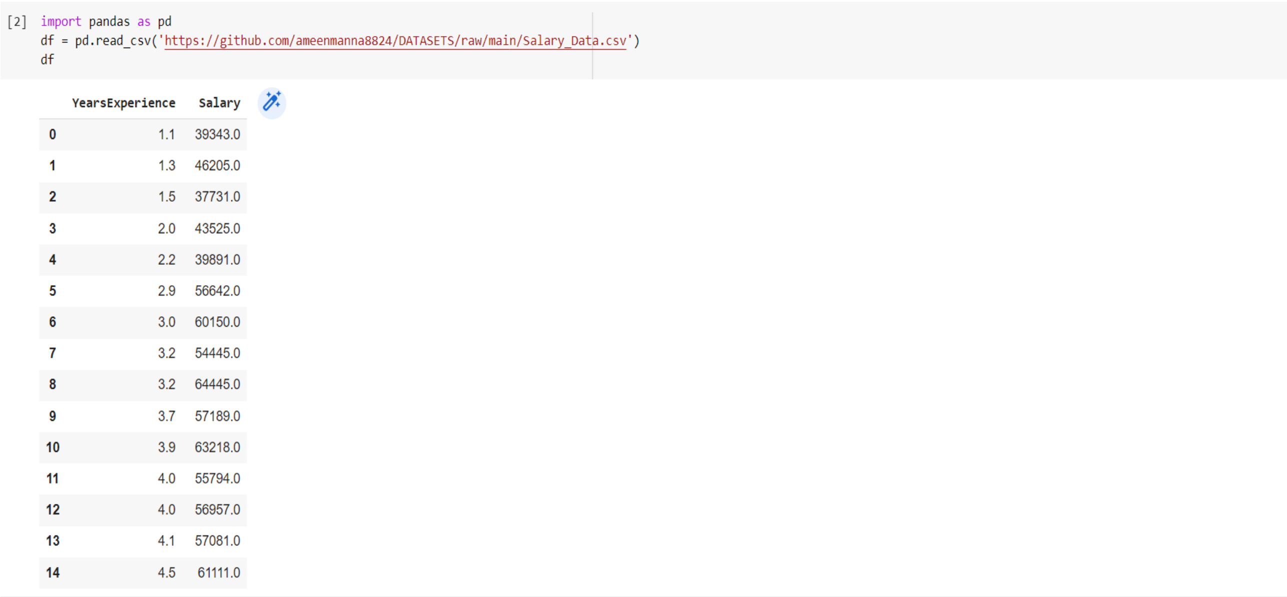
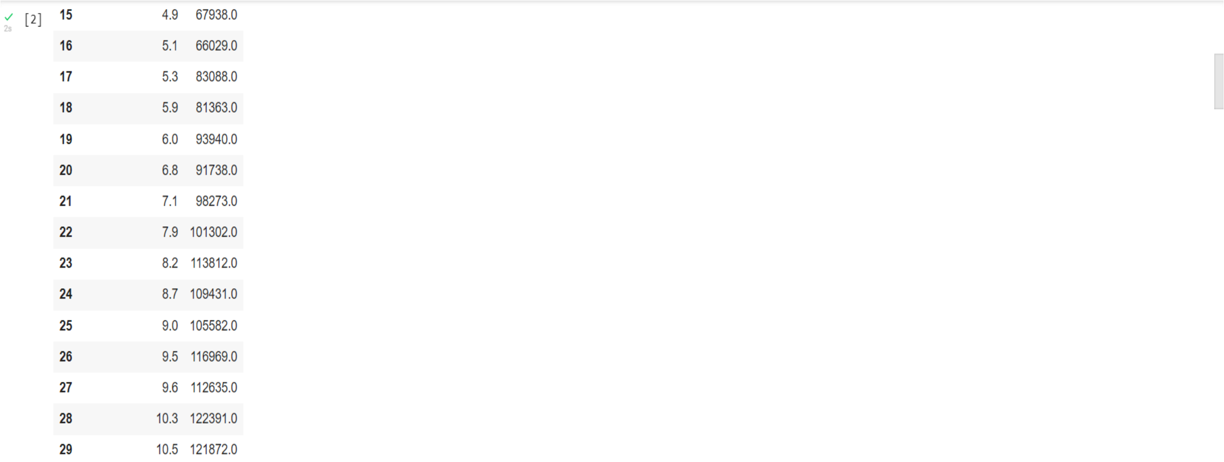
AI MAJOR AUGUST BATCH

