

# Welcome | Day 16 ML | TheiScale |By Nishant Sir

## Import Library

```
In [8]: import pandas as pd
import numpy as np

from sklearn.linear_model import LinearRegression
```

## Import Dataset

```
In [9]: df = np.round(pd.read_csv('50_Startups.csv')[['R&D Spend', 'Administration', 'Marketing
np.random.seed(9)
df = df.sample(5)
df
```

```
Out[9]:
```

	R&D Spend	Administration	Marketing Spend	Profit
21	8.0	15.0	30.0	11.0
37	4.0	5.0	20.0	9.0
2	15.0	10.0	41.0	19.0
14	12.0	16.0	26.0	13.0
44	2.0	15.0	3.0	7.0

## Remove Target Column

```
In [10]: df = df.iloc[:,0:-1]
df
```

```
Out[10]:
```

	R&D Spend	Administration	Marketing Spend
21	8.0	15.0	30.0
37	4.0	5.0	20.0
2	15.0	10.0	41.0
14	12.0	16.0	26.0
44	2.0	15.0	3.0

# NaN value import (Manipulate the Data)

```
In [13]: df.iloc[1,0] = np.NaN  
df.iloc[3,1] = np.NaN  
df.iloc[-1,-1] = np.NaN
```

C:\Users\ASUS\AppData\Local\Temp\ipykernel\_12584\2040708130.py:1: SettingWithCopyWarning:  
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy) ([https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy))

```
df.iloc[1,0] = np.NaN
```

C:\Users\ASUS\AppData\Local\Temp\ipykernel\_12584\2040708130.py:2: SettingWithCopyWarning:  
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy) ([https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy))

```
df.iloc[3,1] = np.NaN
```

C:\Users\ASUS\AppData\Local\Temp\ipykernel\_12584\2040708130.py:3: SettingWithCopyWarning:  
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy) ([https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy))

```
df.iloc[-1,-1] = np.NaN
```

```
In [12]: df.head()
```

```
Out[12]:
```

	R&D Spend	Administration	Marketing Spend
21	8.0	15.0	30.0
37	NaN	5.0	20.0
2	15.0	10.0	41.0
14	12.0	NaN	26.0
44	2.0	15.0	NaN

## Step 1 - Impute all missing values with mean

```
In [14]: df0 = pd.DataFrame()  
  
df0['R&D Spend'] = df['R&D Spend'].fillna(df['R&D Spend'].mean())  
df0['Administration'] = df['Administration'].fillna(df['Administration'].mean())  
df0['Marketing Spend'] = df['Marketing Spend'].fillna(df['Marketing Spend'].mean())
```

In [15]: df0

Out[15]:

	R&D Spend	Administration	Marketing Spend
21	8.00	15.00	30.00
37	9.25	5.00	20.00
2	15.00	10.00	41.00
14	12.00	11.25	26.00
44	2.00	15.00	29.25

## Step 2 - Remove the column 1 imputed value (Left to Right)



In [16]: df1 = df0.copy()  
  
df1.iloc[1,0] = np.NaN  
  
df1

Out[16]:

	R&D Spend	Administration	Marketing Spend
21	8.0	15.00	30.00
37	NaN	5.00	20.00
2	15.0	10.00	41.00
14	12.0	11.25	26.00
44	2.0	15.00	29.25

## Training Data in X (Training Input)

In [19]: X = df1.iloc[[0,2,3,4],1:3]  
X

Out[19]:

	Administration	Marketing Spend
21	15.00	30.00
2	10.00	41.00
14	11.25	26.00
44	15.00	29.25

## Training Data in Y (Corresponding Output)

In [18]: y = df1.iloc[[0,2,3,4],0]  
y

Out[18]:

21	8.0
2	15.0
14	12.0
44	2.0

Name: R&D Spend, dtype: float64

## Step 3 - Predict missing value of column 1

```
In [23]: lr = LinearRegression()  
lr.fit(X,y)  
lr.predict(df1.iloc[1,1:].values.reshape(1,2))
```

C:\Users\ASUS\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names  
warnings.warn(

```
Out[23]: array([23.14158651])
```

```
In [24]: df1.iloc[1,0] = 23.14
```

```
In [25]: df1
```

```
Out[25]:
```

	R&D Spend	Administration	Marketing Spend
21	8.00	15.00	30.00
37	23.14	5.00	20.00
2	15.00	10.00	41.00
14	12.00	11.25	26.00
44	2.00	15.00	29.25

