

# Data Transformation with data.table :: CHEAT SHEET



## Basics

data.table is an extremely fast and memory efficient package for transforming data in R. It works by converting R's native data frame objects into data.tables with new and enhanced functionality. The basics of working with data.tables are:

```
dt[i, j, by]
```

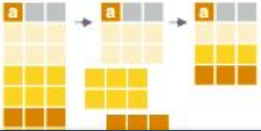
## Manipulate columns with j

### EXTRACT



dt[, c(2)] - extract columns by number. Prefix column numbers with "-" to drop.

## Group according to by



dt[, j, by = .(a)] - group rows by values in specified columns.

dt[, j, keyby = .(a)] - group and simultaneously sort rows by values in specified columns.

# Wrangling 4.6M Rows (375MB)

With data.table

Difficulty: Intermediate



asDT(df) or as.data.table(df) - convert a data frame or a list to



dt[, c:= 1+2] - compute a column based on

chaining

Matt Dancho & David Curry  
Business Science Learning Lab



# Learning Lab Structure

- **Presentation**  
(20 min)
- **Demo's**  
(20 min)
- **Presentation**  
(20 mins)



## Your Hosts!



**Matt Dancho**

Founder of Business Science, Matt designs and executes educational courses and workshops that deliver immediate value to organizations. His passion is **up-leveling future data scientists** coming from **untraditional backgrounds**.



**David Curry**

Founder of Sure Optimize, David works with businesses to help improve website performance and SEO using data science. His passion is **ethical Machine Learning** initiatives.

# Success Story



## Stephen Lung

- Senior Financial Analyst at Toronto Stock Exchange
- Took DS4B 101-R
- Participated in Tableau Challenge
- Zero working knowledge of Tableau
- **Placed 3rd**
- Beat out peers with 2+ years experience with Tableau
- **Secret Weapon?**



*“This is legit a milestone in my development.”*

**Stephen Lung** Jun 28th at 9:02 PM  
Last week, I entered a Tableau data visualization contest. With zero working knowledge, I had used alot of the DSB101 core concepts to explore the data and even used the DataExplorer package to get a good sense of some interesting parts of the data. What blew me away was how applicable the content I learned was through the DSB courses with @Matt Dancho in a time-sensitive nature of the competition. With this I was able to iterate through multiple concepts to come up with a story for my dashboard. Through all of this, I achieved 3rd place and shockingly, I even had multiple people from the audience mention that I deserved 1st place. For me, the biggest amazement factor was that I was doing more in-depth data analysis (like correlation matrixes) than technical people who had 2 yrs of experience with the Tableau tool. Thanks for your time and stay humble, everyone!

2 likes, 2 shares, 4 comments, 2 reposts

2 replies

**Shreyas** 2 days ago  
Wow! That's awesome Stephen. Congratulations!  
1 like

**Matt Dancho** 2 days ago  
That's amazing. Stephen you are rocking it! Sky is the limit for you.  
1 like



# Congrats Stephen!!!



You just crushed your first .  
Tableau contest.

Secret Weapon:



**#BusinessScienceSuccess**

System tray: 100% CPU, Thu Jun 27 3:09 PM, Matthew Dancho

Environment: mortgage\_loans\_datatable

# Agenda

Data

- Data\_A: 426207 obs. of 25 variables
- Data\_P: 4645448 obs. of 31 variables

Values

Acquisitions: "data//Acquisition\_2018Q1.txt"

fileslocation: "data/"

k: 1

Performance: "data//Performance\_2018Q1.txt"

Performance\_ColClas... chr [1:31] "character" "character" "character" "numeric..."

Performance\_Variabl... chr [1:31] "LOAN\_ID" "Monthly.Rpt.Prd" "Servicer.Name" ...

Functions

na.lomf: function (x)

Files | Plots | Packages | Help | Viewer

Home > Desktop > mortgage\_loans\_datatable > data

Name	Size	Modified
..		
..DS_Store	6 KB	Jun 27, 2019, 11:13 AM
2018Q1.zip	56 MB	Jun 27, 2019, 11:10 AM
Acquisition_2018Q1.txt	44.2 MB	Jun 27, 2019, 11:15 AM
Performance_2018Q1.txt	375.5 MB	Jun 27, 2019, 11:15 AM

- **Business Case Study**

- Fannie Mae Home Loan Data
- 1 Quarter
- **4.6M Rows (375 MB)**
- 25GB Total

- **Demo**

- Wrangling 4.6M Rows

- **Large Data Strategies**

- Secret Tactics
- Learning Plan

- **Solution(s)**

- Tools
- data.table

- **Resources**

- Learn FAST
- DT Basics



# Learning Labs PRO

Every 2 Weeks

Get Code

Recordings

Slack Community

\$19/month

[university.business-science.io](https://university.business-science.io)

*Lab 13*  
**Wrangling 4.6M Rows w/  
data.table**

*Lab 12*  
**How I built anomalize**

*Lab 11*  
**Market Basket Analysis w/  
recommenderLab**

*Lab 10*  
**Building API's with  
plumber & postman**

*Lab 9*  
**Finance in R with  
tidyquant**



**Learning Labs Pro**  
Community-Driven Data Science Courses

 Matt Dancho

**\$19/m**

# Business Case Study

Analyze Loan Defaults



# Bank Loan Defaults

## Business Objectives

Loan defaults cost organizations multi-millions

Need to understand which people or institutions will default on loans

**Large Data + Prediction**







# Fannie Mae Loan Data

## Loan Acquisition & Performance

Each quarter = 5M Rows of Data

Since 2000 = 25GB Data

How do we analyze this **massive data set**?

