BABELOMICS

Massive Data Analysis
Class prediction with Babelomics

Valencia, March 2011

Ignacio Medina (*Nacho*)
imedina@cipf.es
http://bioinfo.cipf.es/imedina
Bioinformatics and Genomics Department
Centro de Investigación Príncipe Felipe (CIPF)
(Valencia, Spain)

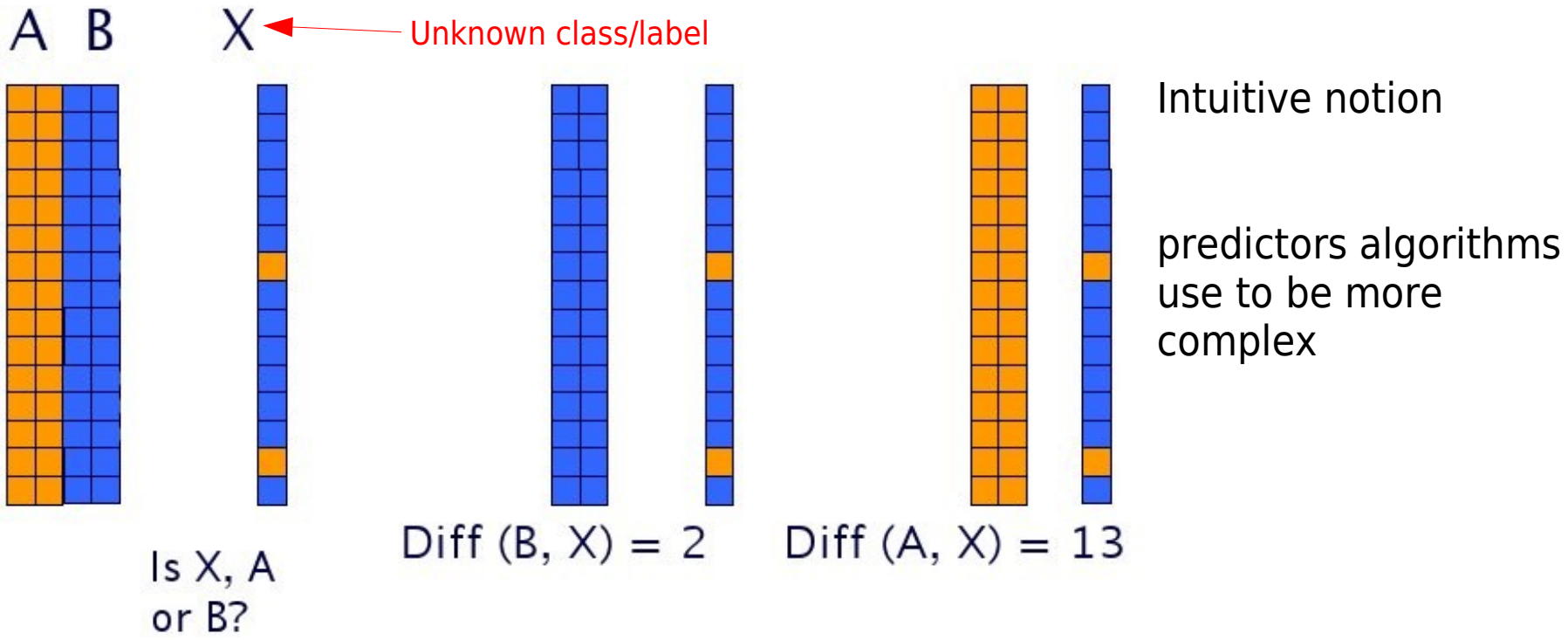
INB

PRINCIPE FELIPE
CENTRO DE INVESTIGACION

ciberer

Introduction
Definition

- Set of algorithms/methods that are going to allow the computer to ***learn*** a specific ***labelled*** problem and then be able to ***predict*** or ***classify*** new ***unlabelled*** samples (***Supervised learning***)
- Predictors/classifiers are a subset of methods of the *Artificial Intelligence* area
- Predictors need to be ***trained***



Introduction
Brief history background

- ***1956***, ***John McCarthy*** coined the term "*Artificial Intelligence*" and defined it as "*the science and engineering of making intelligent machines*."
- ***60-70's***, the mathematical background of some of these algorithms/methods was developed
- ***70-80's***, with the apparition of computers first predictors were developed for many different areas:
  - *handwriting recognition*
  - *weather prediction*
  - *face recognition*
  - *speech recognition*
  - ...

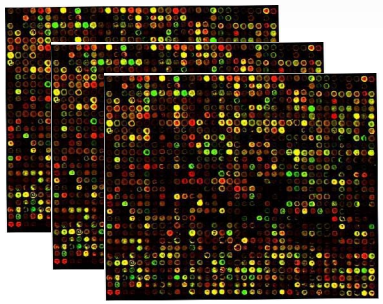
Introduction
Brief history background

- ***90's***, predictors begun to be used in Bioinformatics and Computational Biology:
  - *gene prediction:* sequence based gene annotation
  - *genome annotation:* sequence based TFBS, exons, ... annotation
  - *protein structure prediction*
  - ...
- ***in late 90's,*** DNA microarray technology was developed:
  - sample classification and class prediction was the aim of many gene expression studies, i.e.: *diagnose*
  - microarray data could be classified into a discrete ***classes***, i.e.: *diseases state, different tumor types, outcome of a treatment, ...*
  - each microarray belong to one and just one of the classes

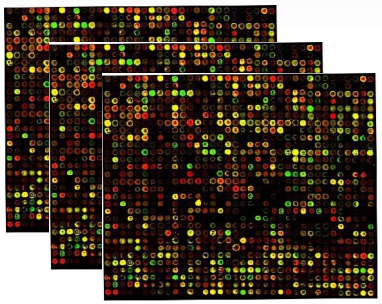
Introduction
Brief history background

- ***in early 2000***, two questions arose:
  - could biological samples be classified according to gene expression?
  - and, could we use computers to help us classifying these microarrays data experiments?

Controls



Cases



- ***2002***, appears the first paper applying predictors method to DNA microarray data, *van't Veer et al. (Nature)*
- since that moment hundreds of papers, applications and new methods have been developed

Introduction

What a predictor does?

genes

class label

known class data (*Babelomics format*)

#VARIABLE *tumor* CATEGORICAL{c1,c2,c3} VALUES{c1,c1,c2,c2,c3,c3} DESCRIPTION{}

#NAMES	GSM26878	GSM26883	GSM26886	GSM26887	GSM26903	GSM26910
1007_s_at	11.08578155	11.04457022	11.02479206	11.00837346	11.04430518	11.01921026
1053_at	7.787503325	8.010263804	7.872064511	7.711140759	7.703846348	7.509845931
117_at	7.487539205	7.526590226	7.442468793	7.394731634	7.450764725	7.558967177
121_at	9.589979282	9.516503297	9.610811352	9.282059896	8.323068371	8.664237594
1255_g_at	5.000099854	5.127166256	4.952998877	4.881038876	4.948734762	5.087888404
1294_at	8.358097049	8.403219181	8.255863646	7.947778797	8.328705461	8.230633848
1316_at	7.187245349	6.652952654	6.445444909	6.463659189	6.399722565	6.404821127
1320_at	5.645994428	5.765206267	5.772052661	5.609287091	5.621417391	5.723352308
1405_i_at	7.138444163	7.490198393	7.382302176	7.379200666	7.541671446	6.493521779
1431_at	4.697298725	4.722480562	4.795825627	4.703361751	4.701914661	4.904298823
1438_at	7.430761532	8.112797873	7.578819384	7.699611607	7.496504531	7.776384116
1487_at	7.646126117	7.544048497	8.754540699	8.476873549	9.084035203	9.028724488
1494_f_at	7.498031252	7.679595836	7.662561072	7.201093115	7.426192546	7.669669586
1598_g_at	10.31770877	10.92530764	10.50092321	9.630201704	10.23473332	10.49766918
160020_at	8.529411037	8.738065073	8.617216353	8.445386532	8.425365655	8.76023381
1729_at	9.607320487	8.171988017	8.73040537	8.978602862	9.156752025	8.033237589
1773_at	6.216319215	6.441555855	6.165785507	6.325464779	6.121753223	6.229420354
177_at	6.535525364	6.453887146	6.519400663	6.333366799	6.385077422	6.407541976

arrays

unknown class data

1007_s_at	11.28578155
1053_at	7.787503325
117_at	7.487539205
121_at	9.489979282
1255_g_at	5.000099854
1294_at	8.358097049
1316_at	7.187245349
1320_at	5.645994428
1405_i_at	7.138444163
1431_at	4.697298725
1438_at	7.430761532
1487_at	8.646126117
1494_f_at	7.498031252
1598_g_at	10.31770877
160020_at	8.529411037
1729_at	9.107320487
1773_at	6.216319215
177_at	6.535525364

which is the class label?

Introduction

How does a predictor work?

Known class label gene expression data from array experiment

	class 1	class 1	class 2	class 2
1007_s_at	11.28578155	11.04457022	11.02479206	11.00837346
1053_at	7.787503325	8.010263804	7.872064511	7.711140759
117_at	7.487539205	7.526590226	7.442468793	7.394731634
121_at	9.589979282	9.516503297	9.610811352	9.282059896
1255_g_at	5.000099854	5.127166256	4.952998877	4.881038876
1294_at	8.358097049	8.403219181	8.255863646	7.947778797
1316_at	7.187245349	6.652952654	6.445444909	6.463659189

Training

Stage 1

predictors

SVM
KNN
PAM
DLDA
SOM

Many different algorithms available

	¿ ?
1007_s_at	11.28578155
1053_at	7.787503325
117_at	7.487539205
121_at	9.589979282
1255_g_at	5.000099854
1294_at	8.358097049
1316_at	7.187245349

Unknown class label gene expression data

Prediction

Stage 2

- **CLASS assigned to the sample**
- **Confidence of the prediction (*error estimation*)**

Feature selection

Finding genes that discriminate classes

#VARIABLE *tumor* CATEGORICAL{c1,c2,c3} VALUES{c1,c1,c2,c2,c3,c3}

#NAMES	GSM26878	GSM26883	GSM26886	GSM26887	GSM26903	GSM26910
1007_s_at	11.08578155	11.04457022	11.02479206	11.00837346	11.04430518	11.01921026
1053_at	7.787503325	8.010263804	7.872064511	7.711140759	7.703846348	7.509845931
117_at	7.487539205	7.526590226	7.442468793	7.394731634	7.450764725	7.558967177
121_at	9.589979282	9.516503297	9.610811352	9.282059896	8.323068371	8.664237594
1255_g_at	5.000099854	5.127166256	4.952998877	4.881038876	4.948734762	5.087888404
1294_at	8.358097049	8.403219181	8.255863646	7.947778797	8.328705461	8.230633848
1316_at	7.187245349	6.652952654	6.445444909	6.463659189	6.399722565	6.404821127
1320_at	5.645994428	5.765206267	5.772052661	5.609287091	5.621417391	5.723352308
1405_i_at	7.138444163	7.490198393	7.382302176	7.379200666	7.541671446	6.493521779
1431_at	4.697298725	4.722480562	4.795825627	4.703361751	4.701914661	4.904298823
1438_at	7.430761532	8.112797873	7.578819384	7.699611607	7.496504531	7.776384116
1487_at	7.646126117	7.544048497	8.754540699	8.476873549	9.084035203	9.028724488
1494_f_at	7.498031252	7.679595836	7.662561072	7.201093115	7.426192546	7.669669586
1598_g_at	10.31770877	10.92530764	10.50092321	9.630201704	10.23473332	10.49766918
160020_at	8.529411037	8.738065073	8.617216353	8.445386532	8.425365655	8.76023381
1729_at	9.607320487	8.171988017	8.73040537	8.978602862	9.156752025	8.033237589
1773_at	6.216319215	6.441555855	6.165785507	6.325464779	6.121753223	6.229420354
177_at	6.535525364	6.453887146	6.519400663	6.333366799	6.385077422	6.407541976

Genes that do not change do not bear any information

Genes that change randomly within classes are not useful for classifying

Feature selection

Finding some patterns in data

#VARIABLE *tumor* CATEGORICAL{c1,c2,c3} VALUES{c1,c1,c2,c2,c3,c3}

#NAMES	GSM26878	GSM26883	GSM26886	GSM26887	GSM26903	GSM26910
1007_s_at	11.08578155	11.04457022	11.02479206	11.00837346	11.04430518	11.01921026
1053_at	7.787503325	8.010263804	7.872064511	7.711140759	7.703846348	7.509845931
117_at	7.487539205	7.526590226	7.442468793	7.394731634	7.450764725	7.558967177
121_at	9.589979282	9.516503297	9.610811352	9.282059896	8.323068371	8.664237594
1255_g_at	5.000099854	5.127166256	4.952998877	4.881038876	4.948734762	5.087888404
1294_at	8.358097049	8.403219181	8.255863646	7.947778797	8.328705461	8.230633848
1316_at	7.187245349	6.652952654	6.445444909	6.463659189	6.399722565	6.404821127
1320_at	5.645994428	5.765206267	5.772052661	5.609287091	5.621417391	5.723352308
1405_i_at	7.138444163	7.490198393	7.382302176	7.379200666	7.541671446	6.493521779
1431_at	4.697298725	4.722480562	4.795825627	4.703361751	4.701914661	4.904298823
1438_at	7.430761532	8.112797873	7.578819384	7.699611607	7.496504531	7.776384116
1487_at	7.646126117	7.544048497	8.754540699	8.476873549	9.084035203	9.028724488
1494_f_at	7.498031252	7.679595836	7.662561072	7.201093115	7.426192546	7.669669586
1598_g_at	10.31770877	10.92530764	10.50092321	9.630201704	10.23473332	10.49766918
160020_at	8.529411037	8.738065073	8.617216353	8.445386532	8.425365655	8.76023381
1729_at	9.607320487	8.171988017	8.73040537	8.978602862	9.156752025	8.033237589
1773_at	6.216319215	6.441555855	6.165785507	6.325464779	6.121753223	6.229420354
177_at	6.535525364	6.453887146	6.519400663	6.333366799	6.385077422	6.407541976

unknown class data

1007_s_at	11.28578155
1053_at	7.787503325
117_at	7.487539205
121_at	9.489979282
1255_g_at	5.000099854
1294_at	8.358097049
1316_at	7.187245349
1320_at	5.645994428
1405_i_at	7.138444163
1431_at	4.697298725
1438_at	7.430761532
1487_at	8.646126117
1494_f_at	7.498031252
1598_g_at	10.31770877
160020_at	8.529411037
1729_at	9.107320487
1773_at	6.216319215
177_at	6.535525364

class 2?

Feature selection
How can we select discriminating genes?

- Fundamental to Machine Learning and statistics is ***feature selection***, which is also known as *variable selection*, *feature reduction*, *attribute selection* or *variable subset selection*, is the technique of **selecting a subset of relevant features** for building **robust learning models** (*Wikipedia*)
- Many algorithms in literature:
  - Correlation-based Feature Selection (CFS)
  - Principal Components Analysis (PCA)
  - Genetic algorithms
  - Greedy forward selection
  - Greedy backward elimination
  - ...

Feature selection
Correlation-based Feature Selection (CFS)

- Hypothesis:
  - *A good feature subset is one that contains **features** highly correlated with (predictive of) the class, yet uncorrelated with (not predictive of) each other*
- Properties:
  - It's simple, fast and automatic
  - Eliminates irrelevant and redundant data
  - In many cases improves the performance of learning algorithms

Feature selection
Principal Components Analysis (PCA)

- It's a mathematical procedure that transforms a number of possibly correlated variables into a smaller number of uncorrelated variables called *principal components*
- The *first principal component* accounts for as much of the *variability* in the data as possible
- Dimensionality reduction is accomplished by choosing enough *eigenvectors* to account for some percentage of the variance in the original data, default 0.95 (95%)
- Attribute noise can be filtered by transforming to the PC space, eliminating some of the *worst eigenvectors*, and then transforming back to the original space

Gene selection

How can we select discriminating genes?

#VARIABLE *tumor* CATEGORICAL{c1,c2,c3} VALUES{c1,c1,c2,c2,c3,c3}

#NAMES	GSM26878	GSM26883	GSM26886	GSM26887	GSM26903	GSM26910
1007_s_at	11.08578155	11.04457022	11.02479206	11.00837346	11.04430518	11.01921026
1053_at	7.787503325	8.010263804	7.872064511	7.711140759	7.703846348	7.509845931
117_at	7.487539205	7.526590226	7.442468793	7.394731634	7.450764725	7.558967177
121_at	9.589979282	9.516503297	9.610811352	9.282059896	8.323068371	8.664237594
1255_g_at	5.000099854	5.127166256	4.952998877	4.881038876	4.948734762	5.087888404
1294_at	8.358097049	8.403219181	8.255863646	7.947778797	8.328705461	8.230633848
1316_at	7.187245349	6.652952654	6.445444909	6.463659189	6.399722565	6.404821127
1320_at	5.645994428	5.765206267	5.772052661	5.609287091	5.621417391	5.723352308
1405_i_at	7.138444163	7.490198393	7.382302176	7.379200666	7.541671446	6.493521779
1431_at	4.697298725	4.722480562	4.795825627	4.703361751	4.701914661	4.904298823
1438_at	7.430761532	8.112797873	7.578819384	7.699611607	7.496504531	7.776384116
1487_at	7.646126117	7.544048497	8.754540699	8.476873549	9.084035203	9.028724488
1494_f_at	7.498031252	7.679595836	7.662561072	7.201093115	7.426192546	7.669669586
1598_g_at	10.31770877	10.92530764	10.50092321	9.630201704	10.23473332	10.49766918
160020_at	8.529411037	8.738065073	8.617216353	8.445386532	8.425365655	8.76023381
1729_at	9.607320487	8.171988017	8.73040537	8.978602862	9.156752025	8.033237589
1773_at	6.216319215	6.441555855	6.165785507	6.325464779	6.121753223	6.229420354
177_at	6.535525364	6.453887146	6.519400663	6.333366799	6.385077422	6.407541976

We can use statistical methods in order to select which genes to use

CFS
PCA

+
genes ranked (PCA only)
-

5 best genes

10 best genes

15 best genes

.
.
.

We can decide to use only the first genes in that ranking for bulding the predictor

Methods
How many predictors exist?

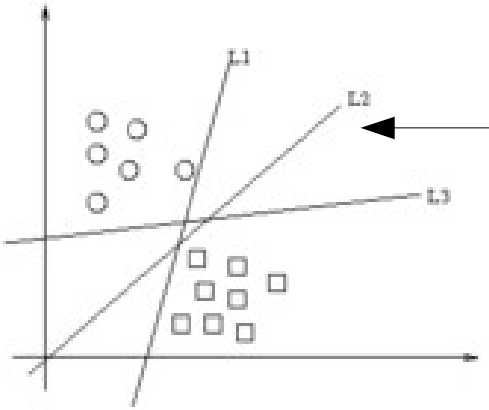
- There are many differents predictors based on many differents strategies:
  - Neural networks
  - Support Vector Machines (SVM) ⇦
  - K-Nearest neighbours (KNN) ⇦
  - Random Forest (RF)
  - PAM (Shrunken centroids)
  - Diagonal Linear Discriminants (DLDA)
  - Naive Bayes
  - ...

Not all of them work fine in all situations

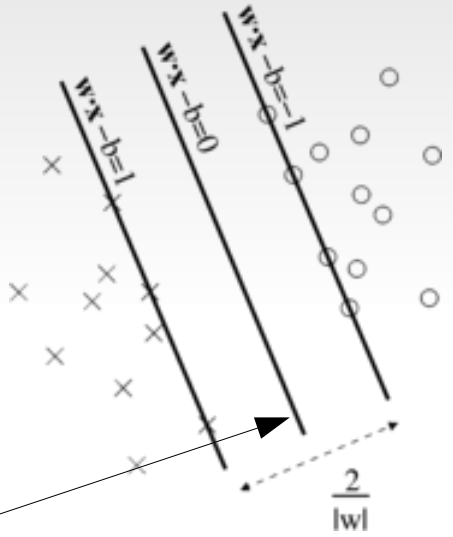
Methods

Support Vector Machine (SVM)

SVM is a linear classifier that is going to find a *line* or *hiperplane* to separate the data

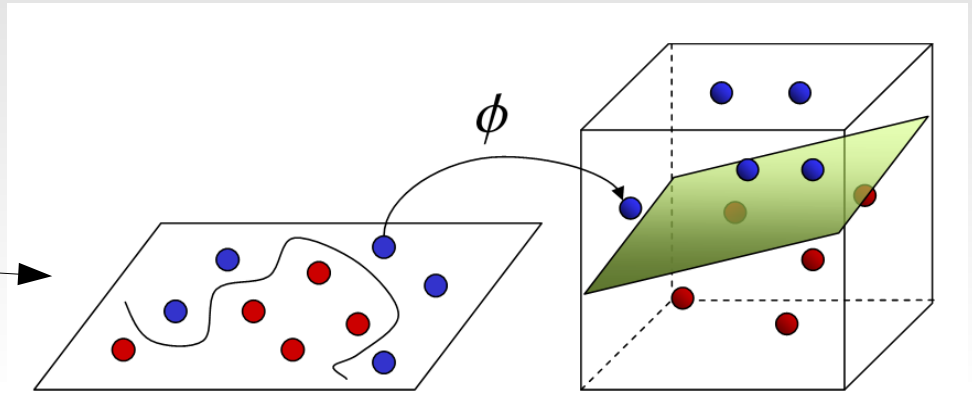


A special property of SVMs is that *maximizes the margin* between the decision hyperplane and the examples in the training set

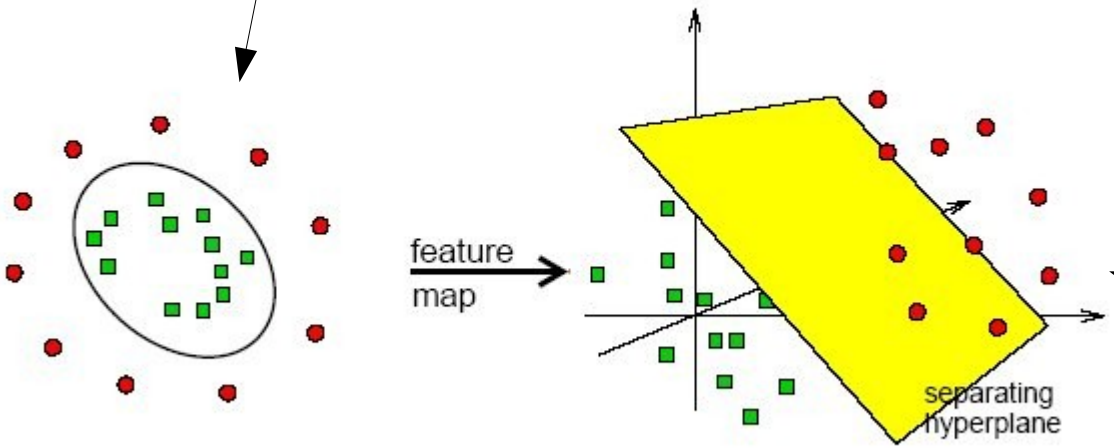


Methods
Support Vector Machine (SVM)

but many times the data does not have a linear solution



Separation may be easier in higher dimensions



complex in low dimensions

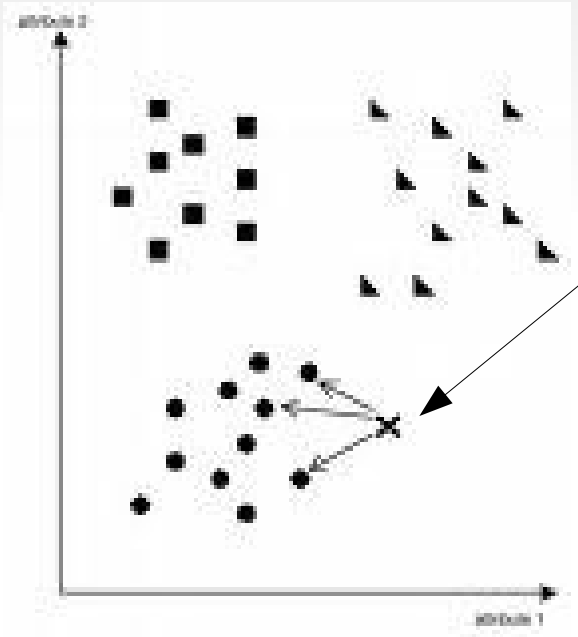
simple in higher dimensions

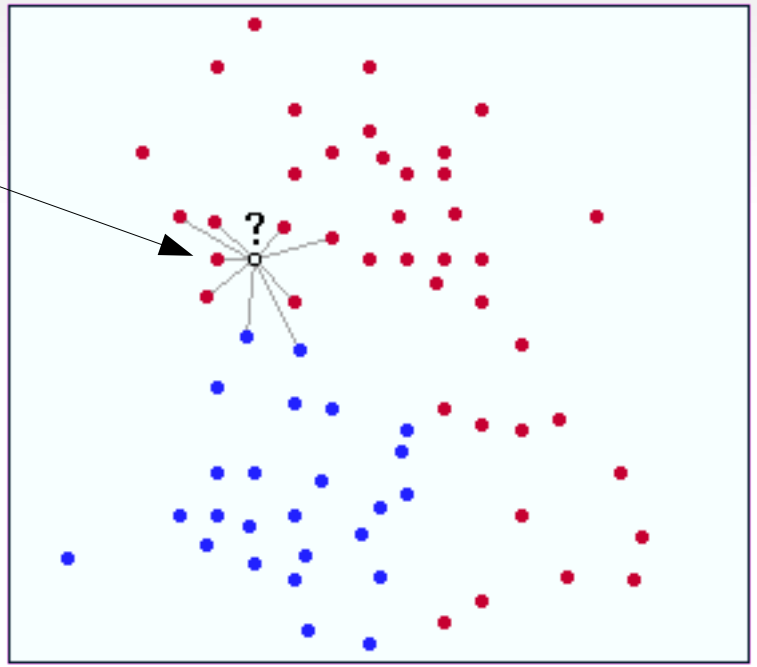
then we can use a *kernel trick* and map the data into a higher dimension space

Methods

Nearest neighbour (KNN)

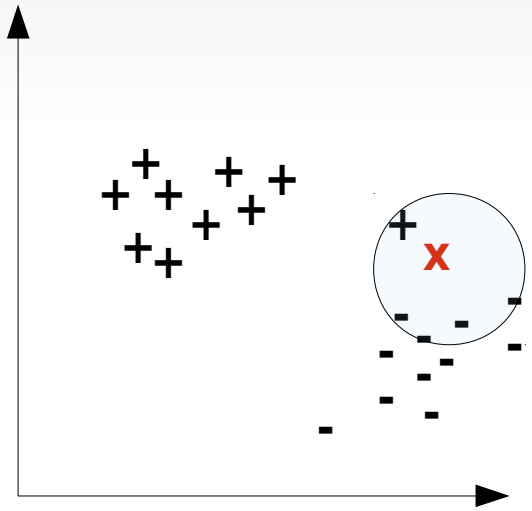
KNN is a distance based method for classifying





Methods
Nearest neighbour (KNN)

… but not always is so easy



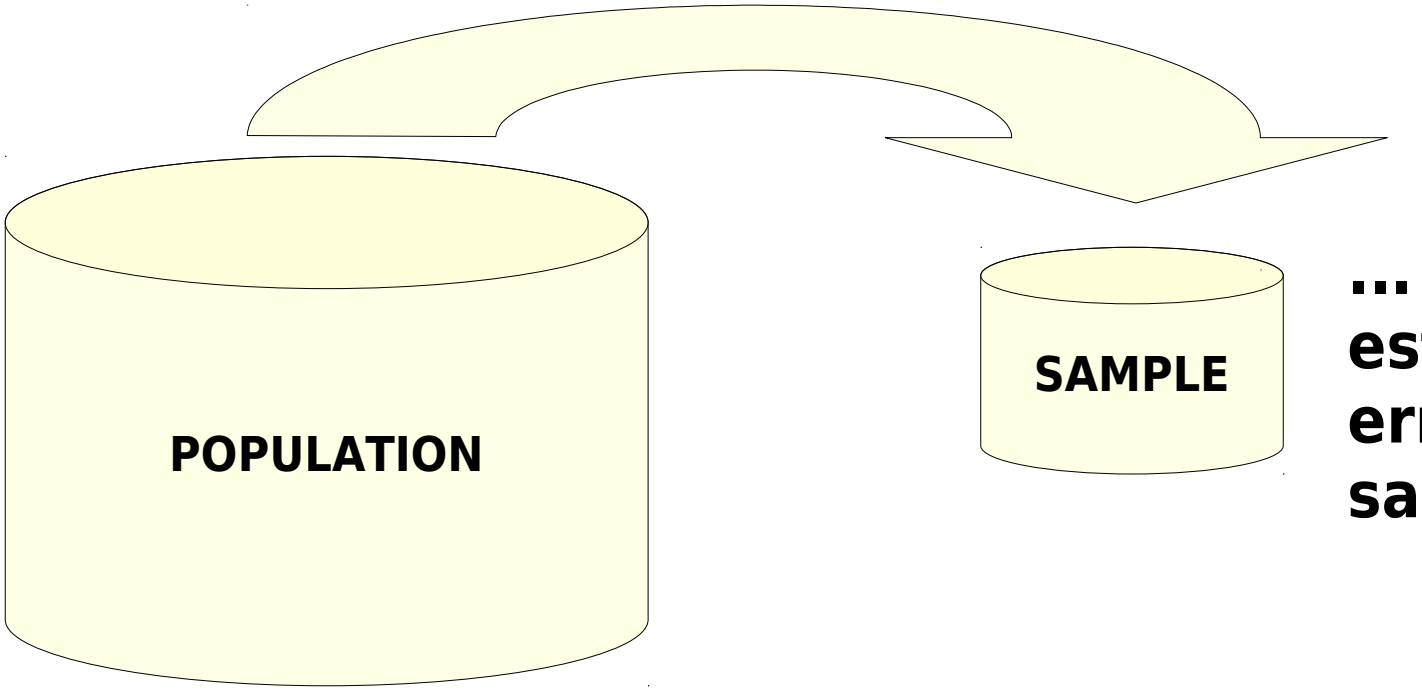
- **x** would be classified as class **+**
- In order to make KNN more robust we are going to look for the ***K*** nearest neighbours instead of only the nearest
- If consider the 5 nearest samples, will find 1 **+** and 4 **-**, so **x** would be classify as **-**
- Predictor tool test automatically with ***K=1..20***

Error estimation

How can we know the error of the predictor?

- It is a crucial aspect of the predictor building, it is going to tell us how much we can trust the predictor, very important in diagnoses

We can NOT know the error



… but we can estimate the error from the samples

Error estimation

Holdout method, a simple idea

#VARIABLE **PHEN** CATEGORICAL{control,tumor} VALUES{control,control,control,control,tumor,tumor}

#NAMES	GSM26878	GSM26883	GSM26886	GSM26887	GSM26903	GSM26910
1007_s_at	11.08578155	11.04457022	11.02479206	11.00837346	11.04430518	11.01921026
1053_at	7.787503325	8.010263804	7.872064511	7.711140759	7.703846348	7.509845931
117_at	7.487539205	7.526590226	7.442468793	7.394731634	7.450764725	7.558967177
121_at	9.589979282	9.516503297	9.610811352	9.282059896	8.323068371	8.664237594
1255_g_at	5.000099854	5.127166256	4.952998877	4.881038876	4.948734762	5.087888404
1294_at	8.358097049	8.403219181	8.255863646	7.947778797	8.328705461	8.230633848
1316_at	7.187245349	6.652952654	6.445444909	6.463659189	6.399722565	6.404821127
1320_at	5.645994428	5.765206267	5.772052661	5.609287091	5.621417391	5.723352308
1405_i_at	7.138444163	7.490198393	7.382302176	7.379200666	7.541671446	6.493521779
1431_at	4.697298725	4.722480562	4.795825627	4.703361751	4.701914661	4.904298823
1438_at	7.430761532	8.112797873	7.578819384	7.699611607	7.496504531	7.776384116

The data set is separated into two sets, called the ***training set*** and the ***testing set***

Use this samples to ***train*** the model

Use this samples to ***predict*** and test if the model is right

Some ***disadvantages***:

- can have a high variance
- depend heavily on which data points end up in the training set and which end up in the test set

Error estimation

So, how can we estimate the error?

