Regression

- Classification: given pt. x, predict class (discrete)
- regression: given pt x, predict a numerical value
  - choose:
    1. form of regression function $h(x;w)$ w/ parameters w — hypothesis
       - ↳ like decision function from classification
    2. cost function to optimize
       - ↳ usually based on loss function, e.g. empirical risk = $E[\text{loss}]$

Some regression fns:

1. linear regression $h(x;w,\alpha) = wx + \alpha$
2. polynomial regression (plug in polynomial features into linear regression)
3. logistic regression $h(x;w,\alpha) = S(w \cdot x + \alpha)$, $S(\cdot)$ is sigmoid, $S(\gamma) = \frac{1}{1+e^{-\gamma}}$
   - ↳ LDA produces logistic posterior probabilities
   - ↳ interpolate probabilities, not numbers

Loss functions:

z = prediction $h(x)$, y is label

- (A) Squared error: $(z-y)^2$
- (B) Absolute error: $|z-y|$ — computationally harder to do
- (C) logistic/cross-entropy: $-y \ln z - (1-y)\ln(1-z)$ — z & y must be in same range

Some Cost fns

- (a) $J(h) = \frac{1}{n}\sum_{i=1}^{n} L(h(x_i), y_i)$ — mean loss
- (b) $J(h) = \max_{i=1}^{n} L(h(x_i), y_i)$ — maximum loss
- (c) $J(h) = \sum_{i=1}^{n} \omega_i \cdot L(h(x_i), y_i)$ — weighted sum
- (d) L2 regularization: $J(h) + \lambda \|w\|_2^2$
- (e) L1 regression: $J(h) + \lambda \|w\|_1$

(d), (e): don't trust large weights

Famous regression methods:

- least-squares linear regression: ① + (A) + (a)
- Weighted LS linear: ① + (A) + (c)
- Ridge Regression: ① + (A) + (d)

(the three above): quadratic cost fn. (convex)

- Lasso: ① + (A) + (e) — quadratic program (like SVMs)
- Logistic Regression: ③ + (C) + (a) — convex cost; minimize w/ gradient descent
- Least absolute deviations: ① + (B) + (a)
- Chebychev criterion: ① + (B) + (b)

(the last two): linear programs

Least-Squares Linear Regression Gauss, 1801

Linear regression fn (1) + Squared loss (A) + cost fn (a)

problem: find w, α that minimizes $\sum_{i=1}^{n} (\underbrace{X_i \cdot w + \alpha}_{\text{prediction}} - y_i)^2$

design matrix:

$$\begin{bmatrix} X_{11} & X_{12} & \dots & X_{1d} & 1 \\ \vdots & & \ddots & \vdots & \vdots \\ X_{n1} & \dots & \dots & X_{nd} & 1 \end{bmatrix} \leftarrow \text{point } X_i^T \qquad \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

feature column X_{*d}

Usually, $n > d$ (tall, not wide)

Recall fictitious dimension trick: $h(x) = xw + \alpha$ as

$$\begin{bmatrix} x_1 \dots x_2 & 1 \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \\ \alpha \end{bmatrix}$$

Now $X \in \mathbb{R}^{n \times (d+1)}$ w is a $d+1$ vector

find w that minimizes $\|Xw - y\|^2 = RSS(w)$, residual sum of squares

Optimize w/ Calculus:

$$\min RSS(w) = w^T X^T X w - 2y^T X w + y^T y$$

$$\nabla RSS = 2X^T X w - 2X^T y = 0$$

$$\Rightarrow X^T X w = X^T y \quad \text{(normal equations)}$$

$$\Rightarrow w^* = \underbrace{(X^T X)^{-1} X^T}_{\text{Pseudo inverse of } X:\ X^+ \in \mathbb{R}^{(d+1) \times n}} y$$

at least PSD

If $X^T X$ is singular, this problem is underconstrained
↳ all points lie on a hyperplane, don't use all $d+1$ dimensions

We can use a linear solver to get w^*

Observe: $X^+ X = (X^T X)^{-1} X^T X = I_{d+1}$
Observe: predicted value of y_i is $\hat{y}_i = w \cdot X_i$

What are $\hat{y}_i$?

$$\hat{y} = \begin{bmatrix} \hat{y}_1 \\ \vdots \\ \hat{y}_n \end{bmatrix} = Xw = XX^{+}y = Hy \qquad H \in \mathbb{R}^{n \times n}$$

$H = XX^{+}$ is the hat matrix

If $n > d+1$, H is singular $\Longrightarrow$ no perfect fit w/ hyperplane

Advantages of LS LR:
- easy to compute, just solve linear system
- usually unique & stable solution

Disadvantages
- very sensitive to outliers since errors are squared
- fails if X^TX is singular

Logistic Regression   David Cox, 1958

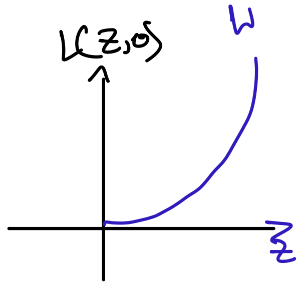
$h = s(X_i \cdot w + \alpha)$ — logistic regression fn ③ + $-y\ln z - (1-y)\ln(1-z)$ — logistic loss fn ② + $\frac{1}{n}\sum L(h(x_i), y_i)$ — cost fn ①

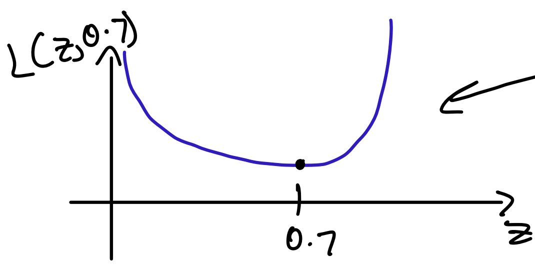
- fits probabilities in range (0, 1)
- usually used for classification
  - ↳ input y_i's might be probabilities, in practice $y_i \in \{0, 1\}$

- QDA & LDA are generative models used to calculate posterior probabilities
- logistic regression: discriminative model
  - ↳ build model of what we want to predict
    - ↳ no need to fit each class

W/ X and w (including fictitious dimension; α is w's last component)

find $w^* = \min_w J = \min \sum_{i=1}^{n} L(s(X_i \cdot w), y_i) = -\sum_{i=1}^{n} \left(y_i \ln s(X_i \cdot w) + (1-y_i)\ln(1 - s(X_i \cdot w)) \right)$

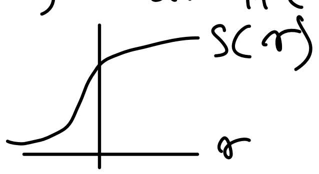


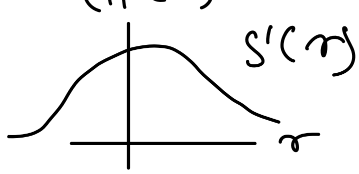


$\leftarrow y_i \notin \{0, 1\}$

$J(w)$ is convex! $\rightarrow$ use gradient descent

$$s'(\gamma) = \frac{d}{d\gamma} \cdot \frac{1}{1+e^{-\gamma}} = \frac{e^{-\gamma}}{(1+e^{-\gamma})^2} = s(\gamma)(1 - s(\gamma))$$





Let $s_i = s(x_i \cdot w)$

$$\hookrightarrow \nabla_w J = -\sum_{i=1}^{n} \left(\frac{y_i}{s_i} \nabla s_i - \frac{1-y_i}{1-s_i} \cdot \nabla s_i \right)$$

$$= -\sum_{i=1}^{n} \left(\frac{y_i}{s_i} - \frac{1-y_i}{1-s_i} \right) s_i(1-s_i)\, x_i$$

$$= -\sum (y_i - s_i)\, x_i$$

element-wise sigmoid of Xw vector

$$= -X^T(y - s(Xw))$$

gradient descent rule: $w \leftarrow w + \epsilon X^T(y - s(Xw))$

SGD: $w \leftarrow w + \epsilon(y_i - s(x_i \cdot w))\, x_i$