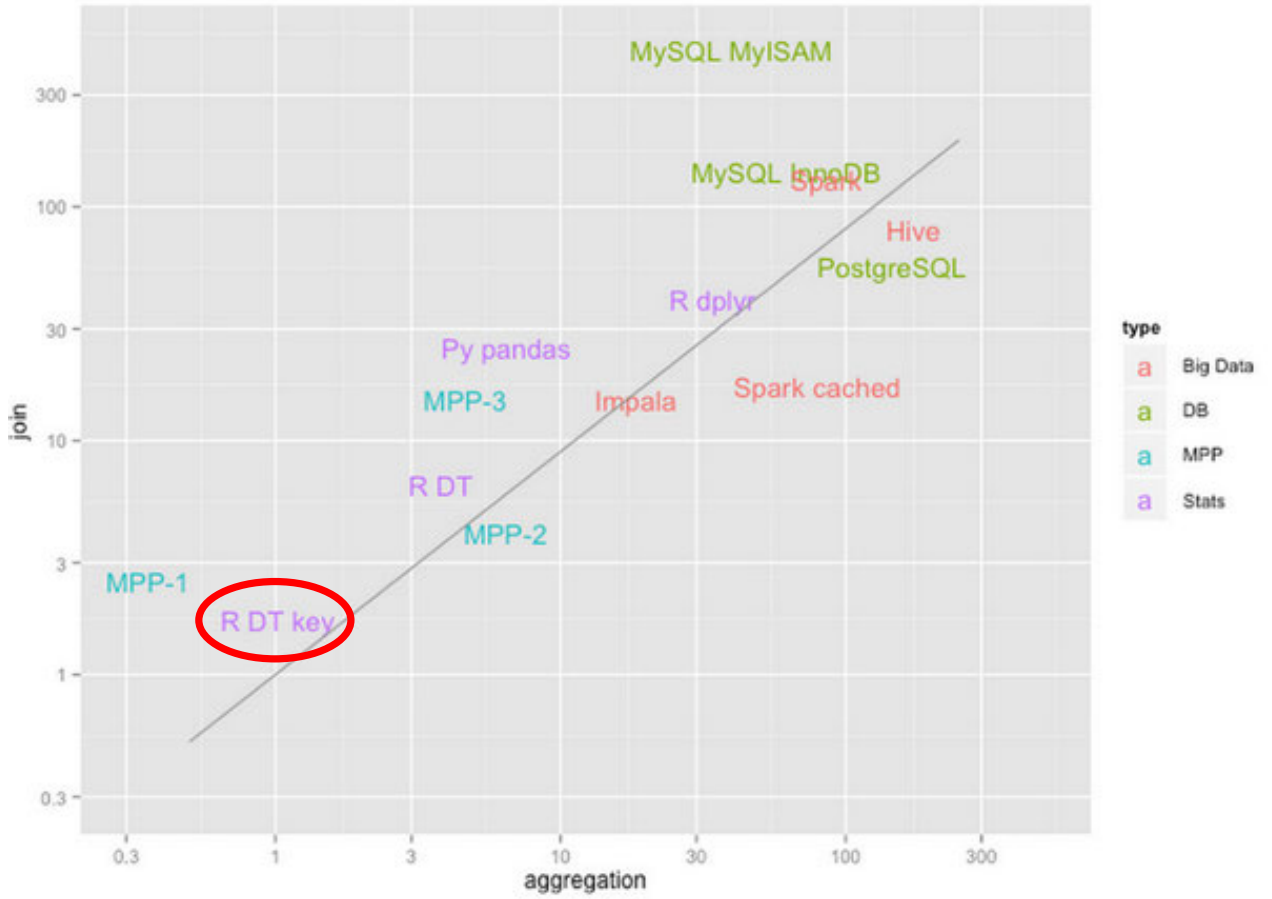
R Workshop

A hands-on introduction

University of Zurich[UZH]

Dr. Markus Meierer | markus.meierer@business.uzh.ch --- Patrick Bachmann | patrick.bachmann@business.uzh.ch

Comparing AGGREGATION of 100M to 1M rows and JOIN of 100 and 1M row table (1M distinct IDs)



https://twitter.com/MattDowle/status/639890171685662720/photo/1

https://github.com/szilard/benchm-databases

Sidenote: Installing multiple packages at once (incl. check if package is already installed)

R Code

```
#List all packages
list.of.packages <- c("ggplot2", "Rcpp")

#Check if packages are already installed
new.packages <-
list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]

#Install all packages which are not yet installed
if(length(new.packages) >0) {install.packages(new.packages)}

#Load all packages
sapply(list.of.packages, library, character.only = TRUE)
```

http://stackoverflow.com/questions/4090169/elegant-way-to-check-for-missing-packages-and-install-them

https://gist.github.com/stevenworthington/3178163

Sidenote: CRAN Task Views – The best way to get into a new field in R

Key task views for marketing analytics are

- For knowledge discovery: Machine Learning
- Because the data sets are large: High Performance Computing
- For data exploration and visualization: Graphics
- For churn modelling: Survival

CRAN Task Views

Bayesian	Bayesian Inference
ChemPhys	Chemometrics and Computational Physics
ClinicalTrials	Clinical Trial Design, Monitoring, and Analysis
Cluster	Cluster Analysis & Finite Mixture Models
DifferentialEquations	Differential Equations
Distributions	Probability Distributions
Econometrics	Econometrics
Environmetrics	Analysis of Ecological and Environmental Data
ExperimentalDesign	Design of Experiments (DoE) & Analysis of Experimental Data
Finance	Empirical Finance
Genetics	Statistical Genetics
Graphics	Graphic Displays & Dynamic Graphics & Graphic Devices & Visualization
HighPerformanceComputing	High-Performance and Parallel Computing with R
MachineLearning	Machine Learning & Statistical Learning
MedicalImaging	Medical Image Analysis
MetaAnalysis	Meta-Analysis
Multivariate	Multivariate Statistics
NaturalLanguageProcessing	Natural Language Processing
NumericalMathematics	Numerical Mathematics
OfficialStatistics	Official Statistics & Survey Methodology
Optimization	Optimization and Mathematical Programming
Pharmacokinetics	Analysis of Pharmacokinetic Data
Phylogenetics	Phylogenetics, Especially Comparative Methods
Psychometrics	Psychometric Models and Methods
ReproducibleResearch	Reproducible Research
Robust	Robust Statistical Methods
SocialSciences	Statistics for the Social Sciences
Spatial	Analysis of Spatial Data

https://cran.r-project.org/web/views/

Sidenote: CRAN Task Views – Machine Learning

CRAN Task View: Machine Learning & Statistical Learning

Maintainer: Torsten Hothorn
Contact: Torsten.Hothorn at R-project.org
Version: 2014-12-18

Several add-on packages implement ideas and methods developed at the borderline between computer science and statistics - this field of research is usually referred to as machine learning. The packages can be roughly structured into the following topics:

- *Neural Networks* : Single-hidden-layer neural network are implemented in package nnet (shipped with base R). Package RSNNS offers an interface to the Stuttgart Neural Network Simulator (SNNS). An interface to the FCNN library allows user-extensible artificial neural networks in package FCNN4R.
- *Recursive Partitioning* : Tree-structured models for regression, classification and survival analysis, following the ideas in the CART book, are implemented in rpart (shipped with base R) and tree. Package rpart is recommended for computing CART-like trees. A rich toolbox of partitioning algorithms is available in Weka , package RWeka provides an interface to this implementation, including the J4.8-variant of C4.5 and M5. The Cubist package fits rule-based models (similar to trees) with linear regression models in the terminal leaves, instance-based corrections and boosting. The C50 package can fit C5.0 classification trees, rule-based models, and boosted versions of these.
  Two recursive partitioning algorithms with unbiased variable selection and statistical stopping criterion are implemented in package party. Function `ctree()` is based on non-parametrical conditional inference procedures for testing independence between response and each input variable whereas `mob()` can be used to partition parametric models. Extensible tools for visualizing binary trees and node distributions of the response are available in package party as well.
  Tree-structured varying coefficient models are implemented in package vcrpart.
  For problems with binary input variables the package LogicReg implements logic regression. Graphical tools for the visualization of trees are available in package maptree.
  Trees for modelling longitudinal data by means of random effects is offered by package REEMtree. Partitioning of mixture models is performed by RPMM.
  Computational infrastructure for representing trees and unified methods for predition and visualization is implemented in partykit. This infrastructure is used by package evtree to implement evolutionary learning of globally optimal trees. Oblique trees are available in package oblique.tree.
- *Random Forests* : The reference implementation of the random forest algorithm for regression and classification is available in package randomForest. Package ipred has bagging for regression, classification and survival analysis as well as bundling, a combination of multiple models via ensemble learning. In addition, a random forest variant for response variables measured at arbitrary scales based on conditional inference trees is implemented in package party. randomForestSRC implements a unified treatment of Breiman's random forests for survival, regression and classification problems. Quantile regression forests quantregForest allow to regress quantiles of a numeric response on exploratory variables via a random forest approach. For binary data, LogicForest is a forest of logic regression trees (package LogicReg. The varSelRF and Boruta packages focus on variable selection by means for random forest algorithms. For large data sets, package bigrf computes random forests in parallel and uses large memory objects to store the data.
- *Regularized and Shrinkage Methods* : Regression models with some constraint on the parameter estimates can be fitted with the lasso2 and lars packages. Lasso with simultaneous updates for groups of parameters (groupwise lasso) is available in package grplasso; the grpreg package implements a number of other group penalization models, such as group MCP and group SCAD. The L1 regularization path for generalized linear models and Cox models can be obtained from functions available in package glmpath, the entire lasso or elastic-net regularization path (also in elasticnet) for linear regression, logistic and multinomial regression models can be obtained from package glmnet. The penalized package provides an alternative implementation of lasso (L1) and ridge (L2) penalized regression models (both GLM and Cox models). Package RXshrink can be used to identify and display TRACEs for a specified shrinkage path and to determine the appropriate extent of shrinkage. Semiparametric additive hazards models under lasso penalties are offered by package ahaz. A generalisation of the Lasso shrinkage technique for linear regression is called relaxed lasso and is available in package relaxo. Fisher's LDA projection with an optional LASSO penalty to produce sparse solutions is implemented in package penalizedLDA. The shrunken centroids classifier and utilities for gene expression analyses are implemented in package pamr. An implementation of multivariate adaptive regression splines is available in package earth. Variable selection through clone selection in SVMs in penalized models (SCAD or L1 penalties) is implemented in package penalizedSVM. Various forms of penalized discriminant analysis are implemented in packages hda, rda, and sda. Package LiblineaR offers an interface to the LIBLINEAR library. The ncvreg package fits linear and logistic regression models under the the SCAD and MCP regression penalties using a coordinate descent algorithm. High-throughput ridge regression (i.e., penalization with many predictor variables) and heteroskedastic effects models are the focus of the bigRR package. An implementation of bundle methods for regularized risk minimization is available form package bmrm.

https://cran.r-project.org/web/views/MachineLearning.html

Agenda - Day2

❶ Descriptive Data Analysis

....

(b) Computing basic statistics

(c) Building a score card model

❷ Predictive Modeling

(a) General workflow for setting up statistical models

(b) Supervised machine learning (Example: Churn)

(c) Probability/regression-based models (Example: CLV, Win-Back)

(d) A/B testing

(e) Enhancing the workflow through control structures

(f) Unsupervised machine learning (Example: Customer Segmentation)

1. Descriptive analyses

We will discuss three major points on descriptive analyses with R

(a) Computing basic statistics

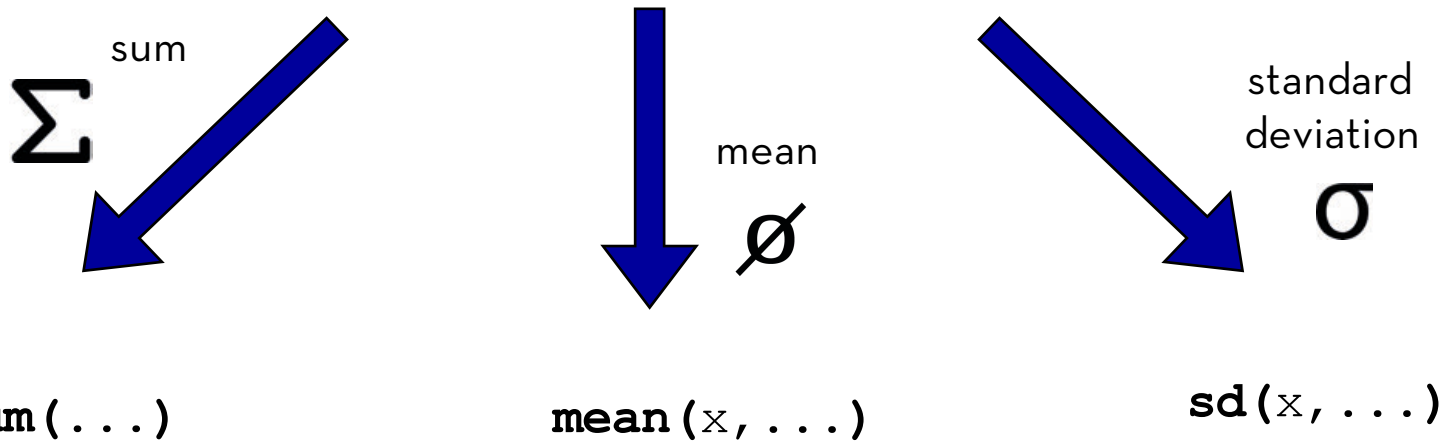
(b) Creating plots

(c) Building a score card model

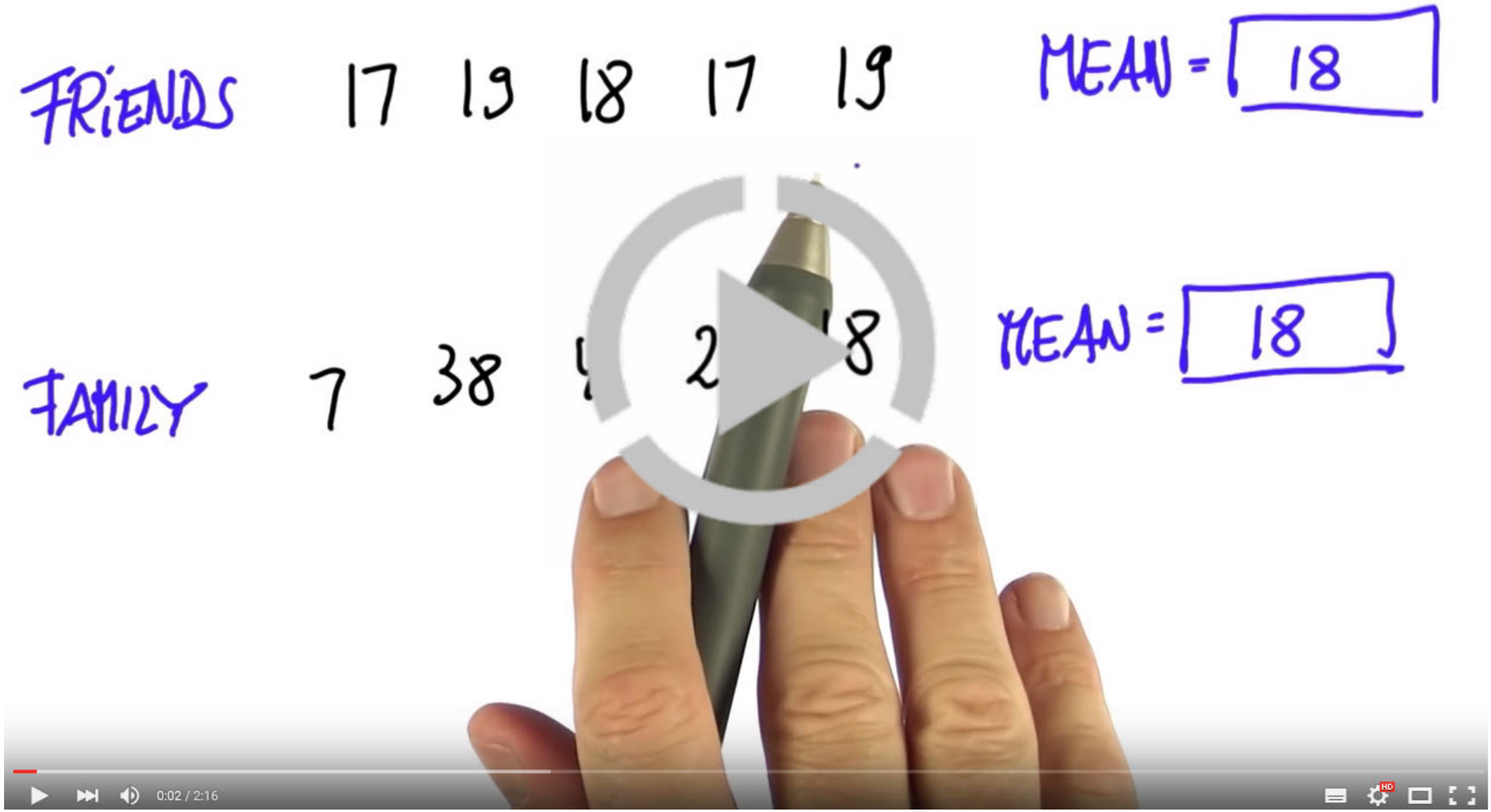
(a) Computing basic statistics

Standard Descriptives

Customer	TransDate	Quantity	PurchAmount	Cost
80365	19.12.2010	1	79.95	25.99
80365	19.12.2010	1	69.95	28.00
42886	21.12.2010	1	349.95	227.00
84374	18.04.2011	1	249.95	105.80
42291	03.05.2011	1	49.95	25.77
...	...	...	...	...

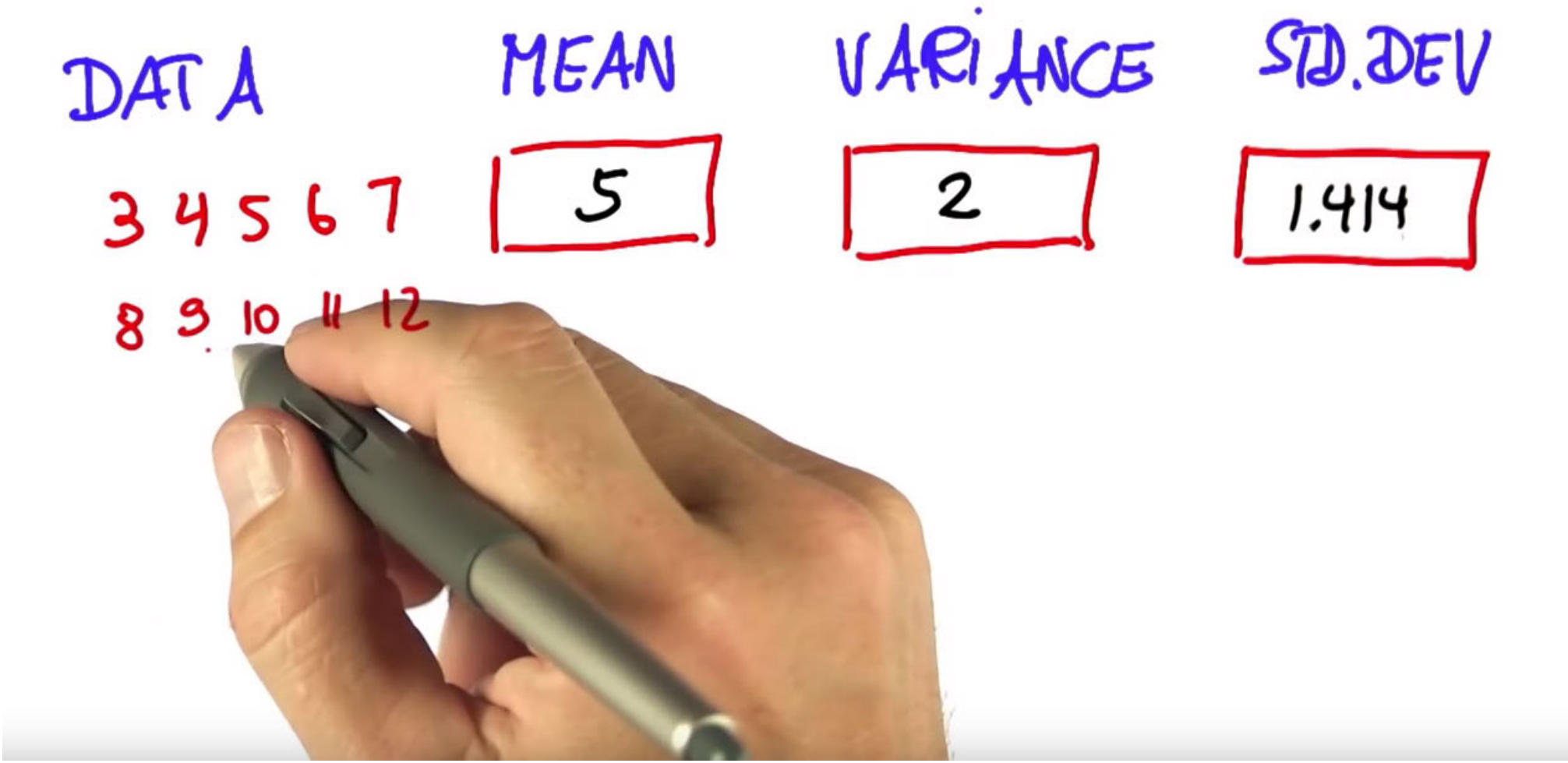


Let's recap those fundamental concepts



Udacity (2015)

Let's recap those fundamental concepts – SOLUTION



Udacity (2015)

Short data summary

Computes the following basic statistics per column:

- min()/max()
- 1st/3rd Quartile
- median()
- mean()

```
summary(mydata)
```

Generate a monthly summary

2009
2010
2011
2012
2013

annual revenue

2010-01
2010-02
2010-03
2010-04
2010-05

monthly revenue

data.table object

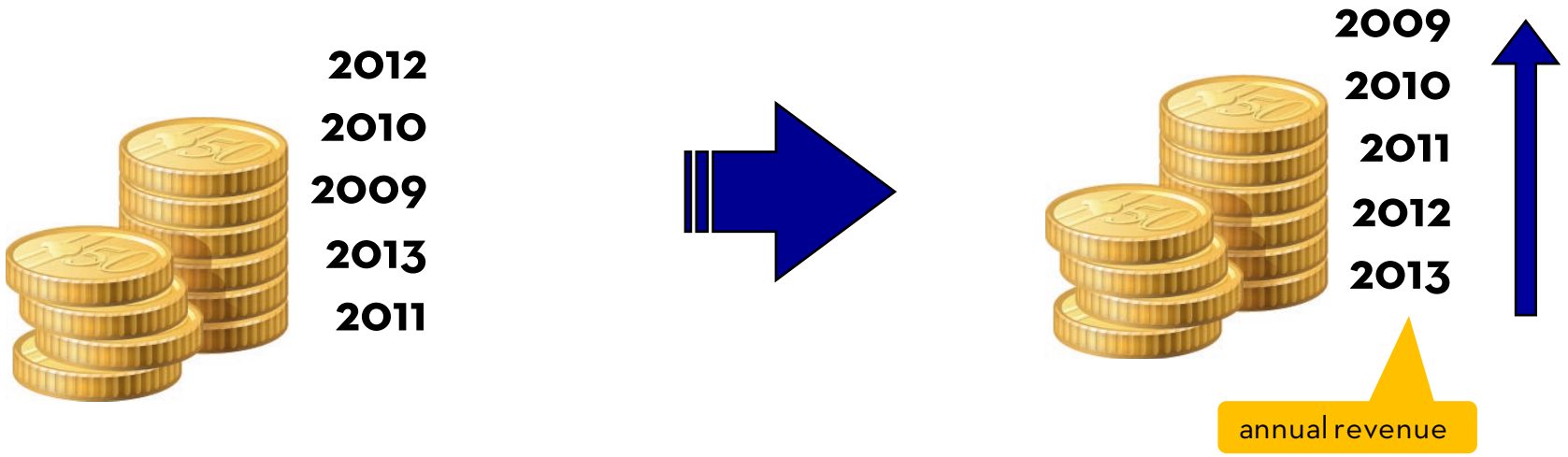
Use "year" to get a monthly summary.

```
mydata[,  sum(PurchAmount),
          by=(Date=floor_date(TransDate,  unit="month"))]
```

command part of lubridate package

Use "month" to get a monthly summary.

