Class 5: Group Data Manipulation and Introductory Machine Learning Terminology

MGSC 310

Prof. Jonathan Hersh

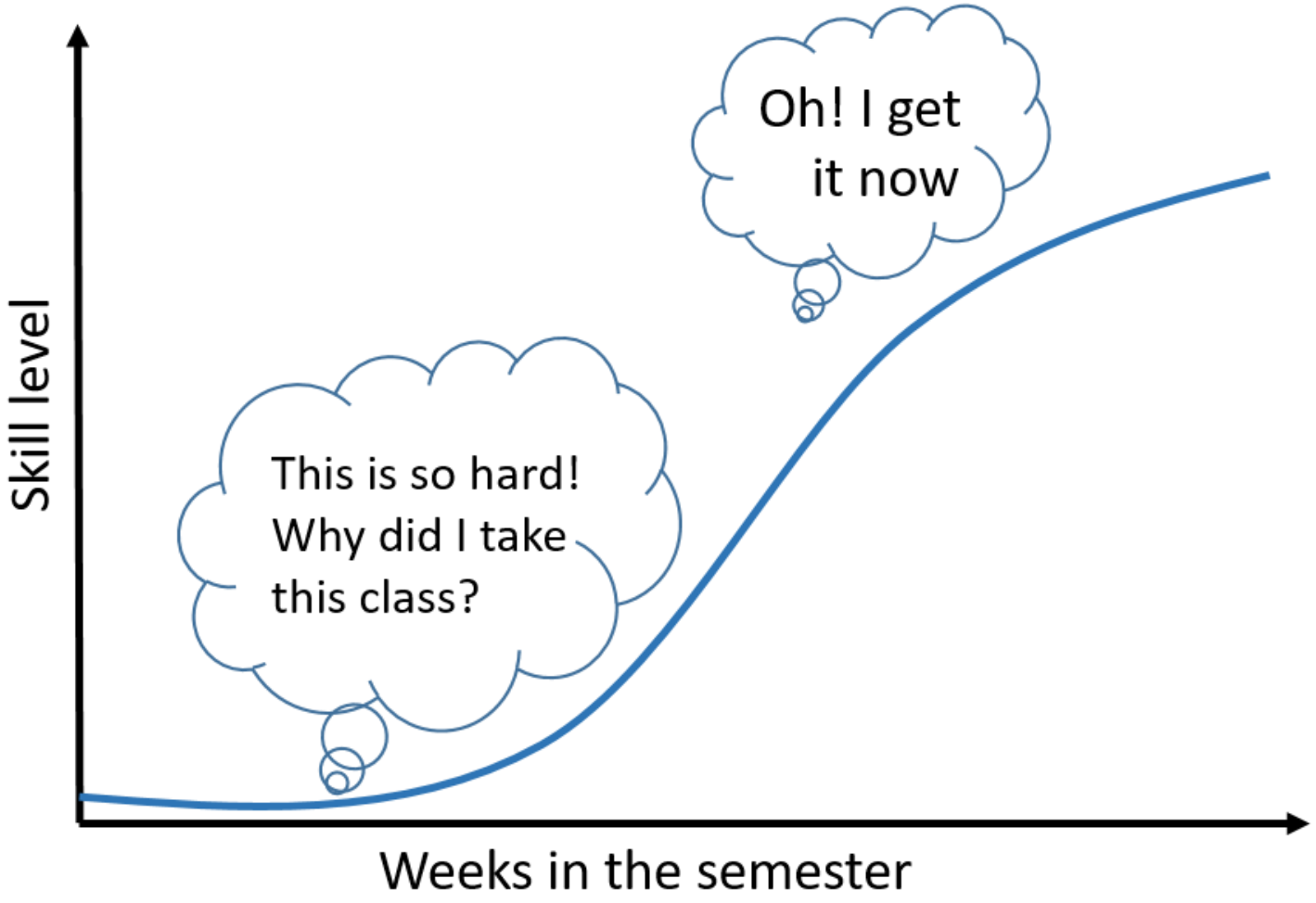
Class 5: Announcements

1. Problem Set 1 – Due Sept 15
   - Must submit compiled HTML file using Rmarkdown
2. TA Office Hours: Tuesdays: 5:30 – 7, Thursdays: 12:30-2; Mondays: 5-6:30
3. Quiz 2 posted, due Thursday @ midnight

the course reading (ISLR pp 15-36 due today)

