10-701/15-781, Machine Learning: Homework 2

Aarti Singh
Carnegie Mellon University

- The assignment is due at 10:30 am (beginning of class) on **Wed, Oct 13, 2010**.
- Separate you answers into five parts, one for each TA, and put them into 5 piles at the table in front of the class. Don't forget to put both your name and a TA's name on each part.
- If you have question about any part, please direct your question to the respective TA who designed the part (however send your email to 10701-instructors@cs list).

1 Linear Regression [Leman, 20 points]

Assume that there are n given training examples $(X1, Y1), (X2, Y2), \ldots, (X_n, Y_n)$, where each input data point X_i has m real valued features. The goal of regression is to learn to predict Y from X.

The linear regression model assumes that the output Y is a linear combination of the input features X plus noise terms ϵ from a given distribution with weights given by β.

We can write this in matrix form by stacking the datapoints as the rows of a matrix X so that x_{ij} is the j-th feature of the i-th datapoint. Then writing Y , β and ϵ as column vectors, we can write the matrix form of the linear regression model as:

$$Y = X\beta + \epsilon$$

where

$$\mathbf{Y} = \begin{bmatrix} Y_1 \\ \vdots \\ Y_n \end{bmatrix}, \epsilon = \begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_n \end{bmatrix}, \beta = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_m \end{bmatrix}, \text{ and } X = \begin{bmatrix} X_{11} & X_{12} & \ldots & X_{1m} \\ X_{21} & X_{22} & \ldots & X_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ X_{n1} & X_{n2} & \ldots & X_{nm} \end{bmatrix}$$

Linear regression seeks to find the parameter vector β that provides the best *fit* of the above regression model. One criteria to measure fitness, is to find β that minimizes a given loss function $J(\beta)$.

In class, we have shown that if we take the loss function to be the square-error, i.e.:

$$J(\beta) = \sum_i (Y_i - X_i^T\beta)^2 = (X\beta - Y)^T(X\beta - Y)$$

Then

$$\hat{\beta} = (X^TX)^{-1}X^TY$$

Moreover, we have also shown that if we assume that $\epsilon_1; \ldots; \epsilon_N$ are IID and sampled from the same zero mean Gaussian that is, $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$, then the least square estimate is also the MLE estimate for $P(Y|X; \beta)$.

Now, let $\hat{Y}$ denote the vector of predictions using $\hat{\beta}$. If we were to plug in the original training set X:

$$\hat{Y} = X\hat{\beta} = X(X^TX)^{-1}X^TY$$

As mentioned above, $\hat{\beta}$, also minimizes the sum of squared errors:

$$SSE = \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2$$

1. **[7 points] Robust Linear Regression** When we perform least squares linear regression, we make certain idealized assumptions about the vector of errors ϵ, namely, that it is distributed $\mathcal{N}(0, \sigma^2)$. In practice departures from these assumptions occur. Particularly, in cases where the error distribution is heavier tailed than the Normal distribution (i.e. has more probability in tails than the Normal), the least square loss is sensitive to outliers and hence robust regression methods are of interest.

   The problem with the least square loss in the existence of outliers (i.e. when the noise term ϵ_i can be arbitrarily large), is that it weights each observation equally in getting parameter estimates. Robust methods, on the other hand, enable the observations to be weighted *unequally*. More specifically, observations that produce large residuals are down-weighted by a robust estimation method.

   In this problem, you will assume that $\epsilon_1; \ldots; \epsilon_m$ are independent and identically distributed according to a Laplacian distribution (rather than according to $\mathcal{N}(0, \sigma^2)$). That is, each $\epsilon_i \sim Laplace(0, b) = \frac{1}{2b} exp(-\frac{|\epsilon_i|}{b})$.

   (a) **[4 points]** Provide the loss function $J_{Laplace}(\beta)$ whose minimization is equivalent to finding the MLE of β under the above noise model.

