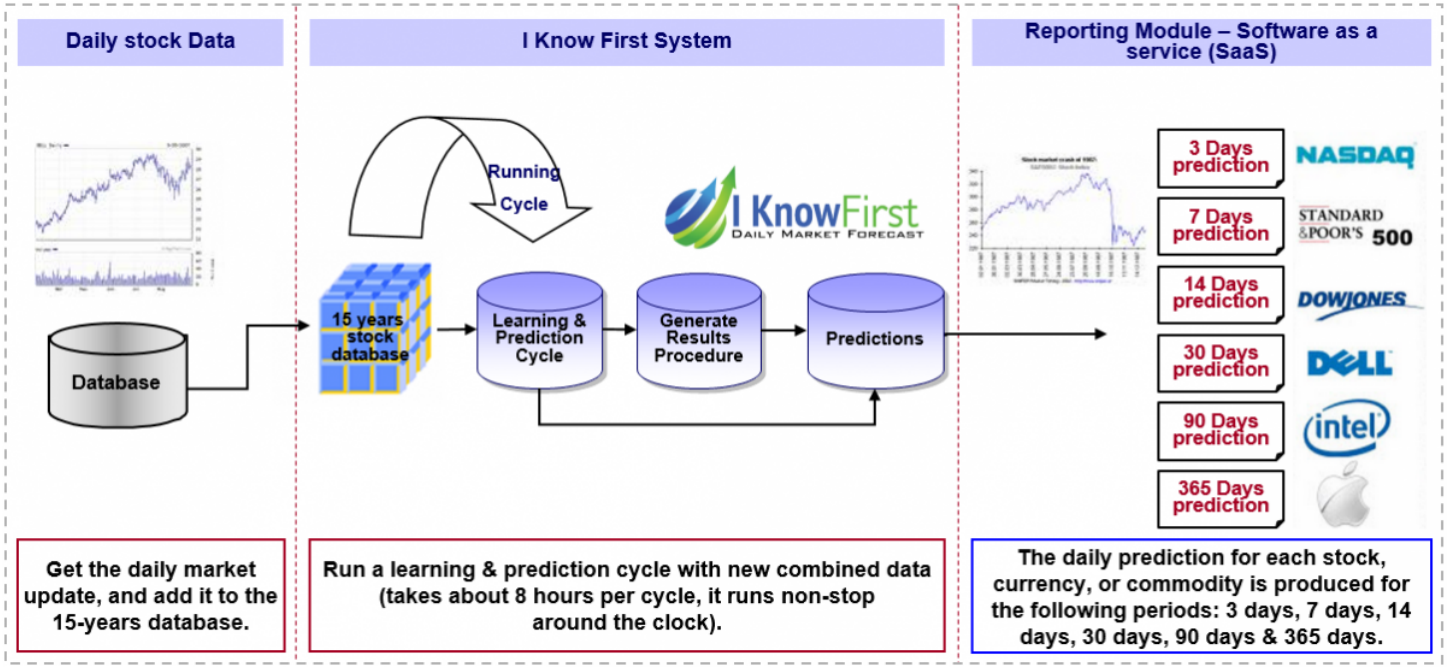
CSC4007 Advanced Machine Learning

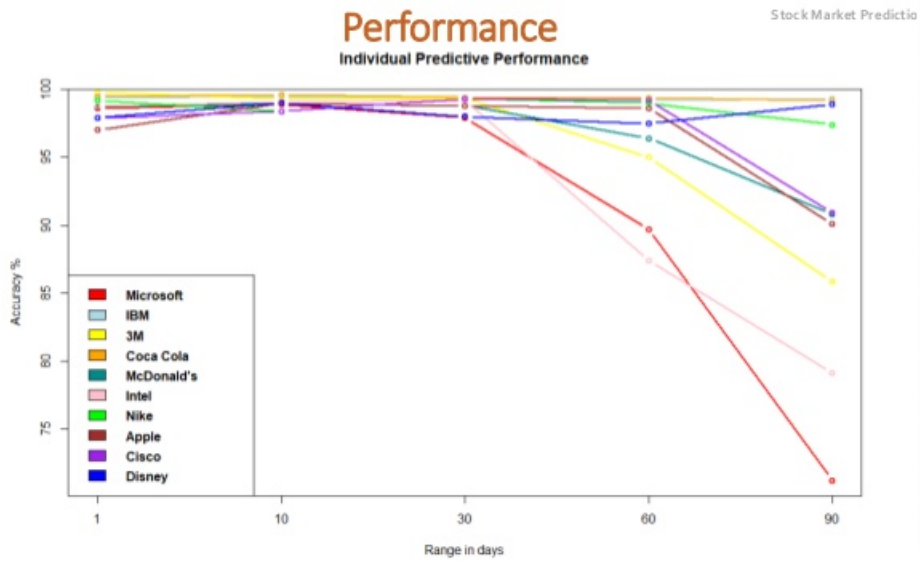
Lesson 03: Linear Regression

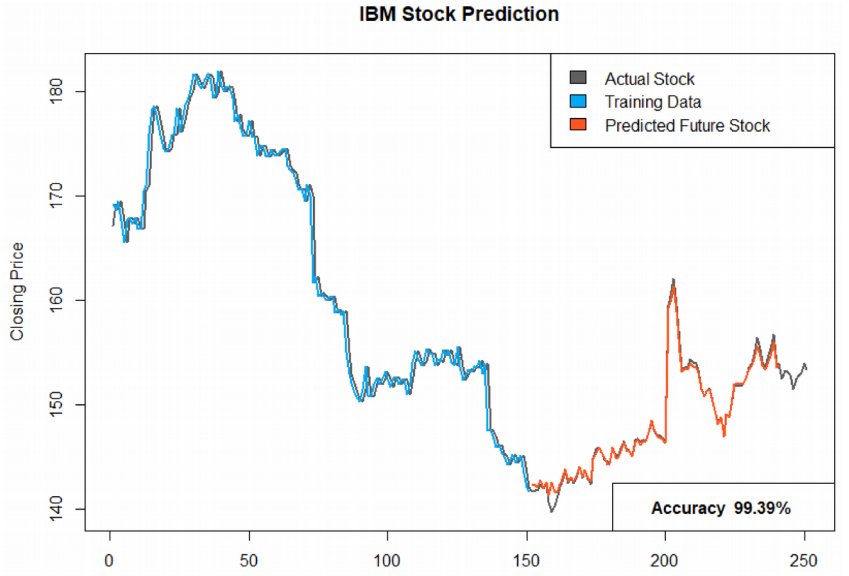
by Vien Ngo
EEECS / ECIT / DSSC

Regression in Stock Market Prediction

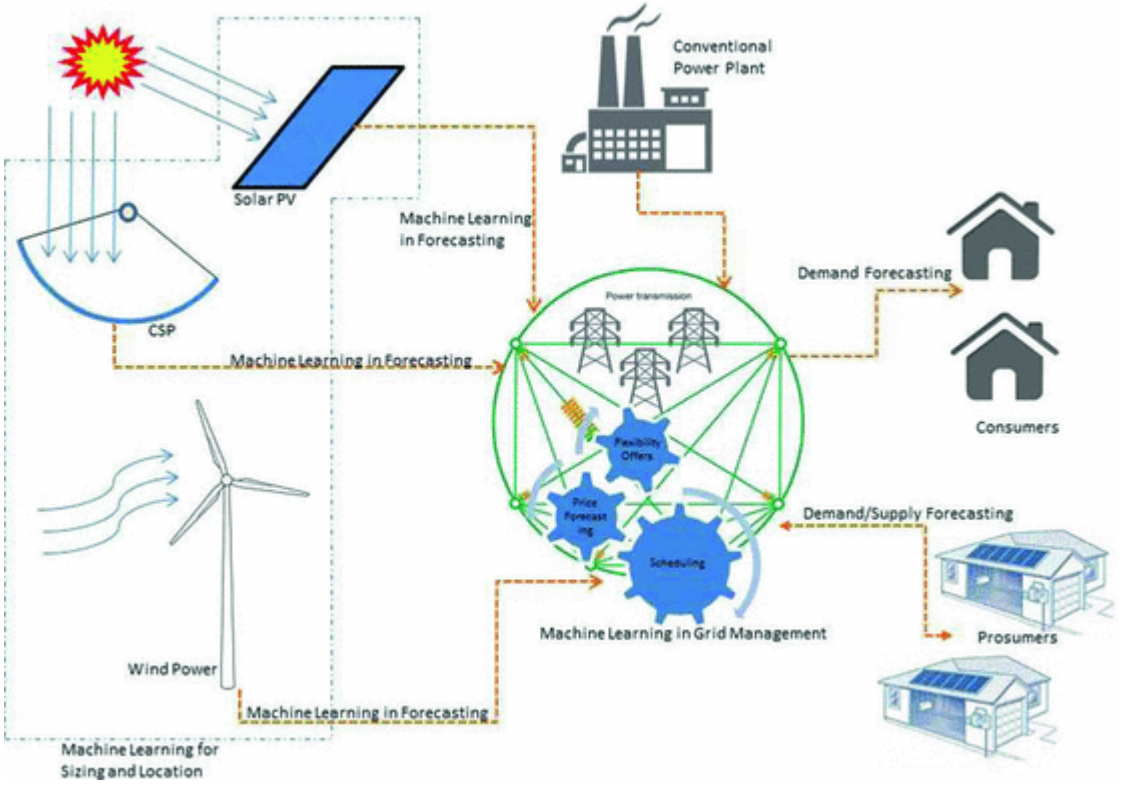


https://www.re-work.co/blog

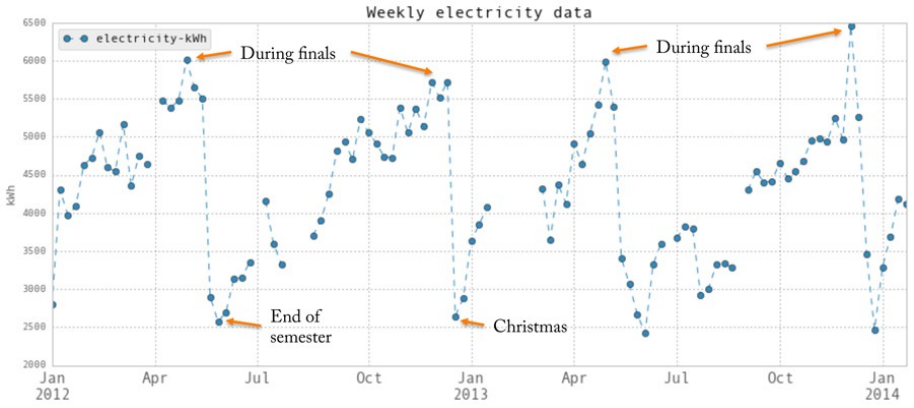


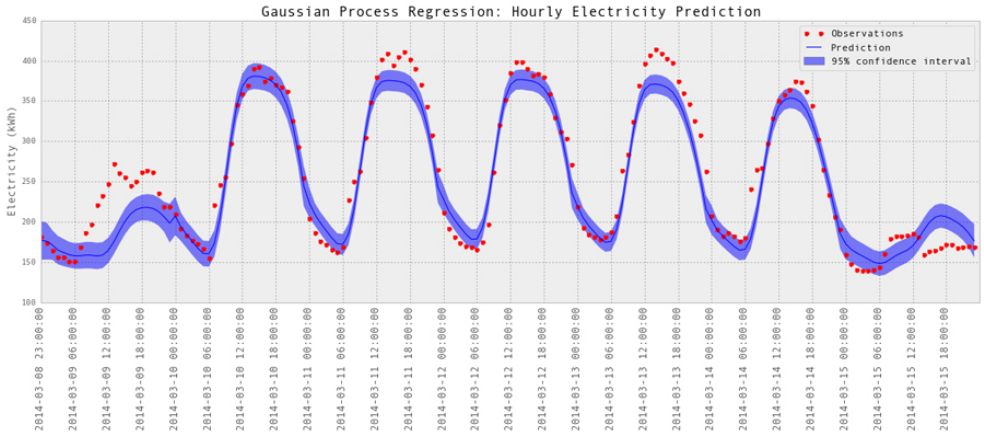


Regression in Renewable Energy Demand Forecasting



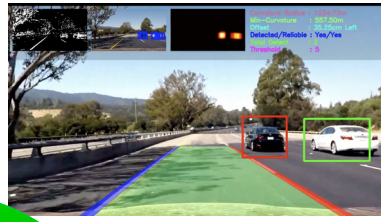
cs109-energy.github.io





Regression in Autonomous Driving

x=



A driver is demonstrating

Imitation learning/
Behaviour cloning

y=

Autonomous driving: y=f(x)

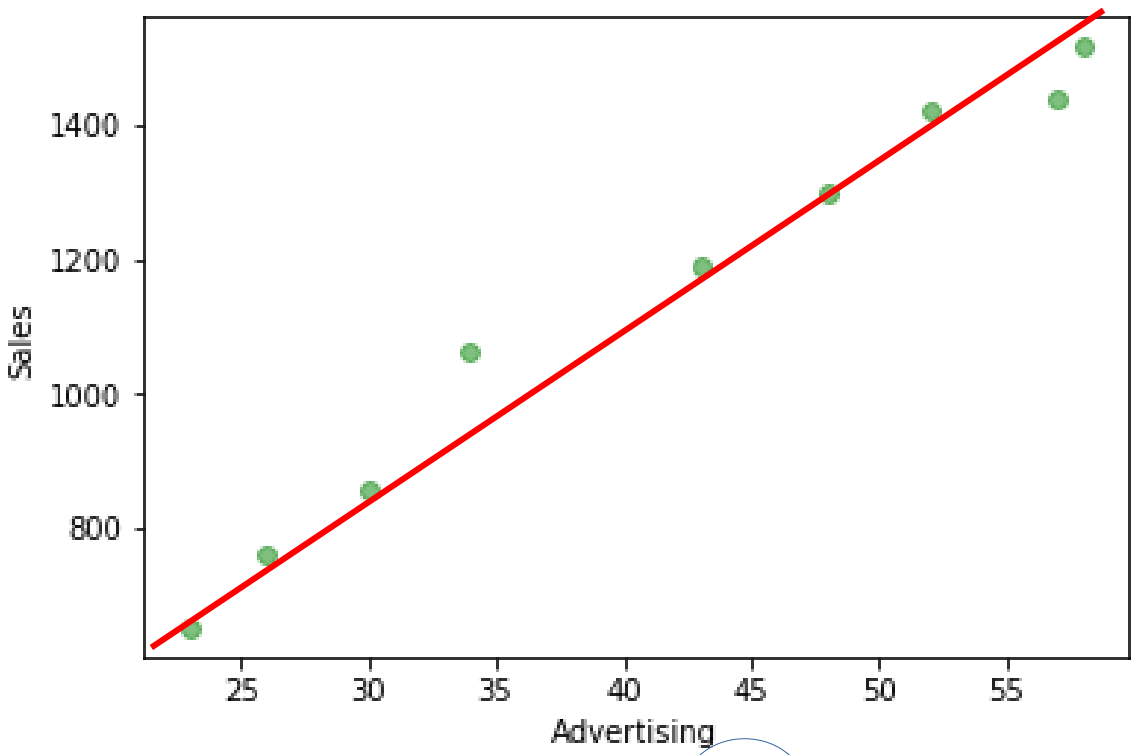
Linear Regression: example 1

- Sale prediction: advertising → sales (million EUR)

Year	Sales (Million Euro)	Advertising (Million Euro)
1	651	23
2	762	26
3	856	30
4	1,063	34
5	1,190	43
6	1,298	48
7	1,421	52
8	1,440	57
9	1,518	58

Regression

→ Predict continuous values (sales)



At year 10, if 60 Mil. EUR is invested for advertising, how much sales is predicted?

x

y

Outline

- Optimal parameters
- (Non-) linear features
- Testing & training error
- Over-fitting vs. under-fitting
- Regularization
- Cross-validation

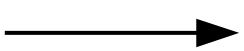
Outline

- **Optimal parameters**
- (Non-) linear features
- Testing & training error
- Over-fitting vs. under-fitting
- Regularization
- Cross-validation

Linear Regression: Example

- Sale prediction: advertising → sales (million EUR)

Year	Sales (Million Euro)	Advertising (Million Euro)
1	651	23
2	762	26
3	856	30
4	1,063	34
5	1,190	43
6	1,298	48
7	1,421	52
8	1,440	57
9	1,518	58



Regression

Predict continuous values (sales)

Training set

Sales (**y**)	Advertising (**x**)
651	23
762	26
856	30
1063	34
1190	43
1298	48
1421	52
1440	57
1518	58

n

Notation:

n=9: number of training examples

x: input variables/features

y: output variables/features

Linear Regression: Example

- Sale prediction: **advertising** → **sales** (million EUR)

Notation:

n=9: number of training examples

x: input variables/features

y: output variables/features

$x \in \Re^d$: x has d input variables/features

Data point 1 $\{x_1, y_1\} = \{23,651\}$

Data point 2 $\{x_2, y_2\} = \{26,762\}$

...

$\{x_i, y_i\}$ denotes data point *i (input i, output i)*

Training set

Sales (**y**)	Advertising (**x**)
651	23
762	26
856	30
1063	34
1190	43
1298	48
1421	52
1440	57
1518	58

Linear Regression: Example 2

Example of **energy demand prediction**

	Wind speed	People inside building	Energy requirement	
x_1	100	2	5	y_1
x_2	50	42	25	y_2
x_3	45	31	22	y_3
x_4	60	35	18	y_4

(from Nando de Freitas's lecture)

- regress: input = {wind-speed, #people}, output={energy-requirement}

n=4: number of training examples

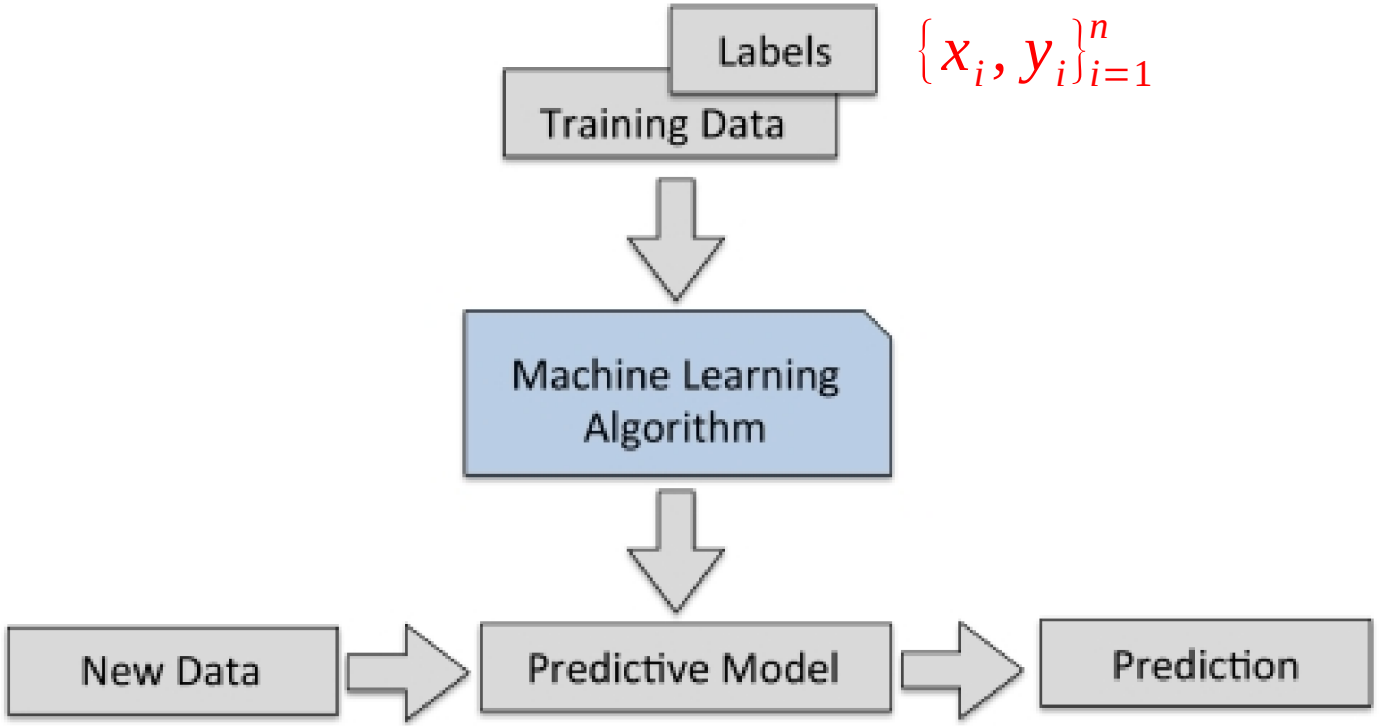
Input: has two variables (d=2)

x_{i1} = wind-speed x_{i2} = #people

e.g. $x_1 = [100, 2]$ where $x_{11} = 100, x_{12} = 2$

Denote: x_{ij} means variable/dimension j of the input/data point i

Learning Paradigm



Model Representation

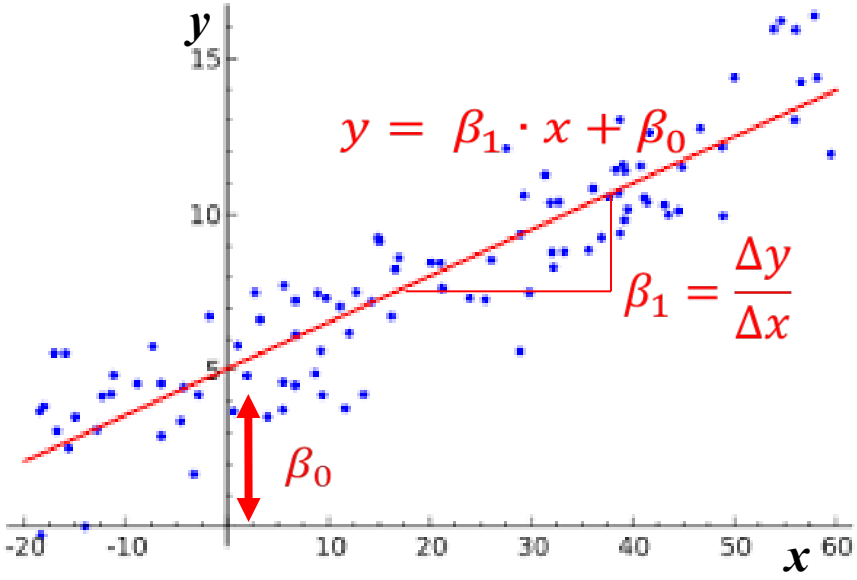
- How to represent hypothesis $\boldsymbol{y} = f(\boldsymbol{x})$
- Linear regression is a simple approach to supervised learning.
- It assumes that the relationship between $\boldsymbol{y}$ and $\boldsymbol{x}$ is **linear**

$$y_i = \beta_0 + \beta_1 x_{i1} + \ldots + \beta_d x_{id}$$

Variable to be predicted

Input variables

Regression coefficients/parameters to be learned/discovered



Linear Regression

$$y = \beta_0 + \beta_1 \cdot x_1 \quad \leftarrow x \in \Re$$

$$y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 \quad \leftarrow x \in \Re^2$$

$$y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \beta_3 \cdot x_3 \quad \leftarrow x \in \Re^3$$

$$y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \cdots + \beta_d \cdot x_d \quad \leftarrow x \in \Re^d$$

- **Summation notation**
  - For convenience

$$y = \beta_0 + \sum_{i=1}^{1} \beta_i \cdot x_i$$

$$y = \beta_0 + \sum_{i=1}^{2} \beta_i \cdot x_i$$

$$y = \beta_0 + \sum_{i=1}^{3} \beta_i \cdot x_i$$

$$\boldsymbol{y = \beta_0 + \sum_{i=1}^{d} \beta_i \cdot x_i}$$

Exercise!

$$\sum_{i=1}^{2} i = 1 + 2$$

$$\sum_{i=1}^{2} x_i = x_1 + x_2$$

Linear Regression

$$y = \beta_0 + \beta_1 \cdot x_1$$
$$y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2$$
$$y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \beta_3 \cdot x_3$$
$$y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \cdots + \beta_d \cdot x_d$$

- **Matrix notation**
  - For convenience
  - It allows the equations/programs to work for any number of variables

Augment the input vector with 1

Transpose

$$\text{If } \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_d \end{bmatrix} \text{ and } \boldsymbol{x} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix} \Rightarrow y = \boldsymbol{x}^T \cdot \beta$$

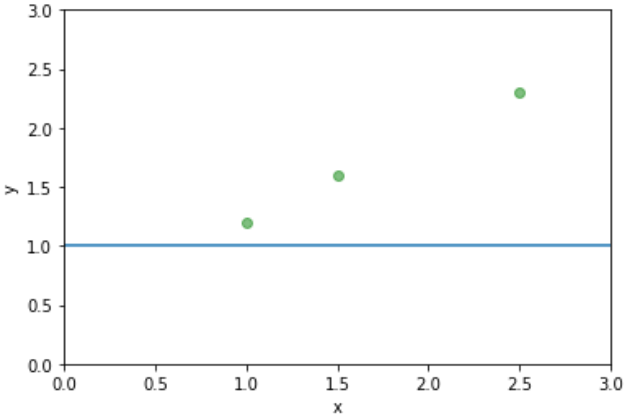
$$y = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix}^T \cdot \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_d \end{bmatrix} = [1 \quad x_1 \quad \dots \quad x_d] \cdot \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_d \end{bmatrix} = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \cdots + \beta_d \cdot x_d$$

$(dx1)^T$ x $(dx1)$ $(1x\mathbf{d})$ x $(\mathbf{d}x1)$ $(1x\mathbf{d})$ x $(\mathbf{d}x1)=1x1$

Linear Regression: Model Representation

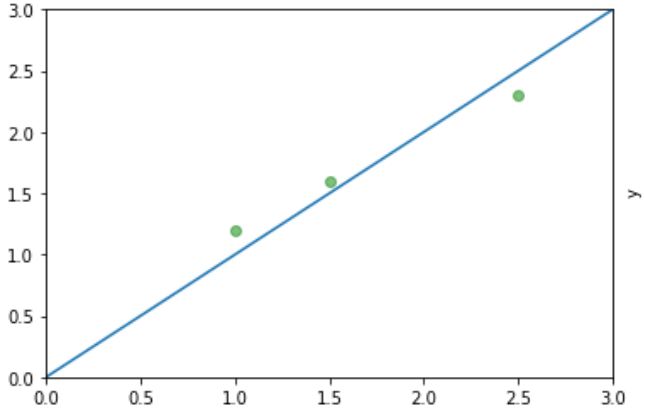
Linear model with **one variable** $y = \beta_0 + \beta_1 x$ where $x \in \Re$

$y = 1 + 0 \times x$



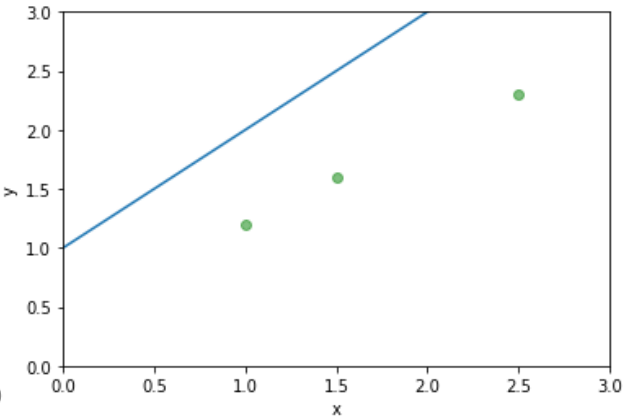
$\beta_0 = 1 ; \beta_1 = 0$

$y = 0 + 1 \times x$



$\beta_0 = 0 ; \beta_1 = 1$

$y = 1 + 1 \times x$



$\beta_0 = 1 ; \beta_1 = 1$

Linear Regression: Model Quality?

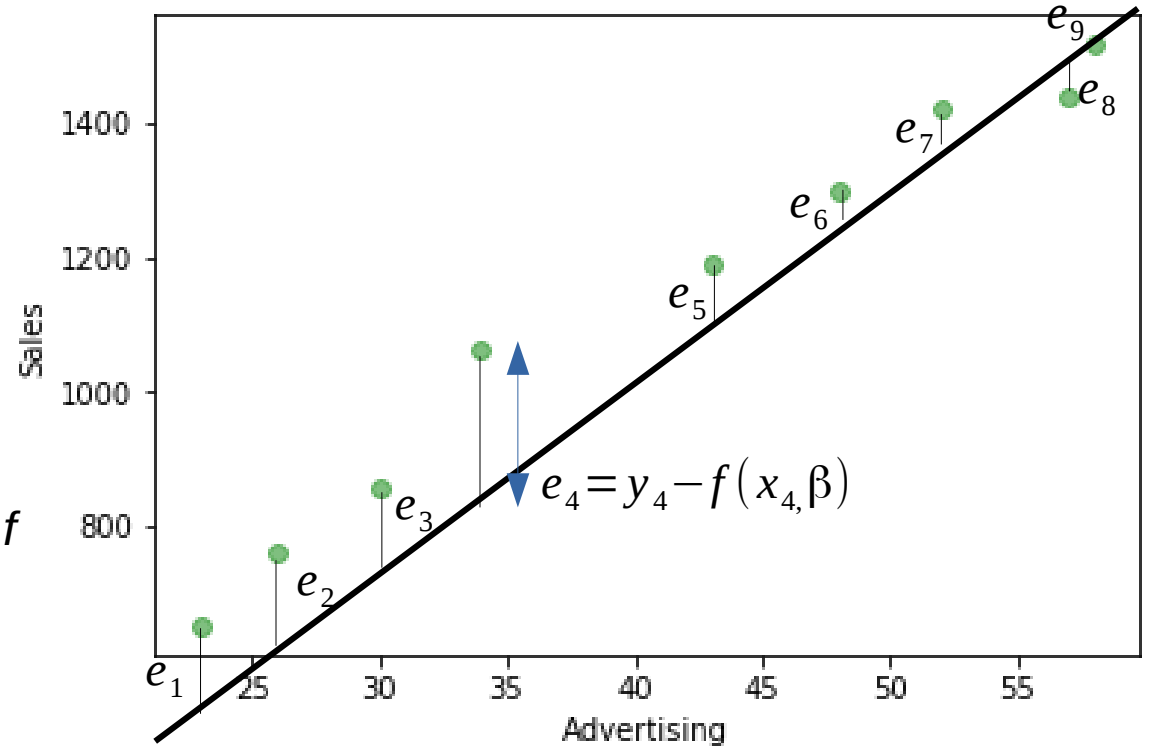
- Sale prediction: advertising → sales (million EUR)

Year	Sales (Million Euro)	Advertising (Million Euro)
1	651	23
2	762	26
3	856	30
4	1,063	34
5	1,190	43
6	1,298	48
7	1,421	52
8	1,440	57
9	1,518	58

Regression

→ Predict continuous values (sales)

$$f(x,\beta)=\beta_0+\beta_1\times advertising$$



Objective: Choose a hypothesis *f* *that predicts*

$$f(x,\beta)=\beta_0+\beta_1\times x$$

as close to *y* of training data *x*

$$f(x_i,\beta)\approx y_i \quad \text{or} \quad e_i=y_i-f(x_i,\beta) \text{ is close to zero}$$

Linear Regression: Model Quality?

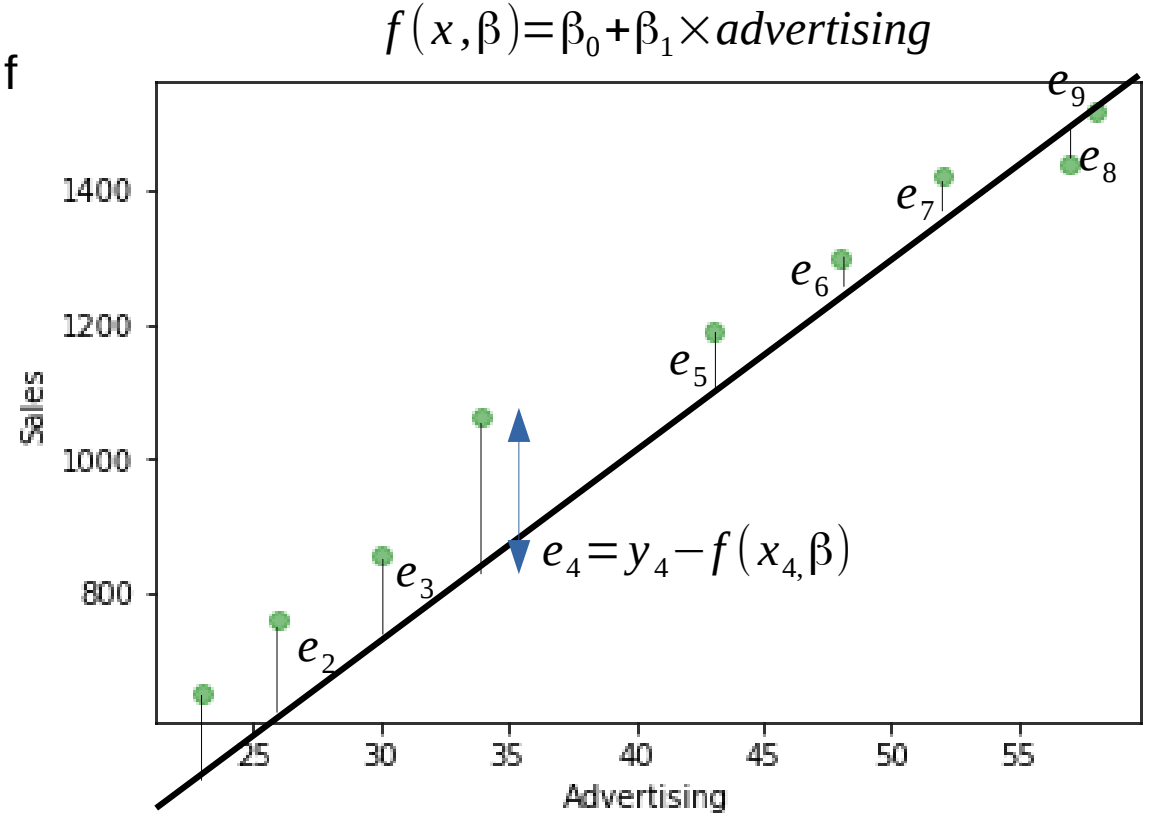
- Sale prediction: advertising → sales (million EUR)

Objective: Choose a hypothesis f *that predicts*

$$f(x,\beta)=\beta_0+\beta_1\times x$$

as close to *y* of training data *x*

$e_1, e_2, \ldots, e_n$ are close to zero



$$\underset{\beta}{minimize}\,(y_1-f(x_1,\beta))^2+\ldots+(y_n-f(x_n,\beta))^2=\underset{\beta}{minimize}\sum_{i=1}^{n}\underbrace{(y_i-f(x_i,\beta))^2}_{e_i^2}$$

Mean Square Error (MSE)

Linear Regression: Model Quality?

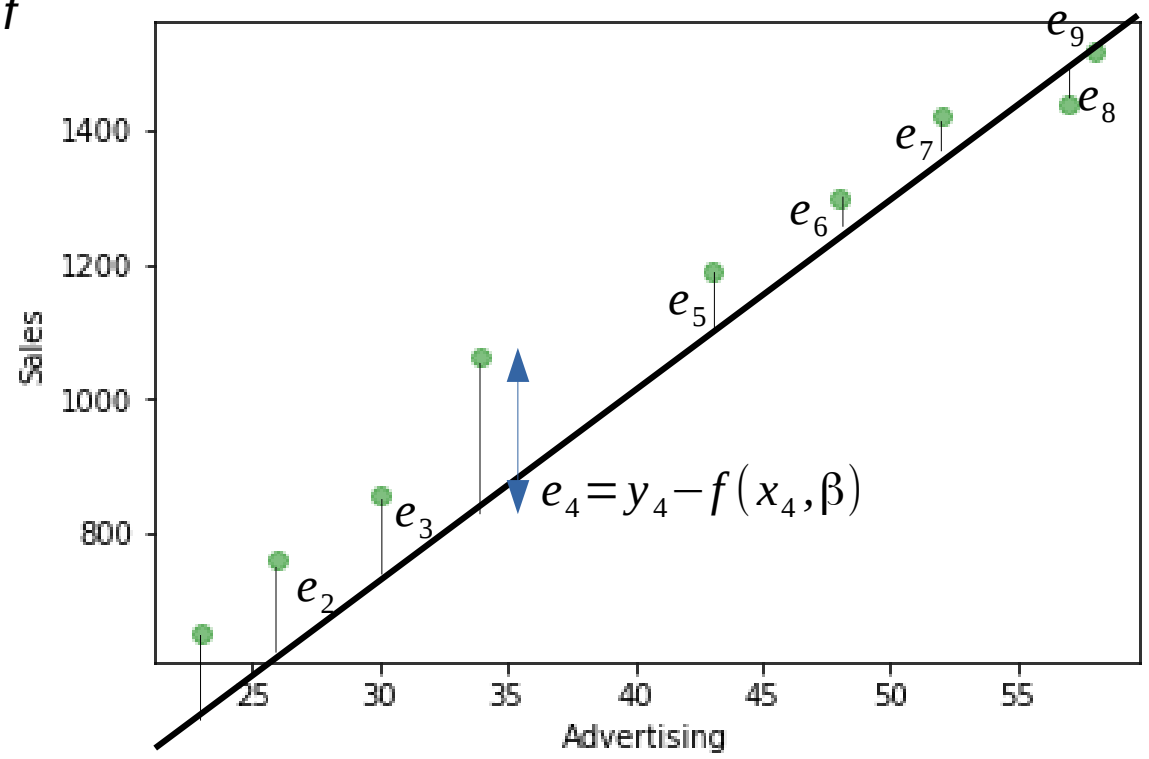
- Sale prediction: advertising → sales (million EUR)

Objective: Choose a hypothesis *f* *that predicts*

$$f(x,\beta)=\beta_0+\beta_1\times x$$

as close to *y* of training data *x*

$$f(x,\beta)=\beta_0+\beta_1\times advertising$$



$e_1, e_2, \ldots, e_n$ are close to zero

$$\underset{\beta}{minimize}\,|y_1-f(x_1,\beta)|+\ldots+|y_n-f(x_n,\beta)|=\underset{\beta}{minimize}\sum_{i=1}^{n}\underbrace{|y_i-f(x_i,\beta)|}_{|e_i|}$$

Mean Absolute Error (MAE)

Linear Regression: Model Quality?

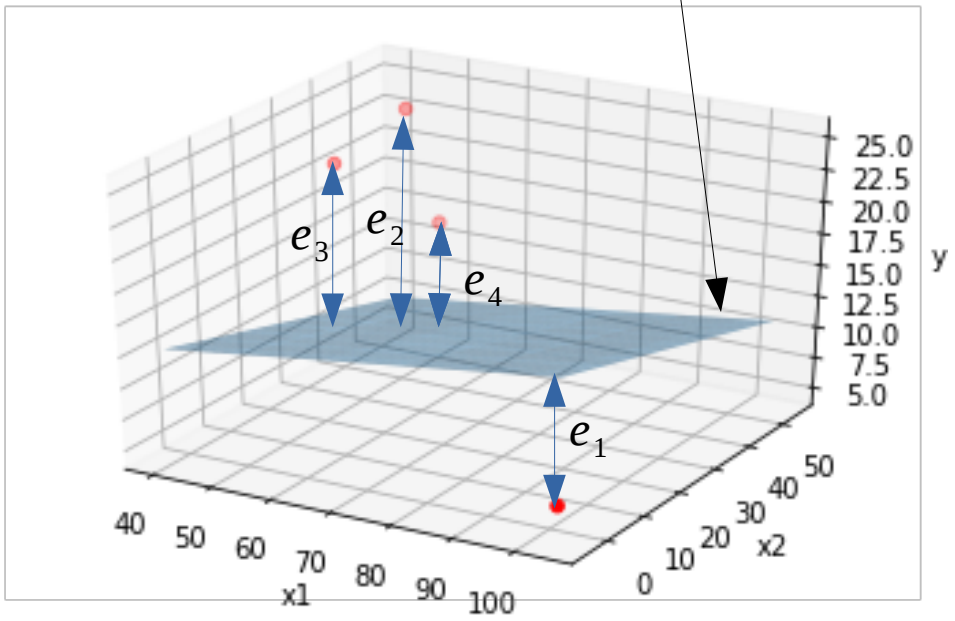
Example of **energy demand prediction**

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

(from Nando de Freitas's lecture)

- regress: input = {wind-speed, #people}, output={energy-requirement}

$$f(x,\beta)=\beta_0+\beta_1 x_1++\beta_2 x_2$$



Objective: Choose a hypothesis f *that predicts*

$$f(x,\beta)=\beta_0+\beta_1 x_1+\beta_2 x_2$$

as close to *y* of training data *x*

$$\underset{\beta}{minimize} \sum_{i=1}^{n} \underbrace{(y_i - f(x_i,\beta))^2}_{e_i^2}$$

$$\beta_0=10;\beta_1=0.06;\beta_2=-0.1$$

Linear Regression: Model Quality?

Example of **energy demand prediction**

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

(from Nando de Freitas's lecture)

- regress: input = {wind-speed, #people}, output={energy-requirement}

Example: $\beta=[1, 0.5, -20]$ $\xrightarrow{\text{Linear model}}$ $f(x_i,\beta)=1+0.5\,x_{i1}-20\,x_{i2}$

Predicted output: $f(x_1,\beta)=1+0.5\,x_{11}-20\,x_{12}=1+100*0.5-20*2=11$

The error: $e_1=y_1-f(x_1,\beta)=5-11=-6$

Absolute error $|e_1|=6$

Square error $e_1^2=36$

Linear Regression: Objective Functions

- Two cost/objective functions:

Mean Square Error (MSE): $\underset{\beta}{minimize} \sum_{i=1}^{n} (y_i - f(x_i, \beta))^2$

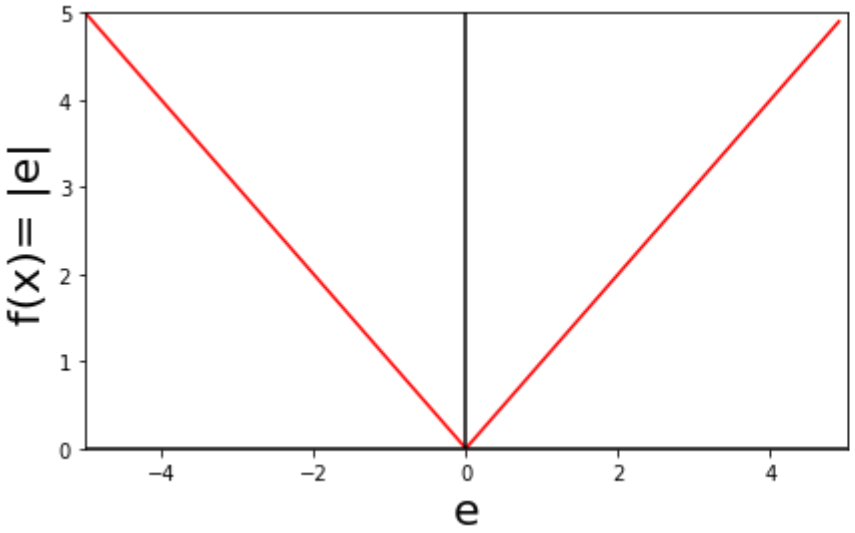
Mean Absolute Error (MAE): $\underset{\beta}{minimize} \sum_{i=1}^{n} |y_i - f(x_i, \beta)|$

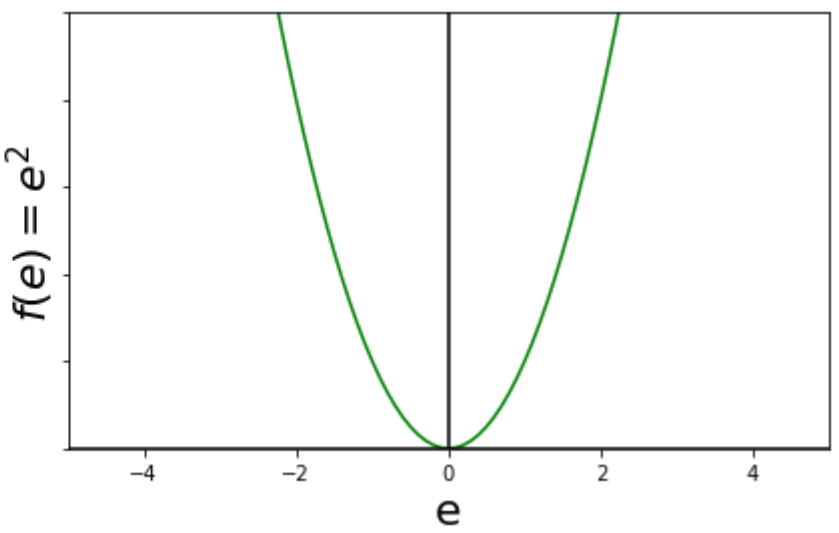
  - MAE "*is not differentiable, second derivative is zero everywhere and not defined in the point zero.*"
  - *"MAE is widely used in finance, where $10 error is usually exactly two times worse than $5 error. On the other hand, MSE metric thinks that $10 error is four times worse than $5 error. MAE is easier to justify than MSE."*
  - MSE is mathematically well-behaved
  - MSE: "If we make a single very bad prediction, the squaring will make the error even worse and it may skew the metric towards overestimating the model's badness."

https://towardsdatascience.com/

Linear Regression: Cost Functions

Mean Absolute Error (MAE) **vs.** **Mean Square Error (MSE)**





- **MSE is mathematically well-behaved**

We choose **MSE** as a measure of the quality of the fitting function!

Linear Regression: Problem Setting

- Given a training dataset of ***n* instances** of input-output

$$\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$$

- **Notation**:
  - Input vector: $x \in R^d$ $\longrightarrow$ $x = [100, 2, 0.3, 1, 4.3, 90, 10]$ (*d* entries)
  - Input dimensions: *d*
  - Output variable: $y \in R$
  - Parameters: $\beta \in R^{d+1}$
  - Linear model: $y = \beta_0 + \beta x_1 + \ldots + \beta_d x_d = \beta^T \bar{x}$ $\longrightarrow$ $\bar{x} = \begin{bmatrix} 1 \\ x \end{bmatrix} = \begin{bmatrix} 1 \\ x_{i1} \\ x_{i2} \\ \vdots \\ x_{id} \end{bmatrix}$
  - Subscript index: *i*th instance/data point x_i, y_i
  - Second subscript index: *j*th entry/variable/dimension of x_{ij}

$\longrightarrow$ x_{ij} : *j*th dimension of data point i

Linear Regression: Problem Setting

- Given a training dataset of ***n* instances** of input-output

$$\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$$

- **Training**: find **optimal parameters** β^* of the linear function $y = f(x, \beta^*)$ such that:

$$y^{(i)} \approx f(x^{(i)}, \beta^*), \forall i \in [1,2,\ldots,n]$$

Mean Square Error:

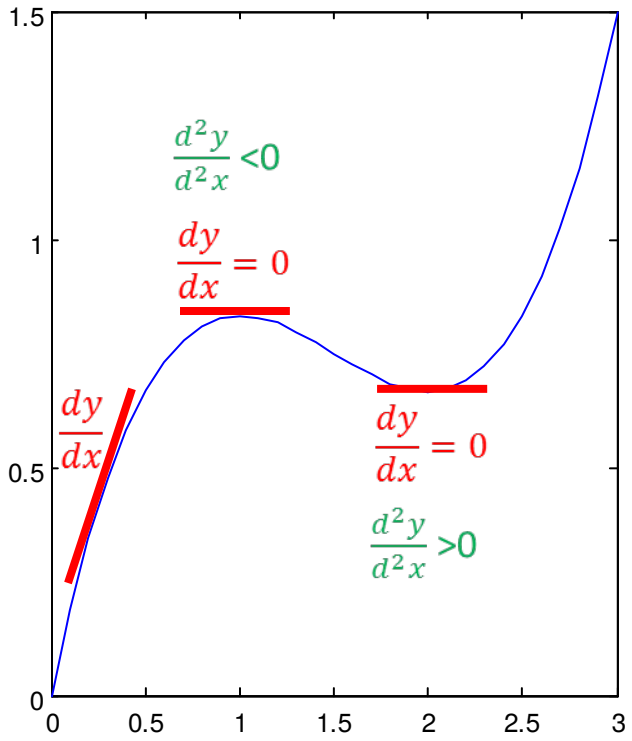
$$\underset{\beta}{minimize} \sum_{i=1}^{n} (y_i - f(x_i, \beta))^2$$

- **Testing/prediction**: given a new input $x^{(new)}$, predicted output is

$$y^{(new)} = f(x^{(new)}, \beta^*) = x^{(new)T} \beta^* = \beta_0^* + \beta_1^* x_1^{(new)} + \ldots + \beta_d^* x_d^{(new)}$$

Finding the maxima/minima of a function

- Given a mathematical function $y = f(x)$, maxima/minima points are found at those positions where the derivative of the function $\frac{dy}{dx}$ is equal to 0
  - The derivative of a function gives you it's "gradient" or "slope"
  - Therefore, if $\frac{dy}{dx}$=0, the slope is 0 is not increasing or decreasing its value anymore



- **STEPS:**
  - Given y=f(x), we calculate the derivative $\frac{dy}{dx}$
  - Set $\frac{dy}{dx}$=0 and solve for x

Which of these points are maxima and which are minima?