5. Data Analytics Accelerator Info Sesh Oct 5 @ 11am

I apologize but we are at peak difficulty



Data Analytics Industry Week

Register on Handshake to get access to the following virtual events!

Data Analytics Accelerator Program Info Session
Monday, October 5 | 11 a.m. PST
Interested in pursuing a career in the growing field of data analytics? The Argyros School of Business is proud to present the new career skills-focused Analytics Accelerator Program. Learn more about what hard skills are needed to land a successful career in data analytics. Hear from Professor Toplansky and Dr. Hersh about how you can propel your success and prepare for 21st Century jobs that pay a premium.

Careers in Data Analytics
Tuesday, October 6 | 12 p.m. PST
Hear from the renowned authors of Build a Career in Data Science, Jacqueline Nolis and Emily Robinson about careers in data analytics.

Data Analytics Industry Panel
Thursday, October 8 | 4:30 p.m. PST
This data analytics panel will feature industry experts in analytics from entertainment, healthcare, technology, and more.

Entertainment Analytics: Turning Data Into Insights
Friday, October 9| 12 p.m. PST
Come see a live demo and learn about turning data into actionable insights in Entertainment Analytics with Andre Vargas Head of the data department at leading entertainment and sports agency, Creative Artists Agency (CAA).

CHAPMAN UNIVERSITY | Argyros School of Business and Economics

May Use Problem Set Rmarkdown Template

Problem Set 1 (R Programming)

Published Edit

See the problem set 1 instructions here MGSC310_pset1.pdf, MGSC310_pset1.html

You might find it useful to use the RMarkdown template available RMarkdown_Pset_Template.Rmd

Datasets:

- IMDB_movies.csv
- IMDB_movies.txt

Points	30
Submitting	a file upload

May Use Problem Set Rmarkdown Template

F:/Dropbox/Chapman/Teaching/MGSC_310 - RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

RMarkdown_Pset_Template.Rmd | RMarkdown_Intro_Help.Rmd

Knit Insert Run

```
43 ```
44
45 ## Question 1
46
47 1a) Response to part a.
48
49 ```{r}
50
51 # code for part a
52
53 ```
54
55
56 1b) Response to part b.
57
58 ```{r}
59
60 # code for part b
61
62 ```
63
64 etc...
65
66
67 ## Question 2
68
69 2a) Response to part a.
70
71
72 ```{r}
73
74 # code for 2a
```

56:25 Question 1 R Markdown

Class 5: Outline

1. Qs from last week?
2. **Basic Data Analysis**
   - Missing values
   - Loops
   - mutate to transform variables
   - Remove duplicates with distinct
   - Outputting “clean” data file”
3. **Data Analysis by Groups**
   - group_by() function
   - summarize() to create group variables
4. Data Analysis Lab Class 5
5. Introductory Machine Learning Concepts

Missing Values

```r
# ---------------------------------------------
# MISSING VALUES are values that are unknown in your dataset
# ---------------------------------------------
# R stores missing values as as NAs
is.na(NA)
1 > NA
1 + 1 == NA
NA == NA
y <- NA
y
x <- 1
y == x
```

lab_class_4_R_Exploratory_Data_Analysi... | movies

Filter

	actor_1_facebook_likes	gross	genres	actor_1_name
	11000	200074175	Action|Adventure|Thriller	Christoph Waltz
	27000	448130642	Action|Thriller	Tom Hardy
	131	NA	Documentary	Doug Walker
	640	73058679	Action|Adventure|Sci-Fi	Daryl Sabara
	24000	336530303	Action|Adventure|Romance	J.K. Simmons
	799	200807262	Adventure|Animation|Comedy|Family|Fantasy|Musical|...	Brad Garrett

Loops in R

```
# ---------------------------------------------
# LOOP through numbers using the FOR loop
# ---------------------------------------------

# for loops are created using the synthax
# for(i in start:end){
# do something with i
# }
```

```
> for(i in 1:10){
+   print(i)
+ }
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
[1] 6
[1] 7
[1] 8
[1] 9
[1] 10
```

LOOP through numbers using the FOR loop

```
# how to see how many missings you have in each column?
# well, we want to sum through every column using a for loop
# then print the variable name using names(movies[i])
# then print the sum of is.na() for just that variable

# for each column in the movies
for(i in 1:ncol(movies)){

  # print the following
  print(

    # first print "Variable: "
    paste0("Variable: ",

           # then print the variable name, then "NAs: "
           names(movies)[i], " NAs: ",

           # then print the sum of the number of missing values
           # for that variable
           sum(is.na(movies %>% select(i)))
          )

        )
}
```

Functions in R

```r
# ---------------------------------------------
# Creating functions
# ---------------------------------------------
# we create a function in R by writing
# function_name <- function(input1, input2,...){
  # function arguments
# }

print_names <- function(data_frame){
  print(names(data_frame))
}
```

```
> print_names(movies)
 [1] "color"                    "director_name"            "num_critic_for_reviews"
 [4] "duration"                 "director_facebook_likes"  "actor_3_facebook_likes"
 [7] "actor_2_name"             "actor_1_facebook_likes"   "gross"
[10] "genres"                   "actor_1_name"             "movie_title"
[13] "num_voted_users"          "cast_total_facebook_likes" "actor_3_name"
[16] "facenumber_in_poster"     "plot_keywords"            "movie_imdb_link"
[19] "num_user_for_reviews"     "language"                 "country"
[22] "content_rating"           "budget"                   "title_year"
[25] "actor_2_facebook_likes"   "imdb_score"               "aspect_ratio"
[28] "movie_facebook_likes"
```

Build a function that prints number of missing values for each variable

```r
# Let's take the code we wrote above and translate
# it to a function called "num_missing".
# We can then call the function and pass our movies dataframe
# to it to export
num_missing <- function(data_frame){
  for(i in 1:ncol(movies)){
    print(
      paste0("Variable: ",
             names(movies)[i], " NAs: ",
             sum(is.na(movies %>% select(i)))
      )

    )
  }
}
```

```
> num_missing(movies)
[1] "Variable: color NAs: 0"
[1] "Variable: director_name NAs: 0"
[1] "Variable: num_critic_for_reviews NAs: 50"
[1] "Variable: duration NAs: 15"
[1] "Variable: director_facebook_likes NAs: 104"
[1] "Variable: actor_3_facebook_likes NAs: 23"
[1] "Variable: actor_2_name NAs: 0"
[1] "Variable: actor_1_facebook_likes NAs: 7"
[1] "Variable: gross NAs: 884"
[1] "Variable: genres NAs: 0"
[1] "Variable: actor_1_name NAs: 0"
[1] "Variable: movie_title NAs: 0"
[1] "Variable: num_voted_users NAs: 0"
[1] "Variable: cast_total_facebook_likes NAs: 0"
[1] "Variable: actor_3_name NAs: 0"
[1] "Variable: facenumber_in_poster NAs: 13"
[1] "Variable: plot_keywords NAs: 0"
[1] "Variable: movie_imdb_link NAs: 0"
[1] "Variable: num_user_for_reviews NAs: 21"
[1] "Variable: language NAs: 0"
[1] "Variable: country NAs: 0"
[1] "Variable: content_rating NAs: 0"
[1] "Variable: budget NAs: 492"
[1] "Variable: title_year NAs: 108"
[1] "Variable: actor_2_facebook_likes NAs: 13"
[1] "Variable: imdb_score NAs: 0"
[1] "Variable: aspect_ratio NAs: 329"
[1] "Variable: movie_facebook_likes NAs: 0"
```

