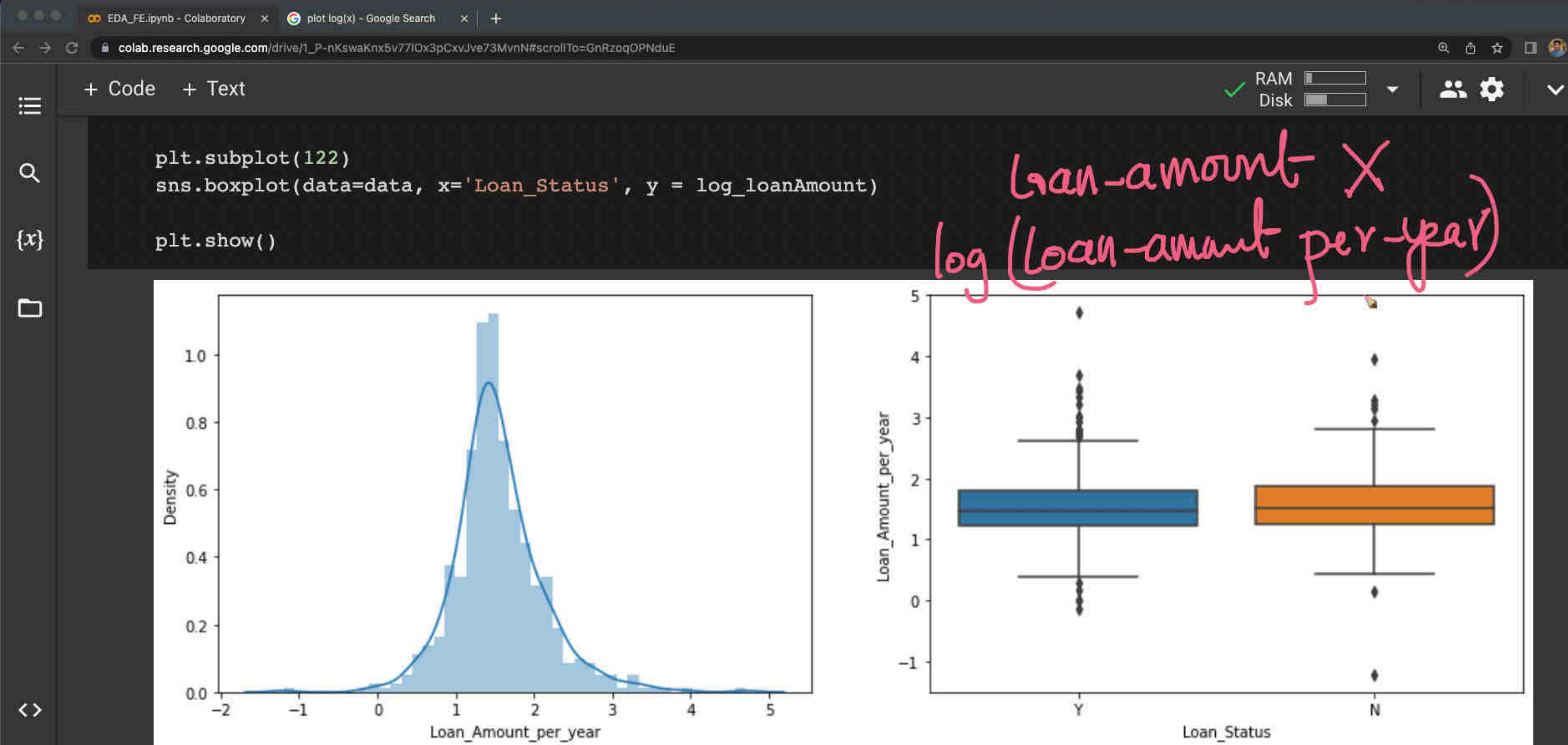
Graph of  $\log(x)$ ,  $\log(x)$  function graph. Logarithm graph.  $y = f(x) = \log_{10}(x)$ .  $\log(x)$  graph properties.  $\log(x)$  is defined for positive values of  $x$ .  $\log(x)$  ...

Videos :

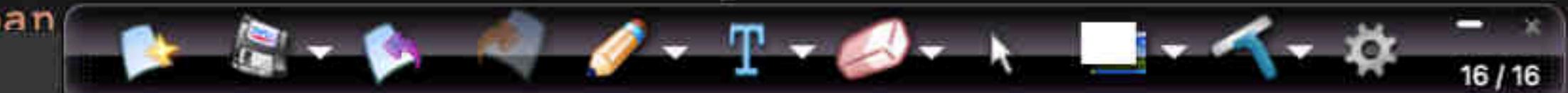
Draw the graph of ' $y=-\log(x)$ ' when the graph of ' $y=\log(x)$ ' is ...

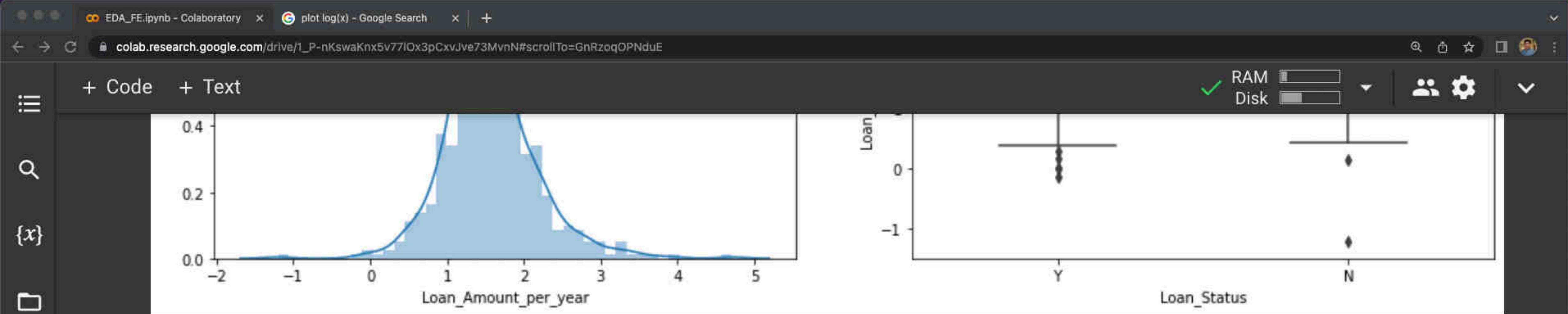
YouTube · DoubtNut 1:55 31-May-2020

GRAPH OF LOG(X) | GR



[ 239 ] # Feature : Calculate the EMI based on the Loan Amount Per year.  
data['EMT'] = data['Loan\_Amount\_per\_year'] \* (data['Interest\_Rate']/100) / 12





[239] # Feature : Calculate the EMI based on the Loan Amount Per year.

data['EMI'] = data['Loan\_Amount\_per\_year']\*1000/12

200,000

[240] #Feature : Able\_to\_pay\_EMI

data['Able\_to\_pay\_EMI'] = (data['TotalIncome']\*0.1 > data['EMI']).astype('int')

apply

[241] sns.countplot(x='Able\_to\_pay\_EMI', data = data, hue = 'Loan\_Status')

#Observation:

##There is 50% chance that you may get the loan approved if you cannot pay the EMI.

##But there, is a 72% chance that you may get the loan approved if you can pay the EMI.

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f3e2eadc950>



EDA\_FE.ipynb - Colaboratory

plot log(x) - Google Search

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=6x0wR3DnZ0ck

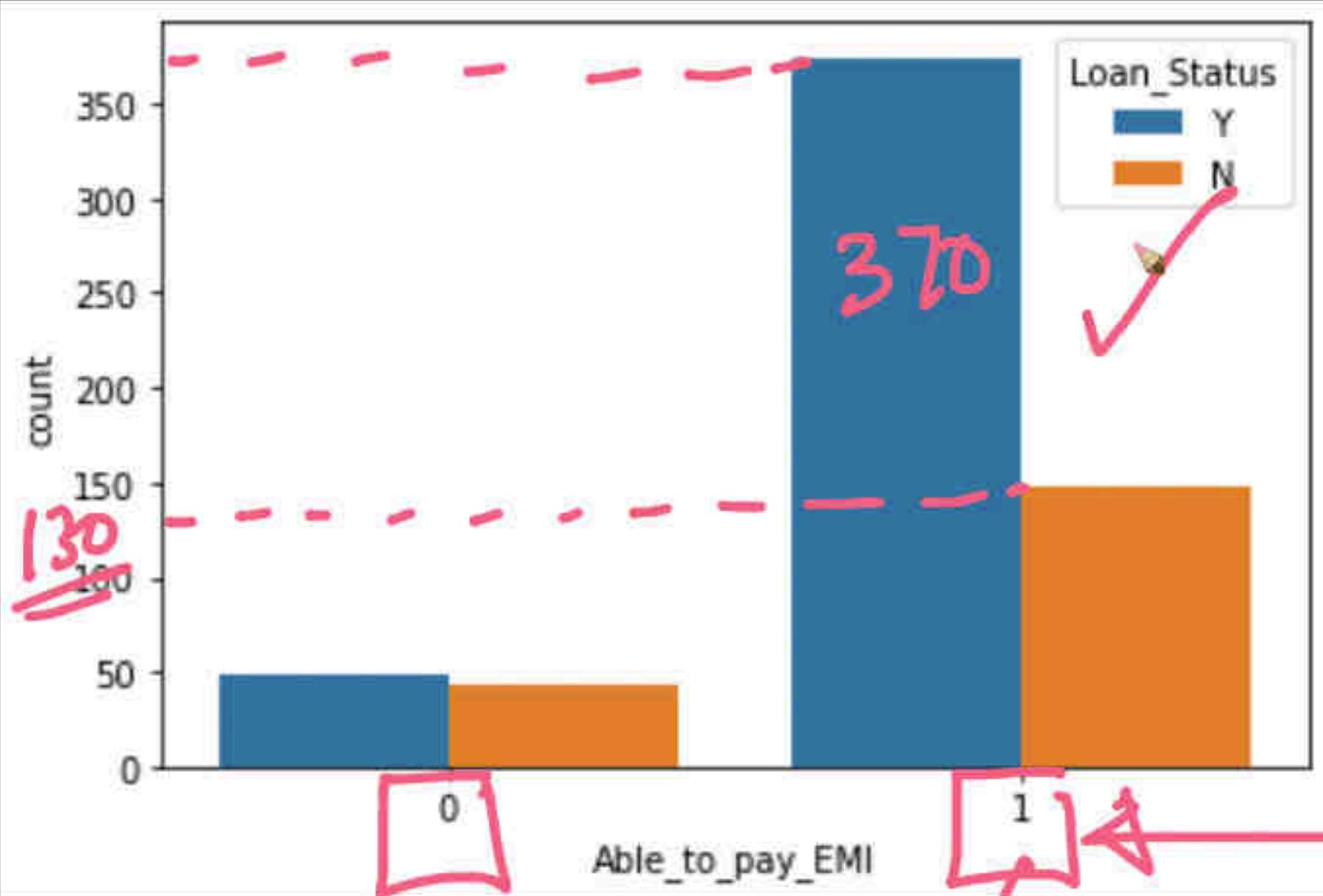
+ Code + Text

RAM Disk



```
sns.countplot(x='Able_to_pay_EMI', data = data, hue = 'Loan_Status')  
#Observation:  
###There is 50% chance that you may get the loan approved if you cannot pay the EMI.  
###But there, is a 72% chance that you may get the loan approved if you can pay the EMI.
```

&lt;matplotlib.axes.\_subplots.AxesSubplot at 0x7f3e2eadc950&gt;



$$\approx \frac{370}{500} =$$

Can-pay-EMI

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x

plot  $\log(x)$  - Google Search

→ G colab.research.google.com/drive/1\_P-nKswaKnx5y7zIOx3pCxyJye73MvpN#scrollTo=6x0wR3DnZQck

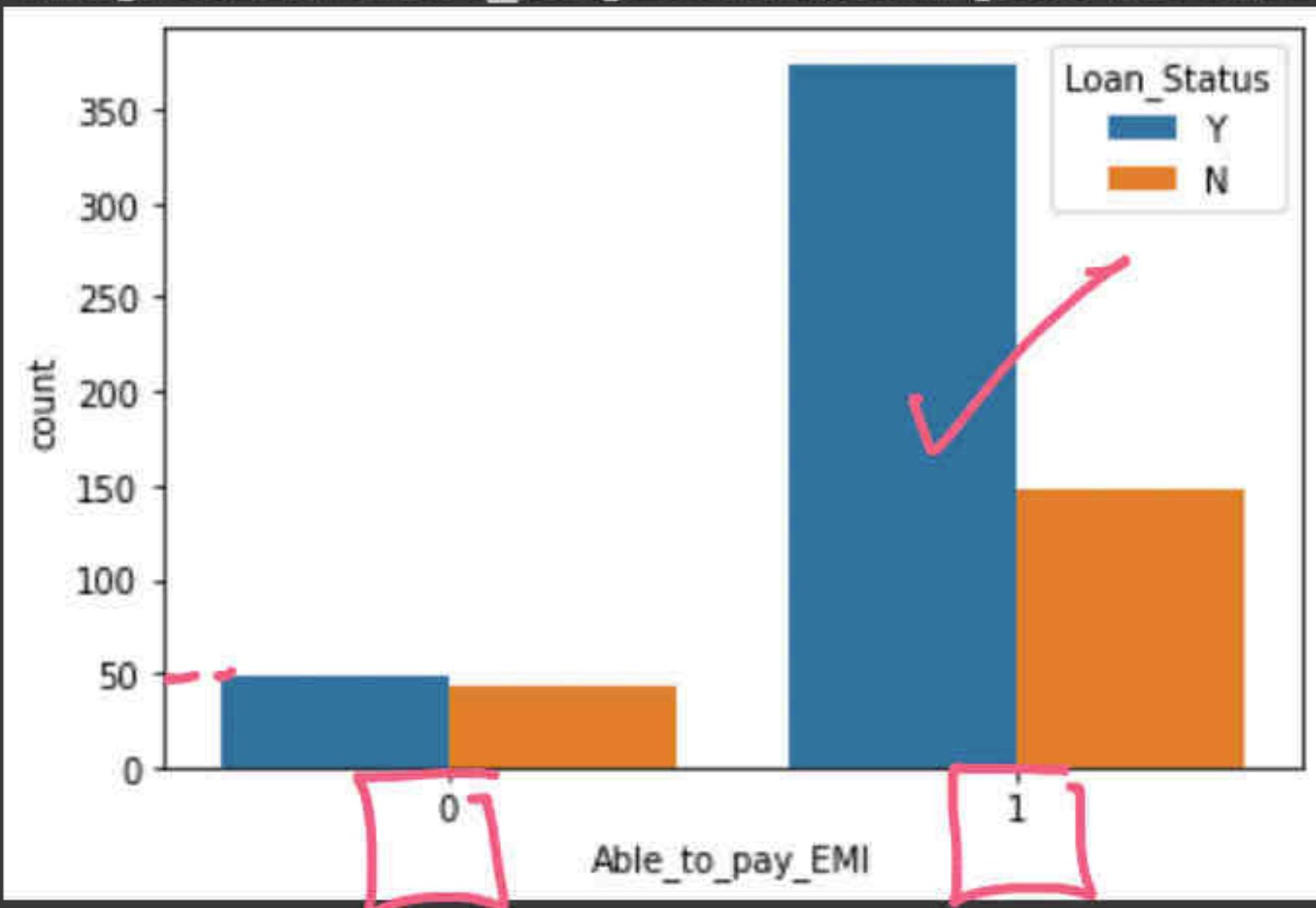
+ Code + Text

✓ RAM Disk



```
sns.countplot(x='Able_to_pay_EMI', data = data, hue = 'Loan_Status')  
#Observation:  
###There is 50% chance that you may get the loan approved if you cannot pay the EMI.  
###But there, is a 72% chance that you may get the loan approved if you can pay the EMI.
```

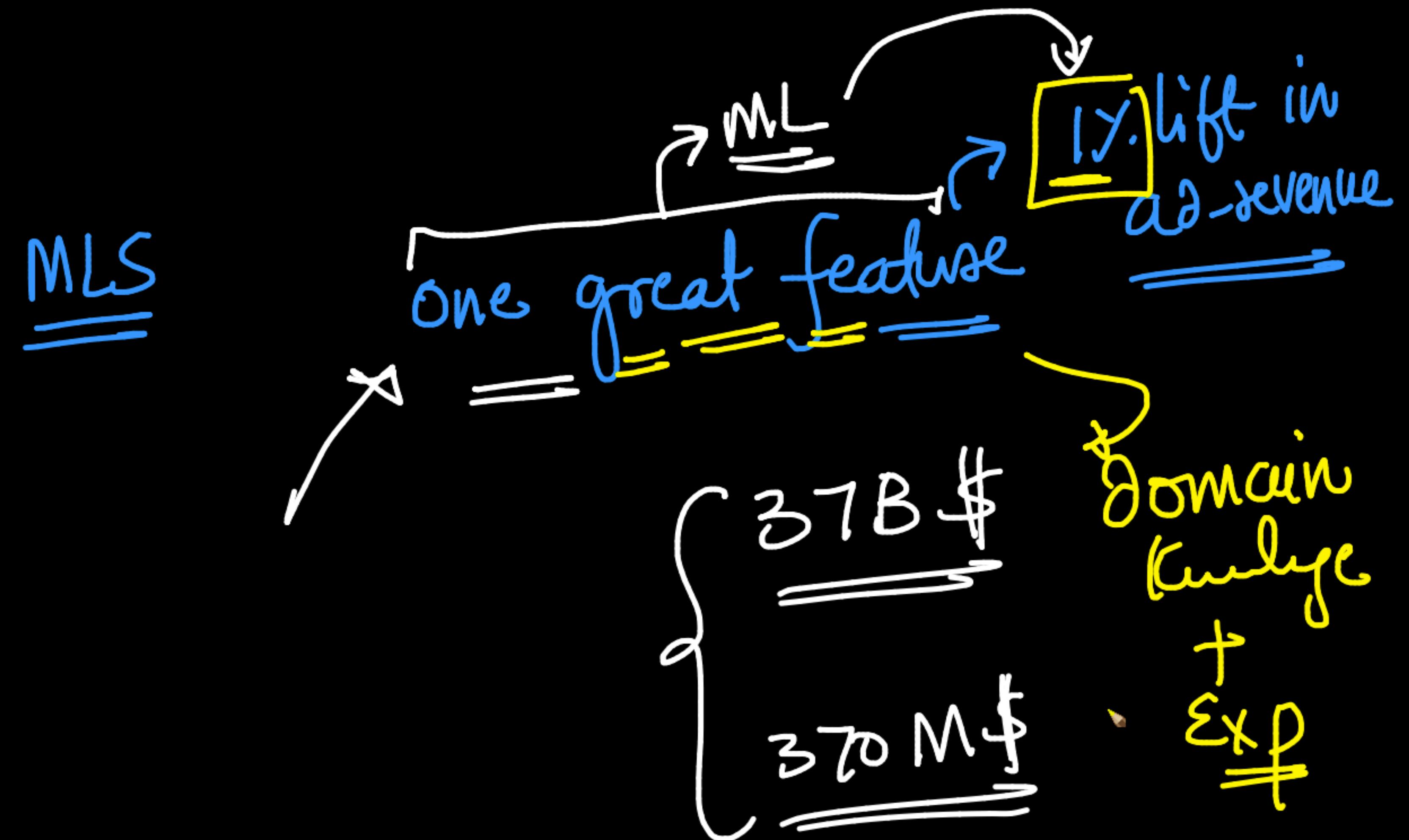
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3e2eadc950>
```



$$P(\text{loan status} = 1 \mid \text{able to pay EMI} = 1)$$

>  $P(\text{loan status} = \text{able to pay time})$   
28%  

---



EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=0eGVXudiZgMy

+ Code + Text RAM Disk

[ ] ↳ 11 cells hidden

{x} ▾ Dependents and Loan approval

↑ ↓ ⌂ ⚙ 📈 📉 :

✓ [242] data['Dependents'].value\_counts()

Dependents	Count
0	345 ✓
1	102
2	101
3+	51

Name: Dependents, dtype: int64

3+ → 3

✓ [243] data['Dependents'].replace('3+', 3, inplace=True)

✓ [244] data['Dependents'] = data['Dependents'].astype('float')

✓ [245] data.dtypes

21/21

EDA\_FE.ipynb - Colaboratory x Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=0eGVXudiZgMy

+ Code + Text RAM Disk

[ ] ↳ 11 cells hidden

{x} ▾ Dependents and Loan approval

↑ ↓ ⌂ ⚙ 📈 📉 :

✓ [242] data['Dependents'].value\_counts()

```
0      345
1      102
2      101
3+     51
Name: Dependents, dtype: int64
```

✓ [243] data['Dependents'].replace('3+', 3, inplace=True)

✓ [244] data['Dependents'] = data['Dependents'].astype('float')

✓ [245] data.dtypes

22/22

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x

[https://colab.research.google.com/drive/1\\_P-nKswaKnx5y77IQx3pCxyJye73MvpN#scrollTo=0eGVXudIZgM](https://colab.research.google.com/drive/1_P-nKswaKnx5y77IQx3pCxyJye73MvpN#scrollTo=0eGVXudIZgM)

+ Code + Text

RAM Disk

[ 1 ] ↳ 11 cells hidden

## {x} ▾ Dependents and Loan approval

```
▶ data[ 'Dependents' ].value_counts()
```

```
0      345  
1      102  
2      101  
3+      51  
Name: Dependents, dtype: int64
```

[243] data[ 'Dependents' ].replace(' 3+', 3, inplace=True)

```
[244] data['Dependents'] = data['Dependents'].astype('float')
```

✓ [245] data.dtypes



EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=0eGVXudiZgMy

+ Code + Text

Able\_to\_pay\_EMI int64  
dtype: object

{x} [246] sns.countplot(data =data, x = 'Dependents', hue = 'Loan\_Status')

#Observations:

## No Dependents and 2 dependents will helps you get loan easily.

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f3e2ecd2790>

Dependents	Loan_Status Y	Loan_Status N
0.0	~220	~110
1.0	~65	~35
2.0	~75	~25
3.0	~35	~20

RAM Disk

24/24

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=0eGVXudiZgMy

+ Code + Text

Able\_to\_pay\_EMI int64  
dtype: object

{x} [246] sns.countplot(data = data, x = 'Dependents', hue = 'Loan\_Status')

#Observations:

## No Dependents and 2 dependents will helps you get loan easily.

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f3e2ecd2790>

The chart displays the following approximate data:

Dependents	Loan_Status Y	Loan_Status N
0.0	220	110
1.0	70	40
2.0	80	25
3.0	40	20

EDA\_FE.ipynb - Colaboratory x Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=lpP2KevSZdLd

+ Code + Text RAM Disk

#Observations:

## No Dependents and 2 dependents will helps you get loan easily.

{x}

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f3e2ecd2790>

Dependents	Loan_Status	Count
0.0	Y	~100
0.0	N	~100
1.0	Y	~60
1.0	N	~30
2.0	Y	~70
2.0	N	~20
3.0	Y	~30
3.0	N	~20

Double-click (or enter) to edit

Credit Score vs Loan Approval

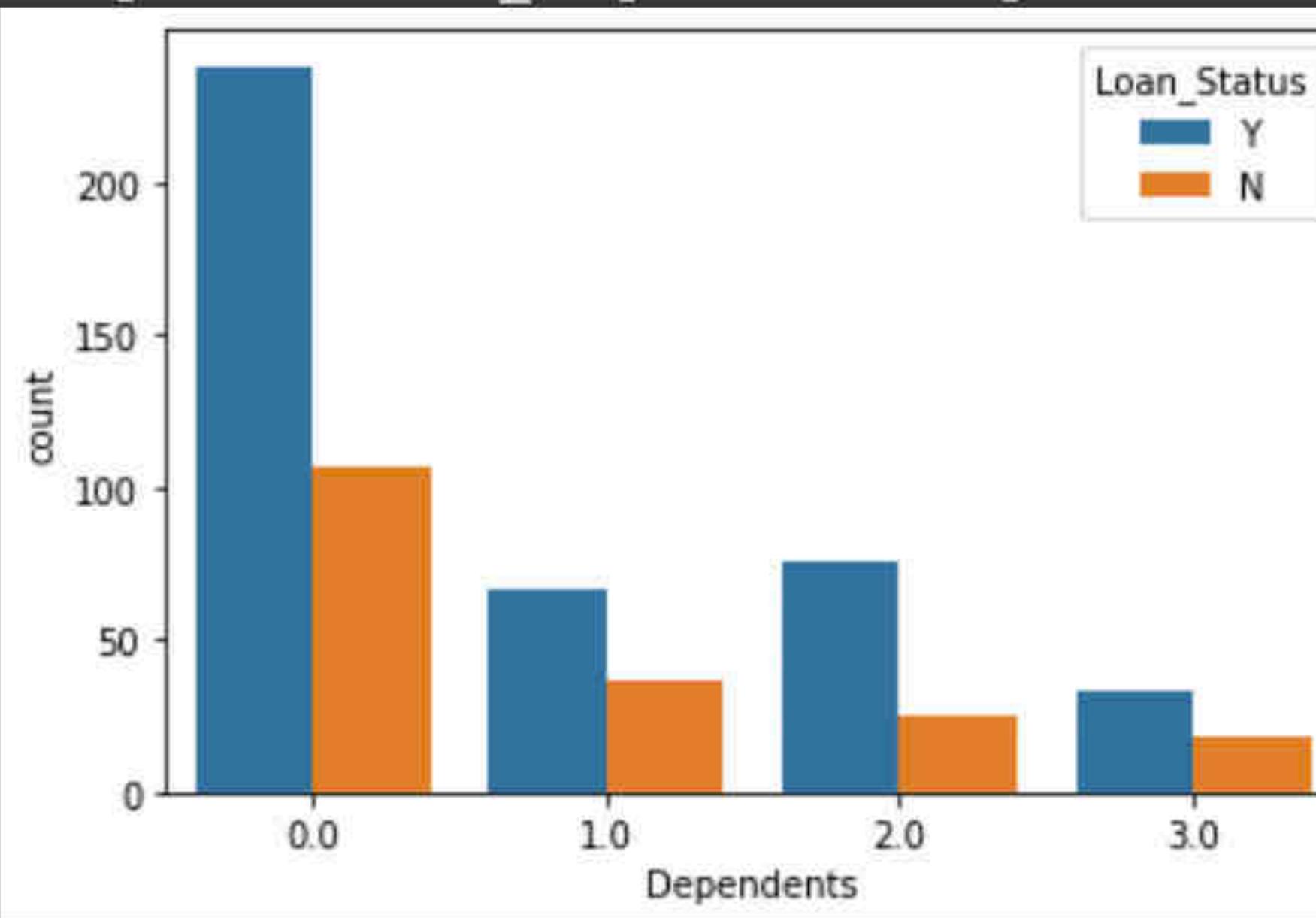
26/26

RAM Disk

#Observations:

## No Dependents and 2 dependents will helps you get loan easily.

&lt;matplotlib.axes.\_subplots.AxesSubplot at 0x7f3e2ecd2790&gt;



✓ t-test

Loan amount vs approval

mean - anil -

$M_1 - M_0$

$M_1 = M_0$

Double-click (or enter) to edit

Credit Score vs Loan Approval

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search

[colab.research.google.com/drive/1\\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=IpP2KevSz](https://colab.research.google.com/drive/1_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=IpP2KevSz)

+ Code + Text

→ 6 cells made

✓ RAM Disk

## ▼ Credit Score vs Loan Approval

