
Python for Data Science Advanced

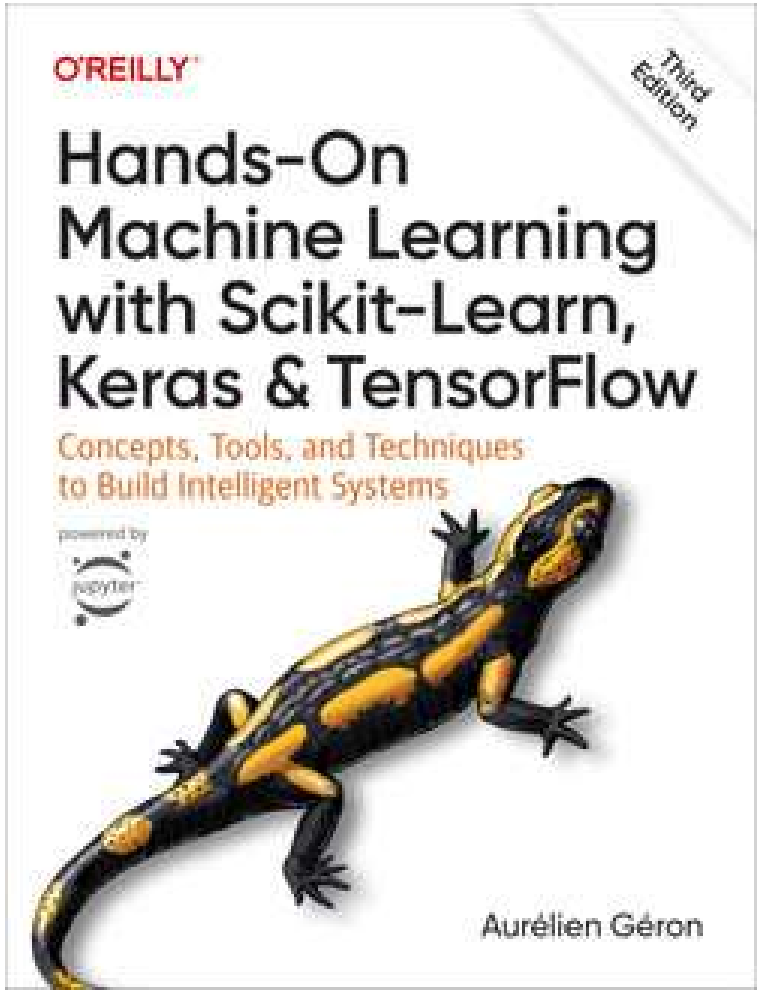
Introductions

Training, materials, trainer

Program

- Part 1 Machine Learning
- Part 2 Regression
- Part 3 Optimalization
- Part 4 Classification
- Part 5 Neural Networks
- Part 6 Unsupervised Learning
- Additional Reinforcement Learning

Material



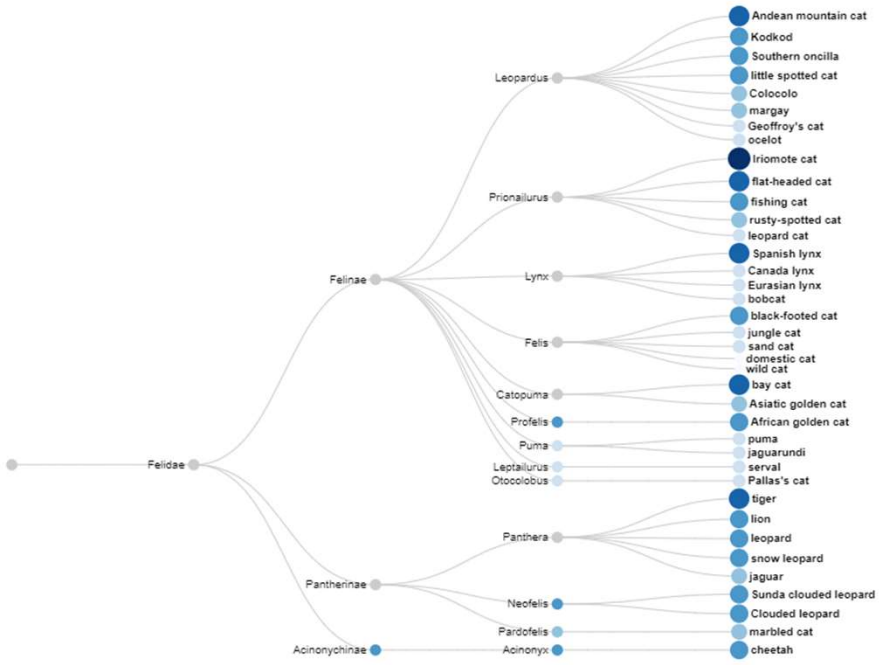
https://github.com/ageron/handson-ml3

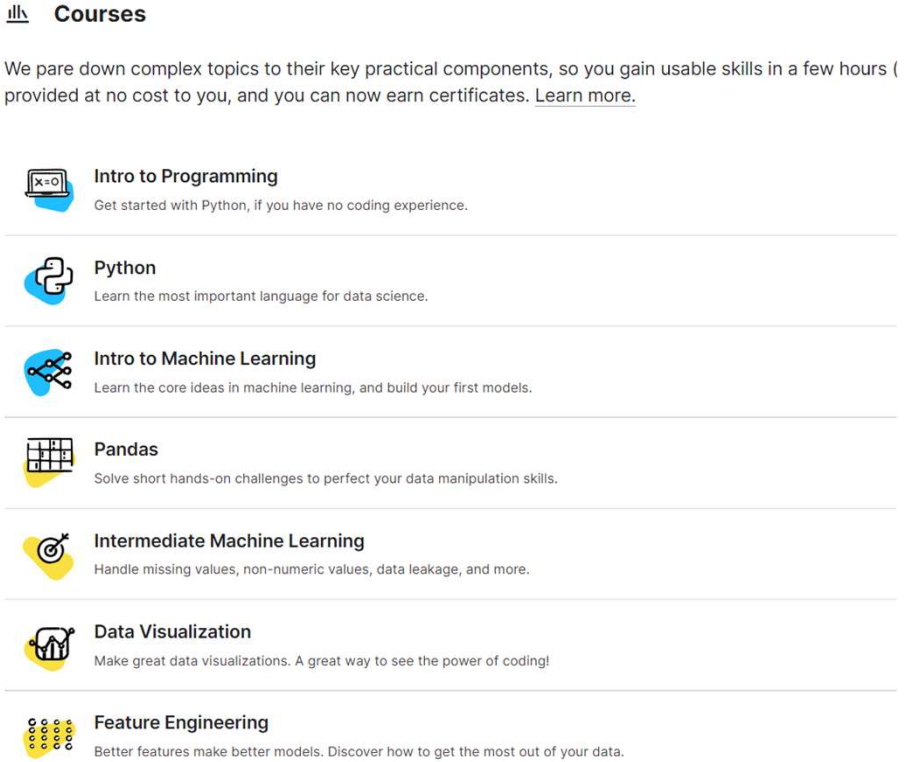
Hello World

- Tessel Haagen
- BSc. & MSc. Artificial Intelligence and Computer Science
- IT Teacher & AI Engineer
- Harry Potter Fan

Materials

- Exercises & slides on E-Connect
- Examples from class:
  https://github.com/TesselHaagen/DataScience-Advanced-11-2024





Part 1

Introduction to Machine Learning

Tools of the Trade

- Python3
- Libraries
- Anaconda (optional)
- Jupyter Notebook
- IPython
- Integrated Development Environments:
  - Spyder
  - PyCharm
  - Visual Studio Code
  - Choose your fighter!
  - Tip: start your IDE in administrator mode if you are on Windows, the installations sometimes require additional permissions.

Numpy

Scipy

Statsmodels

Pandas

Matplotlib

Seaborn

Plotly

Scikit-Learn

TensorFlow

Keras

PyTorch

Jupyter Notebook

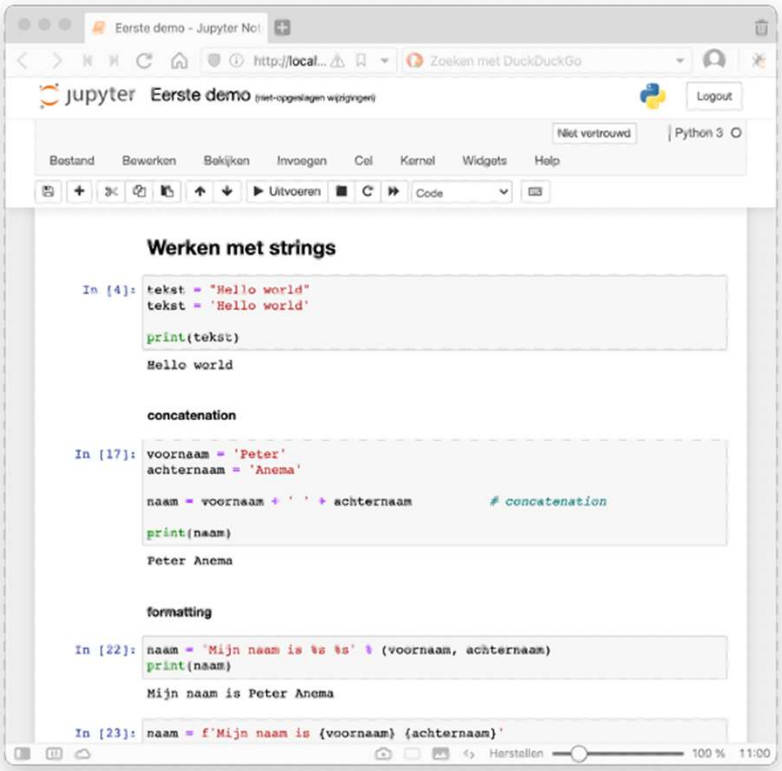
```
$ jupyter notebook
```

- Web-based
- Interactive computational environment where you can combine code execution, text, mathematics, plots and more

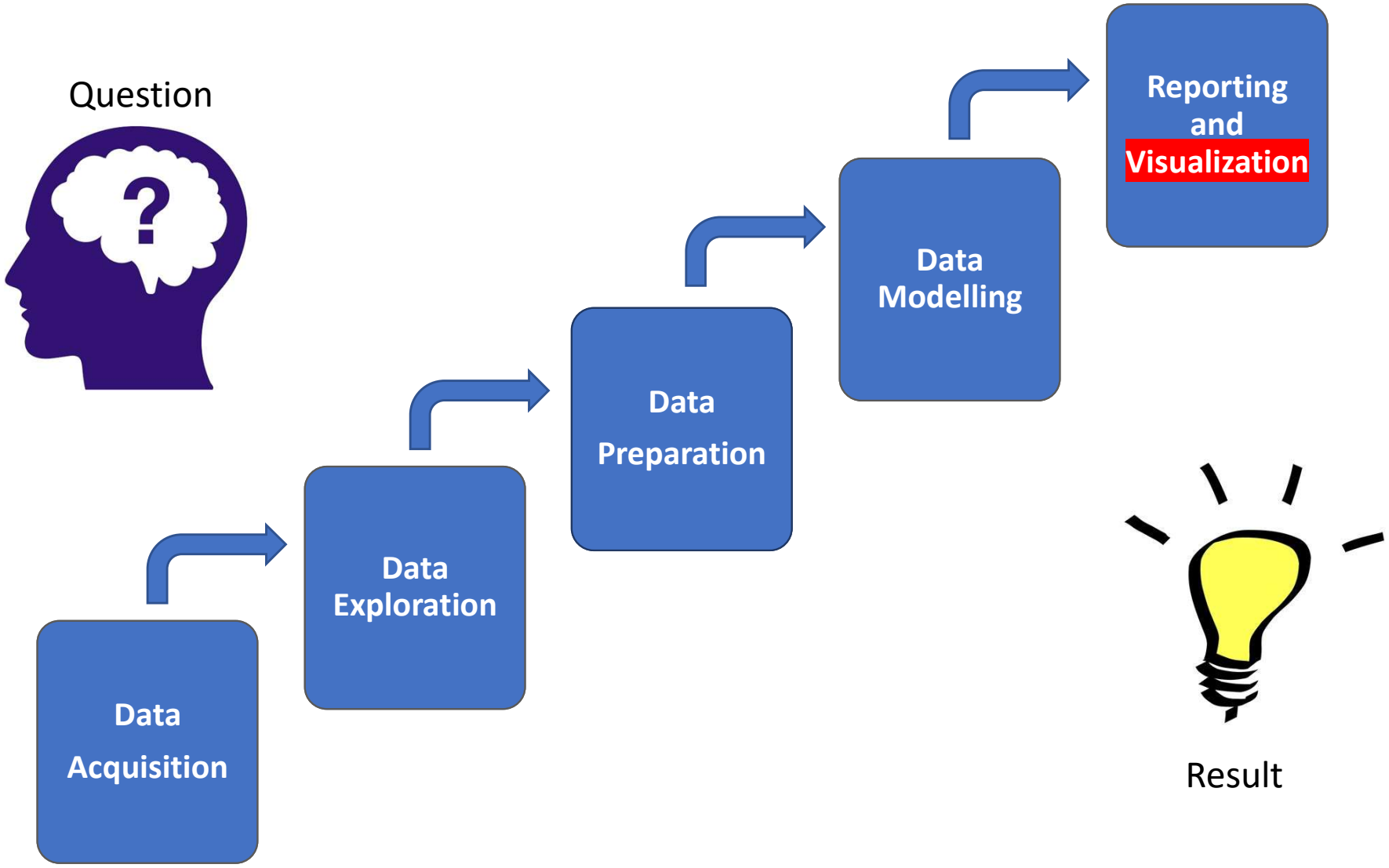
Why (not)?

+ Code documentation & examples

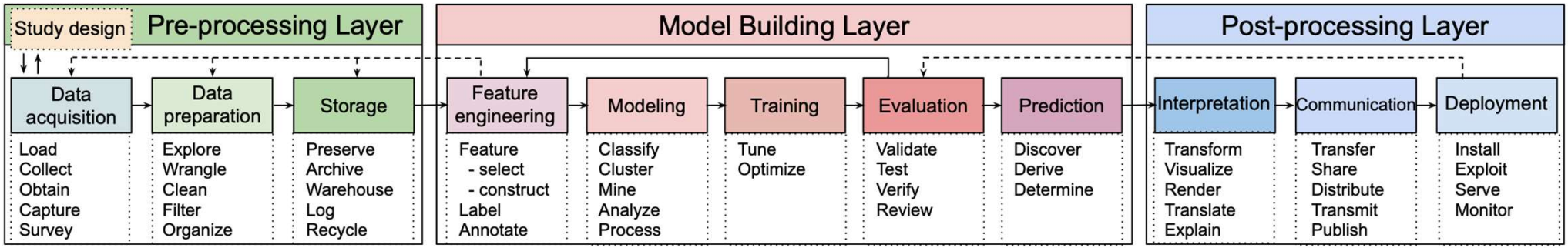
- Running in cells
- Slow development environment
- Not production-ready



Data Science Pipeline



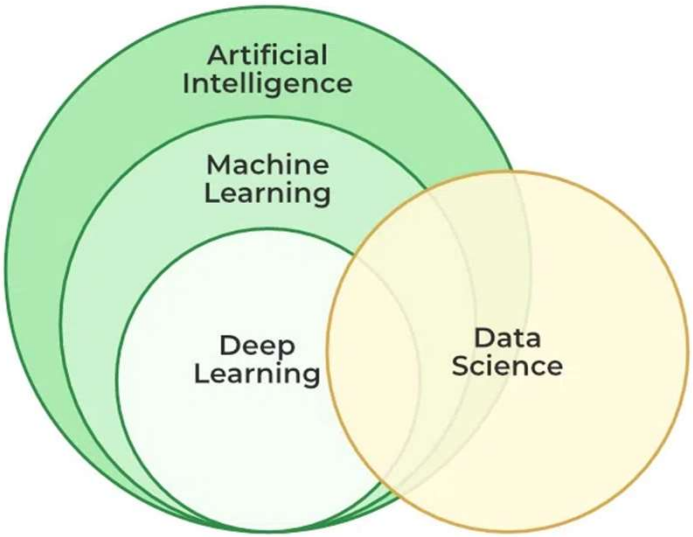
The Data Science Pipeline



https://design.cs.iastate.edu/papers/ICSE-22a/ds-pipeline-icse22.pdf

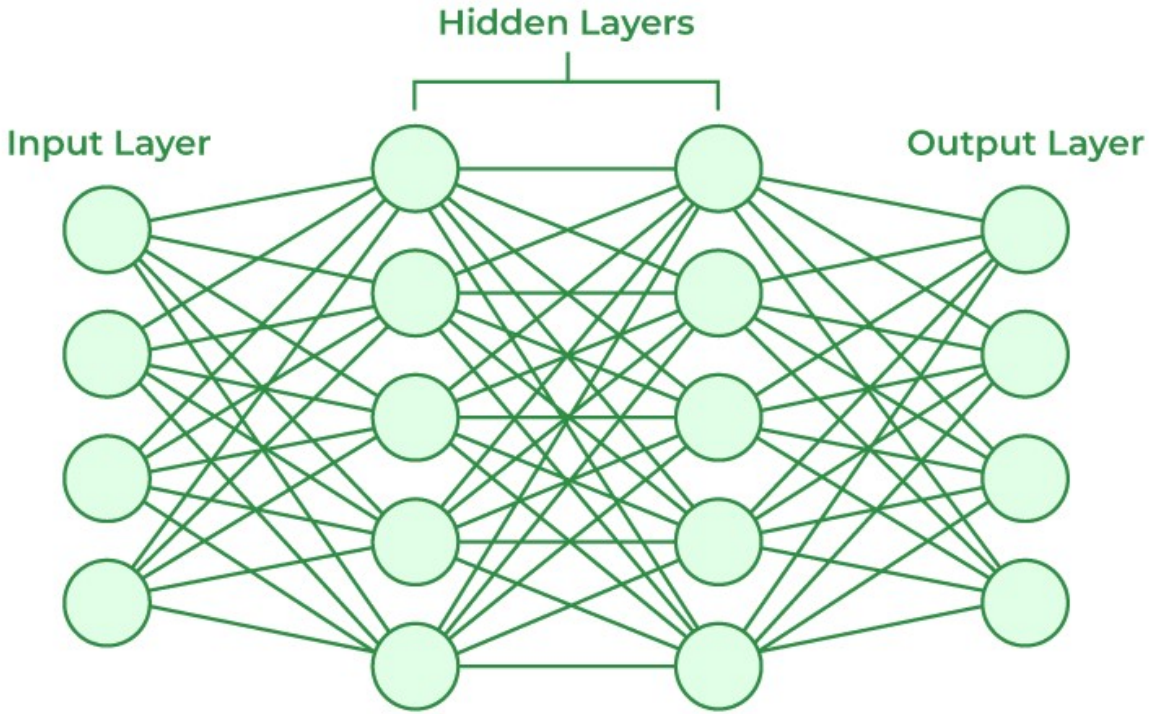
Machine Learning Checklist

- Frame the problem -> big picture
- Obtain/gather data
- Explore the data to gain insights
- Prepare the data
- Explore different models
- Fine-tune your models
- Present your solution
- Launch, monitor and maintain your system



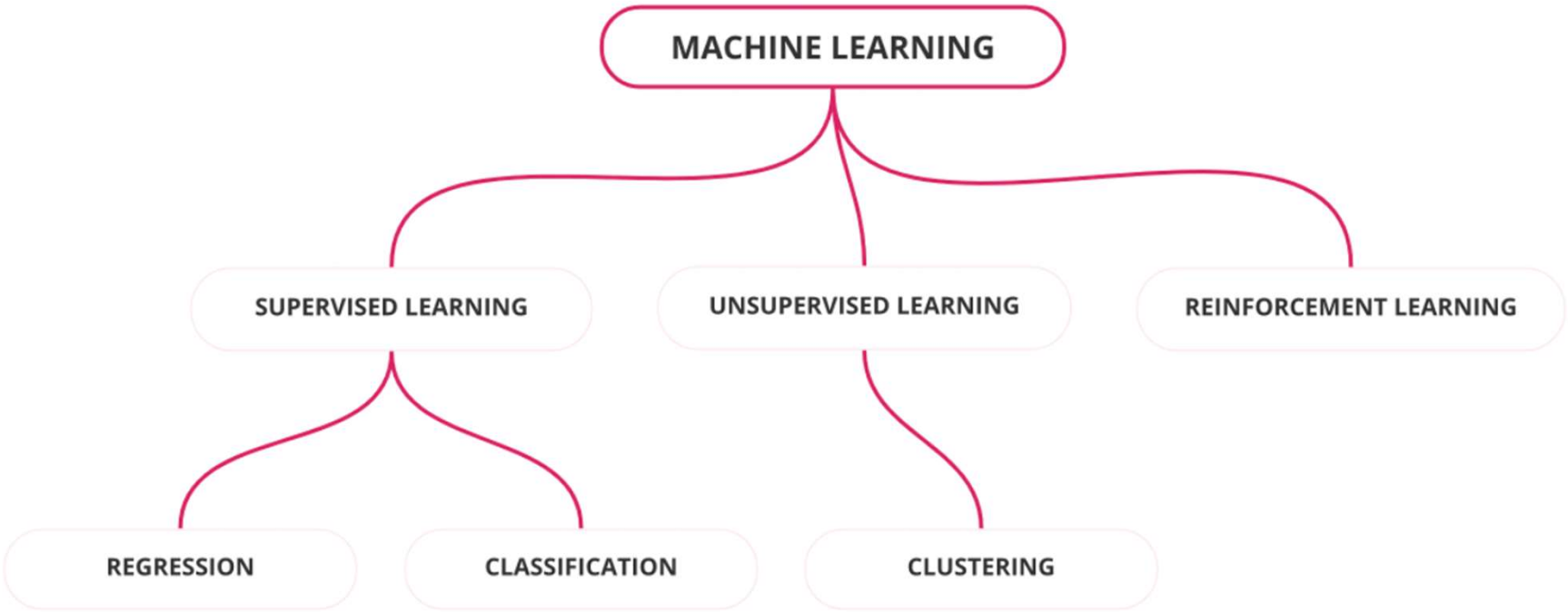
What are machine learning algorithms?

- Tom Mitchell: “A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.”
- In essence: learning from experience
- Usually includes a hard to interpret ‘black box’

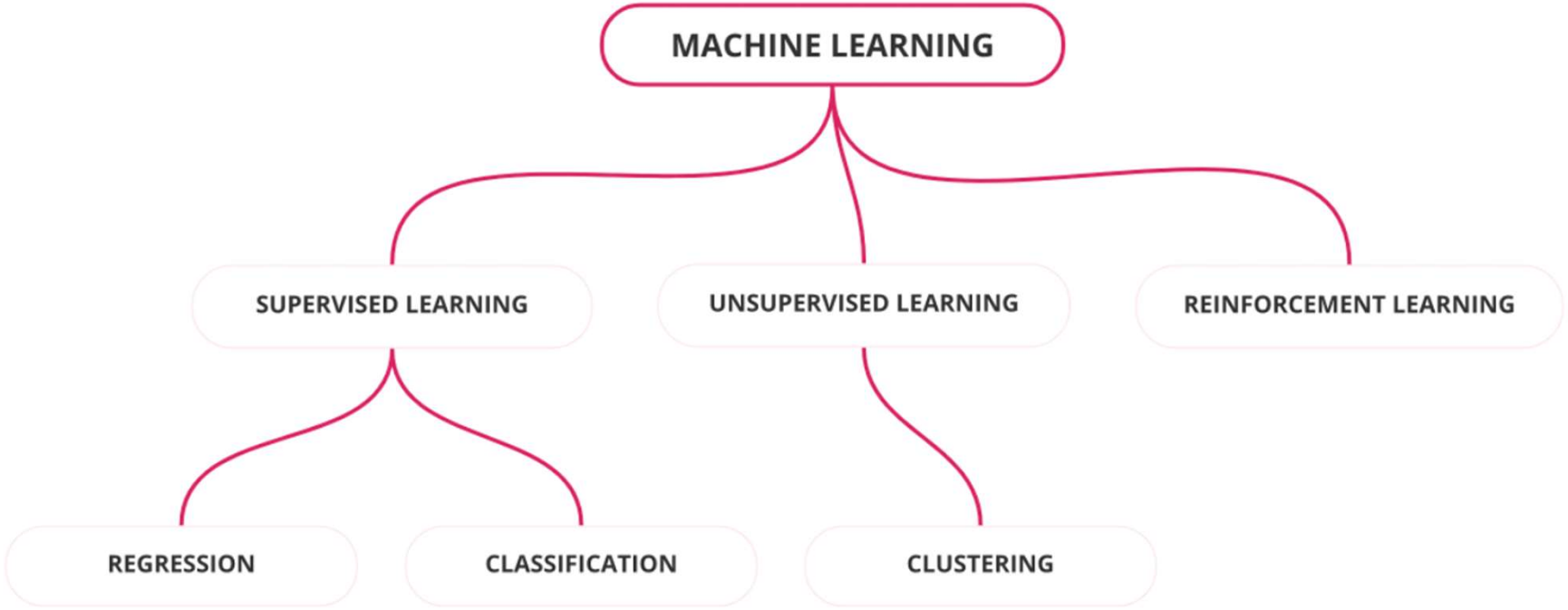


computrain

Machine Learning

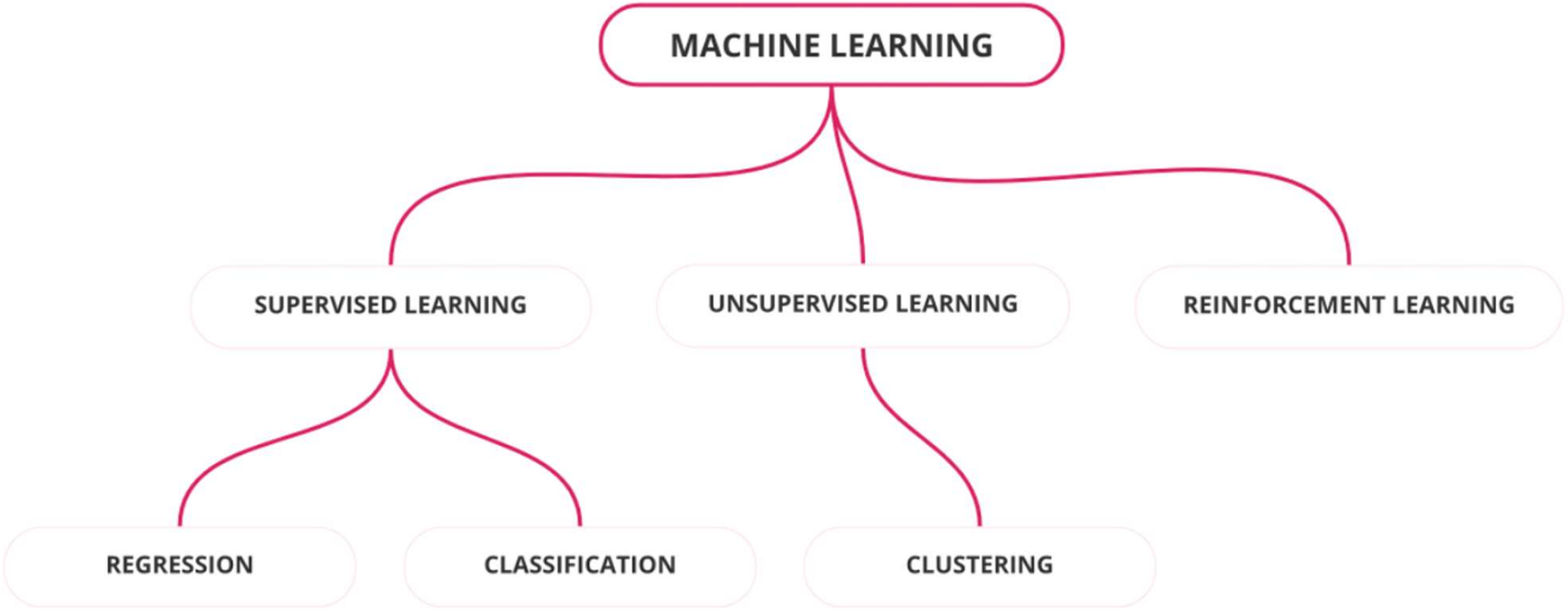


Machine Learning



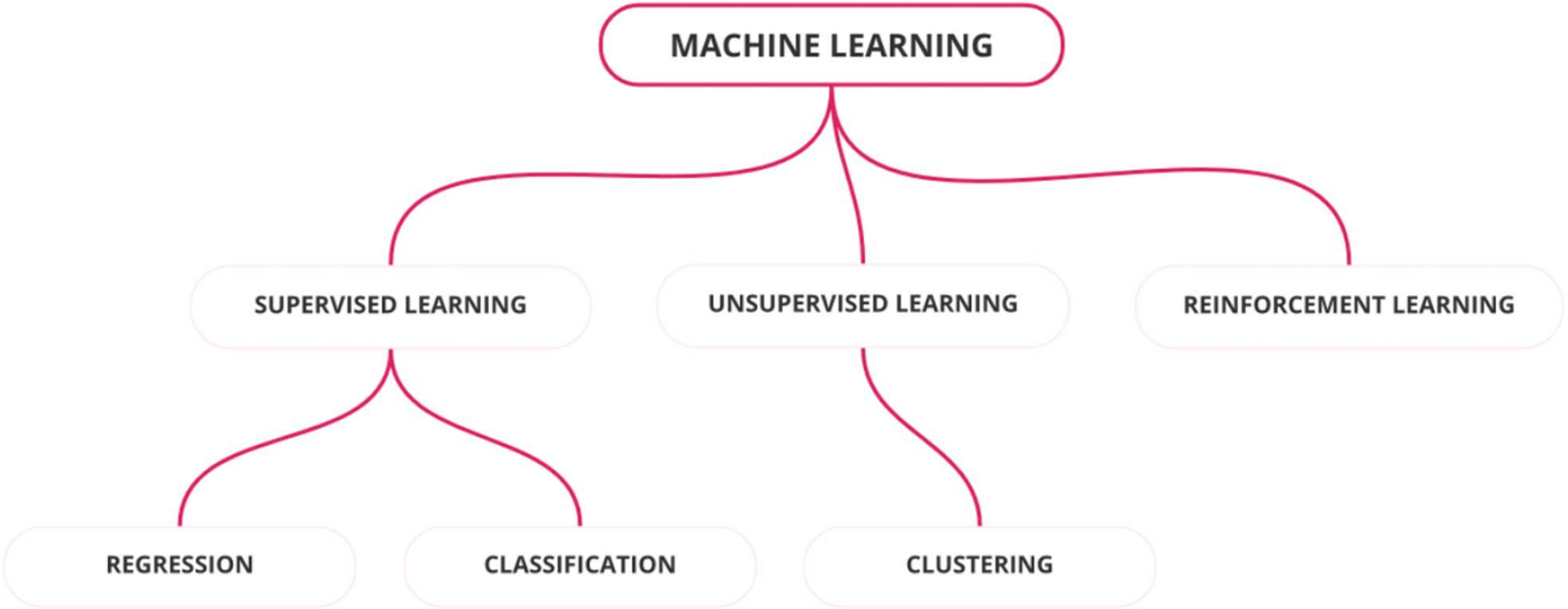
Linear Regression
Support Vector Regression
Gaussian Proces Rregression
K-Nearest Neighbor
Decision Trees
Neural Networks
Ensemble Methodes

Machine Learning

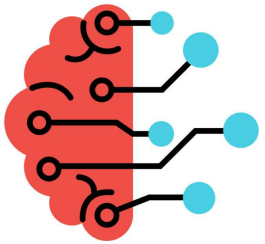


Logistic Regression
Support Vector Machines
Discriminant Analysis
Naive Bayes
K-Nearest Neighbor
Decision Trees
Random Forest
Neural Networks
Ensemble Methodes

Machine Learning

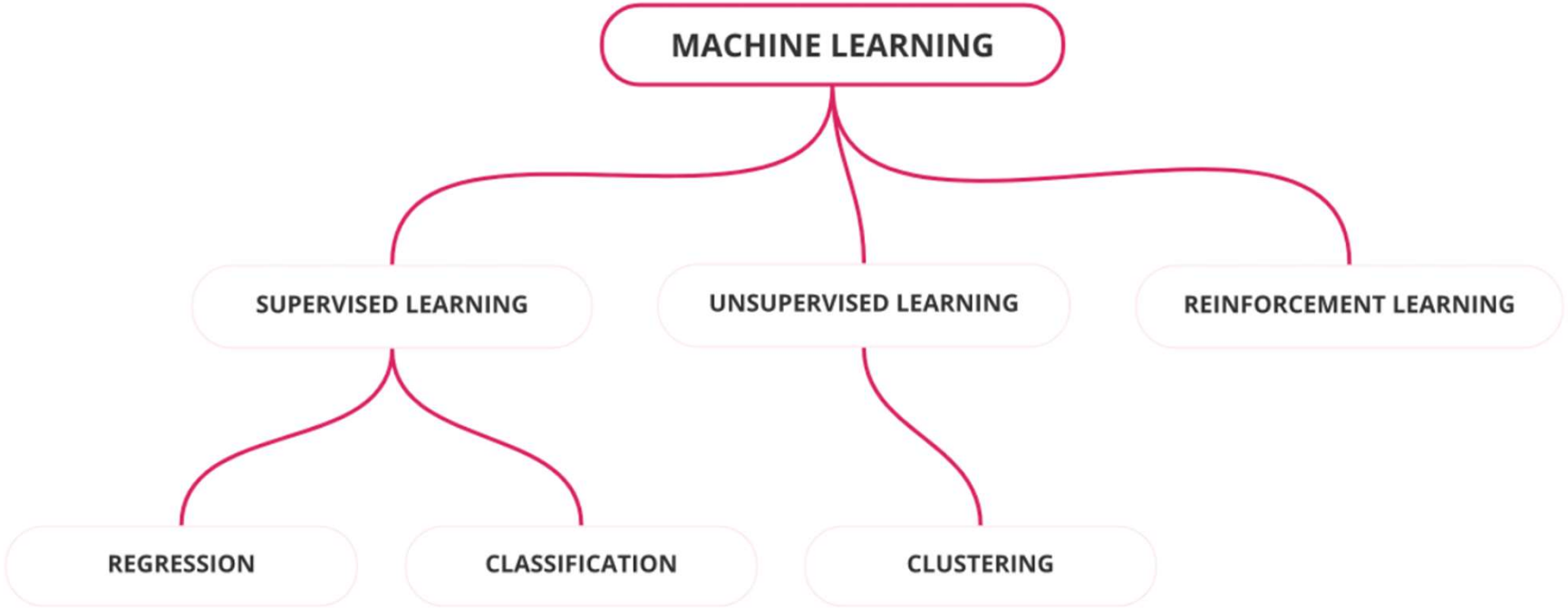


K-Means
K-Median
Hierarchical
Gaussian Mixture
Neural Networks
Hidden Markov Model



NEURAL NETWORK

Machine Learning



Linear Regression	Logistic Regression	K-Means
Support Vector Regression	Support Vector Machines	K-Median
Gaussian Proces Rregression	Discriminant Analysis	Hierarchical
K-Nearest Neighbor	Naive Bayes	Gaussian Mixture
Decision Trees	K-Nearest Neighbor	Neural Networks
Neural Networks	Decision Trees	Hidden Markov Model
Ensemble Methodes	Random Forest	
	Neural Networks	
	Ensemble Methodes	

Neural networks are very universal!

Challenges of Machine Learning

Data Quality and Quantity:

- Insufficient Quantity of Training Data
- Nonrepresentative Training Data
- Poor Quality Data
- Irrelevant Features
- *Big Data*

Model Training Issues

- Overfitting the Training Data (variance)
- Underfitting the Training Data (bias)

Challenges in Computation

- Curse of Dimensionality
- Computational Complexity

Ethical and Social Considerations

- Transparency
- Ethics

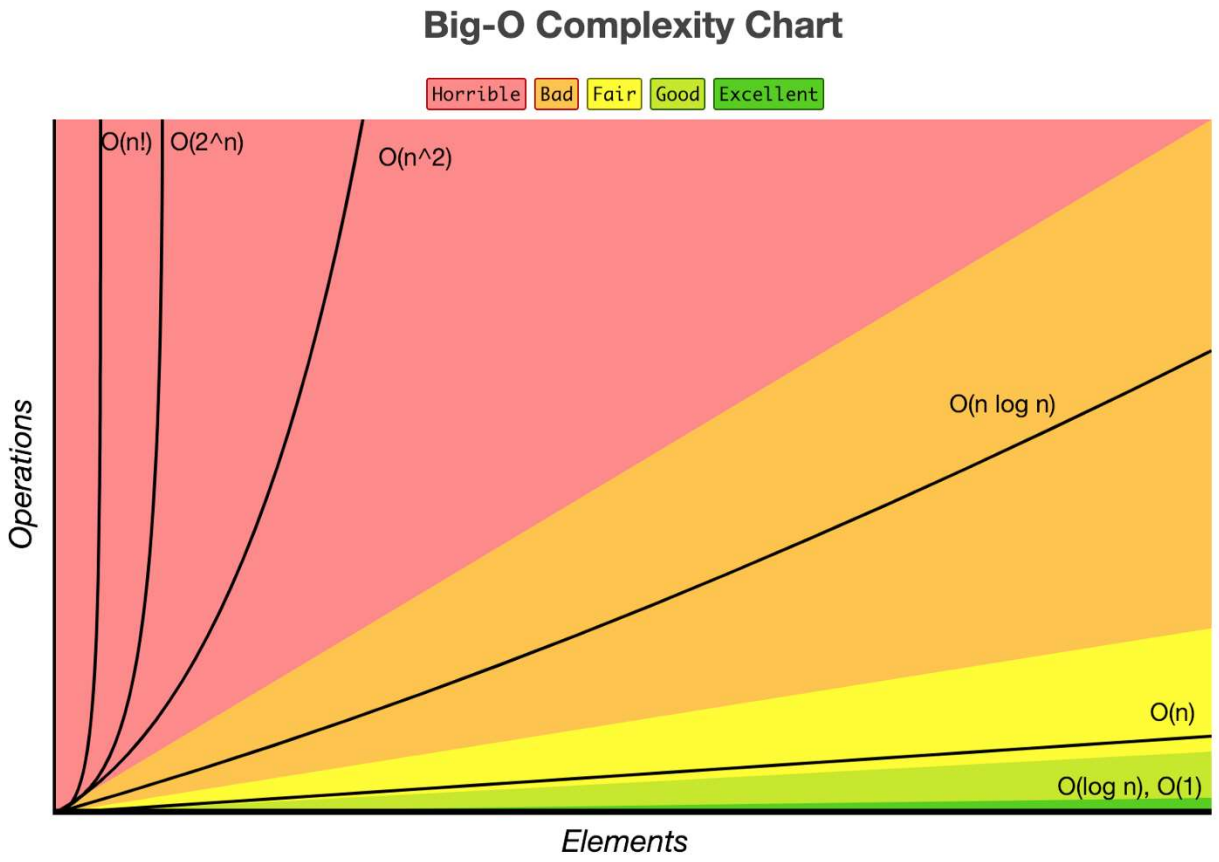
computrain

Computional Complexity

- More input -> more resources needed
- Feasability – Can it ever be computed?
- Big O notation - Landau's symbol to denote algorithm complexity

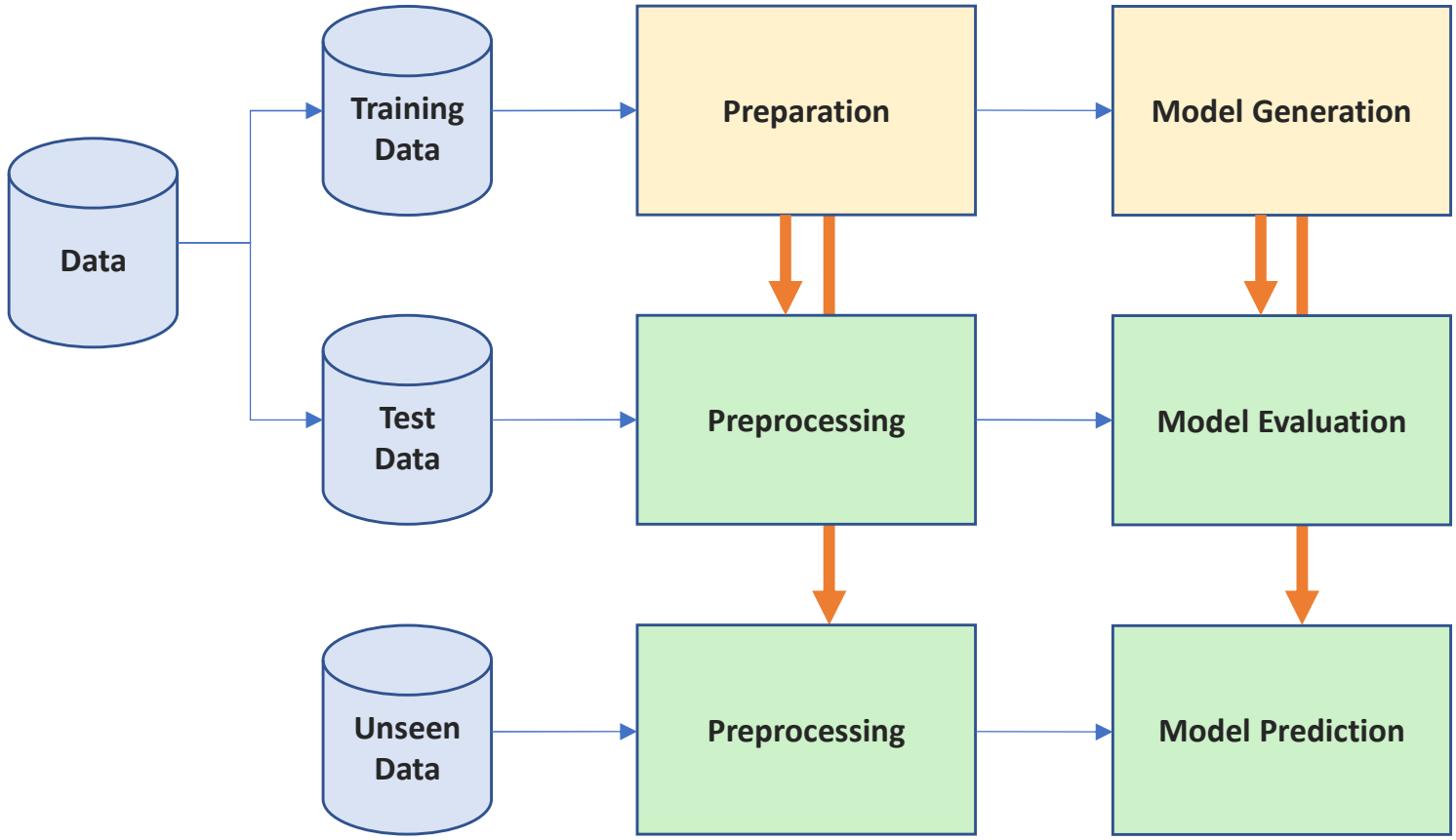
Big O Notation

- Constant O(1)
- Logarithmic O(log n)
- Linear O(n)
- Superlinear O(n log n)
- Polynomial O(n^2)
- Exponential O(2^n)
- Factorial O(n!)



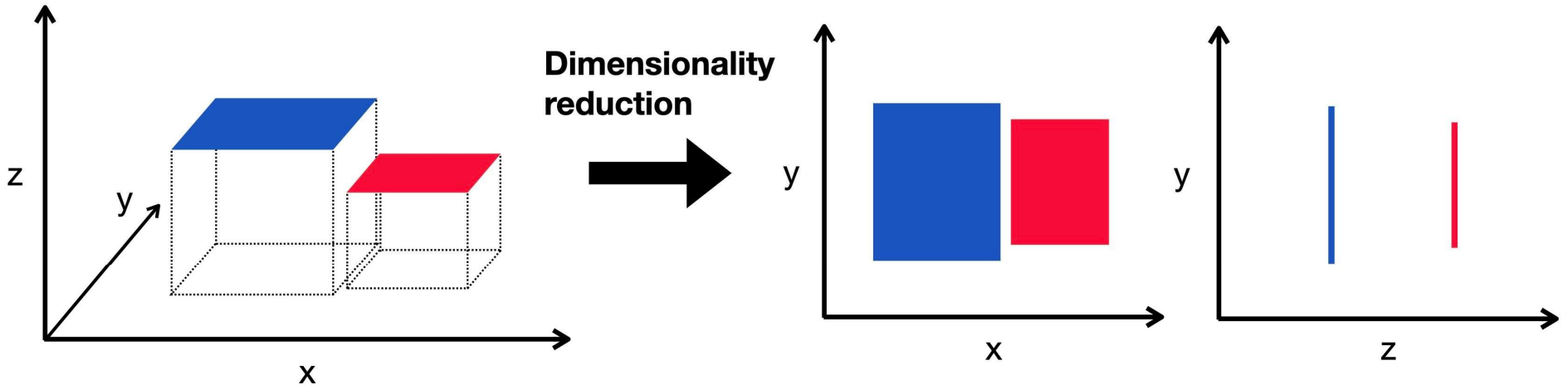
Training and Test Sets

- Split data before training
- Independent training and test subsets



Data Preparation

- Data Cleaning – Missing values, duplicates, incorrect values
- Split training set and test set
- Scaling numeric variables
- Coding categorical variables
- Feature Engineering / Selection
- Dimensionality Reduction (including PCA)



Scaling

Problem: many machine learning algorithms are impacted by large ranges of values

Mean normalisation

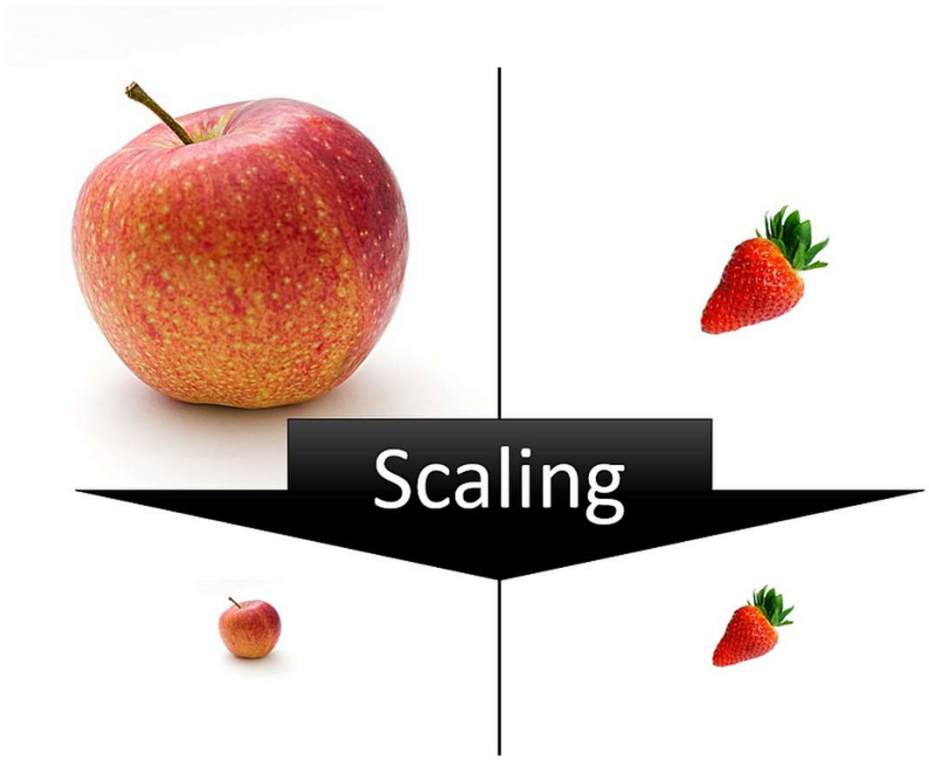
$$\frac{x - Mean(feature)}{Max(feature) - Min(feature)}$$

Standardization or z-score

$$\frac{x - Mean(feature)}{StandardDeviation(feature)}$$

Min-max scaling

$$\frac{x - Min(feature)}{Max(feature) - Min(feature)}$$



computrain

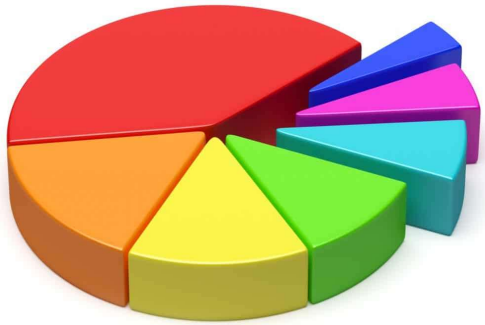
The categorical variable

Definition:

Data divided into multiple categories or labels (often strings in Computer Science).

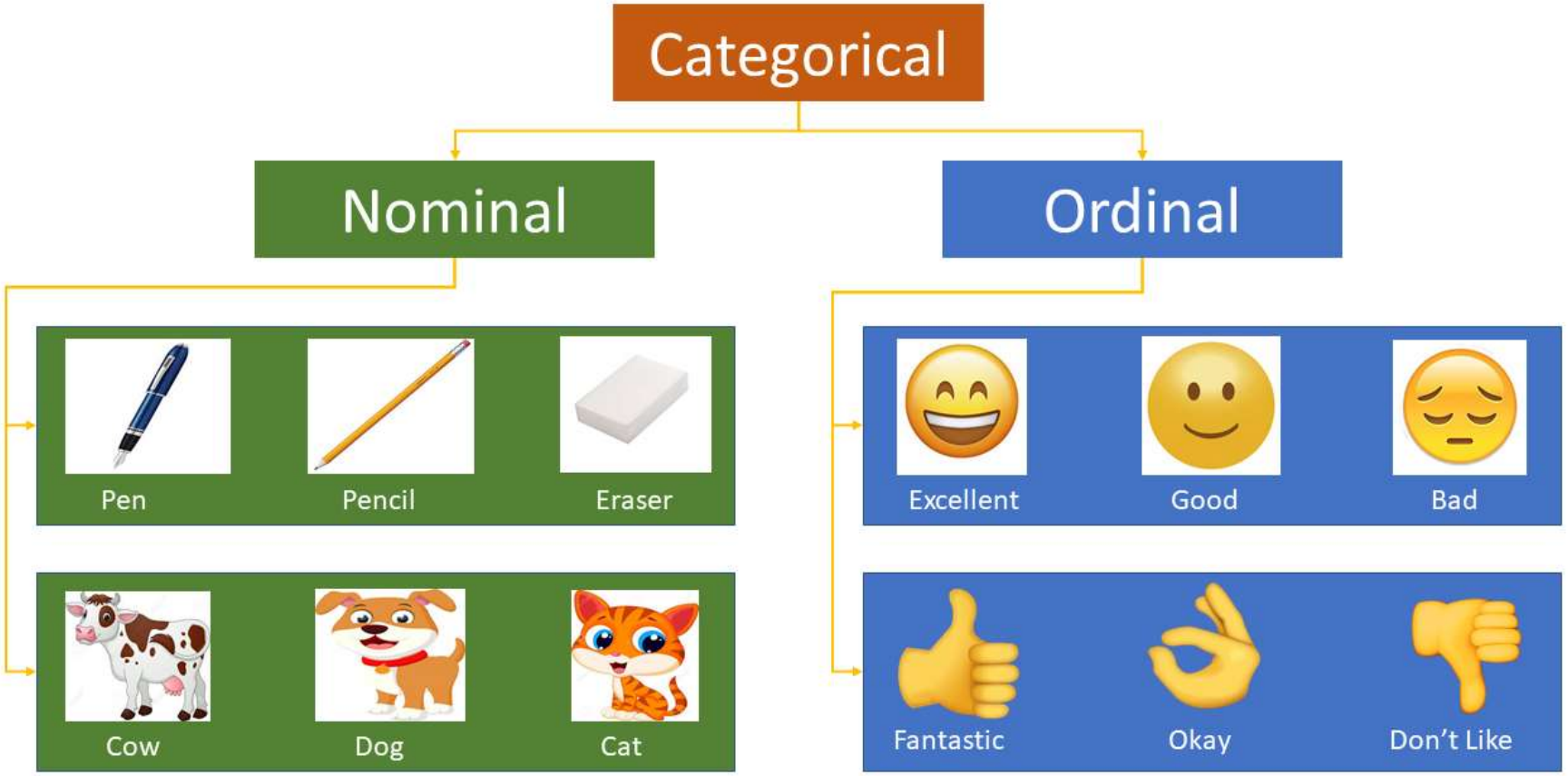
Examples:

- Hair color (usually “blonde”, “red”, “brown”, etc.)
- Some grading systems
- Rankings (think tournament)



But: computers & algorithms much prefer numerical values

Categorical variables: nominal vs. ordinal



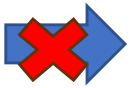
This distinction tells us more about the relationship between different possible values

Categorical Variables – Label encoding

- Label coding is assigning a number to each possible value
- Example: coding a category of
- Implies a linear relation (ordinal relationship)

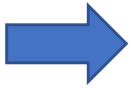
  -> if the number is twice as big, then the category is twice as big
- What does that mean for encoding?

Item_Category
Fitness
Food
Kitchen
Food



Item_Code
1
2
3
2

Performance
Very poor
Poor
Average
Good
Excellent



Performance_code
1
2
3
4
5

Note: this still only works if we can quantify the difference between these different labels

computrain

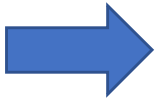
Categorical Variables – One Hot Enc.

Another option:

One-hot encoding

- Adds a feature for every possible value in the category (dummy column)
- **Dummy Variable Trap** (highly correlated variables encoded separately)
- Curse of dimensionality

Item_Category
Fitness
Food
Kitchen
Food



Fitness	Food	Kitchen
1	0	0
0	1	0
0	0	1
0	1	0

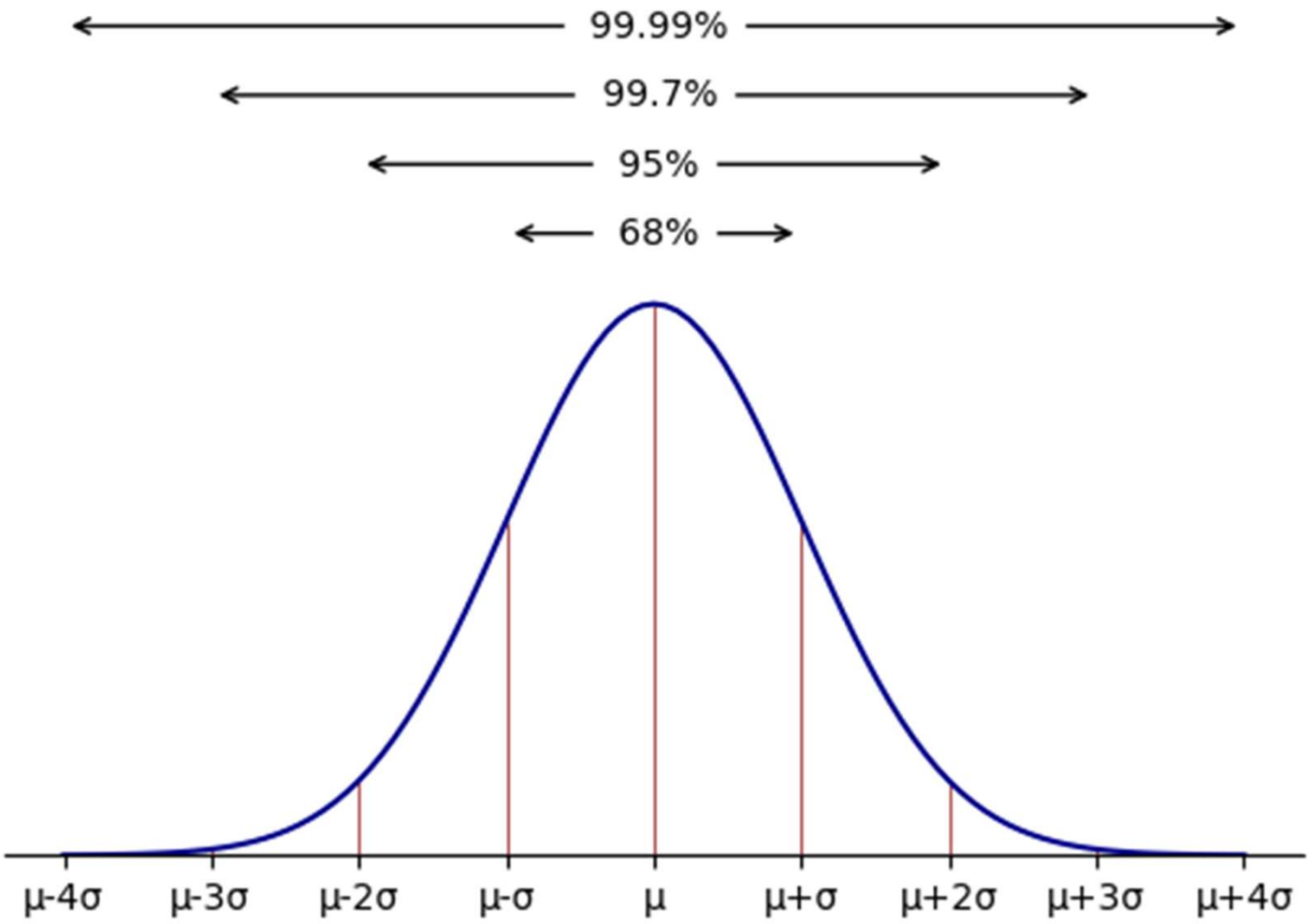
Food for thought: are any of these categories related? How can we capture similarity?