MUTATE to Transform variables in your dataset

```r
# ----------------------------------------------
# MUTATE to Transform variables in your dataset
# ----------------------------------------------

# adding new variables using mutate()
# note %<>% == DF <- DF %>%
# let's create new varibles budgetM and grossM that
# are budget and gross in units of millions
movies %<>% mutate(budgetM = budget/1000000,
                   grossM = gross/1000000,
                   profitM = grossM - budgetM)

movies %>% glimpse()

# so it looks like there's some outliers
# The most expensive movie ever made was Pirates of
# the Caribbean: On Stranger Tides
# which cost $387.8m. Any movies with a budget higher
# than this must be a data anomaly

# Let's use the filter command to remove these

movies_clean <- movies %>% filter(budgetM < 400)
```

Find Duplicate Rows with duplicated()

```
# ----------------------------------------------
# Find Duplicate Rows with duplicated()
# and find_duplicates() (must install hablar package)
# ----------------------------------------------
# number of duplicated rows
movies %>% duplicated() %>% sum()

# view duplicated rows
# install.packages(hablar)
movies %>% hablar::find_duplicates()
```

Output final clean version of dataset

```r
# ------------------------------------------------
# Output final clean version of dataset
# ------------------------------------------------
# remove duplicate rows, create new budget and gross variables,
# rename director and title
# remove budgets greater than 400M,
# order title, year, budget, director and gross first, then store in new file
movies_clean <-
  movies %>%
  distinct() %>%
  mutate(budgetM = budget/1000000,
         grossM = gross/1000000,
         profitM = grossM - budgetM) %>%
  rename(director = director_name,
         title = movie_title,
         year = title_year) %>%
  relocate(title, year, country, director, budgetM, grossM, imdb_score) %>%
  filter(budgetM < 400)

movies_clean %>% glimpse()
```

- Generally we do pre-processing on our dataset starting from a raw file.
- After these transformations we save a “clean” version of the dataset that is used for analysis

Create summary statistics by GROUP using group_by()

```r
# ---------------------------------------------
# Create summary statistics by GROUP using group_by()
# ---------------------------------------------
# group summaries using summarise and group_by
director_avg <-
  movies_clean %>%
    # group_by() is used to indicate the grouping variable
  group_by(director) %>%

    # summarize creates a new variable based on this group
    # here we create averages by director using the 'mean'
    # function
  summarize(gross_avg_director = mean(grossM, na.rm = TRUE))


# view results
director_avg %>% arrange(-gross_avg_director) %>% print()
```

Create averages, count and standard deviation by groups

```
# ------------------------------------------------
# Create grouped variables using the Summarize function
# n() creates counts by
# sd() creates standard deviations
# ------------------------------------------------
# let's create budget by director, gross by director, profit by director,
# number films by director
director_df <-
  movies_clean %>%
  group_by(director) %>%
  summarize(

      # create average budget by director
      budget_avg_director = mean(budgetM, na.rm = TRUE),
      # create average gross by directory
      gross_avg_director = mean(grossM, na.rm = TRUE),
      # create average movie profit by director
      profit_avg_director = mean(profitM, na.rm = TRUE),
      # create variable that lists number of films
      # by director
      num_films = n(),
      # create a standard deviation of profit
      # by director
      profit_sd_director = sd(profitM, na.rm = TRUE)

      )
```

Exercises - 2

1. Print a dataframe with the film director name, and number of films for the 10 directors with the most films in the dataset
2. What movie genres have the highest average profit? (hint, must use a new group_by() command)
3. Print a dataframe with George Lucas' average budget, gross, profit and number of films
4. Why do some directors have "NA" for profit_sd?

Supervised vs Unsupervised Learning

Supervised Learning:

- For every x_i we observe some y_i
- Ex: random forests to predict loan default (y_i) based on applicant characteristics (x_i)

Unsupervised Learning:

- We only observe x_i
- Ex: clustering loan applicants based on characteristics (x_i)

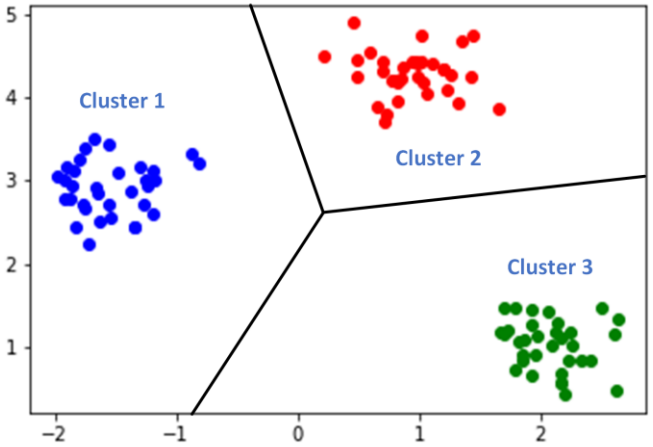
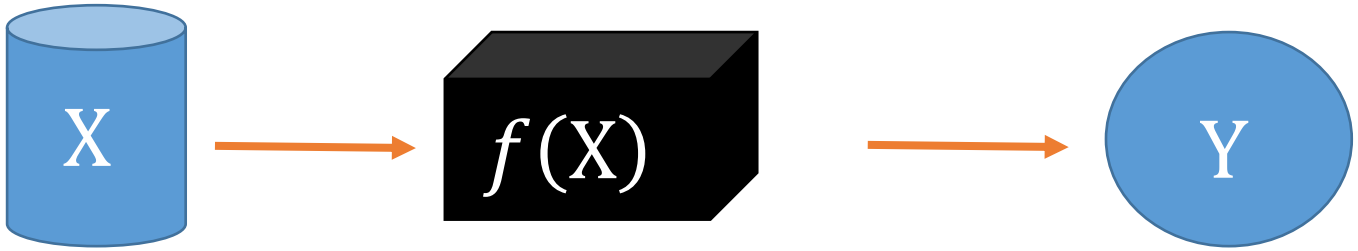


Fig.1. An Example Of Data Clustering

Supervised learning: learning $f(\mathrm{X})$ our predicted out given inputs

$$\mathrm{Y} = f(\mathrm{X}) + \epsilon$$



ϵ = “epsilon” (unexplained portion)

"Estimating" $\hat{f}(X)$

- $\mathrm{Y} = f(\mathrm{X}) + \epsilon$ is the true value
- We can only use data to "guess" at $f(\mathrm{X})$
- We call this guess $\hat{f}(X)$

How do we know when we've selected a "good" $\hat{f}(X)$?

- We reserve a portion of our data into a "test" set, estimate a model on the other part, and see how our model performs on this test set