```
[247] data['Credit History'].value_counts()
```

✓ 1.0 4

0.0

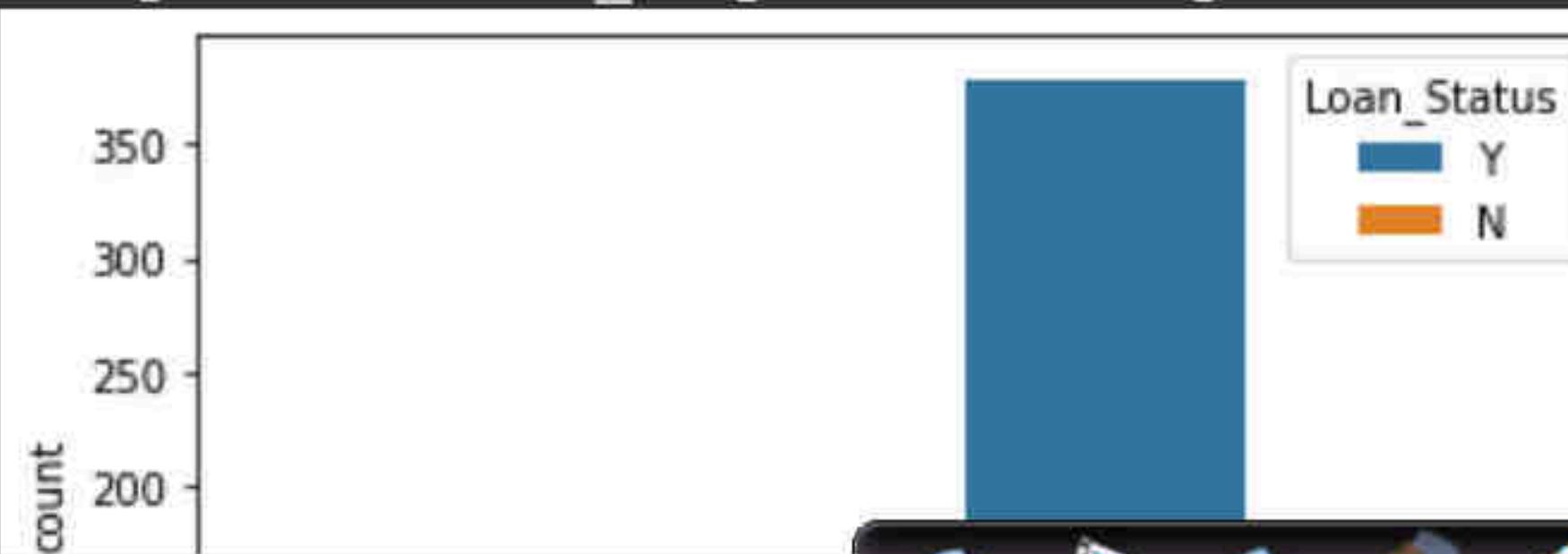
Name: Cr

Name: Credit\_History, dtype: int64

77...  
CIBIL\_Score → Y  
N → A

```
✓   sns.countplot(data =data, x = 'Credit_History', hue = 'Loan_Status')
    #Observation:
    ## We can clearly see that the approval rate is 80% if your credit
    ## Hence this is the most important question that can be considered
```

```
[1]: <matplotlib.axes.AxesSubplot at 0x7f3e2ee68c1>
```



EDA\_FE.ipynb - Colaboratory    X    plot log(x) - Google Search    X

[colab.research.google.com/drive/1\\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=IdP2KevSzD](https://colab.research.google.com/drive/1_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=IdP2KevSzD)

+ Code + Text

→ b cells nieder

RAM Disk



## ▼ Credit Score vs Loan Approval

100

```
[247] data['Credit History'].value_counts()
```

```
1.0    475  
0.0    89  
Name: Credit_History, dtype: int64
```

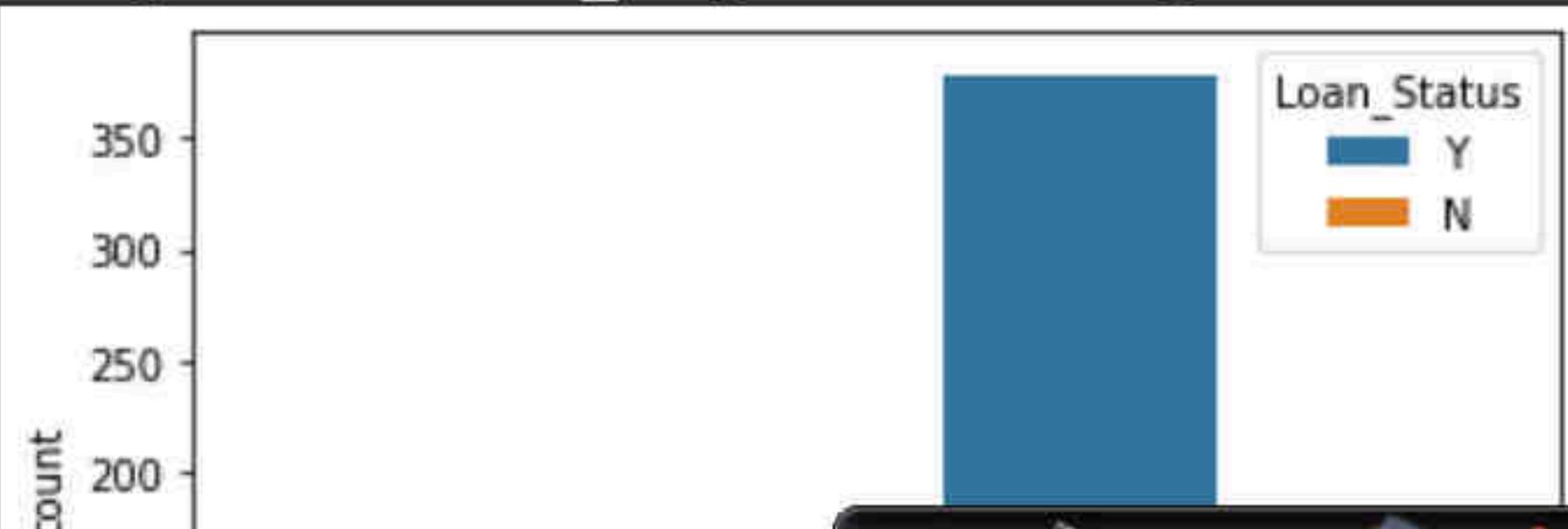
Name > Credit\_History, dtype: int64

```
✓ [248] sns.countplot(data = data, x = 'Credit_History', hue = 'Loan_Status')
```

## #Observation

## Hence this is the most important question that can be considered

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3e2ee68c10>
```



EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=lpP2KevSZdLd

+ Code + Text RAM Disk

1.0 475  
0.0 89  
Name: Credit\_History, dtype: int64

[248] sns.countplot(data = data, x = 'Credit\_History', hue = 'Loan\_Status')  
#Observation:  
## We can clearly see that the approval rate is 80% if your credit history is aligned with the guidelines.  
## Hence this is the most important question that can be considered.

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f3e2ee68c10>

Count

Credit\_History

Loan\_Status

Y

N

0.0

1.0

350  
300  
250  
200  
150  
100  
50  
0

 EDA\_FE.ipynb - Colaboratory  plot log(x) - Google Search

 plot log(x) - Google Search

colab.research.google.com/drive/1\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=FxqCiW7laE

+ Code + Text

✓ RAM Disk

[ ] ↳ 6 cells hidden



## {x} ▶ Credit Score vs Loan Approval

▶ ↳ 3 cells hidden

## ▶ Missing Values & Data Cleaning

[ ] ↳ 9 cells hidden

## ► Categorical to Numerical encoding

## Nominal vs Ordinal variables

## 1. One Hot Encoding

## 2. Label encoding

### 3. Target Encoding

Appropriate encoding depends on what our task is (and) what we do next

Task: features can help  
us determine  
loan-status }

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x + colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=FxqCjW7laETR

+ Code + Text

[ ] ↳ 6 cells hidden

RAM Disk

RAM Disk

↑ ↓ ↻ ☰ 📝 📐 📁 :

{x} ▶ Credit Score vs Loan Approval

▶ Missing Values & Data Cleaning

▶ Categorical to Numerical encoding

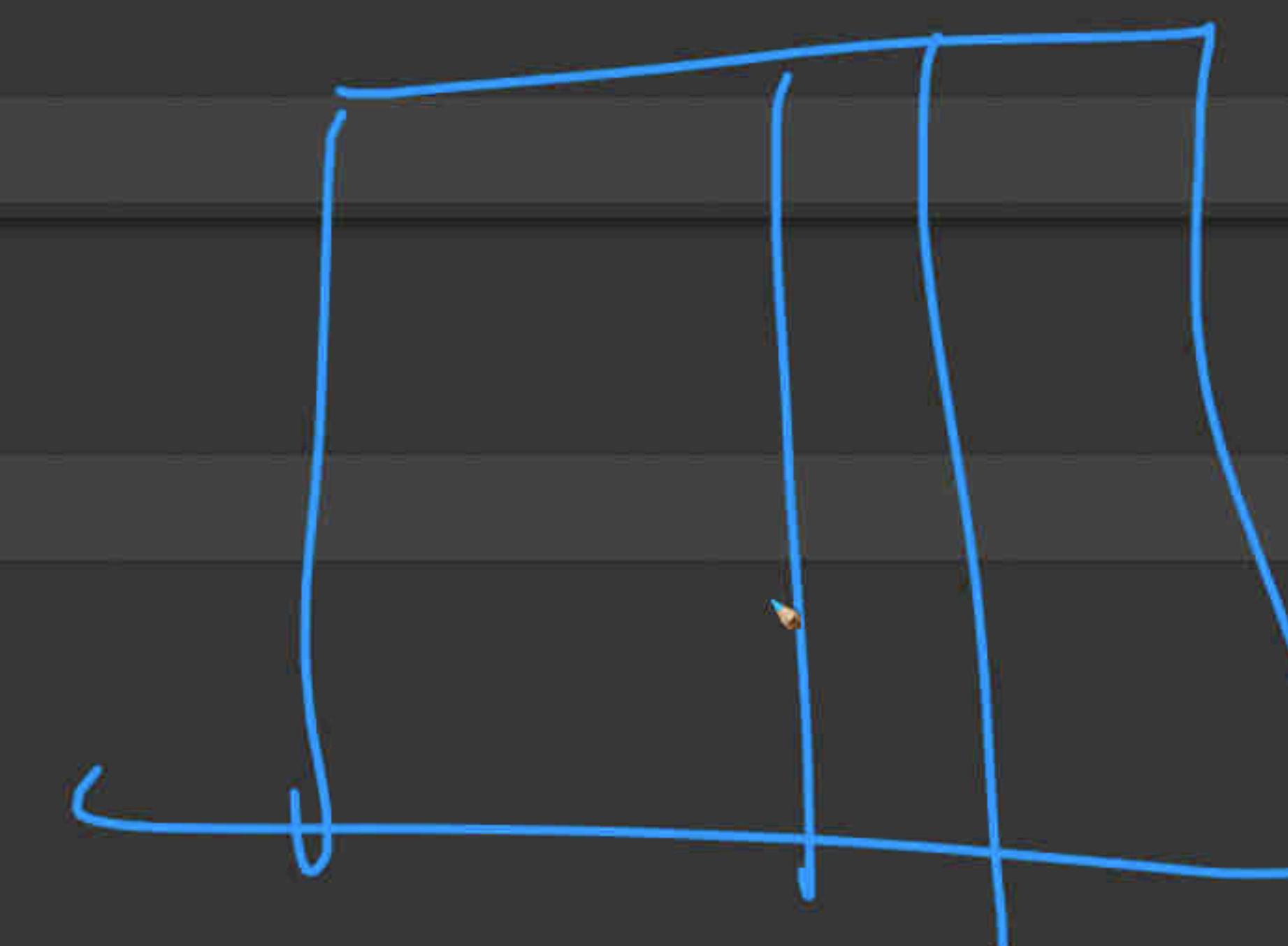
Nominal vs Ordinal variables

1. One Hot Encoding

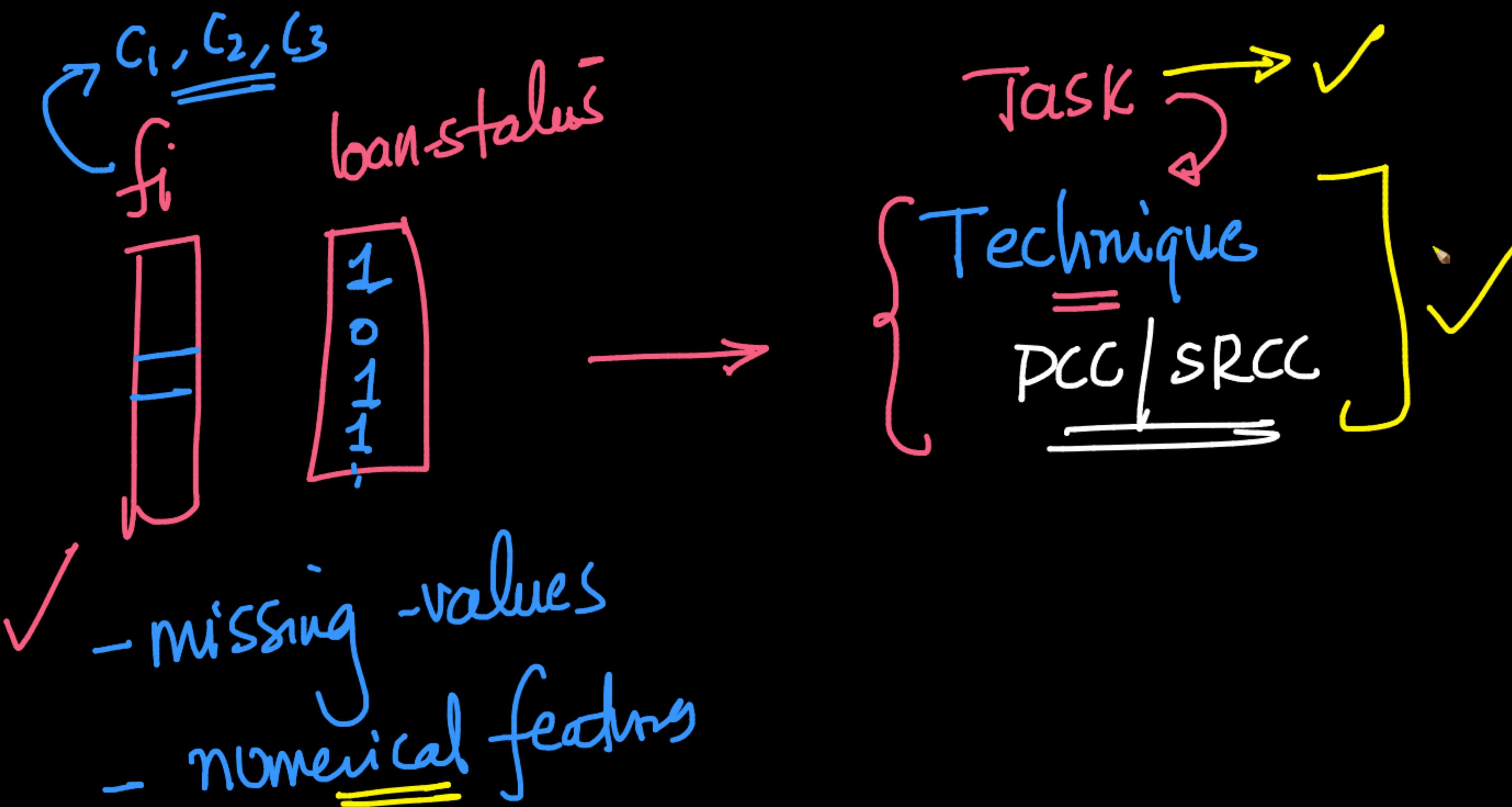
2. Label encoding

3. Target Encoding

Appropriate encoding depends on what our task is (and) what we do next?



32 / 32



 EDA\_FE.ipynb - Colaboratory  plot log(x) - Google Search

 plot log(x) - Google Search

[colab.research.google.com/drive/1\\_P-nKswaKnx5v7zIOx3pCxyJye73MvnN#scrollTo=edpBmBB0](https://colab.research.google.com/drive/1_P-nKswaKnx5v7zIOx3pCxyJye73MvnN#scrollTo=edpBmBB0)

+ Code + Text

RAM Disk

[ ] ↳ 3 cells hidden



## Missing Values & Data Cleaning

[249] data.isna().sum()

✓ Gender  
Married  
Dependents  
Education  
Self\_Employed  
ApplicantIncome  
CoapplicantIncome  
LoanAmount  
Loan\_Amount\_Term  
Credit\_History  
Property\_Area  
Loan\_Status  
TotalIncome  
Loan\_Amount\_per\_ye  
EMI  
Able\_to\_pay\_EMI

13

M, F,

 EDA\_FE.ipynb - Colaboratory  plot log(x) - Google Search

 plot log(x) - Google Search

colab.research.google.com/drive/1\_P-nKswaKnx5vZZIQx3pCxyJye73MyoN#scrollTo=edpBmBB0sb

+ Code + Text

✓ RAM Disk

[ ] ↳ 3 cells hidden



## Missing Values & Data Cleaning

[249] data.isna().sum()

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
TotalIncome	0
Loan_Amount_per_year	36
EMI	36
Able_to_pay_EMI	0



EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x

e.com/drive/1\_P-nKswaKnx5v77/Ox3pCxvJve73MyN#scrollTo= ZYMI6k8bp7h

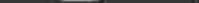
+ Code + Text

```
percent_missing_df = (df.isnull().sum() / df.isnull().count() * 100).sort_values(ascending=True)
missing_data_df = pd.concat([total_missing_df, percent_missing_df], axis=1, keys=['Total', 'Percent'])
return missing_data_df
```

RAM Disk

1

```
missing_df = missing_to_df(data)
missing_df[missing_df['Total'] > 0]
```



	Total	Percent
Credit_History	50	8.143322
Loan_Amount_per_year	36	5.863192
EMI	36	5.863192
Self_Employed	32	5.211726
LoanAmount	22	3.583062
Dependents	15	2.442997
Loan_Amount_Term	14	2.280130
Gender	13	2.117264
Married	3	0.488599

✓ 12521 # Credit History=2 for

 EDA\_FE.ipynb - Colaboratory   plot log(x) - Google Search 

[colab.research.google.com/drive/1\\_P-nKswaKnx5v77lOx3pCvxJve73MvnN#scrollTo=ZYMI6kBbp7h](https://colab.research.google.com/drive/1_P-nKswaKnx5v77lOx3pCvxJve73MvnN#scrollTo=ZYMI6kBbp7h)

+ Code + Text

```
percent_missing_df = (df.isnull().sum() / df.isnull().count() * 100).sort_values(ascending=True)
missing_data_df = pd.concat([total_missing_df, percent_missing_df], axis=1, keys=['Total', 'Percent'])
return missing_data_df
```

✓ RAM Disk

A white gear icon with three people icons to its left, representing user settings.

1

14

100

1

```
missing_df = missing_to_df(data)
missing_df[missing_df['Total'] > 0]
```

A horizontal bar containing several white icons on a dark background. From left to right, the icons are: an upward arrow, a downward arrow, a circular arrow, a magnifying glass, a gear, a clipboard with a checkmark, a trash can, and three vertical dots.

	Total	Percent
Credit_History	50	8.143322
Loan_Amount_per_year	36	5.863192
EMI	36	5.863192
Self_Employed	32	5.211726
LoanAmount	22	3.583062
Dependents	15	2.442997
Loan_Amount_Term	14	2.280130
Gender	13	2.117264
Married	3	0.488599



✓ r2521 # Credit History=2 for

EDA\_FE.ipynb - Colaboratory    X    plot log(x) - Google Search    X

<https://drive.google.com/drive/folders/1P-nKswaKnx5v77jOx3pCxyJye73MyoN#scrollTo=ZYMI6k8b>

+ Code + Text

✓ RAM Disk

A small icon bar containing two white icons: a user profile icon on the left and a gear icon on the right.

1

```
Able_to_pay_EMI  
dtype: int64
```

1

```
def missing_to_df(df):
    #Number and percentage of missing data in training data set for each column
    total_missing_df = df.isnull().sum().sort_values(ascending=False)
    percent_missing_df = (df.isnull().sum()/df.isnull().count()*100).sort_values(ascending=False)
    missing_data_df = pd.concat([total_missing_df, percent_missing_df], axis=1, keys=['Total', 'Percent'])
    return missing_data_df
```

A set of small, light-gray navigation icons located at the bottom right of the slide. From left to right, they include: a double-headed vertical arrow (for zooming), a double-headed horizontal arrow (for navigating between slides), a circular arrow (for refreshing or cycling through content), a magnifying glass (for search), a gear (for settings), a square with a double-headed arrow (for navigating between sections or frames), a trash can (for deleting), and three vertical dots (for more options).

```
missing_df = missing_to_df(data)
missing df[missing df['Total'] > 0]
```

	Total	Percent
Credit_History	50	8.143322
Loan_Amount_per_year	36	5.863192
EMI	36	5.863192
Self_Employed	32	5.211726
LoanAmount	22	3.583062

EDA\_FE.ipynb - Colaboratory x Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=\_ZYMI6k8bp7h

+ Code + Text RAM Disk

```
total_missing_df = df.isnull().sum().sort_values(ascending=False)
percent_missing_df = (df.isnull().sum()/df.isnull().count()*100).sort_values(ascending=False)
missing_data_df = pd.concat([total_missing_df, percent_missing_df], axis=1, keys=['Total', 'Percent'])
return missing_data_df
```

{x}

missing\_df = missing\_to\_df(data)  
missing\_df[missing\_df['Total'] > 0]

Total Percent

	Total	Percent
Credit_History	50	8.143322
Loan_Amount_per_year	36	5.863192
EMI	36	5.863192
Self_Employed	32	5.211726
LoanAmount	22	3.583062
Dependents	15	2.442997
Loan_Amount_Term	14	2.280130
Gender	13	2.117264
Married	3	0.488599

RAM Disk

Up Down Reload Settings Copy Paste Delete More

39/39

 EDA\_FE.ipynb - Colaboratory   plot log(x) - Google Search 

[https://colab.research.google.com/drive/1\\_P-nKswaKnx5y7ZlOx3pCxyJye73MvpN#scrollTo=ZYMI6k8pp7](https://colab.research.google.com/drive/1_P-nKswaKnx5y7ZlOx3pCxyJye73MvpN#scrollTo=ZYMI6k8pp7)

+ Code + Text

RAM Disk

1

✓ [250] # Function to create a data frame with number and percentage of missing data in a data frame

```
def missing_to_df(df):
    #Number and percentage of missing data in training data set for each column
    total_missing_df = df.isnull().sum().sort_values(ascending=False)
    percent_missing_df = (df.isnull().sum()/df.isnull().count()*100).sort_values(ascending=False)
    missing_data_df = pd.concat([total_missing_df, percent_missing_df], axis=1, keys=['Total', 'Percent'])
    return missing_data_df
```

A set of small, light-gray navigation icons located at the bottom right of the slide. From left to right, they include: a double-headed vertical arrow (for zooming), a double-headed horizontal arrow (for navigating between slides), a circular arrow (for refreshing or cycling through content), a magnifying glass (for search), a gear (for settings), a square with rounded corners (likely for a specific feature or mode), a trash can (for deleting), and three vertical dots (for more options).

```
missing_df = missing_to_df(data)
missing df[missing df['Total'] > 0]
```

	Total	Percent
Credit_History	50	8.143322
Loan_Amount_per_year	36	5.863192
EMI	36	5.863192
Self_Employed	32	5.211726
LoanAmount	22	3.583062
Dependents	15	2.442997

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x + colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=\_ZYMI6k8bp7h

+ Code + Text RAM Disk

Property\_Area 0  
Loan\_Status 0  
TotalIncome 0  
Loan\_Amount\_per\_year 36  
EMI 36  
Able\_to\_pay\_EMI 0  
dtype: int64

[250] # Function to create a data frame with number and percentage of missing data in a data frame

```
def missing_to_df(df):
    #Number and percentage of missing data in training data set for each column
    total_missing_df = df.isnull().sum().sort_values(ascending=False)
    percent_missing_df = (df.isnull().sum()/df.isnull().count()*100).sort_values(ascending=False)
    missing_data_df = pd.concat([total_missing_df, percent_missing_df], axis=1, keys=[ 'Total', 'Percent'])
    return missing_data_df
```

missing\_df = missing\_to\_df(data)  
missing\_df[missing\_df['Total'] > 0]

	Total	Percent
Credit_History	50	8.143322
Loan_Amount_per_year	36	5.863192

RAM Disk

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x + colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=FxqCjW7laETR

+ Code + Text RAM Disk

Loan Amount and Loan term

[ ] ↳ 11 cells hidden

{x}

Dependents and Loan approval

[ ] ↳ 6 cells hidden

Credit Score vs Loan Approval

CT

[247] data['Credit\_History'].value\_counts()

1.0 475  
0.0 89  
Name: Credit\_History, dtype: int64

[248] sns.countplot(data = data, x = 'Credit\_History', hue = 'Loan\_Status')  
#Observation:  
## We can clearly see that the approval rate is 80% if your credit history is aligned with the guidelines.  
## Hence this is the most important question that can be considered.

42/42

EDA\_FE.ipynb - Colaboratory x Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=FxqCjW7laETR

+ Code + Text

RAM Disk

Self\_Employed 32 3.211720

	LoanAmount	Dependents	Loan_Amount_Term	Gender	Married
22	3.583062	15	2.442997	14	2.280130
13	2.117264	3	0.488599	12	0.488599

{x}

LoanAmount 22 3.583062

Dependents 15 2.442997

Loan\_Amount\_Term 14 2.280130

Gender 13 2.117264

Married 3 0.488599

[252] # Credit History=2 for nan/missing values.  
data['Credit\_History'] = data['Credit\_History'].fillna(2)

[253] # Self\_Employed = 'Other' for nan  
data.Self\_Employed.unique()

array(['No', 'Yes', nan], dtype=object)

[254] data['Self\_Employed'] = data['Self\_Employed'].fillna('Other')

[255] # median imputation for numerical columns.

The image shows handwritten annotations in yellow ink on the screen. In the data section, there are five rows of numerical values. The first row has a circled '0' above it. The second row has a circled '1' above it. The third row has a circled '2' above it. The fourth row has a circled '1' above it. The fifth row has a circled '2' above it. In the code section, there are two lines of Python code. The first line has a bracket under 'fillna(2)' with a circled '2' above it. The second line has a bracket under 'data.Self\_Employed.unique()' with a circled '2' above it.

EDA\_FE.ipynb - Colaboratory x Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=FxqCjW7laETR

+ Code + Text

RAM Disk

Self\_Employed

	Count	Mean
LoanAmount	22	3.583062
Dependents	15	2.442997
Loan_Amount_Term	14	2.280130
Gender	13	2.117264
Married	3	0.488599

{x}

LoanAmount

Dependents

Loan\_Amount\_Term

Gender

Married

[252] # Credit History=2 for nan/missing values.  
data['Credit\_History'] = data['Credit\_History'].fillna(2)

[253] # Self\_Employed = 'Other' for nan  
data.Self\_Employed.unique()

array(['No', 'Yes', nan], dtype=object)

[254] data['Self\_Employed'] = data['Self\_Employed'].fillna('Other')

[255] # median imputation for numerical columns.

Imputation

Categorical

CH

44/44

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=bQzPBBynqbx2h

+ Code + Text

Dependents 15 2.442997

Loan\_Amount\_Term 14 2.280130

Gender 13 2.117264

Married 3 0.488599

{x}

RAM Disk

Up Down Reload Settings Copy Paste

# Credit History=2 for nan/missing values.  
data['Credit\_History'] = data['Credit\_History'].fillna(2)

[253] # Self\_Employed = 'Other' for nan  
data.Self\_Employed.unique()  
array(['No', 'Yes', nan], dtype=object)

[254] data['Self\_Employed'] = data['Self\_Employed'].fillna('Other')

[255] # median imputation for numerical columns.  
from sklearn.impute import SimpleImputer

num\_missing = ['EMI', 'Loan\_Amount\_per\_year', 'LoanAmount', 'Loan\_Amount\_Term']

Yes  
No  
Other ✓

45/45

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=bQzPBBynqbx2h

+ Code + Text

Dependents 15 2.442997

Loan\_Amount\_Term 14 2.280130

Gender 13 2.117264

Married 3 0.488599

{x}

RAM Disk

Up Down Reload Settings Copy All

# Credit History=2 for nan/missing values.  
data['Credit\_History'] = data['Credit\_History'].fillna(2)

[253] # Self\_Employed = 'Other' for nan  
data.Self\_Employed.unique()

array(['No', 'Yes', nan], dtype=object)

[254] data['Self\_Employed'] = data['Self\_Employed'].fillna('Other')

[255] # median imputation for numerical columns.  
from sklearn.impute import SimpleImputer

num\_missing = ['EMI', 'Loan\_Amount\_per\_year', 'LoanAmount', 'Loan\_Amount\_Term']

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x + colab.research.google.com/drive/1\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=bQzPBBynqbx2h

+ Code + Text RAM Disk

[253] # Self\_Employed = 'Other' for nan  
data.Self\_Employed.unique()  
array(['No', 'Yes', nan], dtype=object)

[254] data['Self\_Employed'] = data['Self\_Employed'].fillna('Other')

[255] # median imputation for numerical columns.  
from sklearn.impute import SimpleImputer  
  
num\_missing = ['EMI', 'Loan\_Amount\_per\_year', 'LoanAmount', 'Loan\_Amount\_Term']  
  
median\_imputer = SimpleImputer(strategy = 'median')  
for col in num\_missing:  
 data[col] = pd.DataFrame(median\_imputer.fit\_transform(pd.DataFrame(data[col])))

[256] # Highest Freq imputation for some categorical columns.  
cat\_missing = ['Gender', 'Married', 'Dependents']  
  
freq\_imputer = SimpleImputer(strategy = 'most\_frequent')  
for col in cat\_missing:

scikit-learn  
---  
classical ML

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x + colab.research.google.com/drive/1\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=bQzPBBynqbx2h

+ Code + Text RAM Disk

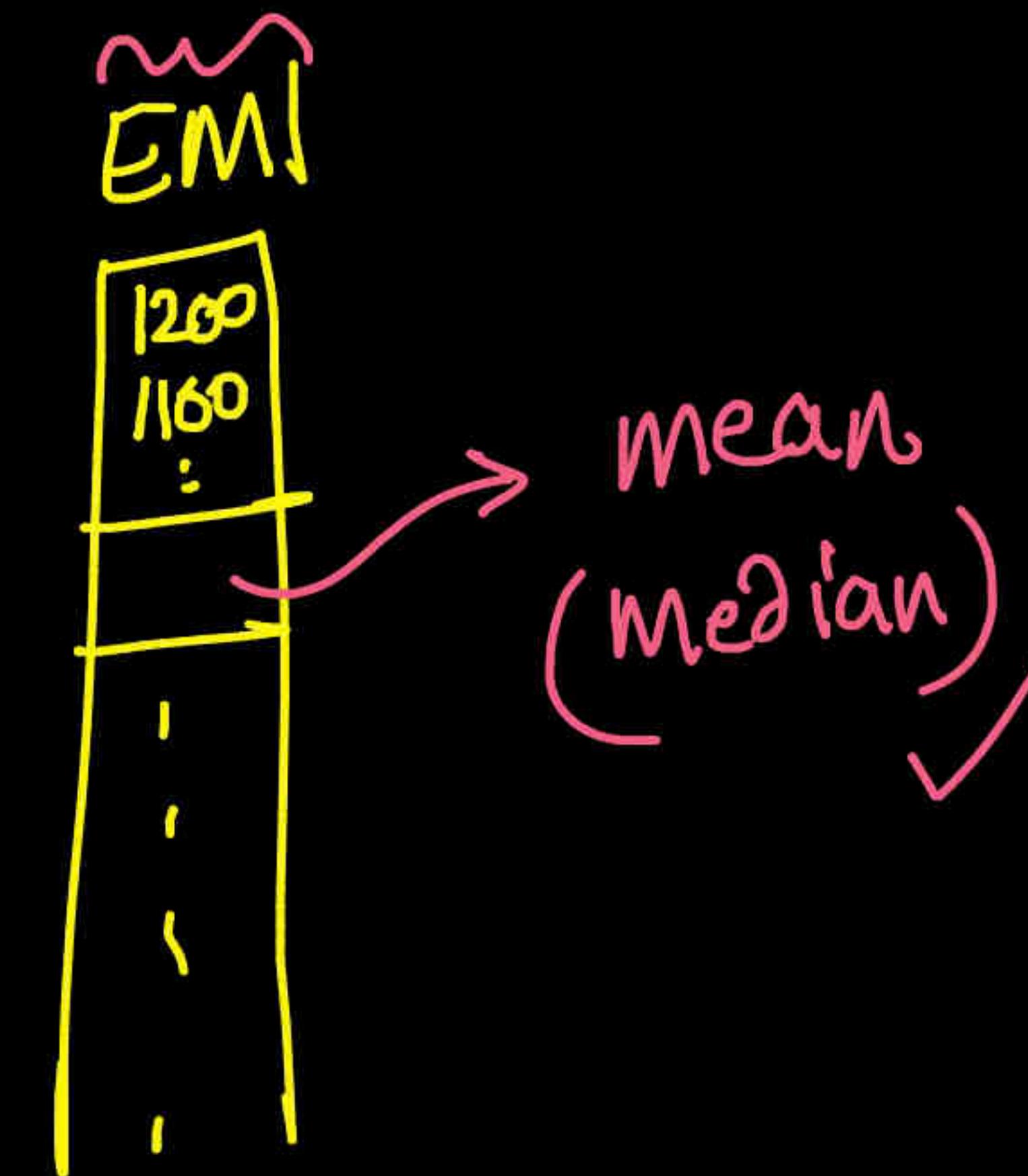
[253] # Self\_Employed = 'Other' for nan  
data.Self\_Employed.unique()

{x} array(['No', 'Yes', nan], dtype=object)

[254] data['Self\_Employed'] = data['Self\_Employed'].fillna('Other')

[255] # median imputation for numerical columns.  
from sklearn.impute import SimpleImputer  
  
num\_missing = ['EMI', 'Loan\_Amount\_per\_year', 'LoanAmount', 'Loan\_Amount\_Term']  
median\_imputer = SimpleImputer(strategy = 'median')  
for col in num\_missing:  
 data[col] = pd.DataFrame(median\_imputer.fit\_transform(pd.DataFrame(data[col])))

[256] # Highest Freq imputation for some categorical columns.  
cat\_missing = ['Gender', 'Married', 'Dependents']  
  
freq\_imputer = SimpleImputer(strategy = 'most\_frequent')  
for col in cat\_missing:  
 data[col] = pd.DataFrame(freq\_imputer.fit\_transform(pd.DataFrame(data[col])))





EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=SkB8xczbRc2

+ Code + Text Reconnect

Q [ ] data['Self\_Employed'] = data['Self\_Employed'].fillna('Other')

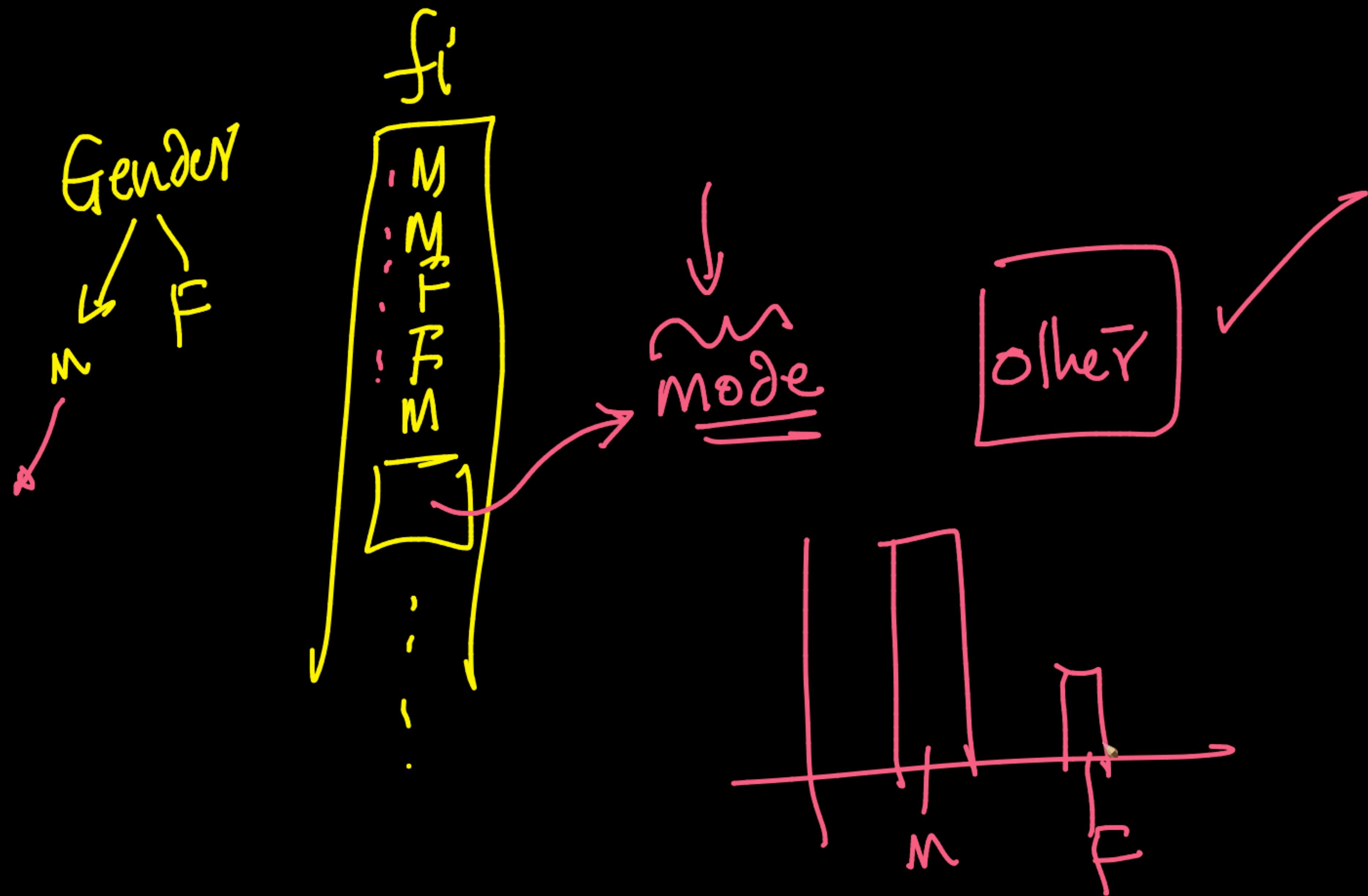
{x}

D [ ] # median imputation for numerical columns.  
from sklearn.impute import SimpleImputer  
  
num\_missing = ['EMI', 'Loan\_Amount\_per\_year', 'LoanAmount', 'Loan\_Amount\_Term']  
  
median\_imputer = SimpleImputer(strategy = 'median')  
for col in num\_missing:  
 data[col] = pd.DataFrame(median\_imputer.fit\_transform(pd.DataFrame(data[col])))  
  
✓ {  
  
[ ] # Highest Freq imputation for some categorical columns.  
cat\_missing = ['Gender', 'Married', 'Dependents']  
  
freq\_imputer = SimpleImputer(strategy = 'most\_frequent')  
for col in cat\_missing:  
 data[col] = pd.DataFrame(freq\_imputer.fit\_transform(pd.DataFrame(data[col])))  
  
<>  
  
[ ] missing\_df = missing\_to\_df(data)  
missing\_df[missing\_df['Total'] > 0]

▶

Page navigation icons: back, forward, search, etc.

Page footer: 51/51



EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x + colab.research.google.com/drive/1\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=SkB8xczbRc2 Reconnect + Code + Text

```
[ ] data['Self_Employed'] = data['Self_Employed'].fillna('Other')

{x}
[ ] # median imputation for numerical columns.
from sklearn.impute import SimpleImputer

num_missing = ['EMI', 'Loan_Amount_per_year', 'LoanAmount', 'Loan_Amount_Term']

median_imputer = SimpleImputer(strategy = 'median')
for col in num_missing:
    data[col] = pd.DataFrame(median_imputer.fit_transform(pd.DataFrame(data[col])))

[ ] # Highest Freq imputation for some categorical columns.
cat_missing = ['Gender', 'Married', 'Dependents']

freq_imputer = SimpleImputer(strategy = 'most_frequent')
for col in cat_missing:
    data[col] = pd.DataFrame(freq_imputer.fit_transform(pd.DataFrame(data[col])))

<> missing_df = missing_to_df(data)
missing_df[missing_df['Total'] > 0]
```

EMI  
100  
100  
100  
✓ Median=100





EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=SkB8xczbRc2

+ Code + Text Reconnect

```
num_missing = ['EMI', 'Loan_Amount_per_year', 'LoanAmount', 'Loan_Amount_Term']

median_imputer = SimpleImputer(strategy = 'median')
for col in num_missing:
    data[col] = pd.DataFrame(median_imputer.fit_transform(pd.DataFrame(data[col])))

[ ] # Highest Freq imputation for some categorical columns.
cat_missing = ['Gender', 'Married', 'Dependents']
freq_imputer = SimpleImputer(strategy = 'most_frequent')
for col in cat_missing:
    data[col] = pd.DataFrame(freq_imputer.fit_transform(pd.DataFrame(data[col])))

[ ] missing_df = missing_to_df(data)
missing_df[missing_df['Total'] > 0]
```

Total Percent

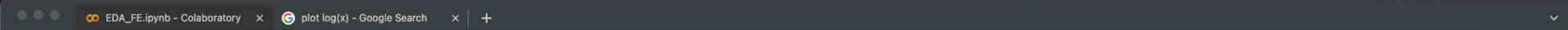


0, 1, 2, 3

Mod

## Categorical to Numerical encoding

Nominal vs Ordinal variables



+ Code + Text

Reconnect



```
num_missing = ['EMI', 'Loan_Amount_per_year', 'LoanAmount', 'Loan_Amount_Term']

median_imputer = SimpleImputer(strategy = 'median')
for col in num_missing:
    data[col] = pd.DataFrame(median_imputer.fit_transform(pd.DataFrame(data[col])))

[ ] # Highest Freq imputation for some categorical columns.
cat_missing = ['Gender', 'Married', 'Dependents']

freq_imputer = SimpleImputer(strategy = 'most_frequent')
for col in cat_missing:
    data[col] = pd.DataFrame(freq_imputer.fit_transform(pd.DataFrame(data[col])))

[ ] missing_df = missing_to_df(data)
missing_df[missing_df['Total'] > 0]
```

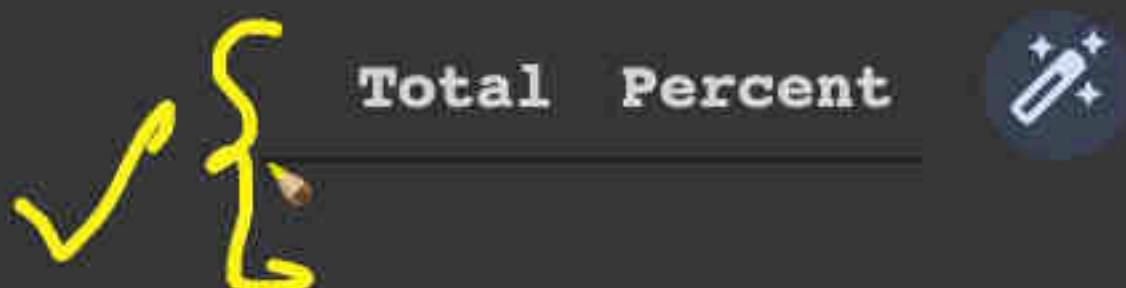
✓ { Total Percent



## ► Categorical to Numerical encoding

### Nominal vs Ordinal variables

```
∞ EDA_FE.ipynb - Colaboratory x Google Search x +  
colab.research.google.com/drive/1_P-nKswaKnx5v77lOx3pCvxJve73MvnN#scrollTo=SkB8xczbRc2  
+ Code + Text Reconnect ▾    
[ ] # Highest Freq imputation for some categorical columns.  
cat_missing = ['Gender', 'Married', 'Dependents']  
  
{x} freq_imputer = SimpleImputer(strategy = 'most_frequent')  
for col in cat_missing:  
    data[col] = pd.DataFrame(freq_imputer.fit_transform(pd.DataFrame(data[col])))  
  
[ ] missing_df = missing_to_df(data)  
missing_df[missing_df['Total'] > 0]
```



Total Percent



## ► Categorical to Numerical encoding

Nominal vs Ordinal variables

1. One Hot Encoding
2. Label encoding
3. Target Encoding

Appropriate encoding depends on what our task is (and) what we do next?

# Imputation

→ common sense  
✓ → median/ Mean  
→ Mode

→ model based imputation (KNN)  
Lader

0,1 2

Median - Imputation

median - Imputation

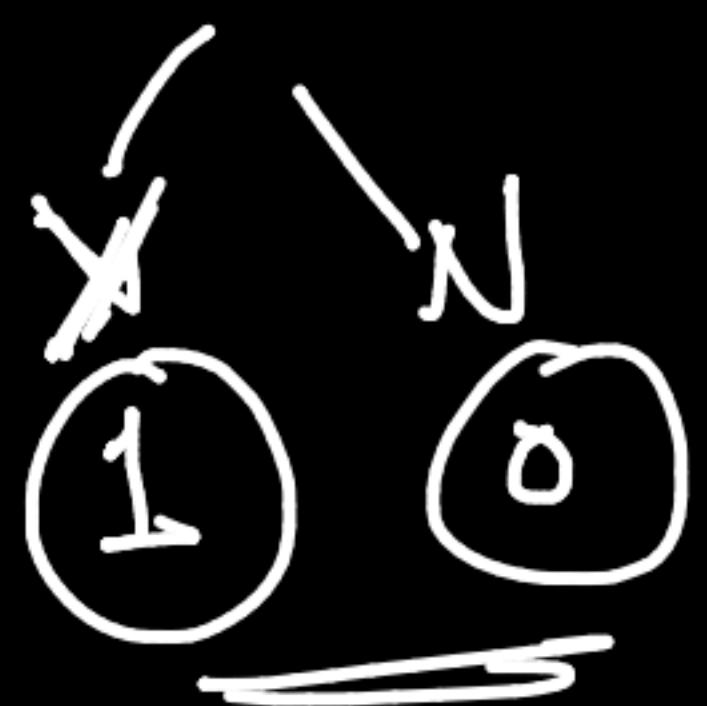
fit\_transform ... does internally

Do Not Disturb  
On

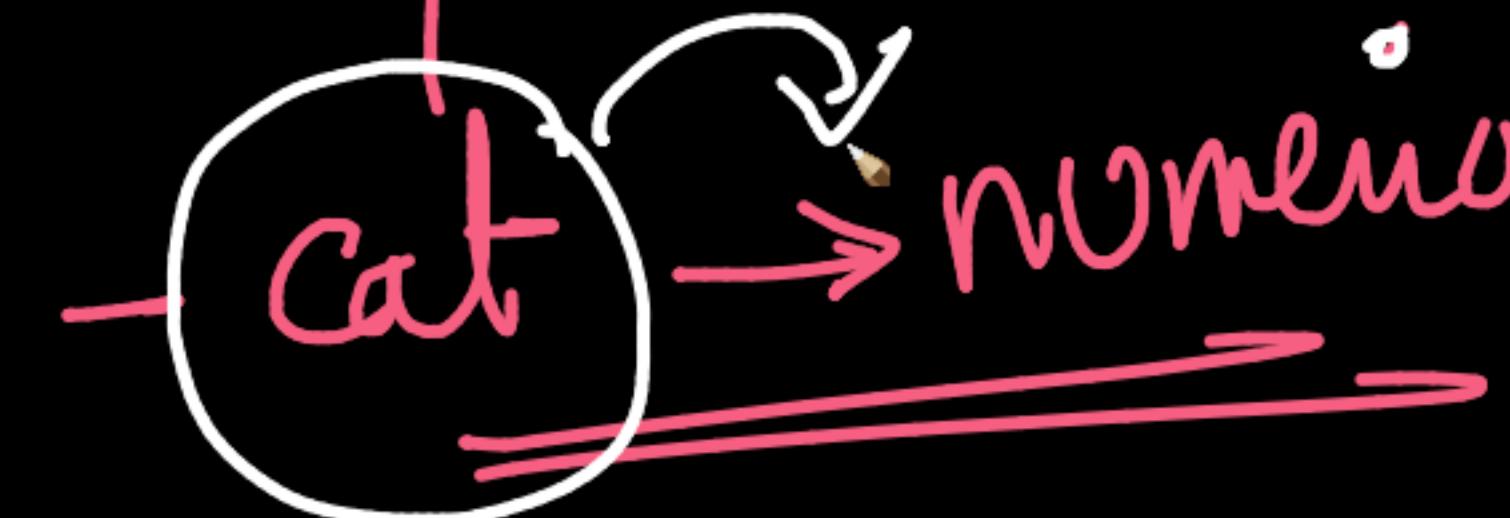
Task: find most useful features for loan-status

