10701/15781 Machine Learning, Fall 2007: Homework 3

Due: Monday, November 5, beginning of the class

Instructions

There are 4 questions on this assignment. Problem 3 involves coding. Do not attach your code to the writeup. Instead, copy your implementation to

`/afs/andrew.cmu.edu/course/10/701/Submit/your_andrew_id/HW2`

To write in this directory, you need a kerberos instance for andrew, or you can log into, for example, unix.andrew.cmu.edu.

Please submit each problem seperately with your name and Andrew ID on each problem. Refer to the webpage for policies regarding collaboration, due dates, and extensions.

1 Kernel Regression and Locally Weighted Regression [Jingrui, 15 points]

Given a set of n examples, (x_j, y_j) (x_j is the input, and y_j is output), $j = 1, \ldots, n$, a linear smoother is defined as follows. For any x, there exists a vector $l(x) = (l_1(x), \ldots, l_n(x))^T$ such that the estimated output $\hat{y}$ of x is $\hat{y} = \sum_{j=1}^{n} l_j(x) y_j =< l(x), Y >$, where Y is a $n \times 1$ vector, $Y_j = y_j$, and $< a, b >$ is the inner product between two vectors a and b.

1. Recall that in linear regression, we assume the data are generated from the linear regression model $y_j = \sum_{i=1}^{k} w_i h_i(x_j) + \epsilon_j$. The least squares estimate for the coefficient vector w is given by $w^* = (H^T H)^{-1} H^T Y$, where H is a $n \times k$ matrix, $H_{ji} = h_i(x_j)$. Given an input x, what is the estimated output $\hat{y}$? (Matrix form solution is required. You may want to use the $k \times 1$ vector $h(x) = [h_1(x), \ldots, h_k(x)]^T$)
2. Based on part 1, prove that linear regression is a linear smoother. Give the vector $l(x)$ for a given input x.
3. In kernel regression, given an input x, what is the estimated output $\hat{y}$?
4. Based on part 3, prove that kernel regression is a linear smoother. Give the vector $l(x)$ for a given input x.
5. In local weighted regression, given an input x, what is the estimated output $\hat{y}$?
6. Based on part 5, prove that locally weighted regression is a linear smoother. Give the vector $l(x)$ for a given input x.
7. If we divide the range (a, b) (a and b are real numbers, and $a < b$) into m equally spaced bins denoted by $B_1, \ldots, B_m$. Define the estimated output $\hat{y} = \frac{1}{k_i} \sum_{j: x_j \in B_i} y_j$, for $x \in B_i$, where k_i is the number of points in B_i. In other words, the estimate $\hat{y}$ is a step function obtained by averaging the y_js over each bin. This estimate is called the regressogram. Is this estimate a linear smoother? If yes, give the vector $l(x)$ for a given input x; otherwise, state your reasons.

2 [20 points] Feature Maps, Kernels, and SVM (Joey)

You are given a data set D in Figure 1 with data from a single feature X_1 in $\mathbb{R}^1$ and corresponding label $Y \in \{+, -\}$. The data set contains four positive examples at $X_1 = \{-3, -2, 3\}$ and three negative examples at $X_1 = \{-1, 0, 1\}$.

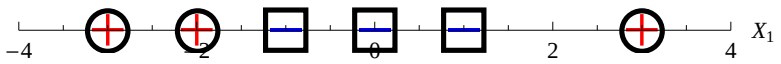

Figure 1: Dataset for SVM feature map task in Question 2.

2.1 Finite Features and SVMs

1. [1pt] Can this data set (in its current feature space) be perfectly separated using a linear separator? Why or why not?
2. [1pt] Lets define the simple feature map $\phi(u) = (u, u^2)$ which transforms points in $\mathbb{R}^1$ to points in $\mathbb{R}^2$. Apply ϕ to the data and plot the points in the new $\mathbb{R}^2$ feature space.
3. [1pt] Can a linear separator perfectly separate the points in the new $\mathbb{R}^2$ features space induced by ϕ? Why or why not?
4. [1pt] Give the analytic form of the kernel that corresponds to the feature map ϕ in terms of only X_1 and X_1'. Specifically define $k(X_1, X_1')$.
5. [3pt] Construct a maximum-margin separating hyperplane. This hyperplane will be a line in $\mathbb{R}^2$, which can be parameterized by its normal equation, i.e. $w_1 Y_1 + w_2 Y_2 + c = 0$ for appropriate choices of w_1, w_2, c. Here, $(Y_1, Y_2) = \phi(X_1)$ is the result of applying the feature map ϕ to the original feature X_1. Give the values for w_1, w_2, c. Also, explicitly compute the margin for your hyperplane. You do not need to solve a quadratic program to find the maximum margin hyperplane. Instead, let your geometric intuition guide you.
6. [1pt] On the plot of the transformed points (from part 3), plot the separating hyperplane and the margin, and circle the support vectors.
7. [1pt] Draw the decision boundary separating of the separating hyperplane, in the original $\mathbb{R}^1$ feature space.
8. [3pt] Compute the coefficients α and the constant b in Equation 1 for the kernel k and the support vectors $SV = \{u_1, u_2\}$ you chose in part 6. Be sure to explain how you obtained these coefficients.

