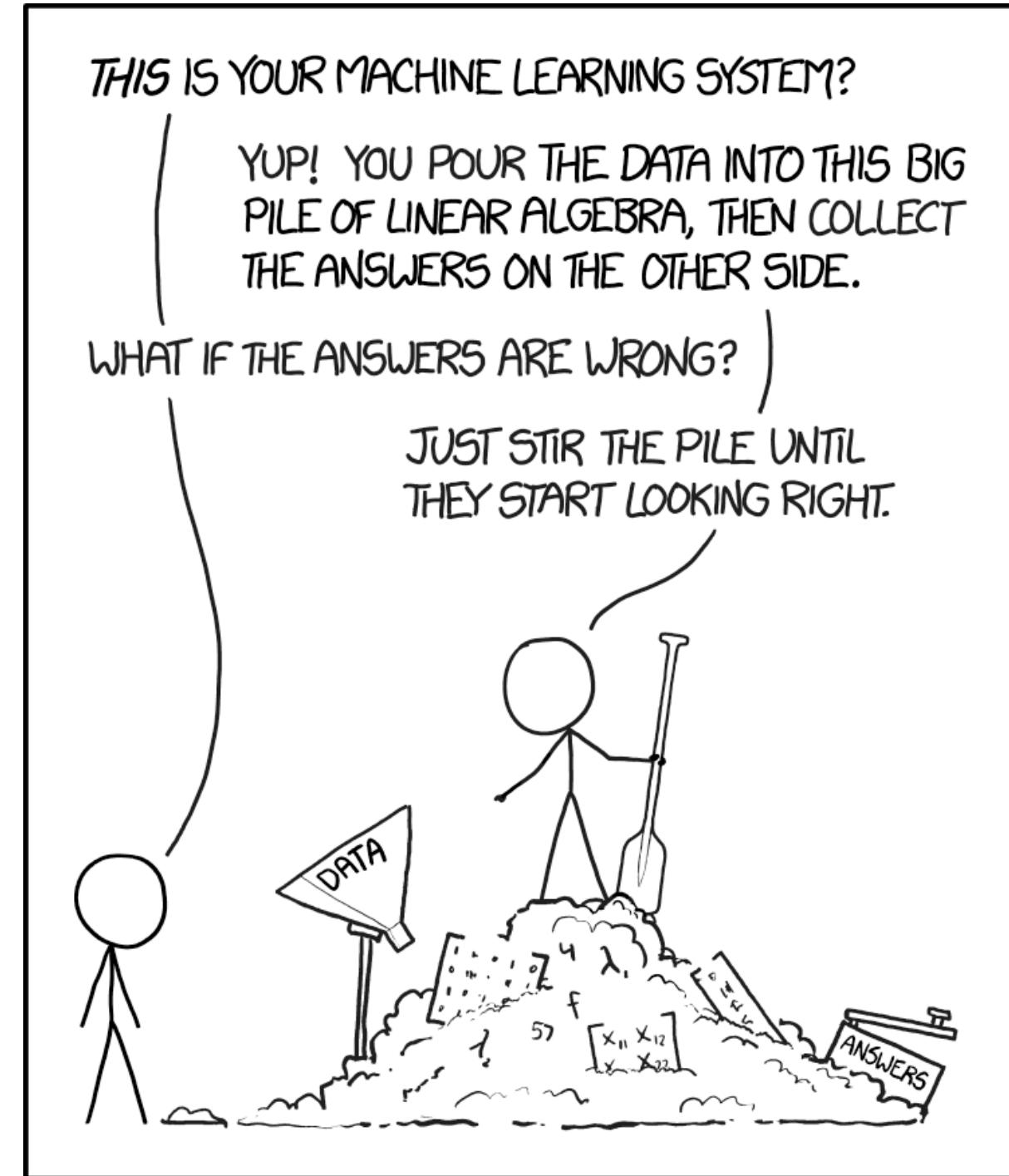


# Machine Learning



From [xkcd](#)

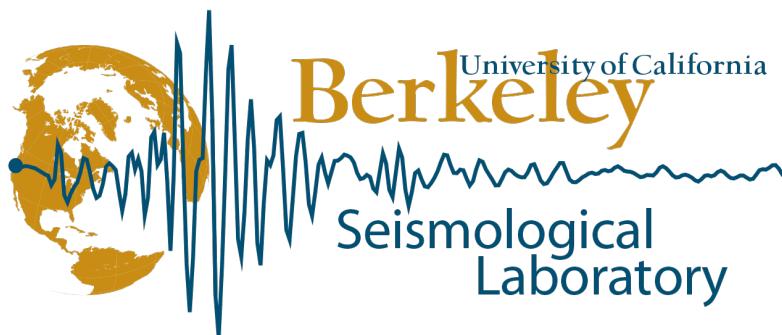
UC Berkeley  
Doctoral



Career Development Initiative  
for the Physical Sciences

# Machine learning - using scikit-learn

Qingkai Kong  
2017-06-28



<http://seismo.berkeley.edu/qingkaikong/>

[https://github.com/qingkaikong/20170628\\_ML\\_sklearn](https://github.com/qingkaikong/20170628_ML_sklearn)

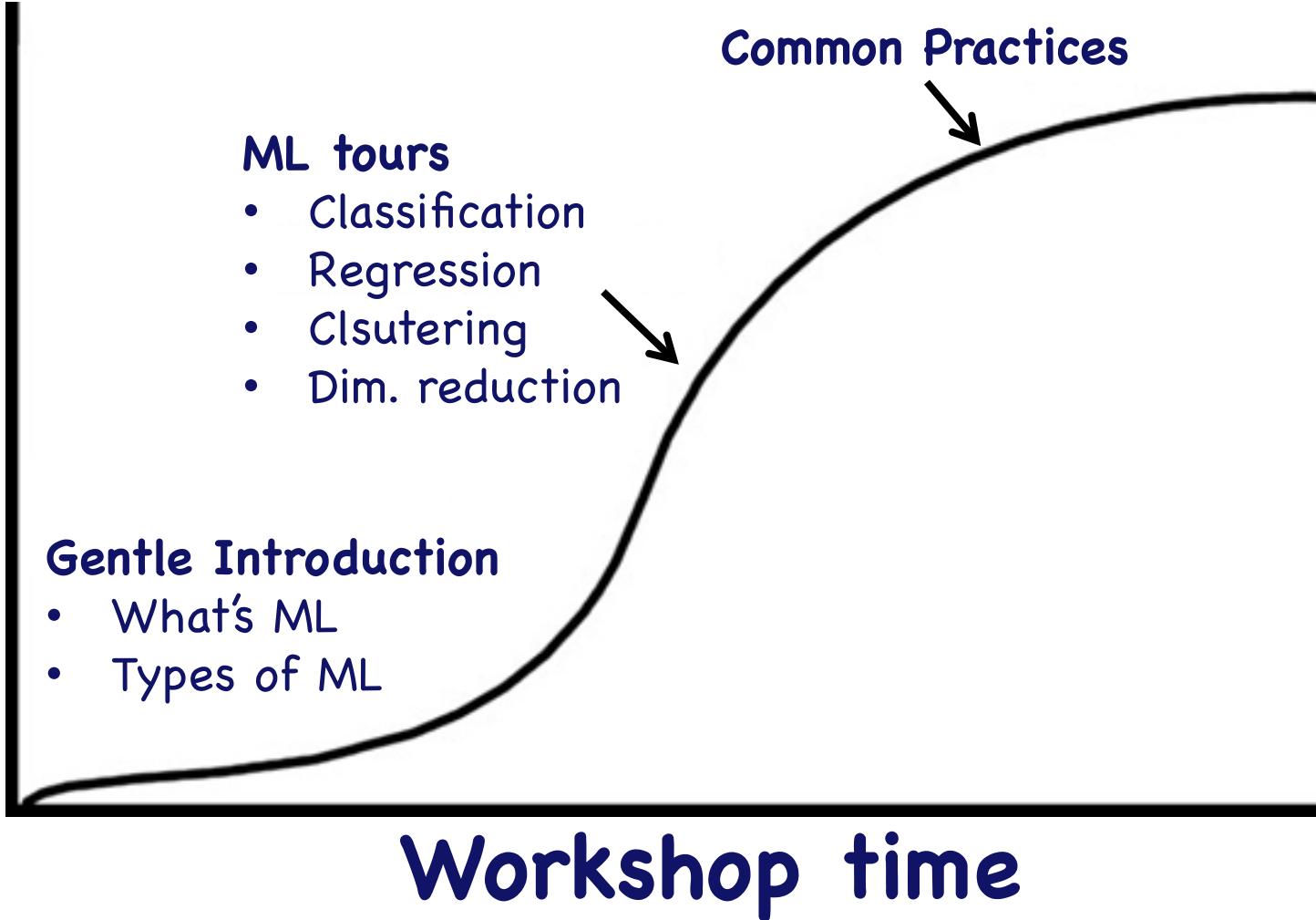
The screenshot shows a GitHub repository page. At the top, the repository name 'qingkaikong / 20170628\_ML\_sklearn' is displayed, along with metrics: 2 watches, 1 star, and 0 forks. Below the header, there are navigation links for 'Code', 'Issues 0', 'Pull requests 0', 'Projects 0', 'Wiki', 'Settings', and 'Insights'. The main content area is titled 'Workshop material for 2017 CDIPS Data Science Workshop' and includes an 'Edit' button. A 'Add topics' link is also present. Below this, a summary bar shows '15 commits', '1 branch', '0 releases', and '1 contributor'. At the bottom of the page are buttons for 'Branch: master ▾', 'New pull request', 'Create new file', 'Upload files', 'Find file', and a prominent green 'Clone or download ▾' button.

## After downloads

The screenshot shows a terminal window on a Mac OS X system. The title bar indicates the window is for '20170628\_ML\_sklearn — bash — 80x24'. The terminal output shows the user navigating to their home directory, cloning the repository, and activating a virtual environment:

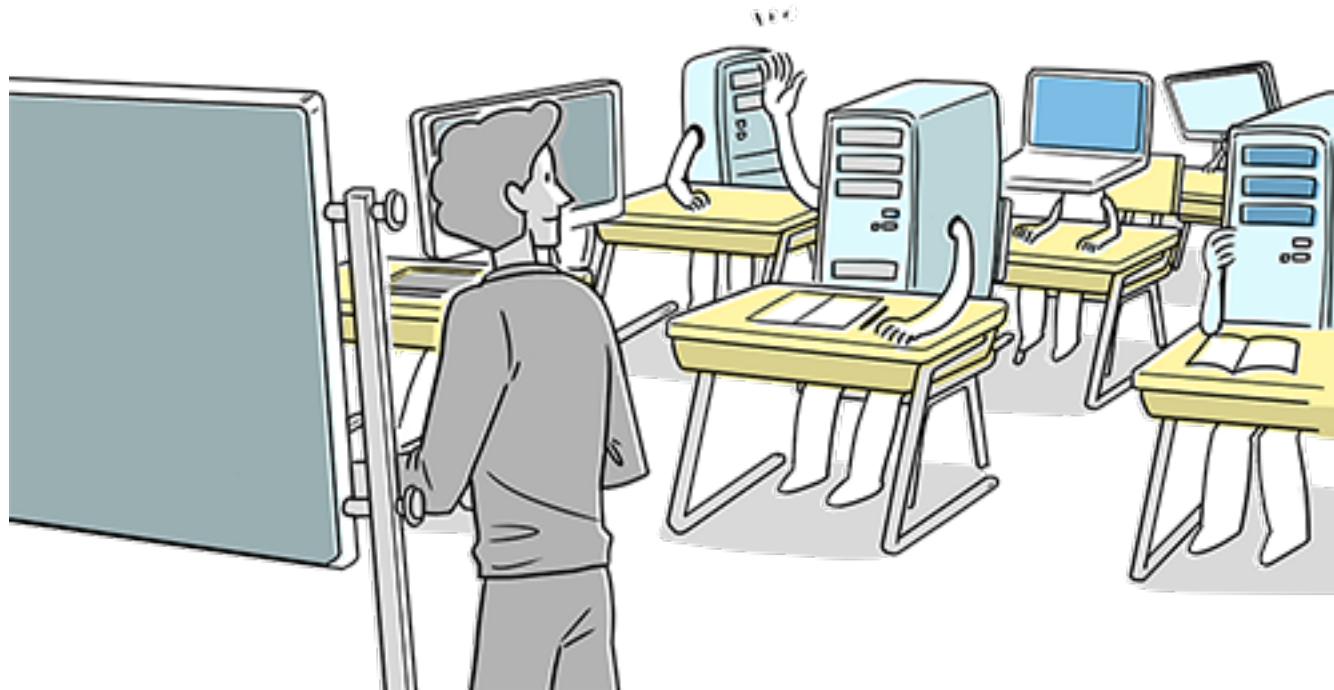
```
[~/Research/workshops_I_teach/20170628_ML_sklearn]
[20:09:49 qingkaikong]$pwd
/Users/qingkaikong/Research/workshops_I_teach/20170628_ML_sklearn
[~/Research/workshops_I_teach/20170628_ML_skLearn]
[20:09:51 qingkaikong]$source activate cdips2017
```

# Learning curve



[https://github.com/qingkaikong/20170628\\_ML\\_sklearn](https://github.com/qingkaikong/20170628_ML_sklearn)

# What is machine learning?



[https://github.com/qingkaikong/20170628\\_ML\\_sklearn](https://github.com/qingkaikong/20170628_ML_sklearn)

Data  
examples

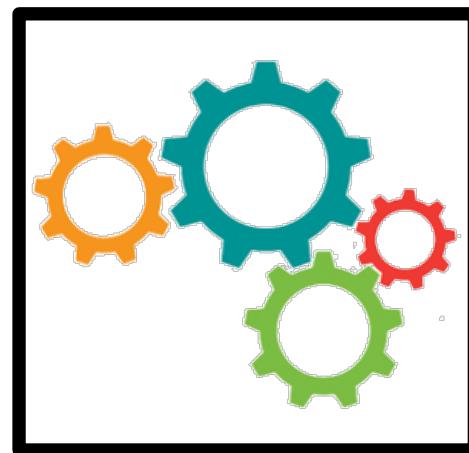
01100  
10110  
11110

Optimization  
algorithm



# Pipeline of training a machine learning model.

Tunable  
Model



Trained  
Model

MAKE  
THINGS  
HAPPEN!



**amazon.com**

**Recommended for You**

Amazon.com has new recommendations for you based on items you purchased or told us you own.

---

 <a href="#">Google Apps Deciphered: Compute in the Cloud to Streamline Your Desktop</a>	 <a href="#">Google Apps Administrator Guide: A Private-Label Web Workspace</a>	 <a href="#">Googlepedia: The Ultimate Google Resource (3rd Edition)</a>
---	--	---

**Self-driving car  
Voice recognition**

...

[https://github.com/qingkaikong/20170628\\_ML\\_sklearn](https://github.com/qingkaikong/20170628_ML_sklearn)

