5. CLASSIFICATION METHODOLOGY

Example: Credibility Evaluation

Internet has become a primary data source

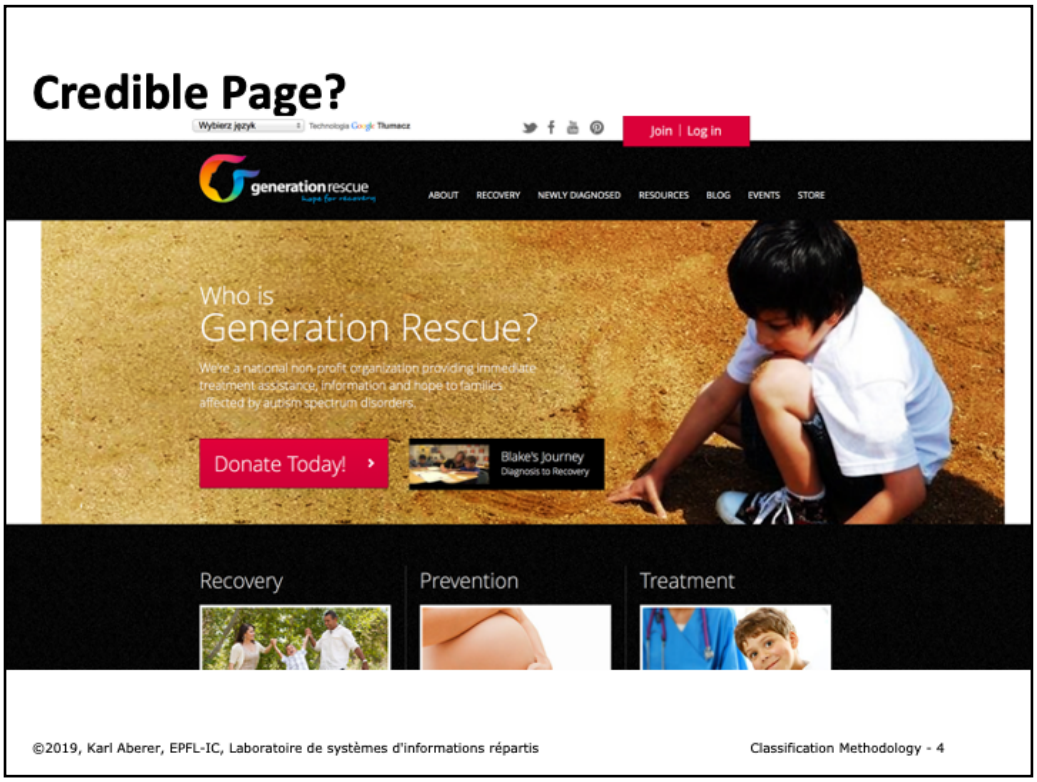
- Data is sometimes questionable, misleading or even erroneous
- Actions taken on the basis of incorrect data can have serious consequences

A classification problem!

Imagine we would like to solve the problem identifying fake news and fake information in the Web. This is definitely an important problem in many domains, like health, politics, eDemocracy. Evaluating fake information is a complex problem with many facets. For example, it concerns both the question of whether the sources of information are on the one hand well-intentioned, and on the other hand knowledgeable. To evaluate these factors we can consider properties both of the content that is published, but also information about the authors. So it constitutes a complex classification problem. In the following, we will describe of how such a problem could be approached, what considerations have to be taken and what pitfalls can occur.

Credible Page?

Even for humans credibility evaluation is a difficult problem: is this page credible?



Or this one? Which of the two is more credible?

Classification Pipeline

Performing a data analysis project …

… is more than knowing and using a classification algorithm

Main steps

1. Data collection and preparation
   - Domain knowledge and understanding essential
2. Model training, selection and assessment
   - Understanding potential and limits of machine learning essential

Tackling a classification problem, such as credibility evaluation requires more than just the knowledge of classification algorithms. First of all, even before starting to learn, an important task, and of the most labor-intensive is the collection and preparation of the data. This step requires in particular a good understanding of the problem domain, the questions that need to be answered and the meaning of the data that is available. Only after this step is performed, in a second step, the learning of the classification model per se can be performed. In this second step it is important to understand not only the working of the different algorithms, but also more largely, the potential and limits of learning a classification function, how the quality of learning depends both on the choice of the model and the data.

Data Collection and Preparation

1. Feature identification
2. Labelling
3. Discretization
4. Feature selection
5. Feature normalization

In the following we will first discuss different relevant aspects when preparing the data …

Model Training

1. Selecting performance metrics
2. Model selection
3. Organizing training and test data

… and those aspects that are relevant for training a model.

DATA COLLECTION AND PREPARATION

1. Feature Identification

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

The first step is collecting data related to the classification task. This step implies the definition of the attributes (or features) that describe a data item and the class label. In general, domain knowledge is needed to know which attributes are relevant and potentially useful for the classification task, as well as to label the data items. The choice of the class labels is also closely related to the problem that should be solved by the classification task, i.e. it represents the question to be answered.

In the case of credibility evaluation we will quickly identify a number of features that are both related to the content of documents, and those that are related to the authors of documents. Performing this step, requires first to have the fundamental insight that both content and social features are of relevance, and then to know what typical features are available, e.g. in a Web document, or about an author in a social network, as well as which of these features can be extracted or acquired.

Features

Different types of features

- Numerical (e.g., age, temperature ...)
- Ordinal (e.g., phone code ...)
- Categorical (e.g., student, weather ...)

Some classifiers require categorical features

- Discretisation

One important task when identifying features is to distinguish them by their type: numerical, ordinal or categorical. The type of the feature has an impact on the type of classifier that can be used, as some can work only with one type of feature (e.g. categorical features). Therefore, it might be necessary to transform features into another space. We will show examples later.

2. Labelling

Collecting lot of data is easy

Labelling data is time consuming, expensive, difficult and sometimes even impossible

Expert in diets is needed

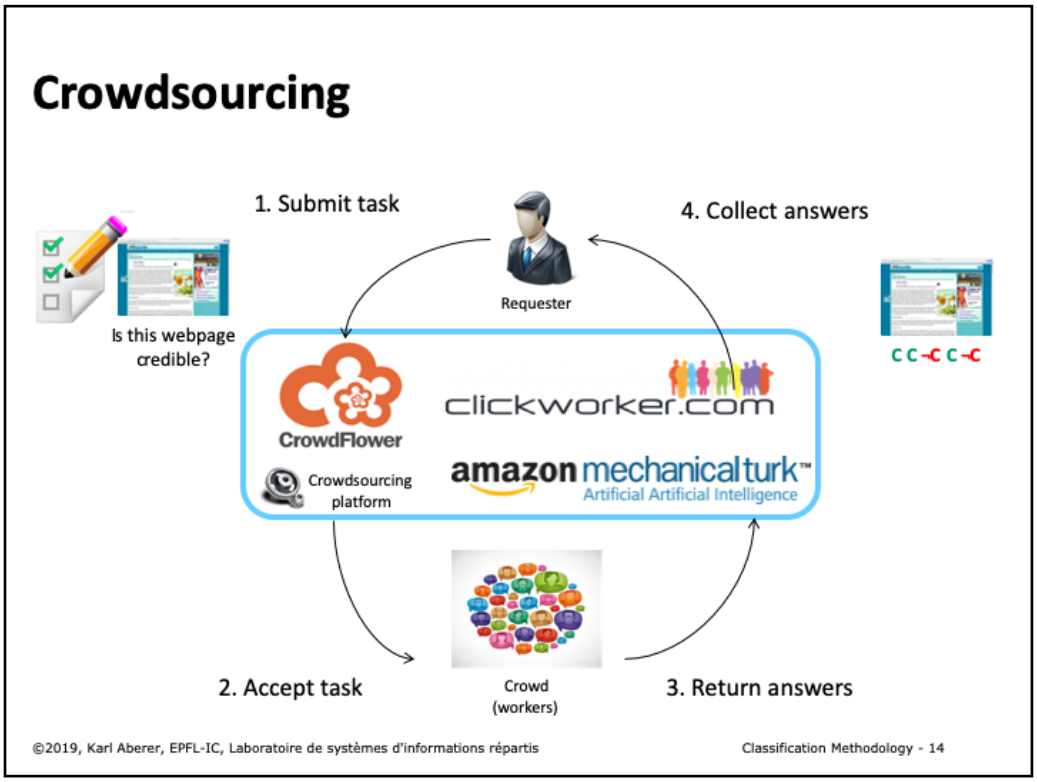
Once the type of features and labels is determined, the big question is how to obtain data WITH LABELS. For example, in the case of building classifiers for credibility evaluation we would first need a corpus of documents where the document have been labelled as being credible or not. In many cases such data is not readily available, and also cannot be easily obtained. In the case of credibility evaluation the problem is complicated by the fact that often specialized expert knowledge is necessary to decide whether data is credible or not, and sometimes even experts would not agree on some questions. A nice example for this would be a document that states that climate change is occurring. This fact is not undisputed as we know.

How to Obtain Labels?

1. Ask experts or do it yourself
   - Expensive, boring, low volume
2. Ask the crowd (crowd-sourcing)
   - Less expensive, popular, unreliable
3. Find some complementary information sources (distant learning)
   - Works in some cases, but …

What to do if no labelled data is available? There exist three main options:

1. We label the data ourselves manually (which is a lot of work and often boring) or we ask experts (or students) to such work, which can be expensive
2. We use crowd-sourcing, i.e. we ask non-experts at large scale. This method has recently become very popular, but has to deal with the issues that non-experts can provide unreliable information
3. We can use some other information sources, that provide labels. For example, if we have text documents, we could identify specific names (e.g. Obama) and find then in databases other information about the name (e.g. that he is president) and use this to label the document (e.g. that this is a document talking about presidents).



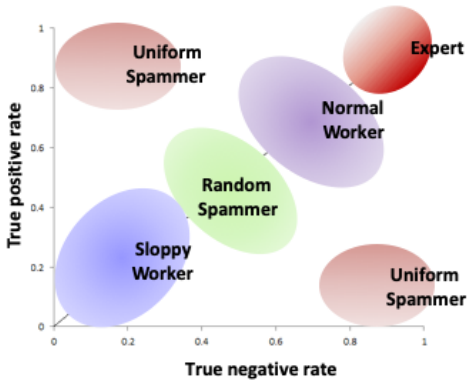
Crowd-sourcing has become popular with platforms such as Amazon mechanical turk and is today widely used for data labelling. Here we describe how crowdsourcing works

- The requester creates a task to be performed by the crowd (= workers). For example, a collection of webpages to be labelled as Credible or Not credible
- The task is submitted to a crowdsourcing platform
- The platform distributes the task to the interested workers that provide their answers; the workers receive a payment for their work
- The requester collects the answers for further analysis

Different Types of Crowd-Workers

Truthful
- Expert
- Normal

Untruthful
- Sloppy
- Uniform spammer
- Random spammer



One of the problems with crowd-sourcing platforms is that not all the workers are equally good . We can have different types of “bad” workers:

- Sloppy Worker: gives many wrong answers due to limited knowledge or misunderstanding of the question
- Random Spammer: gives random answers for any question (e.g. to save time)
- Uniform Spammer: gives the same answer for every question (e.g. to save time)

The figure illustrates the outcome of work for the different types of workers, if a questions allows for two answers (binary category). The expert is in the upper right corner as he has the highest true positive and true negative rates, that means, he detects both positive and negative samples with the highest reliability.

Answer Aggregation Problem

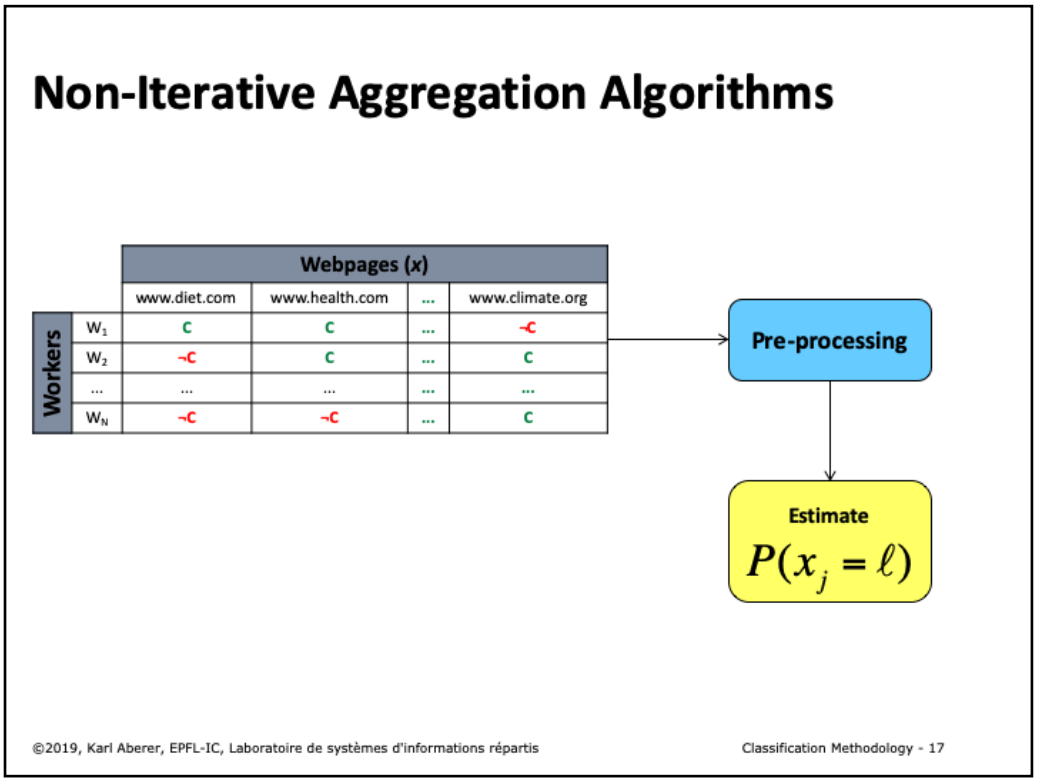
Crowd (workers)

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

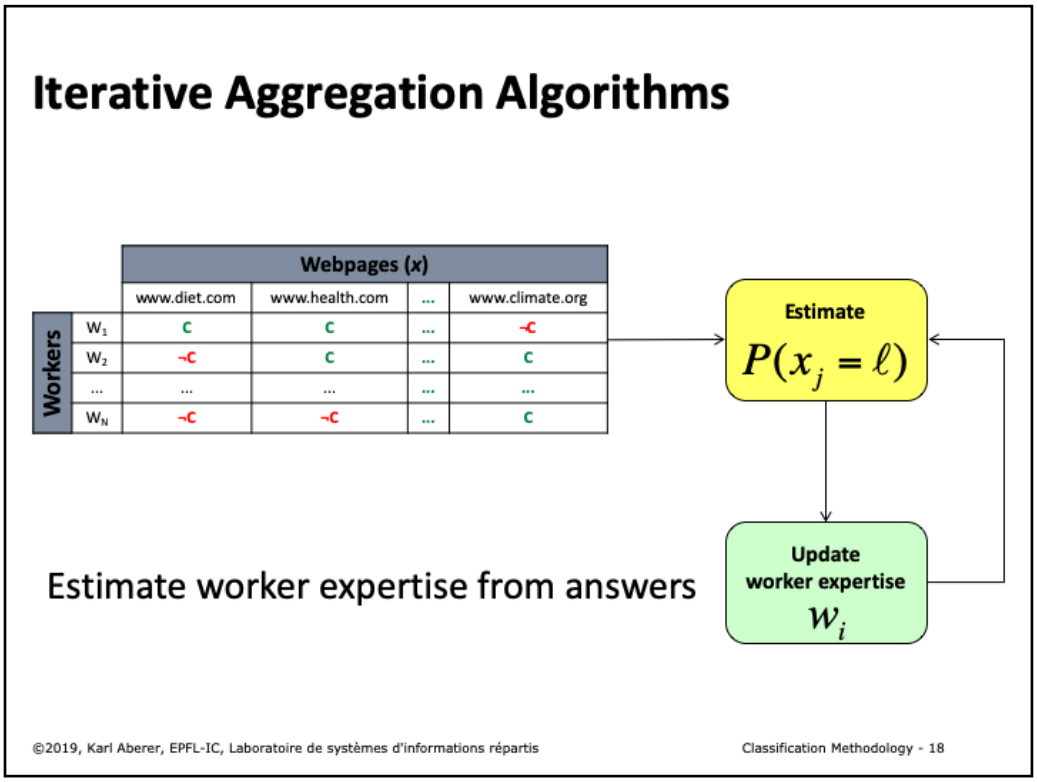
Aggregation

www.diet.com	C

Given that the labelling provided by workers might not coincide for a given question, the requester is faced with the problem of aggregating the (possibly conflicting) answers on a single label.



There are two main classes of answer aggregation algorithms: non-iterative and iterative. Non-iterative algorithms take the matrix of answers provided by the workers, pre-process it and produce an estimate of the probability of the most likely answer to be correct (the most likely correct label for a webpage in our example of credibility evaluation).



Iterative algorithms take the matrix of answers provided by the worker and produce an estimate of the probability that a webpage X is labelled by label L. With this estimate they update the expertise of each worker, that is, a metric that indicates how good is the worker at performing the labelling task. With this new information, the probability that a webpage X is labelled by label L is estimated again. This cycle continues until convergence.

Majority Decision (MD)

Non-iterative aggregation algorithm

- No pre-processing step
- Estimate $P(x_j = \ell)$ as

$$P(x_j = \ell) = \frac{1}{N} \sum_{i=1}^{N} (1 \mid a_i(x_j) = \ell)$$

x_j	webpage to label
N	number of workers
ℓ	label
$a_i(x_j)$	answer of worker *i* to webpage x_j

The MD algorithm is a very fast and appropriate non-iterative aggregation algorithm. It is very sensitive to spammers, since these workers have the same weight in the voting decision as good workers.

Honey Pot (HP)

Non-iterative aggregation algorithm

- Pre-processing step
  - Insert webpages $\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_k$ for which the labels $\hat{\ell}_1, \hat{\ell}_2, \ldots, \hat{\ell}_k$ are known
  - Remove workers and corresponding answers that fail at correctly labelling more than *m%* of webpages (either spammer or sloppy worker)
- Same decision rule as MD

The HP algorithm is an extension of the MD algorithm. It is also fast, but more robust to spammers than MD, thanks to the honey pot traps that helps to filter out spammers. However, trapping questions are not always available, they are often constructed subjectively. Thus truthful and good workers might be misidentified as spammers if trapping questions are too difficult or workers do not think in the same way as the requester.

Expectation Maximisation (EM)

Iterative aggregation algorithm

Iterates in two steps

1. E-Step: estimate the labels from the answers of workers
2. M-Step: estimate the reliability of workers from the consistency of answers

In the EM algorithm, the probability that a webpage X is labelled by label L is computed as a weighted vote. It is the main example of an iterative aggregation algorithm. The idea is to reduce the expertise of spammers (therefore their weight in the voting is less important) during the iteration and to increase the expertise of good workers. The weighting is determined by checking the consistency of answers among the different workers.

Expectation Maximisation (EM)

(E) step: estimate $P(x_j = \ell)$ as

$$P(x_j = \ell) = \frac{1}{\sum_{i=1}^{N} w_i} \sum_{i=1}^{N} (w_i \mid a_i(x_j) = \ell)$$

(M) step: update the expertise w_i as

$$w_i = \frac{1}{M} \sum_{j=1}^{M} (1 \mid a_i(x_j) = \arg\max_{\ell} P(x_j = \ell))$$

w_i expertise of worker *i*

M number of webpages to label

The EM algorithm proceeds in two steps: in the E-Step, the probability of the label is estimated as a weighted vote among all workers, i.e. the normalized sum of all weights of workers voting for a specific label. In the M step, the expertise of each worker is updated as the number of times worker i was correct at selecting the most probable label for object x (based on the probability determined in the E-Step). Since in the M step the worker's weights chance, in the E-Step new probabilities will be computed and so on. The EM algorithm is very accurate and very robust to spammers, but slow to converge. Usually the initial weights of workers are chosen to be equal (unless pre-existing knowledge exist).

Expectation Maximisation (EM)

		Webpages (x)					w_i
		diet.com	health.com	dimate.nasa.gov	climate.org	climatechange.net	
Workers (w)	W1	1	1	1	0	1	1
	W2	0	1	1	1	0	1
	W3	0	1	1	1	0	1
	W4	0	0	1	1	1	1

(E) step: $P(x_j = \ell)$

P(x=0)	0.75	0.25	0	0.25	0.5
P(x=1)	0.25	0.75	1	0.75	0.5

$$P(x_j = \ell) = \frac{1}{\sum_{i=1}^{N} w_i} \sum_{i=1}^{N} (w_i \mid a_i(x_j) = \ell)$$

Expectation Maximisation (EM)

(M) step:

		Webpages (x)					w_i
		diet.com	health.com	dimate.nasa.gov	climate.org	climatechange.net	
Workers (w)	W1	1	1	1	0	1	2/5
	W2	0	1	1	1	0	5/5
	W3	0	1	1	1	0	5/5
	W4	0	0	1	1	1	3/5

(E) step: $P(x_j = \ell)$

P(x=0)	0.75	0.25	0	0.25	0.5
P(x=1)	0.25	0.75	1	0.75	0.5

$$w_i = \frac{1}{M}\sum_{j=1}^{M}(1 \mid a_i(x_j) = \arg\max_{\ell} P(x_j = \ell))$$

Expectation Maximisation (EM)

(M) step:

		Webpages (x)					w_i
		diet.com	health.com	dimate.nasa.gov	climate.org	climatechange.net	
Workers (w)	W1	1	1	1	0	1	2/5
	W2	0	1	1	1	0	5/5
	W3	0	1	1	1	0	5/5
	W4	0	0	1	1	1	3/5

(E) step:

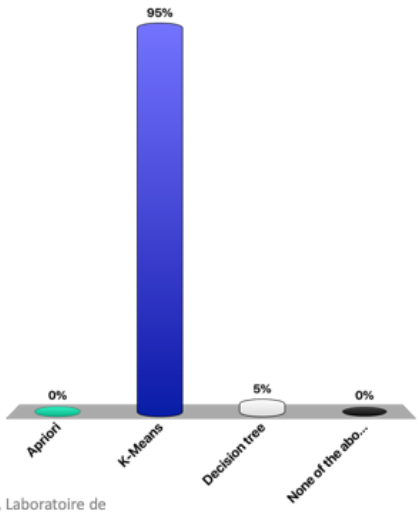
$P(x_j = \ell)$

P(x=0)	0.867	0.2	0	0.133	0.667
P(x=1)	0.133	0.8	1	0.867	0.333

$$P(x_j = \ell) = \frac{1}{\sum_{i=1}^{N} w_i} \sum_{i=1}^{N} (w_i \mid a_i(x_j) = \ell)$$

Which data mining algorithm belongs to the Expectation-Maximization class?

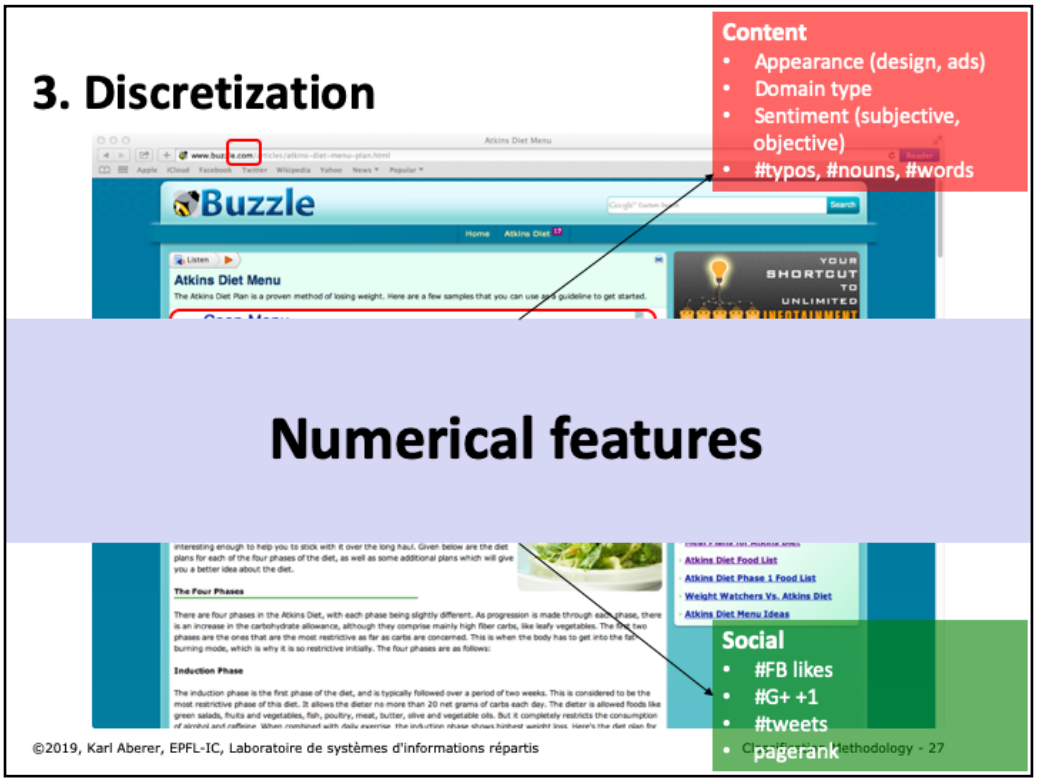
A. Apriori
B. K-Means
C. Decision tree
D. None of the above



Answer is B

the E-step, where each object is assigned to the centroid such that it is assigned to the most likely cluster.

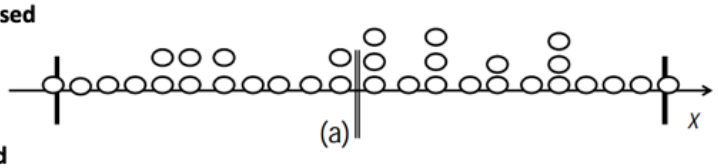
the M-step, where the model (=centroids) are recomputed (= least squares optimization).



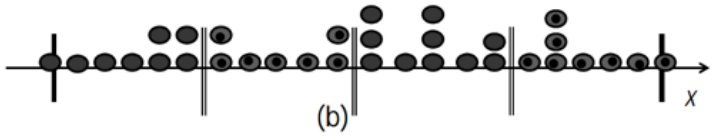
Many features found in our example are numerical, e.g. number of likes and followers, pagerank etc. If we would like to apply a classifier that works only on categorical features (e.g. a basic decision tree induction), we have to first discretize those features.

Discretisation Methods

Unsupervised

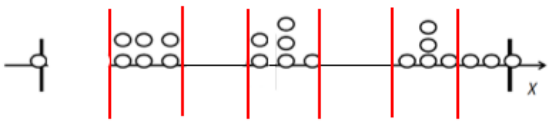


Supervised



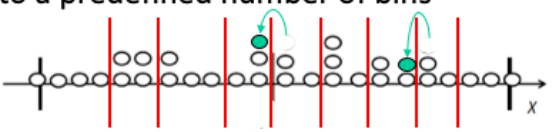
Deciding on the number of intervals or bins without using (a) and after using (b) class information with red and blue ovals (black dot inside the blue).

Unsupervised Discretisation



Equal width

- Divide the range into a predefined number of bins

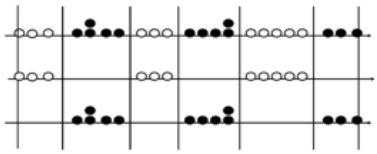


Equal frequency

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

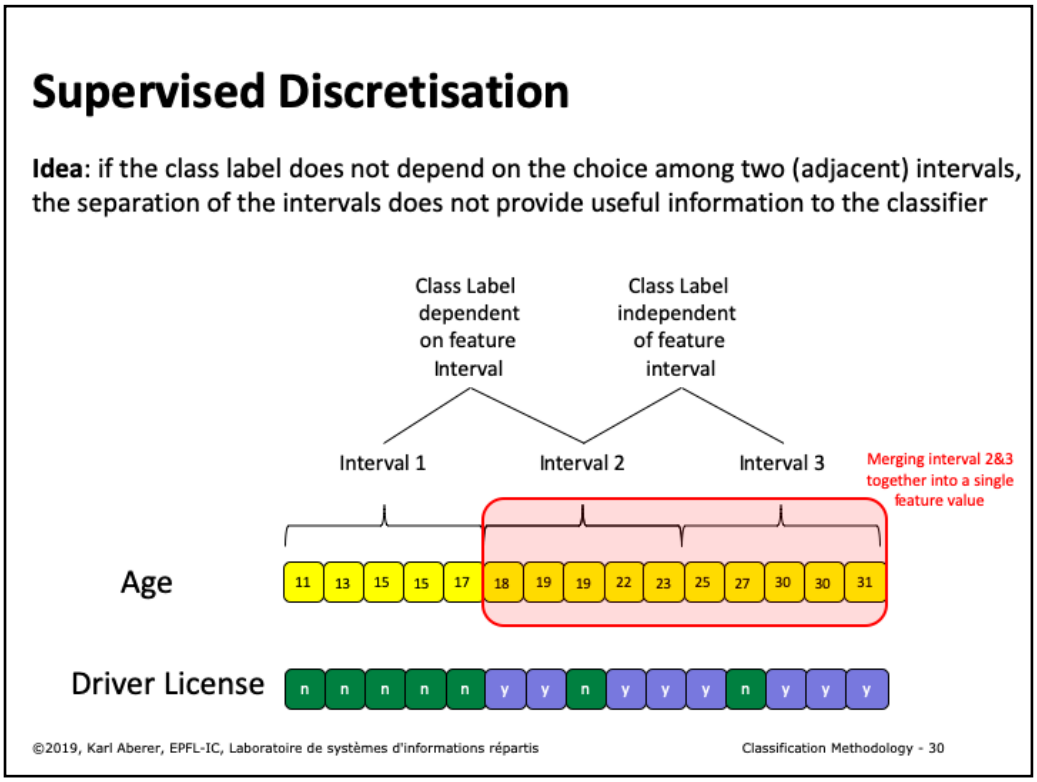
Clustering

- Use any suitable clustering method for multi-dimensional data and assign one feature value per cluster



A first approach to discretization is unsupervised discretization. Unsupervised discretisation do not take class information (labels) into account. These methods have the following strengths and weaknesses:

- The obvious weakness of the equal-width method is that in cases where the feature values are not distributed uniformly, a large amount of important information can be lost after the discretization process, e.g. with many very sparsely populated bins.
- For equal-frequency discretization, many occurrences of the same value could cause those occurrences to be assigned into different bins, which does not make sense. This problem could be addressed by merging neighbouring bins that contain duplicate values.
- The advantage of using clustering as discretisation method is that it can be performed on all the features at the same time, capturing in this way possible interdependencies of the features. Sometimes discretisation can even improve the performance of algorithms that do not need discretisation.



Assuming a discretization is given. We may then ask the question, whether the class labels depends on the interval or not. In the example, the fact of having a driver license clearly depends on whether the age is above or below 18 years. Therefore it makes sense to keep these intervals apart, as the feature of belonging to one of the intervals influences the label. On the other hand, the fact of belonging either to interval 2 or 3 has no incidence on the probability of having a drivers license. Therefore these intervals can be merged into a single feature value

Supervised Discretisation

Independence test: χ^2 statistics

	Class1	Class2	*sum*
Interval 1	$O_{11} = n_{11}$ $E_{11} = (R1 \times C1)/N$	$O_{12} = n_{12}$ $E_{12} = (R1 \times C2)/N$	R1
Interval 2	$O_{21} = n_{21}$ $E_{21} = (R2 \times C1)/N$	$O_{22} = n_{22}$ $E_{22} = (R2 \times C2)/N$	R2
sum	C1	C2	N

O_{ij} observed frequency
E_{ij} expected frequency

$$\chi^2 = \sum_{i=1,2} \sum_{j=1,2} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

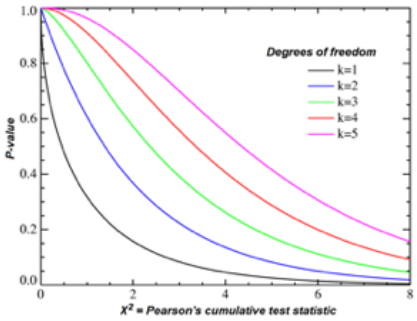
In order to test the independence of two adjacent intervals, one can use the χ^2 statistics. E_{ij} is the expected frequency of the number of distinct values in the ith interval belonging to the jth class, while n_{ij} is the observed frequency. N is the total number of values.

Supervised Discretisation

Null hypothesis: Assumes that the class label is independent of the feature intervals

If $P(\chi^2 \mid DF = 1) > 0.05$ (independent), merge the intervals

DF = degrees of freedom = (#rows-1)*(#cols – 1).



Percentage Points of the Chi-Square Distribution

Degrees of Freedom	Probability of a larger value of x^2								
	0.99	0.95	0.90	0.75	0.50	0.25	0.10	0.05	0.01
1	0.000	0.004	0.016	0.102	0.455	1.32	2.71	3.84	6.63
2	0.020	0.103	0.211	0.575	1.386	2.77	4.61	5.99	9.21
3	0.115	0.352	0.584	1.212	2.366	4.11	6.25	7.81	11.34
4	0.297	0.711	1.064	1.923	3.357	5.39	7.78	9.49	13.28
5	0.554	1.145	1.610	2.675	4.351	6.63	9.24	11.07	15.09
6	0.872	1.635	2.204	3.455	5.348	7.84	10.64	12.59	16.81
7	1.239	2.167	2.833	4.255	6.346	9.04	12.02	14.07	18.48
8	1.647	2.733	3.490	5.071	7.344	10.22	13.36	15.51	20.09
9	2.088	3.325	4.168	5.899	8.343	11.39	14.68	16.92	21.67
10	2.558	3.940	4.865	6.737	9.342	12.55	15.99	18.31	23.21
11	3.053	4.575	5.578	7.584	10.341	13.70	17.28	19.68	24.72
12	3.571	5.226	6.304	8.438	11.340	14.85	18.55	21.03	26.22
13	4.107	5.892	7.042	9.299	12.340	15.98	19.81	22.36	27.69
14	4.660	6.571	7.790	10.165	13.339	17.12	21.06	23.68	29.14
15	5.229	7.261	8.547	11.037	14.339	18.25	22.31	25.00	30.58
16	5.812	7.962	9.312	11.912	15.338	19.37	23.54	26.30	32.00
17	6.408	8.672	10.085	12.792	16.338	20.49	24.77	27.59	33.41
18	7.015	9.390	10.865	13.675	17.338	21.60	25.99	28.87	34.80
19	7.633	10.117	11.651	14.562	18.338	22.72	27.20	30.14	36.19
20	8.260	10.851	12.443	15.452	19.337	23.83	28.41	31.41	37.57
22	9.542	12.338	14.041	17.240	21.337	26.04	30.81	33.92	40.29
24	10.856	13.848	15.659	19.037	23.337	28.24	33.20	36.42	42.98
26	12.198	15.379	17.292	20.843	25.336	30.43	35.56	38.89	45.64
28	13.565	16.928	18.939	22.657	27.336	32.62	37.92	41.34	48.28
30	14.953	18.493	20.599	24.478	29.336	34.80	40.26	43.77	50.89
40	22.164	26.509	29.051	33.660	39.335	45.62	51.80	55.76	63.69
50	27.707	34.764	37.689	42.942	49.335	56.33	63.17	67.50	76.15
60	37.485	43.188	46.459	52.294	59.335	66.98	74.40	79.08	88.38

Given the χ^2 value, it is necessary to consult a χ^2 distribution with a degree of freedom = (#rows-1)*(#cols – 1). If the probability of the computed χ^2 value is greater than a threshold p (typically 0.05 or 0.01), the two intervals are inde pendent and can be merged (Pearson's chi squared test).

In statistics, the number of **degrees of freedom** is the number of values in the final calculation of a statistic that are free to vary.

Pearson's chi-squared test (χ^2) is a statistical test applied to sets of categorical data to evaluate how likely it is that any observed difference between the sets arose by chance.

