Judging Test error

- Training error of a classifier f

$$\frac{1}{n}\sum_{i=1}^{n} 1_{f(X_i)\neq Y_i}$$

Training Data

$$\{X_i, Y_i\}_{i=1}^{n}$$

- What about test error?

  Can't compute it.

- How can we know classifier is not overfitting?

  Hold-out or Cross-validation

Hold-out method

Can judge test error by using an independent sample of data.

Hold - out procedure:

n data points available $D \equiv \{X_i, Y_i\}_{i=1}^n$

1) Split into two sets (randomly and preserving label proportion):

Training dataset $D_T = \{X_i, Y_i\}_{i=1}^m$

Validation/Hold-out dataset $D_V = \{X_i, Y_i\}_{i=m+1}^n$

often m = n/2

2) Train classifier on D_T. Report error on validation dataset D_V.

Overfitting if validation error is much larger than training error

Hold-out method

Drawbacks:

- May not have enough data to afford setting one subset aside for getting a sense of generalization abilities
- Validation error may be misleading (bad estimate of test error) if we get an “unfortunate” split

Limitations of hold-out can be overcome by a family of sub-sampling methods at the expense of more computation.

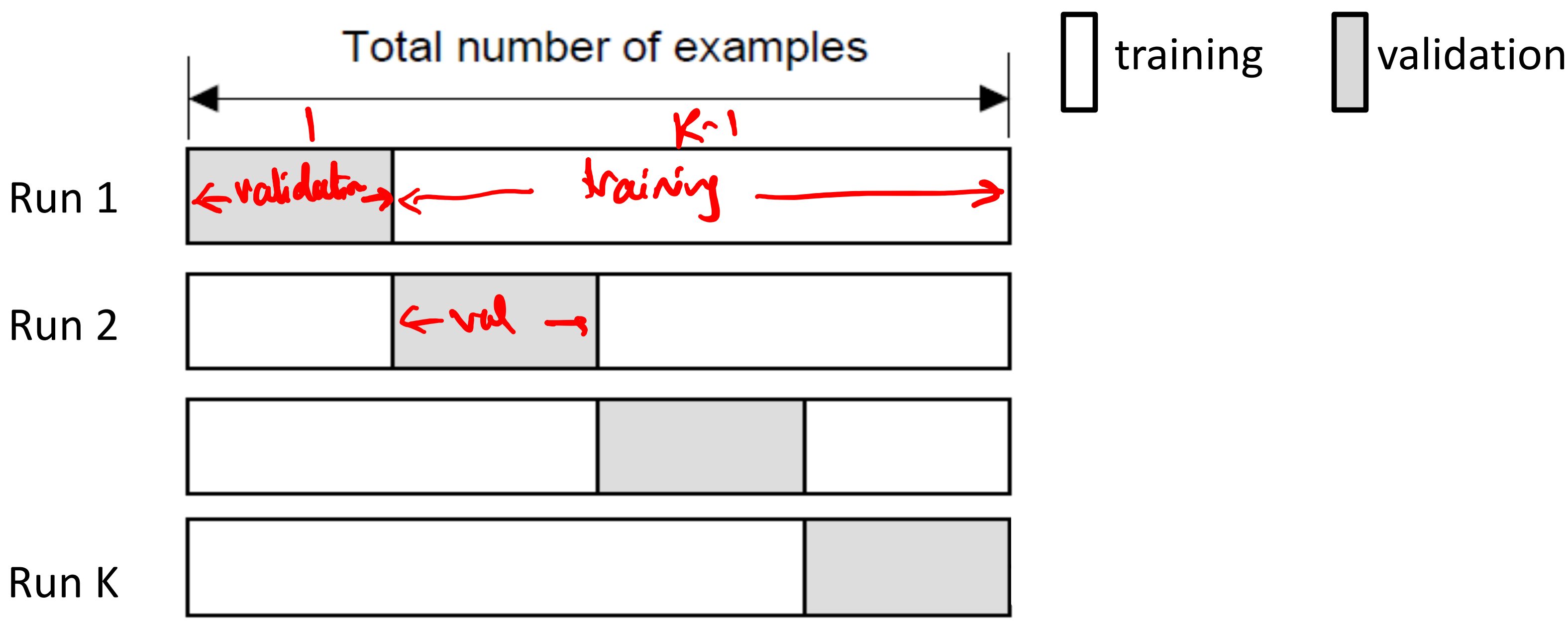
Cross-validation

K-fold cross-validation

Create K-fold partition of the dataset.

