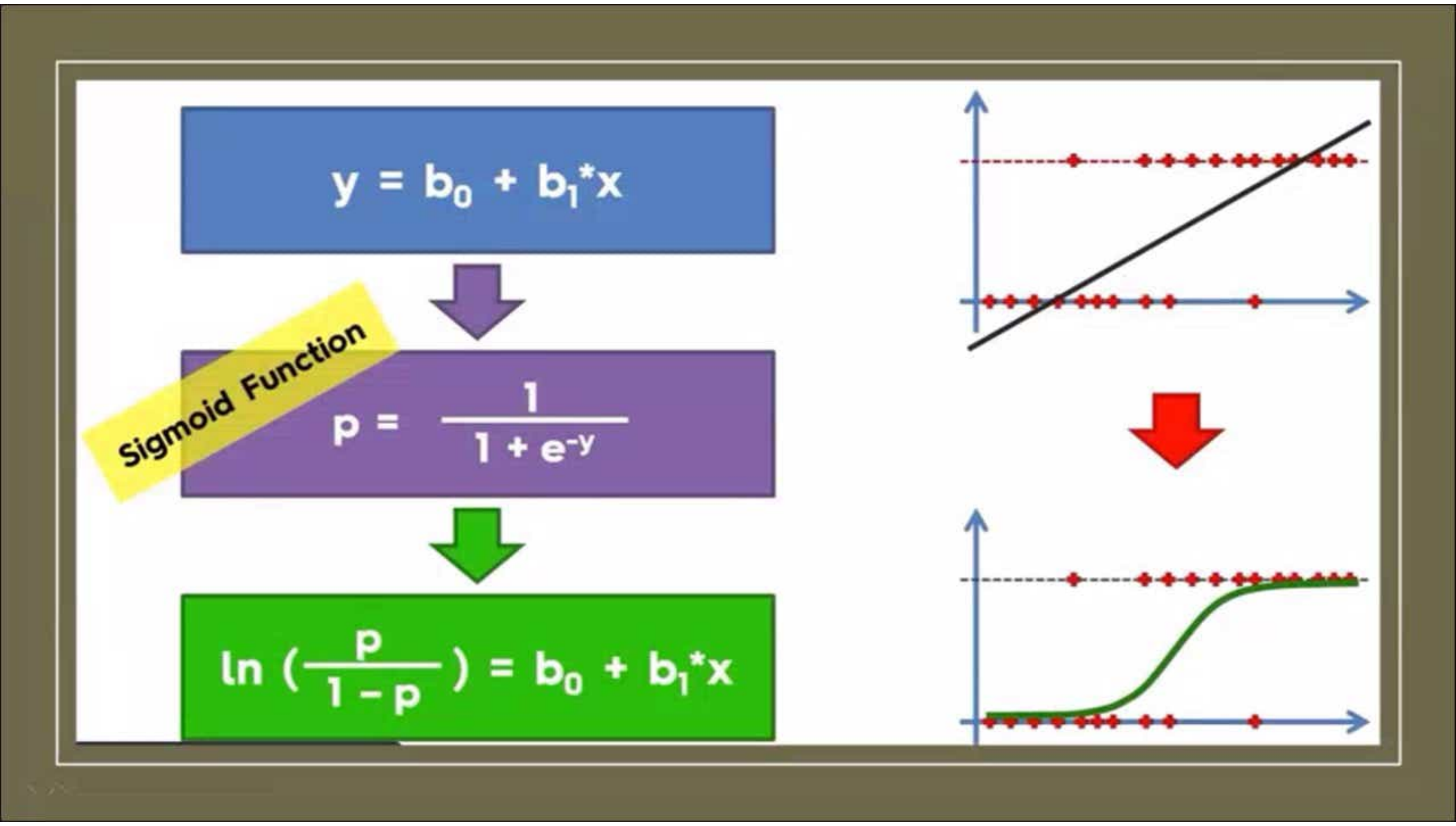
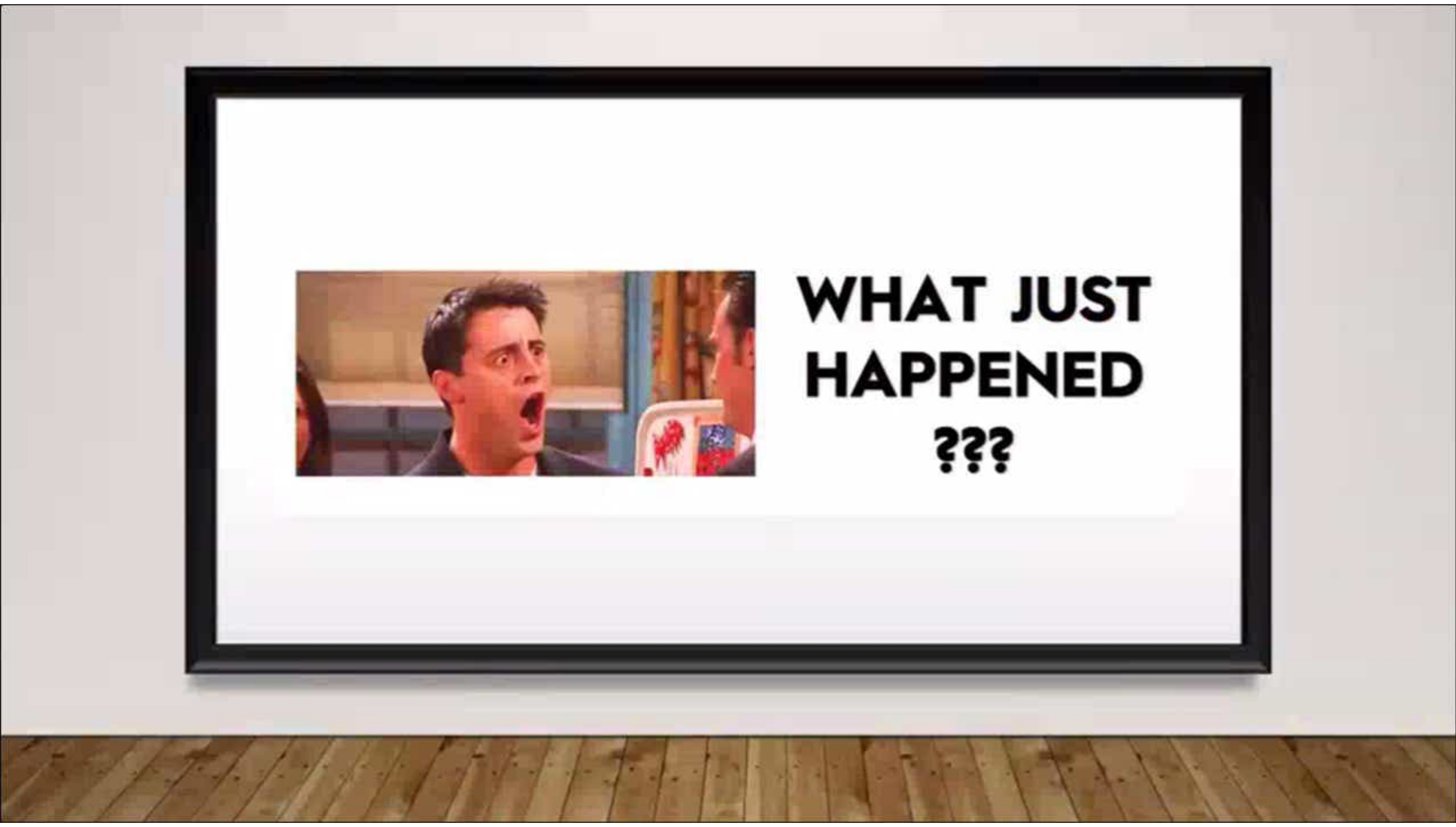
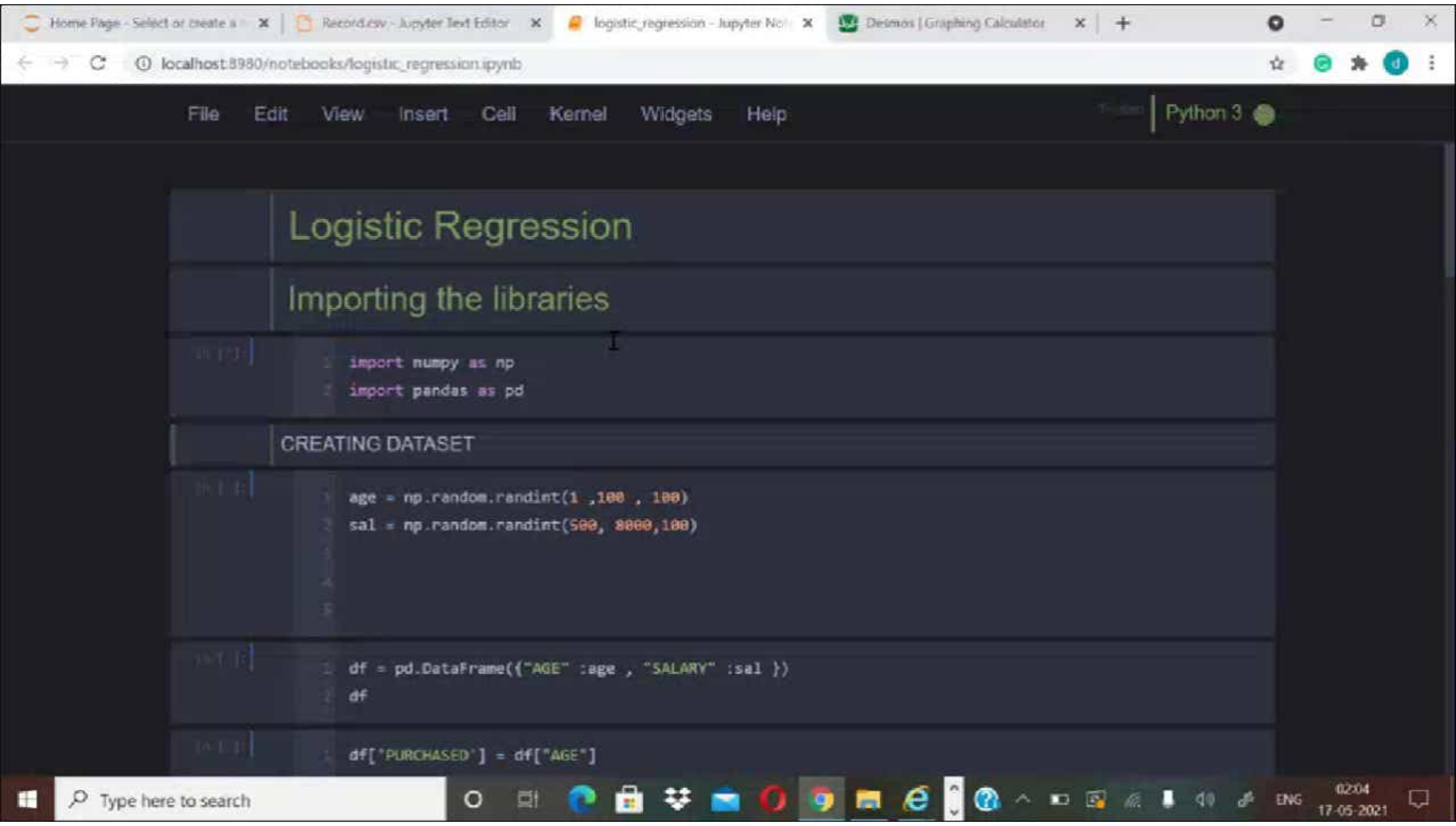

LOGISTIC
REGRESSION
0:11 / 31:19


$y = b_0 + b_1 * x$
Sigmoid Function
$p = \frac{1}{1 + e^{-y}}$
$\ln\left(\frac{p}{1-p}\right) = b_0 + b_1 * x$


WHAT JUST
HAPPENED
???



Home Page - Select or create a
Record.csv - Jupyter Text Editor
logistic_regression - Jupyter No
Desmos | Graphing Calculator
localhost:8980/notebooks/logistic_regression.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python 3
Logistic Regression
Importing the libraries
import numpy as np
import pandas as pd
CREATING DATASET
age = np.random.randint(1 ,100 , 100)
sal = np.random.randint(500, 8000,100)
df = pd.DataFrame({"AGE" :age , "SALARY" :sal })
df
df['PURCHASED'] = df["AGE"]
Type here to search
ENG
02:04
17-05-2021

Home Page - Select or create a | Record.csv - Jupyter Text Editor | logistic_regression - Jupyter Note | Desmos | Graphing Calculator

localhost:8980/notebooks/logistic_regression.ipynb

File Edit View Insert Cell Kernel Widgets Help Python 3

CREATING DATASET

```
age = np.random.randint(1 ,100 , 100)
sal = np.random.randint(500, 8000,100)
```

