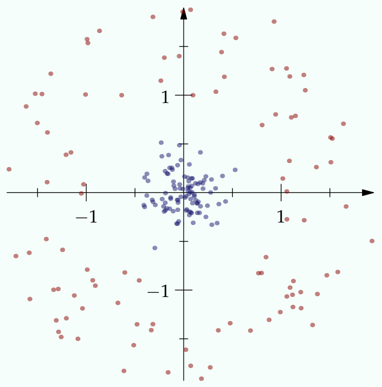
DATA 1010
In-class exercises

Samuel S. Watson
14 November 2018

Problem 1

Support vector machines can be used to find nonlinear separating boundaries, because we can map the feature vectors into a higher-dimensional space and use a support vector machine find a separating hyperplane in that space.

Find a map from $\mathbb{R}^2$ to a higher dimensional space such that the two classes shown in the figure can be separated with a hyperplane.



Problem 2

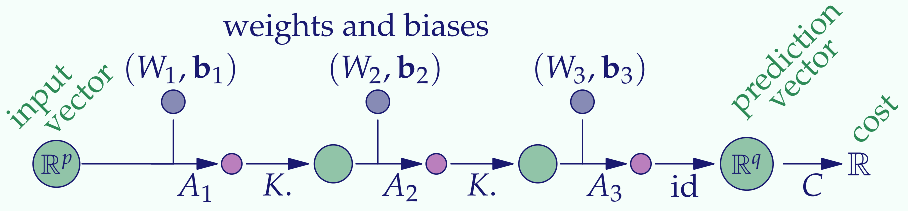
An **affine function** from $\mathbb{R}^t$ to $\mathbb{R}^s$ is a function of the form $\mathbf{x} \mapsto A\mathbf{x} + \mathbf{b}$, where A is an $s \times t$ matrix and $\mathbf{b} \in \mathbb{R}^t$. Show that a composition of affine functions is affine.

Problem 3

Suppose that $K : \mathbb{R} \to \mathbb{R}$ is a differentiable function with derivative $\dot{K}$. Find the derivative of the map $K.$ defined by $K.(x_1, \ldots, x_t) = (K(x_1), \ldots, K(x_t))$

Problem 4

A **neural network** function $N : \mathbb{R}^p \to \mathbb{R}^q$ is a composition of affine transformations and componentwise applications of a function $K : \mathbb{R} \to \mathbb{R}$ called the **activation** function.



We train the neural network for regression on the samples $(\mathbf{X}_i, Y_i)$ by minimizing the loss $L(N) = \sum_{i=1}^{n} C(N(\mathbf{x}_i), \mathbf{y}_i)$ where $C(\mathbf{u}, \mathbf{v}) = |\mathbf{u} - \mathbf{v}|^2$.

Create data types in Julia to represent affine maps and neural net functions. Write an `architecture` method for neural nets which returns the sequence of dimensions of the domains and codomains of its affine maps.

Problem 5

Write a Julia function `forwardprop` which calculates the sequence of vectors obtained by applying each successive map in the neural net.

Problem 6

Write a Julia function `backprop` which calculates the for each node the derivative of the cost function with respect to the value at that node. In other words, calculate the derivative of the composition of maps between that node and the cost node at the end.

Problem 7

Write two functions `weight_gradients` and `bias_gradients` which compute the derivatives with respect to W_i and $\mathbf{b}_i$ of $C(N(\mathbf{x}))$.

Problem 8

Write a function `train` which performs *stochastic gradient descent*: For a randomly chosen subset of the training set, determine the average desired change in weights and biases to reduce the cost function. Update the weights and biases accordingly and iterate to convergence.

Your function should take 7 arguments: one for the architecture of the neural nets to train, one for the activation function K, one for its derivative, one for the training samples, one for the batch size (the number of samples to use in each randomly chosen subset used in a single stochastic gradient descent iteration), one for the learning rate ϵ, and one for the number of iterations to run.

Problem 9

Try training your model on some data which are sampled by taking $\mathbf{X}$ uniform in the unit square and setting $Y = |\mathbf{X}|^2$.