Assignment : Padmanaban T

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
```

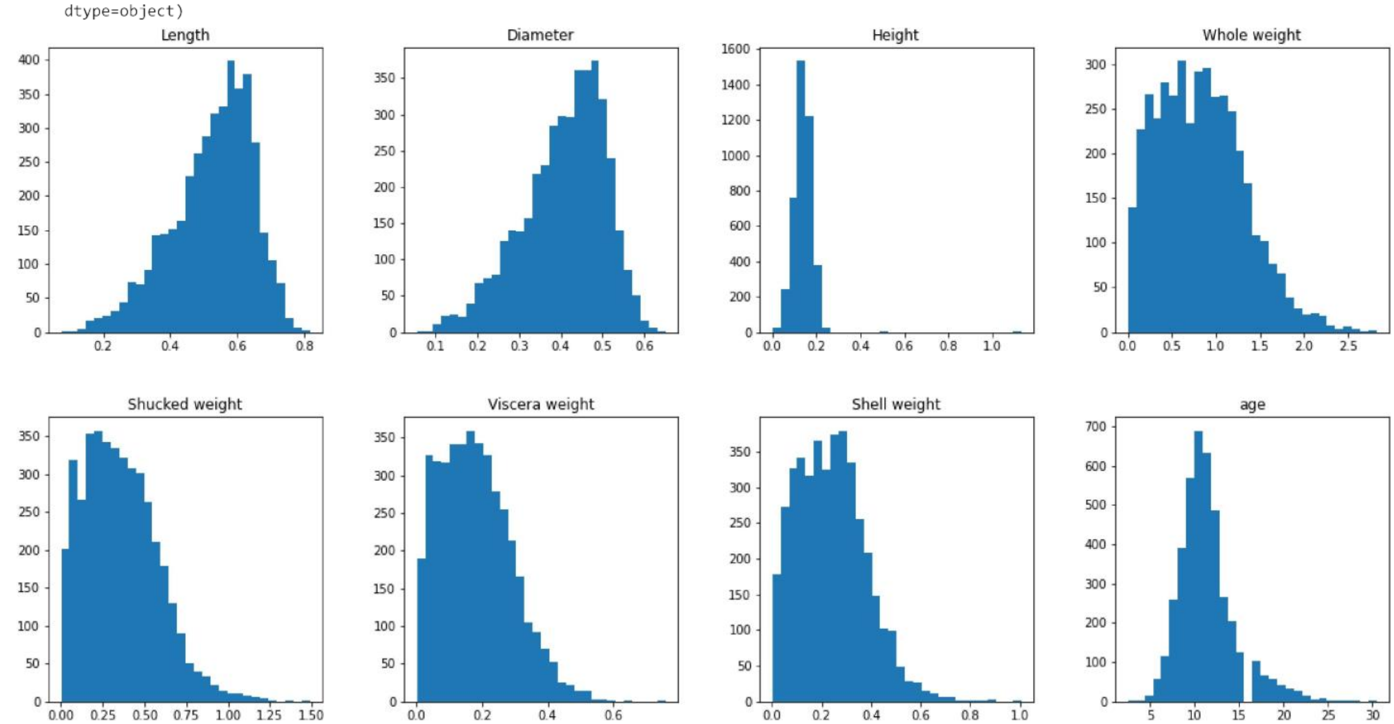
```
df=pd.read_csv("/content/drive/MyDrive/Colab Notebooks/abalone.csv")
```

```
df['age'] = df['Rings']+1.5
df = df.drop('Rings', axis = 1)
```

Univariate Analysis

```
df.hist(figsize=(20,10), grid=False, layout=(2, 4), bins = 30)
```

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7f3d1b0fb690>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3d1ade4d90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3d1adaa390>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3d1ad60990>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7f3d1ad16f90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3d1acda5d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3d1ac8fc50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f3d1ac531d0>]],
      dtype=object)
```



