```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
df=pd.read_csv('/content/CrabAgePrediction.csv')
df
```

	Sex	Length	Diameter	Height	Weight	Shucked Weight	Viscera Weight	Shell Weight	Age
0	F	1.4375	1.1750	0.4125	24.635715	12.332033	5.584852	6.747181	9
1	M	0.8875	0.6500	0.2125	5.400580	2.296310	1.374951	1.559222	6
2	I	1.0375	0.7750	0.2500	7.952035	3.231843	1.601747	2.764076	6
3	F	1.1750	0.8875	0.2500	13.480187	4.748541	2.282135	5.244657	10
4	I	0.8875	0.6625	0.2125	6.903103	3.458639	1.488349	1.700970	6
...	...	...	...	...	...	...	...	...	...
3888	F	1.4625	1.1375	0.3250	24.819987	11.651644	5.854172	6.378637	8
3889	F	1.5500	1.2125	0.4375	34.458817	15.450477	7.172423	9.780577	10
3890	I	0.6250	0.4625	0.1625	2.012815	0.765436	0.524466	0.637864	5
3891	I	1.0625	0.7750	0.2625	10.347568	4.507570	2.338834	2.976698	6
3892	I	0.7875	0.6125	0.2125	4.068153	1.502523	1.346601	1.417475	8

Next steps: Generate code with `df` View recommended plots

```
df.head()
```

```
df.tail()
```

```
df.isna().sum()
```

```
Sex               0
Length            0
Diameter          0
Height            0
Weight            0
Shucked Weight    0
Viscera Weight    0
Shell Weight      0
Age               0
dtype: int64
```

```
df.dtypes
```

```
Sex                object
Length            float64
Diameter          float64
Height            float64
Weight            float64
Shucked Weight    float64
Viscera Weight    float64
Shell Weight      float64
Age                 int64
dtype: object
```

```
x=df.iloc[:,:-1]
x
```

```
y=df.iloc[:,-1]
y
```

```
0        9
1        6
2        6
3       10
4        6
       ..
3888     8
3889    10
3890     5
3891     6
```

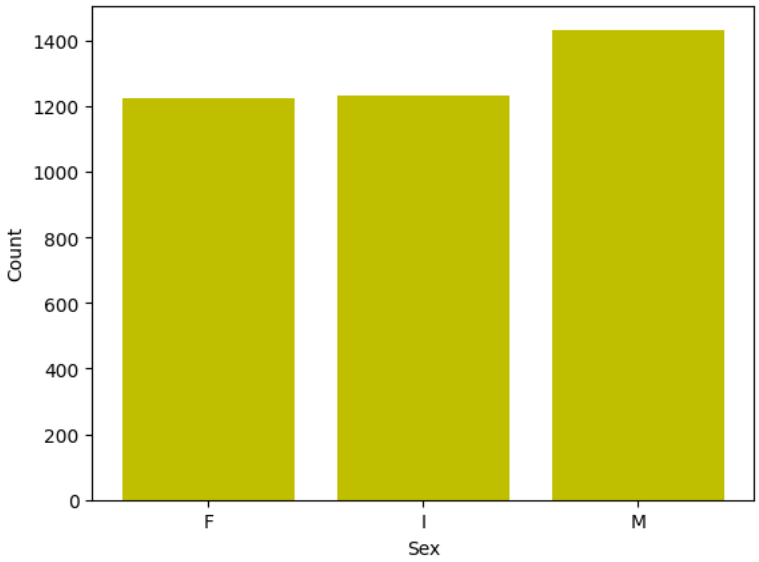
```
3892    8
Name: Age, Length: 3893, dtype: int64
```

```
df.groupby('Sex')['Sex'].count()
```

```
Sex
F    1225
I    1233
M    1435
Name: Sex, dtype: int64
```

