CSE 575: Statistical Machine Learning Assignment #2

Instructor: Prof. Jingrui He
Out: Sep. 21th, 2016; Due: Oct. 20th, 2016

Submit electronically, using the submission link on Blackboard for Assignment #2, a file named `yourFirstName-yourLastName.pdf` *containing your solution to this assignment (a .doc or .docx file is also acceptable, but .pdf is preferred).*

1 SVM [15 points]

1 (**Kernel, 10 points**) Given the following data set in 1-d space (Figure 1), which consists of 3 positive data points $\{-1, 0, 1\}$ and 3 negative data points $\{-3, -2, 2\}$.

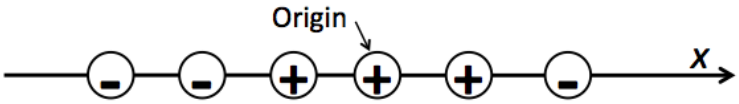

Figure 1: Training Data Set for SVM Classifiers

(1) Find a feature map($\{\mathbf{R}^1 \rightarrow \mathbf{R}^2\}$), which will map the original 1-d data points to 2-d space so that the positive set and negative set are linearly separable with each other. Plot the dataset after mapping in 2-d space.

Solution. $x \rightarrow \{x, x^2\}$. See Figure 2.

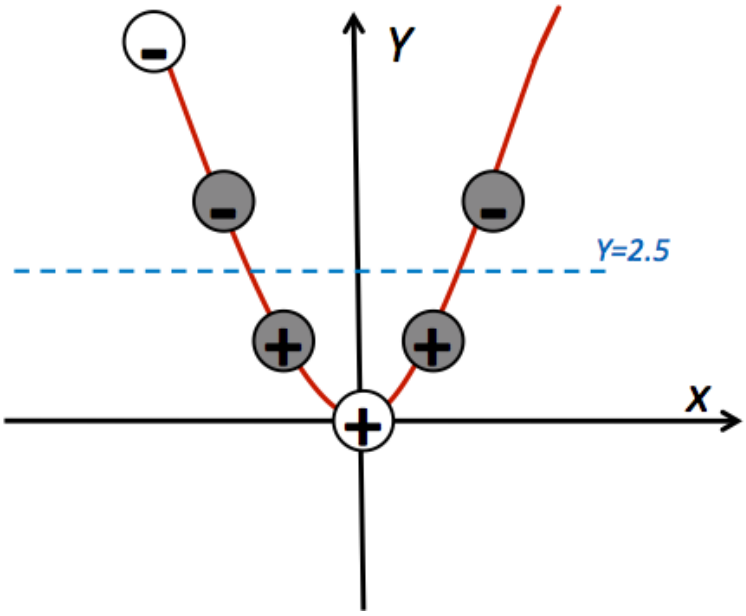

Figure 2: Feature Map

(2) In your plot, draw the decision boundary given by hard-margin linear SVM. Mark the corresponding support vector(s).

Solution. See figure 2 (supported vectors are shadowed ones).

(3) For the feature map you choose, what is the corresponding kernel $K(x_1, x_2)$?

Solution. $K(x_1, x_2) = x_1 x_2 + (x_1 x_2)^2$.

2 (**Hinge Loss, 5 points**) Given m training data points $\{x_i, y_i\}_{i=1}^m$, remember that the soft-margin linear SVM can be formalized as the following constrained quadratic optimization problem:

$$\begin{aligned} &\text{argmin}_{\{w,b\}} \ \frac{1}{2} w^t w + C \sum_{i=1}^{m} \epsilon_i \\ &\text{Subject to :} \ \ y_i(w^t x_i + b) \geq 1 - \epsilon_i \\ &\qquad\qquad \epsilon_i \geq 0 \ \forall i \end{aligned} \tag{1}$$

(1) Prove that the above formulation is equivalent to the following unconstrained quadratic optimization problem:

$$\text{argmin}_{\{w,b\}} w^t w + \lambda \sum_{i=1}^{m} \max(1 - y_i(w^t x_i + b), 0) \tag{2}$$

Solution. By eq. (1), we have $\epsilon_i \geq \max(1 - y_i(w^t x_i + b), 0), \ \forall i$. Since we want to minimize ϵ_i, we must have $\epsilon_i = \max(1 - y_i(w^t x_i + b), 0), \ \forall i$. Plus this into eq. (1), we have eq. (2).