Do K runs: train using K-1 partitions and calculate validation error on remaining partition (rotating validation partition on each run).

Report average validation error

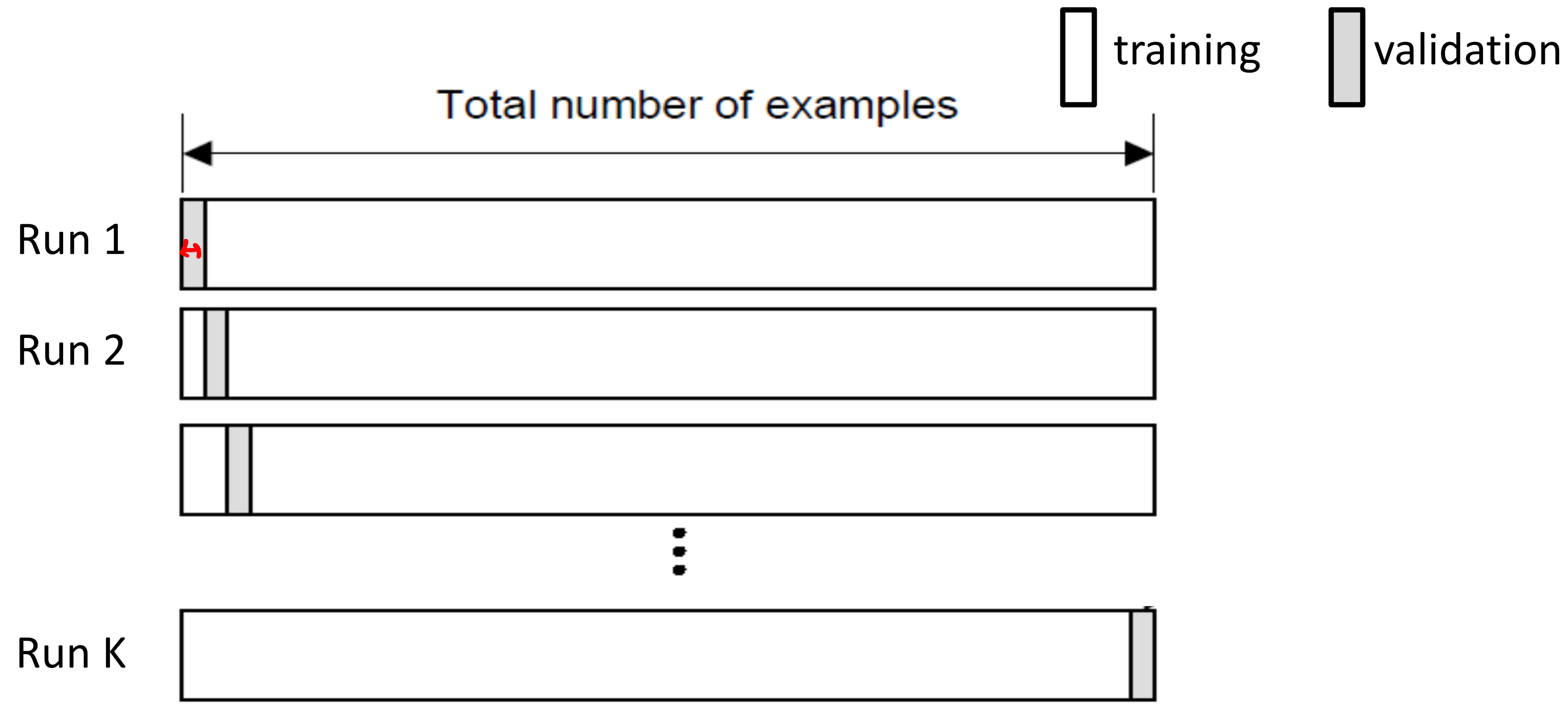


Cross-validation

Leave-one-out (LOO) cross-validation

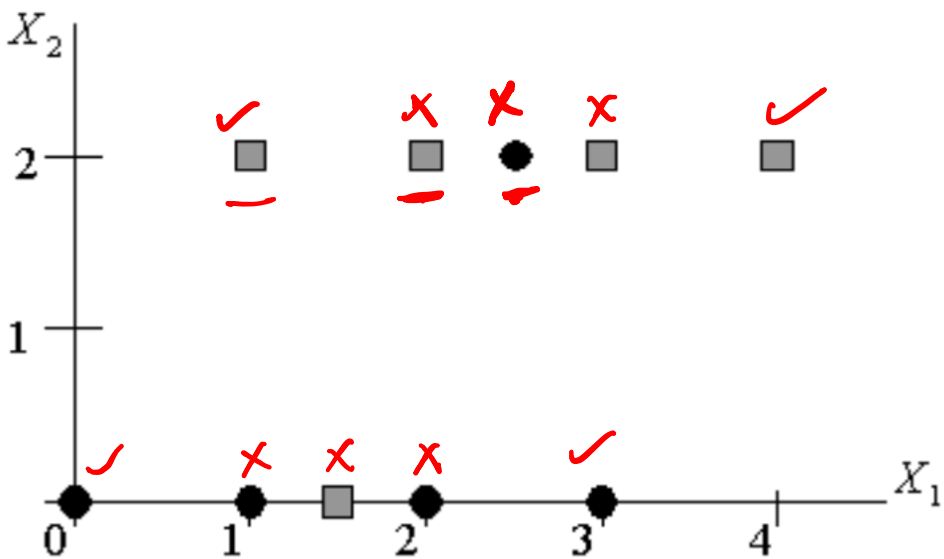
Special case of K-fold with K=n partitions

Equivalently, train on n-1 samples and validate on only one sample per run for n runs



Cross-validation

What is the leave-one-out cross-validation error of the given classifiers on the following dataset?



- **Poll 1:** Depth 1 Decision tree using best feature
- **Poll 2:** 1-NN classifier

Cross-validation

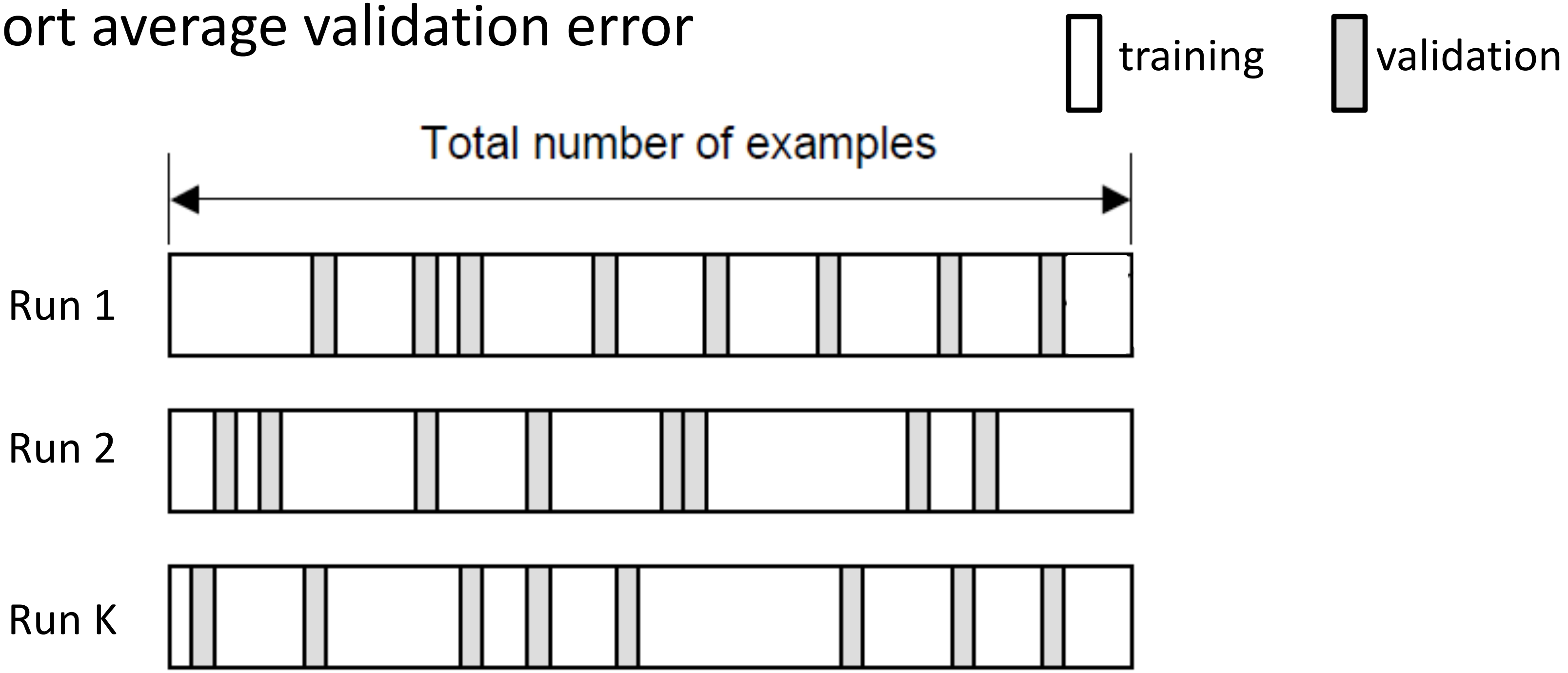
Random subsampling

Randomly subsample a fixed fraction αn $(0 < \alpha < 1)$ of the dataset for validation.

Compute validation error with remaining data as training data.

Repeat K times

Report average validation error



Practical Issues in Cross-validation

How to decide the values for K and α?

- Large K

  + Validation error can approximate test error well

  - Observed validation error will be unstable (few validation pts)

  - The computational time will be very large as well (several runs)

- Small K

  + The #runs and, therefore, computation time are reduced

  + Observed validation error will be stable (many validation pts)

  - Validation error cannot approximate test error well

Common choice: K = 10, α = 0.1 ☺

Model selection using Hold-out/Cross-validation

- Train models of different complexities and evaluate their validation error using hold-out or cross-validation

- Pick model with smallest validation error (averaged over different runs for cross-validation)

➢ When using hold-out or cross-validation for model selection, test error should be reported using independent data

What you should know

- Estimating test error using
  - hold-out
  - cross-validation
- Bias-variance tradeoff
- Model selection using
  - hold-out
  - cross-validation

Linear Regression

Aarti Singh

Machine Learning 10-701
Mar 20, 2023

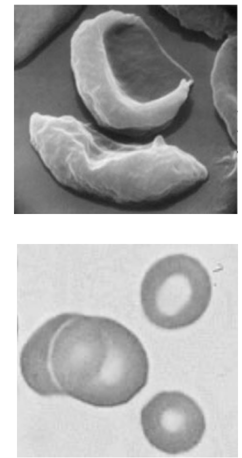
Supervised Learning Tasks

Classification

Sports
Science
News

X = Document

Y = Topic

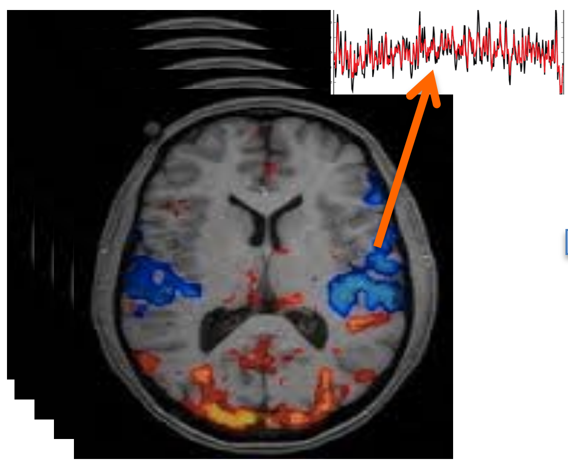


Anemic cell
Healthy cell

X = Cell Image

Y = Diagnosis

Regression

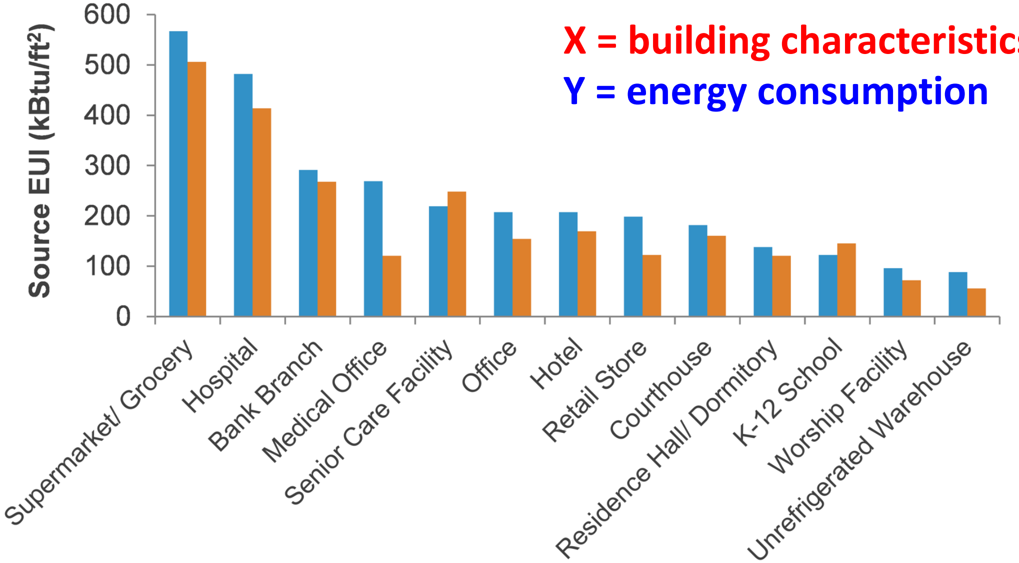


Y = Age of a subject

X = Brain Scan

Regression Tasks

Estimating Energy Usage



X = building characteristics
Y = energy consumption

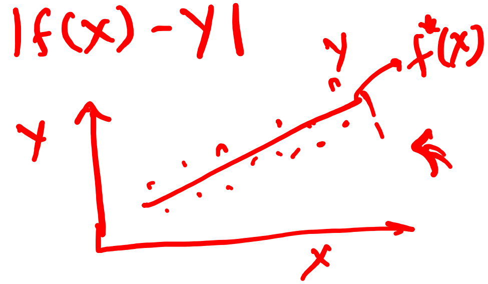
energystar.gov

Estimating Contamination

X = new location
Y = sensor reading

Mean Squared Error (MSE) Minimization

$|f(x) - Y|$



$= E[Y|X]$

Optimal predictor: $$f^* = \arg\min_{f} \mathbb{E}_{XY}[(f(X) - Y)^2]$$

$$\underline{E[(f(x)-Y)^2]} = E[((\underbrace{f(x) - E[Y|X]}_{a}) + (\underbrace{E[Y|X] - Y}_{b}))^2] \qquad (a+b)^2$$

$$= E[(f(x) - E[Y|X])^2 + \cancel{(E[Y|X] - Y)^2} + \underbrace{2(f(x) - \cancel{E[Y|X]})(E[Y|X] - Y)}_{c}]$$

$$c \quad E_{XY}[(f(x) - E[Y|X])(E[Y|X] - Y)]$$

$$= E_X[E_{Y|X}[(f(x) - E[Y|X])(E[Y|X] - Y)]] = 0$$

$$f^* = \arg\min_f E[(f(x) - E[Y|X])^2] = E[Y|X]$$

Mean Squared Error (MSE) Minimization

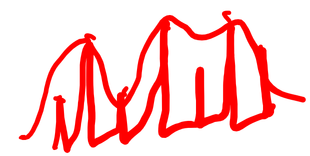
Optimal predictor:

$$f^* = \arg\min_f \mathbb{E}_{XY}[(f(X) - Y)^2] = E[Y|X]$$

Empirical Minimizer:

$$\widehat{f}_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n}\sum_{i=1}^{n}(f(X_i) - Y_i)^2$$

Empirical mean



Law of Large Numbers:

$$\boxed{\frac{1}{n}\sum_{i=1}^{n}[\text{loss}(Y_i, f(X_i))] \xrightarrow{n \to \infty} \mathbb{E}_{XY}[\text{loss}(Y, f(X))]}$$

Restrict class of predictors

Optimal predictor: $$f^* = \arg\min_f \mathbb{E}[(f(X) - Y)^2]$$

Empirical Minimizer: $$\widehat{f}_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n}\sum_{i=1}^{n}(f(X_i) - Y_i)^2$$

Class of predictors

- Why?

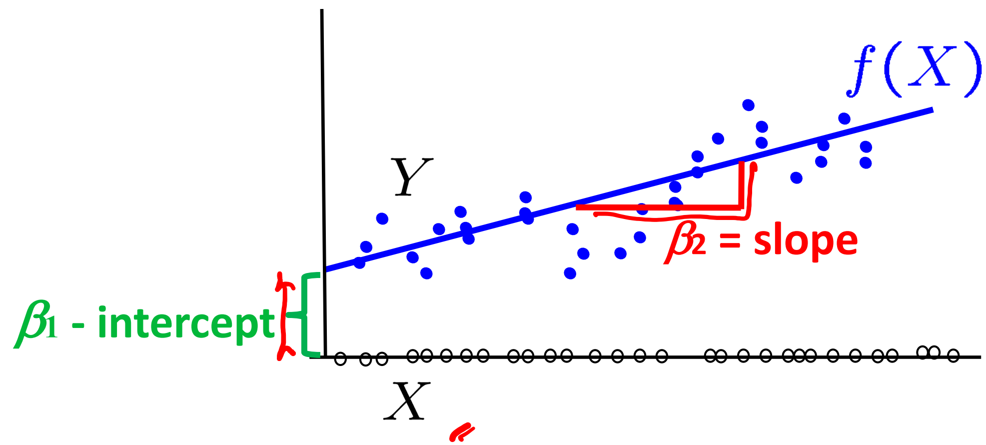
$\mathcal{F}$ - Class of Linear functions
- Class of Polynomial functions
- Class of nonlinear functions

Linear Regression

$$\widehat{f}_n^L = \arg\min_{f \in \mathcal{F}_L} \frac{1}{n} \sum_{i=1}^{n} (f(X_i) - Y_i)^2$$

Least Squares Estimator

$\mathcal{F}_L$ - Class of Linear functions



Uni-variate case:

$$f(X) = \beta_1 + \beta_2 X$$

Multi-variate case:

p features

$$f(X) = f(X^{(1)}, \ldots, X^{(p)}) = \beta_1 \cancel{X^{(1)}} + \beta_2 X^{(2)} + \cdots + \beta_p X^{(p)}$$

1

$$= X\beta \qquad \text{where} \quad X = [X^{(1)} \ldots X^{(p)}], \quad \beta = [\beta_1 \ldots \beta_p]^T$$

Linear Regression (Matrix-vector form)

$$\widehat{f}_n^L = \arg\min_{f \in \mathcal{F}_L} \frac{1}{n} \sum_{i=1}^{n} (f(X_i) - Y_i)^2 \qquad f(X_i) = X_i \beta$$

$$\widehat{\beta} = \arg\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} (X_i \beta - Y_i)^2 \qquad \widehat{f}_n^L(X) = X\widehat{\beta}$$

$$= \arg\min_{\beta} \frac{1}{n} (\mathbf{A}\beta - \mathbf{Y})^T (\mathbf{A}\beta - \mathbf{Y})$$

$$A\beta = \begin{bmatrix} X_1\beta \\ X_2\beta \\ \vdots \\ X_n\beta \end{bmatrix}_{n \times 1}$$

training data matrix

$$\underset{n \times p}{\mathbf{A}} = \begin{bmatrix} X_1 \\ \vdots \\ X_n \end{bmatrix} = \begin{bmatrix} X_1^{(1)} & \dots & X_1^{(p)} \\ \vdots & \ddots & \vdots \\ X_n^{(1)} & \dots & X_n^{(p)} \end{bmatrix} \overset{\beta}{\boxed{}} \qquad \underset{n \times 1}{\mathbf{Y}} = \begin{bmatrix} \mathbf{Y}_1 \\ \vdots \\ \mathbf{Y}_n \end{bmatrix}$$

$$\underset{n \times p}{A} \; \underset{p \times 1}{\beta}$$

Linear Regression

$$\widehat{\beta} = \arg\min_{\beta} \frac{1}{n}(\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y}) = \arg\min_{\beta} J(\beta)$$

$$J(\beta) = (\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y})$$

A = training data matrix

$$= (\beta^T A^T - Y^T)(A\beta - Y)$$

$$= \beta^T \underbrace{A^T A}\beta \cdots$$

- Poll

z^2 $\quad -z^2 \quad$ ✓ az^2, $a \geq 0$

Is the objective convex in β?

A) Convex, quadratic in β
B) Non-convex, A may not be positive semi definite –
C) Depends on conditioning (ratio of max:min eigenvalues) of A^TA
D) Convex, A^TA is positive semi definite

$$\underbrace{v^T A^T A v}_{= \|Av\|^2} \qquad \frac{v^T M v}{v^T v}$$

Linear Regression

① $\dfrac{\partial z^T\beta}{\partial \beta} = z$ ② $\dfrac{\partial \beta^T M \beta}{\partial \beta} = (M+M^T)\beta$

$= 2M\beta$ if $M = M^T$

$$\widehat{\beta} = \arg\min_{\beta} \frac{1}{n}(\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y}) = \arg\min_{\beta} J(\beta)$$

$$J(\beta) = (\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y}) = \beta^T A^T A \beta - \underbrace{Y^T A}_{z^T}\beta - \underbrace{\beta^T A^T Y}_{= Y^T A \beta} + Y^T Y$$

$$\frac{\partial J(\beta)}{\partial \beta} = 2(A^T A)\beta - 2A^T Y \Big|_{\widehat{\beta}} = 0$$

$$(A^T A)\widehat{\beta} = A^T Y$$

$$\left.\frac{\partial J(\beta)}{\partial \beta}\right|_{\widehat{\beta}} = 0$$

Linear regression solution satisfies Normal Equations

$$Y = \begin{bmatrix} Y_1 \\ \vdots \\ Y_n \end{bmatrix} \qquad A = \begin{bmatrix} X_1^{(1)} & \cdots & X_1^{(p)} \\ \vdots & & \vdots \\ X_n^{(1)} & \cdots & X_n^{(p)} \end{bmatrix}$$

$$(\mathbf{A}^T\mathbf{A})\widehat{\beta} = \mathbf{A}^T\mathbf{Y}$$

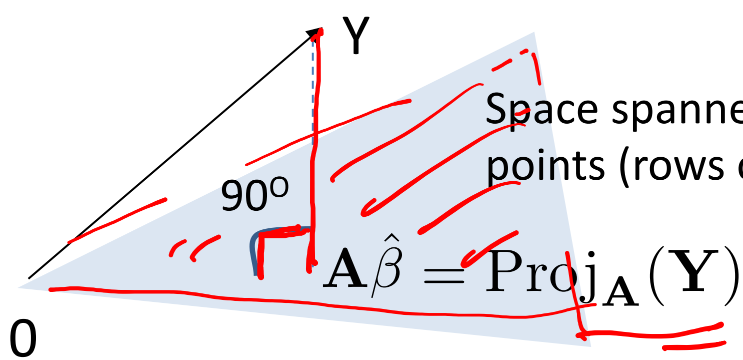
p x p   p x1   p x1

If $(\mathbf{A}^T\mathbf{A})$ is invertible,

$$\boxed{\widehat{\beta} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{Y} \qquad\qquad \widehat{f}_n^L(X) = X\widehat{\beta}}$$

$$\hat{f}_n^L(A) = A\hat{\beta} = \underbrace{A(A^TA)^{-1}A^TY}_{= \mathrm{Proj}_A(Y)}$$

