In [1]:
```
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
import numpy as np

iris = load_iris()
```

In [4]:
```
print(iris.data[0:5])
```

```
[[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]]
```

In [12]:
```
X = iris.data
y = iris.target
```

In [21]:
```
X_new = (np.array([[3,4,5,2]]))
```

In [22]:
```
#instanstiate the model
logreg = LogisticRegression()

#fit the model with the data
logreg.fit(X,y)

#predict the response for new observations
logreg.predict(X_new)
```

Out[22]:
```
array([2])
```

In [24]:
```
# Assesing LR Accuracy

logreg = LogisticRegression()
logreg.fit(X,y)
# predict teh response values for the observations in X
logreg.predict(X)
```

Out[24]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       2, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

In [25]:
```
# Storing the predict response values in y_pred
y_pred = logreg.predict(X)

# Checking length of y_pred
len(y_pred)
```

Out[25]:
```
150
```

In [27]:
```
# Calculating classfication training accuracy using LR
from sklearn import metrics
print(metrics.accuracy_score(y, y_pred))
```

```
0.96
```

In [28]:
```
# Using train_test_split for better accuracy evaluation

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=.4, random_state = 4)
```

In [29]:
```
# training the model on the training set
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
```

Out[29]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

In [30]:
```
# make predictions on the testing set
y_pred = logreg.predict(X_test)

# comparing actual response values (y_test) with predicted response values (y_pred)
print(metrics.accuracy_score(y_test,y_pred))
```

```
0.95
```