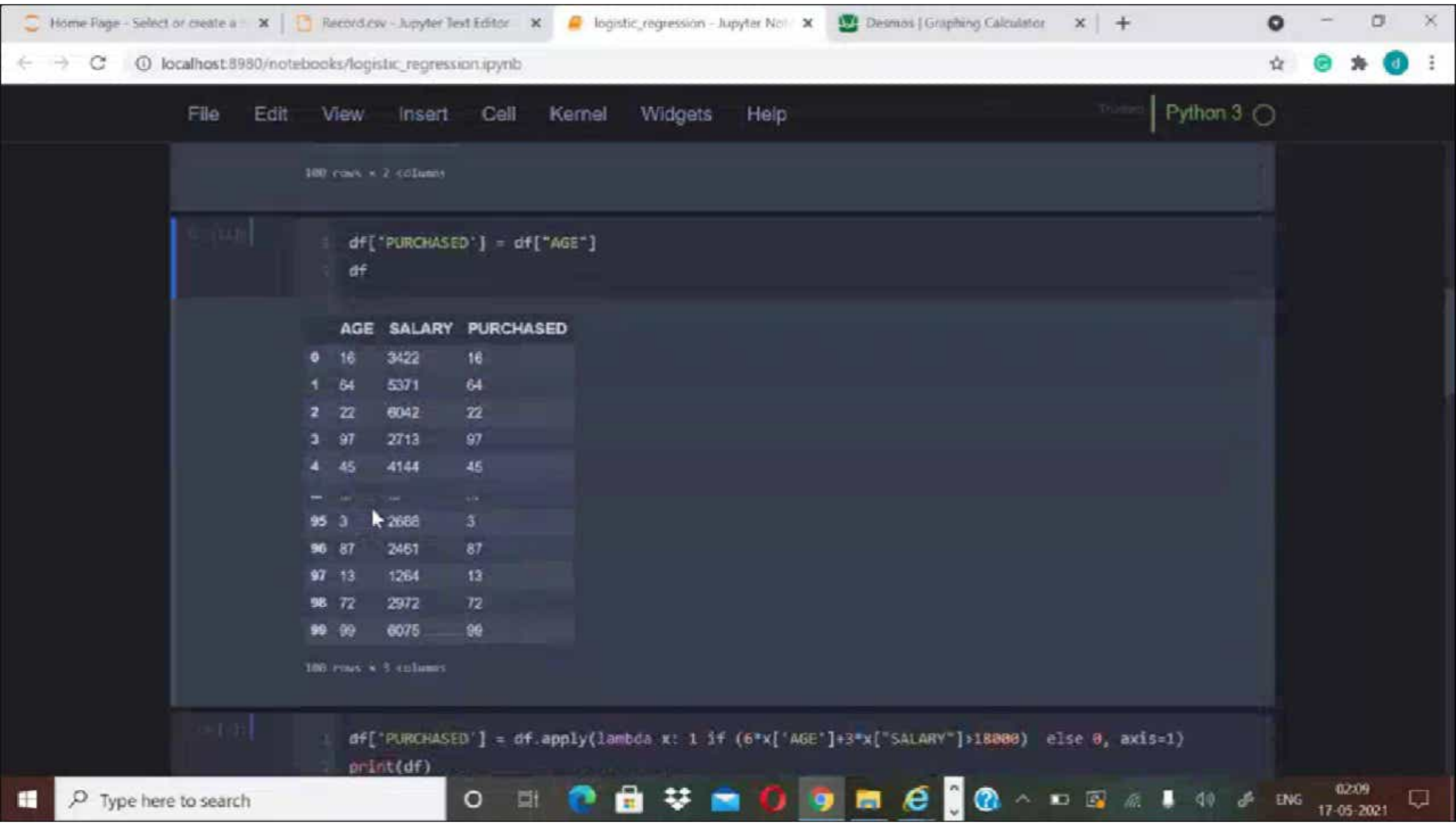
```
df = pd.DataFrame({"AGE" :age , "SALARY" :sal })
df
```

	AGE	SALARY
0	16	3422
1	64	5371
2	22	6042
3	97	2713
4	45	4144
...	...	...
95	3	2688
96	87	2461
97	13	1264
98	72	2972
99	99	6075

100 rows × 2 columns

Type here to search

ENG 02:07 17-05-2021


Home Page - Select or create a
Record.csv - Jupyter Text Editor
logistic_regression - Jupyter Not
Desmos | Graphing Calculator
localhost:8980/notebooks/logistic_regression.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python 3
df['PURCHASED'] = df["AGE"]
df
AGE SALARY PURCHASED
0 16 3422 16
1 64 5371 64
2 22 6042 22
3 97 2713 97
4 45 4144 45
95 3 2686 3
96 87 2461 87
97 13 1264 13
98 72 2972 72
99 99 6075 99
df['PURCHASED'] = df.apply(lambda x: 1 if (6*x['AGE']+3*x["SALARY"]>18000) else 0, axis=1)
print(df)
Type here to search
ENG
02:09
17-05-2021

