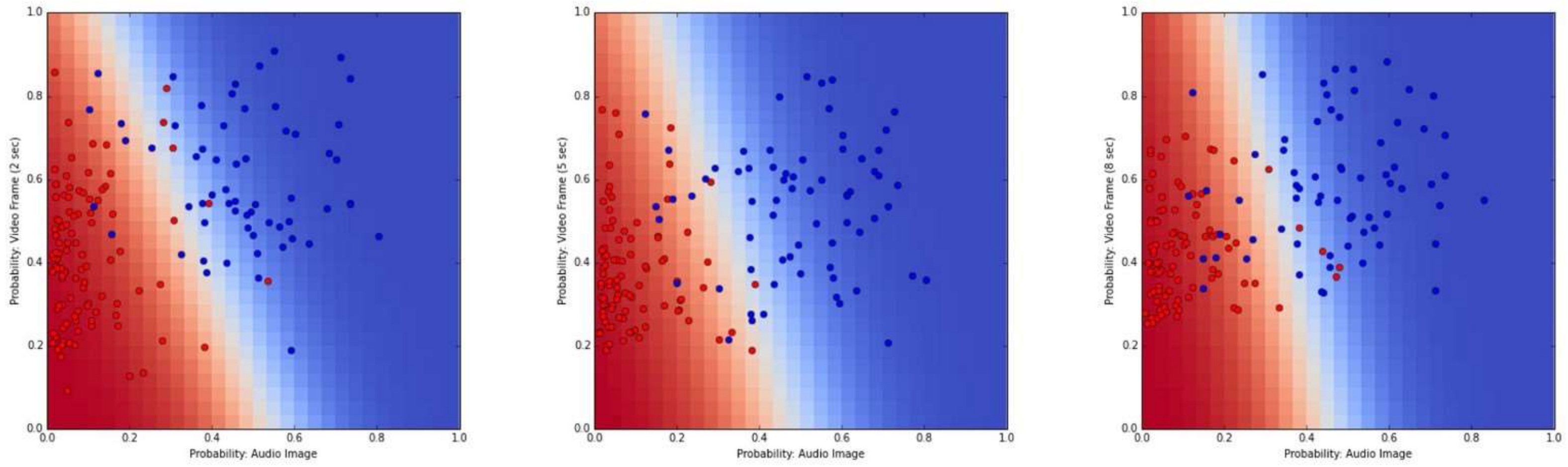
Chapter 5

Evaluation, selection and combination of classifiers



Recommended bibliography: *Pattern Classification (2nd ed) by R. O. Duda, P. E. Hart and D. G. Stork, John Wiley & Sons, 2000*, Chapter 9

Information Theory, Inference, and Learning Algorithms by David J. C. MacKay, Cambridge University Press 2004

Credits: Some figures are taken from *Pattern Classification (2nd ed) by R. O. Duda, P. E. Hart and D. G. Stork, John Wiley & Sons, 2000* with the permission of the authors

INDEX

5.1 Lack of superiority of any classifier

5.2 Complexity

5.3 Bias and variance

5.4 Resampling for classifier design

5.5 Combining classifiers

5.6 Comparing classifiers

5.7 Imbalanced data sets

5.8 Lauching an ML Project

5.9 Ethical risks of ML

1. LACK OF SUPERIORITY OF ANY CLASSIFIER

Mitchell (1980), Schaffer (1994) and Wolpert (1996) showed that no pattern classification algorithm is inherently superior to any other, or even to random guessing, in the lack of knowledge about the problem (e.g. prior distributions, shape of discriminants, etc.).

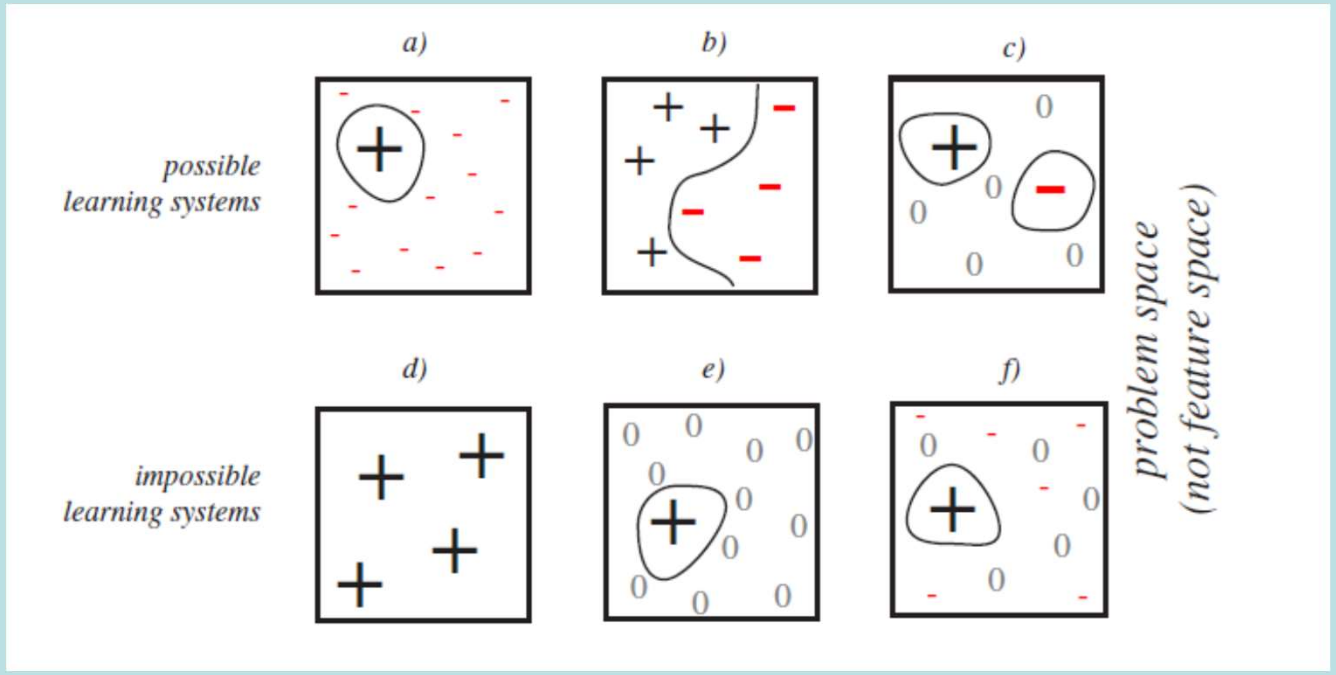
David Wolpert, "The Lack of A Priori Distinctions between Learning Algorithms", Neural Computation, 1996, pp. 1341-1390

In the absence of assumptions there is no privileged feature representation. This is stated in the **Ugly Duckling Theorem**.

If one learning algorithm seems to perform better is just because it is fit to the problem, but we cannot claim this in general. This is stated in the **No Free Lunch Theorem: there is no universally best learning algorithm**

Implications:

- No classification algorithm works well for the full set of the functions F to be learnt.
- No matter what algorithm we use, there is at least one F for which random guessing is better.
- This justifies trying many different classification techniques: **Machine Learning is an empirical discipline.**


a)
b)
c)
d)
e)
f)
possible learning systems
impossible learning systems
problem space (not feature space)

+ : The classification algorithm has generalization higher than average

– : The classification algorithm has generalization lower than average

0 : The classification algorithm has average performance

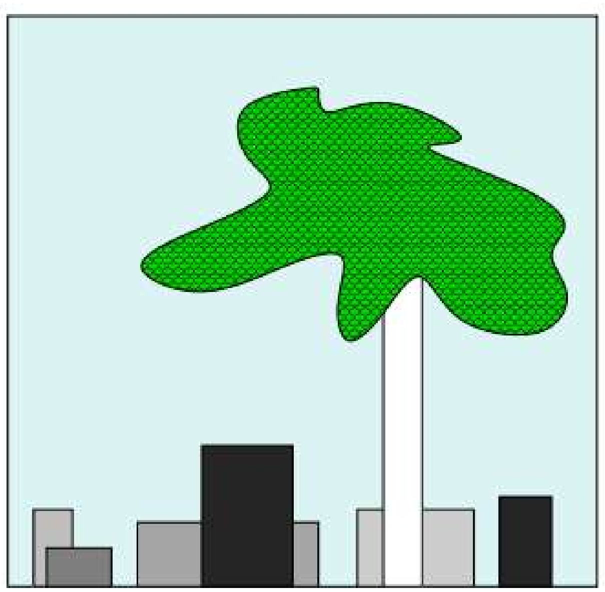
The size of symbol indicates the amount by which performance differs from average

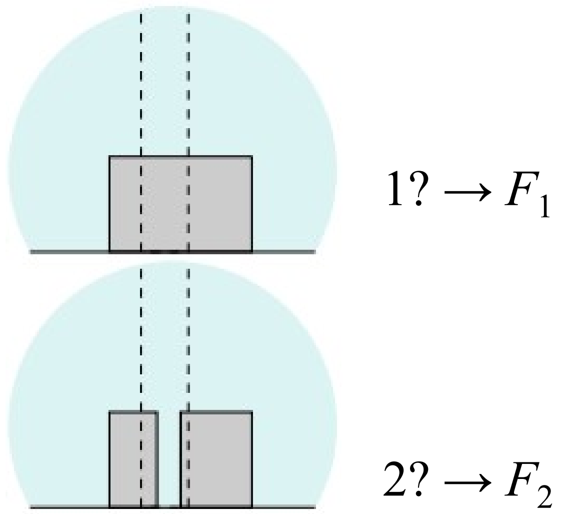
2. COMPLEXITY

In practice it has been observed that low complexity classifiers perform well.

Occam's razor principle: if several explanations are compatible with the observations, adopt the simplest («entia non sunt multiplicanda praeter necessitatem»)

How many boxes are there behind the tree?



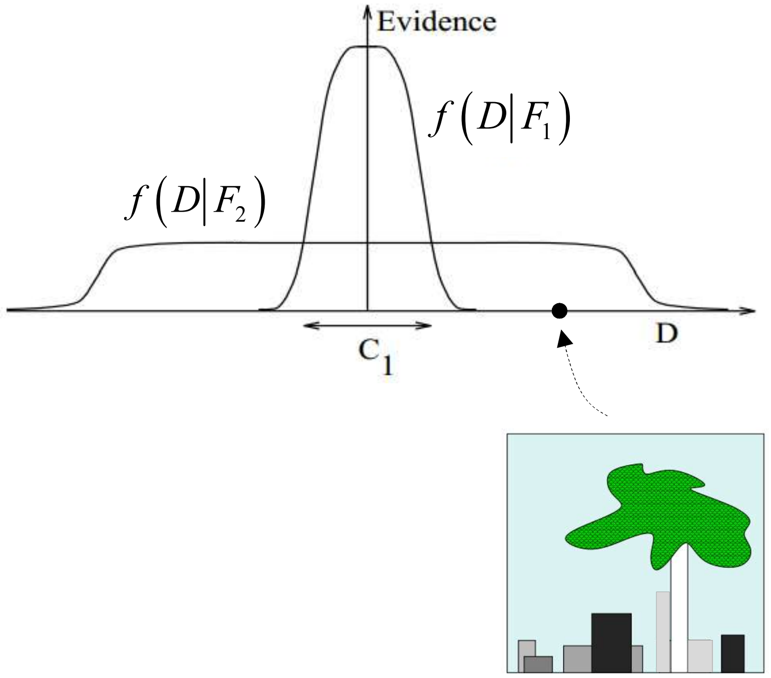


In fact, Bayesian inference automatically embodies **Occam's razor principle**.

$$\frac{\Pr(F_1|D)}{\Pr(F_2|D)} = \frac{\Pr(F_1)}{\Pr(F_2)} \frac{f(D|F_1)}{f(D|F_2)}$$

our prior belief

how well data is predicted by F_1 as compared to F_2



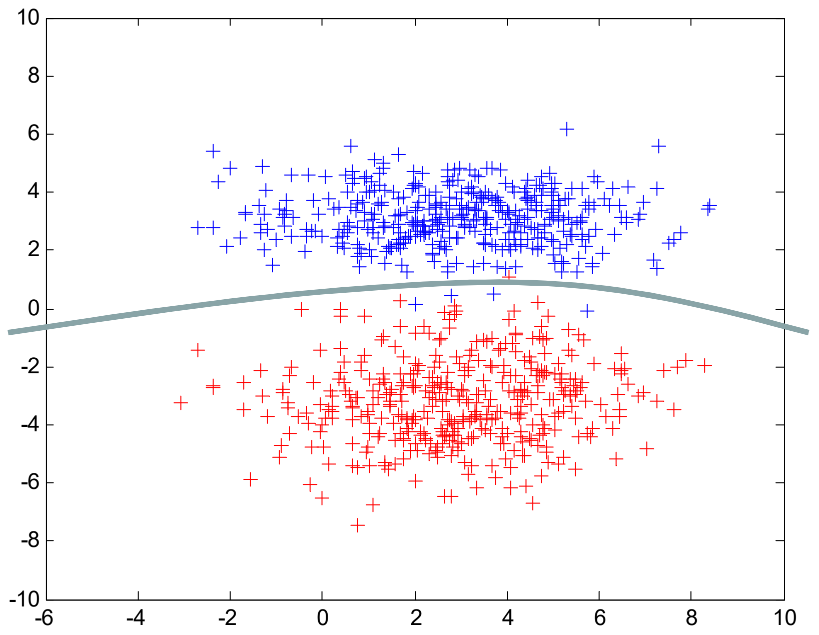
If F_2 is more complex, it spreads its prediction of reality over a wider range of values of observations D: it explains a greater variety of data sets. If our observation falls in the interval C_1 then Bayes rule states that F_1 (simple model) has to be chosen.

In the machine learning context: for a given data set D, which is the most likely hypothesis on the complexity of the function to be learnt $F(\mathbf{x})$? Remember that h should match the unknown F.

$c = 2$
$d = 2$

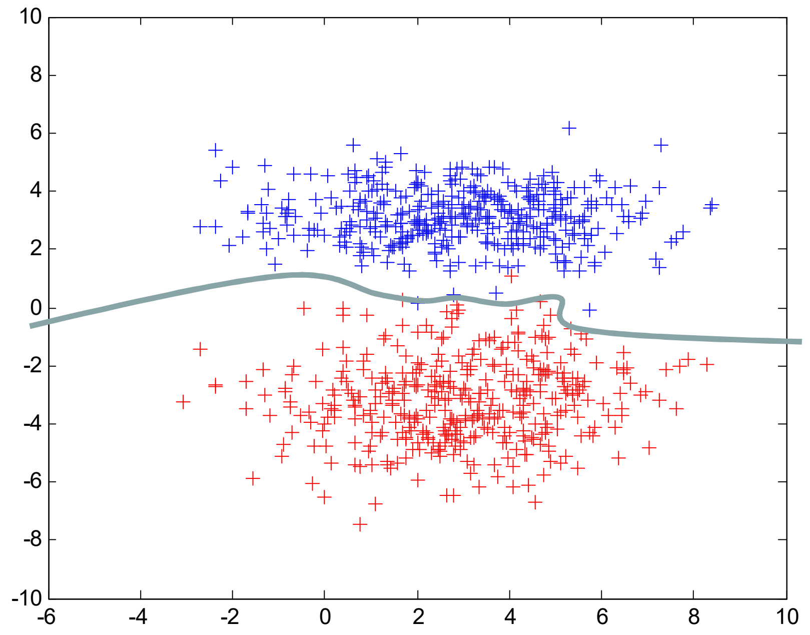
$h(\mathbf{x}) = 0$

F_1: simple model



NN with a few neurons

F_2: complex model



NN with many neurons

It can be proved that classifiers designed to minimize the Kolmogorov-Touring complexity:

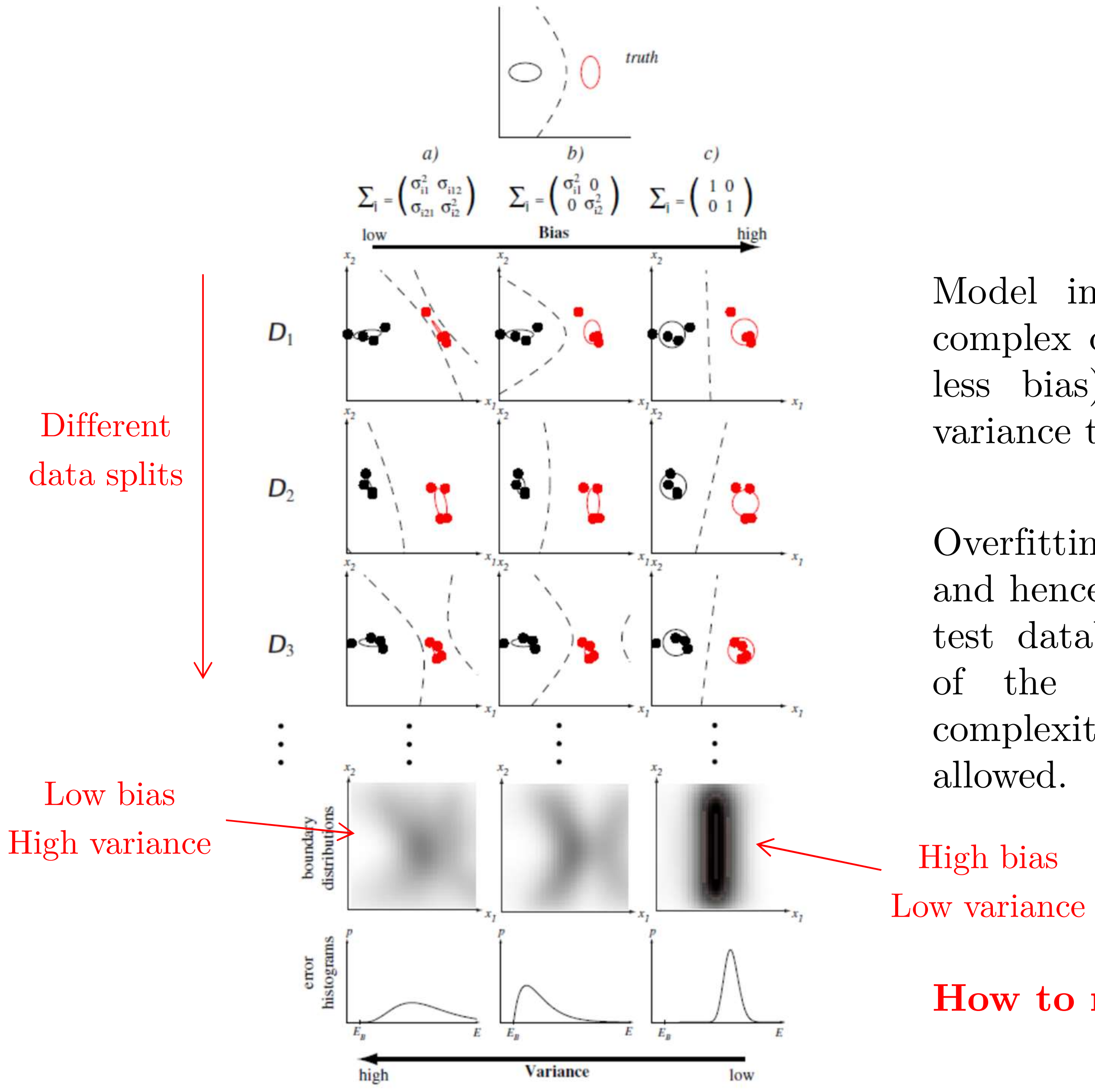
$$K\left(h,D\right)=K\left(h\right)+K\left(D\text{ using }h\right)$$

converge to the ideal model when more and more data is available (not for finite data, which would violate the No Free Lunch Theorem).

Example: Tree classifier

- h specifies the tree and the decisions at each node, complexity is proportional to the number of nodes, i.e. $K(h) = \alpha \cdot h$
- Complexity of D using h can be expressed as the weighted sum of entropies of data at the leaf nodes, i.e. $K(D\ using\ h) = \sum_{leaf\ \text{nodes}} i(N)$
- Pruning the tree based on entropy criteria is minimizing $K(h, D)$

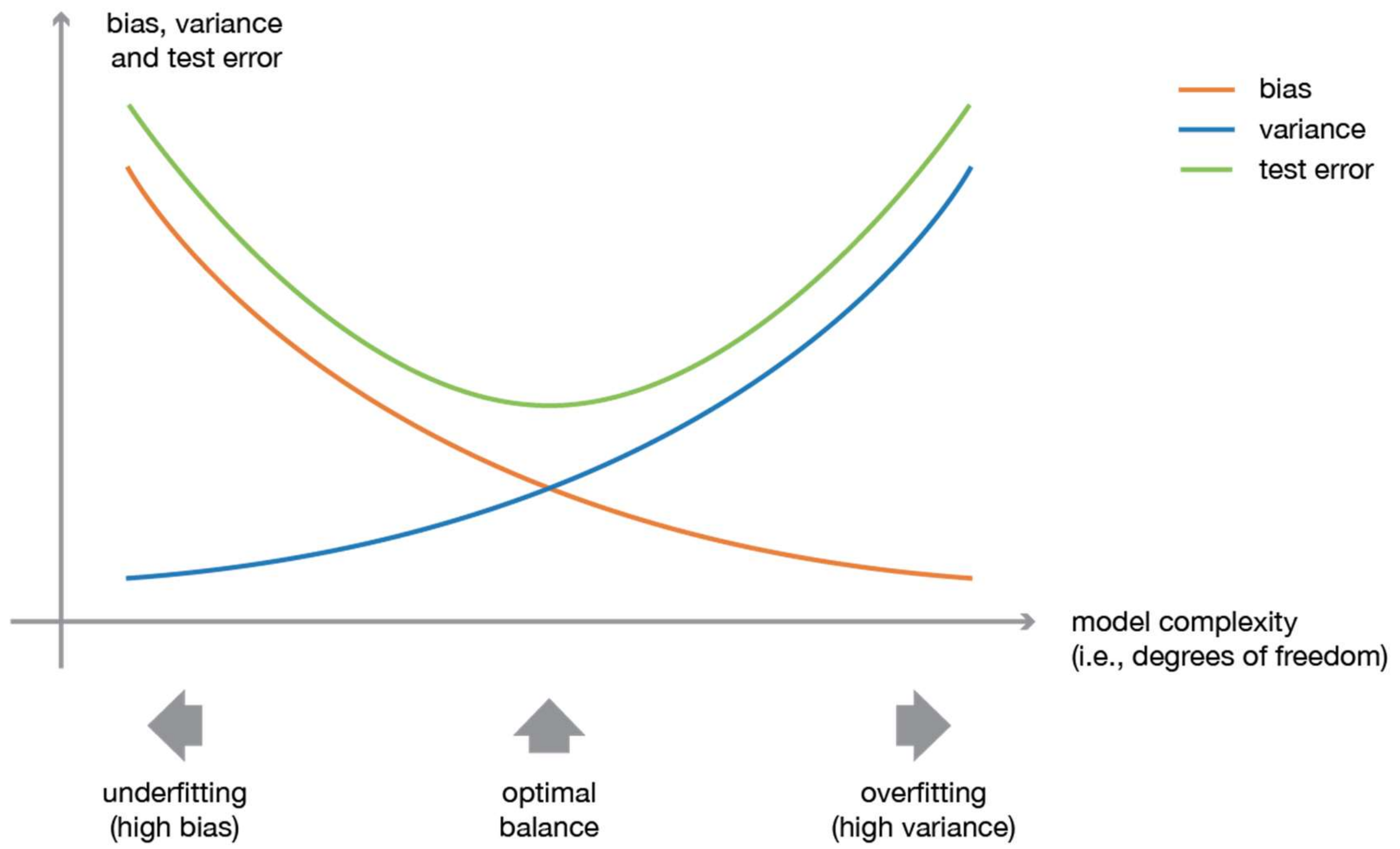
3. BIAS AND VARIANCE


truth
a)
b)
c)
$\Sigma_i = \begin{pmatrix} \sigma_{i1}^2 & \sigma_{i12} \\ \sigma_{i21} & \sigma_{i2}^2 \end{pmatrix}$
$\Sigma_i = \begin{pmatrix} \sigma_{i1}^2 & 0 \\ 0 & \sigma_{i2}^2 \end{pmatrix}$
$\Sigma_i = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
low
Bias
high
Different data splits
D_1
D_2
D_3
x_1
x_2
boundary distributions
Low bias
High variance
High bias
Low variance
error histograms
p
E_B
E
high
Variance
low

Model in column a) allows a more complex decision boundary (and hence less bias) at the expenses of more variance than models b) or c)

Overfitting implies higher variability, and hence poorer performance with the test database. Generalization capacity of the classifier is better if less complexity (and hence some bias) is allowed.

How to reduce variability?


bias, variance
and test error
bias
variance
test error
model complexity
(i.e., degrees of freedom)
underfitting
(high bias)
optimal
balance
overfitting
(high variance)

4. RESAMPLING FOR CLASSIFIER DESIGN

Small changes in the training set may result into big differences in performance (like in tree classifiers). Three ways to reduce bias/variance: **bagging, boosting** and **random forests.**

Bagging: reduce variance by generating diversity

- Generate multiple versions of the training set with $n' < n$ (with replacement[1])
- Train component weak classifiers[2] of the same type, independently between them: bagging mainly reduces variance.
- Generate a decision by voting, that is, we assign the most frequent label among all classifiers outputs.

1. A "small" fraction of samples can be repeated
2. Weak classifiers behave just better than random guessing. They are used here because they require little computational effort in classification.

Random forest: reduce variance by generating diversity

In addition to bagging, one may also draw a random subset of features for training the individual trees (in bagging, each tree uses the full set of features).

Due to the **random feature selection**, the trees are more independent of each other compared to regular bagging (where all component classifiers tend to focus on the most significant features).

The use of RF often results in better variance reduction and often better predictive performance (at the expense of a little more bias).

It is also faster to train, because each tree learns only from a subset of features.

Boosting**: reduce bias and variance by focusing on hard-to-classify patterns**.

Let us construct succesive classifiers from selected samples of the database that take into consideration the errors incurred by previous classifiers (therefore, training is not independent). Improvement is obtained in terms of both bias and variance.

Popular approaches:

- AdaBoost
- Gradient boosting

Let us illustrate the principle by assuming two classes:

- Generate D_1 using n_1 patters from D
- Train classifier C_1
- Generate D_2 using n_2 patterns as the "most informative" set given the output C_1: half of the patterns are correctly classified by C_1, and half erroneously, from the $n - n_1$ remaining
- Train classifier C_2
- Generate D_3 containing the patterns from the remaining $n - n_1 - n_2$ for which C_1 and C_2 disagree. Other samples are disregarded
- Classify by voting

It is preferred to *have data bases of approximately the same size*. Be careful: if the problem is simple n_2 will be small; if it is difficult, it will be unacceptably large. This may require running boosting several times with different values of n_1 .

We can do better by designing M classifiers: select vector $\mathbf{x}_i$ in the k-th data base according to a factor that depends on the classification results of vector $\mathbf{x}_i$ on the $k-1$ previous classifiers. Each classifer targets the error of the previous one...

the AdaBoost procedure

Low (large) values imply good (bad) classification so far, and low (large) chances for sample i of being used

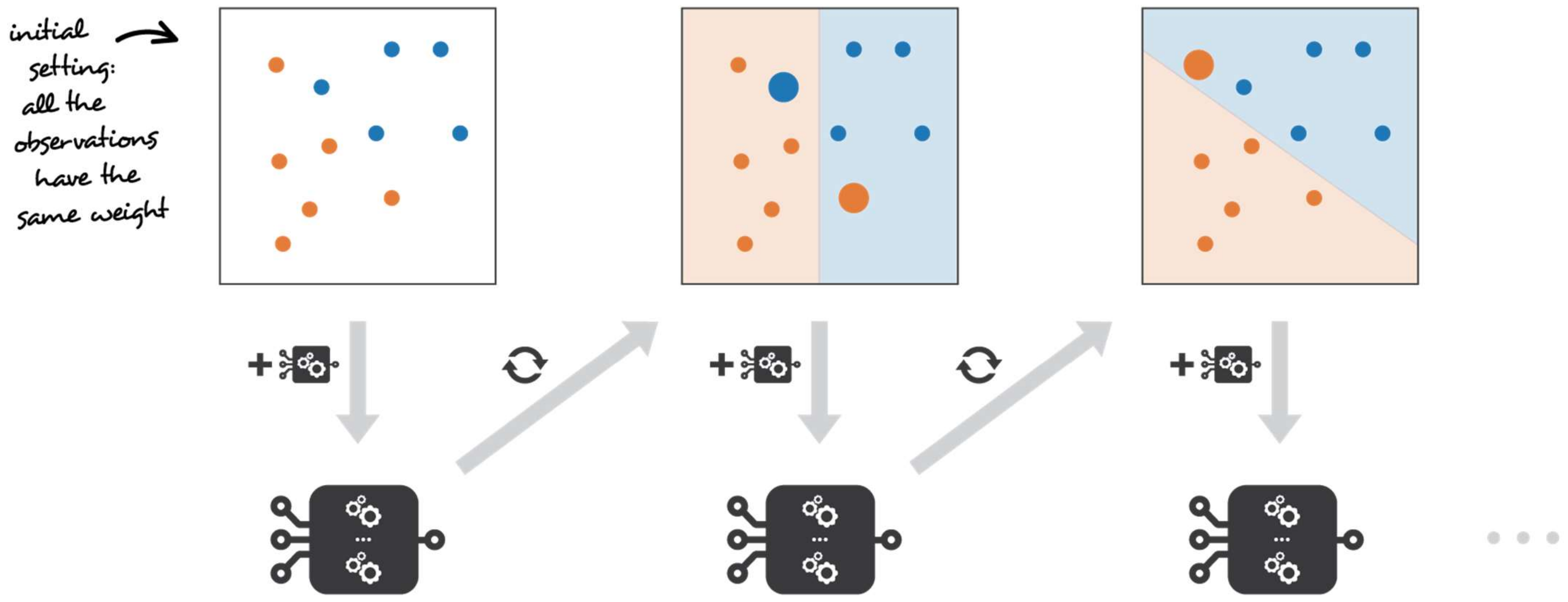
```
begin initialize $\mathcal{D} = \{x_1, y_1, x_2, y_2, \ldots, x_n, y_n\}, W_1(i) = 1/n, M$
      $k \leftarrow 0$
      do $k \leftarrow k+1$
         Train weak learner $C_k$ using $\mathcal{D}$ sampled according to distribution $W_k(i)$
         $E_k \leftarrow$ Training error of $C_k$ measured on $\mathcal{D}$ using $W_k(i)$
         $\alpha_k \leftarrow \frac{1}{2}\ln[(1-E_k)/E_k]$
         $W_{k+1}(i) \leftarrow \frac{W_k(i)}{Z_k} \times \begin{cases} e^{-\alpha_k} & \text{if } h_k(x_i) = y_i \text{ (correctly classified)} \\ e^{\alpha_k} & \text{if } h_k(x_i) \neq y_i \text{ (incorrectly classified)} \end{cases}$
      until $k = M$
      return $C_k$ and $\alpha_k$ for $k = 1$ to $M$   (ensemble of classifiers with weights)
   end
```

Normalization over all samples for component classifier k so that $W_k(i)$ represents a distribution train a weak model and aggregate it to the ensemble model

update the weights of observations misclassified by the current ensemble model

▨ current ensemble model predicts "orange" class

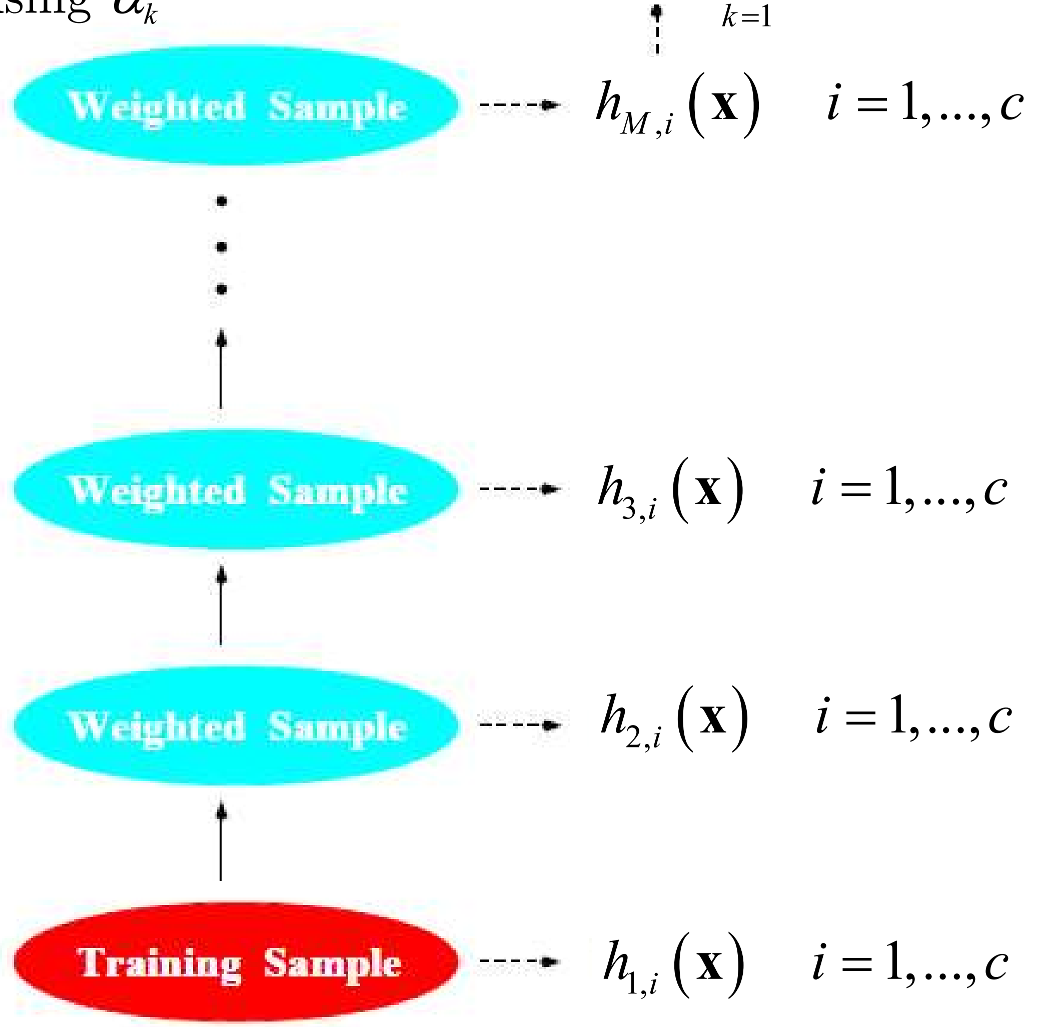
▨ current ensemble model predicts "blue" class



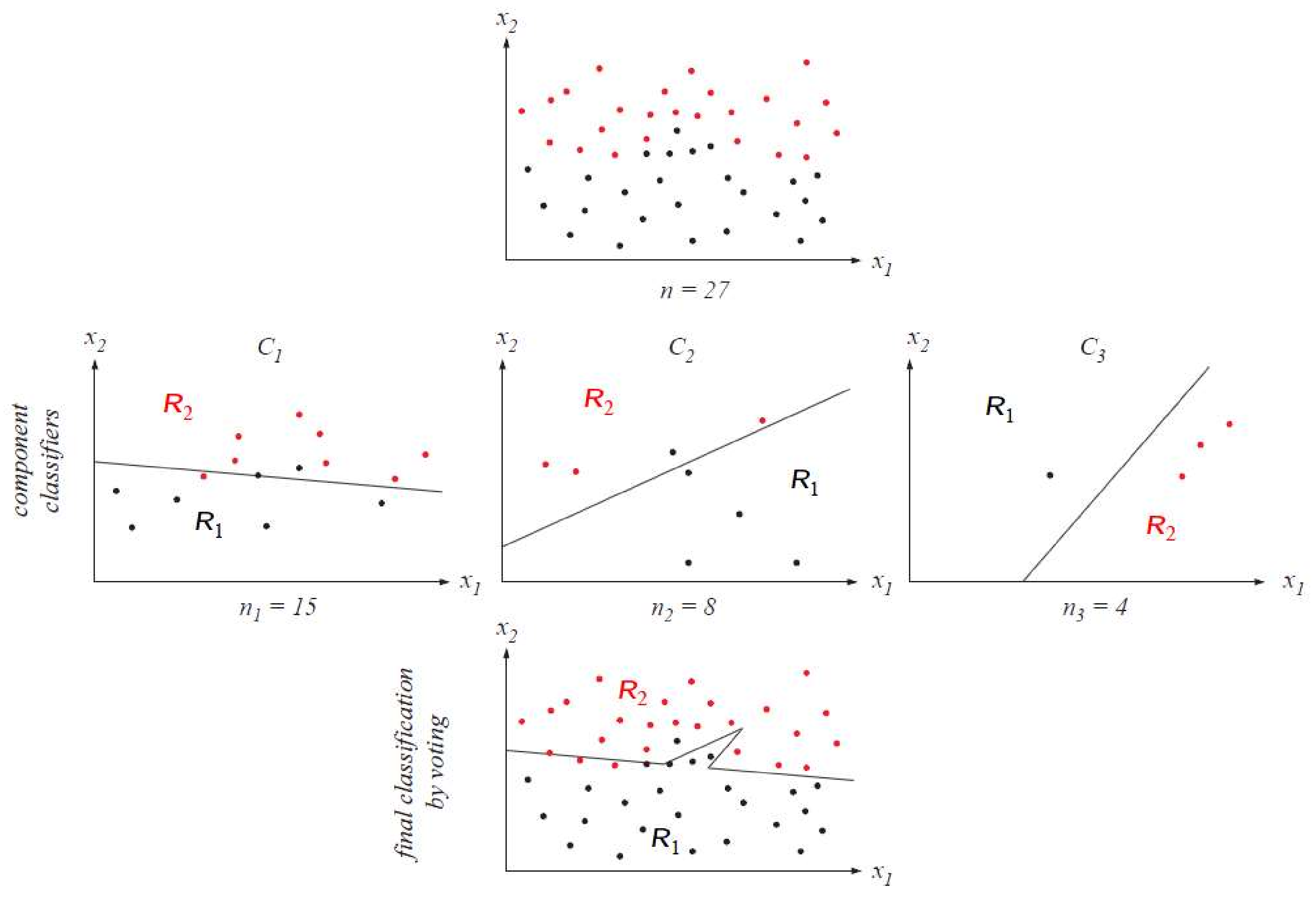
Once trained, classify by deciding on the maximum discriminant, built according to classification errors using α_k

Final classifier

$$g_i(\mathbf{x}) = \sum_{k=1}^{M} \alpha_k h_{k,i}(\mathbf{x}) \quad i = 1,\dots,c$$



Weighted Sample
$h_{M,i}(\mathbf{x}) \quad i=1,\dots,c$
Weighted Sample
$h_{3,i}(\mathbf{x}) \quad i=1,\dots,c$
Weighted Sample
$h_{2,i}(\mathbf{x}) \quad i=1,\dots,c$
Training Sample
$h_{1,i}(\mathbf{x}) \quad i=1,\dots,c$


x_2
x_1
$n = 27$
component classifiers
C_1
R_2
R_1
$n_1 = 15$
C_2
R_2
R_1
$n_2 = 8$
C_3
R_1
R_2
$n_3 = 4$
final classification by voting
R_2
R_1

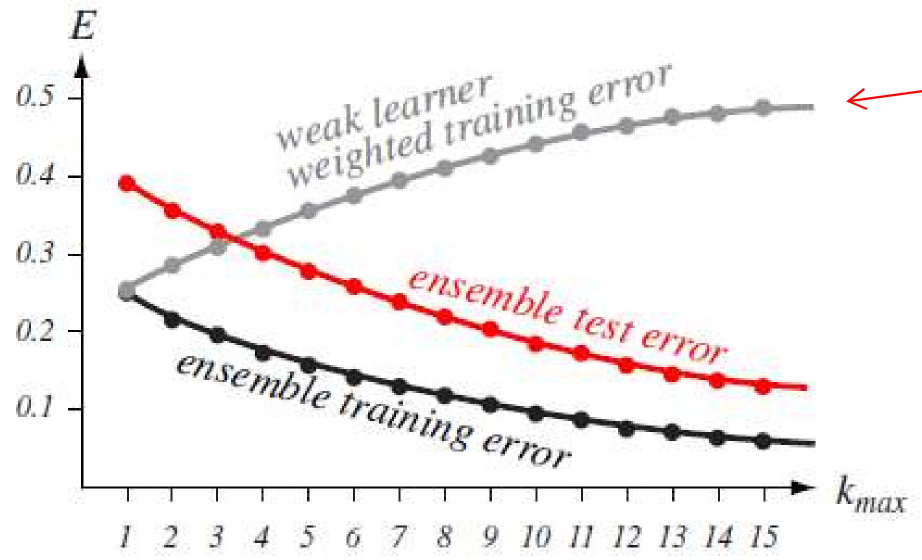
Training error performance of AdaBoost (from Tibsirani's book): can be made arbitrary low by setting the number of component classifiers M large enough...

The training error of a component classifier C_k can be expressed as (in the two classes case):

$$E_k = 1/2 - G_k \quad \text{for some} \quad G_k > 0$$

The ensamble training error can be found as

$$E = \prod_{k=1}^{M} 2\sqrt{E_k(1-E_k)} = \prod_{k=1}^{M} \sqrt{1-4G_k^2} \le \exp\left(-2\sum_{k=1}^{M} G_k^2\right)$$



Each succesive classifier is performing worse than the previous due to the selection of the training data

Are these methods violating the NFL Theorem?

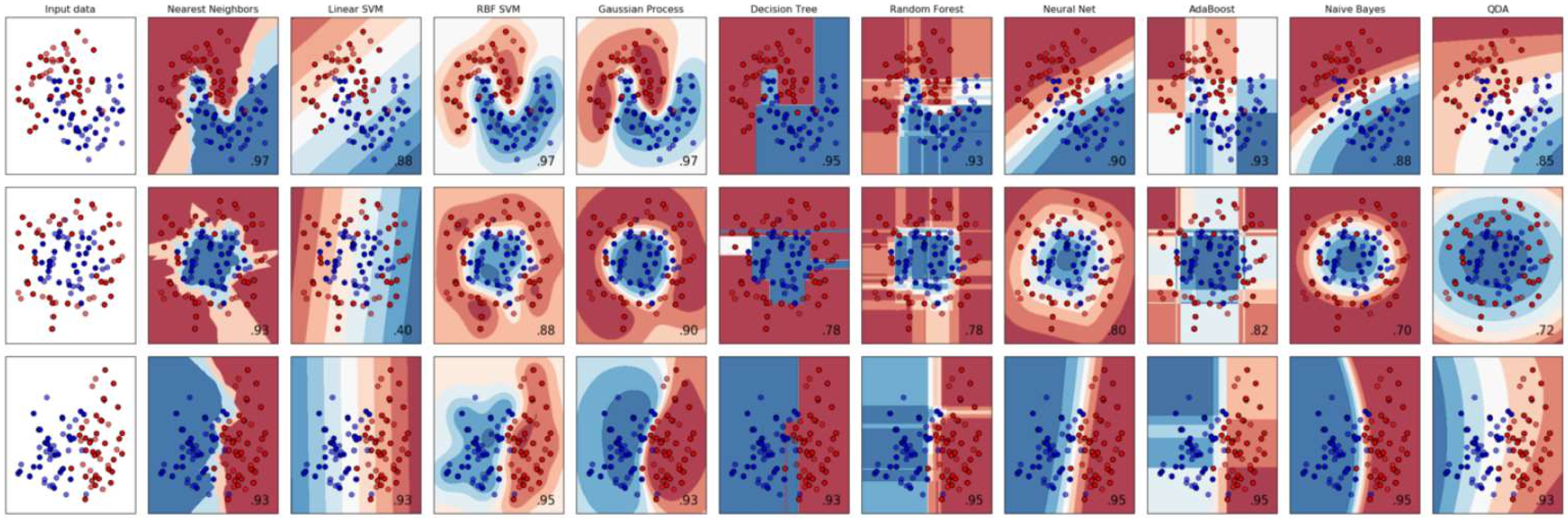
No: boosting improves classification only if component classifiers perform better than random guessing, which cannot be guaranteed in advance. Some knowledge about $F(\mathbf{x})$ (acquired through training) is needed to assert that a classifier is better than random guessing.

Why resampling should do better than training on independent data?

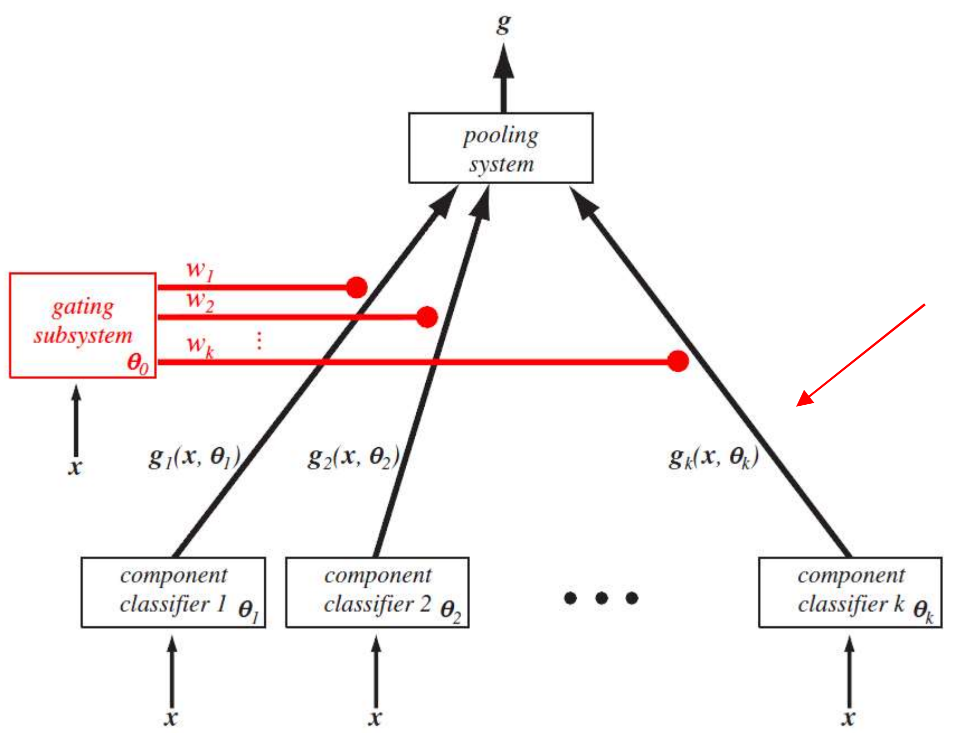
1. We are broadening the class of implementable functions
2. Indirectly, bias and variance are adjusted (better generalization)
3. There is no atempt to fit distributions: all techniques are non-parametric

5. COMBINING CLASSIFIERS

Since every component classifier is more suited to different regions of the space of functions $F(\mathbf{x})$, it might be useful to combine decisions:



Also called stacking:



g
pooling system
gating subsystem θ_0
w_1
w_2
w_k
x
$g_1(x, \theta_1)$
$g_2(x, \theta_2)$
$g_k(x, \theta_k)$
component classifier 1 θ_1
component classifier 2 θ_2
component classifier k θ_k
x
x
x

Each component classifier generates a set of c scalar discriminant values

On each component classifier, the sum of discriminants must be normalised (so some scaling is needed)...

$$\sum_{j=1}^{c} g_{kj} = 1$$

... and the discriminants can be combined by:

- Weighting with the inverse of the probability of error of each component classifier

- Weighting with the inverse of the variance of the discriminants

- Adopting the discriminants as new features and train an outer classifier (called the meta-model) using the predictions of the component classifiers on a second training database (we can use k-folding to avoid having a reduction in the number of data).

If the outputs of the classifiers are of different nature

- **Analog**, do some kind of normalization
- **Rank order**, assume linearity proportional to the ordering
- **One-of-c**, set all to zero except for the category selected

Example: assume $c = 6$ classes

we could also take the fraction of vectors associated to each class among the k-nearest

we could also take the fraction of vectors associated to each class at the end leaf

Neural network

$\widehat{g}_i$	g_i
0.4	0.16
0.6	0.24
0.9	0.36
0.3	0.12
0.2	0.08
0.1	0.04

K-nearest

$\widehat{g}_i$	g_i
3rd	4/21
6th	1/21
5th	2/21
1st	6/21
2nd	5/21
4th	3/21

Tree

$\widehat{g}_i$	g_i
0	0.0
1	1.0
0	0.0
0	0.0
0	0.0
0	0.0

6. COMPARING CLASSIFIERS

Reasons to assess generalization performance (in terms of error probability) of a classifier:

- To determine if it is good enough to be useful
- To compare the performance with a competitor design

Several heuristic methods...

1. K-folding cross-validation: split the data base in subsets and average the errors obtained on each subset. It can be used to:

- Determine the value of k in k-nearest neighbours
- Set the width of the Parzen window
- Stop training or set the number of hidden nodes in a NN
- Prunning trees

The fraction of samples for validation should be low (some 10%).

The error on the validation set gives an estimate of the accuracy of the classifier on the test set. If the (unknown) error rate of the classifier is p, and k out of N independent test samples are misclassified, then:

$$\Pr(k) = \binom{N}{k} p^k (1-p)^{N-k} \qquad \hat{p}_{ML} = k/N$$

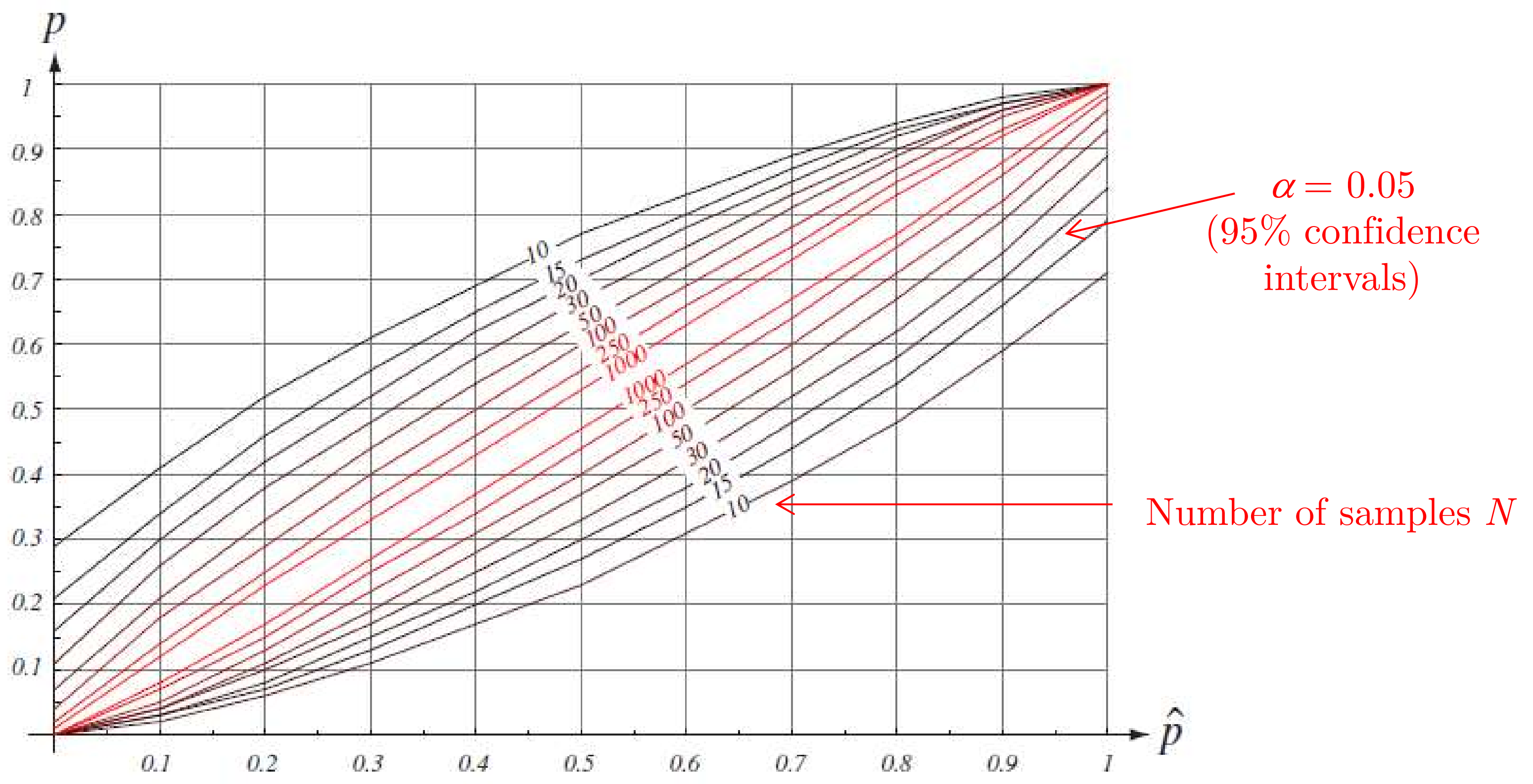
Care should be taken with confidence intervals, specially for low number of samples N: given a certain value of $\hat{p}_{ML}$ what is the confidence interval with a certain probability α?

For a given true value of p, the estimated value $\hat{p}_{ML}$ is within a certain interval $\left[p-\gamma^l, p+\gamma^u\right]$ with probability α:

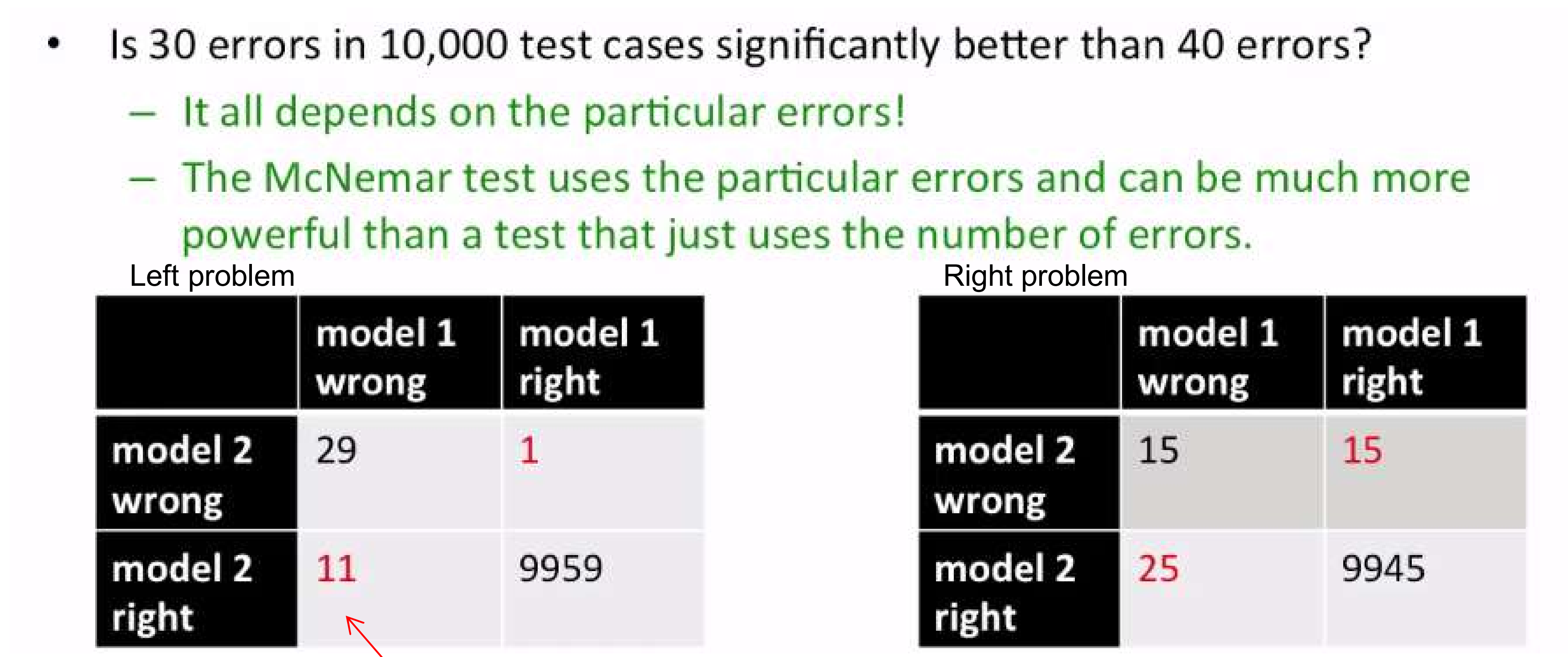
$$\frac{\alpha}{2} \ge \Pr\left\{k/N \le p-\gamma^l\right\} = \sum_{k=0}^{\lfloor N(p-\gamma^l)\rfloor} \binom{N}{k} p^k (1-p)^{N-k}$$

$$\frac{\alpha}{2} \ge \Pr\left\{k/N \ge p+\gamma^u\right\} = 1 - \sum_{k=0}^{\lfloor N(p+\gamma^u)\rfloor} \binom{N}{k} p^k (1-p)^{N-k}$$

Given a value of k/N, we can check on the graph for the confidence interval:



How to detect a significant drop in the error rate?



- Is 30 errors in 10,000 test cases significantly better than 40 errors?
  - It all depends on the particular errors!
  - The McNemar test uses the particular errors and can be much more powerful than a test that just uses the number of errors.

Left problem

	model 1 wrong	model 1 right
model 2 wrong	29	1
model 2 right	11	9959

Right problem

	model 1 wrong	model 1 right
model 2 wrong	15	15
model 2 right	25	9945

Number of vectors where model 2 is right and model 1 is wrong

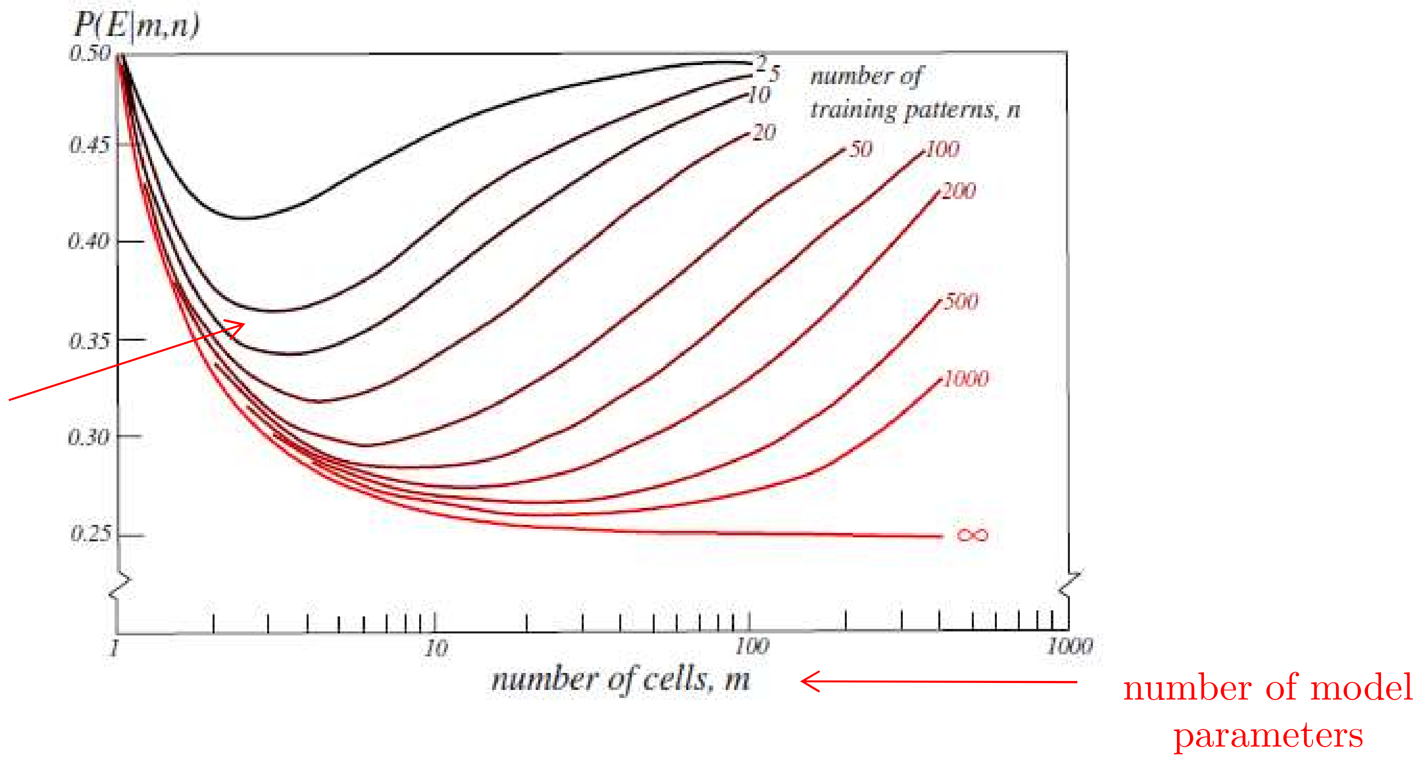
Let us test two models (1 & 2) in two problems (left & right). In both problems model 2 is providing lower error than model 1 (30 vs. 40), but in the right problem model 2 is significantly better, fixing 62,5% errors of model 1 vs only 27,5% in the left problem.

https://en.wikipedia.org/wiki/McNemar%27s_test

2. Problem average error rate, is used to evaluate the performance as a function of the number of samples...

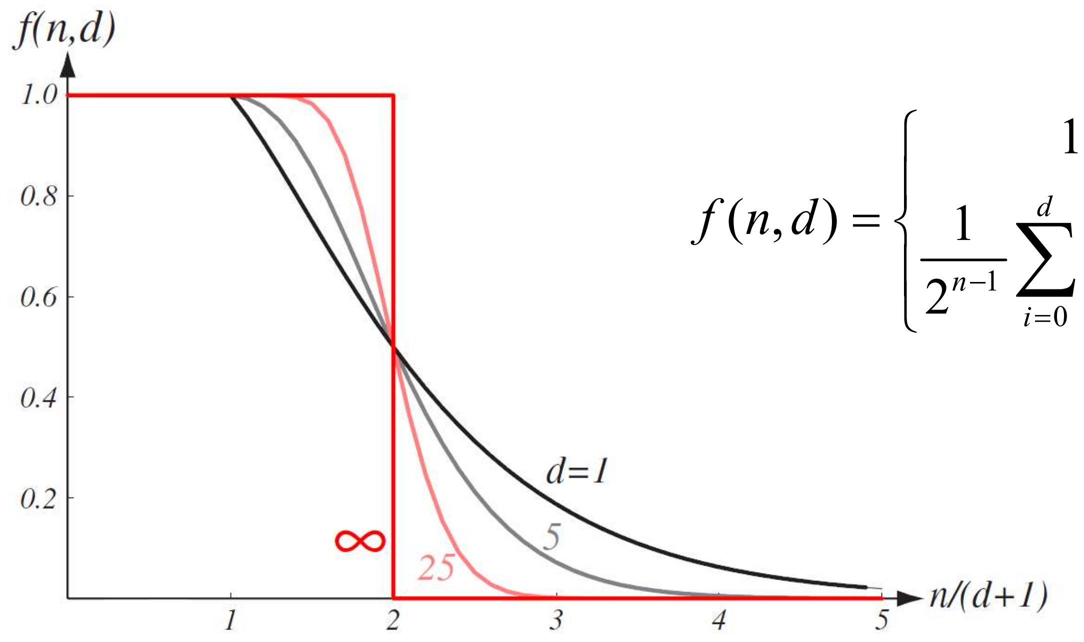
- If the number of samples is low, the classifier does not generalise well and perform badly on new data.

- For a fixed number of samples, increasing the number of parameters could be counterproductive (too many parameters to train and too few samples)



3. The capacity of a separating plane. Under which conditions we can ensure that a plane can separate classes?

Given n points in a d-dimensional space, and two classes, 2^n dichotomies can be built, but only a fraction are linearly separable:



$$f(n,d) = \begin{cases} 1 & \text{for } n \le d+1 \\ \dfrac{1}{2^{n-1}} \displaystyle\sum_{i=0}^{d} \binom{n-1}{i} & \text{for } n > d+1 \end{cases}$$

- We cannot expect linear separability if $n \gg 2(d+1)$
- Linear separation always possible if $n \approx d+1$ (rationale behind kernel-based SVM!)

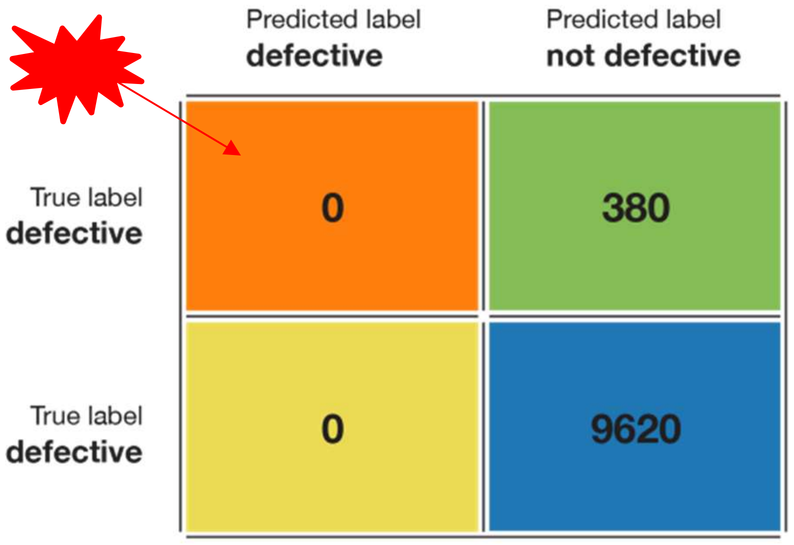
7. IMBALANCED DATA SETS

You are hired to create a model that, based on a database, predicts whether a product is defective or not. You decide to use a classifier, train it on the data and get a 96.2% **accuracy**!

Your boss decides to use the model and some time later he underlines the uselessness of the model: it has not found any defective product from the time it has been used in production.

After some investigations, you find out that around 3.8% of the product made by your company is defective and your model just **always answers** "not defective", leading to a 96.2% accuracy.

This is a usual situation in ML applications (e.g. detection of fraudulent transactions, diagnose of rare diseases, prediction of natural disasters).



	Predicted label **defective**	Predicted label **not defective**
True label **defective**	0	380
True label **not defective**	0	9620

In this case, Precision $= 1 - 380/9620 = 96{,}2\%$

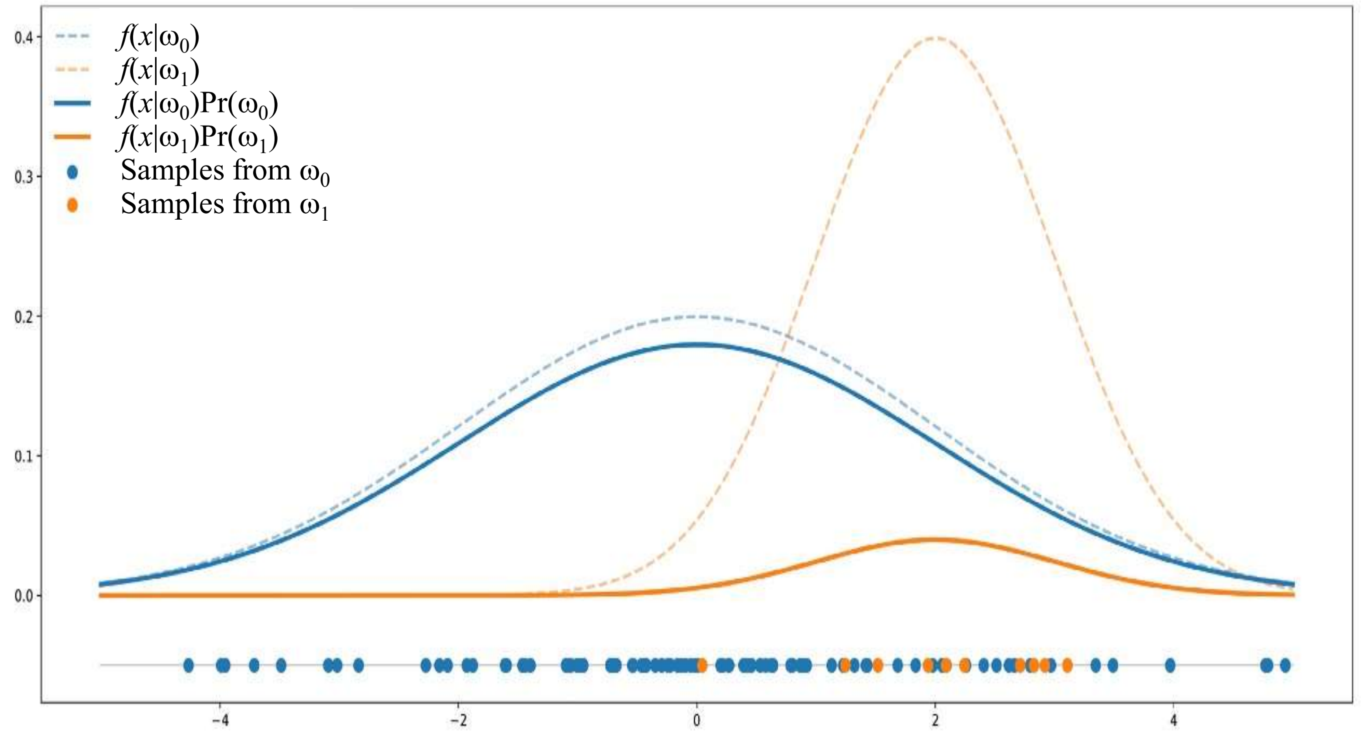
Specificity $= 9620/9620 = 100\%$

Recall $= 0/380 \times 100 = 0\%$

Accuracy: $\Pr(\omega_0) \cdot S + \Pr(\omega_1) \cdot R = 96{,}2\ \%$

The kind of "naive" results obtained is due to the imbalance of the dataset. **Accuracy is the wrong performance measure.**

Assume two classes, one has a very low a priori probability. If using MAP, the best solution is always deciding class ω_0:



Solution 1. Rather than using MAP, resort to minimizing the **Bayesian risk**, or use **Neyman-Pearson** criterion, which allocate larger importance to under-represented classes. The classifier used has to return the a-posteriori probabilities.

Solution 2. Weight classes at training. For instance, when using neural networks, minimize the criterion:

$$\underset{\mathbf{w}}{\text{minimize}} - \sum_{k=1}^{c} \mu_k t_k \log z_k(\mathbf{x})$$

where μ_k is a cost associated to a wrong decision in class k.

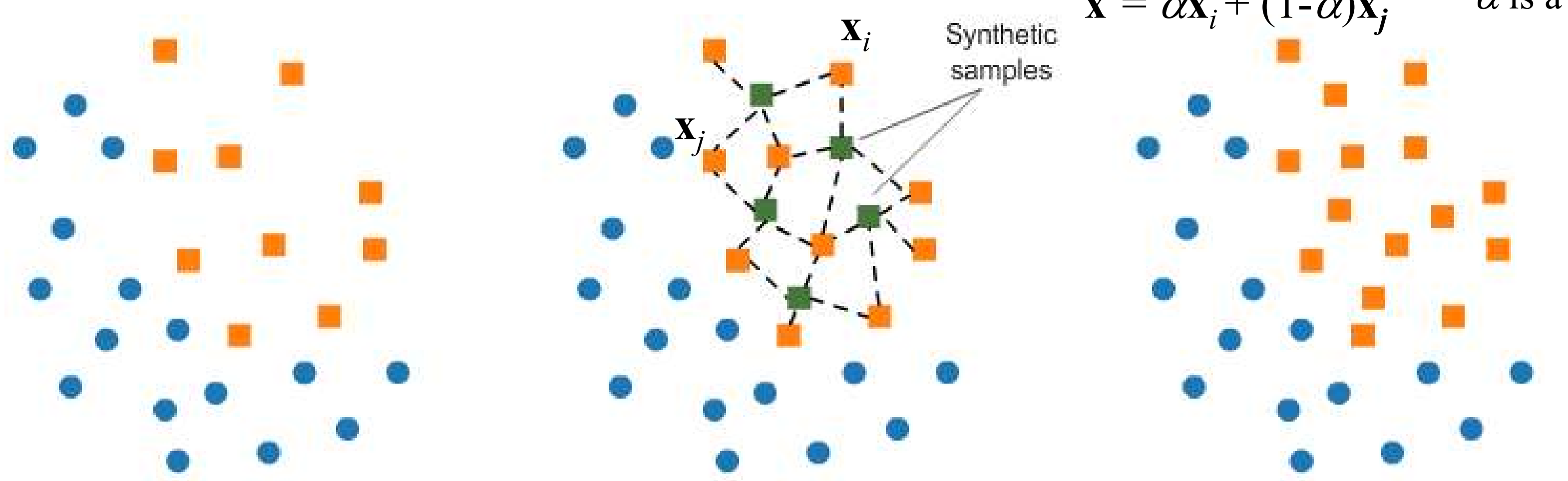
For a two class case, if $\mu_0 > 1$, class 0 is weighted higher, meaning the network is less likely to ignore it (lesser false negatives). Conversely, if $\mu_0 < 1$, class 0 is weighted higher, meaning there will be lesser false positives.

Requires validation.

Solution 3. Resampling the data base:

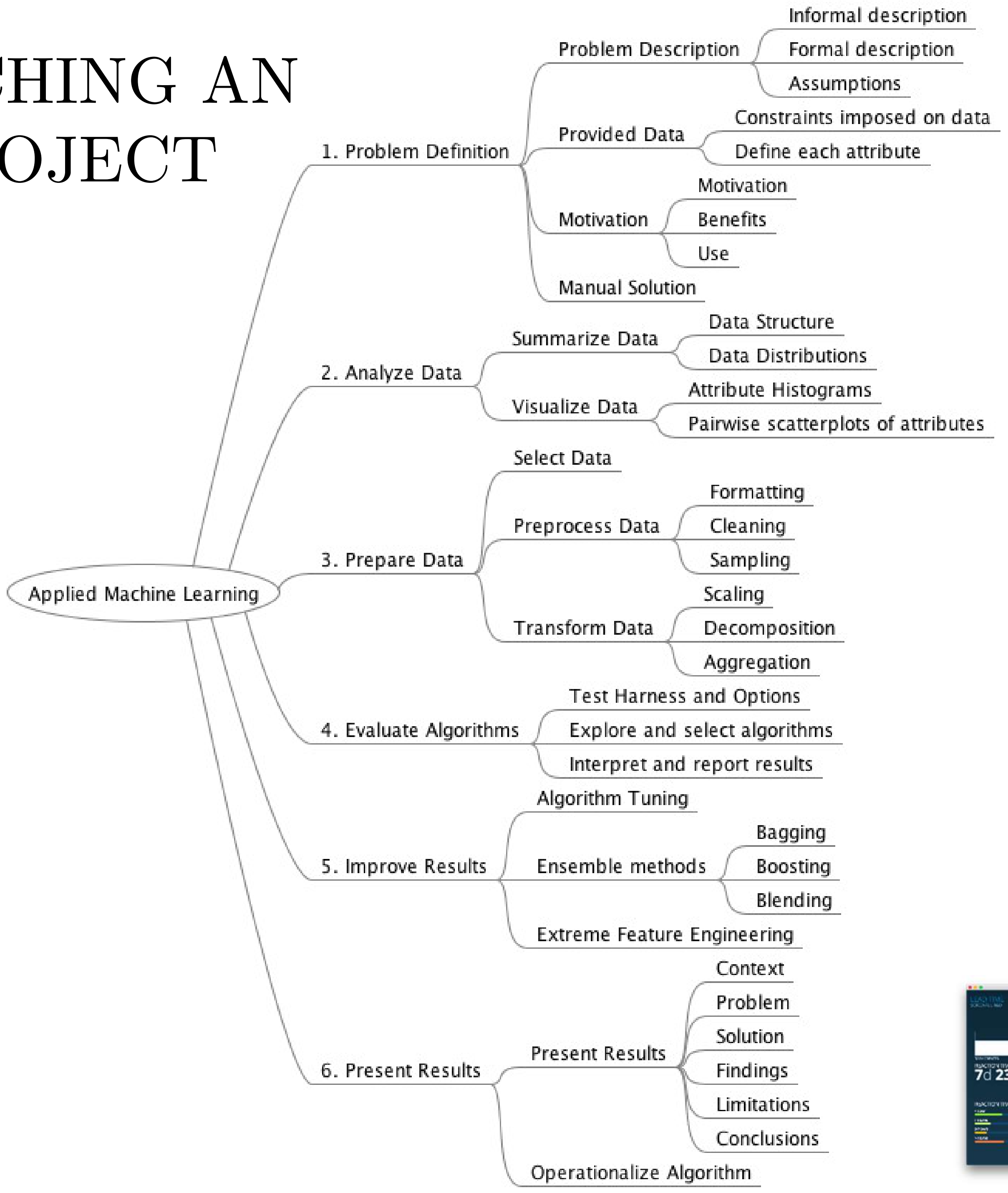
- Undersample the most represented class, may lead to bias.
- Oversample by replication the least represented class, prone to overfit.
- Using GAN
- Creating new synthetic training vectors **x**' from the minority class to increase its cardinality (e.g. SMOTE oversampling):

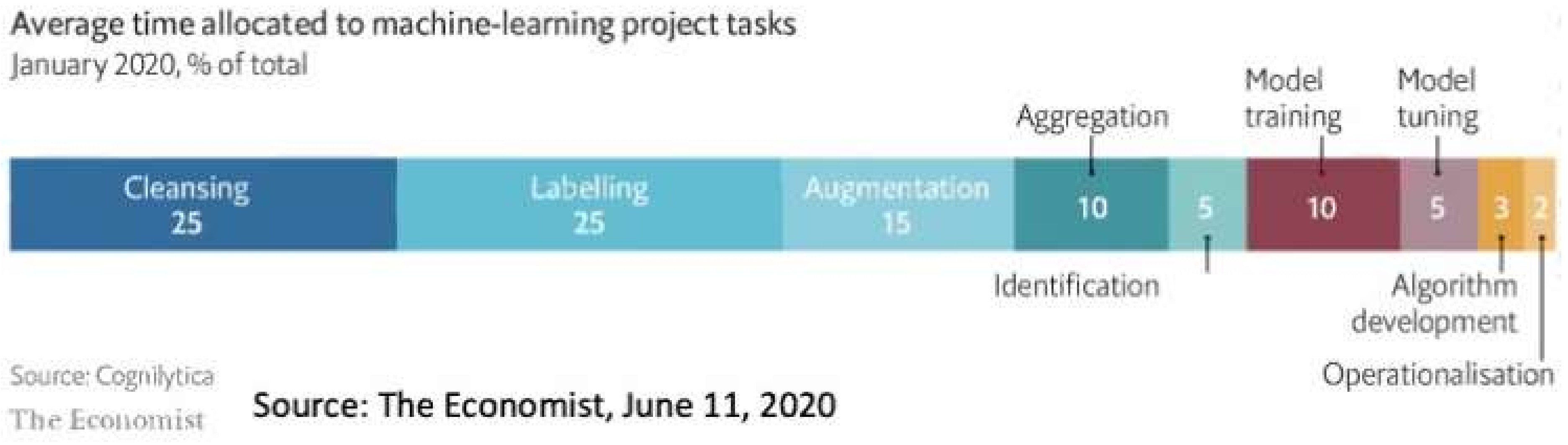
$\mathbf{x}' = \alpha \mathbf{x}_i + (1-\alpha)\mathbf{x}_j$ α is a random value in (0,1)



Imbalanced Classification | Handling Imbalanced Data using Python (analyticsvidhya.com)

8. LAUNCHING AN ML PROJECT





Source: The Economist, June 11, 2020

9. ETHICAL RISKS OF ML

ML are being used as an effective way to manage socio-economic resources (specially in the USA, and growing all over):

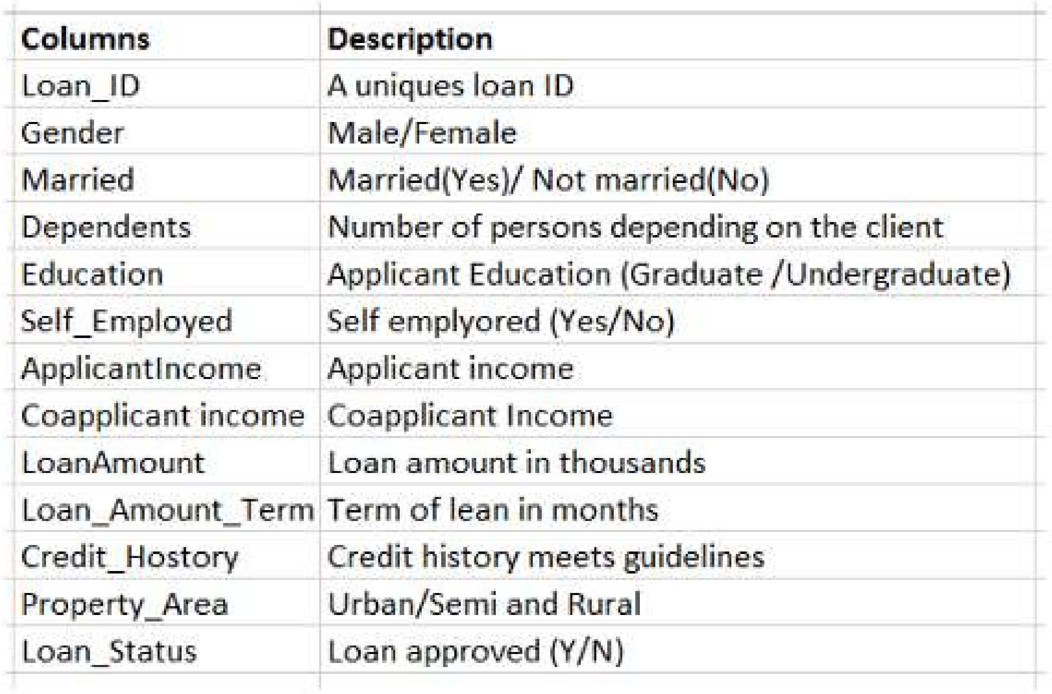
- insurance pricing
- targeted advertising (marketing, political campaigning)
- rating the quality of education
- justice administration
- getting a job

Care has to be taken not to promote misuses, like illegal use of data or making decisions that might harm the poor, reinforce racism, or amplify social inequality.

https://elpais.com/elpais/2018/04/12/planeta_futuro/1523546166_758362.html

Example: If a low income student can't get a loan because a lending model deems him too risky (by virtue of his/her race or zip code), he/she's then cut off from the kind of education that could pull him out of poverty, and a vicious spiral ensues.

Features in a database for a competition on ML-based loan prediction:



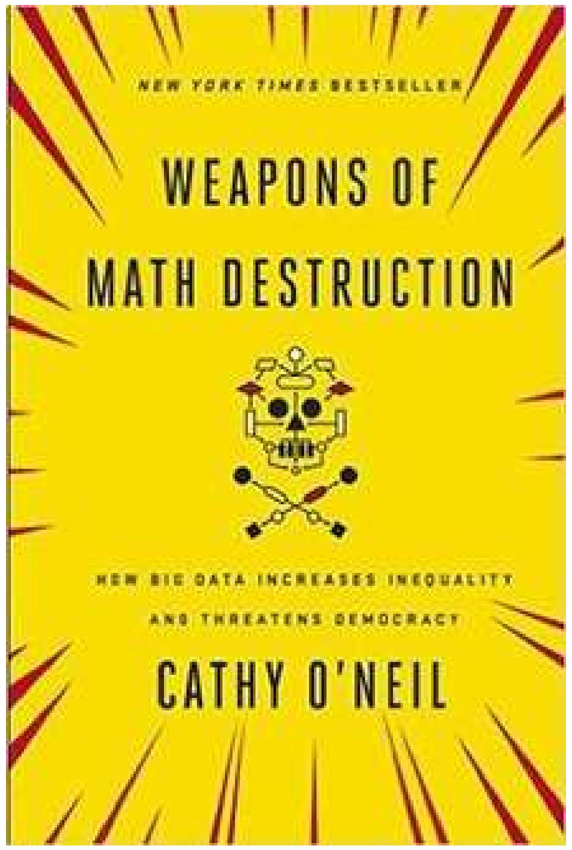
Columns	Description
Loan_ID	A uniques loan ID
Gender	Male/Female
Married	Married(Yes)/ Not married(No)
Dependents	Number of persons depending on the client
Education	Applicant Education (Graduate /Undergraduate)
Self_Employed	Self emplyored (Yes/No)
ApplicantIncome	Applicant income
Coapplicant income	Coapplicant Income
LoanAmount	Loan amount in thousands
Loan_Amount_Term	Term of lean in months
Credit_Hostory	Credit history meets guidelines
Property_Area	Urban/Semi and Rural
Loan_Status	Loan approved (Y/N)

Which ones do you think are critical in generating unfair decisions?

https://www.youtube.com/watch?v=eRUEVYndh9c
https://www.ajl.org/

Wrong use of ML tools leads to prediction machines that are:

- opaque
- based on surrogate data
- unregulated
- based on generic profiles rather than on past personal history
- not revised based on past predictions
- mainly guided by economical profit
- other?

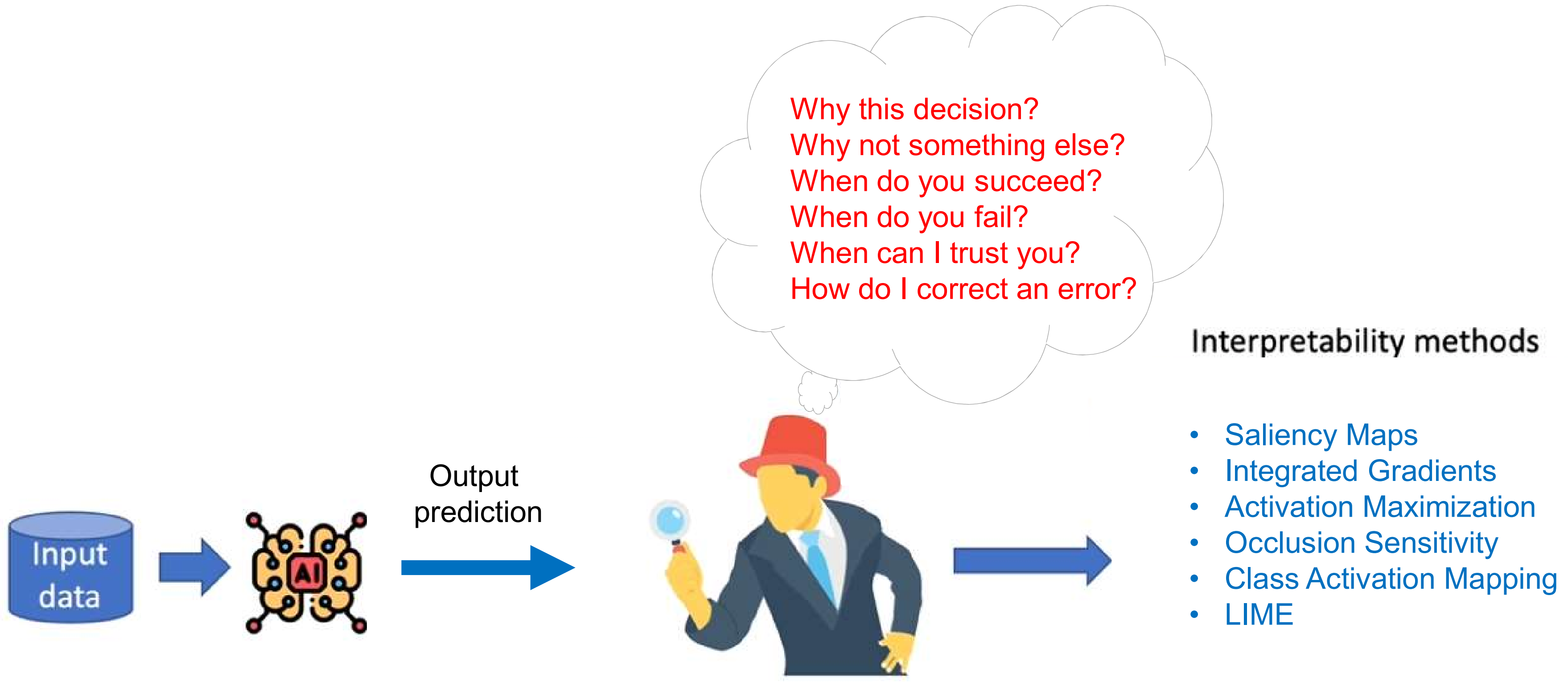

NEW YORK TIMES BESTSELLER
WEAPONS OF MATH DESTRUCTION
HOW BIG DATA INCREASES INEQUALITY AND THREATENS DEMOCRACY
CATHY O'NEIL

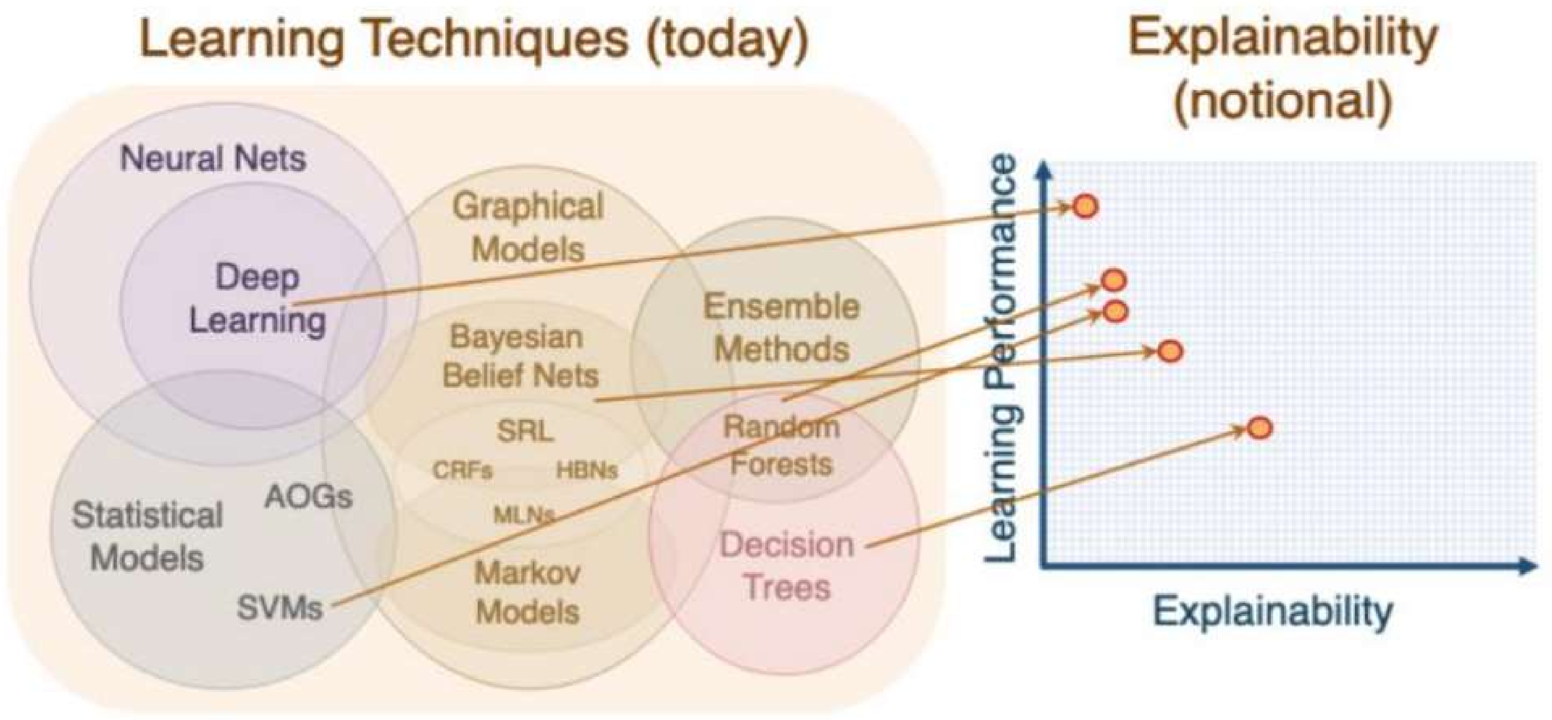
The target of data scientist should be to avoid those pitfalls by…

- Developing transparent and explainable ML algorithms
- Improving the tradeoffs between performance and fairness
- Deriving mechanisms to supervise decisions
- Using data not leading to biased algorithms

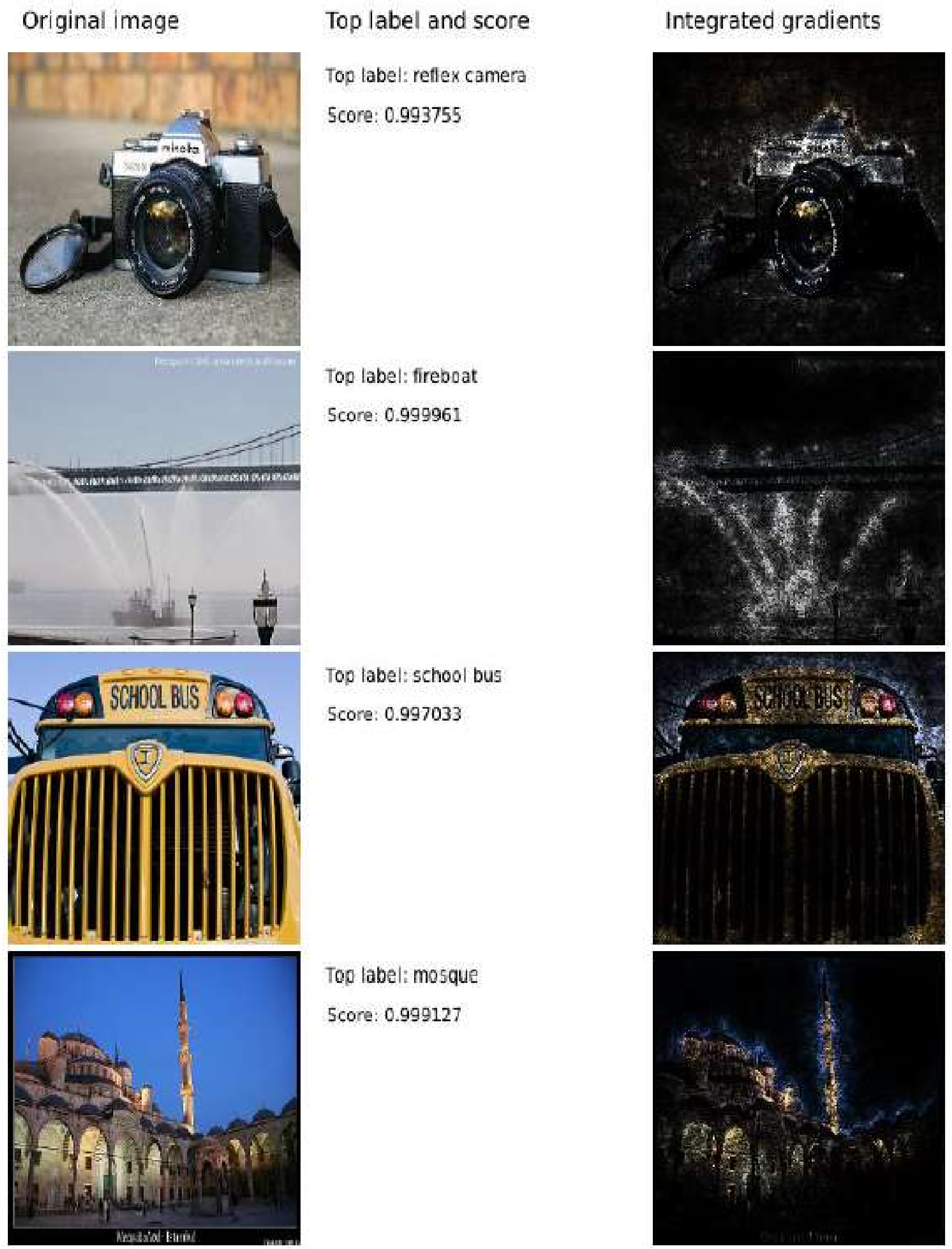
Opacity: not always possible to explain predictions

This is difficulting the adoption of ML in daylife tasks.

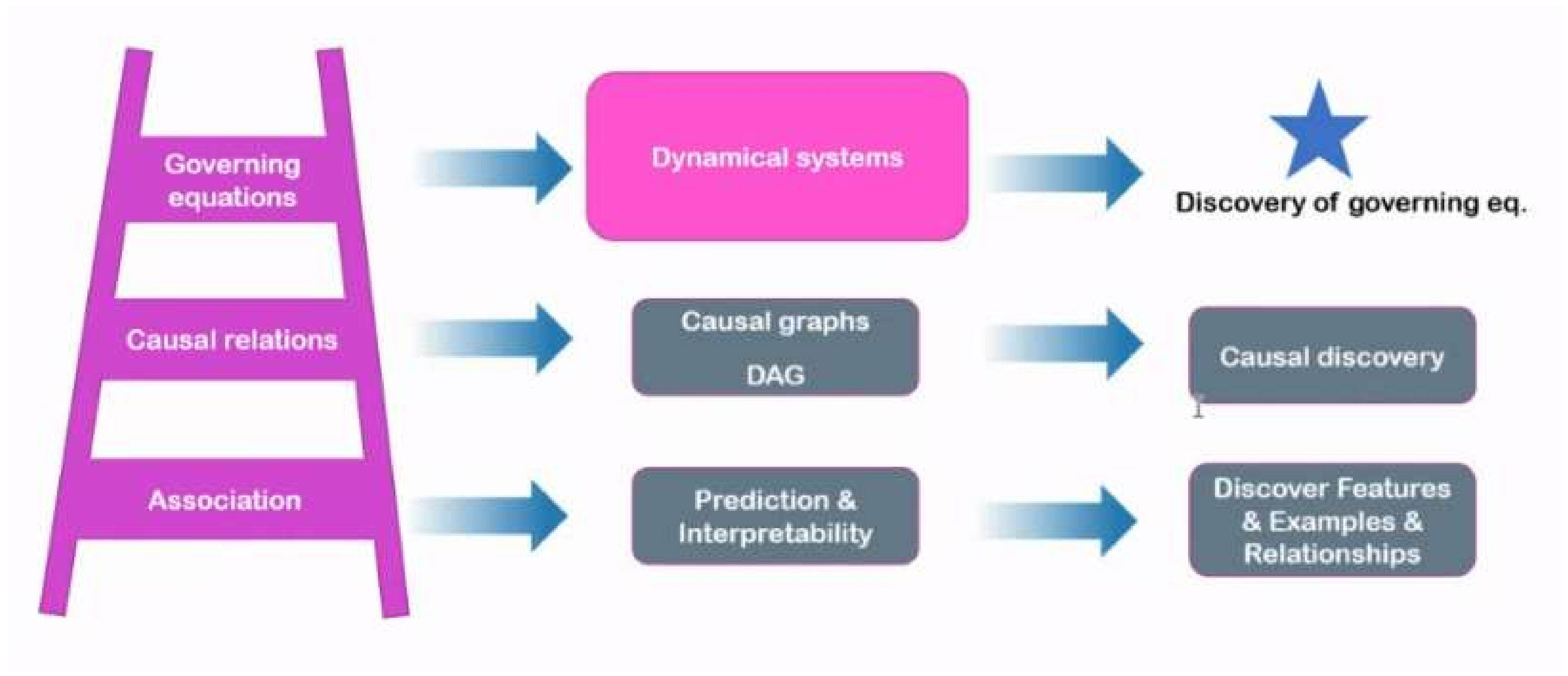




A big initiative has been endeavoured at DARPA
https://www.darpa.mil/program/explainable-artificial-intelligence



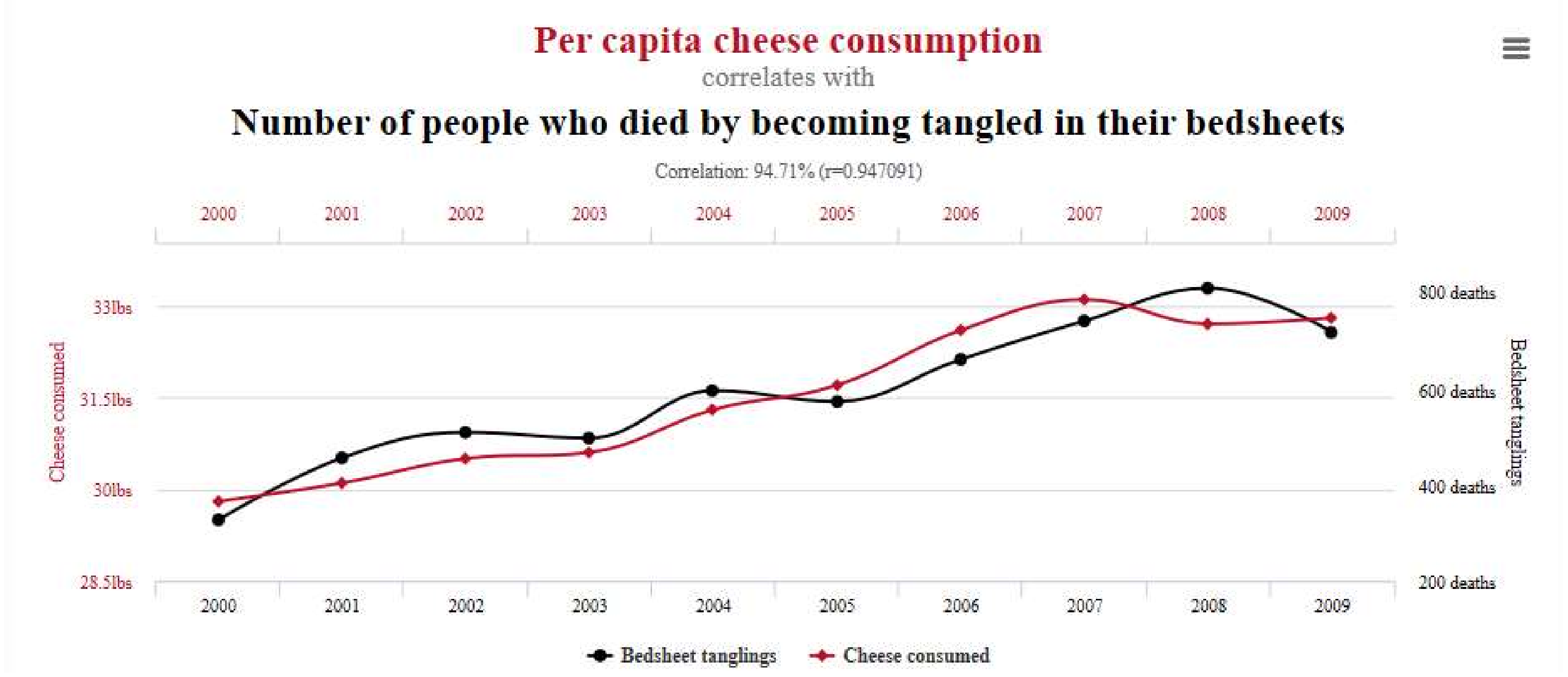
From M. Sundararajan, A. Taly, Q. Yan, “Axiomatic Attribution for Deep Networks”, arxiv 2017


Governing equations
Dynamical systems
Discovery of governing eq.
Causal relations
Causal graphs
DAG
Causal discovery
Association
Prediction & Interpretability
Discover Features & Examples & Relationships

Use of surrogate data: correlation is not causation

Use of surrogate data: correlation is not causation

All ML methods described capture correlation between features and labels, but correlation does not imply causality (https://www.tylervigen.com/spurious-correlations):



Use of surrogate data: correlation is not causation

We need to distinguish between prediction and control in the use of ML. Control requires identifying causation, while prediction doesn't.

https://towardsdatascience.com/be-careful-when-interpreting-predictive-models-in-search-of-causal-insights-e68626e664b6

Correlation between A and B may mean one of these possibilities:

$A \rightarrow B$ $\quad B \rightarrow A$ $\quad A \leftarrow R \rightarrow B$ $\quad$ or $\quad$ pure casuality

Assessing causality requires a fully new techniques, an area still in its infancy.

Ethics in algorithms

ETICAS is a foundation that audits algorithms that impact our daily life: https://www.businessinsider.es/estos-son-algoritmos-deciden-vida-diaria-944501

The list of algorithms ETICAS (and the status of audited or not) is found here: https://airtable.com/shrsAN2oTf68kM6O9/tblG2604tSoMOcwWX?backgroundColor=teal&viewControls=on

The main identified damages are related to age, gender, race or disability.