Project Name:EDA ON SALARY DATASET

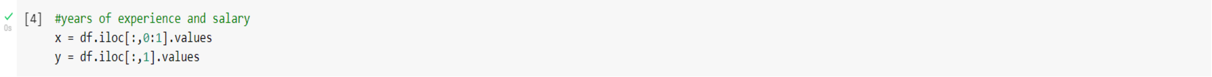


```
[2] import pandas as pd
    df = pd.read_csv('https://github.com/ameenmanna8824/DATASETS/raw/main/Salary_Data.csv')
    df
```

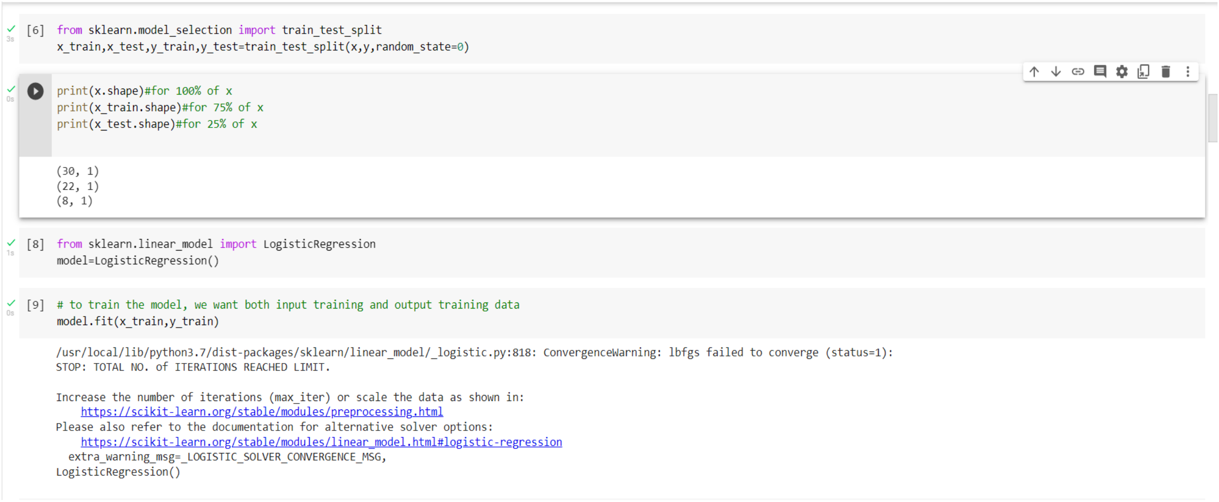
	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0
5	2.9	56642.0
6	3.0	60150.0
7	3.2	54445.0
8	3.2	64445.0
9	3.7	57189.0
10	3.9	63218.0
11	4.0	55794.0
12	4.0	56957.0
13	4.1	57081.0
14	4.5	61111.0



15	4.9	67938.0
16	5.1	66029.0
17	5.3	83088.0
18	5.9	81363.0
19	6.0	93940.0
20	6.8	91738.0
21	7.1	98273.0
22	7.9	101302.0
23	8.2	113812.0
24	8.7	109431.0
25	9.0	105582.0
26	9.5	116969.0
27	9.6	112635.0
28	10.3	122391.0
29	10.5	121872.0



```
✓ [4] #years of experience and salary
      x = df.iloc[:,0:1].values
      y = df.iloc[:,1].values
```

