Abalone Age Prediction

In [1]:
```
from google.colab import drive
drive.mount('/content/drive')
```

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

In [5]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [10]:
```
df = pd.read_csv('/content/abalone.csv')
df.shape
```

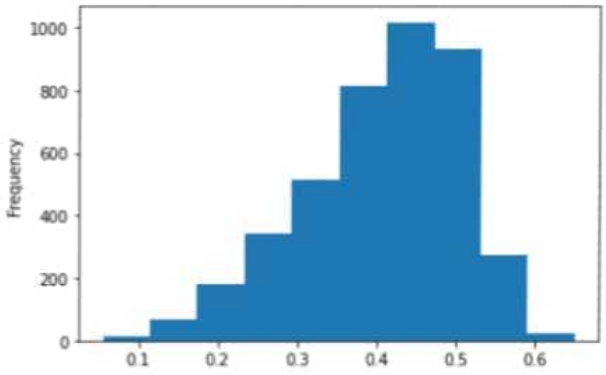
Out[10]:
```
(4177, 9)
```

In [11]:
```
#Load the dataset
df.head()
```

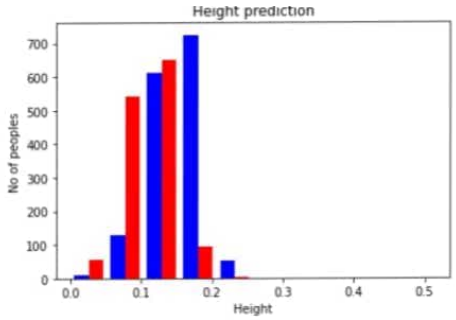
Out[11]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

In [13]:
```
#Univariate Analysis
df["Diameter"].plot(kind='hist');
```

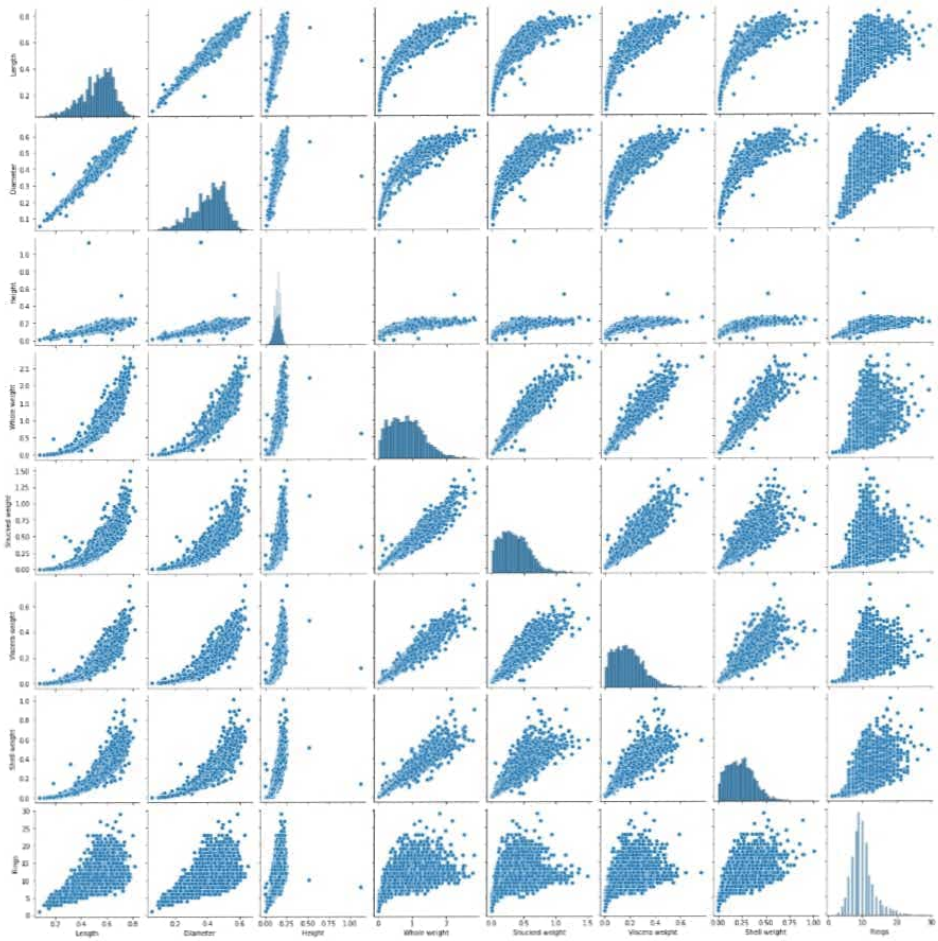


In [14]:
```
#Bi-Variate Analysis
cy=df[df.Sex=="M"].Height
cn=df[df.Sex=="F"].Height
cn=df[df.Sex=="I"].Height
plt.title("Height prediction")
plt.xlabel("Height")
plt.ylabel("No of peoples")
plt.hist([cy,cn],color=['blue','red'],label=["Height=yes"])
plt.show()
```