96	87	2461	87
97	13	1264	13
98	72	2972	72
99	99	6075	99

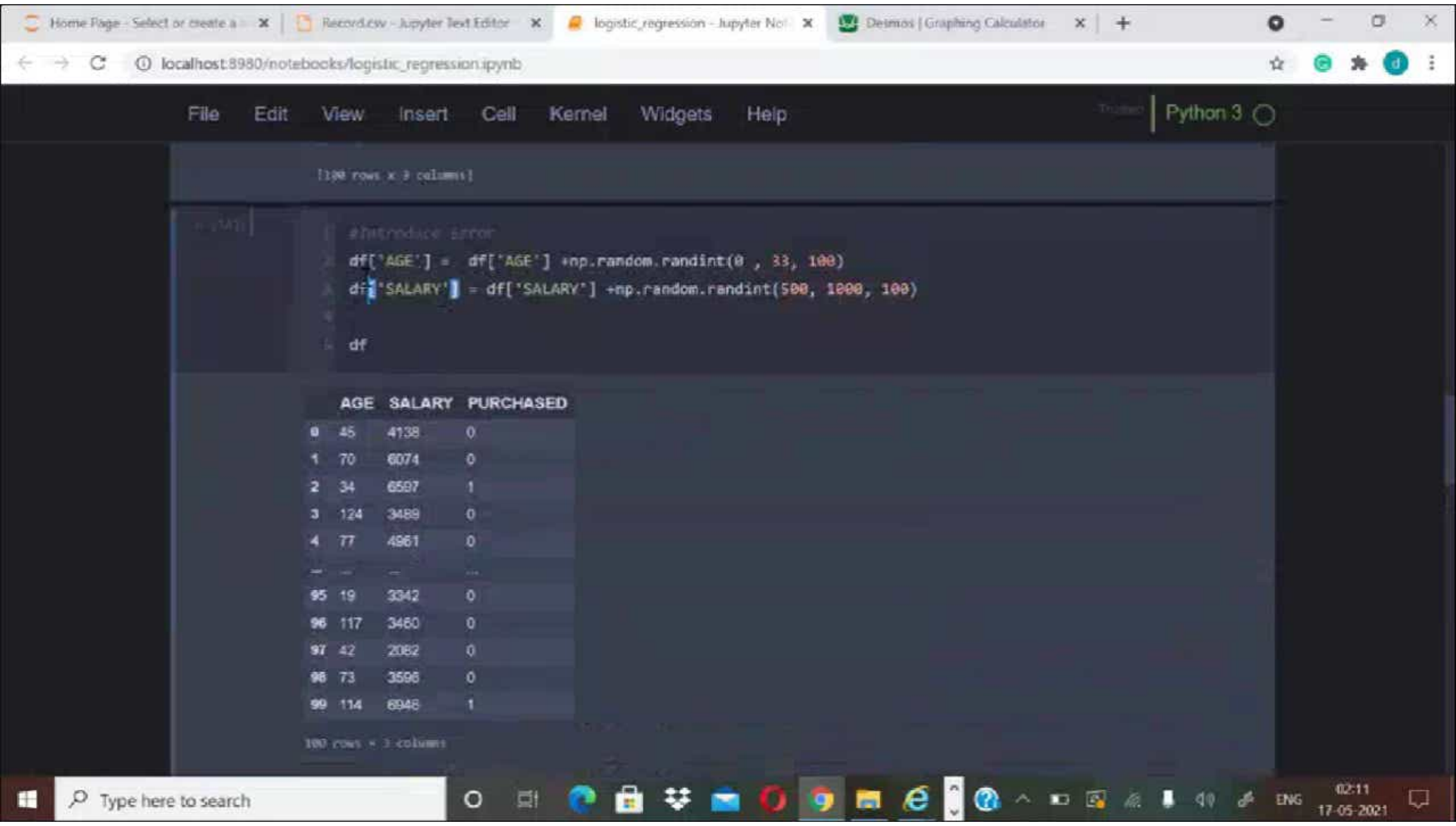
100 rows × 3 columns

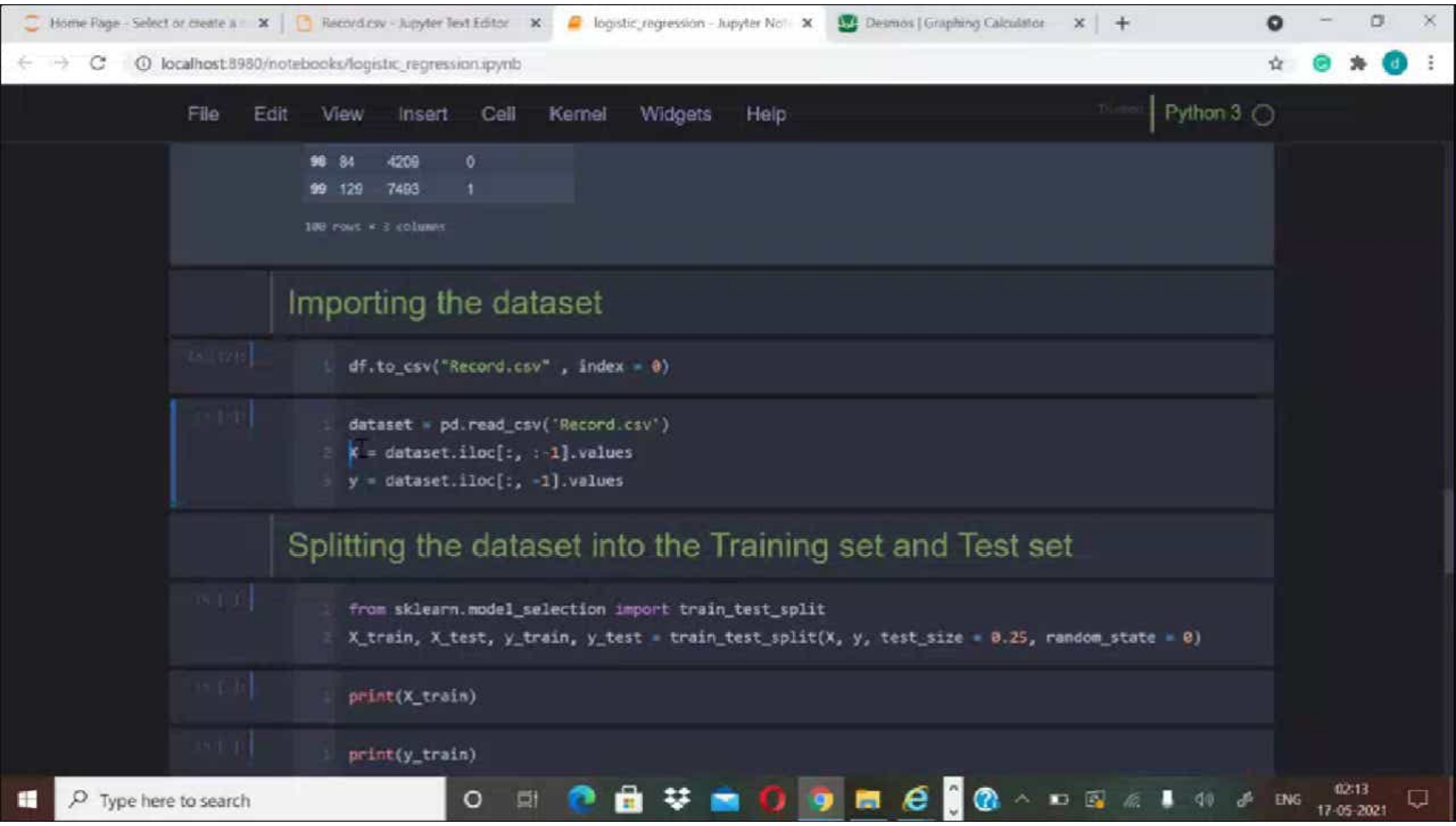
```python
df['PURCHASED'] = df.apply(lambda x: 1 if (6*x['AGE']+3*x["SALARY"]>18000)  else 0, axis=1)
print(df)
```

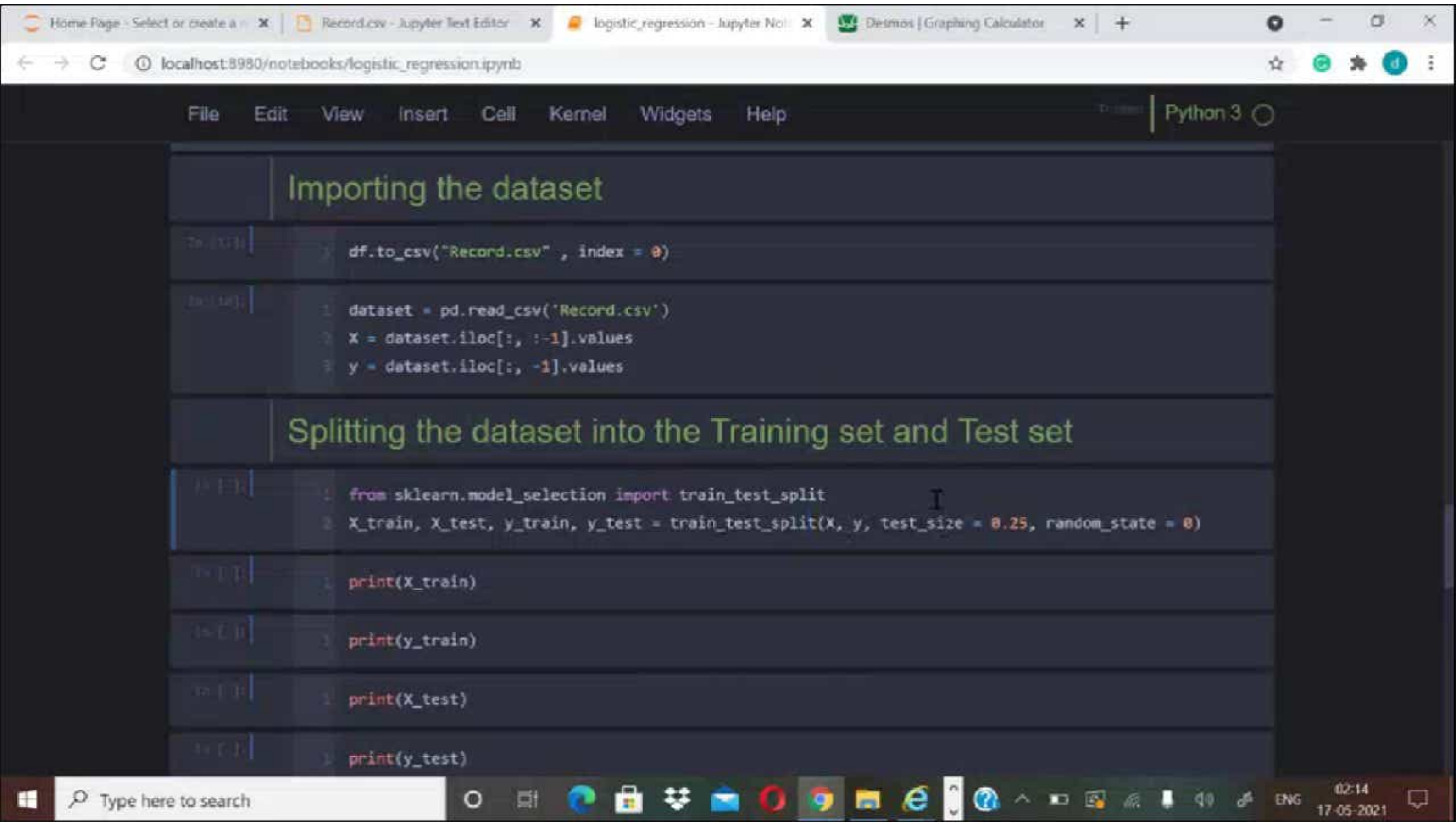
```
    AGE  SALARY  PURCHASED
0    16    3422          0
1    64    5371          0
2    22    6042          1
3    97    2713          0
4    45    4144          0
..  ...     ...        ...
95    3    2688          0
96   87    2461          0
97   13    1264          0
98   72    2972          0
99   99    6075          1

[100 rows x 3 columns]
```

```python
#Introduce Error
```


