Problem Set 3

Brynn Woolley

Problem 1

a. Download the file AUX_I from this location, and determine how to read it into R. Then download the file DEMO_I from this location. Note that each page contains a link to a documentation file for that data set. Merge the two files to create a single `data.frame`. Keep only records which matched. Print out the dimensions of the merged `data.frame`.

```
# helper function
read_xpt_url <- function(url) {
  f <- tempfile(fileext = ".xpt")
  h <- new_handle()
  handle_setheaders(h, "User-Agent" = "Mozilla/5.0")
  curl_download(url, f, mode = "wb", handle = h)
  # quick sanity check
  stopifnot(startsWith(readChar(f, 60, useBytes = TRUE), "HEADER RECORD*******LIBRARY"))
  read_xpt(f)
}

urls <- c(
  AUX  = "https://wwwn.cdc.gov/Nchs/Data/Nhanes/Public/2015/DataFiles/AUX_I.xpt",
  DEMO = "https://wwwn.cdc.gov/Nchs/Data/Nhanes/Public/2015/DataFiles/DEMO_I.xpt"
)


# read
dat <- lapply(urls, read_xpt_url)

# merge
merged <- inner_join(as.data.frame(dat$AUX), as.data.frame(dat$DEMO), by = "SEQN")
print(dim(merged))
```

```
[1] 4582  119
```

b. We'll be using the following variables. Clean up each - ensure all missing values are actually `NA` (rather than `999` or something), and if it's categorical, convert it to `factor` with informative `levels`.

- Gender
- Citizenship status
- Number of children 5 years or younger in the household
- Annual household income - There's also an issue with the ordering of the categories here; take a look, identify the issue, and implement a solution.

```
cleaned <- merged %>%
  mutate(
# gender
gender = factor(RIAGENDR,
  levels = c(1, 2),
  labels = c("Male", "Female")
),

# citizenship status
citizenship = factor(DMDCITZN,
  levels = c(1, 2),
  labels = c("Citizen by birth/naturalization", "Not a U.S. citizen")
),

# # children age 5 or younger in household
# Codebook: 0,1,2,3 (where 3 = "3 or more")
kids_le5 = case_when(
  DMDHHSZA %in% 0:3 ~ DMDHHSZA,
  TRUE ~ NA_real_
),
kids_le5 = factor(
  kids_le5,
  levels = c(0, 1, 2, 3),
  labels = c("0", "1", "2", "3+")
),

# annual household income: convert 77/99 to NA, then ordered factor
indhhin2_code = na_if(INDHHIN2, 77),      # Refused -> NA
indhhin2_code = na_if(indhhin2_code, 99), # Don't know -> NA
```

```
income = factor(
  indhhin2_code,
  levels = 1:12,
  labels = c(
    "< $5,000",
    "$5,000-$9,999",
    "$10,000-$14,999",
    "$15,000-$19,999",
    "$20,000-$24,999",
    "$25,000-$34,999",
    "$35,000-$44,999",
    "$45,000-$54,999",
    "$55,000-$64,999",
    "$65,000-$74,999",
    "$75,000-$99,999",
    "$100,000 and over"
  ),
  ordered = TRUE
)
  ) %>%
  select(SEQN, gender, citizenship, kids_le5, income)
```

c. The Tympanometric width measure is looks approximately like a Poisson distribution. Fit four Poisson regression models predicting a respondent's Tympanometric width in each ear. Each model is defined below, for a specific ear and a specific set of covariates.

- 1R - Right ear: gender
- 2R - Right ear: gender, citizenship status (as categorical), number of children (as continuous), annual household income (as continuous)
- 1L - Left ear: gender
- 2L - Left ear: gender, citizenship status (as categorical), number of children (as continuous), annual household income (as continuous)

Produce a table presenting the estimated incidence risk ratios for the coefficients in each model, along with the sample size for the model, the pseudo-(R^2), and AIC values. (This can be a single table, or one table for coefficients and a separate table for model statistics).

The table(s) should be "nice" - use a function such as `kable` from knitr, or the stargazer package (or find another approach) to generate HTML/LaTeX tables for inclusion. The results should be clearly labeled, rounded appropriately, and easily readable.

```r
# locate outcome variables by label
var_labels <- sapply(merged, function(x) attr(x, "label"))
right_var  <- names(merged)[grep("tympanometric\\s*width.*right", var_labels, ignore.case = T
left_var   <- names(merged)[grep("tympanometric\\s*width.*left",  var_labels, ignore.case = T

# analysis frame
dat <- merged %>%
  select(SEQN,
         y_right = !!sym(right_var),
         y_left  = !!sym(left_var)) %>%
  inner_join(cleaned, by = "SEQN") %>%
  mutate(
    kids_le5_num = as.numeric(as.character(kids_le5)), # 0,1,2,3 (3=3+)
    income_num   = as.numeric(income)                  # 1..12 (ordered)
  )

# model specs
specs <- list(
  `1R` = list(y = "y_right", x = c("gender")),
  `2R` = list(y = "y_right", x = c("gender","citizenship","kids_le5_num","income_num")),
  `1L` = list(y = "y_left",  x = c("gender")),
  `2L` = list(y = "y_left",  x = c("gender","citizenship","kids_le5_num","income_num"))
)

# single fitter that returns both IRRs and model stats
fit_one <- function(tag, y, x, data) {
  f <- reformulate(x, response = y)
  cc <- model.frame(f, data = data, na.action = na.omit)
  m  <- glm(f, family = poisson(link = "log"), data = cc)

  # IRRs (exclude intercept)
  irr <- tidy(m, conf.int = TRUE, exponentiate = TRUE) |>
    filter(term != "(Intercept)") |>
    transmute(
      Model = tag,
      Term  = case_when(
        term == "kids_le5_num" ~ "Children  5 (per +1)",
        term == "income_num"   ~ "Income code (1-12, per +1)",
        str_starts(term, "gender") ~ str_replace(term, "gender", "Gender: "),
        str_starts(term, "citizenship") ~ str_replace(term, "citizenship", "Citizenship: "),
        TRUE ~ term
      ),
```

```
      IRR   = round(estimate, 2),
      `95% CI` = sprintf("[%.2f, %.2f]", conf.low, conf.high),
      p     = ifelse(p.value < 0.001, "<0.001", sprintf("%.3f", p.value))
    )

  # model stats
  ll_full <- as.numeric(logLik(m))
  ll_null <- as.numeric(logLik(update(m, . ~ 1)))
  stats <- tibble(
    Model = tag,
    N = nobs(m),
    `Pseudo-R^2 (McFadden)` = round(1 - ll_full/ll_null, 3),
    AIC = round(AIC(m), 1)
  )

  list(irr = irr, stats = stats)
}

# run all models and bind results
res <- imap(specs, ~ fit_one(.y, .x$y, .x$x, dat))
irr_tbl    <- bind_rows(map(res, "irr"))
stats_tbl  <- bind_rows(map(res, "stats"))

# pretty tables (IRR)
kable(
  irr_tbl,
  align   = c("l","l","r","l","r"),
  caption = "Poisson regression: Incidence Rate Ratios (IRR) with 95\\% CI",
  escape  = TRUE,
  booktabs = TRUE
) |>
  kable_styling(full_width = FALSE, latex_options = c("hold_position"))
```

Table 1: Poisson regression: Incidence Rate Ratios (IRR) with 95% CI

Model	Term	IRR	95% CI	p
1R	Gender: Female	1.01	[1.00, 1.02]	0.006
2R	Gender: Female	1.00	[0.99, 1.01]	0.689
2R	Citizenship: Not a U.S. citizen	1.09	[1.08, 1.10]	<0.001
2R	Children 5 (per +1)	0.99	[0.98, 0.99]	<0.001
2R	Income code (1–12, per +1)	1.00	[1.00, 1.00]	<0.001

1L	Gender: Female	1.01	[1.01, 1.02]	<0.001
2L	Gender: Female	1.00	[0.99, 1.01]	0.555
2L	Citizenship: Not a U.S. citizen	1.06	[1.04, 1.07]	<0.001
2L	Children 5 (per +1)	0.98	[0.97, 0.98]	<0.001
2L	Income code (1–12, per +1)	1.00	[1.00, 1.00]	0.020

```
# model fit table
kable(
  stats_tbl,
  align   = c("l","r","r","r"),
  caption = "Model fit statistics",
  escape  = TRUE,
  booktabs = TRUE
) |>
  kable_styling(full_width = FALSE, latex_options = c("hold_position"))
```

