In [3]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
iris = pd.read_csv(r'C:\Users\chand\OneDrive\Documents\Desktop\Iris.csv')
```

In [4]:
```
print(iris.head())
```

```
   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Speci
es
0   1            5.1           3.5            1.4           0.2  Iris-seto
sa
1   2            4.9           3.0            1.4           0.2  Iris-seto
sa
2   3            4.7           3.2            1.3           0.2  Iris-seto
sa
3   4            4.6           3.1            1.5           0.2  Iris-seto
sa
4   5            5.0           3.6            1.4           0.2  Iris-seto
sa
```

In [5]:
```
print(iris.describe())
```

```
               Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthC
m
count  150.000000     150.000000    150.000000     150.000000    150.00000
0
mean    75.500000       5.843333      3.054000       3.758667      1.19866
7
std     43.445368       0.828066      0.433594       1.764420      0.76316
1
min      1.000000       4.300000      2.000000       1.000000      0.10000
0
25%     38.250000       5.100000      2.800000       1.600000      0.30000
0
50%     75.500000       5.800000      3.000000       4.350000      1.30000
0
75%    112.750000       6.400000      3.300000       5.100000      1.80000
0
max    150.000000       7.900000      4.400000       6.900000      2.50000
0
```

In [21]:
```
print(iris.isna().sum())
print(iris.describe())
```