Quarterly Single Family Eligible Fixed Rate Mortgage Dataset				
Year	Q1 Records	Q2 Records	Q3 Records	Q4 Records
2000	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2001	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2002	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2003	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2004	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2005	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2006	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2007	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2008	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2009	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2010	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2011	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2012	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2013	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2014	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2015	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2016	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2017	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>	<a href="#">Acquisition and Performance</a>
2018	<a href="#">Acquisition and Performance</a>	Not Available	Not Available	Not Available

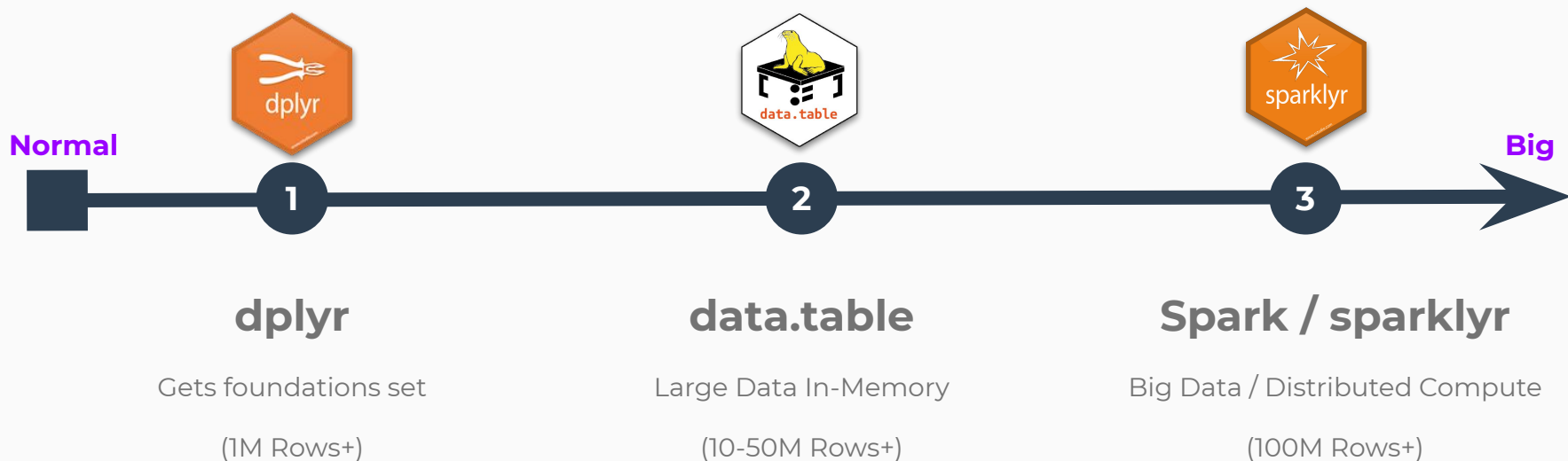


**The Solution(s)**



# Data Wrangling Tools

## by Dataset Size





# What Tools Exist?

## data.table

High-performance version of base R's data.frame

The screenshot shows the GitHub repository page for `Rdatatable / data.table`. At the top, it displays repository statistics: 187 watchers, 1,964 stars, and 765 forks. Below this, there are tabs for Code, Issues (710), Pull requests (13), Wiki, Security, and Insights. The main heading is "Home", with a note that Matt Dowle edited this page on Apr 10 - 160 revisions. A central box contains the `data.table` logo (a yellow dog on a black box) and a row of status badges: CRAN (OK), codecov (98%), downloads (452K/month), Depsy (100th percentile), Lin/Mac (build passing), Win (build passing), Extra (pipeline passed), and Latest news from dev: NEWS. Below this, it says "New presentations in 2018: click Events in sidebar menu." The main text describes `data.table` as one of the 13,000 add-on packages for R, providing a high-performance version of `data.frame` with syntax and feature enhancements. It mentions that as of Nov 2018, `data.table` was the 4th largest Stack Overflow tag, the 10th most starred R package on GitHub, and had over 650 CRAN and Bioconductor packages using it. A link is provided for a benchmark: `h2oai.github.io/db-benchmark`. Below the text is a bar chart comparing `data.table` (blue) and `dplyr` (orange) across various benchmarks. The chart shows `data.table` generally performing better (lower time) than `dplyr`. On the right sidebar, there is a "Pages" section with a list of links: Home, Getting started, Events: Videos & Slides, Articles, Installation, Support, Contributing, `?data.table ?read ?fwrite fread for small data`, Benchmarks: Grouping, Do's and Don'ts, `#rdatable`, `@MattDowle`, `@arun_sriniv`, and `data.table`. At the bottom of the sidebar, there is a yellow box with the text `H2O` and a "Clone this wiki locally" button with a URL: `https://github.com/Rdatatable`.



# How does **data.table** help?

## dplyr

Designed for **readability**.

**Makes copies** through the piping process.

Normally OK.

Large data is **not memory or speed efficient**.

```
199 # data.table
200 tic()
201 combined_data[, gt_3mo_behind_in_1yr := lead(current_loan_delinquency_status, n = 3) >= 1,
202             by = loan_id]
203 toc()
204
205 combined_data
206
207 # dplyr
208 tic()
209 combined_data %>%
210   group_by(loan_id) %>%
211   mutate(gt_3mo_behind_in_1yr_dplyr = lead(current_loan_delinquency_status, n = 3) >= 1) %>%
212   ungroup() %>%
213   filter(gt_3mo_behind_in_1yr_dplyr) |
214 toc()
215
216
```



# How does **data.table** help?

## data.table

Designed for **memory & speed efficient**.