Not always  
working

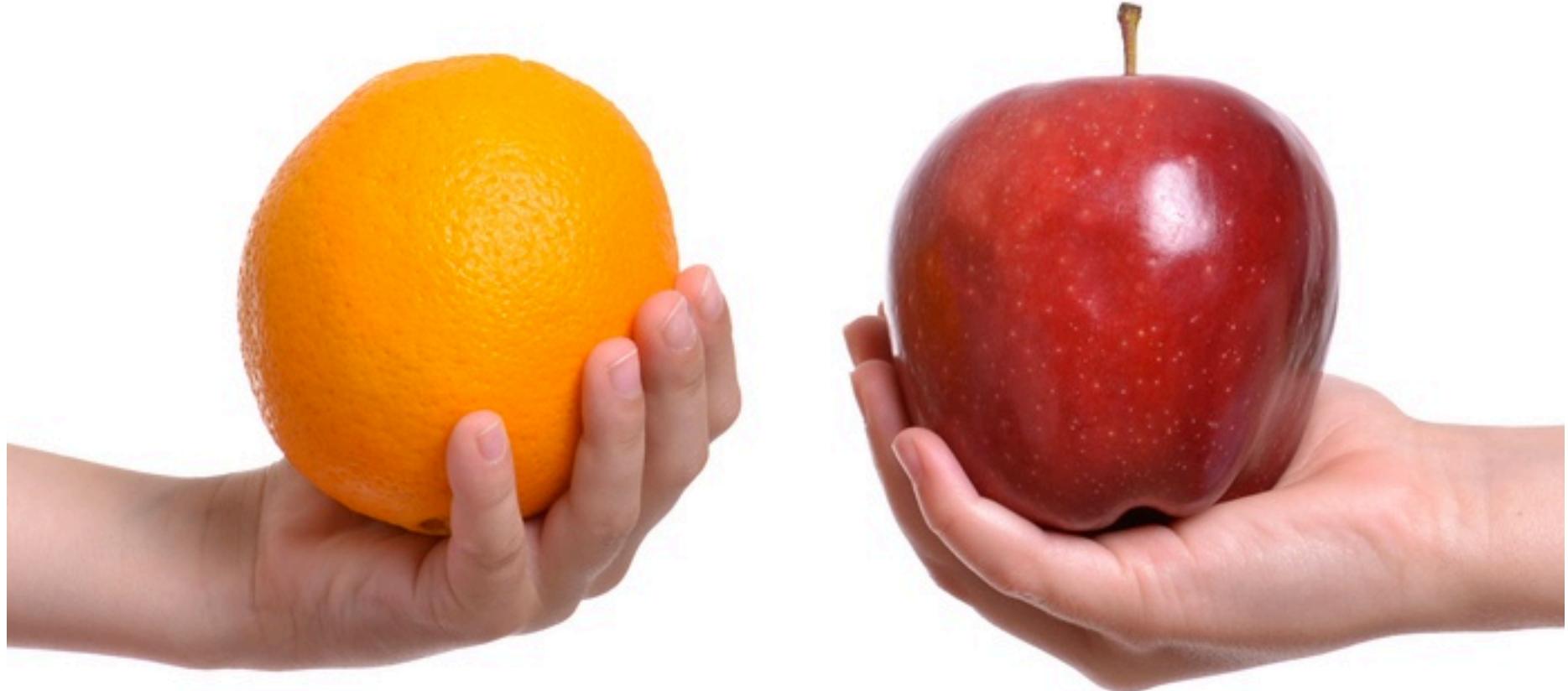


# Common types of machine learning

Supervised  
learning

Unsupervised  
learning

# Supervised learning





Unsupervised learning

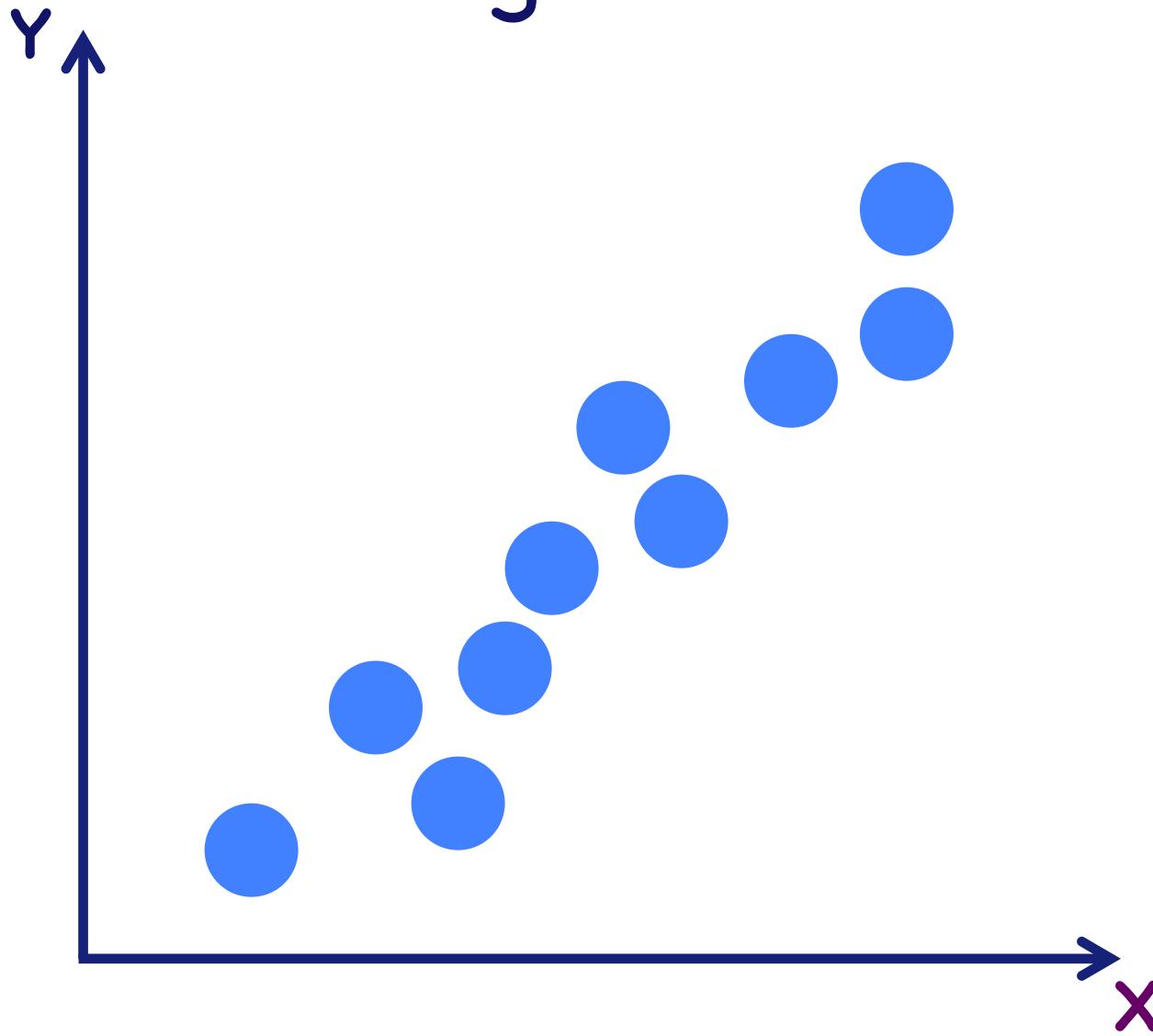


Supervised  
learning

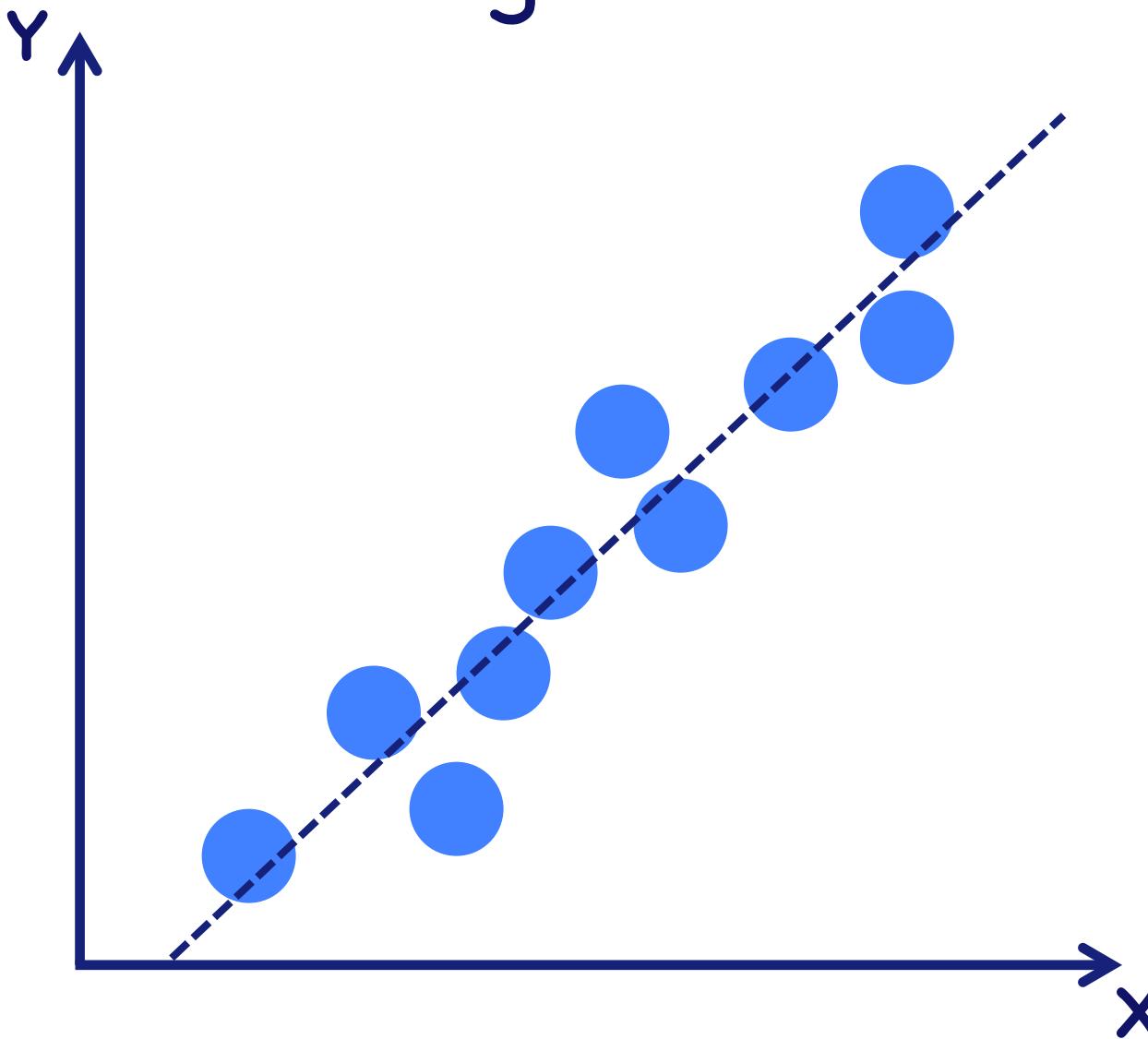
Regression

Classification

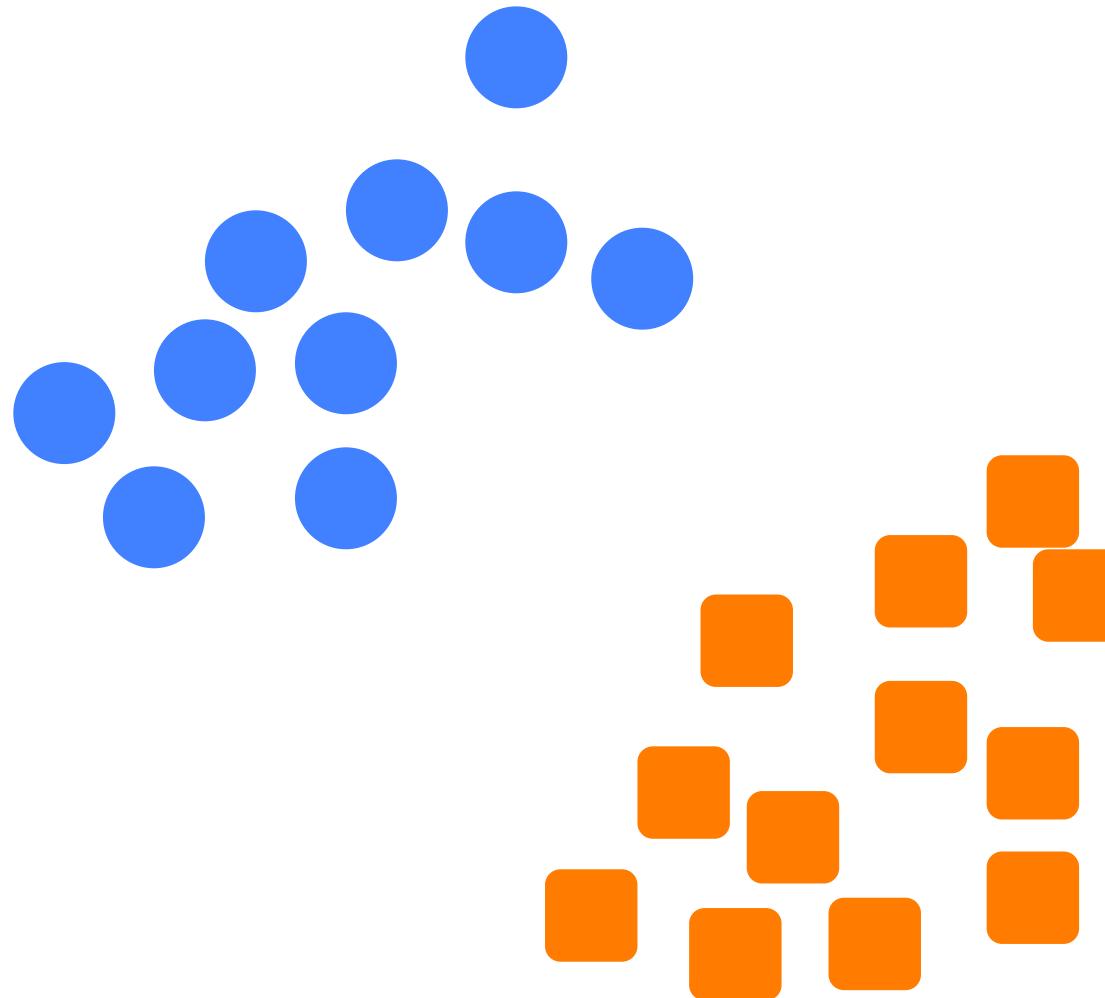
# Regression



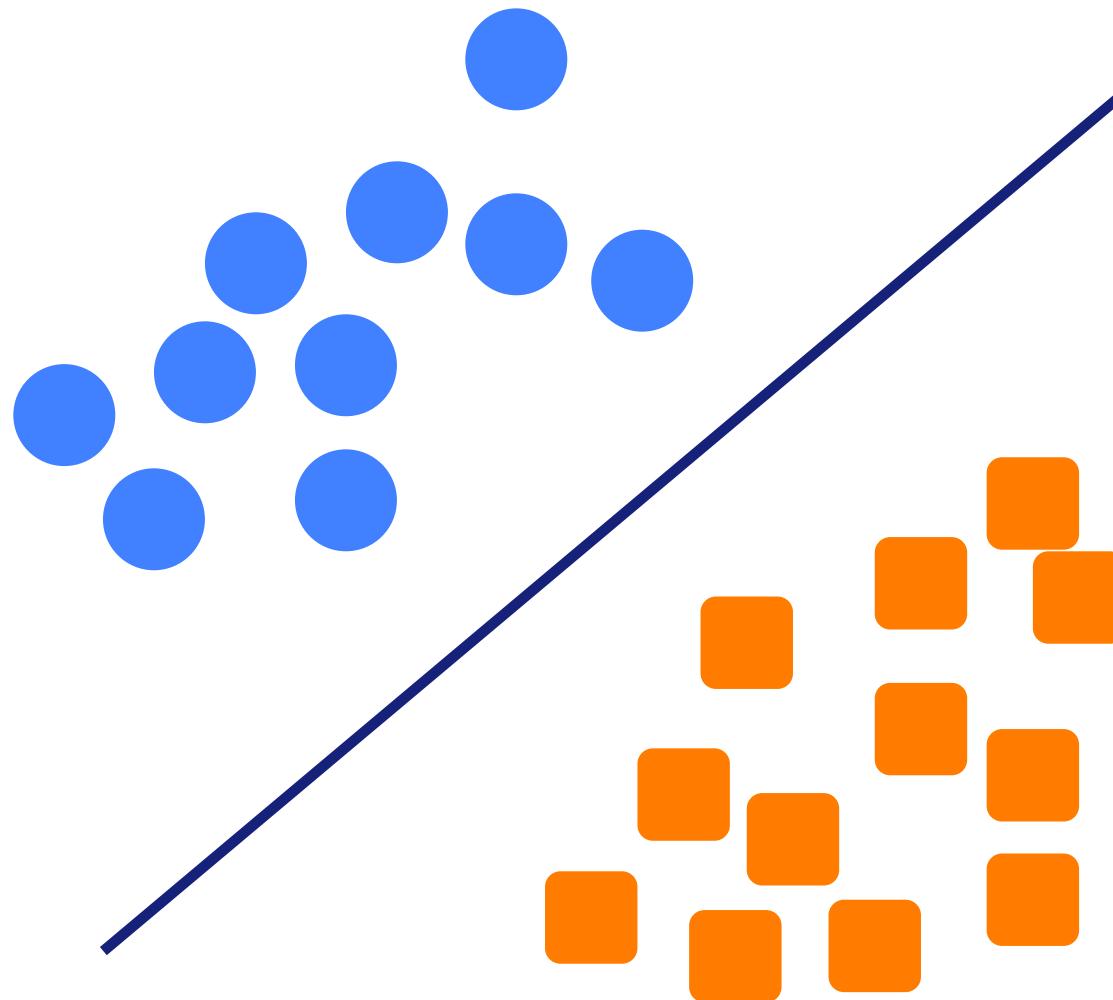
# Regression



# Classification



# Classification



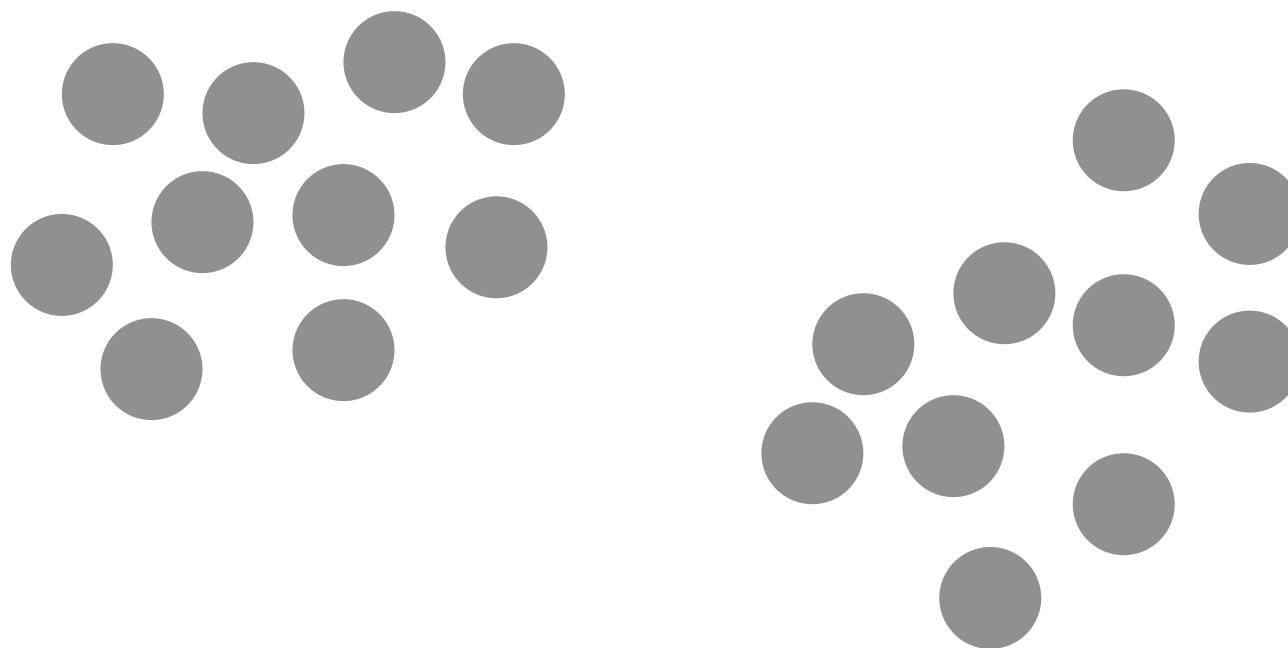


Unsupervised  
learning

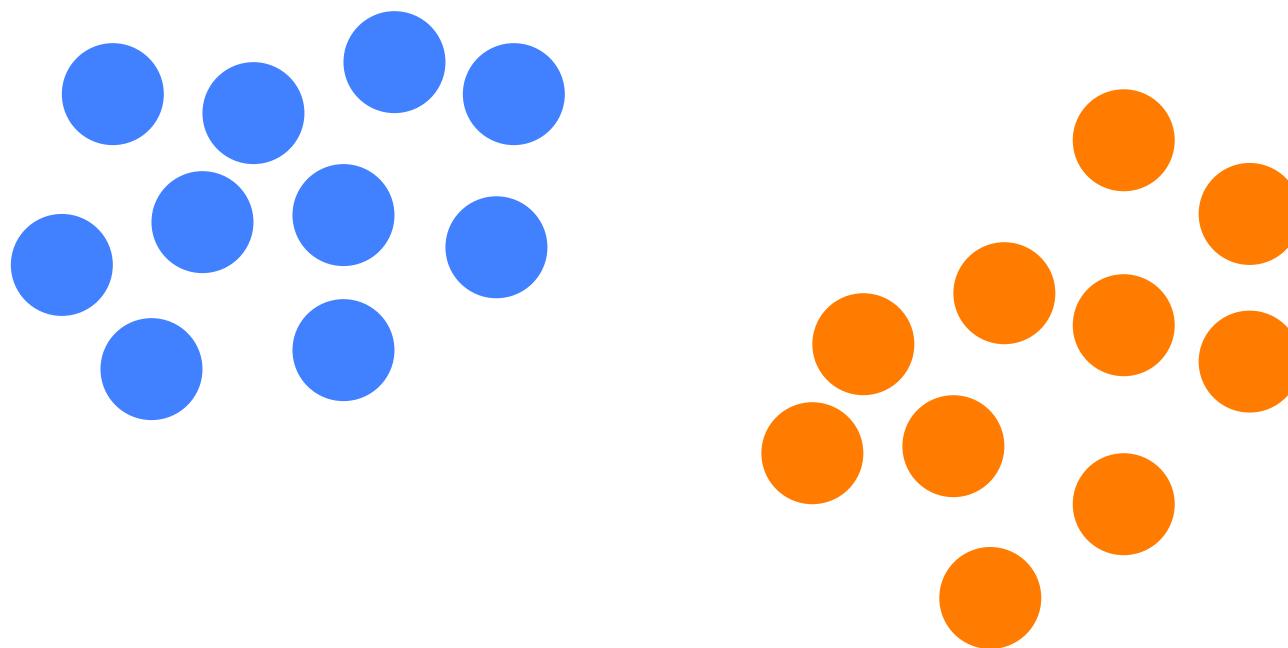
Clustering

Dimensionality  
reduction

# Clustering



# Clustering



# Dimensionality reduction

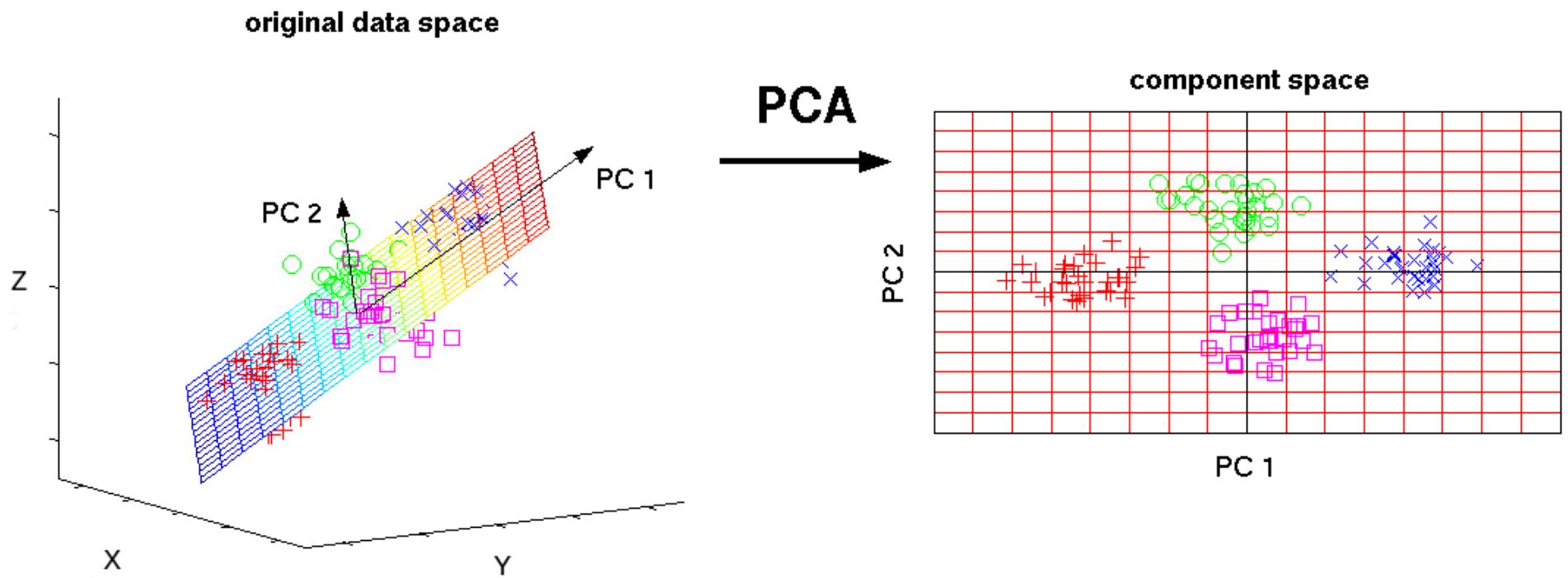


Figure from <https://stats.stackexchange.com/questions/183236/what-is-the-relation-between-k-means-clustering-and-pca>

Data  
examples

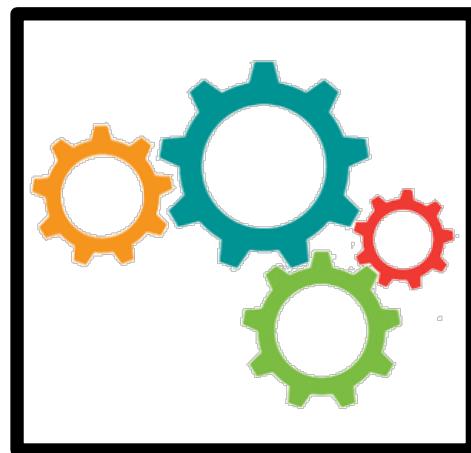
01100  
10110  
11110

Optimization  
algorithm



# Pipeline of training a machine learning model.

Tunable  
Model



Trained  
Model

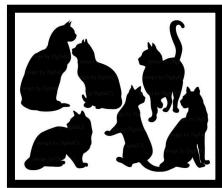
MAKE  
THINGS  
HAPPEN!

