CSE512 - Machine Learning - Homework 1

Yang Wang
Computer Science Department
Stony Brook University
yang.wang@stonybrook.edu

1. Probability

1.1. Expectation

Given

$$X = min(X_1, X_2), \;\; X_1, X_2 \;\; i.i.d \;\; U(0,1)$$

We have

$$\begin{aligned} F_X(x) &= P(X < x) \\ &= 1 - P(X > x) \\ &= 1 - P(X_1 > x)P(X_2 > x) \\ &= 1 - (1-x)(1-x) \\ &= 2x - x^2, \text{ where } x \in [0,1] \end{aligned}$$

Thus

$$f_X(x) = \frac{d}{dx} F_X(x) = 2 - 2x, \text{ where } x \in [0,1]$$

So

$$E(X) = \int_0^1 x f_X(x) dx = \frac{1}{3}$$

1.2. Variance

$$E(X^2) = \int_0^1 x^2 f_X(x) dx = \frac{1}{6}$$

$$Var(X) = E(X^2) - E^2(X) = \frac{1}{6} - (\frac{1}{3})^2 = \frac{1}{18}$$

1.3. Covariance

$$\begin{aligned} f_{X_1,X}(x,y) &= f_{X_1}(x) f_{X|X_1}(y|x) \\ &= 1 \cdot f_{X|X_1}(y|x) \\ &= \begin{cases} (1-x)\delta(y-x), & 0 \le y = x \le 1 \\ 1, & 0 \le y < x \le 1 \end{cases} \end{aligned}$$

$$\begin{aligned} Cov(X_1, X) &= E(X_1 X) - E(X_1)E(X) \\ &= \int_0^1 \int_0^x xy f_{X_1,X}(x,y) dy dx - \frac{1}{2} \times \frac{1}{3} \\ &= \int_0^1 \int_0^x xy dy dx + \int_0^1 x^2(1-x) dx - \frac{1}{6} \\ &= \frac{1}{24} \end{aligned}$$

2. Parameter Estimation

2.1. MLE

2.1.1 Log-likelihood

Given

$$\mathbf{X} = (X_1, ..., X_n), \;\; X_i \;\; i.i.d \;\; Poisson(\lambda)$$

Then

$$\begin{aligned} P(X|\lambda) &= P(X_1, ..., X_n|\lambda) = \prod_{i=1}^n P(X_i|\lambda) \\ &= \prod_{i=1}^n \frac{\lambda^{X_i}}{X_i!} e^{-\lambda} \end{aligned}$$

Thus

$$\begin{aligned} logP(X|\lambda) &= log(\prod_{i=1}^n \frac{\lambda^{X_i}}{X_i!} e^{-\lambda}) \\ &= \sum_{i=1}^n (X_i log(\lambda) - log(X_i!) - \lambda) \end{aligned}$$

2.1.2 MLE in general

$$MLE(\lambda) = \hat{\lambda} = \arg\max_{\lambda} logP(X|\lambda)$$

Set

$$\frac{d}{d\lambda} logP(X|\lambda) = \sum_{i=1}^n (\frac{X_i}{\lambda} - 1) = 0$$

Solve

$$\hat{\lambda} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

2.1.3 MLE for observed data

$$\hat{\lambda} = \frac{4+1+3+5+5+1+3+8}{8} = 3.75$$

2.2. MAP

2.2.1 Posterior

Given

$$\mathbf{X} = (X_1, ..., X_n), \;\; X_i \;\; i.i.d \;\; Poisson(\lambda)$$

Then

$$\begin{aligned} P(\lambda|X) &\propto P(\lambda)P(X|\lambda) \\ &\propto \lambda^{\alpha-1}e^{-\beta\lambda}\prod_{i=1}^{n}\lambda^{X_i}e^{-\lambda} \\ &\propto \lambda^{\sum_{i=1}^{n}X_i+\alpha-1}e^{-(n+\beta)\lambda} \end{aligned}$$

Since $P(\lambda|X)$ is a Gamma distribution,

$$P(\lambda|X) = \frac{\beta'^{\alpha'}}{\Gamma(\alpha')}\lambda^{\alpha'-1}e^{-\beta'\lambda}, \;\; \lambda > 0$$

$$\text{where,} \;\; \alpha' = \alpha + \sum_{i=1}^{n} X_i$$

$$\beta' = \beta + n$$

2.2.2 MAP

$$MAP(\lambda) = \hat{\lambda} = \arg\max_{\lambda} logP(\lambda|X)$$

Set

$$\frac{d}{d\lambda}logP(\lambda|X) = \frac{\alpha'-1}{\lambda} - \beta' = 0$$

Solve

$$\hat{\lambda} = \frac{\alpha'-1}{\beta'} = \frac{\sum_{i=1}^{n}X_i + \alpha - 1}{n+\beta}$$