In [15]:
```
#Multi-variate Analysis
sns.pairplot(df)
```

Out[15]:
```
<seaborn.axisgrid.PairGrid at 0x7fc457c84d90>
```



In [16]:
```
#Descriptive statistics
df.describe()
```

Out[16]:

	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

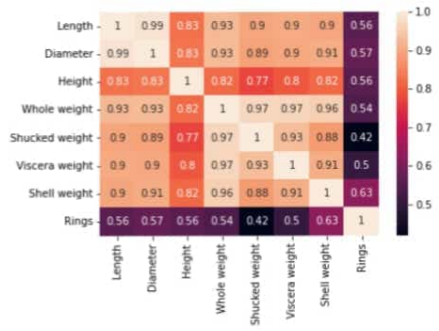
In [17]:
```
df.corr()
```

Out[17]:

	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
Length	1.000000	0.986812	0.827554	0.925261	0.897914	0.903018	0.897706	0.556720
Diameter	0.986812	1.000000	0.833684	0.925452	0.893162	0.899724	0.905330	0.574660
Height	0.827554	0.833684	1.000000	0.819221	0.774972	0.798319	0.817338	0.557467
Whole weight	0.925261	0.925452	0.819221	1.000000	0.969405	0.966375	0.955355	0.540390
Shucked weight	0.897914	0.893162	0.774972	0.969405	1.000000	0.931961	0.882617	0.420884
Viscera weight	0.903018	0.899724	0.798319	0.966375	0.931961	1.000000	0.907656	0.503819
Shell weight	0.897706	0.905330	0.817338	0.955355	0.882617	0.907656	1.000000	0.627574
Rings	0.556720	0.574660	0.557467	0.540390	0.420884	0.503819	0.627574	1.000000

In [18]:
```
sns.heatmap(df.corr(),annot=True)
```

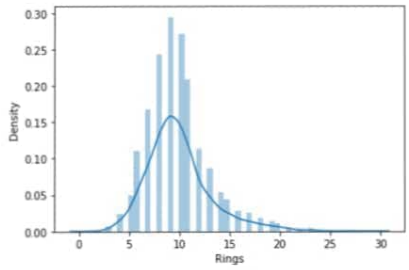
Out[18]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc453040410>
```



In [19]:
```
sns.distplot(df.Rings)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a
deprecated function and will be removed in a future version. Please adapt your code to use either `di
splot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for h
istograms).
  warnings.warn(msg, FutureWarning)
```

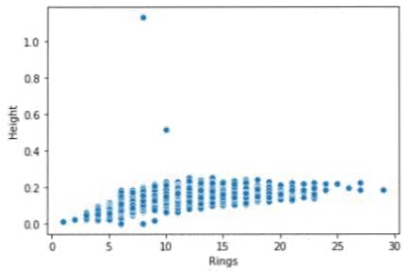
Out[19]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc451750890>
```



In [20]:
```
sns.scatterplot(df.Rings,df.Height)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following v
ariables as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```

Out[20]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc45165b550>
```



In [21]:
```
#Handle The Missing values
df.isnull().any()
```

Out[21]:
```
Sex               False
Length            False
Diameter          False
Height            False
Whole weight      False
Shucked weight    False
Viscera weight    False
Shell weight      False
Rings             False
dtype: bool
```

In [22]:
```
df.isnull().sum()
```

Out[22]:
```
Sex               0
Length            0
Diameter          0
Height            0
Whole weight      0
Shucked weight    0
Viscera weight    0
Shell weight      0
Rings             0
dtype: int64
```

In [23]:
```
#Find the outliers
df.skew()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: FutureWarning: Dropping of nuisance c
olumns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this wil
l raise TypeError.  Select only valid columns before calling the reduction.
```

Out[23]:
```
Length           -0.639873
Diameter         -0.609198
Height            3.128817
Whole weight      0.530959
Shucked weight    0.719098
Viscera weight    0.591852
Shell weight      0.620927
Rings             1.114102
dtype: float64
```

In [25]:
```
#Split data into dependent and independent variables
x=df.iloc[:,3:13].values
y=df.iloc[:,13:14].values
```

In [26]:
```
x.shape
```

Out[26]:
```
(4177, 6)
```

In [27]:
```
y.shape
```

Out[27]:
```
(4177, 0)
```

In [28]:
```
#Categorical colums and encoding
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
ct=ColumnTransformer([("oh",OneHotEncoder(),[1,2])],remainder="passthrough")
x=ct.fit_transform(x)
x.shape
```

Out[28]:
```
(4177, 3948)
```

In [29]:
```
df["Sex"].unique()
```

Out[29]:
```
array(['M', 'F', 'I'], dtype=object)
```

In [30]:
```
#Split the data into training and testing
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,random_state=0)
x_train.shape
```