# Representation of data

## Raw data



Documents



Images



Numbers

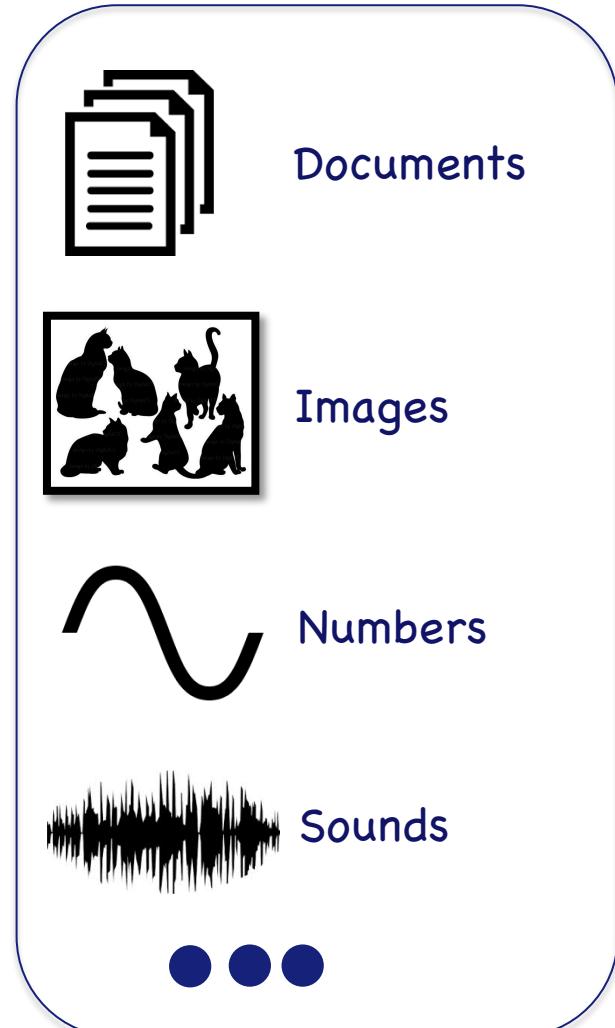


Sounds

• • •

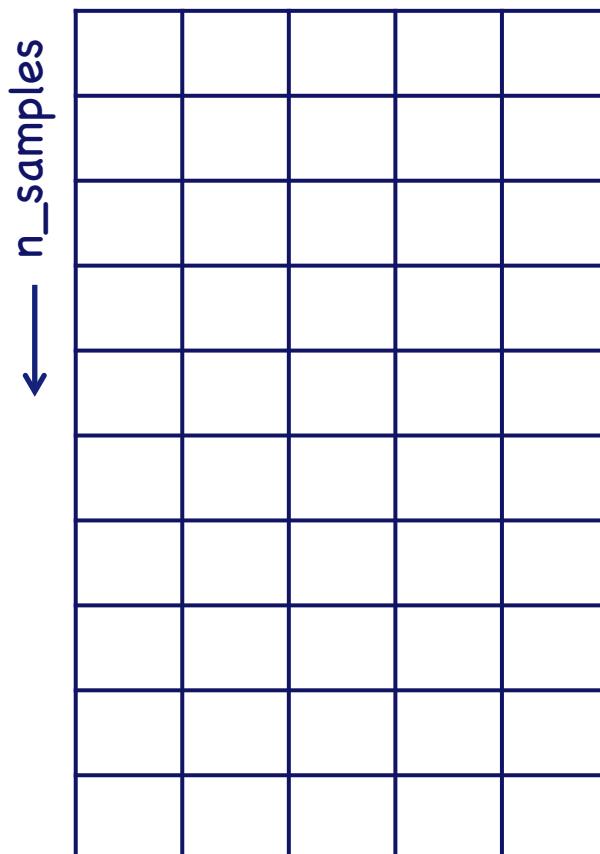
# Representation of data

# Raw data



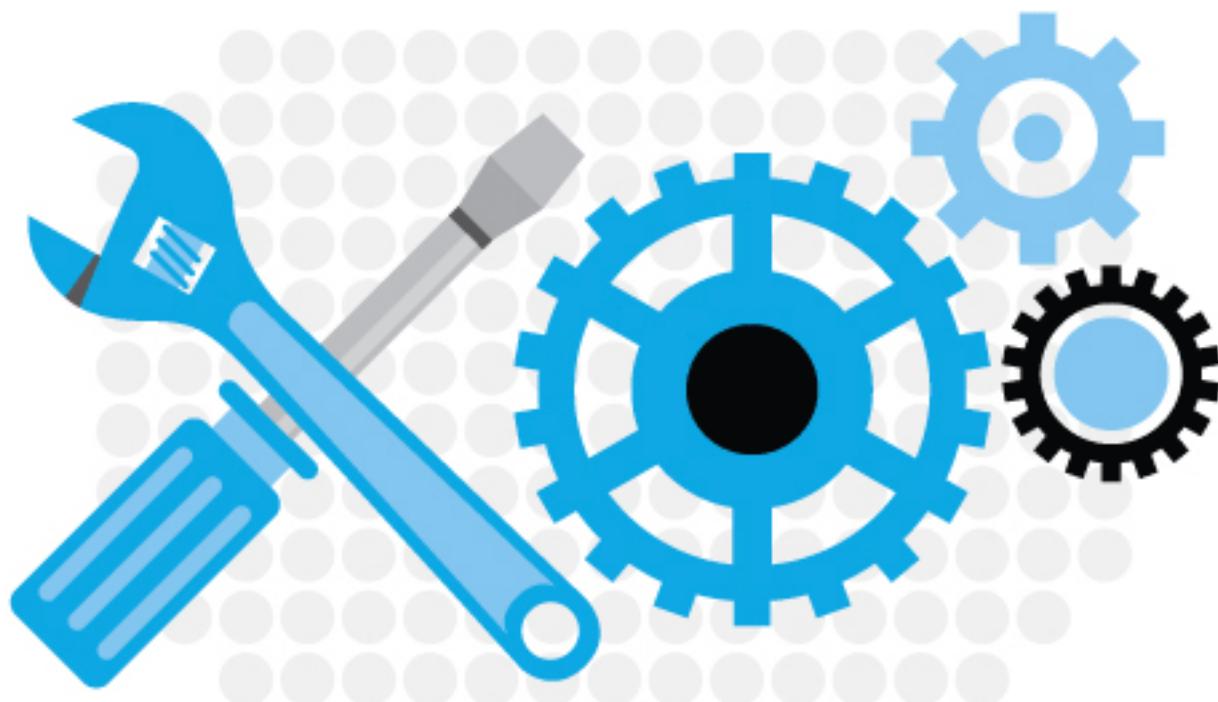
# Feature matrix ( $X$ )

n\_features →

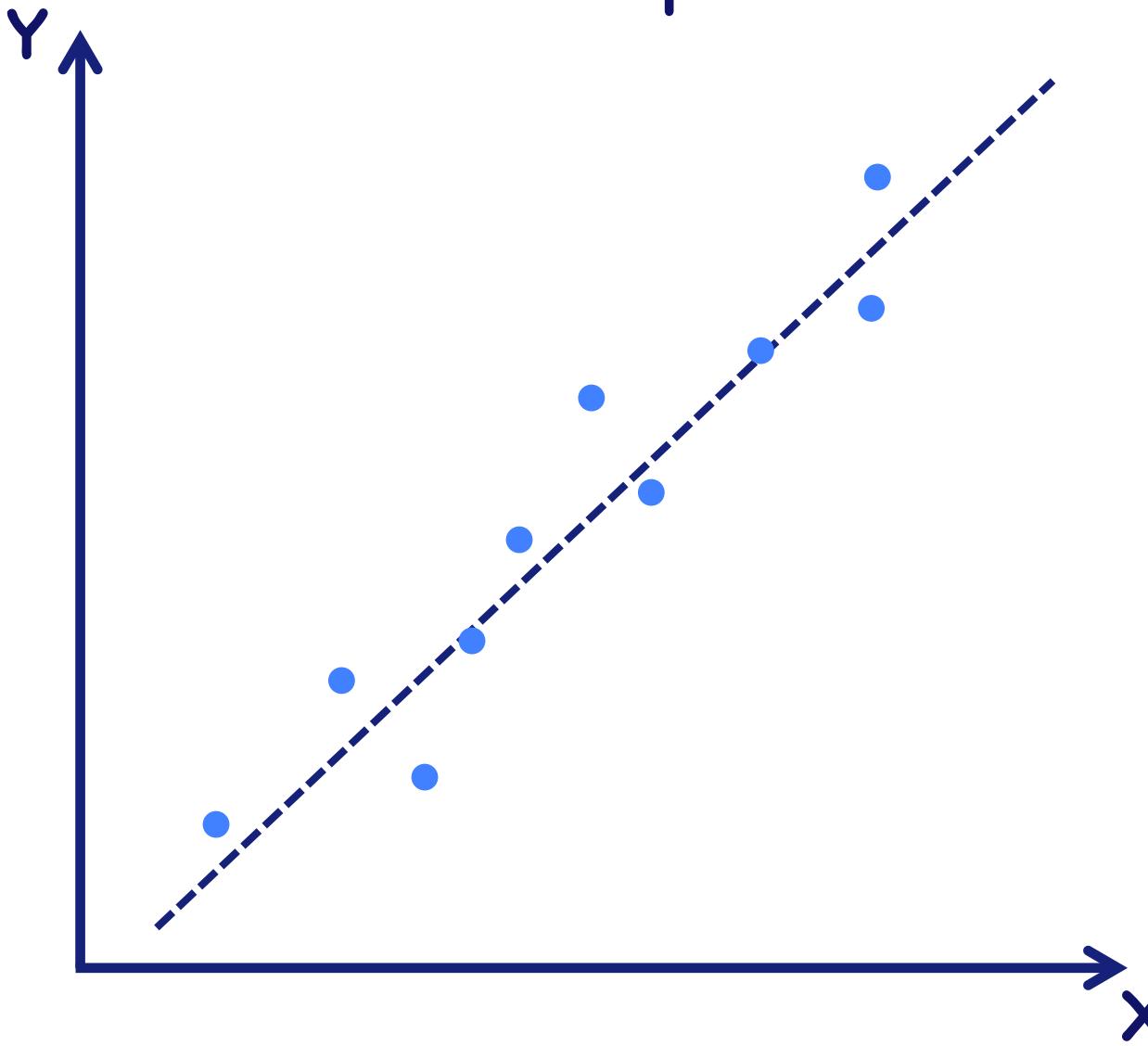


# Target (y)

# Optimization

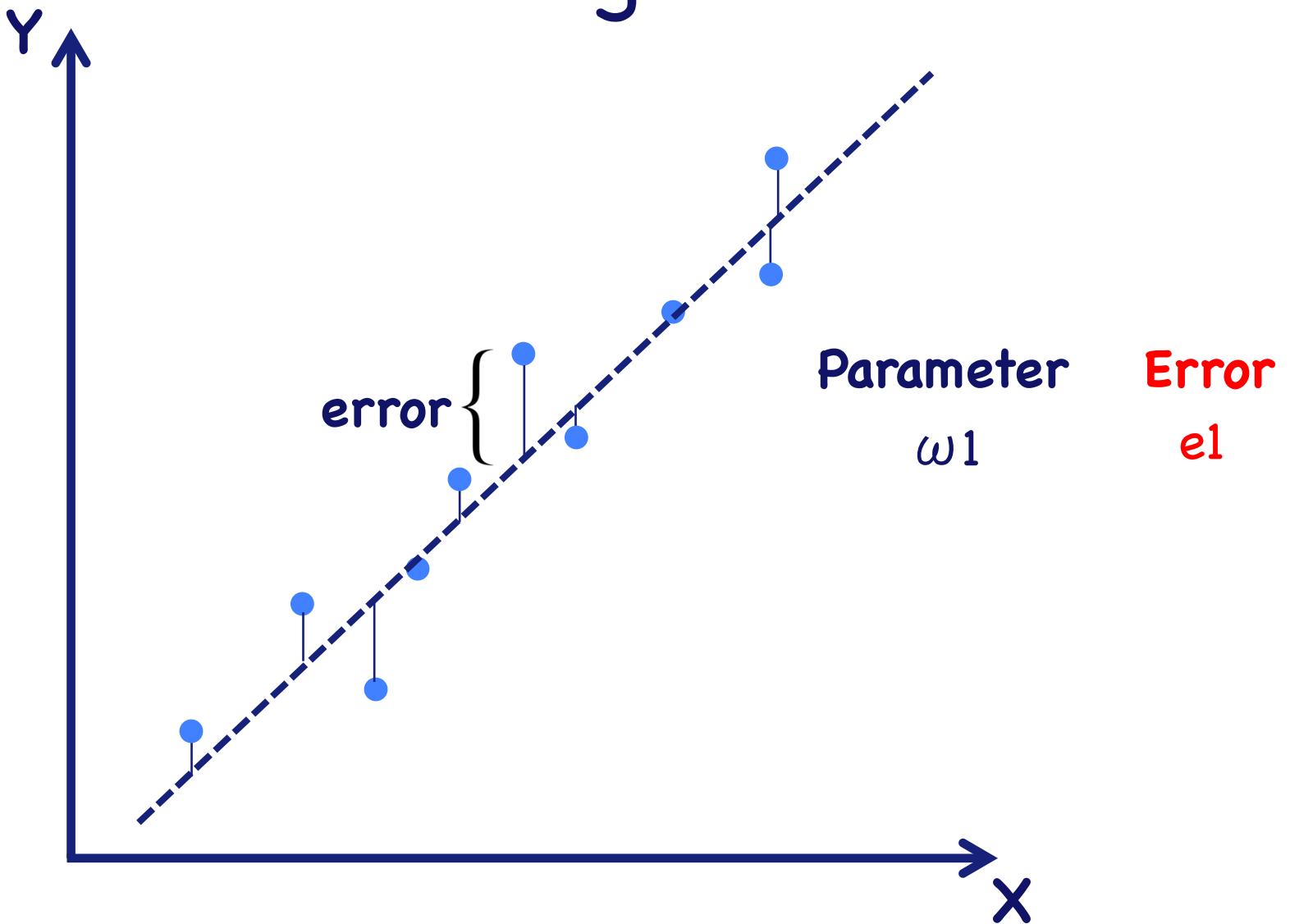


# How we optimize?

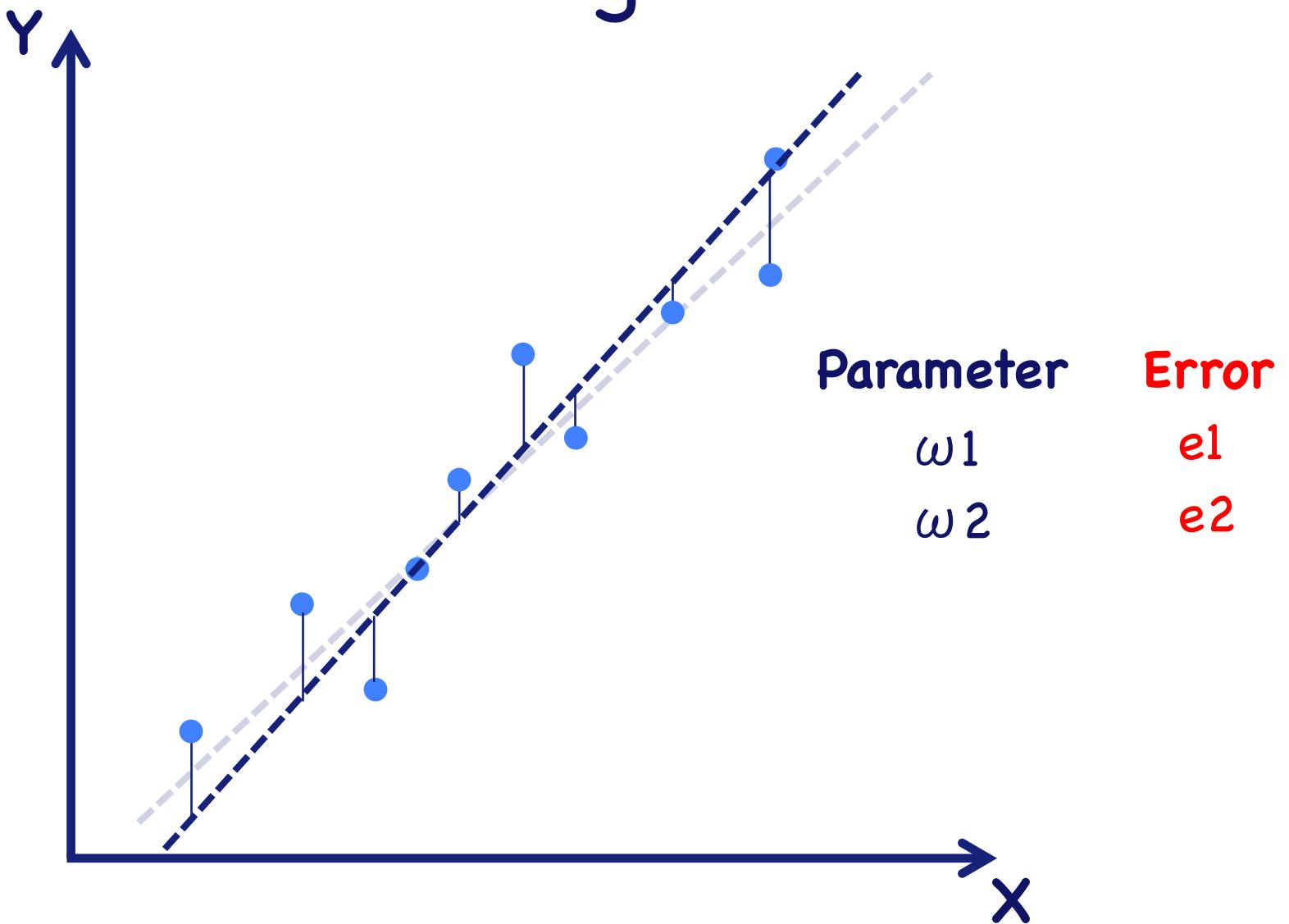


A simple example

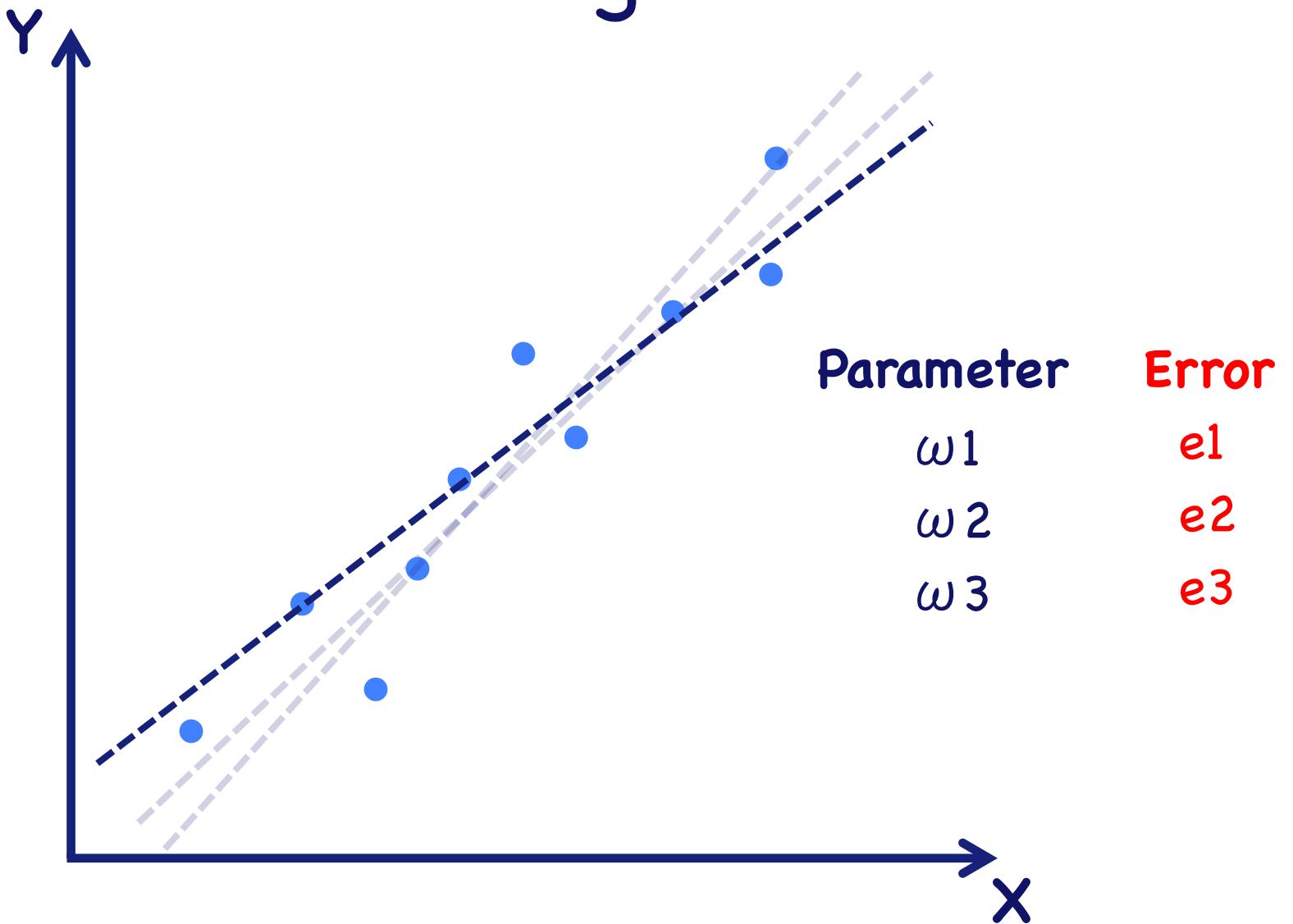
# Measuring error



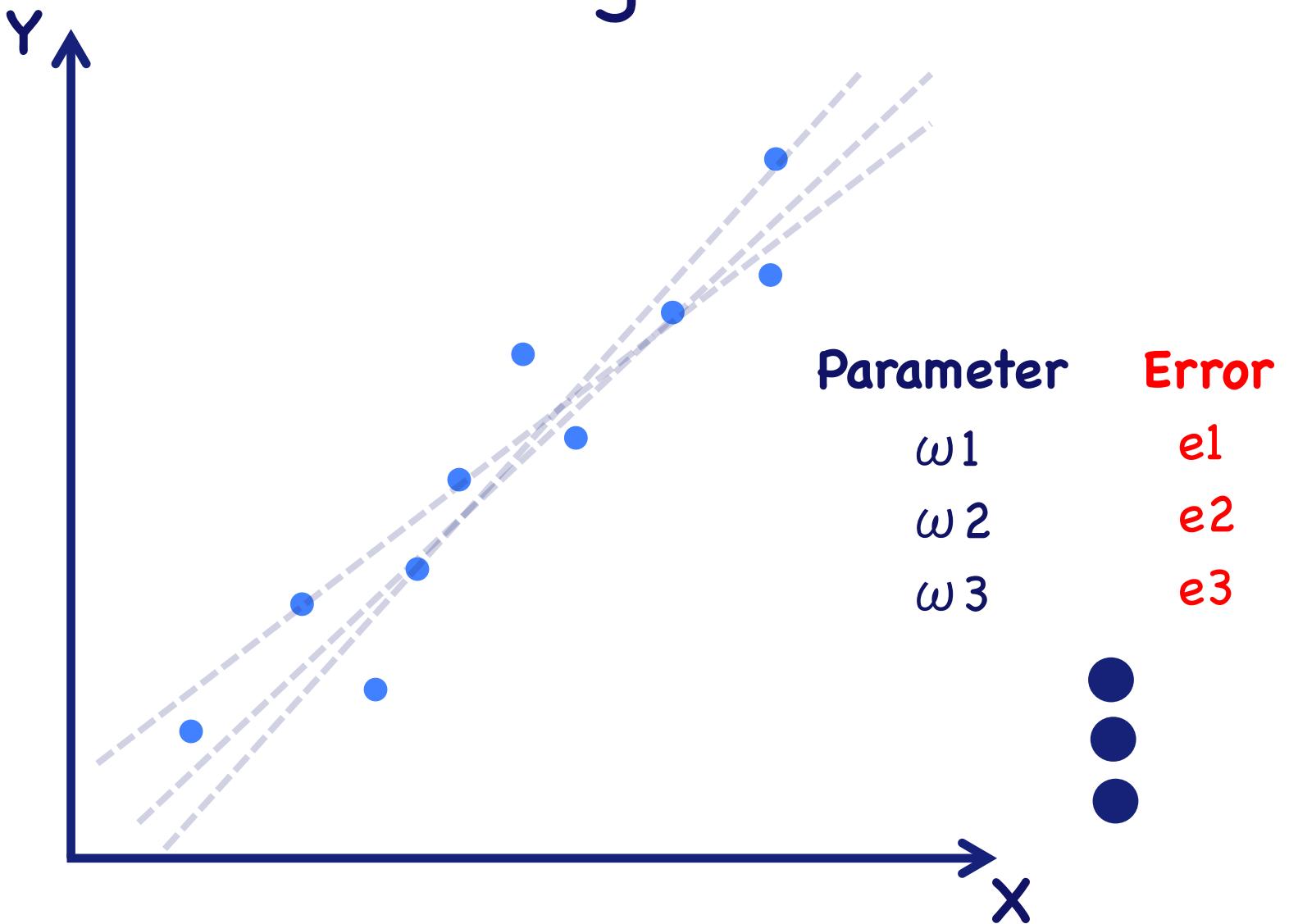
# Measuring error



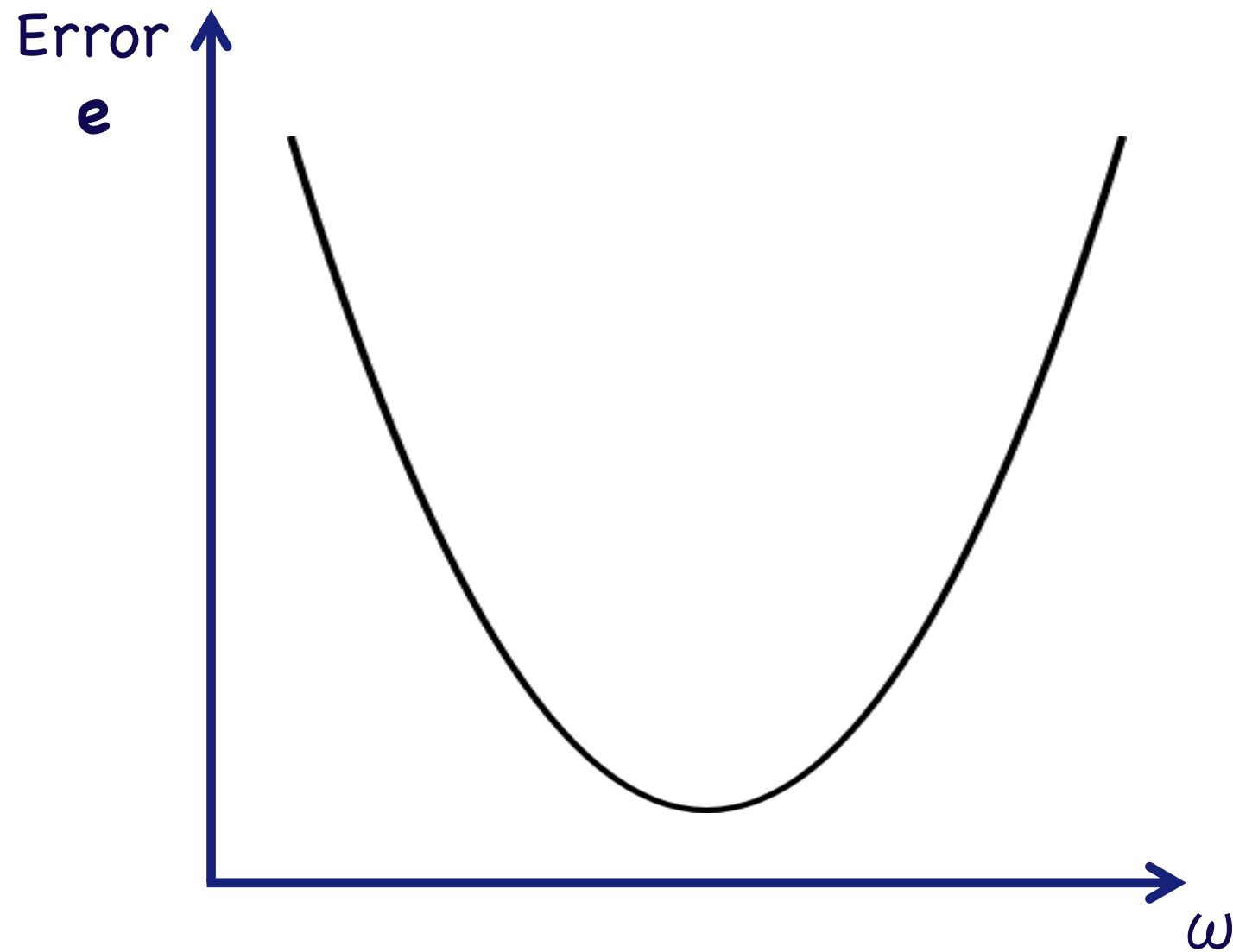
# Measuring error



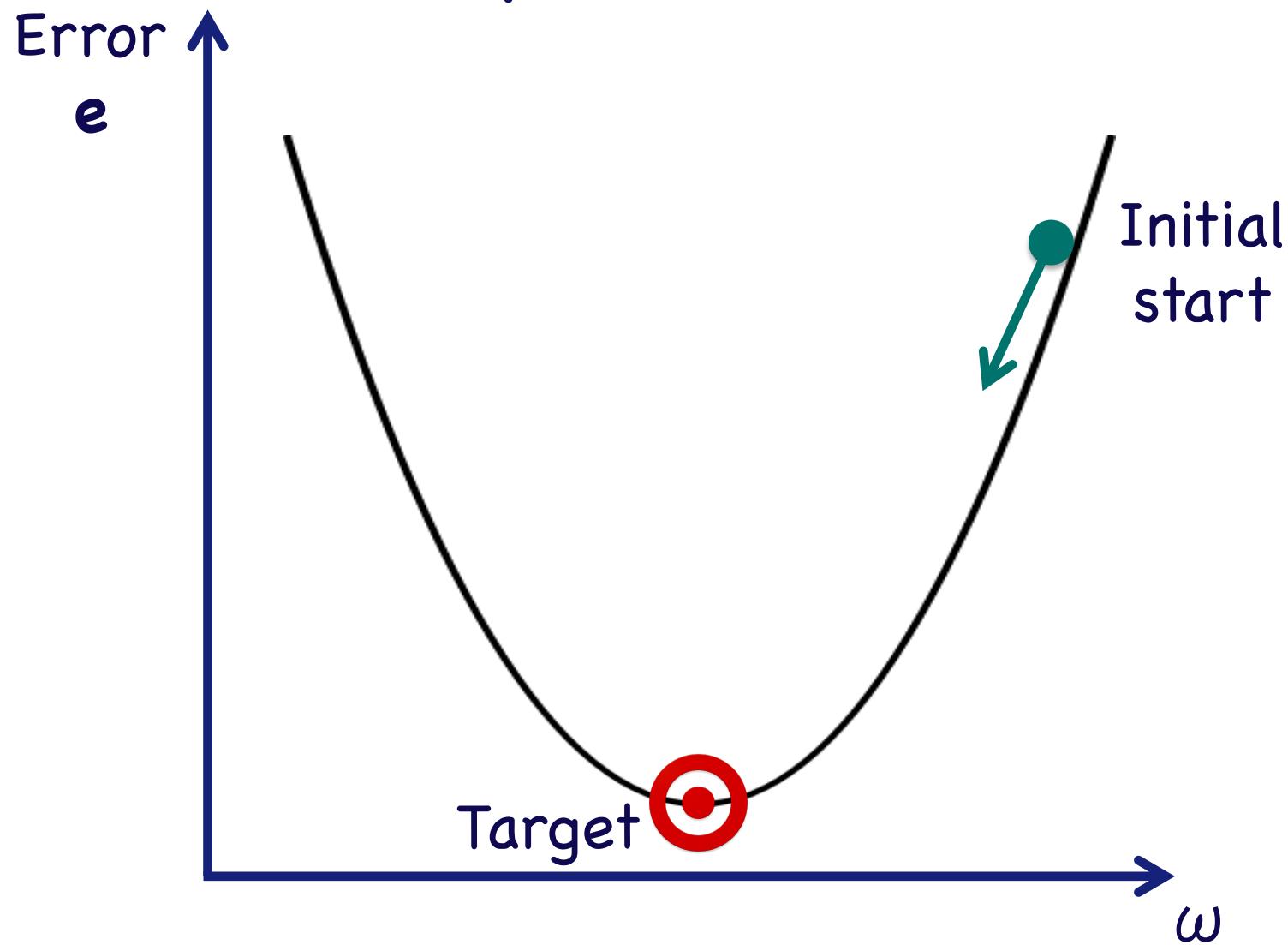
# Measuring error



# Cost function

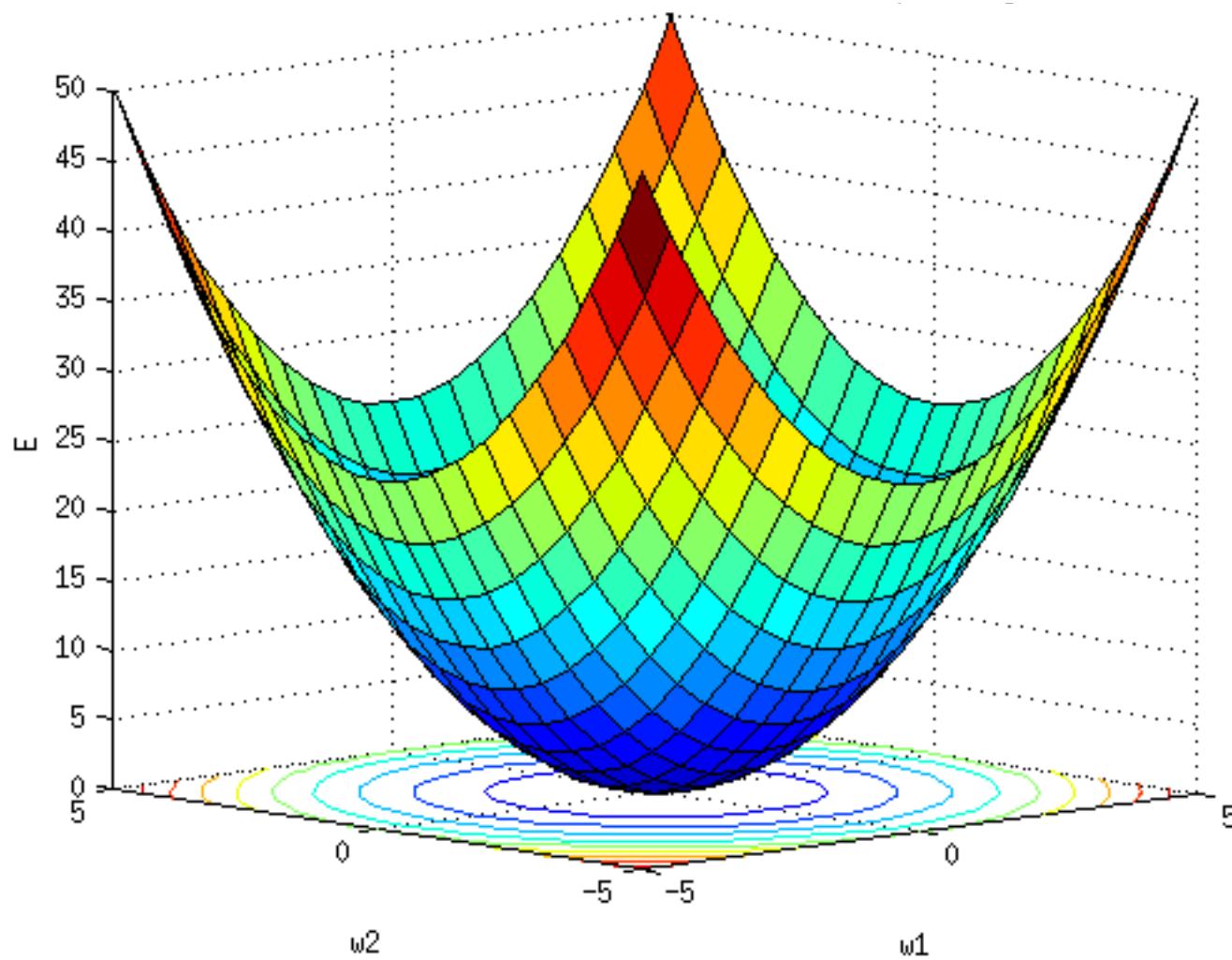


# Optimization



Of course, there are many different types of cost functions and optimization algorithms, but the general idea is very similar.