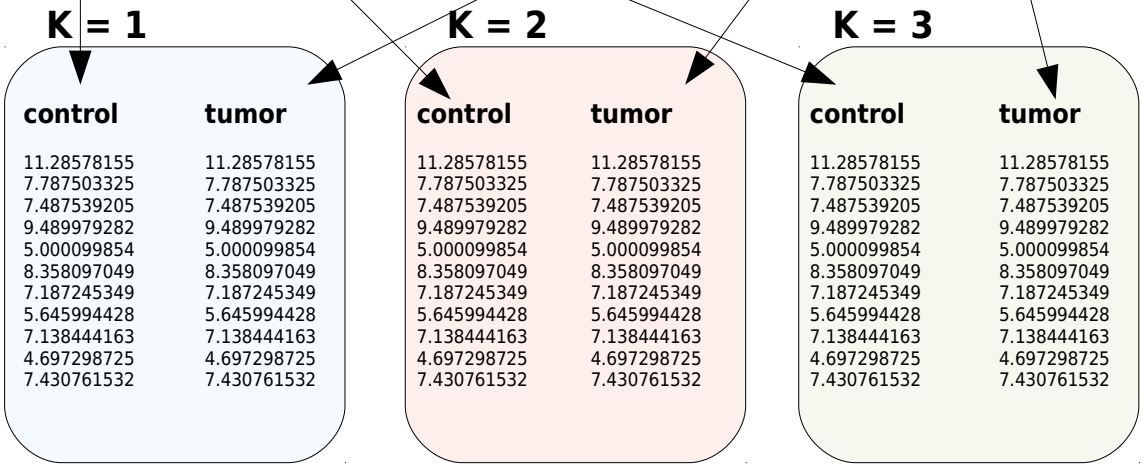
- It is not a simple task, we have to estimate the error that the predictor will have in future gene expression array data
- This estimation can only be done during the ***training stage***
- There are some distinct methods to estimate error, the most common method is ***k-fold cross-validation,*** this method is one way to improve over the *holdout method*
- With this method we are going to split the data in ***k*** partitions and we will use (***k-1***) partitions to train the the other to test the predictor

Error estimation

How does *k-fold cross-validation* work?

#VARIABLE **PHEN** CATEGORICAL{control,tumor} VALUES{control,control,control,tumor,tumor,tumor}

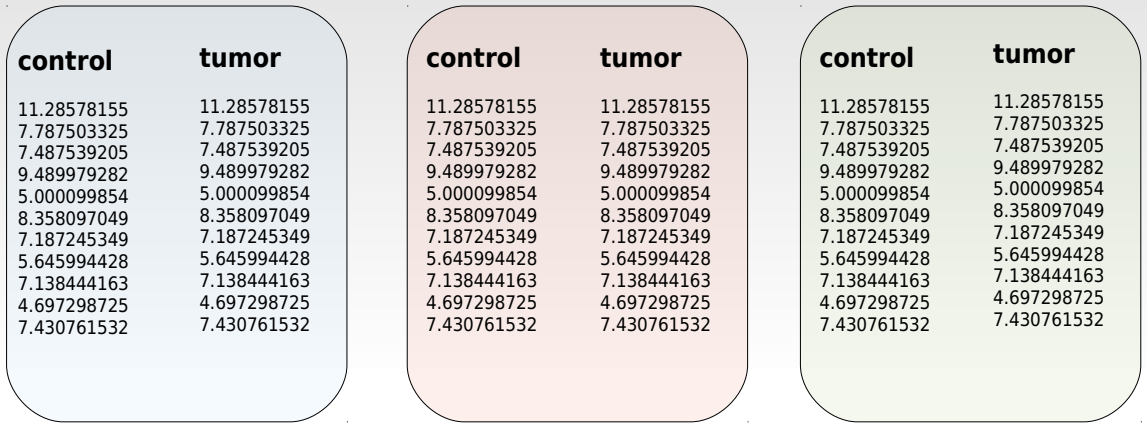
#NAMES	GSM26878	GSM26883	GSM26886	GSM26887	GSM26903	GSM26910
1007_s_at	11.08578155	11.04457022	11.02479206	11.00837346	11.04430518	11.01921026
1053_at	7.787503325	8.010263804	7.872064511	7.711140759	7.703846348	7.509845931
117_at	7.487539205	7.526590226	7.442468793	7.394731634	7.450764725	7.558967177
121_at	9.589979282	9.516503297	9.610811352	9.282059896	8.323068371	8.664237594
1255_g_at	5.000099854	5.127166256	4.952998877	4.881038876	4.948734762	5.087888404
1294_at	8.358097049	8.403219181	8.255863646	7.947778797	8.328705461	8.230633848
1316_at	7.187245349	6.652952654	6.445444909	6.463659189	6.399722565	6.404821127
1320_at	5.645994428	5.765206267	5.772052661	5.609287091	5.621417391	5.723352308
1405_i_at	7.138444163	7.490198393	7.382302176	7.379200666	7.541671446	6.493521779
1431_at	4.697298725	4.722480562	4.795825627	4.703361751	4.701914661	4.904298823
1438_at	7.430761532	8.112797873	7.578819384	7.699611607	7.496504531	7.776384116



- We split arrays into ***k*** partitions of equal size, i.e.: ***k=3***

Error estimation

How does *k-fold cross-validation* work?

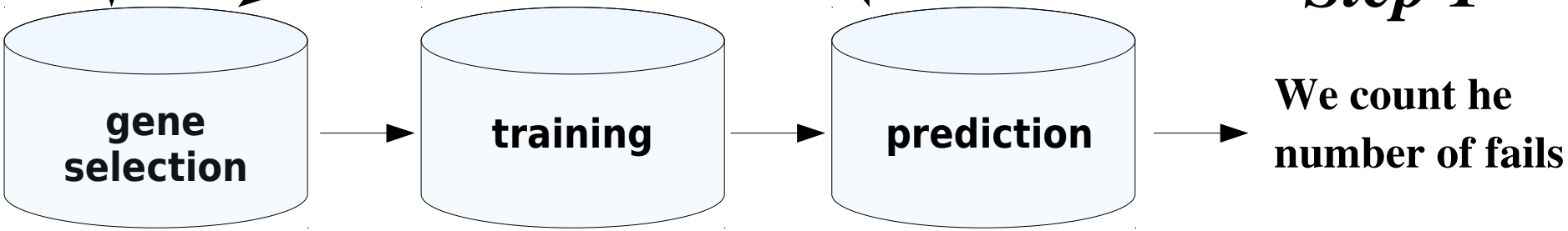


control	tumor
11.28578155	11.28578155
7.787503325	7.787503325
7.487539205	7.487539205
9.489979282	9.489979282
5.000099854	5.000099854
8.358097049	8.358097049
7.187245349	7.187245349
5.645994428	5.645994428
7.138444163	7.138444163
4.697298725	4.697298725
7.430761532	7.430761532

control	tumor
11.28578155	11.28578155
7.787503325	7.787503325
7.487539205	7.487539205
9.489979282	9.489979282
5.000099854	5.000099854
8.358097049	8.358097049
7.187245349	7.187245349
5.645994428	5.645994428
7.138444163	7.138444163
4.697298725	4.697298725
7.430761532	7.430761532

control	tumor
11.28578155	11.28578155
7.787503325	7.787503325
7.487539205	7.487539205
9.489979282	9.489979282
5.000099854	5.000099854
8.358097049	8.358097049
7.187245349	7.187245349
5.645994428	5.645994428
7.138444163	7.138444163
4.697298725	4.697298725
7.430761532	7.430761532

Select genes only with the arrays that participate in training to avoid the *selection bias*



We know the real class of the arrays being predicted

- We train with (k-1) partition and predict with the other one
- Now we repeat ***k*** times the training and the prediction

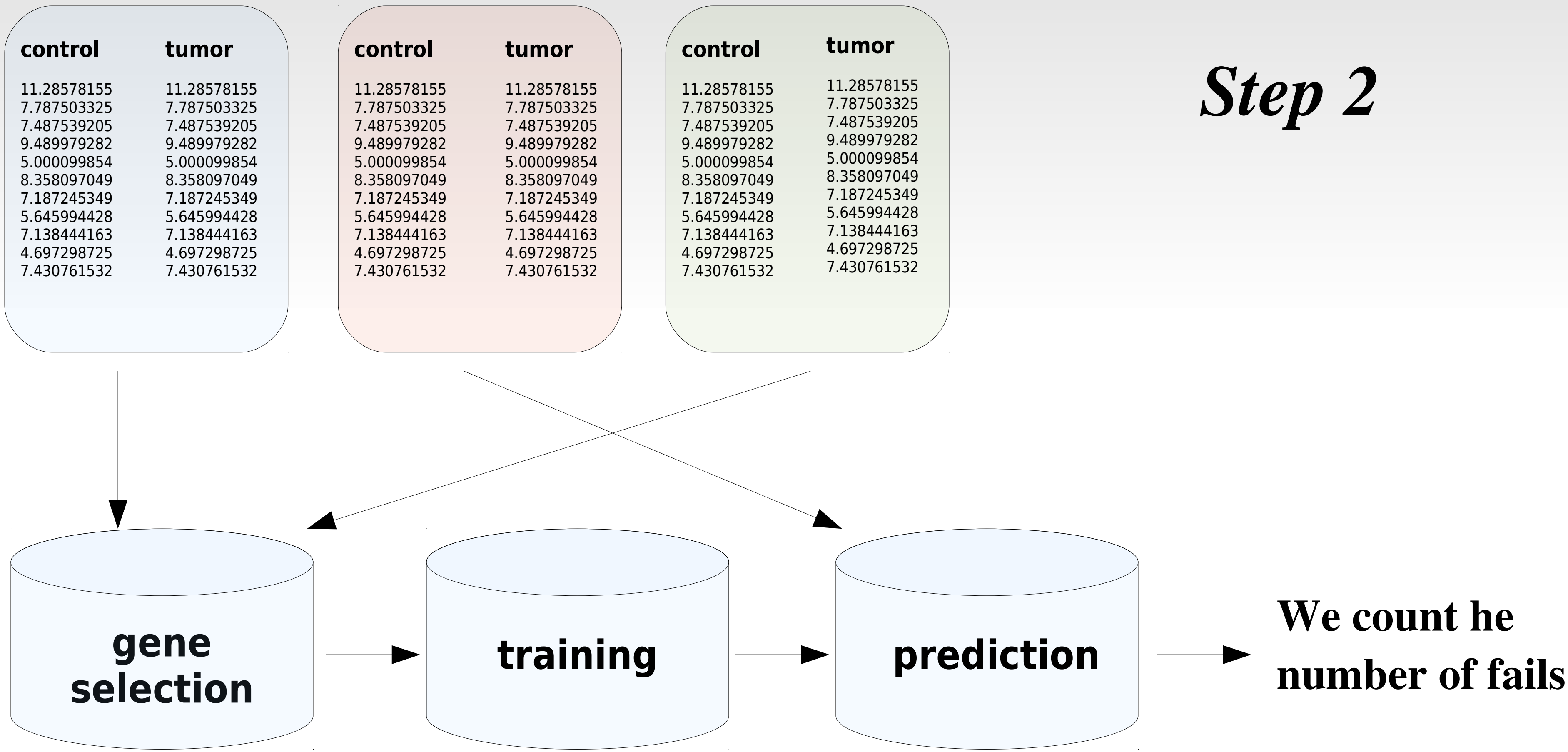
Step 1

We count he number of fails

Error estimation

How does *k-fold cross-validation* work?

Step 2



Error estimation

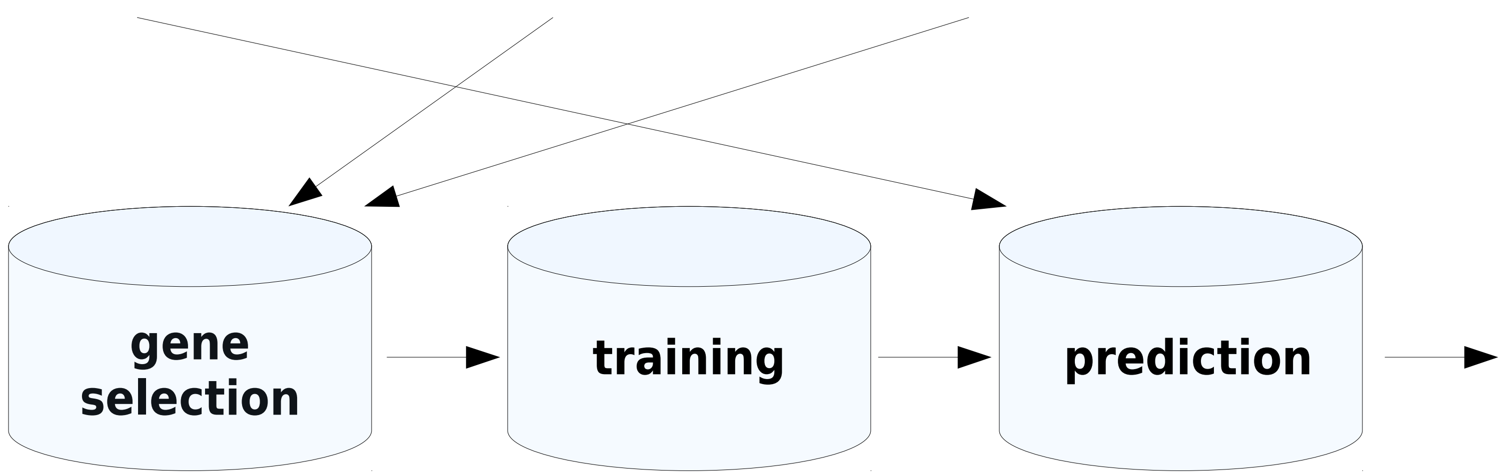
How does *k-fold cross-validation* work?

Step 3

control	tumor
11.28578155	11.28578155
7.787503325	7.787503325
7.487539205	7.487539205
9.489979282	9.489979282
5.000099854	5.000099854
8.358097049	8.358097049
7.187245349	7.187245349
5.645994428	5.645994428
7.138444163	7.138444163
4.697298725	4.697298725
7.430761532	7.430761532

control	tumor
11.28578155	11.28578155
7.787503325	7.787503325
7.487539205	7.487539205
9.489979282	9.489979282
5.000099854	5.000099854
8.358097049	8.358097049
7.187245349	7.187245349
5.645994428	5.645994428
7.138444163	7.138444163
4.697298725	4.697298725
7.430761532	7.430761532

control	tumor
11.28578155	11.28578155
7.787503325	7.787503325
7.487539205	7.487539205
9.489979282	9.489979282
5.000099854	5.000099854
8.358097049	8.358097049
7.187245349	7.187245349
5.645994428	5.645994428
7.138444163	7.138444163
4.697298725	4.697298725
7.430761532	7.430761532



We count he number of fails

Error estimation
How does *k-fold cross-validation* work?

- Notice that the ***k-fold cross-validation***
  - ***all samples*** has been used in ***training*** and ***prediction***
  - the more arrays we used in training the better predictor we will have
  - remember: occurs in ***training stage***
  - after cross-validation we have an error estimation
- Prophet, and many other predictors use a particular case of the *k-fold cross-validation*, it is called ***leaving-one-out cross-validation***

Error estimation

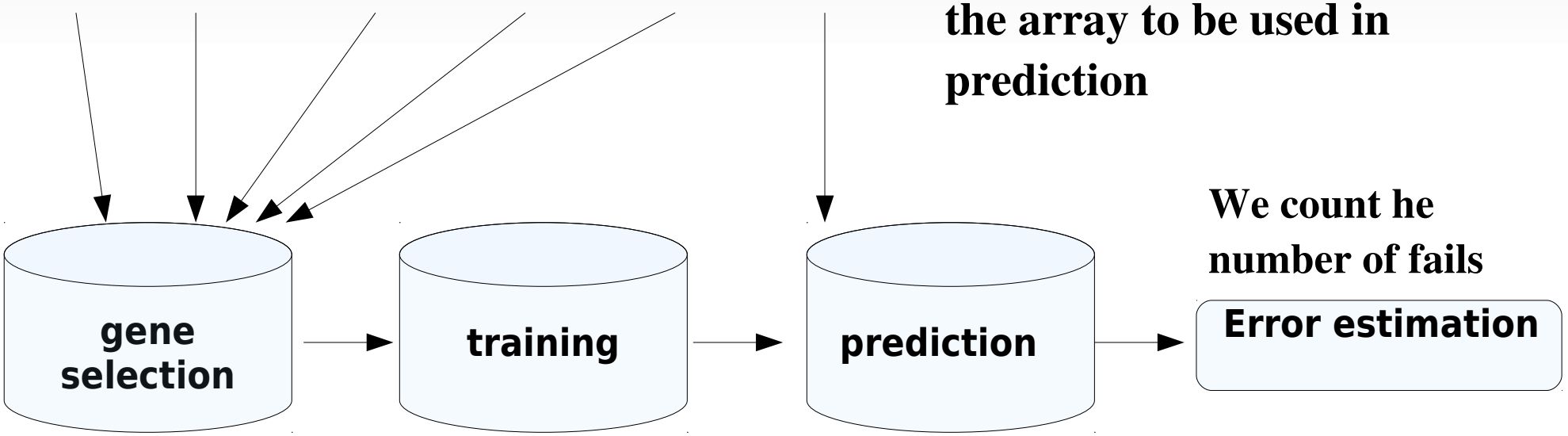
Leaving-one-out cross-validation

control	control	control	tumor	tumor	tumor
11.28578155	11.28578155	11.28578155	11.28578155	11.28578155	11.28578155
7.787503325	7.787503325	7.787503325	7.787503325	7.787503325	7.787503325
7.487539205	7.487539205	7.487539205	7.487539205	7.487539205	7.487539205
9.489979282	9.489979282	9.489979282	9.489979282	9.489979282	9.489979282
5.000099854	5.000099854	5.000099854	5.000099854	5.000099854	5.000099854
8.358097049	8.358097049	8.358097049	8.358097049	8.358097049	8.358097049
7.187245349	7.187245349	7.187245349	7.187245349	7.187245349	7.187245349
5.645994428	5.645994428	5.645994428	5.645994428	5.645994428	5.645994428
7.138444163	7.138444163	7.138444163	7.138444163	7.138444163	7.138444163
4.697298725	4.697298725	4.697298725	4.697298725	4.697298725	4.697298725
7.430761532	7.430761532	7.430761532	7.430761532	7.430761532	7.430761532

- ***K = number of arrays***

  i.e.: ***k=6***

- **Repeat k times changing the array to be used in prediction**

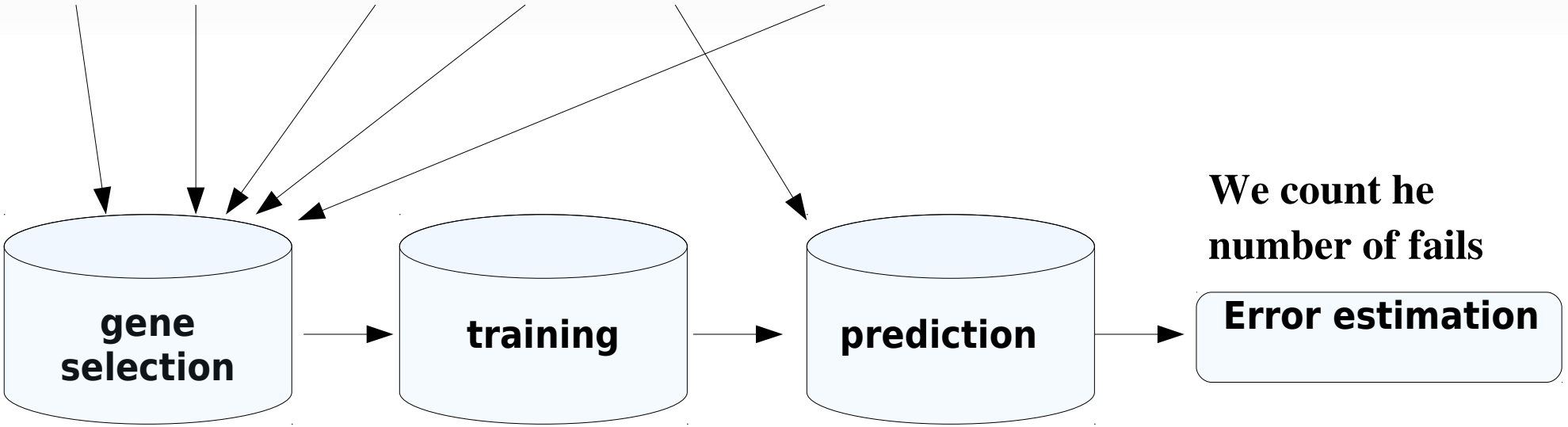


Error estimation

Leaving-one-out cross-validation

control	control	control	tumor	tumor	tumor
11.28578155	11.28578155	11.28578155	11.28578155	11.28578155	11.28578155
7.787503325	7.787503325	7.787503325	7.787503325	7.787503325	7.787503325
7.487539205	7.487539205	7.487539205	7.487539205	7.487539205	7.487539205
9.489979282	9.489979282	9.489979282	9.489979282	9.489979282	9.489979282
5.000099854	5.000099854	5.000099854	5.000099854	5.000099854	5.000099854
8.358097049	8.358097049	8.358097049	8.358097049	8.358097049	8.358097049
7.187245349	7.187245349	7.187245349	7.187245349	7.187245349	7.187245349
5.645994428	5.645994428	5.645994428	5.645994428	5.645994428	5.645994428
7.138444163	7.138444163	7.138444163	7.138444163	7.138444163	7.138444163
4.697298725	4.697298725	4.697298725	4.697298725	4.697298725	4.697298725
7.430761532	7.430761532	7.430761532	7.430761532	7.430761532	7.430761532

Step 2



We count he number of fails

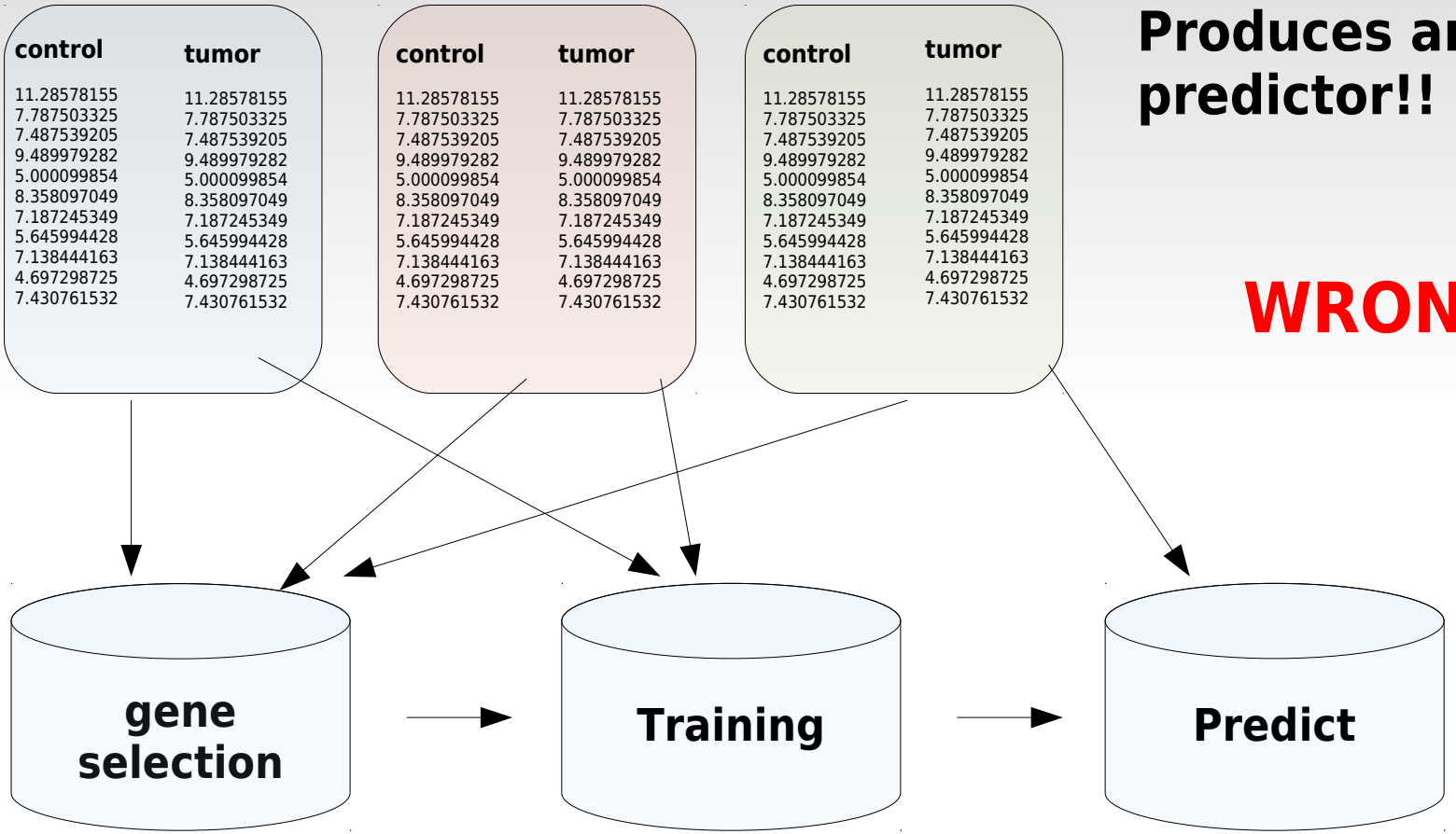
and so on ...

Error estimation
Leaving-one-out cross-validation

- Notice that with leaving-one-out cross-validation:
  - we use all arrays but one for training, the more arrays we used in training the better predictor we will have
  - all of them are also used for error estimation

Error estimation

Selection Bias Problem



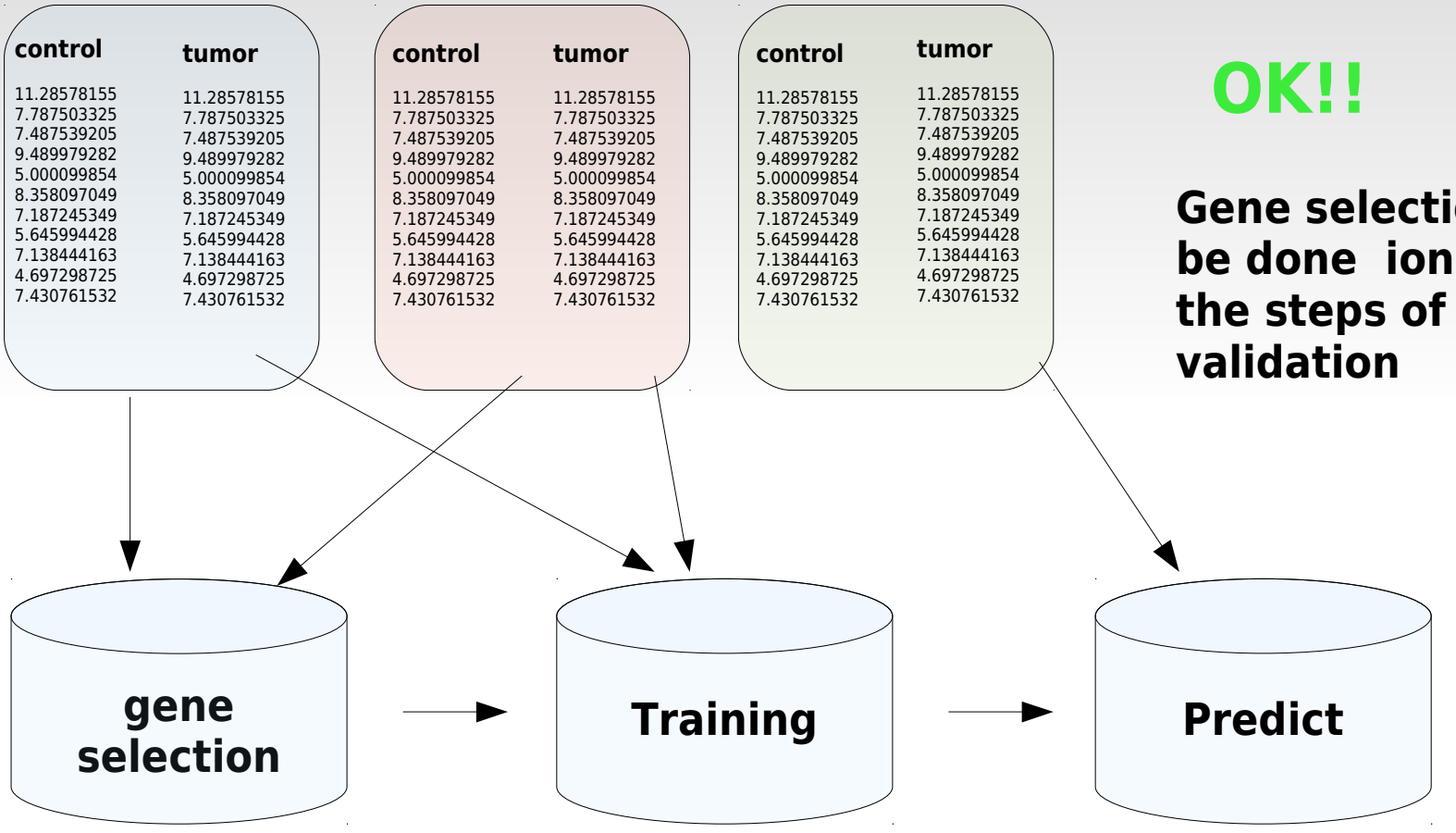
Produces an optimistic predictor!!

WRONG!!

The genes used in training were selected using the partition to be tested

Error estimation

Selection Bias Solution



OK!!

Gene selection must be done ion each of the steps of cross-validation

the partition to be tested simulates new data

Error estimation

Selection Bias in literature

van´t Veer et al. **Gene expression profiling predicts clinical outcome of breast cancer.** *Nature, vol. 415, Jan 2002*

after their initial diagnosis for an interval of at least 5 years (good prognosis group, mean follow-up of 8.7 years), and 34 patients had developed distant metastases within 5 years (poor prognosis group, mean time to metastases 2.5 years) (Fig. 2a). To identify reliably good and poor prognostic tumours, we used a powerful three-step supervised classification method, similar to those used previously[8,17,18]. In brief, approximately 5,000 genes (significantly regulated in more than 3 tumours out of 78) were selected from the 25,000 genes on the microarray. The correlation coefficient of the expression for each gene with disease outcome was calculated and 231 genes were found to be significantly associated with disease outcome (correlation coefficient <-0.3 or >0.3). In the second step, these 231 genes were rank-ordered on the basis of the magnitude of the correlation coefficient. Third, the number of genes in the 'prognosis classifier' was optimized by sequentially adding subsets of 5 genes from the top of this rank-ordered list and

Selection Bias problem detected and solved

Since ***2004*** predictors publications avoid this problem

Selection bias in gene extraction on the basis of microarray gene-expression data

Christophe Ambroise † **and Geoffrey J. McLachlan** ‡ , §

+ Author Affiliations

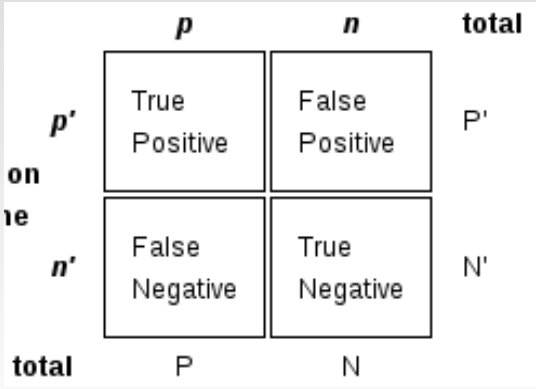
Edited by Stephen E. Fienberg, Carnegie Mellon University, Pittsburgh, PA, and approved March 21, 2002 (received for review February 20, 2002)

Abstract

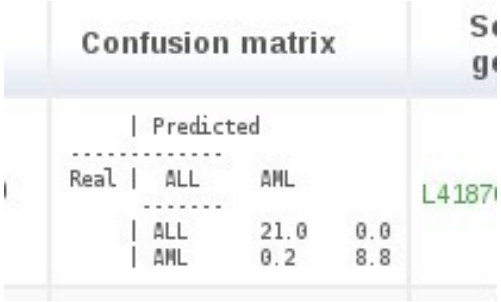
In the context of cancer diagnosis and treatment, we consider the problem of constructing an accurate prediction rule on the basis of a relatively small number of tumor tissue samples of known type containing the expression data on very many (possibly thousands) genes. Recently, results have been presented in the literature suggesting that it is possible to construct a prediction rule from only a few genes such that it has a negligible prediction error rate. However, in these results the test error or the leave-one-out cross-validated error is calculated without allowance for the selection bias. There is no allowance because the rule is either tested on tissue samples that were used in the first instance to select the genes being used in the rule or because the cross-validation of the rule is not external to the selection process; that is, gene selection is not performed in training the rule at each stage of the cross-validation process. We describe how in practice the selection bias can be assessed and corrected for by either performing a cross-validation or applying the bootstrap external to the selection process. We recommend using 10-fold rather than leave-one-out cross-validation, and concerning the bootstrap, we suggest using the so-called .632+ bootstrap error estimate designed to handle overfitted prediction rules. Using two published data sets, we demonstrate that when correction is made for the selection bias, the cross-validated error is no longer zero for a subset of only a few genes.

Error estimation
Some metrics

Confusion matrix



- Accuracy (ACC)
  - ACC = (TP + TN) / (P + N)
  - Works very bad with unbalanced datasets
- Area Under ROC (AUC)
- Matthews correlation coefficient (MCC)
- Root Mean Square Error (RMSE)



Acc = 92%

46	4
4	46

90	0
8	2

Accuracy not a good metric in unbalanced datasets

Preditor's Future

Predictors to be used in clinic

Our group was one the few European groups invited to participate in MAQC-II

The standards and basis for predictors are established

nature biotechnology

ARTICLES

The MicroArray Quality Control (MAQC)-II study of common practices for the development and validation of microarray-based predictive models

MAQC Consortium*

Gene expression data from microarrays are being applied to predict preclinical and clinical endpoints, but the reliability of these predictions has not been established. In the MAQC-II project, 36 independent teams analyzed six microarray data sets to generate predictive models for classifying a sample with respect to one of 13 endpoints indicative of lung or liver toxicity in rodents, or of breast cancer, multiple myeloma or neuroblastoma in humans. In total, >30,000 models were built using many combinations of analytical methods. The teams generated predictive models without knowing the biological meaning of some of the endpoints and, to mimic clinical reality, tested the models on data that had not been used for training. We found that model performance depended largely on the endpoint and team proficiency and that different approaches generated models of similar performance. The conclusions and recommendations from MAQC-II should be useful for regulatory agencies, study committees and independent investigators that evaluate methods for global gene expression analysis.

As part of the United States Food and Drug Administration's (FDA's) Critical Path Initiative to medical product development (http://www.fda.gov/oc/initiatives/criticalpath/), the MAQC consortium began in February 2005 with the goal of addressing various microarray reliability concerns raised in publications[1–9] pertaining to reproducibility of gene signatures. The first phase of this project (MAQC-I) extensively evaluated the technical performance of microarray platforms in identifying all differentially expressed genes that would potentially constitute biomarkers. The MAQC-I found high intra-platform reproducibility across test sites, as well as inter-platform concordance of differentially expressed gene lists[10–15] and confirmed that microarray of guidelines for safe and effective use of preclinical and clinical genomic data. Although previous studies have compared and benchmarked individual steps in the model development process[19], no prior published work has, to our knowledge, extensively evaluated current community practices on the development and validation of microarray-based predictive models.

Microarray-based gene expression data and prediction models are increasingly being submitted by the regulated industry to the FDA to support medical product development and testing applications[20]. For example, gene expression microarray–based assays that have been approved by the FDA as diagnostic tests include the Agendia

Building the predictor

How do we build the predictor?

Training dataset (our sample)

#VARIABLE **PHEN** CATEGORICAL{control,tumor} VALUES{control,control,control,tumor,tumor,tumor}

#NAMES	GSM26878	GSM26883	GSM26886	GSM26887	GSM26903	GSM26910
1007_s_at	11.08578155	11.04457022	11.02479206	11.00837346	11.04430518	11.01921026
1053_at	7.787503325	8.010263804	7.872064511	7.711140759	7.703846348	7.509845931
117_at	7.487539205	7.526590226	7.442468793	7.394731634	7.450764725	7.558967177
121_at	9.589979282	9.516503297	9.610811352	9.282059896	8.323068371	8.664237594
1255_g_at	5.000099854	5.127166256	4.952998877	4.881038876	4.948734762	5.087888404
1294_at	8.358097049	8.403219181	8.255863646	7.947778797	8.328705461	8.230633848
1316_at	7.187245349	6.652952654	6.445444909	6.463659189	6.399722565	6.404821127
1320_at	5.645994428	5.765206267	5.772052661	5.609287091	5.621417391	5.723352308
1405_i_at	7.138444163	7.490198393	7.382302176	7.379200666	7.541671446	6.493521779
1431_at	4.697298725	4.722480562	4.795825627	4.703361751	4.701914661	4.904298823
1438_at	7.430761532	8.112797873	7.578819384	7.699611607	7.496504531	7.776384116

Global process

Training stage

Cross-validation

Error estimation

Use all arrays to train the predictor!!

Final predictor

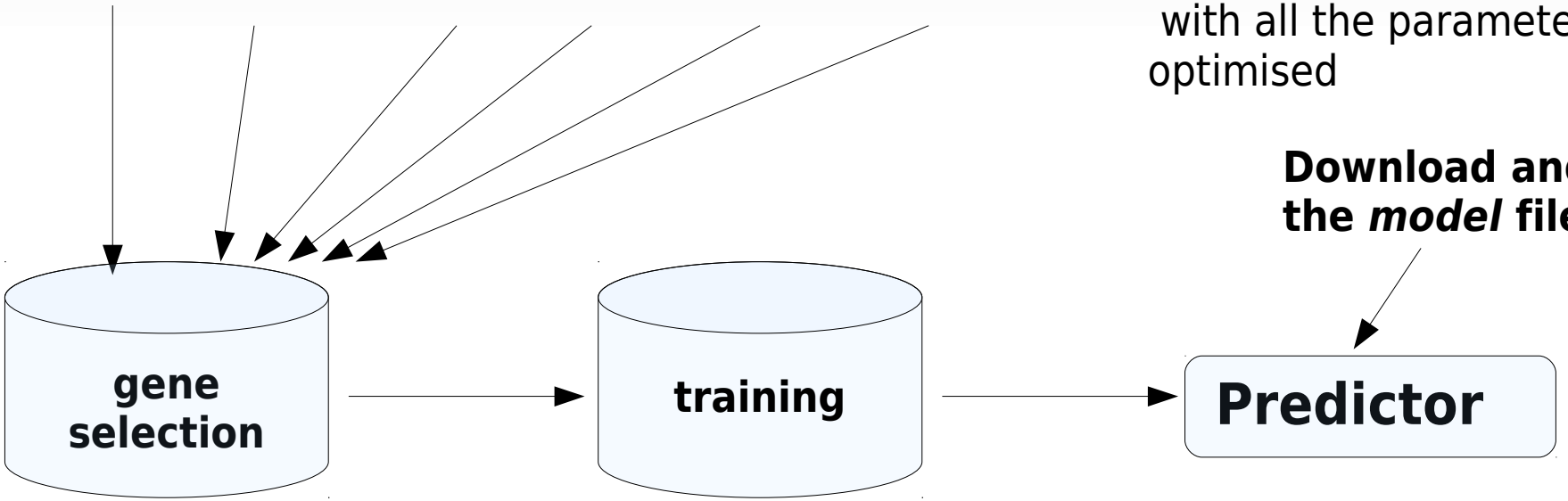
Building the predictor

How do we build the predictor?

control	tumor	control	tumor	control	tumor
11.28578155	11.28578155	11.28578155	11.28578155	11.28578155	11.28578155
7.787503325	7.787503325	7.787503325	7.787503325	7.787503325	7.787503325
7.487539205	7.487539205	7.487539205	7.487539205	7.487539205	7.487539205
9.489979282	9.489979282	9.489979282	9.489979282	9.489979282	9.489979282
5.000099854	5.000099854	5.000099854	5.000099854	5.000099854	5.000099854
8.358097049	8.358097049	8.358097049	8.358097049	8.358097049	8.358097049
7.187245349	7.187245349	7.187245349	7.187245349	7.187245349	7.187245349
5.645994428	5.645994428	5.645994428	5.645994428	5.645994428	5.645994428
7.138444163	7.138444163	7.138444163	7.138444163	7.138444163	7.138444163
4.697298725	4.697298725	4.697298725	4.697298725	4.697298725	4.697298725
7.430761532	7.430761532	7.430761532	7.430761532	7.430761532	7.430761532

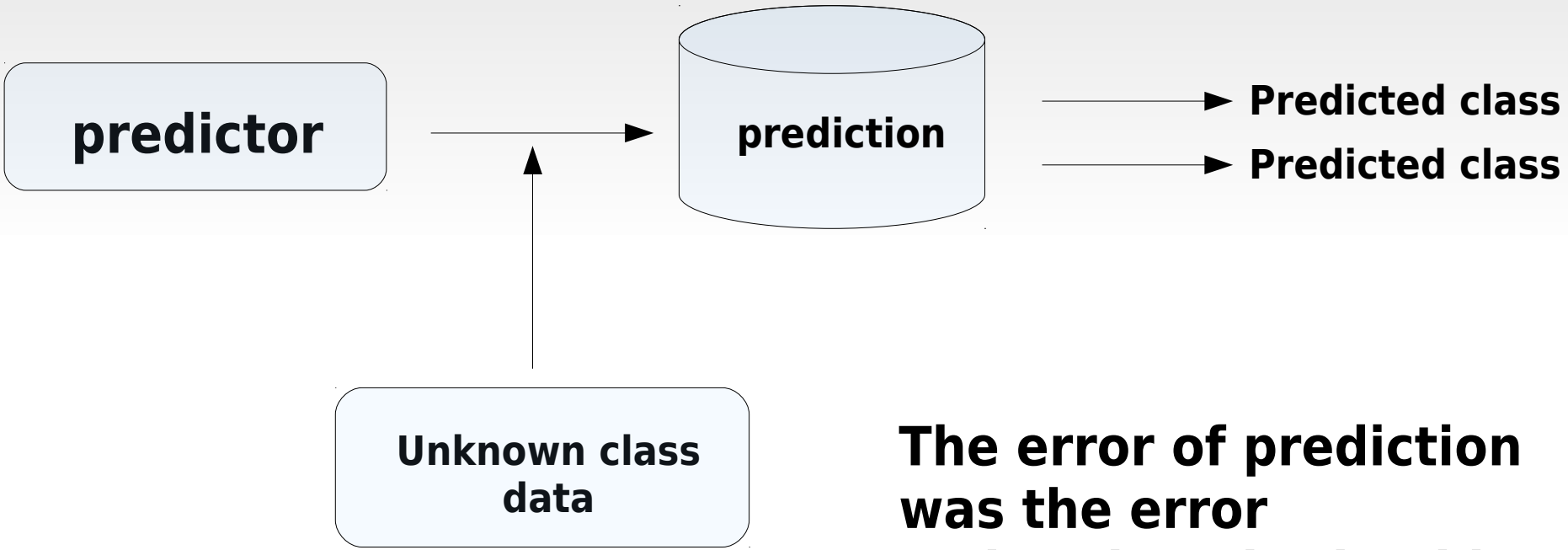
- To build the predictor we use all arrays, in order to get the best predictor

- The predictor is a model with all the parameters optimised



Building the predictor

How do we use the predictor?



The error of prediction was the error estimation obtained in training stage!!

Preparing the data

Training input data format

genes

class labels

The file ...

#VARIABLE *tumor* CATEGORICAL{c1,c2,c3} VALUES{c1,c1,c2,c2,c3,c3} DESCRIPTION{}

NAMES	GSM26878	GSM26883	GSM26886	GSM26887	GSM26903	GSM26910
1007_s_at	11.08578155	11.04457022	11.02479206	11.00837346	11.04430518	11.01921026
1053_at	7.787503325	8.010263804	7.872064511	7.711140759	7.703846348	7.509845931
117_at	7.487539205	7.526590226	7.442468793	7.394731634	7.450764725	7.558967177
121_at	9.589979282	9.516503297	9.610811352	9.282059896	8.323068371	8.664237594
1255_g_at	5.000099854	5.127166256	4.952998877	4.881038876	4.948734762	5.087888404
1294_at	8.358097049	8.403219181	8.255863646	7.947778797	8.328705461	8.230633848
1316_at	7.187245349	6.652952654	6.445444909	6.463659189	6.399722565	6.404821127
1320_at	5.645994428	5.765206267	5.772052661	5.609287091	5.621417391	5.723352308
1405_i_at	7.138444163	7.490198393	7.382302176	7.379200666	7.541671446	6.493521779
1431_at	4.697298725	4.722480562	4.795825627	4.703361751	4.701914661	4.904298823
1438_at	7.430761532	8.112797873	7.578819384	7.699611607	7.496504531	7.776384116
1487_at	7.646126117	7.544048497	8.754540699	8.476873549	9.084035203	9.028724488
1494_f_at	7.498031252	7.679595836	7.662561072	7.201093115	7.426192546	7.669669586
1598_g_at	10.31770877	10.92530764	10.50092321	9.630201704	10.23473332	10.49766918
160020_at	8.529411037	8.738065073	8.617216353	8.445386532	8.425365655	8.76023381
1729_at	9.607320487	8.171988017	8.73040537	8.978602862	9.156752025	8.033237589
1773_at	6.216319215	6.441555855	6.165785507	6.325464779	6.121753223	6.229420354
177_at	6.535525364	6.453887146	6.519400663	6.333366799	6.385077422	6.407541976

arrays

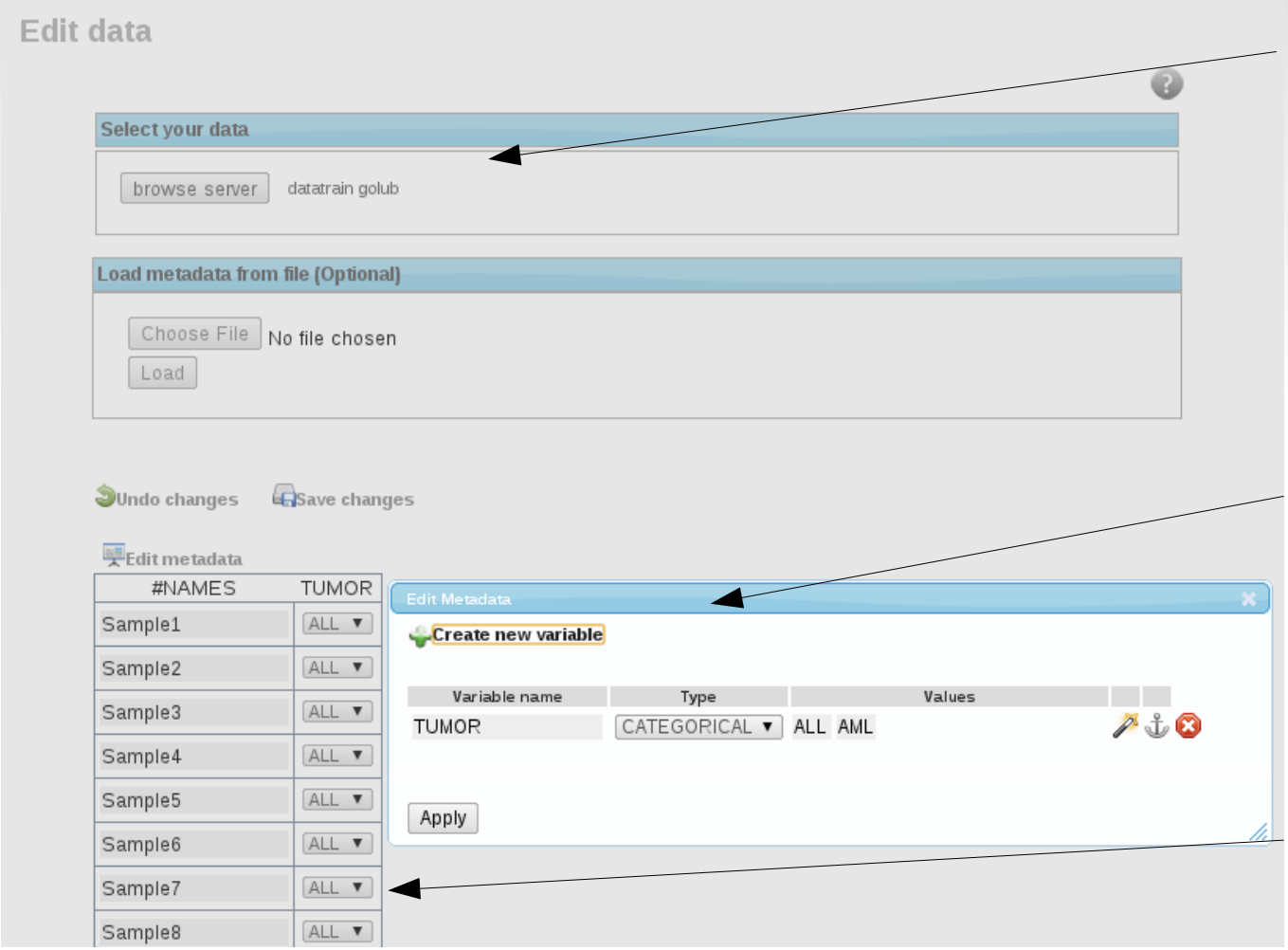
Babelomics datafile format

tab seperated file

lines beginnig with # will not be taking into account

Preparing the data

Training input data format



Select the dataset

Create the VARIABLE

Assign values to the samples

Prophet
Training input form

Online examples

Help button

Prophet
Training results

Summary

Combined results (best 5 per classifier) best_classifiers_table.txt

-class-prediction-
2 visites

rheumatoid clusterir
2010-09-29 17:32:12.0
-clustering-
3 visites

Index	Classifier	Parameters	Accuracy	MCC	RMSE	AUC	Confusion matrix	Selected genes
2	SVM	cost=0.6, features=26	0.9933	0.9842	0.0163	0.99	| Predicted Real | ALL AML | ALL 21.0 0.0 | AML 0.2 8.8	L41870_at,M5515...
3	SVM	cost=0.8, features=26	0.9933	0.9842	0.0163	0.99	| Predicted Real | ALL AML | ALL 21.0 0.0 | AML 0.2 8.8	L41870_at,M5515...
4	SVM	cost=1, features=26	0.9933	0.9842	0.0163	0.99	| Predicted Real | ALL AML | ALL 21.0 0.0 | AML 0.2 8.8	L41870_at,M5515...
5	SVM	cost=1.2, features=26	0.9933	0.9842	0.0163	0.99	| Predicted Real | ALL AML | ALL 21.0 0.0 | AML 0.2 8.8	L41870_at,M5515...
6	SVM	cost=1.4, features=26	0.9933	0.9842	0.0163	0.99	| Predicted Real | ALL AML | ALL 21.0 0.0 | AML 0.2 8.8	L41870_at,M5515...

1) A table comparing the results of all the methods selected with the metrics and Confusion matrix

Prophet
Training results

2) We will get also a table comapring the prediction for each of the samples

Percentage of correct classification per sample/classifier :

Sample	knn=5, features=2	knn=11, features=2	knn=3, features=2	knn=4, features=2	knn=6, features=2
Sample1	100%	100%	100%	100%	100%
Sample2	100%	100%	100%	100%	100%
Sample3	100%	100%	100%	100%	100%
Sample4	100%	100%	100%	100%	100%
Sample5	100%	100%	100%	100%	100%
Sample6	100%	100%	100%	100%	100%
Sample7	100%	100%	100%	100%	100%
Sample8	100%	100%	100%	100%	100%
Sample9	100%	100%	100%	100%	100%
Sample10	100%	100%	100%	100%	100%

prev next page 1 of 15

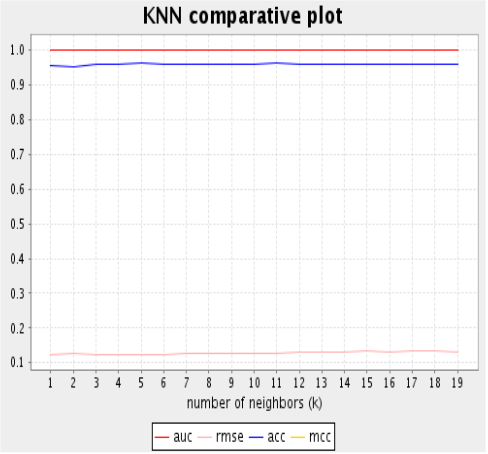
Prophet
Training results

KNN results

KNN classifications : **KNN_table.txt**

Index	Classifier	Parameters	Accuracy	MCC	RMSE	AUC	Confusion matrix	Selected genes
0	KNN	knn=1, features=2	0.9573	1	0.1227	1	| Predicted Real | Iris-setosa Iris-versicolor Iris-virginica | Iris-setosa 50.0 0.0 0.0 | Iris-versicolor 0.0 47.2 2.8 | Iris-virginica 0.0 3.6 46.4	petallength,pet...
1	KNN	knn=2, features=2	0.9533	1	0.1274	1	| Predicted Real | Iris-setosa Iris-versicolor Iris-virginica | Iris-setosa 50.0 0.0 0.0 | Iris-versicolor 0.0 47.6 2.4 | Iris-virginica 0.0 4.6 45.4	petallength,pet...
2	KNN	knn=3, features=2	0.9613	1	0.1228	1	| Predicted Real | Iris-setosa Iris-versicolor Iris-virginica | Iris-setosa 50.0 0.0 0.0 | Iris-versicolor 0.0 47.2 2.8 | Iris-virginica 0.0 3.0 47.0	petallength,pet...
3	KNN	knn=4, features=2	0.9613	1	0.1229	1	| Predicted Real | Iris-setosa Iris-versicolor Iris-virginica | Iris-setosa 50.0 0.0 0.0 | Iris-versicolor 0.0 48.0 2.0 | Iris-virginica 0.0 3.8 46.2	petallength,pet...
4	KNN	knn=5, features=2	0.9627	1	0.1237	1	| Predicted Real | Iris-setosa Iris-versicolor Iris-virginica | Iris-setosa 50.0 0.0 0.0 | Iris-versicolor 0.0 48.0 2.0 | Iris-virginica 0.0 3.6 46.4	petallength,pet...
5	KNN	knn=6, features=2	0.9613	1	0.1244	1	| Predicted Real | Iris-setosa Iris-versicolor Iris-virginica | Iris-setosa 50.0 0.0 0.0 | Iris-versicolor 0.0 47.8 2.2 | Iris-virginica 0.0 3.6 46.4	petallength,pet...
							| Predicted	

parative plot :



For each algorithm

Prophet
Prediction results

Test result

Test result table : **test_result.txt**

Sample_names	KNN k=5, features=2	KNN k=11, features=2	KNN k=3, features=2	KNN k=4, features=2	KNN k=6, features=2
Sample1	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa
Sample2	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa
Sample3	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa
Sample4	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa
Sample5	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa
Sample6	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa
Sample7	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa
Sample8	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa
Sample9	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa
Sample10	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa	Iris-setosa

prev next page 1 of 15

The End

http://babelomics.bioinfo.cipf.es