SGD Algorithm:

$\vec{w}^{(0)} \leftarrow \vec{0}$;
for $t \leftarrow 0$ to $T-1$ do {
  pick $i$ uniformly at random from $\{1, \dots, m\}$;
  $\vec{w}^{(t+1)} \leftarrow \vec{w}^{(t)} - \eta \nabla \ell(\vec{w}^{(t)}, (\vec{x}_i, \vec{y}_i))$;
}
return $\bar{w} = \frac{1}{T} \sum_{t=1}^{T} \vec{w}^{(t)}$;

To simplify the notation, we are going to compute $\nabla \ell(\vec{w}^{(t)}, (\vec{x}_i, \vec{y}_i))$:

$$\nabla \ell(\vec{w}^{(t)}, (\vec{x}_i, \vec{y}_i)) = \begin{cases} 0 & \text{if } y_i \langle \vec{w}, \vec{x}_i \rangle > 0 \\ \nabla(-y_i \langle \vec{w}, \vec{x}_i \rangle) & \text{otherwise} \end{cases}$$

→ $y_i$ correctly classifies $x_i$

Assume that $y_i \langle \vec{w}, \vec{x}_i \rangle < 0$:

$$\nabla(-y_i \langle \vec{w}, \vec{x}_i \rangle) = \left[ \frac{\partial(-y_i \langle \vec{w}, \vec{x}_i \rangle)}{\partial w_1}, \dots, \frac{\partial(-y_i \langle \vec{w}, \vec{x}_i \rangle)}{\partial w_d} \right]^T$$

let $\vec{x}_i = \begin{bmatrix} x_{i1} \\ \vdots \\ x_{id} \end{bmatrix}$. Since $-y_i \langle \vec{w}, \vec{x}_i \rangle = -y_i \sum_{j=1}^{d} (w_j x_{ij})$

$$\frac{\partial(-y_i \langle \vec{w}, \vec{x}_i \rangle)}{\partial w_j} = -y_i x_{ij}$$

$$\nabla \ell(w, (\vec{x}_i, y_i)) = [-y_i x_{i1}, \ldots, -y_i x_{id}]^T$$
$$= -y_i \vec{x}_i$$

Then we can rewrite the pseudo code:

SGD Algorithm:

$\vec{w}^{(0)} \leftarrow \vec{0}$;
for $t \leftarrow 0$ to $T-1$ do {
    pick $i$ uniformly at random from $\{1, \ldots, m\}$;
    if $y_i \langle \vec{w}^{(t)}, \vec{x}_i \rangle < 0$ then {
        $\vec{w}^{(t+1)} \leftarrow \vec{w}^{(t)} + \eta y_i \vec{x}_i$;
    }
}
return $\bar{w} = \frac{1}{T} \sum_{t=1}^{T} \vec{w}^{(t)}$;

Comparison:

| | perceptron | SGD perceptron |
|---|---|---|
| 1) | Choose a missclassified point | choose a point at random |
| 2) | $\eta = 1$ | $\eta$ is a parameter |
| 3) | return "best" $\vec{w}^{(t)}$ | return $\bar{w}$ |

→ the main difference

↳ just a generalization

↳ implementation choice (average / best / ...)

We can speed up the SGD perceptron, at each iteration, by picking a missclassified point at random

$\Downarrow$

SGD perceptron is the perceptron

# Linear Regression

$\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \mathbb{R}$

Regression with linear models (wow!)

Hypothesis class:

Linear

$$\mathcal{H}_{reg} = L_d = \{\mathbf{x} \to \langle \mathbf{w}, \mathbf{x} \rangle + b : \mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}\}$$

This is already a real value

**Note:** $h \in \mathcal{H}_{reg} : \mathbb{R}^d \to \mathbb{R}$

Commonly used loss function: *squared-loss*

$$\ell(\underbrace{h}_{\in \mathcal{H}_{reg}}, (\underbrace{\mathbf{x}}_{\in \mathcal{X}}, \overbrace{y}^{\in \mathcal{Y}})) \stackrel{def}{=} (h(\mathbf{x}) - y)^2$$

$\Rightarrow$ empirical risk function (training error): *Mean Squared Error*

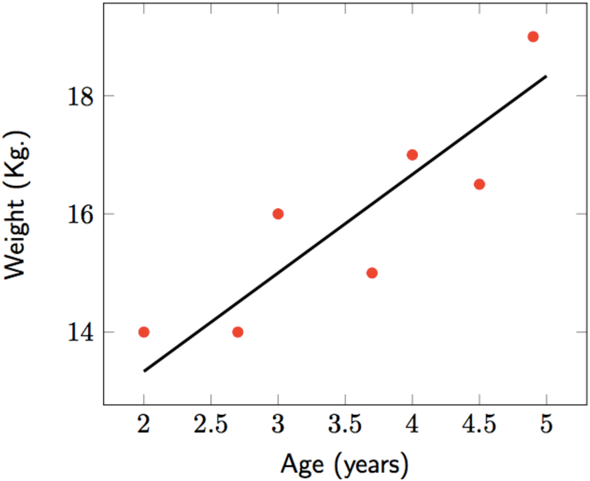
$S = \{(\vec{x}_1, y_1), .., (\vec{x}_m, y_m)\}$

$$L_S(h) = \frac{1}{m} \sum_{i=1}^{m} (h(\mathbf{x}_i) - y_i)^2$$

# Linear Regression - Example

$d = 1$

$h_{\vec{w},b}(\vec{x}) = b + w_1 x_1$ → find the best line



# Least Squares

How to find a ERM hypothesis? *Least Squares* algorithm

Best hypothesis:

$$\arg\min_{\mathbf{w}} L_S(h_{\mathbf{w}}) = \arg\min_{\mathbf{w}} \frac{1}{m}\sum_{i=1}^{m}(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i)^2$$

Equivalent formulation: **w** minimizing *Residual Sum of Squares* (RSS), i.e.

$$\arg\min_{\mathbf{w}} \sum_{i=1}^{m}(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i)^2$$

$w = [b, w_1, \ldots, w_d]^T$

# RSS: Matrix Form

Let

$$\mathbf{X} = \begin{bmatrix} \cdots & \mathbf{x}_1 & \cdots \\ \cdots & \mathbf{x}_2 & \cdots \\ \cdots & \vdots & \cdots \\ \cdots & \mathbf{x}_m & \cdots \end{bmatrix}$$

→ each row is an <u>instance</u> in the training set

**X**: *design matrix*

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

→ observations from the training set

$\Rightarrow$ we have that RSS is

$$\sum_{i=1}^{m} (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2 = (\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w})$$

Want to find $\mathbf{w}$ that minimizes RSS (=*objective function*):

$$\arg\min_{\mathbf{w}} RSS(\mathbf{w}) = \arg\min_{\mathbf{w}} (\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w})$$

↳ parabola $(X^2w^2 - 2yXw + y^2)$


How?

Compute gradient $\frac{\partial RSS(\mathbf{w})}{\partial \mathbf{w}}$ of objective function w.r.t $\mathbf{w}$ and compare it to 0.

$$\frac{\partial RSS(\mathbf{w})}{\partial \mathbf{w}} = -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\mathbf{w})$$

Then we need to find $\mathbf{w}$ such that

$$-2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\mathbf{w}) = 0$$

$$-2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\mathbf{w}) = 0$$

is equivalent to

$$\mathbf{X}^T\mathbf{X}\mathbf{w} = \mathbf{X}^T\mathbf{y}$$

If $\mathbf{X}^T\mathbf{X}$ is invertible $\Rightarrow$ solution to ERM problem is:

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

# Complexity Considerations

We need to compute

$$(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Algorithm:

1. compute $\mathbf{X}^T\mathbf{X}$: product of $(d+1) \times m$ matrix and $m \times (d+1)$ matrix
2. compute $(\mathbf{X}^T\mathbf{X})^{-1}$ inversion of $(d+1) \times (d+1)$ matrix
3. compute $(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$: product of $(d+1) \times (d+1)$ matrix and $(d+1) \times m$ matrix
4. compute $(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$: product of $(d+1) \times m$ matrix and $m \times 1$ matrix

Most expensive operation? Inversion! → Good since d is small compared to m

$\Rightarrow$ done for $(d+1) \times (d+1)$ matrix

# $\mathbf{X}^T\mathbf{X}$ not invertible?

How do we get $\mathbf{w}$ such that

$$\mathbf{X}^T\mathbf{X}\mathbf{w} = \mathbf{X}^T\mathbf{y}$$

if $\mathbf{X}^T\mathbf{X}$ is not invertible?
Let

$$\mathbf{A} = \mathbf{X}^T\mathbf{X}$$

Let $\mathbf{A}^+$ be the *generalized inverse* of $\mathbf{A}$, i.e.:

$$\mathbf{A}\mathbf{A}^+\mathbf{A} = \mathbf{A}$$

**Proposition**

If $\mathbf{A} = \mathbf{X}^T\mathbf{X}$ is not invertible, then $\hat{w} = \mathbf{A}^+\mathbf{X}^T\mathbf{y}$ is a solution to $\mathbf{X}^T\mathbf{X}\mathbf{w} = \mathbf{X}^T\mathbf{y}$.

# Computing the Generalized Inverse of $\mathbf{A}$

Note $\mathbf{A} = \mathbf{X}^T\mathbf{X}$ is symmetric $\Rightarrow$ eigenvalue decomposition of $\mathbf{A}$:

$$\mathbf{A} = \mathbf{V}\mathbf{D}\mathbf{V}^T$$

$$D = \begin{bmatrix} D_{11} & & 0 \\ & \ddots & \\ 0 & & \ddots \end{bmatrix}$$

with

- $\mathbf{D}$: diagonal matrix (entries = eigenvalues of $\mathbf{A}$)
- $\mathbf{V}$: orthonormal matrix ($\mathbf{V}^T\mathbf{V} = \mathbf{I}_{d\times d}$)

$A\vec{x} = \lambda\vec{x}$

eigenvector

eigenvalue

Define $\mathbf{D}^+$ diagonal matrix such that:

$$\mathbf{D}^+_{i,i} = \begin{cases} 0 & \text{if } \mathbf{D}_{i,i} = 0 \\ \frac{1}{\mathbf{D}_{i,i}} & \text{otherwise} \end{cases} = \begin{bmatrix} \frac{1}{D_{1,1}} & & 0 \\ & \ddots & \\ 0 & & \ddots \end{bmatrix}$$

$$DD^+ = \begin{bmatrix} 1 & & 0 \\ & 1 & \\ 0 & & \ddots \end{bmatrix}$$

Let $\mathbf{A}^+ = \mathbf{V}\mathbf{D}^+\mathbf{V}^T$

Then

$$\begin{aligned}
\mathbf{A}\mathbf{A}^+\mathbf{A} &= \mathbf{V}\mathbf{D}\overbrace{\mathbf{V}^T\mathbf{V}}^{I}\mathbf{D}^+\overbrace{\mathbf{V}^T\mathbf{V}}^{I}\mathbf{D}\mathbf{V}^T \\
&= \mathbf{V}\overbrace{\mathbf{D}\mathbf{D}^+\mathbf{D}}^{D}\mathbf{V}^T \\
&= \mathbf{V}\mathbf{D}\mathbf{V}^T \\
&= \mathbf{A}
\end{aligned}$$

$\Rightarrow$ $\mathbf{A}^+$ is a generalized inverse of $\mathbf{A}$.

**In practice**: the Moore-Penrose generalized inverse $\mathbf{A}^\dagger$ of $\mathbf{A}$ is used, since it can be efficiently computed from the Singular Value Decomposition of $\mathbf{A}$.

## Exercise

Consider a linear regression problem, where $\mathcal{X} = \mathbb{R}^d$ and $\mathcal{Y} = \mathbb{R}$, with mean squared loss. The hypothesis set is the set of *constant* functions, that is $\mathcal{H} = \{h_a : a \in \mathbb{R}\}$, where $h_a(\mathbf{x}) = a$. Let $S = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m))$ denote the training set.

- Derive the hypothesis $h \in \mathcal{H}$ that minimizes the training error.
- Use the result above to explain why, for a given hypothesis $\hat{h}$ from the set of all linear models, the coefficient of determination $R^2 = 1 - \frac{\sum_{i=1}^m (\hat{h}(\mathbf{x}_i) - y_i)^2}{\sum_{i=1}^m (y_i - \bar{y})^2}$ where $\bar{y}$ is the average of the $y_i, i = 1, \ldots, m$ is a measure of how well $\hat{h}$ performs (on the training set).

Cerco di altri es su stesso.