

Data Science

Deriving Knowledge from Data at Scale

*Feature extraction and selection are the **most important** but underrated step of machine learning. Better features are better than better algorithms...*

*of machine learning. Better features are better than better algorithms...
Feature extraction and selection are the most important but underrated steps*

Valentine N. Fontama

August 6th, 2015

Deriving Knowledge from Data at Scale



Review...

Lecture 5 Homework 5a, *discussion...*

Targeted Marketing Campaign

In this problem we will use historical data from past customer responses to build a classification model. The model will then be applied to a new set of prospects to whom we may want extend an offer for a PEP. Rather than doing a mass marketing campaign to all new prospects, we would like to target those that are likely to respond positively to our offer (according to our classification model).

There are two data sets available (the data sets are comma delimited, and the first row contains the field names):

- [bank-data.csv](#) - Labelled training data set for Building a Model
- [bank-new.csv](#) - A set of new customers from which to find the "hot prospects" for the next mailing, using the profiles built from the training set.

- Decision Tree, 10-fold cross validation;
- Make predictions with the model against test data set;
- False Positive costs \$10, False Negative results in a \$1000 loss;
- Lift charts over predictive data;

Lecture 5 Homework 5a, *discussion...*

- Would you send out conservatively or with wild abandon?
- Know the value of your errors, type I and type II;
- Know the value of accurate predictions;
- Often secondary considerations involved as well, such as balancing the cost (time) of running a second evaluation (model) versus the value of giving an early warning signal.

K-Means, Session Data (K= 4)

Final cluster centroids:

Attribute	Full Data (100)	Cluster#			
		0 (31)	1 (44)	2 (9)	3 (16)
<hr/>					
Home	0.6	0.0645	0.75	1	1
Products	0.72	1	0.7273	0.2222	0.4375
Search	0.43	0.3548	0.2273	1	0.8125
Prod_A	0.53	0.0323	0.9773	1	0
Prod_B	0.55	1	0.3409	0.6667	0.1875
Prod_C	0.45	0.6774	0.2955	0.7778	0.25
Cart	0.61	0.4516	0.7727	0.4444	0.5625
Purchase	0.39	0.2903	0.5455	0.2222	0.25

- Cluster 2 – Home → Search → Prod_B
- Recommend Prod_A

- Cluster 0 – Products → Prod_C
- Recommend Prod_B

- Cluster 1 – Focused Browsers

- Clusters 2 & 3 – Searchers

- Cluster 2 – Window Shoppers

Nearest Neighbor Classification

Amazon and "Customers who purchased X also purchased Y" feature.

As scalable for a 20-customer database as it is for a 20 million-customer database, and you can define the number of results you want to find. A great technique...

Customer	Age	Income	Purchased Product
1	45	46k	Book
2	39	100k	TV
3	35	38k	DVD
4	69	150k	Car Cover
5	58	51k	???

Step 1: Determine Distance Formula

$$\text{Distance} = \text{SQRT}((58 - \text{Age})^2 + ((51000 - \text{Income})^2))$$

Step 2: Calculate the Score

Customer	Score	Purchased Product
1	.385	Book
2	.710	TV
3	.686	DVD
4	.941	Car Cover
5	0.0	???

```
% -- The attributes are (donated by Riccardo Leardi,  
% riclea@anchem.unige.it )  
% 1) Alcohol  
% 2) Malic acid  
% 3) Ash  
% 4) Alcalinity of ash  
% 5) Magnesium  
% 6) Total phenols  
% 7) Flavanoids  
% 8) Nonflavanoid phenols  
% 9) Proanthocyanins  
% 10)Color intensity  
% 11)Hue  
% 12)OD280/OD315 of diluted wines  
% 13)Proline
```

Classifiers → Lazy → IBk

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Open file... Open URL... Open DB... Generate... Undo Edit... Save...

Filter Choose None Apply

Current relation
Relation: wine
Instances: 178 Attributes: 14

Attributes
All None Invert Pattern

No.	Name
1	Alcohol
2	Malic_acid
3	Ash
4	Alcalinity_of_ash
5	Magnesium
6	Total_phenols
7	Flavanoids
8	Nonflavanoid_phenols
9	Proanthocyanins
10	Color_intensity
11	Hue
12	OD280/OD315_of_diluted_wines
13	Proline
14	class

Remove

Status OK Log x 0

Selected attribute
Name: class
Missing: 0 (0%) Distinct: 3 Type: Nominal Unique: 0 (0%)

No.	Label	Count
1	1	59
2	2	71
3	3	48

Class: class (Nom) Visualize All

The chart displays the frequency distribution of the 'class' attribute. The x-axis represents the three distinct values of the class attribute (1, 2, 3), and the y-axis represents the count of instances for each value. The counts are 59 for class 1, 71 for class 2, and 48 for class 3.

[Preprocess](#) [Classify](#) [Cluster](#) [Associate](#) [Select attributes](#) [Visualize](#)

Classifier

[Choose](#)

IBk -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A {"weka.core.EuclideanDistance -R first-last}"

Test options

Use training set
 Supplied test set [Set...](#)
 Cross-validation Folds 10
 Percentage split % 66
[More options...](#)

(Nom) class

[Start](#) [Stop](#)

Result list (right-click for options)

15:25:57 - lazy.IBk

Classifier output

IB1 instance-based classifier
using 1 nearest neighbour(s) for classification

Time taken to build model: 0 seconds

== Stratified cross-validation ==

== Summary ==

Correctly Classified Instances	169	94.9438 %
Incorrectly Classified Instances	9	5.0562 %
Kappa statistic	0.9238	
Mean absolute error	0.0413	
Root mean squared error	0.1821	
Relative absolute error	9.3973 %	
Root relative squared error	38.8682 %	
Total Number of Instances	178	

== Detailed Accuracy By Class ==

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
1	0.042	0.922	1	0.959	0.983	0.983	1
0.873	0	1	0.873	0.932	0.941	0.941	2
1	0.031	0.923	1	0.96	0.983	0.983	3
Weighted Avg.	0.949	0.022	0.953	0.949	0.949	0.966	

== Confusion Matrix ==

a	b	c	<- classified as
59	0	0	a = 1
5	62	4	b = 2
0	0	48	c = 3

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier
Choose J48 -C 0.25 -M 2

Test options
 Use training set
 Supplied test set Set...
 Cross-validation Folds 10
 Percentage split % 66
More options...

(Nom) class
Start Stop

Result list (right-click for options)
15:30:46 - trees.J48

Classifier output

```
Number of Leaves : 5
Size of the tree : 9
Time taken to build model: 0.03 seconds
==== Stratified cross-validation ====
==== Summary ====
Correctly Classified Instances      167          93.8202 %
Incorrectly Classified Instances   11           6.1798 %
Kappa statistic                      0.9058
Mean absolute error                  0.0486
Root mean squared error              0.2019
Relative absolute error              11.0723 %
Root relative squared error         43.0865 %
Total Number of Instances            178

==== Detailed Accuracy By Class ====

      TP Rate    FP Rate    Precision    Recall    F-Measure    ROC Area    Class
      0.983     0.034      0.935      0.983      0.959      0.977      1
      0.944     0.056      0.918      0.944      0.931      0.937      2
      0.875     0.008      0.977      0.875      0.923      0.946      3
Weighted Avg.      0.938     0.036      0.94       0.938      0.938      0.953

==== Confusion Matrix ====

  a  b  c  <-- classified as
58  1  0  |  a = 1
 3 67  1  |  b = 2
 1  5 42  |  c = 3
```

Sometimes simpler is better...

Ensembling and Random Forests...



Ensemble Methods

Bagging

- Select many independent samples from training data, *with replacement...*
- Learn one model on each sample (training set), can be done in parallel
- Classifier: Majority voting of the various models

Boosting

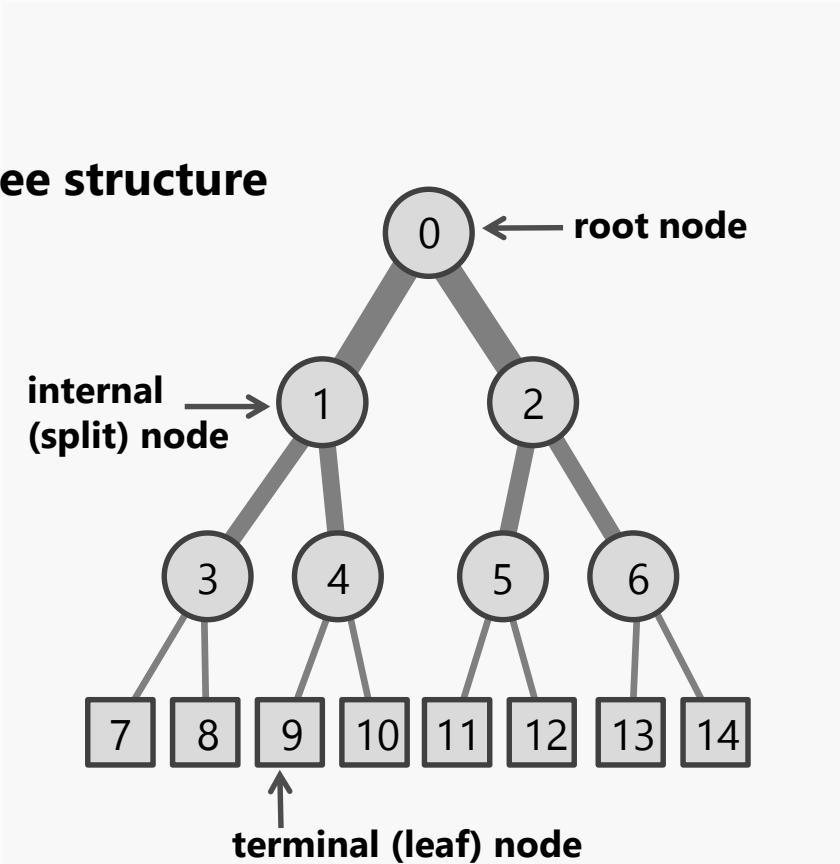
- Learns a sequence of models
- Each model focuses on examples that previous models did badly on
- Learns a weighting of the models

Decision Trees:

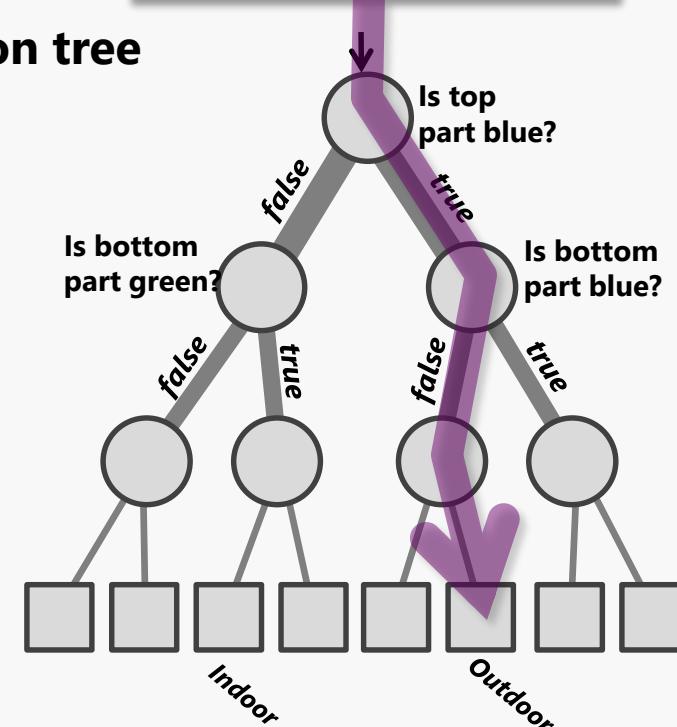
- With **bagging** (+random selection of m features to use with each node):
Random Forest
- With **boosting**: Boosted Decision Trees (Bing ranking of search results)

Decision Trees and Decision Forests

A general tree structure



A decision tree



A forest is an ensemble of trees. The trees are all slightly different from one another.

Decision Forest Model: the randomness model

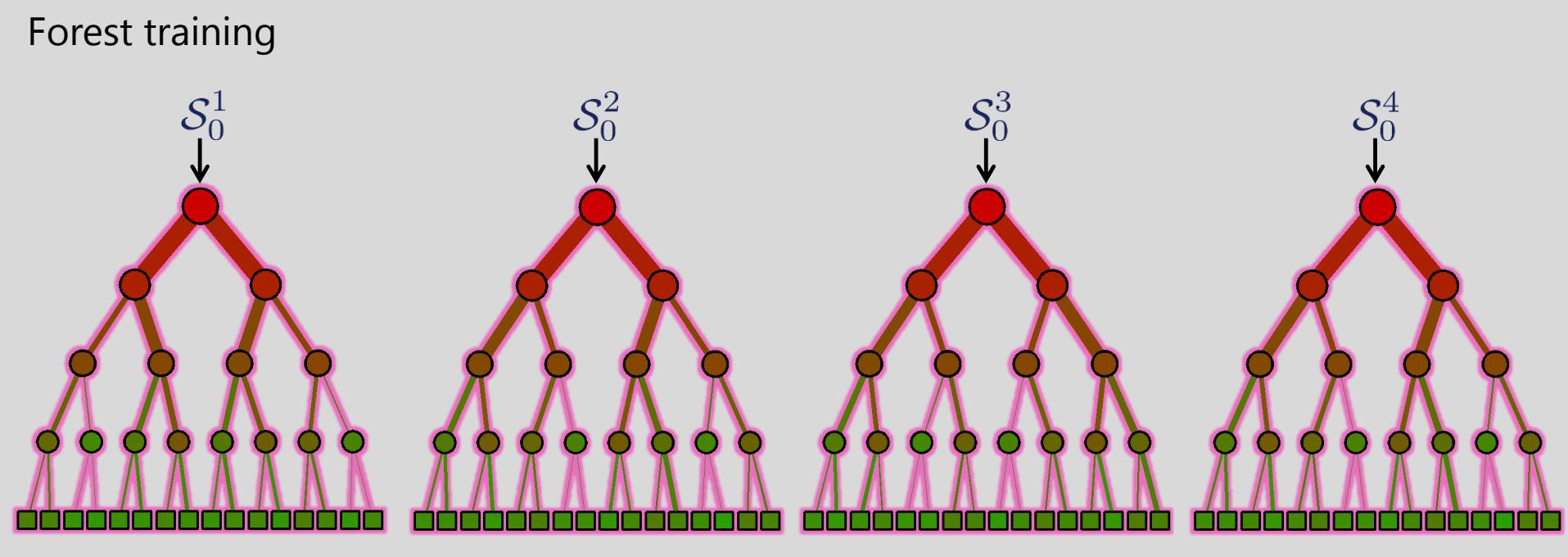
1) Bagging (randomizing the training set)

\mathcal{S}_0

The full training set

$\mathcal{S}_0^t \subset \mathcal{S}_0$

The randomly sampled subset of training data made available for the tree t



Decision Forest Model: the randomness model

2) Randomized node optimization (RNO)

 \mathcal{T}

The full set of all possible node test parameters

 $\mathcal{T}_j \subset \mathcal{T}$

For each node the set of randomly sampled features

 $\rho = |\mathcal{T}_j|$

Randomness control parameter.

For $\rho = |\mathcal{T}|$ no randomness and maximum tree correlation.

For $\rho = 1$ max randomness and minimum tree correlation.

Node training

Node weak learner

$$h(\mathbf{v}, \boldsymbol{\theta}_j)$$

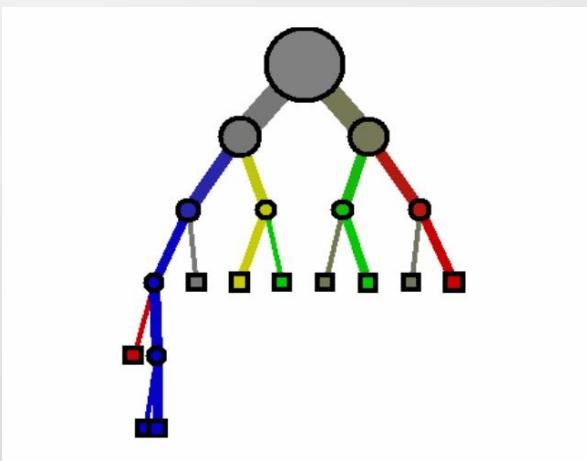
Node test params

$$\boldsymbol{\theta} \in \mathcal{T}_j$$

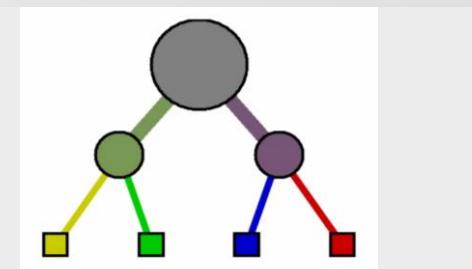


The effect of ρ

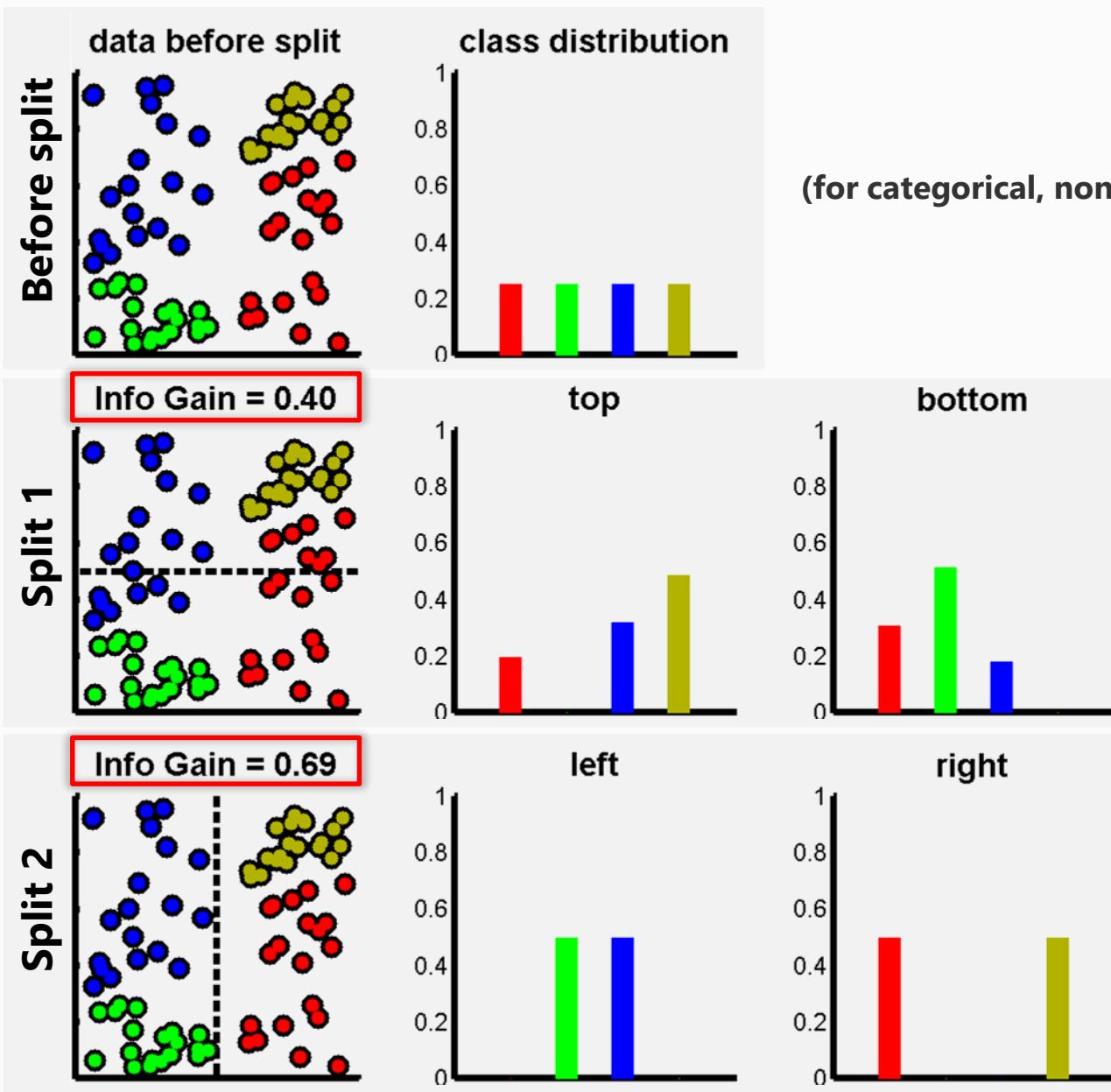
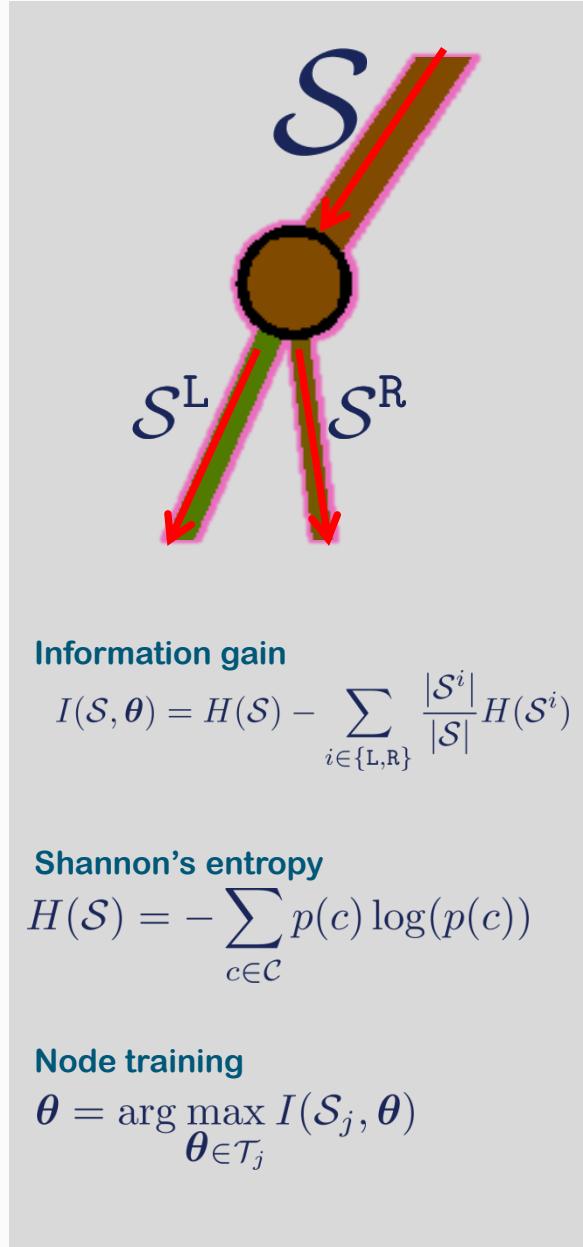
Small value of ρ ; little tree correlation.



Large value of ρ ; large tree correlation.

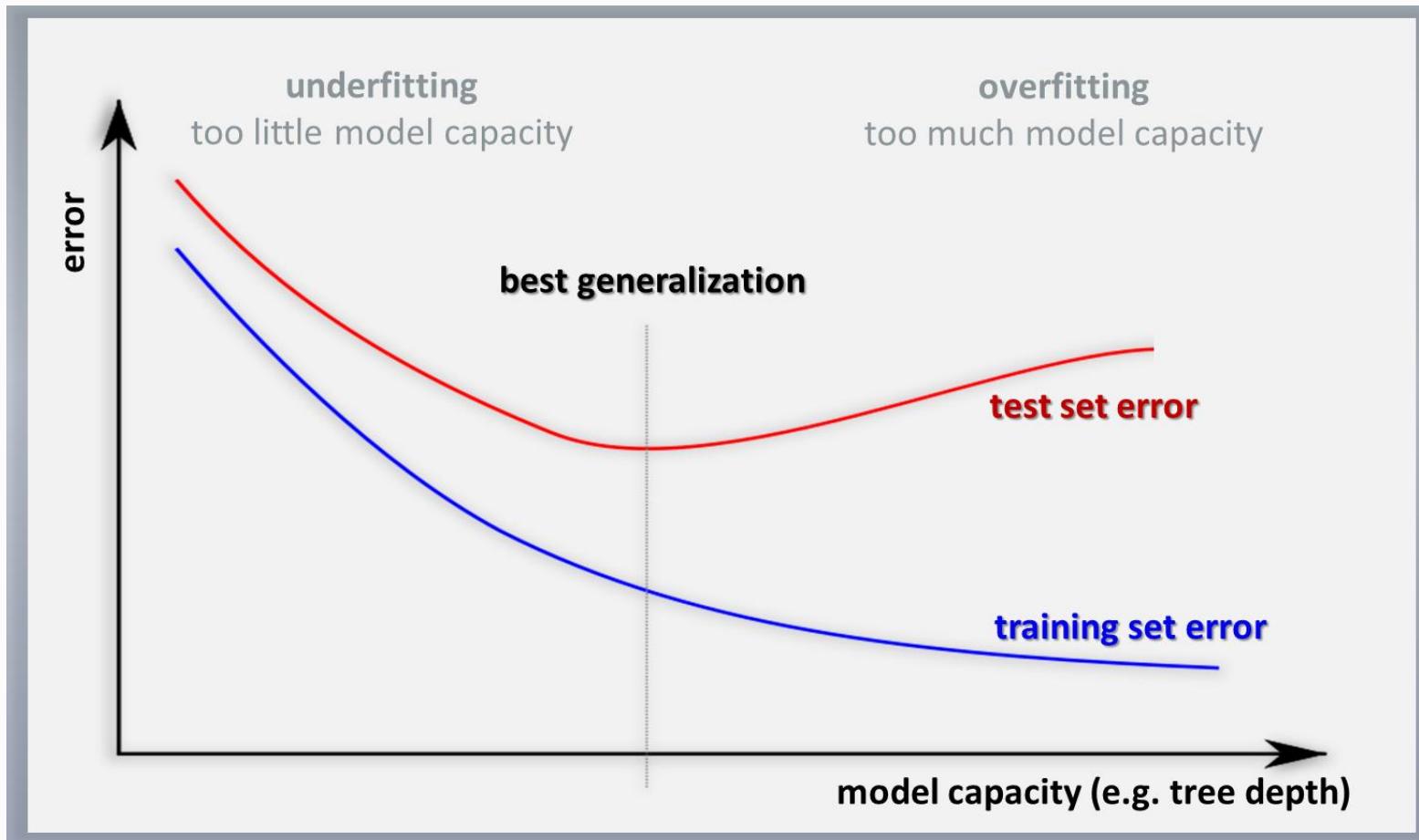


Decision Forest Model: training and information gain

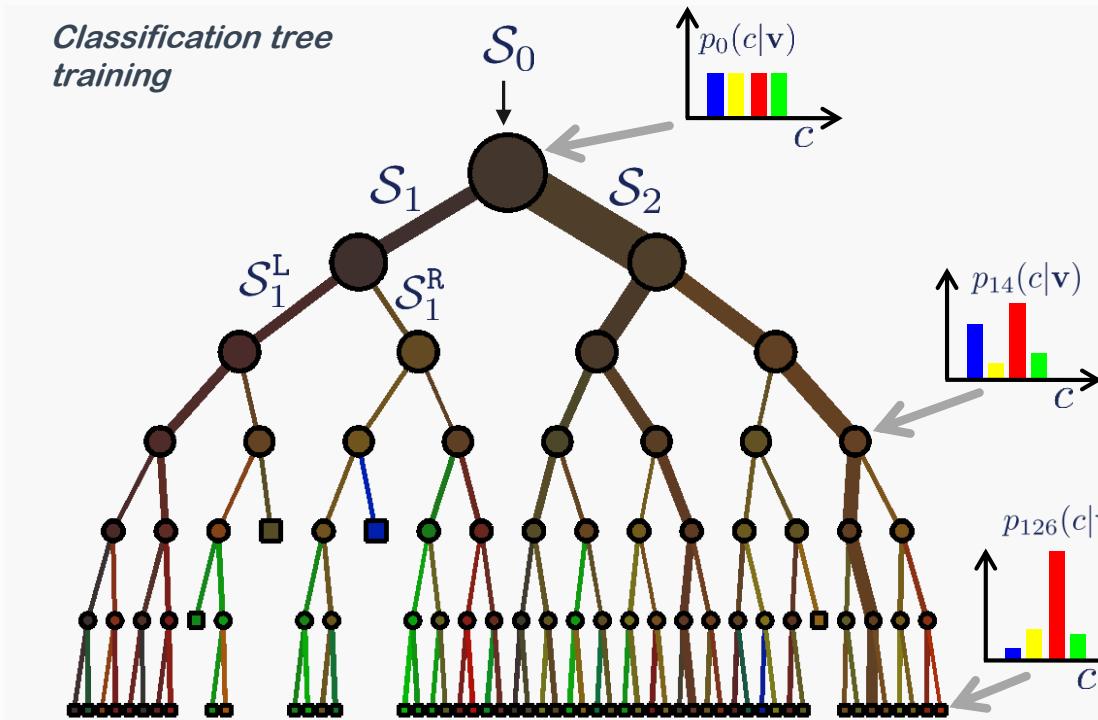
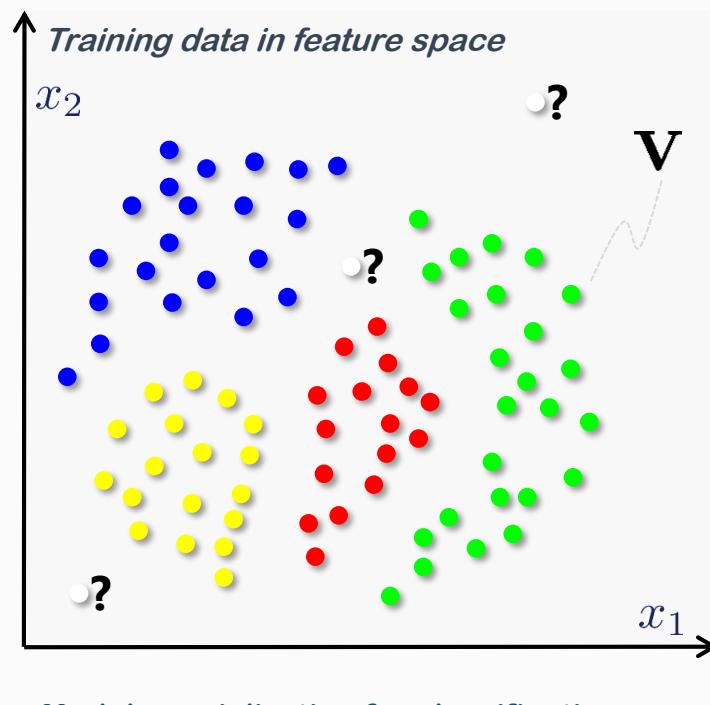


(for categorical, non-parametric distributions)

Why we prune...



Classification Forest



Model specialization for classification

Input data point $\mathbf{v} = (x_1, \dots, x_d) \in \mathbb{R}^d$ (x_i is feature response)

Output is categorical $c \in \mathcal{C}$ with $\mathcal{C} = \{c_k\}$ (discrete set)

Node weak learner $h(\mathbf{v}, \theta) \in \{\text{true, false}\}$

Obj. funct. for node j $I = H(S_j) - \sum_{i=L,R} \frac{|S_j^i|}{|S_j|} H(S_j^i)$ (information gain)

Training node j $\theta_j = \arg \max_{\theta \in \mathcal{T}_j} I(S_j, \theta)$

Predictor model $p(c|\mathbf{v}) = \sum_j p(c|j)p(j|\mathbf{v})$ (class posterior)

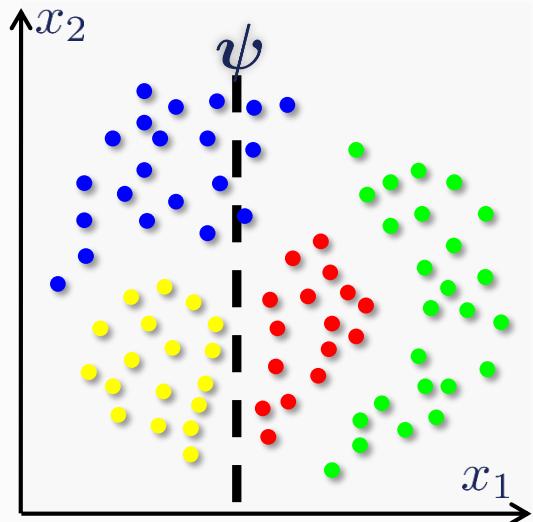
Entropy of a discrete distribution

$$H(\mathcal{S}) = - \sum_{c \in \mathcal{C}} p(c) \log(p(c))$$

$$\text{with } c(\mathbf{v}) : \mathbb{R}^d \rightarrow \mathcal{C}$$

Classification Forest: the weak learner model

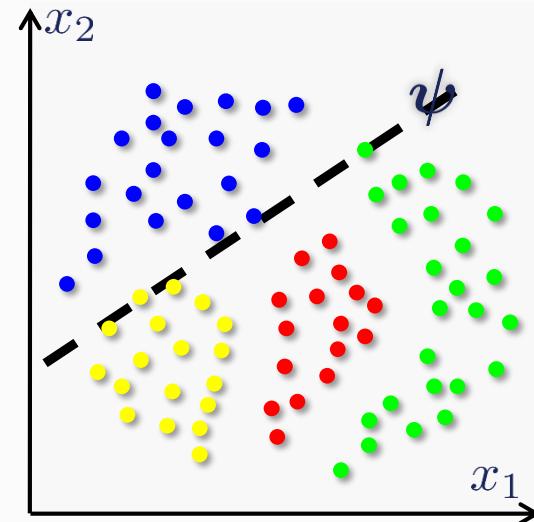
Examples of weak learners



Weak learner: axis aligned

$$h(\mathbf{v}, \theta) = [\tau_1 > \phi(\mathbf{v}) \cdot \psi > \tau_2]$$

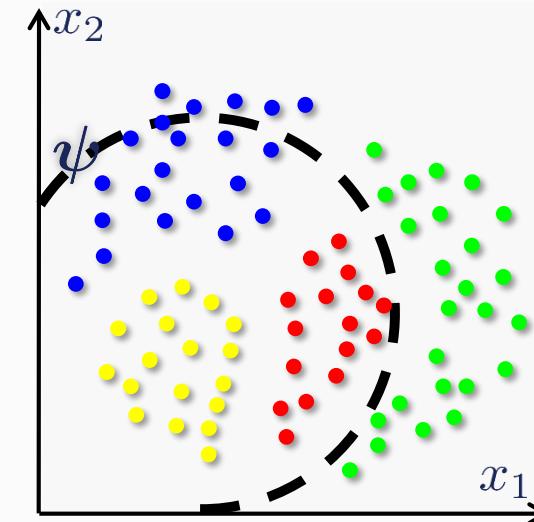
Feature response
for 2D example.
With $\psi = (1 \ 0 \ \psi_3)$ or $\psi = (0 \ 1 \ \psi_3)$



Weak learner: oriented line

$$h(\mathbf{v}, \theta) = [\tau_1 > \phi(\mathbf{v}) \cdot \psi > \tau_2]$$

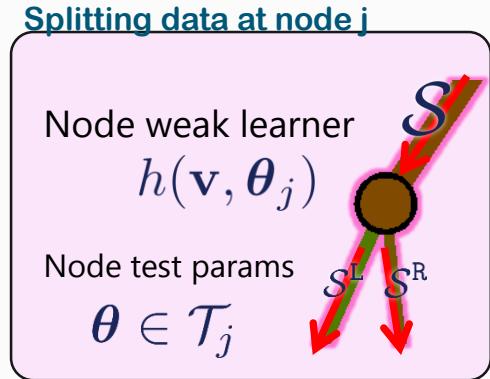
Feature response
for 2D example.
With $\psi \in \mathbb{R}^3$ a generic line in homog. coordinates.



Weak learner: conic section

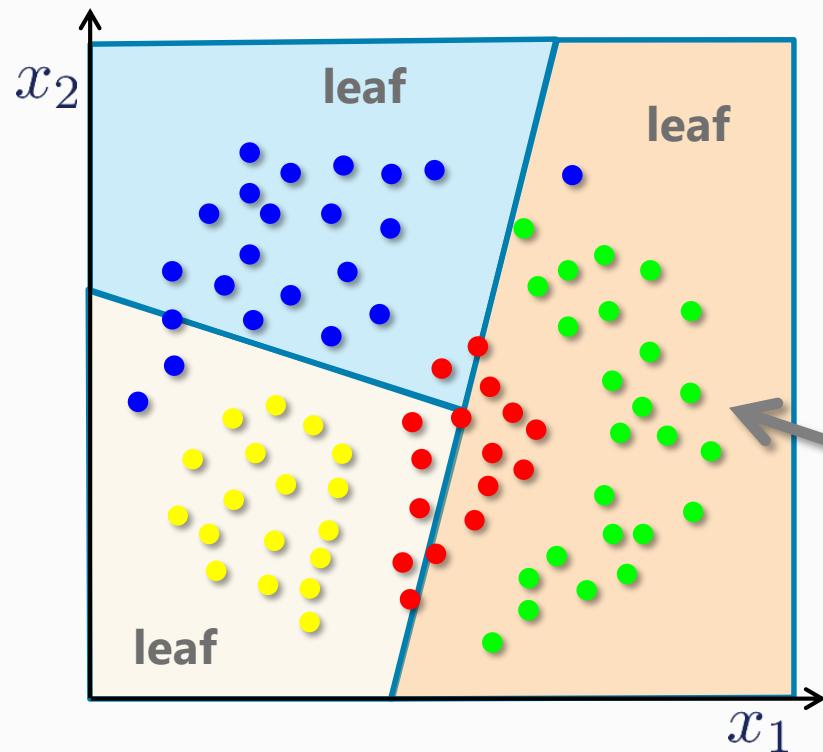
$$h(\mathbf{v}, \theta) = [\tau_1 > \phi^\top(\mathbf{v}) \psi \phi(\mathbf{v}) > \tau_2]$$

Feature response
for 2D example.
With $\psi \in \mathbb{R}^{3 \times 3}$ a matrix representing a conic.

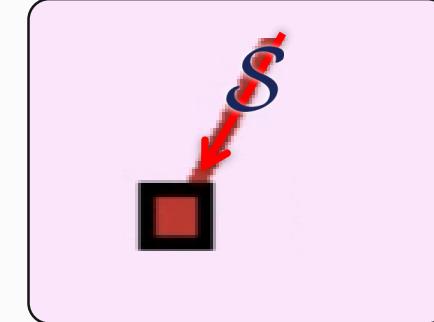


In general ϕ may select only a very small subset of features $\phi(\mathbf{v}) : \mathbb{R}^d \rightarrow \mathbb{R}^{d'+1}, d' \ll d$

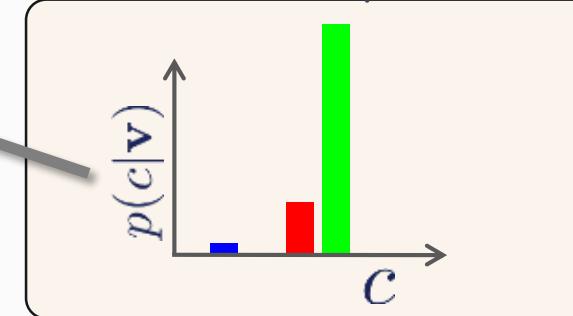
Classification Forest: the prediction model



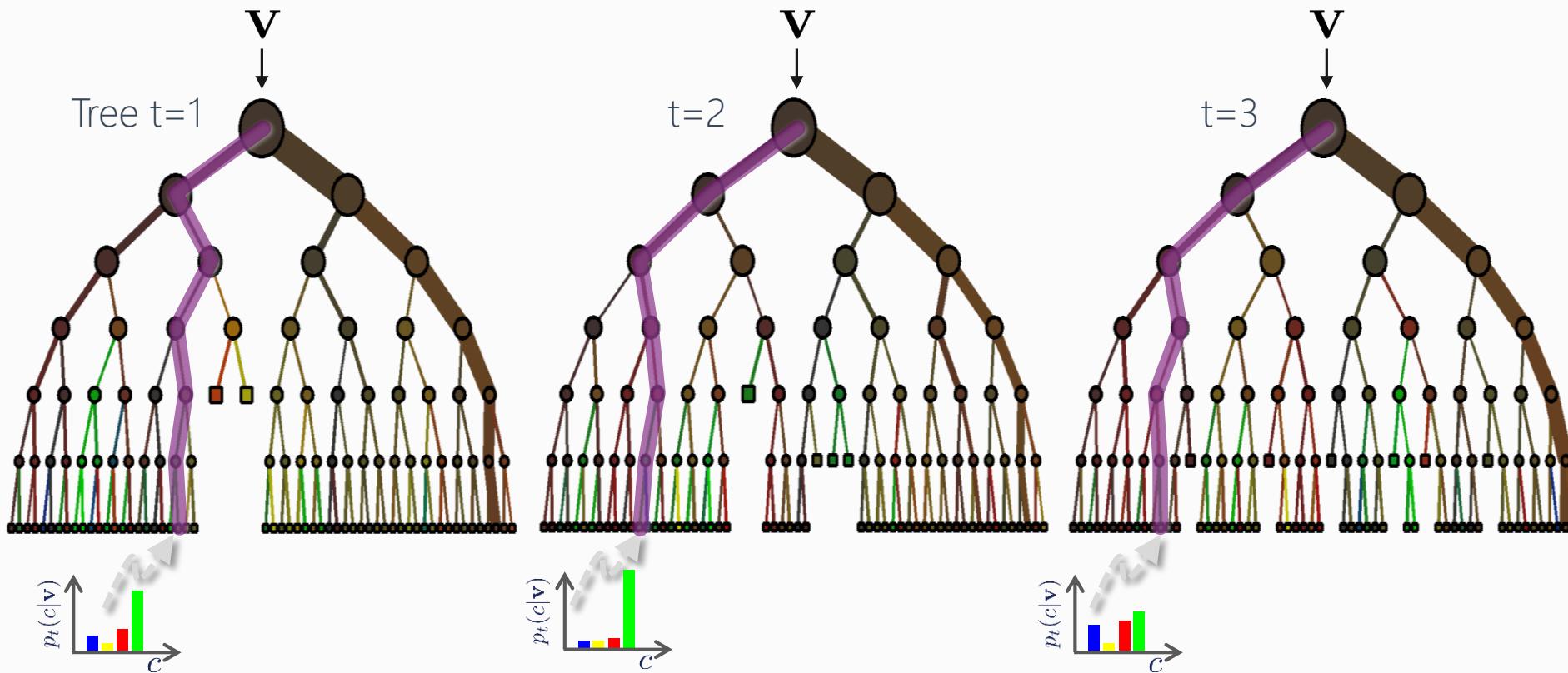
What do we do at the leaf?



Prediction model: probabilistic

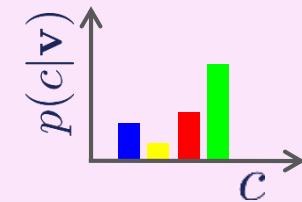


Classification Forest: the ensemble model

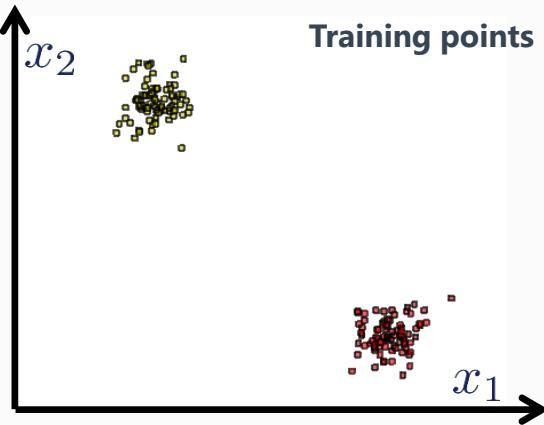


The ensemble model

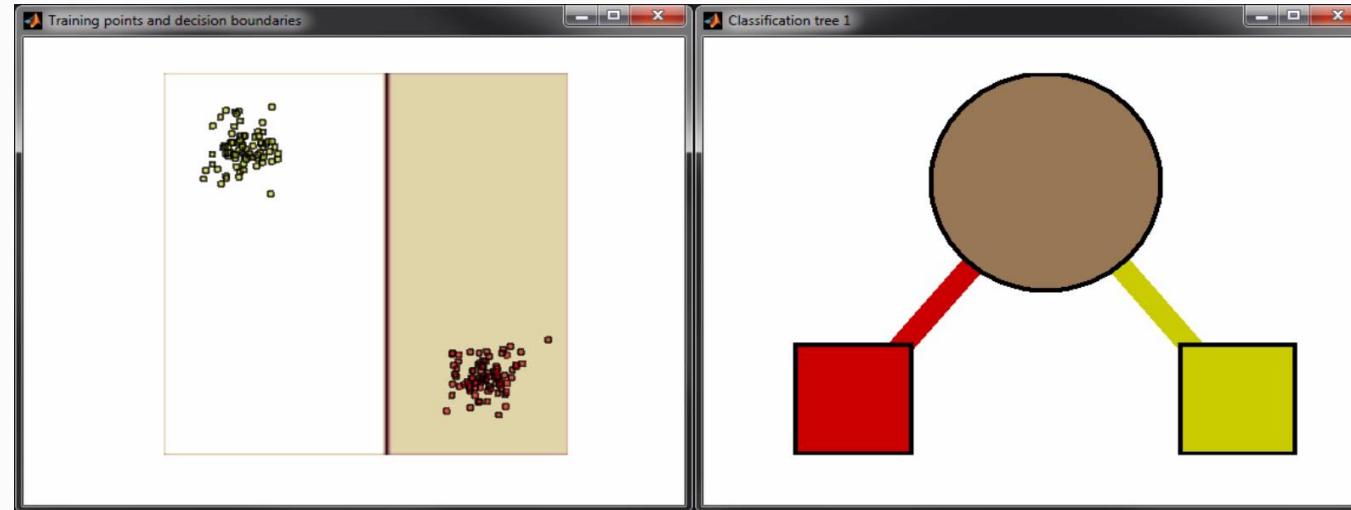
$$\text{Forest output probability} \quad p(c|\mathbf{v}) = \frac{1}{T} \sum_t p_t(c|\mathbf{v})$$



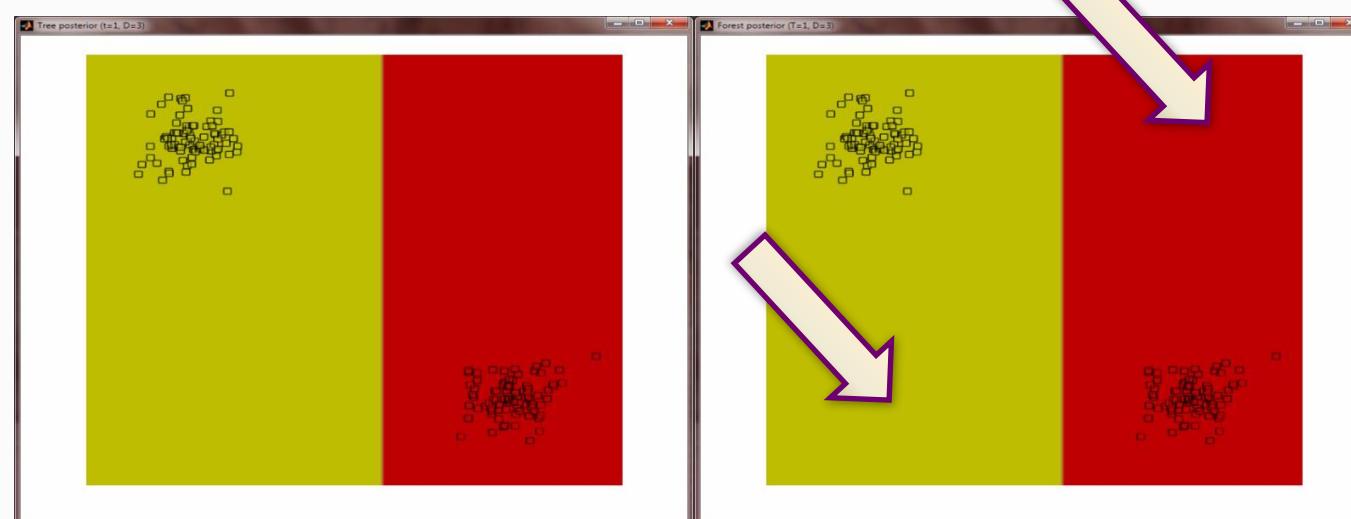
Classification Forest: effect of the weak learner model



Training different trees in the forest



Testing different trees in the forest



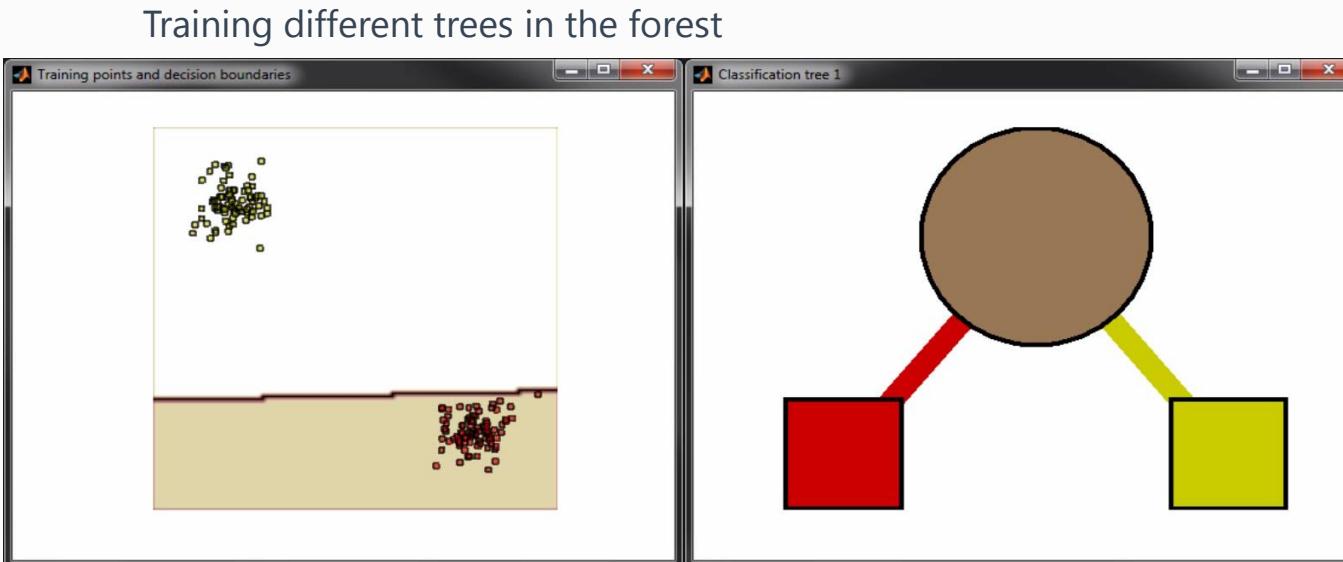
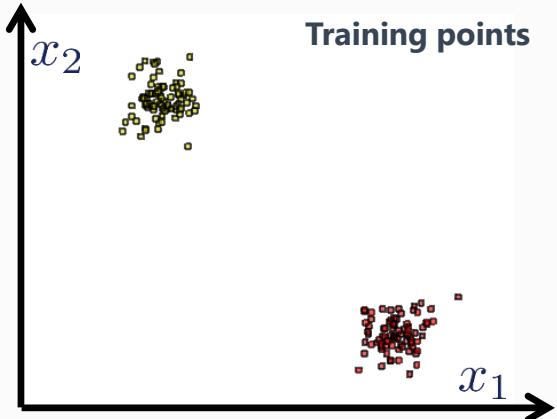
Three concepts to keep in mind:

- "Accuracy of prediction"
- "Quality of confidence"
- "Generalization"

(2 videos in this page)

Parameters: T=200, D=2, weak learner = aligned, leaf model = probabilistic

Classification Forest: effect of the weak learner model

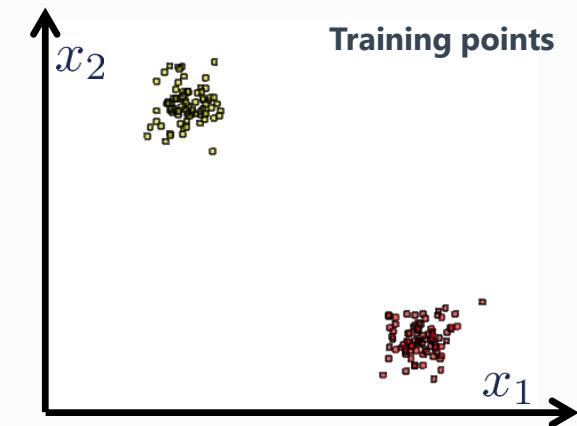


Parameters: T=200, D=2, weak learner = linear, leaf model = probabilistic

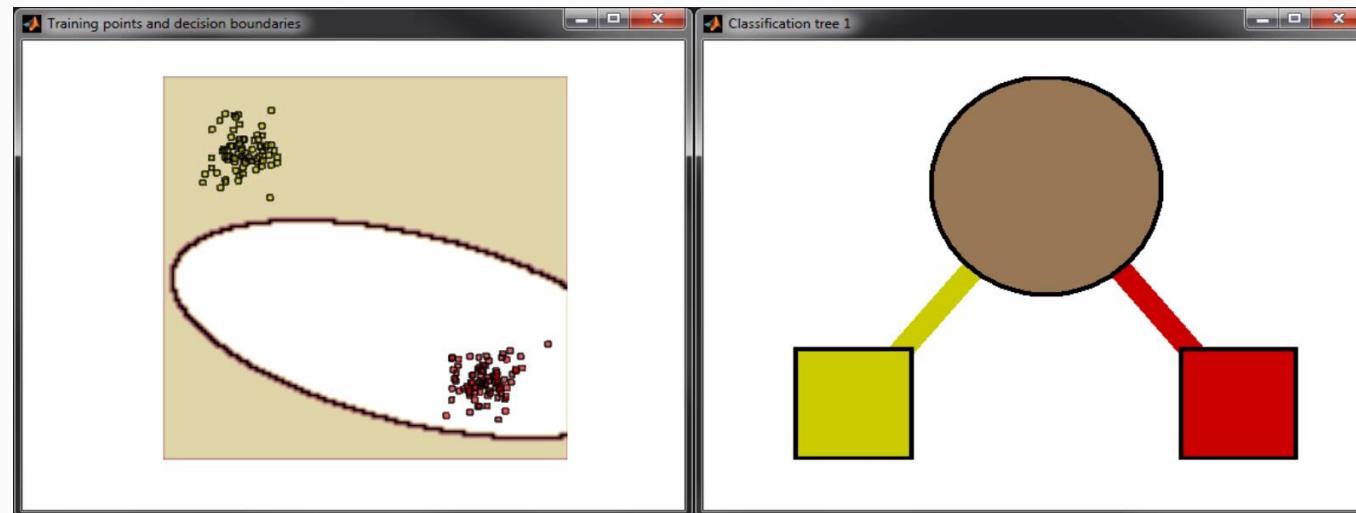
Deriving Knowledge from Data at Scale



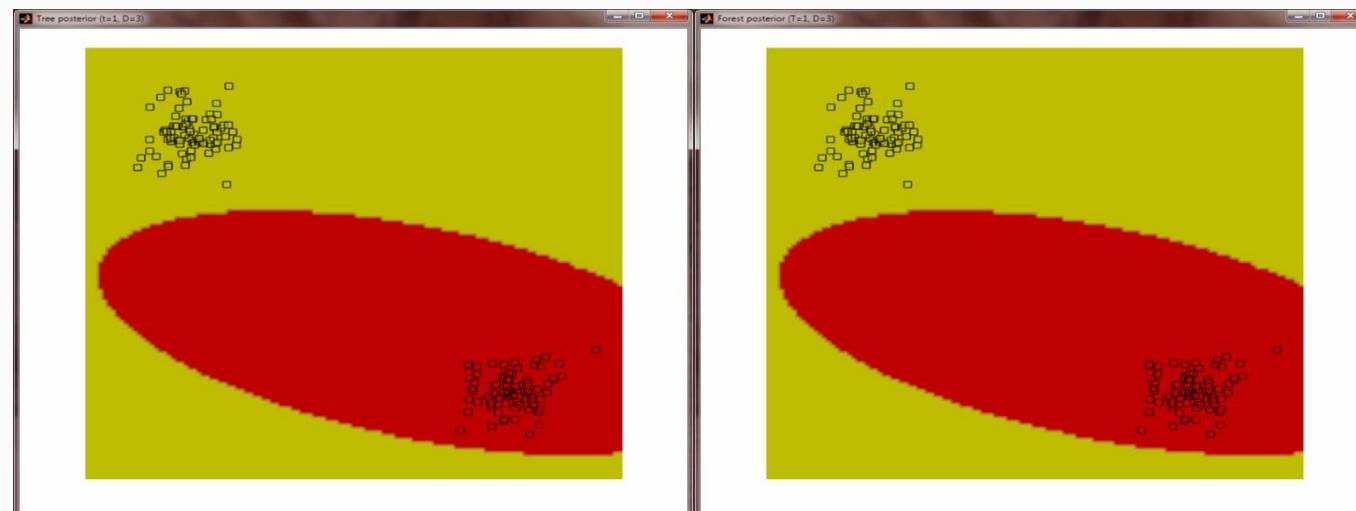
Classification Forest: effect of the weak learner model



Training different trees in the forest



Testing different trees in the forest



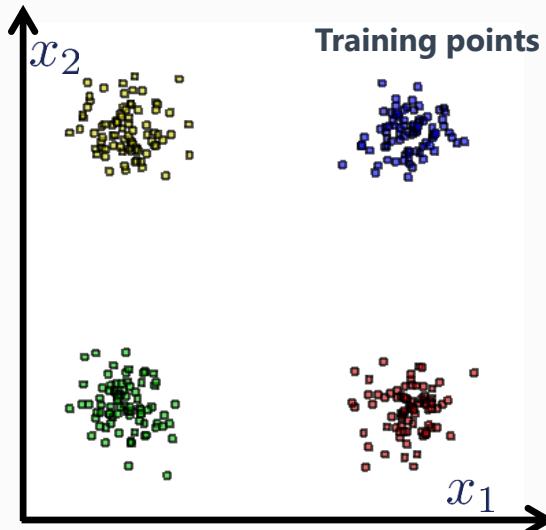
Parameters: T=200, D=2, weak learner = conic, leaf model = probabilistic

(2 videos in this page)

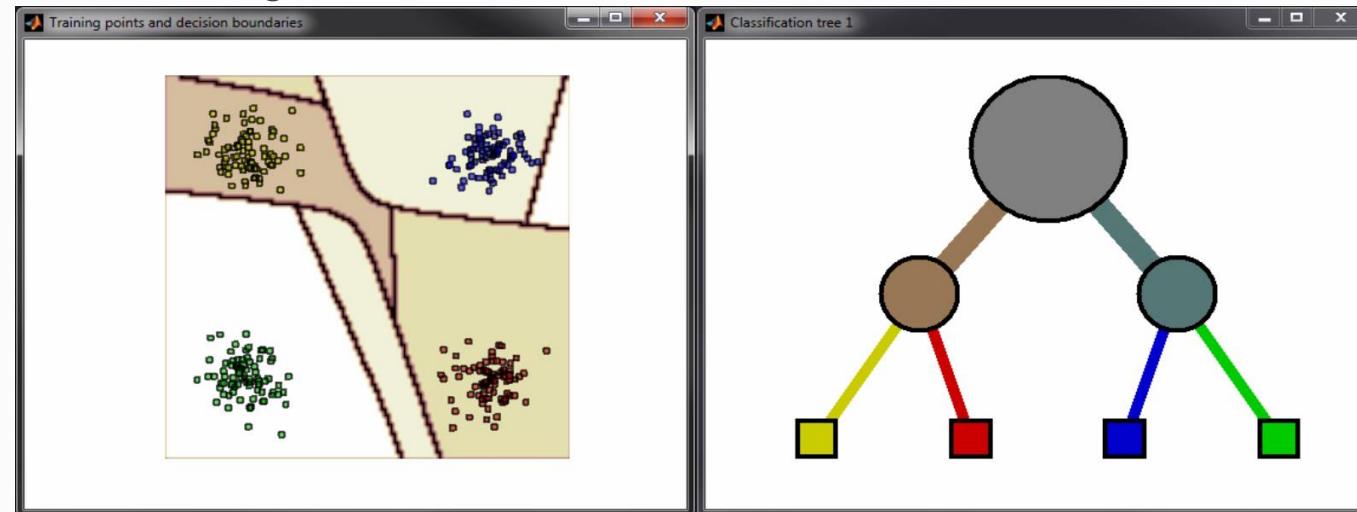
Deriving Knowledge from Data at Scale



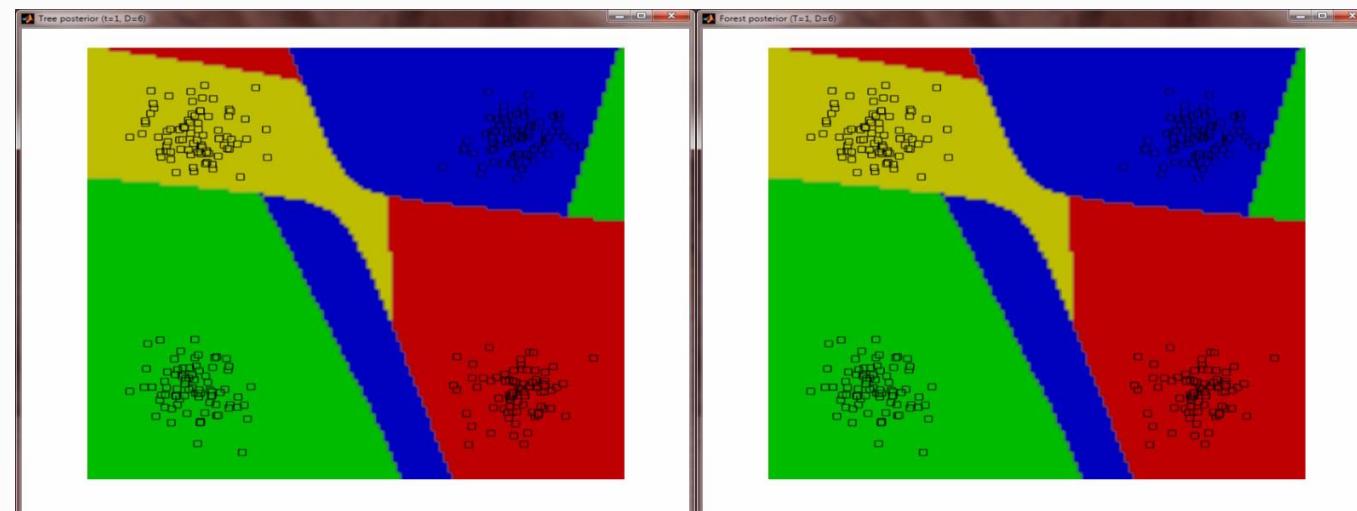
Classification Forest: with >2 classes



Training different trees in the forest



Testing different trees in the forest



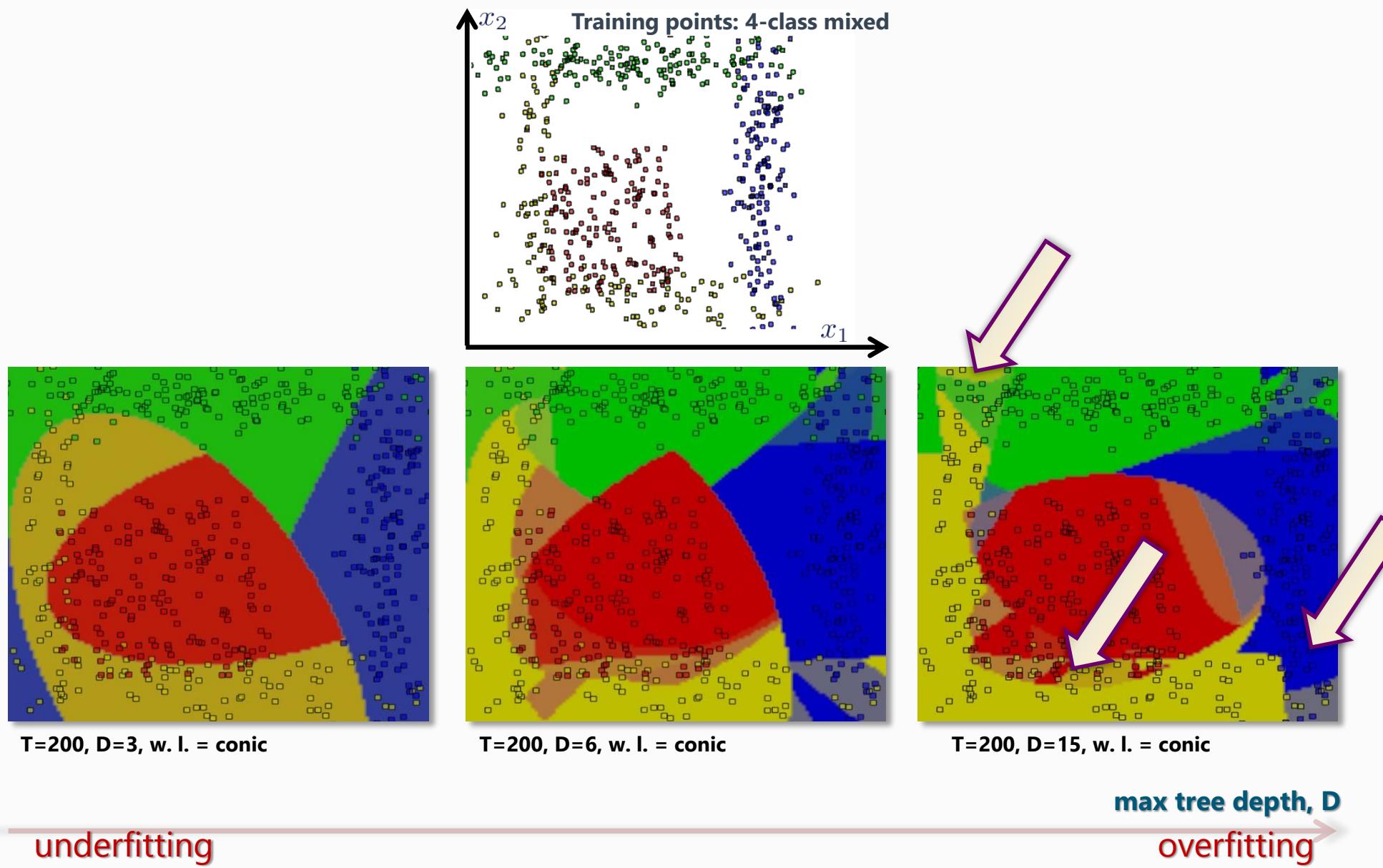
(2 videos in this page)

Parameters: T=200, D=3, weak learner = conic, leaf model = probabilistic

Deriving Knowledge from Data at Scale



Classification Forest: effect of tree depth



Data Characterization...

Data Characterization

1. Unique values
2. Most frequent values
3. Highest and lowest values
4. Location and dispersion – gini, statistical test for dispersion
5. Quartiles

Data Cleaning

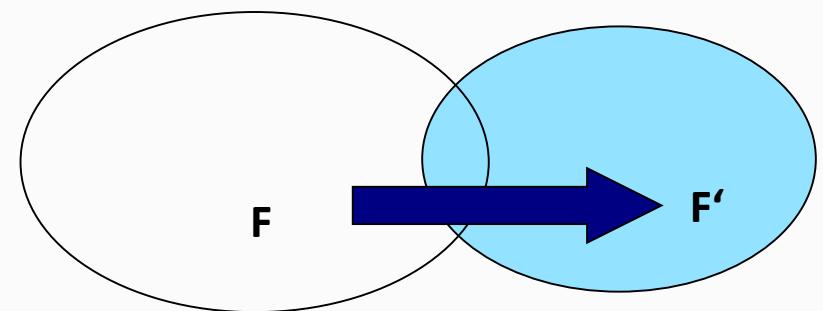
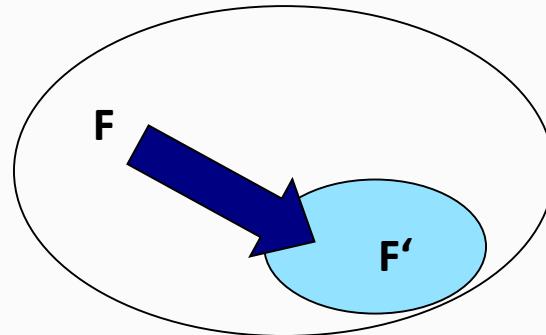
1. Missing values
2. Outliers
3. Coding

Feature Selection

Simple Definition

Given a set of features $F = \{f_1, \dots, f_i, \dots, f_n\}$ the feature selection problem is to find a subset $F' \subseteq F$ that “*maximizes the learners ability to classify patterns*”. Feature extraction creates new features $F \rightarrow F'$...

$$\{f_1, \dots, f_i, \dots, f_n\} \xrightarrow{f.\text{selection}} \{f_{i_1}, \dots, f_{i_j}, \dots, f_{i_m}\}$$

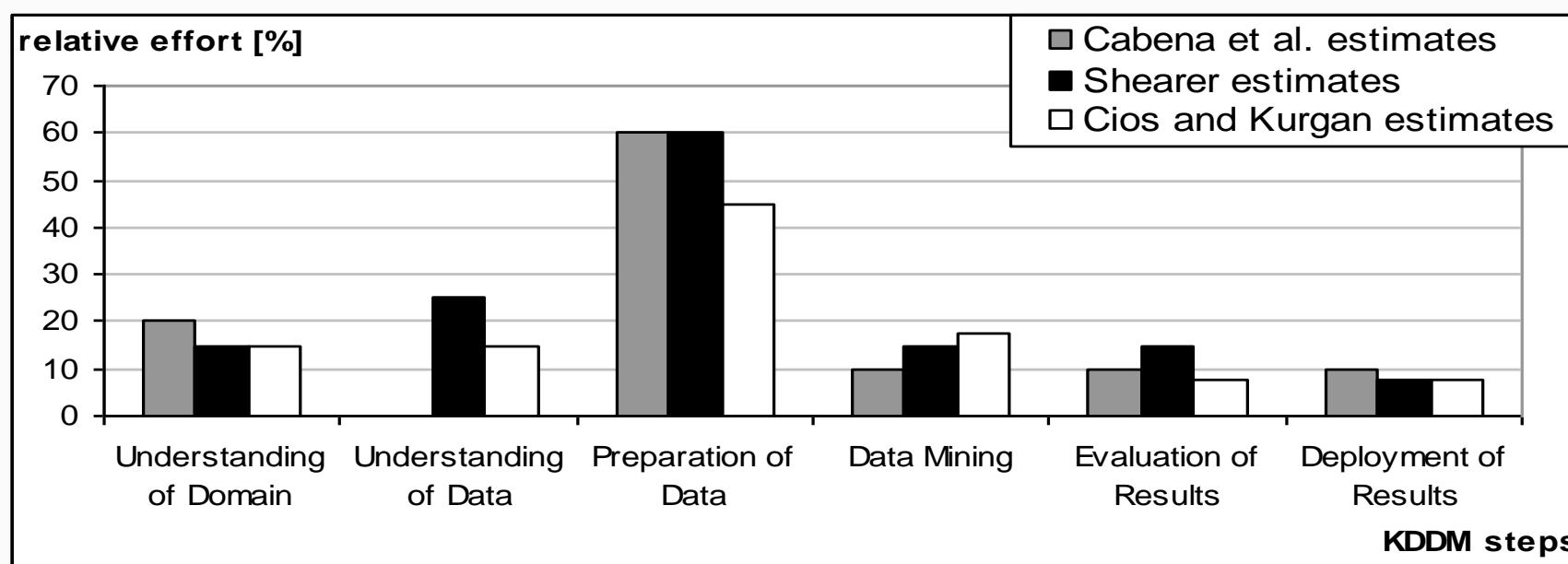


$$\{f_1, \dots, f_i, \dots, f_n\} \xrightarrow{f.\text{extraction}} \{g_1(f_1, \dots, f_n), \dots, g_j(f_1, \dots, f_n), \dots, g_m(f_1, \dots, f_n)\}$$

Time Spent on KDD Components

An important aspect of the KDD contest is relative time to complete each step

- It enables precise scheduling
- Estimates by researchers and practitioners are shown below
 - Specific estimated values depend on existing knowledge about the domain, skill level of the humans, complexity of the problem, etc.
 - Data preparation step is by far the most time consuming step



Out of Class Reading, highly recommended

A Study on the Importance of and Time Spent on Different Modeling Steps

M. Arthur Munson
Sandia National Laboratories
Livermore, CA 94551, USA
mamunso@sandia.gov

ABSTRACT

Applying data mining and machine learning algorithms requires many steps to prepare data and to make use of modeling results. This study investigates two questions: (1) how time consuming are the pre- and post-processing steps? (2) how much research energy is spent on these steps? To answer these questions I surveyed practitioners about their experiences in applying modeling techniques and categorized data mining and machine learning research papers from 2009 according to the modeling step(s) they addressed. Survey results show that model building consumes only 14% of the time spent on a typical project; the remaining time is spent on pre- and post-processing steps. Both survey responses and the categorization of research papers show that data mining and machine learning researchers spend the majority of their energy on algorithms for constructing models and significantly less energy on other steps. These findings collectively suggest that there are research opportunities to simplify the steps that precede and follow model building.

I. INTRODUCTION

In this paper I investigate how time consuming the various modeling steps are for practitioners and how much research effort is focused on each step. To answer these questions I surveyed practitioners about their experiences in applying data mining (machine learning) techniques and manually categorized the 2009 proceedings from two of the top conferences in the area (ICML 2009¹ and KDD 2009²) based on the step(s) they addressed.

There are two main findings in this study. First, in the typical project only 14% of the time is spent building the model. The rest of the time is spent preparing to do model learning and verifying the results after model construction. In contrast, the data mining and machine learning research communities spend the majority of their energy on how to learn a model from data, and moderate energy or less on other modeling steps. This finding is supported both by survey responses and by the distribution of papers at ICML 2009 and KDD 2009. In addition to documenting the current state of practice and research, these findings suggest that there are research opportunities to simplify the steps before and after model building. Such improvements would greatly benefit practitioners and should facilitate further adoption



Out of Class Reading, highly recommended

Journal of Machine Learning Research 3 (2003) 1157-1182

Submitted 11/02; Published 3/03

An Introduction to Variable and Feature Selection

Isabelle Guyon

*Clopinet
955 Creston Road
Berkeley, CA 94708-1501, USA*

ISABELLE@CLOPINET.COM

André Elisseeff

*Empirical Inference for Machine Learning and Perception Department
Max Planck Institute for Biological Cybernetics
Spemannstrasse 38
72076 Tübingen, Germany*

ANDRE@TUEBINGEN.MPG.DE

Editor: Leslie Pack Kaelbling

Abstract

Variable and feature selection have become the focus of much research in areas of application for which datasets with tens or hundreds of thousands of variables are available. These areas include text processing of internet documents, gene expression array analysis, and combinatorial chemistry. The objective of variable selection is three-fold: improving the prediction performance of the predictors, providing faster and more cost-effective predictors, and providing a better understanding of



1. **Do you have domain knowledge?** If yes, construct a better set of "ad hoc" features.
2. **Are your features commensurate?** If no, consider normalizing them.
3. **Do you suspect interdependence of features?** If yes, expand your feature set by constructing conjunctive features or products of features, as much as your computer resources allow you (see example in Section 4.4).
4. **Do you need to prune the input variables** (e.g. for cost, speed or understanding reasons)? If no, construct disjunctive features or weighted sums of features (e.g. by clustering or matrix factorization, see Section 5).
5. **Do you need to assess features individually** (e.g. to understand their influence on the system or because their number is so large that you need to first filter)? If yes, use a variable ranking method (Sect 2 and Sect 7.2); else, do it anyway to get baseline results.
6. **Do you need a predictor?** If no, stop.
7. **Do you suspect your data is "dirty"** (has a few meaningless input patterns and/or noisy outputs or wrong class labels)? If yes, detect outlier examples using the top ranking variables obtained in step 5 as representation; check and/or discard them.
8. **Do you know what to try first?** If no, use a linear predictor. Following the ranking of step 5, construct a sequence of predictors of same nature using increasing subsets of features. Can you match or improve performance with a smaller subset? If yes, try a non-linear predictor with that subset.
9. **Do you have new ideas, time, computational resources, and enough examples?** If yes, compare several feature selection methods, including your new idea, correlation coefficients, backward selection and embedded methods (Sect 4). Use linear and non-linear predictors. Select the best model (Section 6).
10. **Do you want a stable solution** (to improve performance and/or understanding)? If yes, subsample your data and redo your analysis for several "bootstraps" (Section 7.1).

Motivation: Real world examples

Example (1)

- Astro-particle physics dataset
 - Binary classification
 - 4 features; Training set: 3000; Test set: 4000
 - Direct use of SVMs gives 67% accuracy on Test set

Lesson: Correct data transformation is important!

- Using proper data transformation gives 96% accuracy on test set
- Proper tuning of hyper-parameters gives 97% accuracy on test set

Motivation: Real world examples

Example (2): KDD Cup 2001

- Detect binding of proteins to Thrombin
 - Binary classification (imbalanced: only 2% positive)
 - 139,351 features; Training set: 1951; Test set: 636

Lesson: A model that uses lots of features can turn out to be very sub-optimal, however well it is designed!

- Winner selected a subset of 200 features, trained a decision tree and got 68.44% mean accuracy
- Post-competition, one SVM team trained an SVM with a similar feature selection to get 83% mean accuracy.

Motivation: Real world examples

Example (3)

Lesson: Feature selection can be crucial even when the number of features is small!

Binary classification

- 60 features; Training set: 1000; Test set: 2175
- Direct use of SVMs gives 13% error rate on Test set
- Using proper feature selection gives 5% error rate on test set

Motivation: Real world examples

Example (4)

- German credit dataset (Predict a customer's credit worthiness)
 - Binary classification

Lesson: Variations of the same ML method can give vastly different performances!

- Decision tree design in *Knime* gives 32.1% error rate on Test set
- Decision tree design in *tLC* gives 24% error rate on Test set

Kaggle



kaggle

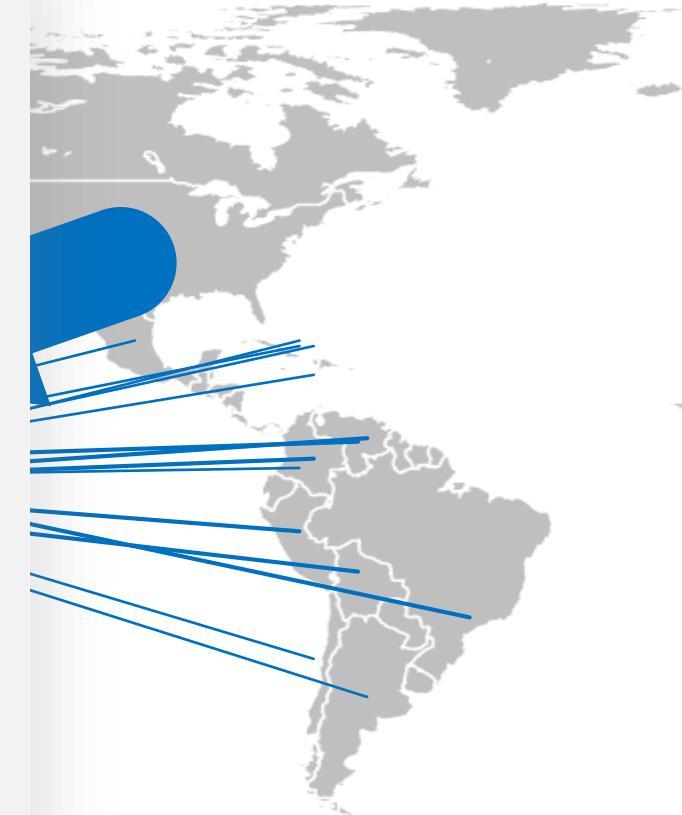
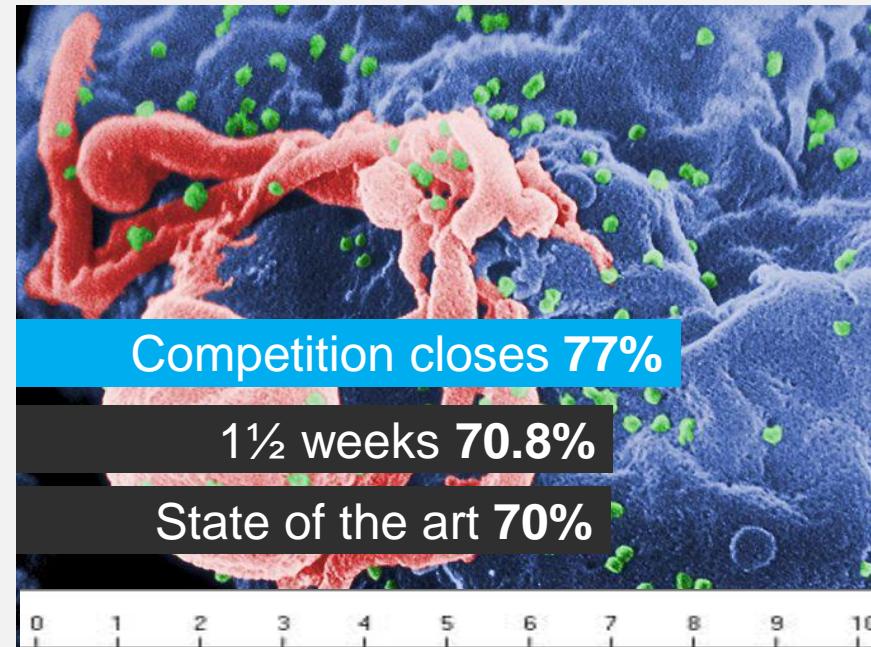
The background of the image shows a clear blue sky with several dark-colored geese flying in various directions. Some geese are in sharp focus, while others are blurred, creating a sense of motion. A large, solid blue rectangular box is positioned in the lower-left quadrant of the image, containing the word "Predictive modeling competitions".

Predictive modeling competitions

Global competitions



Predicting HIV viral load
Improved by 10%



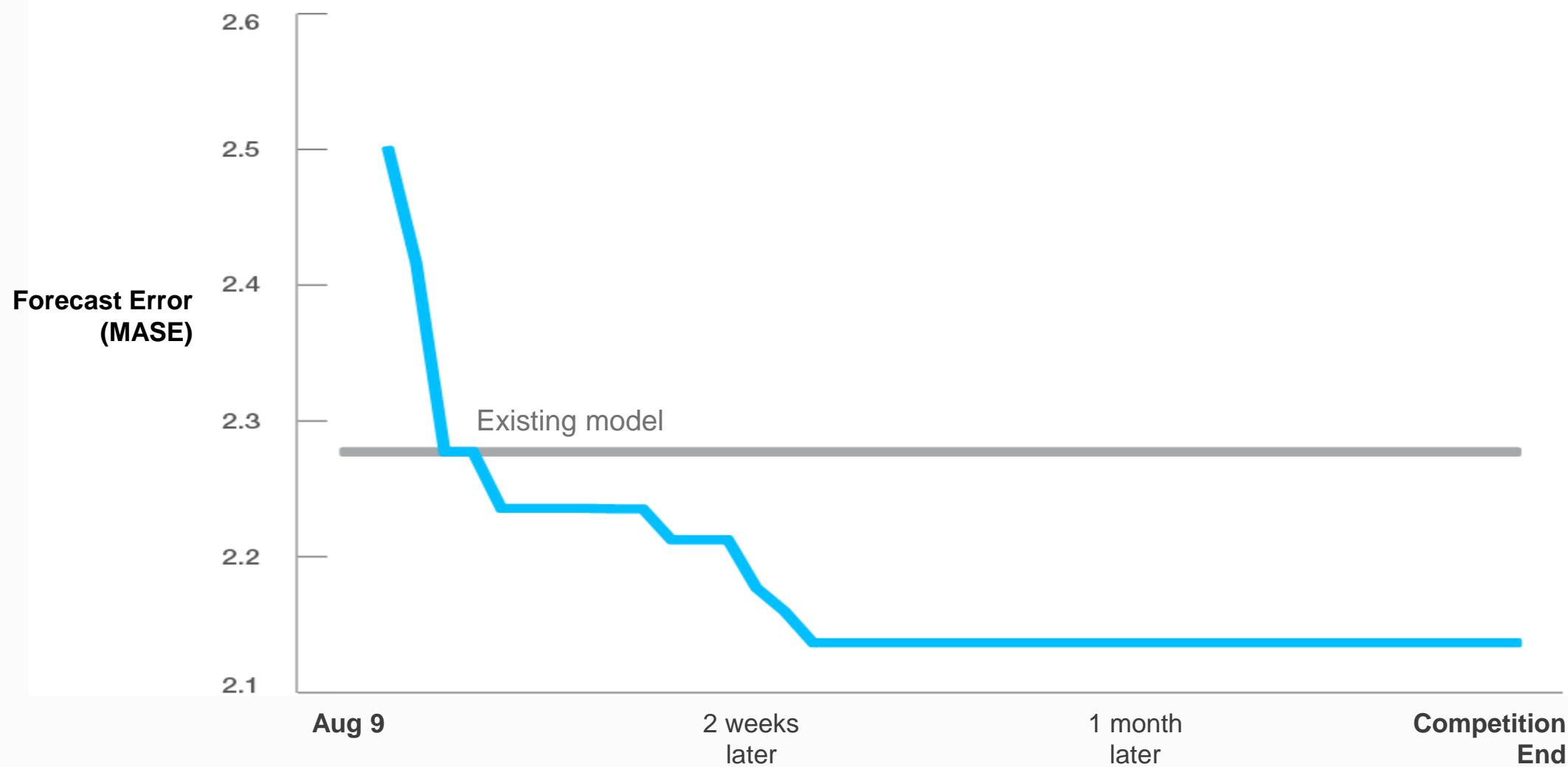


Crowdsourcing

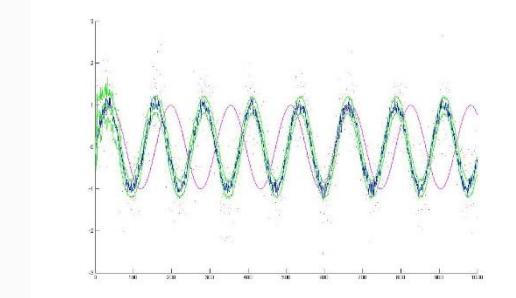
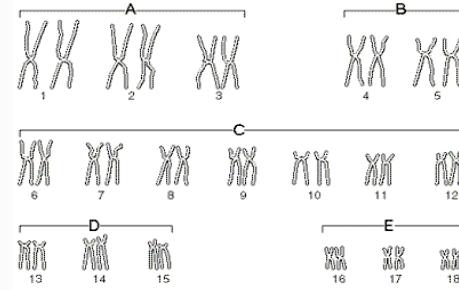
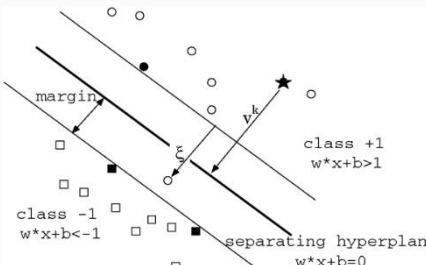
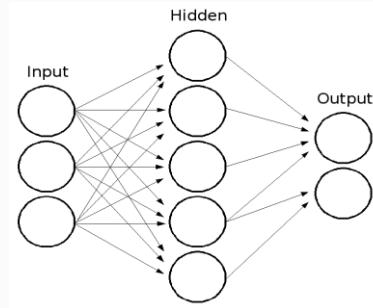


**Mismatch between those with data and
those with the skills to analyse it**

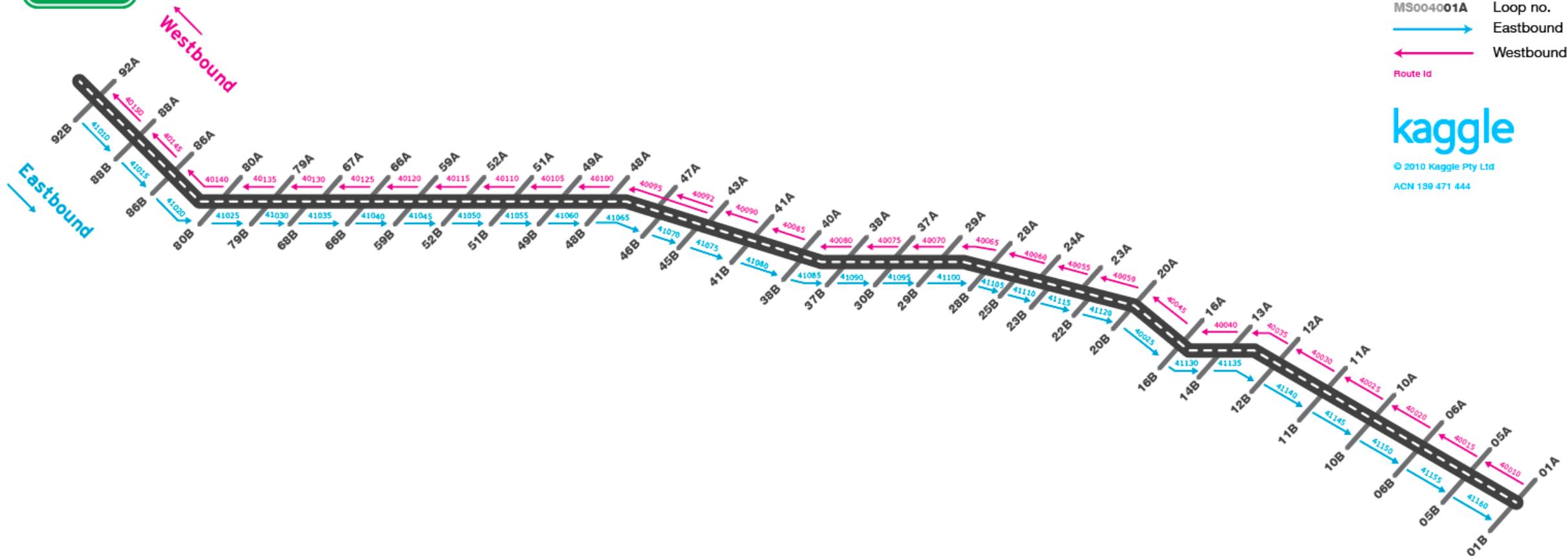
Tourism Forecasting Competition



Users apply different techniques



- neural networks
- logistic regression
- support vector machine
- decision trees
- ensemble methods
- adaBoost
- Bayesian networks
- genetic algorithms
- random forest
- Monte Carlo methods
- principal component analysis
- Kalman filter
- evolutionary fuzzy modeling

M4**NOT TO SCALE**

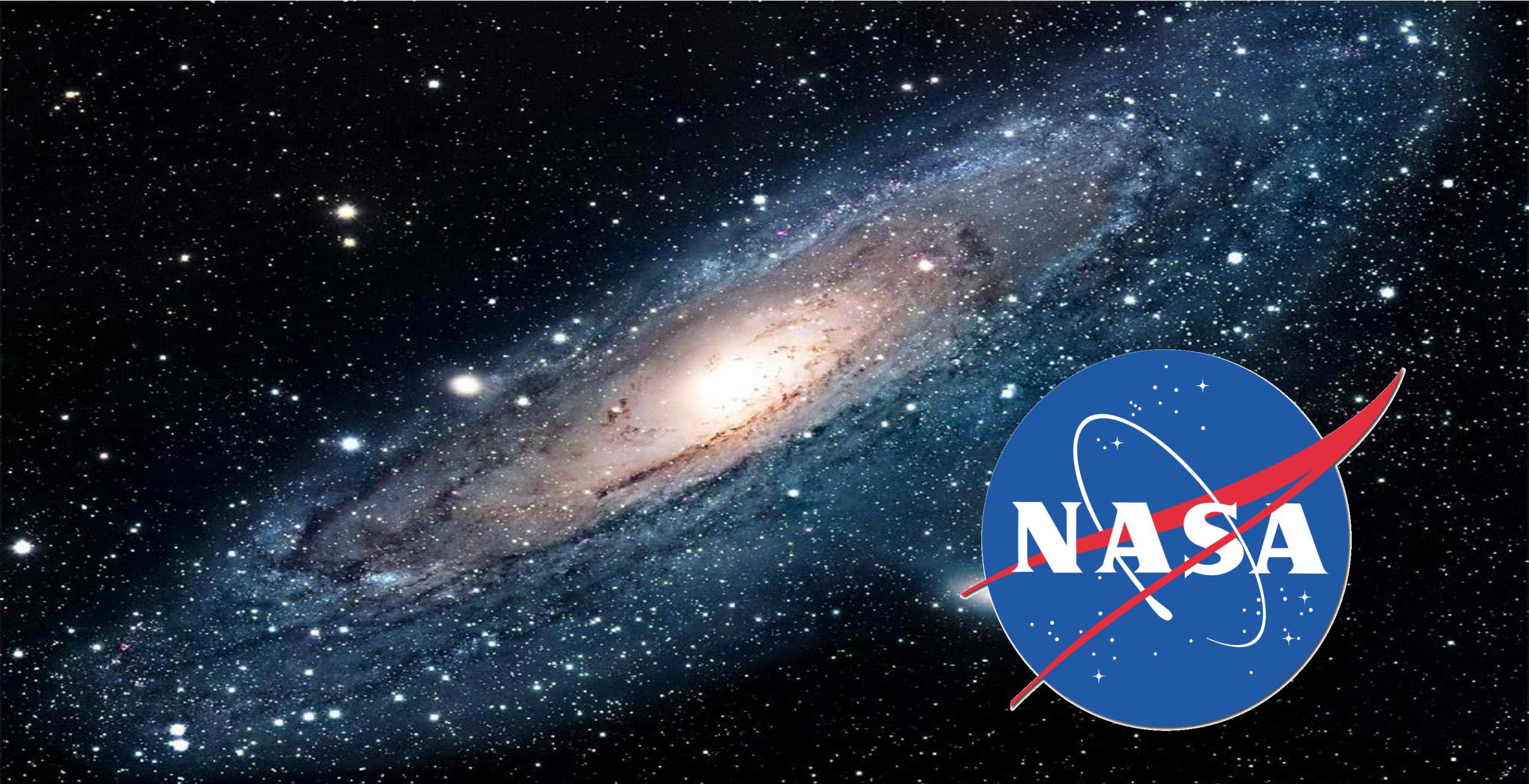
MS004001A Loop no.
→ Eastbound
← Westbound
Route Id

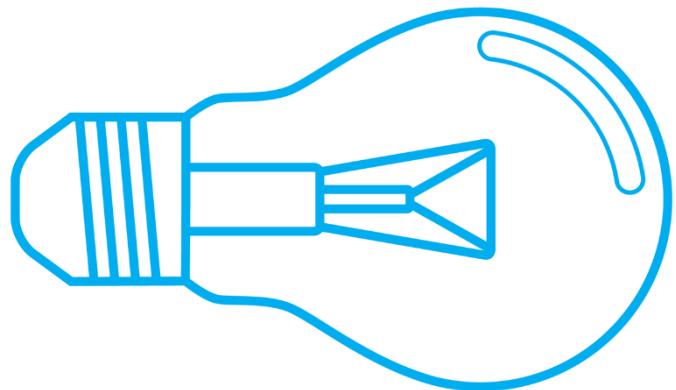
kaggle

© 2010 Kaggle Pty Ltd
ACN 139 471 444

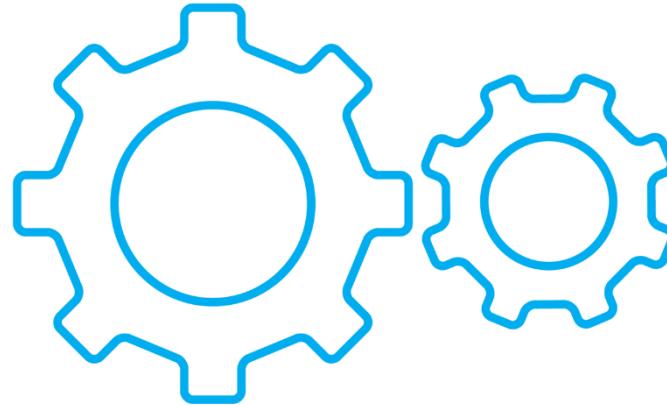
VicRoads has an algorithm they use to forecast travel time on Melbourne freeways (taking into account time, weather, accidents, etc). Their current model is inaccurate and somewhat useless. They want to do better (or at least find out about whether it's possible to do better).



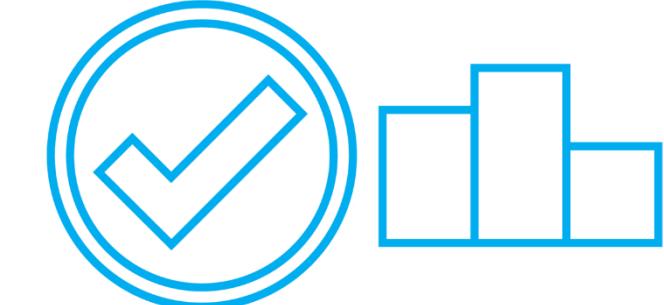




1
Upload



2
Submit



3
**Evaluate &
Exchange**

1. Edit details

2. Competition Format

3. Evaluation Criteria

4. Edit Pages/Upload Data

5. Preview

CREATE A COMPETITION

Welcome to the Kaggle Post-a-Competition wizard. First, please enter the competition details. ([Help](#))

Competition title *

One-sentence outline *

This will be displayed as the summary of your competition throughout the site.
You have 200 characters left.

Deadline for Entries *

6 days 23 hours from now

(at)

Continue

Any questions? [Contact an administrator.](#)

Use the wizard to post a competition

	A	B
1		40010
2	3/08/10 10:43	831.586207
3	3/08/10 10:58	833.931034
4	3/08/10 11:13	829.517241
5	3/08/10 11:28	836.862069
6	3/08/10 11:58	821.931034
7	3/08/10 12:28	820.689655
8	3/08/10 16:28	1756.03448
9	3/08/10 22:28	897.961538
10	4/08/10 4:28	747.259259
11	4/08/10 10:28	823.857143
12	6/08/10 19:10	913.64
13	6/08/10 19:25	912.56
14	6/08/10 19:40	883.84
15	6/08/10 19:55	861.88
16	6/08/10 20:25	831.8
17	6/08/10 20:55	820.72
18	7/08/10 0:55	893.12
19	7/08/10 6:55	776.961538
20	7/08/10 12:55	832.444444
21	7/08/10 18:55	877.037037
22	9/08/10 16:34	1409.32258
23	9/08/10 16:49	1516.64516
24	9/08/10 17:04	1671.96774
25	9/08/10 17:19	1561.90323



MAKE A SUBMISSION ATTACH FILE/DESCRIPTION

This competition only allows you to make 2 entries in a day (reset at midnight, UTC) [\(Having Trouble? Message the Administrator\)](#)

Team Name: All Zero

No file chosen

Your entry must:

- be in csv format;
- have your predictions in column 3;
- contain only numbers; and
- be 120841 lines long (empty lines will be ignored).

Brief description of
your technique

Participants make their entries

#	Team Name	RMSE	Entries	Latest Submission
1	PEW *	0.640871	130	6:00pm, Monday 1 November 2010
2	UriB *	0.646554	118	9:33am, Saturday 30 October 2010
3	Just For Fun *	0.649665	11	2:34am, Thursday 2 September 2010
4	Old Dogs With New Tricks *	0.649922	87	7:49am, Tuesday 2 November 2010
5	JohnL *	0.652753	11	10:10am, Thursday 7 October 2010
6	PunyPetunias *	0.65485	52	12:04pm, Tuesday 21 September 2010
7	ulvund *	0.655488	52	8:59pm, Thursday 28 October 2010
8	Diogo *	0.655815	85	5:57pm, Monday 1 November 2010
9	Jasonb *	0.656661	50	9:43am, Saturday 23 October 2010
10	ChessMaster *	0.65683	44	6:53pm, Friday 17 September 2010

Competitions are judged based on predictive accuracy

Competition Mechanics

<i>Training dataset</i>			
<i>Age</i>	<i>Income</i>	<i>Default</i>	
58	\$ 95,824.00	TRUE	
73	\$ 20,708.00	FALSE	
59	\$ 82,152.00	FALSE	
66	\$ 25,334.00	FALSE	
39	\$ 35,952.00	FALSE	
78	\$ 51,754.00	FALSE	
76	\$ 76,479.00	TRUE	
71	\$ 96,614.00	TRUE	
22	\$ 27,701.00	FALSE	
57	\$ 35,841.00	FALSE	

<i>Test dataset</i>			
<i>Age</i>	<i>Income</i>	<i>Default</i>	
73	\$ 53,445.00		
61	\$ 36,679.00		
47	\$ 90,422.00		
44	\$ 79,040.00		
46	\$ 67,104.00		
30	\$ 69,992.00		
75	\$ 78,139.00		
28	\$ 66,058.00		
24	\$ 75,240.00		
54	\$ 89,503.00		

Competitions are judged on objective criteria

Kaggle

How They Won It...





Completed • \$950 • 176 teams

Stay Alert! The Ford Challenge

Wed 19 Jan 2011 – Wed 9 Mar 2011 (3 years ago)

Dashboard

Home



| Data

Information



Description

Evaluation

Rules

Prizes

Forum



Leaderboard



Public

Private

Leaderboard

1. inference
2. Mick Wagner
3. Dekosuke
4. Swedish Chef
5. Jason Noriega
6. David
7. brainless
8. Tony Liu
9. Chris Choy
10. One Old Dog

Driving while not alert can be deadly. The objective is to design a classifier that will detect whether the driver is alert or not alert, employing data that are acquired while driving.

Driving while distracted, fatigued or drowsy may lead to accidents. Activities that divert the driver's attention from the road ahead, such as engaging in a conversation with other passengers in the car, making or receiving phone calls, sending or receiving text messages, eating while driving or events outside the car may cause driver distraction. Fatigue and drowsiness can result from driving long hours or from lack of sleep.

The objective of this challenge is to design a detector/classifier that will detect whether the driver is alert or not alert, employing any combination of vehicular, environmental and driver physiological data that are acquired while driving.

This competition requires you to choose five entries that count towards the final result. To choose five entries visit [your submissions](#) page and click the star next to the relevant entry to select it. If you do not choose any entries, your last five entries will be chosen by default.

The winner receives free registration to the [2011 International Joint Conference on Neural Networks](#) (San Jose, California July 31 - August 5, 2011), which is valued at \$950. The winner will also be invited to present their solution at the conference.

176 teams

189 players

1402 entries



Ford Challenge - DataSet

- Goal:
 - Predict Driver Alertness
- Predictors:
 - Psychology – P1 .. P8
 - Environment – E1 .. E11
 - Vehicle – V1 ..V11
 - IsAlert ?
- Data statistics meaningless outside the IsAlert context



Ford Challenge – DataSet Files

Three Files

ford_train

- 510 Trials, ~1,200 observations each spaced by 0.1 sec -> 604,330 rows

ford_test

- 100 Trials, ~1,200 observations/trial, 120,841 rows

example_submission.csv

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	TrialID	ObsNum	IsAlert	P1	P2	P3	P4	P5	P6	P7	P8	E1	E2		
2	0	0	?	38.4294	10.9435	1000	60	0.302277	508	118.11	0	0	0		
3	0	1	?	38.3609	15.3212	1000	60	0.302277	508	118.11	0	0	0		
4	0	2	?	38.2342	11.514	1000	60	0.302277	508	118.11	0	0	0		
5	0	3	?	37.9304	12.2615	1000	60	0.302277	508	118.11	0	0	0		
6	0	4	?	37.8085	12.3666	1000	60	0.302277	504	119.048	0	0	0		

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	
1	TrialID	ObsNum	IsAlert	P1	P2	P3	P4	P5	P6	P7	P8	E1	E2	E3	E4	E5
2	0	0	0	34.7406	9.84593	1400	42.8571	0.290601	572	104.895	0	0	0	1	-20	0.015875
3	0	1	0	34.4215	13.4112	1400	42.8571	0.290601	572	104.895	0	0	0	1	-20	0.015875
4	0	2	0	34.3447	15.1852	1400	42.8571	0.290601	576	104.167	0	0	0	1	-20	0.015875
5	0	3	0	34.3421	8.84696	1400	42.8571	0.290601	576	104.167	0	0	0	1	-20	0.015875

A	B	C	
1	TrialID	ObsNum	Prediction
2	0	0	0
3	0	1	0
4	0	2	0

How the Ford Competition was won

Junpei Komiyama (#4)

- To solve this problem, I constructed a Support Vector Machine (SVM), which is one of the best tools for classification and regression analysis, using the libSVM package.
- This approach took more than 3 hours to complete
- I found some data (P3-P6) were characterized by strong noise... Also, many environmental and vehicular data showed discrete values continuously increased and decreased. These suggested the necessity of preprocessing the observation data before SVM analysis for better performance

How the Ford Competition was won

Junpei Komiyama (#4)

Averaging – improved score and processing time

Average 7 data points

Reduced processing by 86% & Increased score by 0.01

Tools

- *Python processing of csv*
- *libSVM*

How the Ford Competition was won

Mick Wagner (#2)

Tools

- Excel, SQL Server

I spent the majority of my time analyzing the data. I inputted the data into Excel and started examining the data taking note of discrete and continuous values, category based parameters, and simple statistics (mean, median, variance, coefficient of variance). I also looked for extreme outliers.

I made the first 150 trials (~30%) be my test data and the remainder be my training dataset (~70%). This single factor had the largest impact on the accuracy of my final model.

I was concerned that using the entire data set would create too much noise and lead to inaccuracies in the model ... so focused on data with state change



How the Ford Competition was won

Mick Wagner (#2)

- After testing the Decision Tree and Neural Network algorithms against each other and submitting models to Kaggle, I found the Neural Network model to be more accurate
- Only used E4, E5, E6, E7, E8, E9, E10, P6, V4, V6, V10, and V11

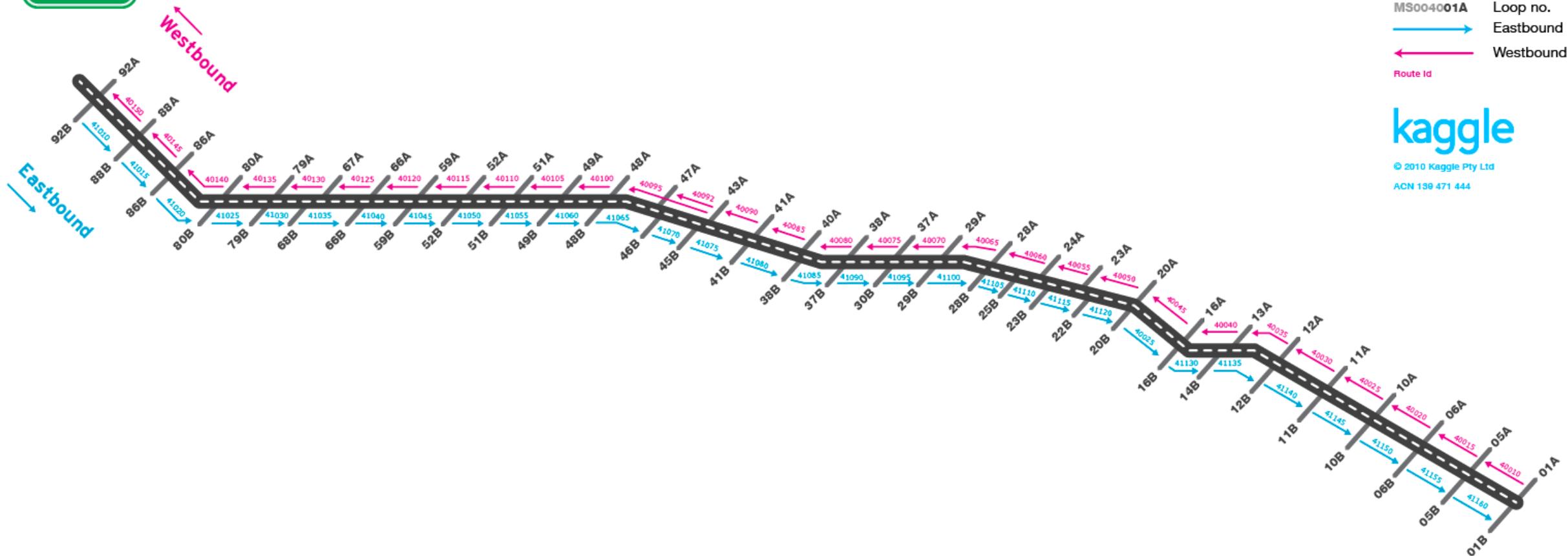
How the Ford Competition was won

Inference (#1)

- “Our first observation is that trials are not homogeneous – so calculated mean, sd et al”
- “Training set & test set are not from the same population” – a good fit for training will result in a low score

Lucky Model (Regression)

- $-.410.6073(sd(E5)) + 0.1494(V11) + 4.4185(E9)$
- (Remember – Data had $P1-P8, E1-E11, V1-V11$)

M4

VicRoads has an algorithm they use to forecast travel time on Melbourne freeways (taking into account time, weather, accidents etc). Their current model is inaccurate and somewhat useless. They want to do better (or at least find out about whether it's possible to do better).

How the RTA was won

- Thanks to
 - François GUILLEM &
 - Andrzej Janusz
- They both used R
- Share their code & algorithms

LEADERBOARD

This competition has completed, this leaderboard reflects the preliminary final standings. The results will become final after the competition organizers verify the results.

* in the money

#	A1w	Team Name	RMSE	Entries	Last Submission UTC (Best Submission - Last)
1	13	Team Irazu *	191.47400	25	Sat, 12 Feb 2011 17:42:55 (-20.1h)
2	-	Mooma	191.83600	134	Tue, 08 Feb 2011 19:25:17 (-41.8h)
3	†27	SuperCow	192.23199	16	Fri, 11 Feb 2011 19:58:38
4	†1	zkolber	193.21300	4	Wed, 05 Jan 2011 22:10:29
5	new	Apache	193.44000	4	Thu, 10 Feb 2011 13:21:41 (-24.3h)
6	†5	Hipoteza	193.56300	49	Sun, 13 Feb 2011 20:30:40 (-6.6d)
7	†2	Poco	195.38000	28	Sun, 13 Feb 2011 16:53:45 (-11.4d)
8	†2	Ghalib	195.82300	78	Sun, 13 Feb 2011 20:22:22 (-27.2h)
9	†1	Femen	196.00000	12	Sun, 13 Feb 2011 17:55:26 (-5d)
10	†2	plig	197.23900	48	Sun, 13 Feb 2011 17:10:18
11	†4	Mop	197.75000	7	Sun, 13 Feb 2011 17:03:32 (-11.3d)
12	new	Maxim	198.22000	6	Sat, 12 Feb 2011 20:34:32
13	†1	vsh	198.48899	5	Sun, 13 Feb 2011 20:30:47 (-4.6d)
14	†3	Igullem	199.76199	37	Sun, 13 Feb 2011 18:33:03 (-64.1d)
15	†1	Maney	199.77901	35	Sun, 13 Feb 2011 21:38:16 (-4.0d)
16	†3	uown	200.14000	26	Sun, 13 Feb 2011 08:46:00 (-0.3h)
17	†4	NosferatoCorp	200.52600	19	Wed, 02 Feb 2011 18:27:42 (-2.6d)
18	†3	Jerzy Kozera	200.76700	4	Fri, 03 Dec 2010 02:01:07 (-0.1h)

How the RTA was won

François GUILLEM (#14)

- I used a simple k-NN approach but the idea was to process data first & to compute some summaries of time series in consecutive timestamps using some standard indicators from technical analysis.

How the RTA was won

#1 used Random Forests

Time, Date & Week as predictors

José P. González-Brenes and Matías Cortés

Regression models for data segments (total~600!)

Tools:

- *Java/Weka*
- *4 processors, 12 GB RAM*
- *48 hours of computations*

Lessons From Kaggle Winners

1. Don't over-fit
2. All predictors (variables) are not needed
3. All data rows are not needed, either
4. Tuning the algorithms will give different results
5. Reduce the dataset (Average, select transition data,...)
6. Test set & training set can differ
7. Iteratively explore & get your head around the data



Out of Class Project – two weeks to complete...

Lecture 7

Monday August 20th
Upload to site...

no free hunch kaggle

▼ Archives

dojo

Tutorials and Winners' Interviews

September 15, 2014 by The Kaggle Team 0

Lessons Learned from the Hunt for Prohibited Content on Kaggle

(Cross-posted from MLWave.com) Kaggle hosted a contest together with Avito.ru. The task was to automatically detect illicit content in the advertisements on their site. Many competitors were using Vowpal Wabbit for this challenge. Some aided by the

August 5, 2014 by Kiran 6

3rd Place interview from the KDD Cup 2014

Kiran placed 3rd in the KDD Cup and shared this interview with No Free Hunch: What was your background prior to entering this challenge? I am a computer science engineer and management post-grad, heading marketing analytics, mobile analytics

August 1, 2014 by David Wind 6

Learning from the best

Guest contributor David Kofoed Wind is a PhD student in Cognitive Systems at The Technical University of Denmark (DTU): As a part of my master's thesis on competitive machine learning, I talked to a series of Kaggle Masters to try to ...

<http://blog.kaggle.com/category/dojo/>

Content is 12 pages of interview on how the team(s) built their models, some have multiple interviews; You will review **at least 10** interviews, bounce around **do not** go sequentially.

1) What model(s) did they use, 2) insights they had that influenced modeling, 3) what feature creation and selection, 4) other observations. I will consolidate all these together and upload as shared document on our site.

5 Minute Break...

Course Project



Digit Recognizer

Knowledge • 649 teams

Digit Recognizer

Wed 25 Jul 2012

Thu 31 Dec 2015 (5 months to go)

[Competition Details](#) » [Get the Data](#) » [Make a submission](#)

Classify handwritten digits using the famous MNIST data

[Get started on this competition through Kaggle Scripts](#)

The goal in this competition is to take an image of a handwritten single digit, and determine what that digit is. As the competition progresses, we will release tutorials which explain different machine learning algorithms and help you to get started.

The data for this competition were taken from the MNIST dataset. The MNIST ("Modified National Institute of Standards and Technology") dataset is a classic within the Machine Learning community that has been extensively studied. More detail about the dataset, including Machine Learning algorithms that have been tried on it and their levels of success, can be found at <http://yann.lecun.com/exdb/mnist/index.html>.

Digit Recognizer

<https://www.kaggle.com/c/digit-recognizer>

- Teams listed in deck, limited switching is possible;
- Take this in stages, weekly discussion in class;
- Create a team name register with Kaggle;
- Get the data, remember what we have discussed;
- It's about learning, especially about Kaggle competitions, not competing, everyone submits;
- Let's discuss further...



Digit Recognizer

Data Files

File Name	Available Formats
train	.csv (73.22 mb)
test	.csv (48.75 mb)

[See benchmark code and a sample submission with the Random Forest Benchmark](#)

Digit Recognizer – Data Format

The data files train.csv and test.csv contain gray-scale images of hand-drawn digits, from zero through nine.

Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 255, inclusive.

The training data set, (train.csv), has 785 columns. The first column, called "label", is the digit that was drawn by the user. The rest of the columns contain the pixel-values of the associated image.

Each pixel column in the training set has a name like pixelx, where x is an integer between 0 and 783, inclusive. To locate this pixel on the image, suppose that we have decomposed x as $x = i * 28 + j$, where i and j are integers between 0 and 27, inclusive. Then pixelx is located on row i and column j of a 28 x 28 matrix, (indexing by zero).

For example, pixel31 indicates the pixel that is in the fourth column from the left, and the second row from the top, as in the ascii-diagram below.

Visually, if we omit the "pixel" prefix, the pixels make up the image like this:

000	001	002	003	...	026	027
028	029	030	031	...	054	055
056	057	058	059	...	082	083
				...		
728	729	730	731	...	754	755
756	757	758	759	...	782	783

The test data set, (test.csv), is the same as the training set, except that it does not contain the "label" column.

Digit Recognizer - Submissions

Your submission file should be in the following format: For each of the 28000 images in the test set, output a single line with the digit you predict. For example, if you predict that the first image is of a 3, the second image is of a 7, and the third image is of a 8, then your submission file would look like:

```
3  
7  
8  
(27997 more lines)
```

The evaluation metric for this contest is the categorization accuracy, or the proportion of test images that are correctly classified. For example, a categorization accuracy of 0.97 indicates that you have correctly classified all but 3% of the images.



Digit Recognizer : Rules

One account per participant

You cannot sign up to Kaggle from multiple accounts and therefore you cannot submit from multiple accounts.

No private sharing outside teams

Privately sharing code or data outside of teams is not permitted. It's okay to share code if made available to all participants on the forums.

Team Mergers

Team mergers are not allowed in this competition.

Team Limits

The maximum size of a team is 100 participants.

Submission Limits

You may submit a maximum of 5 entries per day.

You may select up to 6 final submissions for judging.



Competition Timeline

Start Date: **7/25/2012 8:43:30 PM UTC**

Merger Deadline: **None**

First Submission Deadline: **None**

End Date: **12/31/2015 11:59:00 PM UTC**



Homework 6 - due by Lesson #7

- Perform data analysis, what is the data telling you
 - What were your top 3 – 5 observations from initial data exploration;
 - Describe and provide supporting screen shots;
- What features look most promising for modeling?
 - Describe how you selected these features, screen shots to support your discussion, if appropriate;
- Do you need to deal with missing values?
- Scale the attributes?
- Make initial attempt to model and predict, set your own baseline



Homework 7, part 2 due by Lesson #8

- Construct initial model, post on Kaggle to set your team baseline
 - Screenshot of position on leaderboard;
- Compare with multiple classifiers, model settings
- Opportunity for feature selection, feature transformation such as binning or clustering, etc.

Describe the data cleaning, transformation steps you selected and describe the why and how. Run at least two to three new model evaluations, submit at least one new submission to Kaggle contests site

Your Assignment, part 3 due by Lesson #10

Repeat Part 2 as necessary during 3 part of assignment

- Compare with multiple classifiers, different model settings
 - Write up describing modeling strategy, screen shots of both results in WEKA or tool of choice to illustrate exploration phase, screen shots of all submissions to Kaggle contest site;
- Can you identify an opportunity for model ensembles, if so describe and evaluate;

Teams

TEAM #1

- Clark, James Travis
- Arshad, Syed Sameer
- Bareham, Jeffrey S
- Bhavsar, Pramod S
- Carutasu, Diana A
- Pedroza, Lucas J

TEAM #2

- Carter, Michael Lee
- Cruwys, Scott M
- Fan, Wenhong
- Hardy, Rebecca C
- Jose, Damien
- Kurupati, Aparna

TEAM #3

- Chang, Isaac P
- Smith, Richard J
- LaPoint, Damon I
- Lauzon, Joseph R M
- Maddali, Raghava Rao
- Pidshetti, Sandeep

TEAM #4

- Bidhar, Rakesh Kumar
- Sinha, Arnab
- Merriam, Elliott
- Palraj, Ponsivakumar
- Chawla, Anurag
- Mayfield, Neil A

TEAM #5

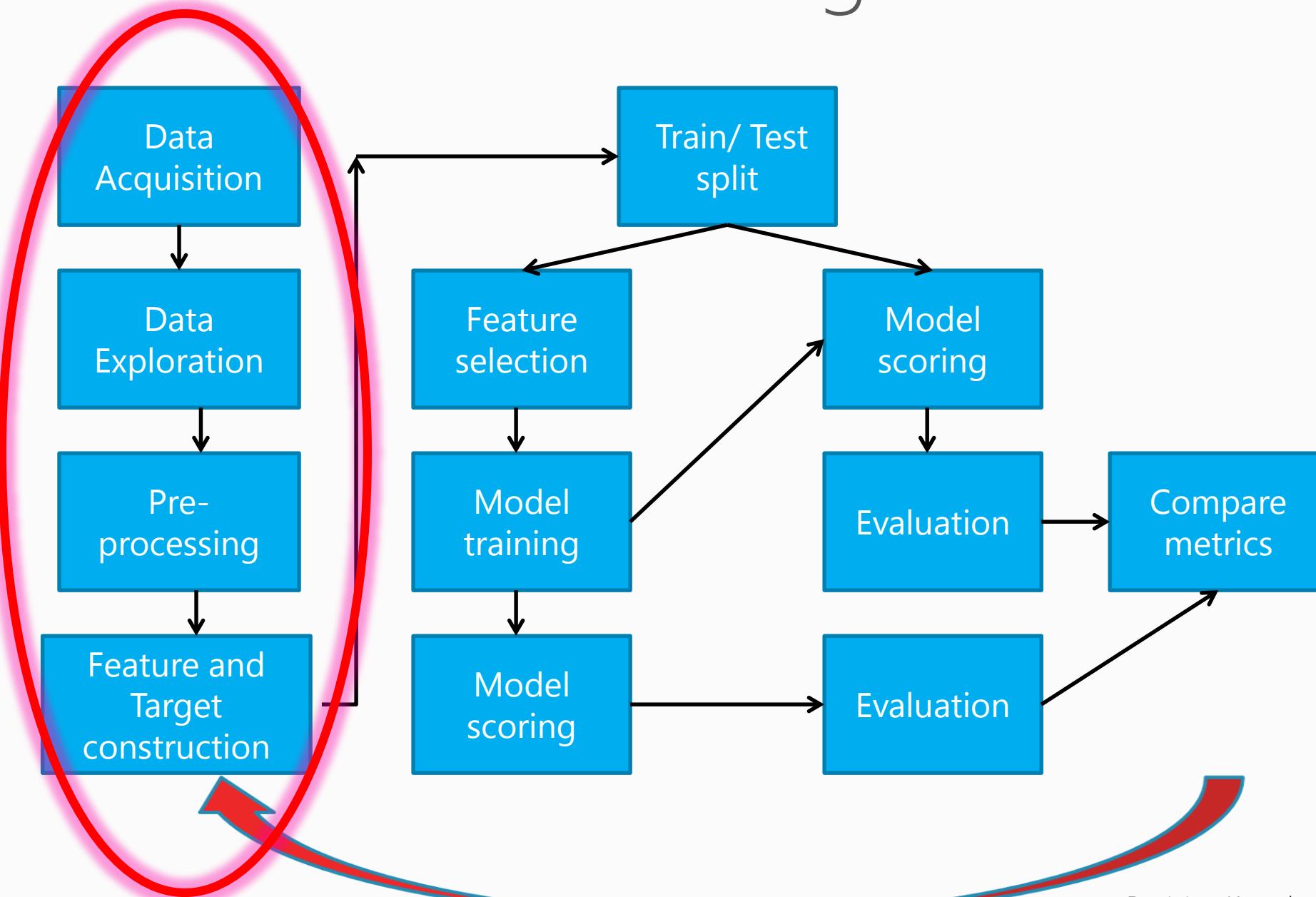
- McKeag, Gregory Alan
- McKeag, Michelle
- Prabhu, Chetan
- Ramirez, Gabriel E
- Shah, Samarth
- Tangen, Cameron



Data Wrangling



Steps in Predictive Modeling Process



Data Acquisition

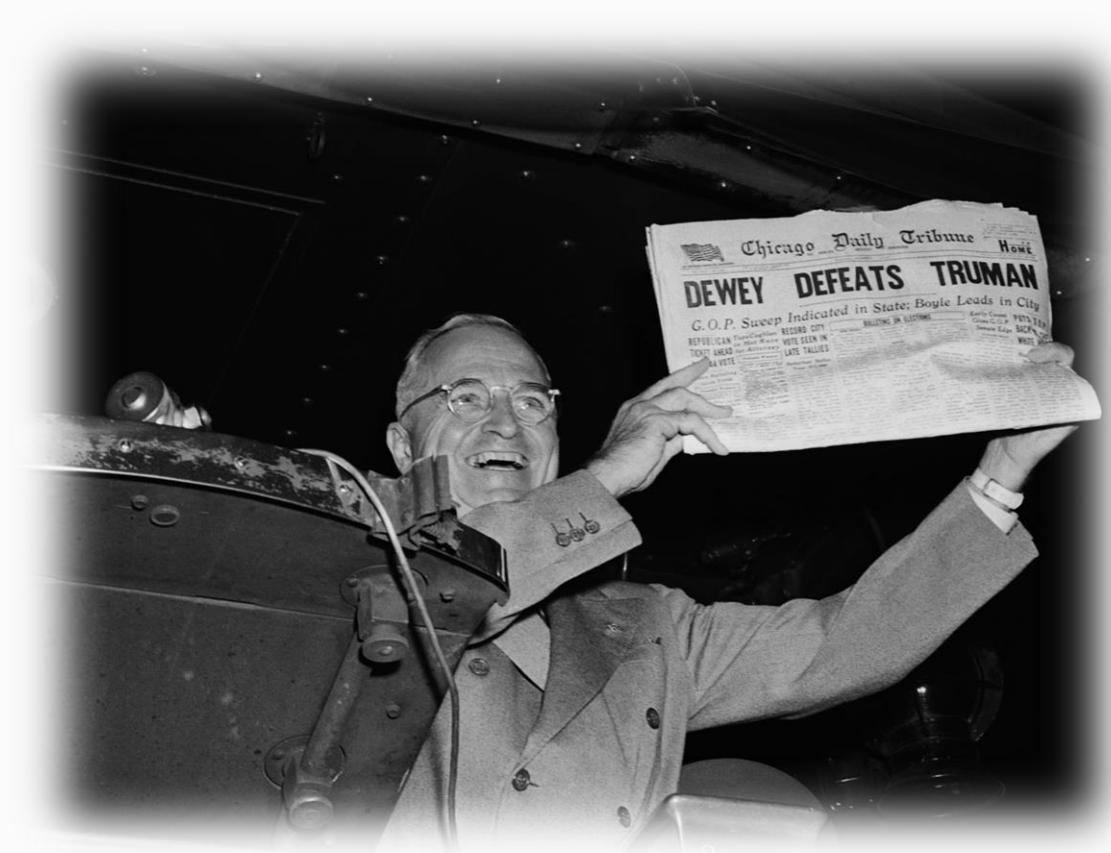
Depends on the domain

- Data can be
 - Sampled
 - Uniformly (e.g. one out of every X data points is recorded)
 - Stratified (e.g. all outages and a sample of normal operations is recorded)
 - Subject to an earlier predictive model
 - E.g. credit card defaults are subject to prior model of credit scores
 - Missing attribute values
 - Missing records (e.g. some sensors have broken down)
 - Misleading (e.g. sensors saturated and show the same value)
 - Biased (e.g. sensors in a portion of field may be ineffective)
 - Often dependent on the problem itself
 - E.g. in drug effectiveness studies, experimental drugs are only administered to critically ill patients

Data Exploration

Critical step in the modeling process

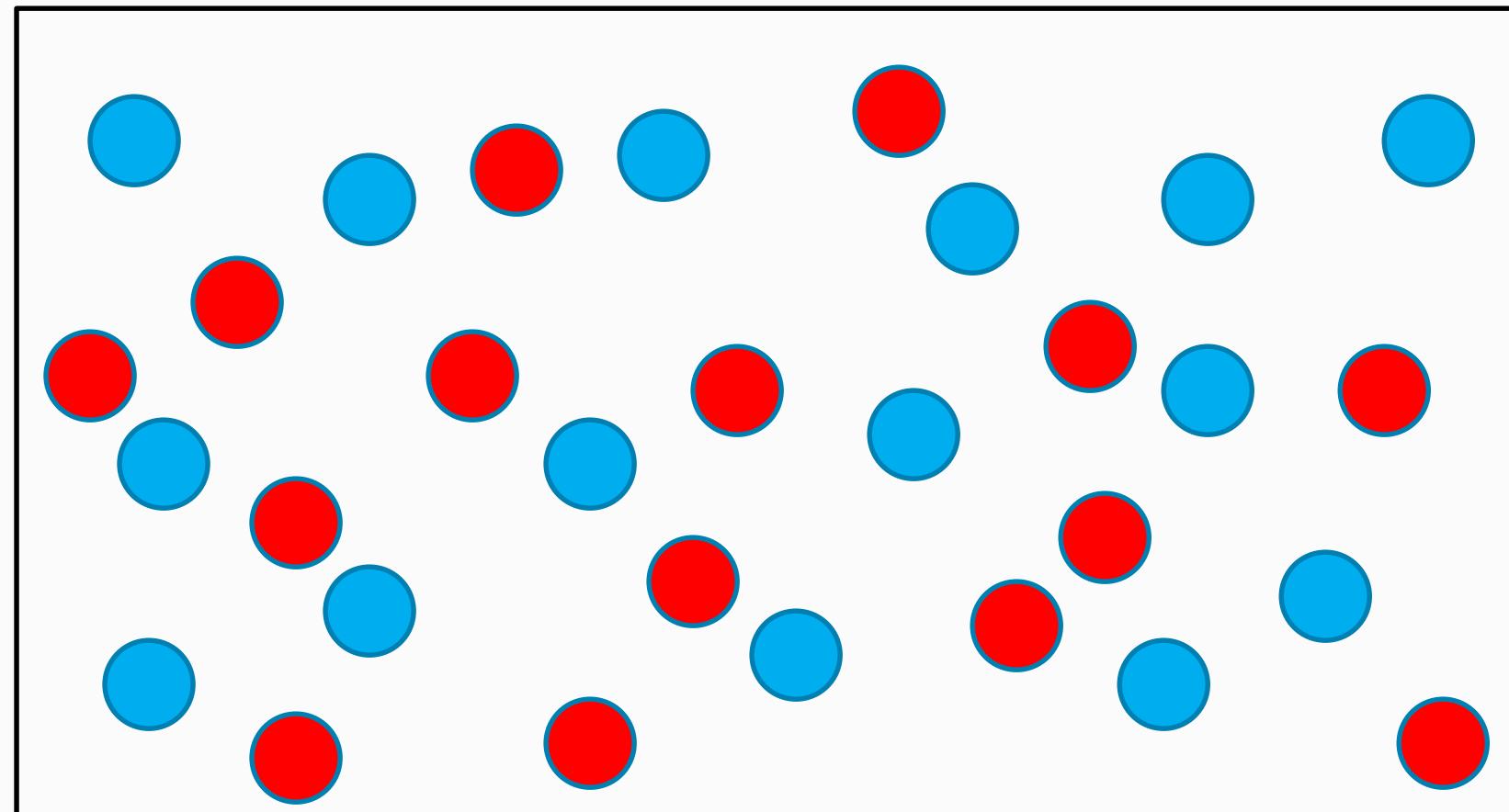
- Important to understand the
 - Quality and Quantity of data
 - Statistical properties of variables
 - Relationships between variables
 - Missing values
 - Biases in the data acquisition



Modeling Dataset

- Split the dataset into a training and testing subset.
- Splits can be
 - Random
 - Each data record is randomly assigned to either train or test
 - Stratified
 - Assign so that each split (train/test) has the same relative proportions of each pair of (x, y) E.g.,
 - 67% of records with $(y = 1)$ and $(y = 0)$ go to train
 - 33% of records with $(y = 1)$ and $(y = 0)$ each go to test
- Temporal datasets can be split either:
 - Horizontal
 - Random over time
 - Vertical
 - Split at a point in time

Modeling Dataset

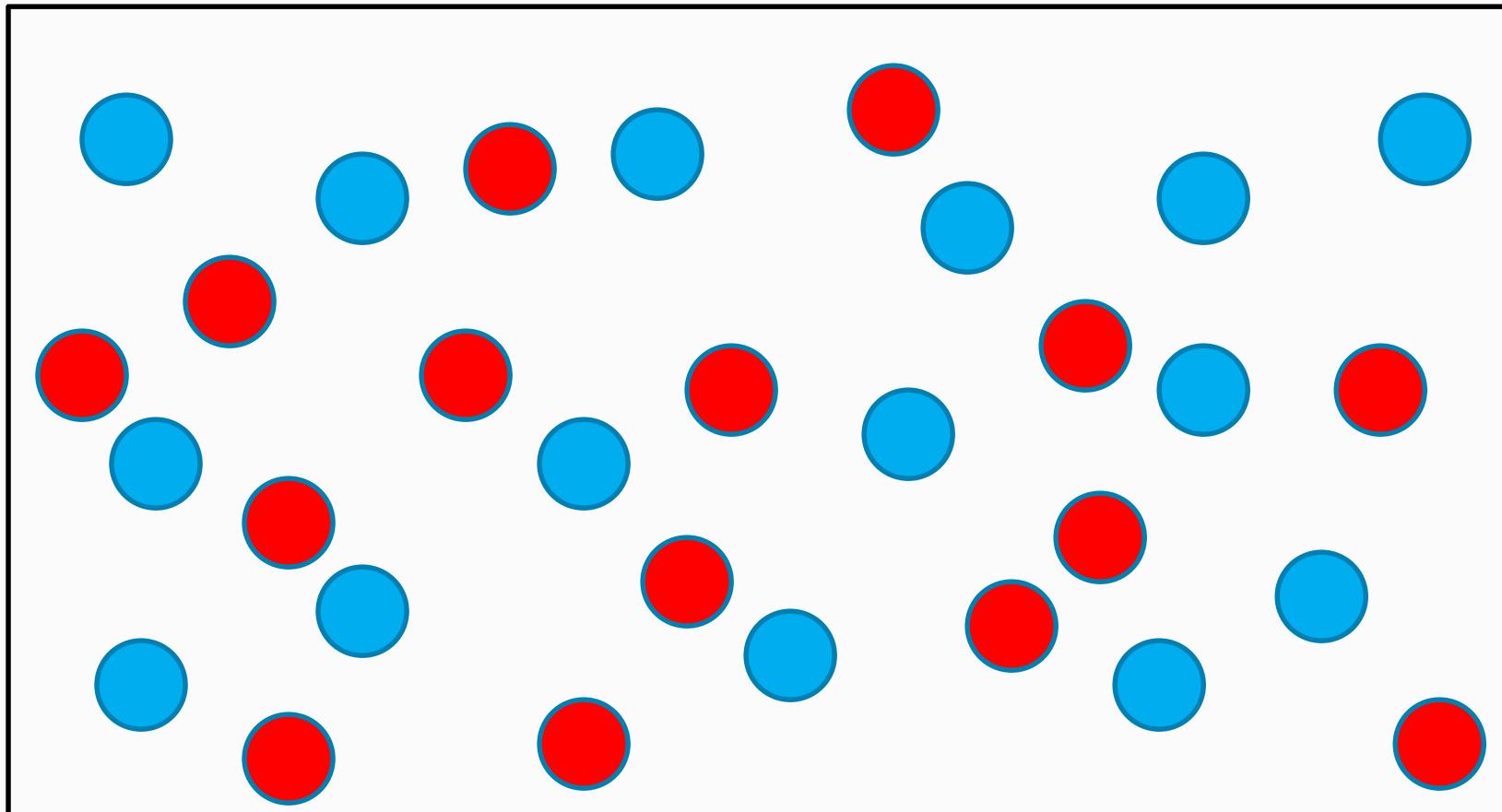


15

15

$$P(\text{blue}) = 0.5$$
$$P(\text{red}) = 0.5$$

Train/Test Split: Random



7

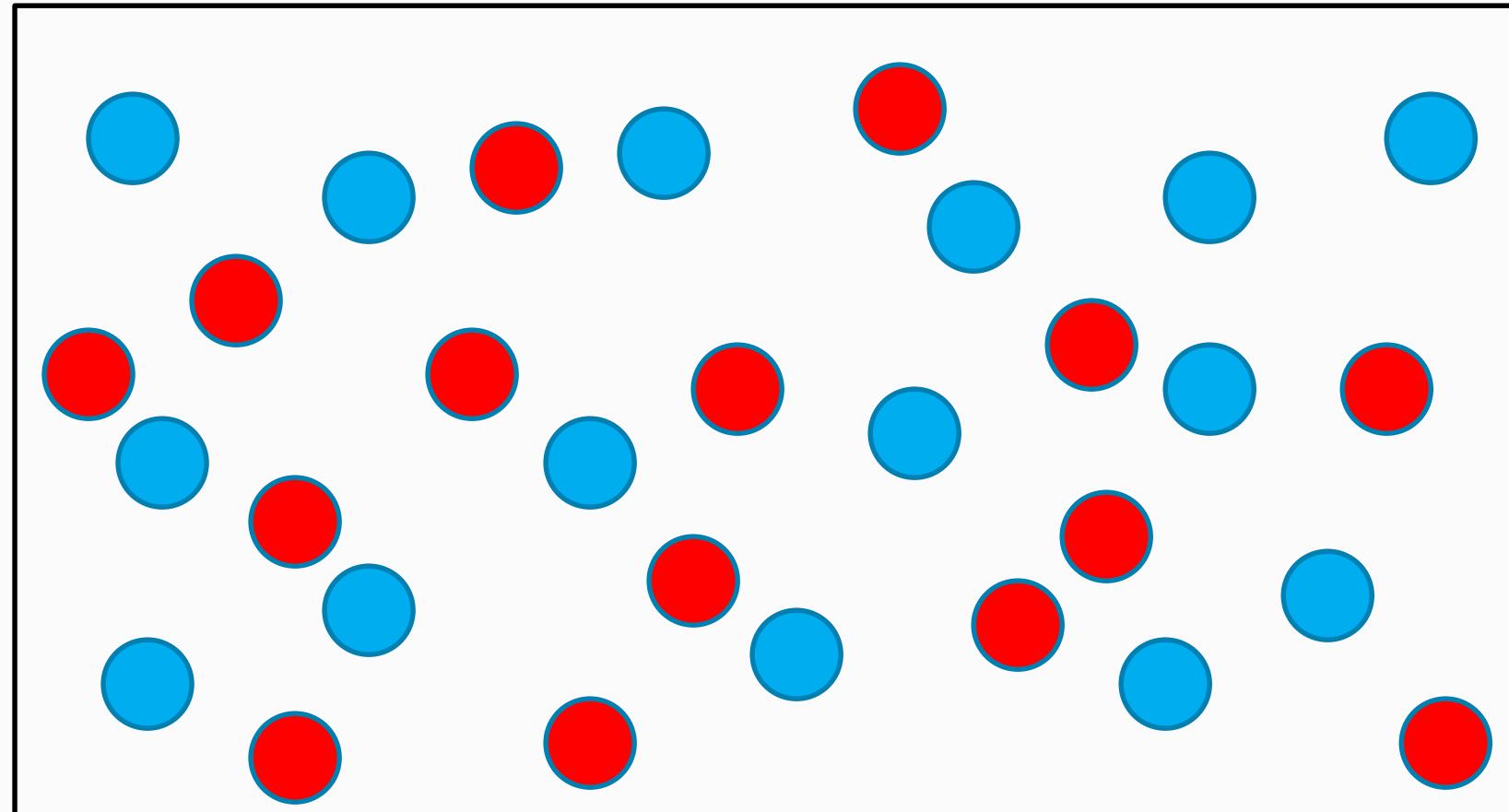


13



$$P(\text{Blue}) = 0.35$$
$$P(\text{Red}) = 0.65$$

Train/Test Split: Stratified

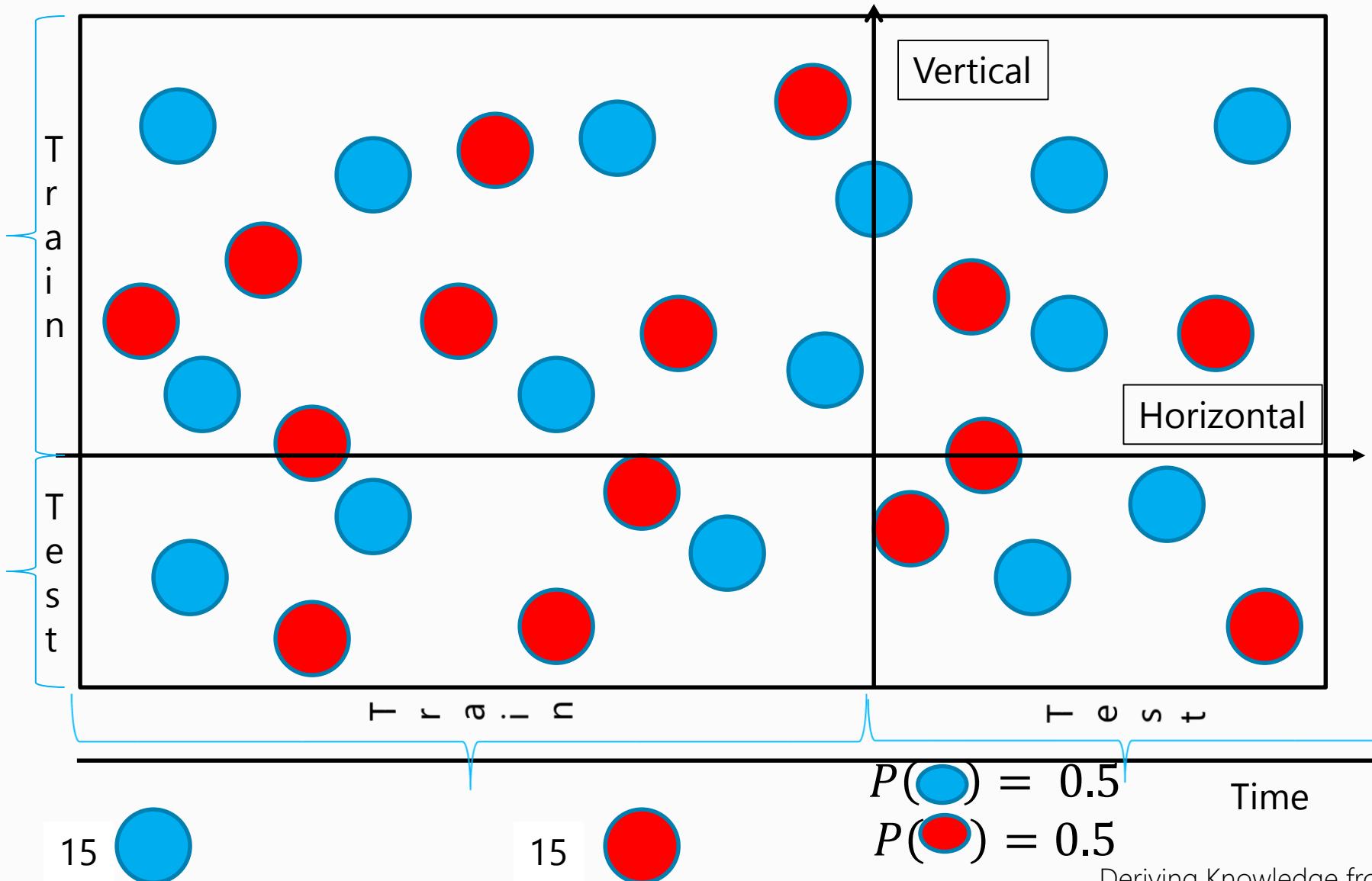


10

10

$$P(\text{Blue}) = 0.5$$
$$P(\text{Red}) = 0.5$$

Train/Test Split: Temporal Splits



Data Characterization

1. Unique values
2. Most frequent values
3. Highest and lowest values
4. Location and dispersion – gini, statistical test for dispersion
5. Quartiles

Data Cleaning

1. Missing values
2. Outliers
3. Coding
4. Constraints

Data Cleaning

Missing values – UCI machine learning repository, 31 of 68 data sets reported to have missing values. "Missing" can mean many things...

MAR: "Missing at Random":

- usually best case
- usually not true

Non-randomly missing

Presumed normal, so not measured

Causally missing

- attribute value is missing because of other attribute values (or because of the outcome value!)



Dealing With Missing Data

Throw away cases with missing values

- in some data sets, most cases get thrown away
- if not missing at random, throwing away cases can bias sample towards certain kinds of cases

Treat “missing” as a new attribute value

- what value should we use to code for missing with continuous or ordinal attributes?
- if missing causally related to what is being predicted?

Impute (fill-in) missing values

- once filled in, data set is easy to use
- if missing values poorly predicted, may hurt performance of subsequent uses of data set

Missing Values: Imputing

Fill-in with mean, median, or most common value

Predict missing values using machine learning

Expectation Minimization (EM):

- Build model of data values (ignore missing values)
- Use model to estimate missing values
- Build new model of data values (including estimated values from previous step)
- Use new model to re-estimate missing values
- Re-estimate model
- Repeat until convergence



Missing Values: Potential Problems

Imputed values may be inappropriate:

- in medical databases, if missing values not imputed separately for male and female patients, may end up with male patients with 1.3 prior pregnancies, and female patients with low sperm counts
- many of these situations will not be so humorous/obvious!

If some attributes are difficult to predict, filled-in values may be random (or worse)

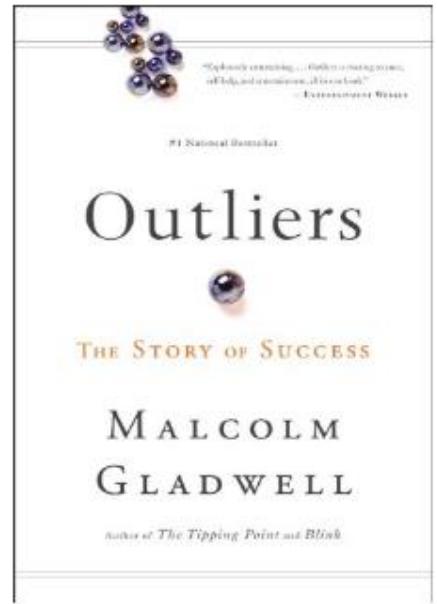
Some of the best performing machine learning methods are impractical to use for filling in missing values (neural nets)

Beware of coding - reliably detect missing cases can be difficult



Data Cleaning

Outliers – may indicate 'bad data' or it may represent something scientifically interesting in the data...



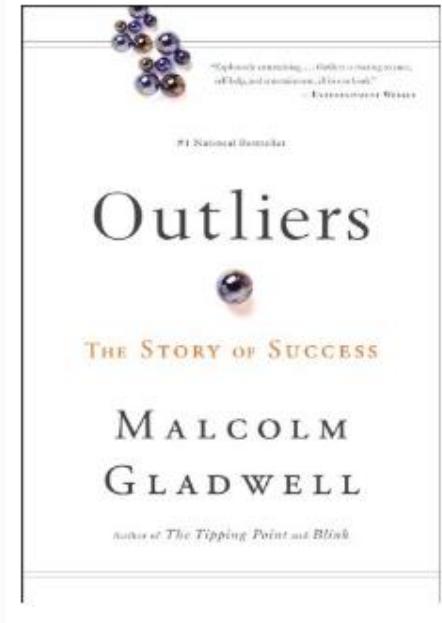
Simple working definition: an outlier is an element of a data sequence S that is inconsistent with expectations, based on the majority of other elements of S.

Sources of outliers

- Insurance company sees niche of sports car enthusiasts, married boomers with kids and second family car. Low risk, lower rate to attract. Simple case where outlier carries meaning for modeling...

Data Cleaning

Outliers – may indicate 'bad data' or it may represent something scientifically interesting in the data...



Outliers can distort the regression results. When an outlier is included in the analysis, it can pull the regression line towards itself. This can result in a solution that is more accurate for the outlier, but less accurate for all the other cases in the data set.

Data Cleaning

Outliers – may indicate 'bad data' or it may represent something scientifically interesting in the data...

Identify outliers

- Question origin, domain knowledge invaluable
- Dispersion – "spread" of a data set, departure from central tendency, use a box plot...

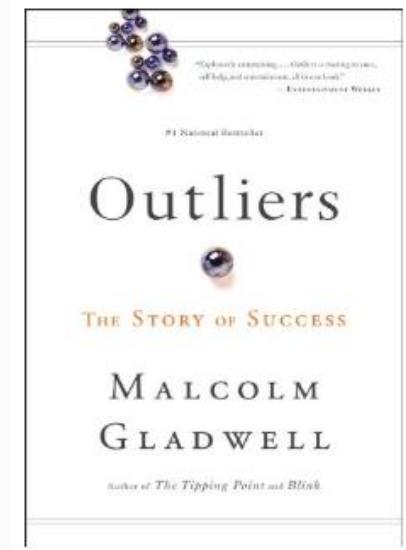
Deal with outliers

- Winsorize – Set all outliers to a specified percentile of the data. Not equivalent to trimming, which simply excludes data. In a Winsorized estimator, extreme values are instead replaced by certain percentiles (the trimmed minimum and maximum). Same as clipping in signal processing.

{92, 19, 101, 58, 153, 91, 26, 78, 10, 13, -40, 101, 86, 85, 15, 89, 89, 25, 2, 41} ($N = 20$)

{92, 19, 101, 58, 101, 91, 26, 78, 10, 13, 2, 101, 86, 85, 15, 89, 89, 25, 2, 41} ($N = 20$)

{92, 19, 101, 58, 91, 26, 78, 10, 13, 101, 86, 85, 15, 89, 89, 25, 2, 41} ($N = 18$)



Data Cleaning

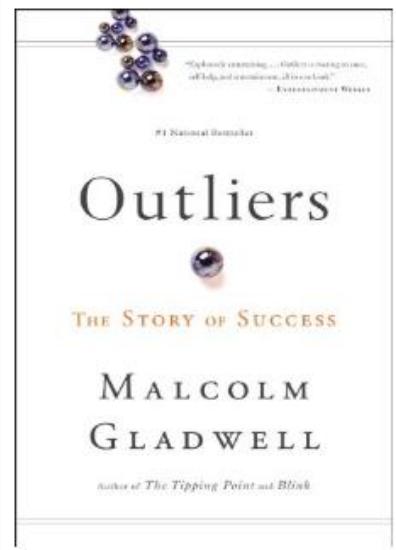
Outliers – may indicate 'bad data' or it may represent something scientifically interesting in the data...

Identify outliers

- Question origin, domain knowledge invaluable
- Dispersion – "spread" of a data set, departure from central tendency, use a box plot...

Deal with outliers

- **Include** – Robust statistics, a convenient way to summarize results when they include a small proportion of outliers. A hot topic for research, see NIPS 2010 Workshop, Robust Statistical learning (robustml).
- Data transformation can eliminate the extreme tendency of the outlier e.g. transforming to Log scale converts extreme values to acceptable range



Data Cleaning

Coding – shape and enrich...

Numerical data

- Binning – a mapping to discrete categories;
- Recenter – shift by c where max, min, avg and median shift, the range and standard deviation **will not** shift;
- Rescale – multiply everything by d , all measures change;
- Standard ND – recenter, make mean 0, divide all previous values by SD

Character data

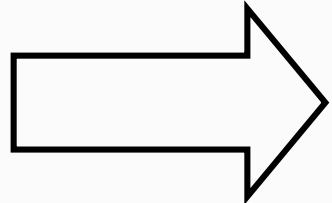
- Lower case
- Spellcheck
- Data extraction (e.g. regular expressions)

Categorical variables

- Non-numeric variables with a finite number of levels
 - E.g. "red", "blue", "green"
- Some ML algorithms can only handle numeric variables
 - Solution: 1-to-N coding

1-to-N Coding

feature
red
blue
green
red
red
green
blue



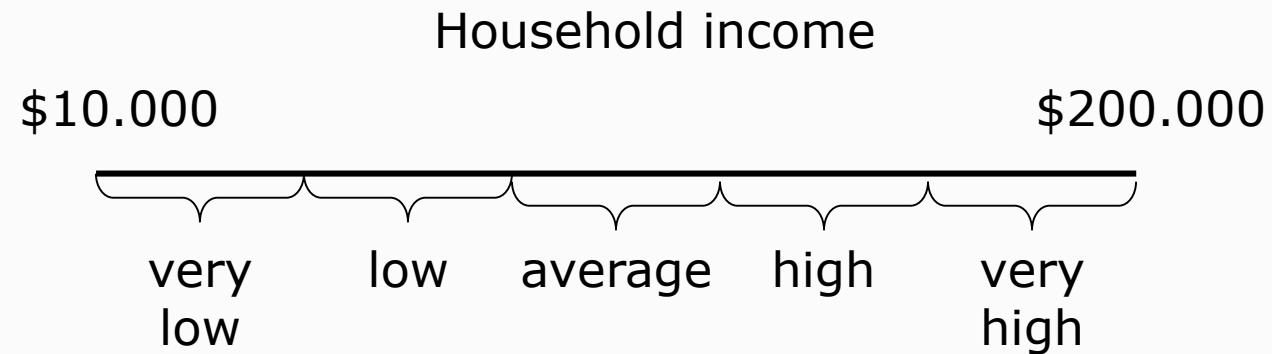
	red	blue	green
1	0	0	0
0	1	0	0
0	0	1	0
1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0

Scaling of Continuous Variables

- Many ML algorithms rely on measuring the distance between 2 samples
- There should be no difference if a length variable is measured in cm, inch, or km
- To remove the unit of measure (e.g. kg, mph, ...) each variable dimension is normalized:
 - subtract mean
 - divide by standard deviation

Output variables

- ML requires categorical output (continuous output = regression)
 - ML methods can be applied by binning continuous output (loss of prediction accuracy)



Data cleansing in WEKA ...

Unsupervised instance filters

- Remove instances incorrectly classified according to a specified classifier
useful for removing outliers
 - *weka.filters.unsupervised.instance.RemoveMisclassified*
- Remove a given percentage of a dataset
 - *weka.filters.unsupervised.instance.RemovePercentage*
- Remove a given range of instances
 - *weka.filters.unsupervised.instance.RemoveRange*
- Filter out instances with certain attribute values
 - *weka.filters.unsupervised.instance.RemoveWithValues*
- Produce random subsample of a dataset, sampling with replacement
 - *weka.filters.unsupervised.instance.Resample*

Feature Selection

- Process of selecting a subset of features that are good predictors of the target
- Useful for
 - Controlling complexity of model
 - Speed up model learning without reducing accuracy
 - Improve generalization capability

Feature Selection, 3 types of methods

Filter Methods, select a subset of features before training a model, e.g.

- Correlation with target,
- Mutual Information between feature and target
- *Simple to implement, and have reasonable performance*

Wrapper Methods, search combination of feature space by training and evaluating model using a subset of features, e.g.

- Forward, backward, step-wise feature selection,
- Genetic algorithms.
- *Computationally expensive and prone to over-fitting*

Embedded Methods, feature subset is chosen as part of model training, e.g.

- LASSO (L-1) regression, Regularized **decision trees, random forests**
- *Typically robust to over-fitting, but has hyper parameters that will need to be fit using a validation data*

Feature Selection and Engineering

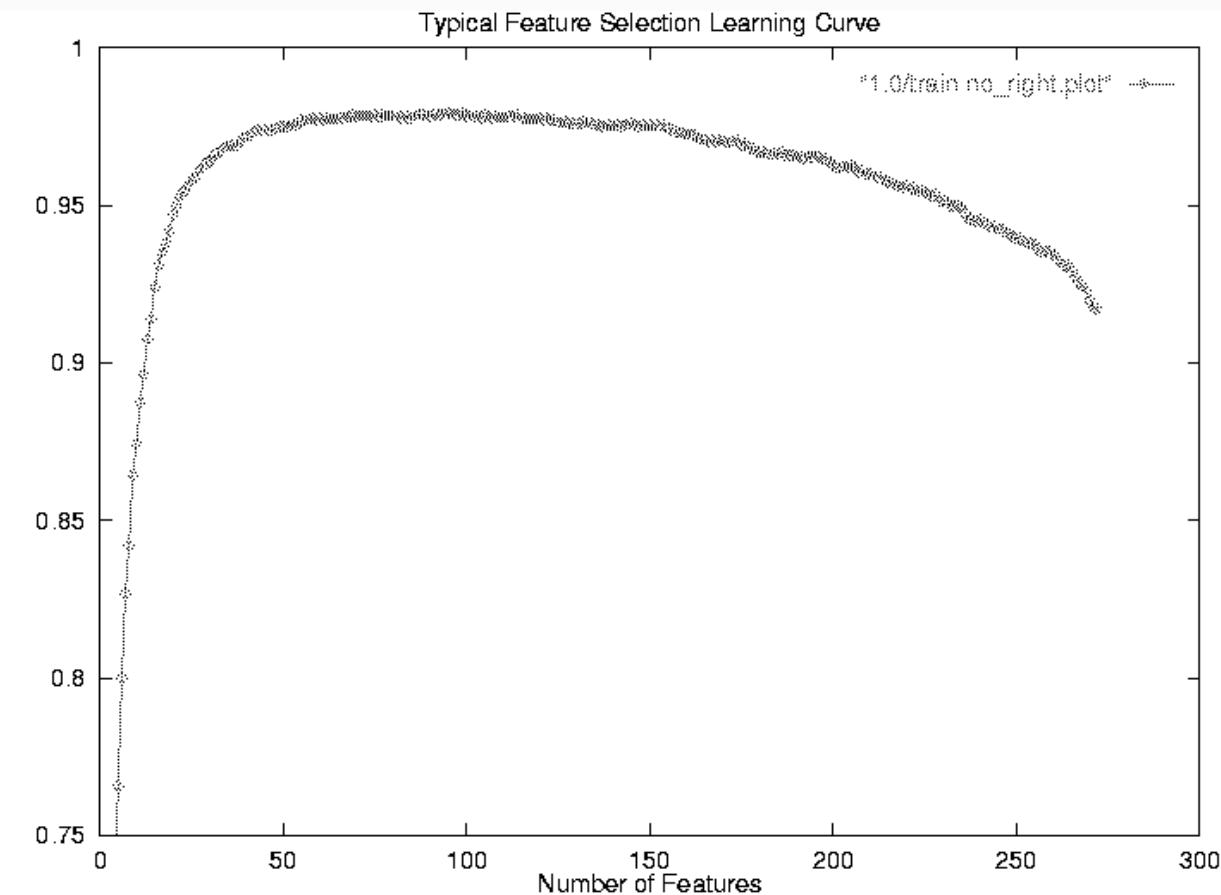
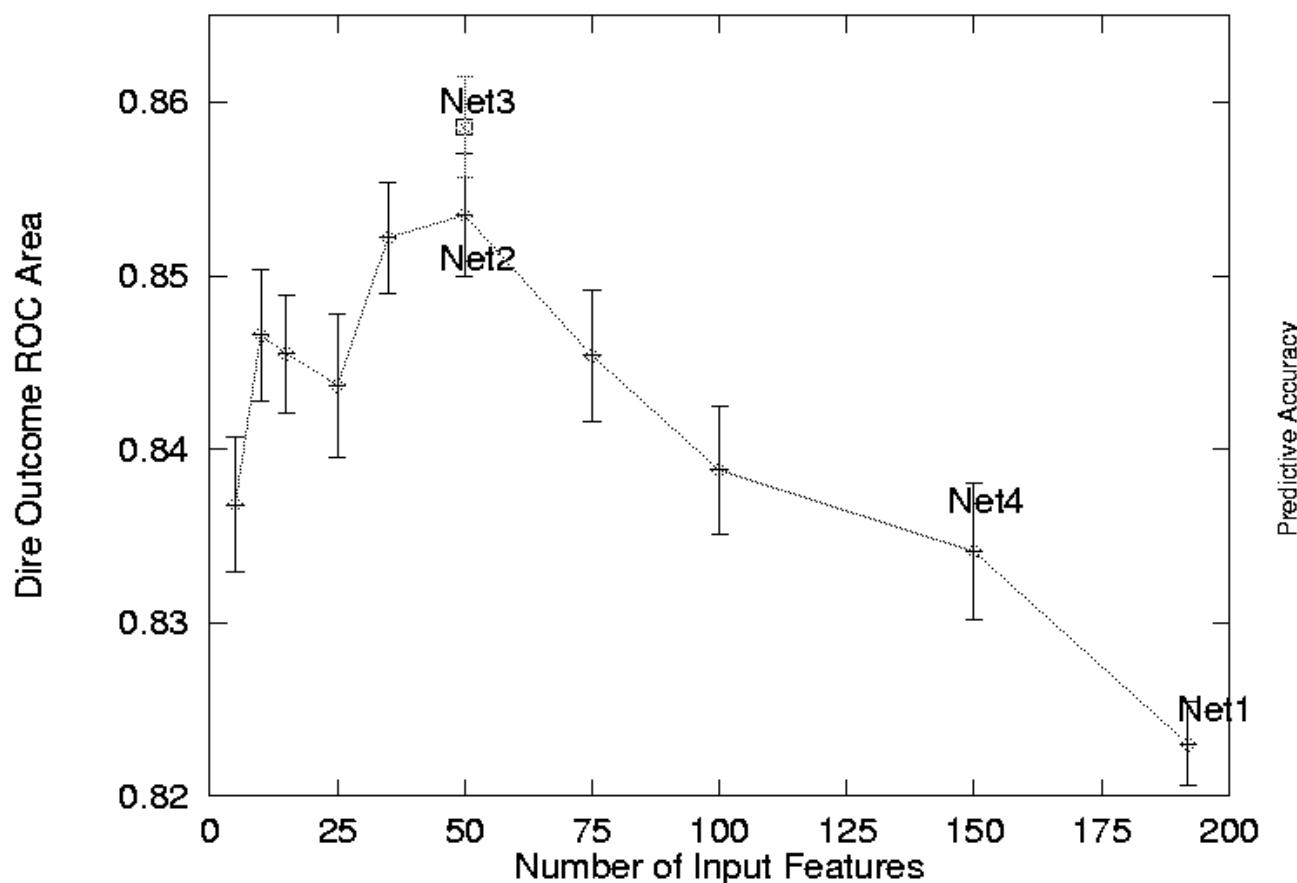
Data Integration

Most learning methods implicitly do feature selection:

- **Decision Trees**: use info gain or gain ratio to decide what attributes to use as tests. Many features don't get used.
- **neural nets**: backprop learns strong connections to some inputs, and near-zero connections to other inputs.
- **kNN (any similarity based learning)**: weights in Weighted Euclidean Distance determine how important each feature is. Weights near zero mean feature is not used.
- **SVMs**: maximum margin hyperplane may focus on important features, ignore irrelevant features.

So why do we need feature **selection**?

Feature Selection and Engineering

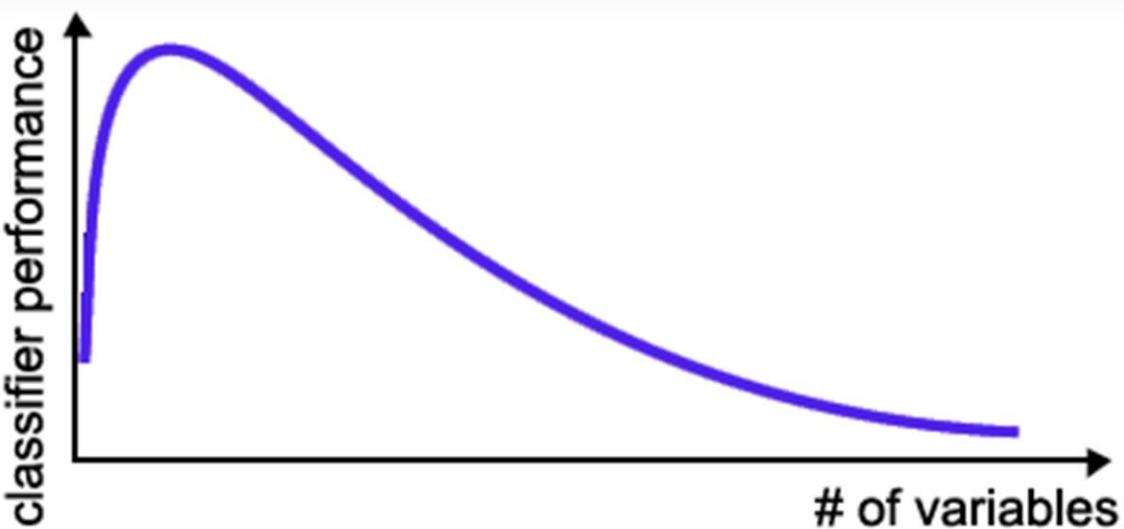


*Too many redundant, irrelevant features confuse learner;
Faster training, better performance, simpler models...*

Feature Selection and Engineering

Curse of Dimensionality

- The required number of samples (to achieve the same accuracy) grows **exponentially** with the number of variables!
- In practice: number of training examples is fixed!
the classifier's performance will degrade for a large number of features!



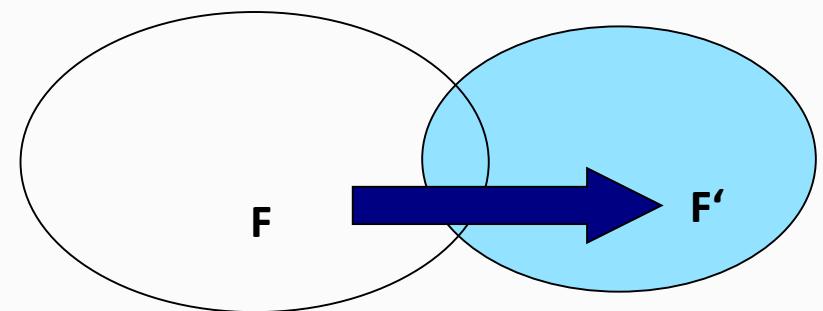
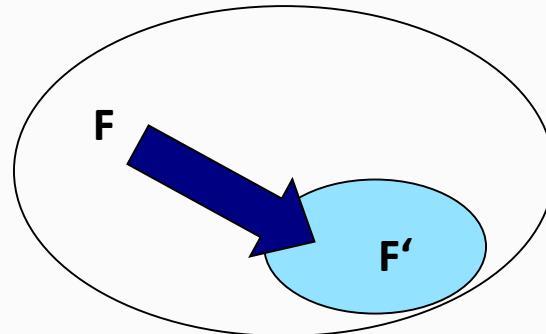
In many cases the information lost by discarding variables is made up for by a more accurate mapping/sampling in the lower-dimensional space !

Feature Selection

Simple Definition

Given a set of features $F = \{f_1, \dots, f_i, \dots, f_n\}$ the feature selection problem is to find a subset $F' \subseteq F$ that “*maximizes the learners ability to classify patterns*”. Feature extraction creates new features $F \rightarrow F'$...

$$\{f_1, \dots, f_i, \dots, f_n\} \xrightarrow{f.\text{selection}} \{f_{i_1}, \dots, f_{i_j}, \dots, f_{i_m}\}$$



$$\{f_1, \dots, f_i, \dots, f_n\} \xrightarrow{f.\text{extraction}} \{g_1(f_1, \dots, f_n), \dots, g_j(f_1, \dots, f_n), \dots, g_m(f_1, \dots, f_n)\}$$

Feature Selection and Engineering

Optimality?

In theory the goal is to find an optimal set of features, one that maximizes the scoring function...

In real world applications this is usually not possible

- For most problems it is computationally intractable to search the whole space of possible feature subsets
- One usually has to settle for approximations of the optimal subset
- Most of the research in this area is devoted to finding efficient search-heuristics



Dealing with Nominal Attributes

Outlook	Temperature	Humidity	Windy	Play
sunny	85	85	false	no
sunny	80	90	true	no
overcast	83	78	false	yes
rain	70	96	false	yes
rain	68	80	false	yes
rain	65	70	true	no
overcast	64	65	true	yes
sunny	72	95	false	no
sunny	69	70	false	yes
rain	75	80	false	yes
sunny	75	70	true	yes
overcast	72	90	true	yes
overcast	81	75	false	yes
rain	71	80	true	no

Standard
Spreadsheet
Format



OutLook	OutLook	OutLook	Temp	Humidity	Windy	Windy	Play	Play
overcast	rain	sunny			TRUE	FALSE	yes	no
0	0	1	85	85	0	1	1	0
0	0	1	80	90	1	0	0	1
1	0	0	83	78	0	1	1	0
0	1	0	70	96	0	1	1	0
0	1	0	68	80	0	1	1	0
0	1	0	65	70	1	0	0	1
1	0	0	64	65	1	0	1	0
.
.

Attributes:

Outlook (overcast, rain, sunny)

Temperature real

Humidity real

Windy (true, false)

Play (yes, no)

Normalization

- Min-max normalization: linear transformation from v to v'
 - $v' = (v - \text{min}) / ((\text{max} - \text{min}) * (\text{newmax} - \text{newmin})) + \text{new min}$
 - Ex: transform \$30000 between [10000..45000] into [0..1]
 $\Rightarrow (30 - 10) / (35(1)) + 0 = 0.5714$
- z-score normalization: normalization of v into v' based on attribute value mean and standard deviation
 - $v' = (v - \text{Mean}) / \text{StandardDeviation}$
- Normalization by decimal scaling
 - moves the decimal point of v by j positions such that j is the minimum number of positions moved so that absolute maximum value falls in [0..1].
 - $v' = v / 10^j$
 - Ex: if v ranges between -56 and 9976, $j=4 \Rightarrow v'$ ranges between -0.0056 and 0.9976

Discretization/Binning

Less features, more discrimination ability

- Discretization is used to reduce the number of values for a given continuous attribute
 - usually done by dividing the range of the attribute into intervals
 - interval labels are then used to replace actual data values
- Discretization can also be used to generate **concept hierarchies**
 - reduce the data by collecting and replacing low level concepts (e.g., numeric values for "age") by higher level concepts (e.g., "young", "middle aged", "old")

Discretization Methods

- Equal-width (distance) partitioning
 - Divides the range into N intervals of equal size: uniform grid
 - The most straightforward, but outliers may dominate presentation
 - Skewed data is not handled well
- Equal-depth (frequency) partitioning
 - Divides the range into N intervals, each containing approximately same number of samples
- Class label based partitioning
 - χ^2 , CAIM, CAIR Discretization
 - Maximum Entropy Discretization

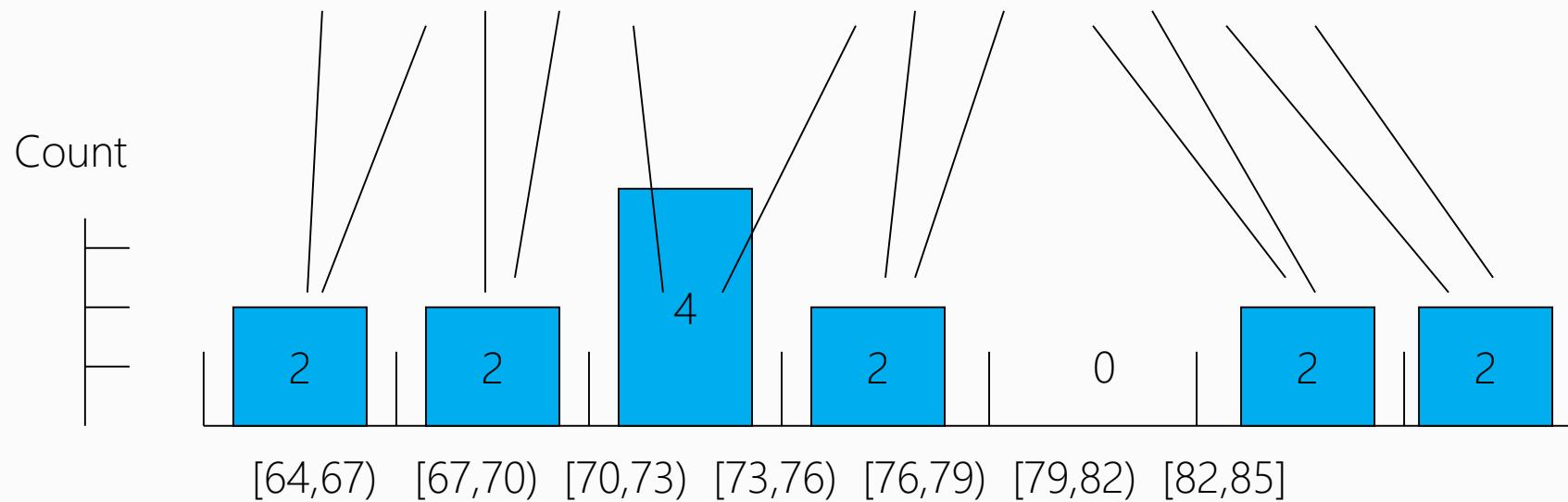
Equal width partitioning

1. Find the minimum and maximum values for the continuous feature/attribute F_i
2. Divide the range of the attribute F_i into the user-specified, n_{F_i} , equal-width discrete intervals

Equal-Width Partitioning

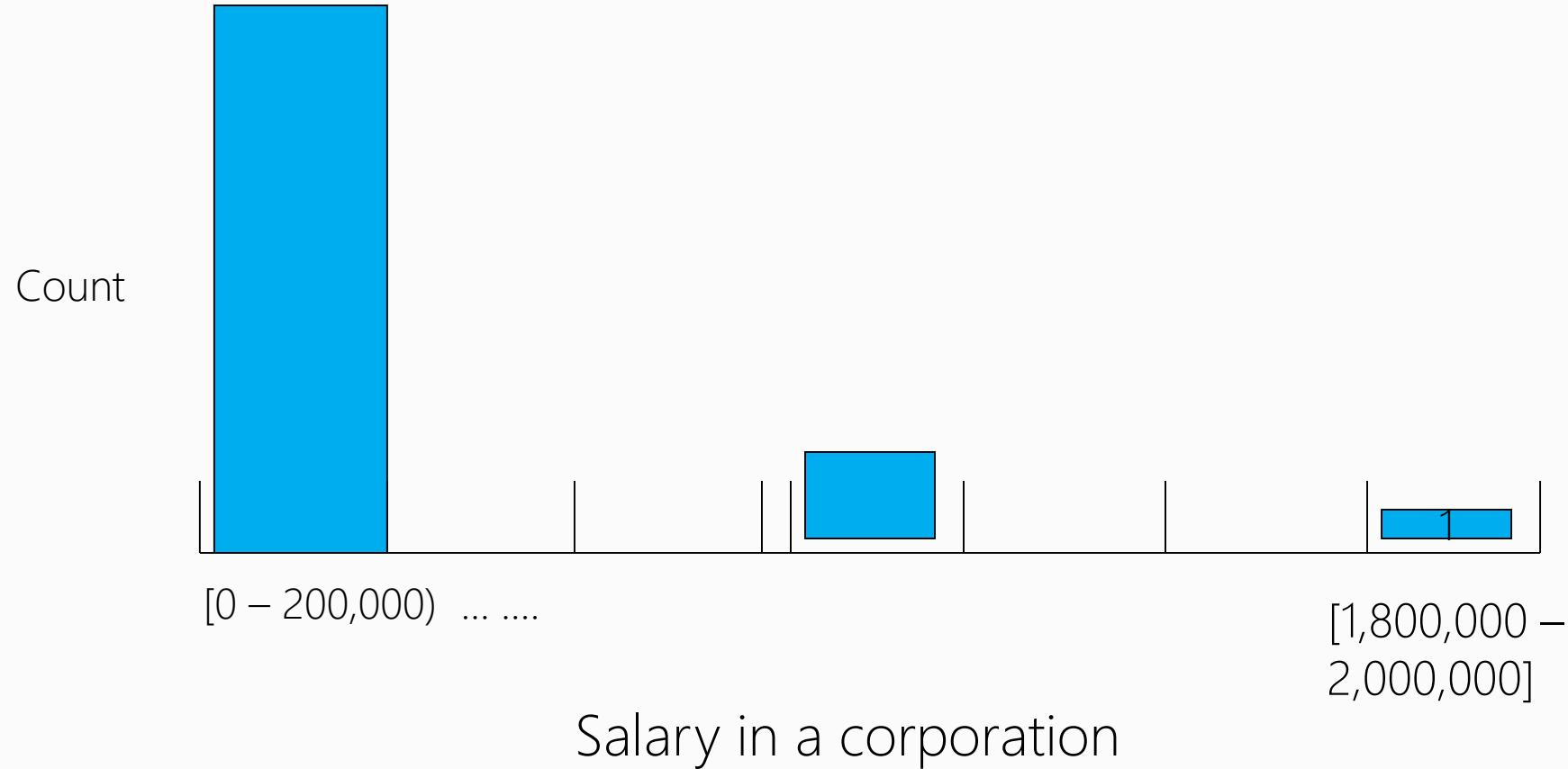
Temperature values:

64 65 68 69 70 71 72 72 75 75 80 81 83 85



Equal Width, bins $\text{Low} \leq \text{value} < \text{High}$

Equal-Width partitioning can produce clumping



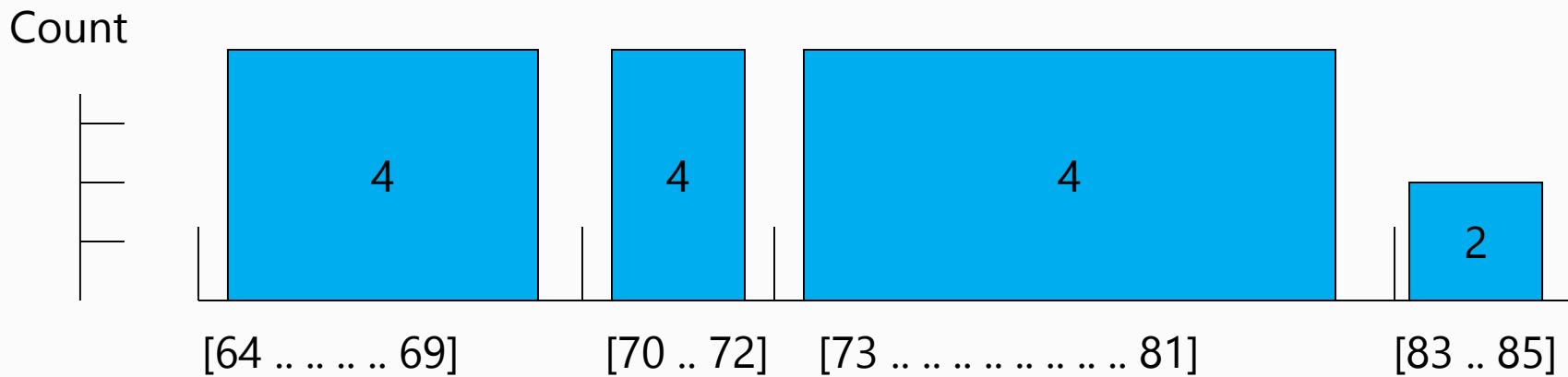
Equal Height partitioning

1. Sort values of the discretized feature F_i in ascending order
2. Find the number of all possible values for feature F_i
3. Divide the values of feature F_i into the user-specified n_{F_i} number of intervals, where each interval contains the same number of sorted sequential values

Equal-Height partitioning

Temperature values:

64 65 68 69 70 71 72 72 75 75 80 81 83 85



Equal Height = 4, except for the last bin

Equal-height partitioning: advantages

- Generally preferred because avoids clumping
- In practice, “almost-equal” height binning is used which avoids clumping and gives more intuitive breakpoints
- Additional considerations:
 - don’t split frequent values across bins
 - create separate bins for special values (e.g. 0)
 - readable breakpoints (e.g. round breakpoints)



Weka Explorer

Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

Choose **None**

Apply

Current relation

Relation: None

Instances: None

Attributes: None

Selected attribute

Name: None

Missing: None

Type: None

Distinct: None

Unique: None

Attributes

All

None

Invert

Pattern



Visualize All

Remove

Status

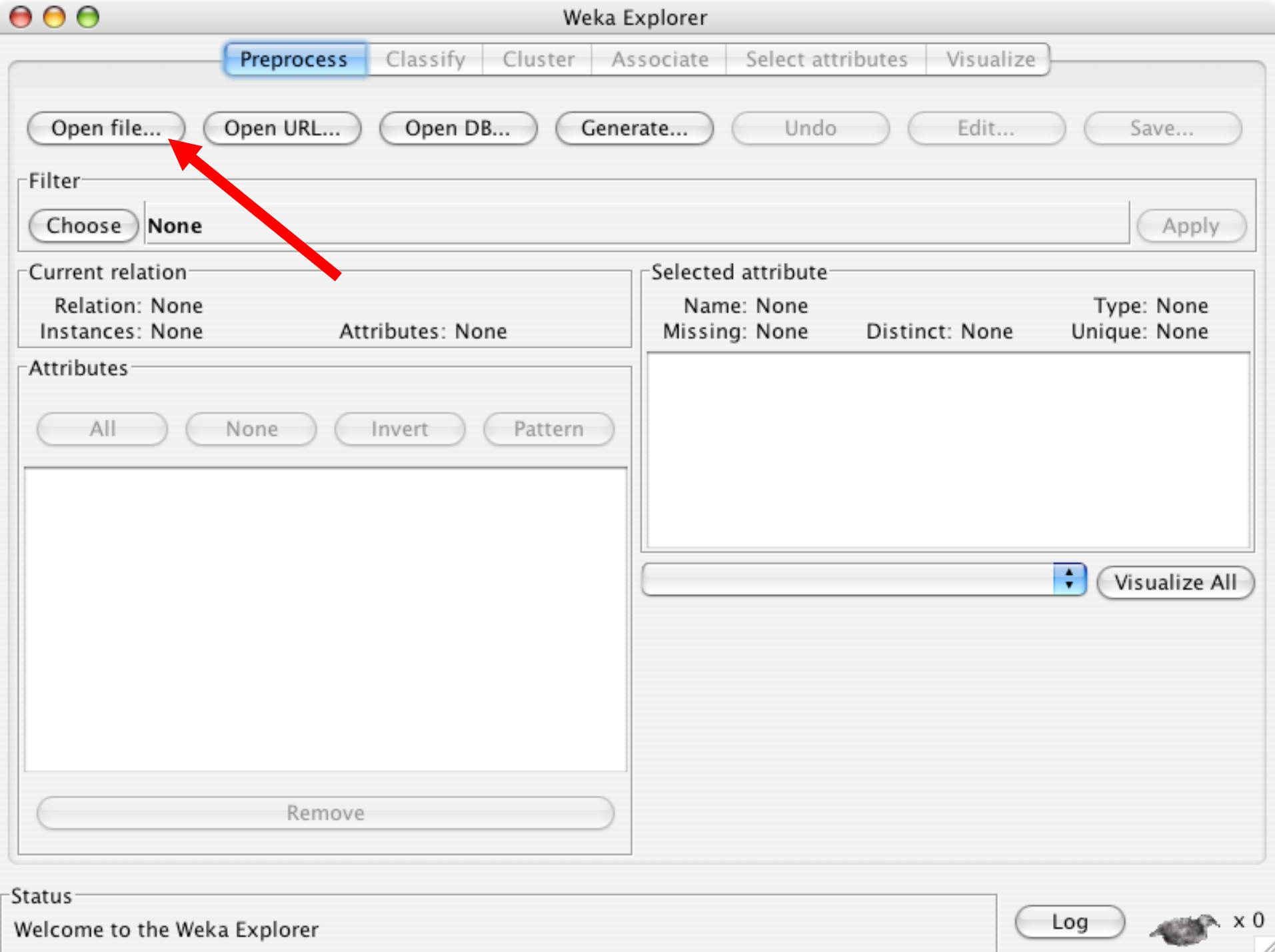
Welcome to the Weka Explorer

Log



Data at Scale







Weka Explorer

Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

Choose **None**

Apply

Current relation

Relation: iris

Instances: 150

Attributes: 5

Attributes

All

None

Invert

Pattern

No.	Name
1	<input checked="" type="checkbox"/> sepallength
2	<input type="checkbox"/> sepalwidth
3	<input type="checkbox"/> petallength
4	<input type="checkbox"/> petalwidth
5	<input type="checkbox"/> class

Remove

Selected attribute

Name: sepallength

Type: Numeric

Missing: 0 (0%)

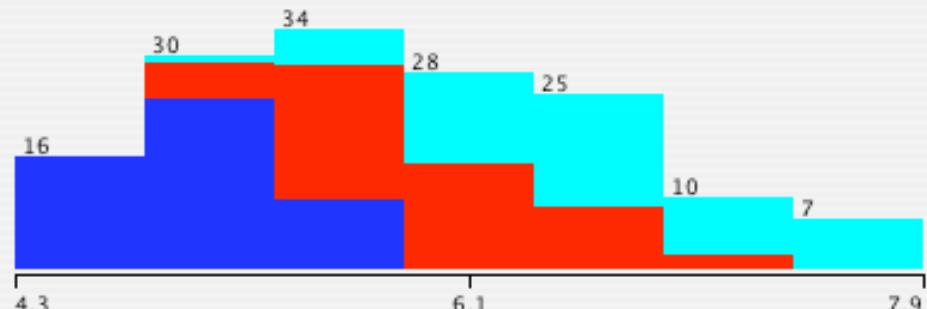
Distinct: 35

Unique: 9 (6%)

Statistic	Value
Minimum	4.3
Maximum	7.9
Mean	5.843
StdDev	0.828

Class: class (Nom)

Visualize All



Status

OK

Log



Data at Scale





Weka Explorer

Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

Choose

None

Apply

Current relation

Relation: iris

Instances: 150

Attributes: 5

Attributes

All

None

Invert

Pattern

No.	Name
1	<input checked="" type="checkbox"/> sepallength
2	<input type="checkbox"/> sepalwidth
3	<input type="checkbox"/> petallength
4	<input type="checkbox"/> petalwidth
5	<input type="checkbox"/> class

Remove

Status

OK

Selected attribute

Name: sepallength

Type: Numeric

Missing: 0 (0%)

Distinct: 35

Unique: 9 (6%)

Statistic	Value
Minimum	4.3
Maximum	7.9
Mean	5.843
StdDev	0.828

Class: class (Nom)

Visualize All

A histogram showing the distribution of Sepal Length for the Iris dataset. The x-axis represents Sepal Length, ranging from 4.3 to 7.9. The y-axis represents frequency. The distribution is skewed right, with most values falling between 4.3 and 6.5. The histogram is composed of three stacked bars: blue (Sepal Length < 5.0), red (5.0 <= Sepal Length < 6.5), and cyan (Sepal Length >= 6.5). The counts for each segment are labeled above the bars: 16 (blue), 30 (red), 34 (cyan), 28 (red), 25 (cyan), 10 (red), and 7 (cyan).

Log

Data at Scale



Weka Explorer

Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

Choose

None

Apply

Current relation

Relation: iris

Instances: 150

Attributes: 5

Attributes

All

None

Invert

Pattern

No.	Name
1	<input type="checkbox"/> sepallength
2	<input type="checkbox"/> sepalwidth
3	<input type="checkbox"/> petallength
4	<input type="checkbox"/> petalwidth
5	<input checked="" type="checkbox"/> class

Remove

Status

OK

Selected attribute

Name: class

Missing: 0 (0%)

Type: Nominal

Distinct: 3

Unique: 0 (0%)

Label	Count
Iris-setosa	50
Iris-versicolor	50
Iris-virginica	50

Class: class (Nom)

Visualize All



Log



Data at Scale





Weka Explorer

Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

Choose None

Apply

Current relation

Relation: iris

Instances: 150

Attributes: 5

Attributes

All

None

Invert

Pattern

No.	Name
1	sepallength
2	sepalwidth
3	petallength
4	petalwidth
5	class

Remove

Selected attribute

Name: class

Missing: 0 (0%)

Type: Nominal

Distinct: 3

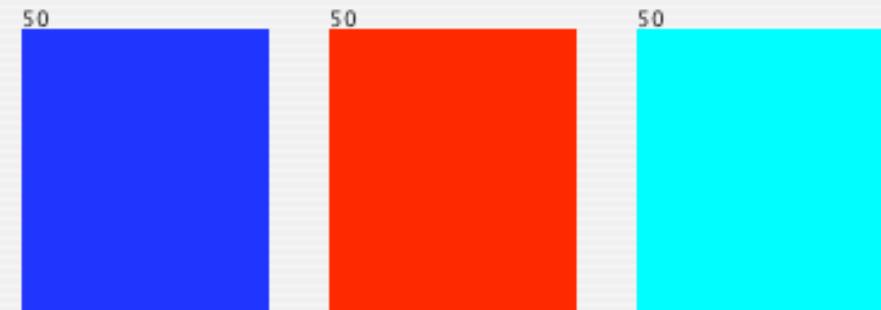
Unique: 0 (0%)

Label	Count
Iris-setosa	50
Iris-versicolor	50
Iris-virginica	50

Class: class (Nom)



Visualize All



Status

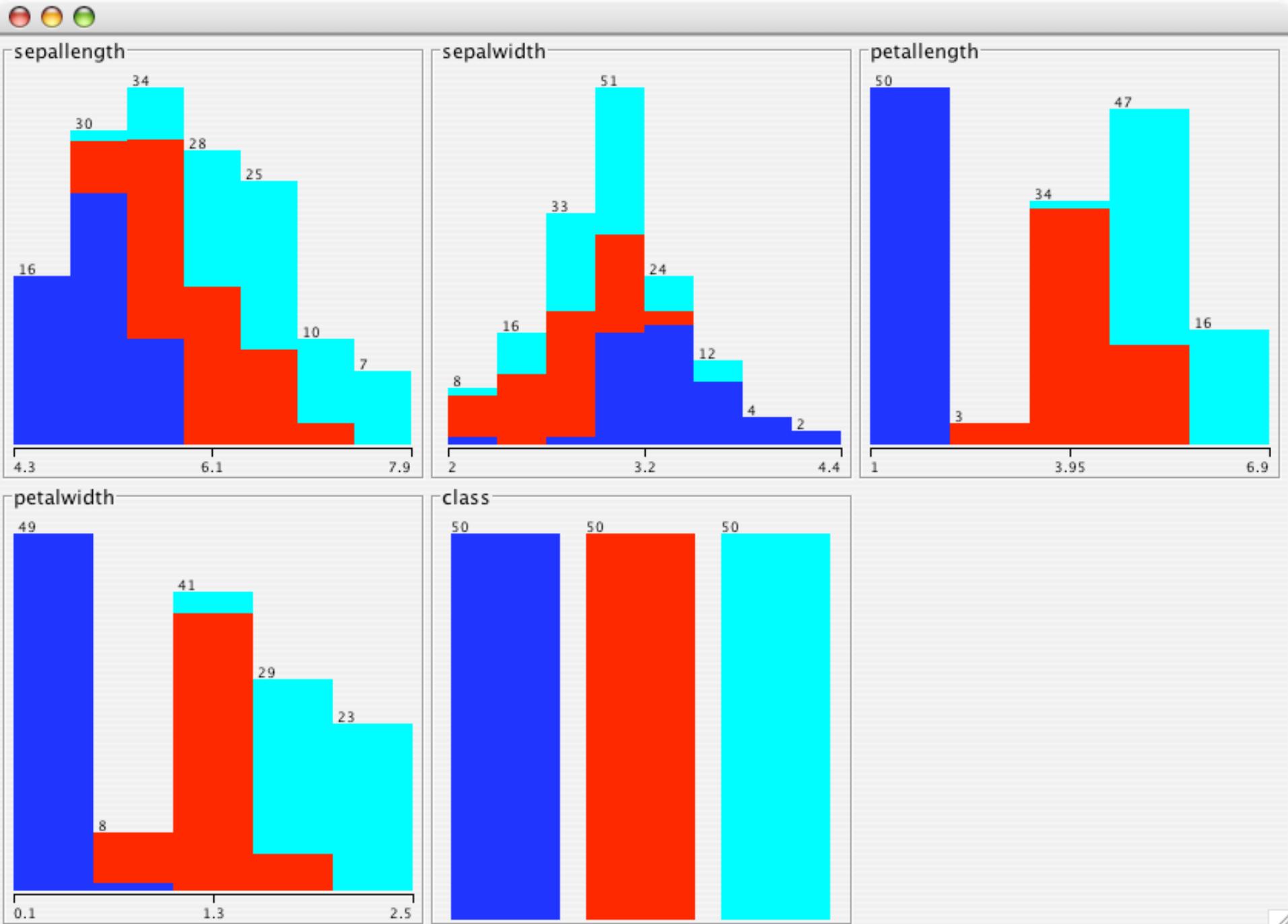
OK

Log



Data at Scale







Weka Explorer

Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

Choose **None**

Apply

Current relation

Relation: iris

Instances: 150

Attributes: 5

Attributes

All

None

Invert

Pattern

No.	Name
1	sepallength
2	sepalwidth
3	petallength
4	petalwidth
5	class

Remove

Selected attribute

Name: petallength

Type: Numeric

Missing: 0 (0%)

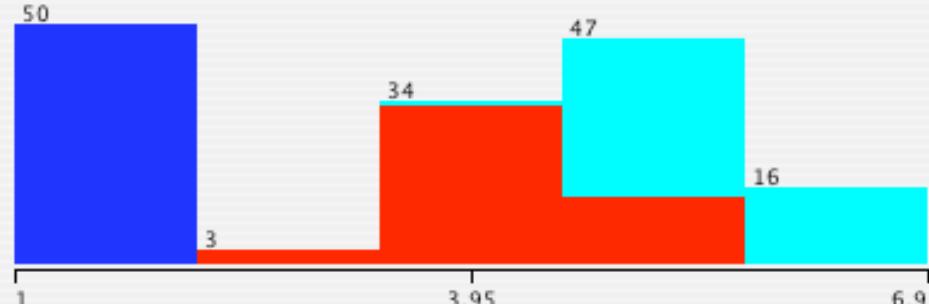
Distinct: 43

Unique: 10 (7%)

Statistic	Value
Minimum	1
Maximum	6.9
Mean	3.759
StdDev	1.764

Class: class (Nom)

Visualize All



Status

OK

Log



Data at Scale





Weka Explorer

Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

Choose None

Apply

Current relation

Relation: iris

Instances: 150

Attributes: 5

Attributes

All

None

Invert

Pattern

No.	Name
1	sepallength
2	sepalwidth
3	<input checked="" type="checkbox"/> petallength
4	petalwidth
5	class

Remove

Selected attribute

Name: petallength

Missing: 0 (0%)

Type: Numeric

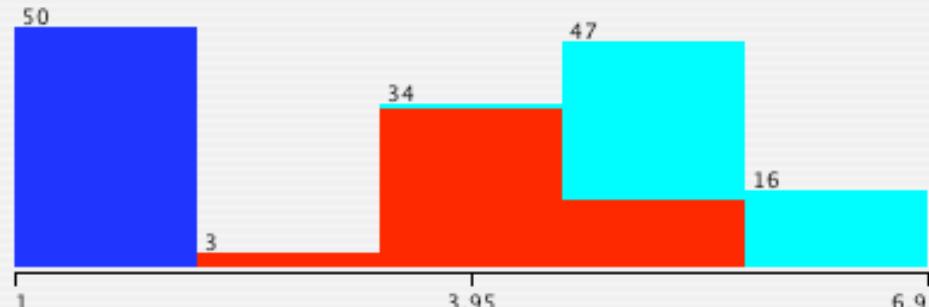
Distinct: 43

Unique: 10 (7%)

Statistic	Value
Minimum	1
Maximum	6.9
Mean	3.759
StdDev	1.764

Class: class (Nom)

Visualize All



Status

OK

Log



Data at Scale



Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

weka None

filters

Current: AllFilter

Ref: MultiFilter

Inst: supervised

Attributes: 5

unsupervised

attribute

instance None

Invert

No.	Name
1	sepallength
2	sepalwidth
3	petallength
4	petalwidth
5	class

Pattern

Remove

Filter...

Remove filter

Close

Selected attribute

Name: petallength

Type: Numeric

Missing: 0 (0%)

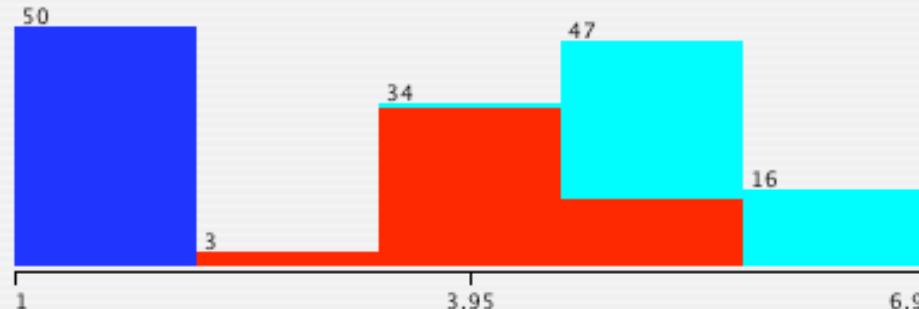
Distinct: 43

Unique: 10 (7%)

Statistic	Value
Minimum	1
Maximum	6.9
Mean	3.759
StdDev	1.764

Class: class (Nom)

Visualize All



OK

Log



Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

weka None

filters

Current: AllFilter

Ref: MultiFilter

Inst: Supervised

Attributes: 5

unsupervised

attribute

instance

No.	Name
1	sepallength
2	sepalwidth
3	petallength
4	petalwidth
5	class

Invert

Filter...

Remove filter

Close

Pattern

Selected attribute

Name: petallength

Type: Numeric

Missing: 0 (0%)

Distinct: 43

Unique: 10 (7%)

Statistic

Value

Minimum

1

Maximum

6.9

Mean

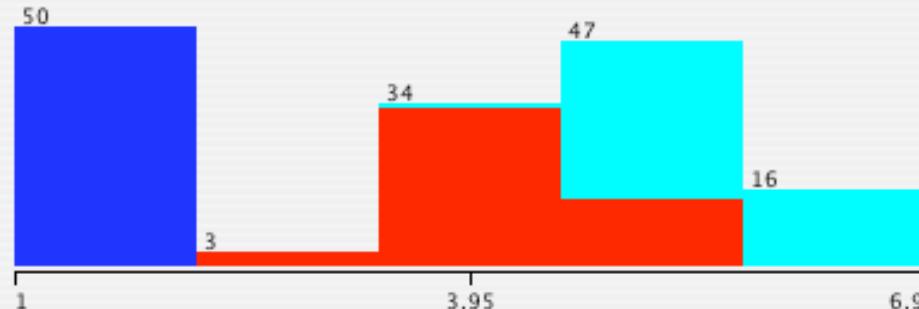
3.759

StdDev

1.764

Class: class (Nom)

Visualize All



OK

Log



Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

weka Discretize -B 10 -M 1.0 -R first

filters

AllFilter

MultiFilter

supervised

unsupervised

attribute

All Add None Invert

No.	Name
1	AddNone
2	AddCluster
3	AddExpression
4	AddID
5	AddNoise
6	AddValues
7	Center
8	ChangeDateFormat
9	ClassAssigner
10	ClusterMembership
11	Copy
12	Discretize
13	FirstOrder
14	InterquartileRange

Remove Filter... Remove Close

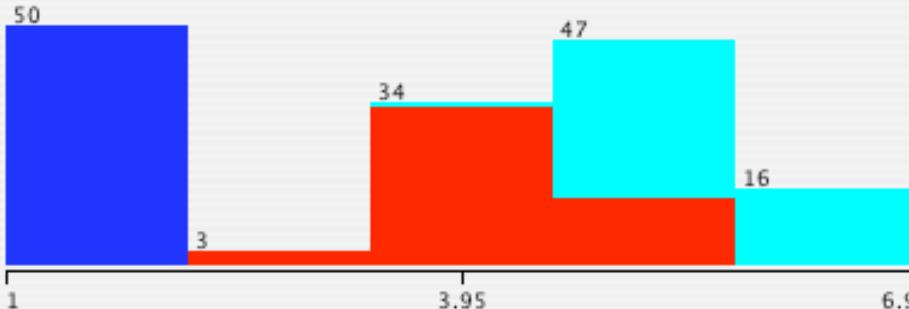
Selected attribute

Name: petallength Type: Numeric
Missing: 0 (0%) Distinct: 43 Unique: 10 (7%)

Statistic	Value
Minimum	1
Maximum	6.9
Mean	3.759
StdDev	1.764

Class: class (Nom)

Visualize All



OK

Log



Data at Scale





Weka Explorer

Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

Choose

Discretize -B 10 -M -1.0 -R first-last

Apply

Current relation

Relation: iris

Instances: 150

Attributes: 5

Attributes

All

None

Invert

Pattern

No.	Name
1	sepallength
2	sepalwidth
3	<input checked="" type="checkbox"/> petallength
4	petalwidth
5	class

Remove

Selected attribute

Name: petallength

Missing: 0 (0%)

Type: Numeric

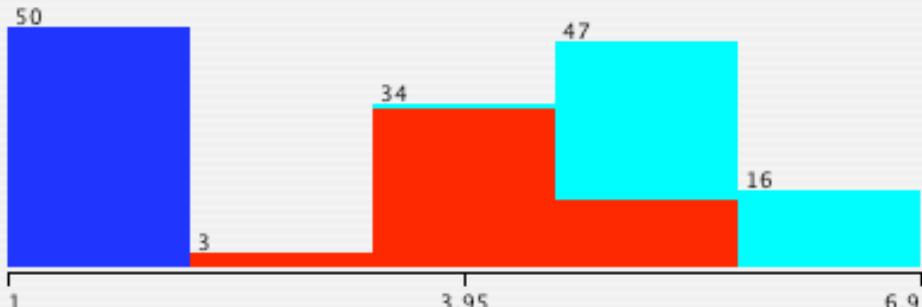
Distinct: 43

Unique: 10 (7%)

Statistic	Value
Minimum	1
Maximum	6.9
Mean	3.759
StdDev	1.764

Class: class (Nom)

Visualize All



Status

OK

Log



Data at Scale



Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Open file... Open URL... Open DB... Generate... Undo Edit... Save...

Filter

Choose Discretize -B 10 -M -1.0 -R first-last Apply

Current relation

Relation: iris Instances: 150 Attributes: 5

Attributes

All None Invert Pattern

No.	Name
1	sepallength
2	sepalwidth
3	petallength
4	petalwidth
5	class

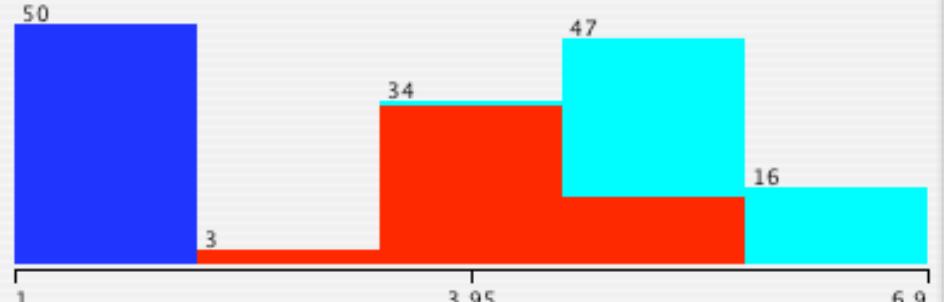
Remove

Selected attribute

Name: petallength Type: Numeric
Missing: 0 (0%) Distinct: 43 Unique: 10 (7%)

Statistic	Value
Minimum	1
Maximum	6.9
Mean	3.759
StdDev	1.764

Class: class (Nom) Visualize All



Status OK Log x 0 Data at Scale W

[Preprocess](#) [Classify](#) [Cluster](#) [Associate](#) [Select attributes](#) [Visualize](#)[Open file...](#)[Open URL...](#)[Open DB...](#)[Generate...](#)[Undo](#)[Edit...](#)[Save...](#)

Filter

[Choose](#)[Discretize](#)

Current relation

Relation: iris

Instances: 150

Attributes

[All](#)[No](#)

No.

- 1 sepallength
- 2 sepalwidth
- 3 petallength**
- 4 petalwidth
- 5 class

weka.gui.GenericObjectEditor

weka.filters.unsupervised.attribute.Discretize

About

An instance filter that discretizes a range of numeric attributes in the dataset into nominal attributes.

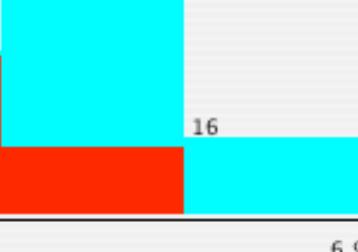
[More](#)[Capabilities](#)Type: Numeric
Unique: 10 (7%)

Value

9

4

47

[Visualize All](#)

attributeIndices first-last

bins 10

desiredWeightOfInstancesPerInterval -1.0

findNumBins False

ignoreClass False

invertSelection False

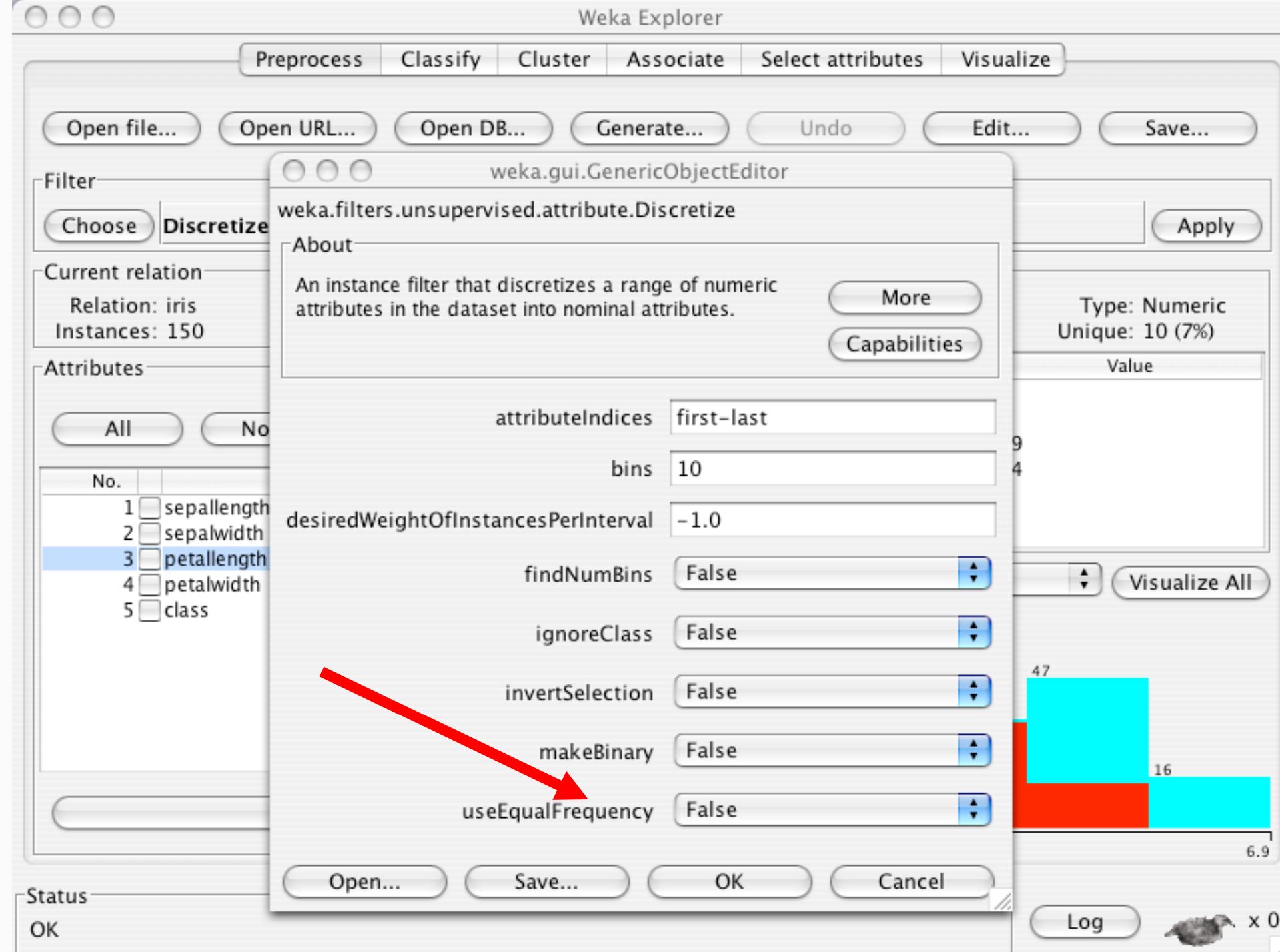
makeBinary False

useEqualFrequency False

[Open...](#)[Save...](#)[OK](#)[Cancel](#)[Log](#)

Status

OK



[Preprocess](#) [Classify](#) [Cluster](#) [Associate](#) [Select attributes](#) [Visualize](#)[Open file...](#)[Open URL...](#)[Open DB...](#)[Generate...](#)[Undo](#)[Edit...](#)[Save...](#)

Filter

[Choose](#) **Discretize**

Current relation

Relation: iris

Instances: 150

Attributes

[All](#)[No](#)

No.

- 1 sepallength
- 2 sepalwidth
- 3 petallength**
- 4 petalwidth
- 5 class

weka.gui.GenericObjectEditor

weka.filters.unsupervised.attribute.Discretize

About

An instance filter that discretizes a range of numeric attributes in the dataset into nominal attributes.

[More](#)[Capabilities](#)Type: Numeric
Unique: 10 (7%)

Value

9

4

[Visualize All](#)

47

16

6.9

attributeIndices first-last

bins 10

desiredWeightOfInstancesPerInterval -1.0

findNumBins False

ignoreClass False

invertSelection False

makeBinary False

useEqualFrequency True

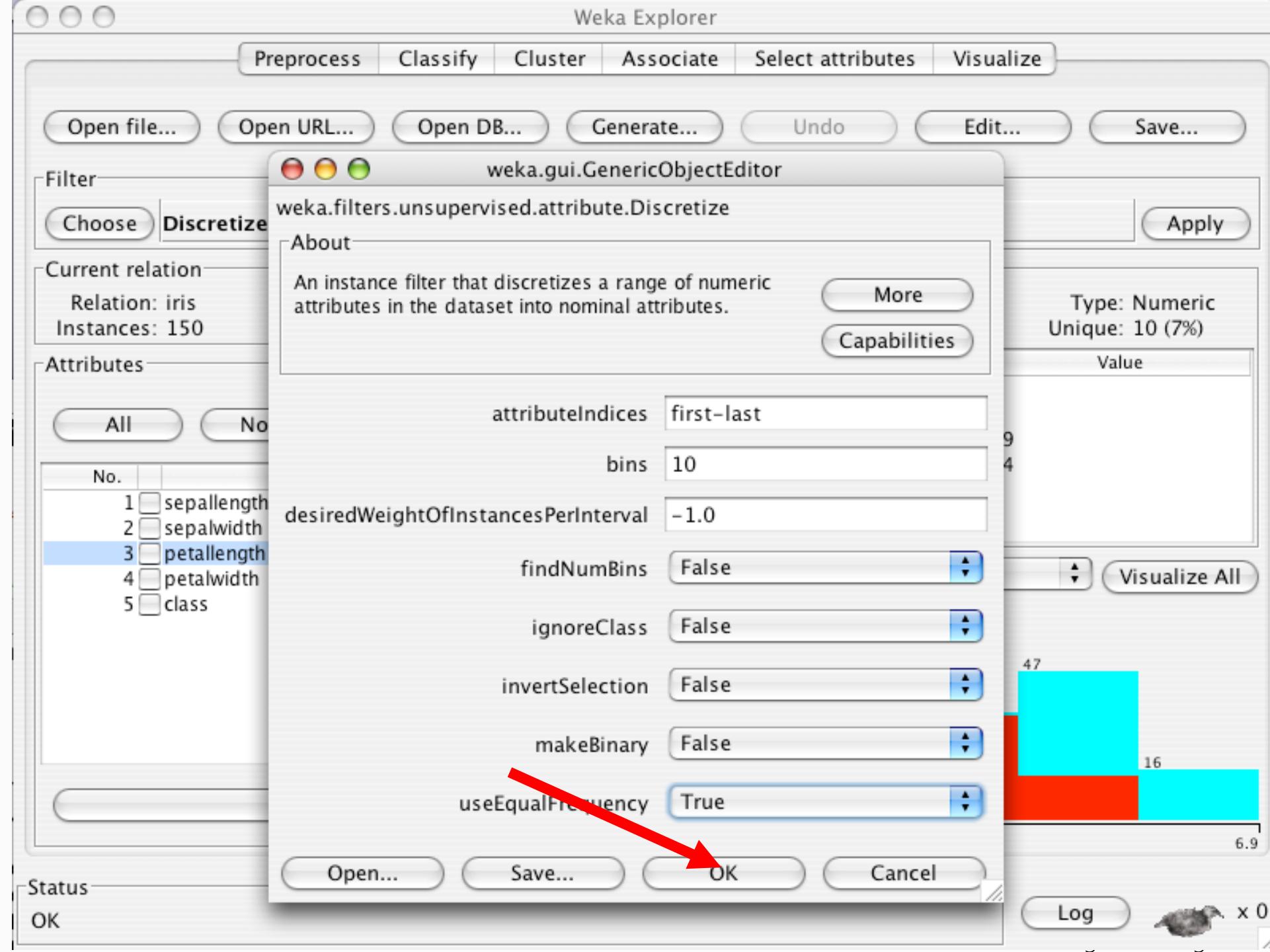
[Open...](#)[Save...](#)[OK](#)[Cancel](#)

Status

OK

Log







Weka Explorer

Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

Choose

Discretize -F -B 10 -M -1.0 -R first-last

Apply

Current relation

Relation: iris

Instances: 150

Attributes: 5

Attributes

All

None

Invert

Pattern

No.	Name
1	sepallength
2	sepalwidth
3	<input checked="" type="checkbox"/> petallength
4	petalwidth
5	class

Remove

Selected attribute

Name: petallength

Missing: 0 (0%)

Type: Numeric

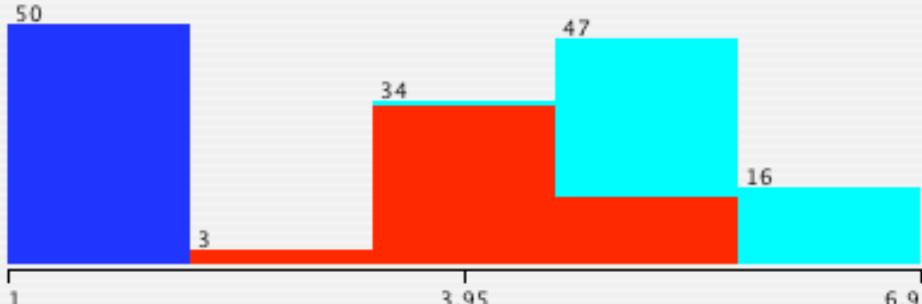
Distinct: 43

Unique: 10 (7%)

Statistic	Value
Minimum	1
Maximum	6.9
Mean	3.759
StdDev	1.764

Class: class (Nom)

Visualize All



Status

OK

Log



Data at Scale





Weka Explorer

Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

Choose

Discretize -F -B 10 -M -1.0 -R first-last

Apply

Current relation

Relation: iris

Instances: 150

Attributes: 5

Attributes

All

None

Invert

Pattern

No.	Name
1	sepallength
2	sepalwidth
3	<input checked="" type="checkbox"/> petallength
4	petalwidth
5	class

Remove

Selected attribute

Name: petallength

Type: Numeric

Missing: 0 (0%)

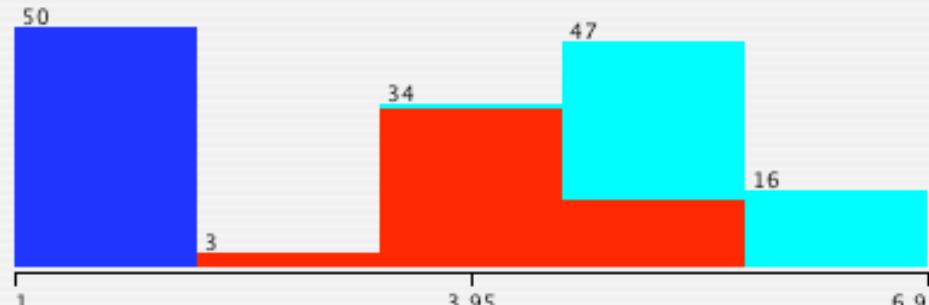
Distinct: 43

Unique: 10 (7%)

Statistic	Value
Minimum	1
Maximum	6.9
Mean	3.759
StdDev	1.764

Class: class (Nom)

Visualize All



Status

OK

Log





Weka Explorer

Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Open file...

Open URL...

Open DB...

Generate...

Undo

Edit...

Save...

Filter

Choose

Discretize -F -B 10 -M -1.0 -R first-last

Apply

Current relation

Relation: iris-weka.filters.unsupervised.attribute.Discretize
Instances: 150 Attributes: 5

Attributes

All

None

Invert

Pattern

No.	Name
1	sepallength
2	sepalwidth
3	<input checked="" type="checkbox"/> petallength
4	petalwidth
5	class

Remove

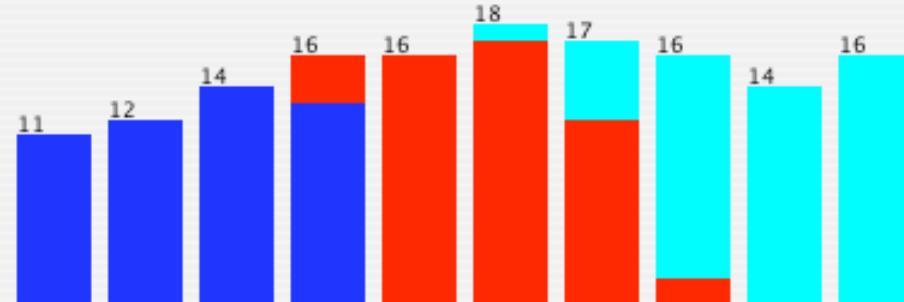
Selected attribute

Name: petallength Type: Nominal
Missing: 0 (0%) Distinct: 10 Unique: 0 (0%)

Label	Count
'(-inf-1.35]'	11
'(1.35-1.45]'	12
'(1.45-1.55]'	14
'(1.55-3.4]'	16
'(3.4-4.15]'	16
'(4.15-4.55]'	18
'(4.55-4.85]'	17

Class: class (Nom)

Visualize All



Status

OK

Log



Data at Scale



Derived Variables

- Better to have a fair modeling method and good variables, than to have the best modeling method and poor variables
- Insurance Example: People are eligible for pension withdrawal at age 59 ½. Create it as a separate Boolean variable!
- Advanced methods exist for automatically examining variable combinations, but they can be computationally very expensive!

Special Transformations

Domain expertise, play a hunch in terms of feature discrimination

Example: *Date/Time* attribute

- Time of a day
- Day of the week
- Day of the month
- Month of the year
- Day of the year
- Quarter of the year
- A holiday or not

Which ones to use depends on the prediction problem being solved

- Ex: For prediction of traffic on a freeway, Time of day, Day of the week, A holiday or not etc. will be useful

Data Science

Deriving Knowledge from Data at Scale

That's all for tonight....

Valentine N. Fontama

August 6th , 2015

Deriving Knowledge from Data at Scale



Support Vector Machines



Support vector machines

Similar to linear regression they aim to find a hyperplane that linearly separates data points belong to different classes

In addition SVMs aim to find the hyperplane that is least likely to over fit the training data

- By design, similar to pruning in Decision Trees, SVMs attempt to regulate the hypothesis space to ensure good performance on validation set...
- Fast in the nonlinear case
 - Use a mathematical trick to avoid creating “pseudoattributes”
 - The nonlinear space is created implicitly

SVM, in a nutshell...

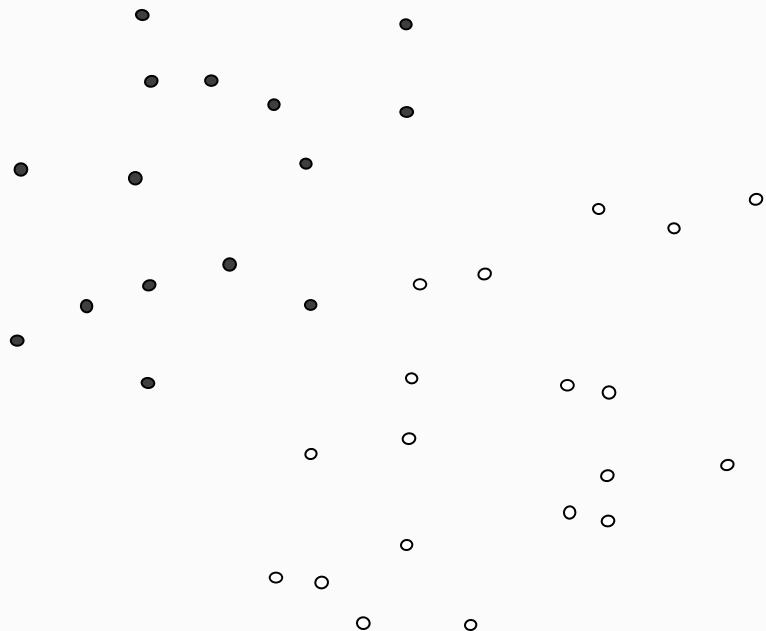
- SVM views the input data as two sets of vectors in an n-dimensional space. It constructs a separating hyperplane in that space, one which maximizes the margin between the two data sets.
- To calculate the margin, two parallel hyperplanes are constructed, one on each side of the separating hyperplane.
- A good separation is achieved by the hyperplane that has the largest distance to the neighboring data points of both classes.
- The vectors (points) that constrain the width of the margin are the support vectors.

Linear Classifiers

Estimation:



- denotes +1
- denotes -1

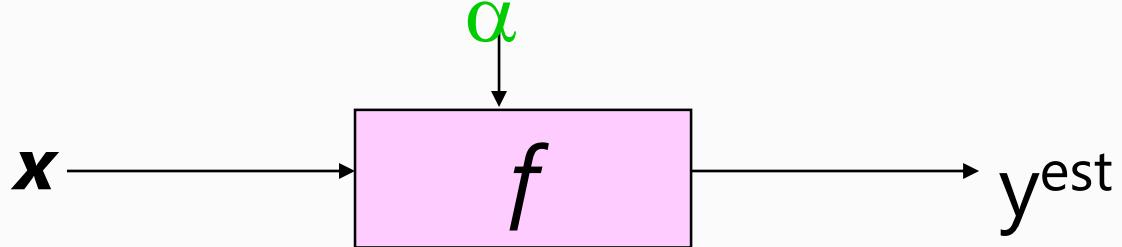


$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

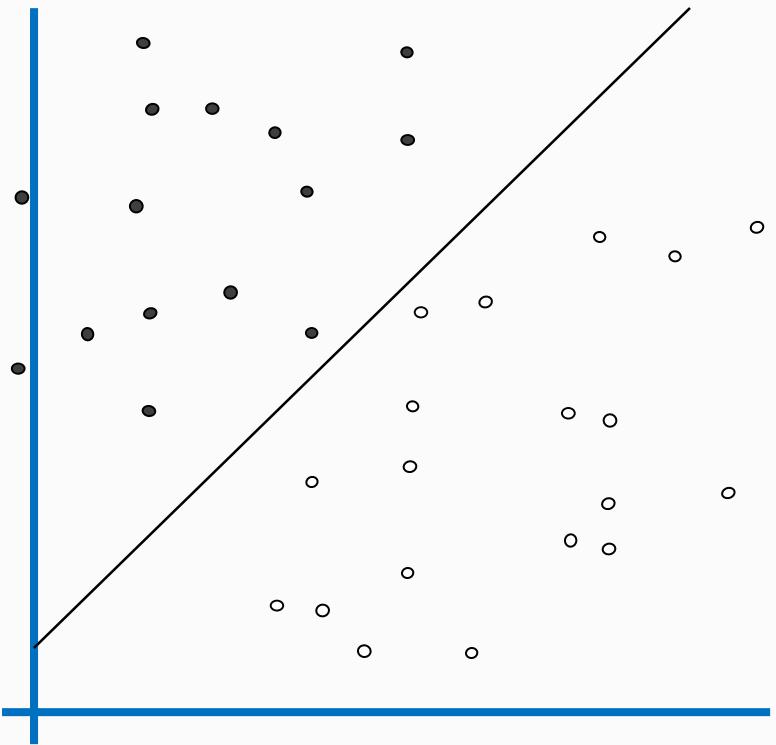
\mathbf{w} : weight vector
 \mathbf{x} : data vector

How would you
classify this data?

Linear Classifiers



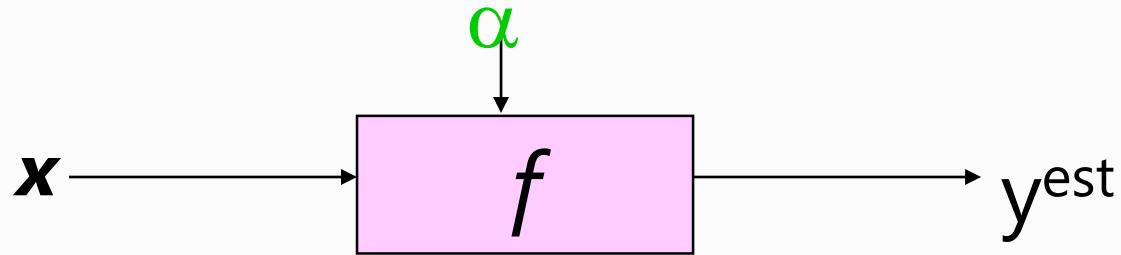
- denotes +1
- denotes -1



$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

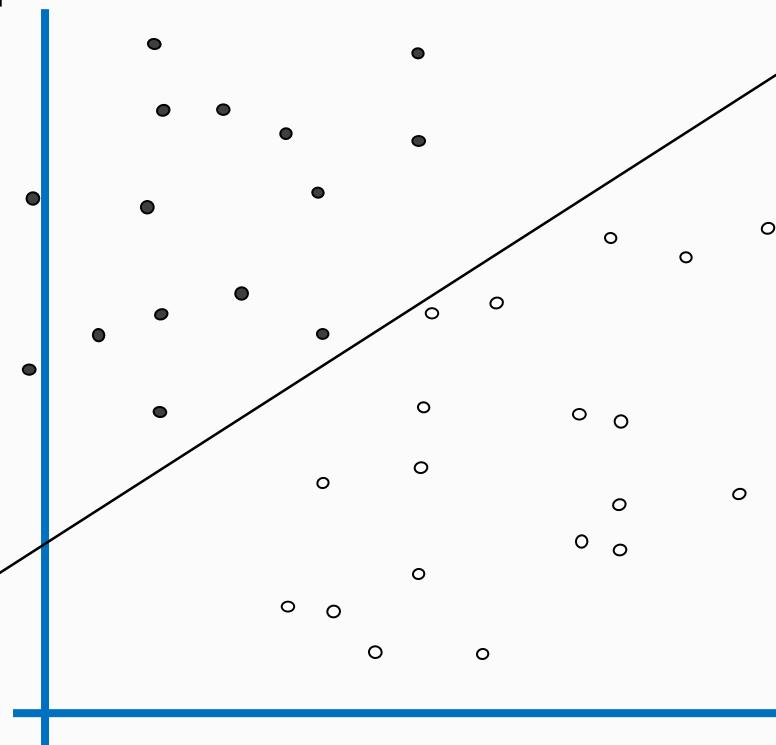
How would you
classify this data?

Linear Classifiers



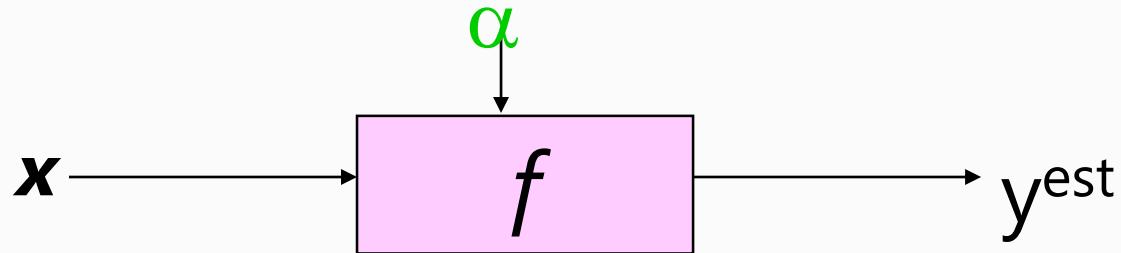
- denotes +1
- denotes -1

$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

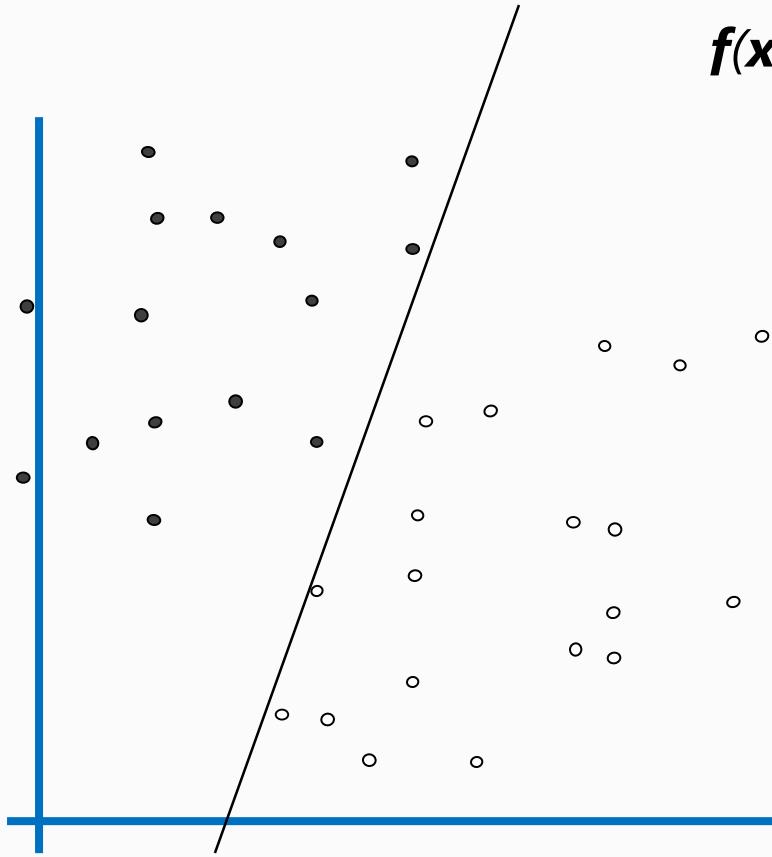


How would you
classify this data?

Linear Classifiers



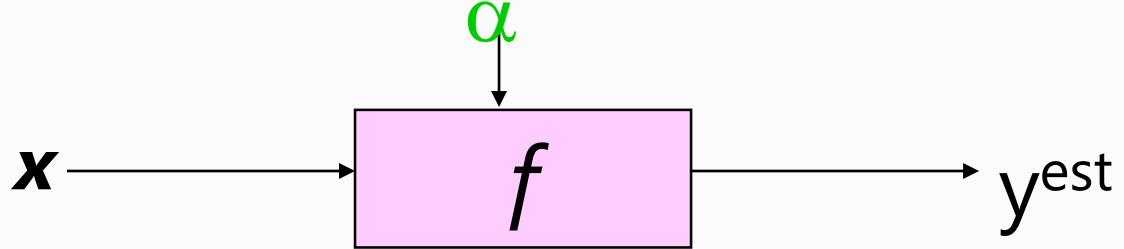
- denotes +1
- denotes -1



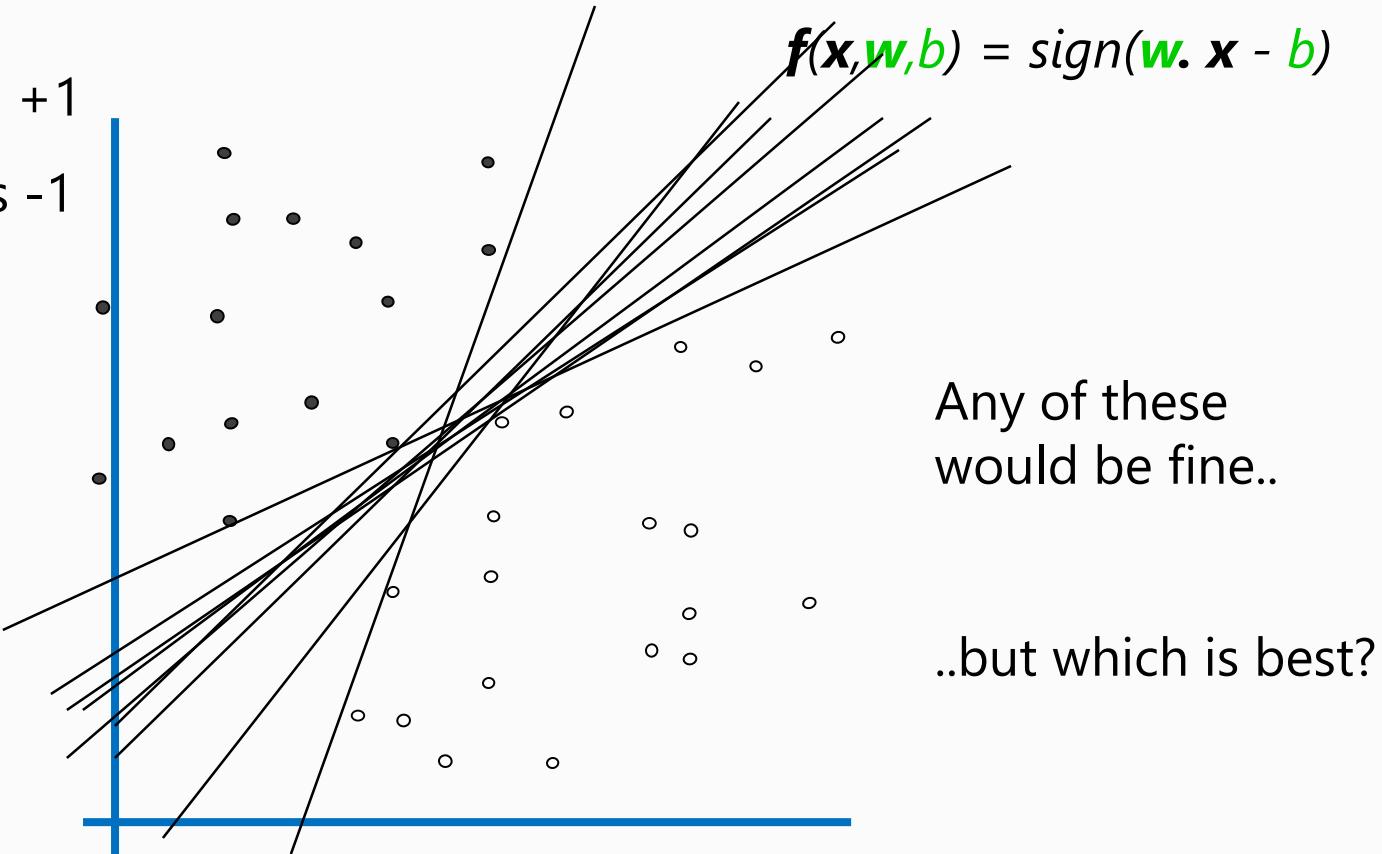
$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

How would you
classify this data?

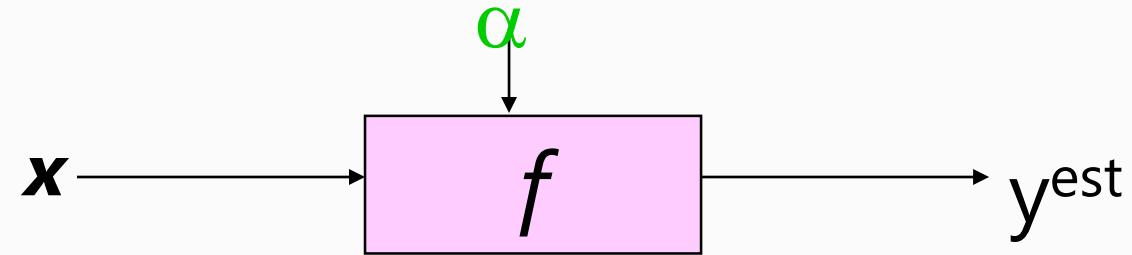
Linear Classifiers



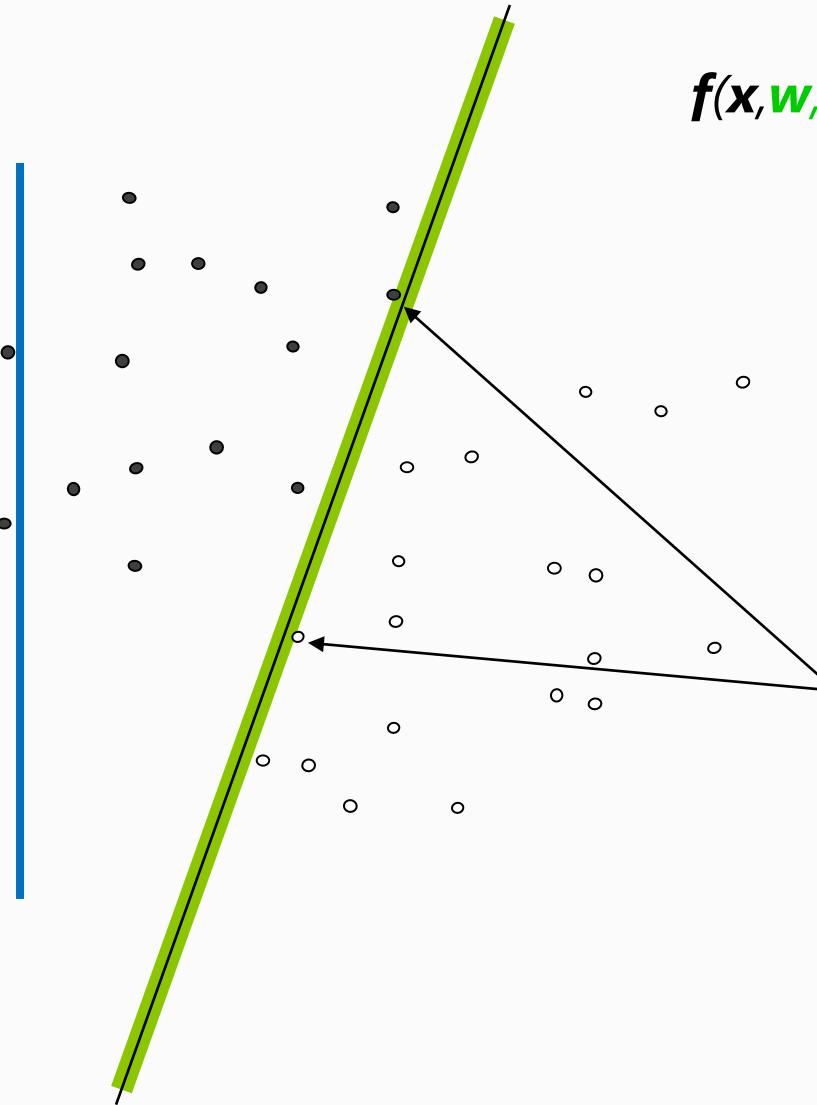
- denotes +1
- denotes -1



Classifier Margin



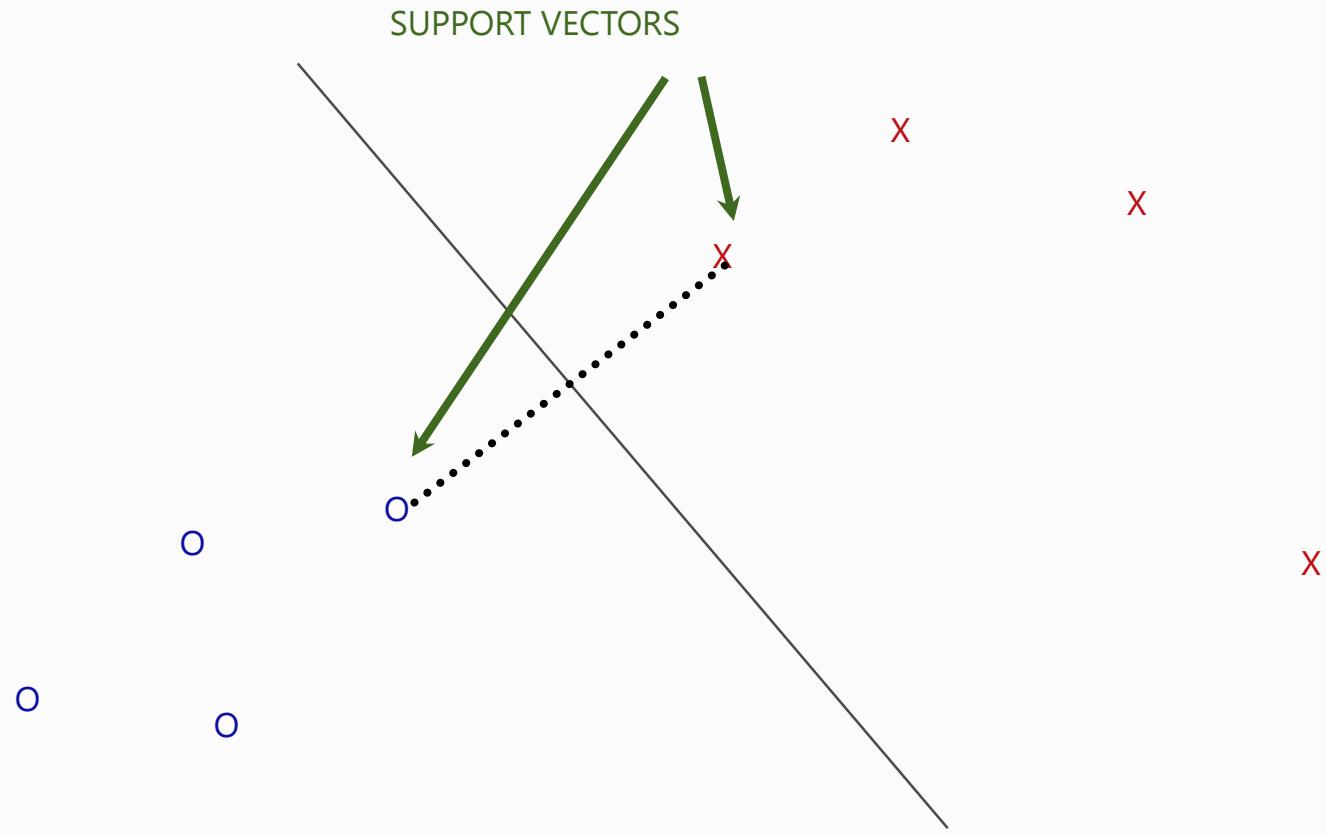
- denotes +1
- denotes -1



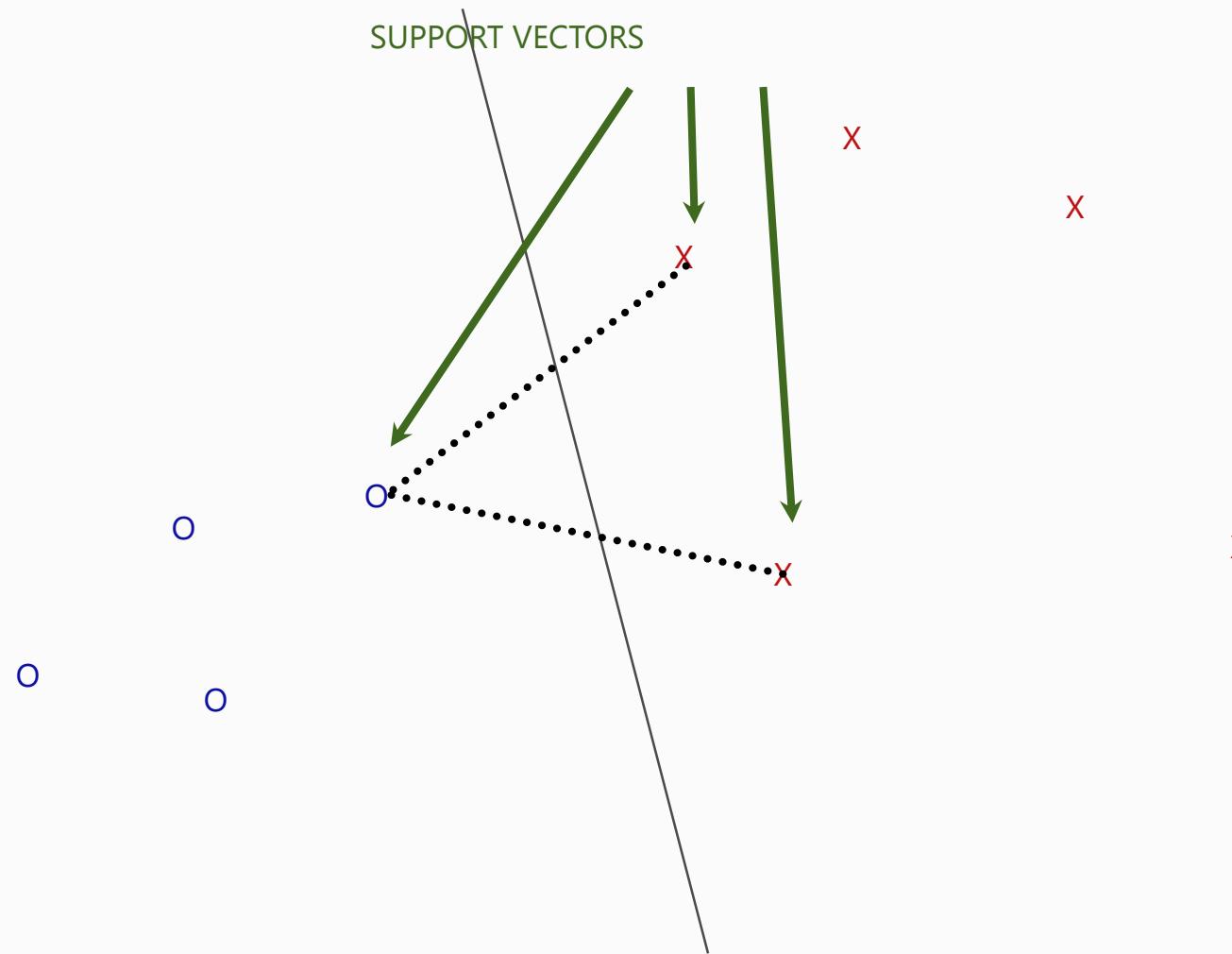
$$\mathbf{f}(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

Define the **margin** of a linear classifier as the width that the boundary could be increased by before hitting a **datapoint**.

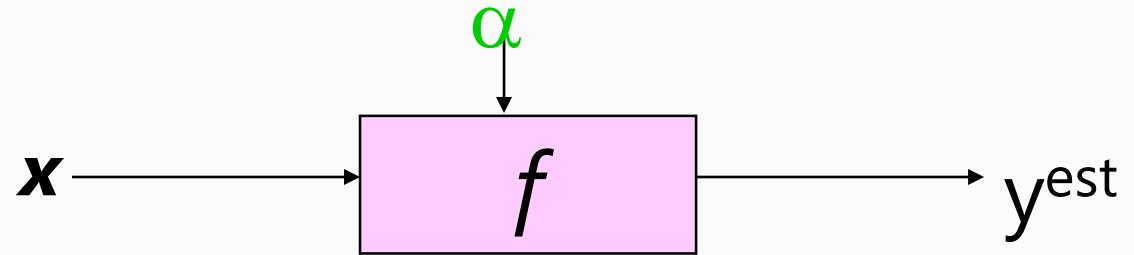
Geometric Intuition



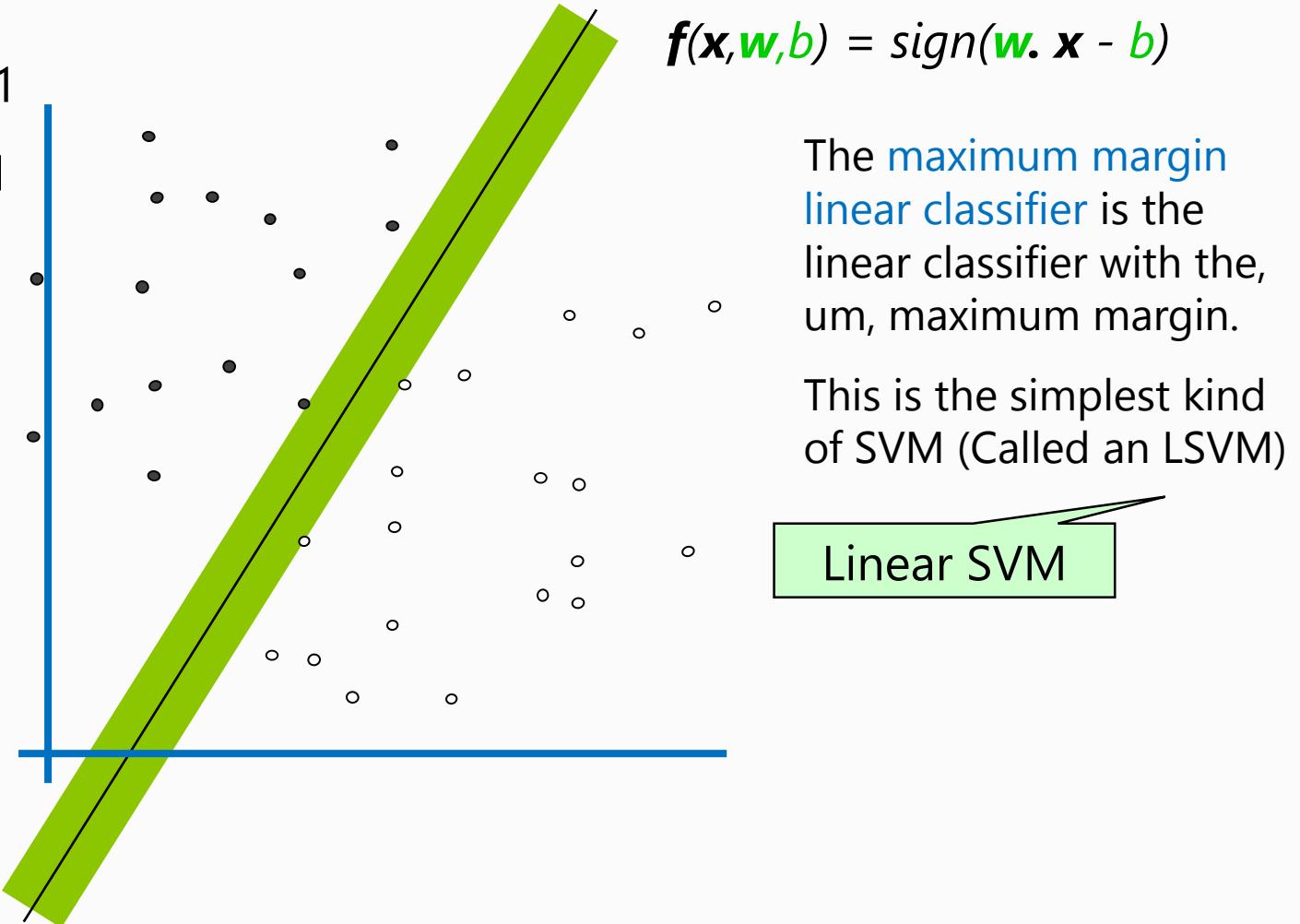
Geometric Intuition



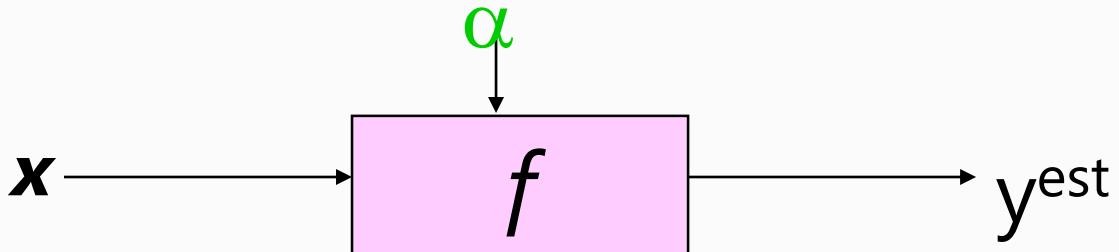
Maximum Margin



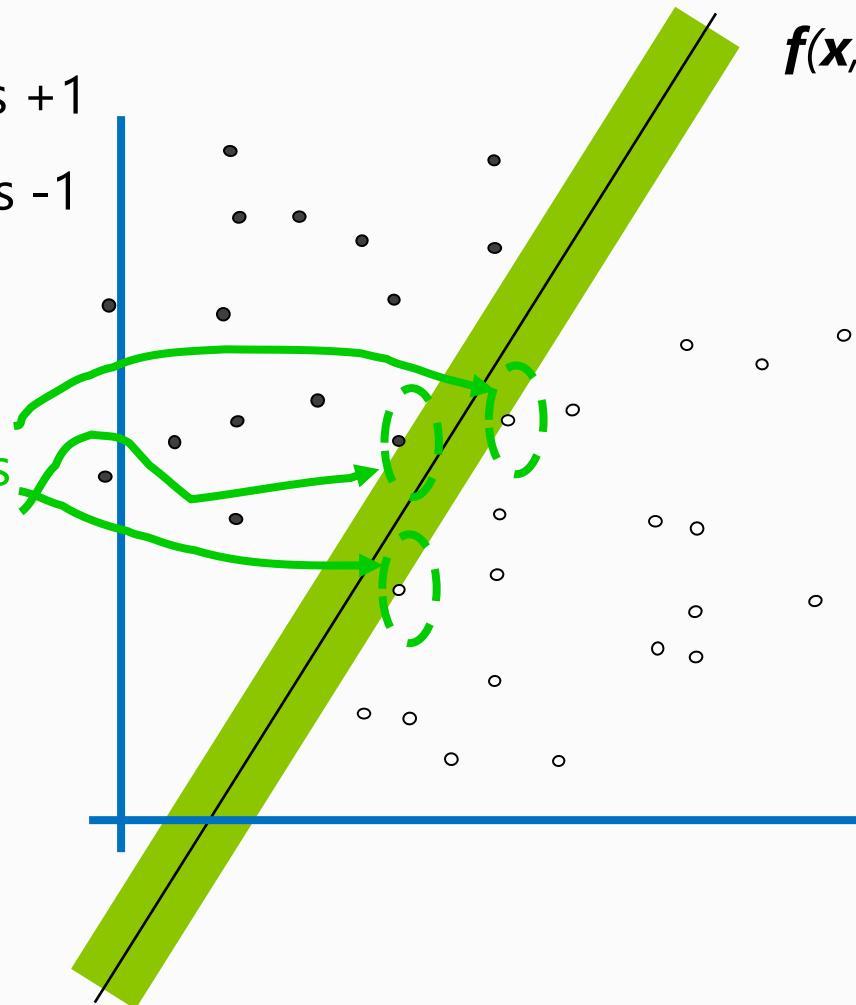
- denotes +1
- denotes -1



Maximum Margin



- denotes +1
 - denotes -1
- Support Vectors are those datapoints that the margin pushes up against



$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)$$

The maximum margin linear classifier is the linear classifier with the, um, maximum margin.

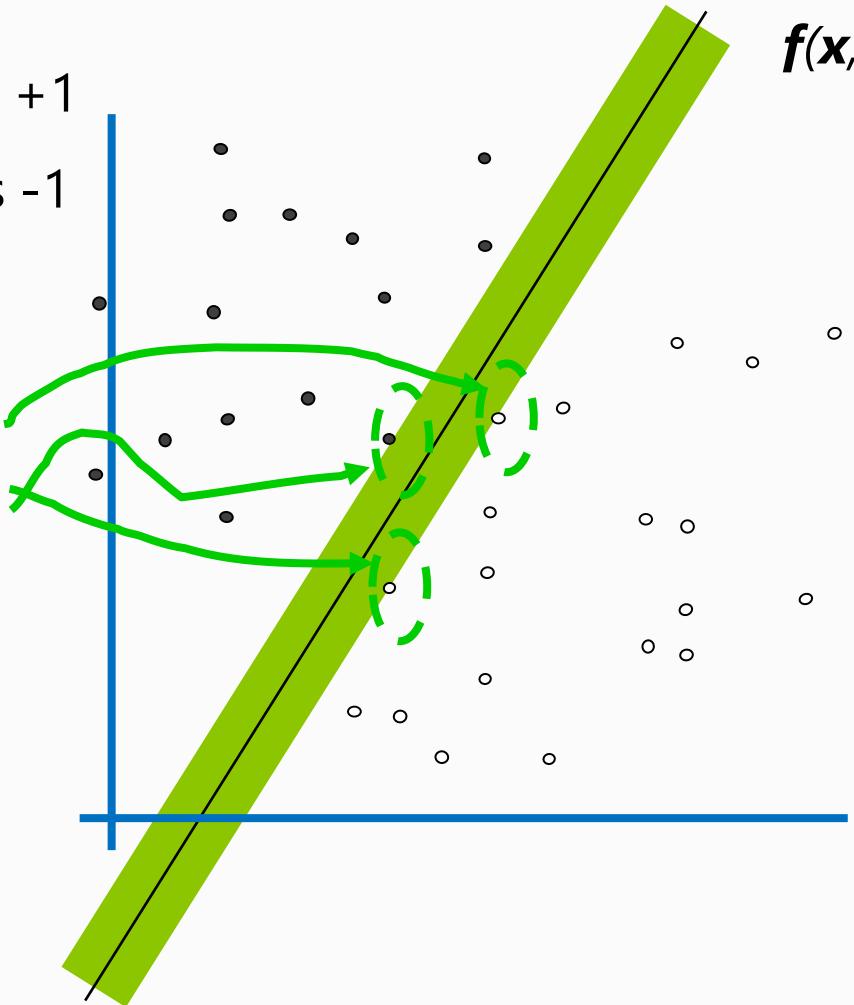
This is the simplest kind of SVM (Called an LSVM)

Linear SVM

Why Maximum Margin?

- denotes +1
- denotes -1

Support Vectors
are those
datapoints that
the margin
pushes up
against



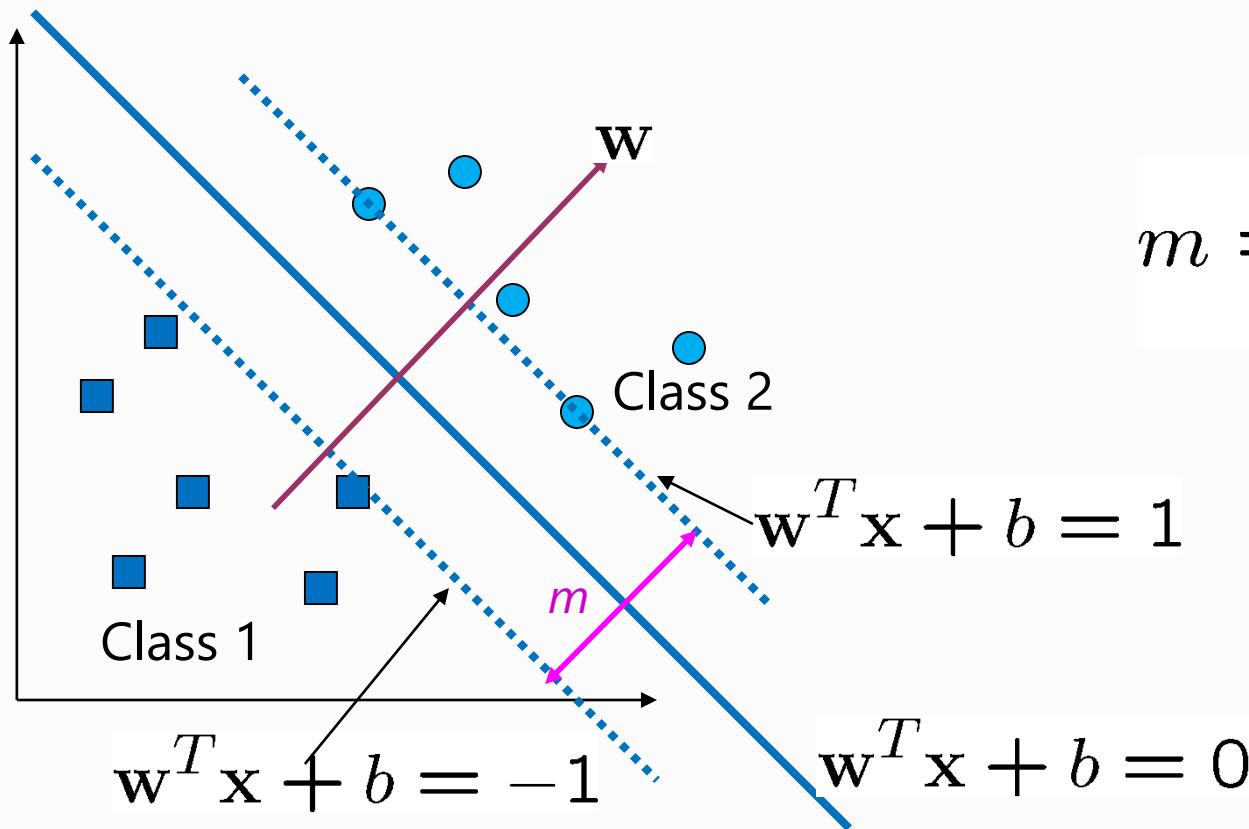
$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} - b)$$

1. Intuitively this feels safest.
2. If we've made a small error in the location of the boundary this gives us least chance of causing a misclassification.
3. LOOCV (leave one out cross validation) is easy since the model is immune to removal of any nonsupport-vector data points.
5. Empirically it works **very** well.

Large-Margin Decision Boundary

The decision boundary should be as far away from the data of both classes as possible

- We should maximize the margin, m
- Distance between the origin and the line $\mathbf{w}^T \mathbf{x} = -b$ is $b/\|\mathbf{w}\|$

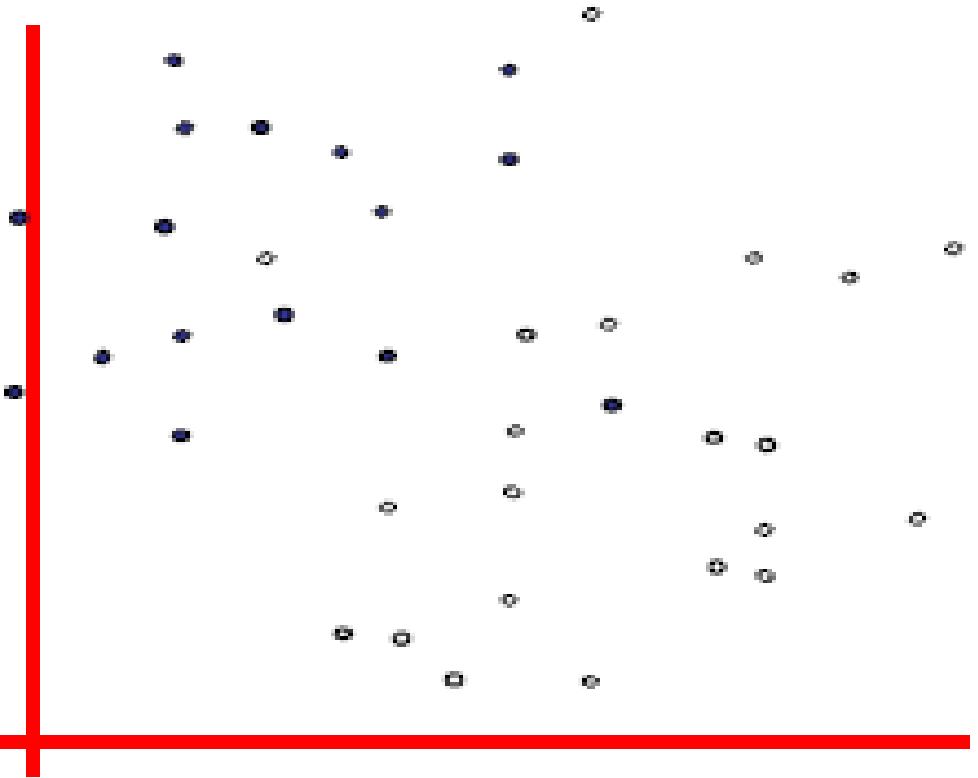


$$m = \frac{2}{\|\mathbf{w}\|}$$

Uh-oh!

This is going to be a problem!

- denotes +1
- denotes -1



Slack Variables

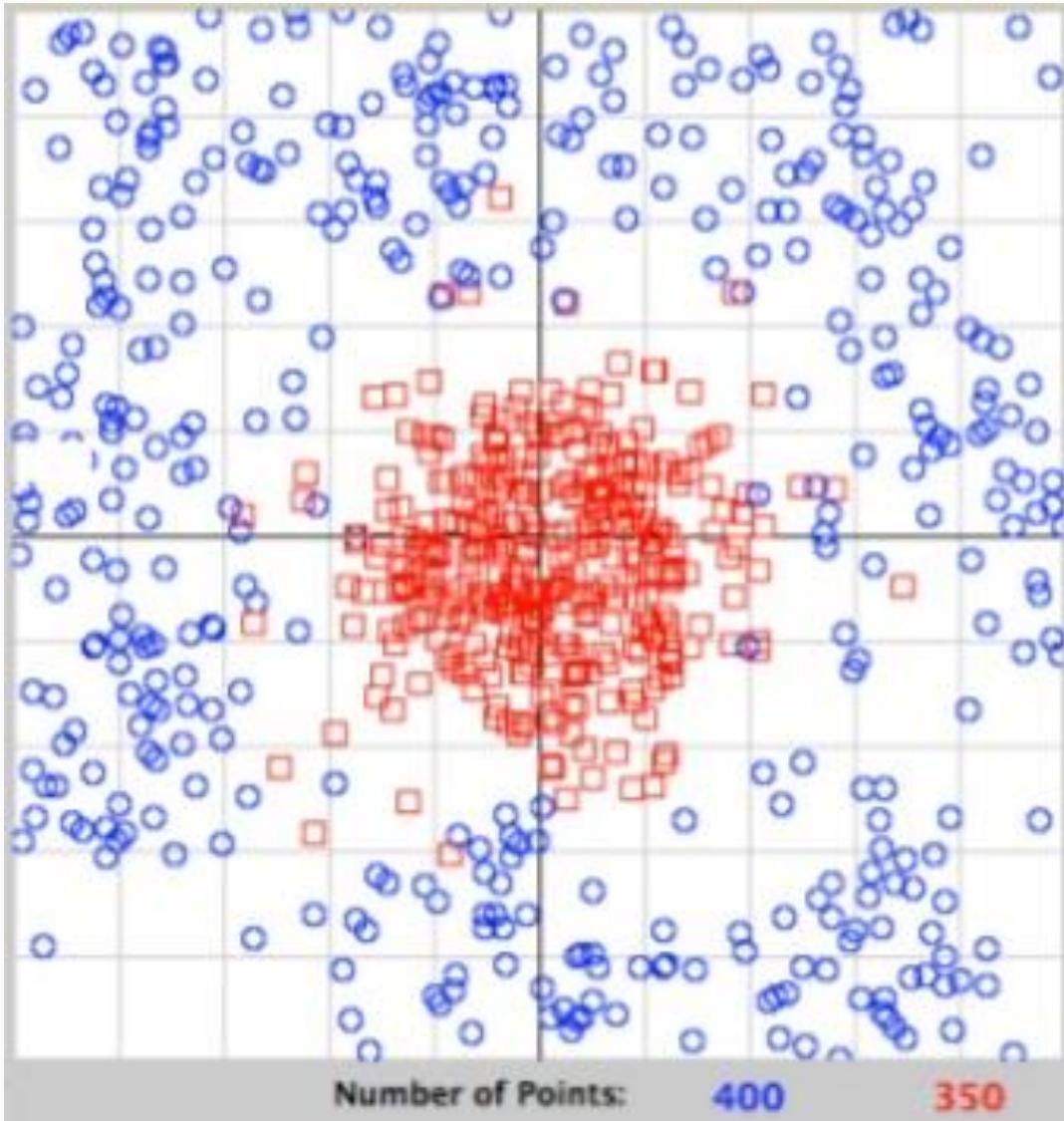
- SVMs are made robust by adding 'slack variables' that allow training error to be non-zero;
- One for each data point. Slack variable == 0 for correctly classified points

Noise

- Have assumed that the data is separable (in original or transformed space)
- Can apply SVMs to noisy data by introducing a “noise” parameter C
- C bounds the influence of any one training instance on the decision boundary
- Still a quadratic optimization problem
- Have to determine C by experimentation

The Kernel Trick

What if the data looks like that below?...

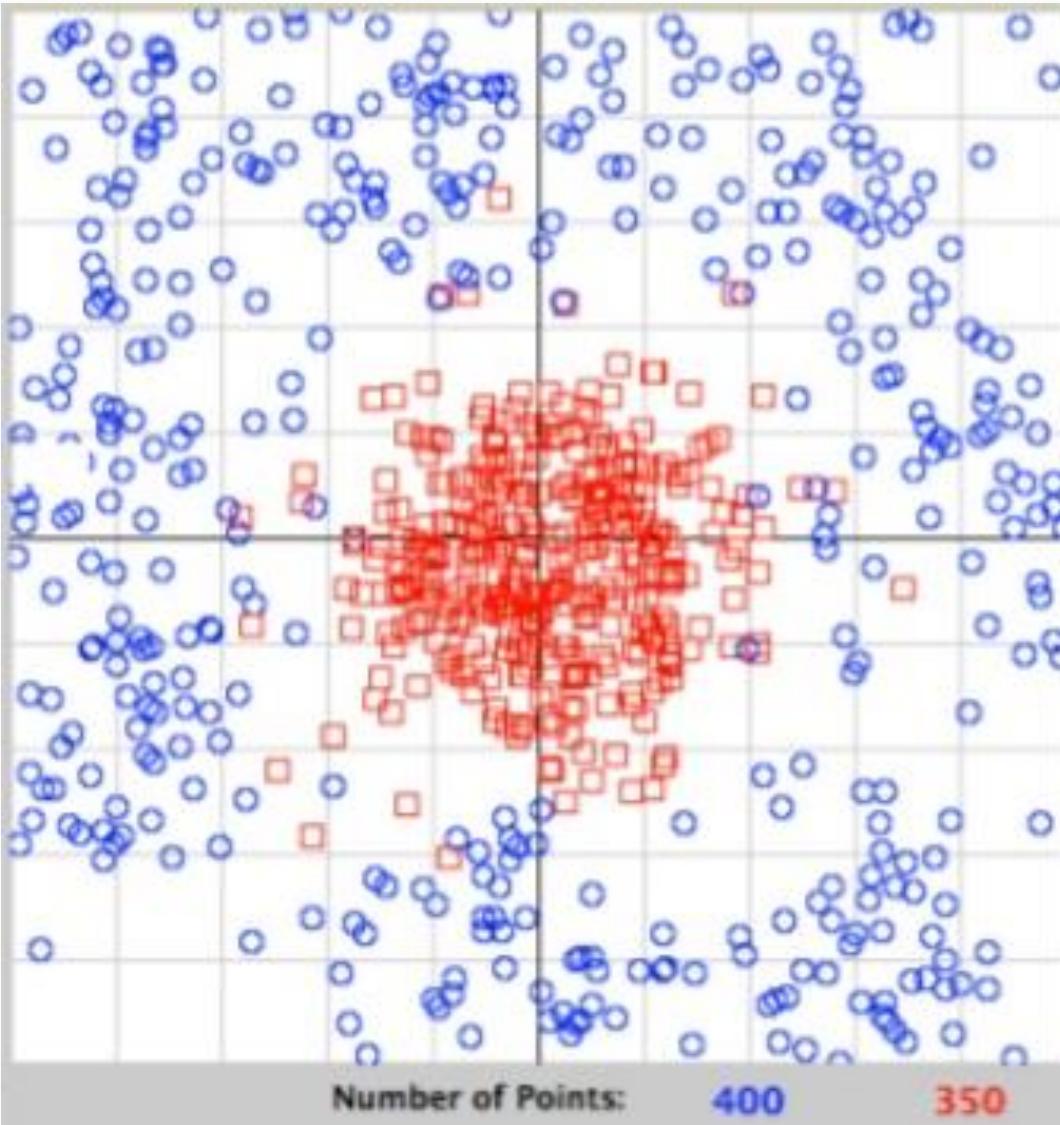


The simplest way to divide two groups is with a straight line, flat plane or an N-dimensional hyperplane. But what if the points are separated by a nonlinear region?

Rather than fitting nonlinear curves to the data, SVM handles this by using a *kernel function* to map the data into a different space where a hyperplane can be used to do the separation.

The Kernel Trick

What if the data looks like that below?...



The kernel trick allows you to use SVMs with non-linear separators

Different kernels

- Polynomial
- Gaussian
- Exponential
- ...

The Kernel Trick

Everything you need to know in one slide...

Sometimes it improves your classifier if you map (i.e. transform) your data into a different feature space.

- Example 1: you have nonlinear data but want to use a linear classifier
- Example 2: your original data aren't numbers – for example they could be sequences

Kernel methods map your original data into a different space so that you can use linear classifiers.

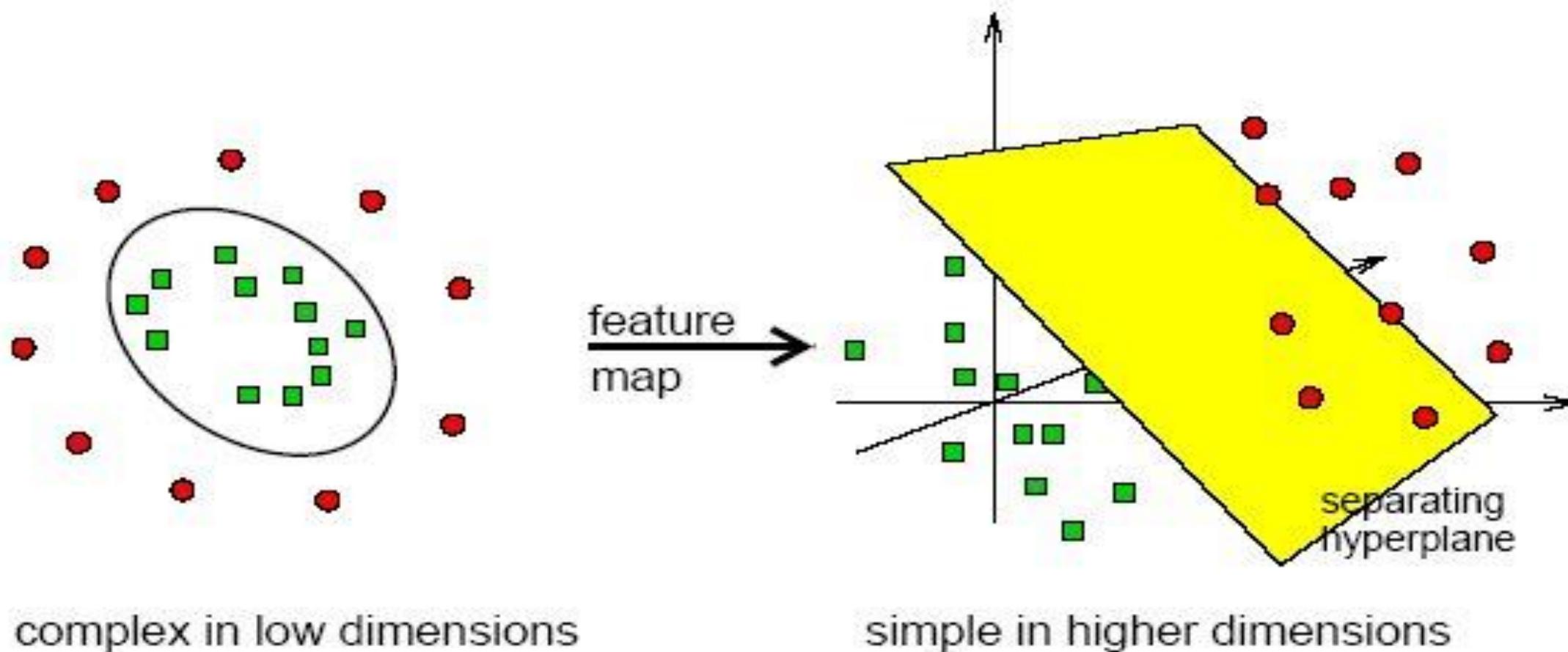
This mapping can often substantially increases the number of features to consider.

- This can be problematic as your number of dimensions grows.

The “Kernel Trick” addresses this by putting a cap on the feature explosion so that the complexity of your classifier increases only linearly with the size of your original data.

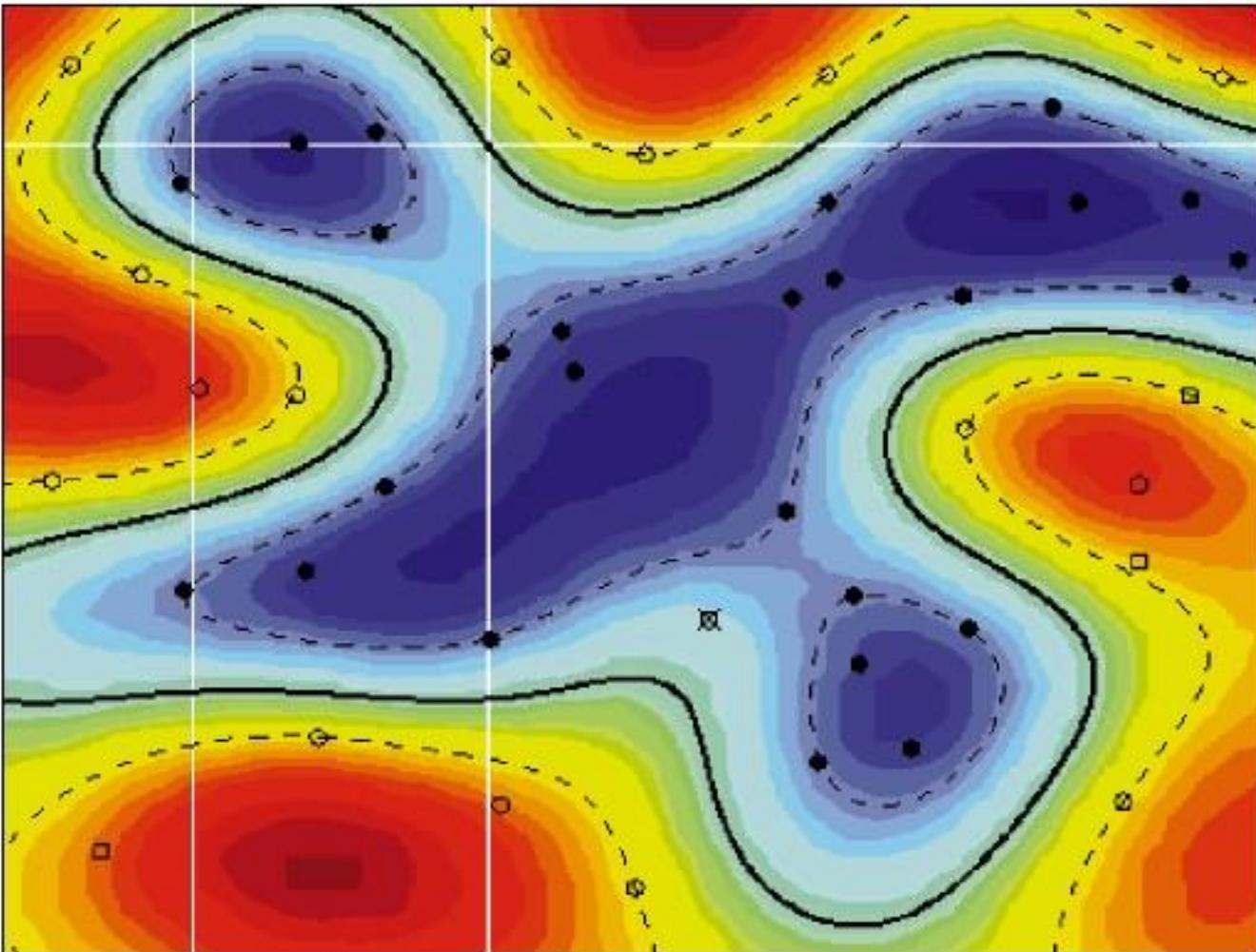
Nonlinear Support Vector Machines

Separation may be easier in higher dimensions



Nonlinear Support Vector Machines

The kernel function transforms the data into a higher dimensional space to make it possible to perform the separation.



Choosing the Kernel Function

Probably the most tricky part of using SVM

RBF is a good first option...

Depends on your data—try several.

- Kernels have even been developed for nonnumeric data like sequences, structures, and trees/graphs.

May help to use a combination of several kernels.

Don't touch your evaluation data while you're trying out different kernels and parameters.

- Use cross-validation for this if you're short on data



Type of Kernel	Inner product kernel $K(\vec{x}, \vec{x}_i), i = 1, 2, \dots, N$	Comments
Polynomial Kernel	$K(\vec{x}, \vec{x}_i) = (\vec{x}^T \vec{x}_i + \theta)^d$	Power p and threshold θ is specified a priori by the user
Gaussian Kernel	$K(\vec{x}, \vec{x}_i) = e^{-\frac{1}{2\sigma^2} \ \vec{x} - \vec{x}_i\ ^2}$	Width σ^2 is specified a priori by the user
Sigmoid Kernel	$K(\vec{x}, \vec{x}_i) = \tanh(\eta \vec{x} \vec{x}_i + \theta)$	Mercer's Theorem is satisfied only for some values of η and θ
Kernels for Sets	$K(\chi, \chi') = \sum_{i=1}^{N_\chi} \sum_{j=1}^{N_{\chi'}} k(x_i, x'_j)$	Where $k(x_i, x'_j)$ is a kernel on elements in the sets χ, χ'
Spectrum Kernel for strings	count number of substrings in common	It is a kernel, since it is a dot product between vectors of indicators of all the substrings.

Table 1: Summary of Inner-Product Kernels [Hay98]

Complexity of the optimization problem remains only dependent on the dimensionality of the input space and not of the feature space!

Sparse Data

- SVM algorithms speed up dramatically if the data is sparse (i.e. many values are 0)
- Why? Because they compute lots and lots of dot products
- Sparse data compute dot products very efficiently
- Iterate only over nonzero values
- SVMs can process sparse datasets with 10,000s of attributes

Doing multi-class classification

- SVMs can only handle two-class outputs (i.e. a categorical output variable with arity 2).
- What can be done?
- Answer: with output arity N, learn N SVM's
 - SVM 1 learns "Output==1" vs "Output != 1"
 - SVM 2 learns "Output==2" vs "Output != 2"
 -
 - SVM N learns "Output==N" vs "Output != N"
- Then to predict the output for a new input, just predict with each SVM and find out which one puts the prediction the furthest into the positive region.

What you need to know...

- First, try logistic regression. Easy, fast, stable. No ‘tuning’ parameters.
- Equivalently, you can first try linear SVMs, but you need to tune ‘C’
- If results are “good enough”, stop
- Else, try SVMs with Gaussian kernels (RBF)

Need to tune bandwidth, C – by using validation data...

Summary: Steps for Classification

- SVMs require vector of real numbers
 - Categorical variables → numeric data {R,G,B} → {0,0,1},...{1,0,0}
 - Scaling to the range [-1, +1] or [0,1]
- Select the kernel function to use
 - RBF is a reasonable first choice, two parameters C and γ
 - Grid search to identify best values for parameters
 - Use v-fold cross validation to ensure good performance on test data
 - Unseen data can be classified using support vectors

Conclusion

- SVMs balance between correctness and generalization
 - Decision boundaries
 - Margins
 - Support vector
- Two key concepts of SVM: maximize the margin and the kernel trick
- Many SVM implementations are available on the web for you to try on your data set!

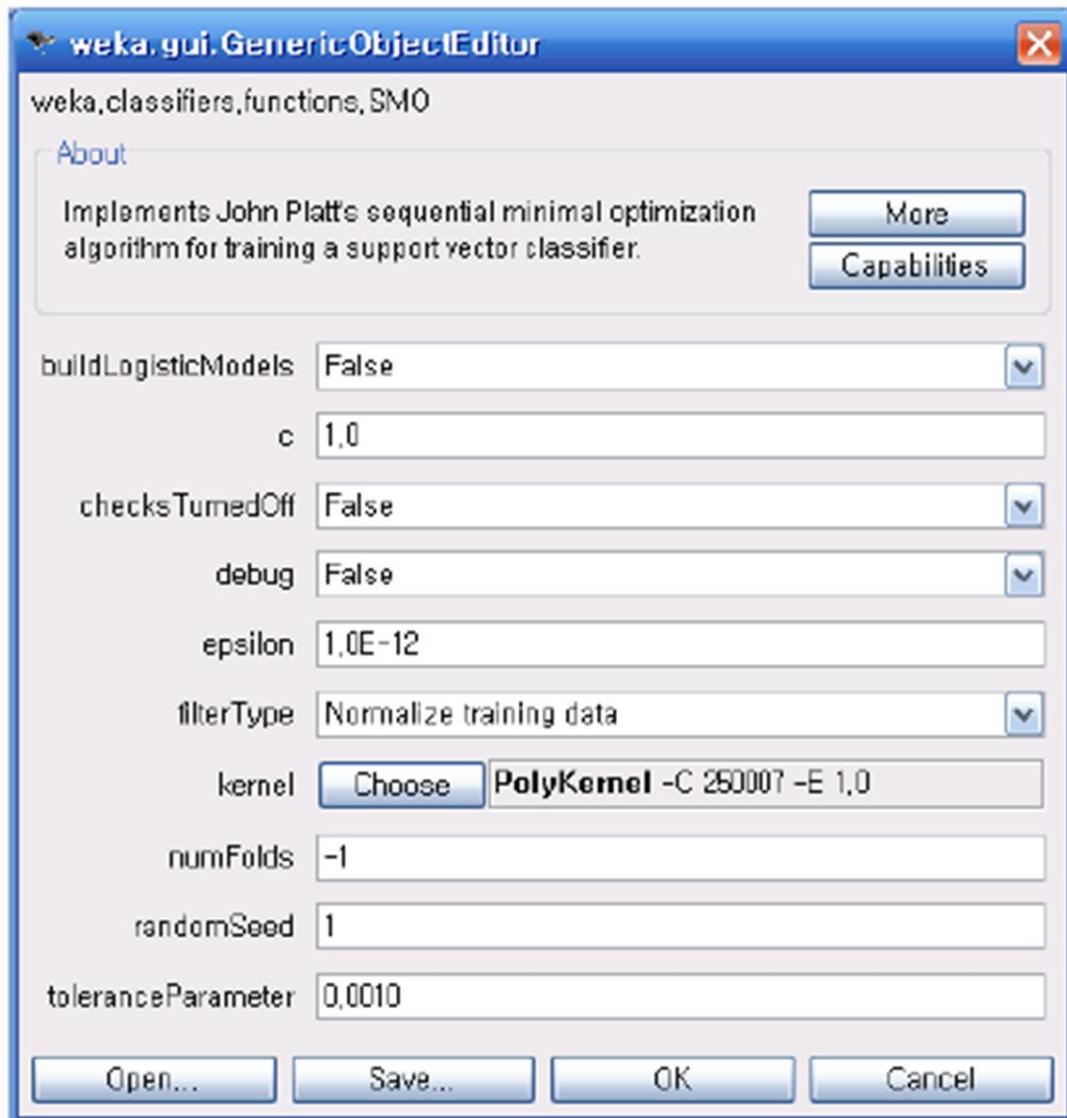
Software

- A list of SVM implementation can be found at
<http://www.kernel-machines.org/software.html>
- Some implementation (such as LIBSVM) can handle multi-class classification
- SVMLight is among one of the earliest implementation of SVM
- Several Matlab toolboxes for SVM are also available

In WEKA ...

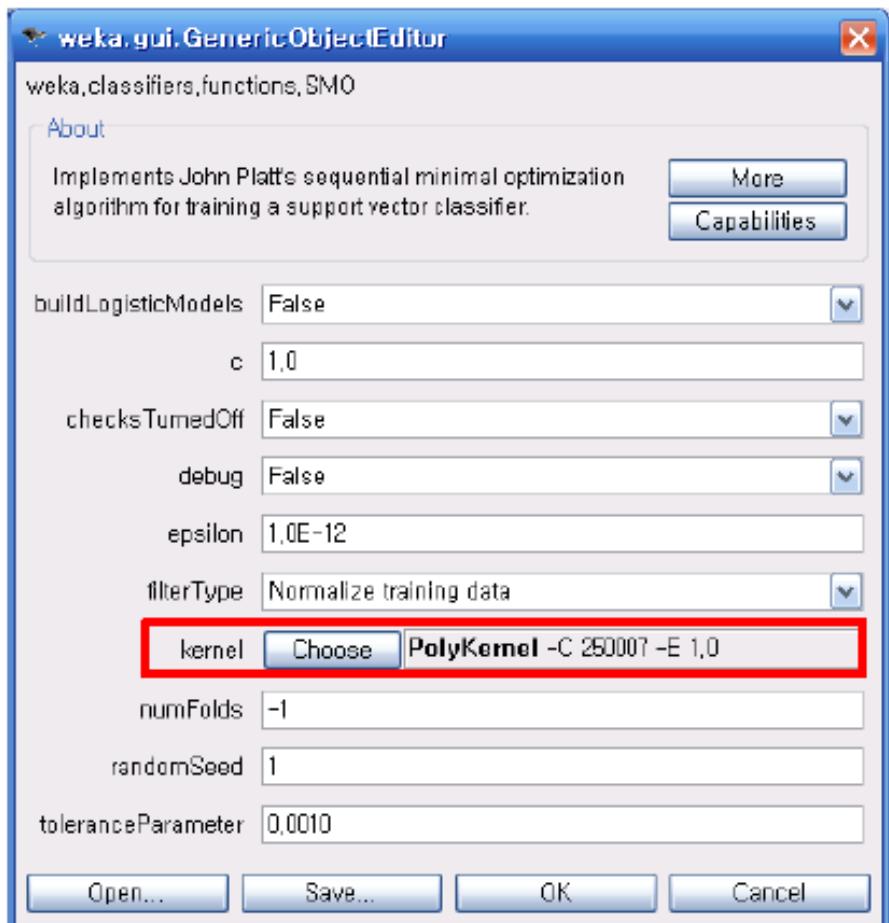
- Sequential minimal optimization (SMO) algorithm support vector *classification*
 - *weka.classifiers.functions.SMO*
- A Library for Support Vector Machines (libSVM)
 - *weka.classifiers.functions.libSVM*
- Sequential minimal optimization (SMO) algorithm support vector *regression*
 - *weka.classifiers.functions.SMOreg*





- ❖ **buildLogisticModels**: Whether to fit logistic models to the outputs (for proper probability estimates)
- ❖ **numFolds**: The number of folds for cross-validation used to generate training data for logistic models
- ❖ **randomSeed**: Random number seed for the cross-validation
- ❖ **c** -- The complexity parameter C. It is the upper bound of alpha's
- ❖ **filterType** -- Determines how/if the data will be transformed.
- ❖ **kernel** -- The kernel to use.

Parameter Setting Guide



❖ Suggested

- buildLogisticModels: True
- numFolds: 3 or 5
- randomSeed: any value
- C (complexity parameter, upper bound of alpha)

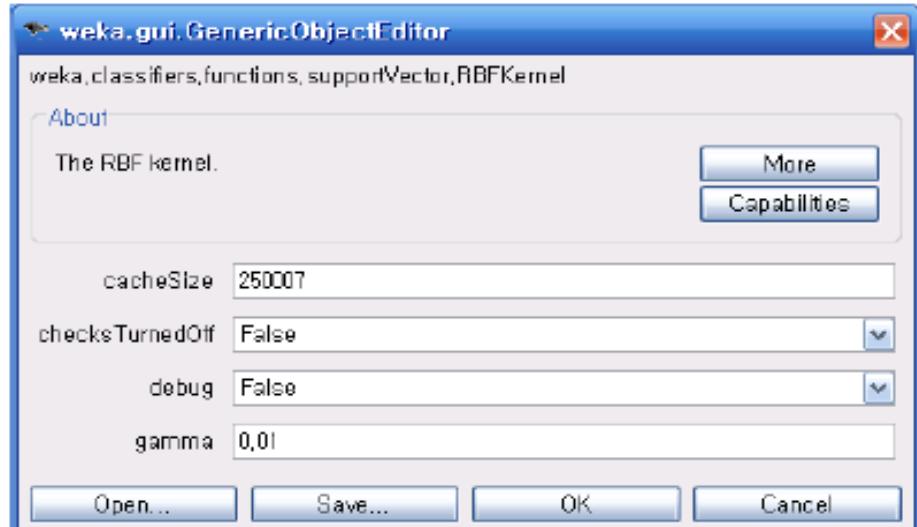
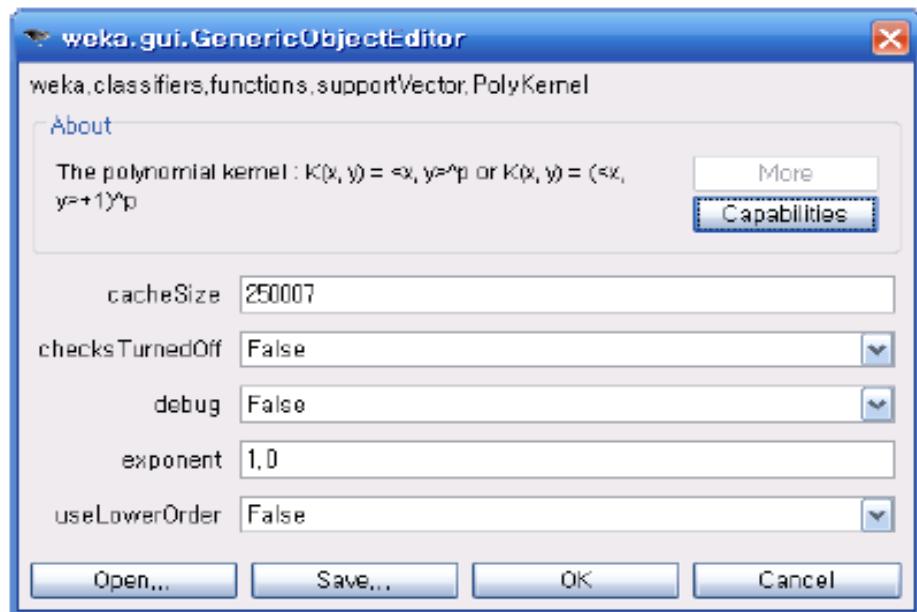
❖ Your own choice

- filterType
- Kernel and its subsequent parameters
- Debug – if on, you can see intermediate results

❖ Do not change!

- epsilon
- checksTurnedOff
- toleranceParameter

Parameter Setting Guide – Kernel



❖ PolyKernel

- Try various exponents
 - Floating points are allowed
- 1.0: linear kernel

❖ RBFKernel

- Try various gammas
- Gamma value corresponds to the inverse of the variance (width of the kernel)

Data Science

Deriving Knowledge from Data at Scale

That's all for tonight....

Roger S. Barga

Aug 7th, 2014

Deriving Knowledge from Data at Scale