Example

Driving License?

observed	No	Yes	SUM
Interval 1	51	0	51
Interval 2	1	50	51
SUM	52	50	102

expected	No	Yes
Interval 1	26.00	25.00
Interval 2	26.00	25.00

chisquare statistics	No	Yes
Interval 1	24.04	25.00
Interval 2	24.04	25.00

chisquare	98.0769231
P(chi | df = 1)	4.0244E-23

Percentage Points of the Chi-Square Distribution

Degrees of Freedom	Probability of a larger value of x^2								
	0.99	0.95	0.90	0.75	0.50	0.25	0.10	0.05	0.01
1	0.000	0.004	0.016	0.102	0.455	1.32	2.71	3.84	6.63

p-value of 4.0244e-23 is less than 0.05 → we reject the null hypothesis
(there is a dependence → no merge)

Example

Driving License?

observed	No	Yes	SUM
Interval 2	49	51	100
Interval 3	50	51	101
SUM	99	102	201

expected	No	Yes
Interval 2	49.25	50.75
Interval 3	49.75	51.25

chisquare statistics	No	Yes
Interval 2	0.00131	0.00127
Interval 3	0.00129	0.00126

chisquare	0.00512601
P(chi | df = 1)	0.94292328

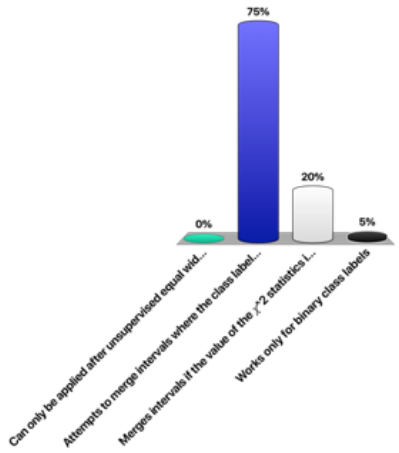
Percentage Points of the Chi-Square Distribution

Degrees of Freedom	Probability of a larger value of x^2								
	0.99	0.95	0.90	0.75	0.50	0.25	0.10	0.05	0.01
1	0.000	0.004	0.016	0.102	0.455	1.32	2.71	3.84	6.63

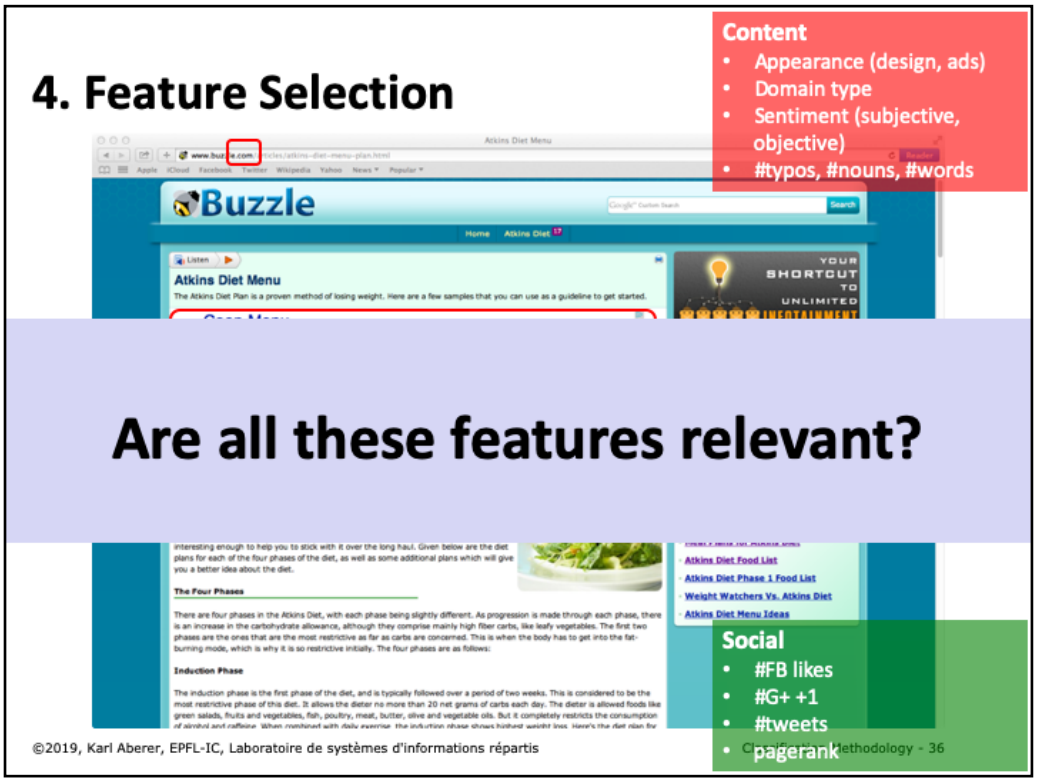
p-value of 0.9429 is greater than 0.05 → we accept the null hypothesis
(there are independent → we can merge)

Supervised discretization …

A. Can only be applied after unsupervised equal width discretization
B. Attempts to merge intervals where the class label does not depend on the attribute value distribution
C. Merges intervals if the value of the χ^2 statistics is above 0.05
D. Works only for binary class labels



Answer is B



Although some features have been considered as relevant by the data analyst and added to the dataset, not all of them are necessarily relevant, and they can be even harmful if they represent noise. Thus an important step is to remove irrelevant features, so as to improve the classification performance as well as the computation speed.

Irrelevant or partially relevant features can negatively impact model performance.

Feature Selection

Reducing the number of N features to an optimal subset of M features, M < N

There are $\binom{N}{M}$ possible subsets

Approaches

- Filtering: consider features as independent
- Wrapper: consider dependencies among features

There are N choose M possible subset (choosing a subset of M features from a fixed set of N features).

Exhaustive exploration of all possible subset of features is impossible, therefore some heuristics must be used.

We will discuss two classes of approaches, filtering and wrapping, that differ in the assumptions made on feature dependency.

Feature Selection: Filtering

Filtering: rank features according to their predictive power and 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

Filtering starts from the assumption that features are independent. Thus the predictive power of each feature can be evaluated independently, and finally the top features are chosen. It also assumes that the choice of features is independent of the classifier used. Thus it can be performed without interacting with the classifier, and potentially as a result revise the selection of the features.

Ranking Features

χ^2 statistics (as before) for the n feature values

	Class c_1	Class c_2	*sum*
Value f_1			
Value f_2			
...			
Value f_n			
sum			

$P(\chi^2 \mid DF = n - 1)$ gives a rank measure

For ranking features we can use the same idea that was used in the discretization of continuous features. We perform an independence test between the feature values and the class labels. Since we now consider multiple feature values (and a binary classification) we have instead of 1 degree of freedom, n-1 degrees of freedom, that have been considered in the independence test.

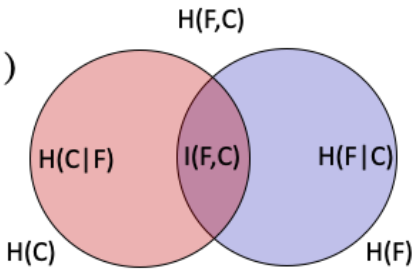
Information-theoretic approach

Mutual information between feature F and class label C

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

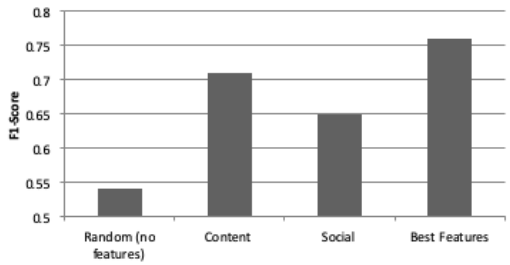
$$H(F) = -\sum_i P(f_i) \log_2 P(f_i)$$

$$H(F,C) = -\sum_i \sum_j P(f_i, c_j) \log_2 P(f_i, c_j)$$



Alternatively, we measure the mutual information between a class label C and a feature F. Mutual information measures the information that F and C share. It measures how much knowing one of these variables reduces uncertainty about the other. If I(F,C) = 0, knowing F does not tell anything about C, when I(F,C) is maximal, by knowing F we already know C.

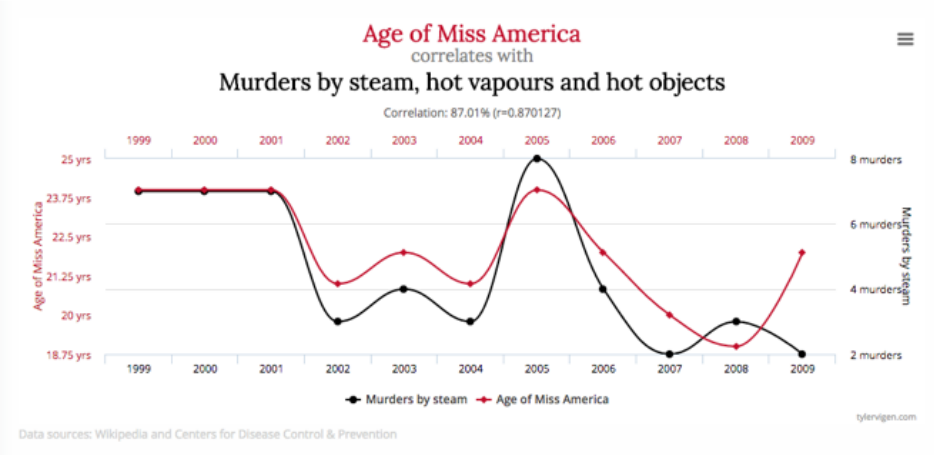
Example: Credibility Features



Here we see the results of choosing the best features for a credibility classifiers. It is for example interesting to observe that many layout related features seem to be good indicators for quality. Evaluations of the resulting classifiers also show that combing the best content and social features delivers finally the best results.

Selection of Features - Pitfalls

Correlation ≠ Causality

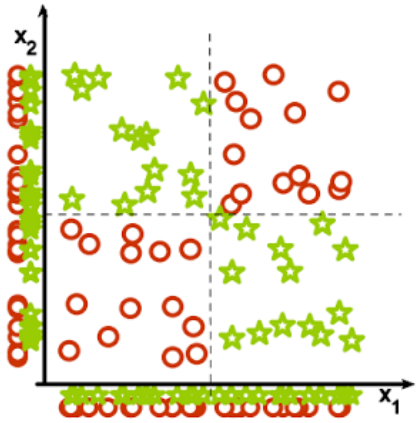


Many more examples: http://www.tylervigen.com/spurious-correlations

When eliminating features, we have to be careful to not emphasize features that are only accidentally correlated with the class labels. In general, correlation does not imply causality, and we can find many examples of so-called spurious correlations illustrating that point. Be aware, that such curious correlations are often used to prove statistically “facts” that or no existent.

Selection of Features - Pitfalls

Collectively relevant features may look individually irrelevant



An important drawback of the filtering approach to feature selection is the independence assumption for features. In fact, it is easy to see, as illustrated by the example above, that combined feature can have a high classification power, whereas each of the features alone have not. This motivates the use of more complex methods for feature selection, such as wrapping.

Selection of Features - Pitfalls

Beware of trusting correlations in a blind way

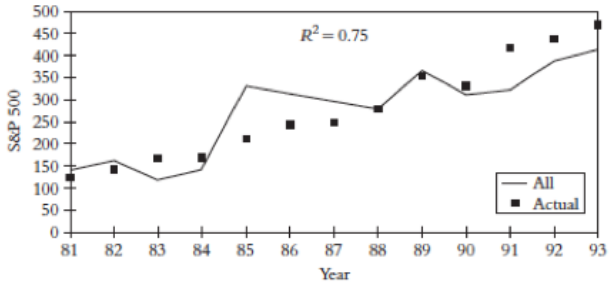


Figure 6.2 Overfitting the S&P 500: butter production in Bangladesh—a single variable that "explains" 75 percent of the S&P's returns.

http://m.shookrun.com/documents/stupidmining.pdf

Overfitting can lead to meaningless "results" in data mining. Here is an example that tries to predict stock markets, based on quite unrelated data.

Selection of Features - Pitfalls

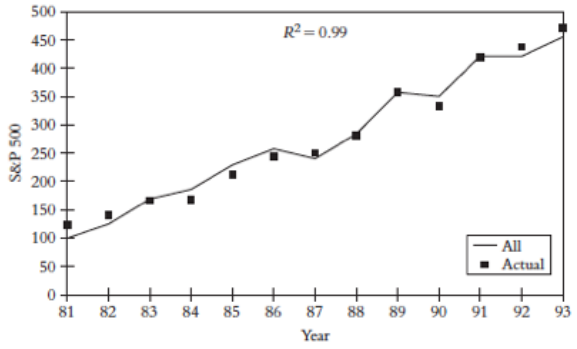


Figure 6.4 Overfitting the S&P 500: butter in Bangladesh and United States, plus U.S. cheese production, as well as sheep population in Bangladesh and United States. Now we're at 99 percent. You can do this as long as you can find data not perfectly correlated with butter, cheese, sheep, and so on. There is no shortage of that.

http://m.shookrun.com/documents/stupidmining.pdf

If sufficient number of time series are found, indeed the fitting can become quite will. Of course, when extending the time series beyond the time interval, results will be bad.

Feature Selection: Wrapping

Iteratively add features

- at each iteration create a classifier for each new feature and evaluate its performance
- Add best feature or stop when no further improvement

Pros

- Interact with the classifier
- No independence assumption

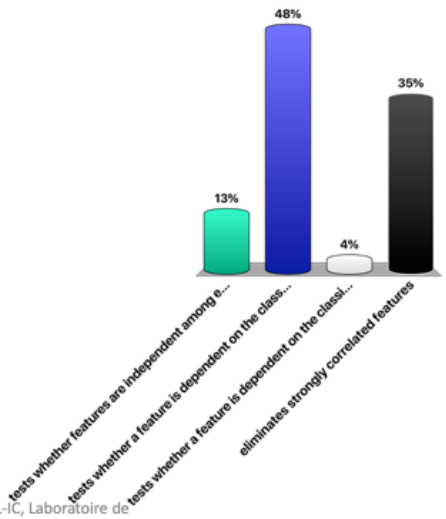
Cons

- Computationally intensive

The wrapper method starts with no features and add the features that provide the highest improvement in classification accuracy The wrapper method can also work the other way around, considering all the features at the beginning and iteratively removing the feature that harms the classification accuracy the least. The method avoids some of the pitfalls of feature filtering, at the price of significantly higher cost.

The filtering approach to feature selection …

A. tests whether features are independent among each other
B. tests whether a feature is dependent on the class label
C. tests whether a feature is dependent on the classifier
D. eliminates strongly correlated features



Answer is B

5. Feature Normalisation

Some classifiers do not manage well features with very different scales

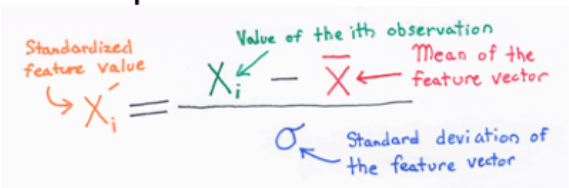
- # followers: 10,000,000
- # tweets: 300

Features with large values dominate the others, and the classifier tend to over-optimise them

Some classifiers and performance evaluation methods are sensitive to the absolute values of the features. Then the features should be normalized to comparable scales.

Standardisation and Scaling

Standardisation: map to a normal distribution N(0, 1)



The new feature x_i' has mean 0 and unit variance

Min-Max Scaling: map to interval [0,1]

Images from https://chrisalbon.com/

Standardization makes the underlying assumption that the (numerical) data is normally distributed. Then it makes sense to map to a normal distribution with mean 0 and variance 1. An alternative is to map directly into a standardized [0,1] interval.

Standardisation vs Scaling

Standardisation

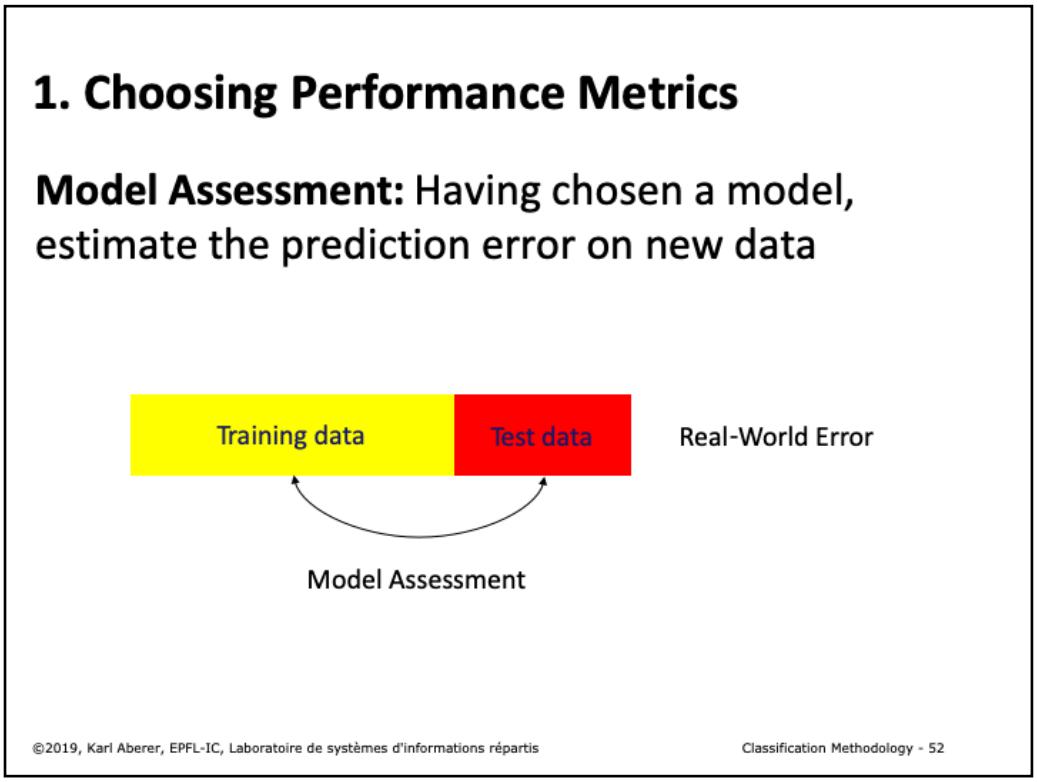
- Assumes that the data has been generated by a Gaussian process (not necessarily true)

Scaling

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

Standardisation assumes that data has been generated by a Gaussian process, which is not always true. Scaling is not good if there are outliers, and almost all datasets have outliers. Such outliers risk to "compress" the relevant data in a very small interval, which would e.g. be harmful of some discretization is applied.

MODEL TRAINING, SELECTION AND ASSESSMENT



Once the dataset is ready, we start to learn a model. For the learnt model we need to evaluate its performance, by testing which error it produces for a test data set. That is why we hold out a subset of the available data as test data, that is not used in the training and is independent of the training data. This step is called model assessment. The error obtained in model assessment is a good estimate of the true error that it will commit once it is “in production”.

Performance Metric for Binary Classification

For categorical binary classification, the usual metrics consider four types of outcomes

Correct results

- True Positive (positive examples classified as positive)
- True Negative (negative examples classified as negative)

Incorrect results

- False Positive (negative examples classified as positive)
- False Negative (positive examples classified as negative)

		Class A	Class B
Classified	A	TP	FP
	B	FN	TN

In order to do model assessment we have to choose the performance metric for evaluating the error. We will present this in the following for the case of binary classification with a categorical label. In such a case we have for possible types of outcomes, true positives and negatives, and false positives and negatives. We represent these outcomes as a matrix.

Accuracy

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

Appropriate metric when

- Classes are not skewed
- Errors have the same importance

The most basic metric is the accuracy, that is, the total correct examples divided by the total examples to be classified. Accuracy is thus in the interval [0,1]. Accuracy is appropriate if the examples are approcimtaely equally split between class A and B, and if FP and FN have the same importance as error.

Accuracy - Pitfall

Classifier 1

Classified	Class: Fraud	Class: ¬Fraud
Fraud	5	10
¬Fraud	5	80

A = 85/100 = 0.85

Always ¬Fraud

Classified	Class: Fraud	Class: ¬Fraud
Fraud	0	0
¬Fraud	10	90

A = 90/100 = 0.90

Here we give an example of skewed classes, where the majority of data available refers to No Fraud, and only few examples belong to the class Fraud. Accuracy as a performance metrics is inappropriate in case of such skewed label distributions. The typical problem is that a trivial classifier that classifies everything as belonging to the majority class, can achieve easily higher accuracies than a classifier that attempts to also correctly classify samples in the minority class.

Which is the “best” classifier?

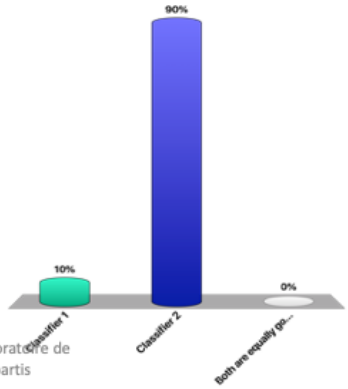
Classifier 1

Classified	Class A	Class B
A	45	20
B	5	30

Classifier 2

Classified	Class A	Class B
A	40	10
B	10	40

A. Classifier 1
B. Classifier 2
C. Both are equally good



Which is the “best” classifier?

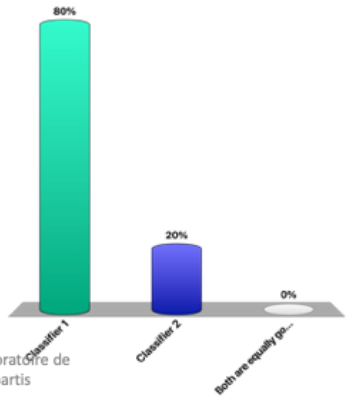
Classifier 1

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

Classifier 2

Classified	Class: Cancer	Class: ¬Cancer
Cancer	40	10
¬Cancer	10	40

A. Classifier 1
B. Classifier 2
C. Both are equally good



Precision and Recall

Precision

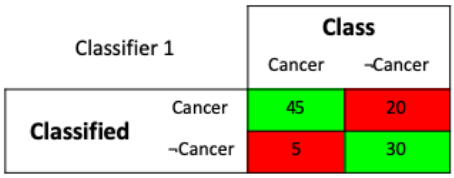
$$P = \frac{TP}{TP + FP}$$

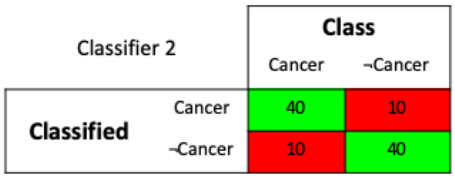
Recall

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

As the example in the previous question illustrates, apart from issues with skewed distributions, accuracy has also an issue when the importance of errors is not equal, e.g. when false negatives are much worse then false positives. In such a case we can use the measures of precision and recall that we have already introduced in the context of information retrieval.

Precision and Recall: Example



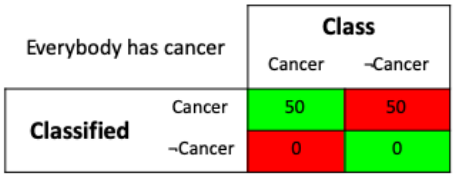


P_1=45/65=0.69

R_1=45/50=0.9

P_2=40/50=0.8

R_2=40/50=0.8



P = 50/100 = 0.5

R = 50/50 = 1

With precision and recall we can better control the kind of results we prefer. For example, for the case of cancer detection we would prefer to have higher recall to miss fewer cancer cases, even if we diagnose erroneously cancer in a few cases. Therefore, classifier 1 would be preferred over classifier 2 (which did better in terms of accuracy). Of course we can increase the recall arbitrarily, e.g. with a trivial classifier diagnosing cancer for everyone. But this also causes that 50% of the patients with no cancer go home worried about their health status

That is why still precision needs to be considered in the evaluation as second measure.

Note that by using precision and recall we make the evaluation “assymmetric”. We focus in the evaluation on the positive class, since higher recall means more positive cases identified.

F-score

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

F-score (or F1-score) $$F1 = 2 \cdot \frac{P \cdot R}{P + R}$$

F-beta score $$F_\beta = \frac{(1+\beta^2)PR}{(\beta^2 P + R)}$$

As in information retrieval, we sometimes would like to compare classifiers using a single measure. In this case we can also use F-Score. F-score is the harmonic mean of precision and recall. Precision and Recall can be differently weighted, if one is more important than the other using F_beta score, where beta is a real number.

F-Score: Example

Classifier 1

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

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

Classifier 2

		Class: Cancer	Class: ¬Cancer
Classified	Cancer	40	10
	¬Cancer	10	40

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

Everybody has cancer

		Class: Cancer	Class: ¬Cancer
Classified	Cancer	50	50
	¬Cancer	0	0

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

With the F1 score still classifier 2 is evaluated as slightly better than classifier 1.

F-beta-Score: Example (beta = 2)

Classifier 1

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

Classifier 2

Classified	Class: Cancer	Class: ¬Cancer
Cancer	40	10
¬Cancer	10	40

$F_1=5*(0.69*0.9)/(4*0.69+0.9) = 0.84$

$F_2=5*(0.8*0.8)/(4*0.8+0.8) =0.8$

Everybody has cancer

Classified	Class: Cancer	Class: ¬Cancer
Cancer	50	50
¬Cancer	0	0

$F=5*(0.5*1)/(4*0.5+1)=0.83$

With the F-beta score, beta=2, we can give more importance to recall and as a result indeed classifier 1 turns out to be the best.

Considering Cost

The "cow case"

- Predict when cows are "in heat"
- Important for fertilization
- Observe different body parameters
- Predictor: never (correct 97% of the time!)

Solution: analyze different types of errors separately and associate (financial) cost and benefits

		Predicted class: yes	Predicted class: no
Actual class	yes	True positive: 30	False negative: -30
	no	False positive: -1	True negative: 1

In the case of cancer we had about equally frequent events (cancer and not cancer) of very different importance. If we have a combination of skewed distribution and importance of errors, the situation is even more complicated. An example of such a case is detecting fertility in cows.

Here also precision and F-beta scores don't necessarily solve the situation. In such cases the « value » of the different classes to be detected needs first to be quantified and based on that the quality of the classifier can be determined in a meaningful way. The weighted costs of the prediction outcomes can then be used to select the most appropriate classifiers.

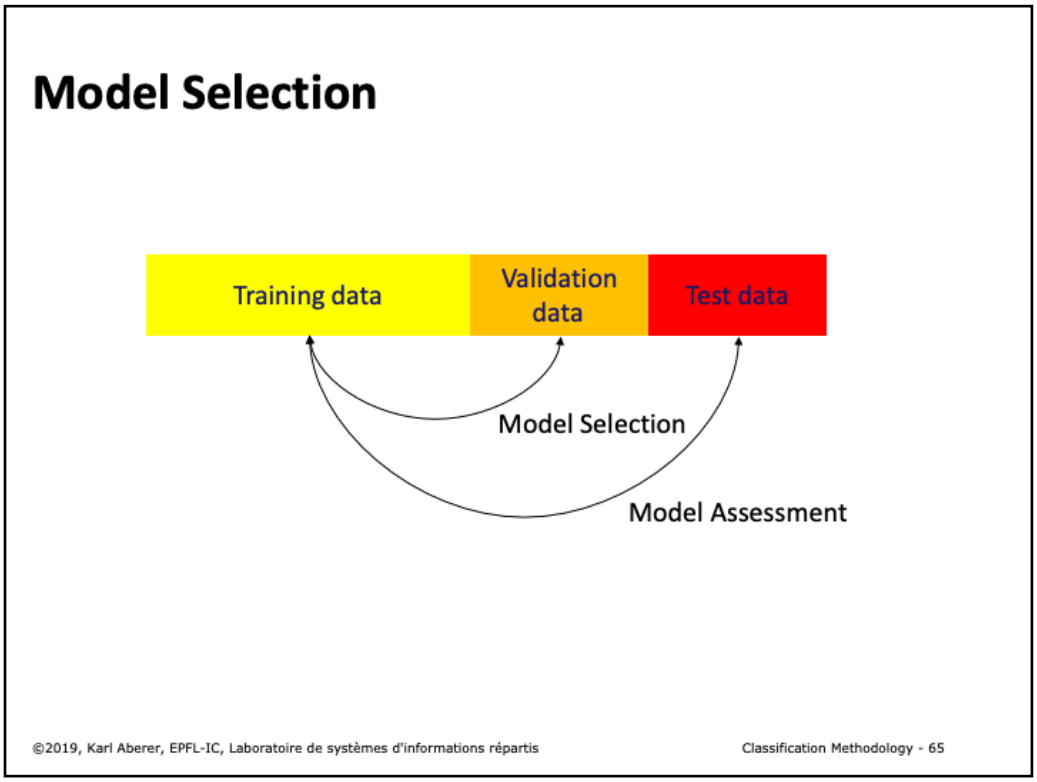
2. Model Selection

Usually a classifier has some parameters to be tuned

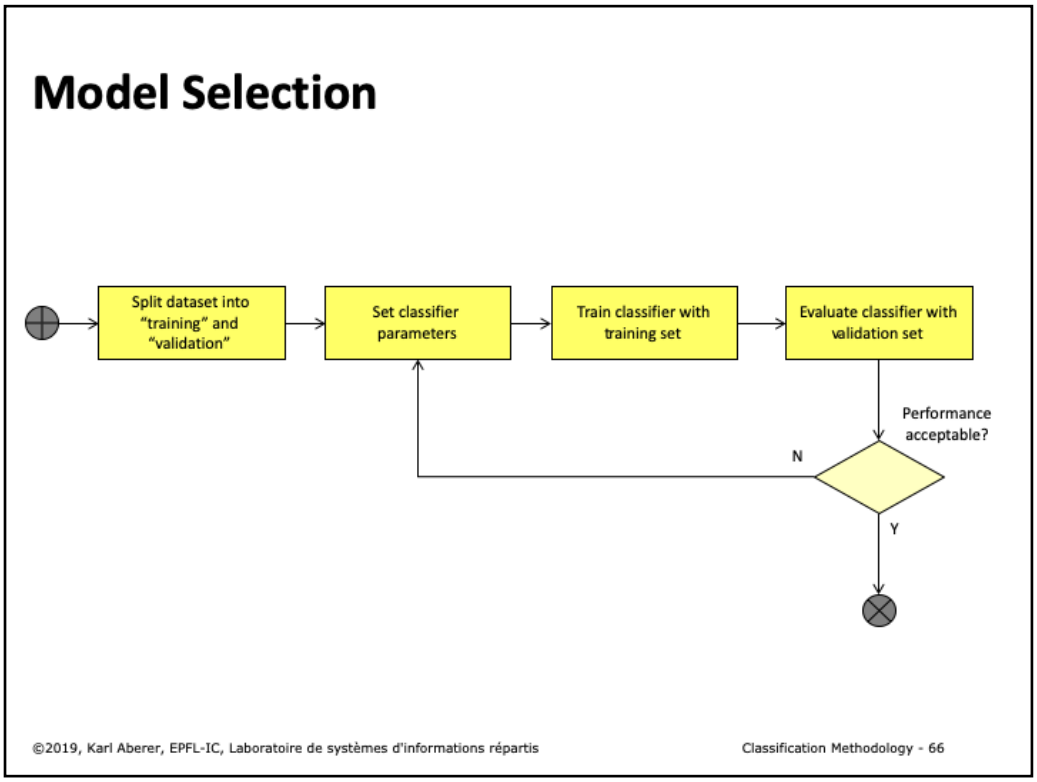
- Regularisation factor
- Threshold
- Distance function
- Number of neighbours

Model Selection: Estimating performances of different models to select the best one

Usually for a given type of classifier, we have one or several parameters that can be tuned. This means that we can produce many different classifiers for the same training set, and then select the one with the best performance (according to the performance metric that has been selected). This process is called model selection.

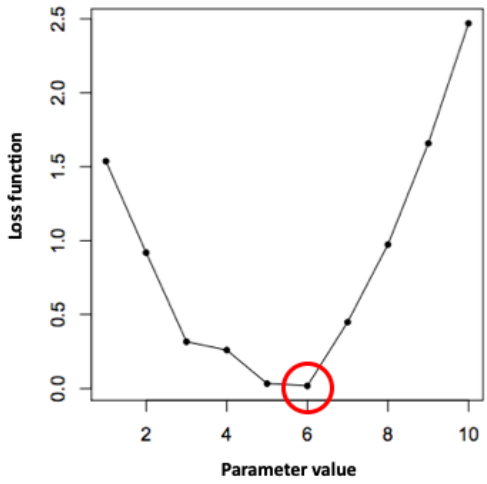


Model selection also needs evaluation of error, but in this case not to determine the performance of a classifier on unknown data, but to select the best model. If, on the other hand, we want to still evaluate the performance of the resulting final model, we need to distinguish two types of data that is used for evaluating model performance. The validation date that is used to select the best model, and the test data that is used to assess the performance of the finally resulting model.



Once the validation data is given, the process of model selection is straightforward: different parameters of the model are chosen, models are trained on the training set using those parameters, the resulting models are evaluated using the validation data, and the process stops when a model with appropriate parameters is found (or the parameter space is exhausted).

Model Selection



The typical situation in model selection is that for a changing parameter the loss function has a local minimum, which indicates the parameter with the best performance. In the context, of model one uses the term loss function instead of performance metric, in order to formulate the search for an optimal model as a minimization problem. However, the are equivalent. For example, if the performance metric would be the accuracy A, then the loss function J would be taken as J = 1 - A

Loss Function (Error Function)

Categorical output

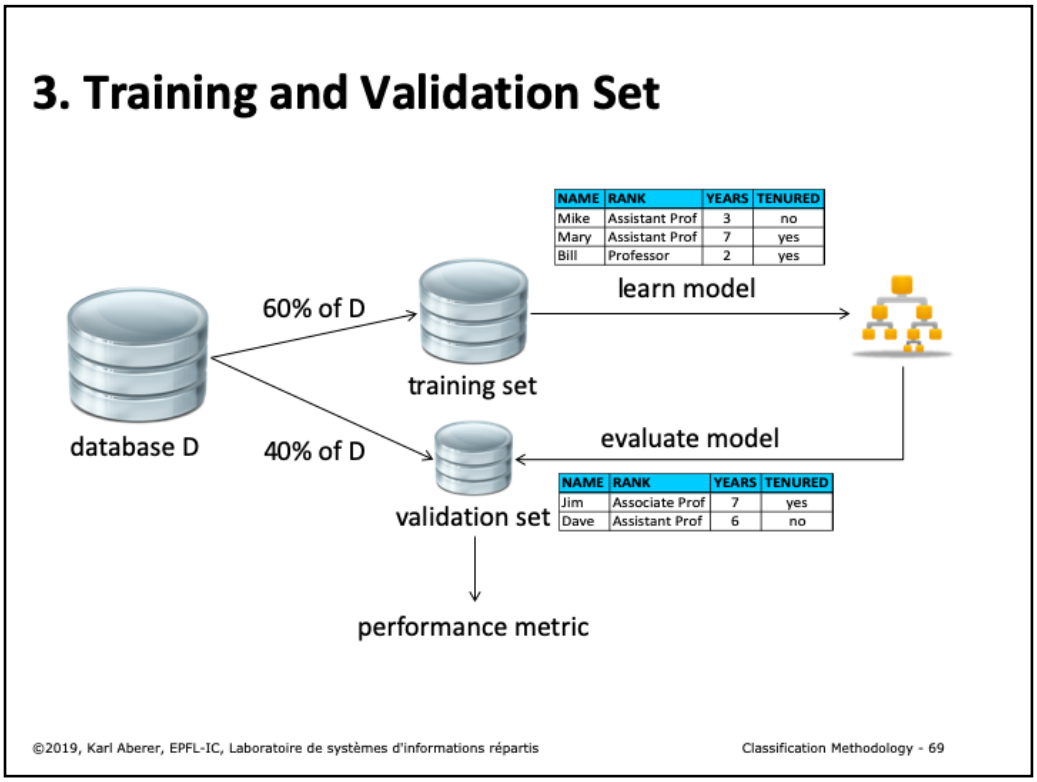
- 0-1 loss function: $J = \sum_{i=1}^{n} \#(y \neq f(x_i))$

Real value output

- Squared error: $J = \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2$
- Absolute error: $J = \frac{1}{n} \sum_{i=1}^{n} |y_i - f(x_i)|$

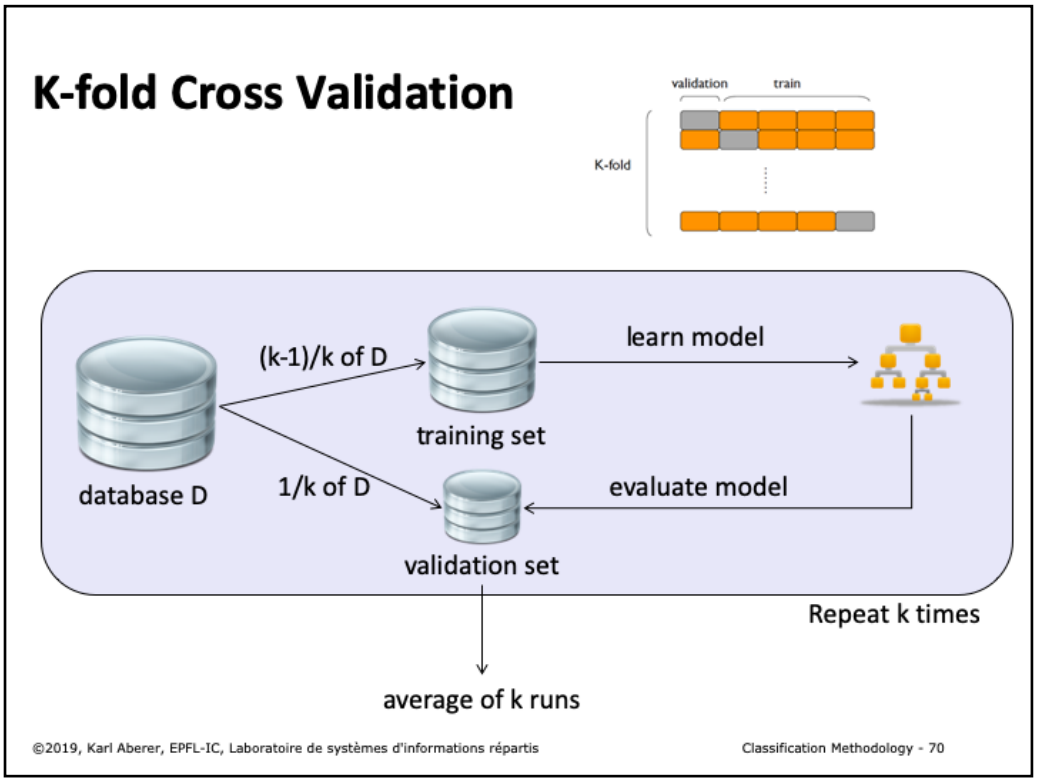
Here we give different examples of loss functions. The 0-1 loss function used for categorical functions to be learnt, essentially corresponds to accuracy (without normalization). For learning functions that return numerical outputs, the squared error or the absolute error can be used as loss function.

Here x_i indicates a feature and y is the class label.

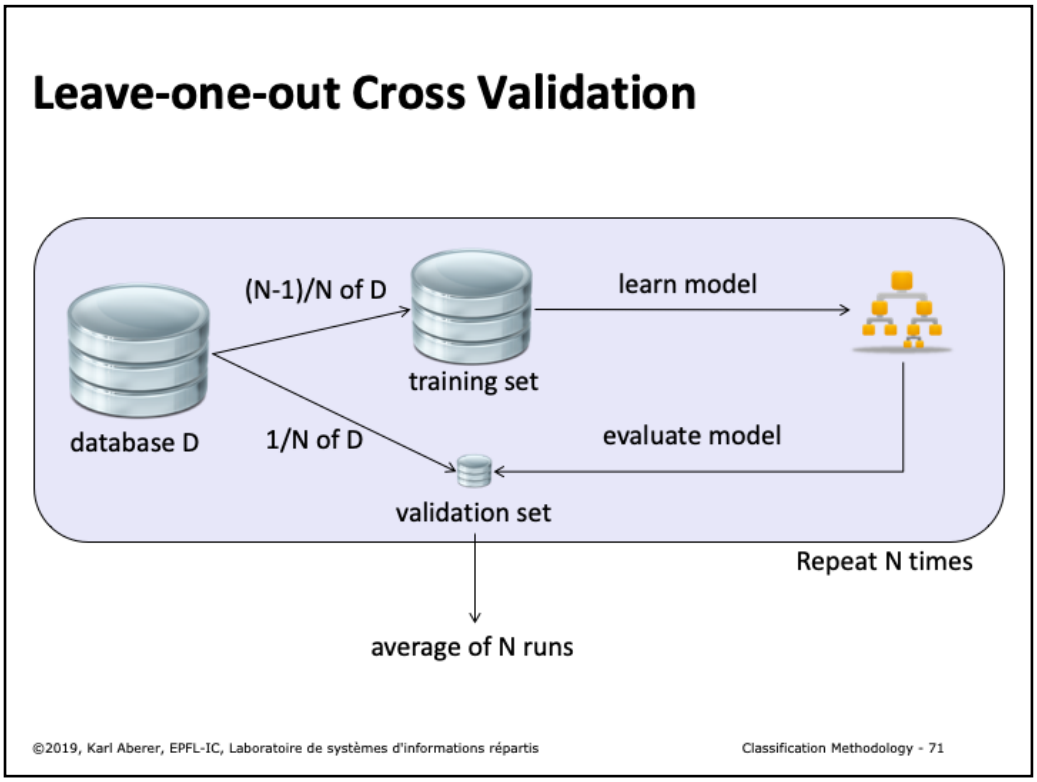


The simplest way to do model selection is constructing a training and validation set.

The drawback of this approach is that it does not use all the data to train, and it is possible that the validation set is not representative for the classification tasks (e.g., rare cases are missed).



In order to address the drawback of holding out a simple validation set, a better approach is to split the training set into k partitions, and train a model for each subset the dataset where one of the partitions has been removed as validation set. Then the model is evaluated against the validation set that has been held out. Finally the performance as averaged over all k runs. K-fold cross-validation is a good compromise between having an unbiased estimate of the true accuracy and reasonable computation time.



Leave-one-out Cross Validation is the extreme form of k-fold cross-validation, where each individual data item is considered as a separate partition. Leave-one-out is most unbiased estimate of the true accuracy, however is time consuming because it performs a number of iterations equal to the number of examples in the dataset.

Skewed Distributions

Some class labels might be heavily skewed, e.g.

- Non-Fake pages 10000
- Fake pages 10

Rare data points missed in validation set

As for performance evaluation, also for creating suitable validation sets, skewed data distributions pose problems. Rare data points might be simply missed in the validation set, and this the performance evaluation becomes questionable.

Fighting Skew

Stratification

- Select validation set as random sample, but assure that each class is (approximately) proportionally represented

Over- and Under-Sampling

- Including over-proportionally number from the smaller class (over-sampling)
- Including under-proportional number from larger class (under-sampling)

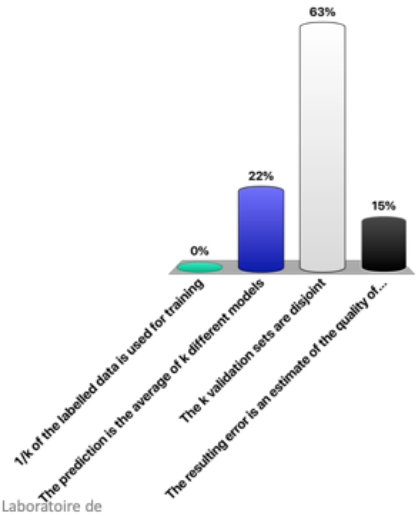
In order to avoid the above mentioned problems more sophisticated approaches for constructing the validation set are required.

Stratification chooses for each class a random sample of a size that is proportionally to the overall representation of the class. In this way it is avoided to miss classes completely in the validation set. This approach can also be taken when using k-fold cross-validation.

With over- and undersampling, small classes are emphasized whereas large classes are under-emphasized. Many variations of such sampling methods have been proposed, including some that generate artificial data points for underrepresented classes.

In k-fold cross-validation …

A. 1/k of the labelled data is used for training
B. The prediction is the average of k different models
C. The k validation sets are disjoint
D. The resulting error is an estimate of the quality of the classifier on real-world data



How Good is a Model?

Model is a function f_D that **estimates** a function

$f: X^d \rightarrow Y$ with $y = f(X)$

Evaluating the error

$$Err(f_D, T) = \frac{1}{|T|} \sum_{X \in T} (f_D(X) - y)^2$$

D = Training set from which the model is learnt

T = Validation set on which error is evaluated

Squared error measure

So far we have assumed that the choice of model parameters influences the quality of a model in model selection, but we have no clear understanding what are the factors that are driving the quality of a model. Having such an understanding is essential, in order to devise strategies to find good models. This is what we want to investigate in the following in greater detail.

To start with, a model is nothing else than a function f_D that approximates some function f. The function f_D has been learnt from a training set D and is evaluated using a validation set T. In the following we will use the squared error model. Under this model we can compute the error of the function when evaluated using the validation set.

Training and Test Error

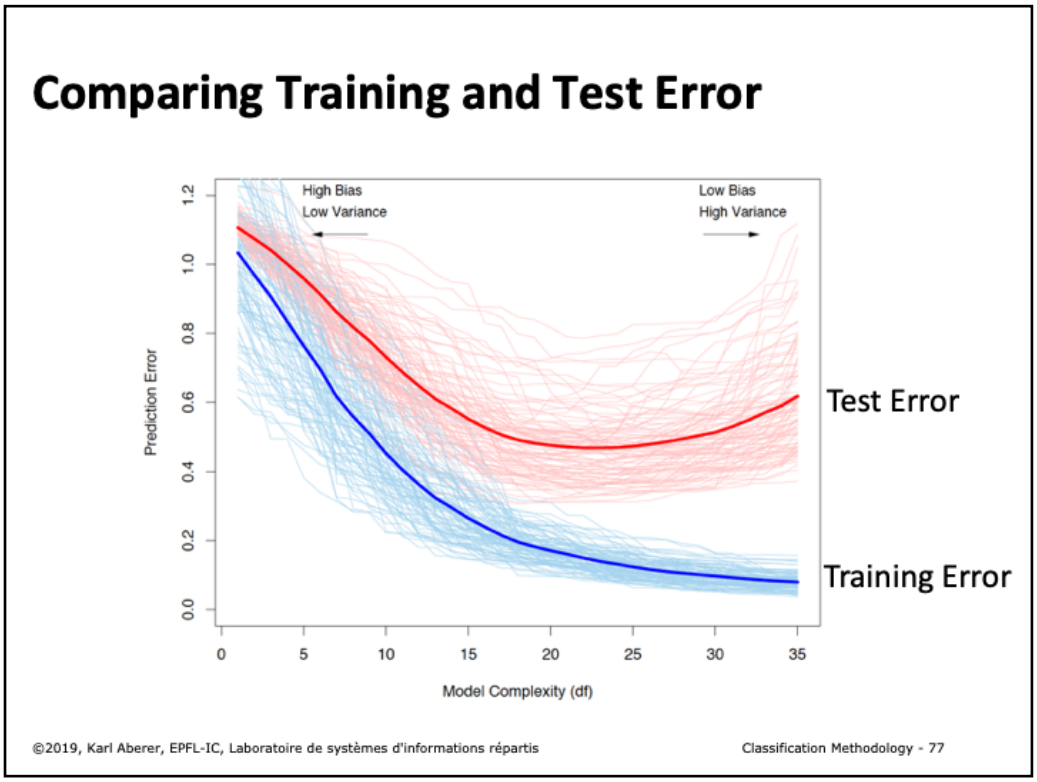
Evaluate error on training set D: **training error**

$$Err_{train} = Err(f_D, D)$$

Test model with an independent test set T: **test error**

$$Err_{test} = Err(f_D, T)$$

The error function can be evaluated on different data sets. We can evaluate on the training set itself, which results in the training error, and we can evaluated it on the test set, which results in the test error. An interesting question is whether it is useful to evaluate the training error, and how it behaves compared to the test error.



This figure illustrates the comparison between training an test error. On the x-axis we see increasing model complexity, on the y-axis increasing prediction error. Models of different complexity have been built multiple times, using different training sets and both the test error (red) and training error (blue) have been evaluated. The bold lines average over the different training sets for which the models have been built. If we look at the training error, we see that it decreases as the model becomes more complex. This corresponds to the fact that the model can capture more properties of the training data, as it has more parameters. However, this does not mean that the model becomes better. This is illustrated when looking at the test error. Initially, also the test error decreases as the model becomes more complex. We see that it has lower bias. But at a certain point the tendency changes and it increases again. This is the point where the model starts to overfit the training data, and this as soon as new data is tested against the model, it performs worse. For the test error, we also observe that the spread of the different lines increases as the model complexity increases. Thus the model is in some sense less stable, we say it has higher variance.

Expected Errors

Repeatedly evaluate error for different models generated from different training sets $D \in \mathcal{D}$ and corresponding test sets $T(D)$

Expected training error

$$EErr_{train} = E_{\mathcal{D}}[Err(f_D, D)] = \frac{1}{|\mathcal{D}|} \sum_{D \in \mathcal{D}} Err(f_D, D)$$

Expected test error

$$EErr_{test} = E_{\mathcal{D},T}\,[Err(f_D, T(D))]$$
$$= \frac{1}{|\mathcal{D}|} \sum_{D \in \mathcal{D}} Err(f_D, T(D))$$

We now will formally introduce the notions of bias and variance, by analyzing the expected errors that we obtain from generating multiple models from different training sets. We can define the expected training error, respectively the expected test error as the expectation of the error that is obtained from building models over multiple training sets. For computing the expected test error we also assume that for each training set D some test set T(D) is used (as e.g. in cross-validation). Note, that does not correspond exactly to the approach that is taken with cross-validation, where D and T(D) are always derived from the same dataset, but this formulation is easier to handle analytically.

Bias and Variance

The error can be rewritten as follows

$$EErr_{test} = Bias^2 + Variance$$

where

$$Bias = E_{\mathcal{D},T}[f_D(X) - y]$$

Deviation of predicted value from true value over all models

and

$$Variance = E_{\mathcal{D},T}[(f_D(X) - \bar{f}(X))^2]$$

Deviation from average predicted value by all models

with

$$\bar{f}(X) = E_{\mathcal{D}}[f_D(X)]$$

Average predicted value

Using some basic arithmetic we can rewrite the expression for the test error in two components, bias and variance. For computing bias we compute the expected error of the model with respect to the true value, averaging over all training sets D and data points X in the test set T(D). For a given data point X we can compute the expected estimate $\bar{f}(X)$ by averaging over all training sets D. Using this expected estimate we can compute the variance as the expected square of the difference between the expected estimate $\bar{f}(X)$ and the model $f_D(X)$, again averaging over all training sets D and data points X in the test set T(D). In this way we have isolated the two different contributions to the test error, called Bias and Variance.

Bias / Variance and Model Complexity

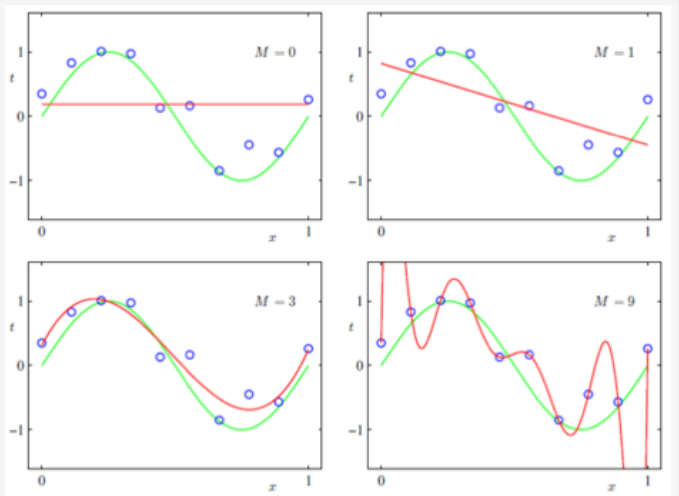
There is usually a bias-variance tradeoff caused by model complexity

Complex models (many parameters) usually have lower bias, but higher variance
→ **over-fitting**

Simple models (few parameters) have higher bias, but lower variance
→ **under-fitting**

Empirical and analytical studies have revealed that there is bias-variance tradeoff that is related to the model complexity. Ideally we would like to have a low bias and low variance. This is however not possible as with decreasing bias of the test error in general variance increases. Generally, complex models lower the bias (up to a certain point) but increase variance, resulting in overfitting, and for simple models the inverse is true.

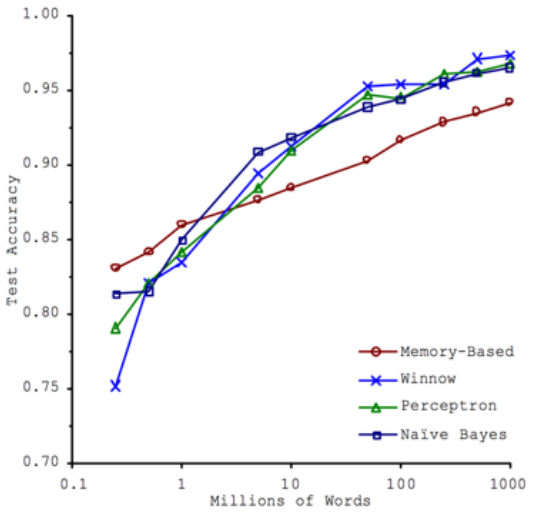
Example



Fitting a data set to different orders of polynomials (Bishop, "Pattern Recognition and Machine Learning")

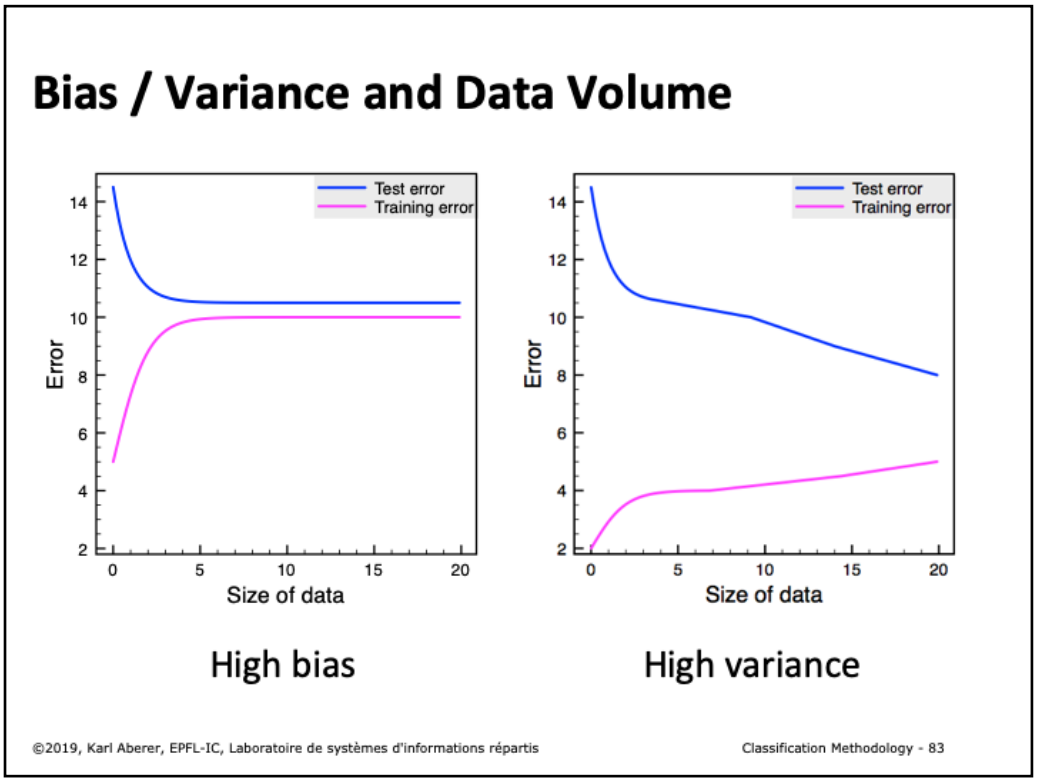
This figure illustrates the point under and over-fitting. We are fitting a model (red line) to data that has been sampled from a function (yellow line). As long as the model is simple (constant or linear function) bias will be high, as the model cannot well fit the data, but variance will be low, as different models will all be very similar. In an intermediate range (the cubic curve) the model will well approximate the true function. As model complexity further increases (polynomial of degree 9), overfitting occurs which leaders to high variance (as different models for different training data will be very different) and the function is not well approximated as well, which will increase bias. In other words with over-fitting the shape of the curve is determined more by the specific data sample that is used for training, than by the underlying function.

Does More Data Help?

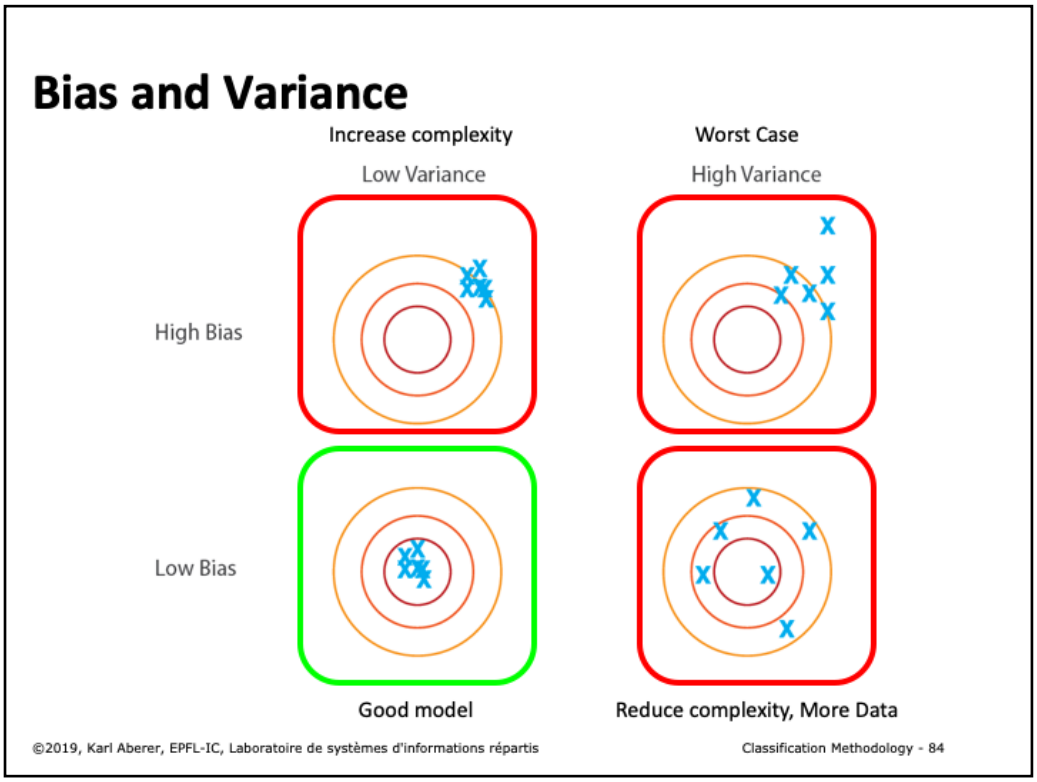


Data > Algorithms

With the recent growth in available data, a controversy has occurred around the question, whether the choice of the learning algorithm is still important, and whether not making available sufficient training data will solve any learning problem. This figure is taken from a paper that is at the origin of this controversy, at it seems to indicate that independently of which algorithm is chosen, the accuracy increases as more data becomes available. Note that one of the 4 learning methods, the memory-based learner is a very simple method that has only limited model complexity. Thus it benefits the least from having more data.



When looking in more detail on the question whether more training data helps the question is not so clear-cut. Other experiments have shown, that more data helps, if the variance is high, i.e. over-fitting occurs. Then more data can indeed help. Intuitively, this is the case because the models are more complex and have thus more capacity to absorb information from the training data.



Putting all these insights together we can come to the following conclusions for the 4 possible scenarios of bias and variance:

- When bias and variance is low, nothing is to be done, we have a good model.
- When bias and variance are both high, nothing is to be done, we have probably a problem that is too difficult to learn
- When bias is low, but variance is high, we have overfitting: we can either reduce the complexity of the model, or add more data.
- When bias is high, but variance is low, we have underfitting: we can increase the complexity of the model

Which is wrong?

A. The lower model complexity, the higher bias
B. The higher model complexity, the higher variance
C. The higher the data volume, the higher the training error
D. The training error is always higher than the test error

References

Textbook

- Friedman, Jerome, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Vol. 1. Springer, Berlin: Springer series in statistics, 2001

Papers

- Banko, Michele, and Eric Brill. "Scaling to very very large corpora for natural language disambiguation." *Proceedings of the 39th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2001
- Chawla, Nitesh V. "Data mining for imbalanced datasets: An overview." *Data mining and knowledge discovery handbook*. Springer US, 2005. 853-867.
- http://m.shookrun.com/documents/stupidmining.pdf
- Olteanu, A., Peshterliev, S., Liu, X., & Aberer, K. "Web credibility: Features exploration and credibility prediction." *European Conference on Information Retrieval*. Springer Berlin Heidelberg, 2013.