In [1]:

```python
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns

```

In [2]:

```python
data = pd.read_csv('D:\IBM PROJECT\dataset/abalone.csv')

# getting the shape
data.shape
```

Out[2]:

```
(4177, 9)
```

In [3]:

```python
data.head()

```

Out[3]:

	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 [4]:

```python
data.describe()
```

Out[4]:

	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight
count	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
std	0.120093	0.099240	0.041827	0.490389	0.221963	0.109614	0.139203
min	0.075000	0.055000	0.000000	0.002000	0.001000	0.000500	0.001500
25%	0.450000	0.350000	0.115000	0.441500	0.186000	0.093500	0.130000
50%	0.545000	0.425000	0.140000	0.799500	0.336000	0.171000	0.234000
75%	0.615000	0.480000	0.165000	1.153000	0.502000	0.253000	0.329000
max	0.815000	0.650000	1.130000	2.825500	1.488000	0.760000	1.005000

In [5]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4177 entries, 0 to 4176
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Sex             4177 non-null   object 
 1   Length          4177 non-null   float64
 2   Diameter        4177 non-null   float64
 3   Height          4177 non-null   float64
 4   Whole weight    4177 non-null   float64
 5   Shucked weight  4177 non-null   float64
 6   Viscera weight  4177 non-null   float64
 7   Shell weight    4177 non-null   float64
 8   Rings           4177 non-null   int64  
dtypes: float64(7), int64(1), object(1)
memory usage: 293.8+ KB
```

In [6]:

```
data.isnull().sum()
```

Out[6]:

```
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 [7]:

```
data.isnull()
```

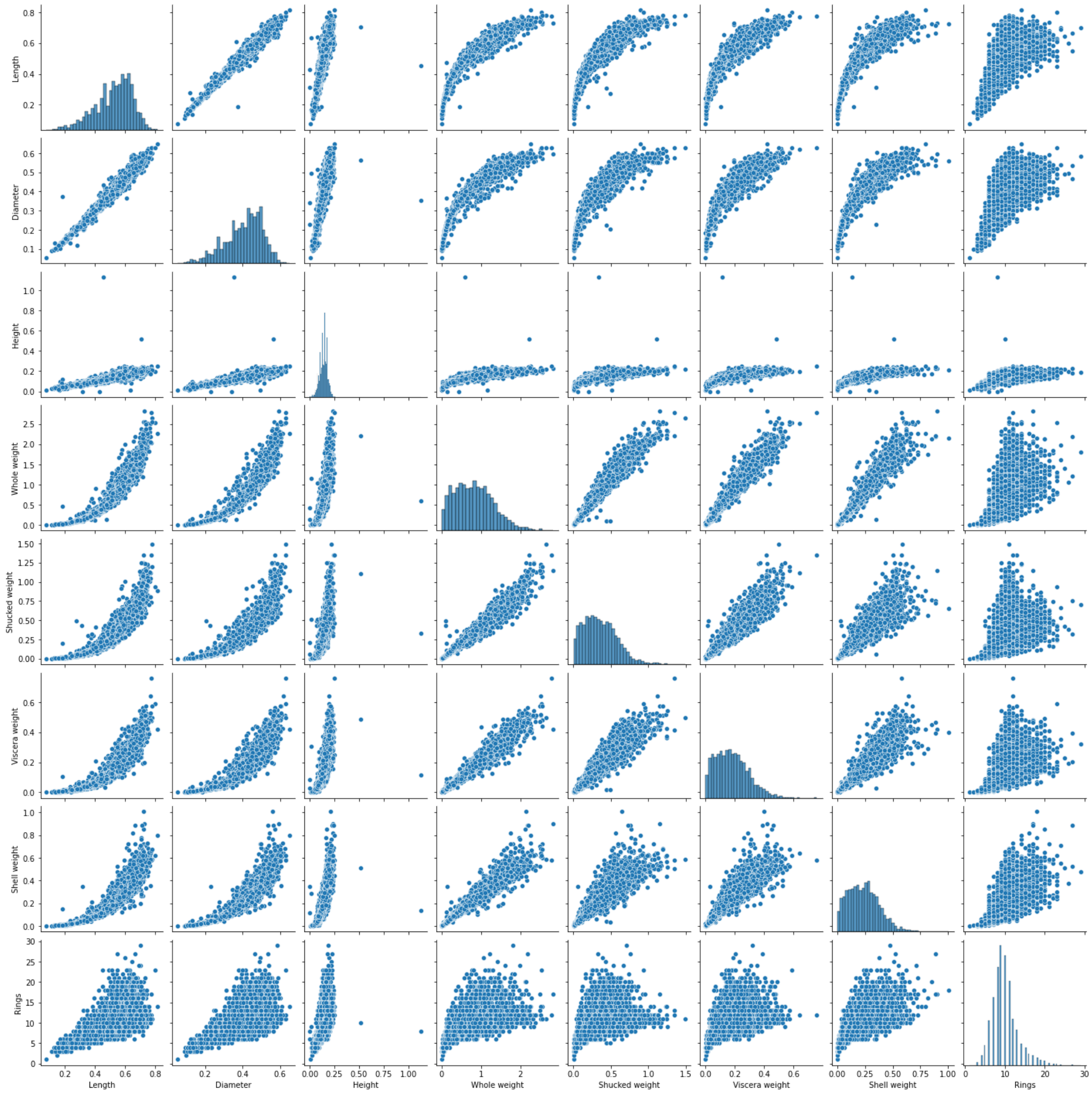
Out[7]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...
4172	False	False	False	False	False	False	False	False	False
4173	False	False	False	False	False	False	False	False	False
4174	False	False	False	False	False	False	False	False	False
4175	False	False	False	False	False	False	False	False	False
4176	False	False	False	False	False	False	False	False	False

4177 rows × 9 columns

In [8]:
```
sns.pairplot(data)
```

Out[8]:
```
<seaborn.axisgrid.PairGrid at 0x24972beac70>
```



In [9]:

```
data.columns

```

Out[9]:

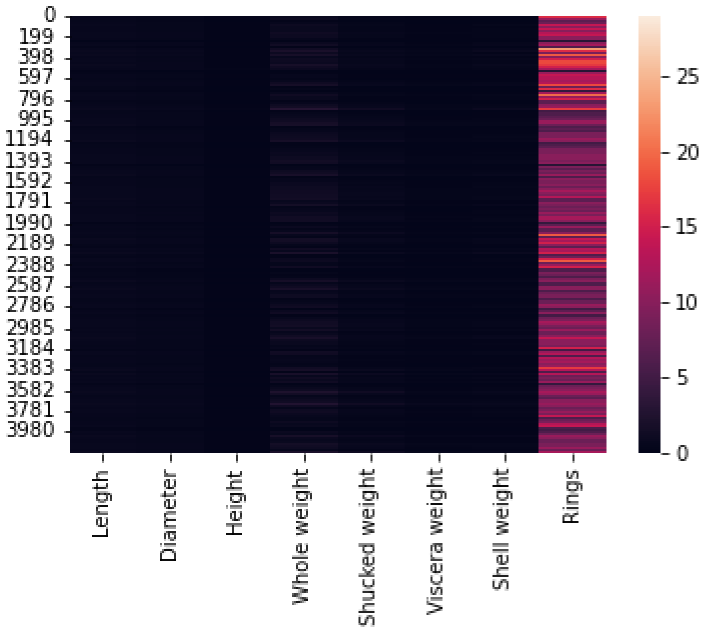
```
Index(['Sex', 'Length', 'Diameter', 'Height', 'Whole weight', 'Shucked weight',
       'Viscera weight', 'Shell weight', 'Rings'],
      dtype='object')
```

In [10]:

```

sns.heatmap(data[[ 'Length', 'Diameter', 'Height', 'Whole weight', 'Shucked
       'Viscera weight', 'Shell weight', 'Rings']])
```

Out[10]:

```
<AxesSubplot:>
```



In [11]:

```
data['Sex'].value_counts()
```

Out[11]:

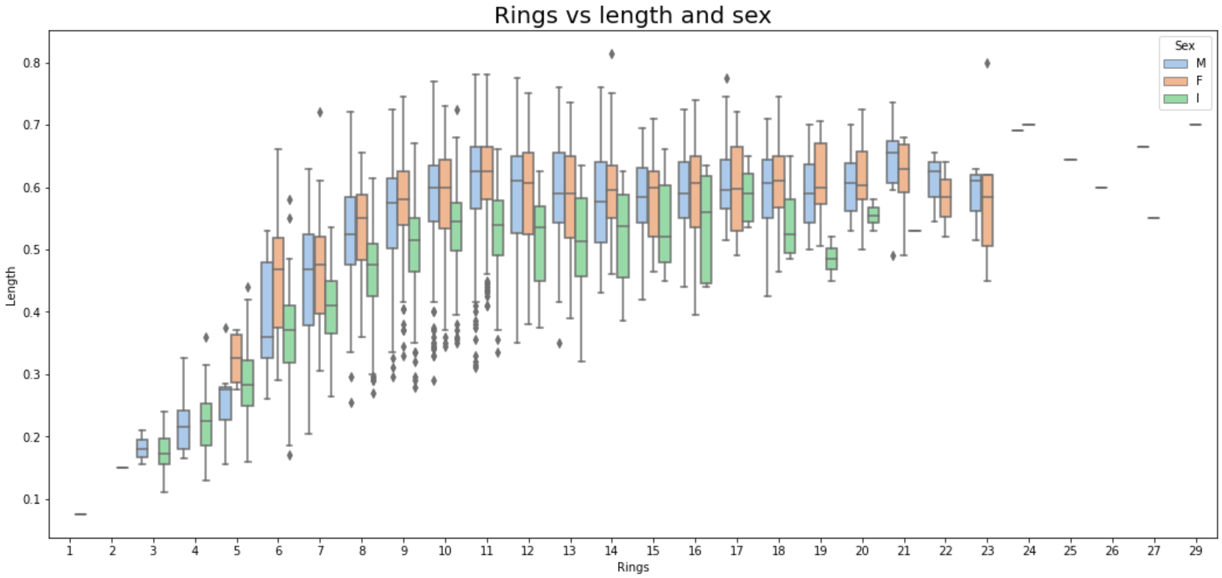
```
M    1528
I    1342
F    1307
Name: Sex, dtype: int64
```

In [12]:

```
plt.rcParams['figure.figsize'] = (18, 8)
sns.boxplot(x=data['Rings'], y=data['Length'], hue = data['Sex'], palette =
plt.title('Rings vs length and sex', fontsize = 20)
```

Out[12]:

```
Text(0.5, 1.0, 'Rings vs length and sex')
```

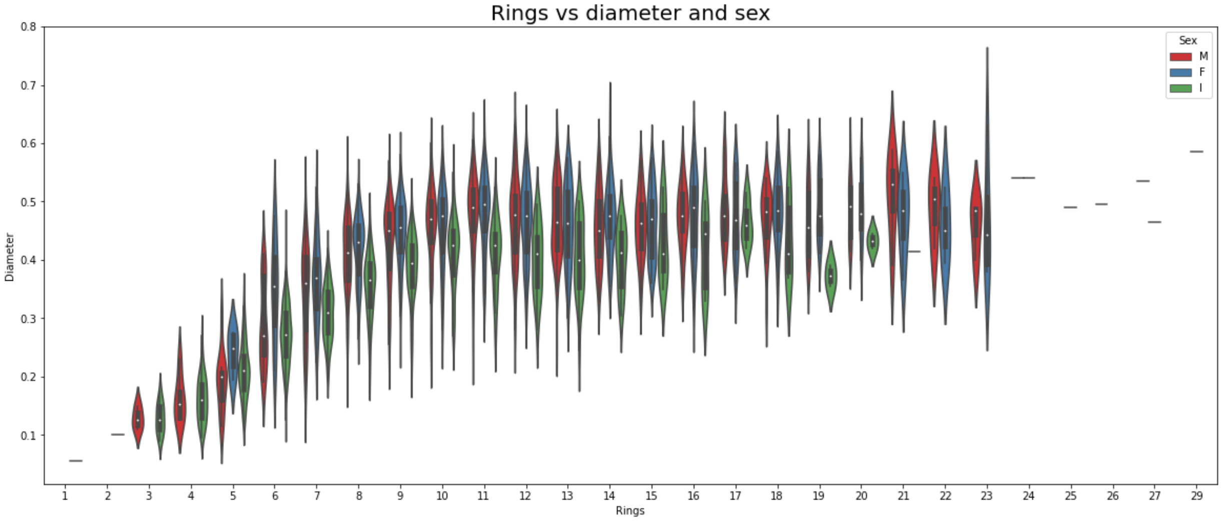


In [13]:

```
plt.rcParams['figure.figsize'] = (20, 8)
sns.violinplot(x=data['Rings'], y=data['Diameter'], hue = data['Sex'], palet
plt.title('Rings vs diameter and sex', fontsize = 20)
```

Out[13]:

```
Text(0.5, 1.0, 'Rings vs diameter and sex')
```

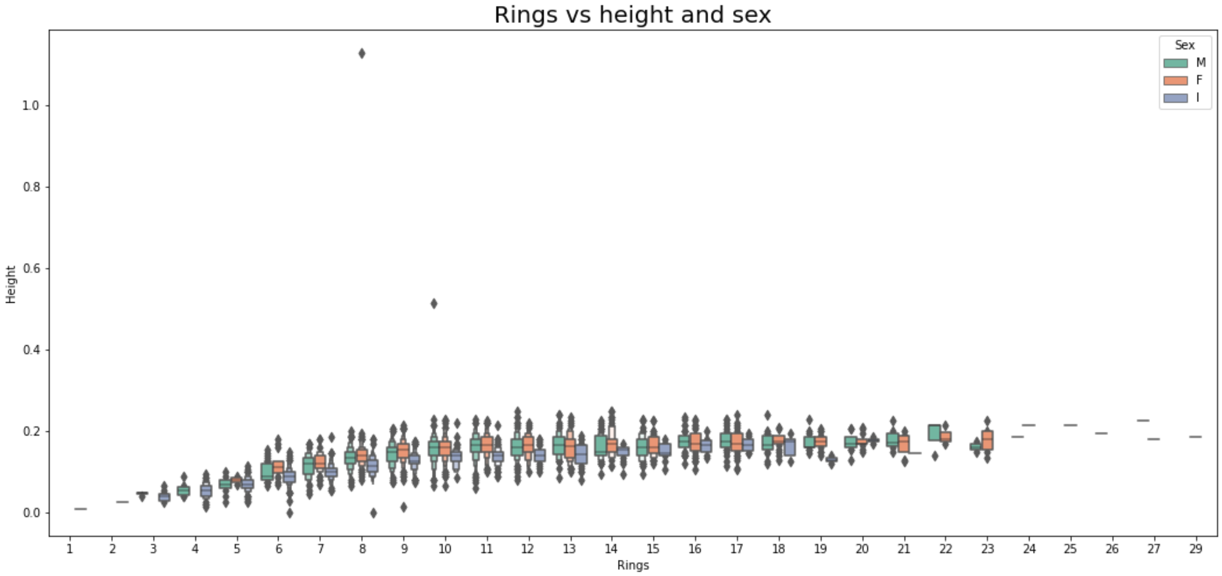


In [14]:

```
plt.rcParams['figure.figsize'] = (18, 8)
sns.boxenplot(x=data['Rings'], y=data['Height'], hue = data['Sex'], palette
plt.title('Rings vs height and sex', fontsize = 20)
```

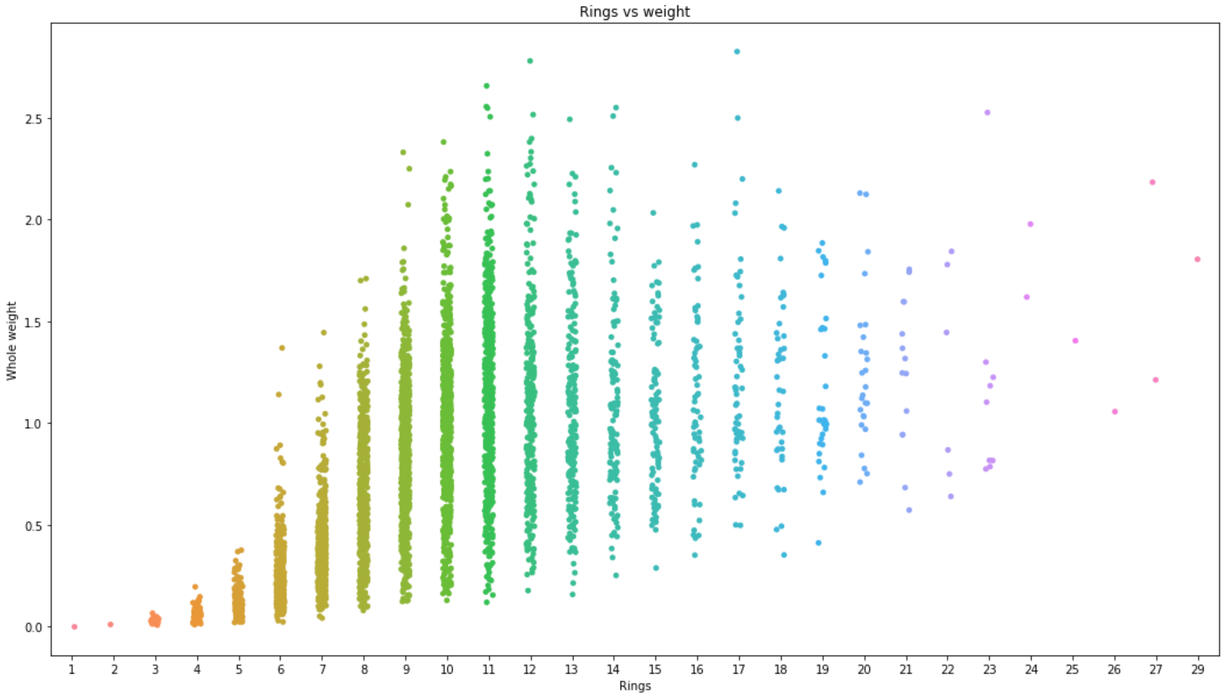
Out[14]:

```
Text(0.5, 1.0, 'Rings vs height and sex')
```



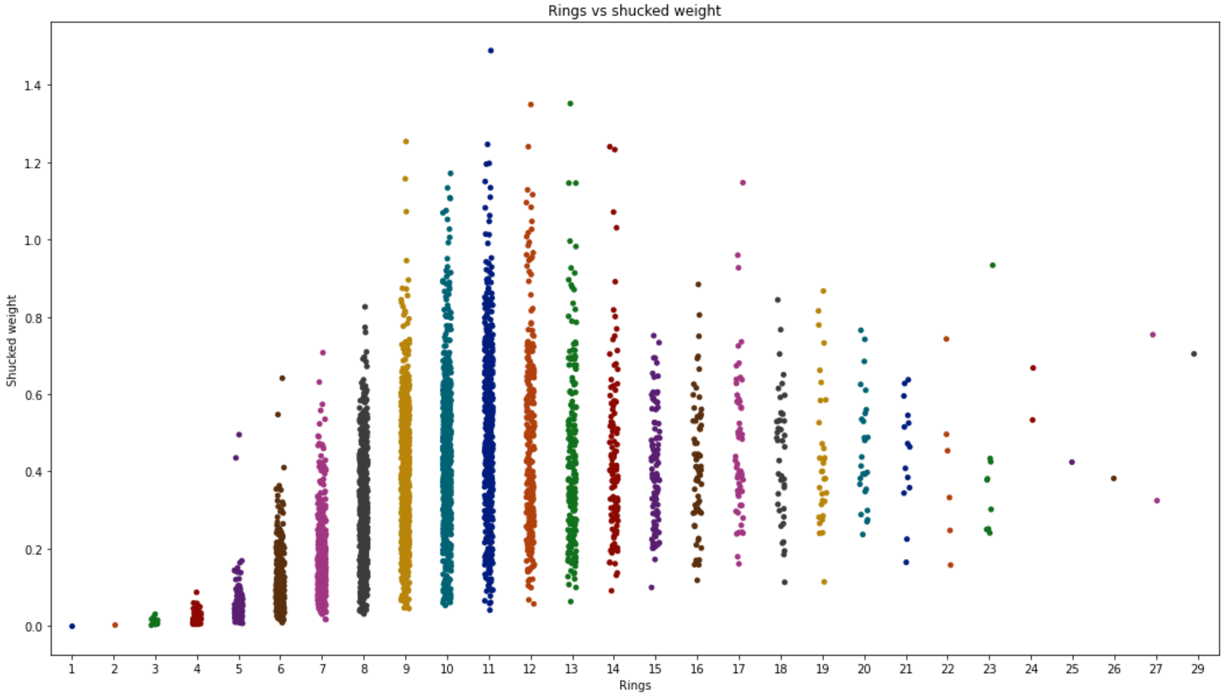
In [15]:
```
plt.rcParams['figure.figsize'] = (18, 10)
sns.stripplot(x=data['Rings'], y=data['Whole weight'])
plt.title('Rings vs weight')
```

Out[15]:
```
Text(0.5, 1.0, 'Rings vs weight')
```



In [16]:
```
plt.rcParams['figure.figsize'] = (18, 10)
sns.stripplot(x=data['Rings'], y=data['Shucked weight'], palette = 'dark')
plt.title('Rings vs shucked weight')
```

Out[16]:
```
Text(0.5, 1.0, 'Rings vs shucked weight')
```

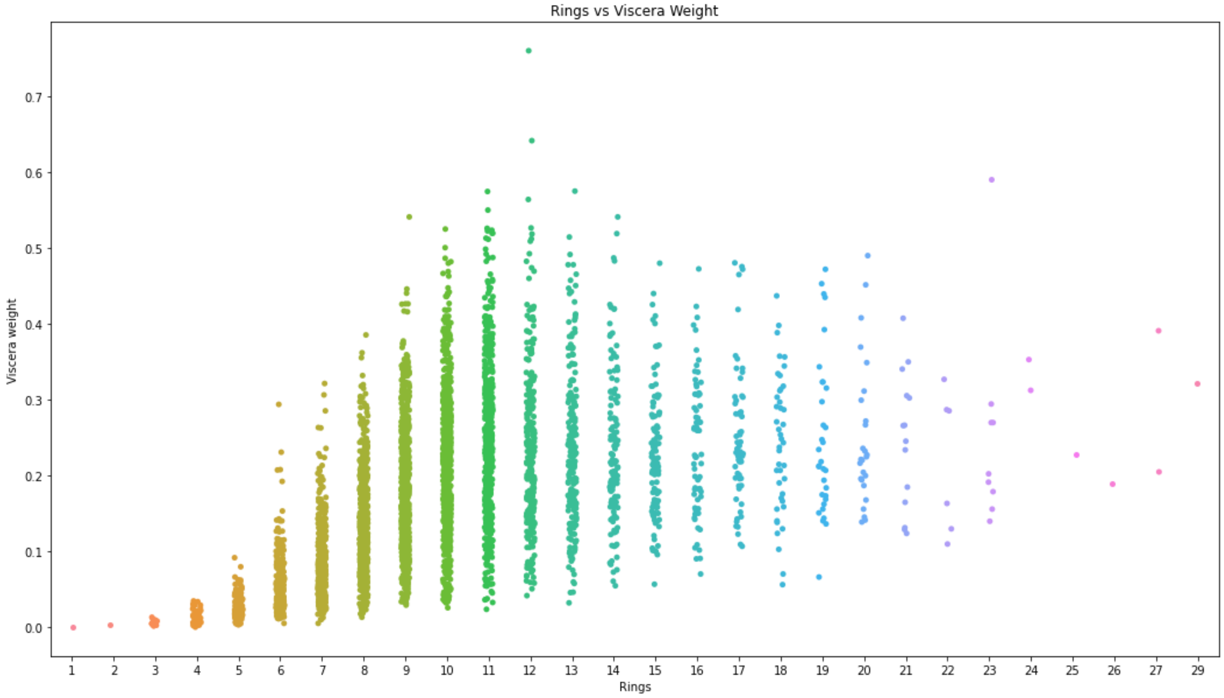


In [17]:

```
plt.rcParams['figure.figsize'] = (18, 10)
sns.stripplot(x=data['Rings'], y=data['Viscera weight'])
plt.title('Rings vs Viscera Weight')
```

Out[17]:

```
Text(0.5, 1.0, 'Rings vs Viscera Weight')
```

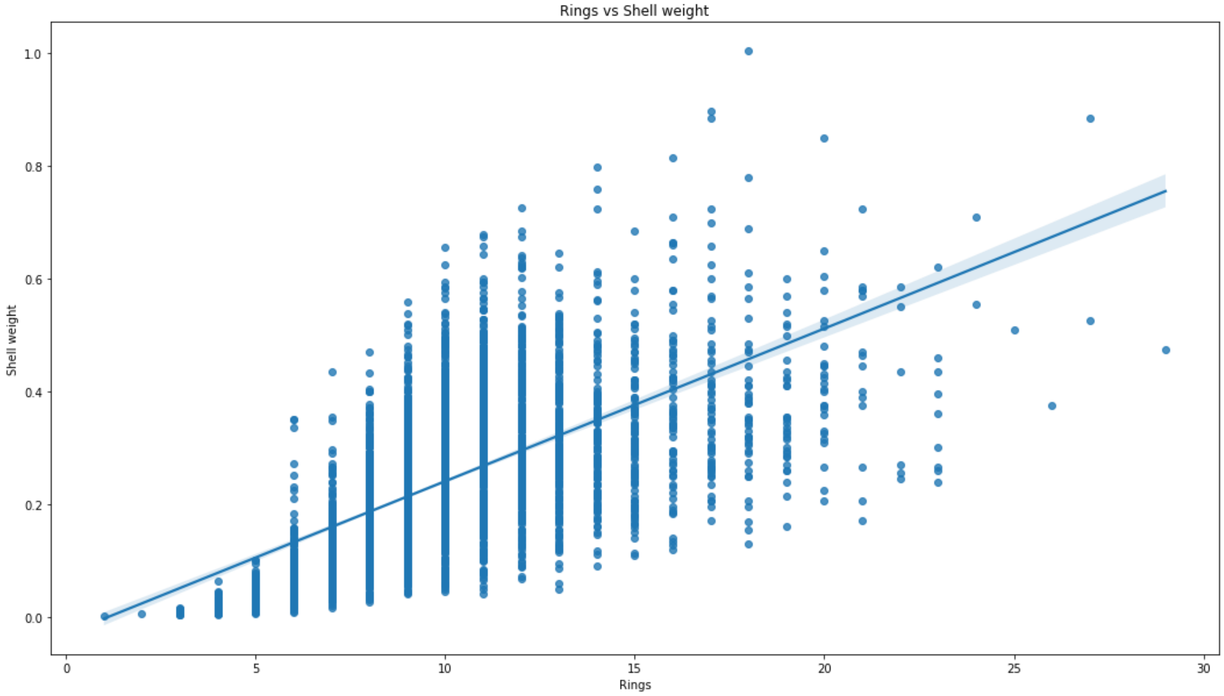


In [18]:

```
plt.rcParams['figure.figsize'] = (18, 10)
sns.regplot(x=data['Rings'], y=data['Shell weight'])
plt.title('Rings vs Shell weight')
```

Out[18]:

```
Text(0.5, 1.0, 'Rings vs Shell weight')
```

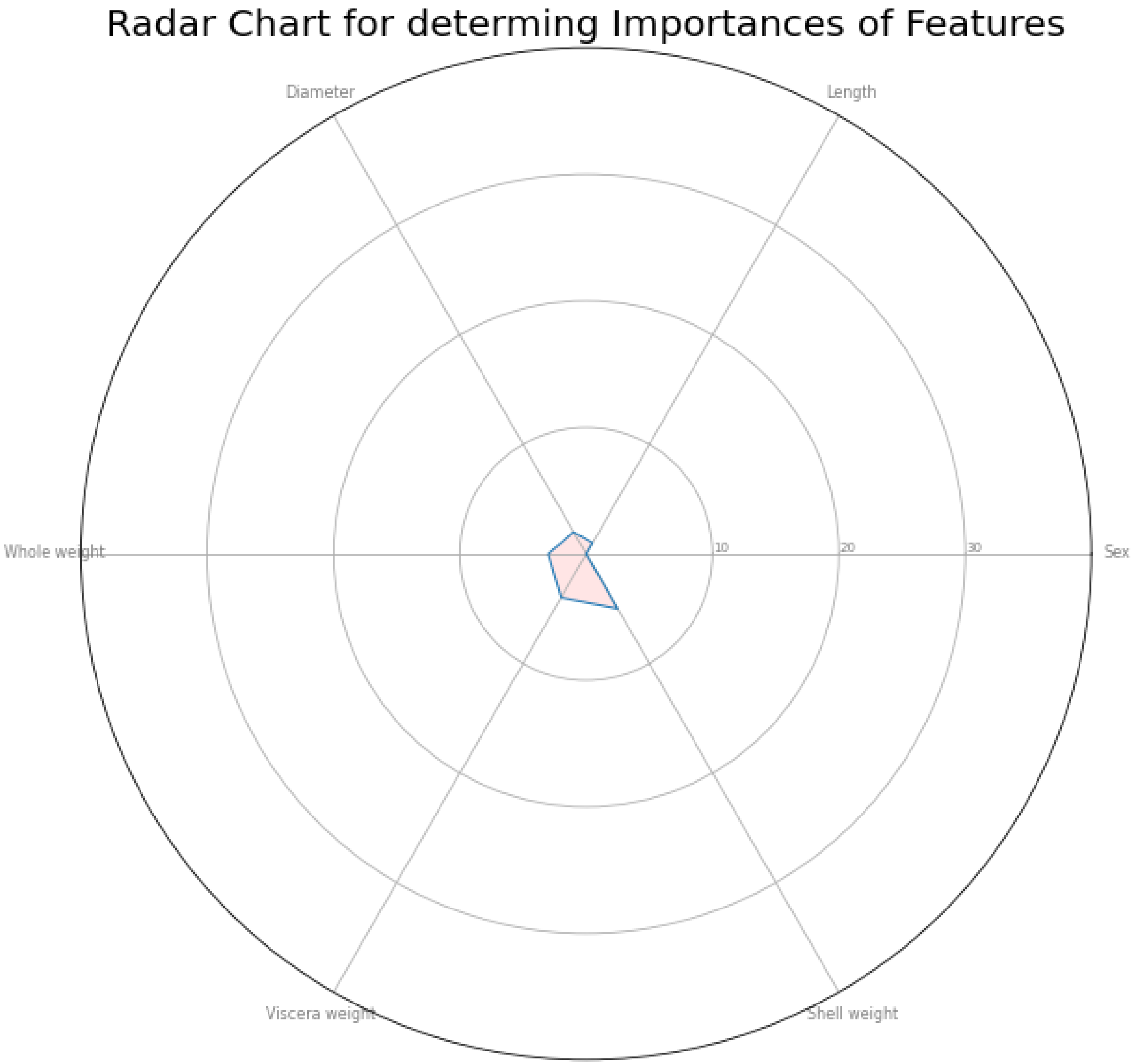


In [19]:

```python
from math import pi

# Set data
df = pd.DataFrame({
'group': [i for i in range(0, 4177)],
'Sex': data['Sex'],
'Length': data['Length'],
'Diameter': data['Diameter'],
'Whole weight':  data['Whole weight'],
'Viscera weight': data['Viscera weight'],
'Shell weight': data['Shell weight']
})

# number of variable
categories=list(df)[1:]
N = len(categories)

# We are going to plot the first line of the data frame.
# But we need to repeat the first value to close the circular graph:
values = df.loc[0].drop('group').values.flatten().tolist()
values += values[:1]
values

# What will be the angle of each axis in the plot? (we divide the plot / num
angles = [n / float(N) * 2 * pi for n in range(N)]
angles += angles[:1]

# Initialise the spider plot
ax = plt.subplot(111, polar=True)

# Draw one axe per variable + add labels labels yet
plt.xticks(angles[:-1], categories, color='grey', size=8)

# Draw ylabels
ax.set_rlabel_position(0)
plt.yticks([10,20,30], ["10","20","30"], color="grey", size=7)
plt.ylim(0,40)

# Plot data
ax.plot(angles, values, linewidth=1, linestyle='solid')
plt.title('Radar Chart for determing Importances of Features', fontsize = 20
# Fill area
ax.fill(angles, values, 'red', alpha=0.1)
```

Out[19]:

```
[<matplotlib.patches.Polygon at 0x24979ee2940>]
```



In [20]:

```
data = pd.get_dummies(data)
```

In [21]:

```python

data.head()
```

Out[21]:

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

In [22]:

```python
y = data['Rings']
data = data.drop(['Rings'], axis = 1)
x = data

# getting the shapes
print("Shape of x:", x.shape)
print("Shape of y:", y.shape)
```

```
Shape of x: (4177, 10)
Shape of y: (4177,)
```

In [23]:

```python
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, r

# getting the shapes
print("Shape of x_train :", x_train.shape)
print("Shape of x_test :", x_test.shape)
print("Shape of y_train :", y_train.shape)
print("Shape of y_test :", y_test.shape)
```

```
Shape of x_train : (3341, 10)
Shape of x_test : (836, 10)
Shape of y_train : (3341,)
Shape of y_test : (836,)
```

In [24]:

```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score

model = RandomForestClassifier()
model.fit(x_train, y_train)
y_pred = model.predict(x_test)

# evaluation
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
print("RMSE :", rmse)

# r2 score
r2 = r2_score(y_test, y_pred)
print("R2 Score :", r2)
```

```
RMSE : 2.5539630523418446
R2 Score : 0.39939775195158833
```

In [25]:

```
!pip install eli5
```

```
Requirement already satisfied: eli5 in c:\users\write\anaconda3\lib\site-packag
es (0.13.0)
Requirement already satisfied: six in c:\users\write\anaconda3\lib\site-package
s (from eli5) (1.16.0)
Requirement already satisfied: scikit-learn>=0.20 in c:\users\write\anaconda3\l
ib\site-packages (from eli5) (1.0.2)
Requirement already satisfied: graphviz in c:\users\write\anaconda3\lib\site-pa
ckages (from eli5) (0.20.1)
Requirement already satisfied: tabulate>=0.7.7 in c:\users\write\anaconda3\lib
\site-packages (from eli5) (0.8.9)
Requirement already satisfied: jinja2>=3.0.0 in c:\users\write\anaconda3\lib\si
te-packages (from eli5) (3.1.2)
Requirement already satisfied: numpy>=1.9.0 in c:\users\write\anaconda3\lib\sit
e-packages (from eli5) (1.21.5)
Requirement already satisfied: scipy in c:\users\write\anaconda3\lib\site-packa
ges (from eli5) (1.7.3)
Requirement already satisfied: attrs>17.1.0 in c:\users\write\anaconda3\lib\sit
e-packages (from eli5) (21.4.0)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\write\anaconda3\lib
\site-packages (from jinja2>=3.0.0->eli5) (2.0.1)
Requirement already satisfied: joblib>=0.11 in c:\users\write\anaconda3\lib\sit
e-packages (from scikit-learn>=0.20->eli5) (1.1.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\write\anaconda3
\lib\site-packages (from scikit-learn>=0.20->eli5) (2.2.0)
```

