CSCI567 Fall 2013 - Assignment 1
Due date: Sept. 18 , 2013

September 5, 2013

Algorithm Component

1 Analysis on K-NN

The computational complexity of classifying a data point $\boldsymbol{x} \in \mathbb{R}^D$, using K-NN classifier on a training data set of N samples is $O(ND)$. In this part, we consider ideas of reducing the complexity. Broadly speaking, there are two strategies, reducing N or reducing D.

1.1 Question 1. (2 point)

Consider a ball centered at $\boldsymbol{x}_c$ with the radius r. This ball $\mathcal{B}(\boldsymbol{x}_c, r)$ is defined as

$$\mathcal{B}(\boldsymbol{x}_c, r) = \{\boldsymbol{x} : \|\boldsymbol{x} - \boldsymbol{x}_c\|_2 \leq r\}$$

i.e., the set of all $\boldsymbol{x}$ such that its distance to the center is less or equal to r. Let us use this ball to cover the training data. Suppose there are N_c data points within the ball. Namely,

$$\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{N_c} \in \mathcal{B}(\boldsymbol{x}_c, r)$$

Suppose I have a testing point $\boldsymbol{x}$. I can compute its distance to the center of the ball as

$$d(\boldsymbol{x}, \boldsymbol{x}_c) = \|\boldsymbol{x} - \boldsymbol{x}_c\|_2$$

Use the above quantity and the triangle inequality of distances to derive an upper bound and an lower bound on the distances between $\boldsymbol{x}$ and any $\boldsymbol{x}_i$ for $i = 1, 2, \ldots N_c$

$$\forall \quad i, \quad l_c \leq d(\boldsymbol{x}, \boldsymbol{x}_i) \leq u_c$$

The bounds l_c and u_c should depend **only** on $d(\boldsymbol{x}, \boldsymbol{x}_c)$ and r.

1.2 Question 2. (2 points)

Note that in Question 1, you are able to *estimate* the distances to all the points in the ball by calculating only its distance to its center. We are going to use this idea one step further to save computational cost for classifying with K-NN.

Now, consider another ball $\mathcal{B}(\boldsymbol{x}_a, r)$. Suppose you also derive the upper and the lower bounds for the testing point $\boldsymbol{x}$'s distances to all training samples inside this ball

$$\forall \quad j, \quad l_a \leq d(\boldsymbol{x}, \boldsymbol{x}_j) \leq u_a$$

where $\boldsymbol{x}_j$ indexes all training samples inside the ball $\mathcal{B}(\boldsymbol{x}_a, r)$.

Obviously, if $u_c < l_a$, then none of $\boldsymbol{x}_j$ can be $\boldsymbol{x}$'s K nearest neighbors, provided $N_c \geq K$.

Derive the condition such that $u_c < l_a$. The condition should depend only on $\boldsymbol{x}$, $\boldsymbol{x}_c$, $\boldsymbol{x}_a$ and r.

1.3 Question 3. (10 points)

Now in Question 2, you have shown that, if two balls satisfy a certain condition, you will be able to narrow down your search for $\boldsymbol{x}$'s K nearest neighbors to points inside only one ball — you do not even have to compute distances to points inside the other ball!

Following that, sketch an idea using your results from Question 1 and 2 so far to speedup K-NN. The strategy should consider the following: i) use a lot of balls to cover training samples; ii) compute distances to centers of those balls; iii) narrow down $\boldsymbol{x}$'s nearest neighbors to points insider a few balls, instead of searching over all training samples.

Your sketch of the idea should tell us how you want to place balls (you have choices to place balls anywhere you want) to maximize your chance such that the condition in Question 2 is satisfied. You should also tell us what to do if the condition is not satisfied. You should also discuss how to choose r. What if r is too small? What if r is too big?

Note that, your idea can be sketchy but needs to be sensible, i.e., not completely baseless. But you do not have to prove its correctness and optimally efficiency in reducing computational complexity rigorously. You are also required to write down at most **200** words for the answer.

1.4 Question 4. (10 points)

Question 1- 3 lead to a strategy in reducing the amount of computation by trying reducing the number of training examples to which we have to compute the distances. Now, we are going to devise another algorithm that try to reduce the amount of computation by reducing the dimensionality of data.

