INFO-H-515:
BIG DATA ANALYTICS

Introduction to big data analytics

Gianluca Bontempi, Yann-ael Le Borgne

Machine Learning Group
Boulevard de Triomphe - CP 212
http://mlg.ulb.ac.be

INFO-H-515: SECOND PART

- Second part of the course Big Data : Distributed Data Management and Scalable Analytics
- Required
  - first part of the course (throughput, latency, distributed computing, MapReduce, Lambda Architectures, Spark, Spark streaming, scalability)
  - foundations of Machine Learning (INFOF422)
  - programming skills in Python, Spark
- From data management perspective to analytics perspective.
- No oral exam: project (50% of the overall course score)

INTRODUCTION

BIG DATA REVOLUTION

- More than .... gigabytes of data are generated every second, and the rate of data creation is only accelerating.
- CPU technology reached a limit: the ability to drive chips at ever higher clock rates (frequency scaling) is beginning to hit a limit.
- Multicore strategy: keep clock frequency fixed but increasing the number of processing cores.
- Technology for data storage keeps on evolving: cost to store 1 TB of data continues to drop by roughly two times every 14 months. Also technology for collecting data (sensors, cameras, sequencing) continues to drop in cost and improve in performance.
- Data is the new oil: expectations of more and more information and value coming from data (from simple statistics to machine-learning based analytics).

BIG DATA REVOLUTION

- From vertical scaling to horizontal scaling: CPUs aren' getting (much) faster, computational resources are shifting toward parallel and distributed architectures, with multicore and cloud computing platforms providing access to hundreds or thousands of processors.
- Elastic clouds availability and cost effectiveness: Infrastructure as a Service (e.g. AWS) allows to rent hardware on demand, whose size can increase or decrease in nearly real time.
- Custom processing units (GPU) and integrated circuits (FPGA).
- Open source ecosystems for Big Data.
- Data science is moving from labs to the factories (production, IoT, industry 4.0).

FROM MACHINE LEARNING TO BIG DATA ANALYTICS

- What should we keep in mind from the INFO-F-422 course ”Statistical foundations of machine learning”?
- Is an additional course really necessary ?
- Is ”big data” just an hype?
- What is new? what is not new?

WHEN BIG IS BIG?

- Pragmatic definition:
  When either the data is too large to fit on a single machine or it would simply take too long to perform that computation on a single machine.
- In other terms data is big when conventional single machine computing is no more feasible or effective.
- Are all analytics problems big data problems? of course not.
- There are still many companies/business/disciplines who claim having big data problems though those are simple conventional analytics problems.
- Note that this happens also in well-known IT companies (”majority of real-world analytic jobs process less than 100GB of input” [1]).

WHAT IS NOT NEW?

Big data analytics remains an instance of a statistical modeling problem, which aims at inferring estimators/predictors from observations.
Useful notions to keep in mind:

- Estimator as function of data.
- Supervised/unsupervised.
- Data preparation.
- Dimensionality reduction.
- Generalization, bias/variance, underfitting/overfitting.
- Model selection and assessment.
- Iterative refinements

WHAT IS NEW?

Special focus on big data issues

- Volume: in terms of variables and/or samples. It calls for scalable storage and a distributed approach to learning.
- Velocity: it is the rate at which data flows into. Streaming data tasks, online learning.
- Variety: beyond conventional numeric measurements, heterogenous sources (blog posts, tweets, social network interactions, photos, logs).
- Beyond accuracy: performance (e.g. throughput, latency).
- Distribution of the computation: scalability.
- New programming languages, paradigms and tools.

WHAT IS NEW?

Many of the basic assumptions made in single-node systems are no more valid.

- Since data must be partitioned across many nodes on a cluster, algorithms that have wide data dependencies will suffer from the fact that network transfer rates are orders of magnitude slower than memory accesses.
- As the number of machines working on a problem increases, the probability of a failure increases.
- Need of a programming paradigm which is adapted to the characteristics of the underlying system and makes it easy to write code that will execute in a highly parallel manner.
- There is a vast literature on distributed learning and data mining, but most of this literature is ad-hoc (specific learning machine) and provides no general approach for distributing machine learning .

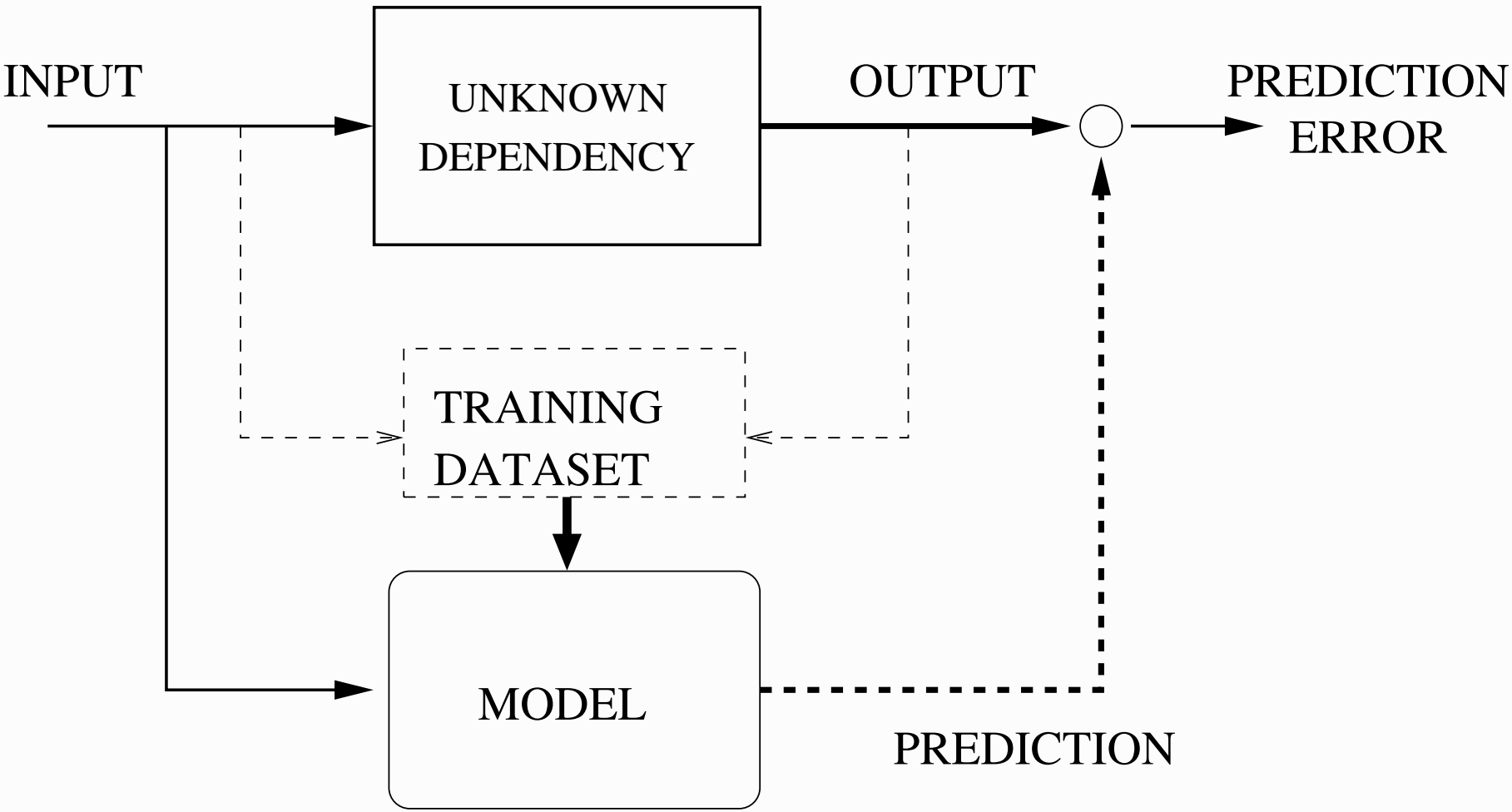
ML REVIEW

SUPERVISED VS UNSUPERVISED TASKS

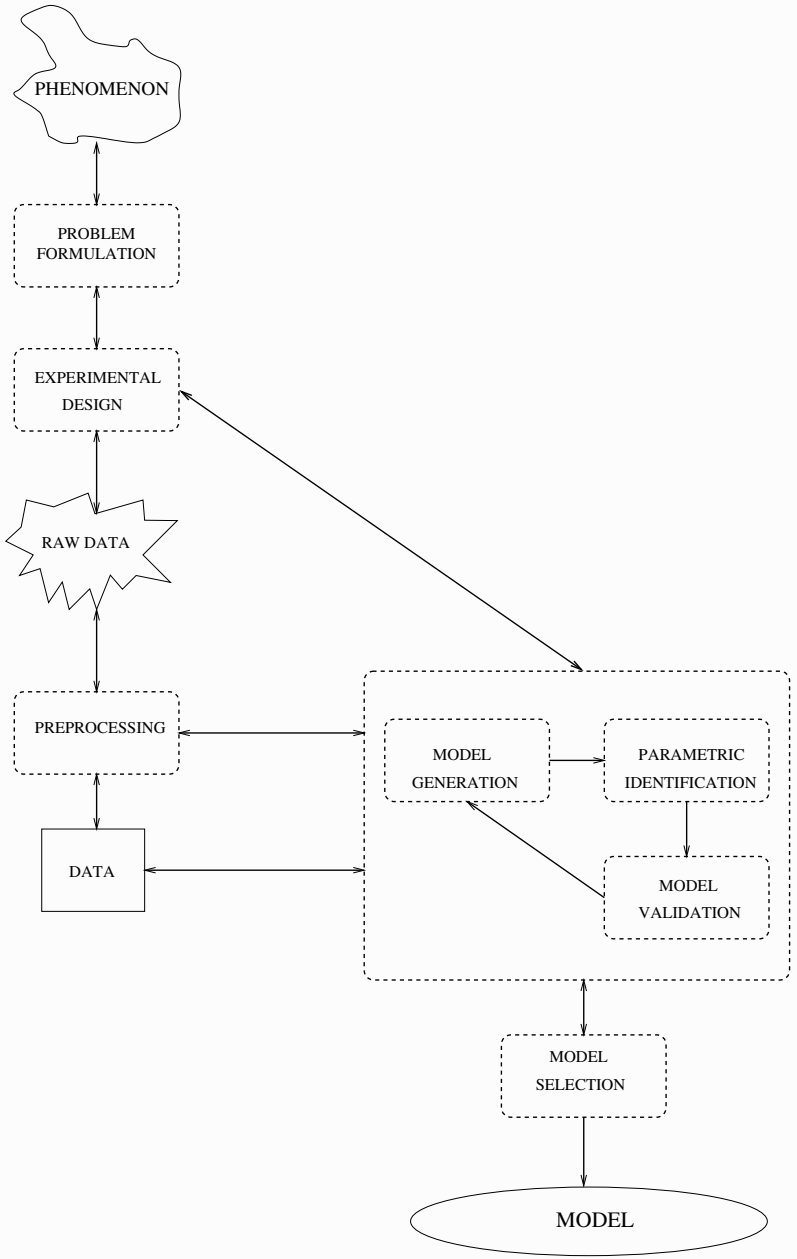
- Supervised
  - Regression
  - Classification
  - Time series forecasting
- Unsupervised
  - Clustering
  - Outlier detection
  - Density estimation

For en extended introduction to the topic refer to [3].

SUPERVISED LEARNING SETTING



MACHINE LEARNING PIPELINE



PREPROCESSING

This step is more and more important and time consuming as data size grows.

- Data collection
- Data cleaning
- Missing data management
- Data formatting

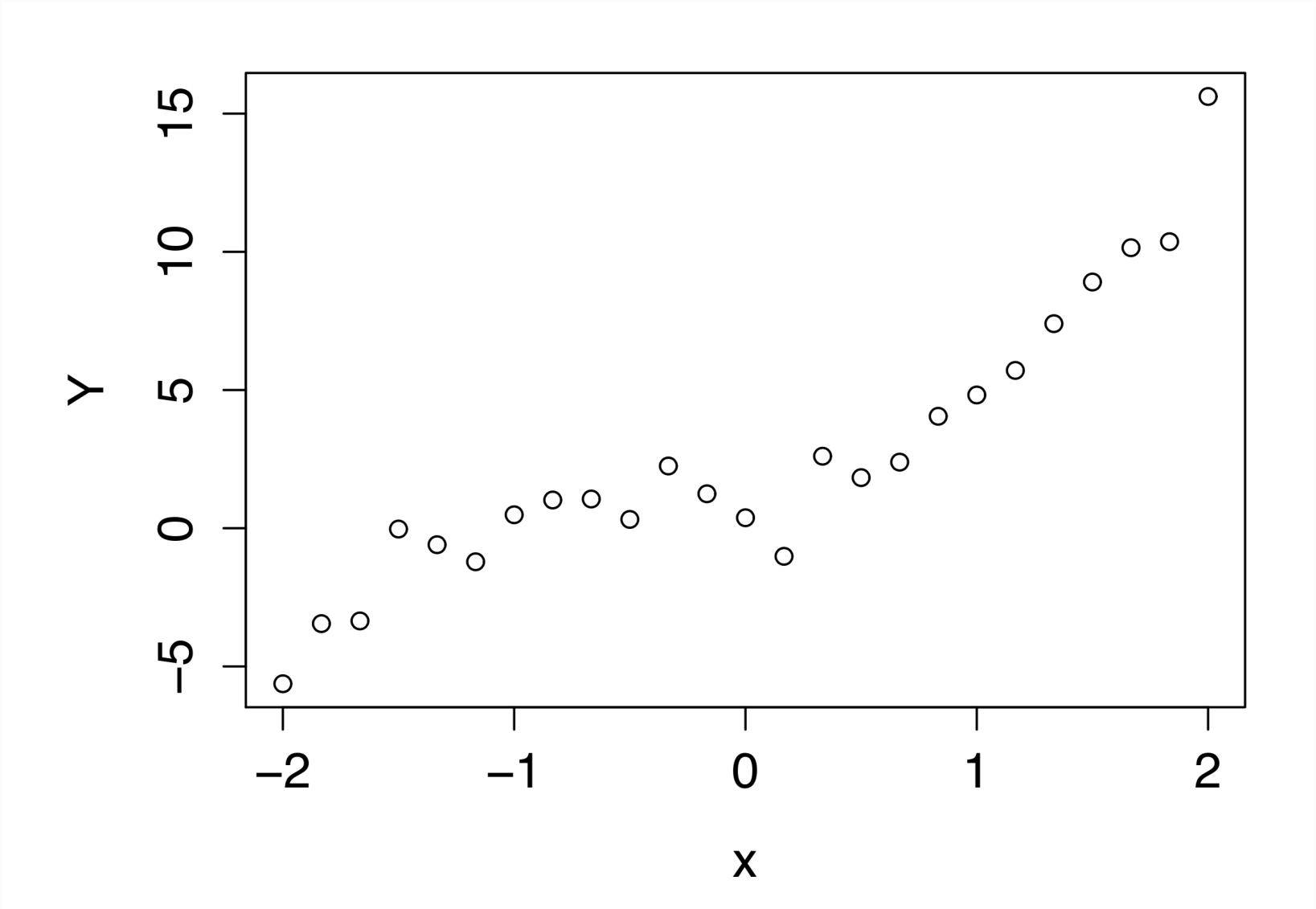
TABULAR DATA FORMATS

In the following we will assume that the training dataset D_N of the supervised learning algorithm can be formatted in the following tabular format

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \quad X = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ \vdots & \vdots & & \vdots \\ x_{N1} & x_{N2} & \dots & x_{Nn} \end{bmatrix}$$

where N is the number of samples and n is the number of features.

SIMPLE REGRESSION TASK

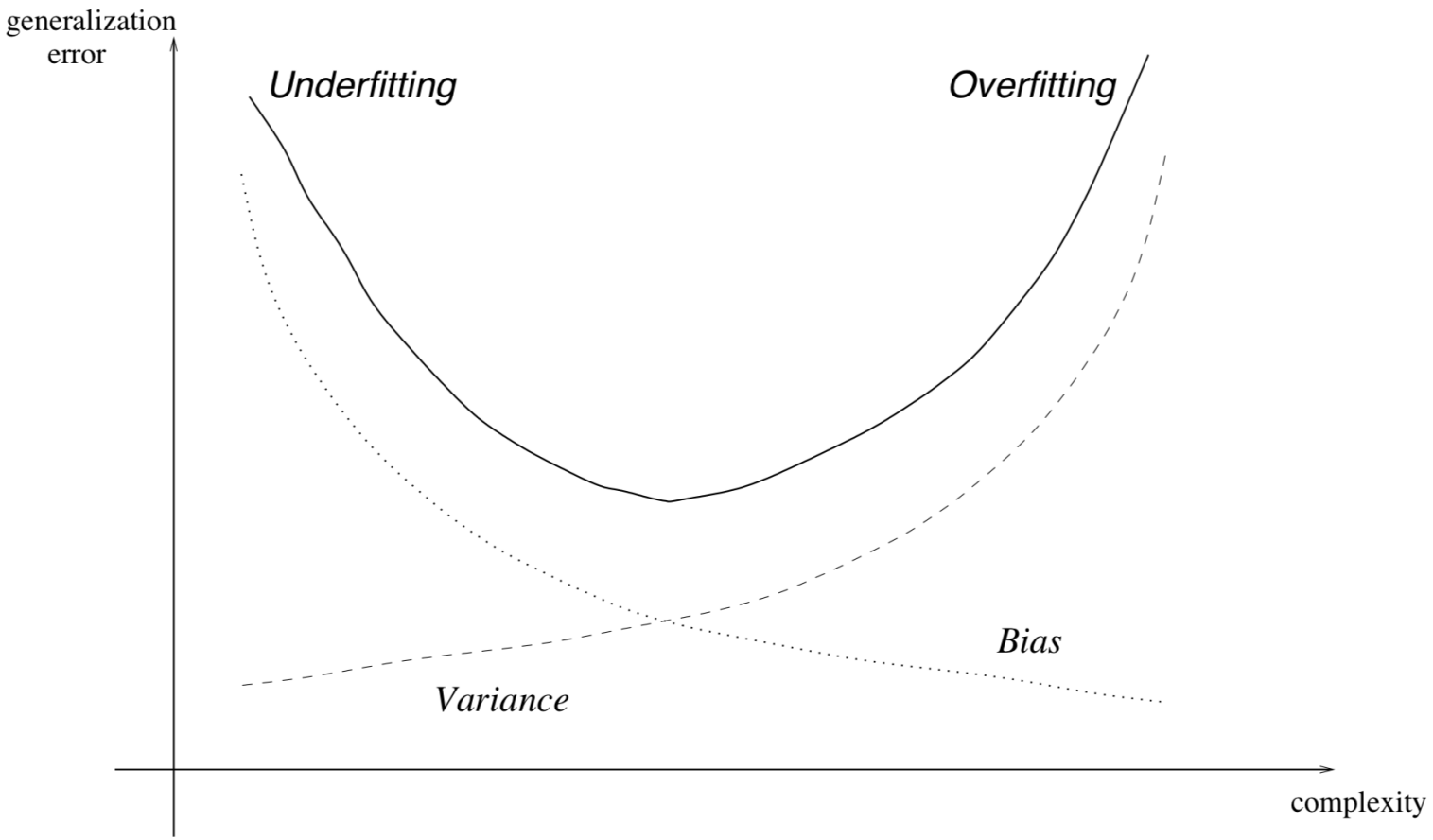


UNDERFITTING/OVERFITTING

UNDERFITTING/OVERFITTING

UNDERFITTING/OVERFITTING

BIAS/VARIANCE TRADE-OFF



THE LEARNING PROCEDURE

A learning procedure [3] aims at:

1. to choose a parametric family of hypothesis $h(x, \alpha)$ which contains or gives good approximation of the unknown function f (**structural identification**).
2. within the family $h(x, \alpha)$, to estimate on the basis of the training set D_N the parameter α_N which best approximates f (**parametric identification**).

A learning procedure is composed of two nested loops:

1. an external structural identification loop which goes through different model structures
2. an inner parametric identification loop which searches for the best parameter vector within the family structure.

PARAMETRIC IDENTIFICATION

The parametric identification of the hypothesis is done according to ERM (Empirical Risk Minimization) principle where

$$\alpha_N = \alpha(D_N) = \arg\min_{\alpha \in \Lambda} \widehat{MISE}_{emp}(\alpha)$$

minimizes the **empirical risk or training error**

$$\widehat{MISE}_{emp}(\alpha) = \frac{\sum_{i=1}^{N} (y_i - h(x_i, \alpha))^2}{N}$$

constructed on the basis of the data set D_N.
The example before showed that the training error is a biased (i.e. optimistic) estimation of the real generalization error (MISE).

VALIDATION TECHNIQUES

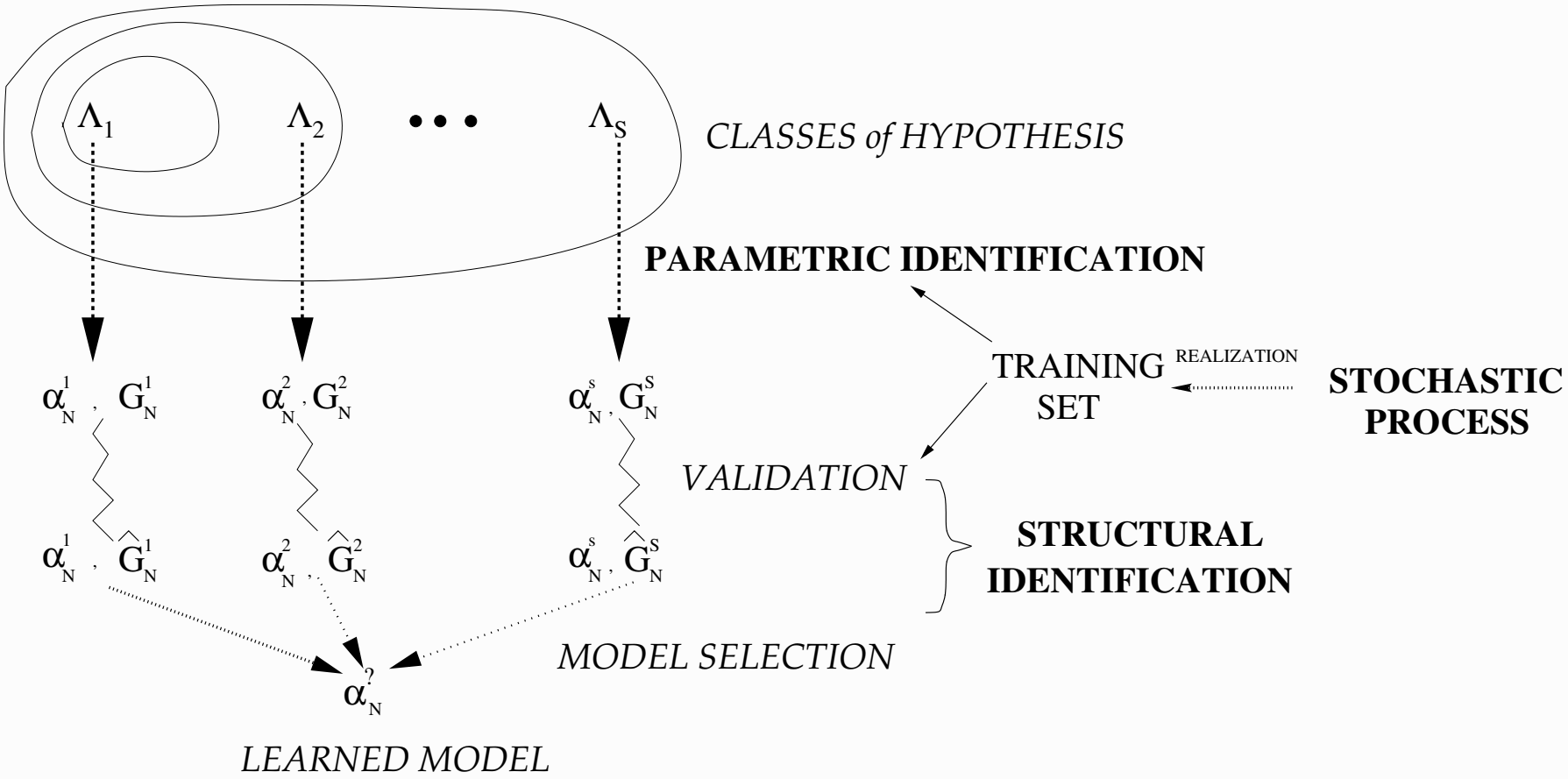
How to measure MISE in a reliable way on a finite dataset? The most common techniques to return an estimate $\widehat{\text{MISE}}$ are

- **Testing:** a testing sequence independent of D_N and distributed according to the same probability distribution is used to assess the quality. In practice, unfortunately, an additional set of input/output observations is rarely available.
- **Holdout:** it partitions the data D_N into two mutually exclusive subsets, the training set D_{tr} and the holdout or test set $D_{N_{ts}}$.
- k**-fold Cross-validation:** the set D_N is randomly divided into k mutually exclusive test partitions of approximately equal size. The cases not found in each test partition are independently used for selecting the hypothesis which will be tested on the partition itself. The average error over all the k partitions is the cross-validated error rate.

MODEL SELECTION

- Model selection concerns the final choice of the model structure in the set that has been proposed by model generation and assessed by model validation.
- Two typical approaches:
  1. the winner-takes-all approach: choose the model structure that minimize an accurate estimate of the generalization error (e.g. cross-validation)
  2. the combination of estimators approach: combination (e.g. averaging) of different models (e.g. different orders, different family).

MODEL SELECTION



MODEL COMBINATION

- The winner-takes-all approach is intuitively the approach which should work the best.
- However, results in machine learning show that the accuracy of the final model can be improved not by choosing the model structure which is expected to predict the best but by creating a model whose output is the combination of the output of models having different structures.
- The reason is that in reality any chosen hypothesis $h(\cdot, \alpha_N)$ is only an estimate of the real target and, like any estimate, is affected by a bias and a variance term.
- See in [3] theoretical results on the combination of estimators.
- Bagging is a well-known example of model combination.

BAGGING

- Consider a dataset D_N and a learning procedure to build an hypothesis α_N from D_N.
- The idea of **bagging** or **bootstrap aggregating** is to imitate the stochastic process underlying the realization of D_N in order to reduce the variance of $h(\cdot, \boldsymbol{\alpha}_N)$.
- A set of B repeated bootstrap samples $D_N^{(b)}$, $b = 1, \ldots, B$ are taken from D_N.
- A model $\alpha_N^{(b)}$ is built for each $D_N^{(b)}$.
- A final predictor is built by aggregating the B models $\alpha_N^{(b)}$.
- In the regression case the bagging predictor is

$$h_{bag}(x) = \frac{1}{B} \sum_{b=1}^{B} h(x, \alpha_N^{(b)})$$

- In the classification case a majority vote is used.

FEATURE SELECTION

Machine learning algorithms are known to degrade in performance (prediction accuracy) when faced with many inputs (aka features) that are not necessary for predicting the desired output.
There are many potential benefits of feature selection:

- facilitating data visualization and data understanding,
- reducing the measurement and storage requirements,
- reducing training and utilization times of the final model,
- defying the curse of dimensionality to improve prediction performance.

FEATURE SELECTION STRATEGIES

1. **Filter methods:** preprocessing methods. They attempt to assess the merits of features from the data, ignoring the effects of the selected feature subset on the performance of the learning algorithm. Examples are methods that select variables by ranking them through compression techniques (like PCA or clustering) or by computing correlation with the output.
2. **Wrapper methods:** these methods assess subsets of variables according to their usefulness to a given predictor. The method conducts a search for a good subset using the learning algorithm itself as part of the evaluation function.
3. **Embedded methods:** selection as part of the learning procedure and are usually specific to given learning machines. Examples are classification trees, random forests, and methods based on regularization techniques (e.g. lasso)

OTHER ISSUES

- Unbalancedness
- Non stationarity, concept drift
- Semi supervised learning, active learning

BIG DATA ANALYTICS APPLICATIONS

BIG DATA ANALYTICS APPLICATIONS

Some examples [2]:

- Credit risk modeling
- Fraud detection
- Net lift response modeling
- Churn prediction
- Monitoring of IoT applications
- Recommender systems
- Web analytics
- Social media analytics

CREDIT RISK MODELING

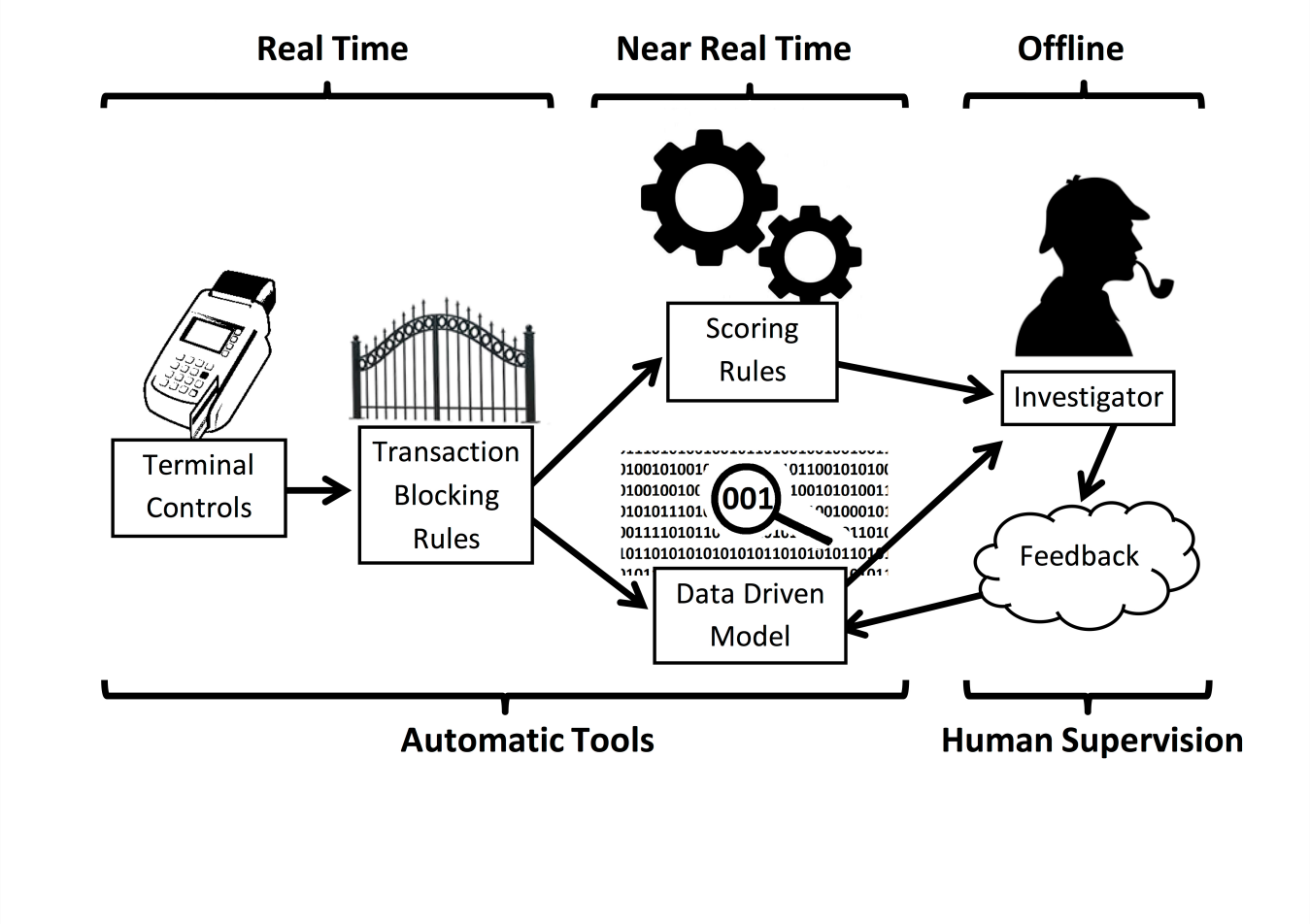
- Business objective: calculate the capital needed to protect against expected losses.
- Analytics objective: estimate the probability of default for a portfolio of new or existing customers (application credit scorecards, behavioural scorecards).
- Features:
  - Application data at loan origination: age, income, marital status, employment status, salary, savings amount.
  - Credit bureau data at loan origination: bureau score, delinquency history, number of bureau checks, number of outstanding credits, address changes, LGD (loss given default, i.e. loss expressed as a percentage of the outstanding loan amount), EAD (exposure at default)
  - Default status informations (12 or 18 months ahead): did (s)he pay so far?
- Methods: supervised (logistic regression, ML classifiers)
- Issues: large volume, average dimensionality, time-varying attributes.

CREDIT-CARD FRAUD DETECTION

- Business objective: preserve the reputation of the company, avoid losses due to frauds as well as false positive (erroneous blocking of unaffected accounts)
- Analytics objective: estimate the probability that transaction is fraudulent in nearly real-time
- Features:
  - Information about cardholders: age, sex, country, social networks information (network, friends)
  - Information about terminal: electronic, face-to-face, country
  - Information about cardholder behavior: statistics about previous spending
- Methods: unsupervised (outlier detection) and supervised (logistic regression, ML classifiers)

- Issues
  - large volume, average dimensionality,
  - streaming data,
  - non stationarity (fraudsters change and adapt their strategy)
  - unbalancedness (many more genuine transactions than frauds),
  - real-time issues,
  - cost and delay of human labelling,
  - black-box vs white-box (e.g. expert based rules) models,
  - accuracy measures: maximize precision in Top-K alerts.

FRAUD DETECTION SYSTEM



From [4].

CHURN PREDICTION

- Business objective: retain profitable customers and avoid churn (defection, loss) of customers. Attracting customer costs six times more than retaining him.
- Analytics objective: predict the probability that a customer is going to churn
- Features:
  - Information about customer: demographics (age, gender, marital status), social network (homophily, i.e. connected clients behave similarly)
  - Information about relationship: length of relationship, number of products purchased
  - RFM:
    - Recency: time frame (days, weeks, months) since last purchase
    - Frequency: number of purchases within a given time frame
    - Monetary: EUR value of purchases
- Methods: supervised (logistic regression, ML classifiers, neural networks) or time-to-event (survival analysis)

- Issues
  - definition of churn (contract termination, service cancellation or nonrenewal)
  - notion of inactivity
  - different perspectives: at company level (identify company related causes of churn: e.g. quality of service, prices) or individual level (understand customer attitude, preferences, other interests, social relationships),
  - decision-making (which actions to take?)
  - black-boxes vs. interpretability of the model
  - causal reasoning (understanding the reasons of a churn)

NET LIFT RESPONSE MODELING

- Business objective: Deepen or recover customer relationships by means of targeted or win-back campaigns (e.g. mails, emails, catalog) to remind the benefits, propose discounts
- Analytics objective: estimate the probability that a customer reacts positively to a campaign
- Features:
  - Information about customer: demographics (age, gender, marital status), social network, churning risk
  - Information about relationship: length of relationship, number of products purchased
  - RFM:
    - Recency: time frame (days, weeks, months) since last purchase
    - Frequency: number of purchases within a given time frame
    - Monetary: EUR value of purchases
- Methods: supervised (logistic regression, ML classifiers)

- Issues:
  - need of different datasets (those who received a marketing campaign and those who did not),
  - segmentation of customers (those who would never buy, those would anyhow buy, and swing clients, i.e. those who would buy if exposed to a marketing offer)
  - causal reasoning (predict the impact of an action and not simply of an observation)
  - interpretability of the model
  - different scores to be estimated: P(buy|targeted) , P(buy| no targeted), or the difference between them.

WEB ANALYTICS

- Business objective: measure, collect, analyze Internet data in order to understand and optimize Web usage
- Analytics objective: estimate the probability that a user performs an action (e.g. order, newsletter sign up)
- Features:
  - Information from web server log: remote host, remote log name, date, time, resource requested, HTTP status, browser and platform, session or persistent cookies (user details), number of transferred bytes
  - Navigation analysis: frequent navigation patterns, other visited pages, click density
  - Site search reports: keywords, bounce rate
- Methods: supervised (logistic regression, ML classifiers), multivariate testing (test of alternative versions of web page)

- Issues and challenges:
  - definition KPI (key performance indicators): page views per visit, new visitors, return visitors, visit length, bounce rate, conversion rate (percentage of visits for which an action is observed)
  - use of visualization by dashboard to transform metrics into actionable insights: trends, seasonality, forecasting
  - segmentation (search traffic or not, geographical region)
  - search engine marketing: improve positions in ranking (inclusion ratio, robot crawl statistics, inbound links, ranking of top keywords)

INTERNET OF THINGS

- Business objective: global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things (sensors, equipment, pipelines, smart meters, OBU). Expected more than 25 billions devices by 2020
- Analytics objective: monitoring, forecasting, detecting changes or faults
- Features:
  - Sensor streams of measures (from real numbers to images)
  - Device properties: geolocalization, measure time, hardware
  - Other sources: weather forecasting, social data
- Methods: supervised (logistic regression, ML classifiers), forecasting, concept drift detectors
- Challenges; large dimensionality, dynamic, non stationarity, real-time , irregular and non structured data, data fusion, anomaly detection spatio-temporal analysis, noise.

BIOINFORMATICS

- Medical objective: assess treatment efficacy on the basis of patient physiological and molecular portrait
- Analytics objective: predict survival on the basis of historical and genomic data
- Features:
  - -omics information (proteomics, genomics, epigenomics)
  - patient physiological data (heart rate, tension, temperature, age)
  - drug information
- Methods: supervised (logistic regression, ML classifiers), survival analysis, feature selection
- Challenges; huge dimensionality, horizontal datasets, privacy.

HOMEWORK: STUDENT MOTIVATION APP

- Think now about an app to motivate students to go back to study.
- Which data could you use it?
- Which variables?
- Is it a supervised or unsupervised problems?
- How to assess the success rate?
- What are the main challenges?

SCALABLE MACHINE LEARNING

SCALABILITY

The aim of using multiprocessors is the scalability of the system, i.e. capability to handle a growing amount of work, or the potential to be enlarged to accommodate that growth.

- Scalability: ability of a system to maintain performance under increased load by adding more resources.
- Load: amount of existing data, rate of incoming data and required quality of service (e.g. number of queries per second)
- Linearly scalable: performance is maintained by adding resources in proportion to the increased load
- A nonlinearly scalable system is typically not very useful
- NB: a scalable system does not necessarily address latency and throughput at the same time.
- Kind of parallelism determined by grain size (i.e. the amount of data per process)

FINE GRAINED PARALLELISM

A program is broken down to a large number of small tasks

- Task parallelism: segmentation of the overall algorithm into parts, some of which may be executed concurrently
- Need redefining the entire algorithm as a dataflow of elementary tasks and eliminating bottlenecks.
- Significant expertise required.
- High number of processors but amount of work per task is low
- Use of supercomputer, special-purpose parallel computers with many processors, shared memory and specialized hardware (GPUs and FPGAs)
- Ad hoc compilers
- Conventional operating system
- Scale-up (vertically): scalability is obtained by adding more resources (e.g. faster PUs or more memory)
- It typically improves latency.

COARSE-GRAINED PARALLELISM

A program is split into large (high granularity) tasks

- Adequate for high volume tasks
- Data parallelism: concurrent calculations over portions of data
- Move code where data is
- Use of clusters, large collection of commodity hardware, including conventional processors (”compute nodes”) connected by Ethernet cables or inexpensive switches.
- distributed file system (DFS), providing replication of data or redundancy against failure (Google File System (GFS), Hadoop Distributed File System (HDFS) and Cloud Store).
- Use of conventional languages extended by API.
- Scale-out (horizontally): scalability is obtained by adding more commodity nodes.
- Increased overhead and communication costs associated with having more machines, risk of imbalance
- It typically improves throughput.

SCALE-UP VS SCALE-OUT SOLUTIONS

- Conventional wisdom in industry and academia is that scaling out using a cluster is better than scaling up by adding more resources to a single server.
- Is this the right approach?
- According to a Microsoft paper, the majority of real-world analytic jobs process less than 100 GB of input, though scale-out infrastructures (Hadoop/MapReduce) were originally designed for petascale processing.

Read the paper ”Nobody ever got fired for buying a cluster .”

EMBARRASSINGLY PARALLEL TASKS

- Tasks that do not have any dependencies (for instance, there is no need to communicate data among them in a message-passing setting).
- These tasks require no task-to-task communication to complete its parts of the computation, so their computing/communication ratio is infinity.
- Example: Monte Carlo simulation tasks running independently, with no synchronization with their peer tasks from start to finish.
- computation/communication ratio: the time a task spends actually computing divided by the time it spends communicating

SCALING UP MACHINE LEARNING

The wide range of platforms and frameworks for parallel and distributed computing presents both opportunities and challenges for data scientists. There are multiple reasons to parallelize learning algorithms

- large volume of data instances and data resident on distributed file systems (vertical big data)
- high input dimensionality (horizontal big data)
- complexity of model parametric identification
- model selection and hyperparameters calibration
- resampling and averaging approaches
- performance requirements (e.g. latency and throughput)

HIGH VOLUMES

Let us suppose the data are stored in a rectangular matrix where the rows are instances and the columns are the features.

- Many instances: data parallelism by partitioning the matrix rowwise into subsets of instances that are then processed independently (e.g., parameters update)
- Many features: data parallelism by splitting it columnwise for algorithms that can decouple the computation across features (e.g., decision trees).

Drawbacks: assumptions of i.i.d samples, independent features

VOLUME: VERTICAL BIG DATA

Big vertical datasets are common in domains where streams of data are measured and collected sequentially like

- internet
- finance (event is a transaction)
- business transactions (e.g. credit card transaction)
- sensors (Internet of Things, robots, cameras)

VOLUME: HORIZONTAL BIG DATA

Big horizontal datasets are common in domains where data instances are described by a very large number of feature (or attributes) like

- Natural language processing: a feature per word
- Bioinformatics: a feature per genetic/genomic feature (e.g. variant, gene expression, methylation)
- Images: a feature per pixel

Data are sometimes sparse yet not always

SCALABLE PARAMETRIC IDENTIFICATION

Parametric identification is an example of multivariate optimization problem.

- In nonlinear models use of long and time intensive training procedures (e.g. multi-layer (deep) neural networks)
- Fine-grained parallel implementation of parametric identification procedures (e.g. backpropagation in neural networks) in multiprocessor or custom processor solutions (e.g. GPU)
- Coarse-grained decomposition at the sample level (e.g. in gradient descent algorithms)
- Ad-hoc solutions for tasks like image or video processing

SCALABLE MODEL ASSESSMENT AND VALIDATION

Examples of embarrassing parallel tasks in machine learning are

- Grid search (parameter sweeping) on large hyperparameter spaces
- Validation procedures like cross-validation, leave-on-out, bootstraping
- Bagging, averaging
- Univariate feature ranking
- Model racing and selection
- Subsampling or Monte Carlo strategies
- Multiple statistical significance testing (e.g. parallel p-value computing)
- Estimation of learners' variance

OTHER SCALABLE TASKS

- Similarity search, hashing
- Data-stream processing, online learning
- Search engine
- Graph mining

METRICS OF PARALLELIZATION PERFORMANCE

- Classical analysis of algorithms complexity is based on O-notation to quantify computational costs.
- This is hardly usable in machine learning, since the execution time (e.g. related to termination conditions) is often dependent on data distribution which is not known a priori
- Key metrics used for analyzing computational performance of parallel algorithms are
  1. Speedup
  2. Scaleup
  3. Sizeup
- Don't forget that these measures should be also be assessed
  - in paired settings (e.g. same dataset)
  - wrt generalization accuracy

SPEED-UP

Let $T(N, p)$ the execution time of an algorithm where N is the size of the input data and p is the number of processors.

- Speed-up refers to how much a parallel algorithm is faster than a corresponding sequential algorithm.
- Speed-up analysis holds the input size and equal to N, and grows the number p of workers. It is defined by the following formula:

$$\text{Speed-up}(p) = \frac{T(N, 1)}{T(N, p)}$$

  where $T(N, p)$ is the execution time with p workers.
- Linear speed-up or ideal speedup is obtained when the speed-up is a linear function of p (e.g. by doubling the number of processors we double the speed).
- Linear speedup is difficult to achieve because of communication costs.

SIZE-UP

- Size-up analysis holds the number of workers in the system constant and equal to p, and grows the size N of the datasets by the factor m.
- Size-up measures how much longer it takes on a given system, when the dataset size is m-times larger than the original dataset.
- It is defined by the following formula:

$$\text{Size-up}(m) = \frac{T(mN, p)}{T(N, p)}$$

where $T(mN, p)$ is the execution time of the algorithm for processing a data set of size mN

SCALE-UP

- Scale-up evaluates the ability of the algorithm to deal with the growth of both the system and the dataset size.
- Scale-up is defined as the ability of a m-times larger system to perform a m-times larger job in the same run-time as the original system.
- It is defined by the following formula:

$$\text{Scale-up}(m) = \frac{T(N, p)}{T(mN, mp)}$$

- Scale-up experiments are performed by increasing the size of the datasets in direct proportion to the number of cores in the system.

Raja Appuswamy, Christos Gkantsidis, Dushyanth Narayanan, Orion Hodson, and Ant Rowstron.
Nobody ever got fired for buying a cluster.
Technical report, January 2013.

B. Baesens.
Analytics in a Big Data World: the essential guide to data science and its applications.
Wiley, 2014.

G. Bontempi.
Statistical Foundations of Machine Learning.
2012.
Handbook of INFOF422 course.

Fabrizio Carcillo, Andrea Dal Pozzolo, Yann-Ael Le Borgne, Olivier Caelen, Yannis Mazzer, and Gianluca Bontempi.
Scarff: A scalable framework for streaming credit card fraud detection with spark.
Information Fusion, 41:182 – 194, 2018.