Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Business Analytics using Data Mining

BU7143

Dr. Nicholas P. Danks
Business Analytics
nicholas.danks@tcd.ie

(Merged) Outline

Session	Date & Venue	Lecture & readings
1		Introduction: Business and Statistical Challenges Classification, Prediction, Forecasting, Clustering, Supervised, etc. Reading: Chap. 2
2		**Dimension reduction, & Performance evaluation** Reading: Chap. 4 and 5
3		**General Regression**: Explanation and Prediction, Stationarity, Variable types Reading: Chap. 6
4		**Time Series Data**: Linear Regression with ext. predictors; Lags; Trend, Seasonality, Level, Noise Reading: Chap. 16 (and feedback on projects)
5		**Smoothing**: Simple, & Exponential smoothing Reading: Chap 17 and 18
7		**Group projects - Forecasting** presentation and feedback Reading: TBD

Tools we will use

Coding language

Install R:
http://www.r-project.org/

Integrated Development Environment

Install RStudio:
http://www.rstudio.com/

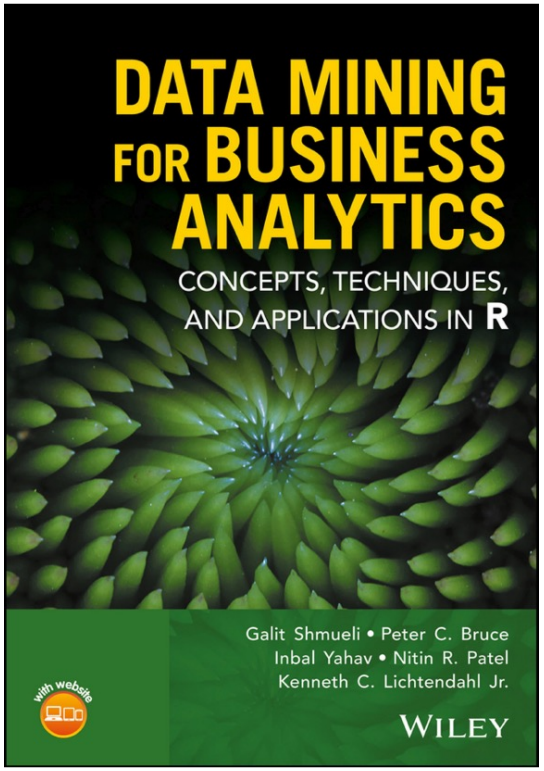
Version control

Join GitHub:
https://github.com/

Textbook

Data Mining for Business Analytics in R

Shmueli, Bruce, Yahav, Patel & Lichtendahl



Grading

Presentation (20%)
15 mins
±10 Slides
Written Report (40%)
4 – 6 pages
1500 words
Homework (40%)
Weekly (40/6 = 6.7% per lesson)

ASSESSMENT
Group Assignment (60%)
The group assignment will take the form of a detailed business challenge translated into a statistical forecasting problem. It will detail the application of several possible methods for generating forecasts, their relative suitability and performance. Students will be evaluated on the business insights and conclusions, predictive performance, and ability to communicate effectively. It will include a group presentation and written. The deadline to submit your assignment is included in the schedule.

Weekly Homework (40%)
Weekly homework assignment will track the progress and learning of students. To help participants prepare for the homework, weekly tutorials will be held to discuss the homework problems.

Presentation & Report

Executive Summary

Problem description

Business goal:

Analytics goal:

Data description

Brief data preparation / cleaning details

Datamining solution

Comparison of performance

Conclusions

Advantages and Limitations

Operational Recommendations

Refer to the demo report and presentation (Blackboard)

Overview of Today's Session

1. Translating Business Problems to Statistical Problems
2. Core tasks/goals of Data Mining
3. The process of Data Mining
4. Sampling
5. Variable types
6. Outliers, missing data, normal data
7. An example

Business Problem -> Statistical Problem

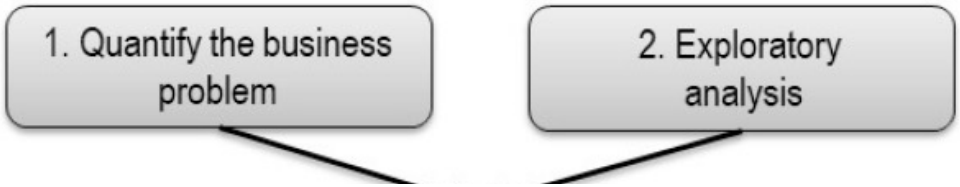
1. Understand & Define the problem
- *Frame the business problem*
- *Prepare for a decision*

2. Set analytic goals and scope your solution
- *Set objectives and define milestones*
- *Design minimum viable product*
- *Identify target metrics*

3. Plan the analysis
- *Plan your datasets*
- *Plan your methods*

https://ayotomiwasalau.medium.com/translate-business-problems-to-data-science-problems-b1693d8f313a

Quantifying the Business Problem and Exploratory Analysis



Conducted in tandem

- The business problem defines what you want to do
- Exploratory analysis provides constraints on what you can do

A business goal is often defined in an abstract manner with implicit meaning:
"We want to target our best customers."
Clarify and quantify:

- WHAT makes a best customer (lifetime value, purchases, $ or unit, profit?)
- What criteria make them BEST (over $50,000, tenure?)
- What Databases are available (sales, manufacturing, marketing?
- What data is stored in the database (individual sales, reports, costs?)
- Is data available real-time or periodically?
- How is the role currently served? What processes and data?

"Identify customers with a potential annual gross profit of over $25,000"

Pandemic Example

What **data** do we have?

How can it be **converted**?

What can be **predicted**?

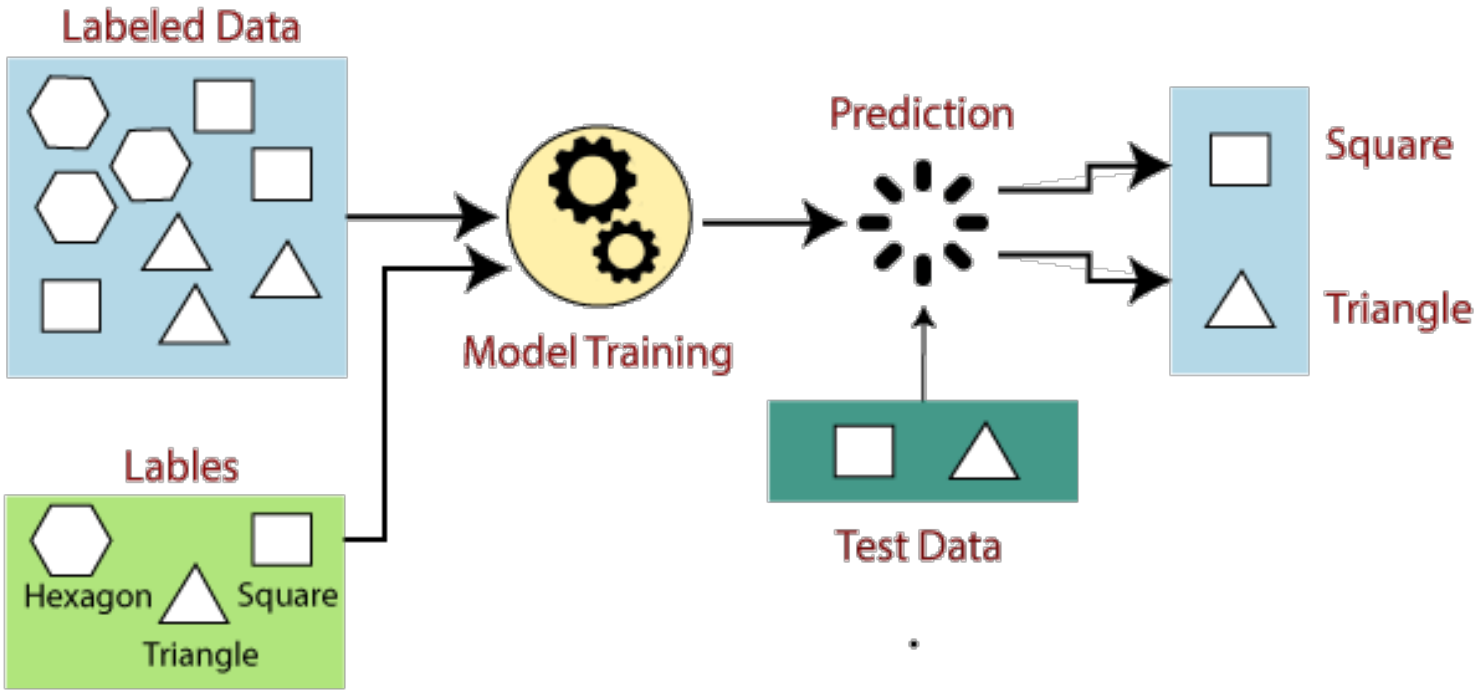
What is the **business value**?

https://youtu.be/TGahNuPH9LY

Vendor serial number	User phone number	Timestamp
111-111-111-111	0851991999	10:45:22-21:05:2021

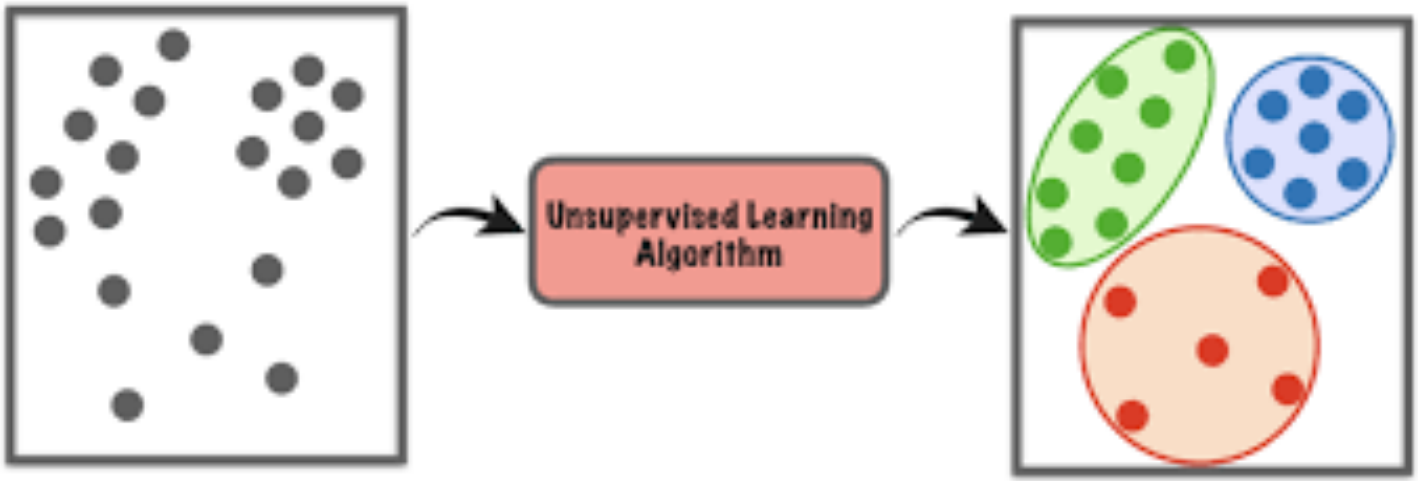
Supervised Learning

- **Task:** Predict a single target or outcome variable
- **Data:** target value is known
- **Goal:** Generate a score for data where value is not known
- **Methods:** Classification and Prediction



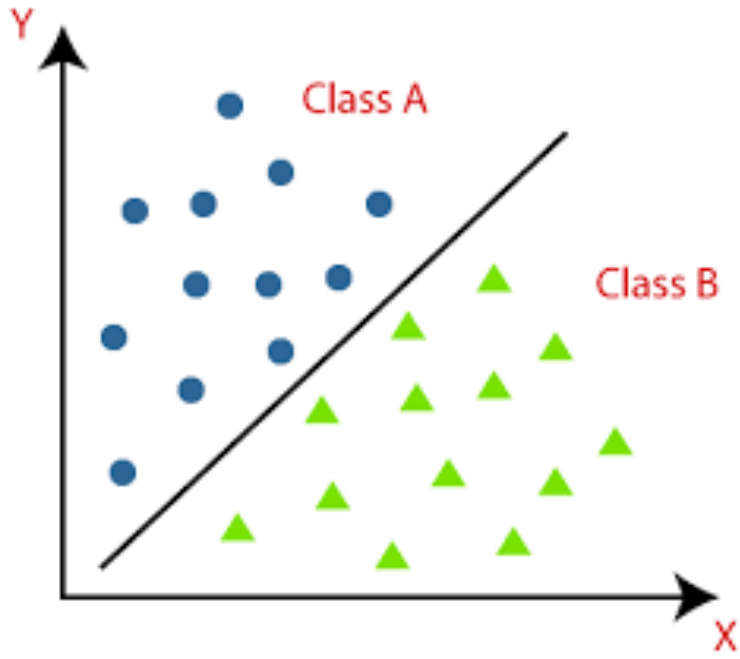
Unsupervised Learning

- **Task:** Segment data into meaningful segments; detect patterns
- **Data**: There is no target (outcome) variable to predict or classify
- **Goal:** Identify which group an obs belongs to
- **Methods:** Association rules, collaborative filters, data reduction & exploration, visualization



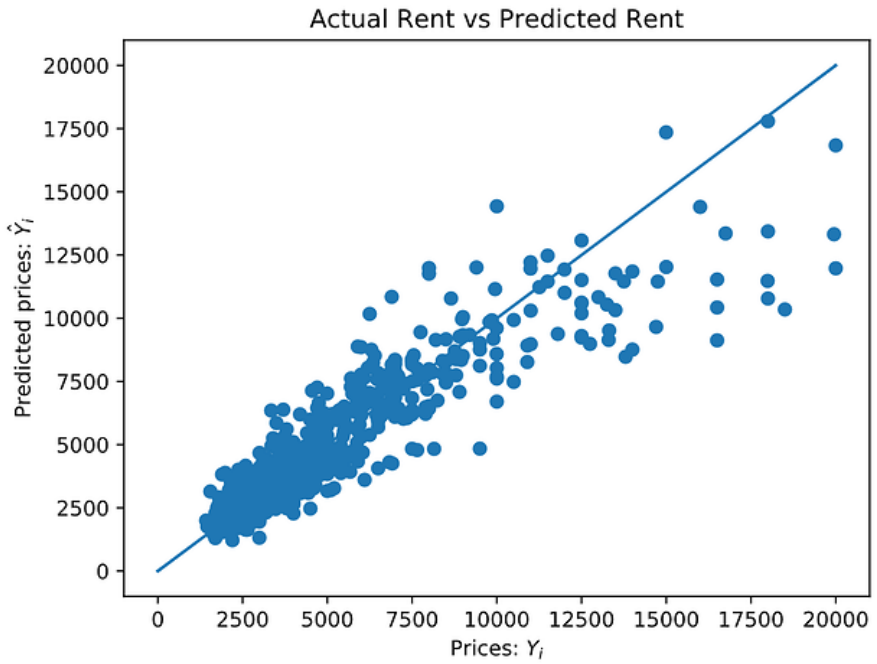
Supervised: Classification

- Goal: Predict categorical **target** (outcome) variable
- Examples: Purchase/no purchase, fraud/no fraud, creditworthy/not creditworthy…
- Each row is a case (customer, tax return, applicant)
- Each column is a variable
- **Target variable** is often binary (yes/no)



Supervised: Prediction

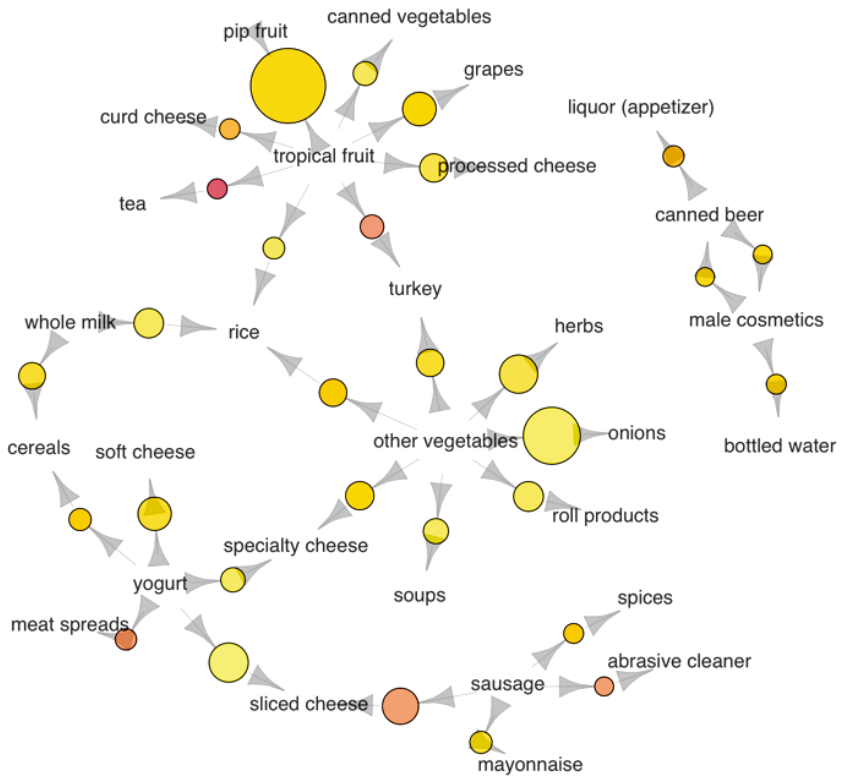
- **Goal:** Predict numerical target (outcome) variable
- Examples: sales, revenue, performance
- Each row is a case (customer, tax return, applicant)
- Each column is a variable



Taken together, classification and prediction constitute “predictive analytics”

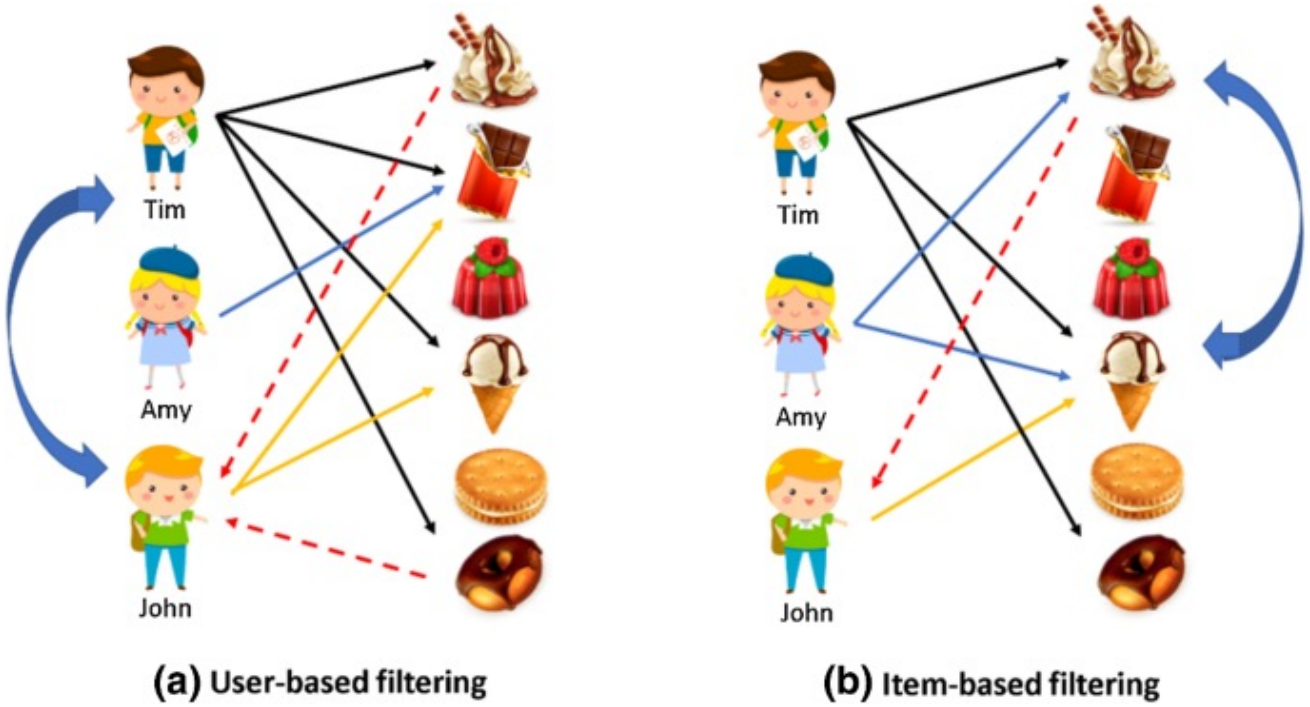
Unsupervised: Association Rules

- **Goal:** define “what goes with what”
- **Example:** “If A was purchased, B was also purchased”
- Rows are transactions
- Used in recommender systems – “Our records show you bought A, you may also like B”
- Also called “affinity analysis”



Unsupervised: Collaborative Filtering

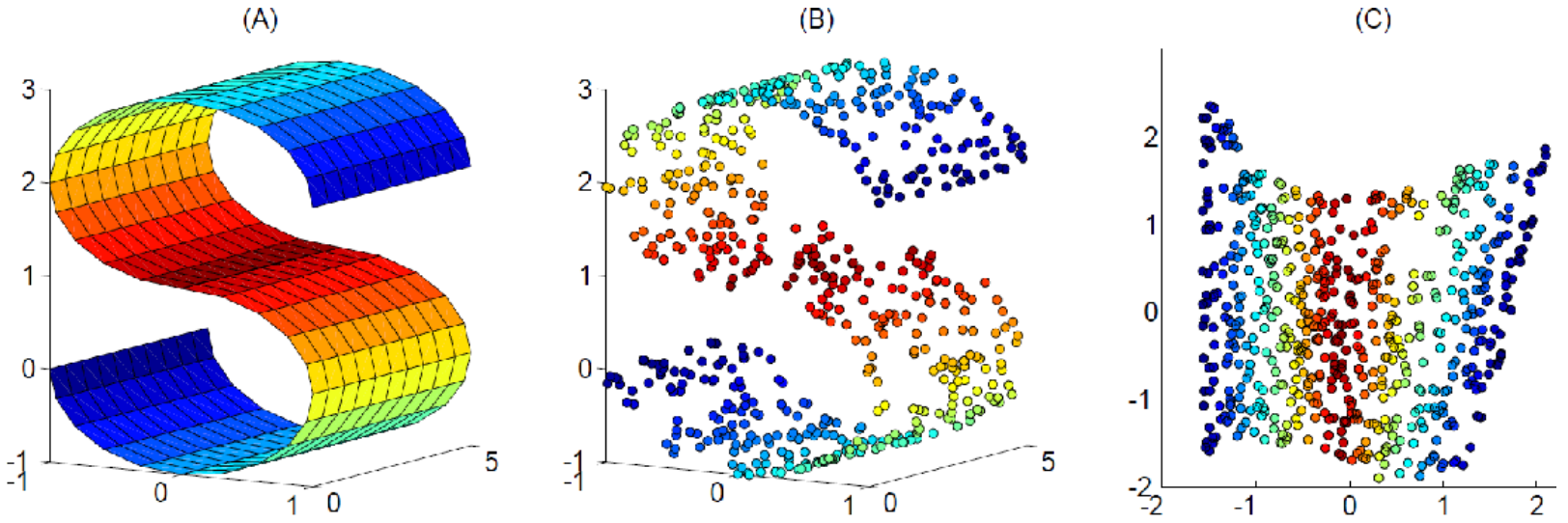
- **Task:** Recommend products to purchase
- **Data:** Based on products that customer rates, selects, views, or purchases
- **Goal:** Recommend products that "customers like you" purchase (user-based); or
- **Goal:** recommend products that share a "product purchaser profile" with your purchases (item-based)



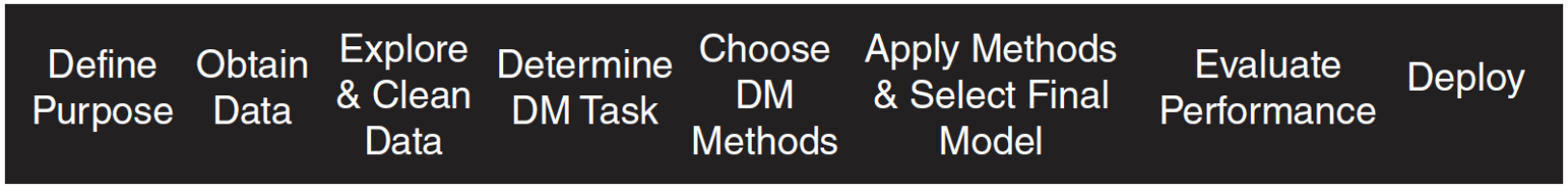
(a) User-based filtering (b) Item-based filtering

Unsupervised: Data Reduction

- Distillation of complex/large data into simpler/smaller data
- Reducing the number of variables/columns (e.g., principal components)
- Reducing the number of records/rows (e.g., clustering)



The Process of Data Mining

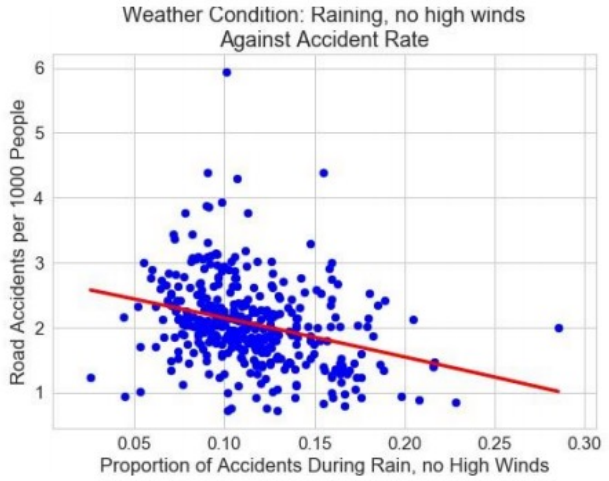


Steps in Data Mining

1. Define/understand purpose
2. Obtain data (may involve random sampling)
3. Explore, clean, pre-process data
4. Reduce the data; if supervised DM, partition it
5. Specify task (classification, clustering, etc.)
6. Choose the techniques (regression, CART, neural networks, etc.)
7. Iterative implementation and “tuning”
8. Assess results – compare models
9. Deploy best model

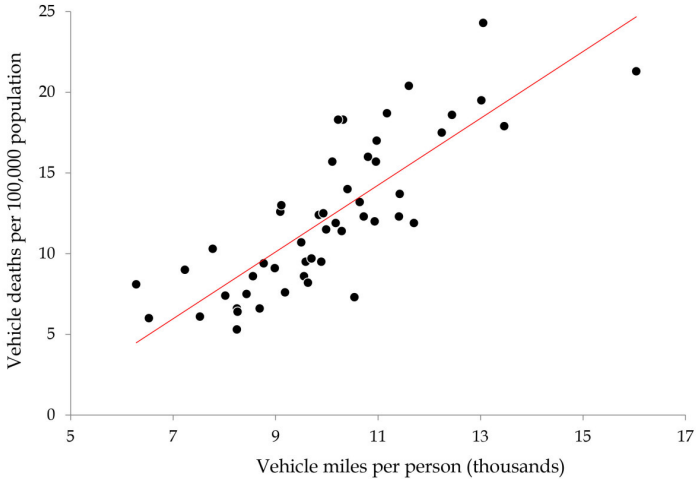
Cross-sectional (Stationary)

	A	B	C	D	E	F	G	H	I	J	K
1	RushHour	WRK_ZONE	WKDY	INT_HWY	LGTCON_day	LEVEL	SPD_LIM	SUR_COND_	TRAF_two_v	WEATHER_a	MAX_SEV
2	1	0	1	1	0	1	70	0	0	1	no-injury
3	1	0	1	0	0	0	55	0	1	0	non-fatal
4	1	0	0	0	0	0	35	0	0	1	no-injury
5	1	0	1	0	0	1	35	0	0	1	no-injury
6	1	0	1	0	0	0	25	0	0	1	non-fatal
7	1	0	1	0	0	0	35	0	0	1	non-fatal
8	1	0	1	0	0	0	60	0	0	0	no-injury
9	1	0	1	0	0	1	45	1	1	0	non-fatal
10	0	0	1	1	0	0	55	1	0	0	no-injury
11	1	0	1	1	0	0	70	1	0	0	non-fatal
12	0	0	1	1	0	0	65	1	0	0	no-injury
13	1	0	1	0	0	0	40	1	0	0	non-fatal
14	1	0	1	0	0	0	45	1	0	0	non-fatal
15	1	0	0	0	0	0	45	1	1	0	non-fatal
16	1	0	1	0	0	0	45	1	1	0	no-injury
17	1	0	1	0	0	0	30	1	1	0	non-fatal
18	1	0	1	0	0	0	55	1	1	0	non-fatal
19	1	0	1	0	0	0	55	1	1	0	no-injury
20	1	0	1	0	0	0	25	1	1	0	no-injury
21	0	0	1	0	0	1	35	0	0	1	no-injury
22	0	0	1	0	0	1	35	0	1	1	no-injury
23	0	0	1	0	0	0	25	0	1	1	no-injury
24	0	0	1	0	0	1	45	0	0	1	no-injury
25	1	0	1	0	0	0	35	0	1	1	no-injury
26	0	0	1	0	0	1	55	0	0	1	non-fatal
27	1	0	1	0	0	1	40	0	0	1	no-injury
28	1	0	0	0	0	1	35	0	1	1	non-fatal
29	0	0	0	0	0	1	25	0	1	0	non-fatal
30	0	0	1	0	0	1	25	0	1	1	no-injury



Does driving cause traffic fatalities?

Miles driven and fatality rate: U.S. states, 2012



single value of dependent variable — slope — single value of independent variable — y-intercept

$$y = mx + b$$

$$Y = \beta_0 + \beta_1 X + \epsilon$$

all observed values for dependent variable — y-intercept a.k.a "bias" — slope a.k.a. "coefficient" — all observed values of independent variable — error*

* additional term

α

Preliminary Exploration in R

loading data, viewing it, summary statistics

code for loading and creating subsets from the data

```
housing.df <- read.csv("WestRoxbury.csv", header = TRUE)  # load data
dim(housing.df)  # find the dimension of data frame
head(housing.df)  # show the first six rows
View(housing.df)  # show all the data in a new tab

# Practice showing different subsets of the data
housing.df[1:10, 1]  # show the first 10 rows of the first column only
housing.df[1:10, ]  # show the first 10 rows of each of the columns
housing.df[5, 1:10]  # show the fifth row of the first 10 columns
housing.df[5, c(1:2, 4, 8:10)]  # show the fifth row of some columns
housing.df[, 1]  # show the whole first column
housing.df$TOTAL_VALUE  # a different way to show the whole first column
housing.df$TOTAL_VALUE[1:10]  # show the first 10 rows of the first column
length(housing.df$TOTAL_VALUE)  # find the length of the first column
mean(housing.df$TOTAL_VALUE)  # find the mean of the first column
summary(housing.df)  # find summary statistics for each column
```

Types of Variables

- Determine the types of pre-processing needed, and algorithms used
- Main distinction: Categorical vs. numeric
- Numeric
  - Continuous
  - Integer
- Categorical
  - Ordered (low, medium, high)
  - Unordered (male, female)

Values: *Quantitative Measurement Scales*

Nominal Scale

- grouping / categorization

Ordinal Scale

- greater-than / less-than comparisons

Interval Scale

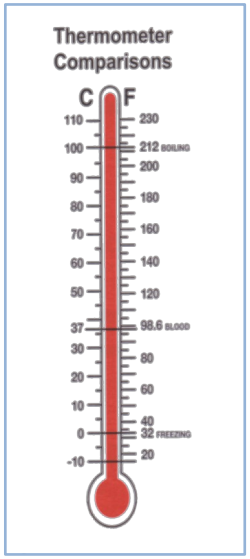
- greater-than / less-than comparisons
- meaningful units
- meaningful distance within scale

Ratio Scale

- greater-than / less-than comparisons
- meaningful units
- meaningful distance within scale
- absolute zero
- meaningful multiples

Men's Singles Rankings

RK	PLAYER	COUNTRY	MOVEMENT	POINTS
1	Roger Federer		0	10105
2	Rafael Nadal		0	9760
3	Marin Cilic		0	4960
4	Grigor Dimitrov		0	4635
5	Alexander Zverev		0	4450
6	Dominic Thiem		0	3810
7	David Goffin		0	3280
8	Kevin Anderson		1	2825
9	Juan Martin del Potro		1	2745
10	Jack Sock		2	2650



source: ESPN.com; thetipsbank.com

Variable handling

- Numeric
  - Most algorithms can handle numeric data
  - May occasionally need to “bin” into categories

- Categorical
  - Naïve Bayes can use as-is
  - In most other algorithms, must create binary dummies (number of dummies = number of categories – 1) [see Table 2.6 for R code]

Creating Binary Dummies – Output

Partial Output

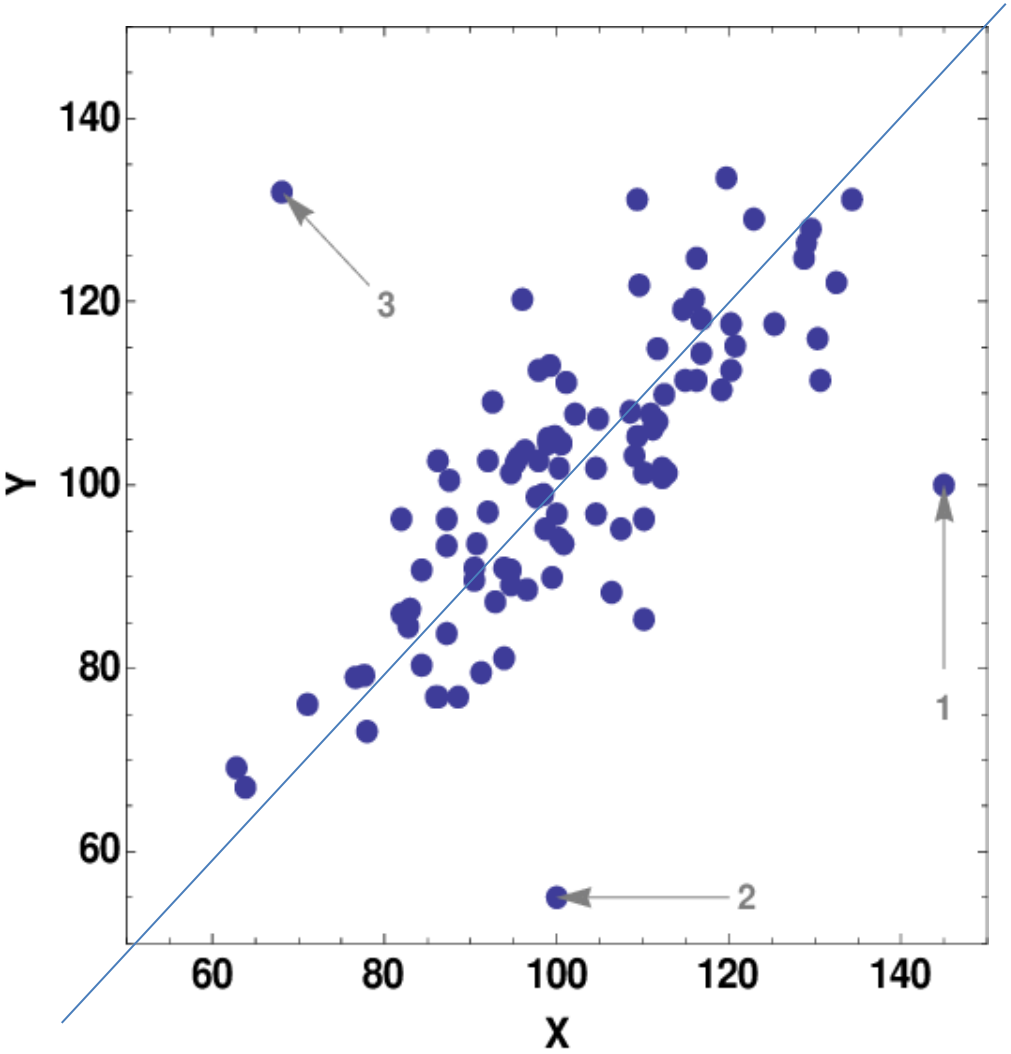
```
> t(t(names(xtotal)))  # Check the names of the dummy variables.
     [,1]
[1,] "BEDROOMS"
[2,] "REMODELNone"
[3,] "REMODELOld"
[4,] "REMODELRecent"

> head(xtotal)
  BEDROOMS REMODELNone REMODELOld REMODELRecent
1        3           1          0             0
2        4           0          0             1
3        4           1          0             0
4        5           1          0             0
5        3           1          0             0
6        3           0          1             0
```

Note: R's lm function automatically creates dummies, so you can skip dummy creation when using lm

Detecting Outliers

- An outlier is an observation that is "extreme", being distant from the rest of the data (definition of "distant" is deliberately vague)
- Outliers can have disproportionate influence on models (a problem if it is spurious)
- An important step in data pre-processing is detecting outliers
- Once detected, domain knowledge is required to determine if it is an error, or truly extreme.



In some contexts, finding outliers is the purpose of the DM exercise (airport security screening). This is called "anomaly detection".

Handling Missing Data

- Most algorithms will not process records with missing values. Default is to drop those records.
- Solution 1: Omission
  - If a small number of records have missing values, can omit them
  - If many records are missing values on a small set of variables, can drop those variables (or use proxies)
  - If many records have missing values, omission is not practical
- Solution 2: Imputation [see Table 2.7 for R code]
  - Replace missing values with reasonable substitutes
  - Let's you keep the record and use the rest of its (non-missing) information

NB: Determine if "missingness" has value!!

Normalizing (Standardizing) Data

- Used in some techniques when variables with the largest scales would dominate and skew results
- Puts all variables on same scale
- Normalizing function: Subtract mean and divide by standard deviation
- Alternative function: scale to 0-1 by subtracting minimum and dividing by the range
  - Useful when the data contain dummies and numeric

$$Z = \frac{x - \mu}{\sigma}$$

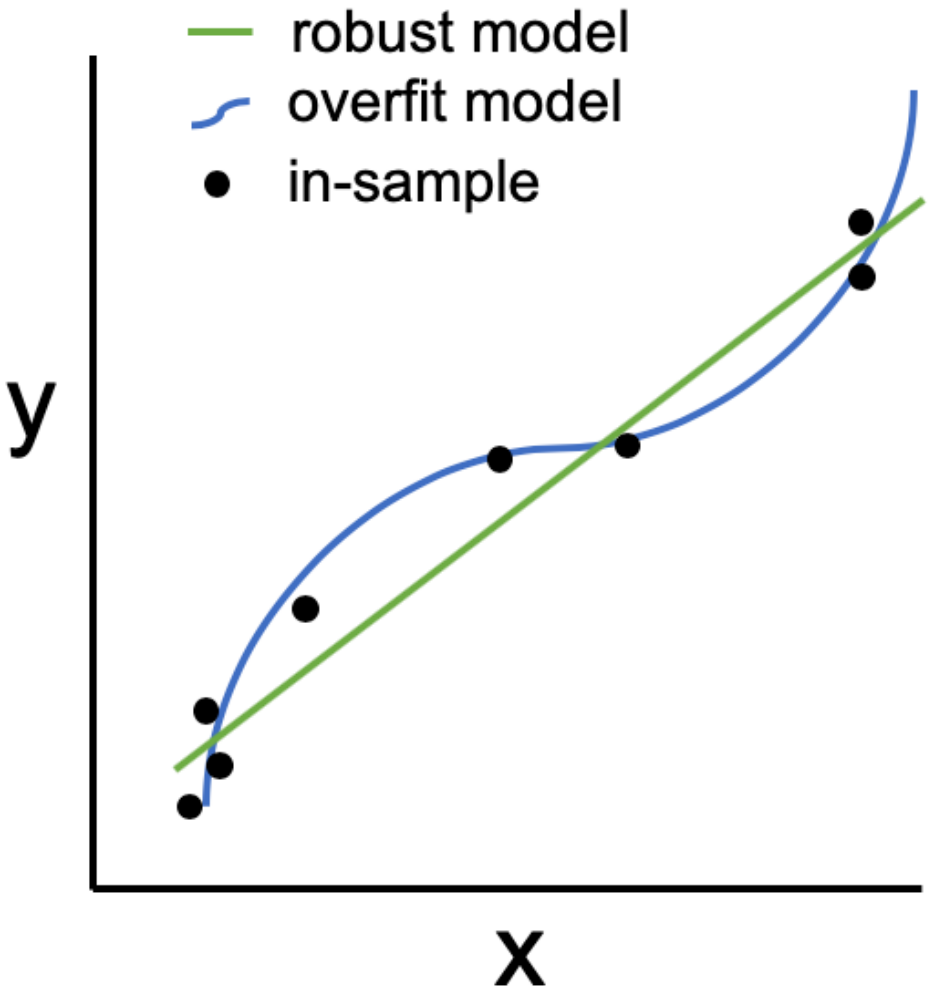
$$z_i = \frac{x_i - min(x)}{max(x) - min(x)}$$

The Problem of Overfitting

- Statistical models can produce highly complex explanations of relationships between variables
- The “fit” may be excellent
- When used with new data, models of great complexity do not do so well.

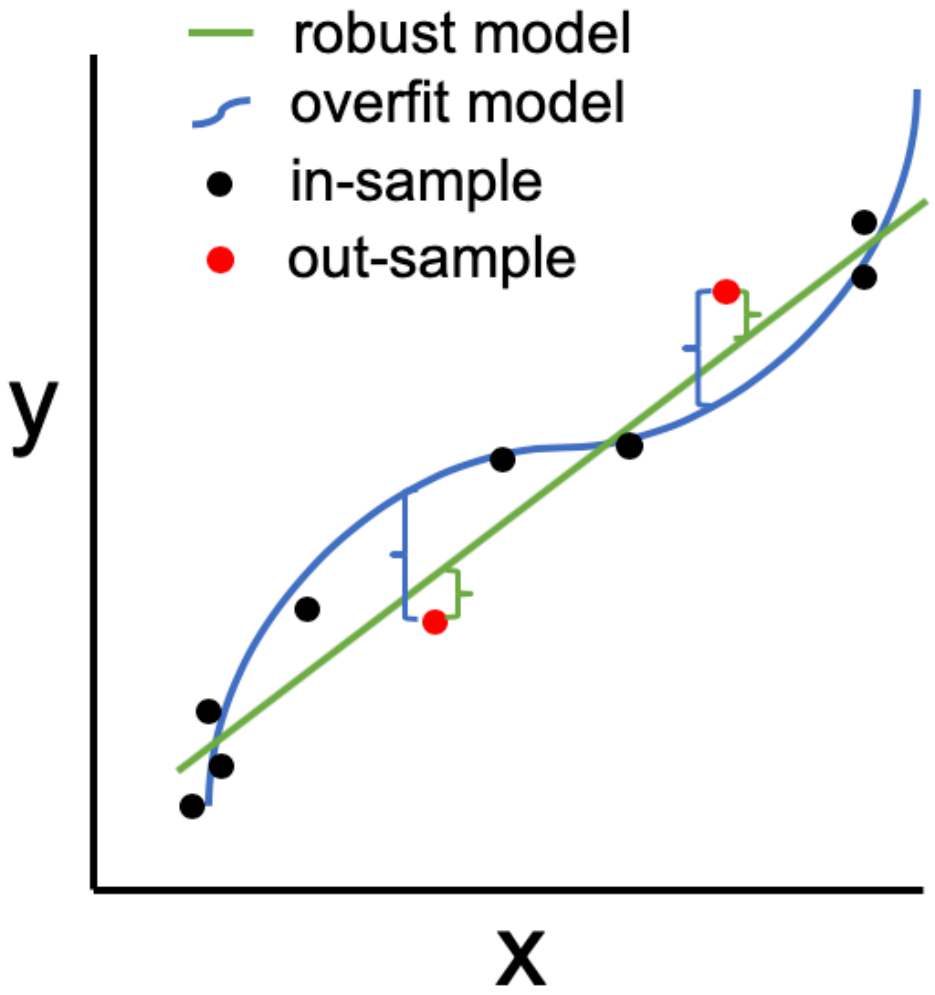
The Problem of Overfitting

100% fit – Excellent!!



The Problem of Overfitting

100% fit – not useful for new data



When used with new data, models of great complexity do not do so well.

Overfitting (cont.)

Causes:

- Too many predictors (too many p, or too few n)
- A model with too many parameters
- Trying many different models

(When p = n, we have perfect fit)

Consequence: Deployed model will not work as well as expected with completely new data.

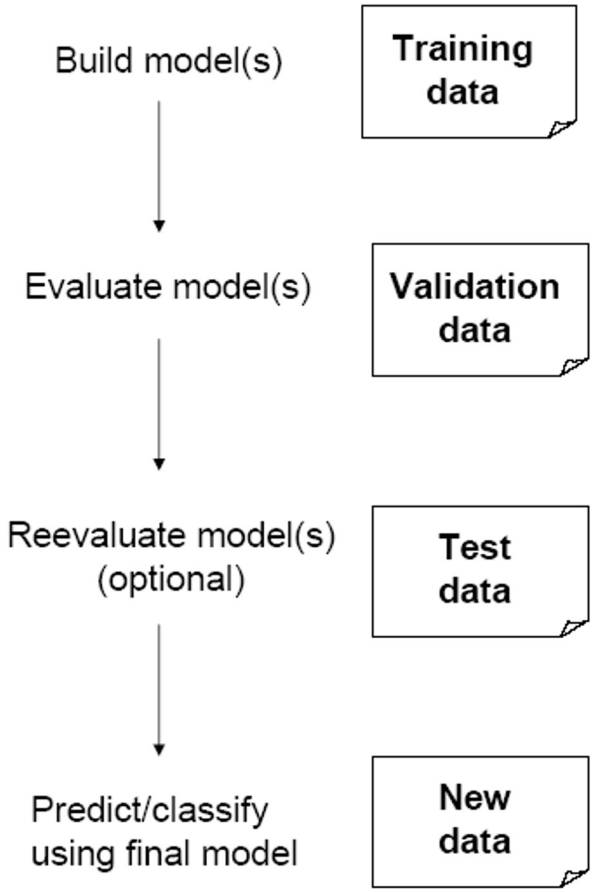
Partitioning the Data

Problem: How well will our model perform with new data?

Solution: Separate data into two parts

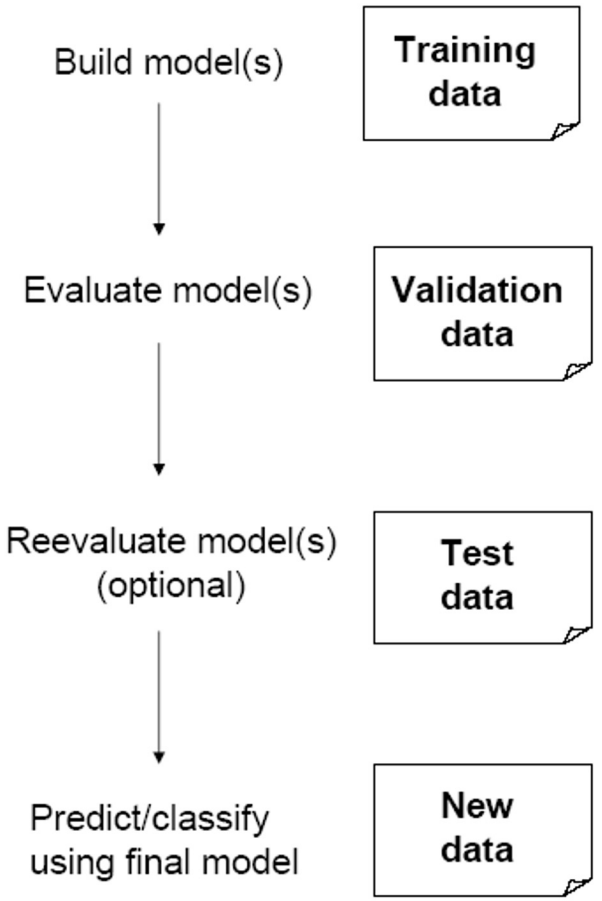
- Training partition to develop the model
- Validation partition to implement the model and evaluate its performance on "new" data

Addresses the issue of overfitting



Test Partition

- When a model is developed on **training data**, it can overfit the training data (hence need to assess on validation)
- Assessing multiple models on same **validation data** can overfit validation data
- Some methods use the validation data to choose a parameter. This too can lead to overfitting the validation data
- Solution: final selected model is applied to a **test partition** to give unbiased estimate of its performance on new data



Error metrics

Error = actual – predicted

ME = Mean error

RMSE = Root-mean-squared error (sd of error)

MSE = mean-squared error (var. of error)

MAE = Mean absolute error

MPE = Pean percentage error

MAPE = Mean absolute percentage error

$$e_i = y_i - \hat{y}_i$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |e_i|$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} e_i^2}{n}}$$

Summary

- Data Mining consists of supervised methods (Classification & Prediction) and unsupervised methods (Association Rules, Data Reduction, Data Exploration & Visualization)
- Before algorithms can be applied, data must be explored and pre-processed
- To evaluate performance and to avoid overfitting, data partitioning is used
- Models are fit to the training partition and assessed on the validation and test partitions
- Data mining methods are usually applied to a sample from a large database, and then the best model is used to score the entire database

Overview of Today's Session

Part 1: Dimension Reduction

1. Simple Approaches
2. Principle Components Analysis

Part 2: Performance Evaluation

1. Variable types
2. Outliers, missing data, normal data
3. An example

Part I
Dimension Reduction

1. Simple strategies for reducing data dimensionality:

- Remove highly correlated predictors
- Using `aggregate` to tabulate counts using multiple variables
- Use functions `melt` and `cast` in `reshape` for pivot tables
- Reducing categories in cat variables by combining categories

2. Principal Components Analysis

Goal: Reduce a set of numerical variables.

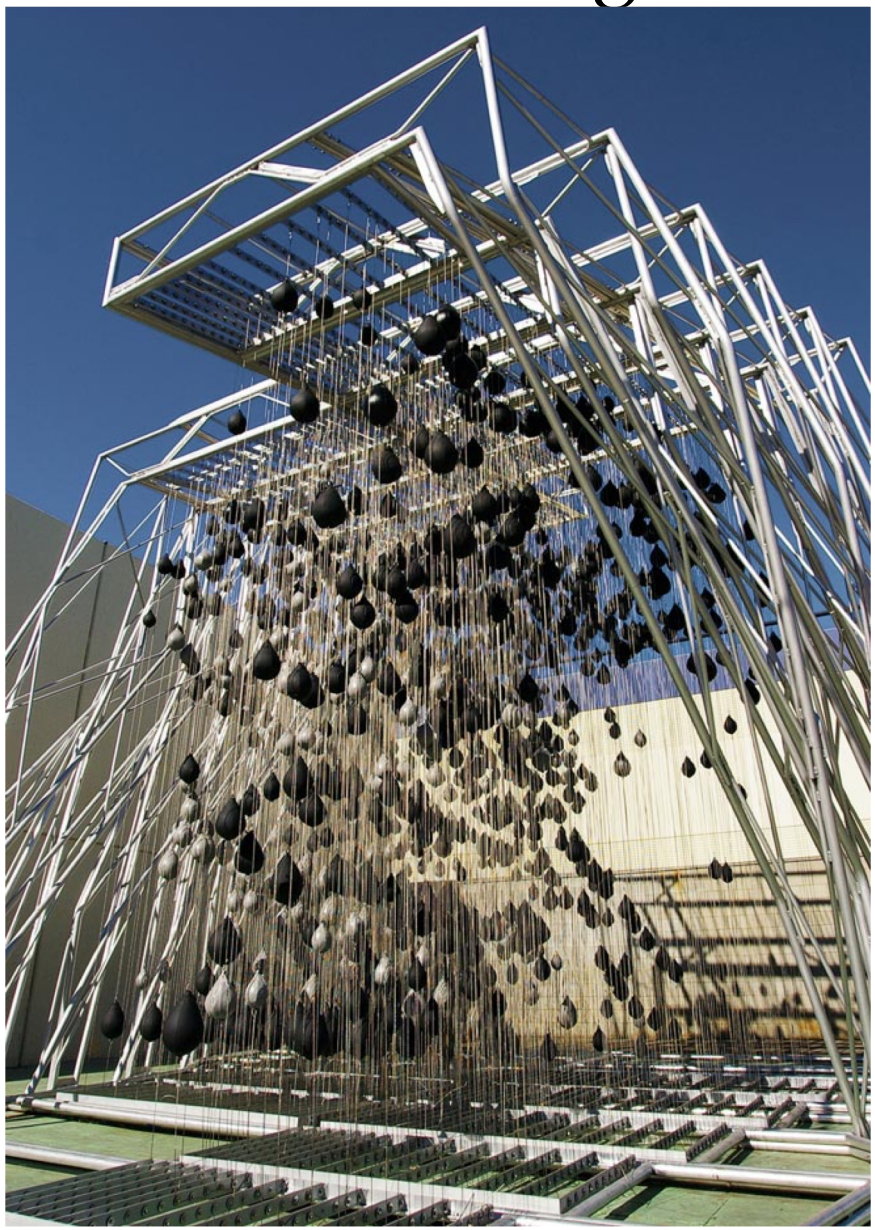
The idea: Remove the overlap of information between these variable. [“Information” is measured by the sum of the variances of the variables.]

Final product: A smaller number of numerical variables that contain most of the information

Principal Components Analysis

- How does PCA do this?
- Create new variables that are linear combinations of the original variables (i.e., they are weighted averages of the original variables).
- These linear combinations are uncorrelated (no information overlap), and only a few of them contain most of the original information.
- The new variables are called *principal components*.

Rotation: Change in



Video: https://www.youtube.com/watch?v=1VcWXwV0vW8

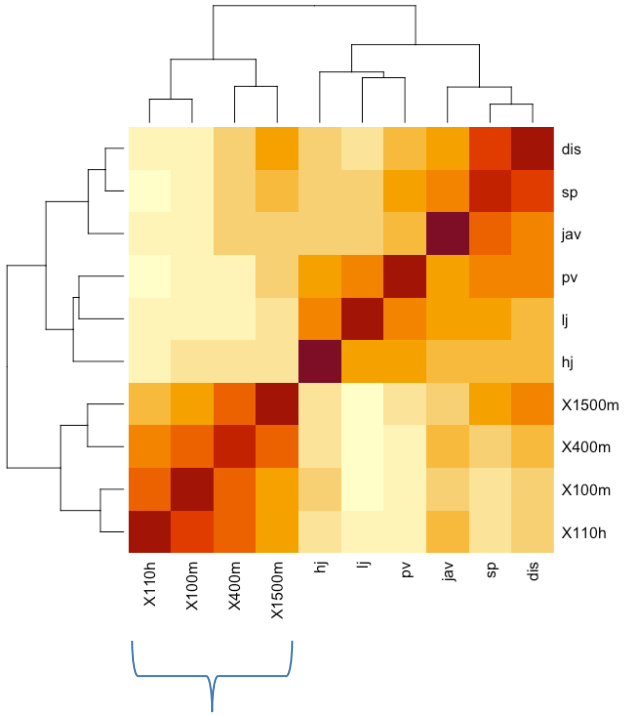
Dimension Reduction: Decathlon

One use of PCA is to reduce the dimensionality of data

Correlates:

```
round(cor(decathlon),2)
```

```
> round(cor(decathlon),2)
       X100m    lj    sp    hj X400m X110h   dis    pv   jav X1500m
X100m   1.00 -0.54 -0.21 -0.15  0.61  0.64 -0.05 -0.39 -0.06   0.26
lj     -0.54  1.00  0.14  0.27 -0.52 -0.48  0.04  0.35  0.18  -0.40
sp     -0.21  0.14  1.00  0.12  0.09 -0.30  0.81  0.48  0.60   0.27
hj     -0.15  0.27  0.12  1.00 -0.09 -0.31  0.15  0.21  0.12  -0.11
X400m   0.61 -0.52  0.09 -0.09  1.00  0.55  0.14 -0.32  0.12   0.59
X110h   0.64 -0.48 -0.30 -0.31  0.55  1.00 -0.11 -0.52 -0.06   0.14
dis    -0.05  0.04  0.81  0.15  0.14 -0.11  1.00  0.34  0.44   0.40
pv     -0.39  0.35  0.48  0.21 -0.32 -0.52  0.34  1.00  0.27  -0.03
jav    -0.06  0.18  0.60  0.12  0.12 -0.06  0.44  0.27  1.00   0.10
X1500m  0.26 -0.40  0.27 -0.11  0.59  0.14  0.40 -0.03  0.10   1.00
>
```



These correlates capture WHAT?

Eigenvalues:

```
decathlon_eigen <- eigen(cor(correlates))

decathlon_eigen$values
  [1] 2.43 0.96 0.35 0.26
```

Eigenvalues: *Variance of each component*

```
sum(decathlon_eigen$values)
  [1] 4
```

Eigenvalues sums up to number of dimensions

```
decathlon_eigen$values / sum(decathlon_eigen$values)
[1] 0.61 0.24 0.09 0.07
```

Ratio of eigenvalue/dimensions is variance captured!

PC1 captures >61% of original data's variance!

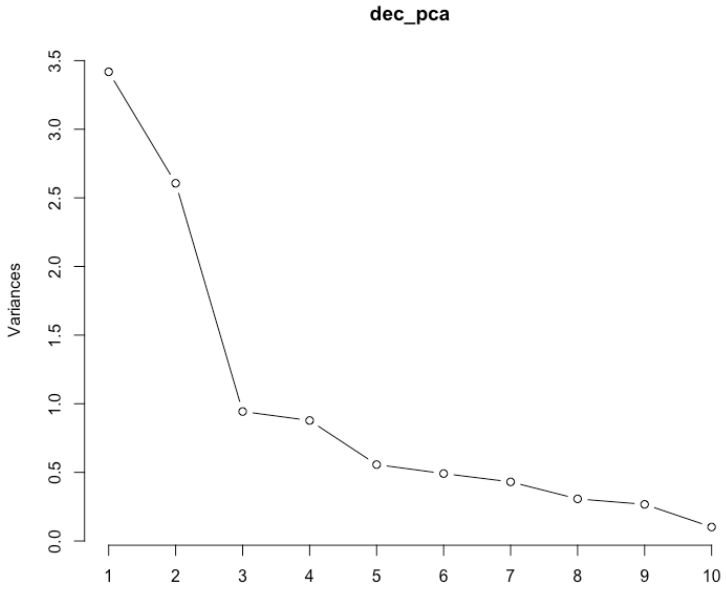
Principal Component Analysis of Decathlon:

```
dec_pca <- prcomp(decathlon, scale. = TRUE)
```

standardize data for PCA

*Using both the '**eigenvalues >1**' and **screeplot** criteria, only three major dimensions seem to exist in these correlates*

```
screeplot(dec_pca, type="lines")
```



Eigenvectors

```
> round(dec_pca$rotation, 2)
         PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   PC9  PC10
X100m  -0.42  0.15 -0.27  0.09 -0.44  0.03  0.25 -0.66  0.11 -0.11
lj      0.39 -0.15 -0.17  0.24  0.37 -0.09  0.75 -0.14 -0.05 -0.06
sp      0.27  0.48  0.10  0.11 -0.01  0.23 -0.11 -0.07 -0.42 -0.65
hj      0.21  0.03 -0.85 -0.39  0.00  0.07 -0.14  0.16  0.10 -0.12
X400m  -0.36  0.35 -0.19 -0.08  0.15 -0.33  0.14  0.15 -0.65  0.34
X110h  -0.43  0.07 -0.13  0.38 -0.09  0.21  0.27  0.64  0.21 -0.26
dis     0.18  0.50  0.05 -0.03  0.02  0.61  0.14 -0.01  0.17  0.53
pv      0.38  0.15  0.14 -0.14 -0.72 -0.35  0.27  0.28  0.02  0.07
jav     0.18  0.37 -0.19  0.60  0.10 -0.44 -0.34 -0.06  0.31  0.13
X1500m -0.17  0.42  0.22 -0.49  0.34 -0.30  0.19 -0.01  0.46 -0.24
```

PC1 is negatively associated with all running events, especially short, high pv and lj (Running??)
PC2 is positively associated with discus, sp, jav (Strength?? Throwing??)

Proportion of variance in data captured by principal components

```
> summary(dec_pca)
Importance of components:
                          PC1    PC2     PC3    PC4     PC5     PC6     PC7     PC8     PC9    PC10
Standard deviation     1.8488 1.6144 0.97123 0.9370 0.74607 0.70088 0.65620 0.55389 0.51667 0.31915
Proportion of Variance 0.3418 0.2606 0.09433 0.0878 0.05566 0.04912 0.04306 0.03068 0.02669 0.01019
Cumulative Proportion  0.3418 0.6025 0.69679 0.7846 0.84026 0.88938 0.93244 0.96312 0.98981 1.00000
```

Interpreting Principal Components: Decathlon Example

Examining the results of PCA

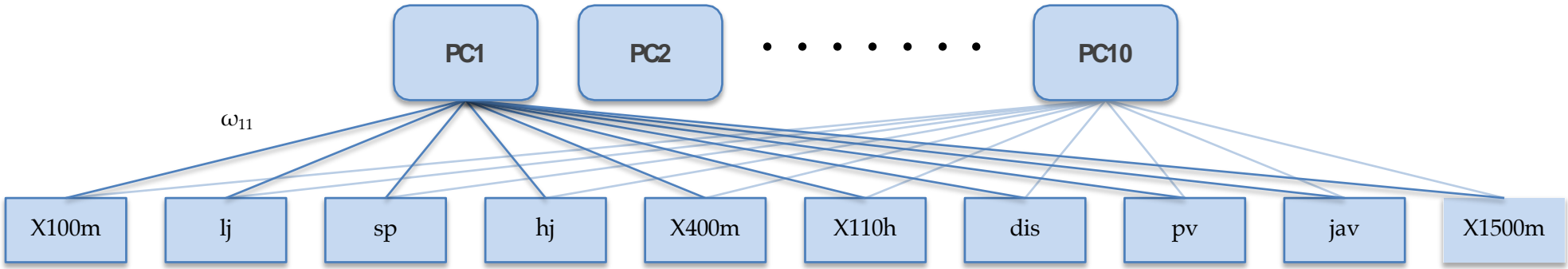
```
dec_pca <- prcomp(dec, scale. = TRUE)
```

```
dec_pca$rotation
          PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   PC9  PC10
X100m   -0.42  0.15 -0.27  0.09 -0.44  0.03  0.25 -0.66  0.11 -0.11
lj       0.39 -0.15 -0.17  0.24  0.37 -0.09  0.75 -0.14 -0.05 -0.06
sp       0.27  0.48  0.10  0.11 -0.01  0.23 -0.11 -0.07 -0.42 -0.65
hj       0.21  0.03 -0.85 -0.39  0.00  0.07 -0.14  0.16  0.10 -0.12
X400m   -0.36  0.35 -0.19 -0.08  0.15 -0.33  0.14  0.15 -0.65  0.34
X110h   -0.43  0.07 -0.13  0.38 -0.09  0.21  0.27  0.64  0.21 -0.26
dis      0.18  0.50  0.05 -0.03  0.02  0.61  0.14 -0.01  0.17  0.53
pv       0.38  0.15  0.14 -0.14 -0.72 -0.35  0.27  0.28  0.02  0.07
jav      0.18  0.37 -0.19  0.60  0.10 -0.44 -0.34 -0.06  0.31  0.13
X1500m  -0.17  0.42  0.22 -0.49  0.34 -0.30  0.19 -0.01  0.46 -0.24
```

ω: ***"weights"*** *are like regression coefficients between PC score and items (but they are still hard to interpret)*

Confirming orthogonality of components

```
round( cor(scores), 2)
     PC1 PC2 PC3 PC4 PC5 PC6 PC7 PC8 PC9 PC10
PC1    1   0   0   0   0   0   0   0   0    0
PC2    0   1   0   0   0   0   0   0   0    0
PC3    0   0   1   0   0   0   0   0   0    0
PC4    0   0   0   1   0   0   0   0   0    0
PC5    0   0   0   0   1   0   0   0   0    0
PC6    0   0   0   0   0   1   0   0   0    0
PC7    0   0   0   0   0   0   1   0   0    0
PC8    0   0   0   0   0   0   0   1   0    0
PC9    0   0   0   0   0   0   0   0   1    0
PC10   0   0   0   0   0   0   0   0   0    1
```



$$PC_i = w_{i1} \cdot X100m + w_{i2} \cdot lj + w_{i3} \cdot sp + w_{i4} \cdot hj + w_{i5} \cdot X400m + w_{i6} \cdot X110h + w_{i7} \cdot dis + w_{i8} \cdot pv + w_{i9} \cdot jav + w_{i10} \cdot X1500m$$

The ***scores*** *of each principal component is a* ***weighted sum*** *of our original dimensions*

PCA in Classification/Prediction

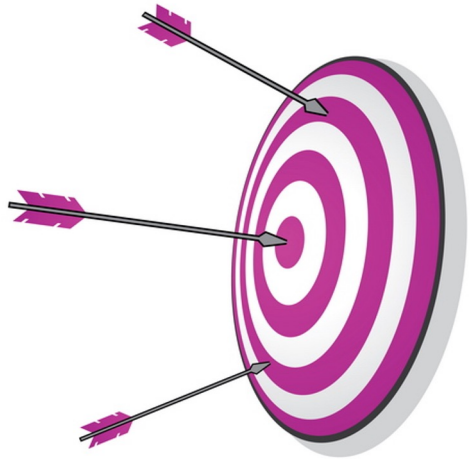
- Apply PCA to training data
- Decide how many PC's to use
- Use variable weights in those PC's with validation/new data
- This creates a new reduced set of predictors in validation/new data

Summary

- **Data summarization** is an important for data exploration
- **Data summaries** include numerical metrics (average, median, etc.) and graphical summaries
- **Data reduction** is useful for compressing the information in the data into a smaller subset
  - Categorical variables can be reduced by combining similar categories
  - Principal components analysis transforms an original set of numerical data into a smaller set of weighted averages of the original data that contain most of the original information in less variables.

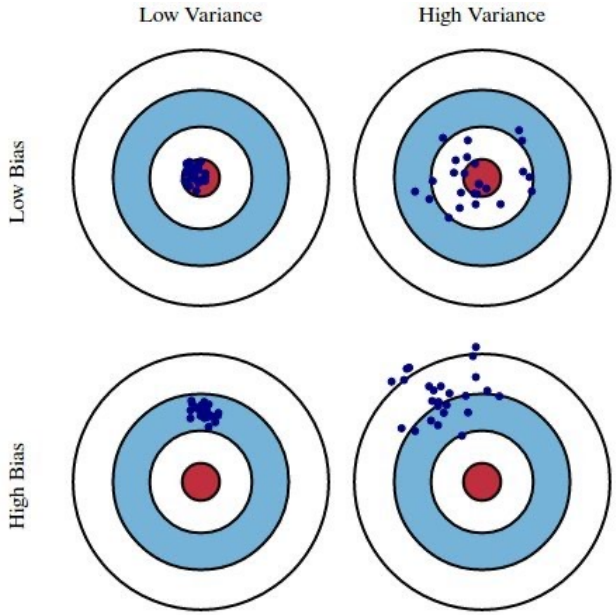
Part II
Performance Evaluation

Why Evaluate?



- Multiple methods are available to classify or predict
- For each method, multiple choices are available for settings
- To choose best model, need to assess each model's performance

Measuring Predictive error



- We want to know how well the model predicts **new data**, not how well it fits the data it was trained with
- Key component of most measures is difference between actual *y* and predicted *y* ("error")

$$e = y - \hat{y}$$

- **MAE or MAD**: Mean absolute error (deviation) Gives an idea of the magnitude of errors

$$\text{MAE} = \frac{\sum |e|}{n}$$

- **Average error** Gives an idea of systematic over- or under-prediction

$$\text{ME} = \frac{\sum e}{n}$$

- **MAPE**: Mean absolute percentage error

$$\text{MAPE} = \frac{100}{n} \sum \left| \frac{e}{y} \right|$$

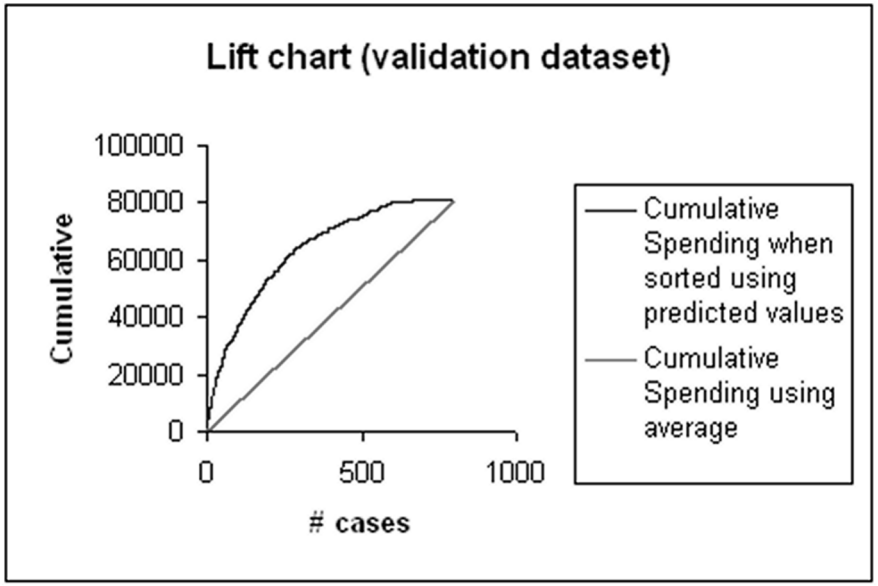
- **RMSE** (root-mean-squared-error): Square the errors, find their average, take the square root

$$\text{RMSE} = \sqrt{\frac{\sum e^2}{n}}$$

Lift Chart for Predictive Error

- Y axis is cumulative value of numeric target variable (e.g., revenue), instead of cumulative count of "responses"
- X axis is cumulative number of cases, sorted left to right in order of predicted value
- Benchmark is average numeric value per record, i.e. not using model

Lift chart example – spending



Misclassification error

- Error = classifying a record as belonging to one class when it belongs to another class.
- Error rate = percent of misclassified records out of the total records in the validation data

Predicted ($\hat{Y}$) \ Actual (Y)	0	1
0	2689	85
1	25	201

$$e = \frac{25+85}{3000} = 3.6\%$$

Naïve Rule

Naïve rule: classify all records as belonging to the most prevalent class

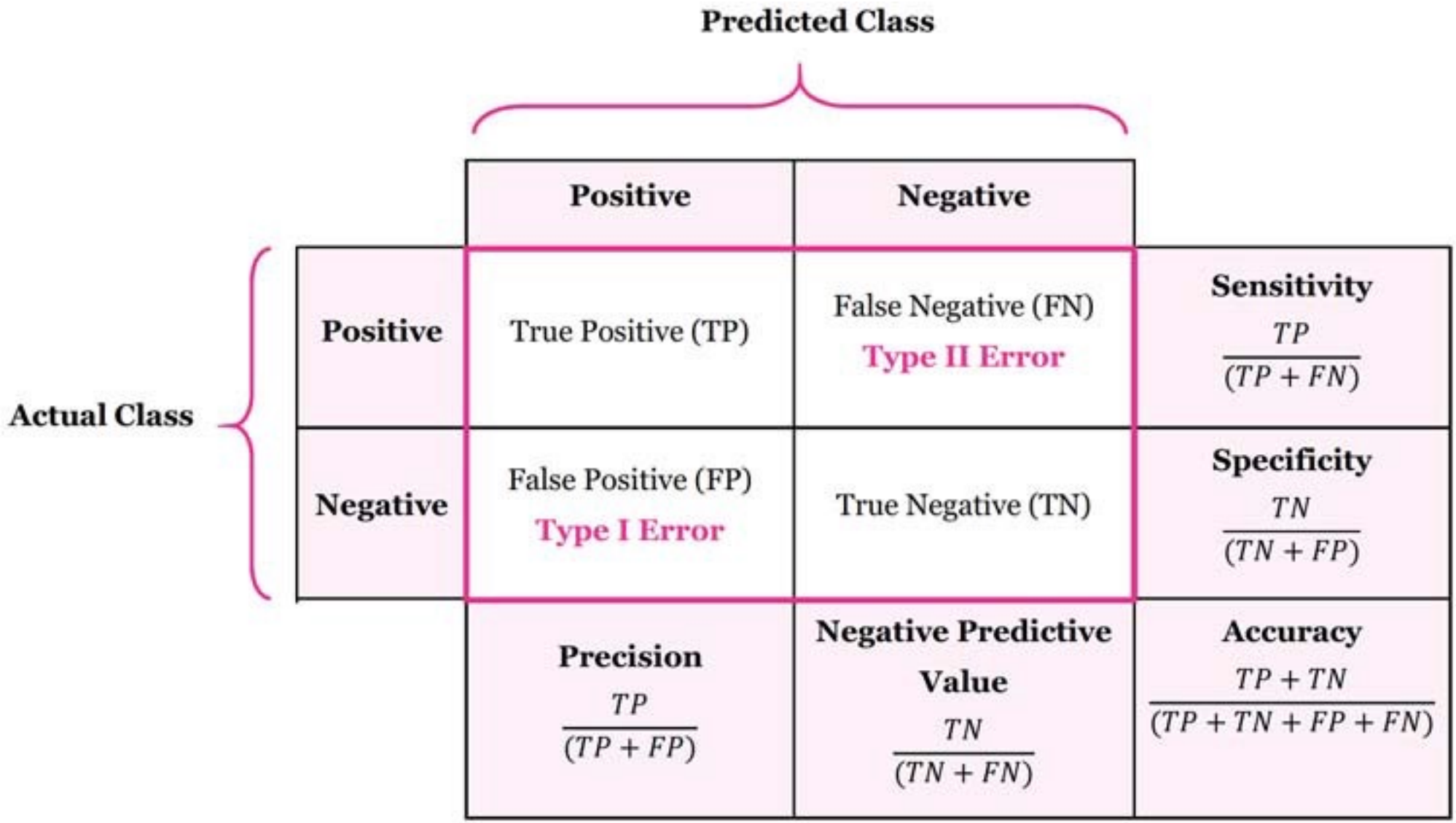
- Often used as benchmark: we hope to do better than that
- Exception: when goal is to identify high-value but rare outcomes, we may do well by doing worse than the naïve rule (see “lift” – later)

Cutoff for classification

- Most DM algorithms classify via a 2-step process:
- For each record,
  - Compute **probability of belonging to class "1"**
  - Compare to cutoff value, and classify accordingly
- Default cutoff value is 0.50
  - If >= 0.50, classify as "1"
  - If < 0.50, classify as "0"
- Can use different cutoff values
- Typically, error rate is lowest for cutoff = 0.50

Cutoff Table

Actual Class	Prob. of "1"	Actual Class	Prob. of "1"
1	0.996	1	0.506
1	0.988	0	0.471
1	0.984	0	0.337
1	0.980	1	0.218
1	0.948	0	0.199
1	0.889	0	0.149
1	0.848	0	0.048
0	0.762	0	0.038
1	0.707	0	0.025
1	0.681	0	0.022
1	0.656	0	0.016
0	0.622	0	0.004


Predicted Class
Positive
Negative
Actual Class
Positive
True Positive (TP)
False Negative (FN)
Type II Error
Sensitivity
$\frac{TP}{(TP+FN)}$
Negative
False Positive (FP)
Type I Error
True Negative (TN)
Specificity
$\frac{TN}{(TN+FP)}$
Precision
$\frac{TP}{(TP+FP)}$
Negative Predictive Value
$\frac{TN}{(TN+FN)}$
Accuracy
$\frac{TP+TN}{(TP+TN+FP+FN)}$

Confusion Matrix for Different Cutoffs

Function `confusionMatrix` requires library `caret`

```
## cutoff = 0.5
> confusionMatrix(ifelse(owner.df$Probability>0.5,
# note: "reference" = "actual"
Confusion Matrix and Statistics

          Reference
Prediction nonowner owner
  nonowner       10     1
  owner           2    11

               Accuracy : 0.875
```

```
## cutoff = 0.25
> confusionMatrix(ifelse(owner.df$Probability>0.25,
Confusion Matrix and Statistics

          Reference
Prediction nonowner owner
  nonowner        8     1
  owner           4    11

               Accuracy : 0.7916667
```

```
## cutoff = 0.75
> confusionMatrix(ifelse(owner.df$Probability>0.75,
Confusion Matrix and Statistics

          Reference
Prediction nonowner owner
  nonowner       11     5
  owner           1     7

               Accuracy : 0.75
```

When One Class is More Important

In many cases it is more important to identify members of one class

- Tax fraud
- Credit default
- Response to promotional offer
- Detecting electronic network intrusion
- Predicting delayed flights

In such cases, we are willing to tolerate greater overall error, in return for better identifying the important class for further attention

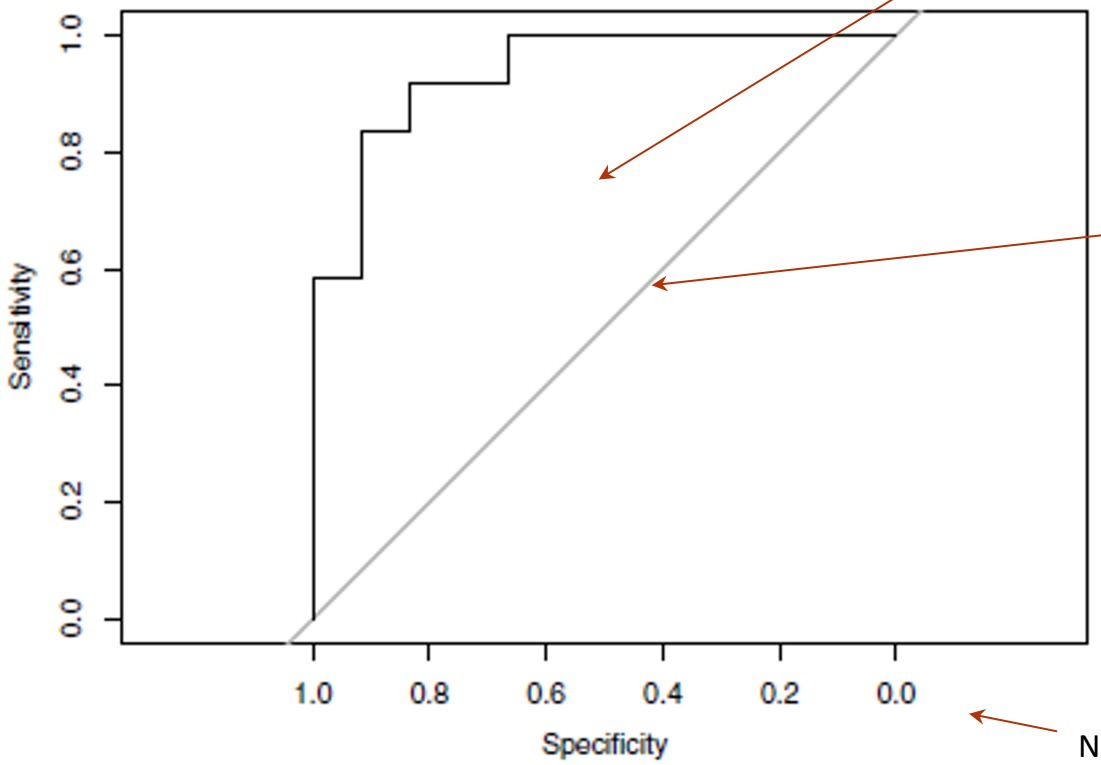
If "C_1" is the important class,

Sensitivity (also called "recall) = % of "C_1" class correctly classified

Specificity = % of "C_0" class correctly classified

Precision= % of predicted "C_1's" that are actually"C_1's"

ROC Curve (library pROC)



Area under the ROC curve ("AUC") is a useful metric

Diagonal reflects random classification on a sliding probability, from 0 to 1, of labeling a record as a "1"

Note that x-axis is in reverse order; alternative is to plot "1-specificity" in ascending order

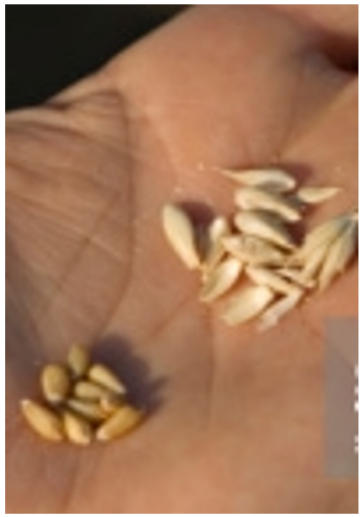
If "C_1" is the important class,

Sensitivity (also called "recall) = % of "C_1" class correctly classified

Specificity = % of "C_0" class correctly classified

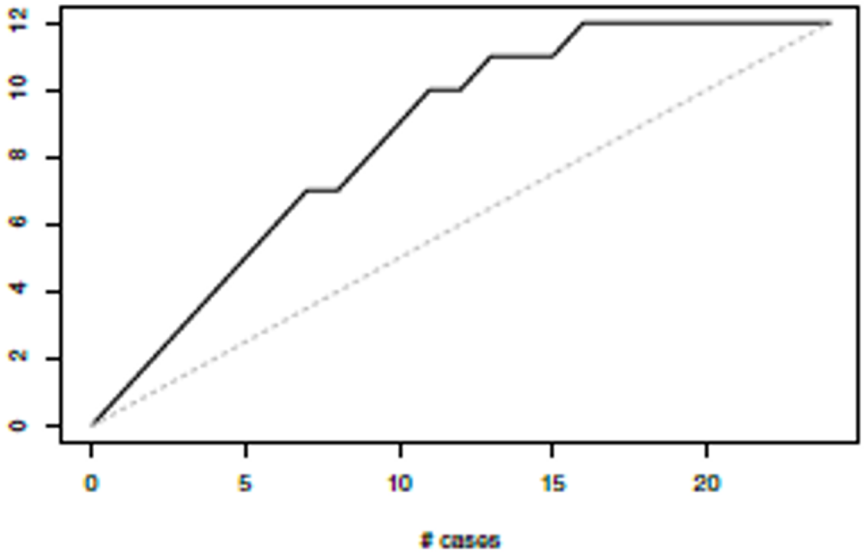
Lift (gains)

(separating the “wheat from the chaff”)



Lift ("gains"): Goal

- Evaluates how well a model identifies the most important class
- Helps evaluate, e.g.,
  - How many tax records to examine
  - How many loans to grant
  - How many customers to mail offer to
- Compare performance of DM model to "no model, pick randomly"
- Measures ability of DM model to identify the important class, relative to the average prevalence of the class
- Charts give explicit assessment of results over a large number of cutoffs



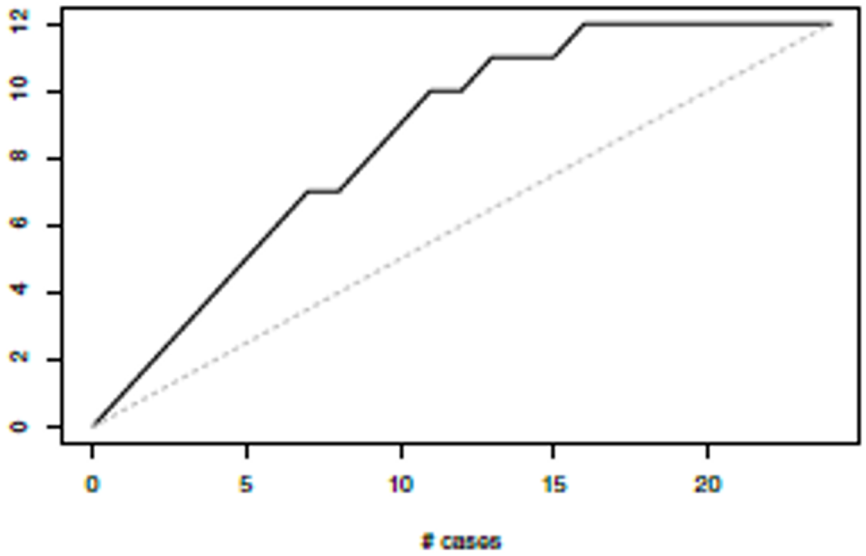
Lift and Decile Charts: How to Use

Sort records by predicted probability of belonging to the important class ("1's")

Move down the list, noting actual class

As you go, compare the number of actual 1's to the number of 1's you would expect with no model

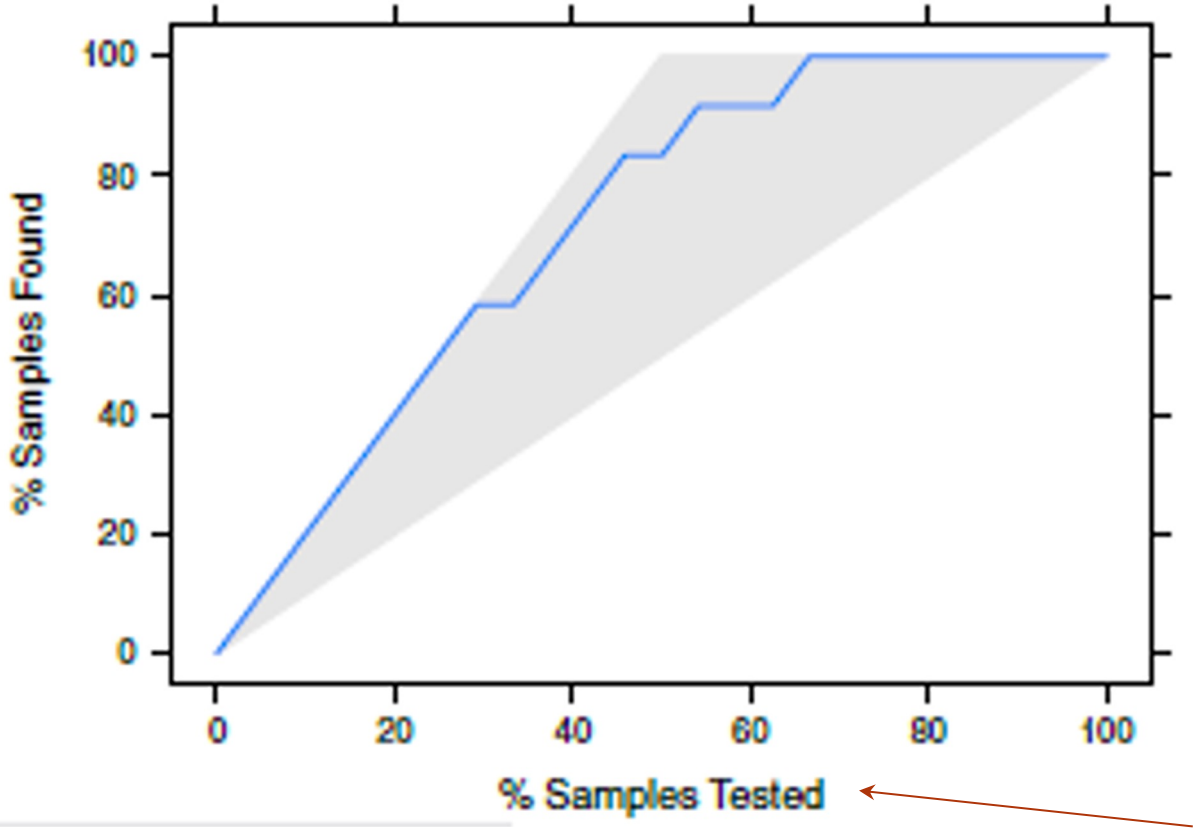
- In lift chart: compare step function to straight line
- In decile chart compare to ratio of 1



After examining (e.g.,) 10 cases (x-axis), 9 owners (y-axis) have been correctly identified

Lift Chart

using %



"Sample" = case

After examining (e.g.,) 40% = 10 of the cases (x-axis), 75% of the owners (y-axis) have been correctly identified

Decile Chart

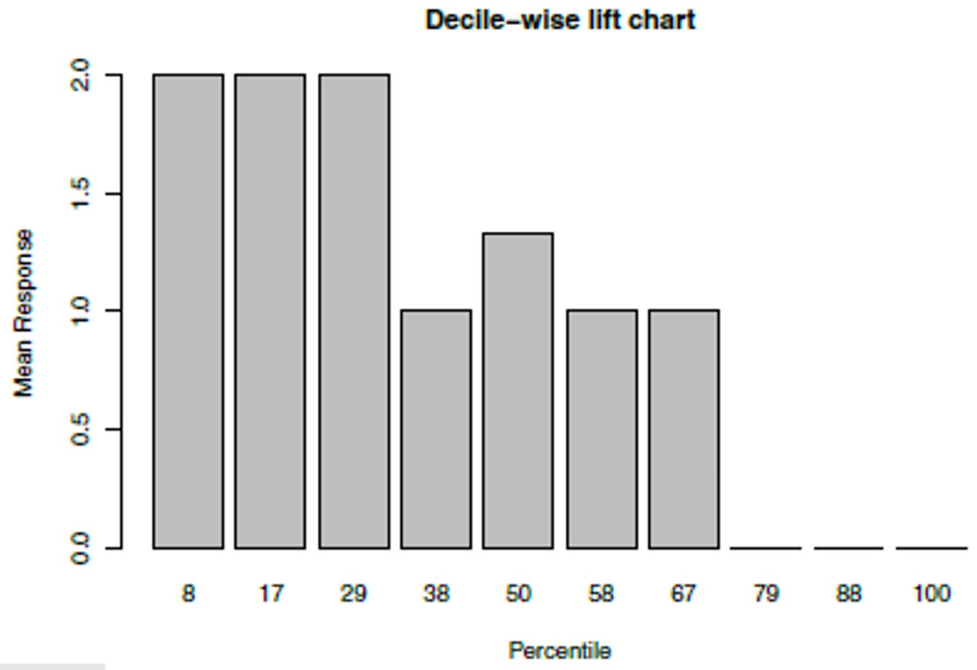


FIGURE 5.7 DECILE LIFT CHART

In “most probable” (top) decile, model is twice as likely to identify the important class compared to avg. prevalence. Percentiles do not match deciles exactly due to small sample of discrete data, with multiple records sharing same decile boundary.

Asymmetric Costs

Misclassification Costs May Differ

Example – Response to Promotional Offer

The cost (benefit) of making a misclassification error may be higher for one class than the other(s)

Suppose we send an offer to 1000 people, with 1% average response rate ("1" = response, "0" = nonresponse)

1. "Naïve rule" (classify everyone as "0")
- error rate of 1% (seems good)

2. Using DM we can correctly classify eight 1's as 1's
- It comes at the cost of misclassifying twenty 0's as 1's and two 1's as 0's.
- Error rate 2.2%

	Actual 0	Actual 1
Predicted 0	970	2
Predicted 1	20	8

Introducing Costs & Benefits

Suppose:

Profit from a "1" is $10

Cost of sending offer is $1

Then:

Under naïve rule, all are classified as "0", so no offers are sent: no cost, no profit

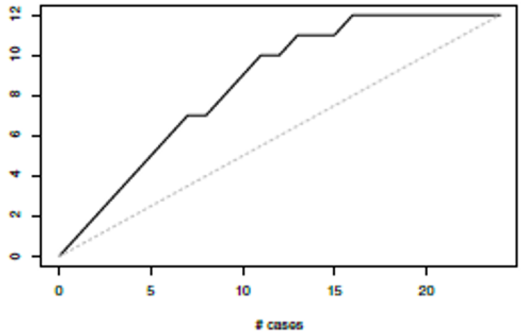
Under DM predictions, 28 offers are sent.

- 8 respond with profit of $10 each
- 20 fail to respond, cost $1 each
- 972 receive nothing (0 cost, 0 profit)

Net profit = $60

Profit Matrix

	Actual 0	Actual 1
Predicted 0	$0	$0
Predicted 1	($20)	$80



Adding costs to the mix, as above, does not change the actual classifications

Better: Use the lift curve and change the cutoff value for "1" to maximize profit

Generalize to Cost Ratio

Sometimes actual costs and benefits are hard to estimate

- Need to express everything in terms of costs (i.e., cost of misclassification per record)
- Goal is to minimize the average cost per record

A good practical substitute for individual costs is the **ratio** of misclassification costs (e,g,, "misclassifying fraudulent firms is 5 times worse than misclassifying solvent firms")

Minimizing Cost Ratio

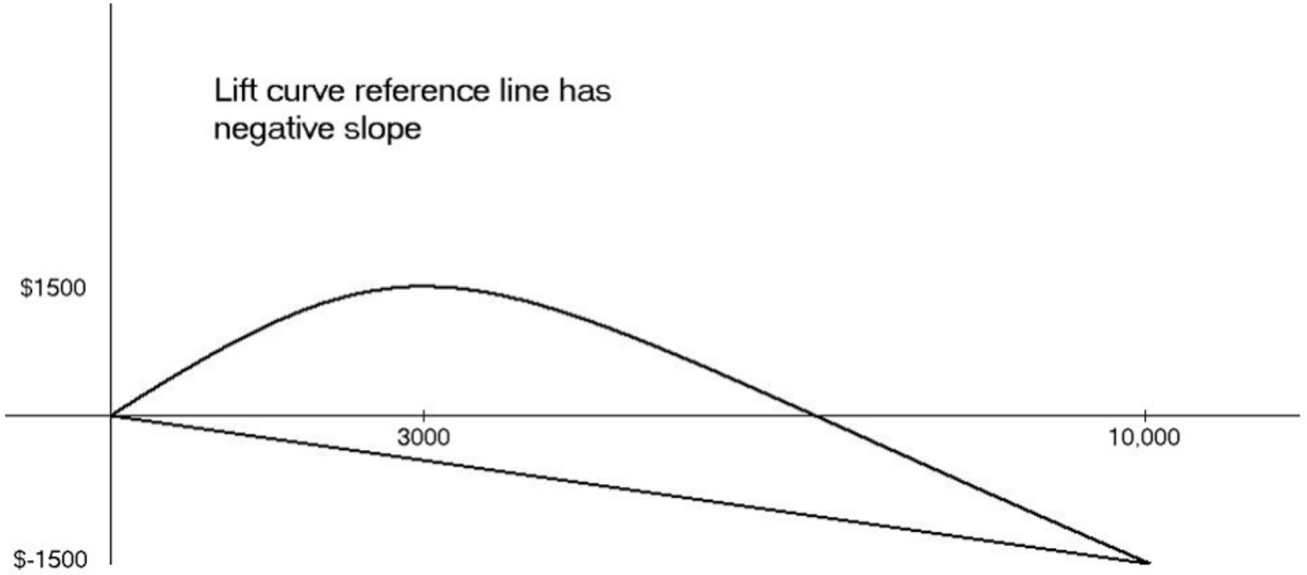
q_1 = cost of misclassifying an actual "1",

q_0 = cost of misclassifying an actual "0"

Minimizing the **cost ratio** q_1/q_0 is identical to minimizing the average cost per record

Adding Cost/Benefit to Lift Curve

- Sort records in descending probability of success
- For each case, record cost/benefit of actual outcome
- Also record cumulative cost/benefit
- Plot all records
  - X-axis is index number (1 for 1st case, n for nth case)
  - Y-axis is cumulative cost/benefit
  - Reference line from origin to y_n (y_n = total net benefit)



Oversampling and Asymmetric Costs

Rare Cases

Asymmetric costs/benefits typically go hand in hand with presence of rare but important class

- Responder to mailing
- Someone who commits fraud
- Debt defaulter

- Often we oversample rare cases to give model more information to work with

- Typically use 50% “1” and 50% “0” for training

Classification Using Triage

Take into account a gray area in making classification decisions

- Instead of classifying as C_1 or C_0, we classify as
  - C_1
  - C_0
  - Can’t say
- The third category might receive special human review

Summary

- Evaluation metrics are important for comparing across DM models, for choosing the right configuration of a specific DM model, and for comparing to the baseline (“no model”)
- Major metrics: confusion matrix, error rate, predictive error
- Other metrics when
  - one class is more important
  - asymmetric costs
- When important class is rare, use oversampling
- In all cases, metrics computed from validation data

HW Suggestions

CREATE well formatted reports

Briefly summarize the question

Format it to distinguish:
question / description / code / output / answers

Show code and relevant text output
use text, not screenshots

Show relevant visualizations
export graphics from Rstudio; not screenshots

CREDIT peers who helped!!

Mention their ID at the top of your assignment!

Peers who help will get extra-credit at end-of-semester