# Ideal Cost Function



# Real-world Cost Function



Convolutional Neural Network Quadratic Discriminant Analysis

Generative Adversarial Networks Hierarchical clustering

Nearest Neighbor

Radial Basis Function Network

Support Vector Machine

Artificial Neural Network

K-means

Linear Regression

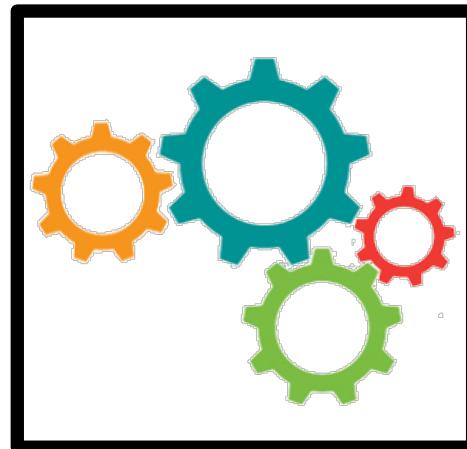
Naïve Bayes

Logistic Regression

AdaBoost Boosting

Self-Organizing Map

Decision Tree



Stepwise Regression

Ridge Regression

Random Forest LASSO

Recurrent Neural Network

Principle Component Analysis

Bayesian Network

Ordinary Least Squares Regression

Linear Discriminant Analysis

Gaussian Mixture Model

Convolutional Neural Network Quadratic Discriminant Analysis

Generative Adversarial Networks Hierarchical clustering

Nearest Neighbor

Radial Basis Function Network

**Support Vector Machine**

**Artificial Neural Network**

K-means

Linear Regression

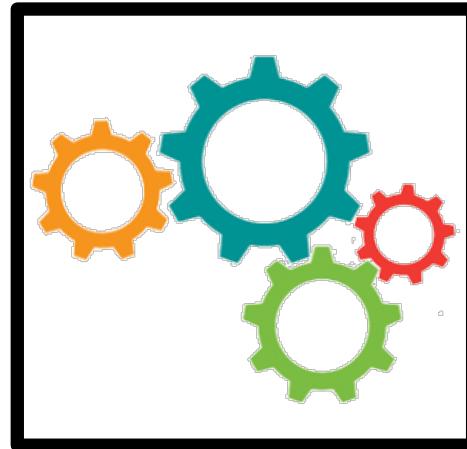
Naïve Bayes

Logistic Regression

AdaBoost Boosting

Self-Organizing Map

Decision Tree



Stepwise Regression

Ridge Regression

**Random Forest**

LASSO

Recurrent Neural Network

**Principle Component Analysis**

Bayesian Network

Ordinary Least Squares Regression

Linear Discriminant Analysis

Gaussian Mixture Model



<http://scikit-learn.org/stable/>

# Machine Learning



what society thinks I do

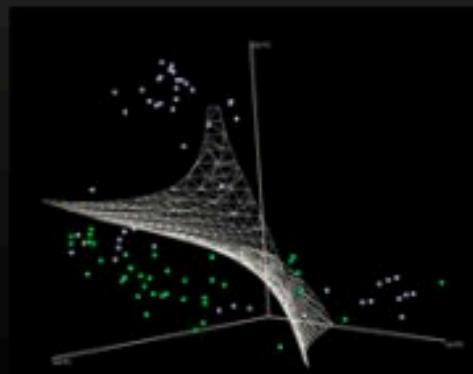


what my friends think I do



what my parents think I do

$$\begin{aligned}L_p &= \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^n \alpha_i y_i (\mathbf{x}_i \cdot \mathbf{w} + b) + \sum_{i=1}^n \alpha_i \\ \alpha_i &\geq 0, \forall i \\ \mathbf{w} &= \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i, \quad \sum_{i=1}^n \alpha_i = 0 \\ \nabla g(\theta_t) &= \frac{1}{n} \sum_{i=1}^n \nabla \ell(x_i, y_i; \theta_t) + \nabla r(\theta_t). \\ \theta_{t+1} &= \theta_t - \eta_t \nabla \ell(x_{i(t)}, y_{i(t)}; \theta_t) - \eta_t \cdot \nabla r(\theta_t) \\ E_{i(t)}[\ell(x_{i(t)}, y_{i(t)}; \theta_t)] &= \frac{1}{n} \sum_{i=1}^n \ell(x_i, y_i; \theta_t).\end{aligned}$$



```
>>> from sklearn import svm
```

what other programmers think I do

what I think I do

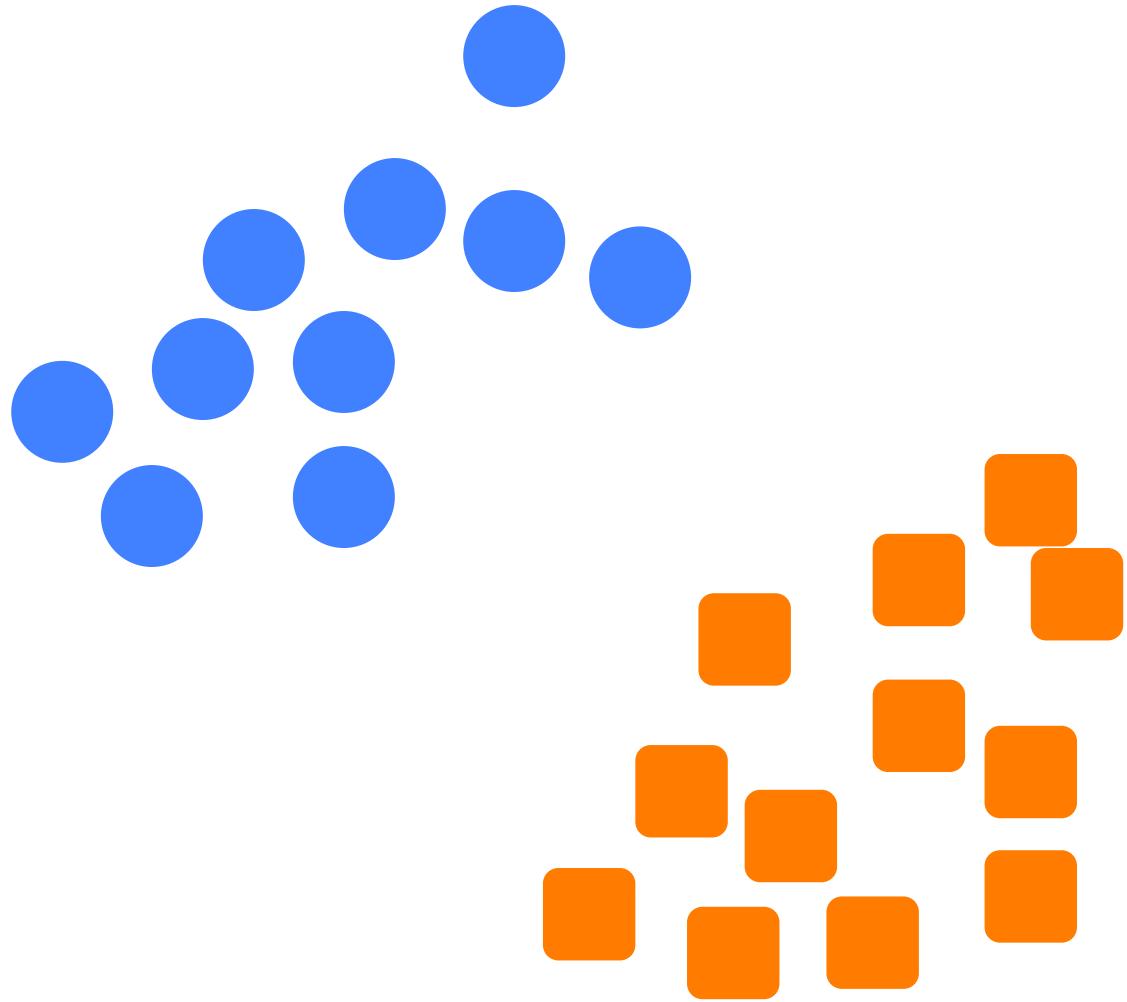
what I really do

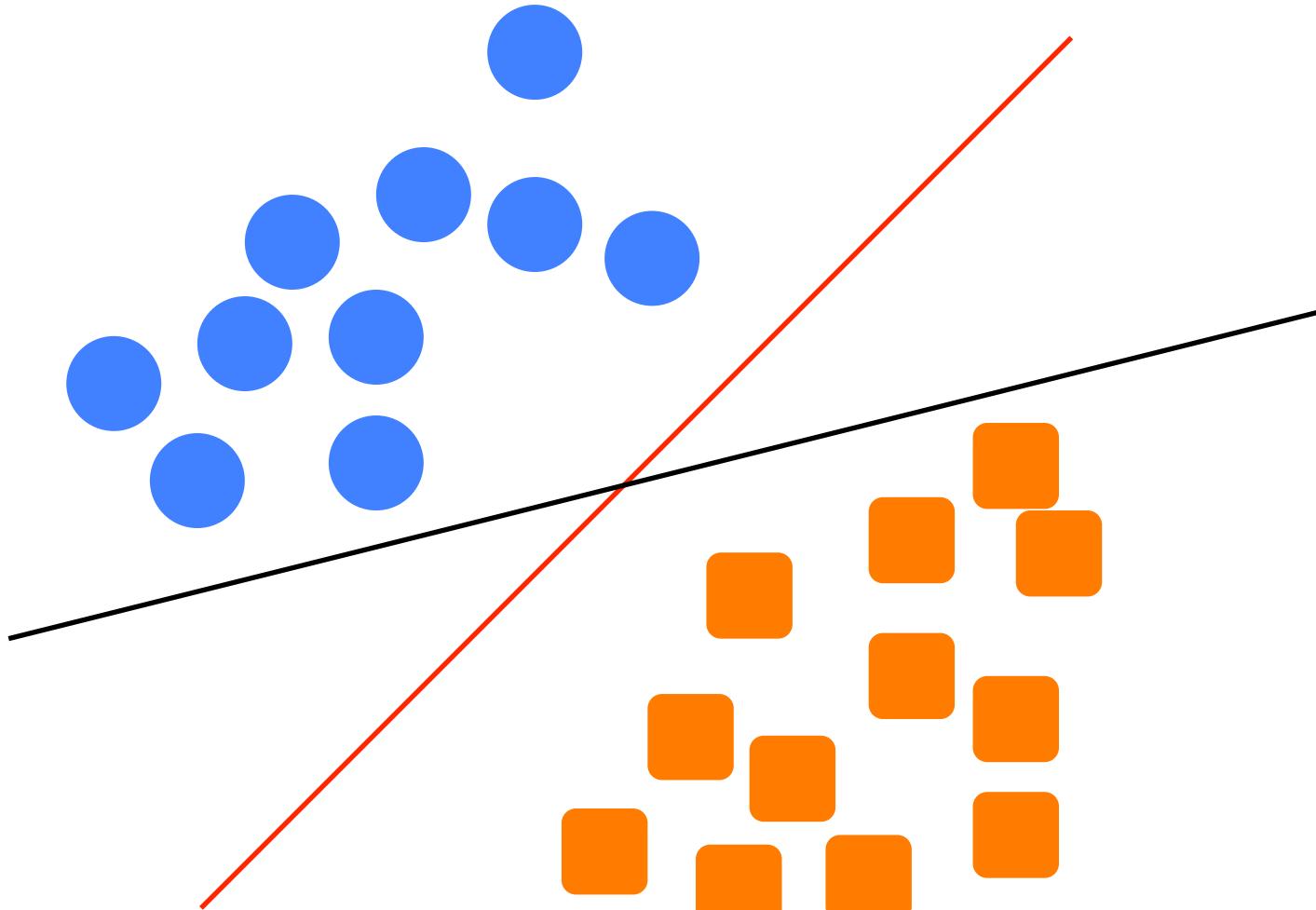
Go to notebook 01

# Classification

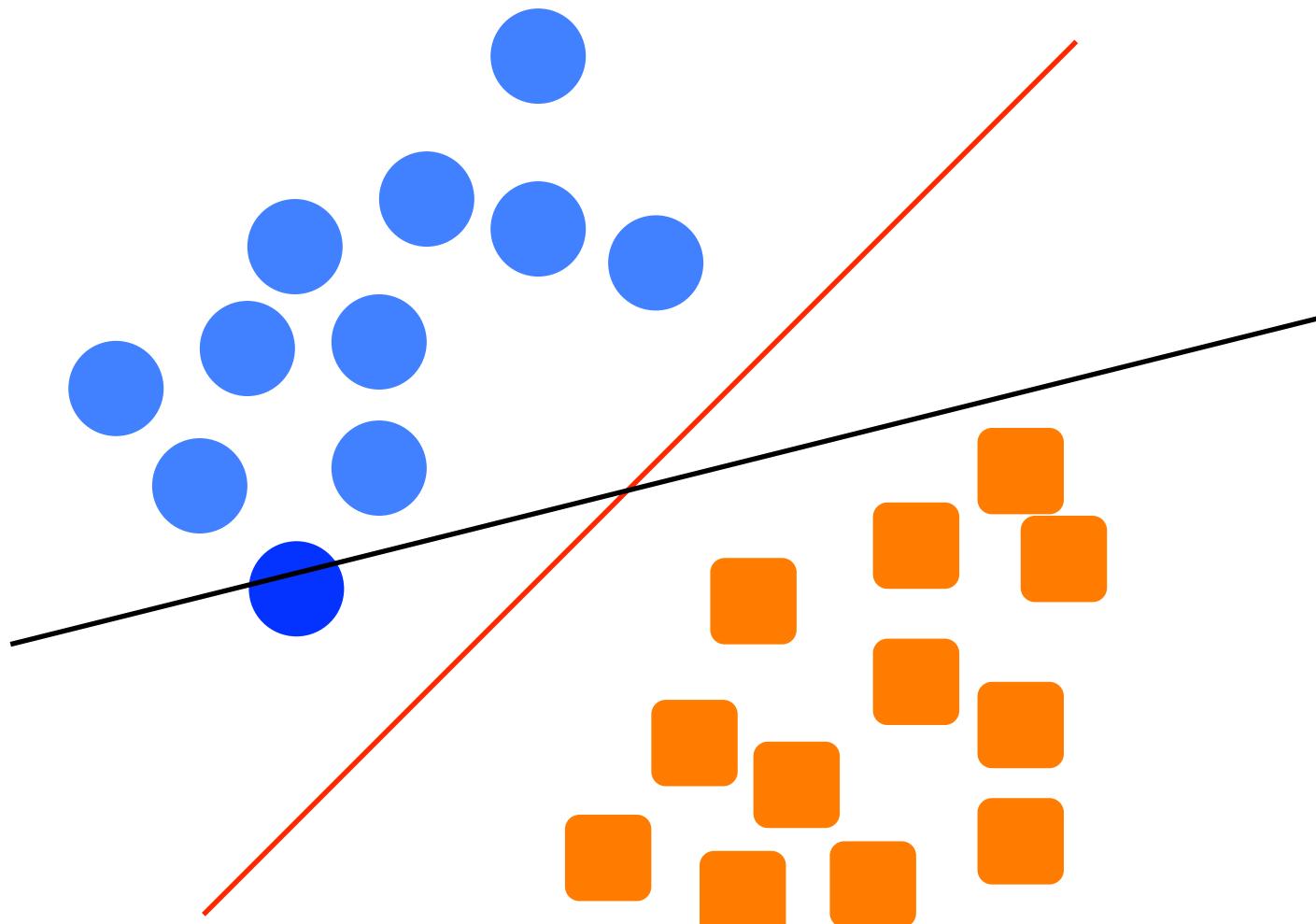
Support Vector Machine  
Artificial Neural Network



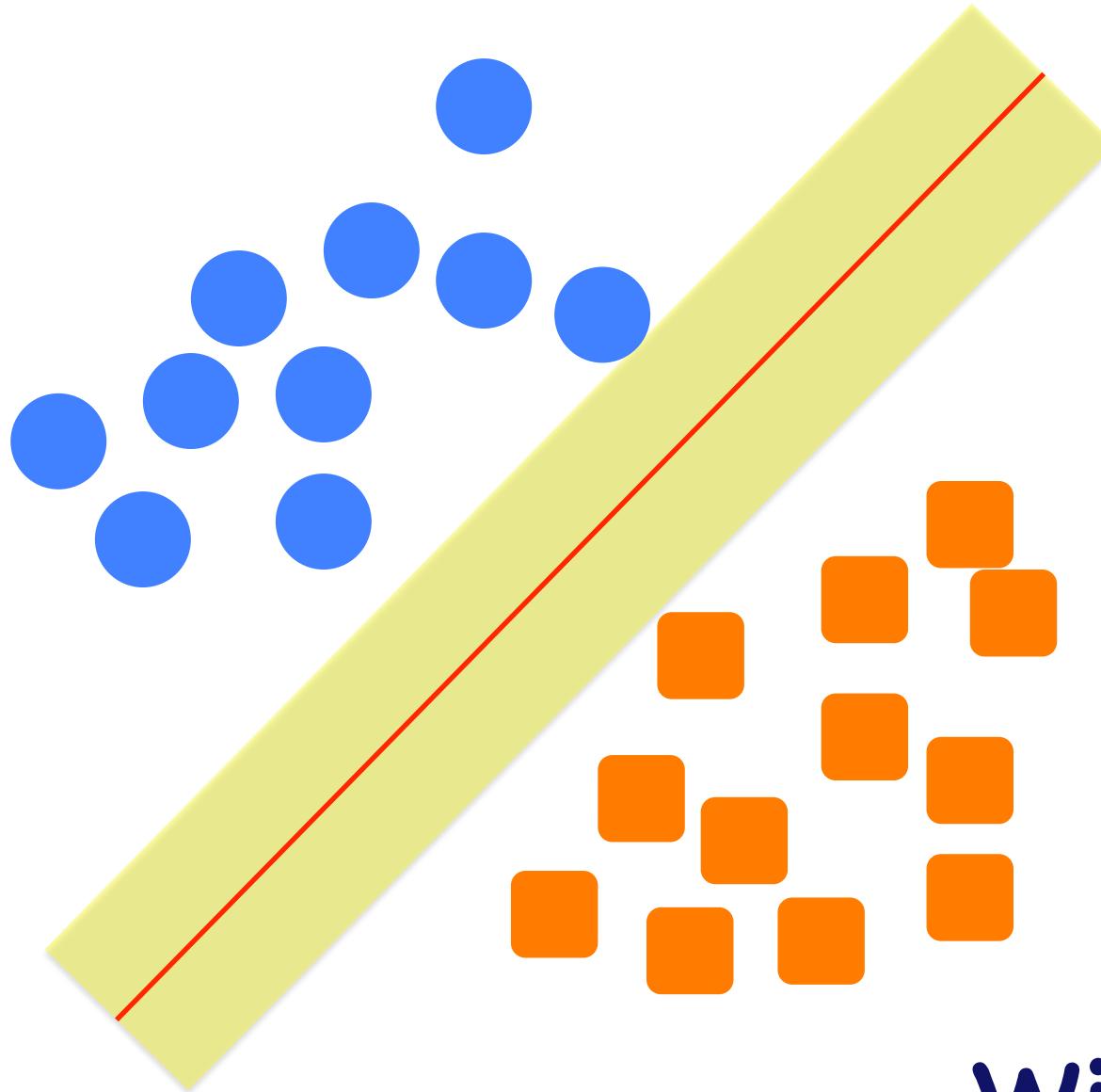




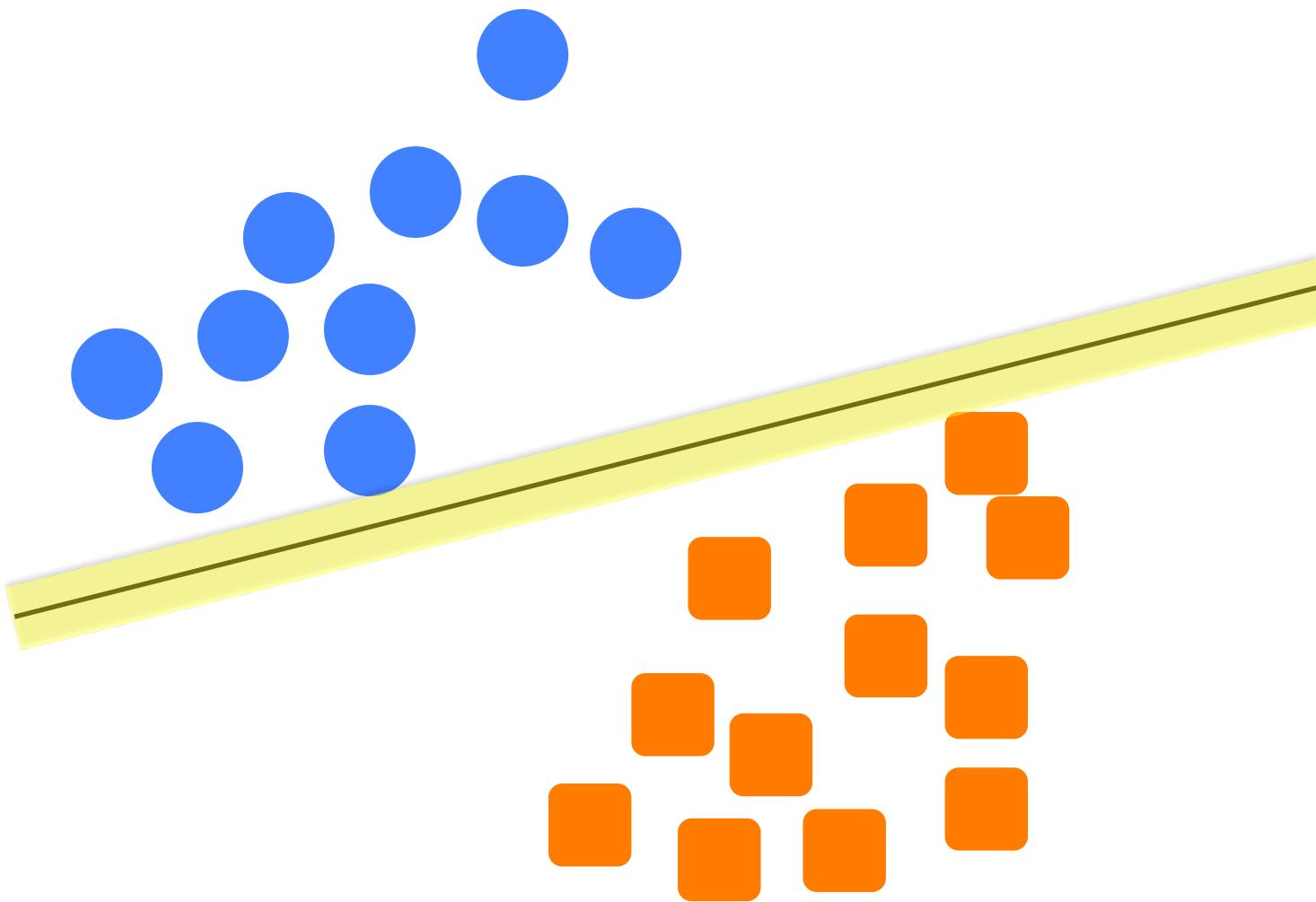
Which one is  
better?



