DATA SCIENCE WITH R

DATA SCIENCE WITH R

Class 4 – Data Manipulation in R

Topic 1

Manipulating Data Using Base R

© Jigsaw Academy Education Pvt Ltd

INDEX

© Jigsaw Academy Education Pvt Ltd

Data Manipulation: Base R

© Jigsaw Academy Education Pvt Ltd

Data Manipulation: Base R

- Sub-setting data
- Selecting specified columns
- Adding new columns
- Reordering data (Ascending/Descending order)
- Group wise operations
- Producing contingency tables

© Jigsaw Academy Education Pvt Ltd

Sub-setting data

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R (Sub setting)

- Sub setting: Selecting a sub set of rows across all columns

```
> head(oj[oj$brand=='tropicana',])
  store     brand week  logmove feat price      AGE60      EDUC     ETHNIC   INCOME   HHLARGE   WORKWOM
1     2 tropicana   40 9.018695    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
2     2 tropicana   46 8.723231    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
3     2 tropicana   47 8.253228    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
4     2 tropicana   48 8.987197    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
5     2 tropicana   50 9.093357    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
6     2 tropicana   51 8.877382    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
    HVAL150 SSTRDIST  SSTRVOL CPDIST5   CPWVOL5
1 0.4638871 2.110122 1.142857 1.92728 0.3769266
2 0.4638871 2.110122 1.142857 1.92728 0.3769266
3 0.4638871 2.110122 1.142857 1.92728 0.3769266
4 0.4638871 2.110122 1.142857 1.92728 0.3769266
5 0.4638871 2.110122 1.142857 1.92728 0.3769266
6 0.4638871 2.110122 1.142857 1.92728 0.3769266
```

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R (Sub setting)

- Can use multiple conditions, | (or), & (and) operator

```
> head(oj[oj$brand=='tropicana'|oj$brand=='dominicks',])
  store     brand week  logmove feat price     AGE60      EDUC    ETHNIC   INCOME   HHLARGE
1     2 tropicana   40 9.018695    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
2     2 tropicana   46 8.723231    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
3     2 tropicana   47 8.253228    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
4     2 tropicana   48 8.987197    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
5     2 tropicana   50 9.093357    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
6     2 tropicana   51 8.877382    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
    WORKWOM   HVAL150 SSTRDIST  SSTRVOL CPDIST5   CPWVOL5
1 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
2 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
3 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
4 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
5 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
6 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
> dim(oj[oj$brand=='tropicana'|oj$brand=='dominicks',])
[1] 19298    17
```

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R (Sub setting)

```
> dim(oj[oj$brand=='tropicana' & oj$feat==0,])
[1] 8045   17
> head(oj[oj$brand=='tropicana' & oj$feat==0,])
  store     brand week  logmove feat price     AGE60      EDUC    ETHNIC   INCOME    HHLARGE
1     2 tropicana   40 9.018695    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
2     2 tropicana   46 8.723231    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
3     2 tropicana   47 8.253228    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
4     2 tropicana   48 8.987197    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
5     2 tropicana   50 9.093357    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
6     2 tropicana   51 8.877382    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
    WORKWOM   HVAL150 SSTRDIST  SSTRVOL CPDIST5   CPWVOL5
1 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
2 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
3 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
4 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
5 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
6 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
>
```

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R (Sub setting)

- So, far logical sub-setting is discussed.
- Use which() operator to get the index for specific rows

```
> index<-which(oj$brand=="dominicks")
> head(index)
[1] 221 222 223 224 225 226
> head(oj[index,])
    store     brand week   logmove feat price     AGE60      EDUC     ETHNIC   INCOME   HHLARGE   WORKWOM   HVAL150
221     2 dominicks   40  9.264829    1  1.59 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853 0.4638871
222     2 dominicks   46  8.987197    0  2.69 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853 0.4638871
223     2 dominicks   47  8.831712    1  2.09 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853 0.4638871
224     2 dominicks   48  7.965546    0  2.09 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853 0.4638871
225     2 dominicks   50  7.377759    0  2.09 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853 0.4638871
226     2 dominicks   51 10.140297    1  1.89 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853 0.4638871
    SSTRDIST  SSTRVOL CPDIST5   CPWVOL5
221 2.110122 1.142857 1.92728 0.3769266
222 2.110122 1.142857 1.92728 0.3769266
223 2.110122 1.142857 1.92728 0.3769266
224 2.110122 1.142857 1.92728 0.3769266
225 2.110122 1.142857 1.92728 0.3769266
226 2.110122 1.142857 1.92728 0.3769266
>
```

© Jigsaw Academy Education Pvt Ltd

Logical vectors Vs. which

- which() removes NA values in the logical vector
- It only returns the indices where the logical vector is TRUE

```
> #Consider vector sales with missing values
> sales<-c(100,200,NA,300,400,NA,500,600,700,NA,1000,1500,NA,NA)
> #subset data using logical operator
> sales[sales>600]
[1]   NA   NA  700   NA 1000 1500   NA   NA
> #subset data using which
> sales[which(sales>600)]
[1]  700 1000 1500
```

© Jigsaw Academy Education Pvt Ltd

Selecting Columns

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R (Selecting)

- Selecting a specified set of columns

```
> head(oj[,c("week","brand")])
  week     brand
1   40 tropicana
2   46 tropicana
3   47 tropicana
4   48 tropicana
5   50 tropicana
6   51 tropicana
> dim(oj[,c("week","brand")])
[1] 28947     2
>
```

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R

- Selecting + Sub-setting

```
> head(oj[oj$brand=='tropicana' & oj$feat==0,c("week","store")])
  week store
1   40     2
2   46     2
3   47     2
4   48     2
5   50     2
6   51     2
> dim(oj[oj$brand=='tropicana' & oj$feat==0,c("week","store")])
[1] 8045    2
```

© Jigsaw Academy Education Pvt Ltd

Adding new columns

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R

- Adding new columns

```
> oj$logInc<-log(oj$INCOME)
> head(oj)
  store     brand week  logmove feat price     AGE60      EDUC    ETHNIC   INCOME   HHLARGE
1     2 tropicana   40 9.018695    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
2     2 tropicana   46 8.723231    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
3     2 tropicana   47 8.253228    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
4     2 tropicana   48 8.987197    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
5     2 tropicana   50 9.093357    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
6     2 tropicana   51 8.877382    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534
    WORKWOM   HVAL150 SSTRDIST  SSTRVOL CPDIST5   CPWVOL5  logInc
1 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266 2.35643
2 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266 2.35643
3 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266 2.35643
4 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266 2.35643
5 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266 2.35643
6 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266 2.35643
>
```