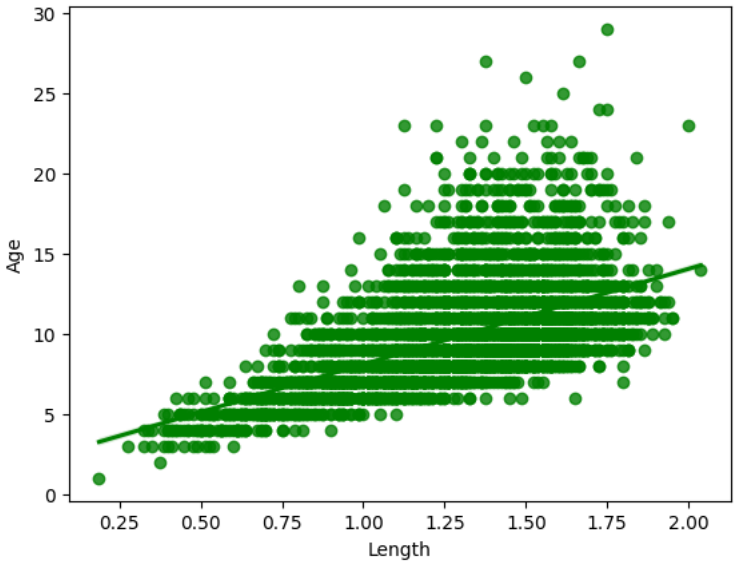
```
sex=['F','I','M']
Count=[1225,1233,1433]
plt.bar(sex,Count,color='y')
plt.xlabel('Sex')
plt.ylabel('Count')
```

```
Text(0, 0.5, 'Count')
```



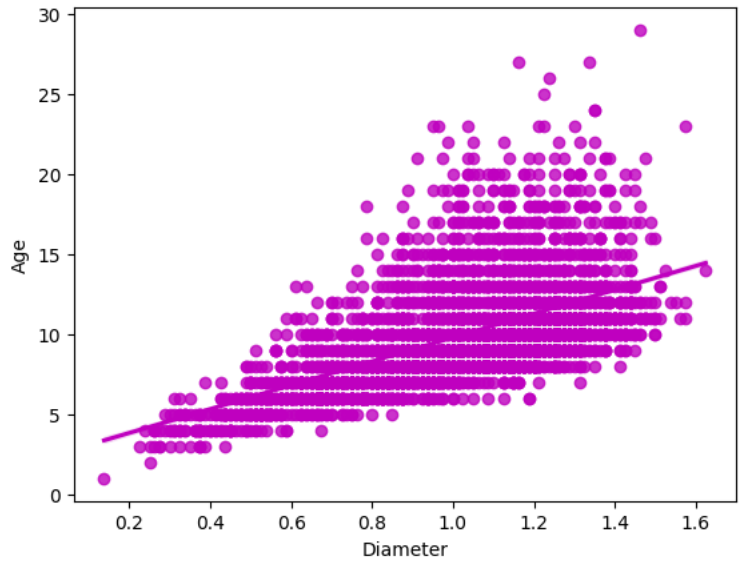

```
sns.regplot(x=df['Length'],y=y,color='g')
```

```
<Axes: xlabel='Length', ylabel='Age'>
```



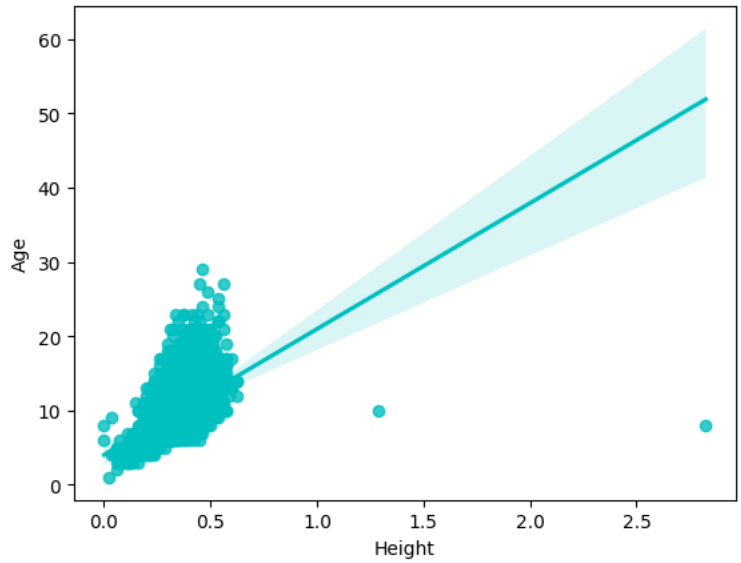

```
sns.regplot(x=df['Diameter'],y=y,color='m')
```

```
<Axes: xlabel='Diameter', ylabel='Age'>
```



