```
#Steps for regression model with statistics

#import library
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt
```

Double-click (or enter) to edit

```
#import csv as pd

df=pd.read_csv('/content/New_Fish_dataset.zip')

df
```

	Species	Weight	Length1	Length2	Length3	Height	Width
0	Bream	242.0	23.2	25.4	30.0	11.5200	4.0200
1	Bream	290.0	24.0	26.3	31.2	12.4800	4.3056
2	Bream	340.0	23.9	26.5	31.1	12.3778	4.6961
3	Bream	363.0	26.3	29.0	33.5	12.7300	4.4555
4	Bream	430.0	26.5	29.0	34.0	12.4440	5.1340
...	...	...	...	...	...	...	...
154	Smelt	12.2	11.5	12.2	13.4	2.0904	1.3936
155	Smelt	13.4	11.7	12.4	13.5	2.4300	1.2690
156	Smelt	12.2	12.1	13.0	13.8	2.2770	1.2558
157	Smelt	19.7	13.2	14.3	15.2	2.8728	2.0672
158	Smelt	19.9	13.8	15.0	16.2	2.9322	1.8792

159 rows × 7 columns

Saved successfully!

```
#Get first five rows
df.head()
```

```
#Get last five rows
df.tail()
```

	Species	Weight	Length1	Length2	Length3	Height	Width
154	Smelt	12.2	11.5	12.2	13.4	2.0904	1.3936
155	Smelt	13.4	11.7	12.4	13.5	2.4300	1.2690
156	Smelt	12.2	12.1	13.0	13.8	2.2770	1.2558
157	Smelt	19.7	13.2	14.3	15.2	2.8728	2.0672
158	Smelt	19.9	13.8	15.0	16.2	2.9322	1.8792

```
#Get information of dataframe
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 159 entries, 0 to 158
Data columns (total 7 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   Species  159 non-null    object 
 1   Weight   159 non-null    float64
 2   Length1  159 non-null    float64
 3   Length2  159 non-null    float64
 4   Length3  159 non-null    float64
 5   Height   159 non-null    float64
 6   Width    159 non-null    float64
dtypes: float64(6), object(1)
memory usage: 8.8+ KB
```

Double-click (or enter) to edit

```
#get summary statistics
```

Saved successfully!

	Weight	Length1	Length2	Length3	Height	Width
count	159.000000	159.000000	159.000000	159.000000	159.000000	159.000000

```
#get column names
df.columns
```

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

50%	273.000000	25.200000	27.300000	29.400000	7.786000	4.248500

```
#Define y(dependent label variabel) and x(independent variable)
y=df['Weight']
```

```
y.shape
```

```
(159,)
```

```
y
```

```
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
```

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

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

```
X.shape
```

Saved successfully!

```
X
```

	Length1	Length2	Length3	Height	Width
0	23.2	25.4	30.0	11.5200	4.0200
1	24.0	26.3	31.2	12.4800	4.3056
2	23.9	26.5	31.1	12.3778	4.6961
3	26.3	29.0	33.5	12.7300	4.4555
4	26.5	29.0	34.0	12.4440	5.1340
...	...	...	...	...	...
154	11.5	12.2	13.4	2.0904	1.3936

```
#add Constand to Features(x) for Intercept Estimation
import statsmodels.api as sm
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tools/_testing.py:19: FutureWarni
  import pandas.util.testing as tm
```

```
X=sm.add_constant(X)
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:117: FutureWarnin
  x = pd.concat(x[::order], 1)
```

```
X.head()
```

	const	Length1	Length2	Length3	Height	Width
0	1.0	23.2	25.4	30.0	11.5200	4.0200
1	1.0	24.0	26.3	31.2	12.4800	4.3056
2	1.0	23.9	26.5	31.1	12.3778	4.6961
3	1.0	26.3	29.0	33.5	12.7300	4.4555
4	1.0	26.5	29.0	34.0	12.4440	5.1340

Saved successfully!

