```
import pandas as pd
import numpy as np
import seaborn as sns
import sklearn
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.feature_selection import SelectKBest
from sklearn.metrics import mean_absolute_error, mean_squared_error
from sklearn.metrics import r2_score
```

```
from google.colab import drive
drive.mount('/content/drive', force_remount=True)
```

```
Mounted at /content/drive
```

```
import os
path="/content/drive/MyDrive/"
os.chdir(path)
```

```
df = pd.read_csv('abalone1.csv')
```

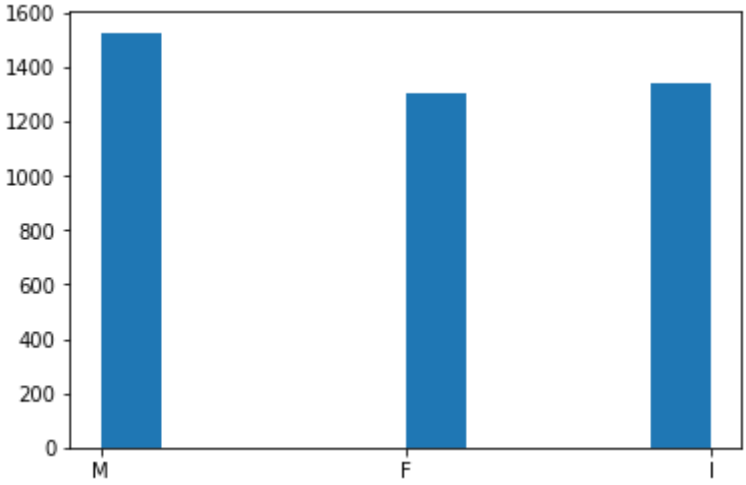
```
df.describe()
```

	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	
count	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4
mean	0.523992	0.407881	0.139516	0.828742	0.359367	0.180594	
std	0.120093	0.099240	0.041827	0.490389	0.221963	0.109614	
min	0.075000	0.055000	0.000000	0.002000	0.001000	0.000500	
25%	0.450000	0.350000	0.115000	0.441500	0.186000	0.093500	
50%	0.545000	0.425000	0.140000	0.799500	0.336000	0.171000	
75%	0.615000	0.480000	0.165000	1.153000	0.502000	0.253000	

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

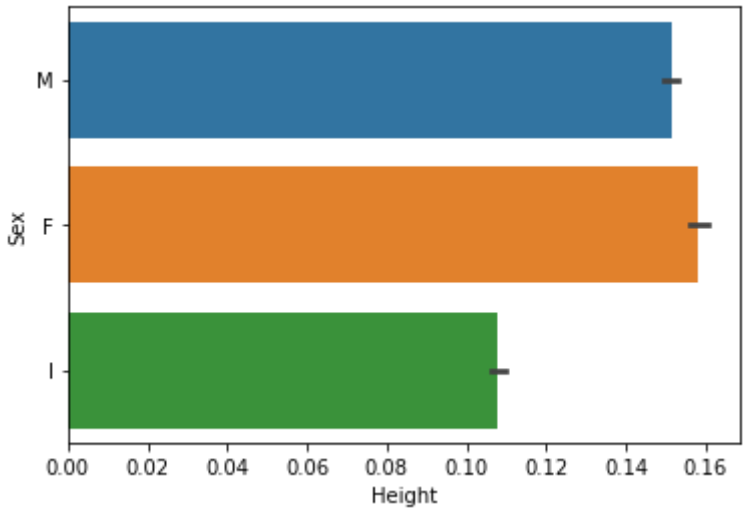
```
plt.hist(df['Sex'])
```

```
(array([1528.,    0.,    0.,    0.,    0., 1307.,    0.,    0.,    0.,
        1342.]),
 array([0. , 0.2, 0.4, 0.6, 0.8, 1. , 1.2, 1.4, 1.6, 1.8, 2. ]),
 <a list of 10 Patch objects>)
```



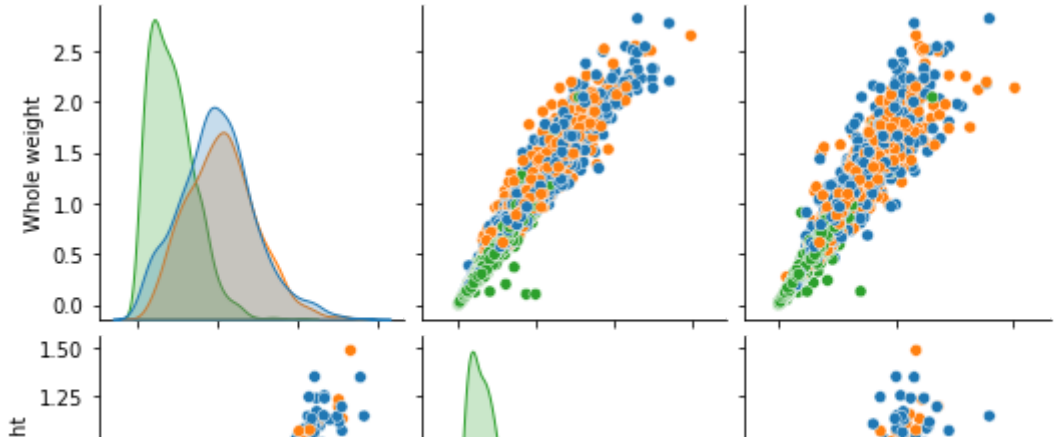

```
sns.barplot(x='Height',y='Sex',data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6e251ad910>
```




```
sns.pairplot(data=df[['Sex','Whole weight','Shucked weight','Shell weight']], hue='Sex')
```

```
<seaborn.axisgrid.PairGrid at 0x7f6e24f37b90>
```

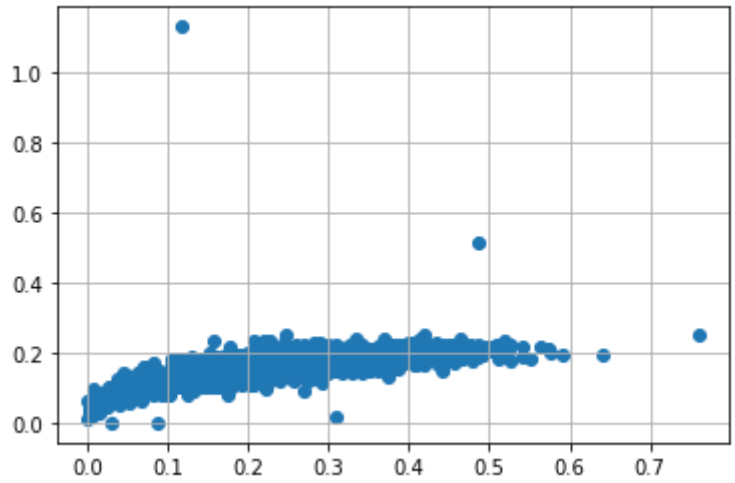



```
df.isnull().sum()
```

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

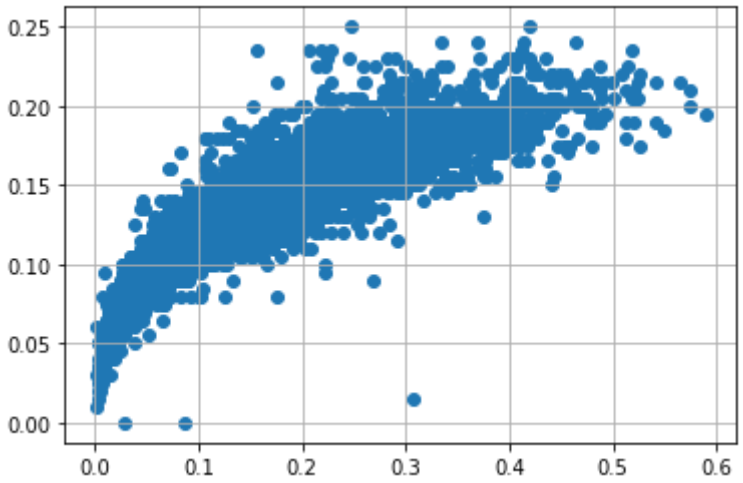

```
df = pd.get_dummies(df)
dummy_df = df
```

