Last time

PCA
principal
components
analysis

data dependent
dimensionality
reduction

Today

- SVD + applications
- least squares
- maybe - perceptron alg.

Projects
- google form
- presenting at Microsoft?

Credit for figures:
Roughgarden & Valiant
Leskovec, Rajaraman, Ullman slides
Emily Fox (CS 446)
Kilian Weinberger (CS4780 Cornell)

PCA & SVD

have data set $x_1, \dots, x_m$ $\quad x_i \in \mathbb{R}^n$

$$X = \begin{pmatrix} - & x_1 & - \\ & \vdots & \\ - & x_m & - \end{pmatrix}$$

PCA: Fix k. find orthonormal vectors $\vec{v}_1, \dots, \vec{v}_k$

s.t. $x_i \approx \sum_{j=1}^{k} (x_i \cdot v_j)\, \vec{v}_j$

Find v_1 to min $\frac{1}{m} \sum_{i=1}^{m} \text{dist}(x_i, \text{line}(v_1))^2$

$\equiv \max \frac{1}{m} \sum_{i=1}^{m} (x_i \cdot v_1)^2$ variance.

v_1 principal eigenvector of matrix $X^T X$

$$\underbrace{X^T X}_{\text{pos semi-definite}} = \underbrace{Q\, D\, Q^T}_{\text{symmetric matrix}} \qquad QQ^T = Q^T Q = I$$

orthogonal.

$$Q = \begin{pmatrix} | & & | \\ q_1 & \cdots & q_n \\ | & & | \end{pmatrix}$$

$q_1 \cdots q_n$ eigenvectors

$\lambda_1 \cdots \lambda_n$

$$D = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix}$$

pos semi-definite.
all eigenvalues nonnegative.

$$X^T X q_1 = \lambda_1 q_1$$

q_1 best choice for v_1

singular value decompostn SVD

$\Rightarrow$ tells us best way to approximate our matrix with a "low rank" matrix

$$
\begin{array}{cc}
 & \text{movies} \\
\text{people} & \begin{bmatrix} 7 & ? & ? \\ ? & 8 & ? \\ ? & 12 & 6 \\ ? & ? & 2 \\ 21 & 6 & ? \end{bmatrix}
\end{array}
$$

can we reconstruct missing entries?

Suppose assume that this matrix is rank 1

every row is a multiple of every other row.

$$
\begin{bmatrix} 7 & 2 & 1 \\ & 8 & \\ & 12 & 6 \\ & \boxed{4} & 2 \\ 21 & 6 & \underline{3} \end{bmatrix} = \begin{pmatrix} 2\,(7 \;\; 2 \;\; 1) \\ 4 \cdot (7 \;\; 2 \;\; 1) \\ 6\,(7 \;\; 2 \;\; 1) \\ 2\,(7 \;\; 2 \;\; 1) \\ 3\,(7 \;\; 2 \;\; 1) \end{pmatrix} = v \cdot u^T
$$

u^T

outer product.

$$
\begin{pmatrix} 1 \\ 4 \\ 6 \\ 2 \\ 3 \end{pmatrix} (7 \;\; 2 \;\; 1)
$$

$$
\begin{pmatrix} 7v & 2v & v \end{pmatrix}
$$

Rank 0 matrix all 0.

Rank 1

$$\mathbf{A} = \mathbf{u}\mathbf{v}^\top = \begin{bmatrix} — & u_1\mathbf{v}^\top & — \\ — & u_2\mathbf{v}^\top & — \\ & \vdots & \\ — & u_m\mathbf{v}^\top & — \end{bmatrix} = \begin{bmatrix} | & | & & | \\ v_1\mathbf{u} & v_2\mathbf{u} & \cdots & v_n\mathbf{u} \\ | & | & & | \end{bmatrix}$$

Rank 2.

$$\mathbf{A} = \mathbf{u}\mathbf{v}^\top + \mathbf{w}\mathbf{z}^\top = \begin{bmatrix} — & u_1\mathbf{v}^\top + w_1\mathbf{z}^\top & — \\ — & u_2\mathbf{v}^\top + w_2\mathbf{z}^\top & — \\ & \vdots & \\ — & u_m\mathbf{v}^\top + w_m\mathbf{z}^\top & — \end{bmatrix} = \underbrace{\begin{bmatrix} | & | \\ \mathbf{u} & \mathbf{w} \\ | & | \end{bmatrix} \cdot \begin{bmatrix} — & \mathbf{v}^\top & — \\ — & \mathbf{z}^\top & — \end{bmatrix}}$$

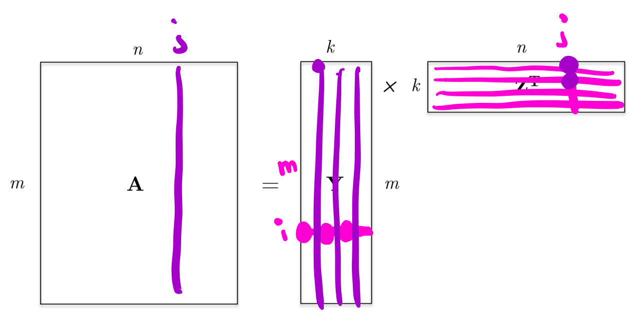


Figure 1: Any matrix $\mathbf{A}$ of rank k can be decomposed into a long and skinny matrix times a short and long one.

SVD of a matrix $m \times n$ matrix

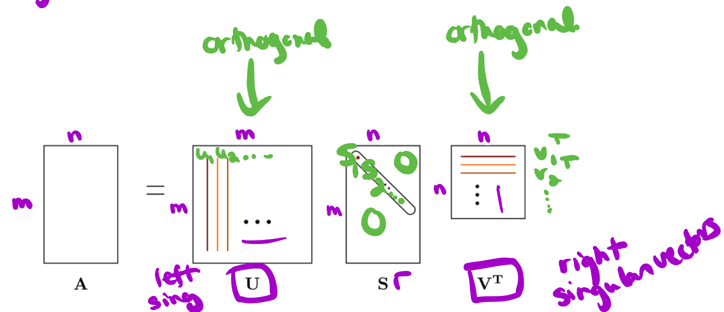


Figure 2: The singular value decomposition (SVD). Each singular value in **S** has an associated left singular vector in **U**, and right singular vector in **V**.

$$\boxed{A = \sum_{i=1}^{\min(m,n)} s_i u_i v_i^T}$$

diag entries of S are called singular values $s_1 \geq s_2 \geq \dots \geq 0$

can be computed in $\min(O(m^2 n), O(n^2 m))$ time.

Suppose we want the "best" rank k approx to A

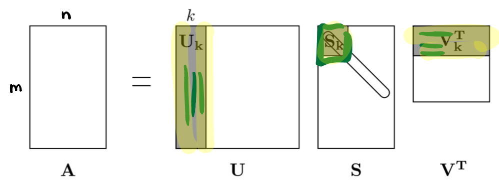


Figure 3: Low rank approximation via SVD. Recall that $\mathbf{S}$ is non-zero only on its diagonal, and the diagonal entries of S are sorted from high to low. Our low rank approximation is $\mathbf{A}_k = \mathbf{U}_k \mathbf{S}_k \mathbf{V}_k^\top$.

$$A_k = \begin{array}{c} \\ m \end{array} \overset{k}{\left[U_k \right]} \; k\overset{k}{\left[S_k \right]} \; k\overset{n}{\left[V_k^T \right]}$$

Thm

This low-rank approx is optimal in the sense that $\forall$ matrix A $(m \times n)$ and rank target $k \geq 1$ and any other rank k matrix B $(m \times n)$

$$\|A - A_k\|_F^2 \leq \|A - B\|_F^2$$

$$\|M\|_F^2 = \sum_i \sum_j m_{ij}^2$$

$$\sum_{i,j} (a_{ij} - a^k_{ij})^2 \leq \sum_{i,j} (a_{ij} - b_{ij})^2$$

Frobenius

Relationship between PCA & SVD.

$(AB)^T = B^T A^T$

$$X^T X = Q D Q^T \qquad X = U S V^T$$

$$X^T X = (USV^T)^T USV^T$$
$$= V^{TT} S^T \underbrace{U^T U}_{I} S V^T$$

$q_1 = v_1$

$\lambda_1 = s_1^2$

$$X^T X = V S^2 V^T$$

$$\boxed{X X^T = U S^2 U^T}$$

$X X^T_{ij}$

$X =$ customers, products.

$$\underset{i^{th}\text{ row}}{x_i} = \sum_{j=1}^{k} \underbrace{(x_i, v_j)}\, \vec{v_j}$$

Application 1: Denoising.

Suppose A is a rank k matrix.

$$C = A + N$$

noise matrix

each entry of N is indep $N(0, \sigma^2)$

Then claim is if variance of noise sufficiently small.

then $\|C_k - A\|_F^2$ small w.h.p.

depend on variance

$$A = \sum_{i=1}^{k} s_i u_i v_i^T$$

in C ; $s_j \ll s_1 \dots s_k$

$j > k$

Thm: $\hat{A}$ $m \times n$ matrix of indep r.v.'s whose variances are bounded by σ^2

If $A = E(\hat{A})$ is rank k

then w.h.p.

$$\|A - \hat{A}_k\|_F^2 = O(k\sigma^2(m+n))$$

$\hat{A} = \begin{pmatrix} & & \\ & X_{ij} & \\ & & \end{pmatrix}$

$E(\hat{A}) = \begin{pmatrix} & & \\ & E(X_{ij}) & \\ & & \end{pmatrix}$

$X_{ij} = \begin{matrix} 1 & \frac{2}{3} \\ 0 & \frac{1}{3} \end{matrix}$

avg/entry

$$\frac{k\sigma^2(m+n)}{m \cdot n} = o(1)$$

$$\hat{A} = \underbrace{E(\hat{A})}_{A} + \underbrace{(\hat{A} - E(\hat{A}))}$$

A: rank k

$\hat{A} - E(\hat{A})$: deviations from exp. mean 0 noise

Collaborative Filtering
recommendations

$p_{ij} = p$

$$\begin{array}{cc} & \text{movies} \\ \text{people} & \begin{bmatrix} 3 & 2 & 1 & 3 \\ 2 & 4 & 4 & 5 \\ 1 & 3 & 1 & 4 \end{bmatrix} \end{array} \longrightarrow \begin{bmatrix} 3 & 2 & ? & 3 \\ 2 & ? & 4 & ? \\ ? & 3 & ? & 4 \end{bmatrix}$$

R — ground truth. $\qquad R^?$

assumption: R is rank k

human victory

Honey I Shrunk movie

R n movies

m people

Amy   .8   .2   .3

humor
violence
romance
indie
⋮

R is rank k

Assume that there is $P = \begin{bmatrix} p_{ij} \end{bmatrix}$

s.t. prob that there is a rating available for that entry is p_{ij}

Let's assume we know P.

$$\begin{array}{cc} & \text{movies} \\ \text{people} & \begin{bmatrix} 3 & 2 & 1 & 3 \\ 2 & 4 & 4 & 5 \\ 1 & 3 & 1 & 4 \end{bmatrix} \end{array} \longrightarrow \begin{bmatrix} 3 & 2 & ? & 3 \\ 2 & ? & 4 & ? \\ ? & 3 & ? & 4 \end{bmatrix}$$

R ↑ ground truth. $\tilde{R}$

$E(\text{matrix}) = P$

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

assumption : R is rank k

Define

$$\hat{R} = \begin{cases} \dfrac{\tilde{R}_{ij}}{p_{ij}} & \text{if entry } (i,j) \text{ is present} \\ 0 & \text{o.w.} \end{cases}$$

Furedi, Komlōs

Thm: $\hat{A}$ $m \times n$ matrix of indep r.v.'s whose variances are bounded by σ^2

If $A = E(\hat{A})$ is rank k

then w.h.p.

$$\|A - \hat{A}_k\|_F^2 = O(k\sigma^2(m+n))$$

$\hat{A} = (\quad x_{ij} \quad)$

$E(\hat{A}) = (\quad E(x_{ij}) \quad)$

$x_{ij} = \begin{matrix} 1 & \frac{2}{3} \\ 0 & \frac{1}{3} \end{matrix}$

$$\hat{A} = A + \underbrace{\hat{A} - A}_{\text{mean } 0}$$ variance.

$$E(\hat{R}_{ij}) = R_{ij}$$
$$= p_{ij}\frac{R_{ij}}{p_{ij}} + (1 - p_{ij})0$$
$$= R_{ij}$$

$\Longrightarrow$ $\hat{R}_k$ is very close to R.

Assume P itself is low rank

given R?
construct
matrix $\hat{P}$

whose (i,j) entry is 1
see rating there and 0
o.w.

$$\text{people}\begin{pmatrix} & \\ & \\ & \end{pmatrix}\overset{\text{movies}}{\begin{pmatrix} & & & \end{pmatrix}}$$

$$E(\hat{P}) = P$$

$\Rightarrow$ $\hat{P}_k$ very close to P

Linear regression.

How much is my house worth?

Data — input, output

$(x_1 = \text{sq.ft.}, y_1 = \$)$

$(x_2 = \text{sq.ft.}, y_2 = \$)$

$(x_3 = \text{sq.ft.}, y_3 = \$)$

$(x_4 = \text{sq.ft.}, y_4 = \$)$

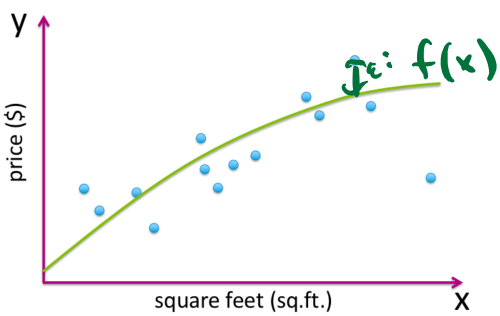
$(x_5 = \text{sq.ft.}, y_5 = \$)$

⋮

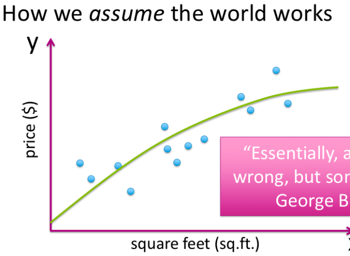
Input vs. Output:
- y is the quantity of interest
- assume y can be predicted from x

$f(\text{squarefootage}) \rightarrow \text{price}$

Model –
How we *assume* the world works

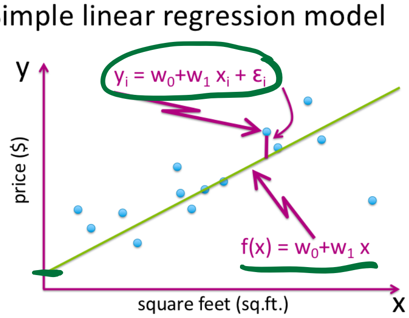


Model –
How we *assume* the world works



"Essentially, all models are wrong, but some are useful." George Box, 1987.

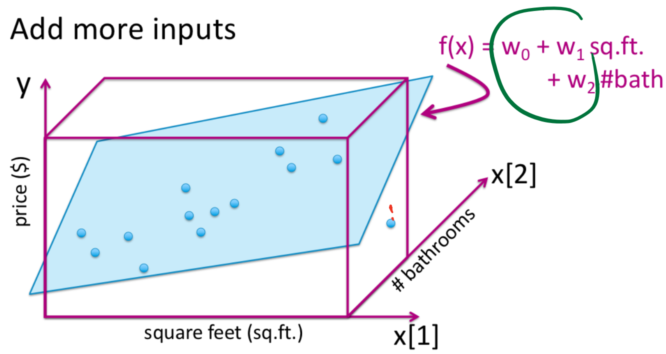
Simple linear regression model



(x_i, y_i) pairs.

Find best choice for w_0, w_1

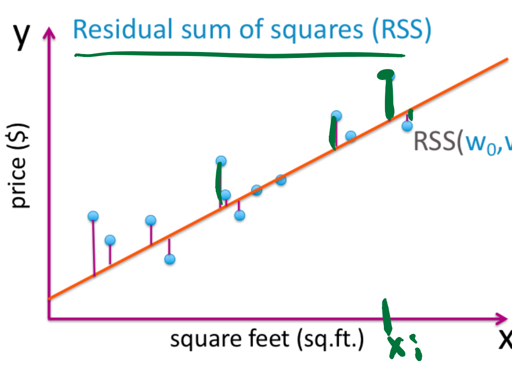
Add more inputs



Many possible inputs

- Square feet
- # bathrooms
- # bedrooms
- Lot size
- Year built
- …

“Cost” of using a given line



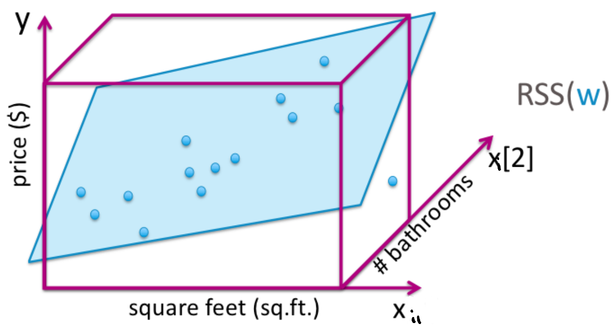
w_0, w_1

Given (x_i, y_i) $1 \le i \le N$

$$\mathrm{RSS}(w_0,w_1) = \sum_{i=1}^{N}(y_i-[w_0+w_1x_i])^2$$

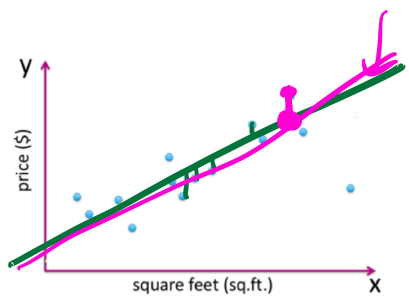
Problem: find w_0 & w_1 to minimize

RSS for multiple regression

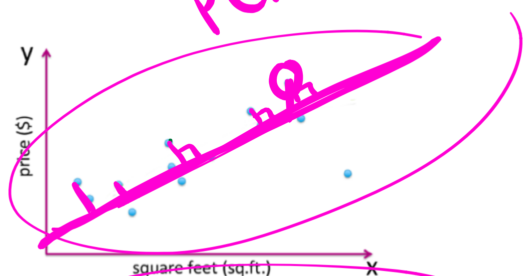


Linear regression

(x_i, y_i) pairs.



PCA



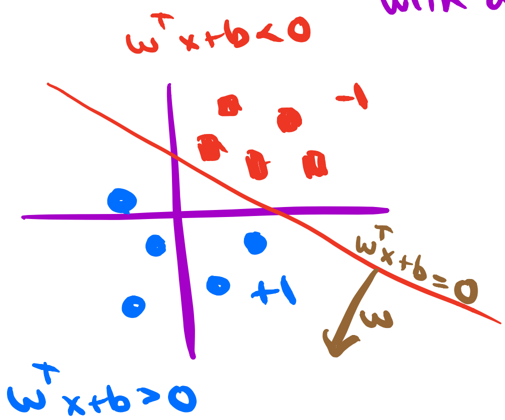
$$\begin{array}{cc} & k \\ m & \begin{pmatrix} \\ \\ \\ \end{pmatrix} \end{array} \begin{array}{cc} & n \\ k & \begin{pmatrix} & & & \end{pmatrix} \end{array}$$

$k(m+n)$

$m \cdot n$

Supervised learning & Perceptron Algorithm

labelled data: use that to come up with a way to give answers on data haven't seen.



$(\vec{x}_i, y_i)$

binary classification. label for each pt $\{-1, 1\}$

Assumption: data set **is** linearly separable.

$\exists\ \vec{w}$ & b s.t.

$y_i = 1 \Rightarrow w^T x + b > 0$

$y_i = -1 \Rightarrow w^T x + b < 0$

Objective: find one

Simplify life by getting rid of additive const b.

$$\vec{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \rightarrow \vec{x}' = \begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_n \end{pmatrix} \qquad \begin{pmatrix} w \end{pmatrix} \rightarrow w' = \begin{pmatrix} b \\ | \\ w \\ | \end{pmatrix}$$

$w \cdot x + b \gtreqless 0 \qquad w' \cdot x' \gtreqless 0$

$y = 1 \qquad w^T x > 0$

$y = -1 \qquad w^T x < 0$

make a mistake if $\boxed{y_i w^T x_i \leq 0}$

m = # mistakes we've made so far

Initialize $\vec{w} = \vec{0}$ // Initialize $\vec{w}$. $\vec{w} = \vec{0}$ misclassifies everything.
while TRUE **do** // Keep looping
  $m = 0$ // Count the number of misclassifications, m
  for $(x_i, y_i) \in D$ **do** // Loop over each (data, label) pair in the dataset, D
    if $y_i(\vec{w}^T \cdot \vec{x_i}) \leq 0$ **then** made mistake // If the pair $(\vec{x_i}, y_i)$ is misclassified
      $\vec{w} \leftarrow \vec{w} + y_i\vec{x_i}$ // Update the weight vector $\vec{w}$
      $m \leftarrow m + 1$ // Counter the number of misclassification
    end if
  end for
  if $m = 0$ **then** // If the most recent $\vec{w}$ gave 0 misclassifications
    break // Break out of the while-loop
  end if
end while // Otherwise, keep looping!

$w + (-1)x$
$w - x$

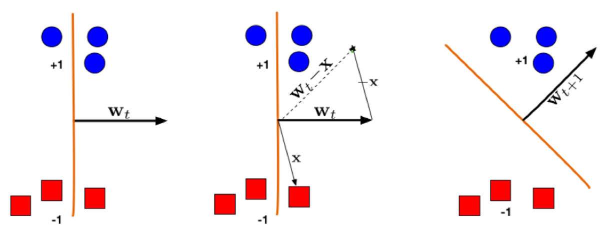


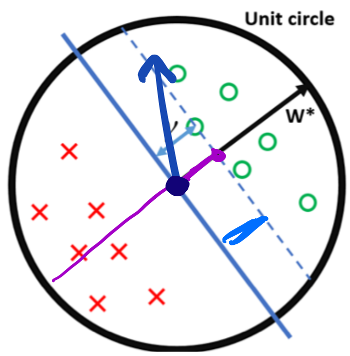
Illustration of a Perceptron update. (Left:) The hyperplane defined by $\mathbf{w}_t$ misclassifies one red (-1) and one blue (+1) point. (Middle:) The red point $\mathbf{x}$ is chosen and used for an update. Because its label is -1 we need to ***subtract*** *$\mathbf{x}$ from $\mathbf{w}_t$. (Right:) The udpated hyperplane $\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{x}$ separates the two classes and the Perceptron algorithm has converged.*

w^* to a correct answer

$\exists\, w^*$ $\underline{y_i\,(\text{sign}(x_i^T w^*)) > 0}$

wlog $\|w^*\| = 1$

to simplify $x_i' = \dfrac{x_i}{\max_j \|x_j\|}$ $\|x_i'\| \leq 1$



γ is called the margin

$$= \min_{(x_i,y_i)} |x_i^T w^*|$$

Thm: $m \le \frac{1}{\gamma^2}$

Look at 2 quantities.

$w^T w^*$ ① increasing

$w^T w$ ② can't be going up fast

When we make a mistake:

$y(w^T x) \le 0$

$y((w^*)^T x) > 0$

①

$$(w + yx)^T w^* = w^T w^* + \underbrace{y \cdot x^T w^*}_{\ge \gamma}$$

every mistake $\Longrightarrow$ increase $w^T w^*$ by at least γ

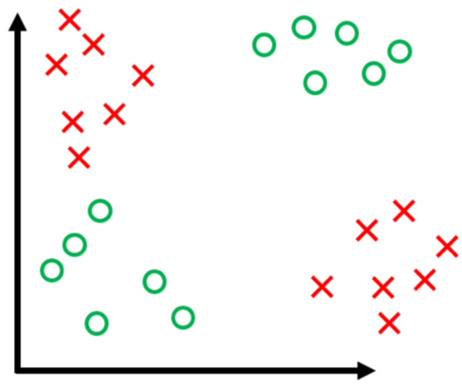
②

$$(w+yx)^T(w+yx) = w^T w + \underbrace{2y x^T w}_{\le 0} + y^2 \underbrace{x^T x}_{\le 1}$$

every mistake, increase $w^T w$ by at most 1

$$0 < m\gamma \le w^T w^*$$
$$= w^T w^*$$
$$\le \|w\| \|w^*\| \cos\theta$$
$$\le \|w\| \|w^*\|$$
$$= \|w\|$$
$$= \sqrt{w^T \cdot w}$$
$$\le \sqrt{m}$$

$$m\gamma \le \sqrt{m}$$
$$m^2\gamma^2 \le m$$
$$m \le \frac{1}{\gamma^2}$$



Bad example for perceptron alg