Machine Learning

Linear Models

Fabio Vandin

October 28^{th}, 2022

Linear Regression $\neq$ Regression

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

Linear Regression

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

Hypothesis class:

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

Linear Regression

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

Hypothesis class:

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

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

Commonly used loss function: *squared-loss*

$$\ell(h, (\mathbf{x}, y)) \stackrel{def}{=} (h(\mathbf{x}) - y)^2$$

ERM for regression with linear models and squared loss

Linear Regression

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

Hypothesis class:

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

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

Commonly used loss function: *squared-loss*

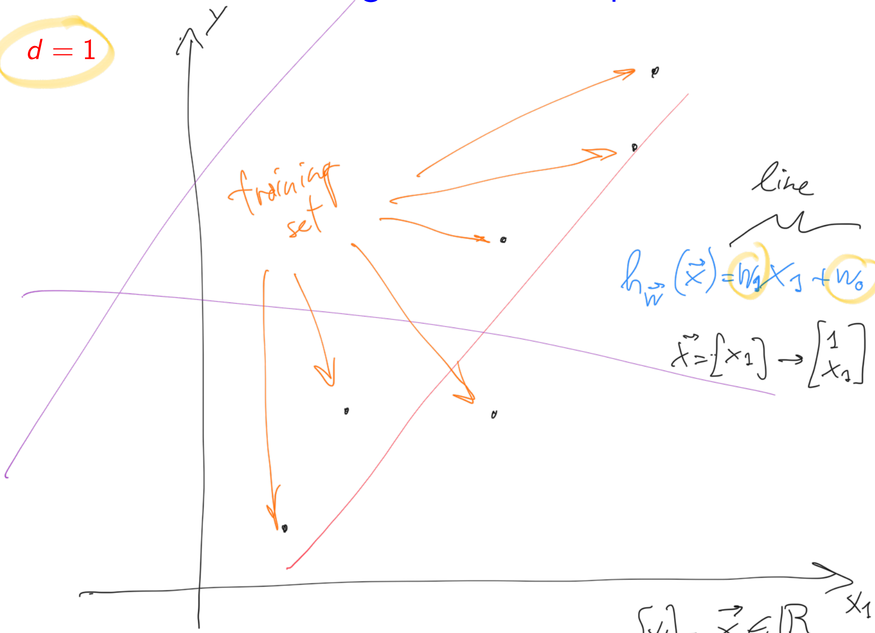
$$\ell(h, (\mathbf{x}, y)) \stackrel{def}{=} (h(\mathbf{x}) - y)^2$$

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

$$S = \{(\vec{x}_1, y_1), \ldots, (\vec{x}_m, y_m)\} \quad L_S(h) = \frac{1}{m} \sum_{i=1}^{m} \underbrace{(h(\mathbf{x}_i) - y_i)^2}_{\ell(h, (\vec{x}_i, y_i))}$$

Linear Regression - Example

$d = 1$



line

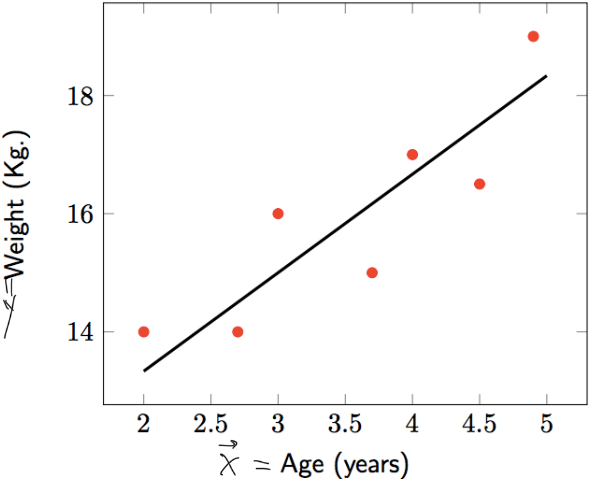
$$h_{\vec{w}}(\vec{x}) = w_1 x_1 + w_0$$

$$\vec{x} = [x_1] \to \begin{bmatrix} 1 \\ x_1 \end{bmatrix}$$

$$[x_1] = \vec{x} \in \mathbb{R}$$

Linear Regression - Example

$d = 1$



Least Squares

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

here : $\vec{x}_i = [1, x_{i1}, \dots, x_{id}]^T$
$\vec{w} = [w_0, w_1, \dots, w_d]^T$

Best hypothesis:

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

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: $\mathbf{w}$ minimizing *Residual Sum of Squares* (RSS), i.e.