Order your results



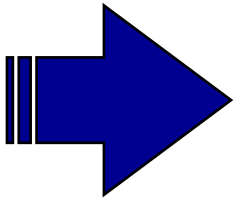
data.table object

sort criteria

```
yearly.summary[order(-year)]
```

Change column names in data.table objects

year	V1
2009	2477088
2010	19.12.2010
2011	2713028
2012	2777414
2013	1313296
...	...



year	Revenue
2009	2477088
2010	19.12.2010
2011	2713028
2012	2777414
2013	1313296
...	...

data.table object | old column name | New column name

```
setnames[yearly.summary, „V1“, „Revenue“]
```

(b) Creating plots

Sidenote: Package “ggplot2”

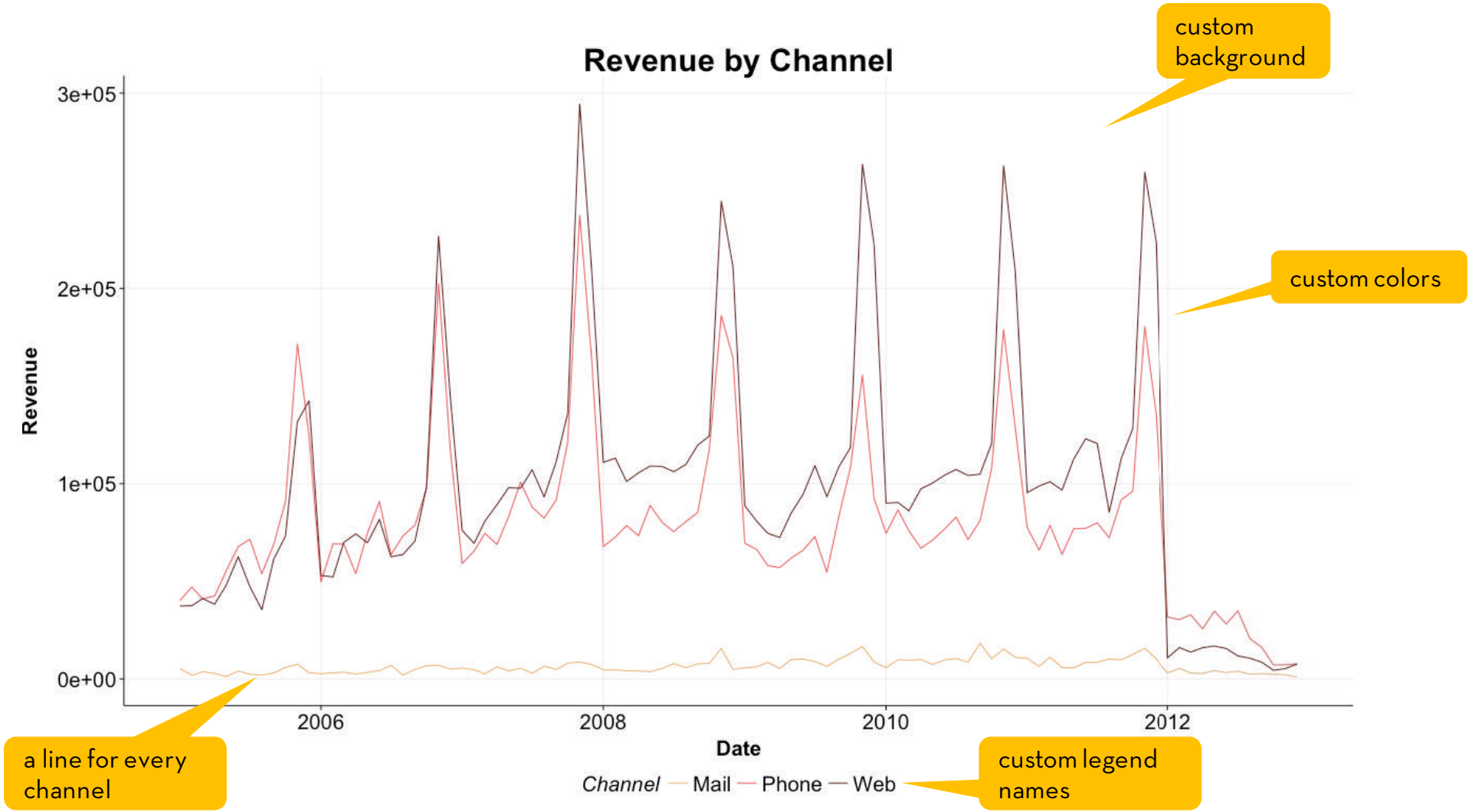
“ggplot2” is a plotting system for R

- Provides a variety of different plots
- Easily adjust plot style to your needs

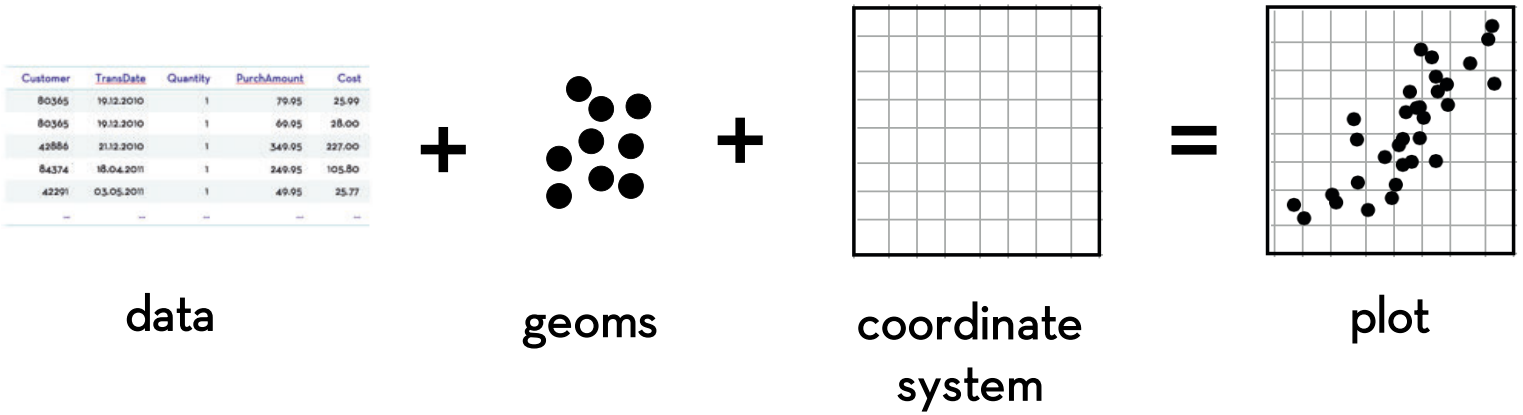
R Code

```
#Install the package “ggplot2”.
install.packages(”ggplot2”)

#Load the package “ggplot2”.
library(ggplot2)
```

Plot: monthly revenue by channel



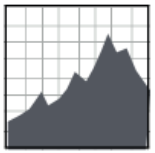
ggplot2: basic components



- **data:** data set and variables
- **geoms:** visual marks to represent the data points
- **coordinate sytem:** location to place the points

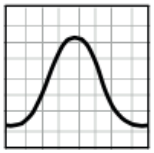
ggplot2: one variable graphs

continuous



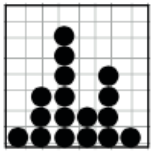
geom_area(stat=bin)

x, y, alpha, color, fill, linetype, size



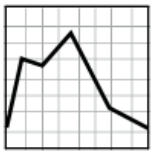
geom_density(kernel="gaussian)

x, y, alpha, color, fill, linetype, size, weight



geom_dotplot()

x, y, alpha, color, fill



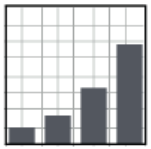
geom_freqpoly()

x, y, alpha, color, linetype, size

geom_histogram(binwith = 5)

x, y, alpha, color, fill, linetype, size, weight

discrete



geom_bar()

x, y, alpha, color, fill, linetype, size

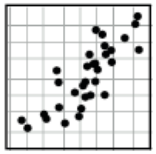
data and variables

```
p <- ggplot(data,  aes(x)
          + geom_line(...)
```

Line plot

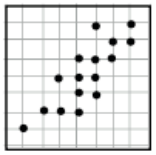
ggplot2: two variable graphs

continuous-continuous



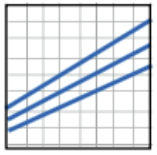
geom_jitter()

x, y, alpha, color, fill, shape, size



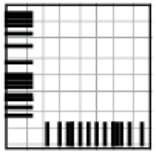
geom_point()

x, y, alpha, color, fill, shape, size



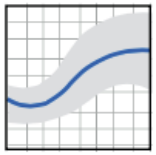
geom_quantile()

x, y, alpha, color, linetype, size, weight



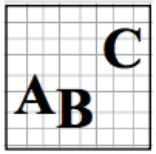
geom_rug()

alpha, color, linetype, size



geom_smooth(method = lm)

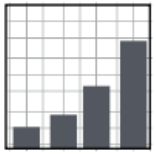
x, y, alpha, color, fill, linetype, size, weight



geom_text(label = *VariableName*)

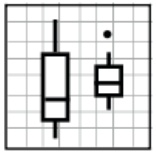
x, y, alpha, angle, color, family, fontface, hjust linehight, size, vjust

discrete-continuous



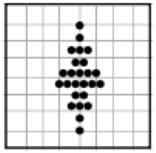
geom_bar(stat = “identity”)

x, y, alpha, color, fill, linetype, size, weight



geom_boxplot()

x, y, alpha, color, fill, size



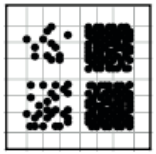
geom_dotplot(binaxis = “y”)

x, y, alpha, color, fill

geom_violin(scale = “area”)

x, y, alpha, color, linetype, size

discrete-discrete

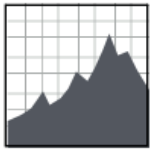


geom_jitter()

x, y, alpha, color, fill, shape, size

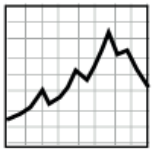
ggplot2: two variable graphs

continuous functions



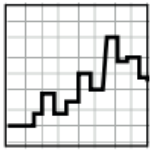
geom_area()

x, y, alpha, color, fill, linetype, size



geom_line()

x, y, alpha, color, linetype, size



geom_step()

x, y, alpha, color, linetype, size

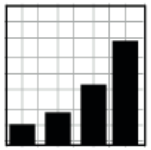
Maps

geom_map(aes(map_id=state), map = map) +
expand_limits(x=map$long, y=map$lat)

map_id, alpha, color, fill, linetype, size

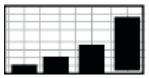
ggplot2: coordinate systems

Coordinate Systems



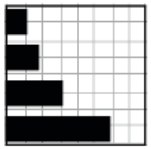
coor_cartesian(xlim=c(0,))

xlim, ylim



coor_fixed(ratio=1/2)

ratio, xlim, ylim

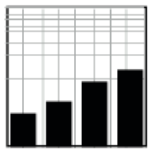


coord_flip()

xlim, ylim

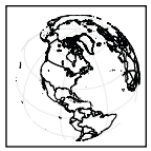
coord_polar(ytrans="sqrt")

theta, start, direction



coord_trans()

ytrans, ytrans, limx, limy

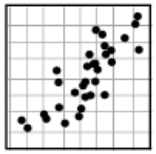


coord_map(projection="ortho", orientation=c(41,-74,0)

projection, orientation, xlim, ylim

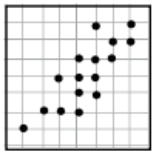
ggplot2: coordinate systems

continuous-continuous



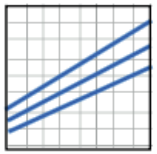
geom_jitter()

x, y, alpha, color, fill, shape, size



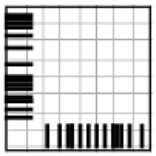
geom_point()

x, y, alpha, color, fill, shape, size



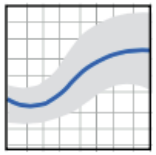
geom_quantile()

x, y, alpha, color, linetype, size, weight



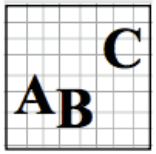
geom_rug()

alpha, color, linetype, size



geom_smooth(method = lm)

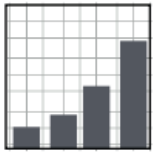
x, y, alpha, color, fill, linetype, size, weight



geom_text(label = *VariableName*)

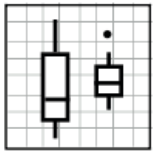
x, y, alpha, angle, color, family, fontface, hjust linehight, size, vjust

discrete-continuous



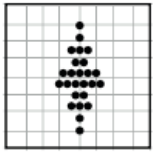
geom_bar(stat = “identity”)

x, y, alpha, color, fill, linetype, size, weight



geom_boxplot()

x, y, alpha, color, fill, size



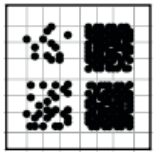
geom_dotplot(binaxis = “y”)

x, y, alpha, color, fill

geom_violin(scale = “area”)

x, y, alpha, color, linetype, size

discrete-discrete

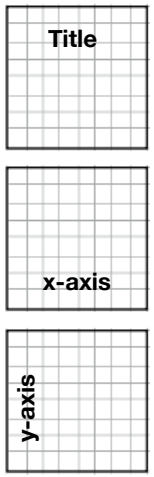


geom_jitter()

x, y, alpha, color, fill, shape, size

ggplot2: lables

Labels

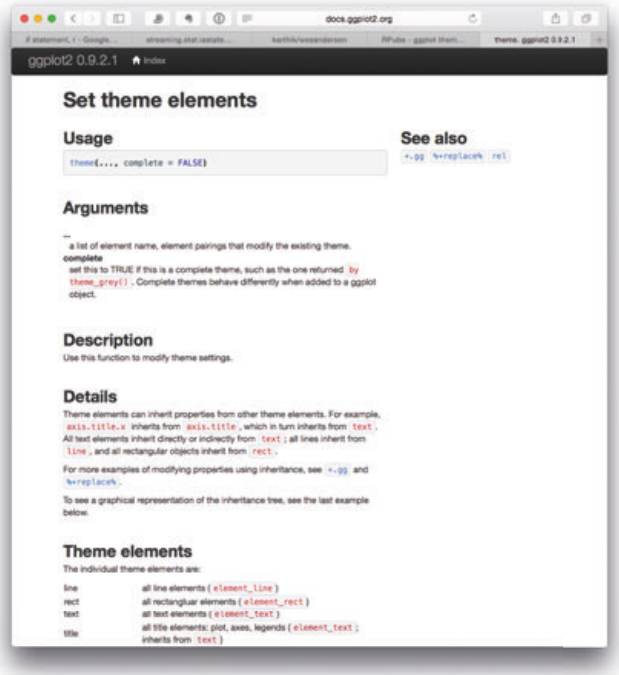


ggtitle("Title")

xlab("x-axis")

ylab("y-axis")

Use **`theme()`** to customize the style of your graph.
http://docs.ggplot2.org/0.9.2.1/theme.html



ggplot2: compose the layers

Every element of the plot is a layer

```
p <- ggplot(....)  + geom_line(...)
p <- p + ggtitle(...)  + xlab(...)  + ylab(...)
p <- p + theme(...)
p
```

"base" plot

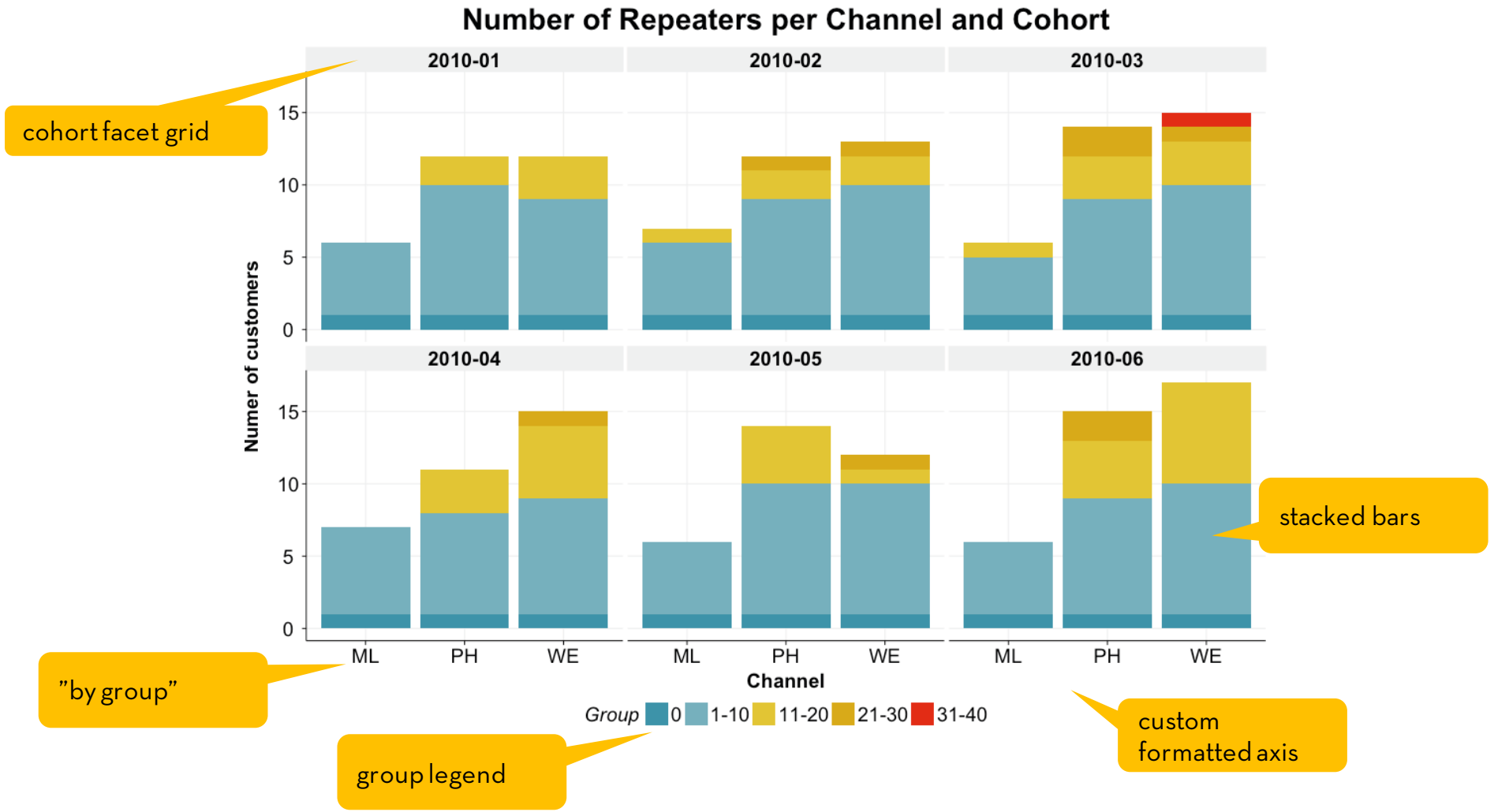
lines (here for a line plot)

Title and axis

Additional themes (e.g, font-sizes, ...)

view the plot

Which channel is best to acquire long-term customers?



Setkey on column

Customer	TransDate	Quantity	PurchAmount	Cost
80365	19.12.2010	1	79.95	25.99
80365	19.12.2010	1	69.95	28.00
42886	21.12.2010	1	349.95	227.00
84374	18.04.2011	1	249.95	105.80
42291	03.05.2011	1	49.95	25.77
...	...	...	...	...

the column gets sorted by reference

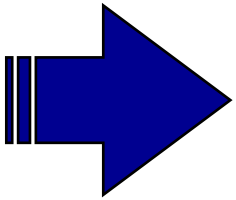
data.table object

```
setkey(mydata,  „Customer")
```

command part of data.table package

Count row numbers by customer

Customer	TransDate	CohortDate
80365	19.12.2010	2008-11
80365	19.12.2010	2008-11
42886	21.12.2010	2008-12
84374	18.04.2011	2008-04
42291	03.05.2011	2008-05
...	...	...



Customer	CohortDate	x
80365	2008-11	2
42886	21.12.2010	1
84374	18.04.2011	1
42291	03.05.2011	1
...	...	...

transaction count

data.table object

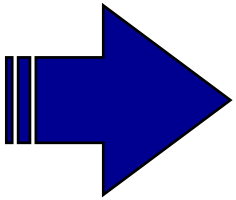
```
mydata[, list(x=.N, CohortDate), by=CUSTOMER]
```

returns number of rows

Note: setkey() on CUSTOMER is required.

Create “bins”

Customer	x
80365	10
42886	2
84374	1
42291	11
...	...



Customer	x	Group
80365	10	1-10
42886	2	1-5
84374	1	6-10
42291	11	>10
...	...	...

bins

```
transactions[,  Group:=cut(transactions$x,
                breaks=c(1,  5,  10,  max(transactions$x)),
                include.lowest=TRUE,  right=FALSE
                labels  =  c(”1-5",  ”6-10",  ”>10"))
```

Sidenote: Package “ggthemes”

Additional themes for the “ggplot2” package

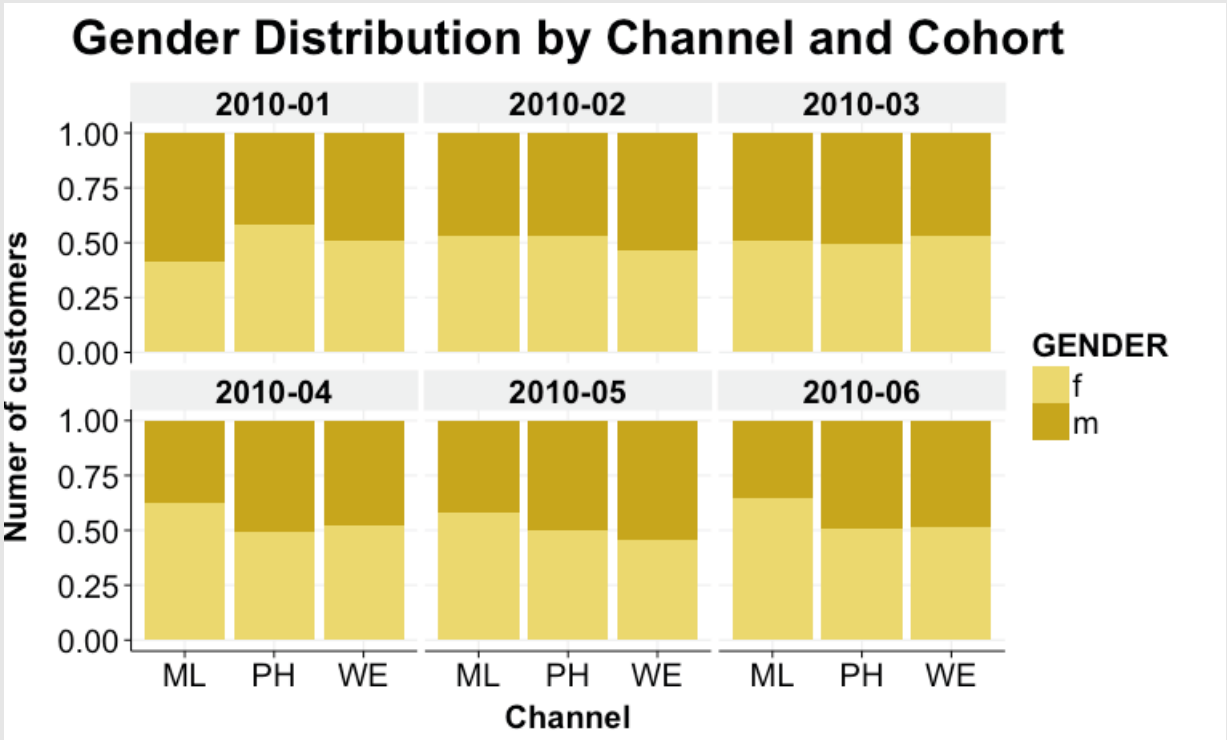
- Additional plots
- Additional themes

R Code

```
#Install the package “ggtheme”.
install.packages(”ggtheme”)

#Load the package “ggtheme”.
library(ggtheme)
```

Exercise
Plotting

Create the following plot:

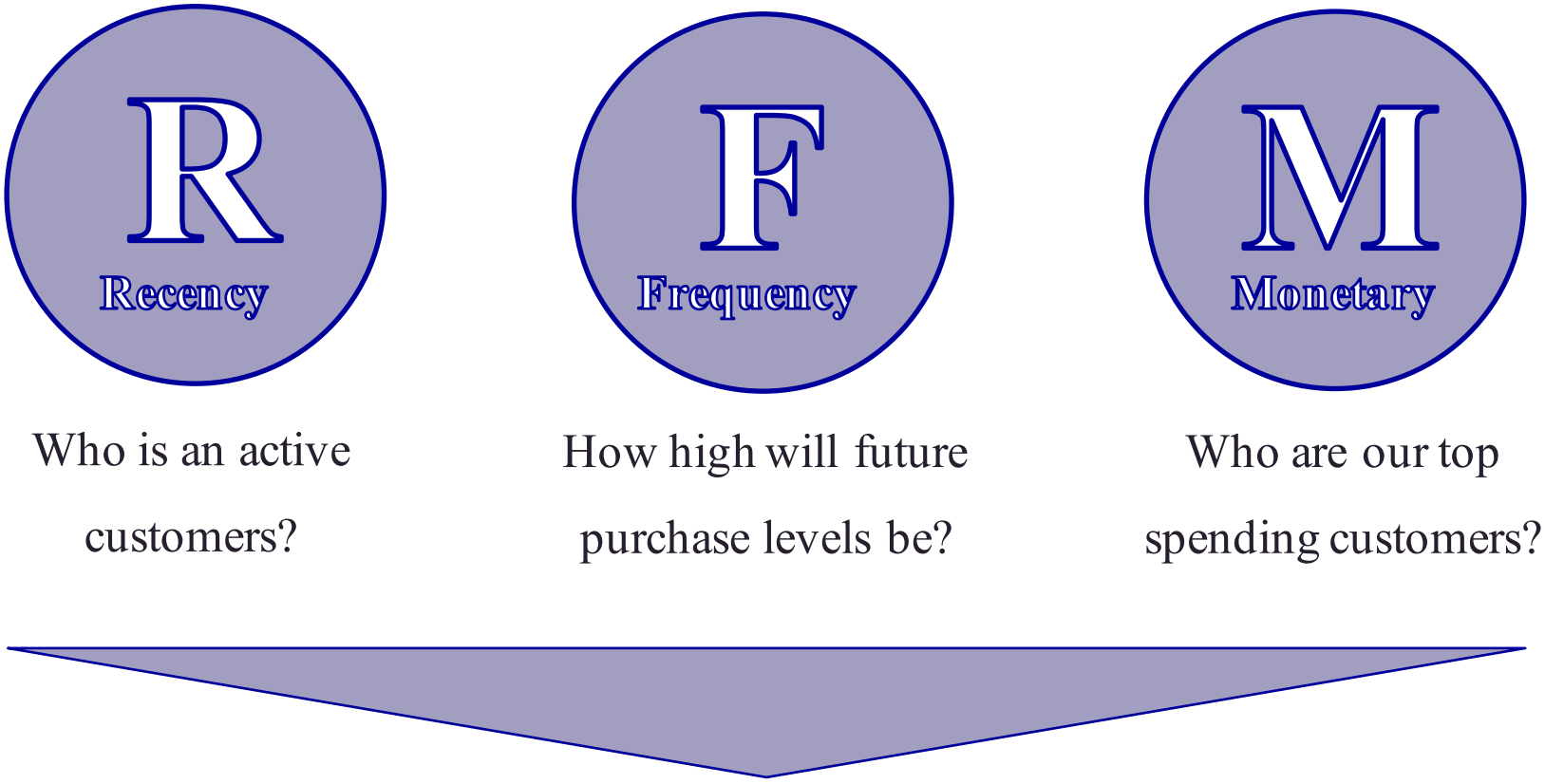


Brackets – Round, squared or curly?

- () Round brackets are used to specify the arguments of a functions.
- [] Single quared brackets are used to reference members of a R objects (e.g. a vector, data.frame, data.table, matrix, or list).
- [[]] Double squared brackets are used to select elements in more or less the same way as single brackets. The difference between single and double is that with double squared brackets no element names are displayed.
- {} Curly brackets are used to group a set of R code statements, e.g. to specifiy the commands that make up a user-defined function.

(c) Building a scoring model

Developing a scoring model to identify the best customers

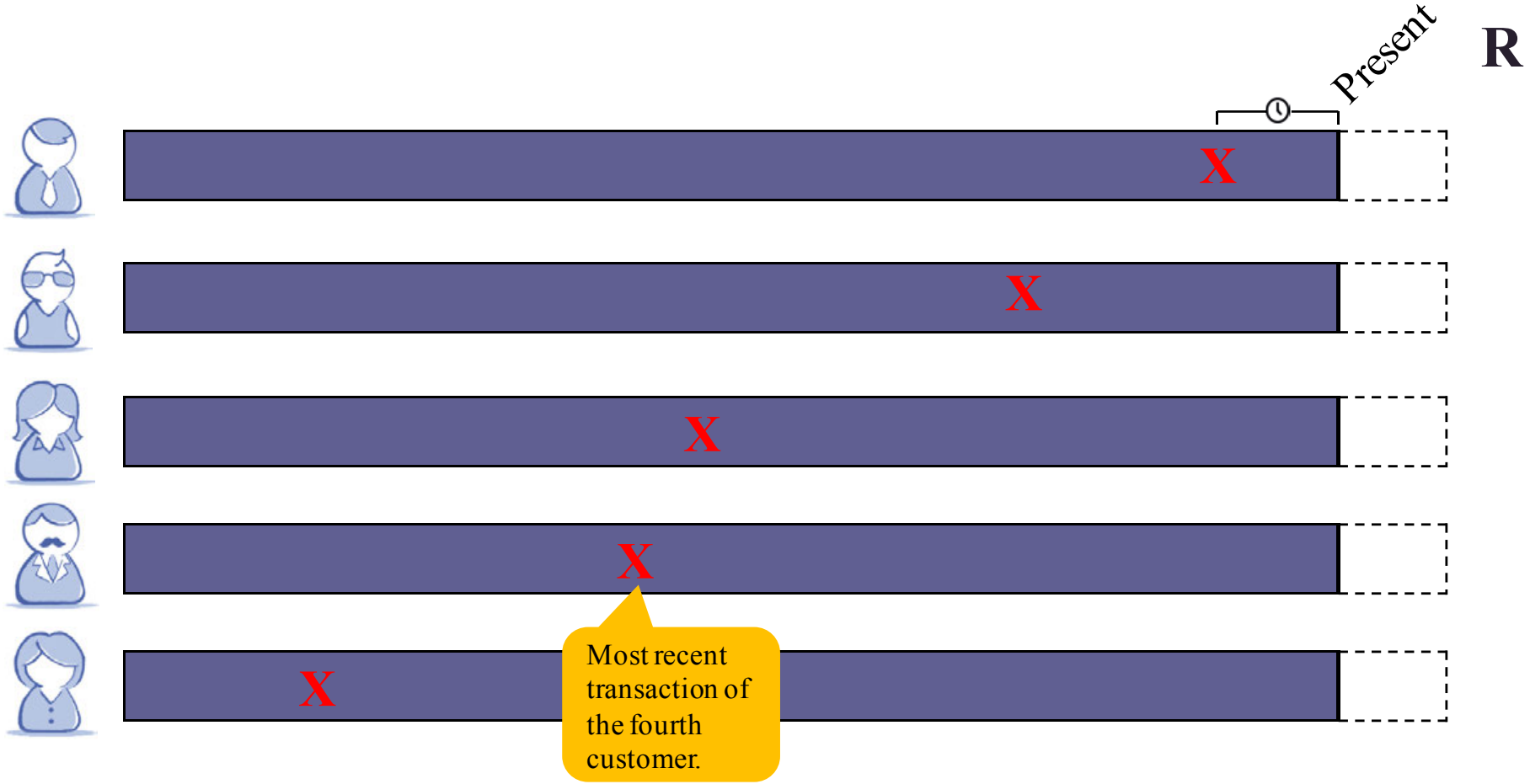


Determining the value of the customer with the RFM framework

- Set a scale to scale to score customers on. For example: 1(worst) – 5 (best)
- **Step 1**: Order and score the customers according to the recency of their last purchase.
- **Step 2**: Order and score the customers according to their frequency of purchases.
- **Step 3**: Order and score the customers according to their average spending per transaction.
- Define the relative weights of each of the categories (R, F, M).
- **Step 4**: Apply the weightings to receive the customers RFM score.

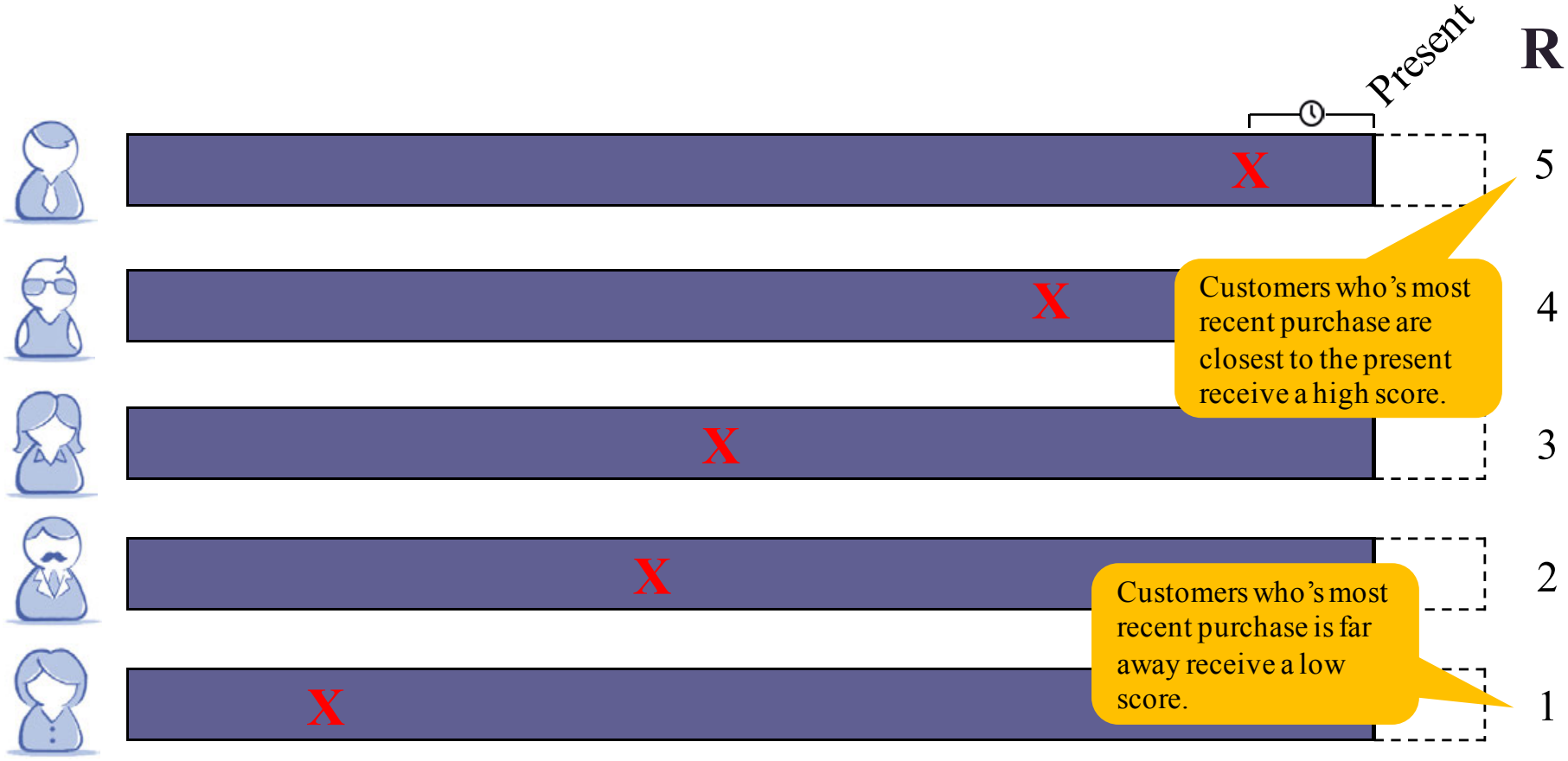
Valuing customers using the RFM framework: Recency

- **Step 1**: Order and score the customers according to the recency of their last purchase.



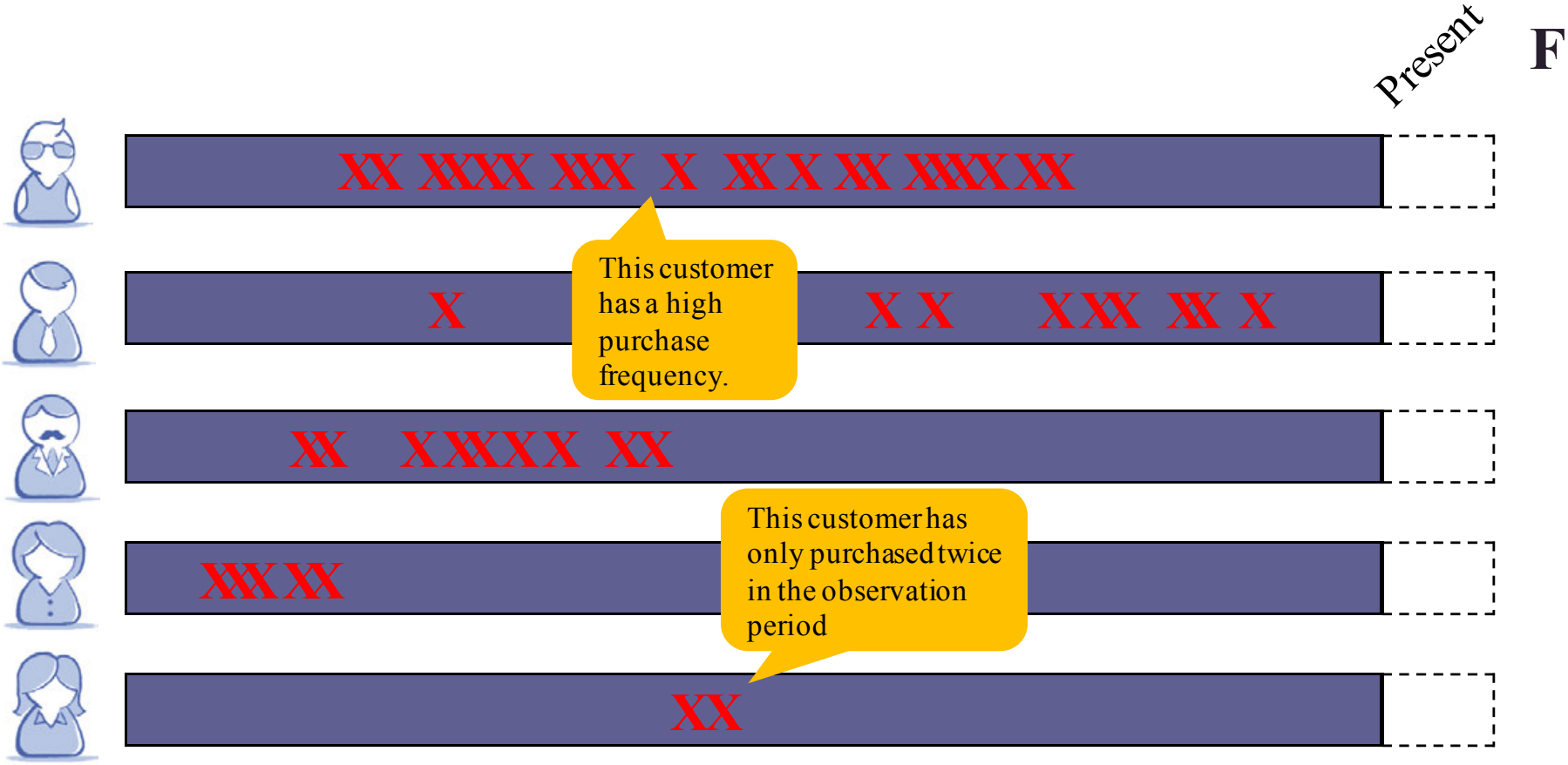
Valuing customers using the RFM framework: Recency

- **Step 1**: Order and score the customers according to the recency of their last purchase.



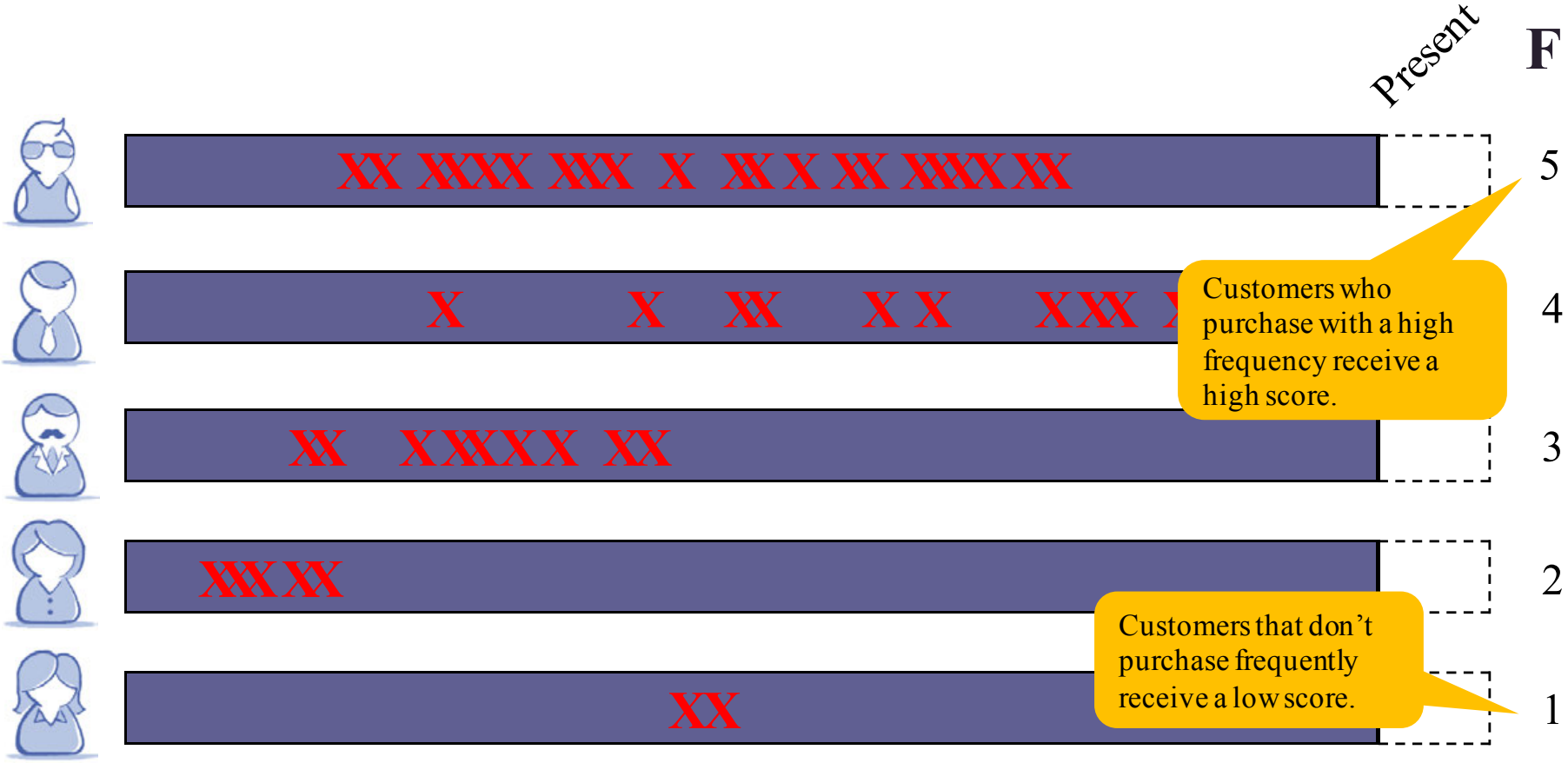
Valuing customers using the RFM framework: Frequency

- **Step 2**: Order and score the customers according to their frequency of purchases.



Valuing customers using the RFM framework: Frequency

- **Step 2**: Order and score the customers according to their frequency of purchases.



Valuing customers using the RFM framework: Monetary value

- **Step 3**: Order and score the customers according to their average spending per transaction.

M

Note: Average spending is used to avoid correlation with frequency.


160$ per transaction


78$ per transaction

This customer spend a large amount per transaction.


54$ per transaction


29$ per transaction


15$ per transaction

Valuing customers using the RFM framework: Monetary value

- **Step 3**: Order and score the customers according to their average spending per transaction.


160$ per transaction — **M** = 5


78$ per transaction — **M** = 4

Customers who spend a lot per transaction receive a high score.


54$ per transaction — **M** = 3


29$ per transaction — **M** = 2

Customers that don't receive a low score.


15$ per transaction — **M** = 1

Valuing customers using the RFM framework: Score summary

	R	F	M
	2	3	5
	1	2	4
	3	1	3
	4	5	2
	5	4	1

Valuing customers using the RFM framework: Defining the weights (OPTIONAL)

- Define the relative weights of each of the categories (R, F, M)
- For example:

R = 100; F = 10; M = 1

This weighting means that recency is most influential on a customers score. Frequency and monetary value follow respectively.

Valuing customers using the RFM framework: RFM score

- **Step 4**: Apply the weightings to receive the customers RFM score.

	R	F	M	RFM score
	2	3	5	235
	1	2	4	124
	3	1	3	313
	4	5	2	452
	5	4	1	541

(1 x 100) + (2 x 10) + (4 x 1) = 124
R F M

Exercise
Extend the RFM model

1. Add weights to the RFM model, thus the final score is a weighted sum of the three components. Use the following weights recency value = 0.5, monetary value = 0.2, frequency = 0.3.
2. Calculate the total revenue and the average revenue for each RFM group. Which group has the highest total revenue, which group the highest average revenue?

2. Predictive Modelling

We will discuss four major points on statistical analyses with R

(a) General workflow for setting up statistical models

(b) Machine Learning – Supervised learning (Example: Churn modelling)

(c) Probability/regression-based models (Example: CLV modelling)

(d) A necessary addition to predictive models: A/B Testing
(Example: Marketing Campaign effectivity)

(e) Enhancing the workflow through control structures

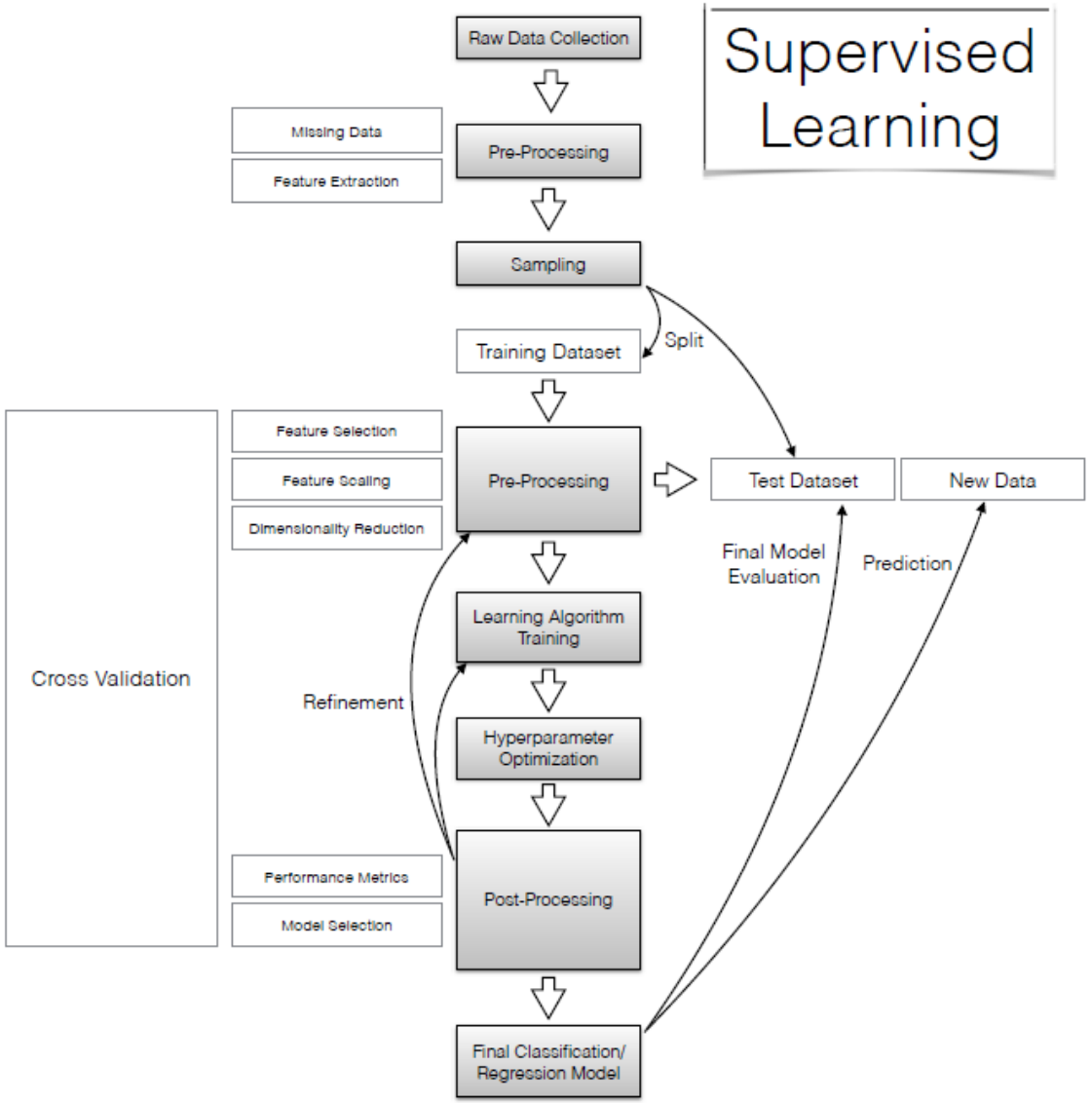
(f) Machine Learning – Unsupervised learning

(a) General workflow

5 steps

1. Question
2. Data Preparation

   (Identify data features)
3. Algorithm
4. Evaluation
5. Deployment



What exactly is the business problem to be solved?

- Which department/group will use the results?
- How exactly needs the result to look like that they can be used by this department/group?

Data Collection

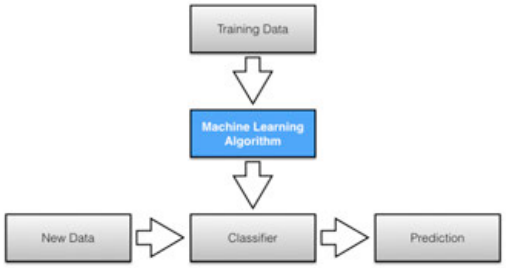
- Compile the necessary data from the **firm's database** and other data sources (such as customer survey data)
- Evaluate the possiblity to add secondary data from **external databases** (e.g. solvency index or geocoding information)

Garbage in, garbage out – Understand and clean the data

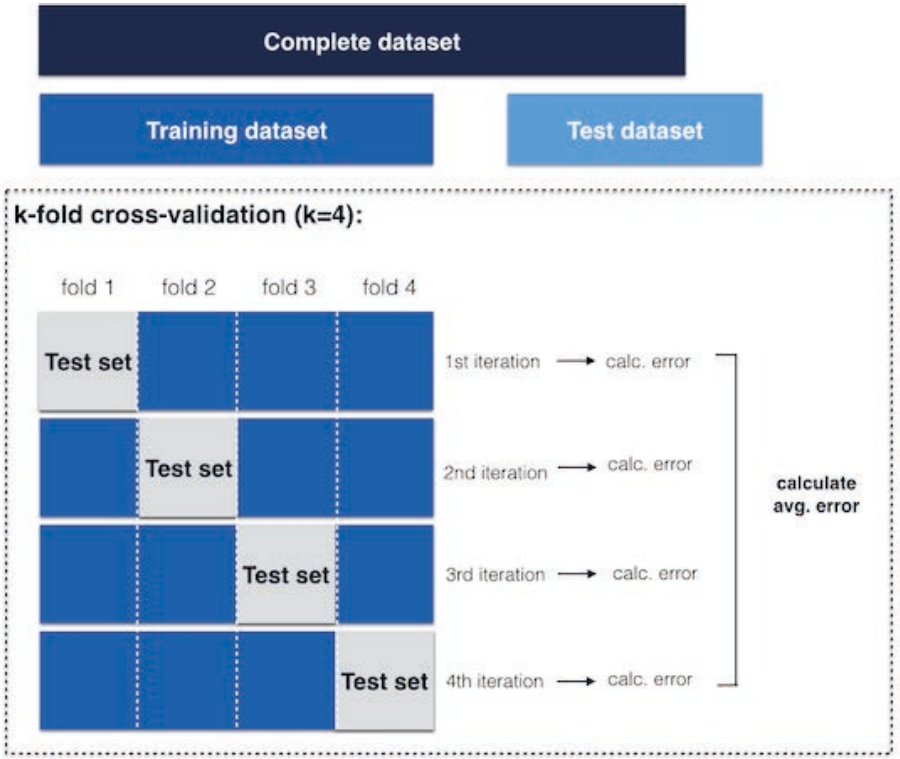
1. Define your **dependent variable**
2. Understand how **each independent variable is measured**
3. **Cleaning** the raw data (e.g. impute or remove observations with missing data)
4. Use **features**
    1. Feature *Extraction* (e.g. by applying a principle component analysis)
    2. Feature *Selection* (e.g. by using correlations to build up a parsimonious model)

Sample your data

- Split the dataset in a **test set** and **training set** to ensure a good out-of sample performance



- Consider applying a **k-fold cross validation**



→ But remember: The purpose of cross-validation is not to come up with our final model!

http://stats.stackexchange.com/questions/52274/choosing-a-predictive-model-after-k-fold-cross-validation

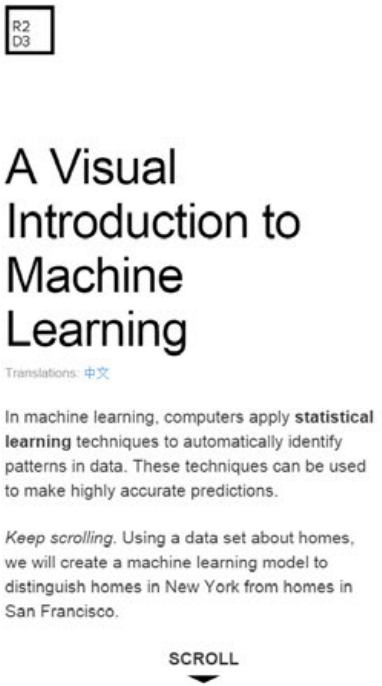
Model estimation usually requires multiple iterations to determine the best algorithm

1. Evaluate multiple algorithms and evaluate their performance
2. Tune the estimation procedure by
    1. adapting the parameters of each particular algorithm
    2. using ensemble methods

→ A good algorithm is
(a) simple,
(b) accurate,
(c) fast,
(d) scalable, and
(e) delivers interpretable results.

http://radar.oreilly.com/2013/09/gaining-access-to-the-best-machine-learning-methods.html

A decision tree is a popular choice, but how does classification with a decision tree work?



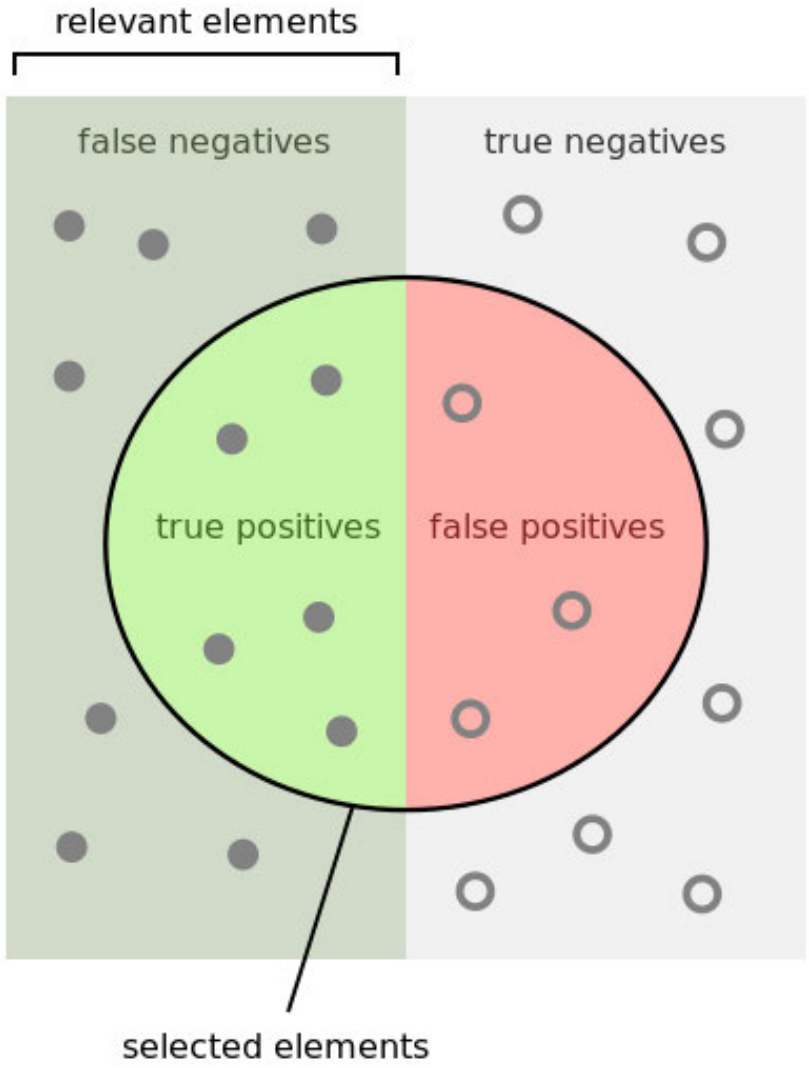
→ Go to **www.r2d3.us** and check out the **visual explanation of a decision tree.**

http://www.r2d3.us/visual-intro-to-machine-learning-part-1/

Evaluating model performance – Precision and accuracy

	Accurate	Inaccurate (systematic error)
Precise		
Imprecise (reproducibility error)		

Evaluating model performance – Precision and recall



How many selected items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{selected elements}}$$

How many relevant items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{relevant elements}}$$

https://en.wikipedia.org/wiki/Precision_and_recall

Evaluating model performance – Confusion matrix: Bringing together accuracy, precision, and recall

Given a confusion matrix, it's easy to compute accuracy, precision and recall:

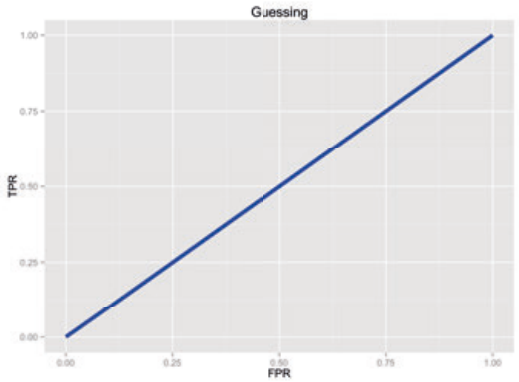
	Predicted class A	Predicted class B	Predicted class C	
Actual class A	50	80	70	200
Actual class B	40	140	120	300
Actual class C	120	220	160	500
	210	440	350	1000

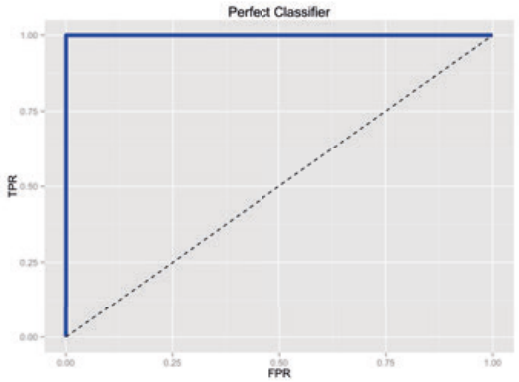
$$Accuracy = \frac{50 + 140 + 160}{1000}$$

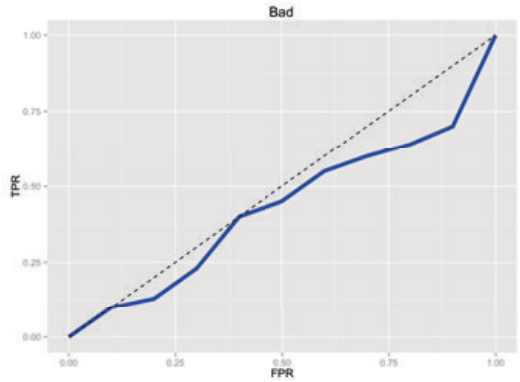
$$Precision_A = \frac{50}{50 + 40 + 120}$$

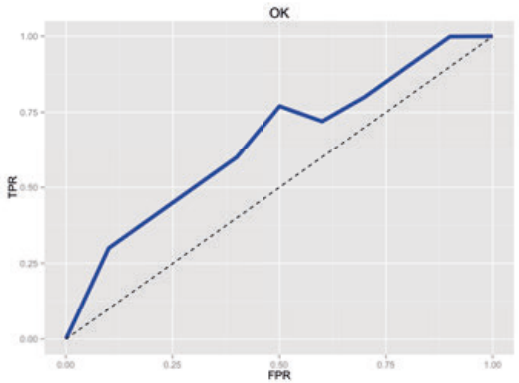
$$Recall_A = \frac{50}{50 + 80 + 70}$$

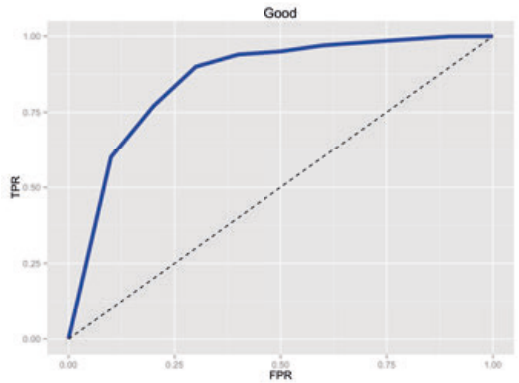
Visualizing model performance – Various ROC curve scenarios (with explanation of AUC)

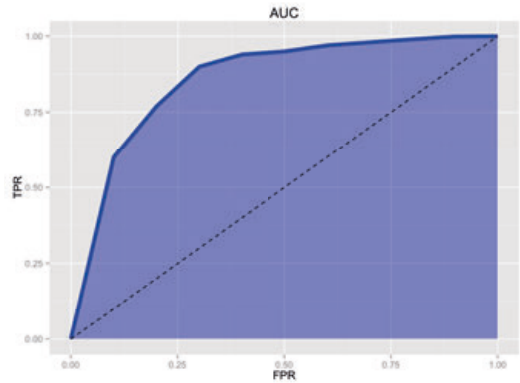












http://blog.yhathq.com/posts/roc-curves.html

Visualizing model performance – Gains and lift charts

Score Table

Target	Score
0	235
1	724
1	556
0	345
0	480
1	676
0	195
1	880
0	368
...	...

Sorted by Score

Target	Score
1	880
1	724
1	676
1	556
0	480
0	368
0	345
0	235
0	195
...	...

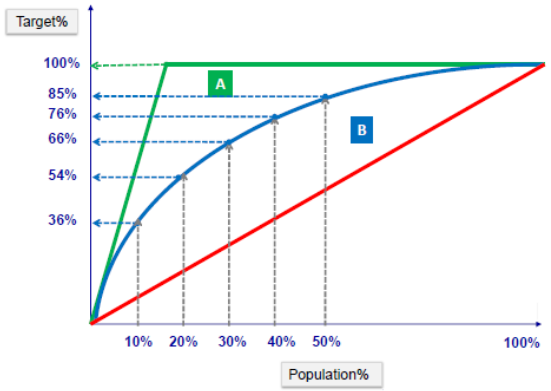
Gain Table

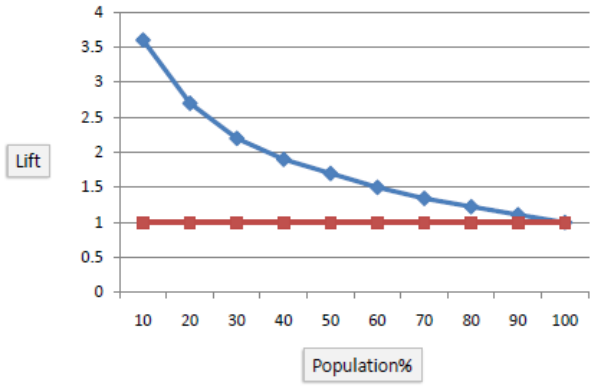
Count%	Target%
10	36
20	54
30	66
40	76
50	85
60	90
70	94
80	98
90	100
100	100

Lift Table

Count%	Lift
10	3.6
20	2.7
30	2.2
40	1.9
50	1.7
60	1.5
70	1.3
80	1.2
90	1.1
100	1

Gains Chart





Lift Chart

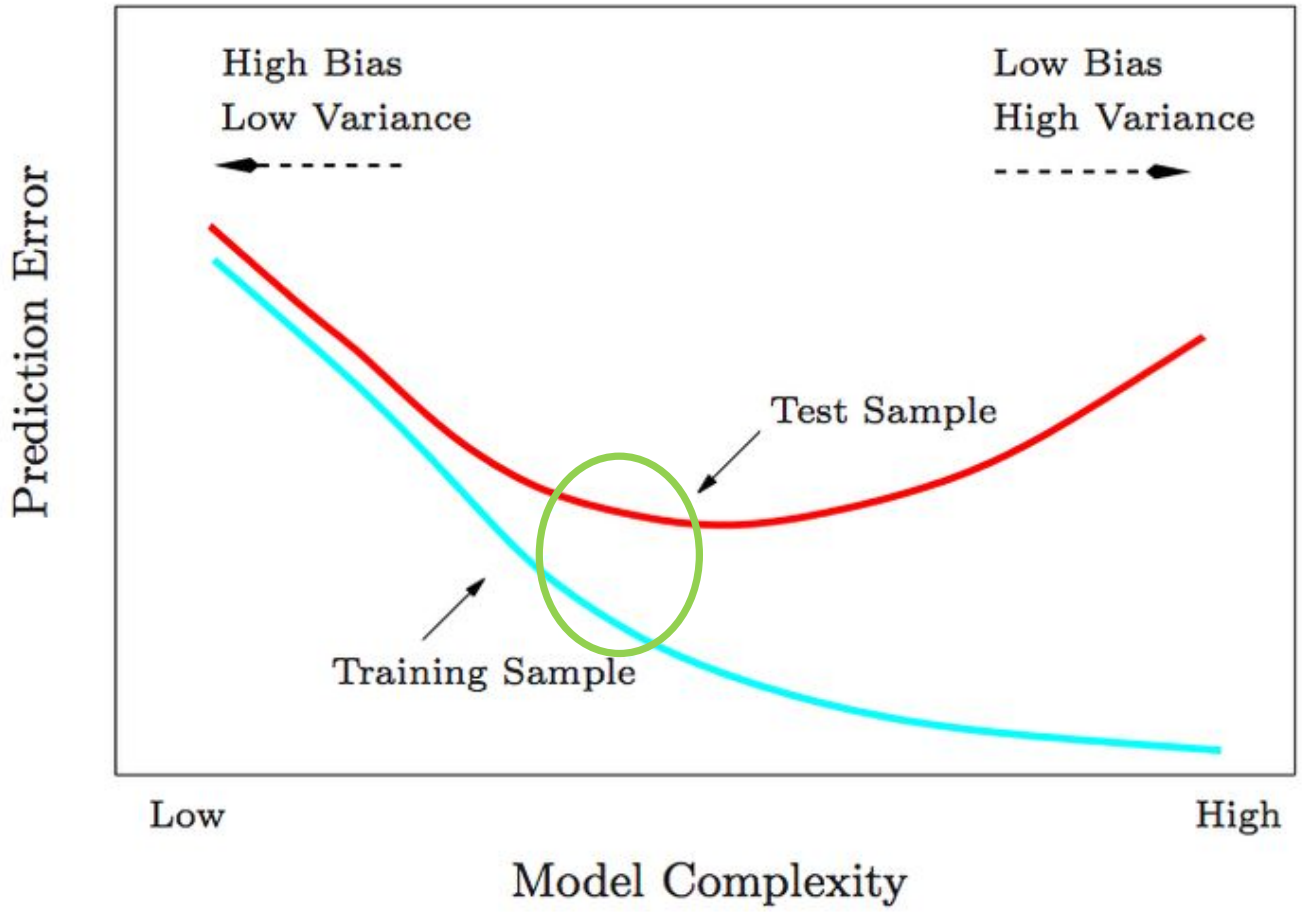
http://www.saedsayad.com/model_evaluation_c.htm

Avoid overfitting to be able to generalize the model results

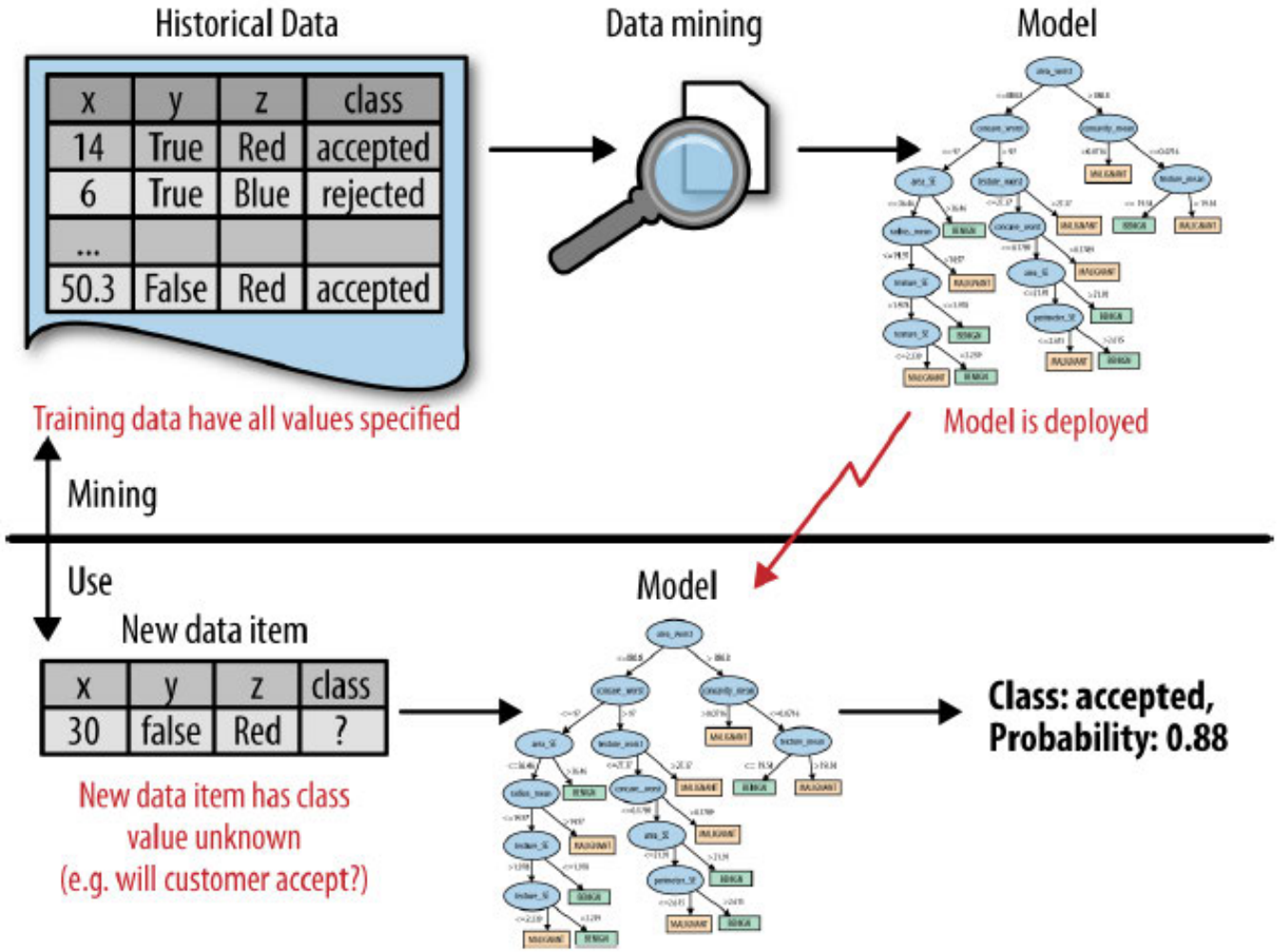
:

https://www.youtube.com/watch?t=79&v=DQWI1kvmwRg

Find the right the right balance between out-of-sample fit and in-sample fit



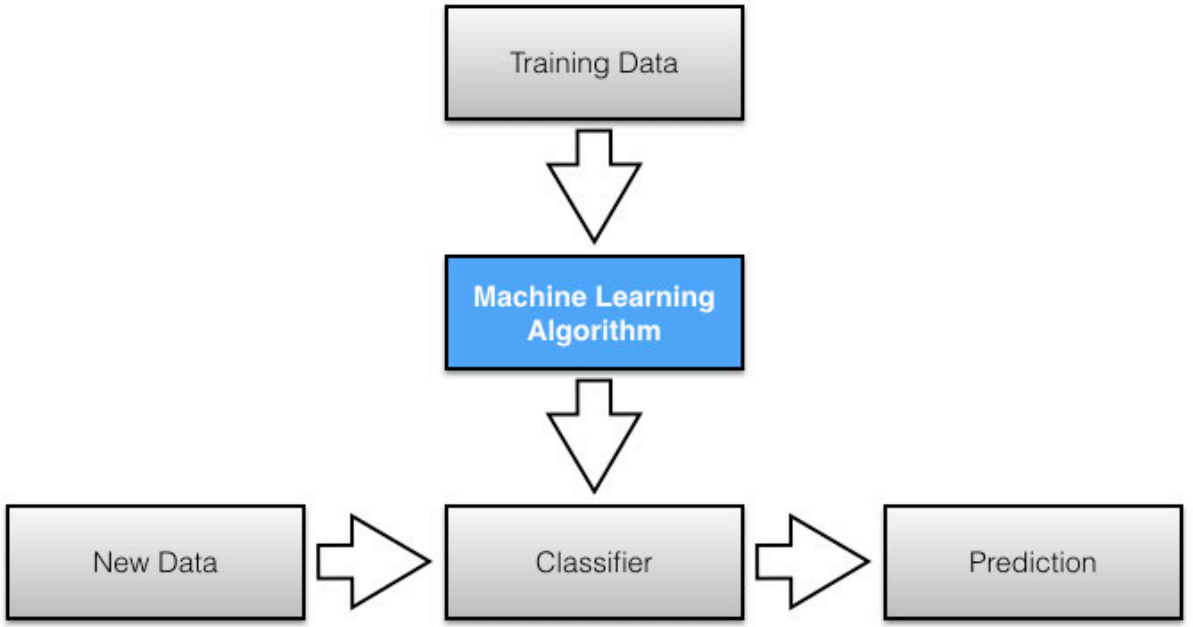
Deploying models



Provost/Fawcett (2013), p. 26

(b) Machine Learning (Supervised)

Let's look at an concrete example how to implement a machine learning model in R



Exercise
Estimate a churn model with supervised ML

1. How would you proceed when you found the best fitting model?
2. Create a churn prediction model for the next quarter and check if the results are comparable.
3. Try if including another variable improves the predictions.

(c) Probability/regression-based models

Apply a customer churn model

We apply the accelerated failure time (AFT) model

dependent variable

$$ln(\text{Duration}_i) = X_i'\beta + \sigma\epsilon_i$$

logarithmic transformation

1. Determine the drivers of customer churn
2. Predict the expected duration of the customers who have yet to churn
3. Determine the predictive accuracy of the model

Input Data for the churn model

B2C business with a sample of 500 customers.

Customer ID	Duration	Censor	Avg_Ret_Exp	Region	Revenue	Rating	Crossbuy	Freq
1	500	0	89.61	1	30.16	1240	6	16
2	730	1	49.89	0	39.80	166	6	10
3	730	1	40.70	0	54.93	1016	2	14
...	...	...	...	...	...	...	...	...

The data is right-censored at 730 days. Therefore we use duration and censor as depended variable in the model

1: customer is in the B2B business, 0: otherwise

total number of purchase occasions

$$ln(\text{Duration}_i, \text{Censor}_i) = X_i'\beta + \sigma\epsilon_i$$

Estimate the model

dependent variables

```
survreg(Surv(Duration,  Censor == 0) ~ Avg_Ret_Exp
                      + Avg_Ret_Exp_SQ
                      + Region
                      + Revenue
                      + Rating
                      + Total_Crossbuy
                      + Total_Freq
                      + Total_Freq_SQ,
                      data = churndata,
                      dist = "lognormal")
```

command part of survival package

squared

Duration is right censored. Therefore we cannot use OLS. We use log-normal.

Estimation results

model estimated

```
Call:
survreg(formula = Surv(Duration, Censor == 0) ~ Avg_Ret_Exp +
    Avg_Ret_Exp_SQ + Region + Revenue + Rating + Total_Crossbuy +
    Total_Freq + Total_Freq_SQ, data = churndata, dist = "lognormal")
                Value Std. Error      z         p
(Intercept)  5.770003   5.22e-02 110.51  0.00e+00
Avg_Ret_Exp  0.008899   8.48e-04  10.50  8.78e-26
Avg_Ret_Exp_SQ -0.000199   7.35e-06 -27.10 9.86e-162
Region      -0.028336   1.91e-02  -1.49  1.37e-01
Revenue      0.003500   5.64e-04   6.21  5.40e-10
Rating       0.000428   2.49e-05  17.20  2.66e-66
Total_Crossbuy  0.097679   6.62e-03  14.74  3.32e-49
Total_Freq   0.027480   6.87e-03   4.00  6.27e-05
Total_Freq_SQ -0.000855   3.05e-04  -2.80  5.04e-03
Log(scale)  -1.844134   4.65e-02 -39.63  0.00e+00

Scale= 0.158

Log Normal distribution
Loglik(model)= -1397.1   Loglik(intercept only)= -1850.9
        Chisq= 907.67 on 8 degrees of freedom, p= 0
Number of Newton-Raphson Iterations: 10
n= 500
```

not significant

Which are the drivers for customer churn?

```
summary(est.results)
```

Interpret the model parameters

Log-linear format: we can calculate the change of an independent variable of in by:

hazard model

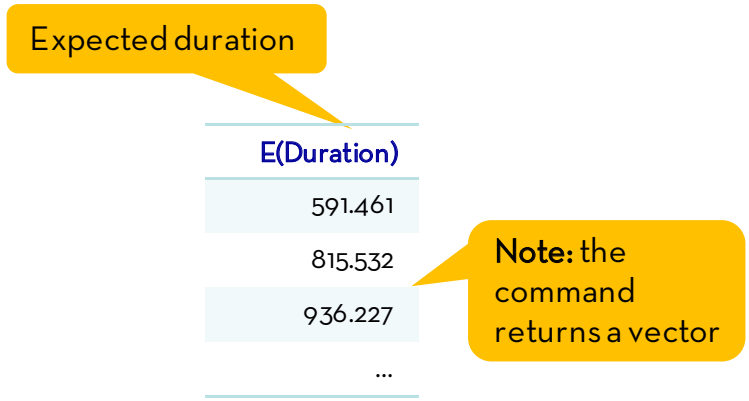
model parameter

$$\frac{T(X_i + 1)}{T(X_i)} = exp(\beta)$$

Variable	Duration ratio
Avg_Ret_Exp	(0.08890.0004*Avg_Ret_Exp)
Crossbuy	1.103
Freq	(0.027-0.002*Freq)
Revenue	1.004
Rationg	1.0004

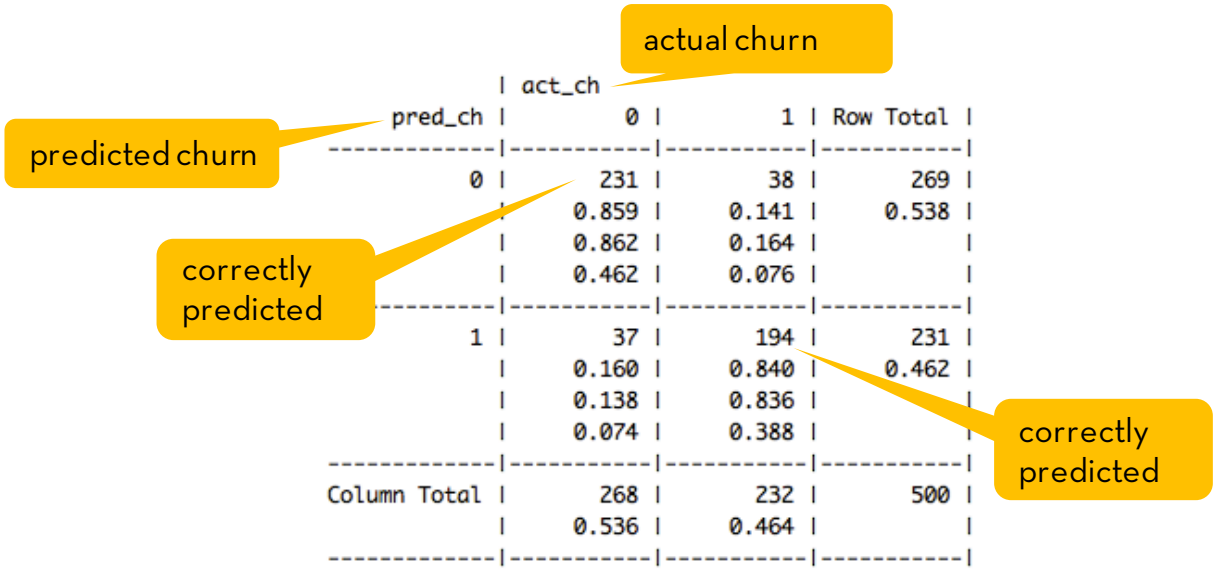
- E.g: If we usually spend 15$ on a given customer, by spending 16$ we get an expected **increase in duration of 1.003** (exp(0.088-0.004*15), which is an **increase of 0.3% for the additional dollar.**

Predict duration




```
predict(est.results,type="response")
```

Predictive accuracy



act_ch			
pred_ch	0	1	Row Total
0	231	38	269
	0.859	0.141	0.538
	0.862	0.164	
	0.462	0.076	
1	37	194	231
	0.160	0.840	0.462
	0.138	0.836	
	0.074	0.388	
Column Total	268	232	500
	0.536	0.464	

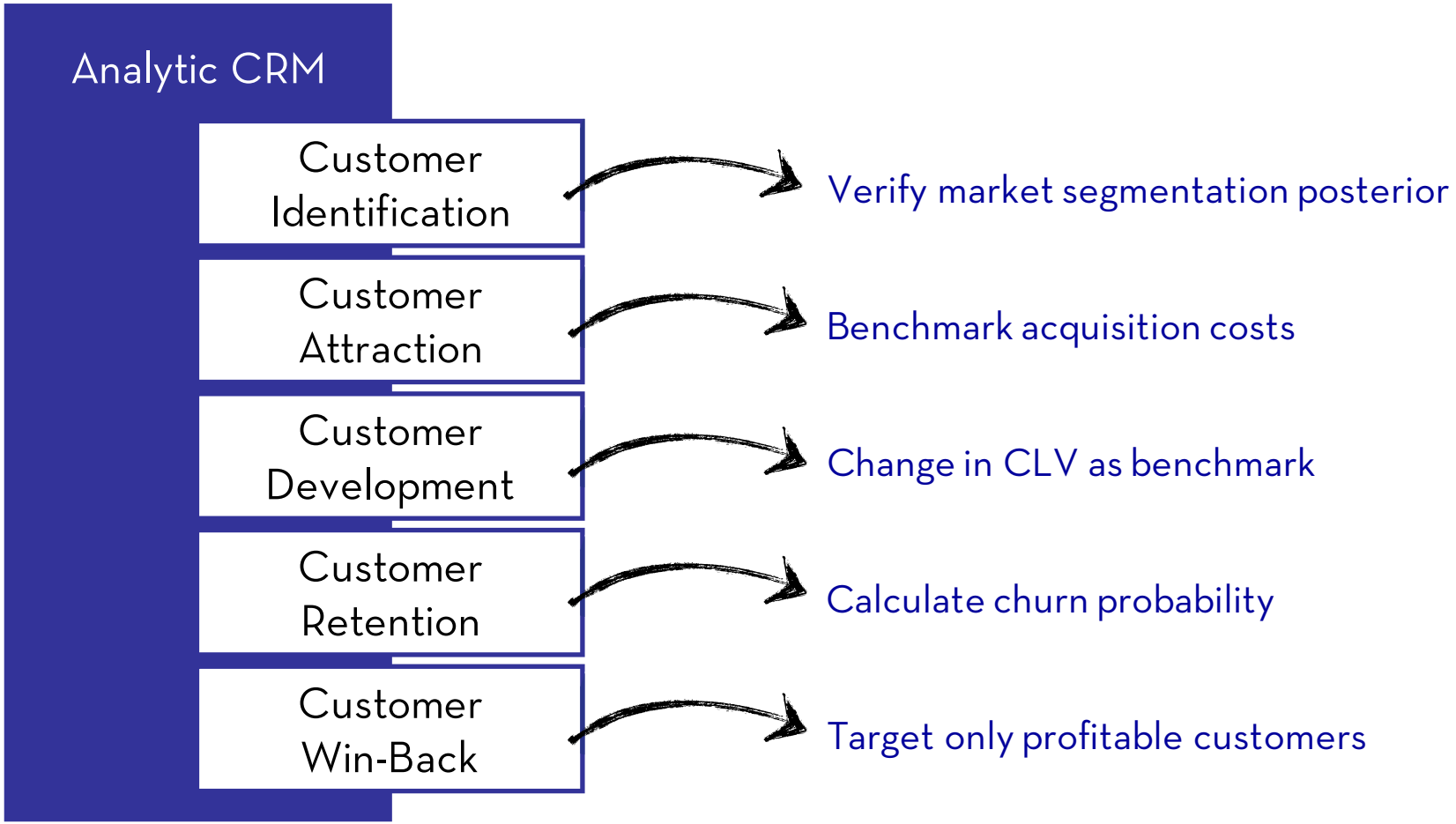
```
pred_ch <- ifelse(predict(est.results,type="response") < 730, 1, 0)
act_ch <- ifelse(churndata$Duration < 730, 1, 0)

CrossTable(pred_ch, act_ch, prop.chisq = FALSE)
```

command part of gmodels package

(c) Probability/regression-based models

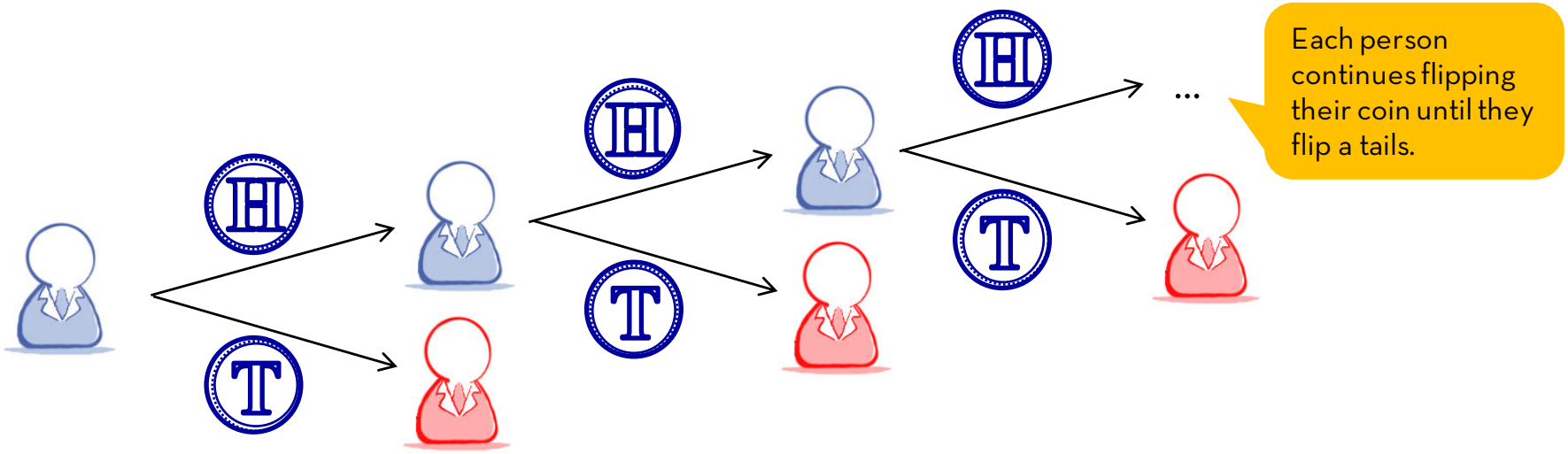
Customer lifetime value: Assign every customer an individual value

Visualizing model performance

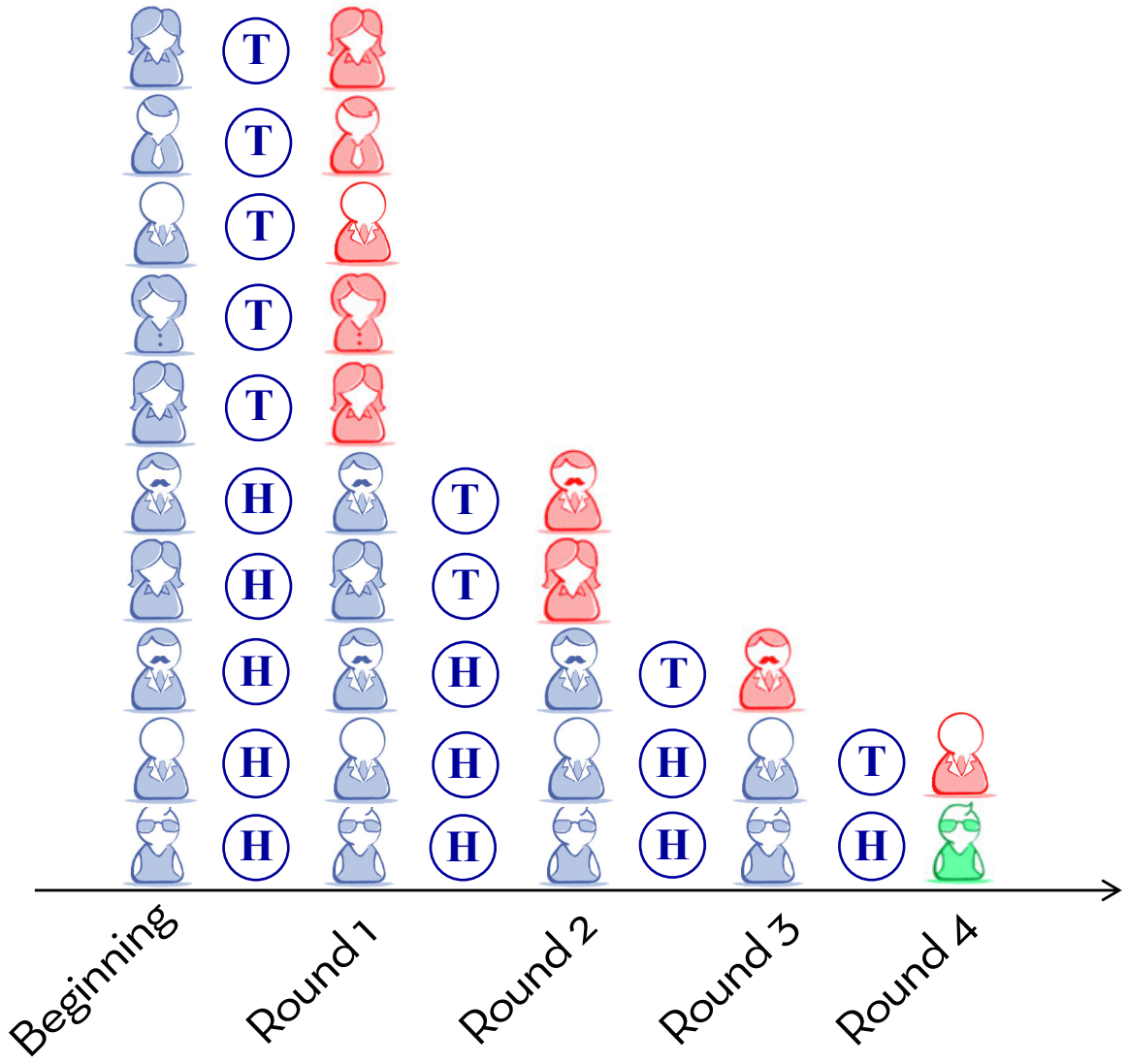


Kumar & Petersen (2012, p. 13-18).

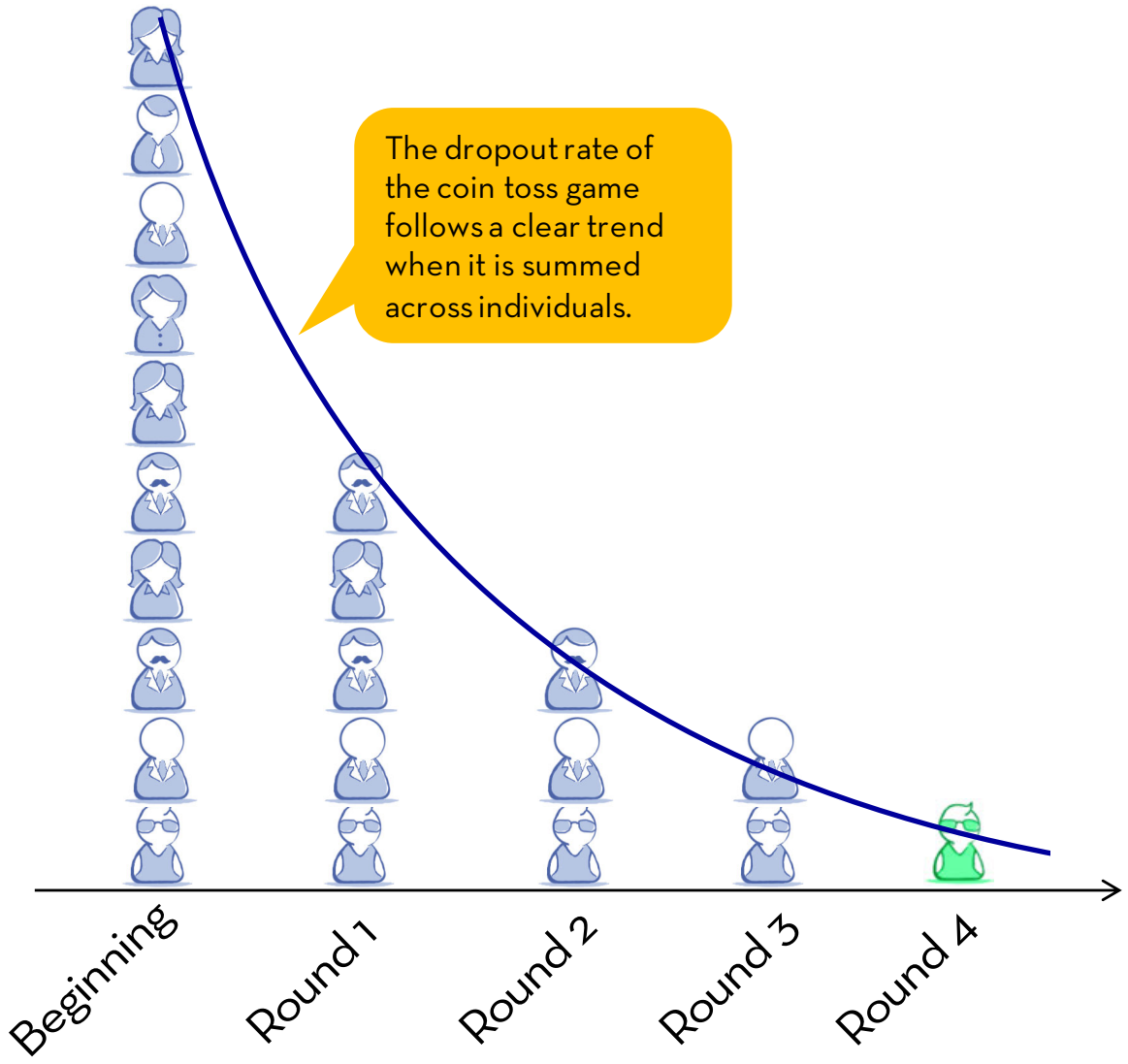
A coin tossing game



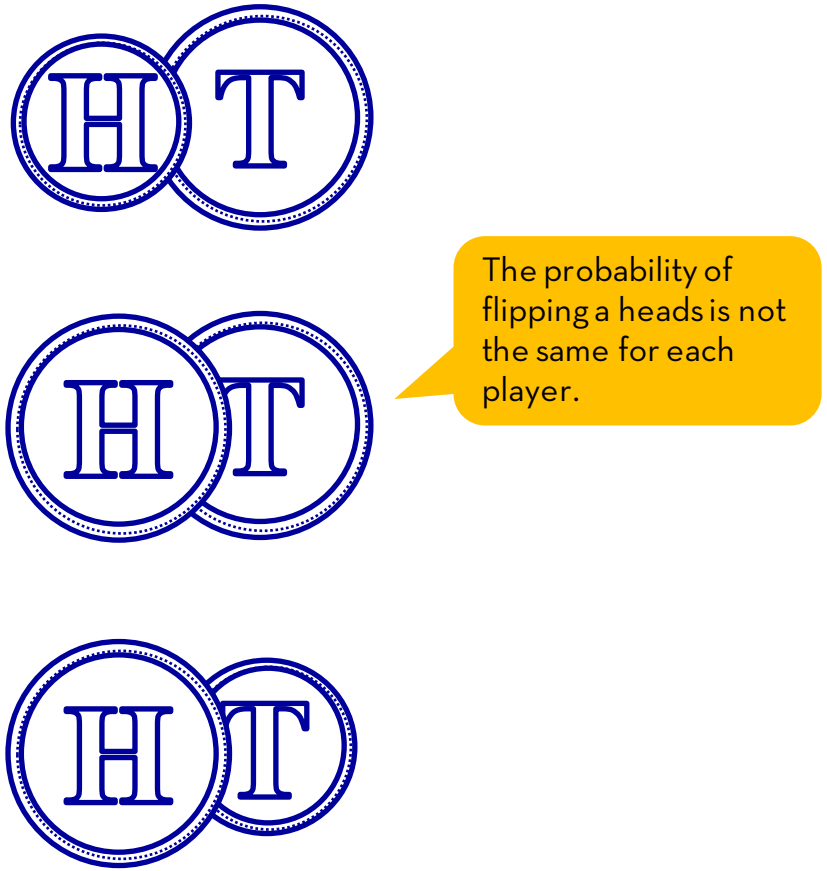
A coin tossing game



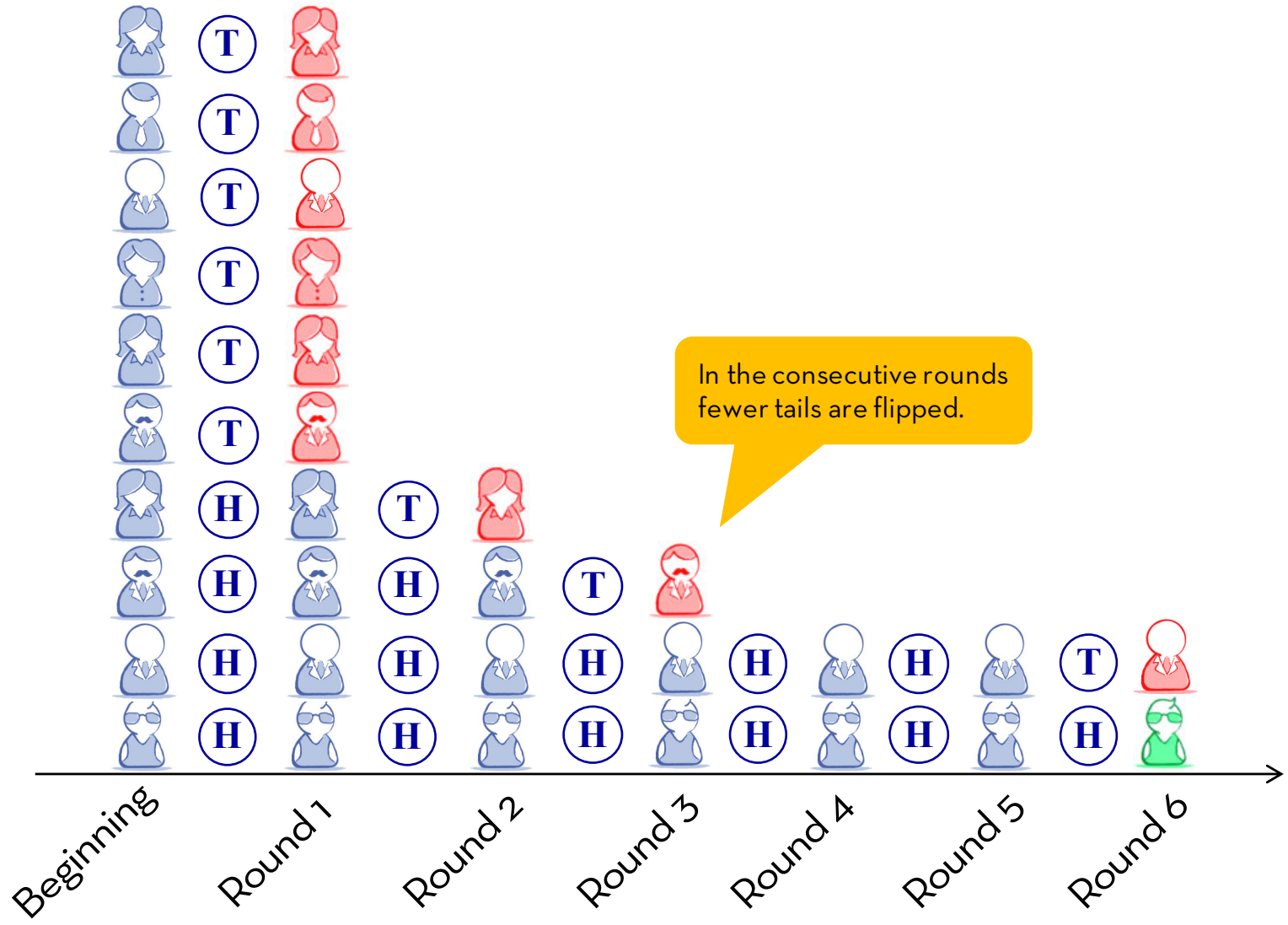
A coin tossing game
The outcome of the game