$$y(x) = \text{sign}\left(\sum_{n=1}^{|SV|} \alpha_n y_n k(x, u_n) + b\right) \tag{1}$$

   Think about the dual form of the quadratic program and the constraints placed on the α values.
9. [1pt] ~~Explain why there can be no more than 3 support vectors in this transformed feature space for a linearly separable data set.~~ This statement is not true so you get a free point.
10. [1pt] If we add another positive ($Y = +$) point to the training set at $X_1 = 5$ would the hyperplane or margin change? Why or why not?

2.2 Infinite Features Spaces and Kernel Magic

Lets define a new (infinitely) more complicated feature transformation $\phi_n : \mathbb{R}^1 \rightarrow \mathbb{R}^n$ given in Equation 2.

$$\phi_n(x) = \left\{ e^{-x^2/2}, e^{-x^2/2}x, \frac{e^{-x^2/2}x^2}{\sqrt{2}}, \ldots, \frac{e^{-x^2/2}x^i}{\sqrt{i!}} \ldots, \frac{e^{-x^2/2}x^n}{\sqrt{n!}} \right\} \tag{2}$$

Suppose we let $n \rightarrow \infty$ and define new feature transformation in Equation 3. You can think of this feature transformation as taking some finite feature vector and producing an infinite dimensional feature vector rather than the simple two dimensional feature vector used in the earlier part of this problem.

$$\phi_\infty(x) = \left\{ e^{-x^2/2}, e^{-x^2/2}x, \frac{e^{-x^2/2}x^2}{\sqrt{2}}, \ldots, \frac{e^{-x^2/2}x^i}{\sqrt{i!}} \ldots \right\} \tag{3}$$

1. [1pt] Can we directly apply this feature transformation to the data. Put another way, can we explicity construct $\phi_\infty(x)$? (This is nearly rhetorical and not a trick question.)

2. [1pt] Is there a finite set of points that cannot be linearly seperated in this feature space? Explain why or why not?

3. [3pt] We know that we can express a linear classifier using only inner products of support vectors in the transformed feature space as seen in Equation 1. It would be great if we could some how use the feature space obtained by the feature transformormation ϕ_∞. However, to do this we must be able to compute the inner product of examples in this infinite vector space. Lets define the inner product between two infinite vectors $a = \{a_1, \ldots, a_i, \ldots\}$ and $b = \{b_1, \ldots, b_i, \ldots\}$ as the infinite sum given in Equation 4.

$$k(a, b) = a \cdot b = \sum_{i=1}^{\infty} a_i b_i \tag{4}$$

   Can we explicity compute $k(a, b)$? What is the explicit form of $k(a, b)$? Hint you may want to use the Taylor series expansion of e^x which is given in Equation 5.

$$e^x = \lim_{n \rightarrow \infty} \sum_{i=0}^{n} \frac{x^i}{i!} \tag{5}$$

4. [1pt] With such a high dimensional feature space should we be concerned about overfitting?

3 [50 points] k-NN, SVM, and Cross-Validation (Sue Ann)

In this question, you will explore how cross-validation can be used to fit "magic parameters." More specifically, you'll fit the constant k in the k-Nearest Neighbor algorithm, and the slack penalty C in the case of Support Vector Machines. For all implementation questions, please electronically submit your source code to

`/afs/andrew.cmu.edu/course/10/701/Submit/your_andrew_id/HW3/`

and supply pseudo-code in your writeup where requested.

3.1 Dataset

1. Download the file `hw3_matlab.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. (This is an "easier" version of the dataset [1] on the project website.) For fun, try `showface(evaldata(1,:))`, `showface(evaldata(2,:))`, ...

`cosineDistance.m` implements the cosine distance, a simple distance function. It takes two feature vectors x and y, and computes a nonnegative, 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)

3.2 k-NN

1. Implement the k-Nearest Neighbor (k-NN) algorithm in Matlab. Hand in pseudo-code. Hint: You might want to precompute the distances between all pairs of points, to speed up the cross-validation later.

2. Implement n-fold cross validation for k-NN. Your implementation should partition the training data and labels into n parts of approximately equal size. Hand in the pseudo-code.

3. For $k = 1, 2, \ldots, 100$, compute and plot the 10-fold (i.e., $n = 10$) cross-validation error for the training data, the training error, and the test error. Don't forget to hand in the plot!

   How do you interpret these plots? Does the value of k which minimizes the cross-validation error also minimize the test set error? Does it minimize the training set error? Either way, can you explain why? Also, what does this tell us about using the training error to pick the value of k?

3.3 SVM

1. Now download libsvm using the link from the course website and unpack it to your working directory. It has a Matlab interface which includes binaries for Windows. It can be used on OS X or Unix but has to be compiled (requires `g++` and `make`) – see the `README` file from the libsvm zip package and/or the instructions on the course homework page.