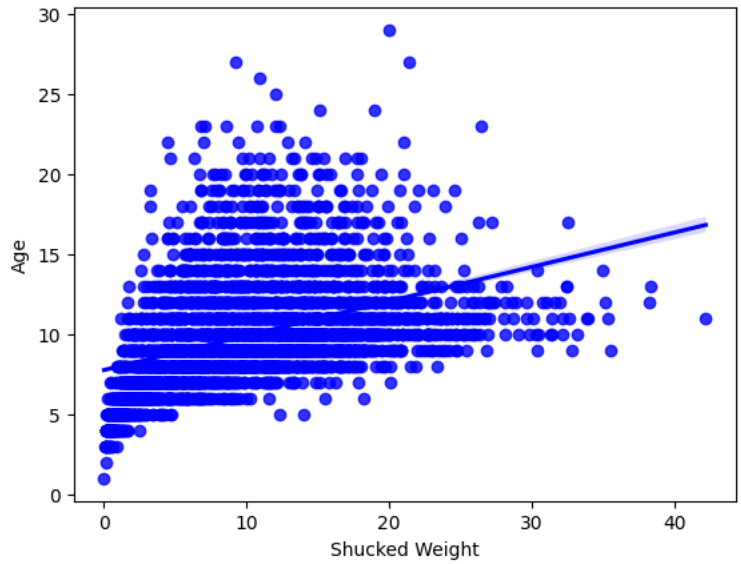

```
sns.regplot(x=df['Height'],y=y,color='c')
```

```
<Axes: xlabel='Height', ylabel='Age'>
```



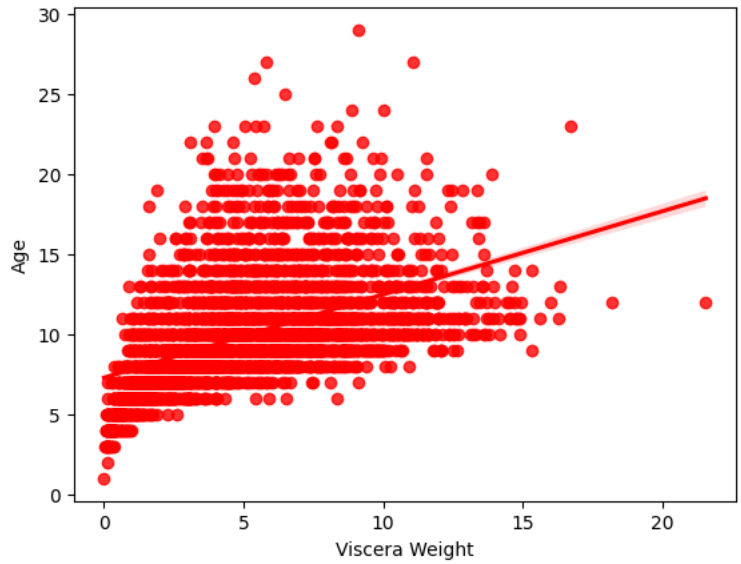

```
sns.regplot(x=df['Shucked Weight'],y=y,color='b')
```

```
<Axes: xlabel='Shucked Weight', ylabel='Age'>
```