Uses **:=** and **set** functions to **modify inplace (no copies)**

### Cons

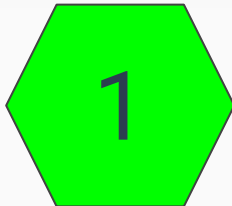
- Less readable
- Doesn't make copies

```
> tic()
> combined_data[, gt_3mo_behind_in_1yr := lead(current_loan_delinquency_status, n = 3) >= 1,
+             by = loan_id]
> toc()
4.278 sec elapsed
> tic()
> temp <- combined_data %>%
+   group_by(loan_id) %>%
+   mutate(gt_3mo_behind_in_1yr_dplyr = lead(current_loan_delinquency_status, n = 3) >= 1) %>%
+   ungroup() %>%
+   filter(gt_3mo_behind_in_1yr_dplyr)
> toc()
8.27 sec elapsed
```

# Data.Table Resources

Get up to speed FAST

# Resource #1: Ultimate R Cheat Sheet



Page 2

### Data Science with R

Web Applications & the "Shinyverse"

Start → Components → Advanced Concepts (Optional) → Testing → Publish

- Start: `Shiny (CS)`, `Flexdashboard`
- Components: `Widgets`, `shinyWidgets`, `shinyWidgets`, `shiny`
- Advanced Concepts (Optional): `crossstalk`, `shinyrouter`, `Avic`, `leaflet`, `leaflet`
- Testing: `testshiny`, `shinytest`, `shinydashboard`
- Publish: `shinyapps.io`, `Shiny Server`, `Connect (Enterprise)`

### Flexdashboard Apps

Flexdashboard is an RMarkdown-based dashboard tool that can be used to integrate shiny components at runtime. Development is fast and efficient, but layouts are not as open to modification as building an app using Shiny.

### Shiny Apps

Shiny is an R-Package that enables web app development from R. Contains R functions for common HTML structures, UI Controls (components/widgets), and web framework tools. The framework is highly flexible, but users require more knowledge of HTML & CSS.

### Themes, Dashboards, & Examples

- Flexdashboard Gallery
- Themes Layouts
- Semantic `shiny.semantic`, `semantic.dashboard`
- Bootstrap 4
- bedDash

"Data Science Courses for Business"

Page 1

### Data Science with R Workflow

If you want to learn R and this workflow for business analysis, take the [R For Business Analysis \(DS48101-R\)](#) course through Business Science University.

Click the links for Documentation

CS = Cheat Sheet

Import → Tidy → Transform → Visualize → Communicate

- Import: `readr (CS)`, `readxl (CS)`, `readf (CS)`, `read_csv (CS)`, `read_excel (CS)`, `read_sql (CS)`, `readr (CS)`
- Tidy: `tidyr (CS)`, `tidy (CS)`
- Transform: `shiny (CS)`, `stringr (CS)`, `lubridate (CS)`, `forcats (CS)`, `base R (CS)`
- Visualize: `ggplot2 (CS)`
- Communicate: `RMarkdown (CS)`, `Shiny (CS)`
- Model: `recipes (CS)`, `rsample (CS)`, `loom (CS)`, `yardstick (CS)`, `parsnip (CS)`

### Important Resources

- R For Data Science Book: <http://r4ds.had.co.nz/>
- Interactive Book: <https://bookdown.org/vjames/rmarkdown/>
- Data Visualization Book: <https://r-graphical.com/>
- More Cheatsheets: <https://www.stypi.com/resources/cheatsheets/>
- Shinyverse packages: <https://www.shiny-verse.com/>
- Connecting to database: <https://rdb.stypi.com/>
- RMarkdown websites: <https://rmarkdown.rstudio.com/>
- Shiny web applications website: <http://shiny.stypi.com/>
- Jenny Bryan's purrr tutorial: <https://jennybryan.net/2016/05/01/purrr/>

"Data Science Courses for Business"

Business Science University [university.business-science.io](http://university.business-science.io)

Page 3

### Data Science with R

Special Topics

#### Time Series Analysis

- Time-aware KStest: `tsibble` & `tblske`
- Convert between classes: `tsibble` & `tblske`
- Time Series Index Summary: `tblske`
- Generating Future Series: `tblske`

#### Forecasting

- ARIMA, ETS, etc: `forecast` & `fable`
- Tidy, glance, augment for forecast models: `forecast`
- Converting forecast prediction to `tblske`: `tblske`

#### Text Analysis & NLP

- Text Mining with R: `Book` `spdfair`
- NLP: `text2vec`, `word_embeddings`, `text2vec`, `text_vectorization`, `topic_modeling`, `spdfair`, `LDPrep`, `C++` & `Java` in R

#### Network Analysis

- Network Data Transformations (Tidy): `tidygraph`
- Network Data Transformations: `igraph`

#### Network Viz

- Static: `igraph` - Graph plotting utilities for `igraph2`
- Interactive Visualization: `vis` - Graph plotting utilities for `igraph2` in R
- `visNetwork` - `visNetwork` in R
- `visNetwork` - `visNetwork` in R

#### Geospatial Analysis

- Geocoding (getting latitude, longitude, & srid): `geocode` - Google API (requires key)
- `geonames` - OpenStreetMap Overpass API
- `leaflet` - OpenStreetMap Nominatum API
- Simple Features (sf objects): `sf (CS)` (body)
- Spatial Objects (sp objects): `sp (non-body)`

#### Geospatial Viz

- Static: `ggmap` - Google API (requires key)
- `ggmap` - Impressive Maps via OSM
- `leaflet` - Thematic Maps via OSM
- `leaflet` - Thematic Maps via OSM
- Interactive Visualization: `leaflet` - `leaflet.js` in R
- `leaflet` - `leaflet.js` in R

#### Machine Learning

- Multi-Threaded/Scalable/Production ML: `H2O (CS)`
- Extensive Gradient Boosting: `xgboost`
- R + Spark: `sparklyr (CS)`
- Sparkling Water (Spark + H2O): `sparkling`
- ML (Tidy): `paradox`
- ML: `caret (CS)`

