

# Predictive Modeling Blueprint

Casey Quinn and Ahmed Alawami

March 29, 2016

# Predictive Modeling Approach: Overview

1. **Define the Problem**
2. **Understand the Data: Data Types**
3. **Understand the Data: Missing Values**
4. **Model Tuning Metric**
5. **Model Evaluations and Tuning**
6. **Github Collaboration**
7. **General Collaboration**

# Define the Problem

## **Model Complexity Constraints**

- ▶ Offline, Time-averaged, "Instantaneous" Implementation
  - ▶ Computation Resources
  - ▶ Computation Time
  - ▶ Data Handling

## **Model Accuracy and Precision Requirements**

- ▶ "Exact" vs "Rough" Predictions

# Understand the Data: Data Types

## Continuous

- ▶ Independent Observations?
  - ▶ Check for Autocorrelation
- ▶ Exact vs. Classification Ranges

## Date/Time

- ▶ Numeric
- ▶ Factor

## Classification

- ▶ Reduce factor lists as needed
- ▶ Variable combinations if many variables

# Understand the Data: Missing Values

## Find Them

- ▶ VIM: aggr function

## Deal with Them

### ▶ Removal

- ▶ Entire Variable
- ▶ Individual Observations

### ▶ Imputation

- ▶ VIM package
- ▶ mice package

### ▶ Set as unique classification

### ▶ Leave as is

### ▶ Handling test/prediction missing data

# Model Tuning Metric Considerations

## Outcome Type

- ▶ Continuous
  - ▶ Prediction
- ▶ Classification
  - ▶ Probabilities
  - ▶ Discrete Outcome

## Data Shape

- ▶ Normal
- ▶ Skewed

# Model Evaluations

## Methods

- ▶ k-Fold Cross Validation
- ▶ Monte Carlo
- ▶ Bootstrap
- ▶ Data Simulation

## Avoid Overfitting

**Reduce Model Size / Number of Parameters**

# Github Collaboration

1. Chose a team leader.
2. Team leader creates a local folder with a descriptive name for the project.
3. Team leader creates a local RProject with the same name as the folder.
4. Team leader creates a Github repository with the same name as the RProject (Don't initialize with README).

Search GitHub Pull requests Issues Gist

Create a new repository  
A repository contains all the files for your project, including the revision history.

Owner Repository name  
csq3555 /

Great repository names are short and memorable. Need inspiration? How about fluffy-octo-spoon.

Description (optional)

☒ Public  
Anyone can see this repository. You choose who can control.

☐ Private  
You choose who can see and control this repository.

☒ Initialize this repository with a README  
This will let you immediately clone this repository to your computer. Skip this step if you're importing an existing repository.

Add a license: None Add a license: None

Create repository

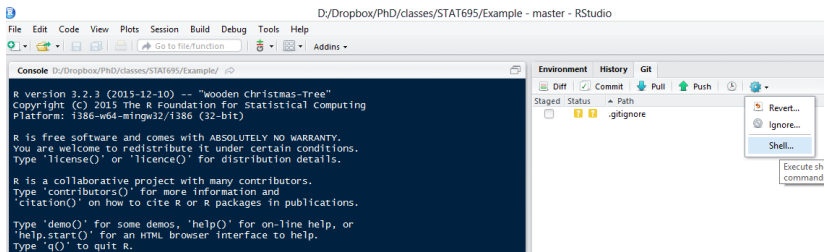
© 2015 GitHub, Inc. Terms Privacy Security Contact Help Status API Training Shop Blog About



# Github Collaboration (cont.)

## 5. Push an existing repository from RStudio Shell.

- ▶ `git remote add origin https://github.com/cwq9999/Example.git`
- ▶ `git push -u origin master`



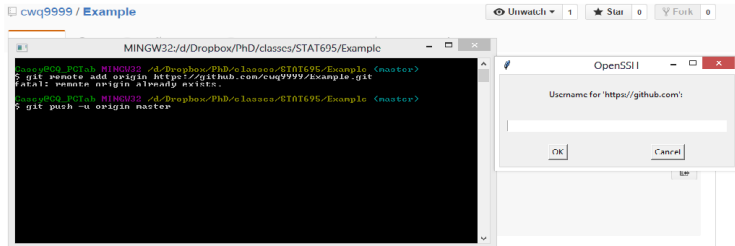
The screenshot shows the RStudio IDE. The title bar indicates the file path: `D:/Dropbox/PhD/classes/STAT695/Example - master - RStudio`. The menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Tools, and Help. The toolbar has icons for file operations and a 'Go to file/function' search bar. The console window displays the R version 3.2.3 (2015-12-10) startup message and the R license text. The Git pane on the right shows the 'Diff' tab with a commit icon, and the 'Staged' section is empty. The 'Path' section shows `.gitignore`. A context menu is open over the Git pane, showing options: 'Revert...', 'Ignore...', 'Shell...', and 'Execute sh command'.

```
R version 3.2.3 (2015-12-10) -- "wooden Christmas-Tree"
Copyright (C) 2015 The R Foundation for Statistical Computing
Platform: i386-w64-mingw32/i386 (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
```



The screenshot shows a terminal window titled `MINGW32:d:/Dropbox/PhD/classes/STAT695/Example`. The terminal output shows the user `cwq9999` at the `PC1ab` machine, running the command `git remote add origin https://github.com/cwq9999/Example.git`. The output shows a fatal error: `FATAL: remote origin already exists.` The user then runs `git push -u origin master`. An `OpenSSH` dialog box is open, asking for the 'Username for 'https://github.com:'. The dialog has 'OK' and 'Cancel' buttons.

```
cwq9999@PC1ab MINGW32: /d:/Dropbox/PhD/classes/STAT695/Example <master>
$ git remote add origin https://github.com/cwq9999/Example.git
fatal: remote origin already exists.
cwq9999@PC1ab MINGW32: /d:/Dropbox/PhD/classes/STAT695/Example <master>
$ git push -u origin master
```

# Github Collaboration (cont.)

## 6. Add team members as collaborators.

- ▶ Collaborators can use Pull/Push buttons under the Git tab.
- ▶ No need to fork the repo or request pull requests.

The screenshot shows the GitHub interface for a repository named 'Example' by user 'cwq9999'. The 'Settings' tab is active, and the 'Collaborators' sub-tab is selected. The page indicates that no collaborators have been added yet and provides a search bar to add new ones. The search bar contains the text 'geanders'.

Options

**Collaborators**

Branches

Webhooks & services

Deploy keys

**Collaborators** Push access to the repository

This repository doesn't have any collaborators yet. Use the form below to add a collaborator.

Search by username, full name or email address

geanders

Add collaborator

# General Collaboration: Team Efficiency (My two cents)

1. Select a team leader
2. Quick Individual Exploratory Data Analysis
3. Initial Exploratory Data Analysis Discussion(s)
  - ▶ Define Training/Test Datasets
  - ▶ Determine Outcome Requirements
  - ▶ Determine Metric(s)
4. Divide into Specialty Groups
  - ▶ General Linear Models
  - ▶ Trees
  - ▶ Support Vector Machines
  - ▶ etc.
5. Present and Compare Models
6. Select Model Type and Refine as a Group or Subgroups

# General Collaboration: Loading Packages (Require vs Library)

## Library

- ▶ Loads a Library and causes an **Error** if Library doesn't exist.
- ▶ Best option for collaboration.

## Require

- ▶ Tries to load a Library and provides a **warning** if library doesn't exist.
- ▶ Not a good option for collaboration.
- ▶ Primarily designed for functions.

# General Collaboration: Relative vs Absolute Pathnames

```
#Set working directories
my_dir1 <- "C:/Docs/Dropbox/PhD/classes/STAT695/BikeShare"
#my_dir2 <- ""

if(dir.exists(my_dir1)){
  setwd(my_dir1)
}#else if(dir.exists(my_dir2)){
#  setwd(my_dir2)
```

# General Collaboration: RProjects and Dropbox

- ▶ Continuously syncs to Dropbox.
- ▶ Remember to close your Rproject.
- ▶ Otherwise you won't be able to sync your Dropbox
- ▶ Better to have separate Projects for each collaborator.

# General Collaboration: Operating Systems

- ▶ Be wary of character differences between operating systems and potentially even versions of R or RStudio.
- ▶ The "~" has been an issue in the past.

# Rmarkdown....

- ▶ It's challenging and takes time to learn.
- ▶ Rmarkdown formatting doesn't work very well!
  - ▶ You can't even bold highlight a bullet list item with the **\*\*\*** command.
  - ▶ If you know HTML or LaTeX don't bother with RMarkdown formatting.



# References

- ▶ Kuhn. Applied Predictive Modeling
- ▶ [www.github.com](http://www.github.com)
- ▶ <http://yihui.name/en/2014/07/library-vs-require/>