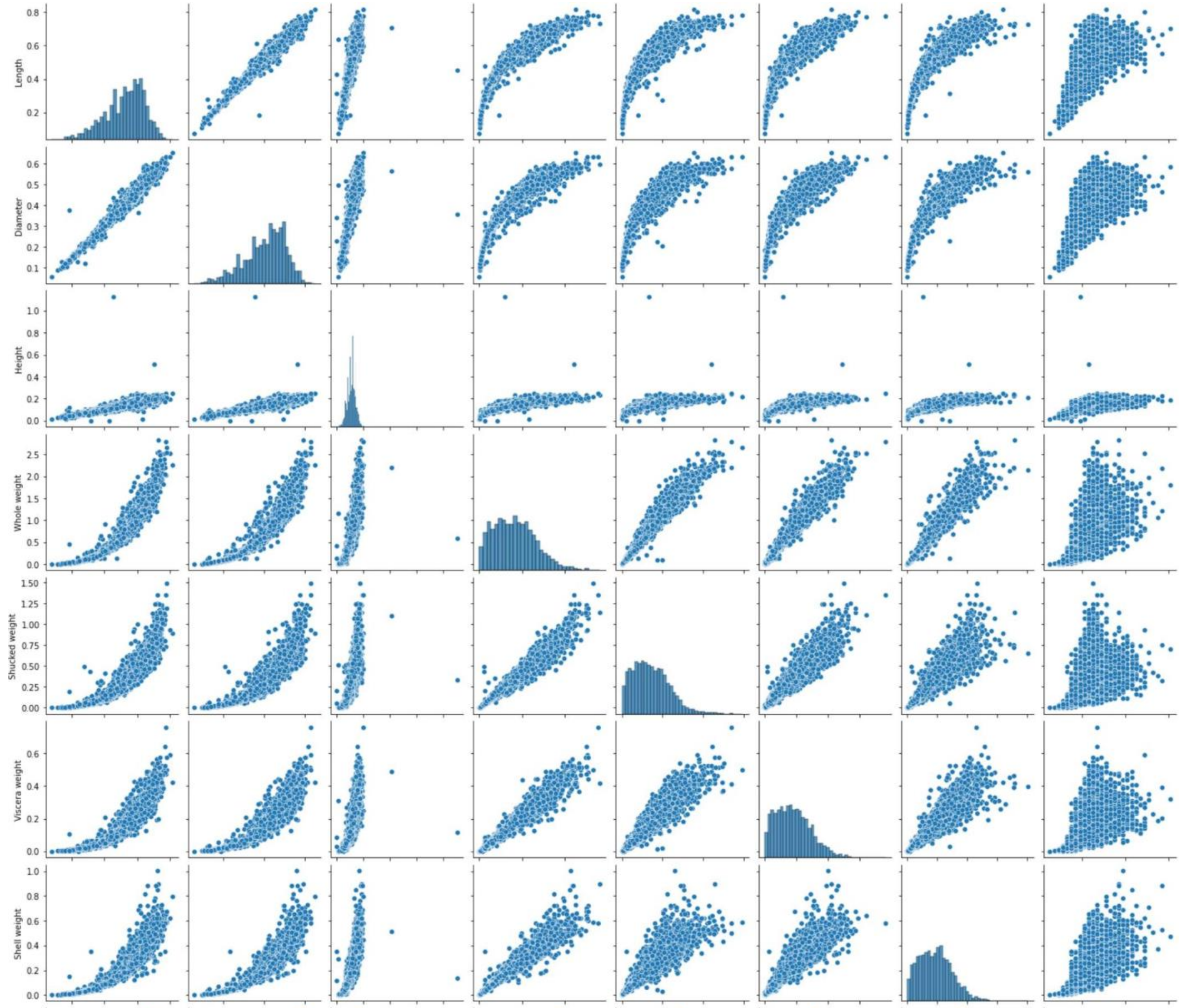

```
df.groupby('Sex')[['Length', 'Diameter', 'Height', 'Whole weight', 'Shucked weight',
       'Viscera weight', 'Shell weight', 'age']].mean().sort_values('age')
```

Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	age
I	0.427746	0.326494	0.107996	0.431363	0.191035	0.092010	0.128182	9.390462
M	0.561391	0.439287	0.151381	0.991459	0.432946	0.215545	0.281969	12.205497
F	0.579093	0.454732	0.158011	1.046532	0.446188	0.230689	0.302010	12.629304

Bivariate Analysis

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

<seaborn.axisgrid.PairGrid at 0x7f3d1a345650>



Descriptive statistics

```
df.describe()
```

	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	age
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	11.433684
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	2.500000
25%	0.450000	0.350000	0.115000	0.441500	0.186000	0.093500	0.130000	9.500000
50%	0.545000	0.425000	0.140000	0.799500	0.336000	0.171000	0.234000	10.500000
75%	0.615000	0.480000	0.165000	1.153000	0.502000	0.253000	0.329000	12.500000
max	0.815000	0.650000	1.130000	2.825500	1.488000	0.760000	1.005000	30.500000

