Machine Learning

Session

0 | ML – Quick intro

Machine Learning

What is Machine Learning?

- Learning general models from particular examples (data)
  - **Data** is (mostly) cheap and abundant
  - **Knowledge** is expensive and scarce
- Example in retail:
  From customer transactions to consumer behavior
  People who bought “Game of Thrones” also bought “Lord of the Rings” [Amazon.com]
- Goal: Build a model that is good and useful approximation to the data

Machine Learning

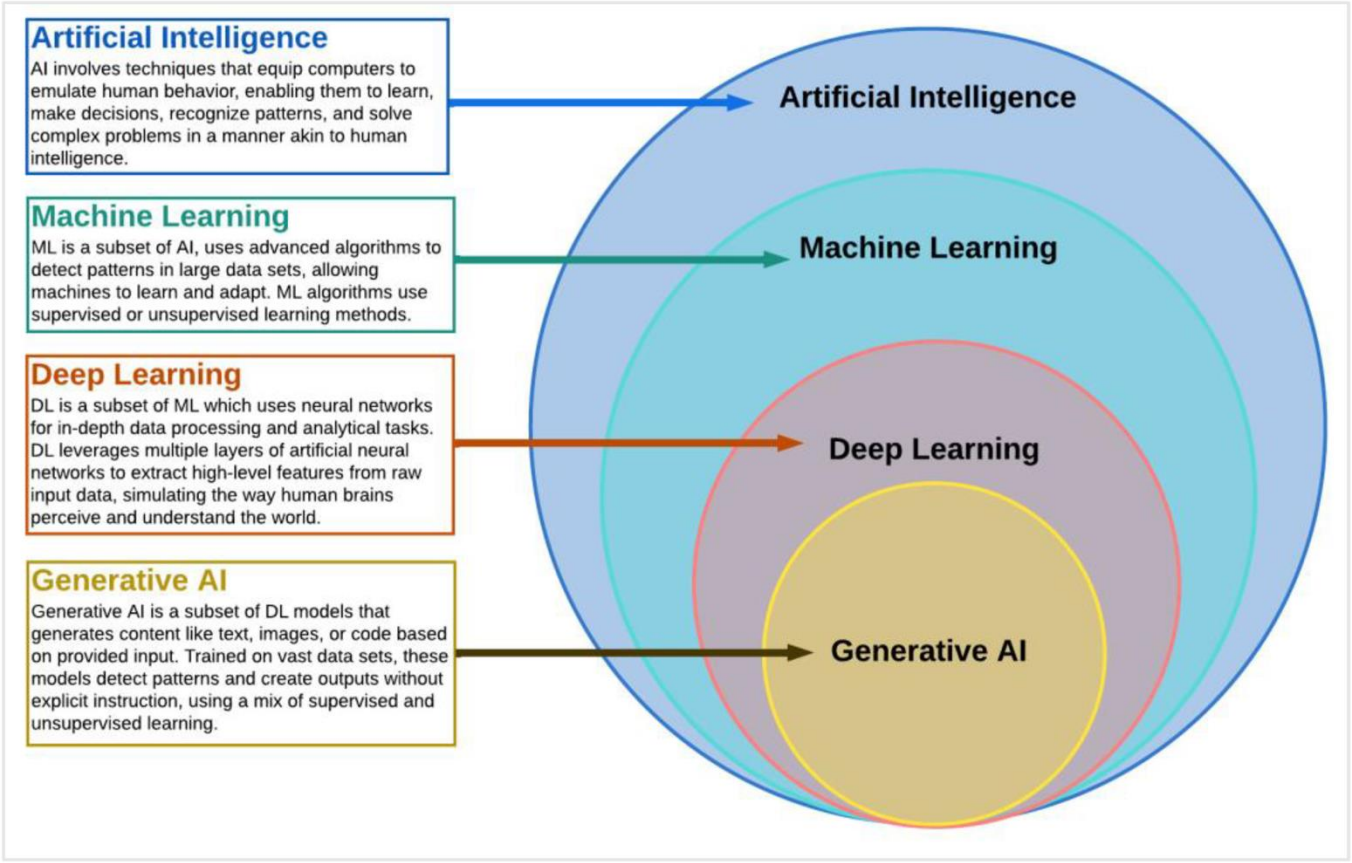
What is Machine Learning?

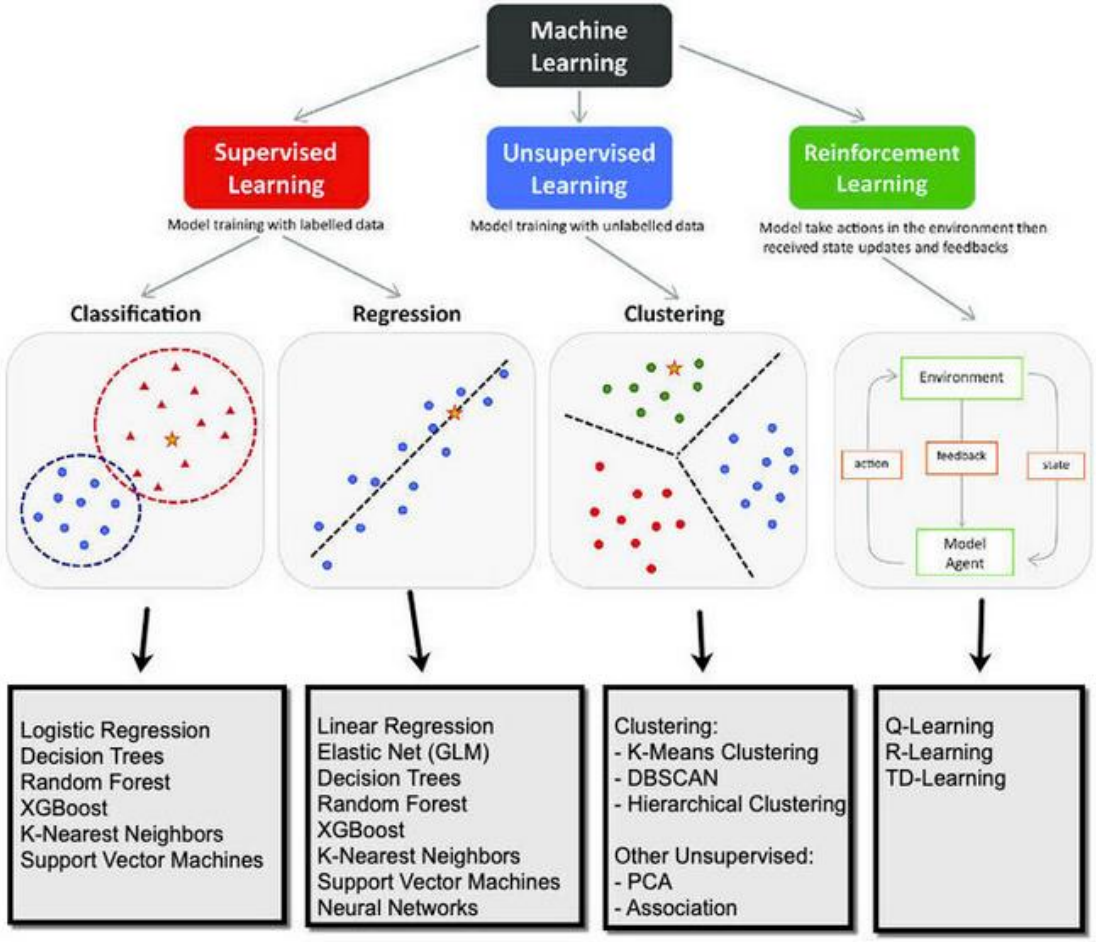
- Optimizing a performance criterion using example data or past experience

- Role of **Statistics**
  - Build mathematical models to make inference from a sample

- Role of **Computer Science**
  - Solve the optimization problem
  - Represent and evaluate the mode for inference

Machine Learning – Global overview



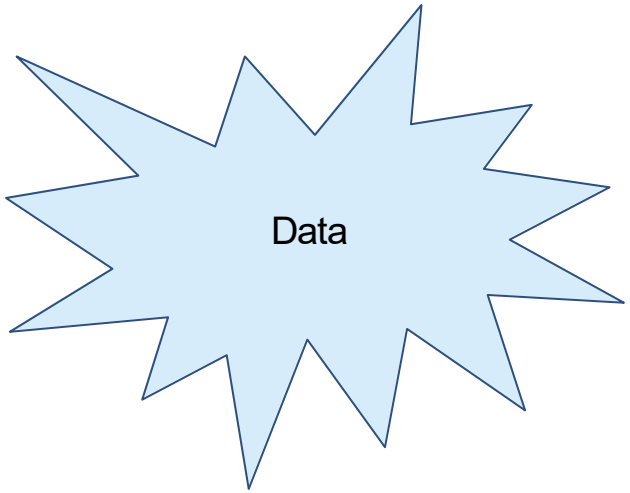


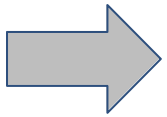
1 | Regression vs Classification

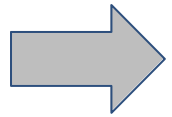
Machine Learning - Linear Models - Theory

Zoo of Machine Learning Problems - Supervised learning

Make **Predictions**







$$f(x) \approx y$$

f is the decision function

$$D = \{x_1, x_2, ..., x_n\}$$

p

n

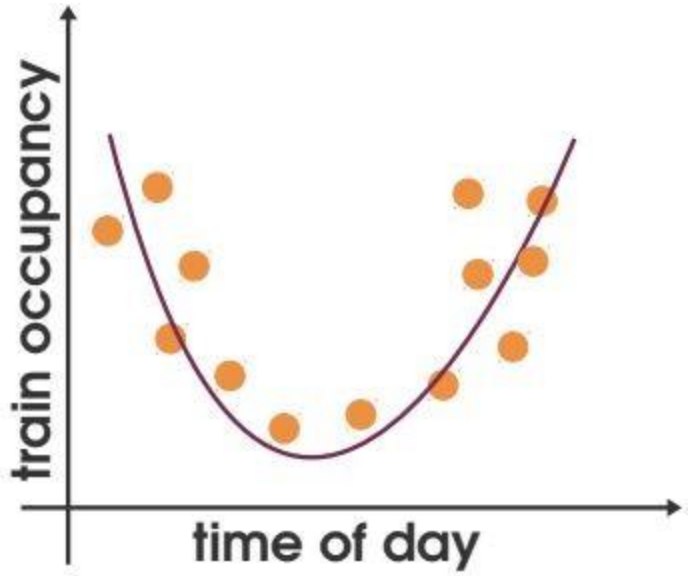
X

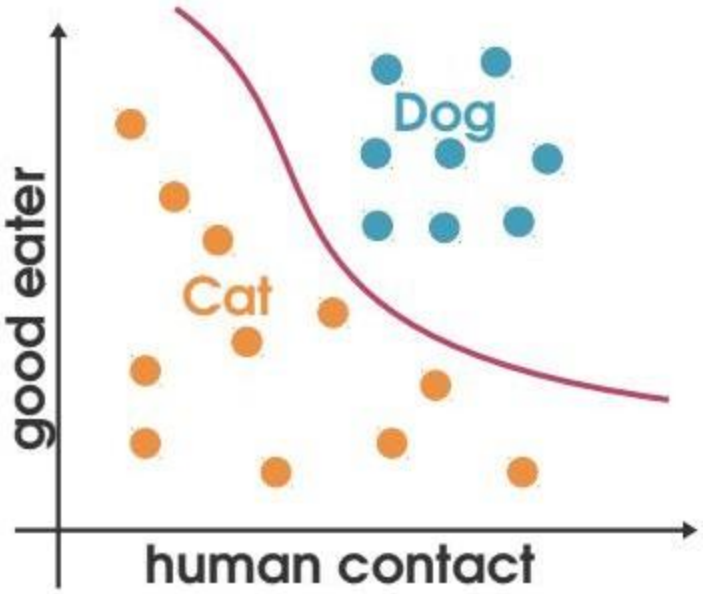
n

y

Machine Learning - Linear Models - Theory

Zoo of Machine Learning Problems - Supervised learning – Regression vs Classification





Machine Learning - Linear Models - Theory

Zoo of Machine Learning Problems - Supervised learning - Regression
Application

- **Click prediction**
  - How many people will click on this ad? Comment on this post? Share this article on social media?

- **Load prediction**
  - How many users will my service have at a given time?

- **Algorithmic trading**
  - What will the price of this share be?

- **Drug Development**
  - What is the binding affinity between this drung candidate and its target? What is the sensibility of the tumor to this drug?

Machine Learning - Linear Models - Theory

Zoo of Machine Learning Problems - Supervised learning - Classification

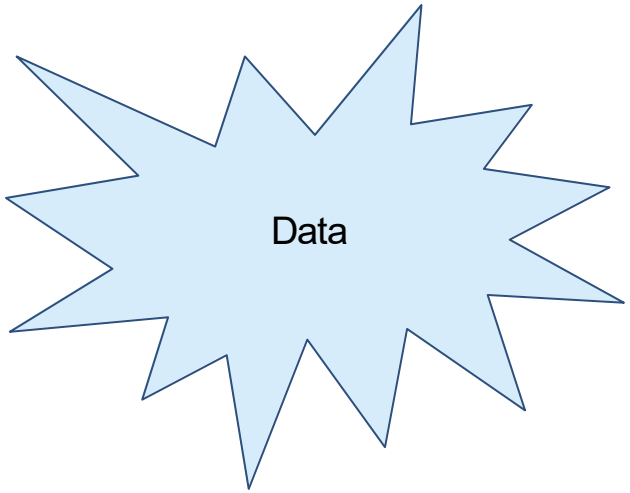
Application

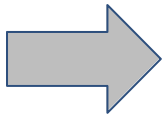
- **Face recognition**
  - Identify faces independently of pose, lighting, occlusion (glasses, beard), make-up, hair style
- **Vehicle identification** (self-driving cars)
- **Character recognition**
  - Read letters or digits independently of different handwriting styles.
- **Sound recognition**
  - Which language is spoken? Who wrote this music? What type of bird is this?
- **Spam detection**
- **Precision medicine**
  - Does this sample come from a sick or healthy person? Will this drug work on this patient?
- **Intent prediction** (Natural Language Processing)

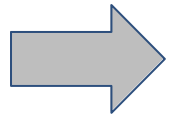
Machine Learning - Linear Models - Theory

Zoo of Machine Learning Problems - Supervised learning - Regression

Make **continuous** Predictions







$$f(x) \approx y$$

$$D = \{x_1, x_2, ..., x_n\}$$

$$y_i \in \mathbb{R}$$

p

n

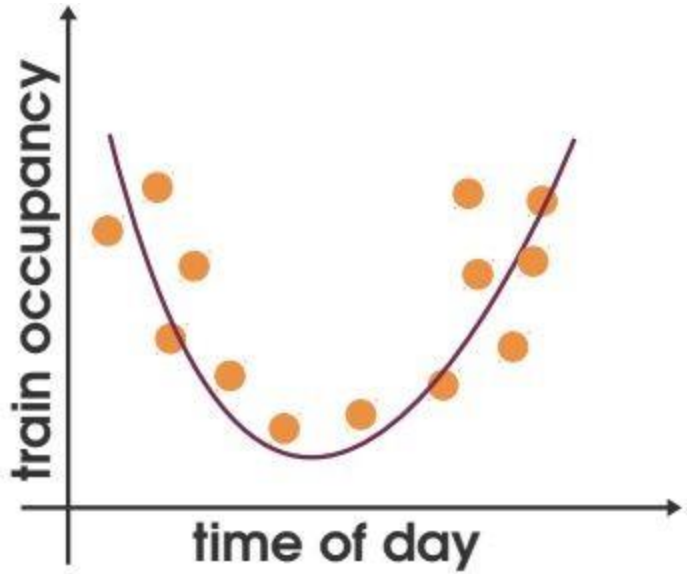
X

n

y

Machine Learning - Linear Models - Theory

Zoo of Machine Learning Problems - Supervised learning - Regression



Training Set

$$D = \{x_i, y_i\}_{i=1,\ldots,n}$$

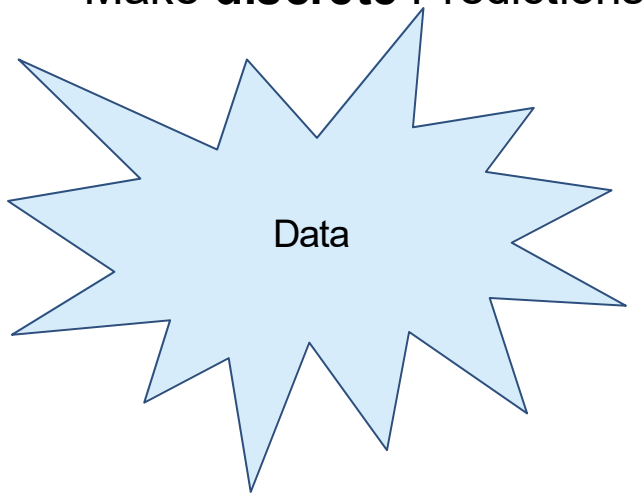
$$y_i \in \mathbb{R}$$

Given D, find F such that $f(x) \approx y$

Machine Learning - Linear Models - Theory

Zoo of Machine Learning Problems - Supervised learning - Classification

Make **discrete** Predictions

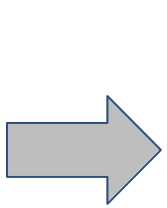


$$D = \{x_1, x_2, ..., x_n\}$$

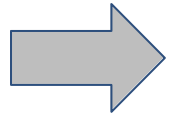
p

n **X**

n **y**



Labels



$$f(x) \approx y$$

f is the decision function

Binary classification

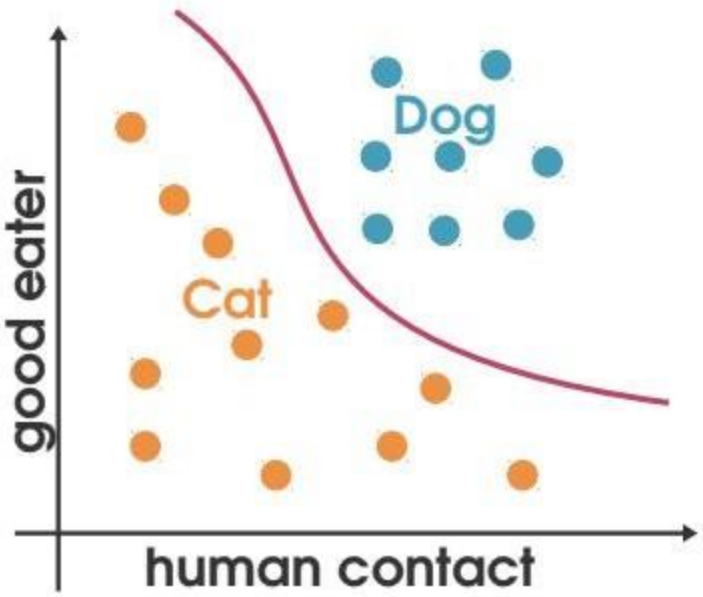
$$y_i \in \{0, 1\}$$

Multi-class classification

$$y_i \in \{0, 1, ..., k\}$$

Machine Learning - Linear Models - Theory

Zoo of Machine Learning Problems - Supervised learning - Classification



Training Set

$$D = \{x_i, y_i\}_{i=1,\ldots,n}$$

$$y_i = \begin{cases} 1 & if\ x_i \in P \\ 0 & if\ x_i \in N \end{cases}$$

Given D, find F such that $f(x) \approx y$

Machine Learning - Linear Models - Theory

Zoo of Machine Learning Problems - Supervised learning - Supervised learning Setting

Given a training set $D = \{x_i, y_i\}_{i=1,\ldots,n}$ find f such that $f(x) \approx y$

Features
Descriptors
Variable
attributes p

Outcome
Target
Label

Observations
Samples
Data points

n

Data matrix
Design matrix

X

n

y

Binary classification

$y_i \in \{0, 1\}$

Multi-class classification

$y_i \in \{0, 1, \ldots, k\}$

Regression

$y_i \in \mathbb{R}$

Machine Learning - Linear Models - Theory

Exercise:
You are working in the retail industry. You want to know the cost to produce and sell a given product in the past. Finance or Accounting teams can give all the associated cost from historical data.

Question: Is it a Machine Learning Problem?

Machine Learning - Linear Models - Theory

Exercise:
You are working in the retail industry. You want to know the cost to produce and sell a given product in the past. Finance or Accounting teams can give all the associated cost from historical data.

Question: Is it a Machine Learning Problem?

Answer: No! There is nothing to be learnt. Every necessary information are to be found in the company financial data.

Machine Learning - Linear Models - Theory

Exercise:
You are working in the retail industry. You want to compute the price elasticity of a given product (= by how much the sales are going down for every €in price product).

Question: Is it a Machine Learning Problem?

Machine Learning - Linear Models - Theory

Exercise:
You are working in the retail industry. You want to compute the price elasticity of a given product (= by how much the sales are going down for every €in price product).

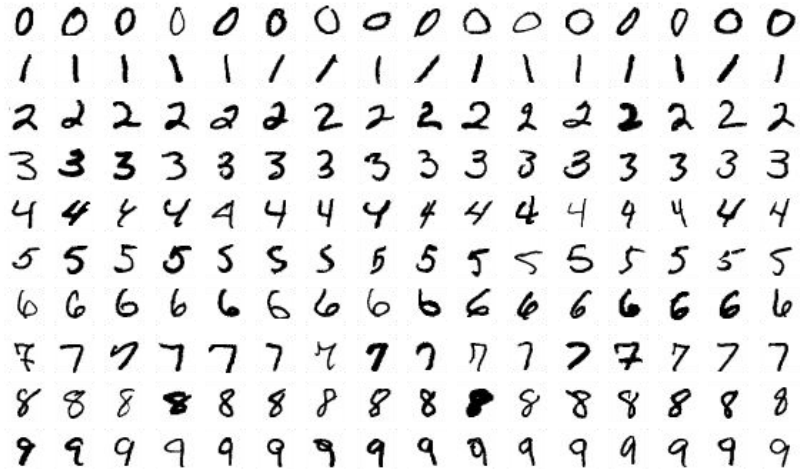
Question: Is it a Machine Learning Problem?

Answer: Yes! You don't know the formula to compute the price elasticity for all possible prices for this product. You need to infer the rules. Your training data is all the product prices with some extra features to help, the targets is the amount of sales.
This can be solved as a Supervised Learning, Regression task.

Machine Learning - Linear Models - Theory

Exercise:
You are working in the banking industry. Your team is responsible to read digits (32x32 pixels) on check.

(MNIST dataset)

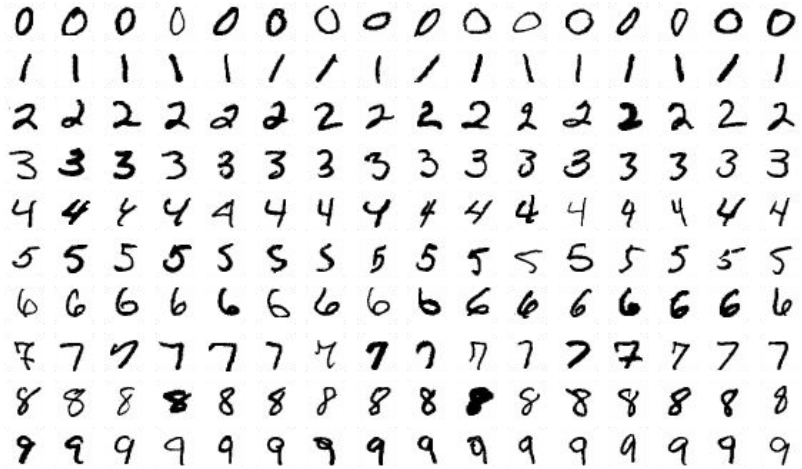


Question: Is it a Machine Learning Problem?

Machine Learning - Linear Models - Theory

Exercise:
You are working in the banking industry. Your team is responsible to read digits (32x32 pixels) on check.

(MNIST dataset)



Question: Is it a Machine Learning Problem?

Answer: Yes! It is supervised learning, classification (the classes are the digits 0,1,...9)

2 | Simple Linear Regression

Machine Learning - Linear Models - Theory

Supervised learning: 3 ingredients

A **decision function**
Eg

$$f(x) = \sum_{j=1}^{p} \beta_j x_j$$

A **Loss Function :** Quantifies how far the decision function is from the truth (=**oracle**)

$$E_{\mathcal{D}} = \sum_{i=1}^{n} \mathcal{L}(y^i, f(\boldsymbol{x}^i))$$

An **Optimization procedure**

$$f^* = \arg\min_{f \in \mathcal{F}} E_{\mathcal{D}}$$

Machine Learning - Simple Linear Regression

What is Simple Linear Regression?
A statistical method that allow us to summarize and study relationshi between two continuous (quantitative) variables:

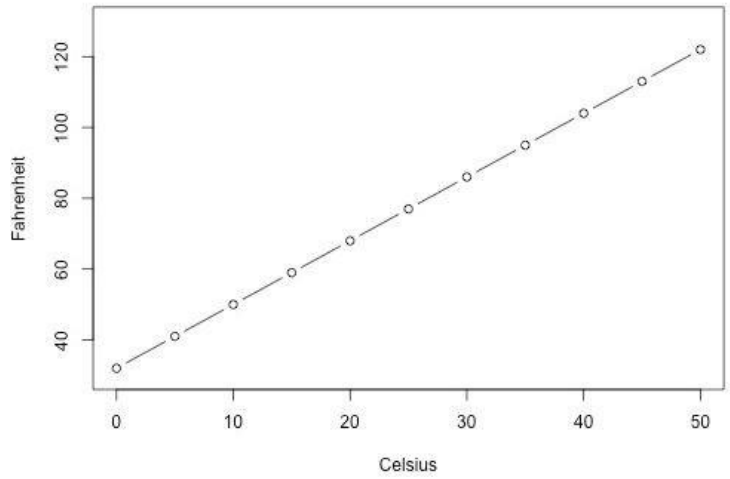
- **x**, is regarded as the **predictor**, **explanatory**, or **independent** variable
- **y** is regarded as the **response**, **outcome** or **dependent** variable

Simple because there is only one predictor (number of features p equal to 1)
In contrast, we will study **Multiple** linear regression later on in this class (two or more predictor variables)

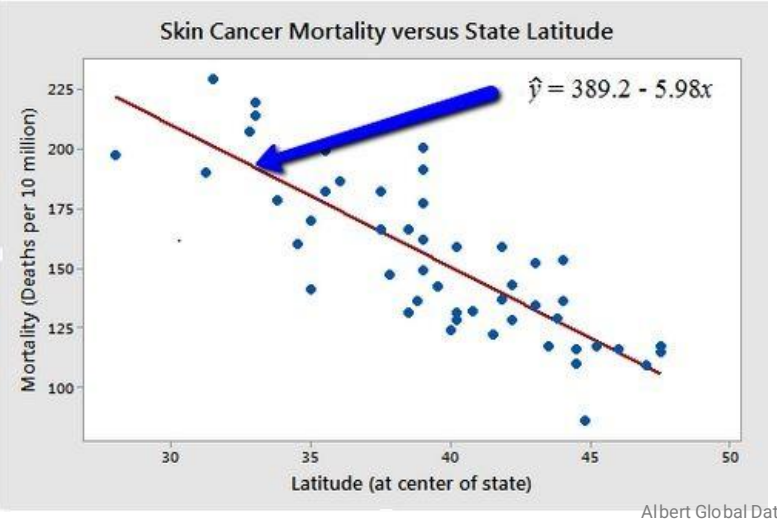
Machine Learning - Simple Linear Regression

Types of relationships

- **Deterministic** or **functional** relationship
  - Eg: Fahrenheit = 9/5 Celsius + 32



- **Statistical** relationship, when the relationship is not “perfect”
  Eg US Skin cancer dataset



