1 DOWNLOAD THE DATASET

2 LOAD THE DATASET

In [1]:

```
import pandas as pd
import numpy as np
df =pd.read_csv('abalone.csv')
df.head()
```

Out[1]:

	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

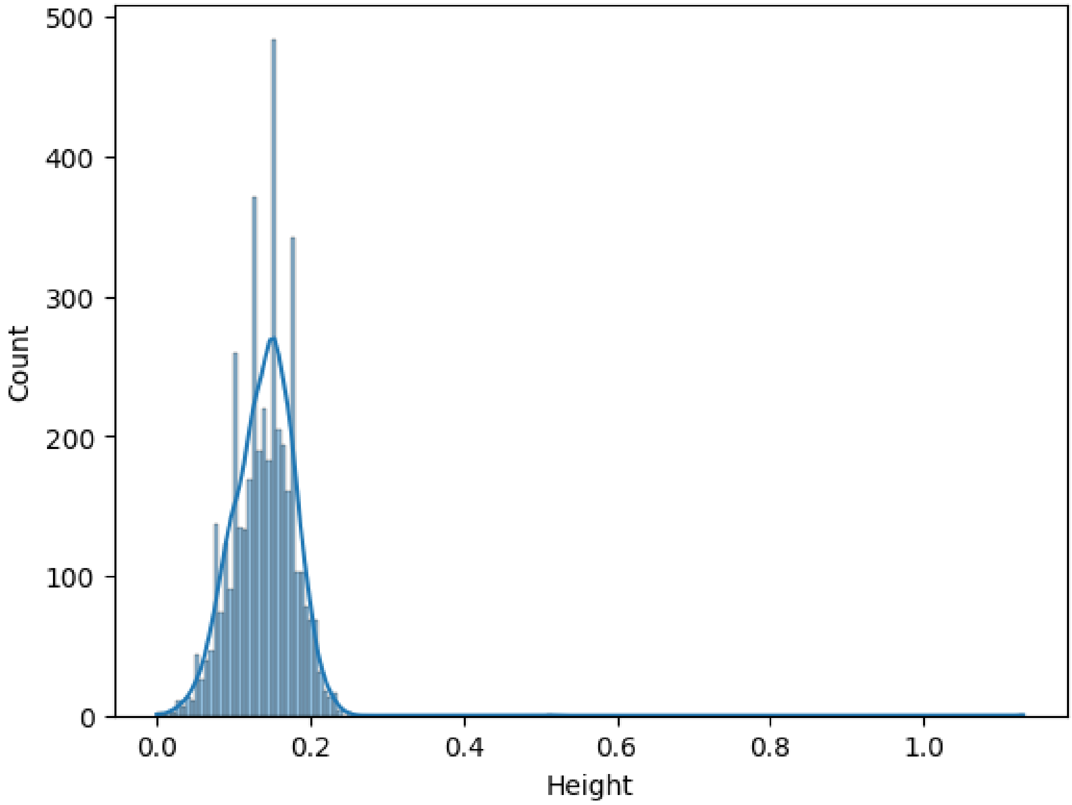
3. PERFORM BELOW VISUALIZATIONS

Univariate Analysis

```
import seaborn as sns
sns.histplot(df.Height,kde=True)
```

Out[2]:

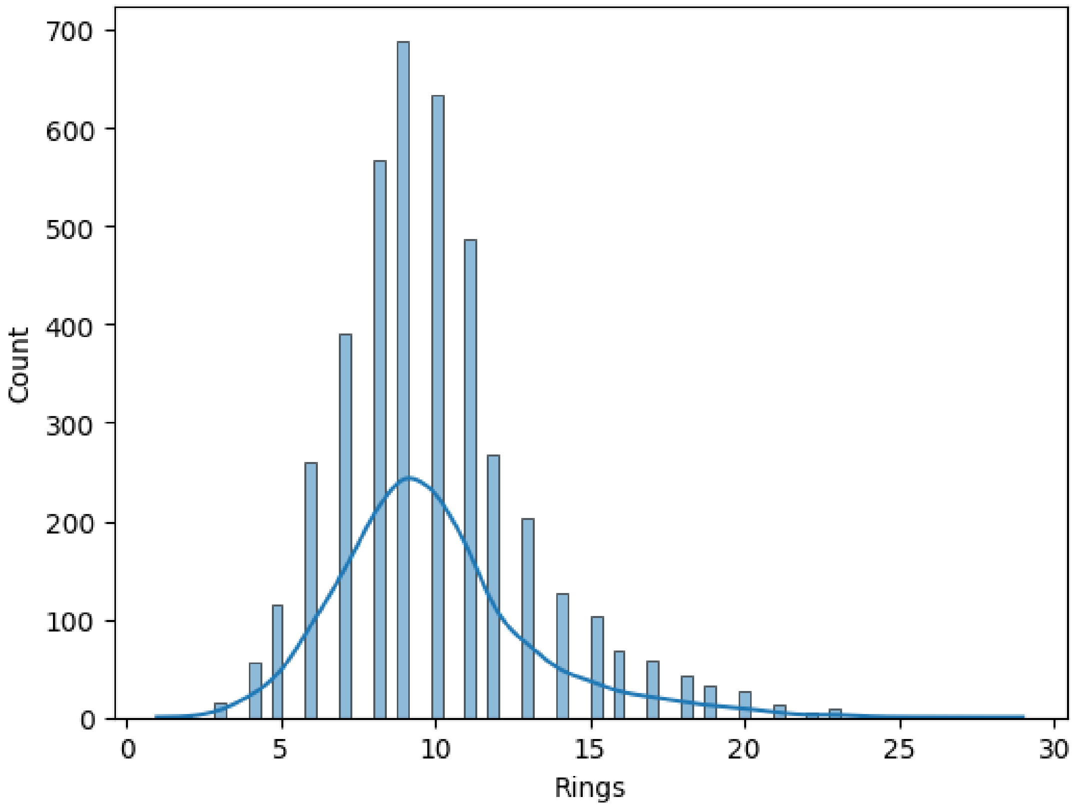
```
<AxesSubplot: xlabel='Height', ylabel='Count'>
```




```
import seaborn as sns
sns.histplot(df.Rings,kde=True)
```

Out[3]:

```
<AxesSubplot: xlabel='Rings', ylabel='Count'>
```



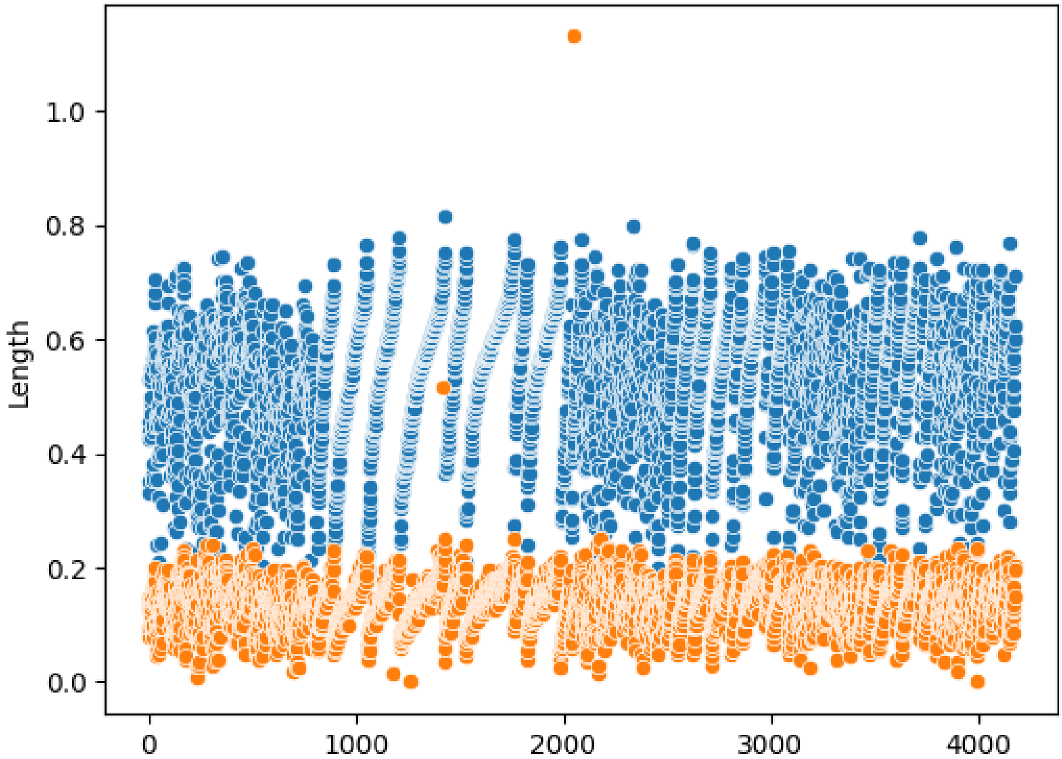
Bivariate analysis

In [4]:

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.scatterplot(df.Length)
sns.scatterplot(df.Height)
```

Out[4]:

```
<AxesSubplot: ylabel='Length'>
```



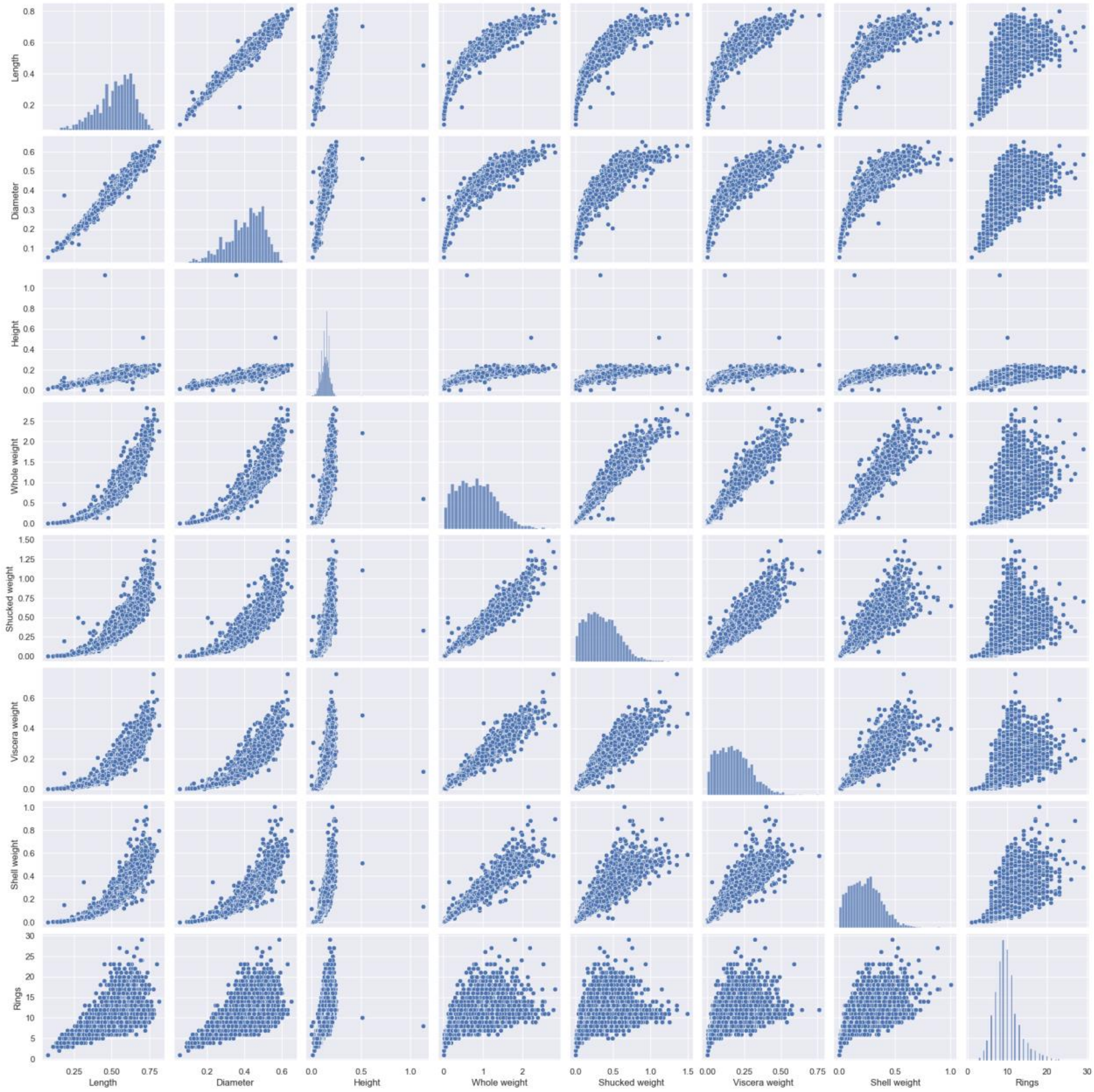
MULTIVARIATE ANALYSIS

In [16]:

```
import seaborn as sns
df= pd.read_csv("abalone.csv")
sns.pairplot(df)
```

Out[16]:

```
<seaborn.axisgrid.PairGrid at 0x18752216650>
```



4. Perform descriptive statistics on the dataset.

In [15]:

```
df=pd.read_csv("abalone.csv")
df.describe(include='all')
```

Out[15]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
count	4177	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
unique	3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
top	M	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
freq	1528	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
mean	NaN	0.523992	0.407881	0.139516	0.828742	0.359367	0.180594	0.238831	9.933684
std	NaN	0.120093	0.099240	0.041827	0.490389	0.221963	0.109614	0.139203	3.224169
min	NaN	0.075000	0.055000	0.000000	0.002000	0.001000	0.000500	0.001500	1.000000
25%	NaN	0.450000	0.350000	0.115000	0.441500	0.186000	0.093500	0.130000	8.000000
50%	NaN	0.545000	0.425000	0.140000	0.799500	0.336000	0.171000	0.234000	9.000000
75%	NaN	0.615000	0.480000	0.165000	1.153000	0.502000	0.253000	0.329000	11.000000
max	NaN	0.815000	0.650000	1.130000	2.825500	1.488000	0.760000	1.005000	29.000000

5. Check for Missing values and deal with them.

```
from ast import increment_lineno
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(color_codes=True)
df=pd.read_csv("abalone.csv")
df.head()
```

Out[14]:

	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

6. Find the outliers and replace the outliers

In [13]:

```
import pandas as pd
import matplotlib
from matplotlib import pyplot as pyplot
%matplotlib inline
matplotlib.rcParams['figure.figsize']=(11,6)
df=pd.read_csv("abalone.csv")
df.sample(10)
```

Out[13]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
2590	F	0.580	0.445	0.145	0.8880	0.4100	0.1815	0.2425	8
687	F	0.535	0.405	0.125	0.9270	0.2600	0.1		
3495	M	0.560	0.415	0.130	0.7615	0.3695	0.1700	0.1955	8
90	M	0.565	0.425	0.135	0.8115	0.3410	0.1675	0.2550	15
2083	M	0.685	0.550	0.200	1.7725	0.8130	0.3870	0.4900	11
3799	F	0.705	0.560	0.170	1.4575	0.6070	0.3180	0.4400	11

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
4134	F	0.595	0.455	0.140	0.9140	0.3895	0.2225	0.2710	9
1137	F	0.575	0.450	0.160	1.0680	0.5560	0.2140	0.2575	10
1041	F	0.675	0.570	0.225	1.5870	0.7390	0.2995	0.4350	10
2292	F	0.380	0.300	0.090	0.3215	0.1545	0.0750	0.0950	9

7. Check for Categorical columns and perform encoding.

In [12]:

```
df=pd.read_csv("abalone.csv")
df.columns
import pandas as pd
import numpy as np
headers=['Sex','Length','Diameter','Height','Whole weight','Shucked
weight','Viscera weight','Shell weight','Rings']
import seaborn as sns
df.head()
```

	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

8. Split the data into dependent and independent variables.

In [11]:

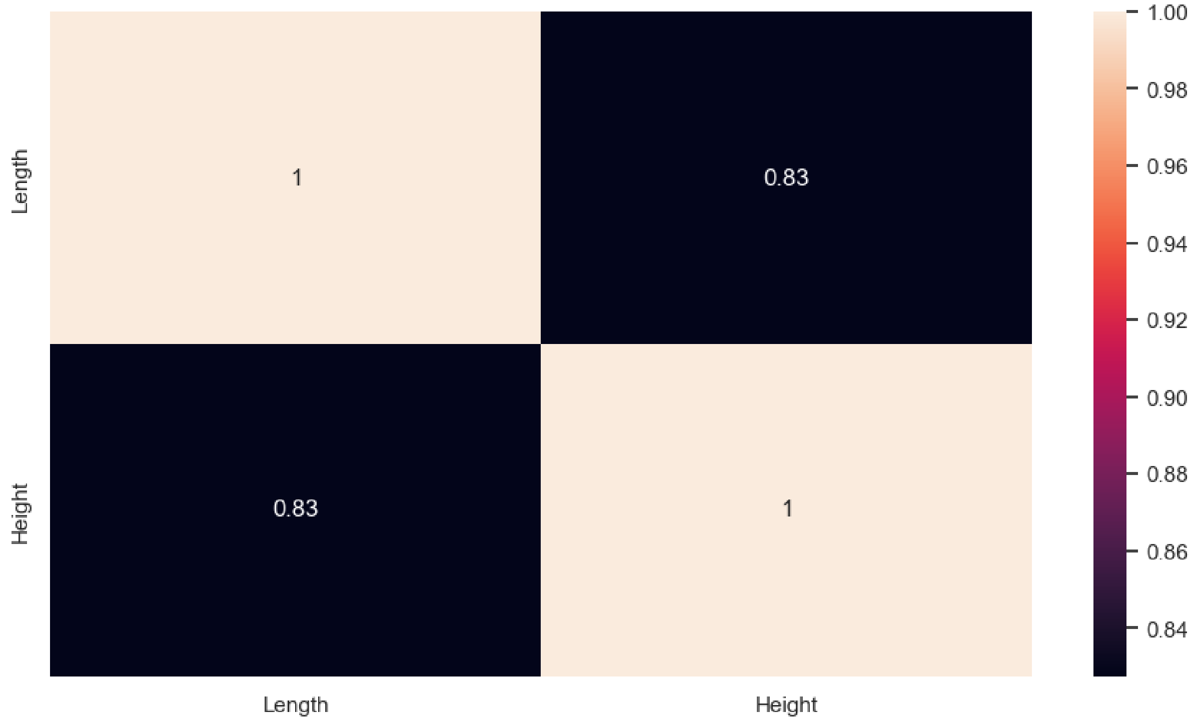
```
x=df.iloc[:,:-1].values
print(x)
y=df.iloc[:,-1]._values
print(y)
```