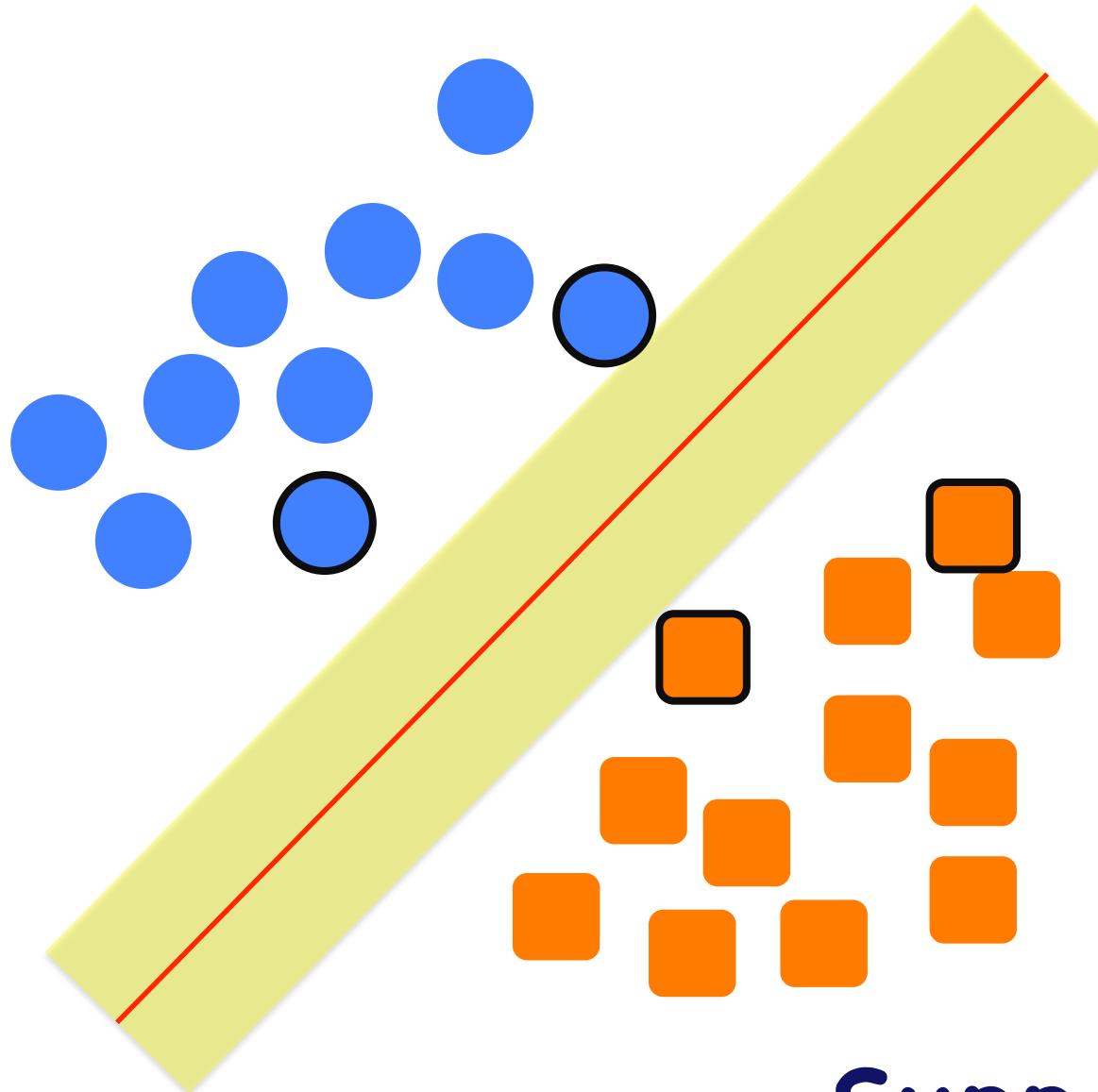
A new data point



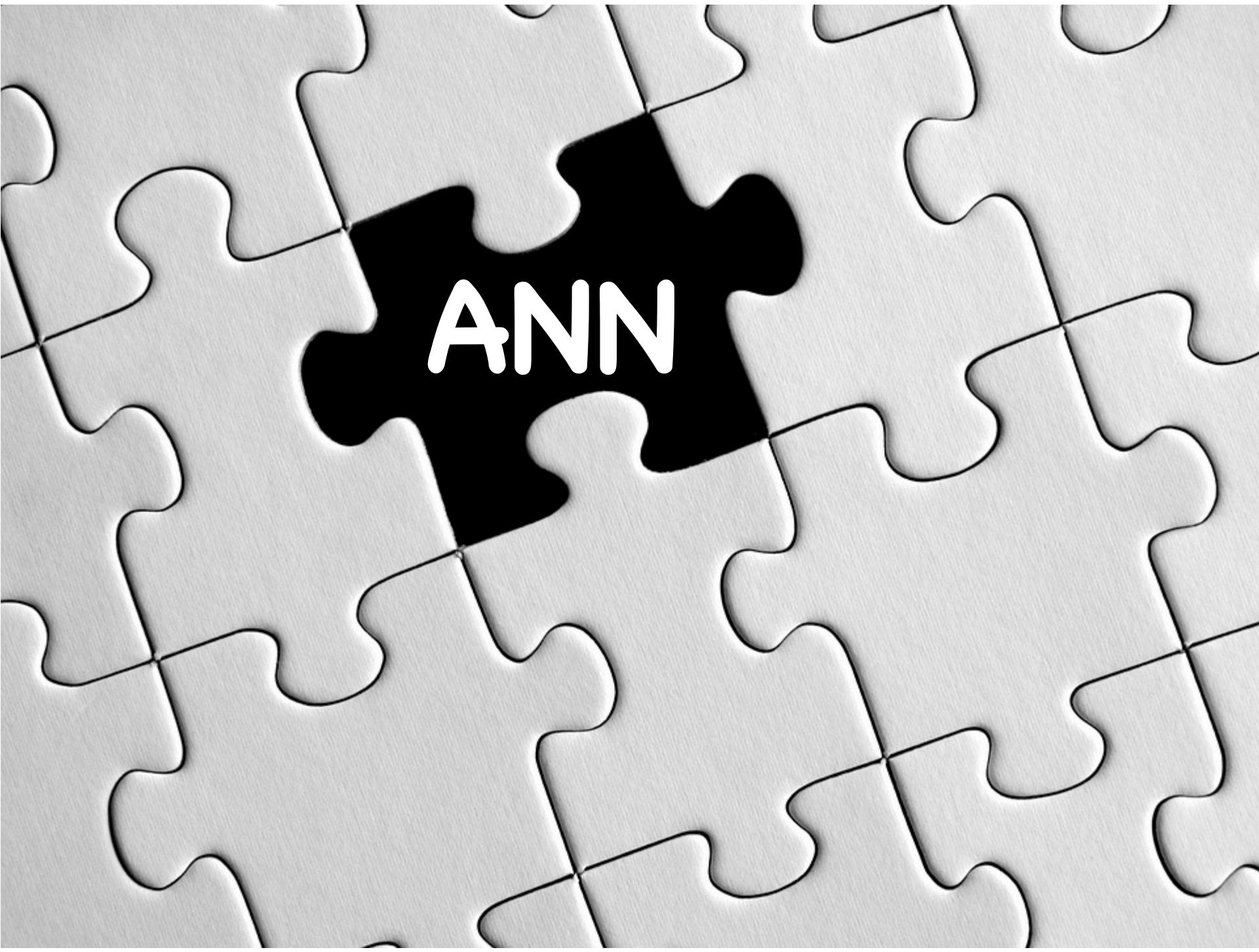
Wide margin



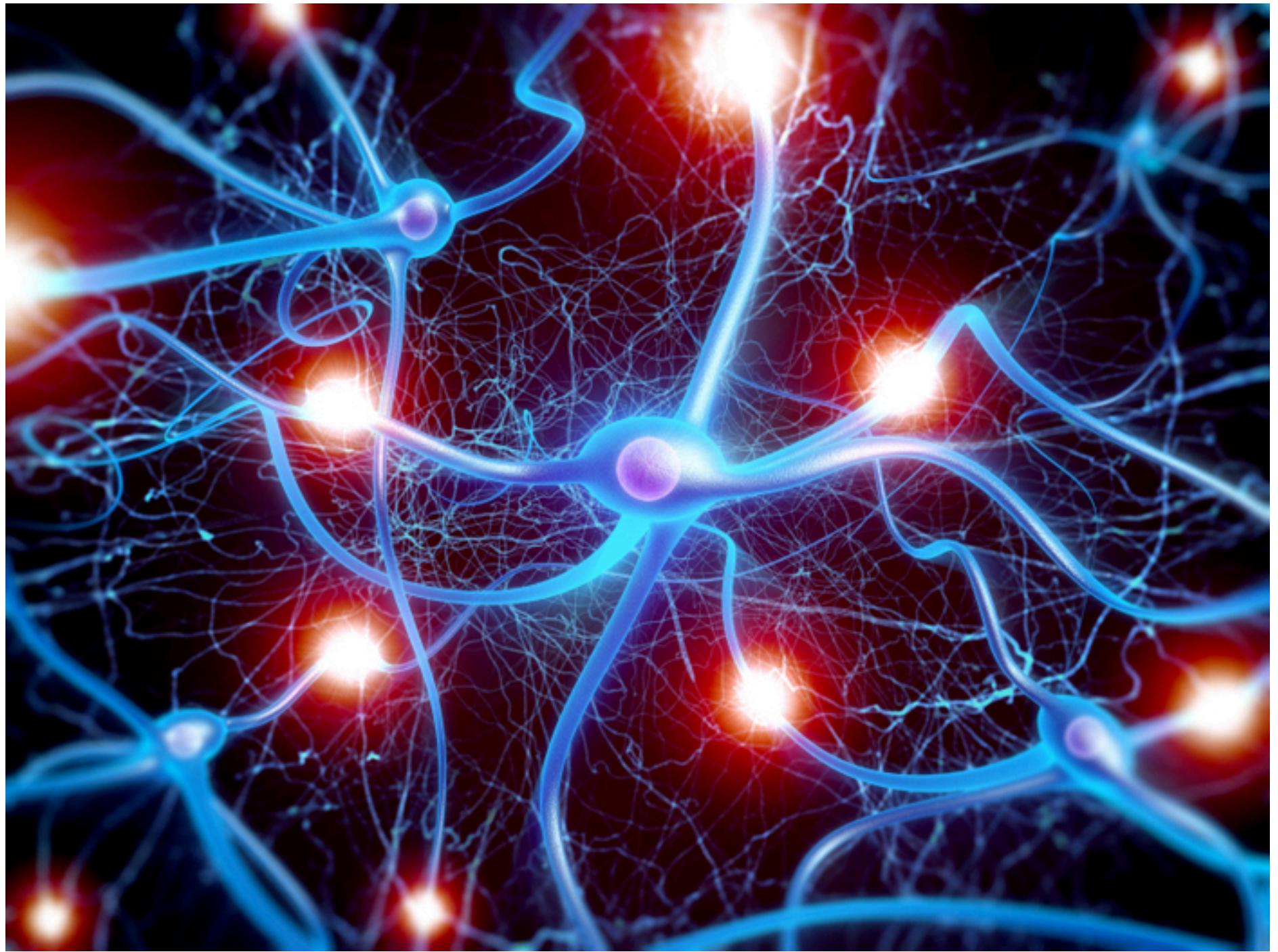
Narrow margin

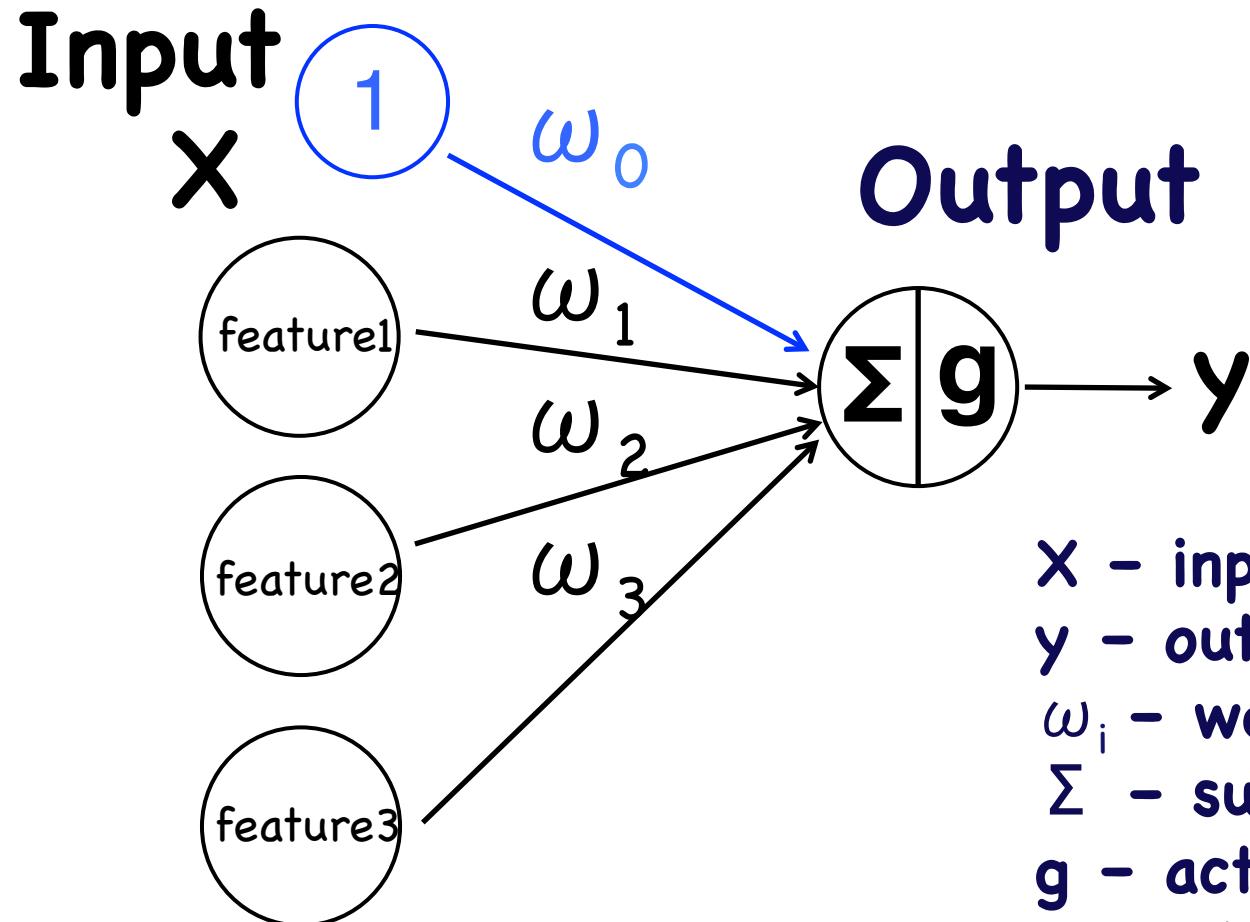


**Support Vectors**



ANN



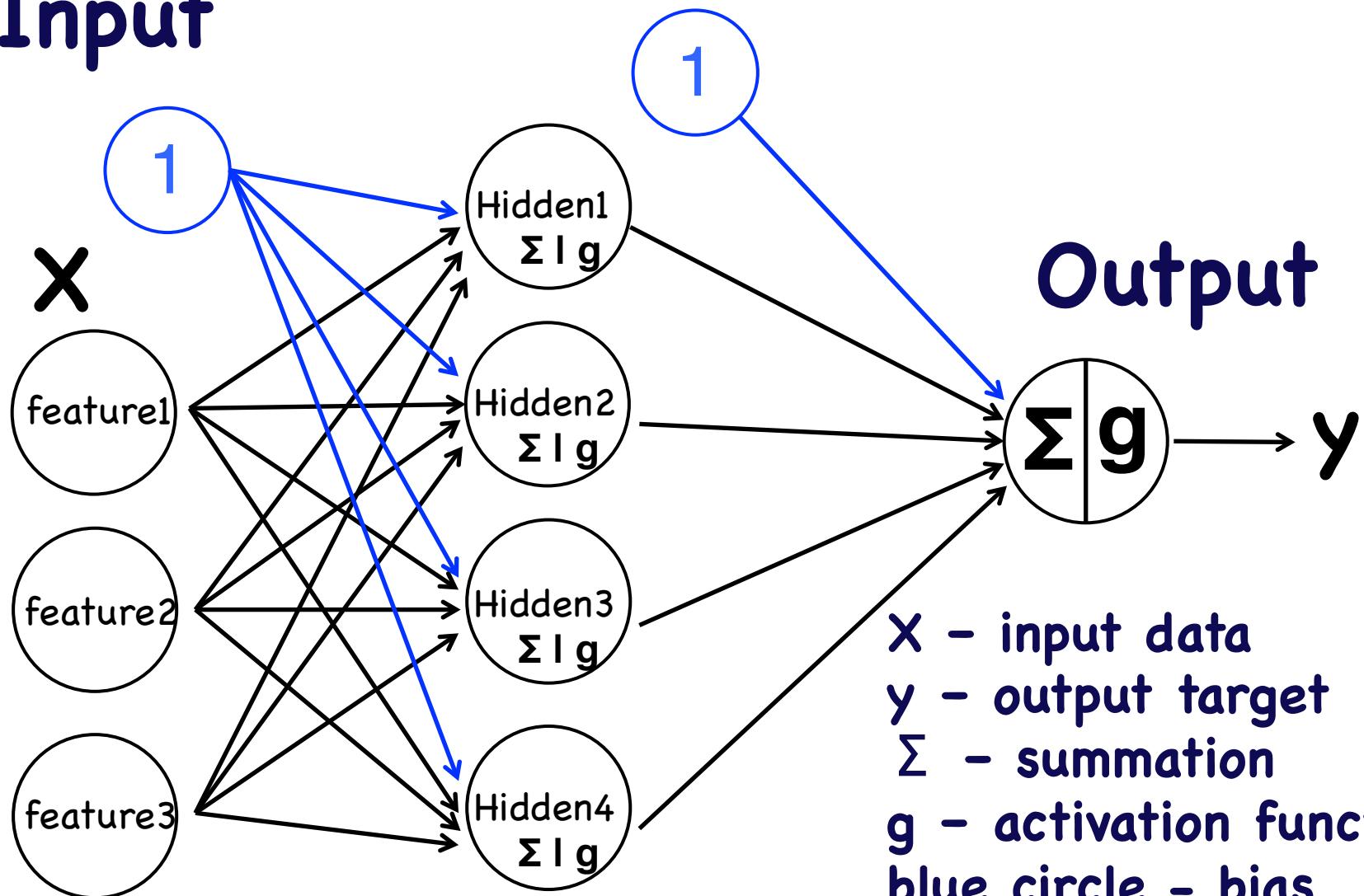


$x$  - input data  
 $y$  - output target  
 $\omega_i$  - weights  
 $\Sigma$  - summation  
 $g$  - activation function  
 Blue circle - bias

$$Z = \Sigma = \omega_0x_0 + \omega_1x_1 + \omega_2x_2 + \omega_3x_3 + \dots + \omega_nx_n$$

$$y = g(\omega_0x_0 + \omega_1x_1 + \omega_2x_2 + \omega_3x_3 + \dots + \omega_nx_n)$$

# Input



$x$  - input data  
 $y$  - output target  
 $\Sigma$  - summation  
 $g$  - activation function  
blue circle - bias



**YOU'RE IN MY SPOT**

GRAPHICS GARAGE

# Input



•  
•  
•



# Intuitive Artificial Neural Network

$$w_1 \quad \downarrow \\ w_2 \quad \downarrow \\ \vdots \\ w_n \quad \uparrow$$

$$F(\text{eye} \times w_1 + \text{nose} \times w_2 + \dots + \text{mouth} \times w_n)$$



