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

```
df = pd.read_csv('abalone.csv')
df.head()
```

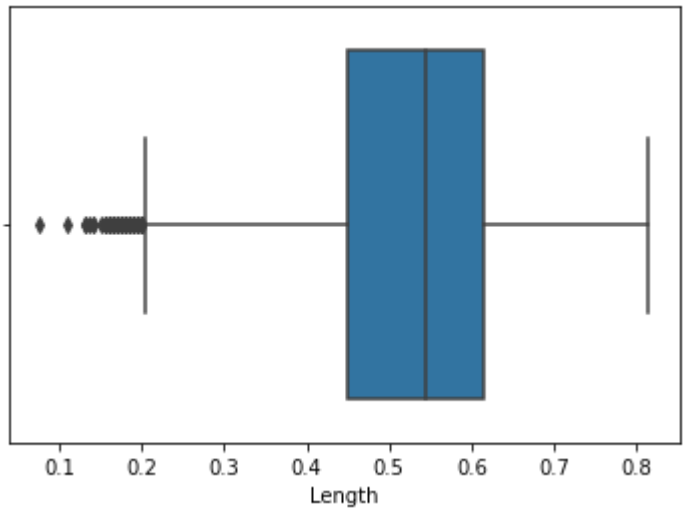
	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shel
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	

Univariate Analysis

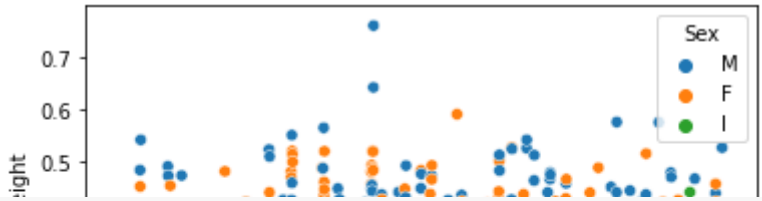
```
import matplotlib.pyplot as plt
import seaborn as sns
```

```
sns.boxplot(x='Length',data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f886dee5810>
```

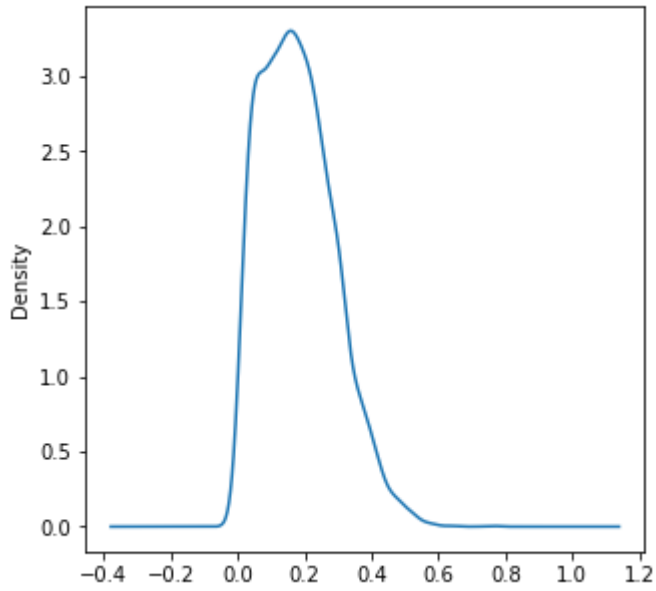



```
sns.scatterplot(x=df.index,y=df['Viscera weight'],hue=df['Sex']);
```




```
plt.figure(figsize=(5,5))
df['Viscera weight'].plot(kind='density')
```

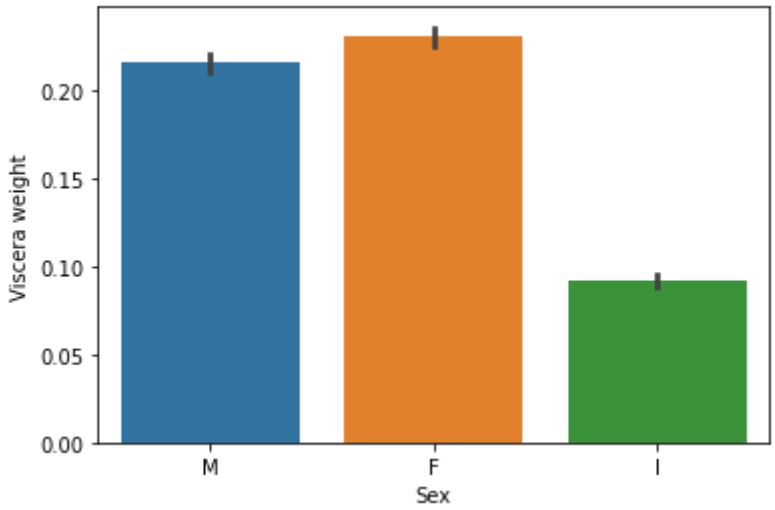
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8857780ad0>
```



Bi-Variate Analysis

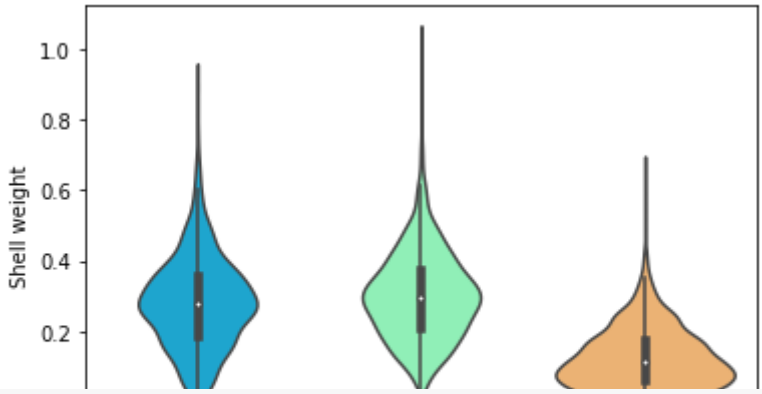
```
sns.barplot(x='Sex',y='Viscera weight',data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8855ef6d10>
```



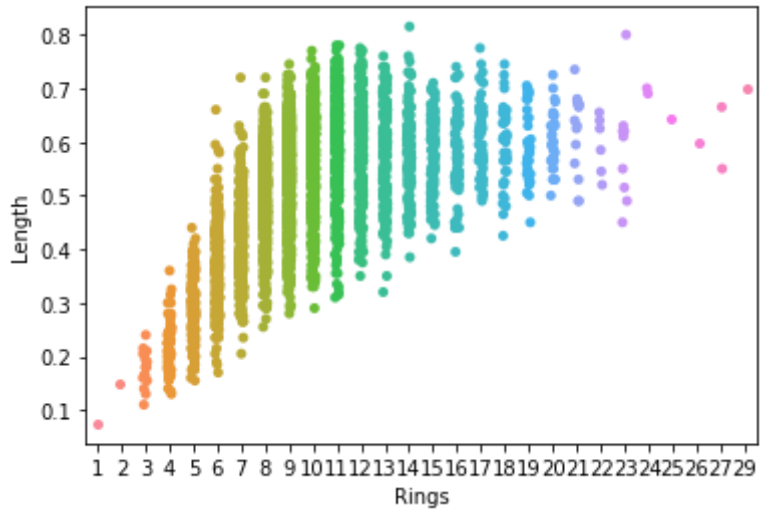

```
sns.violinplot(x="Sex", y="Shell weight", data=df,palette='rainbow')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8855e6d650>
```



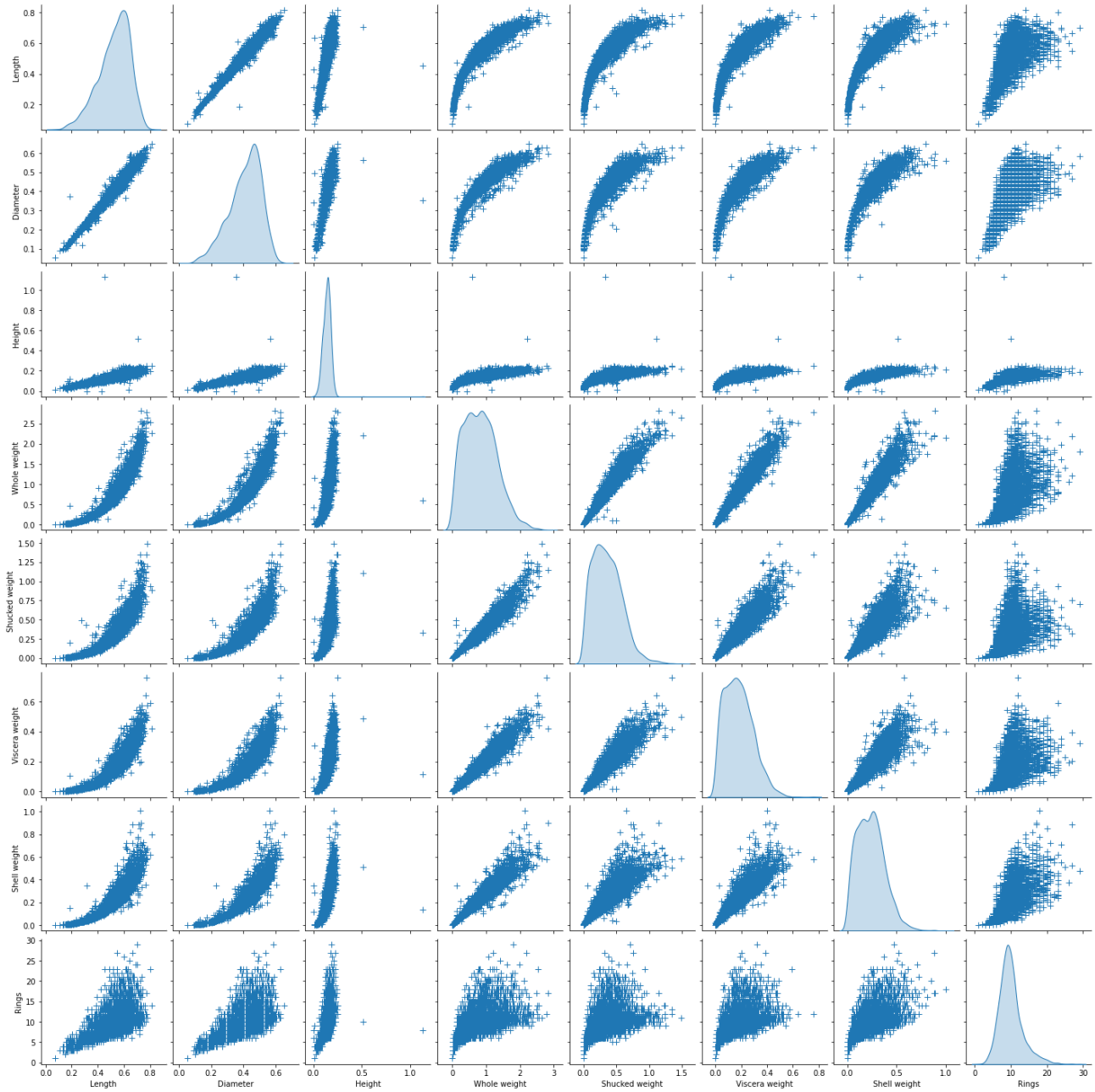

```
sns.stripplot(x="Rings", y="Length", data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8855eeba10>
```

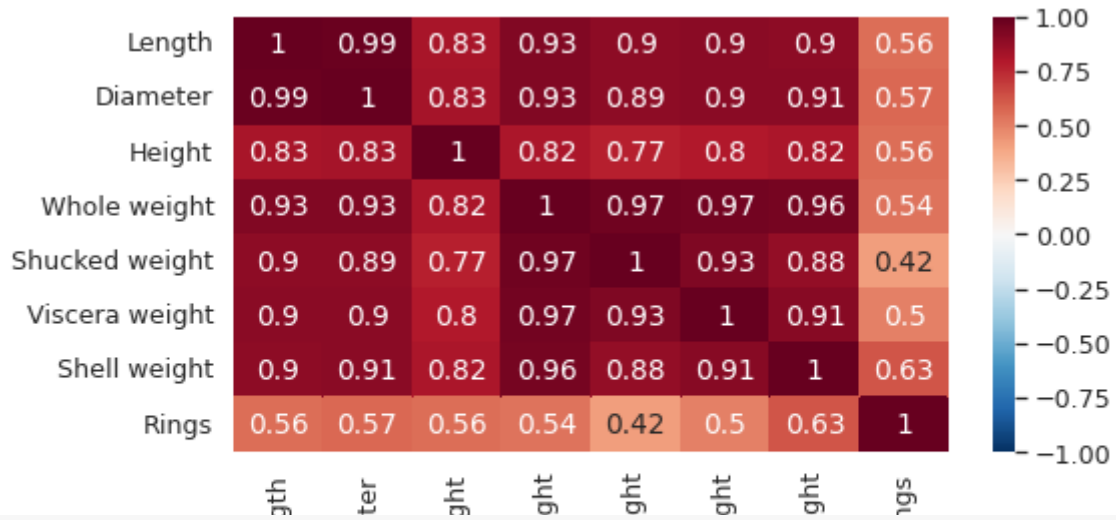


Multi-Variate Analysis

```
graph = sns.pairplot(df, diag_kind="kde", markers="+",
                     plot_kws=dict(s=50, edgecolor="r", linewidth=1),
                     diag_kws=dict(shade=True))
```




```
sns.set(font_scale=1.15)
plt.figure(figsize=(8,4))
sns.heatmap(
    df.corr(),
    cmap='RdBu_r',
    annot=True,
    vmin=-1, vmax=1);
```




```
df.describe()
```

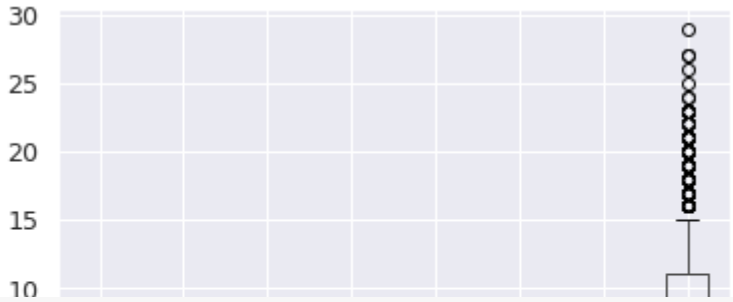
	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	
count	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4
mean	0.523992	0.407881	0.139516	0.828742	0.359367	0.180594	
std	0.120093	0.099240	0.041827	0.490389	0.221963	0.109614	
min	0.075000	0.055000	0.000000	0.002000	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.isnull().sum() #no missing values
```

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

```
df.boxplot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8851b59d90>
```




```
h = df[df['Rings'] < 10]
h
```

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.0700	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.2100	9
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.0550	7
5	I	0.425	0.300	0.095	0.3515	0.1410	0.0775	0.1200	8
8	M	0.475	0.370	0.125	0.5095	0.2165	0.1125	0.1650	9
...	...	...	...	...	...	...	...	...	...
4165	I	0.405	0.300	0.085	0.3035	0.1500	0.0505	0.0880	7
4167	M	0.500	0.380	0.125	0.5770	0.2690	0.1265	0.1535	9
4168	F	0.515	0.400	0.125	0.6150	0.2865	0.1230	0.1765	8
4171	M	0.560	0.430	0.155	0.8675	0.4000	0.1720	0.2290	8
4174	M	0.600	0.475	0.205	1.1760	0.5255	0.2875	0.3080	9

2096 rows × 9 columns

```
col = ['Height','Diameter','Whole_weight','Shucked_weight','Viscera_weight','Rings','Shell
cat_col = df['Sex']
```

```
df['Sex'].replace(['M','F','I'],[0, 1,2], inplace=True)
df.head(5)
```

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

```
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_score, confusion_matrix
from sklearn import metrics
import numpy as np
```

```
convert = StandardScaler()

feat = df.drop(['Sex', 'Rings'], axis = 1)
label = df.Rings

feat = convert.fit_transform(feat)
```

```
f_train, f_test, l_train, l_test = train_test_split(feat, label, random_state = 23, test_s
```

```
model1 = LogisticRegression(random_state=23)
model1.fit(f_train, l_train)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:818: Conver
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG,
LogisticRegression(random_state=23)
```

```
y_predict = model1.predict(f_train)
```

```
accuracy_score(l_train, y_predict)
```

```
0.29003292427416943
```

Colab paid products - Cancel contracts here

✓ 0s completed at 2:17 PM