Name: Sangeeth Kumar V
Register No: 16BIS0072

Artificial Intelligence with Python

Lab Task – 01 (L39 & L40)

Prof Hemprasad Yashwant Patil

Question:

Import the Salary dataset from kaggle website (https://www.kaggle.com/karthickveerakumar/salary-data-simple-linear-regression) to your python environmet. This dataset contanis columns namely 'YearsofExperience' and 'Salary'. Train a linear regression classifier to fit the data. Use Salary as a target column (y).

Form the models with the following configurations
(1) 50% Train, 50% Test
(2) 70% Train, 30% Test
(3) 80% Train, 20% Test
Analyze the MSE, RMSE for each configuration.

Write your own function to determine MSE/RMSE. Crosscheck it with the given by sklearn metrics.

Dataset:

YearsExperience	**Salary**	**YearsExperience**	**Salary**	**YearsExperience**	**Salary**
1.1	39343	3.9	63218	6.8	91738
1.3	46205	4	55794	7.1	98273
1.5	37731	4	56957	7.9	101302
2	43525	4.1	57081	8.2	113812
2.2	39891	4.5	61111	8.7	109431
2.9	56642	4.9	67938	9	105582
3	60150	5.1	66029	9.5	116969
3.2	54445	5.3	83088	9.6	112635
3.2	64445	5.9	81363	10.3	122391
3.7	57189	6	93940	10.5	121872

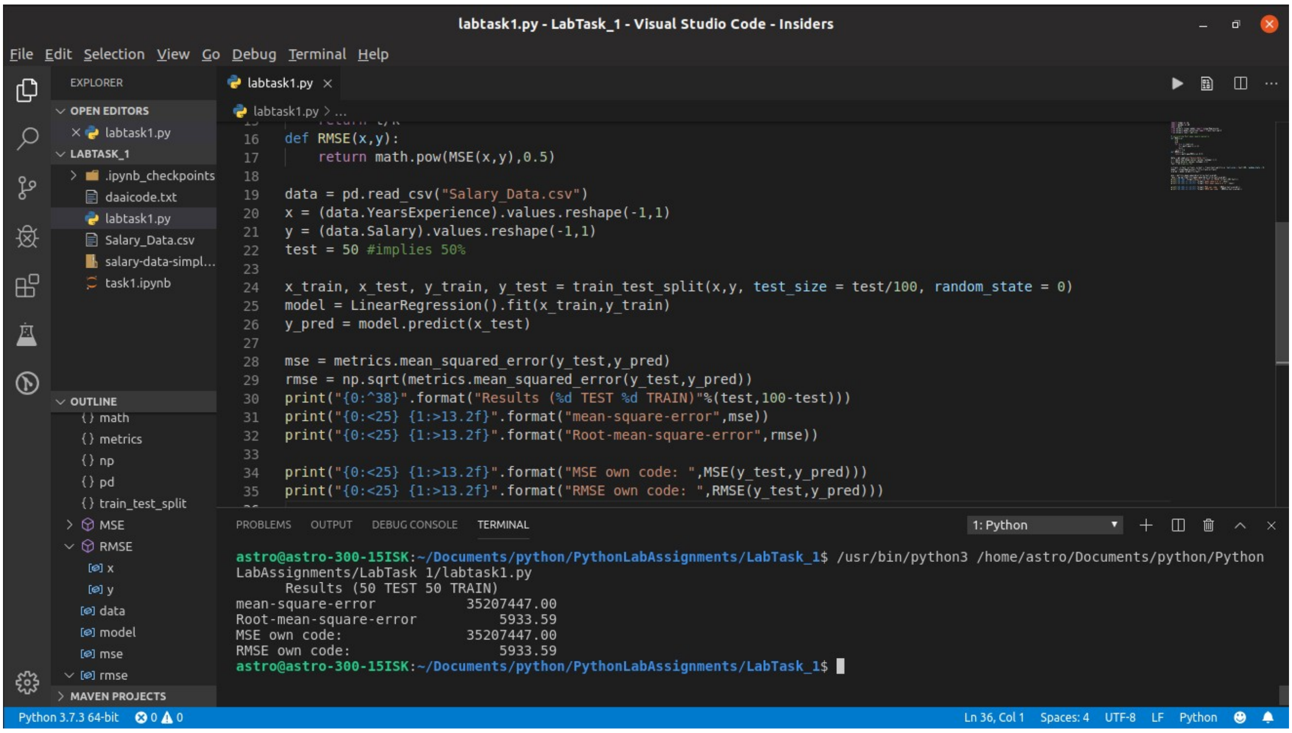
Code Used:

```
import numpy as np
import pandas as pd
import math
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn import metrics

# calculating Root-mean square manually
def MSE(x,y):
    t=0
    k=0
    for i,j in zip(x,y):
        t+= math.pow((i-j),2)
        k+=1
    return t/k
def RMSE(x,y):
    return math.pow(MSE(x,y),0.5)
```

```
data = pd.read_csv("Salary_Data.csv")
x = (data.YearsExperience).values.reshape(-1,1)
y = (data.Salary).values.reshape(-1,1)
test = 50 #implies 50%

x_train, x_test, y_train, y_test = train_test_split(x,y, test_size
= test/100, random_state = 0)
model = LinearRegression().fit(x_train,y_train)
y_pred = model.predict(x_test)

mse = metrics.mean_squared_error(y_test,y_pred)
rmse = np.sqrt(metrics.mean_squared_error(y_test,y_pred))
print("{0:^38}".format("Results (%d TEST %d TRAIN)"%(test,100-
test)))
print("{0:<25} {1:>13.2f}".format("mean-square-error",mse))
print("{0:<25} {1:>13.2f}".format("Root-mean-square-error",rmse))

print("{0:<25} {1:>13.2f}".format("MSE own code:
",MSE(y_test,y_pred)))
print("{0:<25} {1:>13.2f}".format("RMSE own code:
",RMSE(y_test,y_pred)))
```