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

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=2529)
```

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

```
((111, 6), (48, 6), (111,), (48,))
```

```
#Get model Train
```

```
import statsmodels.api as sm

model=sm.OLS(y_train,X_train).fit()


#Get model Prediction
y_pred=model.predict(X_test)


y_pred
```

```
6      485.768263
54     502.247209
80      94.723820
138    876.571171
91     184.078918
48     219.301305
52     322.325322
103    376.223260
57     372.357305
149   -182.675371
153   -160.604868
108    454.335862
90     159.597558
118    843.485252
131    587.216806
100    299.535214
15     597.729508
46     197.146054
132    639.890467
79      91.200679
64     150.954248
35    -103.083206
133    627.197128
116    795.691769
31     814.687330
146   -204.149651
53     329.987469
28     715.892880
1      359.756344
117    792.324392
9      532.703671
12     552.008323
129    433.484727
```



```
120    810.742342
158    -80.062172
51     284.362879
34     907.080360
23     642.582834
127    959.338482
21     675.287923
113    718.863055
109    623.898492
101    376.483470
10     530.838281
157    -86.235707
dtype: float64
```

```
#Get model evaluation
from sklearn.metrics import mean_squared_error,mean_absolute_error,mean_absolute_percentag
```

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

```
16397.34452441147
```

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

```
103.02952922678588
```

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

```
2.508285347160015
```

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

```
0.8349141424416868
```

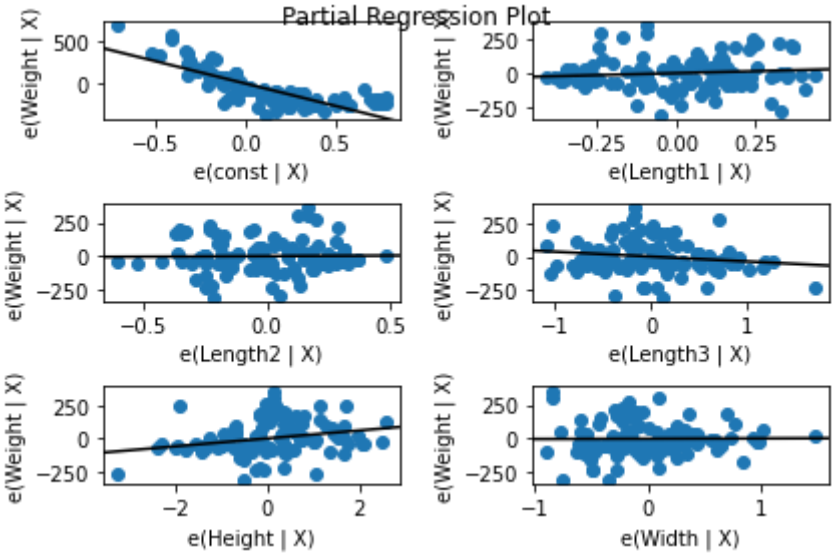
```
#Get Model summary
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 Weight   R-squared:                       0.896
Model:                            OLS   Adj. R-squared:                  0.891
Method:                 Least Squares   F-statistic:                     181.2
Date:                Sun, 24 Apr 2022   Prob (F-statistic):           5.84e-50
Time:                        06:51:07   Log-Likelihood:                -689.20
No. Observations:                 111   AIC:                             1390.
Df Residuals:                     105   BIC:                             1407.
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       -519.2834     34.659    -14.983      0.000    -588.005    -450.562
Length1      58.2270     52.151      1.119      0.266     -45.068     161.743
                         06      0.165      0.869     -94.189     111.256
                         44     -1.686      0.095     -78.671       6.367
Height       29.8643     10.826      2.759      0.007       8.398      51.330
Width         2.2594     26.105      0.087      0.931     -49.502      54.020
==============================================================================
Omnibus:                        5.384   Durbin-Watson:                   2.008
Prob(Omnibus):                  0.068   Jarque-Bera (JB):                4.993
Skew:                           0.391   Prob(JB):                       0.0824
Kurtosis:                       3.684   Cond. No.                         331.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spe
```

Saved successfully!

```
fig=sm.graphics.plot_partregress_grid(model)
```




```
fig=sm.graphics.plot_regress.exog(model,"Width")
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-41-0c939b43c995> in <module>()
----> 1 fig=sm.graphics.plot_regress.exog(model,"Width")

AttributeError: module 'statsmodels.graphics.api' has no attribute 'plot_regress'
```

SEARCH STACK OVERFLOW

```
#Get Futureprediction
df_new=df.sample()
```

```
df_new
```

	Species	Weight	Length1	Length2	Length3	Height	Width
53	Roach	272.0	25.0	27.0	30.6	8.568	4.7736

Saved successfully!

```
gth1','Length2','Length3']]
```

```
X_new=sm.add_constant(X_new,has_constant='add')
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:117: FutureWarnin
  x = pd.concat(x[::order], 1)
```

```
X_new
```

	const	Height	Width	Length1	Length2	Length3
53	1.0	8.568	4.7736	25.0	27.0	30.6

```
X_new.shape
```

```
(1, 6)
```

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

```
y_pred_new
```

Saved successfully!

✓ 0s completed at 12:29 PM

Saved successfully!