Code Sample Workshop

April 26, 2019

```
numbers <- 1:3
words <- c("word1", "word2", "word3")
categories <- as.factor(words)
dtfrm <- data.frame(numbers, words)

attr(numbers, "label")    <- "A numeric vector"
attr(words, "label")      <- "A character vector"
attr(categories, "label") <- "A factor vector"

list1 <- list(dtfrm = dtfrm, y = numbers)
list2 <- list(list
list2$`name with space`
list2$`2` <- c("one", "two")
list3 <- list(abc = categories, list1 = list1)
rm(list1)
```

```
.GlobalEnv | Libraries

   categories A factor vector
   dtfrm
   ├─ numbers
   └─ words
   list2
   ├─ list1
   │  ├─ dtfrm
   │  │  ├─ numbers
   │  │  └─ words
   │  └─ y        A numeric vector
   ├─ abc         A character vector
   ├─ name with space
   └─ 2
```

```
Example.R [+]                    1,1          All
Object_Browser    8,1          Top
```

```
> list1 <- list(dtfrm = dtfrm, y = numbers)
> list2 <- list(list1 = list1, abc = words)
> list2$`name with space` <- 1:10
> list2$`2` <- c("one", "two")
> list3 <- list(abc = categories, list1 = list1)
> rm(list1)
> source('/home/jakron/src/0/im-R-plugin/r-plugin/vimbrowser.R') ; .vim_browser()
```

Outline

- General Tips
- R script submissions
- GitHub submissions

General Tips

A coder’s worst nightmare...

“Please submit a coding sample along with your CV/resume…”

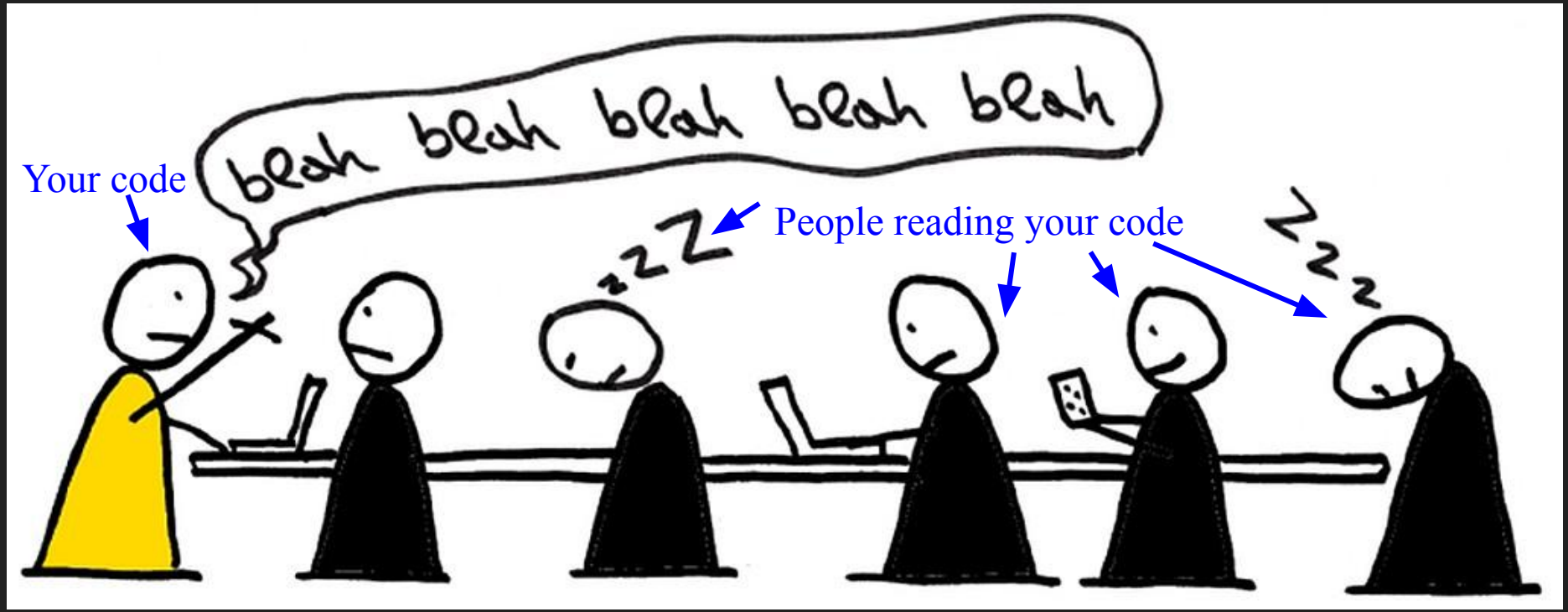
What do I do???