## Step 4 - Remove the column 2 imputed value (Left to Right)

```
In [26]: df1.iloc[3,1] = np.NaN  
  
df1
```

```
Out[26]:
```

	R&D Spend	Administration	Marketing Spend
21	8.00	15.0	30.00
37	23.14	5.0	20.00
2	15.00	10.0	41.00
14	12.00	NaN	26.00
44	2.00	15.0	29.25

## Training Data in X (Training Input) | Column 2

```
In [27]: X = df1.iloc[[0,1,2,4],[0,2]]  
X
```

```
Out[27]:
```

	R&D Spend	Marketing Spend
21	8.00	30.00
37	23.14	20.00
2	15.00	41.00
44	2.00	29.25

## Training Data in Y (Corresponding Output) | Column 2

```
In [29]: y = df1.iloc[[0,1,2,4],1]
y
```

```
Out[29]: 21    15.0
37     5.0
2     10.0
44    15.0
Name: Administration, dtype: float64
```

## Step 5 - Predict missing value of column 2

```
In [30]: lr = LinearRegression()
lr.fit(X,y)
lr.predict(df1.iloc[3,[0,2]].values.reshape(1,2))
```

C:\Users\ASUS\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names  
warnings.warn(

```
Out[30]: array([11.06331285])
```

```
In [31]: df1.iloc[3,1] = 11.06
```

```
In [32]: df1
```

```
Out[32]:
```

	R&D Spend	Administration	Marketing Spend
21	8.00	15.00	30.00
37	23.14	5.00	20.00
2	15.00	10.00	41.00
14	12.00	11.06	26.00
44	2.00	15.00	29.25

## Step 6 - Remove the column 3 imputed value (Left to Right)

```
In [33]: df1.iloc[4,-1] = np.NaN
df1
```

```
Out[33]:
```

	R&D Spend	Administration	Marketing Spend
21	8.00	15.00	30.0
37	23.14	5.00	20.0
2	15.00	10.00	41.0
14	12.00	11.06	26.0
44	2.00	15.00	NaN

## Training Data in X (Training Input) | Column 3

```
In [34]: X = df1.iloc[0:4,0:2]
X
```

```
Out[34]:
```

	R&D Spend	Administration
21	8.00	15.00
37	23.14	5.00
2	15.00	10.00
14	12.00	11.06

## Training Data in Y (Corresponding Output) | Column 3

```
In [35]: y = df1.iloc[0:4, -1]
y
```

```
Out[35]:
```

21	30.0
37	20.0
2	41.0
14	26.0

Name: Marketing Spend, dtype: float64

## Step 7 - Predict missing value of column 3

```
In [37]: lr = LinearRegression()
lr.fit(X,y)
lr.predict(df1.iloc[4,0:2].values.reshape(1,2))
```

C:\Users\ASUS\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names  
warnings.warn(

```
Out[37]: array([31.56351448])
```

```
In [38]: df1.iloc[4, -1] = 31.56
```

```
In [39]: df1
```

```
Out[39]:
```

	R&D Spend	Administration	Marketing Spend
21	8.00	15.00	30.00
37	23.14	5.00	20.00
2	15.00	10.00	41.00
14	12.00	11.06	26.00
44	2.00	15.00	31.56

## Step 8 - Subtract 0th (df0) iteration from 1st (df1) iteration

```
In [40]: df1 - df0
```

```
Out[40]:
```

	R&D Spend	Administration	Marketing Spend
21	0.00	0.00	0.00
37	13.89	0.00	0.00
2	0.00	0.00	0.00
14	0.00	-0.19	0.00
44	0.00	0.00	2.31

## Again Iteration Process

```
In [42]: df2 = df1.copy()

df2.iloc[1,0] = np.NaN

df2
```

```
Out[42]:
```

	R&D Spend	Administration	Marketing Spend
21	8.0	15.00	30.00
37	NaN	5.00	20.00
2	15.0	10.00	41.00
14	12.0	11.06	26.00
44	2.0	15.00	31.56

```
In [43]: X = df2.iloc[[0,2,3,4],1:3]
y = df2.iloc[[0,2,3,4],0]

lr = LinearRegression()
lr.fit(X,y)
lr.predict(df2.iloc[1,1:].values.reshape(1,2))
```

C:\Users\ASUS\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names  
warnings.warn(

```
Out[43]: array([23.78627207])
```

```
In [44]: df2.iloc[1,0] = 23.78
```

```
In [45]: df2.iloc[3,1] = np.NaN
X = df2.iloc[[0,1,2,4],[0,2]]
y = df2.iloc[[0,1,2,4],1]

lr = LinearRegression()
lr.fit(X,y)
lr.predict(df2.iloc[3,[0,2]].values.reshape(1,2))
```

