```
import numpy as np # linear algebra
import pandas as pd
```

```
iris_data = pd.read_csv('/Iris.csv')
```

```
def missing_value_describe(data):
    # check missing values in training data
    missing_value_stats = (data.isnull().sum() / len(data)*100)
    missing_value_col_count = sum(missing_value_stats > 0)
    missing_value_stats = missing_value_stats.sort_values(ascending=False)[:missing_value_col_count]
    print("Number of columns with missing values:", missing_value_col_count)
    if missing_value_col_count != 0:
        # print out column names with missing value percentage
        print("\nMissing percentage (desceding):")
        print(missing_value_stats)
    else:
        print("No misisng data!!!")
missing_value_describe(iris_data)
```

```
Number of columns with missing values: 0
No misisng data!!!
```

```
iris_data.head()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

Next steps: Generate code with iris_data | View recommended plots | New interactive sheet

```
iris_data = iris_data.drop(['Id'], axis=1)
iris_data.columns
```

```
Index(['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
       'Species'],
      dtype='object')
```

```
print("the dimension:", iris_data.shape)
```

```
the dimension: (150, 5)
```

```
print(iris_data.describe())
```

```
       SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
count     150.000000    150.000000     150.000000    150.000000
mean        5.843333      3.054000       3.758667      1.198667
std         0.828066      0.433594       1.764420      0.763161
min         4.300000      2.000000       1.000000      0.100000
25%         5.100000      2.800000       1.600000      0.300000
50%         5.800000      3.000000       4.350000      1.300000
75%         6.400000      3.300000       5.100000      1.800000
max         7.900000      4.400000       6.900000      2.500000
```

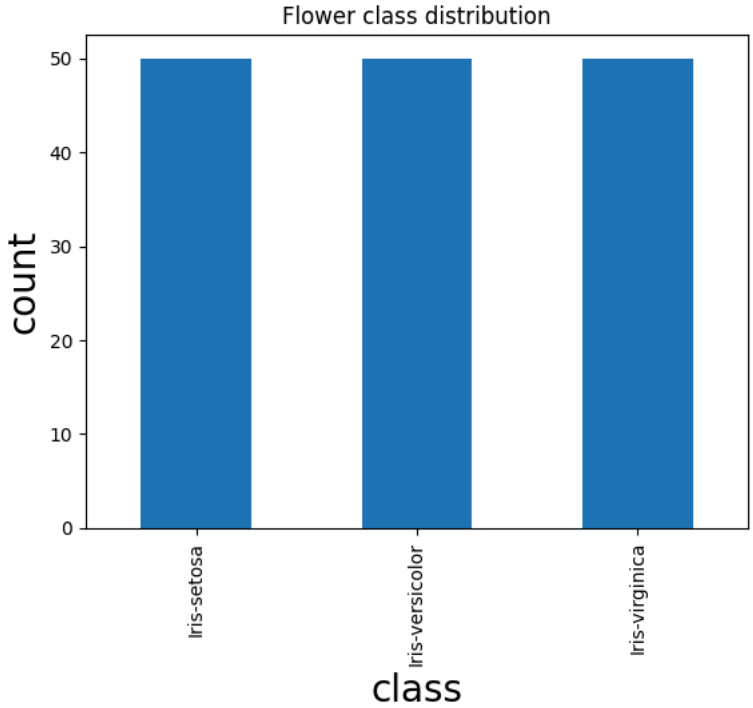
```
# class distribution
print(iris_data.groupby('Species').size())
```

```
Species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
dtype: int64
```

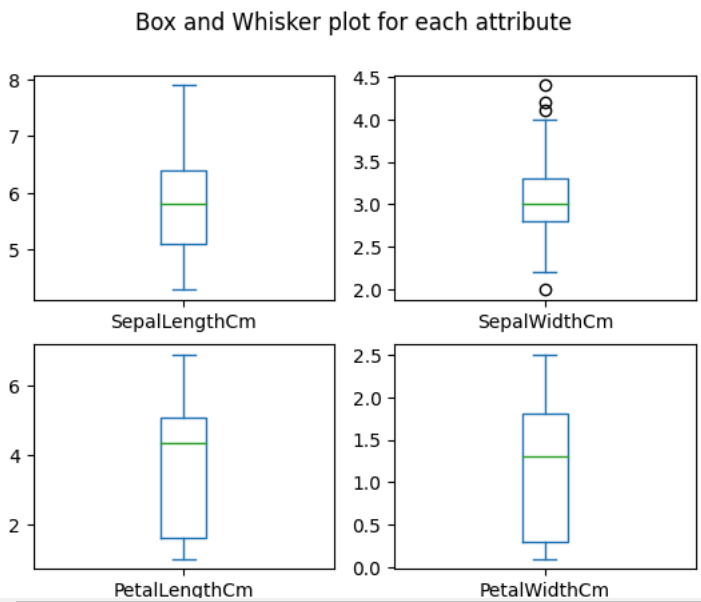
```
import matplotlib.pyplot as plt
```

```
nameplot = iris_data['Species'].value_counts().plot.bar(title='Flower class distribution')
nameplot.set_xlabel('class',size=20)
nameplot.set_ylabel('count',size=20)
```

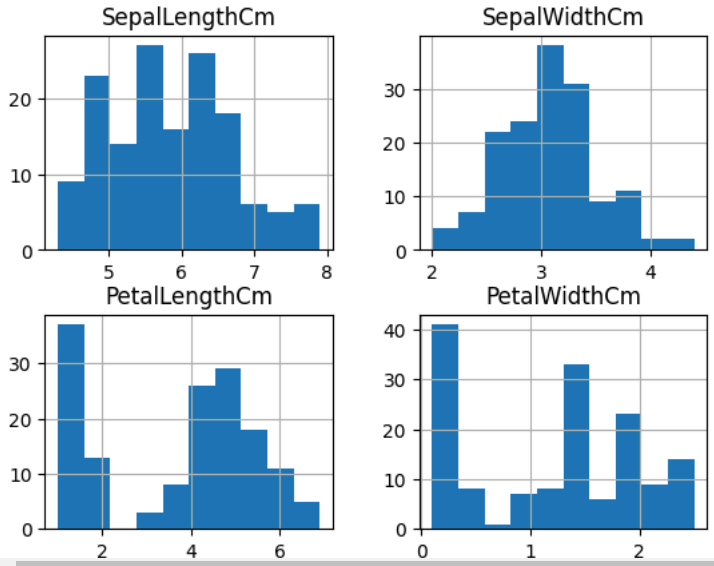
```
Text(0, 0.5, 'count')
```




```
iris_data.plot(kind='box', subplots=True, layout=(2,2),
               sharex=False, sharey=False, title="Box and Whisker plot for each attribute")
plt.show()
```

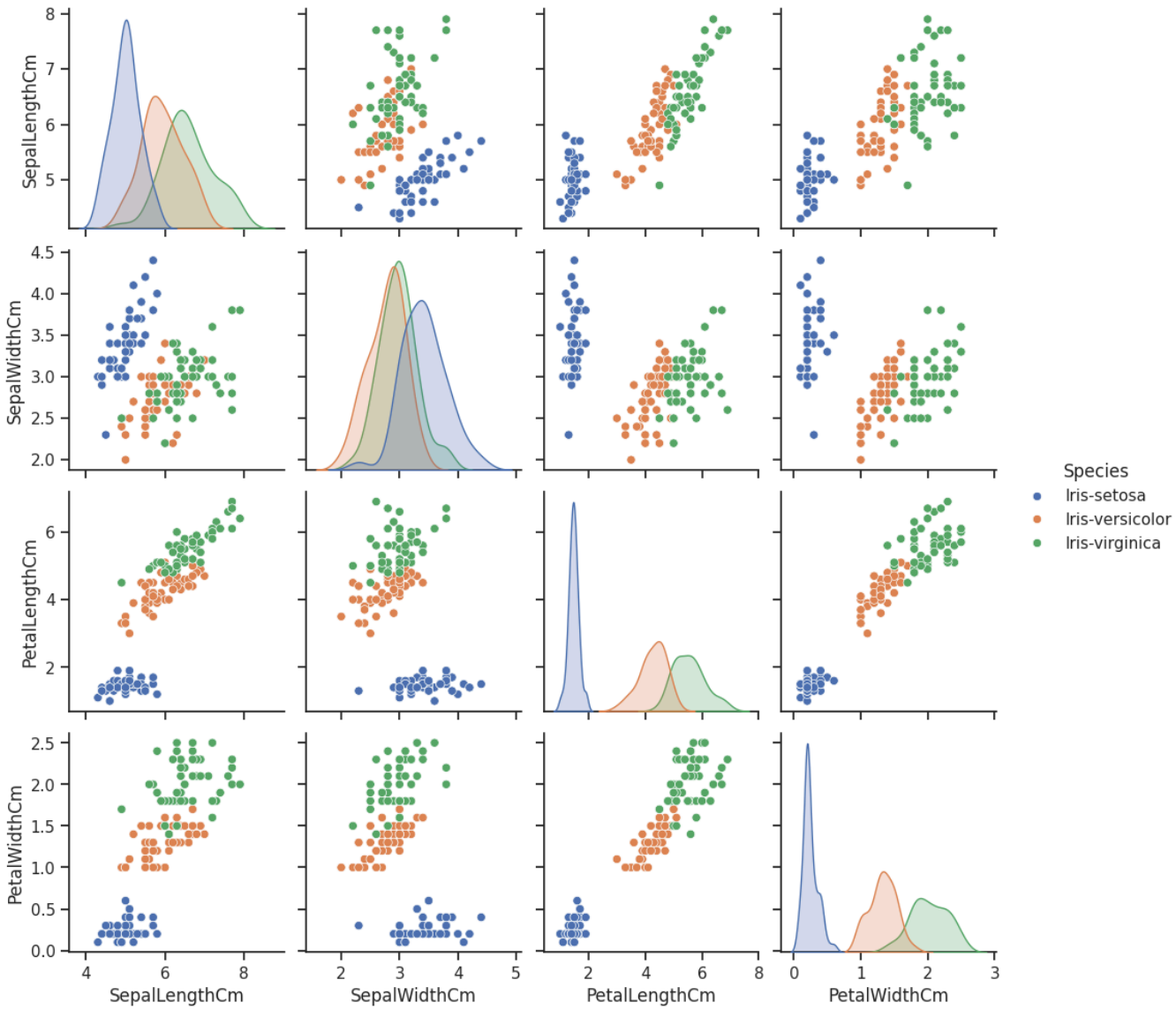



```
iris_data.hist()
plt.show()
```




```
import seaborn as sns
sns.set(style="ticks")
sns.pairplot(iris_data, hue="Species")
```

```
<seaborn.axisgrid.PairGrid at 0x7fb6037db970>
```



```
from sklearn.model_selection import train_test_split
```

```
X = iris_data.drop(['Species'], axis=1)
Y = iris_data['Species']
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=7)
```

```
print("X_train.shape:", X_train.shape)
print("X_test.shape:", X_test.shape)
print("Y_train.shape:", X_train.shape)
print("Y_test.shape:", Y_test.shape)
```

```
X_train.shape: (120, 4)
X_test.shape: (30, 4)
Y_train.shape: (120, 4)
Y_test.shape: (30,)
```