Disclaimer: We're going over general guidelines.
Code samples may be job-dependent.

Some Guidelines

1. Pick something you like and are proud of

We've all been there...



Job Description & Your Code Sample

Job description call for lots of cleaning? **Submit a cleaning script!**

Job description call for lots of data analysis? **Submit an analysis script!**

Job description is very vague? **Submit something that highlights your strengths!**

Some Guidelines

1. Pick something you like and are proud of
2. Err on the side of longer code

Longer code = more to show

BUT not unnecessarily long

A long script should still have coherence

Which leads us to….

Some Guidelines

1. Pick something you like and are proud of
2. Err on the side of longer code
3. Comment the code well

Know your audience

Shares
specific knowledge

Doesn’t share
specific knowledge

User-written functions

- ☑ 1-2 sentences on purpose
- ☑ Inputs
- ☑ Outputs

Example

```
assignType = function(net){

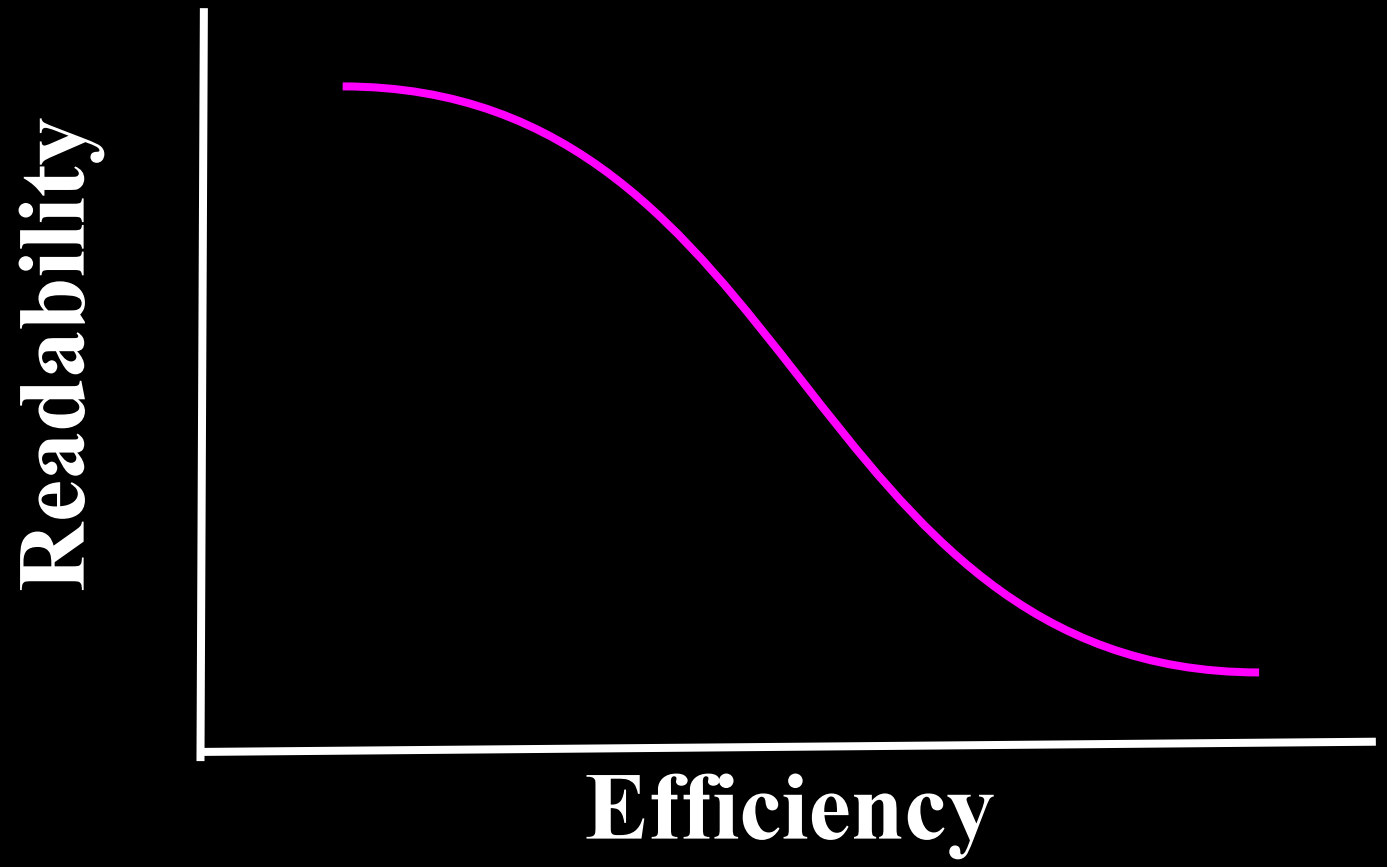
  ## This function assigns type attribute - cooperator or defector
  ## All nodes are defectors except for one random cooperator
  ## @net = igraph network input
  ## assign network with assigned type
```