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




```
df.drop(df[(df['Viscera weight'] > 0.6) &
          (df['Height'] < 0.4)].index, inplace = True)
df.drop(df[(df['Viscera weight']<0.6) & (
df['Height'] > 0.4)].index, inplace = True)
```

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




```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4173 entries, 0 to 4176
Data columns (total 11 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Length          4173 non-null   float64
 1   Diameter        4173 non-null   float64
 2   Height          4173 non-null   float64
 3   Whole weight    4173 non-null   float64
 4   Shucked weight  4173 non-null   float64
 5   Viscera weight  4173 non-null   float64
 6   Shell weight    4173 non-null   float64
 7   age             4173 non-null   float64
 8   Sex_F           4173 non-null   uint8  
 9   Sex_I           4173 non-null   uint8  
 10  Sex_M           4173 non-null   uint8  
dtypes: float64(8), uint8(3)
memory usage: 305.6 KB
```

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

```
categorical_features
```

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

```
X = df.iloc[:, :-1].values
X
```

```
array([[ 0.455,  0.365,  0.095, ..., 16.5  ,  0.   ,  0.   ],
       [ 0.35 ,  0.265,  0.09 , ...,  8.5  ,  0.   ,  0.   ],
       [ 0.53 ,  0.42 ,  0.135, ..., 10.5  ,  1.   ,  0.   ],
       ...,
       [ 0.6  ,  0.475,  0.205, ..., 10.5  ,  0.   ,  0.   ],
```

```
            [ 0.625,  0.485,  0.15 , ..., 11.5  ,  1.   ,  0.   ],
            [ 0.71 ,  0.555,  0.195, ..., 13.5  ,  0.   ,  0.   ]])
```

```
Y = df.iloc[:, -1].values
Y
```

```
array([1, 1, 0, ..., 1, 0, 1], dtype=uint8)
```

```
scaler = StandardScaler()
scaler.fit_transform(X)
```

```
array([[-0.57383628, -0.43128157, -1.14835115, ...,  1.57118153,
        -0.67492846, -0.68850377],
       [-1.44901676, -1.44007503, -1.27840432, ..., -0.90953755,
        -0.67492846, -0.68850377],
       [ 0.05129263,  0.12355484, -0.10792575, ..., -0.28935778,
         1.48163851, -0.68850377],
       ...,
       [ 0.63474628,  0.67839124,  1.7128187 , ..., -0.28935778,
        -0.67492846, -0.68850377],
       [ 0.84312258,  0.77927059,  0.28223378, ...,  0.02073211,
         1.48163851, -0.68850377],
       [ 1.55160201,  1.48542601,  1.45271235, ...,  0.64091188,
        -0.67492846, -0.68850377]])
```

```
X1 = df.drop('age', axis = 1)
Y1 = df['age']
```

```
selectkBest = SelectKBest()
X_new = selectkBest.fit_transform(X1, Y1)

X_train, X_test, y_train, y_test = train_test_split(X_new, Y1, test_size = 0.25)
```

```
lm = LinearRegression()
lm.fit(X_train, y_train)
```

```
LinearRegression()
```

```
y_train_pred = lm.predict(X_train)
y_test_pred = lm.predict(X_test)
```

```
s = mean_squared_error(y_train, y_train_pred)
print('Mean Squared error of training set :%2f'%s)

p = mean_squared_error(y_test, y_test_pred)
print('Mean Squared error of testing set :%2f'%p)
```

```
Mean Squared error of training set :4.745019
Mean Squared error of testing set :4.758223
```

```
s = r2_score(y_train, y_train_pred)
print('R2 Score of training set:%.2f'%s)
```

```
p = r2_score(y_test, y_test_pred)
print('R2 Score of testing set:%.2f'%p)
```

```
     R2 Score of training set:0.54
     R2 Score of testing set:0.55
```