STABILITY AND NUMERICAL ASPECTS OF LEAST SQUARES

Dr. Matthieu R Bloch

Monday, November 29, 2021

LOGISTICS

General announcements

- Assignment 6 posted (*last assignment*)
- Due December 7, 2021 for bonus, deadline December 10, 2021
- 3 lectures left
- Let me know what’s missing

Midterm 2 / Assignment 5

- Grades posted this week

WHAT'S ON THE AGENDA FOR TODAY?

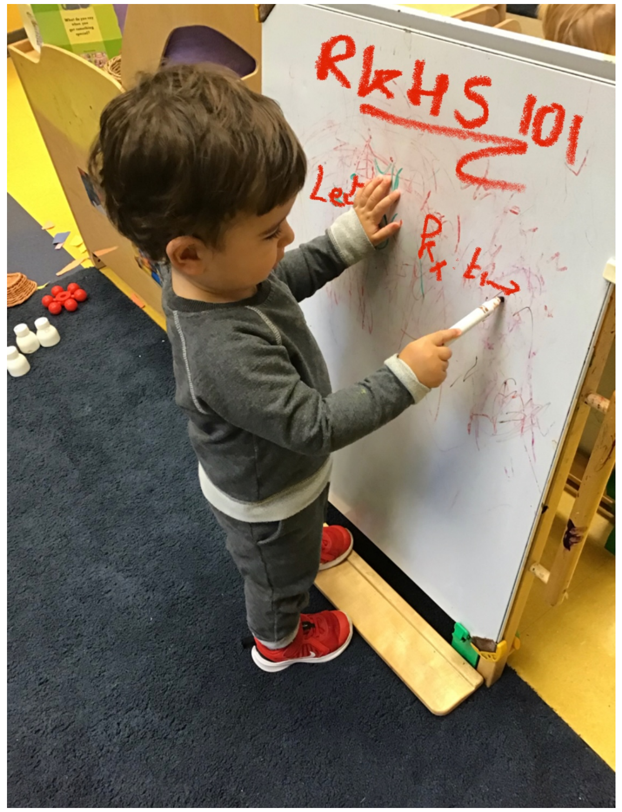
Last time:

- Numerical considerations

Today:

- (Fast discussion) of additional numerical considerations

Reading: lecture notes 14/15/16



Toddlers can do it!

EASY SYSTEMS

Diagonal system

- $\mathbf{A} \in \mathbb{R}^{n \times n}$ invertible and diagonal
- $O(n)$ complexity

Orthogonal system

- $\mathbf{A} \in \mathbb{R}^{n \times n}$ invertible and orthogonal
- $O(n^2)$ complexity

Lower triangular system

- $\mathbf{A} \in \mathbb{R}^{n \times n}$ invertible and lower diagonal
- $O(n^2)$ complexity

General strategy: factorize $\mathbf{A}$ to recover some of the structures above

FACTORIZATIONS

LU factorization

Recall: $A \overset{!}{=} LU = \begin{pmatrix} * & & 0 \\ & \ddots & \\ * & & * \end{pmatrix} \begin{pmatrix} * & & * \\ & \ddots & \\ 0 & & * \end{pmatrix}$

Cost $O(N^3)$

$Ax = b \iff L\underbrace{Ux}_{\overset{!}{=}w} = b$

$\iff \begin{cases} Lw = b \\ Ux = w \end{cases}$

Example: LU factorization $\equiv$ Gaussian elimination + bookkeeping

$$A = \begin{pmatrix} 2 & 1 & -1 \\ -3 & -1 & 2 \\ -2 & 1 & 2 \end{pmatrix}$$

(1) Compute $\underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}}_{L_1} A = \underbrace{\begin{pmatrix} 2 & 1 & -1 \\ -3 & -1 & 2 \\ 0 & 2 & 1 \end{pmatrix}}_{A_1}$

(2) Compute $\underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 3/2 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{L_2} A_1 = \underbrace{\begin{pmatrix} 2 & 1 & -1 \\ 0 & 1/2 & 1/2 \\ 0 & 2 & 1 \end{pmatrix}}_{A_2}$

(3) Compute $\underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -4 & 1 \end{pmatrix}}_{L_3} A_2 = \underbrace{\begin{pmatrix} 2 & 1 & -1 \\ 0 & 1/2 & 1/2 \\ 0 & 0 & -1 \end{pmatrix}}_{A_3 \overset{!}{=} U}$

(4) $\underbrace{L_3 L_2 L_1}_{\text{lower triangular}} A = U$

(5) $A = LU$ w/ $L = L_1^{-1} L_2^{-1} L_3^{-1}$

FACTORIZATIONS

LU factorization

Cholesky factorization

Consider a symmetric matrix $A \in \mathbb{R}^{n \times n}$ semidefinite positive

Cholesky factorization. $A = LL^T$ w/ L lower triangular

Illustration: $A = (a_{ij})$ 1) find a lower triangular matrix R_1 such that

$$R_1 A = \begin{pmatrix} \sqrt{a_{11}} & a'_{12} & \cdots & a'_{1n} \\ 0 & & & \\ \vdots & & & \\ 0 & a'_{n2} & \cdots & a'_{nn} \end{pmatrix}$$

2) $$R_1 A R_1^T = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & & & \\ \vdots & & A_1 & \\ 0 & & & \end{pmatrix}$$

3) iterating $\underbrace{R_n R_{n-1} \cdots R_1}_{\text{all lower triangular matrices}} A R_1^T \cdots R_n^T = I_n$

Set $R = R_n \times R_{n-1} \cdots R_1$ so that $RAR^T = I$ and set $L = R^{-1}$

FACTORIZATIONS

LU factorization

Cholesky factorization

QR decomposition

Let $A \in \mathbb{R}^{n \times n}$; QR decomposition is $A = \underset{\text{orthogonal}}{Q}\ \underset{\text{upper triangular}}{R}$

(Some Gram-Schmidt hidden in the background)

FACTORIZATIONS

LU factorization

Cholesky factorization

QR decomposition

SVD and eigenvalue decompositions

COMPUTING EIGENVALUE DECOMPOSITIONS FOR SYMETRIC MATRICES

Many techniques: we shall only discuss one based on *power iterations*

Let $A \in \mathbb{R}^{n\times n}$ be symmetric positive semi definite

We know that $A = V\Lambda V^T$ w/ V orthogonal and Λ diagonal $= \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix}$ $\lambda_1 \geq \lambda_2 \cdots \geq \lambda_n$

and $V = \begin{pmatrix} | & | & & | \\ v_1 & v_2 & \cdots & v_n \\ | & | & & | \end{pmatrix}$

We want to find λ_1 and v_1

Algorithm: set $q_0 \in \mathbb{R}^n$ w/ $\|q_0\|_2 = 1$ s.t. $q_0 \not\perp v_1$
$k=1$
While "precision not reached" do
$z_k \leftarrow A q_{k-1}$
$q_k \leftarrow \frac{z_k}{\|z_k\|_2}$
$\gamma_k \leftarrow q_k^T A q_k$

Claim: $q_k \xrightarrow[k\to\infty]{} v_1$ $\gamma_k \xrightarrow[k\to+\infty]{} \lambda_1$

Intuition: note that $q_k = \frac{A^k q_0}{\|A^k q_0\|_2}$

Recall: $q_0 = \sum_{i=1}^n \alpha_i v_i$ in the orthobasis of eigenvectors $\{v_i\}_{i=1}^n$
w/ $\alpha_i = \langle q_0, v_i \rangle$

Hence $q_k = \frac{\sum_{i=1}^n \alpha_i \lambda_i^k v_i}{\sqrt{\sum_{i=1}^n \alpha_i^2 \lambda_i^{2k}}} = \frac{\alpha_1 \lambda_1^k}{\alpha_1 \lambda_1^k} \frac{v_1 + \sum_{i=2}^n \frac{\alpha_i \lambda_i^k}{\alpha_1 \lambda_1^k} v_i}{\sqrt{1 + \sum_{i=2}^n \frac{\alpha_i^2 \lambda_i^{2k}}{\alpha_1^2 \lambda_1^{2k}}}}$

$\longrightarrow v_1$ if $\lambda_1 > \lambda_2$

$q_k^T A q_k \longrightarrow v_1^T A v_1 = \lambda_1$

How do we evaluate <u>all</u> eigenvectors and eigenvalues ?

<u>Naive approach</u>: start w/ $Q_0 \in \mathbb{R}^{n\times n}$ w/ orthogonal columns

$Z_k = A Q_0$
normalize Z_k to get orthonormal columns

<u>Trick to make this more efficient</u>: Start w/ Q_0 orthonormal

$Z_k = A Q_{k-1}$

$[Q_k, R_k] = \text{qrdecomposition}(Z_k)$ so that $Z_k = Q_k R_k$

Then as $k \to +\infty$ $Q_k \to V$ and $\Gamma_k \triangleq Q_k^T A Q_k \to \Lambda$

Example: Assume $A \in \mathbb{R}^{n \times n}$ is circulant

$$A = \begin{pmatrix} a_1 & \cdots & & a_n \\ a_n & a_1 & \cdots & a_{n-1} \\ & \ddots & \ddots & \ddots \end{pmatrix}$$

special case of Toeplitz

You can diagonalize A in an orthobasis of eigenvectors that is independent of $\{a_i\}_{i=1}^n$

$$Ax = \begin{pmatrix} a_1 & \cdots & a_n \\ a_n & \ddots & \ddots \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \left(\sum_{i=1}^{n} a_{ij} x_j \right)_{i,j} = \left(\text{circular convolution} \right)$$

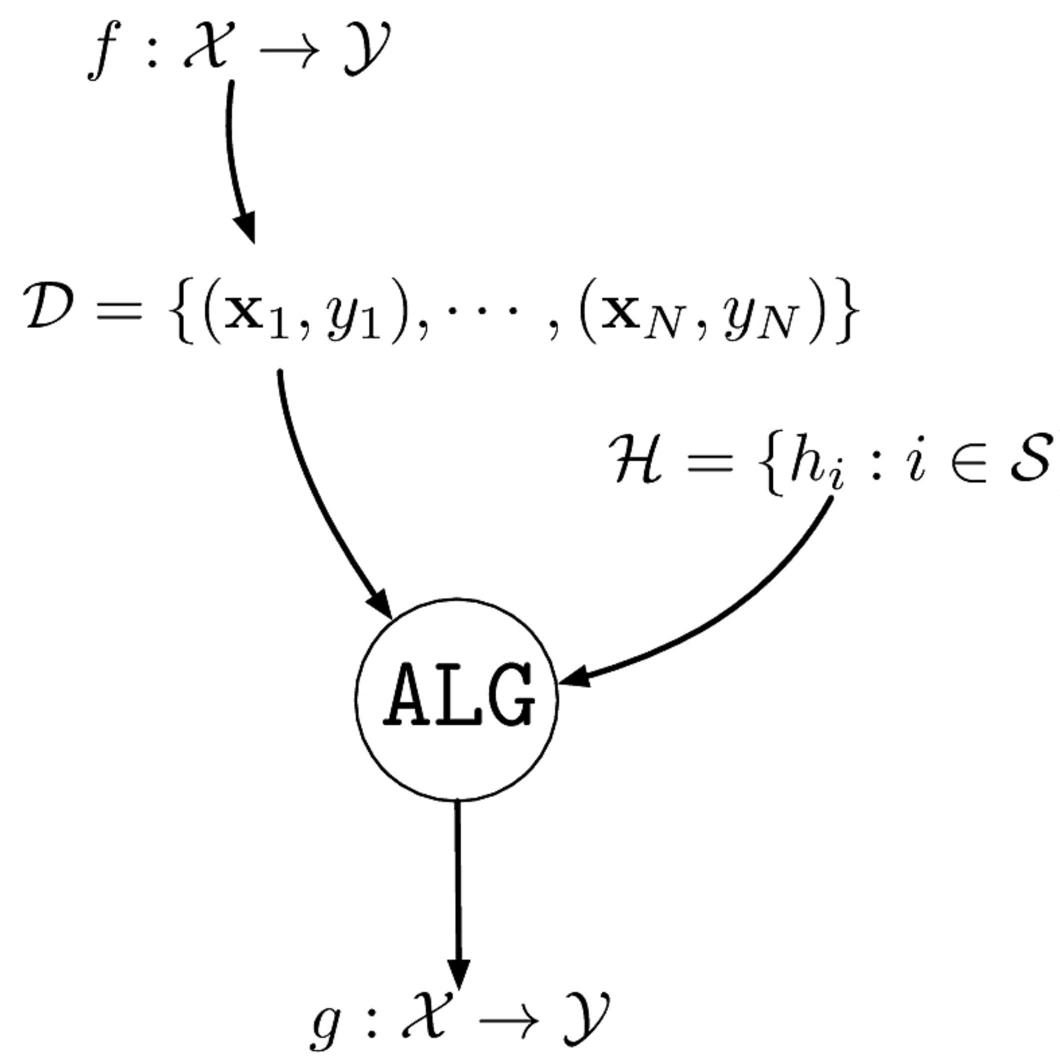
COMPONENTS OF SUPERVISED MACHINE LEARNING

1. An *unknown function* $f : \mathcal{X} \to \mathcal{Y} : \mathbf{x} \mapsto y = f(\mathbf{x})$ to learn
   - The formula to distinguish cats from dogs
2. A dataset $\mathcal{D} \triangleq \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)\}$
   - $\mathbf{x}_i \in \mathcal{X} \triangleq \mathbb{R}^d$: picture of cat/dog
   - $y_i \in \mathcal{Y} \triangleq \mathbb{R}$: the corresponding label cat/dog
3. A *set of hypotheses* $\mathcal{H}$ as to what the function could be
   - Example: deep neural nets with AlexNet architecture
4. An *algorithm* $\texttt{ALG}$ to find the best $h \in \mathcal{H}$ that explains f

Terminology:

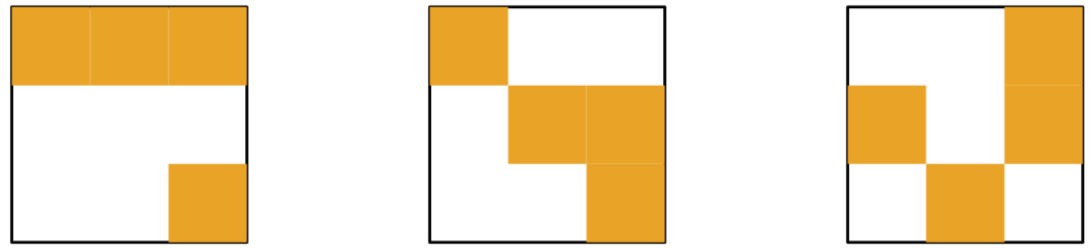
- $\mathcal{Y} = \mathbb{R}$: *regression* problem
- $|\mathcal{Y}| < \infty$: *classification* problem
- $|\mathcal{Y}| = 2$: *binary classification* problem

The goal is to *generalize*, i.e., be able to classify inputs we have *not* seen.



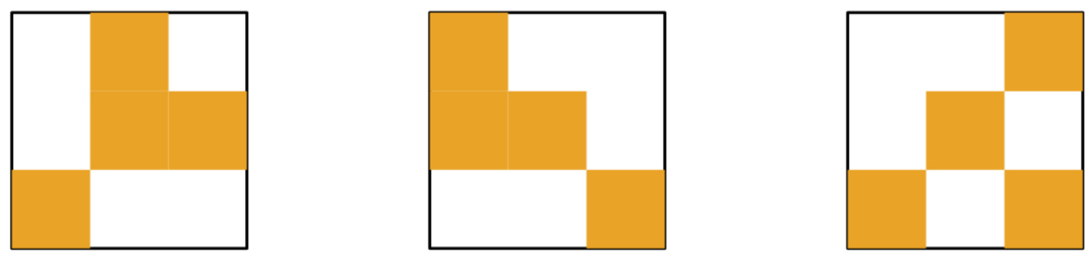
Learning model #1

A LEARNING PUZZLE



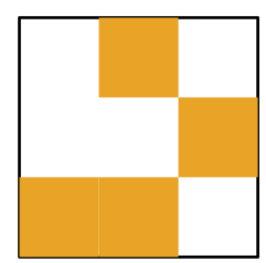
$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$

$$f(\mathbf{x}_1) = f(\mathbf{x}_2) = f(\mathbf{x}_3) = +1$$



$\mathbf{x}_4$ $\mathbf{x}_5$ $\mathbf{x}_6$

$$f(\mathbf{x}_4) = f(\mathbf{x}_5) = f(\mathbf{x}_6) = -1$$



$\mathbf{x}_7$

$$f(\mathbf{x}_7) = ?$$

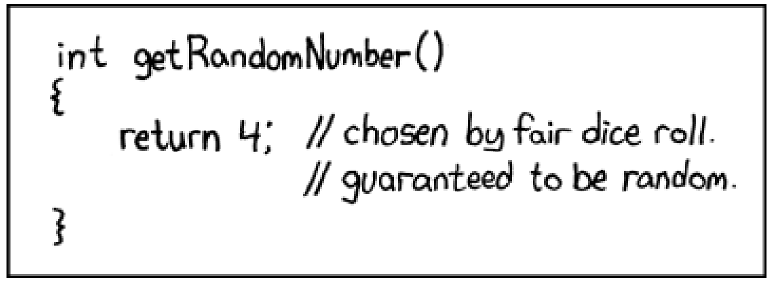
Learning seems *impossible* without additional assumptions!

POSSIBLE VS PROBABLE

Flip a biased coin, lands on head with *unknown* probability $p \in [0, 1]$

$\mathbb{P}\left(\text{head}\right) = p$ and $\mathbb{P}\left(\text{tail}\right) = 1 - p$

Say we flip the coin N times, can we estimate p?



https://xkcd.com/221/

$$\hat{p} = \frac{\#\ \text{head}}{N}$$

Can we relate $\hat{p}$ to p?

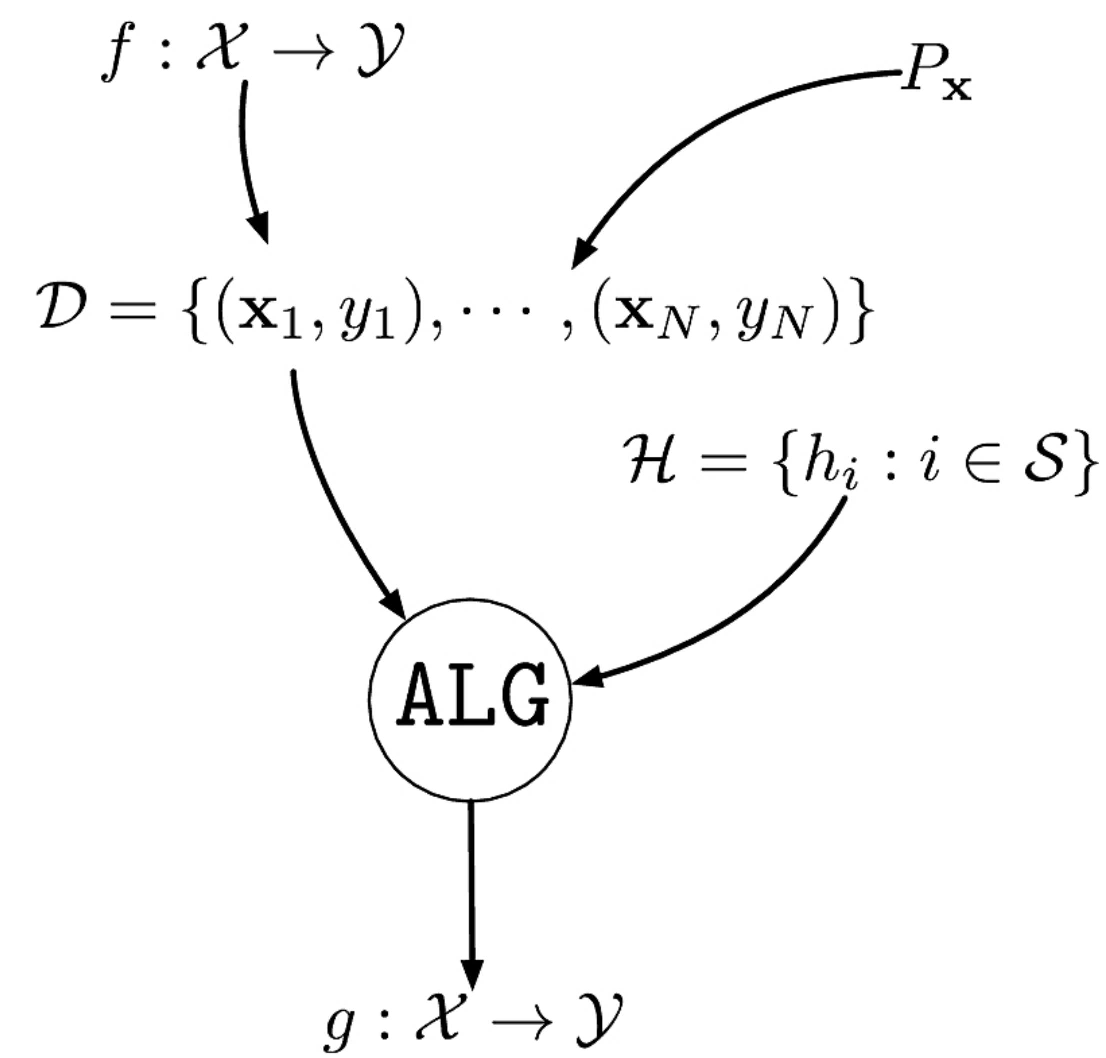
- The law of large numbers tells us that $\hat{p}$ converges in probability to p as N gets large

$$\forall \epsilon > 0 \quad \mathbb{P}\left(\left|\hat{p} - p\right| > \epsilon\right) \underset{N \to \infty}{\longrightarrow} 0.$$

It is *possible* that $\hat{p}$ is completely off but it is not *probable*

COMPONENTS OF SUPERVISED MACHINE LEARNING

1. An *unknown function* $f : \mathcal{X} \to \mathcal{Y} : \mathbf{x} \mapsto y = f(\mathbf{x})$ to learn
2. A *dataset* $\mathcal{D} \triangleq \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)\}$
   - $\{\mathbf{x}_i\}_{i=1}^N$ i.i.d. *from unknown distribution* $P_{\mathbf{x}}$ on $\mathcal{X}$
   - $\{y_i\}_{i=1}^N$ are the corresponding labels $y_i \in \mathcal{Y} \triangleq \mathbb{R}$
3. A *set of hypotheses* $\mathcal{H}$ as to what the function could be
4. An *algorithm* $\mathtt{ALG}$ to find the best $h \in \mathcal{H}$ that explains f



Learning model #2

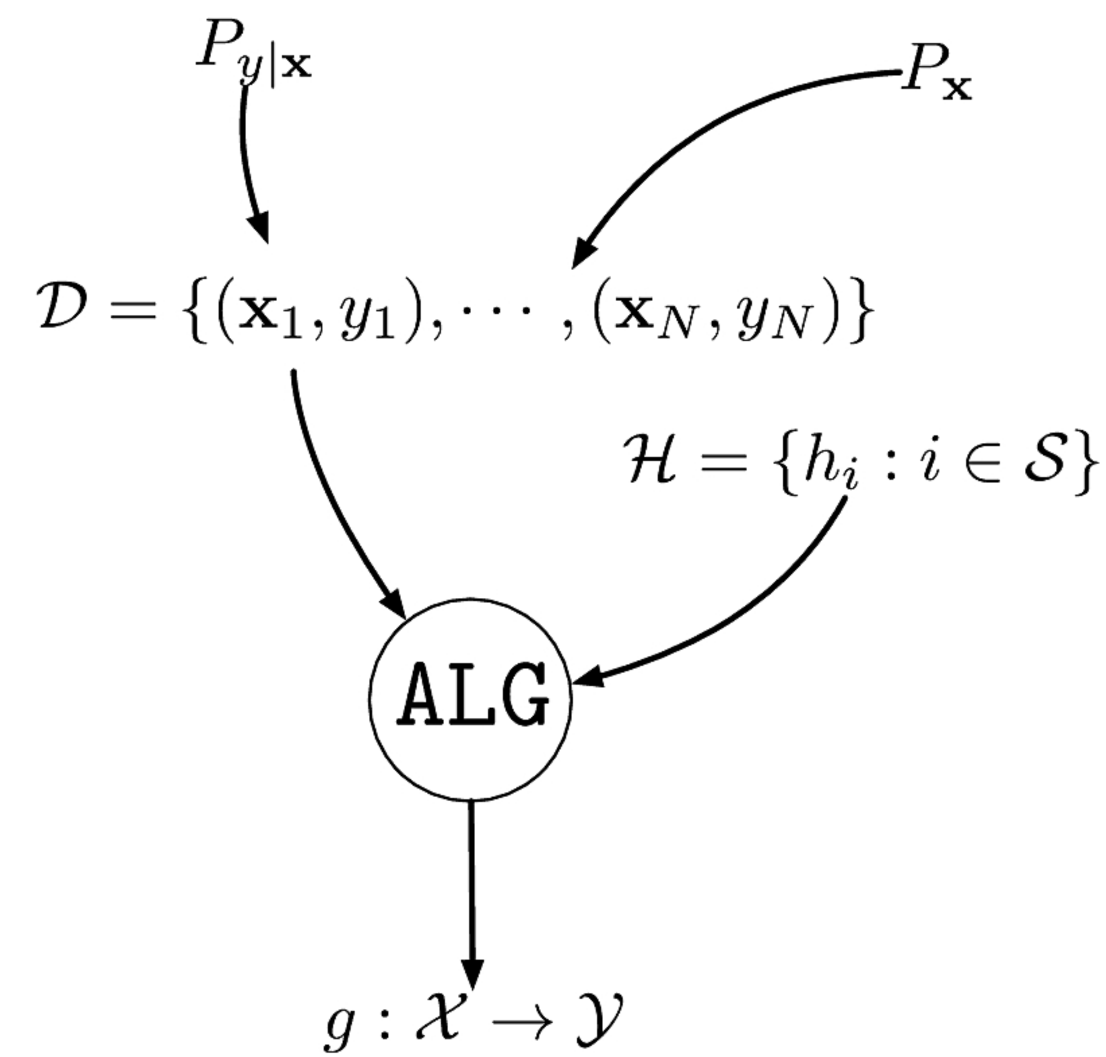
ANOTHER LEARNING PUZZLE

Which color is the dress?

COMPONENTS OF SUPERVISED MACHINE LEARNING

::: nonincremental

1. An *unknown conditional distribution* $P_{y|\mathbf{x}}$ to learn
   - $P_{y|\mathbf{x}}$ models $f : \mathcal{X} \to \mathcal{Y}$ *with noise*
2. A *dataset* $\mathcal{D} \triangleq \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)\}$
   - $\{\mathbf{x}_i\}_{i=1}^N$ i.i.d. from distribution $P_\mathbf{x}$ on $\mathcal{X}$
   - $\{y_i\}_{i=1}^N$ are the corresponding labels $y_i \sim P_{y|\mathbf{x}=\mathbf{x}_i}$
3. A *set of hypotheses* $\mathcal{H}$ as to what the function could be
4. An *algorithm* $\texttt{ALG}$ to find the best $h \in \mathcal{H}$ that explains f :::



Learning model #3

The roles of $P_{y|\mathbf{x}}$ and $P_\mathbf{x}$ are *different*

- $P_{y|\mathbf{x}}$ is what we want to learn, captures the underlying function and the noise added to it
- $P_\mathbf{x}$ models *sampling* of dataset, need *not* be learned

YET ANOTHER LEARNING PUZZLE

Assume that you are designing a fingerprint authentication system

- You trained your system with a fancy machine learning system
- The probability of wrongly authenticating is *1%*
- The probability of correctly authenticating is *60%*
- Is this a good system?

It depends!

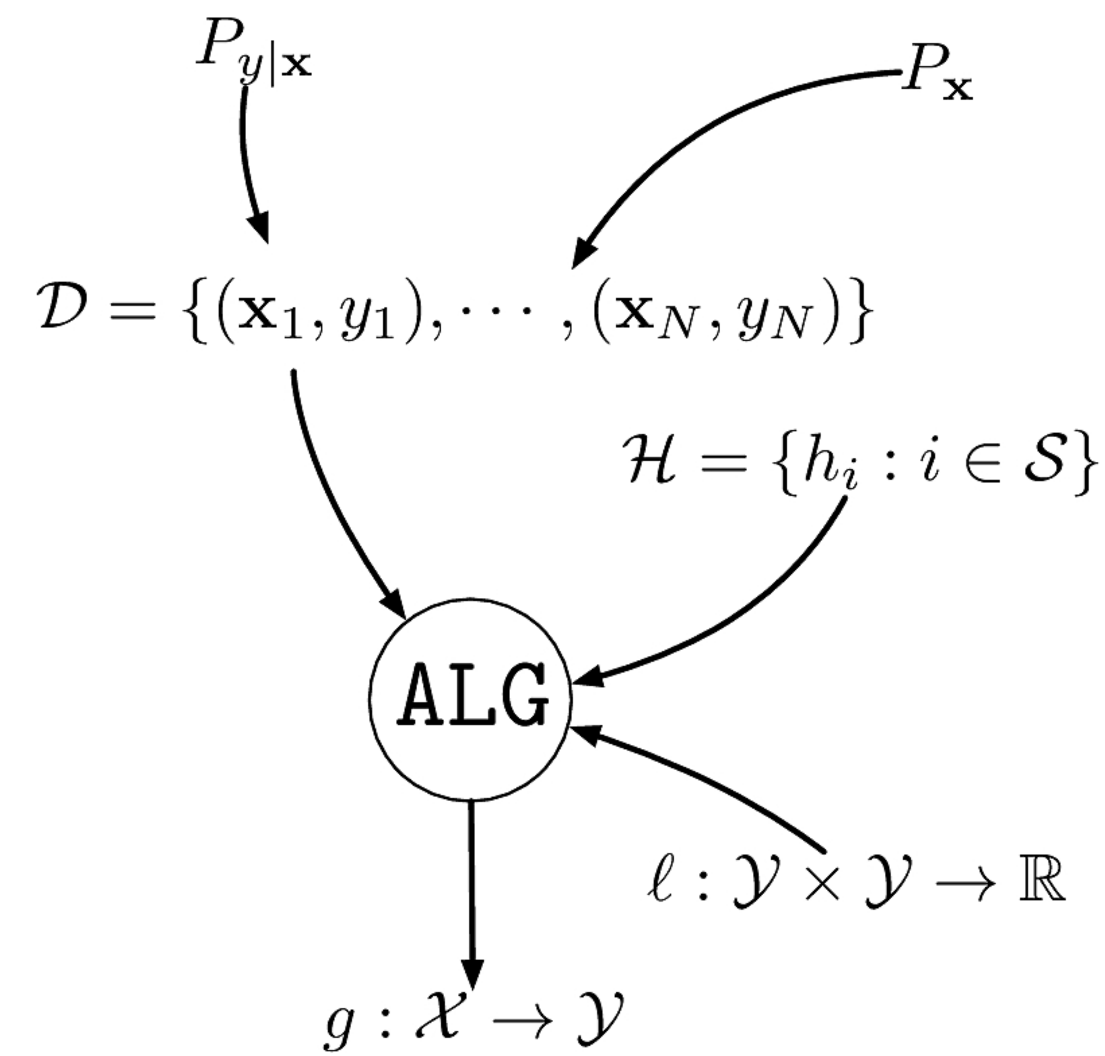
- If you are GTRI, this might be ok (security matters more)
- If you are Apple, this is not acceptable (convenience matters more)

There is an application dependent *cost* that can affect the design

Biometric authentication system

COMPONENTS OF SUPERVISED MACHINE LEARNING

1. A *dataset* $\mathcal{D} \triangleq \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)\}$
   - $\{\mathbf{x}_i\}_{i=1}^N$ i.i.d. from an unknown distribution $P_{\mathbf{x}}$ on $\mathcal{X}$
2. An *unknown conditional distribution* $P_{y|\mathbf{x}}$
   - $P_{y|\mathbf{x}}$ models $f : \mathcal{X} \to \mathcal{Y}$ *with noise*
   - $\{y_i\}_{i=1}^N$ are the corresponding labels $y_i \sim P_{y|\mathbf{x}=\mathbf{x}_i}$
3. A *set of hypotheses* $\mathcal{H}$ as to what the function could be
4. A *loss function* $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ capturing the “cost” of prediction
5. An *algorithm* $\mathtt{ALG}$ to find the best $h \in \mathcal{H}$ that explains f



Final supervised learning model