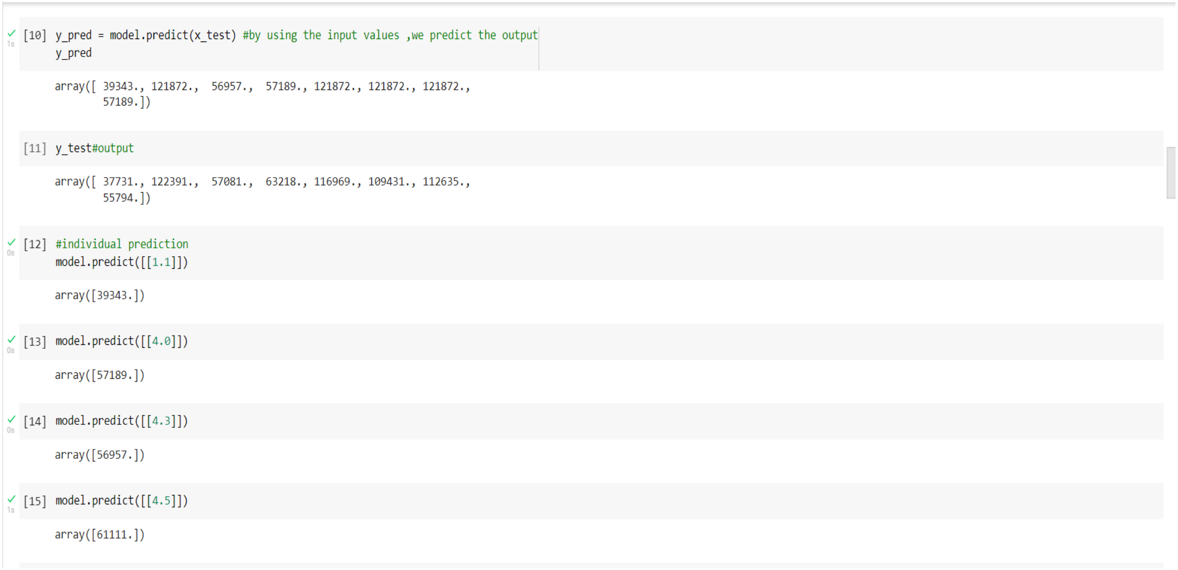


```
[6]  from sklearn.model_selection import train_test_split
     x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=0)
```

```
     print(x.shape)#for 100% of x
     print(x_train.shape)#for 75% of x
     print(x_test.shape)#for 25% of x

     (30, 1)
     (22, 1)
     (8, 1)
```

```
[8]  from sklearn.linear_model import LogisticRegression
     model=LogisticRegression()
```

```
[9]  # to train the model, we want both input training and output training data
     model.fit(x_train,y_train)

     /usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:818: ConvergenceWarning: lbfgs failed to converge (status=1):
     STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

     Increase the number of iterations (max_iter) or scale the data as shown in:
         https://scikit-learn.org/stable/modules/preprocessing.html
     Please also refer to the documentation for alternative solver options:
         https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
       extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG,
     LogisticRegression()
```



```
[10] y_pred = model.predict(x_test) #by using the input values ,we predict the output
     y_pred

     array([ 39343., 121872.,  56957.,  57189., 121872., 121872., 121872.,
             57189.])
```