- Calculate the second derivative

If <0 ▭ local maximum.
If >0 ▭ local minimum.

Finding the maxima/minima of a function

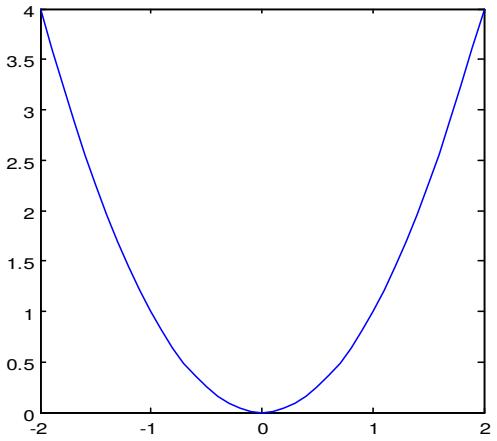
- **STEPS:**
  - Given y=f(x), we calculate the derivative $\frac{dy}{dx}$
  - Set $\frac{dy}{dx}$=0 and solve for x

Example 1:

$$y = x^2$$

$$\frac{dy}{dx} = 2x$$

$$\frac{dy}{dx} = 2x = 0 \Rightarrow x = \frac{0}{2} = 0$$



Is it a minima or a maxima?

$$\frac{d^2y}{d^2x} = 2 \quad >0 \Rightarrow \text{local minimum}$$

Finding the maxima/minima of a function

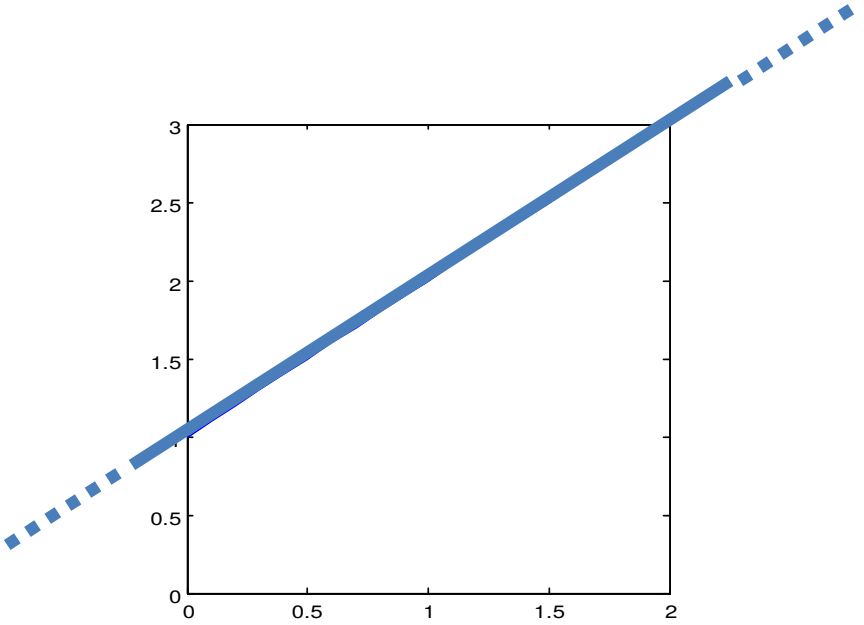
- **STEPS:**
  - Given y=f(x), we calculate the derivative $\frac{dy}{dx}$
  - Set $\frac{dy}{dx}$=0 and solve for x

Example 2:

$$y = x + 1$$

$$\frac{dy}{dx} = 1$$

$$\frac{dy}{dx} = 1 \neq 0$$



No minimum or maximum ⇨ It is always growing and decreasing

Linear Regression: Optimal Parameters

- Given training data $D = \{(x_i, y_i)\}_{i=1}^{n}$ we define the *least squares* cost (or "loss")

$$L^{ls}(\beta) = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

Goal : Minimise the total error → **Find the β^* that minimises the loss function $L^{ls}(\beta)$**

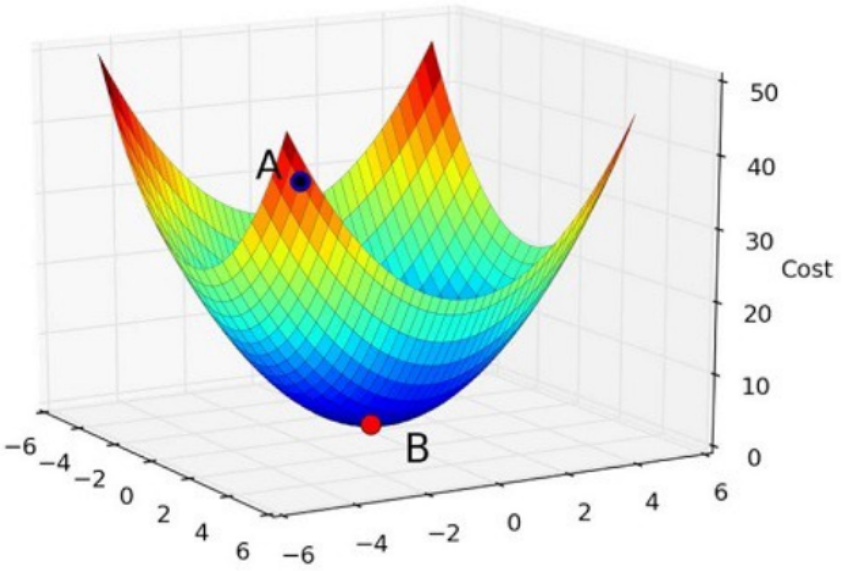
$$\frac{\partial L^{ls}}{\partial \beta} = 0$$

Do I need to calculate the second derivative to know if it is a maxima or a minima?

No! It will always be a minima!

It is a error function:

- **It will be 0 in the very best case**
- **It will grow forever for a very poor fitting**



Linear Regression: Optimal Parameters

$$y = \beta_0 + \beta_1 \cdot x_1$$

$$y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2$$

$$y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \beta_3 \cdot x_3$$

$$y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \cdots + \beta_d \cdot x_d$$

- **Matrix notation**
  - For convenience
  - It allows the equations/programs to work for any number of variables

Transpose

$$\text{If } \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_d \end{bmatrix} \text{ and } \bar{x} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix} \Rightarrow y = \bar{x}^T \cdot \beta$$

$$y = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix}^T \cdot \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_d \end{bmatrix} = [1 \quad x_1 \quad \dots \quad x_d] \cdot \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_d \end{bmatrix} = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \cdots + \beta_d \cdot x_d$$

$(dx1)^T$ x $(dx1)$ $(1x\mathbf{d})$ x $(\mathbf{d}x1)$ $(1x\mathbf{d})$ x $(\mathbf{d}x1)=1x1$

Linear Regression: Optimal Parameters

Coming back to Matrix notation:

$$f(x_i) = \beta_0 + \sum_{j=1}^{d} \beta_j x_{ij} = \begin{pmatrix} 1, & x_{i1}, & x_{i2}, & \dots & , x_{id} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_d \end{pmatrix} = \bar{x}_i^\top \beta$$

I cannot put that matrix to the square since the dimensions will not match
Instead I multiply by the transpose

- Rewrite the sum of squares as:

$$L^{\text{ls}}(\beta) = \sum_{i=1}^{n} (y_i - \bar{x}_i^\top \beta)^2 = (Y - X\beta)^\top (Y - X\beta) = \|Y - X\beta\|^2$$

where $\|a\|^2 = \sum_{i=1}^{n} a_i^2$ for a vector $a \in \mathbb{R}^n$

with $X \in \mathbb{R}^{n \times d+1}, Y \in \mathbb{R}^n$:

$$X = \begin{pmatrix} \bar{x}_1^\top \\ \vdots \\ \bar{x}_n^\top \end{pmatrix} = \begin{pmatrix} 1 & x_{1,1} & x_{1,2} & \cdots & x_{1,d} \\ \vdots & & & & \vdots \\ 1 & x_{n,1} & x_{n,2} & \cdots & x_{n,d} \end{pmatrix}, \quad Y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

Optimal Parameters

Minimize:

$$L^{\text{ls}}(\beta) = \sum_{i=1}^{n}(y_i - \bar{x}_i^{\top}\beta)^2 = (Y - X\beta)^{\top}(Y - X\beta) = \|Y - X\beta\|^2 \quad \xrightarrow{\text{find } \beta} \quad \frac{\partial L^{ls}}{\partial \beta} = 0$$

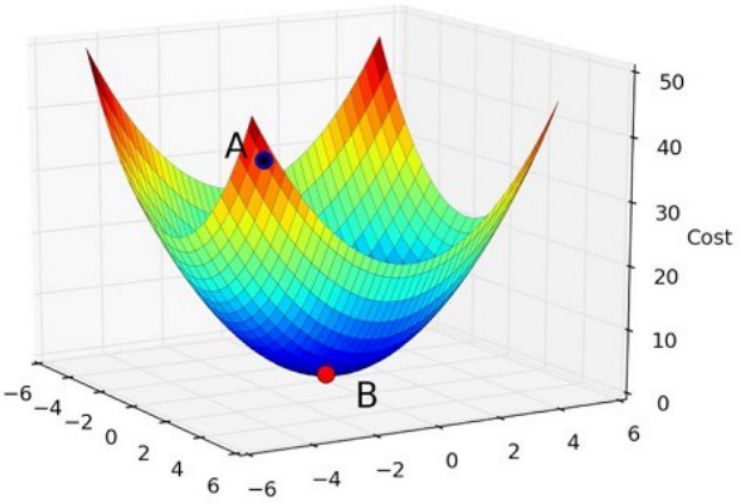
$$\frac{\partial L^{ls}}{\partial \beta} = -2(Y - X\beta)^T X = 0 \longrightarrow (-2(Y - X\beta)^T X)^T = 0^T$$

$$\Longleftrightarrow \quad 0^T = (-2(Y - X\beta)^T X)^T = -2X^T(Y - X\beta)$$

$$\Longleftrightarrow \quad 0^T = -2X^T Y + 2X^T X\beta$$

$$\Longleftrightarrow \quad 2X^T Y = 2X^T X\beta$$

$$\Longleftrightarrow \quad (X^T X)^{-1} X^T Y = \beta$$



Optimal parameters:

$$\mathbf{0}_d^{\top} = \frac{\partial L^{\text{ls}}(\beta)}{\partial \beta} = -2(Y - X\beta)^{\top}X \quad \Longleftrightarrow \quad \mathbf{0}_d = X^{\top}X\beta - X^{\top}Y$$

$$\hat{\beta}^{\text{ls}} = (X^{\top}X)^{-1}X^{\top}Y$$

Example 1: 1 Features (1D space)

- **Fahrenheit to Celsius Conversion**

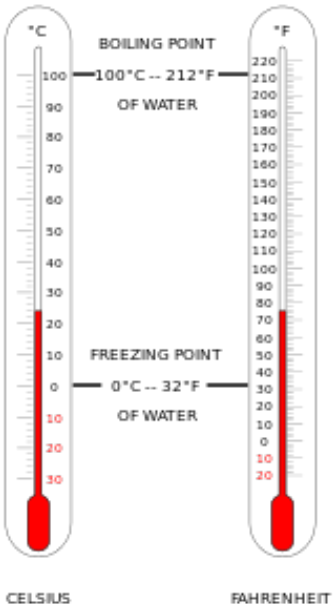
$$T_{(°F)} = T_{(°C)} \times 9/5 + 32$$

- Let's imagine we do not know or we do not remember the formula
- Instead we are going to deduct it by observation

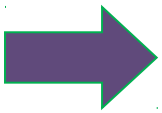
- We have 2 thermometers

$$T(°F) \rightarrow y = \beta_0 + \beta_1 x \leftarrow T(°C)$$

  - We take measurements at 3 different days
  - They are very cheap, so they are not really precise:



25	77
30	86
20	68



24	77
30	88
21	67

Example 1: 1 Features (1D space)

x	y
24	77
30	88
21	67

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 24 \\ 30 \\ 21 \end{bmatrix} \qquad Y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 77 \\ 88 \\ 67 \end{bmatrix}$$

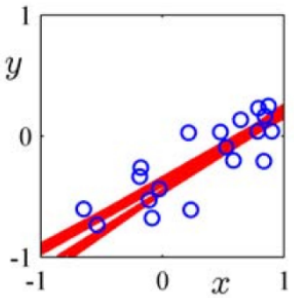
- Regression: input ={T in Celsius}, output={T in farenheit }
  , where . The number of data points:

- **linear relationship**:
- $y = \beta_0 + \beta_1 x$ → $\beta = (\beta_0, \beta_1, .., \beta_d)^\top \in \mathbb{R}^{d+1}$
- the least-square error

$$L^{\text{ls}}(\beta) = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2$$

Bonus: Geometric Interpretation →

fitting a line in 2D



Bishop, Pattern Recognition and Machine Learning

Example 1: 1 Features (1D space)

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 24 \\ 30 \\ 21 \end{bmatrix} \qquad Y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 77 \\ 88 \\ 67 \end{bmatrix} \qquad \Rightarrow \qquad \beta = (\beta_0, \beta_1, .., \beta_d)^\top \in \mathbb{R}^{d+1}$$

- Augment input vector with a 1 in front (**adding bias term**):

$$\bar{x}_i = (1, x_i) = (1, x_{i1}, .., x_{id})^\top \in \mathbb{R}^{d+1};$$

$$X = \begin{pmatrix} \bar{x}_1^\top \\ \vdots \\ \bar{x}_n^\top \end{pmatrix} = \begin{pmatrix} 1 & x_{1,1} & x_{1,2} & \cdots & x_{1,d} \\ \vdots & & & & \vdots \\ 1 & x_{n,1} & x_{n,2} & \cdots & x_{n,d} \end{pmatrix}, \quad Y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

$$X = \begin{bmatrix} 1 & 24 \\ 1 & 30 \\ 1 & 21 \end{bmatrix} \qquad Y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 77 \\ 88 \\ 67 \end{bmatrix}$$

Computing $\beta^* = (X^T X)^{-1} X^T Y$

$$X^T = \begin{bmatrix} 1 & 1 & 1 \\ 24 & 30 & 21 \end{bmatrix}$$

$$X^T \cdot X = \begin{bmatrix} 1 & 1 & 1 \\ 24 & 30 & 21 \end{bmatrix} \begin{bmatrix} 1 & 24 \\ 1 & 30 \\ 1 & 21 \end{bmatrix} = \begin{bmatrix} 1 \cdot 1 + 1 \cdot 1 + 1 \cdot 1 & 1 \cdot 24 + 1 \cdot 30 + 1 \cdot 21 \\ 24 \cdot 1 + 30 \cdot 1 + 21 \cdot 1 & 24 \cdot 24 + 30 \cdot 30 + 30 \cdot 30 \end{bmatrix} = \begin{bmatrix} 3 & 75 \\ 75 & 1917 \end{bmatrix}$$

$$\det(X^T \cdot X) = 3 \cdot 1917 - 75 \cdot 75 = 126$$

$$(X^T \cdot X)^{-1} = \left(\begin{bmatrix} 3 & 75 \\ 75 & 1917 \end{bmatrix} \right)^{-1} = \frac{1}{126} \begin{bmatrix} 1917 & -75 \\ -75 & 3 \end{bmatrix} = \begin{bmatrix} 15.214 & -0.595 \\ -0.595 & 0.0238 \end{bmatrix}$$

Example 1: 1 Features (1D space)

