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…*

Nov 9th , 2015

Review…

Lecture 5 Homework 3a, *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 3a, *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)

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

Final cluster centroids:

Attribute	Full Data (100)	Cluster# 0 (31)	1 (44)	2 (9)	3 (16)
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

- Clusters 2 & 3 – *Searchers*

- Cluster 0 – Products → Prod_C
- Recommend Prod_B

- Cluster 1 – *Focused Browsers*

- Cluster 2 – *Window Shoppers*

Nearest Neighbor Classification

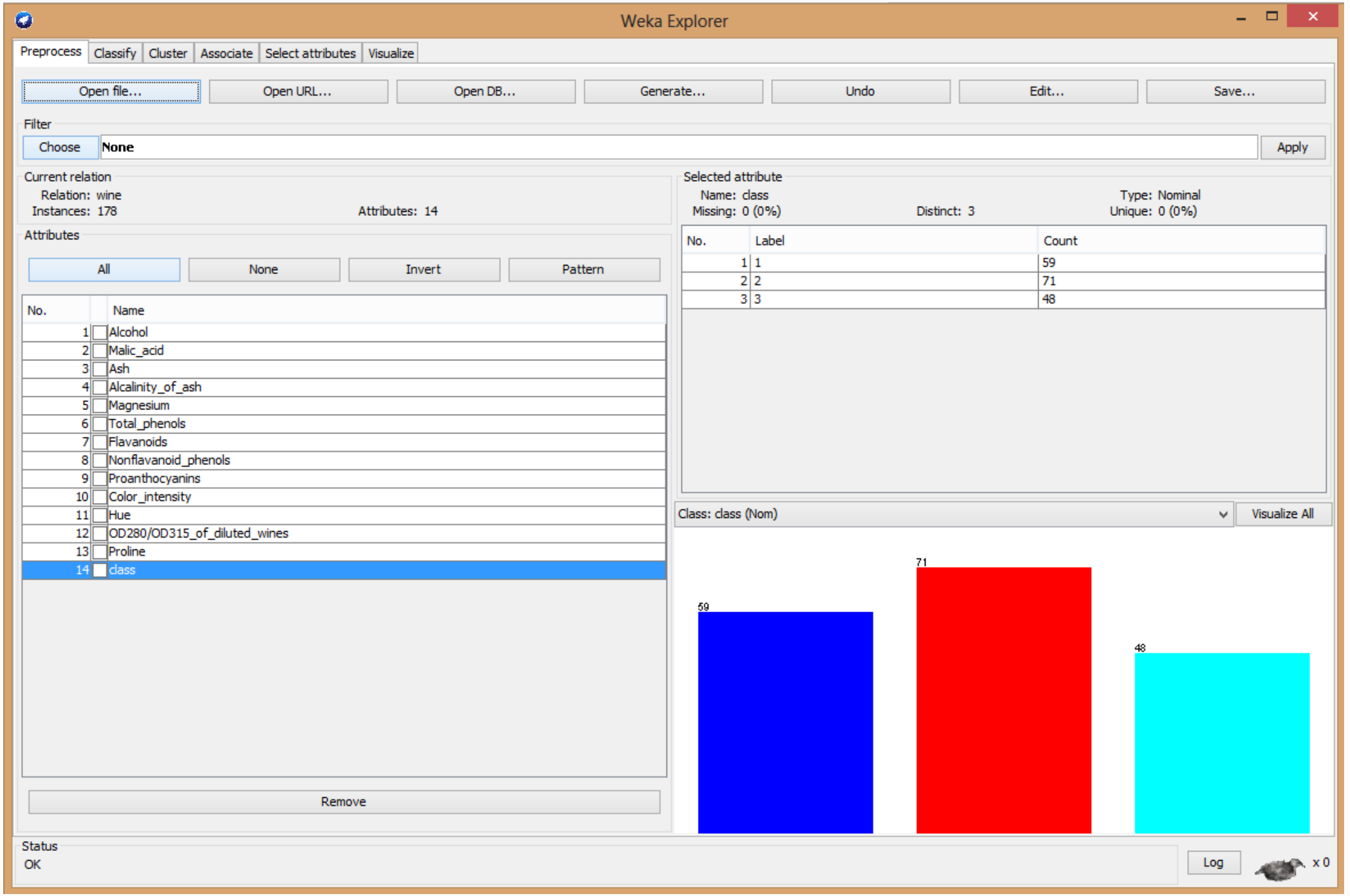
Amazon and "Customers who purchased X also purchased Y" feature.

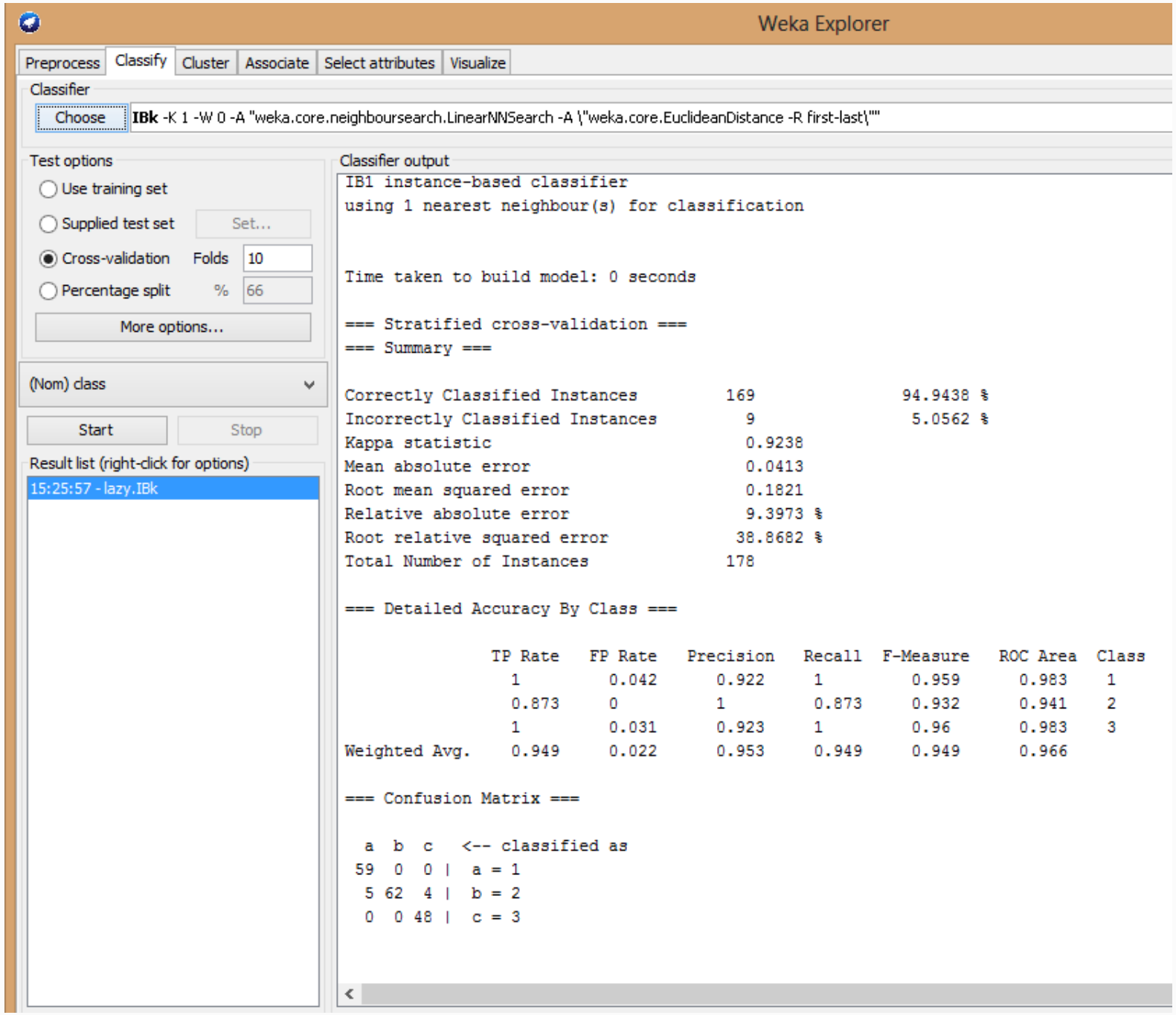
As scalable for a 20-customer database as it is for a 20 million-customer database, and you can define the number of results you want to find. A great technique…

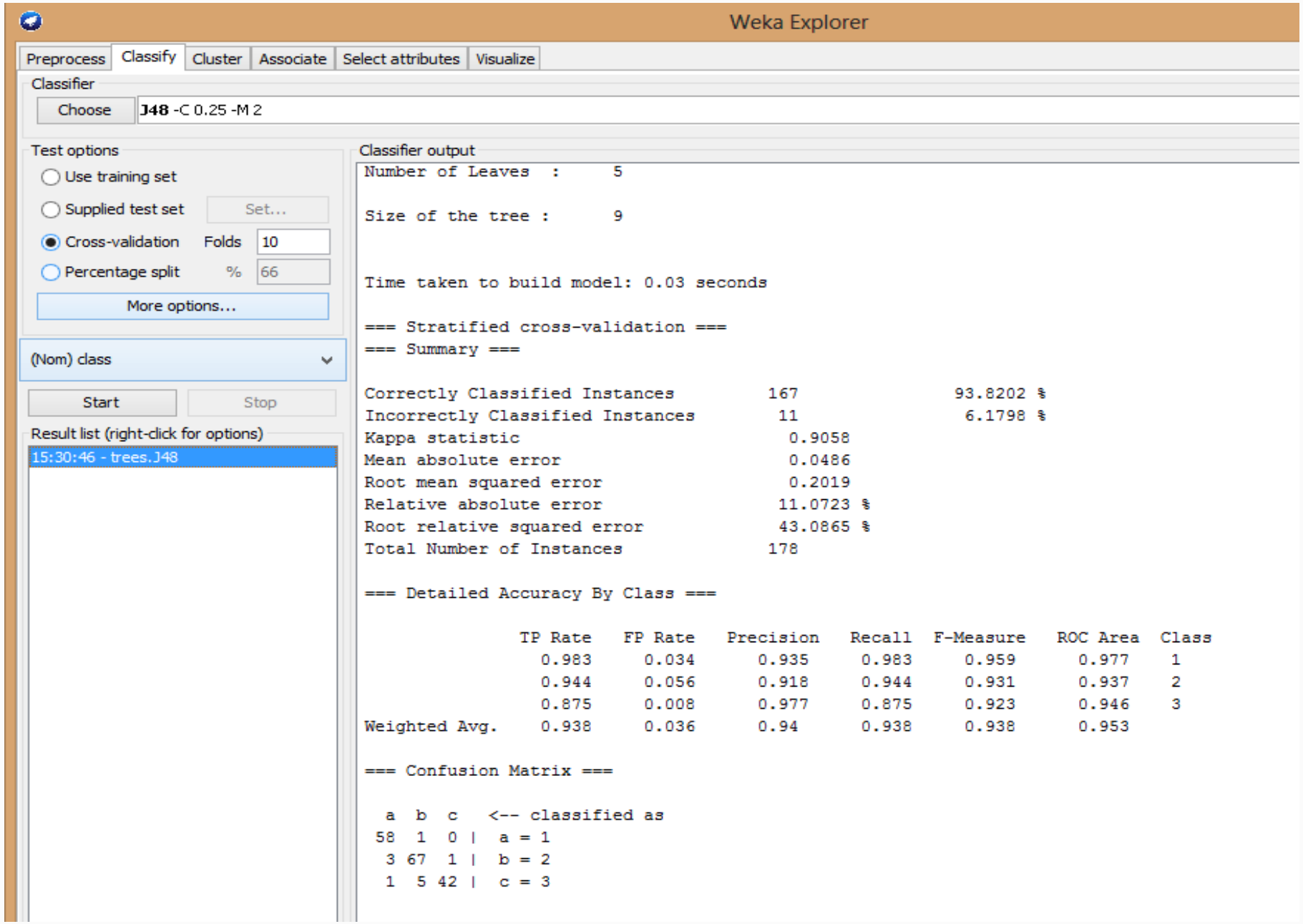
```
Customer      Age      Income      Purchased Product
1             45        46k        Book
2             39        100k       TV
3             35        38k        DVD
4             69        150k       Car Cover
5             58        51k        ???

Step 1:  Determine Distance Formula
Distance = SQRT( ((58 - Age)/(69-35))^2) + ((51000 - Income)/(150000-38000))^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 (dontated 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
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
59
71
48
Status
OK
Log
x 0


Weka Explorer
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 1
0.873 0 1 0.873 0.932 0.941 2
1 0.031 0.923 1 0.96 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



Sometimes simpler is better…

Principal Component Analysis

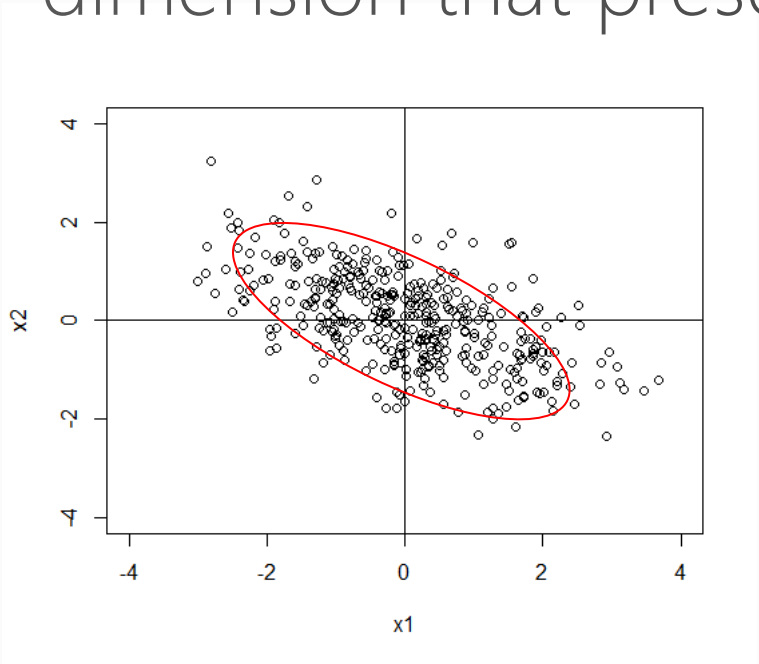
- Designed to find underlying features in high dimensional data
- Dimension reduction technique
- Pattern discovery tool

- Requirements:
  - Data is continuous in nature
  - Missing values could impact this method

- Remark:
  - Correlation instead of covariance
  - No guarantee to find pattern(s)

Principal Component Analysis

- Structured data:
- Observations (n rows) and Attributes/Features (p columns)
- Among these p features, some of them are correlated. Perhaps k of them could represent the original p.
- The basic idea is to project the original data to a lower dimension that preserves majority of the variation.

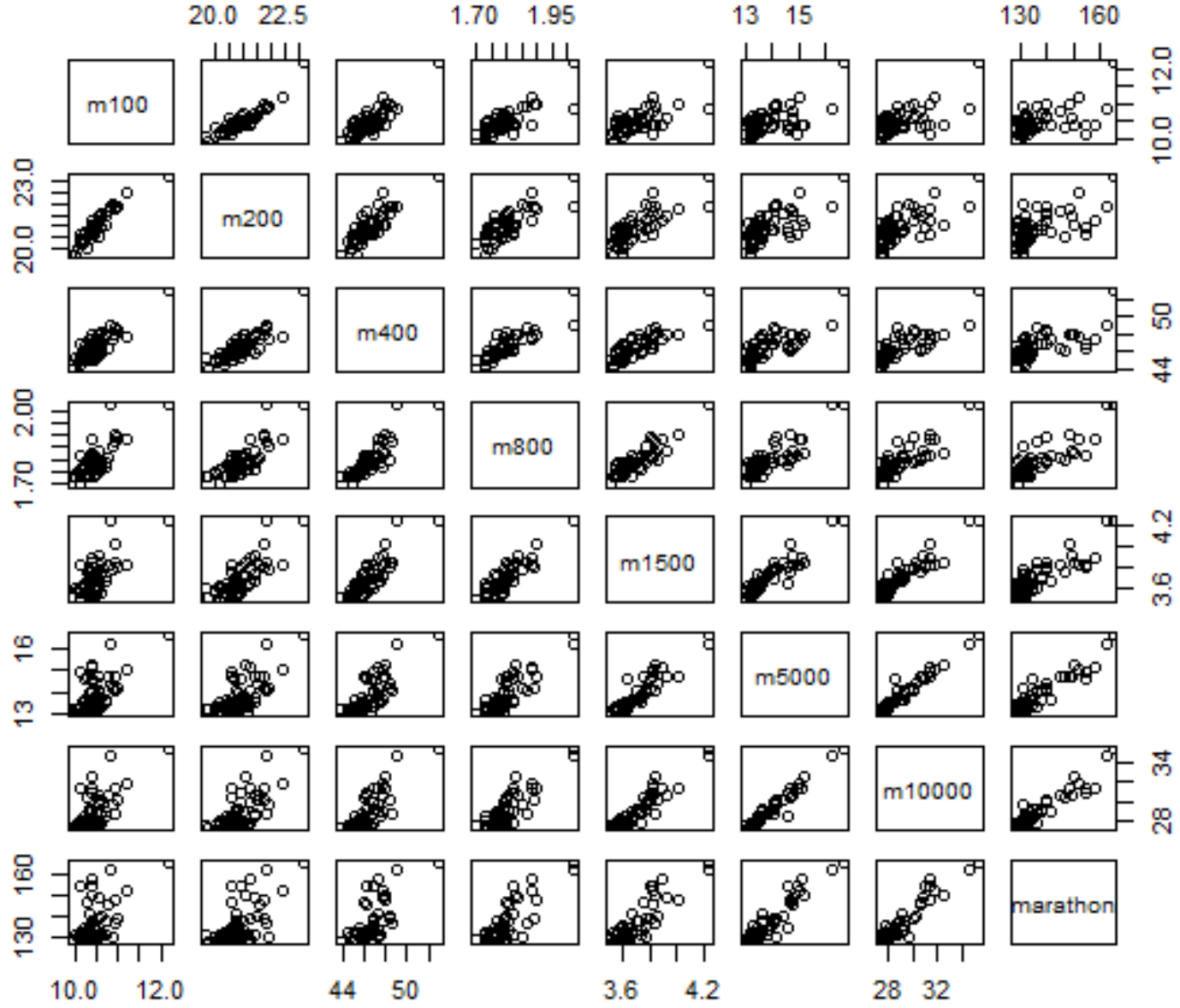


PCA example

- Men Track & Field Records 1984

- 8 different events
- 55 countries
- M100 to M400 seconds
- Rest in minutes

- Correlations among events



Ref: Johnson & Wichern

PCA example

- R: “princomp()”

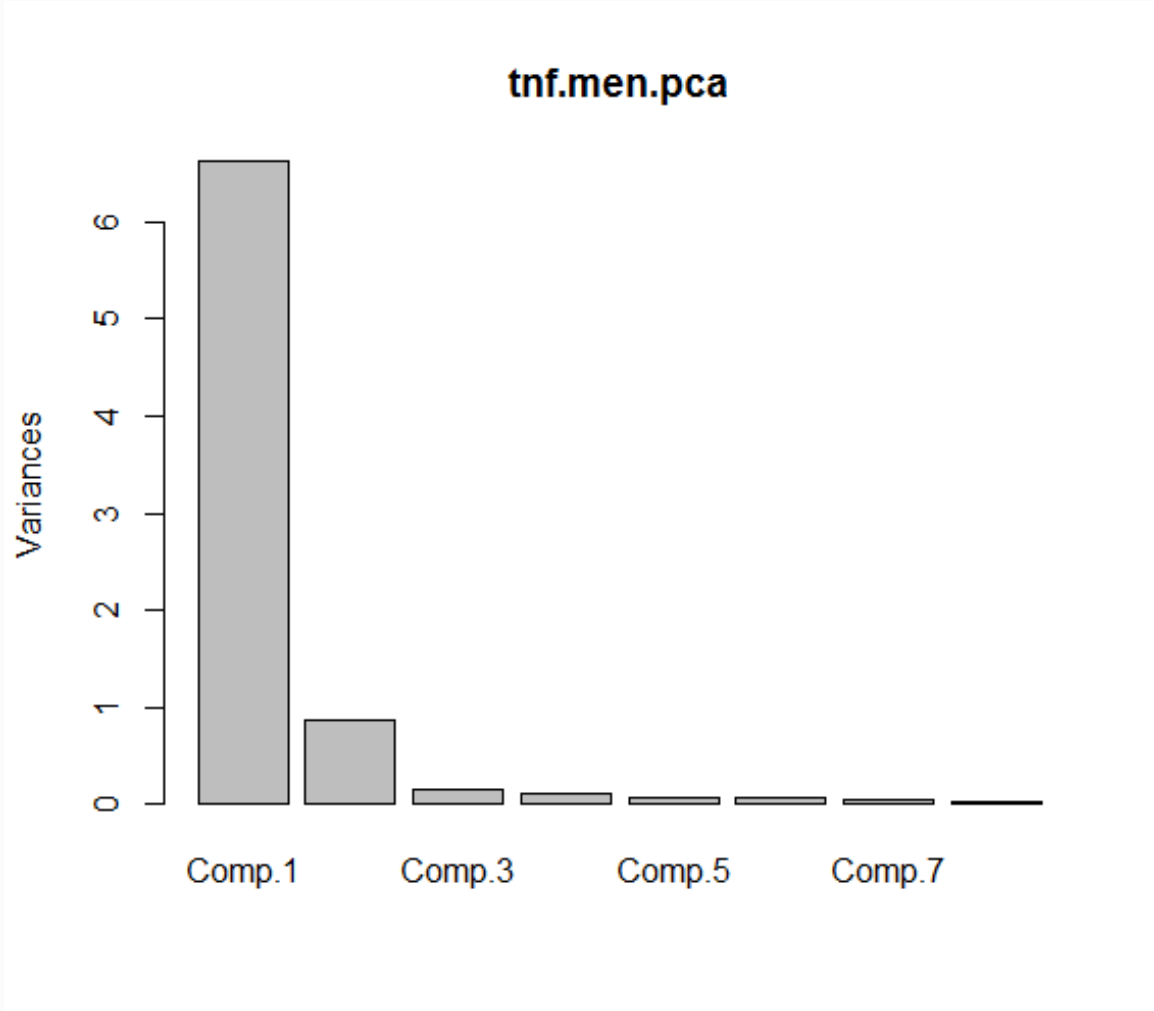
```
tnf.men <- read.csv( 'c:/Users/taams/Documents/Fall 2015 Course/References/men_TnF_1984.csv',
                header=T, sep=',' )

## look at the data in pairwise scatter plot
pairs( tnf.men[,c(1:8) ] )

##
tnf.men.pca <- princomp( tnf.men[,c(1:8)], cor=T )

##
tnf.men.pca$loadings
```

```
> tnf.men.pca$loadings

Loadings:
         Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
m100     -0.318 -0.567  0.332 -0.128  0.263 -0.594  0.136 -0.106
m200     -0.337 -0.462  0.361  0.259 -0.154  0.656 -0.113
m400     -0.356 -0.248 -0.560 -0.652 -0.218  0.157
m800     -0.369        -0.532  0.480  0.540        -0.238
m1500    -0.373  0.140 -0.153  0.405 -0.488 -0.158  0.610 -0.139
m5000    -0.364  0.312  0.190        -0.254 -0.141 -0.591 -0.547
m10000   -0.367  0.307  0.182        -0.133 -0.219 -0.177  0.797
marathon -0.342  0.439  0.263 -0.300  0.498  0.315  0.399 -0.158

               Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000
Proportion Var  0.125  0.125  0.125  0.125  0.125  0.125  0.125  0.125
Cumulative Var  0.125  0.250  0.375  0.500  0.625  0.750  0.875  1.000
```

PCA example

- R: screeplot()
  - Display the amount of variance each component accounted for
  - If the first few bars dominate the chart, these few components capture majority of the variance

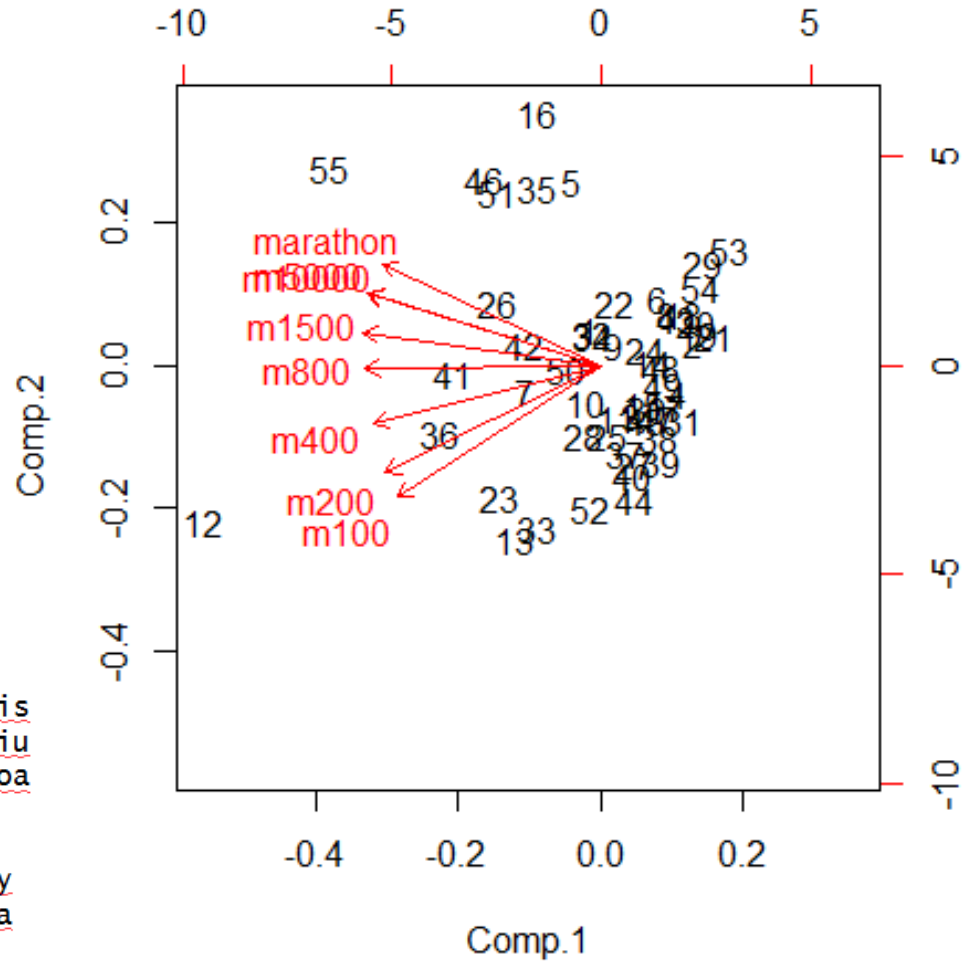
screeplot(tnf.men.pca)



PCA example

- R: biplot()
  - A display to show the first 2 components in relation with the original data and the attributes
  - Sometimes it reveals useful information but not always, depending on the amount of variance captured by the first 2 components

biplot(tnf.men.pca)




```
> tnf.men[c(12,36,55), ]
m100  m200  m400 m800 m1500 m5000 m10000 marathon  country
12 12.18 23.20 52.94 2.02  4.24 16.70  35.38   164.70   cookis
36 11.19 22.45 47.70 1.88  3.83 15.06  31.77   152.23 mauritiu
55 10.82 21.86 49.00 2.02  4.24 16.28  34.71   161.83   wsamoa
> tnf.men[c(29,53), ]
m100  m200  m400 m800 m1500 m5000 m10000 marathon country
29 10.01 19.72 45.26 1.73  3.60 13.23  27.52   131.08   italy
53  9.93 19.75 43.86 1.73  3.53 13.20  27.43   128.22     usa
```

PCA math

$$Z_{n,p} = X_{n,p} Q_{p,p}$$

Linear combination of the original data

original data

Coefficients for the combination

$Q_{p,p}$

Eigenvector of the eigenvalue decomposition of the correlation or covariance matrix of X, sorted according to the magnitude of the eigenvalues

(Principal component loadings)

$Z_{n,p}$

Principal components sorted according to the magnitude of the eigenvalues. The first PC associates with the largest eigenvalue, second PC with the second largest, and so on.

(Principal component scores)

Dimension reduction:

- Take the top k principal components from Z to represent the original X.
- Set p-k columns in Q to zero
- Resulting Z approximates the original X

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

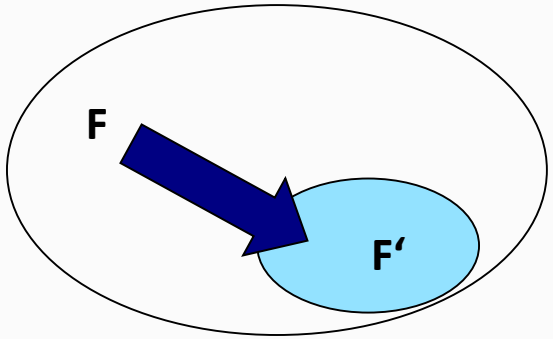
1. Missing values
2. Outliers
3. Coding

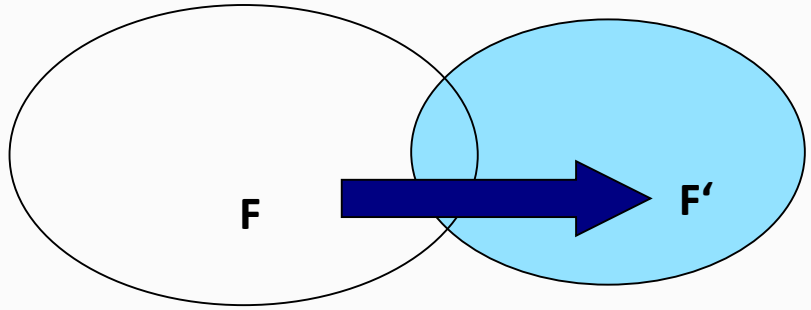
Feature Selection

Simple Definition

Given a set of features $F=\{f_1,\ldots,f_i,\ldots,f_n\}$ 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 → F'…

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



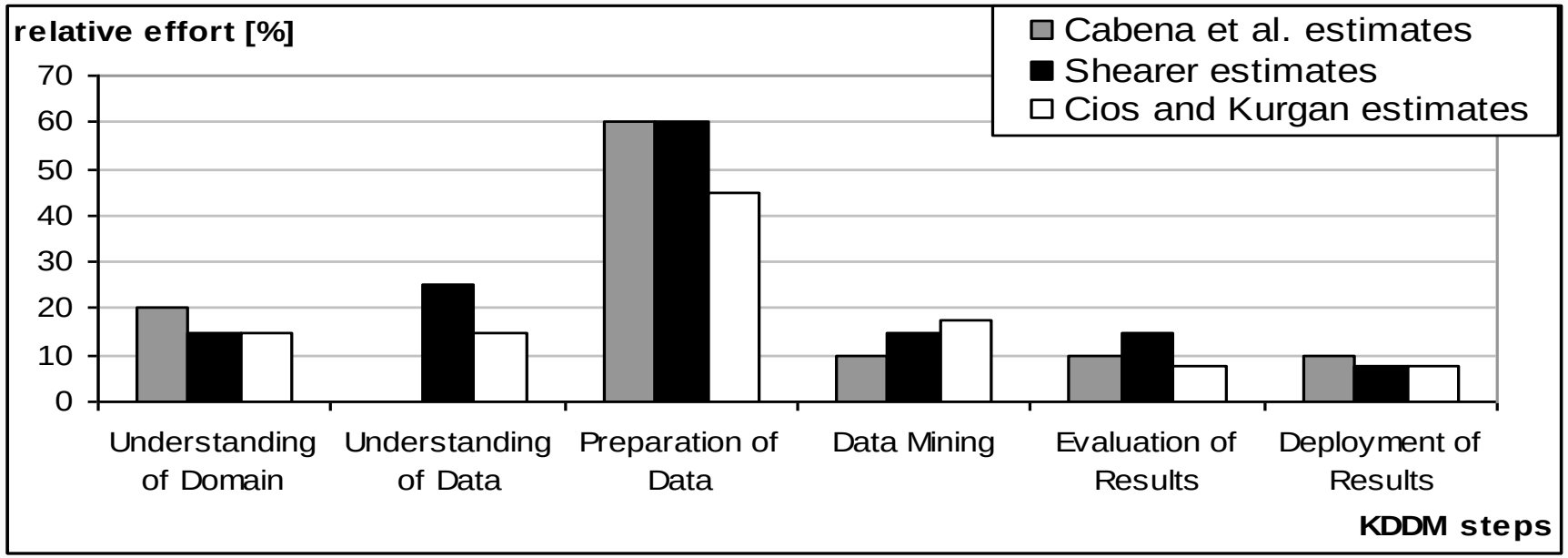


$$\{f_1,\ldots,f_i,\ldots,f_n\}\xrightarrow{f.extraction}\{g_1(f_1,\ldots,f_n),\ldots,g_j(f_1,\ldots,f_n),\ldots,g_m(f_1,\ldots,f_n)\}$$

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.

1. 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 2009[1] and KDD 2009[2]) 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 ISABELLE@CLOPINET.COM
Clopinet
955 Creston Road
Berkeley, CA 94708-1501, USA

André Elisseeff ANDRE@TUEBINGEN.MPG.DE
Empirical Inference for Machine Learning and Perception Department
Max Planck Institute for Biological Cybernetics
Spemannstrasse 38
72076 Tübingen, Germany

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

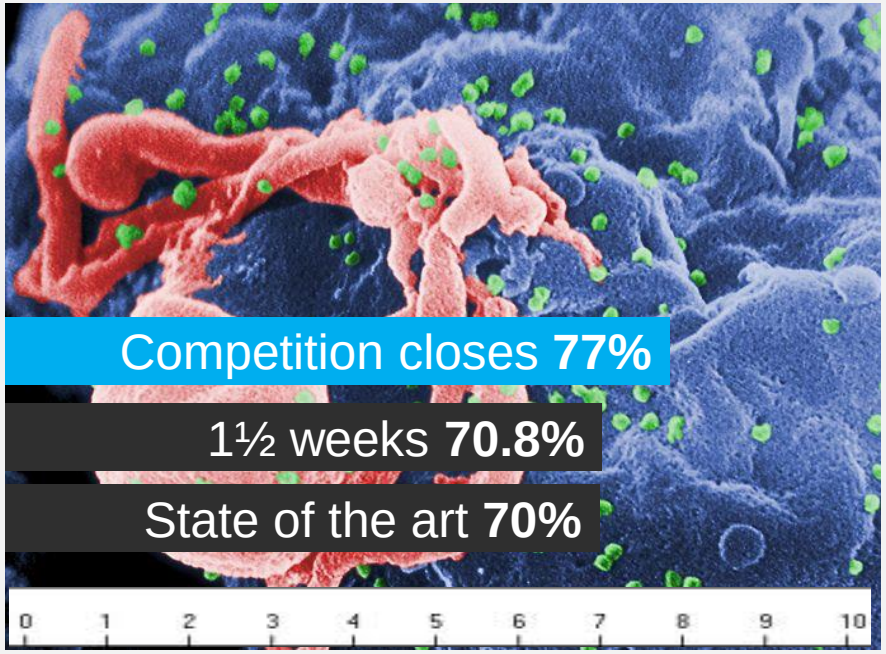
Kaggle

kaggle

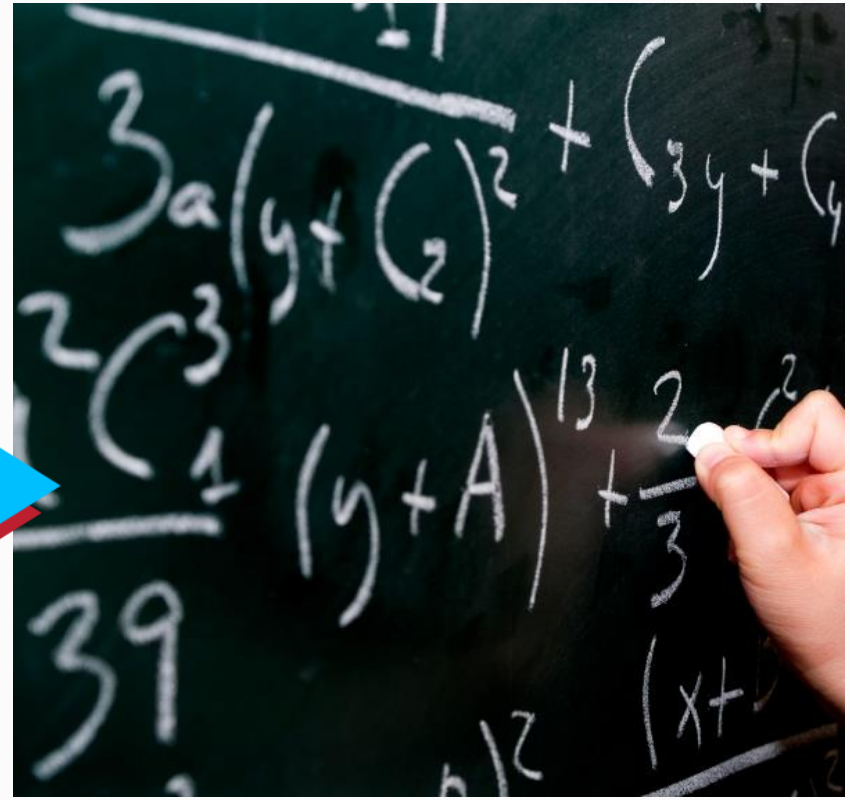
Predictive modeling competitions

Global competitions

Predicting HIV viral load Improved by 10%

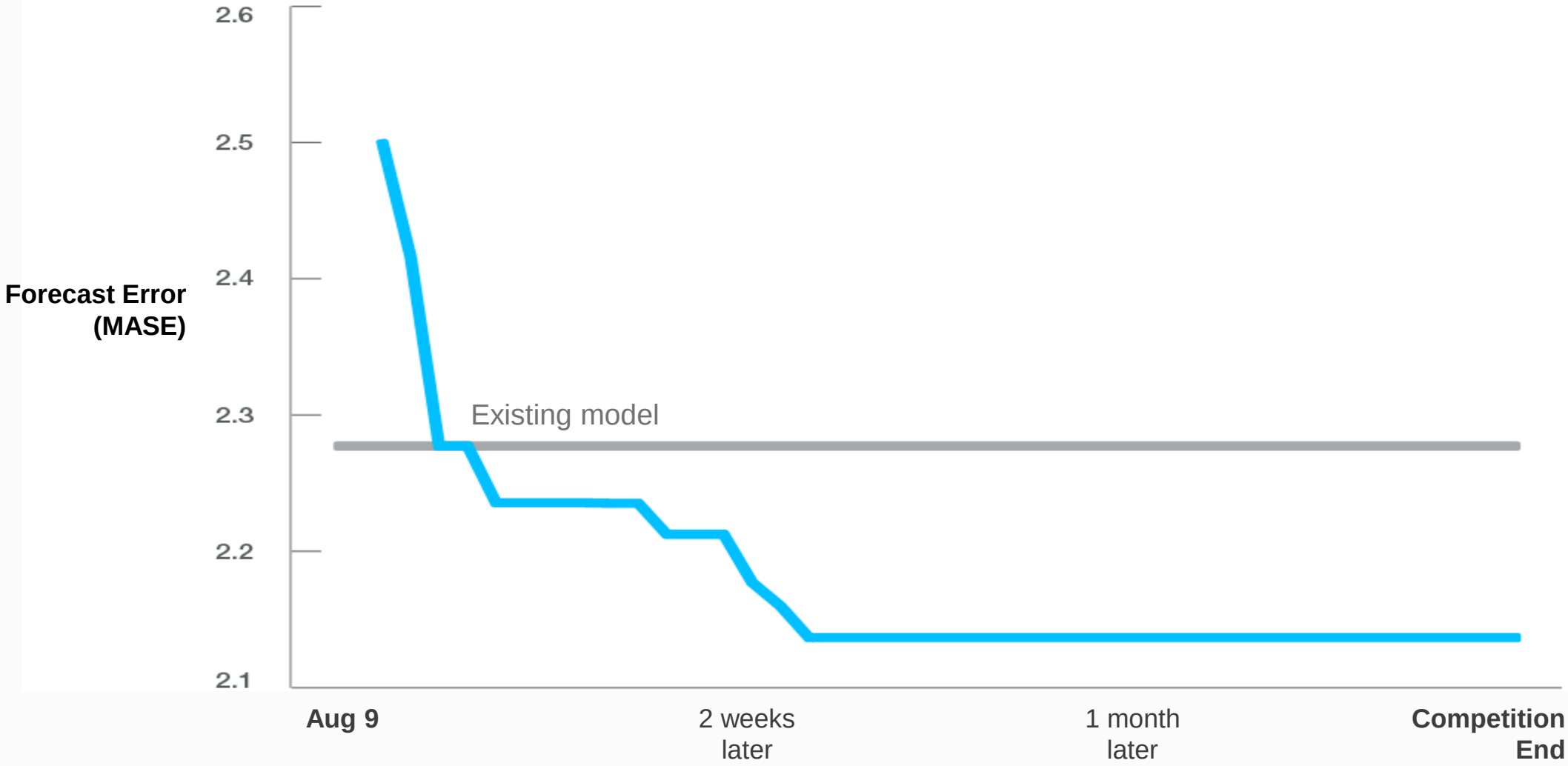


Crowdsourcing

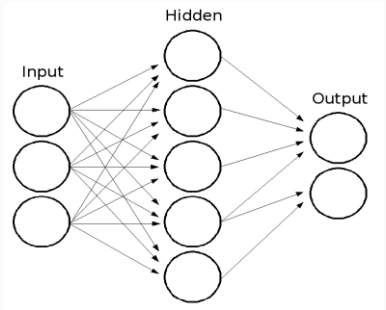


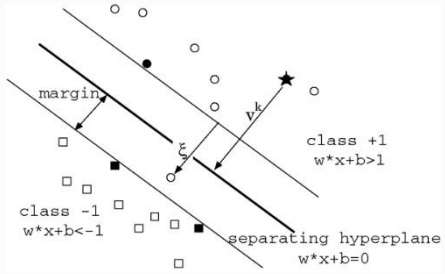
Mismatch between those with data and those with the skills to analyse it

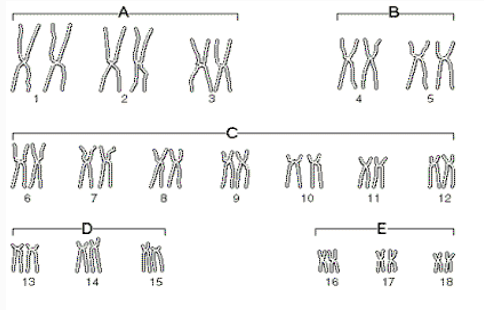
Tourism Forecasting Competition

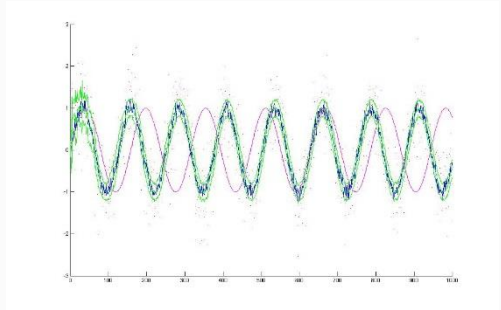


Users apply **different techniques**



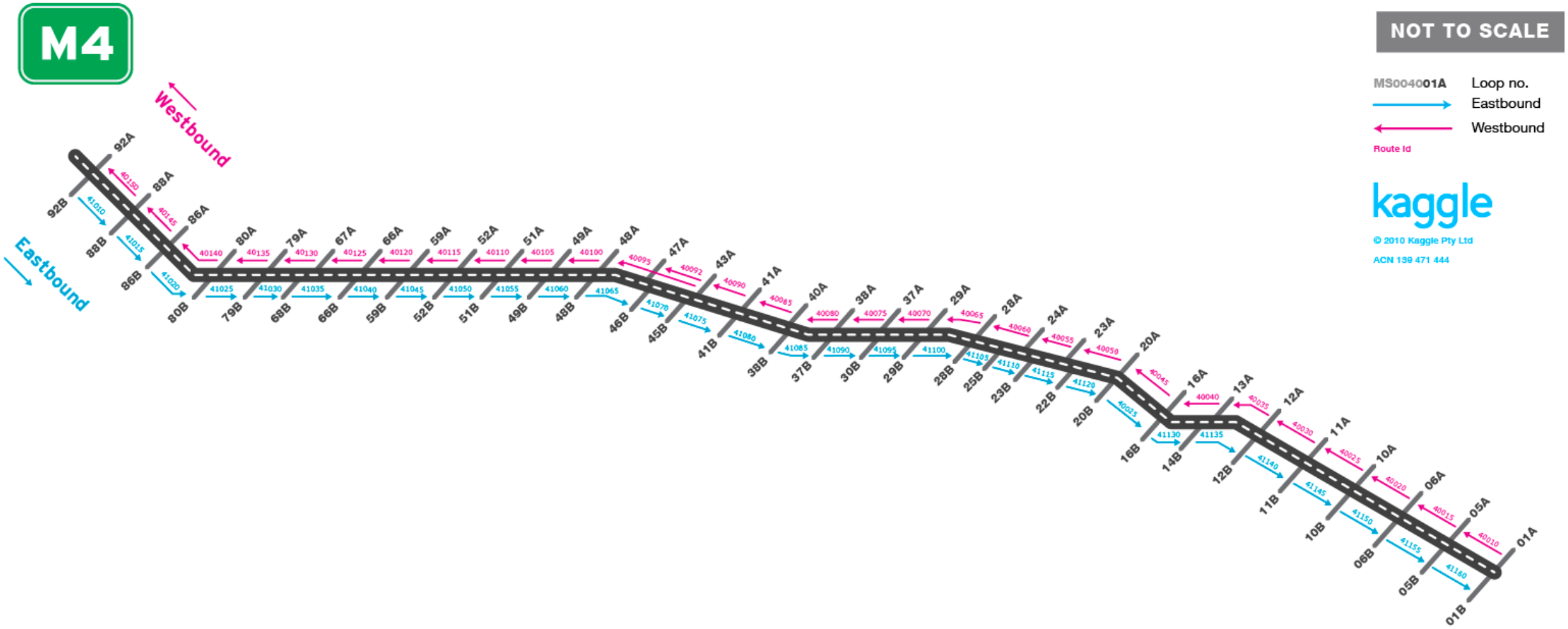




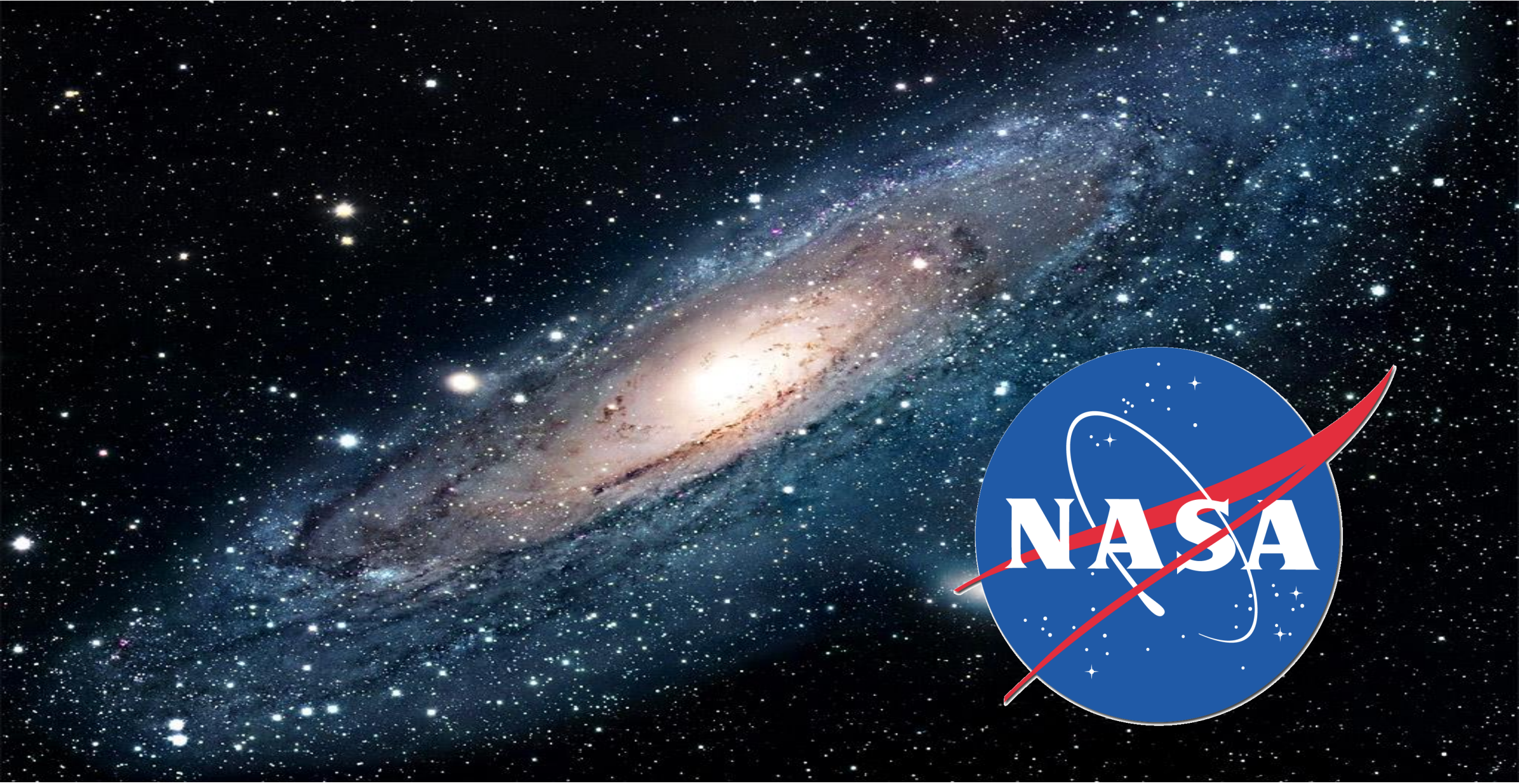


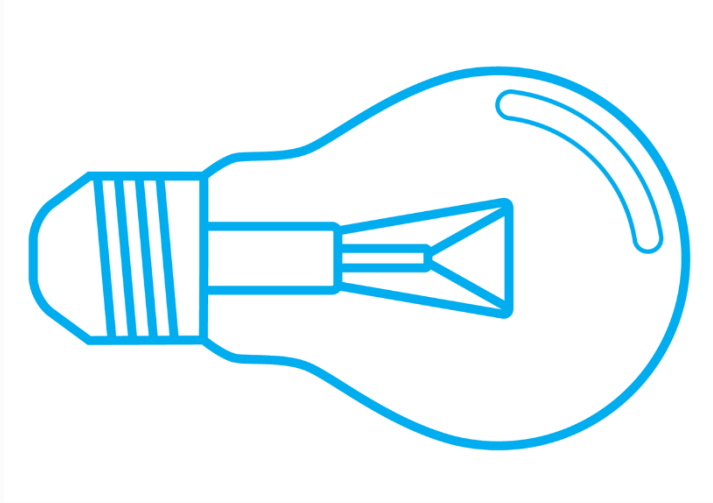
- neural networks
- logistic regression
- support vector machine
- decision trees
- ensemble methods
- adaBoost
- Bayesian networks

- genetic algorithms
- random forest
- Monte Carlo methods
- principal component analysis
- Kalman filter
- evolutionary fuzzy modeling



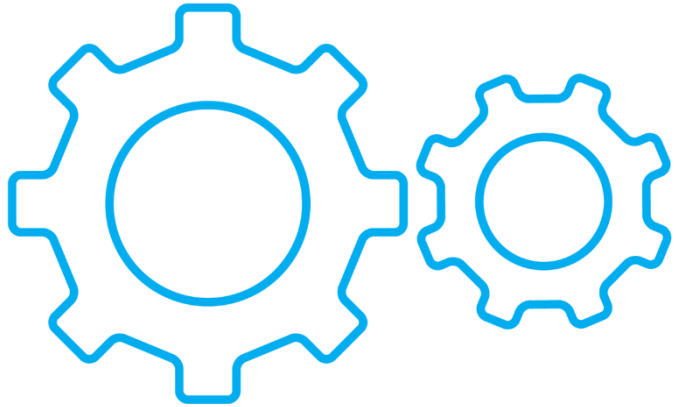
VicRoads has an algorithm they use to forecast travel time on Melbourne freeways (taking into account time, weather, accidents, etc). Their current model is inaccurate and somewhat useless. They want to do better (or at least find out about whether it's possible to do better).


NASA



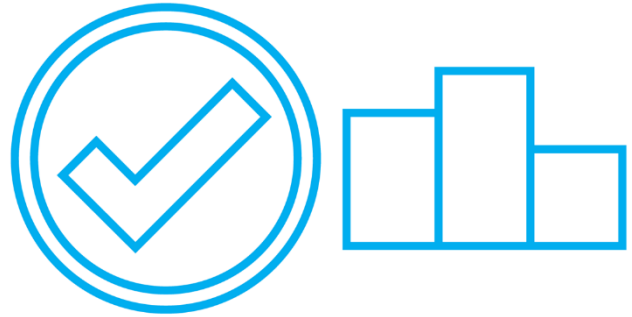
1

Upload



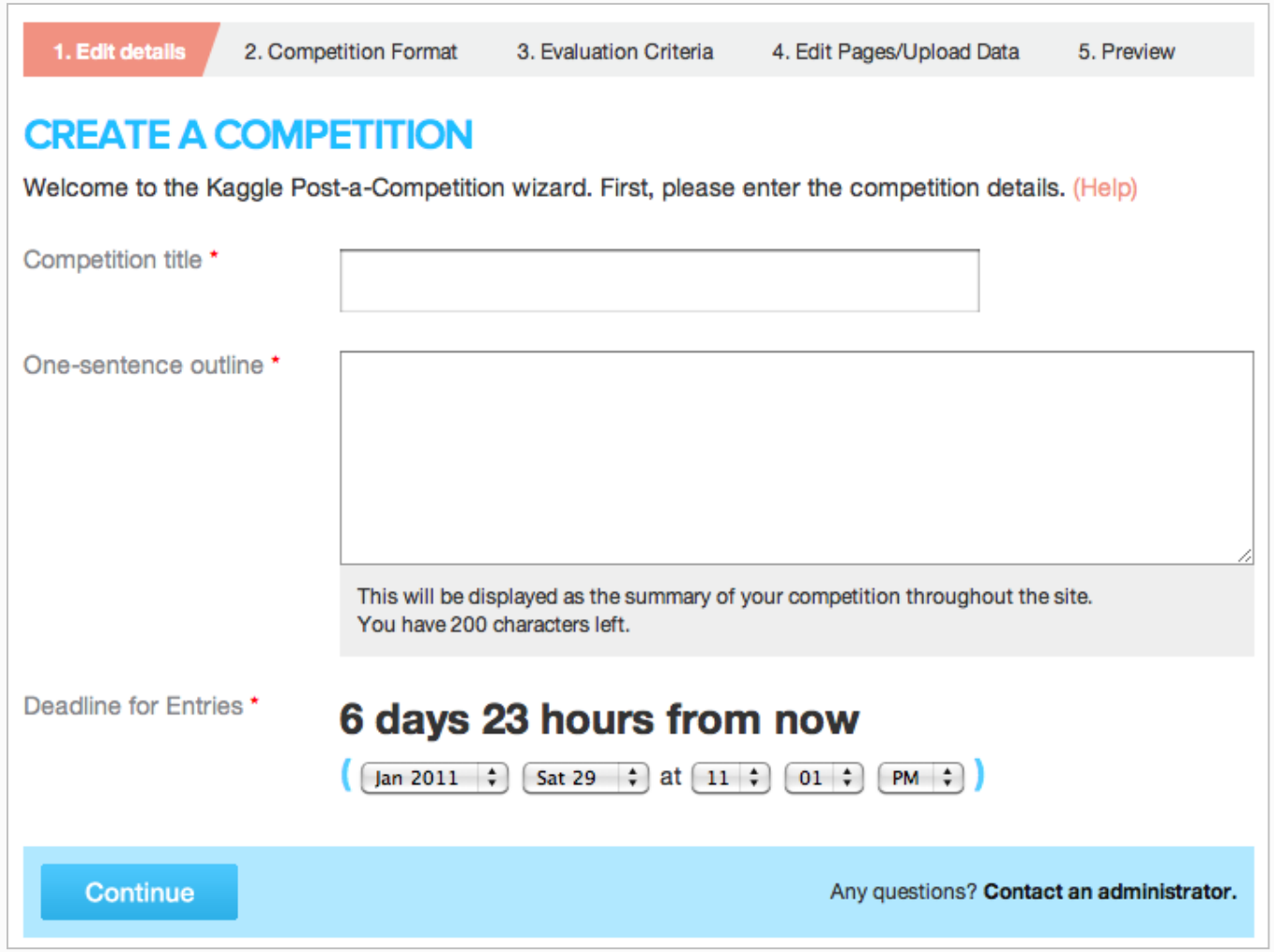
2

Submit



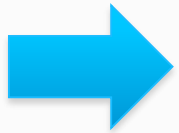
3

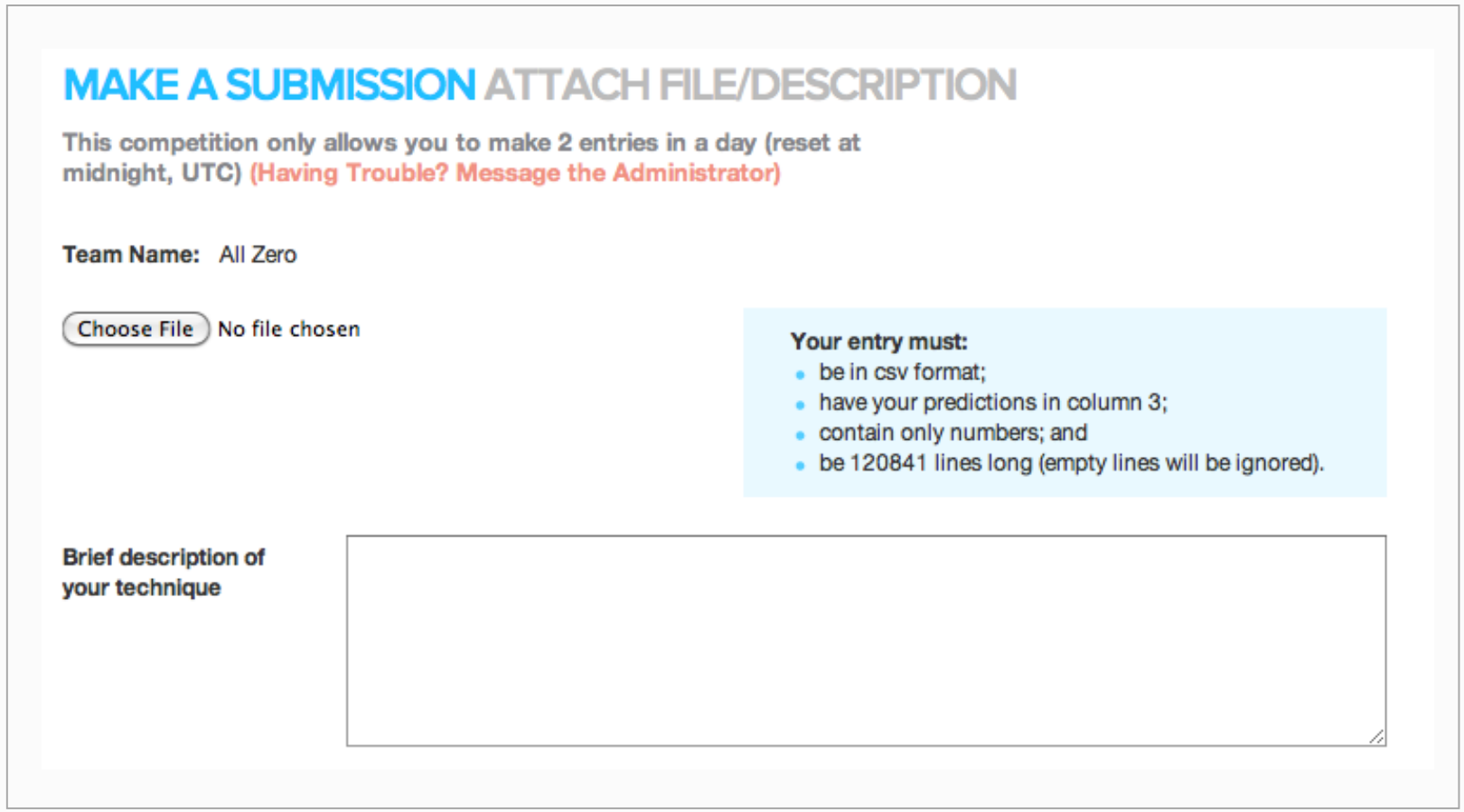
Evaluate & Exchange



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





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

Training dataset		
Age	*Income*	*Default*
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

Test dataset		
Age	*Income*	*Default*
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	
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	
3	0	1	?	38.3609	15.3212	1000	60	0.302277	508	118.11	0	0	
4	0	2	?	38.2342	11.514	1000	60	0.302277	508	118.11	0	0	
5	0	3	?	37.9304	12.2615	1000	60	0.302277	508	118.11	0	0	
6	0	4	?	37.8085	12.3666	1000	60	0.302277	504	119.048	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

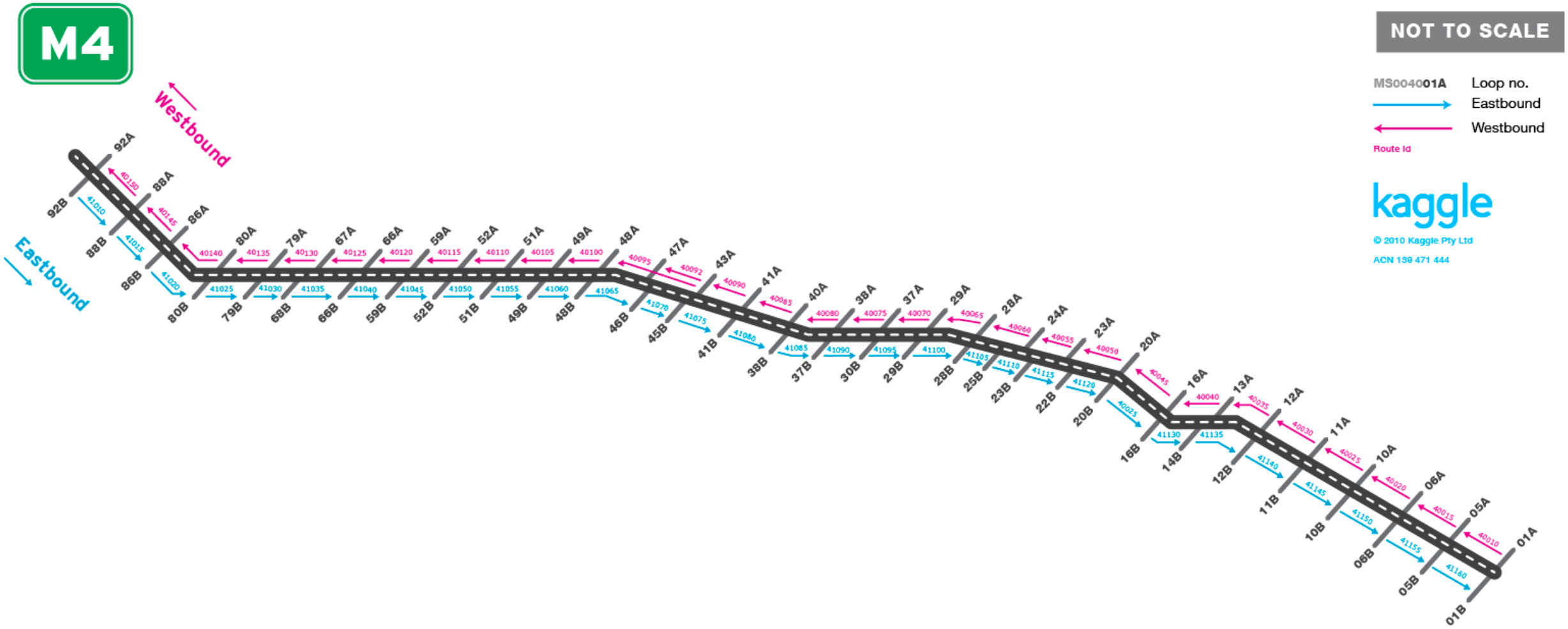
Inference (#1)

- *"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)*



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

#	Δ1w	Team Name	RMSE	Entries	Last Submission UTC (Best Submission – Last)
1	↑3	Team Irazu *	191.47400	25	Sat, 12 Feb 2011 17:42:55 (-20.1h)
2	-	Moomo	191.63600	134	Tue, 08 Feb 2011 19:25:17 (-41.8h)
3	↑27	SuperCow	192.23199	16	Fri, 11 Feb 2011 15:58:38
4	↓1	zkolter	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	195.62300	76	Sun, 13 Feb 2011 20:22:22 (-27.2h)
9	↑1	Femen	196.66000	12	Sun, 13 Feb 2011 17:55:26 (-5d)
10	↓2	pilg	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 (-4.9d)
14	↓2	tguillem	199.76199	37	Sun, 13 Feb 2011 18:33:03 (-64.1d)
15	↓1	Mansy	199.77901	35	Sun, 13 Feb 2011 21:38:16 (-4.9d)
16	↑3	uqwn	200.14000	26	Sun, 13 Feb 2011 06:46:00 (-0.3h)
17	↓4	NosferatoCorp	200.52800	19	Wed, 02 Feb 2011 18:27:42 (-2.6d)
18	↓3	Jerzy Kiełbra	200.76700	4	Fri, 03 Dec 2010 02:01:07 (-0.1h)

- Francois GUILLEM
- Andrzej Janusz

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:

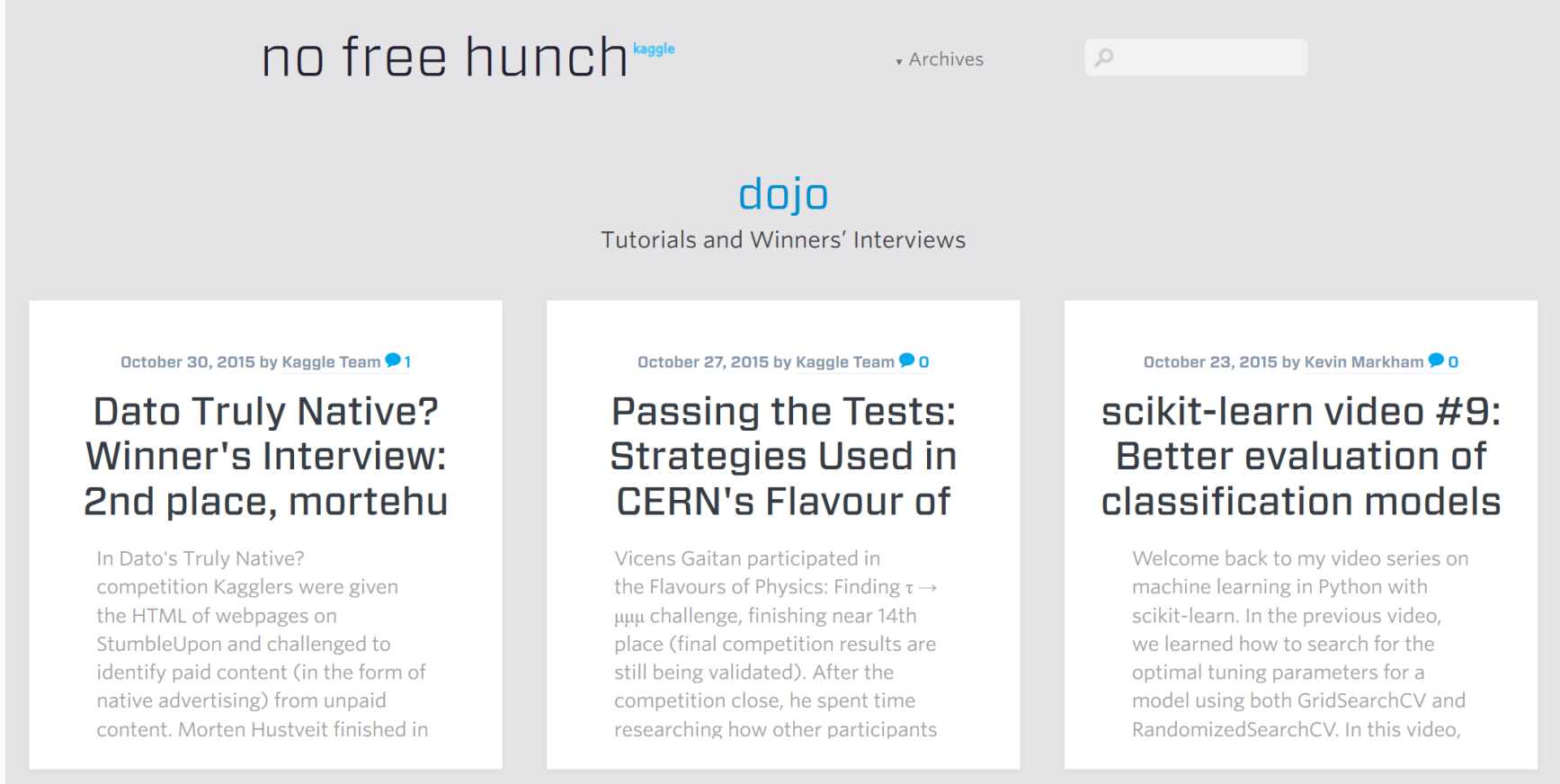
- *Java/Weka*
- *4 processors, 12 GB RAM*
- *48 hours of computations*

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

Kaggle experience…

Homework 4



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

Many pages of interview on how the team(s) built their models, some have multiple interviews;
You will review 5 interviews, search among them, do not go sequentially.
1) Title, 2) What model(s) did they use, 3) What influenced choice of model and insight from model? 4) Did they create new features and did they deploy any feature selection technique? 5) 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.

t Scale

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.

le

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

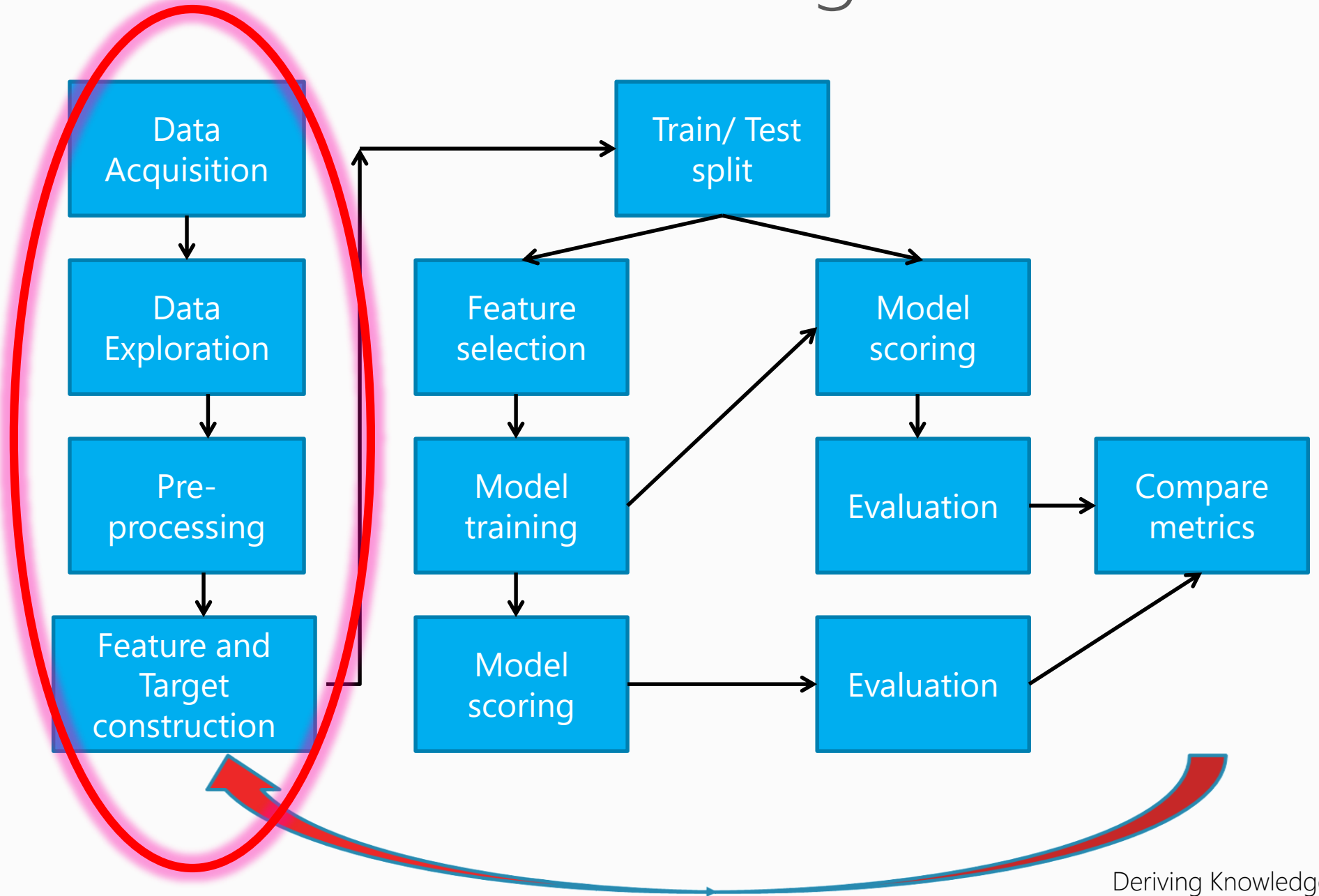
The maximum size of a team is 100 participants.

Submission Limits

You may submit a maximum of 5 entries per day.
You may select up to 6 final submissions for judging.

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

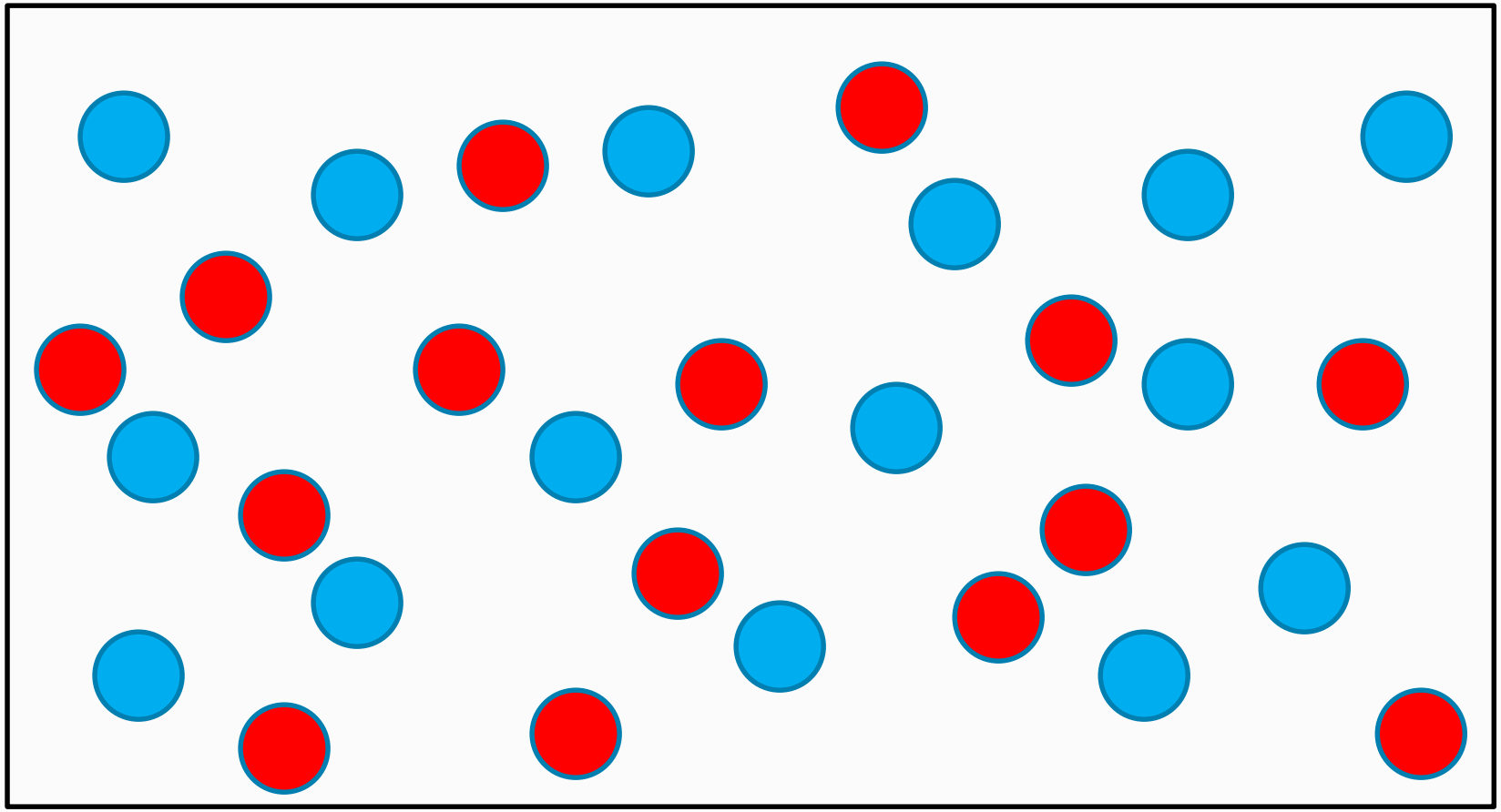
Critical step in the modeling process

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

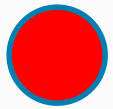
Modeling Dataset

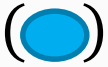
- Split the dataset into a training and testing subset.
- Splits can be
  - Random
    - Each data record is randomly assigned to either train or test
  - Stratified
    - Assign so that each split (train/test) has the same relative proportions of each pair of $(\boldsymbol{x}, \boldsymbol{y})$ E.g.,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

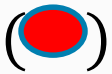
Modeling Dataset



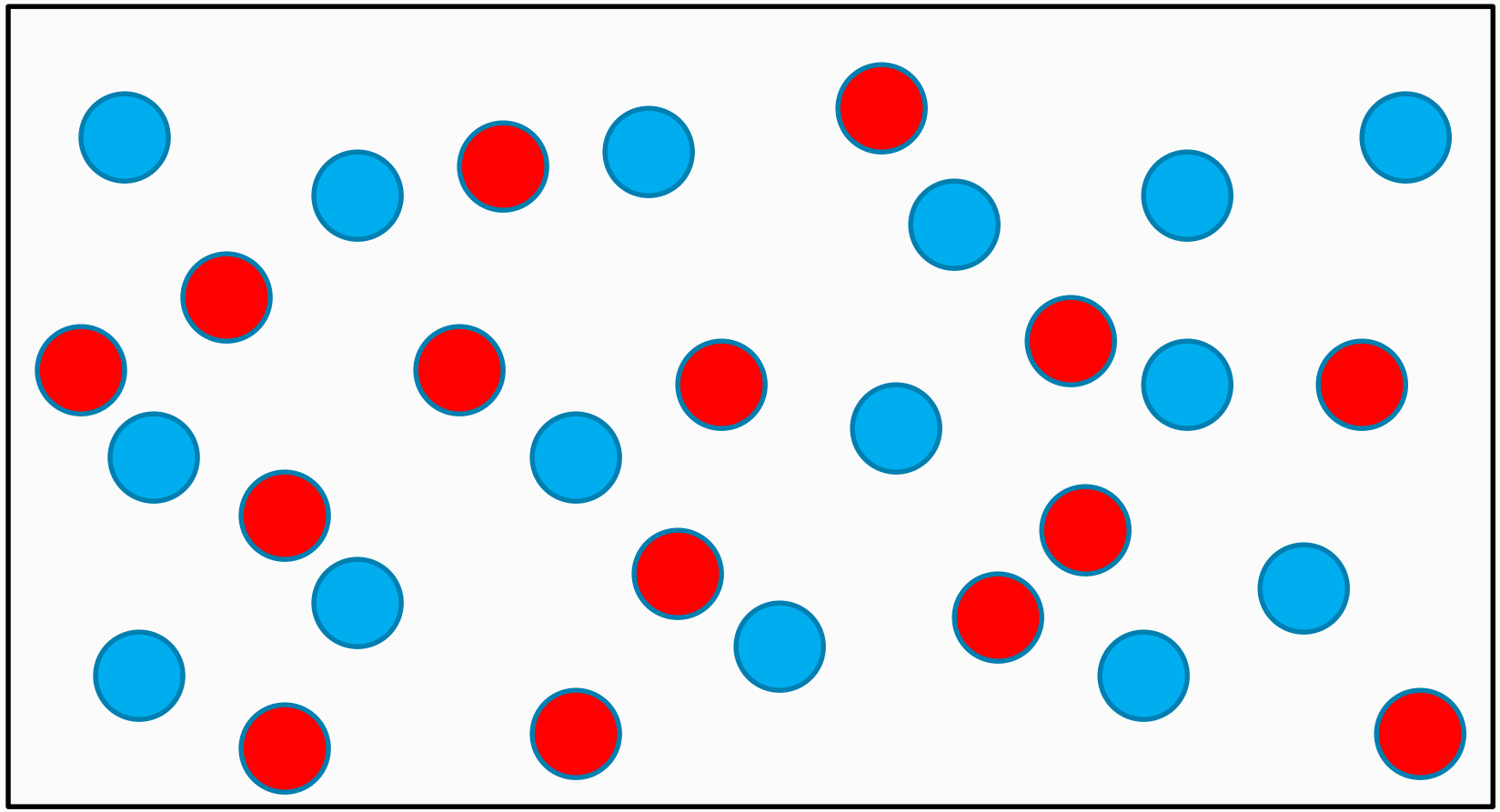
15 

15 

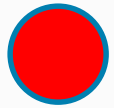
$P($  $) = 0.5$

$P($  $) = 0.5$

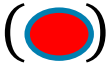
Train/Test Split: Random



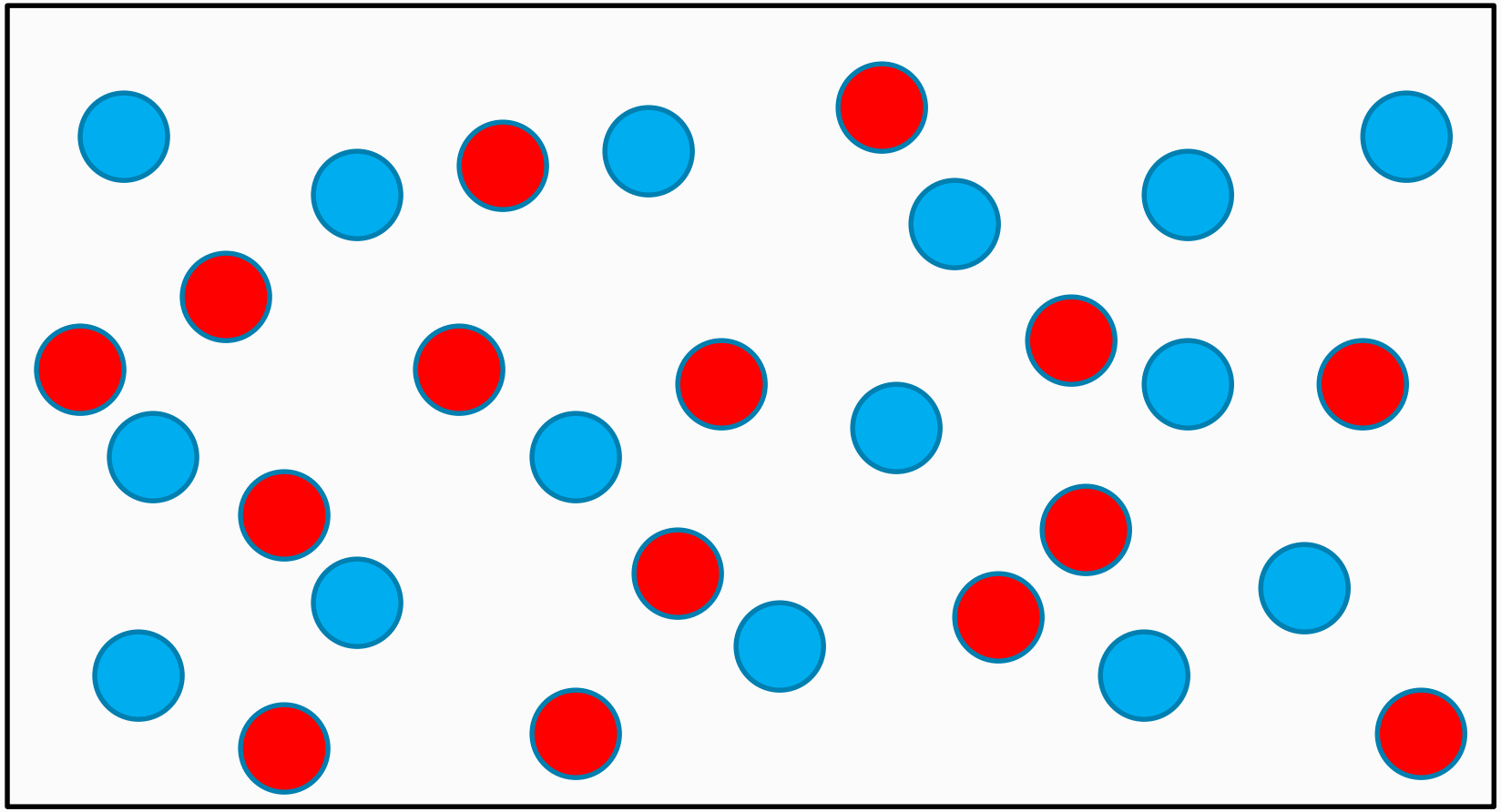
7 

13 

$P($  $) = 0.35$

$P($  $) = 0.65$

Train/Test Split: Stratified



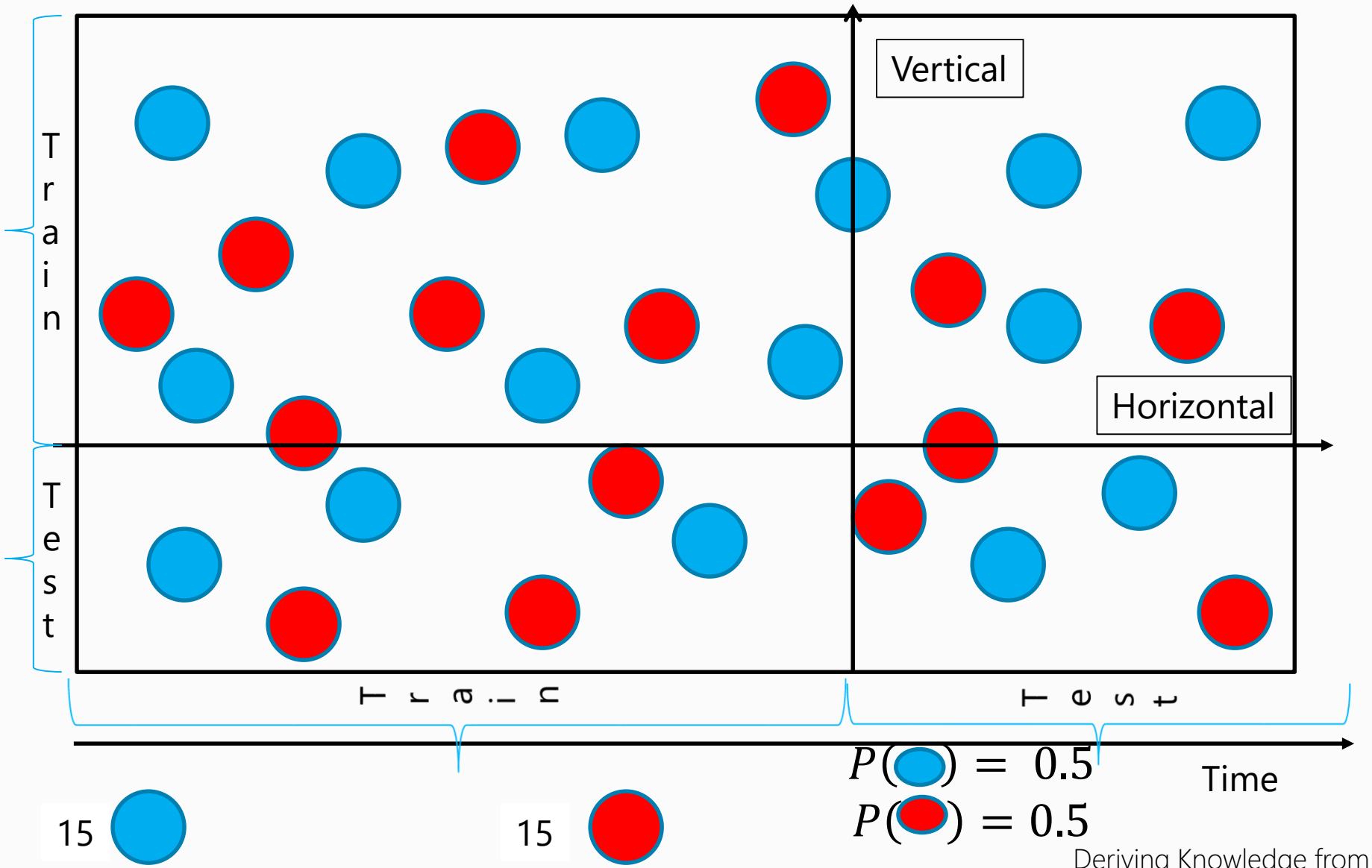
10

10

$P(\bigcirc) = 0.5$

$P(\bigcirc) = 0.5$

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!

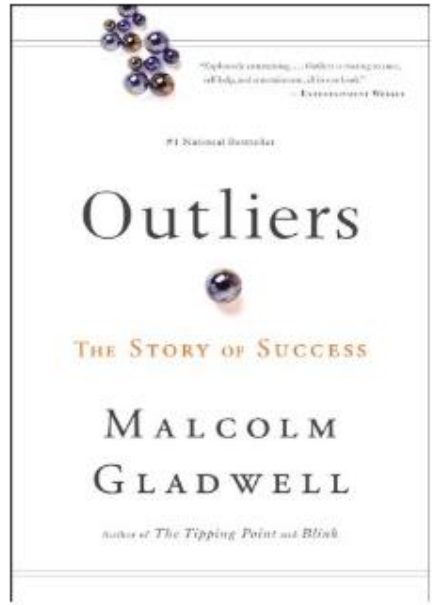
If some attributes are difficult to predict, filled-in values may be random (or worse)

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…*



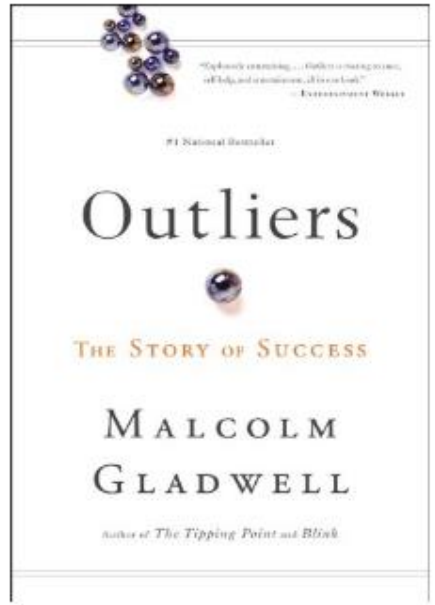
Simple working definition: an outlier is an element of a data sequence S that is inconsistent with expectations, based on the majority of other elements of S.

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…

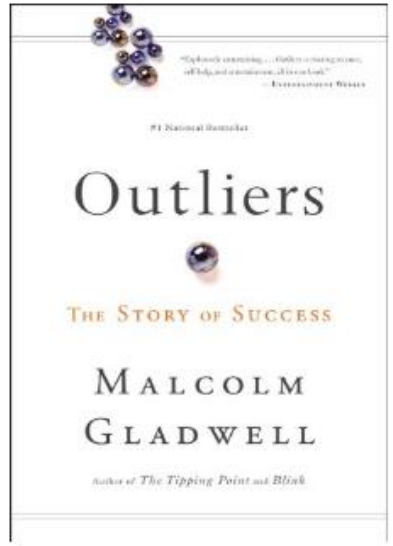
Data Cleaning

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

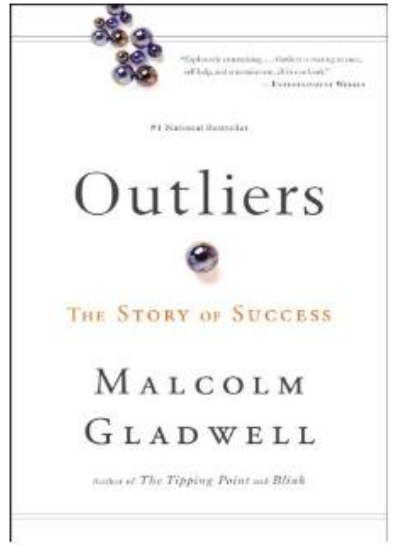
- Winsorize – Set all outliers to a specified percentile of the data. Not equivalent to trimming, which simply excludes data. In a Winsorized estimator, extreme values are instead replaced by certain percentiles (the trimmed minimum and maximum). Same as clipping in signal processing.

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

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

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

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

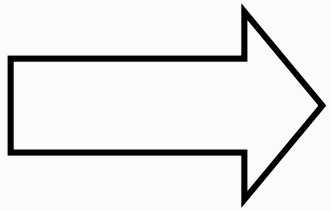
- Lower case
- Spellcheck
- Data extraction (e.g. regular expressions)

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



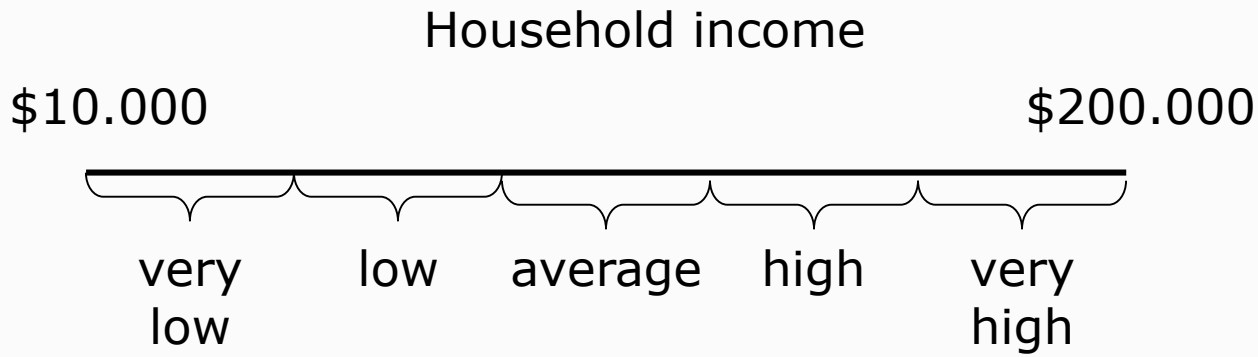
red	blue	green
1	0	0
0	1	0
0	0	1
1	0	0
1	0	0
0	0	1
0	1	0

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

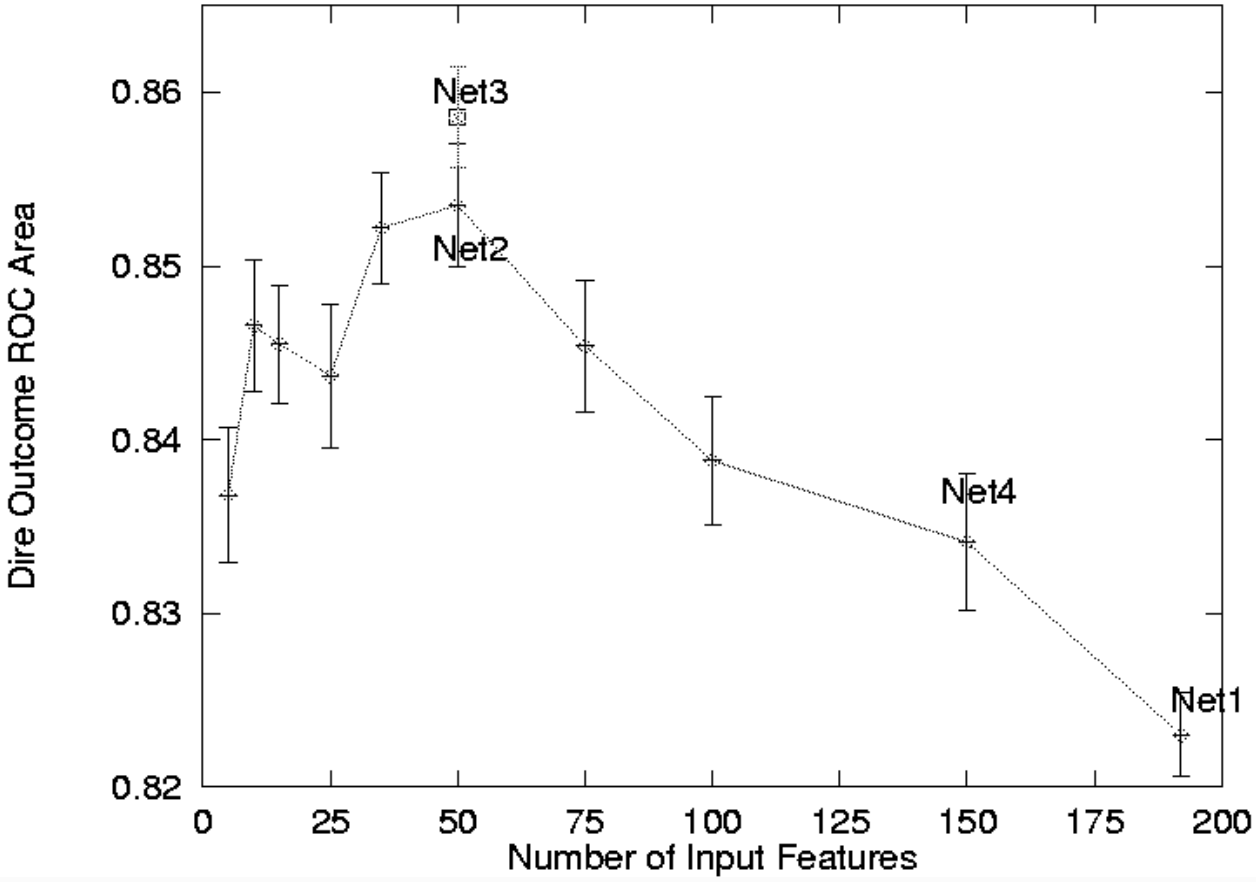
Data Integration

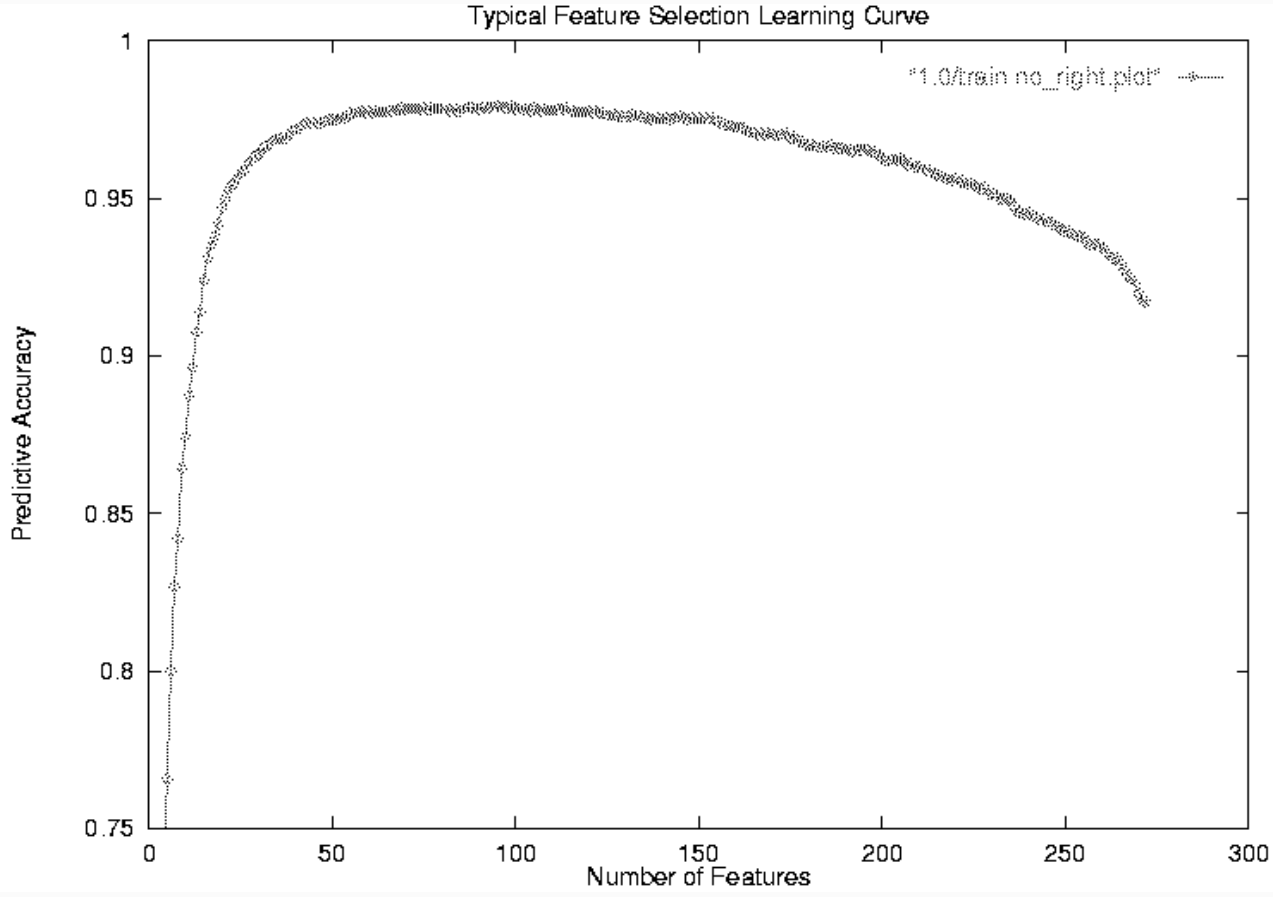
Most learning methods implicitly do feature selection:

- **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





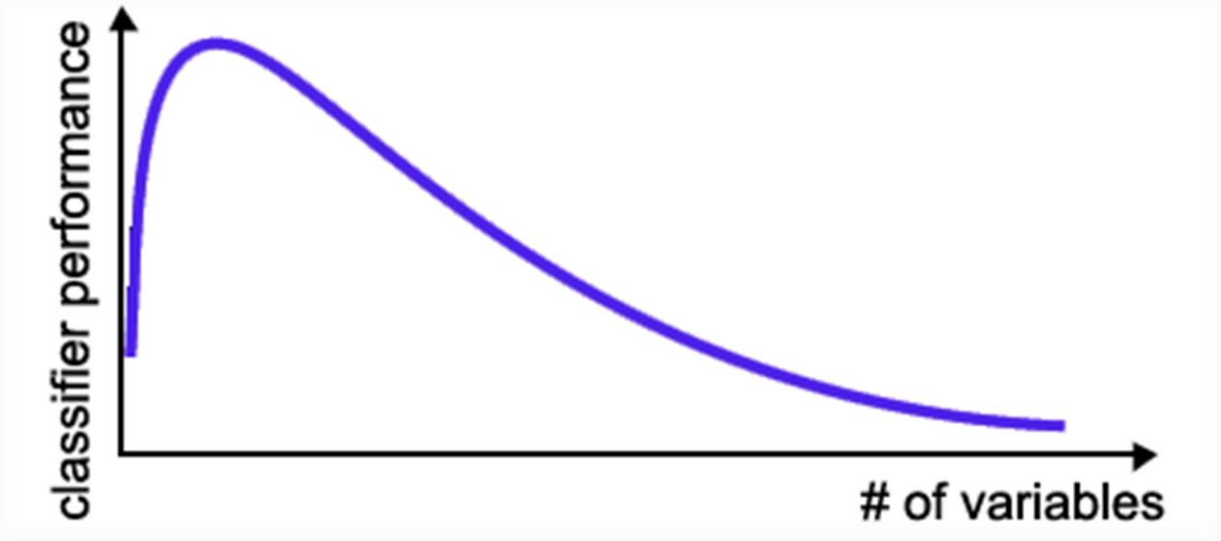
Too many redundant, irrelevant features confuse learner;
Faster training, better performance, simpler models…

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!



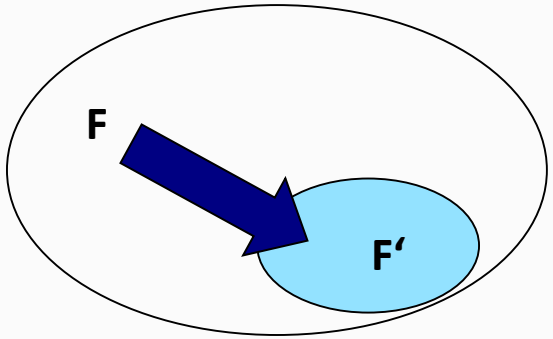
In many cases the information lost by discarding variables is made up for by a more accurate mapping/sampling in the lower-dimensional space !

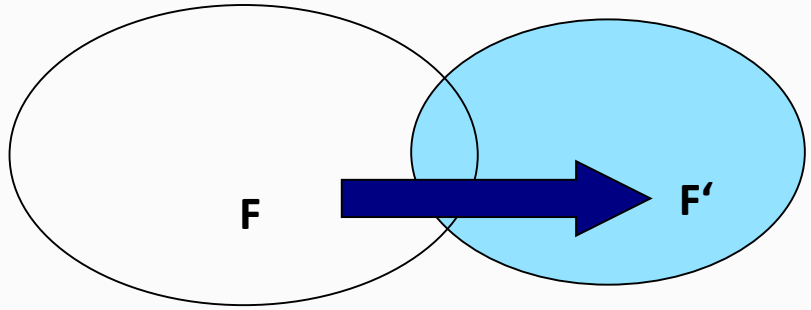
Feature Selection

Simple Definition

Given a set of features $F=\{f_1,\ldots,f_i,\ldots,f_n\}$ 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 → F'…

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





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

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

Attributes:

- Outlook (overcast, rain, sunny)
- Temperature real
- Humidity real
- Windy (true, false)
- Play (yes, 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
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.

Normalization

- **Min-max normalization**: linear transformation from v to v′
  - v′ = (v – min)/((max - min)*(newmax - newmin)) + new min
  - Ex: transform $30000 between [10000..45000] into [0..1]
    ==> (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-Mean)/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 ==> 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

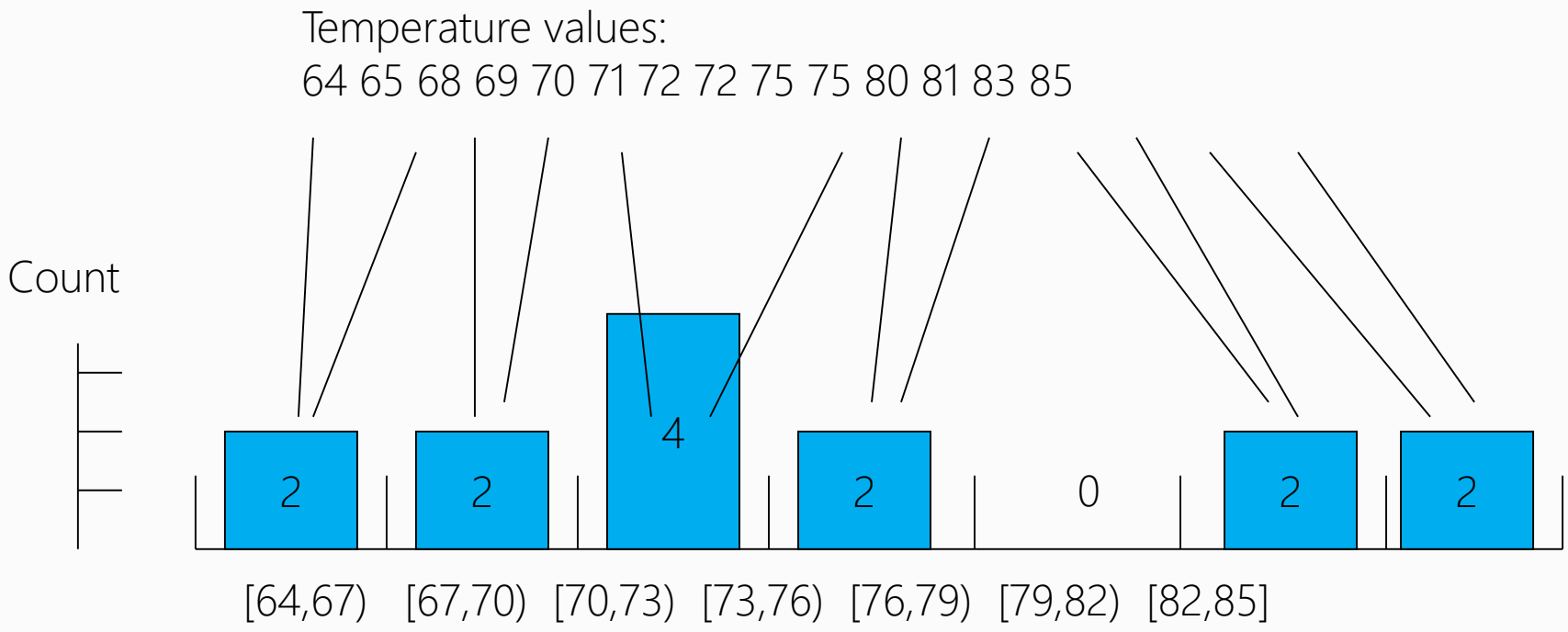
- Class label based partitioning
  - χ^2, CAIM, CAIR Discretization
  - Maximum Entropy Discretization

Equal width partitioning

1. Find the minimum and maximum values for the continuous feature/attribute F_i

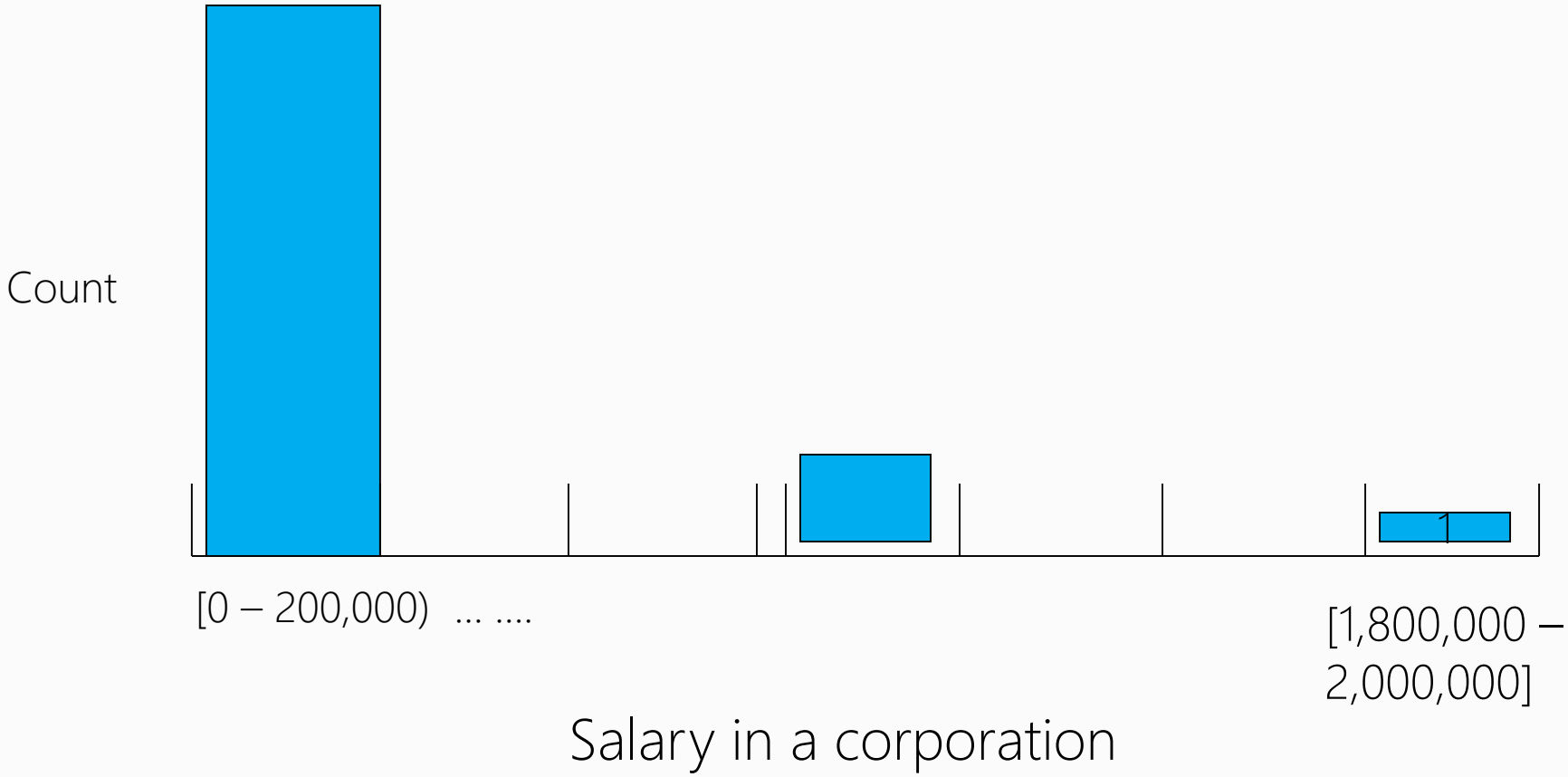
2. Divide the range of the attribute F_i into the user-specified, n_{Fi} ,equal-width discrete intervals

Equal-Width Partitioning



Equal Width, bins Low <= value < High

Equal-Width partitioning can produce clumping



Equal Height partitioning

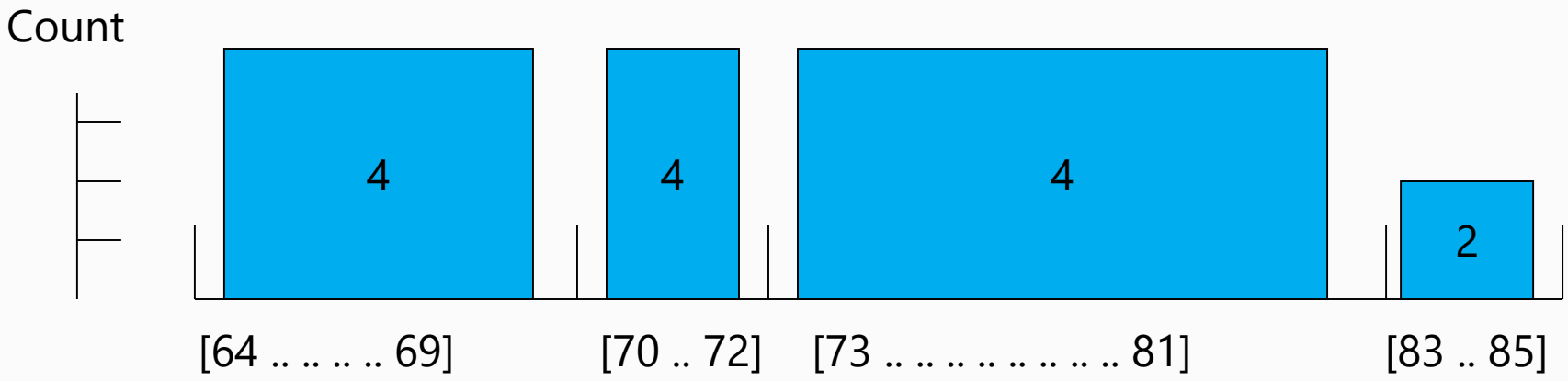
1. Sort values of the discretized feature F_i in ascending order

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_{Fi} number of intervals, where each interval contains the same number of sorted sequential values

Equal-Height partitioning

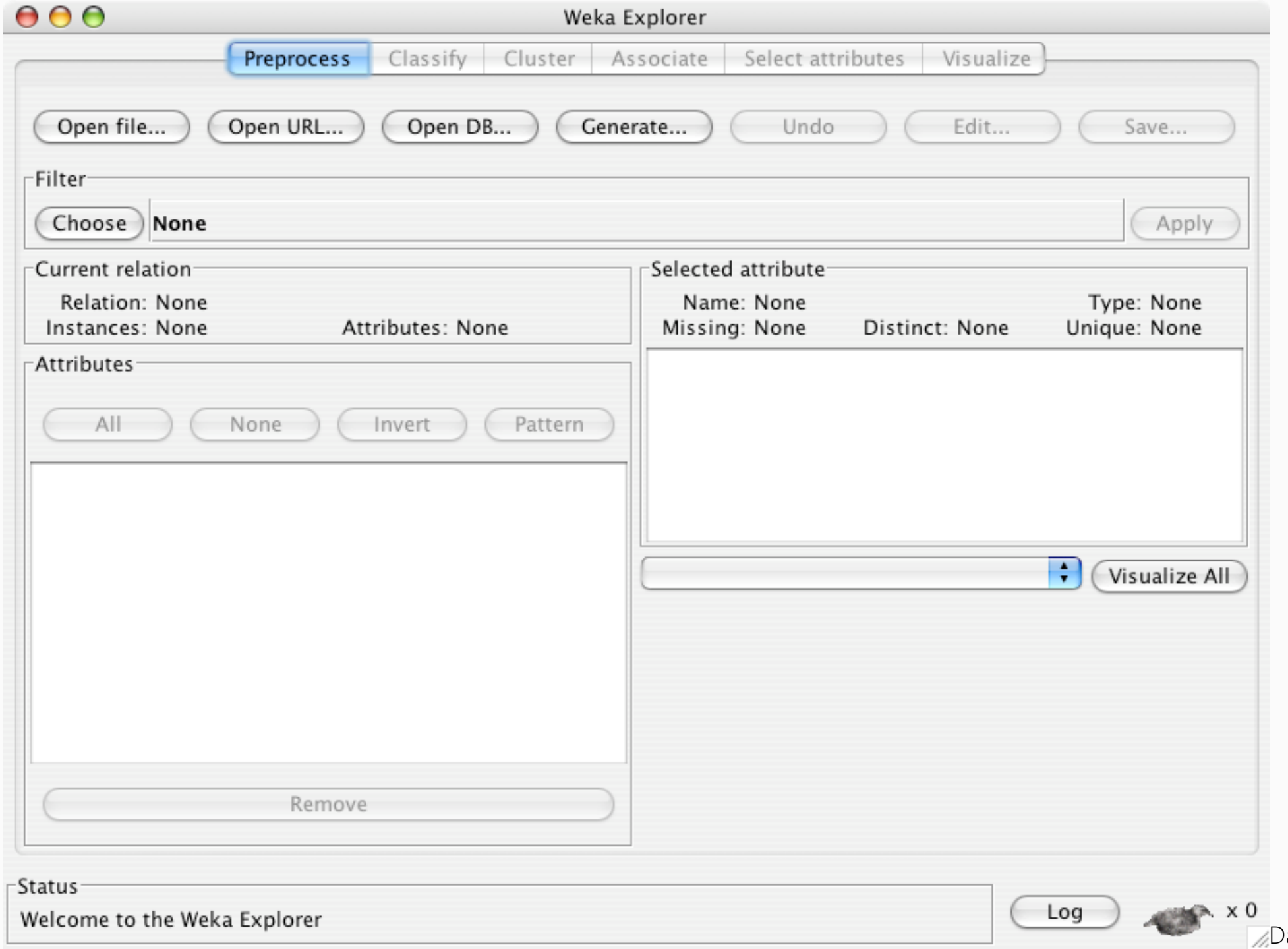
Temperature values:
64 65 68 69 70 71 72 72 75 75 80 81 83 85

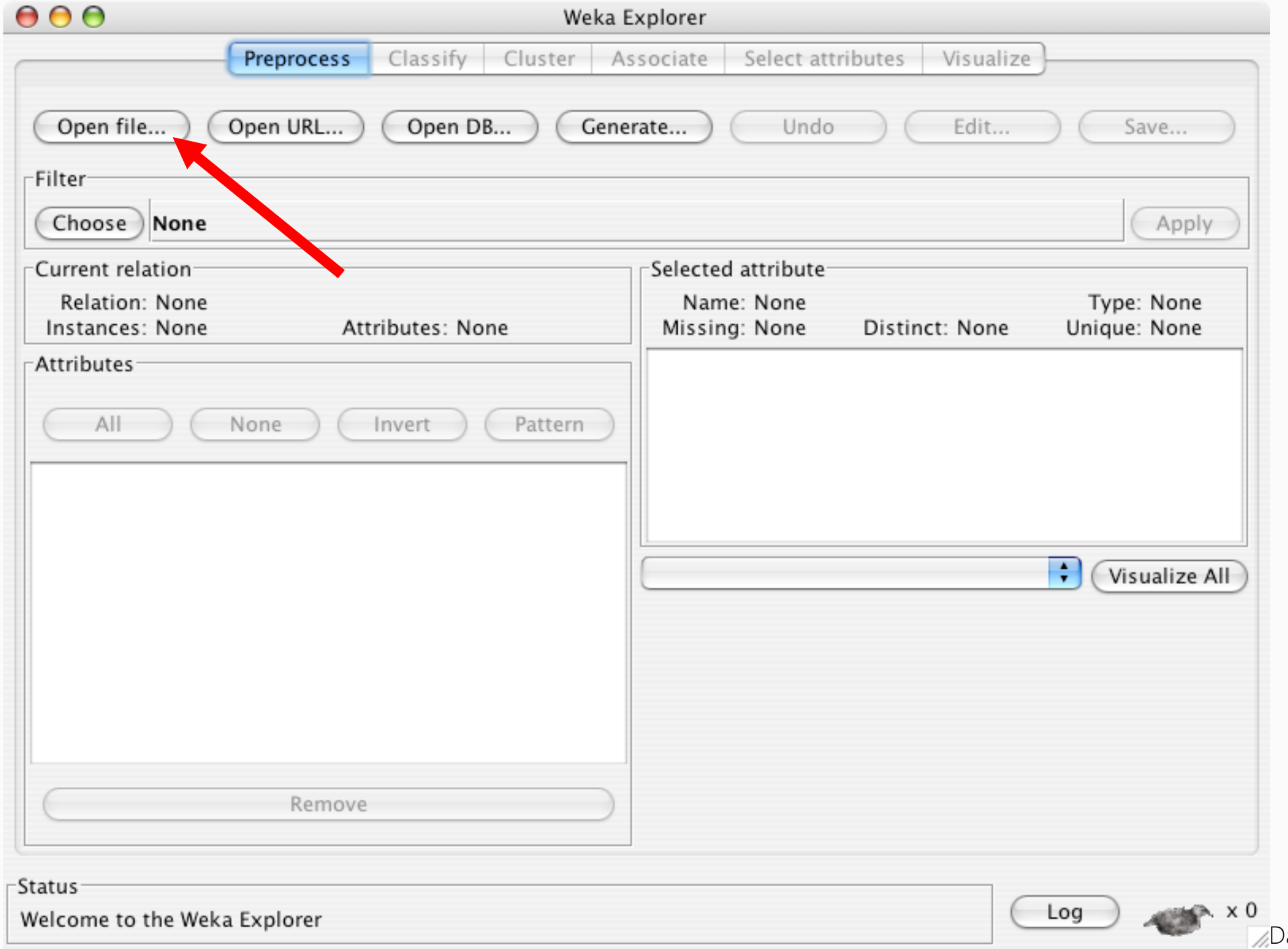


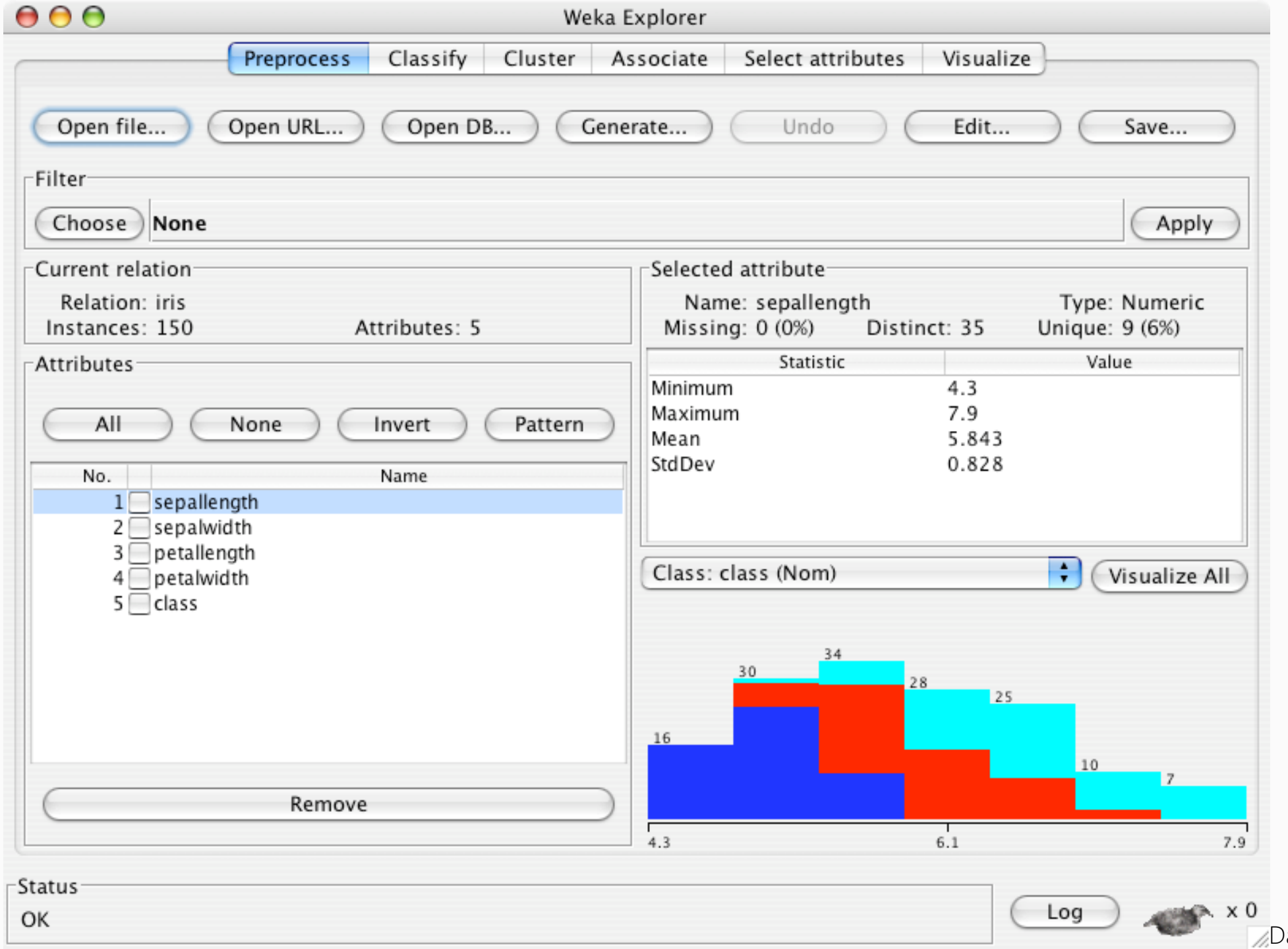
Equal Height = 4, except for the last bin

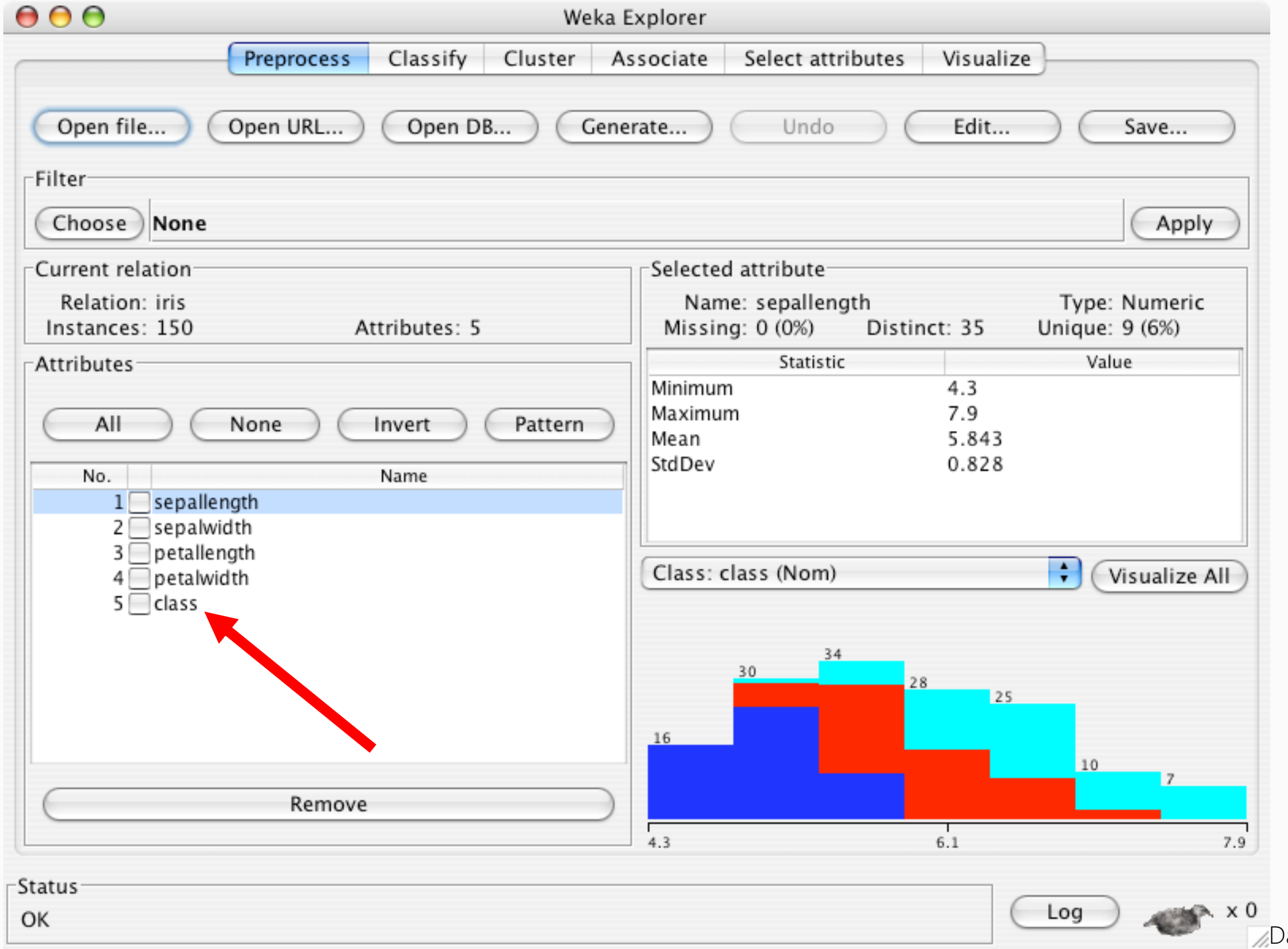
Equal-height partitioning: advantages

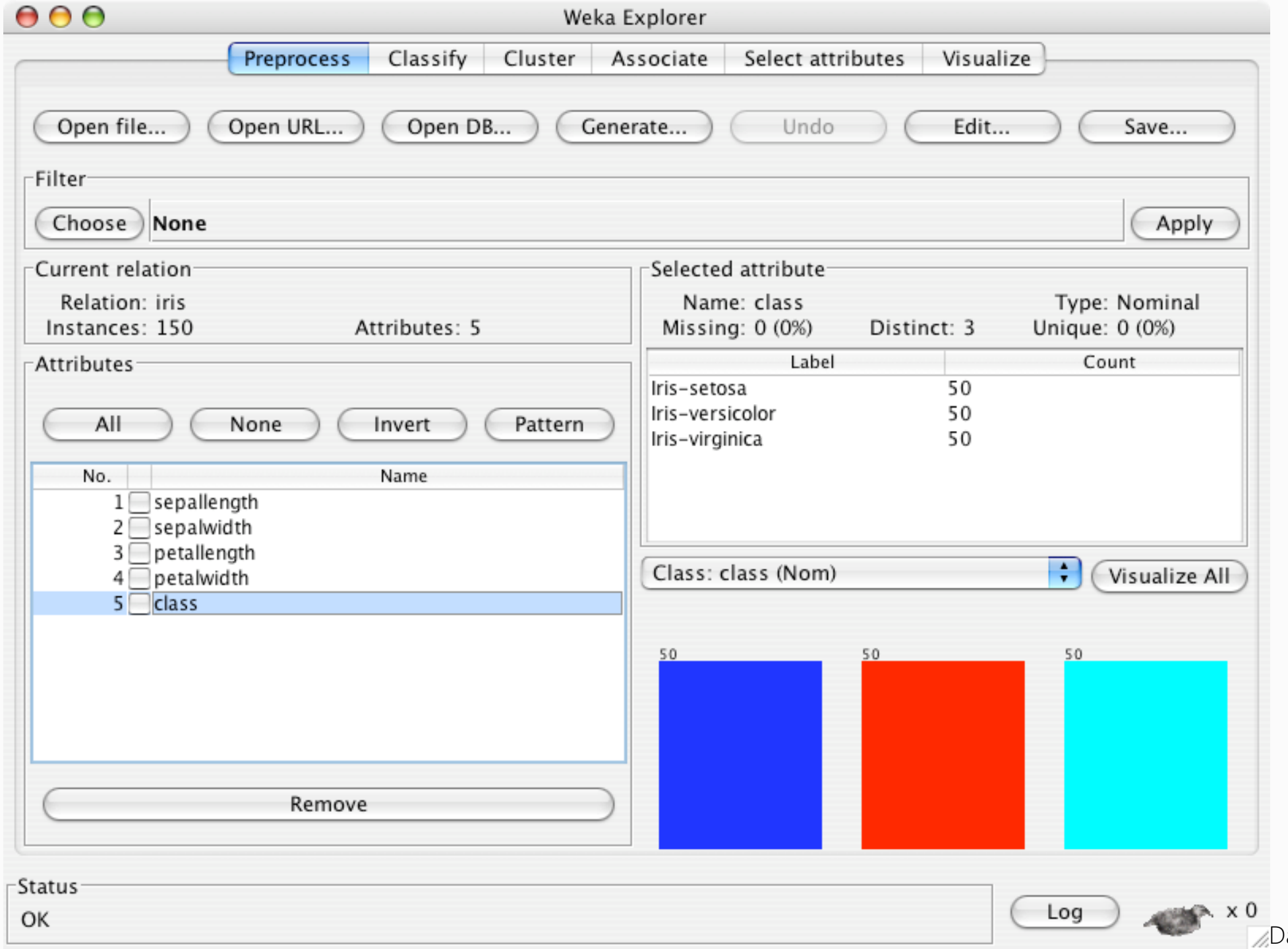
- Generally preferred because avoids clumping
- In practice, "almost-equal" height binning is used which avoids clumping and gives more intuitive breakpoints
- Additional considerations:
  - don't split frequent values across bins
  - create separate bins for special values (e.g. 0)
  - readable breakpoints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

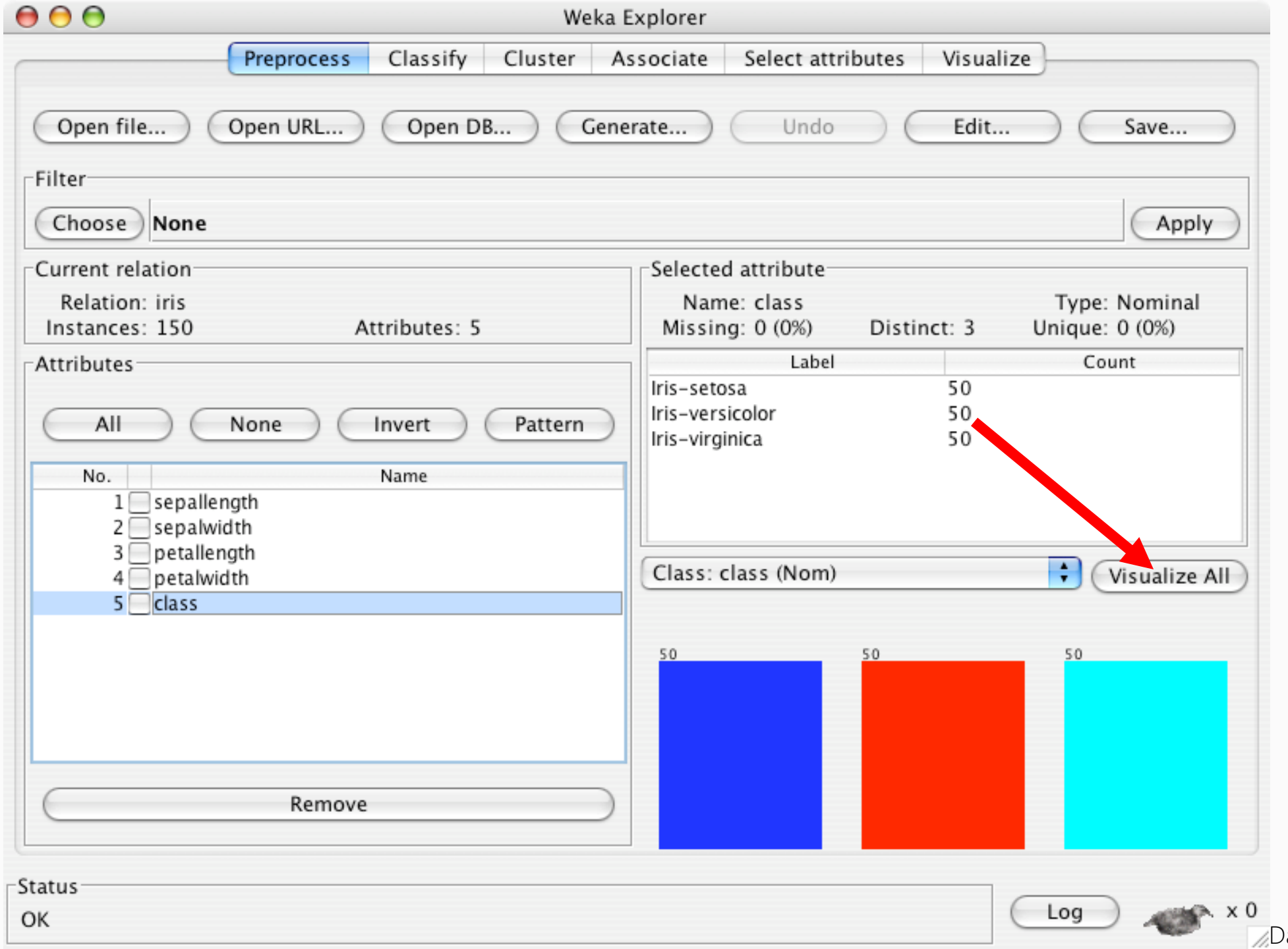

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
None
Apply
Current relation
Relation: None
Instances: None
Attributes: None
Selected attribute
Name: None
Type: None
Missing: None
Distinct: None
Unique: None
Attributes
All
None
Invert
Pattern
Remove
Visualize All
Status
Welcome to the Weka Explorer
Log
x 0

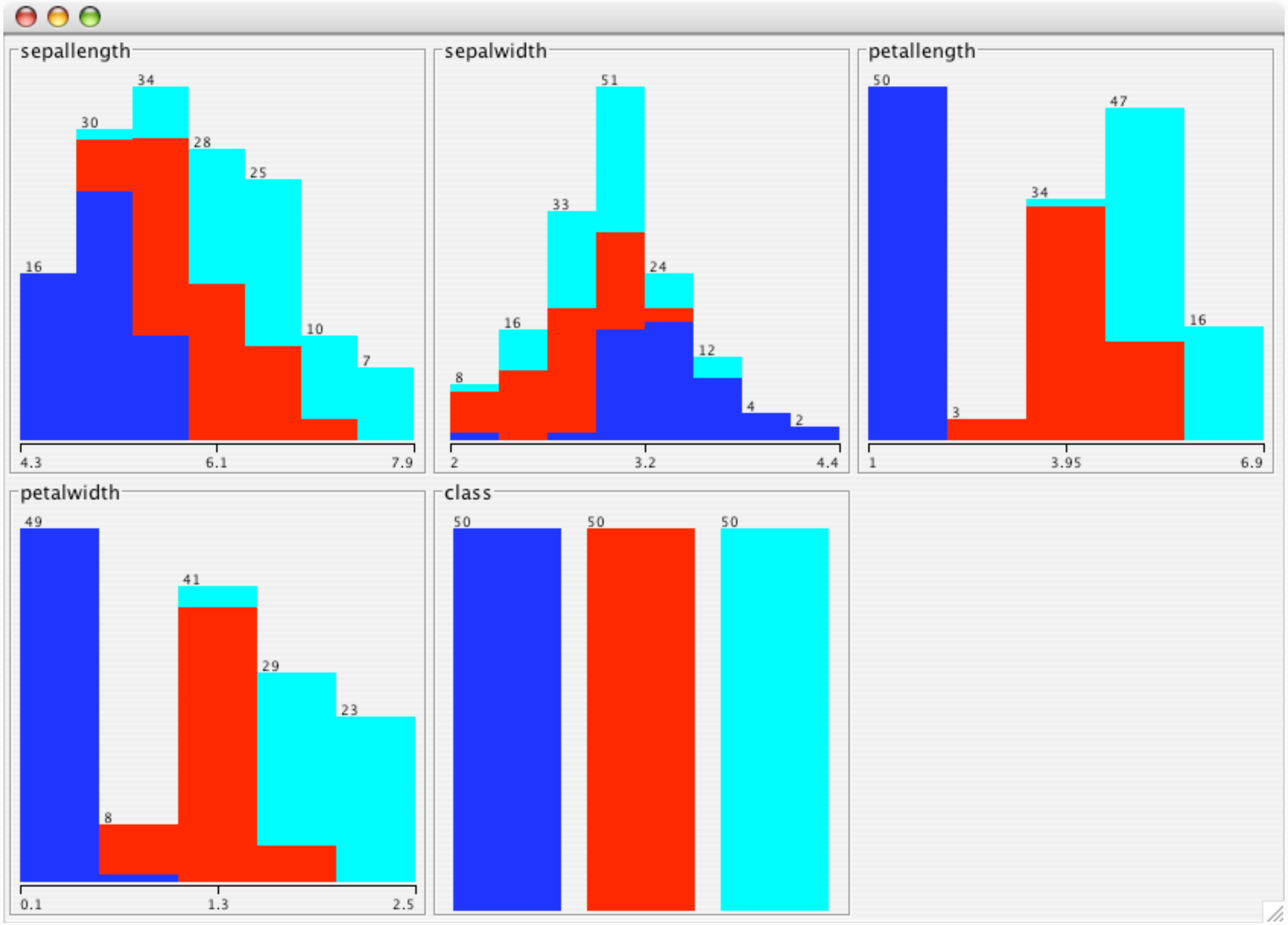

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
None
Apply
Current relation
Relation: None
Instances: None
Attributes: None
Selected attribute
Name: None
Type: None
Missing: None
Distinct: None
Unique: None
Attributes
All
None
Invert
Pattern
Remove
Visualize All
Status
Welcome to the Weka Explorer
Log
x 0

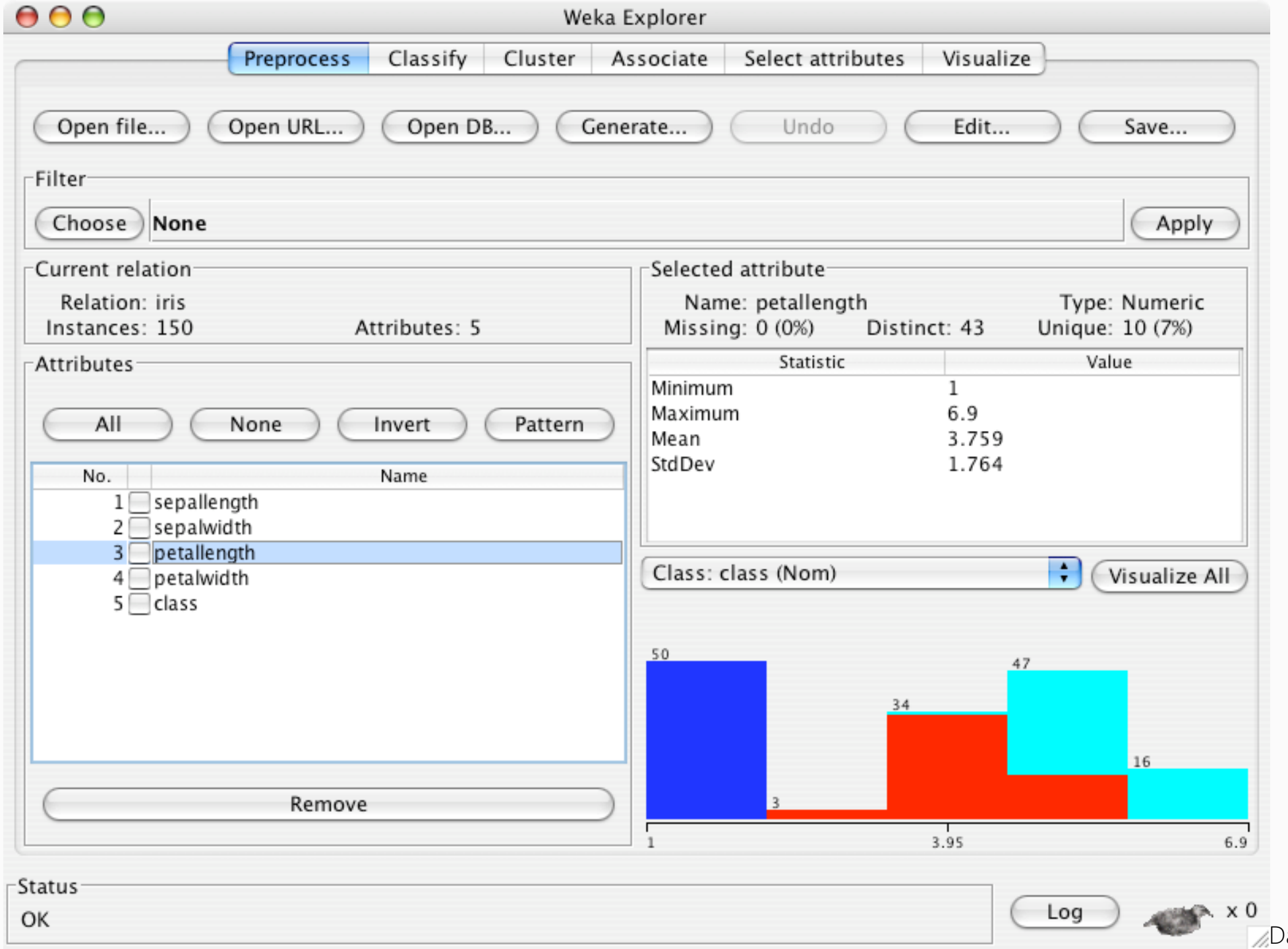

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
None
Apply
Current relation
Relation: iris
Instances: 150
Attributes: 5
Attributes
All
None
Invert
Pattern
No.
Name
1 sepallength
2 sepalwidth
3 petallength
4 petalwidth
5 class
Remove
Selected attribute
Name: 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
34
30
28
25
16
10
7
4.3
6.1
7.9
Status
OK
Log
x 0

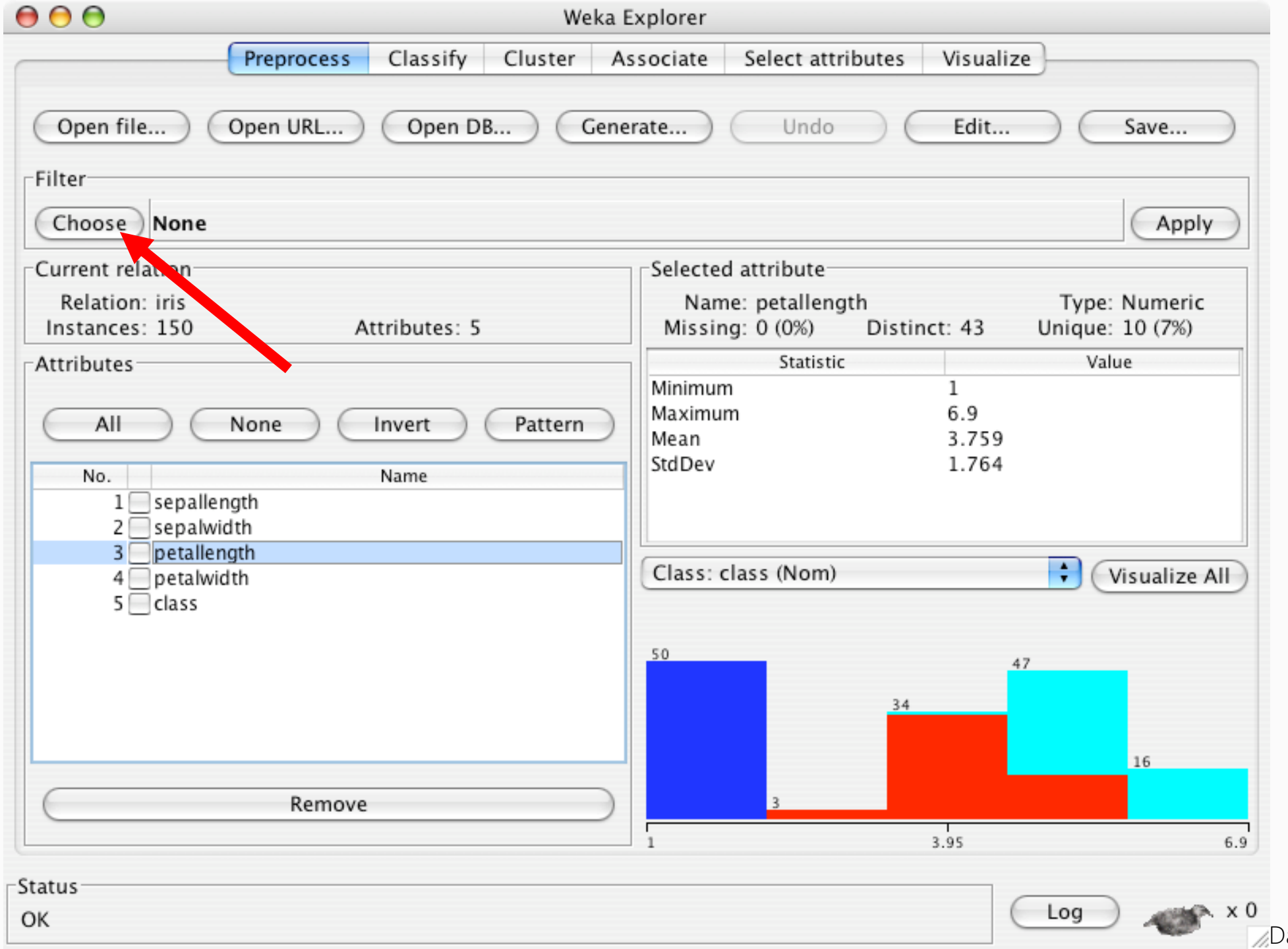

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
None
Apply
Current relation
Relation: iris
Instances: 150
Attributes: 5
Attributes
All
None
Invert
Pattern
No.
Name
1 sepallength
2 sepalwidth
3 petallength
4 petalwidth
5 class
Remove
Selected attribute
Name: 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
34
30
28
25
16
10
7
4.3
6.1
7.9
Status
OK
Log
x 0

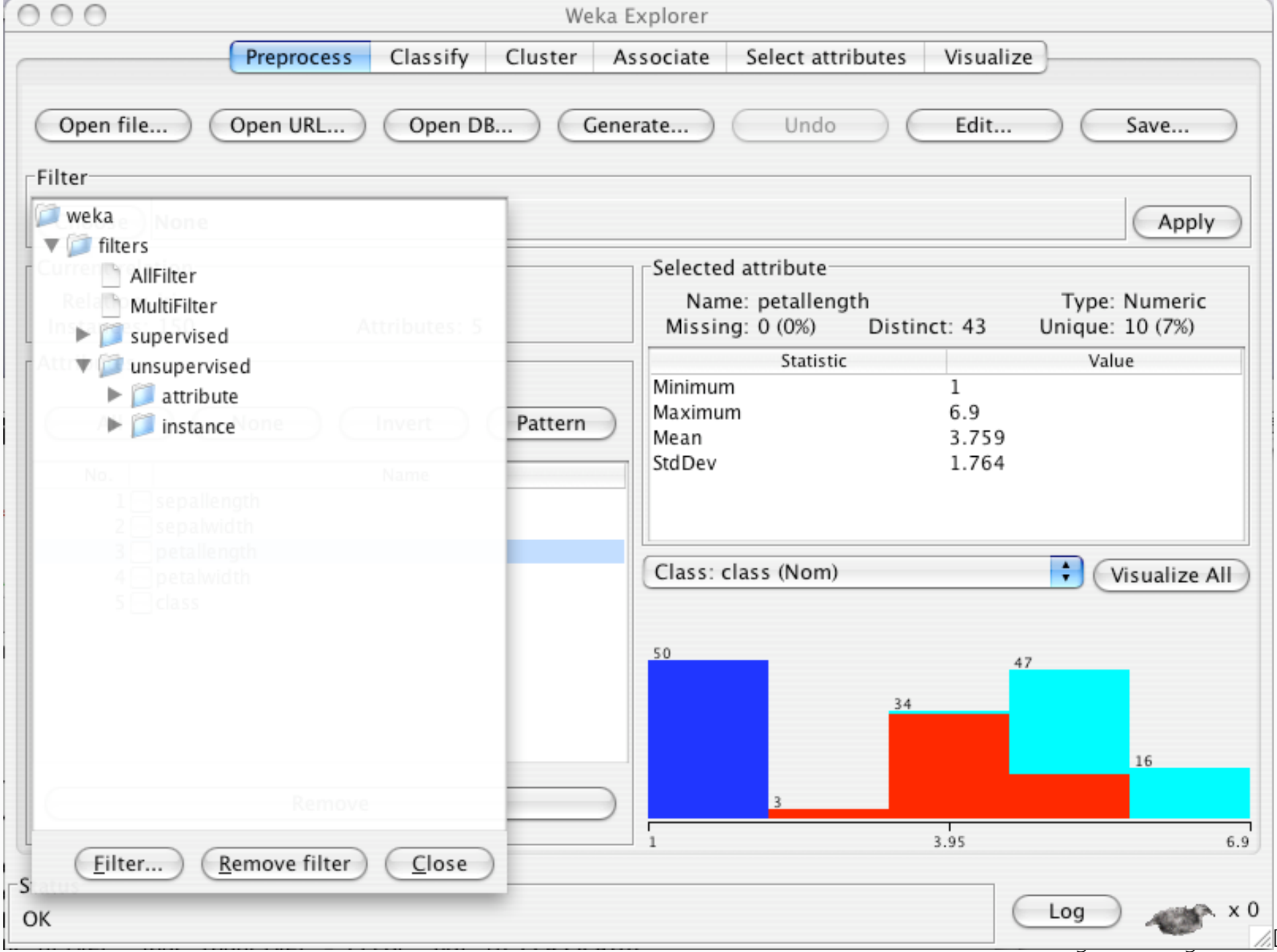

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
None
Apply
Current relation
Relation: iris
Instances: 150
Attributes: 5
Attributes
All
None
Invert
Pattern
No.
Name
1 sepallength
2 sepalwidth
3 petallength
4 petalwidth
5 class
Remove
Selected attribute
Name: class
Type: Nominal
Missing: 0 (0%)
Distinct: 3
Unique: 0 (0%)
Label
Count
Iris-setosa 50
Iris-versicolor 50
Iris-virginica 50
Class: class (Nom)
Visualize All
50
50
50
Status
OK
Log
x 0

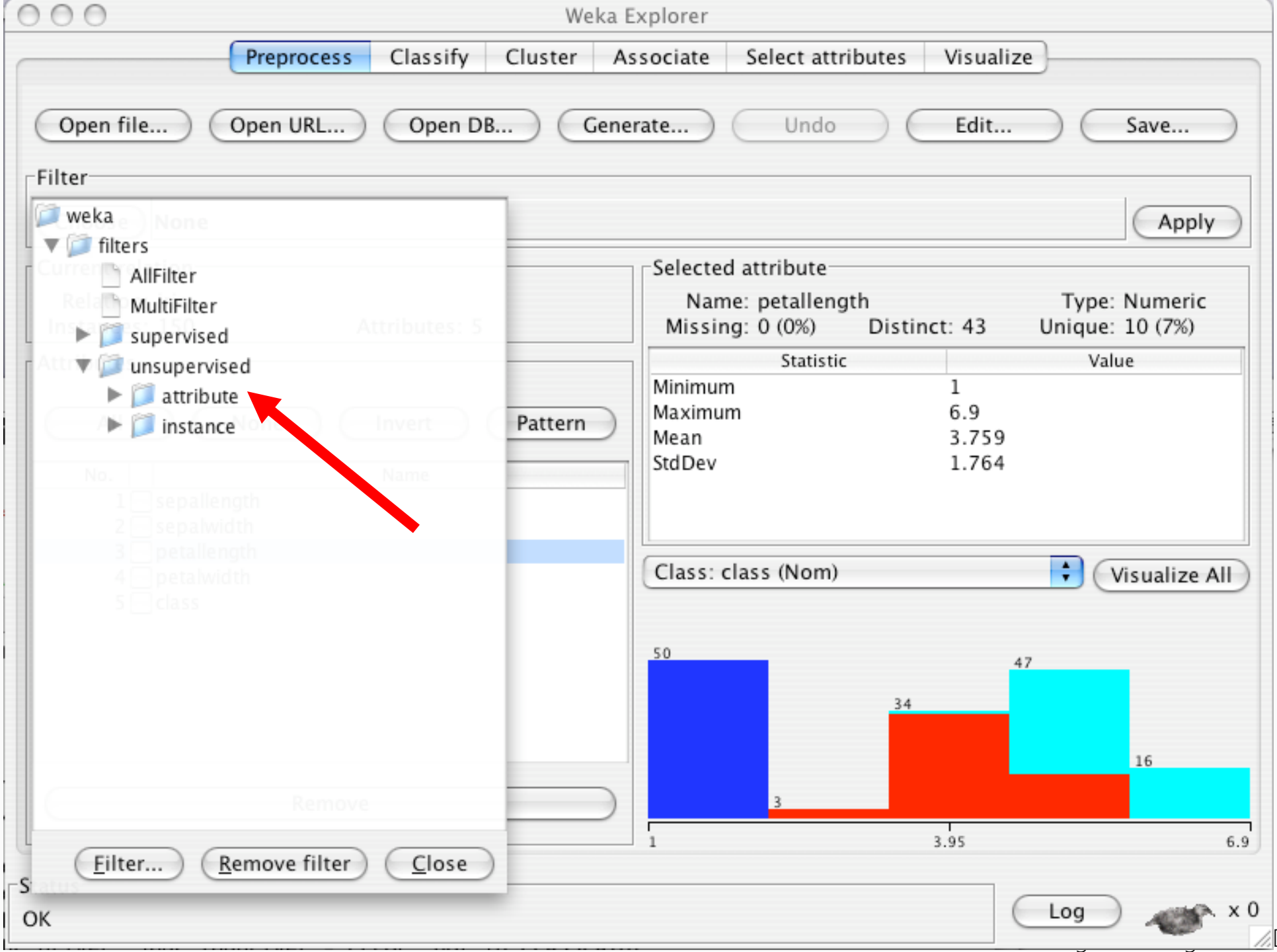

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
None
Apply
Current relation
Relation: iris
Instances: 150
Attributes: 5
Attributes
All
None
Invert
Pattern
No.
Name
1 sepallength
2 sepalwidth
3 petallength
4 petalwidth
5 class
Remove
Selected attribute
Name: class
Type: Nominal
Missing: 0 (0%)
Distinct: 3
Unique: 0 (0%)
Label
Count
Iris-setosa 50
Iris-versicolor 50
Iris-virginica 50
Class: class (Nom)
Visualize All
50
50
50
Status
OK
Log
x 0

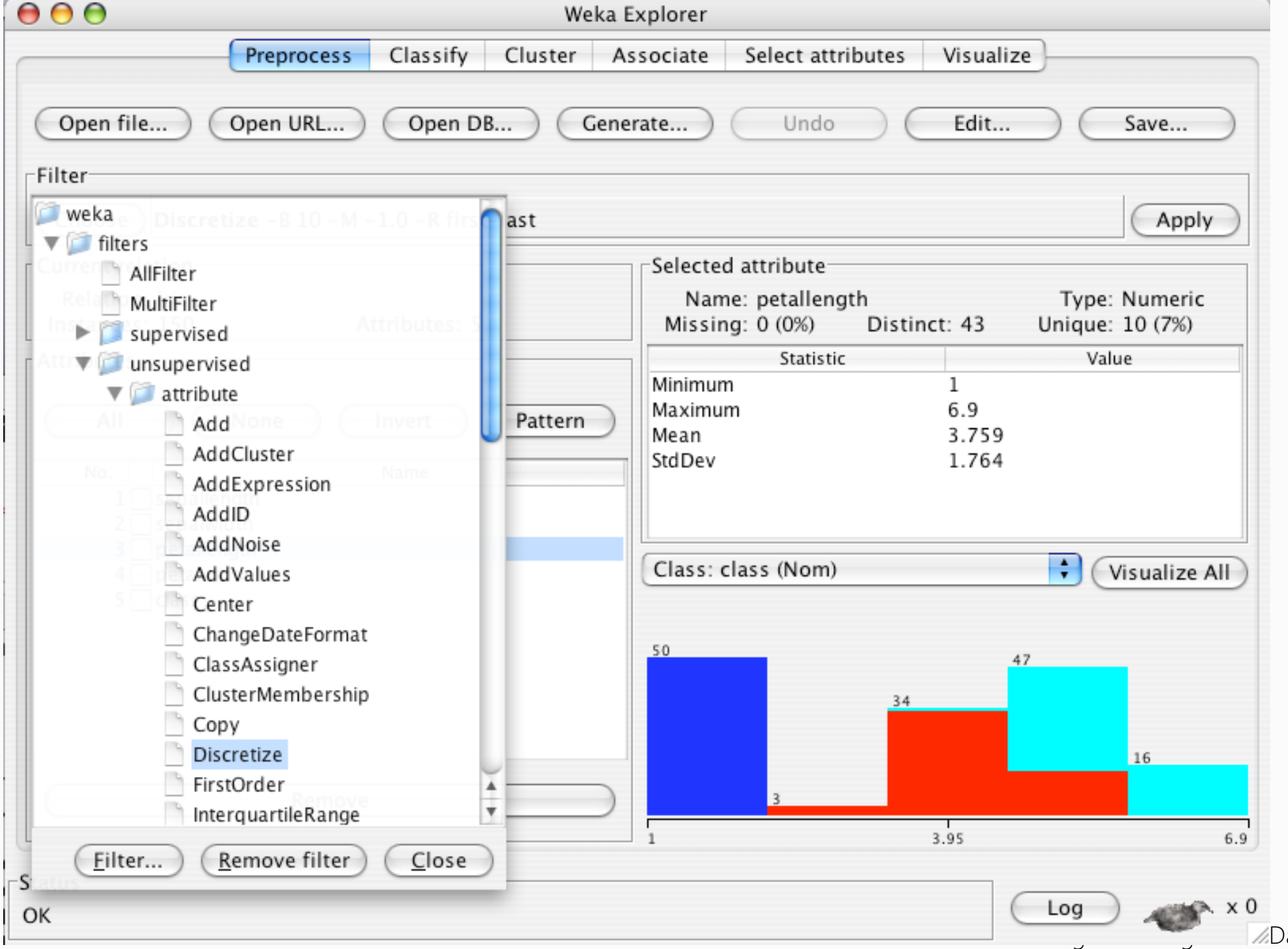

sepallength
34
30
28
25
16
10
7
4.3
6.1
7.9
sepalwidth
51
33
24
16
12
8
4
2
2
3.2
4.4
petallength
50
47
34
16
3
1
3.95
6.9
petalwidth
49
41
29
23
8
0.1
1.3
2.5
class
50
50
50

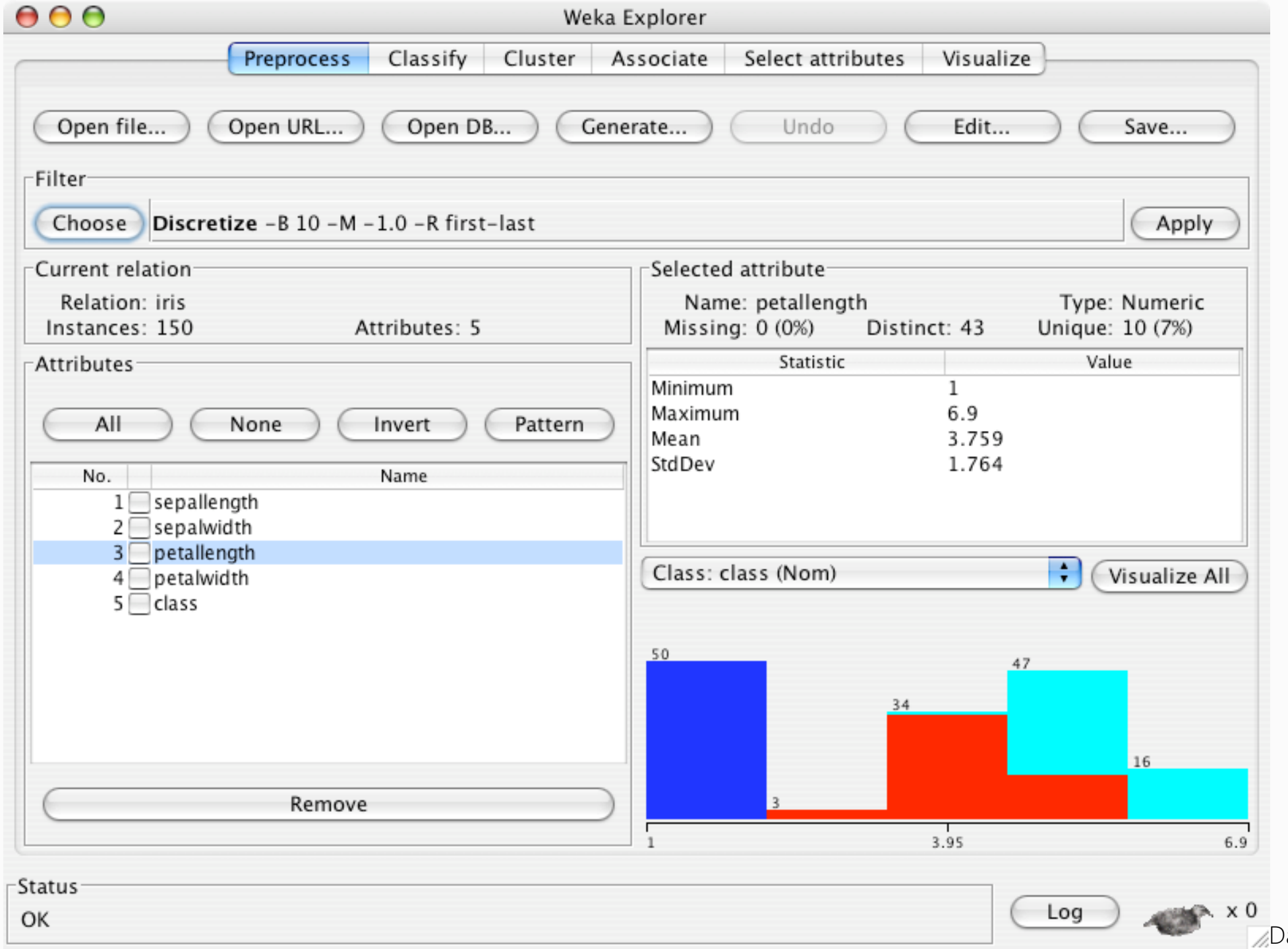

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
None
Apply
Current relation
Relation: iris
Instances: 150
Attributes: 5
Attributes
All
None
Invert
Pattern
No.
Name
1 sepallength
2 sepalwidth
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
50
3
34
47
16
1
3.95
6.9
Status
OK
Log
x 0

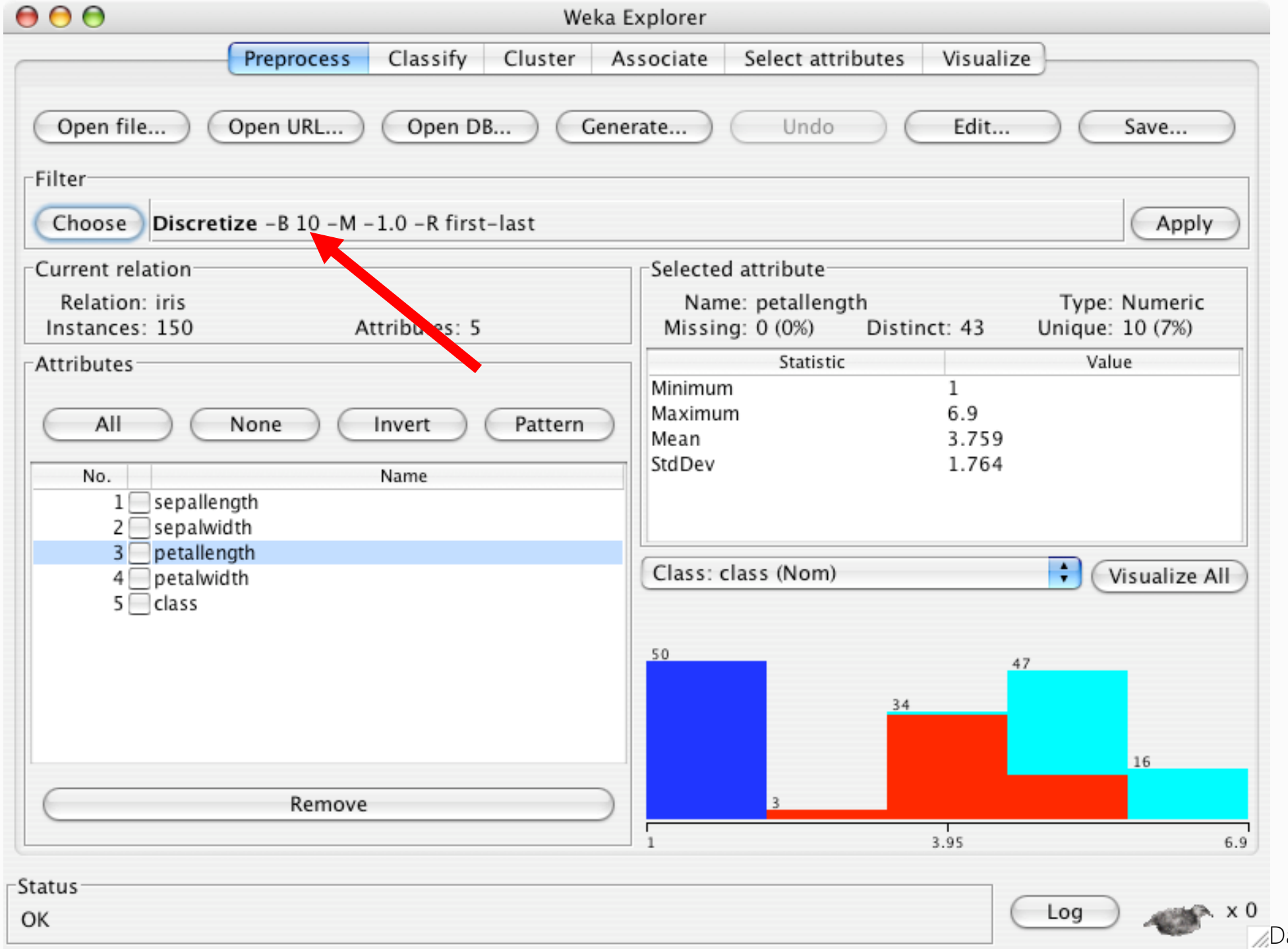

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
None
Apply
Current relation
Relation: iris
Instances: 150
Attributes: 5
Attributes
All
None
Invert
Pattern
No.
Name
1 sepallength
2 sepalwidth
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
50
3
34
47
16
1
3.95
6.9
Status
OK
Log
x 0

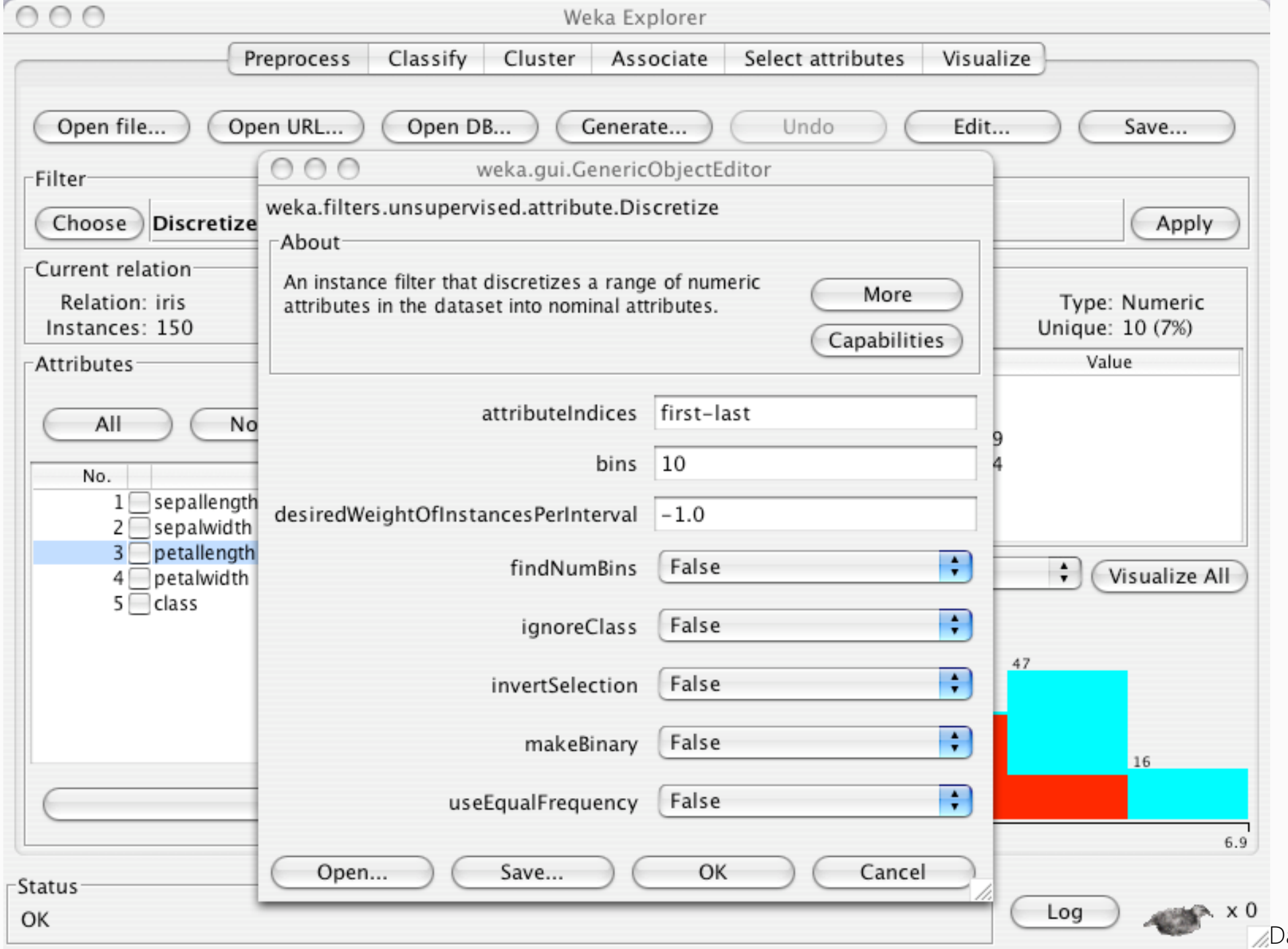

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Apply
weka
filters
AllFilter
MultiFilter
supervised
unsupervised
attribute
instance
Selected attribute
Name: petallength
Type: Numeric
Missing: 0 (0%)
Distinct: 43
Unique: 10 (7%)
Statistic
Value
Minimum
1
Maximum
6.9
Mean
3.759
StdDev
1.764
Class: class (Nom)
Visualize All
50
3
34
47
16
1
3.95
6.9
Filter...
Remove filter
Close
OK
Log
x 0

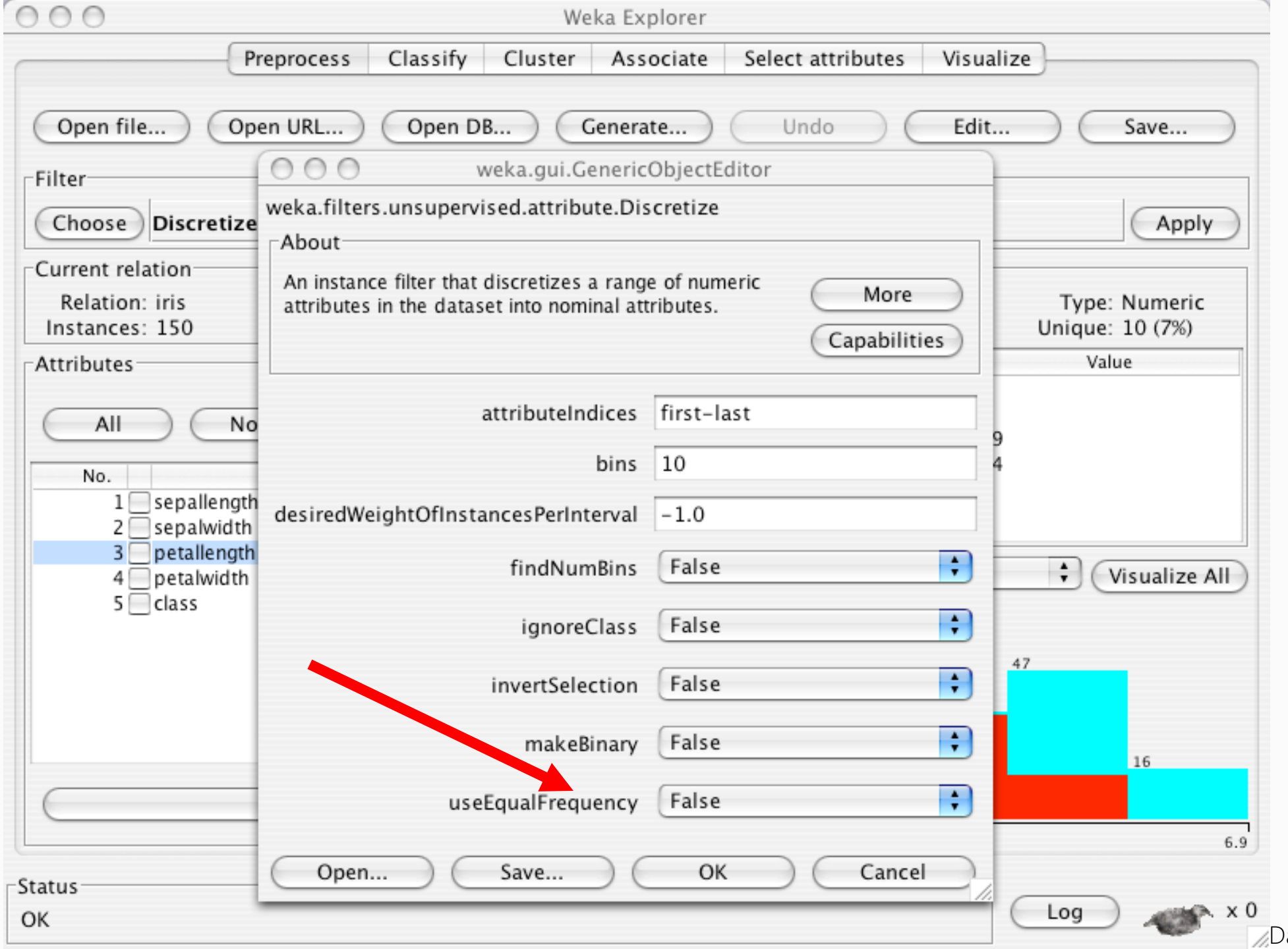

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Apply
weka
filters
AllFilter
MultiFilter
supervised
unsupervised
attribute
instance
Pattern
Selected attribute
Name: petallength
Type: Numeric
Missing: 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
50
3
34
47
16
1
3.95
6.9
Filter...
Remove filter
Close
OK
Log
x 0

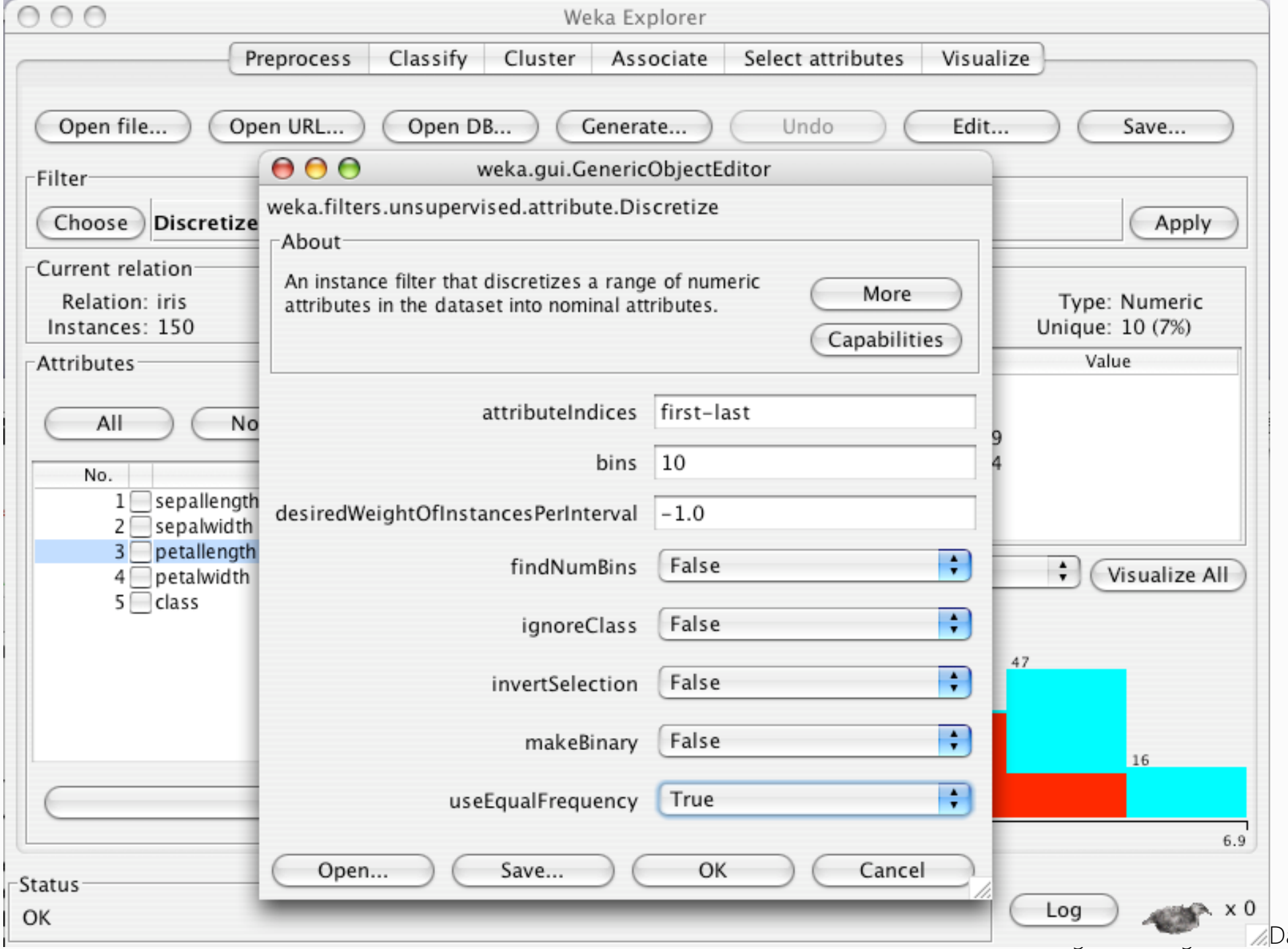

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
weka
filters
AllFilter
MultiFilter
supervised
unsupervised
attribute
Add
AddCluster
AddExpression
AddID
AddNoise
AddValues
Center
ChangeDateFormat
ClassAssigner
ClusterMembership
Copy
Discretize
FirstOrder
InterquartileRange
Apply
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
50
3
34
47
16
1
3.95
6.9
Filter...
Remove filter
Close
OK
Log
x 0

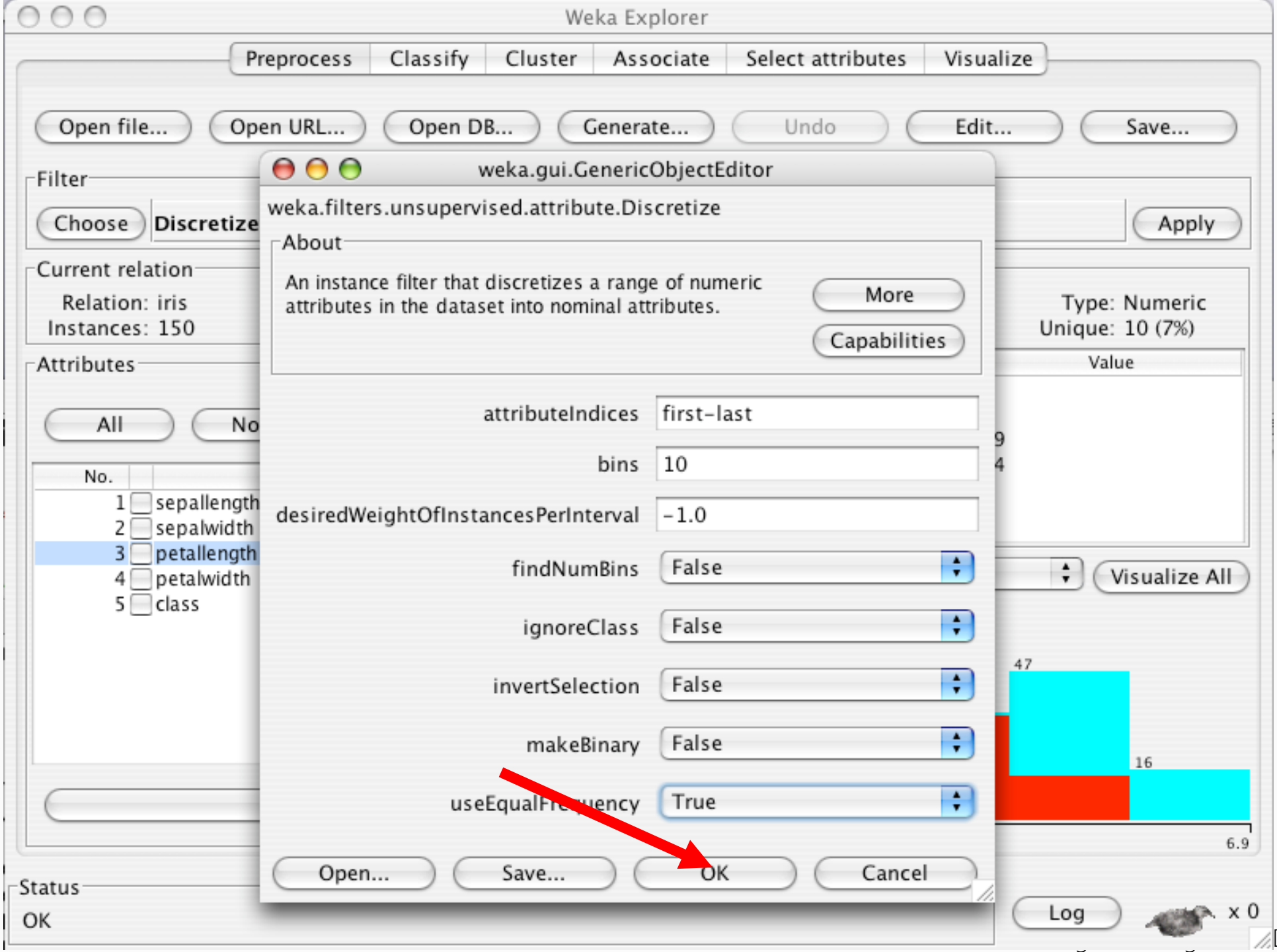

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
Discretize -B 10 -M -1.0 -R first-last
Apply
Current relation
Relation: iris
Instances: 150
Attributes: 5
Attributes
All
None
Invert
Pattern
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
50
3
34
47
16
1
3.95
6.9
Status
OK
Log
x 0

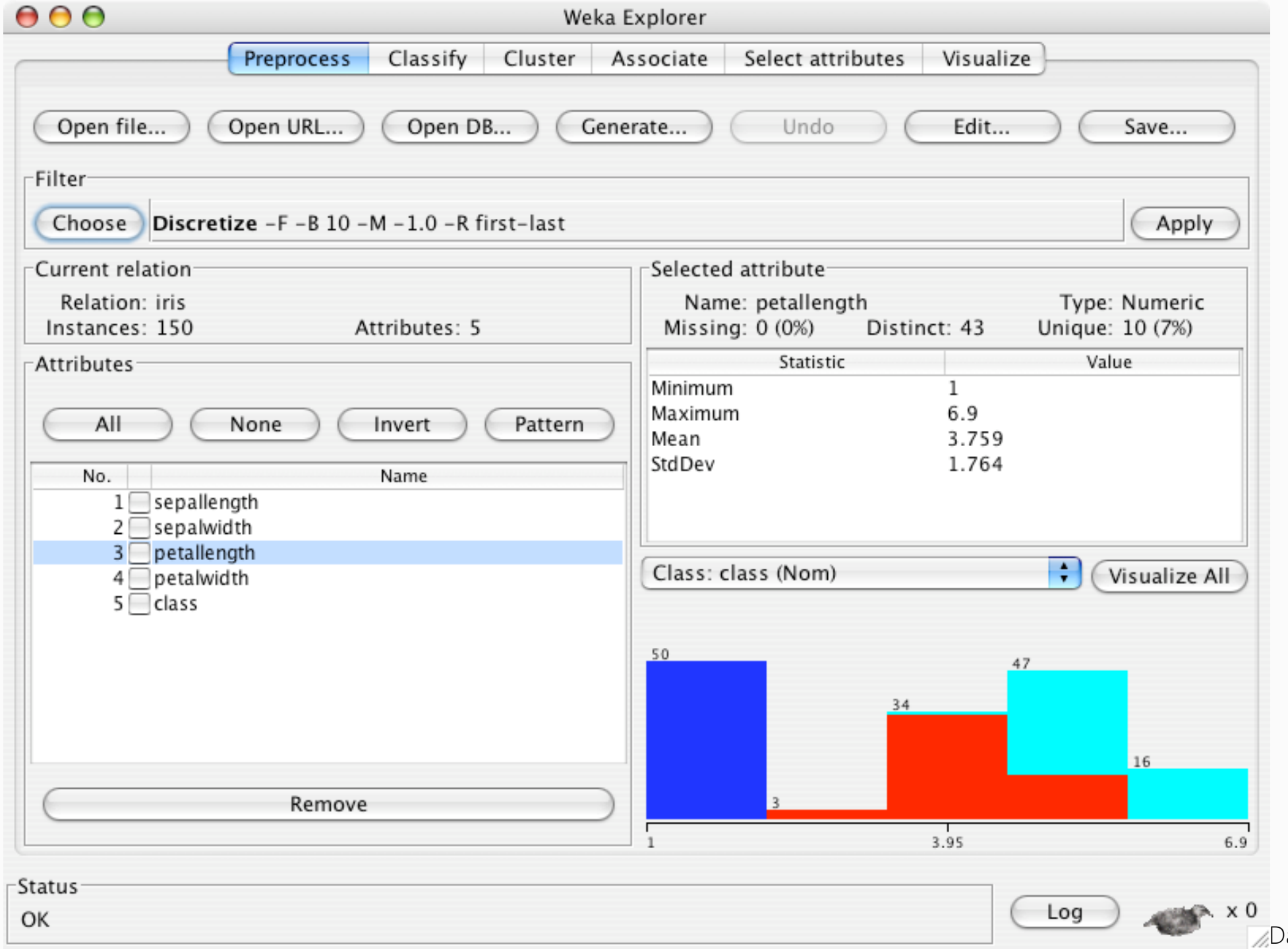

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
Discretize -B 10 -M -1.0 -R first-last
Apply
Current relation
Relation: iris
Instances: 150
Attributes: 5
Attributes
All
None
Invert
Pattern
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
50
3
34
47
16
1
3.95
6.9
Status
OK
Log
x 0

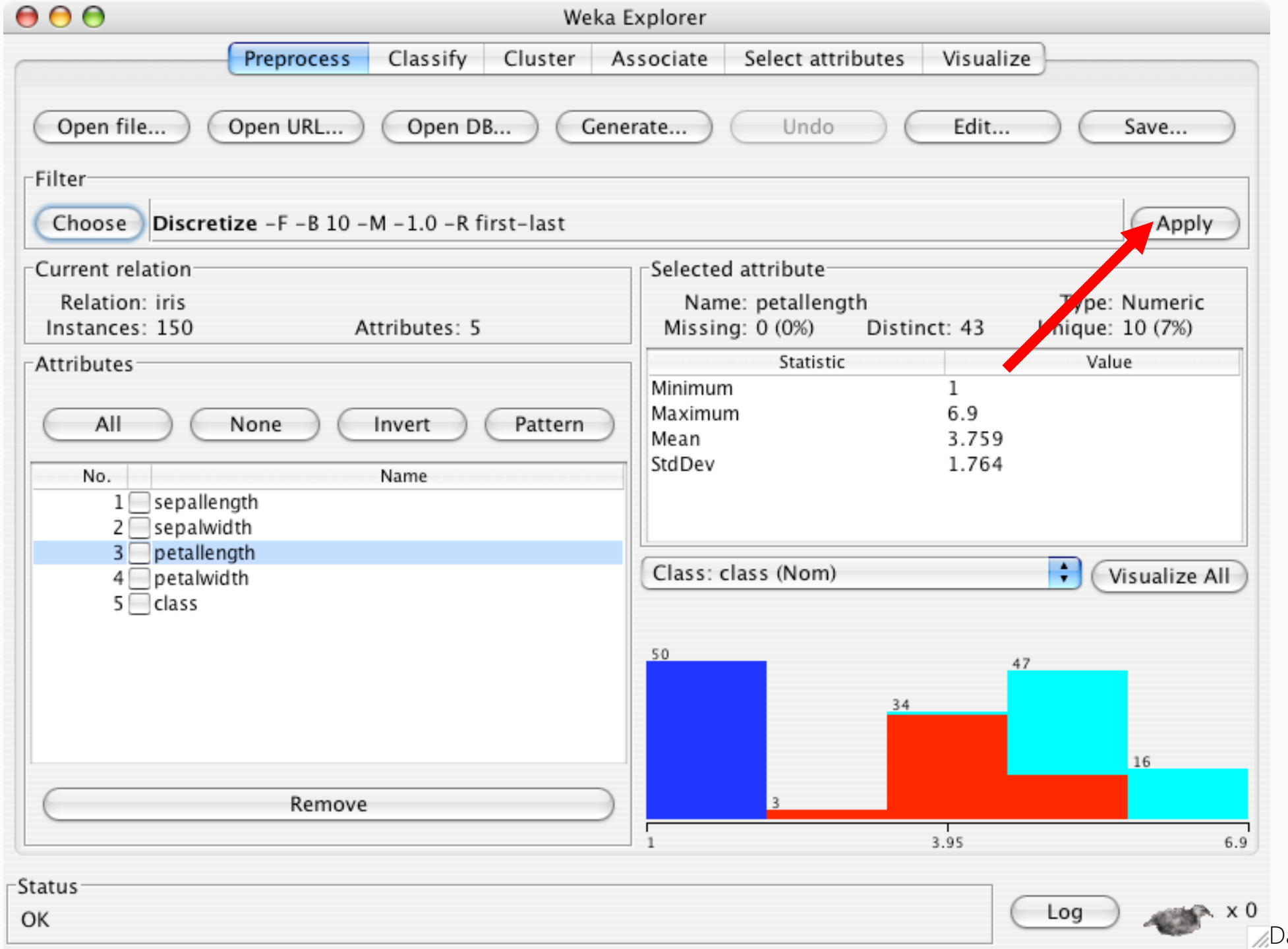

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
Discretize
Apply
Current relation
Relation: iris
Instances: 150
Attributes
All
No
No.
1 sepallength
2 sepalwidth
3 petallength
4 petalwidth
5 class
Type: Numeric
Unique: 10 (7%)
Value
Visualize All
47
16
6.9
Status
OK
Log
x 0
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
attributeIndices
first-last
bins
10
desiredWeightOfInstancesPerInterval
-1.0
findNumBins
False
ignoreClass
False
invertSelection
False
makeBinary
False
useEqualFrequency
False
Open...
Save...
OK
Cancel

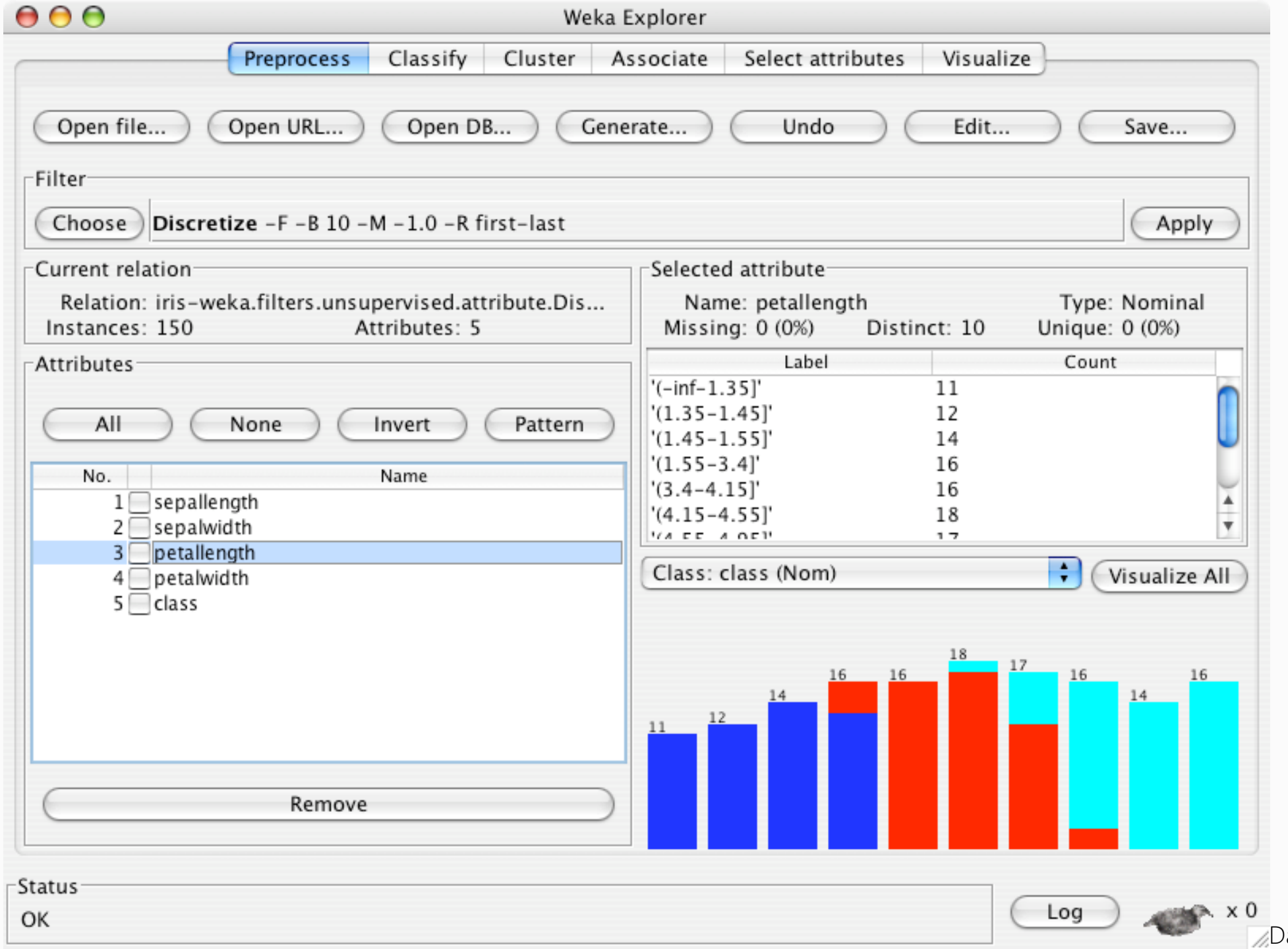

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
Discretize
Apply
Current relation
Relation: iris
Instances: 150
Attributes
All
No
No.
1 sepallength
2 sepalwidth
3 petallength
4 petalwidth
5 class
Type: Numeric
Unique: 10 (7%)
Value
Visualize All
47
16
6.9
Status
OK
Log
x 0
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
attributeIndices
first-last
bins
10
desiredWeightOfInstancesPerInterval
-1.0
findNumBins
False
ignoreClass
False
invertSelection
False
makeBinary
False
useEqualFrequency
False
Open...
Save...
OK
Cancel


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
Discretize
Apply
Current relation
Relation: iris
Instances: 150
Attributes
All
No.
1 sepallength
2 sepalwidth
3 petallength
4 petalwidth
5 class
Type: Numeric
Unique: 10 (7%)
Value
Visualize All
47
16
6.9
Status
OK
Log
x 0
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
attributeIndices
first-last
bins
10
desiredWeightOfInstancesPerInterval
-1.0
findNumBins
False
ignoreClass
False
invertSelection
False
makeBinary
False
useEqualFrequency
True
Open...
Save...
OK
Cancel


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
Discretize
Apply
Current relation
Relation: iris
Instances: 150
Attributes
All
No.
1 sepallength
2 sepalwidth
3 petallength
4 petalwidth
5 class
Type: Numeric
Unique: 10 (7%)
Value
Visualize All
47
16
6.9
Status
OK
Log
x 0
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
attributeIndices
first-last
bins
10
desiredWeightOfInstancesPerInterval
-1.0
findNumBins
False
ignoreClass
False
invertSelection
False
makeBinary
False
True
Open...
Save...
OK
Cancel


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 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
50
3
34
47
16
1
3.95
6.9
Status
OK
Log
x 0


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 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
50
3
34
47
16
1
3.95
6.9
Status
OK
Log
x 0


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.Dis...
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: 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
Class: class (Nom)
Visualize All
11
12
14
16
16
18
17
16
14
16
Status
OK
Log
x 0

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….

Nov 9th , 2015