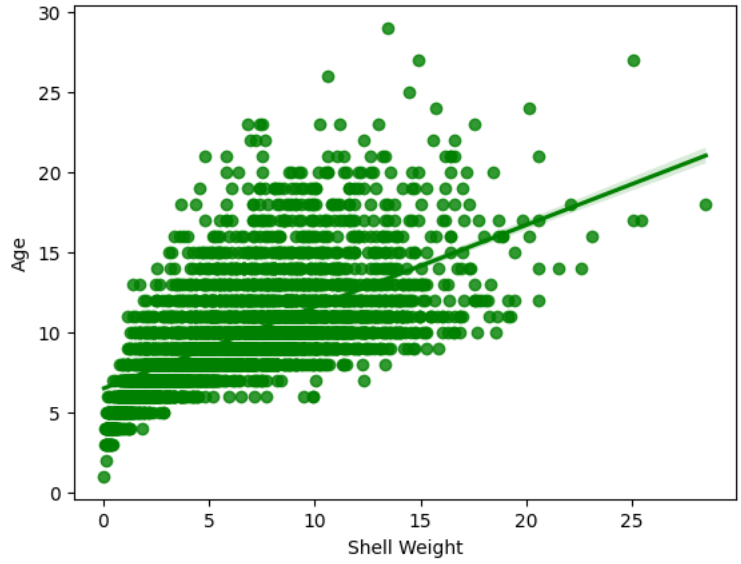

```
sns.regplot(x=df['Viscera Weight'],y=y,color='r')
```

```
<Axes: xlabel='Viscera Weight', ylabel='Age'>
```




```
sns.regplot(x=df['Shell Weight'],y=y,color='g')
```

```
<Axes: xlabel='Shell Weight', ylabel='Age'>
```




```
from sklearn.compose import make_column_transformer
from sklearn.preprocessing import OneHotEncoder
col_trans=make_column_transformer((OneHotEncoder(handle_unknown='ignore'),['Sex']),remainder='passthrough')
x=col_trans.fit_transform(x)
x
```

```
array([[ 1.        ,  0.        ,  0.        , ..., 12.3320325 ,
         5.5848515 ,  6.747181  ],
       [ 0.        ,  0.        ,  1.        , ...,  2.2963095 ,
         1.37495075,  1.5592225 ],
       [ 0.        ,  1.        ,  0.        , ...,  3.231843  ,
         1.60174675,  2.76407625],
       ...,
       [ 0.        ,  1.        ,  0.        , ...,  0.7654365 ,
         0.52446575,  0.63786375],
       [ 0.        ,  1.        ,  0.        , ...,  4.5075705 ,
         2.33883375,  2.9766975 ],
       [ 0.        ,  1.        ,  0.        , ...,  1.5025235 ,
         1.34660125,  1.417475  ]])
```

```
x.shape
```

```
(3893, 10)
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=1)
x_train
```

```
array([[ 1.        ,  0.        ,  0.        , ..., 16.159215  ,
         8.63242275,  9.355335  ],
       [ 0.        ,  1.        ,  0.        , ..., 12.0485375 ,
         5.62737575,  8.50485   ],
       [ 1.        ,  0.        ,  0.        , ..., 16.48523425,
         7.824462  ,  8.9300925 ],
       ...,
       [ 1.        ,  0.        ,  0.        , ..., 11.1980525 ,
         5.5281525 ,  6.3219385 ],
       [ 0.        ,  1.        ,  0.        , ...,  7.6260155 ,
         3.1467945 ,  3.69960975],
       [ 0.        ,  1.        ,  0.        , ...,  2.324659  ,
```

```
from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
y_pred
```

```
array([10.45698802, 10.5114703 ,  8.56863421, ...,  6.80997518,
        7.80010785,  9.77112235])
```

```
df1=pd.DataFrame({'Actual_value':y_test,'Predicted_value':y_pred,'Difference':y_test-y_pred})
df1
```

	Actual_value	Predicted_value	Difference
1413	8	10.456988	-2.456988
2785	9	10.511470	-1.511470
2905	12	8.568634	3.431366
1396	6	5.010462	0.989538
724	8	6.744776	1.255224
...	...	...	...
54	7	8.748558	-1.748558
3561	9	12.479194	-3.479194
138	6	6.809975	-0.809975
3315	9	7.800108	1.199892
1508	9	9.771122	-0.771122

1168 rows × 3 columns

Next steps: Generate code with `df1` View recommended plots

```
print("Slope is",model.coef_)
```

```
Slope is [ 0.25709472 -0.60400843  0.34691371 -0.04165773  4.38141283  3.28961493
  0.2940312  -0.67748047 -0.3403321   0.34492761]
```

```
print("Constant is",model.intercept_)
```

```
from sklearn.metrics import mean_absolute_error,mean_absolute_percentage_error,mean_squared_error,r2_score
print("MAE is", mean_absolute_error(y_test,y_pred))
print("MAPE is", mean_absolute_percentage_error(y_test,y_pred))
print("MSE is", mean_squared_error(y_test,y_pred))
print("r2_score is", r2_score(y_test,y_pred))
RMSE=np.sqrt(mean_squared_error(y_test,y_pred))
print("RMSE is", RMSE)
```