10-701 Machine Learning, Fall 2011: Homework 1

Due 10/3 at the beginning of class.

1 Decision Trees [25 points, Qirong]

1.1 Information gain and entropy

When building decision trees, it is often desirable to keep the tree short, as deep (or even full) trees are prone to overfitting. Because short trees constrain the number of splits we can make, we need to choose more "desirable" attributes to split on. When the attributes are discrete, information gain (IG) provides a useful measure of desirability, in that attributes with higher IG result in "purer" data splits. In this question, you will explore some basic properties of IG, and its relation to entropy H. First, recall some definitions. For these definitions, assume we have two discrete random variables X, Y that take values in $\{1, \ldots, k\}$.

- Entropy of X:

$$H(X) = -\sum_{x=1}^{k} p(X = x) \log p(X = x)$$

- Joint entropy of X, Y:

$$H(X, Y) = -\sum_{x=1}^{k} \sum_{y=1}^{k} p(X = x, Y = y) \log p(X = x, Y = y)$$

- Entropy of Y conditioned on $X = j$:

$$H(Y \mid X = x) = -\sum_{y=1}^{k} p(Y = y \mid X = x) \log p(Y = y \mid X = x)$$

- Conditional entropy of Y given X:

$$H(Y \mid X) = \sum_{x=1}^{k} p(X = x) H(Y \mid X = x)$$

- Information gain (also known as mutual information) between X and Y:

$$IG(X; Y) = H(X) - H(X \mid Y)$$

Using these definitions,

1. [**4 points**] Show that $IG(X; Y) = IG(Y; X)$. What does this tell you about information gain?
2. [**4 points**] Show that $IG(X; Y) = H(X) + H(Y) - H(X, Y)$.
3. [**4 points**] Show that $IG(X; Y) = H(X, Y) - H(X \mid Y) - H(Y \mid X)$.

Weather	Road Traffic	Accident Rate	Counts
Sunny	Heavy	High	17
Sunny	Heavy	Low	22
Sunny	Light	High	13
Sunny	Light	Low	31
Rainy	Heavy	High	20
Rainy	Heavy	Low	5
Rainy	Light	High	12
Rainy	Light	Low	11

Table 1: Daily weather, road traffic and accident rates

1.2 Building decision trees with information gain

Consider the dataset in Table 1 with 3 binary attributes. Using this dataset, answer the following questions:

1. [**3 points**] Suppose we want to build a decision tree that predicts the accident rate given weather and road traffic. Our first order of business is to decide what attribute to split at the root. Without calculating anything, explain how you would use information gain to decide between splitting on weather or on road traffic.

2. [**3 points**] Using your answer to part 1, determine the root attribute. Show your calculations.

3. [**3 points**] Now suppose the dataset contains a fourth attribute, temperature, that takes on continuous values. When a decision tree splits on a continuous attribute X, it divides the data into examples where $X \leq a$ and examples where $X > a$, for some chosen threshold a. Assuming the dataset has K unique values for temperature, how would you determine the optimum threshold a?

4. [**4 points**] Real world datasets are not always perfect - some contain systematic errors such as duplicated attributes or attributes with only one value. Furthermore, not all machine learning algorithms are well-suited to handling such errors. For example, Naive Bayes performs poorly when attributes are duplicated. When learning decision trees, what happens when there are duplicated attributes? What about one-value attributes? Explain your answers in terms of information gain.

2 Linear Regression[25 (+10 bonus points), Nan]

In linear regression, we are given training data of the form, $\mathcal{D} = (\mathbf{X}, \mathrm{y}) = \{(\mathrm{x}_i, y_i)\}, i = 1, 2, ..., N$, where $\mathrm{x}_i \in \mathcal{R}^{1\times M}$, i.e. $\mathrm{x}_i = (x_{i,1}, \cdots, x_{i,M})^{\mathrm{T}}$, $y_i \in \mathcal{R}$, $\mathbf{X} \in \mathcal{R}^{N\times M}$, where row i of $\mathbf{X}$ is $\mathrm{x_i}^{\mathrm{T}}$, and $\mathrm{y} = (y_1, \cdots, y_N)^{\mathrm{T}}$. Assuming a parametric model of the form: $y_i = \mathrm{x_i}^{\mathrm{T}}\beta + \epsilon_i$, where ϵ_i are noise terms from a given distribution, linear regression seeks to find the parameter vector β that provides the best of *fit* of the above regression model. One criteria to measure fitness, is to find β that minimizes a given loss function $\mathrm{J}(\beta)$. In class, we have shown that if we take the loss function to be the square-error, i.e.:

$$\mathrm{J}_1(\beta) = \sum_i (y_i - \mathrm{x_i}^{\mathrm{T}}\beta)^2 = (\mathbf{X}\beta - \mathrm{y})^{\mathrm{T}}(\mathbf{X}\beta - \mathrm{y}) \tag{1}$$

Then

$$\beta^* = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathrm{y} \tag{2}$$

Moreover, we have also shown that if we assume that $\epsilon_1, ..., \epsilon_N$ 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 $p(\mathrm{y}|\mathbf{X}; \beta)$.

2.1 1-D Regression [19 pt]

Consider the simplest case where the parametric model is of the form: $y = a\mathrm{x}+b+\epsilon$. ϵ is a noise term following a zero-mean Gaussian distribution $\epsilon \sim \mathcal{N}(0, \sigma^2)$. We have N training data points, $(\mathrm{x}_i, y_i), i = 1, 2, ..., N$, where $\mathrm{x}_i \in \mathcal{R}$.

(a) [2 pt] Write down the squared-error loss function. Note: Do not use the matrix form. The function should be described using the 1-D parametric model.

(b) [6 pt] Calculate the the minimizer, a^* and b^*, of the lost function, and show that

$$\begin{aligned} a^* &= \frac{N\left(\sum_i y_i x_i\right) - \left(\sum_i x_i\right)\left(\sum_i y_i\right)}{N\left(\sum_i x_i^2\right) - \left(\sum_i x_i\right)^2} \\ b^* &= \frac{\left(\sum_i x_i^2\right)\left(\sum_i y_i\right) - \left(\sum_i x_i\right)\left(\sum_i y_i x_i\right)}{N\left(\sum_i x_i^2\right) - \left(\sum_i x_i\right)^2} \end{aligned}$$

Show your work. Again, do not use the matrix form. Directly minimizing the loss function in problem (a).

(c) [2 pt] Write down the MLE formula of the 1-D parametric model.

(d) [4 pt] Show that the least square estimate in problem (a) is the same as the MLE estimate in problem (c).

(e) [5 pt] Show that the minimizer calculated from problem (b) equals to the matrix form shown in (2).

2.2 Regularization: Ridge and Lasso Regression [6+10 pt]

For this part assume that the noise terms are IID distributed according to $\mathcal{N}(0, \sigma^2)$.

You may want to use the following fact

$$\frac{d|\beta_a|}{d\beta_a} = \begin{cases} 1 \text{ if } \beta_a > 0 \\ -1 \text{ if } \beta_a < 0 \\ \text{undefined if } \beta_a = 0 \end{cases}$$

(a) [6 pt] Ridge regression: Instead of minimizing $\mathrm{J}_1(\beta)$, minimize the following loss function:

$$\mathrm{J}_R(\beta) = \sum_i (y_i - \mathrm{x_i}^{\mathrm{T}}\beta)^2 + \lambda \sum_{j=1}^{M} \beta_j^2 = (\mathbf{X}\beta - \mathrm{y})^{\mathrm{T}}(\mathbf{X}\beta - \mathrm{y}) + \lambda \parallel \beta \parallel_2^2 \tag{3}$$

Derive the value of β^* that minimizes (3) in closed form. [please, show your work **in details** to get full credit]

(b) [Extra 6 pt] Lasso regression: Instead of minimizing $\mathrm{J}_1(\beta)$, minimize the following loss function:

$$\mathrm{J}_L(\beta) = \sum_i (y_i - \mathrm{x_i}^{\mathrm{T}}\beta)^2 + \lambda \sum_{j=1}^{M} |\beta_j| = (\mathbf{X}\beta - \mathrm{y})^{\mathrm{T}}(\mathbf{X}\beta - \mathrm{y}) + \lambda \parallel \beta \parallel_1 \tag{4}$$

Assume $\mathbf{X}^{\mathrm{T}}\mathbf{X} = I$. **Derive** the value of β^* that minimizes (4). Hint: There is no closed form result in some certain range. What does that mean? [please, show you work **in details** to get full credit]

(c) [Extra 4 pt] Assume $\mathbf{X}^{\mathrm{T}}\mathbf{X} = I$. Write down the value of β^* that minimizes (1), (3), and (4) respectively. Compare and explain how different regularizations shrink the value of parameters.

3 Neural Networks [25 points, Bin]

3.1 Representation

Suppose that you have two types of activation functions at hand:

- Hard threshold

$$y = \begin{cases} 1 & \text{if } w_0 + \sum_i w_i x_i \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

- Linear

$$y = w_0 + \sum_i w_i x_i \tag{6}$$

Which of the following functions can be exactly represented by a neural network with one hidden layer, using hard threshold and / or linear activation functions (meaning the 2 layers could use both hard threshold, both linear, or hard threshold for one layer and linear for the other)? For each case, justify your answer: if yes, draw the neural network, with choice of activation functions for both levels, and briefly explain how the function is represented by your neural network; if no, explain why not.

(1) **[2 points]** Polynomials of degree one

(2) **[2 points]** Polynomials of degree two

(3) **[2 points]** Hinge loss $y(x) = max(1 - x, 0)$

(4) **[2 points]** Reversed hard threshold

$$y = \begin{cases} 0 & \text{if } w_0 + \sum_i w_i x_i > 0 \\ -2 & \text{otherwise} \end{cases} \tag{7}$$

(5) **[2 points]** Piece-wise constant function in 1-D

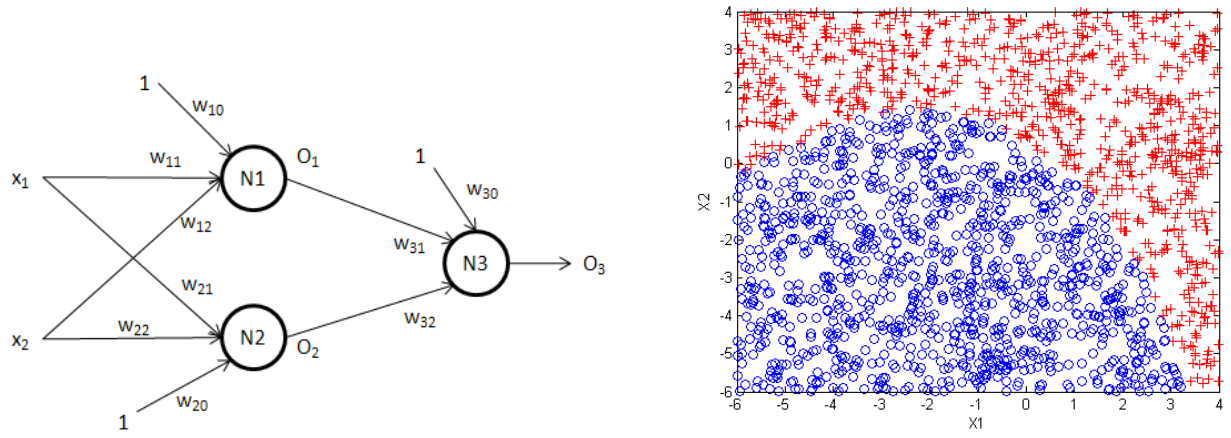
3.2 Decision Boundary

As we discussed in class, one important question we need to ask when learning about a new classifier is what kind of decision boundaries can this classifier learn. Consider a 2-layer Neural Network, recall from class that the input a_j for a node j is given by:

$$a_j = \sum_i w_{ji} o_i \tag{8}$$

Where, w_{ji} is the weight from unit i to unit j, and o_i is the activation/output of unit i. The activation of unit i is the output of a logistic function in this problem :

$$o_i = \sigma(a_i) = \frac{1}{1 + \exp^{(-a_i)}} \tag{9}$$



(a) (b)

Figure 1: (a) 2-layer NN with logistic activation functions at both the hidden and output layers. (b) A 2-class dataset: '+' and 'o' marks positive and negative labels respectively.

Consider the classification task shown in figure 1(a) where '+' and 'o' denotes positive and negative classes, respectively. Consider the 2-layer network in Figure 1(b). This network has 9 weights and a logistic activation function for both the hidden and output layers.

(1) **[4 points]** For each of the following classifiers, state with a one-line explanation whether or not they can learn the decision boundary illustrated in Figure 1(a): 1-KNN, decision tree, Naive Bayes, and logistic regression.

(2) **[2 point]** Express o_1 and o_2 in terms of $x_1, x_2, w_{10}, w_{11}, w_{12}, w_{20}, w_{21}, w_{22}$.

(3) **[2 point]** Write down the decision rule for this 2-layer NN classifier.

(4) **[2 point]** Let's assume that we removed the logistic function form the hidden layer ONLY and instead used an identity function, i.e $o_1 = a_1$ and $o_2 = a_2$, while maintaining the same weight values. Write down the decision rule for this 2-layer NN classifier. Can this neural network learn the correct decision boundary? Justify your answer.

3.3 Cross-Entropy and Noisy Label

In class we discussed that in order to train a NN we need to define an error function E[W] (such as the squared error) that can be minimized using the backpropagation algorithm to find the network weights. We also discussed that this error function can be driven using the M(C)LE principle based on a signal plus noise interpretation in the context of a regression setting. Consider a classification task with Data $\mathcal{D} = (\mathbf{X}, \mathbf{t}) = \{(\mathrm{x}^i, t^i)\}, i = 1, 2, ..., N,$. For example, $\mathbf{x}^i$ might be a face image, and t^i is a binary label equals 0 if the face is for a male and 1 if the face is for a female. Now consider a 2-layer NN based on logistic threshold units at both the hidden and output layers. If we let y denote the real-valued final output of the network, where $y \in [0, 1]$, then we might naturally wish to interpret this output as the probability that the boolean class label t takes on the value $t = 1$; that is, $y = P(t = 1|x; W)$. In this case, as we have done in logistic regression, it is natural to find the NN weights W using the M(C)LE principle as follows:

$$\mathbf{W}_{\mathrm{MLE}} = \arg\max_{\mathbf{w}} \prod_{i=1}^{N} p(t^i|\mathrm{x}^i; \mathbf{w}) \tag{10}$$

(1) **[2 points]** Show that maximizing (10) is equivalent to minimizing the cross-entropy error function given by:

$$\mathrm{E}[W] = -\sum_{i=1}^{N} \left[t^i \ln y^i + (1 - t^i) \ln(1 - y^i) \right] \tag{11}$$

where, y^i is the output of the network corresponding to example i.

(2) **[3 points]** Suppose that there is a probability ϵ that the class label on a training data point has been incorrectly set. Assuming independent and identically distributed data, write down the error function corresponding to the negative log likelihood. Verify that the error function (11) is obtained when $\epsilon = 0$. Explain how this error function makes the model robust to incorrectly labeled data, in contrast to the usual error function (11).

4 Logistic Regression and Naive Bayes[25 points, Suyash]

In this question, you will implement logistic regression and naive bayes classifiers, and compare the two in solving a two-class classification problem.

1. [5 points] Implement a logistic regression classifier using (i) IRLS and (ii) Gradient ascent. Use the dataset q4_1.mat which contains both the training sets (Xtrn,Ytrn) and the test sets (Xtst,Ytst), for training and testing. Report the number of misclassified examples on the test set for each classifier.

2. [5 points] Based on the logistic regression formula you learnt in class, derive the analytical expression for the decision boundary of the logistic regression classifier. What can you say about the shape of the decision boundary?

3. [5 points] Implement a gaussian naive bayes classifier as described in class. Use the dataset from the first part and run the naive bayes classifier on it. Report the number of misclassified examples as before.

4. [5 points] In class, you proved that under certain conditions, for continuous X and boolean Y, the conditional probability $P(Y|X)$ for a naive bayes classifier is given by a logistic function with suitable weights. Prove that the result also holds if both the X and Y variables are boolean.

5. [5 points] Use q4_5.mat as the dataset for this question. Load the dataset, and train the logistic regression classifier on the training set (Xtr,Ytrn). Plot the conditional probability $P(Y = 1|X)$ over the region $\{-3, 6\} \times \{-4, 4\}$ and the decision boundary for the logistic regression classifier. Report the number of misclassified test examples. Repeat the same for the the naive bayes classifier and compare the results (Use the gradient ascent implementation of the logistic regression classifier). How can you explain the difference in results? (Hint: Plot the data and comment.)