Statistical Metrics for data insight

Next step: gather insights

- Central measures/measures of central tendency
  - Mean
  - Median
  - Mode
- Measures of dispersion/measures of variability
  - Variance
  - Standard deviation
  - Inter Quantile Distance
- Skewness
  - Symmetrical/asymmetrical distributions
  - Positively or negatively skewed

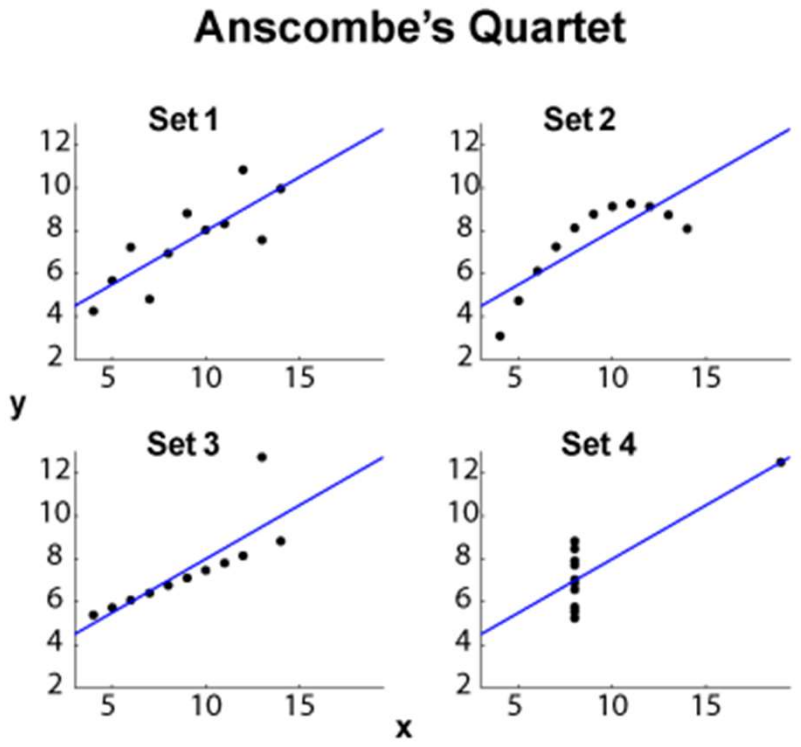
Normal Distribution



Testing for Normality

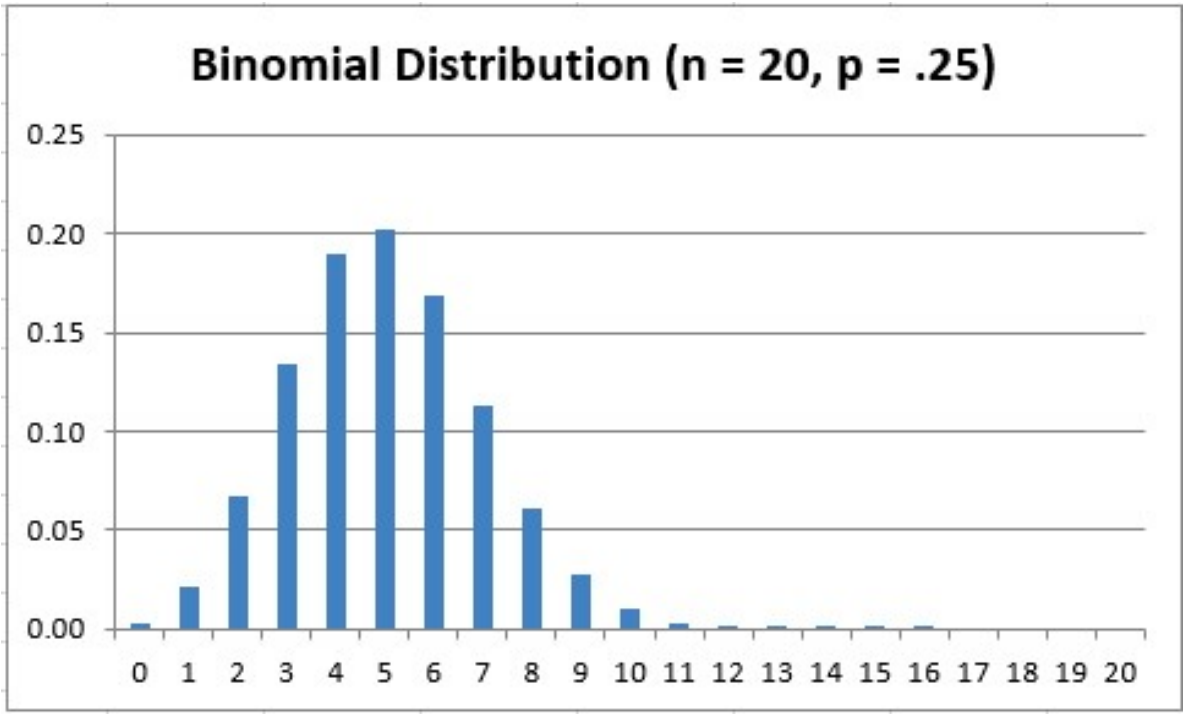
Preferrably normality is assessed both visually and through normality tests

- Histogram
- Boxplot
- QQ Plot
- Shapiro Wilk test (highly recommended)
- Kolmogorov Smirnov test
- Lilliefors test



https://www.statology.org/normality-test-python/

Binominal Distribution



Correlation and Covariance

- Both measure the **linear relationship** and the **dependency** between two variables
- *Covariance* indicates the **direction** of the relationship
- *Correlation* measures **strength** and **direction** of the relationship
- Correlation is a function of the covariance
- Correlation values are **standardized** whereas, covariance values are **not standardized**.

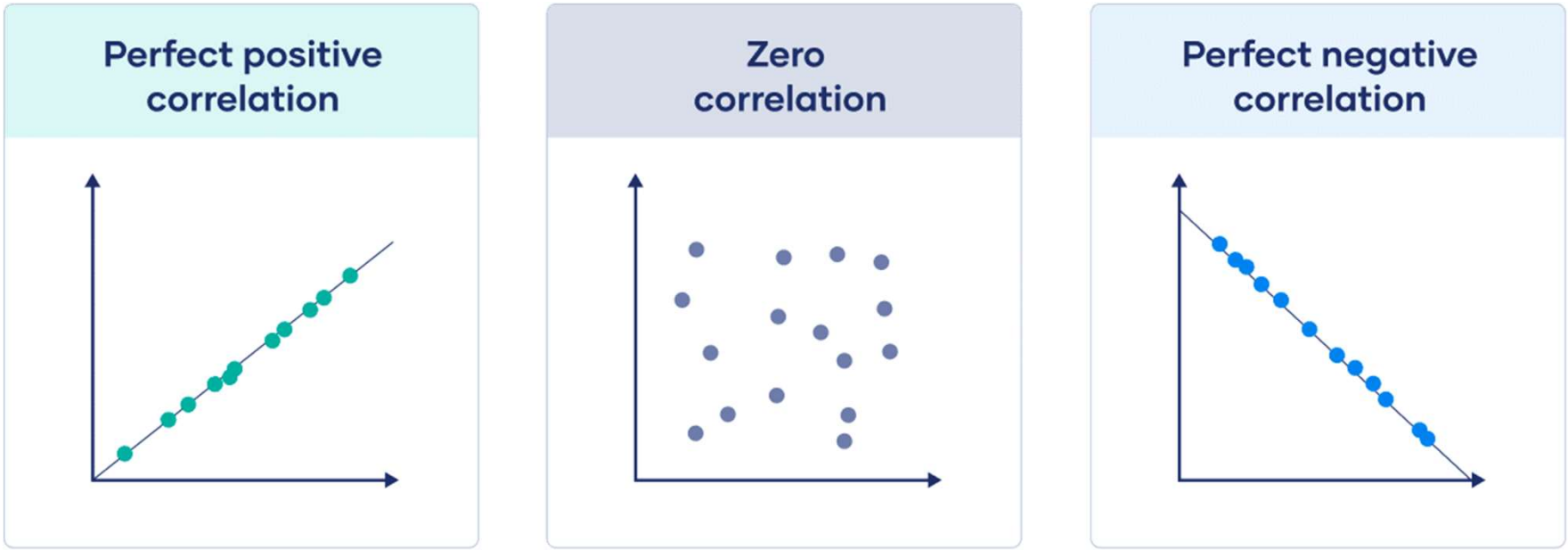
$$\mathbf{VAR(X)} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - E(X))^2$$

$$\mathbf{COV(X,\ Y)} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - E(X))(y_i - E(Y))$$

$$\mathbf{COR(X,\ Y)} = \frac{\mathbf{COV(X,\ Y)}}{\sqrt{\mathbf{VAR(X)VAR(Y)}}}$$

Correlation Coefficient

- Pearson's Correlation Coefficient
- Spearman's Correlation Coefficient



https://www.scribbr.com/statistics/correlation-coefficient/

Correlation

- Pearson product-moment correlation coefficient

$$\rho_{X,Y} = \operatorname{corr}(X,Y) = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{\operatorname{E}[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$$

- Sample correlation coeficient:

$$r_{xy} \overset{\text{def}}{=} \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{(n-1)s_x s_y} = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2 \sum_{i=1}^{n}(y_i-\bar{y})^2}},$$

computrain

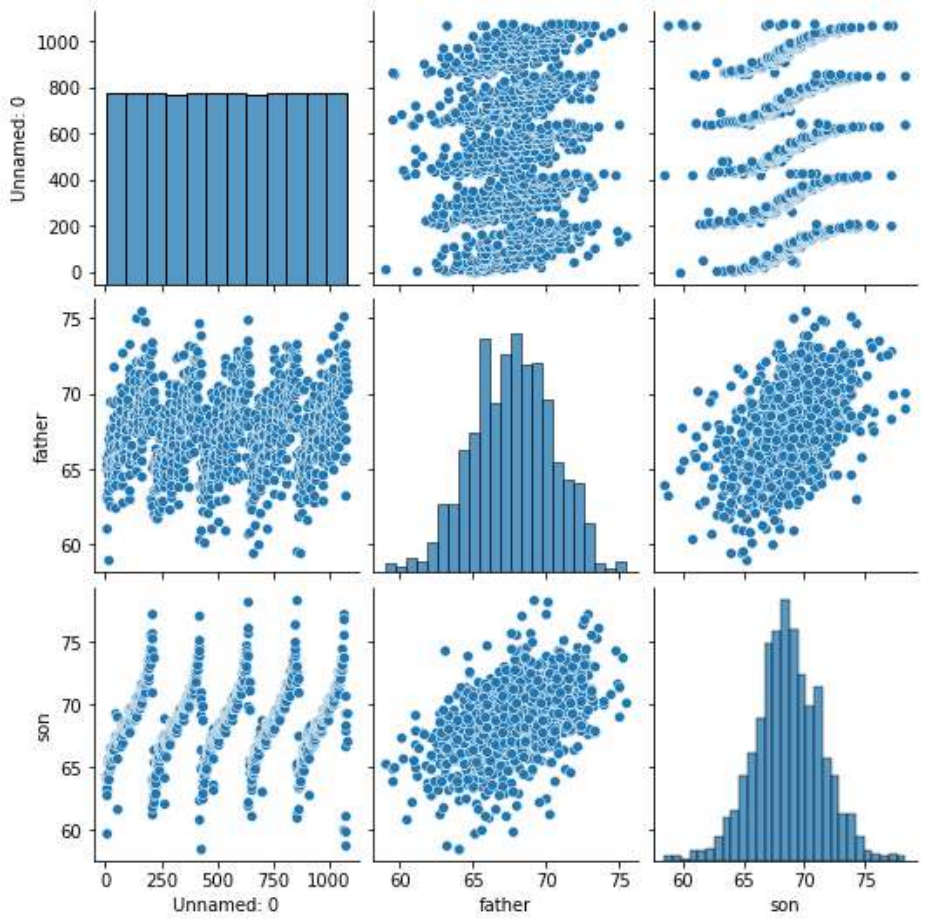
Exercise: data insights

- Download Pearson Father Son dataset (e.g. from Kaggle)
- Import into a Pandas dataframe
- Explore the dataset
  - How many columns and rows are there? What types of variables do we have?
- Display histograms of the different column values: what do you think their distributions are?
- Display scatter plot of height of a father versus height of a son
- Calculate Pearson's correlation coefficient
- Use a student t-test to prove if there is a difference between fathers and sons

This should all be pretty basic, but will form the foundation for our next exercise.

Pearson Father & Son results

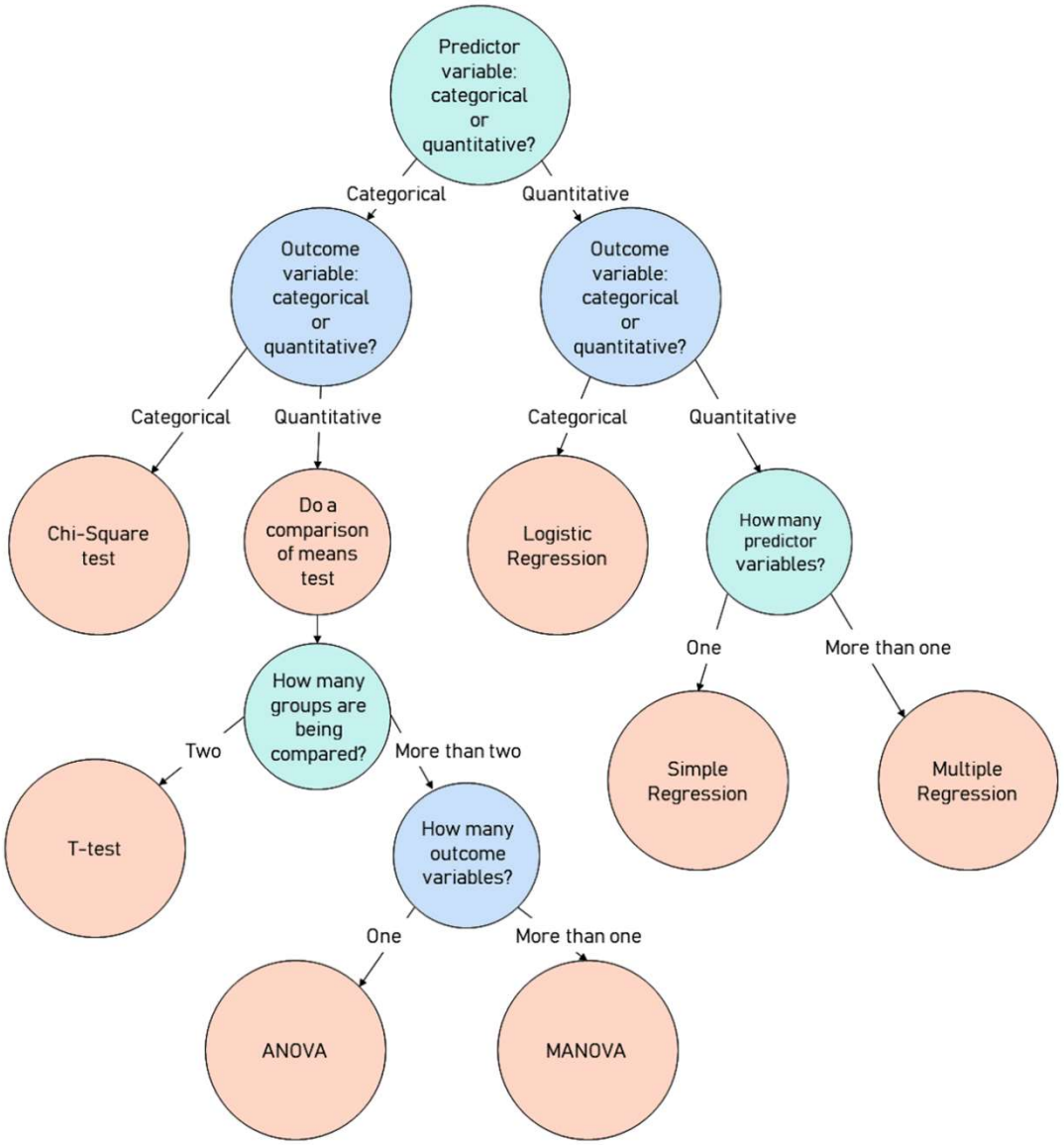
	Unnamed: 0	father	son
Unnamed: 0	1.000000	0.113470	0.153783
father	0.113470	1.000000	0.501338
son	0.153783	0.501338	1.000000




```
TtestResult(statistic=-8.325919805445766, pvalue=1.4675075880562424e-16, df=2154.0)
```

Statistical Tests

- Null hypothesis H0
- Alternative hypothesis Hα
- Quantitive vs. Categorical data
- Student T-test
  - Paired
  - Left, Right or Two Tailed
- Chi-Squared test
- ANOVA



Part 2

Regression Algorithms

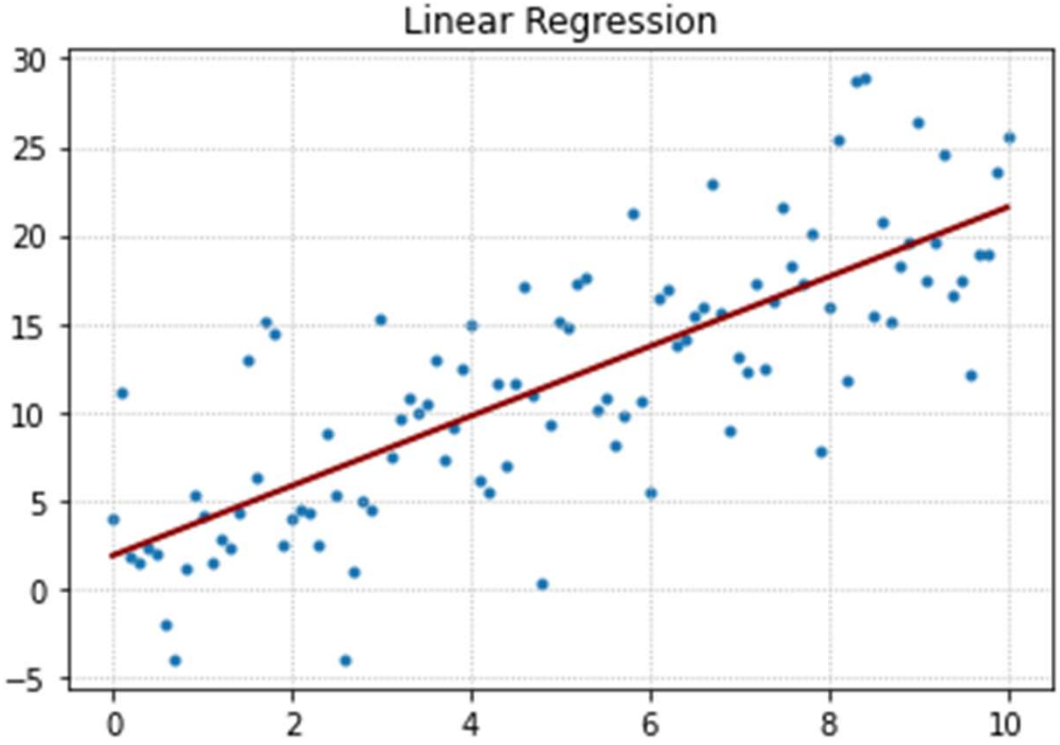
Part 2 - Regression

- The task is to predict a numerical value given a set of input values.

- Linear Regression
- Multiple Regression
- Evaluation Metrics

Linear Regression

- Input features
- Number of dimensions



- Ordinary Least Squares (OLS) optimalisation of coefficients

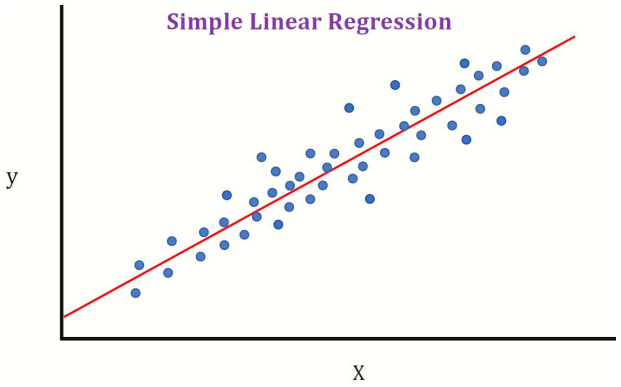
  -> minimizes MSE

computrain

Linear Model

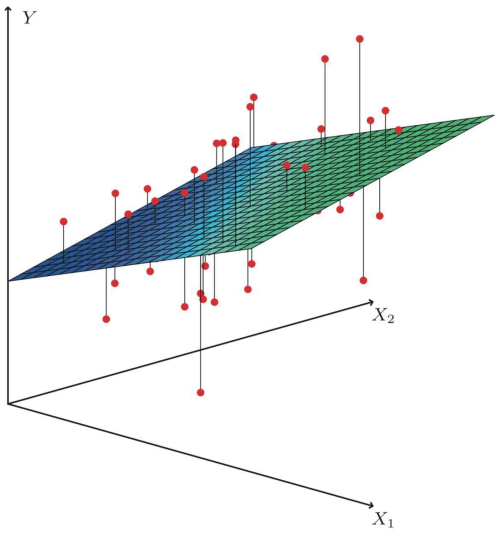
1 – dimensional

- $y = b_0 + b_1.x$



N – dimensional

- $y = b_0 + b_1.x_1 + b_2.x_2 + \ \ldots \ + \ bn.x_n$
- $y = b_0 + \sum_{i=1}^{n} \ bi \ .xi$



Regression

- Linear Regression
- Multiple Regression
- Non-linear Regression
- Custum model optimalisation

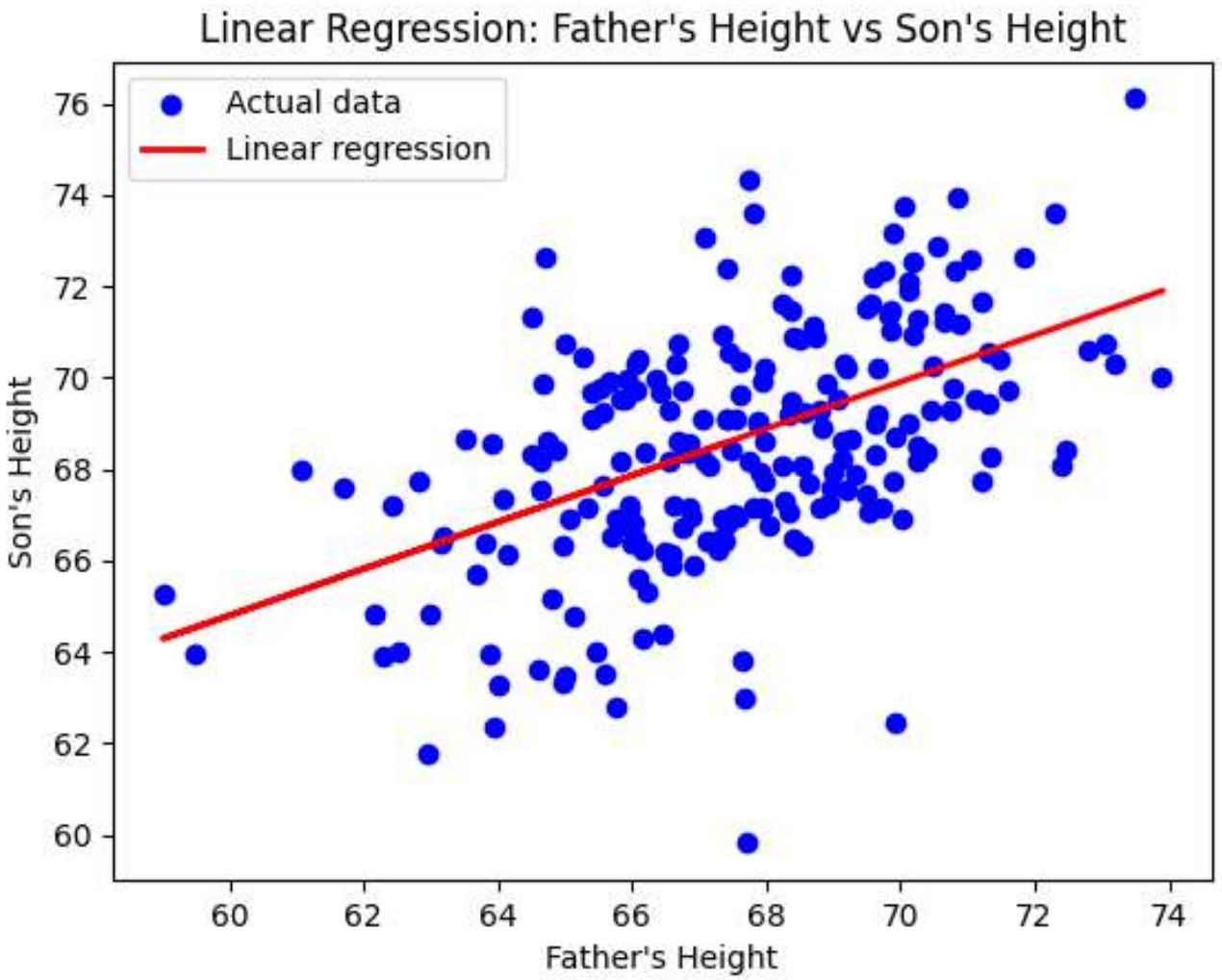
Exercise: linear regression (1 variable)

- Continue from the previous exercise
- Calculate the coefficients for the linear regression
  - Use scikit-learn (LinearRegression is the model)
  - Calculate the coefficients directly
- Visualize the linear regression line using the coëfficients you found

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
```

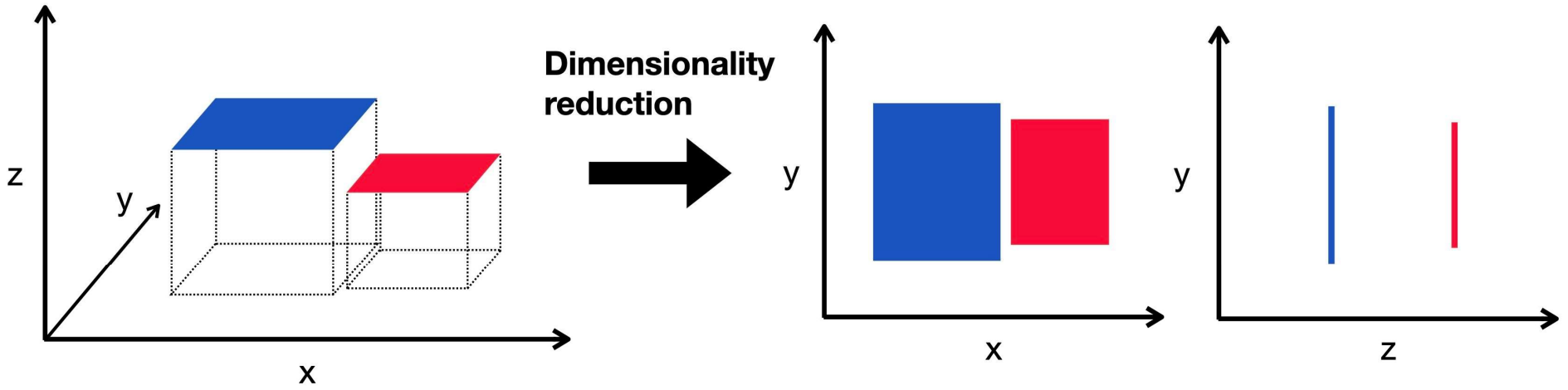
Tip: you might need the following imports

Linear Regression Results

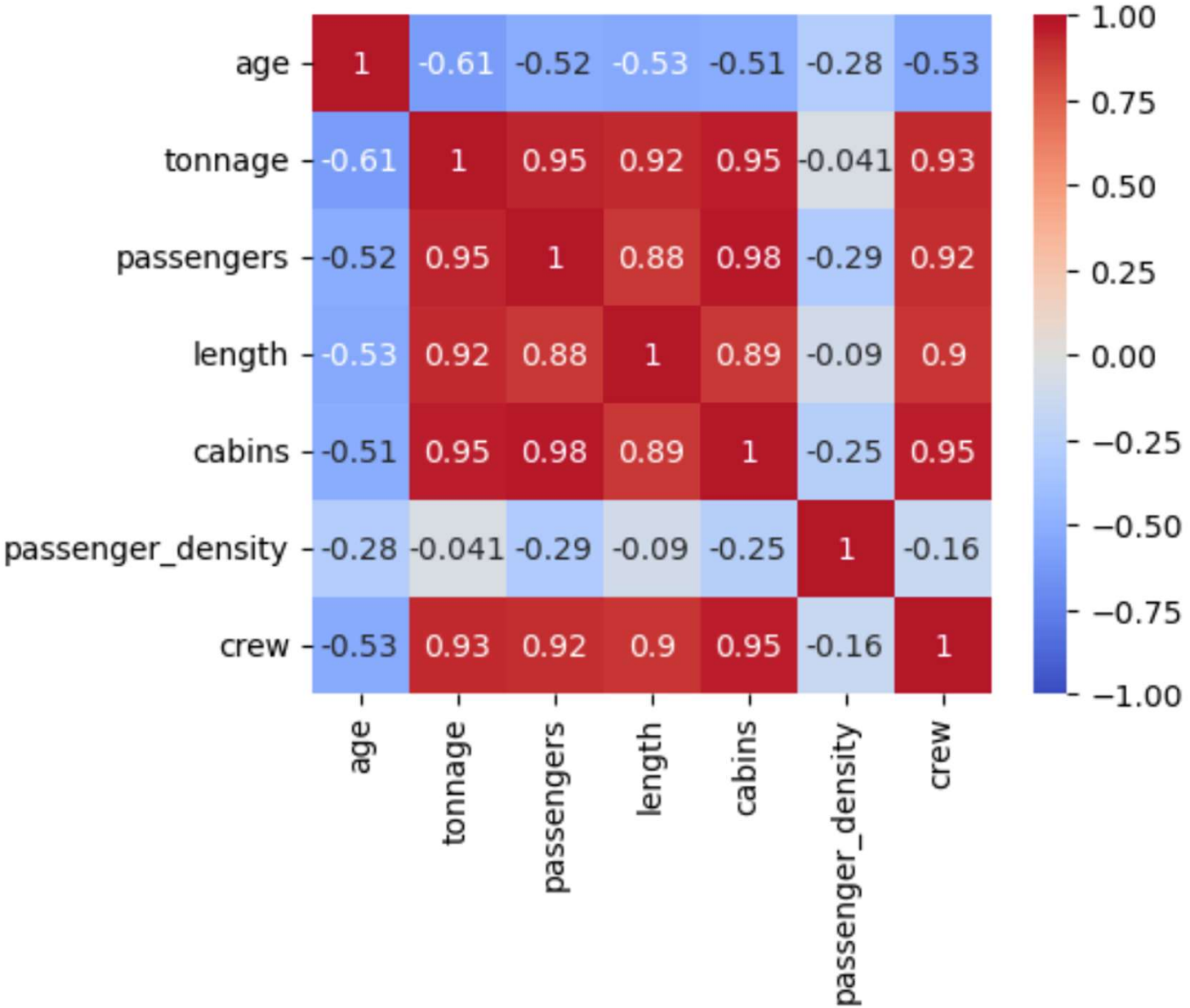


Feature Engineering

- Dimensionality Reduction
  - Principle Component Analysis
- Feature Selection
  - Filter method
  - Wrapper Method
  - Embedded Method
    - Ridge
    - Lasso
    - ElasticNet

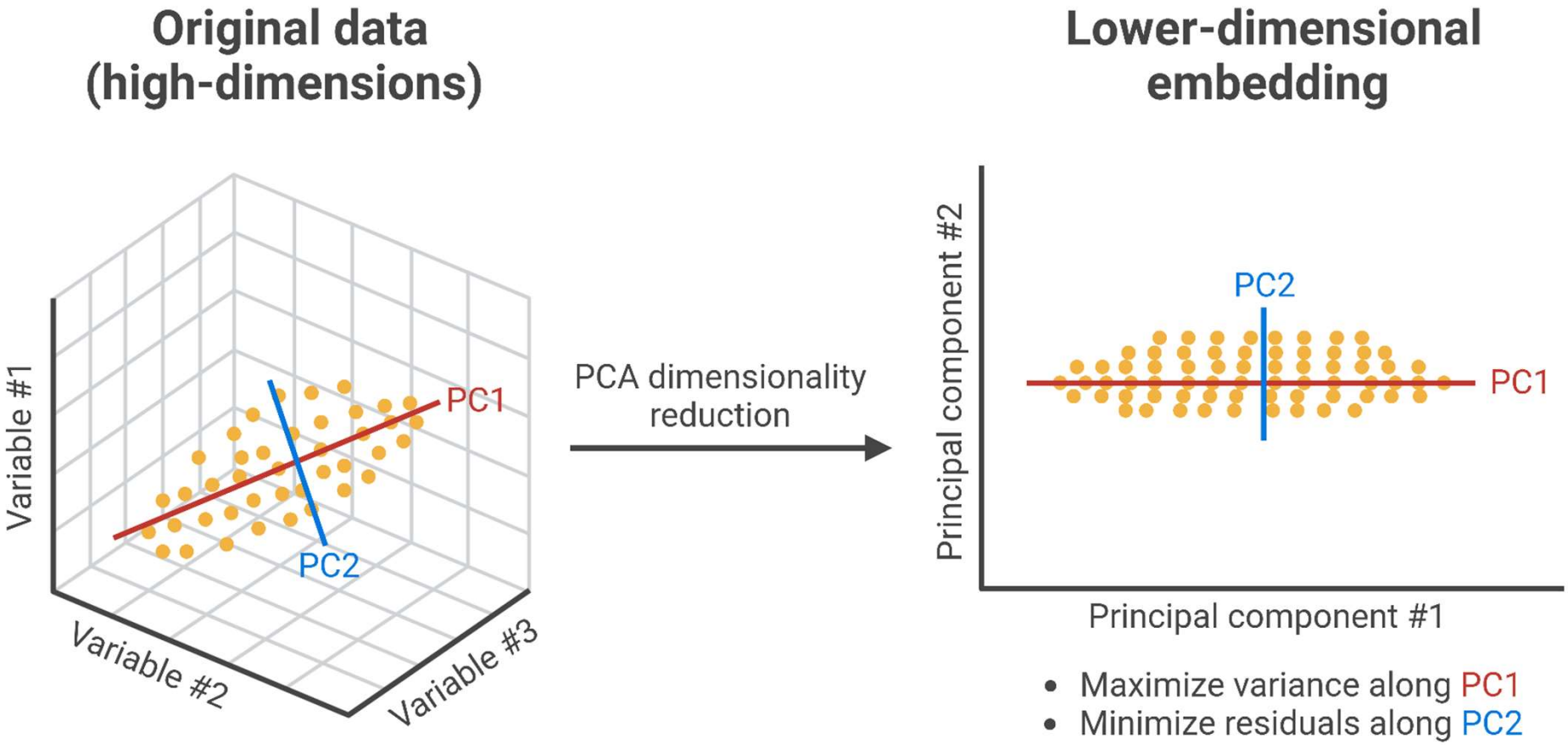


Correlation Matrix



computrain

Principle Component Analysis

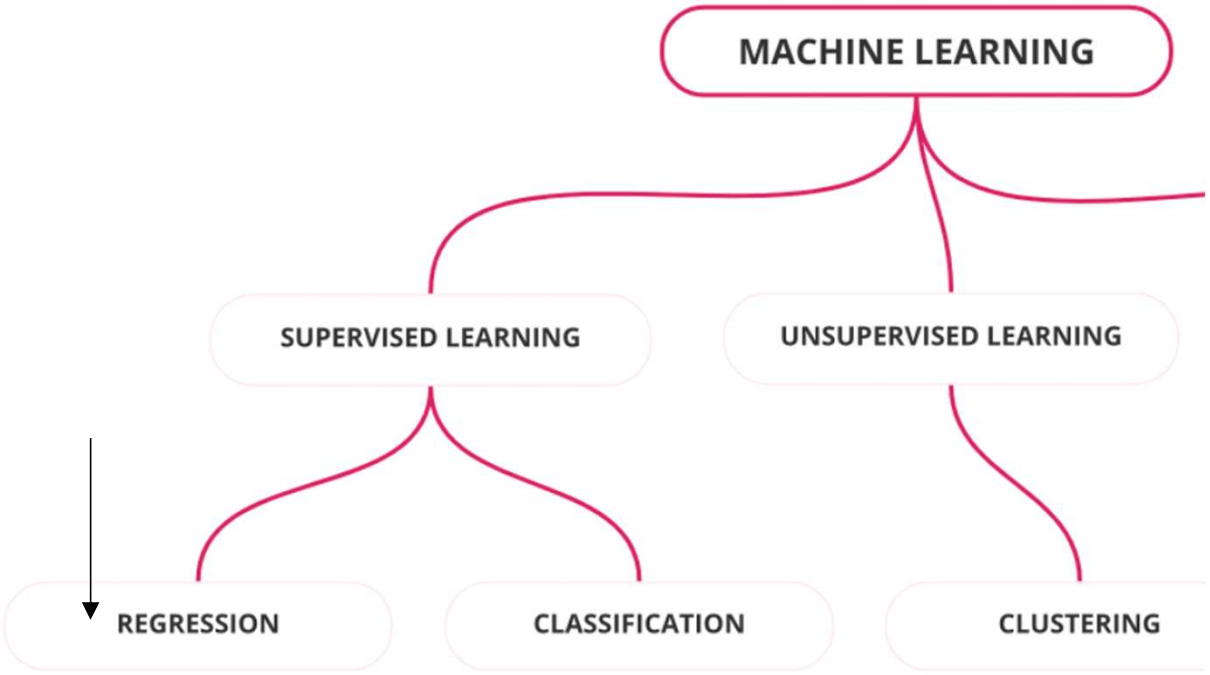


Source: BioRender.com

Regression algorithms

- Linear regression
- Multiple regression
- Non-linear regression (exponential, polynomial)
- Custom model optimalisation - Ordinary Least Squares

Regression
Linear Regression
Support Vector Regression
Gaussian Proces
Rregression
K-Nearest Neighbor
Decision Trees
Neural Networks
Ensemble Methodes

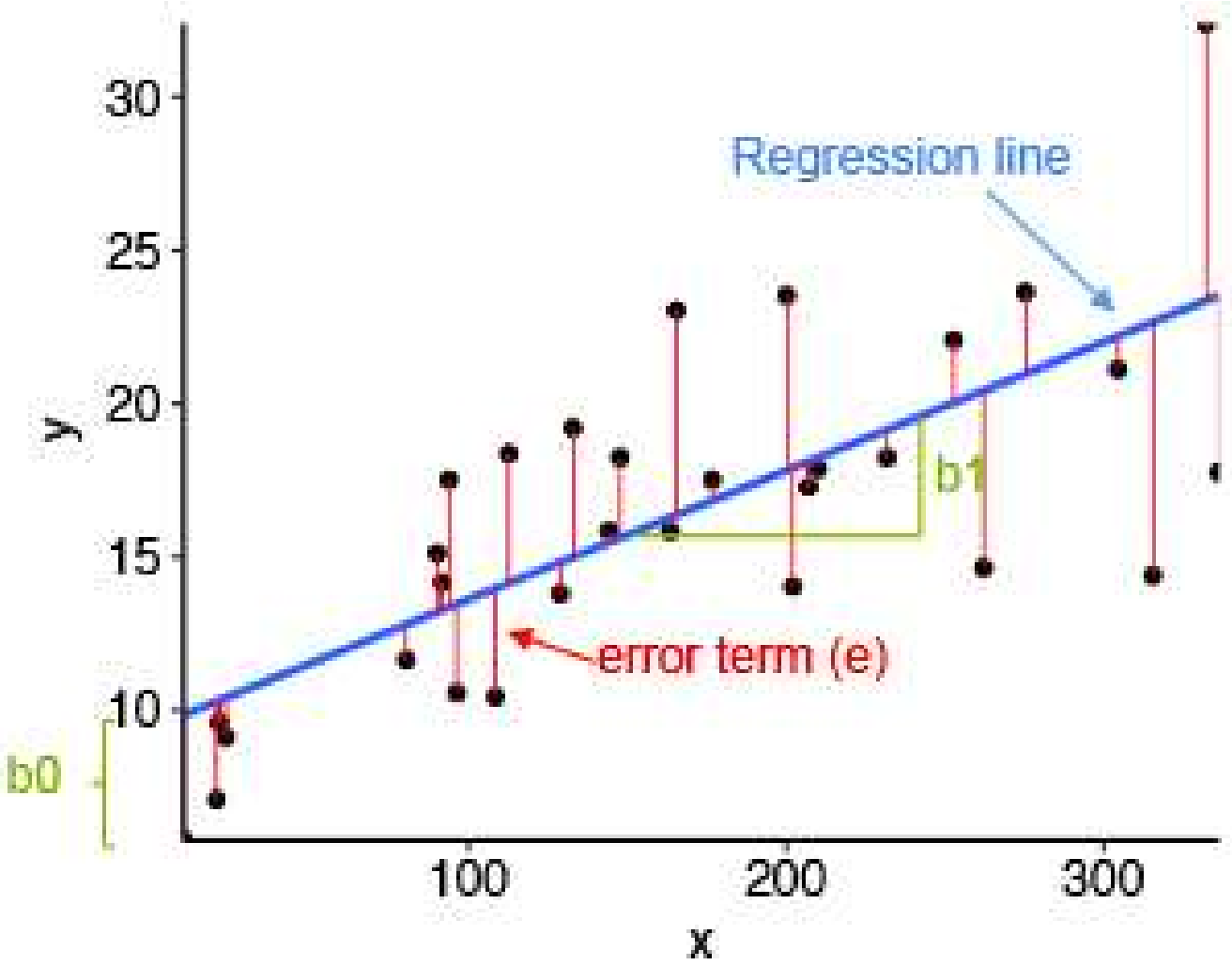


Closed Form Solution

- Normal equation $\theta = \left(X^T X\right)^{-1} X^T \vec{y}$
- Prediction $\hat{y} = \theta^T . x$
- Loss function $\sum_{i=1}^{m} (y_i - \theta^T x_i)^2$

computrain

Least Squares Linear Regression



Evaluation Measures for Regression

- Mean Squared Error (MSE) $\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(yi - \widehat{yi})^2$
- Mean Absolute Error (MAE) $\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|yi - \widehat{yi}\right|$
- Coefficient of Determination

  R-squared (R^2) $R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (yi - \widehat{yi})^2}{\sum_i (yi - \bar{y})^2}$

computrain

Linear Regression with Python

- Numpy — numpy.polyfit & numpy.poly1d
  numpy.linalg.lstsq
- Scipy — scipy.stats.linregress
- Scikit-learn — from sklearn.linear_model import LinearRegression

Optimalization

- scipy.optimize.least_squares
- scipy.optimize.curve_fit

Cost function

- OLS – Ordinary Least Squares

Cross-Validation

- k-Fold

Let’s compare linear regression models

- Implement a linear model as a function
- Select a range of x values
- Determine the corresponding y values
- Add noise to the y values
- Use numpy.polyfit to detrmine the linear regression coefficients
- Do the same using scipy.curve_fit
- Compare the results

Demo comparison Numpy & Scipy

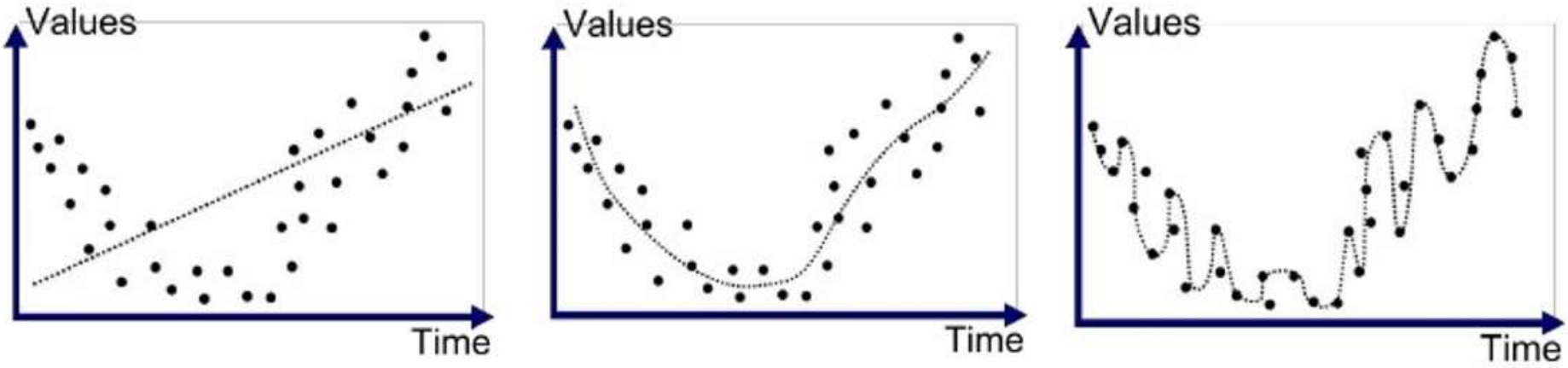
```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit


# Define linear model function
def linear_model(x, m, c):
    return m * x + c


# Select range of x values
x_values = np.linspace(0, 10, 100)

# Determine corresponding y values using linear model
m_true = 2
c_true = 5
y_values_true = linear_model(x_values, m_true, c_true)

# Add noise to y values
noise = np.random.normal(0, 1, len(x_values))
y_values_noisy = y_values_true + noise

# Use numpy.polyfit to determine linear regression coefficients
coefficients_np = np.polyfit(x_values, y_values_noisy, 1)

# Use scipy.curve_fit to determine linear regression coefficients
popt, _ = curve_fit(linear_model, x_values, y_values_noisy)
```