(2) What is your intuition for this new optimization formulation (1 or 2 sentences)?

Solution. Soft-margin SVM tries to balance the simplicity of classifier (the first term) vs. the good prediction on training dataset (the second term).

(3) What is the value for λ (as a function of C)?

Solution. $\lambda = 2C$.

2 More on SVM [30 points]

1. (**5 points**) Suppose we are using a linear SVM (i.e., no kernel), with some large C value, and are given the following data set.

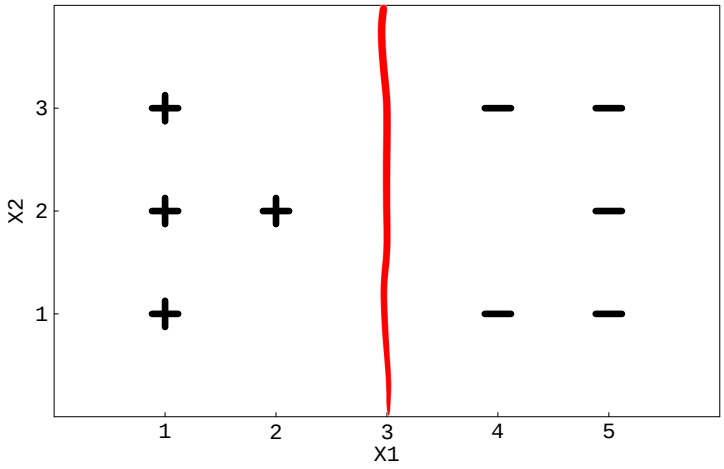

Figure 3

Draw the decision boundary of linear SVM. Jutify your answer.

Solution. Because of the large C value, the decision boundary will classify all of the examples correctly. Furthermore, among separators that classify the examples correctly, it will have the largest margin (distance to closest point).

2. (**5 points**) In the following image, circle the points such that after removing that point (example) from the training set and retraining SVM, we would get a different decision boundary than training on the full sample.

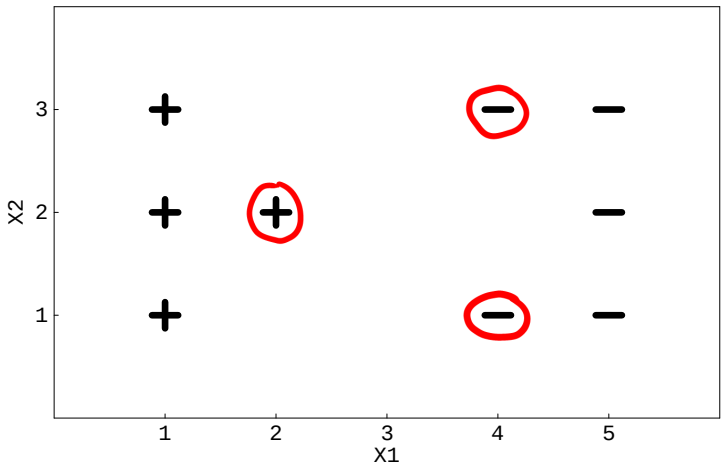

Figure 4

Solution. These examples are the support vectors; all of the other examples are such that their corresponding constraints are not tight in the optimization problem, so removing them will not create a solution with smaller objective function value (norm of w). These three examples are positioned such that removing any one of them introduces slack in the constraints, allowing for a solution with a smaller objective function value and with a different

third support vector; in this case, because each of these new (replacement) support vectors is not close to the old separator, the decision boundary shifts to make its distance to that example equal to the others.

3. (**5 points**) Suppose instead of SVM, we use regularized logistic regression to learn the classifier. That is,

$$(w, b) = \arg \min_{w \in \mathbb{R}^2, b \in \mathbb{R}} \frac{\|w\|^2}{2} - \sum_i \mathbb{1}[y_i = 0] \ln \frac{1}{1 + e^{(w \cdot x_i + b)}} + \mathbb{1}[y_i = 1] \ln \frac{e^{(w \cdot x_i + b)}}{1 + e^{(w \cdot x_i + b)}}.$$

In the following image, circle the points such that after removing that point (example) from the training set and running regularized logistic regression, we would get a different decision boundary than training with regularized logistic regression on the full sample.