   `hw3_matlab.zip`, which you downloaded earlier, contains files `testSVM.m` (an example demonstration script), `trainSVM.m` (for training) and `classifySVM.m` (for classification), which will show you how to use libsvm for training and classifying using an SVM. Run `testSVM`. This should report a test error of 0.4333.

   In order to train an SVM with slack penalty C on training set `data` with labels `labels`, call
   ```
   svmModel = trainSVM(data, labels, C)
   ```
   In order to classify examples `test`, call
   ```
   testLabels = classifySVM(svmModel, test)
   ```
   Train an SVM on the training data with $C = 500$, and report the error on the test set.

2. Now implement n-fold cross-validation for SVMs. Similarly to k-NN, split your training data into n roughly equal parts. Hand in the pseudo-code.

3. For $C = 10, 10^2, 10^3, 10^4, 5 \cdot 10^4, 10^5, 5 \cdot 10^5, 10^6$, compute and plot the 10-fold (i.e., $n = 10$) cross-validation error for the training data, the training error, and the test error, with the axis for C in log-scale (try `semilogx`). Don't forget to hand in the plot!

   How do you interpret these plots? Does the value of C which minimizes the cross-validation error also minimize the test set error? Does it minimize the training set error? Either way, can you explain why? Also, what does this tell us about using the training error to pick the value of C?

[1] Ryan White, Ashley Eden, Michael Maire "Automatic Prediction of Human Attractiveness", CS 280 class report, December 2003.

3.4 DIY

1. Design your favorite classifier: You have to use either k-NN or SVM, but you are allowed to use arbitrary values for k or for C. For k-NN, you can invent different distance functions than the one we gave you or you can try to weigh the influence of training examples by their distance from the test point. If you want, you can do arbitrary feature selection, e.g. you can ignore some columns. You can also perform any linear transformation of the features if you want. Whatever you do, please document it, and apply your algorithm to the `evaldata` data set. Output your class labels for this evaluation set, one label per line, in the order of the examples from the evaluation set. Submit your labels as file `evallabels_yourid.txt` where yourid is your Andrew ID.

   Submit the actual code and the predicted labels (in file `evallabels_yourid.txt`) to

   `/afs/andrew.cmu.edu/course/10/701/Submit/your_andrew_id/HW3/`

   You might find the Matlab `save` function helpful for saving your labels.

2. (Extra credit) Your labels will participate in a competition. The top submissions will receive great honor... and some extra credit.

4 Learning Theory [Steve, 15 points]

4.1 VC Dimension

In this section you will calculate the VC-dimension of some hypothesis classes. Remember that in order to prove that H has VC-dimension d you need to show that

- There exists a set of d points which can be shattered by H. (This step is often easy).
- There exists **no** set of $d+1$ points that can be shattered by H. (This step is hard).

Now find the VC-dimension of the following hypothesis classes, and include your proofs.

1. (5 points) The union of k intervals on the real line. In other words each hypothesis $h \in H$ is associated with k closed intervals $[a_i, b_i],\ i \in \{1, 2, \ldots, k\}$; and $h(x) = 1$ iff $x \in \cup_{i \in \{1,2,\ldots,k\}} [a_i, b_i]$.

2. (5 points) The set of axis aligned rectangles in the n-dimensional reals $\mathbb{R}^n$. That is, any $h \in H$ is characterized by n closed intervals $[a_i, b_i]$ for $i \in \{1, 2, \ldots, n\}$, and for any $\mathbf{x} \in \mathbb{R}^n$,

   $$h(\mathbf{x}) = 1 \text{ iff } \forall i \in \{1, 2, \ldots, n\}, \quad x_i \in [a_i, b_i].$$

   Hint: Can you always find a subset of examples such that labeling those points 1 forces all the other examples to be labeled 1?

4.2 Sample Complexity

In this part, you will use sample complexity bounds to determine how many training examples are needed to find a good classifier.

- (5 points) Let H be the hypothesis class of linear separators. Recall that the VC dimension of linear separators in $\mathbb{R}^n$ is $n + 1$. Suppose we sample a number m of training examples i.i.d. according to some unknown distribution $\mathcal{D}$ over $\mathbb{R}^2 \times \{-1, 1\}$.

  $$(X_1, Y_1), (X_2, Y_2), \ldots, (X_m, Y_m) \sim \mathcal{D}$$

  Prove that if $m \geq 14619$, then with probability at least .99 over the draw of the training examples, the linear separator with smallest training error $\hat{h}_{\text{ERM}} = \arg\min_{h \in H} \text{error}_{train}(h)$ has

  $$\text{error}_{true}(\hat{h}_{\text{ERM}}) - \text{error}_{train}(\hat{h}_{\text{ERM}}) \leq .05$$

  You may not assume $\text{error}_{train}(\hat{h}_{\text{ERM}}) = 0$. You may use any formulas from the lecture slides, textbook, or readings from the website, but please tell us where you found the formula(s) you use.