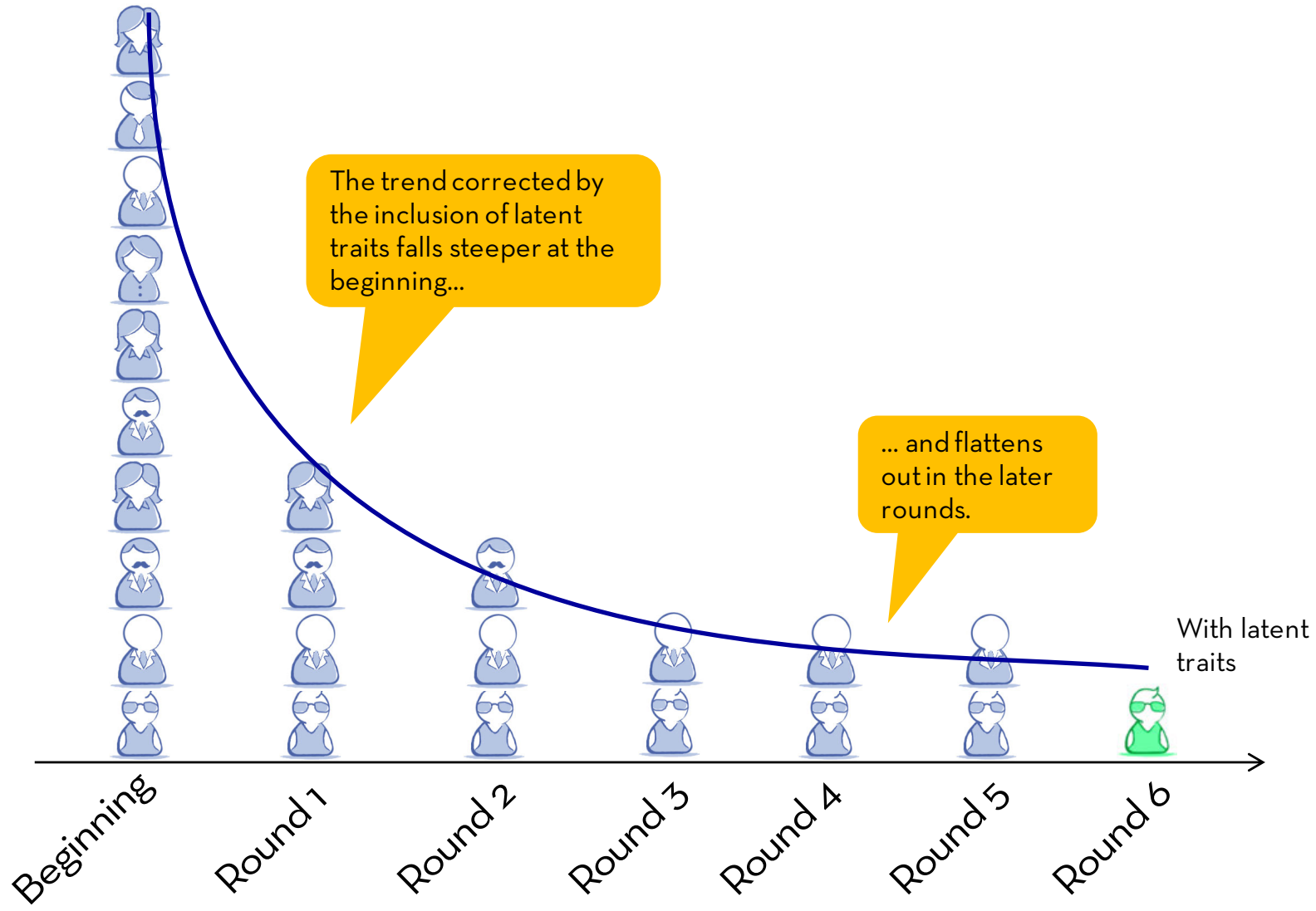
Latent traits are natural occurring differences between people



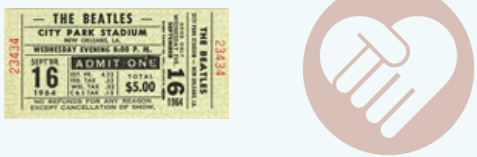
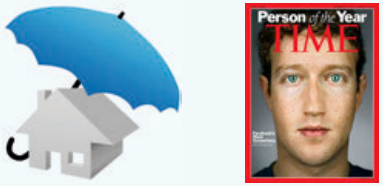
A coin toss game
Playing the game with different sized coins



A coin toss game
The outcome of the game with different sized coins

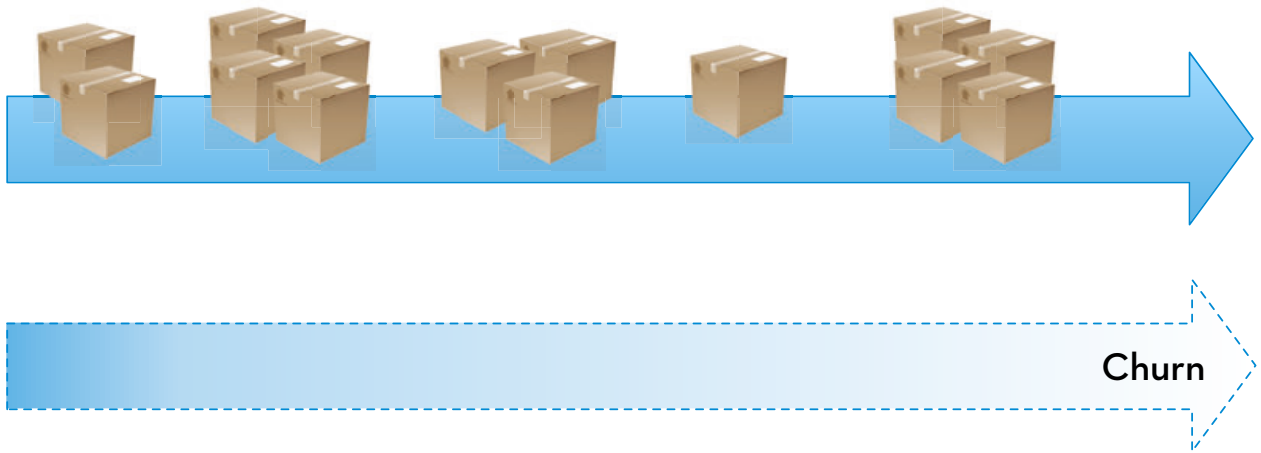


Visualizing model performance

Continuous non-contractual 	Continuous contractual 
Discrete non-contractual 	Discrete Contractual 

Fader&Hardie 2012

Two processes are modeld



- Pareto/NBD Model: “Gold Standard”
- GGompertz/NBD, Normal/NBD, BG/NBD

Bemmaor & Glady (2012), Dipal & Singh (2014, p. 3), Fader, Hardie & Lee (2005).

Different distribution opions

Assumption	Pareto/NBD	
Individual purchase rates	Poisson	NBD
Population mean purchase rates	Gamma	
Individual lifetime probability	Exponential	Pareto
Population mean lifetime probability	Gamma	
Purchases and Lifetimes	Independent	

Bemmaor & Glady (2012), Fader, Hardie & Lee(2005).

Sidenote: shift() in data.table

Note: This feature is only available in version 1.9.5 (only on github)

- shift() allows to create lagged variable
- use: shift(variable, n)

R Code

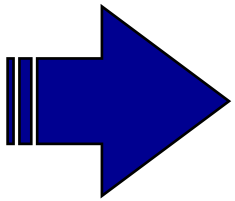
```
#Create a lagged date variable by customer
mydata[, lag.date:=shift(ORDER_DATE, 1), by=CUST_NO]
```

Different distribution opinons

Input

Customer ID	Date	Sales in $
1	01.01.1997	29.33
1	08.01.1997	29.73
1	02.08.1997	14.96
1	12.12.1997	26.48
2	01.01.1997	63.34
2	13.01.1997	11.77
3	01.01.1997	6.79
4	01.01.1997	13.97
...	...	...
1000	25.03.1997	74

Customer "4" purchased on the 1st of January 97 for $ 13.97.



Output

Customer ID	Expected residual transactions	DERT	RLV in $
1	1.8511	0.407	10.22
2	0.2177	0.048	1.80
3	0.1362	0.030	0.20
4	0.1362	0.030	0.42
5	0.1362	0.030	0.72
6	4.7221	1.038	71.80
...	...	...	
	3.2981	0.732	4.38

Customer "6" is expected to make 4.72 transactions (within the defined time horizon of 52 weeks).

Customer "6" is expected to spend $71.80 in the next 52 weeks.

Specify the options

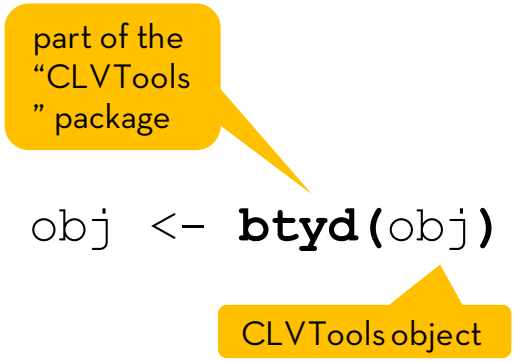
```
obj <- btyd.Data(data,
                 date.format="%Y-%m-%d",
                 aggregate.days=TRUE,
                 id.var="Id",
                 date.var="Date",
                 price.var="Price",
                 cal.duration=54,
                 measurement.scale="week”,
                 repeat.transactions=TRUE)
```

part of the “CLVTools ” package

variable names

training period

Estimate the model parameters



Predict for your customers

part of the "CLVTools" package

```
obj <- predict(obj)
```

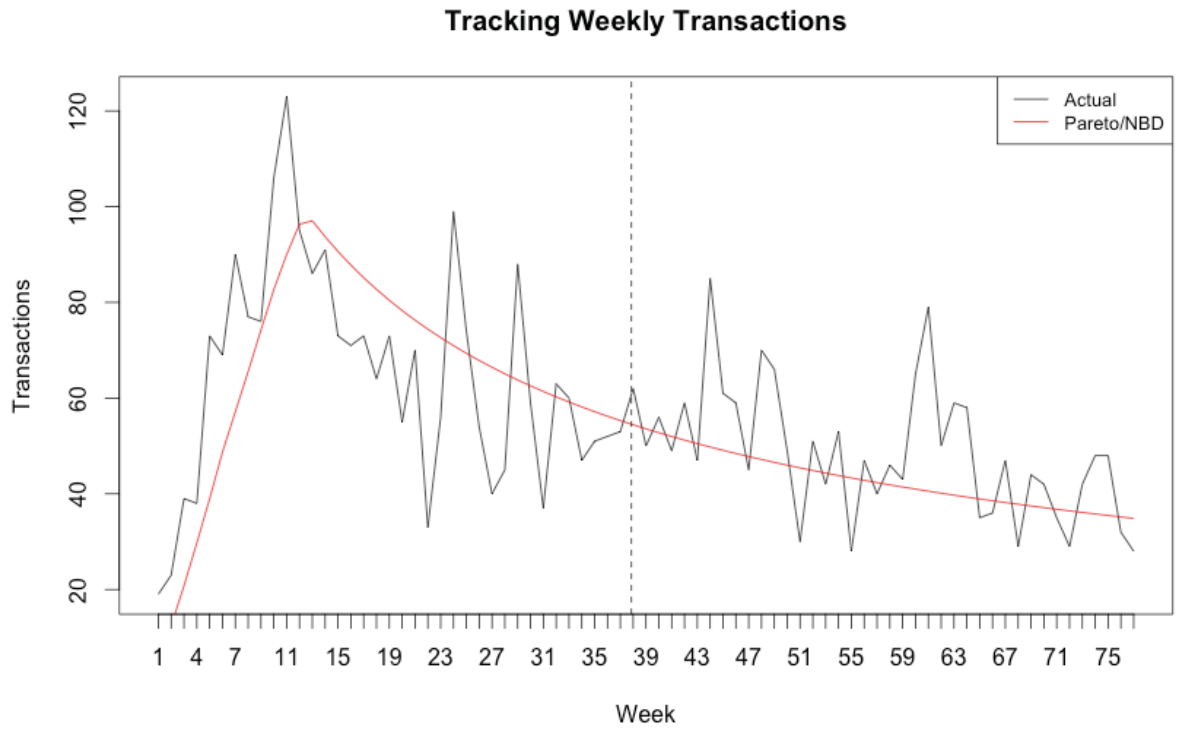
CLVTools object

Customer ID	Expected residual transactions	PAlive	DERT
1	1.8511	0.185	0.407
2	0.2177	0.058	0.048
3	0.1362	0.125	0.030
...	...	...	...

Plot the results

```
obj <- plot(obj, plot.zero=TRUE)
```

CLVTools object



(d) A/B Testing

Understanding campaign performance

- A marketing manager has send a mailing to 40 000 customers.
- 16 675 customers ended up buying the product.

What does that tell us about the Performance of the campaign?

Nothing!

Understanding campaign performance

- A marketing manager has send a mailing to 40 000 customers.
- 16 675 customers ended up buying the product.
- Last month the marketing manager did not run any campaign and 13 208 customers purchased the product.

What does that tell us about the Performance of the campaign?

Nothing!

Understanding campaign performance

- A marketing manager has send a mailing to 40 000 customers.
- 16 675 customers ended up buying the product.
- Last month the marketing manager did not run any campaign and 13 208 customers purchased the product.
- The marketing manager had budgeted / forecast 14 000 sales this month without running the campaign

What does that tell us about the Performance of the campaign?

Nothing!

Understanding campaign performance

- A marketing manager has send a mailing to 40 000 customers.
- 16 675 customers ended up buying the product.
- Last month the marketing manager did not run any campaign and 13 208 customers purchased the product.
- The marketing manager had budgeted / forecast 14 000 sales this month without running the campaign.
- Of 40 000 other customers who did not receive the mailing 12 789 purchased this month.

Nothing!

What does that tell us about the Performance of the campaign?

Test and control groups

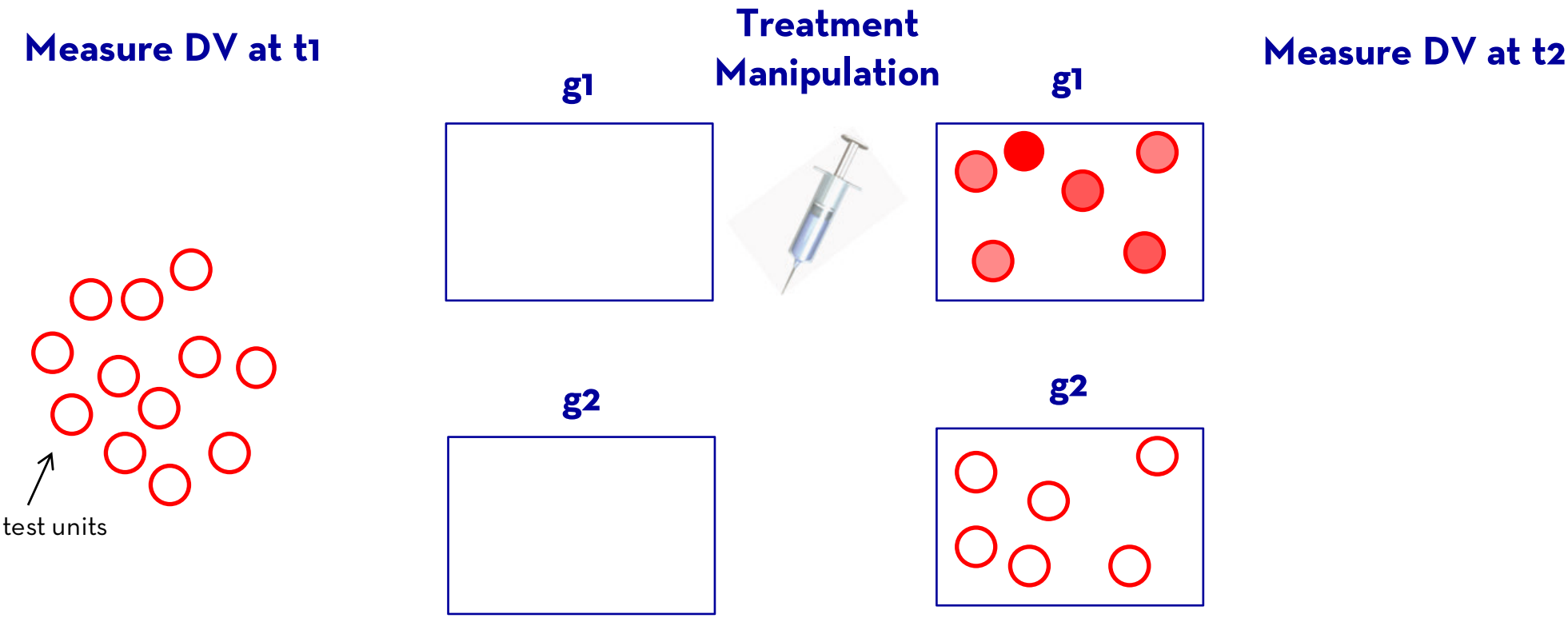
- The ***only*** way of understanding the effectiveness of a campaign is through control groups.
- Select a group of people eligible for the campaign.
- From this group select a random sample as the control group.
- Execute the campaign on the remainder (the test or treatment group)
- Compare the performance of the two groups.(over the same time periods etc.).

Important steps to a smart business experiment:
1. Set treatment and control groups

You need at least (n+1) groups.

n = no. of variations of treatment(s)

+1 = control group

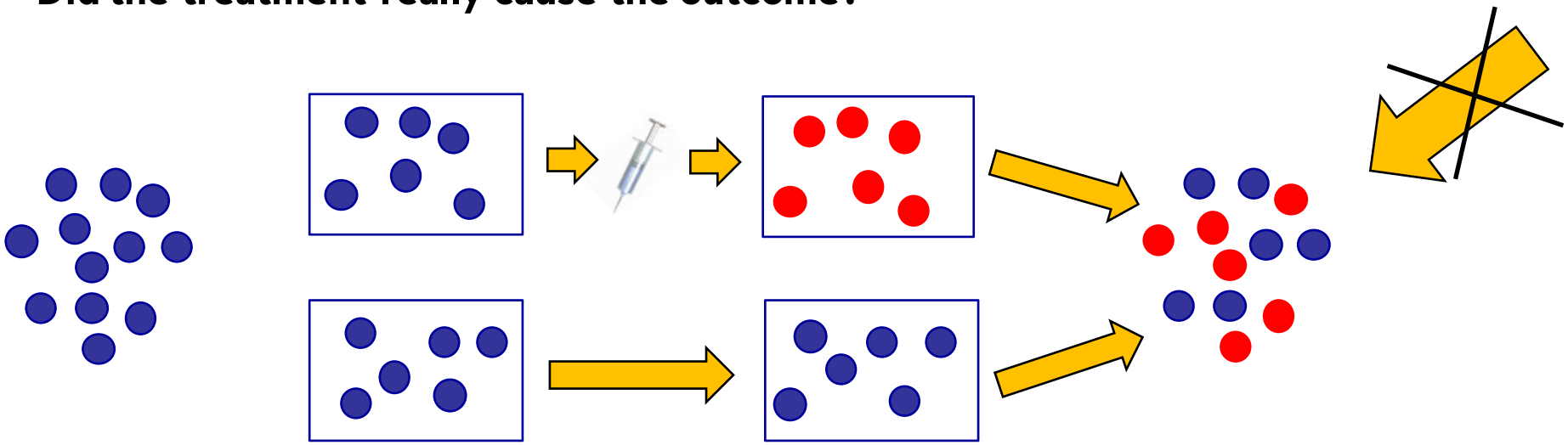


DV = dependent variable, g1 = group 1 (treatment group), g2 = group 2 (control group).

Important steps to a smart business experiment: 2. Provide internal and external validity

- **Internal validity** is a measure of accuracy of an experiment. It measures whether the manipulation of the treatment has actually caused the effects on the dependent variable or if they could have been influenced by extraneous variables.

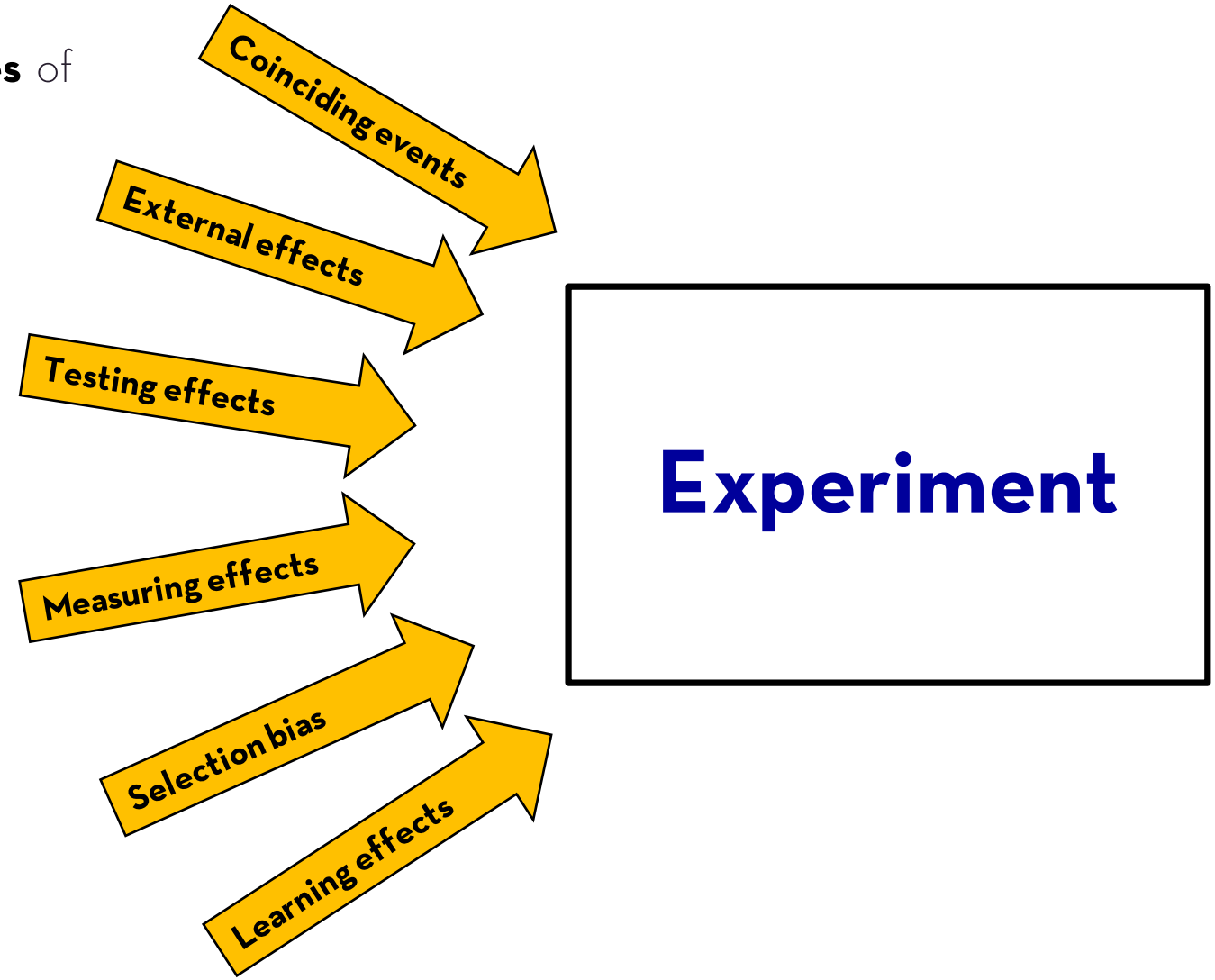
- **Did the treatment really cause the outcome?**



Malhotra 1993, p. 225; own presentation

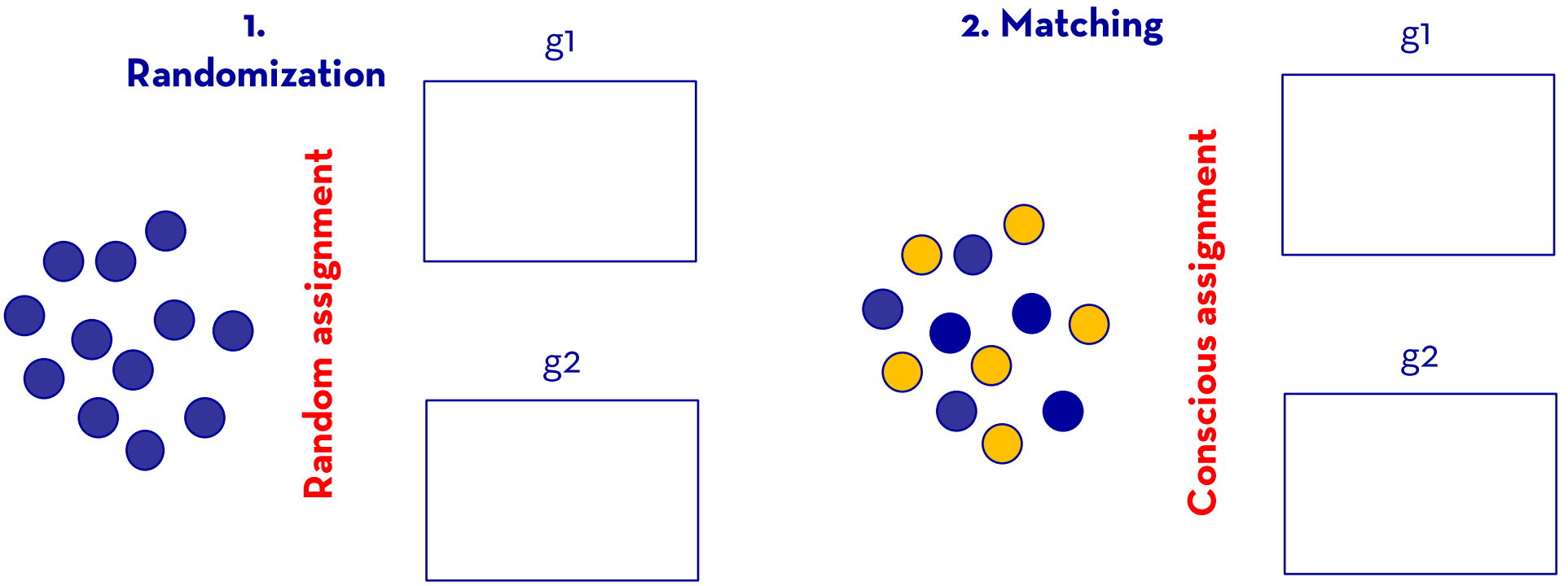
Important steps to a smart business experiment:
3. Control extraneous variables

- There are **various types** of extraneous variables.
- They can lie in the environment of an experiment as well as in the testing itself, the test units, or their selection.



Important steps to a smart business experiment:
3. Control extraneous variables

Many extraneous variables lie **inside the test units** (e.g. men and women often react differently to a treatment). Since you can't possibly eliminate (ore even be aware of) all of them, you have to make sure that they are **distributed equally** to treatment and control group and hence do not bias the outcome. Two possible ways to do so:



All functions to analyze the data collected from an experiment are available in R base

- Depending on the measurement level of the dependent variable use either
  - the `glm` function (for binary dependent variables)
  - or `lm`, `t.est`, or `anova` (for metrics dependent variables)
- For more complex experimental designs R can also help to e.g. setup the design of the experiment or plot interactions.

Uplift modelling or why your A/B test might not show any impact of a marketing campaign

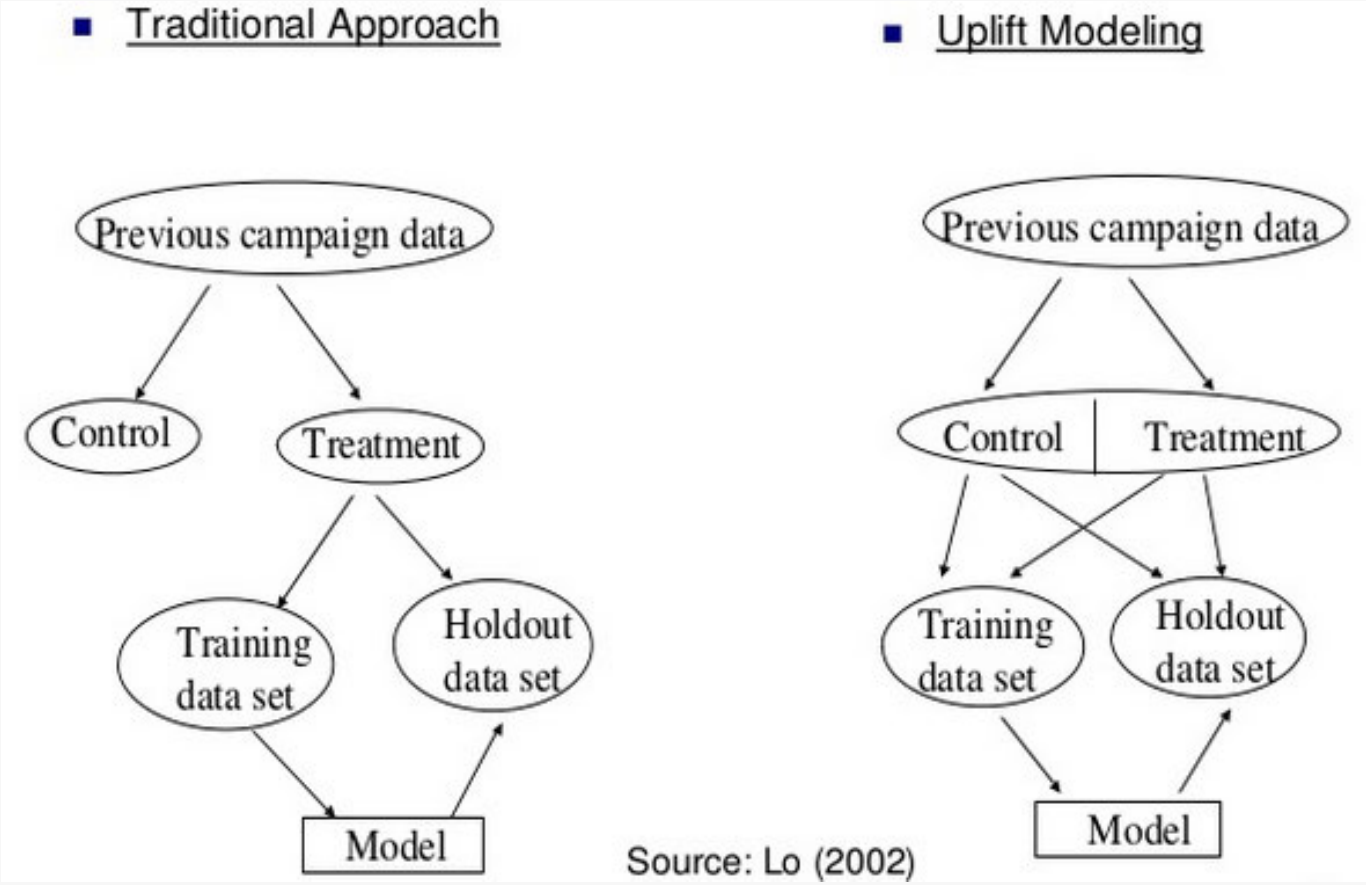
- Your model will identify the persons who are likely to churn correctly.
- However, your A/B testing shows that a campaign targeting the people with the high probablity to churn has no impact. The reason being that the marketing treatment might have no impact on this customer group.

→ Your models are correct, you are just targeting the wrong customers!

→ Possible solution:

Uplift modelling (incremental modelling/true lift modelling/net modelling) is a predictive modelling technique that directly models the incremental impact of a treatment (such as a direct marketing action) on an individual's behavior.

Uplift modelling or why your A/B test might not show any impact of a marketing campaign



Source: Lo (2002)

http://www.sigkdd.org/sites/default/files/issues/4-2-2002-12/lo.pdf

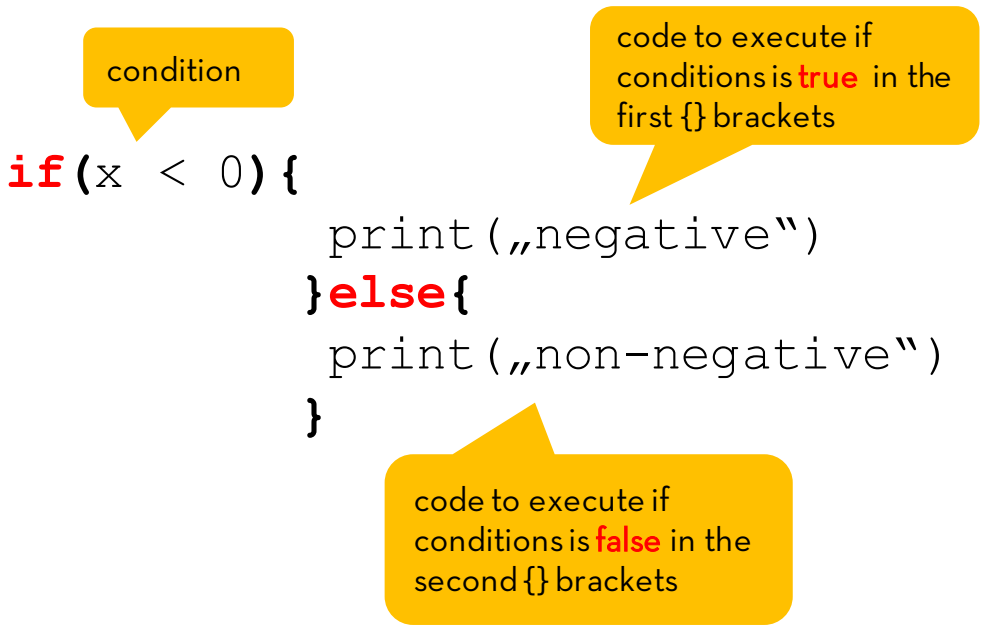
(e) EXCURSUS: Control structures

We will discuss three major points for enhanced data analysis workflow

- IF … ELSE statements for conditioning
- Loops
- Writing functions to simplify repetitive work

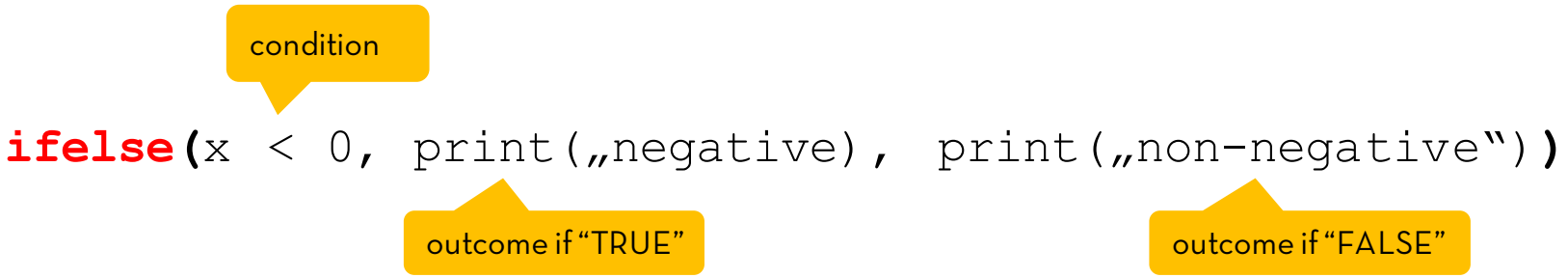
Conditioning with IF ... ELSE statements

Automate repetitive procedure



Conditioning with IF … ELSE statements

Shortcut for simple conditions



FOR Loop

- Loop through code
- Use **break** to terminate the loop
- Use **next** to skip the current cycle

Execute the loop 10 times

code to execute

```
for(i in 1:10){
  print(i)
}
```

Loops are slow! Usually you can avoid loops by vectorization and/or data.table

WHILE Loop

Execute commands until a condition is no longer satisfied

```
while(x < 5){
        x <- x+1
        print(x)
       }
```

code to execute

Define your own functions

- Automate repetitive procedure
- Create functions to work inside data.table objects

name of your function

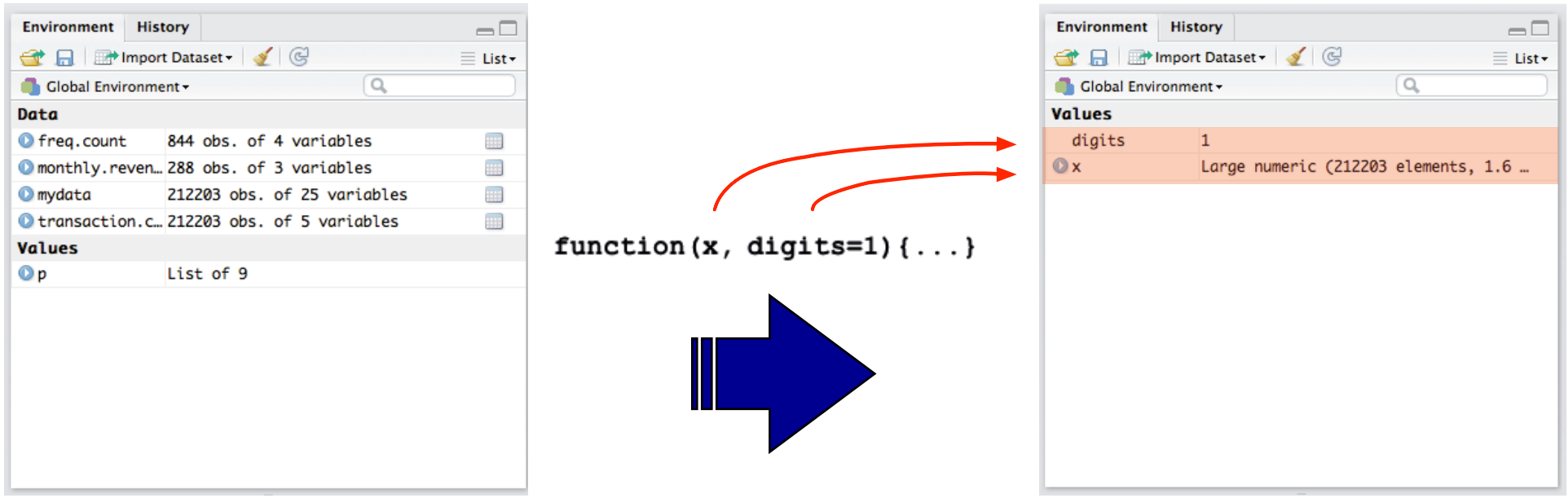
function arguments: use "=" for default values

function body: put your code here

```
YourOwnFunction <- function(x, digits=1){
...
return(ValueToReturn)
}
```

value to be returned by the function. Important: you can only return one value

Make sure all required variables are available in the function environment

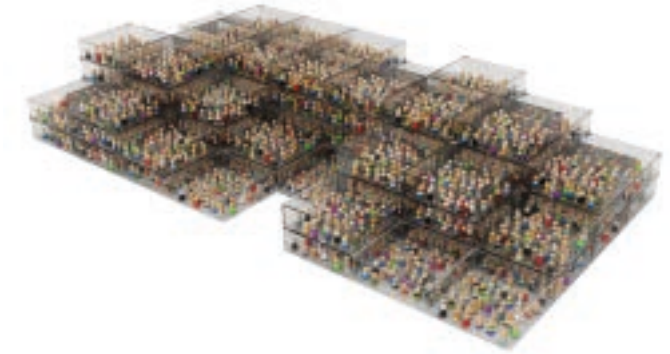


Only the passed arguments are available in the function environment

(e) Machine Learning (Unsupervised)

Customer Segmentation

- Finding...
- Assessing... **CUSTOMER SEGMENTS**
- Predicting...



The steps of clustering

1. Transform the data for the particular clustering method:
   - `kmeans()` and mcclust() need all numeric data,
   - `poLCA()` requires categorical data.
2. Choose the distance measure (similarity/dissimilarity measure) and compute the distance matrix.
3. Apply the clustering algorithm.
4. Examine the solution with regard of the underlying data, i.e.:

   How does it answer your business question?

Questions to check your proposed solution

1. Do the groups differ?
2. Does the differentiation point to some underlying story to tell?
3. Do we see immediately odd results (such as a mean equal to the value of one data level)?

Hierarchical clustering (HC)

- HC is a distance based-algorithm
- **Dissimilarity matrix** reports a metric for the distance between each pair of observations
  - Starts with an own cluster for every operation
  - Adds one by one the observation with the lowest distance
  - Usually the Euclidean distance is used to determine the distance between objects

$$d = \sqrt{\sum (X - Y)^2}$$

use dist() to calculate in R

We need to create the dissimilarity matrix

- We use the **`cluster`** package
- Since the data is not all numeric an alternative method ("Gowers coefficient") is used to calculate the distance between the object.

part of the "cluster" package

```
daisy(clusterdata)
```

daisy() selects automatically the appropriate distance metric. Use **`metric=`**... to override manually

Invoke the clustering algorithm

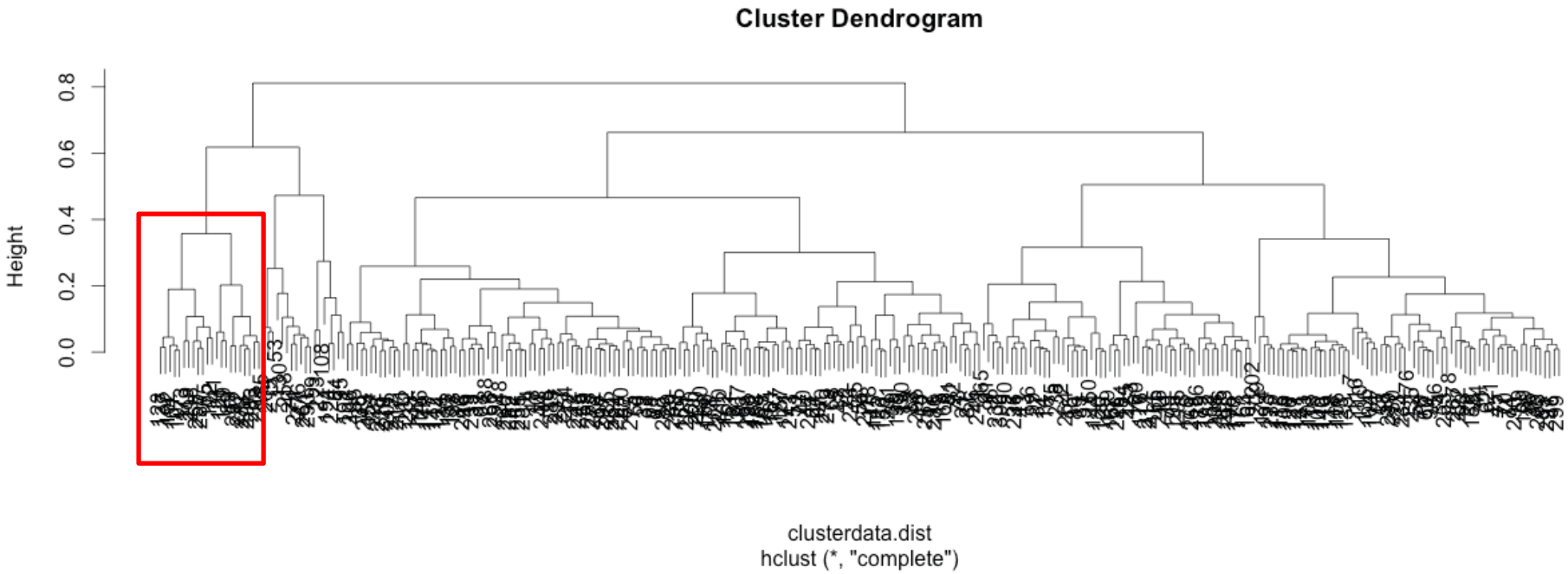
We use the **"complete"** linkage method, which evaluates the distance between every member when combining observations and groups

part of the "cluster" package

```
hclust(clusterdata.dist)
```

Dissimilarity matrix

Plot the dendrogram

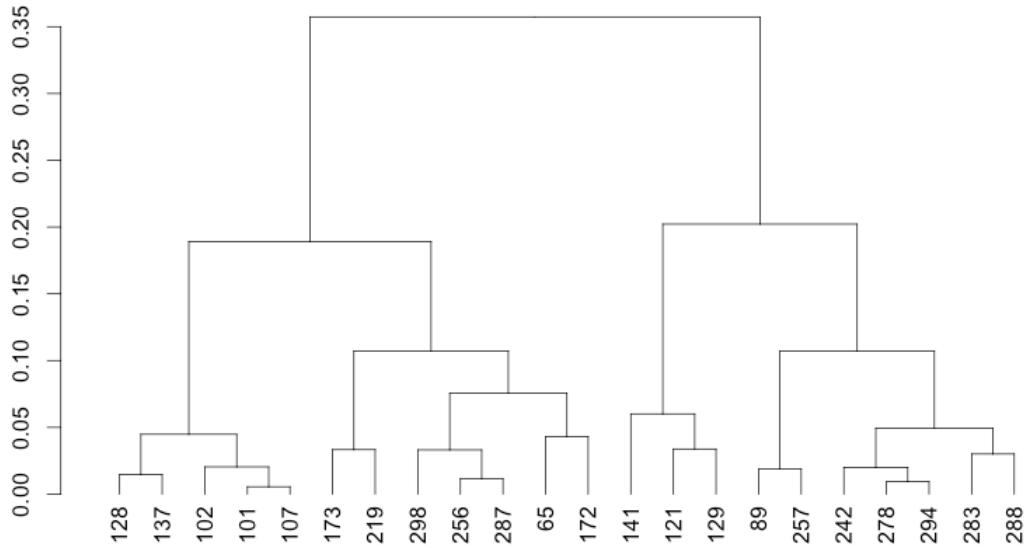


Can only plot clusterdata if package "cluster" is installed

```
plot(clusterdata.hc)
```

result of hclust()

Zoom into the plot



customers

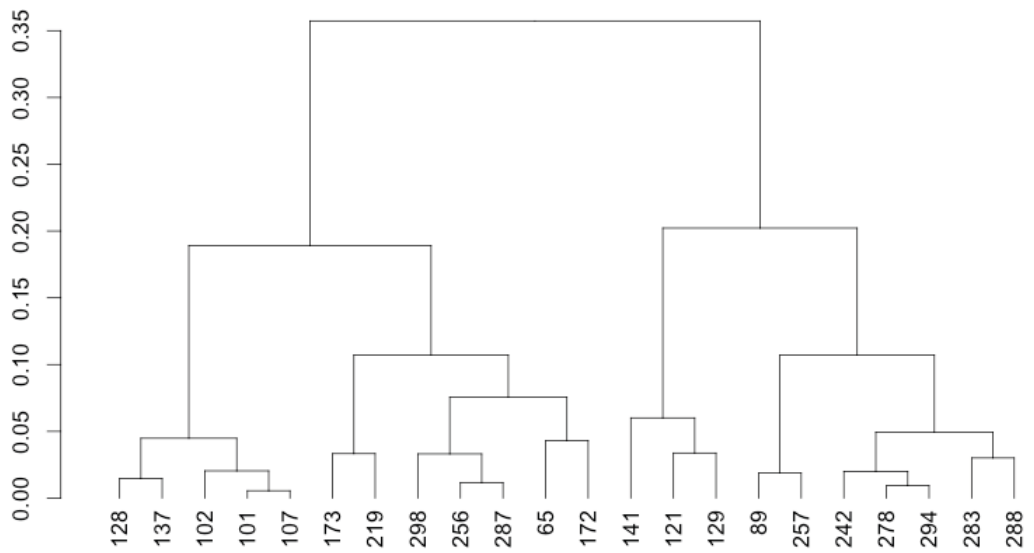
cut the dendrogram to height 0.5

Select a specific branch of the dendrogram

```
plot(cut(as.dendrogram(clusterdata.hc), h=0.5)$lower[[1]])
```

result of hclust()

Zoom into the plot



[row]	age	gender	income	kids	ownHome	subscribe
101	24.73796	Male	18457.85	1	ownNo	subYes
107	23.19013	Male	17510.28	1	ownNo	subYes
...	...	...	...	...	...	...
141	25.17703	Female	20125.80	2	ownNo	subYes
173	64.70641	Male	45517.15	0	ownNo	subYes
	...	...	...	...	...	

similar

less similar

Check goodness of fit

We use the **cophenetic correlation coefficient** (CPPC)

- Compare the dendrogram with the dissimilarity matrix
- Interpretation similar to Pearsons r

correlation

result of hclust()

result of daisy()

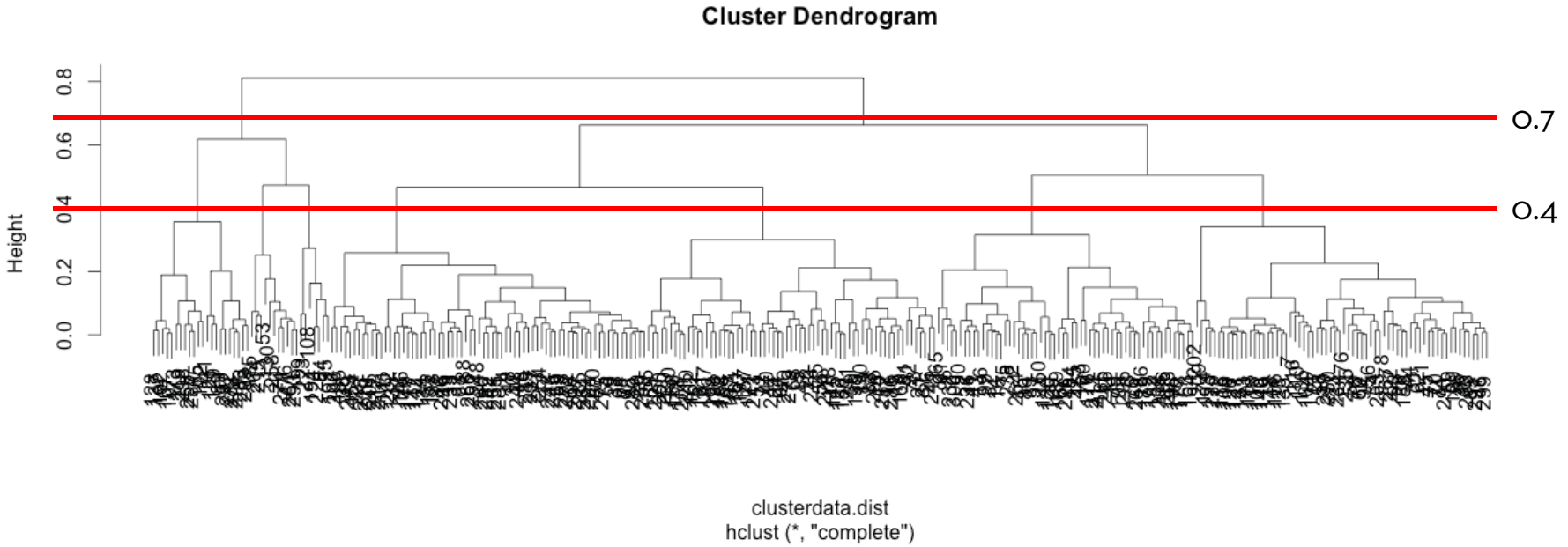
```
cor(cophenetic(clusterdata.hc), clusterdata.dist)
```

cophenetic distance

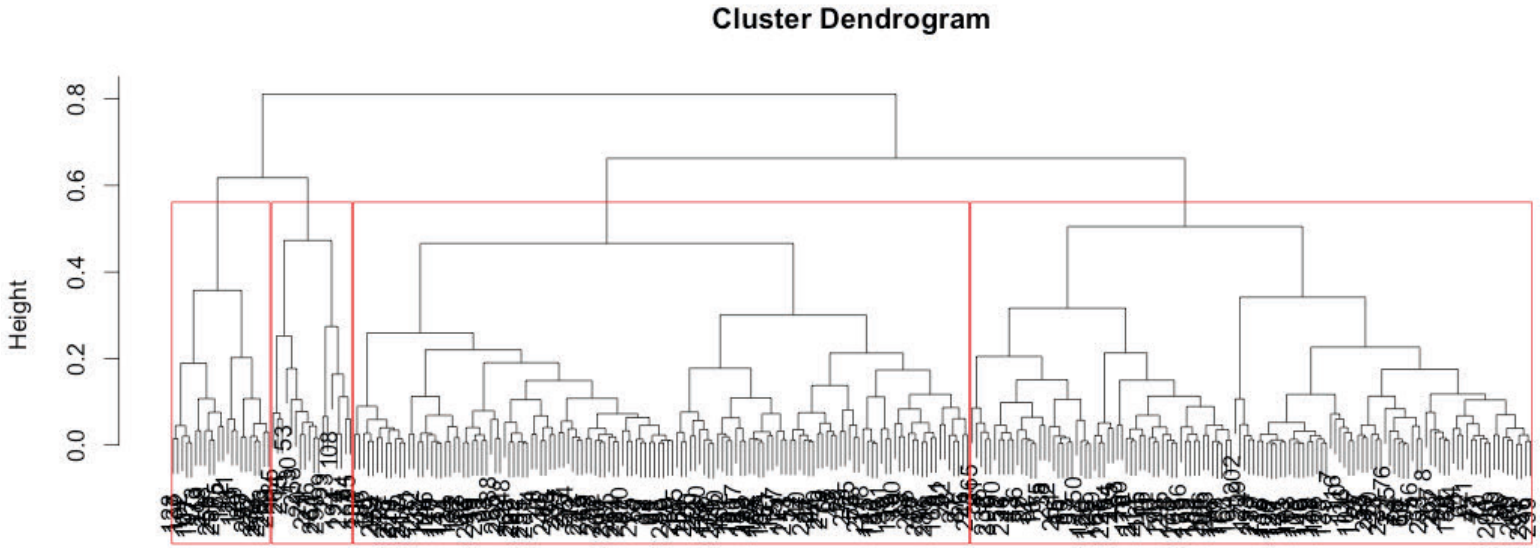
Extract groups

How do we get specific segments assignments?

- cut the dedrogram at any height desired
- e.g. cut at 0.7: 2 groups, cut at 0.4: 7 groups



Zoom into the plot



Plot the dendrogram

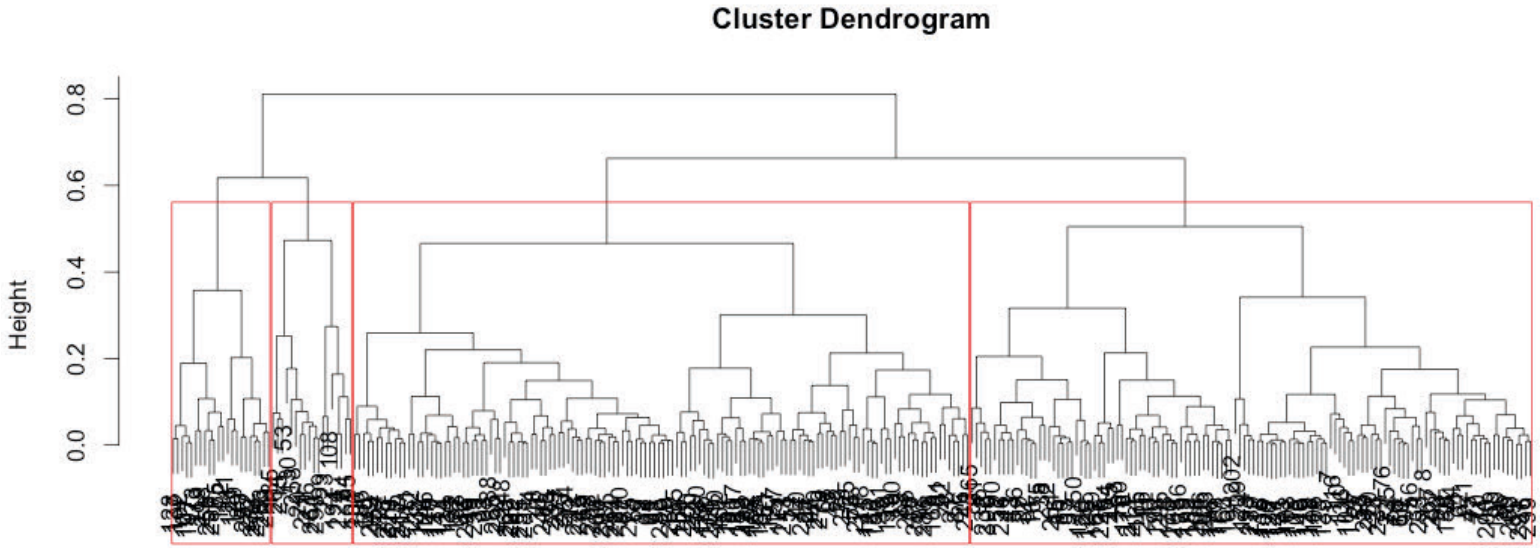
```
plot(clusterdata.hc)
rect.hclust(clusterdata.hc,  k=4, border="red")
```

red border

plot overlay
part of the "cluster"
package

4 groups

Obtain the group assignments

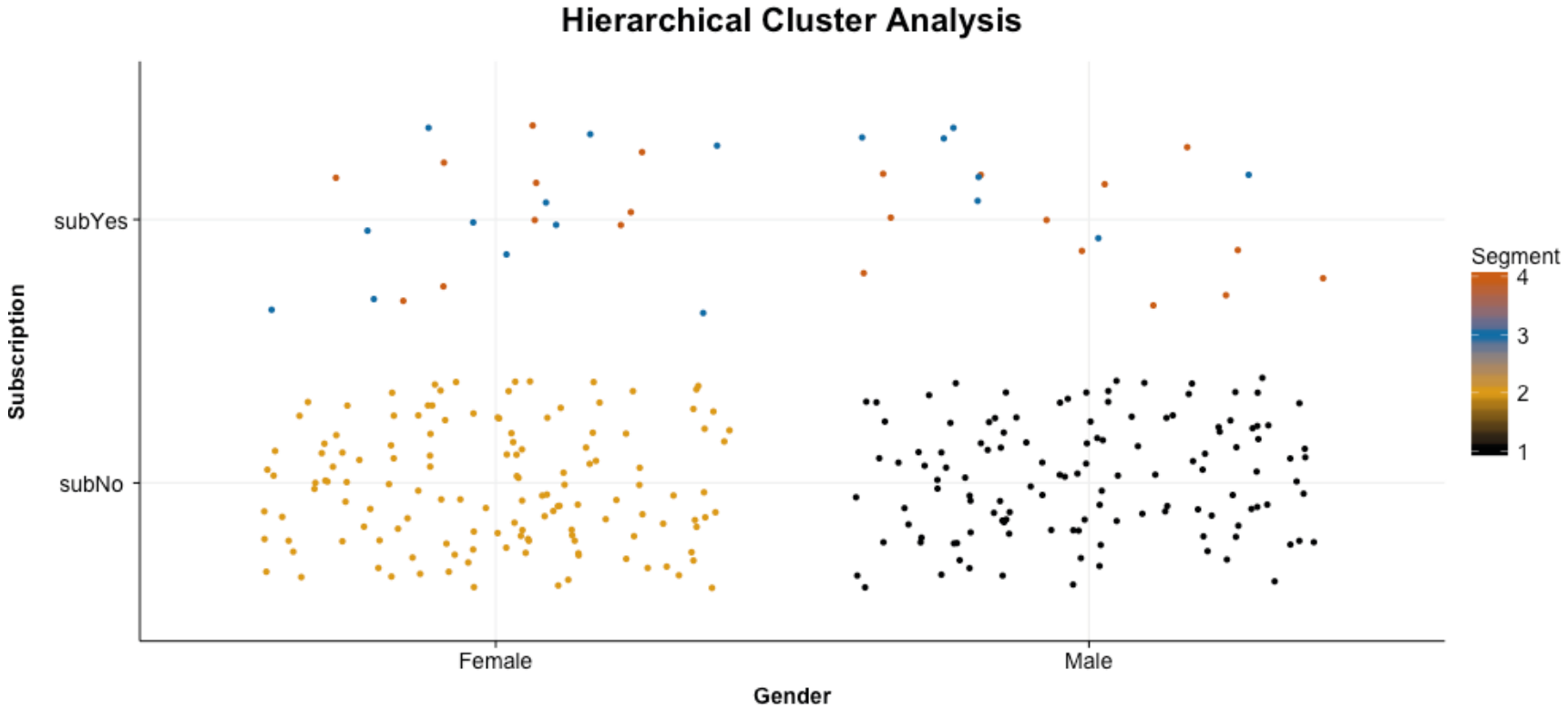



```
cutree(clusterdata.hc,  k=4)
```

part of the "cluster" package

4 groups

Hierarchical cluster results



k-means

- Mean based clustering (in term of mean sum of squares)

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2$$

observation

mean of points in cluster

- **Mean deviation** is calculated based on the Euclidean distance
  - For numeric data or data that can be reasonably coerced to numeric
  - Not optimal for binary values

Recode dataset to numeric

k-means only accepts **numeric data**. Therefore we need to recode our binary values.

condition

if “true” = 0

```
ifelse(clusterdata$gender ==”Male”, 0, 1)
```

if “false” = 1

Run the k-means algorithm

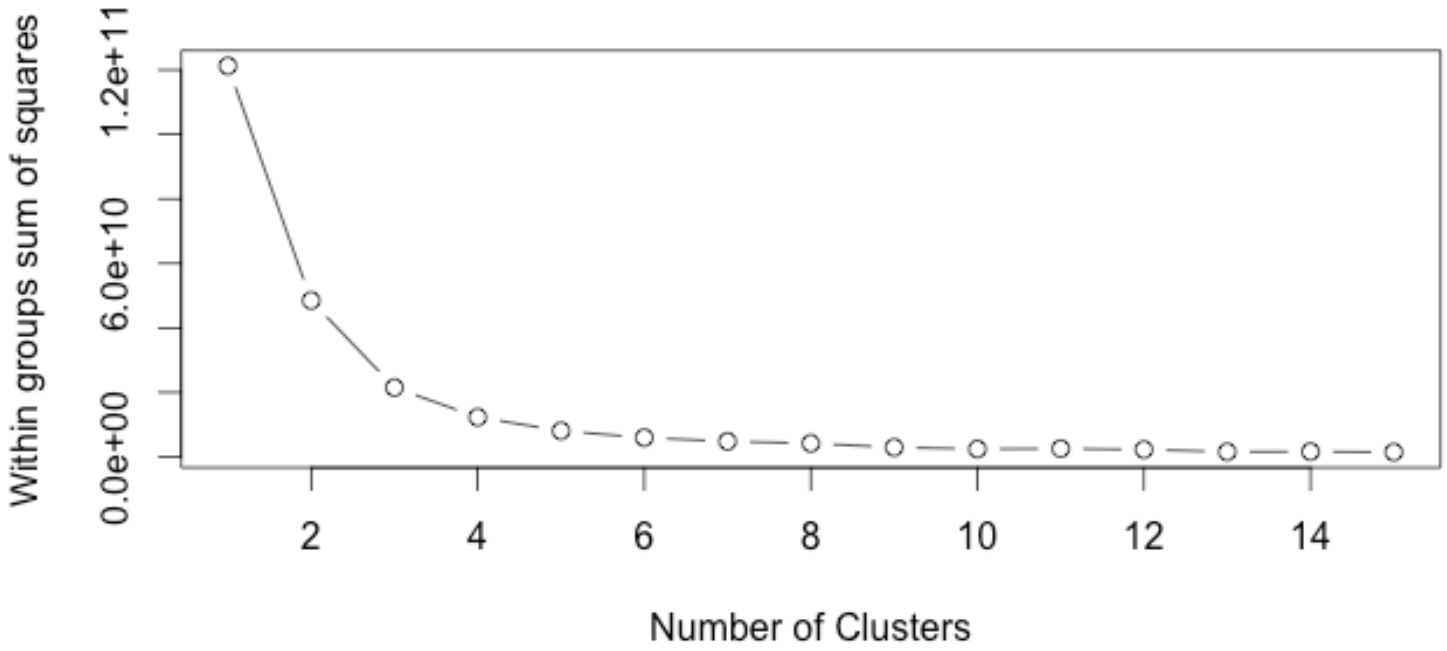
We generate again **4 clusters.**

make sure we get reproducible results

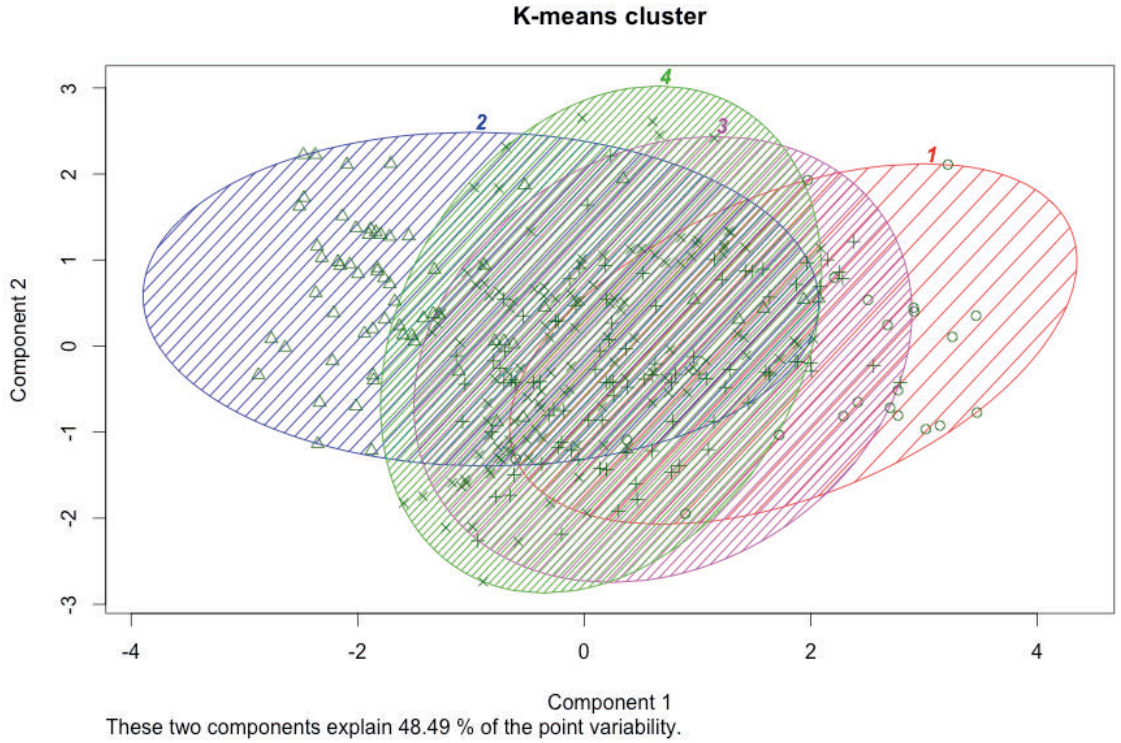
```
set.seed(96743)
kmeans(clusterdata.num,  centers  =  4)
```

We want 4 clusters

The elbow criteria



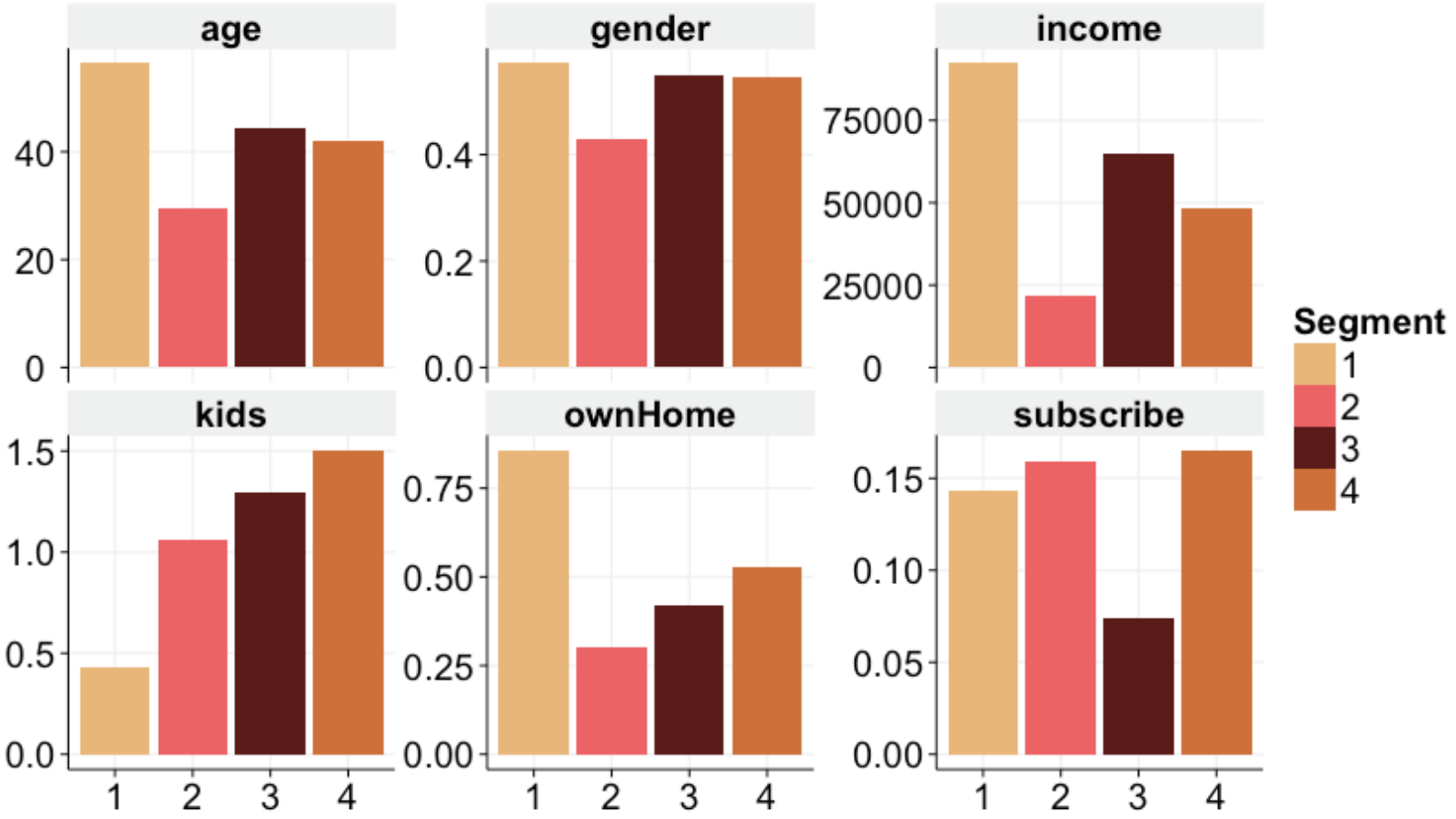
k-mean results




```
clusplot(clusterdata, clusterdata.segments$cluster,
         color=TRUE, shade=TRUE, labels=4,
         lines=0, main="K-means cluster")
```

part of the cluster package

Plot cluster results



R Workshop

A hands-on introduction

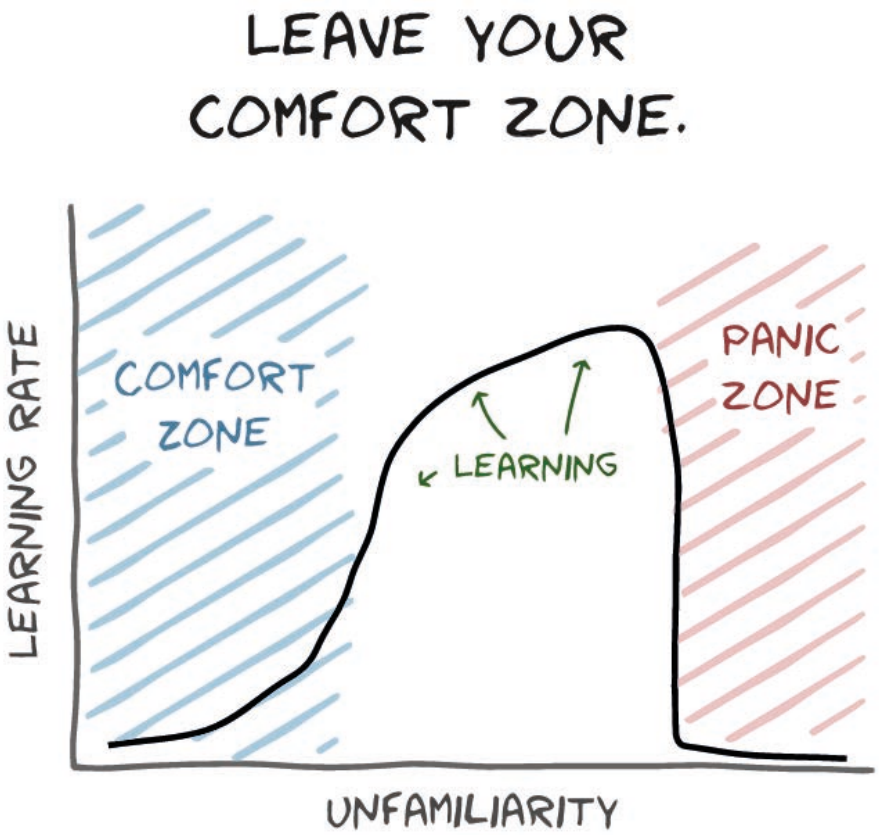
University of Zurich[UZH]

Dr. Markus Meierer | markus.meierer@business.uzh.ch --- Patrick Bachmann | patrick.bachmann@business.uzh.ch

We have promised...

...that AFTER this course you will have a fundamental knowledge on:

1. why to use R
2. setting up the R environment
3. basic data management techniques
4. analyzing data with descriptive modeling techniques
5. analyzing data with predictive modeling techniques



How to proceed?

- Strengthen your R knowledge by actually using R to solve your business problems
- Further topics
  - Digging into the details of each modeling approach – Advanced modeling techniques
  - How to ensure a proper code base that everyone understands and everyone can use (even if the person in charge left the company) – Writing R packages
  - Saving time by automating repetitive analysis tasks – Using Rmarkdown to generate dynamic PDF reports & creating interactive, web applications
  - How to find errors in your code quickly? – Debugging R code
  - Speeding up R code – Using Compiled code and parallel computation in R

Questions? Get in touch!

Dr. Markus Meierer

Phone +41 44 634 2902

Fax: +41 44 634 2940

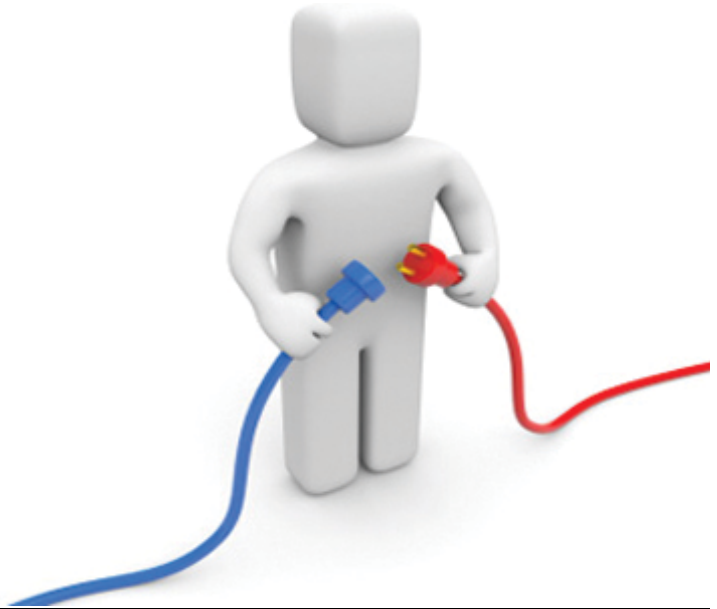
Email: markus.meierer@uzh.ch

linkedin.com/in/markusmeierer

xing.com/profile/markus_meierer

twitter.com/mmeierer

pinterest.com/markusmeierer



Questions? Get in touch!

Patrick Bachmann

Phone +41 44 634 9206

Fax: +41 44 634 2940

Email: patrick.bachmann@uzh.ch

xing.com/profile/patrick_bachmann8

twitter.com/pabachmann