#### Deep Learning

- R Interface to TensorFlow: `tf`
- Keras (CS)
- TF Estimator
- TensorFlow (Docs)

#### Speed & Scale

- Faster than dplyr & parsnip: `data.table (CS)`
- Distributed Cluster (Spark): `sparklyr (CS)`
- Parallel Processing: `furrr`

#### Interoperability

- Python: `reticulate (CS)` - Java: `Java`
- C++: `Rcpp`

#### Miscellaneous Tools

- Interactive Plotting: `interactiveshell (CS)`
- Building R Packages: `R packages Book`
- Build Web Docs: `bookdown (CS)`
- R Templates: `usethis`
- Build Web Docs: `bookdown (CS)`
- Advanced Concepts (Advanced R Book)
- `shiny` & `tidy` Evaluation (CS)
- Making Blogs & Books: `bookdown`, `bookdown`
- Posting Code (GitHub, Stack Overflow): `source`

"Data Science Courses for Business"

Business Science University [university.business-science.io](http://university.business-science.io)



# Resource #2: data.table Cheat Sheet



## Page 3

- TensorFlow (Core)

### Speed & Scale

- Faster than dplyr & pandas: [data.table \(CS\)](#)
- Distributed Cluster (Spark): [sparklyr \(CS\)](#)
- Parallel Processing: [furr](#)

### Interoperability

- Python: [reticulate \(CS\)](#) • Java: [rJava](#)
- C++: [Rcpp](#)

### Miscellaneous Tools

- Interactive Plotting: [htmlwidgets for R](#)
- Building R Packages: [R packages Book](#)
  - Pkg Development Tools: [devtools \(CS\)](#)
  - R Templates: [usethis](#)
  - Build Web Doc's: [pkgdown](#)
- Advanced Concepts ([Advanced R Book](#))
  - [rlang & Tidy Evaluation \(CS\)](#)
- Making Blogs & Books:
  - [blogdown](#) [bookdown](#)
- Posting Code (GitHub, Stack Overflow): [reprex](#)



Business Science University  
[university.business-science.io](http://university.business-science.io)

2

## Data Transformation with data.table :: CHEAT SHEET



### Basics

data.table is an extremely fast and memory efficient package for transforming data in R. It works by converting R's native data frame objects into data.tables with new and enhanced functionality. The basics of working with data.tables are:

`dt[i, j, by]`

Take data.table `dt`, subset rows using `i` and manipulate columns with `j`, grouped according to `by`.

data.tables are also data frames – functions that work with data frames therefore also work with data.tables.

### Create a data.table

`data.table(a = c(1, 2), b = c("a", "b"))` – create a data.table from scratch. Analogous to `data.frame()`.

`setDT(df)` or `as.data.table(df)` – convert a data frame or a list to a data.table.

### Subset rows using i

`dt[1:2, ]` – subset rows based on row numbers.

`dt[a > 5, ]` – subset rows based on values in one or more columns.

### LOGICAL OPERATORS TO USE IN i

<	<=	is.na()	%in%		%like%
>	>=	!is.na()	!	&	%between%

### Manipulate columns with j

#### EXTRACT

`dt[, c(2)]` – extract columns by number. Prefix column numbers with “-” to drop.

`dt[, -(b, c)]` – extract columns by name.

#### SUMMARIZE

`dt[, (x = sum(a))]` – create a data.table with new columns based on the summarized values of rows.

Summary functions like `mean()`, `median()`, `min()`, `max()`, etc. can be used to summarize rows.

#### COMPUTE COLUMNS\*

`dt[, c := 1 + 2]` – compute a column based on an expression.

`dt[a == 1, c := 1 + 2]` – compute a column based on an expression but only for a subset of rows.

`dt[, := (c = 1, d = 2)]` – compute multiple columns based on separate expressions.

#### DELETE COLUMN

`dt[, c := NULL]` – delete a column.

#### CONVERT COLUMN TYPE

`dt[, b := as.integer(b)]` – convert the type of a column using `as.integer()`, `as.numeric()`, `as.character()`, `as.Date()`, etc..

### Group according to by

`dt[, , by = .(a)]` – group rows by values in specified columns.

`dt[, , keyby = .(a)]` – group and simultaneously sort rows by values in specified columns.

#### COMMON GROUPED OPERATIONS

`dt[, (c = sum(b), by = a)` – summarize rows within groups.

`dt[, c := sum(b), by = a]` – create a new column and compute rows within groups.

`dt[, .SD[1], by = a]` – extract first row of groups.

`dt[, .SD[N], by = a]` – extract last row of groups.

### Chaining

`dt[...][...]` – perform a sequence of data.table operations by chaining multiple “[...]”.

### Functions for data.tables

#### REORDER

`setorder(dt, a, -b)` – reorder a data.table according to specified columns. Prefix column names with “-” for descending order.

#### \* SET FUNCTIONS AND :=

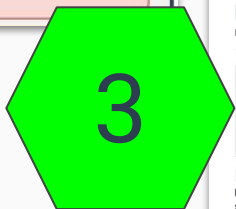
data.table's functions prefixed with “set” and the operator “:=” work without “<-” to alter data without making copies in memory. E.g., the more efficient “setDT(df)” is analogous to “df <- as.data.table(df)”.



## Page 3

### Speed & Scale

- Faster than dplyr & pandas: [data.table \(CS\)](#)
- Distributed Cluster (Spark): [sparklyr \(CS\)](#)
- Parallel Processing: [furry](#)



Home

Matt Dowle edited this page on Apr 10 - 160 revisions

CRAN: OK | codecov: 99% | downloads: 452K/month | Deploy: 100th percentile

Lin/Mac: build passing | Win: build passing | Extra: pipeline passing

Latest news from dev: [NEWS](#)

New presentations in 2018: click [Events](#) in sidebar menu.