Machine Learning - Simple Linear Regression

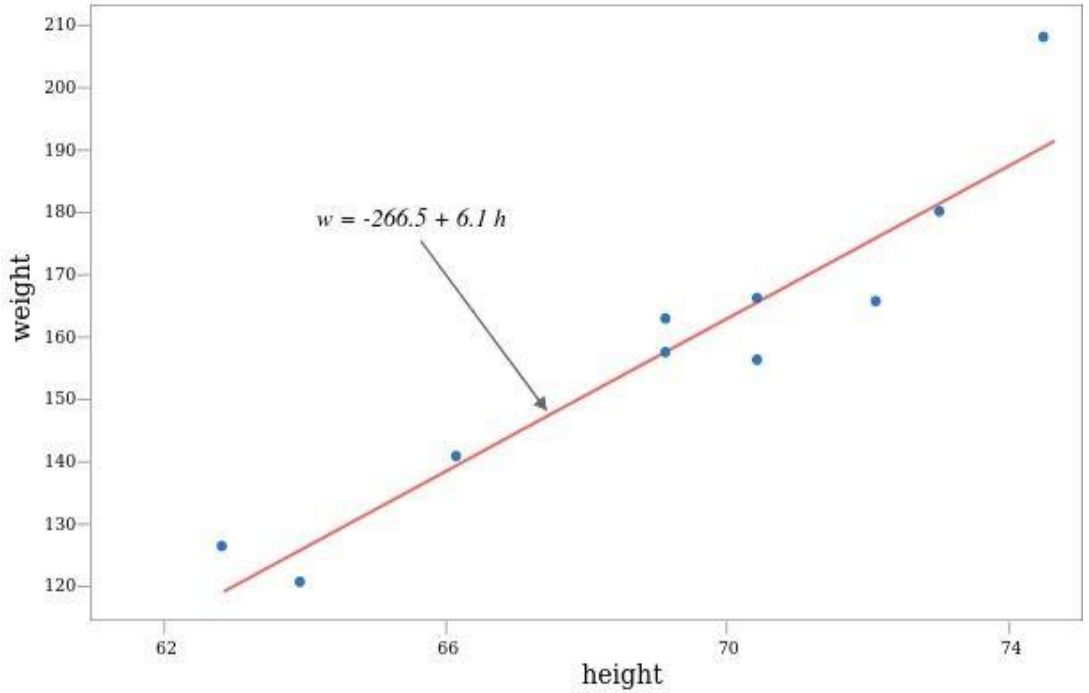
What is the "Best Fitting line"?
ŷ is the **predicted response** or **fitted value**

ŷ = a x + b

a: slope, or **coefficient**
b: intercept to origin

Prediction error e aka **residuals**

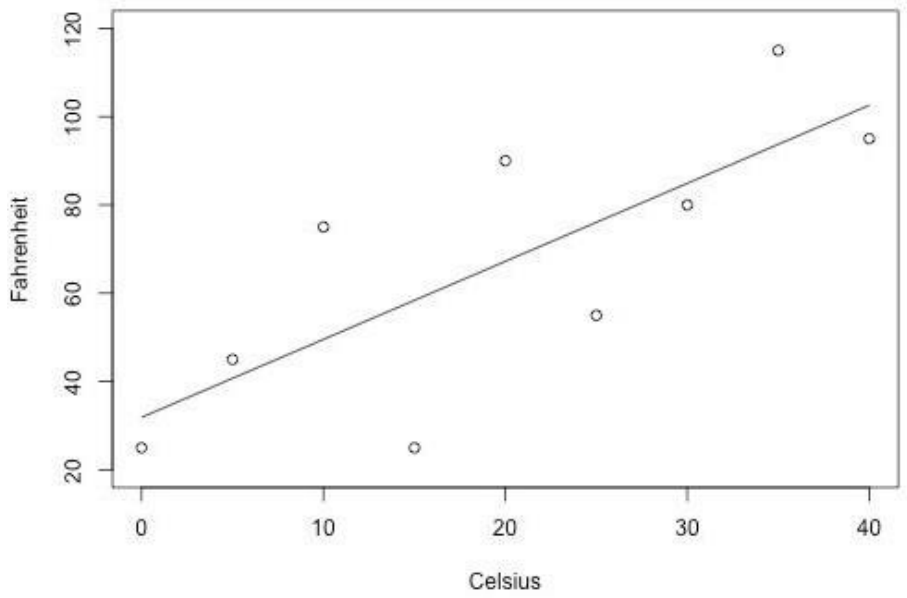
$$e_i = y_i - \hat{y}_i$$



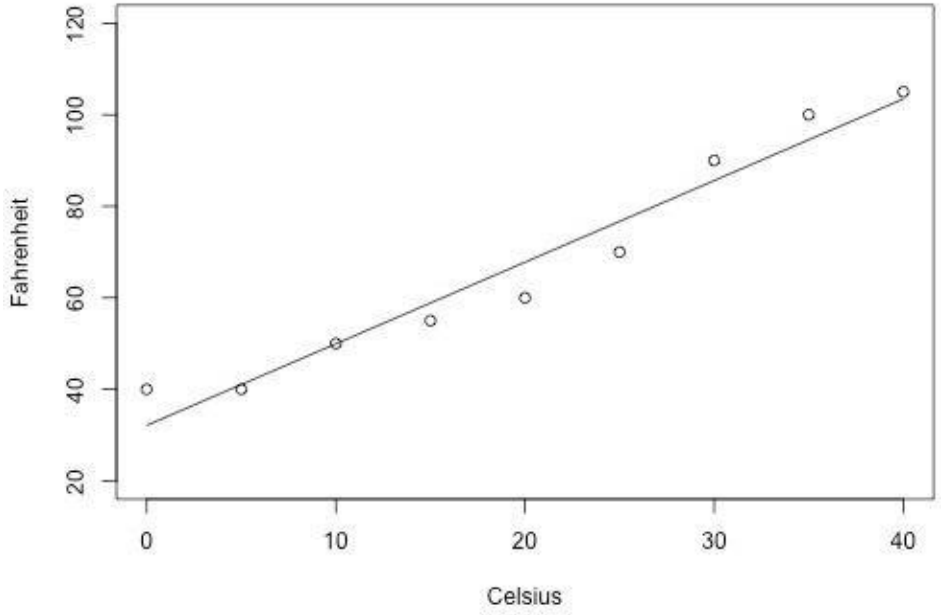
3 | First Loss Functions

Machine Learning - Simple Linear Regression

Loss function - Common Error Measurement
Example: Thermometer comparison



Thermometer A



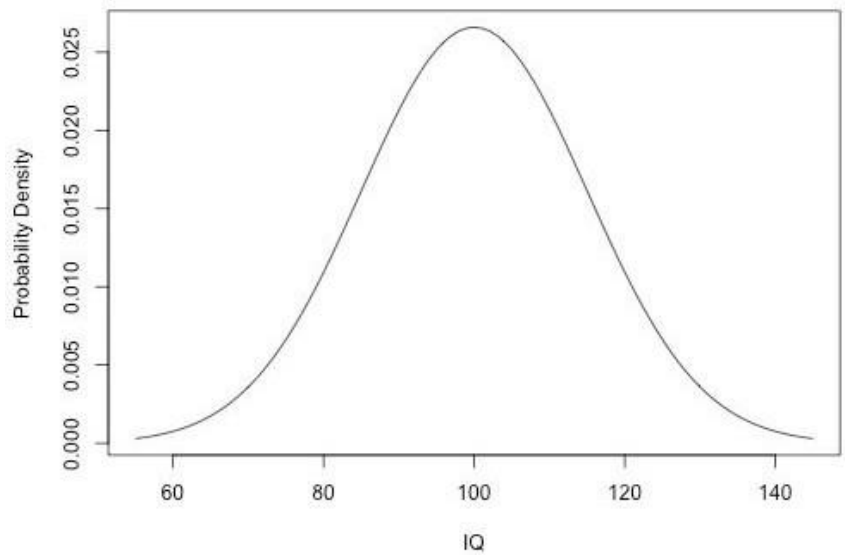
Thermometer B

$$\mu_Y = E(Y) = \beta_0 + \beta_1 x$$

Machine Learning - Simple Linear Regression

Loss function - Common Error Measurement
Recap on variance

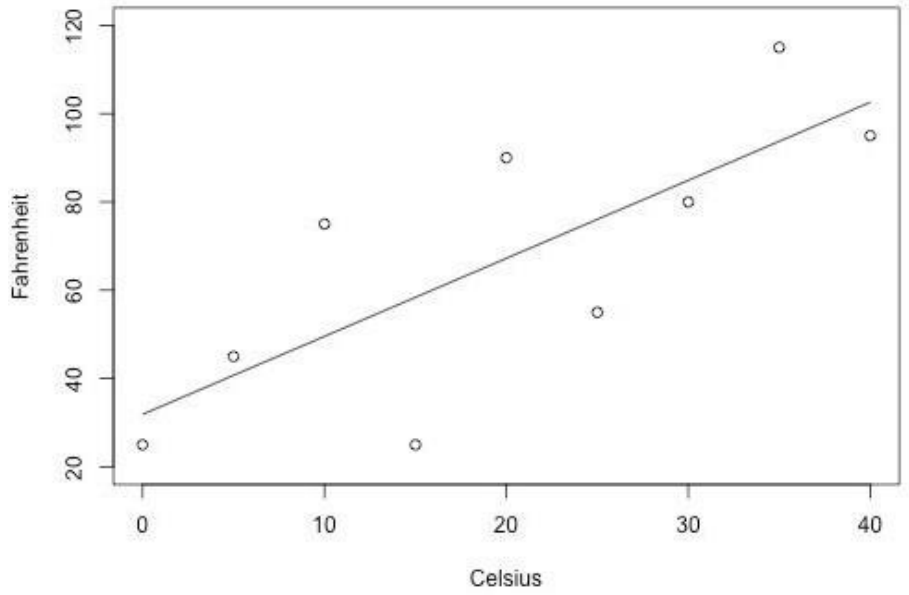
Variance is a measure of data dispersion: quantitatively, how far is every data point to the mean



