Midterm I Study Guide and Review

Deepak Bastola

April 23 2024

Exam I Study Guide

Format: In Class with open-ended questions.

One-sided Cheat-sheet allowed (A4 paper)

Official `ggplot`, `dplyr`, `tidyr`, `lubridate`, `forcats`, `stringr` cheat-sheet will be provided for reference, if needed. You do not need to bring these with you.

- You are not permitted to use a laptop or classroom computer.

Topics

- The exam covers the following topics introduced in class (through Mon. 04/22):
    1. *Data types.* Understand the basic data types in R, including how to access elements in a list.
    2. *Visualizing Data.* Be able to use principles of visual perception to identify and apply appropriate visual cues in a graphic, and to read and write code to create visualizations of data using `ggplot2`.
    3. *Single and two-table dplyr verbs.* Understand the logic and implementation of the essential data wrangling tasks discussed in class.
    4. *Tidy data principles.* Understand the implementation of tidy data principles including appropriate data import conventions, and long to wide transformations and vice-versa.
    5. *Working with Dates and Times using lubridate.* Basic familiarity with `lubridate` functions for manipulating date-time objects in R.
    6. *Categorical Data Manipulation using forcats.* Understand the basics of handling categorical data using `forcats`, including factors reordering and summarizing.
    7. *String Manipulation using stringr*. Gain proficiency in handling and analyzing text data with stringr, focusing on functions for detecting patterns, performing replacements, and extracting meaningful information from strings, essential for effective text data processing.
- You will be tested on your understanding of the R code we have discussed. I will not make you write extremely complicated code from scratch, but be prepared to write small chunks of code. Additional ways I could assess your understanding of R include (but are not limited to):
    - Filling in missing arguments/lines of code.
    - Identifying the error in written code.
    - Putting lines of code in order to complete a specified task.
    - Describing the output resulting from a code chunk, including dimensions.

Sample Midterm Problems for Practice

1. An introduction to the data

Our data set contains information from the Ames Assessor's Office used in computing assessed values for individual residential properties sold in Ames, IA, from 2006 to 2010. The data contains 82 variables measured on each of 2930 properties, but we will focus on the following 12 variables:

- neighborhood = one of 28 neighborhoods in Ames
- price = sale price in $
- yrbuilt = year of original construction
- yrsold = year house was sold (2006 – 2010)
- sqft = total interior square footage
- garage = size of garage in square feet
- cars = number of cars garage holds
- lotarea = lot size in square feet
- contour = flatness of property (Lvl = level, Bnk = banked, HLS = hillside, Low = depression)
- condition = overall condition of house (10 = very excellent to 1 = very poor)
- kitchen = kitchen quality (Ex = excellent, Gd = good, TA = typical/average, Fa = fair, Po = poor)
- centralair = central air conditioning? (Y or N)

Warning: this solution is very succinct and is intended as a guide. These are not thorough answers to the problems.

What does the following code do?

Provide a thorough and intuitive (2-3 sentences) description of the output from each of the following R chunks. The chunks either produce a new data set or a new plot; if it's a new data set, give the dimensions in addition to your description. Write your descriptions in regular English, without using variable names.

a)

```
ames %>%
  group_by(neighborhood) %>%
  summarise(count = n(),
            medprice = median(price, na.rm = TRUE),
            IQRprice = IQR(price, na.rm = TRUE)
            ) %>%
  arrange(desc(medprice))
```

Answer:

b)

```
ames %>%
  group_by(neighborhood) %>%
  summarise(newvar = mean(contour == "Lvl")) %>%
  top_n(-3, newvar)
```

Answer:

c)

```
ames %>%
  group_by(neighborhood) %>%
  filter(n() > 2) %>%
  top_n(3, price) %>%
  select(neighborhood, price, yrbuilt, sqft, garage, condition) %>%
  arrange(neighborhood, desc(price))
```

Answer:

2. Joining Data

Based on the given tribbles, `df_primary` and `df_secondary`, answer the following questions.

```
library(dplyr)
df_primary <- tribble(
  ~ID, ~y,
   "A", 5,
   "B", 5,
   "C", 8,
   "D", 0,
  "F", 9)
df_secondary <- tribble(
  ~ID, ~z,
   "A", 30,
   "B", 21,
   "C", 22,
   "D", 25,
   "E", 29)
```

a) What would be the output of the following function?

```
full_join(df_primary, df_secondary, by ='ID')
```

Answer:

b) What would be the output of the following function?

```
right_join(df_primary, df_secondary, by ='ID')
```

Answer:

c) What would be the output of the following function?

```
anti_join(df_primary, df_secondary, by ='ID')
```

Answer:

3. Cleaning Messy Data

```
# Create a messy dataset
messy <- data.frame(
  country = c("A", "B", "C"),
  q1_2021 = c(0.03, 0.05, 0.01),
  q2_2021 = c(0.05, 0.07, 0.02),
  q3_2021 = c(0.04, 0.05, 0.01),
  q4_2021 = c(0.03, 0.02, 0.04))
messy
##   country q1_2021 q2_2021 q3_2021 q4_2021
## 1       A    0.03    0.05    0.04    0.03
## 2       B    0.05    0.07    0.05    0.02
## 3       C    0.01    0.02    0.01    0.04
```

Following are the steps in tidying up the above data so that the data follows tidy principles. Briefly state the

steps you would follow in making the above data tidy.

a) Now fill in the blanks to achieve your tidy data goals. What would the dimension of the resulting data tibble be?

```
tidier <- messy %>%
  pivot_longer(cols = #### FILL IN 1 ####
               names_to = "quarter",
               values_to = "growth")
```

b) How would you separate quarter column in tidier tibble to make your data the tidiest? Please write your code below.

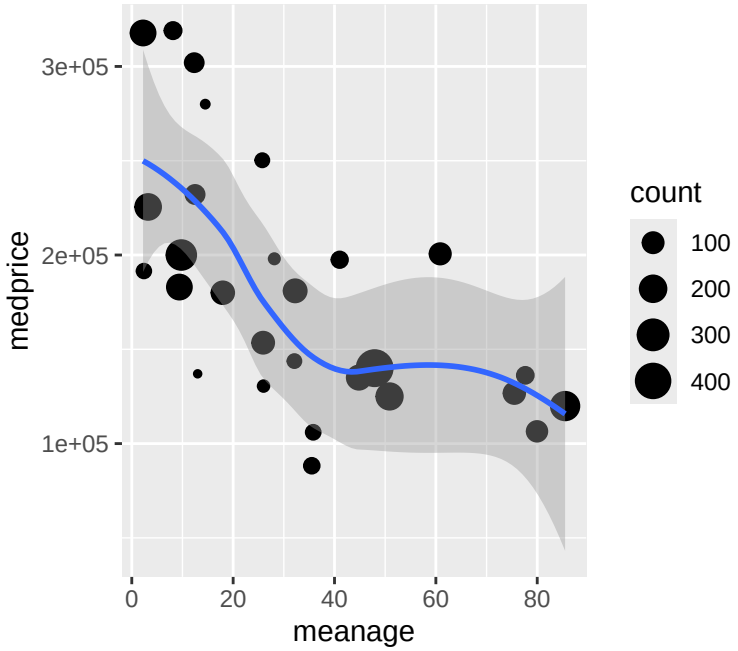
4. Fill in R code.

Provide the necessary lines of R code to produce the given data set or plot, or to complete the described task.

a) Produce the plot below. Note: `age` refers to how old the house when when it was sold.