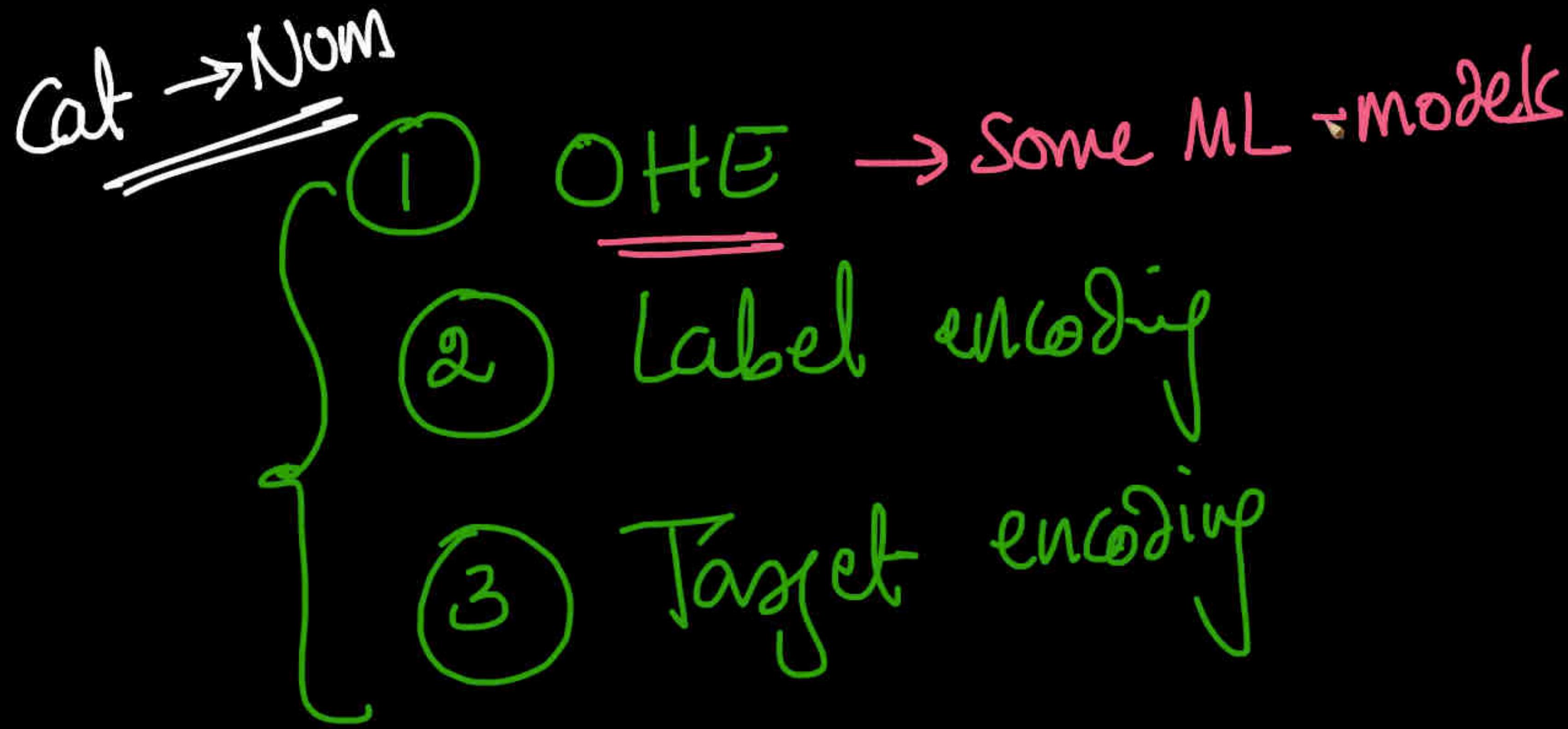
Tech: Correlation : PCC, SRCC

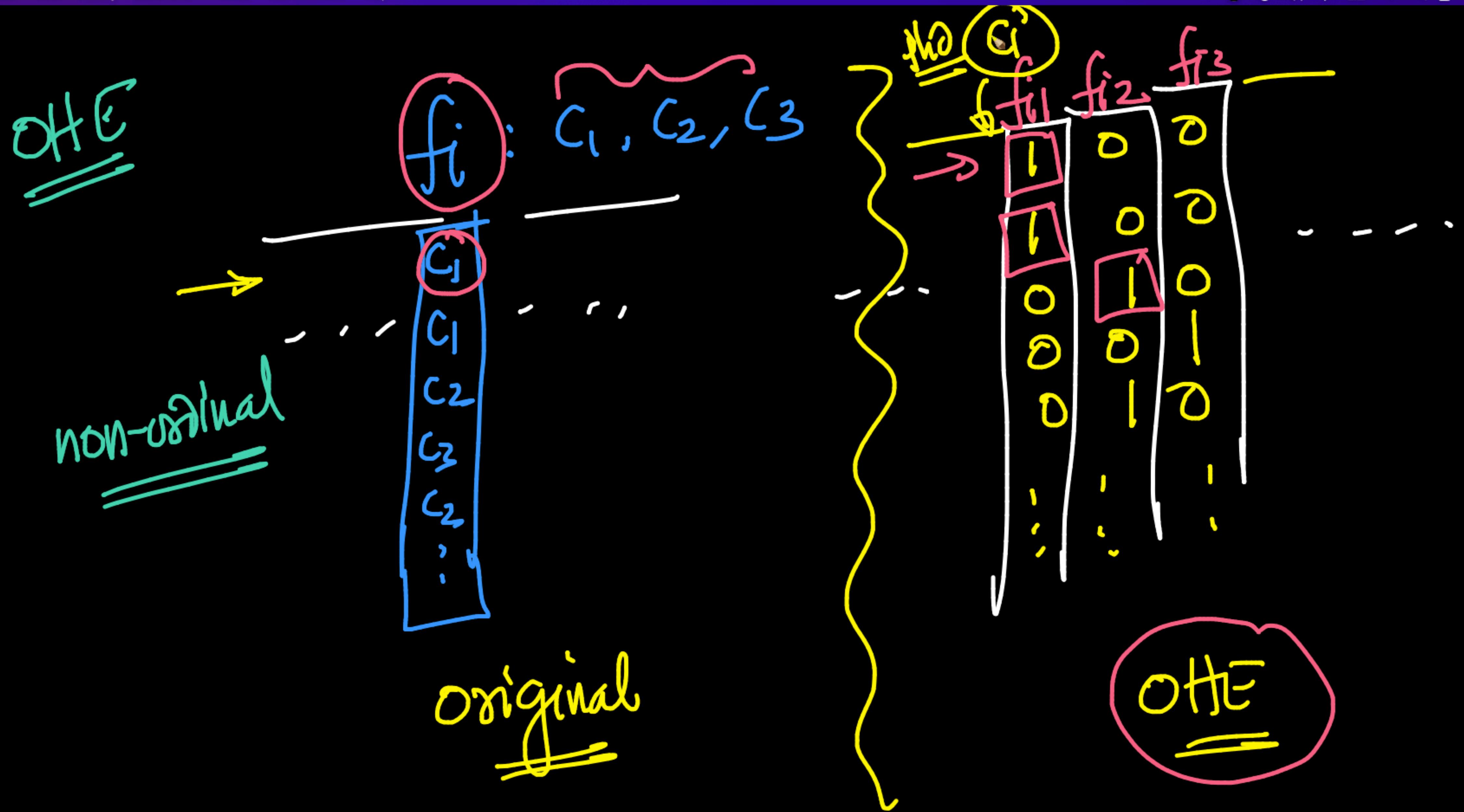
loan status



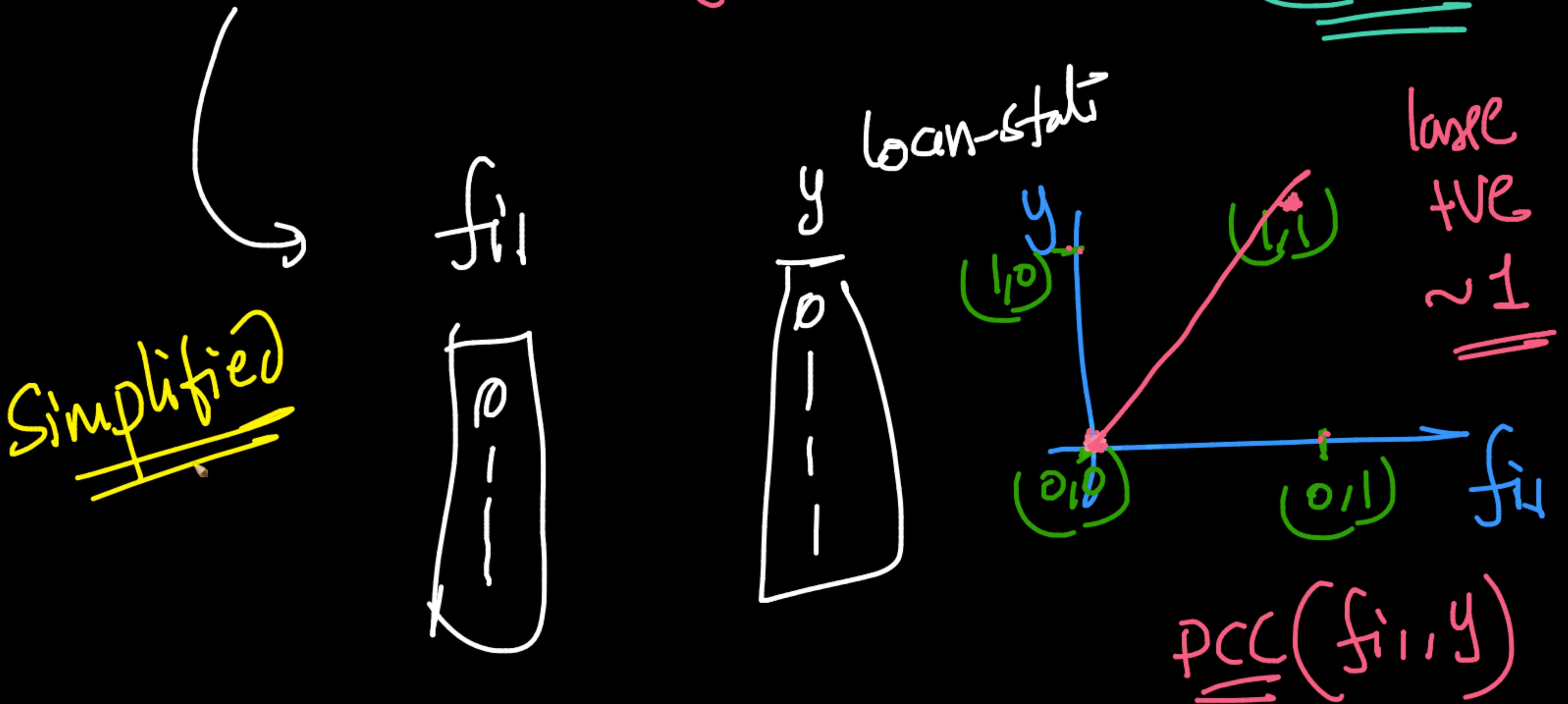
↓  
- impute & avoid missing values

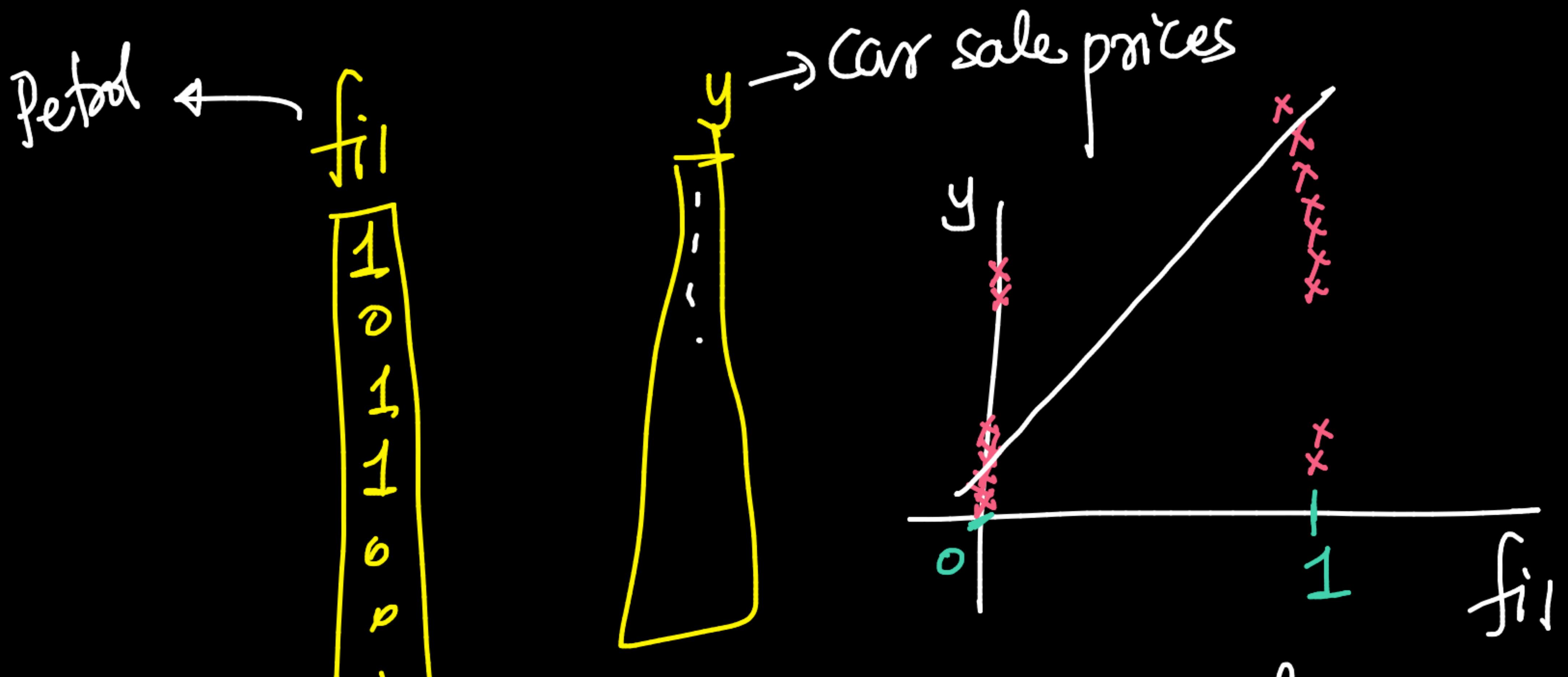






OHE  $\rightarrow$  Lin. Alg + Full-power  $\xrightarrow{\text{high-dim}}$  Geom + ML  
(later)





PCC : high +ve



high -dim → Gausse -d -dim

OHE → Carefully  
{log-Reg} {ly-reg}

(later) - ML

binary  
Cat - variables

loan-status

Y N  
1 0

✓ Label Encoder

Marital-status

M NM  
1 0

{

locality

U  
SU  
R

X Gender — M  
F  
Other — Z

arbitrarily order



binary  
Categorical  
features!

## Loan - status

Y D

We

4

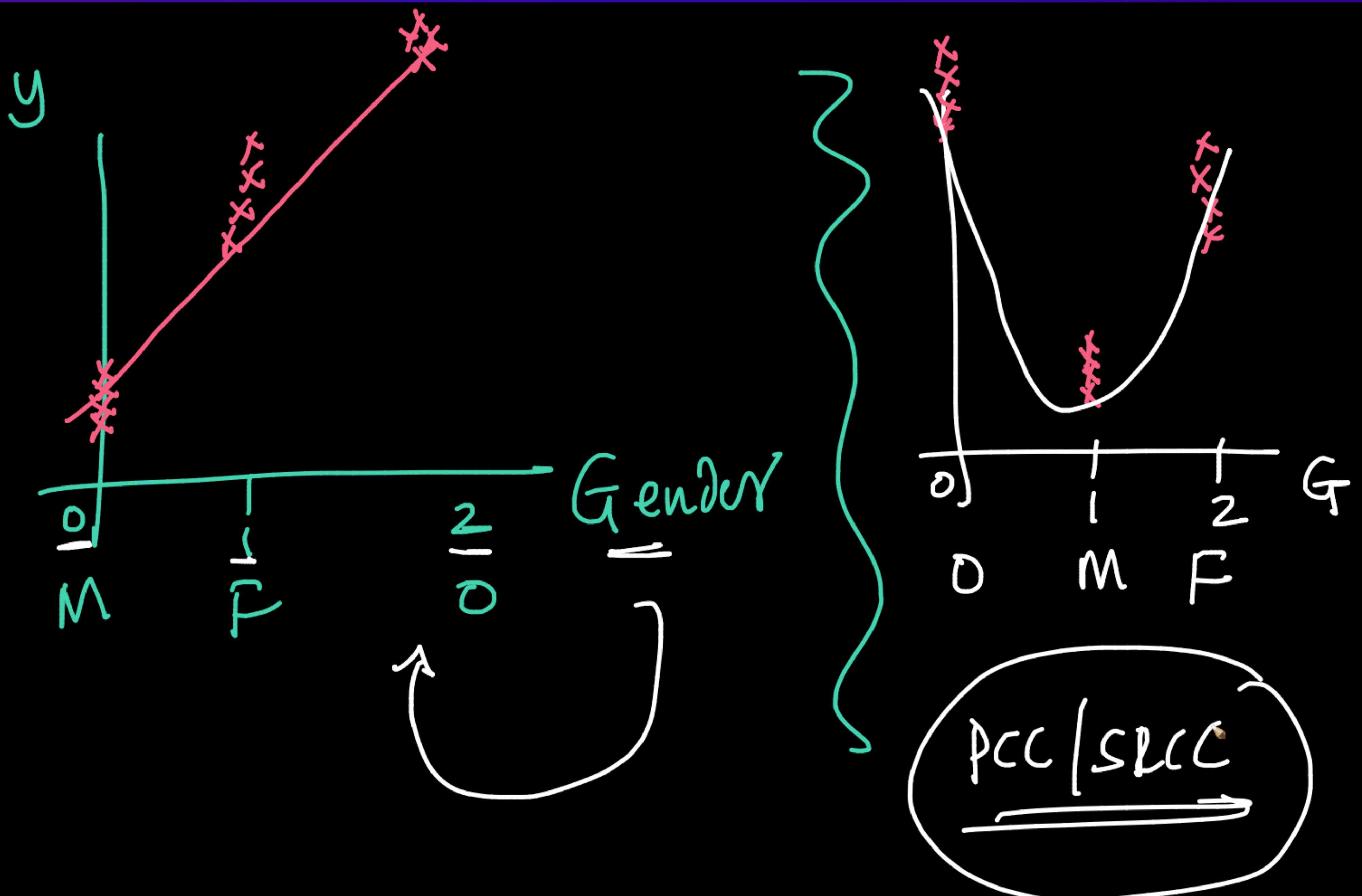
N: 1

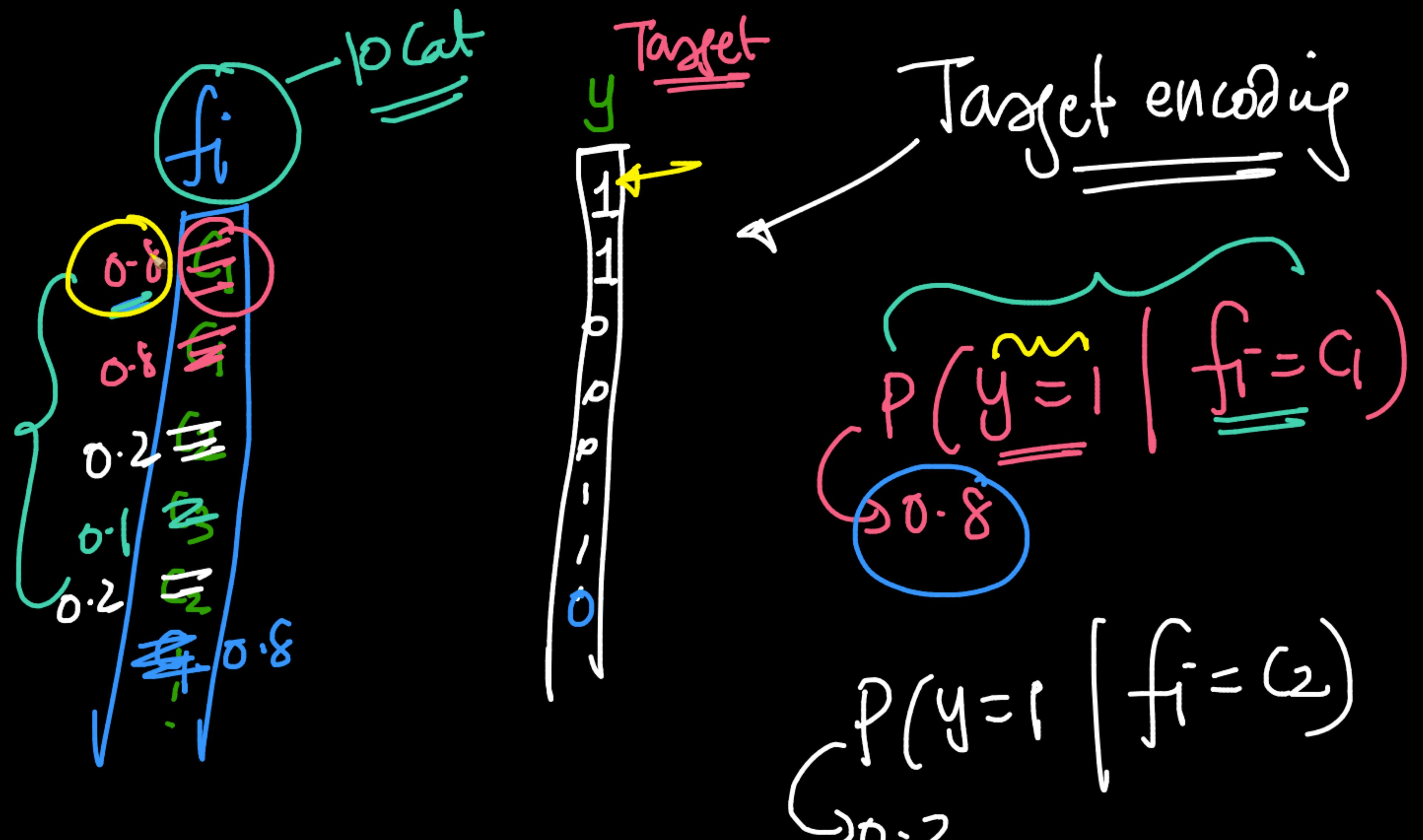
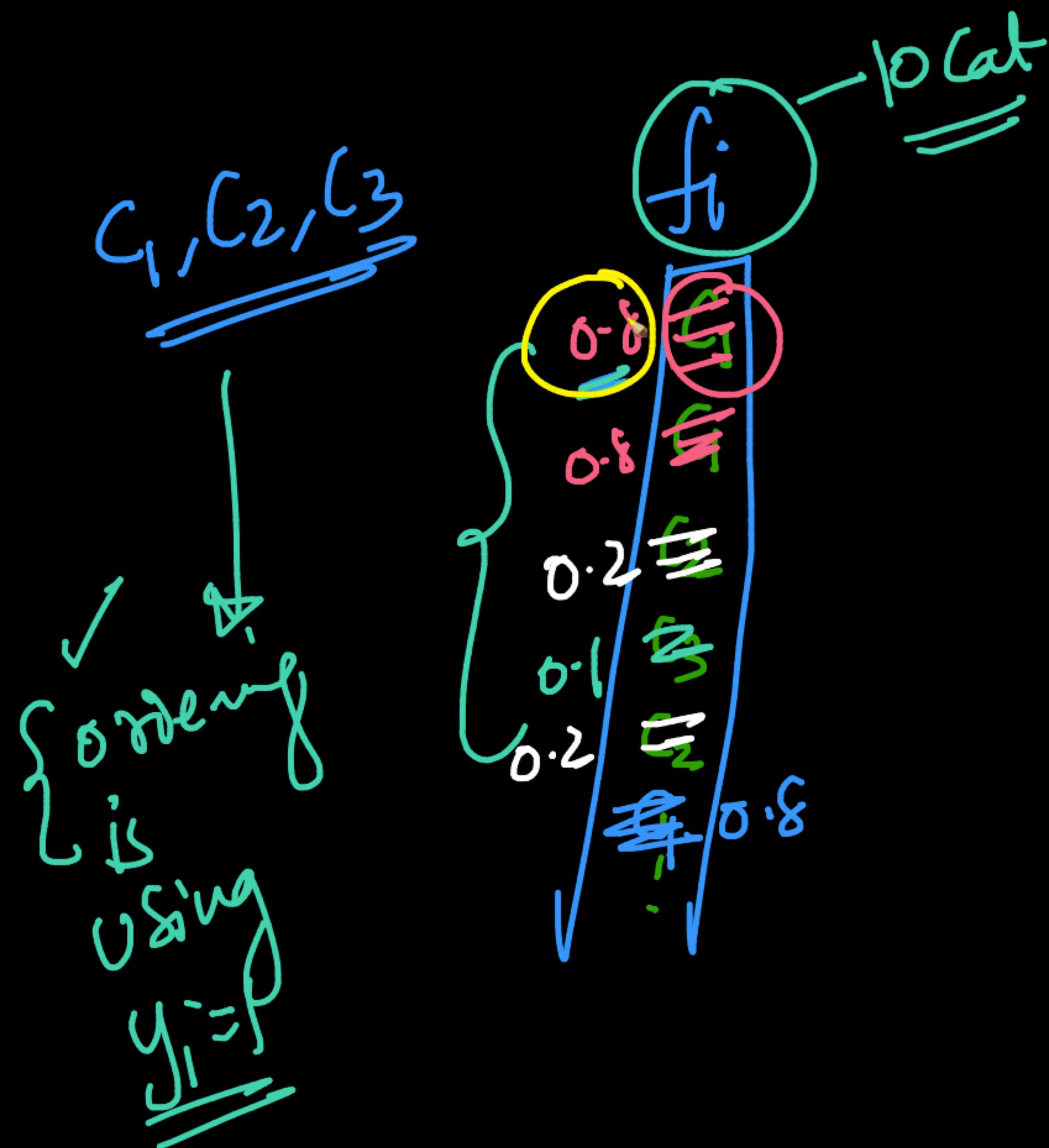
~~x x x x~~

PCL: large  
tree

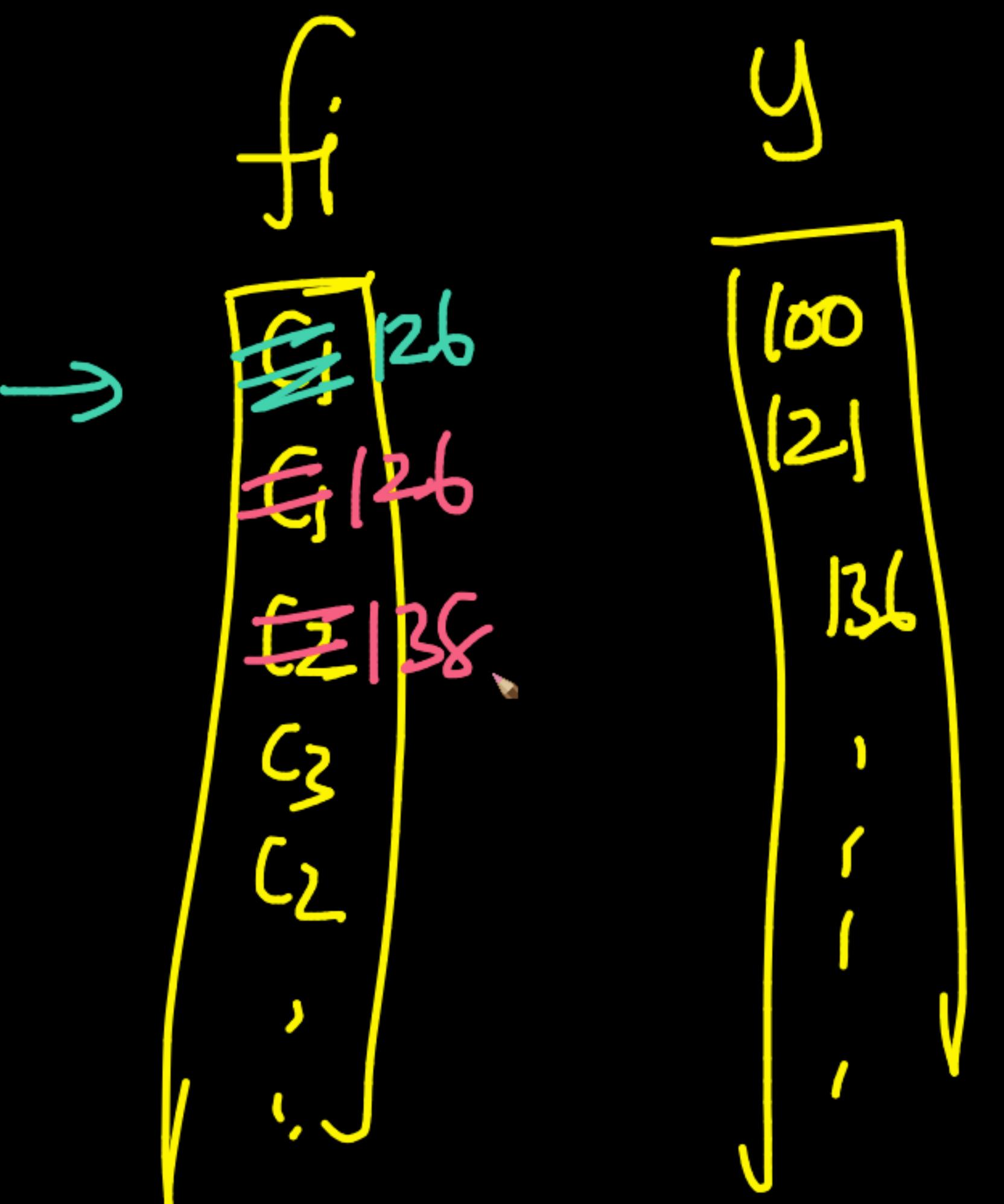
fi

A hand-drawn graph on a black background. The vertical axis is labeled 'y' and has a mark at '1'. The horizontal axis is labeled 'x' and has a mark at '0'. A curve starts at the point (0, 1) and slopes downward to the right, approaching the x-axis as x increases. A vertical line is drawn at x = 0. The region under the curve from x = 0 to the point where the curve meets the x-axis is shaded with red 'X' marks. To the left of the y-axis, there is a symbol resembling a 'Z' with a small triangle at its top. Below the x-axis, near the origin, the words 'large-ve' are written. To the right of the graph, there are some handwritten symbols: 'N', 'D', 've', 'f', and a checkmark.