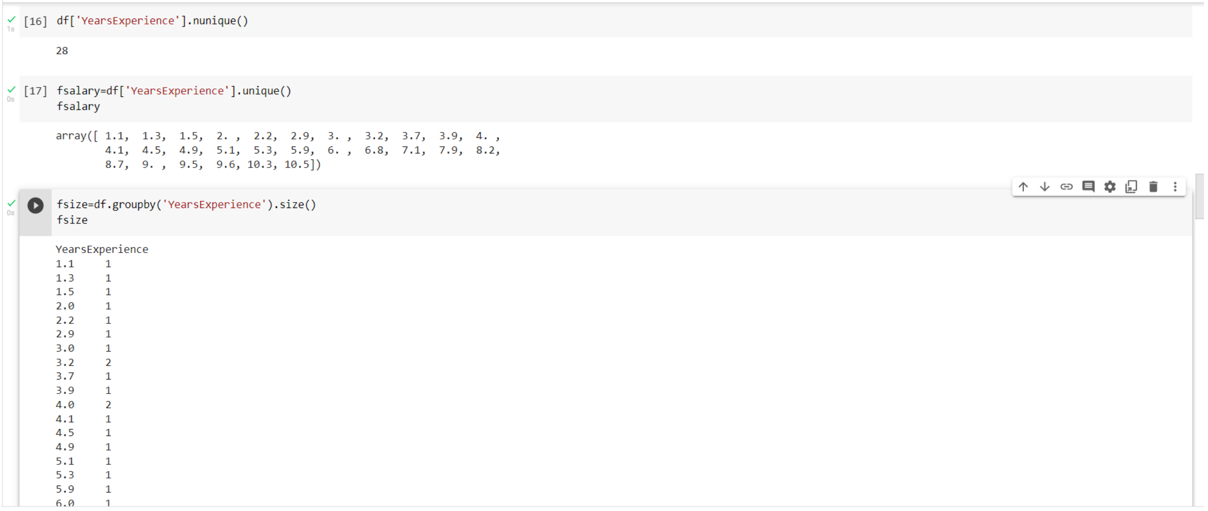
```
[11] y_test#output

     array([ 37731., 122391.,  57081.,  63218., 116969., 109431., 112635.,
             55794.])
```

```
[12] #individual prediction
     model.predict([[1.1]])

     array([39343.])
```

```
[13] model.predict([[4.0]])

     array([57189.])
```

```
[14] model.predict([[4.3]])

     array([56957.])
```

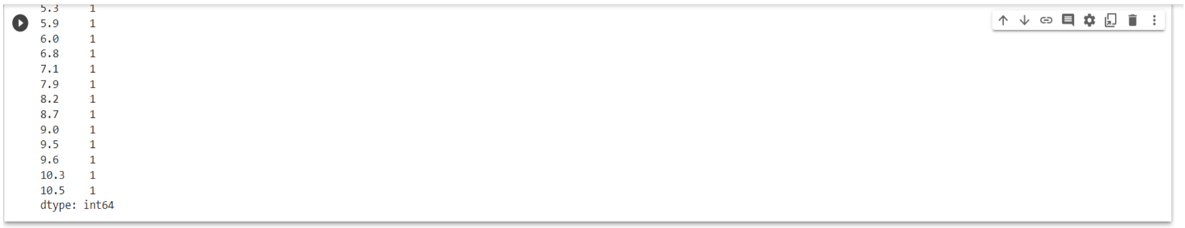
```
[15] model.predict([[4.5]])

     array([61111.])
```