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

RSS: Matrix Form

Let

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

$S = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_m, y_m)\}$

X: *design matrix*

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

RSS: Matrix Form

Let

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

$\mathbf{X}$: *design matrix*

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

$\Rightarrow$ we have that RSS is

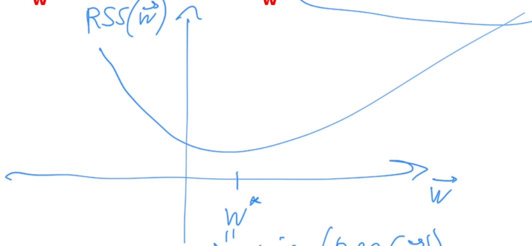
$$\sum_{i=1}^{m} (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2 = (\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w})$$

HW: check that the above is true

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

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

$\vec{w} \in \mathbb{R}$



$$w^* = \arg\min_{\vec{w}} \left(RSS(\vec{w})\right)$$

convex function

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

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

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

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

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

$A\vec{w} = b$

$-2X^T\vec{y} + 2X^TX\vec{w} = 0$

$\cancel{2}X^TX\vec{w} = \cancel{2}X^T\vec{y}$

$X^TX\vec{w} = X^T\vec{y}$

$A^{-1}A\vec{w} = A^{-1}b$

$\vec{w} = A^{-1}b$

$\vec{w} = \dots$

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

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

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

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

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

$\vec{x} = [1\ x_1 \ldots x_d]$

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

Algorithm:

1. compute $\mathbf{X}^T\mathbf{X}$: product of $(d+1) \times m$ matrix and $m \times (d+1)$ matrix

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!

$\Rightarrow$ 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.:

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

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

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

$$D = \begin{bmatrix} \square & & & 0 \\ & \square & & \\ & & \square & \\ 0 & & & \ddots \; \square \end{bmatrix}$$

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

$$I_{d\times d} = \underbrace{\begin{bmatrix} 1 & & & 0 \\ & 1 & & \\ & & \ddots & \\ 0 & & & 1 \end{bmatrix}}_{d} \Bigg\} d$$

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

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

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

with

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

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

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

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

$$D^+ = \begin{bmatrix} 1/D_{1,1} & & & 0 \\ & 1/D_{2,2} & & \\ & & 0 & \\ 0 & & & \ddots \end{bmatrix}$$

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

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

Is it a generalized inverse for A?

Show $AA^+A = A$

$$AA^+A = \overbrace{VDV^T}^{A}\,\overbrace{VD^+V^T}^{A^+}\,\overbrace{VDV^T}^{A}$$

since V is orthonormal: $V^TV = I$

$$= V\underbrace{DD^+D}_{D}V^T$$

$$= VDV^T = A$$

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

Then

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

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

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

Then

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

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

Your friend has developed a new machine learning algorithm for binary classification (i.e., $y \in \{-1, 1\}$) with 0-1 loss and tells you that it achieves a generalization error of only 0.05. However, when you look at the learning problem he is working on, you find out that $\Pr_{\mathcal{D}}[y = 1] = 0.95$...

- Assume that $\Pr_{\mathcal{D}}[y = \ell] = p_\ell$. Derive the generalization error of the (dumb) hypothesis/model that *always* predicts ℓ.
- Use the result above to decide if your friend's algorithm has learned something or not.

Exercise

Assume we have the following training set S, where $\mathbf{x} = [x_1, x_2] \in \mathbb{R}^2$ and $\mathcal{Y} = \{-1, 1\}$:
$S = \{([-3, 4], 1), ([2, -3], -1), ([-3, -4], -1), ([1, 1.5], 1)\}$.
Assume you decide to use $\mathcal{H} = \{h_1, h_2, h_3, h_4\}$ with
$h_1 = sign(-x_1 - x_2)$
$h_2 = sign(-x_1 + x_2)$
$h_3 = sign(x_1 - x_2)$
$h_4 = sign(x_1 + x_2)$
Your algorithm uses the ERM rule and the 0-1 loss.

- What model h_S is produced in output by your ML algorithm?
- Assume the realizability assumption holds. What can you say about the generalization error $L_{\mathcal{D}}(h_S)$ of h_S?

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).

Polynomial Models

"linear in the parameters"

Consider a regression problem.

Can we as hypothesis set the set of polynomials of degree r with the tools we have already developed for linear regression?

Assume: $\mathcal{X} = \mathbb{R}$

polynomial of degree r: $w_0 \cdot 1 + w_1 x + w_2 x^2 + \ldots + w_r x^r$

Given $x \in \mathbb{R}$, compute the following vector: (feature expansions)

$$\vec{x}' = \begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \\ \vdots \\ x^{r-1} \\ x^r \end{bmatrix} \Rightarrow \vec{w} = [w_0, w_1, \ldots, w_r]^T \Rightarrow \langle \vec{w}, \vec{x}' \rangle = w_0 \cdot 1 + w_1 x + w_2 x^2 + \ldots + w_r x^r$$

$\Rightarrow$ the hypothesis class of linear models for $\vec{x}'$ corresponds to the hypothesis class of polynomials of degree r for x.

Given $\vec{x} \in \mathbb{R}^d$. $\vec{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix}$. You can use the following feature expansion:

$$\vec{x}' = \begin{bmatrix} 1 \\ x_1 \\ x_1^2 \\ x_1^3 \\ \vdots \\ x_1^r \\ x_2 \\ x_2^2 \\ \vdots \\ x_2^r \\ \vdots \\ x_d \\ x_d^2 \\ \vdots \\ x_d^r \end{bmatrix}$$

$\Longrightarrow$ use linear models for $\vec{x}'$!

Different feature expansion:

$\vec{x} \in \mathbb{R}^3$: $\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$, r^2

$$\vec{x}' \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_3 \\ x_1^2 \\ x_2^2 \\ x_3^3 \\ x_1 \cdot x_2 \\ x_1 \cdot x_3 \\ x_2 \cdot x_3 \end{bmatrix}$$

$\Longrightarrow$ build linear models for $\vec{x}'$