Q



$$G_2 = \{126, 138\}$$

$\downarrow$

138

{ Target coding } ↓

126  $\rightarrow$  C<sub>1</sub> → {00, 121, 146...}

Mean/Median of all

$y_i$ 's where  
 $f_i = C_1$

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x + colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=35nPGGFZc4L Reconnect

+ Code + Text

1. One Hot Encoding

2. Label encoding

3. Target Encoding

{x}

Appropriate encoding depends on what our task is (and) what we do next?

Task: Compute Correlation (PCC and SRCC) between each feature and the Loan-Status

```
s = (data.dtypes == 'object')
object_cols = list(s[s].index)
object_cols
```

{ 'Gender',
'Married',
'Education',
'Self\_Employed',
'Property\_Area',
'Loan\_Status']

mean ✓

median

SD

MAD

▶ Loan Status

[ ] ↳ 3 cells hidden

72/72

EDA\_FE.ipynb - Colaboratory x Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=35nPGGFZc4L

+ Code + Text

Reconnect

1. One Hot Encoding

2. Label encoding

3. Target Encoding

{x}

Appropriate encoding depends on what our task is (and) what we do next?

Task: Compute Correlation (PCC and SRCC) between each feature and the Loan-Status

↑ ↓ ↻ ☰ ⚙️ 📁 🗑️ :

```
s = (data.dtypes == 'object')
object_cols = list(s[s].index)
object_cols

['Gender',
'Married',
'Education',
'Self_Employed',
'Property_Area',
'Loan_Status']
```

↔ Loan Status

[ ] ↳ 3 cells hidden

73 / 73

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x + colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=p1Bnyafgh7Ka

+ Code + Text Reconnect

Loan Status

{x}

```
[ ] # Loan_Status  
col='Loan_Status'  
data[col].value_counts()
```

Y 422  
N 192  
Name: Loan\_Status, dtype: int64

D [ ] from sklearn.preprocessing import LabelEncoder  
label\_encoder = LabelEncoder()  
col='Loan\_Status'  
data[col] = label\_encoder.fit\_transform(data[col])  
loan-status

```
[ ] data[col].value_counts()
```

1 422  
0 192  
Name: Loan\_Status, dtype: int64

Reconnect

Up Down Left Right Copy Paste Delete More

74 / 74

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x

[https://colab.research.google.com/drive/1\\_P-nKswaKpx5v7zIQx3pCxyJye73MyvN#scrollTo=p1Bovafgh7K](https://colab.research.google.com/drive/1_P-nKswaKpx5v7zIQx3pCxyJye73MyvN#scrollTo=p1Bovafgh7K)

◎ 内 容 ◎

+ Code + Text

Reconnect ▾

V

## ▼ Loan Status

100

```
[ ] # Loan_Status  
col='Loan_Status'  
data[col].value_counts()
```

```
Y      422  
N      192  
Name: Loan_Status, dtype: int64
```

```
[ ] from sklearn.preprocessing import LabelEncoder  
[ ]  
[ ] label_encoder = LabelEncoder()  
[ ] col='Loan_Status'  
[ ] data[col] = label_encoder.fit_transform(data[col])
```

```
[ ] data[col].value_counts()
```

```
1    422  
0    192  
Name: Loan_Status, dtype: int64
```

EDA\_FE.ipynb - Colaboratory    X    plot log(x) - Google Search    X

+ Code + Text

Reconnect ▾

1

## ▼ Loan Status

100

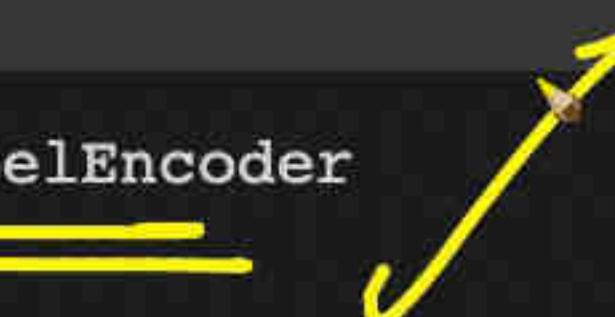
```
[ ] # Loan_Status  
col='Loan_Status'  
data[col].value_counts()
```

```
Y      422  
N      192  
Name: Loan_Status, dtype: int64
```

```
[ ] from sklearn.preprocessing import LabelEncoder  
label_encoder = LabelEncoder()  
col='Loan_Status'  
data[col] = label_encoder.fit_transform(data[col])
```

```
[ ] data[col].value_counts()
```

```
1      422  
8      192  
Name: Loan_Status, dtype: int64
```



EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=U-iTHyZPiAek

+ Code + Text Reconnect

Gender

{x}

```
[ ] #Gender  
data['Gender'].value_counts()
```

Male 502  
Female 112  
Name: Gender, dtype: int64

✓ Male 502  
✓ Female 112

```
[ ] from sklearn.preprocessing import LabelEncoder
```

label\_encoder = LabelEncoder()  
col='Gender'  
data[col] = label\_encoder.fit\_transform(data[col])

```
[ ] data[col].value_counts()
```

✓ 1 502  
✓ 0 112  
Name: Gender, dtype: int64

1 502  
0 112

 EDA\_FE.ipynb - Colaboratory  plot log(x) - Google Search

Code + Text

Reconnect ▾



1

```
1    502  
0    112  
Name: Gender, dtype: int64
```

↑ ↓ ⌂ ⌄ ⌅ ⌆ ⌇ ⌈ ⌉

▼ Married

```
[ ] # Married  
data[ 'Married' ].value_counts()
```

```
Yes      401  
No      213  
Name: Married, dtype: int64
```

```
[ ] label_encoder = LabelEncoder()
col='Married'
data[col] = label_encoder.fit_transform(data[col])
data[col].value_counts()
```

✓ 1 401  
✓ 0 213  
Name: Married, dtype: int64

+ Code + Tex

Reconnect ▾



[ ] ↗ 2 cells hidden



## ▼ Property Are

```
[ ] # col='Property_Area'  
col='Property_Area' ✎  
data[col].value_counts()
```

```
{ Semiurban      233  
Urban          202  
Rural          179  
Name: Property Area, dtype: int64
```

```
[ ] !pip install category_encoders
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: category_encoders in /usr/local/lib/python3.7/dist-packages (2.4.1)
Requirement already satisfied: scikit-learn>=0.20.0 in /usr/local/lib/python3.7/dist-packages (from category...
Requirement already satisfied: patsy>=0.5.1 in /usr/local/lib/python3.7/dist-packages (from category...
Requirement already satisfied: pandas>=0.21.1 in /usr/local/lib/python3.7/dist-packages (from category...
Requirement already satisfied: scipy>=1.0.0 in /usr/local/lib/python3.7/dist-packages (from category...
Requirement already satisfied: statsmodels>=0.10.0 in /usr/local/lib/python3.7/dist-packages (from category...
```

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x

[colab.research.google.com/drive/1\\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=WJABnH](https://colab.research.google.com/drive/1_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=WJABnH)

Q E ⋆ ☐

+ Code + Tex

Reconnect ▾



1

[ ] ↳ 2 cells hidden



#### ▼ Property Are

```
[ ] # col='Property_Area'  
col='Property_Area'  
data[col].value_counts()
```

```
Semiurban      233  
Urban          202  
Rural          179  
Name: Property Area, dtype: int64
```

```
[ ] !pip install category encoders
```



EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x + colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=WJABnHiHigf Reconnect + Code + Text

Requirement already satisfied: statsmodels>=0.9.0 in /usr/local/lib/python3.7/dist-packages (from category\_encoders)

Requirement already satisfied: numpy>=1.14.0 in /usr/local/lib/python3.7/dist-packages (from category\_encoders) (1.2)

Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (from pandas>=0.21.1->category

Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/dist-packages (from pandas>=0.21.1)

Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from patsy>=0.5.1->category\_encoders)

Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.

Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.20.0->ca

```
[ ] from category_encoders import TargetEncoder
      =✓
te = TargetEncoder()
data[col] = te.fit_transform(data[col], data['Loan_Status'])
```

```
[ ] col='Property_Area'
      data[col].value_counts()
```

0.768240	233
0.658416	202
0.614525	179

```
<> Name: Property_Area, dtype: int64
```

Education

81/81

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x + colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=WJABnHiHgIf Reconnect + Code + Text

```
Requirement already satisfied: statsmodels>=0.9.0 in /usr/local/lib/python3.7/dist-packages (from category_encoders)
Requirement already satisfied: numpy>=1.14.0 in /usr/local/lib/python3.7/dist-packages (from category_encoders) (1.2
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (from pandas>=0.21.1->category
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/dist-packages (from pandas>=0.21.1
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from patsy>=0.5.1->category_encoders)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.20.0->ca
```

[ ] from category\_encoders import TargetEncoder  
te = TargetEncoder()  
data[col] = te.fit\_transform(data[col], data['Loan\_Status'])  
→ loan-stt → 1  
→ 0

[ ] col='Property\_Area'  
data[col].value\_counts()

Area

$P(\text{loanStt} = 1 \mid \text{Area} = \text{W})$

Area	Count
0.768240	233
0.658416	202
0.614525	179

Name: Property\_Area, dtype: int64

Education

82/82

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=Qz-vLu5yipUQ

+ Code + Text Reconnect

Self Employed

{x}

```
[ ] col='Self_Employed'  
      data[col].value_counts()
```

No	500
Yes	82
Other	32

Name: Self\_Employed, dtype: int64

from category\_encoders import TargetEncoder

```
te = TargetEncoder()  
data[col] = te.fit_transform(data[col], data['Loan_Status'])  
data[col].value_counts()
```

0.686000	500
0.682927	82
0.718750	32

Name: Self\_Employed, dtype: int64

```
[ ] s = (data.dtypes == 'object')
```

83/83

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=Qz-vLu5yipUQ

+ Code + Text Reconnect

Self Employed

{x}

```
[ ] col='Self_Employed'  
      data[col].value_counts()
```

{ No 500  
Yes 82  
Other 32  
Name: Self\_Employed, dtype: int64

from category\_encoders import TargetEncoder

```
te = TargetEncoder()  
data[col] = te.fit_transform(data[col], data['Loan_Status'])  
data[col].value_counts()
```

0.686000 500  
0.682927 82  
0.718750 32  
Name: Self\_Employed, dtype: int64

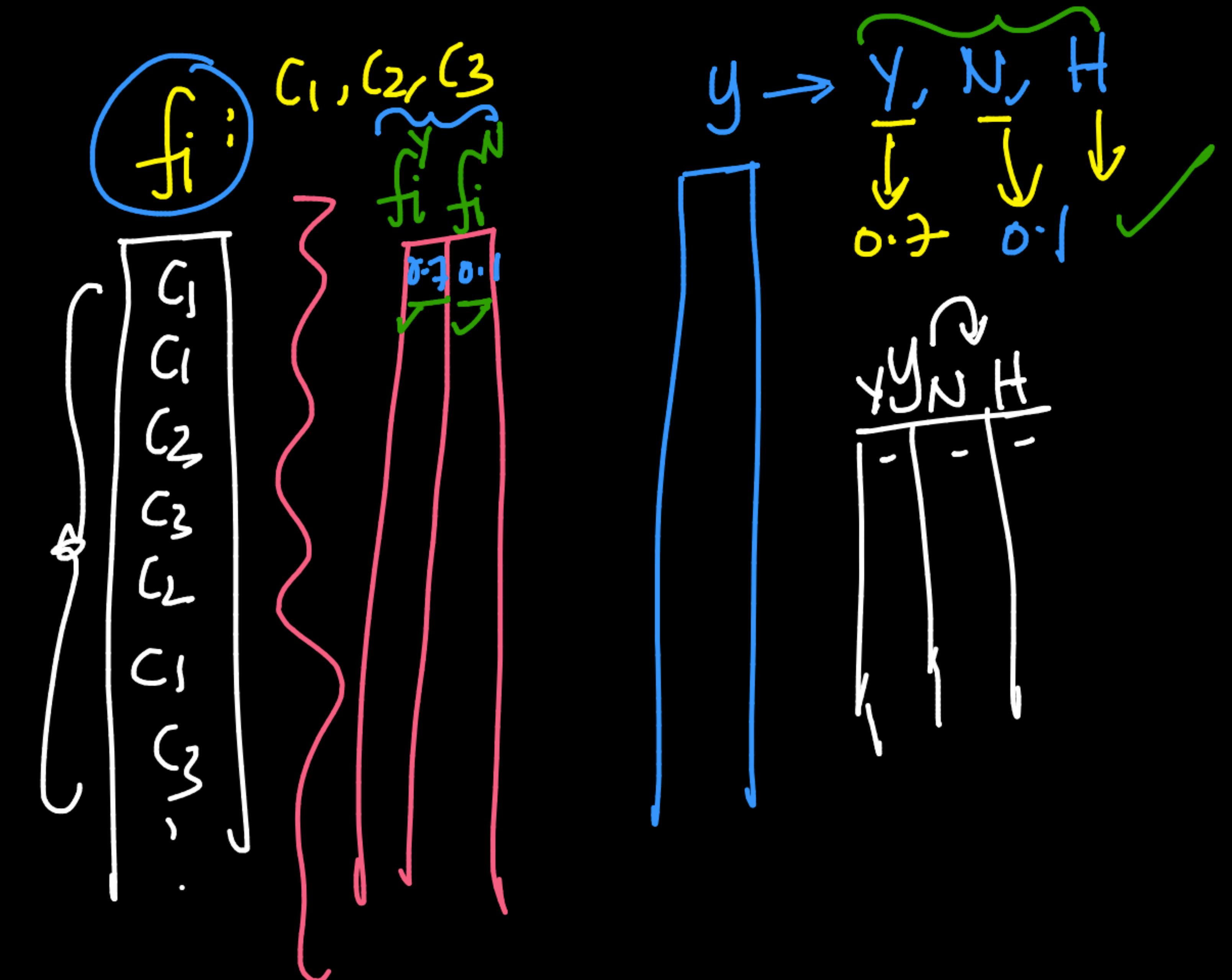
```
[ ] s = (data.dtypes == 'object')
```

84 / 84

Q  
 Why not w~~e~~ counts

$$\{P(y=Y | f_i = C_1) \\ = 0.7$$

$$P(y=N | f_i = C_1) \\ = 0.1$$



$$y = \begin{matrix} Y, N \\ \downarrow \quad \downarrow \\ 0.8 \quad \underline{\underline{0.2}} \end{matrix}$$

