MATHEMATICS FOR MACHINE LEARNING

Nagoya University, Fall 2023

Lecture 2

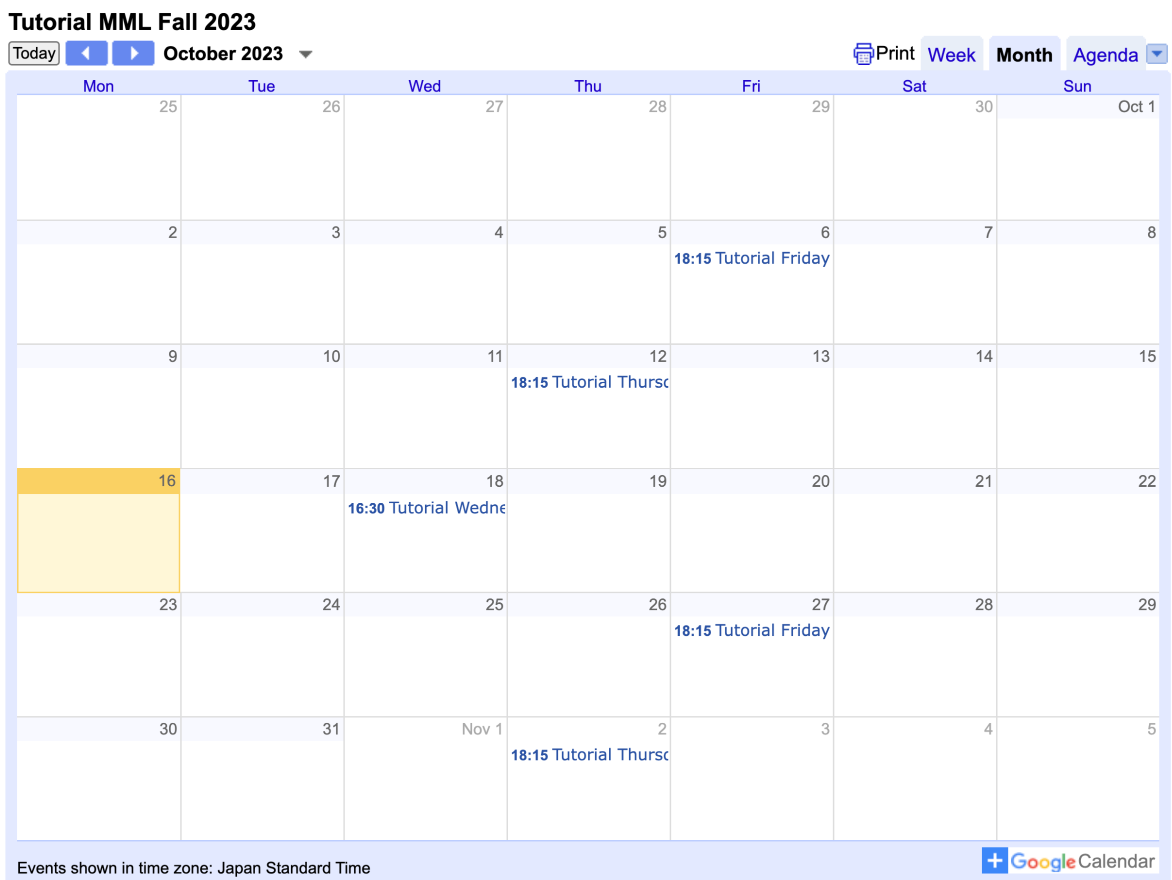
Gradient descent & Linear regression

https://www.henrikbachmann.com/mml2023.html

Tutorial

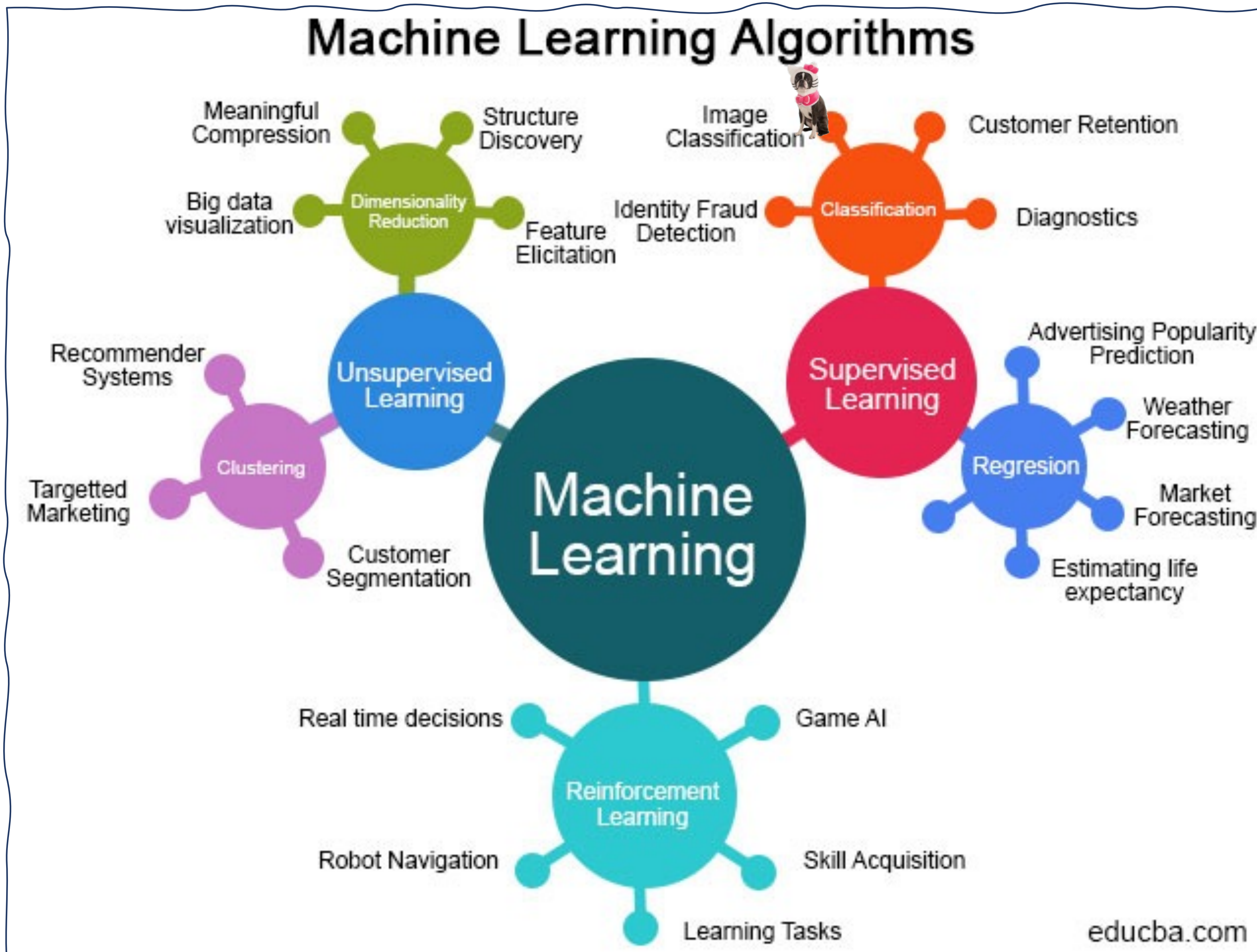
Time: Alternating Thursday (6th period) - Wednesday (5th period) - Friday (6th period)

Next tutorial: This week Wednesday 5th period (16:30) here

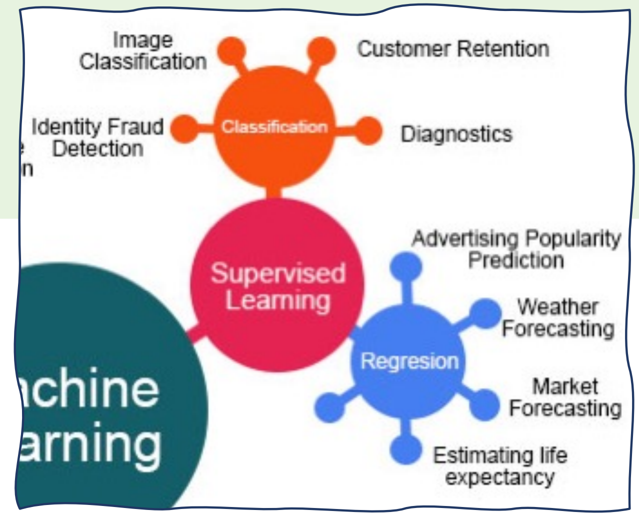


Check the course homepage for a calendar or join Discord for announcements

Recall: Machine learning overview

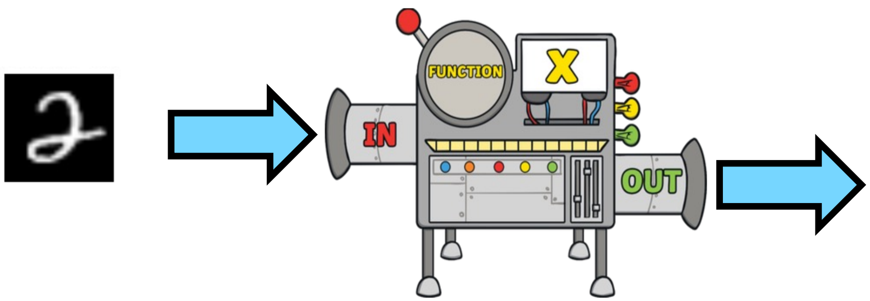


Recall: Supervised learning



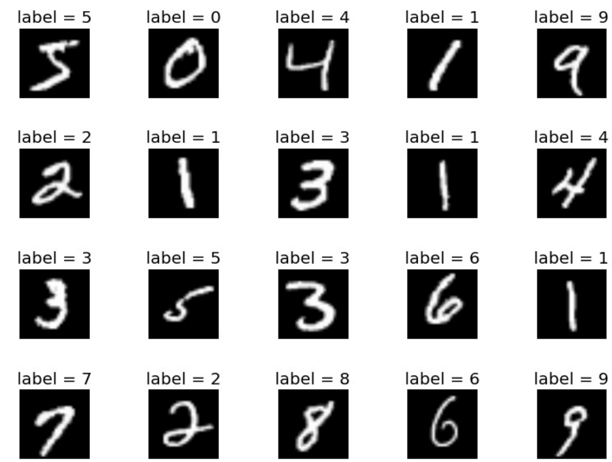
Classification (discrete output)

Output = category (e.g. “dog”, “cat” or “1”,”2”,…,”9”)



”2”

Handwriting recognition

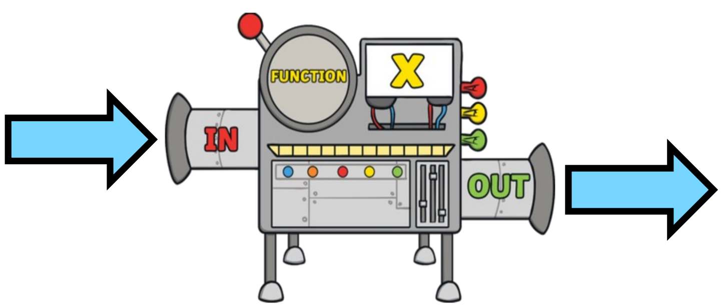


Learning by using some trainingsdata

Regression (continuous output)

Output = real number (like “price”, “weight”,…)

Size: $70m^2$
Rooms: 3



90.000 Yen

Apartment price prediction

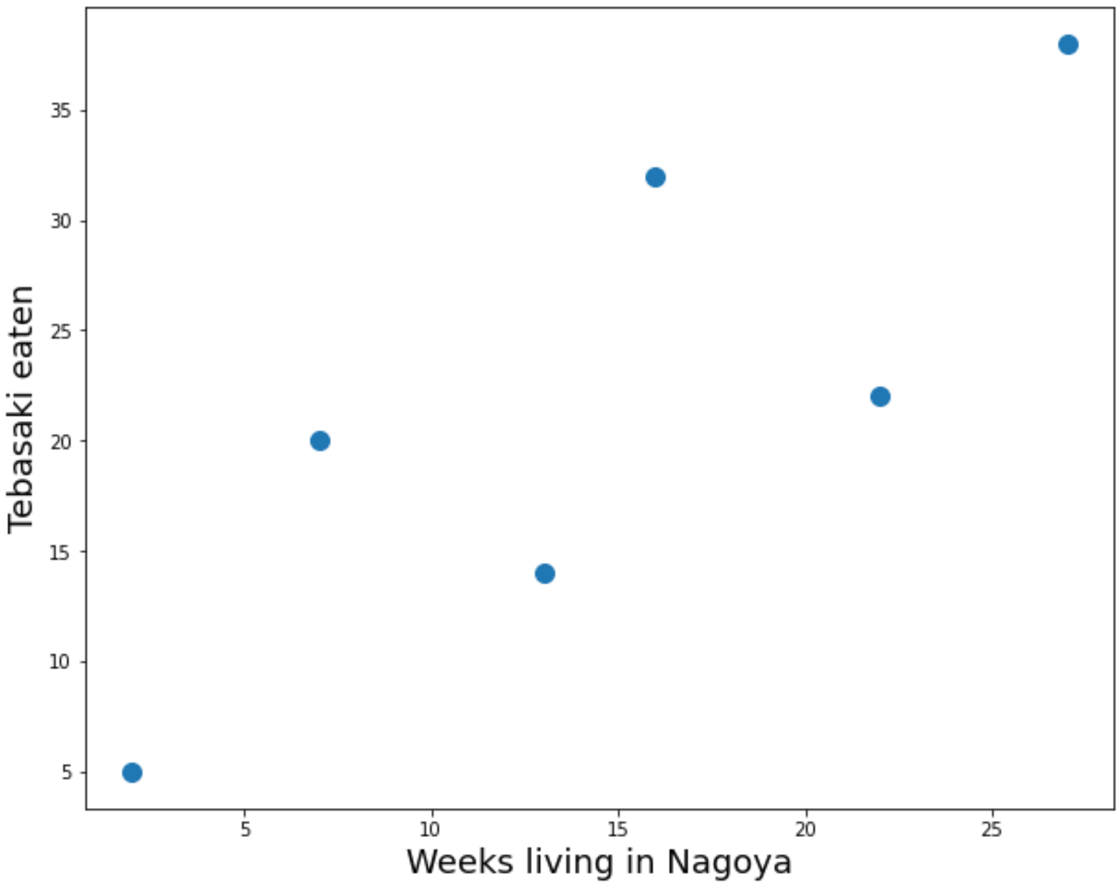
Recall: Supervised learning: Tebasaki example

Have: Some data of ”Weeks living in Nagoya” and “Tebasaki eaten”.

Want: A functions, which creates out of an an arbitrary input for “Weeks living in Nagoya” a prediction for “Tebasaki eaten”.

Weeks living in Nagoya	Tebasaki eaten
2	5
7	20
13	14
16	32
22	22
27	38

Yamachan & Furaibo – possible places to increase the number of ”Tebasaki eaten”



Recall: Supervised learning: Some notations

Training set

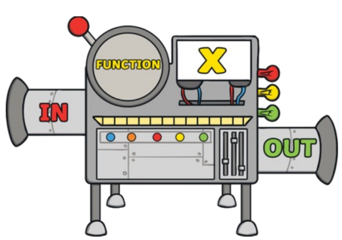
	feature	label (target)
	Weeks living in Nagoya	Tebasaki eaten
	2	5
	7	20
	13	14
Training example	16	32
	22	22
	27	38

Make an assumption how the hyp. looks like

Learning algorithm

(arbitrary number of) Weeks living in Nagoya

(feature)



hypothesis (hyp.)

Predicted number of eaten Tebasaki

(label)

Supervised learning: Notations

- Input values (Feature space): $\mathcal{X}$
- Output value (Label space): $\mathcal{Y}$
- Trainings example: $(x, y) \in \mathcal{X} \times \mathcal{Y}$.
- Trainings set (with n training examples): $\mathcal{T} = \big((x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\big) \in (\mathcal{X} \times \mathcal{Y})^n$.
- hypothesis: A function $h : \mathcal{X} \to \mathcal{Y}$.
- Learning algorithm: An algorithm to create a hypothesis h out of a trainings set $\mathcal{T}$.

Tebaraki example: $X = Y = \mathbb{R}$

Supervised learning – Linear Regression

Learning Algorithm: Linear Regression

Let $\mathcal{X} = \mathbb{R}^d$, i.e. we have d features, and $\mathcal{Y} = \mathbb{R}$. As an Ansatz for the hypothesis we set

$$\boxed{h_\theta(x) := \theta_0 + \theta_1 x_1 + \cdots + \theta_d x_d = \sum_{i=0}^{d} \theta_i x_i\,,}$$

with **parameters (weights)** $\underline{\theta = (\theta_0, \theta_1, \ldots, \theta_d)^T} \in \mathbb{R}^{d+1}$. In the second equation we set $x_0 := 1$.

$X = (x_1)$

Want

Tebaraki: $d = 1$

Want $\theta = \begin{pmatrix} \theta_0 \\ \theta_1 \end{pmatrix}$

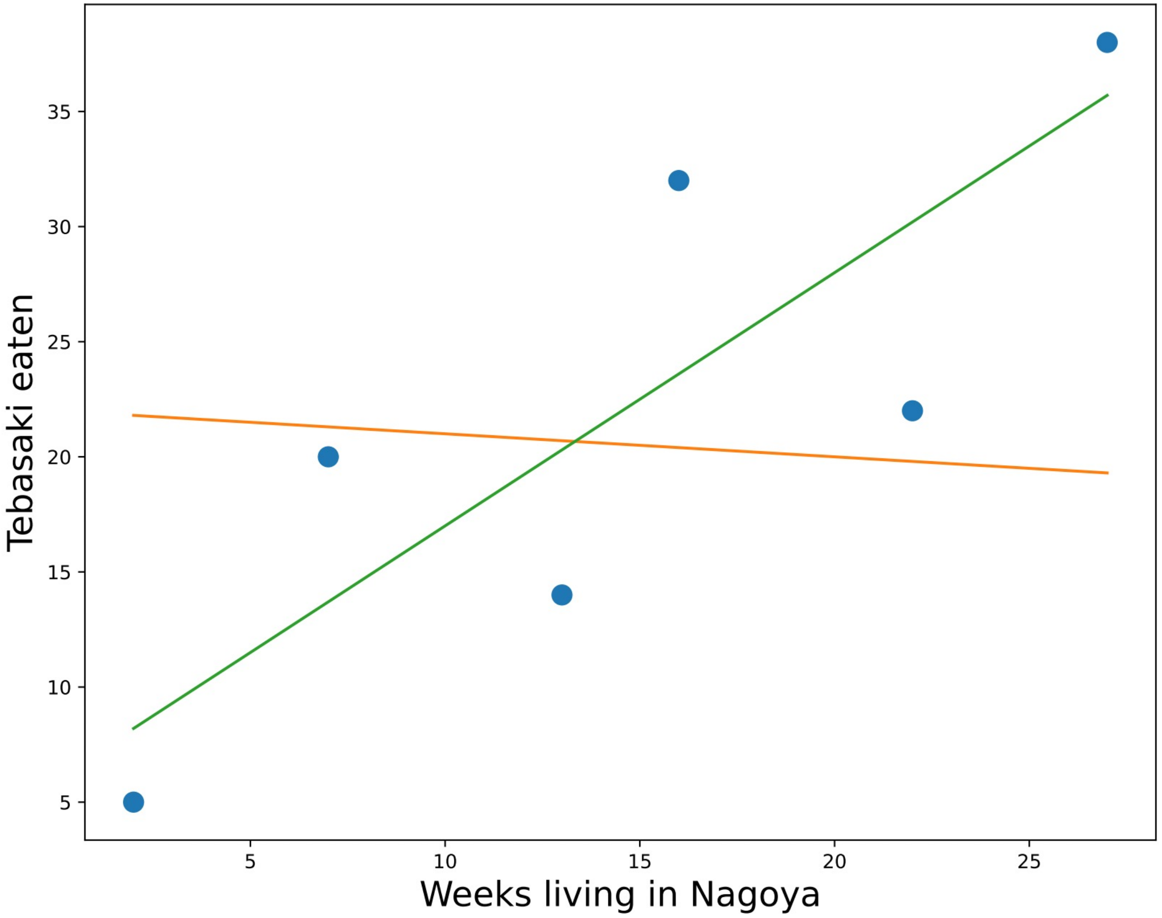
$h_\theta(x) = \theta_0 + \theta_1 x_1$

Goal: Determine the "best "parameters for a given trainings set.

Tebasaki example:

It seems that d=2 works for this case, i.e. we consider

$$h_\theta(x) := \theta_0 + \theta_1 x_1$$



The graphs of $h_{\theta'}$ and h_θ with $\theta' = \begin{pmatrix} 22 \\ -0.1 \end{pmatrix}$ and $\theta = \begin{pmatrix} 6 \\ 1.1 \end{pmatrix}$

Supervised learning – Linear Regression

Measure how good parameters are:

For a given training set $\mathcal{T} = \left((x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\right)$ we define the **cost function** by

$$J(\theta) = \frac{1}{2} \sum_{j=1}^{n} (h_\theta(x^{(j)}) - y^{(j)})^2 \,.$$

The cost function is a function $J : \mathbb{R}^{d+1} \to \mathbb{R}$, which we want to minimize.

Goal rephrased: Minimize the cost function for a given trainings set.

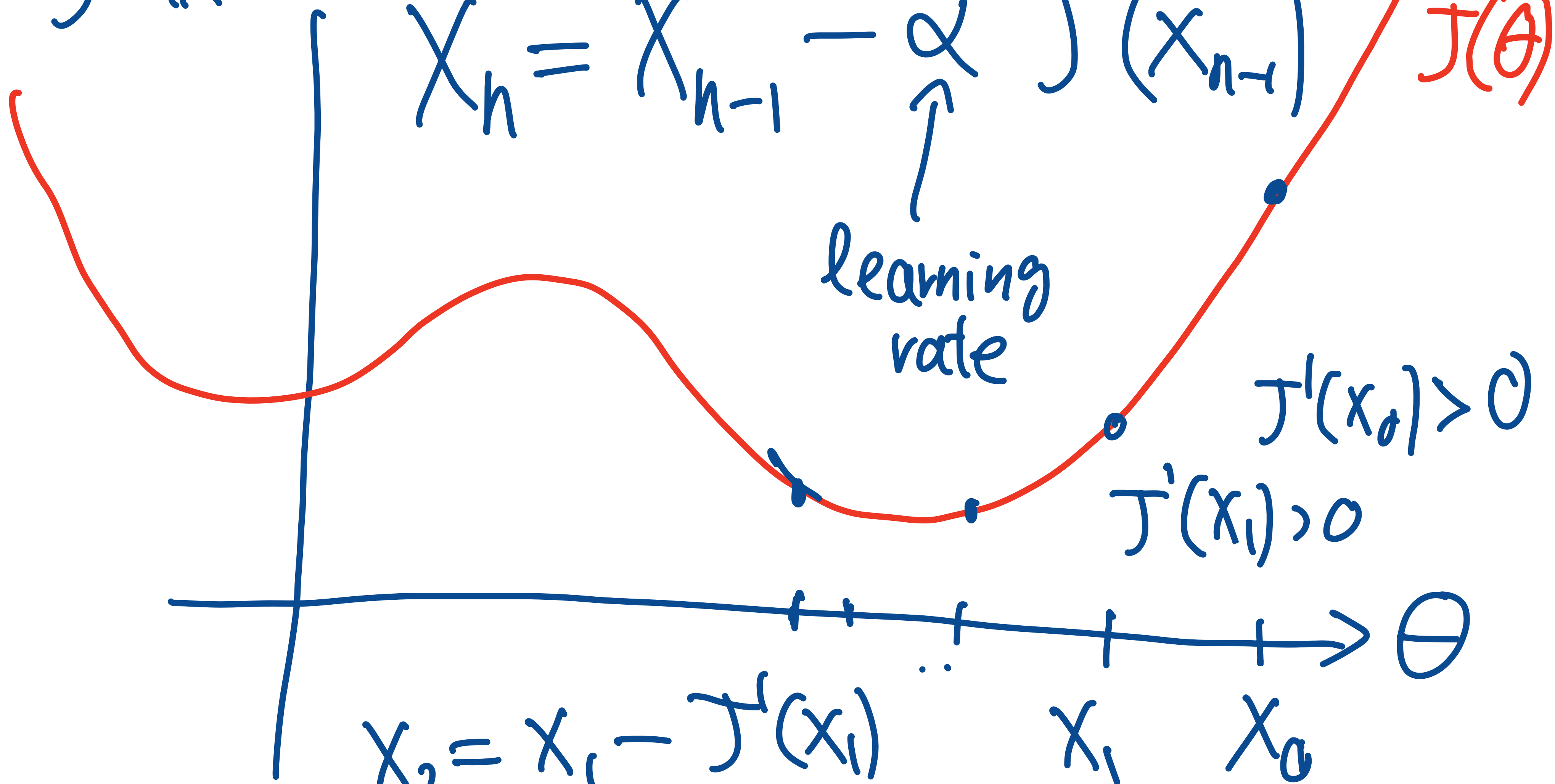
There are several different choices for cost functions. The above choice corresponds is the “least-squares cost function“.

Supervised learning – Minimizing/Gradient

$J: \mathbb{R} \to \mathbb{R}$

$$X_n = X_{n-1} - \alpha J'(X_{n-1})$$

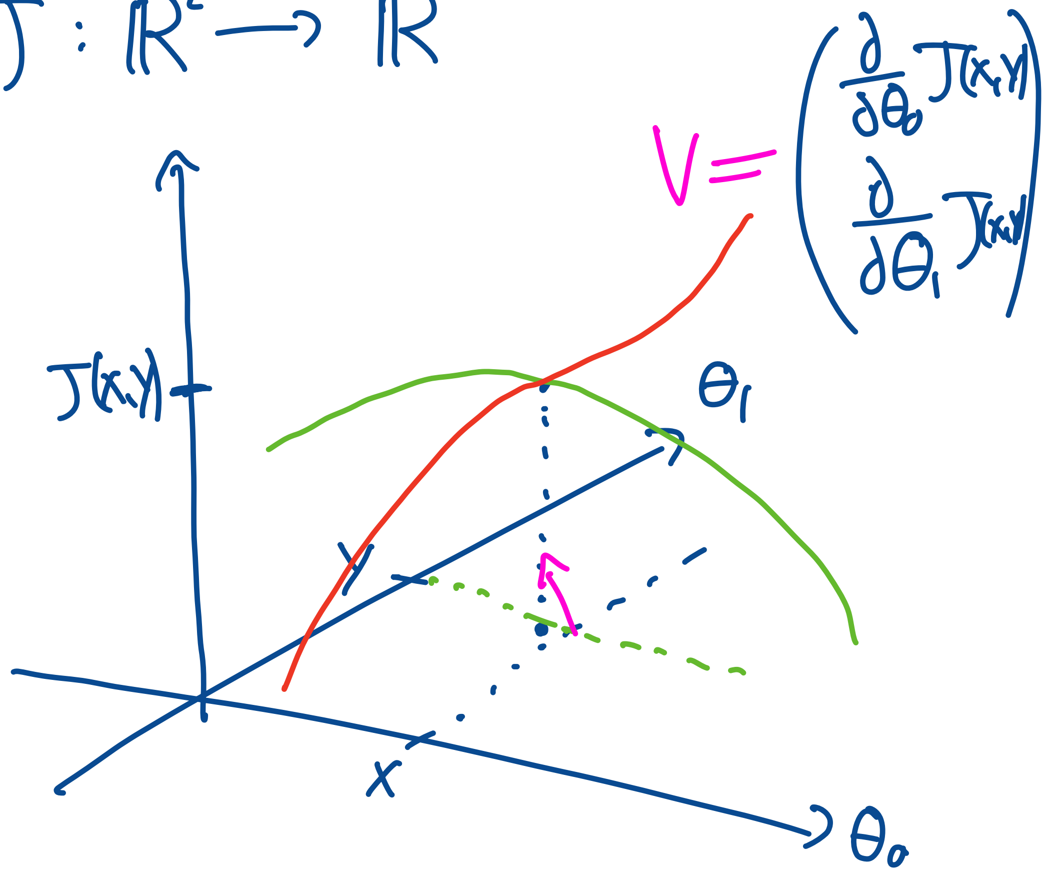
learning rate



$$X_2 = X_1 - J'(X_1)$$

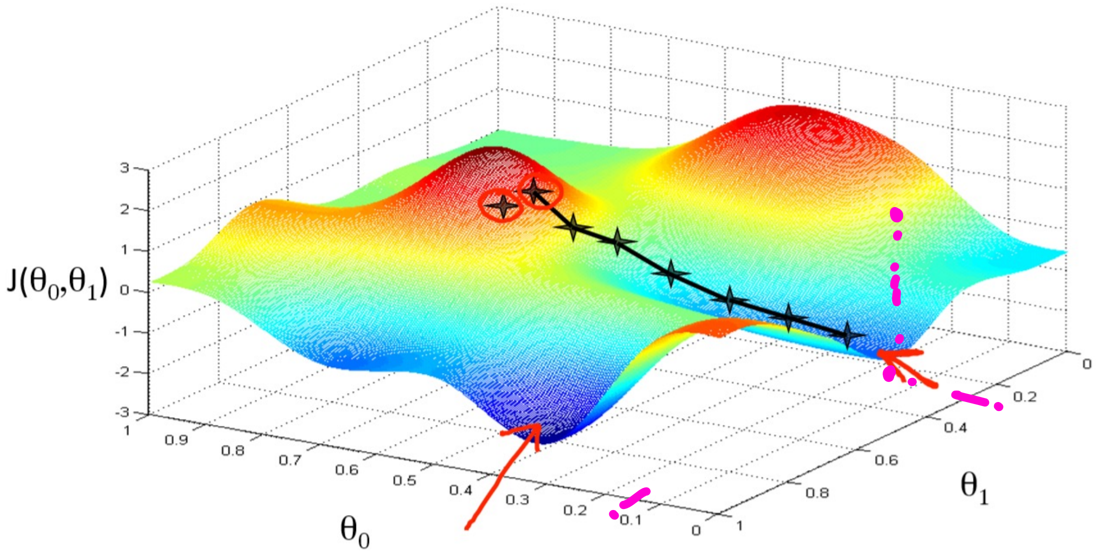
$$X_1 = X_0 - J'(X_0)$$

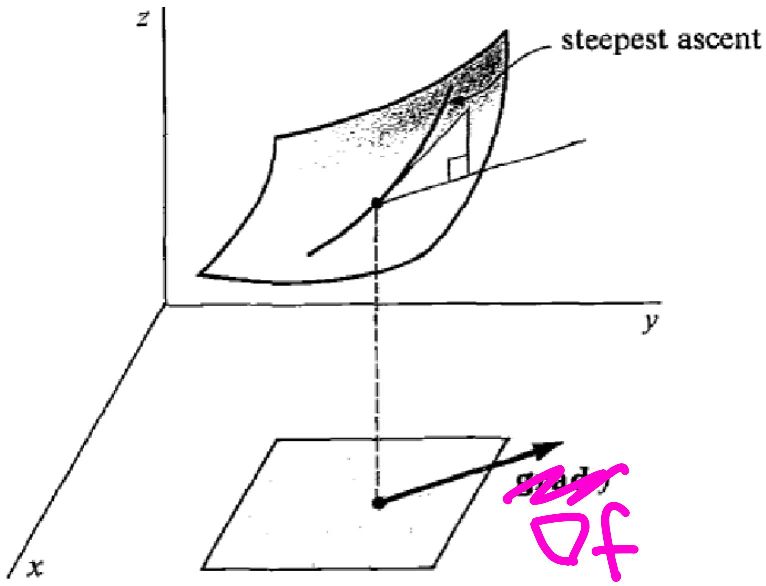
$J : \mathbb{R}^2 \to \mathbb{R}$



Supervised learning – Linear Regression

Gradient descent main idea:

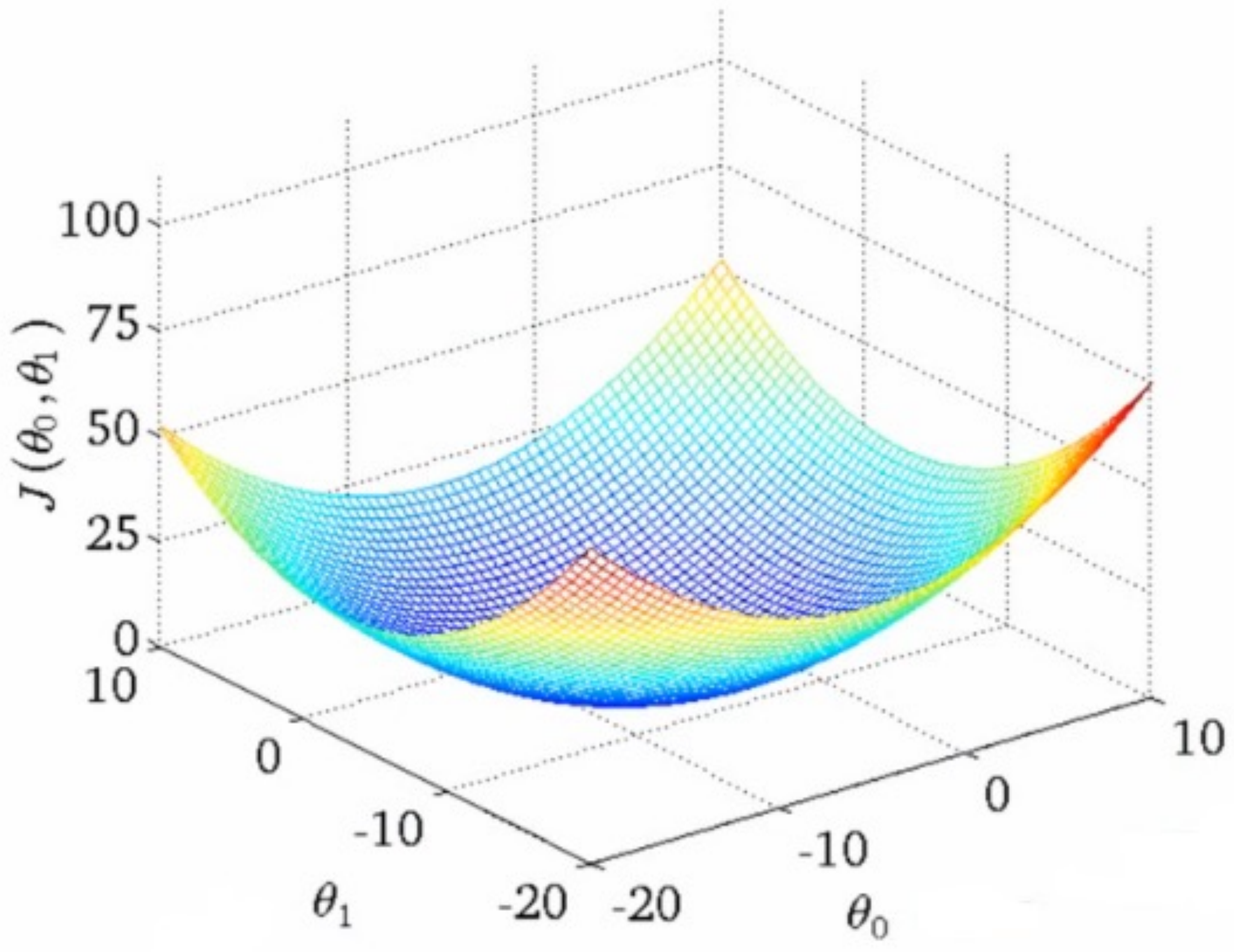




Fact: The gradient shows in the direction of the steepest ascent

Supervised learning – Linear Regression

Gradient descent: d=1 case



Supervised learning – Linear Regression

$J: \mathbb{R}^{d+1} \to \mathbb{R}$

The **gradient of** J is defined by

$$\nabla J = \begin{pmatrix} \frac{\partial}{\partial \theta_0} J \\ \frac{d\partial}{\partial \theta_1} J \\ \vdots \\ \frac{\partial}{\partial \theta_d} J \end{pmatrix}$$

Goal: Find $\theta \in \mathbb{R}^{d+1}$, such that $J(\theta)$ is minimal.

Gradient descent algorithm (rough version).

i) Start with a random starting value for the parameters, e.g. $\theta = 0 = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$.

ii) Change the parameters in the opposite direction of the steepest ascent, i.e. opposite direction of the gradient. This means we want to subtract the gradient from the current parameters, weighted by a factor $\alpha \in \mathbb{R}$, the **learning rate**.

The new parameters θ are therefore given by:

$$\theta := \theta - \alpha \nabla J(\theta).$$

iii) Repeat step ii) until the value $J(\theta)$ does not change anymore.

Supervised learning – Linear Regression

$$x = \begin{pmatrix} x_1 \end{pmatrix}$$

$$\theta = \begin{pmatrix} \theta_0 \\ \theta_1 \end{pmatrix}$$

Gradient descent: d=1 case

Tebasaki example

$$h_\theta(x) = \theta_0 + \theta_1 x$$

$$J(\theta) = \frac{1}{2} \sum_{j=1}^{6} \Big(\underline{h_\theta(x^{(j)}) - y^{(j)}} \Big)^2$$

$$\theta_0 + \theta_1 x^{(j)} - y^{(j)}$$

$$\nabla J(\theta) = \begin{pmatrix} \frac{\partial}{\partial \theta_0} J(\theta) \\ \frac{\partial}{\partial \theta_1} J(\theta) \end{pmatrix} = \begin{pmatrix} \sum_{j=1}^{6} \left(h_\theta(x^{(j)}) - y^{(j)} \right) \\ \sum_{j=1}^{6} \left(h_\theta(x^{(j)}) - y^{(j)} \right) x^{(j)} \end{pmatrix}$$

$$\big(g(f(\theta_0))\big)' = f'(\theta_0)\, g'(f(\theta_0))$$

$$g(\ ) = \frac{1}{2} x^2$$

$$f(x) = \theta_0 + \theta_1 x^{(j)} + y^{(j)}$$

Now Python examples!

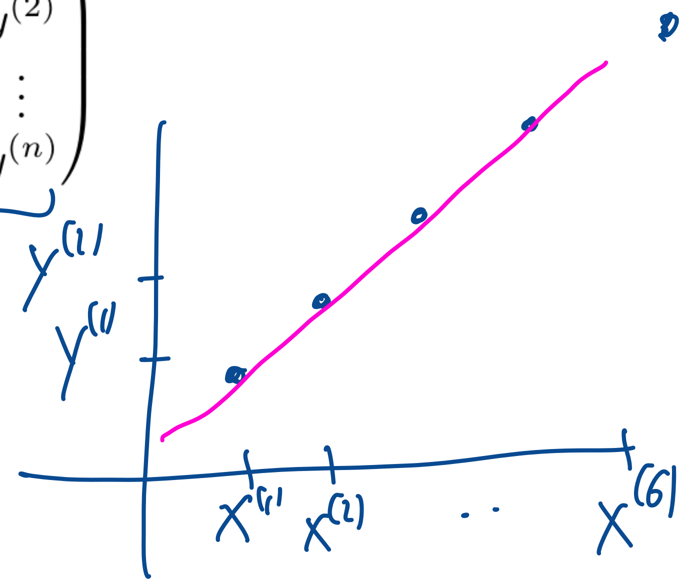
Linear Regression: Using Linear Algebra

Assume we have a training set $\mathcal{T} = ((x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)}))$.

If all training examples lie on a line, then the $\theta = \begin{pmatrix}\theta_0 \\ \theta_1\end{pmatrix}$ we are looking for solves the linear system

$$\begin{pmatrix}\theta_0 + x^{(1)}\theta_1 \\ \theta_0 + x^{(2)}\theta_1 \\ \vdots \\ \theta_0 + x^{(n)}\theta_1\end{pmatrix} = \underbrace{\begin{pmatrix}1 & x^{(1)} \\ 1 & x^{(2)} \\ \vdots & \vdots \\ 1 & x^{(n)}\end{pmatrix}}_{A} \begin{pmatrix}\theta_0 \\ \theta_1\end{pmatrix} = \underbrace{\begin{pmatrix}y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)}\end{pmatrix}}_{Y}$$

$h(x) = \theta_0 + \theta_1 x$



$$A\theta = y$$

$$A\theta - Y = 0$$

Linear Regression: Using Linear Algebra

Assume we have a training set $\mathcal{T} = \big((x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\big)$.

If all training examples lie on a line, then the $\theta = \begin{pmatrix} \theta_0 \\ \theta_1 \end{pmatrix}$ we are looking for solves the linear system

$$\begin{pmatrix} 1 & x^{(1)} \\ 1 & x^{(2)} \\ \vdots & \vdots \\ 1 & x^{(n)} \end{pmatrix} \begin{pmatrix} \theta_0 \\ \theta_1 \end{pmatrix} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{pmatrix}$$

- This is usually not the case (like in the Tebasaki example).
- We are looking for the “best” solution of a linear system.

Goal: Find $\theta \in \mathbb{R}^2$, such that $||A\theta - y||$ is minimal.

Linear Regression: Recalling Linear Algebra

Let $A \in \mathbb{R}^{m \times n}$ *be a matrix.*

The **image of** A *is defined by*

$$\mathrm{im}(A) = \{y \in \mathbb{R}^m \mid Av = y \textit{ for some } v \in \mathbb{R}^n\} \ .$$

The **kernel of** A *is defined by*

$$\ker(A) = \{v \in \mathbb{R}^n \mid Av = 0\} \ .$$

The **dot-product** of two vectors $u, v \in \mathbb{R}^n$ is defined by

$$u \bullet v = u^T v = u_1 v_1 + \cdots + u_n v_n \ .$$

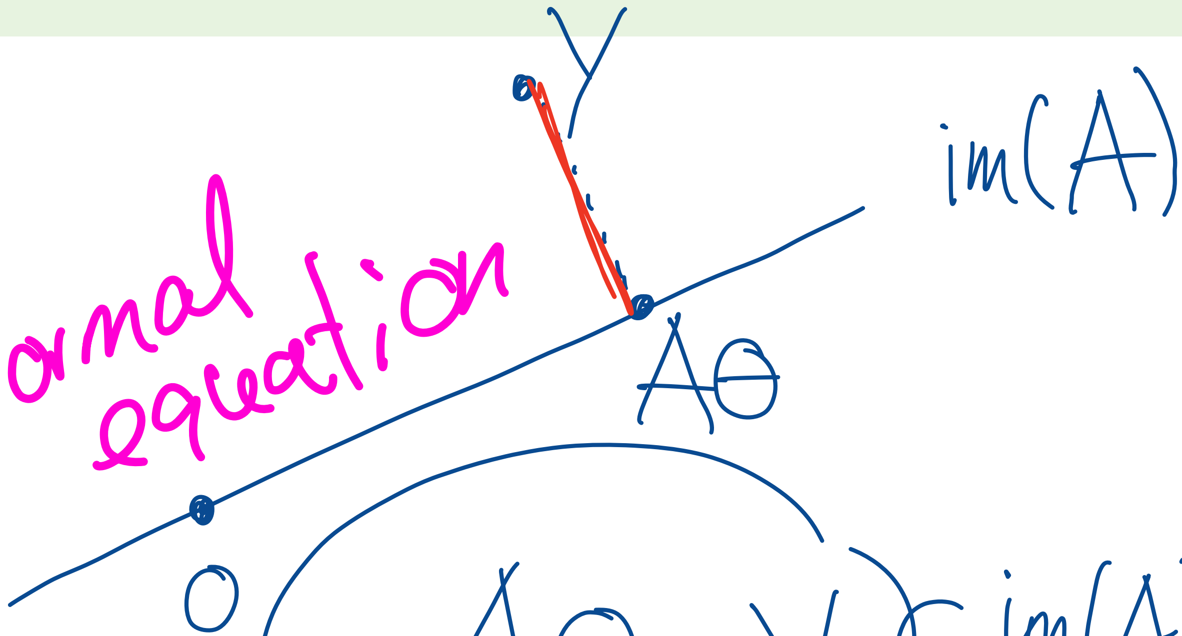
For a subspace $U \subset \mathbb{R}^n$, the **orthogonal complement of** U is defined by

$$U^\perp = \{v \in \mathbb{R}^n \mid u \bullet v = 0, \forall u \in U\} \ .$$

Example

$A\theta = y$ has no sol.

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



Normal equation

$$A\theta - y \in \mathrm{im}(A)^{\perp} = \mathrm{Ker}(A^T)$$

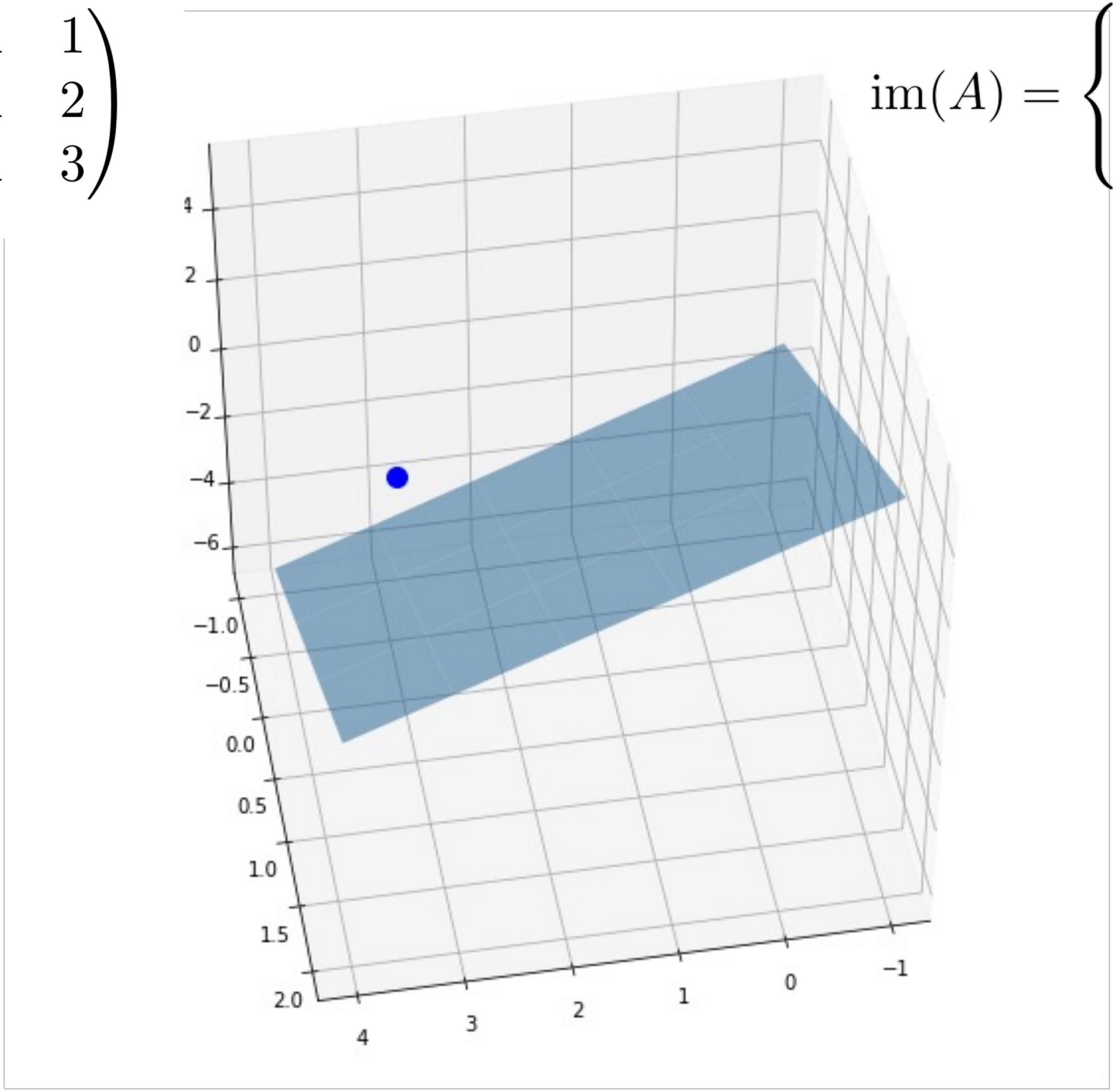
$$\mathrm{im}(A) = \mathrm{span}\left\{ \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \right\}$$

$$\overbrace{\begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \end{pmatrix}}^{A^T} v = \begin{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \cdot v \\ \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \cdot v \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\boxed{A^T A \theta = A^T y} \quad A^T(A\theta - y) = 0$$

Example

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



$$\mathrm{im}(A) = \left\{ \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} \in \mathbb{R}^3 \,\middle|\, v_3 = -v_1 + 2v_2 \right\}$$

$$y = \begin{pmatrix} 3 \\ 0 \\ 0 \end{pmatrix}$$

Goal: Find $\theta \in \mathbb{R}^2$, such that $||A\theta - y||$ is minimal.

Linear Regression: The normal equation

Goal: Find $\theta \in \mathbb{R}^2$, such that $||A\theta - y||$ is minimal.

Theorem: If $\theta \in \mathbb{R}^2$ is a solution to

$$A^T A\theta = A^T y \,,$$

then $||A\theta - y||$ is minimal.

If $\ker(A) = \{0\}$, then $A^T A$ is invertible and an explicit solution for the best θ is given by

$$\theta = (A^T A)^{-1} A^T y \,.$$

Polynomial Regression: Is just linear regression..