Suppose our training data is $\{(\boldsymbol{x}_n, y_n), n = 1, 2, \ldots, N\}$ where $\boldsymbol{x}_n \in \mathbb{R}^D$. Suppose D is very big. This presents an issue in both computing distances and storage cost. We want to modify the data.

Concretely, we will project $\boldsymbol{x}_n$ to z_n where $z_n = \boldsymbol{p}^{\mathrm{T}}\boldsymbol{x}_n$. $\boldsymbol{p} \in \mathbb{R}^D$ is a unit-length vector such that $\|\boldsymbol{p}\|_2 = 1$. We will also project $\boldsymbol{x}$ into $z = \boldsymbol{p}^{\mathrm{T}}\boldsymbol{x}$. Now, instead of looking for nearest neighbors in $\mathbb{R}^D$, we will look for nearest neighbors of z among $\{(z_n, y_n), n = 1, 2, \ldots, N\}$. Since these points are just scalar, it is foreseeable that computing distances among them is easy.

How to choose $\boldsymbol{p}$? Obviously, we cannot be arbitrary. Otherwise, nearest neighbor relationship of $\boldsymbol{x}$ among $\{(\boldsymbol{x}_n, y_n), n = 1, 2, \ldots, N\}$ would not be preserved in our simplified space of z and $\{(z_n, y_n), n = 1, 2, \ldots, N\}$. Thus, we should choose $\boldsymbol{p}$ such that pairwise distances among $\boldsymbol{x}_n$ are preserved as much as possible among z_n. Namely,

$$\min_{\boldsymbol{p}} \sum_{m,n} \left[\|\boldsymbol{x}_m - \boldsymbol{x}_n\|_2^2 - (\boldsymbol{p}^{\mathrm{T}}\boldsymbol{x}_m - \boldsymbol{p}^{\mathrm{T}}\boldsymbol{x}_n)^2\right]$$

Derive $\boldsymbol{p}$. (While $\boldsymbol{p}$ cannot be solved analytically, your correct derivation will result in a matrix eigendecomposition — finding eigenvalues and eigenvectors of a symmetric matrix is considered as a solved problem even if it does not give analytic forms.)

1.5 Question 5 (10 points)

In the lectures, I have mentioned that nearest neighbor classifiers have strong theoretical guarantees. We gave a proof sketch but we did not prove formally. In what follows, we prove it rigorously that the expected risk of our nearest neighbor classifier $R(f)$ is at most twice the expected risk of the Bayes optimal classifier $R(f^*)$.

We first state the following lemma. You do not need to prove it.

Lemma. Let $\eta(x)$ denote $p(y = 1|\boldsymbol{x})$ and let $\boldsymbol{x}(1)$ denote $\boldsymbol{x}$'s nearest neighbor in the training dataset $\mathcal{D} = \{(\boldsymbol{x}_n, y_n), n = 1, 2, \cdots, N\}$. When $N \to +\infty$, $\eta(\boldsymbol{x}(1)) \to \eta(\boldsymbol{x})$. In other words, the posterior probability $p(y = 1|\boldsymbol{x}(1))$ converges to $p(y = 1|\boldsymbol{x})$. In words, when there are an infinite number of data points, $\boldsymbol{x}$ and its nearest neighbor are indistinguishable in distribution.

We are now ready to proceed.

5.1 (Corresponding to Step 2 in the lecture slides.) We will compute the mistake made by our nearest neighbor classifier on $\boldsymbol{x}$.

Our classifier will make a mistake if $\boldsymbol{x}$ and $\boldsymbol{x}(1)$ have different labels — after all, we label $\boldsymbol{x}$ according to $\boldsymbol{x}(1)$'s label. There are two scenarios:

- $\boldsymbol{x}$'s label is 1 while $\boldsymbol{x}(1)$'s label is 0
- $\boldsymbol{x}$'s label is 0 while $\boldsymbol{x}(1)$'s label is 1

Write down the two probabilities corresponding to the two cases, respectively. Your probabilities should depend on $\eta(\boldsymbol{x})$ and $\eta(\boldsymbol{x}(1))$ only. Note that you should take into consideration that $\boldsymbol{x}$ and $\boldsymbol{x}(1)$ are independently sampled from the unknown distribution. So their probabilities of having labels 1 or 0 are independent.

