

***Welcome to***

Advanced R

## Instructor Bio

Dejan Sarka ([dsarka@solidq.com](mailto:dsarka@solidq.com),  
[dsarka@siol.net](mailto:dsarka@siol.net), @DejanSarka)

- 30 years of experience
- SQL Server MVP, MCT,...
- 17 books
- ~20 courses and seminars
- Focus:
  - Data modeling
  - Data mining
  - Data quality

## Data Science

- Data science as the advanced data analysis technique is gaining popularity
  - With modern engines, services, products and packages, like SQL Server Analysis Services (SSAS), Azure ML, R, and Python, data science has become a black box
  - But: not knowing how the algorithms work might lead to many problems, including using the wrong algorithm for a task, misinterpretation of the results, and more
  - A correct data preparation is the key to success
- Learn how how to do the data science in R, including:
  - Do the initial overview and preparation of the data
  - Analyze the data
  - Understand the basicc of the mathematics behind

## Agenda (1)

- A brief introduction
  - Data science projects
  - R data structures
- Data preparation and overview
  - Introductory statistics
  - Basic graphs
- Associations and advanced visualizations
  - Associations between two variables
  - Bayesian inference
  - Linear models

## Agenda (2)

- Matrix operations and feature selection
  - Feature selection in linear models
  - Basic matrix algebra
  - Principal component analysis
  - Exploratory factor analysis
- Unsupervised learning
  - Hierarchical clustering
  - K-means clustering
  - Association rules

6

## Agenda (3)

- Supervised learning
  - Neural Network
  - Logistic Regression
  - Trees and forests
  - K-nearest neighbors
- Modern topics
  - Support vector machines
  - Time series
  - Text mining
  - Deep learning
  - Reinforcement learning

7

## Timing

- 08.30AM - 10.00AM 1<sup>st</sup> slot
- 10.00AM - 10.15AM break
- 10.15AM - 12.00PM 2<sup>nd</sup> slot
- 12.00PM - 01.00PM lunch break
- 01.00PM - 02.30PM 3<sup>rd</sup> slot
- 02.30PM - 02:45PM break
- 02.45PM - 04.30PM 4<sup>th</sup> slot

8

## Slides and Code Download

- Short URL: <https://goo.gl/K9YovT>
- Mail: [dsarka@solidq.com](mailto:dsarka@solidq.com)

9

## Book Promo

For the attendees only – discount for the [Data Science with SQL Server Quick Start Guide](#)

- Promo codes:
  - SWSSSGP15 – 15% p-book
  - DSWSQEB50 – 50% e-book
- Valid from Oct 25<sup>th</sup> to Nov 30<sup>th</sup>
- Create a login on the Packt site [www.packtpub.com](http://www.packtpub.com) and add the book to the cart
  - Use the promo codes above



## Module 01: A brief introduction

## Agenda

- Introduction to data science
- R data structures

12

## Data Science

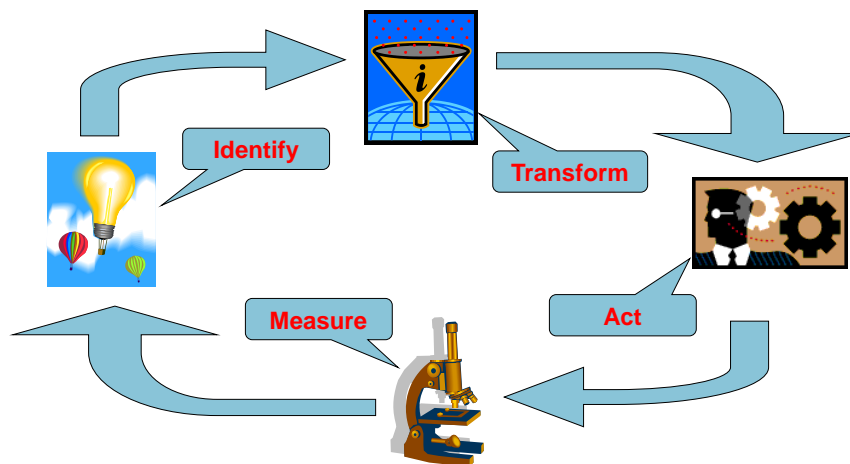
- In short: extracting knowledge from data
  - Statistics and programming
- Many branches (or synonyms)?
  - Statistics
  - Data mining
  - Machine learning
  - Predictive analytics
  - Deep learning
  - Artificial intelligence

13

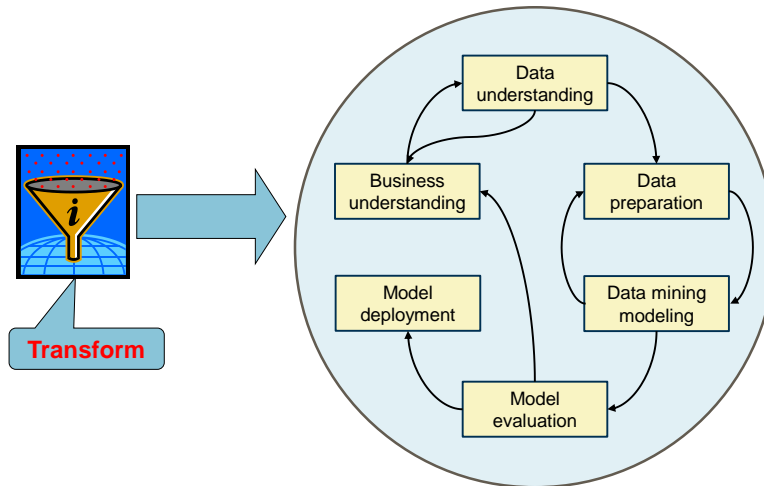
## Statistics vs DM vs ML

- Many DM and ML algorithms are derived from statistics
  - Statistics is a science; DM and ML is business
- Sample sizes (over the thumb):
  - Statistics: small ( $< 30$ ) and medium ( $> 200$ ,  $< 100,000$ ) – historically called large
  - DM: medium ( $> 200$ ,  $< 100,000$ )
  - ML: medium ( $> 200$ ,  $< 100,000$ ) and large, i.e. census ( $> 100,000$ )
- Users:
  - Statistics: data scientists, mathematicians
  - DM: advanced analysts, managers
  - ML: advanced analysts, machines

## DM / ML Virtuous Cycle



## The CRISP Model



CRISP = Cross Industry Standard Process for Data Mining

## Cases and Variables

- Data science algorithms analyze **cases**
  - The case is the entity being categorized and classified, e.g. customer, product, etc.
- Each case encapsulates the available information in the data about the entity in **variables** or **features**
- Cases form a **dataset**
  - The data model is a **matrix**, or a **table** in a RDBMS



## About R

- The R statistical programming language is a free open source package based on the S language developed by Bell Labs
- R written as a research project by Ross Ihaka and Robert Gentleman
  - Now developed by a group of statisticians called 'the R core team', with a home page at [www.r-project.org](http://www.r-project.org)
- R is available free of charge and is distributed under the terms of the [Free Software Foundation's GNU General Public License](#)
  - Available for Windows, Mac OS X, and Linux

## Microsoft R Options

- [Microsoft R Open \(MRO\)](#) - the enhanced free distribution of R from Microsoft
- [Microsoft R Client \(MRC\)](#) – built on MRO, free, with RevoScaleR technology
- [Microsoft ML Server \(MRS\)](#) – commercial, enterprise class server for parallel and distributed workloads of R and Python processes on servers
- [SQL Server ML Services \(In-Database\)](#) – integration of ML Server with the Database Engine

## R Expressions and Variables

- Basic expressions
  - > 1 + 1
  - > 2 + 3 \* 4
  - > 3 ^ 3
  - > sqrt(81)
  - > pi
- Variables
  - Numeric, Boolean, Strings
  - Type determined automatically
  - Created with “<-” operator
  - Name is case sensitive and can include a period

## R Vectors

- Vectors are ordered collections of numbers
- Created with
  - > c() to concatenate elements or sub-vectors
  - > rep() to repeat elements or patterns
  - > seq() or m:n to generate sequences
- Most mathematical functions and operators can be applied to vectors without loops
- Use the [] operator to select elements
  - Select or exclude specific elements

## R Packages

- Packages are optional modules you can download and install
  - Check the installed packages with the `installed.packages()` command
  - Packages are stored in the folder called **library**. You can get the path to the library with the `.libPaths()`
  - The `library()` function lists the packages in your library
  - The `install.packages("packagename")` command searches the CRAN sites for the package, downloads it, unzips it, and installs it

## Collections and Objects

- *Matrices* or more generally *arrays* are multi-dimensional generalizations of vectors
- *Factors* provide compact ways to handle categorical data – distinct values are *levels*
- *Lists* are a general form of vector in which the various elements or *components* need not be of the same type
- *Data frames* are matrix-like structures, in which the columns can be of different types
- *Functions* can be stored in the project's workspace - a simple way to extend R

## Factors

- Variables can store **discrete** or **continuous** values
  - Discrete values can be **nominal**, or **categorical**, where they represent labels only, or **ordinal**, where there is an intrinsic order in the values
- In R, **factors** represent nominal and ordinal variables
  - **Levels** of a factor represent distinct values
  - Levels can be **ordered**

## Lists

- Lists are ordered collections of different data structures
  - Create lists with the **list()** function
  - Refer to objects of a list by position, using the index number enclosed in double parentheses
  - If an element is a vector or a matrix, you can additionally use the position of a value in a vector enclosed in single parentheses
- Lot of advanced analytics algorithms return lists
  - Use **\$** to get to part of list
  - The **unlist()** to separates list to smaller objects

## Data Frames

- The most important data structure in R is a **data frame**
  - Data frames are matrices where each variable can be of a different type
  - Create a data frame with the `data.frame()` function from multiple vectors of the same length
- Read a data frame from a CSV file, SQL Server table, etc.
  - `TM = read.table("C:\\AdvancedR\\TM.csv", sep=";", header=TRUE, row.names = "CustomerKey", stringsAsFactors = TRUE)`

## Accessing Data in a Data Frame

- Use position and name indexes
  - `TM[1:3,c("MaritalStatus", "Gender")]`
- Use the data frame name and column name, separated by the dollar (\$)
  - `TM$Gender`
  - With the `attach()` function, you add the data frame to the search path that R uses to find the objects
  - The `with()` function allows you to name the data frame only once, and then use the variables in a set of statements enclosed in `{ }` brackets inside the body of the function

## Data Table

- The ***data.table*** structure inherits from data frame and introduces more relational-like form
  - The [data.table](#) package
- Subset rows
- Select and compute on columns
- Perform aggregations by group
- And more...

## Resources

- [CRISP](#) - Cross Industry Standard Process for Data Mining
- [RStudio](#) IDE
- R [data.frame](#) documentation
- [Intro to the data.table Package](#) at R bloggers
- [Business Modeling and Data Mining](#) by Dorian Pyle, Morgan Kaufmann, 2003
- [SQL Server 2017 Developer's Guide](#) by Dejan Sarka, Miloš Radivojević, and William Durkin, Packt, 2018

## Module 02:

# Data preparation and overview

## Agenda

- Descriptive statistics
- Graphs
- Data manipulation
- Sampling

## Popular Libraries

- Most popular R packages for data overview and manipulation include:
  - dplyr, plyr, data.table, stringr, zoo for data manipulation
  - moments, descry for descriptive statistics
  - vcd, corrgram for intermediate statistics
  - ggplot2, lattice for graphics

## Frequencies and Dummies

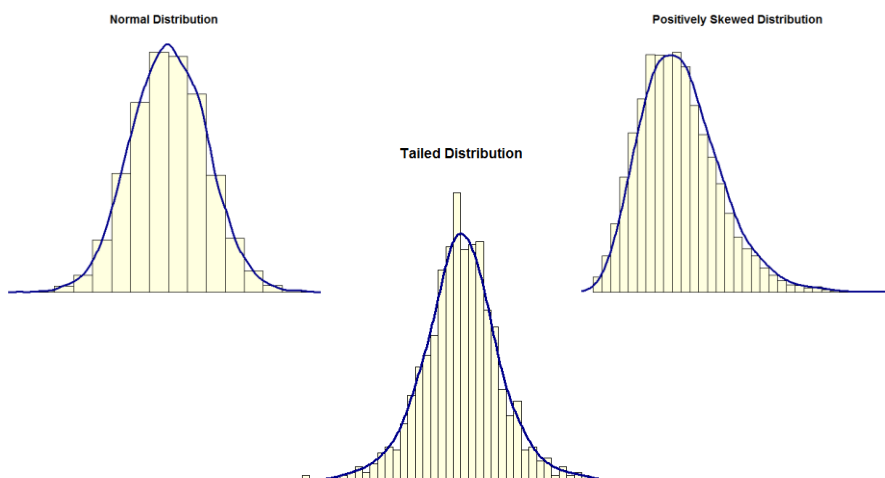
- Frequency tables and graphs are the basic representation of discrete variables
  - Value, absolute frequency, absolute percentage, cumulative frequency, cumulative percent, and histogram
- Order correctly ordinals
- When numerical values are needed, change nominals to indicators
  - Also called dummies
  - Values 0 and 1



## Population Moments

- Centers of a distribution
  - The *median* is the value at the half of the cases
  - The arithmetic *mean* (i.e., the average) is the most common measure for the center of the distribution
- Spread
  - Interquartile range (IQR) = upper quartile – lower quartile
  - Standard deviation
- Higher population moments
  - Skewness
  - Kurtosis

## Distributions



## Graphs

- Always check also the graphs
  - Histograms
  - Bar charts
  - Scatterplots
- The famous Anscombe data set
  - Modern version

36

## Missing Values

- Empty, nonexistent, uncollected data
- Before handling missing values, determine if there is any pattern in the missing data!
- Handling:
  - Do nothing
  - Filter the rows containing the missing data
  - Ignore the column
  - Predict new values
  - Build separate models
  - Modify the operational systems so that you can collect the missing values
  - Replace missing data with common (mean) values

## Outliers

- Rare and far out-of-bound values
- Before handling outliers, determine if there is any pattern in the outliers!
- Approaches:
  - Check if an outlier is an erroneous value
  - Do nothing
  - Filter the rows with the outliers
  - Ignore the column
  - Replace outliers with common (mean) values
  - Bin values into equal height ranges
  - Normalize the data values

## Smoothing

- Moving averages

- Simple moving averages (SMA)

$$S_{SMAi} = \left( \sum_{i=1}^{i+1} v_i \right) / 3$$

- Weighted moving averages (WMA)

$$S_{WMAi} = \left( \sum_{i=2}^i v_i * w_i \right) / 3$$

$$\sum w_i = 1$$

- Exponential moving averages (EMA)

$$S_{EMAi} = (v_i * \alpha) + (S_{EMAi-1} * \beta)$$

$$\alpha + \beta = 1$$

## Data Values Normalization

- Normalize input variables in the same scale (range)

- Range
- Z-score
- Logistic (sigmoid) function
- Hyperbolic tangent function

$$V = \frac{(x - x_{\min})}{(x_{\max} - x_{\min})}$$

$$V = \frac{(x - \mu)}{\sigma}$$

$$V = \frac{1}{(1 + e^{-x})}$$

- Non-linear normalizations useful for smoothing outliers

$$V = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$$

## Derived Variables

- Simple facts are not often useful enough
  - For example, how do you measure churn?
- Most of the time you have hunches
  - Some attributes can be used to extract features from them
  - Sometimes you want to smooth or filter the peak values
  - Summarizations can help with huge numbers of categories or with entities that quickly become obsolescent
  - Use creativity, experience, domain knowledge, etc.

## Combinations of Columns

- Calculated columns are simple derived variables:
  - Height<sup>2</sup> / weight (obesity index)
  - Passengers \* miles
  - Population / area (population density)
  - Activation date – application date (time needed for activation)
  - Credit limit – balance
  - Prospect age / agent age (age ratio)

## Extracting Features

- Many of the columns contain information
  - Recognizing it requires incorporating domain knowledge
- Examples:
  - ZIP codes
  - Web addresses
  - Universal Product Codes (UPCs)
  - Dates

## Package dplyr

- Functions for data manipulation:
  - select() and rename() for projections
  - filter() for filtering
  - arrange() for ordering
  - mutate() and transmute() to add new variables
  - summarise() to aggregate
  - sample\_n() and sample\_frac() for random samples
- The pipe operator %>%

44

## Aggregations

- Might already be present in a data warehouse
- Can be used in combinations
  - Number of distinct operations in a process
  - Order deviations (order value – average order value)
- Can be used for more specific purposes
  - Product taxonomy
  - Aggregate sales data for mobile phone types so that the data does not become obsolete as quickly
- Transposing, pivoting and unpivoting data

## Discretization

- Discretization (i.e., binning or categorization) is done on a single column
  - Call length (long, medium, short)
  - Age
  - Income
- Discretization methods
  - Equal width
  - Equal height
  - Custom

## Discretization

- Entropy reaches the theoretical maximum when all states are represented equally
- More states means a higher theoretical maximum
- When you discretize variables, you are losing information
  - Discretize to buckets with an equal number of cases to maximally preserve the entropy
  - Discretize to buckets with an equal width to maximally preserve the distribution

## Entropy

- Entropy measures uncertainty in a system

$$H(x) = - \sum_{i=1}^n P(x_i) * \log_2 P(x_i)$$

- If a system has states with equal probabilities, the outcome is the most uncertain (the state of maximum entropy)
- Low entropy, compared to the theoretical maximum entropy, suggests that some system states are not well represented
- Low entropy means that the variable is not very useful for analyses

## Resources

- [Anscombe's quartet](#) in Wikipedia
  - [The Datasaurus Dozen](#) - bivariate data with a given mean, median, and correlation in *any* shape you like (even a dinosaur)
- The [dplyr](#) package documentation
- [Introductory Statistics](#) by Sheldon M. Ross, Elsevier, 2017
- [Data Preparation for Data Mining](#) by Dorian Pyle, Morgan Kaufmann, 1999



## Module 03:

# Associations and advanced visualizations

## Agenda

- Associations between two variables
- Bayesian inference
- Linear models

## Associations

### ► Discrete Variables

- *Contingency* tables do not rely on numeric values
- The *Null Hypothesis*: there is no relationship between row and column frequencies
  - So there should be no difference between observed (O) and expected (E) frequencies

Observed			
Gender	Married	Single	Total
F	4745	4388	9133
M	5266	4085	9351
Total	10011	8473	18484
Expected			
Gender	Married	Single	Total
F	4946	4187	9133
M	5065	4286	9351
Total	10011	8473	18484

## Associations

### ► Discrete Variables

- Chi-Squared formula:  $\sum_i \frac{(O - E)^2}{E}$
- For the Chi-Squared distribution there are already prepared tables with critical points for different degrees of freedom and for a specific confidence level
- Degrees of freedom = the product of the degrees of freedom for columns and rows
 
$$DF = (C - 1) * (R - 1)$$

## Chi-Squared Critical Points

DF	$\chi^2$ value										
1	0.004	0.02	0.06	0.15	0.46	1.07	1.64	2.71	3.84	6.64	10.83
2	0.10	0.21	0.45	0.71	1.39	2.41	3.22	4.60	5.99	9.21	13.82
3	0.35	0.58	1.01	1.42	2.37	3.66	4.64	6.25	7.82	11.34	16.27
4	0.71	1.06	1.65	2.20	3.36	4.88	5.99	7.78	9.49	13.28	18.47
5	1.14	1.61	2.34	3.00	4.35	6.06	7.29	9.24	11.07	15.09	20.52
6	1.63	2.20	3.07	3.83	5.35	7.23	8.56	10.64	12.59	16.81	22.46
7	2.17	2.83	3.82	4.67	6.35	8.38	9.80	12.02	14.07	18.48	24.32
8	2.73	3.49	4.59	5.53	7.34	9.52	11.03	13.36	15.51	20.09	26.12
9	3.32	4.17	5.38	6.39	8.34	10.66	12.24	14.68	16.92	21.67	27.88
10	3.94	4.86	6.18	7.27	9.34	11.78	13.44	15.99	18.31	23.21	29.59
Probability	0.95	0.90	0.80	0.70	0.50	0.30	0.20	0.10	0.05	0.01	0.001
	Not significant								Significant		

## Chi-Squared Test in R

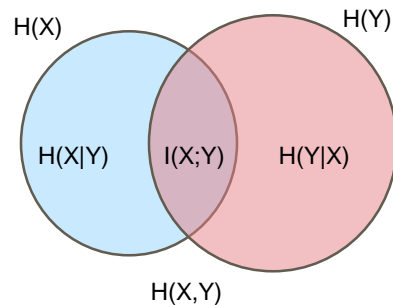
- Use the `table()` or `xtabs()` functions to cross-tabulate two variables
- Store the results in an object
- Use the `chisq.test()` function to test for independence

```
tEduGen <- xtabs(~ Education + Gender);
tNcaBik <- xtabs(~ NumberCarsOwned +
  BikeBuyer);
chisq.test(tEduGen);
chisq.test(tNcaBik);
```

## Mutual Information

- Mutual information of two variables is a measure of the mutual dependence between them
  - Quantifies the amount of information obtained about one variable, through the other variable
  - More general than correlation coefficient

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} P(x,y) \log_2 \left( \frac{P(x,y)}{P(x)P(y)} \right)$$



## Measure the Association

- You can measure the association with the following: **phi coefficient, contingency coefficient, Cramer's V coefficient**
  - The phi coefficient works for two binary variables only
  - Contingency coefficient is not normalized between 0 and 1; for example, the highest possible value for a 2x7 table is 0.707

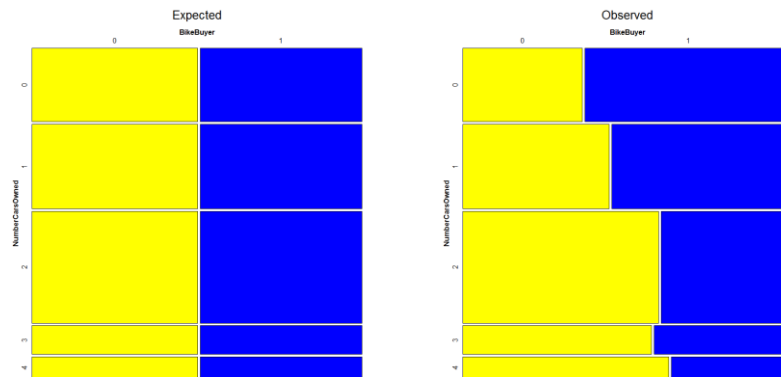
$$\phi = \sqrt{\frac{\chi^2}{n}}, \quad C = \sqrt{\frac{\chi^2}{n + \chi^2}}, \quad V = \sqrt{\frac{\chi^2}{n * (k - 1)}},$$

where  $k = \min(n \text{ of rows}, n \text{ of columns})$

- Use the `assocstats()` function from the `vcd` package

## Visualizing the Association

- The `strucplot()` function from the `vcd` package visualizes the contingency tables



## Associations

### ► Continuous Variables

- The deviation of the actual from the expected probabilities is the covariance

$$\text{cov}(X, Y) = \sum_{i=1}^n (X_i - \mu(X)) * (Y_i - \mu(Y)) * P(X, Y)$$

- The Pearson's correlation coefficient divides the covariance with the product of the standard deviations of both variables

$$\rho(X, Y) = \frac{\text{cov}(X, Y)}{\sigma(X) * \sigma(Y)}$$

- Use the `cov()` and `cor()` functions

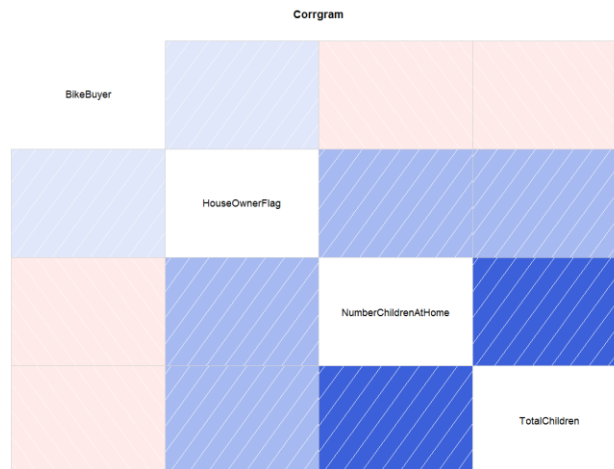
## Association

### ► Ranks

- The Spearman correlation between two variables is equal to the Pearson correlation between the rank values of those two variables
  - Pearson's correlation assesses linear relationships, Spearman's correlation assesses monotonic relationships (linear or not)
- Use the `cor()` function with the `method = "spearman"` parameter

## Visualizing the Association

- The `corrgram()` function in the `corrgram` package



## Associations

► Discrete and Continuous Variables

- ANOVA tests the variance in means between groups
  - Null Hypothesis: the only variance comes from variance within and not between samples
  - Mean squared deviation *between* *a* groups, with  $\hat{X}$  denoting group mean and  $\bar{X}_i$  denoting the total mean

$$MS_A = \frac{SS_A}{DF_A}, \text{ where } SS_A = \sum_{i=1}^a n_i * (\mu_i - \mu), \text{ and } DF_A = (a - 1)$$

## One-Way ANOVA and F-Test

- Mean squared deviation *within* *a* groups, with  $n_j$  cases in each group

$$MS_E = \frac{SS_E}{DF_E}, SS_E = \sum_{i=1}^a \sum_{j=1}^{n_i} (X_{ij} - \bar{X}_j), DF_E = \sum_{i=1}^a (n_i - 1) \quad F = \frac{MS_A}{MS_E}$$

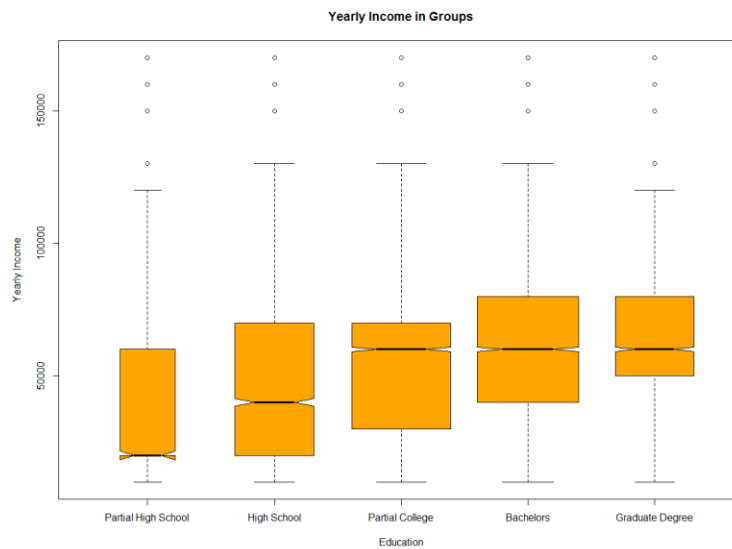
- F ratio
  - The bigger the F ratio, the more sure you can reject the Null Hypothesis
  - Use F tables for critical points
- A simpler test is Student's t-test, used for the differences between means in two groups only

## Associations in R

► Discrete and Continuous Variables

- Use the following functions:
    - `t.test()` for the Student's t-test
    - `aov()` for one-way ANOVA
    - `boxplot()` for visualizing the results
- ```
t.test(YearlyIncome ~ Gender);  
aov(YearlyIncome ~ Education);  
boxplot(YearlyIncome ~ Education,  
        main = "Yearly Income in Groups",  
        notch = TRUE, varwidth = TRUE,  
        col = "orange",  
        ylab = "Yearly Income",  
        xlab = "Education");
```

## Visualizing the Association





## Introducing ggplot2

- A very popular graphical package
- The package provides a comprehensive and coherent grammar for graphical functions
- A simple example

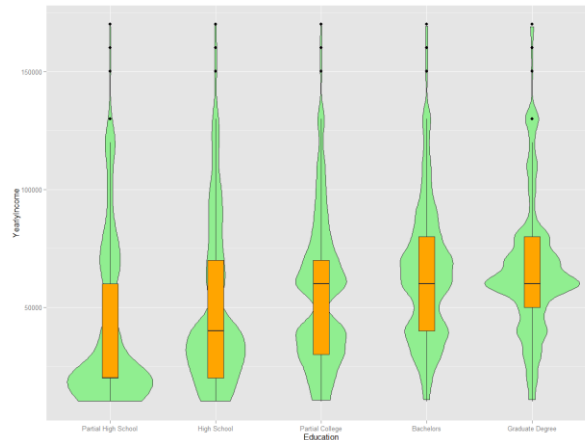
```
install.packages("ggplot2");  
library("ggplot2");  
ggplot (TM, aes(Region, fill=Education)) +  
  geom_bar(position="stack");
```

## ggplot2 Grammar

- The `ggplot()` function defines the dataset used and initializes the plot
  - The `aes()` (short for aesthetics) function defines the roles of the variables for the graph
    - In the example, the graph shows the frequencies of the Region variable, where the Region bars are filled with Education variable, to show the number of cases with each level of education in each region
  - The `geom_bar()` function defines the plot type, in this case a bar plot
  - There are many other `geom_xxx()` functions for other plot types

## Violin Plots

```
ggplot(TM, aes (x = Education, y = YearlyIncome)) +  
  geom_violin(fill = "lightgreen") +  
  geom_boxplot(fill = "orange", width = 0.2);
```



## Trellis Charts

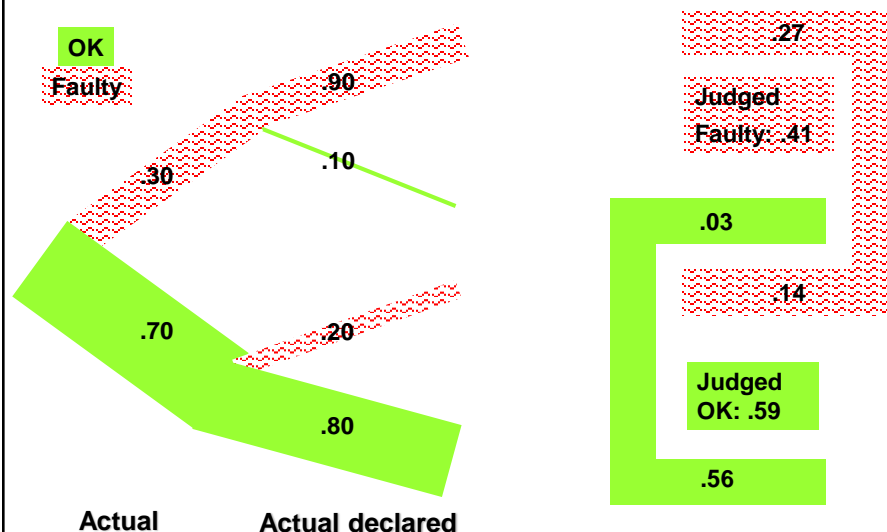
- A trellis chart is a multi-panel chart of small, similar charts, which use the same axes and same scale
  - Trellis graphs are called faceted graphs in the ggplot
  - The `facet_grid()` function defines the discrete variables to be used for splitting

```
ggplot(TM, aes(x = NumberCarsOwned, fill = Region)) +  
  geom_bar(stat = "bin") +  
  facet_grid(MaritalStatus ~ BikeBuyer) +  
  theme(text = element_text(size=30));
```

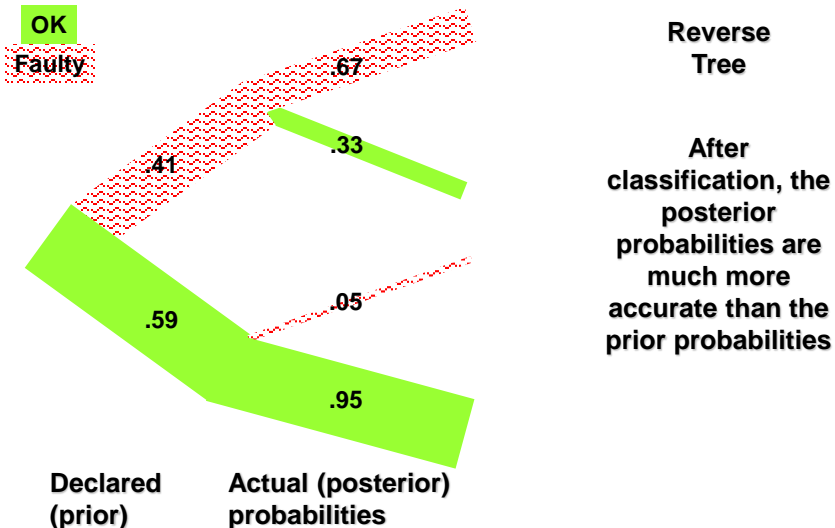
## Bayesian Inference

- The (naive) Bayes quickly builds mining models that can be used for data overview, classification and prediction
- It calculates probabilities for each possible state of the input attribute, given each state of the predictable attribute
  - The probabilities can later be used to predict an outcome of the predicted attribute based on the known input attributes
  - Input attributes are treated as mutually independent

## Naïve Bayes



## Naïve Bayes



## Example

- Table of products with Color, Class, and Weight columns
- If Color is missing, 80% of Weight values are missing as well; if Class is missing, 60% of Weight values are missing as well
  - $0.8 \text{ (Color missing for Weight missing)} * 0.6 \text{ (Class missing for Weight missing)} = 0.48$
  - $0.2 \text{ (Color missing for Weight not missing)} * 0.4 \text{ (Class missing for Weight not missing)} = 0.08$
- The likelihood that Weight is missing is much higher than the likelihood it is not missing when Color and Class are unknown

## Example

- You can convert the likelihoods to probabilities by normalizing their sum to 1:
  - $P(\text{Weight missing if Color and Class are missing}) = 0.48 / (0.48 + 0.08) = 0.857$
  - $P(\text{Weight not missing if Color and Class are missing}) = 0.08 / (0.48 + 0.08) = 0.143$

## Linear Regression Introduction

- Probably the most popular statistical method for expressing associations
- In Linear Regression, you want to show one variable (e.g., sales quantity) as a function of another
  - $Y = a + b * X$

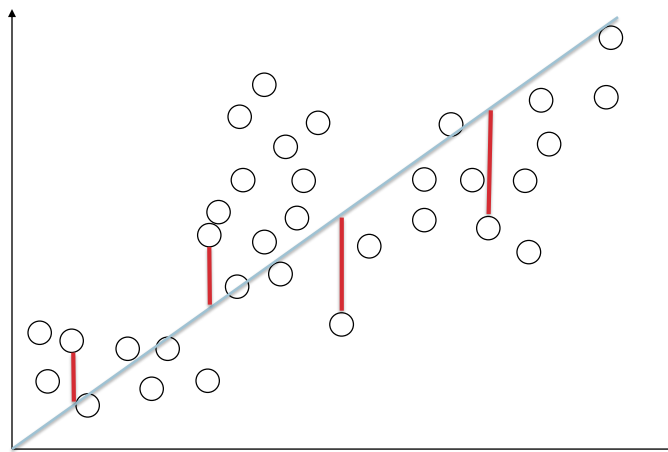
## Slope and Intercept

- You can imagine that the two variables form a two-dimensional plane
  - Their values define coordinates of the points in the plane
  - You are searching for a line that best fits all of the points
  - You need to use the deviations from the line (the difference between the actual value for  $Y_i$  and the line value  $\hat{Y}$ )
  - Some deviations are positive and others negative, so the sum of all the deviations is zero for the best-fit line
  - You square the deviations instead

$$\text{Slope}(Y) = \frac{\sum_{i=1}^n (X_i - \mu(X)) * (Y_i - \mu(Y))}{\sum_{i=1}^n (X_i - \mu(X))^2}$$

$$\text{Intercept}(Y) = \mu(Y) - \text{Slope}(Y) * \mu(X)$$

## Linear Regression



## Advanced Linear Regression

- Multiple regression allows a response variable  $Y$  to be modeled as a linear function of a multidimensional feature vector
  - $Y = b_0 + b_1 * X_1 + b_2 * X_2 + \dots + e$
- In polynomial regression, you express the association with a formula where the independent variable is introduced in the equation as an  $n^{\text{th}}$  order polynomial

## Linear Regression in R

- Use the `lm()` function for all kinds of linear regression
- Multiple

```
LinReg1 <- lm(YearlyIncome ~ Age + ...)
summary(LinReg1)
```
- Polynomial

```
LinReg2 <- lm(YearlyIncome ~ Age + I(Age ^ 2))
summary(LinReg2)
```

## Assumptions

- Linearity: there is a linear relationship
- Non-correlation of errors: errors should not be correlated
- Homoscedasticity: variance of errors is normally distributed constant across the input
- No collinearity: no associations between input variables
- Outliers: there are no such huge outliers that would influence the model
- Check the validity with model plots

80

## Resources

- The [ggplot2](#) library at tidyverse
- [ggMarginal\(\)](#) function from the ggExtra
- The [e1071](#) package
- The [vcd](#) library
- [R in Action](#) by Robert I. Kabacoff, Manning, 2015
- [Applied Data Mining](#) by Paolo Giudici and Silvia Figini, Wiley, 2009



## Module 04:

# Matrix operations and feature selection

## Agenda

- Feature selection in linear models
- Basic matrix algebra
- Principal component analysis
- Exploratory factor analysis

## Feature Selection (FS)

- Input variables, or features, could be numerous
- Collinearity might be a problem
- The question is: which features to select?
  - Can start by manually dropping some of the highly associated ones
  - Can select the features to keep with some advanced methods
  - Expect drop in prediction quality

84

## Basic FS Methods

- Forward stepwise regression: add feature one by one
  - Improves residual sum of squares (RSS) and R-squared, but might not improve the model fit and interpretability
- Backward stepwise regression: remove the least useful fetures one by one
- Best subsets regression: fit the model for all possible feature combinations
  - Performance might not be satisfactory

85

## Regularization Methods

- Ordinal sum of squares method (OSS)  

$$Y = b_0 + b_1 * X_1 + b_2 * X_2 + \dots + e$$
 tries to minimize the error, or residual sum of squares:  

$$RSS = e_1^2 + e_2^2 + e_3^2 + \dots e_n^2$$
- Regularization methods implement **shrinkage penalty**  $\lambda$  for the beta coefficients  
 Minimize  $RSS + \lambda * (\text{normalized coefficients})$
- **Ridge** (L2-norm) regression:  

$$\min(RSS + \lambda * (\text{sum}(B^2)))$$
- **Lasso** (L1-norm) regression:  

$$\min(RSS + \lambda * (\text{sum}(|B|)))$$

86

## Vectors and Matrices

- A vector **x** of order **p** (or dimension p) is a column of p numbers  
 – `x <- c(2,0,0,4)` # Use the *combine* function to generate a vector
- An **m** x **n** matrix **X** is a rectangular array of scalar values  
 – Use the *matrix* function to generate a matrix from a vector  
 – Use the *array* function to generate a 2-dim array from a vector
- Matrix operations  
 – Addition, subtraction  
 – Multiplication – only for matrices where the number of rows of the first one is equal to the number of columns of the second one  
 – Combine by rows or by columns

## Determinants

- With every square matrix you can define a determinant – example 2x2 matrix:

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$$

- 3x3 Leibnitz formula: 
$$\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = a \begin{vmatrix} e & f \\ h & i \end{vmatrix} - b \begin{vmatrix} d & f \\ g & i \end{vmatrix} + c \begin{vmatrix} d & e \\ g & h \end{vmatrix}$$

- General formula: 
$$\det(A) = \sum_{\sigma \in S_n} \left( \text{sgn}(\sigma) \prod_{i=1}^n a_{i,\sigma_i} \right)$$
  
The sum is computed over all permutations  $\sigma$  of the set  $\{1, 2, \dots, n\}$

88

## Eigenvectors and Eigenvalues

- We can deconstruct the data points matrix into eigenvectors and eigenvalues
  - Every eigenvector has a corresponding eigenvalue
  - An eigenvector is a direction of the line
  - An eigenvalue is a number, telling how much variance there is in the data in that direction, or how spread out the data is on the line
  - The eigenvector with the highest eigenvalue is therefore the principal component

## Eigenvectors and Eigenvalues

Let  $A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$ ,  $v = \text{vector}$ ,  $\lambda = \text{scalar value}$ ,  $I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

If  $Av = \lambda v$  then  $v = \text{eigenvector}$  and  $\lambda = \text{eigenvalue}$

$(A - \lambda I)v = 0$  when  $|A - \lambda I| = 0$ ,

characteristic polynomial  $p(\lambda) = |A - \lambda I| = 0$

$p(\lambda) = 3 - 4\lambda + \lambda^2 = 0$  has roots when  $\lambda = 1$  or  $\lambda = 3$

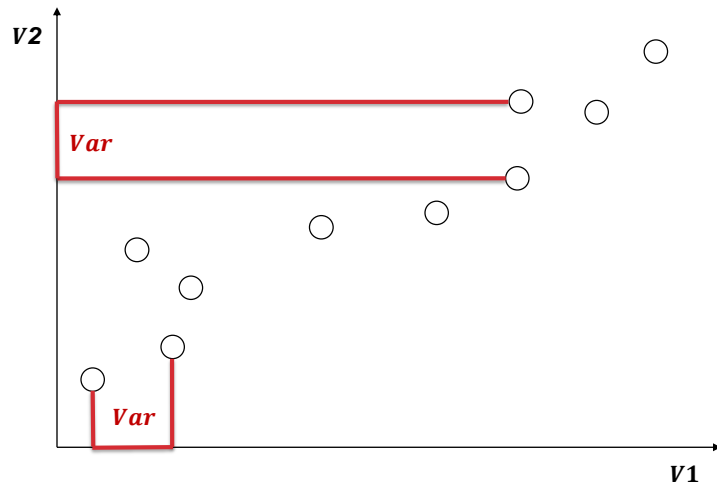
$(A - \lambda I)v = 0$  for eigenvalue  $\lambda = 1$  gives eigenvector  $v = \{1, -1\}$

$(A - \lambda I)v = 0$  for eigenvalue  $\lambda = 3$  gives eigenvector  $v = \{1, 1\}$

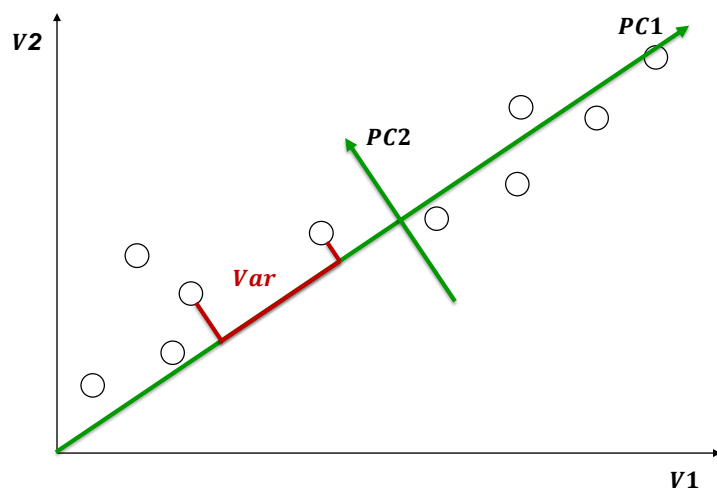
## Principal Component Analysis

- Principal component analysis (PCA) is a technique used to emphasize the majority of the variation and bring out strong patterns in a dataset
  - It's often used to make data easy to explore and visualize
- Closely connected with eigenvectors and eigenvalues
- Variables form a multidimensional space, or matrix, of dimensionality  $m$

## Principal Component Analysis



## Principal Component Analysis



## Rotation

- You can improve the understanding of the PCs by rotating them
  - This means you are rotating the axes of the multidimensional hyperspace
  - The rotation is done in such a way that it maximizes associations of PCs with different subsets of input variables each
  - The rotation can be orthogonal, where the rotated components are still uncorrelated, or oblique, where correlation between PCs is allowed
  - In PCA, use orthogonal rotation, because you want to use uncorrelated components in further analysis

## PCA Usage

- The interpretation of the principal components is up to you
  - Might limit PCA usability for business-oriented problems
  - Used more in machine learning than data mining
- Explore the data to explain the variability
- Reduce the dimensionality – replace the variables with m-n principal components in way that preserves the most of the variability
- Use the residual variability not explained by the PCs for anomaly detection

## PCA in R

- Use the `princomp()` function from base installation
- Even better the `principal()` function from the `psych` package

```
# PCA not rotated
pcaTM_unrotated <- principal(TMPCAEFA, nfactors = 2,
rotate = "none")
pcaTM_unrotated
# PCA rotated
pcaTM_varimax <- principal(TMPCAEFA, nfactors = 2,
rotate = "varimax")
pcaTM_varimax
```

## Exploratory Factor Analysis

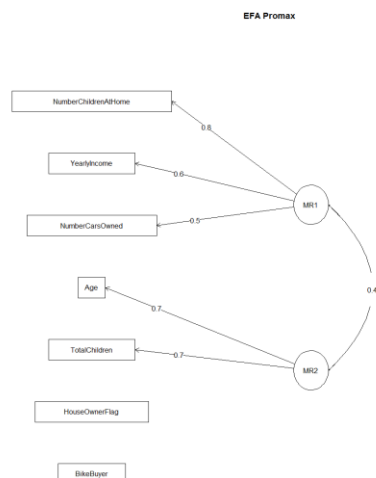
- When you do the EFA, you definitely want to understand the results
  - The factors are the underlying combined variables that help you understand your data
  - This is similar to adding computed variables, just in a more complex way, with many input variables
    - For example, the obesity index could be interpreted as a very simple factor that includes height and weight, and gives you much more information about person's health than the base two variables do
  - Therefore, you typically do rotate the factors, and also allow correlations between them



## EFA with R

- Use the `fa()` function from the `psych` package
    - The promax rotation is an oblique rotation
    - Use `factor.plot()` and `fa.diagram()` functions to show the results of the EFA graphically
- ```
efaTM_promax <- fa(TMPCAEFA, nfactors = 2,
  rotate = "promax")
efaTM_promax;
fa.diagram(efaTM_promax, simple = FALSE,
  main = "EFA Promax")
```

## EFA Graph



## Resources

- The [glmnet](#) package
- The [psych](#) package
- [Basics of Matrix Algebra for Statistics with R](#) by Nick Fieller, CRC Press, 2016
- [SQL Server 2017 Machine Learning Services with R](#) by Tomaž Kaštrun and Julie Koesmarno, Packt, 2018
- [The Elements of Statistical Learning, Data Mining, Inference, and Prediction](#) by Trevor Hastie, Robert Tibshirani and Jerome Friedman, Springer, 2009

## Module 05: Unsupervised learning

## Agenda

- Hierarchical clustering
- K-means clustering
- Association rules

102

## Clustering

- Grouping cases into clusters
  - Objects within a cluster have a high similarity based on attribute values
  - The class label of each object is not known
- Several techniques
  - Partitioning methods
  - Hierarchical methods
  - Density-based methods
  - Model-based methods

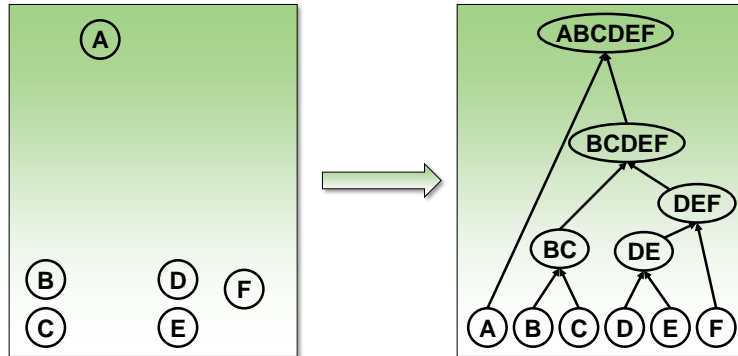
## Hierarchical Clustering

- A hierarchical method creates a hierarchical decomposition of the given set of data objects
  - These methods are *agglomerative* or *divisive*
- The agglomerative (bottom-up) approach starts with each object forming a separate group
  - It successively merges the objects or groups close to one another, until all groups are merged into one
- The divisive (top-down) approach starts with all the objects in the same cluster
  - In each successive iteration, a cluster is split up into smaller clusters, until eventually each object is in one cluster or until a termination condition holds

## Hierarchical Clustering


- A dendrogram is a tree diagram frequently used to illustrate the arrangement of the clusters produced by hierarchical clustering
  - Dendrograms are often used in computational biology to illustrate the clustering of genes or samples

## Hierarchical Clustering - Dendrogram




## Measures of Distance


Euclidean

$\sqrt{2}$	1	$\sqrt{2}$
1		1
$\sqrt{2}$	1	$\sqrt{2}$

Chebyshev

1	1	1
1		1
1	1	1

Manhattan

2	1	2
1		1
2	1	2

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad \max(|x_1 - x_2|, |y_1 - y_2|) \quad |x_1 - x_2| + |y_1 - y_2|$$

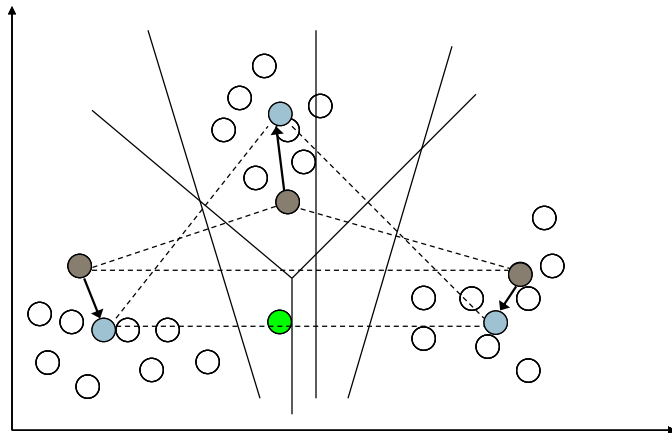
## K-Means

- Divides dataset in predetermined (“k”) number of clusters around the average location (“mean”)
- Algorithm comes from geometry
- Imagine record space with attributes as dimensions
  - Each record (case) is uniquely located in multidimensional space with values of the attributes (variables)

## K-Means

- K-Means initially randomly selects k means (centroids)
  - Because it does not know the clusters yet, these must be fictitious cases
- It assigns each record to the nearest centroid
  - These are the initial clusters
- It calculates new centroids of clusters
  - It reassigns each record to the nearest centroid
- Some records jump from cluster to cluster
  - It iterates the last two steps until cluster boundaries stop changing

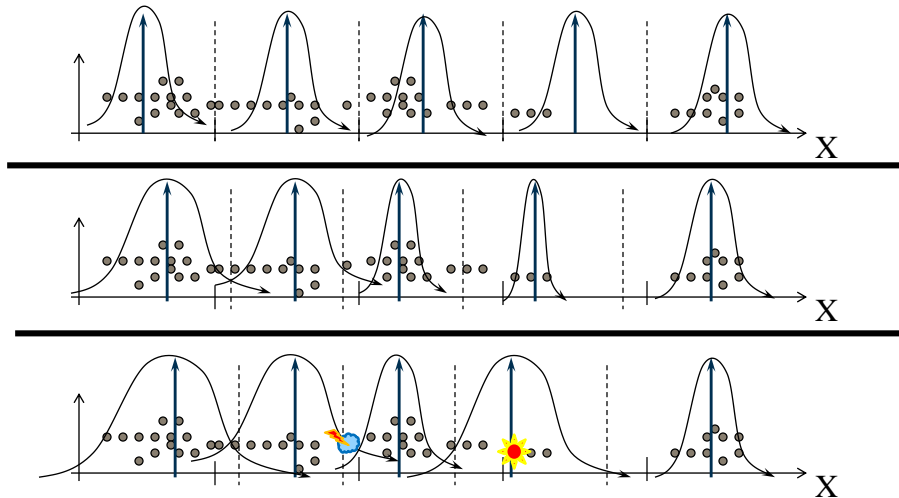
## K-Means



## Hard vs. Soft Clustering

- With the K-Means algorithm, each object is assigned to exactly one cluster
  - This is hard clustering
- Instead of distance, you can use a probabilistic measure to determine cluster membership
  - Cover the objects with bell curves for each dimension with a specific mean and standard deviation – Gaussian Mixture Models (GMM)
  - A case is assigned to every cluster with a probability
  - Because clusters can overlap, this is soft clustering
  - Expectation-Maximization (EM) part changes bell curve parameters to improve covering in each iteration

## Expectation - Maximization



## Clustering Usage

- Cluster analysis segments a heterogeneous population into a number of more homogenous subgroups or clusters
- Typical usage scenarios include:
  - Discovering distinct groups of customers
  - Identifying groups of houses in a city
  - In biology, deriving animal and plant taxonomies
  - Finding outliers (soft clustering only)
  - Detecting fraudulent transactions (soft clustering only)
  - Can even make predictions once the clusters are built



## Association Rules

- The Association Rules algorithm considers each attribute/value pair (such as product/bicycle) as an item
- An itemset is a combination of items in a single transaction
- The algorithm scans through the dataset trying to find itemsets that tend to appear in many transactions

## Association Rules - Support

- *Support* is the number of rows with the itemset compared to the total number of rows:
  - Transaction 1: Frozen pizza, cola, milk
  - Transaction 2: Milk, potato chips
  - Transaction 3: Cola, frozen pizza
  - Transaction 4: Milk, pretzels
  - Transaction 5: Cola, pretzels
- The support for the rule "If customers purchase cola, then they will purchase frozen pizza" is 0.4
  - There are 5 total records, and 2 of them include both cola and frozen pizza

## Association Rules - Confidence

- Which rule is better?
  - If customers purchase milk, then they will purchase potato chips
  - If customers purchase potato chips, then they will purchase milk
- The *confidence* of an association rule is the support for the combination divided by the support for the condition
  - The support for the combination (potato chips + milk) is 20%, occurring in 1 of the 5 transactions
  - The support for milk is 60%, occurring in 3 of the 5 transactions, and the support for potato chips is 20%
- This gives a confidence of 33% (20% / 60%) for the rule "If customers purchase milk, they will purchase potato chips" and a confidence of 100% for the rule "If customers purchase potato chips, then they will purchase milk"

## Association Rules - Importance

- The interesting score of the lift
- Importance of an itemset:
 
$$\text{Importance}(A,B) = P(A,B) / (P(A) * P(B))$$
  - If importance = 1, A and B are independent items
  - If importance > 1, A and B are positively correlated
  - If importance < 1, A and B are negatively correlated
- Importance of a rule:
 
$$\text{Importance}(A \Rightarrow B) = \log( P(B|A) / P(B|\text{not } A) )$$
  - If importance = 0, there is no association between A and B
  - If importance > 0, the probability of B goes up when A is true

## Resources

- The [NbClust](#) function from the NbClust package
- The [arules](#) package
- The [arulesViz](#) package
- [Mastering Machine Learning with R](#) by Cory Lesmeister, Packt, 2017
- [Statistics for Machine Learning](#) by Pratap Dangeti, Packt, 2017

## Module 06: Supervised learning

## Agenda

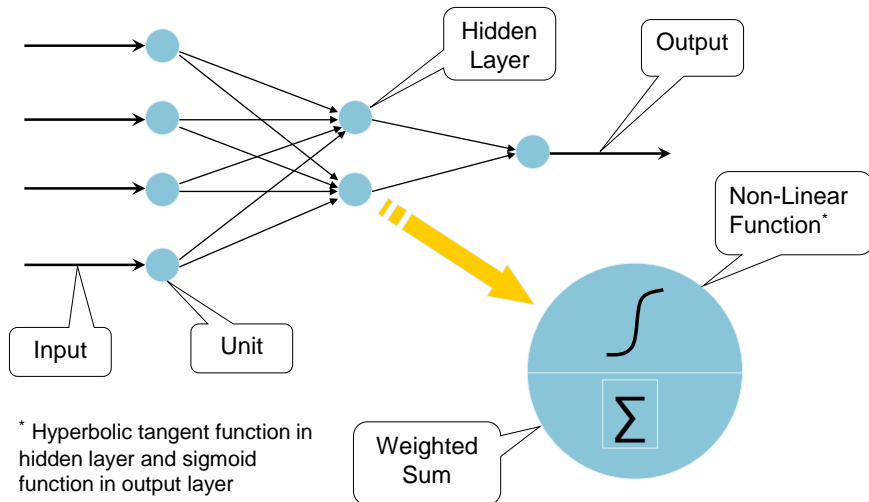
- Neural Network
- Logistic Regression
- Trees and forests
  - Decision trees
  - Random forests and gradient boosting trees
- K-nearest neighbors

120

## Neural Network

- A neural network is a data modeling tool that can capture and represent complex input/output relationships
- Neural networks resemble the human brain in the following two ways:
  - It acquires knowledge through learning
  - It's knowledge is stored within inter-neuron connection strengths known as synaptic weights
- The Neural Network algorithm explores more possible data relationships than the other algorithms

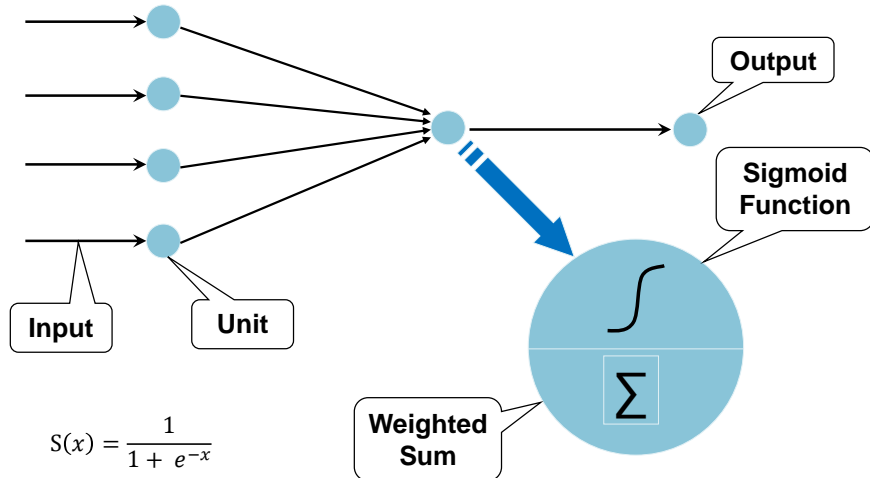
## Neural Network



## Backpropagation

- Training a neural network is the process of setting the best weights on the inputs of each of the units
- The backpropagation process:
  - Gets a training example and calculates outputs
  - Calculate the errors – the difference between the calculated and the expected (known) result
  - Adjusts the weights to minimize the error

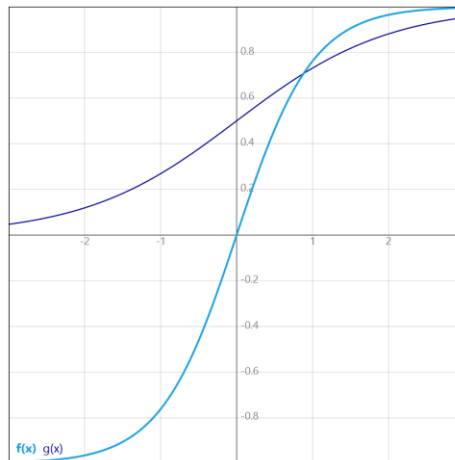
## Logistic Regression



## Activation Functions

$$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

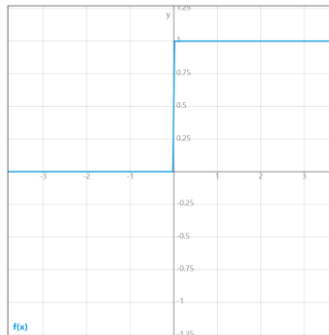
$$g(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$$



## Perceptron Model

- The perceptron method is an early and very simple version of a logistic regression
  - It is linear classifier, using a linear function after combining a set of weights with the feature vector
  - No hidden layer, only unit step discontinued activation function

$$f(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$



## Logistic Regression in R

- Use the `glm()` function from the base installation
- Use the `predict()` function to make the predictions on the test set
- Create multiple models with different input variables to get the best predictions

## Decision Trees

- Decision Trees assign (classify) each case to one of a few (discrete) broad categories of selected attribute (variable) and explains the classification with few selected input variables
- Once built, they are easy to understand
- They are used to predict values of the explained variable
- Recursive partitioning is used to build the tree
  - Data is split into partitions and then split up more
- Initially all cases are in one big box

## Decision Trees

- The algorithm tries all possible breaks in classes using all possible values of each input attribute; it then selects the split that partitions the data to the purest classes of the searched variable
  - Uses several measures of purity, such as frequency distribution, entropy, and Bayesian scoring of prior / posterior probabilities
- It then repeats the splitting process for each new class, again testing all possible breaks
- The problem is where to stop



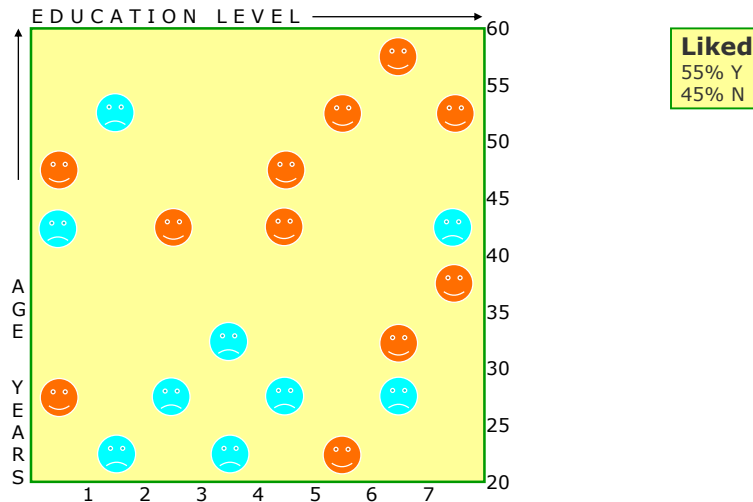
## Decision Trees

- A common problem is over-fitting
- Not useful branches of the tree can be pre-pruned or post-pruned
- Pre-pruning methods try to stunt the growth of the tree before it grows too deep
  - They test each node to see whether a further split would be useful; the tests can be simple (n of cases) or complicated (complexity penalty)
- Post-pruning methods allow the tree to grow and then prune off branches
  - Again the test can be simple (n of cases) or more complex

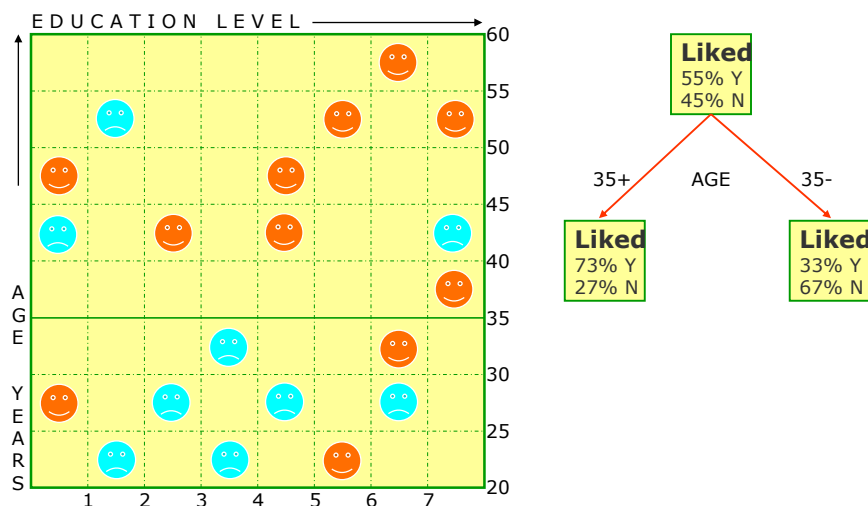
## Example

- Interview of the people who watched the famous “Woodstock” movie
  - You have a population aged between 20 and 60 years old
  - You gathered data about their education and ranged it into 7 classes (1 = lowest, 7 = highest)
- 55% of them liked the movie
- Can you discover the factors that have an influence on whether they liked the movie?

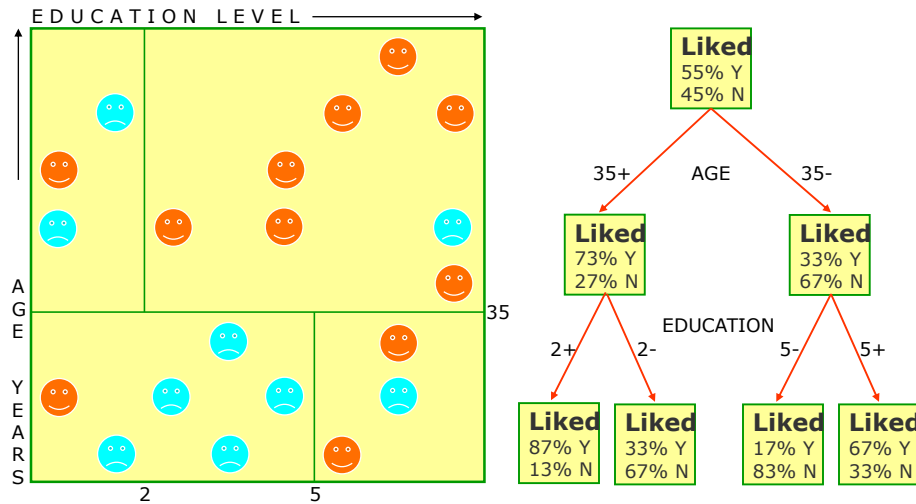
## Example



## Example



## Example



## Decision Trees Usage

- Decision Trees are used for classification and prediction
- Typical usage scenarios include:
  - Predicting which customers will leave
  - Targeting the audience for mailings and promotional campaigns
  - Explaining the reasons for a decision
  - Answering questions such as “What movies do young urban customers buy?”

## Random Forests

- The **random forests** algorithm builds many decision trees on the same training set, using random subsets from the training set
- Then it averages the predictions from all of the models
- This algorithm is very resource-intensive
  - However, it can process multiple trees in parallel

## Gradient-Boosting Trees

- The **gradient-boosting trees** also build many decision trees
  - However, they build one by one, serially
- For each tree, the algorithm selects a random subset of cases from the training set and adds some cases for which the predictions were wrong in the previous step
- This way, the algorithm lowers the false predictions

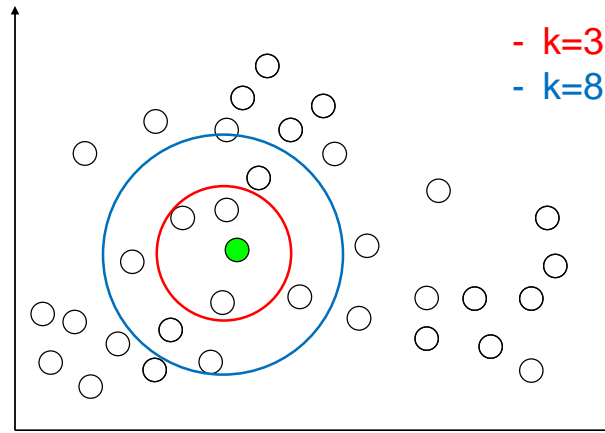
## Decision Trees in R

- Use the `rpart()` function from the base installation
- Use the `prp()` function from the `rpart.plot` package to graphically represent the tree
- Use the `ctree()` function from the `party` package for even better predictions
  - Conditional inference trees
  - The function accepts many parameters
- For the random forests and gradient-boosting trees, use the `rxDForest()` and `rxBTrees()` functions of the RevoScaleR package

## K-Nearest Neighbors (KNN)

- Instance-based learning
  - No action until a new input pattern demands an output value – lazy learning
  - For predicting a class of target variable of a new case, k train cases that most resemble the new instance are searched, using the input features
  - Small k might lead to high variance, large k decreases variance, but raises bias
  - Might get local maximums
- Scale the variables
  - The `scale()` function – Z-score normalization

## KNN



## Resources

- The [neuralnet](#) package
- The [rparty](#) library
- The [RevoScaleR](#) package
- The [ROCR](#) library
- The [MASS](#) package
- [Data Mining: Concepts and Techniques](#) by Jiawei Han, Micheline Kamber, Jian Pei (Author), Morgan Kaufmann, 2012
- [Mastering Data Mining: The Art and Science of Customer Relationship Management](#) by Michael J. A. Berry, Gordon S. Linoff, Wiley, 2000

## Module 07: Modern topics

### Agenda

- Support vector machines
- Time series
- Text mining
- Deep learning
- Reinforcement learning

## Support Vector Machines

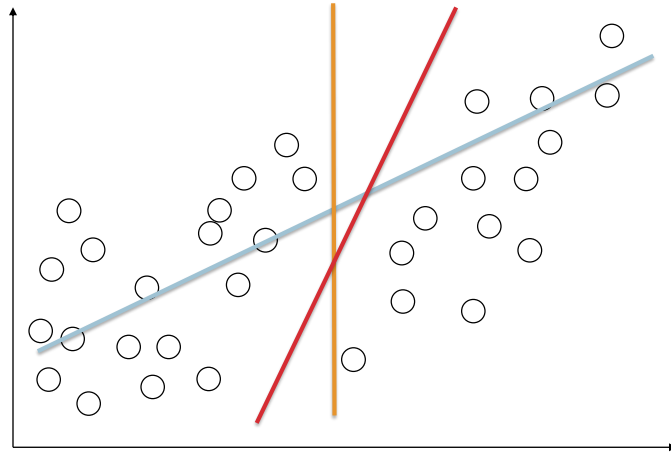
- Support vector machines are both, unsupervised and supervised learning models for classification and regression analysis (supervised) and for anomaly detection (unsupervised)
  - Given a set of training examples, each marked as belonging to one of categories, an SVM training algorithm builds a model that assigns new examples into one category
  - An SVM model is a representation of the cases as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible
  - New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on

## Support Vector Machines

- A support vector machine constructs a hyperplane or set of hyperplanes in a high-dimensional space, which can be used for classification, regression, or other tasks
  - Discrete linear classifier
- A good separation is achieved by the hyperplane that has the largest distance to the nearest training data point of any class (so-called functional margin)
  - The larger the margin the lower the generalization error



## Support Vector Machines



## Support Vector Machines Usage

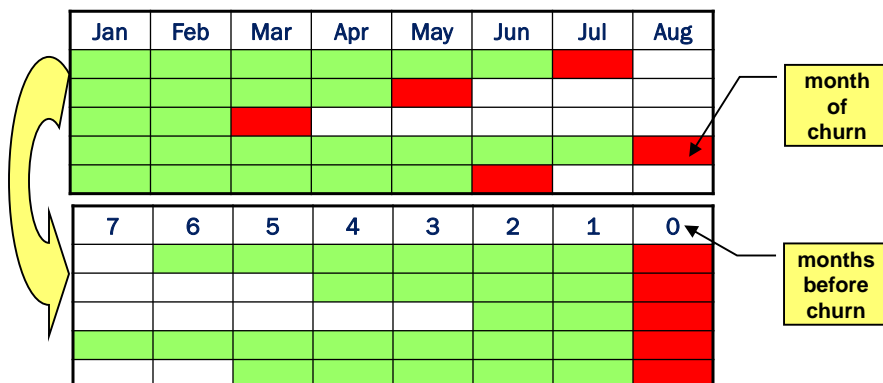
- Support Vector Machines are powerful for some specific classifications:
  - Text and hypertext categorization
  - Images classification
  - Classifications in medicine
  - Hand-written characters recognition
- One-class SVM can be used for anomaly detection
  - Uses a classification function without parameters
  - Dichotomous result: 1=regular case, 0=outlier

## Time Series Data Preparation

- Time series classical decomposition:
  - Trend
  - Seasonality
  - Cycles
  - Noise
- To get accurate results, many different techniques might be necessary
  - De-trending (e.g., inflation)
  - Filtering and smoothing to remove seasonality
  - Standardization (or normalization) of time points

## Time Points Standardization

Examine the last couple of months before churn; churn of different customers occurs at a different time



## Multiple Linear Regression

- Multiple regression allows a response variable  $Y$  to be modeled as a linear function of a multidimensional feature vector

$$Y = a + b_1X_1 + b_2X_2 + \dots$$

- Auto Regression means the value of  $X$  at a time point is a function of the values of  $X$  at previous time points
- Linear regression predicts mean value
  - Can also calculate estimated variation

## Auto Regression Trees – Data Transformation

Month	Product	Sales
Aug	Drama	106
Aug	Action	85
Aug	Sci-Fi	48
Sept	Drama	104
Sept	Action	87
Sept	Sci-Fi	43
Oct	Drama	110
Oct	Action	92
Oct	Sci-Fi	42
Nov	Drama	125
Nov	Action	102
Nov	Sci-Fi	55
Dec	Drama	145
Dec	Action	118
Dec	Sci-Fi	67

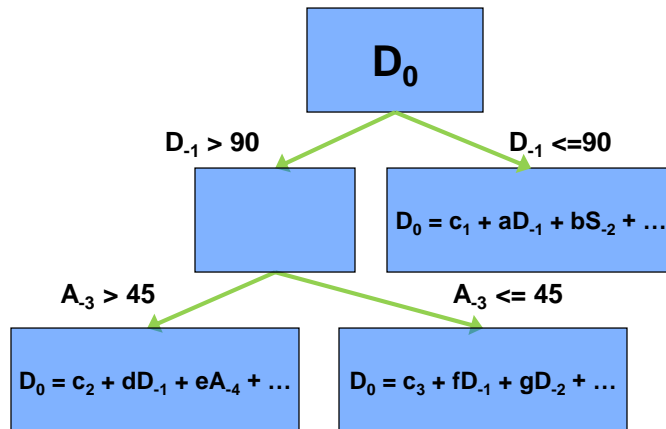


Case	$D_0$	$D_{-1}$	$D_{-2}$	$A_0$	$A_{-1}$	$A_{-2}$	$S_0$	...
1	106	102	103	85	83	80	44	
2	104	106	102	87	85	83	45	
3	110	104	106	92	87	85	42	
4	125	110	104	102	92	87	55	
5	145	125	110	118	102	92	87	
6	105	145	125	86	118	102	45	
7	102	105	145	88	86	118	45	

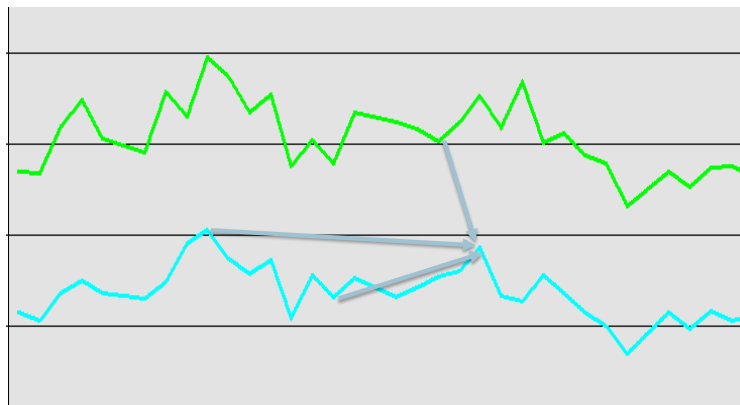


**Predictable Columns**

## Auto Regression Trees



## Auto Regression Trees with Cross Prediction (ARTXP)



## Moving Averages

- Simple moving averages (SMA)

$$S_i = (\sum_{i-1}^{i+1} v_i) / 3$$

- Weighted moving averages (WMA)

$$S_i = (\sum_{i-2}^i v_i * w_i) / 3$$

$$\sum w_i = 1$$

- Exponential moving averages (EMA)

$$S_i = (v_i * \alpha) + (S_{i-1} * \beta)$$

$$\alpha + \beta = 1$$

## Lag and Difference Operators

- Lag operator operates on an element of a time series to produce the previous element

$$LY_t = Y_{t-1}$$

$$\varepsilon_t = Y_t - \sum_{i=1}^p \phi_i * Y_{t-i} = (1 - \sum_{i=1}^p \phi_i * L^i) * Y_t$$

- Can be raised to arbitrary integer powers
- Can have lag polynomials
- Can express error  $\varepsilon$  as a lag polynomial of previous errors

$$\Delta Y_t = Y_t - Y_{t-1} = (1 - L)Y_t$$

$$\Delta^d Y_t = (1 - L)^d Y_t$$

- Difference operator is a special case of a lag polynomial

$$L = \text{Lag}$$

$$Y = \text{Value}$$

$$\varepsilon = \text{Error}$$

$$\phi = \text{Parameter}$$

$$\Delta = \text{Difference}$$

## Auto Regressive Integrated Moving Average (ARIMA)

- Auto Regressive Integrated Moving Average (ARIMA)
- ARIMA(p,d,q)
  - p is the number of autoregressive terms
  - d is the number of non-seasonal differences
  - q is the number of lagged forecast errors in the prediction equation
- General ARIMA equation

$$(1 - \sum_{i=1}^p \phi_i * L^i)(1 - L)^d * Y_t = (1 + \sum_{i=1}^q \theta_i * L^i) * \varepsilon_t$$

## ARIMA Examples

- ARIMA(0,0,1) – random walk
- ARIMA(1,1,0) – differenced first-order autoregressive model
- ARIMA(0,1,1) with constant = simple exponential smoothing with growth
- ARIMA(1,1,1)

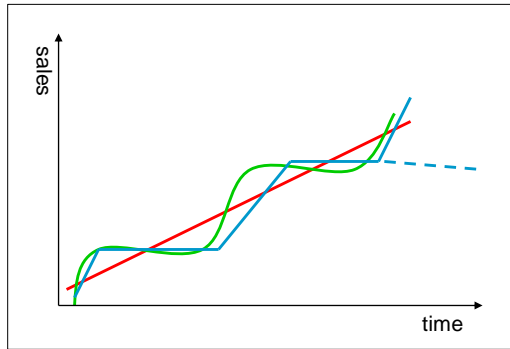
$$Y_t = \mu + Y_{t-1}$$

$$Y_t = \mu + Y_{t-1} + \phi(Y_{t-1} - Y_{t-2})$$

$$Y_t = \mu + Y_{t-1} - \theta * \varepsilon_{t-1}$$

$$Y_t = \mu + Y_{t-1} + \phi(Y_{t-1} - Y_{t-2}) - \theta * \varepsilon_{t-1}$$

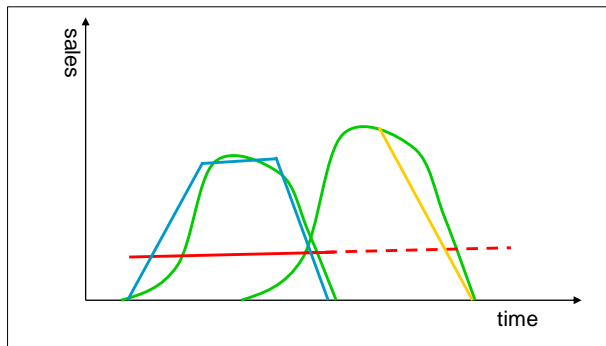
## Which Algorithm Is Better?



Actual sales  
Moving averages smoothed to simple linear regression  
Piecewise linear regression

**Note: better fit does not always mean better prediction!**

## Which Algorithm Is Better?



Actual sales (two products)  
Moving averages smoothed to simple linear regression - all time  
Simple linear regression - recent time  
Piecewise linear regression

**Note: better fit does not always mean better prediction!**

## Which Algorithm Is Better?

- ARTXP is better for short-term forecasting
- ARIMA is better for long-term forecasting
- The correct mixture also depends on data preparation
  - If values are smoothed in the data preparation stage, usefulness of ART arises
- If the algorithm predicted results in the past correctly, you can trust it
  - Can build historical models

## Time Series Usage

- Typical usage of Time Series is forecasting continuous variables
  - What will the sale amount of a specific product be in a specific region next year?
- Planning, budgeting, ...



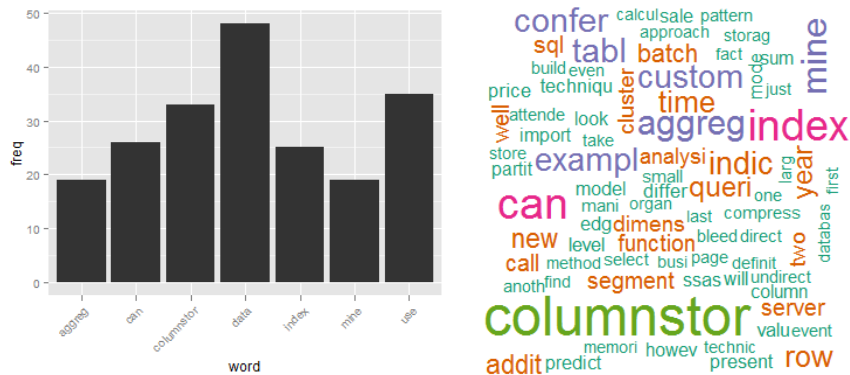
## Analyzing Text in R (1)

- Define “corpus” – set of documents to analyze
- Read the documents from several sources in several supported formats
  - R data frame, directory, URI, R vector, XML
  - CSV, DOC, PDF, XML
- Use different transformations to prepare the text for analysis
  - Change special characters, lowercase, remove punctuation, remove stopwords, strip space
  - Use stemming

## Analyzing Text in R (2)

- Create document term matrix
- Calculate term frequencies
  - Remove sparse terms
  - Find most frequent terms
- Find associations
- Use different plots
  - Term frequency
  - Word clouds
  - Term length frequency
  - Letter frequency

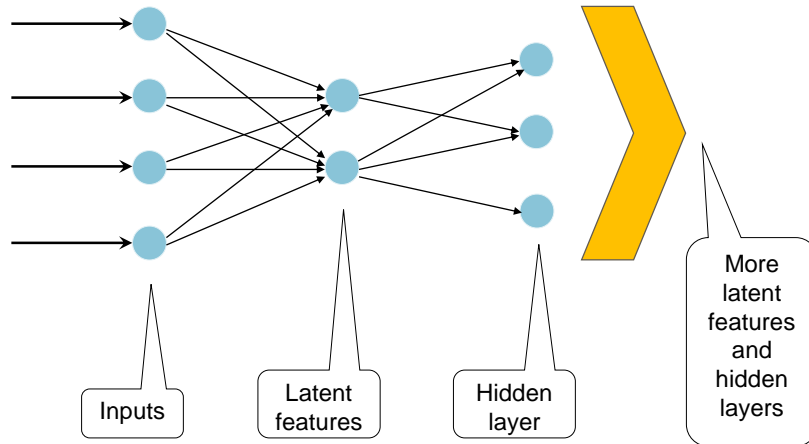
## Analyzing Text in R (3)



## Deep Learning

- Based on neural networks
- Finding the latent structure in data without the response variable
  - Creates multiple hidden layers and layers with latent features between them
  - Penalizes weights with regularization methods (Ridge, LASSO,...)
  - Randomly ignores certain inputs
- Can use the open source H2O platform to test deep learning and other algorithms

## Deep Learning

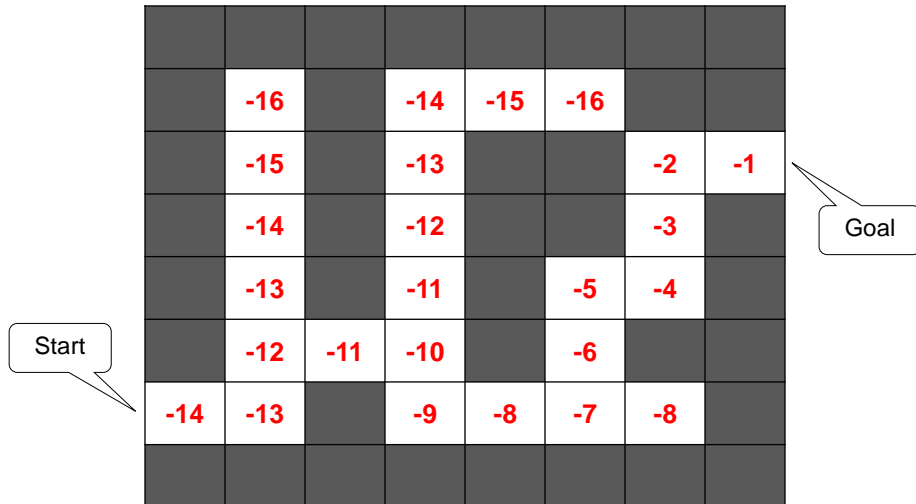


## Reinforcement Learning

- A new major area of machine learning and artificial intelligence
- A process of sequential decisions
  - Interaction with environment, which gives awards
  - Repeating actions for which the award is higher
  - Avoiding moves with low or negative award
  - No target variable; an algorithm searches for the path on its own, based on the signals (awards)
  - The award signal might not be instantaneous, time might be important
  - The decisions affect the subsequent data the algorithm receives

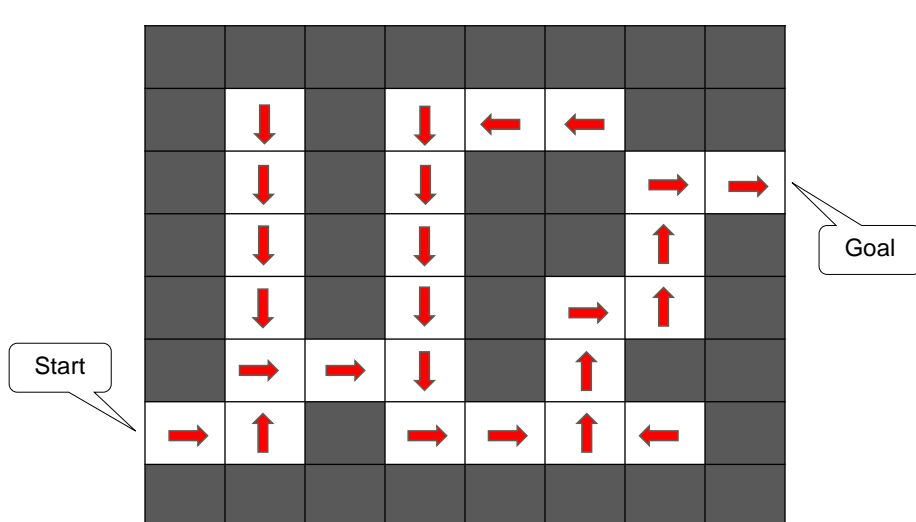
167

## Value Award Example



168

## Policy Award Example



169

## Resources

- [Getting started with deep learning in R](#) by Sigrid Keydana, blog, 2018
- The [H2O](#) open source analytics platform
- [Reinforcement learning in R](#) by Nicolas Proellocks, a short introduction to the ReinforcementLearning package, CRAN, 2018
- [The Essentials of Data Science](#): Knowledge Discovery Using R by Graham J. Williams, CRC Press, 2017
- [Modern R Programming Cookbook](#) by Jaynal Abedin, Packt, 2017
- [Data Science with SQL Server Quick Start Guide](#) by Dejan Sarka, Packt, 2018