CSE446 Machine Learning, Spring 2016: Homework 2

Due: 5/9/2016

Start Early! Submit your assignment to Gradescope. All answers **MUST** be written solely in the boxes provided. Anything written outside the boxes will not be seen by the graders. We will only accept answers in **.pdf** format.

You may print the document and handwrite your answers, paste typing over the PDF, or modify the LaTex source. If you modify the source, you MUST not change the size or position of the answer boxes provided.

1 Linear Regression [23 pts]

Consider the regression problem where you want to predict a variable y given d features x_1, x_2, ... x_d. The error terms η_i are independent and normally distributed. Suppose that you have the following linear regression model:

$$y_i = w_1 x_{1i} + w_2 x_{2i} + ... + w_d x_{di} + \eta_i$$

Assume here that we have M data points.

1. [6 points] Suppose that we fix all weights but w_1 equal to 0. Find the value of w_1 that minimizes the least squared error.

2. *[5 points]* Suppose now that all weights can take any value. Will this new model fit the training data better than the model used in part 1? Briefly discuss your answer (1-2 sentences).

3. *[6 Points]* Recall from lecture that we can model linear regression as MLE for a Gaussian distribution, where the training points are normally distributed with mean $\mu = f(\mathbf{x}) = \mathbf{x} \cdot \mathbf{w}$ and variance σ^2.

   Suppose you knew instead that the variance was not uniform across each data point and each sample vector $\mathbf{w}_i$ has variance σ_i^2.

   Derive the closed form least squares estimator that solves for weight vector $\mathbf{w}$ in this situation. There is no regularization term in this example. Your answer should look similar to the vectorized equation in the notes. (Hint: start by writing down the log likelihood of the data)

4. *[6 Points]* Now assume you have the same objective function that you derived while answering part 3, but with the addition that you are performing L1-regularization using LASSO. Furthermore, the regularization weights are applied per feature and are encoded in the diagonal matrix Λ where the penalty for each feature $\mathbf{w}_i$ is an entry on the diagonal λ_i.

   Write this expression as an *argmin* expression function similar to what you see in the lecture notes. Derive the gradient of this objective and write the expression for w_0 after one update of coordinate descent optimization. You may assume all entries of vector to $\mathbf{w}$ are strictly positive and greater than 0.

2 Regularization [16 pts]

Regularization is an important technique to prevent your models from outfitting. For linear regression, we have covered both LASSO and ridge regression. For the following, recall that the loss function under ridge regression is

$$L_R = \sum_{i=1}^{n}(y_i - x_i \cdot \hat{w})^2 + \lambda \|\hat{w}\|_2^2$$

where

$$\lambda \|\hat{w}\|_2^2 = \lambda \sum_{i=1}^{d}(\hat{w}_i)^2 \tag{1}$$

and λ is our regularization constant.

The loss function to be optimized under LASSO is

$$L_L = \sum_{i=1}^{n}(y_i - x_i \cdot \hat{w})^2 + \lambda \|\hat{w}\|_1$$

where

$$\lambda \|\hat{w}\|_1 = \lambda \sum_{i=1}^{d} |\hat{w}_i|. \tag{2}$$

1. *(8 points)* Suppose we increase our λ from 0 to a large (yet still finite) value, and you are using ridge regression. Briefly discuss what happens to each of the following quantities using 1 or 2 sentences.

   (a) The error on the training set

   (b) The error on the testing set

   (c) The magnitudes of the elements of w

   (d) The number of nonzero elements in w

2. *(8 points)* Suppose now we are using LASSO (and we still increase our λ from 0 to a large value). Briefly discuss what happens to each of the following quantities using 1 or 2 sentences.

   (a) The error on the training set

   (b) The error on the testing set

   (c) The magnitudes of the elements of w

   (d) The number of nonzero elements in w

3 Generalized Logistic Regression [14 pts]

For logistic regression, the distribution of a Bernoulli random variable Y (i.e. a variable that is 1 with probability p and 0 with probability $1 - p$) given feature vector $\mathbf{x}$ is:

$$p(Y = 1|\mathbf{x}) = \frac{1}{1 + e^{-\mathbf{x} \cdot \mathbf{w}}}$$

This is known as binary logistic regression. Consider now the case where Y can take on K possible values. The posterior probability is now:

$$p(Y = k|\mathbf{x}) = \frac{e^{\mathbf{x} \cdot \mathbf{w}_k}}{\sum_{j=1}^{K} e^{\mathbf{x} \cdot \mathbf{w}_j}}$$

$\mathbf{x}$ is a d - length row vector representing an observation of features, $\mathbf{w}_j$ are d-length column vectors representing feature weights. We wish to learn the weight vectors $\mathbf{w}_1, ..., \mathbf{w}_K$ for K possible classes.

1. *[6 points]* We observe N data points $\mathbf{X} = \{\mathbf{x}_i, y_i\}$ for $i = 1$ to N. Derive the log-likelihood of the observed data $\log P(Y|\mathbf{X})$.

2. *[6 points]* Now we add a L2 regularizing term to prevent overfitting. The objective function $f(\mathbf{X})$ is now:

$$f(\mathbf{X}) = \log P(Y|\mathbf{X}) - \lambda \sum_{k=1}^{K} ||\mathbf{w}_k||^2$$

Find the gradient of $f(\mathbf{X})$ with respect to the weight vector $\mathbf{w}$.