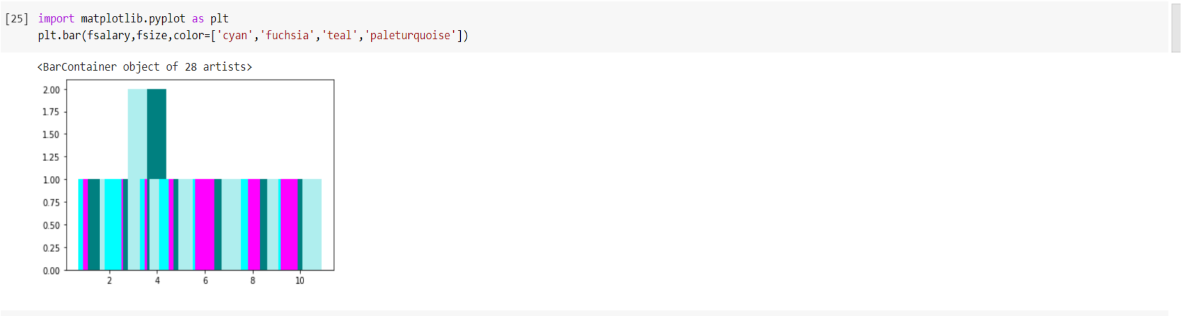
```
[16] df['YearsExperience'].nunique()

     28
```

```
[17] fsalary=df['YearsExperience'].unique()
     fsalary

     array([ 1.1,  1.3,  1.5,  2. ,  2.2,  2.9,  3. ,  3.2,  3.7,  3.9,  4. ,
             4.1,  4.5,  4.9,  5.1,  5.3,  5.9,  6. ,  6.8,  7.1,  7.9,  8.2,
             8.7,  9. ,  9.5,  9.6, 10.3, 10.5])
```

```
     fsize=df.groupby('YearsExperience').size()
     fsize

     YearsExperience
     1.1     1
     1.3     1
     1.5     1
     2.0     1
     2.2     1
     2.9     1
     3.0     1
     3.2     2
     3.7     1
     3.9     1
     4.0     2
     4.1     1
     4.5     1
     4.9     1
     5.1     1
     5.3     1
     5.9     1
     6.0     1
```

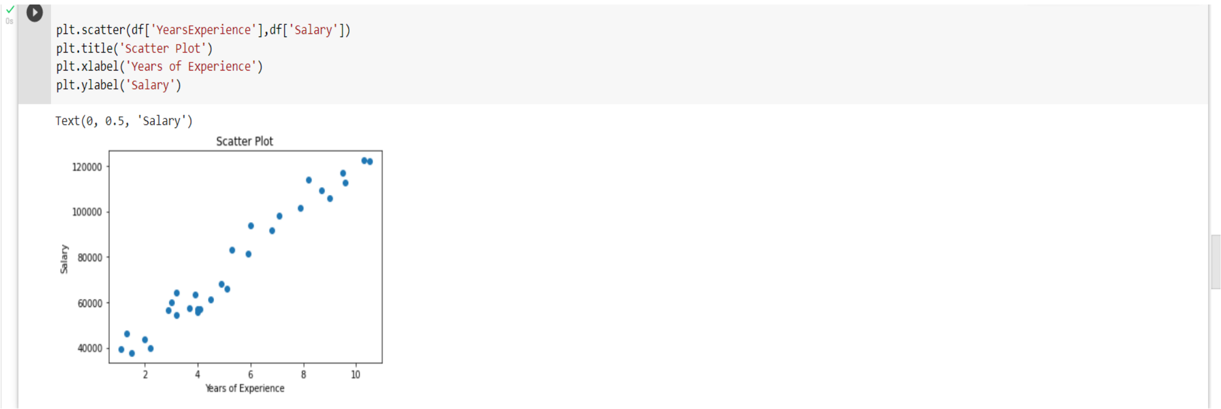


```
5.3     1
5.9     1
6.0     1
6.8     1
7.1     1
7.9     1
8.2     1
8.7     1
9.0     1
9.5     1
9.6     1
10.3    1
10.5    1
dtype: int64
```



