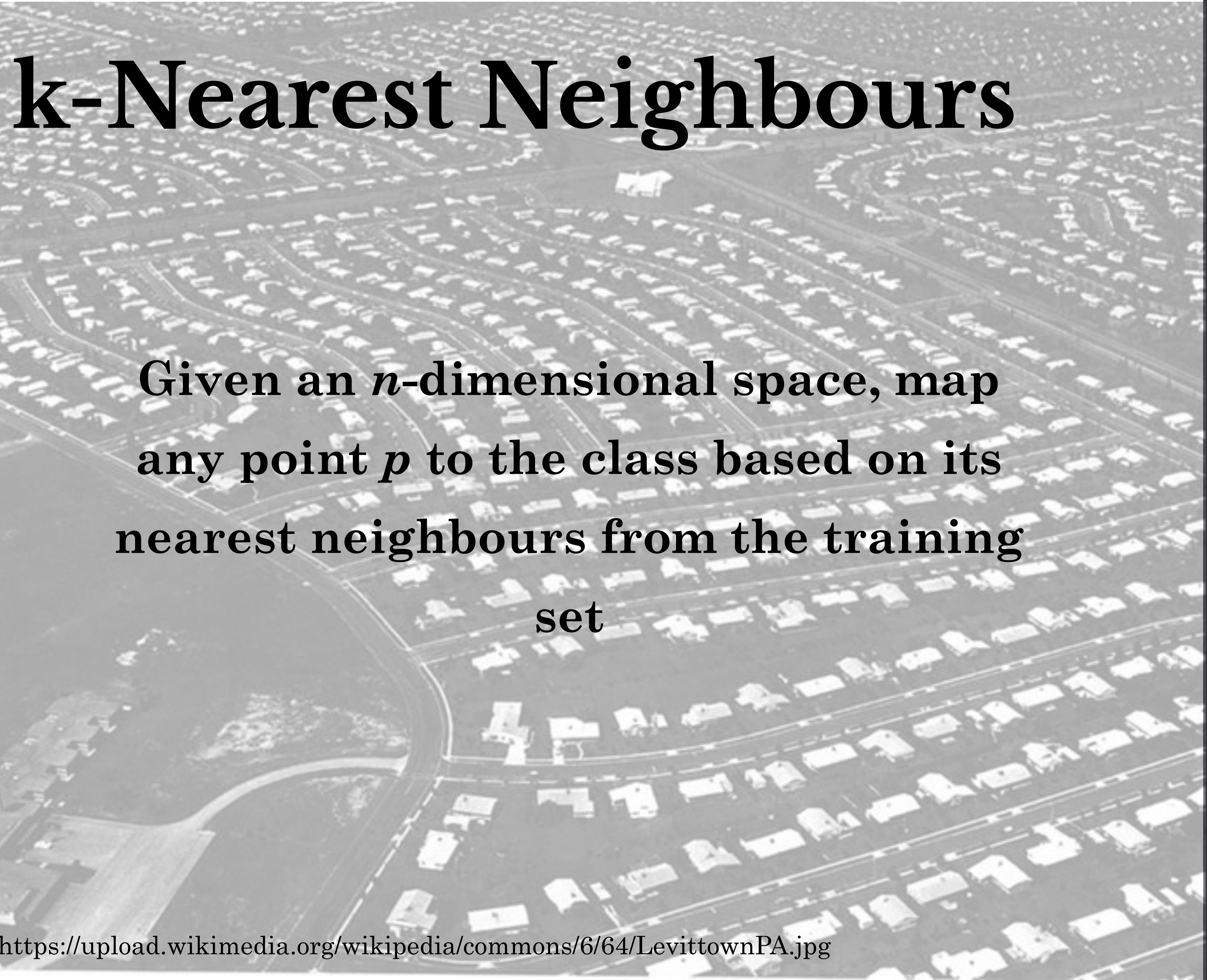
A whirlwind tour of models

CSCI 4181 / 6802

http://pixabay.com/en/road-sign-arrows-arrow-direction-64059/

Overview

1. K-Nearest Neighbours
2. Naïve Bayes
3. SVMs
4. Decision Trees
5. Ensembles - Random Forests
6. Ensembles - Boosting

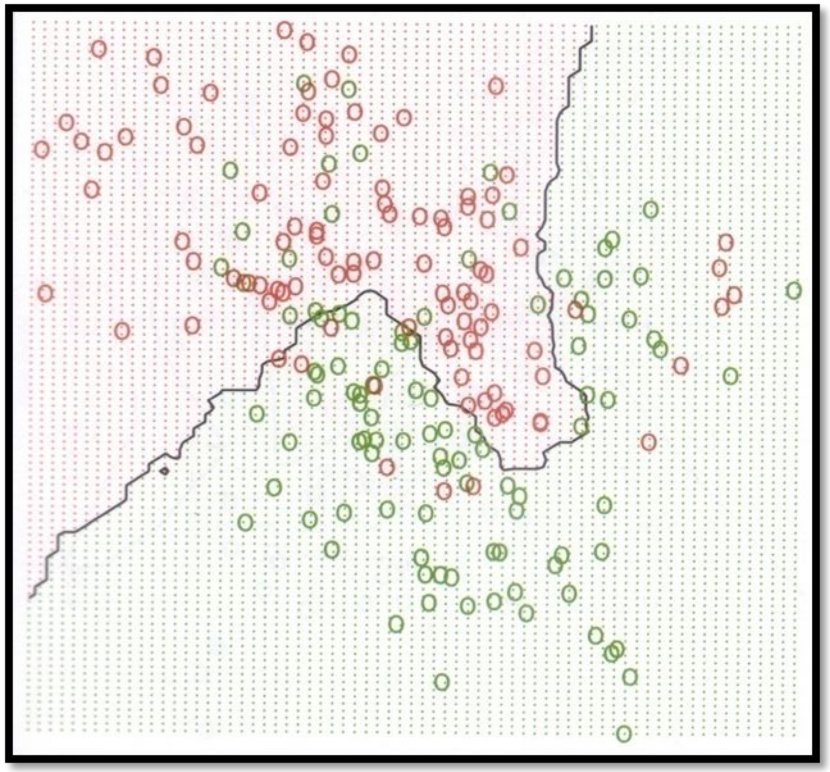


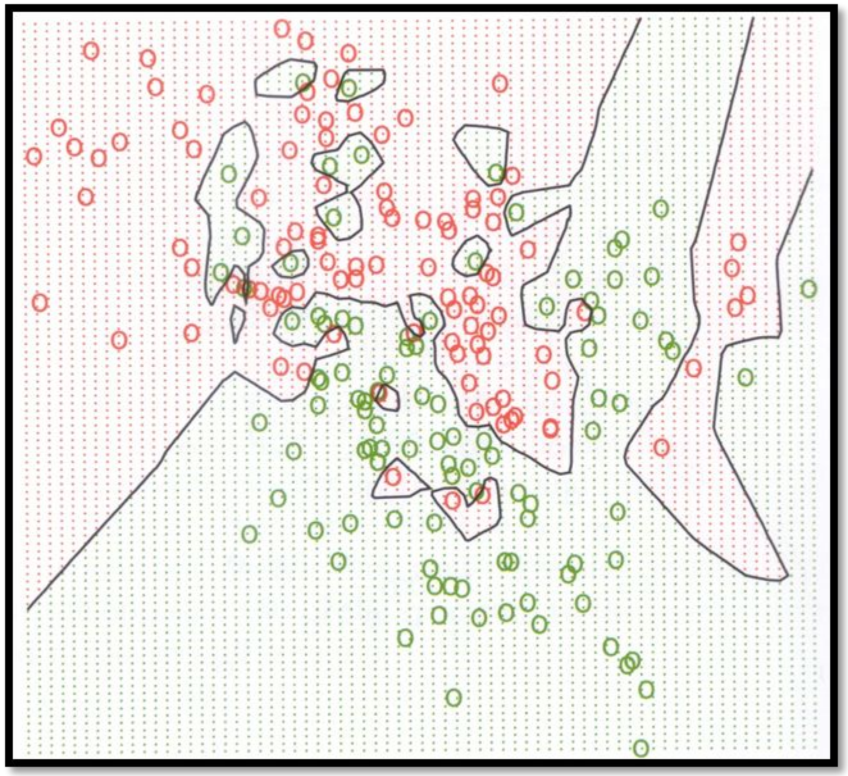
k-Nearest Neighbours

Given an *n*-dimensional space, map any point *p* to the class based on its nearest neighbours from the training set

https://upload.wikimedia.org/wikipedia/commons/6/64/LevittownPA.jpg

Procedure


15-NN


1-NN
(Voronoi
tesselation)

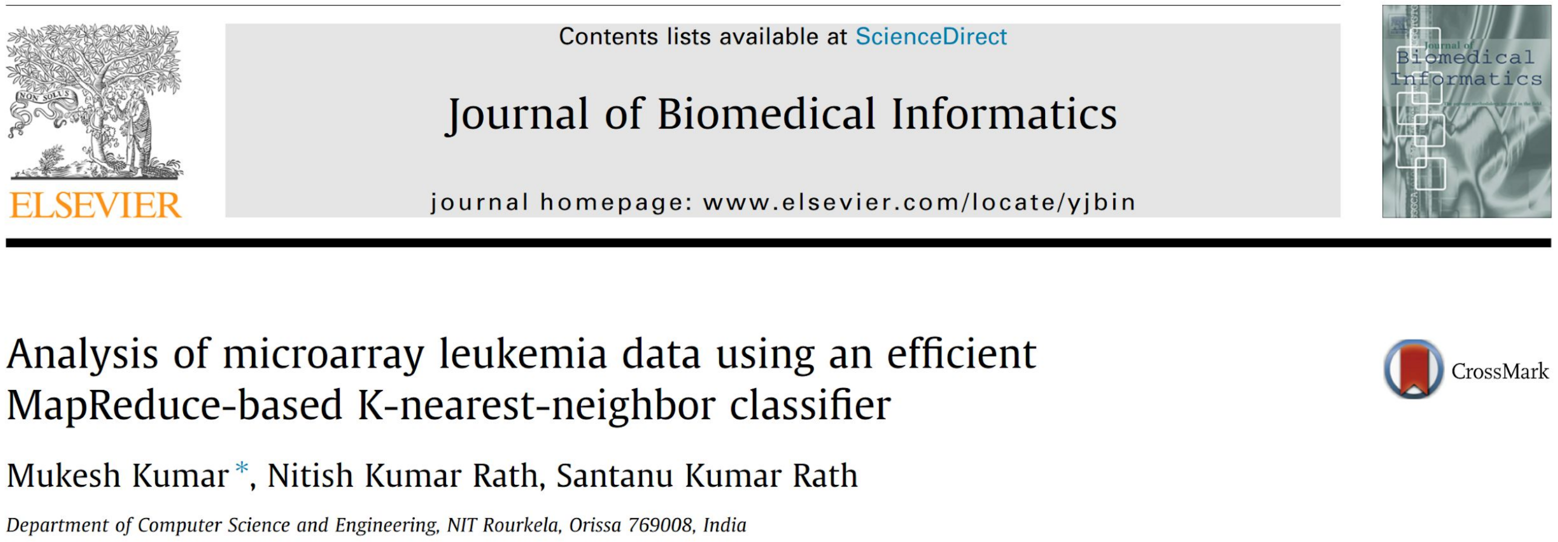
Hastie, p. 15-16

Training

No training *per se*, since modeling the decision boundary could be quite complex

Instead, find the labels of the *k* closest (e.g., Euclidean distance) training vectors

$$D_{Euclidean} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$



Contents lists available at ScienceDirect

Journal of Biomedical Informatics

journal homepage: www.elsevier.com/locate/yjbin

Analysis of microarray leukemia data using an efficient MapReduce-based K-nearest-neighbor classifier

Mukesh Kumar*, Nitish Kumar Rath, Santanu Kumar Rath

Department of Computer Science and Engineering, NIT Rourkela, Orissa 769008, India

Objective: classify different groups of leukemia subjects based on shared patterns of gene expression

Note: this paper spends a *lot* of time on MapReduce aspects of feature selection and k-NN classification – we're not going to discuss that part