Computing $\boxed{\beta^* = (X^T X)^{-1} X^T Y}$

$$X^T = \begin{bmatrix} 1 & 1 & 1 \\ 24 & 30 & 21 \end{bmatrix} \qquad (X^T \cdot X)^{-1} = \begin{bmatrix} 15.214 & -0.595 \\ -0.595 & 0.0238 \end{bmatrix} \qquad Y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 77 \\ 88 \\ 67 \end{bmatrix}$$

$$\beta^* = (X^T X)^{-1} X^T Y = \begin{bmatrix} 15.214 & -0.595 \\ -0.595 & 0.0238 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 24 & 30 & 21 \end{bmatrix} \begin{bmatrix} 77 \\ 88 \\ 67 \end{bmatrix} =$$

$$= \begin{bmatrix} 15.214 \cdot 1 - 0.595 \cdot 24 & 15.214 \cdot 1 - 0.595 \cdot 30 & 15.214 \cdot 1 - 0.595 \cdot 21 \\ -0.595 \cdot 1 + 0.0238 \cdot 24 & -0.595 \cdot 1 + 0.0238 \cdot 30 & -0.595 \cdot 1 + 0.0238 \cdot 21 \end{bmatrix} \begin{bmatrix} 77 \\ 88 \\ 67 \end{bmatrix} =$$

$$= \begin{bmatrix} 0.929 & -2.643 & 2.714 \\ -0.0238 & 0.119 & -0.095 \end{bmatrix} \begin{bmatrix} 77 \\ 88 \\ 67 \end{bmatrix} = \begin{bmatrix} 0.9286 \cdot 77 - 2.6429 \cdot 88 + 2.7143 \cdot 67 \\ -0.0238 \cdot 77 + 0.119 \cdot 88 - 0.095 \cdot 67 \end{bmatrix}$$

$$= \begin{bmatrix} 22.78 \\ 2.26 \end{bmatrix}$$

$$\boldsymbol{\beta}^* = \begin{bmatrix} 22.78 \\ 2.26 \end{bmatrix} = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}$$

$$y_i = \beta_0 + \beta_1 \cdot x_1$$

$$y_i = 22.78 + 2.26 \cdot x_1 \quad \Rightarrow \quad T_{(°F)} = T_{(°C)} \times 1.8 + 32$$

Example 1: 1 Features (1D space)

- **The model is as good as your data**

24	77
30	88
21	67

25	77
30	86
20	68

$$y_i = 32 + 1.8\, x_i$$

$$T_{(°F)} = T_{(°C)} \times 1.8 + 32$$

- **The more data, the better**

24	77
30	88
21	67
17	63
32	87
26	78

$$y_i = 32.76 + 1.75\, x_i$$

$$T_{(°F)} = T_{(°C)} \times 1.8 + 32$$

Example: fitting a line

- a dataset: $\{x_1 = 1, y_1 = 2\}, \{x_2 = 2, y_2 = 4\}$ (generated by function $y = 2x$)

  hence $X =$ [redacted] so $X^\top X =$ [redacted]

  compute the inversion

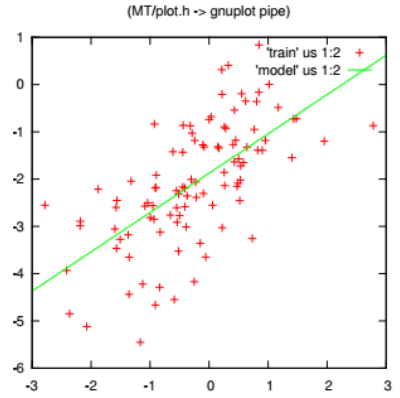
  $A^{-1} = (X^\top X)^{-1} =$ [redacted]

  (see our lecture on Linear algebra)

  the output vector is $Y =$ [redacted]

- assume our parametric linear model is $y = ax + b$, so $\beta = \begin{pmatrix} b \\ a \end{pmatrix}$
- optimal parameters via linear regression

$$\beta = (X^\top X)^{-1} X^\top Y = \text{[redacted]}$$



Example 3: 2 Features (2D space)

Example of **energy demand prediction**

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

x y

(from Nando de Freitas's lecture)

- regress: input = {wind-speed, #people}, output={energy-requirement}

- $x_i \in \mathbb{R}^d$, where $d = 2$. The number of data points: $n = 4$
- data $\{x_{1:4}, y_{1:4}\} = \{x_i, y_i\}_{i=1}^4$:

$$X = \begin{bmatrix} \bar{x}_1^T \\ \bar{x}_2^T \\ \bar{x}_3^T \\ \bar{x}_4^T \end{bmatrix} = \begin{bmatrix} 1 & x_{11} & x_{12} \\ 1 & x_{21} & x_{22} \\ 1 & x_{31} & x_{32} \\ 1 & x_{41} & x_{42} \end{bmatrix} = \begin{bmatrix} 1 & 100 & 2 \\ 1 & 50 & 42 \\ 1 & 45 & 31 \\ 1 & 60 & 35 \end{bmatrix}$$

$$x_1 = [100, 2]^\top, y_1 = 5$$
$$x_2 = [50, 42]^\top, y_2 = 25$$
$$x_3 = [45, 31]^\top, y_3 = 22$$
$$x_4 = [60, 35]^\top, y_4 = 18$$

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 5 \\ 25 \\ 22 \\ 18 \end{bmatrix}$$

- inputs/features? $x_1 = [100, 2]^\top$, so $x_{11} = 100$ and $x_{12} = 2$

Example 3: 2 Features (2D space)

- **linear relationship**: $y_i = \beta_0 + \beta_1 \cdot x_{i1} + \beta_2 \cdot x_{i2}$

$$\beta = (\beta_0, \beta_1, .., \beta_d)^\top \in \mathbb{R}^{d+1}$$

- Augment input vector with a 1 in front (**adding bias term**):
  $\bar{x}_i = (1, x_i) = (1, x_{i1}, .., x_{id})^\top \in \mathbb{R}^{d+1}$;
  ex. (energy prediction): $x_i = [100, 2]$ becomes $\bar{x}_i = [1, 100, 2]^\top$

$$X = \begin{pmatrix} \bar{x}_1^\top \\ \vdots \\ \bar{x}_n^\top \end{pmatrix} = \begin{pmatrix} 1 & x_{1,1} & x_{1,2} & \cdots & x_{1,d} \\ \vdots & & & & \vdots \\ 1 & x_{n,1} & x_{n,2} & \cdots & x_{n,d} \end{pmatrix}, \quad Y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

- write $X = \begin{pmatrix} 1 & 100 & 2 \\ 1 & 50 & 22 \\ 1 & 45 & 31 \\ 1 & 60 & 35 \end{pmatrix}, Y = \begin{pmatrix} 5 \\ 25 \\ 22 \\ 18 \end{pmatrix}$

Example 3: 2 Features (2D space)

- write $X = \begin{pmatrix} 1 & 100 & 2 \\ 1 & 50 & 22 \\ 1 & 45 & 31 \\ 1 & 60 & 35 \end{pmatrix}, Y = \begin{pmatrix} 5 \\ 25 \\ 22 \\ 18 \end{pmatrix}$

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

- check $Y - X\beta = \begin{pmatrix} 5 \\ 25 \\ 22 \\ 18 \end{pmatrix} - \begin{pmatrix} 1 & 100 & 2 \\ 1 & 50 & 22 \\ 1 & 45 & 31 \\ 1 & 60 & 35 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{pmatrix} = \begin{pmatrix} y_1 - \bar{x}_1^\top \beta \\ y_2 - \bar{x}_2^\top \beta \\ y_3 - \bar{x}_3^\top \beta \\ y_4 - \bar{x}_4^\top \beta \end{pmatrix}$ so

$$L^{\text{ls}}(\beta) = (Y - X\beta)^\top (Y - X\beta) = \begin{pmatrix} y_1 - \bar{x}_1^\top \beta & y_2 - \bar{x}_2^\top \beta & y_3 - \bar{x}_3^\top \beta & y_4 - \bar{x}_4^\top \beta \end{pmatrix} \begin{pmatrix} y_1 - \bar{x}_1^\top \beta \\ y_2 - \bar{x}_2^\top \beta \\ y_3 - \bar{x}_3^\top \beta \\ y_4 - \bar{x}_4^\top \beta \end{pmatrix}$$

If the estimated will be:

- If $\beta = [1, 0.5, -20]^\top$, then

$$f(x_i) = \beta_0 + \sum_{j=1}^{d} \beta_j x_{ij} = \begin{pmatrix} 1, & x_{i1}, & x_{i2}, & \dots & , x_{id} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_d \end{pmatrix} = \bar{x}_i^\top \beta$$

$$f(x_1) = \begin{pmatrix} 1 & 100 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 0.5 \\ -1 \end{pmatrix} = 1 + 100 \times 0.5 - 2 \times 20 = 11$$

yields an error $|y_1 - f(x_1)| = |5 - 11| = 6$

But this are not the optimal coefficients !!!

Example 3: 2 Features (2D space)

- write $X = \begin{pmatrix} 1 & 100 & 2 \\ 1 & 50 & 22 \\ 1 & 45 & 31 \\ 1 & 60 & 35 \end{pmatrix}, Y = \begin{pmatrix} 5 \\ 25 \\ 22 \\ 18 \end{pmatrix}$

Computing $\boxed{\beta^* = (X^T X)^{-1} X^T Y}$

$$= \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 100 & 50 & 45 & 60 \\ 2 & 42 & 31 & 35 \end{bmatrix} \begin{bmatrix} 1 & 100 & 2 \\ 1 & 50 & 42 \\ 1 & 45 & 31 \\ 1 & 60 & 35 \end{bmatrix} \right)^{-1} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 100 & 50 & 45 & 60 \\ 2 & 42 & 31 & 35 \end{bmatrix} \begin{bmatrix} 5 \\ 25 \\ 22 \\ 18 \end{bmatrix} =$$

$$\boldsymbol{\beta}^* = \begin{bmatrix} 44.97 \\ -0.39 \\ -0.09 \end{bmatrix} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix}$$

$$y_i = \beta_0 + \beta_1 \cdot x_{i1} + \beta_2 \cdot x_{i2}$$

$$y_i = 44.97 - 0.39 \cdot x_{i1} - 0.09 \cdot x_{i2}$$

Example 3: 2 Features (2D space)

$$y_i = 44.97 - 0.39 \cdot x_{i1} - 0.09 \cdot x_{i2}$$

- **If you give me a new testing sample (not seen before), I can predict the required energy!**

$$x_t = [55 \quad 10]$$

$$y_t = 44.97 - 0.39 \cdot x_{t1} - 0.09 \cdot x_{t2} = 44.97 - 0.39 \cdot 55 - 0.09 \cdot 10$$

$$y_t = 22.62$$

Example of **energy demand prediction**

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

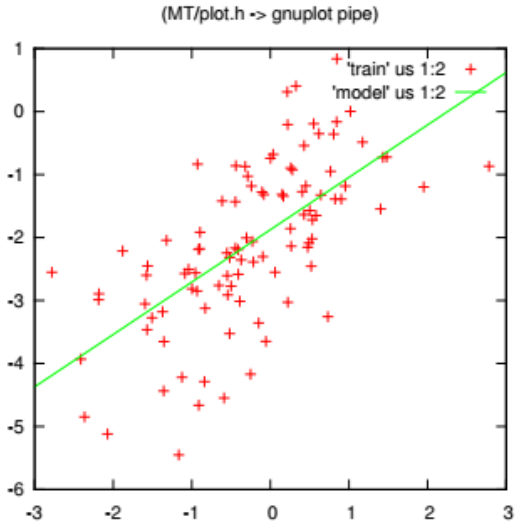
(from Nando de Freitas's lecture)

- regress: input = {wind-speed, #people}, output={energy-requirement}

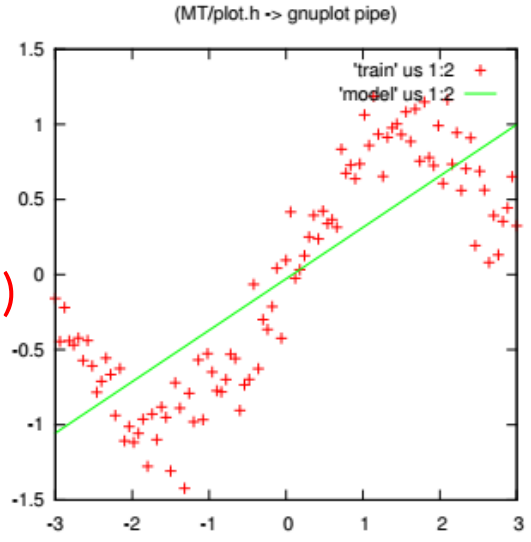
Limitations

- **Only works for predicting linear relationships**

$$y = f(x)$$



Linear space ⟶ **Success!!**



Non linear space ⟶ **Failed!!**

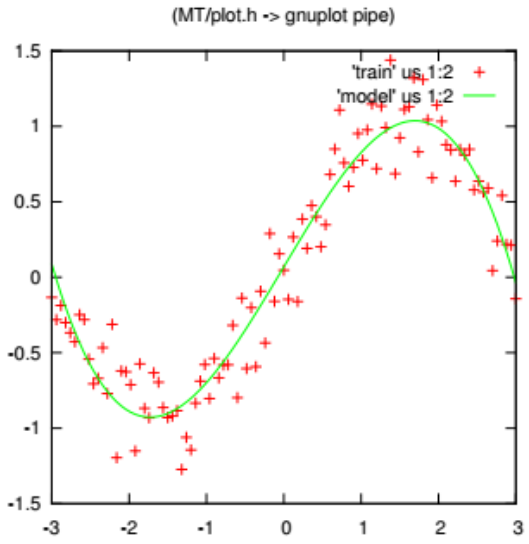
(or poor approximation)

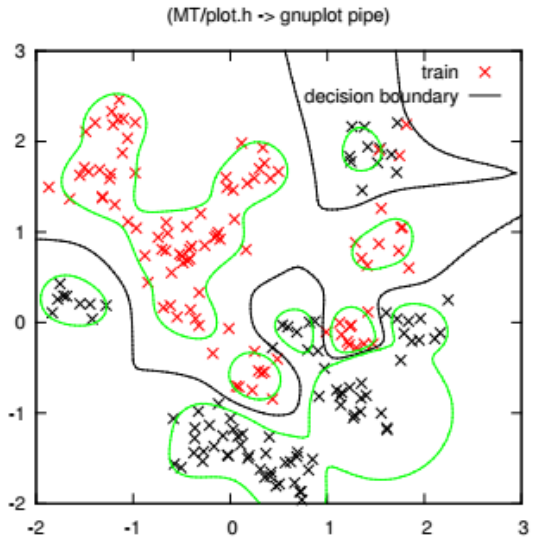
Outline

- Optimal parameters
- **(Non-) linear features**
- Testing & training error
- Over-fitting vs. under-fitting
- Regularization

Non-Linear Features

- **Can we do better?**

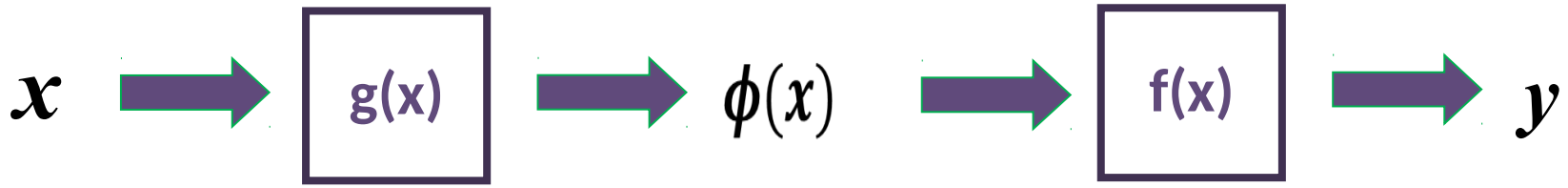




- Are these linear models? Linear in *what*?
  - Input: No.
  - Parameters, features: Yes!

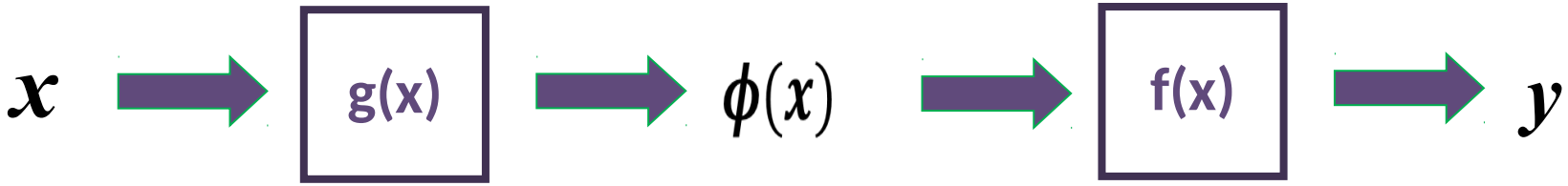
$$y = f(g(\boldsymbol{x}))$$

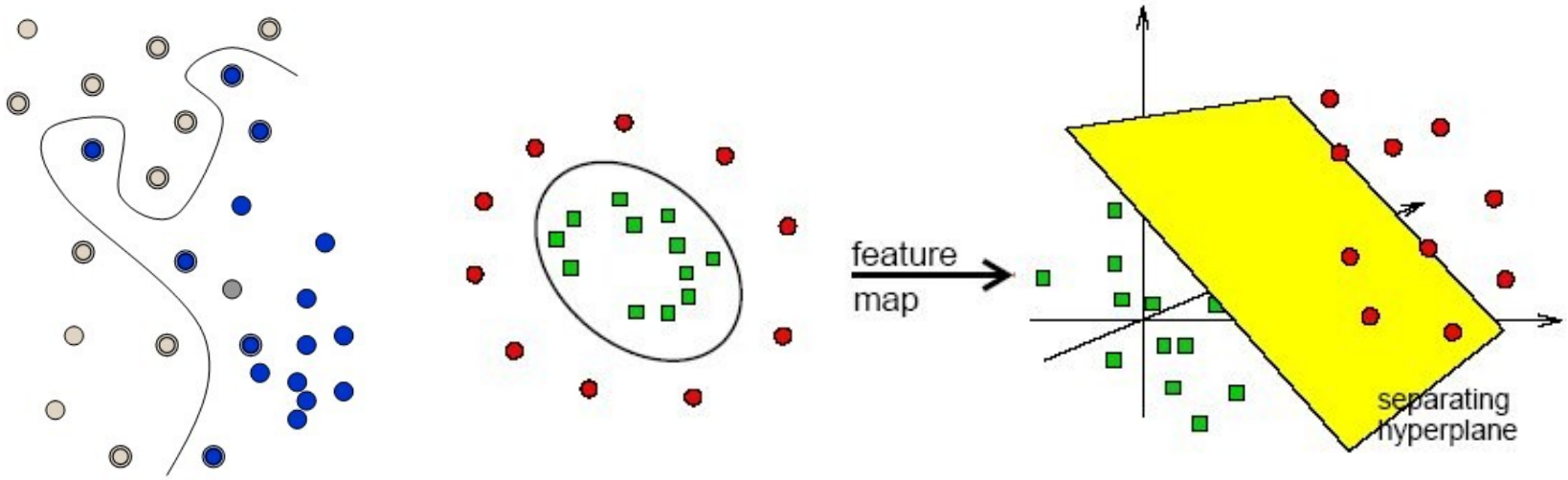
Non Linear space → **Linear space** → **Success!!**



Non Linear Features

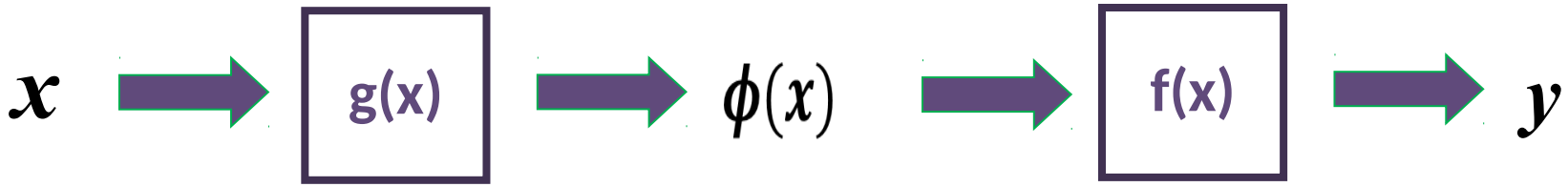
Non Linear space → **Linear space** → **Success!!**





Non Linear Features

Non Linear space → **Linear space** → **Success!!**



Linear

$$f(x_i) = \beta_0 + \sum_{j=1}^{d} \beta_j x_{ij} = (1, \quad x_{i1}, \quad x_{i2}, \quad \dots \quad , x_{id}) \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_d \end{pmatrix} = \bar{x}_i^\top \beta$$

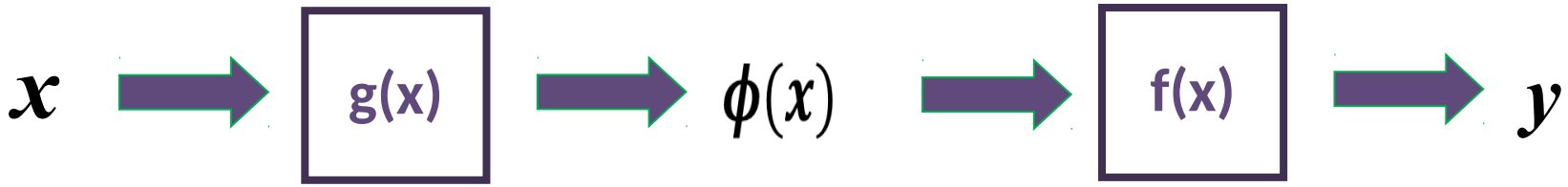
- Replace the inputs $x_i \in \mathbb{R}^d$ by some **non-linear features** $\phi(x_i) \in \mathbb{R}^k$

$$\phi : \mathbb{R}^d \mapsto \mathbb{R}^k$$

$$f(x) = \sum_{j=1}^{k} \phi_j(x)\, \beta_j = \phi(x)^\top \beta$$

Non Linear Features

Non Linear space → **Linear space** → **Success!!**



- Replace the inputs $x_i \in \mathbb{R}^d$ by some **non-linear features** $\phi(x_i) \in \mathbb{R}^k$

$$\phi : \mathbb{R}^d \mapsto \mathbb{R}^k$$

$$f(x) = \sum_{j=1}^{k} \phi_j(x)\, \beta_j = \phi(x)^\top \beta$$

e.g: $\phi : \mathbb{R} \mapsto \mathbb{R}^3$, where $\phi(x) = \begin{pmatrix} 1 \\ x \\ x^2 \end{pmatrix}$ → We receive a **quadratic model**: $f(x) = \beta^\top \phi(x) = \beta_0 + \beta_1 x + \beta_2 x^2$

- The **optimal β is the same**

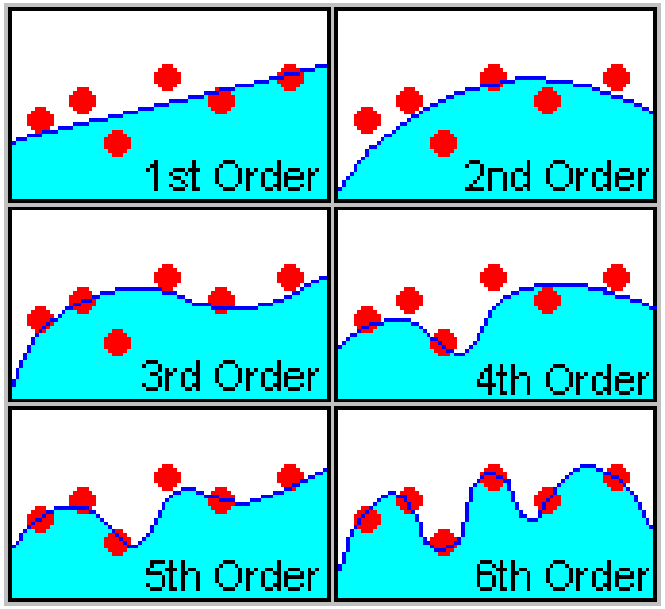
$$\hat{\beta}^{\text{ls}} = (X^\top X)^{-1} X^\top Y \quad \text{but with} \quad X = \begin{pmatrix} \phi(x_1)^\top \\ \vdots \\ \phi(x_n)^\top \end{pmatrix} \in \mathbb{R}^{n \times k}$$

Polynomial Features $\phi(\quad)??$

$$\boldsymbol{\phi}(x) = a_n \cdot x^m + a_{n-1} \cdot x^{m-1} + \cdots + a_1 \cdot x^1 + a_0$$

$$\boldsymbol{\phi}(x) = \sum_{i=0}^{m} a_i \cdot x^i$$

- **Why polynomials?**
  - They are universal approximators!
  - If the polynomial order $m \to \infty$ any continuous function can be approximated with a very small error



Weierstrass approximation theorem: *"every continuous function defined on a closed interval [a, b] can be uniformly approximated as closely as desired by a polynomial function."*

Non Linear Features

- What are "features"? $\boldsymbol{\phi}(\quad)??$

  a) Features are an arbitrary set of basis functions

$$\{\phi_1(x), \phi_2(x), \ldots, \phi_k(x)\}$$

Ex.: $\phi_1(x) = 1, \phi_2(x) = x, \phi_3(x) = x^2$ → Quadratic feature → $\phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}$

b) Any function *linear in* β can be written as $f(x) = \phi(x)^\top \beta$ for some ϕ—which we denote as "features"

Including the linear case we saw before!!!

- Augment input vector with a 1 in front (**adding bias term**):
  $\bar{x}_i = (1, x_i) = (1, x_{i1}, .., x_{id})^\top \in \mathbb{R}^{d+1}$;

$\boldsymbol{\phi(x_i) = \{1, x_{i1}, x_{i2}, \ldots x_{id}\}}$ → **Polynomial order 1**

Polynomial Features

Features are an arbitrary set of basis functions

$$\{\phi_1(x), \phi_2(x), \ldots, \phi_k(x)\}$$

Examples:

- Linear: $\phi(x) = (1, x_1, .., x_d) \in \mathbb{R}^{1+d}$
  then, $f(x) = \beta_0 + \beta_1 x_1 + \ldots + \beta_d x_d$

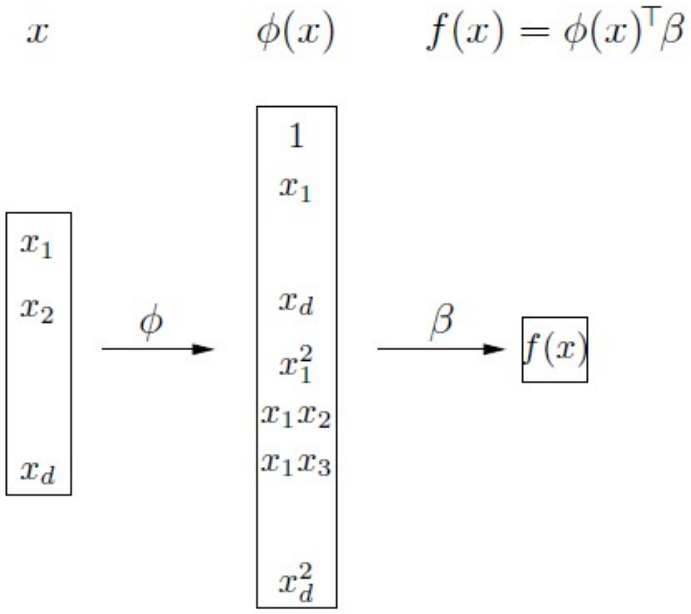
- Quadratic: $\phi(x) = (1, x_1, .., x_d, x_1^2, x_1x_2, x_1x_3, .., x_d^2) \in \mathbb{R}^{1+d+\frac{d(d+1)}{2}}$
  Example 1: $x \in \mathbb{R}$, then $\phi(x) = [1, x, x^2]^\top$, then $f(x) = \beta_0 + \beta_1 x + \beta_2 x$
  then, $f(x) = \beta_0 + \beta_1 x + \beta_2 x^2$
  Example 2: $x \in \mathbb{R}^2$ ($x = [x_1, x_2]^\top$), then $\phi(x) = [1, x_1, x_2, x_1x_2, x_1^2, x_2^2]^\top$
  then, $f(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + \beta_4 x_1^2 + \beta_5 x_2^2$

- Cubic: $\phi(x) = (.., x_1^3, x_1^2x_2, x_1^2x_3, .., x_d^3) \in \mathbb{R}^{1+d+\frac{d(d+1)}{2}+\frac{d(d+1)(d+2)}{6}}$
  Example 3: $x \in \mathbb{R}^2$ ($x = [x_1, x_2]^\top$), then

$$\phi(x) = [1, x_1, x_2, x_1x_2, x_1^2, x_2^2, x_1^2x_2, x_1x_2^2, x_1^3, x_2^3]^\top$$

Polynomial Features

Once the basis is set, we apply regression on the top of it and we estimate the coefficients β



Polynomial Features: Example

Once the basis is set, we apply regression on the top of it and we estimate the coefficients β

- Cubic: $\phi(x) = (.., x_1^3, x_1^2 x_2, x_1^2 x_3, .., x_d^3) \in \mathbb{R}^{1+d+\frac{d(d+1)}{2}+\frac{d(d+1)(d+2)}{6}}$

  Example 3: $x \in \mathbb{R}^2$ $(x = [x_1, x_2]^\top)$, then

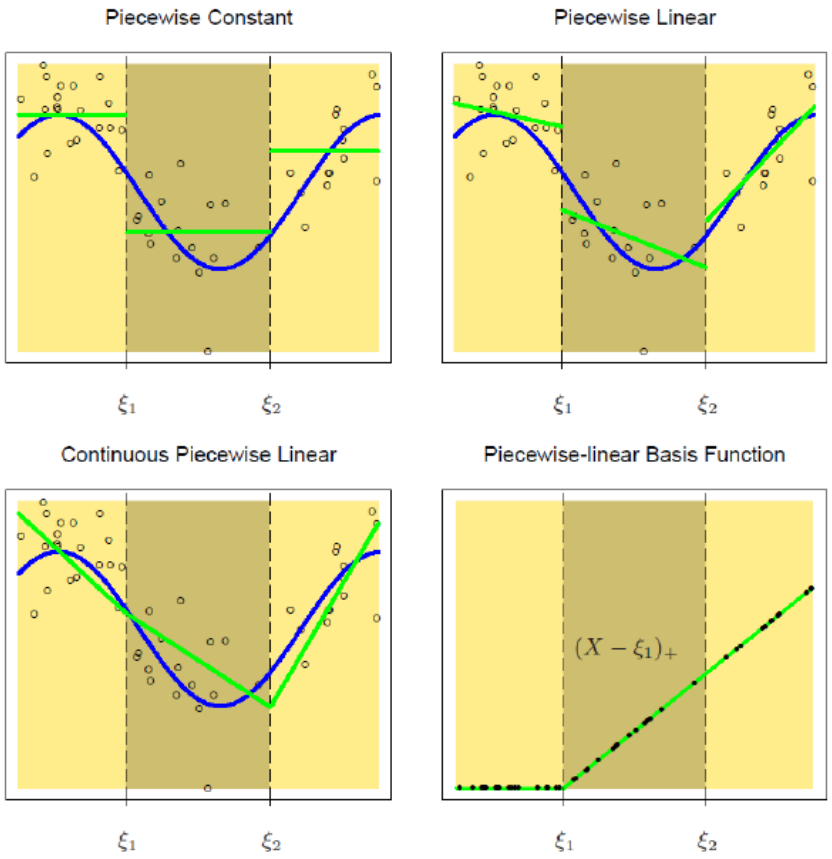
$$\phi(x) = [1, x_1, x_2, x_1 x_2, x_1^2, x_2^2, x_1^2 x_2, x_1 x_2^2, x_1^3, x_2^3]^\top$$

$$\text{then } f(x) = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \beta_3 \cdot x_1 \cdot x_2 + \beta_4 \cdot {x_1}^2 +$$
$$+\beta_5 \cdot {x_2}^2 + \beta_6 \cdot {x_1}^2 \cdot x_2 + \beta_7 \cdot x_1 \cdot {x_2}^2 + \beta_8 \cdot {x_1}^3 + \beta_9 \cdot {x_2}^3$$
$$++++$$

If $\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_8 \end{bmatrix}$ and $X = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ {x_1}^3 \\ {x_2}^3 \end{bmatrix}$  $\beta^* = (X^T X)^{-1} X^T Y$