   (b) **[3 points]** Why do you think that the above model provides a more robust *fit* to data compared to the standard model assuming Gaussian distribution of the noise terms?

2. **[7 points] Regularization**

   When the number of features m is much larger than the number of training instances n (i.e. $m \gg n$), the matrix $X^T X$ is not full rank and thus can not be inverted. Therefore, instead of minimizing $J(\beta)$ we minimize the following loss function:

$$J_R(\beta) = \sum_i (Y_i - X_i^T \beta)^2 + \lambda \sum_{j=1}^{M} \beta_j^2 = (X\beta - Y)^T (X\beta - Y) + \lambda \|\beta\|^2 \tag{1}$$

   We have seen in class that the solution of the above formulation is $\hat{\beta} = (X^T X + \lambda I)^{-1} X^T Y$, and for a proper value of λ, $(X^T X + \lambda I)$ is full rank and can be inverted. So for each λ, we have a solution. In other words, λ traces out a path of solutions.

   In this problem you will study the interpretation of regularization.

   (a) **[4 points]** Instead of viewing β as an unknown deterministic parameter, we can consider β as a random variable whose value is also unknown. In this setting, we then specify a prior distribution $P(\beta)$ on β that expresses our prior beliefs over the parameters. Then we estimate β using the MAP (maximum a posteriori) estimate as:

$$\beta_{MAP} = argmax_\beta \prod_{i=1}^{n} P(Y_i | X_i; \beta) P(\beta) \tag{2}$$

   Show that maximizing Equation 2 can be expressed as minimizing Equation 1 given a Gaussian prior on β (i.e. $P(\beta) \sim \mathcal{N}(0, I\sigma^2/\lambda)$). That is, show that the L2-norm regularization in the linear regression model is effectively imposing a Gaussian prior assumption on the unknown parameter β.

   (b) **[3 points]** What is the probabilistic interpretation if $\lambda \to 0$? How about if $\lambda \to \infty$? *Hint*: Consider how the prior $P(\beta) \sim \mathcal{N}(0, I\sigma^2/\lambda)$ is affected by changing λ.

3. **[6 points] LOOCV using Linear Regression** In class, you learned about using cross validation as a way to estimate the true error of a learning algorithm. The preferred solution is *Leave-One-Out Cross Validation* (LOOCV), which provides an almost unbiased estimate of this true error. In this problem,

you will derive the time complexity for computing the leave-one-out cross validation error for linear regression using Singular Value Decomposition.

Recall that the Leave-One-Out Cross Validation score is defined to be:

$$LOOCV = \sum_{i=1}^{n} (Y_i - \hat{Y}_i^{(-i)})^2$$

where $\hat{Y}_i^{(-i)}$ is the estimator of Y after removing the i-th observation (i.e., it minimizes $\sum_{j \neq i} (Y_j - \hat{Y}_j^{(-i)})^2$.

Assume that you have a black-box implementation of the Singular-Value-Decomposition (SVD) and for a given n-by-m matrix X, it returns three matrices U (n-by-m), a digonal matrix Σ (m-by-m) with non-zero diagonal entries, and V (m-by-m) such that $X = U\Sigma V^T$, where X is rank m.

Using the given SVD package for the inversion of the $X^T X$ matrix, what is the complexity of computing the LOOCV score?

Note 1: LOOCV loops through each point, performing a regression on the $n - 1$ remaining points at each iteration.