Out[30]:
```
(3341, 3948)
```

In [31]:
```
x_test.shape
```

Out[31]:
```
(836, 3948)
```

In [32]:
```
y_test
```

Out[32]:
```
array([], shape=(836, 0), dtype=float64)
```

In [33]:
```
#Scale the independent variables
X = df.iloc[:, :-1].values
print(X)
```

```
[['M' 0.455 0.365 ... 0.2245 0.101 0.15]
 ['M' 0.35 0.265 ... 0.0995 0.0485 0.07]
 ['F' 0.53 0.42 ... 0.2565 0.1415 0.21]
 ...
 ['M' 0.6 0.475 ... 0.5255 0.2875 0.308]
 ['F' 0.625 0.485 ... 0.531 0.261 0.296]
 ['M' 0.71 0.555 ... 0.9455 0.3765 0.495]]
```

In [49]:
```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
```

In [52]:
```
import joblib
joblib.dump(ct,"abalone.pkl")
```

Out[52]:
```
['abalone.pkl']
```

In [51]:
```
joblib.dump(sc,"abalone.pkl")
```

Out[51]:
```
['abalone.pkl']
```

In [37]:
```
#Split the data into training and testing
from sklearn.model_selection import train_test_split

train, test = train_test_split(df, test_size=0.2)
```

In [39]:
```
# Build the Model
my_dict=pd.read_csv("/content/abalone.csv")
df = pd.DataFrame(my_dict)
print(df)
```

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

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

[4177 rows x 9 columns]
```

In [53]:

```
#  Build the Model
import csv
with open("/content/abalone.csv") as csv_file:
    csv_reader = csv.reader(csv_file)
    df = pd.DataFrame([csv_reader], index = None)
for val in list(df[1]):
    print(val)
```

```
['M', '0.455', '0.365', '0.095', '0.514', '0.2245', '0.101', '0.15', '15']
```

In [71]:

```
# Training and Testing Module
from sklearn.model_selection import train_test_split
print(train)
print(test)
```

```
     Sex  Length  Diameter  Height  Whole weight  Shucked weight  \
3771   I   0.575     0.450   0.145        0.7950          0.3640
2385   F   0.450     0.345   0.115        0.4960          0.1905
612    M   0.325     0.240   0.075        0.1550          0.0475
3047   M   0.590     0.435   0.165        0.9765          0.4525
2865   I   0.315     0.235   0.075        0.1285          0.0510
...   ..     ...       ...     ...           ...             ...
3234   F   0.590     0.485   0.205        1.2315          0.4525
1379   F   0.620     0.475   0.160        1.1295          0.4630
1483   M   0.590     0.440   0.150        0.8725          0.3870
946    F   0.470     0.365   0.120        0.5820          0.2900
3867   F   0.500     0.390   0.130        0.6355          0.2505

      Viscera weight  Shell weight  Rings
3771          0.1505        0.2600     10
2385          0.1170        0.1400     12
612           0.0355        0.0600      9
3047          0.2395        0.2350      9
2865          0.0280        0.0405      4
...              ...           ...    ...
3234          0.2380        0.4200     13
1379          0.2685        0.3300     10
1483          0.2150        0.2450      8
946           0.0920        0.1460      8
3867          0.1635        0.1950     15

[3341 rows x 9 columns]
     Sex  Length  Diameter  Height  Whole weight  Shucked weight  \
3582   F   0.625     0.475   0.160        1.3335          0.6050
288    M   0.440     0.355   0.125        0.4775          0.1320
2224   F   0.550     0.425   0.145        0.7970          0.2970
3502   M   0.610     0.470   0.175        1.2140          0.5315
1490   F   0.605     0.485   0.160        1.2010          0.4170
...   ..     ...       ...     ...           ...             ...
875    M   0.630     0.505   0.165        1.2600          0.4525
1220   I   0.330     0.250   0.095        0.2085          0.1020
2613   F   0.635     0.495   0.195        1.2970          0.5560
523    M   0.200     0.140   0.055        0.0350          0.0145
1177   F   0.645     0.480   0.170        1.1345          0.5280

      Viscera weight  Shell weight  Rings
3582          0.2875        0.319     10
288           0.0815        0.190      9
2224          0.1500        0.265      9
3502          0.2835        0.325     10
1490          0.2875        0.380      9
...              ...           ...    ...
875           0.2755        0.406     14
1220          0.0395        0.052      7
2613          0.2985        0.370     11
523           0.0080        0.010      5
1177          0.2540        0.305     10

[836 rows x 9 columns]
```

In [ ]:

```
#Measure the performance using matrics
from __future__ import print_function

import pandas as pd
path = "/abalone.csv"
merged = pd.read_csv(path, error_bad_lines=False, low_memory=False)

X = merged.text
y = merged.grid

from sklearn.feature_extraction.text import CountVectorizer
vect = CountVectorizer()

X_train_dtm = vect.fit_transform(X_train)
X_test_dtm = vect.transform(X_test)

from sklearn.naive_bayes import MultinomialNB
nb = MultinomialNB()

nb.fit(X_train_dtm, y_train)

y_pred_class = nb.predict(X_test_dtm)

from sklearn import metrics
print(metrics.classification_report(y_test, y_pred_class))
```