`data.table` is one of the 13,000 add-on packages for the programming language R which is popular in [these fields](#). It provides a high-performance version of base R's `data.frame` with syntax and feature enhancements for ease of use, convenience and programming speed. As of Nov 2018, `data.table` was the 4th largest Stack Overflow tag about an R package with over 8,000 questions, the 10th most starred R package on GitHub and had over 650 CRAN and Bioconductor packages using it. Here are a further [6,000 accepted answers](#) which use/mention `data.table` but where the question was not specifically about `data.table`.

We have updated the 2014 grouping benchmarks comparing `data.table` to pandas and dplyr, and included Spark and pydatatable. The benchmark is automated and runs regularly against the latest versions of these packages. It is a work in progress: [h2oai.github.io/db-benchmark](#)

Clone this wiki locally

<https://github.com/rdatatable>

```
> require(data.table)
> example(data.table)
```

```
# basic row subset
DT[2]
DT[2:3]
w=2:3; DT[w]
DT[order(x)]
DT[order(x), ]
DT[!y>2]
DT[!y>2 & v>5]
DT[12:4]
DT[-(2:4)]
```

```
# select|compute columns
DT[, v]
DT[, list(v)]
DT[, .(v)]
DT[, sum(v)]
DT[, .sum(v)]
DT[, .(sv=sum(v))]
DT[, .(v, v+2)]
```

```
# subset rows and select|compute
DT[2:3, sum(v)]
DT[2:3, .(sum(v))]
DT[2:3, .(sv=sum(v))]
DT[2:5, cat(v, "\n")]
```

```
# select columns the data.frame way
DT[, 2]
coNum = 2
DT[, ..coNum]
DT[["v"]]
```

```
# grouping operations - j and by
DT[, sum(v), by=x]
DT[, sum(v), keyby=x]
DT[, sum(v), by=x][order(x)]
```

```
# 2nd row
# 2nd and 3rd row
# same
# no need for DT's prefix on column x
# same; the ',' is optional
# all rows where DT$y > 2
# compound logical expressions
# all rows other than 2:4
# same
```

```
# v column (as vector)
# v column (as data.table)
# same; .() is an alias for list()
# sum of column v, returned as vector
# same but return data.table
# same but name column "sv"
# return two column data.table
```

```
# sum(v) over rows 2 and 3
# same, but return data.table
# same, but name column "sv"
# just for j's side effect
```

```
# 2nd column, a data.table always
DT[, ..coNum]
# same as DT[, 2]; ..var => one-up
# same as DT[, v] but lower overhead
```

```
# appearance order of groups preserved
# order the result by group
# same by chaining expressions together
```

# Data.Table Basics


80/20 Concepts & Important  
Operations



# Critical Concept #1

Learn this:

**DATA TABLES**



- think in terms of basic units — *rows*, *columns* and *groups*
- `data.table` syntax provides *placeholder* for each of them

General form: **DT**[*i*, *j*, *by*]

On which rows      What to do?      Grouped by what?



# Critical Concept #2

Understand this:

## Modifying In-Place

**How?**

`:=`

**Why?**

No Copies (Speed boost)

**Example**

```
DT[, unpaid_flag := unpaid_bal >= 1]
```

# Row Operations - Filtering & Arranging



```
# basic row subset
DT[2]                # 2nd row
DT[2:3]              # 2nd and 3rd row
w=2:3; DT[w]         # same
DT[order(x)]         # no need for DT$ prefix on column x
DT[order(x), ]       # same; the ',' is optional
DT[y>2]              # all rows where DT$y > 2
DT[y>2 & v>5]        # compound logical expressions
DT[!2:4]              # all rows other than 2:4
DT[-(2:4)]           # same
```

Similar to dplyr functions

**filter()**  
**arrange()**

# Column Operations - Selecting & Summarizing



```
# select|compute columns
DT[, v]                # v column (as vector)
DT[, list(v)]          # v column (as data.table)
DT[, .(v)]             # same; .() is an alias for list()
DT[, sum(v)]           # sum of column v, returned as vector
DT[, .(sum(v))]        # same but return data.table
DT[, .(sv=sum(v))]     # same but name column "sv"
DT[, .(v, v*2)]        # return two column data.table
```

Similar to dplyr functions

**select()**  
**summarize()**



# Grouping Operations - Grouping & Summarizing



```
# grouping operations - j and by
DT[, sum(v), by=x]                # appearance order of groups preserved
DT[, sum(v), keyby=x]            # order the result by group
DT[, sum(v), by=x][order(x)]     # same by chaining expressions together
```

Similar to dplyr functions  
**group\_by() + summarize()**

# Grouping Operations - Grouping & Mutating



Speedup

**This**

Modifies

**This**

Inplace

```
199 combined_data[, gt_3mo_behind_in_1yr := lead(current_loan_delinquency_status, n = 3) >= 1,  
200 by = loan_id]
```

Similar to dplyr functions  
**group\_by() + mutate()**



```
DT[, sum(v), by=x][order(x)] # same by chaining expressions together
```

Similar to dplyr functions  
**Pipe %>%**



```
# joins as subsets
X = data.table(x=c("c","b"), v=8:7, foo=c(4,2))
X

DT[X, on="x"] # right join
X[DT, on="x"] # left join
DT[X, on="x", nomatch=0] # inner join
DT[!X, on="x"] # not join
DT[X, on=c(y="v")] # join DT$y to X$v
DT[X, on="y==v"] # same
```

Similar to dplyr functions

**left\_join()**  
**right\_join()**  
**etc**



```
# more on special symbols, see also ?"special-symbols"
DT[.N]                                # last row
DT[, .N]                               # total number of rows in DT
DT[, .N, by=x]                         # number of rows in each group
DT[, .SD, .SDcols=x:y]                # select columns 'x' and 'y'
DT[, .SD[1]]                           # first row; same as DT[1,]
DT[, .SD[1], by=x]                     # first row of each group
DT[, c(.N, lapply(.SD, sum)), by=x]    # group size alongside sum
DT[, .I[1], by=x]                      # row number of first row of each group
DT[, grp := .GRP, by=x]                # add a group counter column
X[, DT[.BY, y, on="x"], by=x]         # join within group to use less ram
```

Similar to dplyr functions

**.N == n()**

**.SD == slice()**

# Demo

4.6M rows / 375 MB Data  
*data.table* in action

# Large Data Strategy

Secret Tactics



# Big Data

**Impacts Every Company**

Data storage is increasing exponentially.

How can we deal with it?



## Trick to Solving Big Data Problems. Make them small.

Large datasets can be **sampled**.

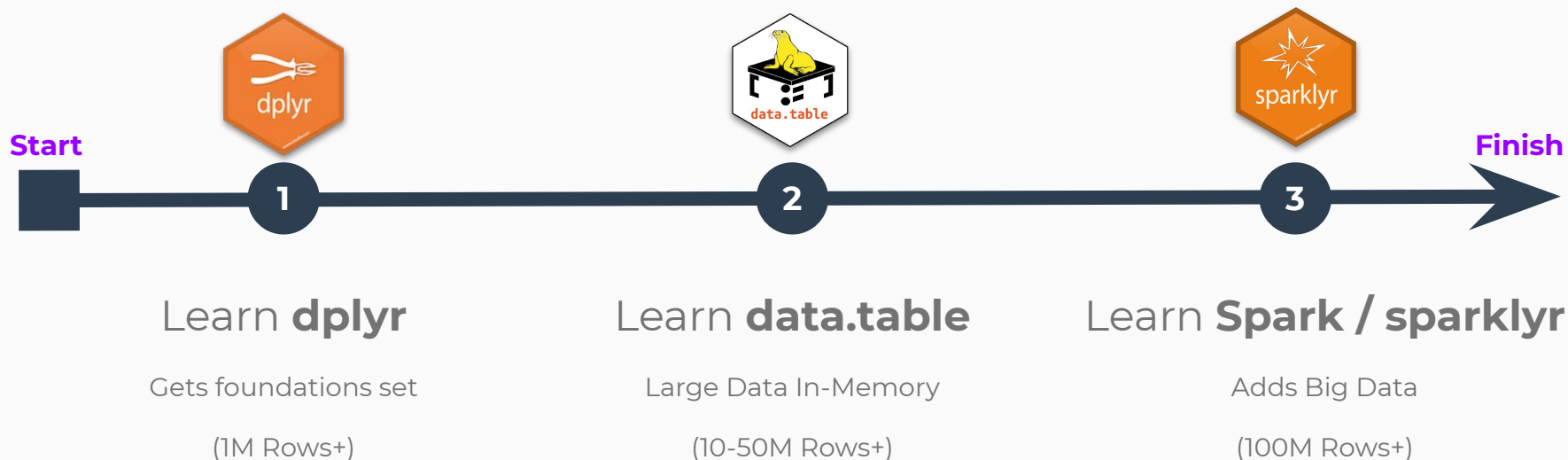
Sampling makes data **manageable**.

**Good sampling strategy:** Loss in ML accuracy is typically low.

Upgrade to **Big Data Tools** once you have a good methodology.



# Big Data Learning Plan



Data Wrangling **Foundations** Are The Key

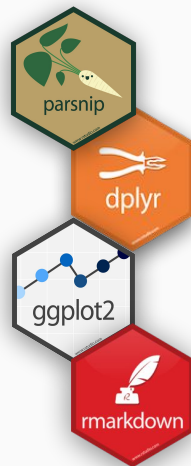


# Big Data Learning Plan

## Start with Foundations

35 Hours of Video Lessons

- Machine Learning (parsnip)
- **Data Manipulation (dplyr)**
- Visualization (ggplot2)
- Reporting (rmarkdown)
- More packages



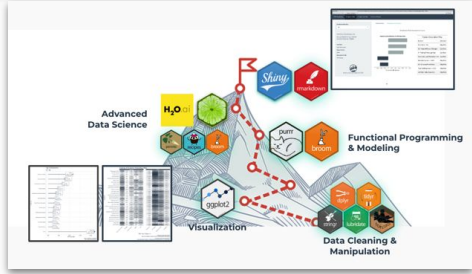
**DS4B 101-R: Business Analysis With R**

Your Data Science Journey Starts Now! Learn the fundamentals of data science for business with the tidyverse.