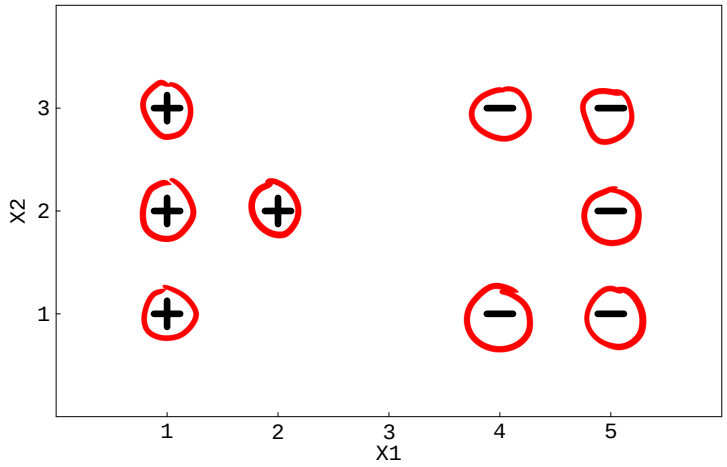

Figure 5

Solution. Because of the regularization, the weights will not diverge to infinity, and thus the probabilities at the solution are not at 0 and 1. Because of this, *every* example contributes to the loss function, and thus has an influence on the solution.

4. (**5 points**) Suppose we have a kernel $K(\cdot, \cdot)$, such that there is an implicit high-dimensional feature map $\phi : \mathbb{R}^d \to \mathbb{R}^D$ that satisfies $\forall x_i, x_j \in \mathbb{R}^d,\ K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$, where $\phi(x_i) \cdot \phi(x_j) = \sum_{l=1}^{D} \phi(x_i)^l \phi(x_j)^l$ is the dot product in the D-dimensional space, and $\phi(x_i)^l$ is the l^{th} element/feature in the D-dimensional space.

Show how to calculate the Euclidean distance in the D-dimensional space

$$\|\phi(x_i) - \phi(x_j)\| = \sqrt{\sum_{l=1}^{D} (\phi(x_i)^l - \phi(x_j)^l)^2}$$

without explicitly calculating the values in the D-dimensional space. For this question, **please provide a formal proof**.
Hint: Try converting the Euclidean distance into a set of inner products.

Solution.

$$
\begin{aligned}
\|\phi(x_i) - \phi(x_j)\| &= \sqrt{\sum_{l=1}^{D} (\phi(x_i)^l - \phi(x_j)^l)^2} \\
&= \sqrt{\sum_{l=1}^{D} (\phi(x_i)^l)^2 + (\phi(x_j)^l)^2 - 2\phi(x_i)^l \phi(x_j)^l} \\
&= \sqrt{\left(\sum_{l=1}^{D} (\phi(x_i)^l)^2\right) + \left(\sum_{l=1}^{D} (\phi(x_j)^l)^2\right) - \left(\sum_{l=1}^{D} 2\phi(x_i)^l \phi(x_j)^l\right)} \\
&= \sqrt{\phi(x_i) \cdot \phi(x_i) + \phi(x_j) \cdot \phi(x_j) - 2\phi(x_i) \cdot \phi(x_j)} \\
&= \sqrt{K(x_i, x_i) + K(x_j, x_j) - 2K(x_i, x_j)}.
\end{aligned}
$$

5. (**5 points**) Assume that we use the RBF kernel function $K(x_i, x_j) = \exp(-\frac{1}{2}\|x_i - x_j\|^2)$. Also assume the same notation as in the last question. Prove that for any two input examples x_i and x_j, the squared Euclidean distance of their corresponding points in the high-dimensional space $\mathbb{R}^D$ is less than 2, i.e., prove that $\|\phi(x_i) - \phi(x_j)\|^2 < 2$.

   Solution. This inequality directly follows from the result from the last question.

6. (**5 points**) Assume that we use the RBF kernel function, and the same notation as before. Consider running One Nearest Neighbor with Euclidean distance in both the input space $\mathbb{R}^d$ and the high-dimensional space $\mathbb{R}^D$. Is it possible that One Nearest Neighbor classifier achieves better classification performance in the high-dimensional space than in the original input space? Why?

   Solution. No.

