Predict Weight grain

October 16, 2021

1 To Predict Weight Gained using sorting time

1.0.1 Importing the Libraries

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

[2]:
```
from sklearn.model_selection import train_test_split
### Splitting the dataset into the Training set and test set
```

[3]:
```
from sklearn.linear_model import LinearRegression
## Training the simple linear Regression Model
```

[4]:
```
import statsmodels.api as sm
```

1.0.2 Importing the dataset

[5]:
```
data_sets = pd.read_csv("calories_consumed.csv")
```

[6]:
```
x = data_sets.iloc[:,:-1].values
y = data_sets.iloc[:, -1].values
```

[7]:
```
print(data_sets)
```

```
    Weight_gained_grams  Calories_Consumed
0                   108               1500
1                   200               2300
2                   900               3400
3                   200               2200
4                   300               2500
5                   110               1600
6                   128               1400
7                    62               1900
8                   600               2800
9                  1100               3900
10                  100               1670
11                  150               1900
```

```
12                  350                 2700
13                  700                 3000
```

[8]:
```
print(x)
```

```
[[ 108]
 [ 200]
 [ 900]
 [ 200]
 [ 300]
 [ 110]
 [ 128]
 [  62]
 [ 600]
 [1100]
 [ 100]
 [ 150]
 [ 350]
 [ 700]]
```

[9]:
```
print(y)
```

```
[1500 2300 3400 2200 2500 1600 1400 1900 2800 3900 1670 1900 2700 3000]
```

1.0.3 Splitting the dataset into the Training set and test set

[10]:
```
data_sets.describe()
```

[10]:

	Weight_gained_grams	Calories_Consumed
count	14.000000	14.000000
mean	357.714286	2340.714286
std	333.692495	752.109488
min	62.000000	1400.000000
25%	114.500000	1727.500000
50%	200.000000	2250.000000
75%	537.500000	2775.000000
max	1100.000000	3900.000000

[11]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.2,␣
↪random_state= 0)
```

1.0.4 Training the simple linear Regression Model on the Training set

[12]:
```
regressor = LinearRegression()
regressor.fit(x_train,y_train)
```

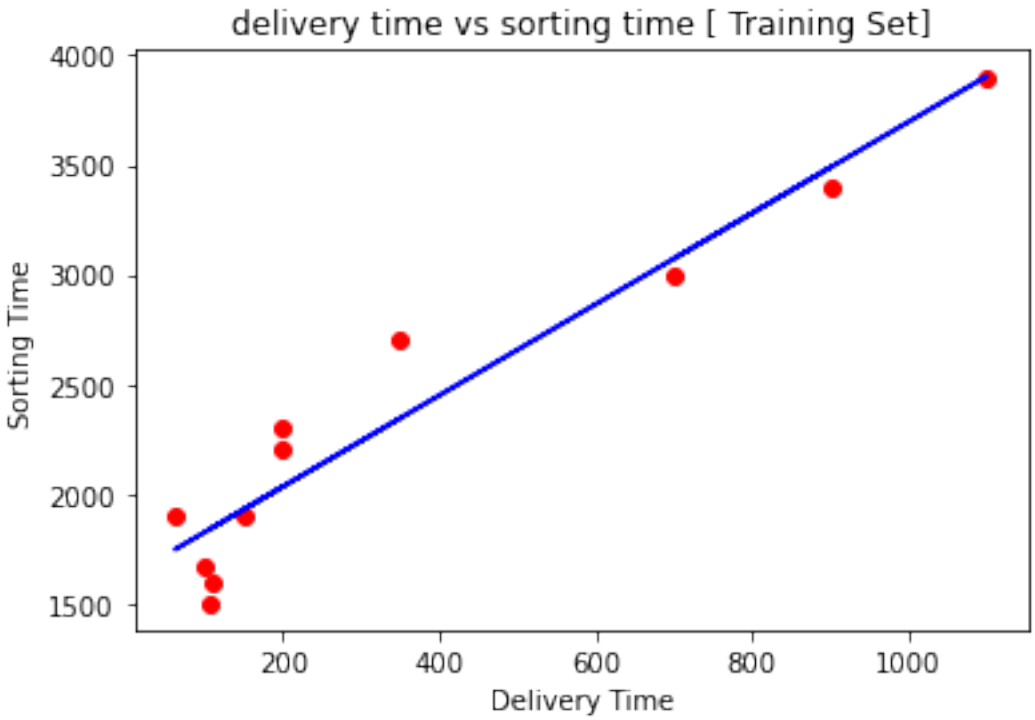
[12]:
```
LinearRegression()
```

1.0.5 Predicting the Test set result

[13]:
```
y_predict = regressor.predict(x_test)
```

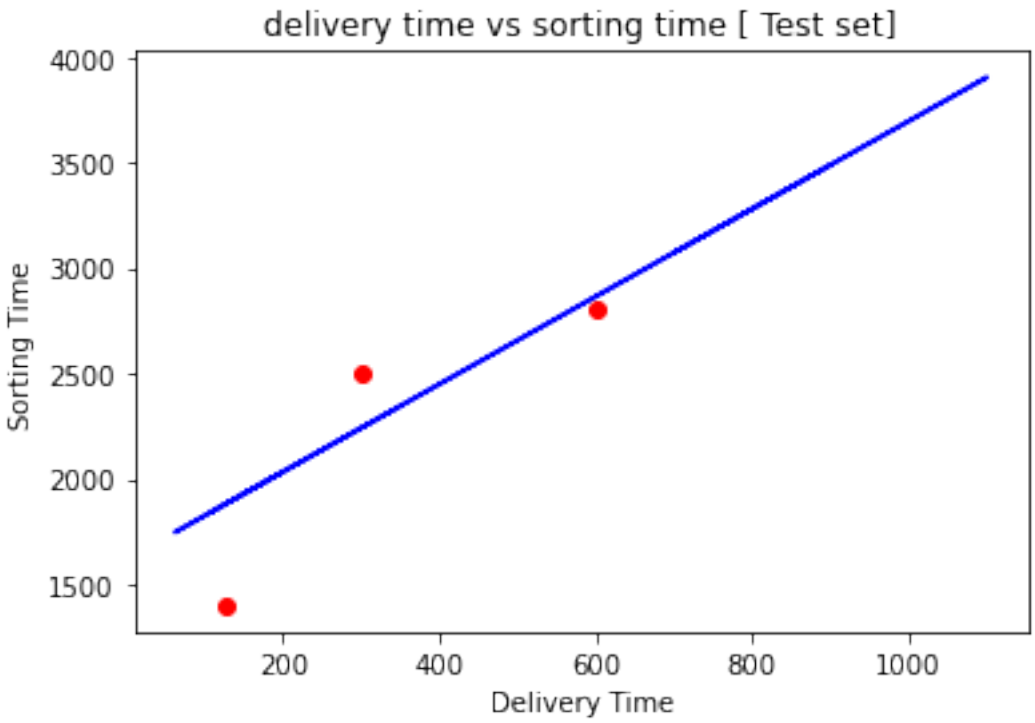
1.0.6 Visualising the Training set results

[14]:
```
plt.scatter(x_train, y_train, color = 'red')
plt.plot(x_train, regressor.predict(x_train), color ="blue")
plt.title("delivery time vs sorting time [ Training Set]")
plt.xlabel("Delivery Time")
plt.ylabel("Sorting Time")
plt.show()
```



1.0.7 Visualising the test set result

[15]:
```
plt.scatter(x_test, y_test, color = 'red')
plt.plot(x_train, regressor.predict(x_train), color ="blue")
plt.title("delivery time vs sorting time [ Test set]")
plt.xlabel("Delivery Time")
plt.ylabel("Sorting Time")
plt.show()
```



1.0.8 Regression Itself

[16]:
```
x_stats = sm.add_constant(x)
results = sm.OLS(y,x_stats).fit()
results.summary()
```

```
C:\Users\maitr\anaconda3\lib\site-packages\scipy\stats\stats.py:1603:
UserWarning: kurtosistest only valid for n>=20 … continuing anyway, n=14
  warnings.warn("kurtosistest only valid for n>=20 … continuing "
```

[16]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.897
Model:                            OLS   Adj. R-squared:                  0.888
Method:                 Least Squares   F-statistic:                     104.3
Date:                Sat, 16 Oct 2021   Prob (F-statistic):           2.86e-07
Time:                        15:37:48   Log-Likelihood:                -96.170
No. Observations:                  14   AIC:                             196.3
Df Residuals:                      12   BIC:                             197.6
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
```

```
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       1577.2007    100.541     15.687      0.000    1358.141    1796.260
x1             2.1344      0.209     10.211      0.000       1.679       2.590
==============================================================================
Omnibus:                        0.254   Durbin-Watson:                   2.308
Prob(Omnibus):                  0.881   Jarque-Bera (JB):                0.425
Skew:                          -0.098   Prob(JB):                        0.808
Kurtosis:                       2.169   Cond. No.                         719.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
"""
```