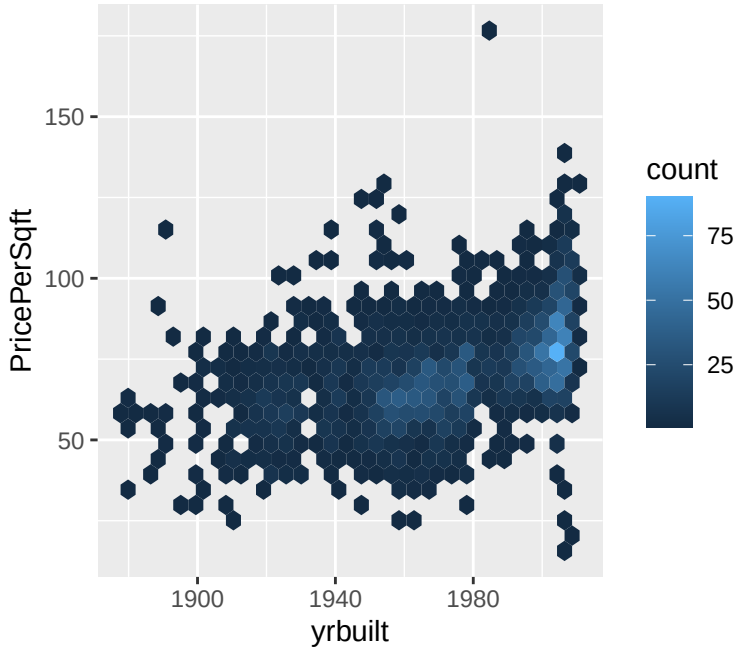
```
by_neighborhood <- ames %>%
  group_by(neighborhood) %>%
  ## LINE 1 ##
  summarise(count = n(),
            medprice = median(price, na.rm = TRUE),
            ## LINE 2 ##
            )

ggplot(data = by_neighborhood, mapping = aes(x = meanage, y = medprice)) +
  ## LINE 3 ##
  geom_smooth()
```

Answer:



b) Rewrite the R code below to be one string of commands connected by pipes.

```
onlyair <- filter(ames, centralair == "Y")
onlyair_small <- select(onlyair, price, sqft, yrbuilt)
onlyair_small <- mutate(onlyair_small, PricePerSqft = price / sqft)
ggplot(data = onlyair_small, mapping = aes(x = yrbuilt, y = PricePerSqft)) +
  geom_hex()
```

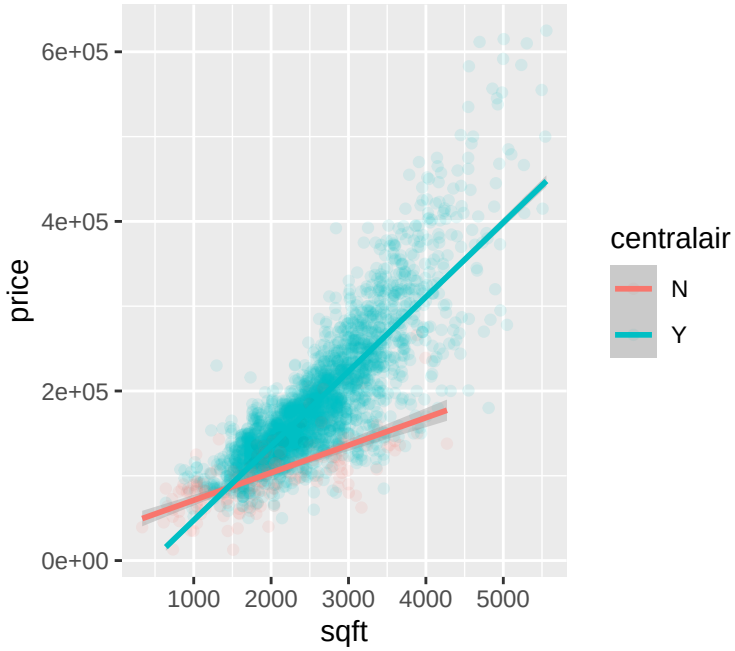


Answer:

5. Explain what would happen (usually 1 sentence will do) if the described changes are made in the R chunks below. Evaluate each change separately.

