

# DATA COLLECTION

## Splitting Dependent and Independent Columns

- Splitting the dataset into the matrix of independent variables and the vector or dependent variable.
- Mathematically, Vector is defined as a matrix that has just one column.
- To read the columns, we will use **iloc () function** of pandas (used to fix the indexes for selection) which takes two parameters:
  - Row selection
  - Column selection

### Syntax for iloc () function:

- **X = data.iloc[:,0:7].values** :- “:” indicates that you are considering all the rows in the dataset and “0:7” indicates that you are considering columns 0 to 7 as input values and assigning them to variable **x**.
- **Y = data.iloc[:,7:0].values** :- “:” indicates you are considering all the rows and “7:” indicates that you are considering only the last column as output value and assigning them to variable **y**.

### Syntax for shape () function:

- **X.Shape()** :- 400 rows with 7 columns.
- **Y.Shape()** :- 400 rows with 1 columns.

### Step-1:

- Firstly, we need to split the independent variables and dependent variables.
- In pandas, **iloc () function** is used to split data by full columns present in the dataset.
- Then, we are assigning **X** variable for full columns and rows and **Y** variable for 1 column and all rows.

The screenshot shows a Jupyter Notebook interface with a file explorer on the left and a code editor on the right. The code editor contains the following Python code:

```
>> Splitting Dependent And Independent Columns

[33]: x = data.iloc[:,0:7].values
      x

[33]: array([[337., 118., 4., ..., 4.5, 9.65, 1. ],
            [324., 107., 4., ..., 4.5, 8.87, 1. ],
            [316., 104., 3., ..., 3.5, 8., 1. ],
            ...,
            [330., 116., 4., ..., 4.5, 9.45, 1. ],
            [312., 103., 3., ..., 4., 8.78, 0. ],
            [333., 117., 4., ..., 4., 9.66, 1. ]])

[34]: y = data.iloc[:,7:].values
      y

[34]: array([[0.92],
            [0.76],
            [0.72],
            [0.8 ],
            [0.65],
            [0.9 ],
            [0.75],
            [0.68],
            [0.5 ],
            [0.45],
            [0.52],
            [0.84],
            [0.78],
            [0.62],
            [0.61],
            [0.54],
            [0.66],
            [0.65],
            [0.63]])
```

## Step-2:

- After splitting the data, we need to check the shape of the splitted variables **X** and **Y**.
- Then, it will show the number of columns and rows.

The screenshot shows the same Jupyter Notebook interface, but now the code editor displays the shapes of the variables X and Y:

```
[34]: array([[0.92],
            [0.76],
            [0.72],
            [0.8 ],
            [0.65],
            [0.9 ],
            [0.75],
            [0.68],
            [0.5 ],
            [0.45],
            [0.52],
            [0.84],
            [0.78],
            [0.62],
            [0.61],
            [0.54],
            [0.66],
            [0.65],
            [0.63]])

[35]: x.shape

[35]: (400, 7)

[36]: y.shape

[36]: (400, 1)

[ ]:
```