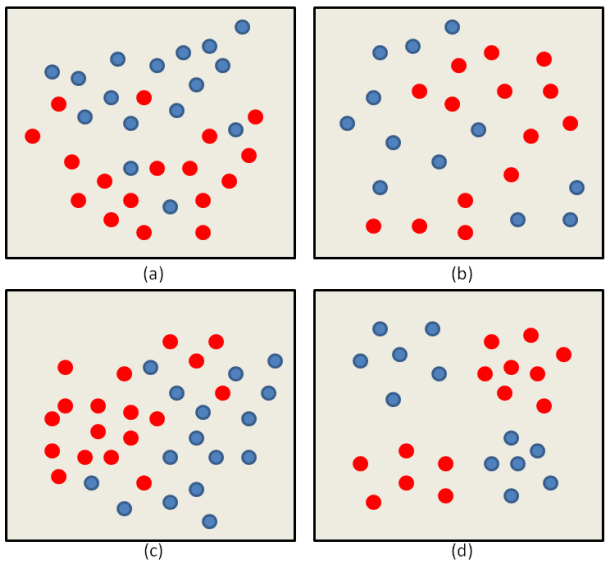
Note 2: Assume $n \gg m$, i.e. $X^T X$ is full-rank.

Note 3: The complexity of SVD for a n-by-m matrix is O(min(nm^2,mn^2)).

2 KNN [TK, 20 points]

2.1 [10 points] Decision Boundaries and Gedankenexperimente

In this problem you will conduct several gedankenexperimente (thought experiments) to understand some properties of the KNN classifier. Consider the following four samples:



where colors indicate class labels. Each sample is typical of an underlying distribution.

1. [4 points] Plot the decision boundaries[1] of the 1NN classifier for the four samples.
2. [2 points] Imagine you repeat drawing training data points from the four distributions represented by these four samples, and look at the decision boundaries given by the KNN classifier. Obviously the decision boundaries will be different across different training samples; an interesting question then is how sensitive the decision boundaries are to the training data. More specifically, how does the number of neighbors used in KNN affect the variability of the decision boundary?
3. [4 points] In addition to KNN, you also train linear classifiers, such as Logistic Regression (LR), on data drawn from the four distributions. Again, you repeatedly draw random samples, train classifiers, and compute prediction error. For each of the four distributions, decide whether LR or KNN would perform better on average, and justify your answer using conceptual arguments.

2.2 [10 points] Downside of KNN

Consider n sample points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ independently and uniformly drawn from a p-dimensional zero-centered unit ball $B := \{\mathbf{x} \mid \sqrt{\mathbf{x}^\top \mathbf{x}} \leq 1, \mathbf{x} \in \mathcal{R}^p\}$. In this problem you will study the size of the 1-nearest neighborhood of the origin $\mathbf{0}$ and how it changes with respect to the dimension p, thereby gain intuition about the downside of KNN in high dimension. More precisely, consider the distance from $\mathbf{0}$ to its nearest neighbor in the sample:

$$d^* := \min_{1 \leq i \leq n} \sqrt{\mathbf{x}_i^\top \mathbf{x}_i},$$

which is a random variable since the sample is random.

1. [2 point] In the special case $p = 1$, what is the cdf of d^*?
2. [2 points] In the general case $p \in \{1, 2, 3, \ldots\}$, what is the cdf of d^*? (Hint: You may find the following fact useful: the volume of a p-dimensional ball with radius r is $\frac{(r\sqrt{\pi})^p}{\Gamma(p/2+1)}$, where $\Gamma(\cdot)$ is the Gamma function.)
3. [2 points] What is the median of the random variable d^*? Your answer should be a function of both the sample size n and the dimension p. Fix $n = 100$, and plot the values of the median function for $p = 1, 2, 3, \ldots, 100$ with the median values on the y-axis and the values of p on the x-axis. What do you see?
4. [2 points] With the cdf you derived in Problem 2.2.2, answer the following question: How large should the sample size n be such that with probability at least 0.9, the distance d^* from $\mathbf{0}$ to its nearest neighbor is less than $1/2$, i.e., half way from $\mathbf{0}$ to the boundary of the ball? Your answer should be a function of p; plot this function for $p = 1, 2, \ldots, 20$ with the function values on the y-axis and values of p on the x-axis. What do you see?
5. [2 point] Having solved the previous problems, what will you say about the downside of KNN in terms of n and p?

