Experiment -05

Logistic Regression Using ScikitLearn

In [1]:
```
print('-------------------------------EXPERIEMENT-05-------------------
---------------')
print('NAME:  Pratyush Srivastava')
print('ROLL NO: 18SCSE1010128')
```

```
-------------------------------EXPERIEMENT-05-------------------------
--------
NAME:  Pratyush Srivastava
ROLL NO: 18SCSE1010128
```

In [2]:
```
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
X, y = load_iris(return_X_y=True)
clf = LogisticRegression(random_state=0).fit(X, y)
clf.predict(X[:2, :])
```

```
C:\Users\praty\New folder\lib\site-packages\sklearn\linear_model\_logis
tic.py:762: 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
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
  n_iter_i = _check_optimize_result(
```

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

In [3]:
```
clf.predict_proba(X[:2, :])
```

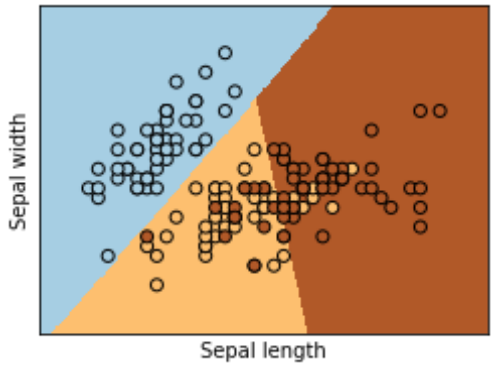
Out[3]:
```
array([[9.81815156e-01, 1.81848297e-02, 1.43980677e-08],
       [9.71755996e-01, 2.82439735e-02, 3.01113493e-08]])
```

In [4]:
```
clf.score(X, y)
```

Out[4]:
```
0.9733333333333334
```

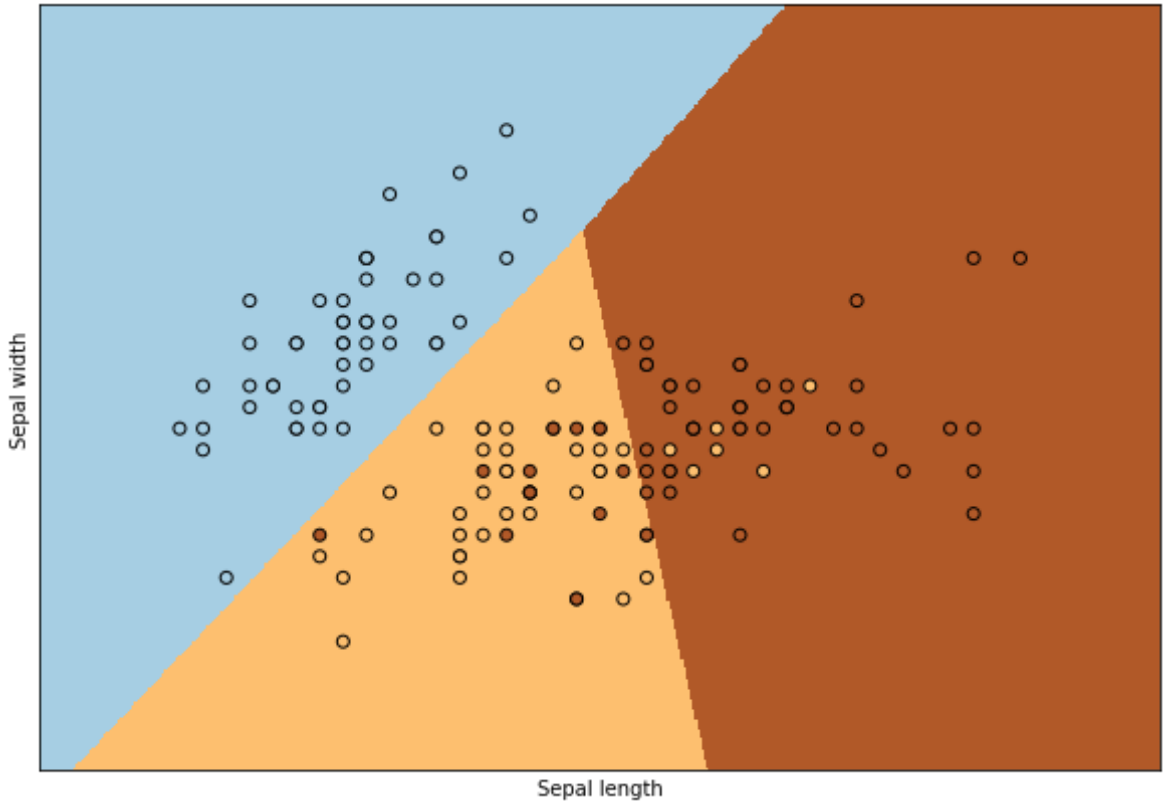
In [5]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn import datasets


iris = datasets.load_iris()
X = iris.data[:, :2]
Y = iris.target

logreg = LogisticRegression(C=1e5)


logreg.fit(X, Y)


x_min, x_max = X[:, 0].min() - .5, X[:, 0].max() + .5
y_min, y_max = X[:, 1].min() - .5, X[:, 1].max() + .5
h = .02  # step size in the mesh
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max
, h))
Z = logreg.predict(np.c_[xx.ravel(), yy.ravel()])


Z = Z.reshape(xx.shape)
plt.figure(1, figsize=(4, 3))
plt.pcolormesh(xx, yy, Z, cmap=plt.cm.Paired)

plt.scatter(X[:, 0], X[:, 1], c=Y, edgecolors='k', cmap=plt.cm.Paired)
```

```
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')

plt.xlim(xx.min(), xx.max())
plt.ylim(yy.min(), yy.max())
plt.xticks(())
plt.yticks(())

plt.show()
```



In [6]:
```
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn import datasets


iris = datasets.load_iris()
X = iris.data[:, :2]
Y = iris.target

logreg = LogisticRegression(C=1e5)


logreg.fit(X, Y)


x_min, x_max = X[:, 0].min() - .6, X[:, 0].max() + .6
y_min, y_max = X[:, 1].min() - .6, X[:, 1].max() + .6
```

```
h = .01
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max
, h))
Z = logreg.predict(np.c_[xx.ravel(), yy.ravel()])


Z = Z.reshape(xx.shape)
plt.figure(1, figsize=(10, 7))
plt.pcolormesh(xx, yy, Z, cmap=plt.cm.Paired)

plt.scatter(X[:, 0], X[:, 1], c=Y, edgecolors='k', cmap=plt.cm.Paired)
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')

plt.xlim(xx.min(), xx.max())
plt.ylim(yy.min(), yy.max())
plt.xticks(())
plt.yticks(())

plt.show()
```



In [7]:
```
print('-------------------------EXPERIMENT-05 Ended--------------------
-------------')
```

```
-------------------------EXPERIMENT-05 Ended--------------------------
------
```

In [ ]:
```

```