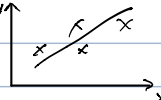
Day 2
9/11/18

Last time:  $h_\theta(x) = \theta_0 + \theta_1 x$ loss/cost function $J(\theta)_{2\times 1} = \sum_{i=1} (y_i - h_\theta(x_i))$

Goal: find $\hat{\theta} = \arg\min J(\theta)$

Normal regression: $y_i \overset{indep}{\sim} N(h_\theta(x_i), \sigma^2)$ or equivalently $y_i = h_\theta(x_i) + \varepsilon_i$, $\varepsilon_i \overset{iid}{\sim} N(0, \sigma^2)$

Question: if we assume Normal distribution, how does it change the loss function?

Estimation by Maximum Likelihood:

$f(y_i | x_i, \theta) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}(y_i - h_\theta(x_i))^2}$ pdf

Reminder: probability distr = f(data/params)
Likelihood = f(params/data)

$$L(\theta | y_1, \dots, y_N, x_1, \dots, x_N) \overset{indep}{=} \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}(y_i - h_\theta(x_i))^2} = \frac{1}{(\sqrt{2\pi}\sigma)^N} e^{-\frac{1}{2\sigma^2}\sum_{i=1}^{N}(y_i - h_\theta(x_i))^2}$$

$$\arg\max_\theta L(\theta | y_1, \dots, y_N, x_1, \dots, x_N) = \arg\max_\theta \log L(\theta, y_1, \dots, y_N, x_1, \dots, x_N)$$

$$= \arg\max_\theta \left[-\frac{1}{2\sigma^2} \sum_{i=1}^{N} (y_i - h_\theta(x_i))^2 \right] = \arg\min_\theta \sum_{i=1}^{N} (y_i - h_\theta(x_i))^2$$

⇒ same criterion as ordinary least squares

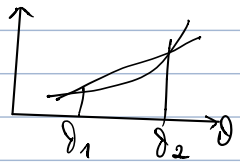
Comments
- Assumption of Normal distribution is not required for OLS (only ML)
- For ML estimation, the marginal distribution of Y is not Normal (only conditional $Y|X$)

least squares optimization with one parameter

$J(\theta) = \sum_{i=1}^{N} (y_i - \theta x_i)^2$
- continuous wrt θ
- smooth (has non-zero 1st and 2nd derivatives)
- convex

A function $J(\theta): \mathbb{R}^p \to \mathbb{R}$ is convex if for any $\theta_1, \theta_2 \in \mathbb{R}^p$ and $t \in [0,1]$

$$J(t\theta_1 + (1-t)\theta_2) \le t J(\theta_1) + (1-t) J(\theta_2)$$



$J(\theta)$ has a single stationary point, i.e. $\theta: \frac{dJ(\theta)}{d\theta} = 0$
⇒ the stationary point is the global minimum

Determining the stationary point:

$$\frac{dJ(\theta)}{d(\theta)} = \frac{d}{d\theta} \sum_{i=1}^{N} (y_i - \theta x_i)^2 = -\sum_{i=1}^{N} 2(y_i - \theta x_i) x_i \overset{\text{set to}}{=} 0$$

$$\Rightarrow -\sum_{i=1}^{N} y_i x_i + \theta \sum_{i=1}^{N} x_i^2 = 0 \quad \Rightarrow \quad \hat{\theta}_1 = \frac{\sum y_i x_i}{\sum x_i^2}$$ - closed-form solution

For strictly convex functions, the stationary point is uniquely identifiable

Optimization with multiple weights:

$$J(\underset{2\times 1}{\theta}) = \sum_{i=1}^{N} (y_i - \theta_0 - \theta_1 x_i)^2$$

Gradient $\nabla J(\underset{2\times 1}{\theta}) = \left[\frac{\partial J(\theta)}{\partial \theta_1}, \frac{\partial J(\theta)}{\partial \theta_2}\right]^T$. Written as a system of equations:

$$\left.\begin{aligned} \frac{\partial J(\theta)}{\partial \theta_0} &= -\sum_{i=1}^{N} 2(y_i - \theta_0 - \theta_1 x_i) = 0 \\ \frac{\partial J(\theta)}{\partial \theta_1} &= -\sum_{i=1}^{N} 2(y_i - \theta_0 - \theta_1 x_i) x_i = 0 \end{aligned}\right\} \text{Normal equations}$$

2 equations
2 unknowns
$\Rightarrow$ unique analytical solution under some conditions

Matrix notation

$$\begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_N \end{bmatrix} = \begin{bmatrix} \overbrace{1 & x_1}^{x_1^T} \\ 1 & x_2 \\ \dots & \dots \\ 1 & x_N \end{bmatrix} \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ \dots \\ e_N \end{bmatrix}$$

$Y_{N\times 1}$ $X_{N\times p}$ $\theta_{p\times 1}$ $e_{N\times 1}$

$$J(\theta) = \sum_{i=1}^{N} (y_i - x_i^T \theta)^2 = \| Y - X\theta \|_2^2$$

$$= \underbrace{(Y - X\theta)^T}_{1\times N} \underbrace{(Y - X\theta)}_{N\times 1}$$

$$= Y^T Y - \theta^T X^T Y - Y^T X\theta + \theta^T X^T X\theta$$

Gradient in matrix notation (i.e., derivative in each dimension of θ)

$$\nabla J(\theta) = -X^T Y - (Y^T X)^T + 2X^T X\theta = -2X^T Y + 2X^T X\theta \overset{\text{set to}}{=} 0$$

$$X^T X\theta - X^T Y = 0 \Rightarrow X^T X\theta = X^T Y \quad \} \text{Normal equations}$$

$$\theta = (X^T X)^{-1} X^T Y \Rightarrow \text{analytical solution}$$

Note: $X^T X$ cannot be singular. I.e., $N > P$
X full rank (no linear dependencies among columns of X)

Prediction

$$\hat{Y} = X\hat{\theta} = \underbrace{X(X'X)^{-1}X'}_{H} Y = HY$$

H - hat matrix
- projection matrix (projects Y on the subspace defined by columns of X)

Note: $\hat{\theta}$ and $\hat{Y}$ are linear combinations of Y. This is why it is called linear regression

Cost of the inverse: $O(\underbrace{p^2 N}_{X'X} + \underbrace{p^3}_{\text{inverse}})$ - expensive if N or p are large

Numeric optimization in high dimensions: gradient descent

Goal: locally compute direction of descent; iterate

Back to the one-parameter case: $J(\theta) = \sum_{i=1}^{N} (y_i - \theta x_i)^2$

Based on Taylor series approximation:

$$J(\theta) = \sum_{t=0}^{\infty} \frac{1}{t!} \frac{dJ(\theta)}{d\theta}\Big|_{\theta_0} (\theta - \theta_0)^t$$

Only keep the first two derivatives:

- $J(\theta) \underset{\text{second-order approximation}}{\approx} J(\theta_0) + \underbrace{(\theta - \theta_0) \frac{dJ(\theta)}{d\theta}\Big|_{\theta_0}}_{\text{first derivative}} + \underbrace{\frac{1}{2}(\theta - \theta_0)^2 \frac{d^2 J(\theta)}{d\theta^2}\Big|_{\theta_0}}_{\text{second derivative}}$

- Stationary point: derivative of the approximation set to 0

$$\frac{dJ(\theta)}{d\theta} \approx \frac{dJ(\theta)}{d(\theta)}\Big|_{\theta_0} + (\theta - \theta_0) \frac{d^2 J(\theta)}{d\theta^2}\Big|_{\theta_0} = 0$$

$$\Rightarrow \theta_1 = \theta_0 - \frac{\frac{dJ(\theta)}{d\theta}\Big|_{\theta_0}}{\frac{d^2 J(\theta)}{d\theta^2}\Big|_{\theta_0}} \quad \text{or} \quad \theta_1 = \theta_0 - \frac{J'(\theta_0)}{J''(\theta_0)}$$

Since the second-order approximation was inaccurate in the first place, we need to iterate through iterations $t = 1, 2, \ldots$ until convergence

$$\theta^{(t+1)} = \theta^{(t)} - \frac{J'(\theta^{(t)})}{J''(\theta^{(t)})}$$ - Newton-Raphson method, or second-order gradient descent

Simplify the algorithm to avoid the second derivative

$$J(\theta) \approx J(\theta_0) + (\theta - \theta_0) \frac{dJ(\theta)}{d\theta}\Big|_{\theta_0} + \frac{1}{2}(\theta - \theta_0)^2 \cdot \underbrace{\frac{1}{\alpha}}_{\text{arbitrary constant replacing the second derivative}}$$

$$\Rightarrow \theta^{(t+1)} = \theta^{(t)} - \underbrace{\alpha}_{\text{learning rate}} \cdot J'(\theta^{(t)})$$

- this is first-order gradient descent

Extending to multiple params of linear regression:

$$\underset{p\times 1}{\theta^{(t+1)}} = \underset{p\times 1}{\theta^{(t)}} - \alpha \underset{p\times 1}{\nabla J(\theta)}\Big|_{\theta^{(t)}} = \underset{p\times 1}{\theta^{(t)}} - \alpha \underset{p\times N}{X^T} \underset{N\times p}{(X}\underset{p\times 1}{\theta^{(t)}} - \underset{N\times 1}{y)}$$

Cost: $O(N)$ expensive for large N

Algo: batch gradient descent for linear regression:

- Initialize θ
- Repeat until convergence {
  - For $j = 1, \dots, p$ {
    $$\theta_j \leftarrow \theta_j + \alpha \frac{d}{d\theta_j} \sum_{i=1}^{N} (y_i - x_i^T \theta)^2$$
    }] can replace with line search
  }

Algo: stochastic gradient descent:

- Initialize θ
- Repeat until convergence {
  - shuffle the order of observations $i = 1, \dots, N$
  - For $i = 1, \dots, N$ {
    - For $j = 1, \dots, p$ {
      $$\theta_j \leftarrow \theta_j + \alpha \frac{d}{d\theta_j} (y_i - x_i^T \theta)^2$$
      }] can replace with line search
    }
  }