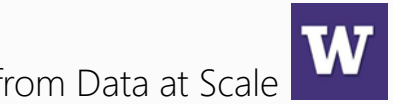
Data Science

Deriving Knowledge from Data at Scale

Winson Taam

Oct 19th, 2015



Lecture 3 Outline, Oct 19th

- Opening Discussion
- Forecasting, continued (2/2)
- Break
- Introducing Weka
- Break
- Decision Trees
- Hands On, Decision Tree in Weka

Quick Review, and a couple additional terms…

Hypothesis Space

- Consists of all models a particular classification method can produce;

Overfitting

- Overlearning training data at expense of generalization, accuracy on unseen data;

Bias Problem

- Hypothesis space of a particular method does not include sufficient hypotheses;

Variance Problem

- The hypothesis space made available is too large for the training data, and as a result the selected hypothesis may not be accurate on unseen data;

Goal

- Model Complexity (-)
- Variance (-)
- Prediction Accuracy (+)

Best Machine Learning and Data Mining Tools for Free.

- RapidMiner
- Weka – *explorer…*
- PSPP
- KNIME – *experimentation…*
- Orange
- Apache Mahout
- jHepWork
- Rattle
- R or Revolution R
- Python

Get proficient in at least two (2) tools…

WEKA http://www.cs.waikato.ac.nz/ml/weka/, *we'll use this in class…*

Machine Learning Group at the University of Waikato

Project **Software** Book Publications People Related

Weka 3: Data Mining Software in Java

Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes.

Found only on the islands of New Zealand, the Weka is a flightless bird with an inquisitive nature. The name is pronounced like **this**, and the bird sounds like **this**.

Weka is open source software issued under the **GNU General Public License**.

Yes, it is possible to apply Weka to **big data**!

Data Mining with Weka is a 5 week MOOC, which was held first in late 2013. Check out the **MOOC site** for video lectures and details on how to enrol into this course and a new, advanced Weka course.

Getting started

- **Requirements**
- **Download**
- **Documentation**
- **FAQ**
- **Getting Help**

Further information

- **Citing Weka**
- **Datasets**
- **Related Projects**
- **Miscellaneous Code**
- **Other Literature**

Developers

- **Development**
- **History**
- **Subversion**
- **Contributors**

Optional Reading

Open Source Data Mining Software Evaluation

Informatics Research and Development Unit
Public Health Informatics & Technology Program Office
Office of Surveillance, Epidemiology, and Laboratory Services
US Centers for Disease Control and Prevention

2010 Report:

Open Source Data Mining Software Evaluation

Background

This evaluation focused on general purpose open source data mining software products. Data mining tools which were developed for a specific domain (e.g. text mining, image mining, microarray data mining, etc.) were not included in this evaluation. The data mining software products in this evaluation include: RapidMiner, Weka, Orange, Rattle, and Knime.

Given the broad potential audience for these products in the domain of public health, only those functions / features available through a graphical user interface (GUI) were evaluated. Specifically, any functions of these tools which required scripting or programming were not considered. For this reason, an additional popular open source data mining tool, known as R, was not included in this evaluation. Please note that, R provides many powerful statistical analyses and data mining functions.

Results

This summary report presents our finding from three perspectives: *general information, system features, and data mining functionality*.

Table 1. General information of selected data mining software products

	RapidMiner (YALE)	Weka	Orange	Rattle	KNIME
Edition/ Version	Community edition, version 5.0	version 3.6.2	version 2.0	version 2.5.21	Desktop edition, version 2.1.2
Company/	Rapid-I	University of	University	Togaware	KNIME.com

Reference

Rexer Annual Analytics Surveys

A Few Useful Things to Know about Machine Learning

Pedro Domingos
Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195-2350, U.S.A.
pedrod@cs.washington.edu

ABSTRACT

Machine learning algorithms can figure out how to perform important tasks by generalizing from examples. This is often feasible and cost-effective where manual programming is not. As more data becomes available, more ambitious problems can be tackled. As a result, machine learning is widely used in computer science and other fields. However, developing successful machine learning applications requires a substantial amount of "black art" that is hard to find in textbooks. This article summarizes twelve key lessons that machine learning researchers and practitioners have learned. These include pitfalls to avoid, important issues to focus on, and answers to common questions.

1. INTRODUCTION

Machine learning systems automatically learn programs from data. This is often a very attractive alternative to manually

correct output y_t for future examples $\mathbf{x_t}$ (e.g., whether the spam filter correctly classifies previously unseen emails as spam or not spam).

2. LEARNING = REPRESENTATION + EVALUATION + OPTIMIZATION

Suppose you have an application that you think machine learning might be good for. The first problem facing you is the bewildering variety of learning algorithms available. Which one to use? There are literally thousands available, and hundreds more are published each year. The key to not getting lost in this huge space is to realize that it consists of combinations of just three components. The components are:

Representation. A classifier must be represented in some formal language that the computer can handle. Con-

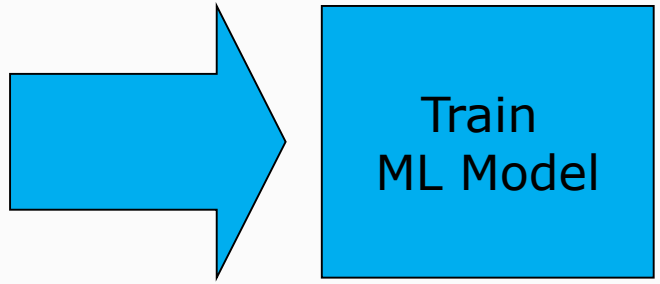
Machine Learning: Setting

gender	age	smoker	eye color	lung cancer
male	19	yes	green	no
female	44	yes	gray	yes
male	49	yes	blue	yes
male	12	no	brown	no
female	37	no	brown	no
female	60	no	brown	yes
male	44	no	blue	no
female	27	yes	brown	no
female	51	yes	green	yes
female	81	yes	gray	no
male	22	yes	brown	no
male	29	no	blue	no

male	77	yes	gray	?
male	19	yes	green	?
female	44	no	gray	?

Machine Learning: Setting

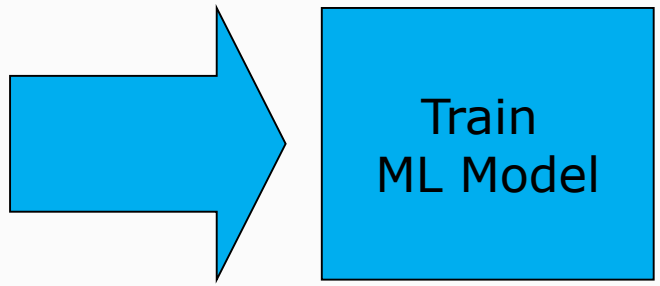
gender	age	smoker	eye color	lung cancer
male	19	yes	green	no
female	44	yes	gray	yes
male	49	yes	blue	yes
male	12	no	brown	no
female	37	no	brown	no
female	60	no	brown	yes
male	44	no	blue	no
female	27	yes	brown	no
female	51	yes	green	yes
female	81	yes	gray	no
male	22	yes	brown	no
male	29	no	blue	no



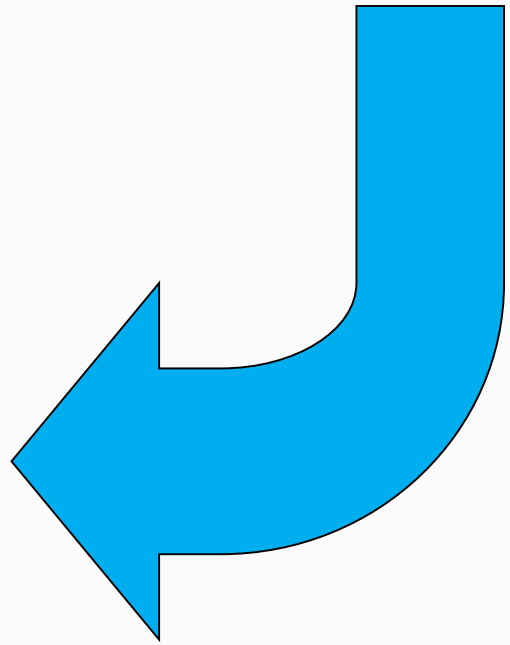
male	77	yes	gray	?
male	19	yes	green	?
female	44	no	gray	?

Machine Learning: Setting

gender	age	smoker	eye color	lung cancer
male	19	yes	green	no
female	44	yes	gray	yes
male	49	yes	blue	yes
male	12	no	brown	no
female	37	no	brown	no
female	60	no	brown	yes
male	44	no	blue	no
female	27	yes	brown	no
female	51	yes	green	yes
female	81	yes	gray	no
male	22	yes	brown	no
male	29	no	blue	no



male	77	yes	gray	yes
male	19	yes	green	no
female	44	no	gray	no



Overfitting has many faces…

Always measure performance of your model on out-of-sample data.

- At least one test where you train on some of your data and test on the rest;
- You should do several, where you split the data differently each time;

If your data is placed in time, say weekly retail sales, do a test where you train on all weeks except one, and test on that week, do this for all weeks in data.

Author focuses on a particular way of analyzing errors of a model, called bias-variance decomposition. We have a lecture dedicated to evaluation metrics for models.

"Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful." – George E.P. Box

Intuition fails in high dimensions…

You have a model, with some input fields and an objective field, and it is not performing as well as you'd like.

- First intuition might be to add more input fields. If the model can do as well as it does with the input fields, surely more information is better, right?
- This often doesn't work. In fact, if you add input fields that are not useful, or redundant to information already in other input fields, you may very well get a model that performs worse than the original.

Curse of Dimensionality

- The more input fields, the more likely the model will see some relationship between one and the objective that isn't real, the product of random noise.
- You can combat this problem in your own modeling efforts by selecting as inputs only the fields you know are relevant to predicting the objective

Theoretical guarantees are not what they seem…

Ignore the fancy mathematics on error bounds

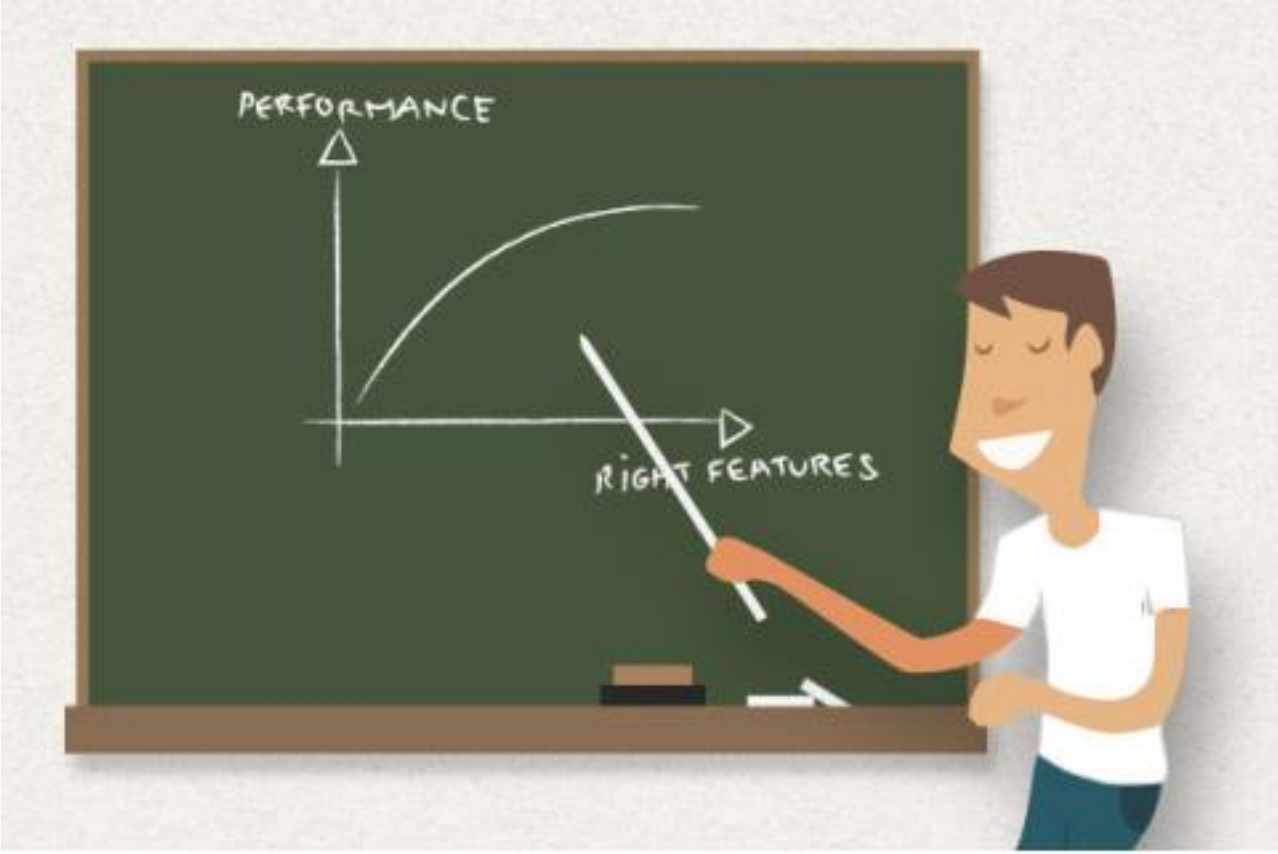
Ignore article asserting one class of algorithms is superior to another class of algorithms for a particular problem. All such biases for and against algorithms are at best misleading

Come to the process with no preconceptions and you are likely to find the best answer.

- The only certain way (that we know of now) to know if an algorithm will model your data well is to try it out.
- You may know some data-specific things that may help you select a learning algorithm before trying it out, you may be surprised by the results anyway

Feature engineering is the key…

"easily the most important factor" in determining the success of a machine learning project – *and he's right…*



Feature engineering is the key…

Construct a model that can predict for any two cities whether the distance is drivable or not.

CITY 1 LAT.	CITY 1 LNG.	CITY 2 LAT.	CITY 2 LNG.	DRIVABLE?
123.24	46.71	121.33	47.34	Yes
123.24	56.91	121.33	55.23	Yes
123.24	46.71	121.33	55.34	No
123.24	46.71	130.99	47.34	No

Probably not going to happen…

Feature engineering is the key…

Even if the machine doesn't have knowledge of how longitudes and latitudes work, you do. So why don't you do it?

Feature engineering, when you use your knowledge about the data to create fields that make machine learning algorithms work better.

How does one engineer a good feature? Rule of thumb is to try to design features where the likelihood of a certain class goes up monotonically with the value of the field.

Great things happen in machine learning when human and machine work together, combining a person's knowledge of how to create relevant features from the data with the machine's talent for optimization..

DISTANCE (MI.)	DRIVABLE?
14	Yes
28	Yes
705	No
2432	No

More data beats a cleverer algorithm…

More data wins. There's increasingly good evidence that, in a lot of problems, very simple machine learning techniques can be levered into incredibly powerful classifiers with the addition of loads of data.

Once you've defined your input fields, there's only so much analytic gymnastics you can do. Computer algorithms trying to learn models have only a relatively few tricks they can do efficiently, and many of them are not so very different. Performance differences between algorithms are typically not large. Thus, if you want better classifiers:

1. Engineer better features
2. Get your hands on more high-quality data

Learn many models, not just one…

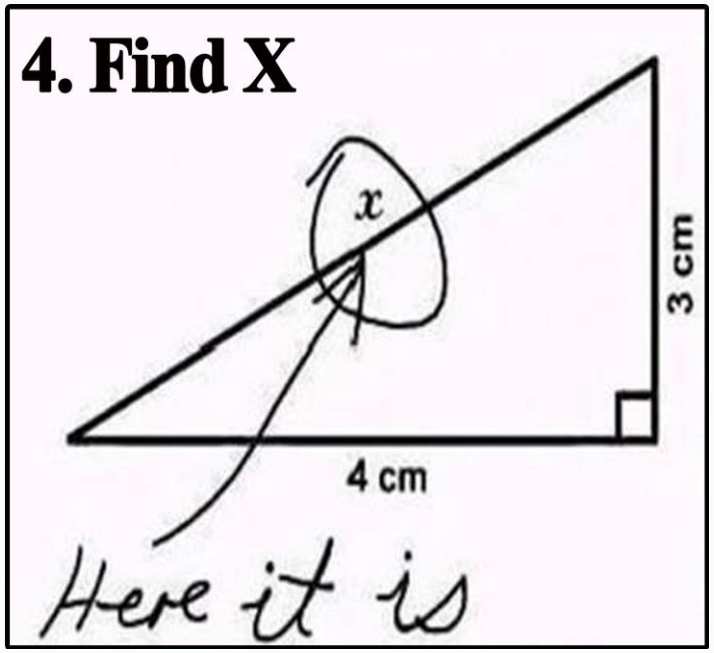
The power of ensembles, we are going discuss an instance of this in next weeks Machine Learning lecture.

One can often make a more powerful model by learning multiple classifiers over different random subsets of the data. Downside is that some interpretability is lost; instead of a single sequence of questions and answers to arrive at a prediction, you now have a sequence for each model, and the models vote for the final prediction.

If your application is very performance sensitive, the loss in interpretability might be worth the increase in power.

Simplicity does not imply accuracy…

Occam's Razor, in original fancy Latin it reads something like, "Plurality is not to be posited without necessity". In layman's terms, if you have two explanations for something, you should generally prefer the simpler one.



If we have two models that fit the data equally well, give preference to the simpler model because it is smaller, faster to fit, and more interpretable, but not necessarily because it will lead to better performance; the only way to know that is to evaluate your model on test data.

Correlation does not imply causation…

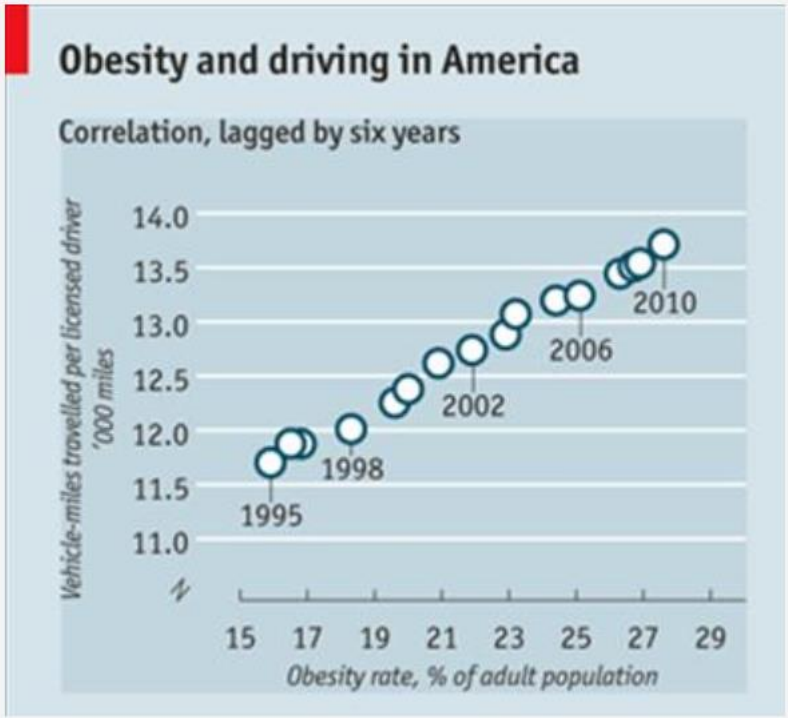
Old adage in statistics, the point of this common saying is that modeling **observational data can only show us that two variables are related, but it cannot tell us the "why"**.

Example from **Freakonomics**, data from public school test scores showed that children who lived in homes with a high number of books tended to have higher standardized test scores than those with a lower number of books in the house. The mayor of Chicago, doubtless while screaming "Science!" at all of his advisors, proposed a program to send free books to the homes of children, which would definitely raise their test scores, right?

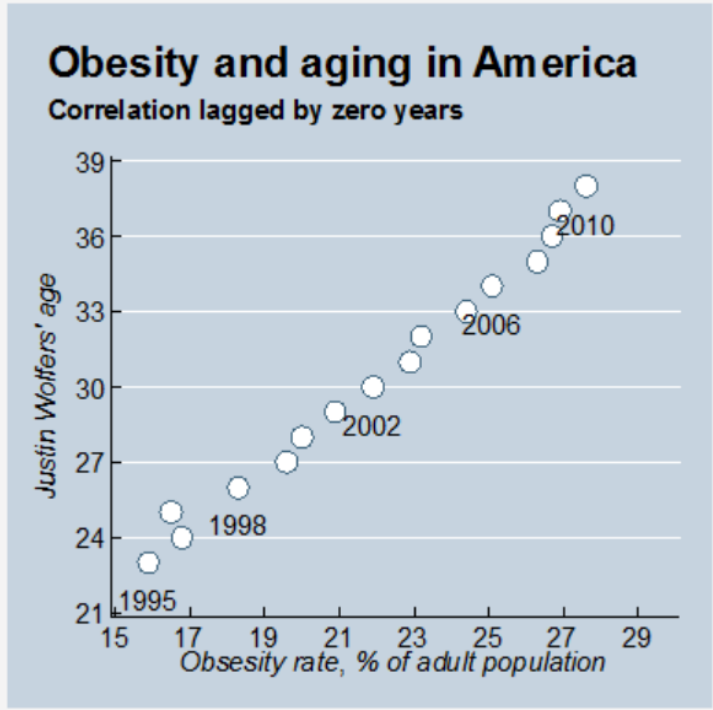
What Drives Obesity? An Economist Takedown of *The Economist*

JUSTIN WOLFERS
06/16/2011 | 11:29 am

Is higher obesity due to the rise in driving? Perhaps. It's an intriguing hypothesis. But our friends at *The Economist* should know better than to **report** nonsensical correlations. Here's the evidence they cite (drawn from **this** entirely unconvincing research paper published in *Transport Policy*):



Looks impressive, right? (Well, apart from putting the explanatory variable on the vertical axis.) But before concluding that there's anything here, let's try a different variable, instead—my age:



Unlike the authors of the original paper, I didn't even need to fiddle with the lag structure to get such a good fit, nor test alternative definitions of the variable. In fact, my variable fits even better than vehicle miles traveled.

Okay, I'm not arguing that my aging is causing higher obesity. Rather, when you see a variable that follows a simple trend, almost any other trending variable will fit it: miles driven, my age, the Canadian population, total deaths, food prices, cumulative rainfall, whatever.

Sure, *The Economist* offered the usual caveat that "correlation does not equal causation." But this is so completely unconvincing as to warrant a different warning: "Not persuasive enough that you should bother reading this article." I'm not saying the relationship doesn't exist, simply that it makes more sense to highlight more **persuasive research** on this question.

P.S.: How cool is Stata's "Economist" scheme? It lets stat-nerds like me replicate magazine-quality art.

Lecture 3 Outline, October 19th

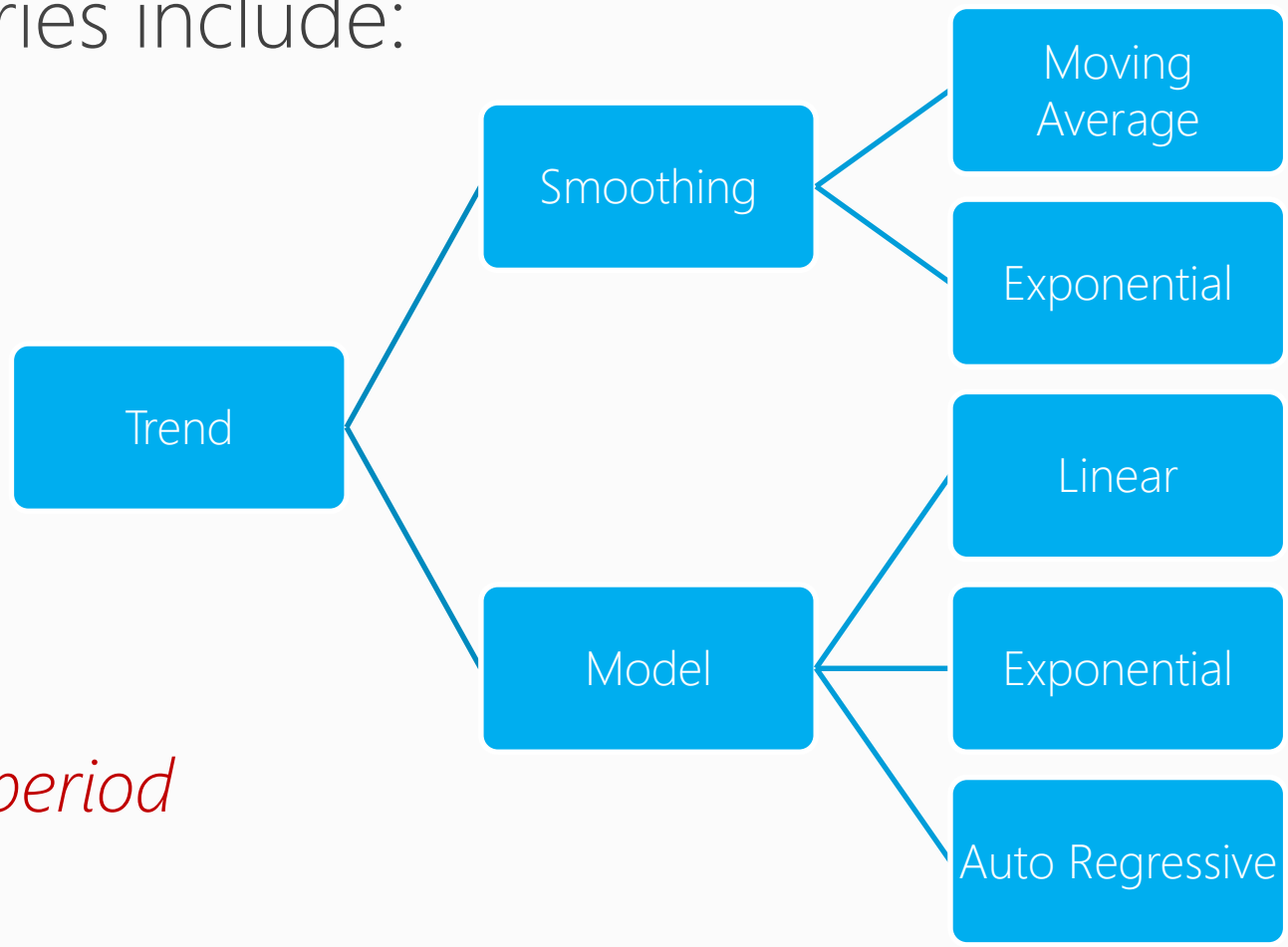
- Opening Discussion
- Forecasting, continued (2/2)
- Break
- Introducing Weka
- Decision Trees
- Hands On, Decision Tree in Weka

Time Series Forecasting

Q. What is a time series?

Q. Elements of a time series include:

1. Trend
2. Seasonality
3. Cyclical
4. Irregular intervention
5. Noise



Q. Seasonal or Cyclical?

Fixed and known period

Rise and fall, not a fixed period

Remark: Try "stl()" function in R

	A	B	C	D	E	F
1	date	open	high	low	close	volume
2	20020514	29.01	29.29	28.7	29	422606
3	20020513	28.81	29.15	28.71	28.93	265548
4	20020510	29.3	29.49	28.38	28.69	497579
5	20020509	29.26	29.78	28.82	29.01	342824
6	20020508	28.5	29.8	28.41	29.7	559890
7	20020507	29.5	29.65	27.85	28.21	716809
8	20020506	29.4	29.74	29.01	29.01	196019
9	20020503	29.35	29.7	28.83	29.23	520990
10	20020502	30.06	30.38	29.22	29.42	456067
11	20020430	30.25	30.65	29.6	30.4	499916
12	20020429	29.59	30.54	29.47	30.5	350735
13	20020426	30.4	30.67	29.65	29.65	677727
14	20020425	30.55	31.19	29.11	30.61	1358092
15	20020424	31	31.5	30.08	30.23	960344
16	20020423	32.3	32.48	31.21	31.22	736359
17	20020422	32.95	32.95	31.59	32.05	469032
18	20020419	33.67	33.67	32.26	32.9	657852
19	20020418	33.8	34.48	32.45	33.5	505319
20	20020417	33.68	34.28	33.01	33.88	383194
21	20020416	33.67	33.73	33.33	33.38	350841
22	20020415	33.9	33.9	33.22	33.7	237563
23	20020412	33.25	34.25	33.25	33.66	351551
24	20020411	33.89	34.15	33.21	33.33	353272
25	20020410	33.06	34.14	33.06	33.8	337708
26	20020409	33.57	33.57	33	33.24	489001
27	20020408	34.21	34.31	31.8	33.33	691079
28	20020405	34.1	34.45	33.84	34.13	469672
29	20020404	34.05	34.36	33.67	34.25	376338
30	20020403	34.35	34.57	33.71	34.2	374836
31	20020402	34.7	34.86	33.89	34.3	449014
32	20020328	34.25	34.7	33.86	34.38	400977
33	20020327	33.61	34.38	33.61	33.92	276849
34	20020326	33.71	34.63	33.55	33.6	466043
35	20020325	33.76	34.71	33.62	34.08	416687
36	20020322	34.12	34.5	33.8	34.05	431296
37	20020319	35.09	35.1	34.01	34.6	384202
38	20020318	34.92	35.38	34.51	34.9	523978

	A	B
1	close	
2	29	
3	28.93	
4	28.69	
5	29.01	
6	29.7	
7	28.21	
8	29.01	
9	29.23	
10	29.42	
11	30.4	
12	30.5	
13	29.65	
14	30.61	
15	30.23	
16	31.22	
17	32.05	
18	32.9	
19	33.5	
20	33.88	
21	33.38	
22	33.7	
23	33.66	
24	33.33	
25	33.8	
26	33.24	
27	33.33	
28	34.13	
29	34.25	
30	34.2	
31	34.3	
32	34.38	
33	33.92	
34	33.6	
35	34.08	

as a .csv

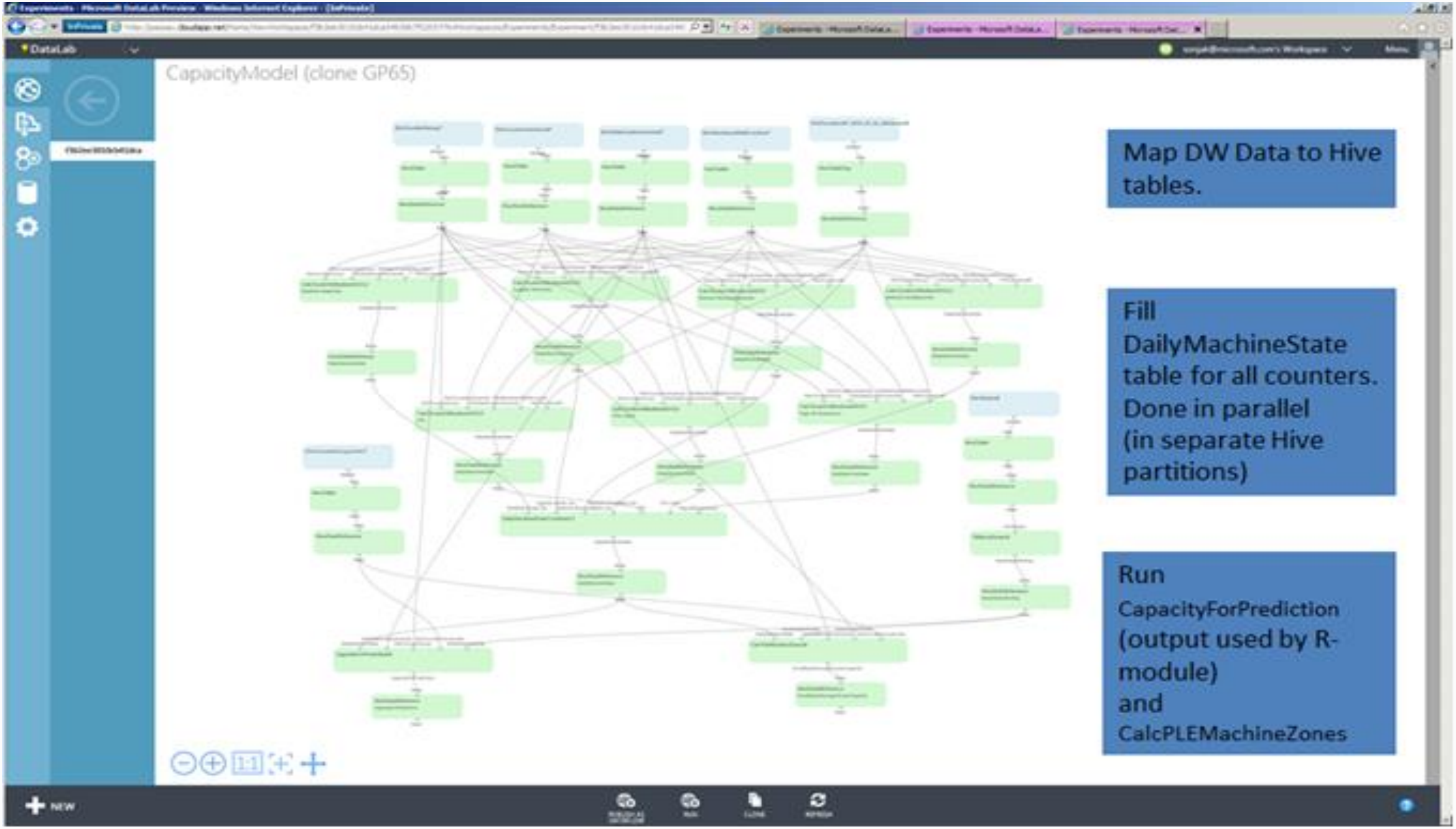
Predictive Modeling: Examples

Rack Capacity Planning

- Time series of cpu, network, disk, i/o/sec for each tenant
- Predict whether we can add additional database(s)
- Could seasonality information be of value here?

Health of Database

- $\boldsymbol{x}_i$ = state of machine over the past 5 minutes
  - Average, min, max of CPU usage, number of processes on box, number of (r/w)/sec
  - Difference in the average state between the past 5 mins. and the past 1 hour
- $\boldsymbol{y}_i$ = # of outage minutes (= 0 for no outage, > 0 for outage)
- Predict whether an outage will occur


DataLab
CapacityModel (clone GP65)
Map DW Data to Hive tables.
Fill DailyMachineState table for all counters. Done in parallel (in separate Hive partitions)
Run CapacityForPrediction (output used by R-module) and CalcPLEMachineZones
NEW

Automatic Forecasting

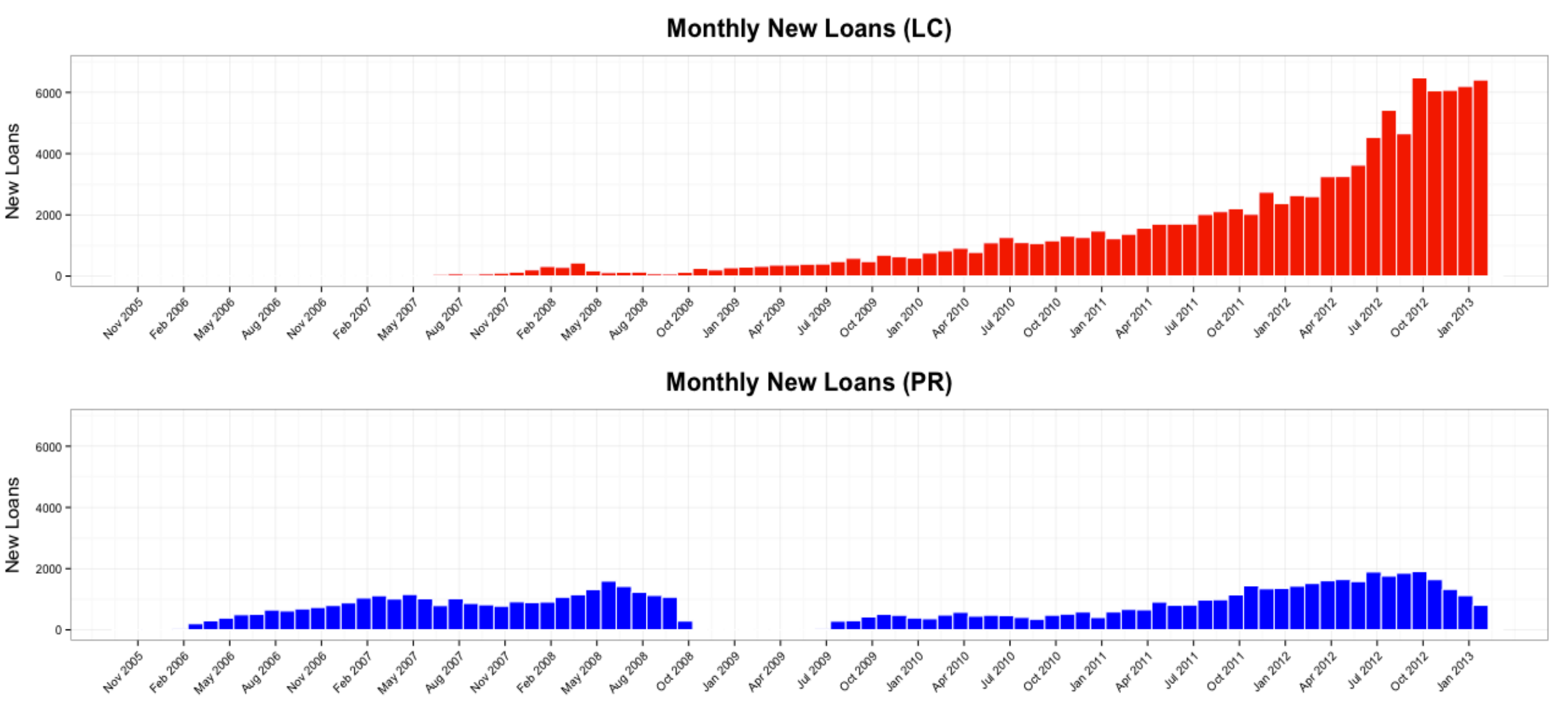
Why use an automatic procedure?

- Most users are not very expert at fitting time series models.
- Most experts cannot beat the best automatic algorithms.
- Many businesses and industries need thousands of forecasts every week/month.
- Some multivariate forecasting methods depend on many univariate forecasts.

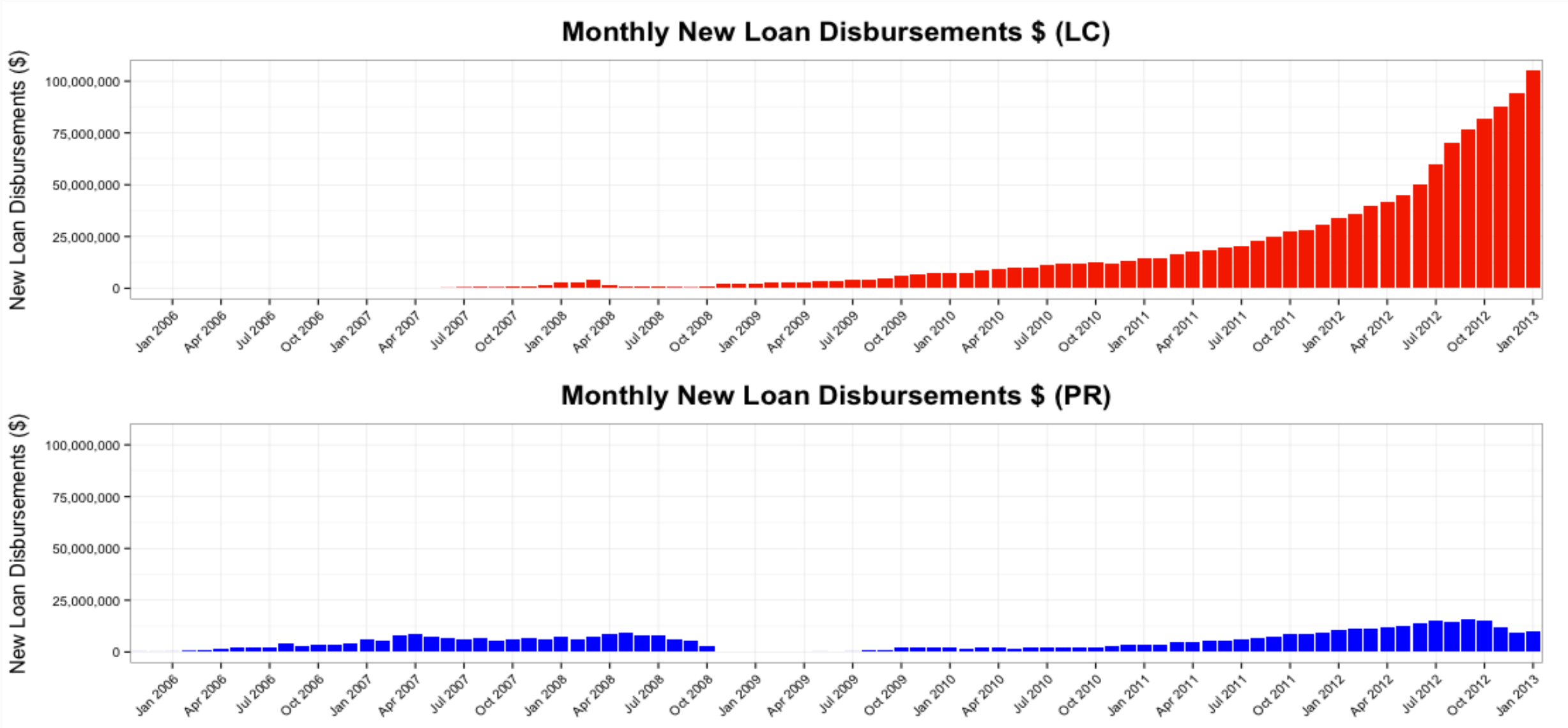
Example: P2P Lending

- Peer-to-Peer (P2P) lending networks facilitate the matching between borrowers who want to borrow money, and peer lenders who are willing to loan them money at a premium.
- Because 'peers' are matched directly, borrowers face lower interest rates than those offered by banks or credit cards and lenders can earn higher returns than those offered by banks or bonds.
- These networks are highly transparent, releasing a plethora of data about the potential borrower. The two largest networks, Lending Club and Prosper continuously release their lending data to the public:
- LendingClub Data Downloads
- Prosper Data Downloads

Example: P2P Lending



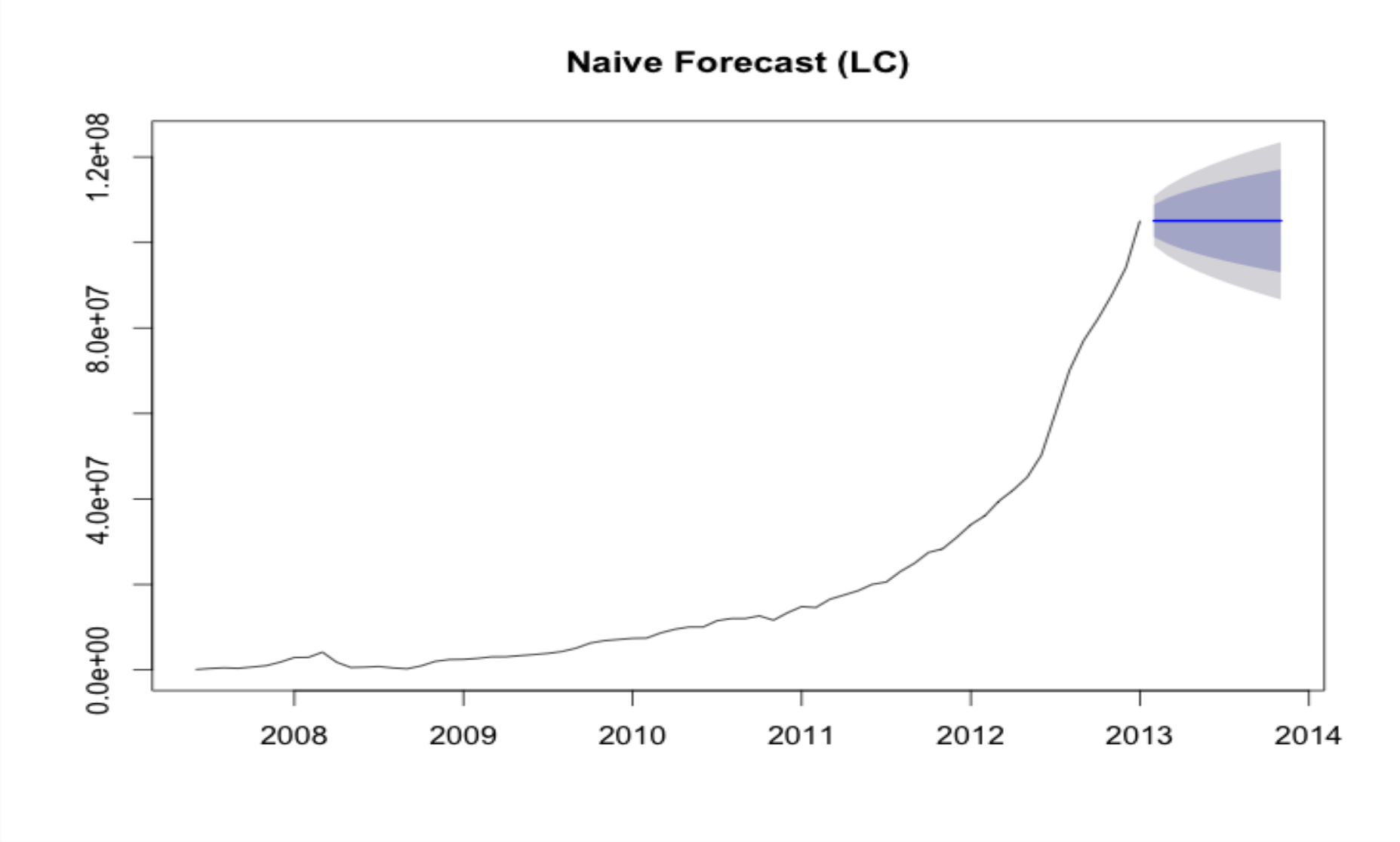
Example: P2P Lending



Example: P2P Lending

How many Loans will Lending Club and Prosper make next month? Next 3 months? Next year?

Naïve Forecast – this period's value is equal to last periods value (ARIMA(0,1,0)).

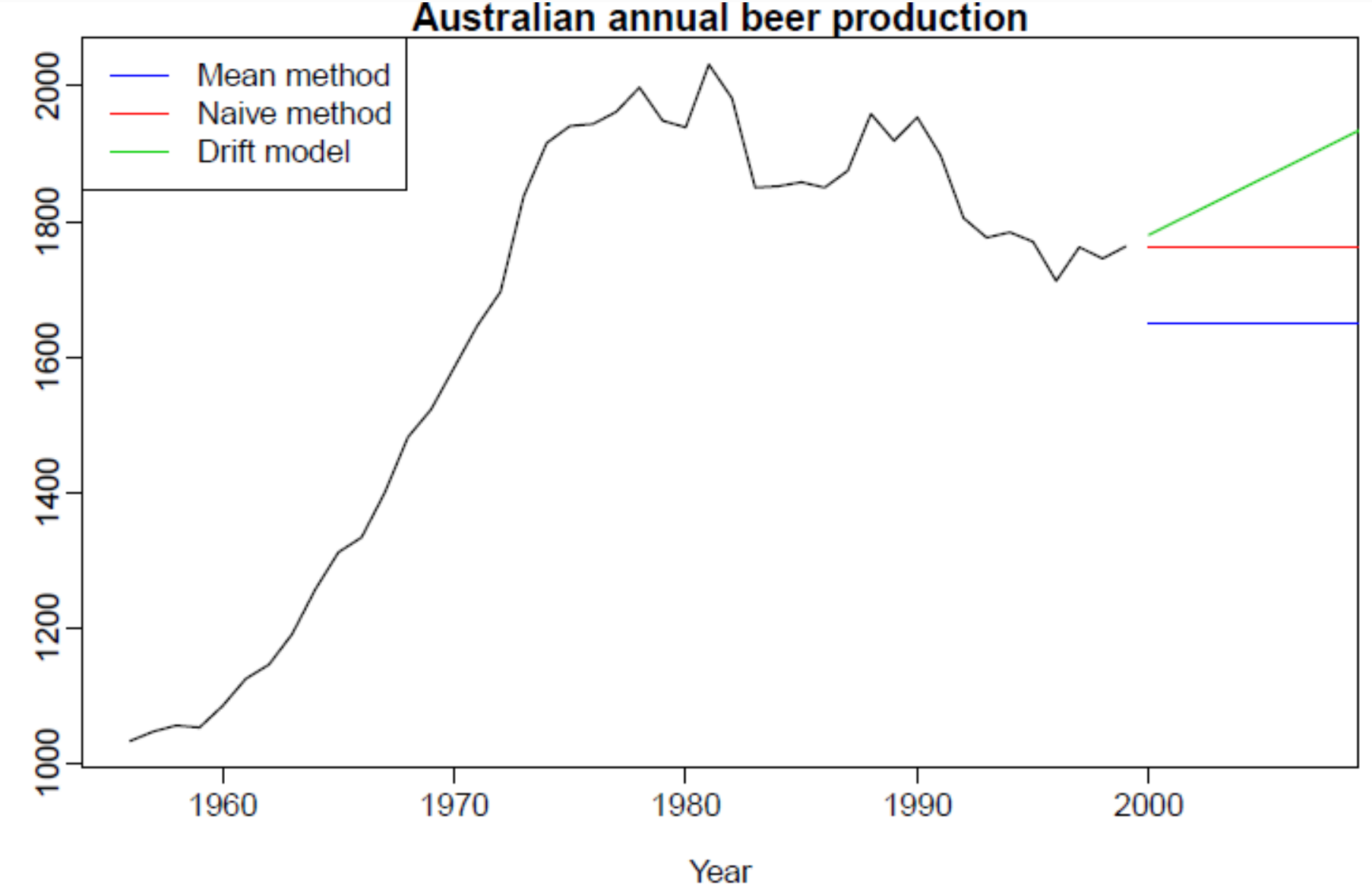


Simple Forecasting Methods

- **Naïve method**: Forecasts equal to last observed value;
- **Mean method**: Forecast of all future values is equal to mean of historical data;
- **Seasonal naïve method**: Forecasts equal to last value from same season.
- **Drift method**: Forecasts equal to last value plus average change.

Always compute the naïve forecasts as baseline for more complex forecasts…

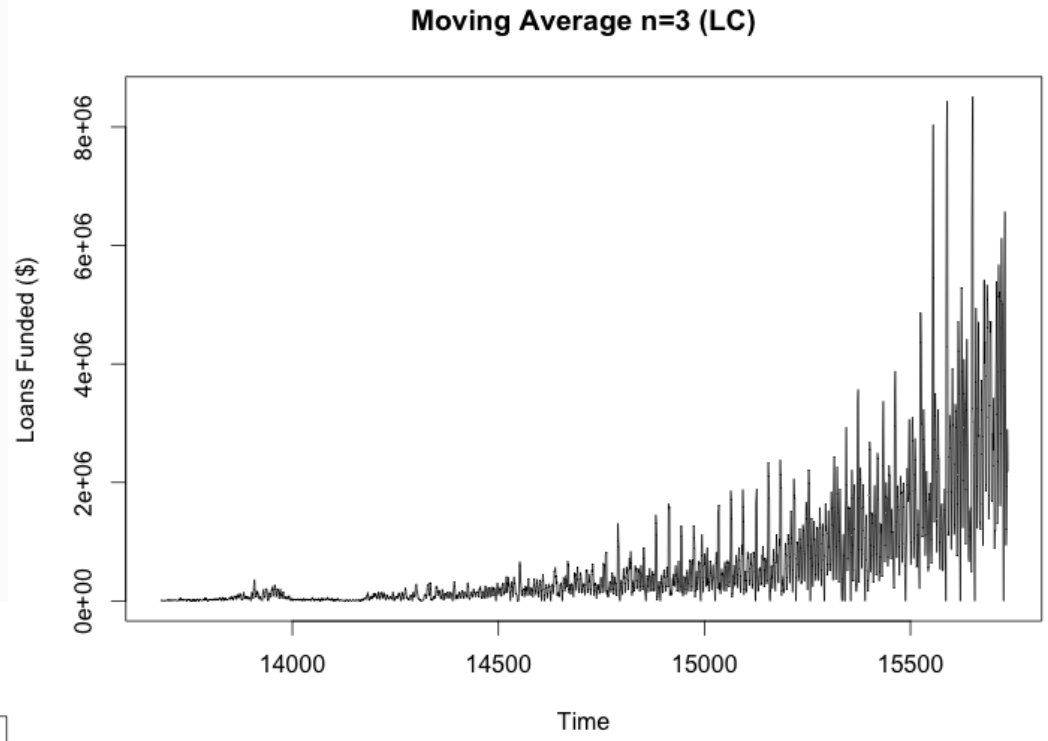
Simple Forecasting Methods

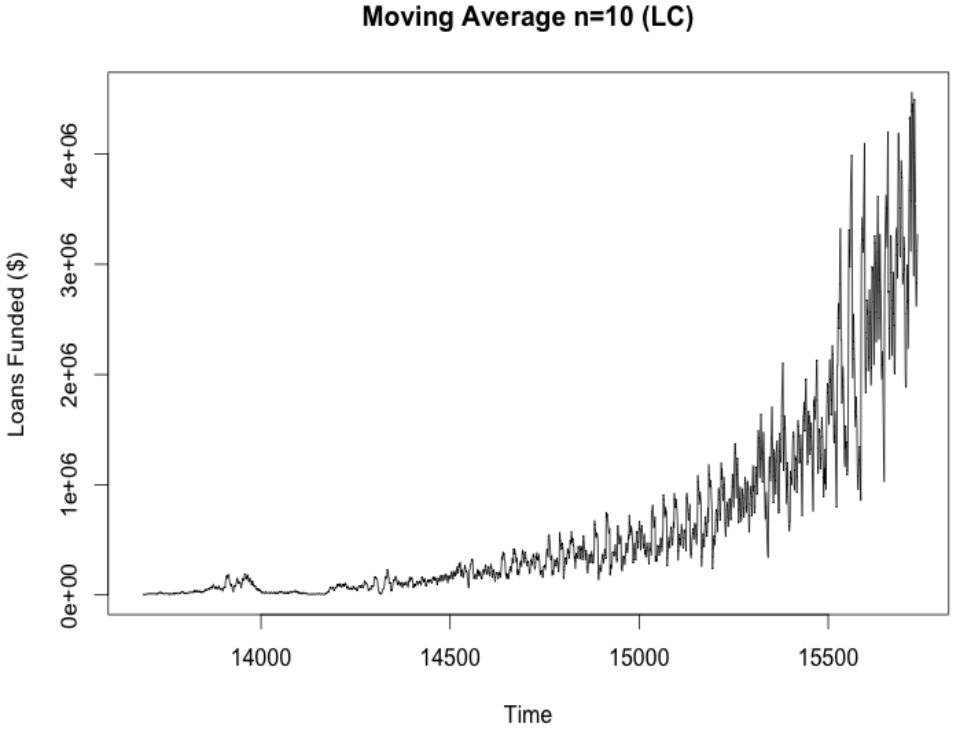


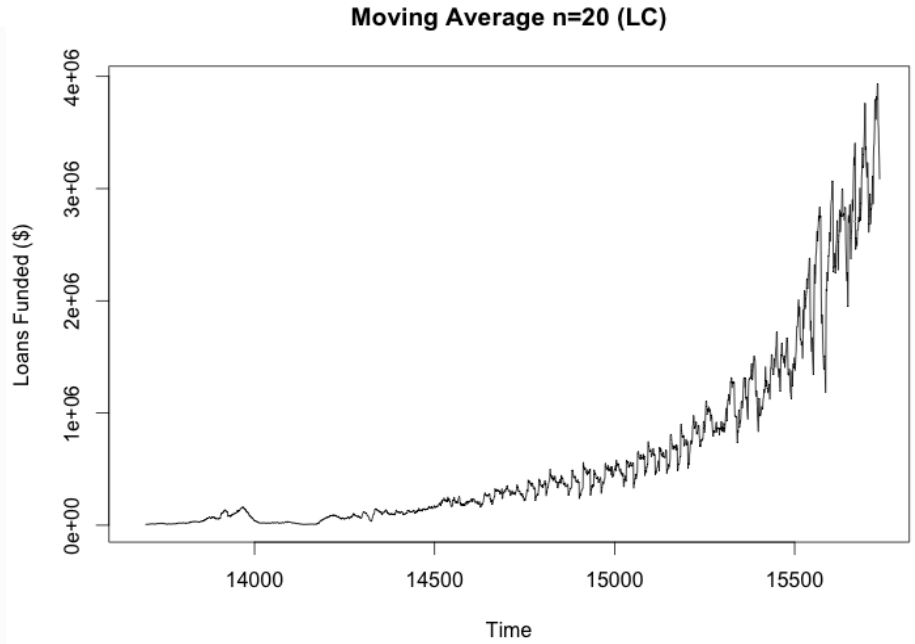
Moving Average – This period's predicted response is equal to the average of the past n periods response. This is known as a MA model of order n.

Useful for smoothing out noise to see trends in the data.

Moving Average





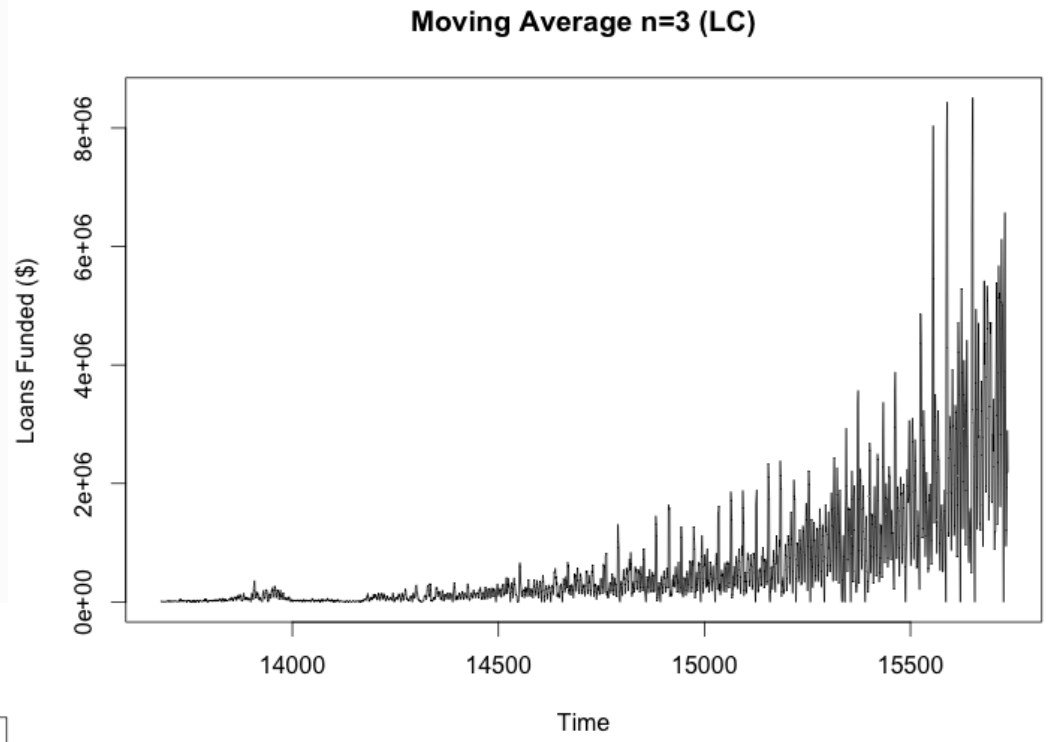


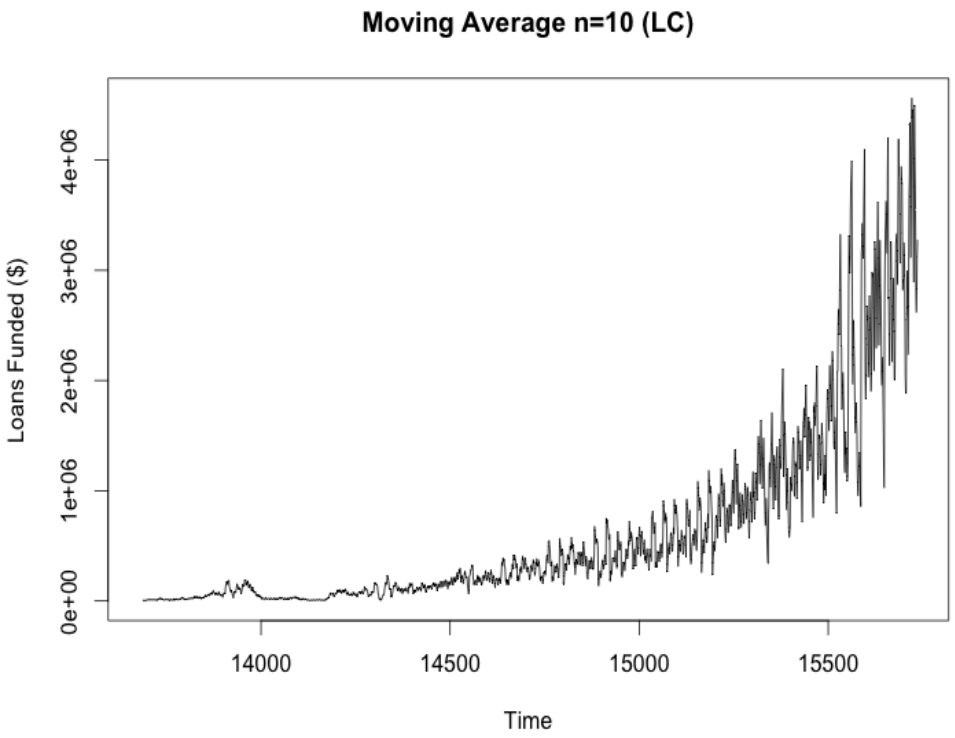
The key to time series analysis is to transform the data into a ‘stable’ time series, *nothing is forecastable until it is stable…*

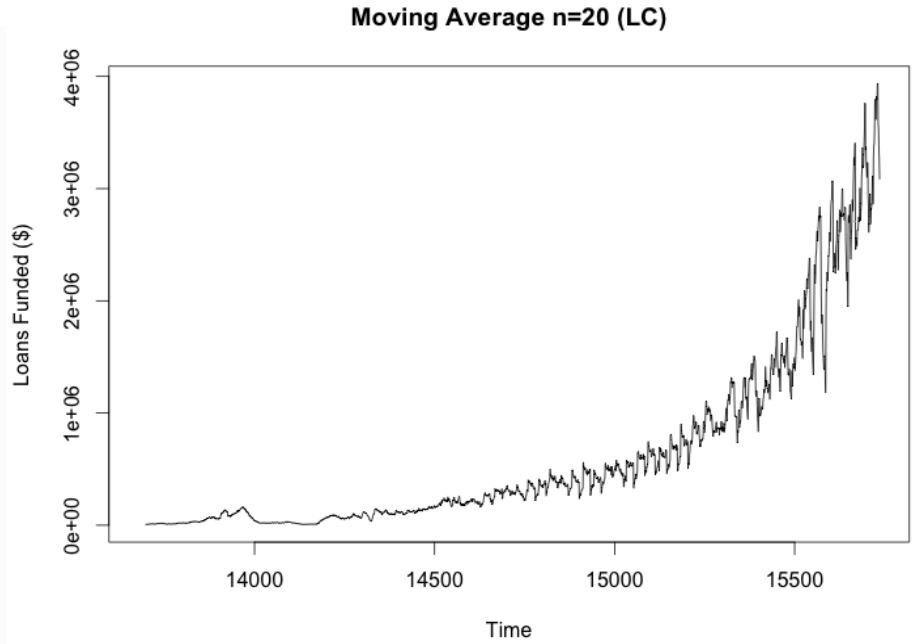
(1) Mean is nearly constant

(2) Volatility is nearly constant

Let's look again…







The key to time series analysis is to transform the data into a 'stable' time series:

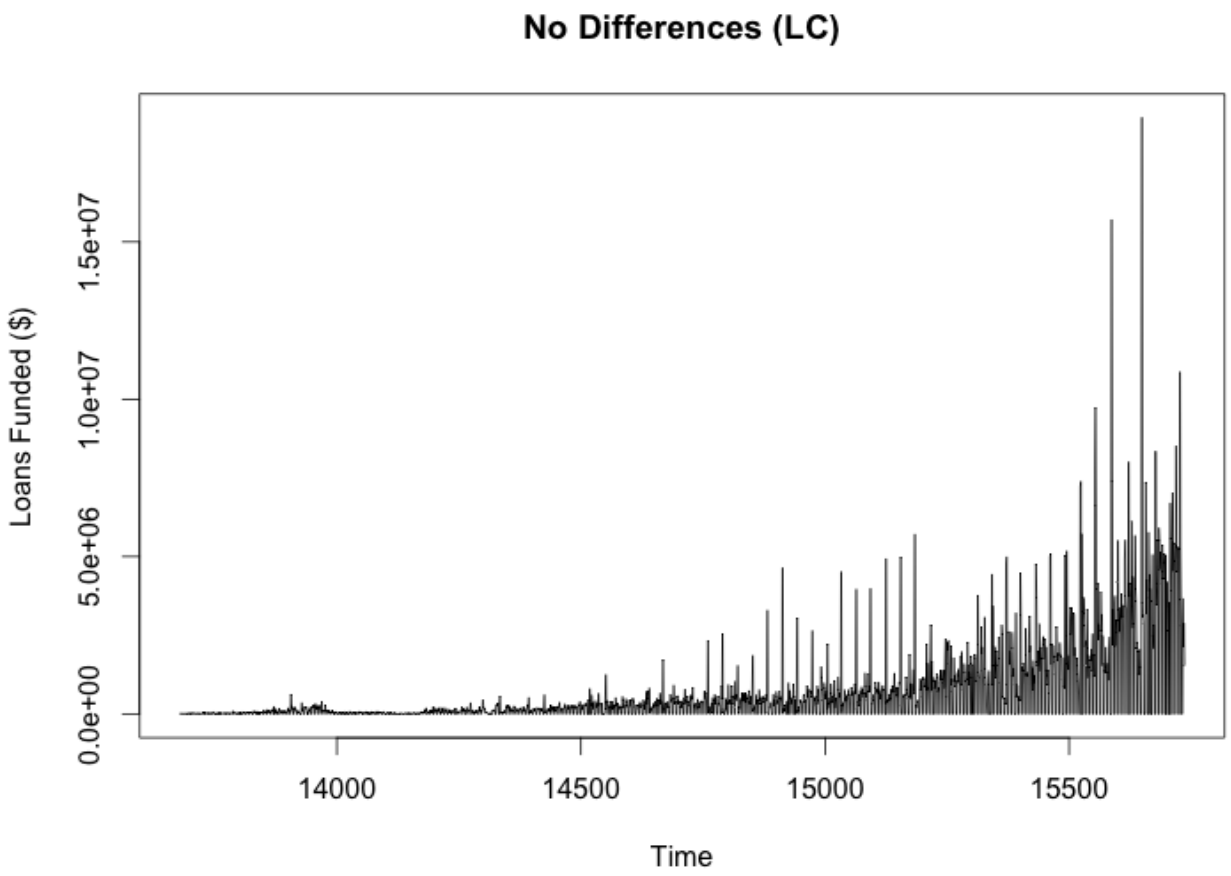
(1) Mean is nearly constant

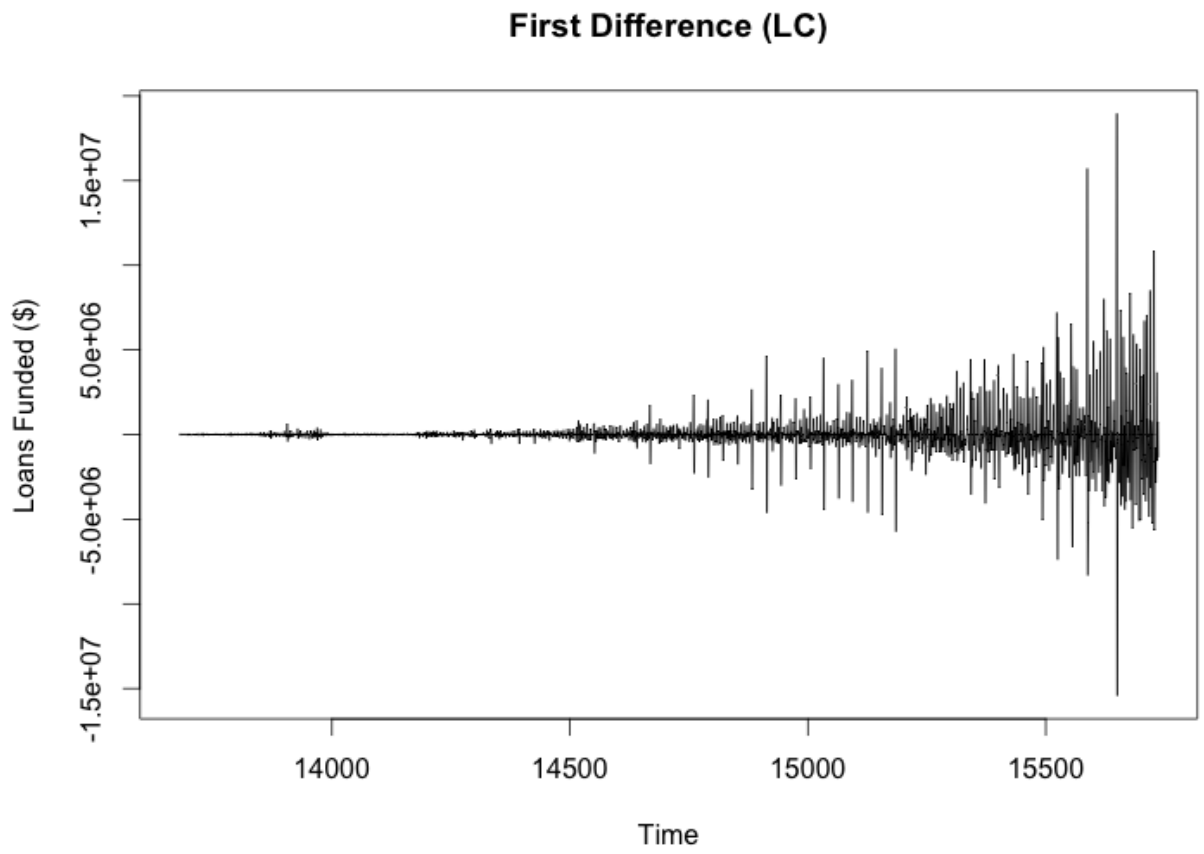
Transformation: take differences (diff() function in R)

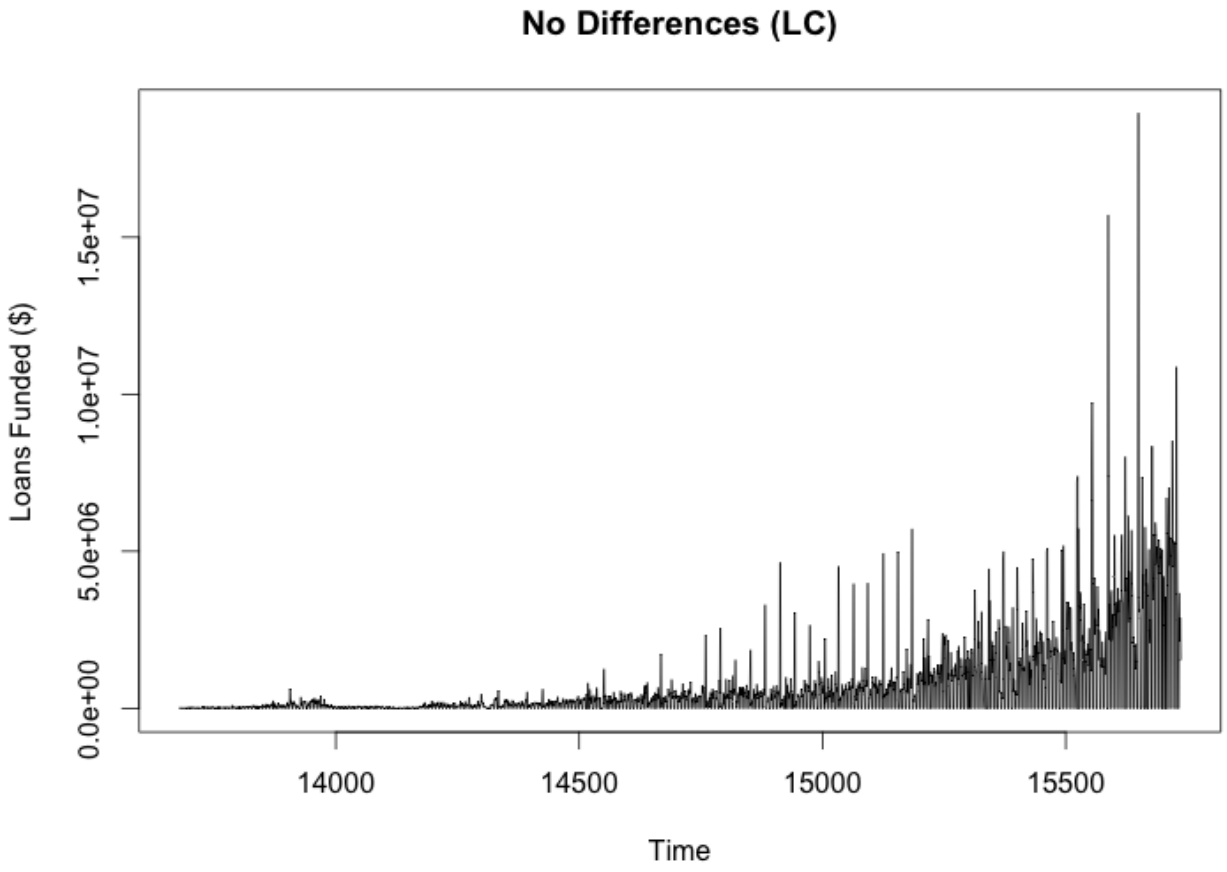
(2) Volatility is nearly constant

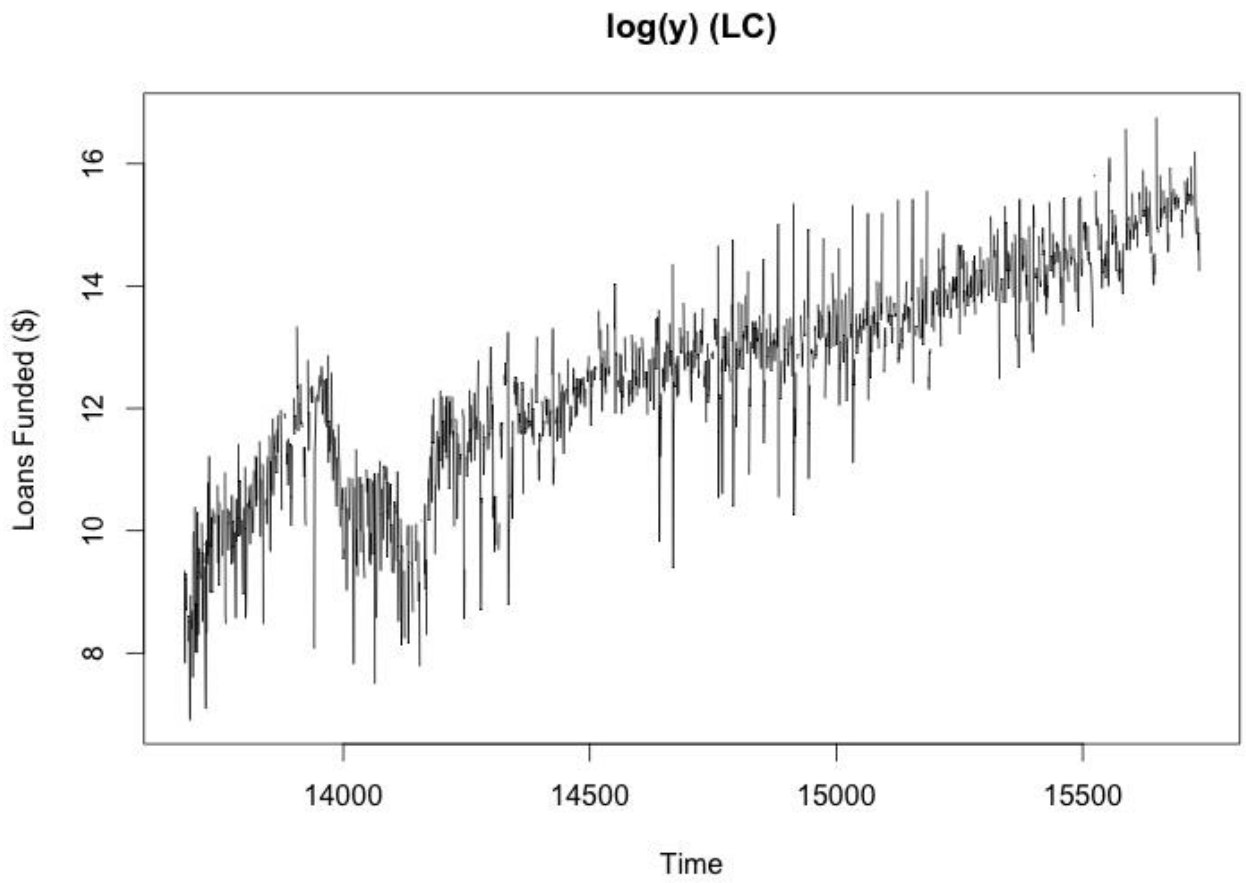
Transformation: take logs or powers. Box-Cox family of transformations flexibly covers both:

Y = (lambda*y + 1)^(1/lambda)

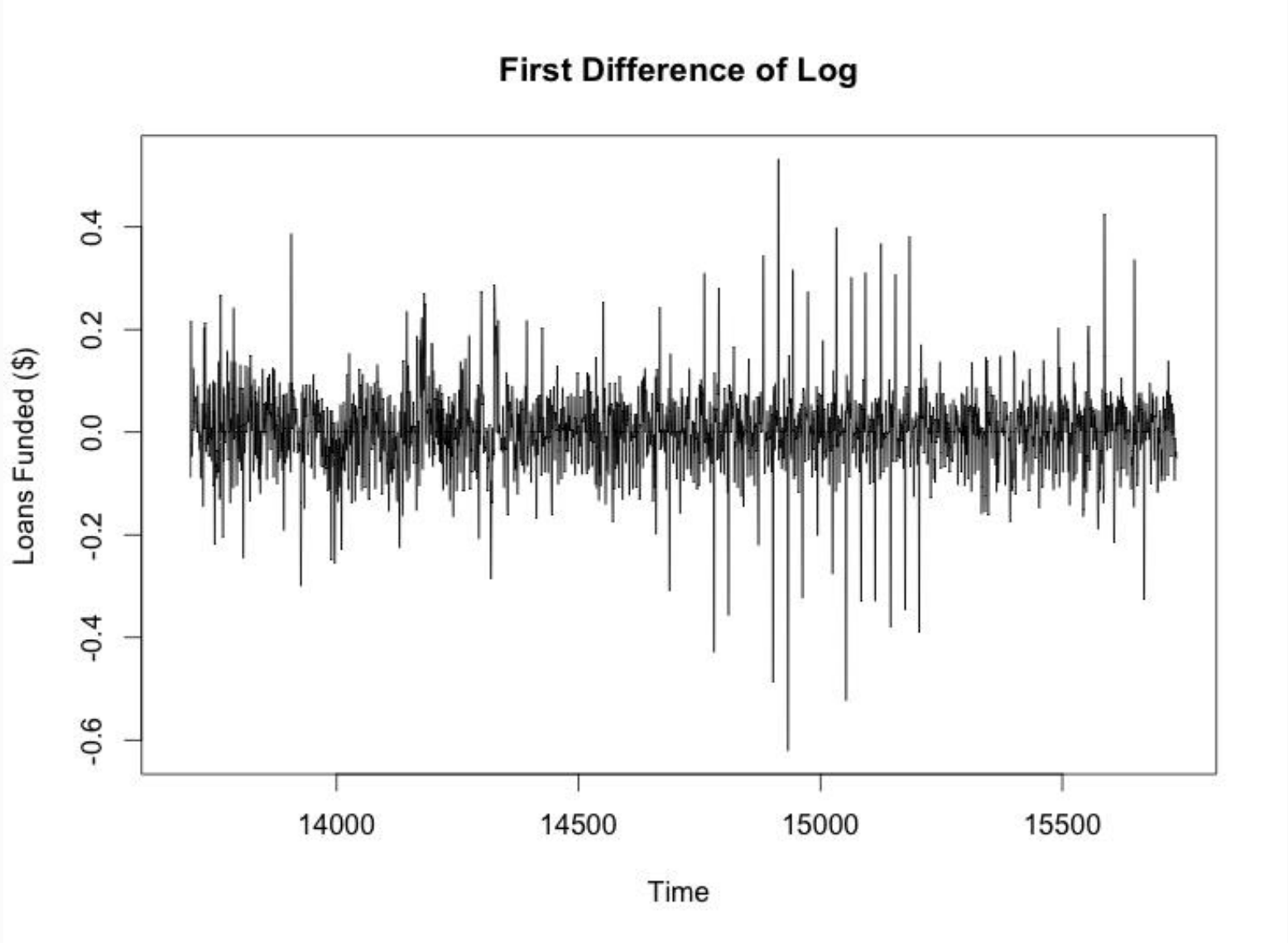

No Differences (LC)
Loans Funded ($)
0.0e+00
5.0e+06
1.0e+07
1.5e+07
14000
14500
15000
15500
Time


First Difference (LC)
Loans Funded ($)
-1.5e+07
-5.0e+06
5.0e+06
1.5e+07
14000
14500
15000
15500
Time

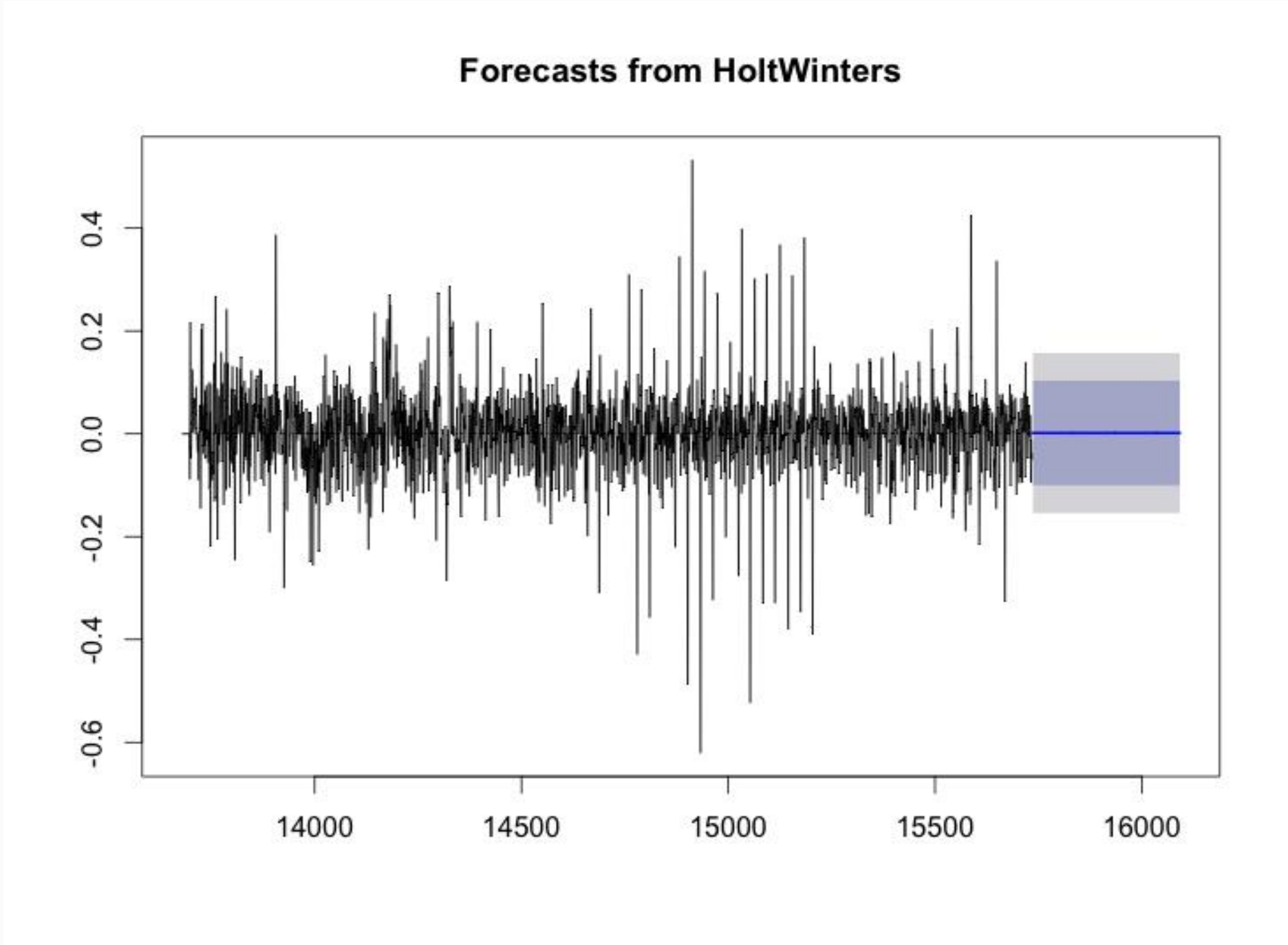

No Differences (LC)
Loans Funded ($)
0.0e+00
5.0e+06
1.0e+07
1.5e+07
14000
14500
15000
15500
Time


log(y) (LC)
Loans Funded ($)
8
10
12
14
16
14000
14500
15000
15500
Time

Ahhhh, Stability at last! Note that taking several differences of the data is not at all uncommon.



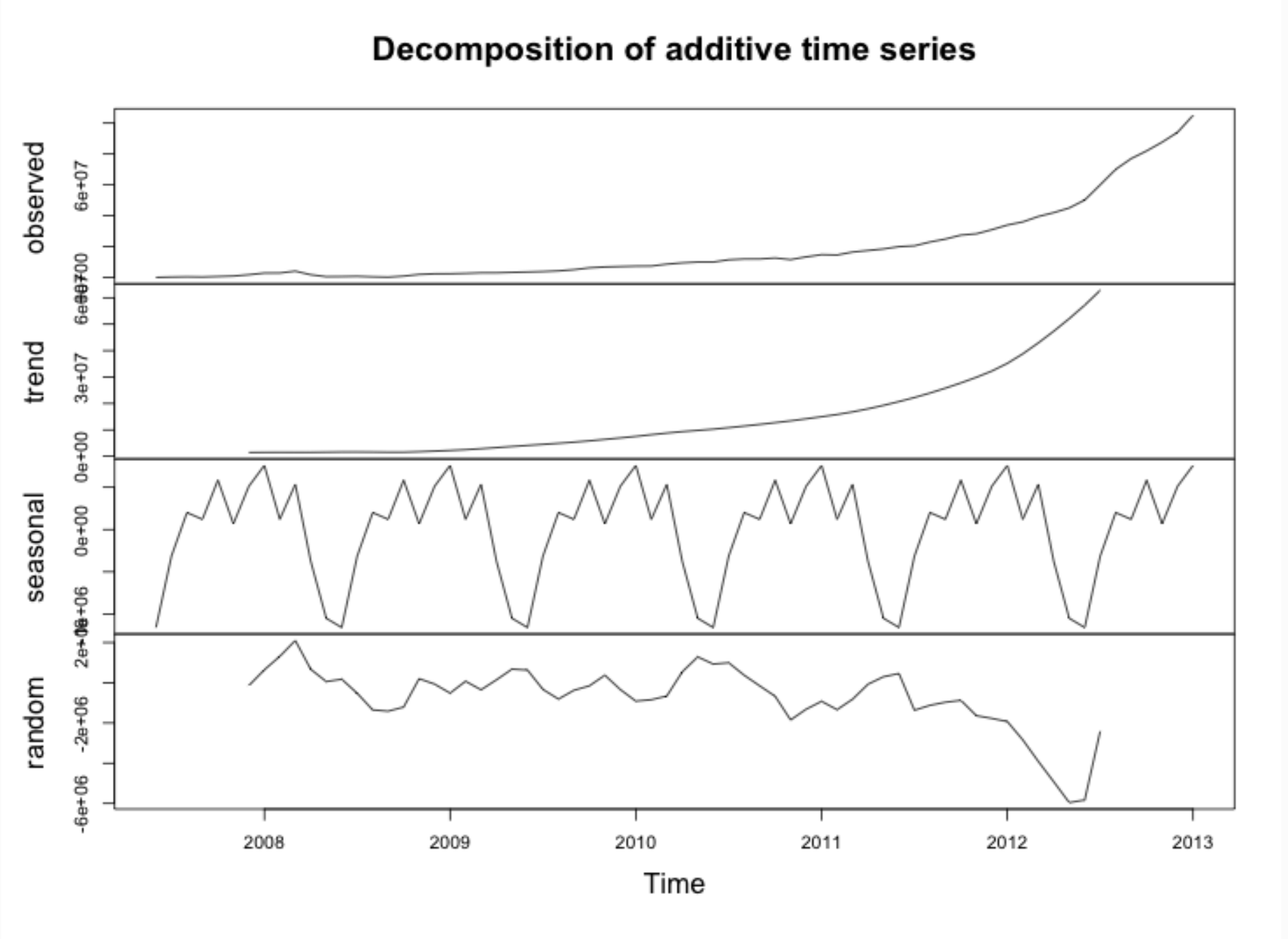
- Once you have a stable time series, you can forecast forward using ‘exponential smoothing’ models. Moving Averages are a special type of ES.
- ES models take in all past data, but put different weights on how recent data should predict the next period.
- One ES model is HoltWinters (HoltWinters() in R) that selects a smoothing parameter automatically.


Forecasts from HoltWinters
0.4
0.2
0.0
-0.2
-0.4
-0.6
14000
14500
15000
15500
16000

Decomposition: Sometimes you'll want to decompose your time series into 'trend' and seasonal components. There are several algorithms to accomplish this (see decompose() and stl() in R).

- Read in the data
- Create a time series data structure
- Forecast with ETS
- Plot output of ETS Forecast

Let's see if there is any seasonality in the monthly lending data:


Decomposition of additive time series
observed
trend
seasonal
random
6e+07
3e+07
0e+00
0e+00
2e+06
-2e+06
-6e+06
2008
2009
2010
2011
2012
2013
Time

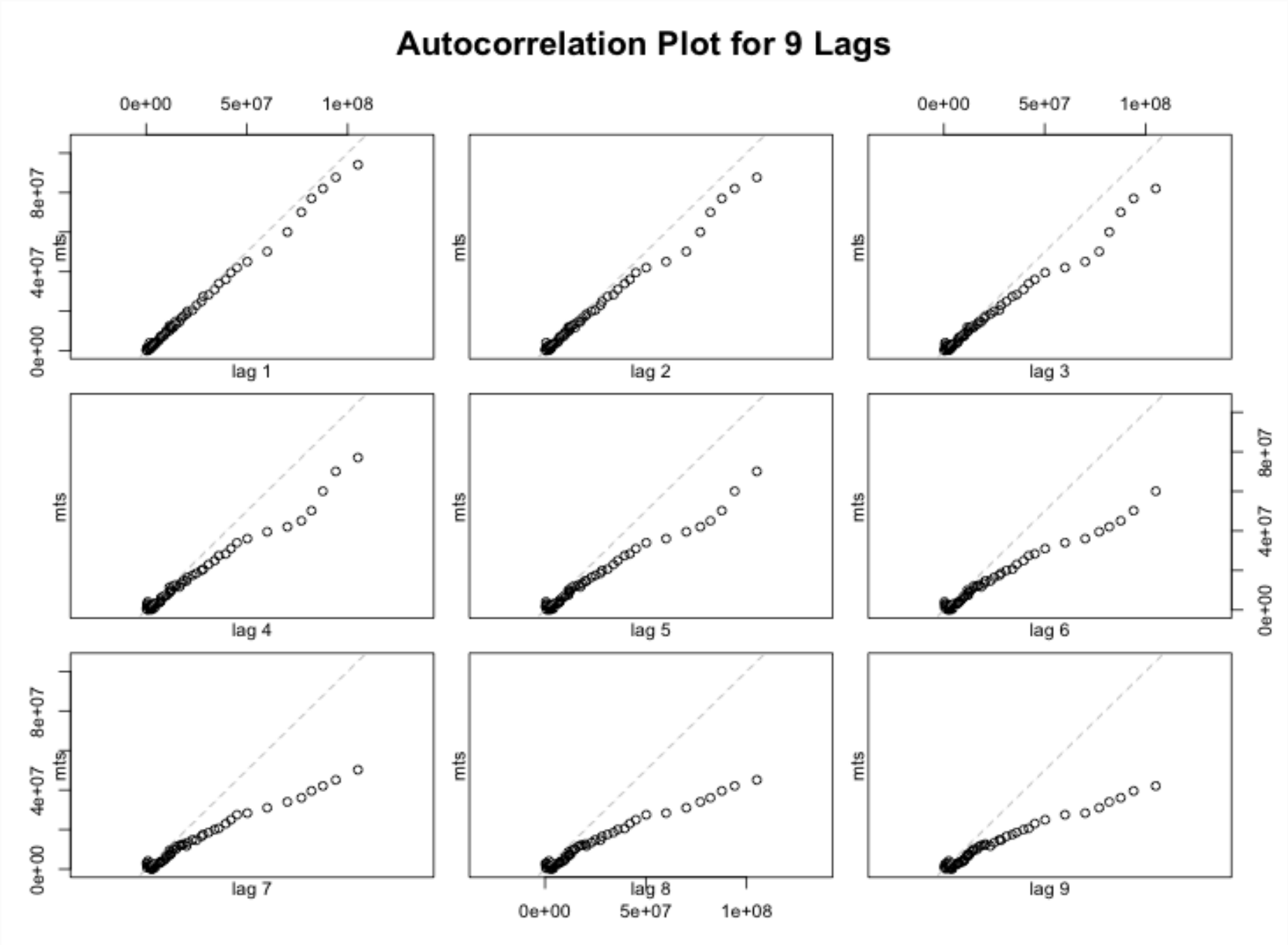
ES models does not assume autocorrelation, or correlation between the errors of successive y values. Not a model base approach.

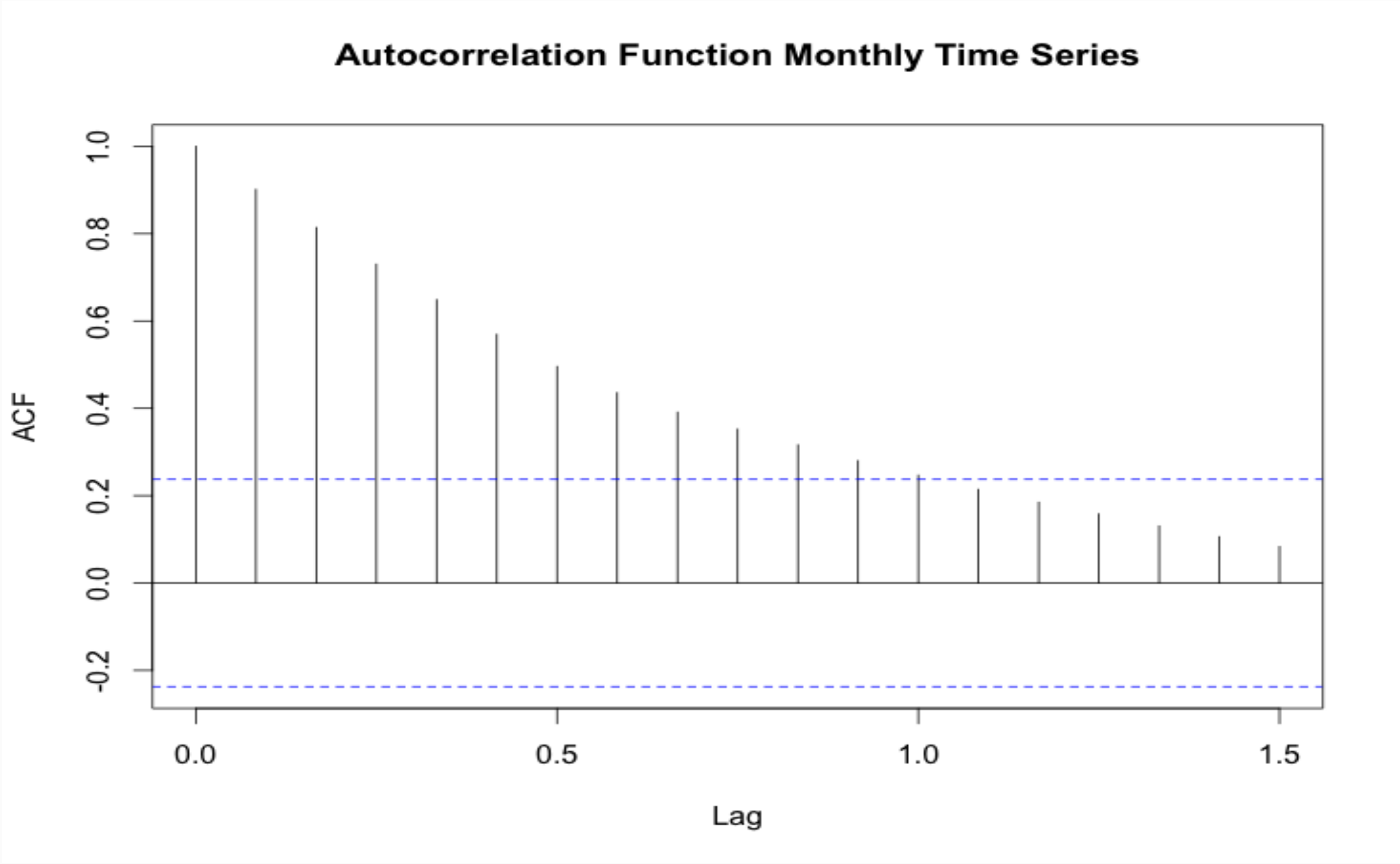
The correct model might need to take into account substantial correlation, for example the true model generating the data could be:

$$y(t) = \alpha * y(t-1) + e$$

Today's value is yesterday's value plus some error. This is an AR1 autoregressive model of order 1.

You can use lag.plot() or acf() in R to see autocorrelation structure:


Autocorrelation Plot for 9 Lags
0e+00
5e+07
1e+08
0e+00
4e+07
8e+07
mts
lag 1
lag 2
lag 3
lag 4
lag 5
lag 6
lag 7
lag 8
lag 9


Autocorrelation Function Monthly Time Series
ACF
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
0.0
0.5
1.0
1.5
Lag

There is a model that incorporates all the concepts of differences, MA, and AR, that is the ARIMA(p, d, q) model.

ARIMA(1, 0, 0) = AR(1)

ARIMA(0, 0, 1) = MA(1)

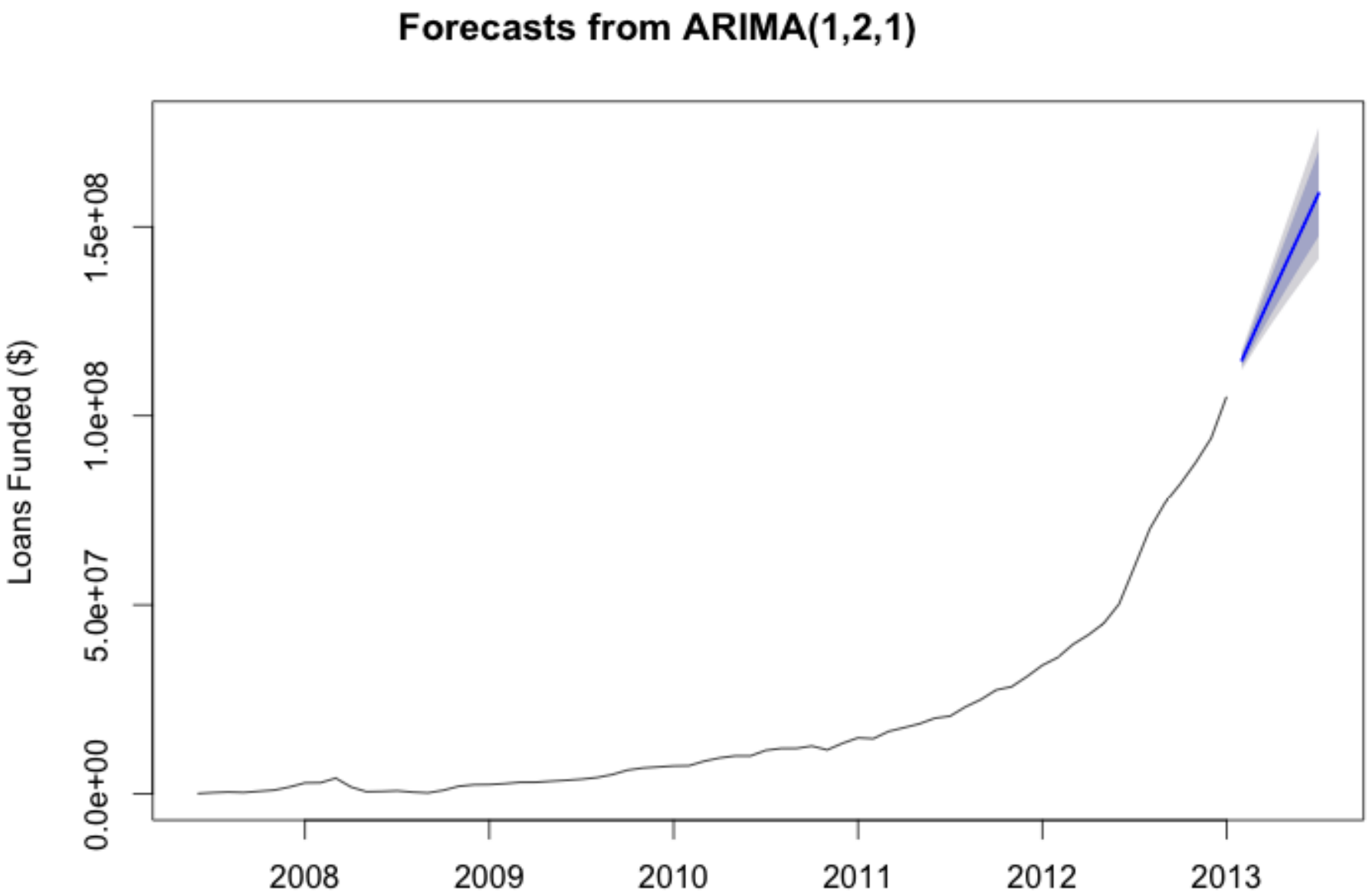
d = number of times we difference the data to obtain stationarity.

I won't go into how to select the values of p, q, it requires learning about the partial correlation function and the correlation function (see references at end of slides).

Let's be lazy and use auto.arima() in R to pick them for us:

R Functions

- The arima() function in the stats package provides seasonal and non-seasonal ARIMA model estimation including covariates.
- However, it does not allow a constant unless the model is stationary
- It does not return everything required for forecast().
- It does not allow re-fitting a model to new data.
- Use the Arima() function in the forecast package which acts as a wrapper to arima().
- Or use the auto.arima() function in the forecast package.


Forecasts from ARIMA(1,2,1)
Loans Funded ($)
0.0e+00
5.0e+07
1.0e+08
1.5e+08
2008
2009
2010
2011
2012
2013

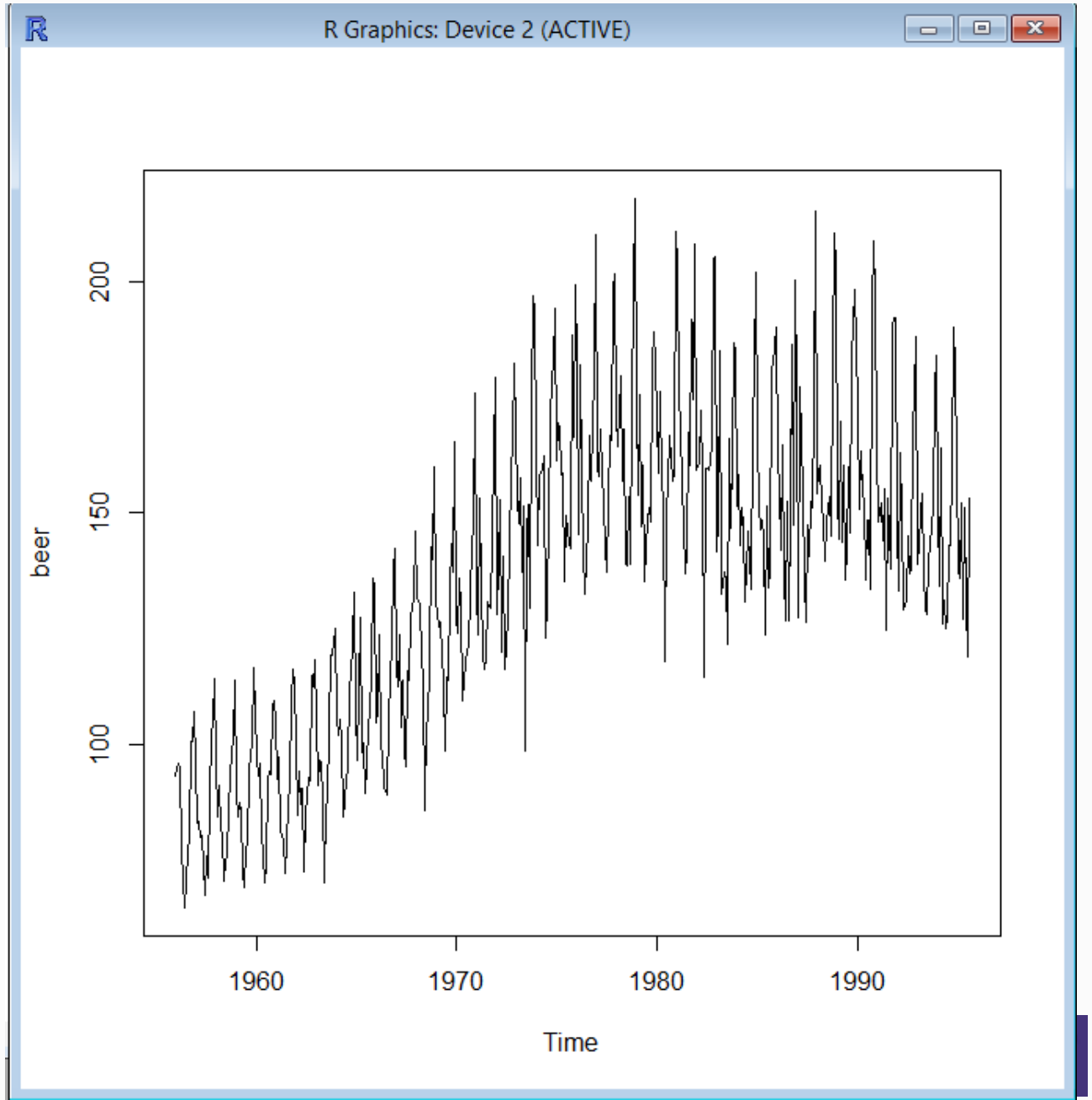
Hands On, *with beer of course…*

- beer<-read.csv("C:/data/beer.csv")
- beer
- beer.ts<-ts(beer, freq=12, start=1956)
- beer.ts

```
> beer.ts<-ts(beer,freq=12,start=1956)
> beer.ts
      Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
1956  93.2  96.0  95.2  77.1  70.9  64.8  70.1  77.3  79.5 100.6 100.7 107.1
1957  95.9  82.8  83.3  80.0  80.4  67.5  75.7  71.1  89.3 101.1 105.2 114.1
1958  96.3  84.4  91.2  81.9  80.5  70.4  74.8  75.9  86.3  98.7 100.9 113.8
1959  89.8  84.4  87.2  85.6  72.0  69.2  77.5  78.1  94.3  97.7 100.2 116.4
1960  97.1  93.0  96.0  80.5  76.1  69.9  73.6  92.6  94.2  93.5 108.5 109.4
1961 105.1  92.5  97.1  81.4  79.1  72.1  78.7  87.1  91.4 109.9 116.3 113.0
1962 100.0  84.8  94.3  87.1  90.3  72.4  84.9  92.7  92.2 114.9 112.5 118.3
1963 106.0  91.2  96.6  96.3  88.2  70.2  86.5  88.2 102.8 119.1 119.2 125.1
1964 106.1 102.1 105.2 101.0  84.3  87.5  92.7  94.4 113.0 113.9 122.9 132.7
1965 106.9  96.6 127.3  98.2 100.2  89.4  95.3 104.2 106.4 116.2 135.9 134.0
1966 104.6 107.1 123.5  98.8  98.6  90.6  89.1 105.2 114.0 122.1 138.0 142.2
1967 116.4 112.6 123.8 103.6 113.9  98.6  95.0 116.0 113.9 127.5 131.4 145.9
1968 131.5 131.0 130.5 118.9 114.3  85.7 104.6 105.1 117.3 142.5 140.0 159.8
1969 131.2 125.4 126.5 119.4 113.5  98.7 114.5 113.8 133.1 143.4 137.3 165.2
1970 126.9 124.0 135.7 130.0 109.4 117.8 120.3 121.0 132.3 142.9 147.4 175.9
1971 132.6 123.7 153.3 134.0 119.6 116.2 118.6 130.7 129.3 144.4 163.2 179.4
1972 128.1 138.4 152.7 120.0 140.5 116.2 121.4 127.8 143.6 157.6 166.2 182.3
1973 153.1 147.6 157.7 137.2 151.5  98.7 145.8 151.7 129.4 174.1 197.0 193.9
1974 164.1 142.8 157.9 159.2 162.2 123.1 130.0 150.1 169.4 179.7 182.1 194.3
1975 161.4 169.4 168.8 158.1 158.5 135.3 149.3 143.4 142.2 188.4 166.2 199.2
1976 182.7 145.2 182.1 158.7 141.6 132.6 139.6 147.0 166.6 157.0 180.4 210.2
1977 159.8 157.8 168.2 158.4 152.0 142.2 137.2 152.6 166.8 165.6 198.6 201.5
1978 170.7 164.4 179.7 157.0 168.0 139.3 138.6 153.4 138.9 172.1 198.4 217.8
1979 173.7 153.8 175.6 147.1 160.3 135.2 148.8 151.0 148.2 182.2 189.2 183.1
1980 170.0 158.4 176.1 156.2 153.2 117.9 149.8 156.6 166.7 156.8 158.6 210.8
1981 203.6 175.2 168.7 155.9 147.3 137.0 141.1 167.4 160.2 191.9 174.4 208.2
1982 159.4 161.1 172.1 158.4 114.6 159.6 159.7 159.4 160.7 165.5 205.0 205.2
1983 141.6 148.1 184.9 132.5 137.3 135.5 121.7 166.1 146.8 162.8 186.8 185.5
1984 151.5 158.1 143.0 151.2 147.6 130.7 137.5 146.1 133.6 167.9 181.9 202.0
1985 166.5 151.3 146.2 148.3 144.7 123.6 151.6 133.9 137.4 181.6 182.0 190.0
```

Hands On, *with beer of course…*

- beer<-read.csv("C:/data/beer.csv")
- beer
- beer.ts<-ts(beer,freq=12,start=1956)
- beer.ts
- plot(beer.ts)



Hands On, *with beer of course…*

Install the forecast package

- install.packages("forecast",dependencies=TRUE)
- library(forecast)
- beer.forecast<-forecast(beer.ts)

```
> library(forecast)
Loading required package: zoo

Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric

Loading required package: timeDate
This is forecast 5.4

> beer.forecast<-forecast(beer.ts)
```

Hands On, *with beer of course…*

- beer.forecast

```
> beer.forecast
         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
Sep 1995       137.6719 126.1027 150.1181 119.7242 155.9623
Oct 1995       163.6163 149.5498 177.9853 142.3458 185.9650
Nov 1995       176.9278 161.8177 192.4684 153.1533 200.3972
Dec 1995       185.1457 168.7186 201.4400 160.4824 210.5974
Jan 1996       148.9856 135.9492 162.2837 129.0337 169.2323
Feb 1996       139.6844 127.2660 151.9764 120.7358 158.9403
Mar 1996       154.4957 140.9662 168.6019 133.3387 175.5120
Apr 1996       140.2195 127.7008 152.9412 120.8261 160.0342
May 1996       138.8370 125.6700 151.6901 119.1452 158.2742
Jun 1996       126.1047 114.3872 137.5936 108.3582 144.3785
Jul 1996       133.9938 121.4964 146.7853 115.1170 153.8506
Aug 1996       142.2314 128.6546 155.8524 121.3206 163.0109
Sep 1996       139.8387 126.2695 153.9301 118.9128 161.1545
Oct 1996       165.9383 149.0454 182.6105 141.3544 192.1316
Nov 1996       179.1921 161.2811 197.9862 151.9140 208.4112
Dec 1996       187.2825 167.9393 207.4322 157.6778 218.2772
Jan 1997       150.5364 134.8514 166.7999 127.2215 175.7824
Feb 1997       140.9959 125.9672 156.6217 118.7823 165.1082
Mar 1997       155.8040 138.9315 172.9308 130.7512 182.5870
Apr 1997       141.2906 125.7667 157.6570 117.9135 167.6736
May 1997       139.7937 124.2062 156.4828 115.8210 165.6742
Jun 1997       126.8886 112.0393 141.6206 104.6596 149.1675
Jul 1997       134.7452 118.8042 150.8047 111.5423 160.4643
Aug 1997       142.9509 126.1972 160.4349 117.9852 170.8102
>
```

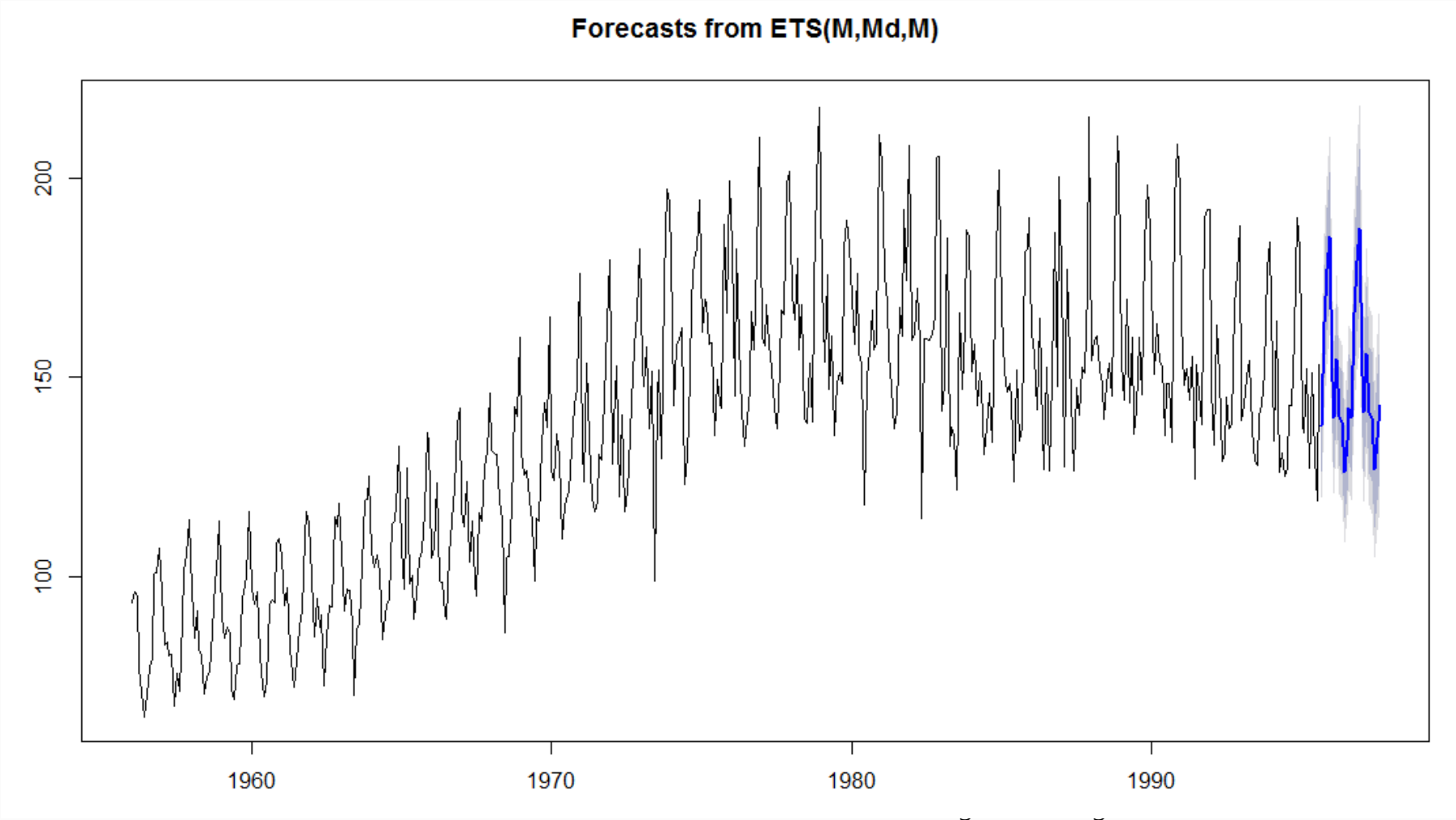
Hands On, *with beer of course…*

- beer.forecast$method

  [1] "ETS(M,Md,M)“ – Holt Winter, multiplicative error, multiplicative damped trend, multiplicative seasonality,

Hands On, *with beer of course…*

- beer.forecast$method
- plot(beer.forecast)



Hands On, *recap*…

- beer<-read.csv, read in the time series;
- beer.ts<-ts(beer, frequency, start label)
- Install.packages("forecast" and bring in all dependent files)
- Library(forecast), load it for use
- Beer.forecast <- forecast(beer.ts)
- plot(beer.forecast)

Residual Diagnostics

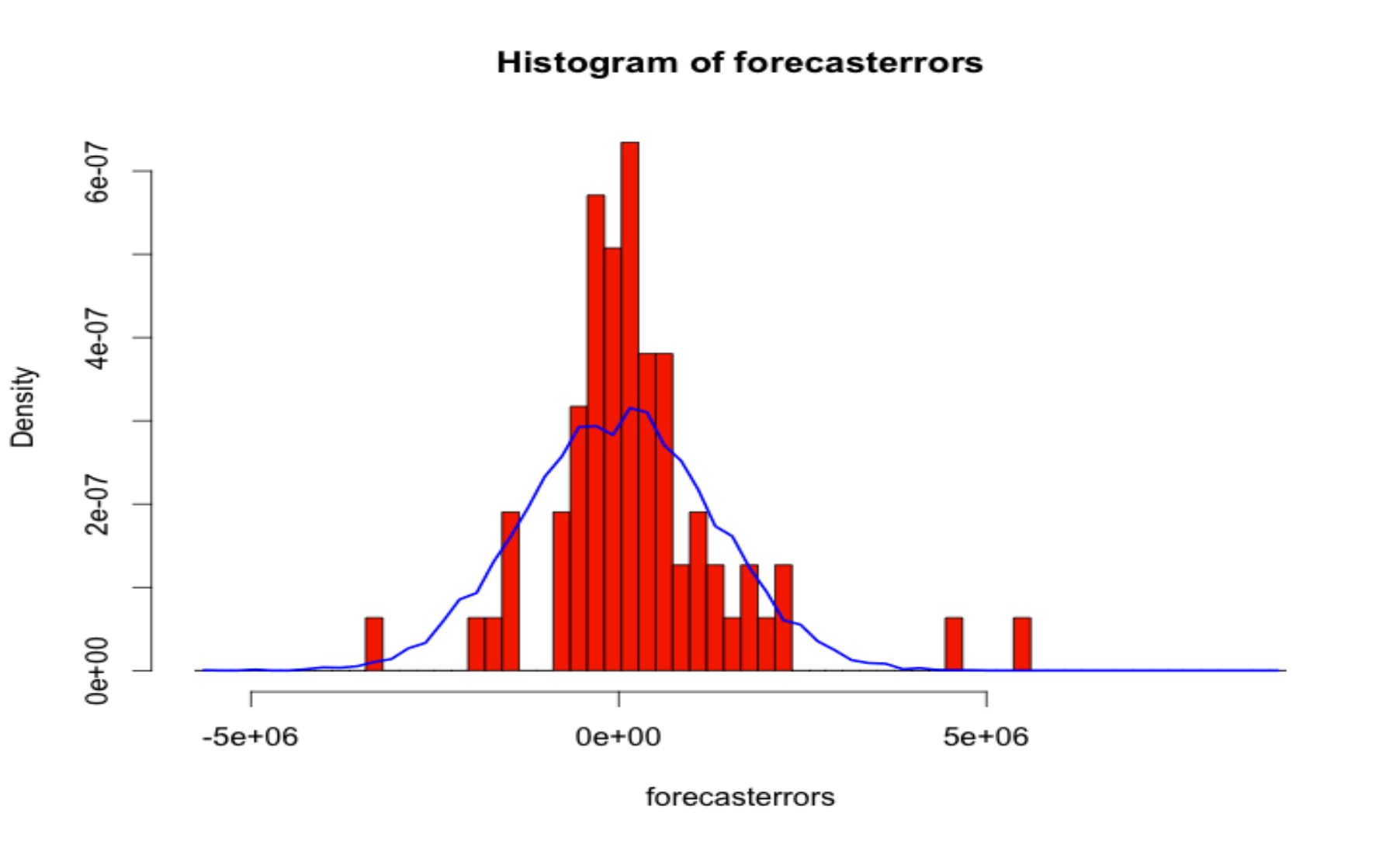
A residual in forecasting is the difference between an observed value and its forecast based on other observations: ei=yi−y^i. For time series forecasting, a residual is based on one-step forecasts; that is y^t is the forecast of yt based on observations y1,...,yt−1.

A good forecasting method will yield residuals with the following properties:

- The residuals are uncorrelated. If there are correlations between residuals, then there is information left in the residuals which should be used in computing forecasts.
- The residuals have zero mean. If the residuals have a mean other than zero, then the forecasts are biased.

ARIMA(1, 2, 1) predictions for Lending Club disbursements:

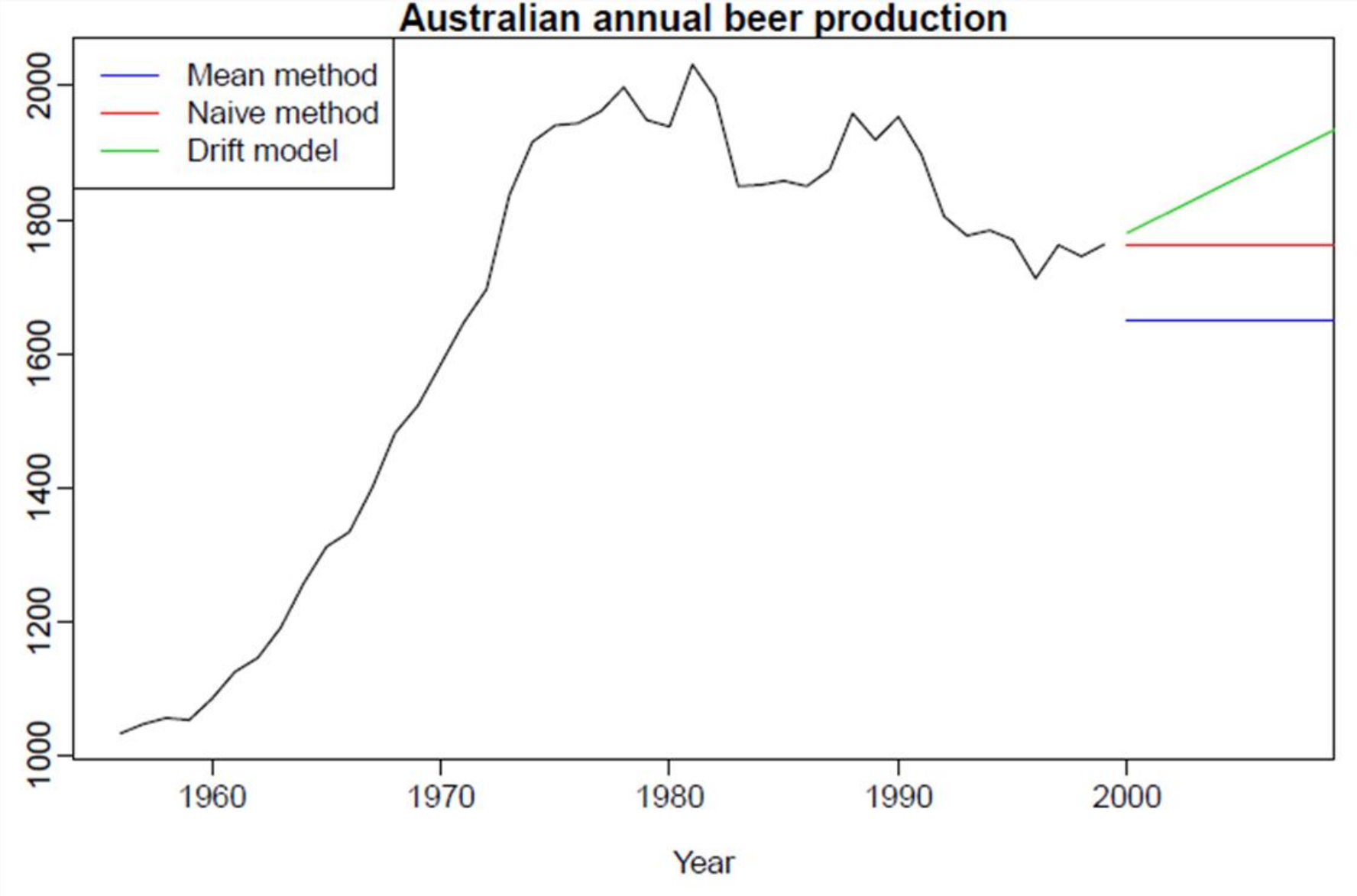
Feb	Mar	Apr	May	Jun	Jul
114,727,363	123,818,067	132,671,221	141,424,018	150,134,416	158,826,902



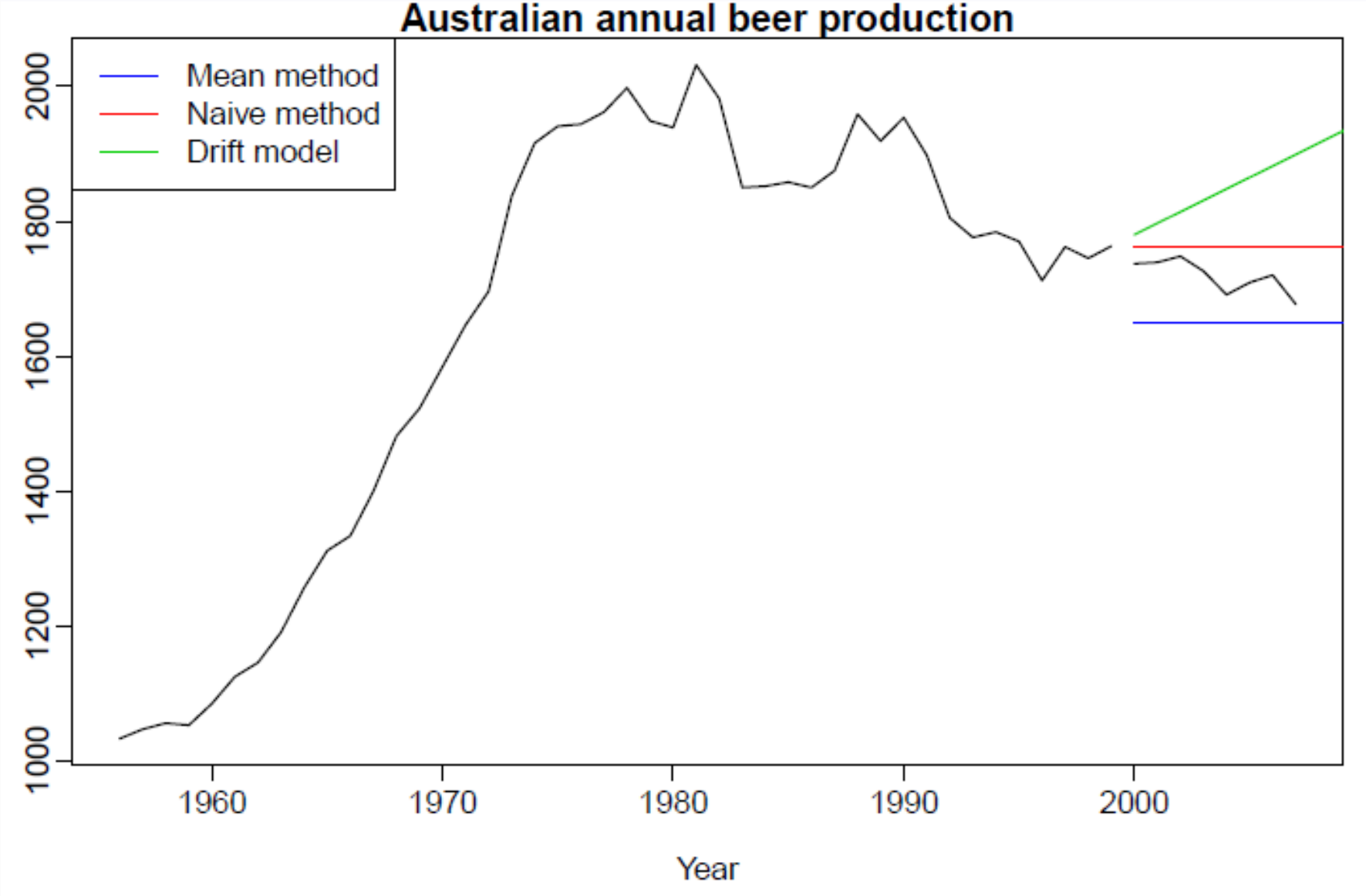
- Fit your model using training data, and validating using test data.
- Two most common model validation metrics are:
  Mean Absolute Error = mean(|error|)
  Root Mean Squared Error = sqrt(mean(error^2))
  These are scale dependent, so OK if comparing forecasts on the same data set or same scale of data…
- Mean Absolute Percentage Error, MAPE
  erri = Yi – Yihat, Pi = 100erri/Yi, MAPE = mean(|Pi|)
- Mean Absolute Squared Error, MASE

There is vociferous debate in the forecasting journals about the best metric to use – very domain specific.

Measures of Forecast Accuracy



Measures of Forecast Accuracy



Measures of Forecast Accuracy

Mean method

RMSE	MAE	MAPE	MASE
72.4223	68.6477	3.9775	1.5965

Naïve method

RMSE	MAE	MAPE	MASE
50.2382	44.6250	2.6156	1.0378

Drift method

RMSE	MAE	MAPE	MASE
134.6788	121.1250	7.0924	2.8169

Cross-Validation

Standard cross-validation

- A more sophisticated version of training/test sets.
  - Select one observation for test set, and use remaining observations in training set;

    Compute error on test observation.
  - Repeat using each possible observation as the test set.
  - Compute accuracy measure over all errors.
  - Does not work for time series because we cannot use future observations to build a model.

Time Series Cross-Validation

Assume k is the minimum number of observations for a training set.

- Select observation k + i + 1 for test set, and use observations at times 1, 2,..., k + i to estimate model. Compute error on forecast for time k + i.
- Repeat for i = 0, 1,..., n – k – 1 where n is the total number of observations.
- Compute accuracy measure over all errors.

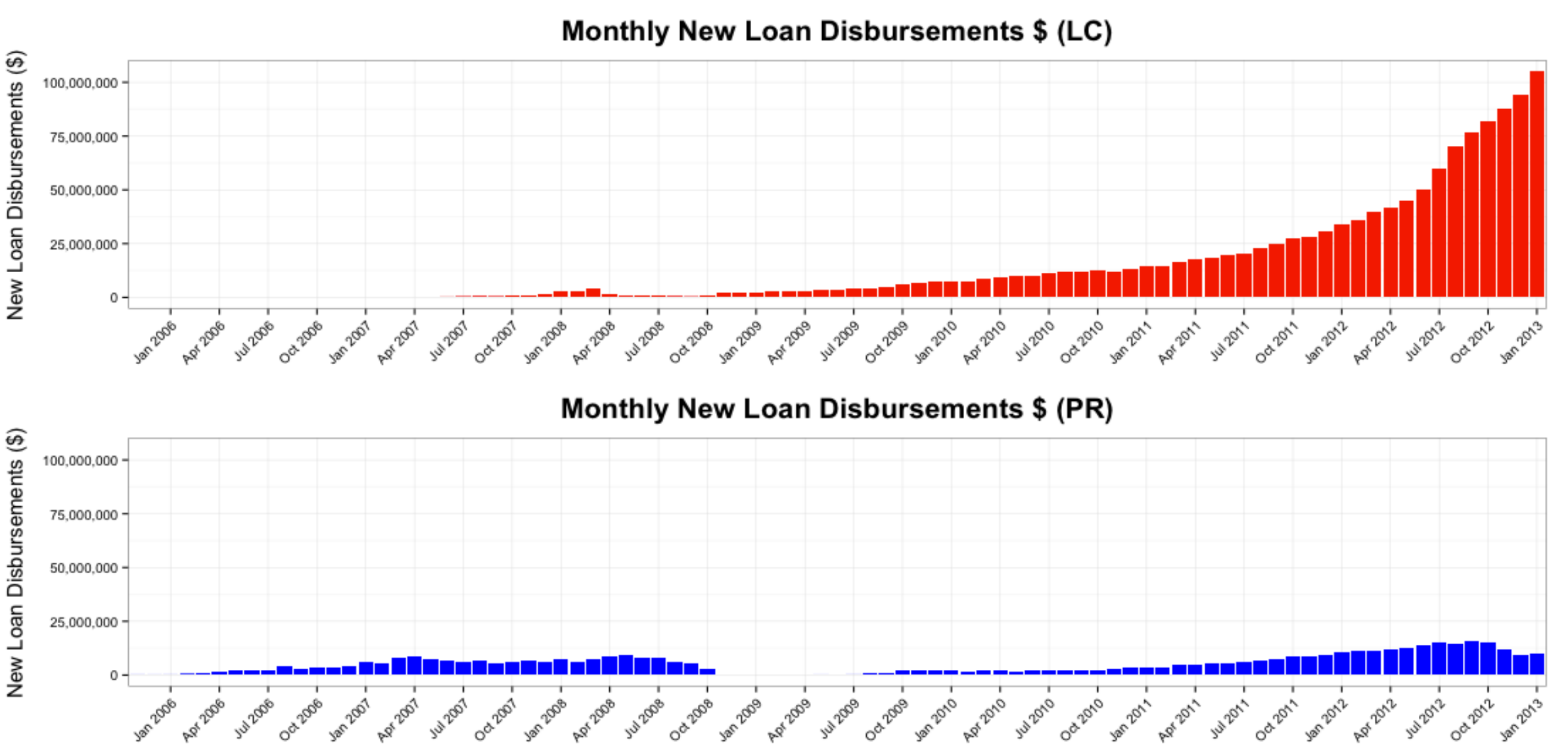
Also called the ***rolling forecasting origin*** *because the origin (k + i - 1) at which forecast is based rolls forward in time.*

Forecasting Process

1) Define problem, gather contextual knowledge

Best Practices

Gather as much contextual knowledge (experts) as possible.



Forecasting Process (manual)

1) Define problem, gather contextual knowledge
2) Plot like crazy to learn data structure (correlation matrices, autocorrelation plots, etc.) of the time series
3) Split data into train/test set.
4) (Time Series) Transform the data to obtain stationarity.
5) Fit appropriate models, test that errors look like white noise.
6) Apply models to test data set, evaluate using error deviation metrics.

Make forecasts.

7) Re-evaluate next period.

Resources: https://www.otexts.org/book/fpp

Forecasting is required in many situations. Stocking an inventory may require forecasts of demand months in advance. Telecommunication routing requires traffic forecasts a few minutes ahead. Whatever the circumstances or time horizons involved, forecasting is an important aid in effective and efficient planning.

This textbook provides a comprehensive introduction to forecasting methods and presents enough information about each method for readers to use them sensibly.

Authors

Rob J Hyndman

George Athanasopoulos

Table of contents

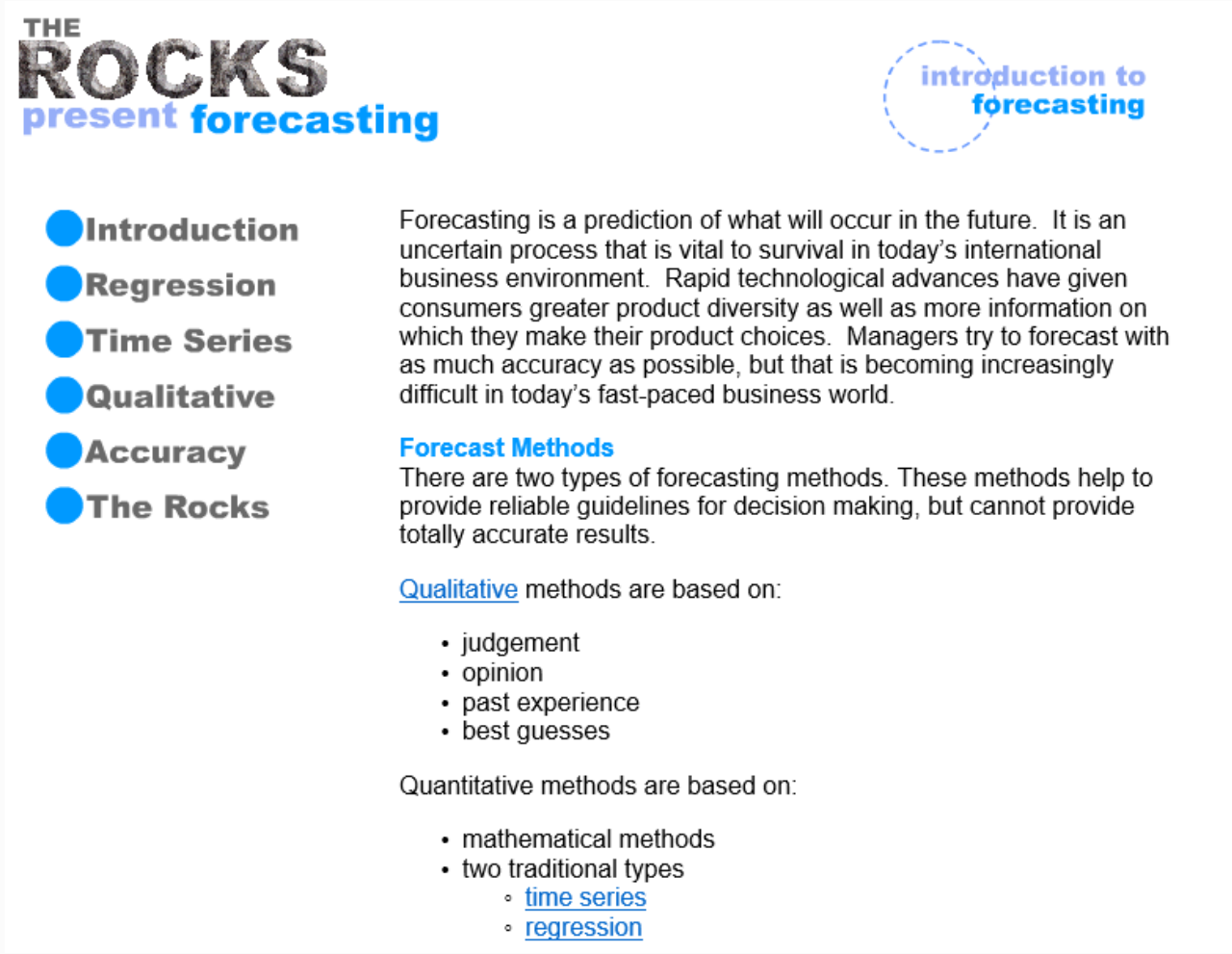
Recently updated pages

Holt-Winters seasonal method - 25 Jun 2014
Resources - 19 Jun 2014
Estimation and order selection - 12 Jun 2014
Some useful predictors - 14 May 2014
Damped trend methods - 01 May 2014
Using R - 11 Apr 2014
Holt's linear trend method - 27 Mar 2014
Stationarity and differencing - 26 Mar 2014
Simple exponential smoothing - 25 Mar 2014
Non-seasonal ARIMA models - 25 Mar 2014

ge from Data at Scale

Resources

Don't bing or google 'forecasting' – instead search on methods (time series, prediction markets, ARIMA, ETS, etc.)



THE ROCKS present forecasting

introduction to forecasting

- Introduction
- Regression
- Time Series
- Qualitative
- Accuracy
- The Rocks

Forecasting is a prediction of what will occur in the future. It is an uncertain process that is vital to survival in today's international business environment. Rapid technological advances have given consumers greater product diversity as well as more information on which they make their product choices. Managers try to forecast with as much accuracy as possible, but that is becoming increasingly difficult in today's fast-paced business world.

Forecast Methods

There are two types of forecasting methods. These methods help to provide reliable guidelines for decision making, but cannot provide totally accurate results.

Qualitative methods are based on:

- judgement
- opinion
- past experience
- best guesses

Quantitative methods are based on:

- mathematical methods
- two traditional types
  - time series
  - regression

Resources

R

Forecasting in R
R zoo package, useful for dates
R forecast package
A Little R Time Series Book

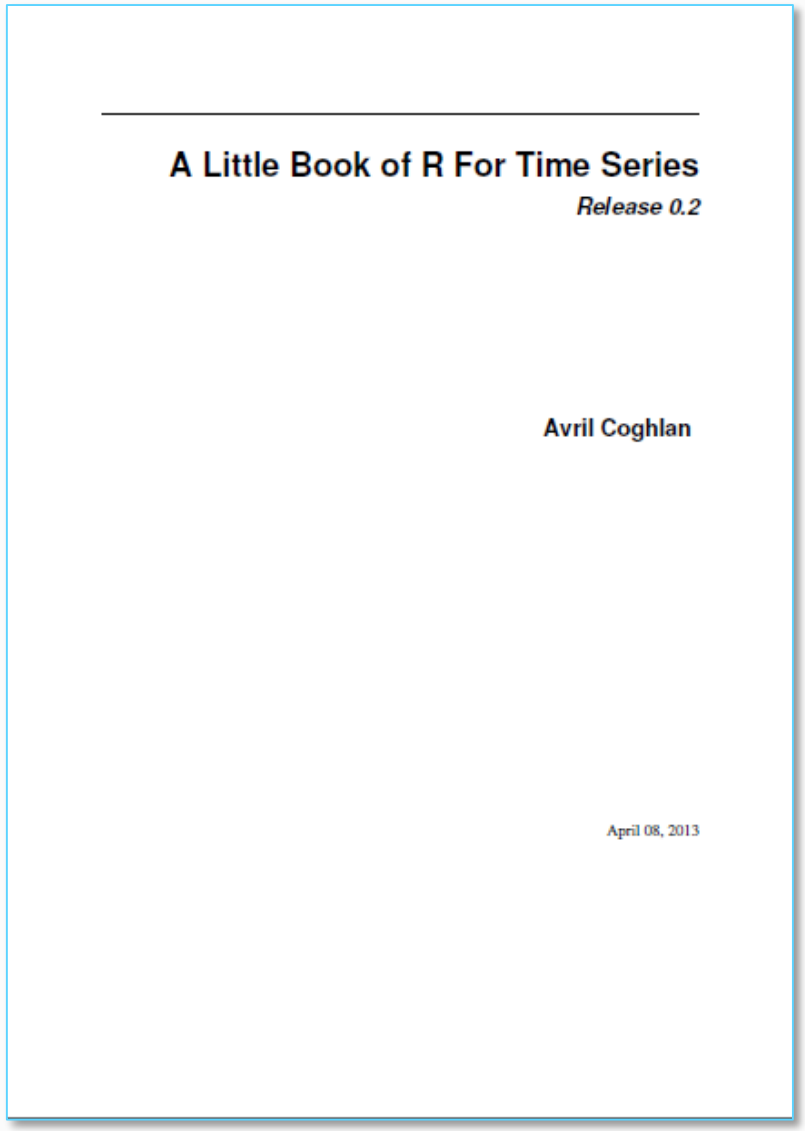
Python/Pandas

Time Series in Pandas
Time Series in Pandas (Video1, Video2, Video3)
statsmodels

Texts

Forecasting: Principles and Practice
Signal and the Noise: Why So Many Predictions Fail — but Some Don't

Out of Class Reading, *optional but very helpful…*



A Little Book of R For Time Series

Release 0.2

Avril Coghlan

April 08, 2013

Introduction to R's time series facilities

Michael Lundholm*

September 22, 2011
Version 1.3

1 Introduction

This is an introduction to R's time series facilities. In an elementary way we deal with reading time series data into R, how time series objects are defined and their properties and finally how time series data can be manipulated. This is *not* an introduction to time series analysis. The note is written as a complement to my colleague Mahmood Arai's "A Brief Guide to R for Beginners in Econometrics" available at `http://people.su.se/~ma/R_intro/R_Brief_Guide.pdf`. It is presumed that R is installed and that the reader has some basic R–knowledge. An interactive R–session run parallel to reading the notes is recommended.

2 Preliminaries

Time series data have the property of being temporally ordered. Each individual observation have a date and these dates are organised sequentially. We consider here only the case when the time interval between any two observations with sequentially following dates is constant (i.e., the same between all observations). This means that we have a time index $t \in \{1, 2, \ldots, T-1, T\}$, where 1 is the first date and T is the date of the last observation (i.e., the length of the time series). The number of observations per year in a time series is called its *frequency*.

The basic function in R that defines time series is `ts()`. It has as its

Lecture 3 Outline, Oct 19th

- Opening Discussion
- Forecasting, continued (2/2)
- Break
- Introducing Weka
- Decision Trees
- Hands On, Decision Tree in Weka

WEKA only deals with "flat" files

```
@relation heart-disease-simplified

@attribute age numeric
@attribute sex { female, male}
@attribute chest_pain_type { typ_angina, asympt, non_anginal, atyp_angina}
@attribute cholesterol numeric
@attribute exercise_induced_angina { no, yes}
@attribute class { present, not_present}

@data
63,male,typ_angina,233,no,not_present
67,male,asympt,286,yes,present
67,male,asympt,229,yes,present
38,female,non_anginal,?,no,not_present
...
```

Flat file in ARFF format

http://weka.wikispaces.com/ARFF+(stable+version)

WEKA only deals with “flat” files

```
@relation heart-disease-simplified

@attribute age numeric
@attribute sex { female, male}
@attribute chest_pain_type { typ_angina, asympt, non_anginal, atyp_angina}
@attribute cholesterol numeric
@attribute exercise_induced_angina { no, yes}
@attribute class { present, not_present}

@data
63,male,typ_angina,233,no,not_present
67,male,asympt,286,yes,present
67,male,asympt,229,yes,present
38,female,non_anginal,?,no,not_present
...
```

Explorer: pre-processing the data

- Data can be imported from a file in various formats: ARFF, CSV, Excel, binary
- Data can also be read from a URL or from an SQL database (using JDBC)
- Pre-processing tools in WEKA are called “filters”
- WEKA contains filters for:
  - Discretization, normalization, resampling, attribute selection, transforming and combining attributes, …

Explorer: building "classifiers"

- Classifiers in WEKA are models for predicting nominal or numeric quantities
- Implemented learning schemes include:
  - Decision trees and lists, instance-based classifiers, support vector machines, multi-layer perceptrons, logistic regression, Bayes' nets, …
- "Meta"-classifiers include:
  - Bagging, boosting, stacking, error-correcting output codes, locally weighted learning, …

Overview of Fisher's Iris Dataset

Fisher's iris dataset is well-known in data mining research.

This dataset is commonly used to illustrate data mining tools.

Fisher's Database - Background

iris setosa

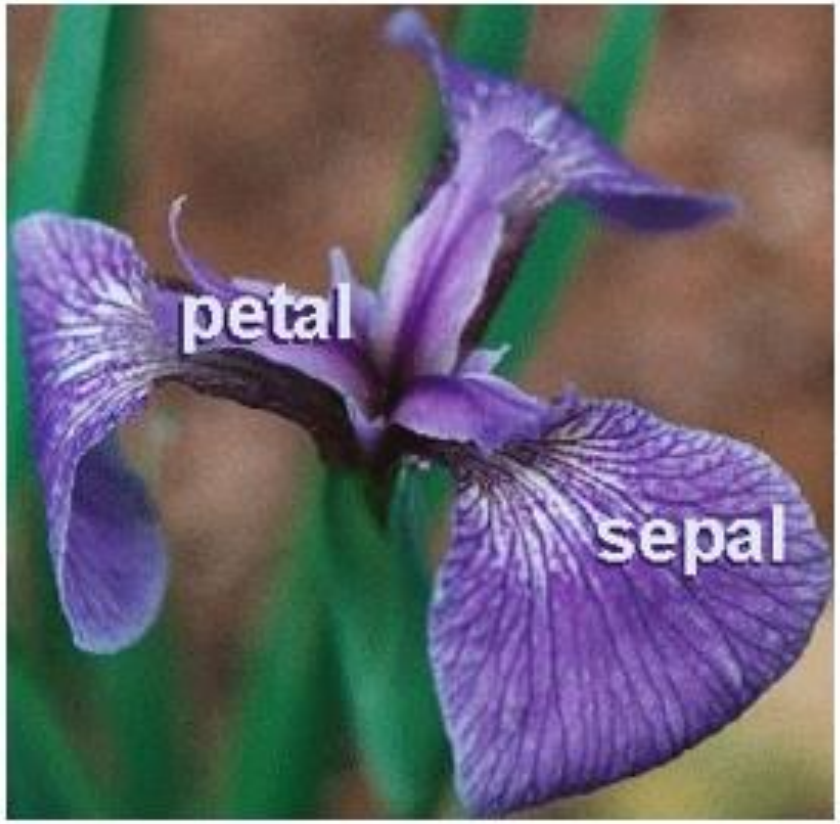
iris versicolor

iris virginica

Dataset contains flower dimension measurements on 50 samples of each species.

Fisher, R.A. (1936). "The Use of Multiple Measurements in Taxonomic Problems". *Annals of Eugenics* **7**: 179–188, available at: http://digital.library.adelaide.edu.au/coll/special//fisher/138.pdf.

Fisher's Dataset- Background



Anderson measured these dimensions:

- sepal length
- sepal width
- petal length
- petal width

Measurements on these iris species:

- setosa
- versicolor
- virginica

Data mining terminology

- The four iris dimensions are termed *attributes*, or *input attributes*.
- The three iris species are termed *classes*, or *output attributes*.
- Each example of an iris is termed a *sample*, or *instance*.

Segment of Fisher's Iris Dataset

Sample

Class

	Input Attributes				Output Attribute
Inst.	Sepal Length	Sepal Width	Petal Length	Petal Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5	3.6	1.4	0.2	setosa

Numerical

Nominal

What Problem are We Trying to Solve?

?

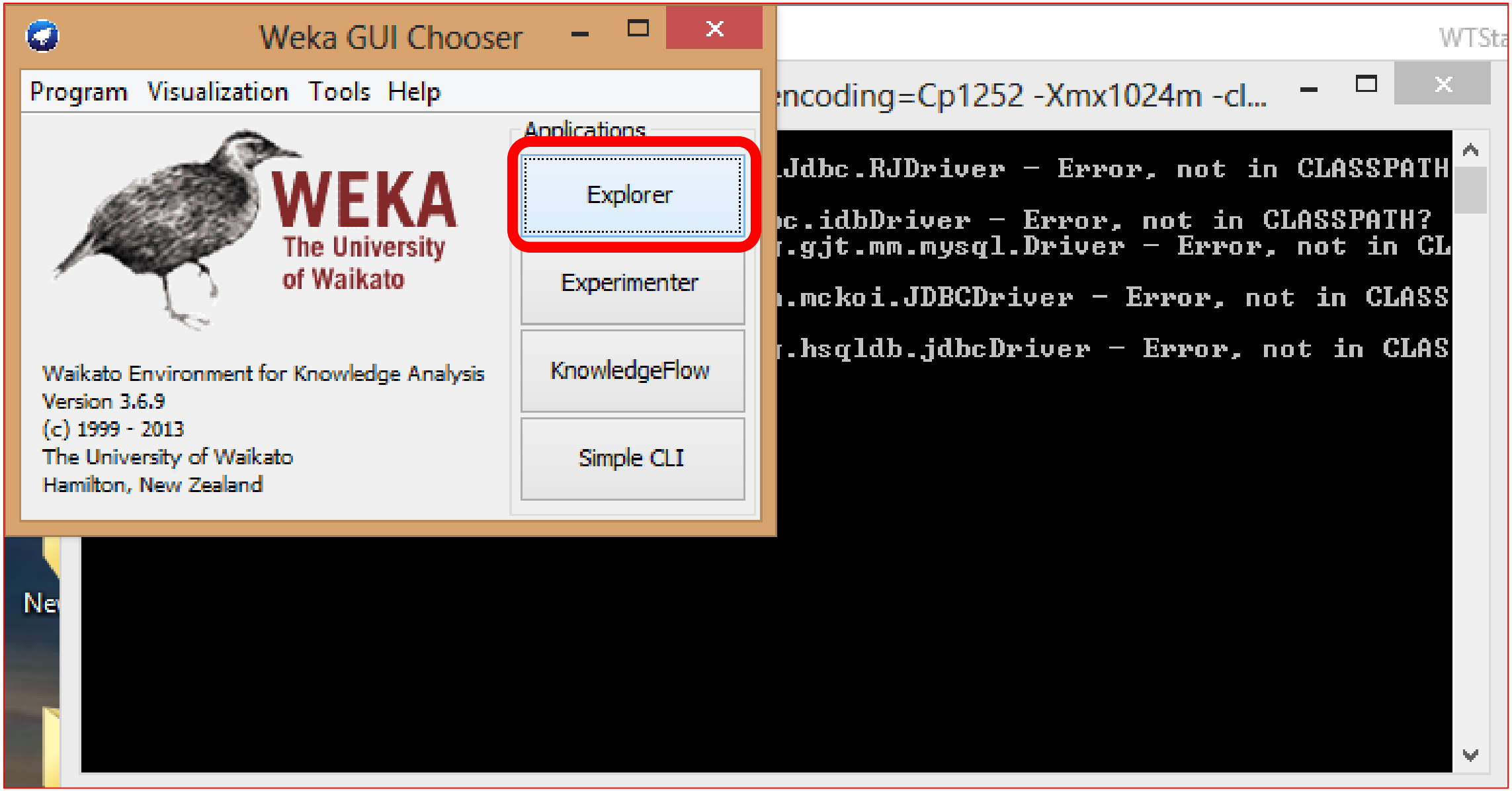
Weka decision tree

Our task:
Classify unknown iris using measurements on 150 iris samples of known species.

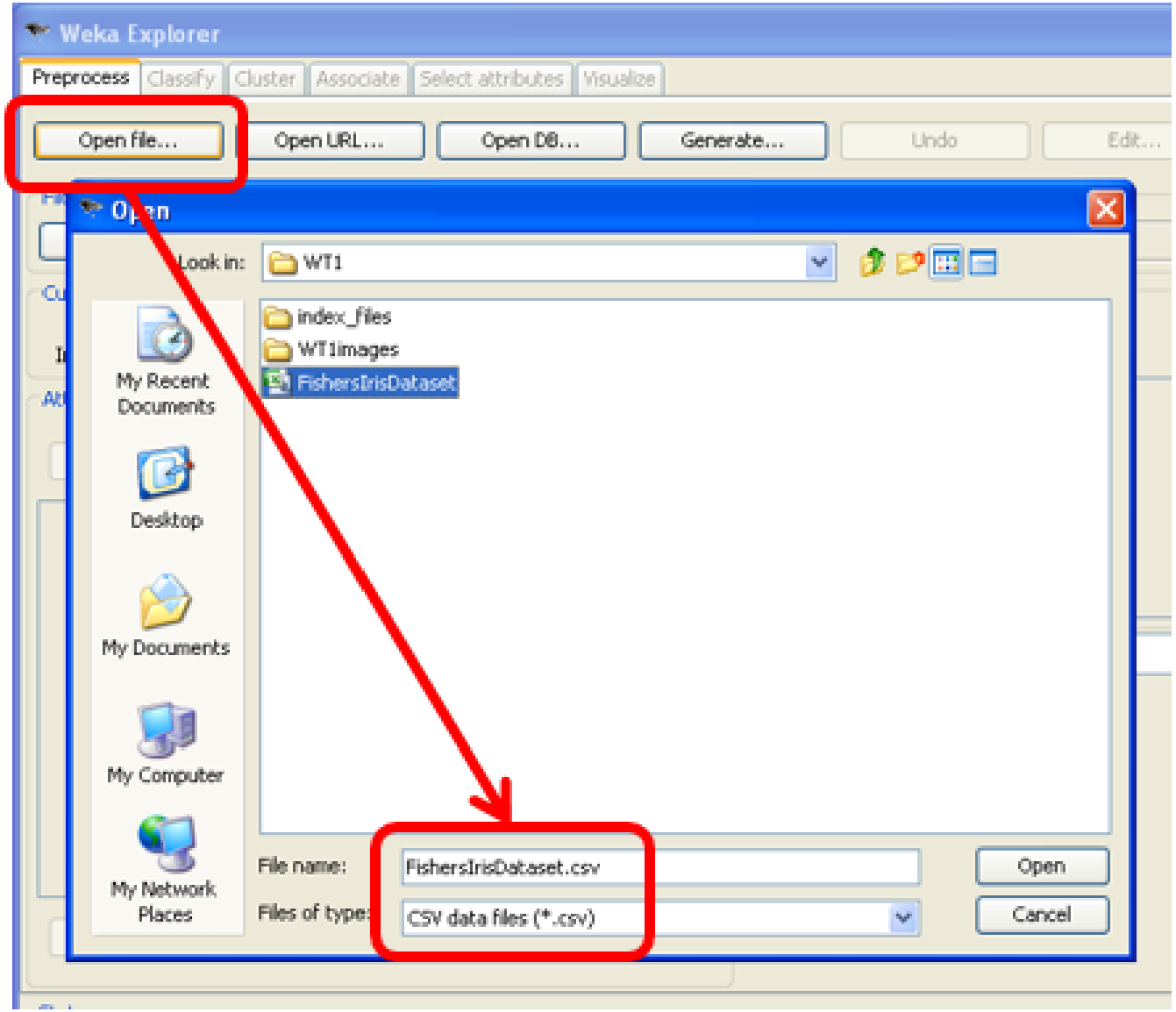
Iris setosa

Iris versicolor

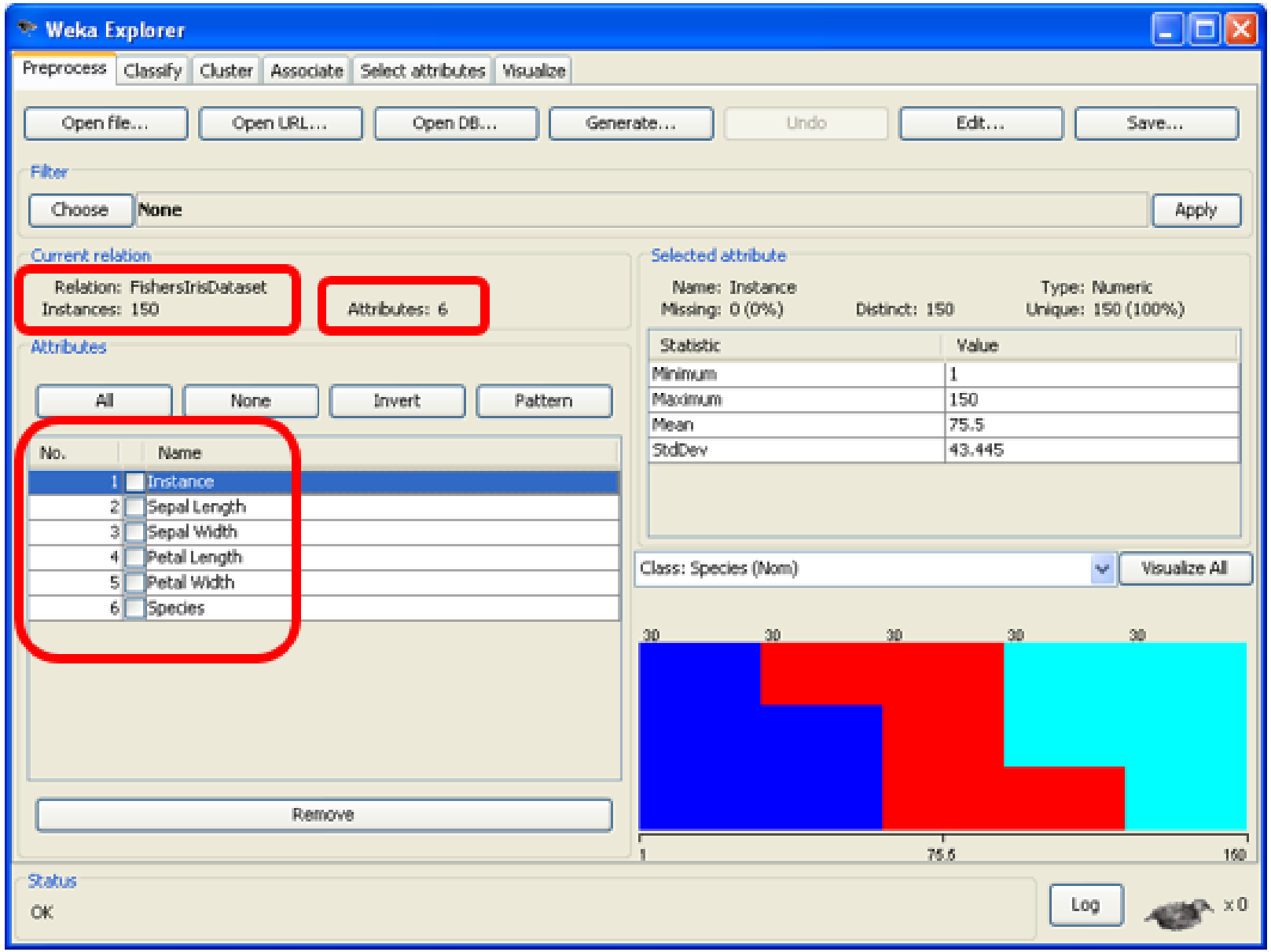
Iris virginica


Weka GUI Chooser
Program Visualization Tools Help
WEKA
The University of Waikato
Applications
Explorer
Experimenter
KnowledgeFlow
Simple CLI
Waikato Environment for Knowledge Analysis
Version 3.6.9
(c) 1999 - 2013
The University of Waikato
Hamilton, New Zealand
encoding=Cp1252 -Xmx1024m -cl...
Jdbc.RJDriver - Error, not in CLASSPATH
c.idbDriver - Error, not in CLASSPATH?
.gjt.mm.mysql.Driver - Error, not in CL
.mckoi.JDBCDriver - Error, not in CLASS
.hsqldb.jdbcDriver - Error, not in CLAS

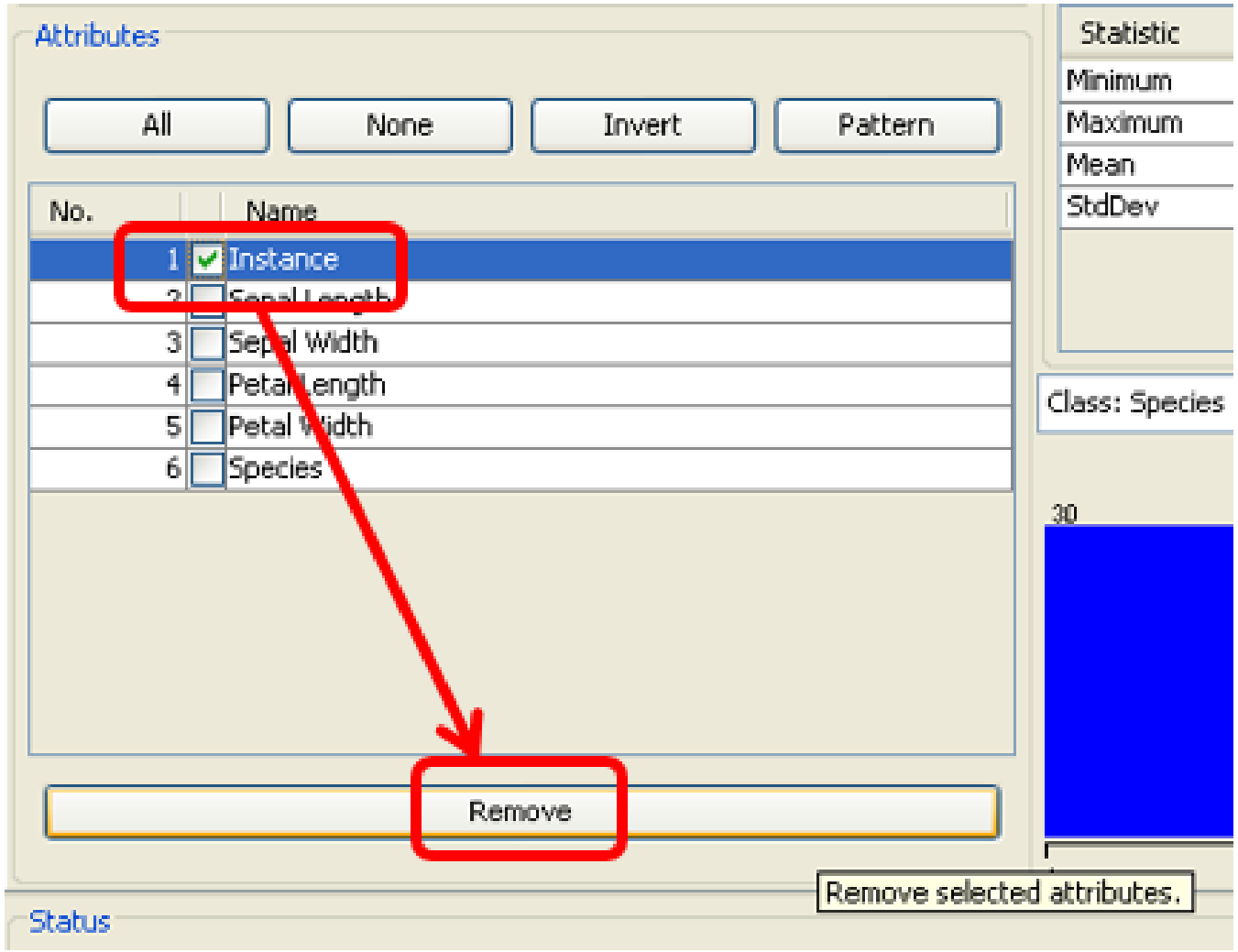
Open the Fisher's iris dataset .csv file



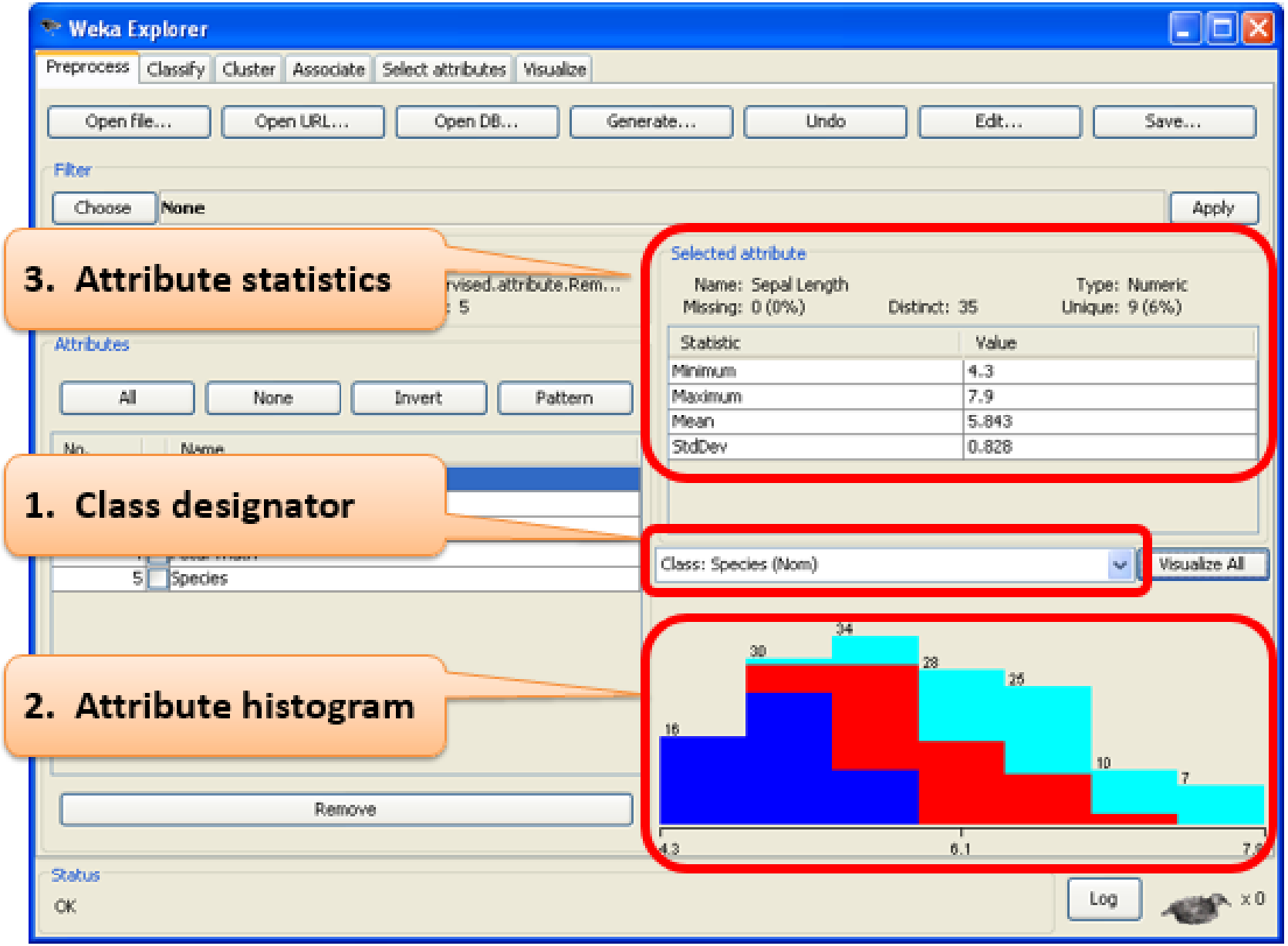
Dataset Has Successfully Opened



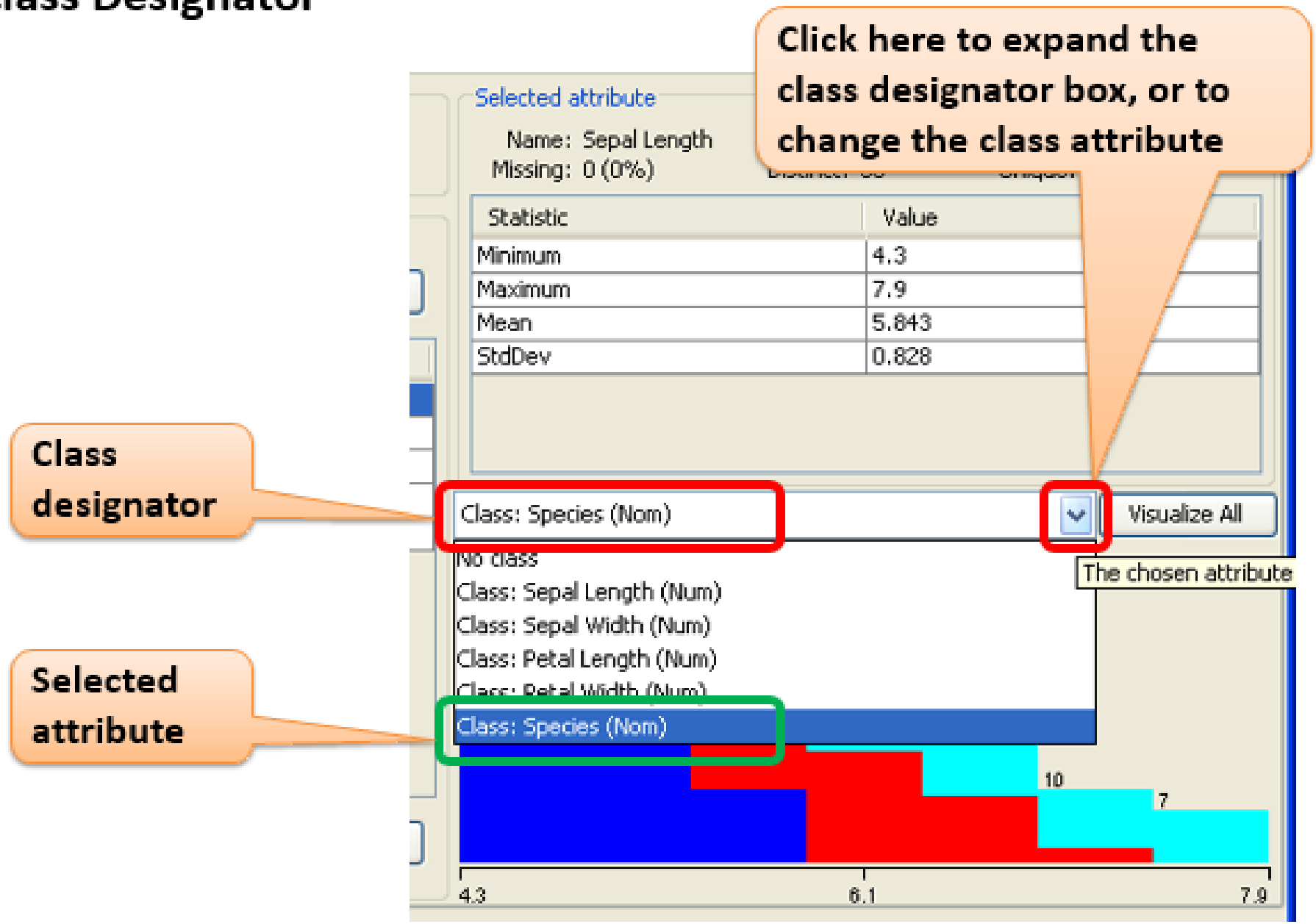
Remove "Instance" Attribute



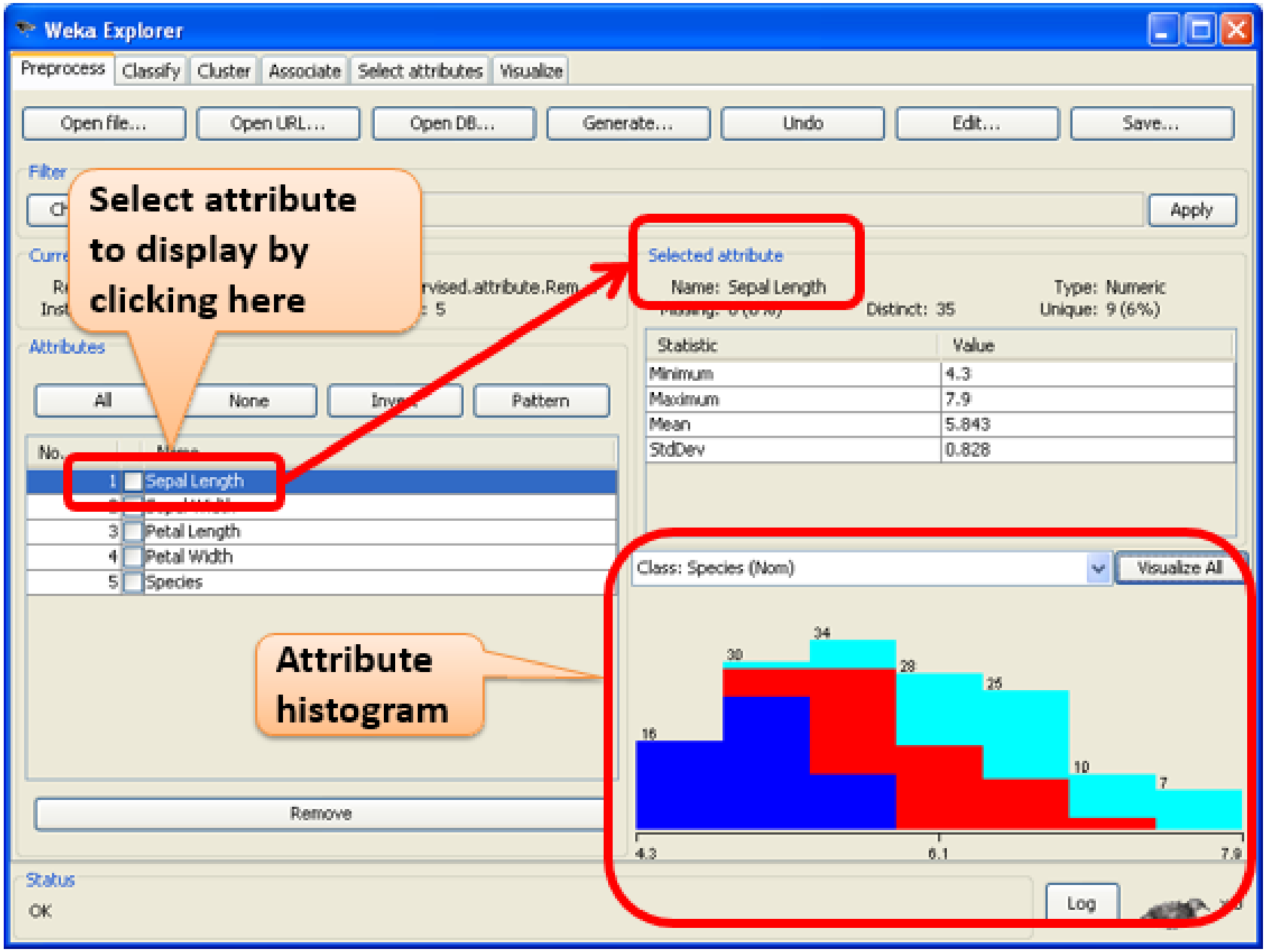
Elements of the Explorer Screen



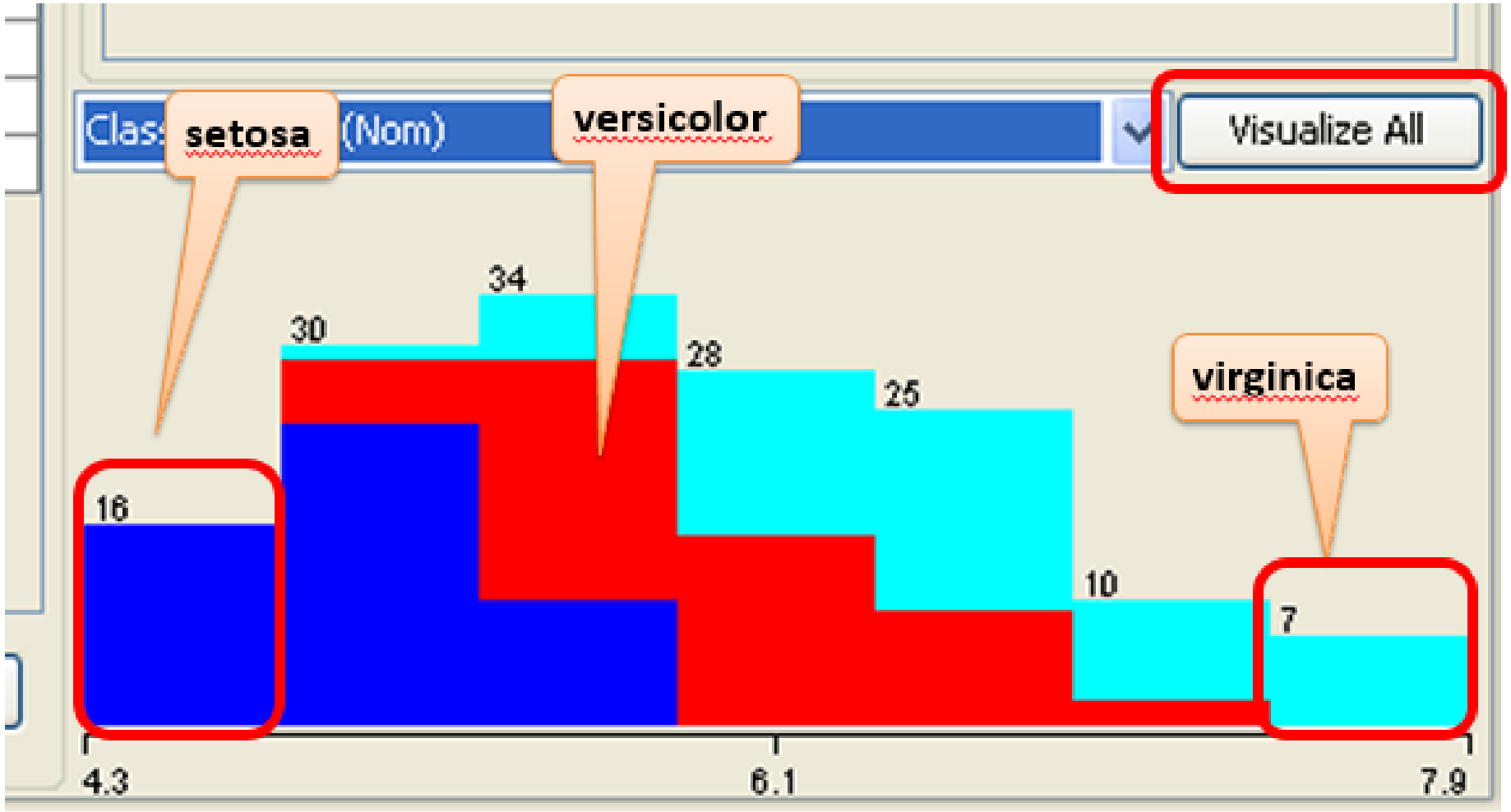
Class Designator



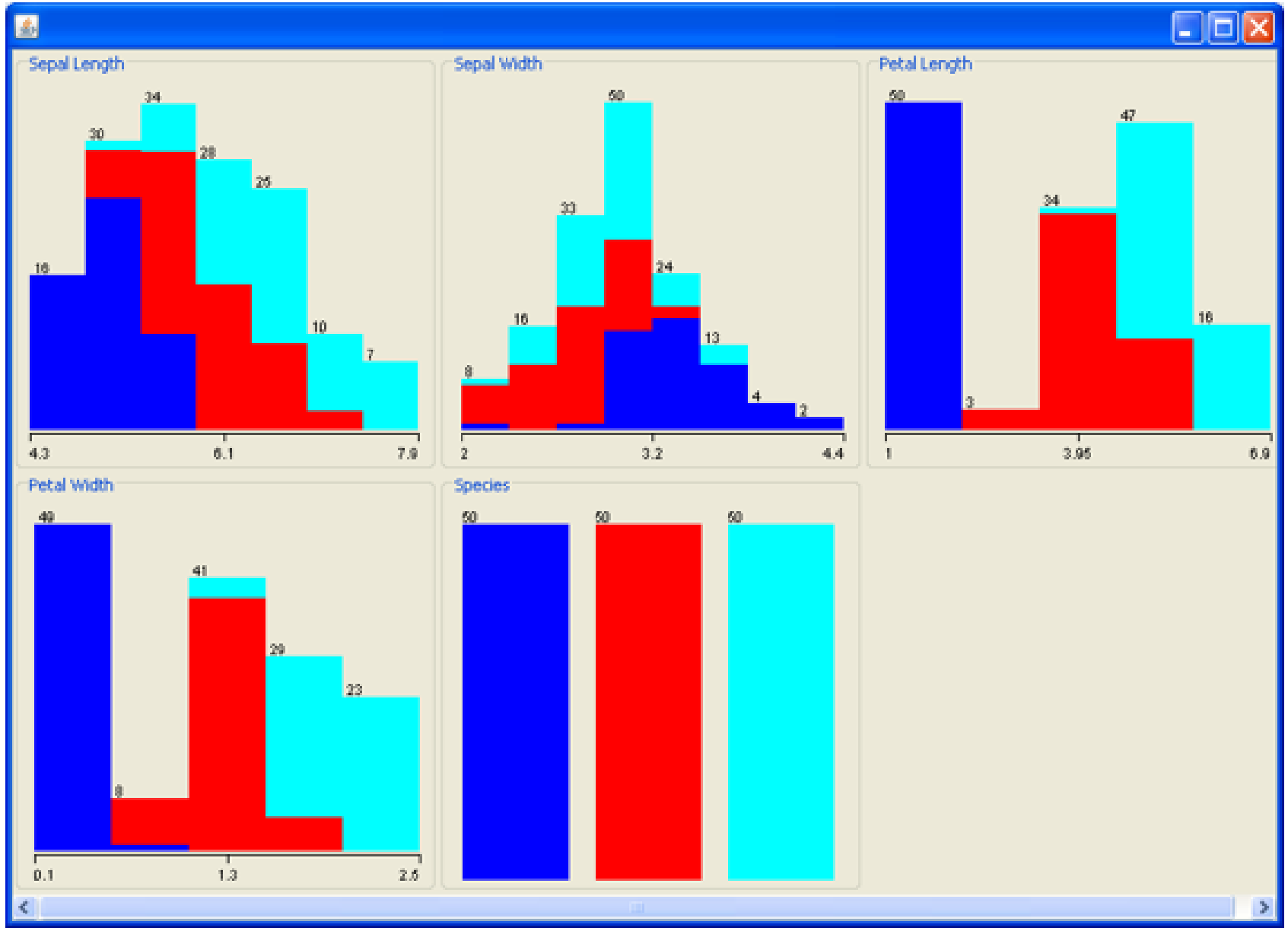
Attribute Histogram



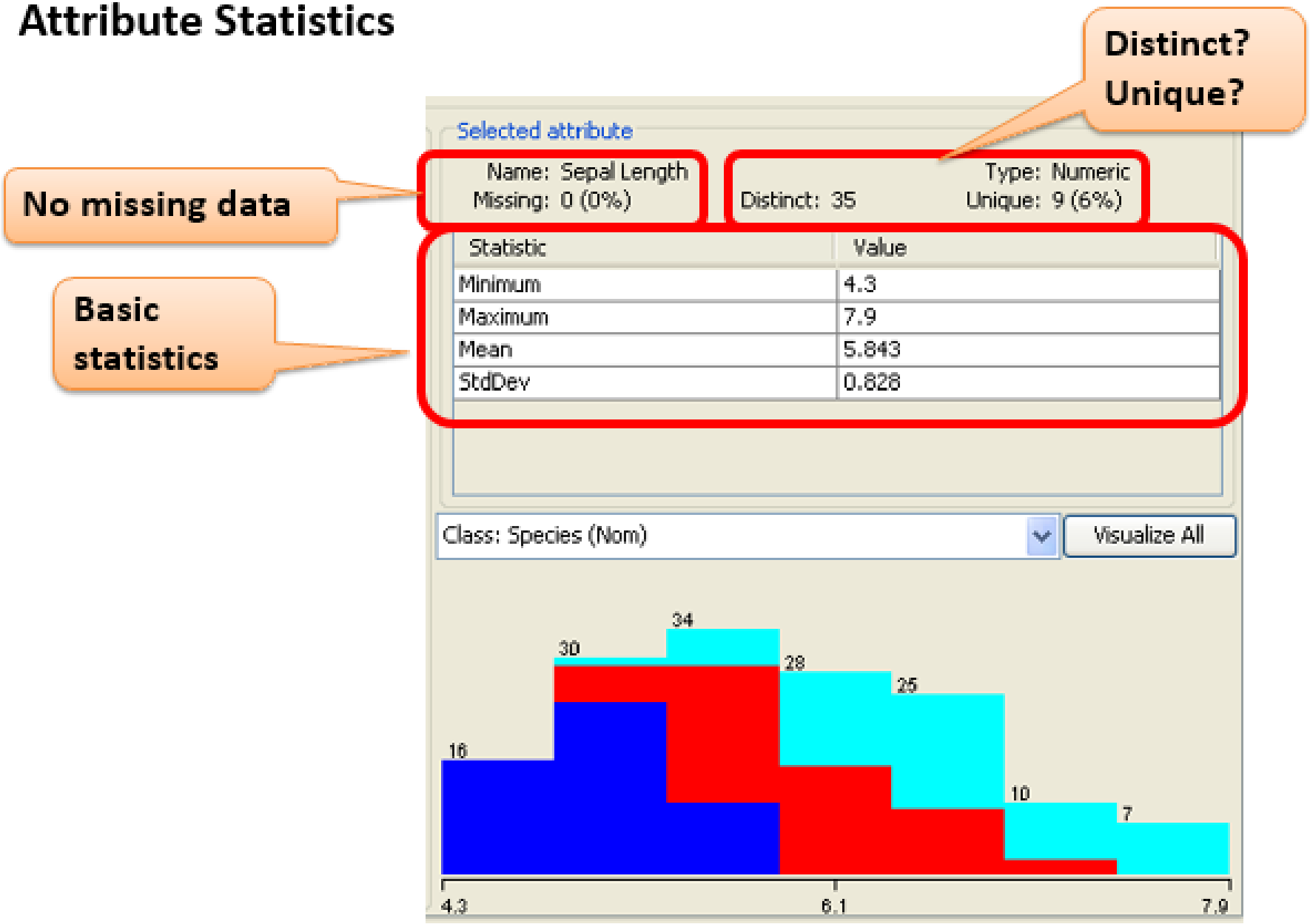
Attribute Histogram



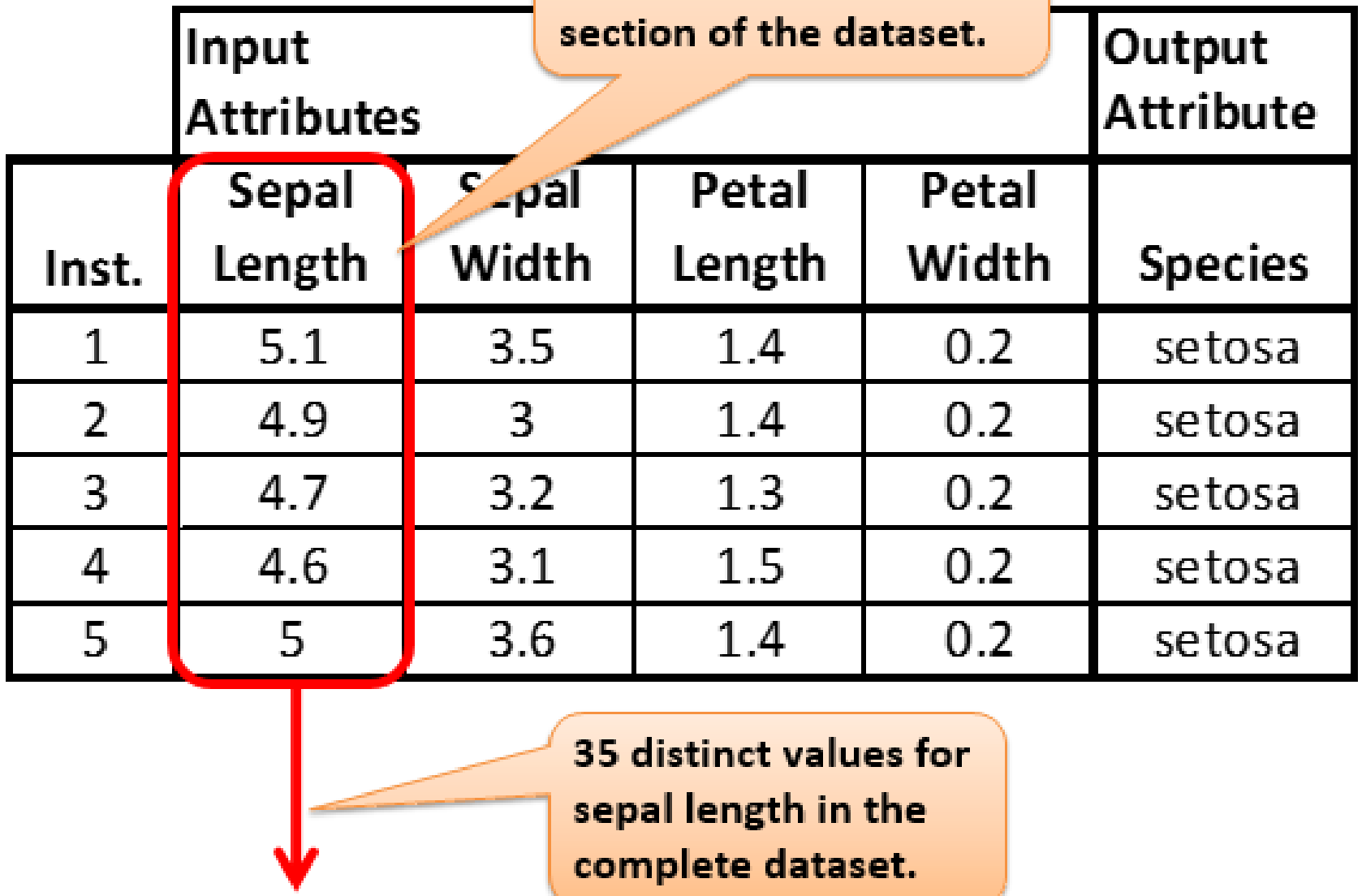
Attribute Histograms



Attribute Statistics



Distinct Values



	Input Attributes				Output Attribute
Inst.	Sepal Length	Sepal Width	Petal Length	Petal Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5	3.6	1.4	0.2	setosa

Unique Values

Instance	Sepal Length	Sepal Width	Petal Length	Petal Width	Species
14	4.3	3	1.1	0.1	setosa
9	4.4	2.9	1.4	0.2	setosa
39	4.4	3	1.3	0.2	setosa
43	4.4	3.2	1.3	0.2	setosa
42	4.5	2.3	1.3	0.3	setosa
4	4.6	3.1	1.5	0.2	setosa
7	4.6	3.4	1.4	0.3	setosa

Only one sample with sepal length of this value.

Total of nine samples with unique value of sepal length.

Lecture 3 Outline, Oct 19th

- Opening Discussion
- Forecasting, continued (2/2)
- Break
- Introducing Weka
- Decision Trees
- Hands On, Decision Tree in Weka

Decision Trees

- Easy to use
- Can represent any Boolean Function (indicator variable)
- Can be viewed as a way to compactly represent a lot of data
- Advantage: non-numeric data
- Natural representation: (20 questions) http://www.20q.net/

- The evaluation of the Decision Tree Classifier is easy
- Given data, there are many ways to represent it as a decision tree.
- Susceptible to Overfitting but can be avoided…

Classification, Regression and Clustering

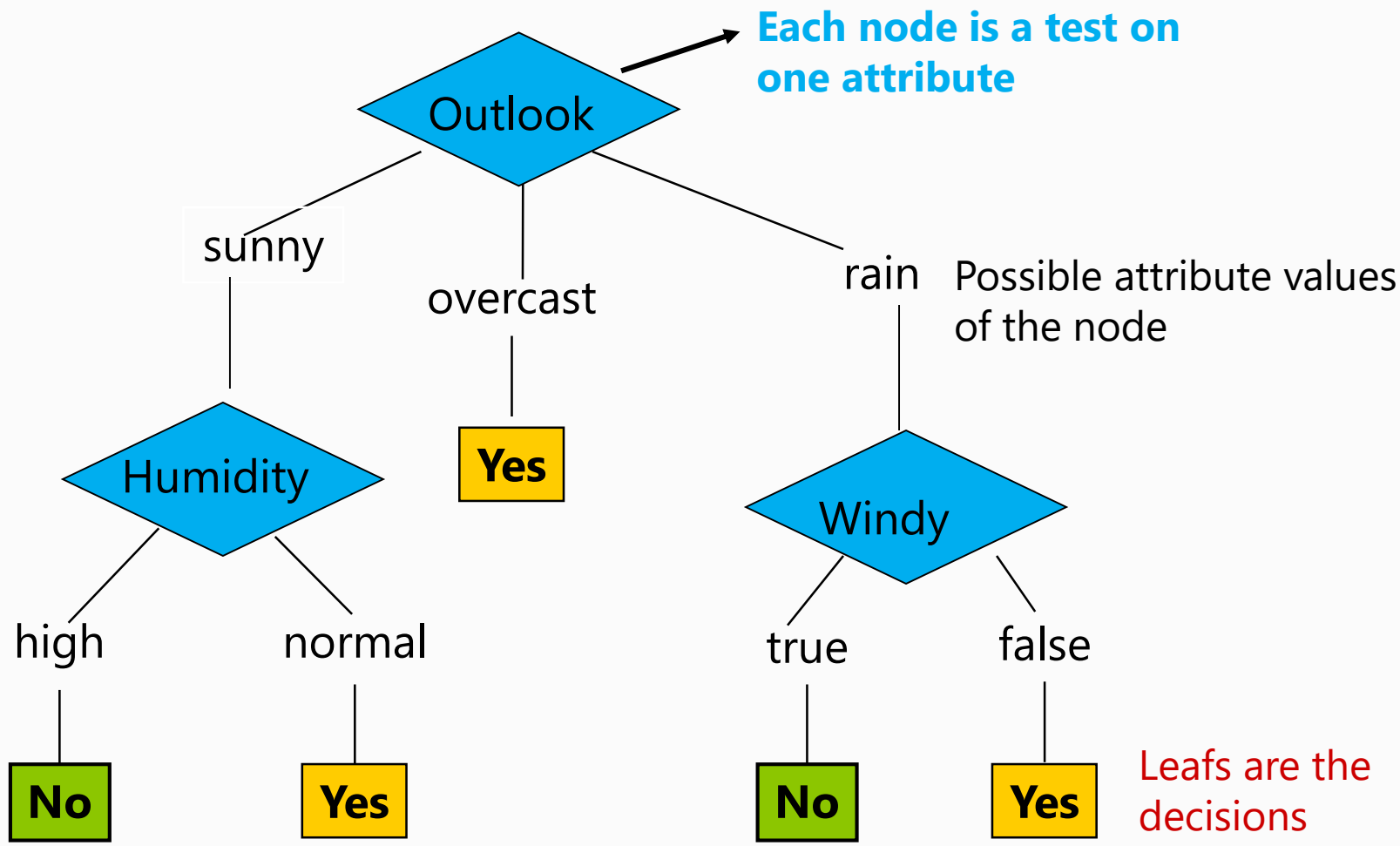
- **Classification** trees represent function C~T(X) where C discrete
  Hence, can be used for concept learning

- **Regression** trees predict numbers in leaves; Y~T(X)
  - Can use a constant (e.g., mean), or linear regression model, or …

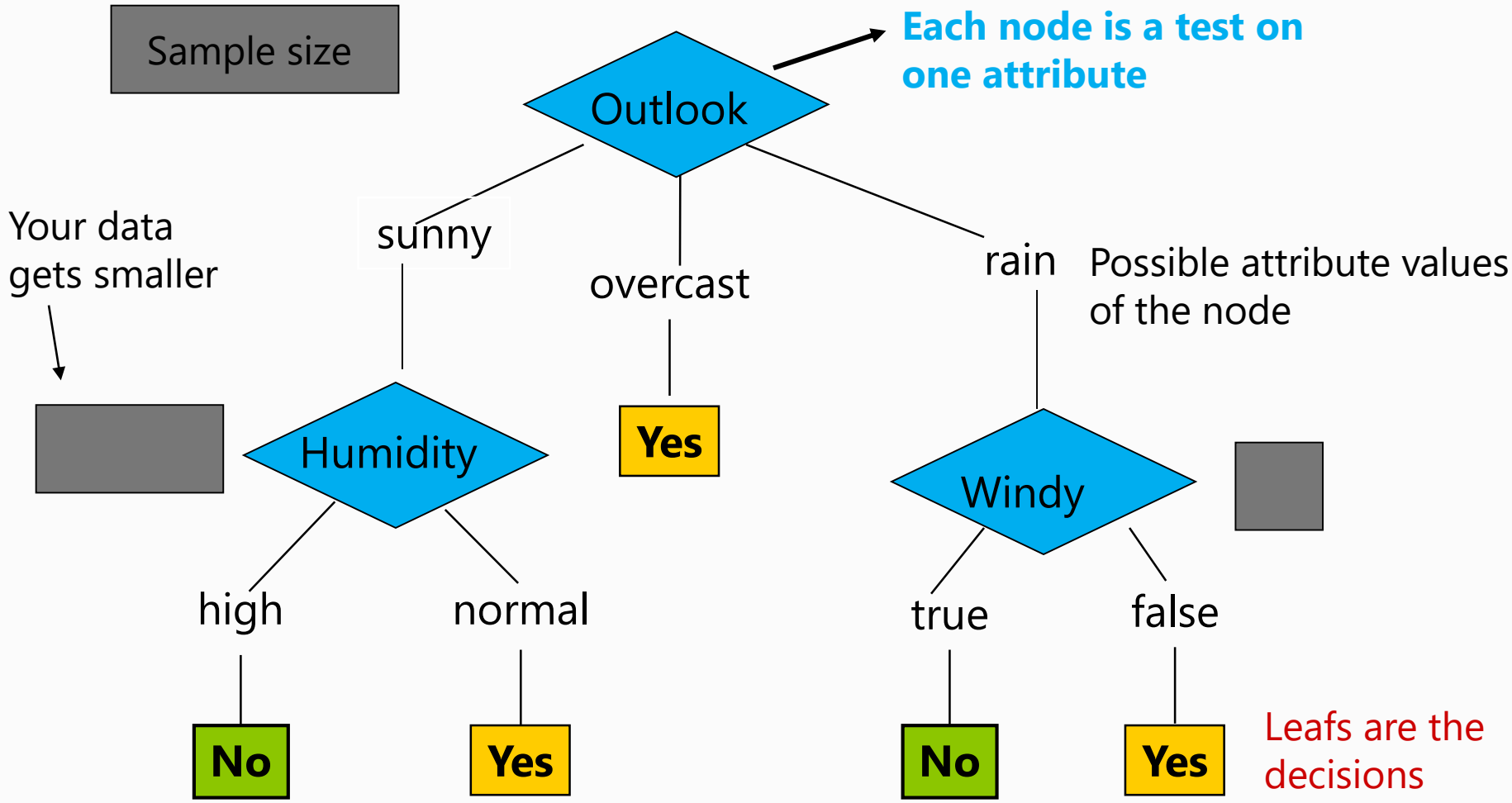
- **Clustering** trees just group data in leaf node; combinations of attributes

Ref: *Classification and Regression Trees* (1984) Breiman, Friedman, Stone, and Olshen.

Anatomy of a decision tree

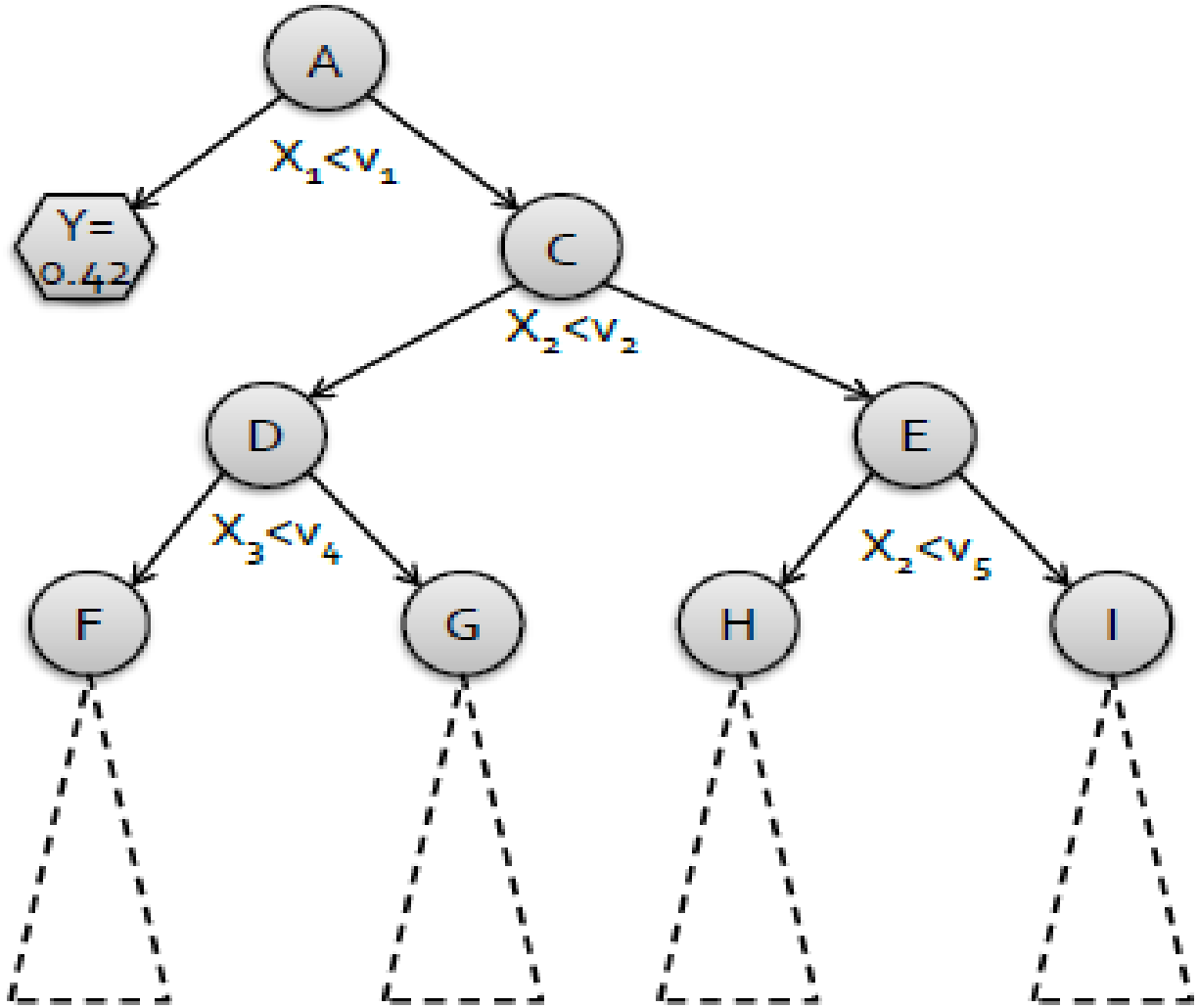


Anatomy of a decision tree

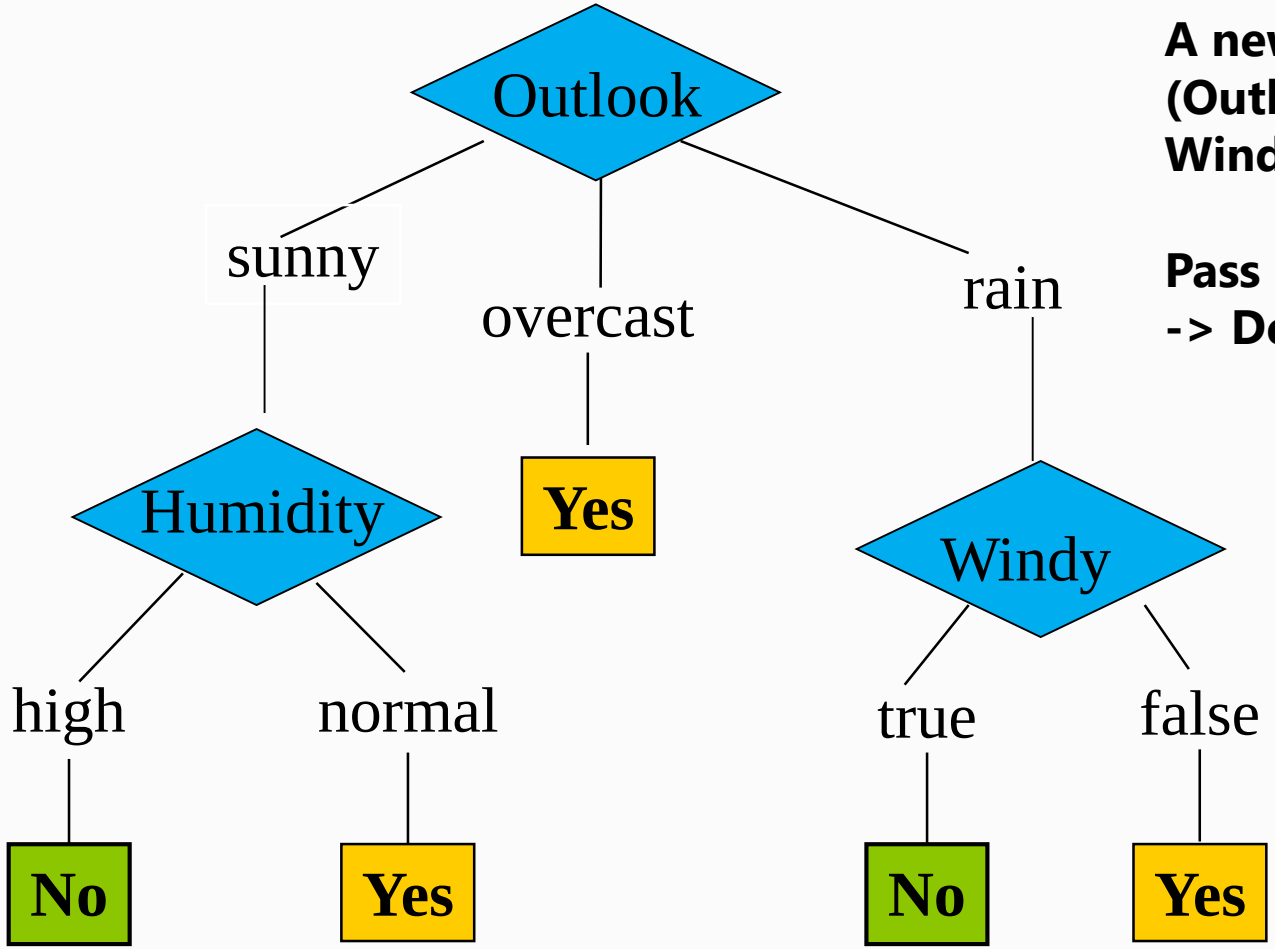


How to make predictions?

- Input: Example x_i
- Output: Predicted y_i'
- "Drop" x_i down the tree until it hits a leaf node
- Predict the value stored in the leaf that x_i hits



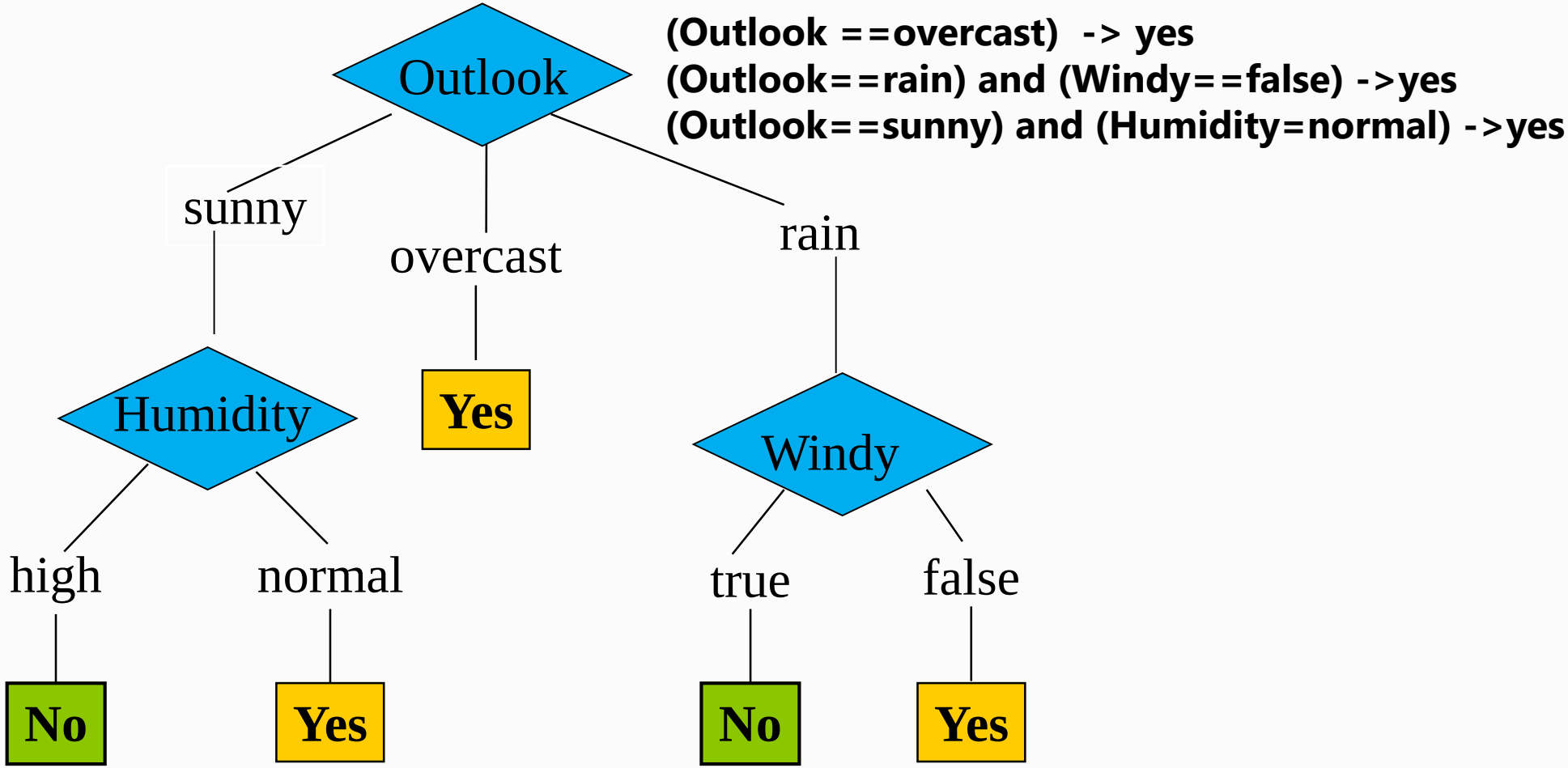
To ‘play tennis’ or not.



A new test example:
(Outlook==rain) and (not Windy==false)

Pass it on the tree
-> Decision is yes.

To ‘play tennis’ or not.



(Outlook ==overcast) -> yes
(Outlook==rain) and (Windy==false) ->yes
(Outlook==sunny) and (Humidity=normal) ->yes

Which attribute to select for splitting?

- The goal is to have the resulting decision tree as *small as possible* (Occam's Razor)
- Finding the minimal decision tree consistent with the data is NP-hard
- Recursive algorithm is a greedy heuristic search for a simple tree, but cannot guarantee optimality.
- Select attributes that split the examples to sets that are relatively pure in one label; this way we are closer to a leaf node.

Top-Down Induction of Decision Trees

Basic algorithm for TDIDT: *(based on ID3; later more formal)*

1. start with full data set
2. find test that partitions examples as good as possible
   = examples with same class, or otherwise similar, are put together
3. for each outcome of test, create child node
4. move examples to children according to outcome of test
5. repeat procedure for each child that is not "pure"

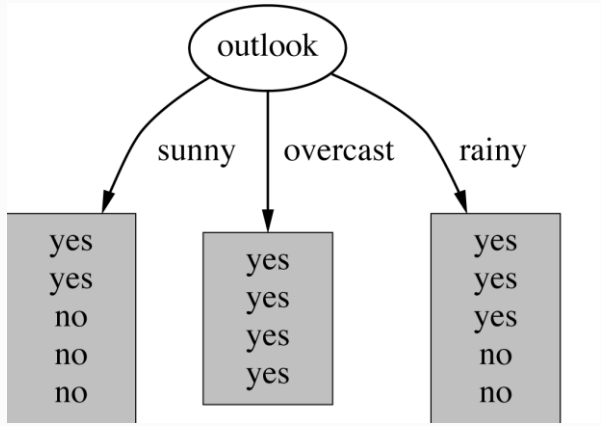
Main questions:

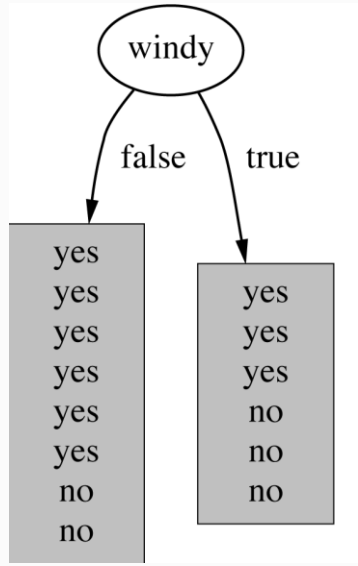
- how to decide which test is "best"
- when to stop the procedure (Overfitting)

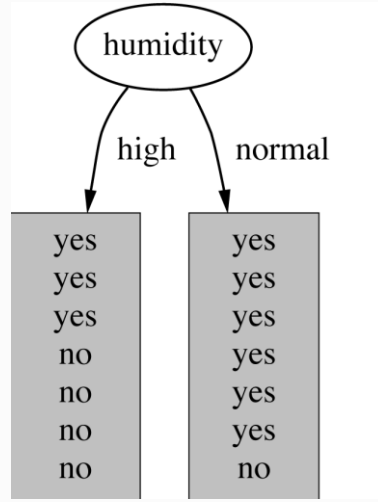
How do we choose the test ?

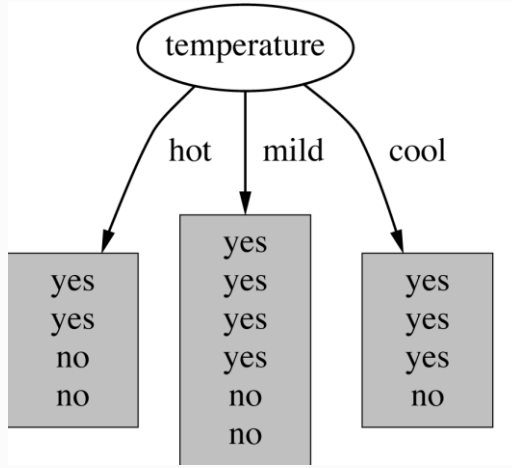
Which attribute should be used as the test?

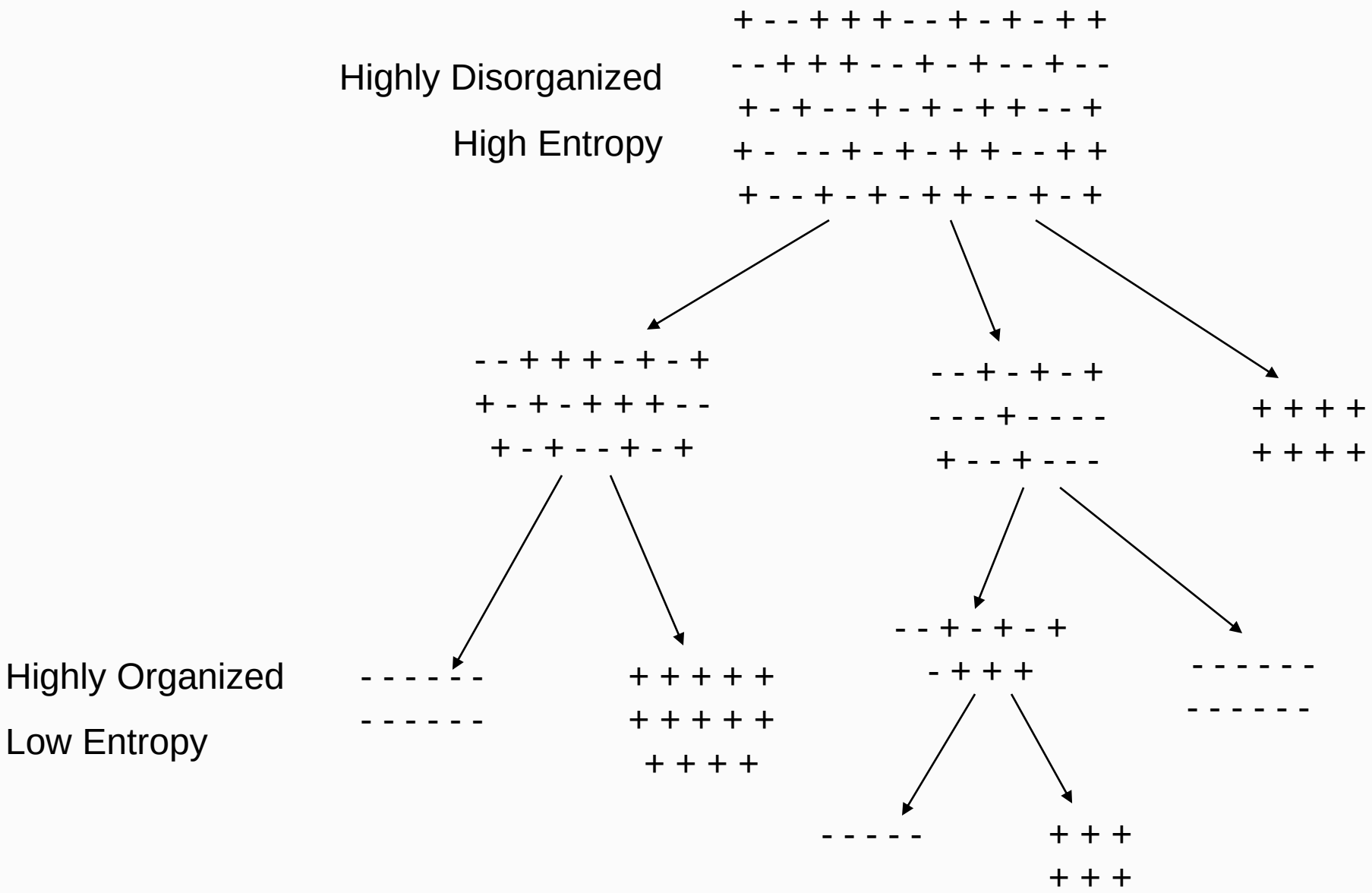
Intuitively, you would prefer the one that *separates* the training examples as much as possible, *reduces the entropy*…










Highly Disorganized
High Entropy
Highly Organized
Low Entropy

Information

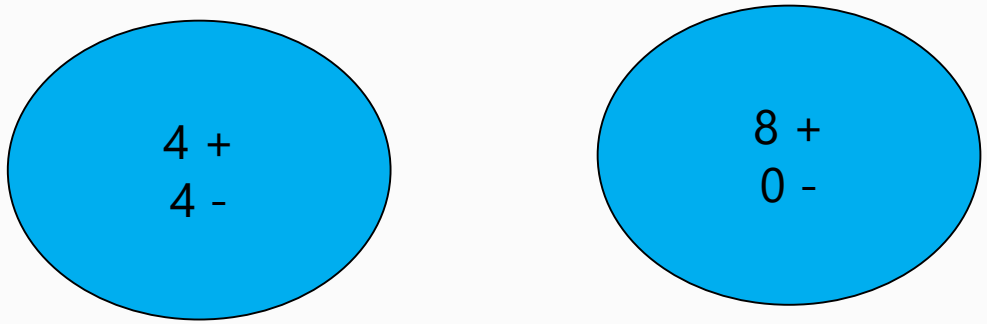
Imagine:

1. Someone is about to tell you your own name
2. You are about to observe the outcome of a dice roll
3. You are about to observe the outcome of a coin flip
4. You are about to observe the outcome of a biased coin flip

Each situation has a different *amount of uncertainty* as to what outcome you will observe.

Entropy, Purity

Entropy measures the purity



The distribution is less uniform
Entropy is lower
The node is purer

Information Gain

Information gain:

(information before split) – (information after split)

Information Gain

Heuristic for choosing a test in a node:

- Choose a test that on average *provides most information about the class*
- This is the test that, on average, *reduces class entropy most*
  - entropy reduction differs according to outcome of test
- Expected reduction of entropy = *information gain*

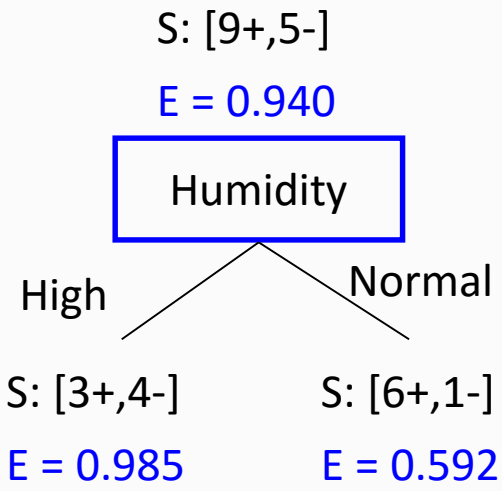
$$Gain(S, A) = E(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} E(S_v)$$

where $S_v = \{ s \in S \mid A(s) = v \}$

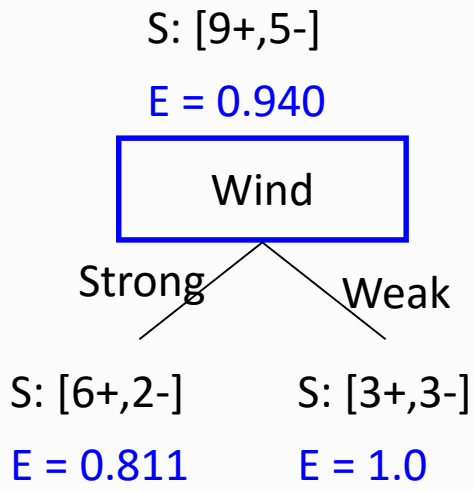
Example

- Assume S has 9 + and 5 - examples; partition according to Wind or **Humidity** attribute

$$E = -\sum_i p_i log_2(p_i)$$



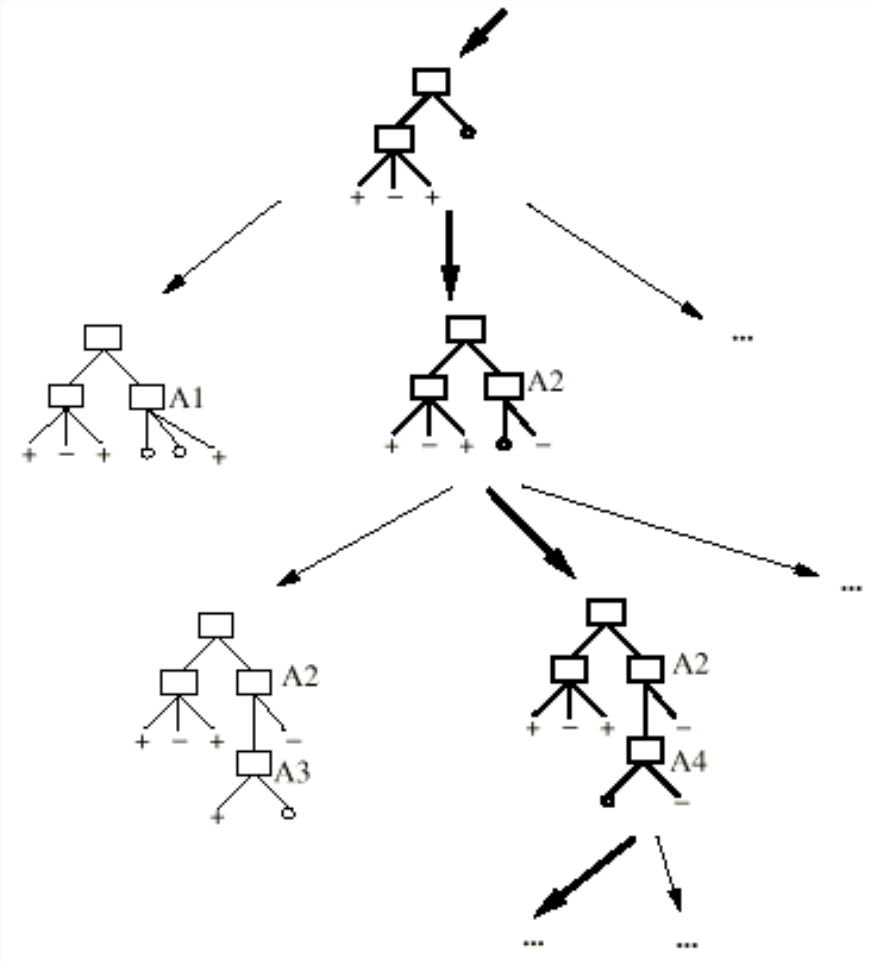
Gain(S, Humidity)
= .940 - (7/14).985 - (7/14).592
= 0.151



Gain(S, Wind)
= .940 - (8/14).811 - (6/14)1.0
= 0.048

Hypothesis space search in TDIDT

- Hypothesis space H = set of all trees
- H is searched in a hill-climbing fashion, from simple to complex
  - maintain a single tree
  - no backtracking



Overfitting

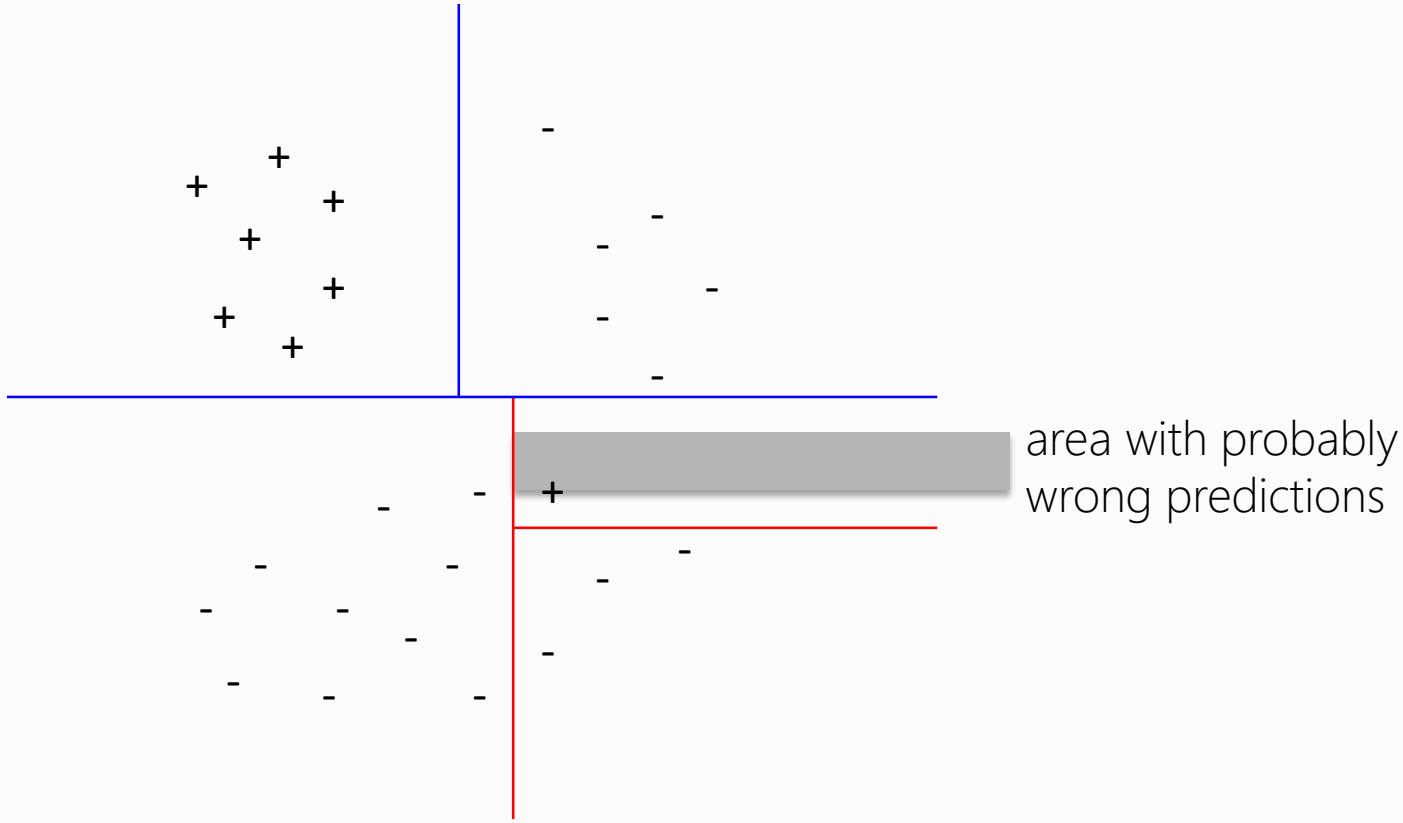
Larger the tree, more likely of Overfitting training data.

- You can perfectly fit to any training data
- Zero bias, high variance

Two approaches:

1. Stop growing the tree when further splitting the data does not yield an improvement (pre-set tolerance)
2. Grow a full tree, then prune the tree, by eliminating nodes.

Overfitting: Example

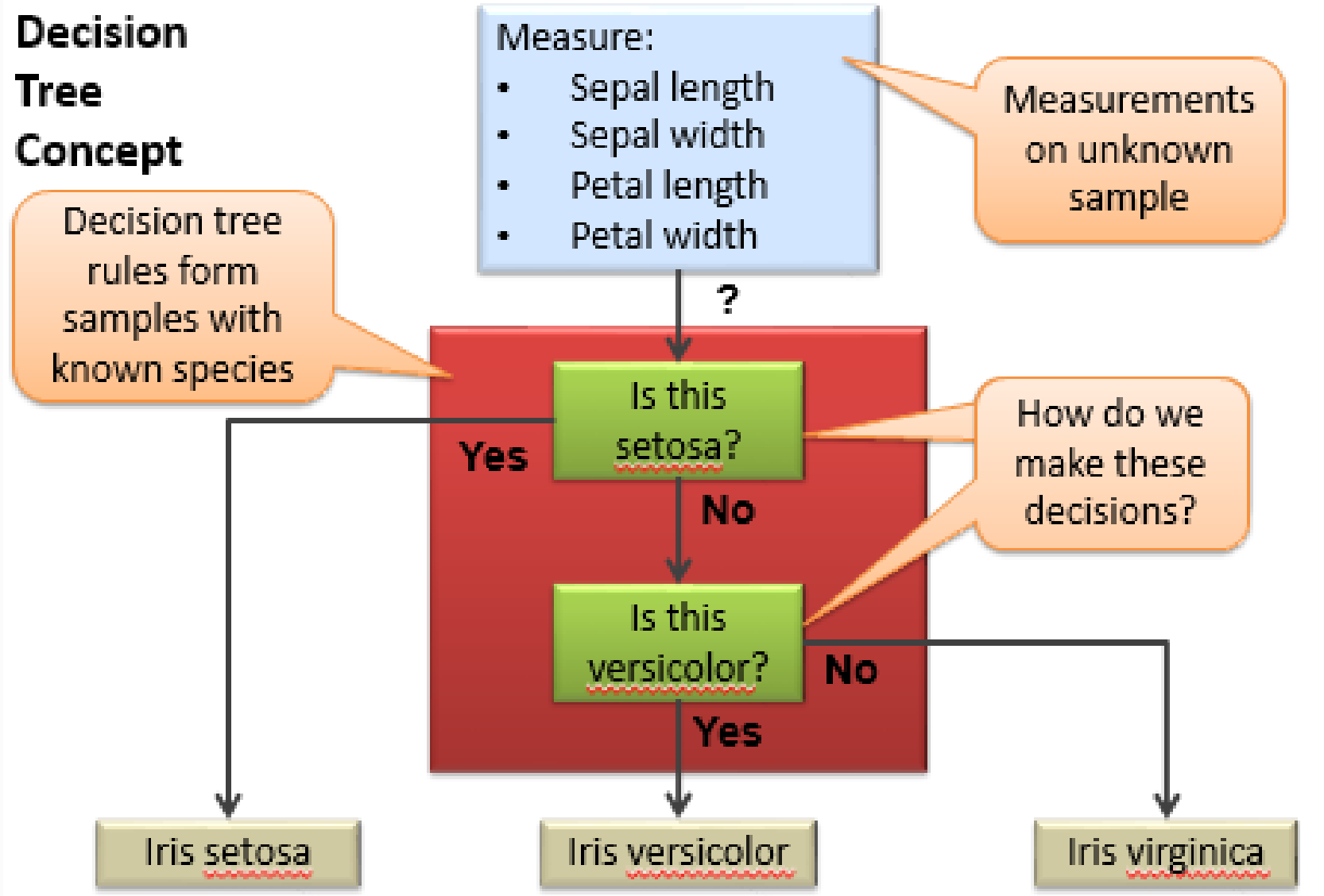


Break…

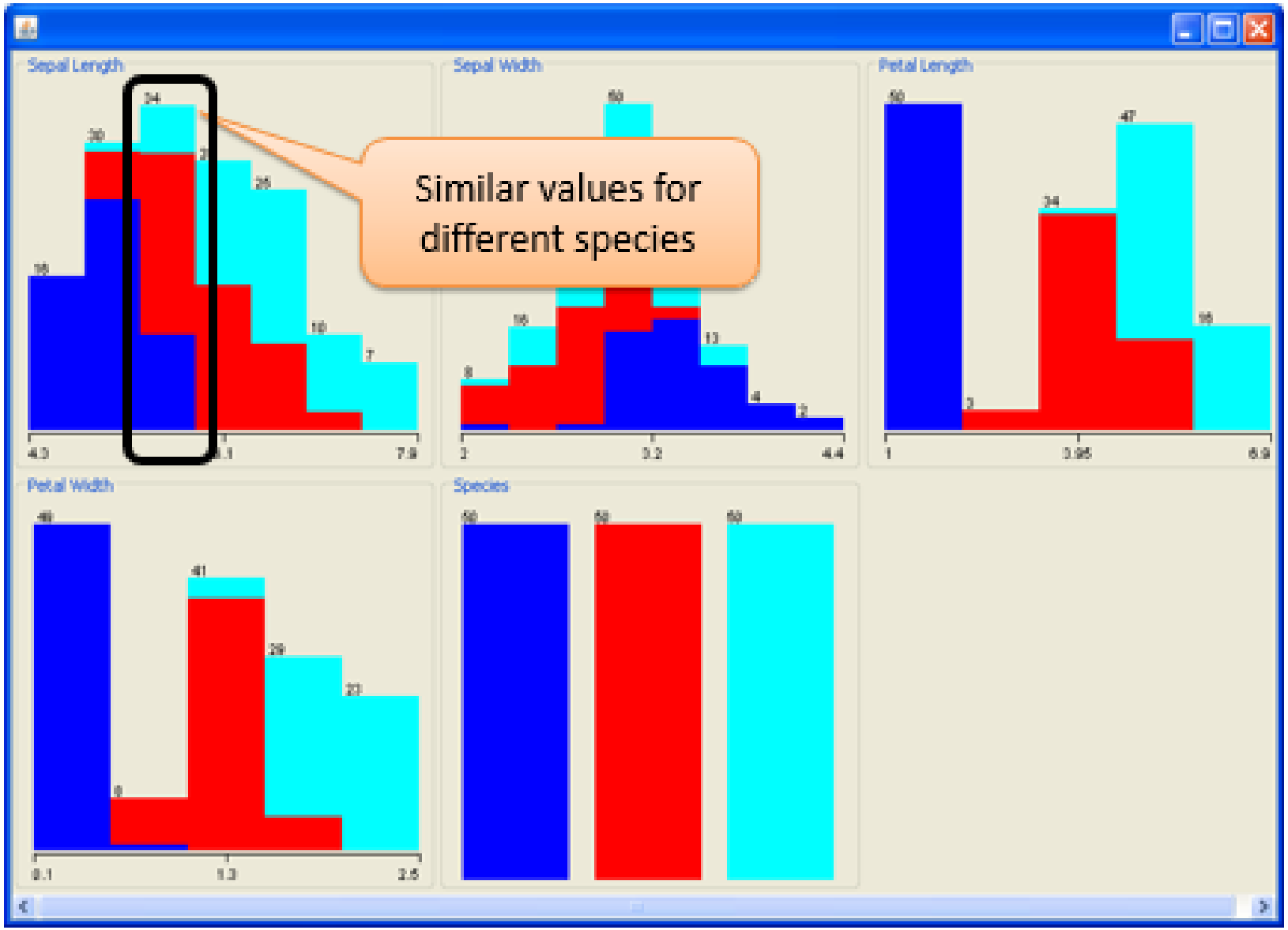
Lecture 3 Outline, Oct 19th

- Opening Discussion
- Forecasting, continued (2/2)
- Break
- Introducing Weka
- Decision Trees
- Hands On, Decision Tree in Weka

Decision Tree Concept



Decision Tree Uncertainty

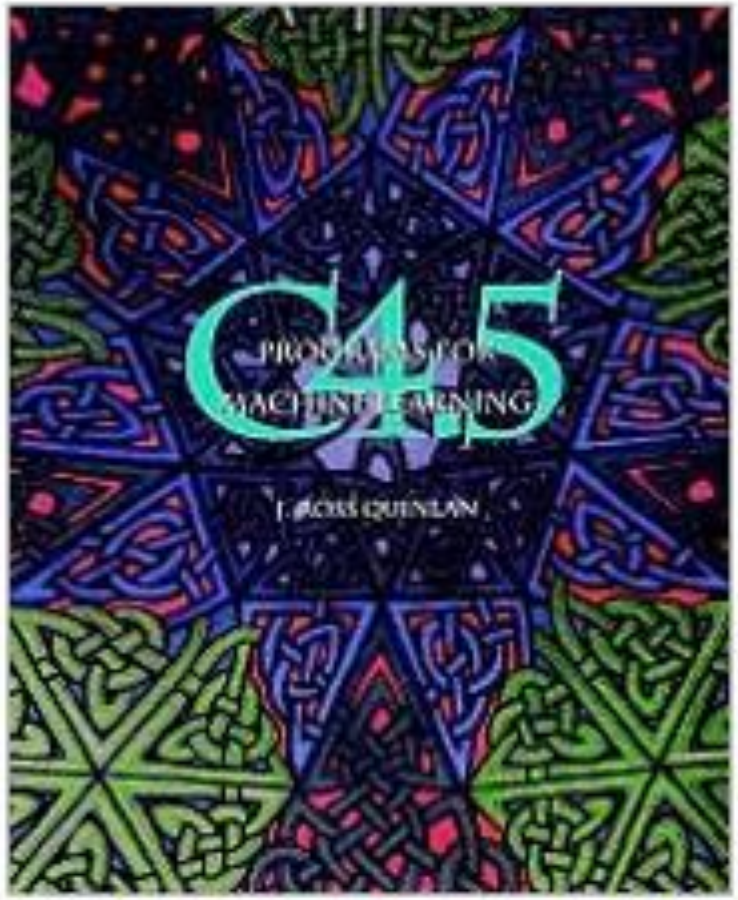


Decision Tree Uncertainty

Instance	Sepal Length	Sepal Width	Petal Length	Petal Width	Species
35	4.9	3.1	1.5	0.2	setosa
38	4.9	3.6	1.4	0.1	setosa
58	4.9	2.4	3.3	1	versicolor
107	4.9	2.5	4.5	1.7	virginica
5	5	3.6	1.4	0.2	setosa

What species is an iris with sepal length of 4.9 cm?

Weka J4.8 Decision Tree Learner

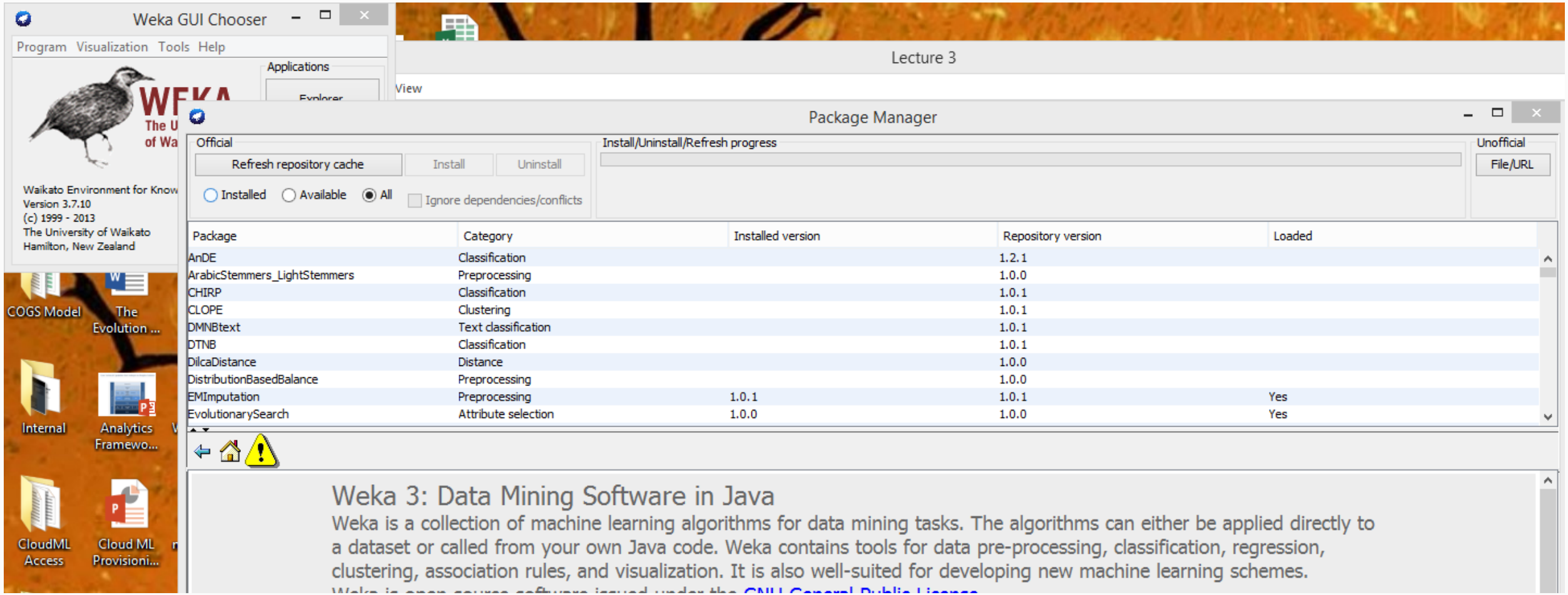


Quinlan, J.R.,
C4.5: Programs for Machine Learning,
San Francisco, Morgan Kaufmann, 1993

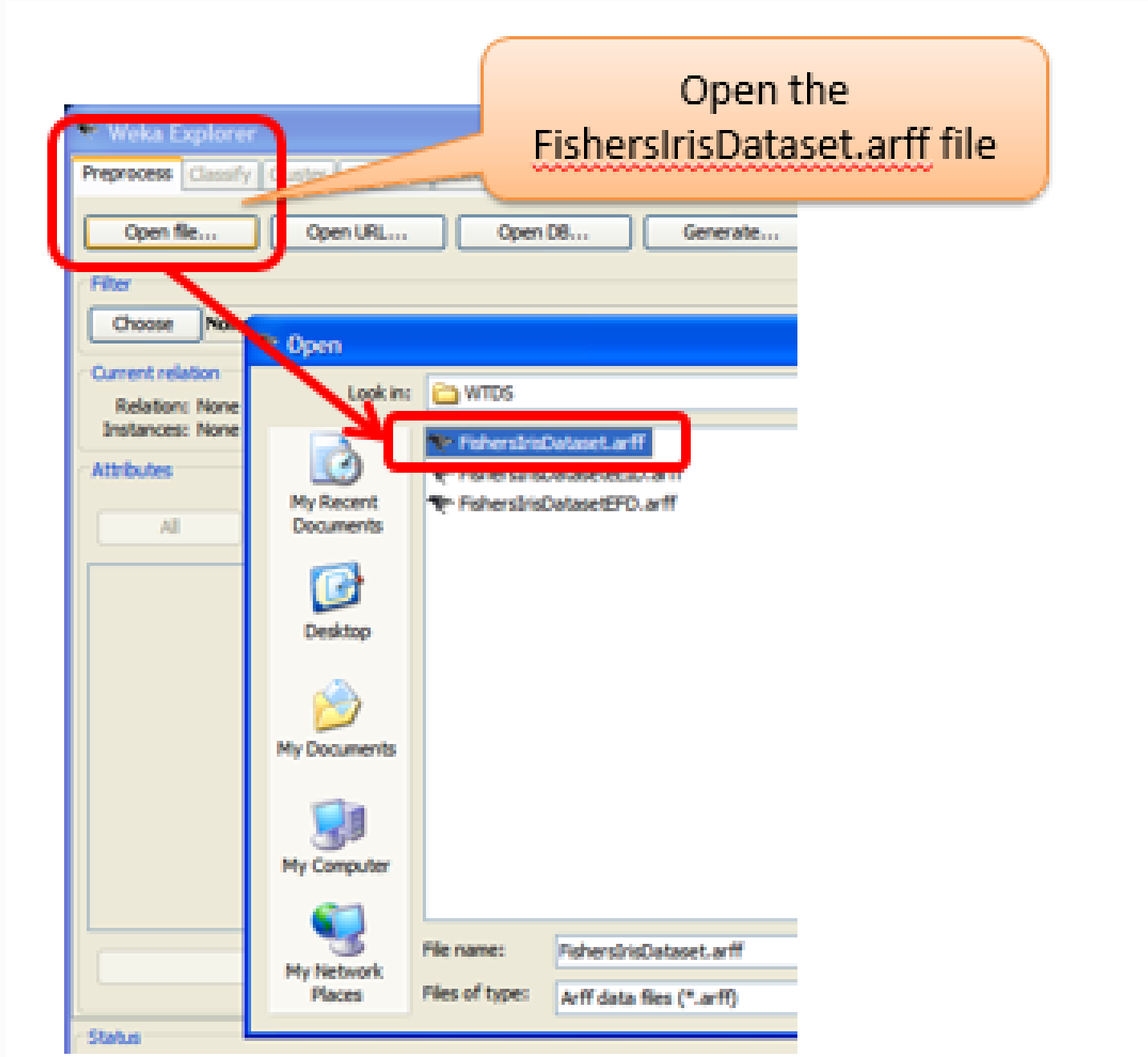
Our tool for this tutorial

Weka J4.8 Decision Tree Learner

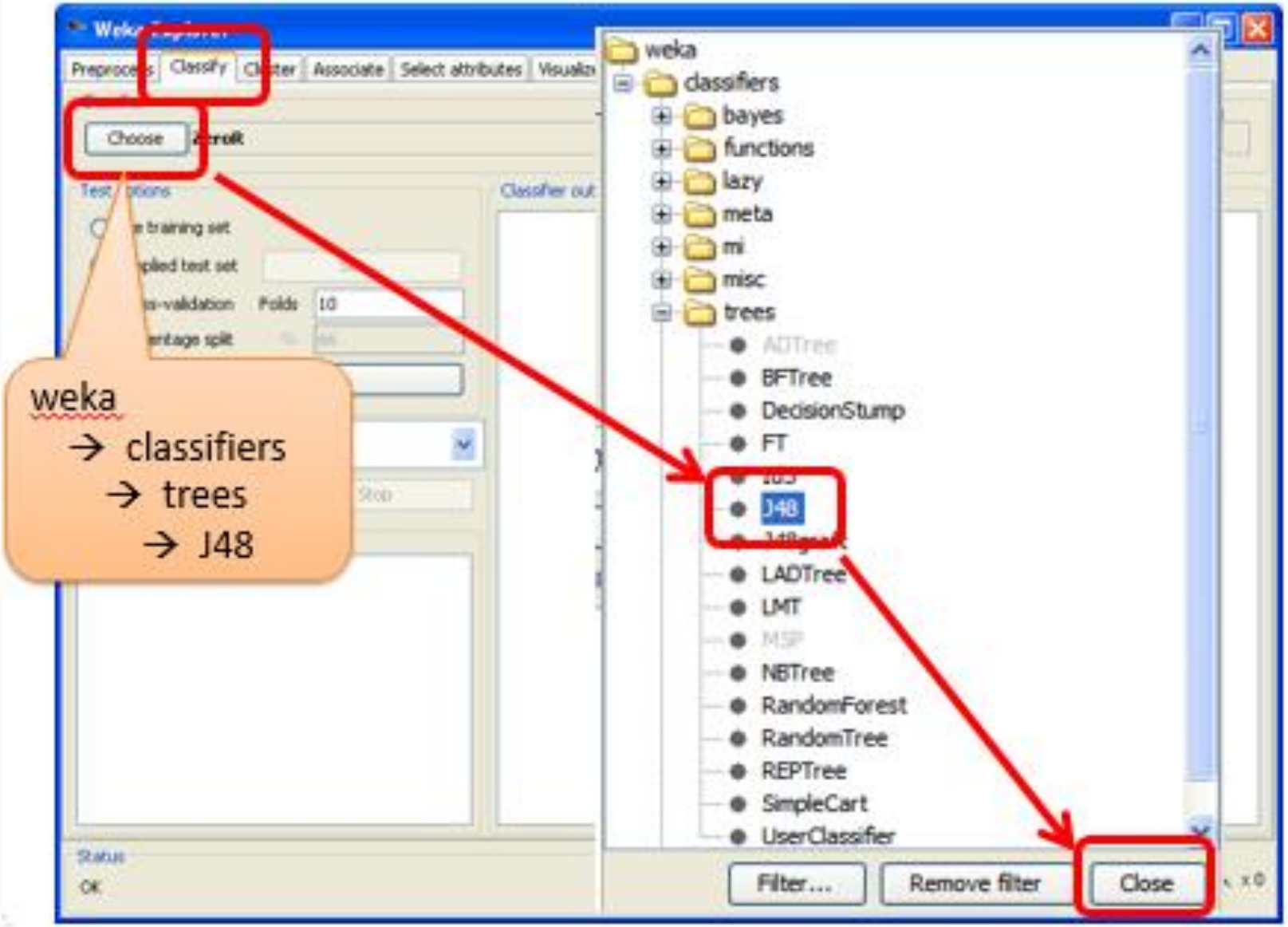
Tools → Package Manager

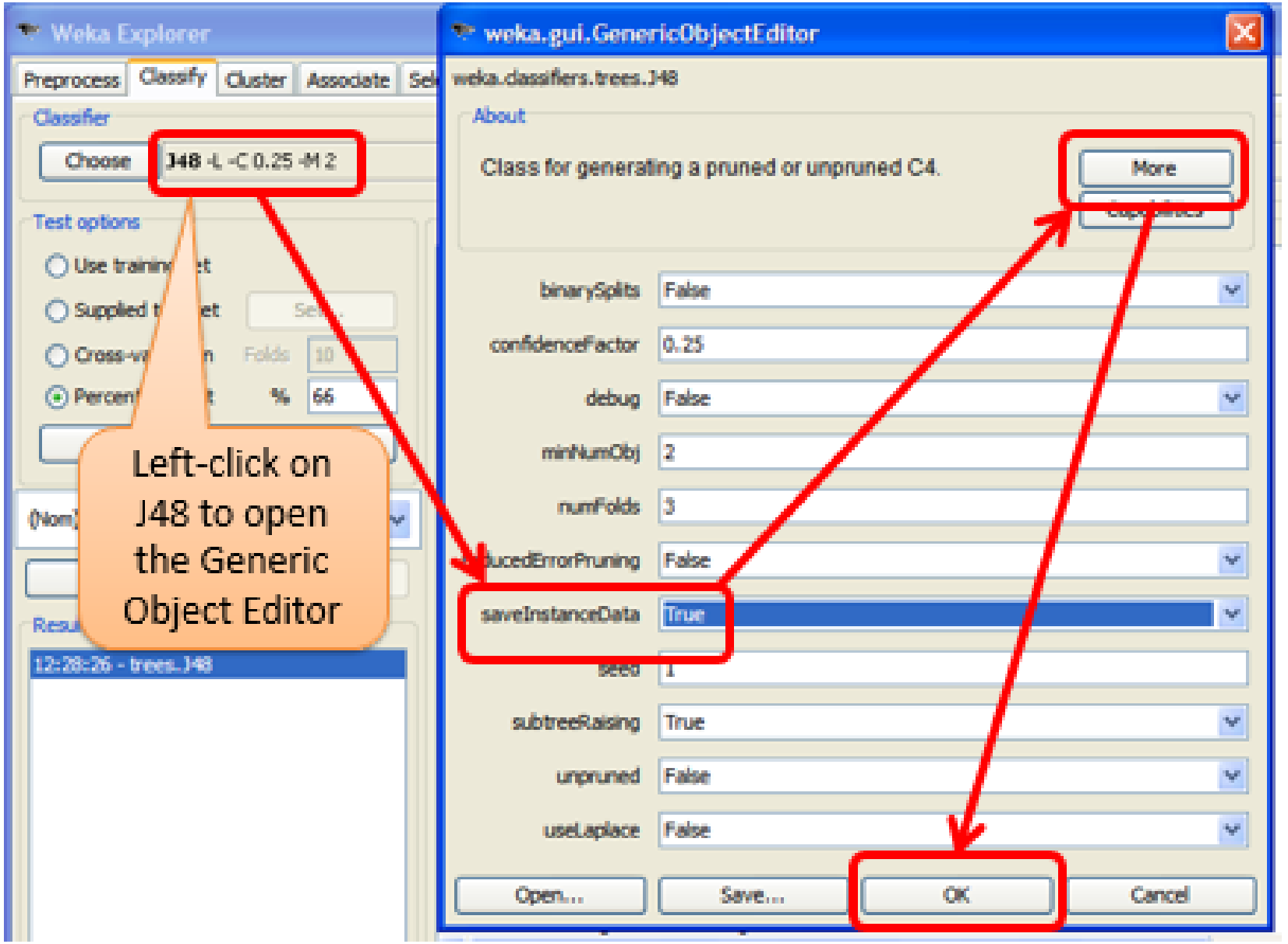


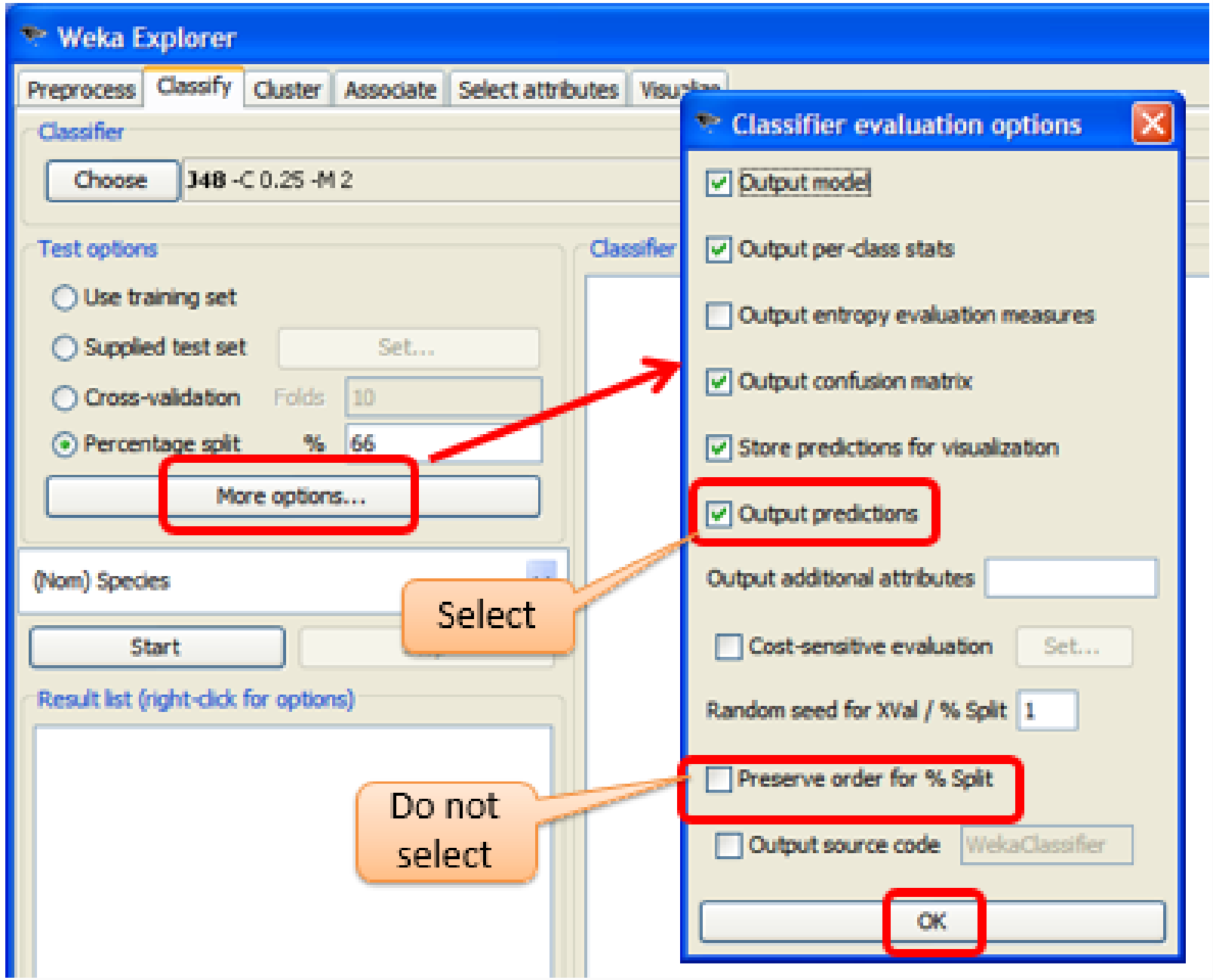
Let's spend a couple minutes looking at different packages…


Open the FishersIrisDataset.arff file
Weka Explorer
Preprocess
Classify
Open file...
Open URL...
Open DB...
Generate...
Filter
Choose
Current relation
Relation: None
Instances: None
Attributes
All
Status
Open
Look in:
WTDS
FishersIrisDataset.arff
FishersIrisDatasetEFD.arff
My Recent Documents
Desktop
My Documents
My Computer
My Network Places
File name:
FishersIrisDataset.arff
Files of type:
Arff data files (*.arff)

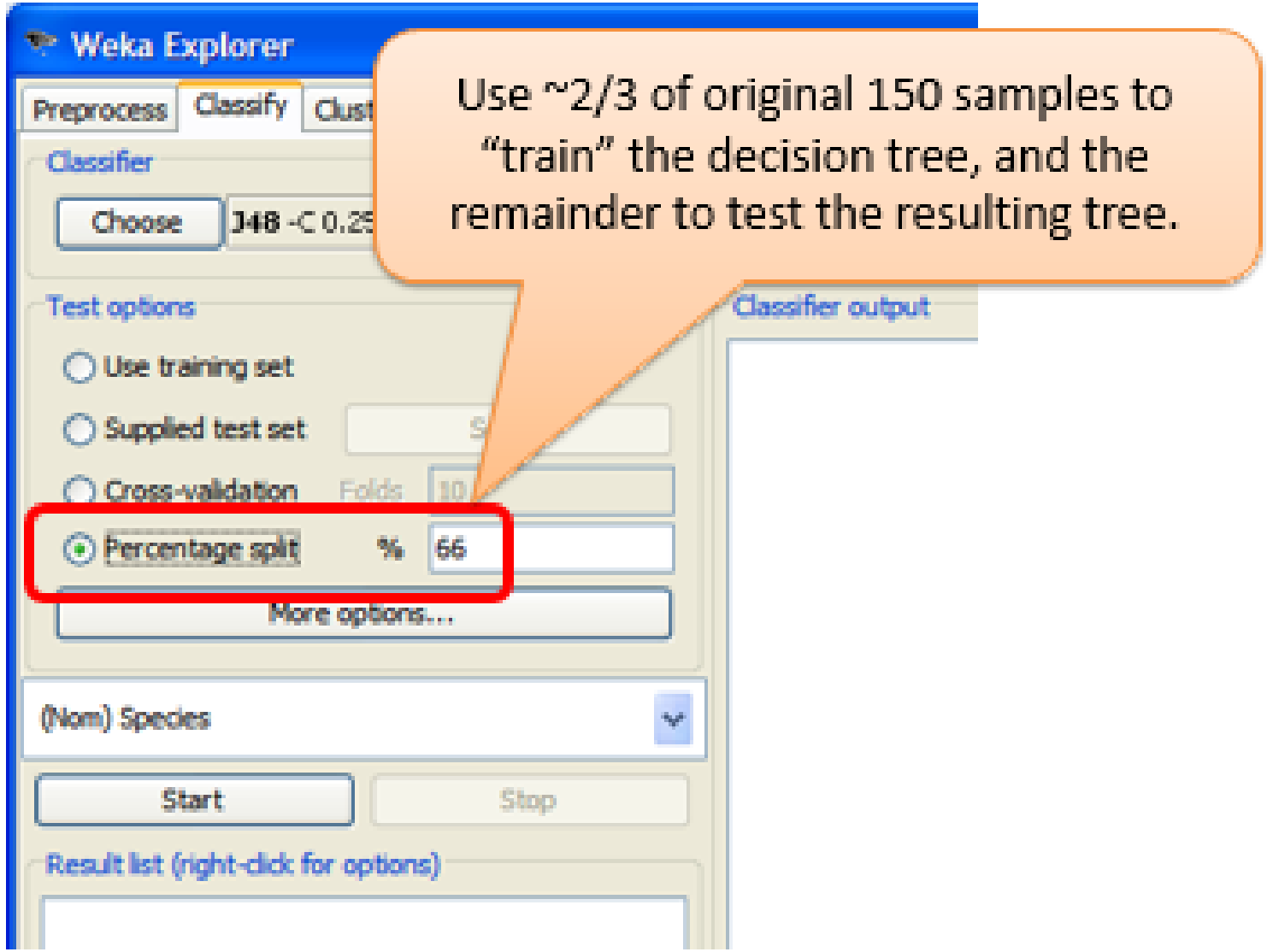
Choose the Decision Tree Algorithm

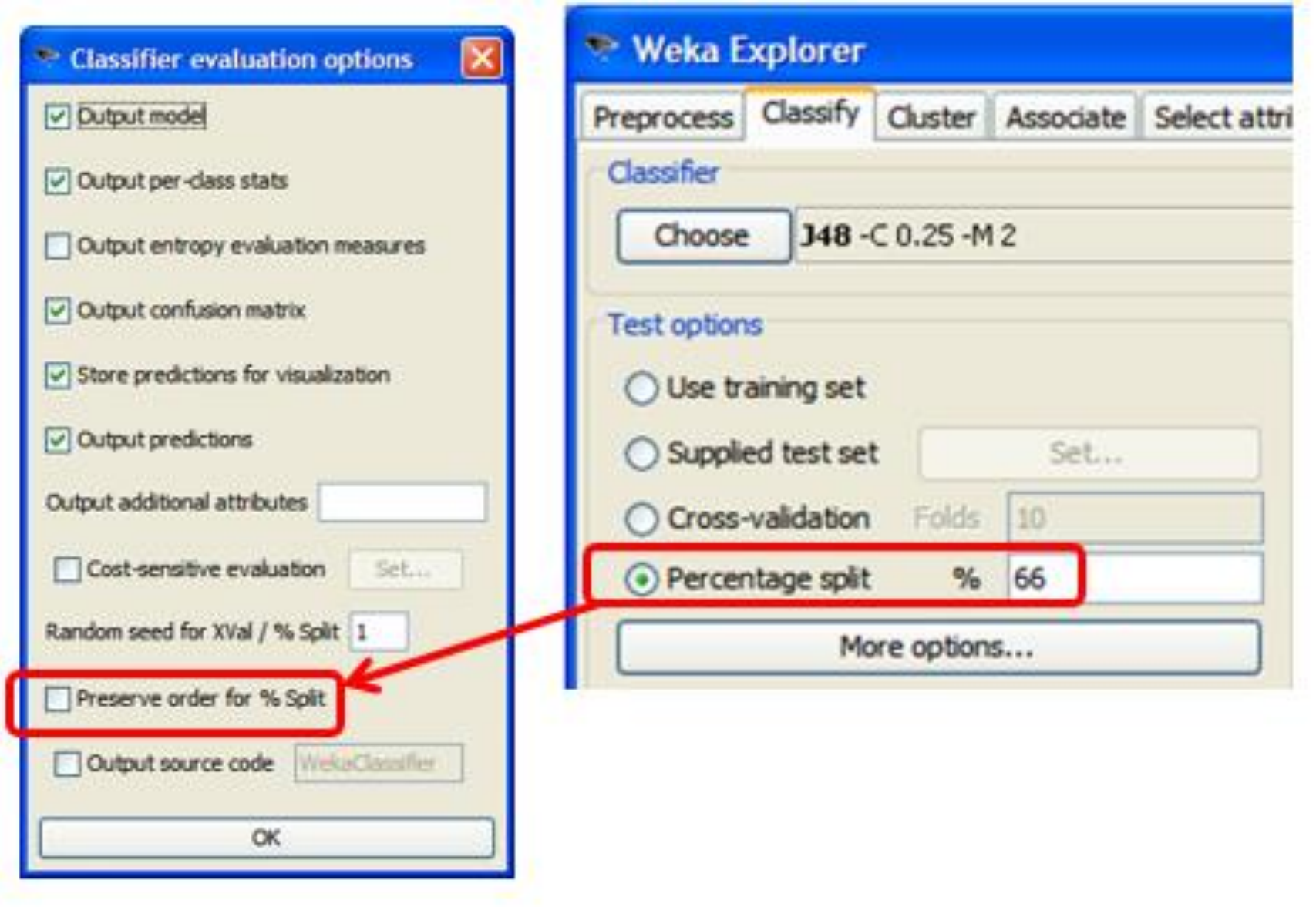


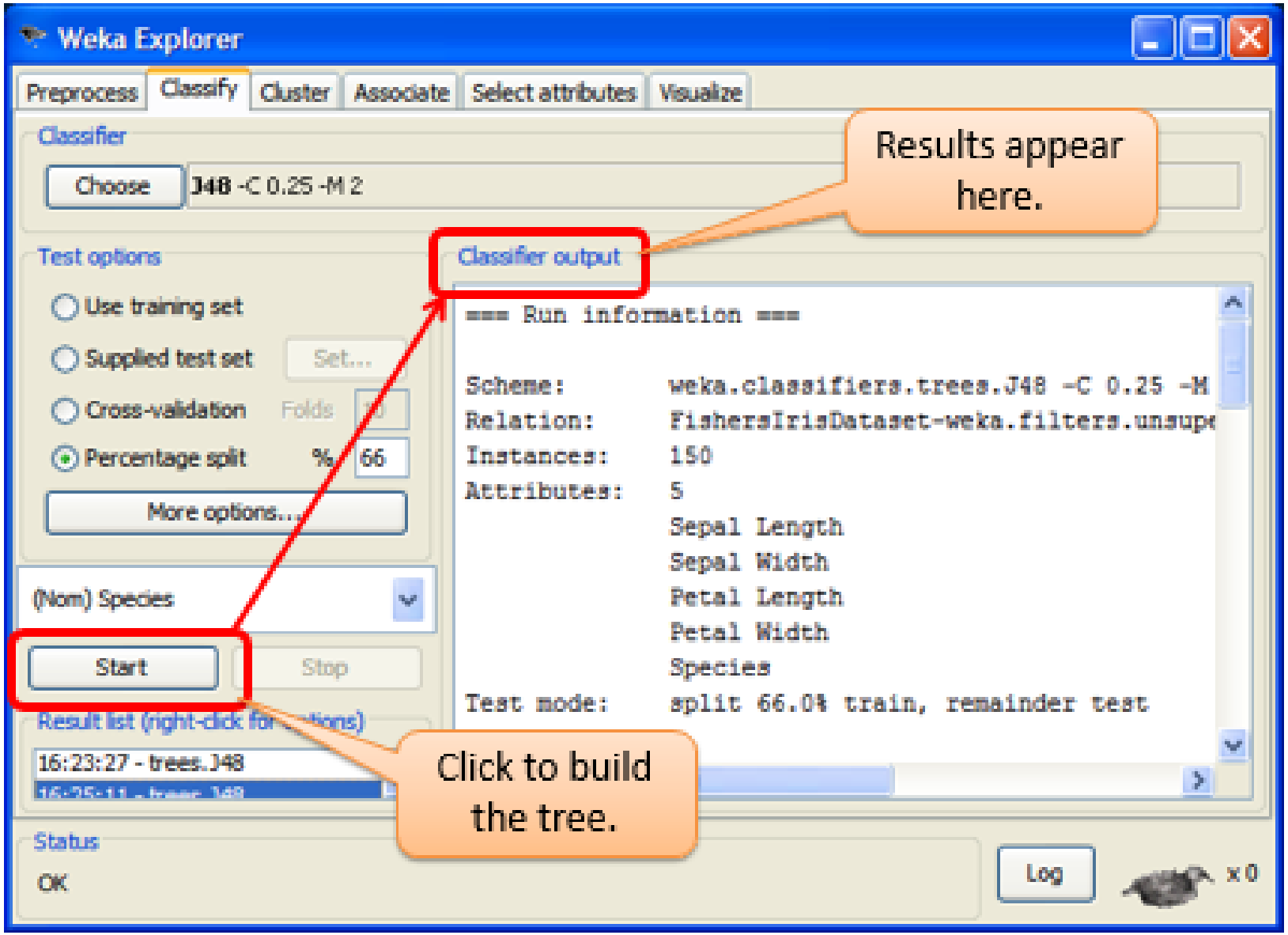

Weka Explorer
Preprocess
Classify
Cluster
Associate
Classifier
Choose
J48 -L -C 0.25 -M 2
Test options
Left-click on J48 to open the Generic Object Editor
12:28:26 - trees.J48
weka.gui.GenericObjectEditor
weka.classifiers.trees.J48
About
Class for generating a pruned or unpruned C4.
More
binarySplits
False
confidenceFactor
0.25
debug
False
minNumObj
2
numFolds
3
saveInstanceData
True
subtreeRaising
True
unpruned
False
useLaplace
False
Open...
Save...
OK
Cancel

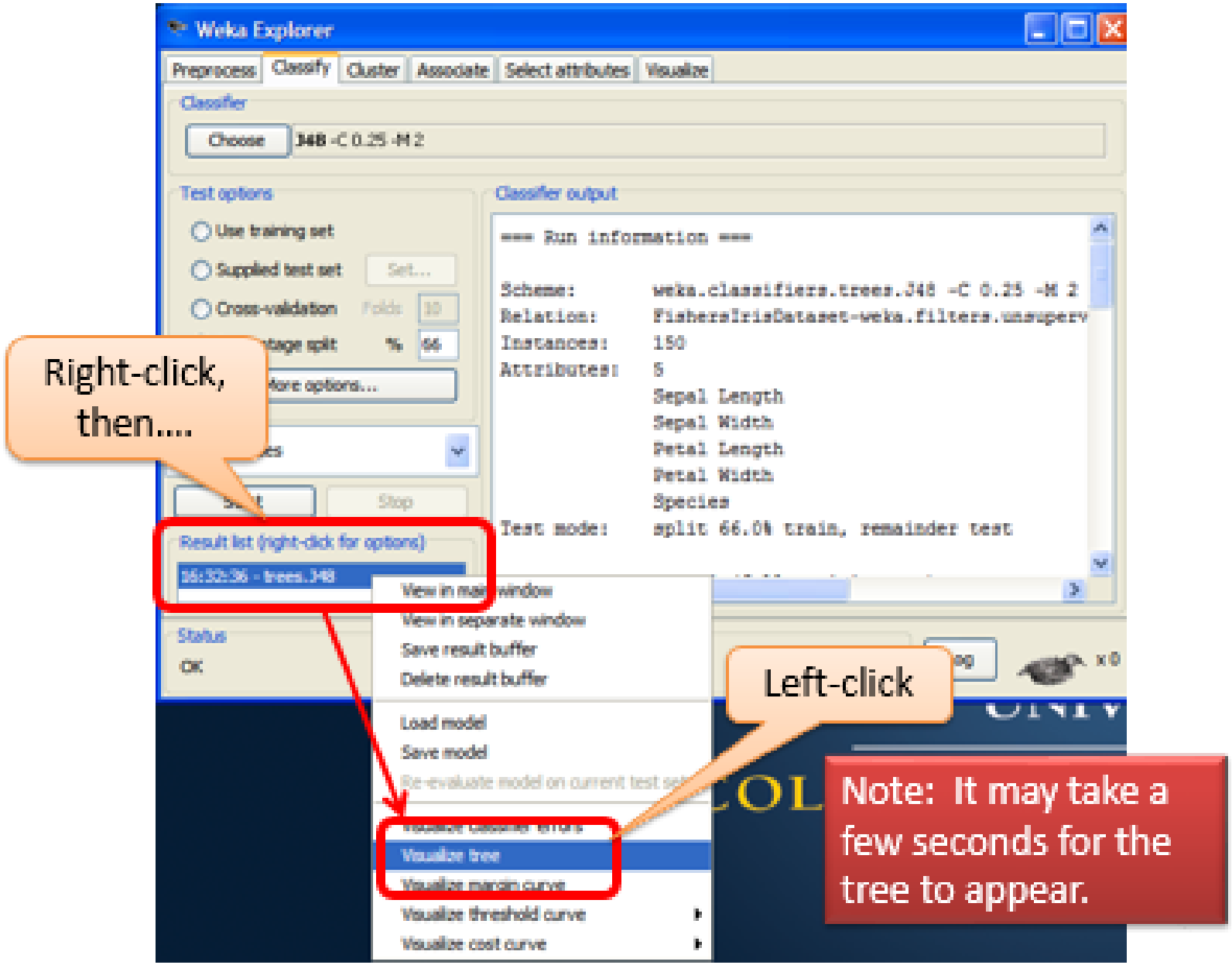

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Classifier
Choose
J48 -C 0.25 -M 2
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) Species
Start
Result list (right-click for options)
Classifier evaluation options
Output model
Output per-class stats
Output entropy evaluation measures
Output confusion matrix
Store predictions for visualization
Output predictions
Output additional attributes
Cost-sensitive evaluation
Set...
Random seed for XVal / % Split
1
Preserve order for % Split
Output source code
WekaClassifier
OK
Select
Do not select

Percentage Split Option

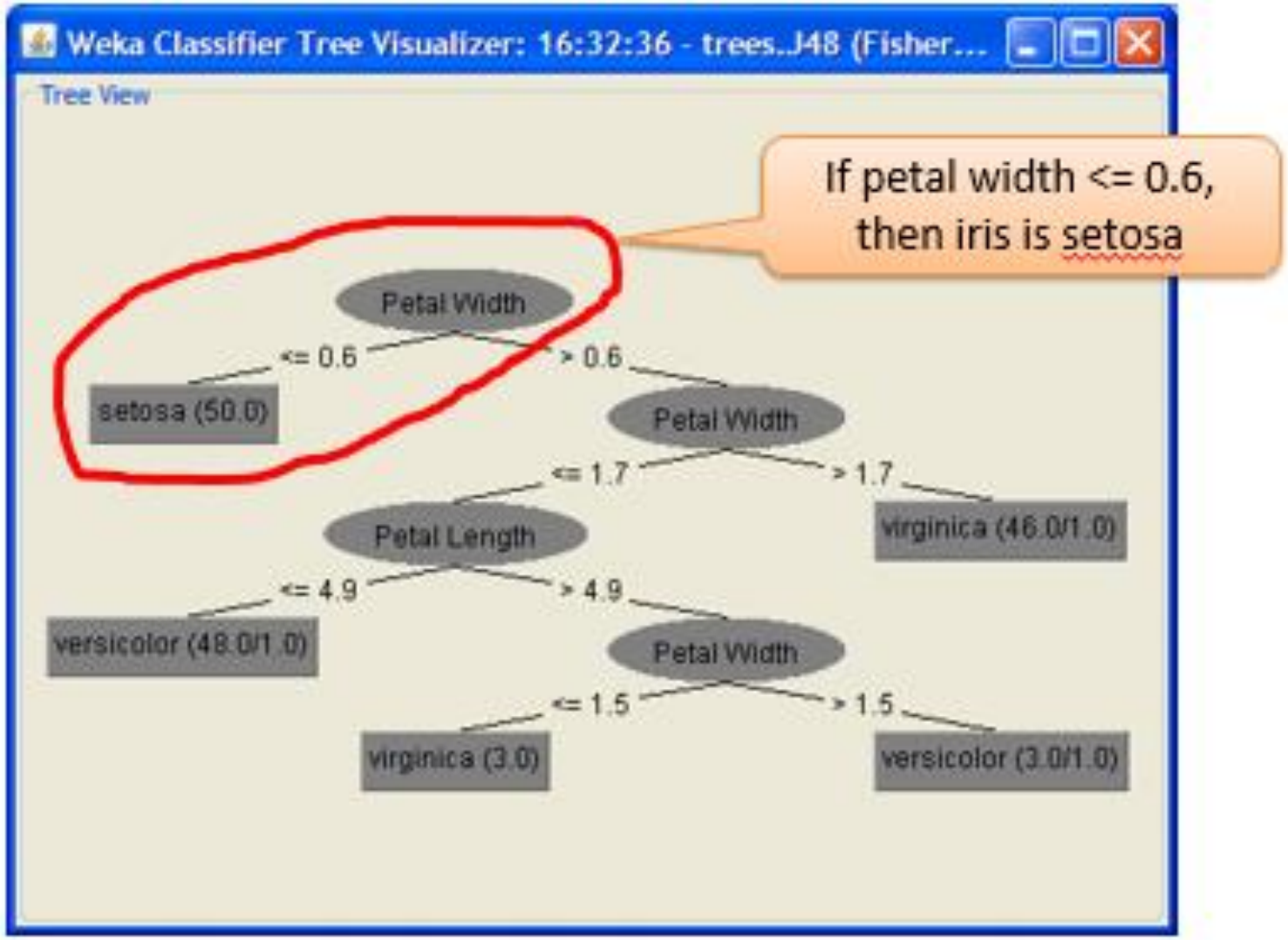


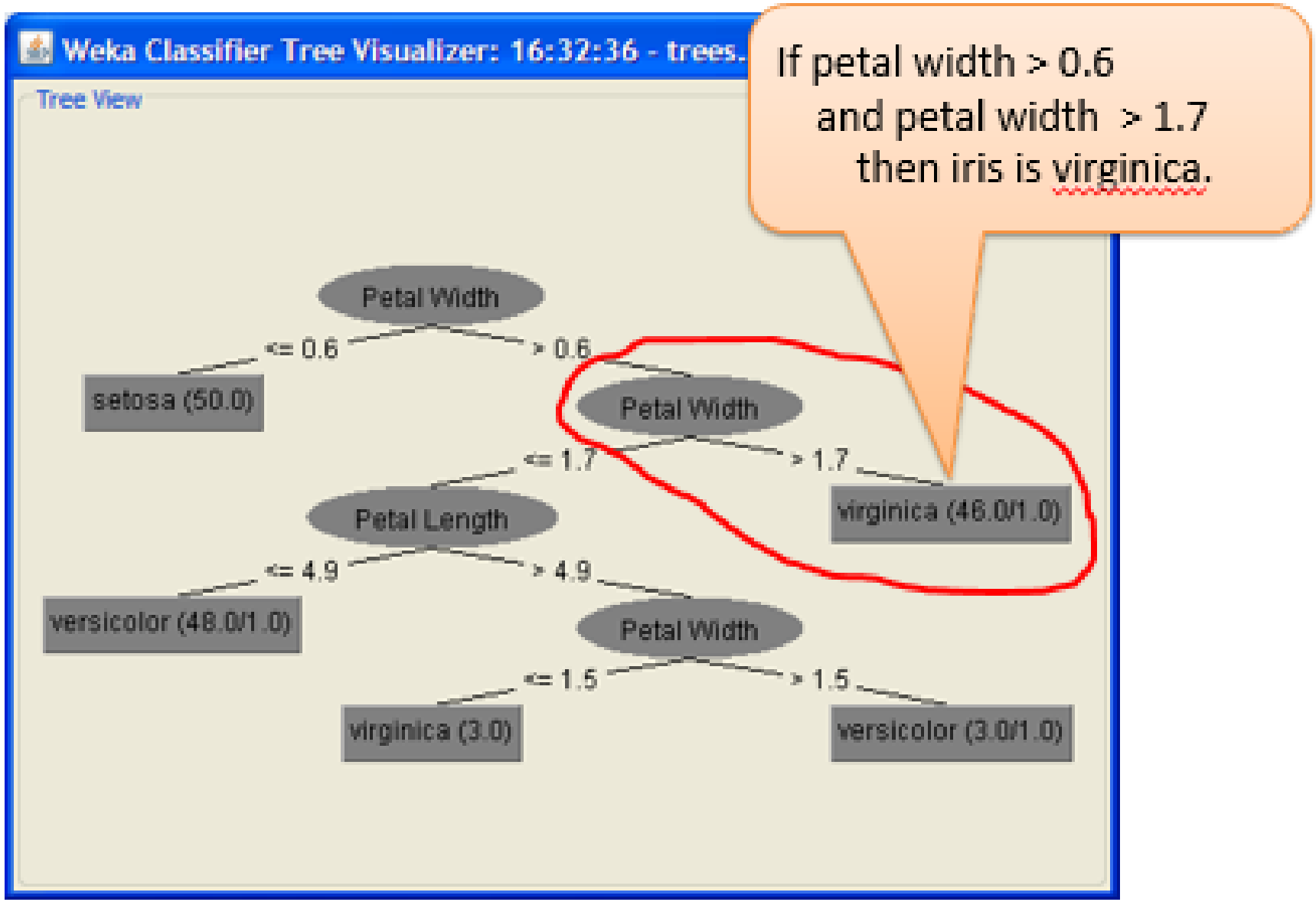

Classifier evaluation options
Output model
Output per-class stats
Output entropy evaluation measures
Output confusion matrix
Store predictions for visualization
Output predictions
Output additional attributes
Cost-sensitive evaluation
Set...
Random seed for XVal / % Split
1
Preserve order for % Split
Output source code
OK
Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attri
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...


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
Percentage split
%
66
More options...
(Nom) Species
Start
Stop
Result list (right-click for options)
16:23:27 - trees.J48
Classifier output
=== Run information ===
Scheme: weka.classifiers.trees.J48 -C 0.25 -M
Relation: FishersIrisDataset-weka.filters.unsupe
Instances: 150
Attributes: 5
Sepal Length
Sepal Width
Petal Length
Petal Width
Species
Test mode: split 66.0% train, remainder test
Results appear here.
Click to build the tree.
Status
OK
Log
x 0

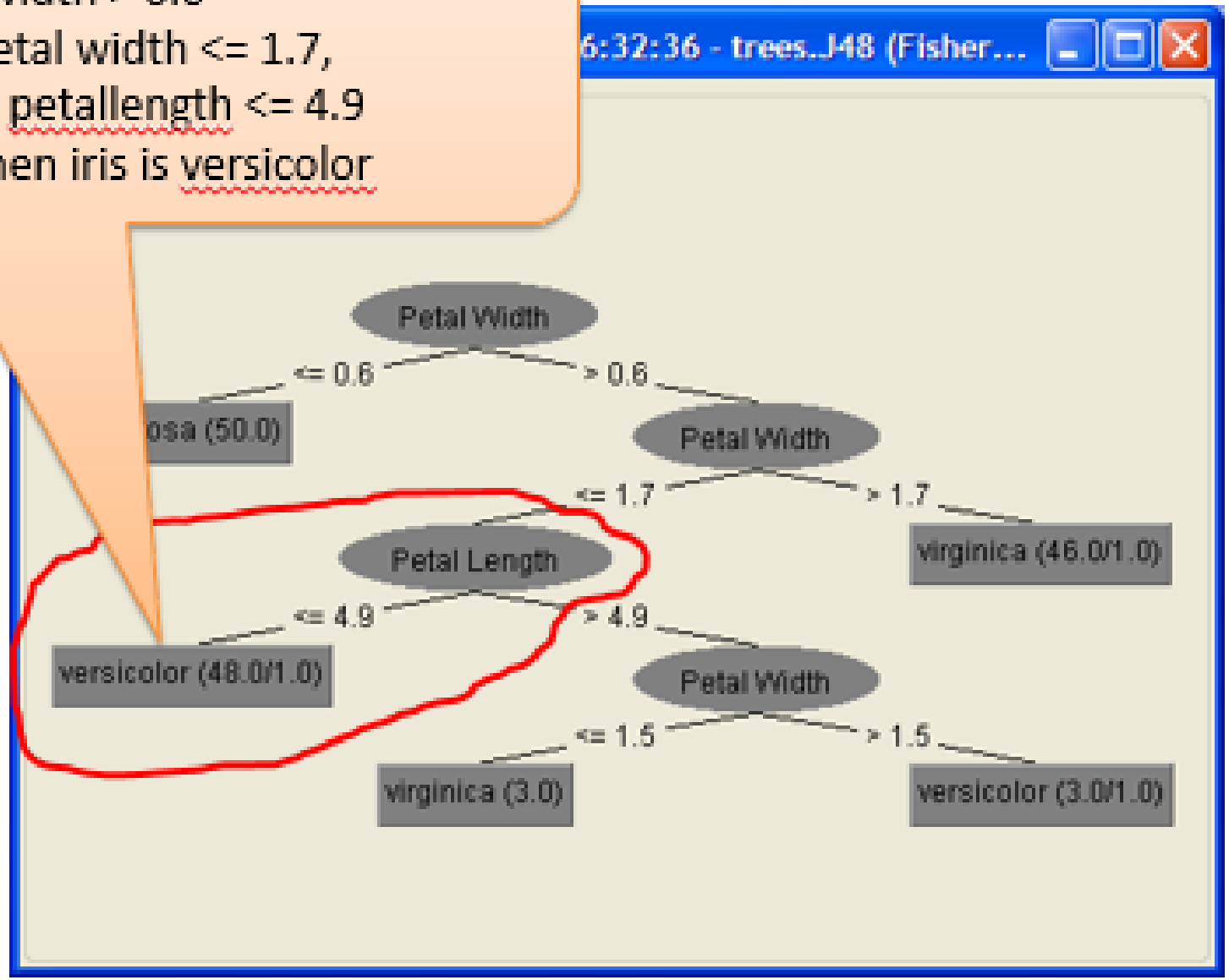

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
J48 -C 0.25 -M 2
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
% split
%
66
More options...
Classifier output
=== Run information ===
Scheme: weka.classifiers.trees.J48 -C 0.25 -M 2
Relation: FisherIrisDataset-weka.filters.unsuperv
Instances: 150
Attributes: 5
Sepal Length
Sepal Width
Petal Length
Petal Width
Species
Test mode: split 66.0% train, remainder test
Stop
Result list (right-click for options)
View in main window
View in separate window
Save result buffer
Delete result buffer
Load model
Save model
Visualize tree
Visualize threshold curve
Visualize cost curve
Status
OK
Right-click, then....
Left-click
Note: It may take a few seconds for the tree to appear.

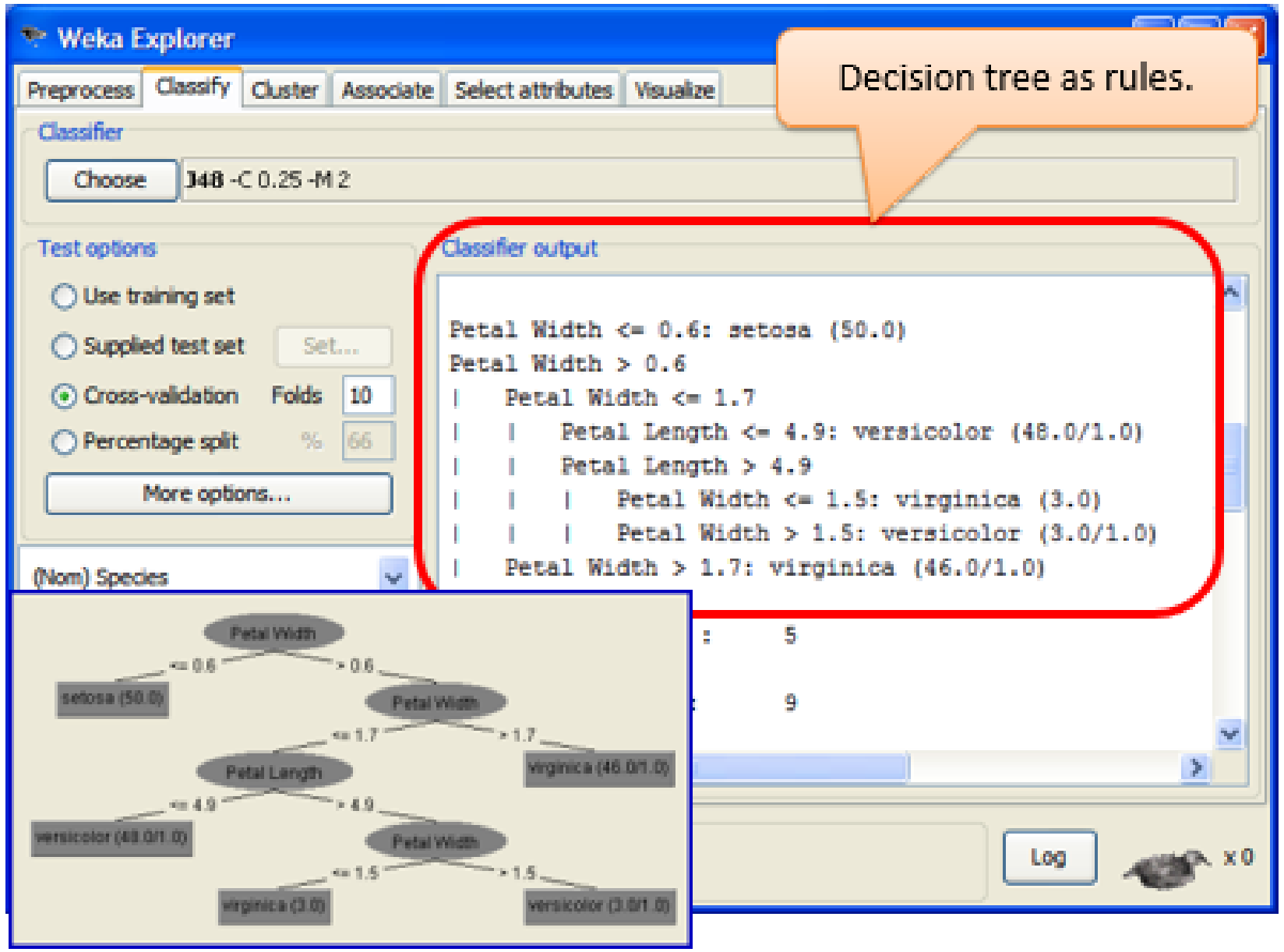
Reading the Decision Tree

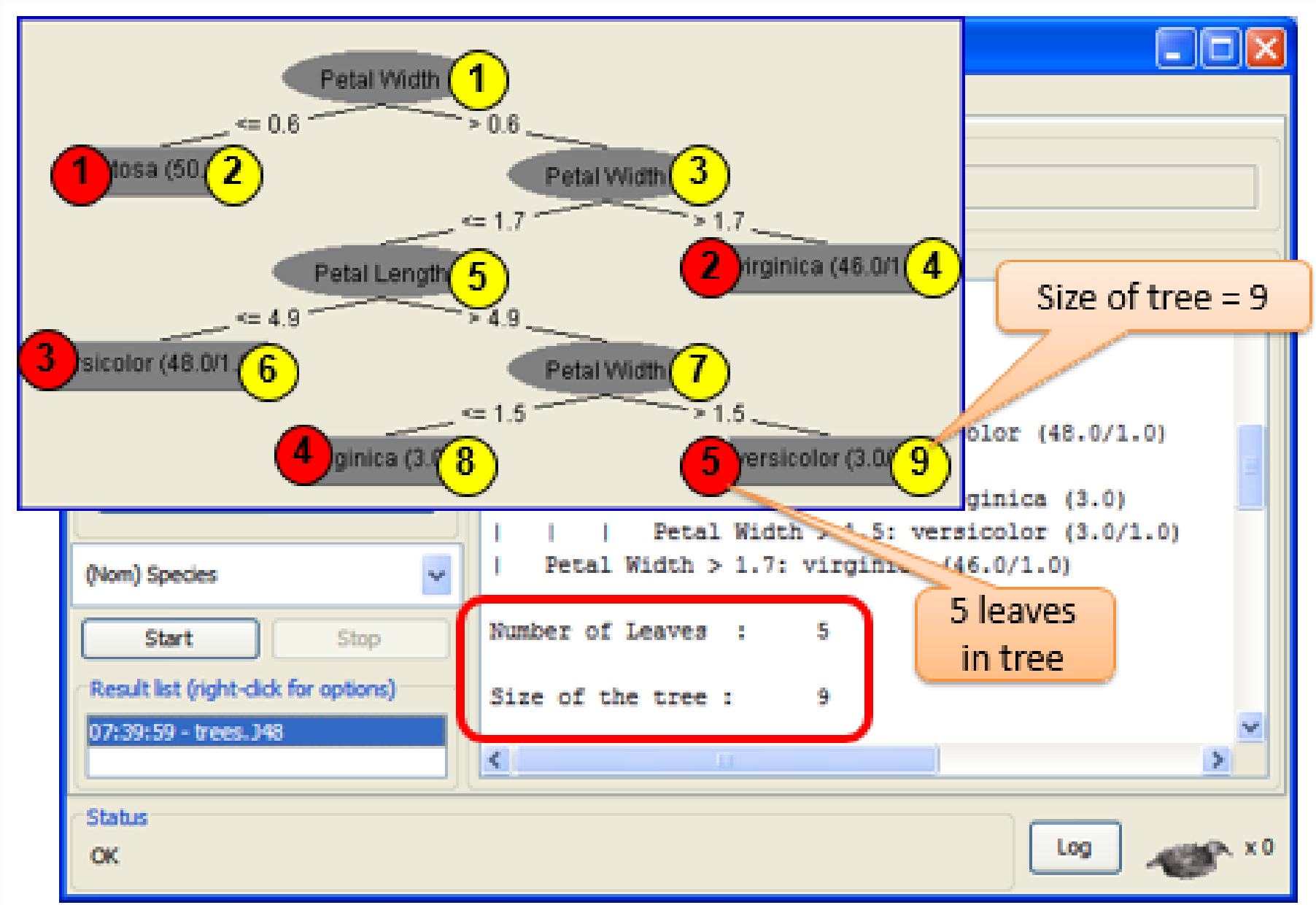



Weka Classifier Tree Visualizer: 16:32:36 - trees.
Tree View
Petal Width
<= 0.6
> 0.6
setosa (50.0)
Petal Width
<= 1.7
> 1.7
Petal Length
virginica (46.0/1.0)
<= 4.9
> 4.9
versicolor (48.0/1.0)
Petal Width
<= 1.5
> 1.5
virginica (3.0)
versicolor (3.0/1.0)
If petal width > 0.6
and petal width > 1.7
then iris is virginica.

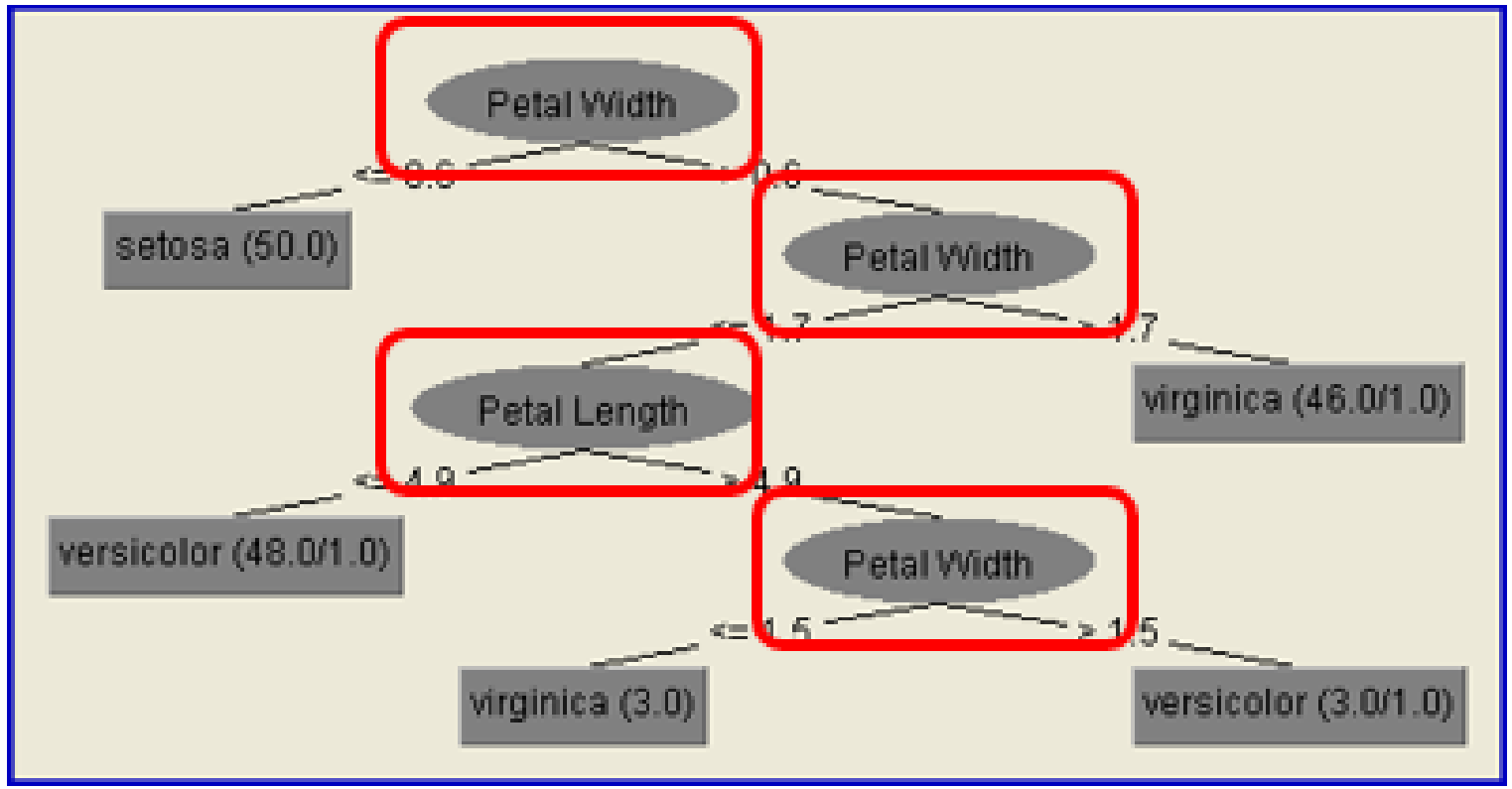
If petal width > 0.6
and petal width <= 1.7,
and petallength <= 4.9
then iris is versicolor




Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Decision tree as rules.
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...
Classifier output
Petal Width <= 0.6: setosa (50.0)
Petal Width > 0.6
| Petal Width <= 1.7
| | Petal Length <= 4.9: versicolor (48.0/1.0)
| | Petal Length > 4.9
| | | Petal Width <= 1.5: virginica (3.0)
| | | Petal Width > 1.5: versicolor (3.0/1.0)
| Petal Width > 1.7: virginica (46.0/1.0)
(Nom) Species
Petal Width
<= 0.6
> 0.6
setosa (50.0)
Petal Width
<= 1.7
> 1.7
Petal Length
virginica (46.0/1.0)
<= 4.9
> 4.9
versicolor (48.0/1.0)
Petal Width
<= 1.5
> 1.5
virginica (3.0)
versicolor (3.0/1.0)
Log
x 0


Petal Width
1
<= 0.6
> 0.6
1
tosa (50.
2
Petal Width
3
<= 1.7
> 1.7
Petal Length
5
2
irginica (46.0/1
4
<= 4.9
> 4.9
3
sicolor (48.0/1
6
Petal Width
7
<= 1.5
> 1.5
4
ginica (3.
8
5
ersicolor (3.0/
9
Size of tree = 9
olor (48.0/1.0)
ginica (3.0)
| | | Petal Width > 1.5: versicolor (3.0/1.0)
| Petal Width > 1.7: virgini (46.0/1.0)
(Nom) Species
Start
Stop
Result list (right-click for options)
07:39:59 - trees.J48
Number of Leaves : 5
Size of the tree : 9
5 leaves in tree
Status
OK
Log
x 0

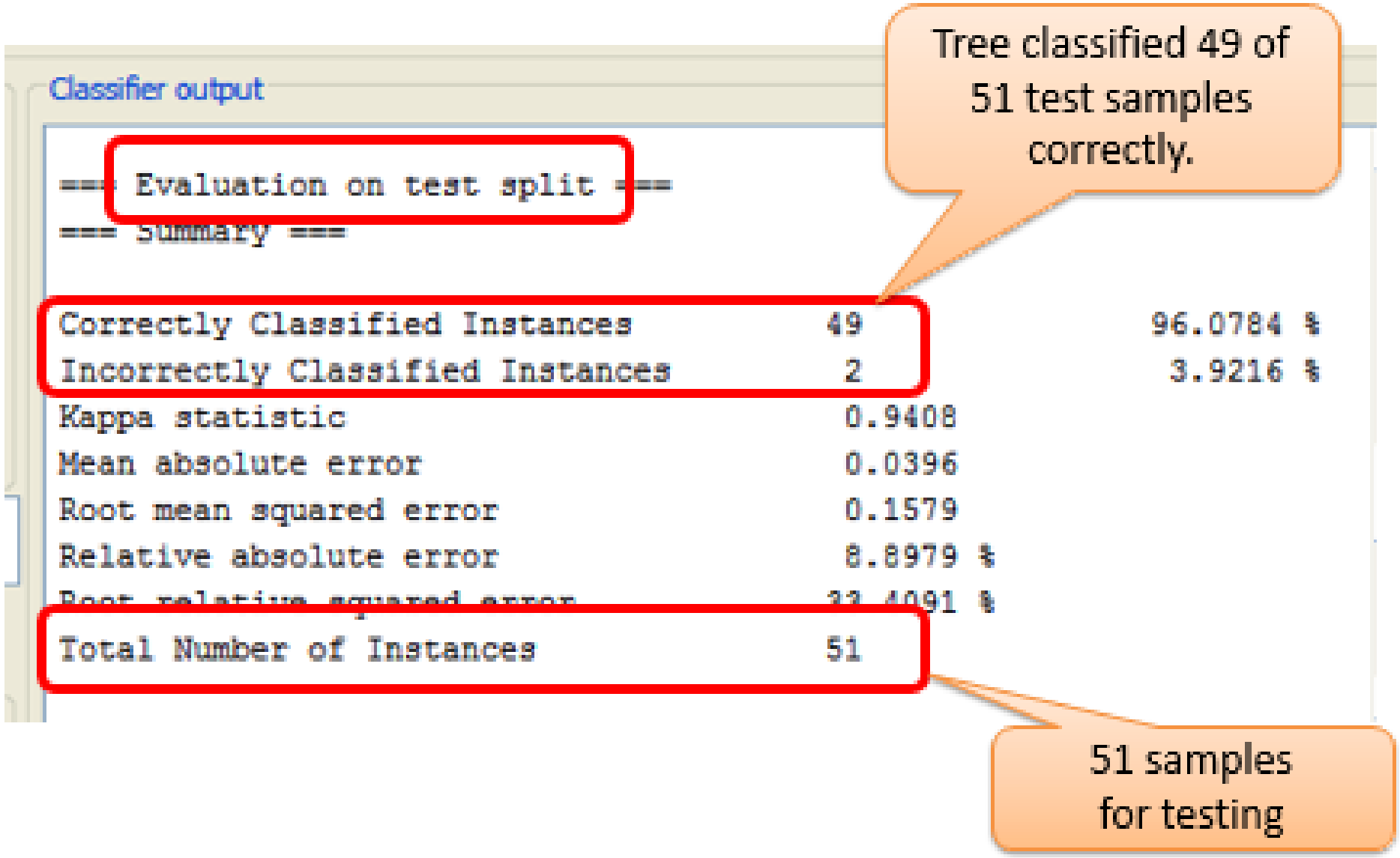
Only need to measure petal length and petal width!

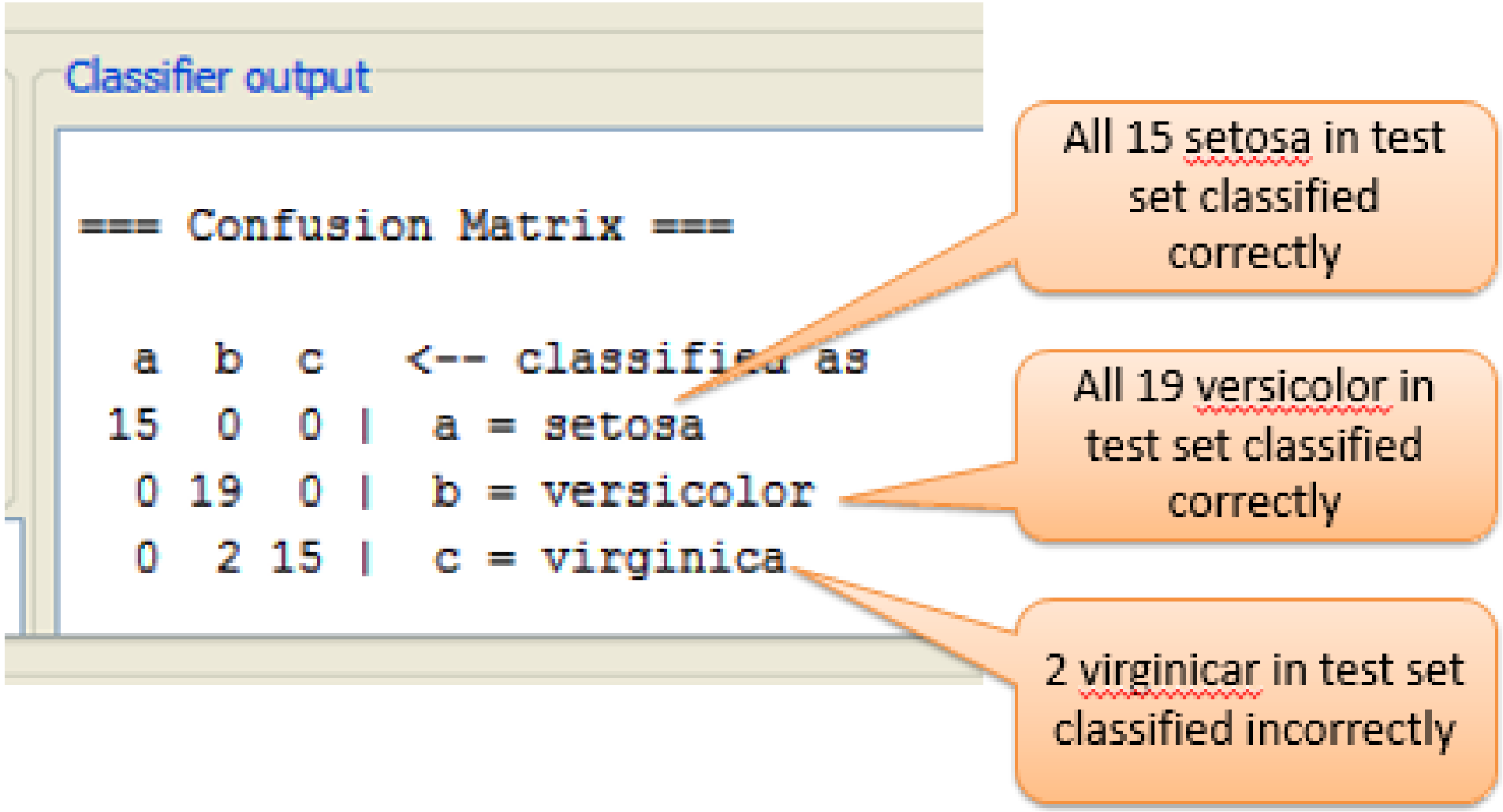


Goal is to:

Classify the most test samples correctly,

While minimizing tree size and number of leaves.


Classifier output
=== Evaluation on test split ===
=== Summary ===
Correctly Classified Instances 49 96.0784 %
Incorrectly Classified Instances 2 3.9216 %
Kappa statistic 0.9408
Mean absolute error 0.0396
Root mean squared error 0.1579
Relative absolute error 8.8979 %
Total Number of Instances 51
Tree classified 49 of 51 test samples correctly.
51 samples for testing


Classifier output
=== Confusion Matrix ===
a b c <-- classified as
15 0 0 | a = setosa
0 19 0 | b = versicolor
0 2 15 | c = virginica
All 15 setosa in test set classified correctly
All 19 versicolor in test set classified correctly
2 virginicar in test set classified incorrectly

Lecture 3 Homework 1

When completed, submit this assignment to Moodle for homework Lecture 4

Lecture 3 Homework 1

Your task for this assignment: Design a simple, low-cost sensor that can distinguish between red wine and white wine.

Your sensor must correctly distinguish between red and white wine for at least 95% of the samples in a set of 6497 test samples of red and white wine.

Your technology is capable of sensing the following wine attributes:

- Fixed acidity
- Volatile acidity
- Citric acid
- Residual sugar
- Chlorides
- Density
- Free sulphur dioxide
- Total sulphur dioxide
- Sulphates
- pH
- Alcohol

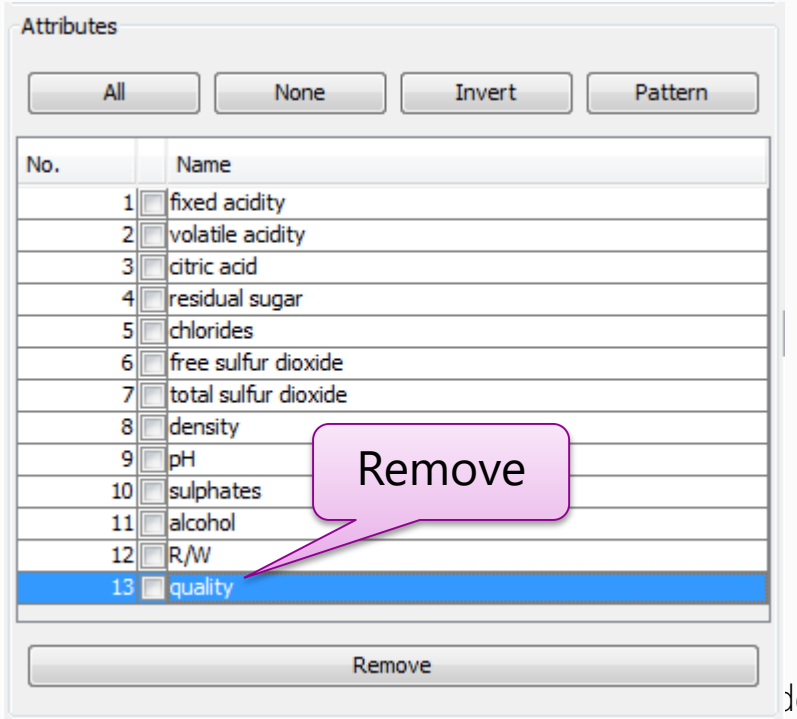
To keep your sensor cheap and simple, **you need to sense as few of these attributes as possible to meet the 95% requirement**.

Question: Which attributes should your sensor be capable of measuring?

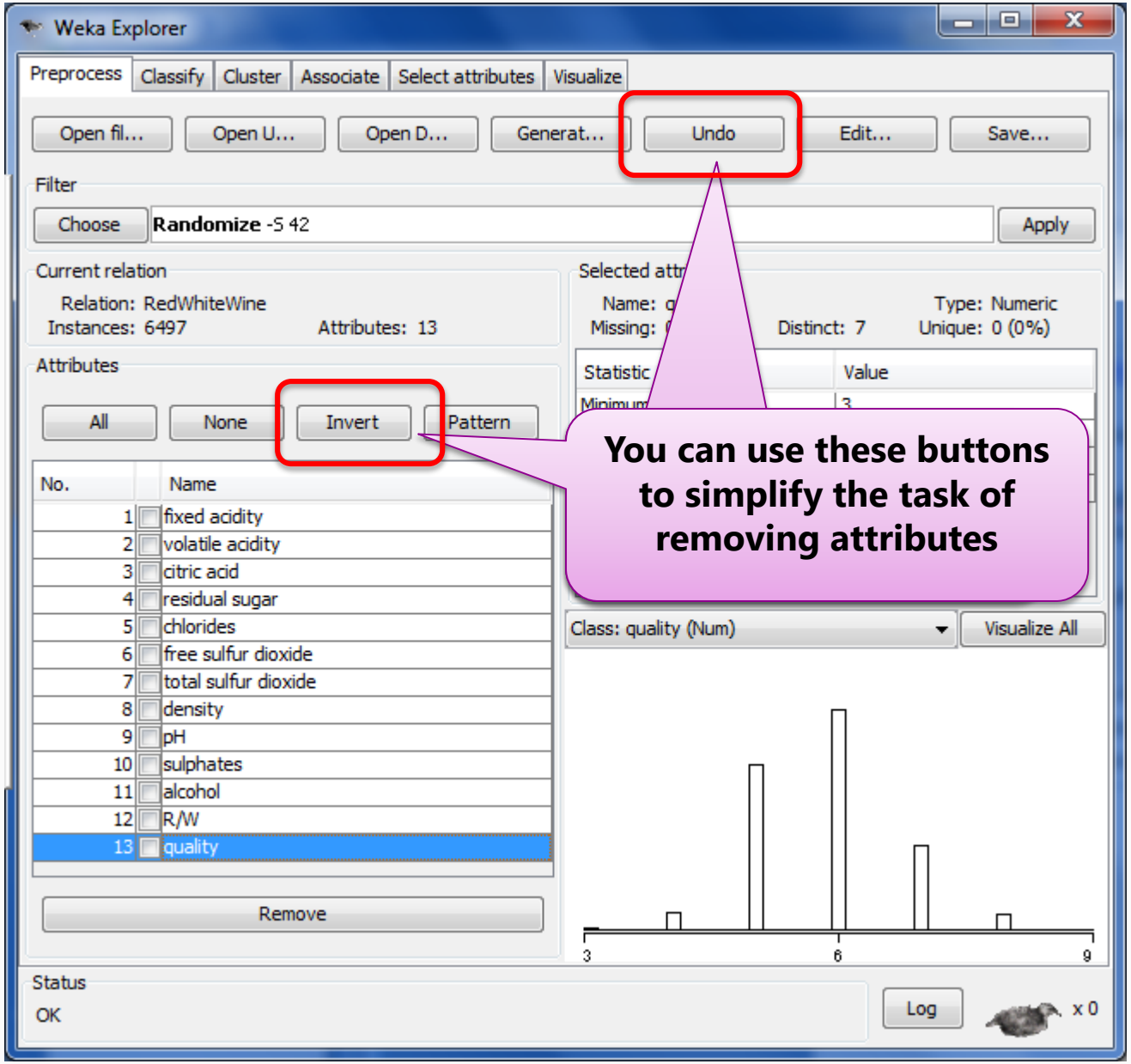
Lecture 3 Homework 1

1. Go to our class website, Lecture 3, and download the associated homework files
2. Read WineQuality.pdf.

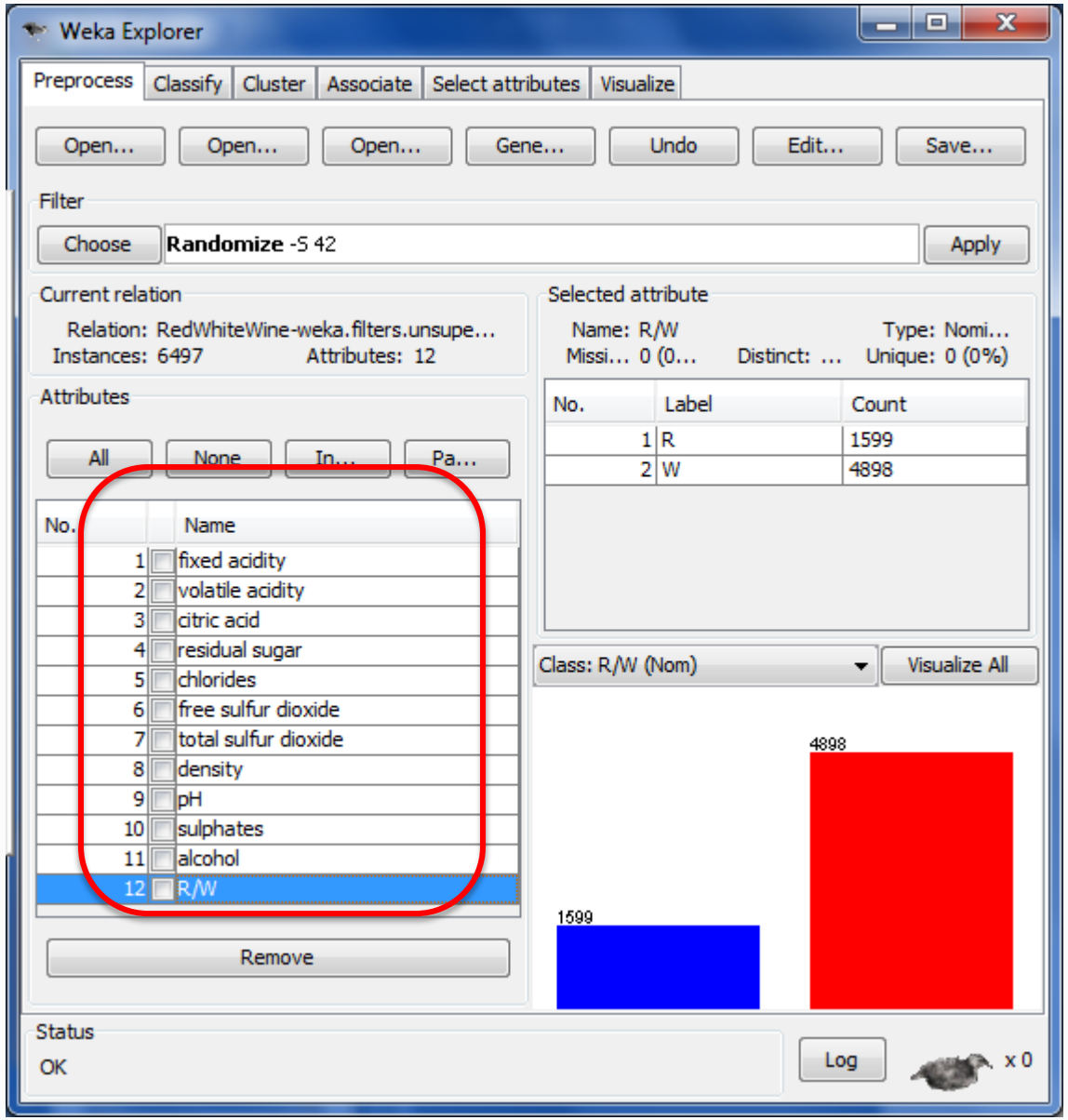
3. Open the RedWhiteWine.arff file in Weka, and remove the *quality* attribute, which you will not need for this assignment.
4. Run J48 with default setting to see what kind of percent correct classification results you get using all attributes.
5. Remove attributes to find the minimum number of attributes needed to meet the 95% correct classification requirement.



Lecture 3 Homework 1

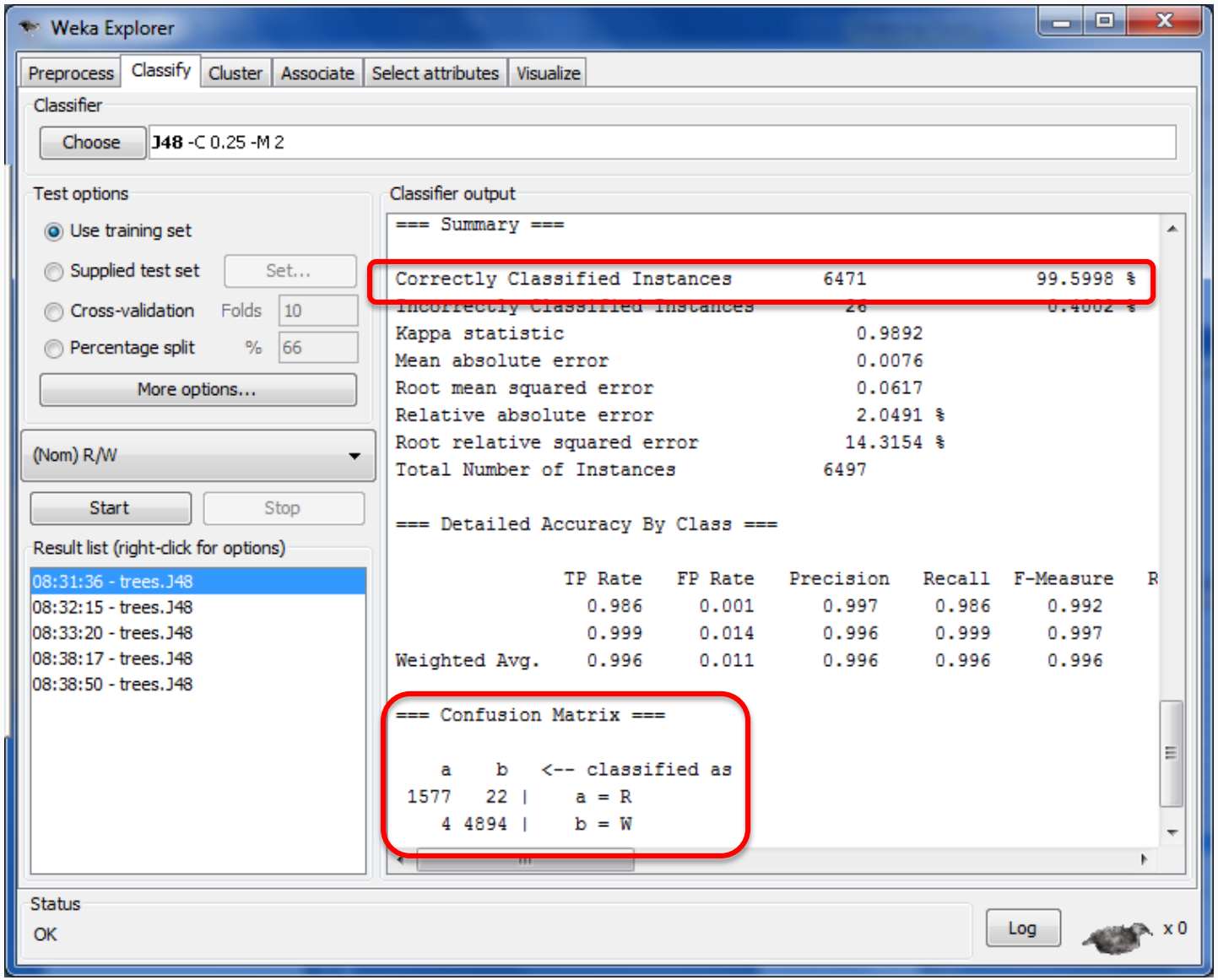


Lecture 3 Homework 1



(Paste a screenshot showing your minimum attribute set here)

Lecture 3 Homework 1



(Paste a screenshot showing your results for your minimum attribute set here)

Data Science

Deriving Knowledge from Data at Scale

That's all for tonight….