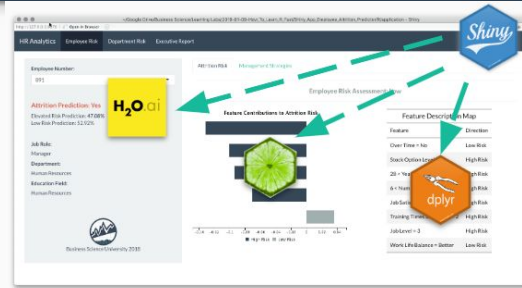
 Matt Dancho

# Big Data Learning Plan

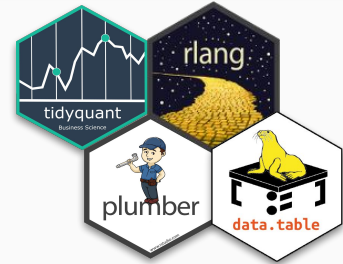
# YOUR Transformation



**Do Business Projects**  
Climb the Hill



**Build Production-Ready**  
Web Apps



**Complete 1-Hour Courses**  
Domain Analysis & Tool Courses

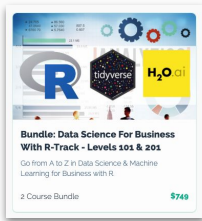
Start

Finish

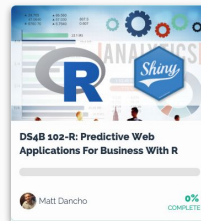
1

2

3



Analysis Courses



App Development  
Courses



Learning Labs PRO

Everything is **Taken Care of** For You in Our Platform

# 3-Course R-Track System



**Business Analysis with R**  
(DS4B 101-R)

**Data Science For Business with R**  
(DS4B 201-R)

**R Shiny Web Apps For Business**  
(DS4B 102-R)

**Project-Based Courses with Business Application**

Data Science Foundations  
**7 Weeks**

Machine Learning & Business Consulting  
**10 Weeks**

Web Application Development  
**4 Weeks**



**DS4B 101-R: Business Analysis With R**  
Your Data Science Journey Starts Now! Learn the fundamentals of data science for business with the tidyverse.  
 Matt Dancho



**DS4B 201-R: Data Science For Business With R**  
Solve a real-world churn problem with H2O AutoML (automated machine learning) & LIME black-box model explanations using R  
 Matt Dancho



**DS4B 102-R: Shiny Web Applications For Business (Level 1)**  
Build a predictive web application using Shiny, Flexdashboard, and XGBoost  
 Matt Dancho

# Key Benefits

- Fundamentals - Weeks 1-5 (25 hours of Video Lessons)
  - Data Manipulation (dplyr)
  - Time series (lubridate)
  - Text (stringr)
  - Categorical (forcats)
  - Visualization (ggplot2)
  - Programming & Iteration (purrr)
  - 3 Challenges
- **Machine Learning - Week 6 (8 hours of Video Lessons)**
  - **Clustering (3 hours)**
  - **Regression (5 hours)**
  - **2 Challenges**
- Learn Business Reporting - Week 7
  - RMarkdown & plotly
  - 2 Project Reports:
    1. Product Pricing Algo
    2. Customer Segmentation

# Business Analysis with R (DS4B 101-R)



Data Science Foundations  
**7 Weeks**

**Regression (Supervised Learning)**

**Summary:**

**Terminology:**

**Resources:**

**Data Science with R Workflow**

Click the links for Documentation

CS e Cheat Sheet

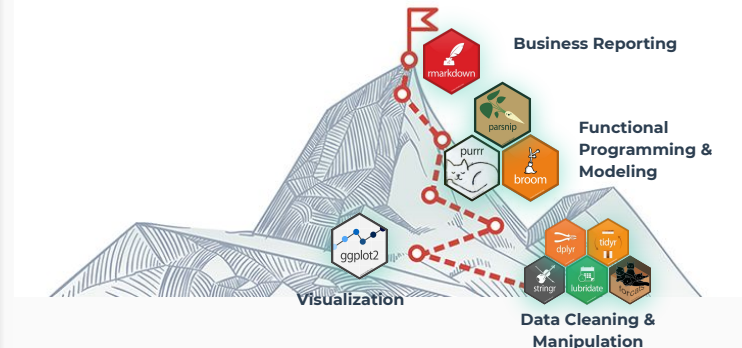
**Important Resources:**

**Segmentation & Clustering**

**Summary:**

Type	Project Address	View	File Path/Name
Clustering	Customer Segmentation	Customer Segmentation	Customer Segmentation
Clustering	Customer Segmentation	Customer Segmentation	Customer Segmentation
Clustering	Customer Segmentation	Customer Segmentation	Customer Segmentation

**Resources:**



**DS4B 101-R: Business Analysis With R**

Your Data Science Journey Starts Now! Learn the fundamentals of data science for business with the tidyverse.

Matt Dancho



# Key Benefits

## End-to-End Churn Project

Understanding the Problem &  
Preparing Data - Weeks 1-4

- Project Setup & Framework
- Business Understanding / Sizing Problem
- Tidy Evaluation - rlang
- EDA - Exploring Data - GGally, skimr
- Data Preparation - recipes
- Correlation Analysis
- 3 Challenges

