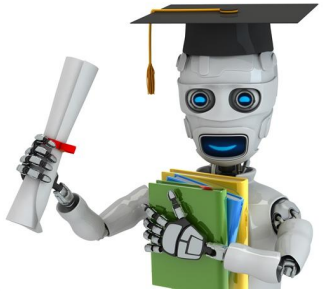
Machine learning para problemas de hoy:

un enfoque para la industria.

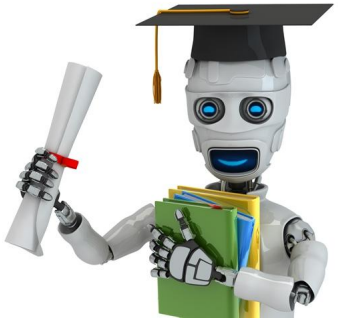


Machine learning Meet-up Medellín

CIENTIFICAS DE DATOS

ML Meetup

+ Meetups



Facebook:
Machine Learning Colombia

OPEN DATA

Open Data

CIENTIFICAS
DE DATOS

Team

Juan Camilo Vásquez

Email:

juancamilo0628@gmail.com

Twitter: **@jcvasquezc1**

http://jcvasquezc.wix.com/home

Sebastián Pineda

Email:

sepia92@hotmail.com

Andres Velez

Email:

andres.velez.e@gmail.com

linkedIn: anvelezec

What is Machine Learning?


Time, Money, Human Resources


BIG
DATA
STORAGE
ANALYTICS
TECHNOLOGIES
USE
ONE
SINCE
SOFTWARE
UBIQUITOUS
REQUIRING
BUSINESS
MOVING
ORGANIZATIONS
RADIO-FREQUENCY
COMPLEXITY
SOLID
WIRELESS
SAN
TOLERABLE
PARALLEL
SIZE
NEEDED
QUALITIES
BURIED
PETABYTES
PROCESSING
LOGS
INTERNET
USING
TYPES
WORKING
GARTNER
MASSIVELY
RELATED
ALSO
PERFORMANCE
BIOLOGICAL
HUNDREDS
CREATED
DISK
RELATIONAL
ZETTABYTES
GENOMICS
COMPUTING
TOOLS
SET
SHARED
COMBAT
SIGNIFICANT
TIME
WITHIN
PROCESS
INCLUDE
SYSTEMS
DEFINITION
BIOGEOCHEMICAL
NETWORKS
RECONSIDER
SEARCH
OPPORTUNITIES
INFORMATION
CONNECTOMICS
RECORDS
DESKTOP
CURRENTLY
FC
TENS
CAPACITY
FORMS
PRESENTATIONS
WORLD'S
PRACTITIONERS
NOW
USED
DISTRIBUTED
CAPTURE
MAY
MANAGEMENT
MANAGE
GROW
SOCIAL
LARGE
STORE
DEFINING
CASE
EVERY
LARGER
COST
CONTINUES
CITATION
TERABYTES
MPP
RESEARCH
COMPLEX
DATABASES
INDEXING
DIFFICULTY
TARGET
ABILITY
SETS
EXAMPLES
APPLIED
AMOUNT
DESCRIBING
ELAPSED
CURRENT
THOUGHT
SENSOR
ARCHIVES

Machine learning: key concept

Available data

Machine learning algorithm

Predictions for new data

PetApp

My Pets

husky

Shadow

golden

Spike

PetApp

My Pets

husky

Shadow

golden

Spike

???

Nuke

Traditional approach

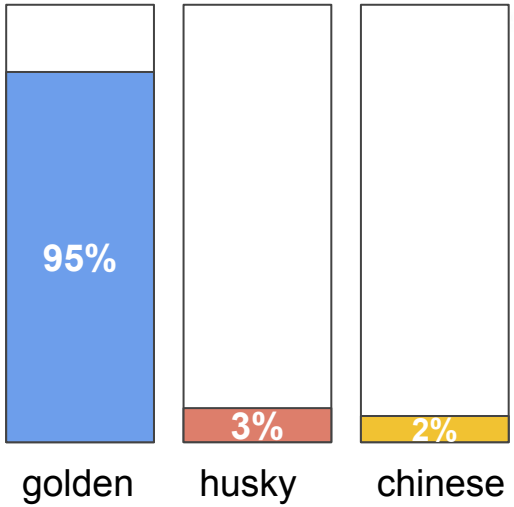
```
class DogClassifier(object):

    def __init__(self, image):
        # ...

    def calculate_dog_size(self):
        # ...

    def estimate_dog_color(self):
        # ...

    def estimate_tail_length(self):
        # ...

    # ....

    def is_there_even_a_dog_in_the_image(self):
        # ...
```

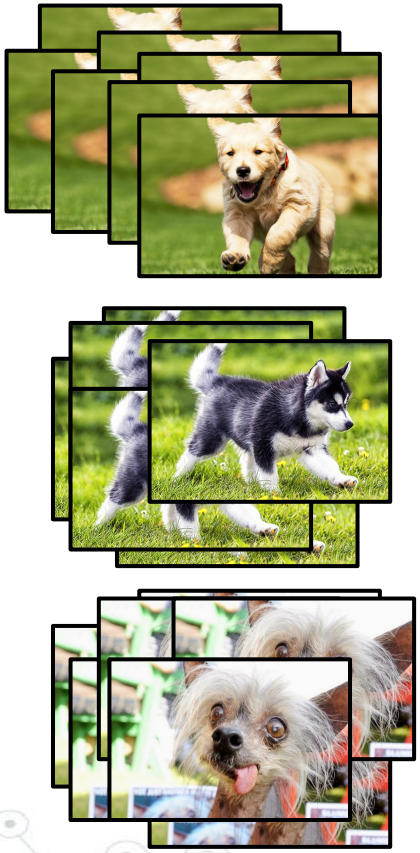
Traditional approach

```
class DogClassifier(object):

    def __init__(self, image):

    de          dog

    def es             (self):
        #

                          f):

    def is_there_even_a_dog_in_the_image(self):
        # ...
```


L

score=98%





golden

husky

chinese

x **y**

$$y = f(x \mid \theta)$$

f

golden

x

y

Machine Learning as an Industry

50,000+
employees

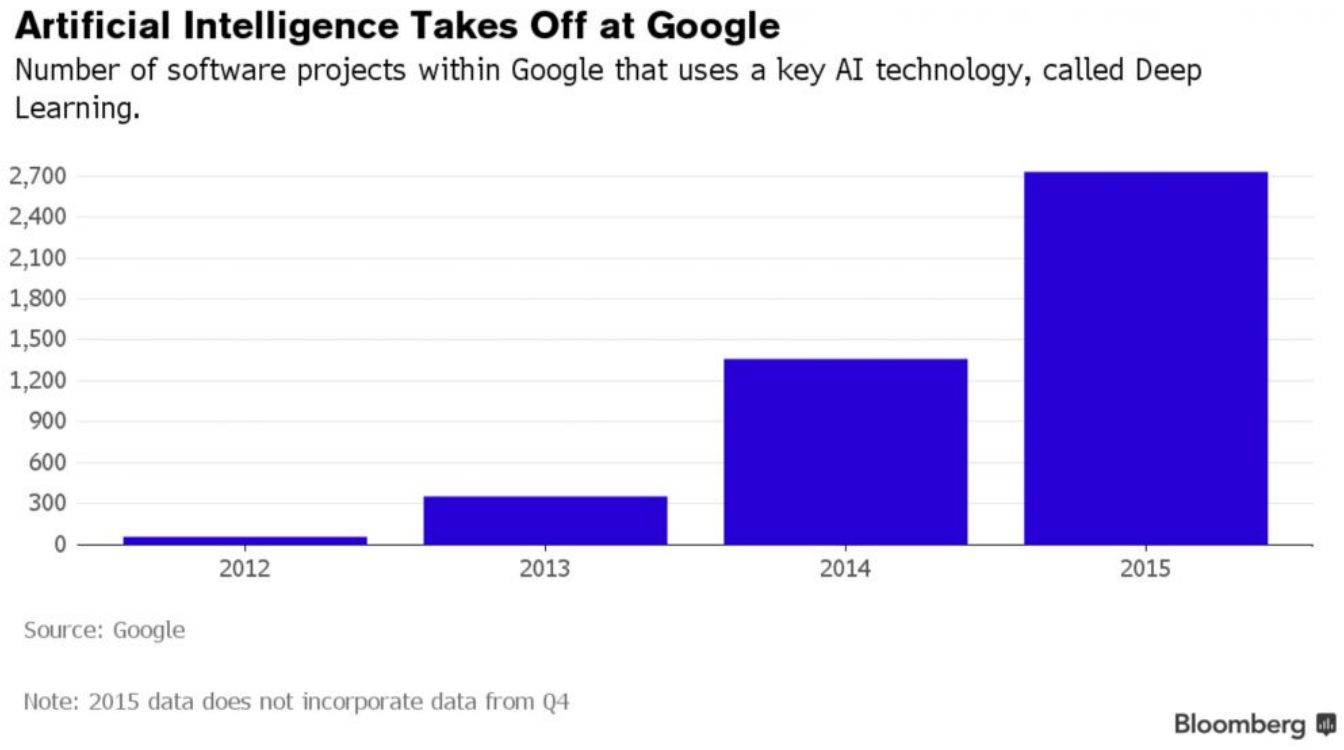
247
Companies/Products

7
Billion Dollar Companies

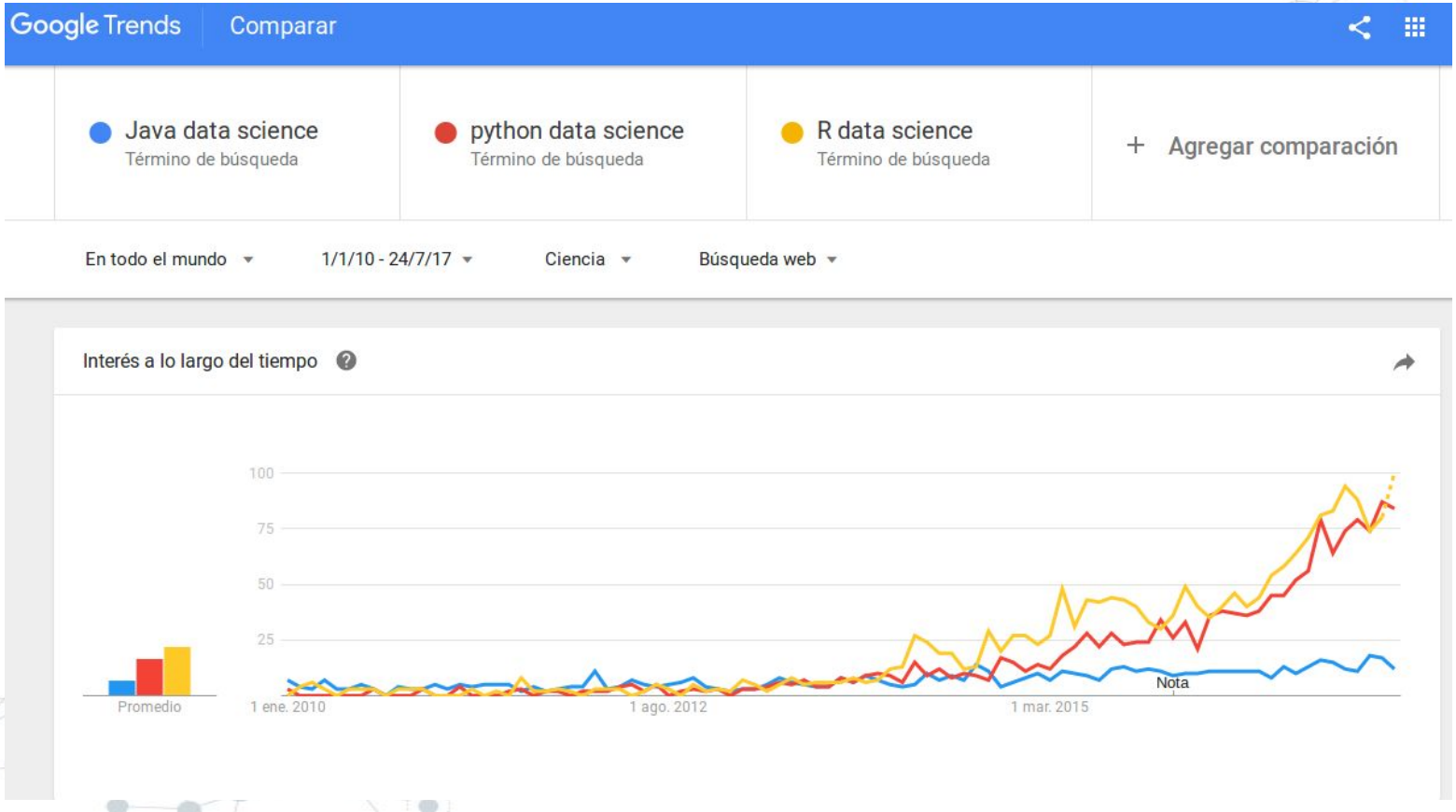
$23B
funding

$107B
Value Created

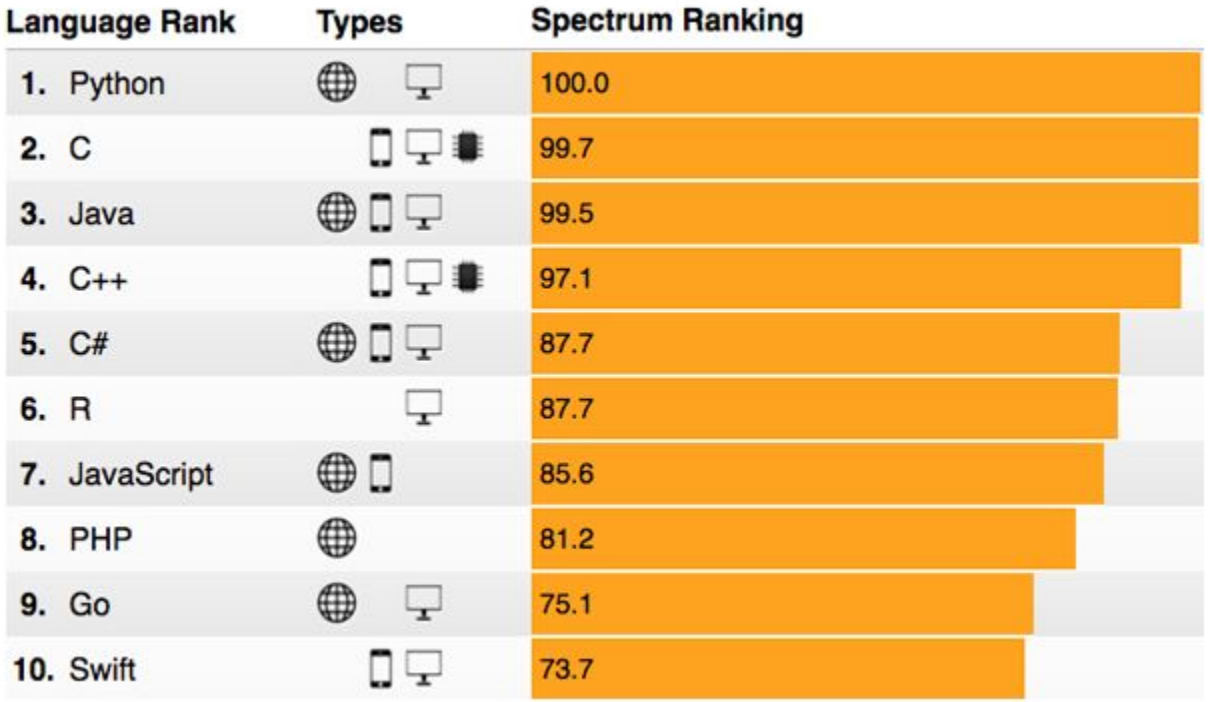
Trends


Artificial Intelligence Takes Off at Google
Number of software projects within Google that uses a key AI technology, called Deep Learning.
2,700
2,400
2,100
1,800
1,500
1,200
900
600
300
0
2012
2013
2014
2015
Source: Google
Note: 2015 data does not incorporate data from Q4
Bloomberg

Trends

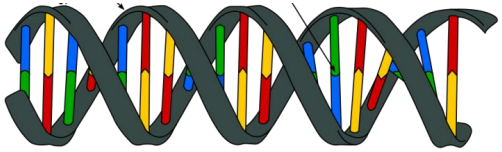

Google Trends
Comparar
Java data science
Término de búsqueda
python data science
Término de búsqueda
R data science
Término de búsqueda
Agregar comparación
En todo el mundo
1/1/10 - 24/7/17
Ciencia
Búsqueda web
Interés a lo largo del tiempo
100
75
50
25
Promedio
1 ene. 2010
1 ago. 2012
1 mar. 2015
Nota

Trends

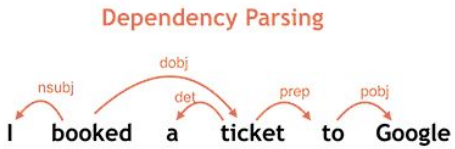


Source: IEEE Spectrum 2017:
http://spectrum.ieee.org/computing/software/the-2017-top-programming-languages

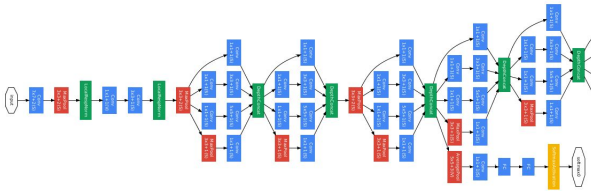
Applications



Genetics

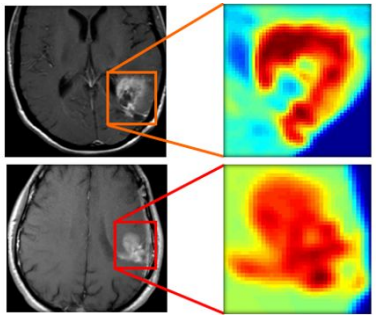


Natural Language Processing



Deep Learning

Self Driving Cars

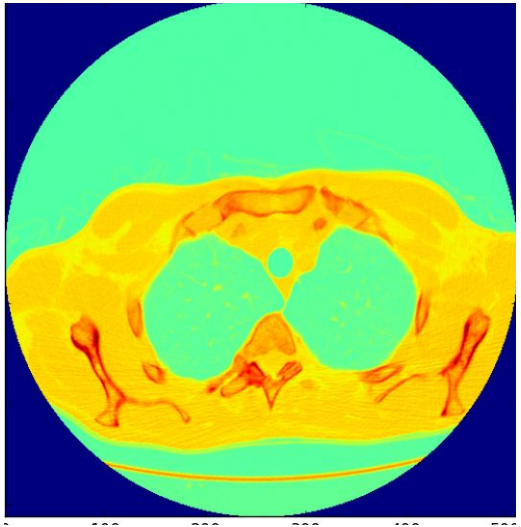


Healthcare

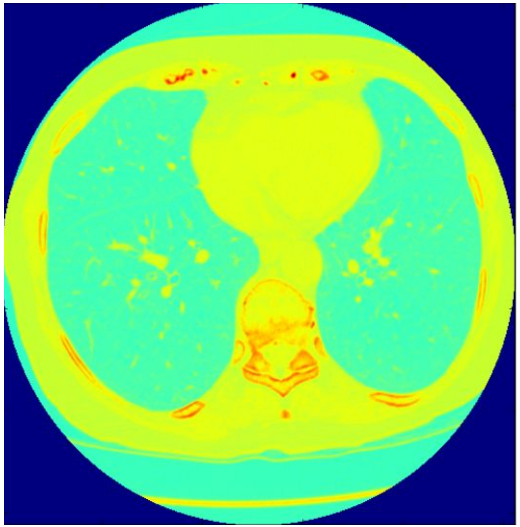
2017 Worldwide

Applications: human computer interaction

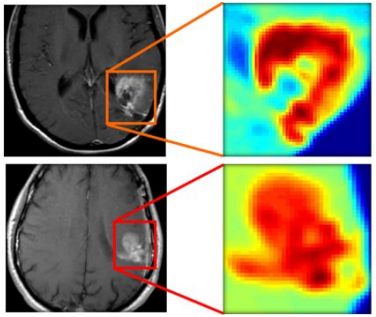
Applications: healthcare



cancer



healthy



Applications: business analytics

GRUPO BIMBO

Grupo Bimbo Inventory Demand

Maximize sales and minimize returns of bakery goods

$25,000 · 1,969 teams · a year ago

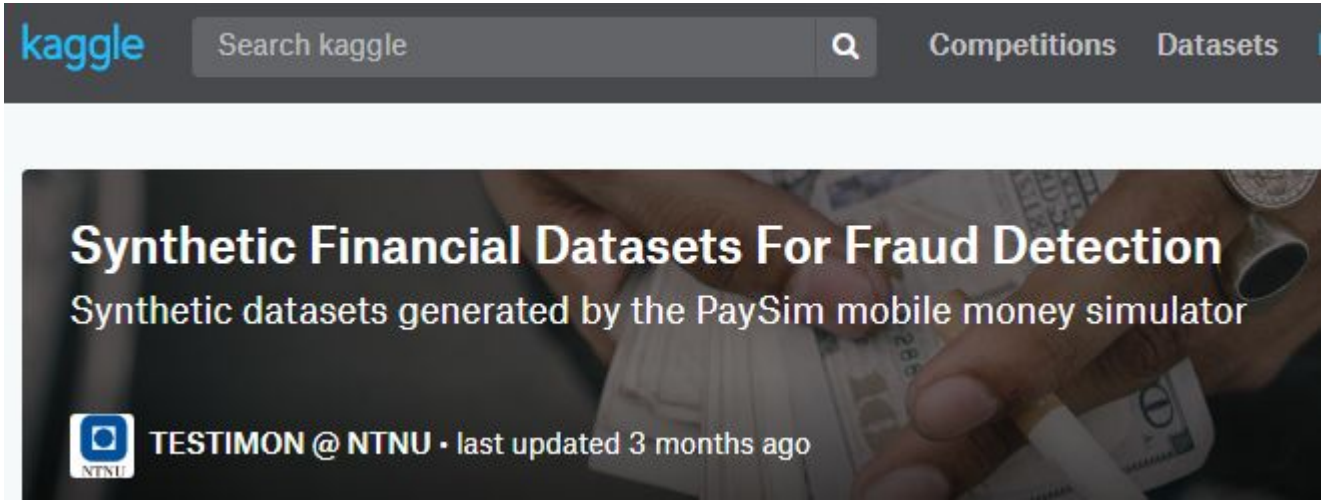
Santander

Santander Customer Satisfaction

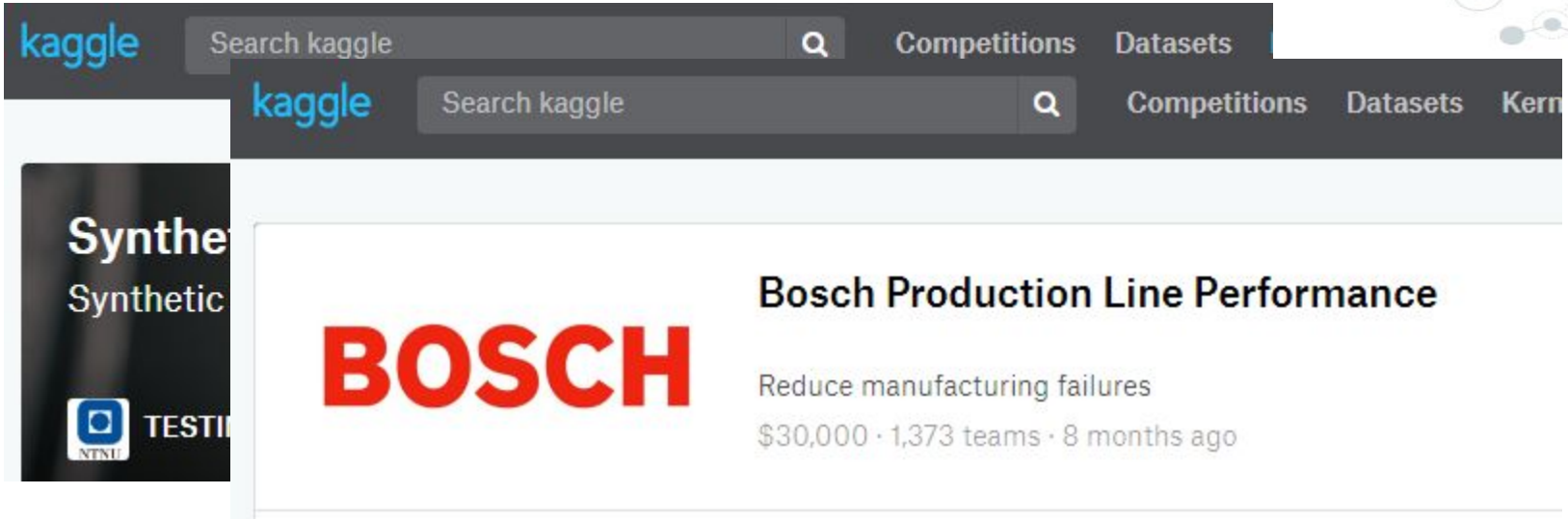
Which customers are happy customers?

$60,000 · 5,123 teams · a year ago

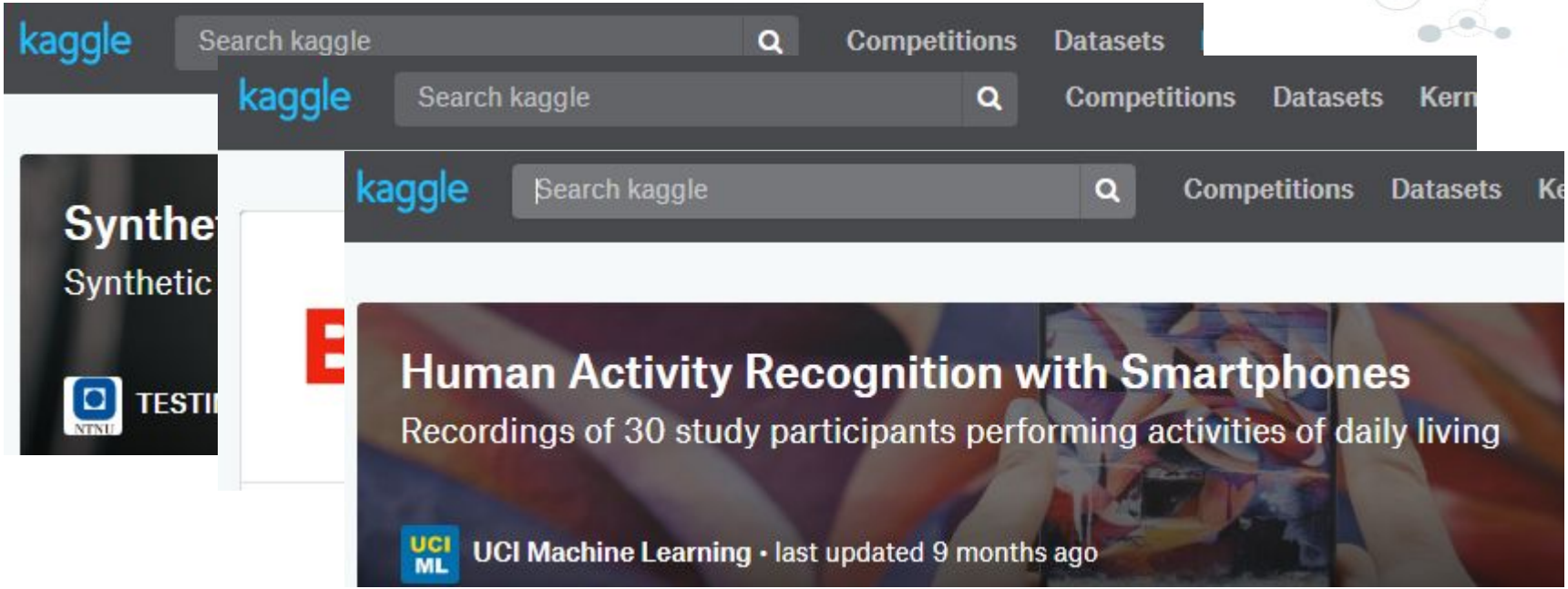
Applications



Applications



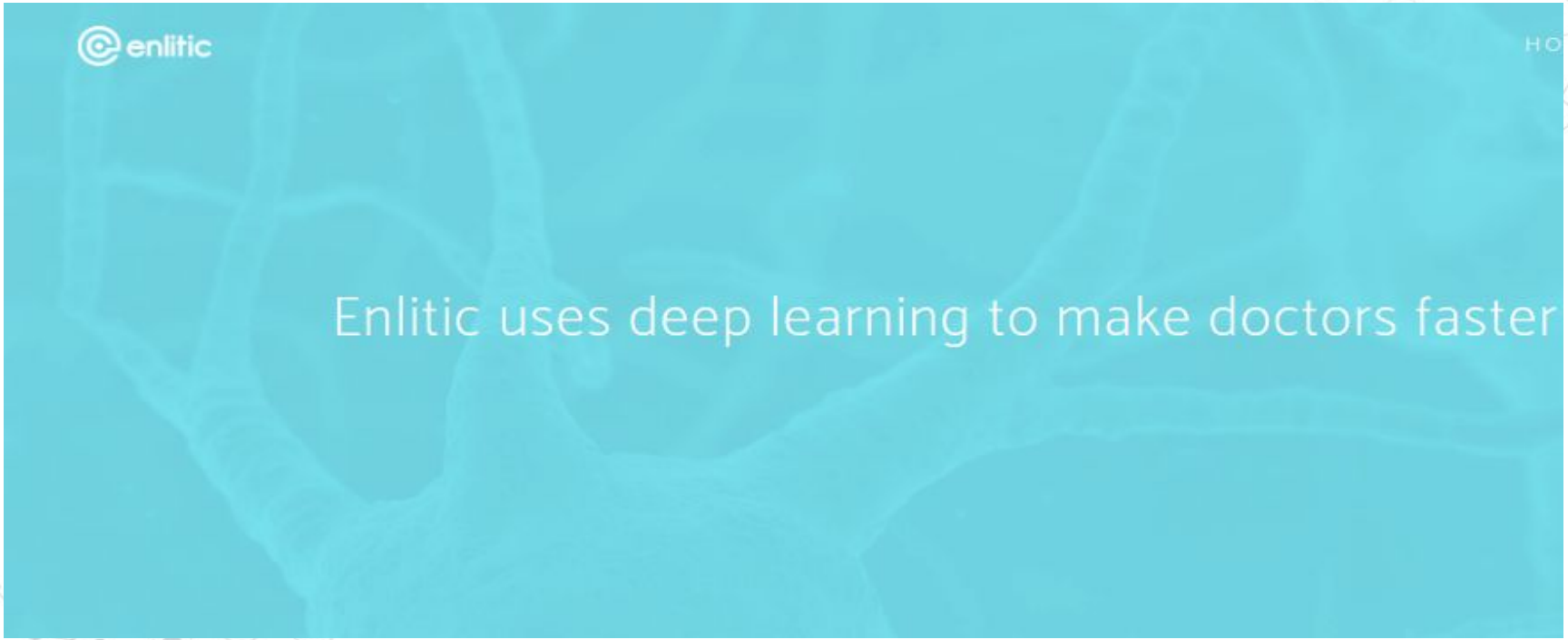
Applications



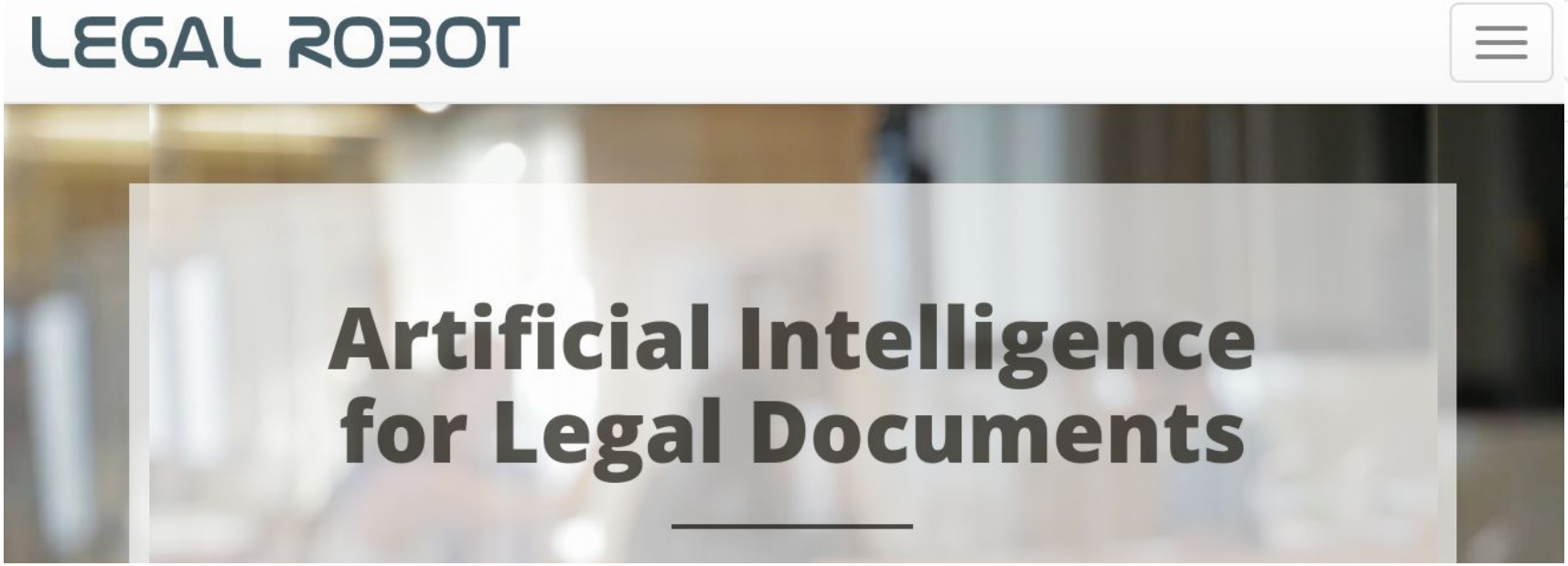
Success stories


NETFLIX
Movie Recommendations

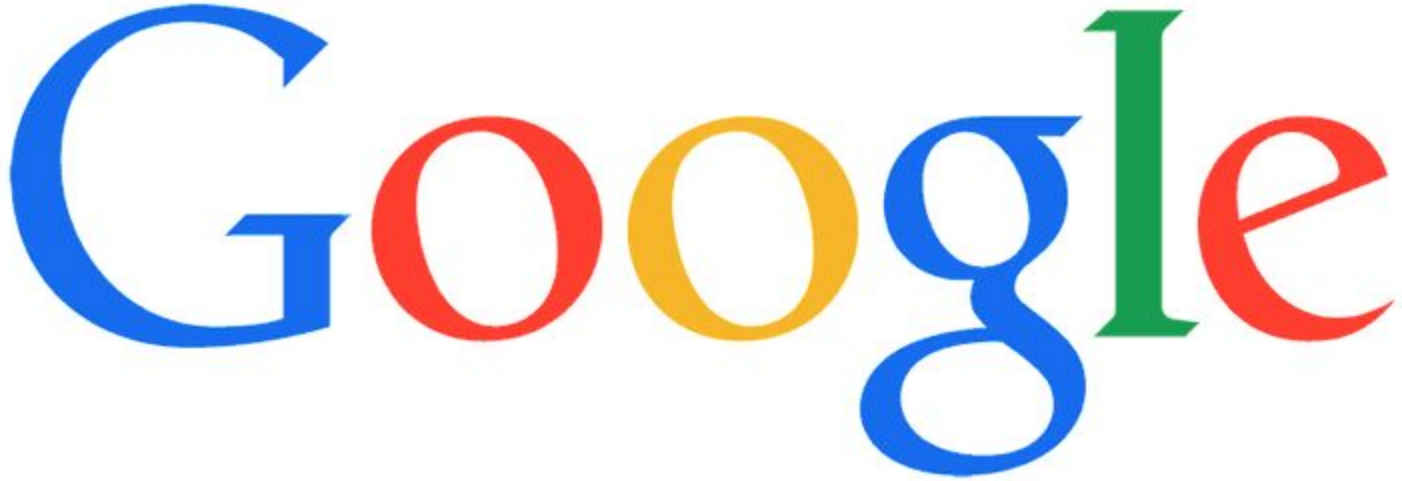
Enlitic



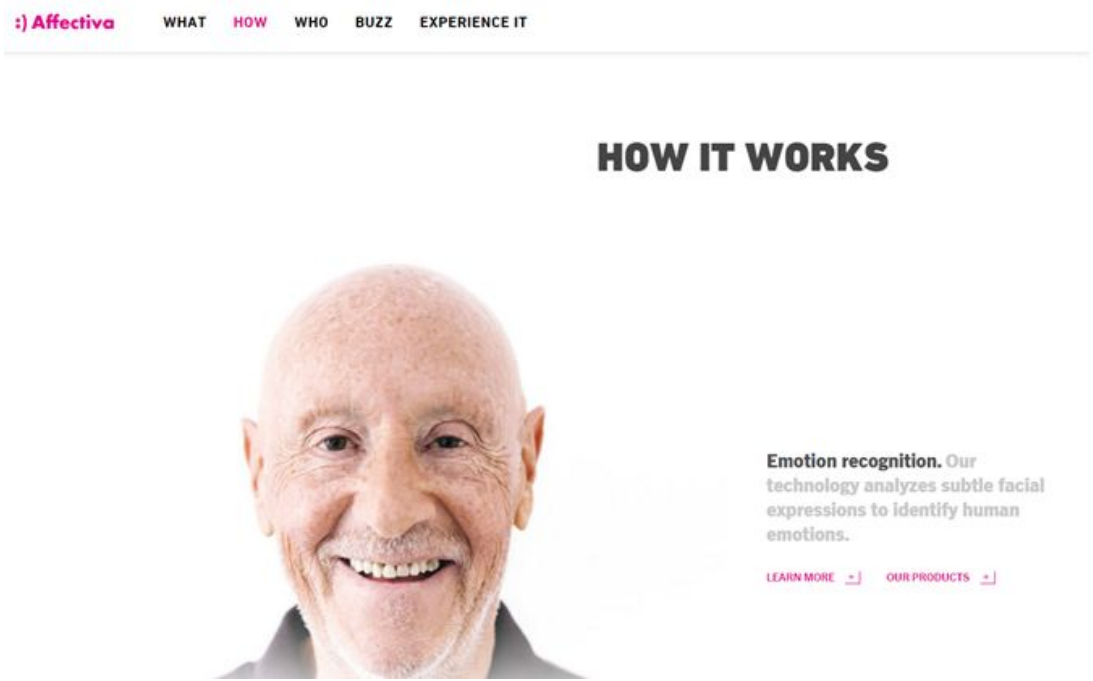
Product Recommendations

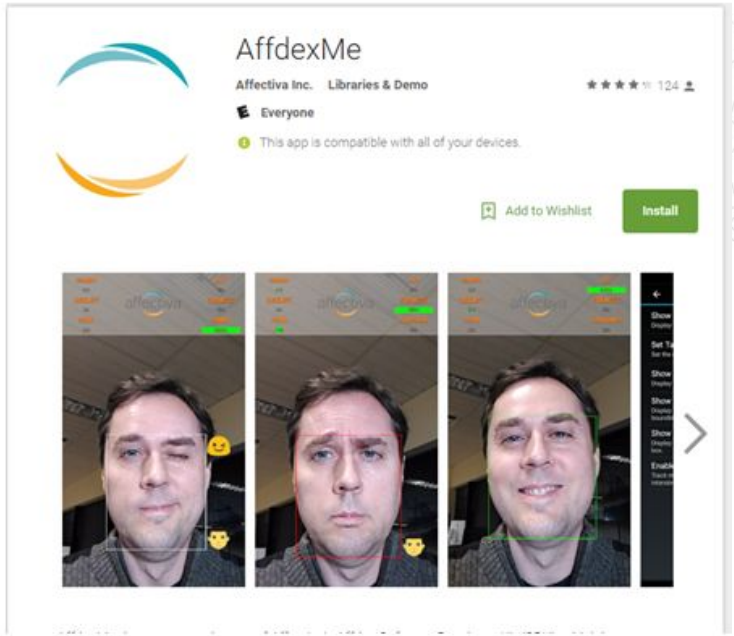


Error checking, Correction Recommendations, Risk prediction



Search, Translation, Ads, Google Now, Google Images, Self Driving Cars, ...



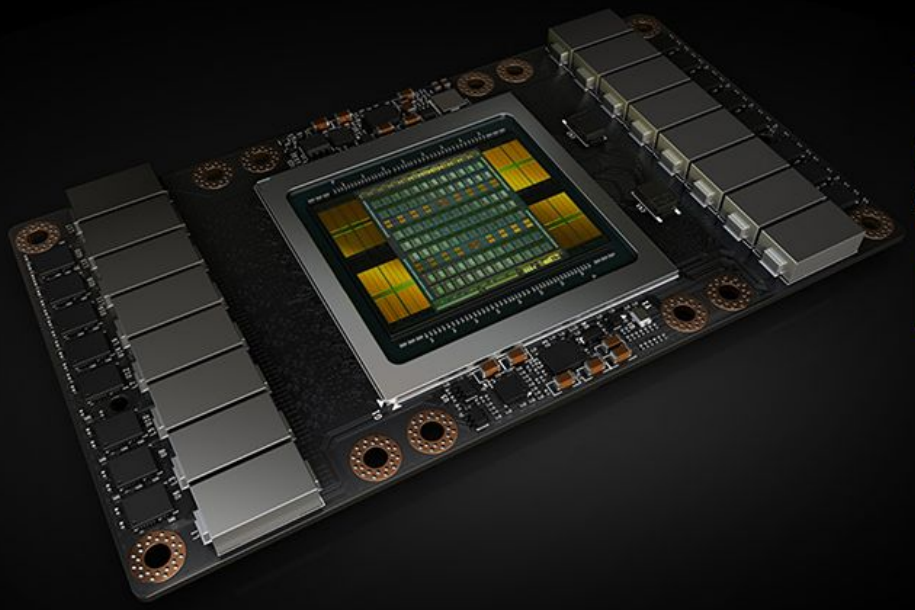


Emotion recognition from facial expressions and speech

NVIDIA

DEEP LEARNING AI

WHAT'S NEW | INDUSTRIES | FOR DEVELOPERS | PRODUCTS | EDUCATION | AI STARTUPS



THE CORE OF AI

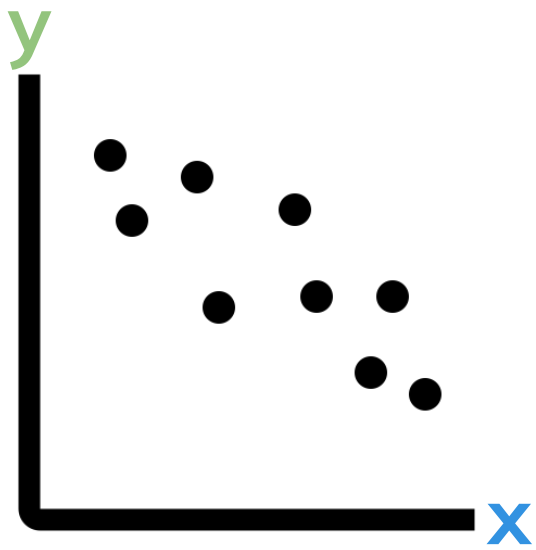
NVIDIA Volta is the new driving force behind artificial intelligence.

LEARN MORE

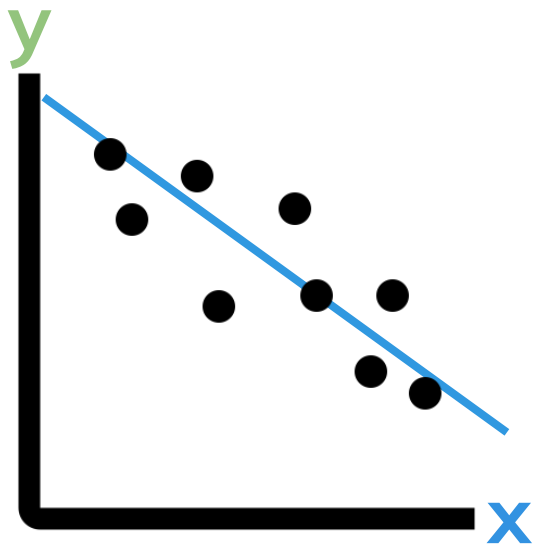
Machine learning algorithms

Regression

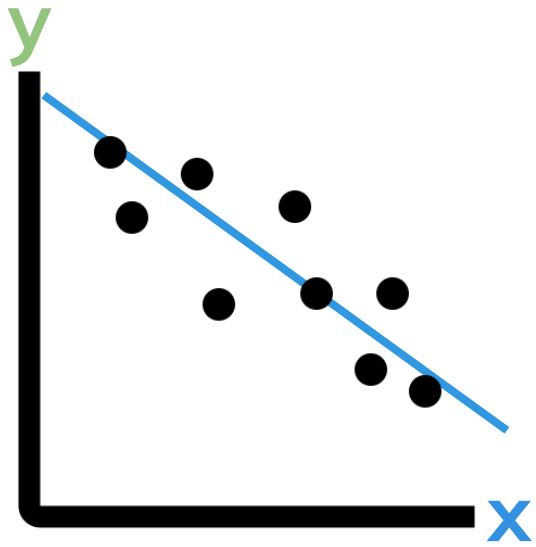
Regression



Regression

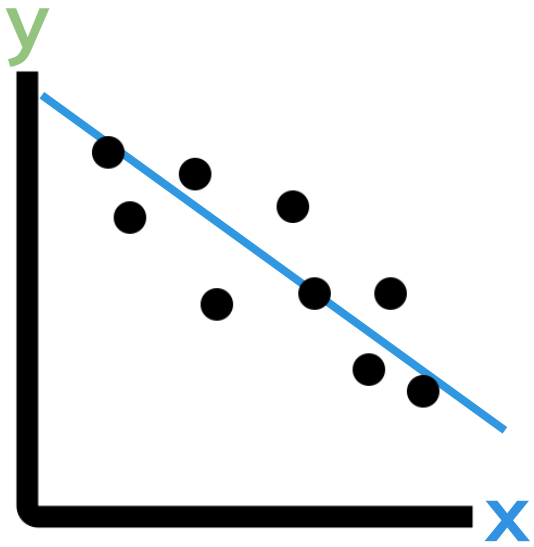


Regression



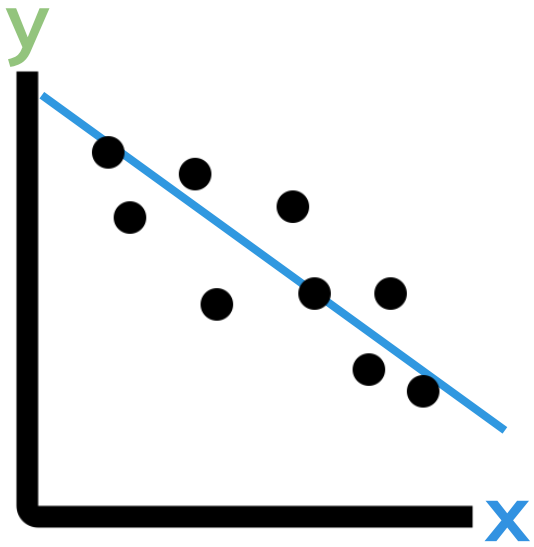
Classification

Regression

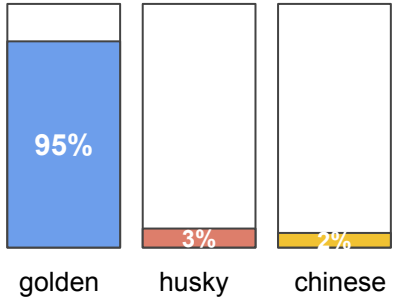


Classification

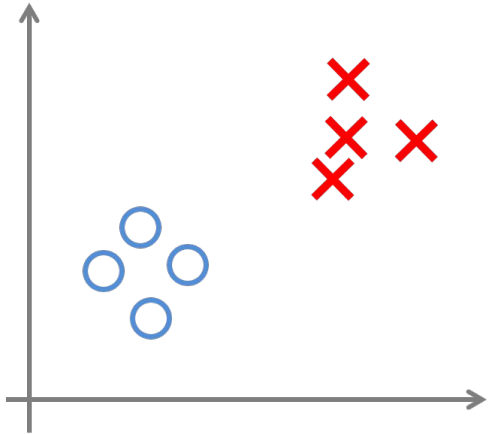
Regression



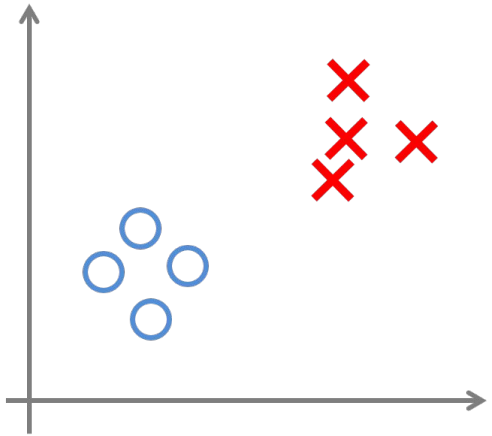
Classification



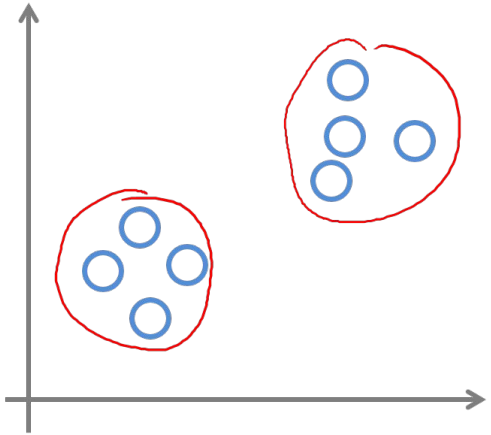
Supervised



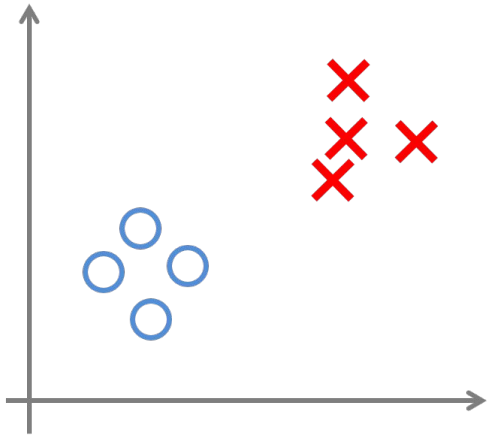
Supervised



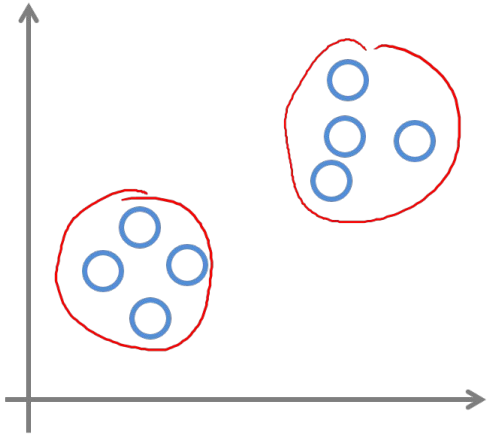
Unsupervised



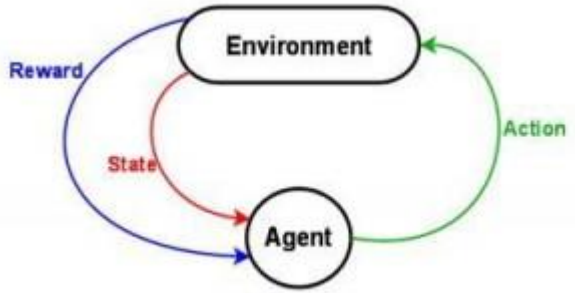
Supervised



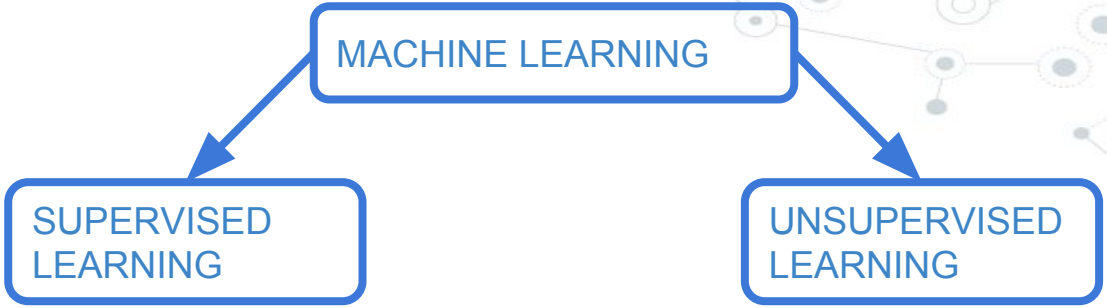
Unsupervised

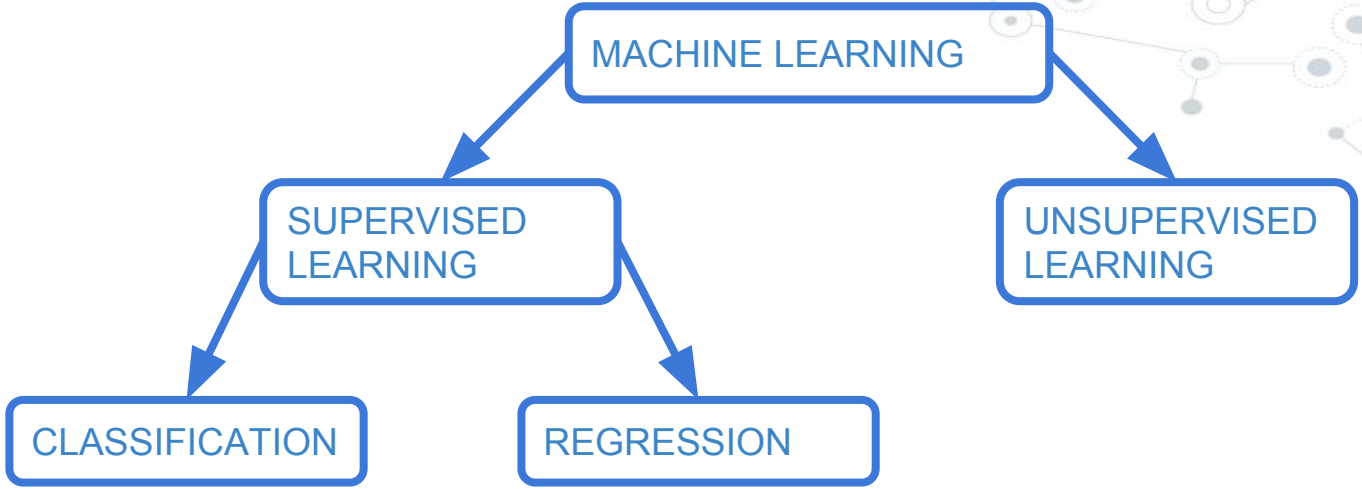


Reinforcement



MACHINE LEARNING


MACHINE LEARNING
SUPERVISED LEARNING
UNSUPERVISED LEARNING


MACHINE LEARNING
SUPERVISED LEARNING
UNSUPERVISED LEARNING
CLASSIFICATION
REGRESSION

MACHINE LEARNING

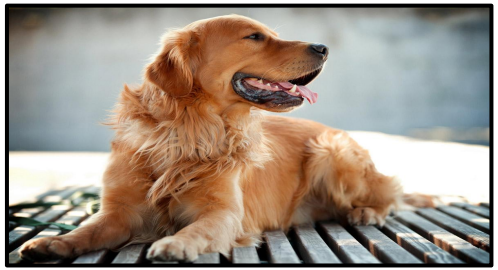
SUPERVISED LEARNING

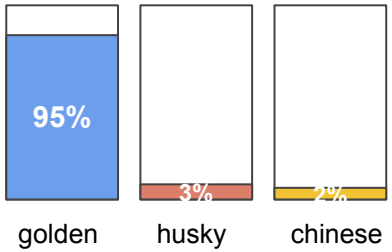
UNSUPERVISED LEARNING

CLASSIFICATION

REGRESSION

- Logistic regression
- Nearest neighbor
- Decision trees
- Support vector machines
- Neural networks





MACHINE LEARNING

SUPERVISED LEARNING

UNSUPERVISED LEARNING

CLASSIFICATION

REGRESSION

Logistic regression

Nearest neighbor

Decision trees

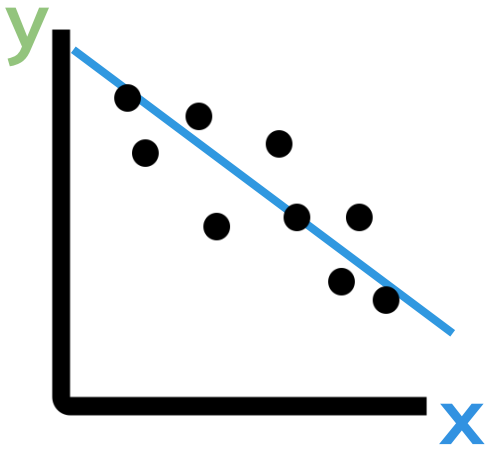
Support vector machines

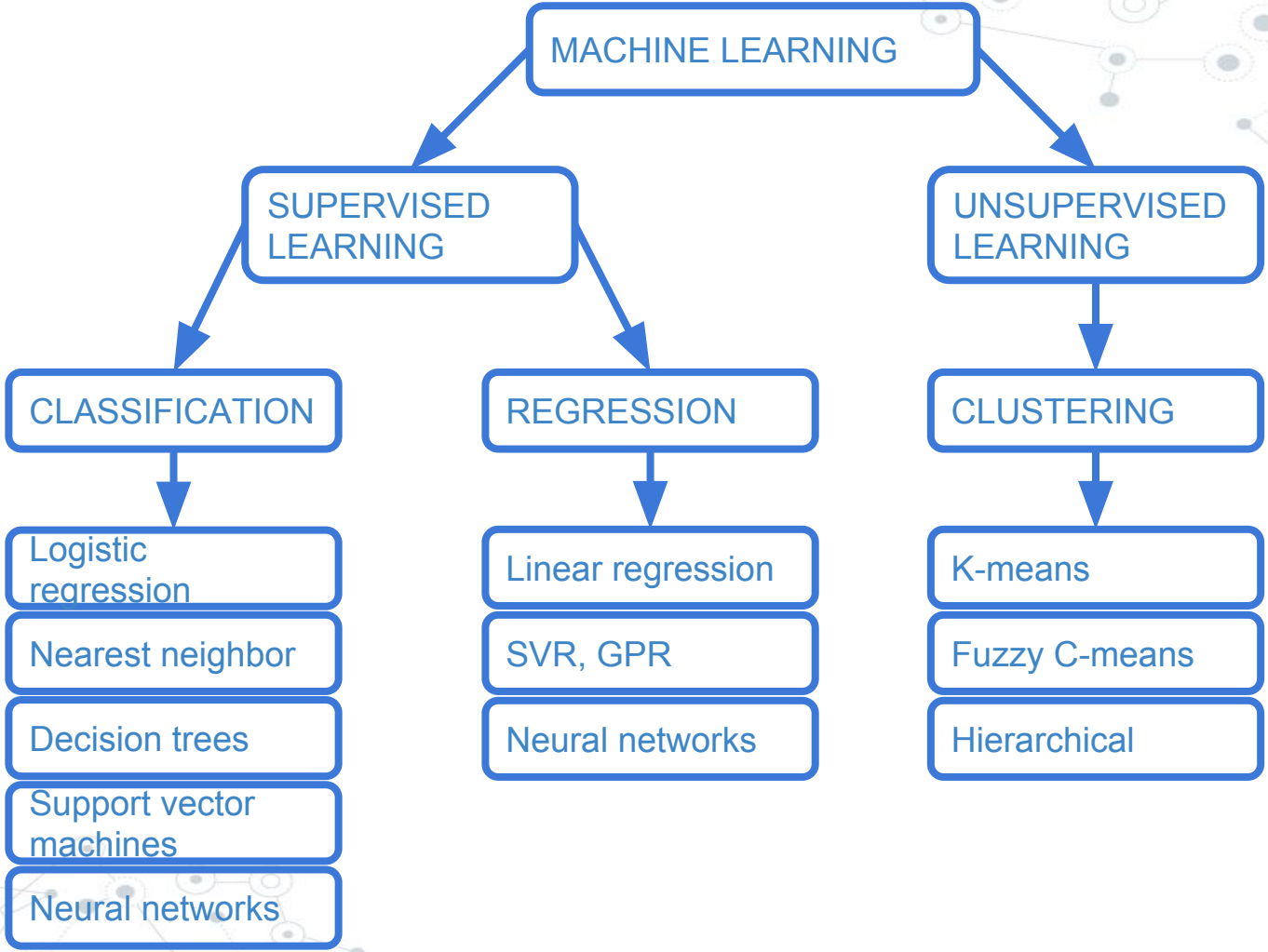
Neural networks

Linear regression

SVR, GPR

Neural networks




MACHINE LEARNING
SUPERVISED LEARNING
UNSUPERVISED LEARNING
CLASSIFICATION
REGRESSION
CLUSTERING
Logistic regression
Nearest neighbor
Decision trees
Support vector machines
Neural networks
Linear regression
SVR, GPR
Neural networks
K-means
Fuzzy C-means
Hierarchical

What makes it possible

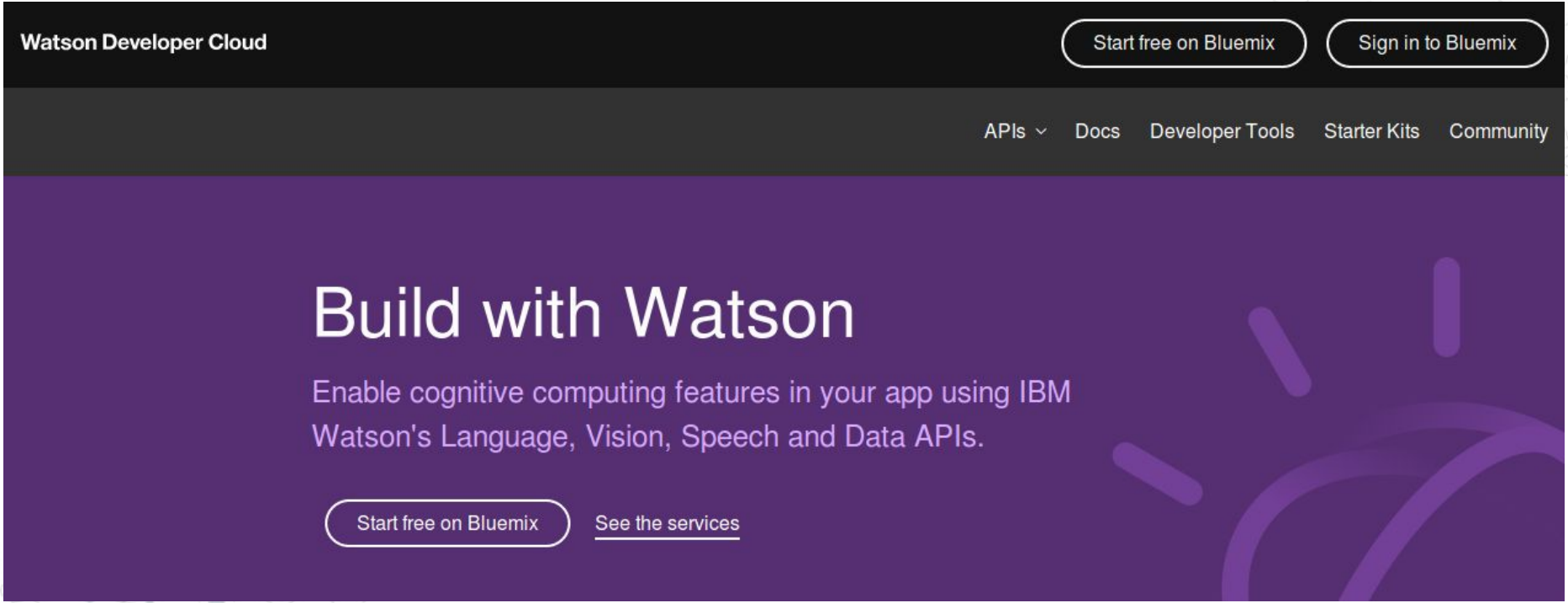

BIG
DATA
USE
ONE
SOFTWARE
USING
GARTNER
MASSIVELY
WORKING
BIOLOGICAL
HUNDREDS
DISK
SHARED
TIME
SYSTEMS
INCLUDE
NETWORKS
INFORMATION
RECORDS
SEARCH
PROCESS
WITHIN
TOOLS
SET
GENOMICS
PERFORMANCE
RELATIONAL
TENS
CAPACITY
PRACTITIONERS
ANALYTICS
BUSINESS
SAN
PARALLEL
SIZE
TOLERABLE
NEEDED
PETABYTES
INTERNET
TECHNOLOGIES
PROCESSING
LOGS
MAY
DISTRIBUTED
MANAGEMENT
CAPTURE
SOCIAL
LARGE
MANAGE
GROW
EVERY
LARGER
CASE
STORAGE
RESEARCH
DATABASES
COMPLEX
TERABYTES
CONTINUES
CITATION
COST
DIFFICULTY
TARGET
ABILITY
SETS
SENSOR
ARCHIVES
EXAMPLES
APPLIED
CURRENT
ELAPSED


BIG
DATA
ANALYTICS
STORAGE
TECHNOLOGIES
INFORMATION
SOFTWARE
USE
USING
GARTNER
MASSIVELY
BIOLOGICAL
HUNDREDS
DISK
SHARED
TIME
SYSTEMS
INCLUDE
NETWORKS
SEARCH
RECORDS
COST
CONTINUES
CITATION
COMPLEX
DATABASES
TERABYTES
RESEARCH
SETS
ABILITY
EXAMPLES
APPLIED
SENSOR
ARCHIVES
TARGET
DIFFICULTY
CURRENT
ELAPSED
SAN
PARALLEL
SIZE
NEEDED
TOLERABLE
PETABYTES
INTERNET
PROCESSING
LOGS
DISTRIBUTED
MANAGEMENT
SOCIAL
LARGE
EVERY
LARGER
CASE
CAPTURE
MANAGE
GROW
MAY
TENS
CAPACITY
PRACTITIONERS
WORKING
BUSINESS
MOVING
PERFORMANCE
ALSO
RELATIONAL
GENOMICS
SET
TOOLS
WITHIN
PROCESS
DEFINITION
ONE

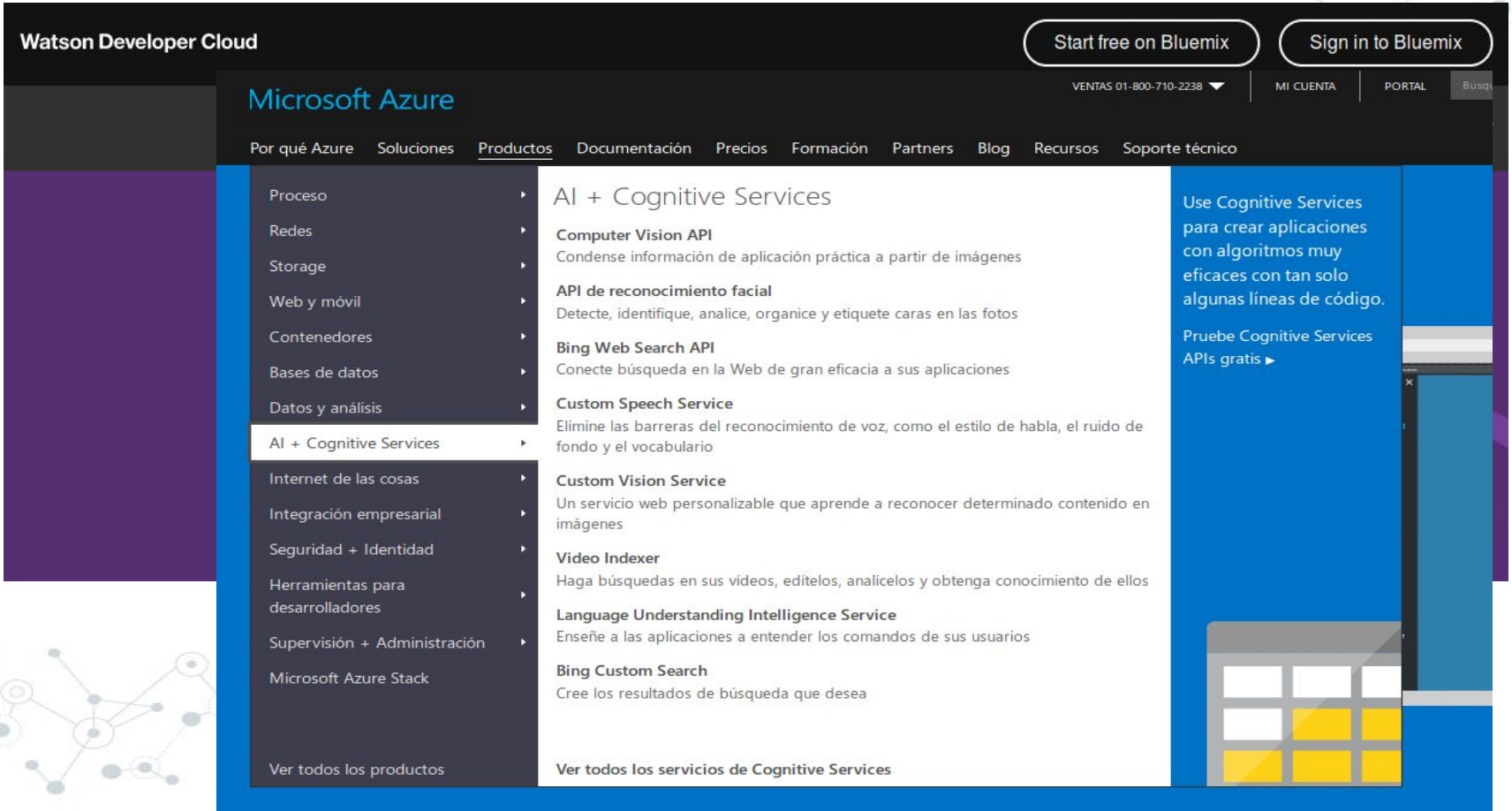

BIG
DATA
ANALYTICS
STORAGE
SOFTWARE
TECHNOLOGIES
INFORMATION
NETWORKS
SYSTEMS
INCLUDE
SHARED
TIME
DISK
HUNDREDS
BIOLOGICAL
MASSIVELY
GARTNER
USING
SEARCH
RECORDS
COMPLEX
DATABASES
RESEARCH
SETS
EXAMPLES
ABILITY
SENSOR
ARCHIVES
TERABYTES
CITATION
COST
CONTINUES
DIFFICULTY
TARGET
APPLIED
CURRENT
ELAPSED
SOCIAL
MANAGEMENT
LARGE
EVERY
LARGER
CASE
CAPTURE
MANAGE
GROW
DISTRIBUTED
INTERNET
PETABYTES
SIZE
NEEDED
TOLERABLE
PARALLEL
SAN
BUSINESS
PROCESSING
LOGS
PRACTITIONERS
CAPACITY
TENS
WITHIN
PROCESS
DEFINITION
TOOLS
USE
ONE
WORKING


NVIDIA
TESLA
intel
Core™ i7

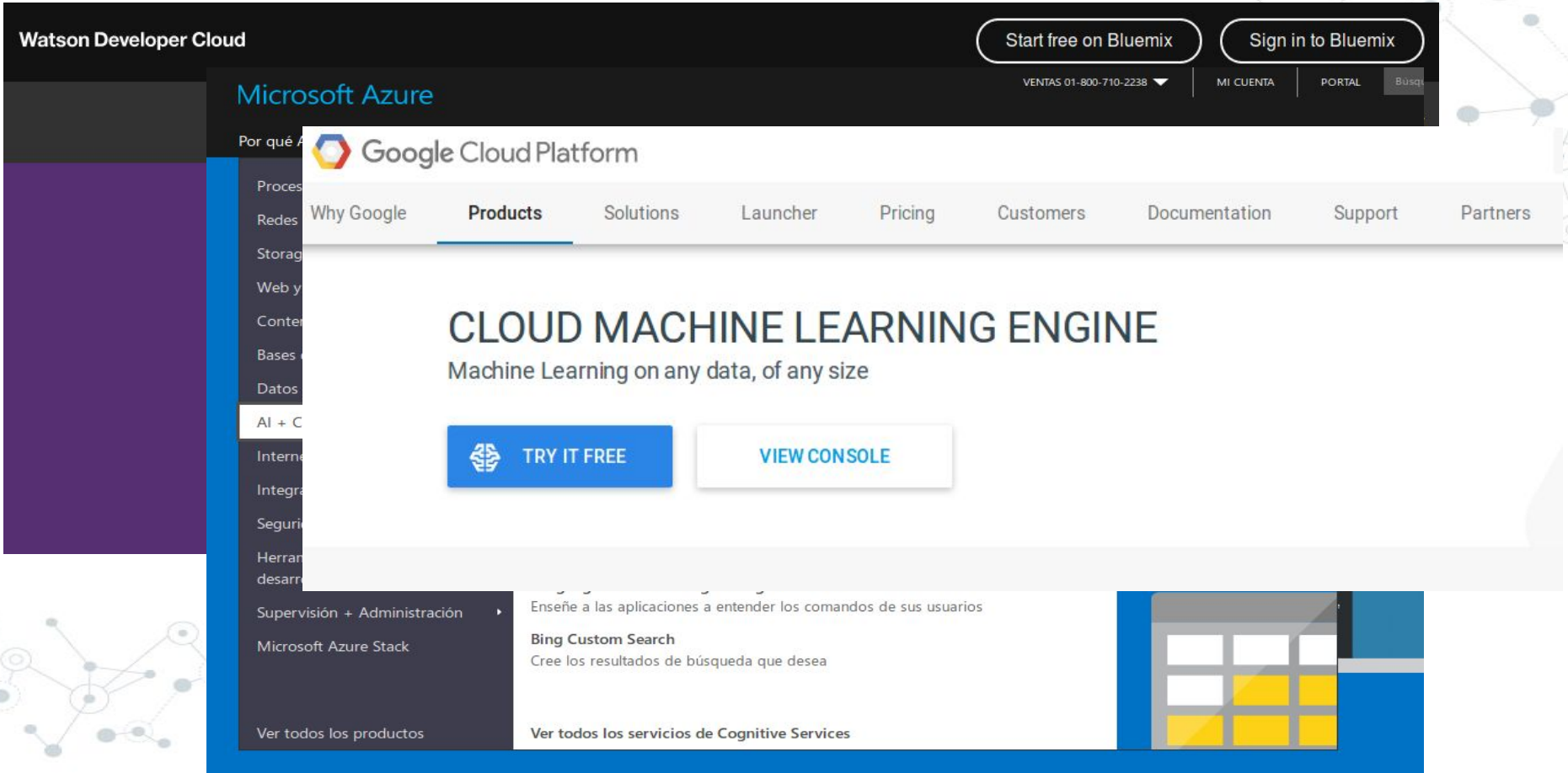
Cloud machine learning tools



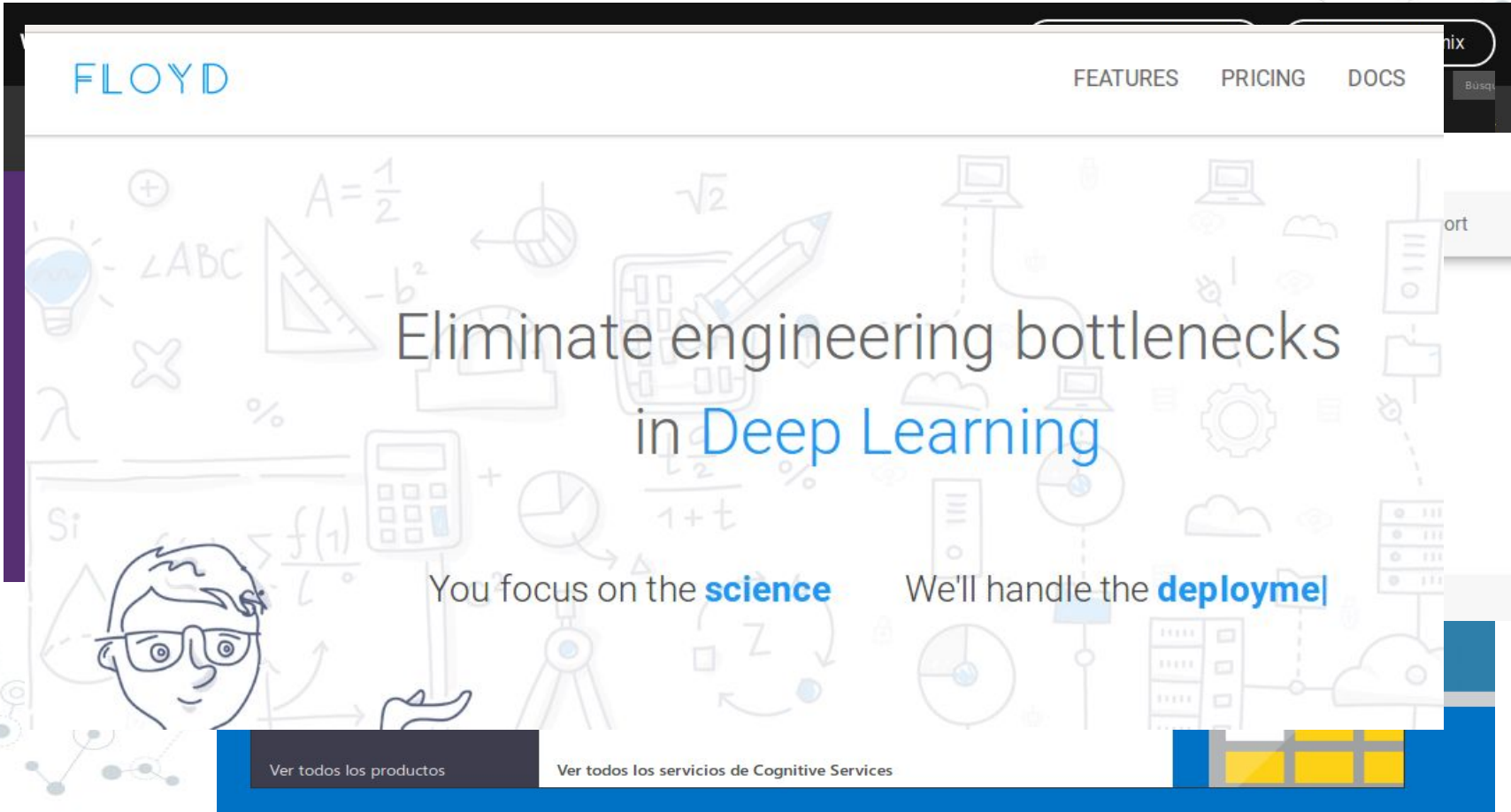
Cloud machine learning tools



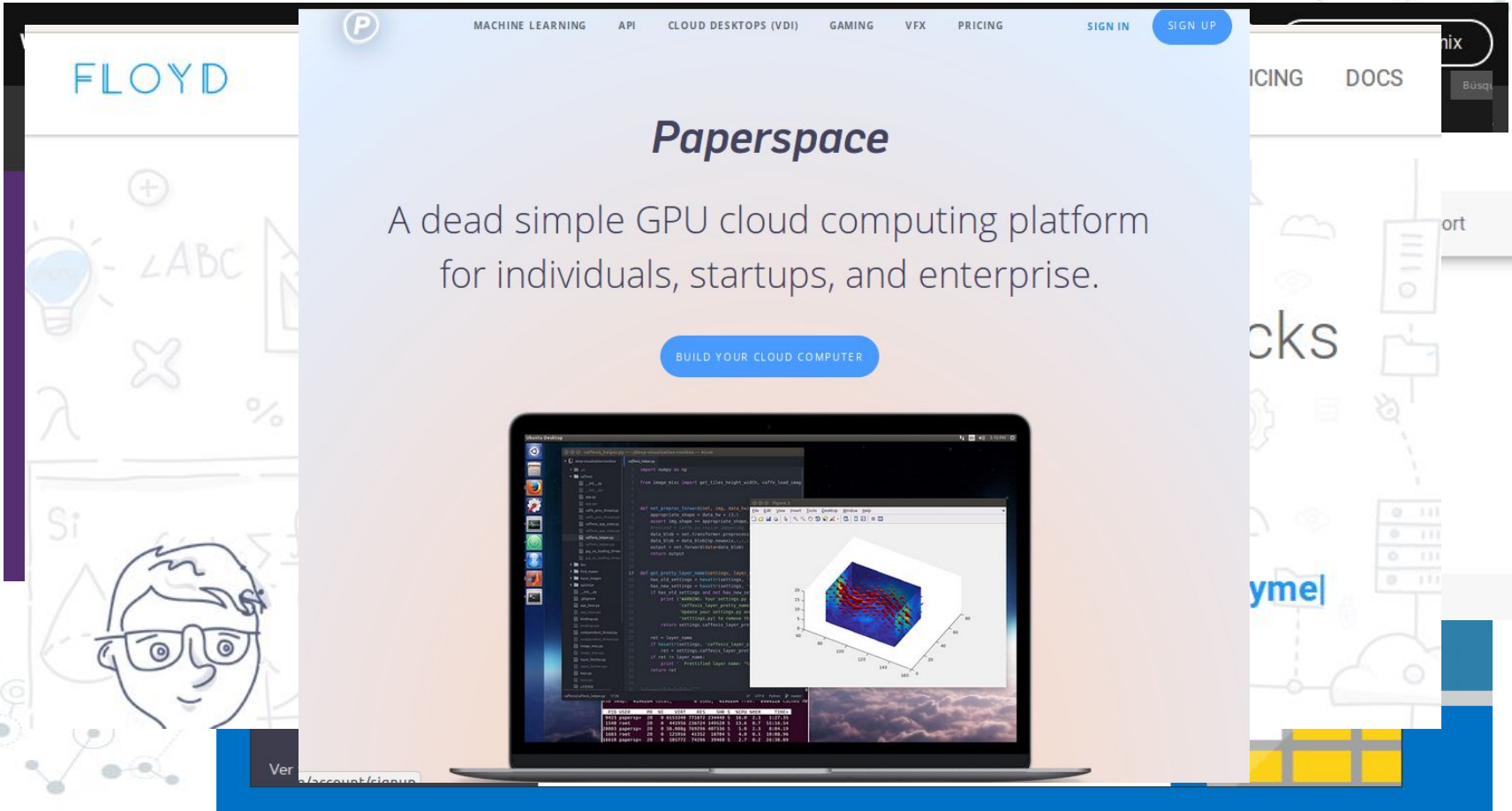
Cloud machine learning tools



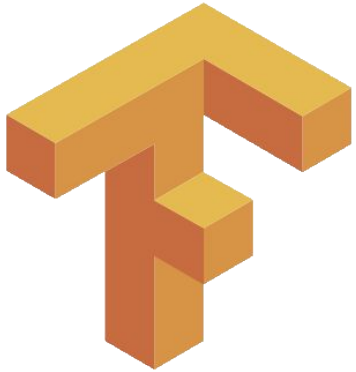
Cloud machine learning tools



Cloud machine learning tools



Additional resources



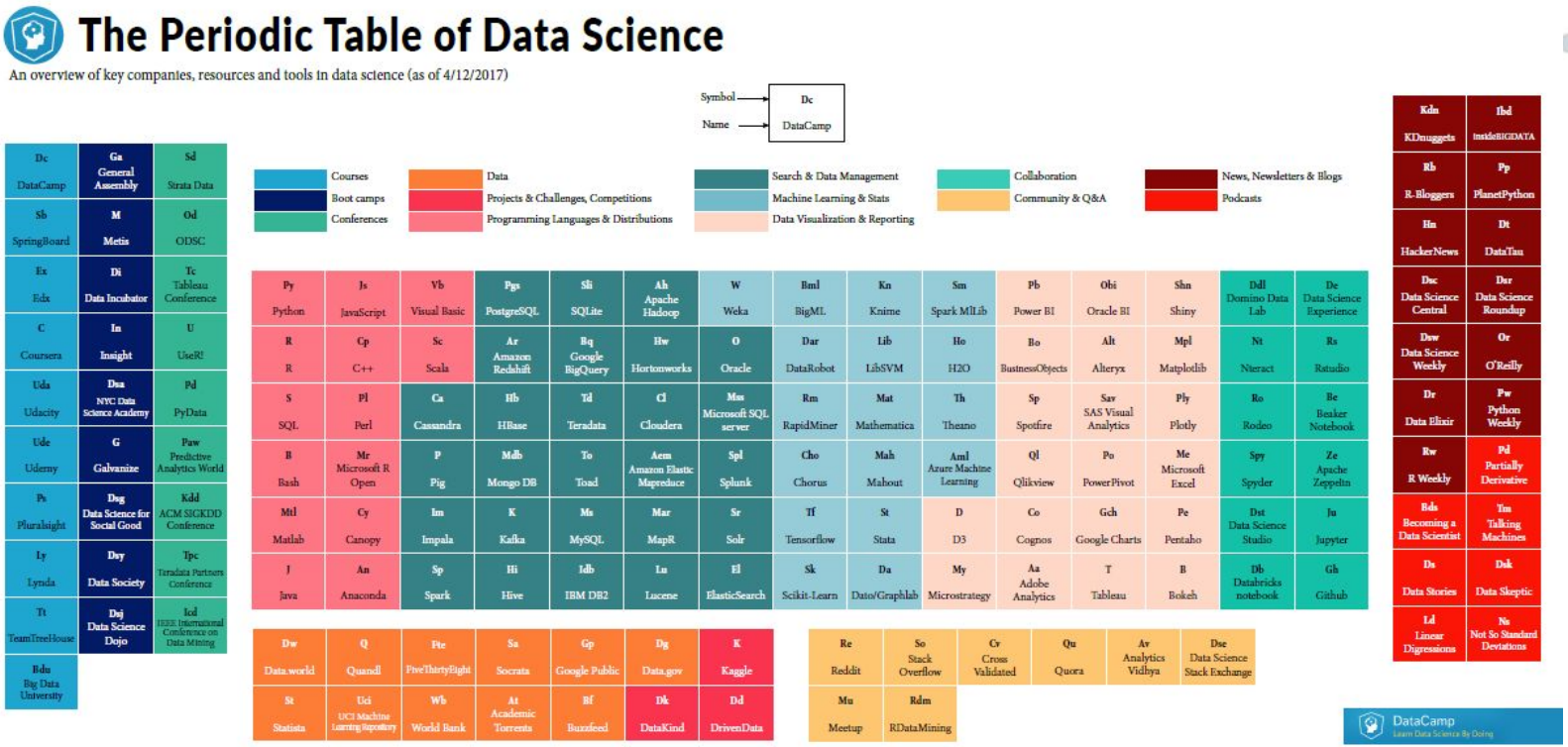

APACHE
Spark
TM


scikit
learn


Caffe

Important tools

Tools



Source: https://s3.amazonaws.com/assets.datacamp.com/blog_assets/Data-Science-Periodic-Table.pdf

Courses, bootcamps and conferences

Dc DataCamp	**Ga** General Assembly	**Sd** Strata Data
Sb SpringBoard	**M** Metis	**Od** ODSC
Ex Edx	**Di** Data Incubator	**Tc** Tableau Conference
C Coursera	**In** Insight	**U** UseR!
Uda Udacity	**Dsa** NYC Data Science Academy	**Pd** PyData
Ude Udemy	**G** Galvanize	**Paw** Predictive Analytics World
Ps Pluralsight	**Dsg** Data Science for Social Good	**Kdd** ACM SIGKDD Conference
Ly Lynda	**Dsy** Data Society	**Tpc** Teradata Partners Conference
Tt TeamTreeHouse	**Dsj** Data Science Dojo	**Icd** IEEE International Conference on Data Mining
Bdu Big Data University		

Programming languages, distributions and data management

Py Python	**Js** JavaScript	**Vb** Visual Basic	**Pgs** PostgreSQL	**Sli** SQLite	**Ah** Apache Hadoop	**W** Weka
R R	**Cp** C++	**Sc** Scala	**Ar** Amazon Redshift	**Bq** Google BigQuery	**Hw** Hortonworks	**O** Oracle
S SQL	**Pl** Perl	**Ca** Cassandra	**Hb** HBase	**Td** Teradata	**Cl** Cloudera	**Mss** Microsoft SQL server
B Bash	**Mr** Microsoft R Open	**P** Pig	**Mdb** Mongo DB	**To** Toad	**Aem** Amazon Elastic Mapreduce	**Spl** Splunk
Mtl Matlab	**Cy** Canopy	**Im** Impala	**K** Kafka	**Ms** MySQL	**Mar** MapR	**Sr** Solr
J Java	**An** Anaconda	**Sp** Spark	**Hi** Hive	**Idb** IBM DB2	**Lu** Lucene	**El** ElasticSearch

Machine learning frameworks and visualization

Bml BigML	**Kn** Knime	**Sm** Spark MlLib	**Pb** Power BI	**Obi** Oracle BI	**Shn** Shiny
Dar DataRobot	**Lib** LibSVM	**Ho** H2O	**Bo** BusinessObjects	**Alt** Alteryx	**Mpl** Matplotlib
Rm RapidMiner	**Mat** Mathematica	**Th** Theano	**Sp** Spotfire	**Sav** SAS Visual Analytics	**Ply** Plotly
Cho Chorus	**Mah** Mahout	**Aml** Azure Machine Learning	**Ql** Qlikview	**Po** PowerPivot	**Me** Microsoft Excel
Tf Tensorflow	**St** Stata	**D** D3	**Co** Cognos	**Gch** Google Charts	**Pe** Pentaho
Sk Scikit-Learn	**Da** Dato/Graphlab	**My** Microstrategy	**Aa** Adobe Analytics	**T** Tableau	**B** Bokeh

IDEs and collaborative tools

Ddl Domino Data Lab	**De** Data Science Experience
Nt Nteract	**Rs** Rstudio
Ro Rodeo	**Be** Beaker Notebook
Spy Spyder	**Ze** Apache Zeppelin
Dst Data Science Studio	**Ju** Jupyter
Db Databricks notebook	**Gh** Github

Data sources and challenges

Dw Data.world	Q Quandl	Fte FiveThirtyEight	Sa Socrata	Gp Google Public	Dg Data.gov	K Kaggle
St Statista	Uci UCI Machine Learning Repository	Wb World Bank	At Academic Torrents	Bf Buzzfeed	Dk DataKind	Dd DrivenData

Community

Re Reddit	**So** Stack Overflow	**Cv** Cross Validated	**Qu** Quora	**Av** Analytics Vidhya	**Dse** Data Science Stack Exchange
Mu Meetup	**Rdm** RDataMining				

Newsletters and podcasts

Kdn KDnuggets	**Ibd** insideBIGDATA
Rb R-Bloggers	**Pp** PlanetPython
Hn HackerNews	**Dt** DataTau
Dsc Data Science Central	**Dsr** Data Science Roundup
Dsw Data Science Weekly	**Or** O'Reilly

Dr Data Elixir	**Pw** Python Weekly
Rw R Weekly	**Pd** Partially Derivative
Bds Becoming a Data Scientist	**Tm** Talking Machines
Ds Data Stories	**Dsk** Data Skeptic
Ld Linear Digressions	**Ns** Not So Standard Deviations

Tools used in this session

- Jupyter
- Numpy
- Pandas
- Scikit-learn
- Matplotlib

Jupyter

Interactive environment to create documents, code, interactive widgets, graphs, texts and equations.

jupyter notebook Last Checkpoint: a minute ago (autosaved)

File Edit View Insert Cell Kernel Help Python 3

Markdown Cell Toolbar: None

Simple spectral analysis

An illustration of the Discrete Fourier Transform using windowing, to reveal the frequency content of a sound signal.

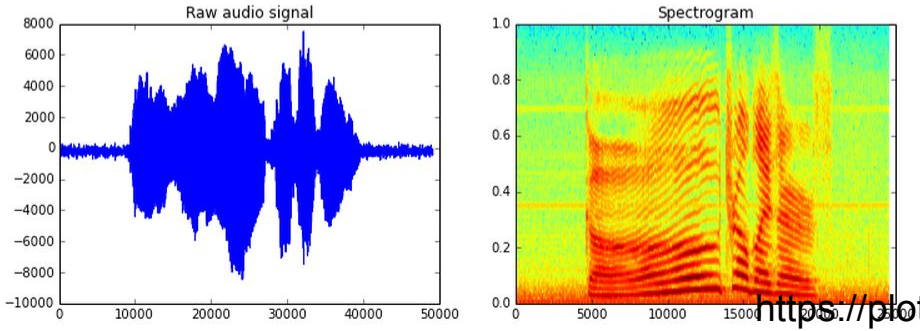
$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn} \qquad k = 0, \ldots, N-1$$

We begin by loading a datafile using SciPy's audio file support:

In [1]:
```
from scipy.io import wavfile
rate, x = wavfile.read('test_mono.wav')
```

And we can easily view its spectral structure using matplotlib's builtin `specgram` routine:

In [2]:
```
%matplotlib inline
from matplotlib import pyplot as plt
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 4))
ax1.plot(x); ax1.set_title('Raw audio signal')
ax2.specgram(x); ax2.set_title('Spectrogram');
```



https://plot.ly/python/ipython-vs-python/

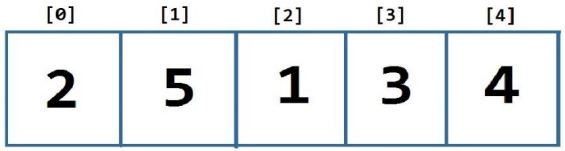
Tools used in this session

- Jupyter
- Numpy
- Pandas
- Scikit-learn
- Matplotlib

Numpy

- Array, matrix, vectorial operations.

- `Import numpy as np`



Tools used in this session

- Jupyter
- Numpy
- Pandas
- Scikit-learn
- Matplotlib

Pandas

- Data structure and tools for data analysis.
  - DataFrames
  - GroupBy, merge, join
  - Data reading and writing

- `Import pandas as pd`

Tools used in this session

- Jupyter
- Numpy
- Pandas
- Scikit-learn
- Matplotlib

Scikit-learn

- Machine learning library.
  - Linear regression
  - Logistic regression
  - Decision trees
  - Support vector machines
  - K-Nearest Neighbors
  - Other methods for regression and classification
  - Unsupervised methods

- `Import sklearn`

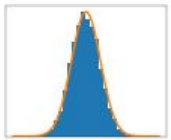
Tools used in this session

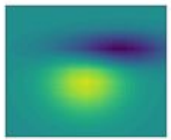
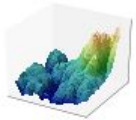
- Jupyter
- Numpy
- Pandas
- Scikit-learn
- Matplotlib

Matplotlib

- Graphics in Python (Frequency diagram, scatter-plot, 3-D graphics, etc.)

- `Import matplotlib`



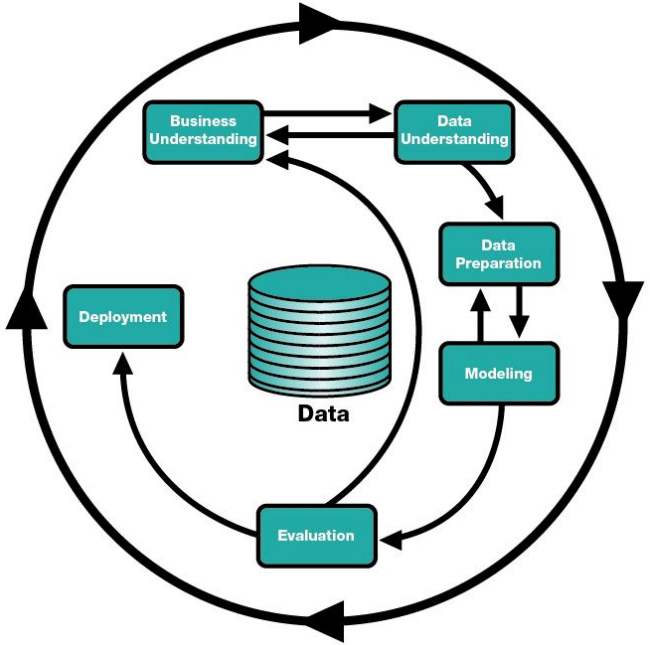
matplotlib

Machine Learning Pipeline

Pipeline


Data Extraction
Preprocessing
Modeling
Validation
Deployment

CRISP-DM Methodology



Source: http://crisp-dm.eu/

SEMMA

Data acquisition

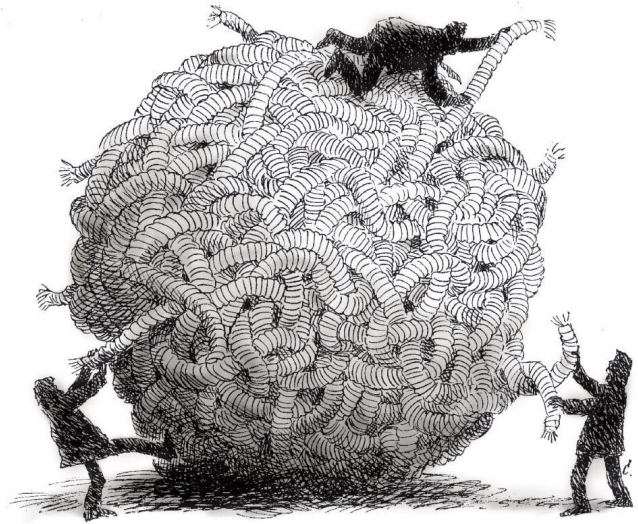
Raw material: Data

Data come in several formats:

- Web pages
- Video
- Audio
- CSV/TXT/Excel
- Databases

Source:
https://cdn.sisense.com/wp-content/uploads/messy-data.png

Preprocessing

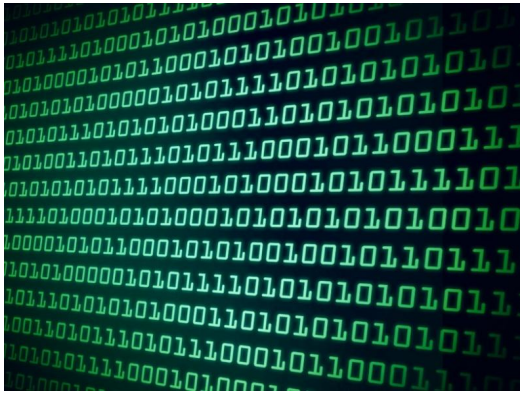


Get data ordered

- Scaling
- Missing values
- Feature extraction
- Feature transformation

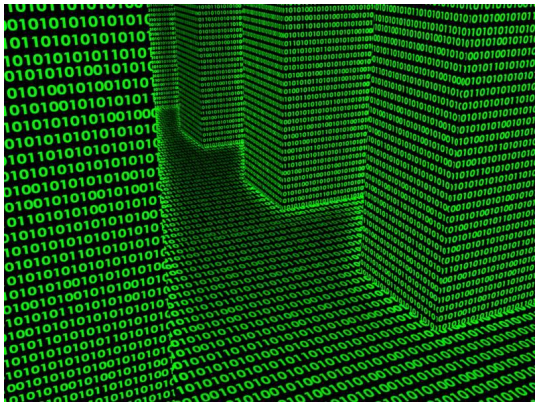
source:
https://ghotchkiss.files.wordpress.com/2015/04/messymarketing.jpeg

Modeling



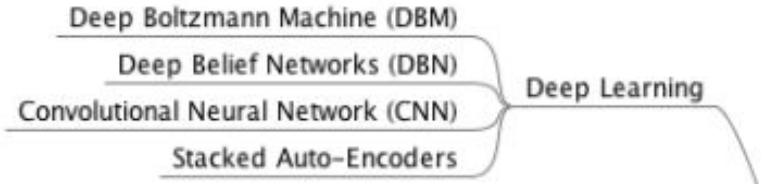
$$A = \pi r^2$$

$$y = W \cdot X + b$$

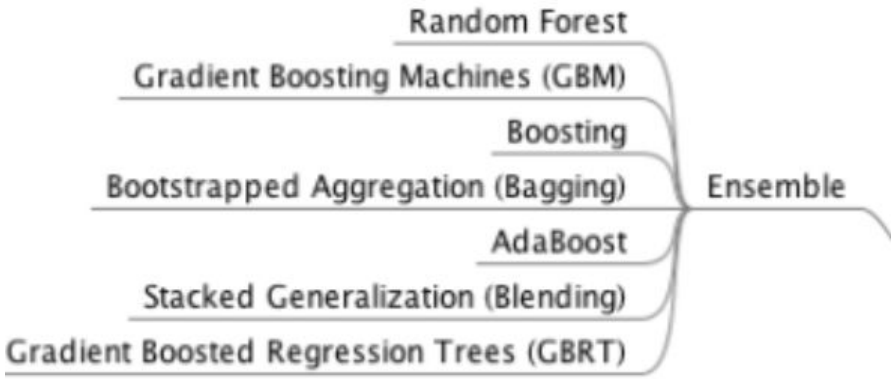


source: https://jixta.wordpress.com/2015/07/17/machine-learning-algorithms-mindmap/

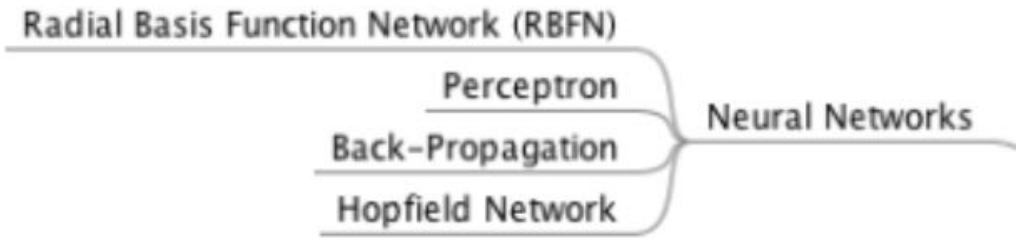
Deep Learning



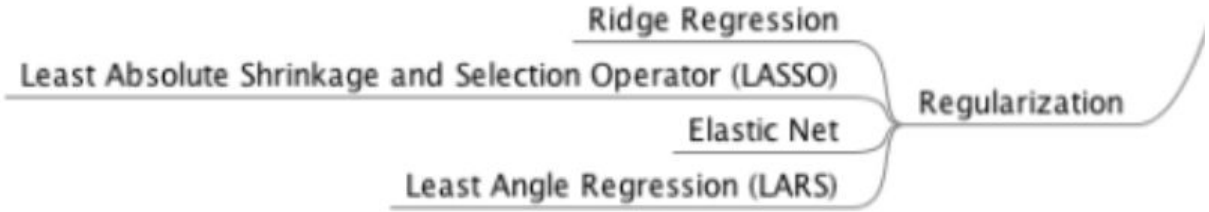
Ensemble


Random Forest
Gradient Boosting Machines (GBM)
Boosting
Bootstrapped Aggregation (Bagging)
AdaBoost
Stacked Generalization (Blending)
Gradient Boosted Regression Trees (GBRT)
Ensemble

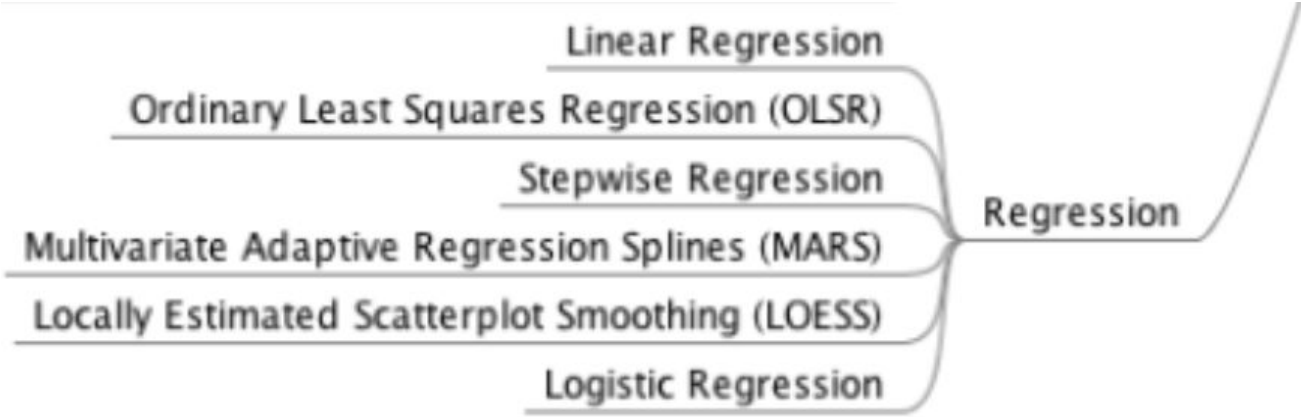
Neural Networks



Regularization



Regression

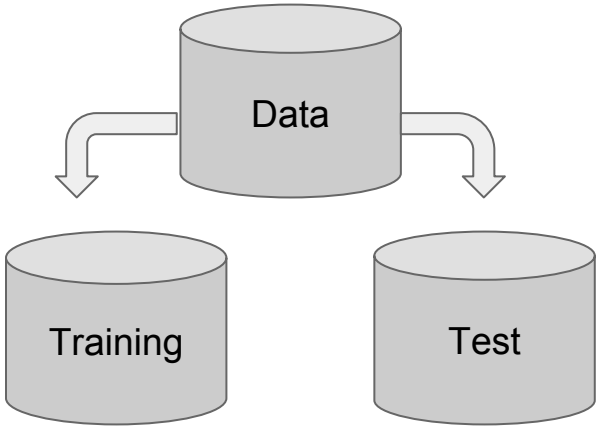

Linear Regression
Ordinary Least Squares Regression (OLSR)
Stepwise Regression
Multivariate Adaptive Regression Splines (MARS)
Locally Estimated Scatterplot Smoothing (LOESS)
Logistic Regression
Regression

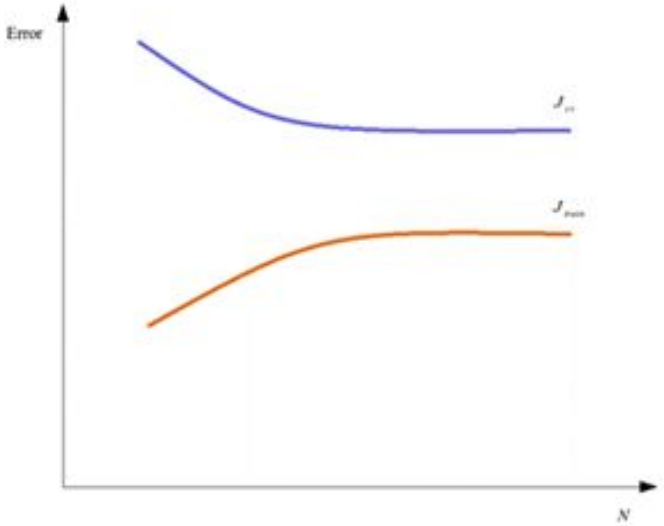
Validation

Find performance of the modeling stage

- Mean square error
- Precision/Recall
- F1-score
- ROC Curve

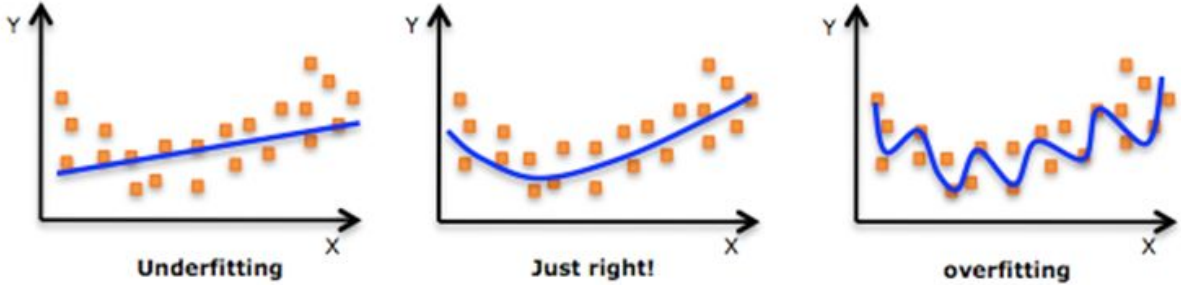
Generalization

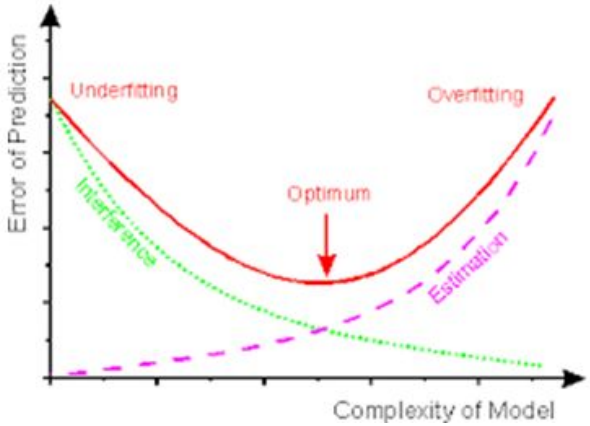




Generalization is the main goal of machine learning: predict on new, unseen data.

Overfitting and underfitting





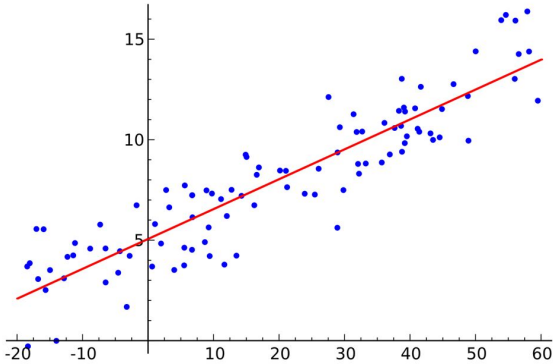
Deployment

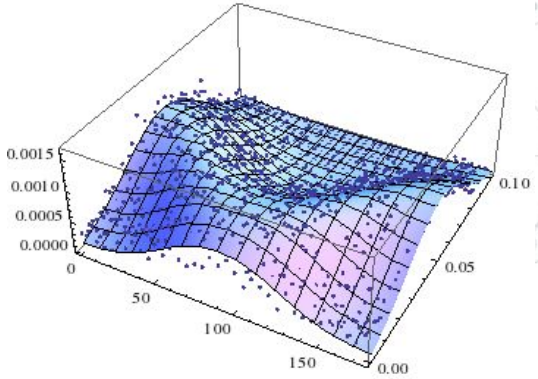
source: https://marketing4ecommerce.net/como-crear-un-dashboard-para-tu-ecommerce/

Dimension Reduction and Data Visualization
Demo

https://github.com/neuraldevs/ML-ND-CD/blob/master/PythonLibraries/Librer%C3%ADas.ipynb

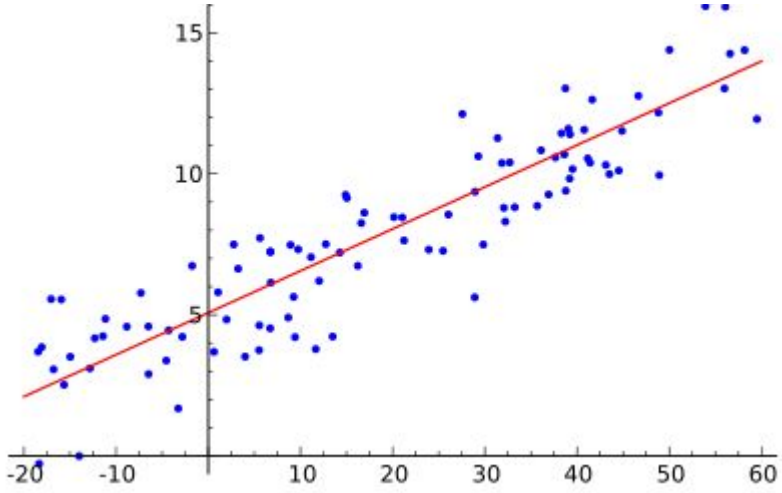
Regression


15
10
5
-20
-10
10
20
30
40
50
60


0.0015
0.0010
0.0005
0.0000
0
50
100
150
0.00
0.05
0.10

Regression

$$Y = XB + \varepsilon$$



Regression

(minimization: substract gradient term because we move towards local minima)

(the derivative of f with respect to a)

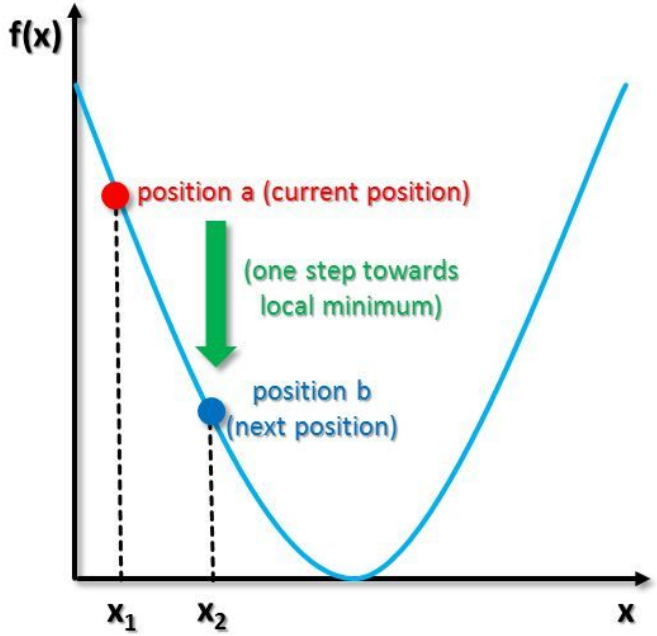
$$b = a - \gamma \nabla f(a)$$

(old position before the step)

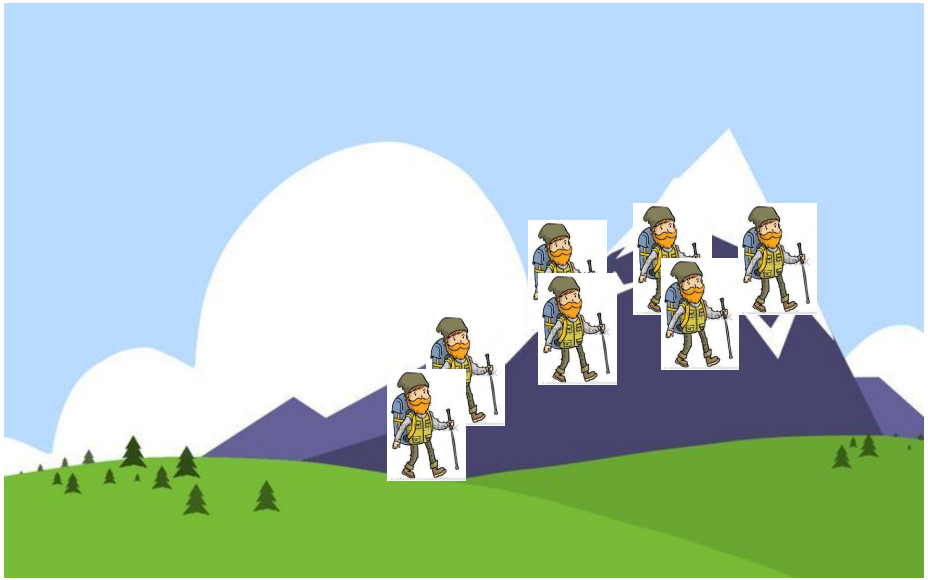
(gradient term is steepest ascent)

(new position after the step)

(weighting factor known as step-size, can change at every iteration, also called learning rate)



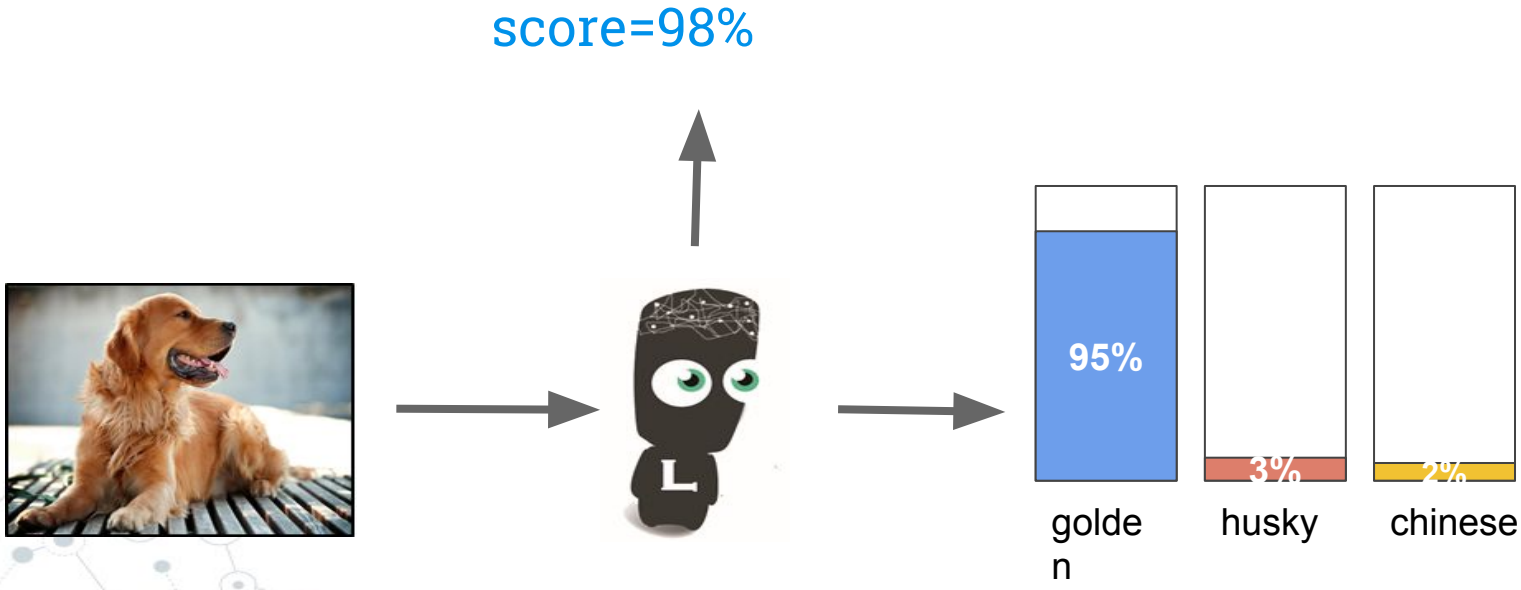
Gradient Descendent



Regression Demo

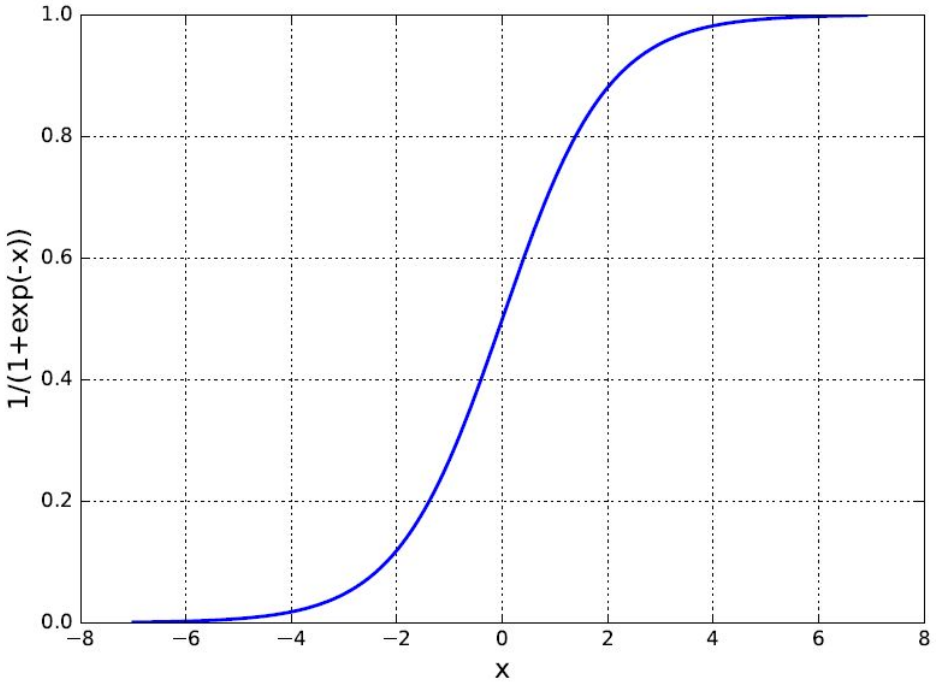
- https://github.com/neuraldevs/ML-ND-CD/blob/master/Regression/RegressionBasic.ipynb
- https://github.com/neuraldevs/ML-ND-CD/blob/master/Regression/Regression.ipynb

Supervised classification



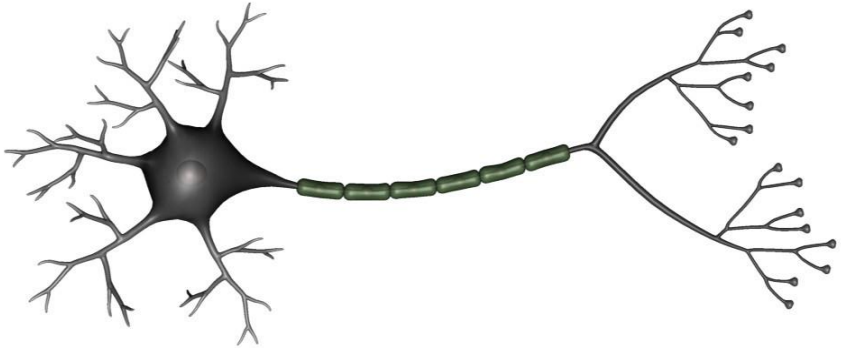
Methods

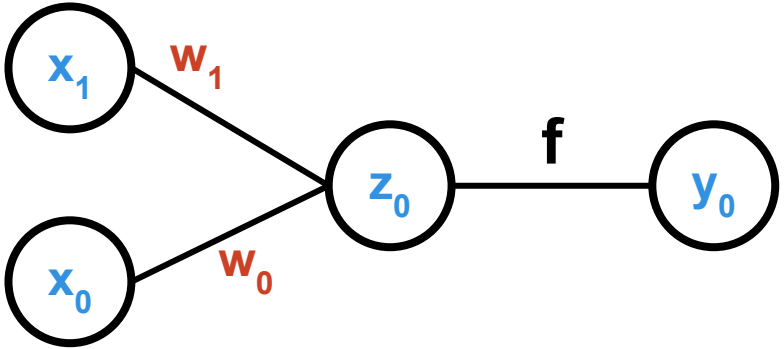
- Logistic regression



Methods

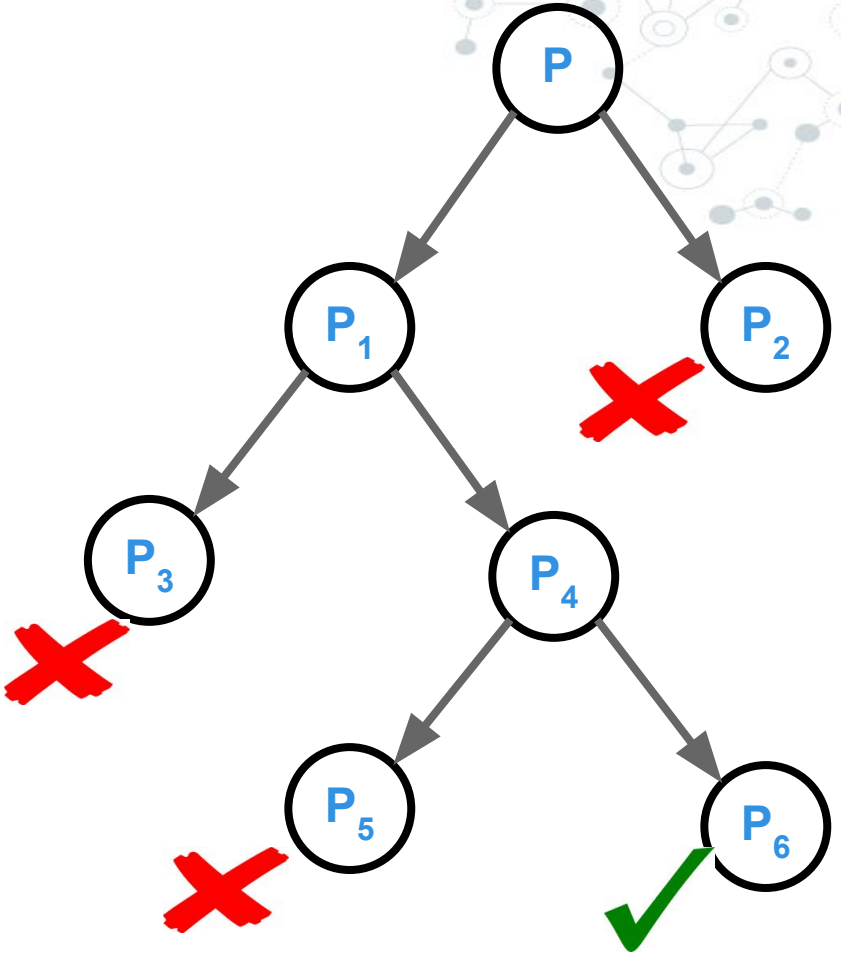
- Neural networks





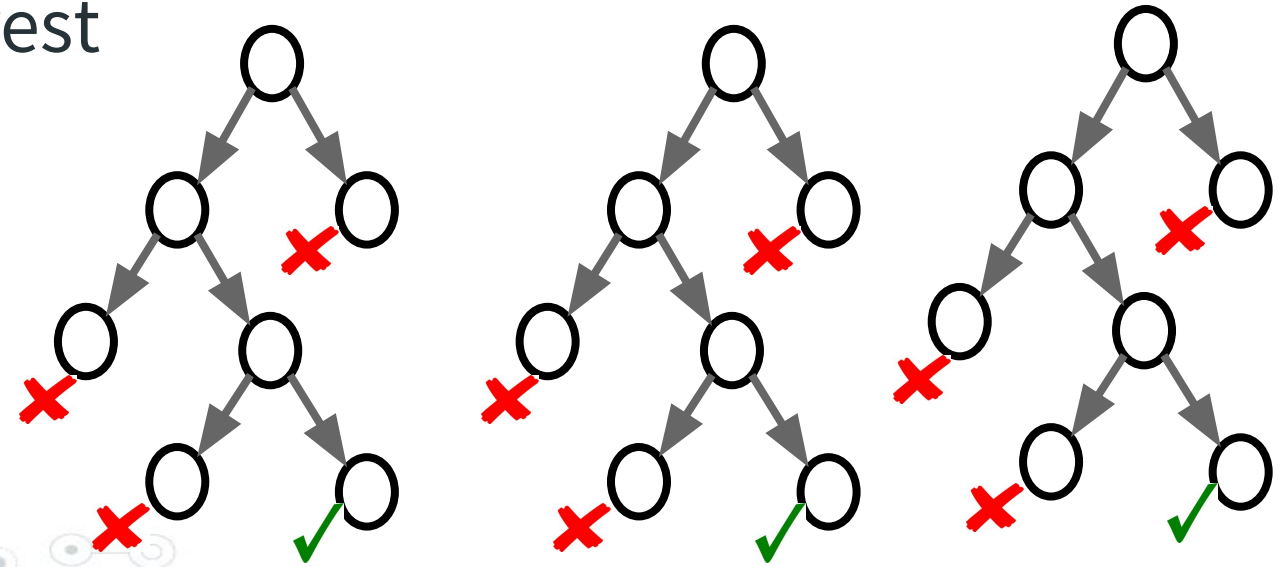
Methods

- Decision trees



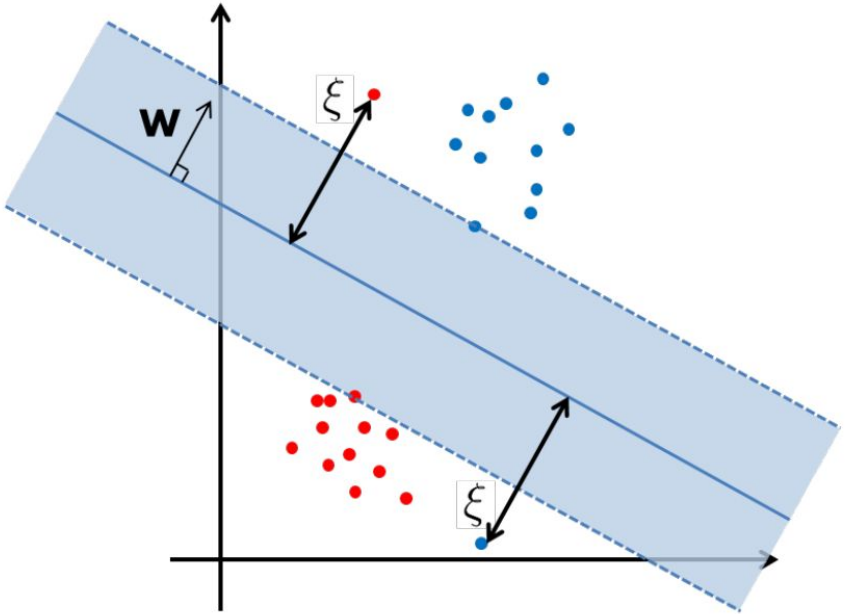
Methods

- Random Forest



Methods

- Support vector machines



Logistic regression

$$W \cdot X + b = y$$

Logistic regression

$$W \cdot X + b = y$$

Logistic regression

$$W \cdot X + b = y$$

golden

chinese

Logistic regression

$$W \cdot X + b = y$$

golden

chinese

Weights

bias

Logistic regression

$$W \cdot X + b = y = \begin{bmatrix} 2.0 \\ 0.5 \end{bmatrix}$$

golden

chinese

Logistic regression

$$W \cdot X + b = y = \begin{bmatrix} 2.0 \\ 0.5 \end{bmatrix}$$

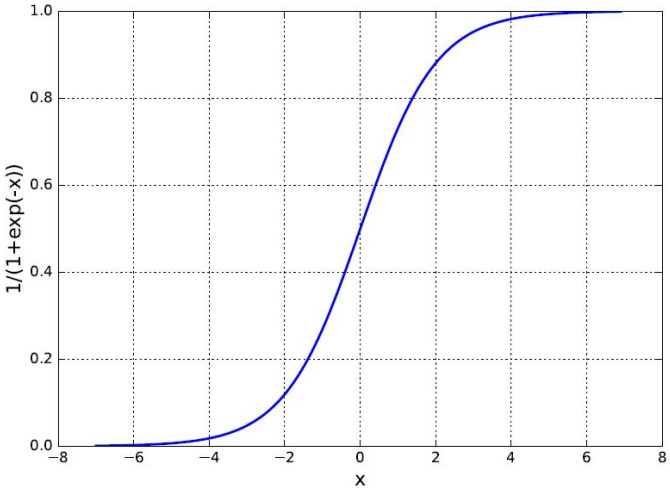
golden ✓
P(X)=0.99

chinese ✗
P(X)=0.05

Logistic regression

$$y = \begin{bmatrix} 2.0 \\ 0.1 \end{bmatrix}$$

$$p(y) = \frac{1}{1+e^{(-y)}}$$



$$p = \begin{bmatrix} 0.99 \\ 0.05 \end{bmatrix}$$

Logistic regression

+ Easy to train
+ Free adjustable hyper-parameters

- Could be very simple
- Only for linearly-separable classes

How to find W and b?