Check for missing values

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

Outlier handling

```
df = pd.get_dummies(df)
dummy_data = df.copy()
var = 'Viscera weight'
plt.scatter(x = df[var], y = df['age'],)
plt.grid(True)
```

```
# outliers removal
df.drop(df[(df['Viscera weight']> 0.5) & (df['age'] < 20)].index, inplace=True)
df.drop(df[(df['Viscera weight']<0.5) & (df['age'] > 25)].index, inplace=True)
var = 'Shell weight'
plt.scatter(x = df[var], y = df['age'],)
plt.grid(True)
#Outliers removal
df.drop(df[(df['Shell weight']> 0.6) & (df['age'] < 25)].index, inplace=True)
df.drop(df[(df['Shell weight']<0.8) & (df['age'] > 25)].index, inplace=True)
```

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

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

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

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

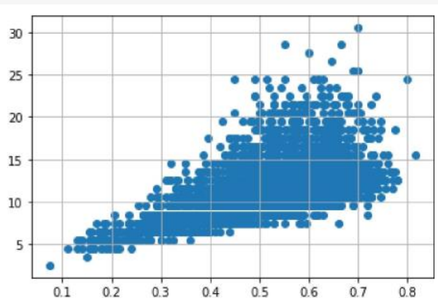
```
var = 'Diameter'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)

df.drop(df[(df['Diameter'] <0.1) &
          (df['age'] < 5)].index, inplace = True)
df.drop(df[(df['Diameter']<0.6) & (
df['age'] > 25)].index, inplace = True)
df.drop(df[(df['Diameter']>=0.6) & (
df['age'] < 25)].index, inplace = True)
```

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

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

df.drop(df[(df['Length'] <0.1) &
          (df['age'] < 5)].index, inplace = True)
df.drop(df[(df['Length']<0.8) & (
df['age'] > 25)].index, inplace = True)
df.drop(df[(df['Length']>=0.8) & (
df['age'] < 25)].index, inplace = True)
```



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 siler
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', 'age'],
      dtype='object')
```

```
categorical_features
```

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

ENCODING

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

```
M    1525
I    1341
F    1301
Name: Sex, dtype: int64
```

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

	Sex	Length	Diameter	Height	Whole weight
0	M	0.455	0.365	0.095	0.5140
1	M	0.350	0.265	0.090	0.2255
2	F	0.530	0.420	0.135	0.6770
3	M	0.440	0.365	0.125	0.5160
4	I	0.330	0.255	0.080	0.2050
...	...	...	...	...	...
4172	F	0.565	0.450	0.165	0.8870
4173	M	0.590	0.440	0.135	0.9660
4174	M	0.600	0.475	0.205	1.1760
4175	F	0.625	0.485	0.150	1.0945
4176	M	0.710	0.555	0.195	1.9485

4167 rows × 5 columns

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

	Shucked weight	Viscera weight	Shell weight	age
0	0.2245	0.1010	0.1500	16.5
1	0.0995	0.0485	0.0700	8.5
2	0.2565	0.1415	0.2100	10.5
3	0.2155	0.1140	0.1550	11.5
4	0.0895	0.0395	0.0550	8.5
...	...	...	...	...
4172	0.3700	0.2390	0.2490	12.5
4173	0.4390	0.2145	0.2605	11.5
4174	0.5255	0.2875	0.3080	10.5
4175	0.5310	0.2610	0.2960	11.5
4176	0.9455	0.3765	0.4950	13.5

4167 rows × 4 columns

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)
```

Model Building

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

Train and Test model

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

	Sex	Length	Diameter	Height	Whole weight
661	I	0.535	0.450	0.170	0.781
370	F	0.650	0.545	0.165	1.566
2272	M	0.635	0.510	0.210	1.598
1003	M	0.595	0.455	0.150	1.044
1145	M	0.580	0.455	0.195	1.859

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

	Shucked weight	Viscera weight	Shell weight	age
661	0.3055	0.1555	0.295	12.5
370	0.6645	0.3455	0.415	17.5
2272	0.6535	0.2835	0.580	16.5
1003	0.5180	0.2205	0.270	10.5
1145	0.9450	0.4260	0.441	10.5

Feature Scaling

```
from sklearn.preprocessing import StandardScaler
ss=StandardScaler()
x_train=ss.fit_transform(x_train)
mlrpred=mlr.predict(x_test[0:9])
mlrpred
```

Performance measure

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

0.5597133867640833