```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
                Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthC
m
count  150.000000     150.000000    150.000000     150.000000    150.00000
0
mean    75.500000       5.843333      3.054000       3.758667      1.19866
7
std     43.445368       0.828066      0.433594       1.764420      0.76316
1
min      1.000000       4.300000      2.000000       1.000000      0.10000
0
25%     38.250000       5.100000      2.800000       1.600000      0.30000
0
50%     75.500000       5.800000      3.000000       4.350000      1.30000
0
75%    112.750000       6.400000      3.300000       5.100000      1.80000
0
max    150.000000       7.900000      4.400000       6.900000      2.50000
0
```

In [22]:
```
iris.head()
```

Out[22]:

	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

In [23]:
```
iris.head(150)
```

Out[23]:

	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
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

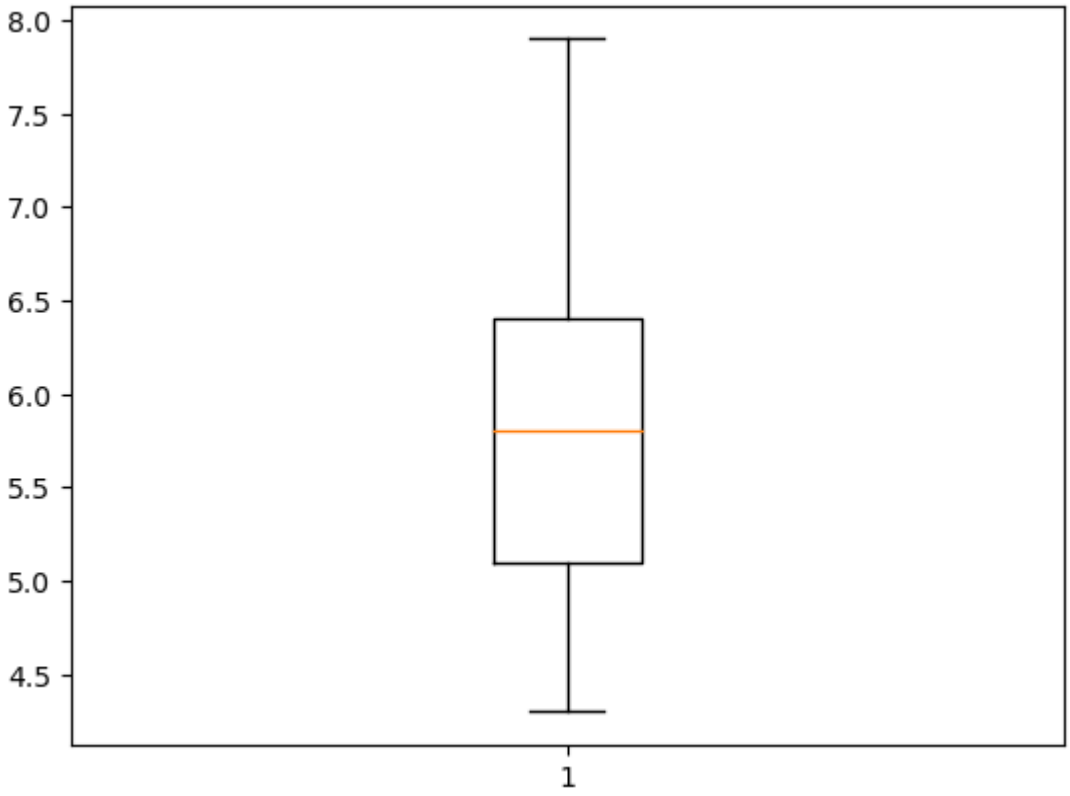
In [24]:
```
iris.tail(100)
```

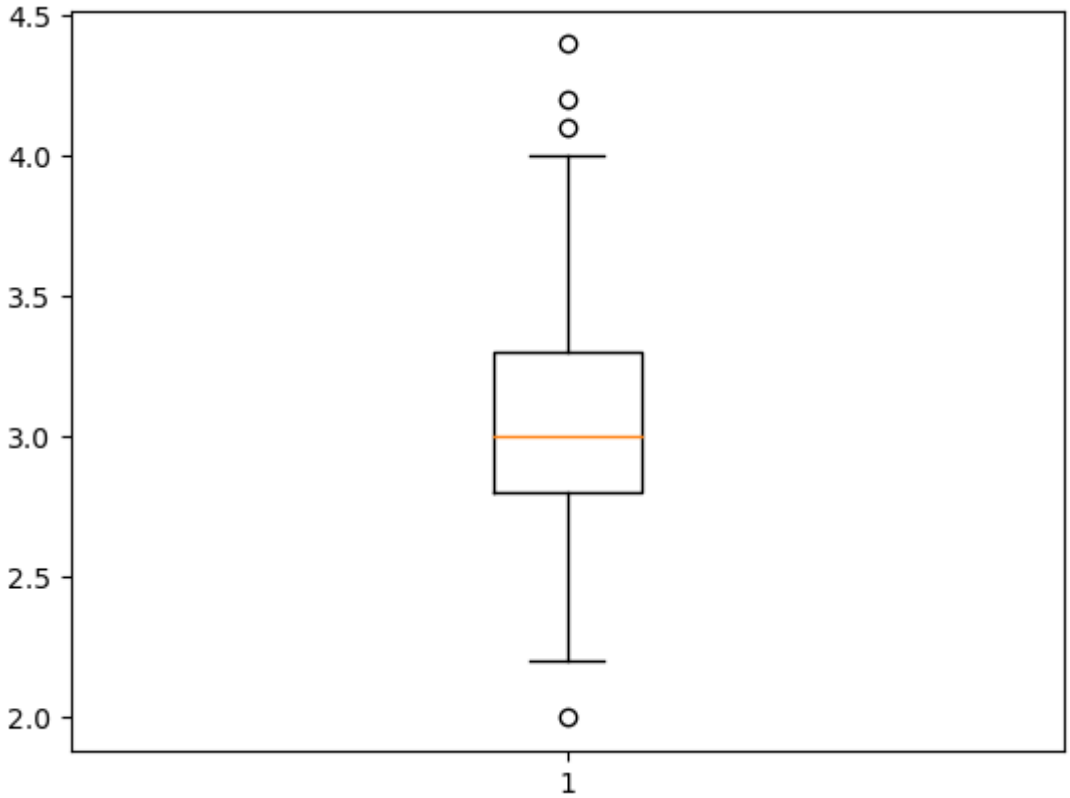
Out[24]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
50	51	7.0	3.2	4.7	1.4	Iris-versicolor
51	52	6.4	3.2	4.5	1.5	Iris-versicolor
52	53	6.9	3.1	4.9	1.5	Iris-versicolor
53	54	5.5	2.3	4.0	1.3	Iris-versicolor
54	55	6.5	2.8	4.6	1.5	Iris-versicolor
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

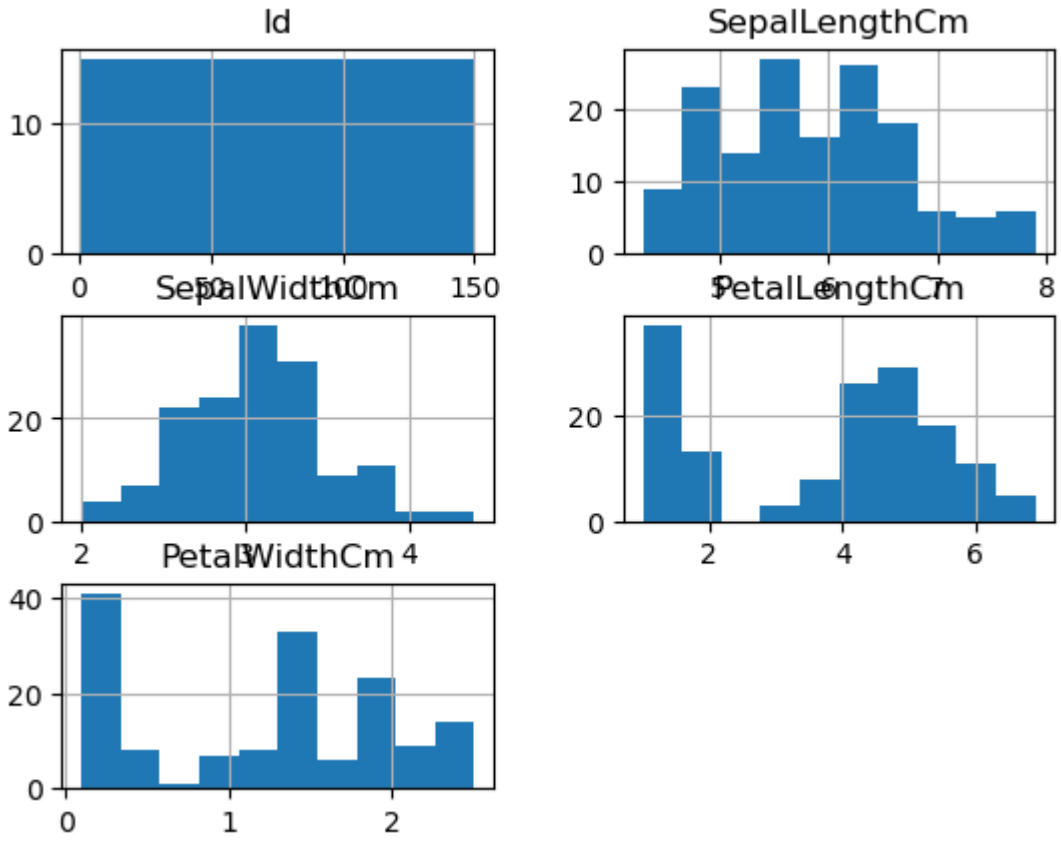
100 rows × 6 columns

In [27]:
```
import matplotlib.pyplot as plt
plt.figure(1)
plt.boxplot([iris['SepalLengthCm']])
plt.figure(2)
plt.boxplot([iris['SepalWidthCm']])
plt.show()
```



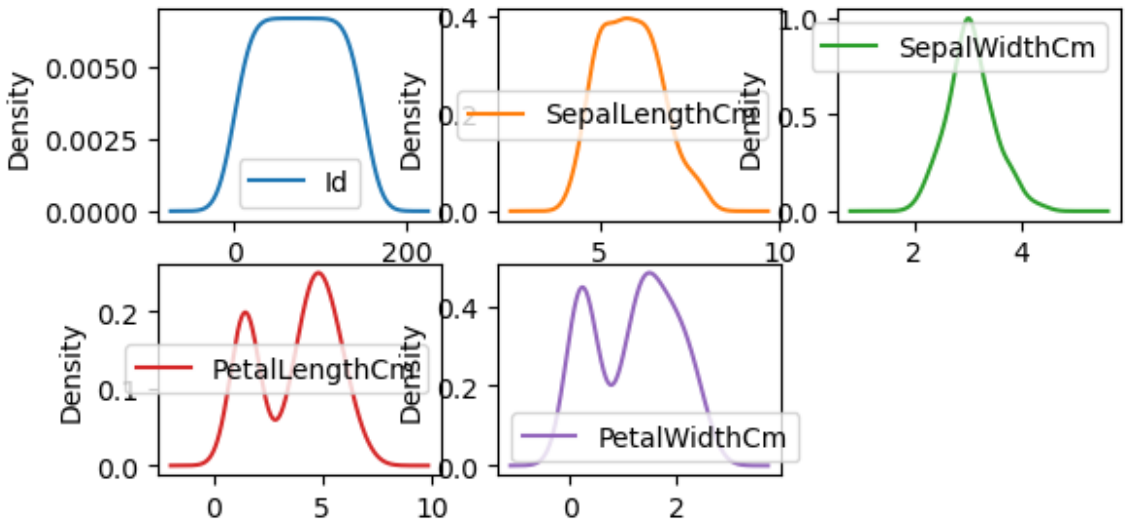


In [30]:
```
iris.hist()
plt.show()
```



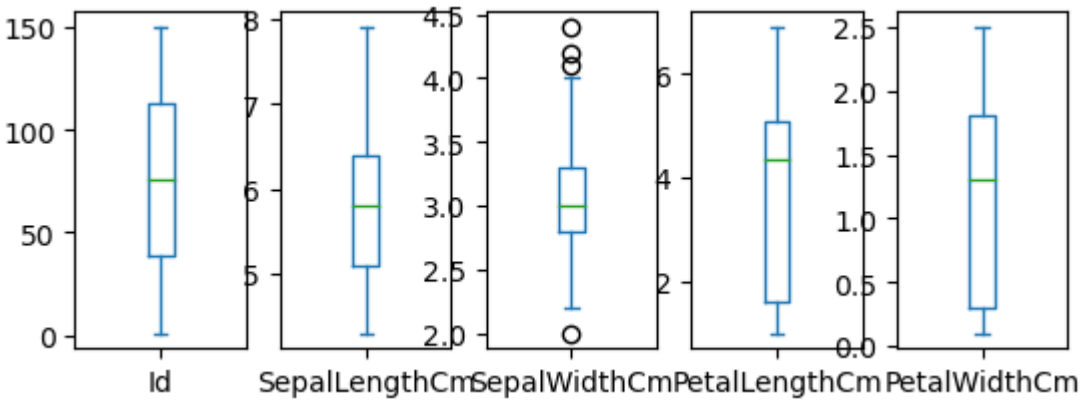
In [31]:
```
iris.plot(kind ='density',subplots = True, layout =(3,3),sharex = False)
```

Out[31]:
```
array([[<Axes: ylabel='Density'>, <Axes: ylabel='Density'>,
        <Axes: ylabel='Density'>],
       [<Axes: ylabel='Density'>, <Axes: ylabel='Density'>,
        <Axes: ylabel='Density'>],
       [<Axes: ylabel='Density'>, <Axes: ylabel='Density'>,
        <Axes: ylabel='Density'>]], dtype=object)
```



In [32]:
```
iris.plot(kind ='box',subplots = True, layout =(2,5),sharex = False)
```

Out[32]:
```
Id                Axes(0.125,0.53;0.133621x0.35)
SepalLengthCm    Axes(0.285345,0.53;0.133621x0.35)
SepalWidthCm      Axes(0.44569,0.53;0.133621x0.35)
PetalLengthCm    Axes(0.606034,0.53;0.133621x0.35)
PetalWidthCm     Axes(0.766379,0.53;0.133621x0.35)
dtype: object
```



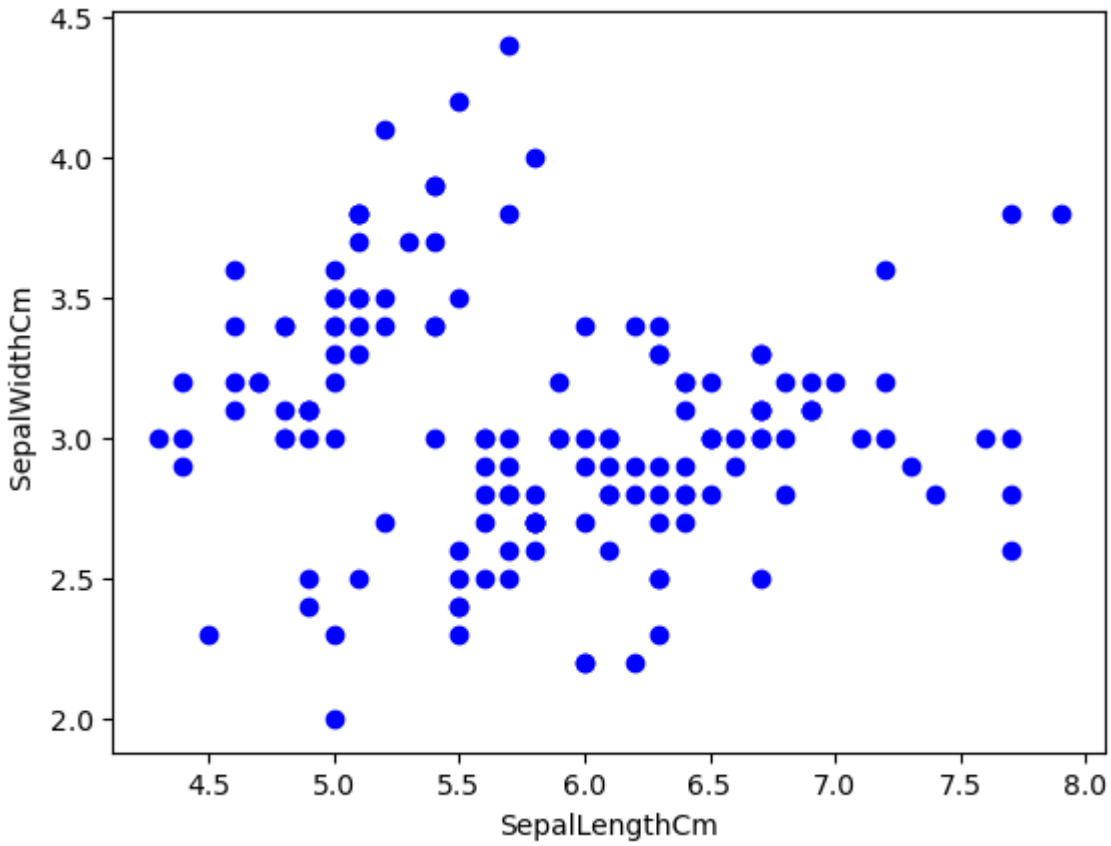
In [34]:
```
X = iris['SepalLengthCm'].values.reshape(-1,1)
print(X)
```

