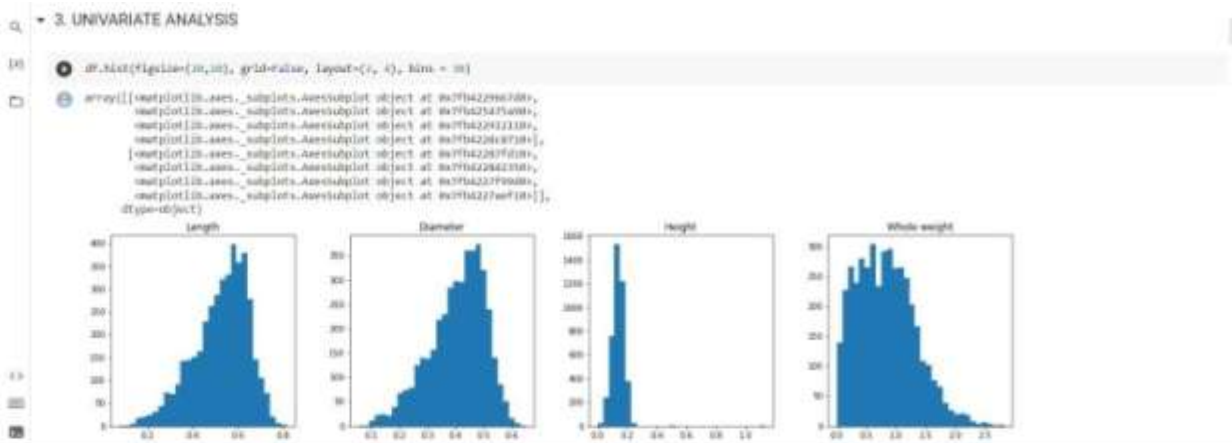
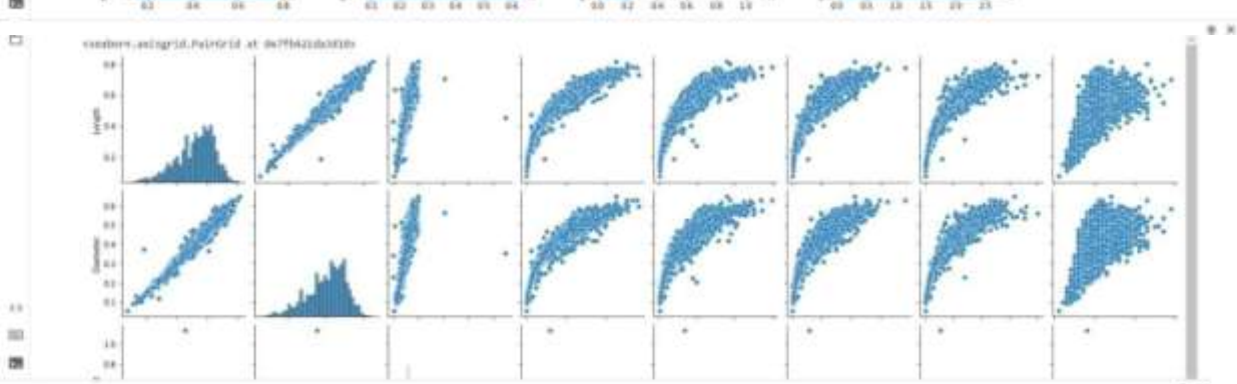
Assignment-4

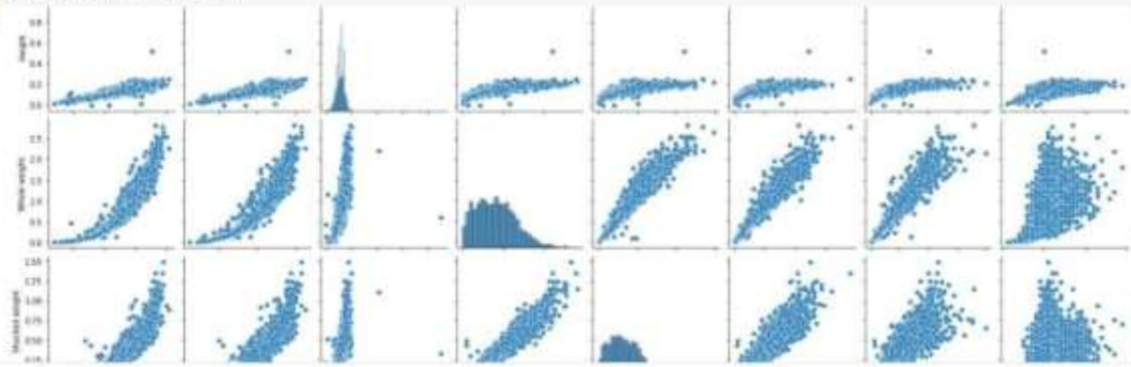
Date	**10.11.2022**
Team Id	**PNT2022TMID24138**
Project Name	**Estimated The Crop Yelid using Data Analytics**
Team Leader	**D.Sandhiya**
Team Member	**M.Ruthra P.Priyadharshini D.Tejasvani**





3. BIVARIATE ANALYSIS & MULTIVARIATE ANALYSIS

```
numerical_features = df.select_dtypes(include = [np.number]).columns
sns.pairplot(df[numerical_features])
```



4. Descriptive statistics

```
df.describe()
```

	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
count	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000
mean	0.523992	0.407881	0.139516	0.828742	0.359367	0.180594	0.238831	9.933684
std	0.120093	0.099240	0.041827	0.490389	0.221963	0.109614	0.139203	3.224169
min	0.075000	0.055000	0.000000	0.002000	0.001000	0.000500	0.001500	1.000000
25%	0.450000	0.350000	0.115000	0.441500	0.186000	0.093500	0.130000	8.000000
50%	0.545000	0.425000	0.140000	0.799500	0.336000	0.171000	0.234000	9.000000
75%	0.615000	0.480000	0.165000	1.153000	0.502000	0.253000	0.329000	11.000000
max	0.815000	0.650000	1.130000	2.825500	1.488000	0.760000	1.005000	29.000000

5. Check for Missing Values

```
df.isnull().sum()
```

```
Sex               0
Length            0
Diameter          0
Height            0
Whole weight      0
Shucked weight    0
Viscera weight    0
Shell weight      0
Rings             0
dtype: int64
```

6. OUTLIER HANDLING

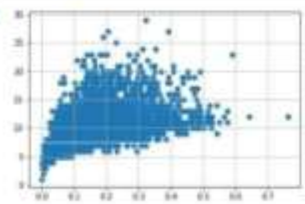
```
df = pd.get_dummies(df)
dummy_data = df.copy()
```

```
var = 'Viscera weight'
plt.scatter(x = df[var], y = df['Rings'],)
plt.grid(True)
```

6. OUTLIER HANDLING

```
df = pd.get_dummies(df)
dummy_data = df.copy()
```

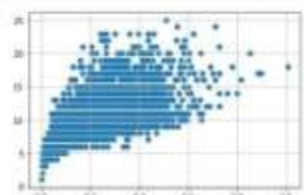
```
var = 'Viscera weight'
plt.scatter(x = df[var], y = df['Rings'],)
plt.grid(True)
```



```
# outliers removal
df.drop(df[(df['Viscera weight']> 0.5) & (df['Rings'] < 20)].index, inplace=True)
```

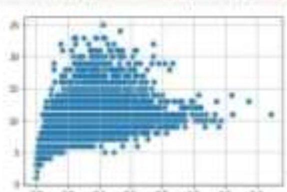
```
# outliers removal
df.drop(df[(df['Viscera weight']> 0.5) & (df['Rings'] < 20)].index, inplace=True)
df.drop(df[(df['Viscera weight']<0.5) & (df['Rings'] > 25)].index, inplace=True)
```

```
var = 'Shell weight'
plt.scatter(x = df[var], y = df['Rings'],)
plt.grid(True)
#Outliers removal
df.drop(df[(df['Shell weight']> 0.6) & (df['Rings'] < 25)].index, inplace=True)
df.drop(df[(df['Shell weight']<0.8) & (df['Rings'] > 25)].index, inplace=True)
```



```
var = 'Shucked weight'
plt.scatter(x = df[var], y = df['Rings'],)
plt.grid(True)

#Outlier removal
df.drop(df[(df['Shucked weight']>= 1) & (df['Rings'] < 20)].index, inplace=True)
df.drop(df[(df['Shucked weight']<1) & (df['Rings'] > 20)].index, inplace=True)
```

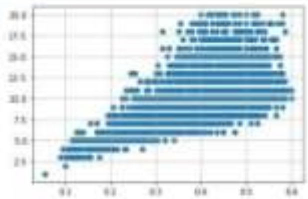


```
var = 'Whole weight'
plt.scatter(x = df[var], y = df['Rings'])
plt.grid(True)

df.drop(df[(df['Whole weight'] >= 2.5) &
          (df['Rings'] < 25)].index, inplace = True)
```

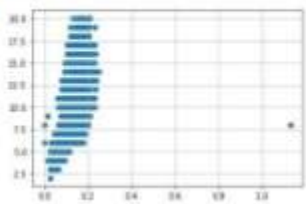
```python
var = 'Diameter'
plt.scatter(x = df[var], y = df['Rings'])
plt.grid(True)

df.drop(df[(df['Diameter'] <0.1) &
          (df['Rings'] < 5)].index, inplace = True)
df.drop(df[(df['Diameter']<0.4) & (
df['Rings'] > 25)].index, inplace = True)
df.drop(df[(df['Diameter']>=0.4) & (
df['Rings'] < 25)].index, inplace = True)
```



```python
[ ] var = 'Height'
    plt.scatter(x = df[var], y = df['Rings'])
    plt.grid(True)
```