C:\Users\ASUS\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names  
warnings.warn(

```
Out[45]: array([11.22020174])
```

```
In [46]: df2.iloc[3,1] = 11.22
```

```
In [47]: df2.iloc[4,-1] = np.NaN
```

```
X = df2.iloc[0:4,0:2]  
y = df2.iloc[0:4,-1]
```

```
lr = LinearRegression()
```

```
lr.fit(X,y)
```

```
lr.predict(df2.iloc[4,0:2].values.reshape(1,2))
```

C:\Users\ASUS\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names  
warnings.warn(

```
Out[47]: array([38.87979054])
```

```
In [48]: df2.iloc[4,-1] = np.NaN
```

```
X = df2.iloc[0:4,0:2]  
y = df2.iloc[0:4,-1]
```

```
lr = LinearRegression()
```

```
lr.fit(X,y)
```

```
lr.predict(df2.iloc[4,0:2].values.reshape(1,2))
```

C:\Users\ASUS\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names  
warnings.warn(

```
Out[48]: array([38.87979054])
```

```
In [49]: df2.iloc[4,-1] = 31.56
```

```
In [50]: df2
```

```
Out[50]:
```

	R&D Spend	Administration	Marketing Spend
--	-----------	----------------	-----------------

21	8.00	15.00	30.00
----	------	-------	-------

37	23.78	5.00	20.00
----	-------	------	-------

2	15.00	10.00	41.00
---	-------	-------	-------

14	12.00	11.22	26.00
----	-------	-------	-------

44	2.00	15.00	31.56
----	------	-------	-------

```
In [51]: df2 - df1
```

```
Out[51]:
```

	R&D Spend	Administration	Marketing Spend
--	-----------	----------------	-----------------

21	0.00	0.00	0.0
----	------	------	-----

37	0.64	0.00	0.0
----	------	------	-----

2	0.00	0.00	0.0
---	------	------	-----

14	0.00	0.16	0.0
----	------	------	-----

44	0.00	0.00	0.0
----	------	------	-----



```
In [52]: df3 = df2.copy()

df3.iloc[1,0] = np.NaN

df3
```

```
Out[52]:
```

	R&D Spend	Administration	Marketing Spend
21	8.0	15.00	30.00
37	NaN	5.00	20.00
2	15.0	10.00	41.00
14	12.0	11.22	26.00
44	2.0	15.00	31.56

```
In [53]: X = df3.iloc[[0,2,3,4],1:3]
y = df3.iloc[[0,2,3,4],0]

lr = LinearRegression()
lr.fit(X,y)
lr.predict(df3.iloc[1,1:].values.reshape(1,2))
```

C:\Users\ASUS\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names  
warnings.warn(

```
Out[53]: array([24.57698058])
```

```
In [54]: df3.iloc[1,0] = 24.57
```

```
In [55]: df3.iloc[3,1] = np.NaN
X = df3.iloc[[0,1,2,4],[0,2]]
y = df3.iloc[[0,1,2,4],1]

lr = LinearRegression()
lr.fit(X,y)
lr.predict(df3.iloc[3,[0,2]].values.reshape(1,2))
```

C:\Users\ASUS\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names  
warnings.warn(

```
Out[55]: array([11.37282844])
```

```
In [56]: df3.iloc[3,1] = 11.37
```

```
In [57]: df3.iloc[4,-1] = np.NaN

X = df3.iloc[0:4,0:2]
y = df3.iloc[0:4,-1]

lr = LinearRegression()
lr.fit(X,y)
lr.predict(df3.iloc[4,0:2].values.reshape(1,2))
```

C:\Users\ASUS\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names  
warnings.warn(

```
Out[57]: array([45.53976417])
```

In [58]:

df3.iloc[4,-1] = 45.53

In [59]:

df2.iloc[3,1] = 11.22

In [60]:

df3

Out[60]:

	R&D Spend	Administration	Marketing Spend
21	8.00	15.00	30.00
37	24.57	5.00	20.00
2	15.00	10.00	41.00
14	12.00	11.37	26.00
44	2.00	15.00	45.53

In [61]:

df3 - df2

Out[61]:

	R&D Spend	Administration	Marketing Spend
21	0.00	0.00	0.00
37	0.79	0.00	0.00
2	0.00	0.00	0.00
14	0.00	0.15	0.00
44	0.00	0.00	13.97

In [ ]: