CSE 546 Machine Learning, Autumn 2013
Homework 3

Due: Monday, November 18, beginning of class

1 Fitting an SVM classifier by hand [25 Points]

(Source: Murphy text, Exercise 14.1) Consider a dataset with 2 points in 1d: $(x_1 = 0, y_1 = -1)$ and $(x_2 = \sqrt{2}, y_2 = 1)$. Consider mapping each point to 3d using the feature vector $\phi(x) = [1, \sqrt{2}x, x^2]^T$. (This is equivalent to using a second order polynomial kernel.) The max margin classifier has the form

$$min||w||^2 \qquad s.t. \tag{1}$$
$$y_1(w^T\phi(x_1) + w_0) \geq 1 \tag{2}$$
$$y_2(w^T\phi(x_2) + w_0) \geq 1 \tag{3}$$

1. *(5 Points)* Write down a vector that is parallel to the optimal vector w. Hint: recall from Figure 14.12 (page 500 in the Murphy text) that w is perpendicular to the decision boundary between the two points in the 3d feature space.

2. *(5 Points)* What is the value of the margin that is achieved by this w? Hint: recall that the margin is the distance from each support vector to the decision boundary. Hint 2: think about the geometry of 2 points in space, with a line separating one from the other.

3. *(5 Points)* Solve for w, using the fact the margin is equal to $1/||w||$.

4. *(5 Points)* Solve for w_0 using your value for w and Equations 1 to 3. Hint: the points will be on the decision boundary, so the inequalities will be tight. A "tight inequality" is an inequality that is as strict as possible. For this problem, this means that plugging in these points will push the left-hand side of Equations 2 and 3 as close to 1 as possible.

5. *(5 Points)* Write down the form of the discriminant function $f(x) = w_0 + w^T\phi(x)$ as an explicit function of x. Plot the 2 points in the dataset, along with $f(x)$ in a 2d plot. You may generate this plot by hand, or using a computational tool like python.

Show your work.

2 Manual calculation of one round of EM for a GMM [30 points]

(Extended version of: Murphy Exercise 11.7) In this question we consider clustering 1D data with a mixture of 2 Guassians using the EM algorithm. You are given the 1-D data points $x = [1 \;\; 10 \;\; 20]$.

M step

Suppose the output of the E step is the following matrix:

$$R = \begin{pmatrix} 1 & 0 \\ 0.4 & 0.6 \\ 0 & 1 \end{pmatrix}$$

where entry $r_{i,c}$ is the probability of observation x_i belonging to cluster c (the responsibility of cluster c for data point i). You just have to compute the M step. You may state the equations for maximum likelihood estimates of these quantities (which you should know) without proof; you just have to apply the equations to this data set. You may leave your answer in fractional form. Show your work.

1. *[3 points]* Write down the likelihood function you are trying to optimize.
2. *[6 points]* After performing the M step for the mixing weights π_1, π_2, what are the new values?
3. *[6 points]* After performing the M step for the means μ_1 and μ_2, what are the new values?
4. *[6 points]* After performing the M step for the standard deviations σ_1 and σ_2, what are the new values?

E step

Now suppose the output of the M step is the answer to the previous section. You will compute the subsequent E step.

1. *[3 points]* Write down the formula for the probability of observation x_i belonging to cluster c.
2. *[6 points]* After performing the E step, what is the new value of R?

3 Programming Question [45 Points]

In this problem, we seek to perform a digit recognition task, where we are given an image of a handwritten digit and wish to predict what number it represents. This is a special case of an important area of language processing known as Optical Character Recognition (OCR). We will be simplifying our goal to that of a binary classification between two relatively hard-to-distinguish numbers (specifically, predicting a '3' versus a '5'). To do this, you will implement a kernelized Perceptron and linear support vector machine (SVM).

3.1 Dataset

The digit images have been taken from a Kaggle competition, `http://www.kaggle.com/c/digit-recognizer/data`. This data was originally from the MNIST database of handwritten digits, but was converted into a easier-to-use file format.