3 Programming – Kernel Smoothing and Risk [Rob Hall, 20 points]

In this section we will study leave-one-out cross validation applied to the kernel smoother (aka the Nadaraya-Watson estimator). We will stick to data in a single dimension for the sake of simplicity.

We will have a "dataset" $\{(x_i, y_i)\}_{i=1}^n$. The data follows:

[1] No need of being super precise; qualitative correctness is enough.

$$y_i = f(x_i) + \epsilon_i,\ \epsilon_i \overset{\text{iid}}{\sim} N(0, \sigma^2)$$

The goal of regression is to estimate $f(x)$ with a function $\hat{f}(x)$. Recall that the Nadaraya-Watson estimator is given by:

$$\hat{f}(x) = \frac{\sum_{i=1}^{n} y_i K(\frac{|x_i - x|}{h})}{\sum_{i=1}^{n} K(\frac{|x_i - x|}{h})} \tag{3}$$

Where $K(\cdot)$ is the so called Kernel, and h is the bandwidth. In this example we will use the Gaussian Kernel:

$$K(a) = (2\pi)^{-1/2} \exp\left\{\frac{-a^2}{2}\right\}$$

So here we see that h will takes the place of the standard deviation, and the data point x will take the place of the mean – in the form of a gaussian distribution.

We will experiment with cross validation and compare to the minimization of the empirical risk. To have control over this experiment we will simulate data from a known distribution.

- Sample $x_i \sim U(-5, 5)$.
- Sample $\epsilon_i \sim N(0, 0.1)$ and set $y_i = \sin(x_i) + \epsilon_i$.

The type of loss we will concern ourselves with is square error:

$$\ell(y, \hat{f}(x)) = (y - \hat{f}(x))^2$$

1. (1 Point) Sample a set of size $n = 100$ and plot it, along with the true regression function $f(x)$.
2. (17 points) Write a program which performs the following:
   - Sample a "training set" of size $n = 100$, and a testing set of size $m = 100$.
   - Compute the kernel smoother for a particular choice of h, along with its empirical error, leave one out cross-validation error, and testing error.
   - Compute these measurements on the same data sample for the folowing values of h:
     $h \in \{1.0, 0.75, 0.5, 0.25, 0.1, 0.05, 0.01, 0.005, 0.001\}$
     Construct scatter plots of test error vs empirical error, and test error vs leave-one-out cross validation error. Test error should be on the y-axis.
   - Choose the function $\hat{f}$ which minimizes the leave-one-out cross validation error, and plot the training data sample along with the value of this function evaluated on the training data x values.
3. (2 points) Explain why it is a bad idea to merely minimize the empirical risk in problems like this. Reffer to the second and third plots from above.

4 Bias-Variance Tradeoff, Regression Trees [Jayant, 20 pts]

1. Construct a data set for which the training error of a linear least squares fit is zero. What will be the quadratic least squares fit to the same training data? What is the training error of the quadratic least squares fit? **[4pts]**

2. (a) Sketch the fit obtained using Nadaraya-Watson kernel regression using a box kernel with bandwidth $h = 2, 6$, to the following piece-wise constant training data: **[4pts]**

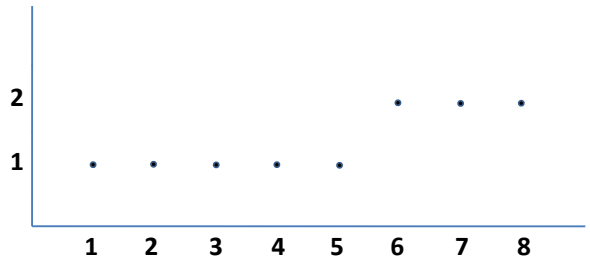


Figure (3a)

(We suggest you manually plot a few points, then heuristically fill in the middle. Try to preserve important properties of the curve, but don't worry about getting the curve exactly right.)

(b) What value of h (bandwidth) would you choose and why? **[2pts]**

(c) Now consider that the data is corrupted by noise as below.

