SLR with future salary prediction

Code

```
# SLR with future salary prediction

# import packages
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# call dataset and split data into iv and dv
ds = pd.read_csv(r'D:\1. Professionall\Data Science\08-11-2023\Salary_Data.csv')

X = ds.iloc[:, :-1].values
y = ds.iloc[:, 1].values

# Lets split the data into 80-20%

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state = 0)

# as the dataset has 2 attributes we use SLR algorithm

from sklearn.linear_model import LinearRegression

regressor = LinearRegression()

# Built regression model
regressor.fit(X_train, y_train)

# Test the model and prepare future prediction table
y_pred = regressor.predict(X_test)

# visualize train data point ( 24 data)
plt.scatter(X_train, y_train, color = 'red')
plt.plot(X_train, regressor.predict(X_train), color = 'blue')
plt.title('Salary vs Experience (Training set)')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.show()

```

```python
# visulaize test data point
plt.scatter(X_test, y_test, color = 'red')
plt.plot(X_train, regressor.predict(X_train), color = 'blue')
plt.title('Salary vs Experience (Testing set)')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.show()

# To get value of slope
m = regressor.coef_
m

# To get intercept or constant
c = regressor.intercept_
c

# predict or forcast the future the data which we not trained before
y_12 = int(m) * 12 + c
y_12

y_20 = int(m) * 20 + c
y_20


# to check overfitting  ( low bias high variance)
bias = regressor.score(X_train, y_train)
bias


# to check underfitting (high bias low variance)
variance = regressor.score(X_test,y_test)
variance
```

Dataset

ds - DataFrame

Index	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2	43525
4	2.2	39891
5	2.9	56642
6	3	60150
7	3.2	54445
8	3.2	64445
9	3.7	57189
10	3.9	63218

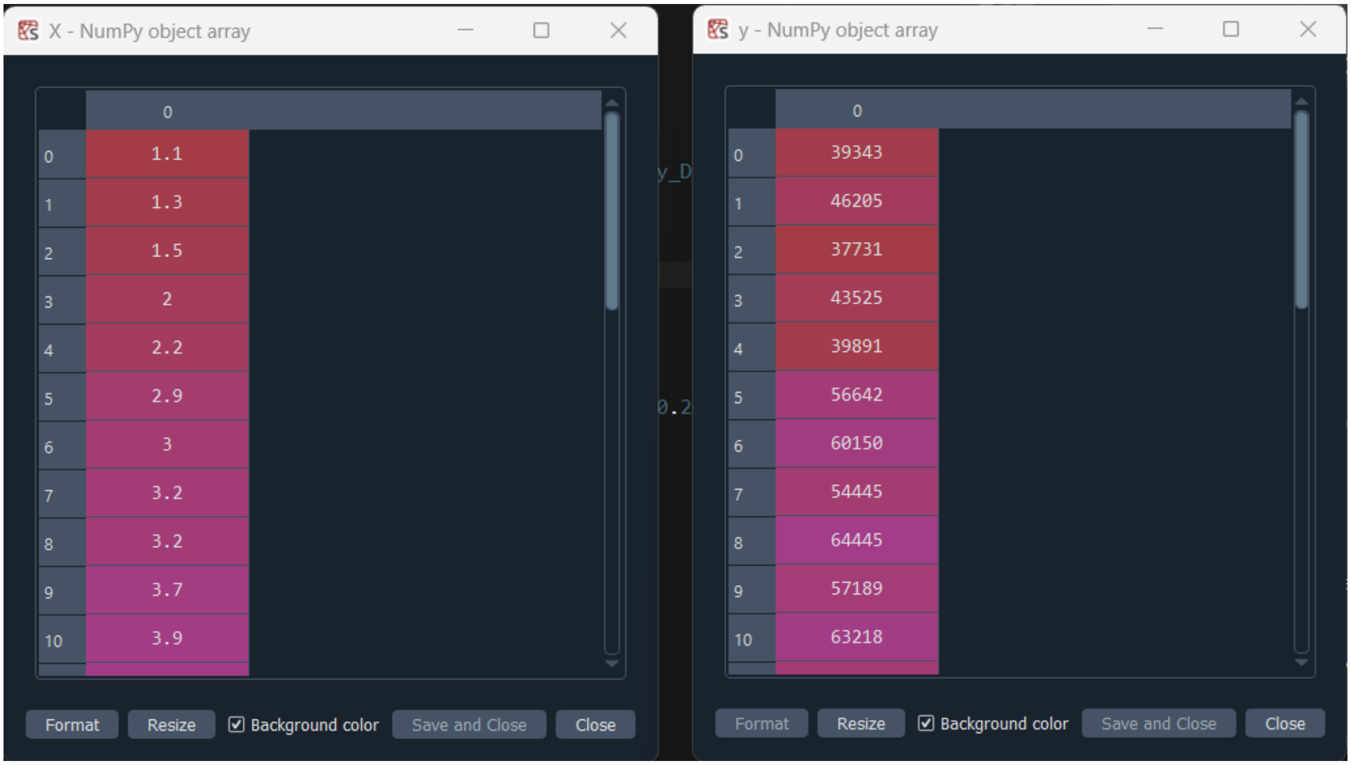
Format | Resize | ☑ Background color | ☑ Column min/max | Save and Close | Close

ds - DataFrame

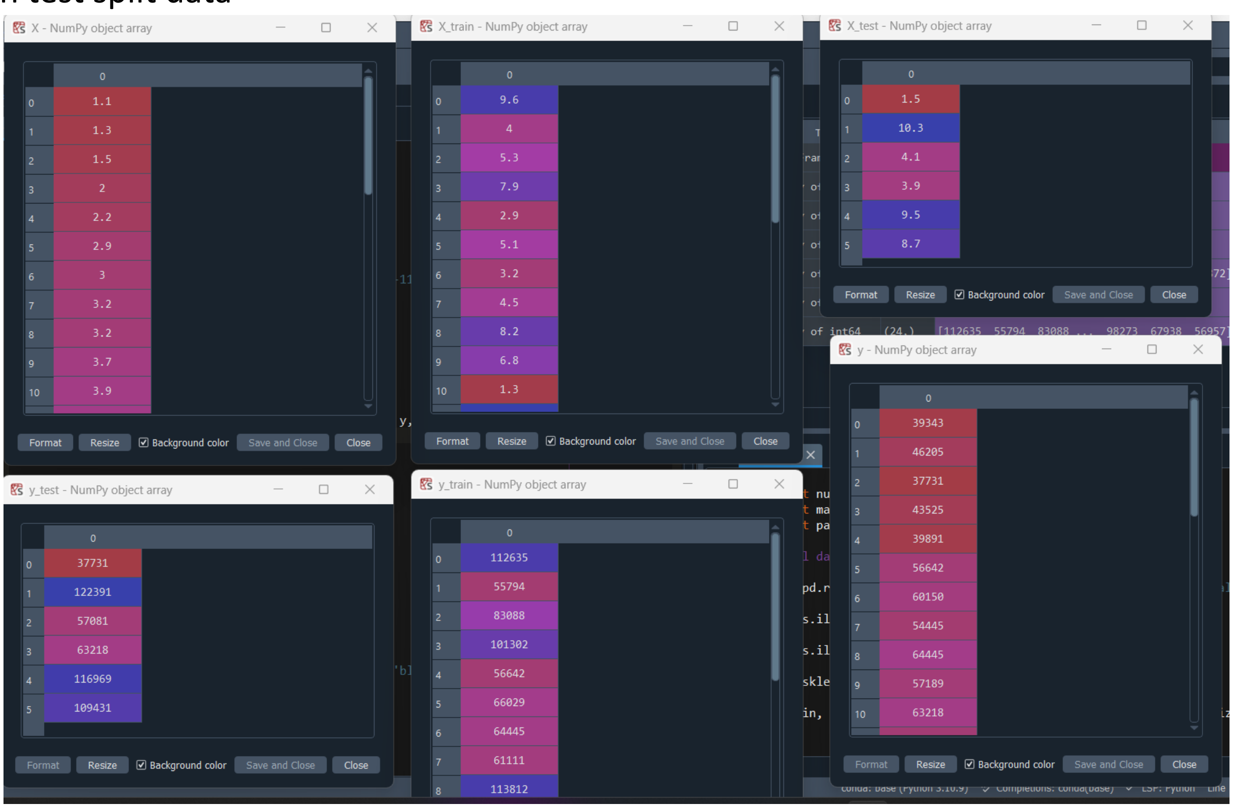
Index	YearsExperience	Salary
19	6	93940
20	6.8	91738
21	7.1	98273
22	7.9	101302
23	8.2	113812
24	8.7	109431
25	9	105582
26	9.5	116969
27	9.6	112635
28	10.3	122391
29	10.5	121872

Format | Resize | ☑ Background color | ☑ Column min/max | Save and Close | Close

X & y data sets



Train test split data



80 – 20 % split results


```
In [19]: m = regressor.coef_
    ...: m
Out[19]: array([9312.57512673])

In [20]: c = regressor.intercept_
    ...: c
Out[20]: 26780.099150628186

In [21]: y_12 = int(m) * 12 + c
    ...: y_12
Out[21]: 138524.0991506282

In [22]: y_20 = int(m) * 20 + c
    ...: y_20
Out[22]: 213020.0991506282

In [23]:
```

```
In [23]: bias = regressor.score(X_train, y_train)
    ...: bias
Out[23]: 0.9411949620562126

In [24]: variance = regressor.score(X_test,y_test)
    ...: variance
Out[24]: 0.988169515729126

In [25]:
```

75 – 25 % split results


```
In [13]: m = regressor.coef_
    ...: m
Out[13]: array([9379.71049195])

In [14]: c = regressor.intercept_
    ...: c
Out[14]: 26986.691316737248

In [15]: y_12 = int(m) * 12 + c
    ...: y_12
Out[15]: 139534.69131673724

In [16]: y_20 = int(m) * 20 + c
    ...: y_20
Out[16]: 214566.69131673724

In [17]:
```

```
In [17]: bias = regressor.score(X_train, y_train)
    ...: bias
Out[17]: 0.9395413526983522

In [18]: variance = regressor.score(X_test,y_test)
    ...: variance
Out[18]: 0.9779208335417602

In [19]:
```

70 – 30 % split results


```
In [13]: m = regressor.coef_
    ...: m
Out[13]: array([9360.26128619])

In [14]: c = regressor.intercept_
    ...: c
Out[14]: 26777.391341197632

In [15]: y_12 = int(m) * 12 + c
    ...: y_12
Out[15]: 139097.39134119762

In [16]: y_20 = int(m) * 20 + c
    ...: y_20
Out[16]: 213977.39134119762

In [17]:
```

```
In [17]: bias = regressor.score(X_train, y_train)
    ...: bias
Out[17]: 0.9423777652193379

In [18]: variance = regressor.score(X_test,y_test)
    ...: variance
Out[18]: 0.9740993407213511

In [19]:
```