2.3. Estimator Bias

2.3.1 MLE of η

$$\hat{\eta} = MLE(\eta) = \arg\max_{\eta} P(X| - \frac{1}{2}log\eta)$$

Set

$$\begin{aligned} \frac{d}{d\eta}logP(X| - \frac{1}{2}log\eta) &= \frac{d}{d\eta}log(\frac{(-\frac{1}{2}log\eta)^X}{X!}e^{\frac{1}{2}log\eta}) \\ &= \frac{X}{\eta log\eta} + \frac{1}{2\eta} \\ &= 0 \end{aligned}$$

Solve

$$\hat{\eta} = e^{-2X}$$

2.3.2 Bias of $\hat{\eta}$

$$\begin{aligned} E_X(\hat{\eta}) &= \sum_{k=0}^{\infty} e^{-2k}\frac{\lambda^k}{k!}e^{-\lambda} \\ &= e^{-\lambda}\sum_{k=0}^{\infty}\frac{(\frac{\lambda}{e^2})^k}{k!} \\ &= e^{\lambda(1/e^2-1)} \end{aligned}$$

$$Bias_\eta(\hat{\eta}) = \eta - E_X(\hat{\eta}) = e^{-2\lambda} - e^{\lambda(1/e^2-1)}$$

2.3.3 unbiased estimate of η

$$\begin{aligned} E_X[(-1)^X] &= \sum_{k=0}^{\infty}(-1)^k\frac{\lambda^k}{k!}e^{-\lambda} \\ &= e^{-\lambda}\sum_{k=0}^{\infty}\frac{(-\lambda)^k}{k!} \\ &= e^{-2\lambda} \end{aligned}$$

$$Bias = \eta - E_X[(-1)^X] = 0$$

This estimator of η is bad, although unbiased, because (1) it is highly unstable with respect to the observed data; (2) it jumps back and forth between -1 and 1 while the true value of $\eta = e^{-2\lambda}$ only resides in $(0, 1)$.

3. Regression and MLE

3.1. Weighted Least Squares

Given

$$y_i = w^T x_i + \epsilon_i, \text{ where } \epsilon_i \sim N(0, \sigma_i^2)$$

We have

$$\begin{aligned} P(y|X, w, \sigma) &= \prod_{i=1}^{n} P(y_i|x_i, w, \sigma_i) \\ &= \prod_{i=1}^{n}\frac{1}{\sqrt{2\pi}\sigma_i}e^{-\frac{(y_i-w^Tx_i)^2}{2\sigma_i^2}} \\ &= [\frac{(2\pi)^{-\frac{n}{2}}}{\prod_{i=1}^{n}\sigma_i}]e^{-\sum_{i=1}^{n}\frac{1}{2\sigma_i^2}(y_i-w^Tx_i)^2} \end{aligned}$$

So,

$$\begin{aligned} &MLE(w) \\ &= \arg\max_{w} P(y|X, w, \sigma) \\ &= \arg\min_{w}\sum_{i=1}^{n} s_i(y_i - w^Tx_i)^2, \quad s_i = \frac{1}{\sigma_i^2} \\ &= \arg\min_{w}(y - X^Tw)^T S(y - X^Tw), \quad S = diag(s) \end{aligned}$$

Set

$$\begin{aligned}&\frac{d}{dw}(y-X^Tw)^TS(y-X^Tw)\\=&-2XS(y-X^Tw)\\=&0\end{aligned}$$

Finally,

$$\begin{aligned}MLE(w)&=(XSX^T)^{-1}XSy\\\text{where, } S&=diag(s),\ s_i=\frac{1}{\sigma_i^2}\end{aligned}$$

3.2. Robust Regression

Given

$$y_i=w^Tx_i+\epsilon_i,\ \text{where } p(\epsilon_i)=\frac{1}{2b}e^{-\frac{|\epsilon_i|}{b}}$$