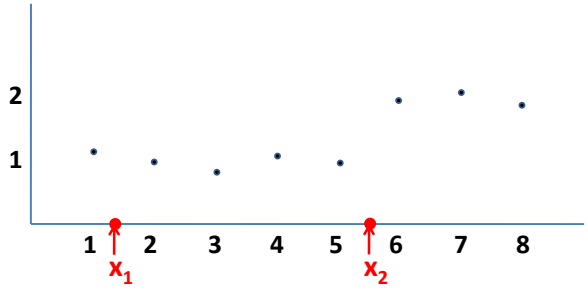


Figure (3b)

What value of h (bandwidth) would you choose to predict the label for test point x_1 and why? Would the same bandwidth provide a good prediction of the label for test point x_2? Why? (No computations needed, only answer qualitatively) **[4pts]**

3. **Regression Trees**
An alternate approach to handle piece-wise constant data is to use regression trees. Suppose that we only allow mid-point splits, i.e. each split divides the attribute's current interval into half. The tree estimate performs a least square fit in each leaf. Draw the best tree estimate with (i) three leaves, (ii) four leaves for the data in Figure (3a). **[4pts]**

(iii) Is there any advantage of using regression trees over kernel estimator? Comment. **[2pts]**

5 Decision Trees [Min Chi, 20 points]

5.1 ID3 with Descrete Attributes Only

Table 1 describes positive and negative instances of people who were and were not granted credit card. Each row indicates the values observed, and how many times that set of values was observed. For example, (F,

Low, +) was observed 10 times, while (F, Low, +) was observed 80 times.

Table 1: Credit Card Application With Two Attributes

Gender	Income	Approved	Counts
F	Low	+	10
F	High	+	95
M	Low	+	5
M	High	+	90
F	Low	-	80
F	High	-	20
M	Low	-	120
M	High	-	30

1. **[2 points:]** Compute the sample entropy H for this training data (with logarithms base 2)?

2. **[2 points:]** Calculate the information gains (IG)

$$IG(Gender) = H(Approved) - H(Approved|Gender) \tag{4}$$

and

$$IG(Income) = H(Approved) - H(Approved|Income) \tag{5}$$

for this sample of training data?

3. **[2 points:]** Draw the decision tree that would be learned by ID3 (without postpruning) from this sample of training data.

5.2 Decision Tree with Continous Attribute

Next we will add another attribute, age, to the training data. Table 2 describes the positive and negative instances of people who were and were not granted credit card. Each applicant either gets accepted (+) or rejected (-). Here each instances have three attributes: gender (F, M), income (Low, High), and age.

Table 2: Credit Card Application With Three Attributes

Gender	Income	Age	Approved
M	Low	22	+
M	High	32	+
M	High	38	+
M	High	39	-
M	High	31	-
F	Low	23	-
F	High	32	+

Note the continuous attribute age. To increase the complexity of a decision tree by the same amount for any decision, only binary splits of the form $age < A$ vs. $age \geq A$ are allowed, ~~but there can be multiple such splites in one path from root to leaf~~.

1. **[2 points]** How many possible values of A do we need to consider to determine the optimal root split for the attribute age (note that some age values are repeated more than once)?

2. **[2 points]** Draw the decision tree that would be learned by ID3 (the information-gain based algorithm presented in the lecture) and annotate each non-leaf node in the tree with the information gain attained by the respective split.

3. **[3 points]**Change one input attribute of one example in the above data set, so that the learned tree will contain at least one additional node.

4. **[7 points]** We call a training example consistent with a decision tree if it is classified correctly be the tree. Is it possible to add new examples to the original training set which are consistent with the tree learned in (2), but nevertheless result in the ID3 algorithm run on the enlarged training set to learn a tree whose root node is different from the original trees and whose number of nodes is larger than the original trees? Justify your answer by explaining informally why this is impossible, or explaining the new data you would add.