CIS 520, Machine Learning, Fall 2015: Assignment 3

Arpit Panwar

October 2, 2015

Collaborator:

Anusha Fernando

1 Linear Regression and LOOCV

In the last homework, you learned about using cross validation as a way to estimate the true error of a learning algorithm. A solution that provides an almost unbiased estimate of this true error is *Leave-One-Out Cross Validation* (LOOCV), but it can take a really long time to compute the LOOCV error. In this problem, you will derive an algorithm for efficiently computing the LOOCV error for linear regression using the *Hat Matrix*. [1]

Assume that there are n given training examples, $(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)$, 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 weighted *linear* combination of the input features with weights given by $\mathbf{w}$, plus some Gaussian noise.

We can write this in matrix form by stacking the data points as the rows of a matrix X so that x_{ij} is the j-th feature of the i-th data point. Then writing Y, $\mathbf{w}$ and ϵ as column vectors, we can express the linear regression model in matrix form as follows:

$$Y = X\mathbf{w} + \epsilon$$

where:

$$Y = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}, X = \begin{bmatrix} x_{11} & x_{12} & \ldots & x_{1m} \\ x_{21} & x_{22} & \ldots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{nm} \end{bmatrix}, \mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix}, \quad \text{and } \epsilon = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

Assume that ϵ_i is normally distributed with variance σ^2. We saw in class that the maximum likelihood estimate of the model parameters $\mathbf{w}$ (which also happens to minimize the sum of squared prediction errors) is given by the *Normal equation*:

$$\hat{\mathbf{w}} = (X^T X)^{-1} X^T Y$$

Define $\hat{Y}$ to be the vector of predictions using $\hat{\mathbf{w}}$ if we were to plug in the original training set X:

$$\begin{aligned} \hat{Y} &= X\hat{\mathbf{w}} \\ &= X(X^T X)^{-1} X^T Y \\ &= HY \end{aligned}$$

where we define $H = X(X^T X)^{-1} X^T$ (H is often called the *Hat Matrix*).

[1] Unfortunately, such an efficient algorithm may not be easily found for other learning methods.

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

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

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

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

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

1. To begin with, we should consider when it is possible to compute $\hat{\mathbf{w}}$ in this framework.
   (a) $\hat{\mathbf{w}}$ is not well defined because for $m > n$ the matrix $X^T X$ is not invertible
   (b) If X is not invertible then it will not be well defined

   For the rest of question 1, assume $\hat{\mathbf{w}}$ is well-defined.
2. $O(nm^3)$
3. $$\hat{Y}_i = H * Y$$
   $$\hat{Y}_i = \sum_j H_{ij} Y_j$$
4. $$SSE = \sum_{i=1}^{n} \left(Y_i - \hat{Y}_i\right)^2$$
   Similarly
   $$SSE = \sum_{i=1}^{n} \left(Z_i - \hat{Z}_i\right)^2$$
   Replacing the values of Z
   $$= \sum_{i=1;j\neq i}^{n} \left(Y_j - \hat{Z}_i\right)^2 + \left(\hat{Y}_i^{(-i)} - \hat{Z}_i\right)$$
   The equation can be minimized when $\hat{Z}_i = \hat{Y}_i^{(-i)}$
   $$= \sum_{i=1;j\neq i}^{n} \left(Y_j - \hat{Z}_i\right)^2 + 0$$
5. We have $\hat{Y}_i = \sum_i H_{ij} Y_i$
   We know from 4 that $Z_j = Y_j j \neq i$ and $Z_j = \hat{Y}_i^{(-i)} j = i$
   Since H is just dependent on X By analogy we can say $\hat{Y}_i^{(-i)} = \sum_i H_{ij} Z_i$

6. Using 5 and 3

$$\hat{Y}_i - \hat{Y}_i^{(-i)} = \sum_{j \neq i} H_{ij} Y_j + H_{ii} Y_i - \sum_{j \neq i} H_{ij} Z_j + H_{ii} Z_i$$

$$Using\ 4\ and\ substituting\ Z_i = \hat{Y}_i^{(-i)}\ and\ Z_j = Y_j\ and\ solving$$

$$\hat{Y}_i - \hat{Y}_i^{(-i)} = H_{ii} Y_i - H_{ii} \hat{Y}_i^{(-i)}$$

7. From 6 we have $\hat{Y}_i - \hat{Y}_i^{(-i)} = H_{ii} Y_i - H_{ii} \hat{Y}_i^{(-i)}$

$$Solving\ the\ equation$$

$$\hat{Y}_i^{(-i)} * (1 - H_{ii}) = \hat{Y}_i - H_{ii} Y_i$$

$$\hat{Y}_i^{(-i)} = \frac{\hat{Y}_i - H_{ii} Y_i}{1 - H_{ii} Y_i}$$

$$Entering\ the\ value\ in\ equation\ for\ LOOCV$$

$$LOOCV = \sum_{i=1}^{n} \left(\frac{Y_i \left(1 - H_{ii}\right) - \hat{Y}_i + H_{ii} Y_i}{\left(1 - H_{ii}\right)} \right)^2$$

$$= \sum_{i=1}^{n} \left(\frac{Y_i - \hat{Y}_i}{1 - H_{ii}} \right)^2$$

Complexity for it depends on the complexity of $X(X^T X)^{-1} X^T Y$
Which is the complexity of matrix multiplication $= O(m^2 * n^2 + m^3 * n + n^3 * m)$

2 Logistic regression and Naive Bayes

A common debate in machine learning has been over generative versus discriminative models for classification. In this question we will explore this issue by considering Naive Bayes and logistic regression.