The data have also undergone some preprocessing. They have been filtered to just those datapoints whose labels are 3 or 5, which have then been relabeled to 1 and -1 respectively. Then, 1000-point samples have been created, named *validation.csv* and *test.csv* on the class website. Each row in these files represents an image. The first column is the label of the image, and the remaining columns give the grayscale values for each pixel.

3.2 Perceptron

1. *(5 points)* Write the prediction rule of kernelized Perceptron on the t-th iteration of its training procedure in terms of a kernel function $k(u, v) = \phi(u) \cdot \phi(v)$, where u and v are data points and ϕ is a projection to a feature space. That is, give an expression for $\text{sign}\left(w^{(t)} \cdot \phi(x)\right)$ in terms of k and previously misclassified data points, where $w^{(t)}$ is the weight vector estimated on the t-th iteration, and x is a data point whose label we want to predict. You will use this expression to implement Perceptron. Ignore the intercept, as it will appear naturally when we apply kernel functions. You may cite the lecture slides.

2. Implement Perceptron. Initially start all the weights at 0. Use one pass over the data. Write it so that it takes a kernel function as a parameter.

3. *(6 Points)* Consider the kernel $k_p^1(u, v) = u \cdot v + 1$. ($k_p^1$ is what the standard dot product would give us, if we had added a constant term 1.) Run Perceptron on the **validation** set with this kernel, and plot the average loss $\bar{L}$ as a function of the number of steps T, where

$$\bar{L}(T) = \frac{1}{T} \sum_{j=1}^{T} \mathbb{I}(\hat{y}^{(t)} \neq y^{(t)})$$

   Here $\hat{y}^t$ is the label that Perceptron predicts for datapoint t as it runs, and $\mathbb{I}$ is an indicator function, which is 1 if its condition is true and 0 otherwise. Only show the average loss every 100 steps, e.g. [100, 200, 300, ...].

4. *(6 Points)* For a positive integer d, the polynomial kernel $k_p^d(u, v) = (u \cdot v + 1)^d$ maps x into a feature space of all polynomials of degree up to d. For the set $d = [1, 3, 5, 7, 10, 15, 20]$, run Perceptron for a single pass over the **validation** set with k_p^d, and plot the average loss over the validation set $\bar{L}(1000)$ as a function of d.

5. *(6 Points)* For $\sigma > 0$, the Gaussian kernel $k_G^\sigma(u, v) = \exp\left(-\frac{\|u-v\|^2}{2\sigma^2}\right)$ is a map to *all* polynomials of x, where σ is a tuning constant that roughly corresponds to the "window size" of the distribution. The Gaussian kernel, also known as the radial basis function (RBF) kernel, is one of the most popular kernel functions because it performs well on a variety of data, making it an ideal choice as a default kernel.

   Tuning on the validation set has produced a value of $\sigma = 1000$. For the best d in the previous step, run Perceptron with both k_p^d and k_G^{1000} for a single pass over the **testing** set. (Use a fresh run of the algorithm; do not reuse the set of mistakes from the validation set.) For each of these two kernels, plot the average loss $\bar{L}(T)$ as a function of the number of steps on the same graph. As above, show the average loss for every 100 steps.

3.3 SVM

In this problem we consider only linear SVMs—that is, we do not use kernels. Although kernels would perform better, their implementation is quite complicated. Linear SVMs, on the other hand, can be trained by stochastic gradient descent, though this method is not the best known.

1. *(5 points)* Write the stochastic gradient descent update rules for both intercept and non-intercept weights in linear SVMs. The formula in the lecture slides has a typo, which is left for you to find.

2. Implement linear SVMs. Initially start all the weights at 0. Use one pass over the data.

3. *(6 points)* Run SVM on the **testing** set with $\eta = 10^{-5}$ and $C = 1$. (Use a fresh run of the algorithm; do not use weights learned from the validation set.) Plot the average loss for every 100 steps.

4. *(5 points) Removed because the premise was incorrect. Everyone who submitted the assignment will receive full credit on this question.*

5. *(6 points)* Since C should matter, describe how the expected behavior of linear SVMs changes as C increases and decreases. What is the relationship to Perceptron?