We have

$$\begin{aligned}P(y|X,w,b)&=\prod_{i=1}^{n}P(y_i|x_i,w,b)\\&=\prod_{i=1}^{n}\frac{1}{2b}e^{-\frac{|y_i-w^Tx_i|}{b}}\\&=[(2b)^{-n}]e^{-\frac{1}{b}\sum_{i=1}^{n}|y_i-w^Tx_i|}\end{aligned}$$

So,

$$\begin{aligned}MLE(w)&=\arg\max_w P(y|X,w,b)\\&=\arg\min_w\sum_{i=1}^{n}|y_i-w^Tx_i|\end{aligned}$$

3.3. Robustness

The L_2 regression (Least Squares) assumes Gaussian noise. It gives L_2 penalty to the differences between our predictions and the observed data. This constraint is too strict when outliers exist in the dataset. A single outlier can lead to a big shift in the linear model. On the other hand, L_1 regression (Robust regression) assumes Laplacian noise and gives L_1 penalty, which can accommodate well and remain robust to the existence of outliers.

4. Ridge Regression and LOOCV

4.1. solution of Ridge regression

Reformulate the Ridge regression problem as

$$\overline{w}=\arg\min_{\overline{w}}\lambda\overline{w}^T\overline{I}\overline{w}+\|\overline{X}^T\overline{w}-y\|^2$$

Set

$$\begin{aligned}&\frac{d}{d\overline{w}}\{\lambda\overline{w}^T\overline{I}\overline{w}+\|\overline{X}^T\overline{w}-y\|^2\}\\=&2\lambda\overline{I}\overline{w}+2\overline{X}(\overline{X}^T\overline{w}-y)\\=&2((\overline{X}\overline{X}^T+\lambda\overline{I})\overline{w}-\overline{X}y)\\=&2(C\overline{w}-d)\\=&0\end{aligned}$$

So,

$$\overline{w}=C^{-1}d$$

4.2. Leave-One-Out solution

$$\begin{aligned}C_{(i)}&=\overline{X_{(i)}X_{(i)}}^T+\lambda\overline{I}\\&=\sum_{j\neq i}\overline{x_j}\overline{x_j}^T+\lambda\overline{I}\\&=\overline{X}\overline{X}^T-\overline{x_i}\overline{x_i}^T+\lambda\overline{I}\\&=C-\overline{x_i}\overline{x_i}^T\end{aligned}$$

$$d_{(i)}=\overline{X_{(i)}}y=\overline{X}y-y_i\overline{x_i}=d-y_i\overline{x_i}$$

4.3. Leave-One-Out solution

$$C_{(i)}^{-1}=(C-\overline{x_i}\overline{x_i}^T)^{-1}=C^{-1}+\frac{C^{-1}\overline{x_i}\overline{x_i}^TC^{-1}}{1-\overline{x_i}^TC^{-1}\overline{x_i}}$$

4.4. Leave-One-Out solution

$$\begin{aligned}&\overline{w}_{(i)}\\=&C_{(i)}^{-1}d_{(i)}\\=&[C^{-1}+\frac{C^{-1}\overline{x_i}\overline{x_i}^TC^{-1}}{1-\overline{x_i}^TC^{-1}\overline{x_i}}][d-y_i\overline{x_i}]\\=&C^{-1}d-y_iC^{-1}\overline{x_i}+\frac{C^{-1}\overline{x_i}\overline{x_i}^TC^{-1}[d-y_i\overline{x_i}]}{1-\overline{x_i}^TC^{-1}\overline{x_i}}\\=&\overline{w}+\frac{C^{-1}\overline{x_i}\overline{x_i}^TC^{-1}[d-y_i\overline{x_i}]-(1-\overline{x_i}^TC^{-1}\overline{x_i})y_iC^{-1}\overline{x_i}}{1-\overline{x_i}^TC^{-1}\overline{x_i}}\\=&\overline{w}+\frac{(C^{-1}\overline{x_i})[\overline{x_i}^TC^{-1}(d-y_i\overline{x_i})-y_i(1-\overline{x_i}^TC^{-1}\overline{x_i})]}{1-\overline{x_i}^TC^{-1}\overline{x_i}}\\=&\overline{w}+\frac{(C^{-1}\overline{x_i})[\overline{x_i}^T(C^{-1}d)-y_i]}{1-\overline{x_i}^TC^{-1}\overline{x_i}}\\=&\overline{w}+\frac{(C^{-1}\overline{x_i})[\overline{x_i}^T\overline{w}-y_i]}{1-\overline{x_i}^TC^{-1}\overline{x_i}}\\=&\overline{w}+(C^{-1}\overline{x_i})\frac{-y_i+\overline{x_i}^T\overline{w}}{1-\overline{x_i}^TC^{-1}\overline{x_i}}\end{aligned}$$

