10-701 Machine Learning, Fall 2012: Homework 2

Due Monday 10/29 at the beginning of class.

Instructions: Remember to submit your solution to each problem separately in different piles on the table at the front of the classroom at the beginning of class on the day the homework is due. At the *top* of the first page of each solution you hand in, clearly write the class number (10701), the number of the problem (i.e. "Problem 1", "Problem 2", etc.), your first and last name (assuming you have both), and your Andrew ID.

1 Learning with L1 norm (Avi)[25 pts]

Suppose you want to predict an unknown value $Y \in \mathbb{R}$, but you are only given a sequence of noisy observations $x_1, \ldots, x_n$ of Y with iid noise ($x_i = Y + \epsilon_i$).

We have seen in the last homework, that if we assume the noise is i.i.d. Gaussian($\epsilon_i \sim N(0, \sigma^2)$), finding the maximum likelihood estimate for Y is equivalent to finding the value $\hat{y}$ which minimizes the sum of the least square errors to the x's. That is to say

$$\hat{y} = \arg\min_y \sum_{i=1}^{n} (y - x_i)^2 \tag{1}$$

And there is a simple closed form solution:

$$\hat{y} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{2}$$

It was also suggested that if we assume that the noise is i.i.d. Laplace ($\epsilon_i \sim \mathit{Laplace}(0, b)$) with pdf

$$f_{\epsilon_i}(x) = \frac{1}{2b} \exp\left(-\frac{|x|}{b}\right) \tag{3}$$

we end up with a maximum likelihood estimator that is in some sense more robust. We will show this more rigorously in this part.

1 [2 pts] Begin by showing that finding the MLE for Y, assuming Laplace noise, is equivalent to finding the value $\hat{y}$ that minimizes the sum of absolute error. That is

$$L(y) = \sum_{i=1}^{n} |y - x_i| \tag{4}$$

$$\hat{y} = \arg\min_y L(y) \tag{5}$$

2 [4 pts] A standard way to minimize a loss function is to take the derivative and set it to zero. This loss function is not directly differentiable. However, it is easy to see that the function is not differentiable only where y has same value as any of the x's.

Assume that the x's are distinct and are sorted in ascending order ($\forall i, \forall j > i, x_j > x_i$).

Find an expression for the gradient $\frac{dL(y)}{dy}$ under the constraint that y lies between two consecutive values of x (That is to say $x_i < y < x_{i+1}$). (**Hint:** You may have to consider x's which are $> y$ separately from the x's which are $< y$)

3 [3 pts] Assuming that there are an even number of x's, what are all the values of y for which $\frac{dL(y)}{dy} = 0$?

4 [3 pts] If we have an odd number of x's, there is no value of y where $\frac{dL(y)}{dy} = 0$. However there is a y_0 such that $\frac{dL(y)}{dy} < 0$ if $y < y_0$ and $\frac{dL(y)}{dy} > 0$ if $y > y_0$. What is y_0?

5 [4 pts] Your answer in the last two parts are therefore the solution to $\hat{y}$. Give an explanation of what the solution represents (Hint: Its either mean, median or mode). Give a brief explanation why this solution may be more robust against outlier in the data (as compared to least square errors).

Now we will see the how L1 penalty leads to feature selection. You are given a set of points and the corresponding outputs: $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, where $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$ for $i = 1, \ldots, n$. You want to use this data to train a linear predictor $y = w^T F(x)$ where $w^T = (w_1, \ldots, w_K)$ and $F(x)^T = (f_1(x), \ldots, f_K(x))$, where K is finite. Here f_i is the ith feature for our learning problem. Consider the following objective function:

$$J(w, \lambda) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{2}(y_i - w^T F(x_i))^2 + \lambda ||w||_1 \tag{6}$$

where $||w||_1 = \sum_{i=1}^{n} |w_i|$ ($||w||_1$ is called the L1 norm of the vector w). We use our data to learn the vector w by minimizing $J(w, \lambda)$, i.e.

$$w^* = \arg \min_{w \in \mathbb{R}^k} J(w, \lambda) \tag{7}$$

The above optimization criterion typically leads to an effective feature selection by picking a large value for parameter λ. In other words if we pick a large value of λ many w_is will be 0. Therefore the corresponding feature function will be unimportant for our predictor.

6 [2 pts] Write the derivative of the first term ($\frac{1}{n} \sum_{i=1}^{n} \frac{1}{2}(y_i - w^T F(x_i))^2$) of the function wrt w_k as $a_k w_k - r_k$. (ie find a_k and r_k).

7 [7 pts] It can be shown that at optima the following conditions is satisfied:-

$$w_k^* = \begin{cases} \frac{r_i + \lambda}{a_i} & \text{if } r_k < -\lambda \\ 0 & \text{if } r_k \in [-\lambda, \lambda] \\ \frac{r_i - \lambda}{a_i} & \text{if } r_k > \lambda \end{cases} \tag{8}$$

Let's explore the relation between the regularization parameter λ, the weight of the kth parameter w_k and the quantity r_k What is the meaning of r_k (2 pt)? Provide a plot of w_i^* vs r_i. Where does λ appear in the plot (3 pt)? What can you say about when the value of w_k^* is zero (2 pts)?

2 k Nearest Neighbor and Kernel Regression (25pt, Zeyu)

In this problem, we are going to explore the relationship between k-NN and Kernel Regression. Given labeled training data of the form $(x_1, y_1), ..., (x_n, y_n)$ for $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$, kernel regression is a widely used non-parametric regression method that applies a kernel function $\mathbf{K}(\cdot)$ to smooth the data y_i:

$$\widehat{r}(x) = \frac{\sum_{i=1}^{n} \mathbf{K}(x - x_i) y_i}{\sum_{i=1}^{n} \mathbf{K}(x - x_i)} \tag{9}$$

where n is the number of data points, and $\widehat{r}(x)$ is the estimator of y (a scalar) at the point x (which is d-dimensional). There are many choices for the function $\mathbf{K}$ (not to be confused with the k in k-NN), possibly the most common one being the Gaussian kernel $\mathbf{K}(x) = e^{-\|x\|^2/\sigma^2}$. (The word "kernel" in this context has nothing to do with the kernel trick you saw in SVMs or logistic regression.) Note that in general kernel regression is a method for *regression* (as the name suggests), while k-NN is used for classification.

We can re-express $\widehat{r}$ as a weighed sum of the y_i as follows:

$$\widehat{r}(x) = \sum_{i=1}^{n} w_i(x, x_i) y_i, \text{ where } w_i(x, x_i) = \frac{\mathbf{K}(x - x_i)}{\sum_{j=1}^{n} \mathbf{K}(x - x_j)}. \tag{10}$$

In the following questions we are going to look at 1-dimensional ($d = 1$) classification data with two classes, i.e. training data are in the form of (x_i, y_i) where $x_i \in \mathbb{R}$ and $y_i \in \{0, 1\}$, for $i = 1, ..., n$.

(a) [5pts] Suppose that $x_i = i$ for all $i = 1, ..., n$, and to predict which class any new point $x_0 \in [0, n]$ belongs to, we use the decision rule

$$\widehat{y}_0 = \mathbf{I}(\widehat{r}(x_0) > 0.5),$$

where $\mathbf{I}$ is the indicator function that equals to 1 if the expression within $\mathbf{I}$ is true, and 0 if not true. Can you think of a kernel function $\mathbf{K}$ such that this decision rule gives the same answer as the k-nearest neighbor algorithm? (**Hint:** the function $\mathbf{K}(x)$ should be defined at some fixed range like $[a, b)$; you can ignore the case where x_0 is near the margins, i.e. close to 0 or n.)

(b) In general, if training data is drawn from some marginal distribution $p(x)$, we are unable to find a nice-looking kernel to simulate k-NN. One way to solve this problem is to use "locally weighted regression (LWR)" instead of kernel regression. The LWR is very similar to kernel regression except that we only calculate the weighted sum of k points near the given new point x:

$$\widehat{r}(x) = \sum_{i \text{ s.t. } x_i \in k\text{-NN of } x} w_i(x, x_i) y_i \tag{11}$$

Now consider the case of weighted k-nearest neighbor (WKNN). Weighted k-NN penalizes the vote of every point z within the k-NN range by $d(x, z)$, the distance from x to z, meaning that having a point in class j in the k-neighbor will increase the vote for this class by $1/d(x, z)$.

(b.1) [5pts] What should we put in for $w_i(x, x_i)$ so that the decision rule $\widehat{y} = \mathbf{I}(\widehat{r}(x) > 0.5)$ gives the same answer as (*un*weighed) k-NN?