Home Page - Select or create a
Record.csv - Jupyter Text Editor
logistic_regression - Jupyter Not
Desmos | Graphing Calculator
localhost:8980/notebooks/logistic_regression.ipynb
File Edit View Insert Cell Kernel Widgets Help
Python 3
[100 rows x 3 columns]
#introduce error
df['AGE'] = df['AGE'] +np.random.randint(0 , 33, 100)
df['SALARY'] = df['SALARY'] +np.random.randint(500, 1000, 100)
df
AGE SALARY PURCHASED
0 45 4138 0
1 70 6074 0
2 34 6597 1
3 124 3489 0
4 77 4961 0
... ...
95 19 3342 0
96 117 3460 0
97 42 2082 0
98 73 3596 0
99 114 6946 1
100 rows × 3 columns
Type here to search
ENG
02:11
17-05-2021


Home Page - Select or create a
Record.csv - Jupyter Text Editor
logistic_regression - Jupyter Not
Desmos | Graphing Calculator
localhost:8980/notebooks/logistic_regression.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python 3
98 84 4209 0
99 129 7493 1
Importing the dataset
df.to_csv("Record.csv" , index = 0)
dataset = pd.read_csv('Record.csv')
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
Splitting the dataset into the Training set and Test set
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 = 0)
print(X_train)
print(y_train)
Type here to search
ENG
02:13
17-05-2021


localhost:8980/notebooks/logistic_regression.ipynb
File Edit View Insert Cell Kernel Widgets Help
Python 3
Importing the dataset
df.to_csv("Record.csv" , index = 0)
dataset = pd.read_csv('Record.csv')
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
Splitting the dataset into the Training set and Test set
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 = 0)
print(X_train)
print(y_train)
print(X_test)
print(y_test)
Type here to search
02:14
17-05-2021

FEATURE SCALING

DAMAN VIRDI

WHY FEATURE SCALING?

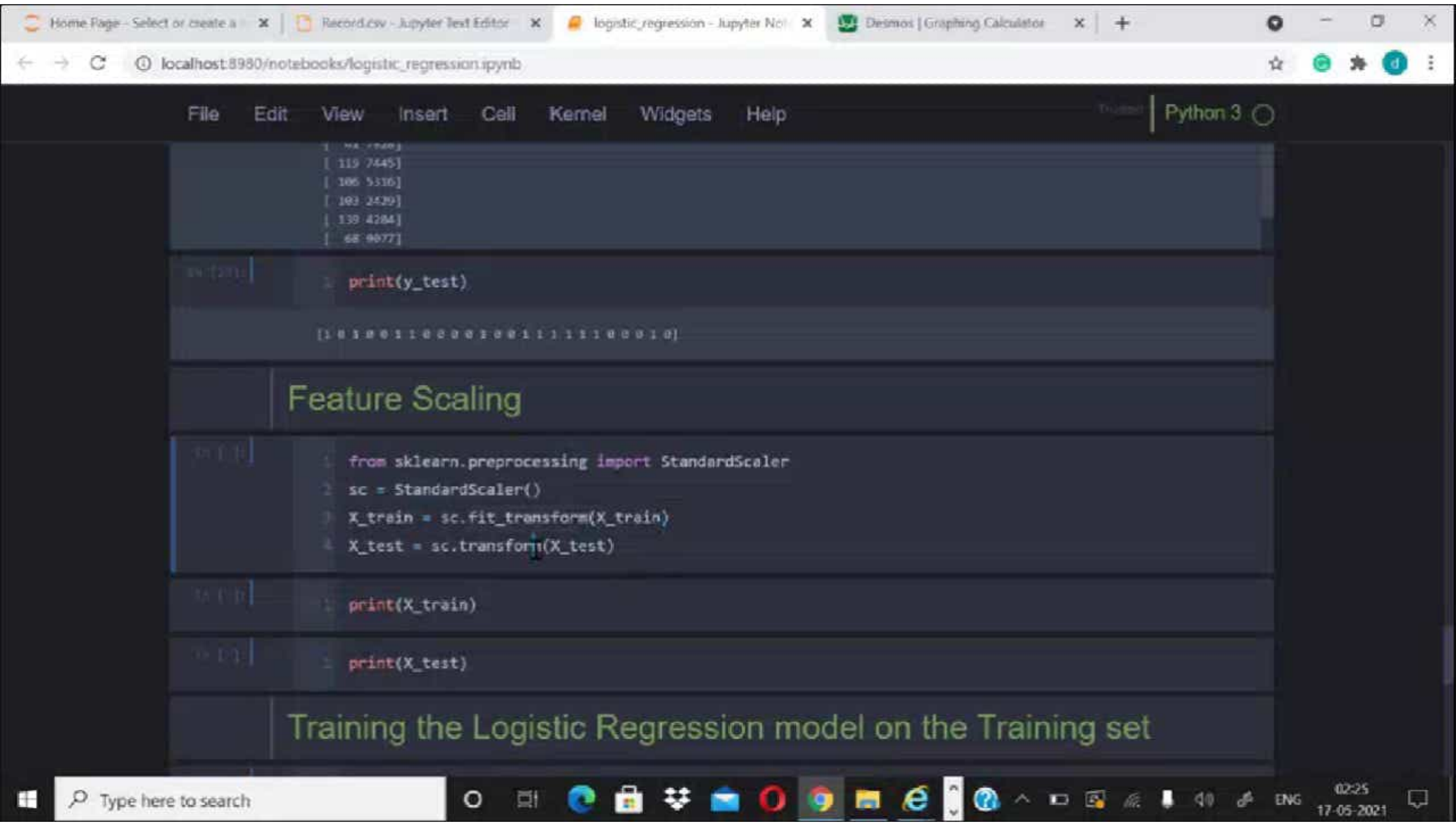
- To put all the features on same scale
- For some ML models, to avoid some features dominating other features.
- We need not apply it to all the models
    - Because like in multiple linear regression : $y = bo + b1*x1 + b2*x2 + b3*x3$
        - b1, b2, b3 are there to avoid dominance of any feature
        - Coeff would take high values for the small-valued features

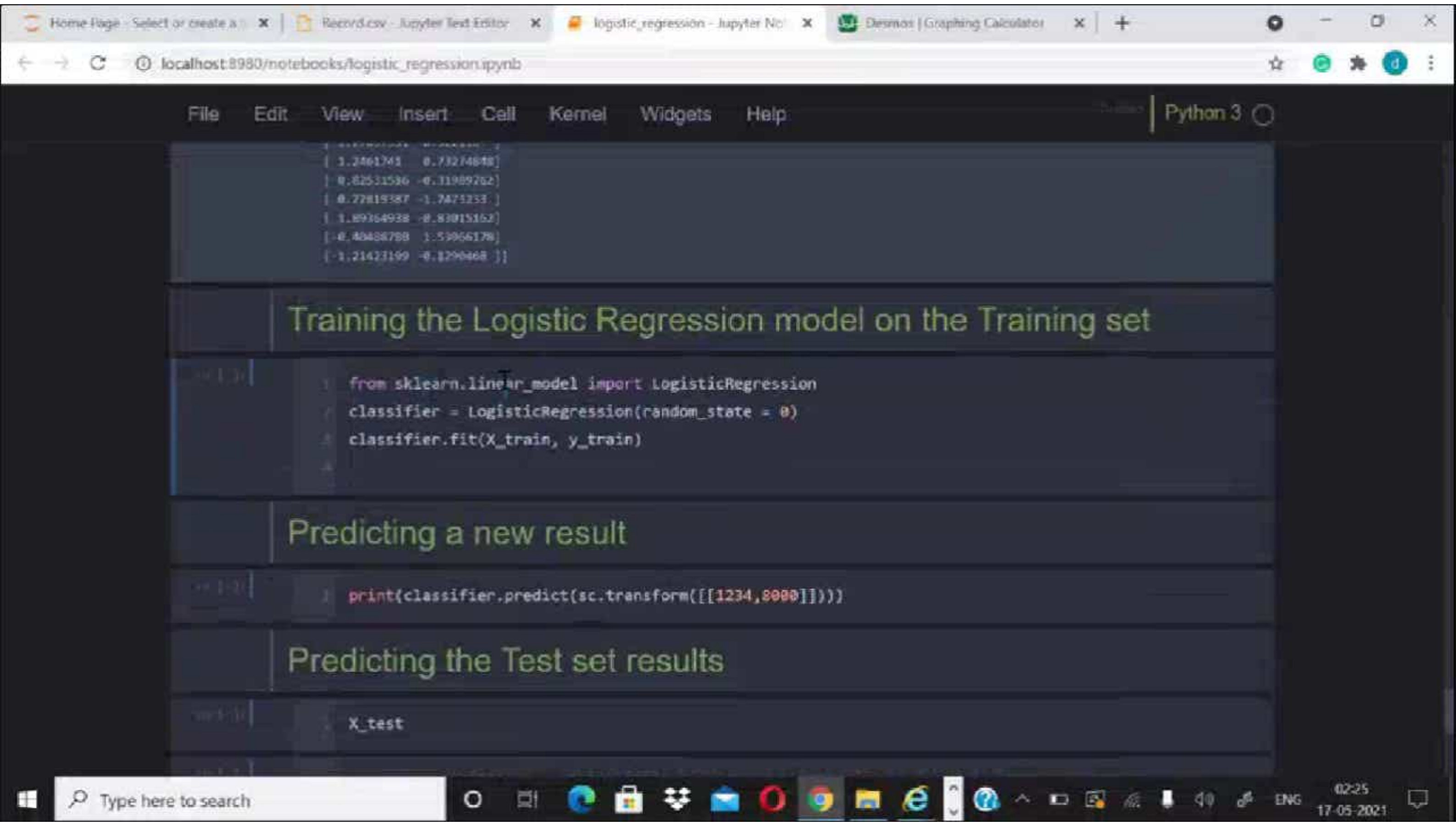
Standardisation	Normalization
$X_{stand} = x - \frac{mean(x)}{standard\ deviation(x)}$	$X_{norm} = \frac{x - \min(x)}{\max(x) - \min(x)}$
Value ranges from -3 to 3	Value ranges from 0 to 1
Can be applied to all type of data	Can only be applied to most of features having normal distribution

- Scaler will be fitted only to x_train first
- And then transformed to x_test

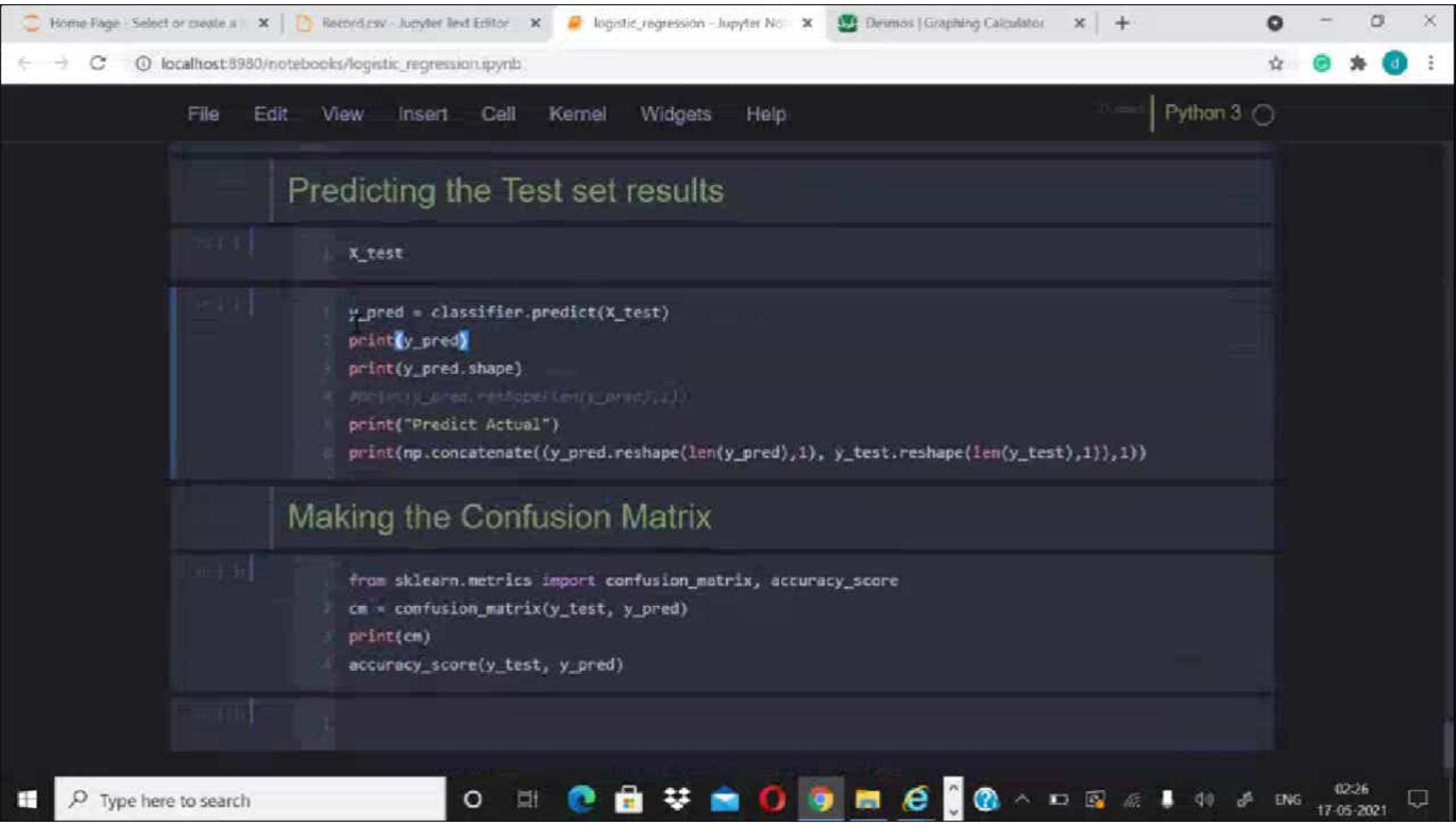
- Need not apply transformation on dummy variable(e.g.-one we obtained after encoding categorical data.