Example from: https://github.com/biabbiassago/cooperation_evolution/ (Bianca's code)

Better to over-comment than under-comment

Some Guidelines

1. Pick something you like and are proud of
2. Err on the side of longer code
3. Comment the code well
4. Efficiency vs Readability???


Readability
Efficiency

Clear code > Efficient code

But still make sure your code is efficient where it should be

Some Guidelines

1. Pick something you like and are proud of
2. Err on the side of longer code
3. Comment the code well
4. Go for efficiency -- don’t pick something overly complex
5. Make sure it’s reproducible

Reproducibility

- Relative file paths

```
path <-"Data/Clean/"
```

Reproducibility

- Soft-code

```
for(i in 1:length(n))
```

VS

```
for(i in 1:41)
```

Reproducibility

- Provide data

  If you can’t share data, provide “fake data”

Script Submission

Structure Checklist

- [x] 1-2 sentences on purpose
- [x] Load libraries in beginning*
- [x] Comment start of sections

*Unless there's an issue with a package overwriting the functions of another package that you need later

Structure: Purpose

```
### Introduction

The goal of this code file is to generate an imputed data set
```

```
# *This R script is for combining the datasets on the English Premier League results from 2000/2001
#  to 2017/2018. Data for the current season will be updated weekly.
```

```
## Introduction

In this project, I look at clustering careers from the National Longitudinal Survey of Youth (NLSY),
```

Examples from https://github.com/ZarniHtet13/Asynch_Longitudinal_Mirror (Zarni's code) & https://github.com/CClingain/EPL_Data (Clare's code) & https://github.com/kaushik12/clustering-project (Kaushik's code)

Load libraries in beginning

```
suppressPackageStartupMessages(require(httr))
suppressPackageStartupMessages(require(gtfsway))
suppressPackageStartupMessages(require(lubridate))
suppressPackageStartupMessages(require(tidyr))
```

```
#### R Libraries

This block has all the *required* libraries for this code file.

```{r, R.options=FALSE, warning=FALSE, message=FALSE}
#For the dta raw files
library(foreign)
#For importing different types of data set without specification
library(rio)
#For processing long form data
library(dplyr)
#GTools library for ordering numeric variables
library(gtools)
#For filling NA values
library(tidyr)
#Loading Rmarkdown library for rendering
library(rmarkdown)
#knitr library for rendering
library(knitr)
#for missing data
library(mi)
#Sourcing the code file
source("../zarni/01b_Function_LInterpolation.R")
```
```

Examples from
https://github.com/ZarniHtet13/Asynch_Longitudinal_Mirror
(Zarni's code) &
https://github.com/EDSP19/edsp2019project-CClingain
(Clare's code)

Comment start of sections

```
#### I: Uploading Raw data
```

```
#### II: Data Exploration
```

```
#join with penal codes
joined <- left_join(stat.is, pl, by = c("law_code"))

#find failed ones, reason: some have extra 0s at the end
failed <- joined[is.na(joined$PDCODE_VALUE) & is.na(joined$LIT_LONG) & is.na(joined$CATEGOR
failed <- unique(failed$law_code)

#for those that failed originally, remove the padded 0s, rematch with the penal law codes
joined <- joined %>%
  mutate(law_code = ifelse(law_code %in% failed, gsub("00$", "", law_code), law_code)) %>%
  left_join(pl, by = c("law_code")) %>%
  select(-CATEGORY.x, -LIT_LONG.x, -LIT_SHORT.x, -X, -PDCODE_VALUE.x)
```

Examples from https://github.com/ZarniHtet13/Asynch_Longitudinal_Mirror (Zarni's code) & https://github.com/madisonvolpe/drugcrimepolicing_nyc (Madison & Frankie's code)

GitHub Submission

You may be asked to submit your GitHub instead

Prepare a project that you can pin to your repository!

I may be slow to respond.