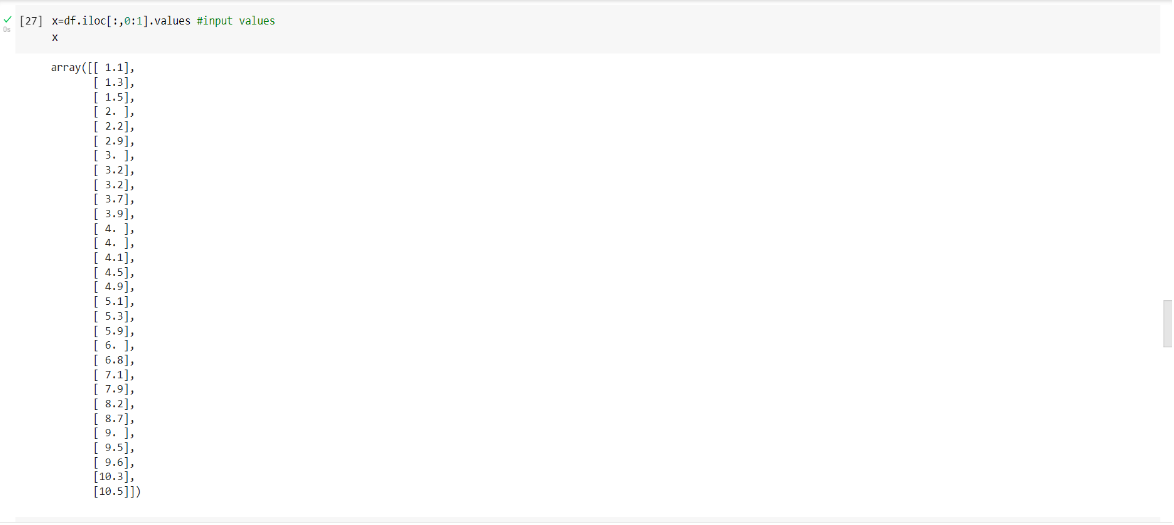
```
[25] import matplotlib.pyplot as plt
     plt.bar(fsalary,fsize,color=['cyan','fuchsia','teal','paleturquoise'])
```

```
<BarContainer object of 28 artists>
```



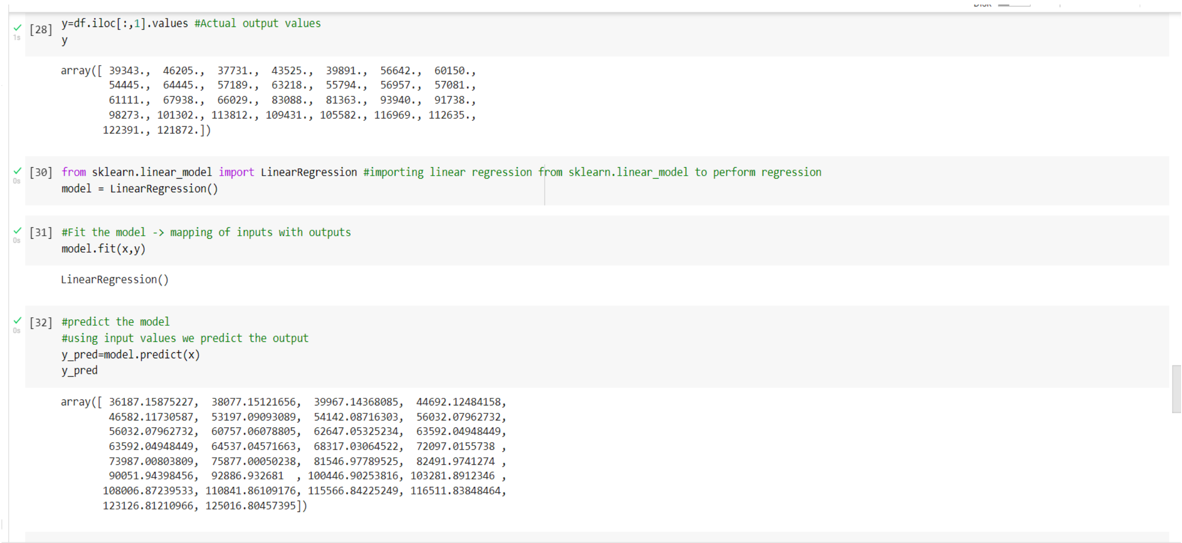
```
plt.scatter(df['YearsExperience'],df['Salary'])
plt.title('Scatter Plot')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
```

```
Text(0, 0.5, 'Salary')
```

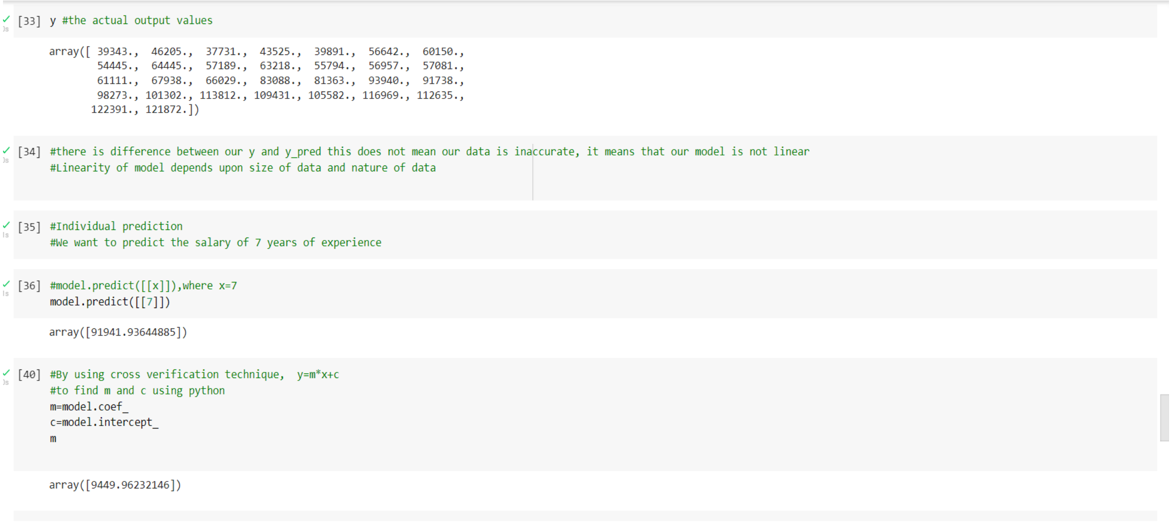


```
[27] x=df.iloc[:,0:1].values #input values
     x
```