```
[['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]]
[15  7  9 ...  9 10 12]
```

9. Scale the independent variables

In [10]:

```
import seaborn as sns
df=pd.read_csv("abalone.csv")
dff=df[['Length','Height']]
sns.heatmap(dff.corr(), annot=True)
sns.set(rc={'figure.figsize':(40,40)})
```



10. Split the data into training and testing

In [23]:

```
from scipy.sparse.construct import random
x=df.iloc[:, 1:2].values
```

```
y=df.iloc[:,2].values
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)
print('Row count of x_train table'+'-'+str(f"{len(x_train):,}"))
print('Row count of y_train table'+'-'+str(f"{len(y_train):,}"))
print('Row count of x_test table'+'-'+str(f"{len(x_test):,}"))
print('Row count of y_test table'+'-'+str(f"{len(y_test):,}"))
```

```
Row count of x_train table-3,341
Row count of y_train table-3,341
Row count of x_test table-836
Row count of y_test table-836
```

11. Build the Model

In [26]:

```
from sklearn.linear_model import LinearRegression
model=LinearRegression()
```

12. Train the Model

In [27]:

```
model.fit(x_train,y_train)
```

Out[27]:

LinearRegression()

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

13. Test the Model

In [28]:

```
pred=model.predict(x_test)
```

```
pred=model.predict(x_test)
pred
```

Out[28]:

```
array([0.42910815, 0.38837123, 0.48613984, 0.16024445, 0.50650831,
       0.510582  , 0.35985538, 0.37615015, 0.31097107, 0.48613984,
       0.36800276, 0.22134984, 0.31504476, 0.39244492, 0.16839184,
       0.45355031, 0.26616045, 0.44132923, 0.47799246, 0.35985538,
       0.29874999, 0.27838153, 0.35578169, 0.26208676, 0.43725554,
       0.43725554, 0.10728645, 0.49836092, 0.44540292, 0.41688707,
       0.29060261, 0.11136014, 0.457624  , 0.57983477, 0.33948692,
       0.4005923 , 0.36800276, 0.48206615, 0.33541323, 0.49836092,
       0.44132923, 0.38022384, 0.404666  , 0.41688707, 0.46984508,
       0.35985538, 0.47799246, 0.52687677, 0.41688707, 0.31911846,
       0.457624  , 0.30282369, 0.33541323, 0.46577138, 0.39244492,
       0.36392907, 0.1887603 , 0.25801307, 0.31504476, 0.28652892,
       0.48206615, 0.48613984, 0.457624  , 0.38022384, 0.31097107,
```

0.510582 , 0.55131892, 0.50650831, 0.42096077, 0.53095046,
0.40873969, 0.57576108, 0.49428723, 0.457624 , 0.46169769,
0.38837123, 0.45355031, 0.40873969, 0.47799246, 0.38022384,
0.48613984, 0.20912876, 0.31097107, 0.44947661, 0.43725554,
0.36800276, 0.47799246, 0.50650831, 0.12358122, 0.28245522,
0.45355031, 0.44132923, 0.35578169, 0.27023415, 0.41688707,
0.21727615, 0.404666 , 0.28245522, 0.404666 , 0.56761369,
0.48206615, 0.36392907, 0.53502415, 0.49021354, 0.15617076,
0.50243461, 0.38837123, 0.27838153, 0.27023415, 0.48206615,
0.56761369, 0.55946631, 0.38837123, 0.44947661, 0.45355031,
0.56354 , 0.51465569, 0.46169769, 0.20912876, 0.41688707,
0.27838153, 0.42503446, 0.28652892, 0.36800276, 0.44947661,
0.48613984, 0.47391877, 0.53095046, 0.31911846, 0.49836092,
0.48206615, 0.36800276, 0.39244492, 0.510582 , 0.53502415,
0.48613984, 0.39651861, 0.4005923 , 0.28652892, 0.510582 ,
0.35985538, 0.351708 , 0.26616045, 0.44540292, 0.46169769,
0.44132923, 0.35578169, 0.38429753, 0.37207646, 0.2417183 ,
0.38429753, 0.53502415, 0.2946763 , 0.49836092, 0.25801307,
0.28245522, 0.46984508, 0.34356061, 0.46577138, 0.23764461,
0.38837123, 0.25393938, 0.41281338, 0.46984508, 0.17653922,
0.46984508, 0.27838153, 0.2417183 , 0.42503446, 0.43318184,
0.44947661, 0.15209707, 0.35578169, 0.49021354, 0.49021354,
0.49836092, 0.18468661, 0.3476343 , 0.28652892, 0.43318184,
0.40873969, 0.47391877, 0.404666 , 0.42096077, 0.43318184,
0.41688707, 0.457624 , 0.38429753, 0.24986568, 0.38837123,
0.45355031, 0.4005923 , 0.38837123, 0.48206615, 0.43318184,
0.35578169, 0.31504476, 0.40873969, 0.12358122, 0.53502415,
0.36392907, 0.38837123, 0.56761369, 0.43318184, 0.47799246,
0.47799246, 0.20912876, 0.52280308, 0.404666 , 0.3476343 ,
0.1887603 , 0.44947661, 0.42910815, 0.46577138, 0.47391877,
0.44132923, 0.46169769, 0.52280308, 0.1887603 , 0.54724523,
0.404666 , 0.510582 , 0.37615015, 0.50243461, 0.50650831,
0.42910815, 0.48613984, 0.40873969, 0.60835062, 0.49021354,
0.42503446, 0.40873969, 0.2417183 , 0.53502415, 0.4005923 ,
0.46169769, 0.40873969, 0.17653922, 0.40873969, 0.43318184,
0.39244492, 0.46577138, 0.42910815, 0.55539262, 0.42096077,
0.46577138, 0.48206615, 0.36800276, 0.48206615, 0.51465569,
0.49836092, 0.37615015, 0.53502415, 0.56761369, 0.43318184,
0.37615015, 0.28652892, 0.38429753, 0.47799246, 0.51465569,
0.49021354, 0.36392907, 0.44540292, 0.24986568, 0.41281338,
0.36800276, 0.37207646, 0.351708 , 0.31504476, 0.26616045,
0.44132923, 0.32319215, 0.27430784, 0.31504476, 0.22542353,
0.50243461, 0.47391877, 0.351708 , 0.49428723, 0.34356061,
0.2417183 , 0.58390846, 0.26208676, 0.50243461, 0.51872938,
0.45355031, 0.35985538, 0.48206615, 0.43725554, 0.46577138,
0.37615015, 0.404666 , 0.22542353, 0.45355031, 0.28245522,
0.37207646, 0.25801307, 0.20098138, 0.46169769, 0.50243461,
0.351708 , 0.37207646, 0.54317154, 0.41688707, 0.35578169,
0.32726584, 0.39651861, 0.44540292, 0.44540292, 0.45355031,
0.50243461, 0.510582 , 0.35985538, 0.48206615, 0.44540292,
0.43725554, 0.38837123, 0.27023415, 0.351708 , 0.41688707,
0.52687677, 0.43725554, 0.33948692, 0.54724523, 0.44947661,
0.41688707, 0.41281338, 0.42096077, 0.48613984, 0.49021354,
0.39651861, 0.54317154, 0.15617076, 0.52687677, 0.45355031,
0.53502415, 0.29874999, 0.37615015, 0.49836092, 0.42910815,
0.37207646, 0.38837123, 0.50243461, 0.48613984, 0.43318184,
0.351708 , 0.21727615, 0.39244492, 0.44132923, 0.29874999,

```
       0.20912876, 0.44947661, 0.52687677, 0.58390846, 0.47799246,
       0.38022384, 0.48206615, 0.49021354, 0.35578169, 0.39651861,
       0.37615015, 0.47799246, 0.37207646, 0.46577138, 0.49836092,
       0.20505507, 0.20505507, 0.404666  , 0.42503446, 0.38837123,
       0.457624  , 0.30282369, 0.48613984, 0.351708  , 0.52280308,
       0.49428723, 0.50243461, 0.33541323, 0.38837123, 0.11950753,
       0.38022384, 0.46169769, 0.46169769, 0.47799246, 0.20505507,
       0.51465569, 0.57168739, 0.36392907, 0.32319215, 0.22949722,
       0.54724523, 0.46577138, 0.42910815, 0.37207646, 0.52280308,
       0.30282369, 0.50243461, 0.37615015, 0.44540292, 0.38837123,
       0.34356061, 0.457624  , 0.28245522, 0.54317154, 0.33133953,
       0.55131892, 0.20912876, 0.50650831, 0.51465569, 0.38429753,
       0.38837123, 0.36800276, 0.53502415, 0.42910815, 0.50650831,
       0.44132923, 0.45355031, 0.47391877, 0.55539262, 0.54724523,
       0.44132923, 0.42096077, 0.44132923, 0.50243461, 0.21727615,
       0.44540292, 0.42910815, 0.50650831, 0.47799246, 0.43725554,
0.41688707, 0.51872938, 0.39244492, 0.44540292, 0.33948692,
       0.36800276, 0.53502415, 0.42910815, 0.44540292, 0.49021354,
       0.31504476, 0.25393938, 0.44132923, 0.32319215, 0.28652892,
       0.28652892, 0.45355031, 0.18468661, 0.39244492, 0.50243461,
       0.10728645, 0.27430784, 0.49428723, 0.26616045, 0.20505507,
       0.31911846, 0.48206615, 0.42910815, 0.35578169, 0.47391877,
       0.31911846, 0.60020323, 0.39244492, 0.45355031, 0.36392907,
       0.55131892, 0.2417183 , 0.57576108, 0.33948692, 0.37207646,
       0.45355031, 0.41281338, 0.35578169, 0.49428723, 0.41281338,
       0.351708  , 0.33541323, 0.33948692, 0.27023415, 0.49836092,
       0.4005923 , 0.51465569, 0.34356061, 0.41688707, 0.50650831,
       0.51872938, 0.43725554, 0.39651861, 0.40873969, 0.50650831,
       0.11543383, 0.43318184, 0.43725554, 0.43725554, 0.25393938,
       0.30689738, 0.30689738, 0.22134984, 0.44947661, 0.49428723,
       0.29060261, 0.49836092, 0.30282369, 0.45355031, 0.49428723,
       0.49021354, 0.39244492, 0.48206615, 0.27430784, 0.20912876,
       0.38022384, 0.46169769, 0.51872938, 0.48613984, 0.35985538,
       0.37207646, 0.44540292, 0.54317154, 0.510582  , 0.44132923,
       0.42910815, 0.42096077, 0.50243461, 0.48206615, 0.24986568,
       0.49836092, 0.3476343 , 0.38022384, 0.39651861, 0.38837123,
       0.31911846, 0.33133953, 0.31097107, 0.40873969, 0.17246553,
       0.45355031, 0.49021354, 0.52687677, 0.41688707, 0.51465569,
       0.4005923 , 0.44132923, 0.31911846, 0.457624  , 0.27430784,
       0.44540292, 0.39244492, 0.35578169, 0.47391877, 0.47799246,
       0.43318184, 0.51465569, 0.33541323, 0.51872938, 0.49428723,
       0.3476343 , 0.36800276, 0.42910815, 0.46577138, 0.40873969,
       0.40873969, 0.47391877, 0.50243461, 0.43318184, 0.56761369,
       0.19283399, 0.33133953, 0.44947661, 0.44540292, 0.351708  ,
       0.43725554, 0.41688707, 0.39651861, 0.44947661, 0.41688707,
       0.31504476, 0.41688707, 0.44947661, 0.56761369, 0.12765491,
       0.50243461, 0.20505507, 0.1887603 , 0.22134984, 0.39244492,
       0.44540292, 0.56354   , 0.44132923, 0.4005923 , 0.48613984,
       0.19690768, 0.351708  , 0.49021354, 0.27023415, 0.47799246,
       0.52687677, 0.09506537, 0.3476343 , 0.23764461, 0.49021354,
       0.38022384, 0.53502415, 0.46984508, 0.47799246, 0.49428723,
       0.38837123, 0.46169769, 0.24986568, 0.42910815, 0.49428723,
       0.31911846, 0.52687677, 0.44947661, 0.4005923 , 0.42910815,
       0.54724523, 0.37615015, 0.31097107, 0.55131892, 0.44540292,
       0.39244492, 0.52687677, 0.30689738, 0.44132923, 0.46984508,
       0.43725554, 0.34356061, 0.47391877, 0.32726584, 0.42910815,
       0.31911846, 0.27838153, 0.29060261, 0.33541323, 0.2946763 ,
```