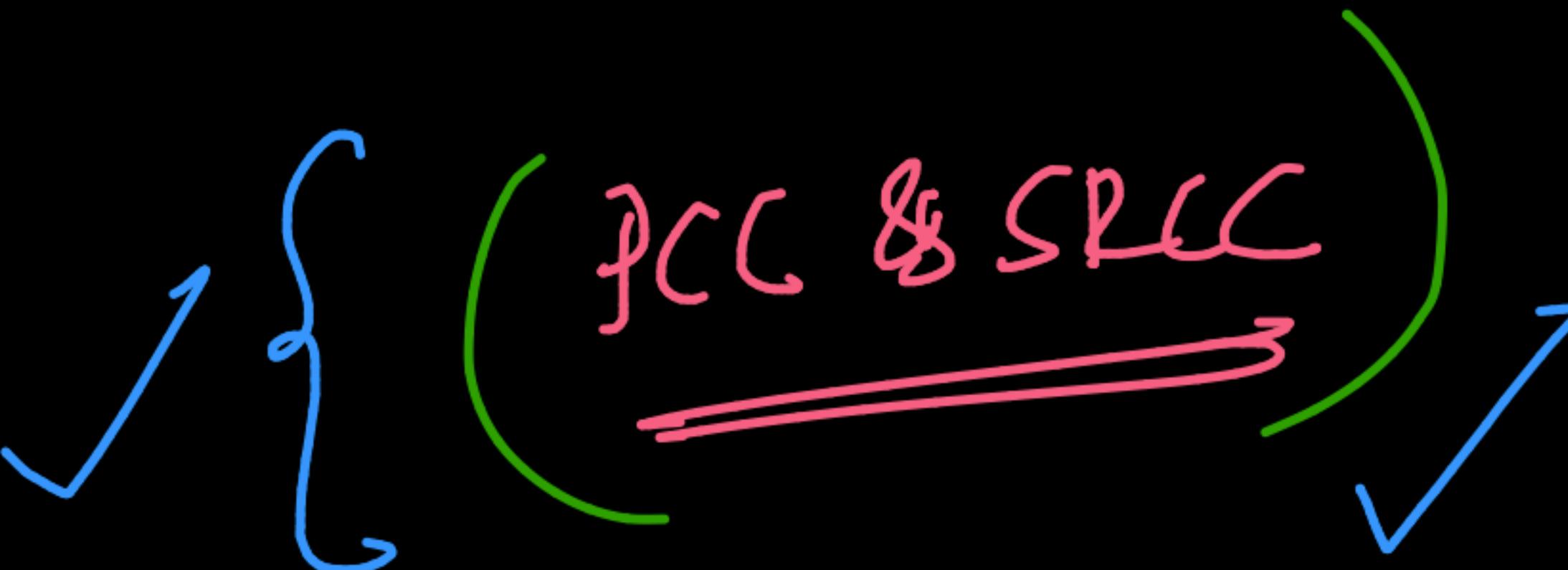
10:48



Missing-values



Cat  $\rightarrow$  Numeric



Standardization

$$\frac{x - \mu}{\sigma}$$

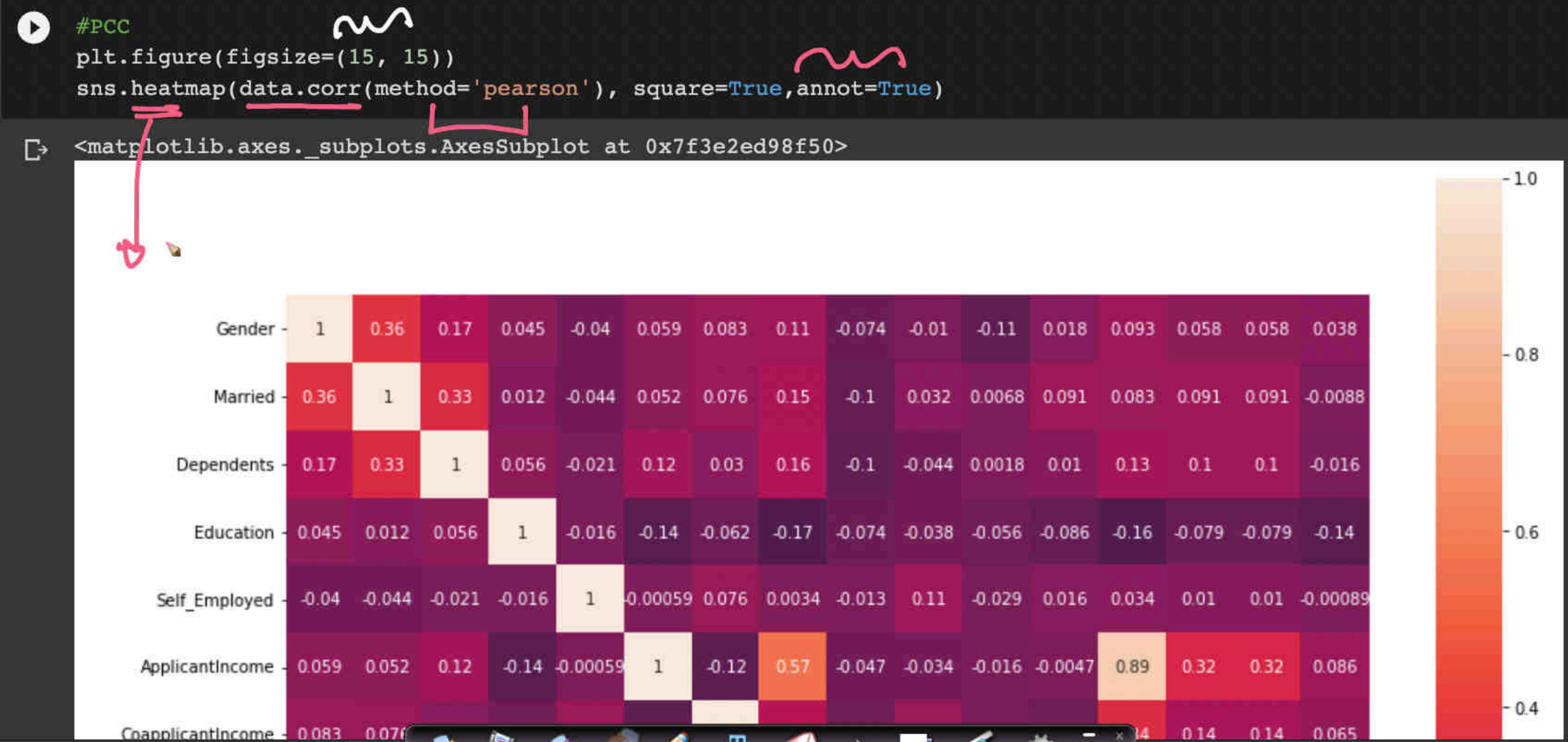


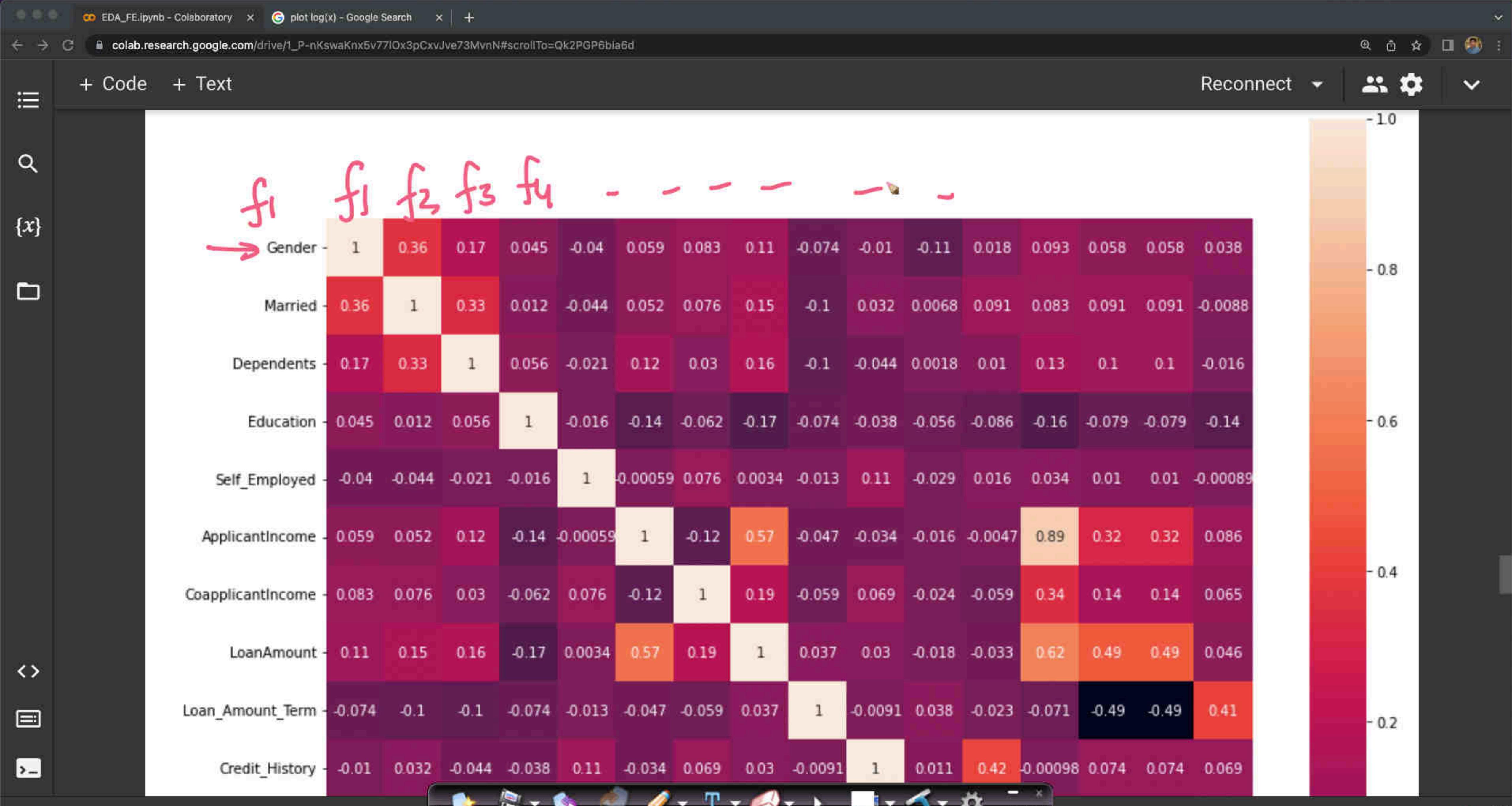
Simple  
extension

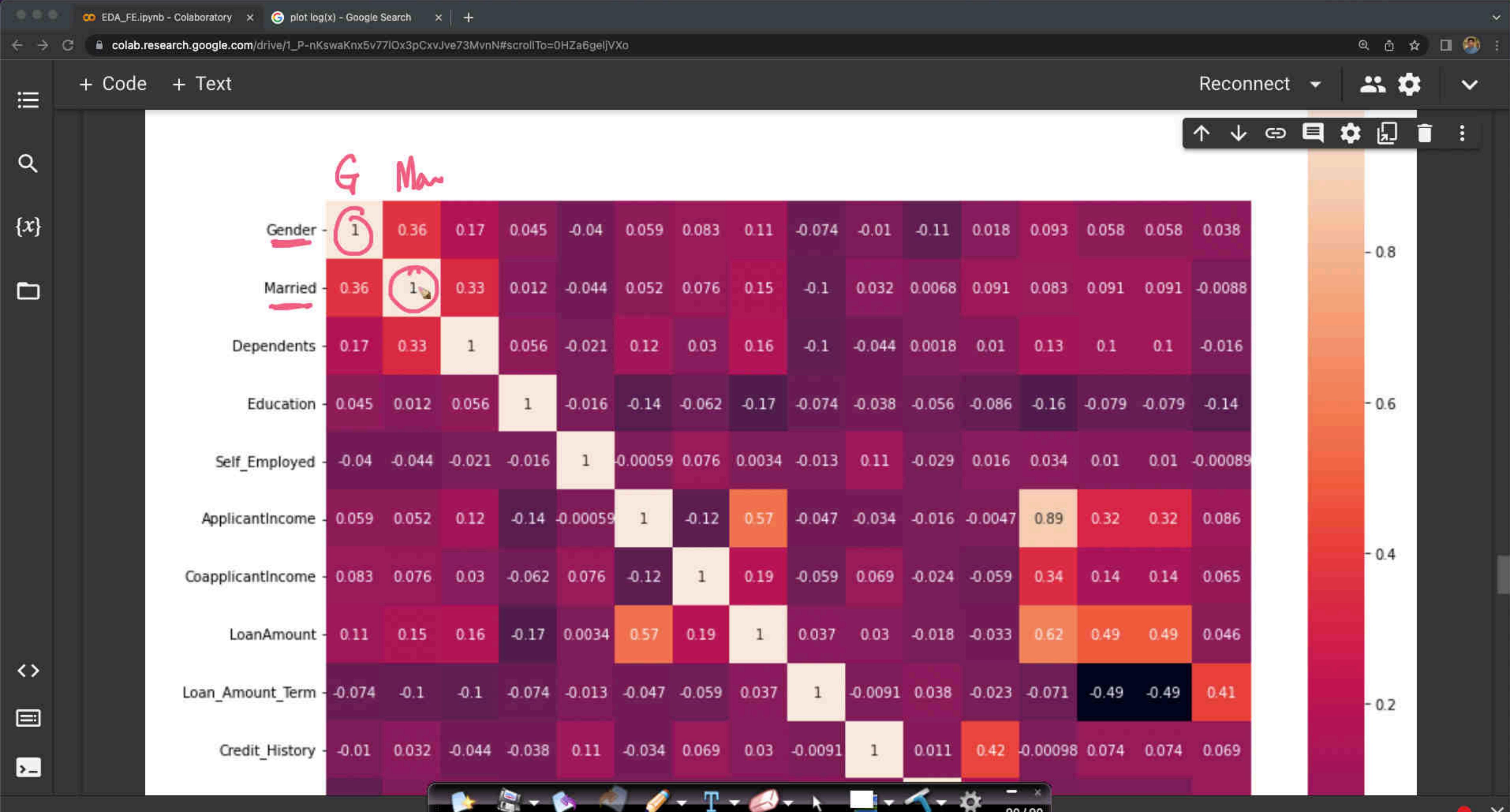


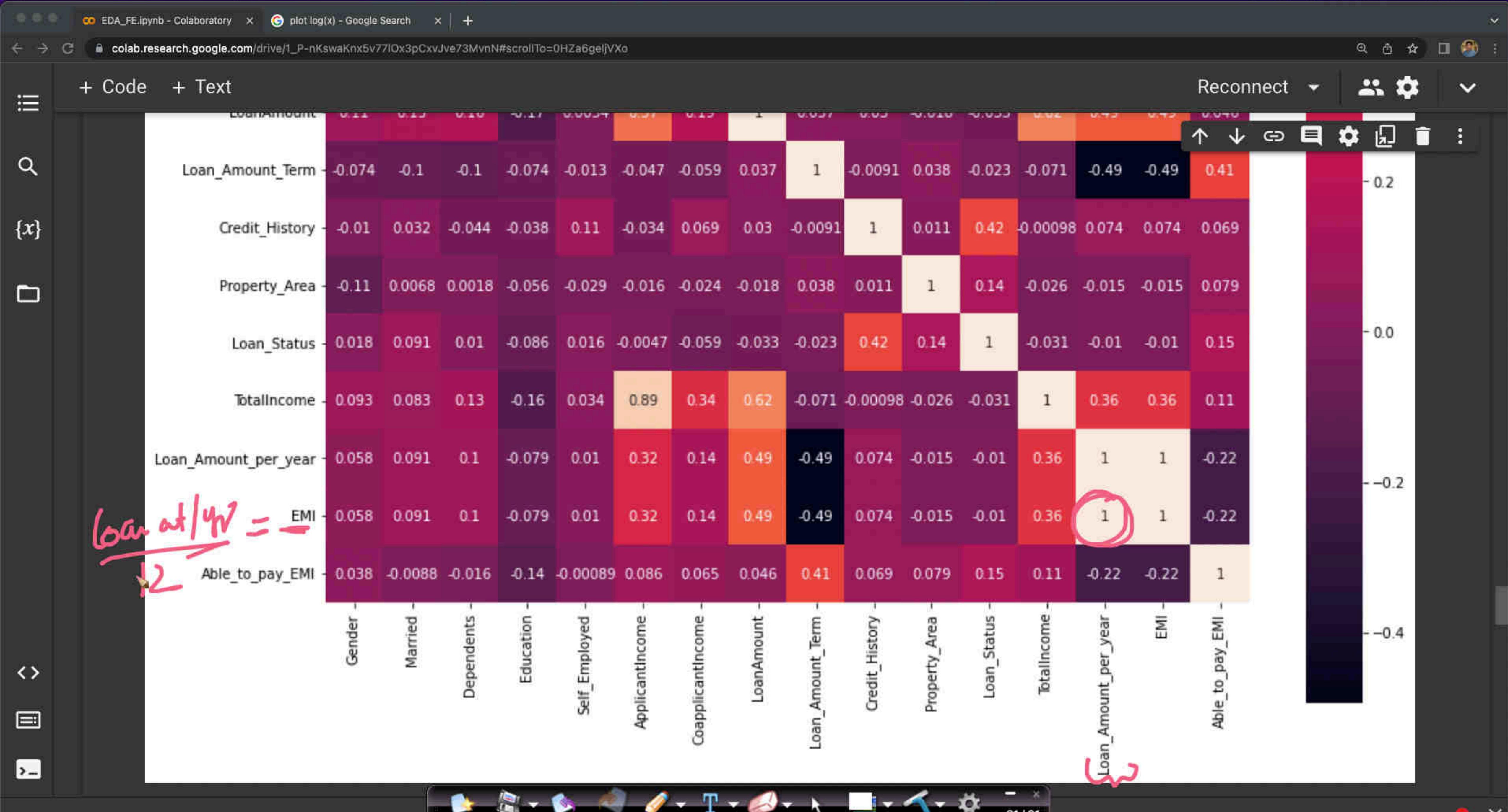
+ Code + Text

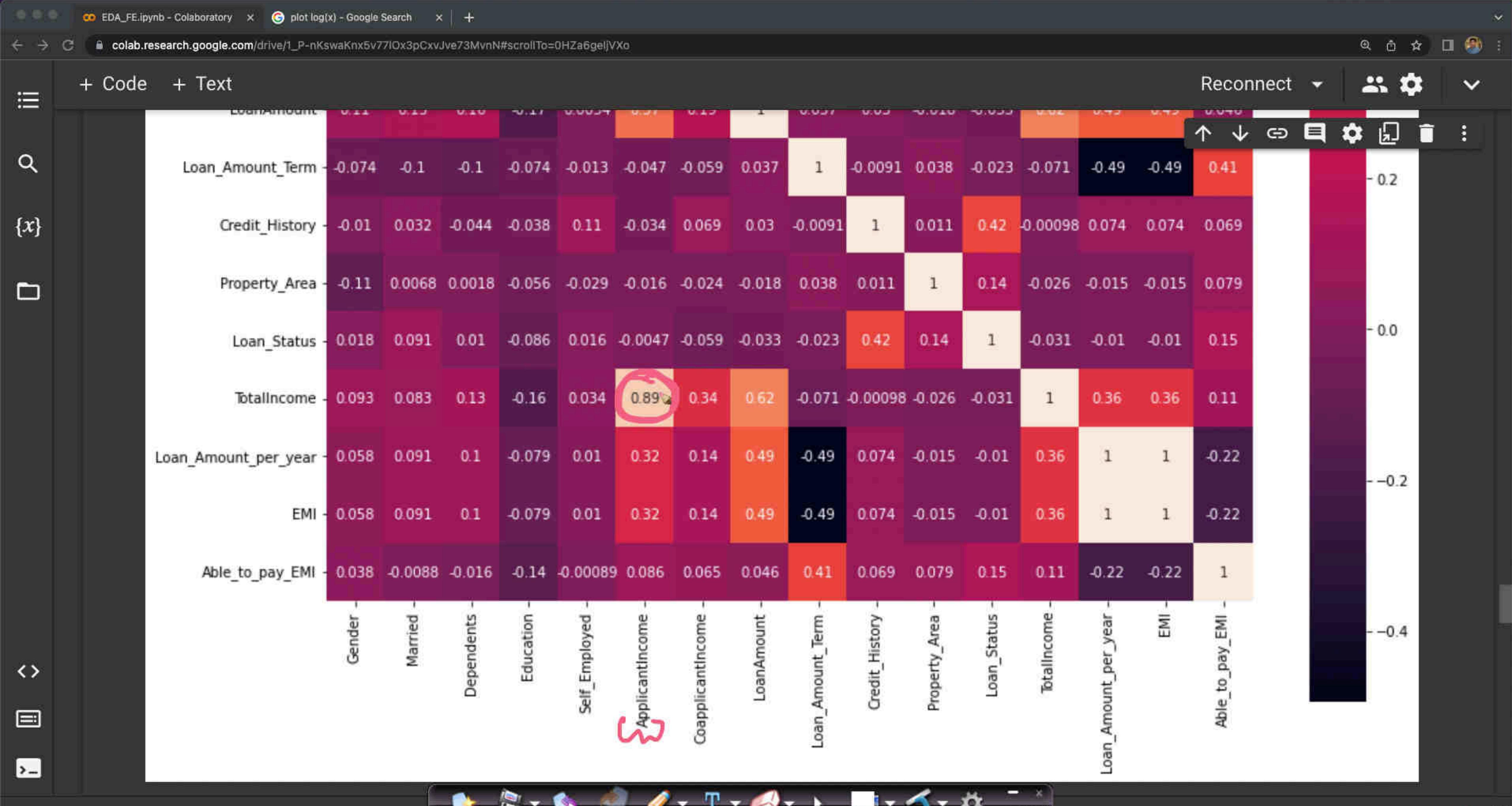
Reconnect ▾

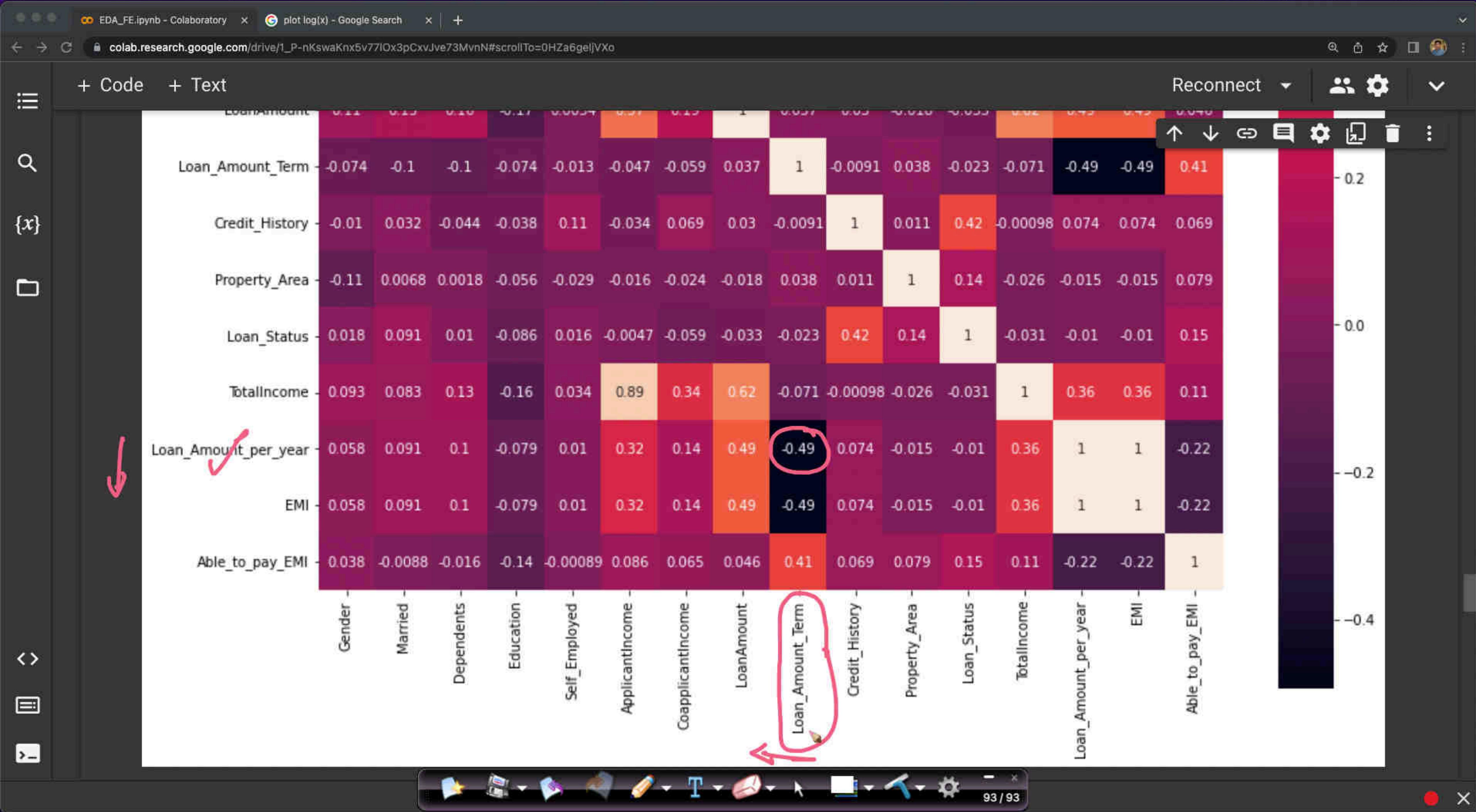












+ Code + Tex

## Reconnect

v

A heatmap visualization of a correlation matrix for various loan application features. The matrix is color-coded from dark purple (-0.4) to light yellow (1.0). A red arrow points to the 'Loan\_Status' row, which has a circled value of 0.42. A blue 'X' is placed over the diagonal element at the bottom right. The features listed on the axes are: Gender, Married, Dependents, Education, Self\_Employed, ApplicantIncome, CoapplicantIncome, LoanAmount, Loan\_Amount\_Term, Credit\_History, Property\_Area, Loan\_Status, TotalIncome, Loan\_Amount\_per\_year, EMI, and Able\_to\_pay\_EMI.

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status	TotalIncome	Loan_Amount_per_year	EMI	Able_to_pay_EMI
LoanAmount	0.11	0.15	0.16	-0.17	0.0034	0.57	0.19	1	0.037	0.03	-0.018	-0.033	0.62	0.49	0.49	↑
Loan_Amount_Term	-0.074	-0.1	-0.1	-0.074	-0.013	-0.047	-0.059	0.037	1	-0.0091	0.038	-0.023	-0.071	-0.49	-0.49	0.41
Credit_History	-0.01	0.032	-0.044	-0.038	0.11	-0.034	0.069	0.03	-0.0091	1	0.011	0.42	-0.00098	0.074	0.074	0.069
Property_Area	-0.11	0.0068	0.0018	-0.056	-0.029	-0.016	-0.024	-0.018	0.038	0.011	1	0.14	-0.026	-0.015	-0.015	0.079
Loan_Status	0.018	0.091	0.01	-0.086	0.016	-0.0047	-0.059	-0.033	-0.023	0.42	0.14	X	-0.031	-0.01	-0.01	0.15
TotalIncome	0.093	0.083	0.13	-0.16	0.034	0.89	0.34	0.62	-0.071	-0.00098	-0.026	-0.031	1	0.36	0.36	0.11
Loan_Amount_per_year	0.058	0.091	0.1	-0.079	0.01	0.32	0.14	0.49	-0.49	0.074	-0.015	-0.01	0.36	1	1	-0.22
EMI	0.058	0.091	0.1	-0.079	0.01	0.32	0.14	0.49	-0.49	0.074	-0.015	-0.01	0.36	1	1	-0.22
Able_to_pay_EMI	0.038	-0.0088	-0.016	-0.14	-0.00089	0.086	0.065	0.046	0.41	0.069	0.079	0.15	0.11	-0.22	-0.22	1

+ Code + Tex

## Reconnect

1

The figure is a correlation matrix heatmap for a dataset of loan applications. The x-axis and y-axis both list the following features: Gender, Married, Dependents, Education, Self\_Employed, ApplicantIncome, CoapplicantIncome, LoanAmount, Loan\_Amount\_Term, Credit\_History, Property\_Area, Loan\_Status, TotalIncome, Loan\_Amount\_per\_year, EMI, and Able\_to\_pay\_EMI. The color scale indicates the strength and sign of the correlation, ranging from -0.4 (dark blue) to 1.0 (white/yellow). A legend on the right side shows a gradient from dark purple (-0.4) to light yellow (1.0).

Annotations in the matrix:

- A blue arrow points to the cell for 'Loan\_Status' at row 4, column 1.
- Blue circles highlight several correlations:
  - Row 4, columns 11-12: Correlations between 'Loan\_Status' and 'Property\_Area' (0.11), 'Credit\_History' (0.11), and 'TotalIncome' (0.11).
  - Row 4, columns 13-14: Correlations between 'Loan\_Status' and 'Loan\_Status' (1.0), 'TotalIncome' (-0.031), 'Loan\_Amount\_per\_year' (-0.01), and 'EMI' (-0.01).
  - Row 4, column 15: Correlation between 'Loan\_Status' and 'Able\_to\_pay\_EMI' (0.15).
- A blue checkmark is placed over the cell for 'Property\_Area' at row 5, column 12.
- A blue circle highlights the strong positive correlation between 'TotalIncome' and 'EMI' (0.36).
- A large blue circle highlights the diagonal elements of the matrix, representing the correlation of each feature with itself (all 1.0).
- A blue arrow points to the cell for 'Able\_to\_pay\_EMI' at row 13, column 14.

+ Code + Tex

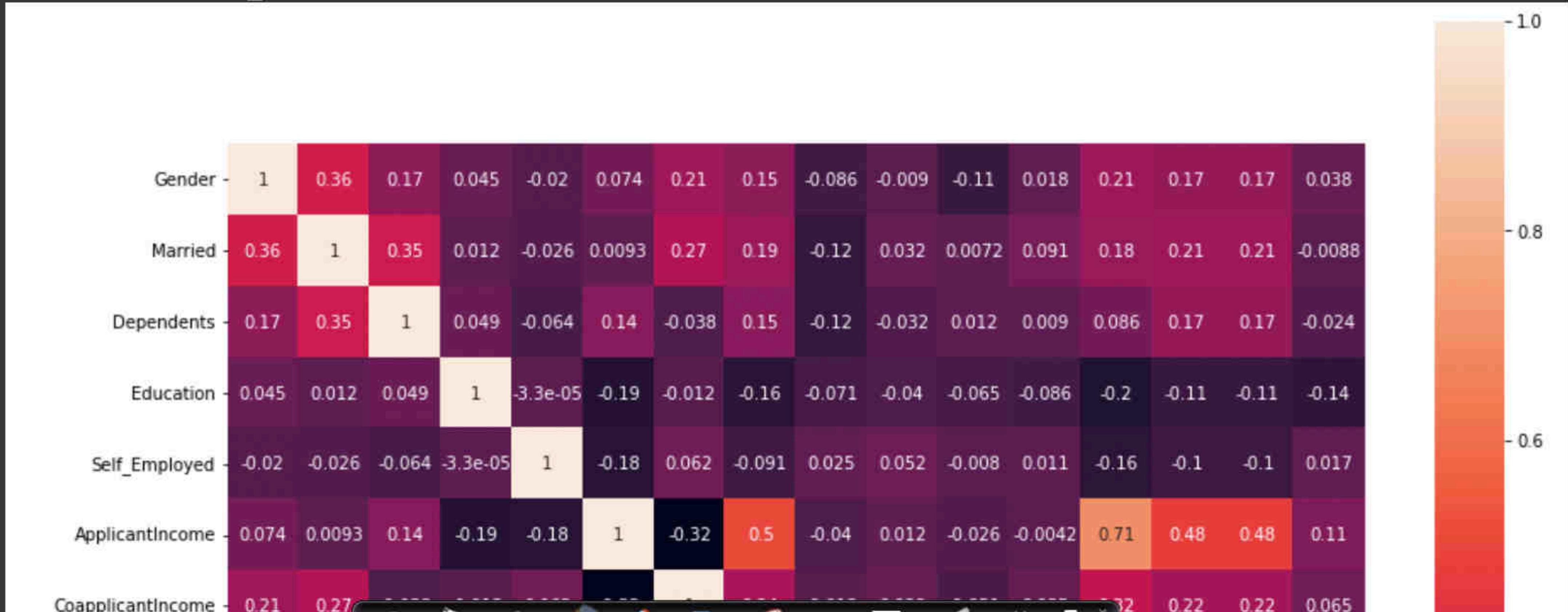
Reconnect ▾



#SRC

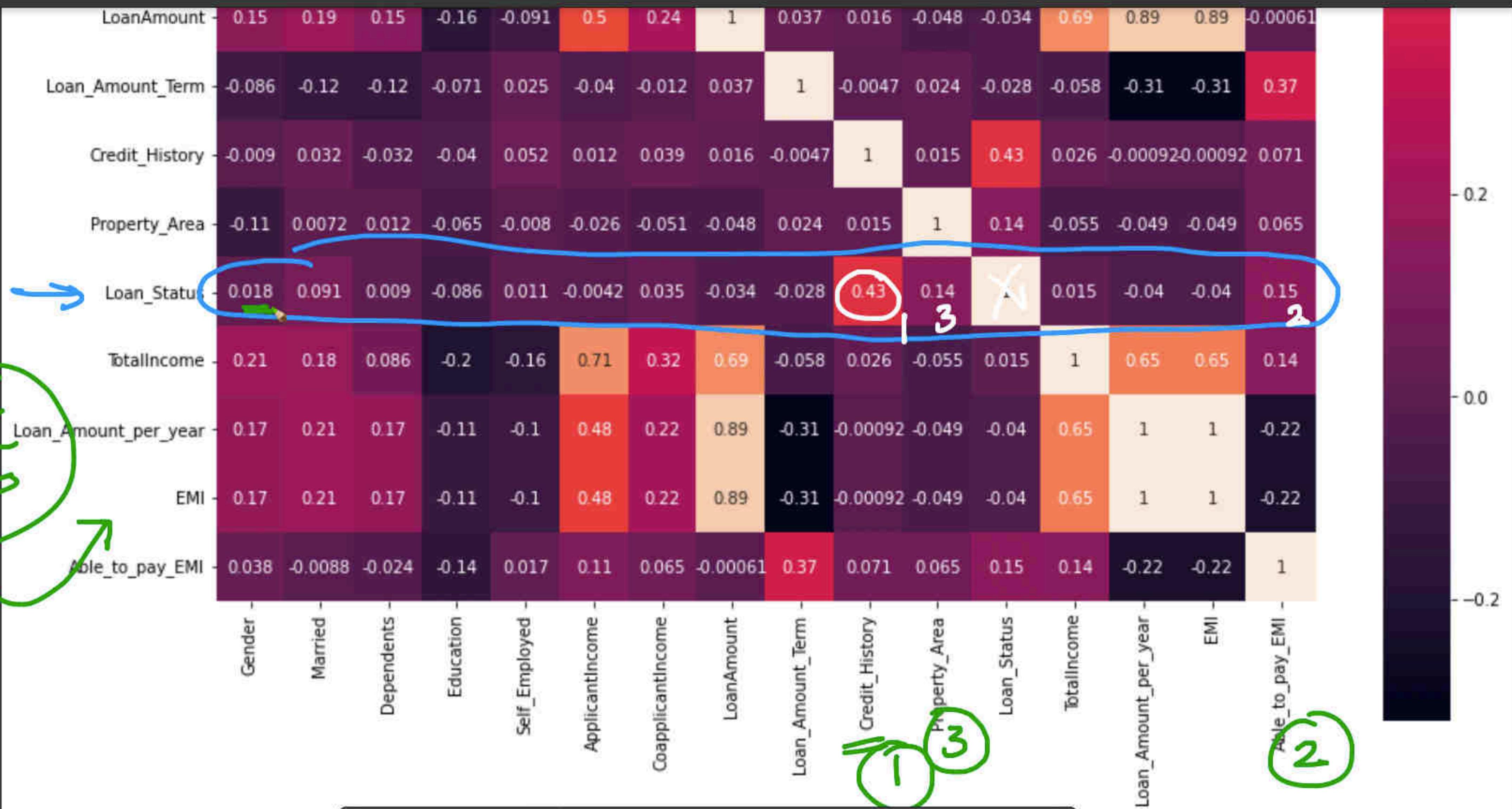
```
plt.figure(figsize=(15, 15))
sns.heatmap(data.corr(method='spearman'), square=True, annot=True)
```

```
[1]: <matplotlib.axes._subplots.AxesSubplot at 0x7f3e2f27f950>
```



