Steps for Regression Model with Sklearn

Basic steps for regression model with sklearn

Import Library

In [30]:

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

import CSV as DataFrame

In [31]:

```
df = pd.read_csv(r"https://github.com/YBI-Foundation/Dataset/raw/main/Fish.csv")
```

Get first five rows of DataFrame

In [32]:

```
df.head()
```

Out[32]:

	Category	Species	Weight	Height	Width	Length1	Length2	Length3
0	1	Bream	242.0	11.5200	4.0200	23.2	25.4	30.0
1	1	Bream	290.0	12.4800	4.3056	24.0	26.3	31.2
2	1	Bream	340.0	12.3778	4.6961	23.9	26.5	31.1
3	1	Bream	363.0	12.7300	4.4555	26.3	29.0	33.5
4	1	Bream	430.0	12.4440	5.1340	26.5	29.0	34.0

Get information of Dataframe

In [33]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 159 entries, 0 to 158
Data columns (total 8 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Category  159 non-null    int64  
 1   Species   159 non-null    object 
 2   Weight    159 non-null    float64
 3   Height    159 non-null    float64
 4   Width     159 non-null    float64
 5   Length1   159 non-null    float64
 6   Length2   159 non-null    float64
 7   Length3   159 non-null    float64
dtypes: float64(6), int64(1), object(1)
memory usage: 10.1+ KB
```

Get the summary of Statistics

In [34]:

```
df.describe()
```

Out[34]:

	Category	Weight	Height	Width	Length1	Length2	Length3
count	159.000000	159.000000	159.000000	159.000000	159.000000	159.000000	159.000000
mean	3.264151	398.326415	8.970994	4.417486	26.247170	28.415723	31.227044
std	1.704249	357.978317	4.286208	1.685804	9.996441	10.716328	11.610246
min	1.000000	0.000000	1.728400	1.047600	7.500000	8.400000	8.800000
25%	2.000000	120.000000	5.944800	3.385650	19.050000	21.000000	23.150000
50%	3.000000	273.000000	7.786000	4.248500	25.200000	27.300000	29.400000
75%	4.500000	650.000000	12.365900	5.584500	32.700000	35.500000	39.650000
max	7.000000	1650.000000	18.957000	8.142000	59.000000	63.400000	68.000000

Get Shape of Dataframe

In [35]:

```
df.shape
```

Out[35]:

```
(159, 8)
```

Get Columns Name

In [36]:

```
df.columns
```

Out[36]:

```
Index(['Category', 'Species', 'Weight', 'Height', 'Width', 'Length1',
       'Length2', 'Length3'],
      dtype='object')
```

Define y(dependent or label or target variable) and X(Independent or feature or attribute variable)

In [37]:

```
y = df['Weight']
```

In [38]:

```
y.shape
```

Out[38]:

```
(159,)
```

In [39]:

```
y
```

Out[39]:

```
0      242.0
1      290.0
2      340.0
3      363.0
4      430.0
       ...
154     12.2
155     13.4
156     12.2
157     19.7
158     19.9
Name: Weight, Length: 159, dtype: float64
```

In [40]:

```
X = df[['Height', 'Width', 'Length1','Length2', 'Length3']]
```

In [41]:

```
X.shape
```

Out[41]:

```
(159, 5)
```

In [42]:

```
X
```

Out[42]:

	Height	Width	Length1	Length2	Length3
0	11.5200	4.0200	23.2	25.4	30.0
1	12.4800	4.3056	24.0	26.3	31.2
2	12.3778	4.6961	23.9	26.5	31.1
3	12.7300	4.4555	26.3	29.0	33.5
4	12.4440	5.1340	26.5	29.0	34.0
...	...	...	...	...	...
154	2.0904	1.3936	11.5	12.2	13.4
155	2.4300	1.2690	11.7	12.4	13.5
156	2.2770	1.2558	12.1	13.0	13.8
157	2.8728	2.0672	13.2	14.3	15.2
158	2.9322	1.8792	13.8	15.0	16.2

159 rows × 5 columns

In [43]:

```
X = df.drop(['Category', 'Species', 'Weight'], axis=1)
```

In [44]:

```
X.shape
```

Out[44]:

```
(159, 5)
```

In [45]:

```
X
```

Out[45]:

	Height	Width	Length1	Length2	Length3
0	11.5200	4.0200	23.2	25.4	30.0
1	12.4800	4.3056	24.0	26.3	31.2
2	12.3778	4.6961	23.9	26.5	31.1
3	12.7300	4.4555	26.3	29.0	33.5
4	12.4440	5.1340	26.5	29.0	34.0
...	...	...	...	...	...
154	2.0904	1.3936	11.5	12.2	13.4
155	2.4300	1.2690	11.7	12.4	13.5
156	2.2770	1.2558	12.1	13.0	13.8
157	2.8728	2.0672	13.2	14.3	15.2
158	2.9322	1.8792	13.8	15.0	16.2

159 rows × 5 columns

Get Train-test-split

In [46]:

```
from sklearn.model_selection import train_test_split
```

In [47]:

```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3, random_state=2254)
```

In [48]:

```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

Out[48]:

```
((111, 5), (48, 5), (111,), (48,))
```

In [49]:

```
from sklearn.linear_model import LinearRegression
```

In [50]:

```
model = LinearRegression()
```

In [51]:

```
model.fit(X_train,y_train)
```

Out[51]:

```
LinearRegression()
```

Get Model Predict

In [52]:

```
y_pred = model.predict(X_test)
```

In [53]:

```
y_pred.shape
```

Out[53]:

```
(48,)
```

In [54]:

```
y_pred
```

Out[54]:

```
array([  97.60167573,  207.54843243,  780.99118651,  314.35487958,
        184.0777877 ,  264.00423771,  910.99010236,  675.57027966,
       -142.73337046,  136.4269669 ,  366.75314755,  148.83449827,
        854.9511332 , -156.28548494,  603.99701666,  772.93235553,
        805.87775427,  -73.78143302,  107.69984752,  336.836344  ,
        141.18908263,  540.15716962,  269.71879309,  557.81810424,
        537.78191451, 1069.34781139,  140.95436387,  851.50030651,
        140.661948  ,  544.52810125,  168.05667942,   34.51185187,
        211.52578448,  222.41188512, 1173.29180627, -140.50715608,
        685.82296775,  168.59291828,  116.85419698,   48.06538509,
        804.04105263,  173.80858453,  656.36650026,  688.94027394,
        745.79272923,  389.38869184,  379.89889292,  650.11124831])
```

Get Model Evaluation

In [55]:

```
from sklearn.metrics import mean_squared_error, mean_absolute_error, mean_absolute_percenta
```

In [56]:

```
mean_squared_error(y_test,y_pred)
```

Out[56]:

```
21543.221470572124
```

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

MSE = mean squared error
n = number of data points
Yi = observed values
hat(Yi) = predicted values

In [57]:

```
mean_absolute_error(y_test,y_pred)
```

Out[57]:

```
101.58037604551602
```

$$\text{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

MAE = mean absolute error
yi = prediction
xi = true value
n = total number of data points

In [58]:

```
mean_absolute_percentage_error(y_test,y_pred)
```

Out[58]:

```
1.2883877664090626
```

$$M = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right|$$

M = mean absolute percentage error
n = number of times the summation iteration happens
A_t = actual value
F_t = forecast value

In [60]:

```
r2_score(y_test, y_pred)
```

Out[60]:

```
0.870932399883171
```

$$R^2 = 1 - \frac{RSS}{TSS}$$

R^2 = coefficient of determination
RSS = sum of squares of residuals
TSS = total sum of squares

Get Future Predictions

Lets select a random sample from existing dataset as new value

Steps to follow
1. Extract a random row using sample function
2. Separate X and y
3. Predict

In [61]:

```
df_new = df.sample(1)
```

In [62]:

```
df_new
```

Out[62]:

	Category	Species	Weight	Height	Width	Length1	Length2	Length3
34	1	Bream	950.0	17.6235	6.3705	38.0	41.0	46.5

In [63]:

```
X_new = df_new[['Height','Width','Length1','Length2','Length3']]
```

In [64]:

```
X_new.shape
```

Out[64]:

```
(1, 5)
```

In [65]:

```
y_pred_new = model.predict(X_new)
```

In [66]:

```
y_pred_new
```

Out[66]:

```
array([884.8896656])
```

In [ ]:

```

```