Zarni Htet

ZarniHtet13

Unfollow

Block or report user

PRO

Overview | Repositories 30 | Projects 0 | Stars 41 | Followers 24 | Following 88

Pinned

Asynch_Longitudinal_Mirror

This repository highlights my ongoing work with Professor Marc Scott and Professor Daphna Harel in exploring the relationship between BMI and Media Exposure in infants and toddlers.

R

ArlingtonProbabilisticLinkage

This repository highlights some of my work in deduplicating administrative data of Department of Human Services in Arlington as a summer data science fellow at Social Decision Analytic Lab in 2017.

R

WashingtonMetro

This repository highlights some of my work in tackling fare evasion for the Washington Metro as a summer data science fellow at Social Decision Analytic Lab in 2017.

R

Pinned repositories

All things are easier with a guide!

backpack

One package to rule them all, one package to find them, one package to bring them all and in the darkness bind them!

Easily discover, collect, and learn to use packages for statiscal modeling. Use built-in binders (collections) of packages for topics in statistics. Create your own binders to store collections of packages for easy installing and loading within your projects. Also, explore lessons on multivariate statistical modeling in R.

Instructions

To download this package, ensure you have devtools installed

If not:

```
install_packages('devtools')
```

README Checklist

- [x] Introduce project (Overview)
- [x] Goals
- [x] Data sources
- [x] Directory Structure
- [x] References

Checklist: Introduce project (Overview)

Introduction

The objective of this project to understand the relationship structure between topics in Physics by observing the order in which concepts in Physics are taught by looking at undergraduate physics textbooks.

My motivation for this project comes from the desire to understand how people learn, particularly the order of the topics in their learning process. If we know how different people learn, maybe we can provide access to resources that aid learning in a way that is customized to each individual. More specifically, can we provide an order of topics to get from current knowledge to a learning objective. Think Google Maps for learning.

This repository hold the content from tutorials/workshops and other events organised by the Stats Club.

Each event content are located in the respective sub-directory. A brief description of each is presented below

1. `GitHub-Basics` conatins the presentation and a markdown file with useful commands for getting started with GitHub.
2. `RShiny` contains the presentation PDF and the base plot and ggplot code for the two examples in the slides.
3. `SQL-Workshop` contains the presentation PDF for learning SQL basics.
4. `WebScrapingTutorial` contains the presentation and R code to learn basics of web scraping.
5. `ggplot-Workshop` contains the Rmd file with code, explanations, and exercises on using `ggplot`

Examples from https://github.com/EDSP19/edsp2019project-kaushik12 & https://github.com/priism-center/stats-club

Checklist: Goals

Objective

Our goal is to use existing administrative data along with American Community Survey data at the census block level to find a narrative on where the most fare evasion is happening and why it is so.

Example from https://github.com/ZarniHtet13/WashingtonMetro

Checklist: Data sources

Data Sources

1. NYC Annualized Property Sales Data (2012-2017)
2. MapPLUTO (18v1)
3. Geoclient API v1.1
4. Property Assessment Roll Archives
5. NYPD Complaint Data Historic
6. 311 Service Requests from 2010 to Present

We are using the following data sources:

- NYPD Arrest Data YTD ✅
- NYPD Arrest Data Historic ✅
- EMS Incident Dispatch Data ✅
- 311 Service Requests from 2010 to Present 🚧

Examples from https://github.com/tsdataclinic/newerhoods & https://github.com/madisonvolpe/drugcrimepolicing_nyc

Checklist: Directory Structure

Directory structure

The project directory is structured into 10 main folders:

- `src/` includes source code for pipeline, feature analysis, data wrangling for visualizations, and our website report.
- `notebooks/` includes python notebooks for data exploration and analysis with descriptive text in a human readable format.
- `viz/` includes source code for visualizations.
- `docs/` includes documents and reports produced during the fellowship with early results and commentary.
- `dev/` includes development scripts which were used during the early development stages and are still in a rough format.
- `pipeline/` includes the pipeline for the models. The models should be run from this folder.
- `db/` includes SQL and python codes to transform data to formats ready for analysis and organize onto a database.
- `examples/` includes python notebooks of examples of analyses and output.
- `results/` includes final analyses outputs.
- `website/` includes all the website related files

Example from https://github.com/dssg/tuscany-tourism-public

Checklist: References

References

1. ClustGeo: an R package for hierarchical clustering with spatial constraints
2. Making Neighborhoods - Understanding New York City Transitions 2000-2010

Examples from https://github.com/tsdataclinic/newerhoods

Review

- General Tips
- R script submissions
- GitHub submissions