3. *[2 points]* If the weight vector for class k at iteration t is $\mathbf{w}_k^{(t)}$, state the batch gradient descent update rule at iteration $t+1$ for learning rate η in terms of your answer to part 2.

4 Naive Bayes [18 pts]

Suppose you wish to learn a model that maps X to Y, where Y is a Boolean random variable and X is a vector containing n Boolean random variables. For example, we could have $Y = 0$ and $X =< 0, 1, 0, ...X_n >$. Our goal here is to learn $P(Y|X)$.

1. *[3 points]* Let us say that you know $P(< X_1, ..., X_n > |Y)$ and $P(Y)$. Write down $P(Y|X)$ in terms of $P(< X_1, ..., X_2 > |Y)$ and $P(Y)$.

2. *[3 points]* How many parameters do you need to estimate for $P(< X_1, ..., X_n > |Y)$ and $P(Y)$?

3. *[3 points]* Naive Bayes assumes conditional independence of X_i to other $X_j s$ given Y. Use the conditional independence assumption to further simplify your answer in part 1.

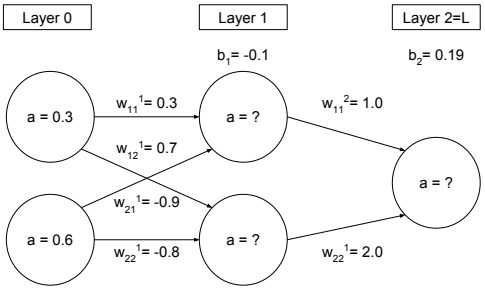
4. *[3 points]* How many parameters do you need to estimate now?

5. *[3 points]* What is the maximum likelihood estimate for $P(X_i|Y)$?

6. *[3 points]* Both Naive Bayes and logistic regression are binary classifiers. When would Naive Bayes significantly underperform compared to logistic regression?

5 Backpropagation [28 points]

Consider the following 2 layer neural network:



$\mathbf{W}_{j,k}^{(i)}$ denotes the weight on the edge between the kth node in the $(i-1)$th layer and the jth node in the ith layer and $b^{(i)}$ denotes the bias value for the nodes in the ith layer, such that

$$z_j^i = \left(\sum_k W_{jk}^i a_k^{i-1} \right) + b_i$$

is the pre-synaptic activation for jth node in ith layer. The full vector of pre-synaptic activations can also be written as

$$\mathbf{z}^{(i)} = \mathbf{W}^{(i)} \mathbf{a}^{(i-1)} + \mathbf{b}^{(i)}.$$

The inputs are activation at layer "zero," so $\mathbf{a}^{(0)} = \mathbf{x}$ and $\mathbf{a}^{(L)} = \hat{y}$, where L is the number of layers. The post-synaptic activations pass through a nonlinearity, so that

$$a_j^{(i)} = \sigma(z_j^{(i)}).$$

In this assignment, we will use the soft rectifier nonlinearity $\sigma(z) = \ln(1 + e^z)$. This is a popular smooth approximation of the rectifier function $f(z) = max(0, z)$. Finally, we will define the loss function to be

$$\ell(y, \hat{y}) = \tfrac{1}{2} ||y - \hat{y}||_2^2,$$

where y is the output label in the dataset.

5.1 Forward propagation

1. *[4 points]* Compute $a_1^{(1)}$ for the weights and input shown in the neural network diagram numerically, but be sure to show your work! (here and in all subsequent questions)

2. *[4 points]* Compute $a_2^{(1)}$.

3. *[4 points]* Compute $a_1^{(L)} = a_1^{(2)} = \hat{y}$.

4. *[4 points]* The soft rectifier function $\ln(1 + e^z)$ overflows when z is a large positive number. Rewrite the soft rectifier function in a different form such that it does not overflow with large z. How would you compute this function in practice to avoid these issues?

5.2 Back propagation

1. *[4 points]* The backpropagation algorithm calculates the partial derivative of the cost function with respect to each weight and uses $\frac{\partial \ell}{\partial W^i_{jk}}$ to optimize w^i_{jk} via gradient decent. First, express $\frac{\partial \ell}{\partial a^{(L)}_j}$, $\frac{\partial a^{(L)}_j}{\partial z^{(L)}_j}$, and $\frac{\partial z^{(L)}_j}{\partial W^{(L)}_{jk}}$ in terms of y, $a^{(L)}_j$, $a^{(L-1)}_k$, and $z^{(L)}_j$ (symbolically).

2. *[4 points]* Suppose we have $y = 1$ and the learning rate $\eta = 1$. Calculate $\frac{\partial \ell}{\partial W^{(L)}_{11}}$ and update $W^{(L)}_{11}$ (numerically).

3. *[4 points]* Now, for the hidden layers, calculate $\frac{\partial \ell}{\partial W^{(1)}_{11}}$ and update $W^{(1)}_{11}$ with the learning rate $\eta = 1$.

6 Programming

The programming portion of this assignment will be announced on Wednesday, 4/20/2016.