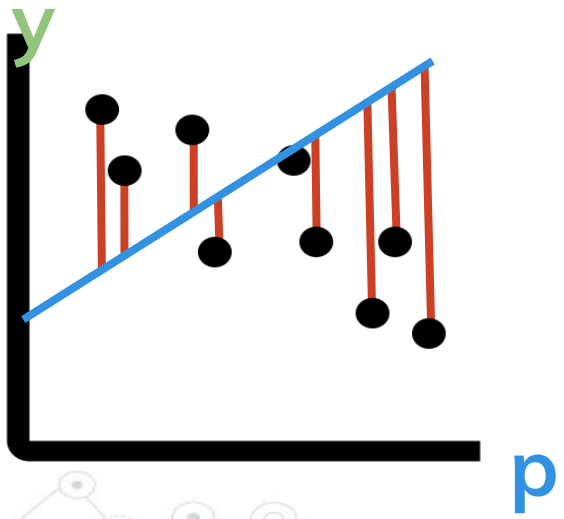
Answer:

Gradient descendent algorithm

1. p = f(x) #logistic function
2. e = E(p, y; θ) #get error
3. $\Delta = \nabla_{\theta} e$ #error's gradient / derivative
4. $\theta := \theta - \alpha \Delta$ #lower error moving against gradient
5. repeat

How to find W and b?

Error function: Mean square error



$$E = \frac{\sum (p - y)^2}{N}$$

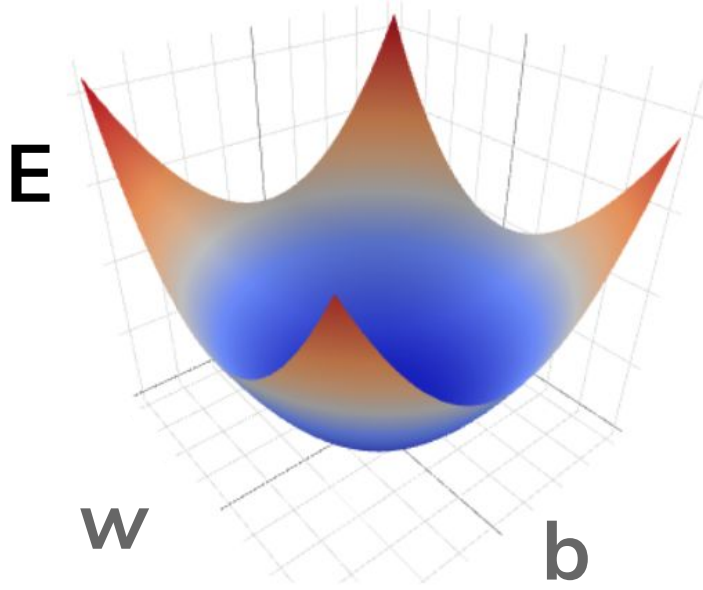
How to find W and b?

Error function 2:

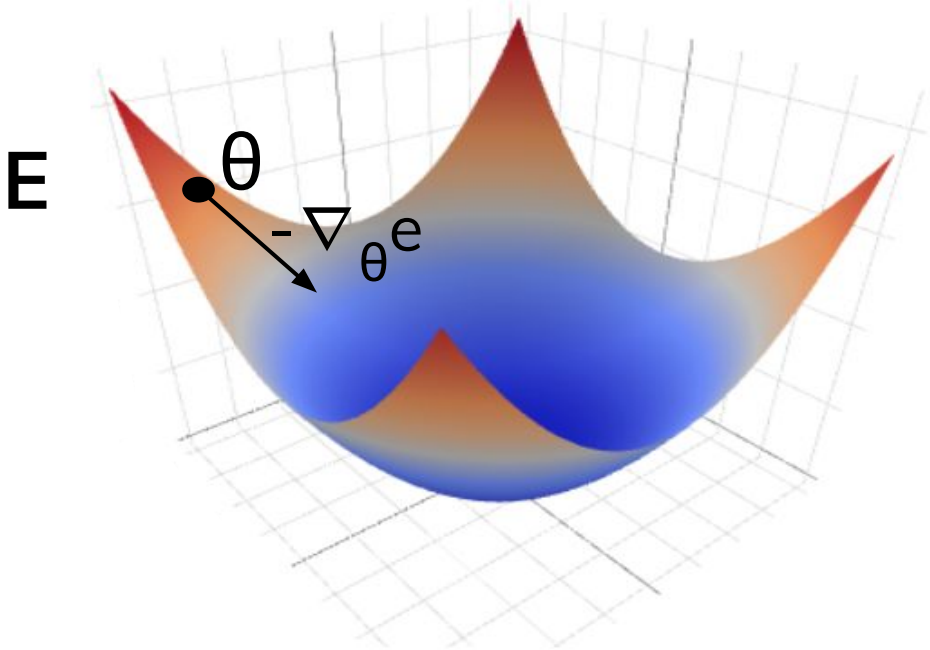
cross-entropy

Distance between two probability vectors

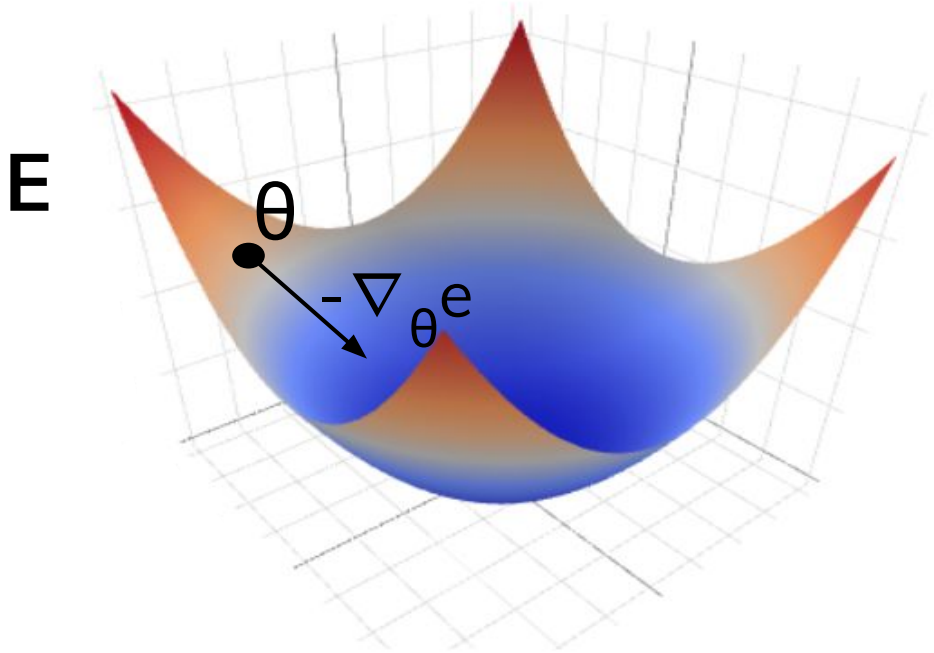
$$E = \frac{\sum (y \log(p))}{N}$$



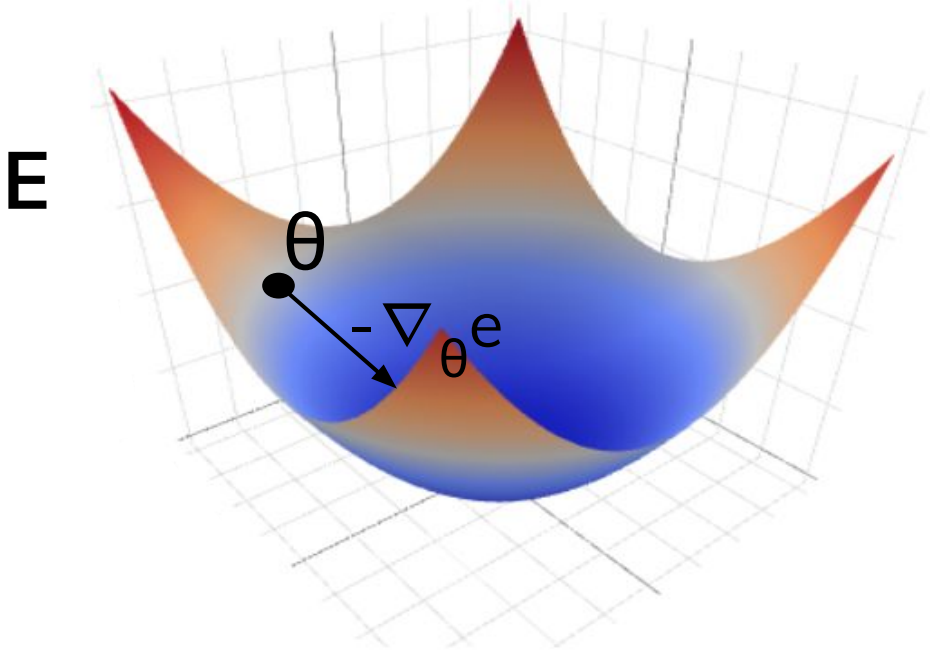
How to find $\boldsymbol{W}$ and $\boldsymbol{b}$?



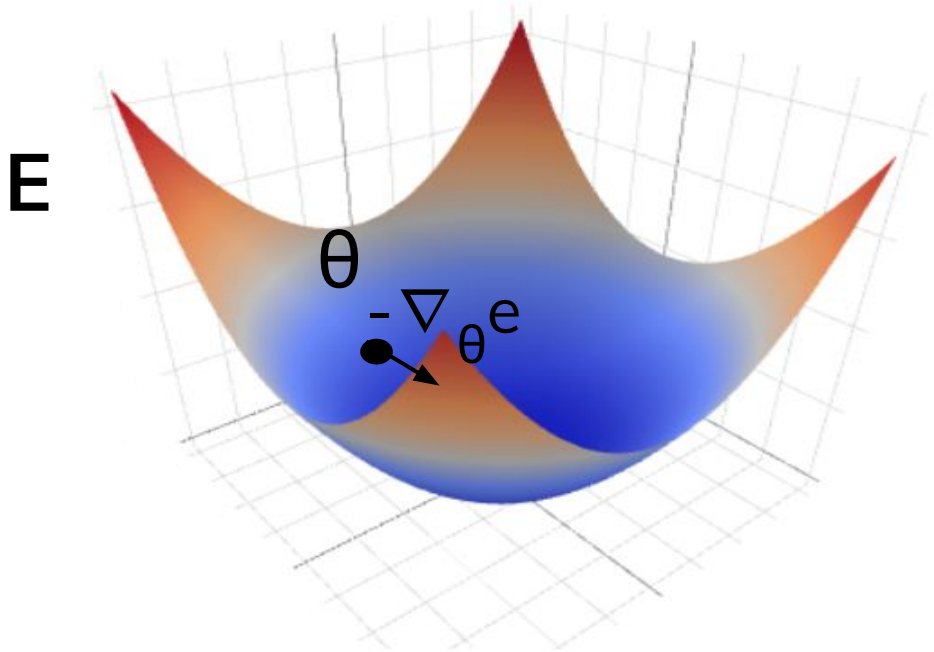
How to find W and b?



How to find $\boldsymbol{W}$ and $\boldsymbol{b}$?

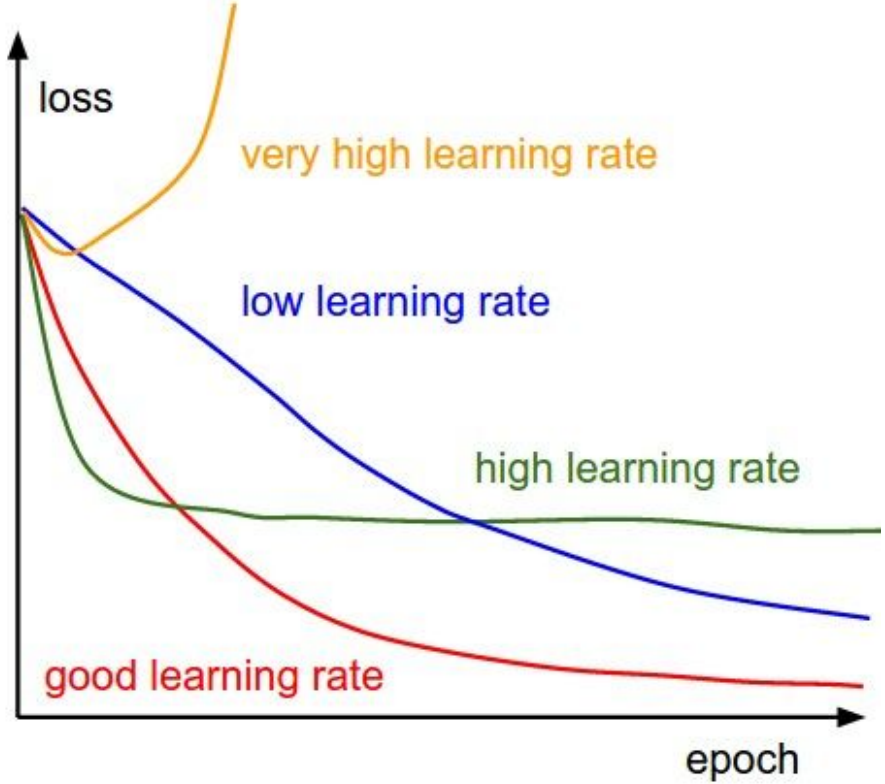


How to find W and b?



And so on...

How to find $\boldsymbol{W}$ and $\boldsymbol{b}$?

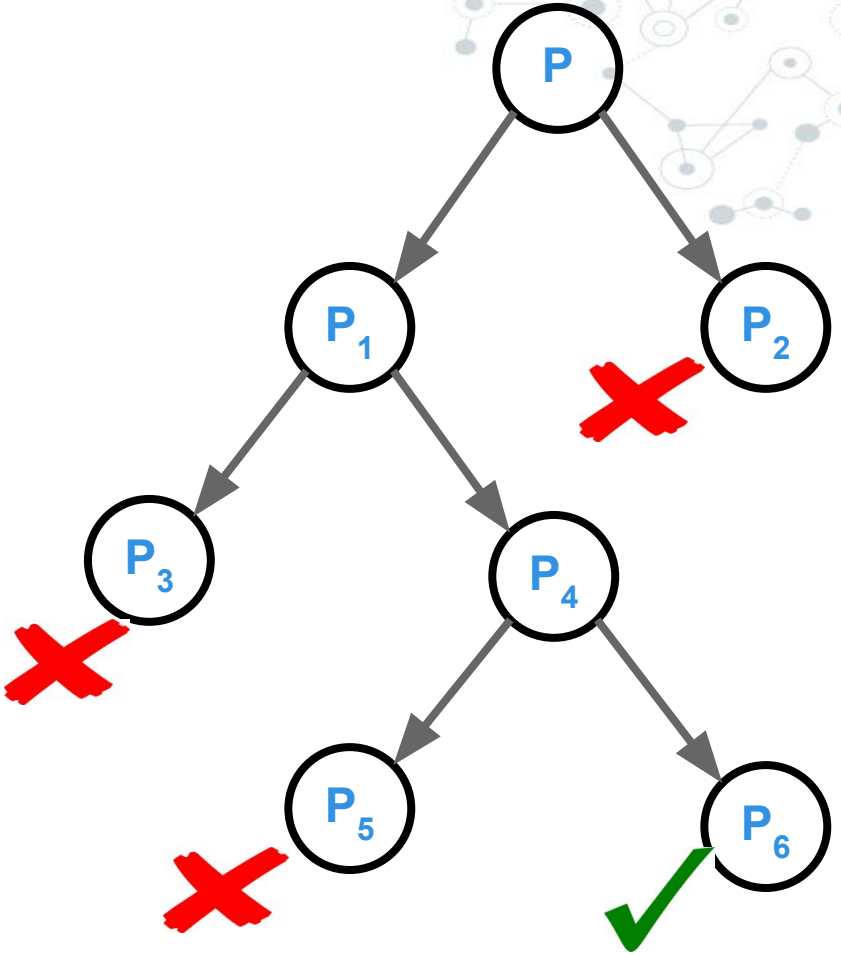


Logistic regression
Demo

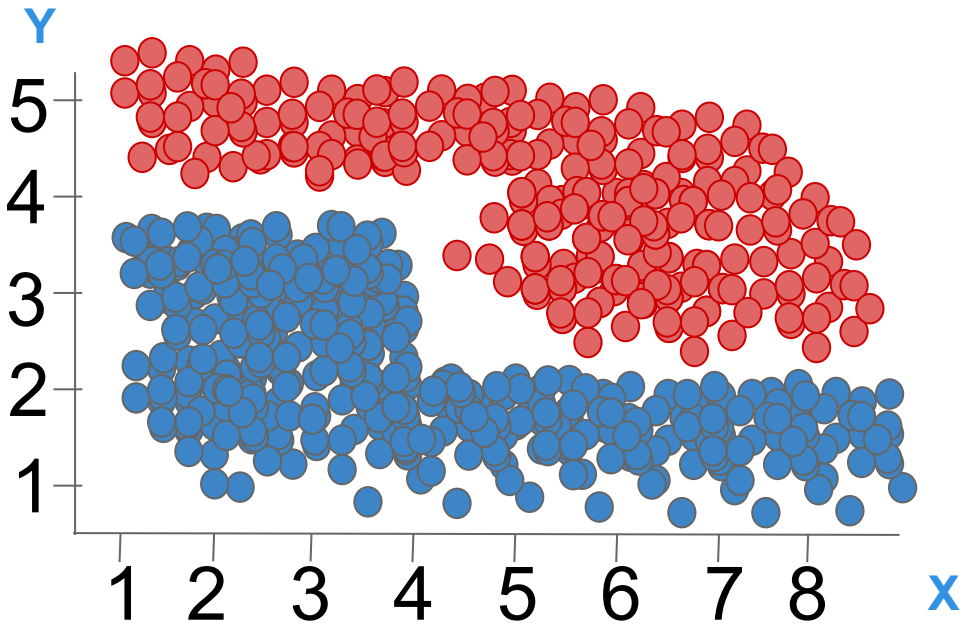
https://github.com/neuraldevs/ML-ND-CD/tree/master/cancer

Methods

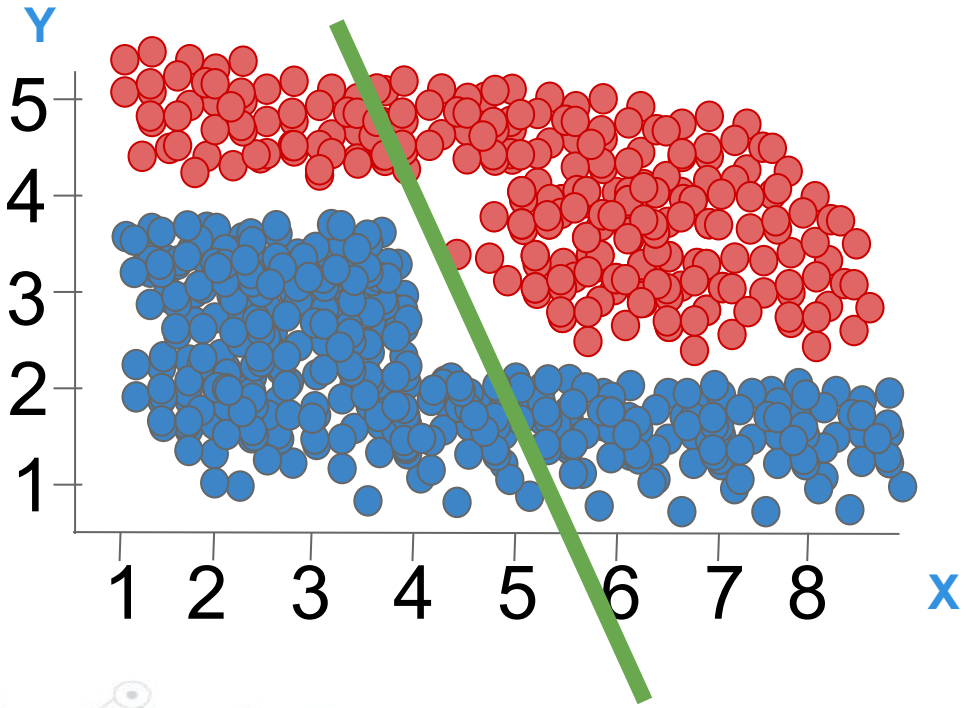
- Decision trees



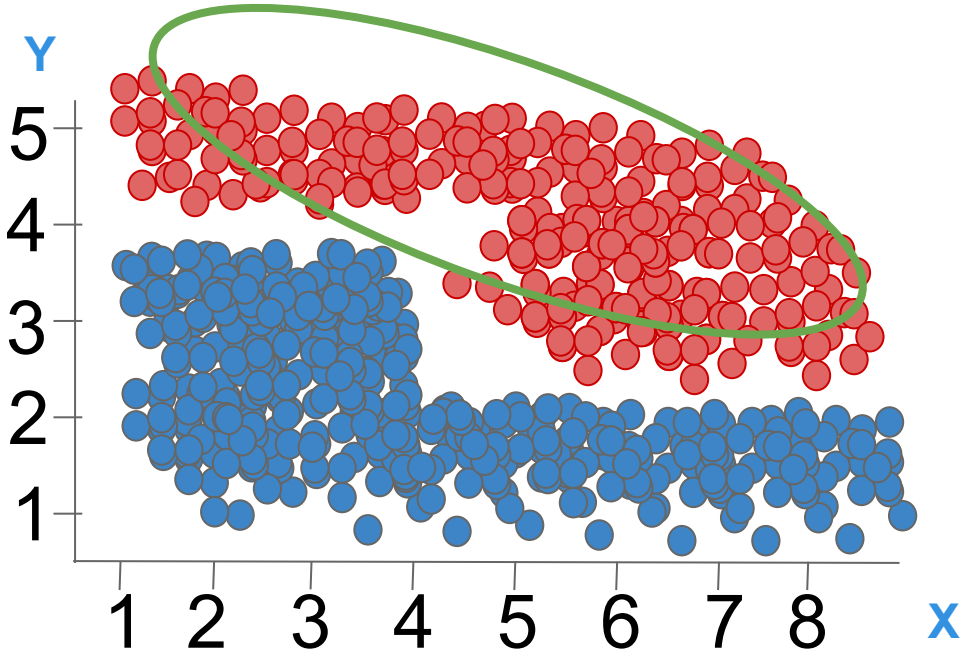
Decision trees



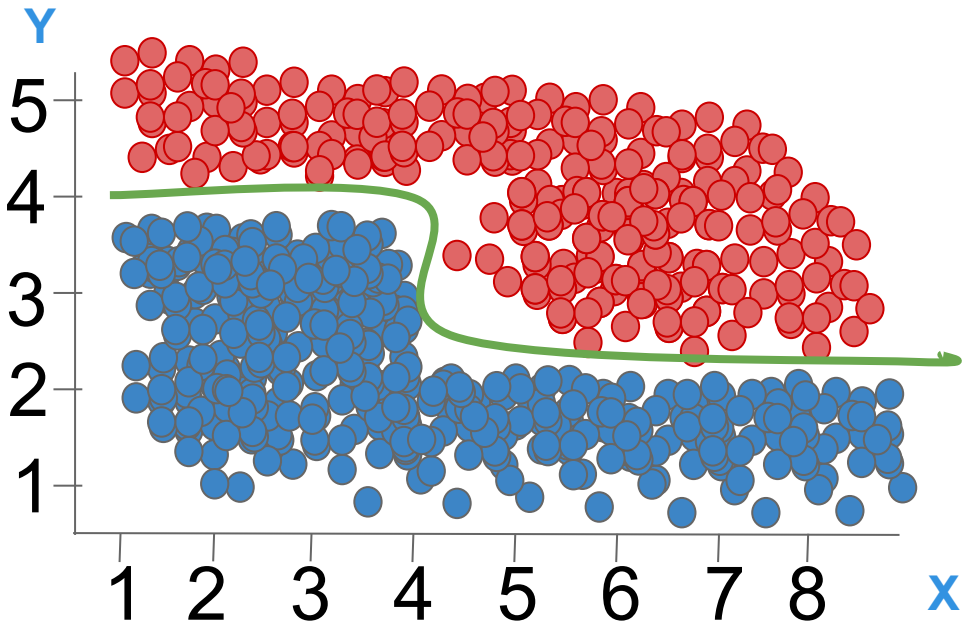
Decision trees



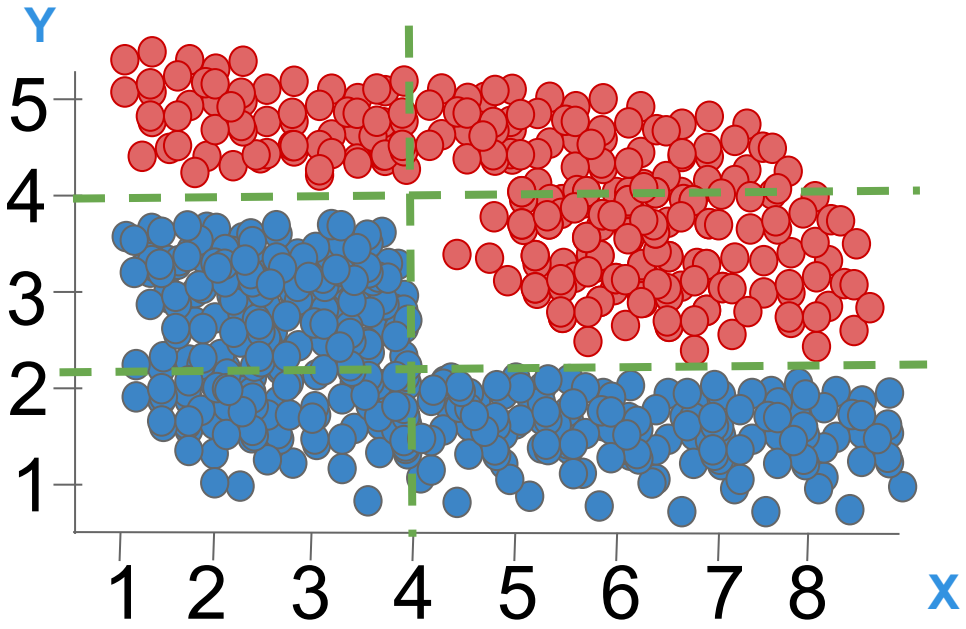
Decision trees

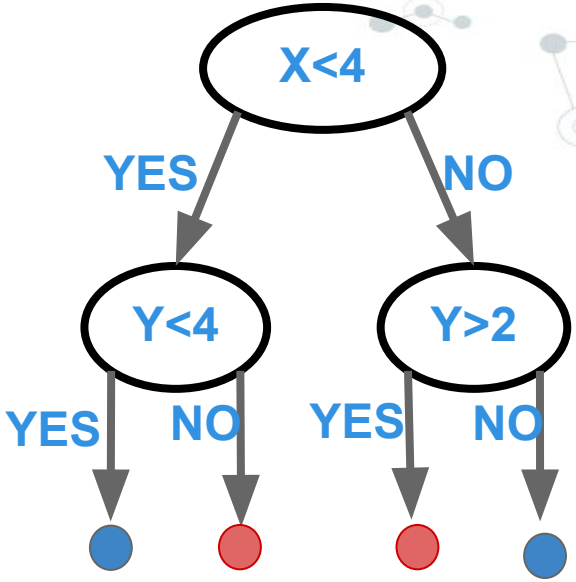


Decision trees



Decision trees





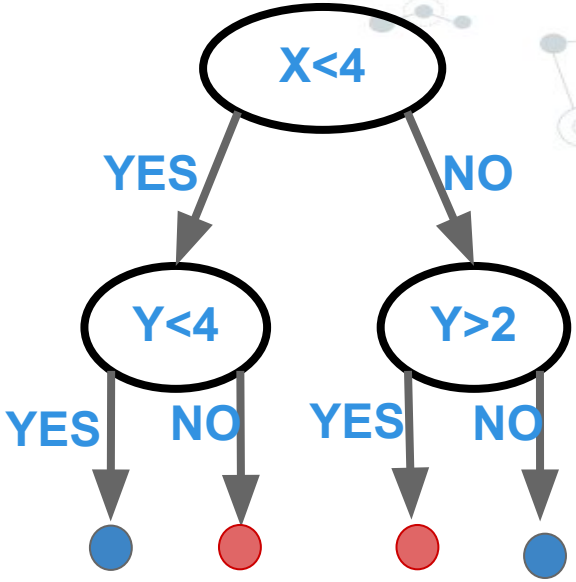
Decision trees

Goal:

- Achieve perfect classification with minimal number of decisions

Advantages:

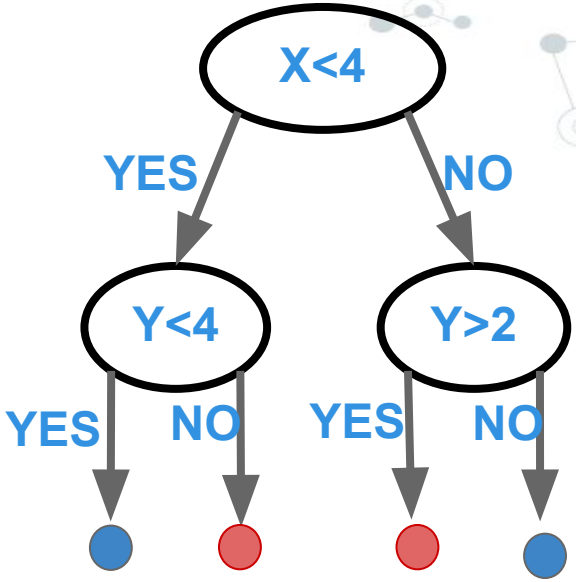
- Simple to understand
- Requires little data preparation
- Trees can be visualized
- Only one adjustable hyper-parameter (number of leafs)



Decision trees

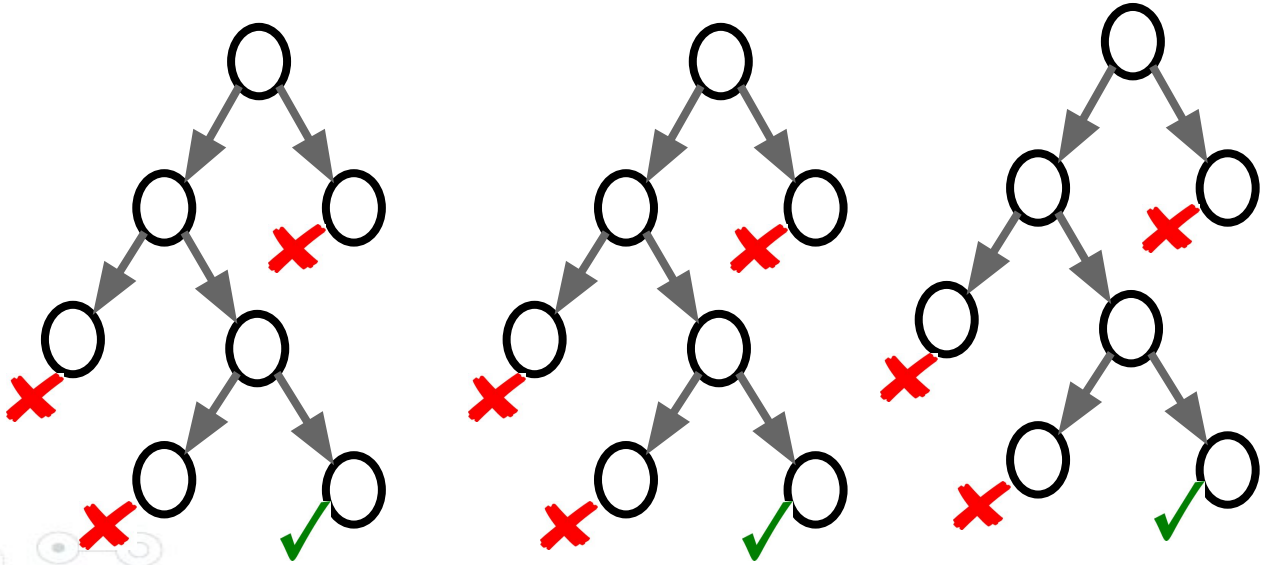
Disadvantages:

- Over-complex trees that not generalize well
- Unstable due to small variations in training data



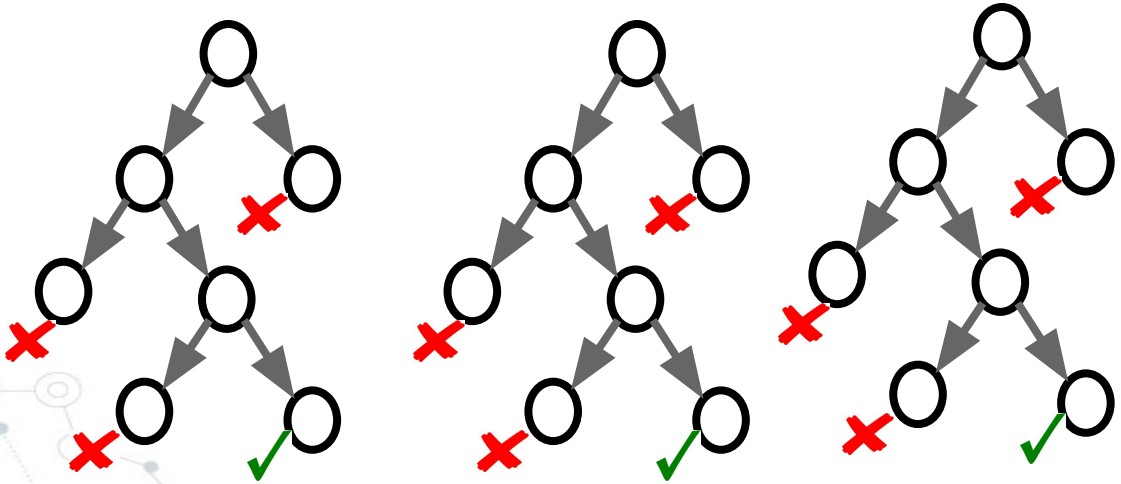
Random Forest

Ensemble method formed with several decision trees

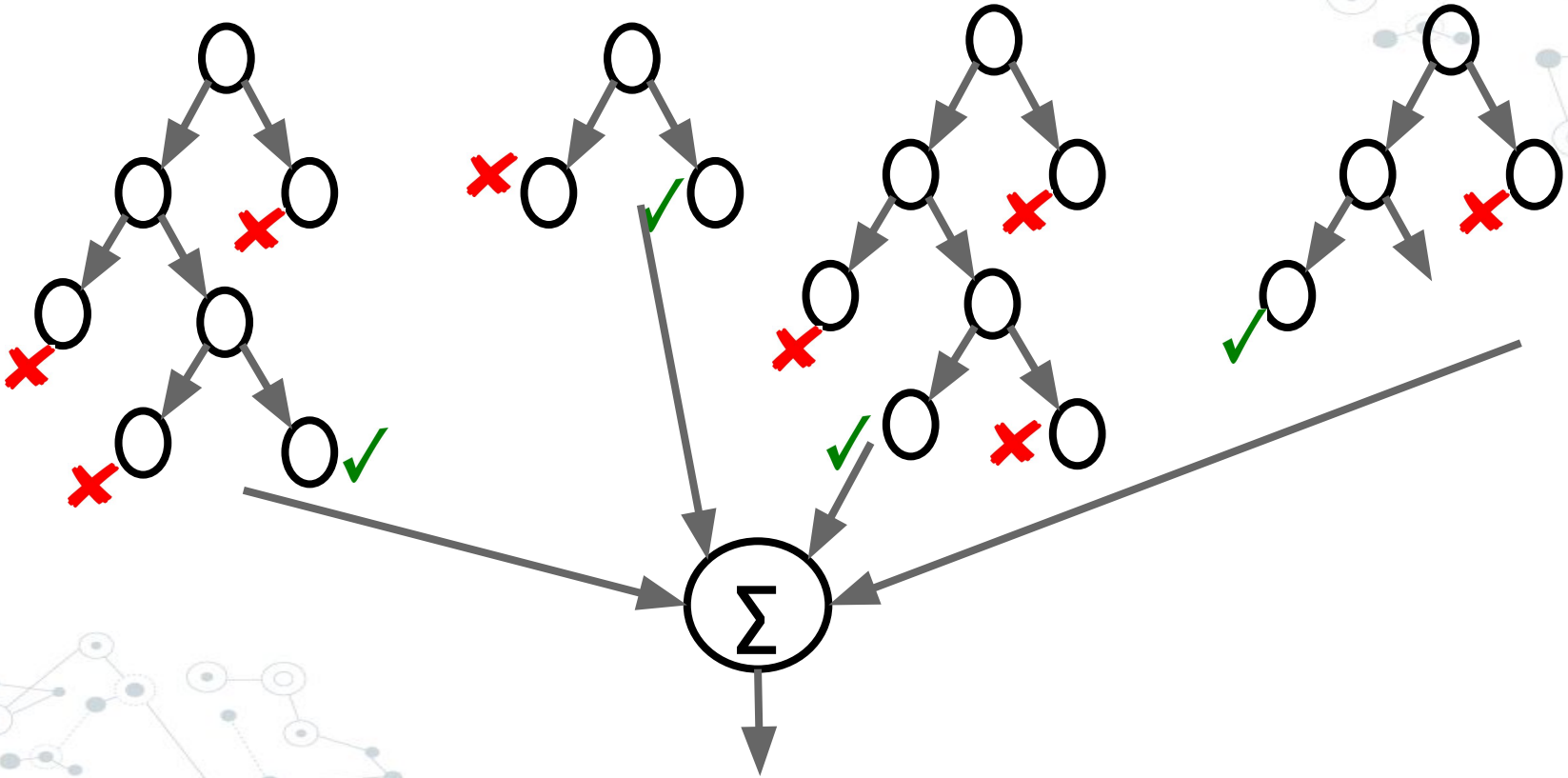


Random Forest

- Collection of tree-structured classifiers {h(**x**,θ), k=1,....}
- θ_k are i.i.d. random trees
- Each tree casts a unit vote for the final classification of input **X**
- The final class of the random forest is chosen is voted by weighted values of each tree



Random Forest



Decision tree vs. Random forest

+ Trees yield insight into decision rules
+ Rather fast
+ Easy to tune parameters
- Prediction of trees tend to have a high variance

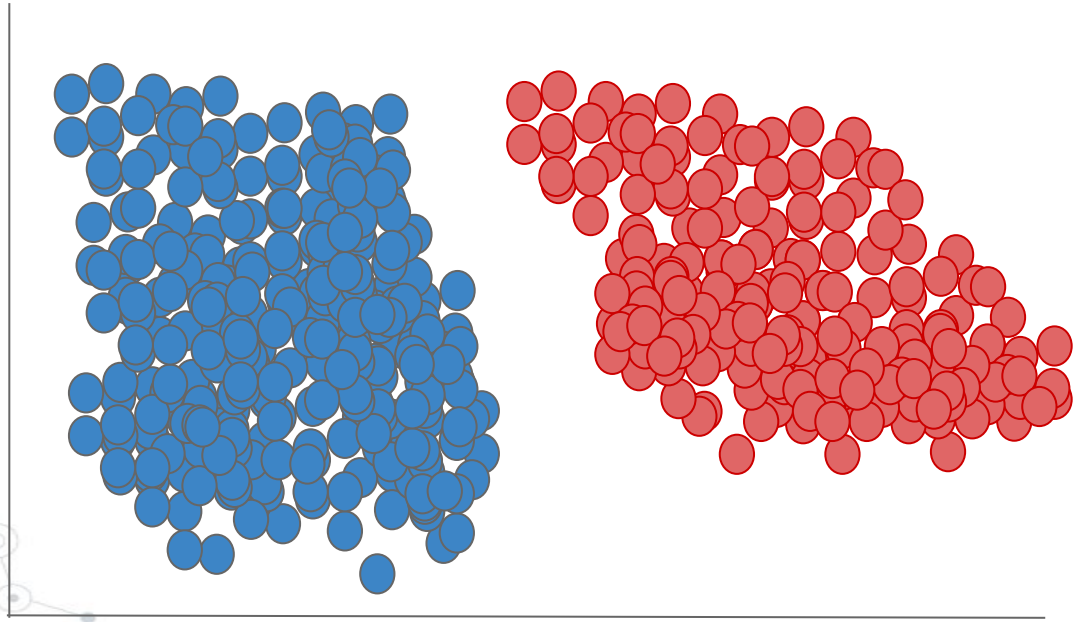
+ Usually a better general performance
+ Easy to tune parameters
- Rather slow
- “Black Box”: Rather hard to get insights into decision rules

Decision Tree Demo

https://github.com/neuraldevs/ML-ND-CD/tree/master/cancer

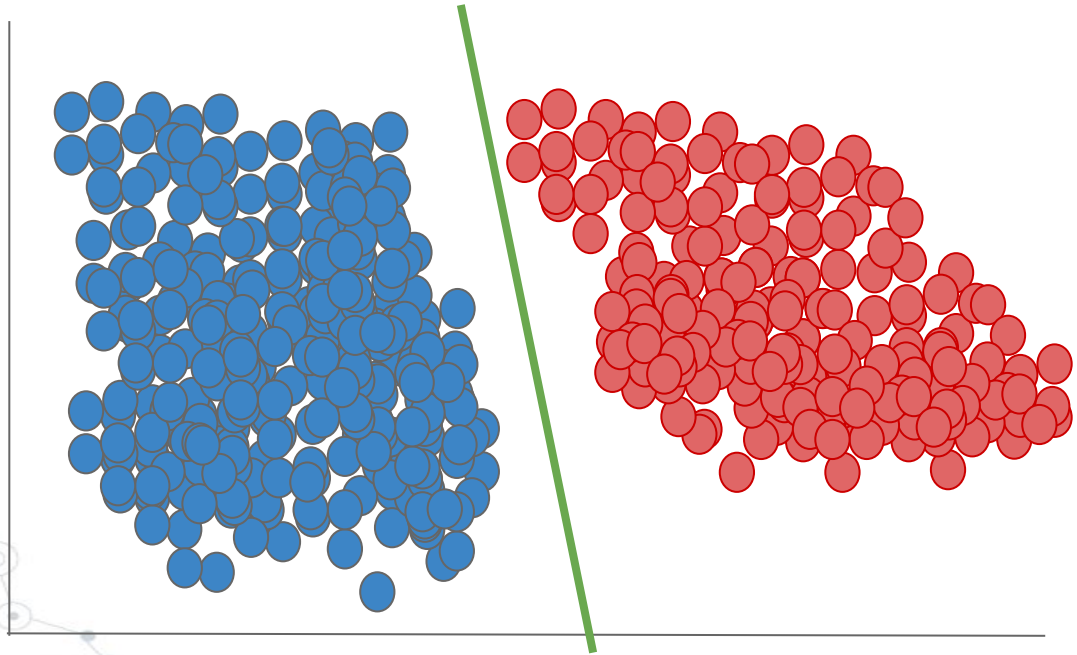
Support vector machines

Which of the linear separators is optimal?



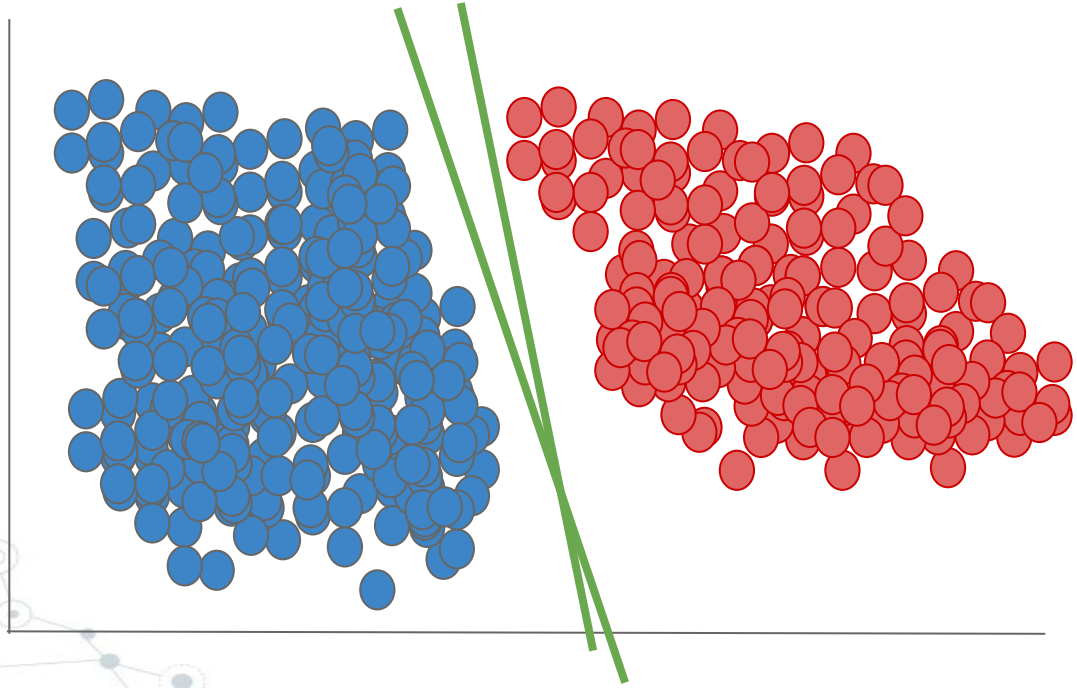
Support vector machines

Which of the linear separators is optimal?



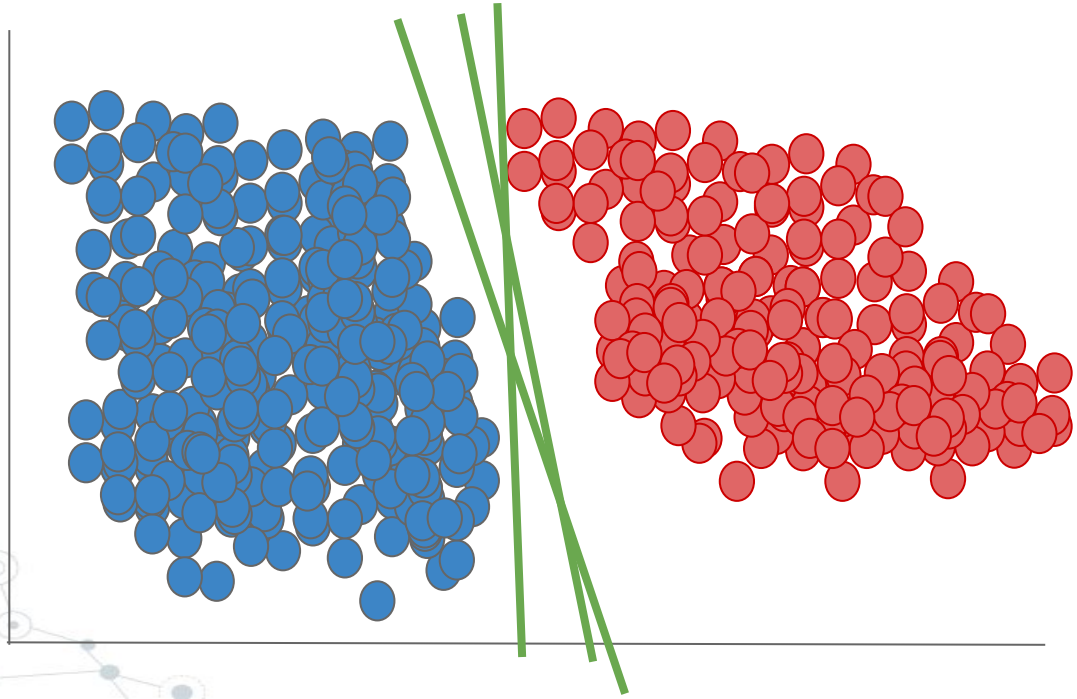
Support vector machines

Which of the linear separators is optimal?



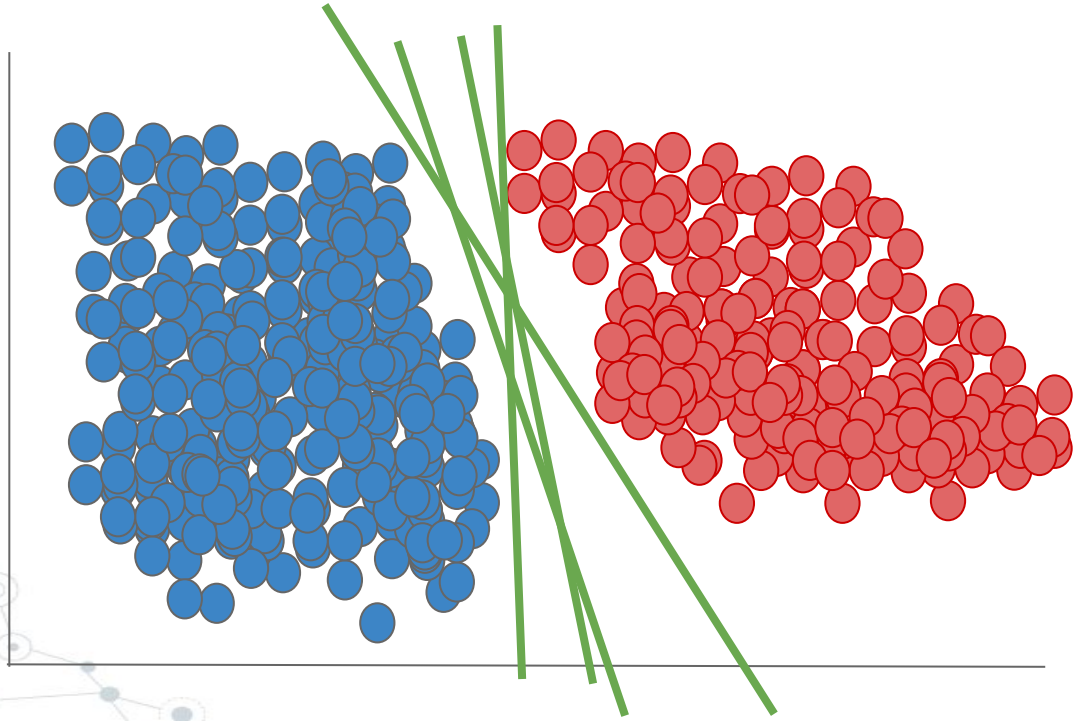
Support vector machines

Which of the linear separators is optimal?



Support vector machines

Which of the linear separators is optimal?

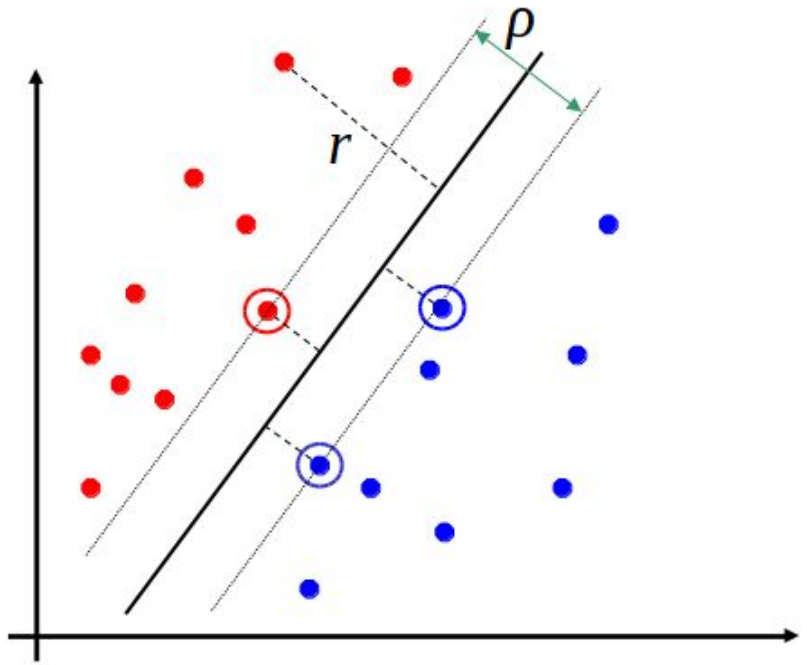


Support vector machines

Which of the linear separators is optimal?

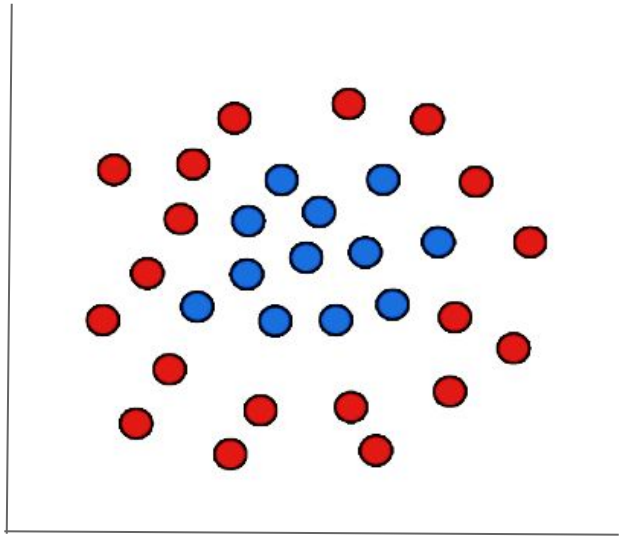
Examples closest to the hyperplane are ***support vectors***.

Margin **ρ** of the separator is the distance between support vectors.



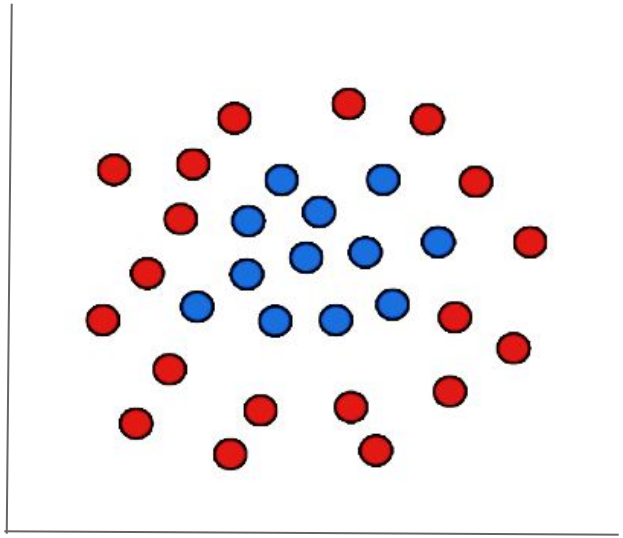
Support vector machines: kernel

Original feature space

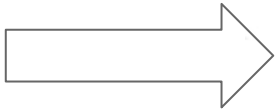


Support vector machines: kernel

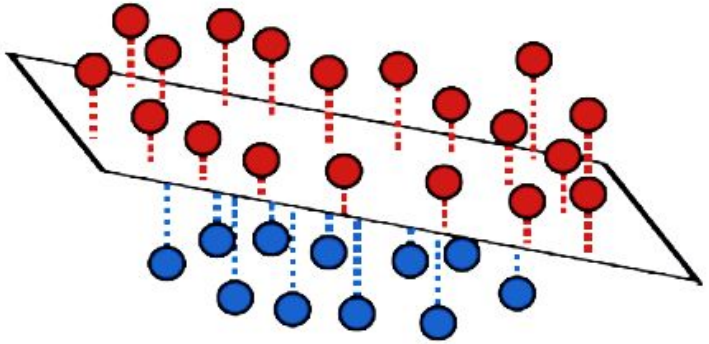
Original feature space



kernel function



Transformed feature space



Support vector machines

+ Effective in high dimensional spaces: high number of features
+ Memory efficient.
+ Kernel functions allow non-linear estimation.
+ Suitable for low number of samples

Support vector machines

+ Effective in high dimensional spaces: high number of features
+ Memory efficient.
+ Kernel functions allow non-linear estimation.
+ Suitable for low number of samples

- Very slow for data with high number of samples
- Totally discriminative method.
- Tricky hyper-parameter tuning: complexity C and kernel parameters (d, gamma)

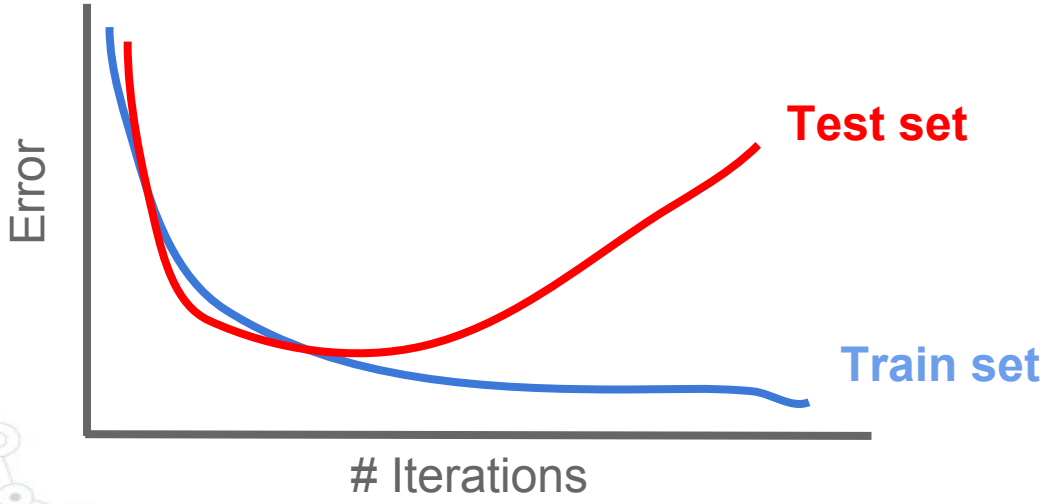
Support vector machines
Demo

https://github.com/neuraldevs/ML-ND-CD/tree/master/cancer

Validation methods

Evaluating estimator performance

- Learning the parameters of a prediction function and testing it on the same data is a **methodological mistake (overfitting)**
- Hold out part of the available data as a **test set**



Validation methods

Evaluating estimator performance

- When evaluating different settings ("**hyperparameters**") for estimators such as **maximum depth** of a decision tree there is still a risk of **overfitting** *on the* **test set.**

- Knowledge about the **test set** can "**leak**" into the model.

Validation methods

Evaluating estimator performance

- When evaluating different settings ("**hyperparameters**") for estimators such as **maximum depth** of a decision tree there is still a risk of **overfitting** *on the* **test set.**

- Knowledge about the **test set** can "**leak**" into the model.

- Another part of the dataset can be held out as a so-called "**validation set**"

- Training proceeds on the **training set**, after which evaluation is done on the **validation set**, and when the experiment seems to be successful, final evaluation can be done on the **test set**.

Validation methods

Performance Measures

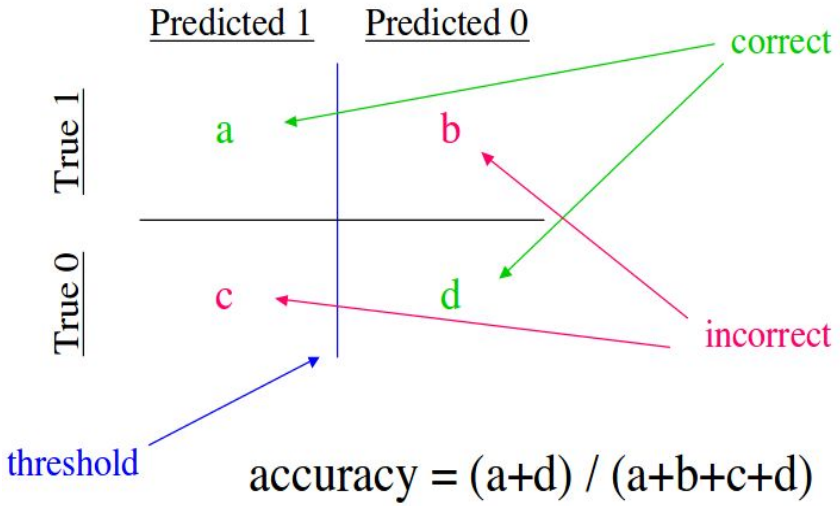
- Accuracy

$$\texttt{accuracy}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} 1(\hat{y}_i = y_i)$$

Validation methods

Performance Measures

- Confusion Matrix

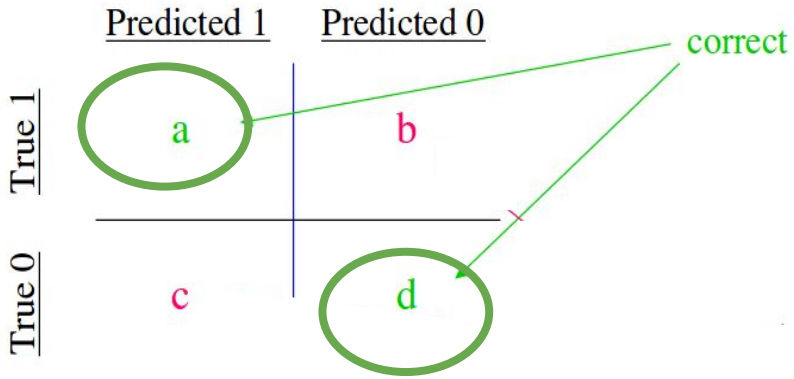


accuracy = (a+d) / (a+b+c+d)

Validation methods

Performance Measures

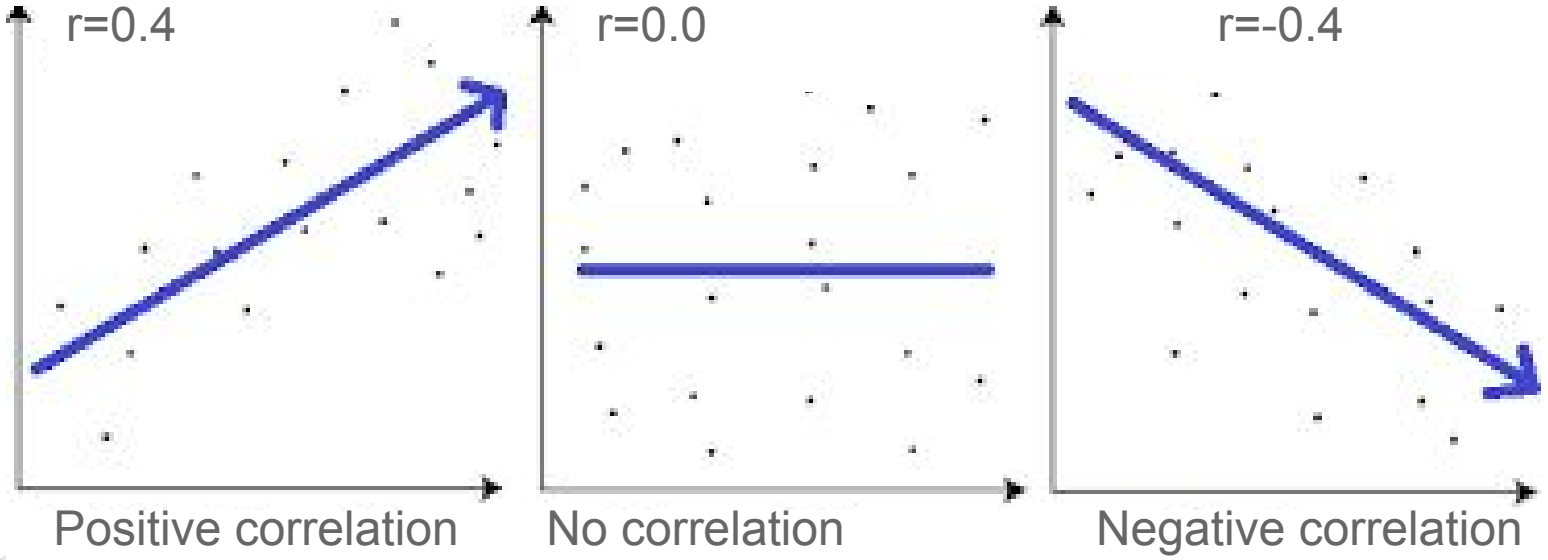
- Specificity / Sensitivity



Validation methods

Performance Measures

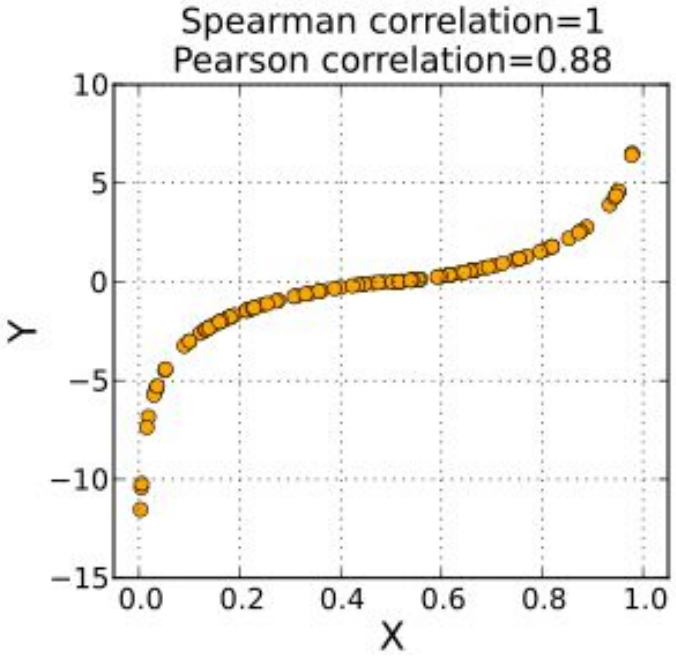
- Pearson correlation coefficient



Validation methods

Performance Measures

- Spearman correlation coefficient



SIATA air quality in Medellín: complete use case

Demo

https://github.com/neuraldevs/ML-ND-CD/tree/master/Siata

Other technologies

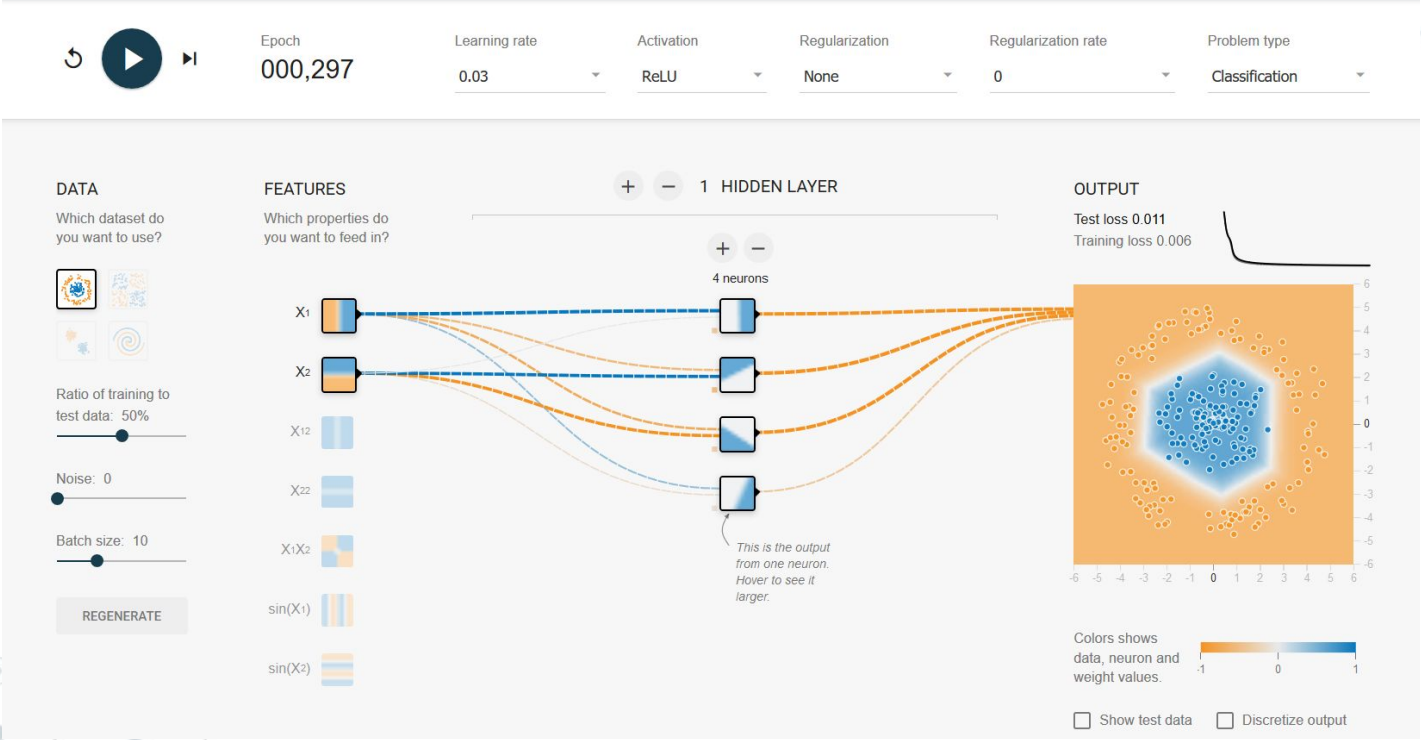
TensorFlow

- Open source software library for numerical computation using data flow graphs.
  - Nodes = represent mathematical operations,
  - tensors = The central unit of data.

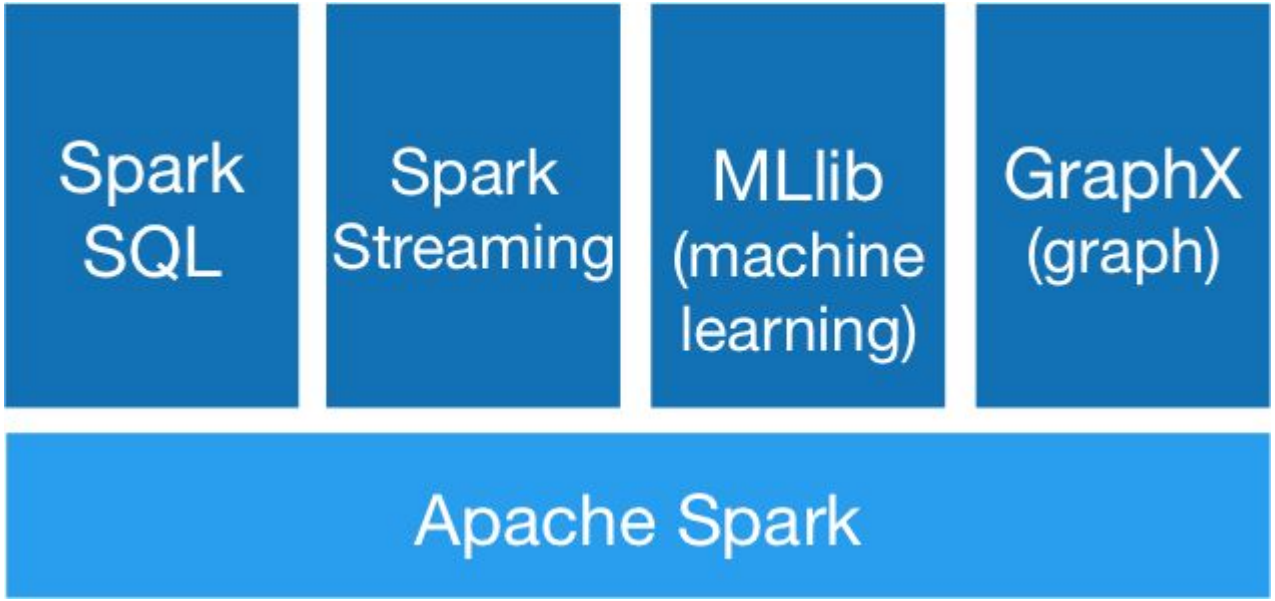

```
node1 = tf.constant(3.0, dtype=tf.float32)
node2 = tf.constant(4.0) # also tf.float32 implicitly

node3 = tf.add(node1, node2)
sess.run(node3)
```

- CPUs
- GPUs
- desktop,
- Server
- Mobile device with a single API.

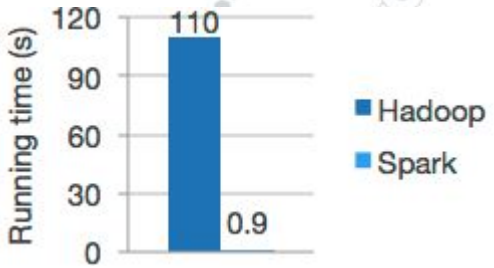
TensorFlow (http://playground.tensorflow.org)



Spark


Spark SQL
Spark Streaming
MLlib (machine learning)
GraphX (graph)
Apache Spark

Spark MLib



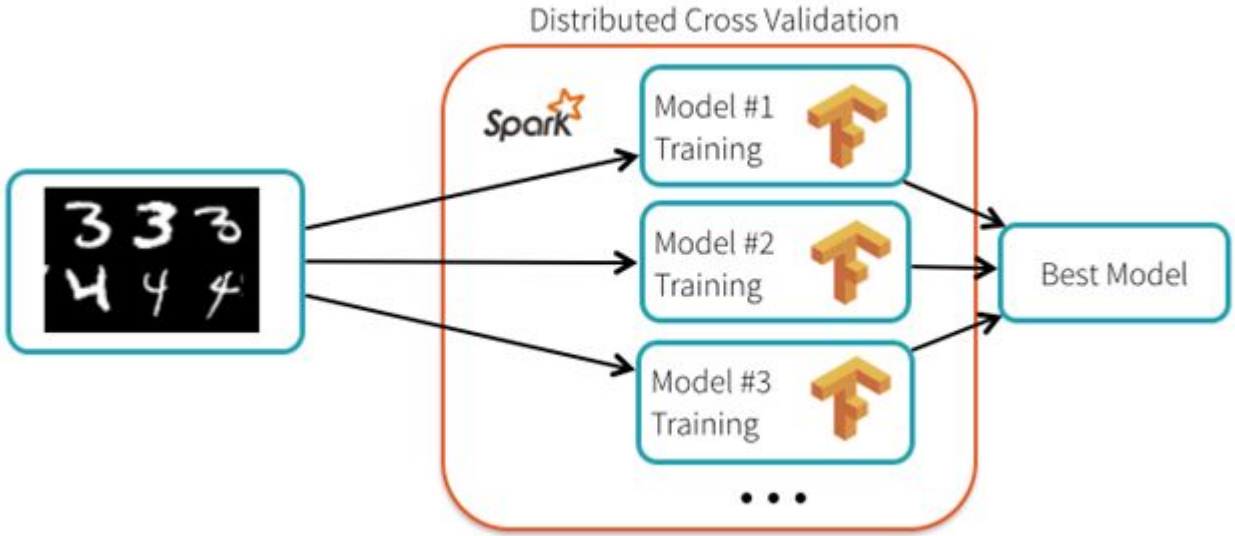
- Run programs up to 100x faster than Hadoop MapReduce in memory, or 10x faster on disk (acyclic data flow).
- Write applications quickly in Java, Scala, Python, R.
- MLlib contains many algorithms and utilities.
- Sparkling Water (spark + h20)

H20

- In an open source, in-memory, distributed, fast, and scalable machine learning and predictive analytics platform that allows you to build machine learning models on big data and provides easy productionalization of those models in an enterprise environment.

Spark + TensorFlow

Hyperparameter Tunning



R Language

- Open source language.
- provides a wide variety of techniques (linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering, …) and graphical techniques
- Highly extensible.
- More that 10000 packages.
- Easily integration with Spark and TensorFlow.

References

- http://blog.revolutionanalytics.com/2017/01/cran-10000.html
- https://spark.apache.org/
- http://docs.h2o.ai/h2o/latest-stable/h2o-docs/welcome.html
- https://databricks.com/blog/2016/01/25/deep-learning-with-apache-spark-and-tensorflow.html
- https://www.tensorflow.org/

Exercises and practice

https://github.com/colomb-ia/retos