(b.2) [5pts] What type of k-NN does kernel regression with an arbitrary kernel function $\mathbf{K}(x)$ simulate exactly? (Specify the value of k, whether the k-NN is weighted or non-weighted, and, if weighed, what the weights are.)

(c) [5pts] How do you modify locally weighted regression method if we want to use it to the simulate weighted k-NN algorithm with more than 2 classes?

(d) [5pts] Now consider how training and test error change when varying the kernel function. Sort the kernel functions plotted below in order from smallest to largest training error; same for *bias* of testing error; same for *variance* of testing error.

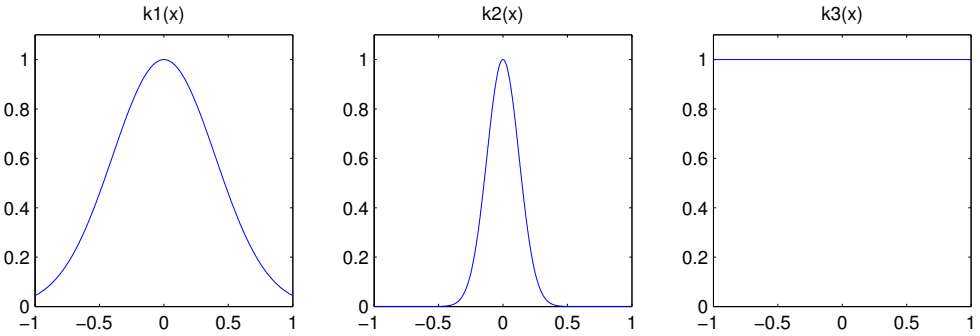


Figure 1: Kernel functions

Which one's performance is closer to 1-NN and which one's is closer to n-NN? (n is the number of data)

3 Variance and Bias Tradeoff, Model Selection [25pt, Derry]

In the following questions, assume zero Bayes error, i.e. zero noise variance.

1. **[12 points]** In class we define True Risk as Mean Squared Error between our model and the true model as:

$$R(f) = \mathbb{E}[(f(X) - Y)^2] \tag{12}$$

We also show this risk in terms of Bias-Variance Tradeoff, i.e:

$$R(f) = \mathbb{E}[(f(X) - Y)^2] = Variance + Bias^2 \tag{13}$$

where $Y = f^*(X)$ (since we assume noise free problem).

We can also define risk in terms of our estimated parameter $\hat{\theta}$ (obtained using MLE or MAP or density estimator, etc) and the true parameter θ as:

$$R(\theta, \hat{\theta}) = \mathbb{E}_\theta(\hat{\theta} - \theta)^2 = Var_\theta(\hat{\theta}) + bias^2 \tag{14}$$

where $bias = \mathbb{E}_\theta[\hat{\theta}] - \theta$

Let $X_1, ..., X_n \sim Bernoulli(p)$, be our coin-flip example where each X_i is an independent flip where $X_i = 1$ indicates flipping a head, $X_i = 0$ indicates flipping a tail, and p is the probability of getting a head.

Consider two estimators for p, $\hat{p}_1 = \frac{1}{n}\sum_i X_i$ (the MLE estimate) and $\hat{p}_2 = \frac{\sum_i X_i + \alpha}{\alpha + \beta + n}$ (the mean of the posterior Beta distribution $P(p|D)$ when we use $Beta(\alpha, \beta)$ as prior).

(a) **[1 point]** Compute the risk of $\hat{p}_1$, i.e. $R(p, \hat{p}_1)$

(b) **[4 point]** Compute the risk of $\hat{p}_2$, i.e. $R(p, \hat{p}_2)$

(c) **[2 point]** Which estimator $\hat{p}_1$ or $\hat{p}_2$ that you will prefer when there is less data and which will you prefer when there is more data? (Hint: consider bias-variance tradeoff)

(d) **[3 point]** Given a particular n, find the value of α and β that will make the risk of $\hat{p}_2$ constant.

(e) **[2 point]** Using Hoeffding's inequality, and knowing that $\mathbb{P}(0 \leq X_i \leq 1) = 1$, find an upper bound of $|\hat{p}_1 - p|$ with a probability of at least $1 - \gamma$.

Note: Whenever appropriate, give the answer in terms of α, β, γ, p, and n.

2. **[8 points]** Consider the case when X_i's have continuous value, and i.i.d according to a probability density g; i.e. $X_1, ..., X_n \sim g$.

   Let $\hat{g}$ denotes some estimator of g. The risk $R(g, \hat{g})$ in this case can be expressed as $R(g, \hat{g}) = \mathbb{E}[L(g, \hat{g})]$ where

$$L(g, \hat{g}) = \int (\hat{g}(x) - g(x))^2 dx \tag{15}$$

   (a) **[3 point]** Given $\tilde{R}(g, \hat{g}) = \mathbb{E}[\tilde{L}(g, \hat{g})]$ where

$$\tilde{L}(g, \hat{g}) = \int (\hat{g}(x))^2 dx - 2 \int \hat{g}(x) g(x) dx \tag{16}$$

   Show that minimizing $\tilde{R}(g, \hat{g})$ over $\hat{g}$ is equivalent to minimizing $R(g, \hat{g})$.

   (b) **[5 point]** Given a second sample used as validation set, $V_1, ..., V_n \sim g$, we define the risk on this validation set as

$$\hat{R}(g, \hat{g}) = \int (\hat{g}(x))^2 dx - \frac{2}{n} \sum_{i=1}^{n} \hat{g}(V_i) \tag{17}$$

   where $\hat{g}$ is still based on $X_1, ..., X_n$.

   Show that $\mathbb{E}[\hat{R}(g, \hat{g})] = \tilde{R}(g, \hat{g})$. Hence, $\hat{R}(g, \hat{g})$ can be used as an estimate of the risk.

3. **[5 points]** In this problem you will implement L1 regularization to logistic regression. Use a step size around .0001.

   The training set for this task is given at `http://www.cs.cmu.edu/~epxing/Class/10701/hw2-train.csv`. The test set is given at `http://www.cs.cmu.edu/~epxing/Class/10701/hw2-test.csv`. The data is comma-separated (no header), with the first column being the class name. There are 2 classes: 0 and 1. Each feature can take a value: 1, 2, or 3.

   We will use cross validation to select the model class (i.e., the appropriate weight (λ) for the regularizer) which has the smallest empirical error on the validation set.

   Use **Leave-One-Out** cross validation on the training data to pick appropriate weight (λ) **between 0 and 50** for the regularizer.

(a) **[2 points]** If there are more than one values of λ that minimizes the empirical error on the validation set, i.e.

$$\hat{\lambda} = argmin_{\lambda} \frac{1}{K} \sum_{k=1}^{K} \hat{R}_{V_k}(\hat{f}_{k,\lambda}) \tag{18}$$

Which value of λ will you pick (the largest? the smallest?) and why?

(b) **[1 points]** Based on your answer to the previous question and your experiment result, what is the value of λ you will select for the test set?

(c) **[1 points]** What is the empirical error on the validation set?

(d) **[1 points]** What is the empirical error on the test set?

Submit your code (zipped and named with your andrew ID or your email, in case you do not have an andrew ID) via email to dwijaya@andrew.cmu.edu.

4 SVMs [25pt, Martin]

1. **[6 points]** The Radial Basis Function (a.k.a. Gaussian) kernel with inverse width $\kappa > 0$ is defined as

$$K(\mathbf{u}, \mathbf{v}) = e^{-\kappa \|\mathbf{u}-\mathbf{v}\|^2}.$$

In Figure 2 we have plotted the decision boundaries and margins for SVM learned on the same data set using the following parameters (not in the same order as the figures):

i. Linear kernel, $C = 10$;

ii. Linear kernel, $C = 1$;

iii. Linear kernel, $C = 0.1$;

iv. RBF kernel with $\kappa = 1$, $C = 3$;

v. RBF kernel with $\kappa = 10$, $C = 1$;

vi. RBF kernel with $\kappa = 0.1$, $C = 15$.

Match each one of the figures with one of these parameter settings. Explain your matchings in one or two words each.

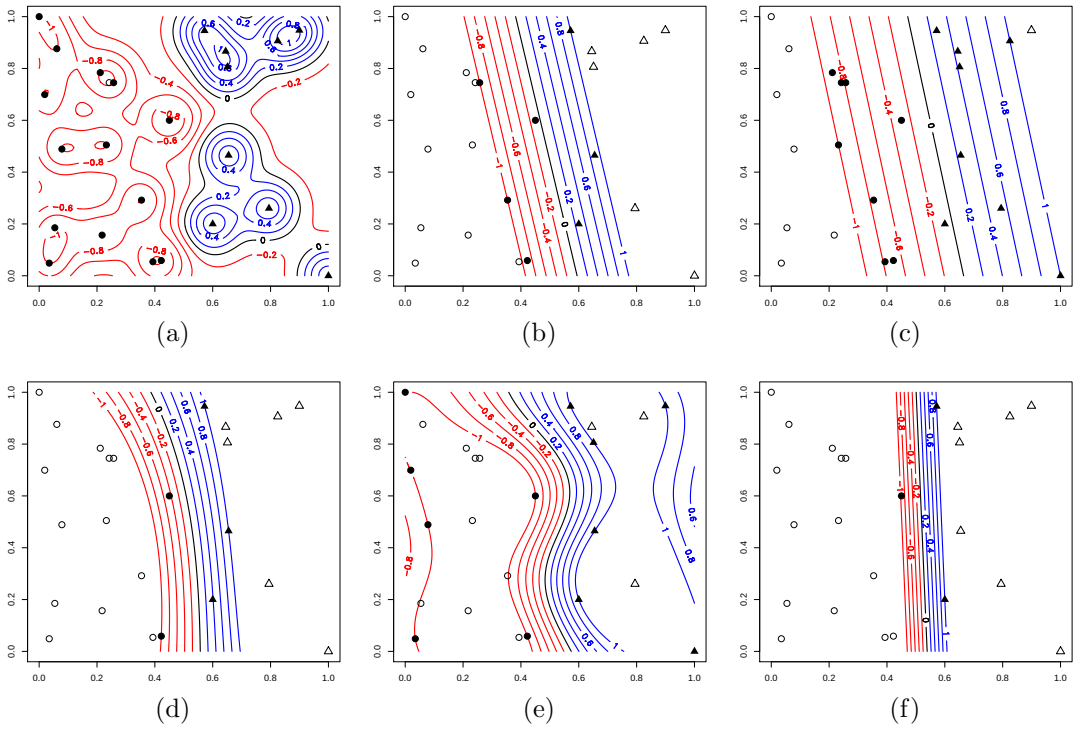
2. **[3 points]** In Figure 2e, some of the support vectors for each class are far away from any points from the other class. For example, there are four support vectors from Class 1 near the leftmost edge of the plot (corresponding to small values for the coordinate plotted on the horizontal axis), even though there are no points from Class 2 nearby. Explain why.

3. **[16 points]** Recall that we can restate the non-kernelized version of SVM as minimizing the sum of hinge losses per sample, defined as $\text{Loss}_{\text{SVM}}(f(x_i), y_i) = (1-(w \cdot x_i + b)y_i)_+$, regularized by $w \cdot w$:

$$\min_{w,b} \; w \cdot w + C \sum_i \text{Loss}_{\text{SVM}}(f(x_i), y_i)$$

where (x_i, y_i) are pairs of points and labels, with $y_i \in \{-1, +1\}$.

Figure 2: SVM decision boundaries and margins learned on the same data set for several parameter settings. Circles and triangles denote Class 1 and 2 respectively, solid points are support vectors.



Suppose you are given a data set $(x_1, y_1), ..., (x_n, y_n)$ with $n = 2000000$, where for $i = 1, ..., n/2$ we have $x_i = 0$ and $y_i = -1$, and for $i = n/2 + 1, ..., 2n$ we have $x_i = 2$ and $y_i = +1$. In other words, you are given 1 million copies of the point 0 (in one dimension, of course), all labeled -1, and 1 million copies of the point 2, all labeled $+1$.

(a) [7 points] Find w and b to minimize the SVM objective for this data, assuming $C = 1$.

What is the decision boundary and the margin of the resulting SVM? (How would the answer change if we changed C to be progressively smaller, tending to 0?)

(b) [7 points] Now suppose in addition to those $n = 2000000$ data points, we were also given an $n + 1$'th point:

$$x_{n+1} = 100, \quad y_{n+1} = -1.$$

What are the new optimal values of w and b (still using $C = 1$)?

(c) [2 points] Intuitively, how is the behavior of the SVM in part (b) different from what would happen if we used logistic regression instead?