```
[[5.1]
 [4.9]
 [4.7]
 [4.6]
 [5. ]
 [5.4]
 [4.6]
 [5. ]
 [4.4]
 [4.9]
 [5.4]
 [4.8]
 [4.8]
 [4.3]
 [5.8]
 [5.7]
 [5.4]
 [5.1]
 [5.7]
```

In [36]:
```
Y = iris['SepalWidthCm'].values.reshape(-1,1)
print(Y)
```

```
[[3.5]
 [3. ]
 [3.2]
 [3.1]
 [3.6]
 [3.9]
 [3.4]
 [3.4]
 [2.9]
 [3.1]
 [3.7]
 [3.4]
 [3. ]
 [3. ]
 [4. ]
 [4.4]
 [3.9]
 [3.5]
 [3.8]
```

In [37]:
```
plt.xlabel("SepalLengthCm")
plt.ylabel("SepalWidthCm ")
plt.scatter(X,Y,color='b')
plt.show()
```



In [39]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn import svm
from sklearn import metrics
from sklearn.tree import DecisionTreeClassifier
```

In [40]:
```
train, test = train_test_split(iris, test_size = 0.25)
print(train.shape)
print(test.shape)
```

```
(112, 6)
(38, 6)
```

In [41]:
```
train_X = train[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm',
                 'PetalWidthCm']]
train_y = train.Species

test_X = test[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm',
                 'PetalWidthCm']]
test_y = test.Species
```

In [42]:
```
train_X.head()
```

Out[42]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
129	7.2	3.0	5.8	1.6
131	7.9	3.8	6.4	2.0
108	6.7	2.5	5.8	1.8
94	5.6	2.7	4.2	1.3
33	5.5	4.2	1.4	0.2

In [43]:
```
test_y.head()
```

Out[43]:
```
9          Iris-setosa
92     Iris-versicolor
109     Iris-virginica
10         Iris-setosa
119     Iris-virginica
Name: Species, dtype: object
```

In [46]:
```
model = LogisticRegression()
model.fit(train_X, train_y)
prediction = model.predict(test_X)
print('Accuracy:',metrics.accuracy_score(prediction,test_y))
```

```
Accuracy: 0.9736842105263158

C:\ProgramData\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.
py:460: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown i
n:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sc
ikit-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-reg
ression (https://scikit-learn.org/stable/modules/linear_model.html#logisti
c-regression)
  n_iter_i = _check_optimize_result(
```