+ Code + Tex

## Reconnect



EDA\_FE.ipynb - Colaboratory    X    plot log(x) - Google Search    X

[colab.research.google.com/drive/1\\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=0HZa6gelV](https://colab.research.google.com/drive/1_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=0HZa6gelV)

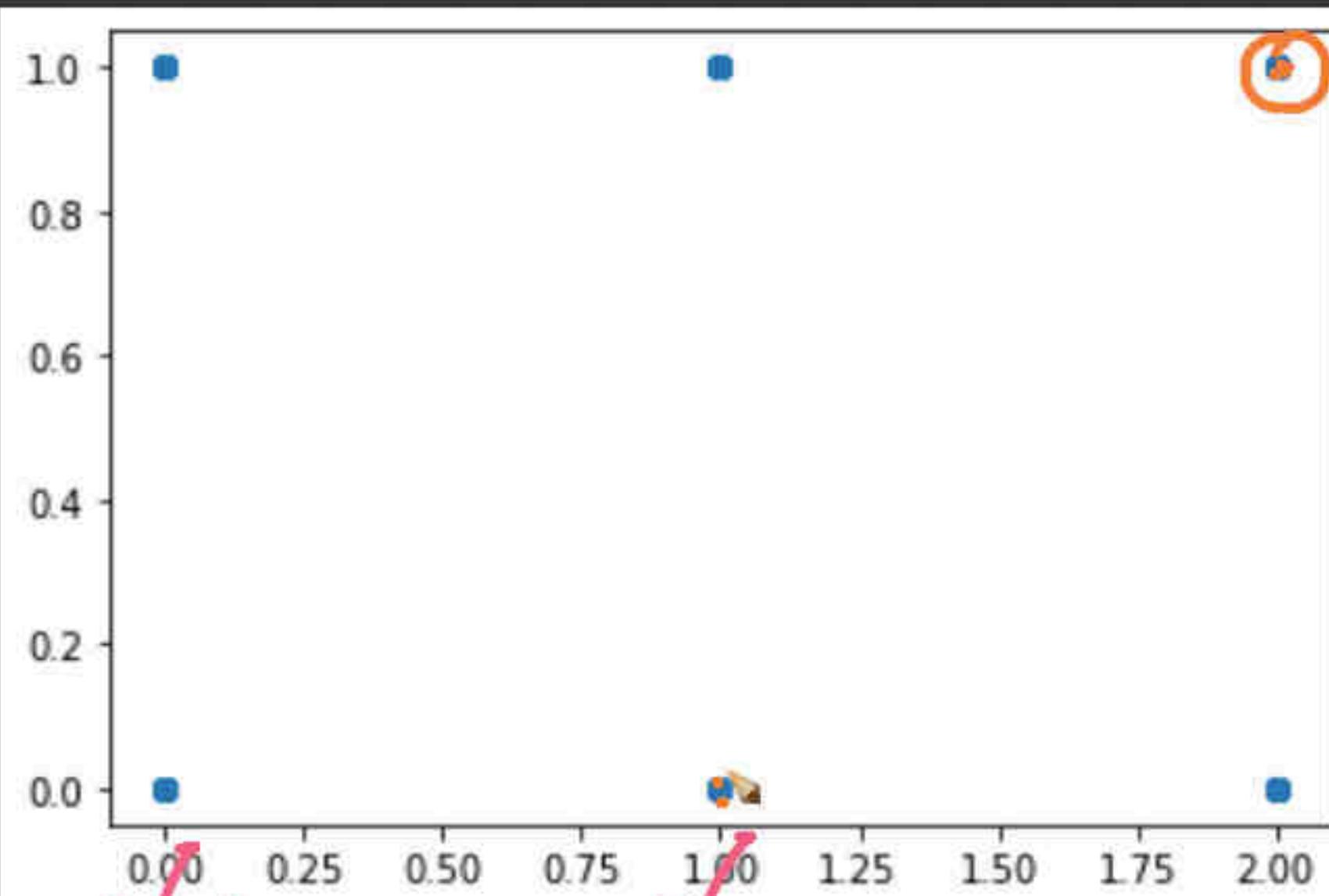
Q 内 ☆

+ Code + Text

Reconnect ▾

1

```
[ ] plt.scatter(data['Credit_History'], data['Loan_Status'])
plt.show()
#sometimes scatter plots can be misleading do to catgeorical nature of the data
```



0.41 ✓ PCC & SNC

```
► sns.countplot(data =data, x = 'Credit_History', hue = 'Loan_Status')
```

EDA\_FE.ipynb - Colaboratory

plot log(x) - Google Search

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=0HZa6gejVXo

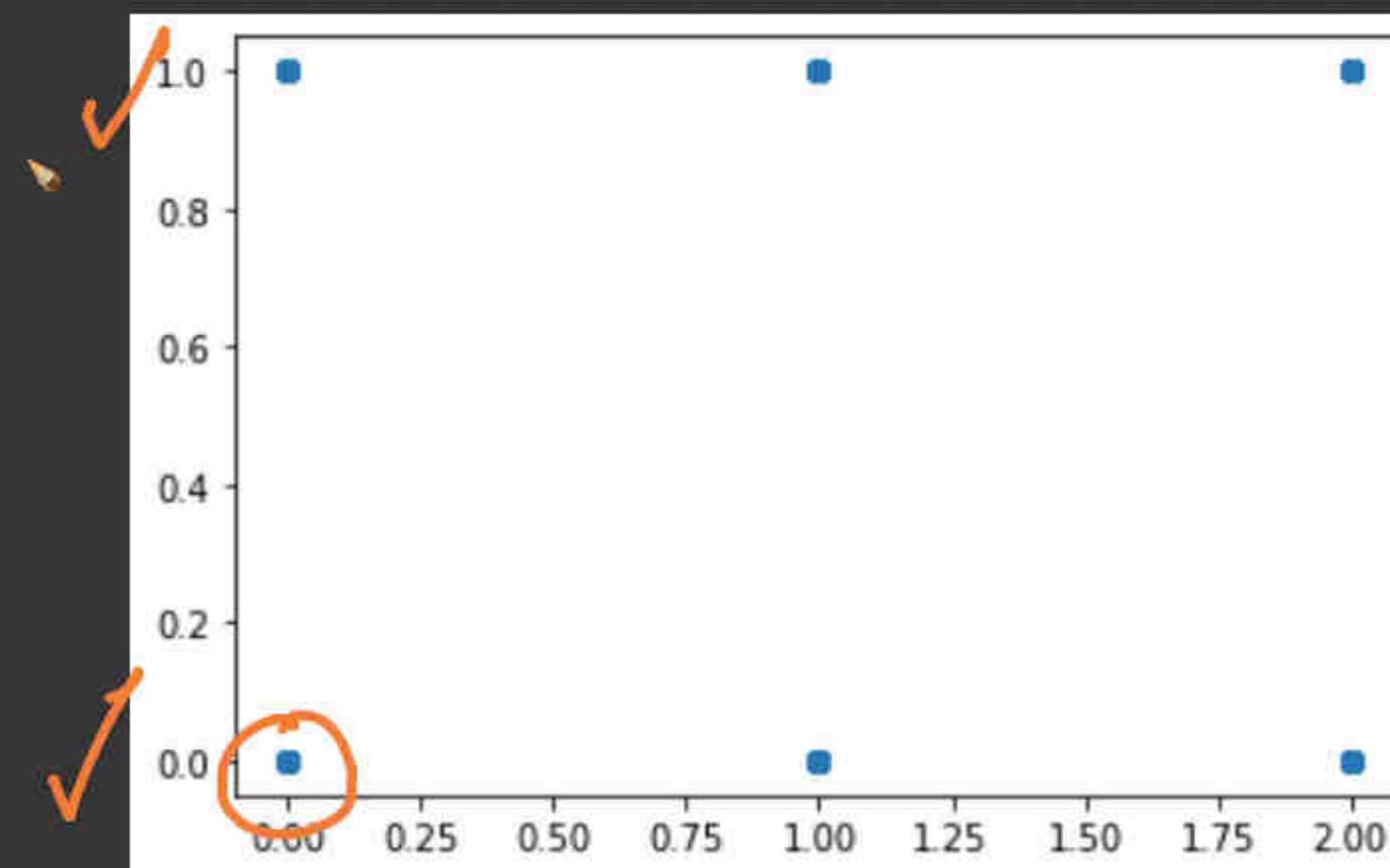


+ Code + Text

Reconnect



```
[ ] plt.scatter(data['Credit_History'], data['Loan_Status'])  
plt.show()  
#sometimes scatter plots can be misleading do to catgeorical nature of the data
```



— PCC / SPCG  
— plot by date

```
▶ sns.countplot(data = data, x = 'Credit_History', hue = 'Loan_Status')
```

```
▶ <matplotlib.axes._subplots.AxesSubplot at 0x7f3020b7d950>
```



EDA\_FE.ipynb - Colaboratory x Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=0HZa6gejVXo

+ Code + Text Reconnect

{x}

```
[ ] plt.scatter(data['Credit_History'], data['Loan_Status'])
plt.show()
#sometimes scatter plots can be misleading do to catgeorical nature of the data
```

□

<>

```
▶ sns.countplot(data = data, x = 'Credit_History', hue = 'Loan_Status')
```

EDA\_FE.ipynb - Colaboratory x Google Search x +

colab.research.google.com/drive/1\_P-nKswaKnx5v77lOx3pCxvJve73MvnN#scrollTo=0HZa6geljVXo

+ Code + Text Reconnect

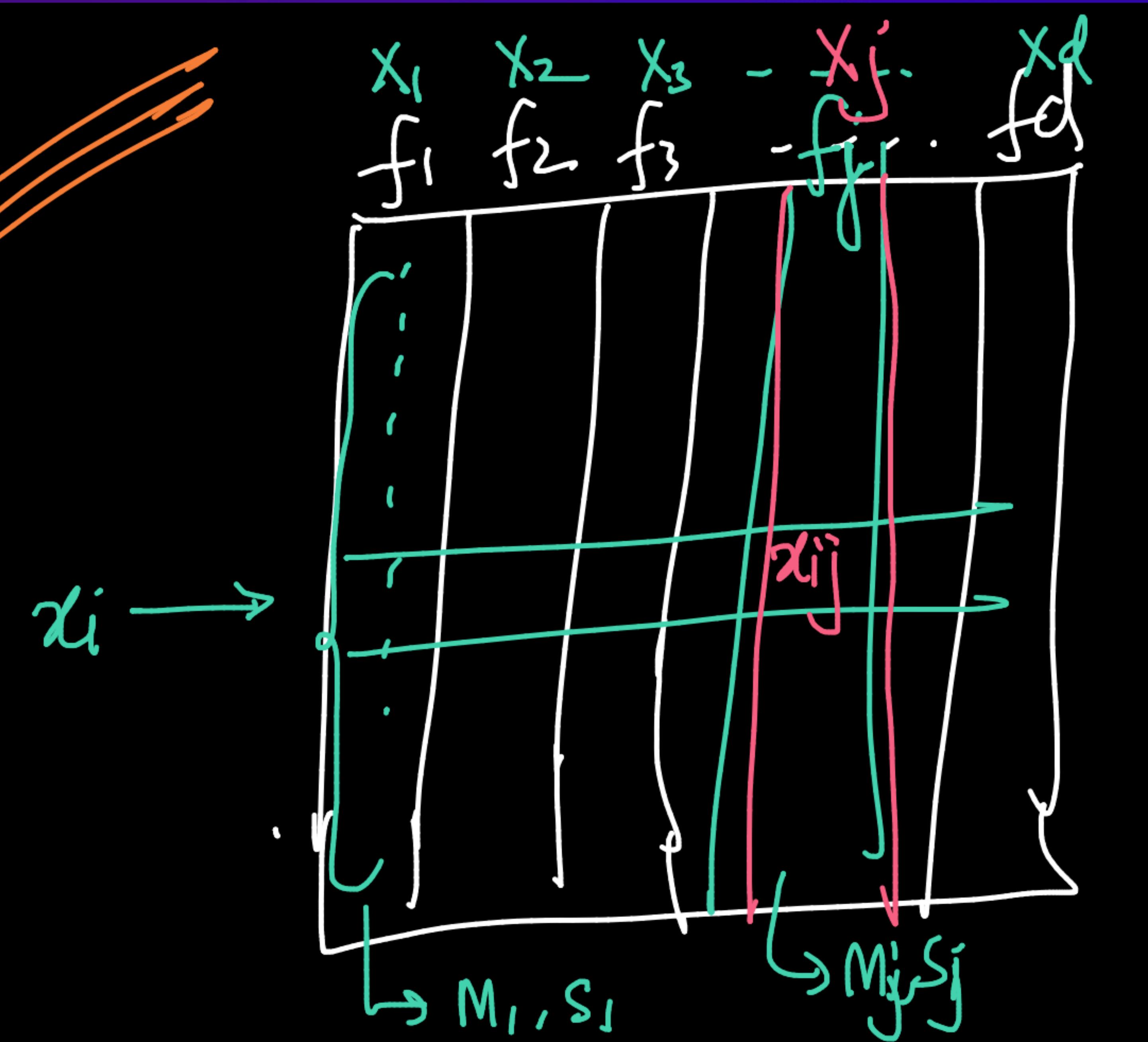
{x}

```
[ ] sns.countplot(data = data, x = 'Credit_History', hue = 'Loan_Status')
```

<matplotlib.axes.\_subplots.AxesSubplot at 0x7f3e2eb7d950>

Credit_History	Loan_Status	Count
0.0	0	~85
0.0	1	~120
1.0	0	~100
1.0	1	~370
2.0	0	~15
2.0	1	~40

Column Standardization and Normalization



$$x_i - \mu : s_i$$

Column shift

$$x_{ij} = x_{ij} - M_j$$

$$S_j$$

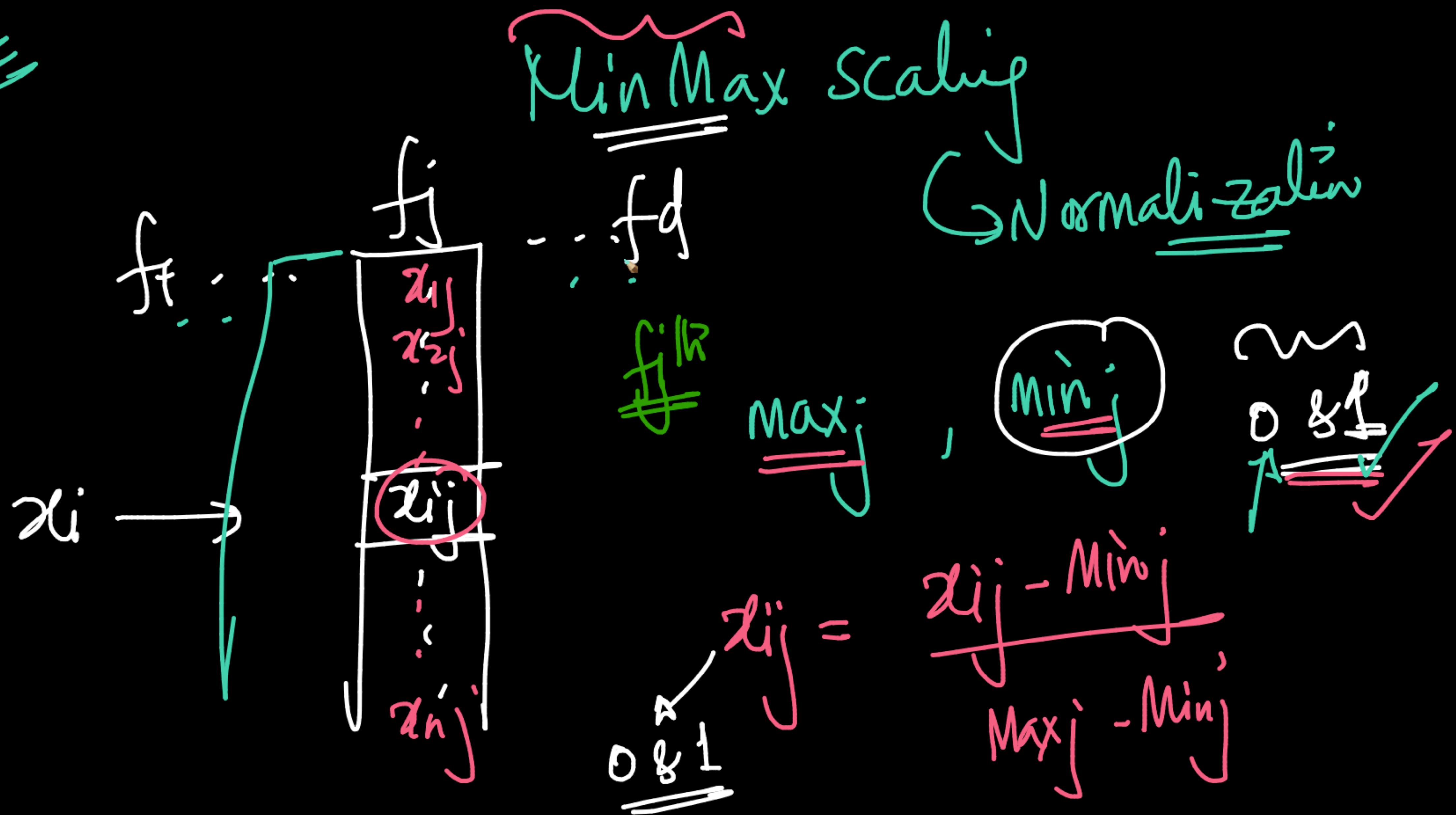
Col. std  $\rightarrow$  ML Optimization;  $F \equiv \dots$

$$\text{std. normalized-var}$$
$$x - \mu \over \sigma$$

↳ features which are in  
different scales/  
units

$$h, w$$

✓ col. std / Standard Scaling  
↓  
{ mean-centering & var-scaling per column  
=====



Min Max Scaling → not widely used in ML



→ Computer Vision  
(DL)

EDA\_FE.ipynb - Colaboratory x plot log(x) - Google Search x + colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=A-B0OnWTn1WV

+ Code + Text Reconnect

Column Standardization and Normalization

- Mean centering and Variance scaling (Standard Scaling)
- MinMax Scaling

```
[ ] from sklearn.preprocessing import StandardScaler, MinMaxScaler  
scaler = StandardScaler()  
std_data = scaler.fit_transform(data)  
std_data = pd.DataFrame(std_data, columns=data.columns)  
std_data.head()
```

m<sub>j</sub>, s<sub>j</sub> t<sub>j</sub>:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount
0	0.472343	-1.372089	-0.737806	-0.528362	-0.174052	0.072991	-0.554487	-0.211241	0.1
1	0.472343	0.728816	0.253470	-0.528362	-0.174052	-0.134412	-0.038732	-0.211241	0.1
2	0.472343	0.728816	-0.737806	-0.528362	-0.586643	-0.393747	-0.554487	-0.948996	0.1
3	0.472343	0.728816	-0.737806	1.892641	-0.174052	-0.462062	0.251980	-0.306435	0.1
4	0.472343	-1.372089	-0.737806	-0.528362	-0.174052	0.097728	-0.554487	-0.056551	0.1

EDA\_FE.ipynb - Colaboratory

plot log(x) - Google Search

colab.research.google.com/drive/1\_P-nKswaKnx5v77IOx3pCxvJve73MvnN#scrollTo=A-B0OnWTn1WV

+ Code + Text

Reconnect



## Column Standardization and Normalization

- Mean centering and Variance scaling (Standard Scaling)
- MinMax Scaling

```
[ ] from sklearn.preprocessing import StandardScaler, MinMaxScaler  
  
scaler = StandardScaler()  
std_data = scaler.fit_transform(data)  
std_data = pd.DataFrame(std_data, columns=data.columns)  
std_data.head()
```

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount
0	0.472343	-1.372089	-0.737806	-0.528362	-0.174052	0.072991	-0.554487	-0.211241	0.1
1	0.472343	0.728816	0.253470	-0.528362	-0.174052	-0.134412	-0.038732	-0.211241	0.1
2	0.472343	0.728816	-0.737806	-0.528362	-0.586643	-0.393747	-0.554487	-0.948996	0.1
3	0.472343	0.728816	-0.737806	1.892641	-0.174052	-0.462062	0.251980	-0.306435	0.1
4	0.472343	-1.372089	-0.737806	-0.528362	-0.174052	0.097728	-0.554487	-0.056551	0.1

any-dis

$$\cancel{X} = X - \mu$$

$\mu$

$\mathcal{S}$

STD:

↳ mean = 0  
STD = 1  
for each column

**MOST**

NORM

↳  $0 - 1$  for each column

v-few  
~~CV~~  
CV

ML  
Out of context

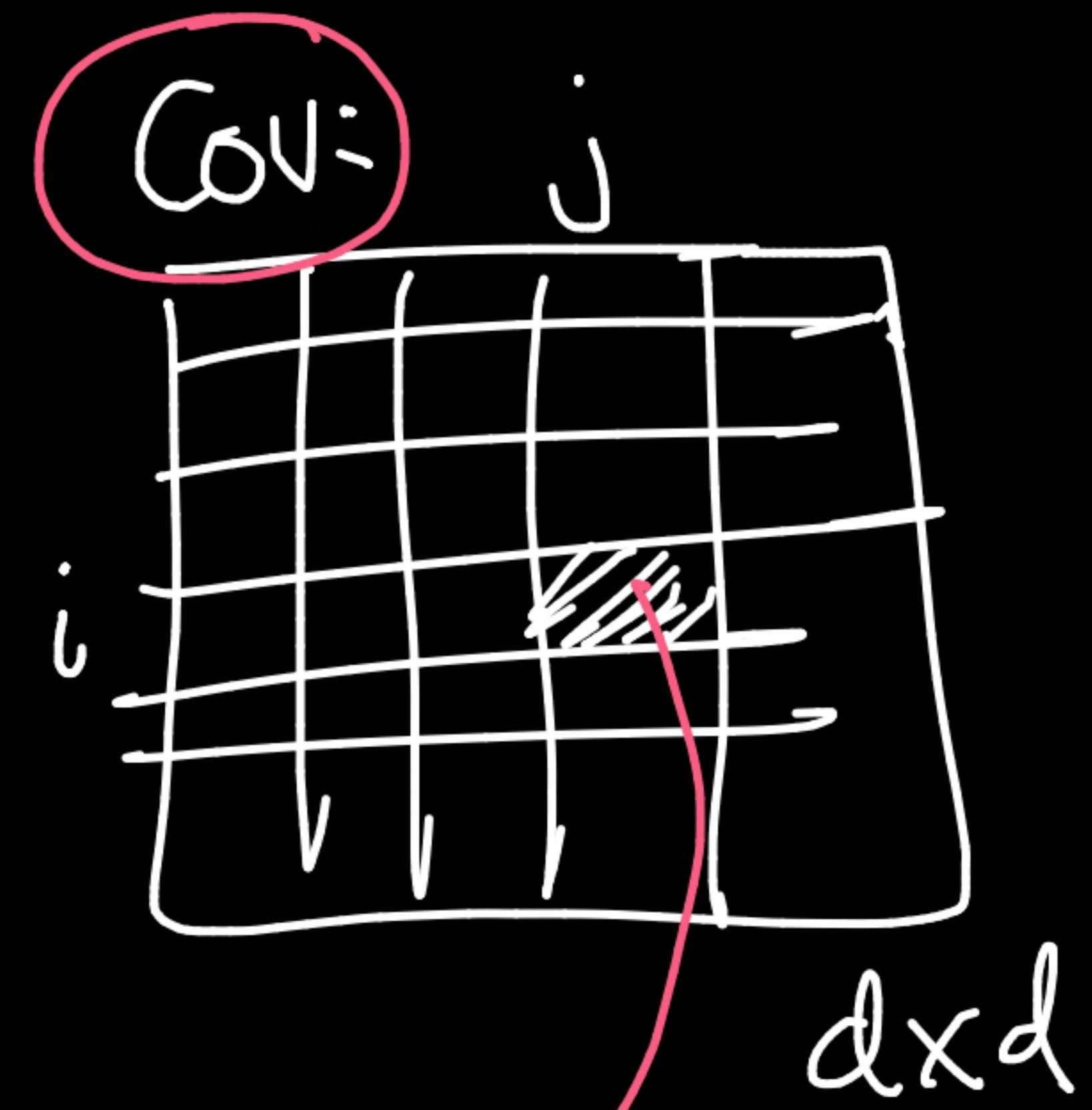
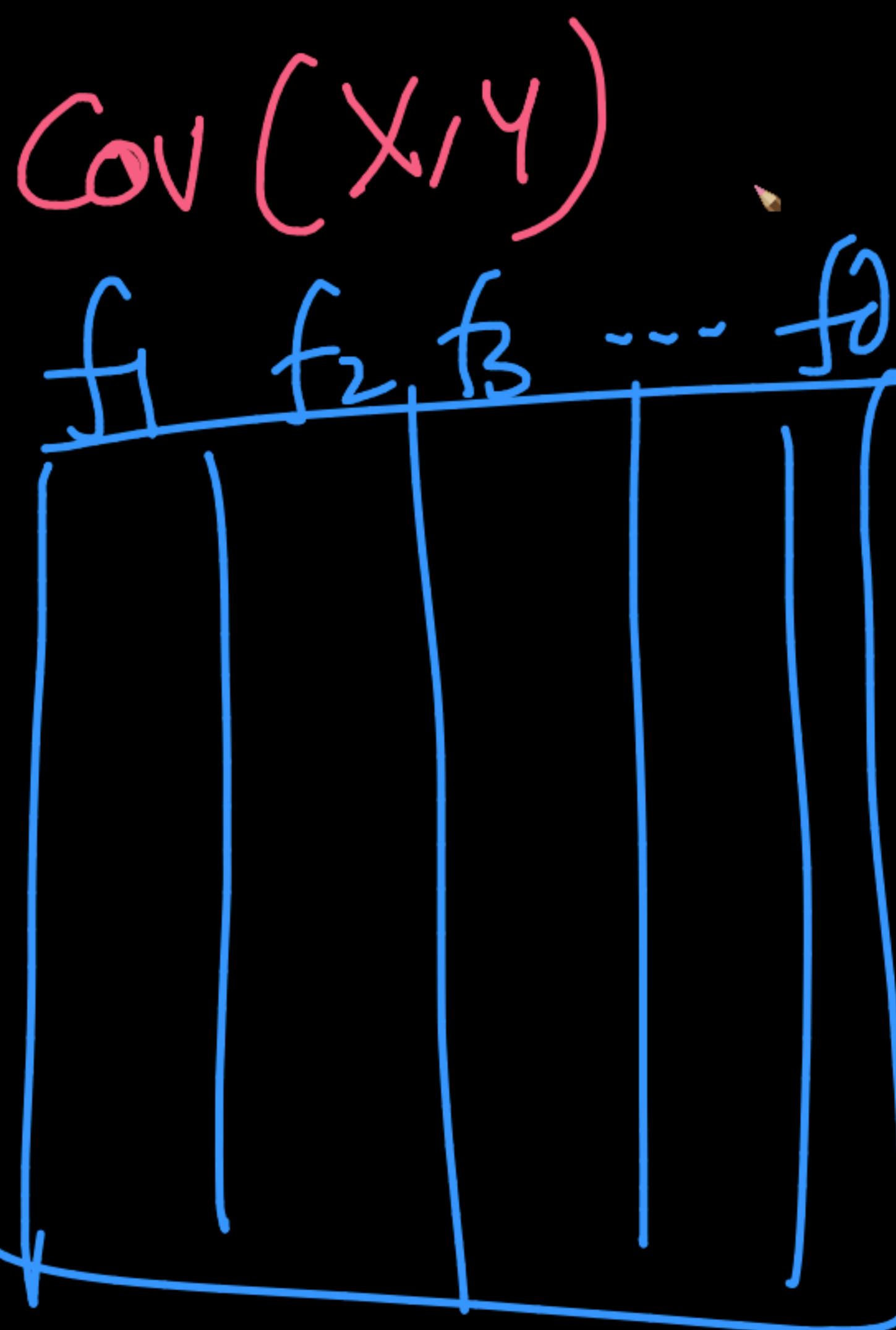
fi fi  
Cor↑

9

Multicollinearity  
↳ Br-βef

Assessments ✓

✓ Case studies → solve reasonably well  
for wr<sup>2</sup> ✓

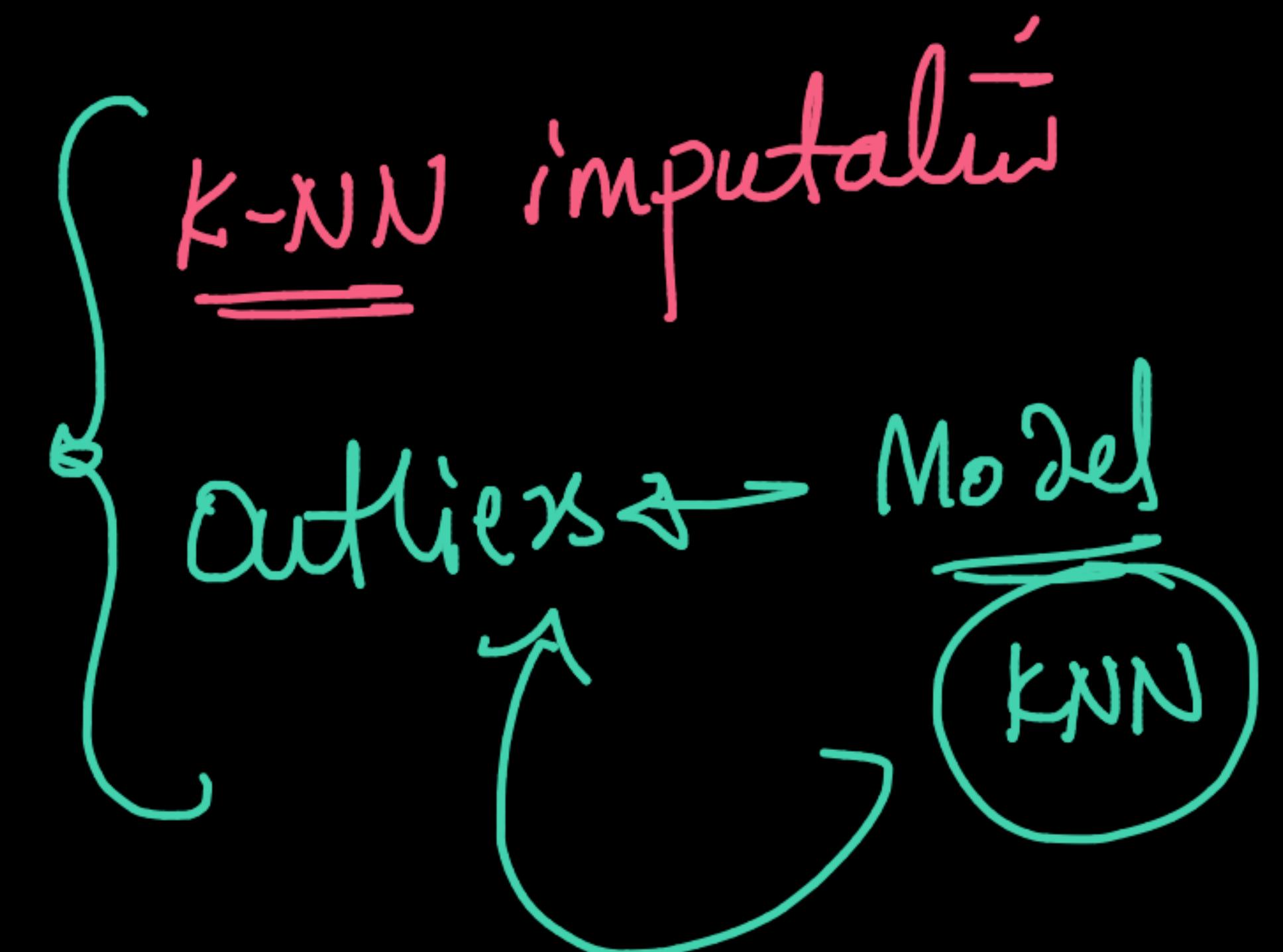


$\text{Cov}(f_i, f_j)$

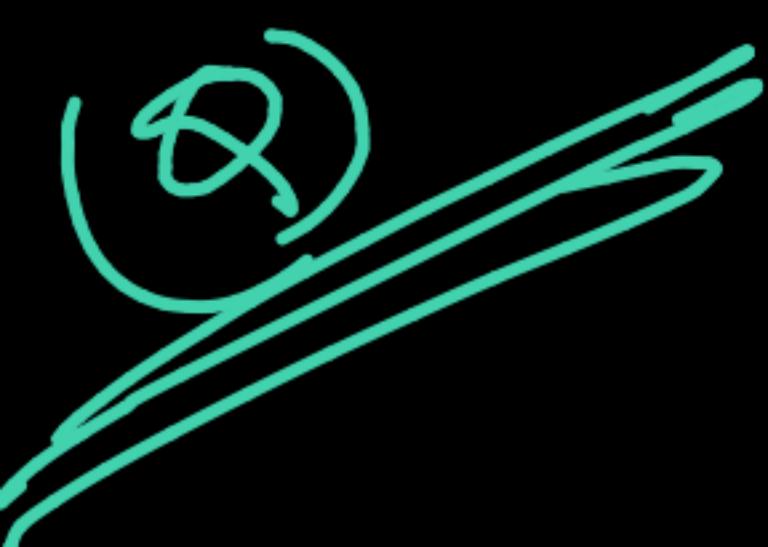
Red lines

RANSAC

Denoising AutoEncoders

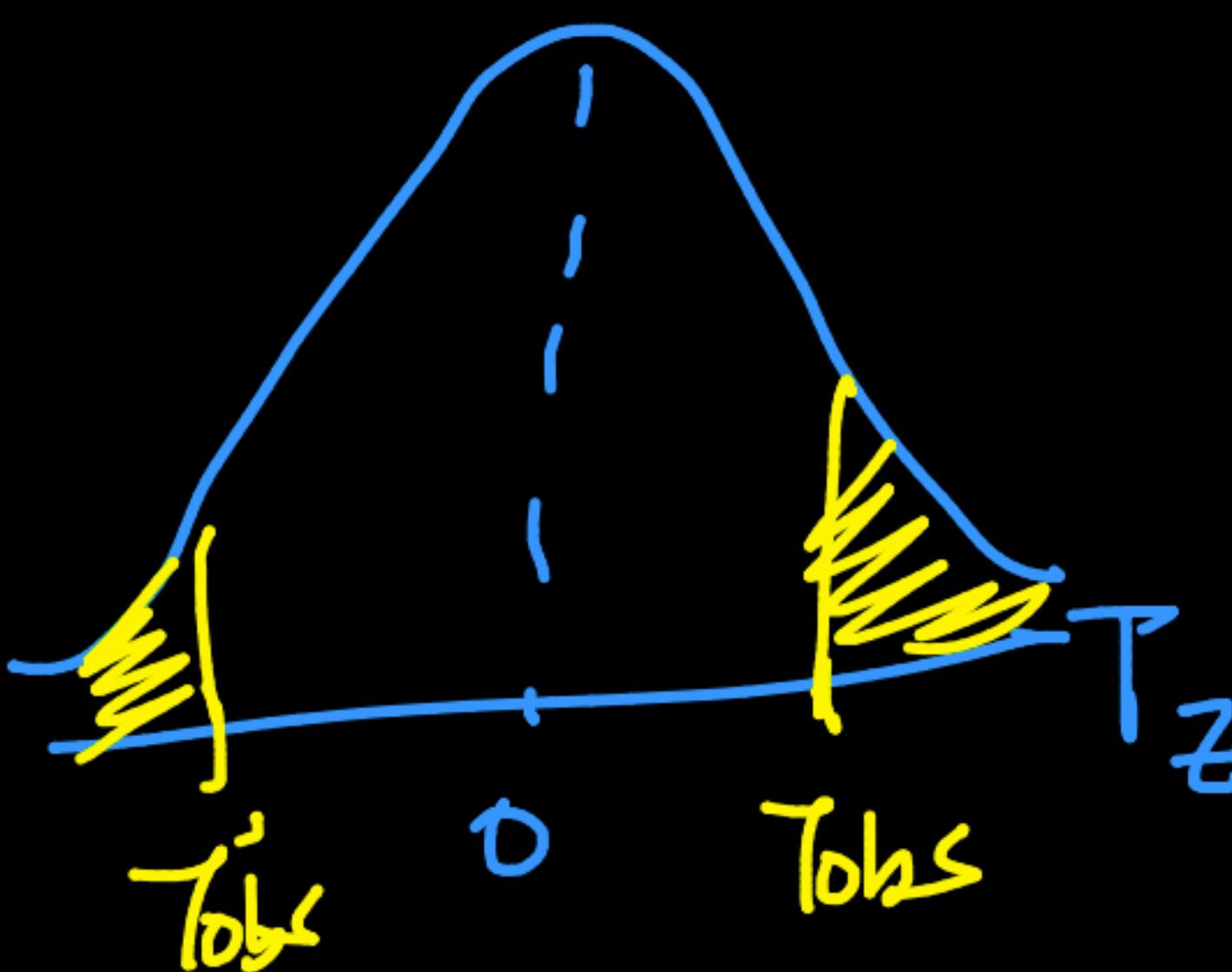


TTS or MWF



2-tailed z test

$$T_{obs} = \frac{M_1 - M_2}{\sqrt{\dots}} \text{ or } \frac{M_2 - M_1}{\sqrt{\dots}} = T_{obs}$$

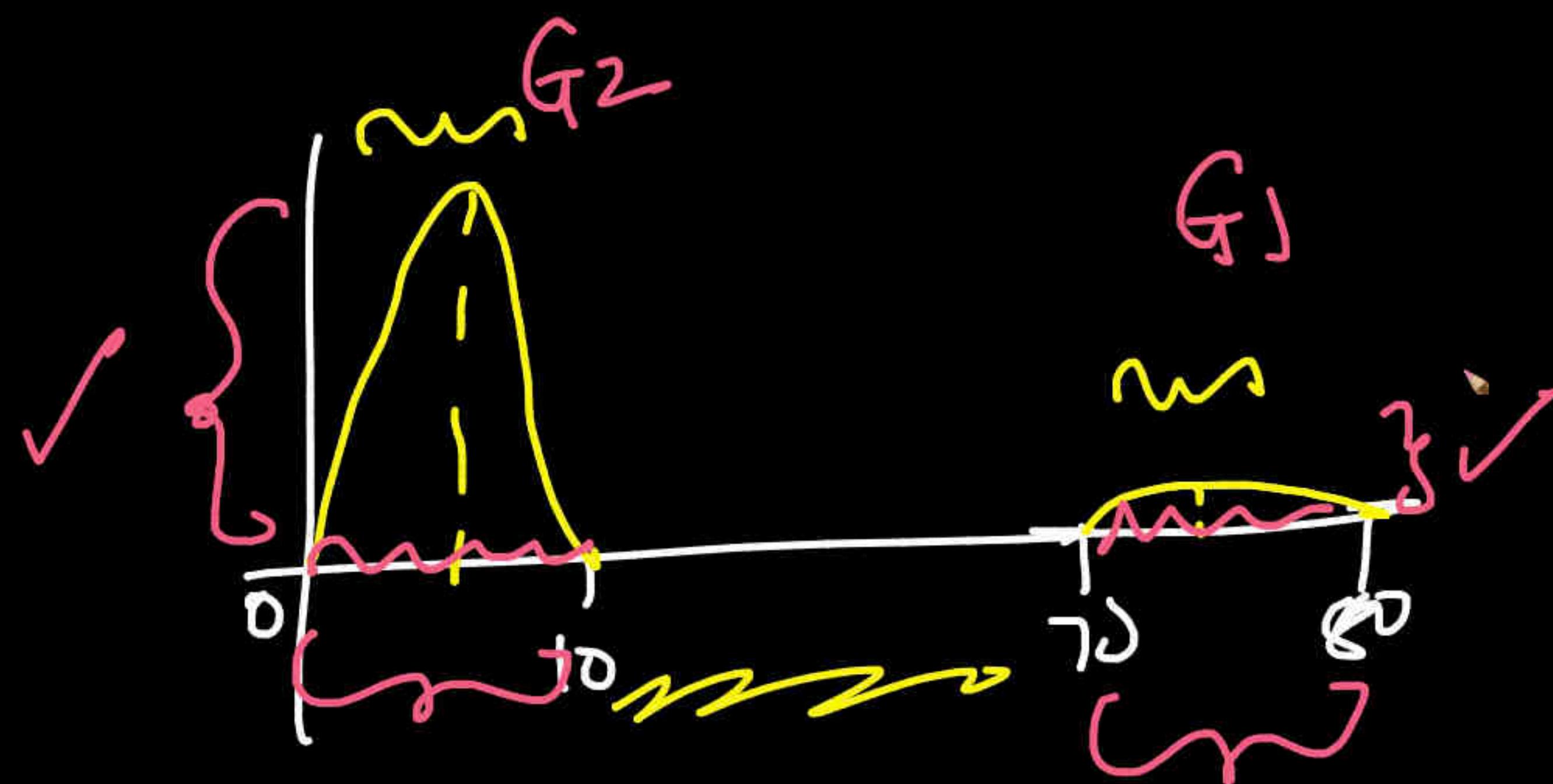


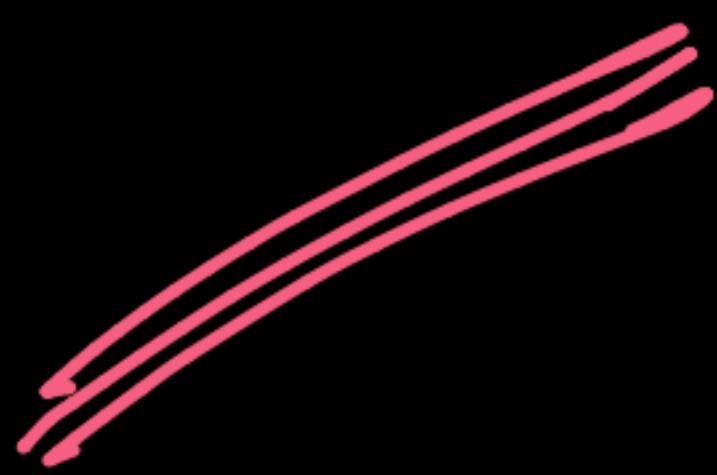
p-value: 2-tails



99% — (0 - 10)

1% — (70 - 80)





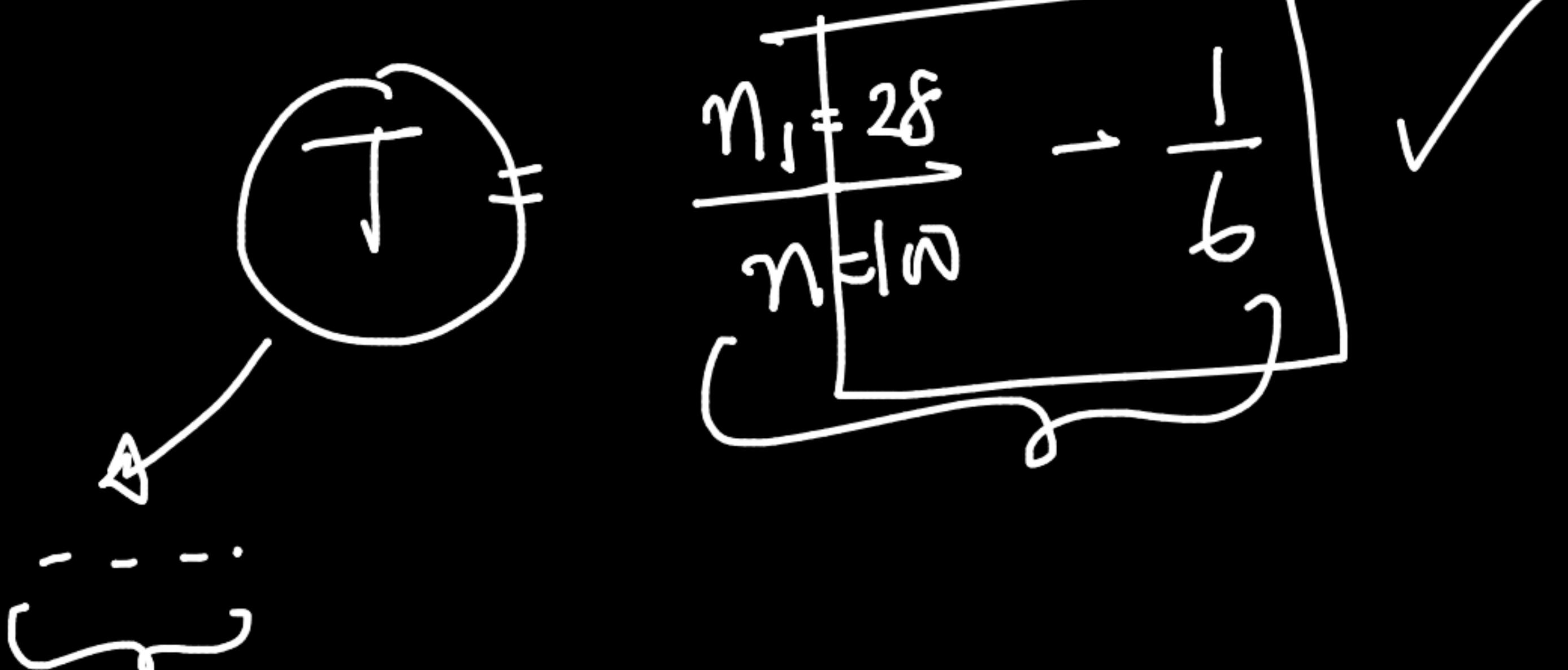
$$\rightarrow \frac{1}{6}$$

$H_0$ : not biased

dice

$\boxed{1}$  vs not one

$H_a$ : biased trials  $\neq$



SQL

...

Math fw ML



WIKIPEDIA  
The Free Encyclopedia

Main page  
Contents  
Current events  
Random article  
About Wikipedia  
Contact us  
Donate

Contribute

Help  
Learn to edit  
Community portal  
Recent changes  
Upload file

Tools

What links here  
Related changes  
Special pages  
Permanent link  
Page information

Article Talk

Read

Edit

View history

Search Wikipedia



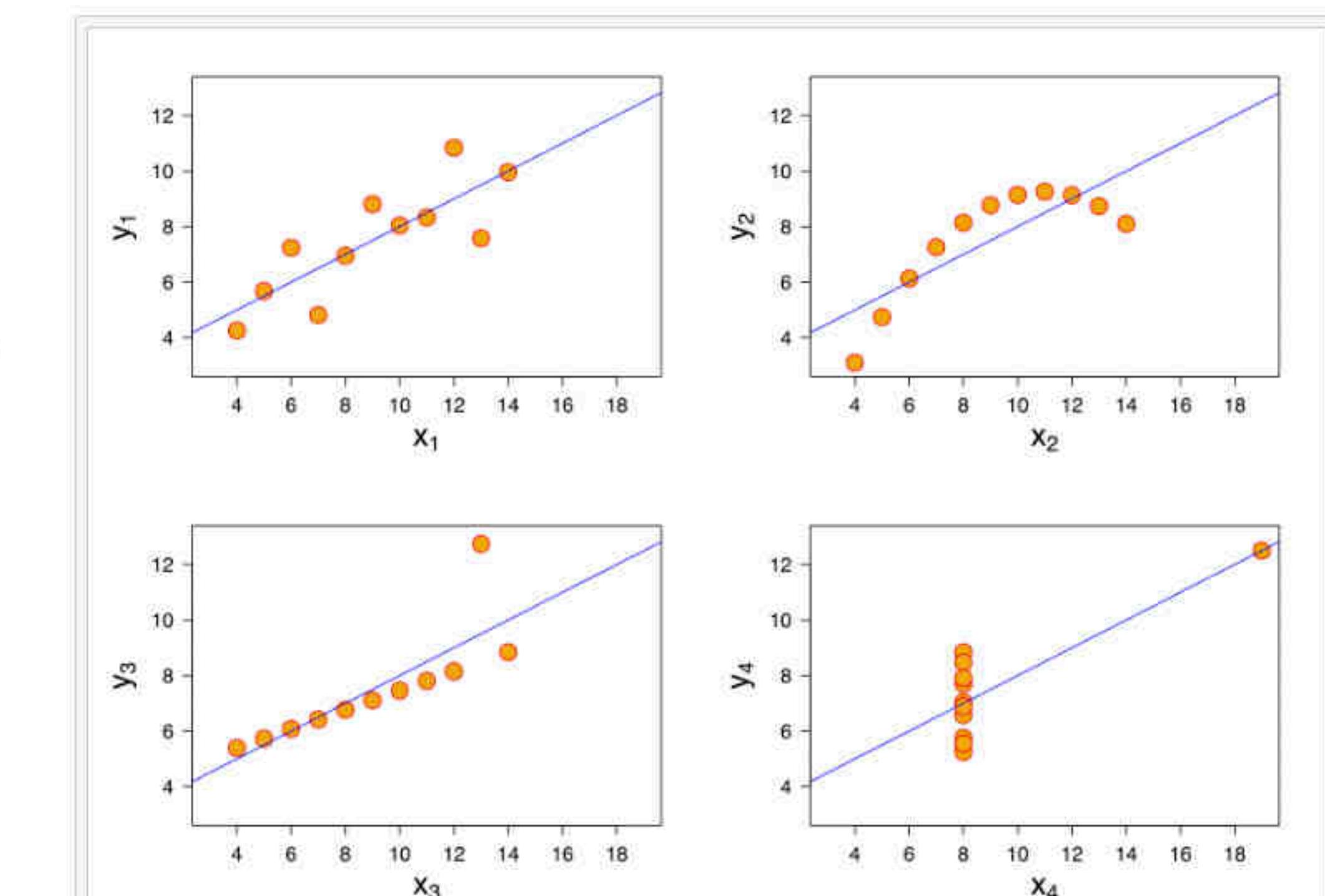
# Anscombe's quartet

From Wikipedia, the free encyclopedia

**Anscombe's quartet** comprises four [data sets](#) that have nearly identical simple [descriptive statistics](#), yet have very different [distributions](#) and appear very different when [graphed](#). Each dataset consists of eleven  $(x,y)$  points. They were constructed in 1973 by the statistician [Francis Anscombe](#) to demonstrate both the importance of graphing data when analyzing it, and the effect of [outliers](#) and other [influential observations](#) on statistical properties. He described the article as being intended to counter the impression among statisticians that "numerical calculations are exact, but graphs are rough."<sup>[1]</sup>

## Contents [hide]

- 1 Data
- 2 See also
- 3 References
- 4 External links



All four sets are identical when examined using simple summary statistics, but differ considerably when graphed



120 / 120

Data [edit]