```
       0.45355031, 0.15617076, 0.56354   , 0.36392907, 0.42910815,
       0.34356061, 0.55539262, 0.49428723, 0.48206615, 0.46577138,
       0.39651861, 0.31911846, 0.42910815, 0.48613984, 0.37615015,
       0.29874999, 0.38429753, 0.33948692, 0.45355031, 0.28652
0.47799246, 0.31097107, 0.32319215, 0.43725554, 0.49428723,
       0.35578169, 0.33541323, 0.46169769, 0.41281338, 0.38022384,
       0.38837123, 0.36392907, 0.24986568, 0.25393938, 0.41281338,
       0.44947661, 0.60835062, 0.51872938, 0.51465569, 0.16839184,
       0.33133953, 0.33133953, 0.2417183 , 0.42503446, 0.55131892,
       0.46577138, 0.43318184, 0.48206615, 0.351708  , 0.41688707,
       0.33541323, 0.42096077, 0.457624  , 0.41688707, 0.55539262,
       0.47391877, 0.457624  , 0.40873969, 0.51872938, 0.404666  ,
       0.50243461, 0.55131892, 0.42503446, 0.41281338, 0.33541323,
       0.27430784, 0.46169769, 0.48613984, 0.457624  , 0.38837123,
       0.45355031, 0.38837123, 0.351708  , 0.51465569, 0.48206615,
       0.23764461, 0.17246553, 0.20912876, 0.46577138, 0.32726584,
       0.41688707, 0.44540292, 0.40873969, 0.33133953, 0.33541323,
       0.44132923, 0.14802337, 0.39651861, 0.27430784, 0.46577138,
       0.20505507, 0.50243461, 0.39651861, 0.40873969, 0.48613984,
       0.42910815, 0.49836092, 0.46984508, 0.54317154, 0.404666  ,
       0.40873969, 0.40873969, 0.51465569, 0.30689738, 0.21320245,
       0.44132923, 0.42096077, 0.26616045, 0.33541323, 0.44132923,
       0.29874999, 0.47391877, 0.43725554, 0.39244492, 0.33948692,
       0.44540292, 0.43725554, 0.27023415, 0.53502415, 0.31504476,
       0.47799246, 0.38022384, 0.29874999, 0.44947661, 0.49021354,
       0.27430784, 0.22542353, 0.27430784, 0.20505507, 0.45355031,
       0.44947661, 0.53095046, 0.48206615, 0.47391877, 0.35985538,
       0.27838153, 0.32726584, 0.41281338, 0.49428723, 0.41281338,
       0.404666  , 0.29060261, 0.404666  , 0.30282369, 0.46169769,
       0.52280308, 0.42096077, 0.3476343 , 0.24986568, 0.3476343 ,
       0.44947661, 0.404666  , 0.33133953, 0.49836092, 0.42910815,
       0.41688707, 0.53502415, 0.53095046, 0.43725554, 0.44540292,
       0.44947661, 0.42910815, 0.47391877, 0.11543383, 0.40873969,
       0.36392907, 0.33133953, 0.33541323, 0.22134984, 0.24986568,
       0.33541323, 0.4005923 , 0.51872938, 0.31911846, 0.38429753,
       0.37207646, 0.35985538, 0.44540292, 0.41688707, 0.55946631,
       0.404666  , 0.46577138, 0.46577138, 0.42096077, 0.36392907,
       0.22542353, 0.42910815, 0.510582  , 0.25801307, 0.38837123,
       0.43725554, 0.48206615, 0.37615015, 0.25393938, 0.25393938,
       0.41281338, 0.47799246, 0.31504476, 0.34356061, 0.42096077,
       0.18468661])
```

In [29]:

```
y_p=model.predict([[2.2]])
y_p
```

Out[29]:

```
array([1.77342665])
```

In [ ]: