Lecture #20

Recall: Given observations $\{(x_i, y_i)\}_{i=1}^n \subset \mathbb{R}^2$, the optimal least-squares linear fit to $(\hat{\beta}_0, \hat{\beta}_1) = \underset{(\beta_0,\beta_1)\in\mathbb{R}^2}{\arg\min} \frac{1}{n}\sum_{i=1}^n |y_i - \beta_0 - \beta_1 x_i|^2$ is given by the parameters

$$\hat{\beta}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

- This can be generalized to vector-valued inputs through some linear algebra.
- Recall that for a matrix $X \in \mathbb{R}^{n\times d}$, the transpose of X is defined as $(X^T)_{jk} = X_{kj}$. For a square matrix $X \in \mathbb{R}^{m\times m}$, the inverse of X, X^{-1}, (if it exists) is defined as $XX^{-1} = X^{-1}X = I$.
- Let us imagine data $\{(\vec{x}_i, y_i)\}_{i=1}^n$ where $\vec{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$. Then we can think that our goal is to learn the best way to predict y given $\vec{x}$, where we are allowed to take linear combinations of the elements of $\vec{x}$.
- That is, we want to choose a coefficient vector $\beta \in \mathbb{R}^d$ s.t.

$$y \approx \langle \beta, \vec{x} \rangle + \beta_0$$
$$= \sum_{j=1}^{d} \beta_j \vec{x}_j + \beta_0 \quad (\bigstar)$$

- In terms of matrices, we can think $\beta \in \mathbb{R}^{(d+1)\times 1}$ (column vector) and $\vec{x} \in \mathbb{R}^{1\times d}$ (row vector). Then $(\bigstar)$ just becomes the matrix multiplication of $\vec{x}$ and β: $y \approx \vec{x}\beta_{1:d} + \beta_0$.

- As before, we are given data $\{(\vec{x}_i, y_i)\}_{i=1}^n$ and want to use it to inform a good choice of β. This can be done as before by choosing $\hat{\beta} = (\hat{\beta}_0, \hat{\beta}_{1:d})$ s.t. $\hat{\beta} = \underset{\beta \in \mathbb{R}^{(d+1)\times 1}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} | y_i - \beta_{1:d}\vec{x}_i - \beta_0 |^2$ $(\bigstar\bigstar)$

- We can convert this to a matrix algebra problem as follows. Let:

- $\vec{y} = (y_1, \dots, y_n)^T \in \mathbb{R}^{n\times 1}$

- $\underline{X} = \begin{pmatrix} 1 & - & \vec{x}_1 & - \\ 1 & - & \vec{x}_2 & - \\ & & \vdots & \\ 1 & - & \vec{x}_n & - \end{pmatrix} \in \mathbb{R}^{n\times(d+1)}$

- $\beta = (\beta_0, \beta_1, \dots, \beta_d)^T \in \mathbb{R}^{(d+1)\times 1}$

- Then $(\bigstar\bigstar) = \underset{\beta}{\operatorname{argmin}} \| \underset{\sim}{\vec{y}} - \underset{\sim}{\underline{X}}\beta \|_2^2$ (vector, vector). This looks hard— an

optimization problem with matrices!

But, it can be solved using matrix calculus:

Theorem: Let $\vec{y}, X, \beta$ be as above. Then $\hat{\beta} = (X^T X)^{-1} X^T \vec{y}$.

Proof: Let $f(\beta) = \|\vec{y} - X\beta\|_2^2$. Our goal is to minimize this with respect to β. [redacted] So, we can expand $\|\vec{y} - X\beta\|_2^2$ and then differentiate:

$$\|\vec{y} - X\beta\|_2^2 = (\vec{y} - X\beta)^T(\vec{y} - X\beta)$$

$$= (\vec{y}^T - \beta^T X^T)(\vec{y} - X\beta)$$

$$= \vec{y}^T\vec{y} - \vec{y}^T X\beta - \beta^T X^T \vec{y} + \beta^T X^T X\beta$$

$$\Rightarrow \frac{\partial}{\partial\beta} f(\beta) = \frac{\partial}{\partial\beta}\left[\vec{y}^T\vec{y} - \vec{y}^T X\beta - \beta^T X^T\vec{y} + \beta^T X^T X\beta\right]$$

$$= \frac{\partial}{\partial\beta}\left[\vec{y}^T\vec{y}\right] - \frac{\partial}{\partial\beta}\left[\vec{y}^T X\beta\right] - \frac{\partial}{\partial\beta}\left[\beta^T X^T\vec{y}\right] + \frac{\partial}{\partial\beta}\left[\beta^T X^T X\beta\right]$$

$$= 0 \qquad - X^T\vec{y} \qquad - X^T\vec{y} \qquad + 2X^T X\beta$$

$$\Rightarrow \frac{\partial}{\partial\beta} f(\beta) = 0 \quad \text{iff} \quad -2X^T\vec{y} + 2X^T X\beta = 0$$

$$\Leftrightarrow \quad X^T X\beta = X^T\vec{y}$$

$$\Leftrightarrow \quad (X^T X)^{-1} X^T\vec{y} = \beta. \quad \blacksquare$$

Remarks: ⚠ Clearly $\frac{\partial}{\partial \vec{z}} \|\vec{z}\|_2^2 = 2\vec{z}$. So, why can we not proceed by solving $\vec{y} - X\beta = 0$? X is not invertible!

⚠!! Here, we are using $\frac{\partial}{\partial \beta}[\beta^T \vec{v}]$

$$= \frac{\partial}{\partial \beta}\left[\sum_{i=1}^{n} \beta_i \vec{v}_i\right]$$

$$= \vec{v}, \text{ for any vector } \vec{v}$$

- So, what does this look like in practice? Well, imagine $\vec{x}$ represents a long list of variables we can use to predict y. So, we get lots of observations $\{(\vec{x}_i, y_i)\}_{i=1}^n$, and develop the predictor

$$\hat{y} = \hat{\beta}_{1:d} \vec{x} + \hat{\beta}_0.$$

Is this a good idea?!

- From the standpoint of "fitting the data," having lots of coordinates in $\vec{x}$ is a good thing - it gives us more fitting power.
- From the standpoint of predicting well on unseen data, it is quite dangerous to use $\vec{x} \in \mathbb{R}^{d\times 1}$ with d too large. Why? Overfitting!
- This is the foundational principal of statistical learning / theoretical machine learning,

the idea that I should predict based on what I observe, with the understanding that I will be evaluated on unseen data.

- In this case, statistical learning theory (e.g. topics in MATH 260) suggest the value of parsimony: picking the simplest model that does well.

- In the context of linear regression, when d is large, there may be unhelpful or redundant features in the data. We may want to try to remove/collapse/minimize the importance of these features if they can be identified.

- In general, the bias in model selection ought to be towards simplicity:

ex. Given observations, which predictor seems more likely to do well in the future?