Generative Models (Just-so-stories)

- Assume data comes from probabilistic model.
- Reason about outcomes assuming this generative process.

(This is never really true! But it can be instructive)

Example: prediction error.

Assume $y = \vec{w}_\star^T \vec{x} + \text{noise}$

Given data, how well can we predict y?

Let $y_i^{true} = \vec{w}_\star^T \vec{x}_i$

For a learned model $\hat{w}$, $y_i^{pred} = \hat{\vec{w}}^T \vec{x}_i$

prediction error: $\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}(y_i^{pred} - y_i^{true})^2\right]$

In homework, we showed that for OLS prediction error had expected value $\frac{d}{n}$, provided noise was normally distributed w/ mean zero and variance 1.

OUT OF SAMPLE (generalization)

We don't really care about performance on data we've already seen. What about new data.

$$\underbrace{(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)}_{S} \quad \text{what happens on } (\vec{x}_{new}, y_{new})?$$

Clearly old and new data have to be related.

Assume $(\vec{x}, y)$ generated by same random process.

$$\boxed{\text{PROCESS}} \longrightarrow \underbrace{(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n), (\vec{x}_{new}, y_{new})}_{\text{I.I.D.}}$$

Want to find prediction function $f: \mathcal{X} \to \mathcal{Y}$ s.t.

$$R[f] \equiv \mathbb{E}_{(x,y)}[\text{loss}(f(x), y)] \quad \text{is small}$$

Can compute

$$R_S[f] = \frac{1}{n} \sum_{i=1}^{n} \text{loss}(f(x_i), y_i)$$

When if $R[f]$ close to $R_S[f]$?

Law of large numbers

If $z_1, z_2, \ldots, z_n, z$ i.i.d. then

$$\frac{1}{n}\sum_{i=1}^{n} z_i \approx \mathbb{E}[z]$$

How close the sample average is to the average is a matter that depends on many properties of z.

Example; biased coin

$$\text{If} \quad z = \begin{cases} 1 & \text{w.p. } p \quad \text{heads} \\ 0 & \text{otherwise (w.p. } 1-p) \quad \text{tails} \end{cases}$$

after n coin flips, expect to see np heads

true number appears to have Gaussian distribution

$$\Pr(\#\text{ heads} \le pn - t) \le \exp\left(-2\frac{t^2}{n}\right)$$

$$\Pr(\#\text{ heads} \ge pn + t) \le \exp\left(-2\frac{t^2}{n}\right)$$

Example: 2 biased coins, $\mathbb{E}[C_1] = p_1$, $\mathbb{E}[C_2] = p_2$

Flip coin 1 and 2 n times, see H_i heads for C_i

$H_1 \geq H_2$: Is $p_1 \geq p_2$? How can you tell?

Assume $p_2 \geq p_1$

$$\Pr(H_1 \geq H_2) = \Pr(H_1 - H_2 \geq 0)$$

$$= \Pr(H_1 - H_2 \geq -(p_2 - p_1)n + (p_2 - p_1)n)$$

$$\leq \exp\left(-\frac{2(p_2-p_1)^2 n^2}{2n}\right) = \exp(-n(p_2-p_1)^2)$$

If you want to be sure ~~alternative~~ that the probability of $H_1 \geq H_2$ is less than 0.05 when you assume $p_2 \geq p_1$, then you need

$$\boxed{n \geq \frac{3}{(p_2-p_1)^2}}$$

closer two alternatives are, the more you need.

Risk $\quad R[f] = \mathbb{E}_{(x,y)}\left[\operatorname{loss}(f(x), y)\right]$

Empirical Risk $\quad R_S[f] = \frac{1}{n}\sum_{i=1}^{n} \operatorname{loss}(f(x_i), y_i)$

For fixed f: $\quad R_S[f] \approx R[f]$

But when is $\quad \min_f R_S[f] \approx \min_f R[f]$?

This is like the biased coin: more possible functions means we need more data.

Idea: Do whatever you'd like on your data S to produce predictor f_S

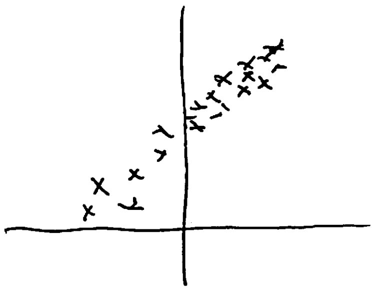
Gather new i.i.d. data S_{NEW} and evaluate

$$R_{S_{NEW}}[f_S] = \frac{1}{|S_{NEW}|}\sum_{i \in S_{NEW}} \operatorname{loss}(f_S(\vec{x}_i), y_i)$$

<u>Example:</u> n example points

$\bar{X}$ features are $\vec{x}_i = \begin{bmatrix} x_i \\ 1 \end{bmatrix}$

$\bar{Z}$ features are $\vec{z}_i = \begin{bmatrix} 1 \\ x_i \\ \vdots \\ x_i^{n-1} \end{bmatrix}$ (square)



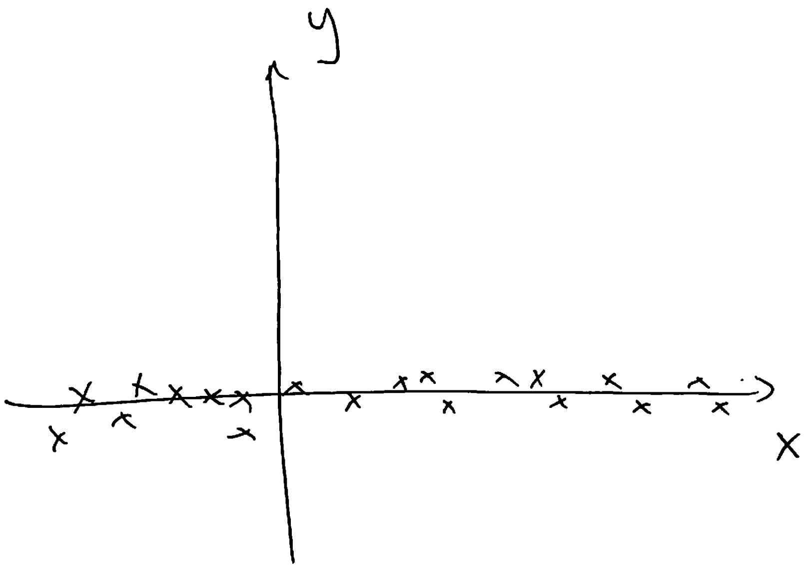
$$\min_{\vec{w}} \|\bar{X}\bar{w} - \bar{y}\| > 0$$

$$\min_{\vec{w}} \|\bar{Z}\bar{w} - \bar{y}\| = 0 \quad !$$

Don't trust training error.

<u>Example:</u> $\arg\min_{\vec{w}} \|\bar{Z}\bar{w} - y\|^2 + \gamma\|\bar{w}\|^2$

Is there a value of γ s.t. the fit is best?



$n = 20$

Assume $y \sim N(0, \sigma^2)$

$\sigma = 0.01$

$$R_S[0] = \frac{1}{n} \sum_{i=1}^{n} y_i^2$$

Alternative: Let $\vec{x}$ = degree 19 polynomials

(20 parameters) $\bar{X} = \begin{bmatrix} \vec{x}_1^T \\ \vdots \\ \vec{x}_n^T \end{bmatrix}$

$$\min_{\vec{w}} \| \bar{X} \vec{w} - \vec{y} \|^2 = 0$$

$$\arg\min \rightarrow \vec{w}_{POLY} = (\bar{X}^T \bar{X})^{-1} \bar{X}^T \vec{y}$$

$$R_S[\vec{w}_{POLY}] = 0$$

Simulating new data:

(1) The holdout method. Break data into chunks: (at random!)

TRAIN SET $\quad n_T$ examples

HOLDOUT SET $\quad n_H$ examples $\qquad n_T + n_H = n$

Algorithm (Train Set) returns f_T

$$\text{Evaluation (Holdout Set)} = \frac{1}{n_H} \sum_{i \in \text{Holdout}} \text{loss}(f_T(x_i), y_i)$$

(Don't test too many hypotheses on one holdout)

(2) CROSS VALIDATION: Break data into F chunks (called folds)

$$\boxed{S} = \left\{ \boxed{S_1} \ \cdots \ \boxed{S_2} \ \cdots \ \boxed{S_F} \right\}$$

for $j = 1, \ldots, F$:

Let train set $= \bigcup_{i \neq j} S_i = S \setminus S_j$

Algorithm (train set) returns $f_{S \setminus S_j}$

Evaluate: $R_j = \frac{1}{|S_j|} \sum_{i \in S_j} \text{loss}(f_{S \setminus S_j}(\vec{x}_i), y_i)$

return $\frac{1}{F} \sum_{j=1}^{F} R_j$

intuition, this should have lower variance.

(3) Leave-one-out error (CV to the extreme)

for $k = 1, \dots, n$:

Let train set $= S \setminus \{(\vec{x}_k, y_k)\}$
(delete kth point from training set)

Algorithm (train set) returns $f_{S \setminus k}$

Evaluate $R_k = \text{loss}(f_{S \setminus k}(\vec{x}_k, y_k))$

return $\frac{1}{n} \sum_{i=1}^{k} R_k$

<u>Example</u> RIDGE REGRESSION

$$\vec{w}_{RIDGE} = \arg\min_{\vec{w}} \frac{1}{n} \sum_{i=1}^{n} (\vec{w}^T \vec{x}_i - y_i)^2 + \gamma \|\vec{w}\|^2$$

How do you pick γ? Choose γ to minimize holdout, cross validation, or leave-one-out error

Typically: only look at few γ, ~~logarithmical~~ spaced logarithmically (don't look at <u>all</u> of the γ)

Should have $\gamma < \sigma_{max}(\bar{X})$
data matrix