## Machine Learning - Weeks 5, 6, 7

- H2O AutoML - Modeling Churn
- ML Performance
- LIME Feature Explanation

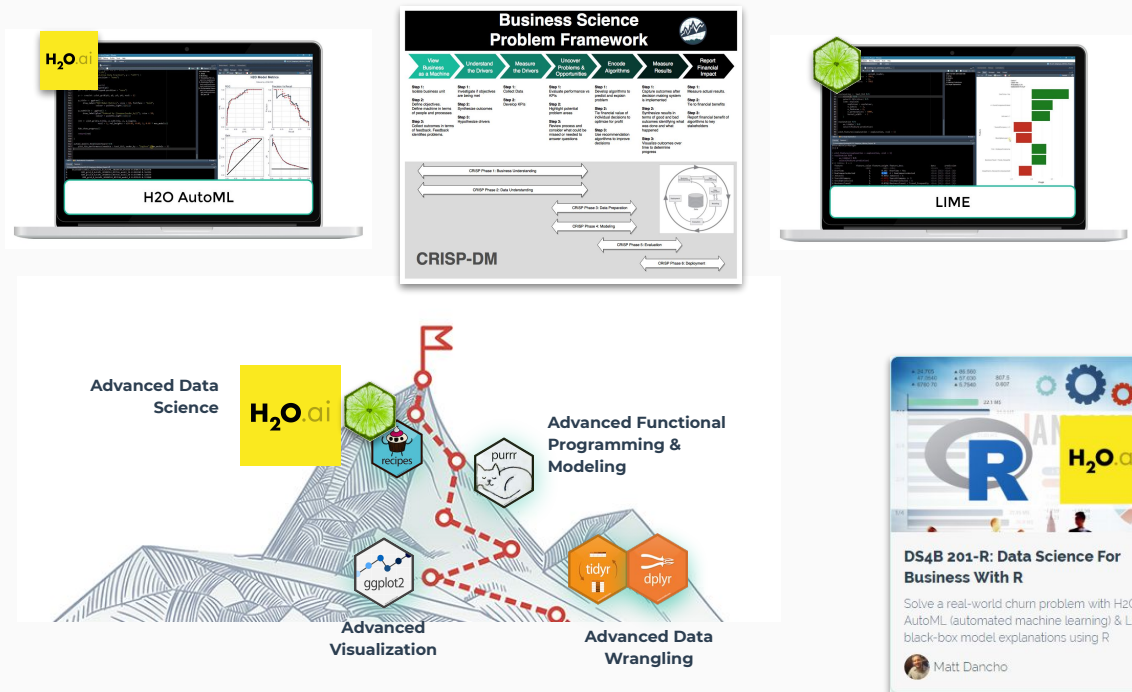
Return-On-Investment - Weeks 7, 8, 9

- Expected Value Framework
- Threshold Optimization
- Sensitivity Analysis
- Recommendation Algorithm

# Data Science For Business (DS4B 201-R)



Machine Learning & Business Consulting  
**10 Weeks**



# Key Benefits

Learn Shiny & Flexdashboard

- Build Applications
- Learn Reactive Programming
- Integrate Machine Learning

## App #1: Predictive Pricing App

- Model Product Portfolio
- XGBoost Pricing Prediction
- Generate new products instantly

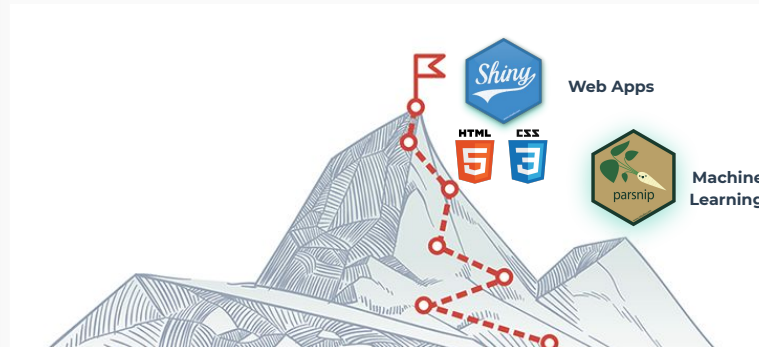
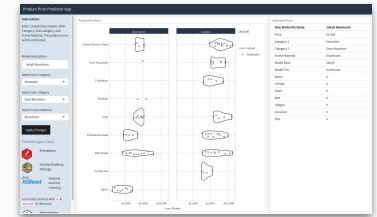
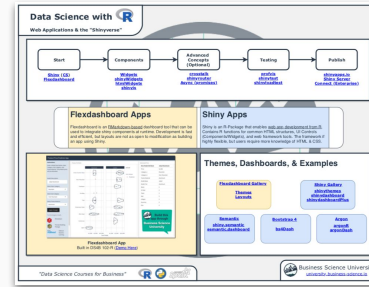
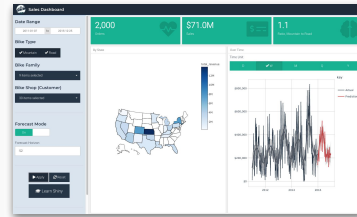
## App #2: Sales Dashboard with Demand Forecasting

- Model Demand History
- Segment Forecasts by Product & Customer
- XGBoost Time Series Forecast
- Generate new forecasts instantly

# Shiny Apps for Business (DS4B 102-R)



Web Application Development  
**4 Weeks**



**DS4B 102-R: Shiny Web Applications For Business (Level 1)**

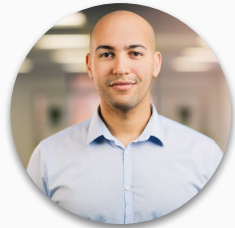
Build a predictive web application using Shiny, Flexdashboard, and XGBoost

Matt Dancho



*“Your program allowed me to cut down to **50% of the time** to deliver solutions to my clients.”*

*-Rodrigo Prado, Managing Partner Big Data Analytics & Strategy at Genesis Partners*



*“I can already **apply** a lot of the early gains from the course to current working projects.”*

*-Adam Mitchell, Data Analyst with Eurostar*




*“My work became **10X easier**. I can spend quality time asking questions rather than wasting time trying to figure out syntax.”*

*-Mohana Chittor, Data Scientist with Kabbage, Inc*

Achieve  
**Results** that  
Matter to  
the  
**Business**

# PROMO Code: **learninglabs**



**Bundle - DS For Business + Web Apps (Level 1): R-Track - Courses 101, 102,**

---

3 Course Bundle **0%**  
COMPLETE



**DS4B 101-R: Business Analysis With R**

Your Data Science Journey Starts Now! Learn the fundamentals of data science for business with the tidyverse.


 Matt Dancho



**DS4B 102-R: Shiny Web Applications For Business (Level 1)**


Build a predictive web application using Shiny, Flexdashboard, and XGBoost.

 Matt Dancho



**DS4B 201-R: Data Science For Business With R**

Solve a real-world churn problem with H2O AutoML (automated machine learning) & LIME black-box model explanations using R.

 Matt Dancho

## **R-TRACK BUNDLE**

MSRP: ~~\$234/mo~~

**6 Low Monthly Payments**

**\$199/mo**

Save: \$35/mo

# Begin Learning Today

[university.business-science.io](https://university.business-science.io)

