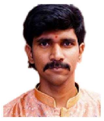


Step - 1 : Business Problem Understanding

- Predict **Salary** of a person based on input variables

In [1]:

```
import numpy as np
import pandas as pd
```

Step - 2 : Data Understanding

Load Data & Understand every variable

In [2]:

```
dataset = pd.read_csv('hiring.csv')
dataset
```

Out[2]:

	experience	test_score	interview_score	salary
0	NaN	8.0	9	50000
1	NaN	8.0	6	45000
2	5.0	6.0	7	60000
3	2.0	10.0	10	65000
4	7.0	9.0	6	70000
5	3.0	7.0	10	62000
6	10.0	NaN	7	72000
7	11.0	7.0	8	80000

Dataset Understanding

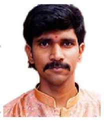
In [3]:

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8 entries, 0 to 7
Data columns (total 4 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   experience       6 non-null      float64
 1   test_score       7 non-null      float64
 2   interview_score  8 non-null      int64  
 3   salary           8 non-null      int64  
dtypes: float64(2), int64(2)
memory usage: 388.0 bytes
```

Step - 3 : Data Preprocessing

Data Cleaning



In [4]:
```
dataset.isnull().sum()
```

Out[4]:
```
experience         2
test_score         1
interview_score    0
salary             0
dtype: int64
```

In [5]:
```
dataset['experience'].fillna(0, inplace=True)
dataset['test_score'].fillna(dataset['test_score'].mean(), inplace=True
```

In [6]:
```
dataset
```

Out[6]:

	experience	test_score	interview_score	salary
0	0.0	8.000000	9	50000
1	0.0	8.000000	6	45000
2	5.0	6.000000	7	60000
3	2.0	10.000000	10	65000
4	7.0	9.000000	6	70000
5	3.0	7.000000	10	62000
6	10.0	7.857143	7	72000
7	11.0	7.000000	8	80000

X&y

In [7]:
```
X = dataset.drop("salary",axis=1)
y = dataset["salary"]
```

Train-Test Split

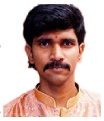
In [8]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2
```

Step - 4 : Modelling

In [9]:
```
from sklearn.linear_model import LinearRegression
lr_model = LinearRegression()
lr_model.fit(X_train, y_train)
```

Out[9]:
```
▾ LinearRegression

LinearRegression()
```



In [10]:

```
lr_model.coef_
```

Out[10]:

```
array([2716.4868386 , 2126.92037616, 1612.77951649])
```

In [11]:

```
lr_model.intercept_
```

Out[11]:

```
20722.331847729445
```

Predictions

In [12]:

```
train_predictions = lr_model.predict(X_train)
test_predictions = lr_model.predict(X_test)
```

Step - 5 : Evaluation

In [13]:

```
lr_model.score(X_train,y_train)  # Train R2
```

Out[13]:

```
0.9451713252248061
```

In [14]:

```
lr_model.score(X_test,y_test)    # Test R2
```

Out[14]:

```
0.9287916364000984
```

Saving a model

In [15]:

```
from joblib import dump
dump(lr_model, 'lr_model.joblib')
```

Out[15]:

```
['lr_model.joblib']
```

In [16]:

```
from pickle import dump
dump(lr_model, open('lr_model.pkl','wb'))
```

Q :which should be selected either pickle or joblib?

Ans: as per the requirements of deployment team

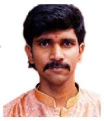
Prediction on New Data

In [17]:

```
new_data = pd.DataFrame({"experience":[0],"test_score":[9],"interview_s
new_data
```

Out[17]:

	experience	test_score	interview_score
0	0	9	9



In [18]:

```python
from pickle import load
loaded_model = load(open('lr_model.pkl','rb'))
loaded_model.predict(new_data)
```

Out[18]:

```
array([54379.63088154])
```