Example: Piece-wise features (in 1D)

- Piece-wise constant: $\phi_j(x) = [\xi_j < x \leq \xi_{j+1}]$
- Piece-wise linear: $\phi_j(x) = (1, x)^\top [\xi_j < x \leq \xi_{j+1}]$
- Continuous piece-wise linear: $\phi_j(x) = [x - \xi_j]_+$ (and $\phi_0(x) = x$)

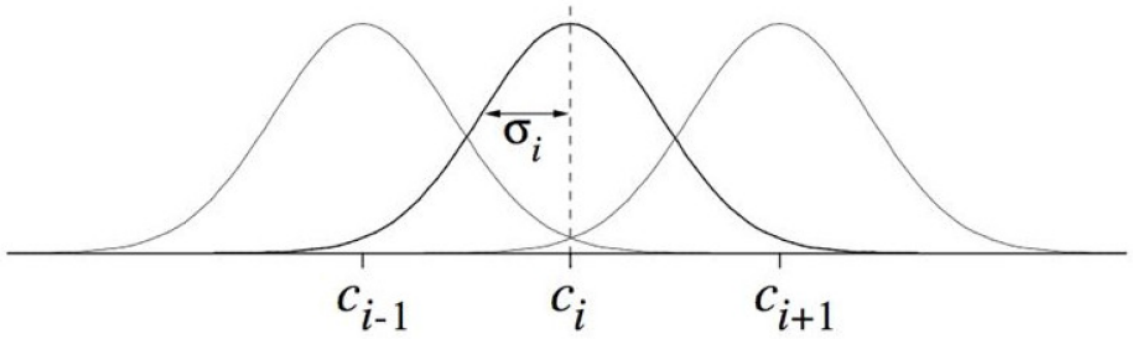


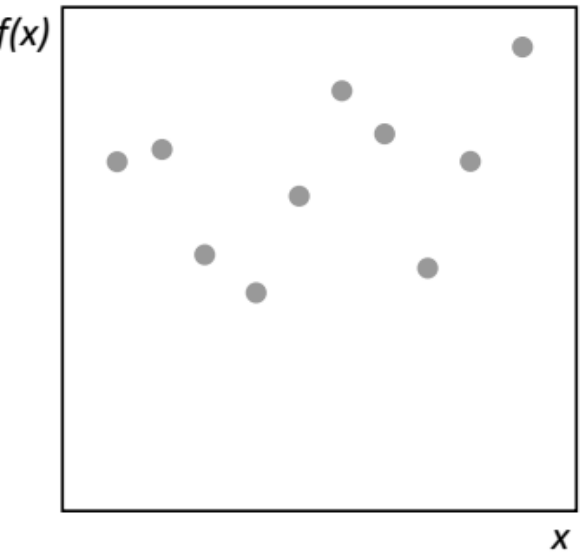
Radial Basis Functions (RBF)

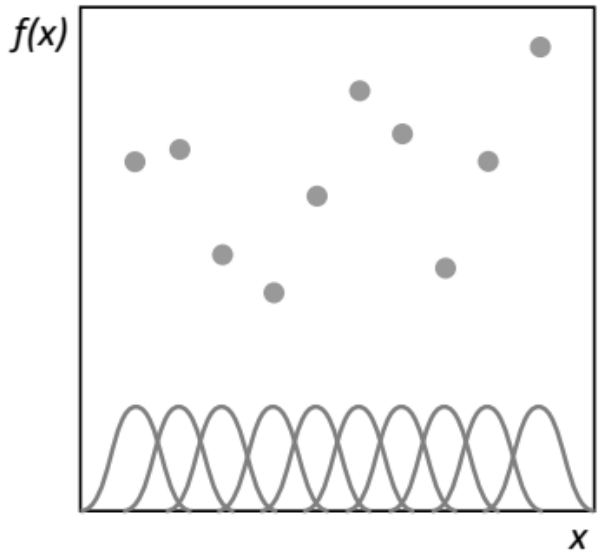
- Given a set of centers $\{c_j\}_{j=1}^{k}$, define

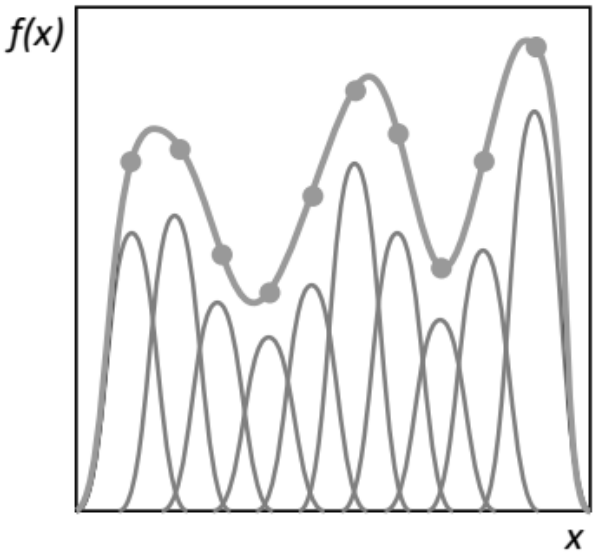
$$\phi_j(x) = b(x, c_j) = e^{-\frac{\|x - c_j\|^2}{2\sigma_i^2}} \in [0, 1]$$

Each $\phi_j(x)$ measures similarity with the center c_j









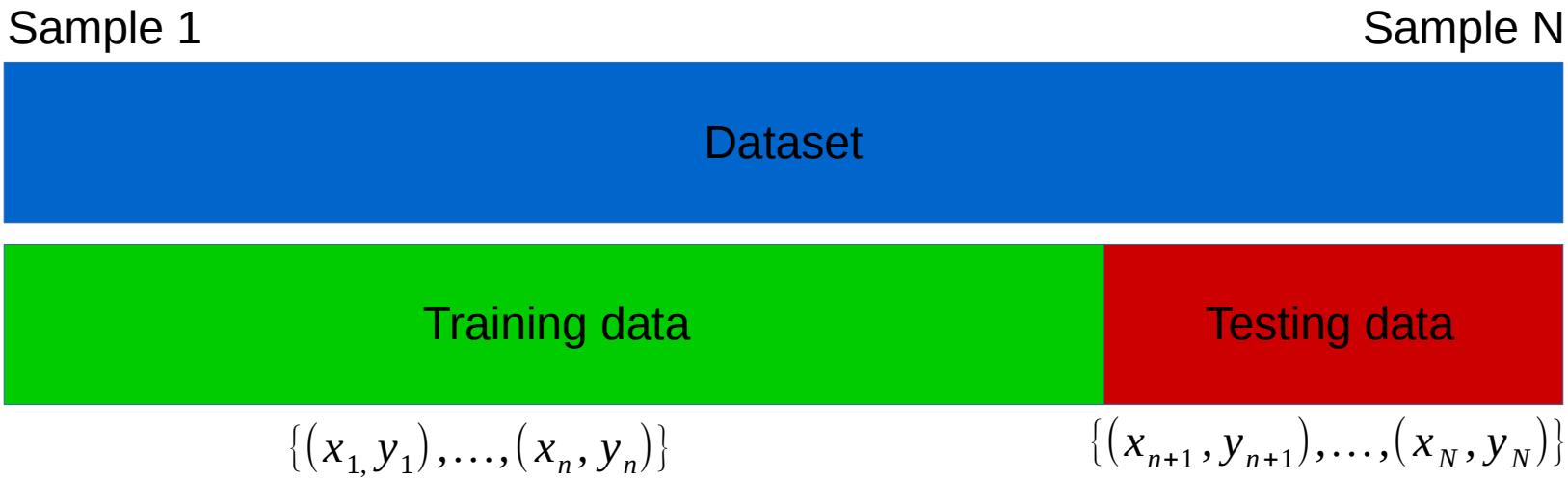
Example: Linear regression using non-linear features

- Demonstrating tasks that are also used in practical lab 3 about linear regression with non-linear features …….

Outline

- Optimal parameters
- (Non-) linear features
- **Testing & training error**
- Over-fitting vs. under-fitting
- Regularization
- Cross-validation

Testing & training error



e.g. linear regression

$$\underset{\beta}{minimize} \sum_{i=1}^{n} (y_i - f(x_i, \beta))^2$$

Training

Fitted model $f(x, \beta^*)$

Testing error $\sum_{i=n+1}^{N} (y_i - f(x_i, \beta^*))^2$

Training error $\sum_{i=1}^{n} (y_i - f(x_i, \beta^*))^2$

Outline

- Optimal parameters
- (Non-) linear features
- Testing & training error
- **Over-fitting vs. under-fitting**
- Regularization
- Cross-validation

Over-fitting vs. under-fitting

Pro: We can approximate any relationship/function

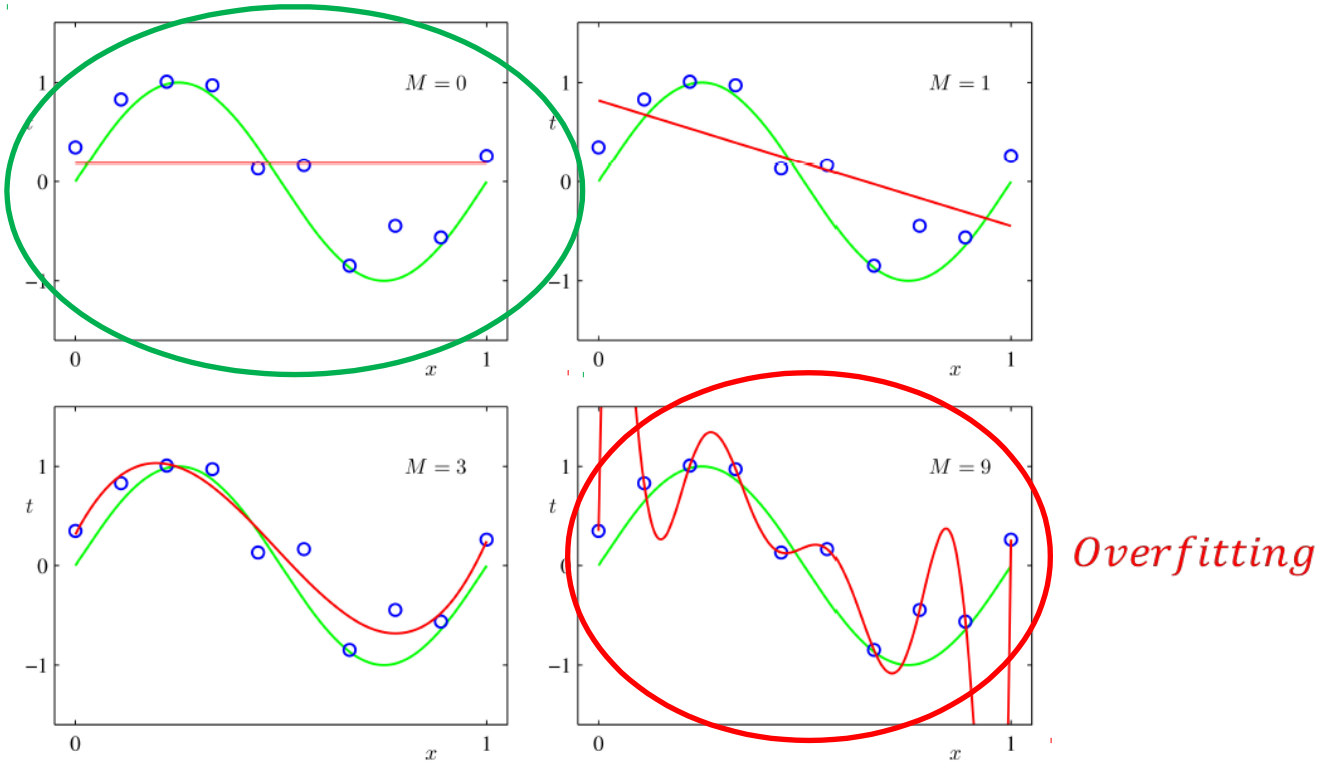
Con: We do not really know the underlying relationship and its order

- Try empirically different options and see how they predict
- You may be tempted to choose a very high order features

e.g.1: Using cubic polynomial features to learn the Celsius to Fahrenheit relation

e.g.2: Fit a M-order polynomial to a data set generated from y = sin(x):

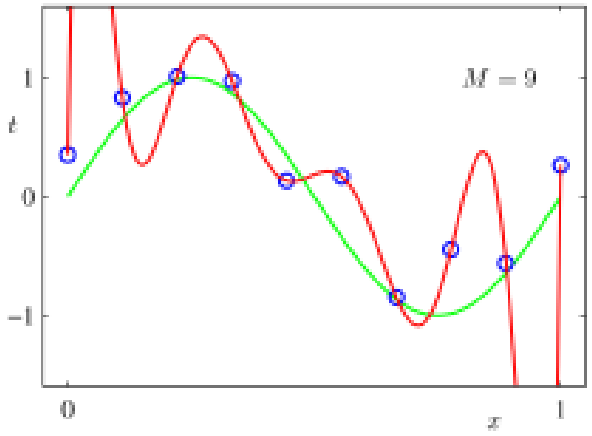
Underfitting

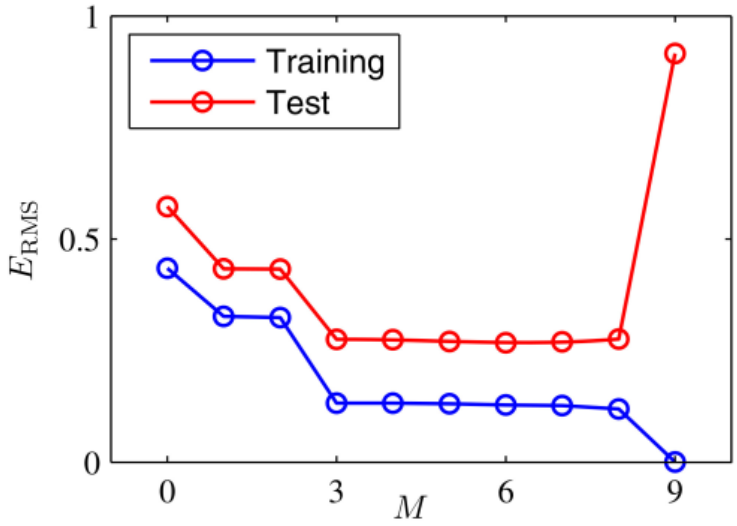


Overfitting

Over-fitting vs. under-fitting

- **Over-fitting is as bad as under-fitting**
- **Under-fitting:** Poor approximation ⇨ Poor performance
- **Over-fitting:** Learn the training data too literally
  Will not be able to generalize to new test data



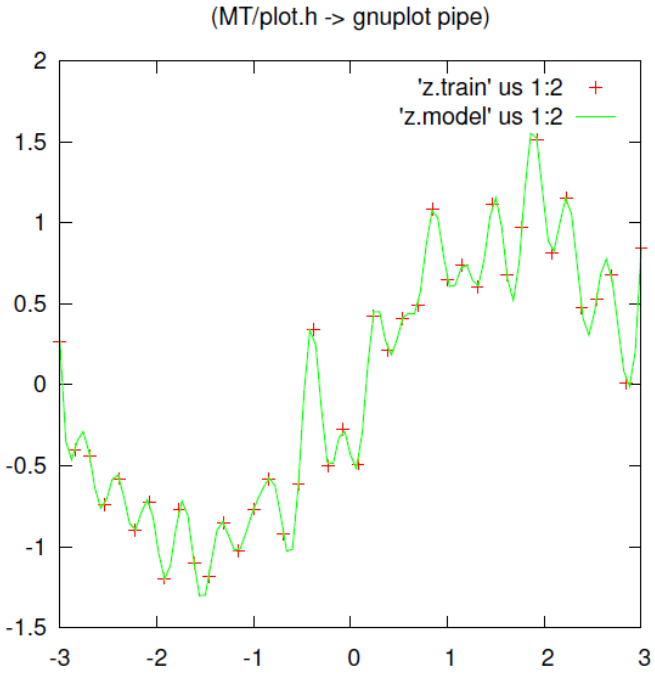


Outline

- Optimal parameters
- (Non-) linear features
- Testing & training error
- Over-fitting vs. under-fitting
- **Regularization**
- Cross-validation

The need for regularization

Noisy $\sin$ data fitted with radial basis functions



- Overfitting & generalization:
  The model overfits to the data—and generalizes badly

Regularisation will try to avoid overfitting even when choosing a too complex model

Concept: During optimisation, an oposing force will try to limit/reduce the complexity of the model and/or stop fitting too much to the training data

Ridge regression: L_2-regularization

- We add a *regularization* to the cost:

$$L^{\text{ridge}}(\beta) = \sum_{i=1}^{n}(y_i - \phi(x_i)^\top\beta)^2 + \lambda\sum_{j=2}^{k}\beta_j^2$$

A weighting factor to balance both constraints

Minimise the Loss function

Minimise the error between prediction and training data

Fit to the training data

Minimise the sum of all the regression coefficients to the square

Most coefficients will be very small

NOTE: β_1 is usually *not* regularized!