3 1NN-Classifier Decision Boundary [20 points]

Given two training data points as shown in Figure 6, what is the decision boundary of 1NN classifier if we use L_2 distance (10 points)? What will be the decision boundary if we use L_∞ distance instead (10 points)?Justify your answer.

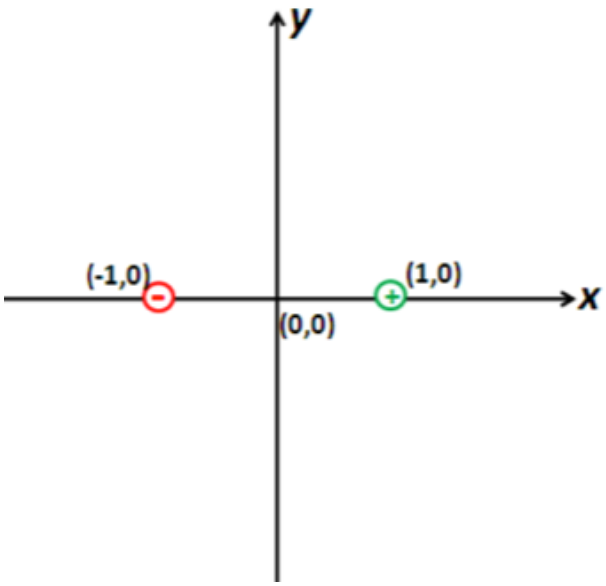

Figure 6: Training Data Set for 1NN Classifiers

Solutions: (1) the y-axis. (2) shown in the following figure.

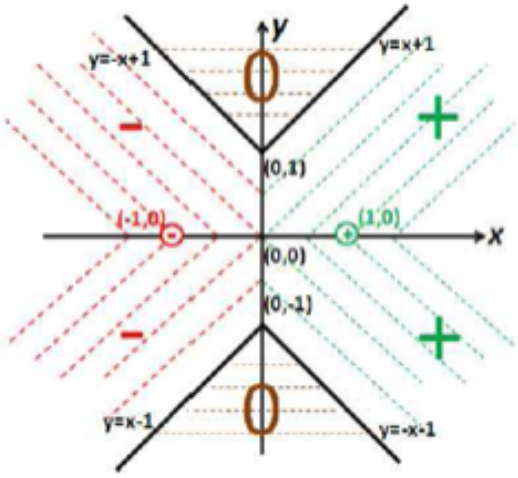

Figure 7: Decision Boundary of 1NN Classifiers with L_∞ Distance.

4 k-NN Implementation [35 points]

Download the file hw2.zip and unpack it. The file faces.mat contains the Matlab variables traindata (training data), trainlabels (training labels), testdata (test data), testlabels (test labels) and evaldata (evaluation data, needed later). This is a facial attractiveness classification task: given a picture of a face, you need to predict whether the average rating of the face is hot or not. So, each row corresponds to a data point (a picture). Each column is a feature, a pixel. The value of the feature

is the value of the pixel in a grayscale image. cosineDistance.m implements the cosine distance, a simple distance function. It takes two feature vectors x and y, and computes a nengative, symmetric distance between x and y. To check your data, compute the distance between the first training example from each class. (It should be 0.2617)

Implement the k-Nearest Neighbor (k-NN) algorithm. *You might want to pre-compute the distance between all pairs of points, to speed up the computation.* For $k = 1, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100$, compute and plot the training error and the test error. Include the plot and the pseudocode in your pdf file (or .doc/.docx file). Submit the actual code as a single zip file named yourFirstName-yourLastName.zip IN ADDITION TO the pdf file (or .doc/.docx file).

Based on your plot, does the value of k which minimizes the training error also minimize the test error? Either way, can you explain why? Also, what does this tell us about using the training error to pick the value of k?

Solutions: The value of k that minimizes the training error ($k = 1$) does NOT minimize the test error because of overfitting. We should NEVER use training error to pick the optimal value of k

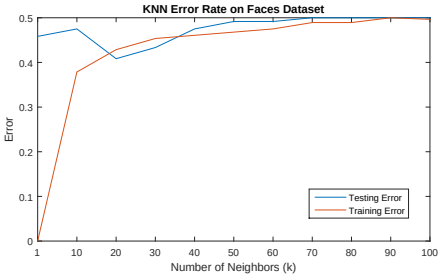

Figure 8: Training/test error of KNN.