# Output



# Input



•  
•  
•



# Intuitive Artificial Neural Network

# Output

$$F(\text{eye} \times w_1 + \text{nose} \times w_2 + \dots + \text{mouth} \times w_n)$$



error  
feedback



# Input



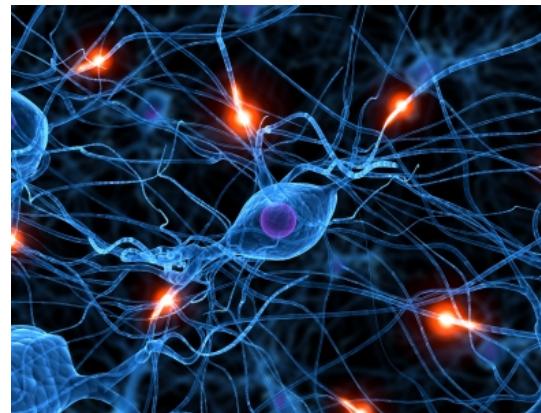
•  
•  
•



# Intuitive Artificial Neural Network

# Output

$$F(\text{eye} \times w_1 + \text{nose} \times w_2 + \dots + \text{mouth} \times w_n)$$



error  
feedback

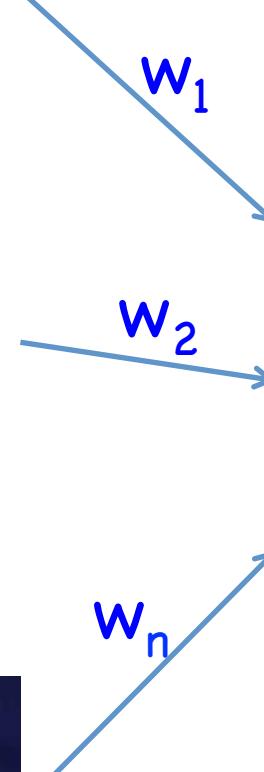
# Input



•  
•  
•



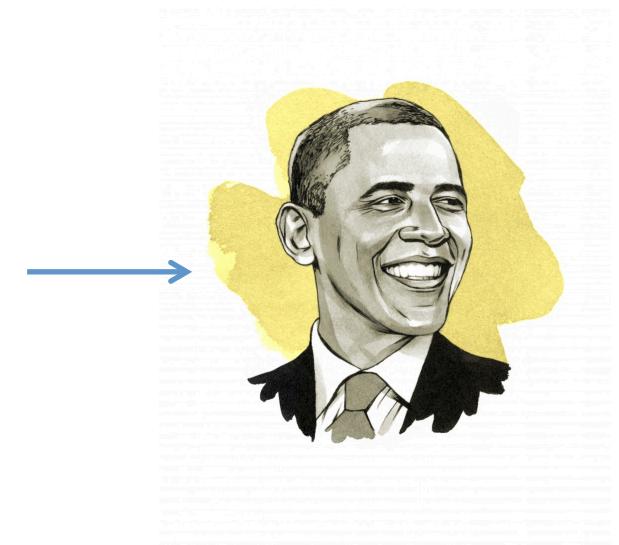
# Intuitive Artificial Neural Network



$$F(\text{eye} \times w_1 + \text{nose} \times w_2 + \dots + \text{mouth} \times w_n)$$



# Output

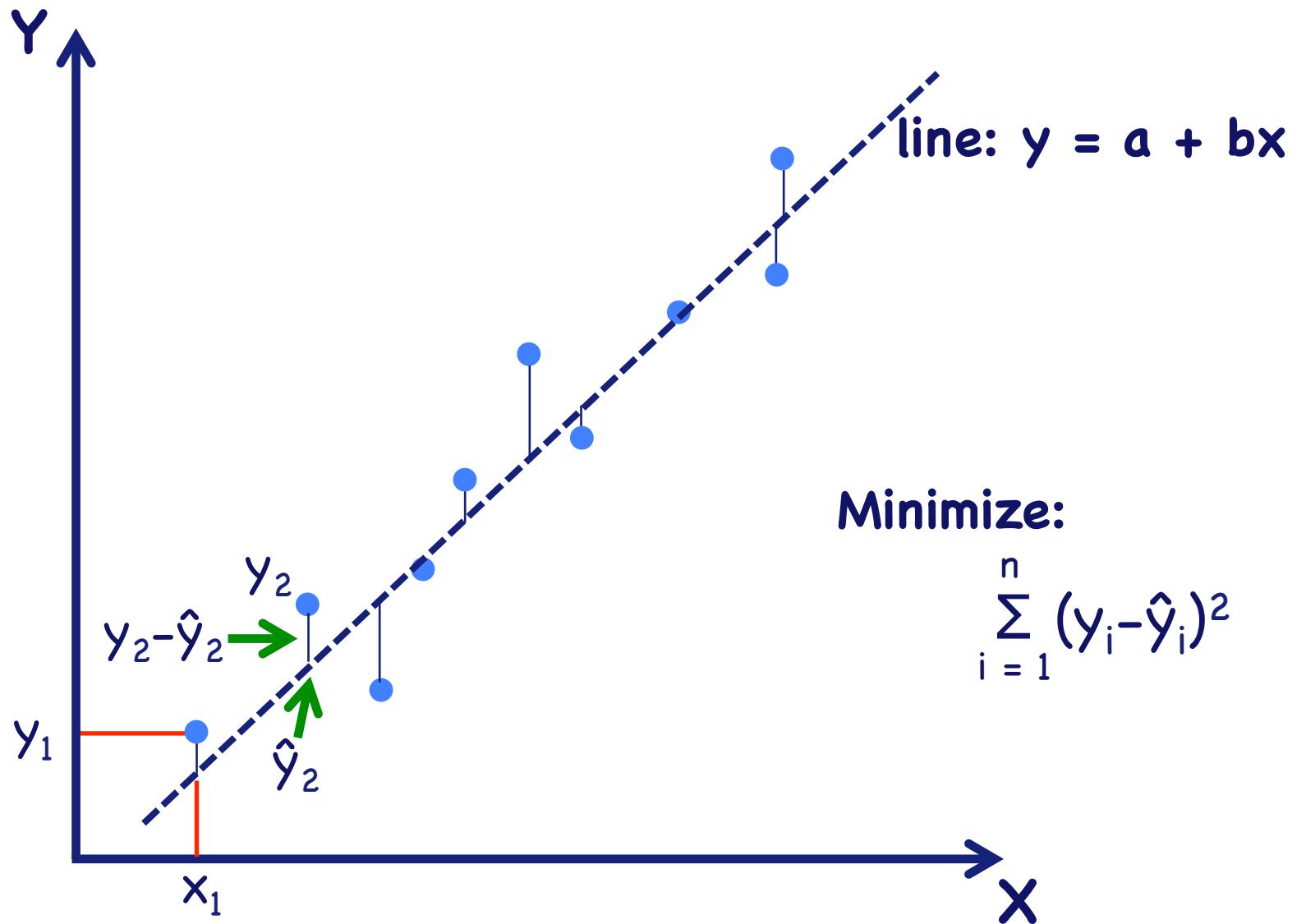


Go to notebook 02

# Regression

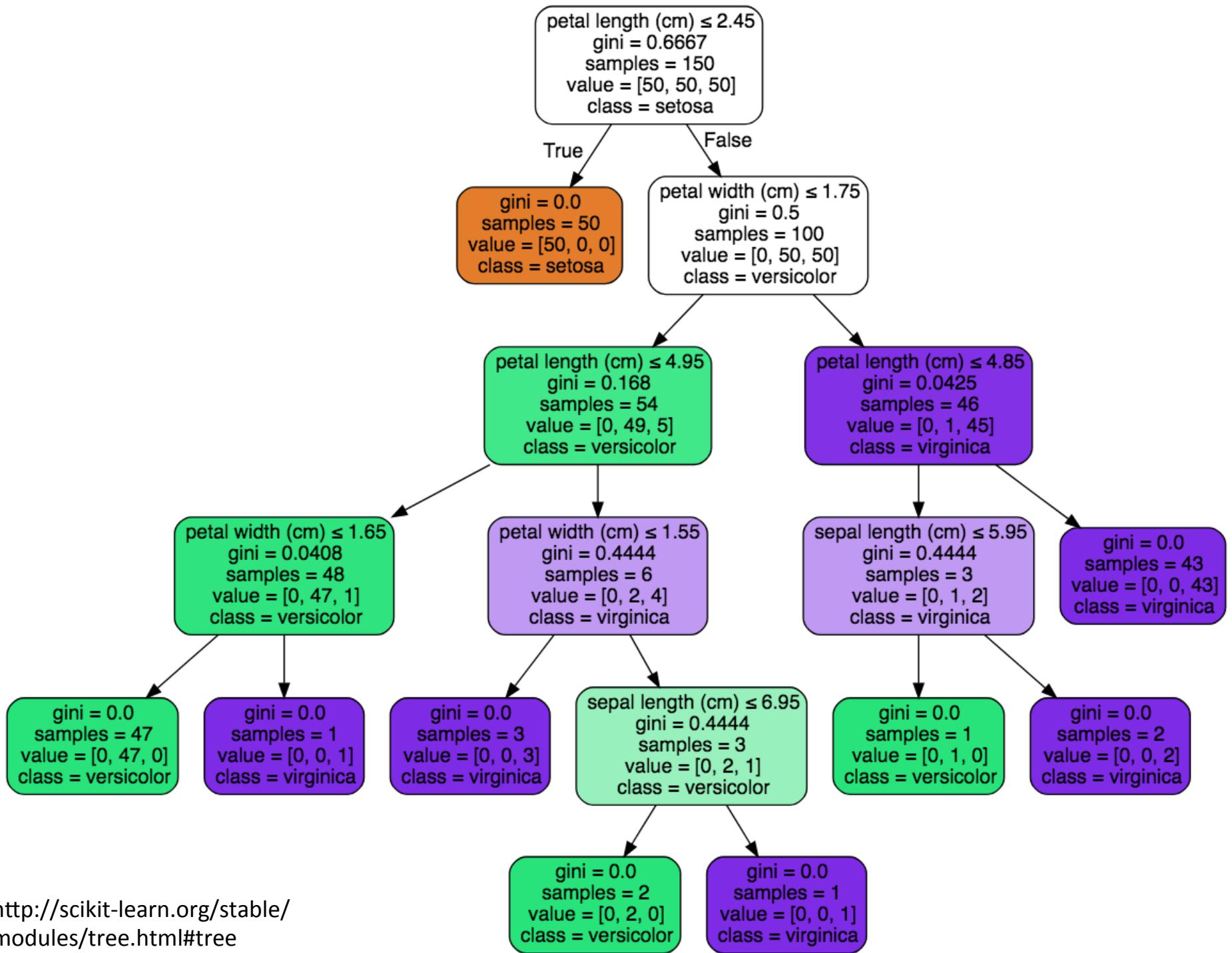
Supervised  
learning

# Simple linear regression

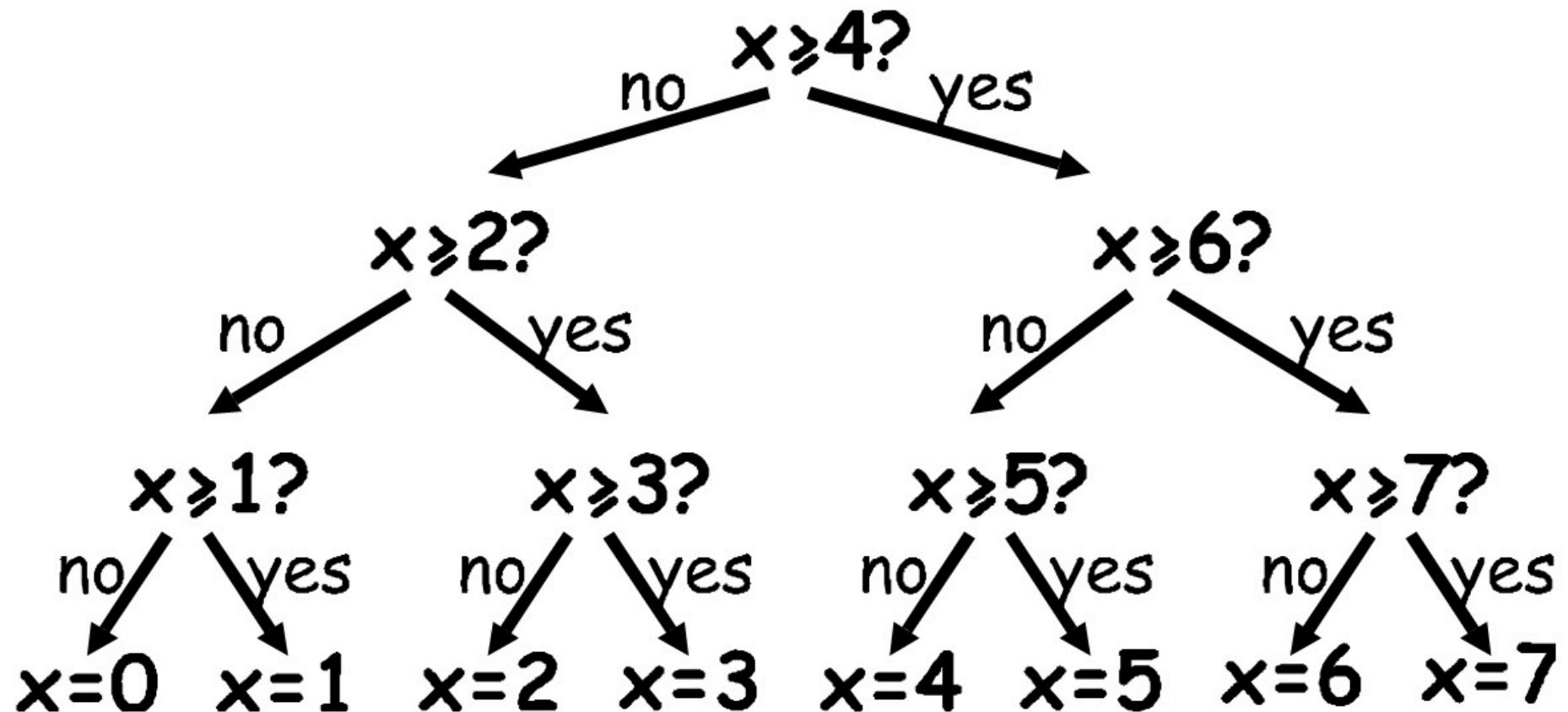


# Random Forest

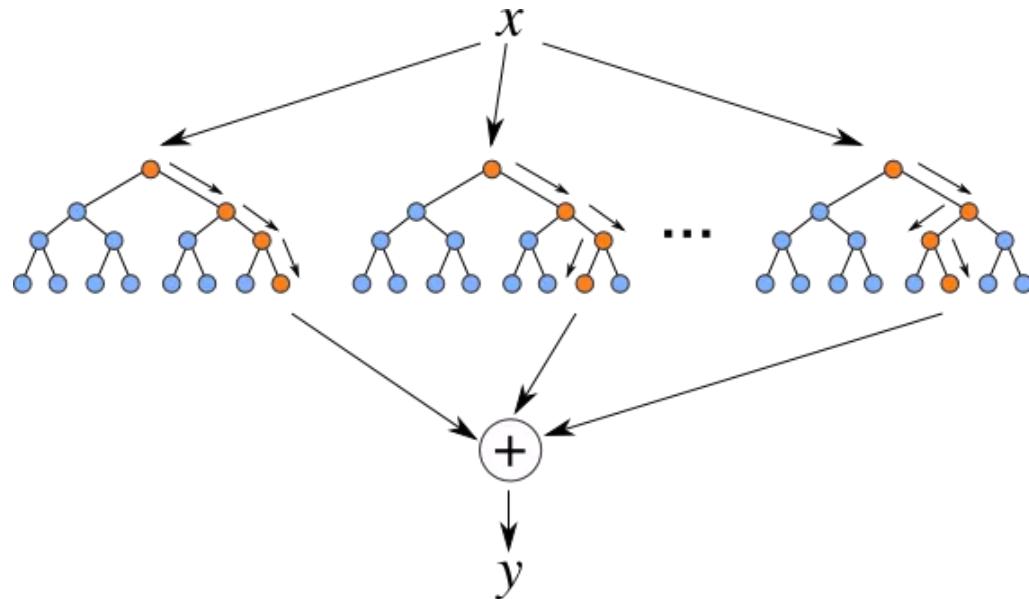




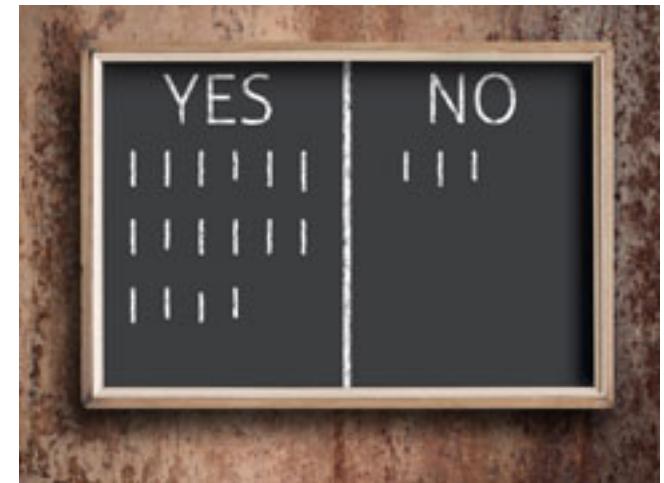
# Decision Trees



# Random Forest



Majority vote



Go to notebook 03

# Unsupervised

Principle component analysis  
K-means

# PCA

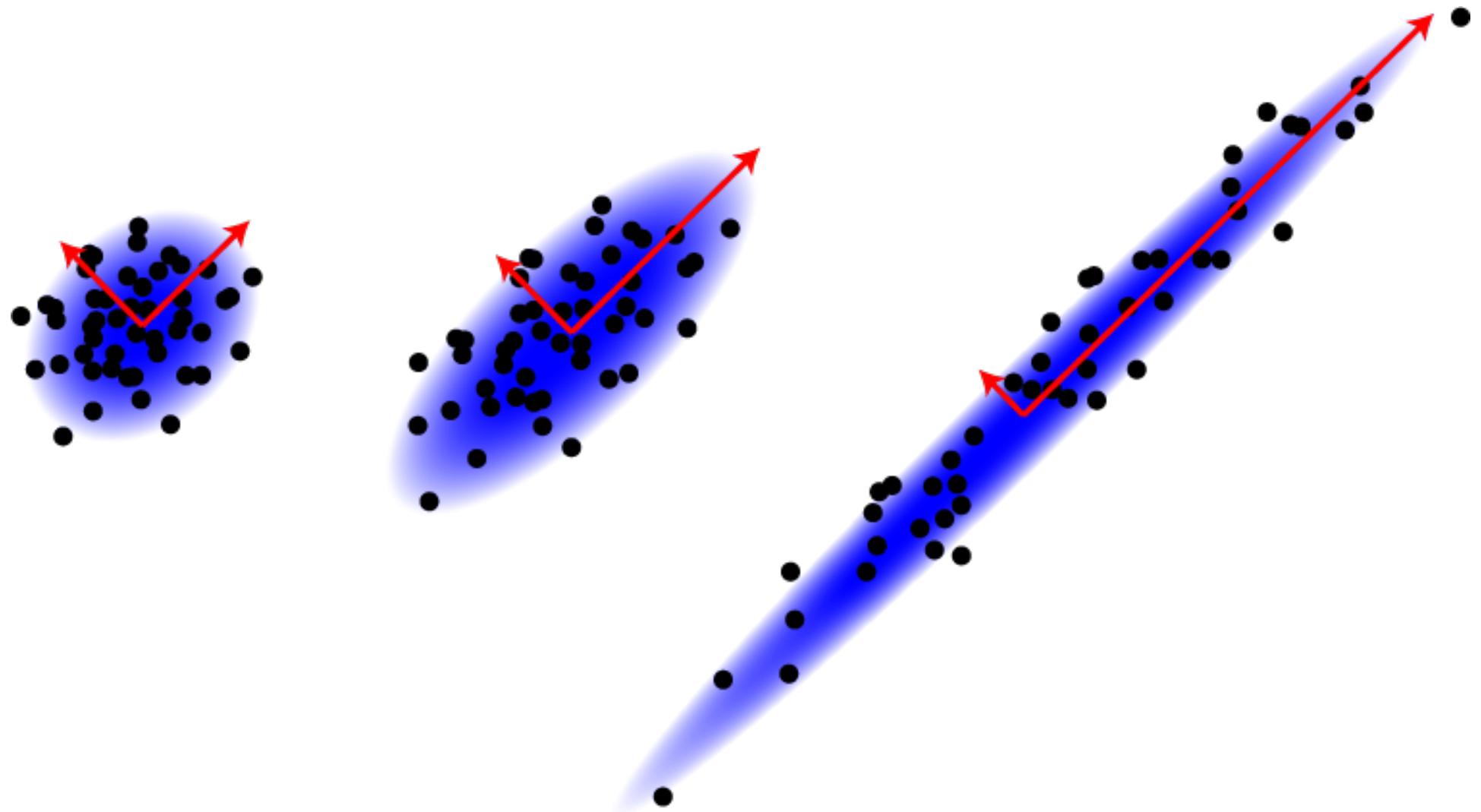
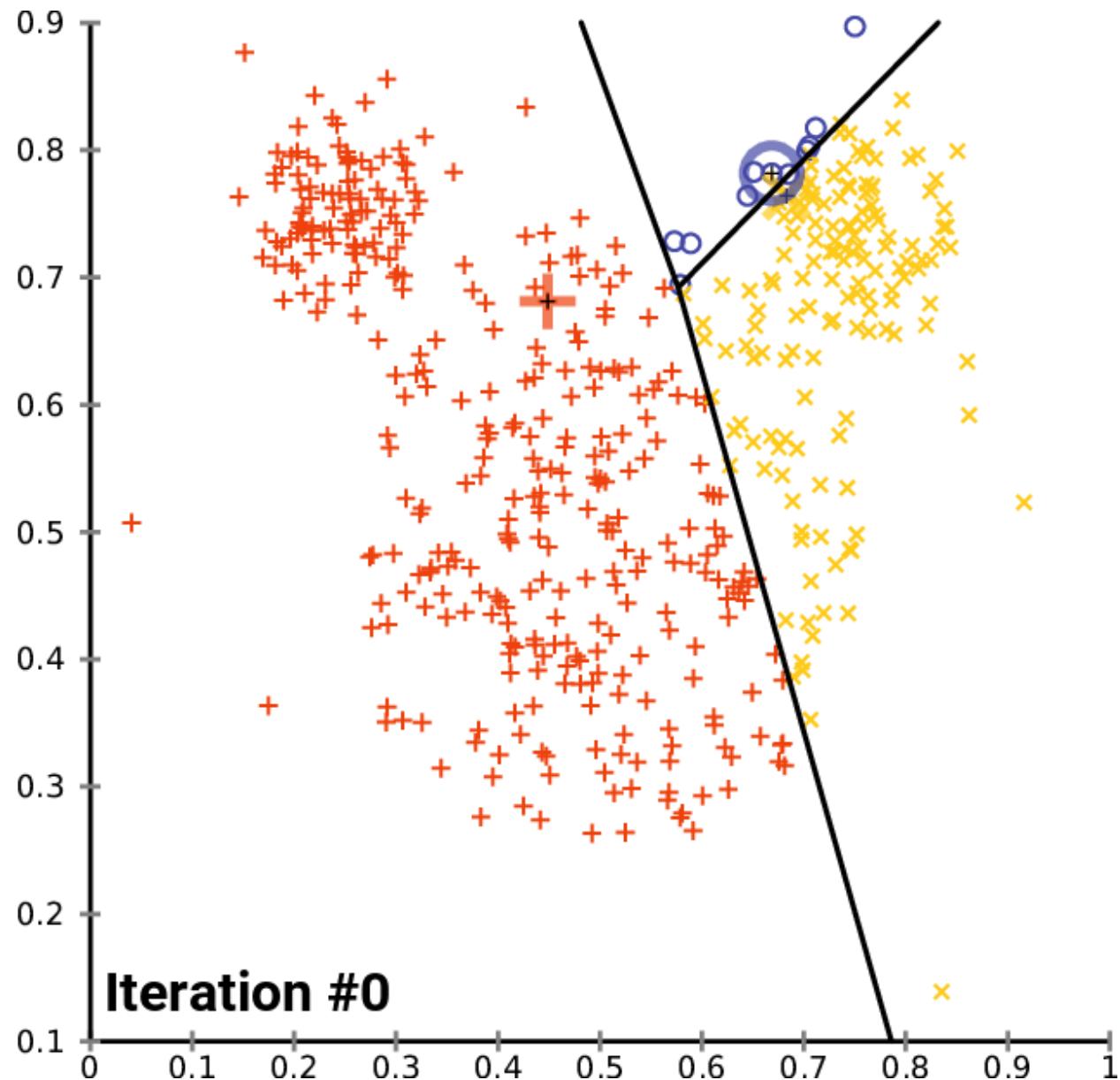