Testing Training Data Subsets

Training set: (observation-wise) subset of data used to develop models

Testing/Training Split

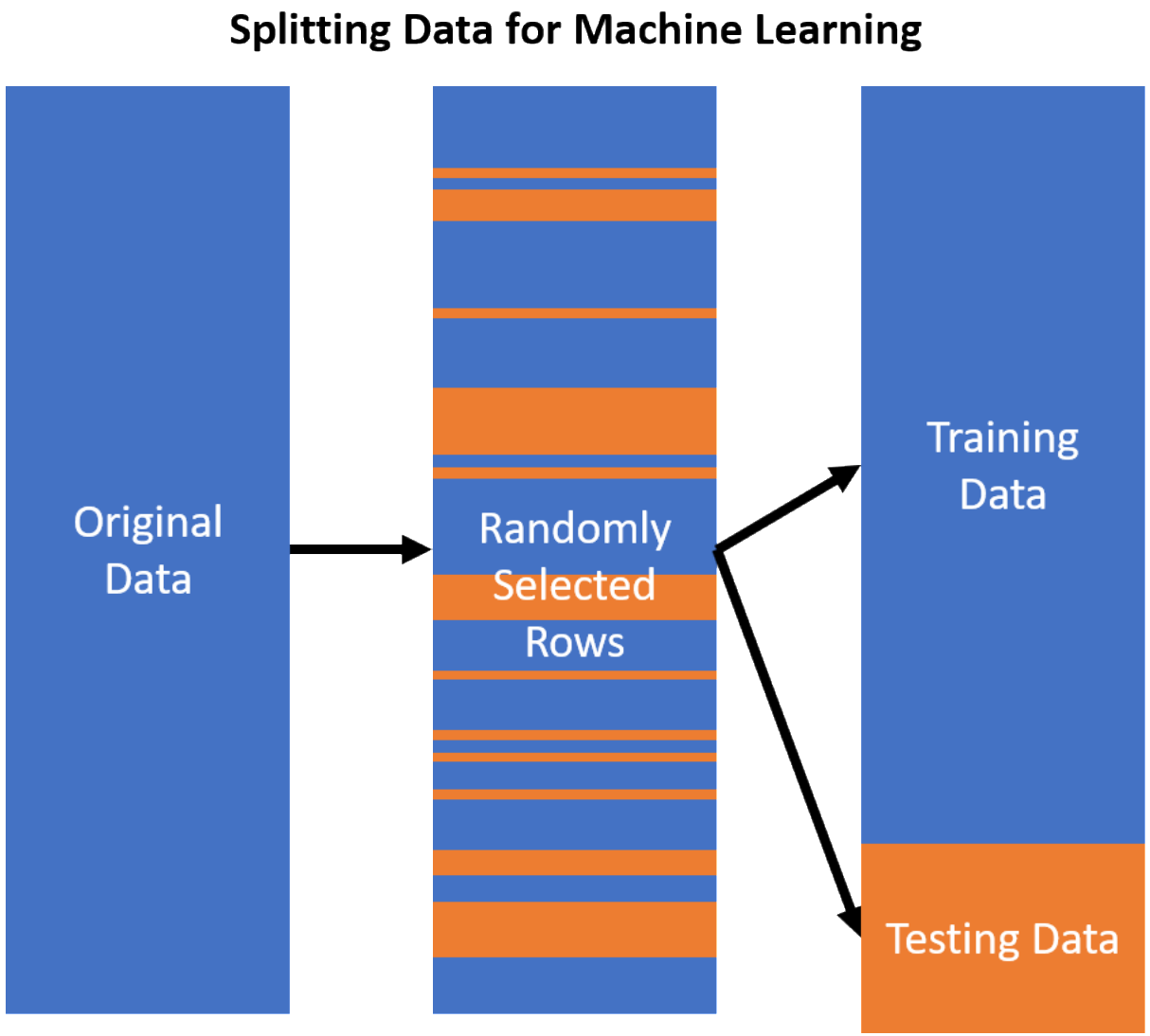
Training set: (observation-wise) subset of data used to develop models

Test set: subset of data used during intermediate stages to “tune” model parameters

Rule of thumb 75% training 25% test -ish

Randomly Selecting Rows for Test or Training Sets

- Observations are randomly selected into either testing or training splits of the data

