PSS718 - Data Mining

Lecture 3

Asst.Prof.Dr. Burkay Genç

Hacettepe University, IPS, PSS

October 10, 2016

Data is important

Data -> Information -> Knowledge -> Wisdom

Data Nomenclature

Dataset	a collection of data, a.k.a. matrix, table.
Observation	a row of a dataset, a.k.a. entity, row, record, object.
Variable	a column of a dataset, a.k.a. field, column, attribute, characteristic, feature.
Dimension	(of a dataset) is the number of observations and variables

Data Nomenclature

Input Variables measured or preset data items, a.k.a. predictors, covariates, independent variables, observed variables, descriptive variables

Output Variables variables that are "influenced" or "determined" by the input variables, a.k.a. target, response, or dependent variables.

Identifiers variables that uniquely define the observations

Types of Data

Usually data comes in two main types:

Numeric variables Integers or real numbers.

Categoric data a variable that takes its value from a fixed set of values.
Have three sub-types:

- Nominal variables that cannot be ordered, such as eye color. a.k.a. qualitative variables or factors.
- Ordinal variables that can be naturally ordered, such as age group.
- Logical variables that can have only two values, such as true or false, yes or no, on or off.

Note that, some data may be evaluated as categorical or numerical based on the scenario, such as Date and Time data.

Data Partitioning

A dataset can (must) be partitioned into the following:

Training Dataset Used to train the model

Validation Dataset Used to assess the trained model's performance and tune its parameters

Testing Dataset Used to test the trained model

We usually partition based on a 70/15/15 or 40/30/30 ratio.

Summary

Nomenclature

A **dataset** consists of **observations** recorded using **variables**, which consist of a mixture of **input variables** and **output variables**, either of which may be **categoric** or **numeric**.

How it works with R

- Dataset -> dataframe
- Variable -> vector
- Numeric -> numeric, integer
- Categoric -> factor, logical, character

Issues

No real world data is perfect. We need to understand the issues:

- Consistency
- Accuracy
- Completeness
- Interpretability
- Accesibility
- Timeliness

Consistency

- Different people entering data
- Direct conversation with clients
- Interpreting data fields differently
- Different formats for dates
- Different currencies in the same form

Accuracy

- Some data is more accurate: bank transactions
- Some data is less accurate: address info, past events
- When data accuracy is critical, extra resources are employed

Completeness

- Less important data may be omitted
- Some data may be hard to collect

Interpretation

- Understand data thoroughly
- Meanings change by time
- Codes change by time
- Financial values may need to be adjusted

Accessibility

- Which copy of the data do we need? Original vs fixed?
- Complex data access procedures

Timeliness

- Especially important in realtime analysis
- Data may be available in 1-2 days after being collected
- May need to change processes to get timely data

Identifiers

- If two datasets rely on the same unique identifier, this may be really easy
- Other times, we need to match for certain values
  - Names
  - Age
  - Model
  - Make
- Same data may be recorded differently in different forms

Indexing

Given *df*, a dataframe with 100 observations and 10 variables:

- df[40, 5] -> return 5^{th} variable of 40^{th} observation
- df[10:20, 5:8] -> return 5^{th} to 8^{th} variables of 10^{th} to 20^{th} observations
- df[,] -> return everything, same as "df"
- df[3,] -> return all variables of 3^{rd} observation
- df[,5] -> return 5^{th} variable (as a vector)

dim(dataframe)

dim(dataframe)

returns the dimensions of the dataframe (or any other object)

Example

```
> dim(weather)
[1] 366 24
```

Calling by Name

```
> vars <- c("Evaporation", "Sunshine")
> weather[100:105, vars]
```

	Evaporation	Sunshine
100	5.4	5.6
101	4.0	8.9
102	5.8	9.6
103	5.0	10.7
104	6.6	5.9
105	3.2	0.4

Obtaining a Variable

- weather[2] -> returns a dataframe containing only the second variable
- weather[[2]] -> returns a vector of the second variable
- weather$MinTemp -> returns a vector of the MinTemp variable
- weather["MinTemp"] -> returns a dataframe containing only "MinTemp"
- weather[,"MinTemp"] -> returns a vector of "MinTemp"

CSV Data

- Use Rattle's file loader to load your file
- Use R's own csv loader:

```
> crs$dataset <- read.csv("file:.../weather.csv",
                          na.strings=c(".", "NA", "", "?"),
                          strip.white=TRUE)
```

- Also loads directly from the web

Example

```
> ds <- read.csv("http://rattle.togaware.com/weather.csv")
```

Parameters

na.strings	is used to replace certain strings with NA values
strip.white	is used to remove extra whitespace characters
sep	is used to declare the separator character
header	is used to declare whether there is a header row or not

ARFF Data

- Use Rattle
- Use read.arff()

ODBC Data

ODBC

the (O)pen (D)ata(B)ase (C)onnectivity

- Standard for connecting to databases and data warehouses.
- Based on SQL (Structured Query Language)
- Rattle can connect to DBs using ODBC
- Alternatively use R

Example

```
> library(RODBC)
> channel <- odbcConnect("myDWH", uid="kayon", pwd="toga")
```

SPSS

Example

```
> library(foreign)
> mydataset <- read.spss(file="mydataset.sav")
```

Clipboard

	A	B	C	D	E	F	G
1	Date	Expense	Total				
2	17-Nov-2005	19.5	19.5				
3	23-Nov-2005	-15	4.5				
4	10-Dec-2005	30	34.5				
5	23-Jan-2006	-110	-75.5				
6	28-Jan-2006	-20	-95.5				
7	14-Feb-2006	-10	-105.5				
8	14-Feb-2006	300	194.5				
9	26-Feb-2006	220.41	414.91				
10	03-Mar-2006	-20	394.91				
11	14-Jul-2006	50	444.91				
12	17-Jul-2006	-5	439.91				
13	08-Sep-2006	-120	319.91				
14	08-Sep-2006	-130	189.91				
15	22-Oct-2006	55	244.91				
16	23-Nov-2006	135	379.91				
17	11-Jan-2007	-90	289.91				
18	22-Jan-2007	-20	269.91				
19							
20							

Example

```
> expenses <- read.table(file("clipboard"), header=TRUE)
```

Date Type

```
> expenses$Date <- as.Date(expenses$Date,
                           format="%d-%b-%Y")
> head(expenses)
        Date Expense  Total
1 2005-11-17    19.5   19.5
2 2005-11-23   -15.0    4.5
3 2005-12-10    30.0   34.5
4 2006-01-23  -110.0  -75.5
5 2006-01-28   -20.0  -95.5
6 2006-02-14   -10.0 -105.5
```