Figure from: <https://shapeofdata.wordpress.com/2013/04/09/principle-component-analysis/>

# Kmeans



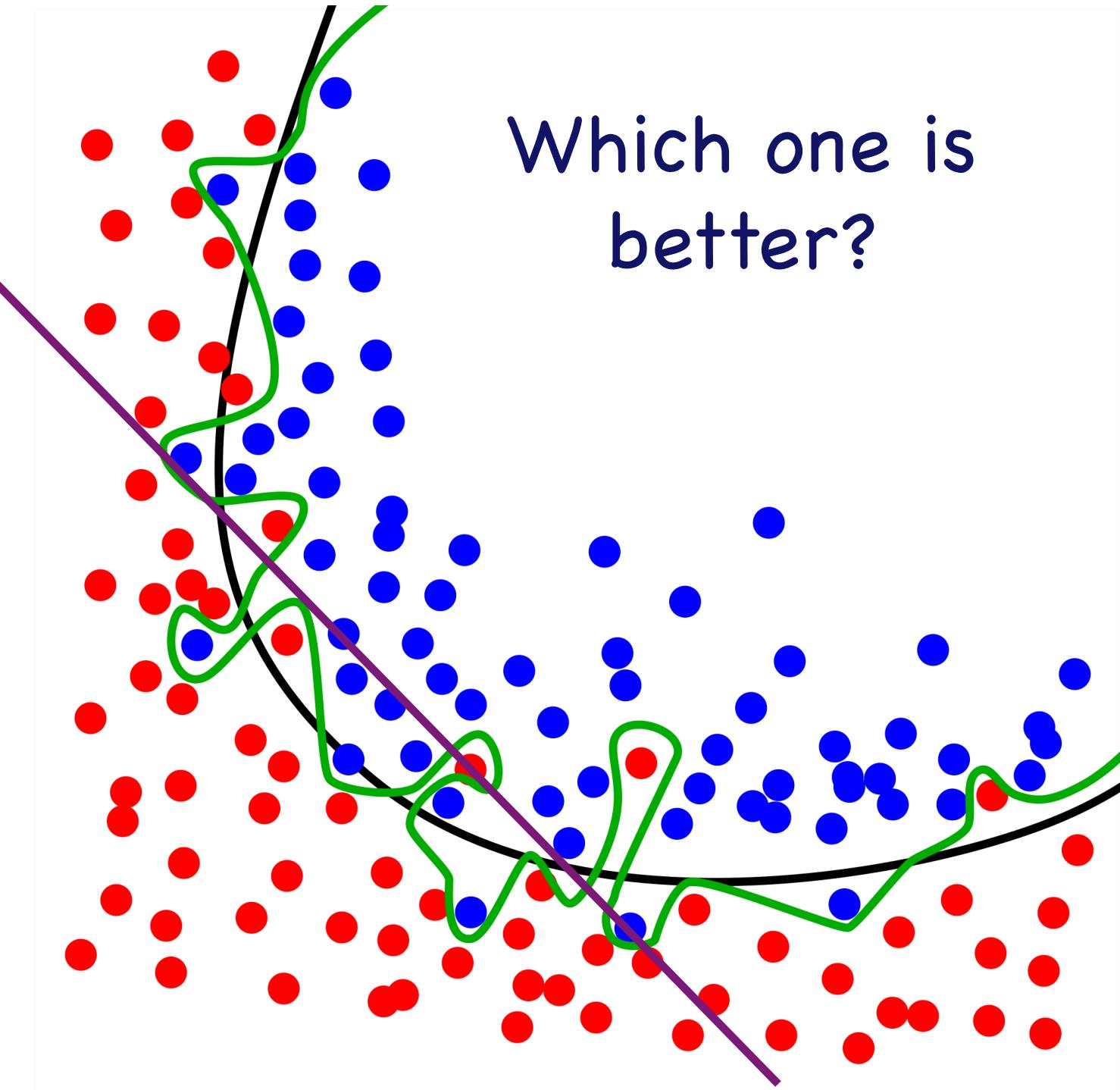
Go to notebook 04

# More on common practices

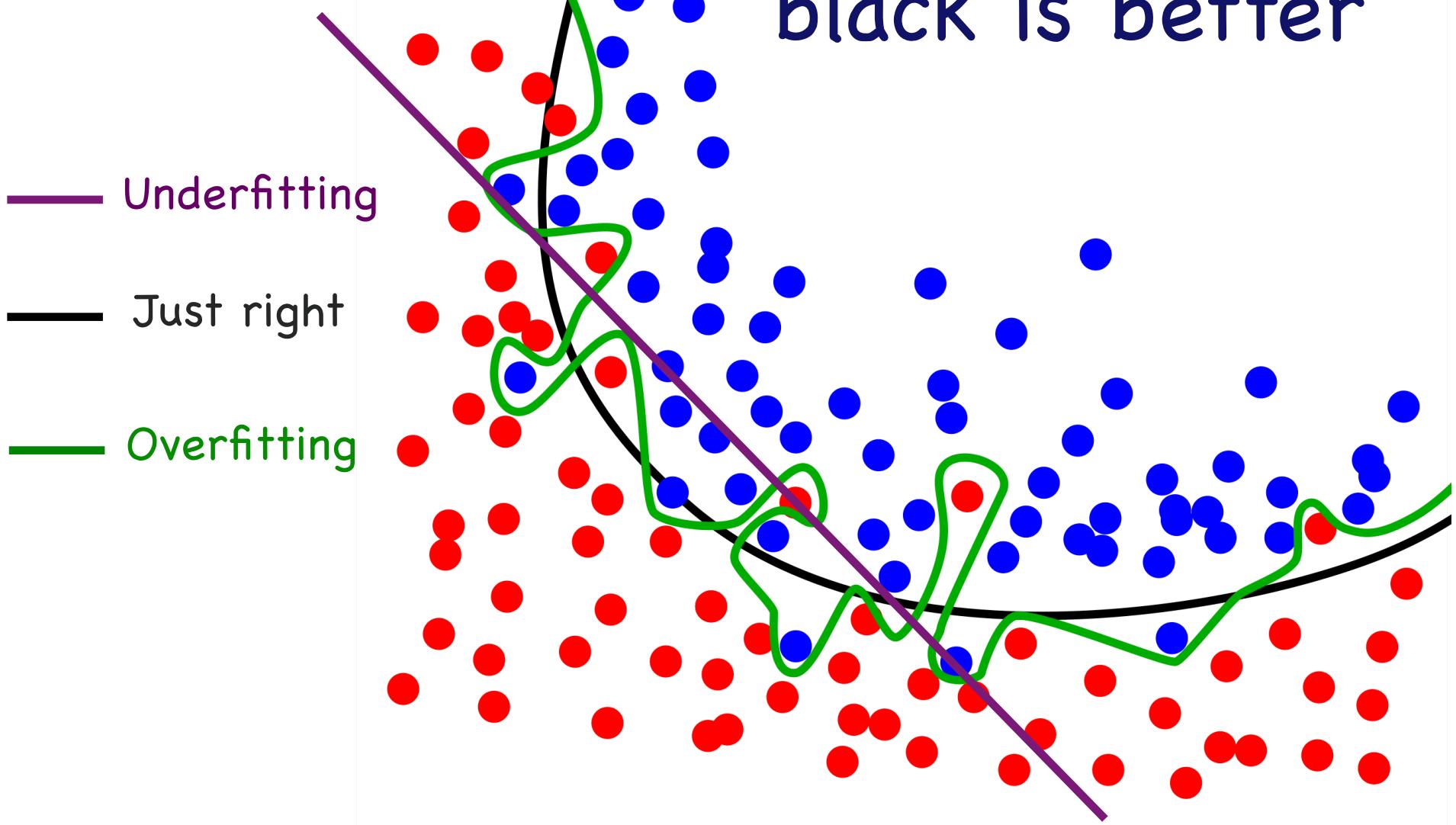
Machine learning is all about  
**Generalization**

Which one is  
better?

— ?  
— ?  
— ?



Clearly,  
black is better

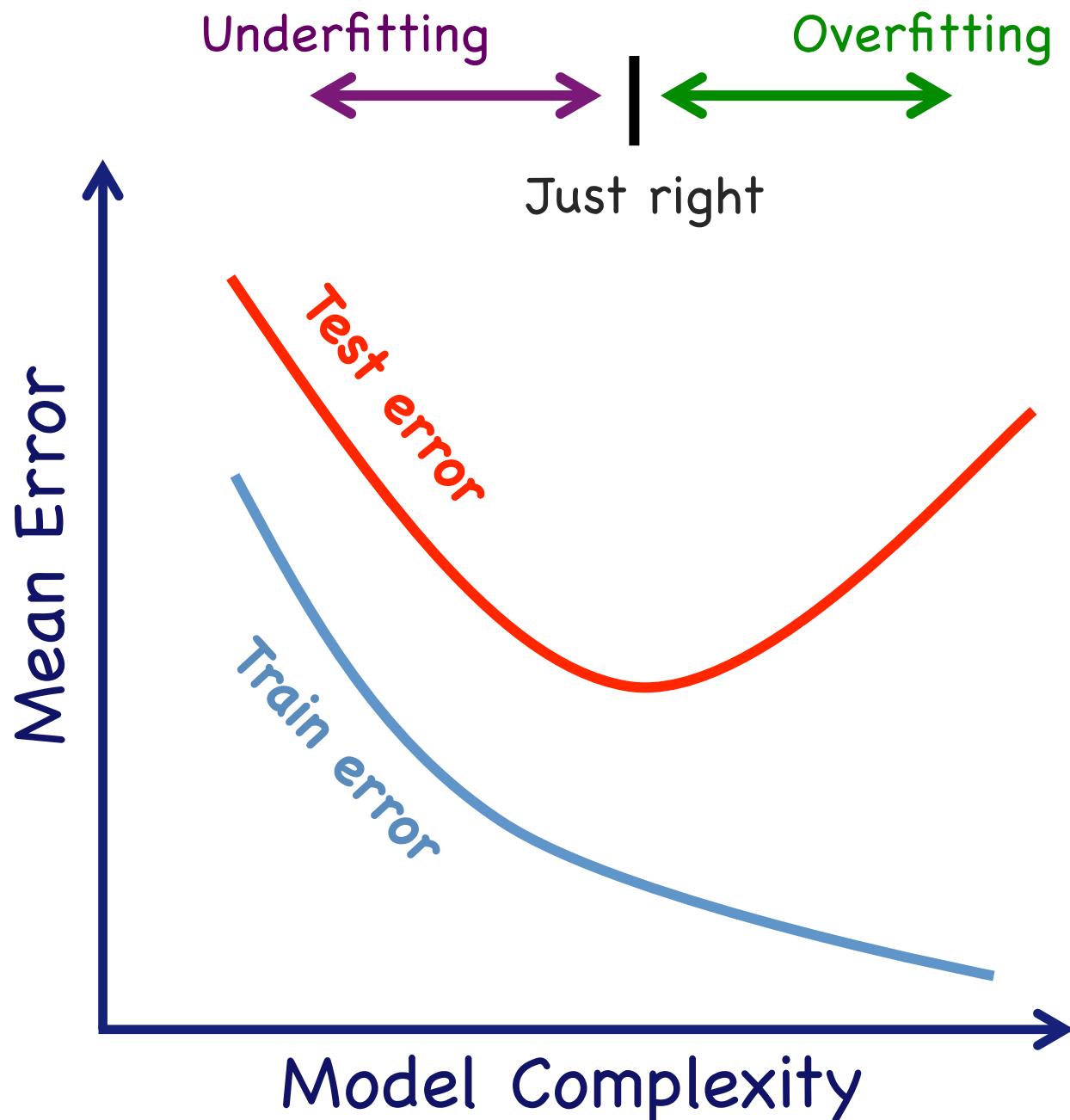


# Train/test dataset split



# Train/test dataset split

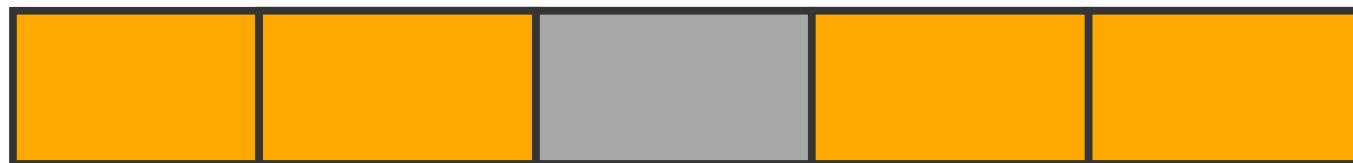




# 5-fold cross-validation

Test

Train



Data  
processing

Shift

# Why do we need?

Age	Salary
24	\$110,000
36	\$130,000
38	\$80,000
44	\$420,000
27	\$420,000
43	\$12,000,000
...	...

# Many ways

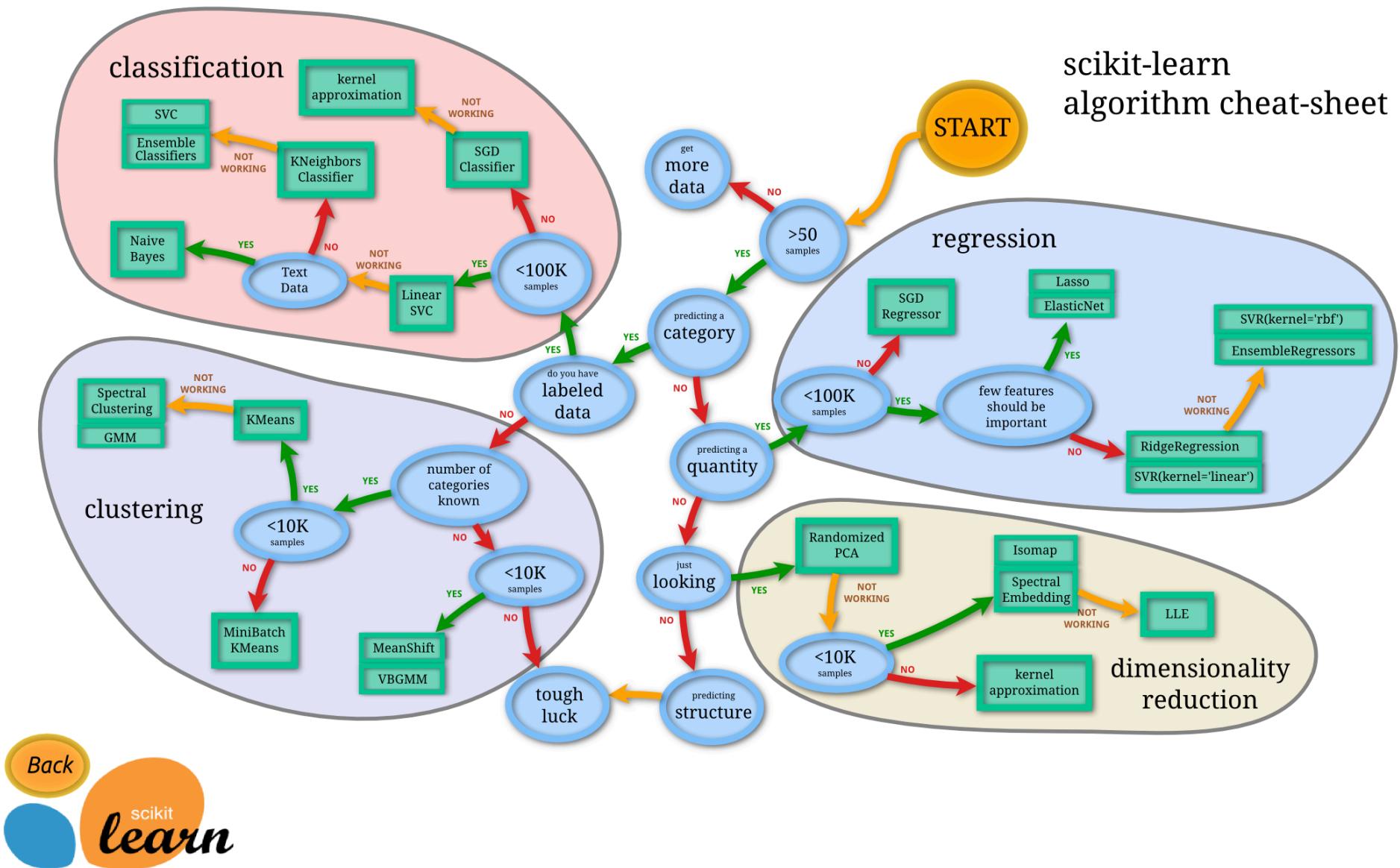
- **Standardization**
  - Zero mean and unit variance
- **Scale to a range**
  - i.e. (0, 1)
- **Normalization**
  - Unit norm
- ...

Check out: <http://scikit-learn.org/stable/modules/preprocessing.html>

Go to notebook 05

# More resources

# scikit-learn algorithm cheat-sheet



Interactive version: [http://scikit-learn.org/stable/tutorial/machine\\_learning\\_map/index.html](http://scikit-learn.org/stable/tutorial/machine_learning_map/index.html)

Conclude with notebook 06