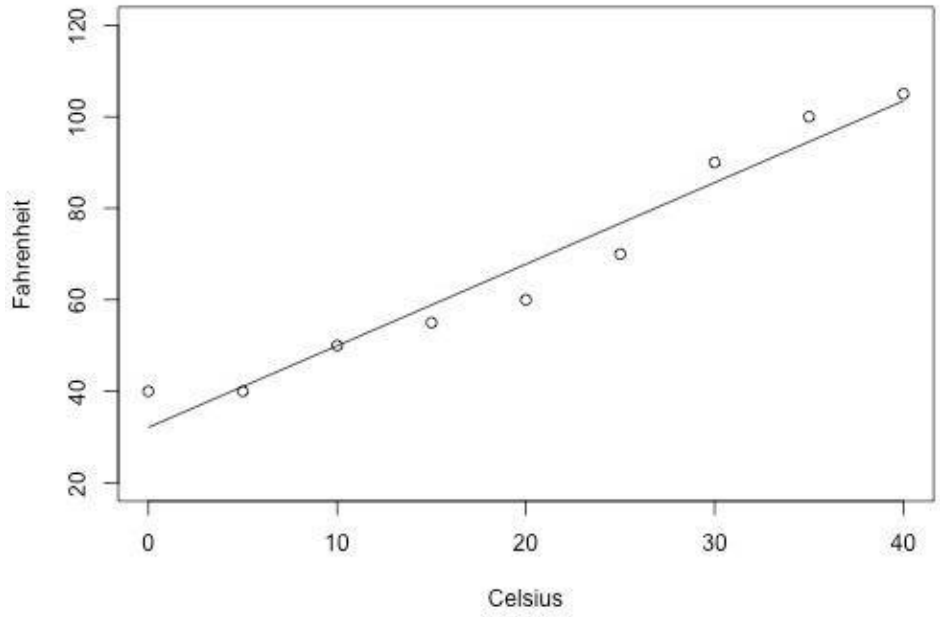
$$s^2 = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^2}{n-1}$$

Machine Learning - Simple Linear Regression

Loss function - Common Error Measurement
Example: Thermometer comparison



Thermometer A



Thermometer B

$$\mu_Y = E(Y) = \beta_0 + \beta_1 x$$

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

Machine Learning - Simple Linear Regression

Loss function - Common Error Measurement
Mean Squared Error

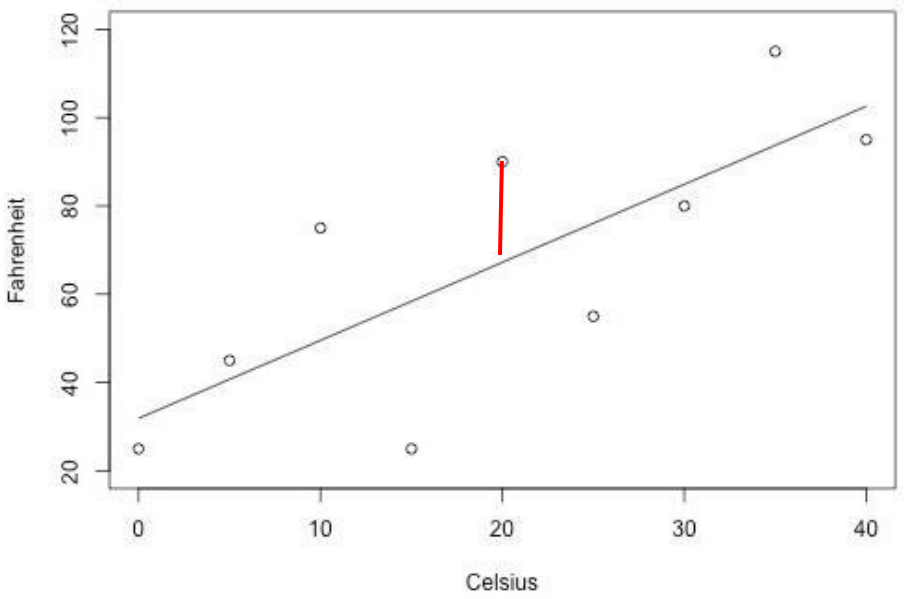
$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n-2}$$

As we are in regression settings, because predicted y to estimate μ_Y, we effectively estimate two parameters β_0 β_1 So we lose two degree of freedom

Beware MSE is sensitive to extreme value.

Root Mean Squared Error is a variant of MSE

$$RMSE = \sqrt{MSE}$$



Machine Learning - Simple Linear Regression

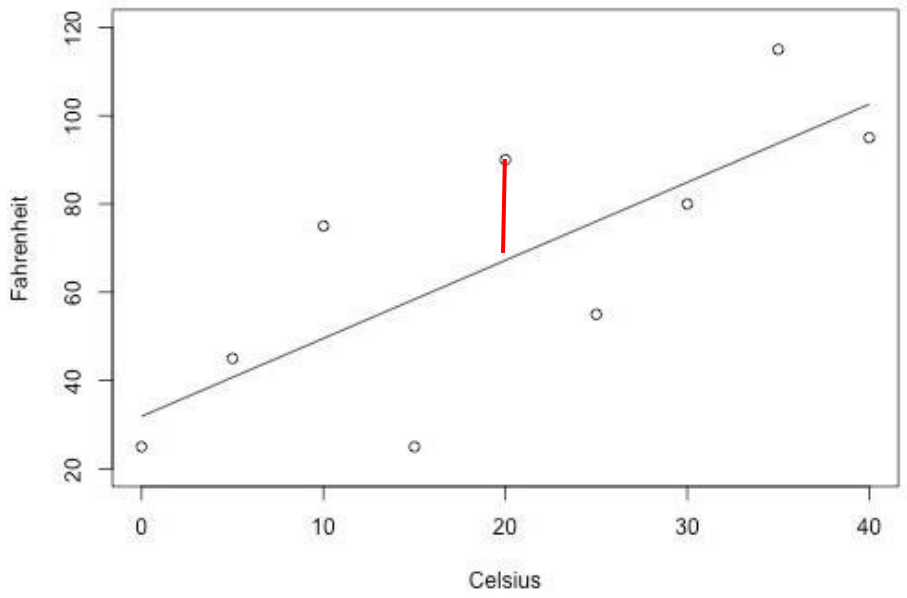
Loss function - Common Error Measurement

Coefficient of determination R^2

$$SSR = \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2$$

$$SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$SSTO = \sum_{i=1}^{n}(y_i - \bar{y})^2$$



- **SSR** is the "regression sum of squares" and quantifies how far the estimated sloped regression line, is from the horizontal "no relationship line," the sample mean.
- **SSE** is the "error sum of squares" and quantifies how much the data points, vary around the estimated regression line.
- **SSTO** is the "total sum of squares" and quantifies how much the data points, vary around their mean.

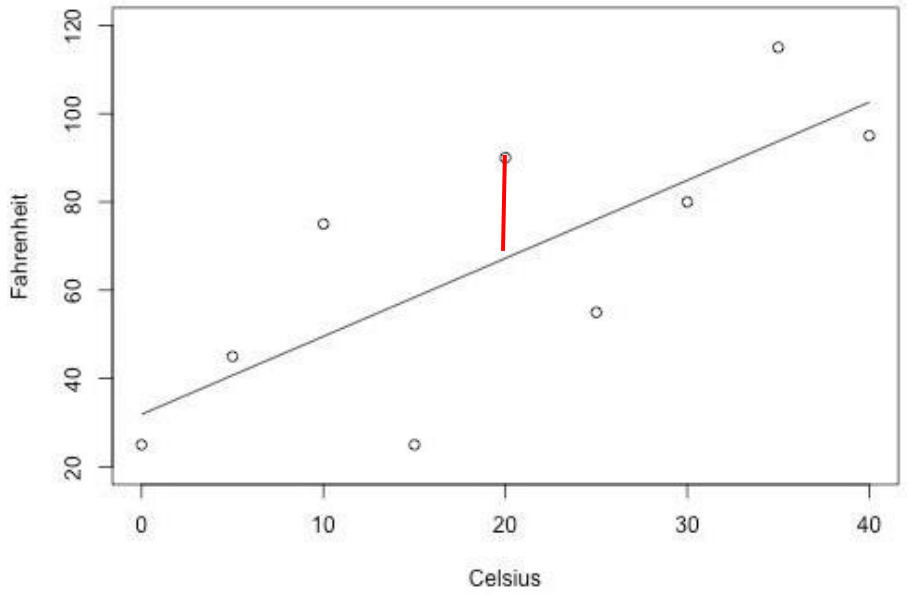
Machine Learning - Simple Linear Regression

Loss function - Common Error Measurement
Coefficient of determination R^2

$$SSR = \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2$$

$$SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$SSTO = \sum_{i=1}^{n}(y_i - \bar{y})^2$$



- SSR is the "regression sum of squares" and quantifies how far the estimated sloped regression line, is from the horizontal "no relationship line," the sample mean.
- SSE is the "error sum of squares" and quantifies how much the data points, vary around the estimated regression line.
- SSTO is the "total sum of squares" and quantifies how much the data points, vary around their mean.

Property: *SSTO = SSR + SSE*

Machine Learning - Simple Linear Regression

Loss function - Common Error Measurement

Coefficient of determination R^2

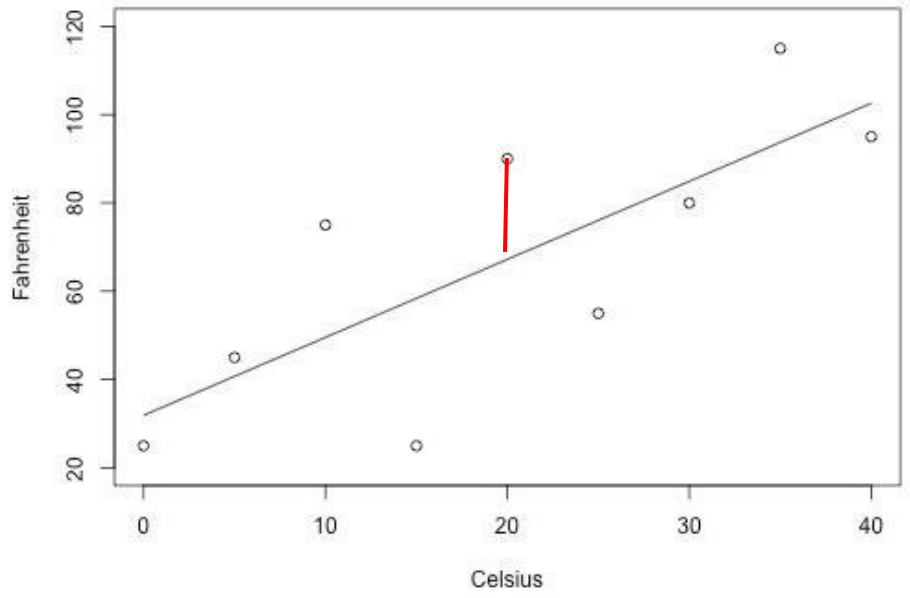
$$SSR = \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2$$

$$SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$SSTO = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

Property: *SSTO* = *SSR* + *SSE*

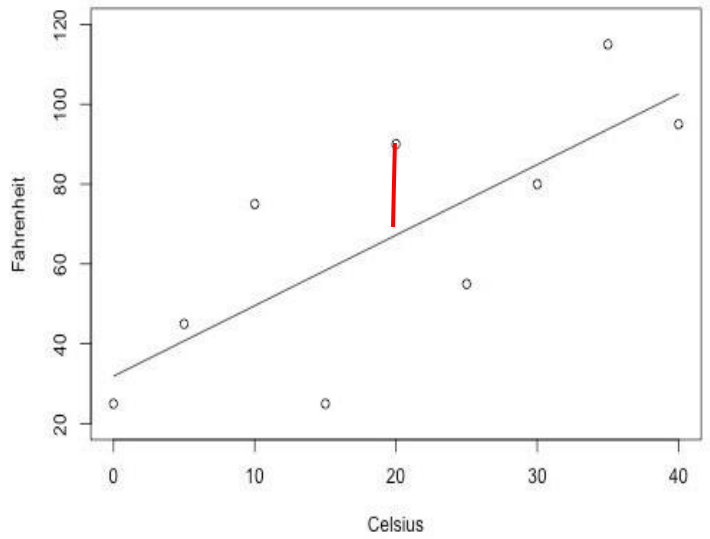
$$R^2 = \frac{SSR}{SSTO} = 1 - \frac{SSE}{SSTO}$$



Machine Learning - Simple Linear Regression

Loss function - Common Error Measurement
Coefficient of determination R^2
Property: *SSTO* = *SSR* + *SSE*

$$R^2 = \frac{SSR}{SSTO} = 1 - \frac{SSE}{SSTO}$$



- Since is a proportion, it is always a number between 0 and 1.
- If $R^2 = 1$, all of the data points fall perfectly on the regression line. The predictor *x* accounts for ***all*** of the variations in *y*!
- If $R^2 = 0$, the estimated regression line is perfectly horizontal. The predictor *x* accounts for ***none*** of the variations in *y*!
- Interpretation: **R^2 ×100 percent of the variation in *y* is 'explained by the variation in predictor *x*.**

Machine Learning - Simple Linear Regression

Loss function - Common Error Measurement

(Pearson) Correlation Coefficient r

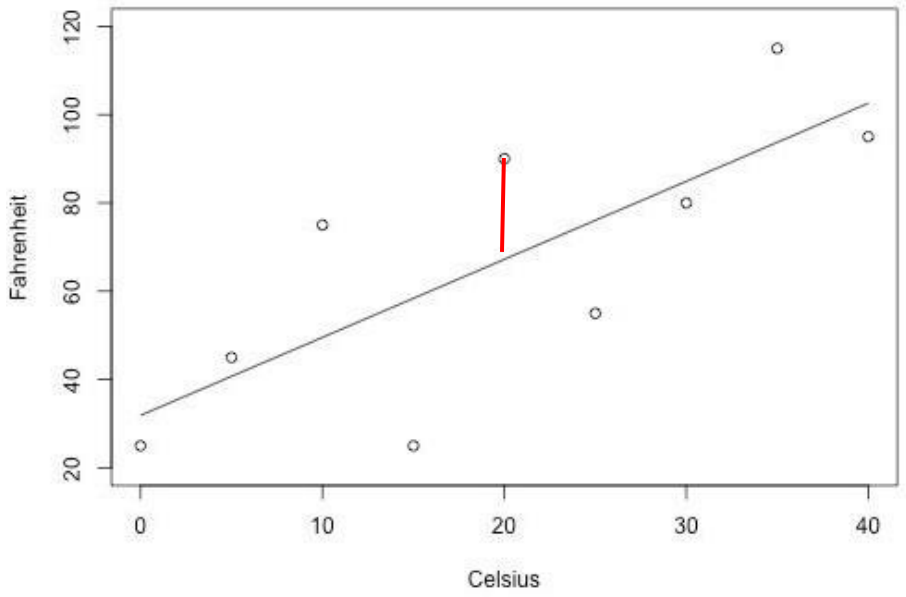
$$r = \pm\sqrt{R^2}$$

The sign of r depends on the sign of the estimated slope coefficient **a**:

- If **a** is negative, then r takes a negative sign.
- If **a** is positive, then r takes a positive sign.

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

$$r = \frac{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \times a$$



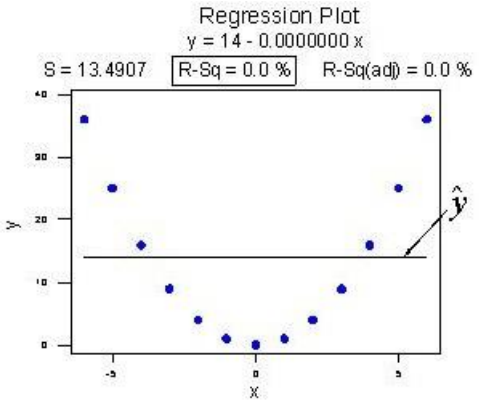
Machine Learning - Simple Linear Regression

Loss function - Common Error Measurement
Caution on R^2

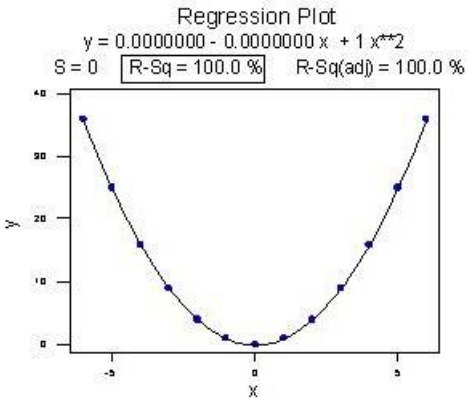
The coefficient of determination R^2 and the correlation coefficient *r* quantify the strength of a *linear* relationship. It is possible that $R^2 = 0\%$ and $r = 0$, suggesting there is no linear relation between *x* and *y*, and yet a perfect curved (or "curvilinear" relationship) exists.

They can be greatly affecting by one data point.

Correlation (or association) does not imply causation.



Pearson correlation of x and y = 0.000



Pearson correlation of x and y = 0.000

Machine Learning - Simple Linear Regression

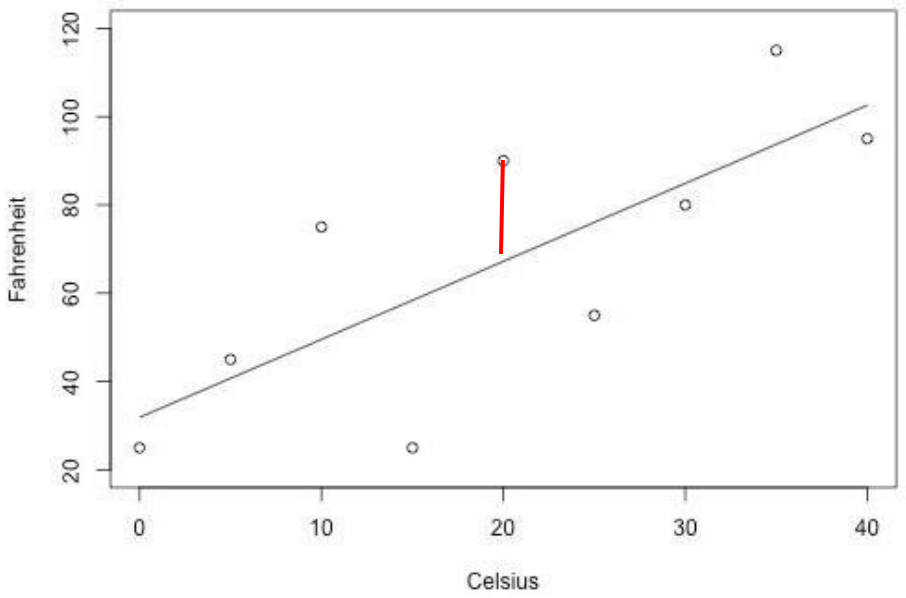
Loss function - Common Error Measurement

Mean Average Error

$$\mathrm{MAE}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} |y_i - \hat{y}_i| \,.$$

Easier to interpret

Less sensitive to extreme values, compared to r or R^2



4 | Minimizing Loss Function

Machine Learning - Simple Linear Regression

Loss function - How to find the best fitting line?
Let's minimize Mean Squared Error

(Math on the board)

Exact solution:

$$a = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

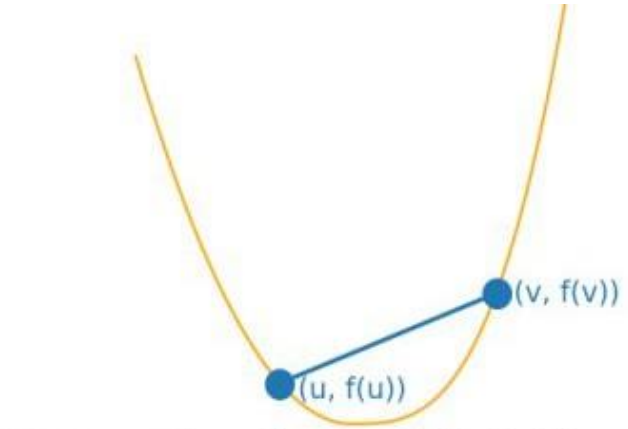
$$b = \bar{y} - a\bar{x}$$

Machine Learning - Simple Linear Regression

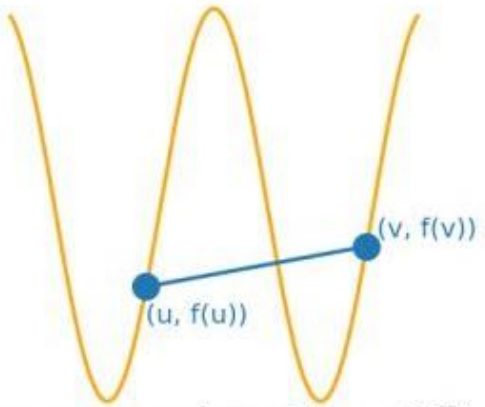
Loss function - How to find the best fitting line? Gradient Descent

Let's do Gradient Descent

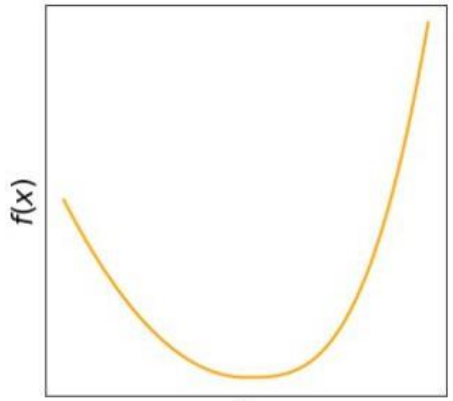
- Works if there is no exact solution (most cases)
- If the parametric function is differentiable
- Gives the minimum value when dealing with convex function. Otherwise, it gives local minimum



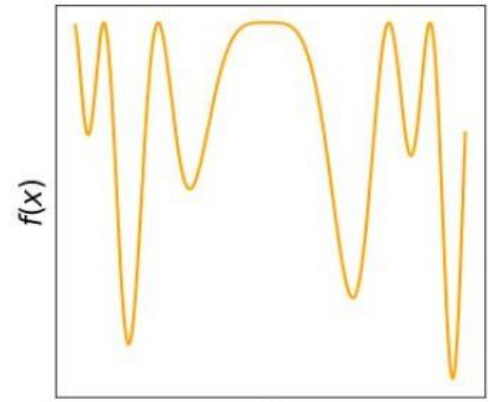
Convex function of $\mathbb{R} \rightarrow \mathbb{R}$



Non-convex function of $\mathbb{R} \rightarrow \mathbb{R}$



convex



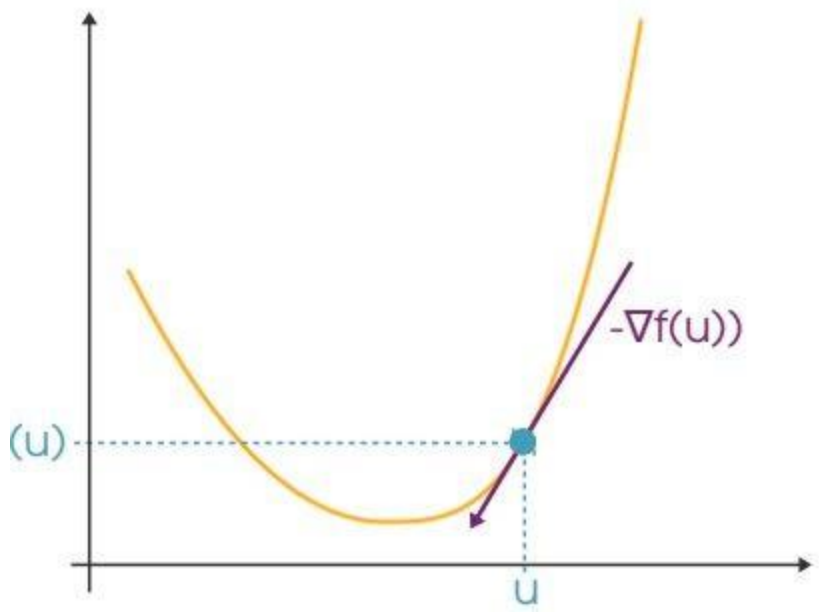
non-convex

Machine Learning - Simple Linear Regression

Loss function - How to find the best fitting line? Gradient Descent

- Start from a random point **u**
- **How do I get closer to the solution?**

  →Follow the opposite of the gradient (the gradient indicated the direction of steepest increase)

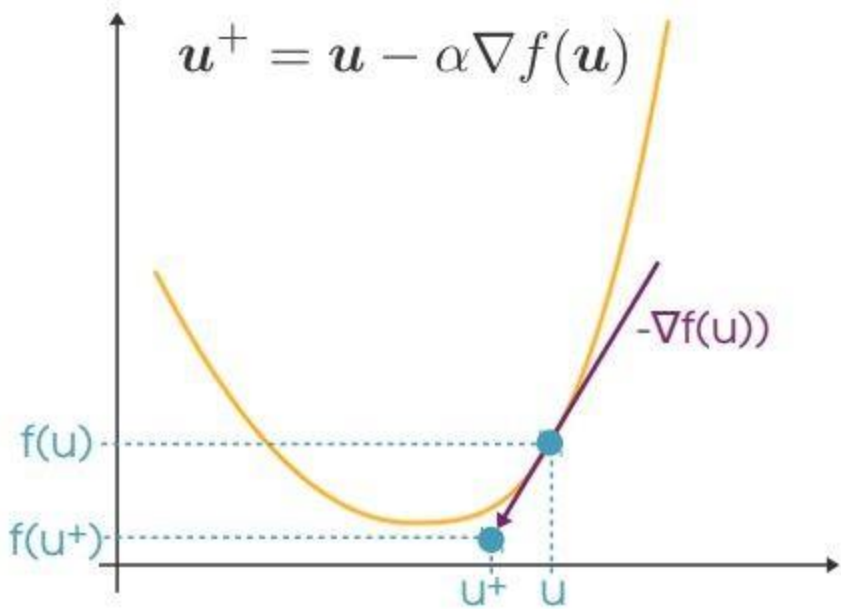


Machine Learning - Simple Linear Regression

Loss function - How to find the best fitting line? Gradient Descent

- Start from a random point **u**
- **How do I get closer to the solution?**

  →Follow the opposite of the gradient (the gradient indicated the direction of steepest increase)



Machine Learning - Simple Linear Regression

Loss function - How to find the best fitting line? Gradient Descent

- Start from a random point

$$\mathbf{u}^{(0)} \in \mathbb{R}$$

- Repeat for k=1,2,3…

$$\mathbf{u}^{(k)} = \mathbf{u}^{(k-1)} - \alpha_k \nabla f(\mathbf{u}^{(k-1)})$$

- Stop at some point

Machine Learning - Simple Linear Regression

Loss function - How to find the best fitting line? Gradient Descent

- Start from a random point

$$\mathbf{u}^{(0)} \in \mathbb{R}$$

- Repeat for k=1,2,3…

$$\mathbf{u}^{(k)} = \mathbf{u}^{(k-1)} - \alpha_k \nabla f\left(\mathbf{u}^{(k-1)}\right)$$

α_k Is the **step size**

- Stop at some point (**stopping criterion**)
  Usually: stop when

$$\left\| \nabla f\left(\mathbf{u}^{(k)}\right) \right\|^2 \leq \epsilon$$ With $$\epsilon = 10^{-m}$$ m could be 1,2,3,..

Machine Learning - Simple Linear Regression

Loss function - How to find the best fitting line? Gradient Descent

- Start from a random point
- Repeat for k=1,2,3…

$$\mathbf{u}^{(0)} \in \mathbb{R}$$

$$\mathbf{u}^{(k)} = \mathbf{u}^{(k-1)} - \alpha_k \nabla f(\mathbf{u}^{(k-1)})$$

α_k Is the **step size**

What is the problem of having a step too large ? too small ?

Machine Learning - Simple Linear Regression

Loss function - How to find the best fitting line? Gradient Descent

- Start from a random point
- Repeat for k=1,2,3…

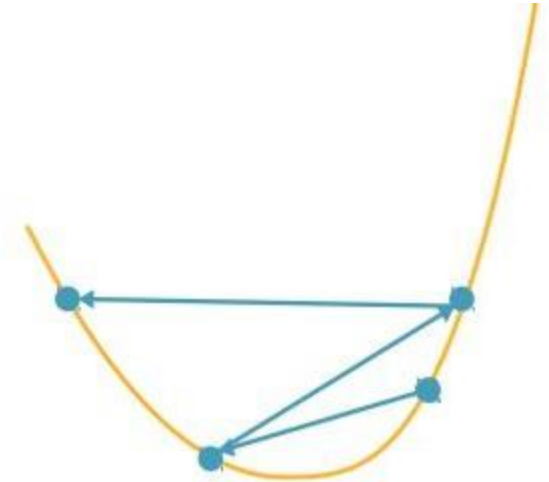
$$\mathbf{u}^{(0)} \in \mathbb{R}$$

$$\mathbf{u}^{(k)} = \mathbf{u}^{(k-1)} - \alpha_k \nabla f(\mathbf{u}^{(k-1)})$$

α_k Is the **step size**

What is the problem of having a step too large ? too small ?

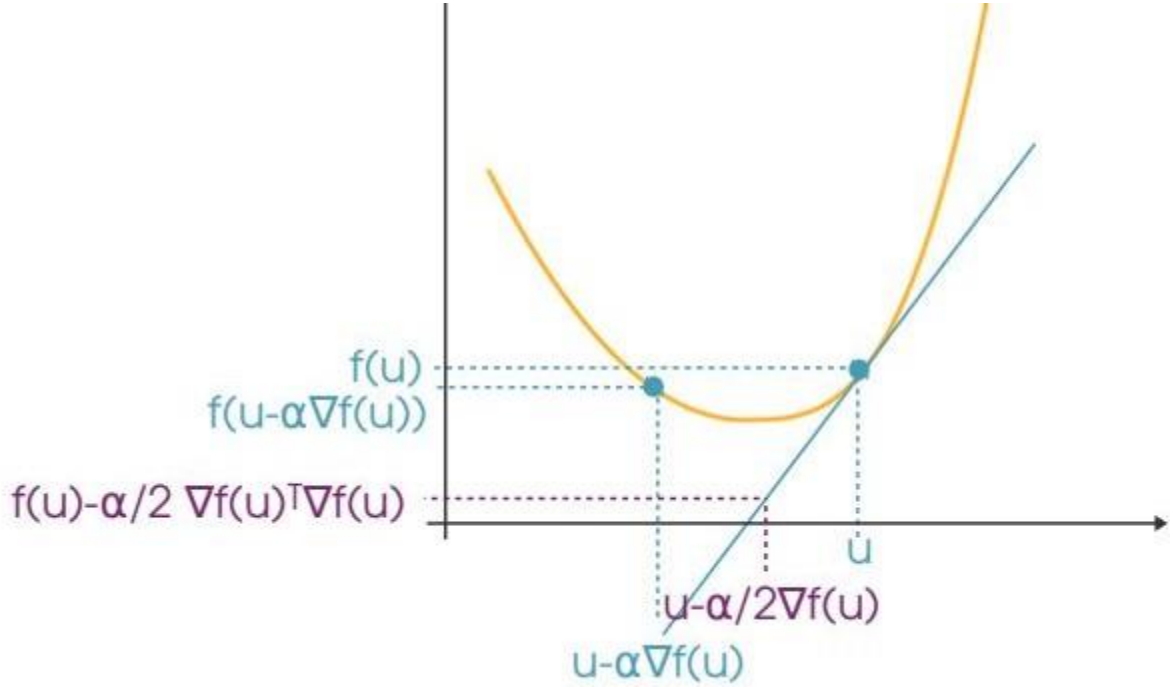
→ **Backtracking line search** makes it possible to choose the step size **adaptively**

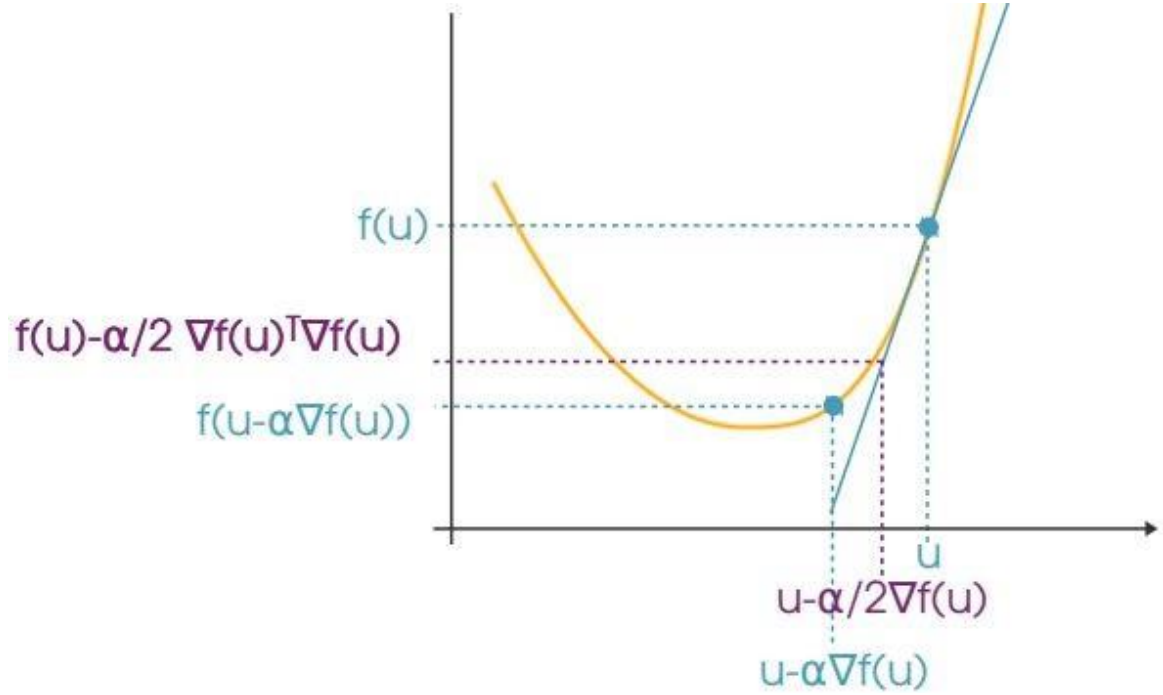


Machine Learning - Simple Linear Regression

Loss function - How to find the best fitting line? Gradient Descent

→ **Backtracking line search** makes it possible to choose the step size **adaptively**





Criterion on step size

$$f(u-\alpha\nabla f(u)) \leq f(u)-\alpha/2\ \nabla f(u)^T\nabla f(u)$$

Machine Learning - Simple Linear Regression

Loss function - How to find the best fitting line? Gradient Descent

- Shrinking parameter $0 < \beta < 1$, initial step size α_0

- Start from a random point $\mathbf{u}^{(0)} \in \mathbb{R}$
-
- Repeat for k=1,2,3…

  If $f(\mathbf{u}^{(k-1)} - \alpha_{k-1}\nabla f(\mathbf{u}^{(k-1)})) > f(\mathbf{u}^{(k-1)}) - \frac{1}{2}\alpha_{k-1}\left\|\nabla f(\mathbf{u}^{(k-1)})\right\|_2^2$

  Shrink the step size $\alpha_k = \beta\alpha_{k-1}$

  Otherwise: $\alpha_k = \alpha_{k-1}$

  Update $\mathbf{u}^{(k)} = \mathbf{u}^{(k-1)} - \alpha_k\nabla f(\mathbf{u}^{(k-1)})$

α_k Is the **step size**

- Stop at some point (**stopping criterion**)

  Usually: stop when $\left\|\nabla f(\mathbf{u}^{(k)})\right\|^2 \leq \epsilon$ With $\epsilon = 10^{-m}$ m could be 1,2,3,..

Machine Learning - Simple Linear Regression

Loss function - How to find the best fitting line? Gradient Descent
→ **Backtracking line search** makes it possible to choose the step size **adaptively**

Other well known techniques to adapt step size: **Conjugate gradient method**, **Newton method**, **BFGS** or **LBFGS**

Machine Learning - Simple Linear Regression

Recap

We look at our first decision function to solve the Simple Regression problem: Simple Linear Regression.

We present several loss functions (or error estimations): MSE, RMSE, R^2, pearson r, MAE.

We present 2 techniques to choose the best “fitted-line”:

- One exact solution by minimizing MSE (we get lucky in this simple problem)
- One working in most cases with gradient descent

Let’s implement all of this in a notebook!

To go further, you can look at Scikit-Learn documentation!