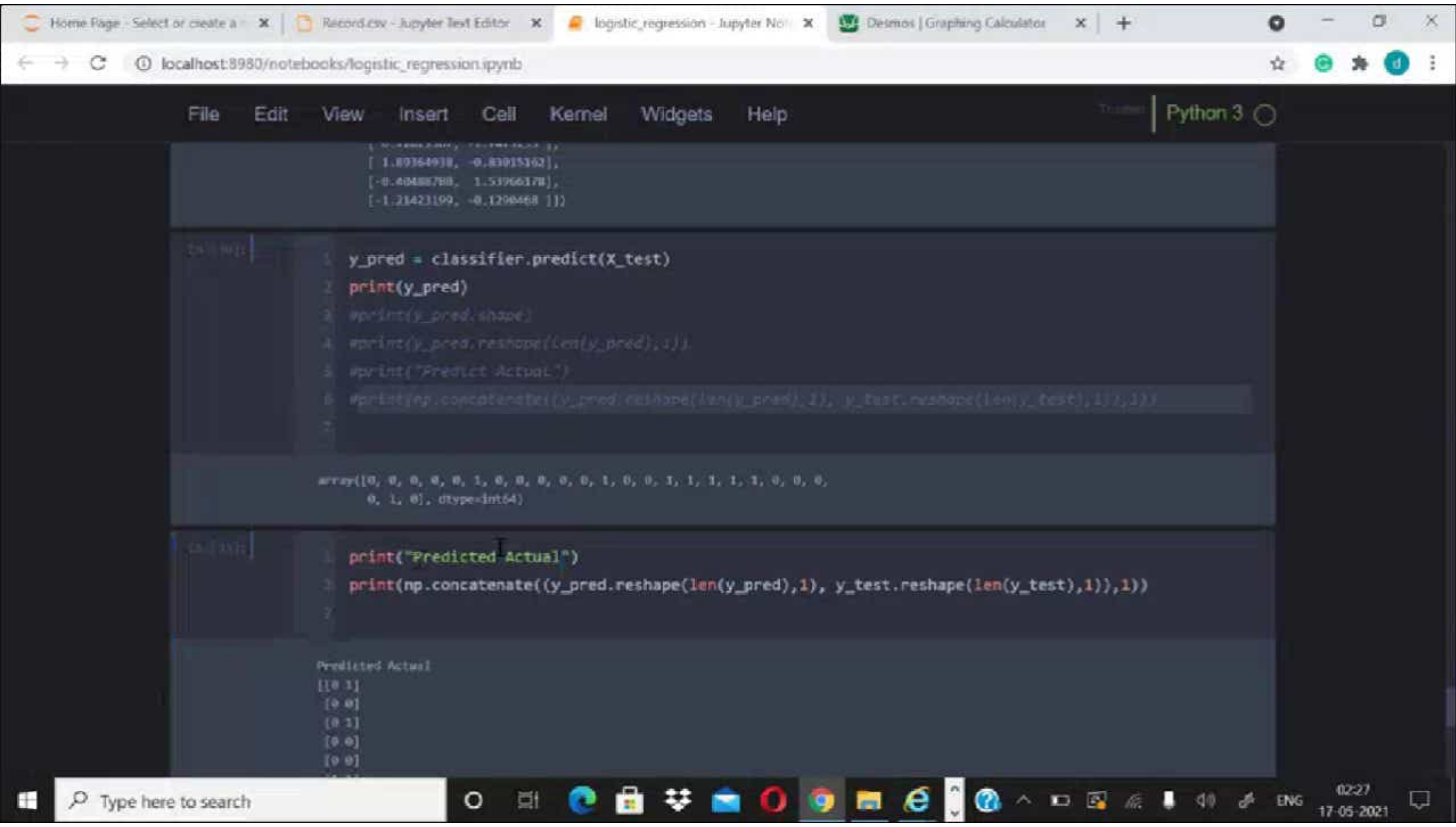
- Because after standardisation, value should range from -3 to 3.
- Dummy variables are already having value 0, 1, which already lies in that range.

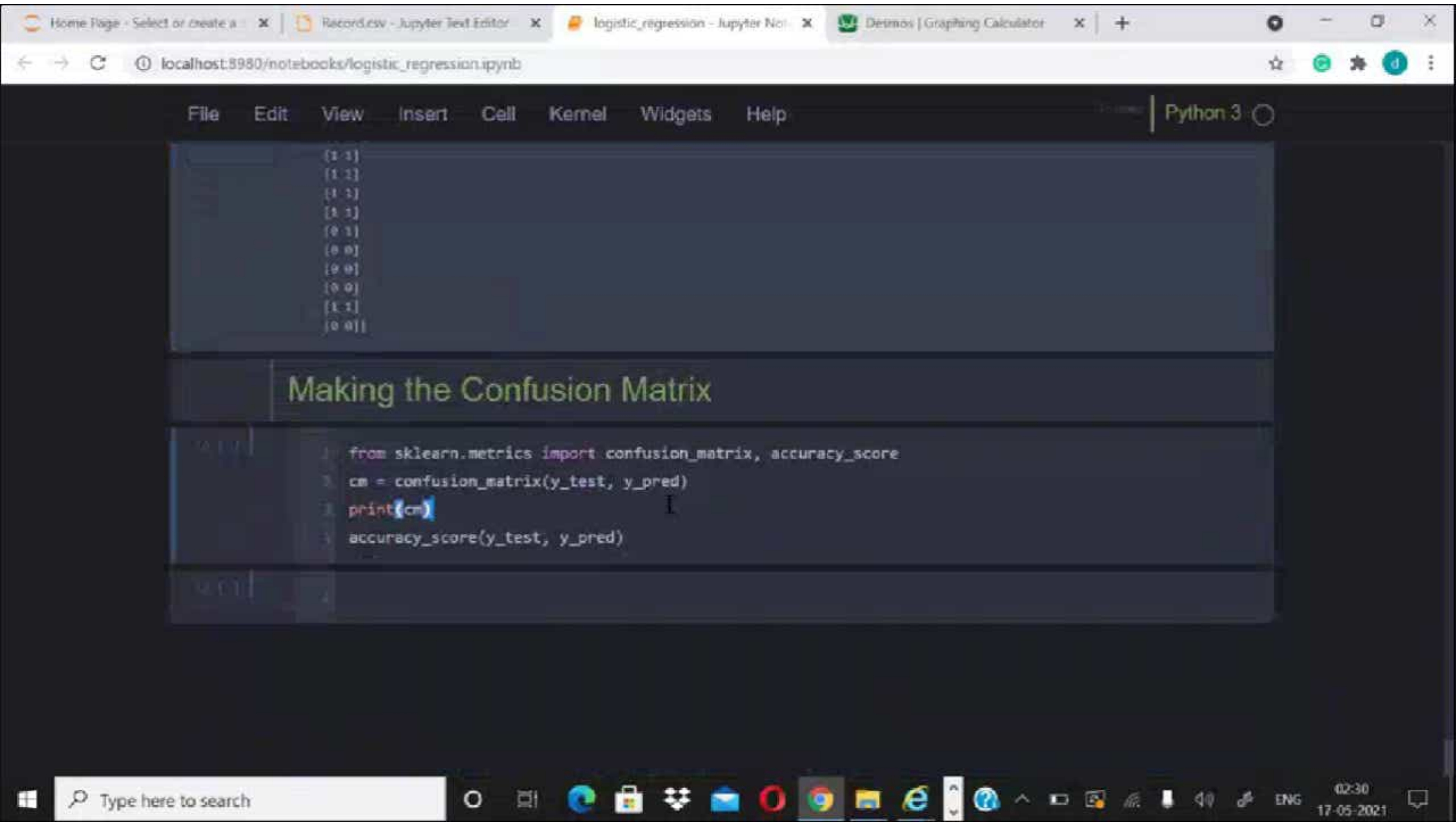

Home Page - Select or create a
Record.csv - Jupyter Text Editor
logistic_regression - Jupyter Not
Desmos | Graphing Calculator
localhost:8980/notebooks/logistic_regression.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python 3
[119 7445]
[106 5316]
[103 2429]
[139 4284]
[68 9077]
print(y_test)
[1 0 1 0 0 1 1 0 0 0 0 1 0 0 1 1 1 1 1 1 [illegible] 0 0 0 1 [illegible]]
Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
print(X_train)
print(X_test)
Training the Logistic Regression model on the Training set
Type here to search
ENG
02:25
17-05-2021



