In [1]:

```
#SHEETAL KHAROLIWAL
import pandas
print('pandas version is: {}'.format(pandas.__version__))
import numpy
print('numpy version is: {}'.format(numpy.__version__))
import seaborn as sns
import sklearn
import matplotlib.pyplot as plt
%matplotlib inline
```

```
pandas version is: 1.1.3
numpy version is: 1.19.2
```

In [2]:

```
import pandas as pd
iris=pd.read_csv('flower.csv')
```

In [3]:

```
iris.head(10)
```

Out[3]:

	sepallength	sepalwidth	petallength	petalwidth	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
5	5.4	3.9	1.7	0.4	Iris-setosa
6	4.6	3.4	1.4	0.3	Iris-setosa
7	5.0	3.4	1.5	0.2	Iris-setosa
8	4.4	2.9	1.4	0.2	Iris-setosa
9	4.9	3.1	1.5	0.1	Iris-setosa

In [4]:

```
print(len(iris['class']))
```

```
150
```

In [5]:

```
for col in iris.columns:
    print(col)
```

```
sepallength
sepalwidth
petallength
petalwidth
class
```

In [6]:

```
print(iris.groupby('class').size())
```

```
class
Iris-setosa        50
Iris-versicolor    50
```

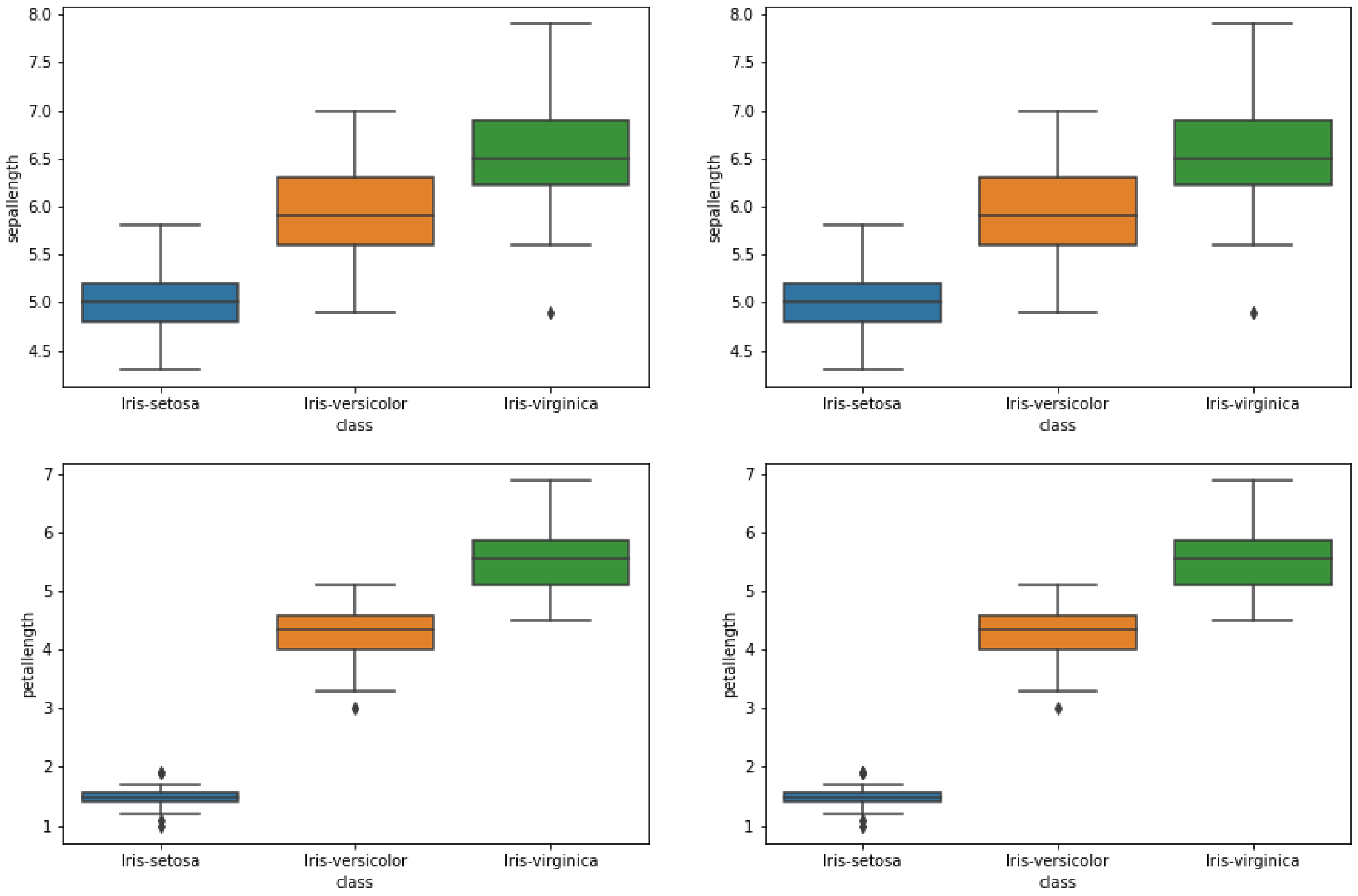
```
Iris-virginica     50
dtype: int64
```

In [10]:

```
plt.figure(figsize=(15,10))
plt.subplot(2,2,1)
sns.boxplot(x='class',y='sepallength',data=iris)
plt.subplot(2,2,2)
sns.boxplot(x='class',y='sepallength',data=iris)
plt.subplot(2,2,3)
sns.boxplot(x='class',y='petallength',data=iris)
plt.subplot(2,2,4)
sns.boxplot(x='class',y='petallength',data=iris)
```

Out[10]:

```
<AxesSubplot:xlabel='class', ylabel='petallength'>
```



In [12]:

```
#data cleaning
iris.isnull().values.any()
```

Out[12]:

```
False
```

In [13]:

```
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   sepallength  150 non-null    float64
 1   sepalwidth   150 non-null    float64
 2   petallength  150 non-null    float64
 3   petalwidth   150 non-null    float64
 4   class        150 non-null    object 
 4   class         150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [21]:

```
from sklearn.model_selection import train_test_split
array = iris.values
X =array[:,0:4]
Y =array[:,4]
x_train,x_test,y_train,y_test = train_test_split(X,Y,test_size=0.3,random_state=0)
```

In [22]:

```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
svc = SVC(max_iter=1000,gamma='auto')
svc.fit(x_train, y_train)
y_pred = svc.predict(x_test)
acc_svc = round(accuracy_score(y_pred,y_test), 2)*100
print("Accuracy :",acc_svc)
```

```
Accuracy : 98.0
```

In [25]:

```
from sklearn.linear_model import LogisticRegression
logreg=LogisticRegression(max_iter=1000)
logreg.fit(x_train,y_train)
y_pred = logreg.predict(x_test)
acc_logreg = round(accuracy_score(y_pred, y_test), 2)*100
print("accuracy: ",acc_logreg)
```

```
accuracy:  98.0
```

In [26]:

```
from sklearn.tree import DecisionTreeClassifier
decisiontree = DecisionTreeClassifier(random_state=0)
decisiontree.fit(x_train, y_train)
y_pred = decisiontree.predict(x_test)
acc_decisiontree = round(accuracy_score(y_pred,y_test), 2)*100
print("Accuracy :",acc_decisiontree)
```

```
Accuracy : 98.0
```

In [ ]: