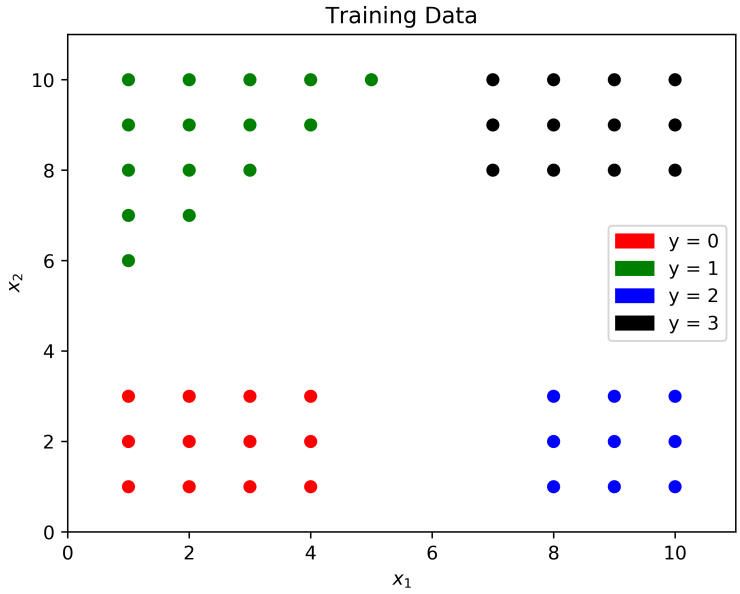
Data Separators & Classification

Eric Rice
ericjrice95@gmail.com

This project presents examples using two types of classifiers which function by separating d-dimensional data via a $d-1$-dimensional surface known as a *decision boundary*. For linearly separable data, we will see an example of the Perceptron classifier. This extends to the Kernel Perceptron classifier for use in nonlinearly separable data. Secondly, we will examine the Kernel Support Vector Machine (SVM) in the case of general data.

1 Linearly Separable Data

In this example, our training set consists of $n = 48$ samples $\{(x_i, y_i)\}_{i=1}^n$ of 2-dimensional data points with 4 possible labels. That is to say, if $(x, y) \in \{(x_i, y_i)\}_{i=1}^n$, then $x = (x_1, x_2) \in \mathbb{R}^2$ and $y \in \{0, 1, 2, 3\}$. The given data is visualized below. Notice that the points are linearly separable.



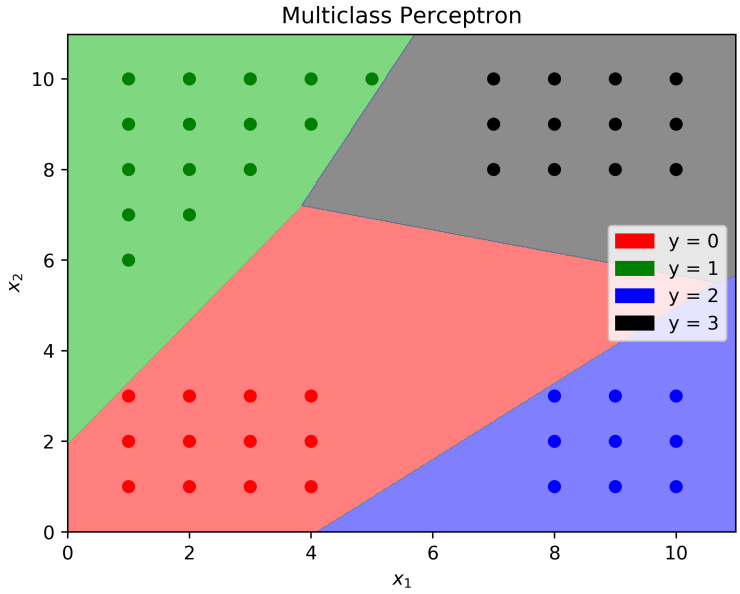
The Multiclass Perceptron algorithm computes binary linear separators $w_1, w_2, w_3, w_4 \in \mathbb{R}^2$ along with intercept terms $b_1, b_2, b_3, b_4 \in \mathbb{R}$ (respectively, for each label) that correctly classify the training data using the decision function

$$\hat{y}(x) := \underset{j \in \{0,1,2,3\}}{\arg\max} \left[w_j \cdot x + b_j \right].$$

This is achieved using an algorithm outlined in the following pseudocode:

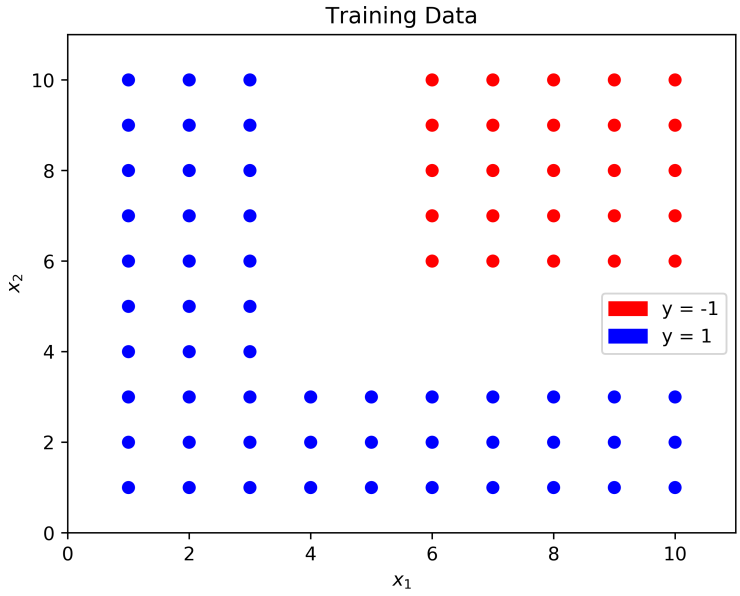
initialize $w_j = (0, 0)$ and $b_j = 0$ for all $j = 0, 1, 2, 3$
while some training point (x, y) is misclassified (i.e. $y \neq \hat{y}(x)$):
    $w_y = w_y + x$
    $b_y = b_y + 1$
    $w_{\hat{y}(x)} = w_{\hat{y}(x)} - x$
    $b_{\hat{y}(x)} = b_{\hat{y}(x)} - 1$

After training, the Multiclass Perceptron algorithm yields the following decision boundaries and regions:



2 Nonlinearly Separable Data

Next, we consider data which is not linearly separable. We are given a training set $\{(x_i, y_i)\}_{i=1}^n$ where $n = 76$ and again $x_i \in \mathbb{R}^2$, but our labels are now binary $y_i \in \{-1, 1\}$. The given data is plotted below.



To classify this data, we will implement a quadratic decision boundary using the kernel function

$$k(x, z) = (1 + x \cdot z)^2.$$

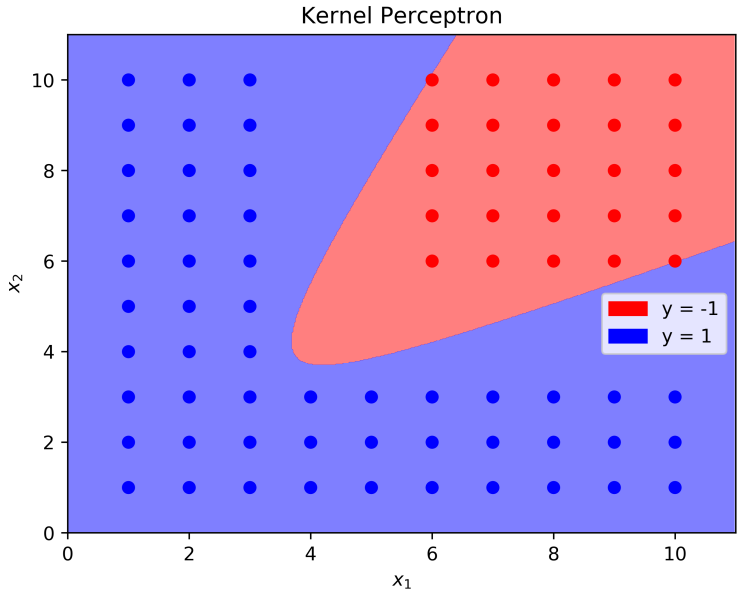
This is accomplished by solving the dual form of the Perceptron method which involves the decision function

$$\hat{y}(x) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i k(x_i, x) + b\right),$$

where $\alpha \in \mathbb{R}^n$ and $b \in \mathbb{R}$ are parameters to be learned. This learning is done in a way described by the pseudocode below.

initialize $\alpha = \vec{0}$
while some training point (x_i, y_i) is misclassified (i.e. $y \neq \hat{y}(x)$):
    $\alpha_i = \alpha_i + 1$
    $b = b + y_i$

After training, the Kernel Perceptron algorithm yields the following decision boundary and regions:



3 General Data

We now consider the MNIST data set of handwritten digits, which contains 60,000 training images and 10,000 test images. Below are test results after training a linear separator using an SVM via `sklearn.svm.LinearSVC`, for different values of the parameter C.

$C =$	Training Error %	Test Error %
0.01	11.46	12.12
0.1	10.23	11.02
1	12.64	13.24
10	12.78	13.55
100	12.52	12.70

As is evident from the table above, the error rate is high regardless of the choice of parameter C. Thus, we can assume with high confidence that our data is not linearly separable. (Though this was intuitively true as well.) If instead we train a quadratic separator using an SVM via `sklearn.svm.SVC` with `kernel=''poly''` and `degree=2`, we get very different results.

$C =$	Training Error %	Test Error %
1	0.00	1.94

There were 8652 total support vectors in this model.