1. Naive Bayes : $P(X, Y)$ and Logistic Regression: $P(Y|X)$

2.

$$\textit{Using Bayes rule}$$

$$P(Y=1|X) = \frac{P(Y=1)*P(X|Y=1)}{P(Y=0)*P(X|Y=0)+P(Y=1)*P(X|Y=1)}$$

$$\textit{Dividing by } P(Y=1)*P(X|Y=1)$$

$$= \frac{1}{1+\frac{P(Y=0)*P(X|Y=0)}{P(Y=1)*P(X|Y=1)}}$$

$$\textit{Putting in exponential and logarithms}$$

$$= \frac{1}{1+e^{\log\left(\frac{P(Y=0)*P(X|Y=0)}{P(Y=1)*P(X|Y=1)}\right)}}$$

$$\textit{Since } X \textit{ is conditionally independent}$$

$$= \frac{1}{1+e^{\left(\log\left(\frac{P(Y=0)}{P(Y=1)}\right)+\log\left(\frac{P(X|Y=0)}{P(X|Y=1}\right)\right)}}$$

$$= \frac{1}{1+e^{\left(\log\left(\frac{1-\pi}{\pi}\right)+\log\left(\frac{\theta_{i0}^{X_i}*(1-\theta_{i0})^{(1-X_i)}}{\theta_{i1}^{X_i}*(1-\theta_{i1})^{(1-X_i)}}\right)\right)}} \ldots\ldots (i)$$

$$\textit{Solving the exponent in denominator}$$

$$= \log\left(\theta_{i0}^{X_i}*(1-\theta_{i0})^{(1-X_i)}\right) - \log\left(\theta_{i1}^{X_i}*(1-\theta_{i1})^{(1-X_i)}\right)$$

$$= \log\theta_{i0}^{X_i} + \log\left((1-\theta_{i0})^{(1-X_i)}\right) - \log\theta_{i1}^{X_i} - \log\left((1-\theta_{i1})^{(1-X_i)}\right)$$

$$= (1-X_i)\left[\log(1-\theta_{i0}) - \log(1-\theta_{i1})\right] + X_i\left[\log\theta_{i0} - \log\theta i1\right]$$

$$\textit{Solving}$$

$$= X_i\left[\log\theta_{i0} - \log\theta_{i1} + log(1-\theta_{i1}) - \log(1-\theta_{i0})\right] + \log\left(\frac{(1-\theta_{i0})}{(1-\theta_{i1})}\right)$$

$$= X_i\left[\log\left(\frac{\theta_{i0}}{\theta_{i1}}*\frac{1-\theta_{i1}}{1-\theta_{i0}}\right)\right] + \log\left(\frac{(1-\theta_{i0})}{(1-\theta_{i1})}\right)$$

$$\textit{Replacing the values in } (i)$$

$$= \frac{1}{1+e^{\left(\log\left(\frac{1-\pi}{\pi}\right)+X_i\left[\log\left(\frac{\theta_{i0}}{\theta_{i1}}*\frac{1-\theta_{i1}}{1-\theta_{i0}}\right)\right]+\log\left(\frac{(1-\theta_{i0})}{(1-\theta_{i1})}\right)\right)}}$$

$$\textit{Rearranging and comparing with logistic regression} \frac{1}{1+e^{w_0+w_1*X_i}}$$

$$w_0 + w_1 * X_i = \left(\log\left(\frac{1-\pi}{\pi}\right) + \log\left(\frac{(1-\theta_{i0})}{(1-\theta_{i1})}\right)\right) + X_i\left[\log\left(\frac{\theta_{i0}}{\theta_{i1}}*\frac{1-\theta_{i1}}{1-\theta_{i0}}\right)\right]$$

3 Double-counting the evidence

1. we need 5 parameters

X1	X2	Y=T
T	T	0.8*0.5 = 0.4
T	F	0.8*0.5=0.4
F	T	0.2*0.5 = 0.1
F	F	0.2*0.5 = 0.1

Table 1: Values for Y=T

X1	X2	Y=F
T	T	0.3*0.1 = 0.03
T	F	0.3*0.9 = 0.27
F	T	0.7*0.1 = 0.07
F	F	0.2*0.5 = 0.1

Table 2: Values for Y=F

X1	X2	Y Prediction
T	T	T
T	F	T
F	T	T
F	F	F

Table 3: Prediction

2.

3. For the Naive Bayes decision function $f(X_1, X_2)$, the error rate is:

$$\sum_{X_1, X_2, Y} \mathbf{1}\left(Y \neq f(X_1, X_2)\right) P(X_1, X_2, Y).$$

For this question, we will assume that the true data distribution is exactly the same as the Naive Bayes distribution, so we can write $P(X_1, X_2, Y)$ as $P(Y)P(X_1 \mid Y)P(X_2 \mid Y)$.