In [26]:

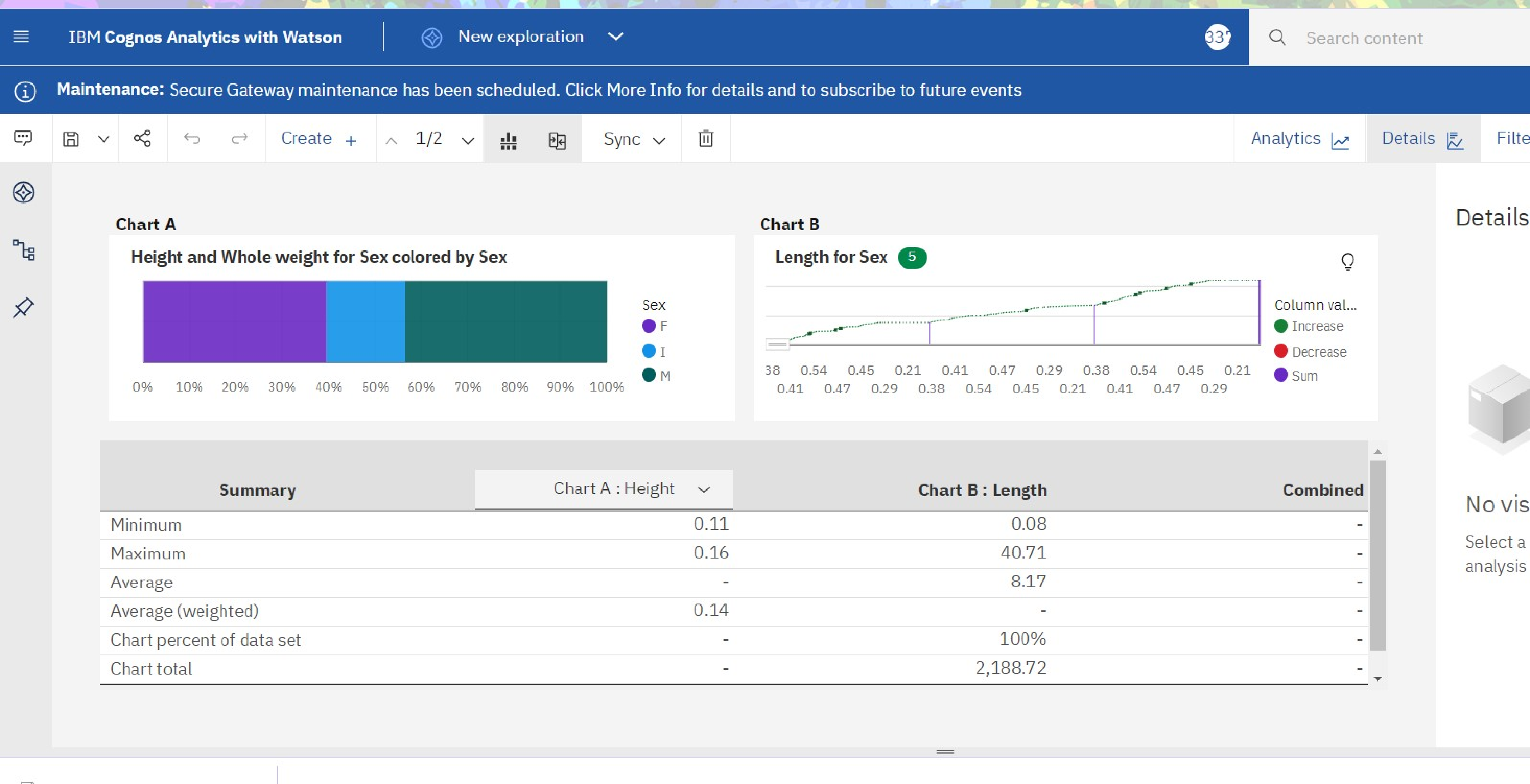
```
import eli5
from eli5.sklearn import PermutationImportance

perm = PermutationImportance(model, random_state = 0).fit(x_test, y_test)
eli5.show_weights(perm, feature_names = x_test.columns.tolist())
```

Out[26]:

Weight	Feature
0.0388 ± 0.0399	Shell weight
0.0297 ± 0.0231	Shucked weight
0.0172 ± 0.0110	Length
0.0160 ± 0.0113	Viscera weight
0.0084 ± 0.0062	Height
0.0072 ± 0.0086	Sex_I
-0.0014 ± 0.0111	Sex_F
-0.0019 ± 0.0135	Whole weight
-0.0048 ± 0.0079	Sex_M
-0.0077 ± 0.0120	Diameter

In [ ]:

```

```



Maintenance: Secure Gateway maintenance has been scheduled. Click More Info for details and to subscribe to future events

Summary	Chart A : Height	Chart B : Length	Combined
Minimum	0.11	0.08	-
Maximum	0.16	40.71	-
Average	-	8.17	-
Average (weighted)	0.14	-	-
Chart percent of data set	-	100%	-
Chart total	-	2,188.72	-