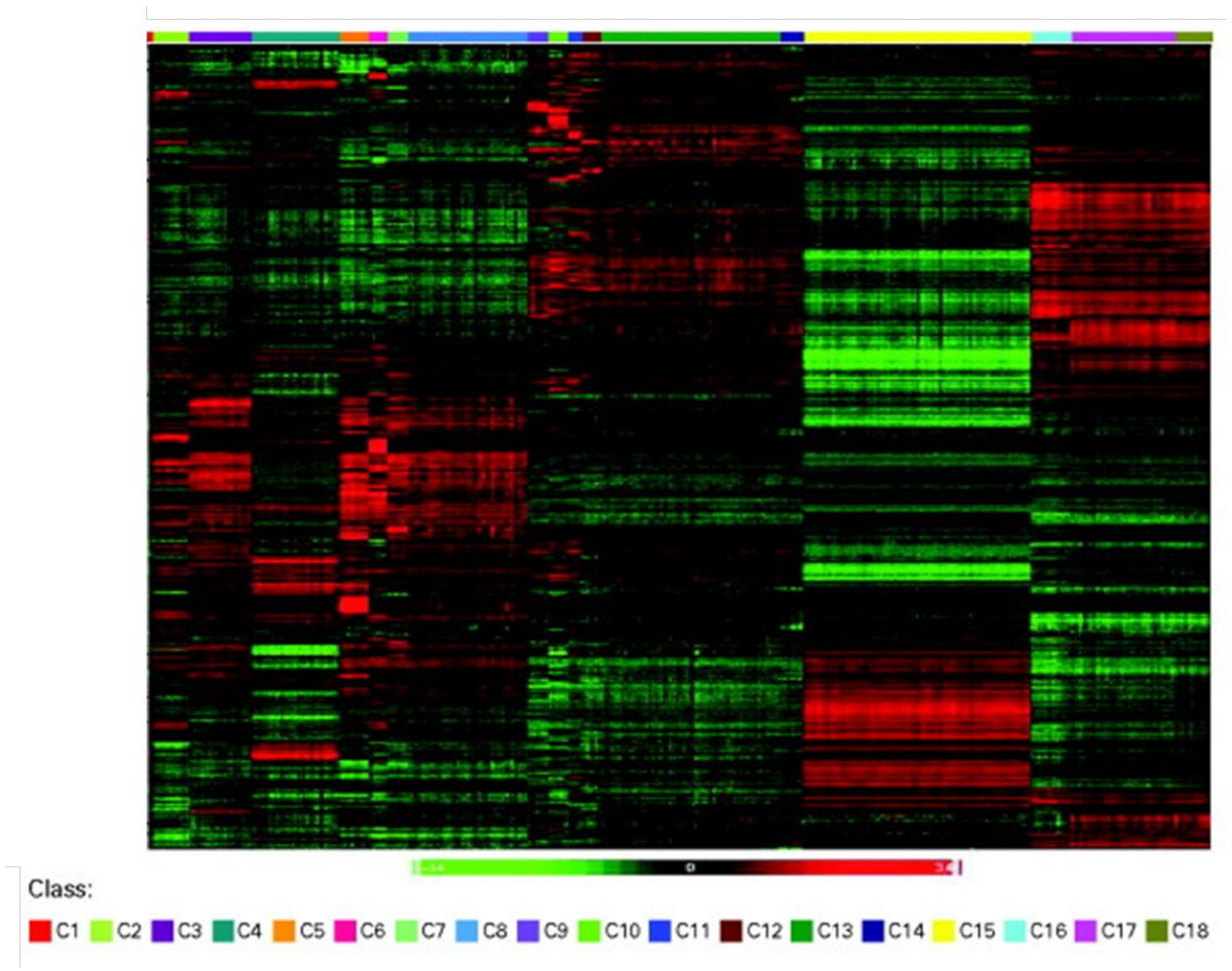
Kumar et al. (2016)

Expression Microarrays

Patients
(by class)

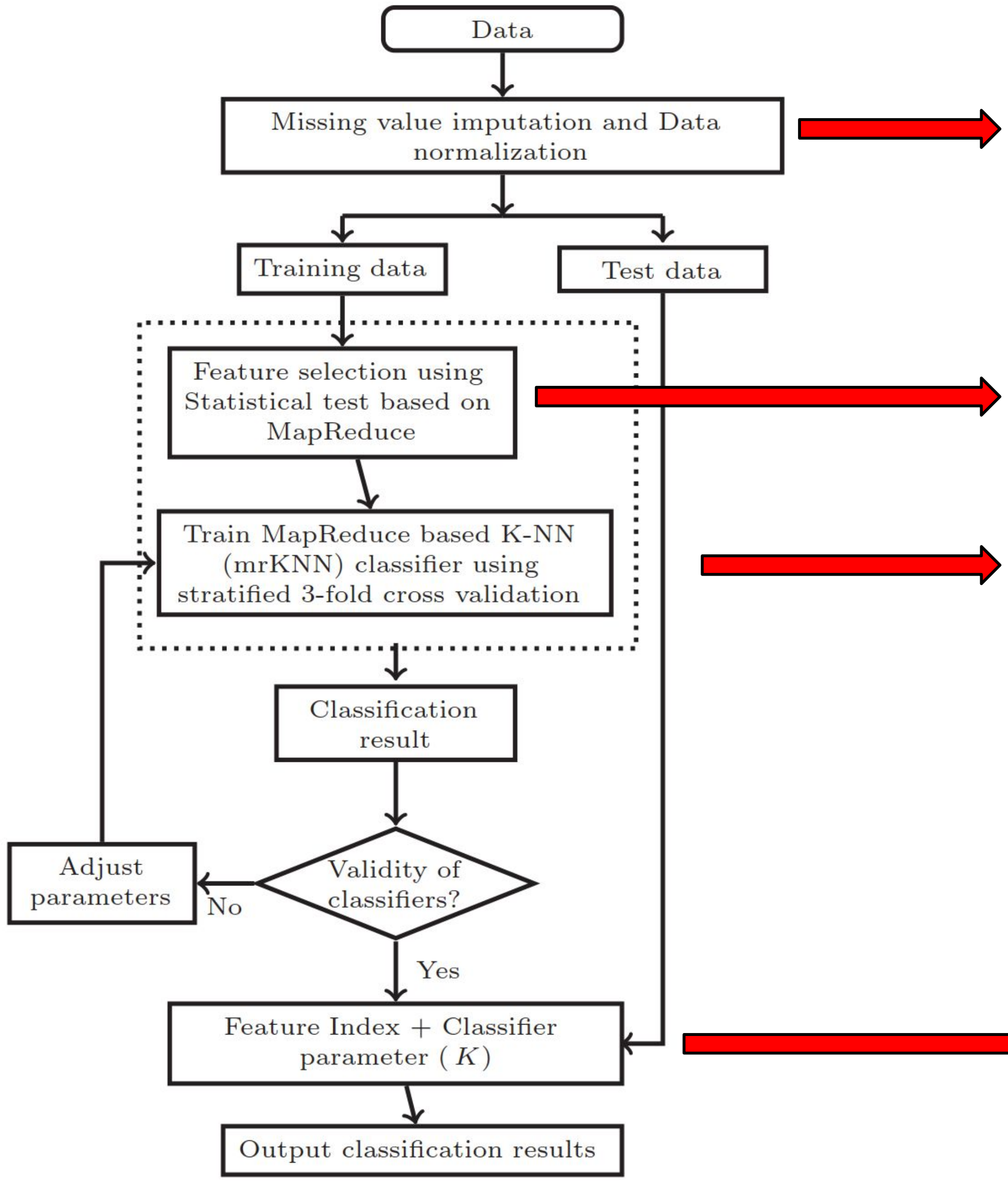
Genes

HIGH expression
LOW expression



Haferlach et al. (2010) *J Clin Oncol*

Workflow


Data
Missing value imputation and Data normalization
Training data
Test data
Feature selection using Statistical test based on MapReduce
Train MapReduce based K-NN (mrKNN) classifier using stratified 3-fold cross validation
Classification result
Validity of classifiers?
No
Adjust parameters
Yes
Feature Index + Classifier parameter (K)
Output classification results

Filling in the blanks

Simple statistical tests (e.g., ANOVA)

Choose K using cross-validation in the *training* set

Now try the test data

Datasets

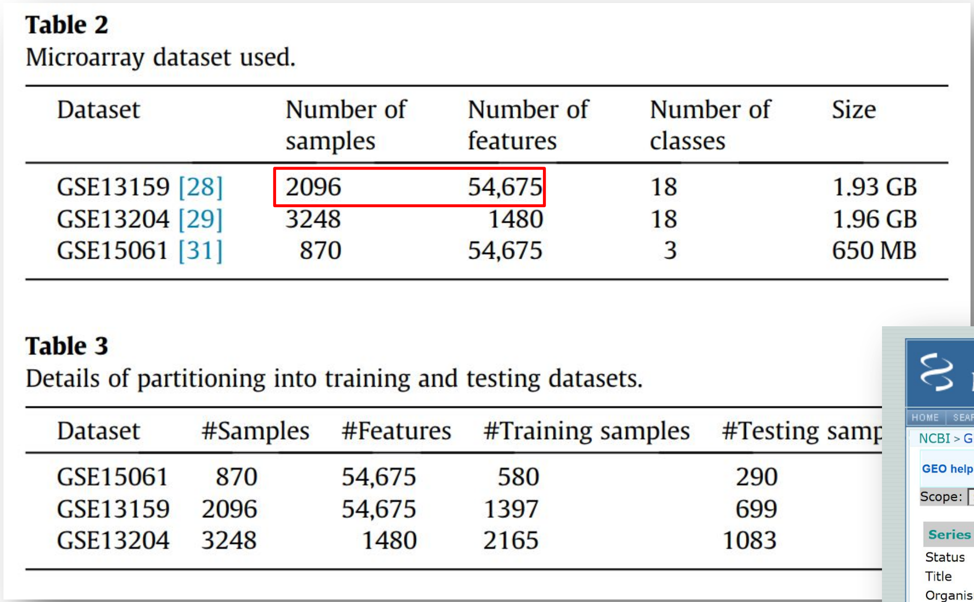
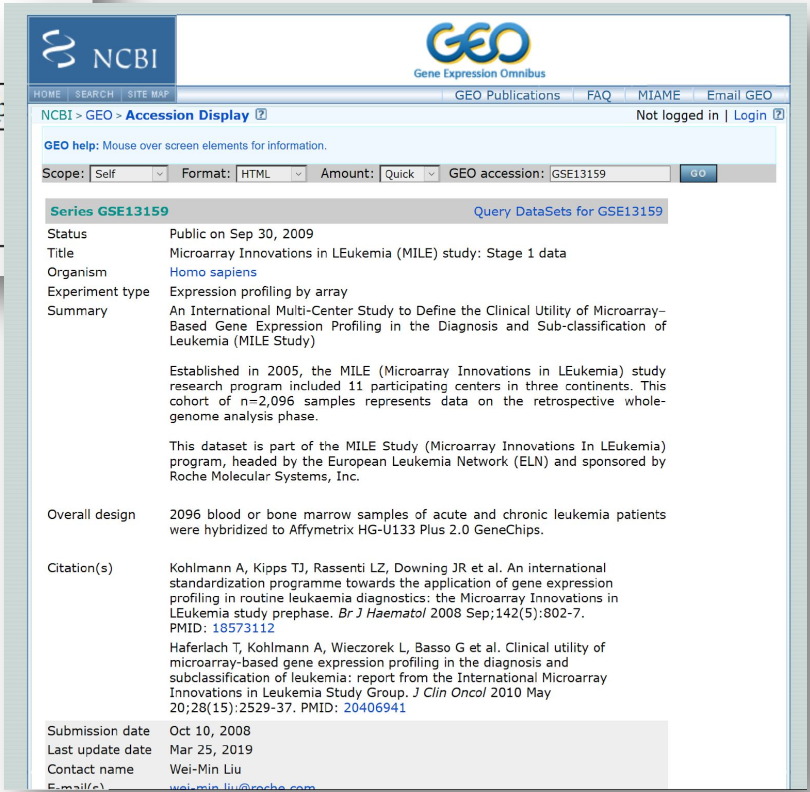


Table 2
Microarray dataset used.

Dataset	Number of samples	Number of features	Number of classes	Size
GSE13159 [28]	2096	54,675	18	1.93 GB
GSE13204 [29]	3248	1480	18	1.96 GB
GSE15061 [31]	870	54,675	3	650 MB

Table 3
Details of partitioning into training and testing datasets.

Dataset	#Samples	#Features	#Training samples	#Testing samp
GSE15061	870	54,675	580	290
GSE13159	2096	54,675	1397	699
GSE13204	3248	1480	2165	1083



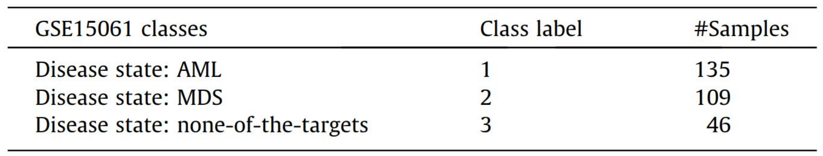
GEO help: Mouse over screen elements for information.

Scope: Self   Format: HTML   Amount: Quick   GEO accession: GSE13159

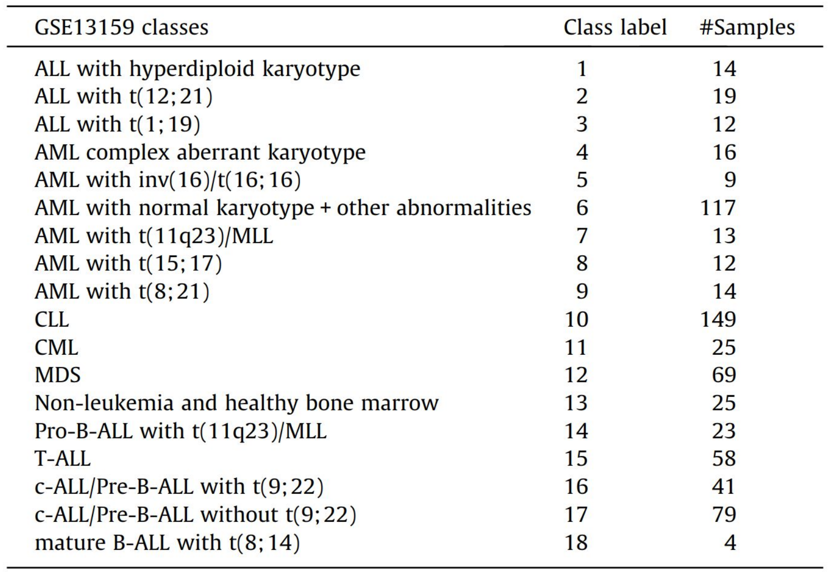
Series GSE13159   Query DataSets for GSE13159

Status	Public on Sep 30, 2009
Title	Microarray Innovations in LEukemia (MILE) study: Stage 1 data
Organism	Homo sapiens
Experiment type	Expression profiling by array
Summary	An International Multi-Center Study to Define the Clinical Utility of Microarray–Based Gene Expression Profiling in the Diagnosis and Sub-classification of Leukemia (MILE Study) Established in 2005, the MILE (Microarray Innovations in LEukemia) study research program included 11 participating centers in three continents. This cohort of n=2,096 samples represents data on the retrospective whole-genome analysis phase. This dataset is part of the MILE Study (Microarray Innovations In LEukemia) program, headed by the European Leukemia Network (ELN) and sponsored by Roche Molecular Systems, Inc.
Overall design	2096 blood or bone marrow samples of acute and chronic leukemia patients were hybridized to Affymetrix HG-U133 Plus 2.0 GeneChips.
Citation(s)	Kohlmann A, Kipps TJ, Rassenti LZ, Downing JR et al. An international standardization programme towards the application of gene expression profiling in routine leukaemia diagnostics: the Microarray Innovations in LEukemia study prephase. *Br J Haematol* 2008 Sep;142(5):802-7. PMID: 18573112 Haferlach T, Kohlmann A, Wieczorek L, Basso G et al. Clinical utility of microarray-based gene expression profiling in the diagnosis and subclassification of leukemia: report from the International Microarray Innovations in Leukemia Study Group. *J Clin Oncol* 2010 May 20;28(15):2529-37. PMID: 20406941
Submission date	Oct 10, 2008
Last update date	Mar 25, 2019
Contact name	Wei-Min Liu
E-mail(s)	wei-min.liu@roche.com

An impressive number of classes



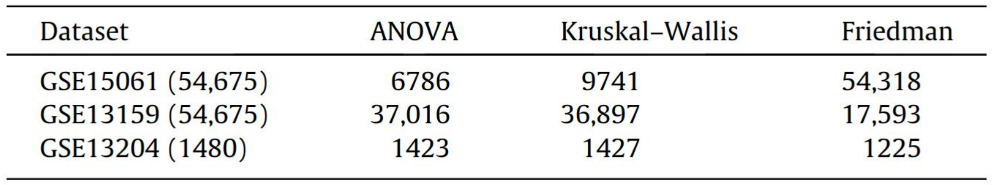
GSE15061 classes	Class label	#Samples
Disease state: AML	1	135
Disease state: MDS	2	109
Disease state: none-of-the-targets	3	46



GSE13159 classes	Class label	#Samples
ALL with hyperdiploid karyotype	1	14
ALL with t(12;21)	2	19
ALL with t(1;19)	3	12
AML complex aberrant karyotype	4	16
AML with inv(16)/t(16;16)	5	9
AML with normal karyotype + other abnormalities	6	117
AML with t(11q23)/MLL	7	13
AML with t(15;17)	8	12
AML with t(8;21)	9	14
CLL	10	149
CML	11	25
MDS	12	69
Non-leukemia and healthy bone marrow	13	25
Pro-B-ALL with t(11q23)/MLL	14	23
T-ALL	15	58
c-ALL/Pre-B-ALL with t(9;22)	16	41
c-ALL/Pre-B-ALL without t(9;22)	17	79
mature B-ALL with t(8;14)	18	4

Feature selection

- We probably don’t want to use over 54,000 features.
- The authors used basic statistical approaches such as ANOVA to determine which features best differentiated different classes.
- p-value filter for inclusion
- Feature selection results were…weird

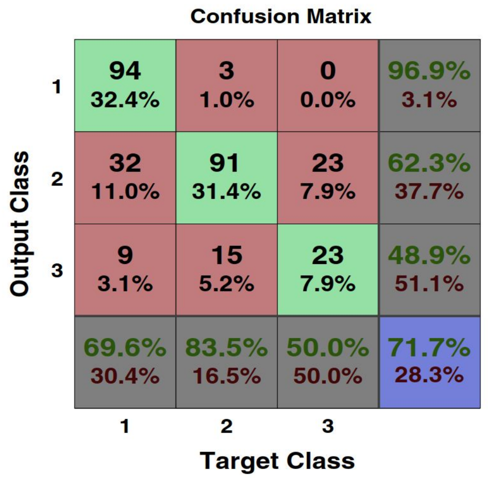


Dataset	ANOVA	Kruskal–Wallis	Friedman
GSE15061 (54,675)	6786	9741	54,318
GSE13159 (54,675)	37,016	36,897	17,593
GSE13204 (1480)	1423	1427	1225

K-NN operation

- For each test data point, identify the *K* closest points from the training set (i.e., most similar profiles according to Euclidean distance)
- What are the labels of these *K* points?
- Choose the modal class, i.e., the class that is most frequently represented in the neighbour set
- Accuracy = % of all samples that were correctly classified

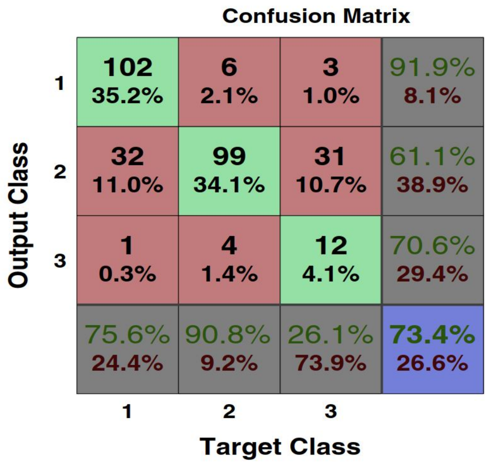
Results



Confusion Matrix

Output Class \ Target Class	1	2	3	
1	94 32.4%	3 1.0%	0 0.0%	96.9% 3.1%
2	32 11.0%	91 31.4%	23 7.9%	62.3% 37.7%
3	9 3.1%	15 5.2%	23 7.9%	48.9% 51.1%
	69.6% 30.4%	83.5% 16.5%	50.0% 50.0%	71.7% 28.3%

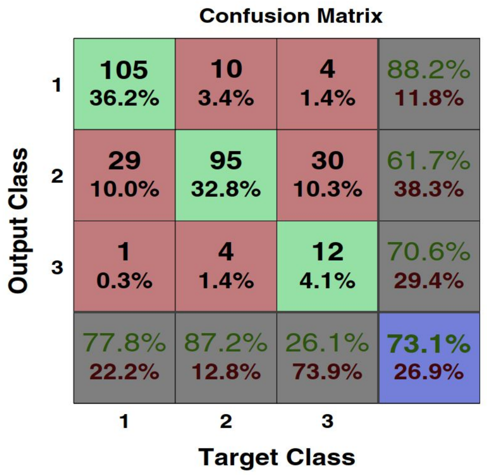
(a) ANOVA (f=6,786,K=21)



Confusion Matrix

Output Class \ Target Class	1	2	3	
1	102 35.2%	6 2.1%	3 1.0%	91.9% 8.1%
2	32 11.0%	99 34.1%	31 10.7%	61.1% 38.9%
3	1 0.3%	4 1.4%	12 4.1%	70.6% 29.4%
	75.6% 24.4%	90.8% 9.2%	26.1% 73.9%	73.4% 26.6%

(b) Kruskal-Wallis (f=9,741,K=17)



Confusion Matrix

Output Class \ Target Class	1	2	3	
1	105 36.2%	10 3.4%	4 1.4%	88.2% 11.8%
2	29 10.0%	95 32.8%	30 10.3%	61.7% 38.3%
3	1 0.3%	4 1.4%	12 4.1%	70.6% 29.4%
	77.8% 22.2%	87.2% 12.8%	26.1% 73.9%	73.1% 26.9%

(c) Friedman (f=54,318,K=21)

Accuracy is similar across all three feature sets

K is reasonably consistent

Modifications to basic k-NN

Try different distance definitions

Treat different dimensions differently
(e.g., normalize, kernel methods)

Naïve Bayes

http://www.polarbearsinternational.org/sites/default/files/legacy/D216674.jpg

The point

- Assign samples to different classes using a probabilistic approach
  - Think of it as competitive matching based on distributions of features

- Features are treated independently
  - A simplifying assumption that makes NB very fast

- Classes are assigned probabilities which can be modified by priors

Naïve Bayes

For a set of *n* classes C_1, C_2, C_3, …, C_n,

and a problem instance *x*

(usually represented with a feature vector),

Prior probability of class *i*

Likelihood of *x* given model C_i

$$p(C_i \mid x) = \frac{p(C_i)\,p(x \mid C_i)}{p(x)}$$

Probability of membership in class *i*

Probability of *x* (generally ignored)

Predicted class: C_i that maximizes $p(C_i \mid x)$

Priors

What are the expected probabilities of different classes?

Flat prior: $p(C_1) = p(C_2) = p(C_n) = 1 / n$

Informative prior: $p(C_i) \neq p(C_j)$ for some $i \neq j$

(based on what?)

Calculating likelihoods

"The probability of the data, given the model"

What is the likelihood of x, given class C_i?

Product over all features in x

$$p(x \mid C_i) = \prod_{j=1}^{q} p(x_j \mid C_i)$$

Likelihood of x, given C_i

Likelihood of x_j, given C_i

Calculating likelihoods: the independence assumption

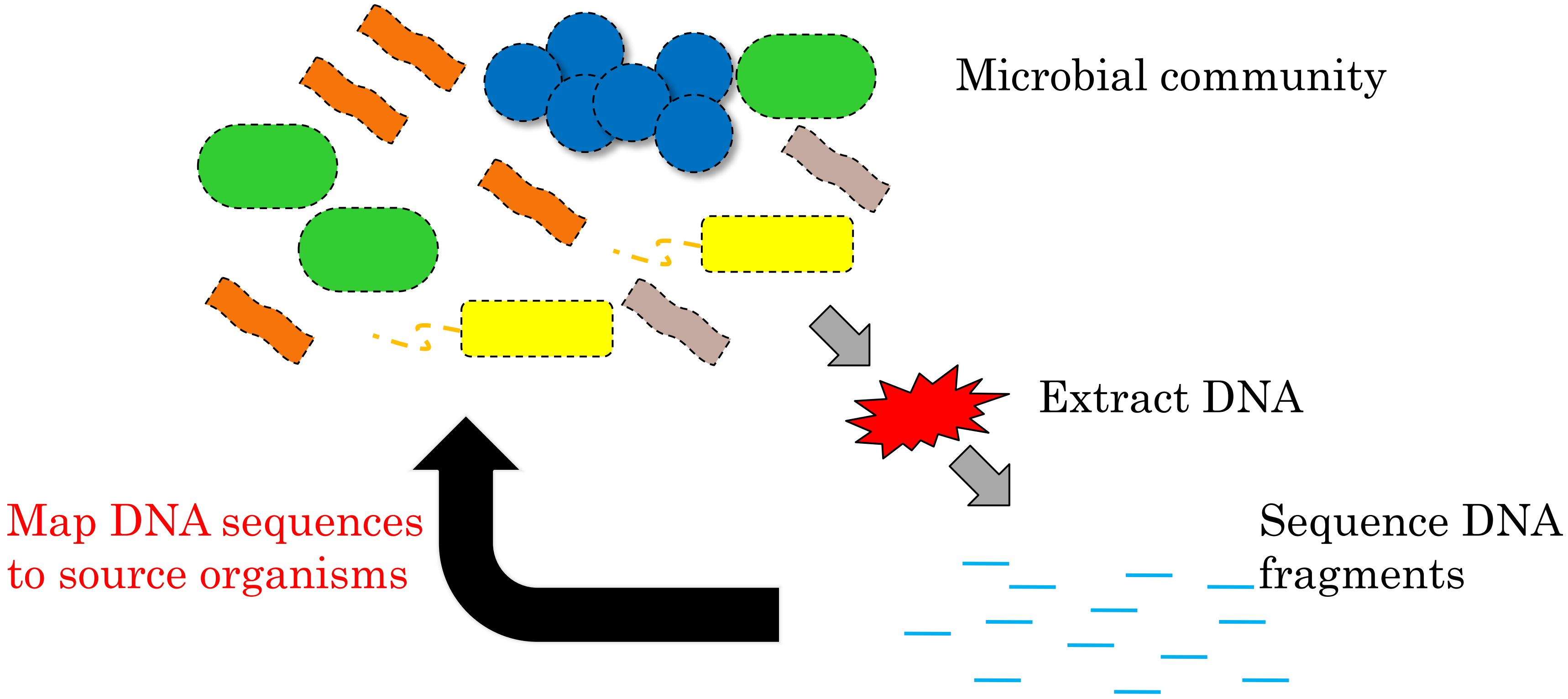
The calculation can be very complicated if x has many elements and we consider all possible dependencies among elements

(as most classifiers do)

The solution is to treat each element of x independently

NB example

Fragment Classification Package (FCP): Assigning DNA sequence fragments to originating organisms from a metagenomic sample



How do we classify these sequences?

k-mer decomposition: Build compositional models for each genome in a reference database

C_i	x
E. coli	{ AA = 0.05, AC = 0.03, AG = 0.08, AT = 0.04, CA = …
Y. pestis	{ AA = 0.05, AC = 0.04, AG = 0.10, AT = 0.02, CA = …
C. difficile	{ AA = 0.01, AC = 0.05, AG = 0.08, AT = 0.02, CA = …
E. faecium	{ AA = 0.03, AC = 0.03, AG = 0.09, AT = 0.01, CA = …

How do we classify these sequences?

k-mer decomposition: Build compositional models for each genome in a reference database

$$p(x \mid C_i) = \prod_{j=1}^{q} p(x_j \mid C_i)$$

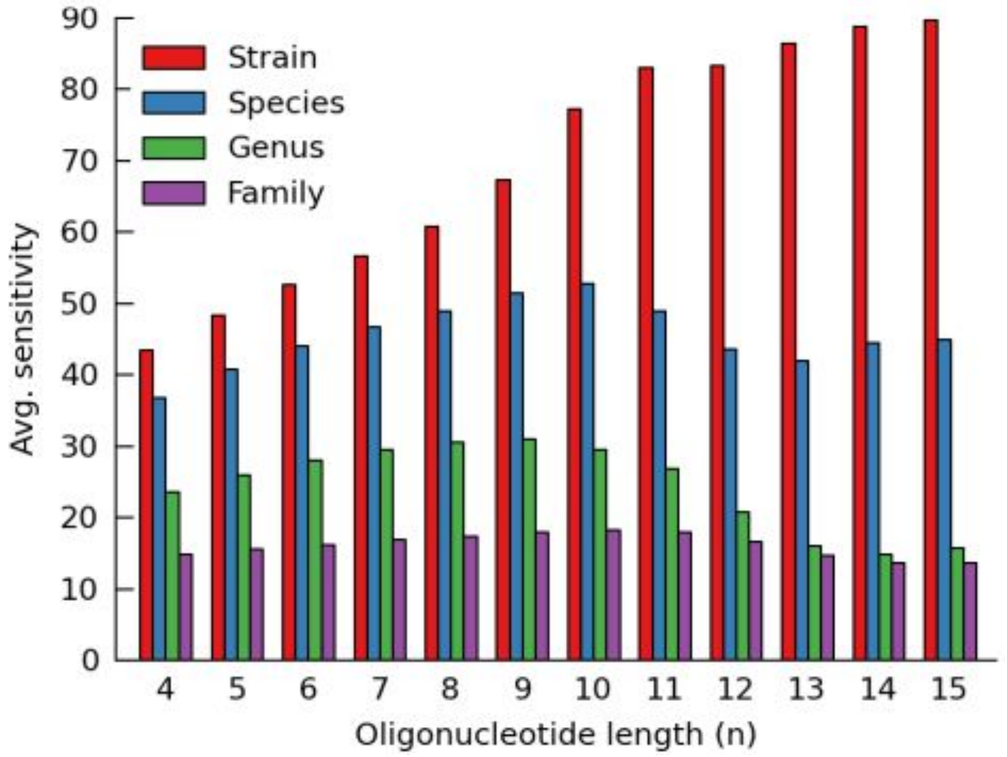
For all k-mers in fragment x…

Equal to the frequency of the k-mer j in model (= genome) C_i

The "winning" genome is the one that maximizes the likelihood (assuming a flat prior)

Trials

Simulated data



Optimal k = 9 from trials

Challenges in sequence classification

(1) *k*-mer frequencies are averages calculated over the entire genome: individual genes can and do vary

Worst offenders: recent acquisitions from other genomes (e.g. plasmids!), viral genes, rapidly evolving genes

(2) We are restricted to the genomes in our training set; new genomes are not modeled

"rank-flexible" classification: classify at a reasonable taxonomic level

(3) Sparse representations when *k* is large

Spoiler: it doesn't seem to matter all that much (we checked!)

Parks DH, MacDonald NJ, and Beiko, RG (2011) *BMC Bioinformatics*

Example: classifying glacier metagenomes

Rank flexible

A

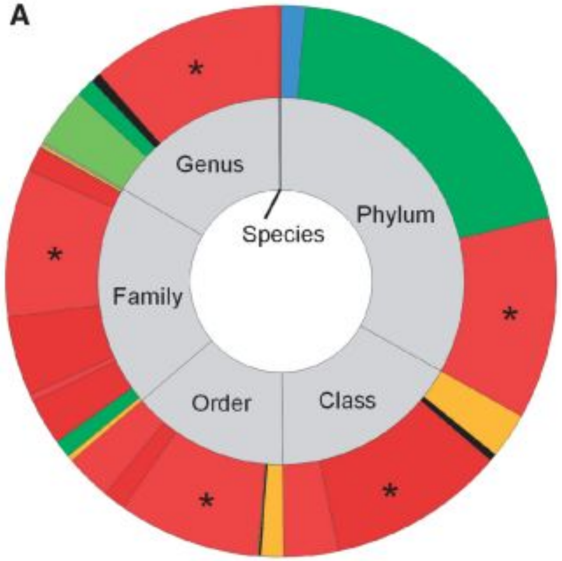


Figure 5. RITA classifications of the glacier metagenome of (26). (**A**) Rank-flexible classifications in Groups 1–3 to ranks between species and phylum. The inner ring identifies the rank at which different fragments were classified, while the outer ring shows the distribution across different labels at that rank, colored by the phylum to which the taxon belongs. Phylum colors: blue = Acidobacteria, green = Bacteroidetes, red = Proteobacteria, orange = Actinobacteria, black = other. Alternating shades of the same color are used to distinguish different taxa at the same rank from the same phylum. The taxonomic lineage of *Polaromonas* is identified with asterisks. (**B**) Rank-specific classifications at the phylum (outer ring) and genus (inner ring) levels, with color scheme as in panel A. Deepest red and green represent aggregated 'other' genera of Proteobacteria and Bacteroidetes.

MacDonald NJ, Parks DH, Beiko RG (2012) Rapid identification of high-confidence taxonomic assignments for metagenomic data. *Nucleic Acids Res.*

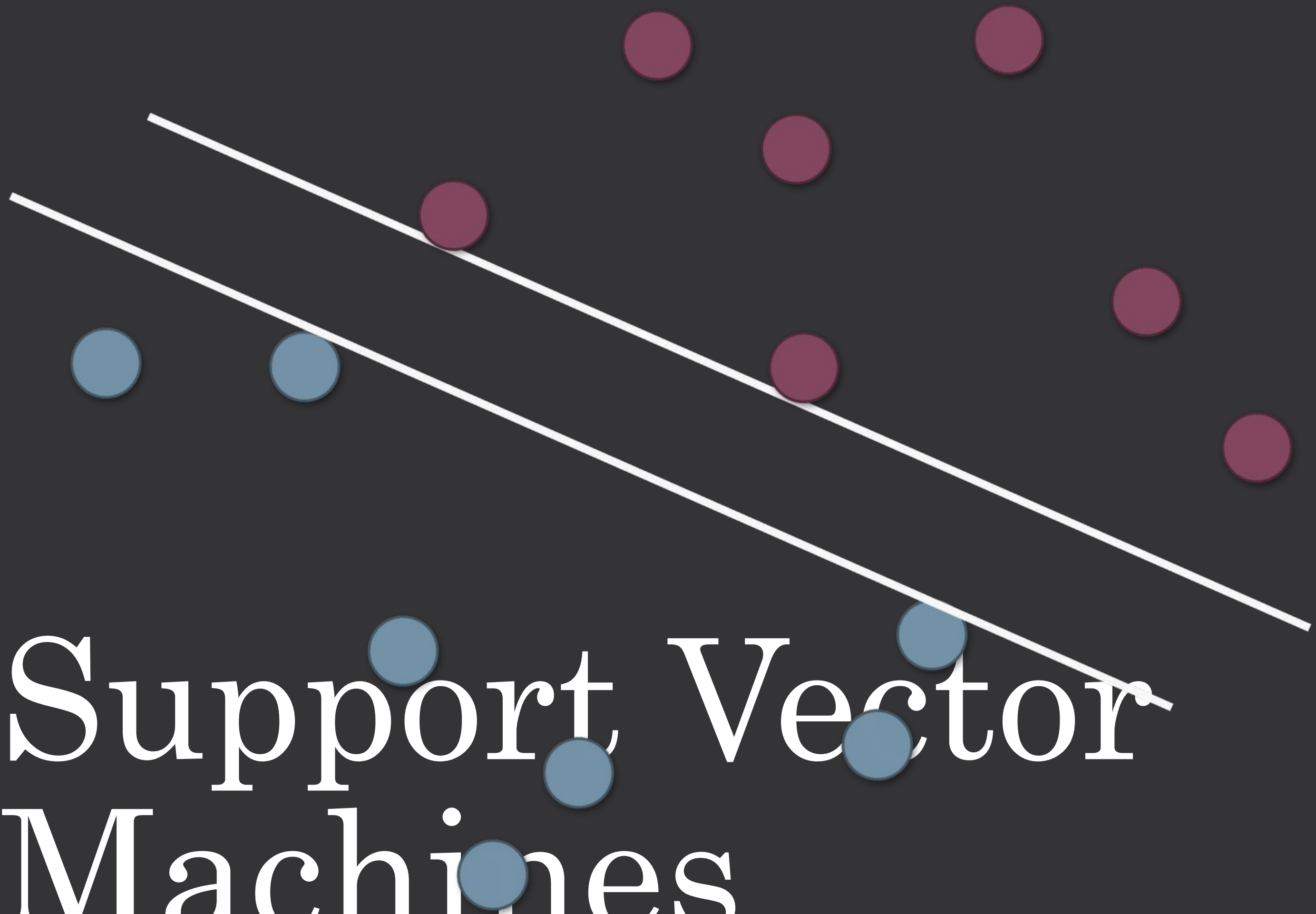
What about the independence assumption?

AGGGCCTAGCATT gets decomposed into

AGGG, GGGC, GGCC, GCCT, …

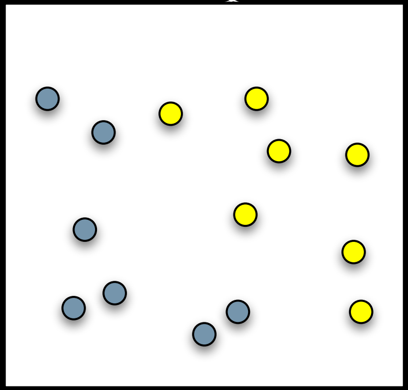
Which will be highly correlated!

Whatever.



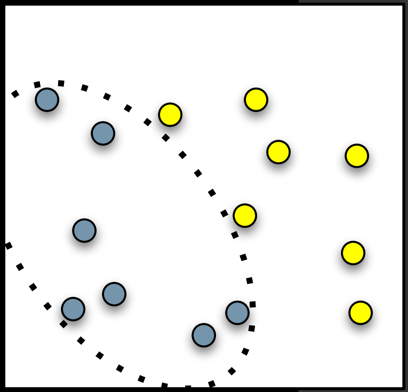
Support Vector Machines

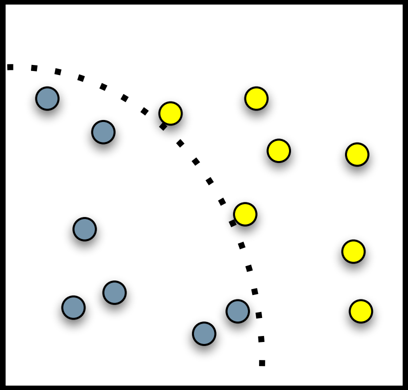
FEAR... THE DECISION SURFACE OF ARBITRARINESS



M
W
A

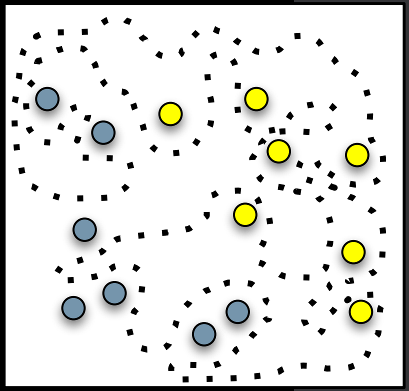
H
A



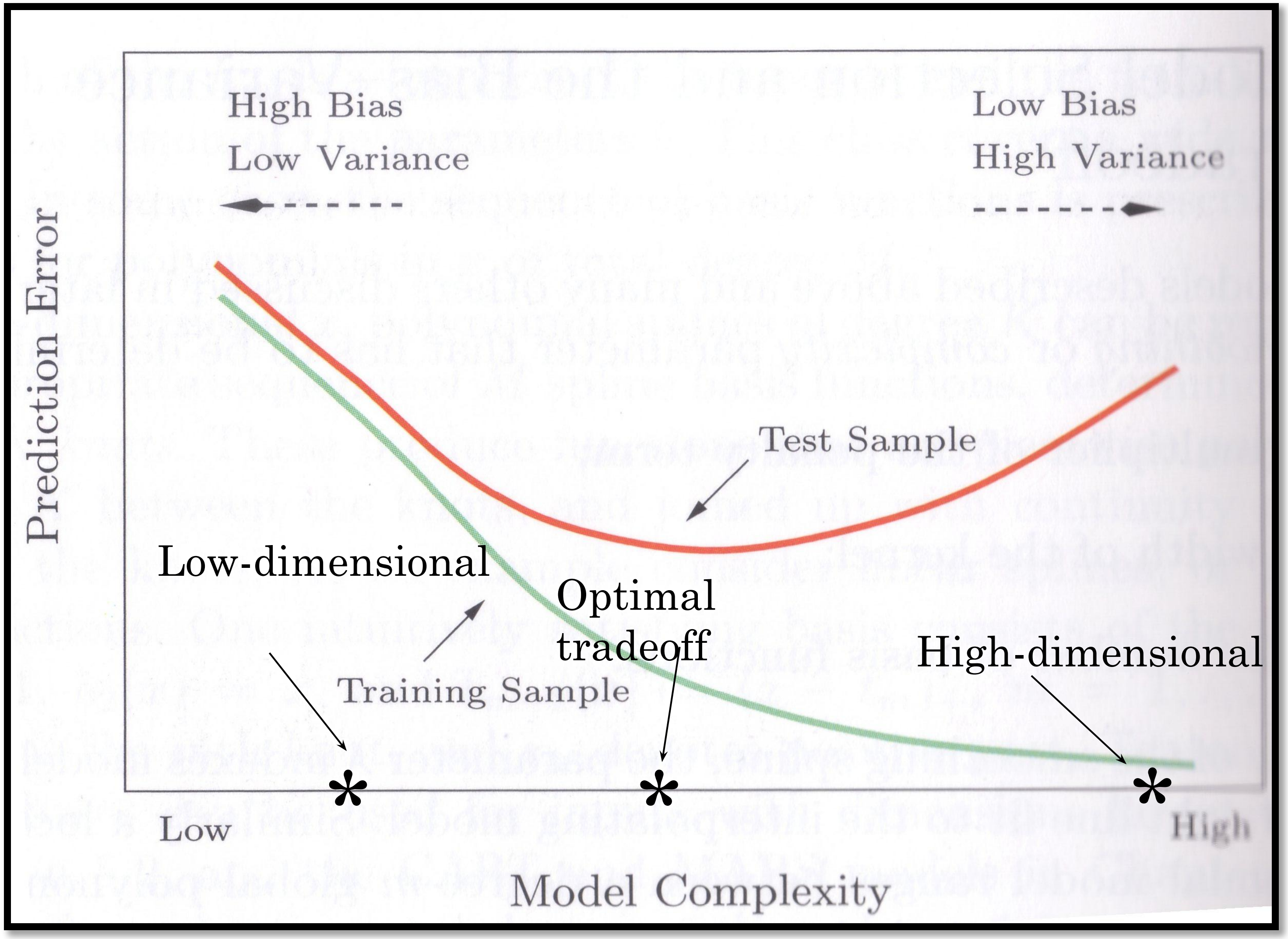


H
A

H
A!
!

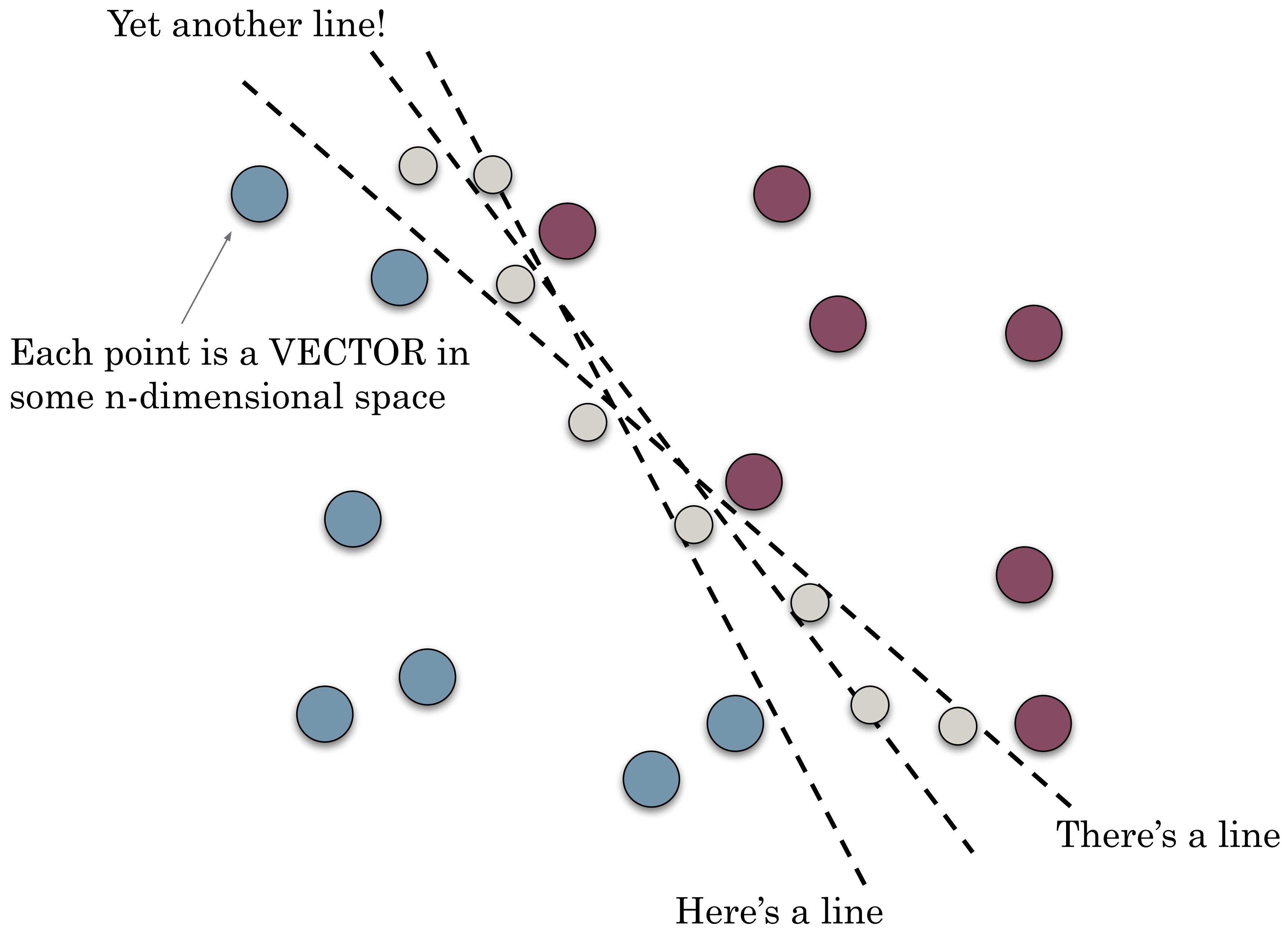


The bias-variance tradeoff

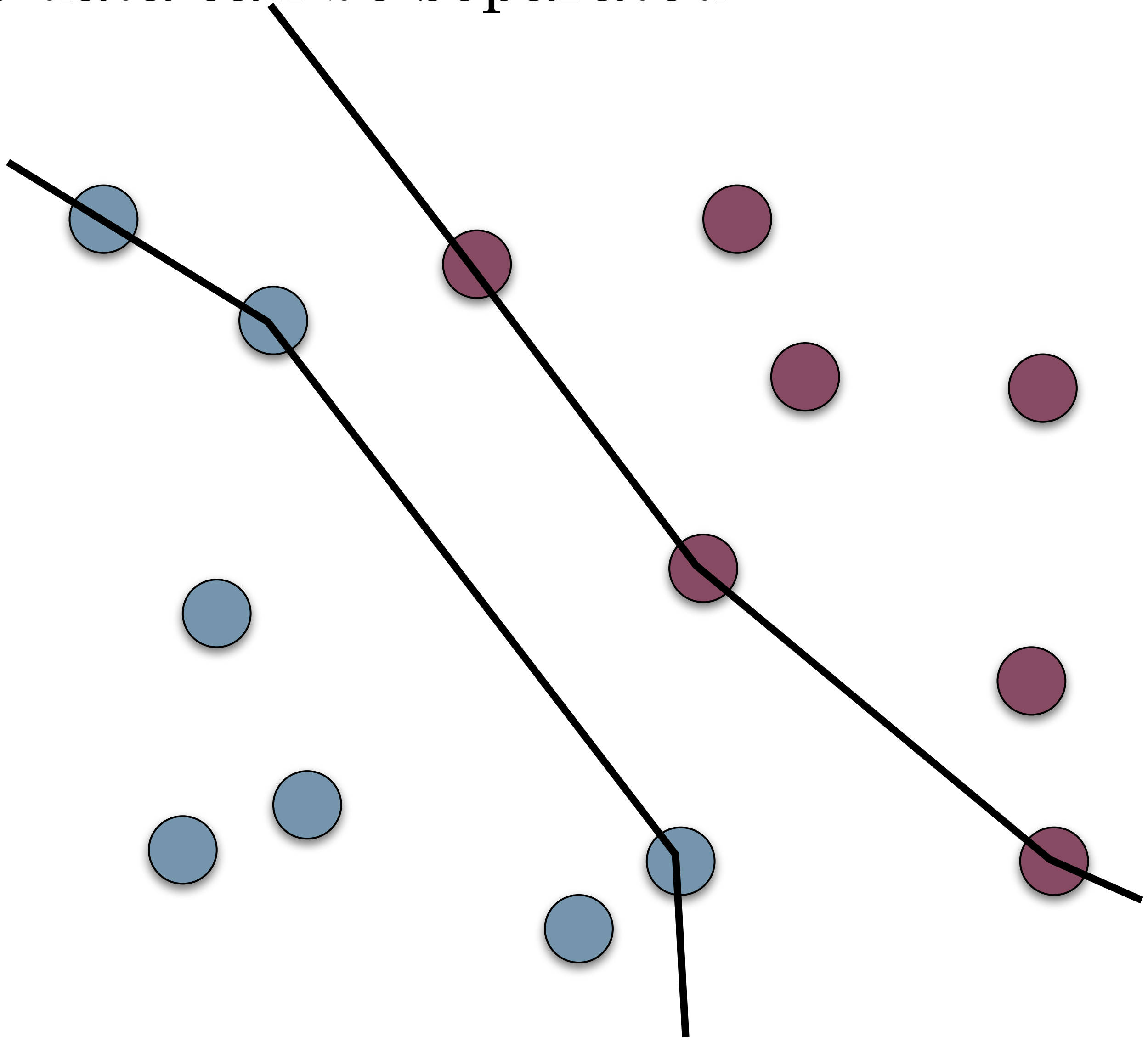


Hastie, p.38

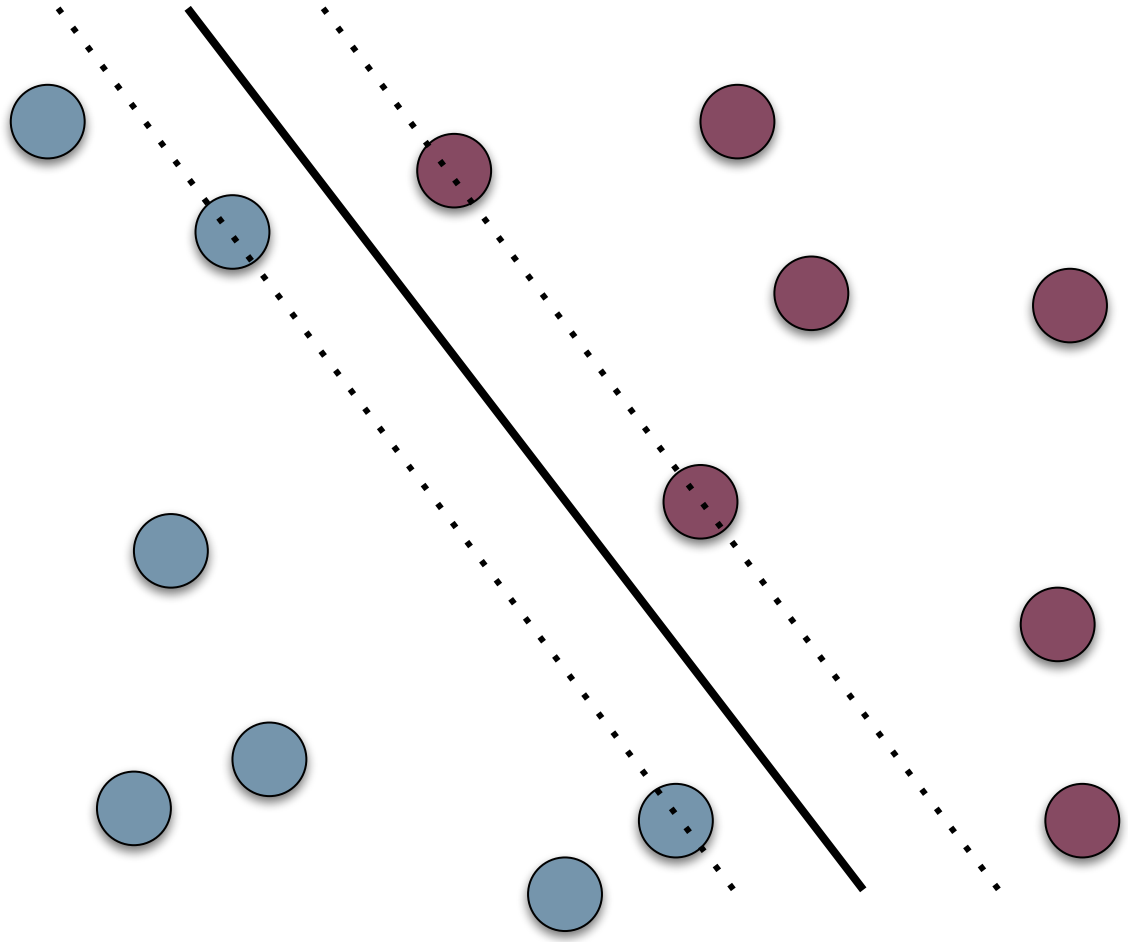
A linearly separable problem



The CONVEX HULLS determine how the data can be separated

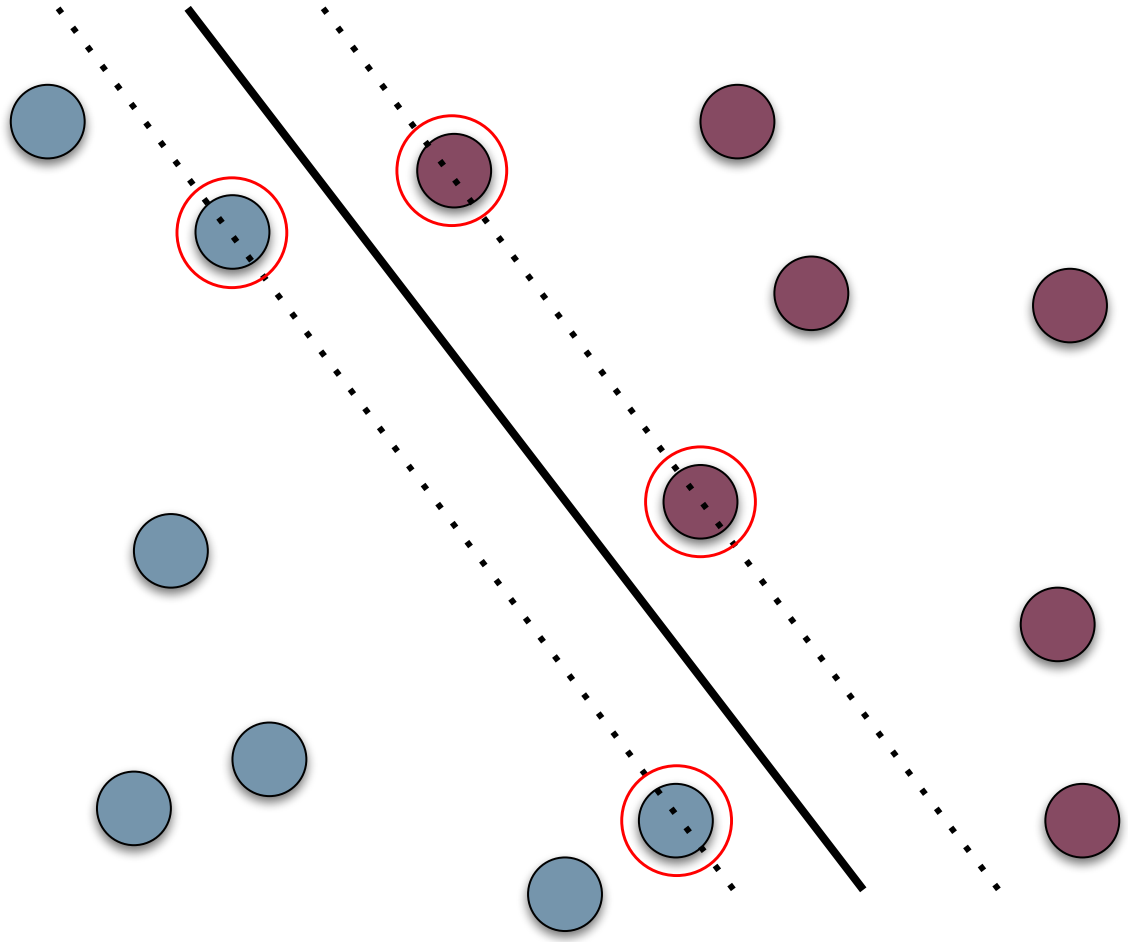


The CONVEX HULLS can define a maximum margin line (plane, hyperplane)



The **maximum margin hyperplane** separating two groups provides the optimal tradeoff between training set accuracy and function complexity

The SUPPORT VECTORS are the only points needed to define the decision boundary



The **support vector machine** aims to find the maximum margin hyperplane and its corresponding support vectors

Optimizing an SVM: **quadratic programming**

Maximize the margin, given the constraints that each class instance must lie on the "correct" side of the margin

This leads to a weighting of vectors – most vector weights will be zero (i.e., not support vectors)

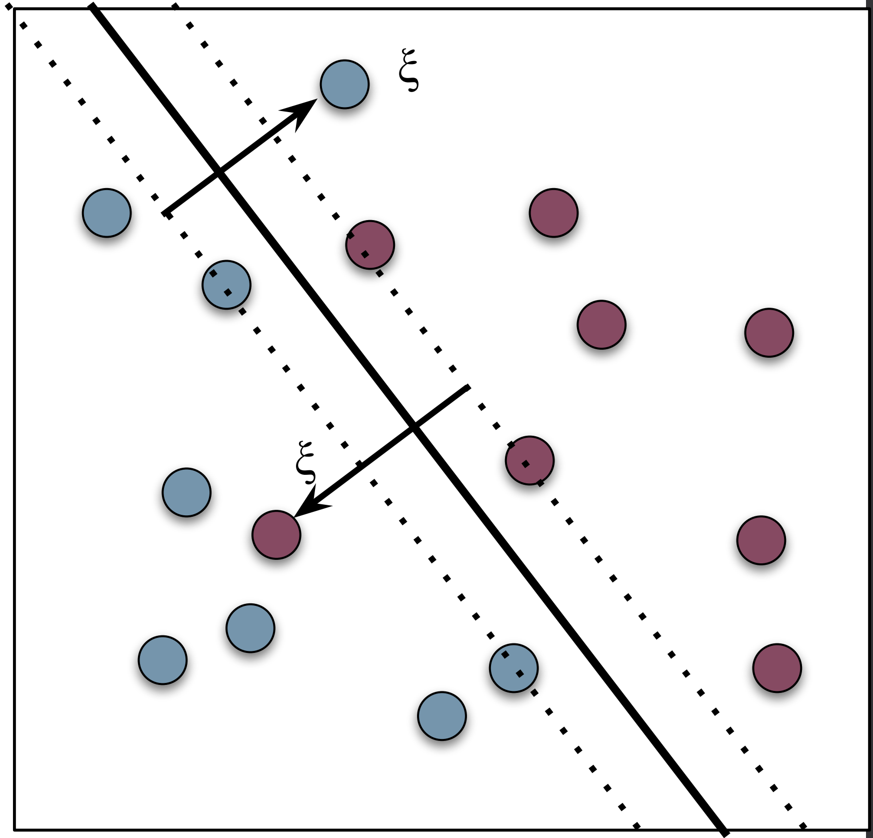
NOT ALL PROBLEMS ARE
LINEARLY SEPARABLE
imgflip.com

Maximum Margin with Errors

If a training set is not linearly separable given a function class of some complexity, we can introduce a bounded *error* term to tolerate misclassification. SOFT MARGINS

What is the appropriate value for the bound?

We can use **cross-validation** to find out



Getting the most from your linear classifier

- The SVM algorithm is based solely on dot products of vectors

Standard formulation:

$$\langle x_1, x_2 \rangle = \sum_{i=1}^{n} x_{1_i} \cdot x_{2_i}$$

Kernel trick: Substitute any positive semi-definite function *$K(x_1,x_2)$* for the dot product

Positive semi-definite matrix: all eigenvalues ≥ 0

Generic Kernels

Polynomial:

$$k(x_i, x_j) = (x_i \cdot x_j + 1)^d$$

Needs to be optimized!

Radial Basis:

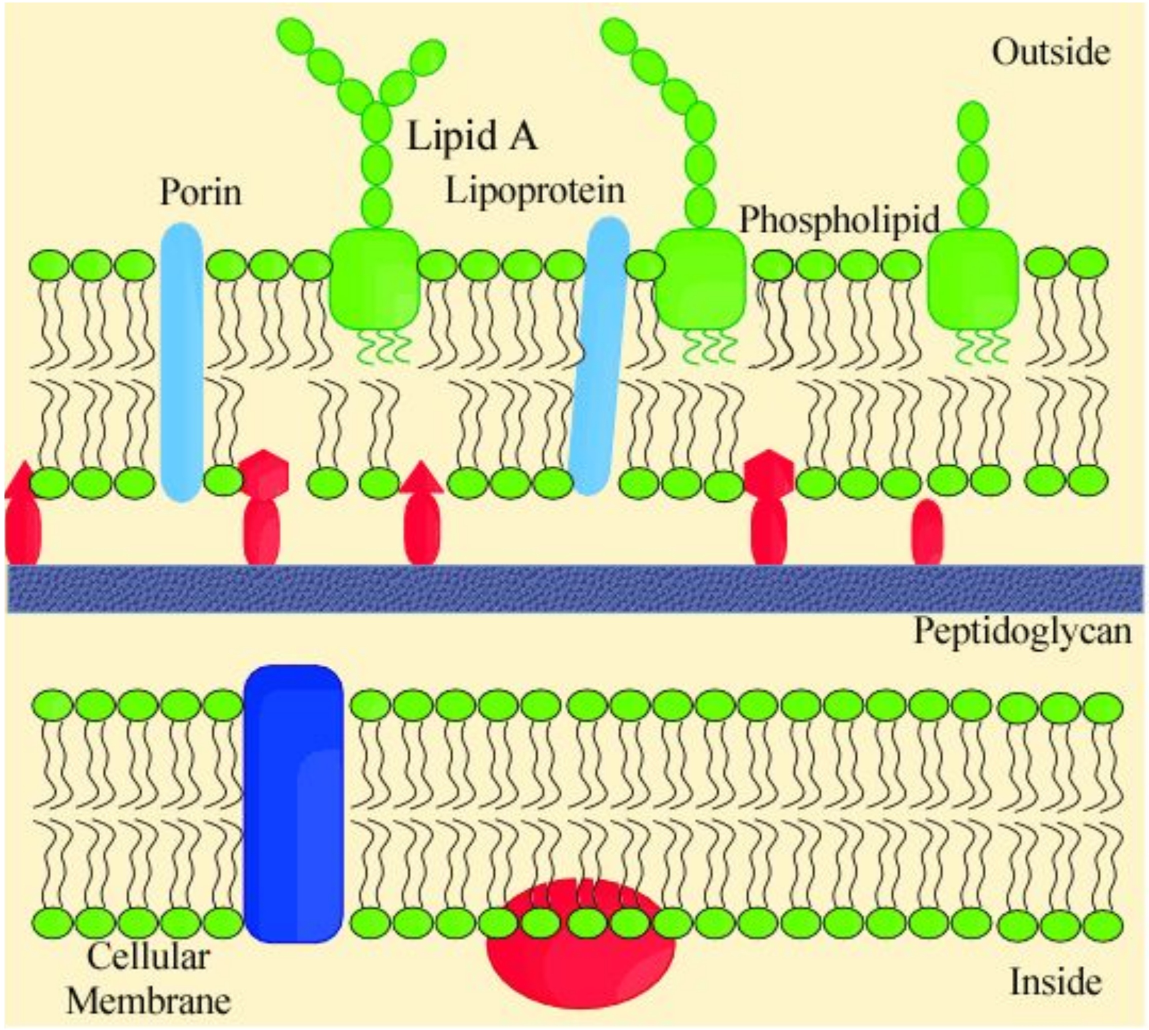
$$k(x_i, x_j) = \exp\left(-\gamma \left\| x_i - x_j \right\|^2\right)$$

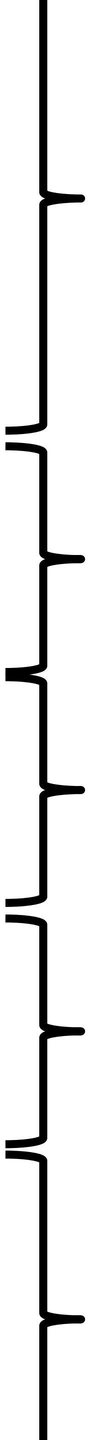
Sigmoid:

$$k(x_i, x_j) = \tanh(\kappa x_i \cdot x_j + c)$$

The classifier only ‘sees’ the resulting combinations of input features, and is still trying to optimize a **linear** solution

Generic kernels are neat, but the use of custom kernels allows you to use **domain-specific** knowledge to build the classifier


Outside
Lipid A
Porin
Lipoprotein
Phospholipid
Peptidoglycan
Cellular Membrane
Inside


Secreted
Outer membrane
Periplasmic space
Inner membrane
Cytoplasm

Challenge

Predict targeting of proteins
based on their primary structure
(= amino acid sequence)

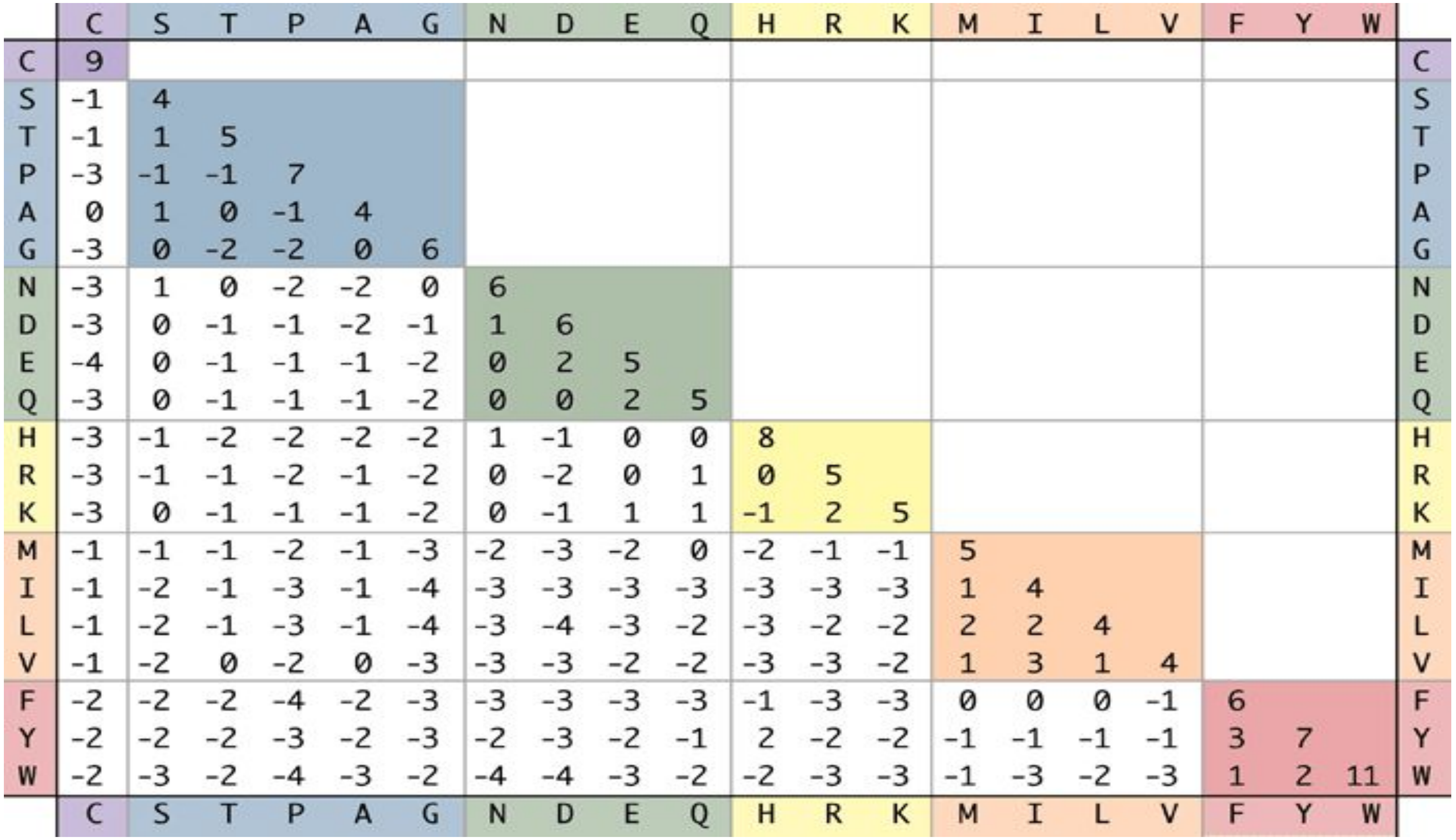
Sequence Kernels

The dot product is a way to capture the similarity between sequence vectors

Lots of ways to build kernel functions!

Substitution kernel

BLOSUM62 matrix for scoring residue similarity



	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	
C	9																				C
S	-1	4																			S
T	-1	1	5																		T
P	-3	-1	-1	7																	P
A	0	1	0	-1	4																A
G	-3	0	-2	-2	0	6															G
N	-3	1	0	-2	-2	0	6														N
D	-3	0	-1	-1	-2	-1	1	6													D
E	-4	0	-1	-1	-1	-2	0	2	5												E
Q	-3	0	-1	-1	-1	-2	0	0	2	5											Q
H	-3	-1	-2	-2	-2	-2	1	-1	0	0	8										H
R	-3	-1	-1	-2	-1	-2	0	-2	0	1	0	5									R
K	-3	0	-1	-1	-1	-2	0	-1	1	1	-1	2	5								K
M	-1	-1	-1	-2	-1	-3	-2	-3	-2	0	-2	-1	-1	5							M
I	-1	-2	-1	-3	-1	-4	-3	-3	-3	-3	-3	-3	-3	1	4						I
L	-1	-2	-1	-3	-1	-4	-3	-4	-3	-2	-3	-2	-2	2	2	4					L
V	-1	-2	0	-2	0	-3	-3	-3	-2	-2	-3	-3	-2	1	3	1	4				V
F	-2	-2	-2	-4	-2	-3	-3	-3	-3	-3	-1	-3	-3	0	0	0	-1	6			F
Y	-2	-2	-2	-3	-2	-3	-2	-3	-2	-1	2	-2	-2	-1	-1	-1	-1	3	7		Y
W	-2	-3	-2	-4	-3	-2	-4	-4	-3	-2	-2	-3	-3	-1	-3	-2	-3	1	2	11	W
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	

carverlab.org/testing/epp.sh

Local alignment (LA) kernel

K(S1,S2) = Alignment score of S1 with S2

```
. . .   NQILAL   . . .
        E-IGAL
```

Alignment score: Add matches, subtract gap penalties (details to come later)

$s(\mathrm{N,E}) - \boldsymbol{g} + s(\mathrm{I,I}) + s(\mathrm{L,G}) + s(\mathrm{A,A}) + s(\mathrm{L,L})$

Interesting questions

- Can SVMs trained using <u>different kernels</u> be combined to yield more accurate predictions?
- Are there proteins that are particularly hard to classify?
- What information can be used to improve the kernel?

SVM variants

- Support vector regression

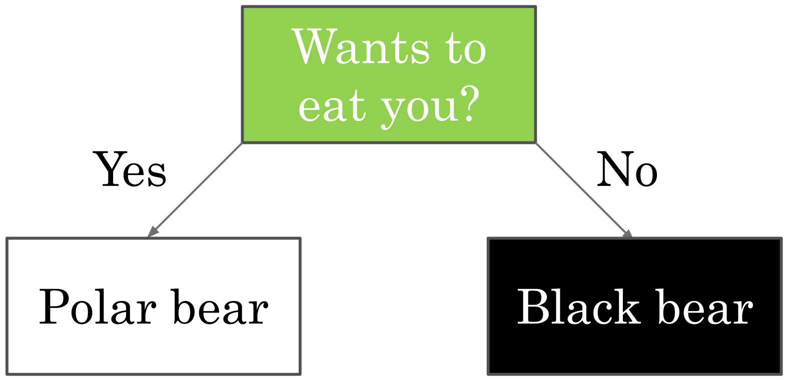
- Multi-class training

- Variations on error weighting

Random Forests

Decision trees

- Classify two or more groups by creating *decision nodes* based on specific criteria



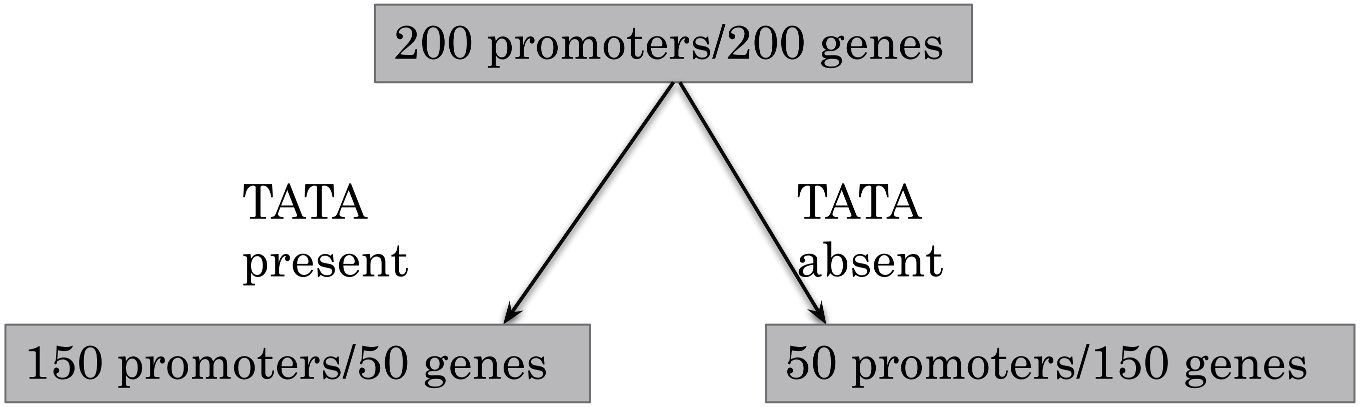
Training

```
Decision Node (cases c consisting of variables $x_{ic}$):

If (stopping criterion not reached)
    1. Find variable $x_i$ and threshold $t$ such that
       optimal separation is achieved between cases
       with different labels
    2. Decision Node ($c_{xi} \le t$)
    3. Decision Node ($c_{xi} > t$)
```

Optimal separation

Minimize node ‘impurity’



Misclassification error: % of cases not belonging to majority class

Gini index: $\sum_c (\Pr(c)) \times (1 - \Pr(c))$

Cross-entropy: $-\sum_c (\Pr(c)) \times \log \Pr(c)$

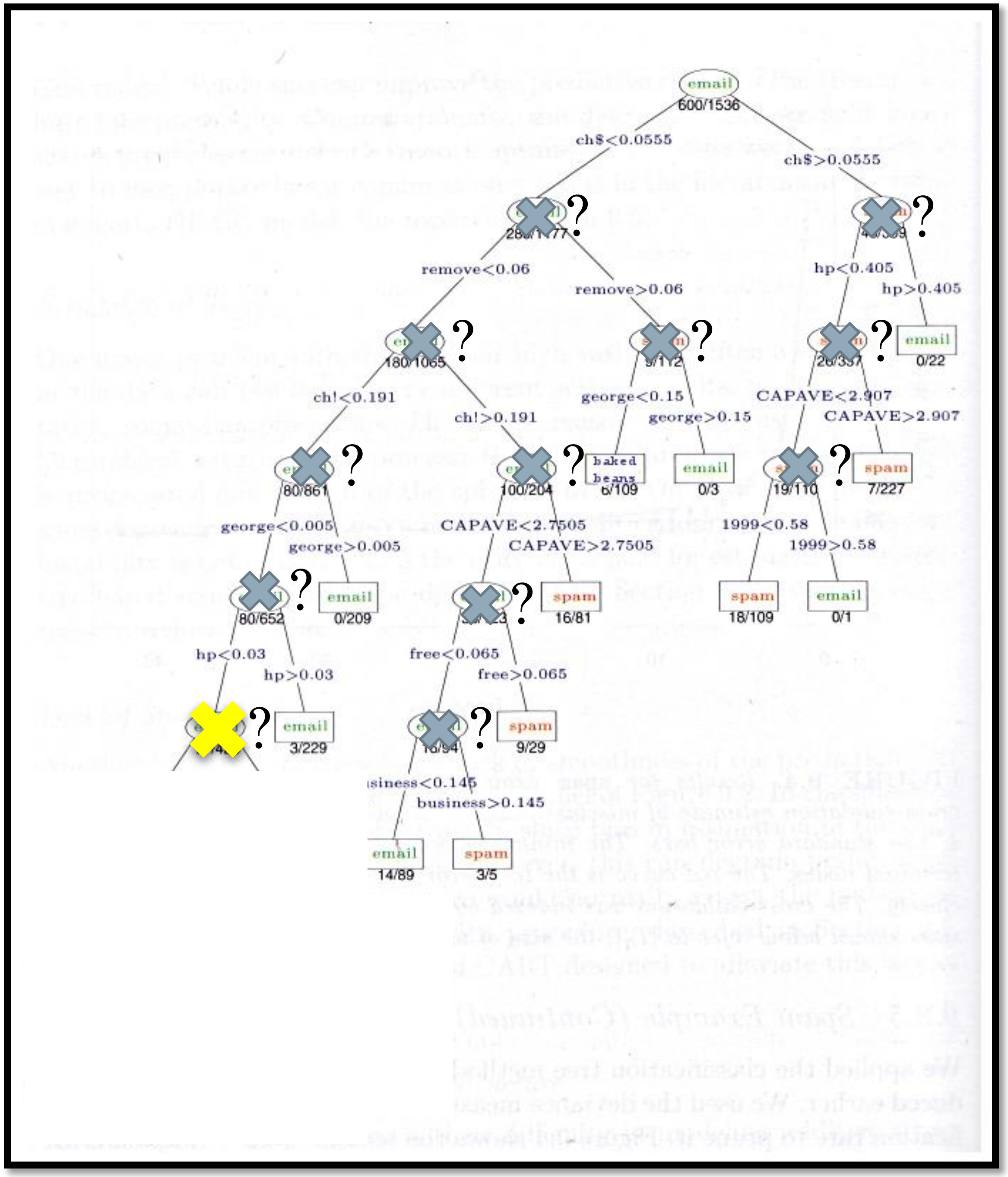
A greedy approach is most common

Is this guaranteed to find the best solution?

~~Yes~~ Of course not!!!

Stopping criteria

- There is no need to subdivide a pure class
- Other criteria (such as minimum number of classes at a node) might be used as well
- We can use *pruning* strategies to roll back a tree and achieve an optimal balance between size and separation (**bias** vs. **variance**)



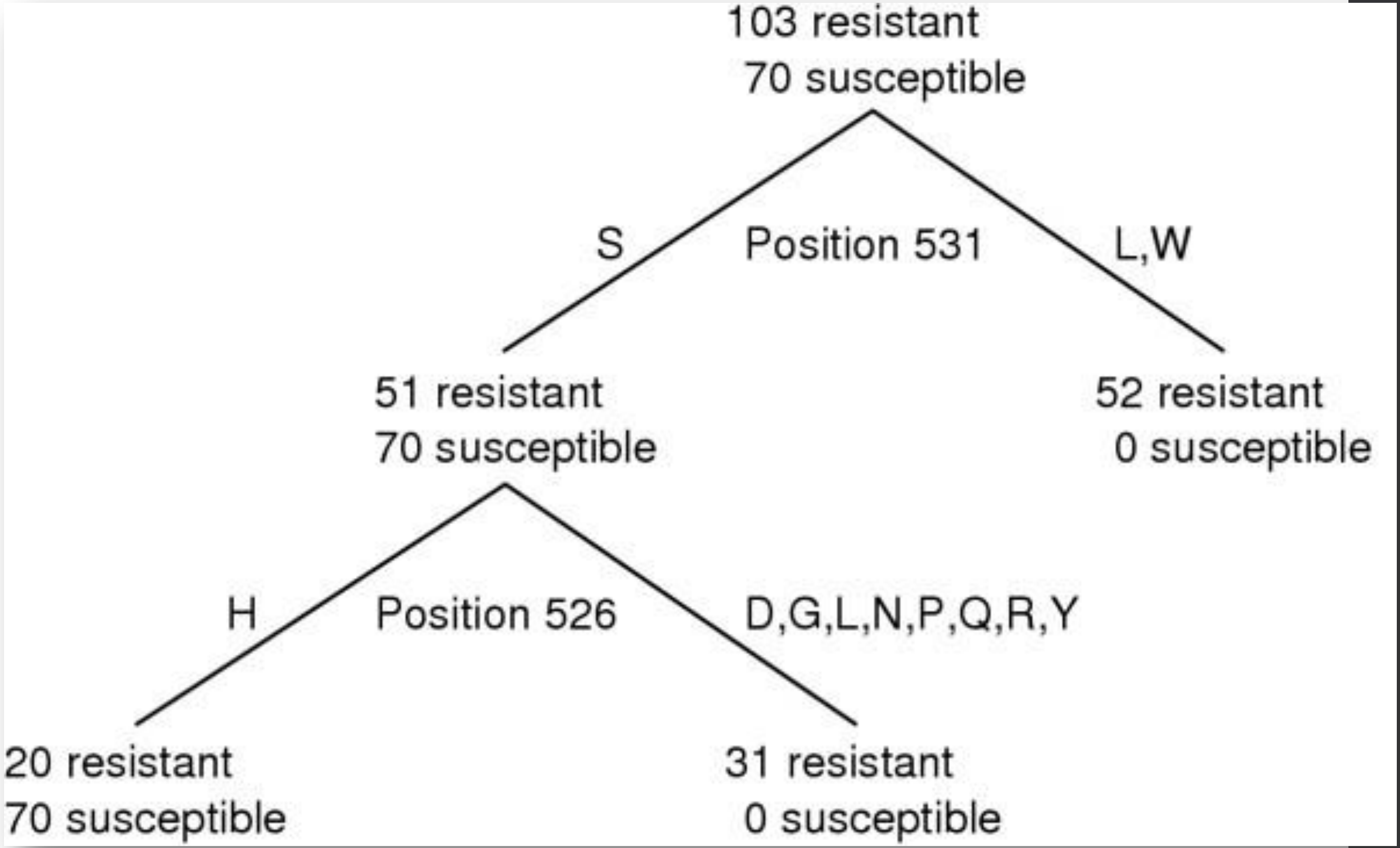
Weakest link pruning:

Find the subtree T_α that minimizes a cost function that balances accuracy and complexity

Rifampin sensitivity of bacterial RNA polymerase

- Rifampin interferes with transcription in *Mycobacterium tuberculosis*
- However, mutations can arise in certain parts of the *rpoB* gene that lead to rifampin resistance
- Can we classify *rpoB* variants based on the amino acids encoded by the gene?

Cummings *et al. BMC Bioinformatics* 2004 **5**:137

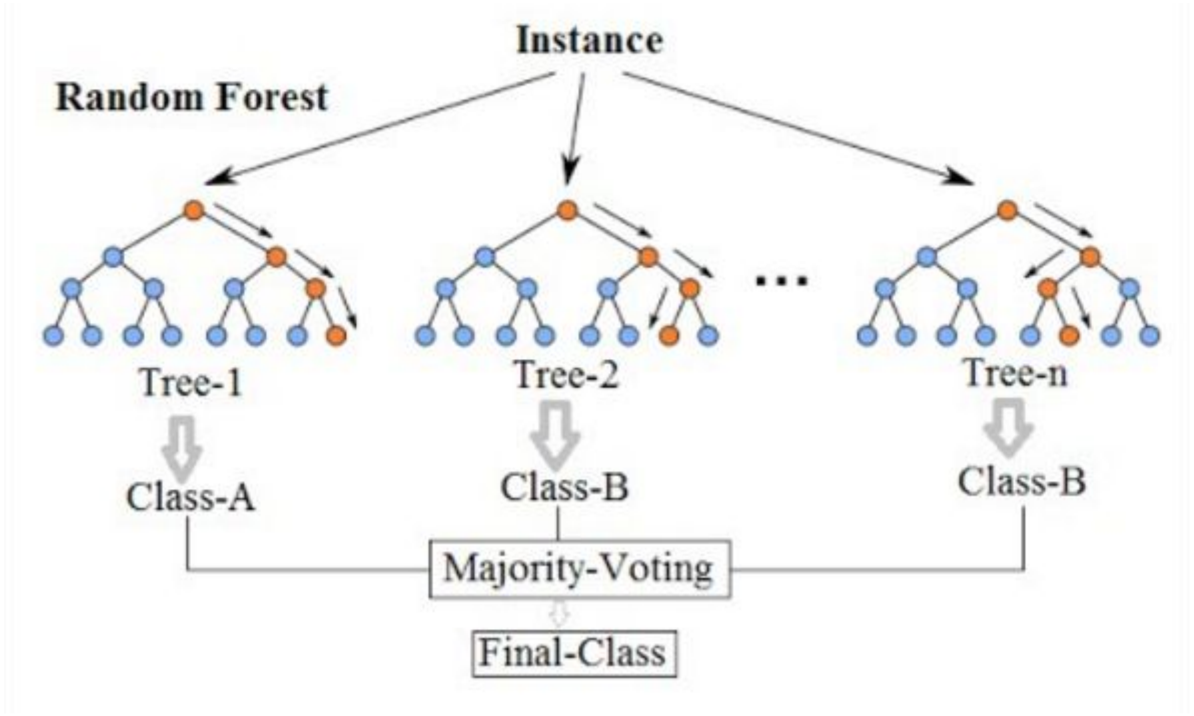


88.4% correctly classified (10-fold cross-validation)
Many nearby polymorphic sites (e.g. 511, 512, 515, 521 and 529) **rejected** as potentially good classifiers

That's great, but…

- Decision trees are **very** susceptible to overfitting during training
- Decisions are based on the training set and "nearly neutral" splitting decisions cannot be revisited
- So….?

Random forests:
why use only one tree when you can use many?



These trees must not all make the exact same predictions!

RFs part 1: trees, trees, trees!

- Each tree is trained on a randomly regenerated sample of the dataset, *with replacement*

Training Dataset: $\{T_1, T_2, T_3, T_4, \ldots, T_n\}$

Bootstrapped dataset 1: $\{T_1, T_2, T_3, T_2, \ldots, T_n\}$

Bootstrapped dataset 2: $\{T_3, T_1, T_3, T_4, \ldots, T_n\}$

Bootstrapped dataset 3: $\{T_4, T_4, T_1, T_1, \ldots, T_n\}$

Each time we *overrepresent* some cases, and *eliminate* others = boostrap aggregation (bagging)

RFs part 2: playing with features

- At each node in each tree, select only a random *subset* of features to draw from for decision making

- ???

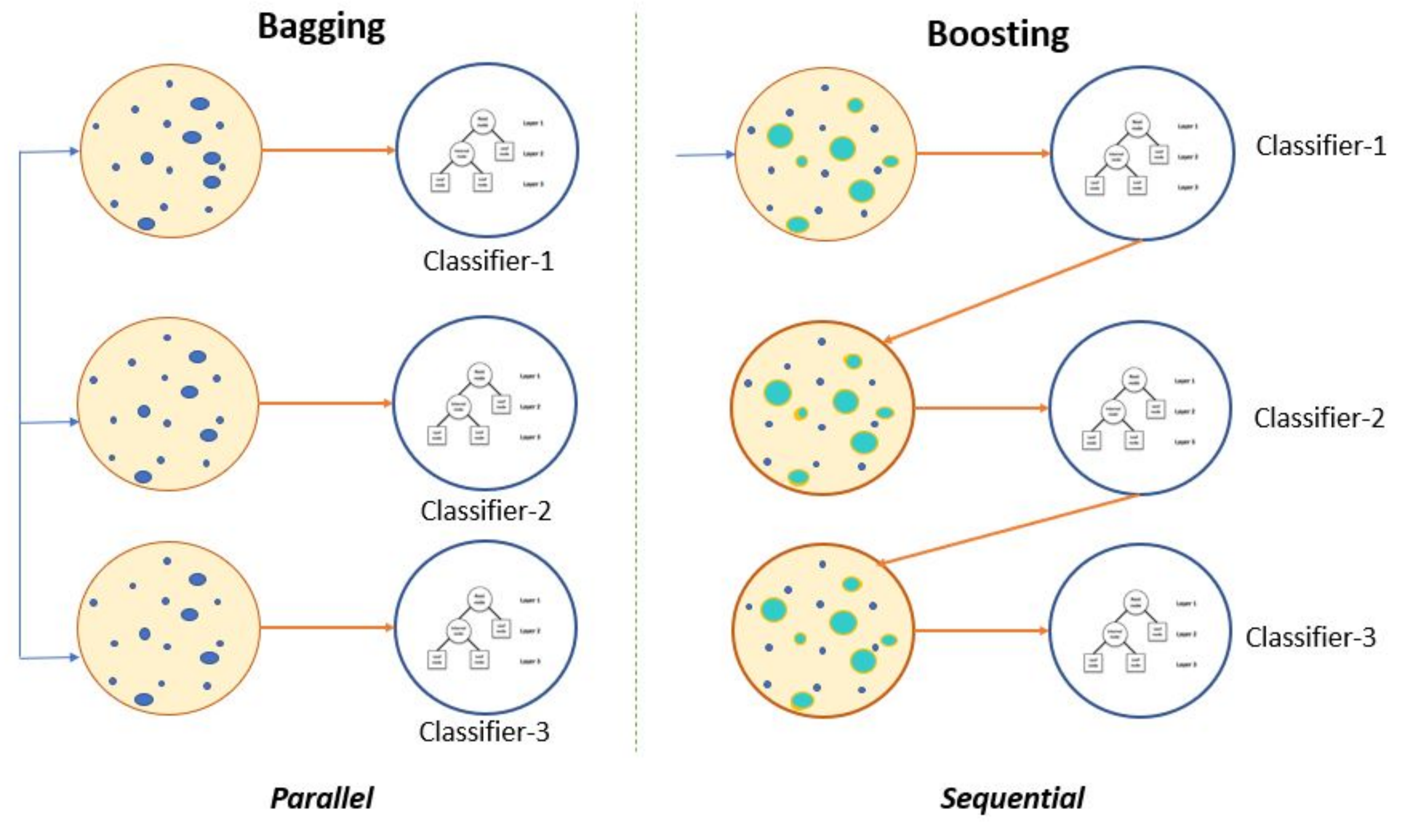
- The big idea: if you use the same features every time, **you'll get the same tree many times**. Random subset selection addresses this.

RFs: the point

- Individual trees can overfit, but taking many trees (averaging or voting) can smooth out the consequences of overfitting

- RFs can be very accurate, even if many of the individual trees aren’t very good!

More than 1 way to ensemble



https://www.pluralsight.com/guides/ensemble-methods:-bagging-versus-boosting

Summary

1. Start simple and carefully tune to establish baselines
2. Exploit your knowledge of the data with SVM kernels
3. Make use of ensemble techniques to get effective models with relatively little work!