Chapter 6 - Other Popular Machine Learning Methods

Segment 2 - A neural network with a Perceptron

```
import numpy as np
import pandas as pd
import sklearn

from pandas import Series, DataFrame
from sklearn import datasets
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report


from sklearn.linear_model import Perceptron


iris = datasets.load_iris()

X = iris.data
y = iris.target

X[0:10,]

     array([[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],
            [5.4, 3.9, 1.7, 0.4],
            [4.6, 3.4, 1.4, 0.3],
            [5. , 3.4, 1.5, 0.2],
            [4.4, 2.9, 1.4, 0.2],
            [4.9, 3.1, 1.5, 0.1]])


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

```
standardize = StandardScaler()

standardized_X_test = standardize.fit_transform(X_test)

standardized_X_train = standardize.fit_transform(X_train)

standardized_X_test[0:10,]

     array([[ 0.25491318, -0.54344897,  0.43157675,  0.47066431],
            [-1.34857295,  0.31462835, -1.38558852, -1.34675235],
            [ 0.62494844, -1.11550053,  0.65872241,  0.89006816],
            [ 0.13156809,  0.31462835,  0.60193599,  0.89006816],
            [-1.22522786,  0.02860258, -1.21522927, -1.34675235],
            [-0.97853769,  0.88667991, -1.27201569, -1.34675235],
            [ 2.35177965, -1.4015263 ,  1.7944507 ,  1.58907457],
            [-0.73184751,  1.17270568, -1.27201569, -1.34675235],
            [ 0.00822301, -1.11550053,  0.77229524,  1.02986944],
            [ 0.87163861, -0.2574232 ,  0.82908165,  1.16967072]])


perceptron = Perceptron(max_iter=50, eta0=0.15, tol=1e-3, random_state=15)

perceptron.fit(standardized_X_train, y_train.ravel())

     Perceptron(alpha=0.0001, class_weight=None, early_stopping=False, eta0=0.15,
                fit_intercept=True, max_iter=50, n_iter_no_change=5, n_jobs=None,
                penalty=None, random_state=15, shuffle=True, tol=0.001,
                validation_fraction=0.1, verbose=0, warm_start=False)


y_pred = perceptron.predict(standardized_X_test)


print(y_test)

     [1 0 2 1 0 0 2 0 2 2 1 0 0 2 2 0 1 1 0 2 1 0 2 1 1 1 1 1 1 0]


print(y_pred)

     [2 0 2 2 0 0 2 0 2 2 0 0 0 2 2 0 0 2 0 2 1 0 2 1 1 0 0 0 1 0]


print(classification_report(y_test, y_pred))

                   precision    recall  f1-score   support
```

```
              0       0.67      1.00      0.80        10
              1       1.00      0.33      0.50        12
              2       0.73      1.00      0.84         8

       accuracy                           0.73        30
      macro avg       0.80      0.78      0.71        30
   weighted avg       0.82      0.73      0.69        30
```