Running code under the given configurations mentioned in the question with different percentage of training and testing datasets from the given dataset.

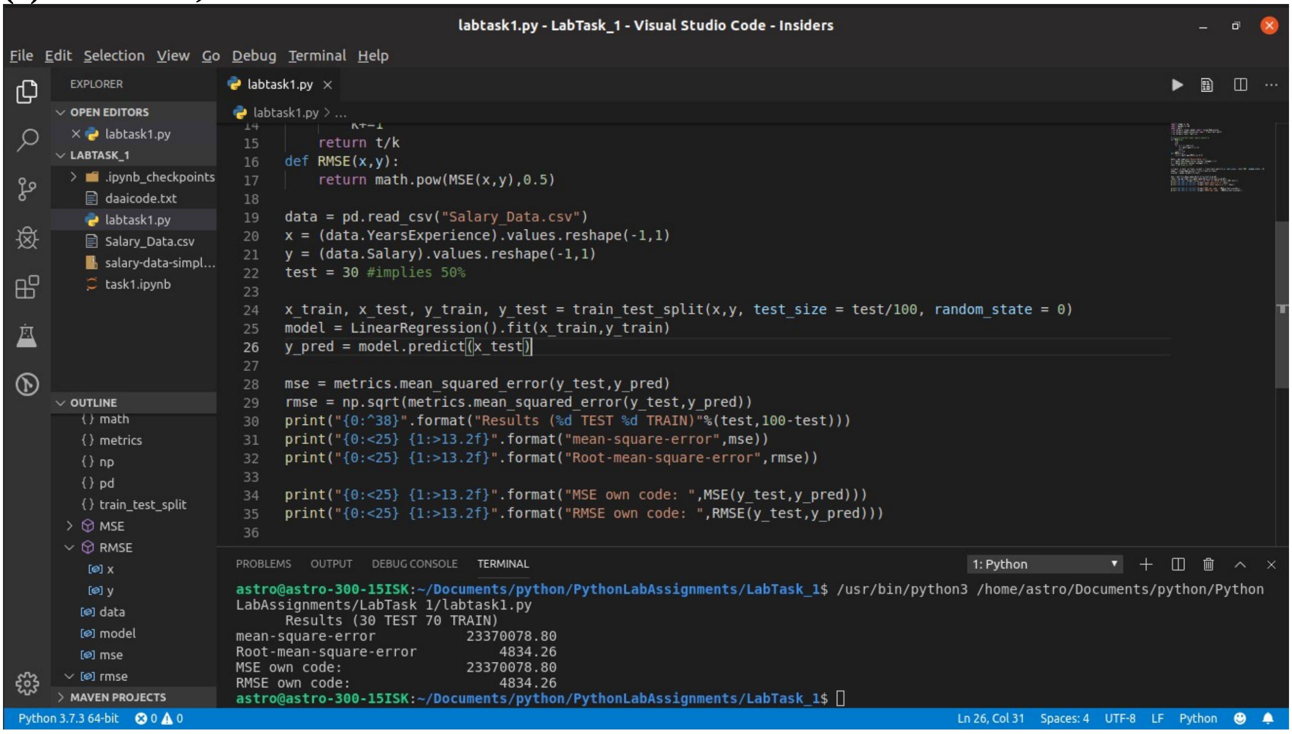
(1) 50% Train, 50% Test



```
def RMSE(x,y):
    return math.pow(MSE(x,y),0.5)

data = pd.read_csv("Salary_Data.csv")
x = (data.YearsExperience).values.reshape(-1,1)
y = (data.Salary).values.reshape(-1,1)
test = 50 #implies 50%

x_train, x_test, y_train, y_test = train_test_split(x,y, test_size = test/100, random_state = 0)
model = LinearRegression().fit(x_train,y_train)
y_pred = model.predict(x_test)

mse = metrics.mean_squared_error(y_test,y_pred)
rmse = np.sqrt(metrics.mean_squared_error(y_test,y_pred))
print("{0:^38}".format("Results (%d TEST %d TRAIN)"%(test,100-test)))
print("{0:<25} {1:>13.2f}".format("mean-square-error",mse))
print("{0:<25} {1:>13.2f}".format("Root-mean-square-error",rmse))

print("{0:<25} {1:>13.2f}".format("MSE own code: ",MSE(y_test,y_pred)))
print("{0:<25} {1:>13.2f}".format("RMSE own code: ",RMSE(y_test,y_pred)))
```

```
astro@astro-300-15ISK:~/Documents/python/PythonLabAssignments/LabTask_1$ /usr/bin/python3 /home/astro/Documents/python/Python
LabAssignments/LabTask 1/labtask1.py
      Results (50 TEST 50 TRAIN)
mean-square-error            35207447.00
Root-mean-square-error           5933.59
MSE own code:                35207447.00
RMSE own code:                   5933.59
astro@astro-300-15ISK:~/Documents/python/PythonLabAssignments/LabTask_1$
```