```
Numpy polyfit coefficients: [1.92392055 5.36082167]
Scipy curve_fit coefficients: [1.92392055 5.36082167]
```


computrain

What if it doesn't fit?

- Machine learning is a trade off between bias and variance
- Overfitting -> model does not generalize well
- Bias -> model is too generalized



Underfitted | Good Fit/Robust | Overfitted

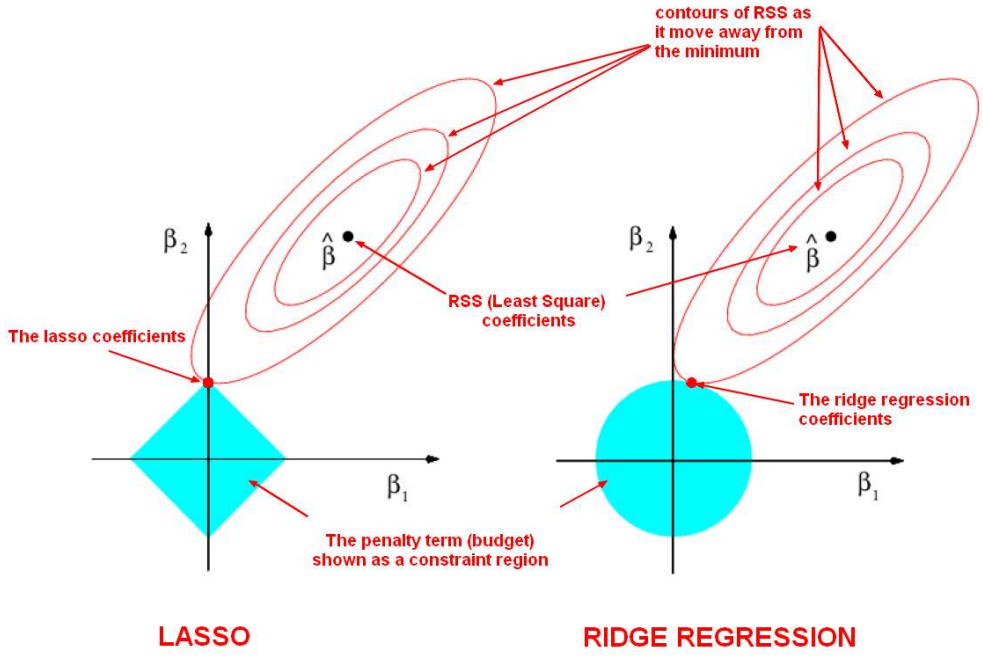
Source: Anup Bhande, Medium

Lasso

- **L**east **A**bsolute **S**hrinkage and **S**election **O**perator
- Regression analysis method that performs **variable selection** and **regularization**
- Generally improves prediction accuracy and interpretability of regression models
- Selects a reduced set of the known covariates for use in a model
- Parameter λ: controls shrinkage. Larger values increase shrinkage
- Penalty called a shrinkage penalty is applied to encourage shrinking parameters to 0 (effectively eliminating them)

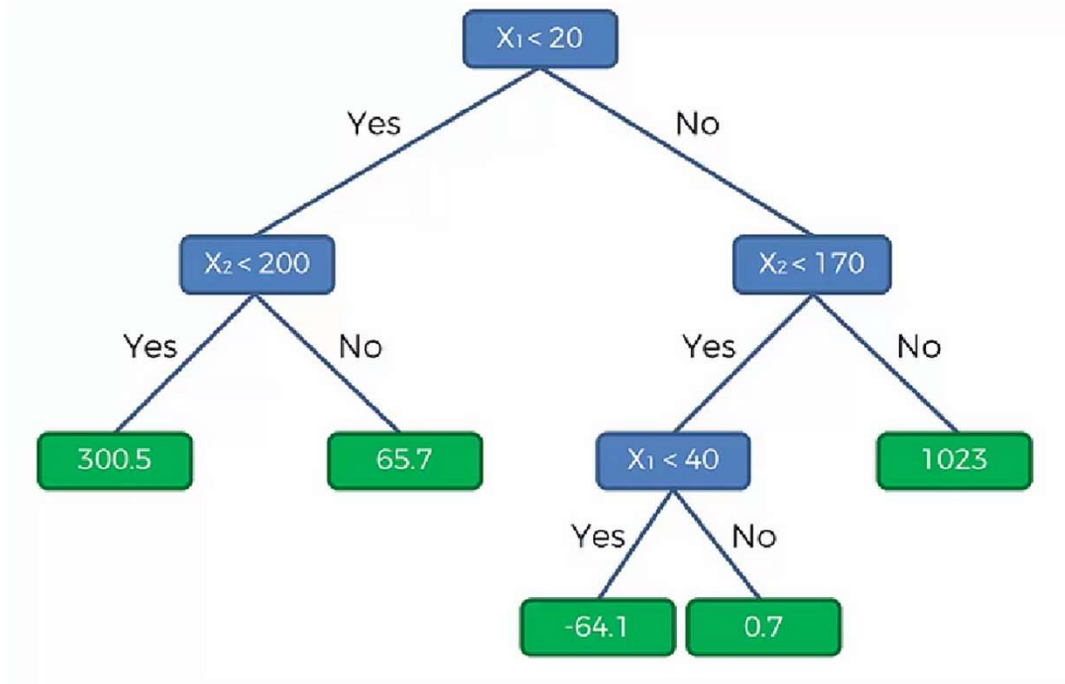
Regularization

- Ridge Regression
- Least Absolute Shrinkage and Selection Operator (LASSO)
- Elastic Net
- Least-Angle Regression (LARS)



Other Regression Algorithms

- Nearest Neighbor Regression
- Decision Tree
- Random Forest
- Robust Regression — RANSAC
- Gaussian process regression
- Support Vector Regression



Many algorithms can be used for both regression and classification

Exercise: multiple linear regression

Use multiple linear regression to estimate the house price from the other features in the dataset.

- Get the data from housing.csv.
- Explore the data.
- Clean the data. Remove missing values.
- Prepare the data for modelling. Encode categorical variables and scale numerical variables. Determine target variable. Split in training and test sets.
- Use scikit-learn to train a linear model with sklearn.linear_model.LinearRegression.
- Evaluate the model by calculating various metrics (R squared, MSE, MAE).
- Display the coefficients. What does this tell you?
- Repeat the modeling with other models like Ridge, Lasso, ElasticNet.
- Also build a model with RandomForestRegressor and display the feature importances.

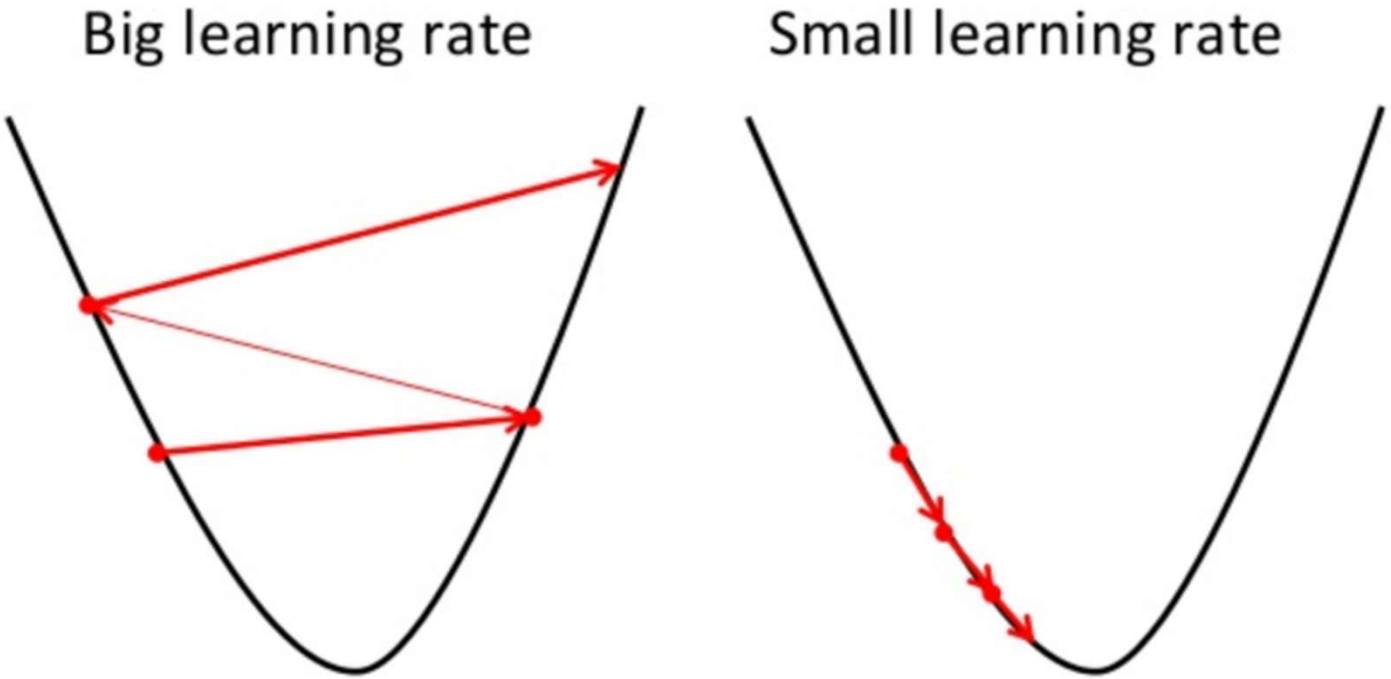
Part 3

Optimalising your solutions

Part 3 - Optimalization

- Cost Function
- Least Squares Optimalisation
- Gradient Descent
- Monte Carlo

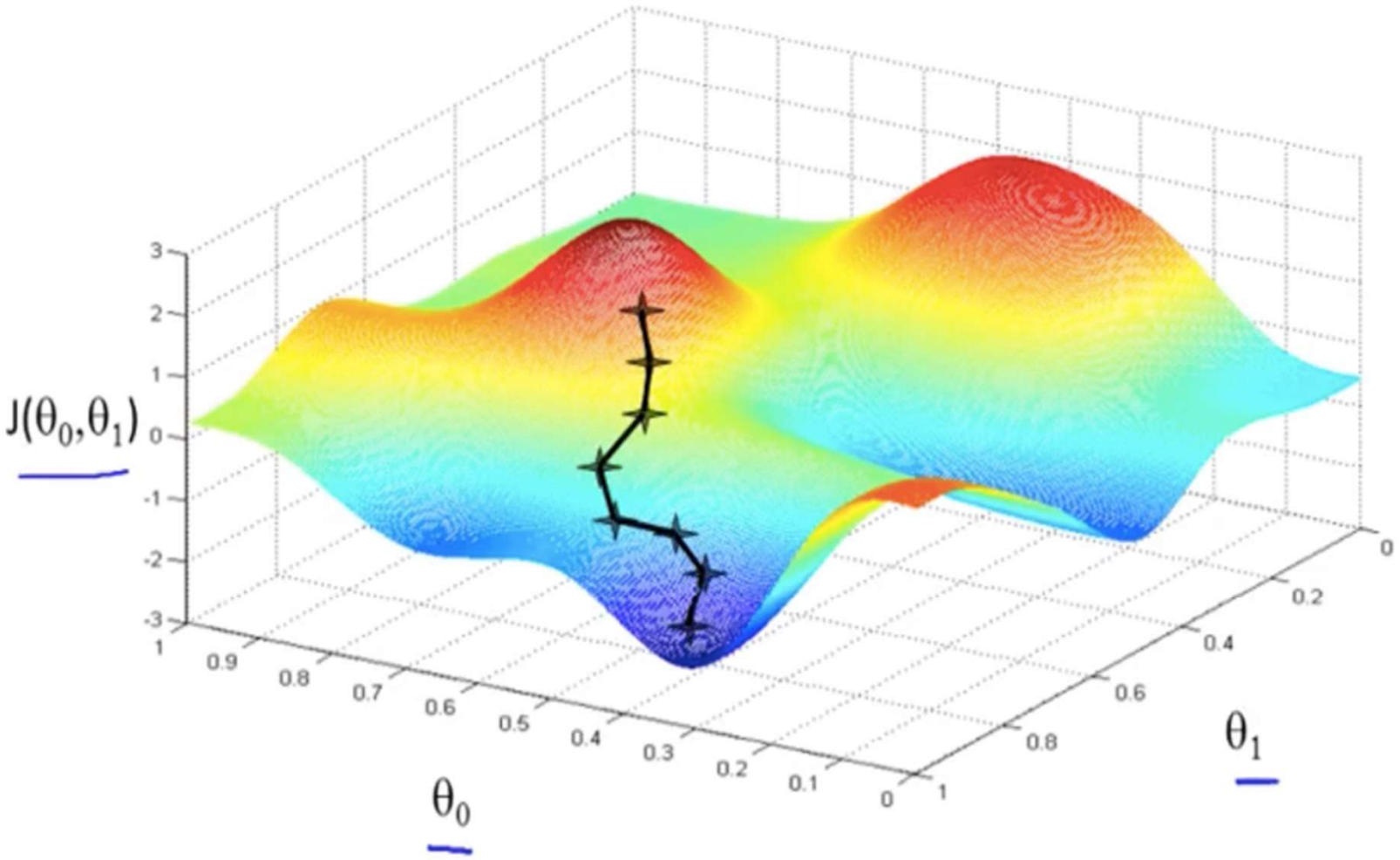
Learning rate



We need to carefully consider the hyperparameters of our machine learning models, or they will not work as intended.

computrain

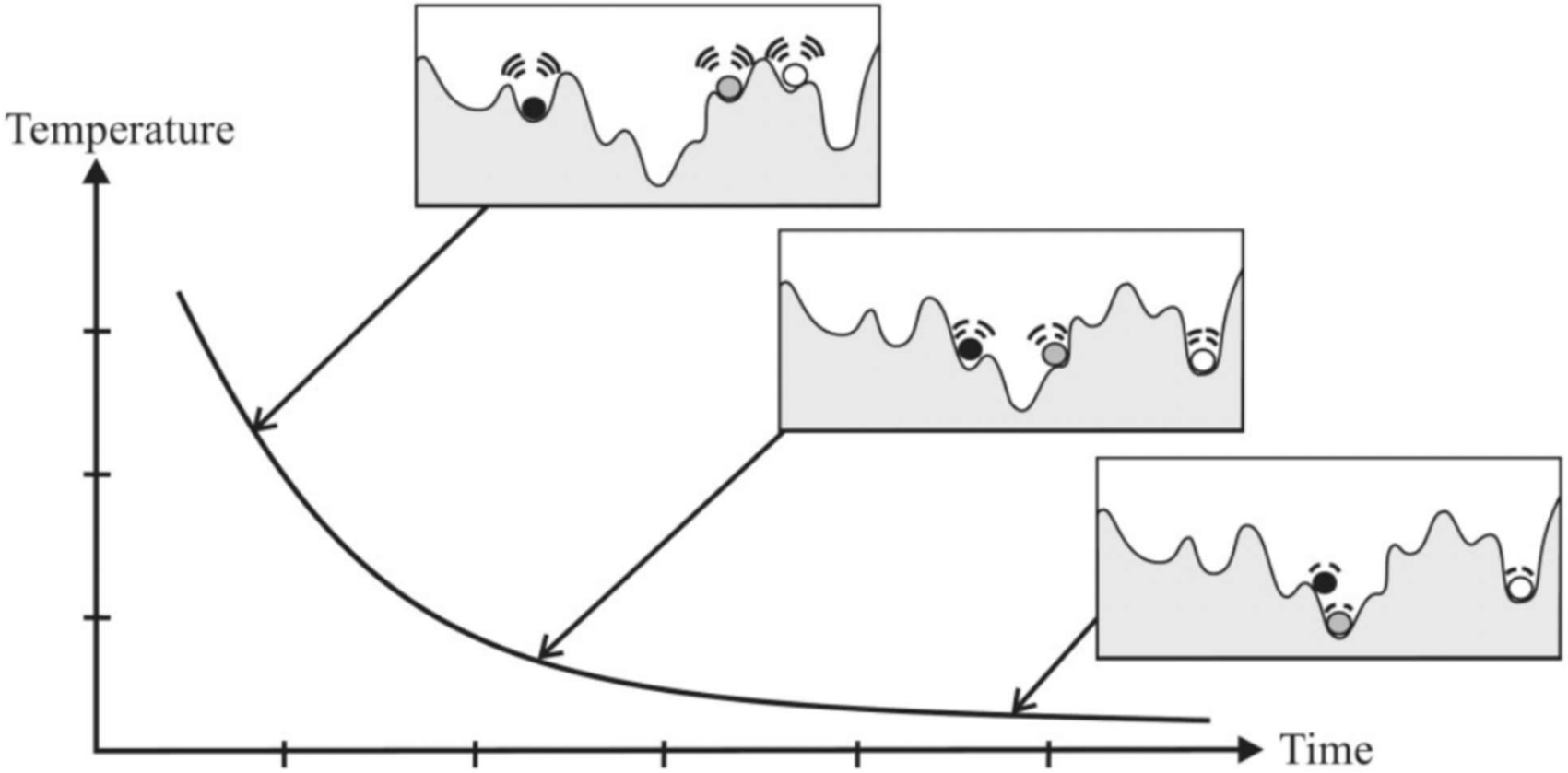
Optimalization – Gradient Descent



Randomized Optimalization

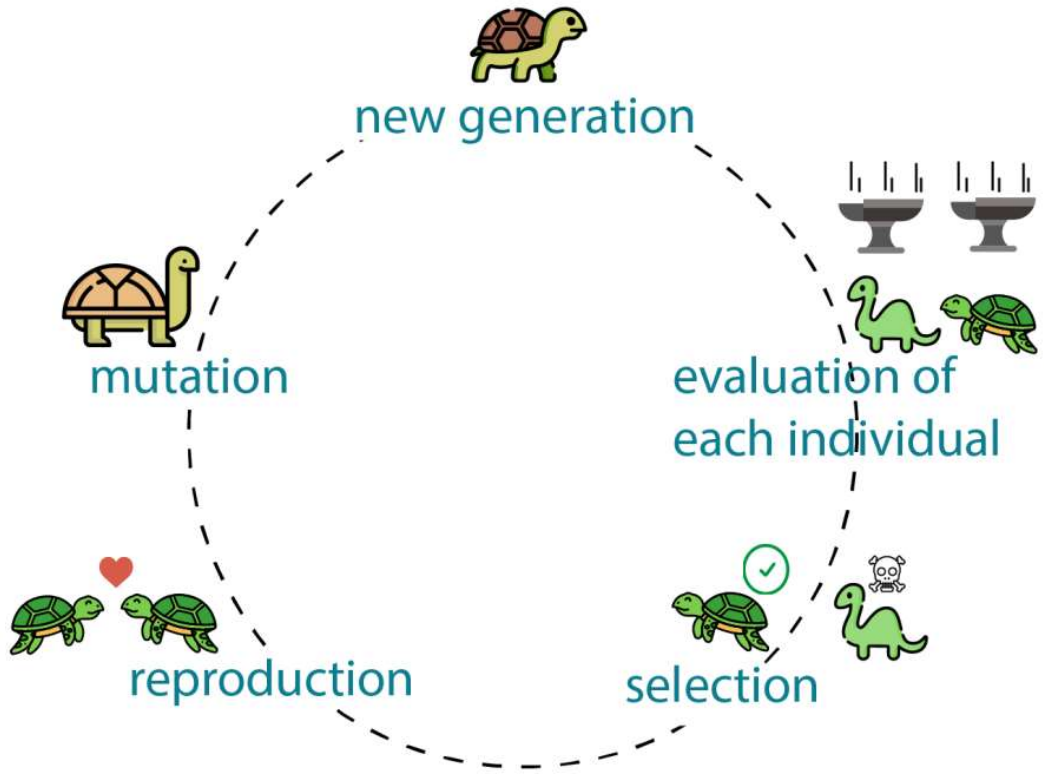
- (random-restart) Hill Climbing
- Randomized Hill Climbing (also known as stochastic hill climbing)
- Simulated Annealing
- Genetic Algorithm
- MIMIC (Mutual-I)
- Information-Maximizing Input Clustering

Simulated Annealing



Genetic Algorithms

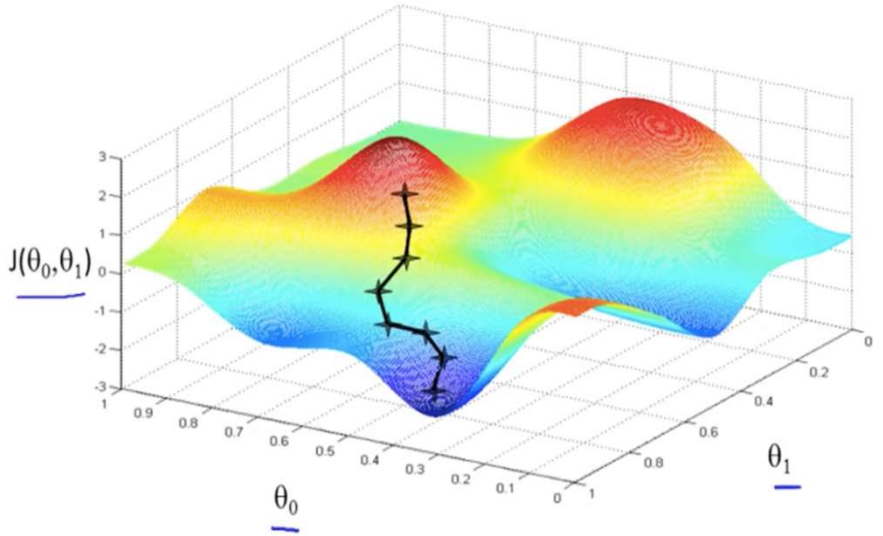
- Algorithm that works with different models to optimise
- Often uses a string representation that corresponds to a vector, tree, graph or other data structure
- New algorithms are produced by combining traits from parents and mutation



Source: Andriy Strepetilov, LinkedIn

Opdracht 3.1 – Gradient Descent

- Create points along a 2-degree polynomial function
- Determine the gradient function.
- What is the minimum?
- Build a program that starts at a random point and iterates to the maximum.
- Use the gradient function.



Opdracht 3.2 – Hill Climbing

- In the exercise a random hill landscape is created
- Define a function that can find the maximum by iterating steps
- A second landscape is defined
- Change the function to step over the same hills.
- Or use a random jump the start somewhere else.

Opdracht 3.4 - Getij

- 1. Download a dataset with tidal information for one year from: **https://www.rijkswaterstaat.nl/water/waterdata-en-waterberichtgeving/waterdata/getij**
- 2. Import it into a pandas dataframe.
- 3. Perform Exploratory Data Analysis (EDA).
- 4. Select columns with measurement moments and measurement values.
- 5. Convert measurement moments to datetime type.
- 6. Answer the following questions:
- - What is the highest value of the tide level during ebb?
- - What is the lowest value of the tide level during flood?
- - What cycles are visible in the data? How long do they last?
- 7. Create a model based on a sinusoidal function:
- - Use `from scipy.optimize import curve_fit` to find the optimal parameters.
- - What is the duration of a cycle?

```
def model(x, amplitude, offset, cyclus, shift):
        return amplitude * np.sin(2 * np.pi * ((x - shift) / cyclus)) + offset
```


computrain

Part 4

Classification Algorithms

Part 4 - Classification

- Comparing Algorithms
- Ensemble Methods
- Evaluation Metrics

Classification methods

- Naive Bayes
- K-Nearest Neighbor
- Logistic Regression
- Support Vector Machines
- Discriminant Analysis
- Decision Trees
- Random Forest
- Neural Networks

Background

Variations – Metaparameters

Pro's

Con's

Application

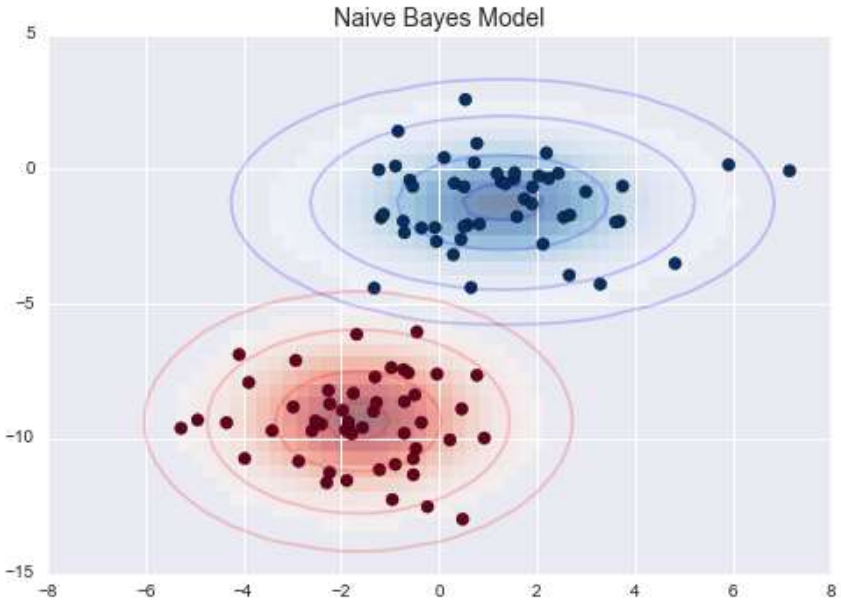
computrain

Naive Bayes

$$P(A|B) = \frac{P(B|A).P(A)}{P(B)}$$

Formula for conditional probability

- Gaussian Naive Bayes
  - Assumes the data is characterized by a Gaussian distribution
  - No covariance (**independent** dimensions) between the parameters
  - Prior probabilities are equal
  - Always check your assumptions



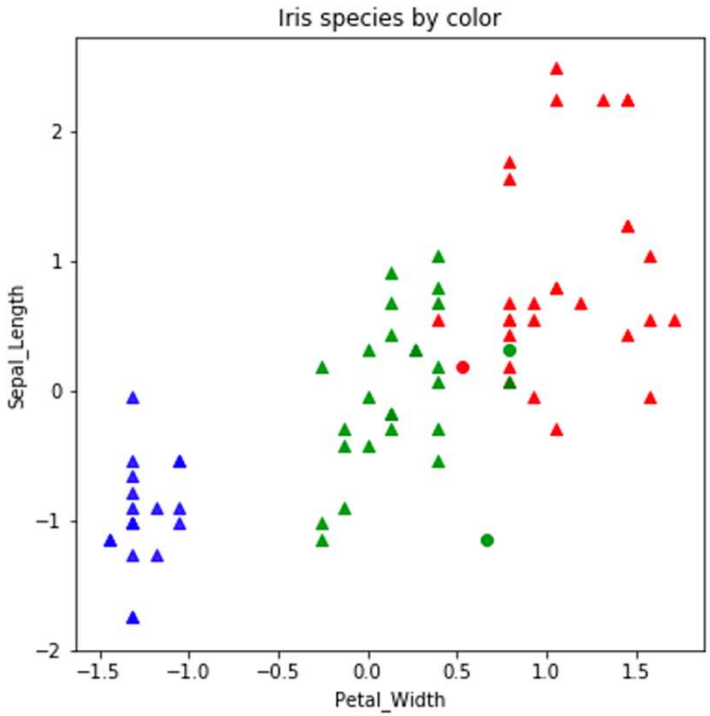
K Nearest Neighbor

Distance metric

- Euclidian
- City block

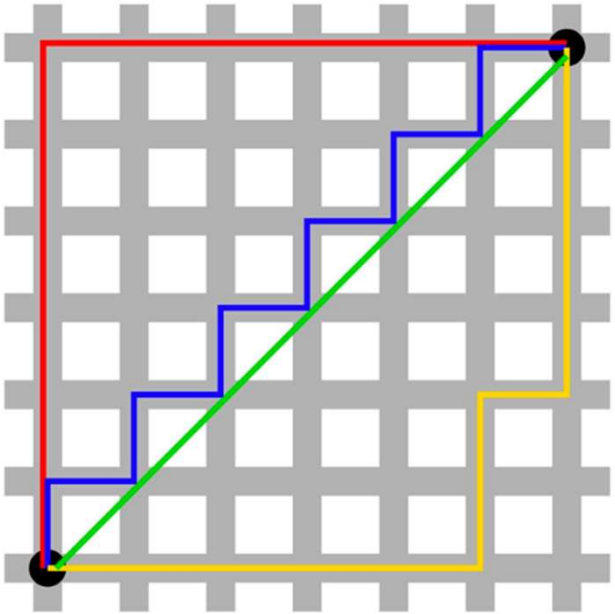
- Scaling
  - min-max
  - z-score (standardization)

- Curse of Dimensionality



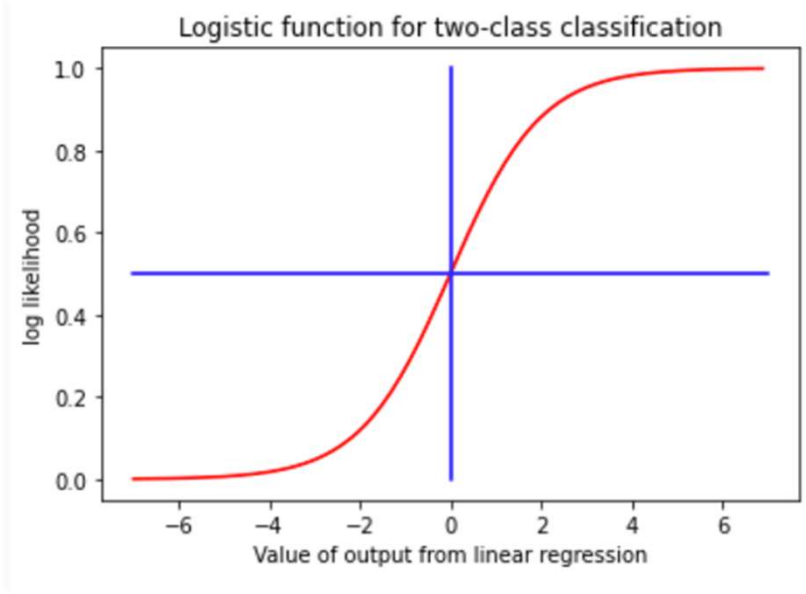
Distance metric

- Euclidean distance $d = \sqrt{(x1 - x0)^2 + (y1 - y0)^2}$
- Manhattan (or city block) distance $d = |x1 - x0| + |y1 - y0|$



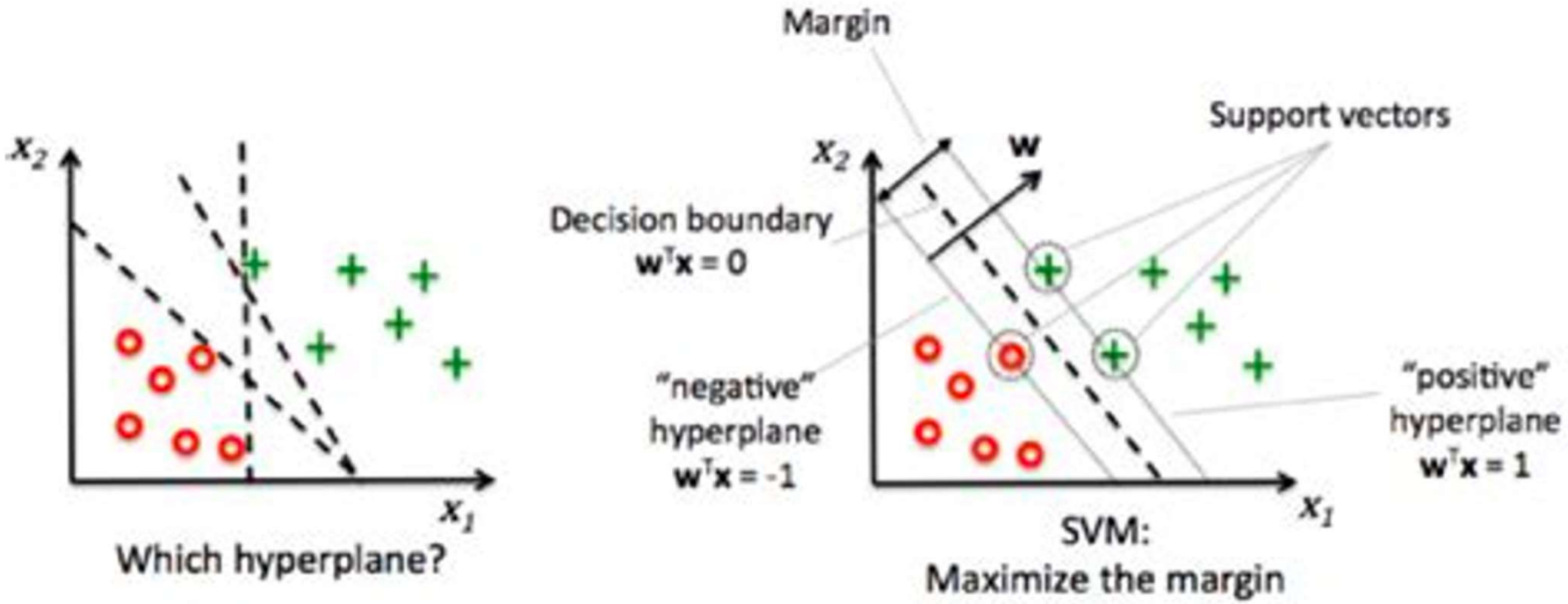
Logistic Regression

- logarithmic loss



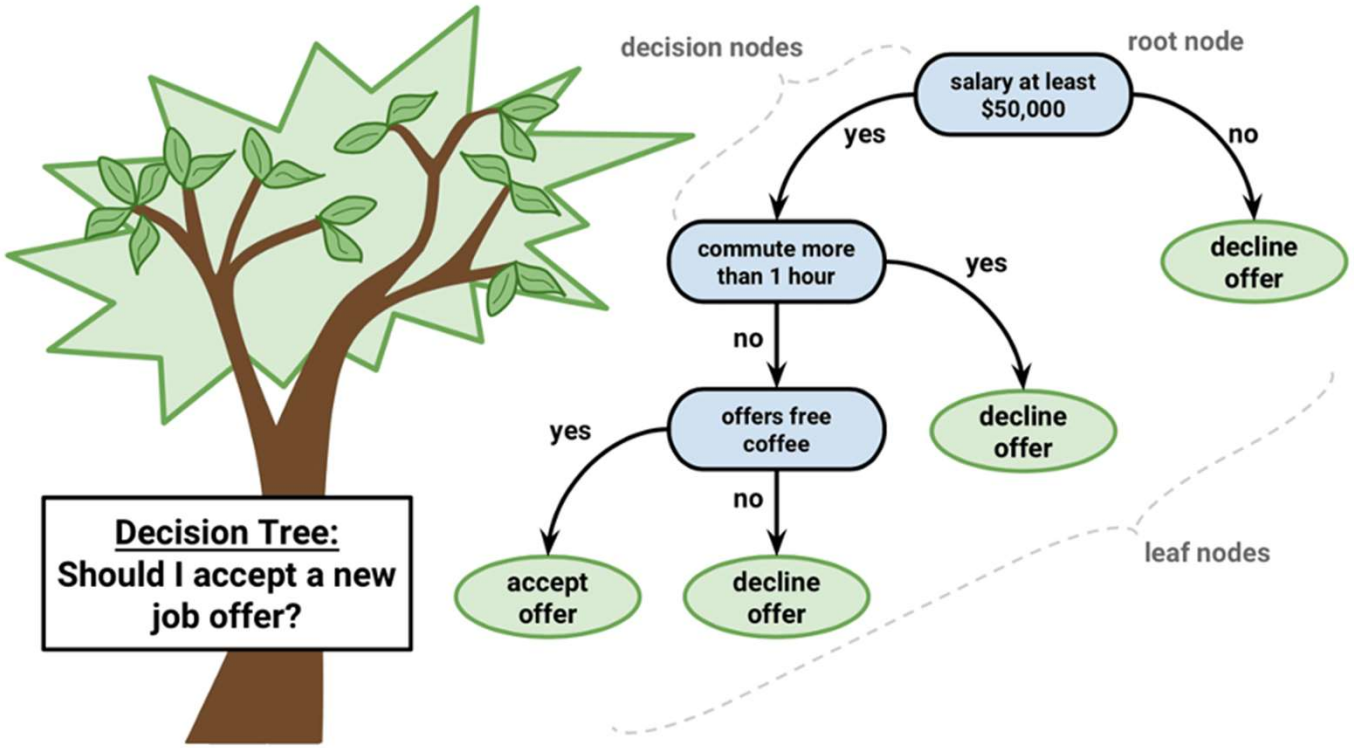
Support Vector Machine

- Kernels
  - Linear, Polynomial, Radial



Decision Trees

- Impurity
- Transparancy
- Variable importance



https://www.ibm.com/topics/decision-trees

Random Forest

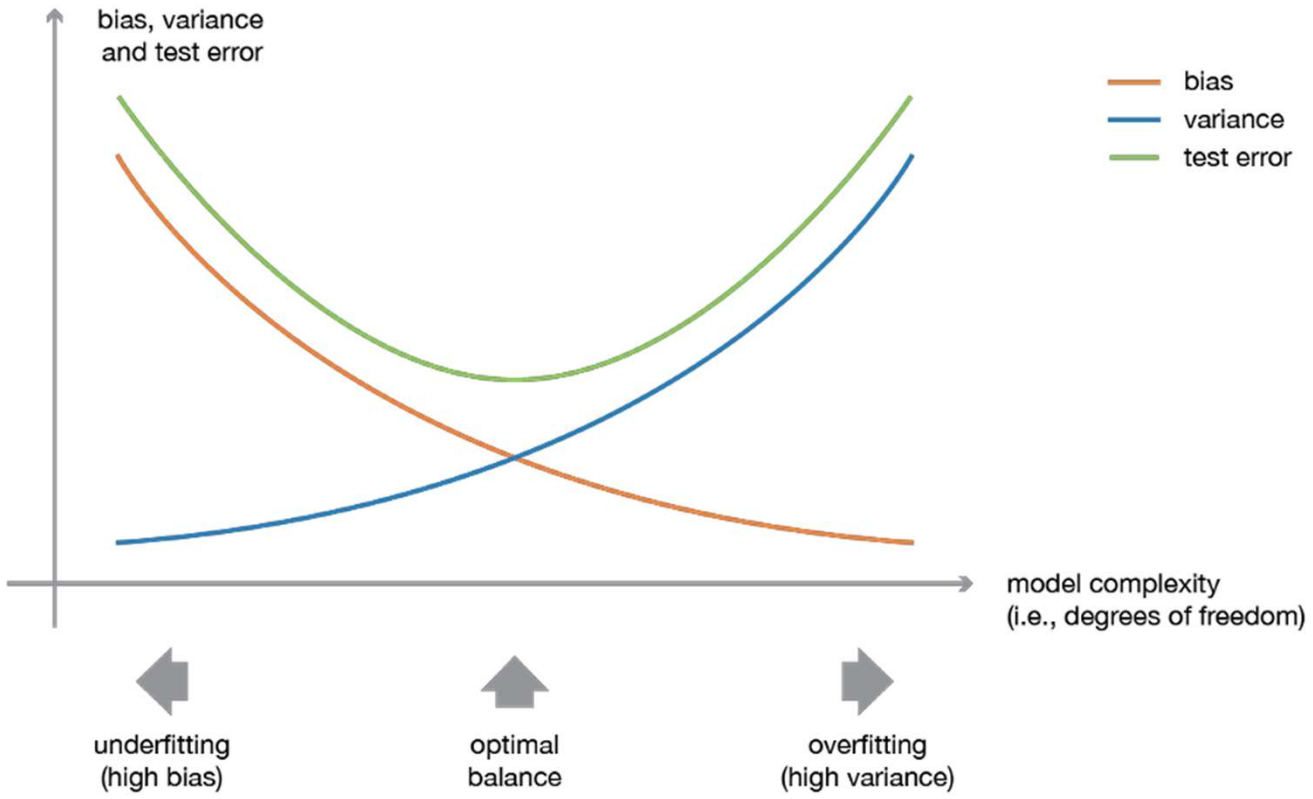
- Multiple Decision Trees
- Ensemble method - Feature Bagging
  - a random subset of the features for each tree

Ensemble Learning

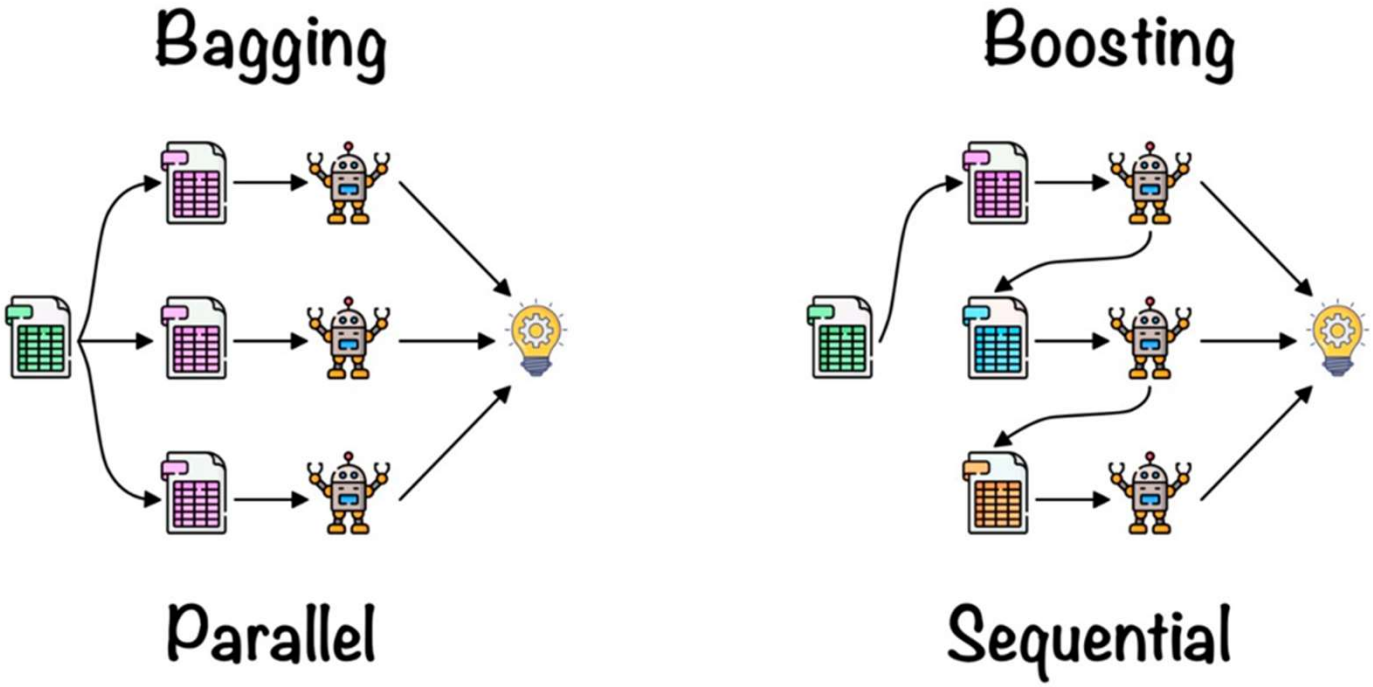
- Ensemble Learning is a technique that combines predictions from multiple models to get a prediction that would be more stable and generalize better.

Bias Variance Tradeoff

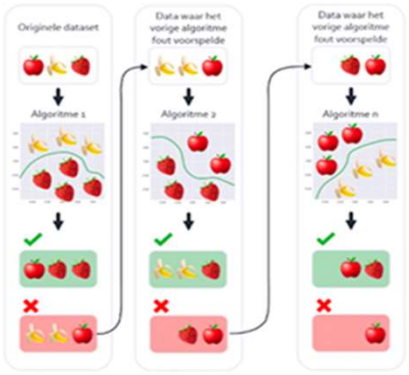
- Bias
- Variance



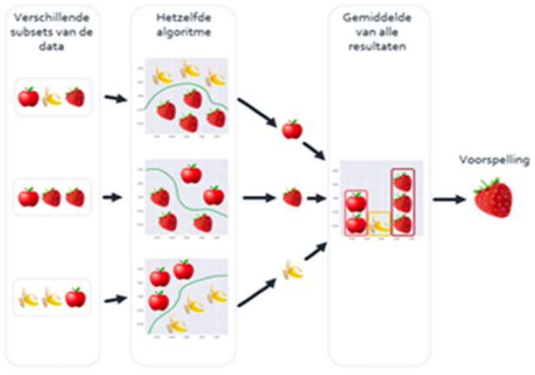
Bagging and Boosting



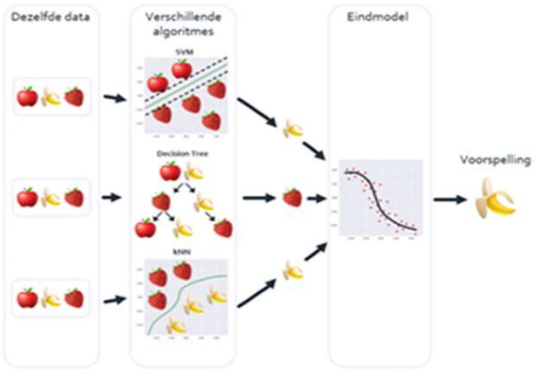
Bagging and Boosting and Stacking



Boosting



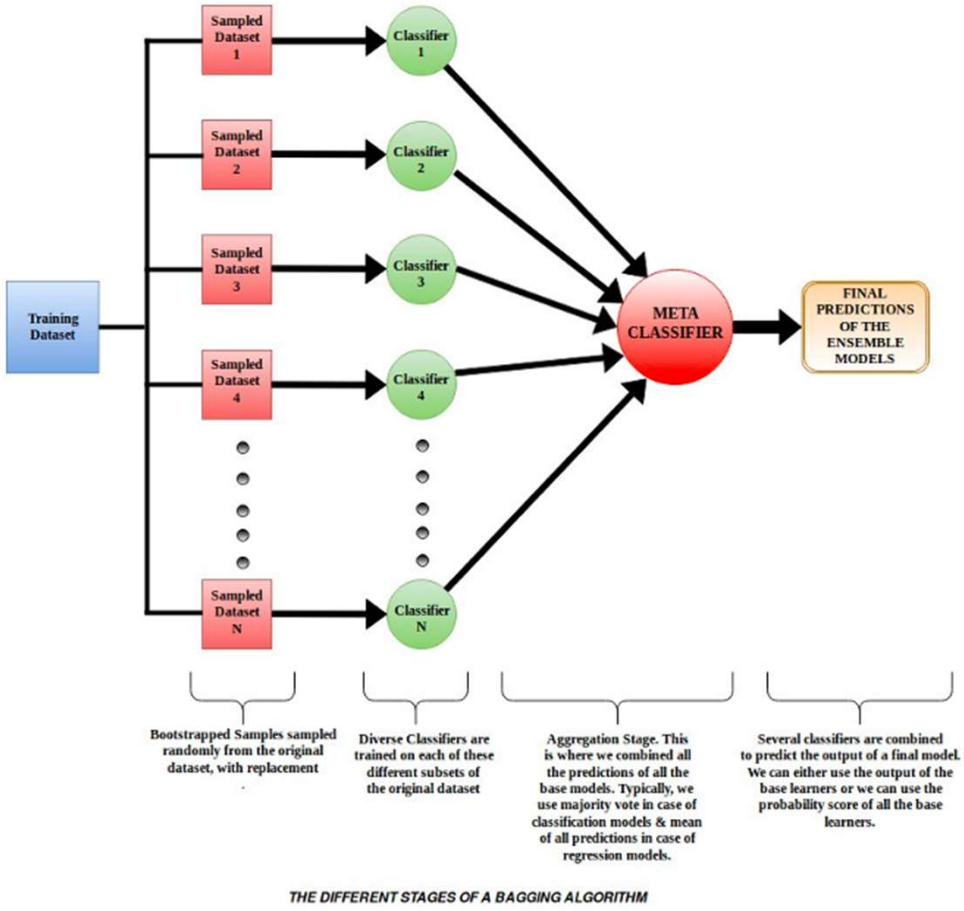
Bagging



Stacking

Bagging

- Bootstrap Aggregating
- train on random samples with replacement
- majority vote



THE DIFFERENT STAGES OF A BAGGING ALGORITHM

Boosting

- In boosting technique, the data for the training is resampled and combined in an adaptive manner such that the weights in the resampling are increased for those data points which got mis-classified more often.

AdaBoost

- The core principle of AdaBoost is to fit a sequence of weak learners (i.e., models that are only slightly better than random guessing, such as small decision trees) on repeatedly modified versions of the data.

https://vitalflux.com/adaboost-algorithm-explained-with-python-example/
https://youtu.be/-DUxtdeCiB4

Gradient Boosting

- CatBoost (Category Boosting),
- LightGBM (Light Gradient Boosted Machine)
- XGBoost (eXtreme Gradient Boosting)

Evaluation Measures for Classification

- Confusion Matrix
- Accuracy
- Precision-Recall Curve
- F1 Score
- Area Under the ROC Curve (AUC)
- Log/Cross Entropy Loss

https://www.justintodata.com/machine-learning-model-evaluation-metrics/

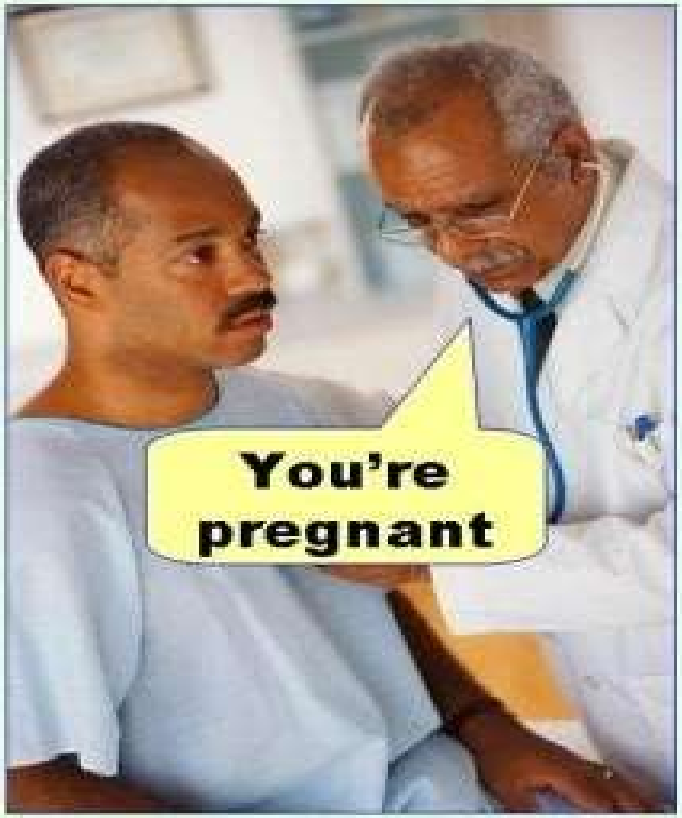
Binary Classification

- Confusion matrix

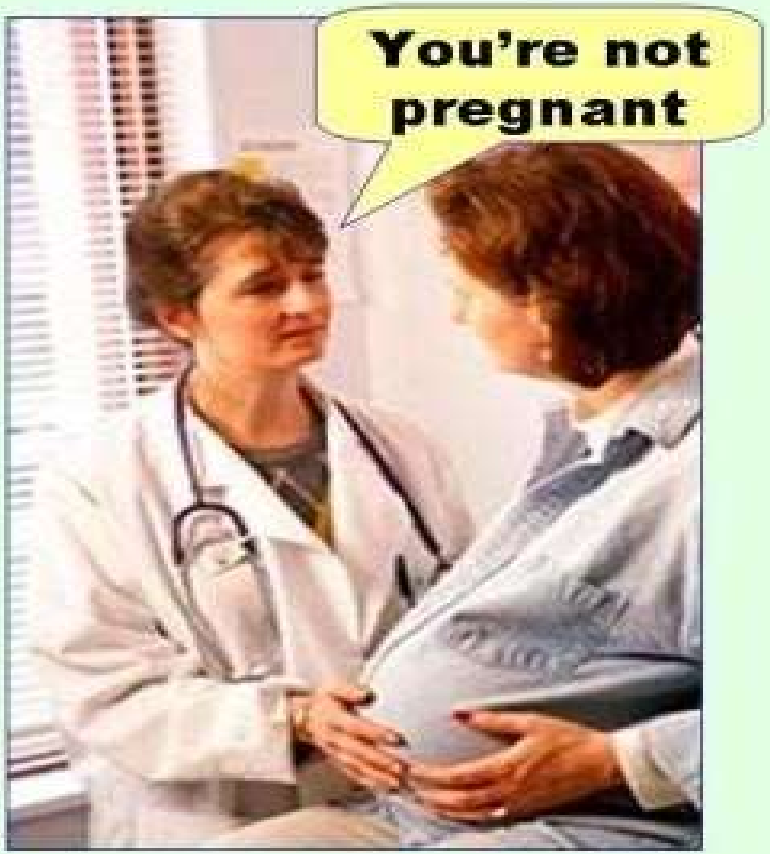
	Scored Positive	Scored Negative
Actual Positive	True Positive	False Negative
Actual Negative	False Positive	True Negative

Type I and Type II Errors

Type I error
(false positive)



Type II error
(false negative)



Model Evaluation Measures

- Confusion Matrix

		Predicted: Negative	Predicted: Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

- Accuracy $\frac{TP+TN}{TP+TN+FP+FN}$

		Predicted: Negative	Predicted: Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

- Precision $\frac{TP}{TP+FP}$

- Recall $\frac{TP}{TP+FN}$

		Predicted: Negative	Predicted: Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

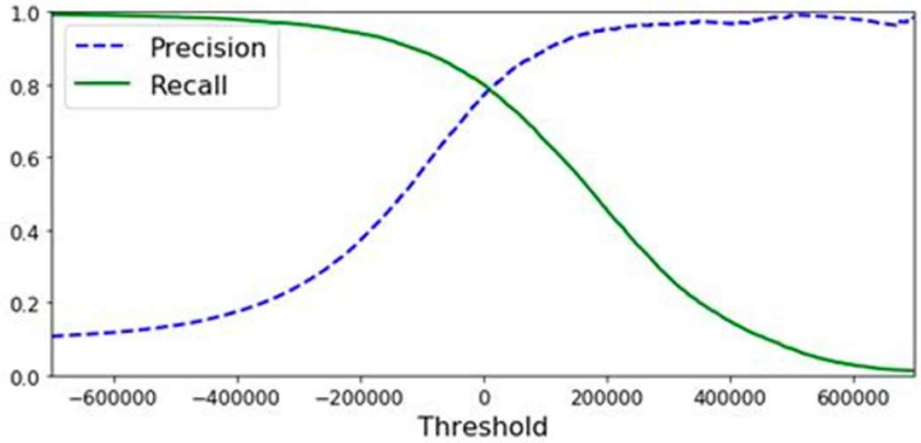
- F1 $2 \times \frac{Precision * Recall}{Precision+Recall}$

computrain

Precision vs Recall

- **Precision** is the measure of how many observations our model correctly predicted over the amount of correct and incorrect predictions.

- **Recall** is the measure of how many observations our model correctly predicted over the total amount of observations.



F1 – Precision versus Recall

- $F_1 = 2 \cdot \frac{precision \cdot reca}{precision + reca}$
- $F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot reca}{(\beta^2 \cdot precision) + recall}$

hardloopwedstrijd 10 km	<- 2 * a
5 km – 10 km/u	<- $v1$
5 km – 5 km/u	<- $v2$

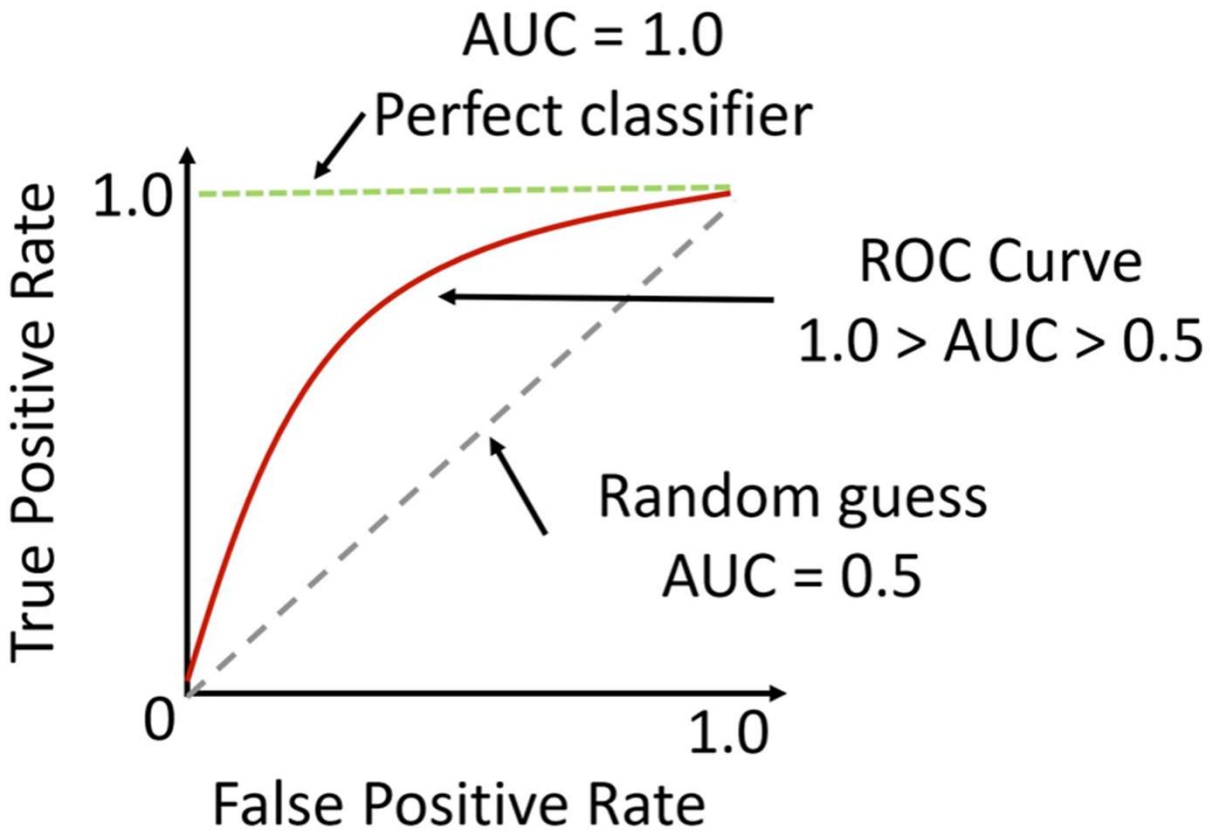
Wat is de gemiddelde snelheid?

Totale afstand	10 km
Totale tijd	0.5 uur + 1 uur
Gemiddelde snelheid	10 km / 1.5 uur
	6.7 km/u

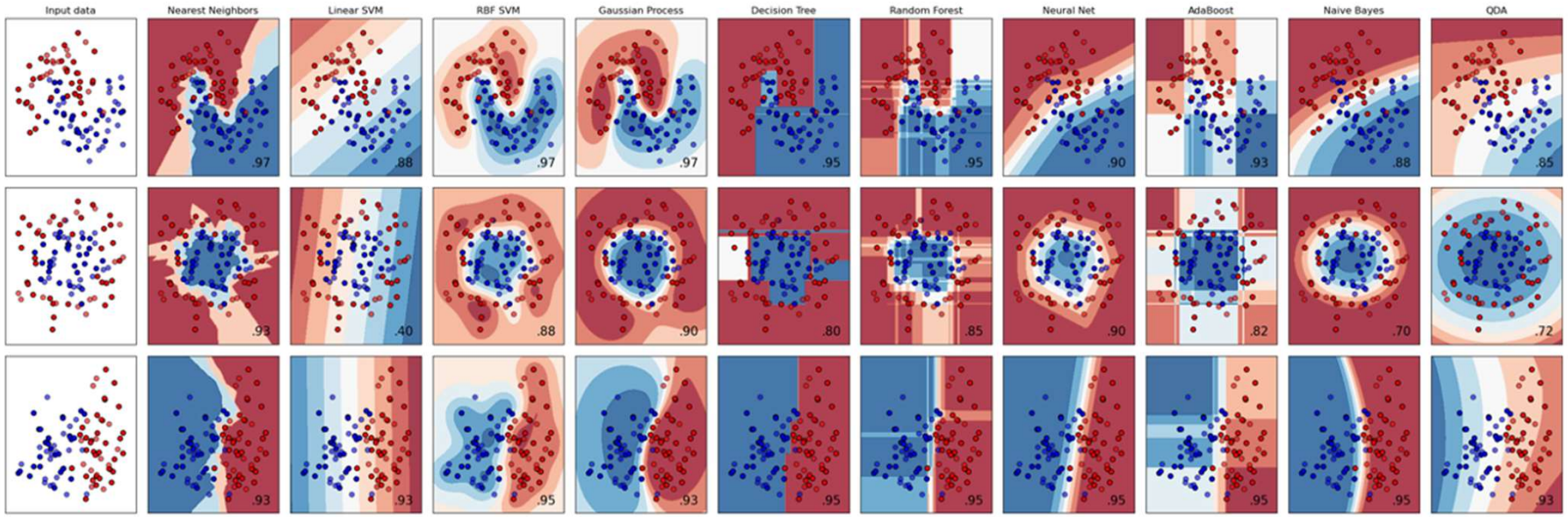
Harmonische gemiddelde $= 2 \cdot \frac{v1 \cdot v}{v1 + v2}$

computrain

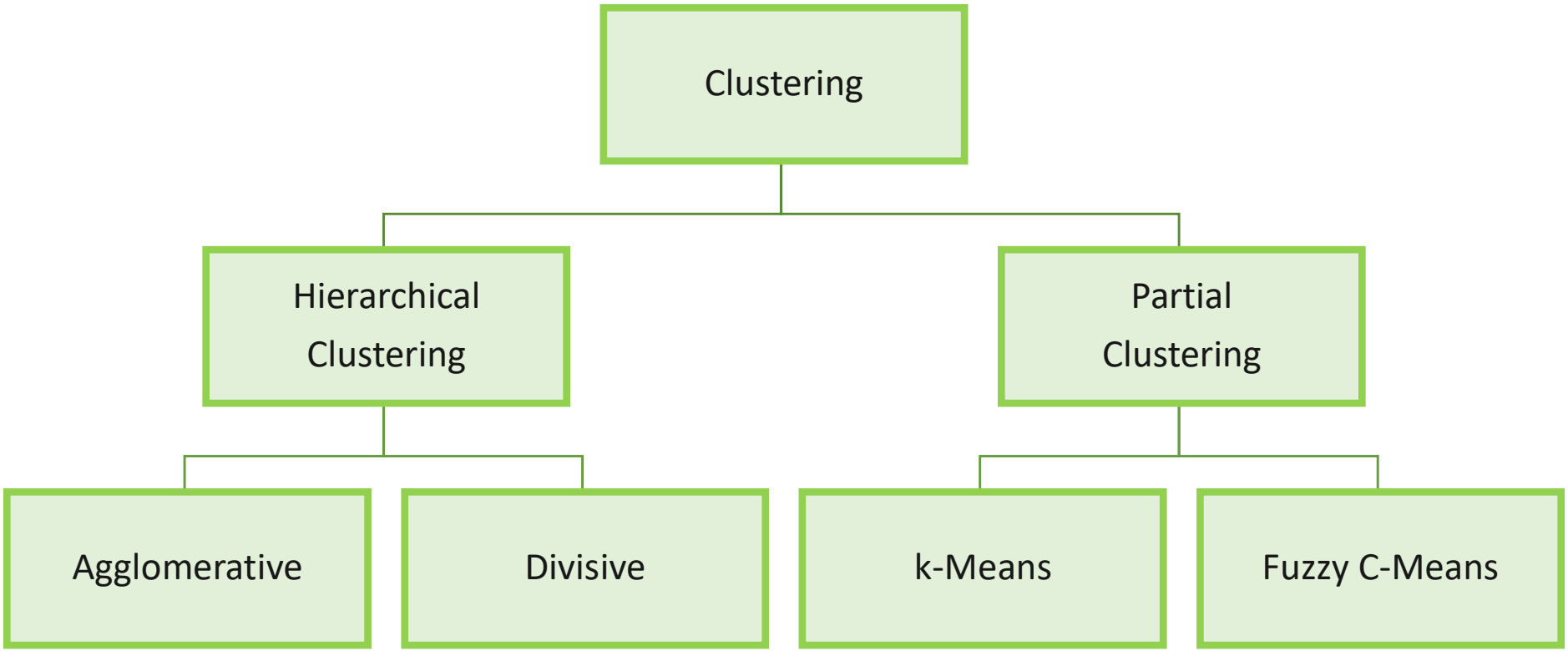
ROC en AUC



Classifier comparison



Clustering



Clustering

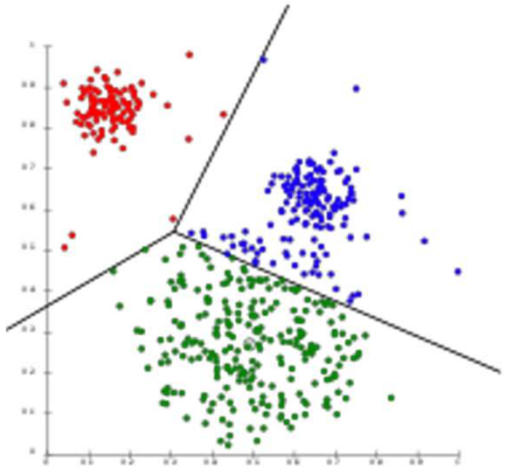
- K-Means
- DBSCAN
- Affinity Propagation
- Agglomerative Clustering
- BIRCH
- Mini-Batch K-Means
- Mean Shift
- OPTICS
- Spectral Clustering

Outlier Detection

- IQR
- Isolation Forest
- Extended Isolation Forest
- Local Outlier Factor
- DBSCAN
- One Class SVM

K Means Clustering Algorithm

- Choose number of expected centroids
- Randomly choose centroid positions
- Calculate distance between each point an the centroids
- Assign point to nearest centroid
- Calculate new centroids based on mean distance to assigned points



DBSCAN

- **D**ensity-**b**ased **s**patial **c**lustering of **a**pplications with **n**oise - DBSCAN
- Neighbors of the neighbor
- Preset neighborhood radius and minimum number of neighbors

computrain

Opdracht

- Download de German Credit dataset (https://archive.ics.uci.edu/ml/datasets/)
- Lees in een pandas dataframe
- Doe een EDA. Hoe ziet de data eruit?
- Maak vertaal dictionaries voor de codes. Gebruik hiervoor de documentatie.
- Gebruik map om de data leesbaar te maken.
- De meeste kolommen zijn categorien. Gebruik One-hot encoding.
- Maak hiervan een training en een test dataset.
- Train een classificatie model
  - k-nearest neighbor
  - naive bayes
  - logistic regression
- Kijk naar de resultaten en bepaal daarvan enkele metrics.

Opdracht

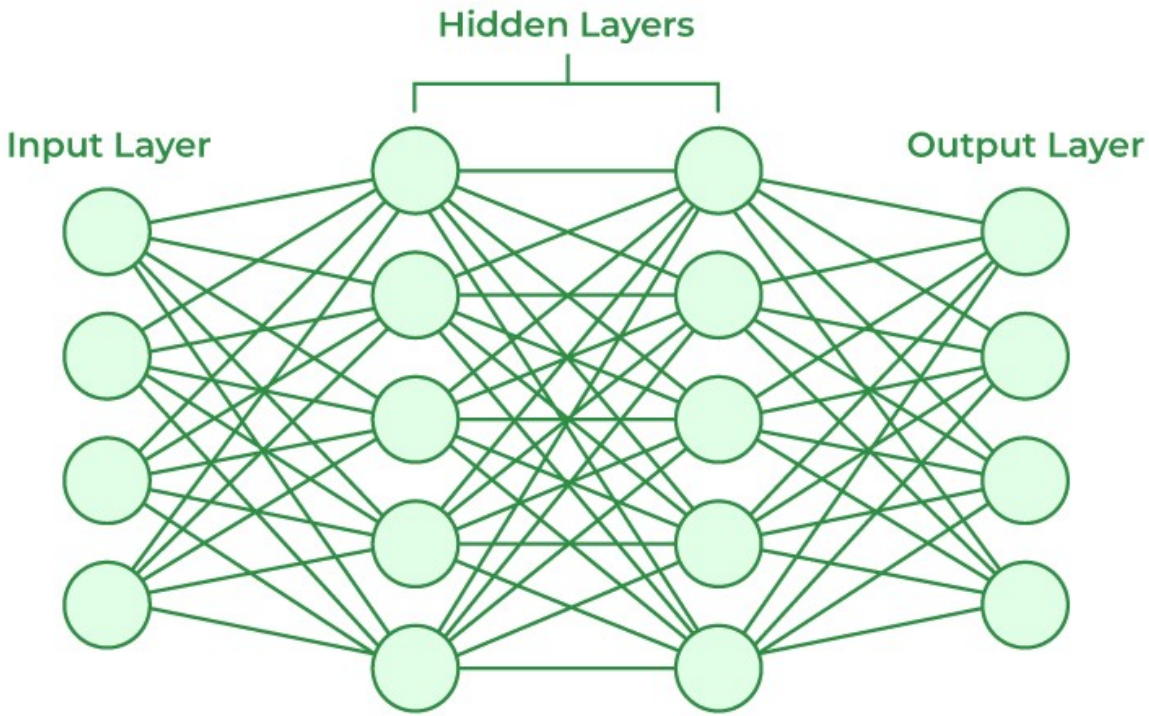
- Lees de dataset penguins van seaborn in
- Gebruik als X de features 'flipper_length_mm’ en 'body_mass_g’
- Gebruik K-Means om clusters te maken.

Part 5

Neural Networks

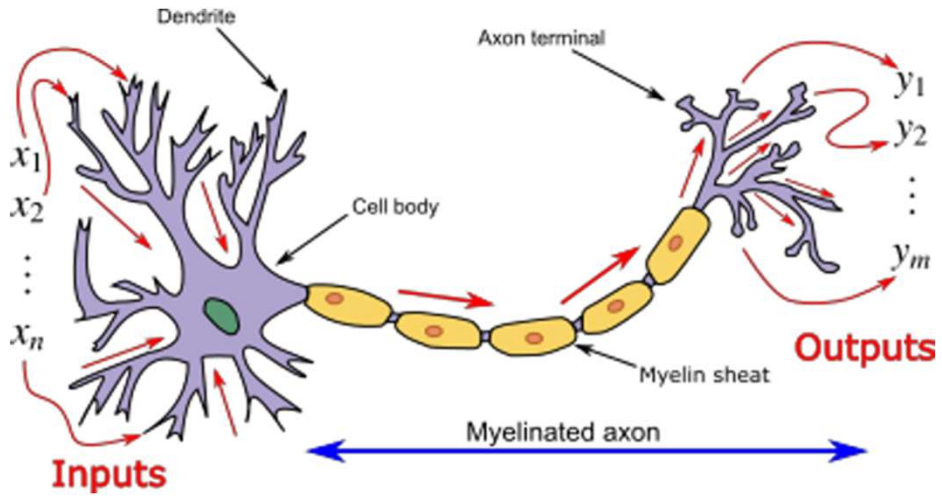
Part 5 - Neural Networks

- Multi Layer Perceptron
- Backpropagation
- Neural Network Architecture
- Applications

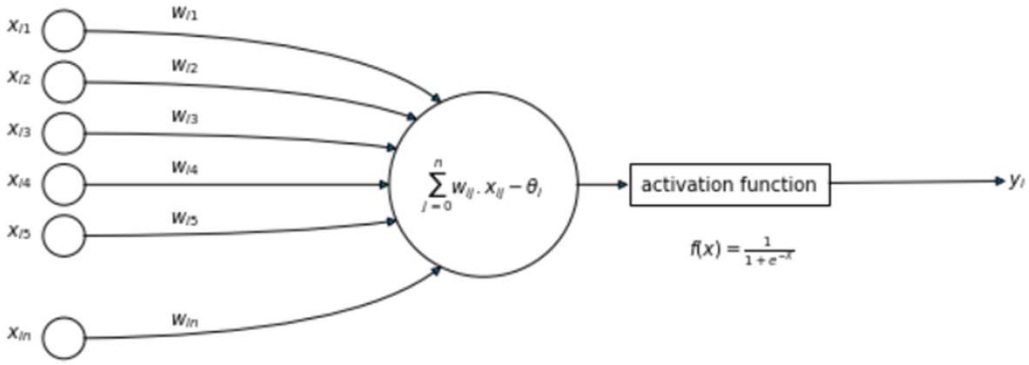


Neural Network

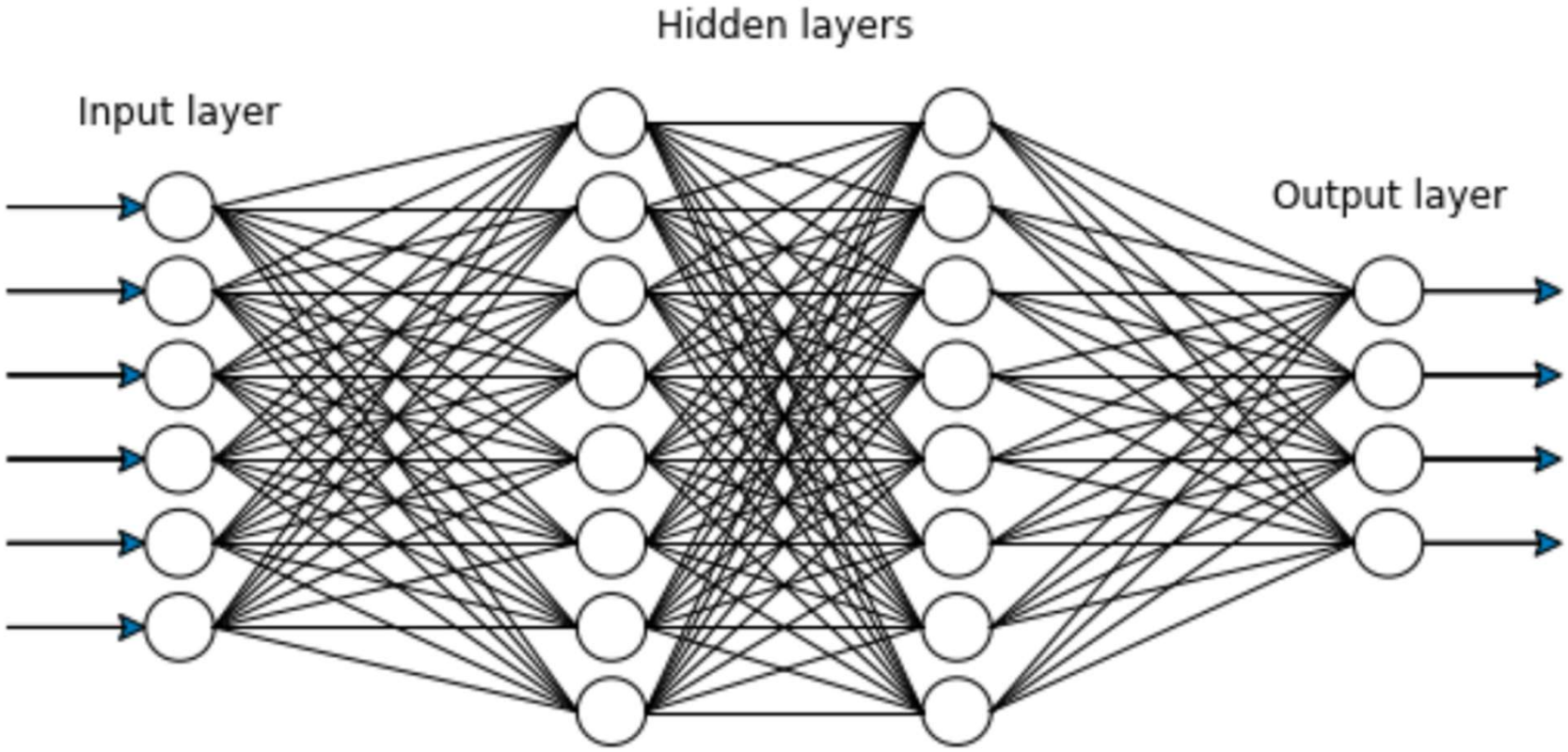
- Biological Neuron



- Artificial Neuron



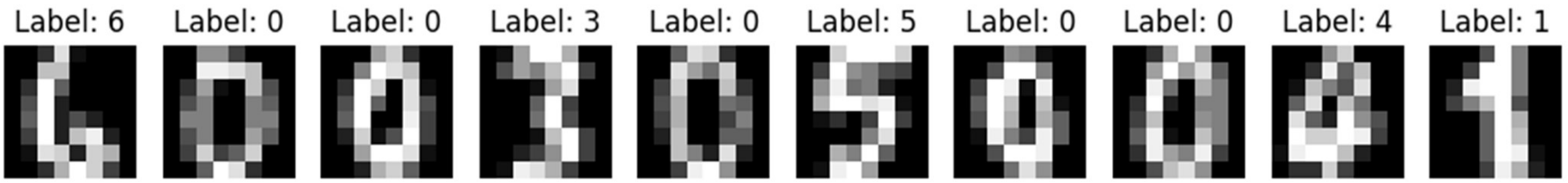
Artificial Neural Network - ANN



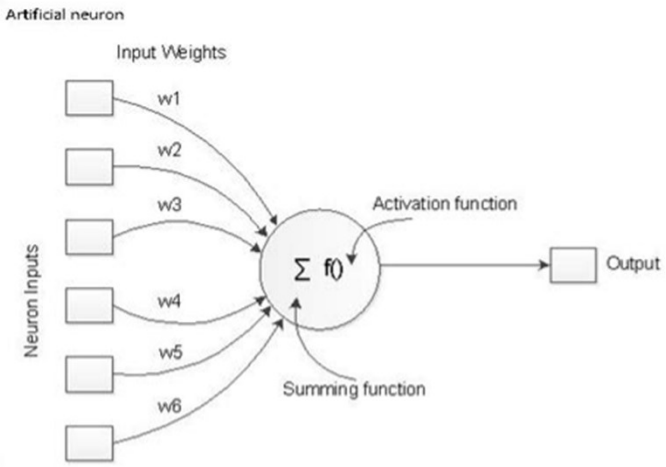
MNIST dataset: ”Hello World” for NN

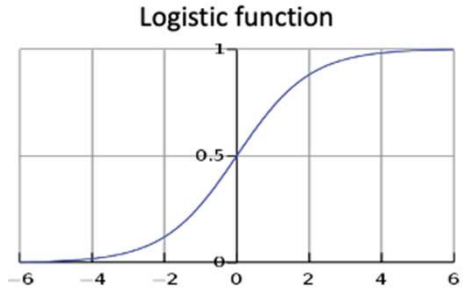
- Ray Kurzweil, 1970’s
- Developed Optical Character Recognition with Neural Networks
- Still used as an example today
- Try it!

http://www.ccom.ucsd.edu/~cdeotte/programs/MNIST.html



Artificial Neuron - Perceptron

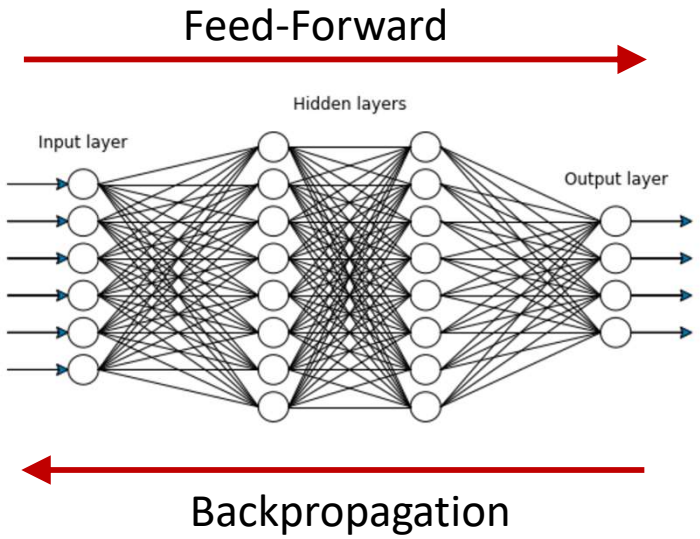


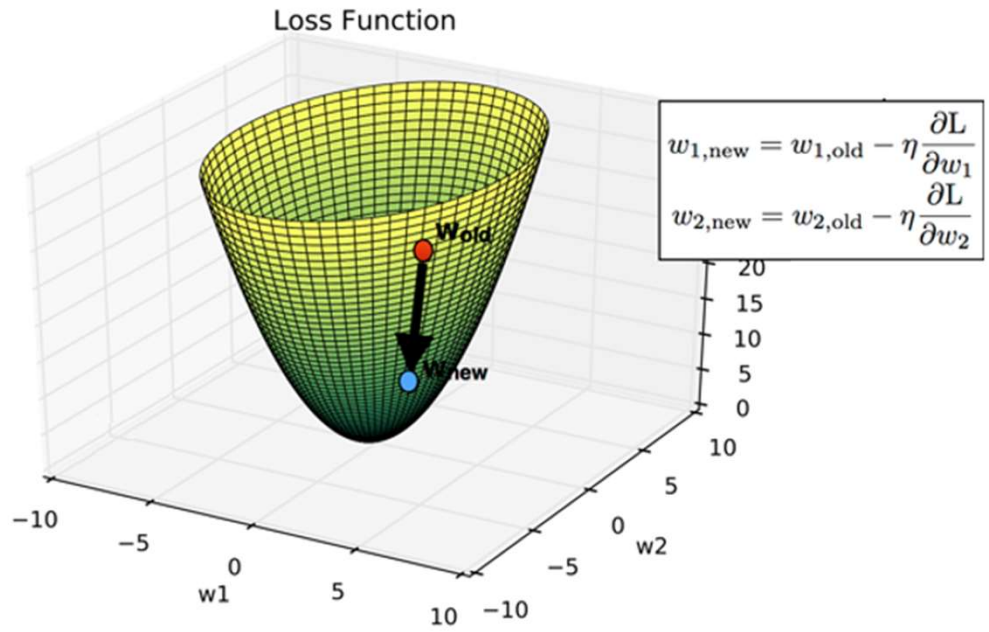


Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Multi-layer Neural Networks	
Rectifier, ReLU (Rectified Linear Unit)	$\phi(z) = max(0, z)$	Multi-layer Neural Networks	
Rectifier, softplus	$\phi(z) = \ln(1 + e^z)$	Multi-layer Neural Networks	

Backpropagation

- Training involves determining weights for each input to the perceptron
- Perceptron = TLU, Threshold Logic Unit
- Uses a loss function that can be optimized with gradient descent
- Backpropagation provides backward feedback to weights in the previous layers





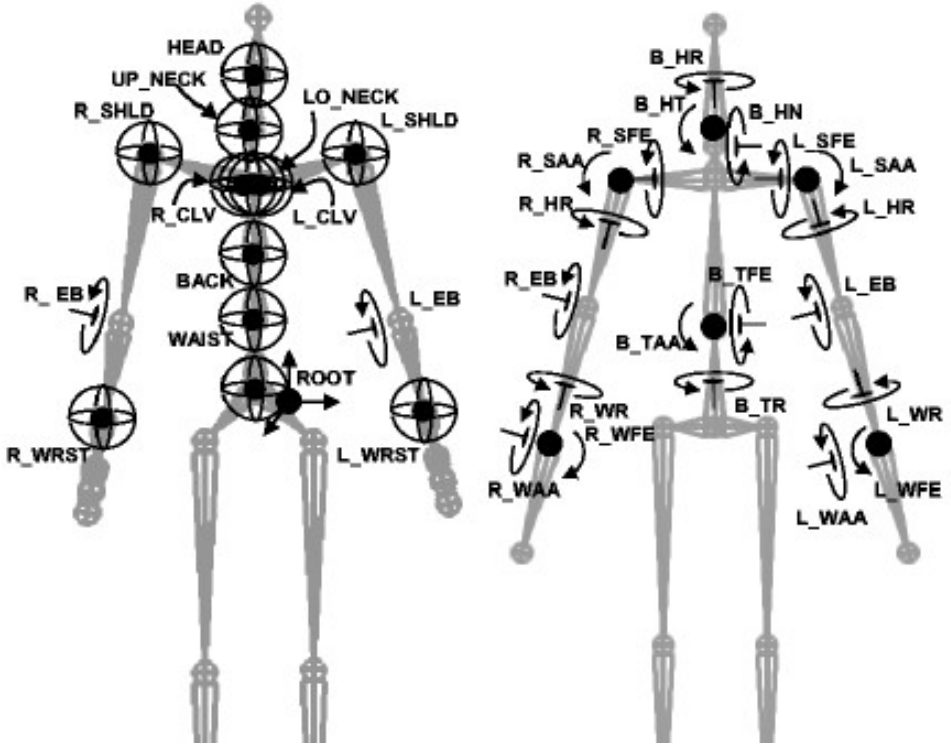
Machine vs. Human

Hard for humans:

- Large calculations
- Executing repetitive tasks reliably
- Detecting patterns in numbers

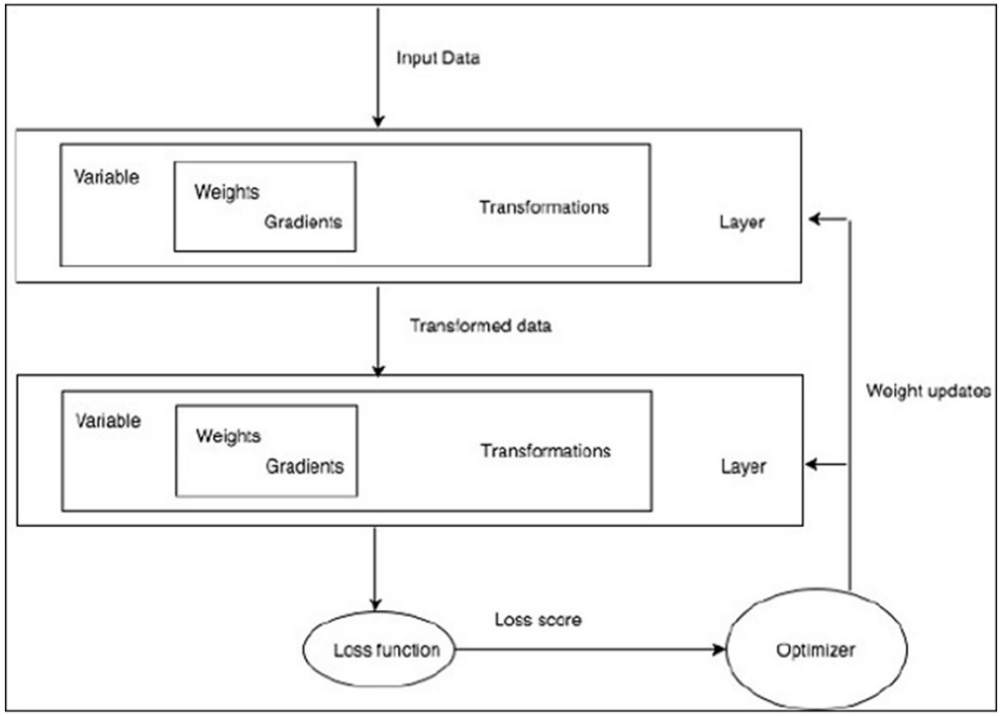
Hard for machines:

- Speech recognition & synthesis
- Working with many degrees of freedom in movement
- Humor, creativity, soul?
- Emotions



Training a deep learning algorithm

- Training a deep learning algorithm involves the following steps
  - Building a data pipeline (for example sklearn preprocessing)
  - Building a network architecture
  - Evaluating the architecture using a loss function
  - Optimizing the network architecture weights using an optimization algorithm



Opdracht

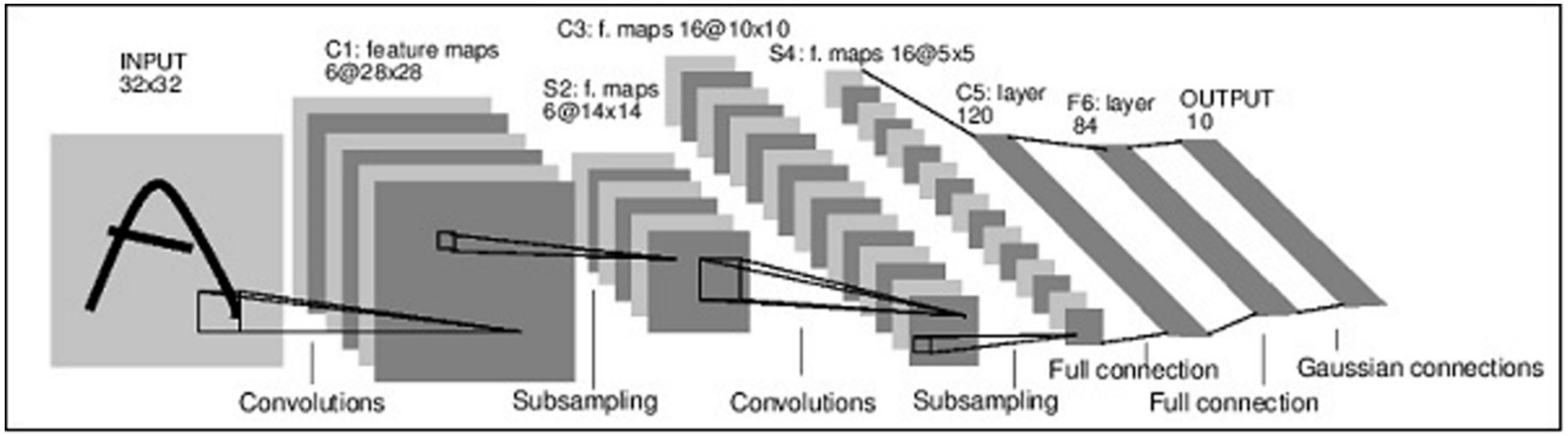
Maak een Multilayer Perceptron Classifier voor de MNST data

```
# Imports for machine learning
from sklearn.datasets import load_digits
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
```

Neural Network Architectures

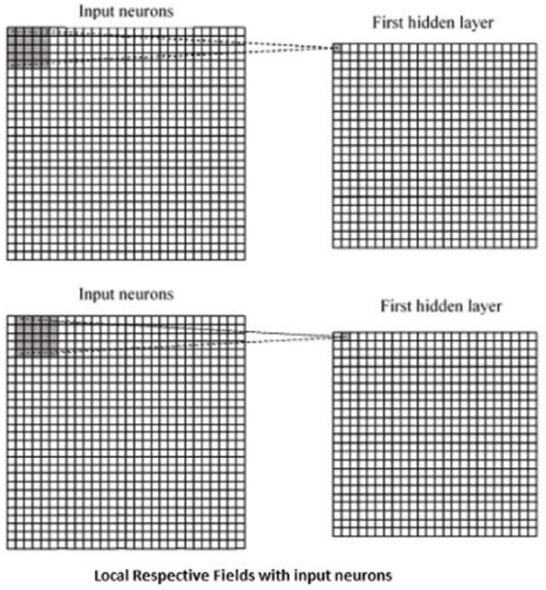
- Convolutional Neural Network – CNN
- Recurrent Neural Network - RNN

Convolutional Neural Network



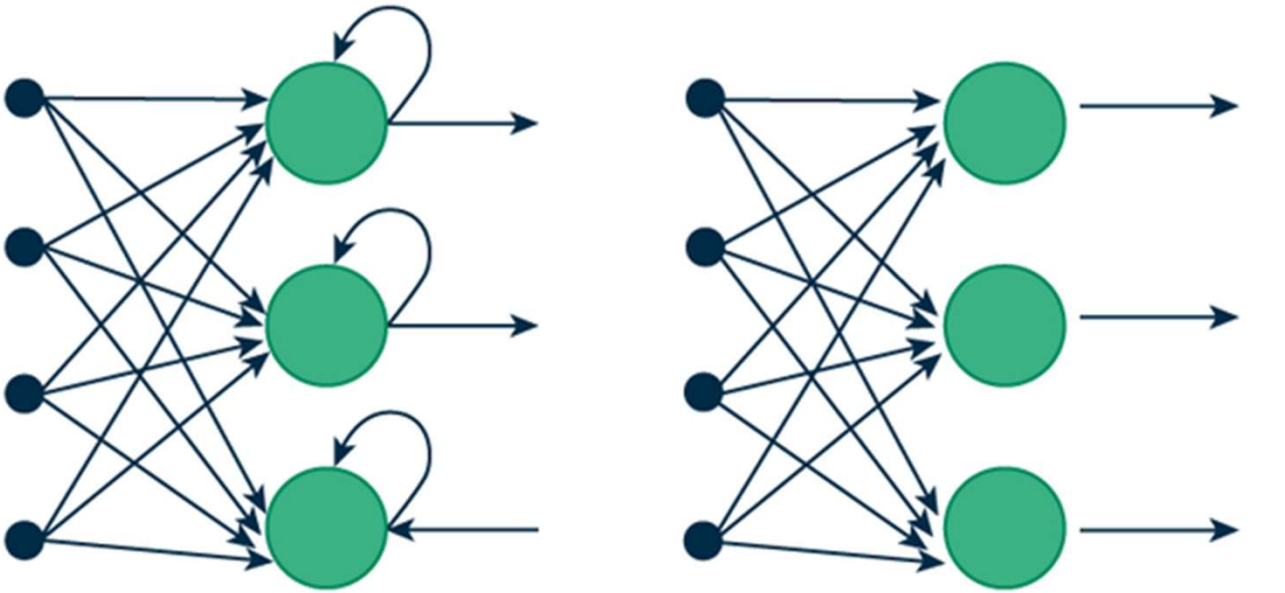
Key differences:

- Convolutional layers
- Pooling layers
- Is the context any different than long features of vectors?



Local Respective Fields with input neurons

Recurrent Neural Network



(a) Recurrent Neural Network (b) Feed-Forward Neural Network

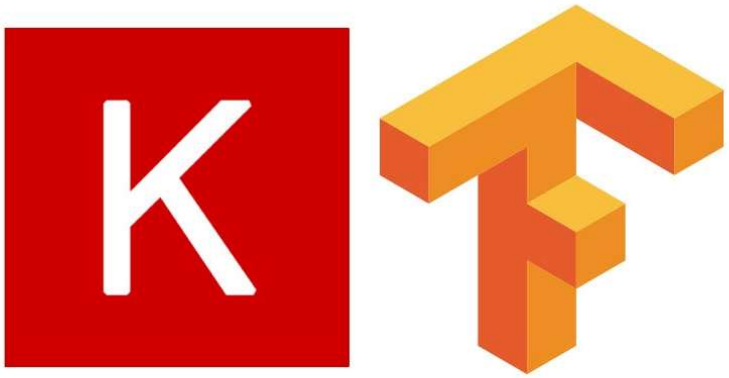
Keras & Tensorflow

TensorFlow

- Open source deep learning from Google
- Dataflow programing
- Built to run on (multiple) CPU/GPU's
- Wrappers in several languages like Python, C++ and Java

Keras

- Open source deep learning that became part of Tensorflow
- Modular, fast and easy to use
- Developed by Google Engineer François Chollet



PyTorch

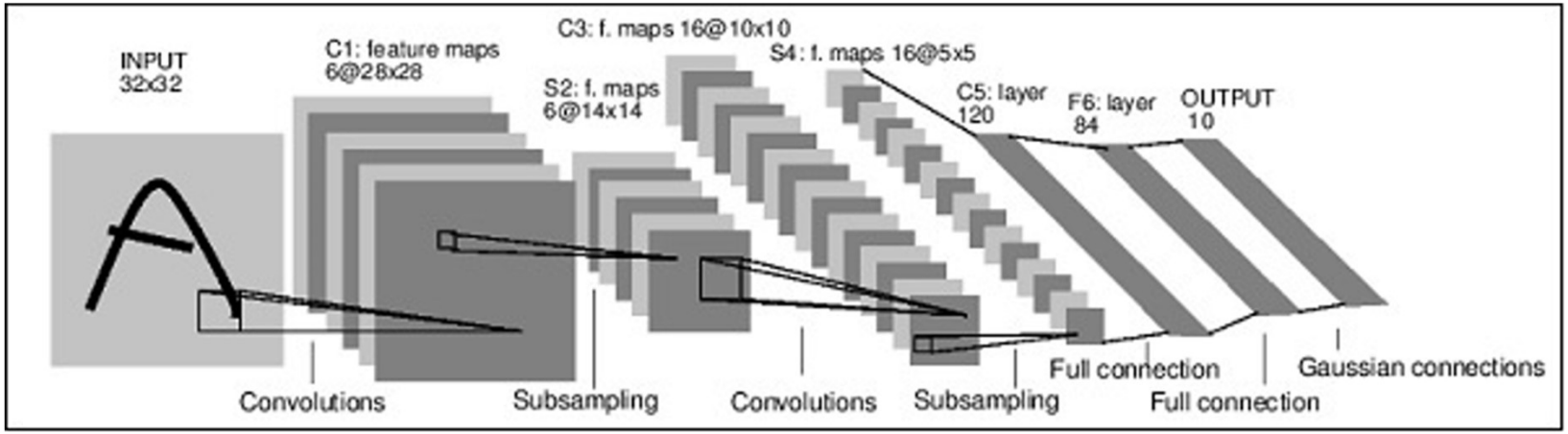
PyTorch

- Open source machine learning framework
- Based on the Torch library
- Used for applications such as computer vision and natural language processing
- Primarily developed by Facebook's AI Research lab (FAIR)

Keras

- Layers
  - Core layers — Dense — output = activation(dot(input, kernel) + bias)
  - Convolution layers — Convolution2D — spatial convolution over images
  - Pooling layers — MaxPooling2D — takes the maximum value over an pool window
  - Recurrent layers
  - Preprocessing layers
  - Normalization layers
  - Regularization layers
  - Attention layers
  - Reshaping layers
  - Merging layers
  - Locally-connected layers
  - Activation layers

LeNet5



Opdracht

Build a Neural Network with Keras to predict the category of the Fashion MNIST dataset.

1. Load the FashionMNIST daatset
2. Explore the dataset - EDA
3. Split in Training and Test dataset
4. Build a neural network with Keras, with correct output layer
5. Evaluate the results

https://playground.tensorflow.org/

Other Python Libraries

- PyCaret
- Statsmodels