Ridge regression: L_2-regularization

Example of **energy demand prediction**

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

x y

(from Nando de Freitas's lecture)

- regress: input = {wind-speed, #people}, output={energy-requirement}

With a linear feature and parameters $\beta = [\beta_0, \beta_1 \beta_2]^\top$

$$L^{\text{ls}}(\beta) = (y_1 - \bar{x}_1^\top \beta)^2 + (y_2 - \bar{x}_2^\top \beta)^2 + (y_3 - \bar{x}_3^\top \beta)^2 + (y_4 - \bar{x}_4^\top \beta)^2 + \lambda(\beta_0^2 + \beta_1^2 + \beta_2^2)$$

Regularization term

Ridge regression: L_2-regularization

- We add a *regularization* to the cost:

$$L^{\text{ridge}}(\beta) = \sum_{i=1}^{n} (y_i - \phi(x_i)^\top \beta)^2 + \lambda \sum_{j=2}^{k} \beta_j^2$$

NOTE: β_1 is usually *not* regularized!

Minimisation: $$\frac{\partial L^{ridge}}{\partial \beta} = 0$$

- **Optimum**:

$$\hat{\beta}^{\text{ridge}} = (X^\top X + \lambda I)^{-1} X^\top Y$$

(where $I = \mathbf{I}_k$, or with $I_{1,1} = 0$ if β_1 is not regularized)

Ridge regression: L_2-regularization

Minimize:

$$L^{\text{ridge}}(\beta) = \sum_{i=1}^{n}(y_i - \phi(x_i)^\top\beta)^2 + \lambda\sum_{j=2}^{k}\beta_j^2 \quad \xrightarrow{\text{find } \beta} \quad \frac{\partial L^{ridge}}{\partial \beta} = 0$$

$$\frac{\partial L^{ls}}{\partial \beta} = -2(Y - X\beta)^T X + 2\lambda\beta^T = 0 \longrightarrow (-2(Y - X\beta)^T X)^T + 2\lambda\beta = 0^T$$

$$\Longleftrightarrow \quad 0^T = -2X^T(Y - X\beta) + 2\lambda\beta$$

$$\Longleftrightarrow \quad 0^T = -2X^T Y + 2X^T X\beta + 2\lambda\beta$$

$$\Longleftrightarrow \quad 2X^T Y = 2(X^T X + \lambda I)\beta$$

$$\Longleftrightarrow \quad (X^T X + \lambda I)^{-1} X^T Y = \beta$$

Choosing λ: generalization error & cross validation

- We add a *regularization* to the cost:

$$L^{\text{ridge}}(\beta) = \sum_{i-1}^{n} (y_i - \phi(x_i)^\top \beta)^2 + \lambda \sum_{j=2}^{k} \beta_j^2$$

NOTE: β_1 is usually *not* regularized!

- $\lambda = 0$ will always have a lower *training* data error
  We need to estimate the *generalization* error on test data

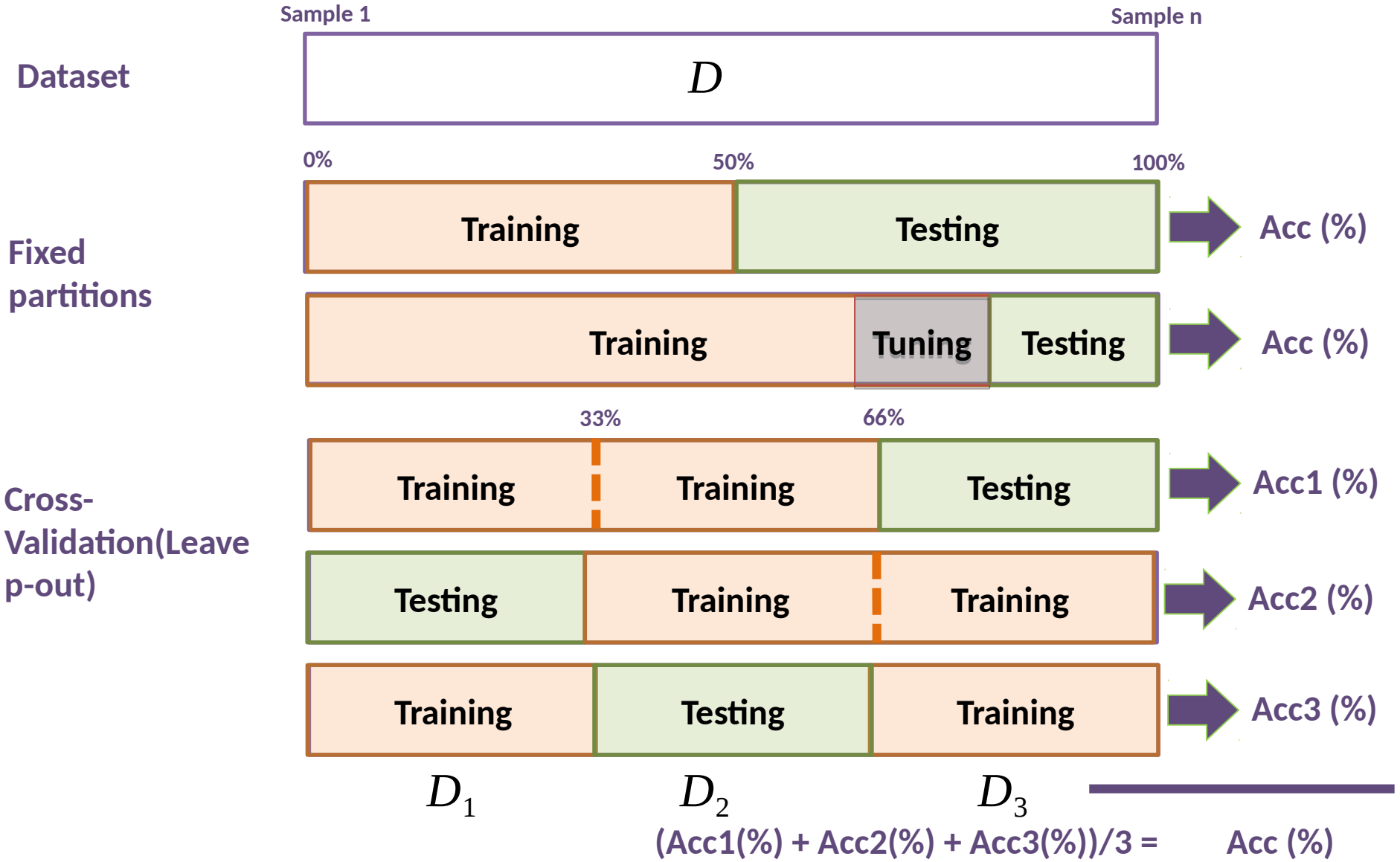
- $\lambda \to \infty$: we get a trivial but useless solution $\beta = 0$

- λ is a parameter of the method
  - Tuned empirically
  - Dedicate a percentage of the dataset to evaluate different values and choose the best

Outline

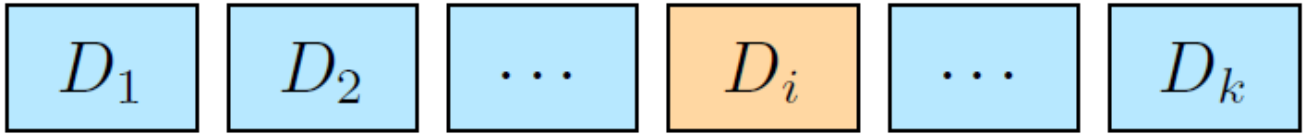
- Optimal parameters
- (Non-) linear features
- Testing & training error
- Over-fitting vs. under-fitting
- Regularization
- **Cross-validation**

Cross-validation



Choosing λ: generalization error & cross validation

- *k-fold cross-validation:*

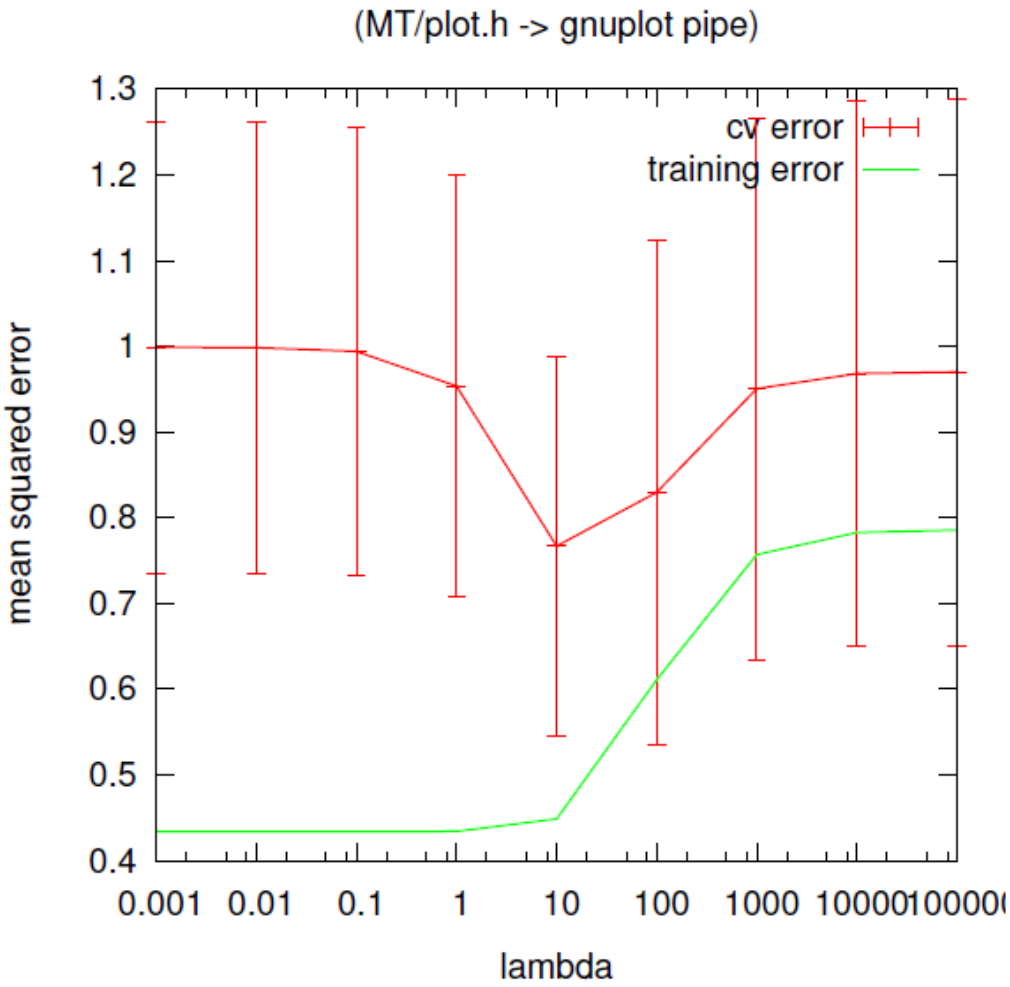


1: Partition data D in k equal sized subsets $D = \{D_1, .., D_k\}$
2: **for** $i = 1, .., k$ **do**
3: compute $\hat{\beta}_i$ on the training data $D \setminus D_i$ leaving out D_i
4: compute the error $\ell_i = L^{\text{ls}}(\hat{\beta}_i, D_i)/|D_i|$ on the validation data D_i
5: **end for**
6: report mean squared error $\hat{\ell} = 1/k \sum_i \ell_i$ and variance $1/(k\text{-}1)[(\sum_i \ell_i^2) - k\hat{\ell}^2]$

- **Choose λ for which $\hat{\ell}$ is smallest**

Choosing λ: generalization error & cross validation

quadratic features on sinus data:



Lasso: L_1-regularization

- We add a L_1 regularization to the cost:

$$L^{\text{lasso}}(\beta) = \sum_{i=1}^{n} (y_i - \phi(x_i)^\top \beta)^2 + \lambda \sum_{j=1}^{k} |\beta_j|$$

NOTE: β_1 is usually not regularized!

Minimise the sum of all absolute values of the regression coefficients

Ridge Regularisation:

Most coefficients will be very small

Many coefficients will be zero

- Has no closed form expression for optimum

Ridge Regularisation vs. Lasso

Ridge Regularisation:

Most coefficients will be very small

Lasso Regularisation:

Many coefficients will be zero

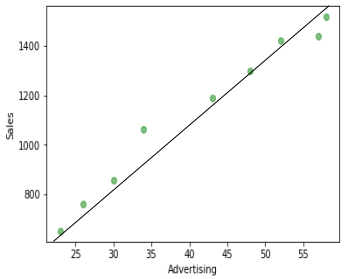
$$y_i = \beta_0 + \beta_1 x_{i1} + \ldots + \beta_d x_{id}$$

All the features x_{ij} multiplied by a zero coefficient β_j is not used in the regression

- Lasso $\rightarrow$ sparsity! feature selection!

Summary

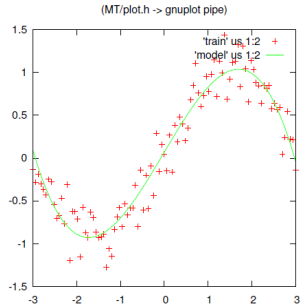
Training Data $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$



Data is linearly separable
Linear feature
$f(x) = \beta^T x$

where $\phi(x) = \begin{bmatrix} 1 \\ x \end{bmatrix}$

Data is not linearly separable
Non-linear feature
$f(x) = \beta^T \phi(x)$



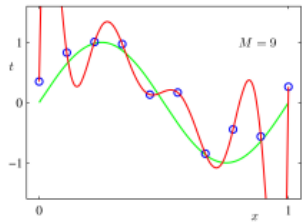
Linear Regression

Least square error

$$\underset{\beta}{minimize} \sum_{i=1}^{n} (y_i - f(x_i, \beta))^2$$

$$\beta^* = (X^T X)^{-1} X^T Y$$

Overfitting



$$L^{\text{ridge}}(\beta) = \sum_{i=1}^{n} (y_i - \phi(x_i)^\top \beta)^2 + \lambda \sum_{j=2}^{k} \beta_j^2$$

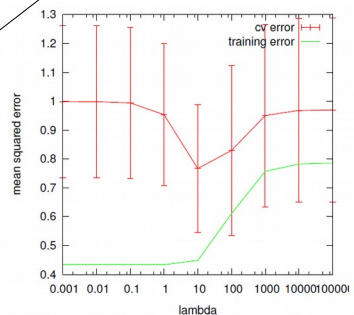
$$\beta^* = (X^T X + \lambda I)^{-1} X^T Y$$

Optimum generalization error

Note: $X = \begin{pmatrix} \phi(x_1)^\top \\ \vdots \\ \phi(x_n)^\top \end{pmatrix} \in \mathbb{R}^{n \times k}$

Cross-validation

e.g. choose optimum λ



Summary

- **Representation:** choice of features

$$f(x) = \phi(x)^\top \beta$$

- **Evaluation:** squared error + Ridge/Lasso regularization

$$L^{\text{ridge}}(\beta) = \sum_{i=1}^{n} (y_i - \phi(x_i)^\top \beta)^2 + \lambda \|\beta\|_I^2$$

- **Optimization:** analytical (or quadratic program for Lasso)

$$\hat{\beta}^{\text{ridge}} = (X^\top X + \lambda I)^{-1} X^\top y$$

Summary

- **Linear models** on non-linear features—extremely powerful

linear polynomial RBF kernel	Ridge Lasso	regression classification*

*logistic regression

- Generalization $\leftrightarrow$ **Regularization** $\leftrightarrow$ complexity/DoF penalty

DoF:Degree of freedom

- **Cross validation** to estimate generalization empirically $\rightarrow$ use to choose regularization parameters