Predicted labels for training points $\mathbf{A}\hat{\beta} = \mathrm{Proj}_{\mathbf{A}}(\mathbf{Y})$



Linear regression solution satisfies Normal Equations

$$\underbrace{(\mathbf{A}^T\mathbf{A})}_{p \times p}\underbrace{\widehat{\beta}}_{p \times 1} = \underbrace{\mathbf{A}^T\mathbf{Y}}_{p \times 1}$$

If $(\mathbf{A}^T\mathbf{A})$ is invertible,

$$\boxed{\widehat{\beta} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{Y} \checkmark \qquad \widehat{f}_n^L(X) = X\widehat{\beta}}$$

Later: When is $(\mathbf{A}^T\mathbf{A})$ invertible ?

Now: What if $(\mathbf{A}^T\mathbf{A})$ is invertible but expensive (p very large)?

Gradient Descent

Even when $(\mathbf{A}^T\mathbf{A})$ is invertible, might be computationally expensive if **A** is huge.

$$\widehat{\beta} = \arg\min_{\beta} \frac{1}{n}(\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y}) = \arg\min_{\beta} J(\beta)$$

Since J(β) is convex, move along negative of gradient

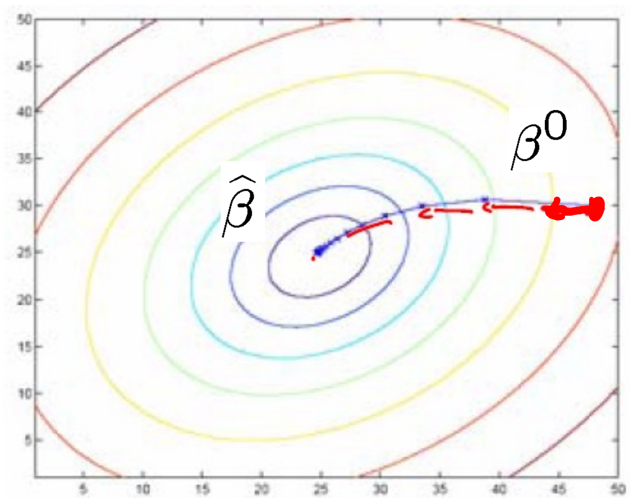
Initialize: β^0

step size

Update: $$\beta^{t+1} = \beta^t - \frac{\alpha}{2}\left.\frac{\partial J(\beta)}{\partial \beta}\right|_t$$

$$= \beta^t - \alpha\, \mathbf{A}^T(\underbrace{\mathbf{A}\beta^t - Y}_{0 \text{ if } \widehat{\beta} = \beta^t})$$

predicted labels



Stop: when some criterion met e.g. fixed # iterations, or $\left.\frac{\partial J(\beta)}{\partial \beta}\right|_{\beta^t} < \varepsilon$.

Least Square solution satisfies Normal Equations

$$\left.\frac{\partial J(\beta)}{\partial \beta}\right|_{\widehat{\beta}} = 0 \qquad \text{gives} \qquad \underset{p \times p}{(\mathbf{A}^T\mathbf{A})}\underset{p \times 1}{\widehat{\beta}} = \underset{p \times 1}{\mathbf{A}^T\mathbf{Y}}$$

If $(\mathbf{A}^T\mathbf{A})$ is invertible,

1) If dimension p not too large, analytical solution:

$$\boxed{\widehat{\beta} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{Y} \qquad \widehat{f}_n^L(X) = X\widehat{\beta}}$$

2) If dimension p is large, computing inverse is expensive $O(p^3)$

Gradient descent since objective is convex ($\mathbf{A}^T\mathbf{A} \succeq 0$)

$$\boxed{\begin{aligned} \beta^{t+1} &= \beta^t - \frac{\alpha}{2}\left.\frac{\partial J(\beta)}{\partial \beta}\right|_t \\ &= \beta^t - \alpha\ \mathbf{A}^T(\mathbf{A}\beta^t - Y) \end{aligned}}$$