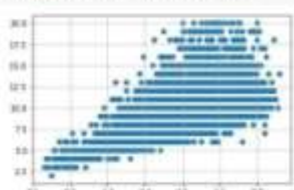
```python
var = 'Height'
plt.scatter(x = df[var], y = df['Rings'])
plt.grid(True)
df.drop(df[(df['Height'] > 0.4) &
          (df['Rings'] < 15)].index, inplace = True)
df.drop(df[(df['Height']<0.4) & (
df['Rings'] > 25)].index, inplace = True)
```



```python
[ ] var = 'Length'
    plt.scatter(x = df[var], y = df['Rings'])
    plt.grid(True)
```

```python
var = 'Length'
plt.scatter(x = df[var], y = df['Rings'])
plt.grid(True)

df.drop(df[(df['Length'] <0.1) &
          (df['Rings'] < 5)].index, inplace = True)
df.drop(df[(df['Length']<0.5) & (
df['Rings'] > 25)].index, inplace = True)
df.drop(df[(df['Length']>=0.5) & (
df['Rings'] < 25)].index, inplace = True)
```



7. Categorical columns

```
[ ] numerical_features = df.select_dtypes(include = [np.number]).columns
    categorical_features = df.select_dtypes(include = [np.object]).columns
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: DeprecationWarning: `np.object` is a deprecated alias for the builtin `object`. To silence this warning, use `object` by
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
```

```
numerical_features
```

```
Index(['Length', 'Diameter', 'Height', 'Whole weight', 'Shucked weight',
       'Viscera weight', 'Shell weight', 'Rings', 'Sex_F', 'Sex_I', 'Sex_M'],
      dtype='object')
```

```
[ ] categorical_features
```

```
Index([], dtype='object')
```

ENCODING

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
print(df.Length.value_counts())
```

```
0.575    93
0.580    92
0.550    93
0.625    91
0.600    85
         ..
0.135     1
0.130     1
0.755     1
0.150     1
0.760     1
Name: Length, Length: 126, dtype: int64
```

8. Split the dependent and independent variables

```
[ ] x=df.iloc[:,:5]
    x
```

	Length	Diameter	Height	Whole weight	Shucked weight

8. Split the dependent and independent variables

```
x=df.iloc[:,:5]
x
```


	Length	Diameter	Height	Whole weight	Shucked weight
0	0.455	0.365	0.095	0.5140	0.2245
1	0.350	0.265	0.090	0.2255	0.0995
2	0.530	0.420	0.135	0.6770	0.2565
3	0.440	0.365	0.125	0.5160	0.2155
4	0.330	0.255	0.080	0.2050	0.0895
...	...	...	...	...	...
4172	0.565	0.450	0.165	0.8870	0.3700
4173	0.590	0.440	0.135	0.9660	0.4390
4174	0.600	0.475	0.205	1.1760	0.5255
4175	0.625	0.485	0.150	1.0945	0.5310
4176	0.710	0.555	0.195	1.9485	0.9455

4049 rows × 5 columns

```
y=df.iloc[:,4:]
y
```

	Viscera weight	Shell weight	Rings	Sex_F	Sex_I	Sex_M
0	0.1010	0.1500	15	0	0	1
1	0.0485	0.0700	7	0	0	1
2	0.1415	0.2100	9	1	0	0
3	0.1140	0.1550	10	0	0	1
4	0.0395	0.0550	7	0	1	0
...	...	...	...	...	...	...
4172	0.2390	0.2490	11	1	0	0
4173	0.2145	0.2605	10	0	0	1
4174	0.2875	0.3080	9	0	0	1
4175	0.2610	0.2960	10	1	0	0
4176	0.3765	0.4950	12	0	0	1

4040 rows × 6 columns

9. Feature Scaling

```
from sklearn.preprocessing import StandardScaler
ss=StandardScaler()
x=ss.fit_transform(x)
```

10. Train , Test , Split

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

11. Model building

```
from sklearn.linear_model import LinearRegression
mlr=LinearRegression()
mlr.fit(x_train,y_train)
```

```
LinearRegression()
```

12 & 13. Train and Test the model

```
x_test[0:5]
```

```
array([[-0.[illegible], 0.[illegible], 0.[illegible], -0.[illegible], 0.[illegible]],
       [ 0.[illegible], 0.[illegible], 0.[illegible], 0.[illegible], 0.[illegible] ],
       [-0.[illegible], -0.[illegible], -0.[illegible], -0.[illegible], -0.[illegible]],
       [-0.[illegible], -0.[illegible], -0.[illegible], -0.[illegible], -0.[illegible]],
       [-0.[illegible], -0.[illegible], -0.[illegible], -1.[illegible], -1.[illegible]]])
```

```
y_test[0:5]
```

	Viscera weight	Shell weight	Rings	Sex_F	Sex_I	Sex_M
3484	0.1505	0.1845	8	0	1	0
2459	0.2525	0.3500	20	1	0	0
3252	0.1420	0.1700	12	0	0	1
2648	0.1205	0.1300	7	0	1	0
3638	0.0810	0.1250	8	0	1	0

14. Measure the performance using metrics

```
from sklearn.metrics import r2_score
r2_score(mlr.predict(x_test),y_test)
```

[illegible]