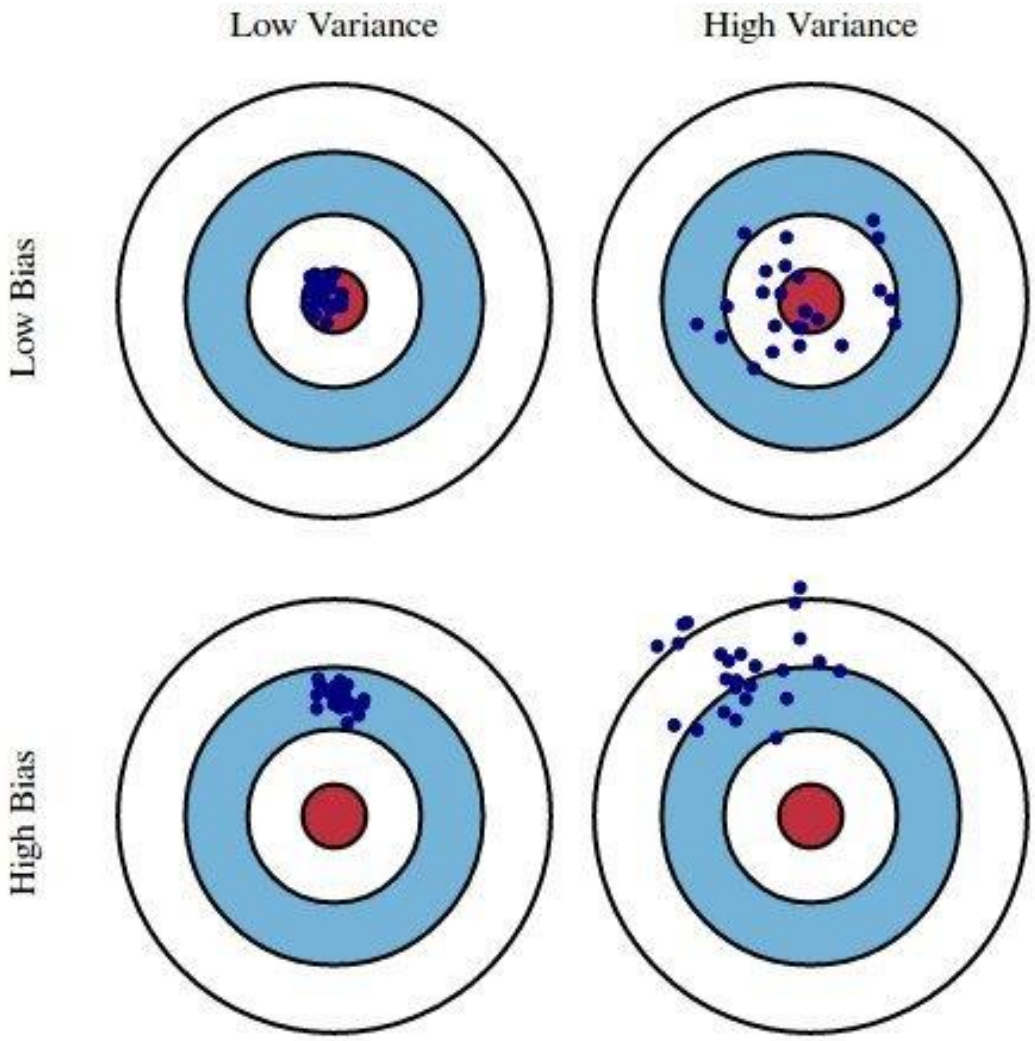


Bias and Variance

<u>Bias</u>: **Tendency of an in-sample statistic to over or under estimate the statistic in the *population***

<u>Variance</u>: **Tendency to noisily estimate a statistic**. E.g., sensitivity to small fluctuations in the training dataset.

Bias-Variance Tradeoff



Class 5 Summary

- **Missing values** (NAs) indicate we don't know the value of a variable for that observation
  - Will need to make assumptions on how to treat these that can influence our results!
- **Functions** create "more readable" code.
  - Any procedure done more than one needs a function
- "**Clean**" version of datasets have been processed and are ready for analysis
- Use **group_by()** and **summarize()** to create statistics by groups (averages, standard deviations)
- **Supervised** models contain a y_i (target/outcome variable) for every x_i (descriptor variables)
- **Unsupervised** models contain only x_i
- **Training** data is the data we will use to estimate our model parameters
- **Testing** data is the data used to evaluate our model performance
  - Never estimate model parameters on the testing data!
- **Bias:** tendency of an in-sample statistic to over or underestimate the true value
- **Variance:** tendency to noisily estimate that statistic