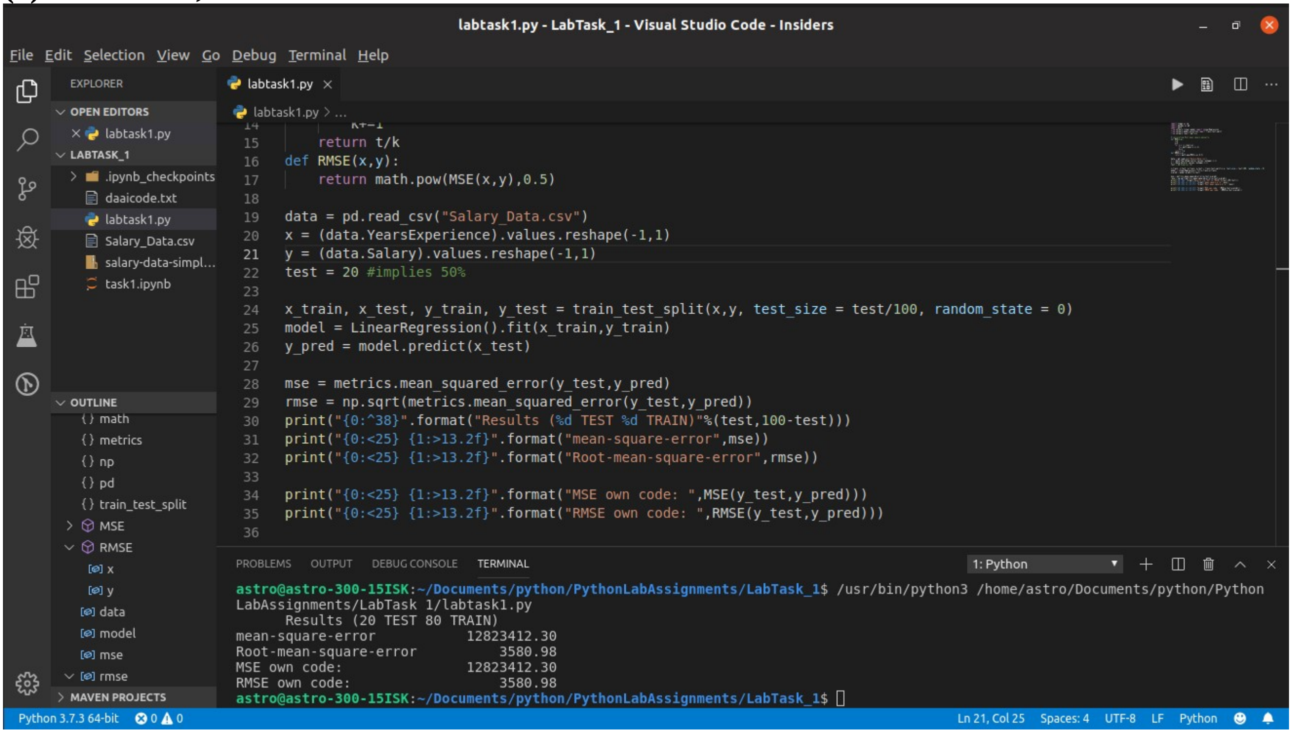
(2) 70% Train, 30% Test



```python
        k+=1
    return t/k
def RMSE(x,y):
    return math.pow(MSE(x,y),0.5)

data = pd.read_csv("Salary_Data.csv")
x = (data.YearsExperience).values.reshape(-1,1)
y = (data.Salary).values.reshape(-1,1)
test = 30 #implies 50%

x_train, x_test, y_train, y_test = train_test_split(x,y, test_size = test/100, random_state = 0)
model = LinearRegression().fit(x_train,y_train)
y_pred = model.predict(x_test)

mse = metrics.mean_squared_error(y_test,y_pred)
rmse = np.sqrt(metrics.mean_squared_error(y_test,y_pred))
print("{0:^38}".format("Results (%d TEST %d TRAIN)"%(test,100-test)))
print("{0:<25} {1:>13.2f}".format("mean-square-error",mse))
print("{0:<25} {1:>13.2f}".format("Root-mean-square-error",rmse))

print("{0:<25} {1:>13.2f}".format("MSE own code: ",MSE(y_test,y_pred)))
print("{0:<25} {1:>13.2f}".format("RMSE own code: ",RMSE(y_test,y_pred)))
```

```
astro@astro-300-15ISK:~/Documents/python/PythonLabAssignments/LabTask_1$ /usr/bin/python3 /home/astro/Documents/python/Python
LabAssignments/LabTask_1/labtask1.py
       Results (30 TEST 70 TRAIN)
mean-square-error            23370078.80
Root-mean-square-error           4834.26
MSE own code:                23370078.80
RMSE own code:                   4834.26
astro@astro-300-15ISK:~/Documents/python/PythonLabAssignments/LabTask_1$
```

(3) 80% Train, 20% Test



```python
        k+=1
    return t/k
def RMSE(x,y):
    return math.pow(MSE(x,y),0.5)

data = pd.read_csv("Salary_Data.csv")
x = (data.YearsExperience).values.reshape(-1,1)
y = (data.Salary).values.reshape(-1,1)
test = 20 #implies 50%

x_train, x_test, y_train, y_test = train_test_split(x,y, test_size = test/100, random_state = 0)
model = LinearRegression().fit(x_train,y_train)
y_pred = model.predict(x_test)

mse = metrics.mean_squared_error(y_test,y_pred)
rmse = np.sqrt(metrics.mean_squared_error(y_test,y_pred))
print("{0:^38}".format("Results (%d TEST %d TRAIN)"%(test,100-test)))
print("{0:<25} {1:>13.2f}".format("mean-square-error",mse))
print("{0:<25} {1:>13.2f}".format("Root-mean-square-error",rmse))

print("{0:<25} {1:>13.2f}".format("MSE own code: ",MSE(y_test,y_pred)))
print("{0:<25} {1:>13.2f}".format("RMSE own code: ",RMSE(y_test,y_pred)))
```

```
astro@astro-300-15ISK:~/Documents/python/PythonLabAssignments/LabTask_1$ /usr/bin/python3 /home/astro/Documents/python/Python
LabAssignments/LabTask_1/labtask1.py
       Results (20 TEST 80 TRAIN)
mean-square-error            12823412.30
Root-mean-square-error           3580.98
MSE own code:                12823412.30
RMSE own code:                   3580.98
astro@astro-300-15ISK:~/Documents/python/PythonLabAssignments/LabTask_1$
```

Comparing the results obtained

S.no	Test data percentage	MSE	RMSE
1	50	35207447.00	5933.59
2	30	23370078.80	4834.26
3	20	12823412.30	3580.98

Conclusion

From the table we can realize that RMSE and MSE decreases as the training data increases. In order to obtain sufficient accuracy for our training model we requre much more data, thus it reduces the MSE/RMSE thereby increasin the accuracy of our model.