

How to do Quantitative Empirical Research

TDT70 - AI Masterclass - Guest Lecture

Ole Jakob Mengshoel

Department of Computer Science

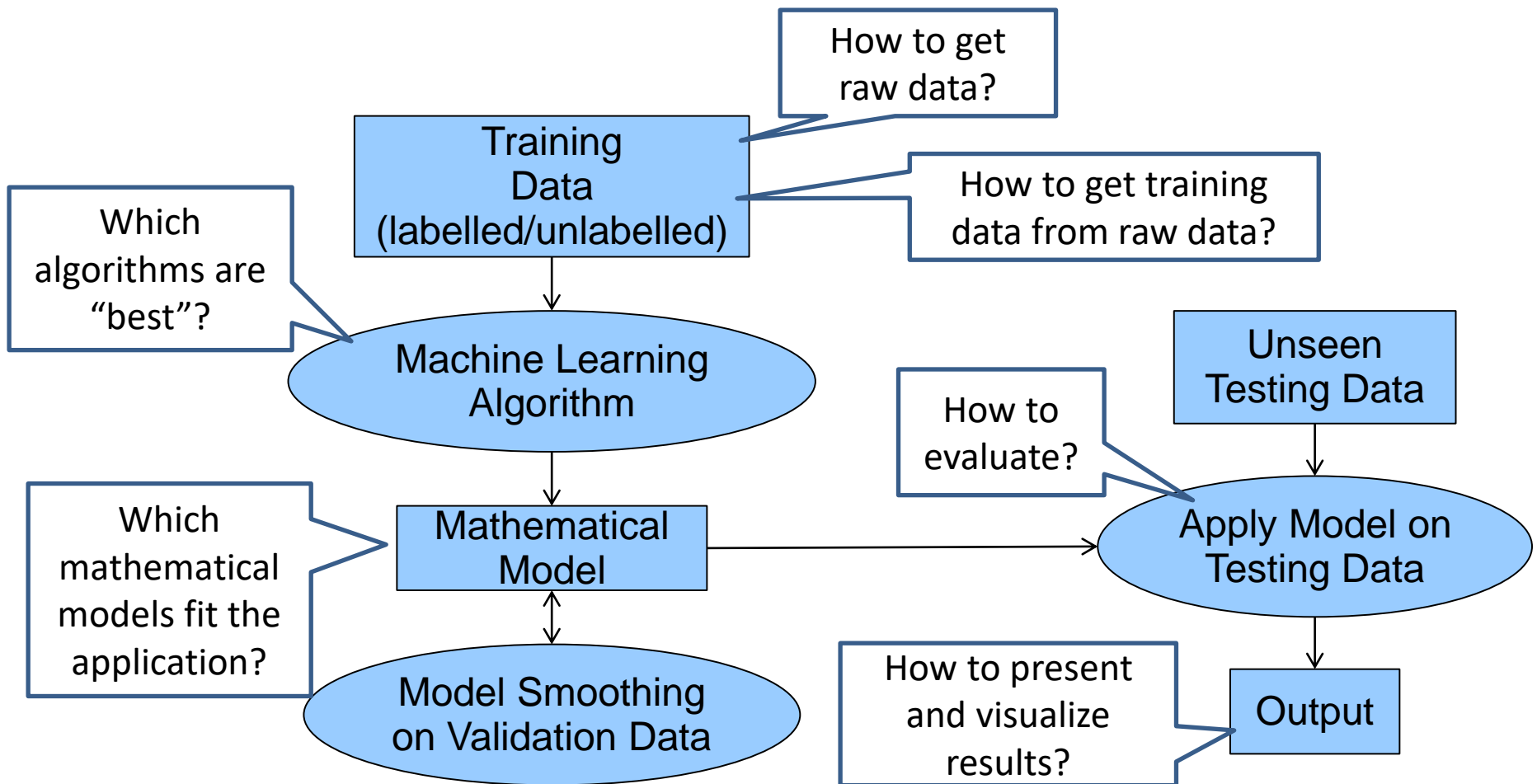
NTNU, Trondheim

Monday 30.10.2023

Today's Agenda

- “How to do Quantitative Empirical Research”:
 - A big topic, even if we restrict ourselves to CS, AI, or ML
 - With one hour at disposal, and trying not to rush, we have to focus on a certain key topics
- I will focus on quantitative empirical research in:
 - ML and BioAI – some key ideas
 - But I will try to paint a somewhat bigger picture than what you may have seen previously – emphasize workflow, different metrics, different perspectives, etc.
- Disclaimers:
 - Many subfield of AI and ML have their own evaluation methods or metrics – make sure you know and use them
 - In many areas, the quality of evaluation methods (metrics) are under debate
 - Critical evaluation of metrics and developing new metrics can be very valuable research topics
 - I don't drill down into the details of probability and statistics

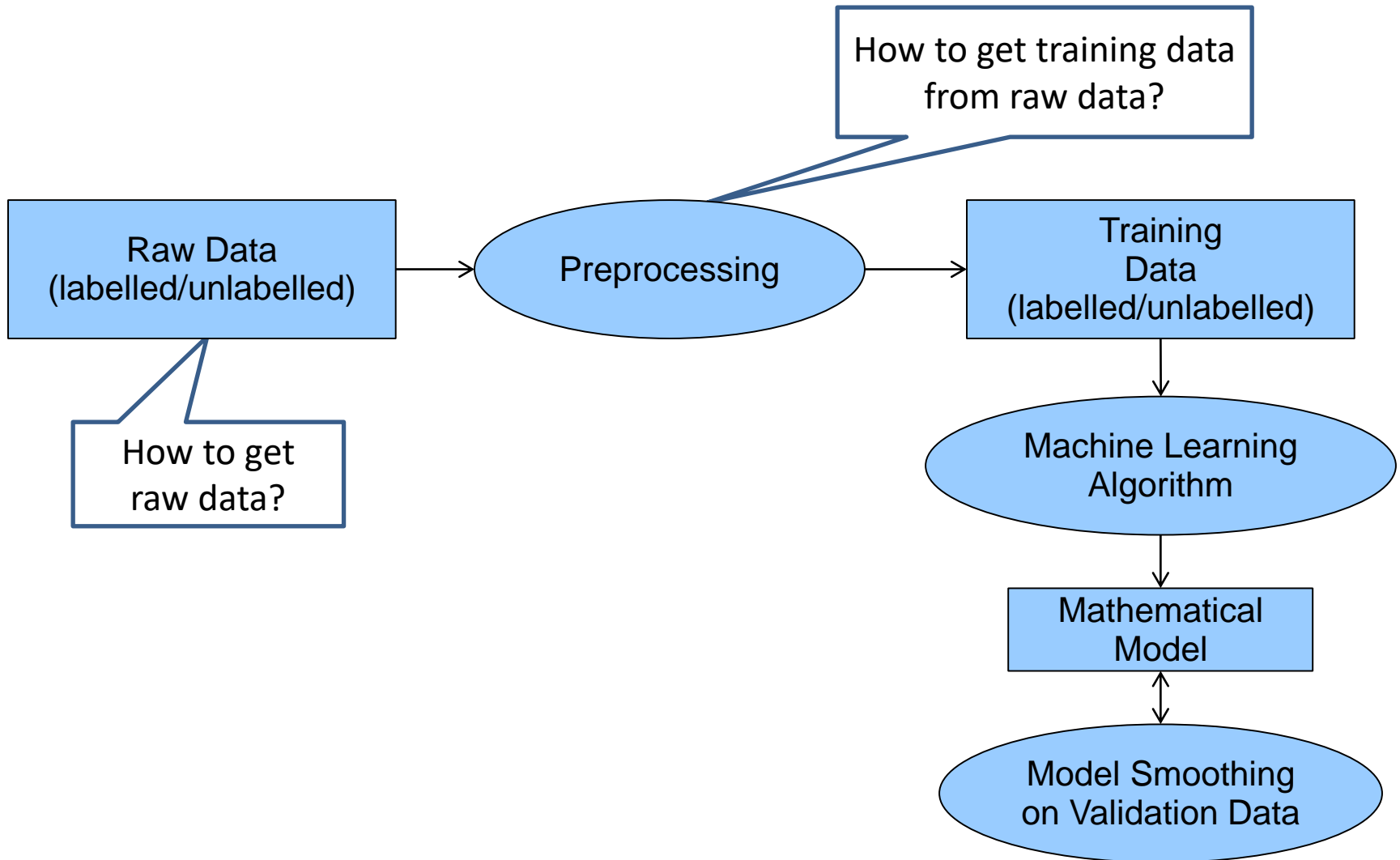
ML Pipeline – Some Questions



Data Sets

- Lot's of interesting data sets available:
 - Data is not the bottleneck is used to be just a few years ago
- Sign that you have a good data set:
 - A data set your sponsor, advisor, and you think is important
 - Competition or challenge data sets (recent or currently active)
 - New, exciting, timely, real-world, ... data set
- Be careful with data sets that:
 - Do not exist yet (this is more than a data-collection course)
 - Require in-depth understanding of a technical/scientific area you do not currently understand at all (too time-consuming?)
 - Are too simple (stay away from Statistics 101 data sets)
 - Do not exist in one of the commonly used file formats (there might not be time to write a complicated parser)
 - Have no documentation or support (you'll be on your own)
 - Are extremely large – unless you know how to handle this

From Raw Data to Training Data



Which Algorithms?

“Tribes” in machine learning (and AI?) [Domingos, 2015]:

- **Bayesians:** learning as inference using - Bayes rule, Bayesian networks, and probabilistic graphical models [Duda & Hart, 1973] [Pearl, 1988] [Jelinek, 1997][Darwiche, 2009] [Koller & Friedman, 2009] [Blake, 2011].
- **Symbolists:** intelligence as symbol manipulation [Newell & Simon, 1976] [Michalski et al., 1983] [Breiman et al., 1984] [Quinlan, 1992].
- **Analogizers:** learning by recognizing similarities [Boser et al., 1992] [Kolodner, 1993] [Cristianini & Shawe-Taylor, 2000].
- **Evolutionaries:** use methods from evolution and genetics - evolutionary algorithms, genetic algorithms, and genetic programming [Darwin, 1859] [Holland, 1975] [Goldberg, 1989].
- **Connectionists:** reverse engineer the brain [Werbos, 1974] [Rumelhart & McClelland, 1986] [Bengio, 2009] [Goodfellow et al., 2016].

Which algorithm(s) is (are) “best” depends on your project – data, goal, skills, resources, ...

Machine Learning Experimentation and Evaluation

Evaluation and Experimentation

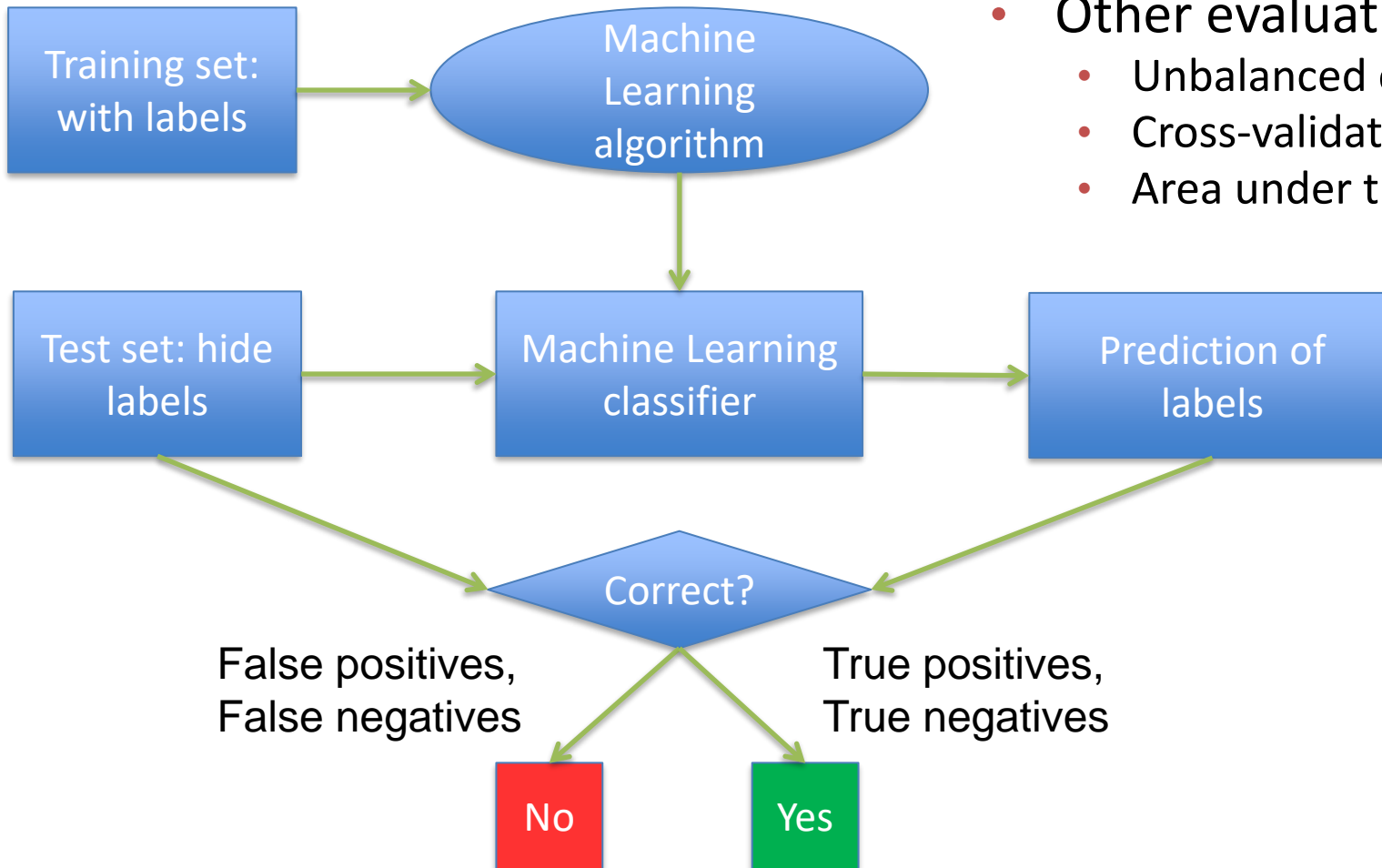
- Evaluation using data: Training, validation, testing:
 - “Simple”: Split original dataset into training data and test data, use test data to evaluate accuracy of machine learning model
 - “Complex”: Split original dataset into training data, validation data, and test data
- Evaluation baselines:
 - Other ML methods – “bakeoff”
 - Experimental results from literature
 - Comparison against theory
 - Evaluation by human experts
 - Comparison to results achieved by other software (simulation)
 - ...
- Evaluation depends on project type:
 - Application-oriented project: Models induced using different ML algorithms – “solve application problem using different tools”
 - Method-oriented project: Datasets from different application areas – “demonstrate generality of method across multiple data sets”

Evaluation

- Need a clear idea of what we are trying to achieve
- Should be able to connect the results of machine learning to the goal(s) of an organization (business)
- However, it is often difficult to measure the ultimate business goal(s), due to inadequate or complex data
 - We can measure a surrogate in such cases
 - Need to decide the surrogate through careful analysis
 - In machine learning, the surrogate is often the ML model:
 - Created from a training data set + prior knowledge
 - Evaluated on a testing data set
 - Here: focus on classifier as the ML model

Evaluation: Overview

- Focus on classifiers
- Evaluating classifiers: accuracy
- Other evaluation topics:
 - Unbalanced classes
 - Cross-validation
 - Area under the curve



Evaluating Binary Classifiers

- Assumption: binary (0/1, Yes/No, Positive/Negative) classification
- Positives and negatives - in machine learning terminology:
 - **Negatives** are the uninteresting outcomes
 - **Positives** are the outcomes of interest (sometimes few)
- We have the following 4 possibilities:
 - False Positives (FP): Test *incorrectly* reports a value as *positive*
 - True Positives (TP): Test *correctly* reports a value as *positive*
 - True Negatives (TN): Test *correctly* reports a value *negative*
 - False Negatives (FN): Test *incorrectly* reports a value as *negative*

Our aim is to reduce FPs and FNs. The number of FPs may dominate the number of FNs, but the cost of mistakes made on FNs may be higher. (More about this later.)

Confusion Matrix

- Confusion Matrix :
 - An $n \times n$ matrix for a classification problem with n classes
 - For binary classification: 2 x 2 confusion matrix
 - Main diagonal (**green**) contains the correct outputs of the classifier

		Predicted class	
		Positive (1)	Negative (0)
Actual class	Positive (1)	True positive	False negative
	Negative (0)	False positive	True negative

Comes from binary classifier

Comes from test data or "real world"

Evaluation Metric: Accuracy

- Define (based on the confusion matrix):
 - TPs: Number of true positives
 - TNs: Number of true negatives
 - FPs: Number of false positives
 - FNs: Number of false negatives
- Accuracy a - proportion of correct decisions

$$a = \frac{TPs+TNs}{TPs+TNs+FPs+FNs} = 1 - e$$

- A typical goal of machine learning is to maximize accuracy a and minimize error rate e

Beyond Accuracy

- Accuracy, and closely related metrics, are good starting points for evaluation
- However, there are some potential problems:
 1. Unbalanced classes
 2. Desire to use “test data” during training
 3. Sensitivity to varying parameters
 4. ...
- Below we study these problems in some detail, and sketch solutions

Accuracy and Unbalanced Classes

- Is plain accuracy sufficient to evaluate a model?
- *Perhaps not.* In classification problems where one class is rare, the class distribution becomes highly skewed
 - E.g.: credit card transactions
 - 100 transactions: 98 legitimate, 2 fraudulent
 - Classifier classifies all transactions as legitimate
 - Accuracy $a = 98/100 = 98\%$
 - Is this a good classifier?

Unequal Costs

- Simple classification accuracy makes no distinction between FPs and FNs
- In applications, the gravity of FPs versus FNs can vary significantly
- Examples:
 - In medical diagnosis: a FN (a disease was not caught) can be life threatening
 - In fraud detection: a FP (a transaction was flagged as fraudulent but was not) can affect customer relations and involve legal issues

Expected Value

- Expected value is the weighted average of all possible outcomes, where the weight is the probability of occurrence

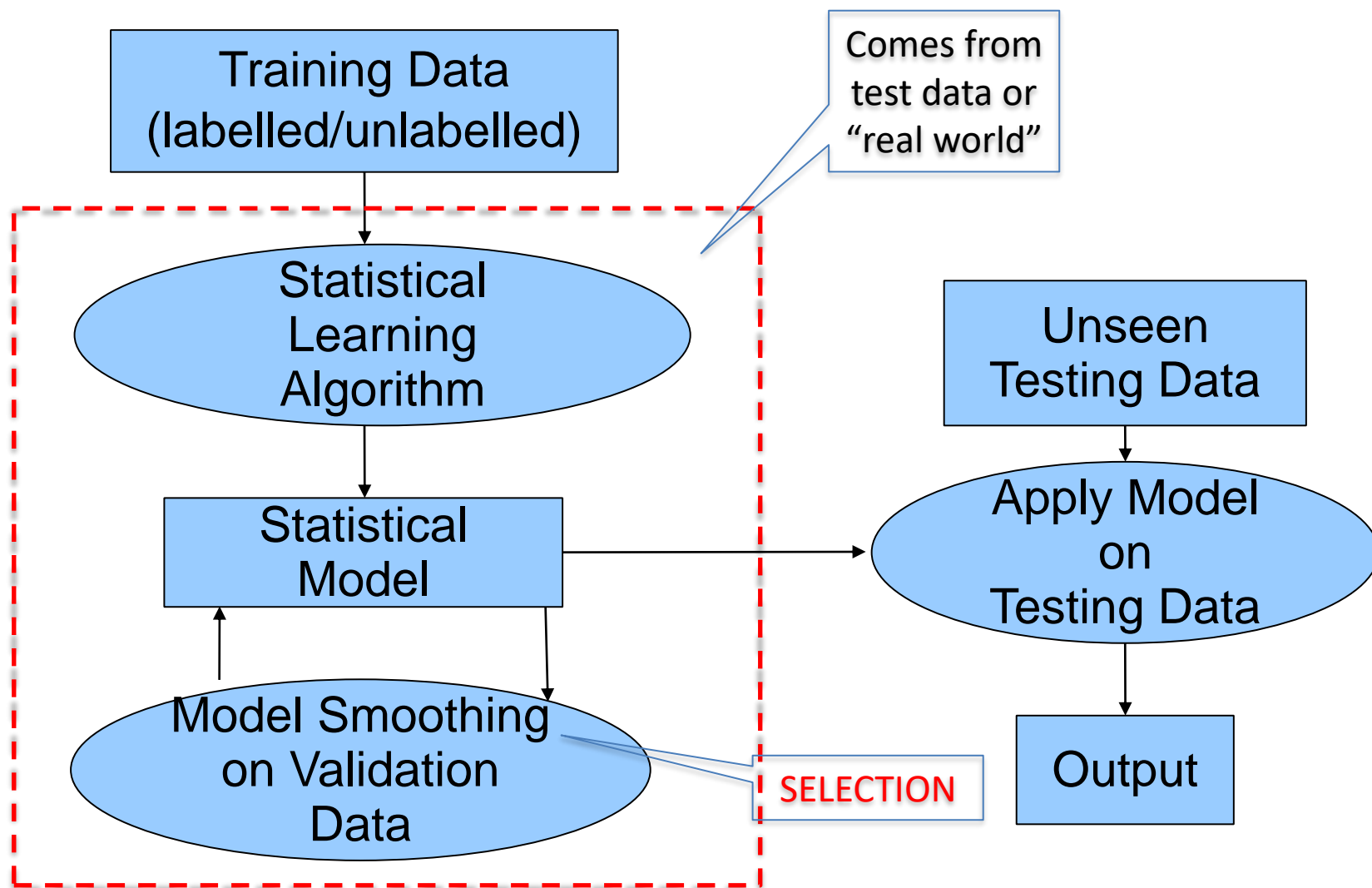
$$Ev = \sum P(o_j) * V(o_j)$$

- Where:
 - o_j is a possible decision outcome,
 - $P(o_j)$ is its probability, and
 - $V(o_j)$ is its value

Using Expected Value to Evaluate a Classifier

- How do we decide if a data driven decision is better than a decision taken intuitively?
- Expected value can – given information about outcomes, probabilities, and their values – be used to determine best decisions for a particular model
- Expected value aggregates all possible outcomes to decide whom to target or where to spend

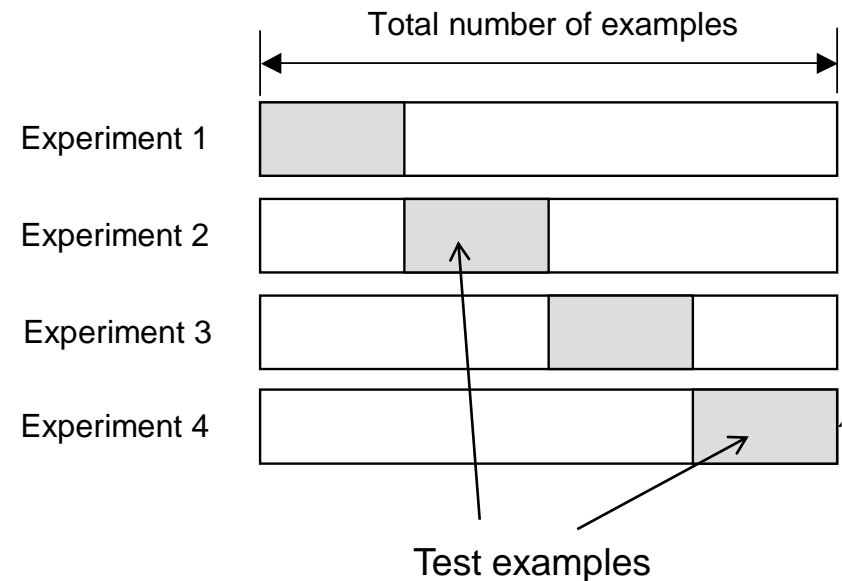
Cross-Validation



Cross-Validation: Details

- A model selection technique: assesses how the predictions will generalize to an independent dataset
- K-Fold Cross-validation:
 - Create a K-fold partition of the dataset
 - Perform K experiments as
 - Use K-1 folds for training and the remaining one for testing
 - Average error rate:

$$E = \frac{1}{K} \sum_{i=1}^K E_i$$

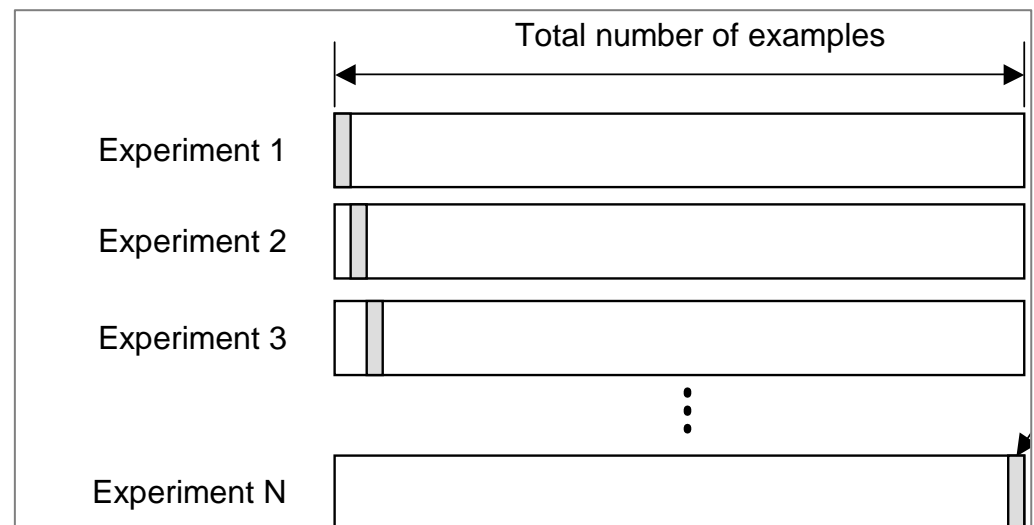


* http://research.cs.tamu.edu/prism/lectures/iss/iss_l13.pdf

Leave-One-Out Cross Validation

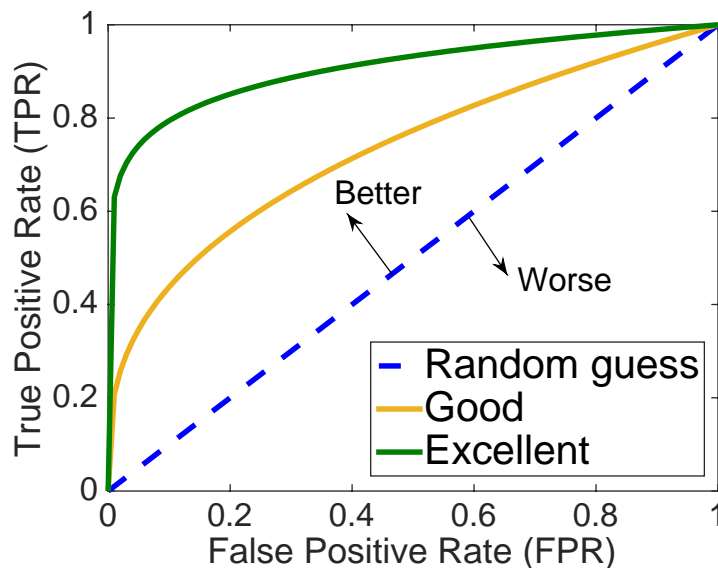
- Degenerate case of K-Fold Cross Validation
 - $K = \text{total number of examples } (N)$
- For a dataset with N examples, perform N experiments
 - Use $N-1$ examples for training and the remaining example for testing
 - Average error rate:

$$E = \frac{1}{N} \sum_{i=1}^N E_i$$



Receiver Operating Characteristic (ROC)

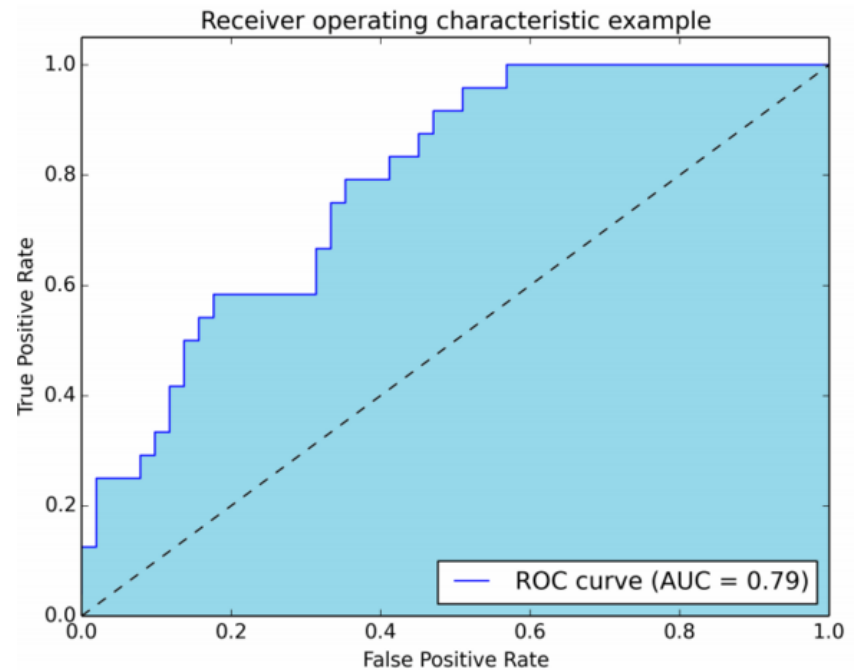
- True positive rate:
 - $TPR = TP / (TP + FN)$
- False positive rate:
 - $FPR = FP / (FP + TN)$
- ROC curve: TPR versus FPR – the plot represents the performance of a binary classifier as its discrimination threshold is varied
- ROC analysis provides tools to select possibly optimal models



	Truth: positive	Truth: negative
Predicted: positive	True positive	False positive
Predicted: negative	False negative	True negative

Area Under the Curve (AUC)

- Probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one
- Classification analysis: which model predicts the classes best?
 - Model with higher AUC
- Some research¹ shows: AUC is noisy as a classification measure



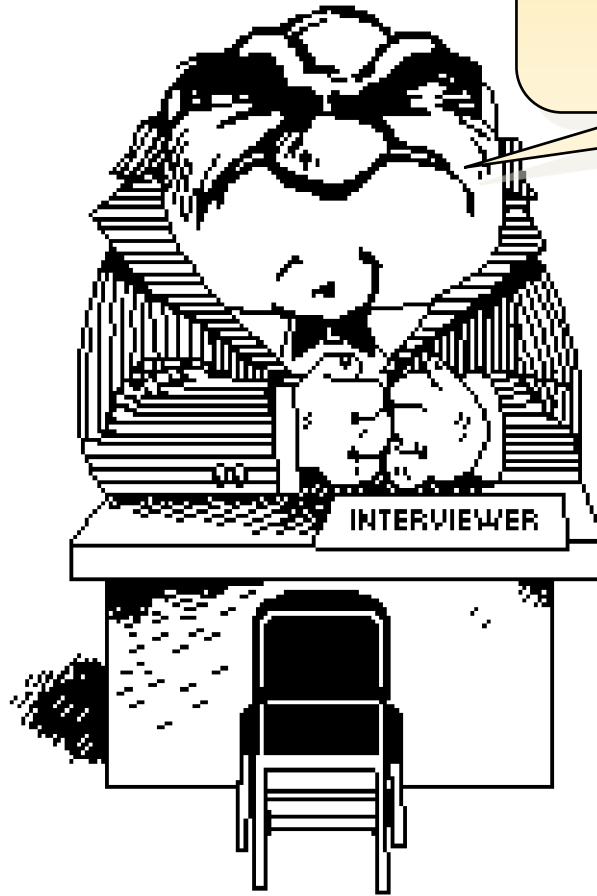
1. Hanczar, Blaise; Hua, Jianping; Sima, Chao; Weinstein, John; Bittner, Michael; and Dougherty, Edward R. (2010); Small-sample precision of ROC-related estimates, *Bioinformatics* 26 (6): 822–830

<http://stats.stackexchange.com/questions/132777/what-does-auc-stand-for-and-what-is-it>

Summary

- Accuracy, and closely related metrics, are good starting points for evaluation
- Issues related to accuracy include:
 1. Unbalanced classes:
 2. Desire to use “test data” during training: cross-validation
 3. Sensitivity to varying parameters
 4.
- Other evaluation problems: User studies, evaluation of “business impact,” ...

Questions?
Comments?



Evolutionary Computing



Chapter 9

Eiben & Smith: Introduction to Evolutionary Computing (Natural Computing Series), 2nd ed., 2015.

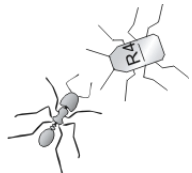
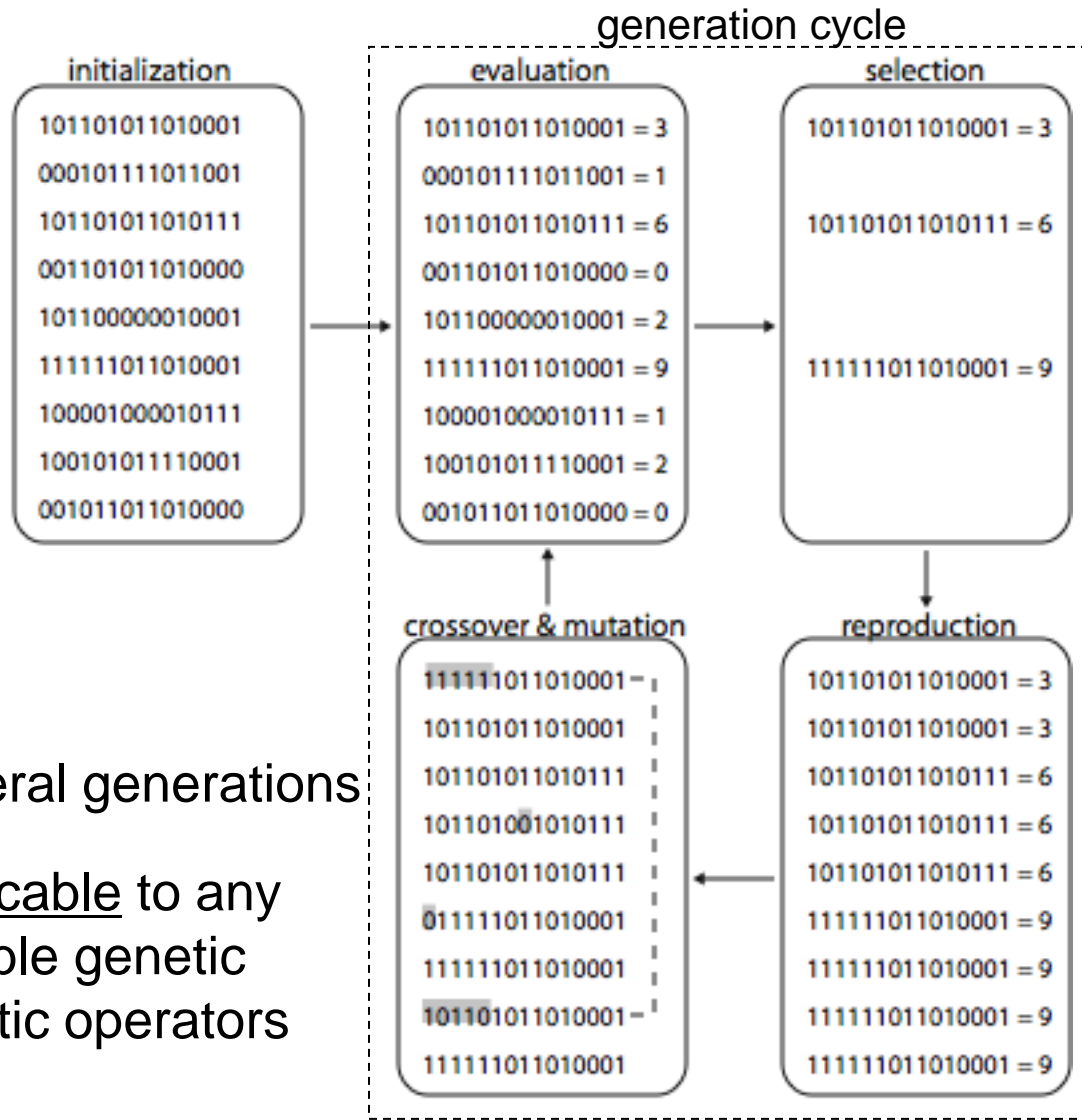
Evolutionary Algorithm

- Devise genetic representation
- Build a population
- Design a fitness function
- Choose selection method
- Choose crossover & mutation
- Choose data analysis method

Repeat generation cycle until:

- maximum fitness value is found
- solution found is good enough
- no fitness improvement for several generations

Evolutionary algorithms are applicable to any problem domain as long as suitable genetic representation, fitness, and genetic operators are chosen.



Chapter 9:

Working with Evolutionary Algorithms

- Experiment design
- Algorithm design
- Test problems
- Measurements and statistics
- Some tips and summary

Experimentation

- Has a **goal** or goals
- Involves **algorithm** design and implementation
- Needs **problem(s)** to run the algorithm(s) on
- Amounts to **running** the algorithm(s) on the problem(s)
- Delivers **measurement data**, the results
- Is concluded with **evaluating** the results in the light of the given goal(s)
- Is often **documented**

Experimentation: Varying Goals

- Get a good solution for a given problem
- Show that EC is applicable in a (new) problem domain
- Show that *my_EA* is better than *benchmark_EA*
- Show that EAs outperform traditional algorithms (sic!)
- Find best setup for parameters of a given algorithm
- Understand algorithm behavior (e.g. pop dynamics)
- See how an EA scales-up with problem size
- See how performance is influenced by parameters
- ...

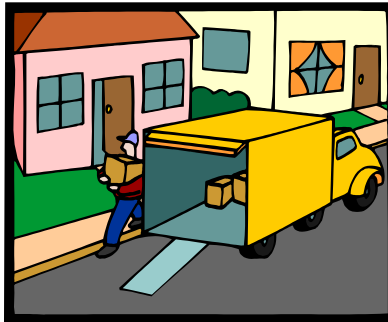
Perspectives of goals

- **Design** perspective:
find a **very good** solution at least **once**
- **Production** perspective:
find a **good** solution at **almost every run**
- **Publication** perspective:
must meet **scientific standards**
- **Application** perspective:
good enough is **good enough**

These perspectives have very different implications on evaluating the results (yet often left implicit)

Example: Production Perspective

- Optimising Internet shopping delivery route
 - Different destinations each day
 - Limited time to run algorithm each day
 - Must *always* be *reasonably* good route in limited time



Example: Design Perspective

- Optimising spending on improvements to national road network
 - Total cost: billions of Euro
 - Computing costs negligible
 - Six months to run algorithm on hundreds computers
 - Many runs possible
 - Must produce *very good result just once*



Algorithm design

- Design a representation
- Design a way of mapping a genotype to a phenotype
- Design a way of evaluating an individual
- Design suitable mutation operator(s)
- Design suitable recombination operator(s)
- Decide how to select individuals to be parents
- Decide how to select individuals for the next generation (how to manage the population)
- Decide how to start: initialization method
- Decide how to stop: termination criterion

Getting Problem Instances (1/3)

- Testing on **real data**
- Advantages:
 - Results could be considered as very relevant viewed from the application domain (data supplier)
- Disadvantages
 - Can be over-complicated
 - Can be few available sets of real data
 - May be commercial sensitive – difficult to publish and to allow others to compare
 - Results are hard to generalize

Getting Problem Instances (2/3)

- Standard data sets in problem **repositories**, e.g.:
 - OR-Library
<http://www.ms.ic.ac.uk/info.html>
 - UCI Machine Learning Repository
www.ics.uci.edu/~mlearn/MLRepository.html
- Advantage:
 - Well-chosen problems and instances (hopefully)
 - Much other work on these → results comparable
- Disadvantage:
 - Not real – might miss crucial aspect
 - Algorithms get tuned for popular test suites

Getting Problem Instances (3/3)

- **Problem instance generators** produce simulated data for given parameters, e.g.:
 - GA/EA Repository of Test Problem Generators
<http://www.cs.uwyo.edu/~wspears/generators.html>
- **Advantage:**
 - Allow very systematic comparisons for they
 - can produce many instances with the same characteristics
 - enable gradual traversal of a range of characteristics (hardness)
 - Can be shared allowing comparisons with other researchers
- **Disadvantage**
 - Not real – might miss crucial aspect
 - Given generator might have hidden bias

Basic rules of experimentation

- EAs are stochastic →
never draw any conclusion from a single run
 - perform sufficient number of independent runs
 - use statistical measures (averages, standard deviations)
 - use statistical tests to assess reliability of conclusions
- EA experimentation is about comparison →
always do a fair competition
 - use the same amount of resources for the competitors
 - try different comp. limits (to coop with turtle/hare effect)
 - use the same performance measures

Things to measure - metrics

Many different ways. Examples:

- Average result in given time
- Average time for given result
- Proportion of runs within % of target
- Best result over n runs
- Amount of computing required to reach target in given time with % confidence
- ...

What time units do we use?

- Elapsed time?
 - Depends on computer, network, etc...
- CPU Time?
 - Depends on skill of programmer, implementation, etc...
- Generations?
 - Difficult to compare when parameters like population size change
- Evaluations?
 - Evaluation time could depend on algorithm, e.g. direct vs. indirect representation

Measures

- Performance measures (off-line)
 - Efficiency (alg. speed)
 - CPU time
 - No. of steps, i.e., generated points in the search space
 - Effectivity (alg. quality)
 - Success rate
 - Solution quality at termination
- “Working” measures (on-line)
 - Population distribution (genotypic)
 - Fitness distribution (phenotypic)
 - Improvements per time unit or per genetic operator
 - ...

Performance measures

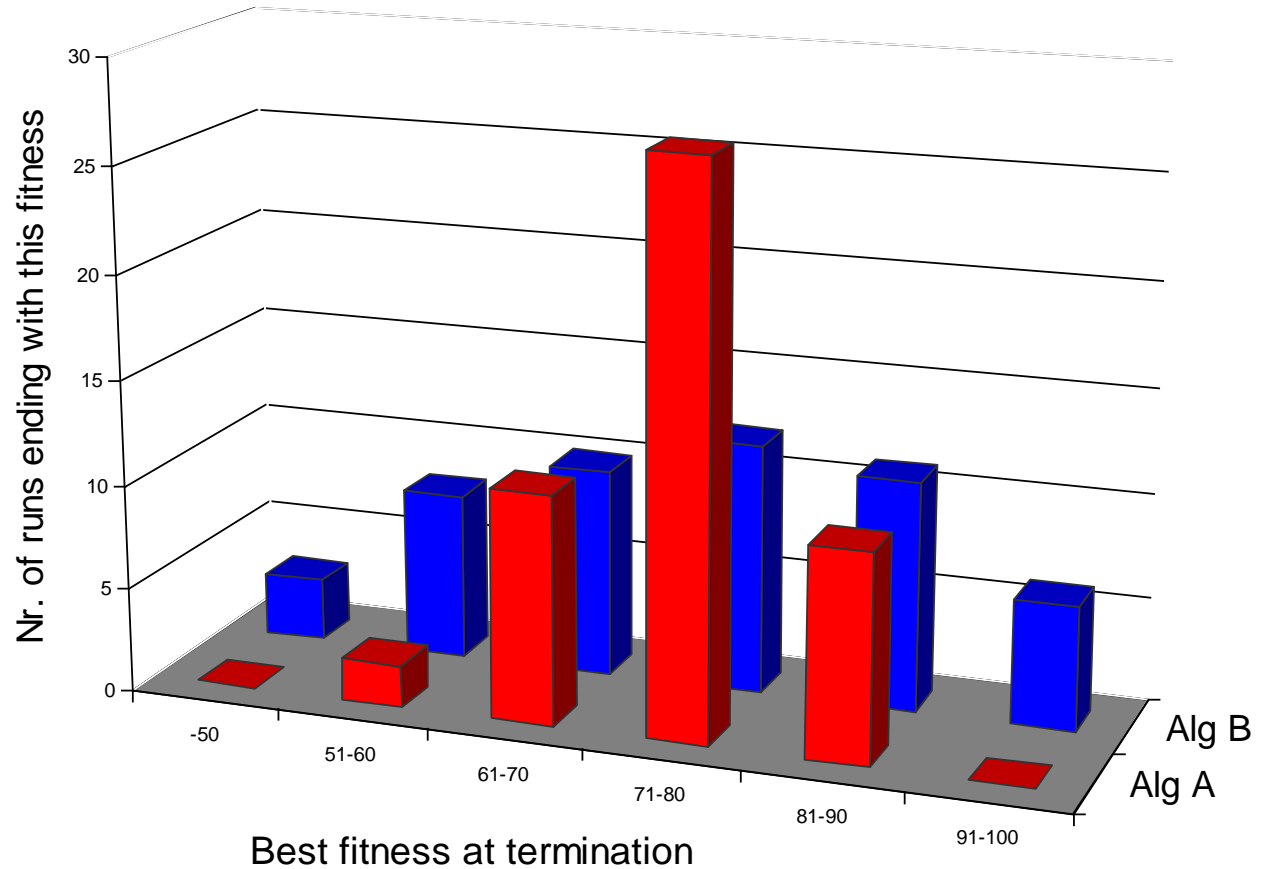
- No. of generated points in the search space
= no. of fitness evaluations
(don't use no. of generations!)
- **AES: average no. of evaluations to solution**
- **SR: success rate** = % of runs finding a solution
(individual with acceptable quality / fitness)
- **MBF: mean best fitness** at termination, i.e., best per run,
mean over a set of runs

Fair experiments

- **Basic rule: use the same computational limit for each competitor**
- Allow each EA the same no. of evaluations, but
 - Beware of hidden labour, e.g. in heuristic mutation operators
 - Beware of possibly fewer evaluations by smart operators
- EA vs. heuristic: allow the same no. of steps:
 - Defining “step” is crucial, might imply bias!
 - Scale-up comparisons eliminate this bias

Example: off-line performance measure evaluation

Which algorithm is better?
Why?
When?



Statistical Comparisons and Significance

- Algorithms are stochastic, results have element of “luck”
- If a claim is made “Mutation A is better than mutation B”, need to show statistical significance of comparisons
- Fundamental problem: two series of samples (random drawings) from the SAME distribution may have DIFFERENT averages and standard deviations
- Tests can show if the differences are significant or not
 - T-test: Are two sets of experimental results significantly different?
 - F-test: Is there statistical difference between experimental results of three or more algorithms?

See Appendix B of Simon’s textbook for more on test these tests, and their application to EAs.

Statistical tests

- T-test assumes:
 - Data taken from continuous interval or close approximation
 - Normal distribution
 - Similar variances for too few data points
 - Similar sized groups of data points
- Other tests:
 - Wilcoxon – preferred to t-test where numbers are small or distribution is not known.
 - F-test – tests if two samples have different variances.

Example: problem setting

- I invented myEA for problem X
- Looked and found 3 other EAs and a traditional benchmark heuristic for problem X in the literature
- Asked myself when and why is myEA better

Example: experiments

- Found/made problem instance generator for problem X with 2 parameters:
 - n (problem size)
 - k (some problem specific indicator)
- Selected 5 values for k and 5 values for n
- Generated 100 problem instances for all combinations
- Executed all alg's on each instance 100 times (benchmark was also stochastic)
- Recorded AES, SR, MBF values w/ same comp. limit (AES for benchmark?)
- Put my program code and the instances on the Web

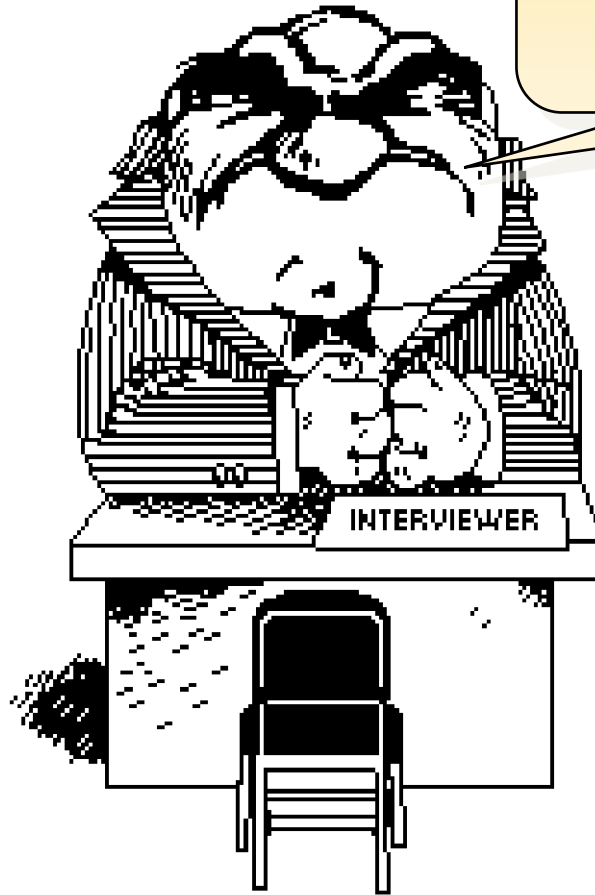
Example: evaluation

- Arranged results “in 3D” (n, k) + performance (with special attention to the effect of n , as for scale-up)
- Assessed statistical significance of results
- Found the niche for myEA:
 - Weak in ... cases, strong in - - - cases, comparable otherwise
 - Thereby I answered the “when question”
- Analyzed the specific features and the niches of each algorithm thus answering the “why question”
- Learned a lot about problem X and its solvers
- Achieved generalizable results, or at least claims with well-identified scope based on solid data
- Facilitated reproducing my results → further research

Wrapping up: Some tips

- **Be organized**
- Decide what you want & define appropriate measures
- Choose test problems carefully
- Make an experiment plan (estimate time when possible)
- Perform sufficient number of runs
- Keep all experimental data (never throw away anything)
- Use good statistics (“standard” tools from Web, MS, R)
- Present results well (figures, graphs, tables, ...)
- Watch the scope of your claims
- Aim at generalizable results
- Publish code for reproducibility of results (if applicable)
- Publish data for external validation (open science)

Questions?
Comments?



BACKUP

Data Preprocessing

- Metadata: Information about data set and its attributes
- Statistics: Mean, std. dev., outliers, clusters, correlation,...
- Missing values and data cleansing
- Normalization: satisfy statistical and/or visualization constraints
- Continuous versus discrete:
 - Segmentation and discretization: continuous to discrete
 - Nominal to ordinal mapping: discrete to continuous
- Sampling and sub-setting
- Dimension reduction: reduce to smaller number of dimensions
- Aggregation and summarization
- Smoothing and filtering: signal processing techniques
- ...

If data pre-processing is performed, it is often important to (1) clearly indicate so and (2) provide drill-down capability.

Feature Engineering

- Scenarios for features:
 - Many, independent, predictive features: Easy learning
 - Few, dependent, non-predictive features: Hard learning
- Applied machine learning project:
 - Much (most?) time might be spent on feature engineering
- Feature engineering is typically application-specific
 - Feature construction is often manual
 - Approaches to features selection:
 - Filter: First feature selection, then machine learning
 - Wrapper: Iterate between feature selection and machine learning
- Towards automation of feature engineering
 - Holy grail of ML: Automated construction of features
 - Traditional:
 - Generate: Feature construction
 - Test: Feature selection

Machine Learning: From the Fringe to the Center

The central role of learning

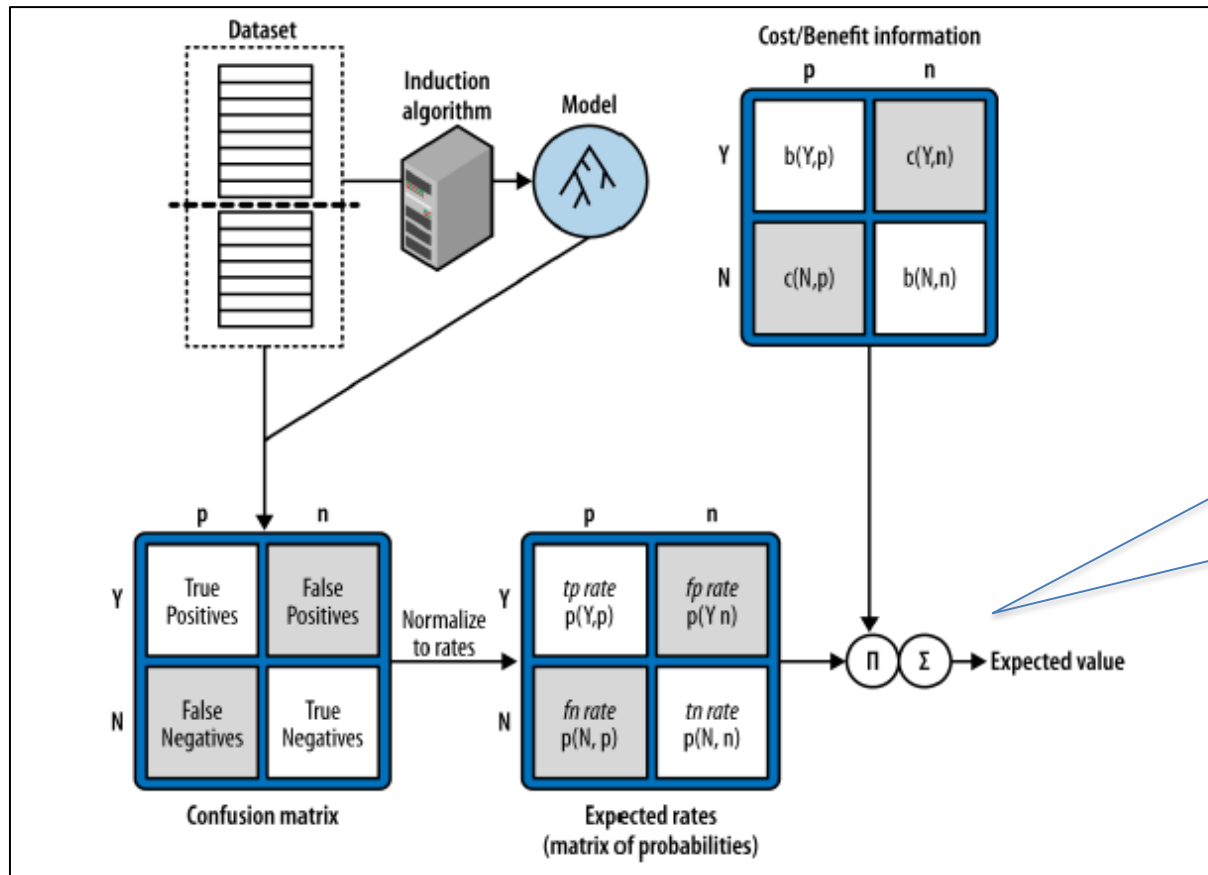
Although researchers in artificial intelligence and psychology have long recognized the importance of learning, this topic has not always been the central focus of these fields. In the first years of AI, considerable attention was given to learning issues, but as pattern recognition and AI developed separate identities, learning research became associated with the former while the latter concentrated on problems of representation and performance. A similar phenomenon occurred in psychology. The behaviorist paradigm was almost exclusively concerned with learning phenomena, but as information processing psychology gained in popularity, psychologists turned their sights towards memory and performance phenomena and all but abandoned efforts to explain the learning process.

However, the past five years have seen a resurgence of interest in learning within both artificial intelligence and cognitive psychology. This has resulted partly from dissatisfaction with pure performance models of intelligence. One of the major insights of both fields has been that, except in the simplest domains, intelligent behavior requires significant knowledge of those domains. Although this insight has led to successful applied AI systems and to accurate psychological models of domain-specific performance, it has not led to systems or theories having any great degree of generality. By refocusing their efforts on learning, many researchers hope to discover more general principles of intelligence. In the case of psychology, such principles would lead to more encompassing theories of human behavior that move beyond particular domains. In the case of applied AI, general learning methods might let one automate the construction of knowledge-intensive systems, saving man-years of effort for each application area.

Pat Langley's Editorial in the "Machine Learning" journal's Inaugural Issue, 1986.

Feature Engineering

Expected Value Calculation: Details



When classifying, maximize expected value instead of posterior probability.

From Provost & Fawcett, p. 197.

Metrics from Bio-Medicine

- Specificity = True positive rate = $\frac{TP}{TP + FN}$
- Sensitivity = True negative rate = $\frac{TN}{TN + FP}$

Metrics from Information Retrieval

- Precision = $\frac{TP}{TP + FP}$, commonly used in information retrieval:

- Eg:
$$\frac{\text{NumberOf RelevantDocuments Retrieved}}{\text{TotalNumberOfDocuments Retrieved}}$$

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

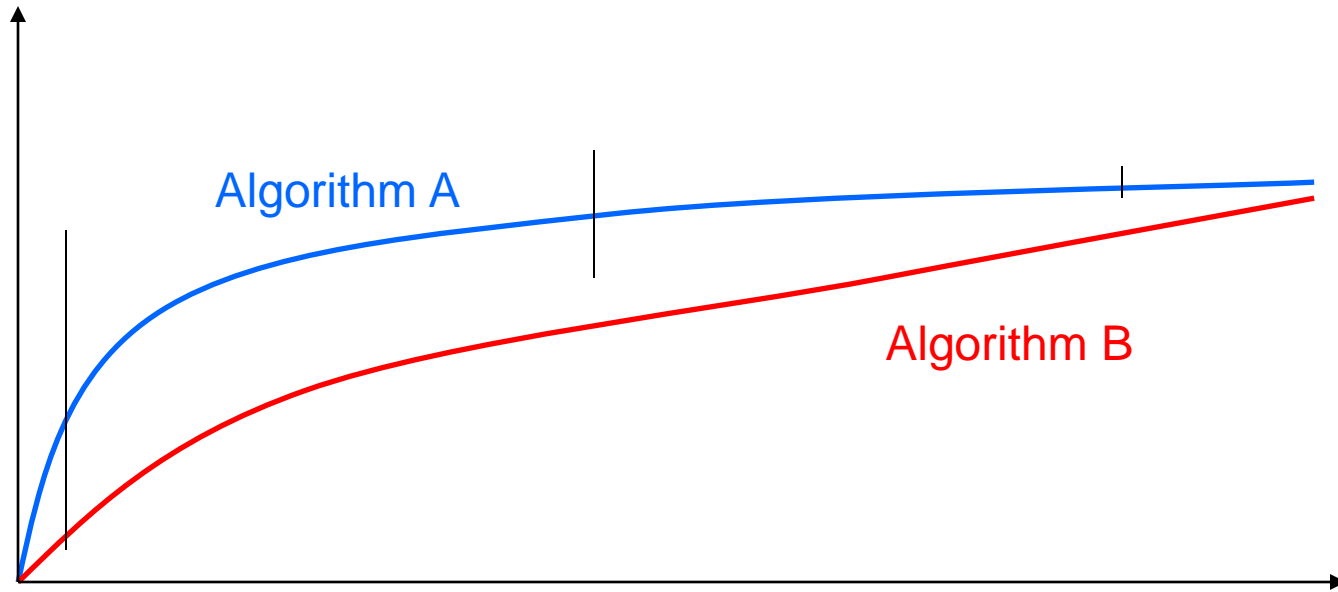
- Eg:
$$\frac{\text{NumberOf RelevantDocuments Retrieved}}{\text{TotalNumberOf RelevantDocuments}}$$

- F-Measure = harmonic mean of precision and recall

$$= 2 * \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

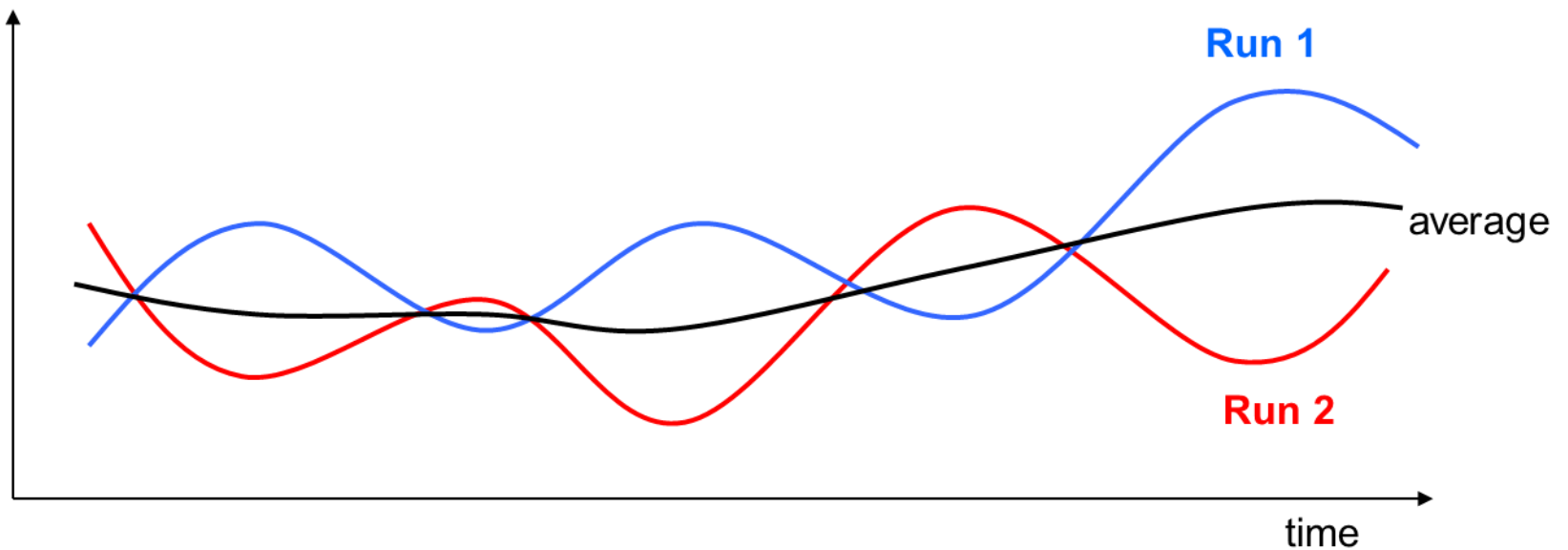
Example: on-line performance measure evaluation

Populations mean (best) fitness



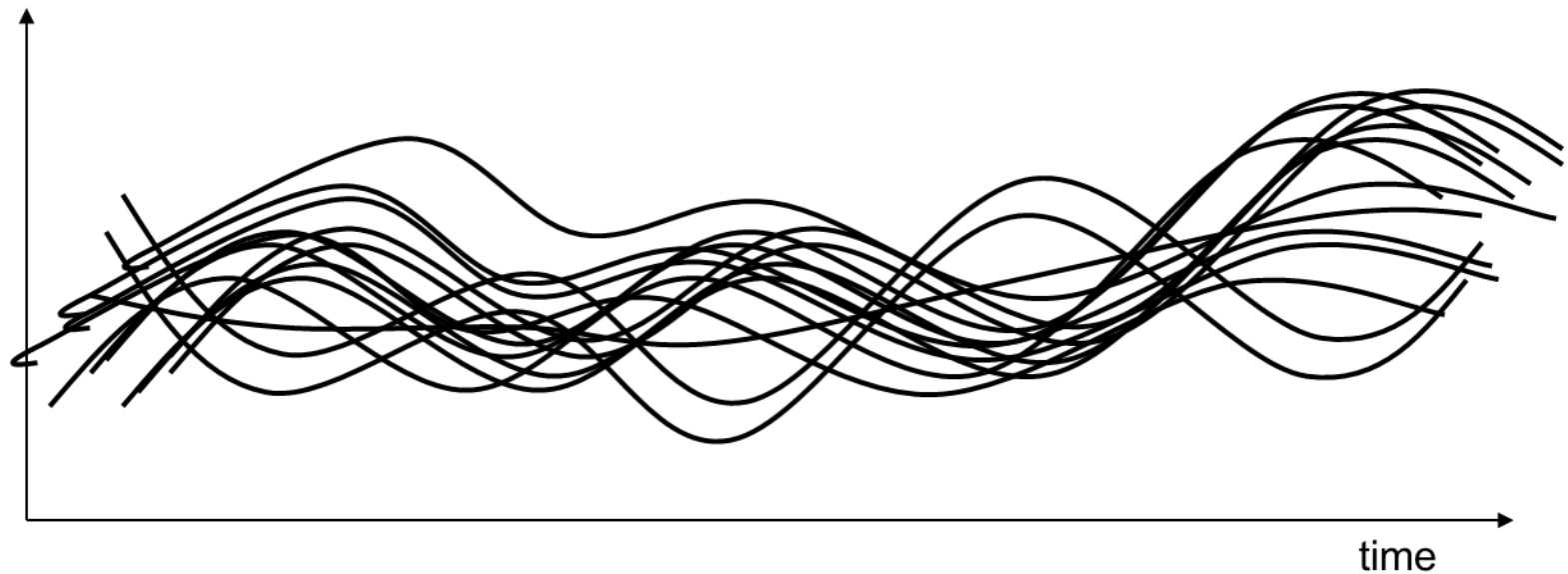
Which algorithm is better? Why? When?

Example: averaging on-line measures



Averaging can “choke” interesting information

Example: overlaying on-line measures



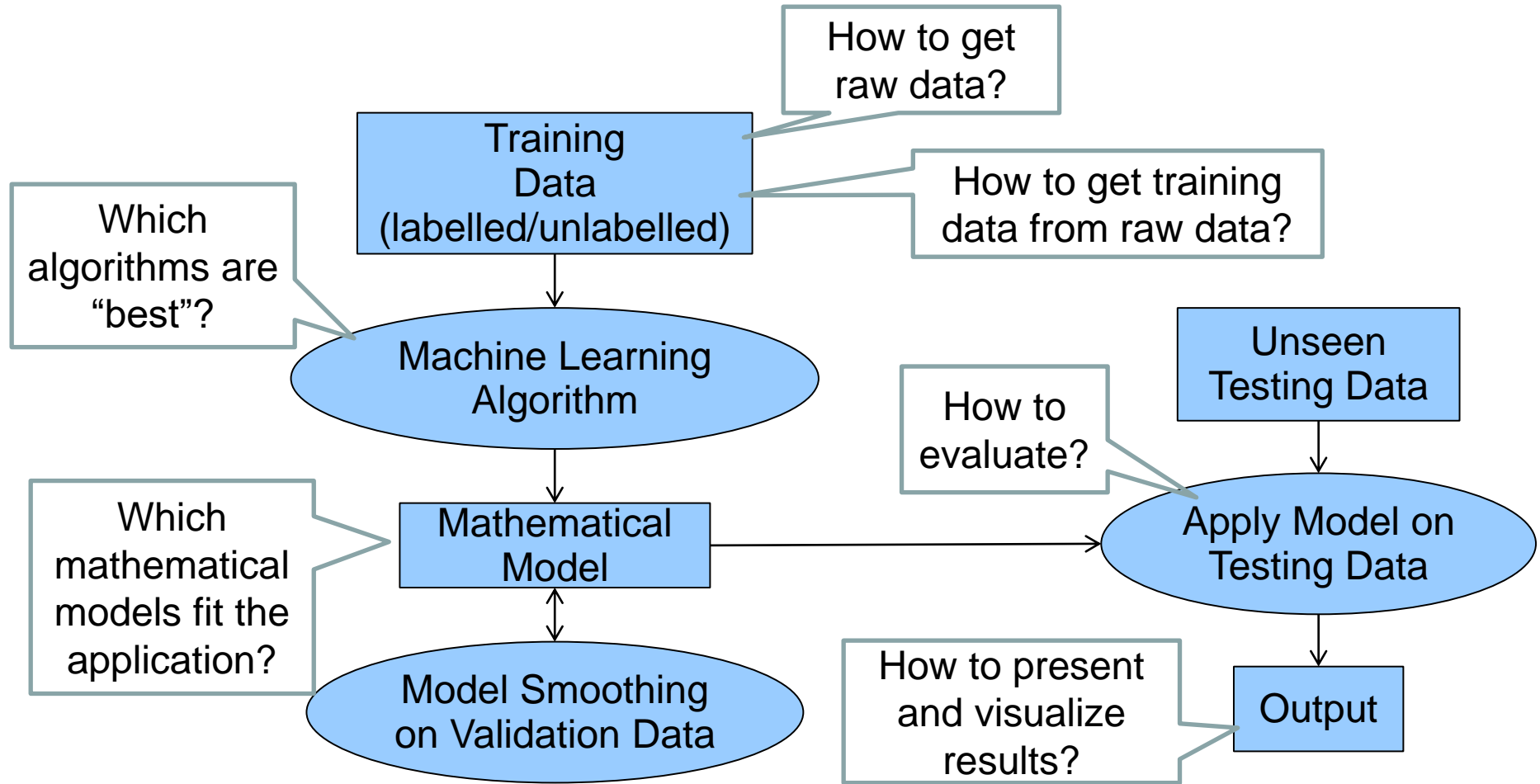
Overlay of curves can lead to very “cloudy” figures

ML Pipeline

- Data: Assume that data is already available and structured
- Data preprocessing:
 - Identification of parts of data set(s) to be analyzed
 - Integration, cleaning, warehousing, feature construction, feature selection, ...
- Machine learning/Data mining: Analyzing data to produce a model
- Pattern (model) evaluation: Evaluating accuracy, robustness, stability, precision, ... of resulting model
- Presentation and visualization: Present resulting model, perhaps in combination with data

Machine Learning Experimentation and Evaluation

ML Pipeline – Some Questions



Which Algorithms?

“Tribes” in machine learning (and AI?) [Domingos, 2015]:

- **Bayesians:** learning as inference using - Bayes rule, Bayesian networks, and probabilistic graphical models [Duda & Hart, 1973] [Pearl, 1988] [Jelinek, 1997][Darwiche, 2009] [Koller & Friedman, 2009] [Blake, 2011].
- **Symbolists:** intelligence as symbol manipulation[Newell & Simon, 1976] [Michalski et al., 1983] [Breiman et al., 1984] [Quinlan, 1992].
- **Analogizers:** learning by recognizing similarities [Boser et al., 1992] [Kolodner, 1993] [Cristianini & Shawe-Taylor, 2000].
- **Evolutionaries:** use methods from evolution and genetics - evolutionary algorithms, genetic algorithms, and genetic programming [Darwin, 1859] [Holland, 1975] [Goldberg, 1989].
- **Connectionists:** reverse engineer the brain [Werbos, 1974] [Rumelhart & McClelland, 1986] [Bengio, 2009] [Goodfellow et al., 2016].

Which algorithm(s) is (are) “best” depends on your project – data, goal, skills, resources, ...

ML: From the Fringe to the Center

The central role of learning

Although researchers in artificial intelligence and psychology have long recognized the importance of learning, this topic has not always been the central focus of these fields. In the first years of AI, considerable attention was given to learning issues, but as pattern recognition and AI developed separate identities, learning research became associated with the former while the latter concentrated on problems of representation and performance. A similar phenomenon occurred in psychology. The behaviorist paradigm was almost exclusively concerned with learning phenomena, but as information processing psychology gained in popularity, psychologists turned their sights towards memory and performance phenomena and all but abandoned efforts to explain the learning process.

However, the past five years have seen a resurgence of interest in learning within both artificial intelligence and cognitive psychology. This has resulted partly from dissatisfaction with pure performance models of intelligence. One of the major insights of both fields has been that, except in the simplest domains, intelligent behavior requires significant knowledge of those domains. Although this insight has led to successful applied AI systems and to accurate psychological models of domain-specific performance, it has not led to systems or theories having any great degree of generality. By refocusing their efforts on learning, many researchers hope to discover more general principles of intelligence. In the case of psychology, such principles would lead to more encompassing theories of human behavior that move beyond particular domains. In the case of applied AI, general learning methods might let one automate the construction of knowledge-intensive systems, saving man-years of effort for each application area.

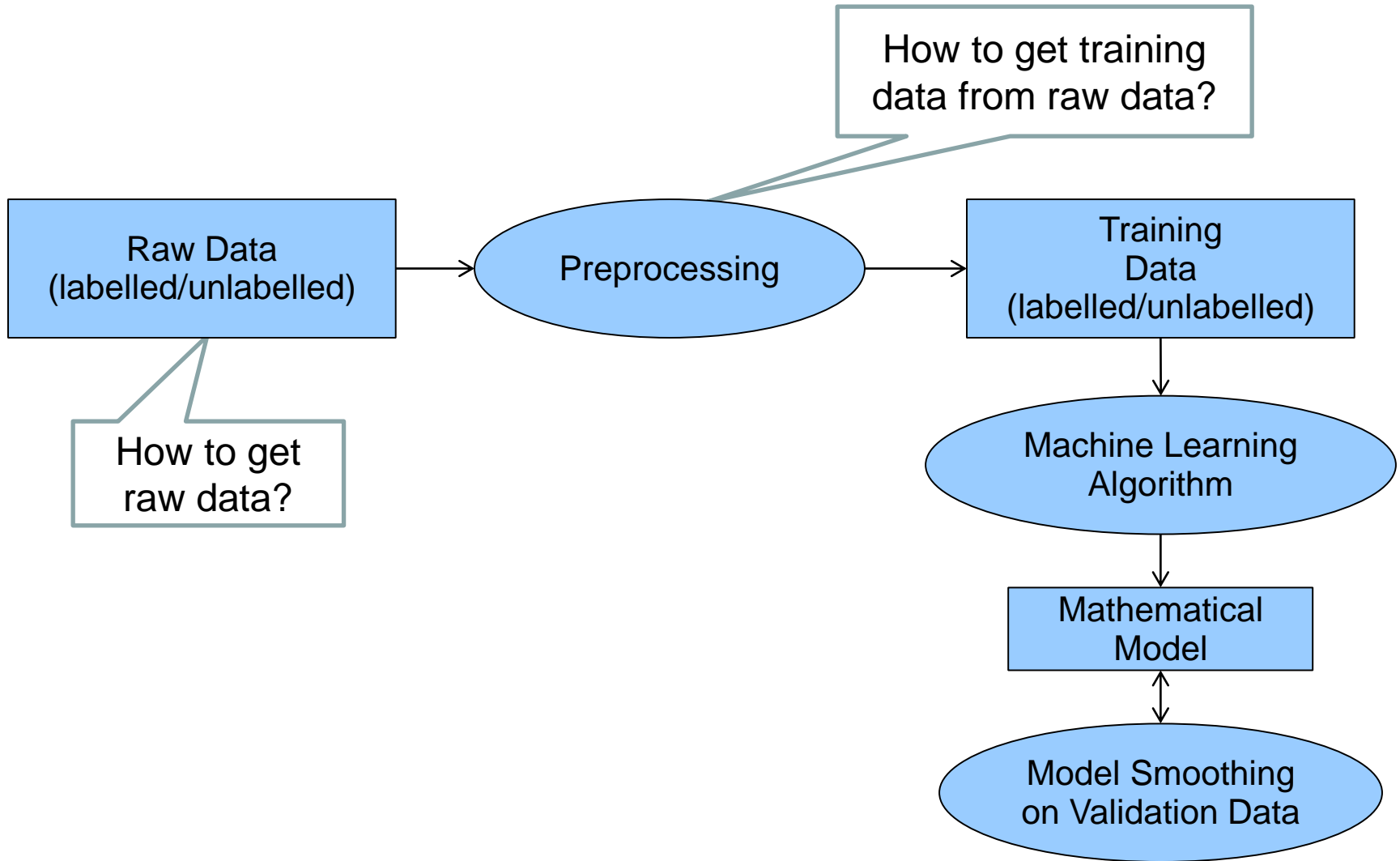
Pat Langley's Editorial in the "Machine Learning" journal's Inaugural Issue, 1986.

Feature Engineering

Data Sets

- Lot's of interesting data sets available:
 - Data is not the bottleneck is used to be just a few years ago
- Sign that you have a good data set:
 - A data set your sponsor, advisor, and you think is important
 - Competition or challenge data sets (recent or currently active)
 - New, exciting, timely, real-world, ... data set
- Be careful with data sets that:
 - Do not exist yet (unless you're in a data-collection project or course)
 - Require in-depth understanding of a technical/scientific area you do not currently understand at all (too time-consuming?)
 - Are too simple (stay away from Statistics 101 data sets)
 - Do not exist in one of the commonly used file formats (there might not be time to write a complicated parser)
 - Have no documentation or support (you'll be on your own)
 - Are extremely large – unless you know how to handle this

From Raw Data to Training Data



Data Preprocessing

- Metadata: Information about data set and its attributes
- Statistics: Mean, std. dev., outliers, clusters, correlation,...
- Missing values and data cleansing
- Normalization: satisfy statistical and/or visualization constraints
- Continuous versus discrete:
 - Segmentation and discretization: continuous to discrete
 - Nominal to ordinal mapping: discrete to continuous
- Sampling and sub-setting
- Dimension reduction: reduce to smaller number of dimensions
- Aggregation and summarization
- Smoothing and filtering: signal processing techniques
- ...

If data pre-processing is performed, it is often important to (1) clearly indicate so and (2) provide drill-down capability.

Feature Engineering

- Scenarios for features:
 - Many, independent, predictive features: Easy learning
 - Few, dependent, non-predictive features: Hard learning
- Applied machine learning project:
 - Much (most?) time might be spent on feature engineering
- Feature engineering is typically application-specific
 - Feature construction is often manual
 - Approaches to features selection:
 - Filter: First feature selection, then machine learning
 - Wrapper: Iterate between feature selection and machine learning
- Towards automation of feature engineering
 - Holy grail of ML: Automated construction of features
 - Traditional:
 - Generate: Feature construction
 - Test: Feature selection

Evaluation and Experimentation

• Evaluation and Experimentation

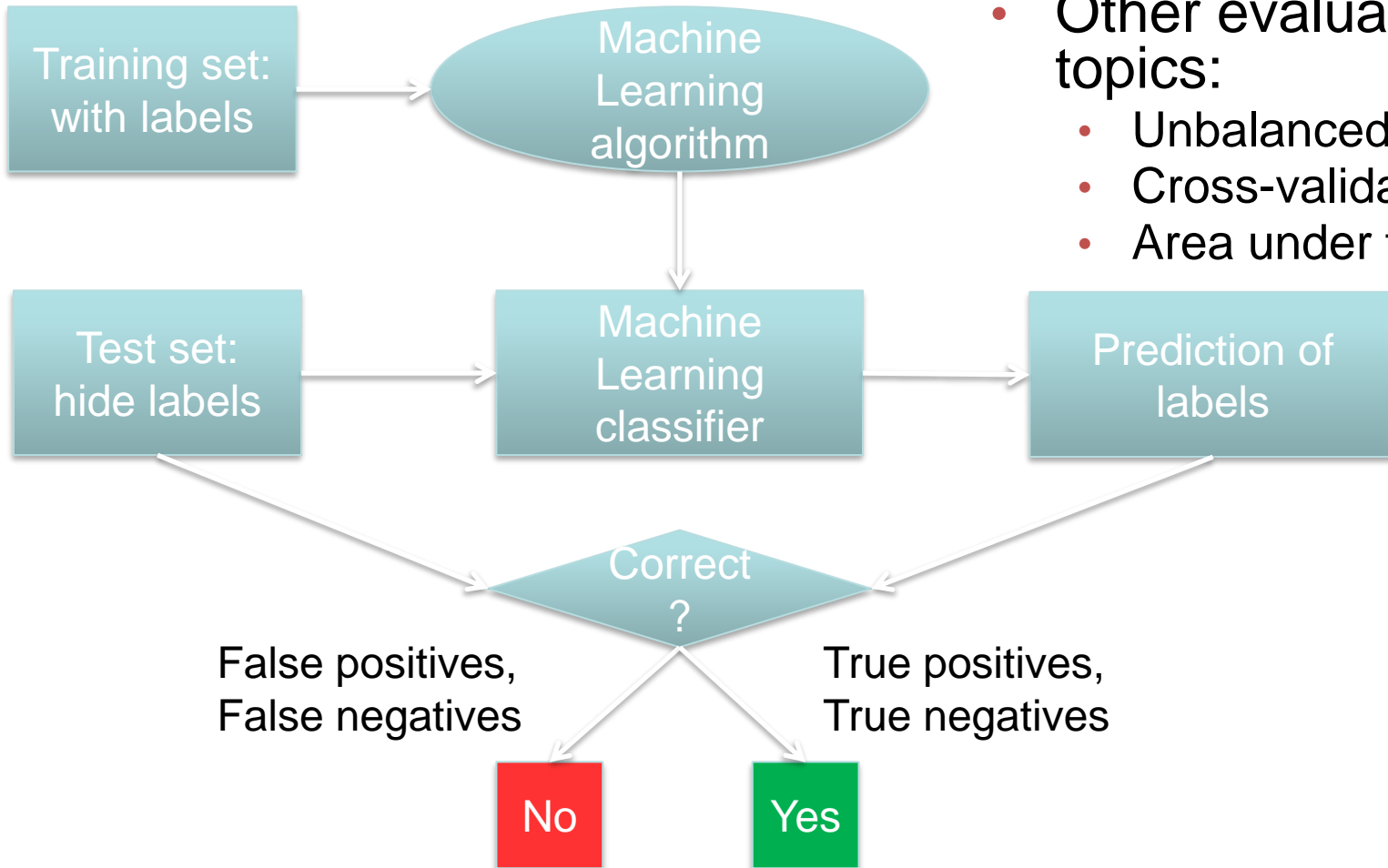
- Evaluation using data: Training, validation, testing:
 - “Simple”: Split original dataset into training data and test data, use test data to evaluate accuracy of machine learning model
 - “Complex”: Split original dataset into training data, validation data, and test data
- Evaluation baselines:
 - Other ML methods – “bakeoff”
 - Experimental results from literature
 - Comparison against theory
 - Evaluation by human experts
 - Comparison to results achieved by other software (simulation)
 - ...
- Evaluation depends on project type:
 - Application-oriented project: Models induced using different ML algorithms – “solve application problem using different tools”
 - Method-oriented project: Datasets from different application areas – “demonstrate generality of method across multiple data sets”

Evaluation

- Need a clear idea of what we are trying to achieve
- Should be able to connect the results of machine learning to the goal(s) of an organization (business)
- However, it is often difficult to measure the ultimate business goal(s), due to inadequate or complex data
 - We can measure a surrogate in such cases
 - Need to decide the surrogate through careful analysis

Evaluation: Overview

- Focus on classifiers
- Evaluating classifiers: accuracy
- Other evaluation topics:
 - Unbalanced classes
 - Cross-validation
 - Area under the curve



Evaluating Binary Classifiers

- Assumption: binary (0/1, Yes/No, Positive/Negative) classification
- Positives and negatives - in machine learning terminology:
 - **Negatives** are the uninteresting outcomes
 - **Positives** are the outcomes of interest (sometimes few)
- We have the following 4 possibilities:
 - False Positives (FP): Test *incorrectly* reports a value as *positive*
 - True Positives (TP): Test *correctly* reports a value as *positive*
 - True Negatives (TN): Test *correctly* reports a value *negative*
 - False Negatives (FN): Test *incorrectly* reports a value as *negative*

Our aim is to reduce FPs and FNs. The number of FPs may dominate the number of FNs, but the cost of mistakes made on FNs may be higher. (More about this later.)

Confusion Matrix

- Confusion Matrix :
 - An $n \times n$ matrix for a classification problem with n classes
 - For binary classification: 2×2 confusion matrix

– Main diagonal

		Predicted class	
		Positive (1)	Negative (0)
Actual class	Positive (1)	True positive	False negative
	Negative (0)	False positive	True negative

Correct

Comes from test data or "real world"

Comes from binary classifier

Evaluation Metric: Accuracy

- Define (based on the confusion matrix):
 - TPs: Number of true positives
 - TNs: Number of true negatives
 - FPs: Number of false positives
 - FNs: Number of false negatives

- Accuracy a - proportion of correct decisions $\bar{e} = \frac{TPs+TNs}{TPs+TNs+FPs+FNs} = 1 - e$

➤ A typical goal of machine learning is to maximize

Beyond Accuracy

- Accuracy, and closely related metrics, are good starting points for evaluation
- However, there are some potential problems:
 1. Unbalanced classes
 2. Desire to use “test data” during training
 3. Sensitivity to varying parameters
- Below we study these problems in more detail, and sketch solutions

Accuracy and Unbalanced Classes

- Is plain accuracy sufficient to evaluate a model?
- *Perhaps not.* In classification problems where one class is rare, the class distribution becomes highly skewed
 - E.g.: credit card transactions
 - 100 transactions: 98 legitimate, 2 fraudulent
 - Classifier classifies all transactions as legitimate
 - Accuracy $a = 98/100 = 98\%$
 - Is this a good classifier?

Unequal Costs

- Simple classification accuracy makes no distinction between FPs and FNs
- In applications, the gravity of FPs versus FNs can vary significantly
- Examples:
 - In medical diagnosis: a FN (a disease was not caught) can be life threatening
 - In fraud detection: a FP (a transaction was flagged as fraudulent but was not) can affect customer relations and involve legal issues

Expected Value

- Expected value is the weighted average of all possible outcomes, where the weight is the probability of occurrence

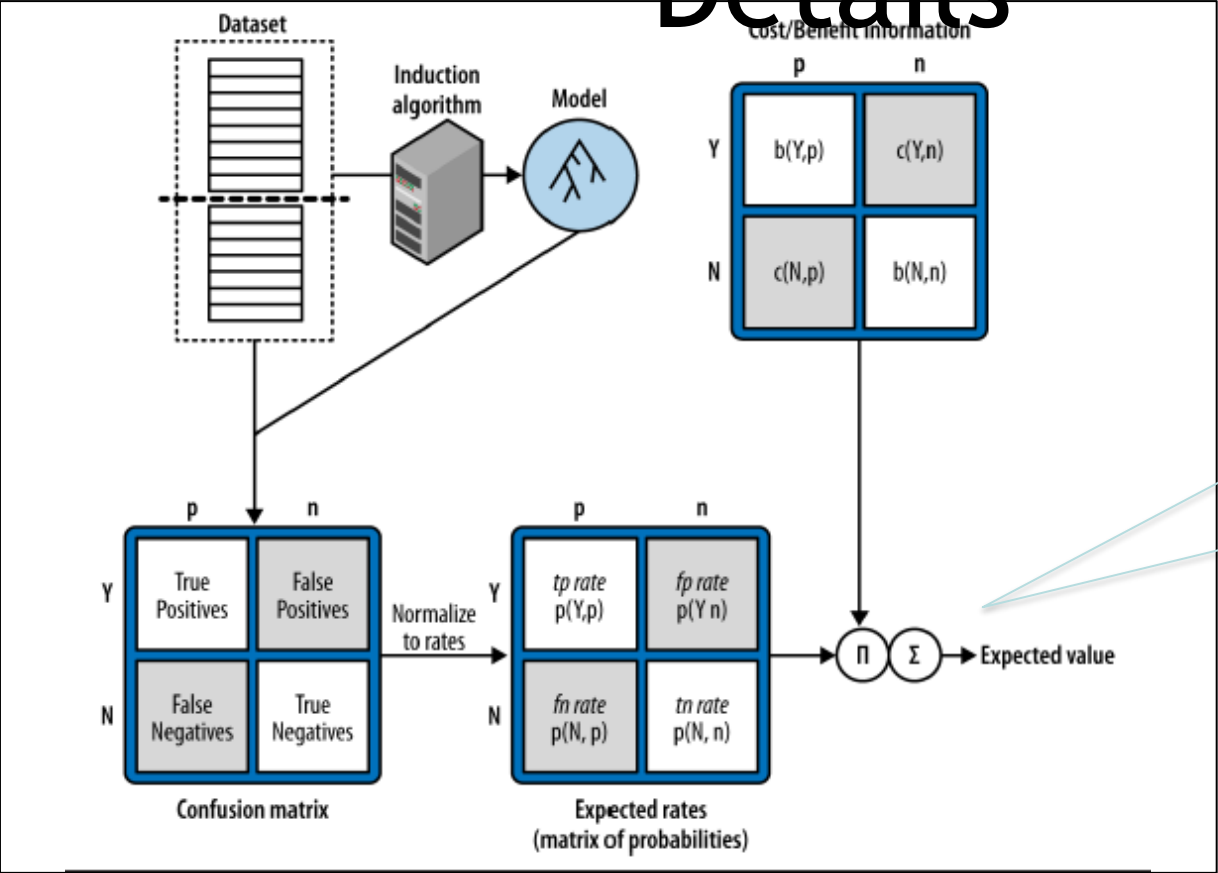
$$Ev = \sum P(o_j) * V(o_j)$$

- Where:
 - o_j is a possible decision outcome,
 - $P(o_j)$ is its probability, and
 - $V(o_j)$ is its value

Using Expected Value to Evaluate a Classifier

- How do we decide if a data driven decision is better than a decision taken intuitively?
- Expected value can – given information about outcomes, probabilities, and their values – be used to determine best decisions for a particular model
- Expected value aggregates all possible outcomes to decide whom to target or where to spend

Expected Value Calculation: Details



When classifying, maximize expected value instead of posterior probability.

From **P&F** p. 197.


Metrics from Bio-Medicine

- Specificity = True positive rate $\frac{TP}{TP + FN}$
- Sensitivity = True negative rate $\frac{TN}{TN + FP}$

Metrics from Information Retrieval

- Precision $\frac{TP}{TP + FP}$, commonly used in information retrieval:

– Eg: $\frac{\text{NumberOf RelevantDocuments Retrieved}}{\text{TotalNumberOfDocuments Retrieved}}$

- Recall $\frac{TP}{TP + FN}$ 
– Eg: $\frac{\text{NumberOf RelevantDocuments Retrieved}}{\text{TotalNumberOf RelevantDocuments}}$

- F-Measure $\frac{Precision \cdot Recall}{Precision + Recall}$ harmonic mean of precision and recall

=

Cross-Validation: Details

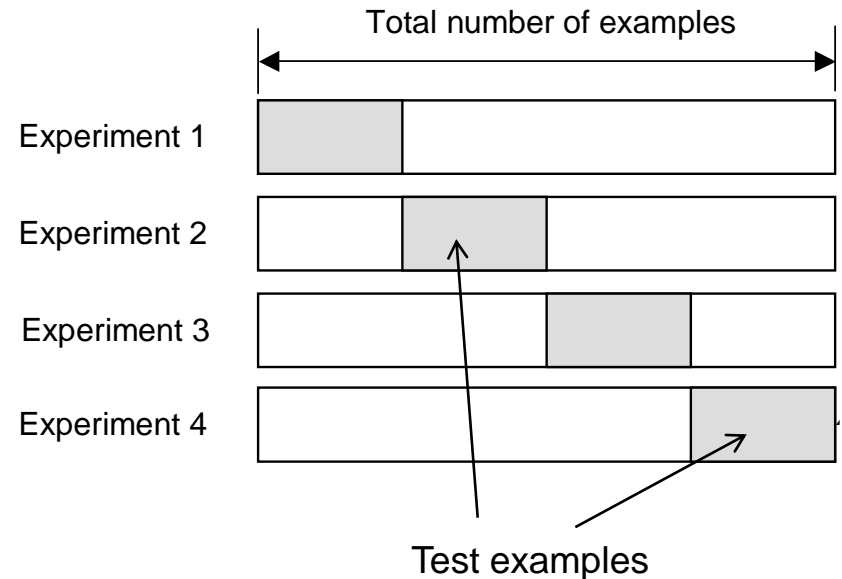
- A model selection technique: assesses how the predictions will generalize to an independent dataset

- **K-Fold Cross-validation:**

- Create a K-fold partition of the dataset

- Perform K experiments as

- Use $K-1$ folds for training and the remaining one for



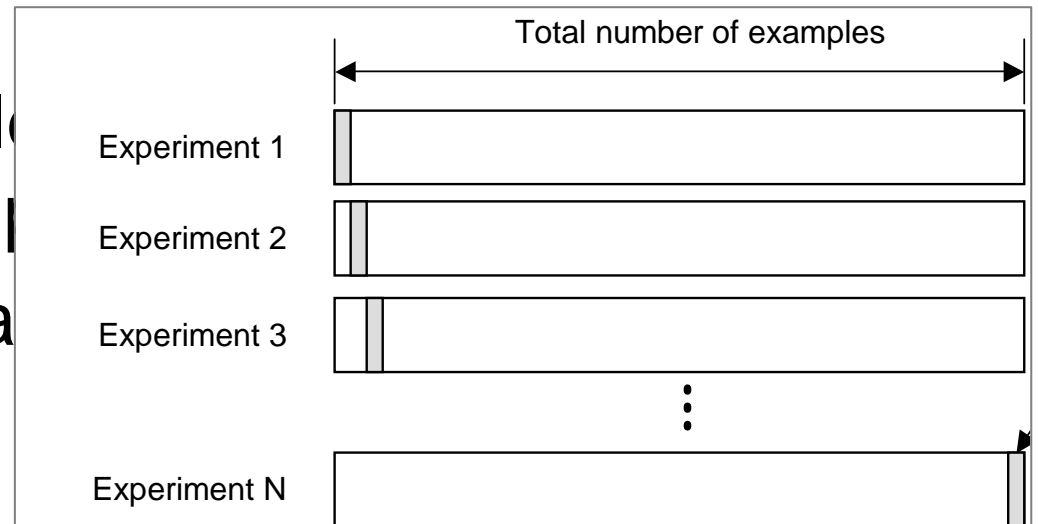
* http://research.cs.tamu.edu/prism/lectures/iss/iss_l13.pdf

Leave-One-Out Cross Validation

- Degenerate case of K-Fold Cross Validation
 - $K = \text{total number of examples } (N)$
- For a dataset with N examples, perform N experiments

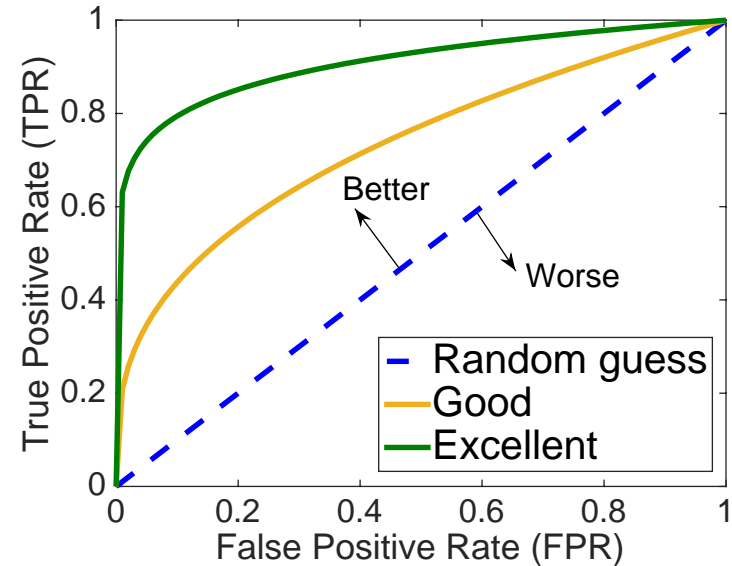
- Use $N-1$ examples for training, and the remaining example for testing
- Average error rate

$$E = \frac{1}{N} \sum_{i=1}^N E_i$$



Receiver Operating Characteristic (ROC)

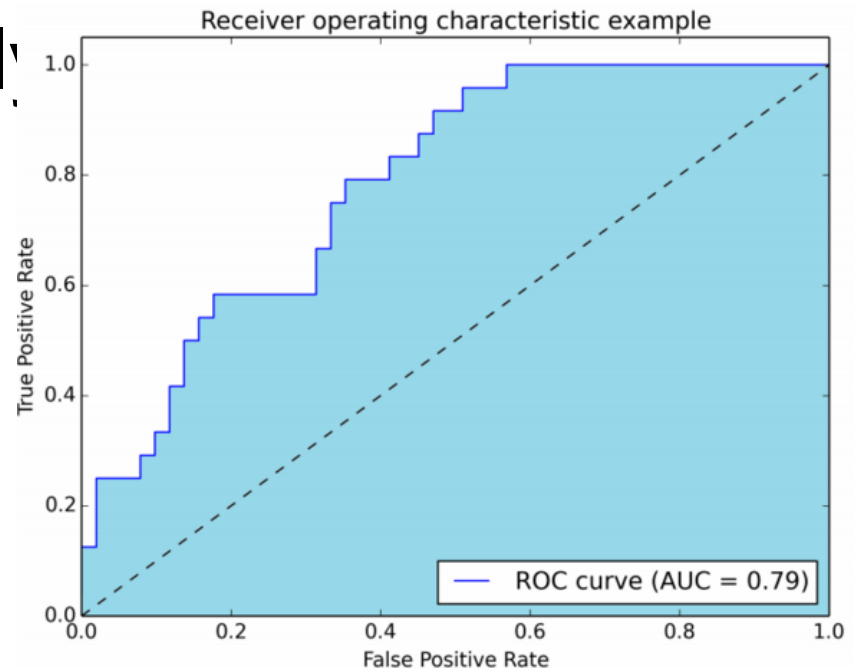
- ROC curve: TPR versus FPR plot representing the performance of a binary classifier as its discrimination threshold is varied
- True positive rate (TPR): recall
- ROC analysis provides tools to select possibly optimal models



	Truth: positive	Truth: negative
Predicted : positive	True positive	False positive
Predicted : negative	False negative	True negative

Area Under the Curve (AUC)

- Probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one
- Classification analysis: which model predicts the classes best?
 - Model with higher AUC
- Recent research¹ shows: AUC is noisy as a classification measure



1. Hanczar, Blaise; Hua, Jianping; Sima, Chao; Weinstein, John; Bittner, Michael; and Dougherty, Edward R. (2010); Small-sample precision of ROC-related estimates, *Bioinformatics* 26 (6): 822–830

<http://stats.stackexchange.com/questions/132777/what-does-auc-stand-for-and-what-is-it>

Summary

- Accuracy, and closely related metrics, are good starting points for evaluation
- To handle limitations of accuracy, we have:
 1. Unbalanced classes:
 2. Desire to use “test data” during training: cross-validation
 3. Sensitivity to varying parameters:
- Other evaluation problems: User studies, evaluation of “business impact,” ...

Questions?

