COMS 4771 Fall 2016 Homework 3 solutions

Problem 1

(a) No for centering, yes for standardization.

Centering will shift all the estimated means of class conditionals by the same amount; classification is based on Euclidean distances, which are invariant to translation.

Here is an example where standardization affects the learning algorithm. The data set is comprised of four examples.

- Example 1: $x = (-10^6 - 1, 1)$, $y = 1$
- Example 2: $x = (10^6 - 1, 1)$, $y = 1$
- Example 3: $x = (-10^6 + 1, -1)$, $y = 0$
- Example 4: $x = (10^6 + 1, -1)$, $y = 0$

Now consider what happens to the learned classifier on the test point $(1, 1/2)$.

(b) No for centering, yes for standardization.

Classification is based on Euclidean distances, which are invariant to translation.

Here is an example where standardization affects the learning algorithm. The data set is comprised of four examples.

- Example 1: $x = (-1, 10^6)$, $y = 1$
- Example 2: $x = (-1, -10^6)$, $y = 1$
- Example 3: $x = (1, 0)$, $y = 0$
- Example 4: $x = (1, 0)$, $y = 0$

Now consider what happens to the 1-NN classifier on the test point $(1, 10^6)$.

(c) No for centering, no for standardization.

The set of possible splits along coordinates remains the same after either pre-processing step.

(d) No for centering, no for standardization.

Centering and standardization are invertible affine transformations, so the set of possible linear classifiers remains the same after either pre-processing step.

Problem 2

The following are results when using just 200000 training data.

Five-fold cross validation error rates:

- Averaged Perceptron with unigram features: 0.11296
- Averaged Perceptron with tf-idf features: 0.12408
- Averaged Perceptron with bigram features: 0.09887

So we would select Averaged Perceptron with bigram features.

Training error rates (for classifier learned using all 200000 training data):

- Averaged Perceptron with unigram features: 0.10322
- Averaged Perceptron with tf-idf features: 0.08602
- Averaged Perceptron with bigram features: 0.06740

Test error rates (for classifier learned using all 200000 training data):

- Averaged Perceptron with unigram features: 0.11178
- Averaged Perceptron with tf-idf features: 0.12122
- Averaged Perceptron with bigram features: 0.09753

The cross validation error rates turn out to be very similar to the test error rates, whereas the training error rates are much smaller.

The following are results when using all 1000000 training data.

Five-fold cross validation error rates:

- Averaged Perceptron with unigram features: 0.10565
- Averaged Perceptron with tf-idf features: 0.11176
- Averaged Perceptron with bigram features: 0.08811

Training error rates (for classifier learned using all 1000000 training data):

- Averaged Perceptron with unigram features: 0.10119
- Averaged Perceptron with tf-idf features: 0.09339
- Averaged Perceptron with bigram features: 0.06903

Test error rates (for classifier learned using all 1000000 training data):

- Averaged Perceptron with unigram features: 0.10494
- Averaged Perceptron with tf-idf features: 0.11005
- Averaged Perceptron with bigram features: 0.08676

Problem 3

(a) $\hat{\sigma}^2 = \frac{1}{nd}\sum_{i=1}^{n} \|x_i - \bar{\mu}\|_2^2$, where $\bar{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i$.

(b) If $y = 0$, then $\hat{\theta} = 2x$; if $y = 1$, then $\hat{\theta} = x$. A compact way to write this is $\hat{\theta} = 2x/(1+y)$.

This problem turns out to be more involved than originally intended. The likelihood of θ given (x, y) is 0 if $\theta < x$, and is $x^y(\theta - x)^{1-y}/\theta^2$ if $\theta \geq x$. Therefore, the log-likelihood of θ is $-\infty$ if $\theta < x$, and is $y\ln(x) + (1-y)\ln(\theta - x) - 2\ln(\theta)$ if $\theta \geq x$. The usual way we try to maximize the log-likelihood is by finding when its derivative (with respect to θ) is equal to zero. But this does not work because the log-likelihood function is not concave on its domain[1]. Fortunately, the log-likelihood function is still relatively simple, and we can analytically determine its maximizer.

First, suppose $y = 0$. In this case, the function is $\ln(\theta - x) - 2\ln(\theta)$ (for $\theta \geq x$). The derivative of this function is zero only when $\theta = 2x$. However, we need to check if $\theta = 2x$ is a local maximizer, a local minimizer, or an inflection point. This is done by examining the second-derivative of the function. The second-derivative of the function at $\theta = 2x$, so it is a local maximizer. Since $\theta = 2x$ is the only stationary point and is a local maximizer, we can conclude that it is the global maximizer over all $\theta \geq x$. Fortunately, $\theta = 2x$ is also a positive integer, so it is also the global maximizer over the positive integers. So the MLE in this case is $\hat{\theta} = 2x$.

Now, suppose instead that $y = 1$. In this case, the function is $\ln(x) - 2\ln(\theta)$. This is a convex function of θ on its domain. That would be great if we were trying to minimize the function, but we are trying to maximize it. Fortunately, it is clear that the function is strictly decreasing on its domain. So the maximizer must be at the left boundary, which is $\theta = x$ (again, a positive integer). So the MLE in this case is $\hat{\theta} = x$.

Somewhat conicidentally, if you had ignored the fact that the log-likelihood function is $-\infty$ when $\theta < x$, and simply determined the point where the derivative of the log-likelihood is equal to zero, you would have obtained a formal expression of $\theta = 2x/(1+y)$.

[1] Here, we say the *domain* of a function is the set of points on which the function is finite.