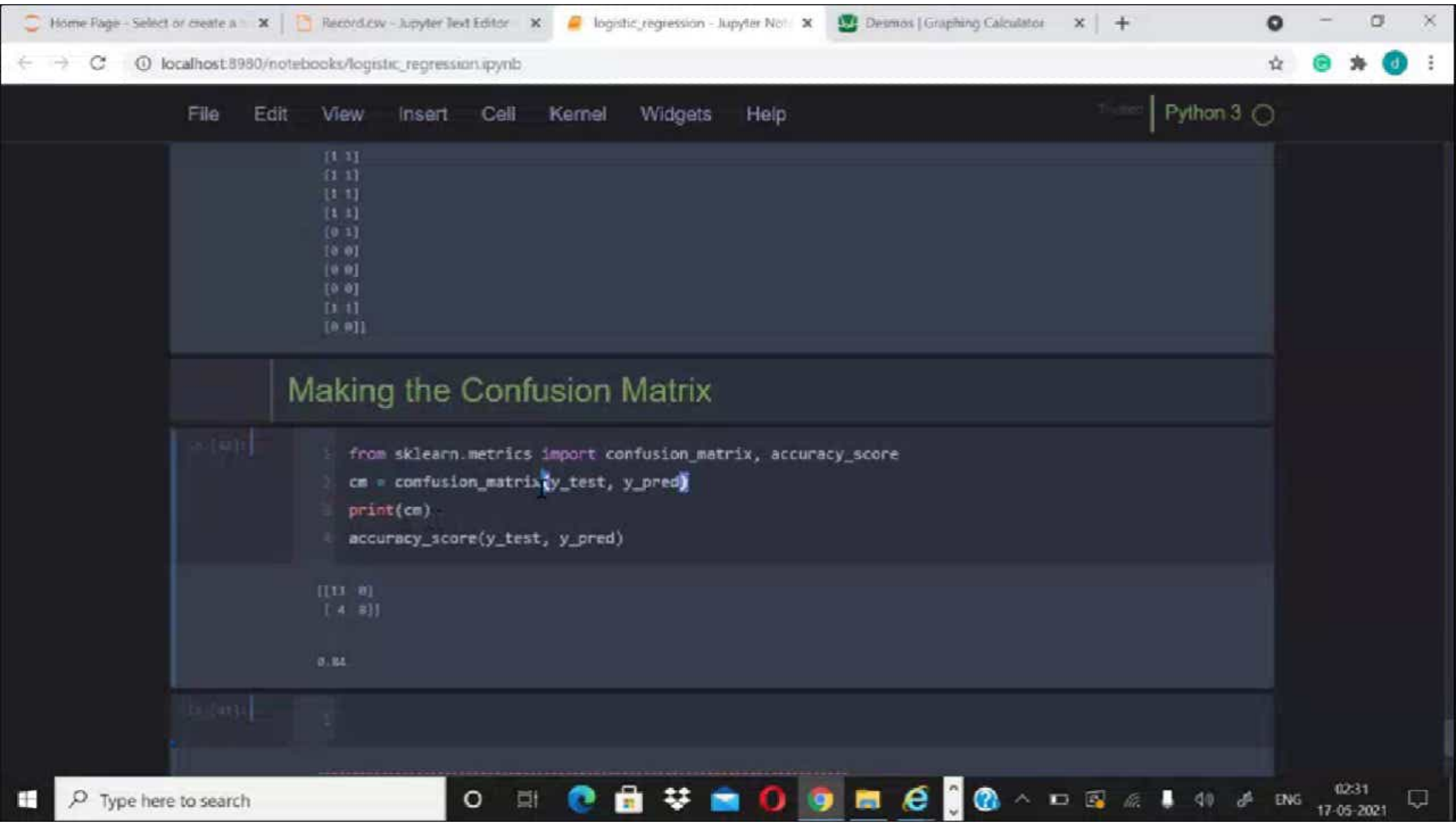
localhost:8980/notebooks/logistic_regression.ipynb
File Edit View Insert Cell Kernel Widgets Help
Python 3
Training the Logistic Regression model on the Training set
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train, y_train)
Predicting a new result
print(classifier.predict(sc.transform([[1234,8000]])))
Predicting the Test set results
X_test
Type here to search
ENG
02:25
17-05-2021


localhost:8980/notebooks/logistic_regression.ipynb
File Edit View Insert Cell Kernel Widgets Help
Python 3
Predicting the Test set results
X_test
y_pred = classifier.predict(X_test)
print(y_pred)
print(y_pred.shape)
print("Predict Actual")
print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))
Making the Confusion Matrix
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)
Type here to search
02:26
17-05-2021


Home Page - Select or create a
Record.csv - Jupyter Text Editor
logistic_regression - Jupyter No
Desmos | Graphing Calculator
localhost:8980/notebooks/logistic_regression.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python 3
[1.89364938, -0.83015162],
[-0.40488788, 1.53966178],
[-1.21423199, -0.1290468]])
y_pred = classifier.predict(X_test)
print(y_pred)
#print(y_pred.shape)
#print(y_pred.reshape(len(y_pred),1))
#print("Predict Actual")
#print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))
array([0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0,
0, 1, 0], dtype=int64)
print("Predicted Actual")
print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))
Predicted Actual
[[0 1]
[0 0]
[0 1]
[0 0]
[0 0]
Type here to search
ENG
02:27
17-05-2021



Home Page - Select or create a
Record.csv - Jupyter Text Editor
logistic_regression - Jupyter Not
Desmos | Graphing Calculator
localhost:8980/notebooks/logistic_regression.ipynb
File Edit View Insert Cell Kernel Widgets Help
Python 3
[1 1]
[1 1]
[1 1]
[1 1]
[0 1]
[0 0]
[0 0]
[0 0]
[1 1]
[0 0]]
Making the Confusion Matrix
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)
Type here to search
ENG
02:30
17-05-2021


Home Page - Select or create a
Record.csv - Jupyter Text Editor
logistic_regression - Jupyter Not
Desmos | Graphing Calculator
localhost:8980/notebooks/logistic_regression.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python 3
[1 1]
[1 1]
[1 1]
[1 1]
[0 1]
[0 0]
[0 0]
[0 0]
[1 1]
[0 0]]
Making the Confusion Matrix
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)
[[11 0]
[4 0]]
0.84
Type here to search
ENG
02:31
17-05-2021