Outline

This lecture introduces us to the topic of supervised learning. Here the data consists of input-output pairs. Inputs are also often referred to as covariates, predictors and features; while outputs are known as variates, targets and labels. The goal of the lecture is for you to:

- Understand the supervised learning setting.
- Understand linear regression (aka least squares)
- Understand how to apply linear regression models to make predictions.
- Learn to derive the least squares estimate.

Linear supervised learning

- Many real processes can be approximated with linear models.

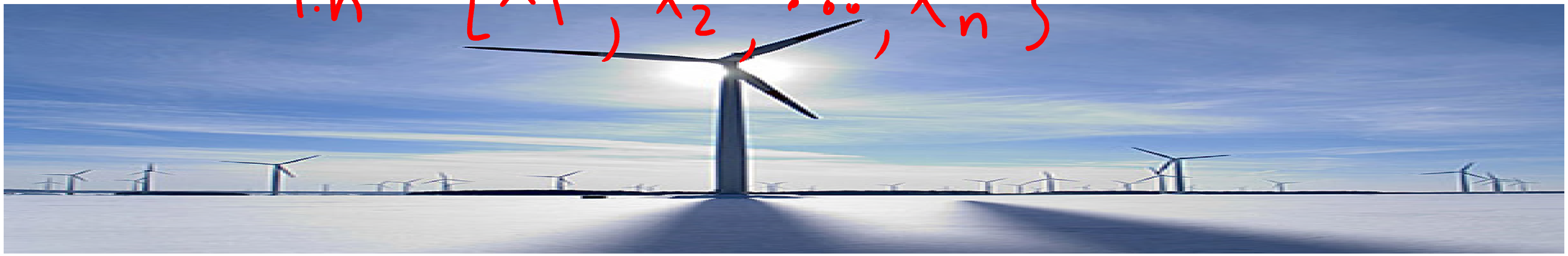
- Linear regression often appears as a module of larger systems.

- Linear problems can be solved analytically.

- Linear prediction provides an introduction to many of the core concepts of machine learning.

We are given a training dataset of n instances of input-ouput pairs $\{\mathbf{x}_{1:n}, \mathbf{y}_{1:n}\}$. Each input $\mathbf{x}_i \in \mathbb{R}^{1\times d}$ is a vector with d attributes. The inputs are also known as predictors or covariates. The output, often referred to as the target, will be assumed to be univariate, $\mathbf{y}_i \in \mathbb{R}$, for now.

$$X_{1:n} = \{X_1, X_2, \ldots, X_n\}$$



A typical dataset with $n = 4$ instances and 2 attributes would look like the following table:

	Wind speed	People inside building		Energy requirement	
x_1	100	2		y_1 5	
	50	42	$d=2$	25	
x_3	45	31		22	y_3
	60	35		18	

Energy demand prediction

Given the training set $\{\mathbf{x}_{1:n}, \mathbf{y}_{1:n}\}$, we would like to learn a model of how the inputs affect the outputs. Given this model and a new value of the input $\mathbf{x}_{n+1}$, we can use the model to make a prediction $\widehat{y}(\mathbf{x}_{n+1})$.

1. TRAINING
   learning

   DATA $\{x_{1:n}, y_{1:n}\} \rightarrow \boxed{\text{Learner}} \rightarrow$ model parameters Θ

2. TESTING
   prediction

   $x_{n+1}, \Theta \rightarrow \boxed{\text{Predictor}} \rightarrow \hat{y}_{n+1}$

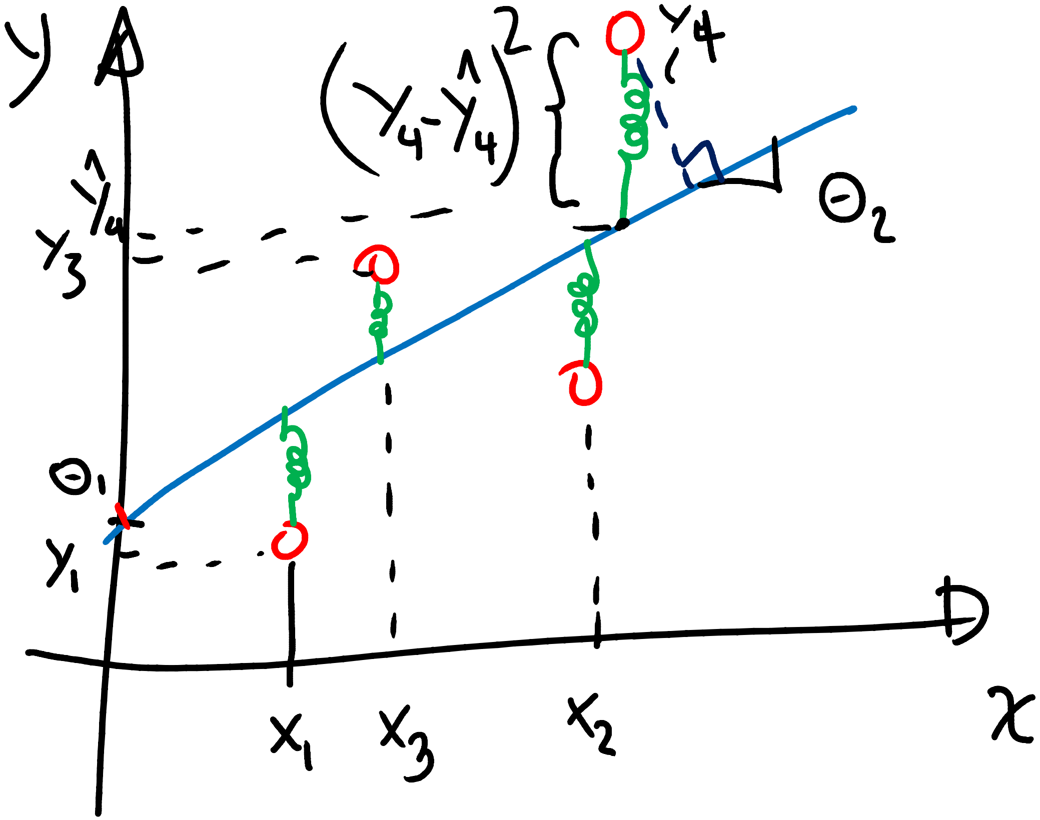
$$\hat{y}(\mathbf{x}_i) = 1\theta_1 + x_i\theta_2$$

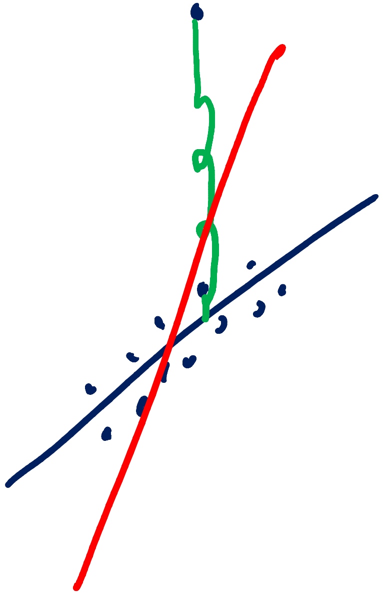
$$J(\boldsymbol{\theta}) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}(y_i - \theta_1 - x_i\theta_2)^2$$

Objective function

Energy

Loss





Linear prediction

In general, the linear model is expressed as follows:

$$\hat{y}_i = \sum_{j=1}^{d} x_{ij}\theta_j, \quad = \overset{1}{\cancel{x_{i1}}}\theta_1 + x_{i2}\theta_2 + \cdots + x_{id}\theta_d$$

where we have assumed that $x_{i1} = 1$ so that θ_1 corresponds to the intercept of the line with the vertical axis. θ_1 is known as the bias or offset.

In matrix form, the expression for the linear model is:

$$\boxed{\hat{\mathbf{y}} = \mathbf{X}\boldsymbol{\theta},}$$

with $\hat{\mathbf{y}} \in \mathbb{R}^{n \times 1}$, $\mathbf{X} \in \mathbb{R}^{n \times d}$ and $\boldsymbol{\theta} \in \mathbb{R}^{d \times 1}$. That is,

$$\begin{array}{cc} i & \begin{bmatrix} \hat{y}_1 \\ \vdots \\ \hat{y}_n \end{bmatrix} \end{array} = \begin{bmatrix} x_{11} & \cdots & x_{1d} \\ \vdots & \vdots & \vdots \\ x_{n1} & \cdots & x_{nd} \end{bmatrix} \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_d \end{bmatrix}.$$

Wind speed	People inside building	Energy requirement
100	2	5
50	42	25
45	31	22
60	35	18

For our energy prediction example, we would form the following matrices with $n = 4$ and $d = 3$:

$$\mathbf{y} = \begin{bmatrix} 5 \\ 25 \\ 22 \\ 18 \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} 1 & 100 & 2 \\ 1 & 50 & 42 \\ 1 & 45 & 31 \\ 1 & 60 & 35 \end{bmatrix}, \quad \boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}.$$

Suppose that $\boldsymbol{\theta} = [1 \quad 0 \quad 0.5]^T$. Then, by multiplying $\mathbf{X}$ times $\boldsymbol{\theta}$, we would get the following predictions on the training set:

$$\widehat{\mathbf{y}} = \begin{bmatrix} 2 \\ 22 \\ 16.5 \\ 18.5 \end{bmatrix} = \begin{bmatrix} 1 & 100 & 2 \\ 1 & 50 & 42 \\ 1 & 45 & 31 \\ 1 & 60 & 35 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0.5 \end{bmatrix}.$$

Linear prediction on test data

Likewise, for a point that we have never seen before, say x = [50 20], we generate the following prediction:

$$\hat{y}(x) = [1\ 50\ 20] \begin{bmatrix} 1 \\ 0 \\ 0.5 \end{bmatrix} = 1 + 0 + 10 = 11.$$

$$J(\boldsymbol{\theta}) = (\mathbf{y} - \mathbf{X}\boldsymbol{\theta})^T(\mathbf{y} - \mathbf{X}\boldsymbol{\theta}) = \sum_{i=1}^{n}(y_i - \mathbf{x}_i^T\boldsymbol{\theta})^2$$

$\underline{x} = (x_1 \;\; x_2)^T$ $\quad$ $\underline{y} = (y_1 \;\; y_2)^T$ $\quad$ $\underline{x}^T\underline{y} = [x_1 \;\; x_2]\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$

$$= x_1y_1 + x_2y_2$$

$$= \sum_i x_i y_i$$

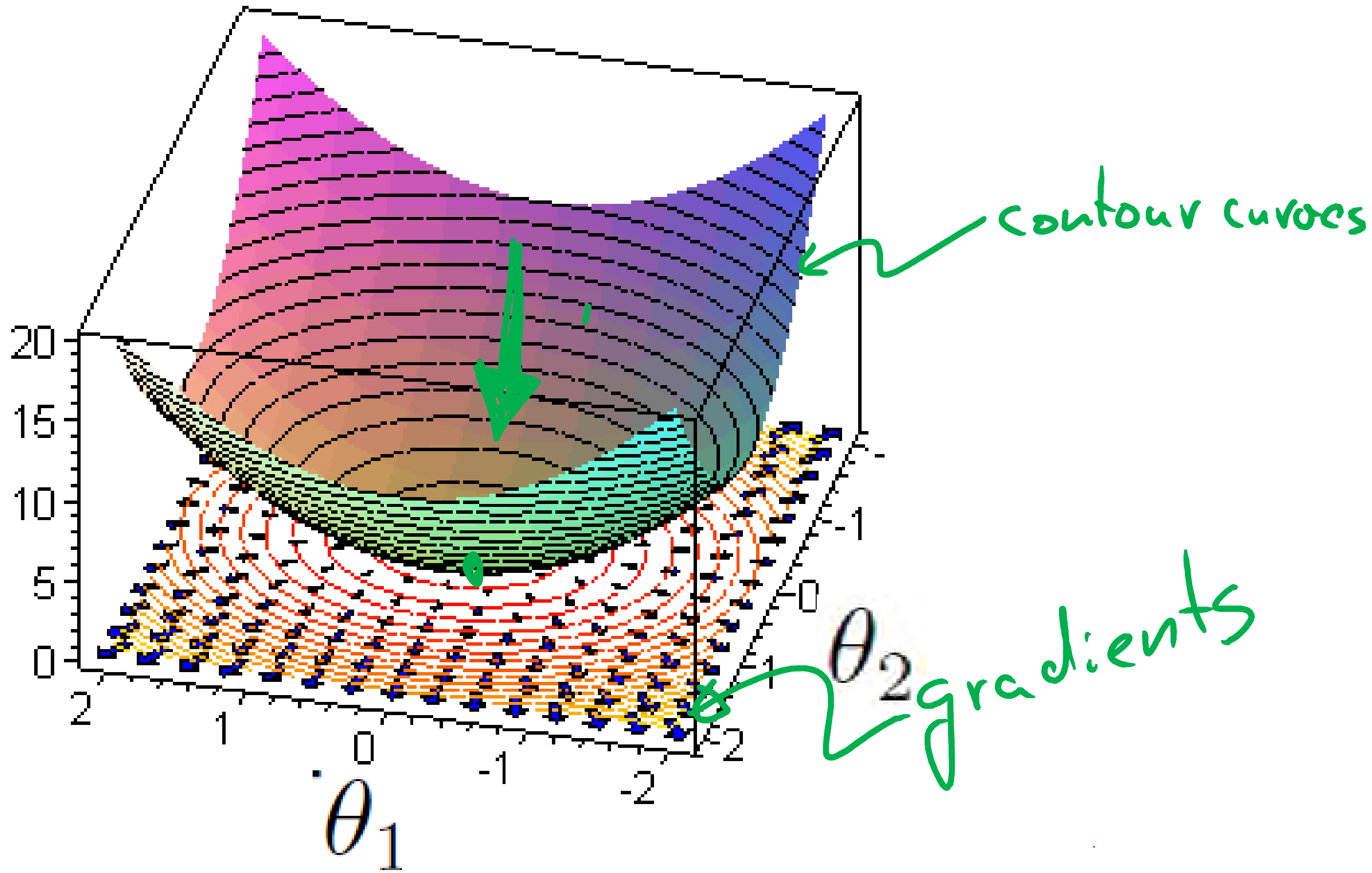
$$\overbrace{\left(\begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} - \begin{bmatrix} x_1^T \\ x_2^T \\ \\ x_n^T \end{bmatrix} [\theta] \right)}$$

Optimization approach

Our aim is to mininimise the quadratic cost between the output labels and the model predictions

$$J(\boldsymbol{\theta}) = \underbrace{(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})^T(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})} = \underbrace{\sum_{i=1}^{n}(y_i - \mathbf{x}_i^T\boldsymbol{\theta})^2}$$

$\underline{\theta \in \mathbb{R}^2}$



Finding the solution by differentiation

$$J(\boldsymbol{\theta}) = \sum_{i=1}^{n}(y_i - \widehat{y}_i)^2 = \sum_{i=1}^{n}(y_i - \theta_1 - x_i\theta_2)^2$$

$$\left.\begin{array}{l}\dfrac{d\,J(\theta)}{d\theta_1} \longrightarrow g(\theta_1, \theta_2) \\[2em] \dfrac{d\,J(\theta)}{d\theta_2} \longrightarrow g_2(\theta_1, \theta_2)\end{array}\right\} \text{Normal Eqns.}$$

Least squares

Wiki

Finding the solution by differentiation

$$J(\boldsymbol{\theta}) = (\underbrace{\mathbf{y}}_{}^{n\times 1} - \overset{n\times d}{\mathbf{X}}\overset{d\times 1}{\boldsymbol{\theta}})^T(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})$$

$$X = \begin{bmatrix} x_1^T \\ x_2^T \end{bmatrix}$$

We will need the following results from matrix differentiation:

$$\frac{\partial \mathbf{A}\theta}{\partial \boldsymbol{\theta}} = \mathbf{A}^T \text{ and } \frac{\partial \boldsymbol{\theta}^T\mathbf{A}\boldsymbol{\theta}}{\partial \boldsymbol{\theta}} = 2\mathbf{A}^T\boldsymbol{\theta}$$

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \frac{\partial}{\partial \theta}\left(y^T y - 2y^T X\theta + \theta^T X^T X \theta \right)$$

$$= 0 - 2X^T y + X^T X \theta \overset{\text{equals}}{=} 0$$

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

Torch: Data

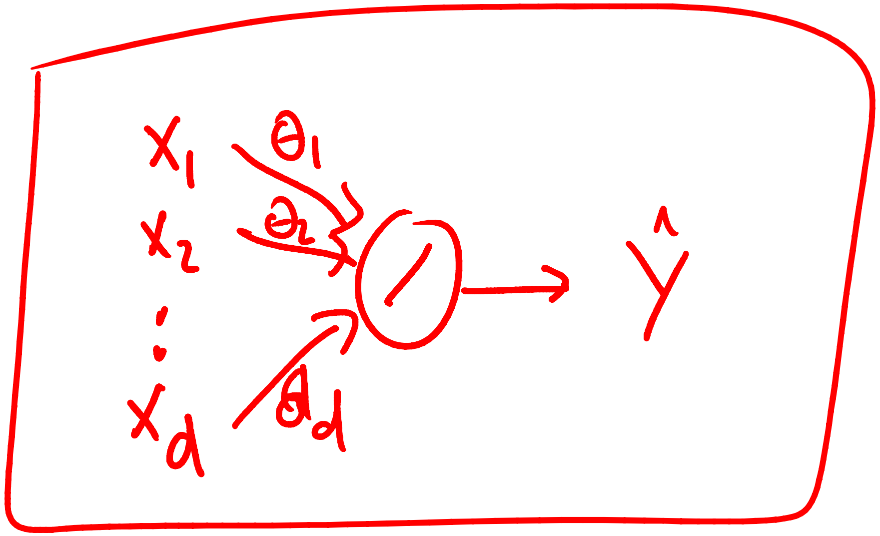
Torch 7

```lua
--  {corn, fertilizer, insecticide}
data = torch.Tensor{
   {40,  6,  4},
   {44, 10,  4},
   {46, 12,  5},
   {48, 14,  7},
   {52, 16,  9},
   {58, 18, 12},
   {60, 22, 14},
   {68, 24, 20},
   {74, 26, 21},
   {80, 32, 24}
}
```

http://torch.cogbits.com/doc/tutorials_supervised/
https://github.com/torch/demos/blob/master/linear-regression/example-linear-regression.lua

Torch: Model

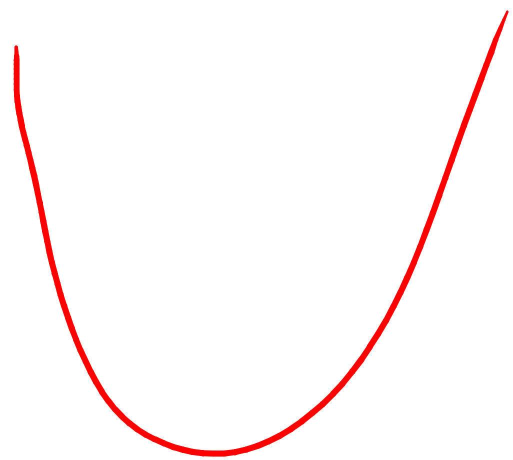
```
model = nn.Sequential()
ninputs = 2; noutputs = 1
model:add(nn.Linear(ninputs, noutputs))
```



http://torch.cogbits.com/doc/tutorials_supervised/
https://github.com/torch/demos/blob/master/linear-regression/example-linear-regression.lua

Torch: Loss / objective

```
criterion = nn.MSECriterion()
```



http://torch.cogbits.com/doc/tutorials_supervised/
https://github.com/torch/demos/blob/master/linear-regression/example-linear-regression.lua

Torch: Compute parameters

http://torch.cogbits.com/doc/tutorials_supervised/
https://github.com/torch/demos/blob/master/linear-regression/example-linear-regression.lua