I)

```
ames %>%
  filter(!is.na(sqft), sqft < 6000) %>%
  ggplot(mapping = aes(x = sqft, y = price, colour = centralair)) +
    geom_point(alpha = 1/10) +
    geom_smooth(method = "lm")
```



a) remove the "!" from `!is.na(sqft)`

Answer:

b) replace the "," after `!is.na(sqft)` with "&"

Answer:

c) replace the third line with `ggplot(mapping = aes(x = sqft, y = price), colour = centralair)`

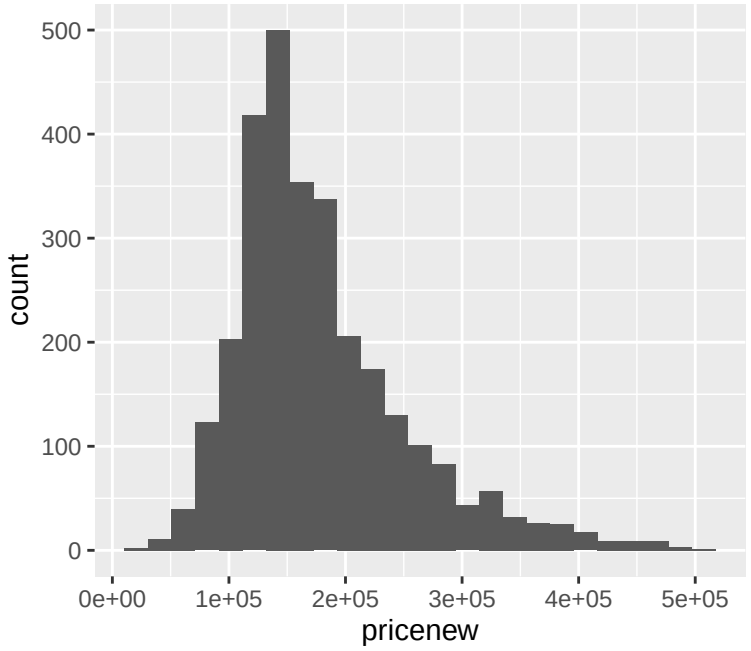
Answer:

d) replace the last line with `geom_smooth()`

Answer:

II.

```
ames %>%
  mutate(pricenew = ifelse(price > 500000, NA, price)) %>%
  ggplot(mapping = aes(x = pricenew)) +
    geom_histogram(bins = 25)
```



a) replace the last line with `geom_freqpoly(bins = 25)`

Answer:

b) replace the last line with `geom_density()`

Answer:

c) replace the `ifelse()` statement with `ifelse(price > 500000, price, NA)`

Answer:

6. Consider the following objects to answer the questions below.

```
x <- -2:3
x
## [1] -2 -1  0  1  2  3
y <- data.frame(y1 = c(25, 16), y2 = c(TRUE, FALSE))
y
##   y1    y2
## 1 25  TRUE
## 2 16 FALSE
z <- list(z1 = x, z2 = y, z3 = c("good","luck!"))
z
## $z1
## [1] -2 -1  0  1  2  3
##
## $z2
##   y1    y2
## 1 25  TRUE
## 2 16 FALSE
##
## $z3
## [1] "good"  "luck!"
```

(a)

Consider the objects x, y and z. Which are atomic vectors and which are lists?

Answer:

(b)

What does the following command evaluate to? Briefly explain your answer.

```
y$y1 + c(-1,1)*y$y2
```

Answer:

(c)

Explain why the first line of code below returns the word "luck!" while the second line of code doesn't return anything (i.e. a null value).