4.5. Leave-One-Out error

Given $C = C^T$, we have $(C^{-1})^T = (C^T)^{-1} = C^{-1}$

$$
\begin{aligned}
&\overline{w}_{(i)}^T \overline{x_i} - y_i \\
=&[\ \overline{w} + (C^{-1}\overline{x_i})\frac{-y_i + \overline{x_i}^T\overline{w}}{1 - \overline{x_i}^T C^{-1}\overline{x_i}}\ ]^T \overline{x_i} - y_i \\
=&\overline{w}^T\overline{x}_i - y_i + \frac{-y_i + \overline{x_i}^T\overline{w}}{1 - \overline{x_i}^T C^{-1}\overline{x_i}}[\overline{x_i}^T C^{-1}\overline{x_i}] \\
=&(\overline{w}^T\overline{x}_i - y_i)[1 + \frac{\overline{x_i}^T C^{-1}\overline{x_i}}{1 - \overline{x_i}^T C^{-1}\overline{x_i}}] \\
=&\frac{\overline{w}^T\overline{x}_i - y_i}{1 - \overline{x_i}^T C^{-1}\overline{x_i}}
\end{aligned}
$$

4.6. Time complexity

Using the formula above, we can efficiently calculate the LOOCV error. First we compute C $(O(nk^2))$, C^{-1} $(O(k^3))$, d $(O(nk))$, then solve $\overline{w} = C^{-1}d$ $(O(nk))$. Then we loop through n training data. In each loop, we compute the error using $\frac{\overline{w}^T\overline{x}_i - y_i}{1 - \overline{x_i}^T C^{-1}\overline{x_i}}$ $(O(k^2))$. Thus, the time complexity is $O(nk^2 + k^3 + nk + nk^2) = O(k^3 + nk^2)$.

In the usual way of computing LOOCV, we still loop through n training data. In each loop, we have to recompute $C_{(i)}^{-1}$ $(O(k^3))$. This changes the time complexity to $O(nk^2 + k^3 + nk^3) = O(nk^3)$.

5. Lasso Programming

5.1. Time complexity and making your code fast

5.1.1 residule

$$\mathbf{r} = \mathbf{y} - (\mathbf{X}^T\mathbf{w} + b\mathbf{1}_n), \quad O(\|\mathbf{X}\|_0 + n)$$

5.1.2 bias

$$\Delta b = \frac{1}{n}\mathbf{1}_n^T\mathbf{r}, \quad O(n)$$

$$b^{new} = b^{old} + \Delta b$$

5.1.3 residule

$$\mathbf{r}^{new} = \mathbf{r}^{old} - \Delta b\mathbf{1}_n, \quad O(n)$$

5.1.4 c_k

$$
\begin{aligned}
c_k &= 2\sum_{i=1}^{n} x_{ik}(y_i - (b + \sum_{j\neq k} w_j x_{ij})) \\
&= 2\sum_{i=1}^{n} x_{ik}(r_i + w_k x_{ik}) \\
&= 2\sum_{i=1}^{n} x_{ik} r_i + 2w_k \sum_{i=1}^{n} x_{ik}x_{ik} \\
&= 2\mathbf{X}_{(k,:)}\mathbf{r} + w_k\alpha_k, \quad O(z_k)
\end{aligned}
$$

5.1.5 residule

$$\mathbf{r}^{new} = \mathbf{r}^{old} - (w_k^{new} - w_k^{old})\mathbf{X}_{(k,:)}^T, \quad O(z_k)$$

5.1.6 per iteration complexity

$$O(\|\mathbf{X}\|_0 + n + n + n + \sum_{k=1}^{d}(z_k + z_k)) = O(\|\mathbf{X}\|_0 + n)$$

5.2. Matlab Implementation

I've implemented the coordinate descent method for solving lasso problem in Matlab. You can see the function in '**code/lasso.m**'. I also provided a Matlab function in '**code/lasso_path.m**' for solving the lasso problem over a regularization path.