   (a) When using both attirbutes we need to add error when Y=T and when Y=F
   Using table 3 above
   For Y = T we have error when $X_1 = F \; and \; X_2 = F$
   For Y = F we have error in the other 3 cases
   Adding the errors based on the formula above
   $= 0.1 * 0.5 + 0.03 * 0.5 + 0.27 * 0.5 + 0.07 * 0.5 = 0.235$

   (b) When using only X_1 we will get error when $X_1 = TwhenY = FandX_1 = FwhenY = T$
   error $= 0.3 * 0.5 + 0.2 * 0.5$
   $= 0.250$

   (c) Similarly for X_2 we can computer error
   error $= 0.5 * 0.5 + 0.1 * 0.5$
   $= 0.300$

   (d) Error rate decreases

4. Now, suppose that we create a new attribute X_3, which is an exact copy of X_2. So, for every training example, attributes X_2 and X_3 have the same value, $X_2 = X_3$.

   (a) No they are co-related

X1	X2	X3	Y=T	Y=F	Y Pred
T	T	T	0.2	0.003	T
T	F	T			
T	F	F	0.2	0.243	F
T	T	F			
F	T	F			
F	T	T	0.050	0.007	T
F	F	F	0.050	0.567	F
F	F	T			

Table 4: Prediction for Y

(b) As in 3.3 above calculating the error based on true probabilities mentioned in table 1 and 2. error = 0.3

5. Naive Bayes assumes conditional independence but X_2 and X_3 are co-related

6. No because logistic regression assumes no such conditions and will not choose either X_2 or X_3

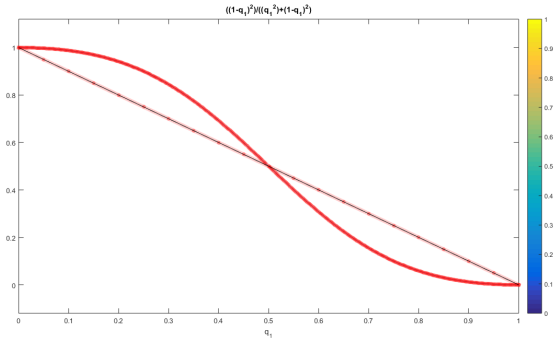
7. :

(a) Using the Bayes decision rule and as given X_2 and X_3 are equal. Substituting these values and since we want to choose $P(Y = T|X_i)$, the decision should be greater than or equal to 1

$$\begin{aligned}
\frac{p * q * q}{(1-p)(1-q)(1-q)} &\geq 1 \\
pq^2 &\geq (1-p)(1-q)^2 \\
pq^2 &\geq (1-q)^2 - p(1-q)^2 \\
p(q^2 + (1-q)^2) &\geq (1-q)^2 \\
p &\geq \frac{(1-q)^2}{(q^2 + (1-q)^2)}
\end{aligned}$$

(b) Similar to part above using True rule

$$\begin{aligned}
\frac{p * q}{(1-p)(1-q)} &\geq 1 \\
pq &\geq (1-p)(1-q) \\
p &\geq (1-q)
\end{aligned}$$



(c)

4 Feature Selection

We saw in class that one can use a variety of regularization penalties in linear regression.

$$\hat{w} = \arg\min_{w} \quad \|Y - Xw\|_2^2 + \lambda \|w\|_p^p$$

Consider the three cases, $p = 0$, 1, and 2. We want to know what effect these different penalties have on estimates of w.

Let's see this using a simple problem.

Use the following data (also provided in matlab format). Assume the constant term in the regression is zero, and assume $\lambda = 1$, except, of course, for question (1). You don't need to write code that solves these problems in their full generality; instead, feel free to use matlab to do the main calculations, and then just do a primitive search over parameter space by plugging in a few different values. Matlab function fminsearch will be helpful. (*Note:* If you are not familiar with function handles, please review the code from HW 2 or see Matlab documentation.)

1. $\hat{w}_{MLE} = [0.8891; -0.8260; 4.1902]$
2. $\hat{w} = [0.8646; -0.8210; 4.1218]$
3. $\hat{w} = [0.8749; -0.8182; 4.1829]$
4. Calculating the values of $\hat{w}$ for 8 cases i.e. permuting a 3x1 vector of 0's and 1's and putting in the values of
   w computed in the first part.
   We get the minimum values for [1;1;1] since the L_0 norm will return 0 thus in that case the value will be equal to W_{MLE}
   Thhus w is [0.8891;-0.8260;4.1902] and the min value is 3.1078e+03
5.
6. When $\lambda > 0$, we make a trade-off between minimizing the sum of squared errors and the magnitude of $\hat{w}$. In the following questions, we will explore this trade-off further. For the following, use the same data from data.mat.
   (a) 0.0061
   (b) i. Yes the error doubles because the bias increases in the training data.
   ii. Nothing happens
   (c) $\lambda = 4$
   (d) $\lambda = 28$