```
In [184]: print(model.summary())
                            OLS Regression Results
==============================================================================
Dep. Variable:                      B   R-squared:                       0.000
Model:                            OLS   Adj. R-squared:                 -0.125
Method:                 Least Squares   F-statistic:                 0.0005452
Date:                Tue, 10 Oct 2017   Prob (F-statistic):              0.982
Time:                        11:44:28   Log-Likelihood:                -33.201
No. Observations:                  10   AIC:                             70.40
Df Residuals:                       8   BIC:                             71.01
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [95.0% Conf. Int.]
------------------------------------------------------------------------------
const        100.4481     32.127      3.127      0.014        26.363   174.533
A             -0.0076      0.327     -0.023      0.982        -0.762     0.746
==============================================================================
Omnibus:                        2.865   Durbin-Watson:                   2.352
Prob(Omnibus):                  0.239   Jarque-Bera (JB):                0.994
Skew:                          -0.191   Prob(JB):                        0.608
Kurtosis:                       1.504   Cond. No.                     1.33e+03
==============================================================================
```

MLaaS

- Machine Learning as a Service
- Microsoft Azure Machine Learning
- Google Cloud Machine Learning Engine
- AWS Machine Learning
- IBM Watson Machine Learning

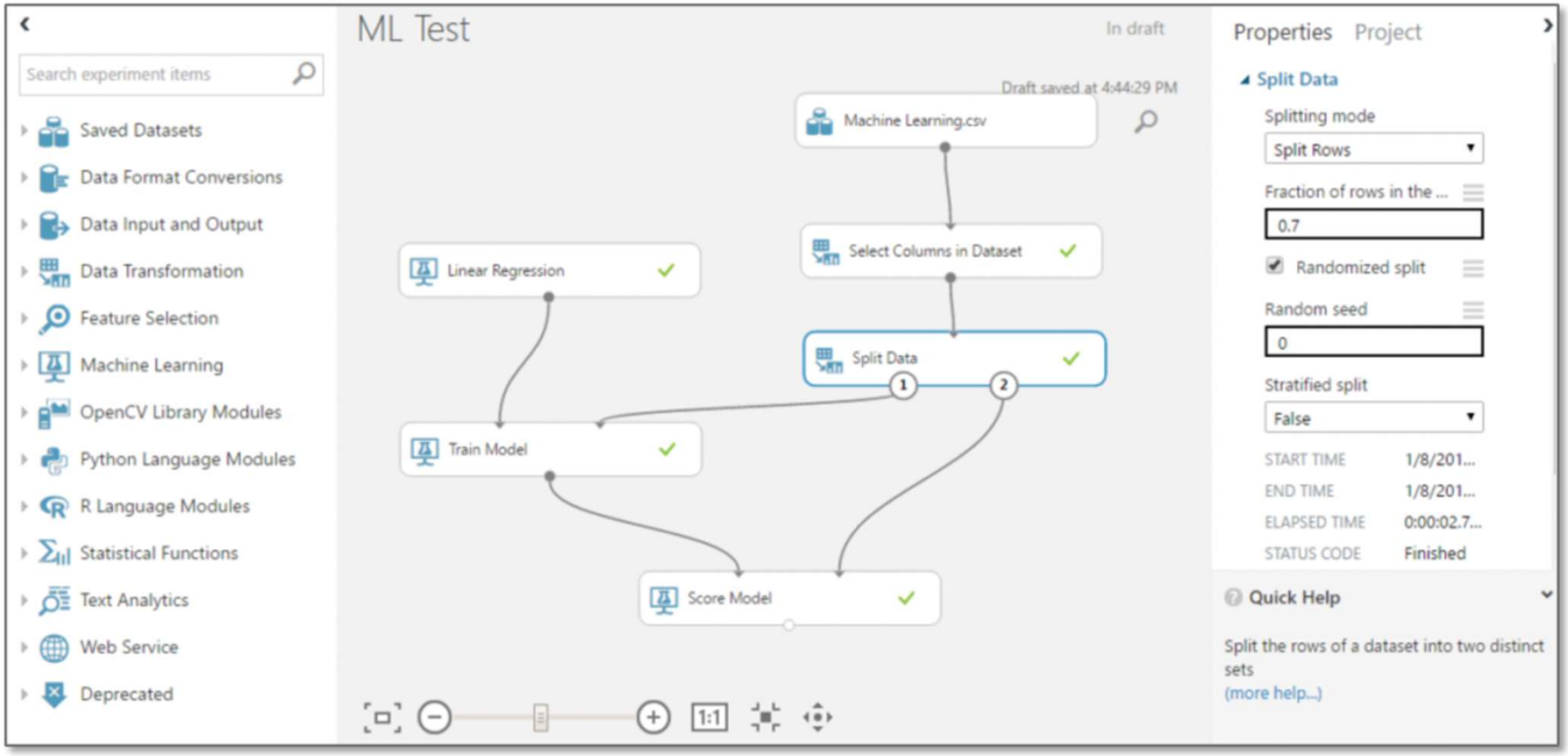
No-code Machine Learning

- BigML
- Create ML
- Data Robot
- Fritz AI
- Google Cloud AutoML
- Google ML Kit
- MakeML
- Microsoft Azure Automated Machine Learning
- Obviously AI
- RunawayML
- SuperAnnotate
- Teachable Machine
- RapidMiner

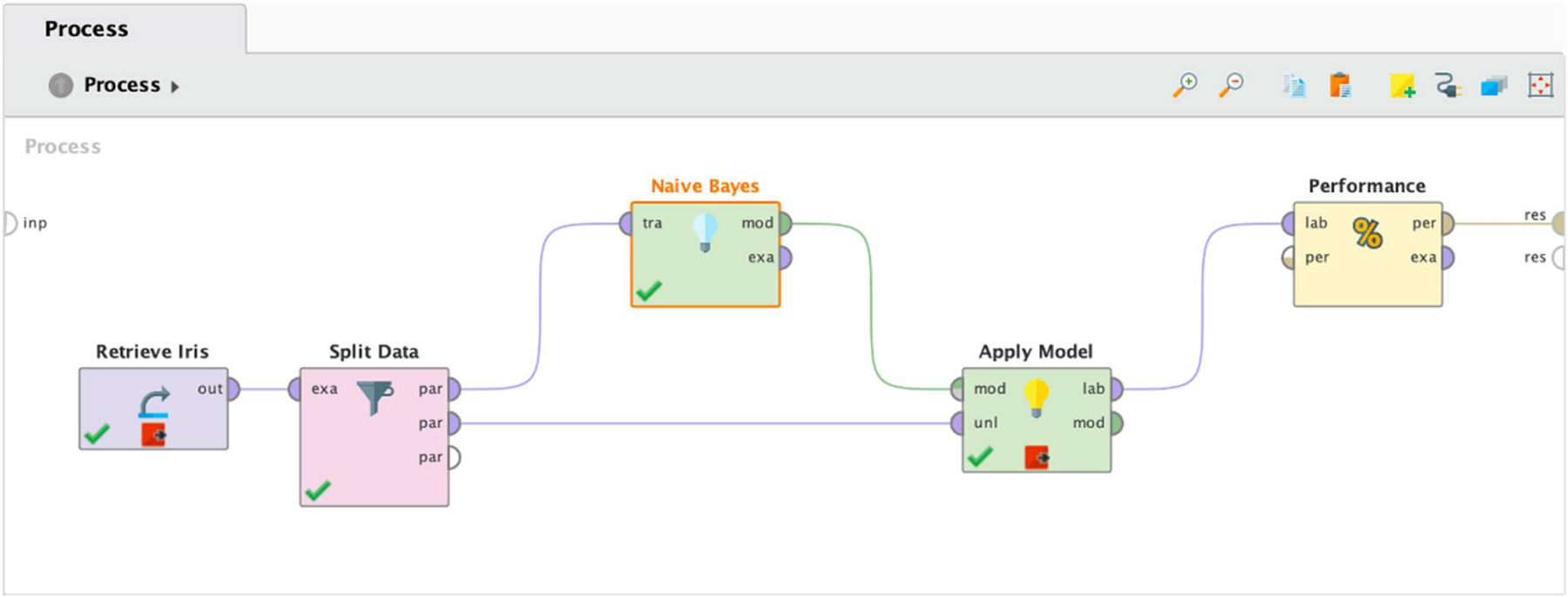
https://analyticsindiamag.com/top-12-no-code-machine-learning-platforms-in-2021/

Azure Machine Learning Studio

- A service for end-to-end machine learning lifcycle

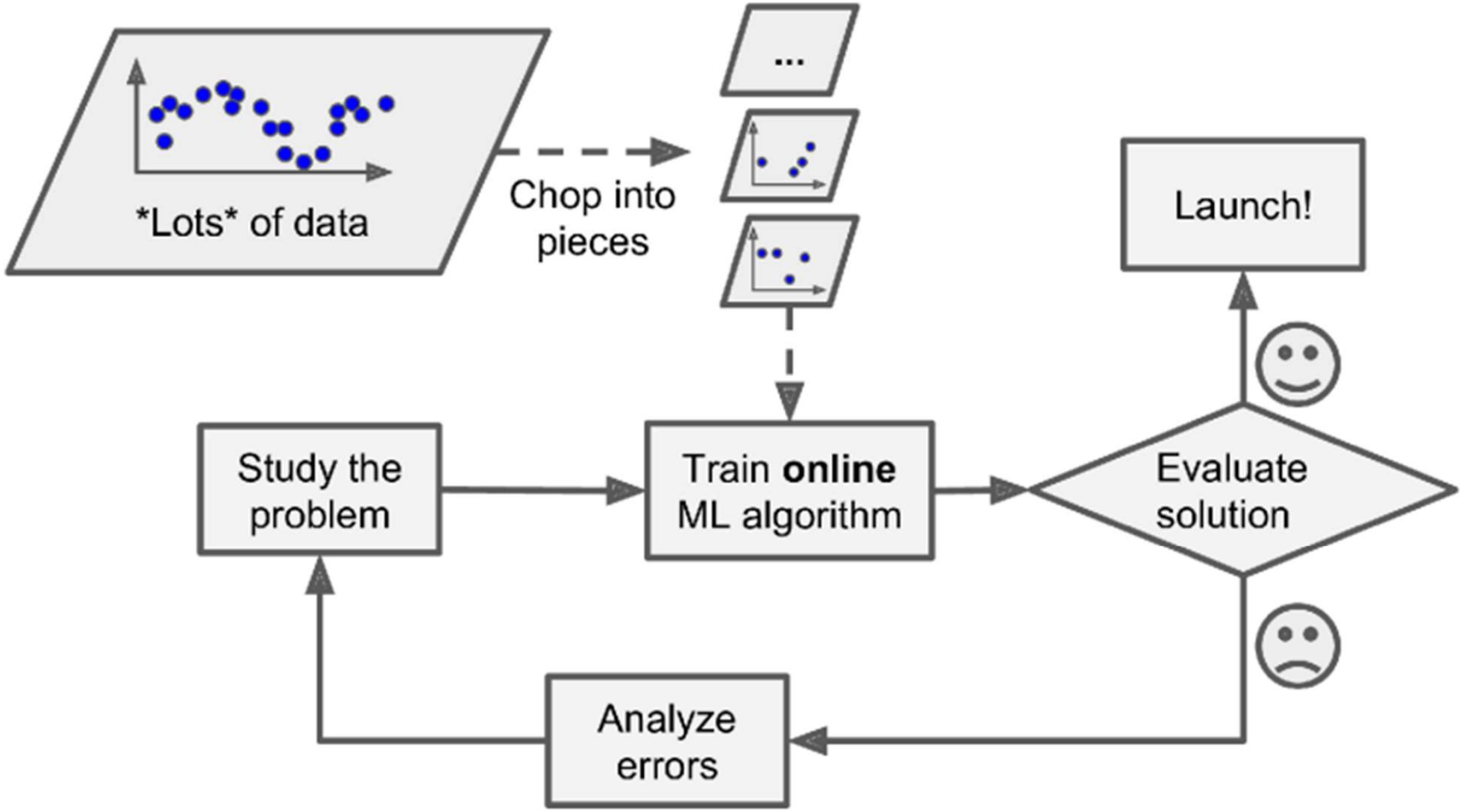


RapidMiner

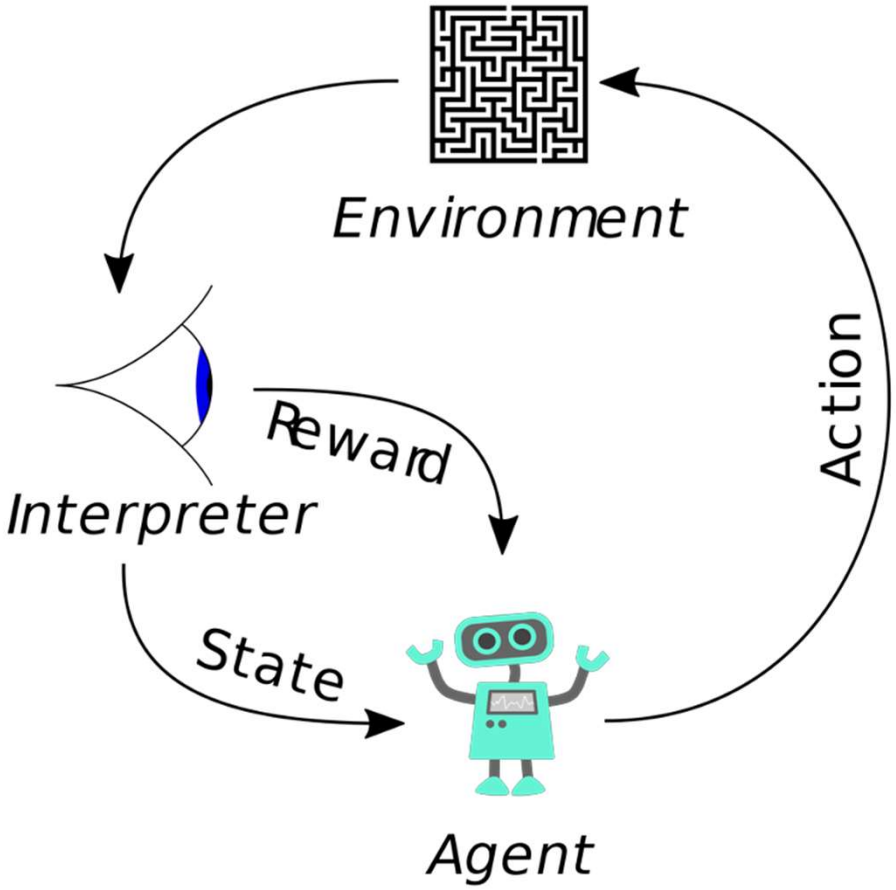


https://rapidminer.com

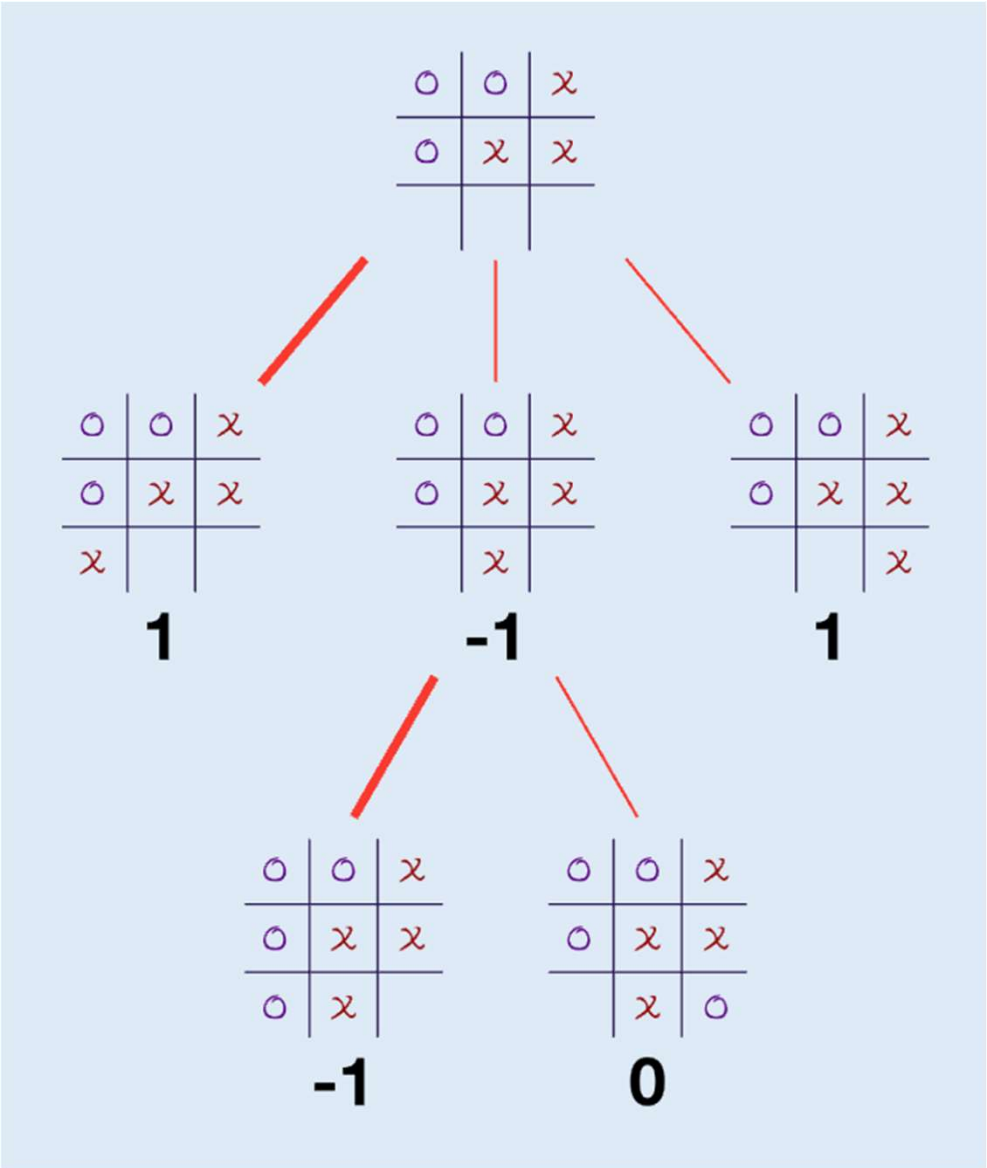
Online learning



Part 6 - Reinforcement Learning

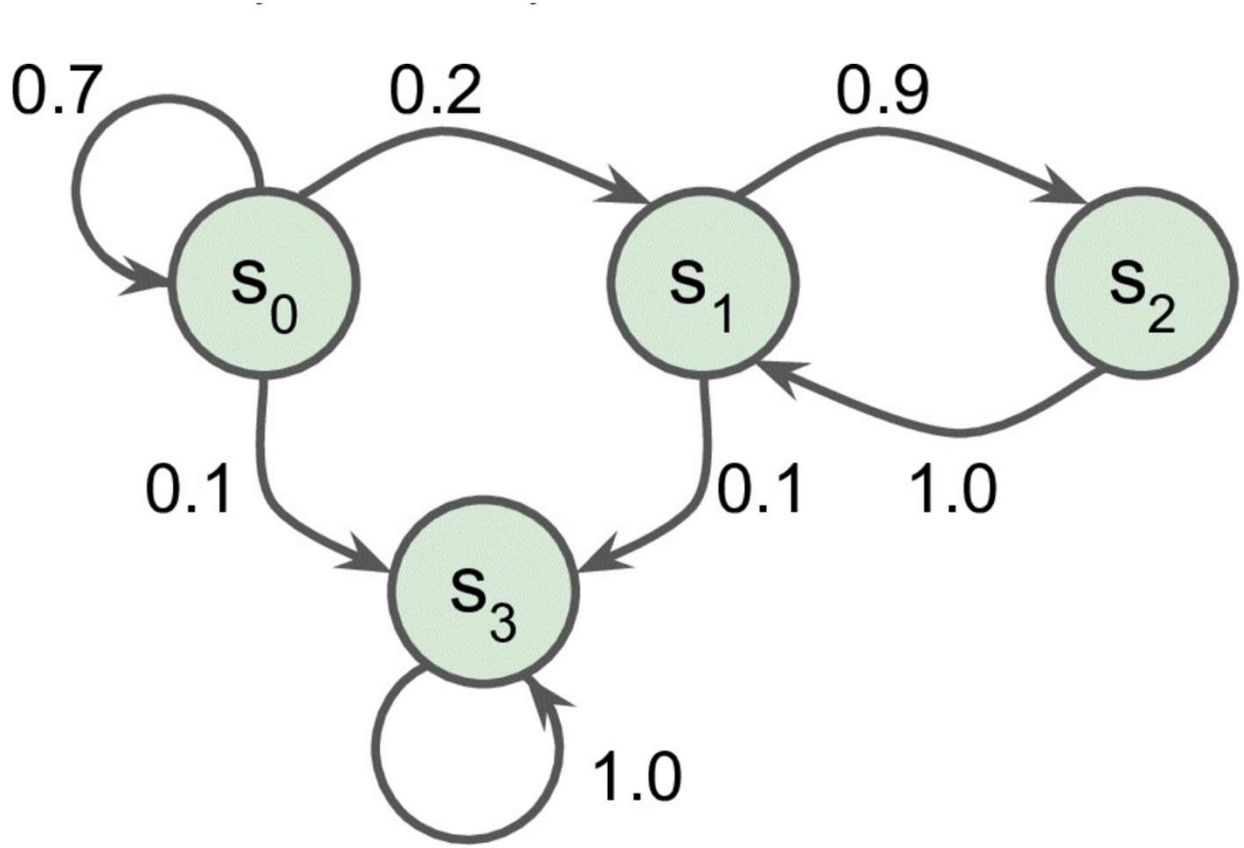


Tic Tac Toe



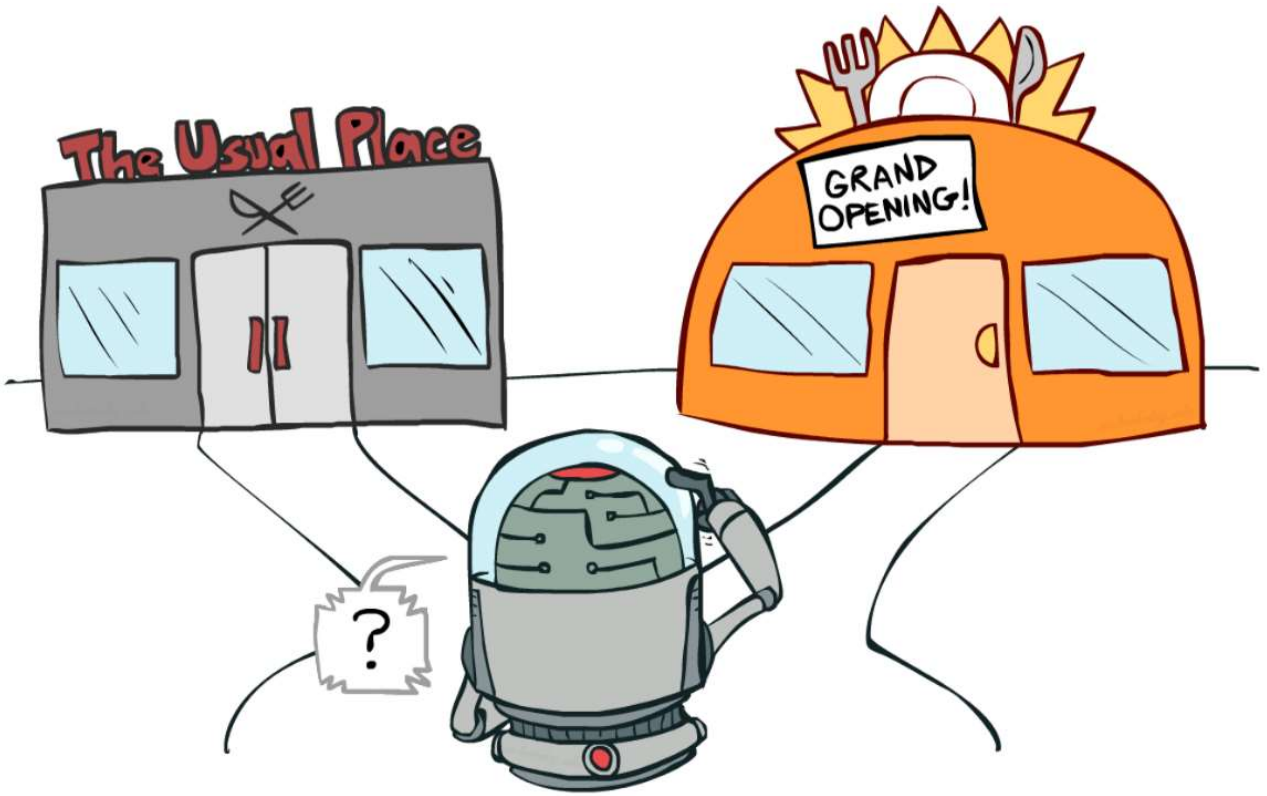
Policy

- Welke actie is het beste voor elke state?
  - Markov chains



Policy

- Exploration vs Exploitation



Algorithms

- Q-Learning
- Monte Carlo
- Deep Q Network

https://neptune.ai/blog/reinforcement-learning-basics-markov-chain-tree-search

Opdracht

Multi-Armed Bandits

```
n_arms = 3
arms = [{'mu': random.randint(100,5000), 'sigma’:
random.uniform(10,100)} for _ in range(n_arms)]

# GET REWARD FROM A SELECTED ARM IN YOUR TRIALS
arm_idx = 0
reward = np.random.normal(arms[arm_idx]['mu'],arms[arm_idx]['sigma']
```

Algorithm of Epsilon-Greedy

1. Initialize the estimated values of all arms to zero or a small positive number.
2. For each trial:
    1. Generate a random number between 0 and 1
    2. If the number is less than ϵ, select a random arm (exploration).
    3. Otherwise, select the arm with the highest estimated reward (exploitation).
    4. Update the estimated reward of the selected arm based on the observed reward.

Fair AI

- Fairness

Explainable AI

- White Box Models
  - Ad-hoc explanations
- Explainability Methods
  - Post-hoc Feature Importance
  - LIME – Local Interpretable Model-agnostic Explanations
  - SHAP - SHapley Additive exPlanations
  - LOCO - Leave One Covariate Out
- Bias Detection

Data Sets

- CBS
- NL Overheid
- WorldBank
- Kaggle
- OECD Better Life Index - https://stats.oecd.org/index.aspx?DataSetCode=BLI
- IMF - https://www.imf.org/en/Data

Open Data Sources

- Popular open data repositories
  - UC Irvine Machine Learning Repository
  - Kaggle datasets
  - Amazon's AWS datasets
- Meta portals (they list open data repositories)
  - Data Portals
  - OpenDataMonitor
  - Quandl
- Other pages listing many popular open data repositories
  - Wikipedia's list of Machine Learning datasets
  - Quora.com
  - The datasets subreddit

Links

- TensorFlow Neural Network Playground - https://playground.tensorflow.org
- Hide and Seek - https://youtu.be/kopoLzvh5jY
- Coast Runners AI - https://youtu.be/O5xeyoRL95U?t=1198
- MNIST Digit Recognizer - http://www.ccom.ucsd.edu/~cdeotte/programs/MNIST.html