5.2 (Corresponding to Step 3 in the lecture slides.) We now compute the expected conditional risk $R(f, \boldsymbol{x})$ for our classifier. The risk is the expected penalty when we make mistakes. The penalty is 1 (every time a mistake is made, we pay 1 buck). Thus, the expected conditional risk is

$R(f, \boldsymbol{x}) = 1\times$probability of making mistakes$+0\times$probability of not making mistakes = probability of making mistakes

Using the results from 5.1, write down $R(f, \boldsymbol{x})$ in terms of $\eta(\boldsymbol{x})$ and $\eta(\boldsymbol{x}(1))$.

Now, in light of the Lemma, you will realize that $\eta(\boldsymbol{x}(1))$ is the same as $\eta(\boldsymbol{x})$. Then, you will realize

$$R(f, \boldsymbol{x}) = 2\eta(\boldsymbol{x})(1 - \eta(\boldsymbol{x}))$$

5.3 (Corresponding to Step 4 in the lecture slides.) We now compute the expected conditional risk for the Bayes optimal classifier. Prove that

$$R(f*, \boldsymbol{x}) = \min\{\eta(\boldsymbol{x}, 1 - \eta(\boldsymbol{x})\}$$

Concretely, analyze two cases

- If $\eta(\boldsymbol{x}) > 1 - \eta(\boldsymbol{x})$, then $f^*(\boldsymbol{x}) = 1$ (according to the classifier's rule). In this case, what is the chance the classifier makes a mistake?
- If $\eta(\boldsymbol{x}) < 1 - \eta(\boldsymbol{x})$, then $f^*(\boldsymbol{x}) = 1$. In this case, what is the chance the classifier make a mistake?

One of the two cases has to be true. Thus, using these two probabilities to show the chance the classifier f^* makes the mistake is indeed

$$\min\{\eta(\boldsymbol{x}, 1-\eta(\boldsymbol{x})\}$$

Thus, the expected conditional risk is the expected penalty under the chance the classifier makes a mistake, which will be

$$R(f*, \boldsymbol{x}) = \min\{\eta(\boldsymbol{x}, 1-\eta(\boldsymbol{x})\}$$

5.4 (Corresponding to step 5 in the lecture notes Now, consider all the above results to show that

$$R(f, \boldsymbol{x}) = 2R(f^*, \boldsymbol{x})(1 - R(f^*, \boldsymbol{x}))$$

5.5 (corresponding to "one more step" in the lecture notes) We need to compute the expected risks under the different classifiers. Identify the missing item in the following derivation,

$$\begin{aligned}
R(f) &= \mathbb{E}_{\boldsymbol{x}} R(f, \boldsymbol{x}) \\
&= \mathbb{E}_{\boldsymbol{x}} 2R(f^*, \boldsymbol{x})(1 - R(f^*, \boldsymbol{x})) \\
&= 2\mathbb{E}_{\boldsymbol{x}} R(f^*, \boldsymbol{x}) - 2\mathbb{E}_{\boldsymbol{x}} R(f^*, \boldsymbol{x})^2 \\
&= 2R(f^*) - [2(R(f^*))^2 + \text{missing item}] \\
&= 2R(f^*)(1 - R(f^*)) - \text{missing item} \\
&\leq 2R(f^*)(1 - R(f^*))
\end{aligned}$$

Also, explain why in the last step, we change from equality to inequality by dropping the missing item?

Submission instruction

Your answers must be typeset with LaTeXand generated as a PDF file. No handwritten submission will be permitted. A template is provided for you. Please consult it for additional instructions. There are many free integrated LaTeXeditors that are convenient to use, please google search them and choose the one(s) you like the most. This `http://www.andy-roberts.net/writing/latex` seems like a good tutorial.

Due Date and Time

3:15pm, Wednesday, Sept. 18, 2013. You need to logon to Blackboard (for the time being) to submit your solutions. Your PDF file should be named as abcd_hw1_AB.pdf where abcd is the last 4 digits of your USC ID and AB are the initials of your first and last names

A single two-day extension or two one-day extensions are allowed. You need to apply for it by Tuesday Sept 17 2013, 3:15pm and get my approval. Please send your application to me via email. Late submissions will not be accepted if not previously approved. My email address is `feisha@usc.edu`

Collaboration

You may collaborate. However, you need to write your own solution and submit separately. You also need to list with whom you have discussed. For any team, at most 4 members can participate. You cannot participate more than 1 team.