© Jigsaw Academy Education Pvt Ltd

Ordering data

© Jigsaw Academy Education Pvt Ltd

Ordering

- order() returns the element order that results in a sorted vector

```
> students<-c("John","Tim","Alice","Zeus")
> students
[1] "John"  "Tim"   "Alice" "Zeus"
> order(students)
[1] 3 1 2 4
> students[order(students)]
[1] "Alice" "John"  "Tim"   "Zeus"
```

- Application: Very useful for sorting dataframes

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R

- Ordering data

```
> head(oj[order(oj$week),])
    store       brand week  logmove feat price     AGE60      EDUC     ETHNIC   INCOME    HHLARGE
1       2   tropicana   40 9.018695    0  3.87 0.2328647 0.2489349 0.11427995 10.55321 0.1039534
111     2 minute.maid   40 8.407378    0  3.17 0.2328647 0.2489349 0.11427995 10.55321 0.1039534
221     2   dominicks   40 9.264829    1  1.59 0.2328647 0.2489349 0.11427995 10.55321 0.1039534
331     5   tropicana   40 8.680672    0  3.66 0.1173680 0.3212257 0.05387528 10.92237 0.1030916
447     5 minute.maid   40 8.348538    0  2.99 0.1173680 0.3212257 0.05387528 10.92237 0.1030916
563     5   dominicks   40 7.491088    1  1.59 0.1173680 0.3212257 0.05387528 10.92237 0.1030916
      WORKWOM   HVAL150 SSTRDIST   SSTRVOL  CPDIST5   CPWVOL5
1   0.3035853 0.4638871 2.110122 1.1428571 1.927280 0.3769266
111 0.3035853 0.4638871 2.110122 1.1428571 1.927280 0.3769266
221 0.3035853 0.4638871 2.110122 1.1428571 1.927280 0.3769266
331 0.4105680 0.5358834 3.801998 0.6818182 1.600573 0.7363068
447 0.4105680 0.5358834 3.801998 0.6818182 1.600573 0.7363068
563 0.4105680 0.5358834 3.801998 0.6818182 1.600573 0.7363068
```

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R

- Ordering data

```
> head(oj[order(-oj$week),])
    store       brand week   logmove feat price     AGE60      EDUC      ETHNIC   INCOME   HHLARGE
110     2   tropicana  160  8.669743    0  2.97 0.2328647 0.2489349 0.11427995 10.55321 0.1039534
220     2 minute.maid  160 10.626582    1  2.19 0.2328647 0.2489349 0.11427995 10.55321 0.1039534
330     2   dominicks  160  9.064158    0  1.82 0.2328647 0.2489349 0.11427995 10.55321 0.1039534
446     5   tropicana  160  8.921057    0  2.78 0.1173680 0.3212257 0.05387528 10.92237 0.1030916
562     5 minute.maid  160 10.825840    1  2.19 0.1173680 0.3212257 0.05387528 10.92237 0.1030916
678     5   dominicks  160  8.723231    0  1.85 0.1173680 0.3212257 0.05387528 10.92237 0.1030916
      WORKWOM   HVAL150 SSTRDIST   SSTRVOL  CPDIST5   CPWVOL5
110 0.3035853 0.4638871 2.110122 1.1428571 1.927280 0.3769266
220 0.3035853 0.4638871 2.110122 1.1428571 1.927280 0.3769266
330 0.3035853 0.4638871 2.110122 1.1428571 1.927280 0.3769266
446 0.4105680 0.5358834 3.801998 0.6818182 1.600573 0.7363068
562 0.4105680 0.5358834 3.801998 0.6818182 1.600573 0.7363068
678 0.4105680 0.5358834 3.801998 0.6818182 1.600573 0.7363068
```

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R

- Subsetting data: Using logical subsets and which() statement
- Selecting columns: Using column names at column index
- Adding new columns: Use of $ operator
- Re-ordering data: order()
- Group Wise Summaries
- Producing Contingency tables

© Jigsaw Academy Education Pvt Ltd

GroupWise operations

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R

- GroupWise operations
- tapply(), aggregate()
- What is the mean price of each brand of juice across all stores?

```
> aggregate(oj$price,by=list(oj$brand),mean)
      Group.1        x
1   dominicks 1.735809
2 minute.maid 2.241162
3   tropicana 2.870493
> class(aggregate(oj$price,by=list(oj$brand),mean))
[1] "data.frame"
> tapply(oj$price,oj$brand,mean)
  dominicks minute.maid   tropicana
   1.735809    2.241162    2.870493
> class(tapply(oj$price,oj$brand,mean))
[1] "array"
>
```

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R

- GroupWise operations
- tapply(), aggregate()
- What is the mean income level corresponding to brand of juice across all stores?

```
> aggregate(oj$INCOME,by=list(oj$brand),mean)
      Group.1        x
1   dominicks 10.61673
2 minute.maid 10.61673
3   tropicana 10.61673
> class(aggregate(oj$INCOME,by=list(oj$brand),mean))
[1] "data.frame"
> tapply(oj$INCOME,oj$brand,mean)
  dominicks minute.maid   tropicana
   10.61673    10.61673    10.61673
> class(tapply(oj$INCOME,oj$brand,mean))
[1] "array"
```

© Jigsaw Academy Education Pvt Ltd

Contingency tables

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R

- Category wise counts: Contingency tables

Income	Age	Gender	Location
10,000,000	24	M	Arizona
20,000,000	32	F	California
15,000,000	28	M	Arizona
18,000,000	26	F	California

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R

- Category wise counts: Contingency tables

Counts	California	Arizona
Male	0	2
Female	2	0

Income	California	Arizona
Male	0	10,000,000+15,000,000
Female	20,000,000+18,000,000	

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R

- Category wise counts: Contingency tables
- table(), xtab()
- Number of people who bought different brands categorized by presence of advertising campaigns

```
> table(oj$brand,oj$feat)

                0    1
dominicks    7169 2480
minute.maid  6865 2784
tropicana    8045 1604
```

© Jigsaw Academy Education Pvt Ltd

Manipulating data: Base R

- Category wise counts: Contingency tables
- table(), xtab()
- Total income categorized by brand and presence of advertisements

```
> xtabs(oj$INCOME~oj$brand+oj$feat)
              oj$feat
oj$brand             0        1
  dominicks    76110.24 26330.63
  minute.maid 72887.96 29552.91
  tropicana    85410.46 17030.41
```

© Jigsaw Academy Education Pvt Ltd

DATA SCIENCE WITH R

RECAP

- Sub-setting data
- Selecting specified columns
- Adding new columns
- Reordering data (Ascending/Descending order)
- Group wise operations
- Producing contingency tables

© Jigsaw Academy Education Pvt Ltd

Class 4 – Data Manipulation in R

Topic 2

Using dplyr to Manipulate Data

© Jigsaw Academy Pvt Ltd

INDEX

© Jigsaw Academy Pvt Ltd

Data Manipulation: dplyr

© Jigsaw Academy Pvt Ltd

Manipulating data: dplyr

- dplyr: Whats and Whys
- Sub-setting data using filter()
- Selecting columns using select()
- Adding new columns using mutate()
- Ordering data using arrange()
- Summarizing using summarize() and group_by()
- Using functional pipelines to do more than one manipulation task

© Jigsaw Academy Pvt Ltd

Manipulating data: dplyr

- Base R: Good for Medium sized data sets, Awkward Syntax
- dplyr: Faster and elegant syntax
- dplyr: Dataframes
- install.packages(“dplyr”)
- library(dplyr)

© Jigsaw Academy Pvt Ltd

Sub-setting: filter()

© Jigsaw Academy Pvt Ltd

Manipulating data: dplyr

- Sub-setting the data using filter(), base R equivalents: logical subsets and which()
- Only that portion of data such that brand bought is “tropicana”

```
> library(dplyr)
> head(filter(oj,brand=="tropicana"))
  store     brand week  logmove feat price      AGE60      EDUC     ETHNIC    INCOME    HHLARGE    WORKWOM
1     2 tropicana   40 9.018695    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
2     2 tropicana   46 8.723231    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
3     2 tropicana   47 8.253228    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
4     2 tropicana   48 8.987197    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
5     2 tropicana   50 9.093357    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
6     2 tropicana   51 8.877382    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
    HVAL150 SSTRDIST  SSTRVOL CPDIST5   CPWVOL5
1 0.4638871 2.110122 1.142857 1.92728 0.3769266
2 0.4638871 2.110122 1.142857 1.92728 0.3769266
3 0.4638871 2.110122 1.142857 1.92728 0.3769266
4 0.4638871 2.110122 1.142857 1.92728 0.3769266
5 0.4638871 2.110122 1.142857 1.92728 0.3769266
6 0.4638871 2.110122 1.142857 1.92728 0.3769266
```

© Jigsaw Academy Pvt Ltd

Manipulating data: dplyr

- Sub-setting the data using filter(), base R equivalents: logical subsets and which()
- Only that portion of data such that brand bought is “tropicana” or “dominicks”

```
> head(filter(oj,brand=="tropicana"|brand=="dominicks"))
  store     brand week  logmove feat price     AGE60      EDUC     ETHNIC   INCOME   HHLARGE   WORKWOM   HVAL150 SSTRDIST
1     2 tropicana   40 9.018695    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853 0.4638871 2.110122
2     2 tropicana   46 8.723231    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853 0.4638871 2.110122
3     2 tropicana   47 8.253228    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853 0.4638871 2.110122
4     2 tropicana   48 8.987197    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853 0.4638871 2.110122
5     2 tropicana   50 9.093357    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853 0.4638871 2.110122
6     2 tropicana   51 8.877382    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853 0.4638871 2.110122
   SSTRVOL CPDIST5   CPWVOL5
1 1.142857 1.92728 0.3769266
2 1.142857 1.92728 0.3769266
3 1.142857 1.92728 0.3769266
4 1.142857 1.92728 0.3769266
5 1.142857 1.92728 0.3769266
6 1.142857 1.92728 0.3769266
```

© Jigsaw Academy Pvt Ltd

Selecting Columns: select()

© Jigsaw Academy Pvt Ltd

Manipulating data: dplyr

- Selecting columns from data using select(), base R equivalents: index subsets
- Selecting columns brand and income

```
> head(select(oj,brand,INCOME,feat))
      brand   INCOME feat
1 tropicana 10.55321    0
2 tropicana 10.55321    0
3 tropicana 10.55321    0
4 tropicana 10.55321    0
5 tropicana 10.55321    0
6 tropicana 10.55321    0
>
```

© Jigsaw Academy Pvt Ltd

Manipulating data: dplyr

- Selecting columns from data using select(), base R equivalents: index subsets
- Dropping columns brand and income

```
> head(select(oj,-brand,-INCOME,-feat))
  store week  logmove price     AGE60      EDUC    ETHNIC   HHLARGE   WORKWOM   HVAL150 SSTRDIST  SSTRVOL CPDIST5   CPWVOL5
1     2   40 9.018695  3.87 0.2328647 0.2489349 0.1142799 0.1039534 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
2     2   46 8.723231  3.87 0.2328647 0.2489349 0.1142799 0.1039534 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
3     2   47 8.253228  3.87 0.2328647 0.2489349 0.1142799 0.1039534 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
4     2   48 8.987197  3.87 0.2328647 0.2489349 0.1142799 0.1039534 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
5     2   50 9.093357  3.87 0.2328647 0.2489349 0.1142799 0.1039534 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
6     2   51 8.877382  3.87 0.2328647 0.2489349 0.1142799 0.1039534 0.3035853 0.4638871 2.110122 1.142857 1.92728 0.3769266
```

© Jigsaw Academy Pvt Ltd

Creating New Columns: mutate()

© Jigsaw Academy Pvt Ltd

Manipulating data: dplyr

- Adding columns to data using mutate(),
- Adding a new column, log(income)

```
> dim(oj)
[1] 28947    17
> head(mutate(oj,logIncome=log(INCOME)))#Changes not made in oj but its copy
  store     brand week  logmove feat price     AGE60      EDUC    ETHNIC   INCOME   HHLARGE   WORKWOM
1     2 tropicana   40 9.018695    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
2     2 tropicana   46 8.723231    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
3     2 tropicana   47 8.253228    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
4     2 tropicana   48 8.987197    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
5     2 tropicana   50 9.093357    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
6     2 tropicana   51 8.877382    0  3.87 0.2328647 0.2489349 0.1142799 10.55321 0.1039534 0.3035853
    HVAL150 SSTRDIST  SSTRVOL CPDIST5   CPWVOL5 logIncome
1 0.4638871 2.110122 1.142857 1.92728 0.3769266   2.35643
2 0.4638871 2.110122 1.142857 1.92728 0.3769266   2.35643
3 0.4638871 2.110122 1.142857 1.92728 0.3769266   2.35643
4 0.4638871 2.110122 1.142857 1.92728 0.3769266   2.35643
5 0.4638871 2.110122 1.142857 1.92728 0.3769266   2.35643
6 0.4638871 2.110122 1.142857 1.92728 0.3769266   2.35643
> dim(oj)
[1] 28947    17
```

© Jigsaw Academy Pvt Ltd

Ordering data: arrange()

© Jigsaw Academy Pvt Ltd

Manipulating data: dplyr

- Ordering data using order_by(),
- Order whole data by income in ascending order

```
> head(arrange(oj,INCOME))
  store     brand week  logmove feat price     AGE60      EDUC    ETHNIC   INCOME     HHLARGE   WORKWOM
1    75 tropicana   40 8.971067    0  3.87 0.2076995 0.2195485 0.4159995 9.867083 0.06396471 0.3155833
2    75 tropicana   41 8.392990    0  3.87 0.2076995 0.2195485 0.4159995 9.867083 0.06396471 0.3155833
3    75 tropicana   42 9.018695    0  3.87 0.2076995 0.2195485 0.4159995 9.867083 0.06396471 0.3155833
4    75 tropicana   43 8.624791    0  3.87 0.2076995 0.2195485 0.4159995 9.867083 0.06396471 0.3155833
5    75 tropicana   44 8.476371    0  3.87 0.2076995 0.2195485 0.4159995 9.867083 0.06396471 0.3155833
6    75 tropicana   45 8.877382    0  3.87 0.2076995 0.2195485 0.4159995 9.867083 0.06396471 0.3155833
  HVAL150 SSTRDIST  SSTRVOL  CPDIST5   CPWVOL5
1   0.496 7.192667 2.230769 1.375126 0.7031819
2   0.496 7.192667 2.230769 1.375126 0.7031819
3   0.496 7.192667 2.230769 1.375126 0.7031819
4   0.496 7.192667 2.230769 1.375126 0.7031819
5   0.496 7.192667 2.230769 1.375126 0.7031819
6   0.496 7.192667 2.230769 1.375126 0.7031819
```

© Jigsaw Academy Pvt Ltd

Manipulating data: dplyr

- Ordering data using order_by(),
- Order whole data by income in descending order

```
> head(arrange(oj,-INCOME)
+ )
  store     brand week  logmove feat price     AGE60      EDUC    ETHNIC  INCOME    HHLARGE   WORKWOM
1    62 tropicana   40 9.373819    0  3.87 0.2225343 0.5177603 0.0265109 11.2362 0.1039793 0.3227652
2    62 tropicana   41 9.368369    0  3.87 0.2225343 0.5177603 0.0265109 11.2362 0.1039793 0.3227652
3    62 tropicana   42 9.570529    0  3.87 0.2225343 0.5177603 0.0265109 11.2362 0.1039793 0.3227652
4    62 tropicana   43 9.400630    0  3.87 0.2225343 0.5177603 0.0265109 11.2362 0.1039793 0.3227652
5    62 tropicana   44 9.329367    0  3.87 0.2225343 0.5177603 0.0265109 11.2362 0.1039793 0.3227652
6    62 tropicana   45 9.631154    0  3.87 0.2225343 0.5177603 0.0265109 11.2362 0.1039793 0.3227652
    HVAL150 SSTRDIST   SSTRVOL CPDIST5   CPWVOL5
1 0.9166995 5.452685 0.7058824 2.18405 0.2017224
2 0.9166995 5.452685 0.7058824 2.18405 0.2017224
3 0.9166995 5.452685 0.7058824 2.18405 0.2017224
4 0.9166995 5.452685 0.7058824 2.18405 0.2017224
5 0.9166995 5.452685 0.7058824 2.18405 0.2017224
6 0.9166995 5.452685 0.7058824 2.18405 0.2017224
```

© Jigsaw Academy Pvt Ltd

Summarizing data: summarize() and group_by()

© Jigsaw Academy Pvt Ltd

Manipulating data: dplyr

- Summarizing data using summarize() and group_by()
- group_by() makes grouped table, summarize() can take this grouped table and produce summaries for different columns
- Mean level of income and standard deviation of income for each brand of orange juice

```
> gr_brand<-group_by(oj,brand)
> summarize(gr_brand,mean(INCOME),sd(INCOME))
Source: local data frame [3 x 3]

        brand mean(INCOME) sd(INCOME)
1   dominicks     10.61673  0.2823234
2 minute.maid     10.61673  0.2823234
3   tropicana     10.61673  0.2823234
```

© Jigsaw Academy Pvt Ltd

Functional Pipelines: %>%

© Jigsaw Academy Pvt Ltd

Manipulating data: dplyr

- dplyr becomes a powerful tool when combined with %>% (pipe) operator
- Several data manipulation tasks can be accomplished in just one line of code
- Traditionally functional composition is achieved by using nested function calls
- For example, Find the mean price for all people whose income is >=10.5

```
> #Base R code
> mean(oj[oj$INCOME>=10.5,"price"])
[1] 2.270229
> #dplyr code
> summarize(filter(oj,INCOME>=10.5),mean(price))
  mean(price)
1    2.270229
```

© Jigsaw Academy Pvt Ltd

Manipulating data: dplyr

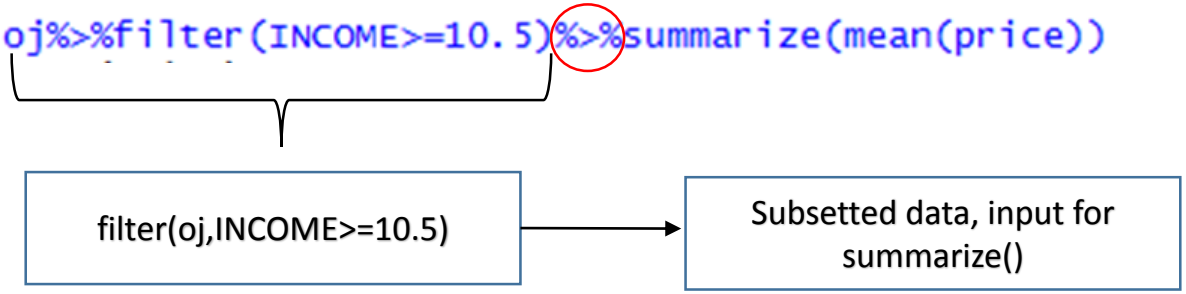
```
> oj%>%filter(INCOME>=10.5)%>%summarize(mean(price))
```

filter(oj,INCOME>=10.5)

© Jigsaw Academy Pvt Ltd

Manipulating data: dplyr

```
> oj%>%filter(INCOME>=10.5)%>%summarize(mean(price))
```



© Jigsaw Academy Pvt Ltd

Manipulating data: dplyr

- Clearly the code looks very messy, using a %>% operator, we can make it more readable

```
> oj%>%filter(INCOME>=10.5)%>%summarize(mean(price))
  mean(price)
1    2.270229
```

- This can be easily read as:
- Take data oj, filter it based on income
- Take this filtered data frame and compute the mean of price

© Jigsaw Academy Pvt Ltd

Manipulating data: dplyr

- Subset the data based on price>=2.5, create a column logIncome, compute the mean, standard deviation and median of column logIncome

```
> oj%>%filter(price>=2.5)%>%mutate(logIncome=log(INCOME))%>%summarize(mean(logIncome),median(logIncome),sd(
logIncome))
  mean(logIncome) median(logIncome) sd(logIncome)
1        2.360997          2.363903    0.02800802
```

© Jigsaw Academy Pvt Ltd

RECAP

- dplyr: better manipulation functionality
- Sub-setting data using filter()
- Selecting columns using select()
- Adding new columns using mutate()
- Ordering data using arrange()
- Summarizing using summarize() and group_by()
- Using functional pipelines to do more than one manipulation task

© Jigsaw Academy Pvt Ltd

Class 4 – Data Manipulation in R

Topic 3

Working with Date Objects

© Jigsaw Academy Pvt Ltd

INDEX

© Jigsaw Academy Pvt Ltd

Manipulating Date objects

© Jigsaw Academy Pvt Ltd

Manipulating date objects

- Dates are treated as a special data type in most programming languages
- R also treats dates as a separate data type
- Doing so, one can easily work with dates
- Usual date operations include:
  - Finding time interval between two points in data: age
  - Extracting months and week days

© Jigsaw Academy Pvt Ltd

Manipulating date objects

- Character to date conversion-Date Class
- Extracting months and weekdays
- Using difftime()
- Manipulating data involving dates
- POSIXct and POSIXlt Classes
- Working with lubridate()

© Jigsaw Academy Pvt Ltd

Manipulating date objects

- Using Date class to convert a character into a Date

```
> fd<-read.csv("Fd.csv")
> str(fd)
'data.frame': 30443 obs. of  25 variables:
 $ FlightDate        : Factor w/ 74 levels "01-Feb-14","01-Jan-14",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ UniqueCarrier     : Factor w/ 13 levels "AA","AS","B6",..: 4 4 4 4 11 11 11 11 11 4 ...
 $ AirlineID         : int  19790 19790 19790 19790 20355 20355 20355 20355 20355 19790 ...
 $ Carrier           : Factor w/ 13 levels "AA","AS","B6",..: 4 4 4 4 11 11 11 11 11 4 ...
 $ TailNum           : Factor w/ 2816 levels "","D942DN","N0EGMQ",..: 2641 2512 2490 2581 1657 34 60 17
98 1443 473 ...
 $ FlightNum         : int  335 1095 2422 1607 657 894 1843 2041 413 2030 ...
 $ OriginAirportID   : int  11057 11057 11057 11057 11057 11057 11057 11057 11057 13232 ...
 $ OriginAirportSeqID: int  1105703 1105703 1105703 1105703 1105703 1105703 1105703 1105703 1105703 132
```

© Jigsaw Academy Pvt Ltd

Manipulating date objects

- Using Date class to convert a character into a Date

```
> fd$FlightDate<-as.Date(fd$FlightDate,"%d-%b-%y")
> str(fd)
'data.frame': 30443 obs. of  25 variables:
 $ FlightDate        : Date, format: "2014-01-01" "2014-01-01" "2014-01-01" ...
 $ UniqueCarrier     : Factor w/ 13 levels "AA","AS","B6",..: 4 4 4 4 11 11 11 11 11 4 ...
 $ AirlineID         : int  19790 19790 19790 19790 20355 20355 20355 20355 20355 19790 ...
 $ Carrier           : Factor w/ 13 levels "AA","AS","B6",..: 4 4 4 4 11 11 11 11 11 4 ...
 $ TailNum           : Factor w/ 2816 levels "","D942DN","N0EGMQ",..: 2641 2512 2490 2581 1657 34 60 17
98 1443 473 ...
 $ FlightNum         : int  335 1095 2422 1607 657 894 1843 2041 413 2030 ...
 $ OriginAirportID   : int  11057 11057 11057 11057 11057 11057 11057 11057 11057 13232 ...
 $ OriginAirportSeqID: int  1105703 1105703 1105703 1105703 1105703 1105703 1105703 1105703 1105703 132
```

© Jigsaw Academy Pvt Ltd

Manipulating date objects

- Using Date class to convert a character into a Date

Code	Value
%d	Day of month (decimal number)
%m	Month (decimal number)
%b	Month (abbreviated)
%B	Month (full name)
%y	Year (2 digits)
%Y	Year (4 digits)

- 25/Aug/04: “%d/%b/%y
- 25-August-2004:%d-%B-%Y

© Jigsaw Academy Pvt Ltd

Manipulating date objects

- Extracting months and weekdays from data:

```
> head(months(fd$FlightDate))
[1] "January" "January" "January" "January" "January" "January"
> unique(months(fd$FlightDate))
[1] "January"  "February" "March"
> head(weekdays(fd$FlightDate))
[1] "Wednesday" "Wednesday" "Wednesday" "Wednesday" "Wednesday" "Wednesday"
> unique(weekdays(fd$FlightDate))
[1] "Wednesday" "Thursday"  "Friday"    "Saturday"  "Sunday"    "Monday"    "Tuesday"
```

© Jigsaw Academy Pvt Ltd

Manipulating date objects

- Computing time intervals and using difftime()

```
> fd$FlightDate[60]-fd$FlightDate[900]
Time difference of -3 days
> difftime(fd$FlightDate[3000],fd$FlightDate[90],units = "weeks")
Time difference of 1.571429 weeks
> difftime(fd$FlightDate[3000],fd$FlightDate[90],units = "days")
Time difference of 11 days
> difftime(fd$FlightDate[3000],fd$FlightDate[90],units = "hours")
Time difference of 264 hours
```

© Jigsaw Academy Pvt Ltd

Manipulating date objects

- Manipulating data involving dates
- Sub-setting data: All rows when the day is Sunday

```
> library(dplyr)
> #Subset the data for day=Sunday
> dim(fd)
[1] 30443    25
> fd_s<-fd%>%filter(weekdays(FlightDate)=="Sunday")
> dim(fd_s)
[1] 4015   25
```

© Jigsaw Academy Pvt Ltd

Manipulating date objects

- Manipulating data involving dates
- Find the number of flights on Sundays for destination Atlanta

```
> #Find the number of flights on Sundays for destination Atlanta
> fd%>%filter(weekdays(FlightDate)=="Sunday",DestCityName=="Atlanta, GA")%>%nrow()
[1] 683
```

© Jigsaw Academy Pvt Ltd

Manipulating date objects

- Manipulating data involving dates
- Find the number of flights on Sundays for all cities

```
> #Find the number of flights on Sundays for all cities
> fd%>%filter(weekdays(FlightDate)=="Sunday")%>%group_by(DestCityName)%>%summarize(n())
Source: local data frame [10 x 2]

      DestCityName n()
1      Atlanta, GA 683
2    Charlotte, NC 342
3      Chicago, IL 193
4       Denver, CO 448
5      Houston, TX 155
6    Las Vegas, NV 507
7  Los Angeles, CA 603
8     New York, NY 349
9      Phoenix, AZ 466
10  Washington, DC 269
```

© Jigsaw Academy Pvt Ltd

Manipulating date objects

- Whenever data has time information along with date, R uses POSIXct and POSIXlt classes to deal with dates

```
> date1<-Sys.time()
> date1
[1] "2015-03-02 17:35:47 IST"
> class(date1)
[1] "POSIXct" "POSIXt"
> weekdays(date1)
[1] "Monday"
> months(date1)
[1] "March"
```

© Jigsaw Academy Pvt Ltd

Manipulating date objects

- Whenever data has time information along with date, we use POSIXct and POSIXlt classes to deal with dates

```
> date2<-as.POSIXlt(date1)
> date2
[1] "2015-03-02 17:35:47 IST"
> str(date2)
 POSIXlt[1:1], format: "2015-03-02 17:35:47"

> date2$wday
[1] 1
> date2$zone
[1] "IST"
> date2$hour
[1] 17
> date2$wday
[1] 1
> date2$zone
[1] "IST"
> date2$hour
[1] 17
```

© Jigsaw Academy Pvt Ltd

Manipulating date objects

- lubridate() is a package that is a wrapper for POSIXct class
- It has a very simple syntax

```
> library(lubridate)
> fd$FlightDate<-dmy(fd$FlightDate)
> str(fd)
'data.frame': 30443 obs. of  25 variables:
 $ FlightDate        : POSIXct, format: "2014-01-01" "2014-01-01" "2014-01-01" ...
 $ UniqueCarrier     : Factor w/ 13 levels "AA","AS","B6",..: 4 4 4 4 11 11 11 11 11 4 ...
 $ AirlineID         : int  19790 19790 19790 19790 20355 20355 20355 20355 20355 19790 ...
 $ Carrier           : Factor w/ 13 levels "AA","AS","B6",..: 4 4 4 4 11 11 11 11 11 4 ...
 $ TailNum           : Factor w/ 2816 levels "","D942DN","N0EGMQ",..: 2641 2512 2490 2581 1657 34 60 17
98 1443 473 ...
 $ FlightNum         : int  335 1095 2422 1607 657 894 1843 2041 413 2030 ...
 $ OriginAirportID   : int  11057 11057 11057 11057 11057 11057 11057 11057 11057 13232 ...
 $ OriginAirportSeqID: int  1105703 1105703 1105703 1105703 1105703 1105703 1105703 1105703 1105703 132
3202 ...
 $ OriginCityMarketID: int  31057 31057 31057 31057 31057 31057 31057 31057 31057 30977 ...
 $ Origin            : Factor w/ 10 levels "ATL","CLT","DCA",..: 2 2 2 2 2 2 2 2 2 9 ...
 $ OriginCityName    : Factor w/ 10 levels "Atlanta, GA",..: 2 2 2 2 2 2 2 2 2 3 ...
 $ OriginState       : Factor w/ 10 levels "AZ","CA","CO",..: 6 6 6 6 6 6 6 6 6 5 ...
```

© Jigsaw Academy Pvt Ltd

Manipulating date objects

- lubridate() is a package that is a wrapper for POSIXct class
- It has a very simple syntax.

Function	Date
dmy()	26/11/2008
ymd()	2008/11/26
mdy()	11/26/2008
dmy_hm()	26/11/2008 20:15
dmy_hms()	26/11/2008 20:15:30

© Jigsaw Academy Pvt Ltd

RECAP

- Character to date conversion-Date Class
- Extracting months and weekdays
- Using difftime()
- Manipulating data involving dates
- POSIXct and POSIXlt Classes
- Working with lubridate()

© Jigsaw Academy Pvt Ltd

Class 4 – Data Manipulation in R

Topic 4

★ Merging Tables ★

© Jigsaw Academy Pvt Ltd

INDEX

© Jigsaw Academy Pvt Ltd

Joining dataframes

© Jigsaw Academy Pvt Ltd

Joining dataframes

- Just like tables can be joined in sql, we can perform joins on dataframes in R
- Following types of joins can be accomplished
- Inner Join
- Left outer join
- Right outer join
- Full outer join

© Jigsaw Academy Pvt Ltd

Joining dataframes

- Inner join: Joining two tables based on a key column, such that rows matching in both tables are selected

	CustomerId	Product
1	1	Toaster
2	2	Toaster
3	3	Toaster
4	4	Radio
5	5	Radio
6	6	Radio

	CustomerId	State
1	2	Alabama
2	4	Alabama
3	6	Ohio

© Jigsaw Academy Pvt Ltd

Joining dataframes

- Inner join: Joining two tables based on a key column, such that rows matching in both tables are selected

	CustomerId	Product
1	1	Toaster
2	2	Toaster
3	3	Toaster
4	4	Radio
5	5	Radio
6	6	Radio

	CustomerId	State
1	2	Alabama
2	4	Alabama
3	6	Ohio

```
> merge(x=df1,y=df2,by="CustomerId")#Inner Join/Intersection of both tables
  CustomerId Product   State
1          2 Toaster Alabama
2          4   Radio Alabama
3          6   Radio    Ohio
```

© Jigsaw Academy Pvt Ltd

Joining dataframes

- Full Outer Join: Two tables are joined irrespective of any match between the rows

```
  CustomerId Product
1          1 Toaster
2          2 Toaster
3          3 Toaster
4          4   Radio
5          5   Radio
6          6   Radio
```

```
  CustomerId   State
1          2 Alabama
2          4 Alabama
3          6    Ohio
```

```
> merge(x = df1, y = df2, by = "CustomerId", all = TRUE)#Outer join:
  CustomerId Product   State
1          1 Toaster    <NA>
2          2 Toaster Alabama
3          3 Toaster    <NA>
4          4   Radio Alabama
5          5   Radio    <NA>
6          6   Radio    Ohio
```

© Jigsaw Academy Pvt Ltd

Joining dataframes

- Left Outer Join: All the rows of left table are retained while matching rows of right table are displayed

```
  CustomerId Product
1          1 Toaster
2          2 Toaster
3          3 Toaster
4          4   Radio
5          5   Radio
6          6   Radio
```

```
  CustomerId   State
1          2 Alabama
2          4 Alabama
3          6    Ohio
```

```
> merge(x = df1, y = df2, by = "CustomerId", all.x=TRUE)#Left join
  CustomerId Product   State
1          1 Toaster    <NA>
2          2 Toaster Alabama
3          3 Toaster    <NA>
4          4   Radio Alabama
5          5   Radio    <NA>
6          6   Radio    Ohio
```

© Jigsaw Academy Pvt Ltd

Joining dataframes

- Right Outer Join: All the rows of right table are retained while matching rows of left table are displayed

```
  CustomerId Product
1          1 Toaster
2          2 Toaster
3          3 Toaster
4          4   Radio
5          5   Radio
6          6   Radio
```

```
  CustomerId   State
1          2 Alabama
2          4 Alabama
3          6    Ohio
```

```
> merge(x = df1, y = df2, by = "CustomerId", all.y=TRUE)#Right join
  CustomerId Product   State
1          2 Toaster Alabama
2          4   Radio Alabama
3          6   Radio    Ohio
```

© Jigsaw Academy Pvt Ltd

RECAP

- Inner Join
- Left outer join
- Right outer join
- Full outer join

© Jigsaw Academy Pvt Ltd

Class 4 – Data Manipulation in R

Topic 5

★ Missing Value Treatment ★

© Jigsaw Academy Pvt Ltd

INDEX

© Jigsaw Academy Pvt Ltd

Missing Values

© Jigsaw Academy Pvt Ltd

Missing Values

- Identifying missing values in a column
- Imputing missing values

© Jigsaw Academy Pvt Ltd

Missing Values

- Using is.na() to find out the total number of missing values

```
> a<-c(1,2,3,4,5,6,NA,NA,NA,7,8,9)
> is.na(a)
 [1] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE FALSE FALSE FALSE
> sum(is.na(a))
[1] 3
```

© Jigsaw Academy Pvt Ltd

Missing Values

- Using is.na() to find out the total number of missing values

```
> air<-airquality
> head(air)
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
```

© Jigsaw Academy Pvt Ltd

Missing Values

- Using is.na() to find out the total number of missing values

```
> sum(is.na(air$Ozone))
[1] 37
> sum(is.na(air$Solar.R))
[1] 7
> summary(air)
     Ozone           Solar.R           Wind             Temp           Month            Day      
 Min.   :  1.00   Min.   :  7.0   Min.   : 1.700   Min.   :56.00   Min.   :5.000   Min.   : 1.0  
 1st Qu.: 18.00   1st Qu.:115.8   1st Qu.: 7.400   1st Qu.:72.00   1st Qu.:6.000   1st Qu.: 8.0  
 Median : 31.50   Median :205.0   Median : 9.700   Median :79.00   Median :7.000   Median :16.0  
 Mean   : 42.13   Mean   :185.9   Mean   : 9.958   Mean   :77.88   Mean   :6.993   Mean   :15.8  
 3rd Qu.: 63.25   3rd Qu.:258.8   3rd Qu.:11.500   3rd Qu.:85.00   3rd Qu.:8.000   3rd Qu.:23.0  
 Max.   :168.00   Max.   :334.0   Max.   :20.700   Max.   :97.00   Max.   :9.000   Max.   :31.0  
 NA's   :37       NA's   :7                                                                      
```

© Jigsaw Academy Pvt Ltd

Missing Values

- Imputing missing values

```
> #Imputing Missing values
> air$Ozone[is.na(air$Ozone)]<-45
> summary(air)
     Ozone          Solar.R          Wind             Temp           Month            Day
 Min.   :  1.00   Min.   :  7.0   Min.   : 1.700   Min.   :56.00   Min.   :5.000   Min.   : 1.0
 1st Qu.: 21.00   1st Qu.:115.8   1st Qu.: 7.400   1st Qu.:72.00   1st Qu.:6.000   1st Qu.: 8.0
 Median : 45.00   Median :205.0   Median : 9.700   Median :79.00   Median :7.000   Median :16.0
 Mean   : 42.82   Mean   :185.9   Mean   : 9.958   Mean   :77.88   Mean   :6.993   Mean   :15.8
 3rd Qu.: 46.00   3rd Qu.:258.8   3rd Qu.:11.500   3rd Qu.:85.00   3rd Qu.:8.000   3rd Qu.:23.0
 Max.   :168.00   Max.   :334.0   Max.   :20.700   Max.   :97.00   Max.   :9.000   Max.   :31.0
                  NA's   :7
```

© Jigsaw Academy Pvt Ltd

Missing Values

- Imputing missing values

```
> #Imputing Missing values
> air$Solar.R[is.na(air$Solar.R)]<-mean(air$Solar.R,na.rm=TRUE)
> summary(air)
     Ozone           Solar.R          Wind             Temp           Month            Day      
 Min.   :  1.00   Min.   :  7.0   Min.   : 1.700   Min.   :56.00   Min.   :5.000   Min.   : 1.0  
 1st Qu.: 21.00   1st Qu.:120.0   1st Qu.: 7.400   1st Qu.:72.00   1st Qu.:6.000   1st Qu.: 8.0  
 Median : 45.00   Median :194.0   Median : 9.700   Median :79.00   Median :7.000   Median :16.0  
 Mean   : 42.82   Mean   :185.9   Mean   : 9.958   Mean   :77.88   Mean   :6.993   Mean   :15.8  
 3rd Qu.: 46.00   3rd Qu.:256.0   3rd Qu.:11.500   3rd Qu.:85.00   3rd Qu.:8.000   3rd Qu.:23.0  
 Max.   :168.00   Max.   :334.0   Max.   :20.700   Max.   :97.00   Max.   :9.000   Max.   :31.0  
```

© Jigsaw Academy Pvt Ltd

RECAP

- Identifying missing values in a column
- Imputing missing values

© Jigsaw Academy Pvt Ltd

Class 4 – Data Manipulation in R

Topic 6

Transposing data using reshape2()

© Jigsaw Academy Pvt Ltd

INDEX

© Jigsaw Academy Pvt Ltd

Transposing data: reshape2()

© Jigsaw Academy Pvt Ltd

Transposing data

- Understanding wide and long data formats
- Converting data in wide format to long
- Converting data in long format to wide

© Jigsaw Academy Pvt Ltd

Transposing data

- Understanding wide and long data formats
- Most structured data is in wide format: Variables are columns and Row labels identify observations

Person	Age	Weight
Sankar	26	70
Aiyar	24	60
Singh	25	65

© Jigsaw Academy Pvt Ltd

Transposing data

- Understanding wide and long data formats
- The same data can be represented as follows (long format)

Persons	Variable	Value
Sankar	Age	26
Sankar	Weight	70
Aiyar	Age	24
Aiyar	Weight	60
Singh	Age	25
Singh	Weight	65

© Jigsaw Academy Pvt Ltd

Transposing data

- Converting a wide format data to long format: melt()

```
> library(reshape2)
> person<-c("Sankar","Aiyar","Singh")
> age<-c(26,24,25)
> weight<-c(70,60,65)
> wide<-data.frame(person,age,weight)
> wide
  person age weight
1 Sankar  26     70
2  Aiyar  24     60
3  Singh  25     65
```

© Jigsaw Academy Pvt Ltd

Transposing data

- Converting a wide format data to long format: melt()

```
> melt(wide)
Using person as id variables
  person variable value
1 Sankar      age    26
2  Aiyar      age    24
3  Singh      age    25
4 Sankar   weight    70
5  Aiyar   weight    60
6  Singh   weight    65
> melt(wide,id.vars="person")
  person variable value
1 Sankar      age    26
2  Aiyar      age    24
3  Singh      age    25
4 Sankar   weight    70
5  Aiyar   weight    60
6  Singh   weight    65
```

© Jigsaw Academy Pvt Ltd

Transposing data

- Converting a wide format data to long format: melt()

```
> melt(wide,id.vars="person",variable.names="Demographics",value.name ="Demo_Value" )
  person variable Demo_Value
1 Sankar      age         26
2  Aiyar      age         24
3  Singh      age         25
4 Sankar   weight         70
5  Aiyar   weight         60
6  Singh   weight         65
> melted<-melt(wide,id.vars="person",variable.names="Demographics",value.name ="Demo_Value" )
```

© Jigsaw Academy Pvt Ltd

Transposing data

- Converting a long format data to wide format: dcast()

```
> dcast(melted,person~variable,value.var = "Demo_Value")
  person age weight
1  Aiyar  24     60
2 Sankar  26     70
3  Singh  25     65
```

© Jigsaw Academy Pvt Ltd

RECAP

- Understanding wide and long data formats
- Converting data in wide format to long
- Converting data in long format to wide

© Jigsaw Academy Pvt Ltd

Class 4 – Data Manipulation in R

Topic 7

★ Manipulating Character Strings and Using sqldf ★

© Jigsaw Academy Pvt Ltd

INDEX

© Jigsaw Academy Pvt Ltd

Manipulating Strings in R

© Jigsaw Academy Pvt Ltd

Manipulating Strings in R

- We'll cover the following R functions used to manipulate character data
- substr()
- nchar()
- tolower(), toupper()
- strsplit(),paste()
- grep(),grepl(),
- sub(),gsub()

© Jigsaw Academy Pvt Ltd

Manipulating Strings in R

- substr(), nchar(),tolower(),toupper()

```
> a<-"Batman"
> substr(a,start=2,stop=6)
[1] "atman"
> nchar(a)
[1] 6
> tolower(a)
[1] "batman"
> toupper(a)
[1] "BATMAN"
```

© Jigsaw Academy Pvt Ltd

Manipulating Strings in R

- strsplit(), paste(),grep(),grepl(),sub(),gsub()

```
> b<-"Bat-Man"
> c<-"Bat/Man"
> strsplit(c,split="/")
[[1]]
[1] "Bat" "Man"

> paste(b,split=c)
[1] "Bat-Man Bat/Man"
> grep("-",b)
[1] 1
> grepl("/",c)
[1] TRUE
> sub("-","/",b)
[1] "Bat/Man"
> d<-"Bat-Ma-n"
> sub("-","/",d)
[1] "Bat/Ma-n"
> gsub("-","/",d)
[1] "Bat/Ma/n"
```

© Jigsaw Academy Pvt Ltd

INDEX

© Jigsaw Academy Pvt Ltd

Using sql commands within R

© Jigsaw Academy Pvt Ltd

SQL queries within R: sqldf

- One can easily use sql queries within R using a package called sqldf
- This package is available at CRAN
- install.package(sqldf)
- library(sqldf)

© Jigsaw Academy Pvt Ltd

Manipulating Strings in R

- Using select statement

```
> library(sqldf)
> #Using SELECT statement
> oj_s<-sqldf("select brand, income, feat from oj ")
```

- Using where statement

```
> #Subseting using where statement
> oj_s<-sqldf("select brand, income, feat from oj where price<3.8 and income<10")
```

- Using order by statement

```
> #Order by statement
> oj_s<-sqldf("select store,brand,week,logmove,feat,price, income from oj order by income asc")
```

© Jigsaw Academy Pvt Ltd

SQL queries within R: sqldf

- Using sql functions

```
> #Demo sql functions
> sqldf("select avg(income) from oj")
  avg(income)
1    10.61673
> sqldf("select min(price) from oj")
  min(price)
1       0.52
```

© Jigsaw Academy Pvt Ltd

RECAP

- Basic string manipulation functions: tolower, toupper,grep, grepl, sub, gsub
- Executing sql queries in R, using sqldf() package

© Jigsaw Academy Pvt Ltd