Applied Data Analysis (CS401)

ada

Lecture 8
Applied machine learning
9 Nov 2022

~~Robert West~~
Akhil Arora

EPFL

dlab

Announcements

- Homework H1
  - Feedback has been released
  - Postmortem: recorded video to be released on Fri 11 Nov
- Project milestone P2 due on Fri 18 Nov 23:59
- Friday's lab session: two parallel tracks:
  - Track 1: exercise on applied machine learning (BCH 2201)
  - Track 2: project office hours (Zoom)
    - Logistics: see Ed post
    - Do come and ask for feedback -- everyone will win!

Give us feedback on this lecture here:
https://go.epfl.ch/ada2022-lec8-feedback

- What did you (not) like about this lecture?
- What was (not) well explained?
- On what would you like more (fewer) details?
- Where is Pumpkin Pete?
- …

Why an extra class on applied ML?

Machine Learning that Matters

Kiri L. Wagstaff KIRI.L.WAGSTAFF@JPL.NASA.GOV
Jet Propulsion Laboratory, California Institute of Technology, 4800 Oak Grove Drive, Pasadena, CA 91109 USA

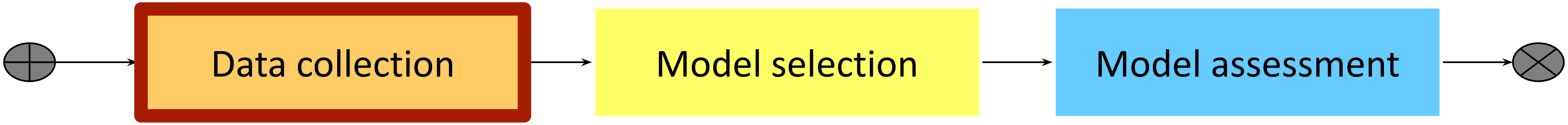
link

Classic ML class

ADA

It is easy to sit in your office and run a Weka (Hall et al., 2009) algorithm on a data set you downloaded from the web. It is very hard to identify a problem for which machine learning may offer a solution, determine what data should be collected, select or extract relevant features, choose an appropriate learning method, select an evaluation method, interpret the results, involve domain experts, publicize the results to the relevant scientific community, persuade users to adopt the technique, and (only then) to truly have made a difference (see Figure 1). An ML researcher might well feel fatigued or daunted just contemplating this list of activities. However, each one is a necessary component of any research program that seeks to have a real impact on the world outside of machine learning.

Classification pipeline



Data collection

The first step is collecting data related to the classification task.

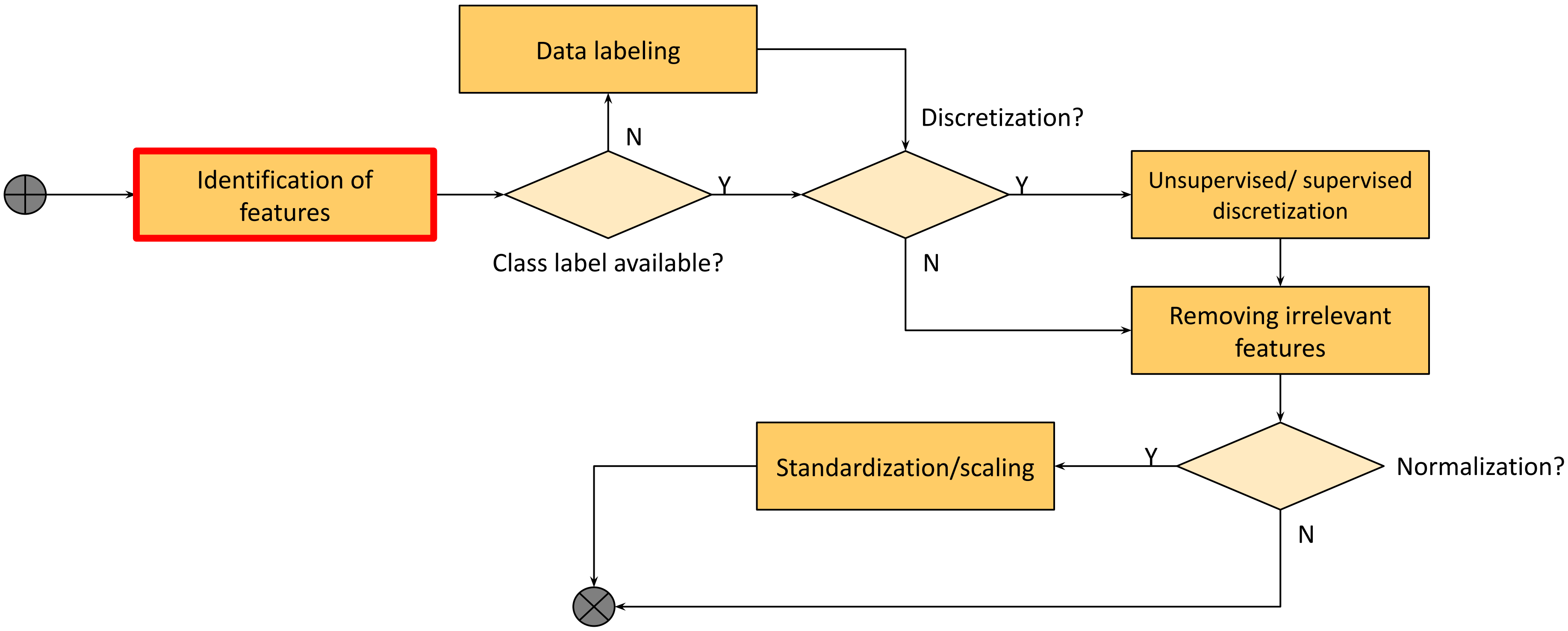
- Definition of the attributes (or features) that describe a data item and the class label.

Domain knowledge is needed.

What if assigning the class label would be too time-consuming or even impossible?

→ Unsupervised methods (e.g., clustering); cf. next lecture!

Data collection



Features

Different types of features [more]

- Continuous (e.g., height, temperature ...)
- Ordinal (e.g., “agree”, “don’t care”, “disagree” ...)
- Categorical (e.g., country, gender ...)

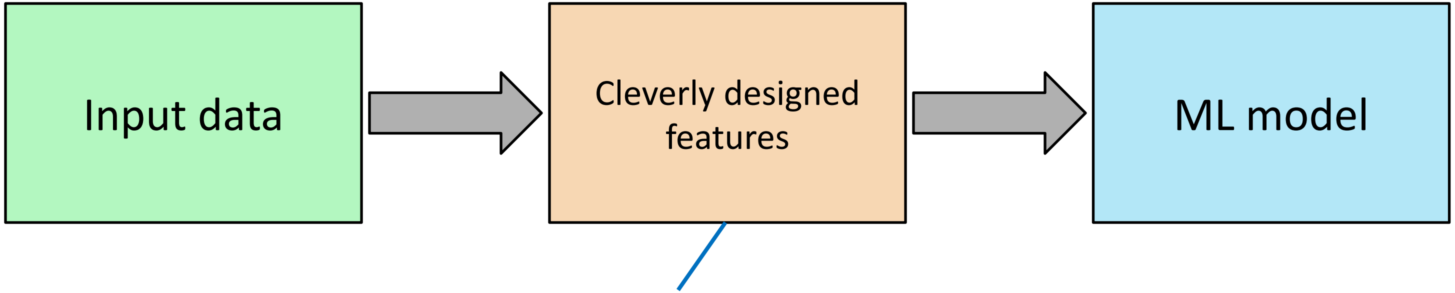
New features can be generated from **simple stats**

- *Feature engineering* is considered a form of art, therefore it is sometimes useful to find what other people did for similar problems

Some classifiers require categorical features => **discretization**

ML before 2012*

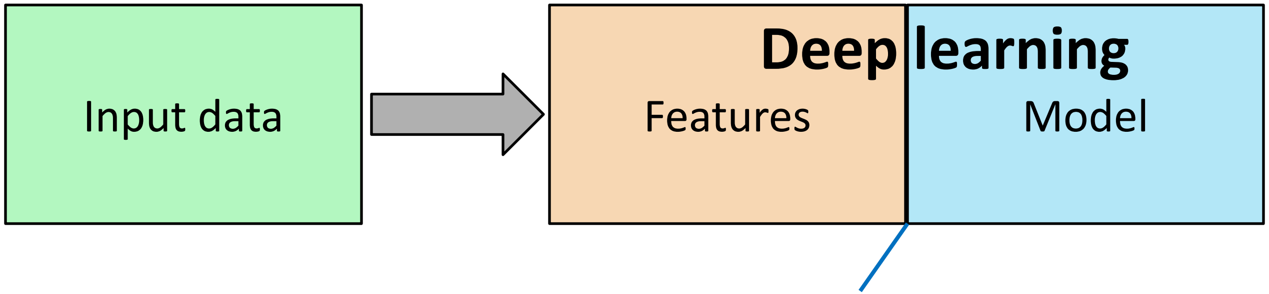
(but still very common today)



Most of the “heavy lifting” in here.
Final performance only as good as the feature set.

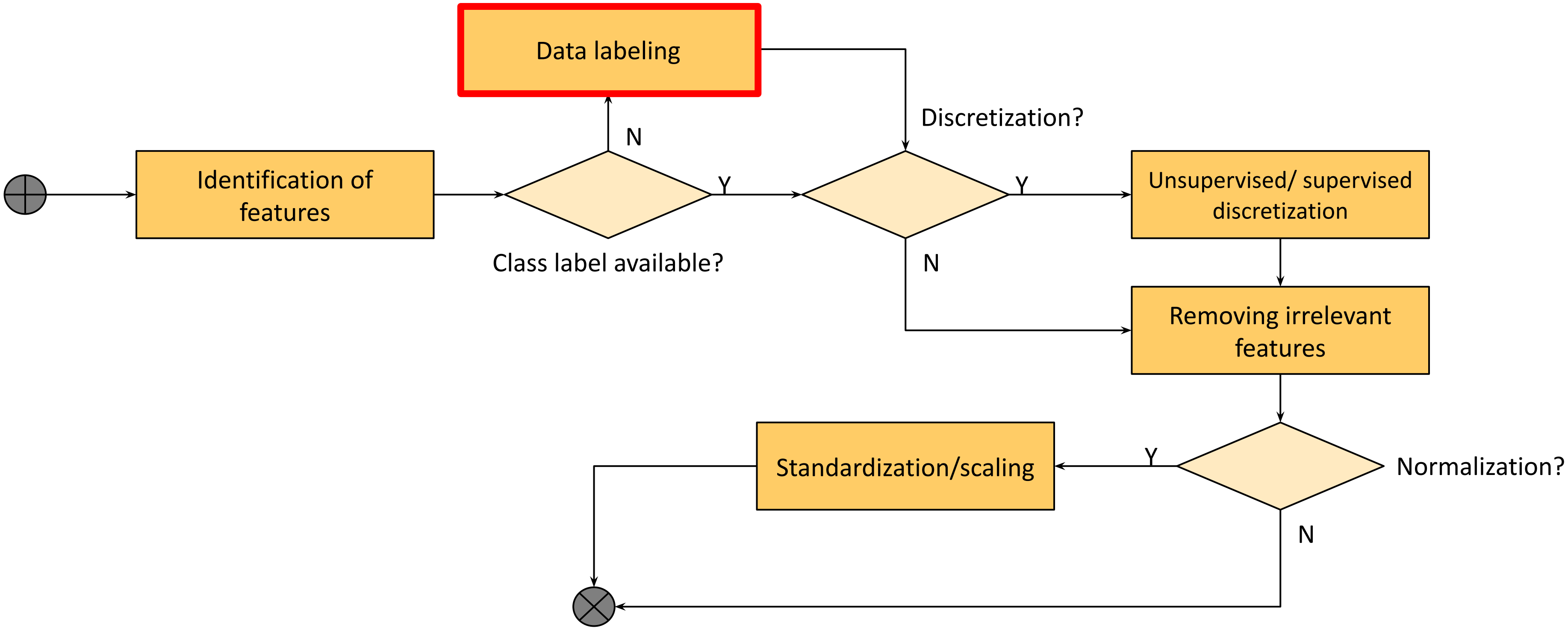
* Before publication of Krizhevsky et al.’s ImageNet CNN paper

A typical ML approach after 2012



Features and model learned together, mutually reinforcing

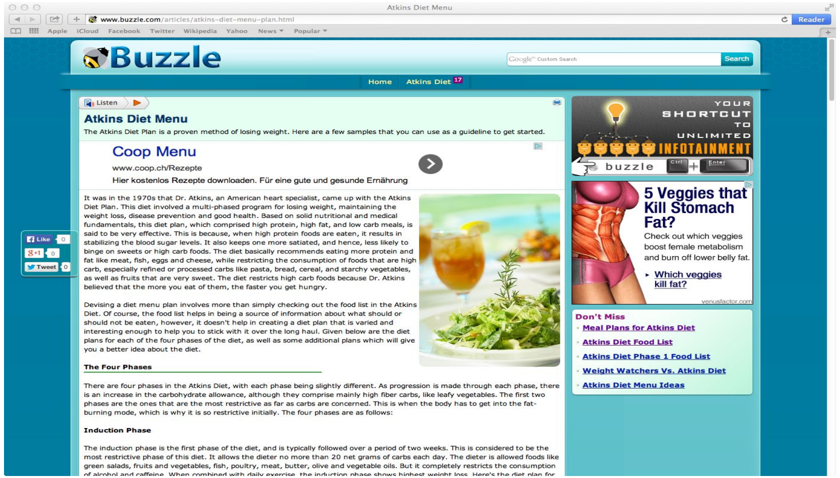
Data collection



Labels

Collecting a lot of data (features) is often easy.

Labeling data is time consuming, difficult, and sometimes even impossible.



Label: "Is page credible?"

Human dietary expert is needed

Potential labelers

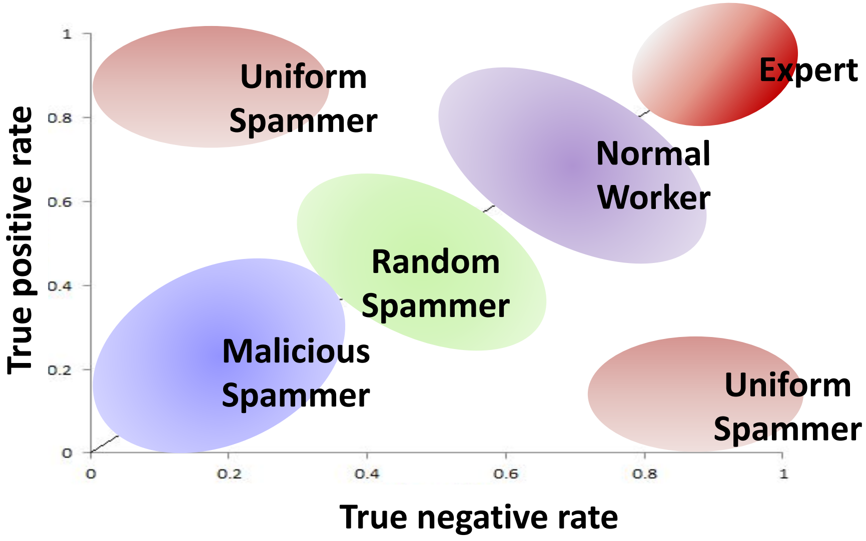
- You
- Older days:
  - Undergraduate students
  - Domain experts ($$$)
- Now: crowdsourcing
  - Can get both amateurs (~ undergrad students)
  - and experts


1. Submit task
Requester
4. Collect answers
Is this webpage credible?
C C ¬C C ¬C
Crowdsourcing platform
CrowdFlower
clickworker.com
amazon mechanical turk™
Artificial Artificial Intelligence
2. Accept task
3. Return answers
Crowd (workers)

Crowdsourcing

Different types of workers

- Truthful
  - Expert
  - Normal
- Untruthful
  - Uniform spammer
  - Random spammer
  - Malicious spammer (a.k.a. a$s#*le)



Catching malicious spammers

- Insert obvious examples for which you already know the labels (“honeypot”)
  - Tell workers they won’t be paid if they don’t get those right
  - Filter out workers who don’t get them right
- Aggregate multiple answers
  - p.t.o.

Crowdsourcing

Answer aggregation problem

- Have each example labeled by several workers, aggregate:
  - e.g., majority vote (works if only a minority is malicious)
  - e.g., “peer prediction”, “prediction markets” (game theory: workers are paid more if they agree with others)

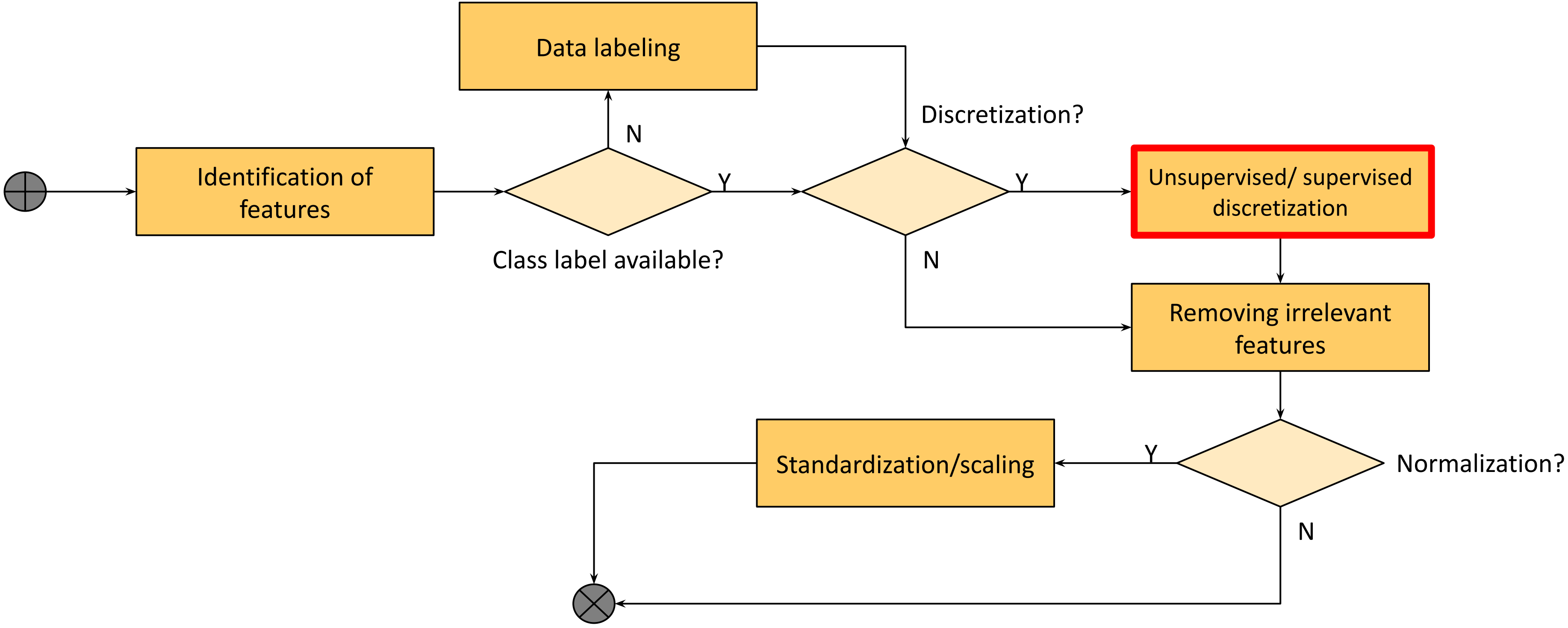
Crowd (workers)

Worker	Webpage	Credible
W_1	www.diet.com	C
W_2	www.diet.com	¬C
W_3	www.diet.com	C
...	...	...

Aggregation

www.diet.com	C

Data collection



Discretization

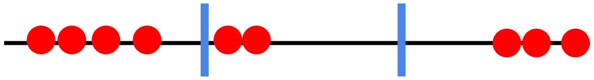
Why?

- Some classifiers want discrete features (e.g., simplest kinds of decision trees)
- Discrete features let a linear classifier learn non-linear decision boundaries
- Certain feature selection methods require discrete (or even binary) features

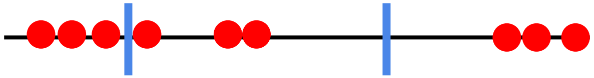
Discretization

Unsupervised

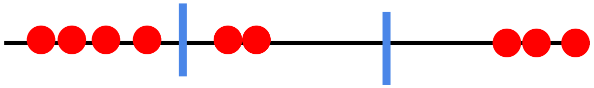
- Equal width



  - Divide the range into a predefined number of bins (bad for skewed data, e.g., from a power law)
- Equal frequency



  - Divide the range into a predefined number of bins so that every interval contains the same number of values
- Clustering

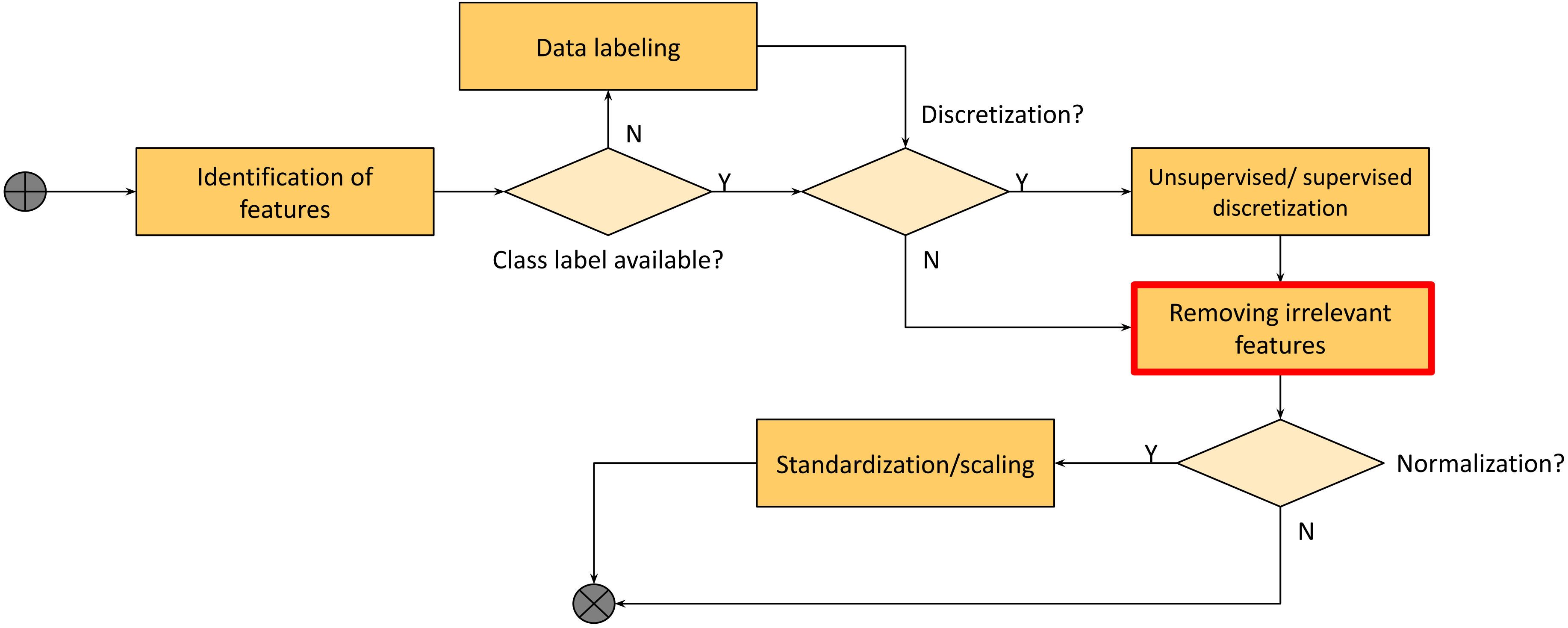


Discretization

Supervised

- Start with very fine-grained discretization
- Test the hypothesis that membership in two adjacent intervals of a feature is independent of the class
- If they are independent, they should be merged
- Otherwise they should remain separate
- Independence test: χ^2 test (“chi-squared test”) [example]
- Continue recursively

Data collection



Removing irrelevant features: Feature selection

Reducing the number of N features to a subset of the best size M < N

There are 2^N possible subsets

Solutions

- Filtering as preprocessing ("offline")
- Iterative feature selection ("online")

Offline feature selection

Rank features according to their individual predictive power; select the best ones

- Pros
  - Independent of the classifier (performed only once)
- Cons
  - Independent of the classifier (ignore interaction with the classifier)
  - Assumes features are independent

Ranking of features

Continuous features (and ideally labels)

- Pearson's correlation coefficient (capturing strength of linear [!] dependence)

$$r = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}}$$

Categorical features and labels

- Mutual information (goes beyond linear dependence)

$$I(F;C) = H(C) - H(C \mid F) = H(F) + H(C) - H(F,C)$$

$$H(F) = -\sum_{i} P(f_i)\log_2 P(f_i)$$

$$H(F,C) = -\sum_{i}\sum_{j} P(f_i, c_j)\log_2 P(f_i, c_j)$$

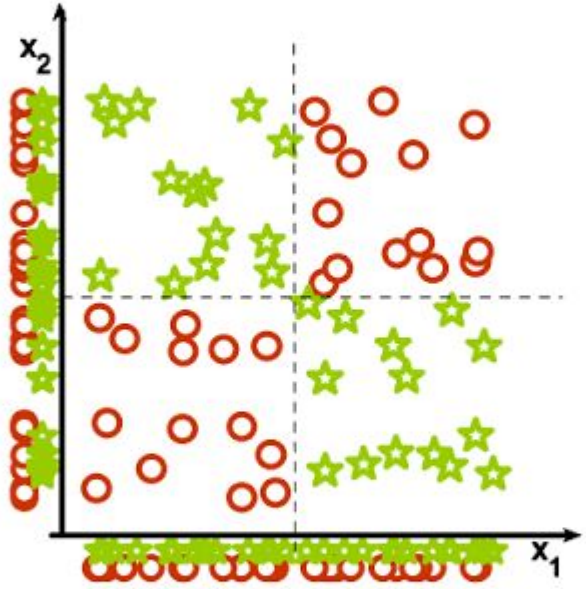
Ranking of features

Categorical features and labels (cont'd)

- χ^2 **method** ("chi-squared")
  - Test whether feature is independent of label
  - Difference w.r.t. mutual information: the χ^2 test checks the independence of the class and the feature, without indicating the strength or direction of any existing relationship (you just get a significance, a.k.a. p-value)

Ranking of features

Beware: collectively relevant features may look individually irrelevant!



Online feature selection

Forward feature selection: greedily **add** features; evaluate on validation dataset; stop when no improvement

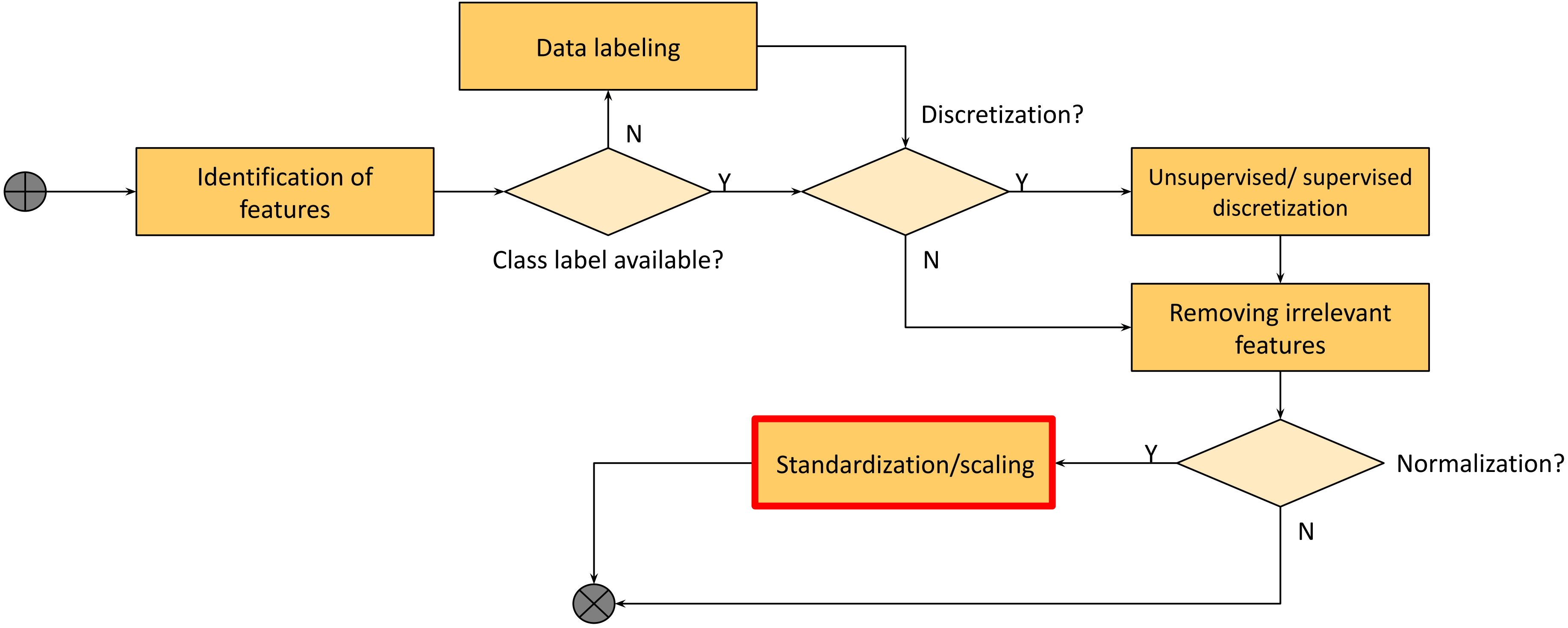
- Pros
  - Interact with the classifier
  - No feature-independence assumption
- Cons
  - Computationally intensive

Online feature selection

Backward selection (a.k.a. ablation): greedily **remove** features; evaluate on validation dataset; stop when no improvement

- Pros
  - Interact with the classifier
  - No feature-independence assumption
- Cons
  - Computationally intensive

Data collection



Feature normalization

- Some classifiers do not manage well in the face of features with very different scales, e.g.
  - Revenue in CHF: 10,000,000
  - # of employees: 300
- Features with large values dominate the others, and the classifier tends to over-optimize for them
- Even single feature may span many orders of magnitude, e.g.
  - City size (most cities small, some huge)

Logarithmic scaling

$$x_i' = \log(x_i)$$

- Consider order of magnitude, rather than direct value
- Good for heavy-tailed features (e.g., from power laws)

Min-max scaling

$$x_i' = (x_i - m_i)/(M_i - m_i)$$

where M_i and m_i are the max and min values of feature x_i respectively

The new feature x_i' lies in the interval [0,1]

Standardization

$$x_i' = (x_i - \mu_i)/\sigma_i$$

where μ_i is the mean value of feature x_i, and σ_i is the standard deviation

The new feature x_i' has mean 0 and standard deviation 1

Dangers of standardization and scaling

Standardization:

- Assume that the data has been generated by a Gaussian
- Uses mean and std → not meaningful for heavy-tailed data (may be mitigated by log-scaling)

Min-max scaling:

- If the data has outliers, they scale the typical values to a very small interval

Classification pipeline

Model selection: high level

Need to choose type of model

- Logistic regression?
- Decision trees?
- Random forest?
- Gradient-boosted trees?
- Support vector machine?
- Deep learning?
- …

Model selection: low level

Usually a classifier has some “hyperparameters” to be tuned

- Set of features to include
- Decision threshold (e.g., logistic regression)
- Distance function (e.g., k-NN)
- Number of neighbors (e.g., k-NN)
- Number of trees (e.g., random forest)
- Regularization parameter
- Learning rate (for gradient descent algorithms)

Loss function (more of them later!)

Categorical output

- e.g., 0-1 loss function, risk (= 1 minus accuracy):

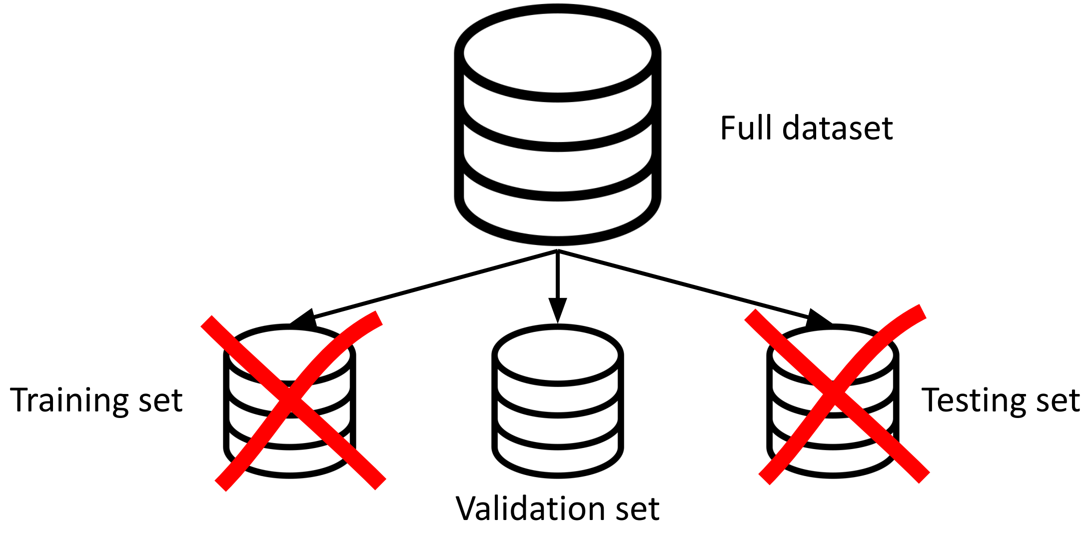
$$J = \frac{1}{n}\sum_{i=1}^{n} \#(y \neq f(x_i))$$

Real-valued output

- e.g., squared error: $J = \frac{1}{n}\sum_{i=1}^{n} (y_i - f(x_i))^2$

- e.g., absolute error: $J = \frac{1}{n}\sum_{i=1}^{n} \left|y_i - f(x_i)\right|$

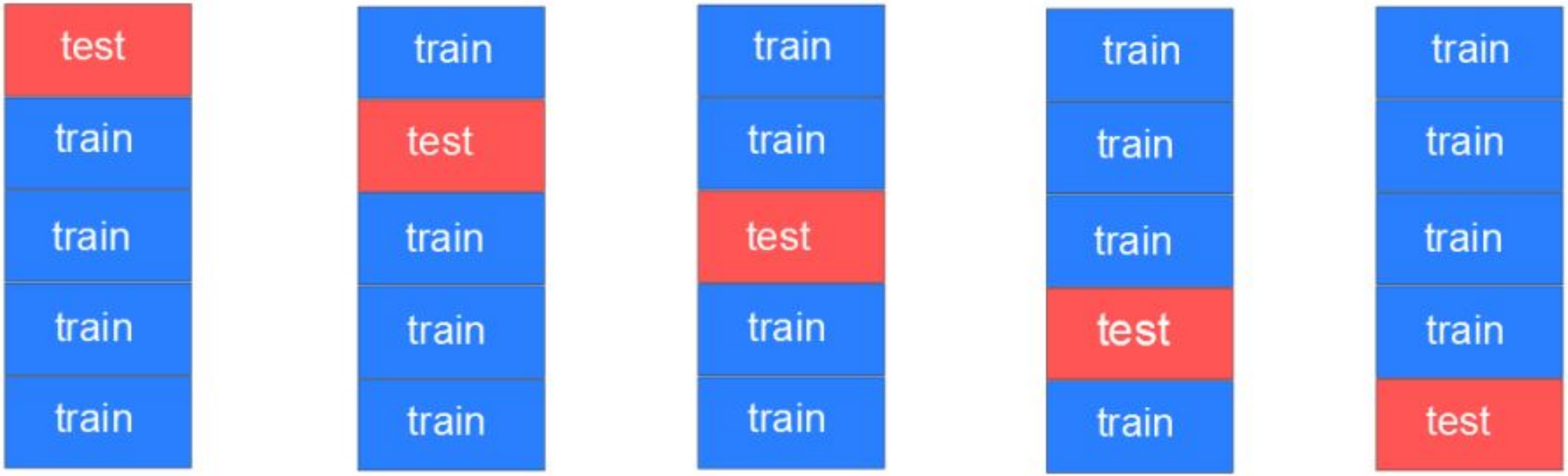
Model selection: on what data to evaluate?



What if you can’t afford a 3-way split because you have too little data?

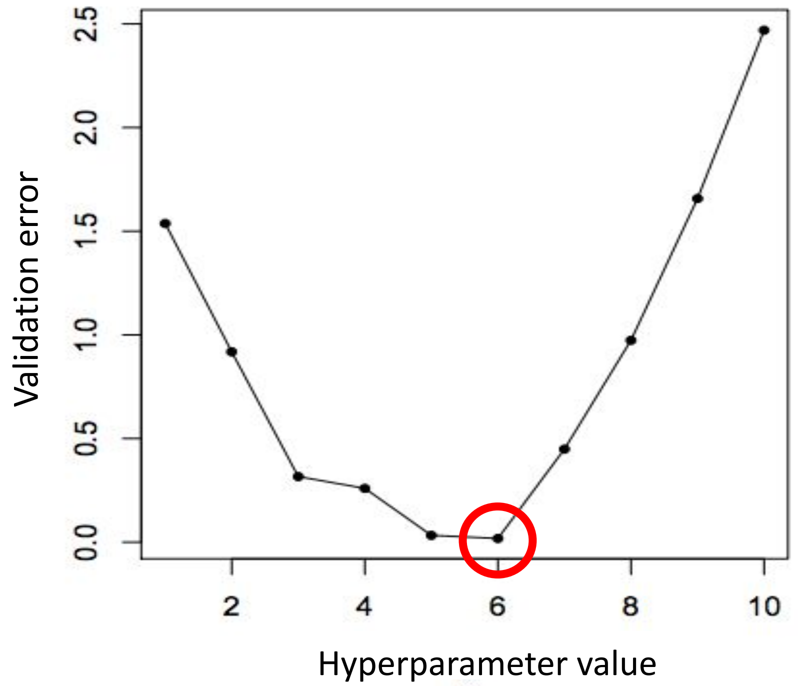
→ Cross-validation! (p.t.o.)

Cross-validation



- Last lecture: [leave-one-out cross-validation]() == N-fold cross-validation (where N is #data points)
- More efficient: m-fold cross-validation (in above picture: m = 5)
- Average performance over the m red portions → validation error
- Repeat procedure for all candidate models and pick the one with the lowest validation error

Model selection



Performance metrics for binary classification

For categorical binary classification, the usual metrics are based on the **confusion matrix,** which has 4 values:

- True Positives (positive examples classified as positive)
- True Negatives (negative examples classified as negative)
- False Positives (negative examples classified as positive)
- False Negatives (positive examples classified as negative)

		Class	
		Pos	Neg
Classified	Pos	TP	FP
	Neg	FN	TN

Accuracy

		Class	
		Pos	Neg
Classified	Pos	TP	FP
	Neg	FN	TN

$$A = \frac{TP+TN}{TP+TN+FP+FN} = \frac{TP+TN}{N}$$

Appropriate metric when

- classes are not skewed
- errors (FP, FN) have the same importance

Accuracy (skewed example)

Classifier 1		Class	
		Fraud	¬Fraud
Classified	Fraud	5	10
	¬Fraud	5	80

Accuracy = 85/100 = 85%

Always ¬Fraud		Class	
		Fraud	¬Fraud
Classified	Fraud	0	0
	¬Fraud	10	90

Accuracy = 90/100 = 90%

Question time

Which classifier is better?

- Classifier 1
- Classifier 2
- Both are equally good

Classifier 1		Class A	Class B
Classified	A	45	20
	B	5	30

100 data points

Classifier 2		Class A	Class B
Classified	A	40	10
	B	10	40

100 data points

POLLING TIME

- Scan QR code or go to https://web.speakup.info/room/join/66626

Question time

Classifier 1		Class: Cancer	Class: ¬Cancer
Classified	Cancer	45	20
	¬Cancer	5	30

Classifier 2		Class: Cancer	Class: ¬Cancer
Classified	Cancer	40	10
	¬Cancer	10	40

Which classifier is better?

- ○ Classifier 1
- ○ Classifier 2
- ○ Both are equally good

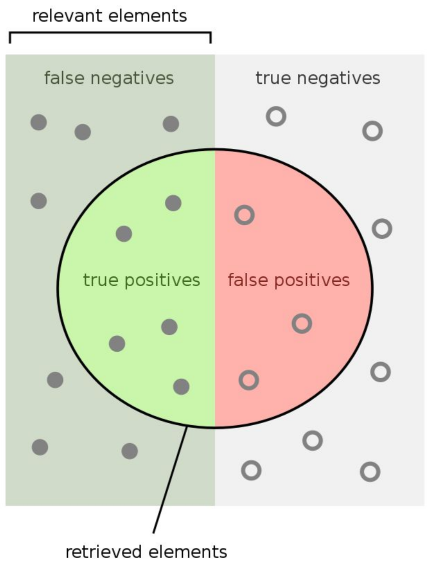
Precision and recall

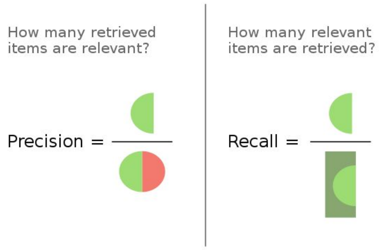
Precision: what fraction of positive predictions are actually positive?

$$P = \frac{TP}{TP + FP}$$

Recall: what fraction of actually positive examples did I recognize as such?

$$R = \frac{TP}{TP + FN}$$





[source]

Precision and recall

Classifier 1		Class: Cancer	Class: ¬Cancer
Classified	Cancer	45	20
	¬Cancer	5	30

$P_1=45/65=0.69$

$R_1=45/50=0.9$

Classifier 2		Class: Cancer	Class: ¬Cancer
Classified	Cancer	40	10
	¬Cancer	10	40

$P_2=40/50=0.8$

$R_2=40/50=0.8$

Everybody has cancer		Class: Cancer	Class: ¬Cancer
Classified	Cancer	50	50
	¬Cancer	0	0

$P = 50/100 = 0.5$

$R = 50/50 = 1$

F-score

$$P = \frac{TP}{TP + FP}$$

$$R = \frac{TP}{TP + FN}$$

Sometimes it's necessary to have a single metric to compare classifiers

F-score (or F1-score): harmonic mean of precision and recall

F1 = 1 / (0.5 * (1/P + 1/R)) = $2 \cdot \frac{P \cdot R}{P + R}$

Precision and recall can be **differently weighted,** if one is more important than the other

Precision and recall

Classifier 1		Class	
		Cancer	¬Cancer
Classified	Cancer	45	20
	¬Cancer	5	30

Classifier 2		Class	
		Cancer	¬Cancer
Classified	Cancer	40	10
	¬Cancer	10	40

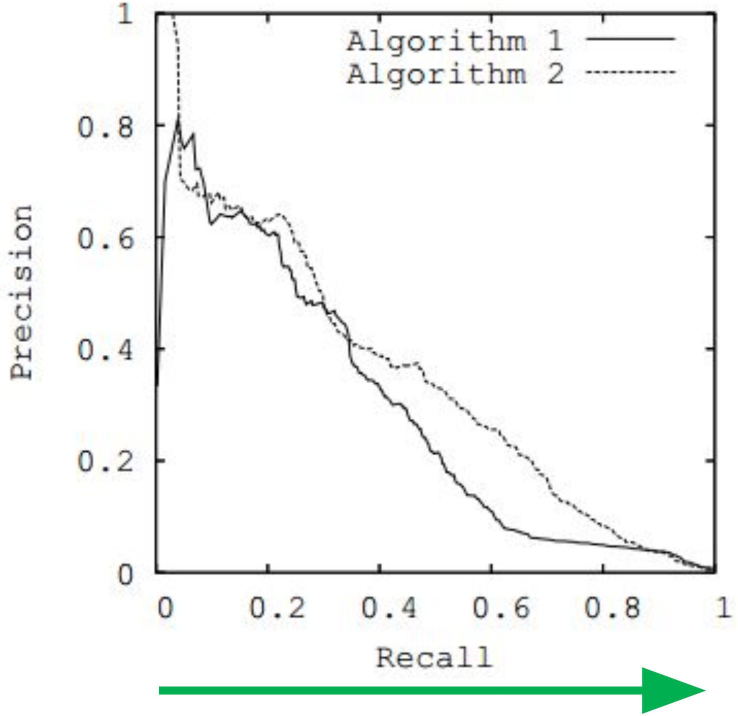
$F_1=2*(0.69*0.9)/(0.69+0.9) = 0.78$

$F_2=2*(0.8*0.8)/(0.8+0.8) =0.8$

Everybody has cancer		Class	
		Cancer	¬Cancer
Classified	Cancer	50	50
	¬Cancer	0	0

$F=2*(0.5*1)/(0.5+1)=0.66$

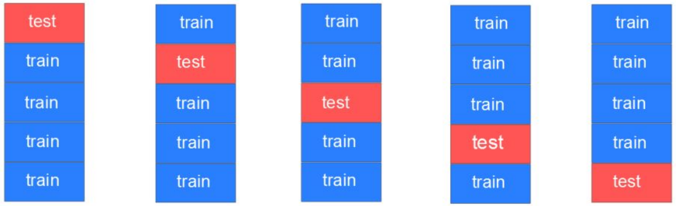
Precision/recall curve



Decreasing classification threshold

Recap

- Model selection:
  - Use training data in cross-validation
  - Need evaluation metric
    - Typically based on confusion matrix
    - e.g., accuracy, precision, recall



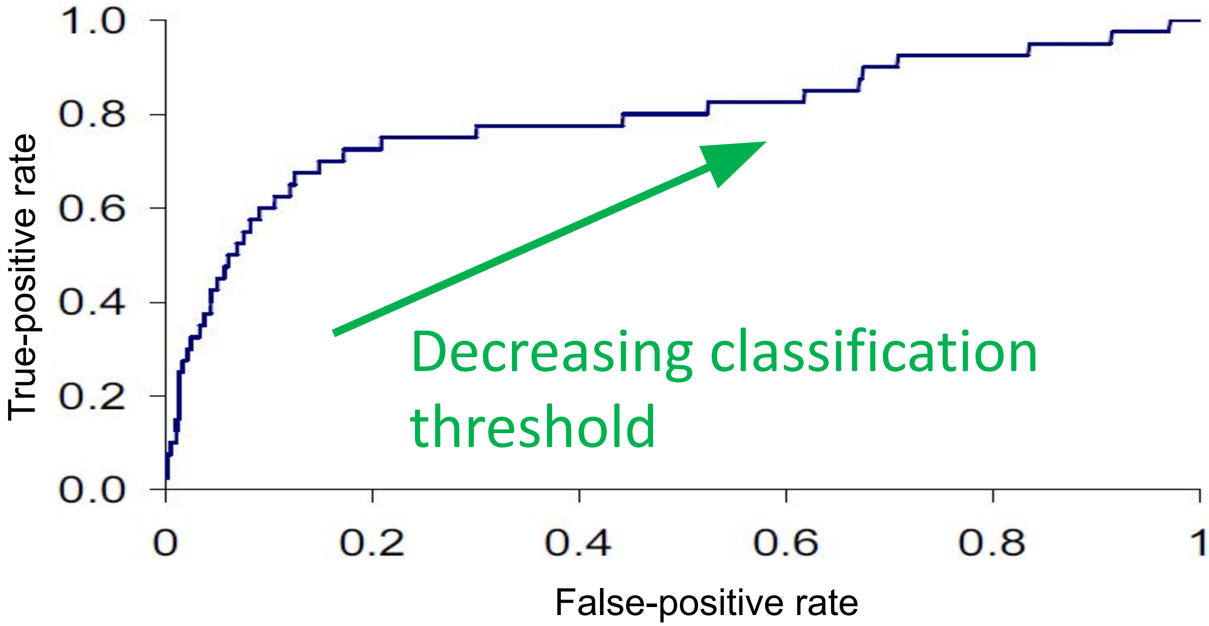
		Class	
		A	B
Classified	A	TP	FP
	B	FN	TN

ROC curve

ROC = Receiver-Operating Characteristic (WTF?!)

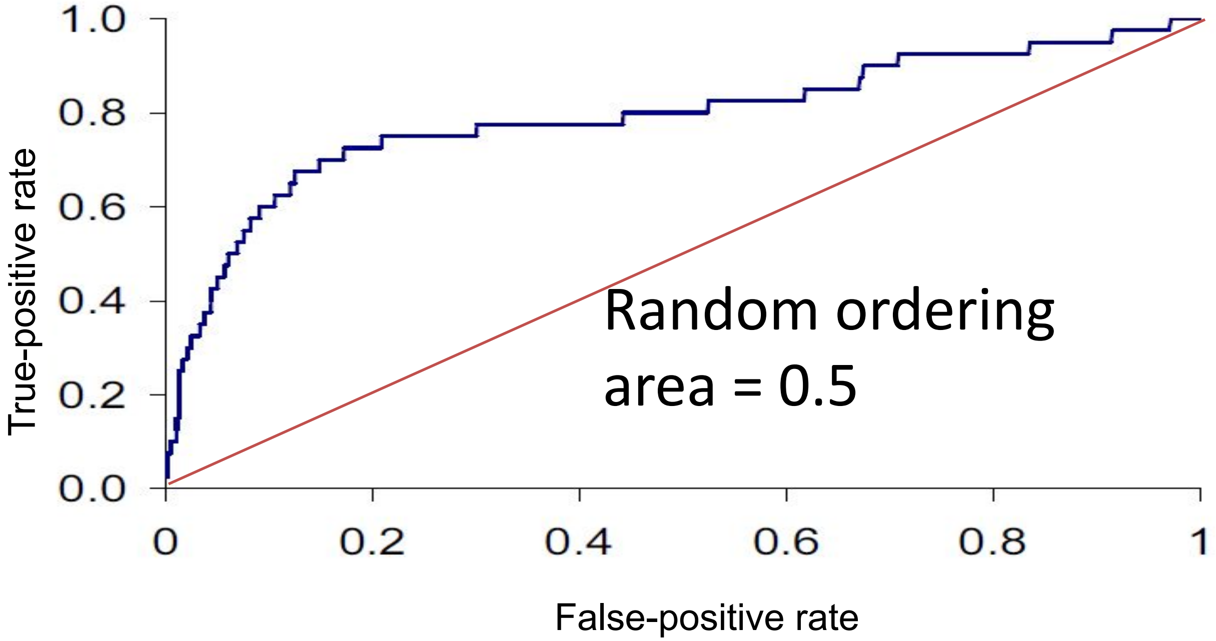
Y-axis: true-positive rate = TP/(TP + FN), a.k.a. recall

X-axis: false-positive rate = FP/(FP + TN)

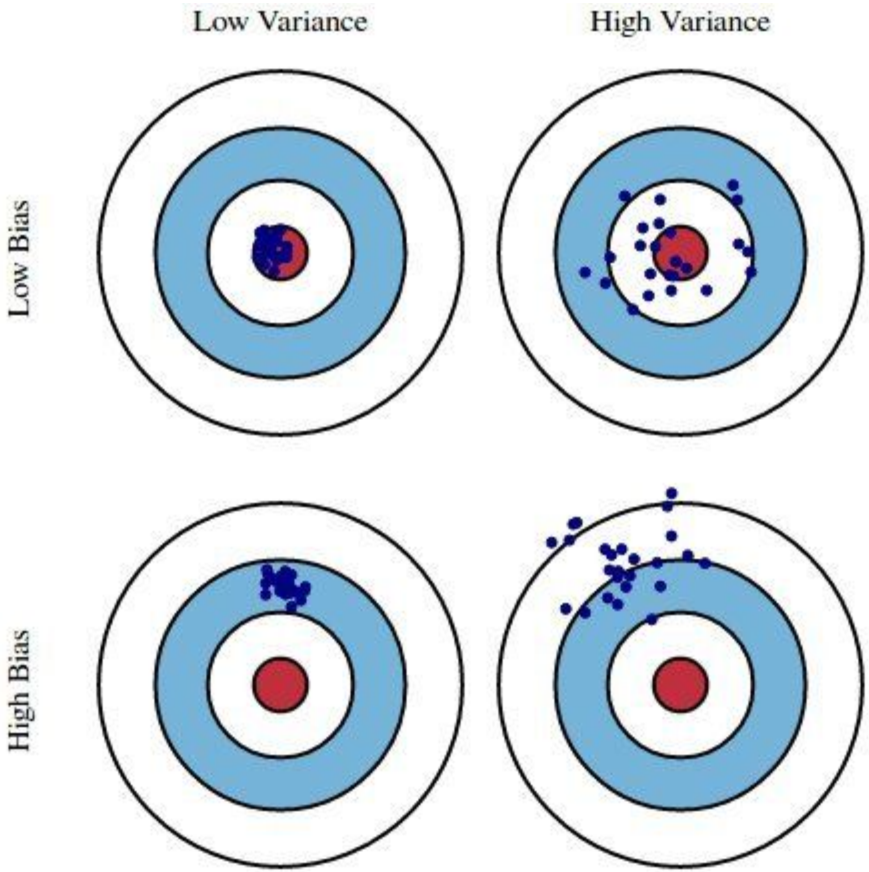


ROC AUC

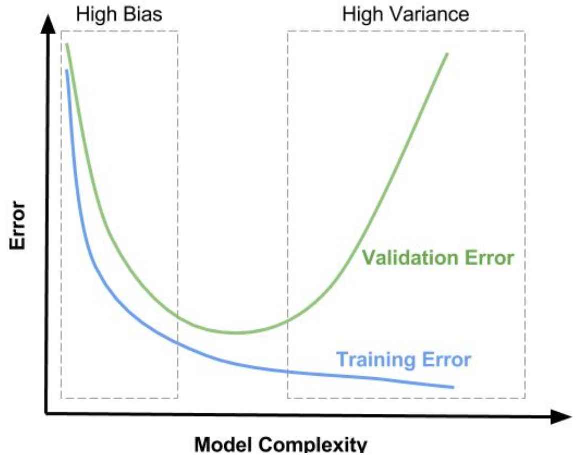
ROC AUC is the "area under the curve" – a single number that captures the overall quality of the classifier. It should be between 0.5 (random classifier) and 1.0 (perfect).

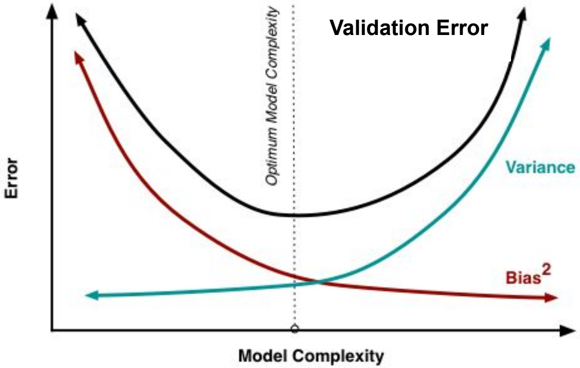


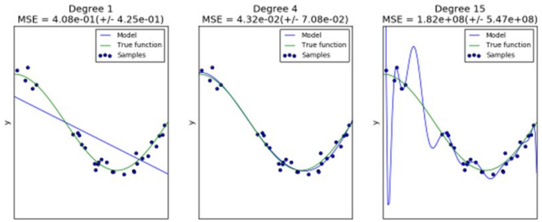
Bias and variance



[lecture 7]



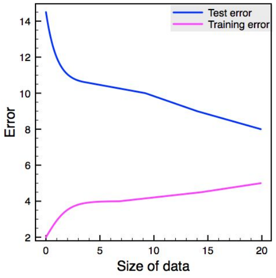




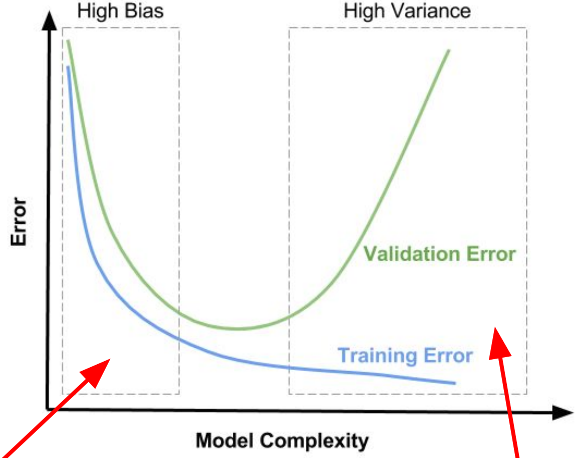
How to know where on the x-axis you are (without fiddling with model complexity)?

Bias and variance

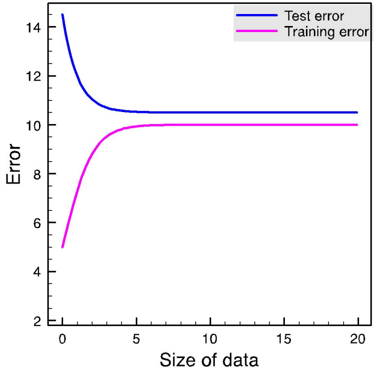
Bias and variance can be assessed by comparing the error metric on the training set and the testing set => always **plot learning curves** (training set size vs. training/testing errors)

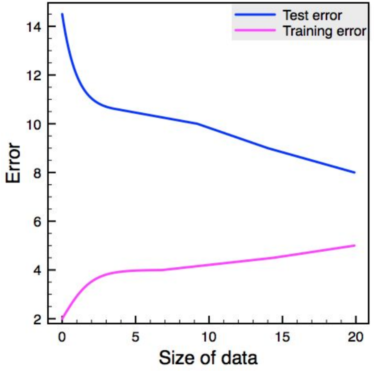


When more data helps



Fixed data set size
Varying model complexity





Fixed model complexity
Varying data set size

High bias

More data doesn't help

High variance

More data helps

For curious ADAventurers:
"Reconciling modern machine-learning practice and the classical bias–variance trade-off"
https://www.pnas.org/doi/abs/10.1073/pnas.1903070116

More data often beats better algorithms

EXPERT OPINION

Contact Editor: **Brian Brannon,** bbrannon@computer.org

The Unreasonable Effectiveness of Data

Alon Halevy, Peter Norvig, and Fernando Pereira, *Google*

Large Language Models in Machine Translation

Thorsten Brants Ashok C. Popat Peng Xu Franz J. Och Jeffrey Dean

Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94303, USA
{brants,popat,xp,och,jeff}@google.com

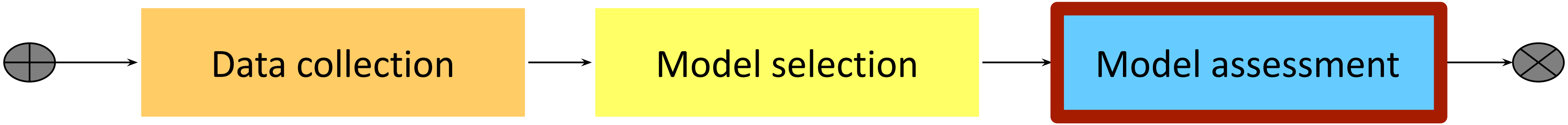
Abstract

This paper reports on the benefits of large-scale statistical language modeling in machine translation. A distributed infrastructure is proposed which we use to train on up to 2 trillion tokens, resulting in language models having up to 300 billion n-grams. It is capable of providing smoothed probabilities for fast, single-pass decoding. We introduce a new smoothing method, dubbed *Stupid Backoff*, that is inexpensive to train on large data sets and approaches the quality of Kneser-Ney Smoothing as the amount of training data increases.

How might one build a language model that allows scaling to very large amounts of training data? (2) How much does translation performance improve as the size of the language model increases? (3) Is there a point of diminishing returns in performance as a function of language model size?

This paper proposes one possible answer to the first question, explores the second by providing learning curves in the context of a particular statistical machine translation system, and hints that the third may yet be some time in answering. In particular, it proposes a *distributed* language model training and deployment infrastructure, which allows direct and efficient integration into the hypothesis-search algorithm rather than a follow-on re-scoring phase.

Classification pipeline



Model assessment

- Model assessment is the goal of estimating the performance of a fixed model (i.e., the best model found during model selection)
- Ideally under real-world conditions
- Use held-out test set that you've never seen during training

Useful reads

Machine Learning that Matters

Kiri L. Wagstaff KIRI.L.WAGSTAFF@JPL.NASA.GOV
Jet Propulsion Laboratory, California Institute of Technology, 4800 Oak Grove Drive, Pasadena, CA 91109 USA

A Few Useful Things to Know about Machine Learning

Pedro Domingos
Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195-2350, U.S.A.
pedrod@cs.washington.edu

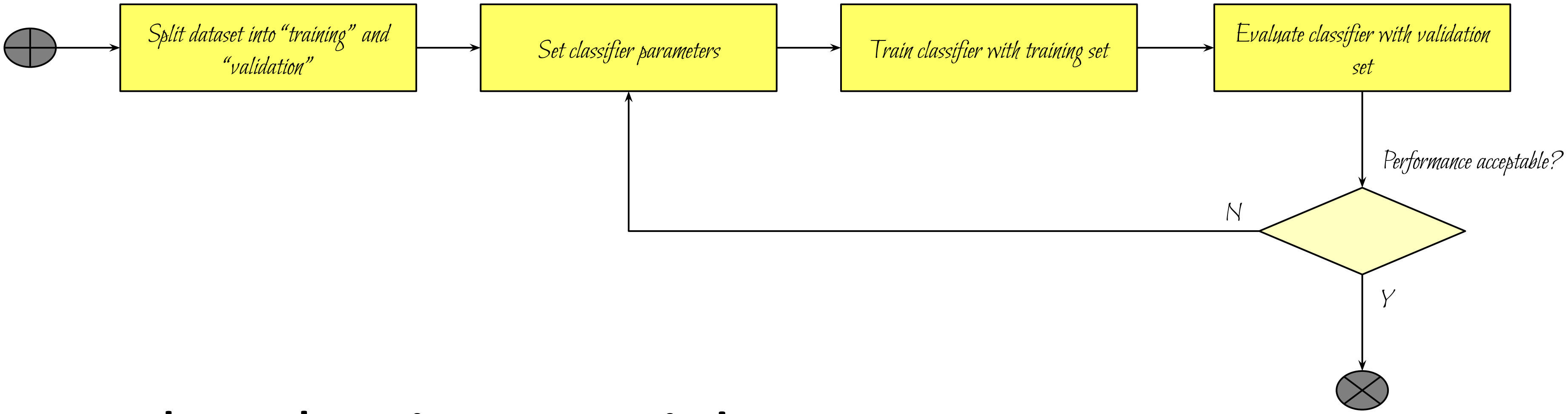
slides

Feedback

Give us feedback on this lecture here:
https://go.epfl.ch/ada2022-lec8-feedback

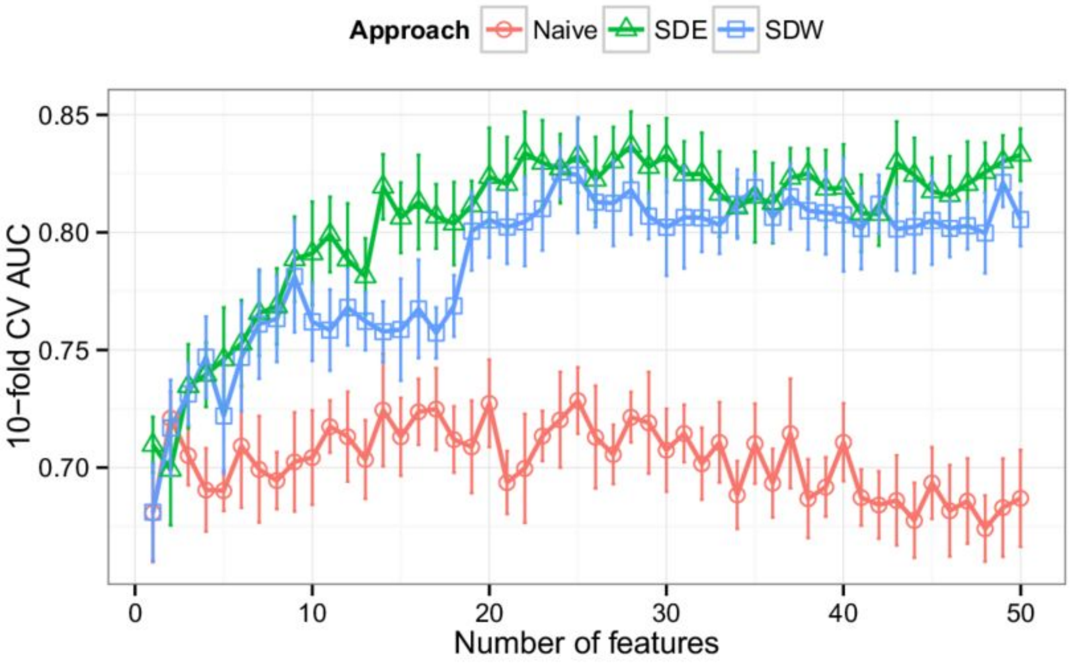
- What did you (not) like about this lecture?
- What was (not) well explained?
- On what would you like more (fewer) details?
- Where is Pumpkin Pete?
- …

Model selection

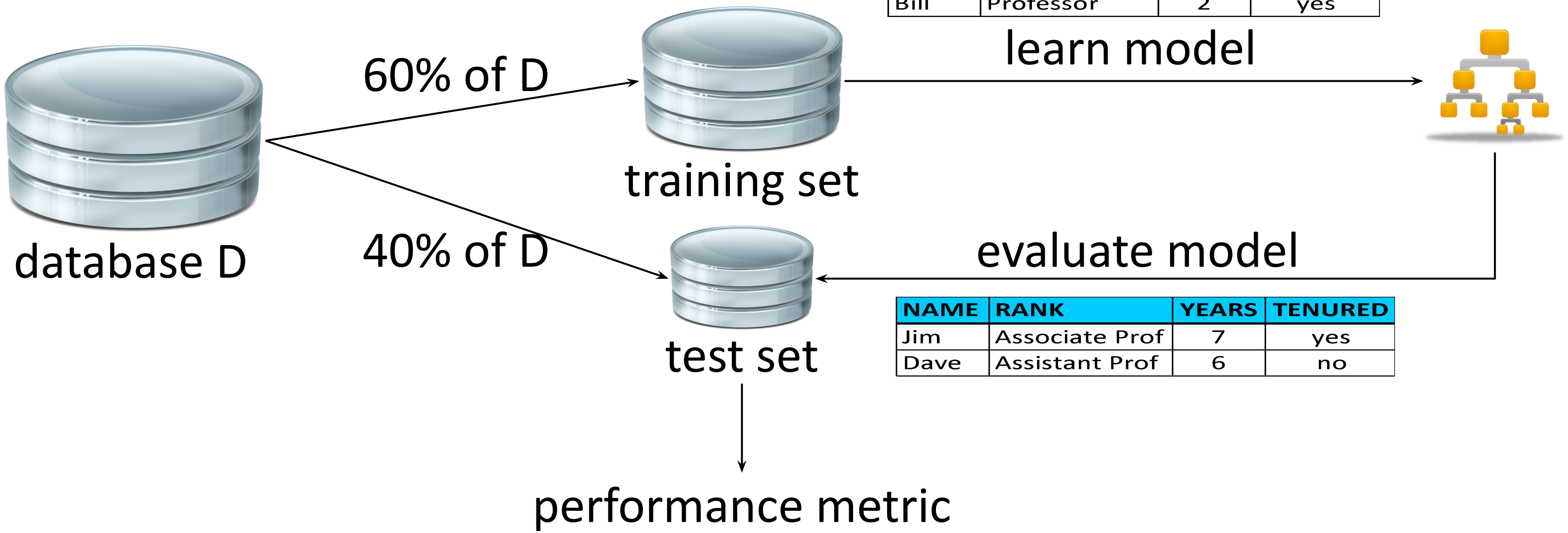


Need evaluation metric!

Fwd-selected features vs. performance



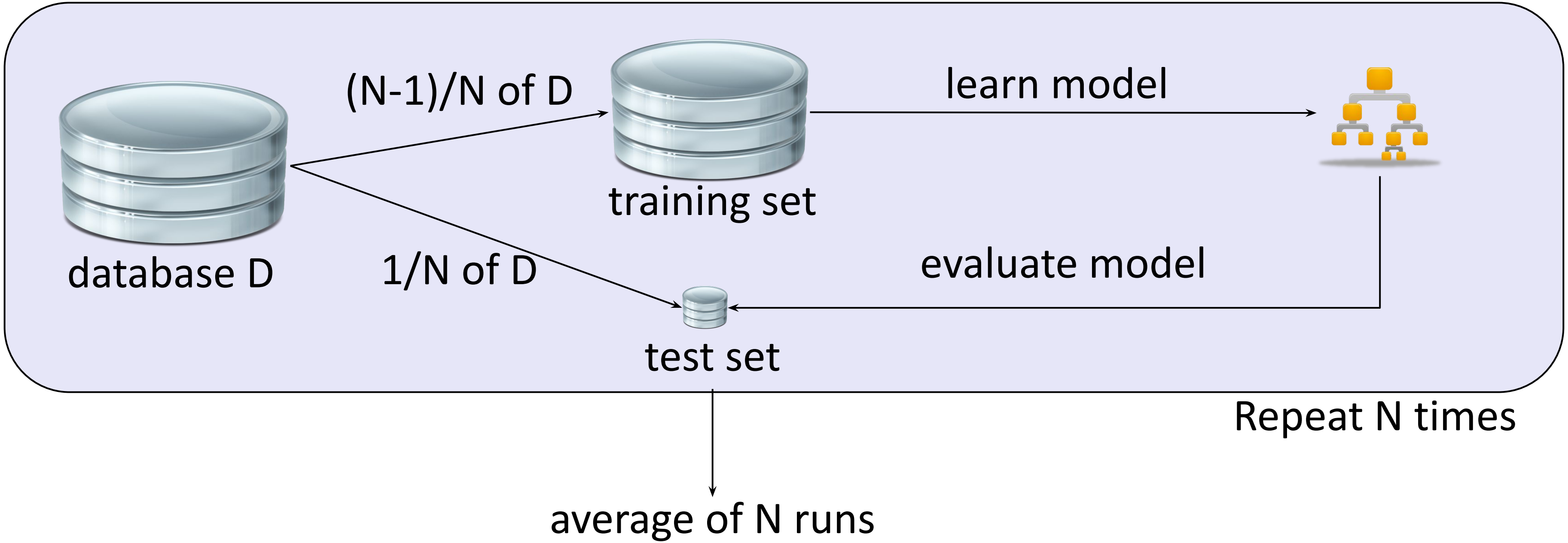
Training and testing with heaps of data



NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes

NAME	RANK	YEARS	TENURED
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no

Data-efficient training and testing: Leave-one-out cross-validation



Data-efficient training and testing: k-fold cross validation