COS 511: Theoretical Machine Learning

Lecturer: Rob Schapire
Scribe: Güngör Polatkan

Lecture #16
April 7, 2008

1 Regression (continued)

Let's review our previous meteorologist problem where we were trying to select one of the applicants to hire for our prediction job. Each applicant is represented by their prediction function h_A and h_B, respectively. The question is which prediction function is better. Here is the mathematical formulation of this model:

- $x \rightarrow$ weather condition today
- $y \rightarrow$ rain tomorrow, $y \in \{0, 1\}$
- (x, y) drawn from the distribution D
- Goal: estimate $p(x) = \Pr[y = 1|x] = \mathrm{E}[y|x]$

Our goal is to estimate the function $p(x)$. This is called regression. But the main problem here is that we can not observe $p(x)$. What we can observe are x, y, $h_A(x)$, $h_B(x)$ where h_A and h_B are some prediction rules.

In the classification problem what we did was to look how many mistakes h makes. In this case we can not do that. What we can do is to look at $|h(x) - y|$ as a measure of the discrepancy between $h(x)$ and y. This is called a *loss function* or *cost function*. However instead of absolute value of difference, we use the squared difference which has more favorable properties. In this case $(h(x) - y)^2$ is called *square* or *quadratic loss*. This is a good function when used as a penalty function to compare how well the two hypotheses do. It will be shown later why this idea works. Since x, y are from the distribution D we need to look at $\mathrm{E}[(h(x) - y)^2]$. In this case the question is : Does minimizing $\mathrm{E}[(h(x) - y)^2]$ help us to know $p(x)$?

Proposition 1.1 $\mathrm{E}[(h(x) - y)^2]$ *is minimized when* h *is equal to* p.

Proof Fix x (if this is true for one x then it is true for all x). So we can write

$$p = p(x) \text{ and } h = h(x).$$

Then, the penalty function can be written as:

$$E = \mathrm{E}_y[(h - y)^2] = p(h - 1)^2 + (1 - p)h^2.$$

To find the minimum, we simply take the derivative and set it equal to zero:

$$\frac{dE}{dh} = 2(h - p) = 0 \Rightarrow h = p.$$

Now, we will try to justify the choice of the penalty function we used above. By the following theorem, we will show that minimizing the observed squared difference between the prediction and actual outcomes is equivalent to minimizing the squared difference between the prediction and the true values.

Theorem 1.2

$$\underbrace{\mathrm{E}[(h(x)-p(x))^2]}_{\textit{Goal:minimize this value}} = \underbrace{E[(h(x)-y)^2]}_{\textit{observation}} - \underbrace{E[(p(x)-y)^2].}_{\textit{intrinsic randomnes, no } h}$$

Proof Fix x (it is enough to prove for a single x since $\mathrm{E}_{x,y}[\cdot] = \mathrm{E}_x[\mathrm{E}_{y|x}[\cdot]]$). So

$$p = p(x), h = h(x).$$

Then, we can compute the left-hand side and right-hand side separately:

$$\begin{aligned}
LHS &= \mathrm{E}[(h-p)^2] = (h-p)^2 \\
RHS &= \mathrm{E}[(h-y)^2] - \mathrm{E}[(p-y)^2] \\
&= \mathrm{E}[(h^2 - 2hy + y^2 - p^2 - y^2 + 2py)] \\
&= h^2 - 2h\underbrace{\mathrm{E}[y]}_{p} - p^2 + 2p\underbrace{\mathrm{E}[y]}_{p} \\
&= h^2 + p^2 - 2hp \\
&= (h-p)^2.
\end{aligned}$$

Hence, LHS = RHS. We are done.

In the classification case we were looking at the classification loss (0-1 loss)

$$\Pr[h(x) \neq y] = \mathrm{E}[1[h(x) \neq y]].$$

Our problem is now to minimize $\mathrm{E}[(h(x)-y)^2]$.

Given the data points $(x_1, y_1), (x_2, y_2).....(x_m, y_m)$ where $y_i \in \mathbb{R}$ we can estimate the expectation $\mathrm{E}[(h(x)-y)^2]$ by empirical average:

$$\hat{\mathrm{E}}[(h(x)-y)^2] = \frac{1}{m}\sum_{i=1}^{m}(h(x_i)-y_i)^2.$$

If we define $L_h(x,y) = (h(x)-y)^2$, then we want

$$\mathrm{E}[L_h] \approx \hat{\mathrm{E}}[L_h]$$

for all h in some class of functions $\mathcal{H}$. One can use Chernoff bounds to show that these will be close for a single h. If $\mathcal{H}$ is finite then we can use the union bound to generalize the result. For $\mathcal{H}$ infinite, VC-style proofs can be used.

So far we tried to justify the use of the loss function that we proposed as a good penalty function. However, we need to find a method to minimize this cost function. In the next section we will see how this problem can be solved for linear predictors.

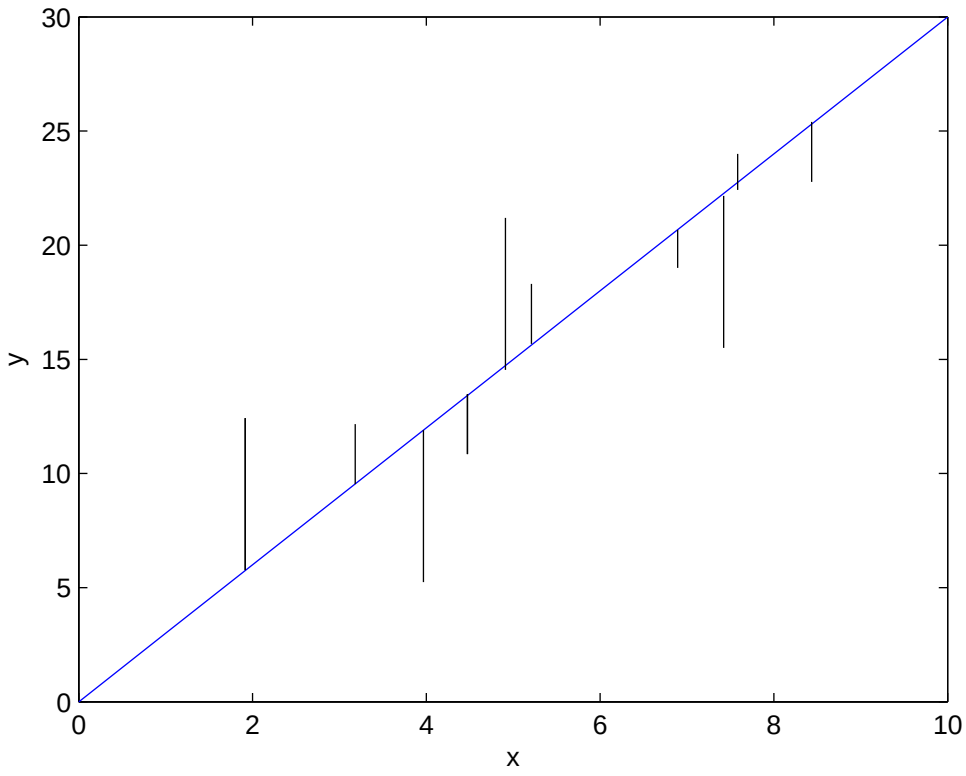

Figure 1: Linear Regression

1.1 Linear Regression

Given $\mathbf{x} \in \mathbb{R}^n$, we want to approximate y by some linear combination of the coordinates of $\mathbf{x}$, i.e., by $\mathbf{w} \cdot \mathbf{x}$. The problem is to find $\mathbf{w}$.

Given the data points $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2).....(\mathbf{x}_m, y_m)$, the problem is to find $\mathbf{w}$ to minimize

$$\Phi = \sum_{i=1}^{m} (\mathbf{w} \cdot \mathbf{x}_i - y_i)^2.$$

We can rewrite Φ in matrix form, where the rows of M are formed by the transpose of the $\mathbf{x}_i$'s :

$$\Phi = \left\| \underbrace{\begin{pmatrix} \leftarrow & \mathbf{x}_1^T & \rightarrow \\ \leftarrow & \mathbf{x}_2^T & \rightarrow \\ \leftarrow & \mathbf{x}_3^T & \rightarrow \\ \vdots & \vdots & \vdots \end{pmatrix}}_{M} \underbrace{\begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \end{pmatrix}}_{\mathbf{w}} - \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \end{pmatrix}}_{\mathbf{b}} \right\|^2$$

$$= \|M\mathbf{w} - \mathbf{b}\|^2$$

We need to minimize this function. Hence by taking the gradients, we get

$$\nabla\Phi = 2M^T(M\mathbf{w} - \mathbf{b}) = 0$$

$$M^T M\mathbf{w} = M^T\mathbf{b}.$$

The unique solution to the last equation occurs under the condition that $M^T M$ is invertible. If $M^T M$ is not invertible we use numerical methods to find an approximate solution. In the case that $M^T M$ is invertible we have:

$$\mathbf{w} = \underbrace{(M^T M)^{-1} M^T}_{\text{pseudoinverse of M}} \mathbf{b}.$$

1.2 Online Linear Regression

The online version of linear regression can be given as following:

Initialize $\mathbf{w}_1$
for $t = 1, 2, \ldots, T$

- get $\mathbf{x}_t \in \mathbb{R}^n$
- predict $\hat{y}_t = \mathbf{w}_t \cdot \mathbf{x}_t$
- observe $y_t \in \mathbb{R}$
- loss=$(\hat{y}_t - y_t)^2$
- update $\mathbf{w}_{t+1}$

Here our goal is to minimize the cumulative loss of the learning algorithm A:

$$L_A = \sum_{t=1}^{T} (\hat{y}_t - y_t)^2.$$

More specifically we want to show that:

$$L_A \leq \min_u L_u + (\text{small amount})$$

where

$$L_{\mathbf{u}} = \sum_{t=1}^{T} (\mathbf{u} \cdot \mathbf{x}_t - y_t)^2.$$

Here, $L_{\mathbf{u}}$ is the loss of a linear predictor defined by $\mathbf{u}$. Thus, the goal is to do almost as well as the best linear predictor (where "best" is defined in hindsight after all of the data has been observed).

1.3 Widrow-Hoff Algorithm (WH)

We will study the Widrow-Hoff algorithm for solving the problem above. An example where this algorithm is used is echo cancelation in telephone networks. WH listens in both directions of a telephone call. At the receiver part it tries to predict the echo so it can cancel it by sending the negative of this signal. Hence the main problem is a prediction problem.

The *WH* algorithm is defined as follows:

- $\mathbf{w}_1 = \mathbf{0}$
- $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta(\mathbf{w}_t \cdot \mathbf{x}_t - y_t)\mathbf{x}_t$

where $\eta > 0$ is a parameter. There are two motivations for this update rule:

Motivation 1:

Our loss function is defined as:

$$L(\mathbf{w}, \mathbf{x}, y) = (\mathbf{w} \cdot \mathbf{x} - y)^2.$$

To minimize the loss function, we can take a step in the direction of steepest descent, i.e., in the direction of the negative gradient. In this case, we have:

$$\nabla_{\mathbf{w}} L = 2(\mathbf{w} \cdot \mathbf{x} - y)\mathbf{x}.$$

This gives the following update equation:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \nabla_{\mathbf{w}} L(\mathbf{w}_t, \mathbf{x}_t, y_t).$$

Motivation 2:

We have two competing goals:

- We want the loss on $(\mathbf{x}_t, y_t)$ to be small. Hence, we want to minimize $(\mathbf{w}_{t+1} \cdot \mathbf{x}_t - y_t)^2$
- We want to stay close to $\mathbf{w}_t$. So, we want to minimize $\|\mathbf{w}_{t+1} - \mathbf{w}_t\|_2^2$

By considering the two goals above we can choose $\mathbf{w}_{t+1}$ to minimize their weighted sum:

$$\eta(\mathbf{w}_{t+1} \cdot \mathbf{x}_t - y_t)^2 + \|\mathbf{w}_{t+1} - \mathbf{w}_t\|_2^2.$$

By taking the derivative and making equal to zero as usual, we get

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta(\mathbf{w}_{t+1} \cdot \mathbf{x}_t - y_t)\mathbf{x}_t.$$

Approximating $\mathbf{w}_{t+1}$ by $\mathbf{w}_t$ on the right-hand side gives the WH update.

By using the update function above we can prove the following theorem.

Theorem 1.3 *Assume* $\|\mathbf{x}_t\|_2 \leq 1$. *Then*

$$L_{WH} \leq \min_{\boldsymbol{u}} \left[\frac{L_{\mathbf{u}}}{1-\eta} + \frac{\|\mathbf{u}\|}{\eta} \right].$$

Moreover, if we divide by T, the total number of time steps, and for η being small we get

$$\frac{L_{WH}}{T} \lesssim (1+\eta)\frac{L_{\mathbf{u}}}{T} + \frac{\|\mathbf{u}\|}{\eta T}.$$

This is basically to say that the rate of the loss of the algorithm is getting close to the rate of loss of the best vector $\mathbf{u}$. The proof is given below.

Proof Fix $\mathbf{u}$. For our potential function, we define

$$\Phi = \|\mathbf{w}_t - \mathbf{u}\|^2.$$

We also define

$$\begin{aligned} \ell_t &= \mathbf{w}_t \cdot \mathbf{x}_t - y_t \\ g_t &= \mathbf{u} \cdot \mathbf{x}_t - y_t. \end{aligned}$$

Then ℓ_t^2 is WH's loss on round t, and g_t^2 is $\mathbf{u}$'s loss. We also define

$$\Delta_t = \mathbf{w}_{t+1} - \mathbf{w}_t = \eta(\mathbf{w}_t.\mathbf{x}_t - y_t)\mathbf{x}_t = \eta\ell_t\mathbf{x}_t.$$

We claim that:

$$\Phi_{t+1} - \Phi_t \leq -\eta\ell_t^2 + \frac{\eta}{1-\eta}g_t^2.$$

We will prove this later. Once proved, the theorem will follow because then:

$$\begin{aligned} -\|\mathbf{u}\|^2 = -\Phi_1 &\leq \Phi_{T+1} - \Phi_1 \\ &= \sum_{t=1}^{T}(\Phi_{t+1} - \Phi_t) \\ &\leq \sum_{t=1}^{T}(-\eta\ell_t^2 + \frac{\eta}{1-\eta}g_t^2) \\ &= -\eta\underbrace{\sum_{t=1}^{T}\ell_t^2}_{L_{WH}} + \frac{\eta}{1-\eta}\underbrace{\sum_{t=1}^{T}g_t^2}_{L_{\mathbf{u}}}. \end{aligned}$$

Solving for L_{WH} gives exactly the statement of the theorem.