5.3. Try lasso on synthetic data

5.3.1 $\sigma = 1$

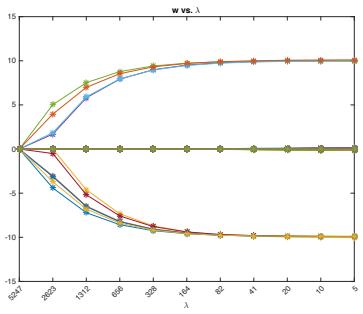

Figure 1. **w vs.** λ $(\sigma = 1)$

When noise is relatively small ($\sigma = 1$), my Lasso solver can discover the true nonzeros in the linear weights very well. Figure 1 shows how the linear weights change over the regularization path. Figure 2 shows the precision and recall of nonzeros versus λ. As shown in Figure 2, with λ decreasing from λ_{max}, we are able to discover very quickly all the true nonzeros without introducing any false nonzeros. The window of λ for high precision and recall stretches from 150 to 1300.

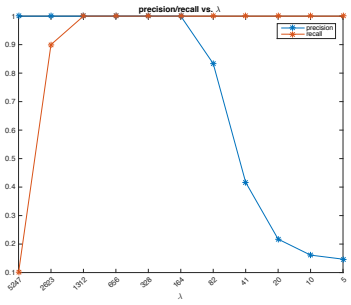

Figure 2. **precision/recall vs.** λ ($\sigma = 1$)

5.3.2 $\sigma = 10$

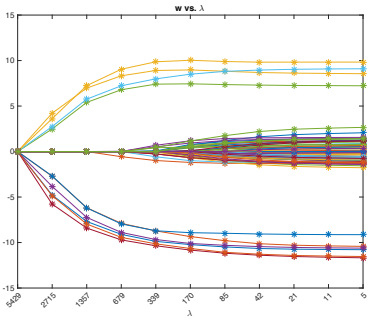

Figure 3. **w vs.** λ ($\sigma = 10$)

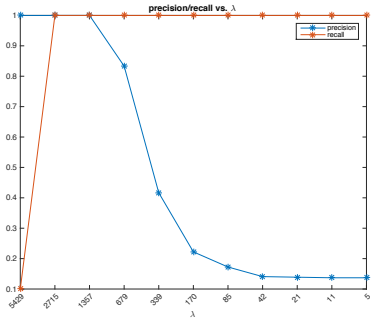

Figure 4. **precision/recall vs.** λ ($\sigma = 10$)

When noise is relatively strong ($\sigma = 10$), my Lasso solver's ability to discover the true nonzeros is impaired. Figure 3 shows how the linear weights change over the regularization path. Figure 4 shows the precision and recall of nonzeros versus λ. As shown in Figure 4, the window of λ for high precision and recall only covers between 1300 and 2700. If I use a value of λ, say 500, that worked well when $\sigma = 1$, now the regularization appears weak and the precision is low (there are many false nonzeros in the linear weights). In order to achieve better precision and recall, we have to use stronger regularization (larger λ) to battle stronger noise.

5.4. Predicting the number of a Yelp review's stars

5.4.1

Figure 5 shows the number of nonzeros in the linear weights steadily increases over the regularization path. Figure 6 shows the training/validation error versus λ.

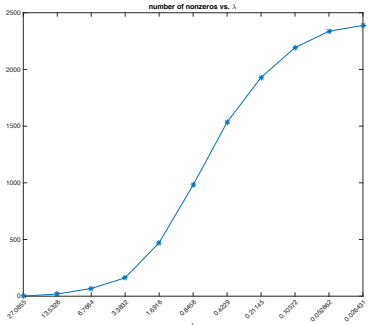

Figure 5. **number of nonzeros vs.** λ

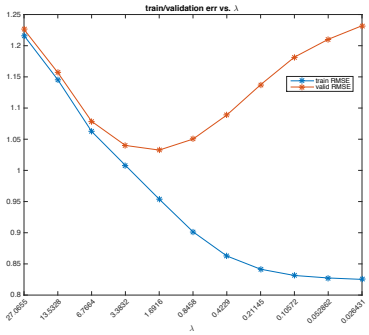

Figure 6. **RMSE vs.** λ

5.4.2

The best validation performance is achieved when $\lambda = 1.7$. we use this λ to recompute a model on the combined training and validation dataset.

The top 10 positive features are: 'great', 'best', 'amazing', 'love', 'awesome', 'delicious', 'perfect', 'wonderful', 'excellent', 'fantastic'.

The top 10 negative features are: 'no', 'overpriced', 'nothing', 'horrible', 'will never', 'bland', 'terrible', 'rude', 'the worst', 'not'.

Of cause, these features are intuitively very important to predict the number of stars of a Yelp review.