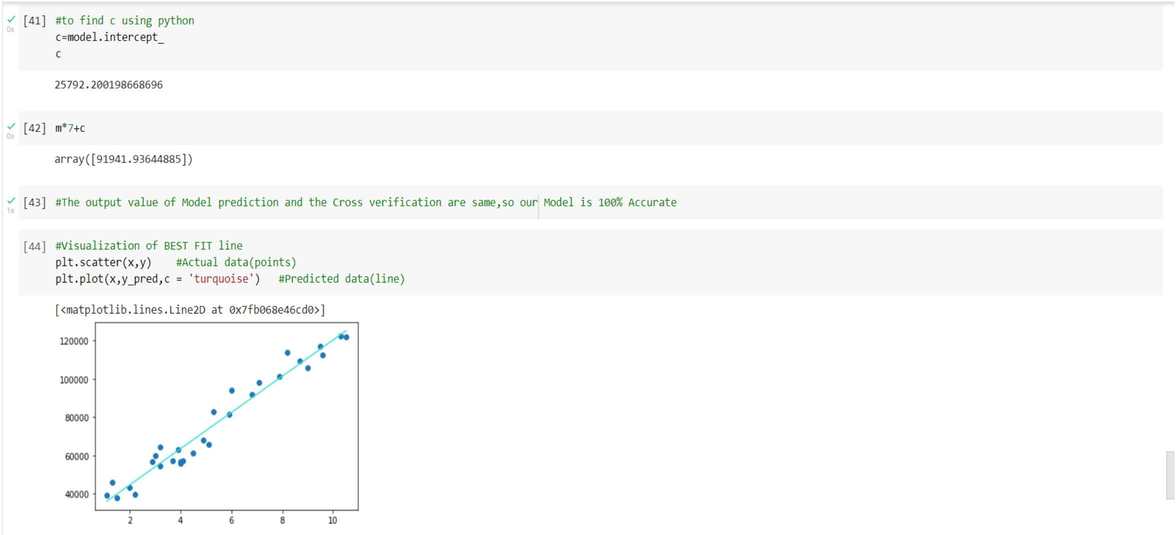
```
array([[ 1.1],
       [ 1.3],
       [ 1.5],
       [ 2. ],
       [ 2.2],
       [ 2.9],
       [ 3. ],
       [ 3.2],
       [ 3.2],
       [ 3.7],
       [ 3.9],
       [ 4. ],
       [ 4. ],
       [ 4.1],
       [ 4.5],
       [ 4.9],
       [ 5.1],
       [ 5.3],
       [ 5.9],
       [ 6. ],
       [ 6.8],
       [ 7.1],
       [ 7.9],
       [ 8.2],
       [ 8.7],
       [ 9. ],
       [ 9.5],
       [ 9.6],
       [10.3],
       [10.5]])
```



```
[28] y=df.iloc[:,1].values #Actual output values
     y

     array([ 39343.,  46205.,  37731.,  43525.,  39891.,  56642.,  60150.,
             54445.,  64445.,  57189.,  63218.,  55794.,  56957.,  57081.,
             61111.,  67938.,  66029.,  83088.,  81363.,  93940.,  91738.,
             98273., 101302., 113812., 109431., 105582., 116969., 112635.,
            122391., 121872.])

[30] from sklearn.linear_model import LinearRegression #importing linear regression from sklearn.linear_model to perform regression
     model = LinearRegression()

[31] #Fit the model -> mapping of inputs with outputs
     model.fit(x,y)

     LinearRegression()

[32] #predict the model
     #using input values we predict the output
     y_pred=model.predict(x)
     y_pred

     array([ 36187.15875227,  38077.15121656,  39967.14368085,  44692.12484158,
             46582.11730587,  53197.09093089,  54142.08716303,  56032.07962732,
             56032.07962732,  60757.06078805,  62647.05325234,  63592.04948449,
             63592.04948449,  64537.04571663,  68317.03064522,  72097.0155738 ,
             73987.00803809,  75877.00050238,  81546.97789525,  82491.9741274 ,
             90051.94398456,  92886.932681  , 100446.90253816, 103281.8912346 ,
            108006.87239533, 110841.86109176, 115566.84225249, 116511.83848464,
            123126.81210966, 125016.80457395])
```



```
[33] y #the actual output values

     array([ 39343.,  46205.,  37731.,  43525.,  39891.,  56642.,  60150.,
             54445.,  64445.,  57189.,  63218.,  55794.,  56957.,  57081.,
             61111.,  67938.,  66029.,  83088.,  81363.,  93940.,  91738.,
             98273., 101302., 113812., 109431., 105582., 116969., 112635.,
            122391., 121872.])

[34] #there is difference between our y and y_pred this does not mean our data is inaccurate, it means that our model is not linear
     #Linearity of model depends upon size of data and nature of data

[35] #Individual prediction
     #We want to predict the salary of 7 years of experience

[36] #model.predict([[x]]),where x=7
     model.predict([[7]])

     array([91941.93644885])

[40] #By using cross verification technique,  y=m*x+c
     #to find m and c using python
     m=model.coef_
     c=model.intercept_
     m

     array([9449.96232146])
```



```
[41] #to find c using python
     c=model.intercept_
     c

     25792.200198668696

[42] m*7+c

     array([91941.93644885])

[43] #The output value of Model prediction and the Cross verification are same,so our Model is 100% Accurate

[44] #Visualization of BEST FIT line
     plt.scatter(x,y)    #Actual data(points)
     plt.plot(x,y_pred,c = 'turquoise')   #Predicted data(line)

     [<matplotlib.lines.Line2D at 0x7fb068e46cd0>]
```

TEAM MEMBERS:

1.Aishwarya R

2.Asad Pangarkar

3.Sai kiran

4.Sarumathe Jayakumar

5.Shailendra Singh

6.Vivek Mangale