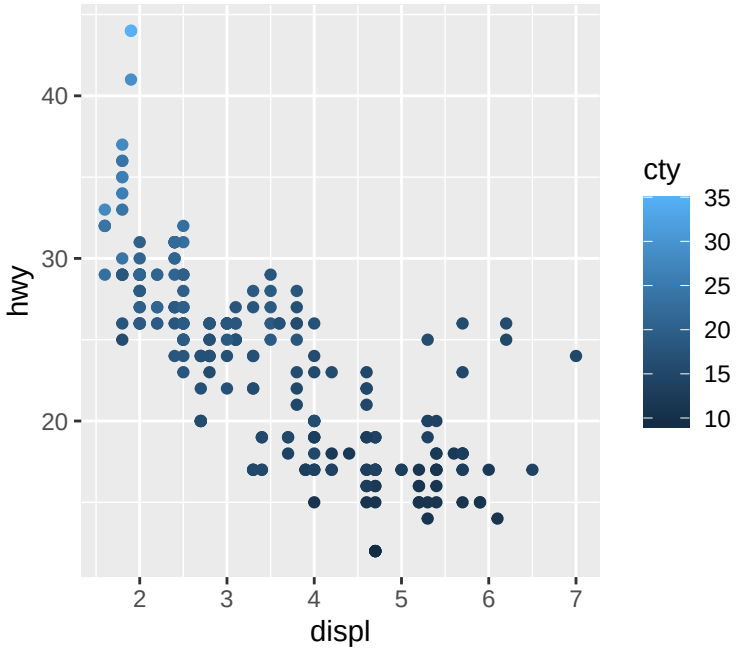
```
z[[3]][2]    # line 1
z[3][2]      # line 2
```

Answer:

7. Miscelleaneus

a) How could you improve upon the following plot? Please provide your code modification and explain your reasoning.

```
ggplot(mpg, aes(x = displ, y = hwy, colour = cty)) +
  geom_point()
```



Answer:

b) What is the dimension of the resulting data tibble from the code chunk below? Explain.

```
tibble(x = sample(1:9),
       group = rep(c("a", "b", "c"), each = 3)) %>%
  mutate(x_mean = mean(x)) %>%
  group_by(group) %>%
  summarize(x_mean_2 = mean(x))
```

Answer:

c) What is the dimension of the resulting data tibble from the code chunk below? Explain.

```
tibble(x = runif(9),
       group = rep(c("a", "b", "c"), each = 3)) %>%
  mutate(x_mean = mean(x)) %>%
  group_by(group) %>%
  arrange(x) %>%
  slice(1)
```

Answer:

d) What is the dimension of the resulting data tibble from the code chunk below? Explain.

```
tibble(group = rep(c("a", "b", "c"), each = 3),
       x = runif(9)) %>%
  group_by(group) %>%
  arrange(x) %>%
  mutate(lag_x = lag(x)) %>%
  tidyr::drop_na(lag_x)
```

Answer:

e) What does the following code chunk do?

```
library(forcats)
ames <- ames %>%
  mutate(neighborhood = fct_reorder(neighborhood, price, median, na.rm = TRUE))
```

Answer:

f) Suppose you have a dataset with a factor variable `f` having levels "low", "medium", "high". Write a code snippet to reverse the order of levels in `f`.

Answer:

```
f <- factor(c("low", "medium", "high"), levels = c("low", "medium", "high"))
f <-
```

g) Assuming you have a data frame project_data that contains the columns project_start and project_end (both in "YYYY-MM-DD" format), write a snippet to add a new column project_length to this data frame. project_length should contain the duration of each project in terms of the number of whole weeks.

```
library(lubridate)
project_data <- data.frame(
  project_id = 1:3,
  project_start = ymd(c("2020-01-01", "2020-02-15", "2020-03-20")),
  project_end = ymd(c("2020-01-22", "2020-03-10", "2020-04-25"))
)
```

Answer:

7 String manipulations

a. Given a vector of words, write a function named `find_vowel_5_letter_words` that identifies all five-letter words starting and ending with a vowel. The function should return the indices of these words within the original vector. For this problem, assume vowels are "a", "e", "i", "o", and "u", and consider case insensitivity.

b. Write a command to remove all instances of # from the `comments` vector.

```
comments <- c("Loving the new #season!", "Can't wait for #Friday!")
```

c. Write a command to insert dashes (-) into the dates vector to achieve the desired format.

```
dates <- c("01012023", "15032024")
```

d. Use the `stringr` functions to separate the names and email addresses into two vectors, `names` and `emails`.

```
info <- "John Doe:john.doe@example.com, Jane Smith:jane.smith@example.com"
```