Table 2: Model fit statistics

Model	N	Pseudo-R^2 (McFadden)	AIC
1R	4149	0.000	96618.5
2R	2635	0.004	65010.2
1L	4103	0.000	98685.1
2L	2606	0.002	64976.8

d. From model 2L, provide evidence whether there is a difference between males and females in terms of their incidence risk ratio. Test whether the predicted value of Tympanometric width measure of the left ear differs between men and women. Include the results of the each test and their interpretation.

```
# refit Model 2L only (same as previous formula)
mod_2L <- glm(
  y_left ~ gender + citizenship + kids_le5_num + income_num,
  family = poisson(link = "log"),
  data = dat
)

# extract coefficient for gender
coef_gender <- broom::tidy(mod_2L, exponentiate = TRUE, conf.int = TRUE) %>%
  filter(str_detect(term, "gender"))

kable(
```

```
  coef_gender %>%
    transmute(
      Term = term,
      IRR = round(estimate, 2),
      `95% CI` = sprintf("[%.2f, %.2f]", conf.low, conf.high),
      p = ifelse(p.value < 0.001, "<0.001", sprintf("%.3f", p.value))
    ),
  caption = "Model 2L - Gender Effect (IRR and 95\\% CI)",
  escape = TRUE
) |>
  kable_styling(full_width = FALSE)
```

Table 3: Model 2L – Gender Effect (IRR and 95% CI)

Term	IRR	95% CI	p
genderFemale	1	[0.99, 1.01]	0.555

```
# test predicted mean outcome
emm_gender <- emmeans(mod_2L, ~ gender, type = "response")  # type='response' gives mean in o
contrast_gender <- contrast(emm_gender, "pairwise")

# display estimated means and difference
kable(summary(emm_gender), caption = "Predicted Tympanometric Width Means (by Gender)") |>
  kable_styling(full_width = FALSE)
```

Table 4: Predicted Tympanometric Width Means (by Gender)

gender	rate	SE	df	asymp.LCL	asymp.UCL
Male	88.02603	0.2950555	Inf	87.44963	88.60623
Female	87.80629	0.2841778	Inf	87.25107	88.36503

```
kable(summary(contrast_gender), caption = "Pairwise Comparison (Male vs Female)") |>
  kable_styling(full_width = FALSE)
```

Table 5: Pairwise Comparison (Male vs Female)

contrast	ratio	SE	df	null	z.ratio	p.value

Male / Female	1.002503	0.0042456	Inf	1	0.5901977	0.5550581

Problem 2 - Sakila

```
# Ensure the database file exists
db_path <- "sakila_master.db"
if (!file.exists(db_path)) {
  curl::curl_download(
    "https://raw.githubusercontent.com/bradleygrant/sakila-sqlite3/main/sakila_master.db",
    db_path, mode = "wb"
  )
}

# Open the connection for all Problem 2 chunks
con <- DBI::dbConnect(RSQLite::SQLite(), db_path)
stopifnot(DBI::dbIsValid(con))
stopifnot("customer" %in% DBI::dbListTables(con))
```

Use the "sakila" database discussed in class. It can be downloaded from https://github.com/bradleygrant/sakila-sqlite3.

For these problems, do not use any of the tables whose names end in `_list`.

For each of the following questions, solve them in two ways: First, use SQL query or queries to extract the appropriate table(s), then use regular R operations on those `data.frames` to answer the question. Second, use a single SQL query to answer the question. Compare each approach using microbenchmark.

a. For each store, how many customers does that store have, and what percentage of customers of that store are active in the system?

```
# single SQL query: customers per store + % active
res <- dbGetQuery(con, "
  SELECT
      c.store_id,
      COUNT(*)            AS n_customers,
      ROUND(AVG(c.active) * 100.0, 1) AS pct_active
  FROM customer AS c
  GROUP BY c.store_id
  ORDER BY c.store_id;
")
```

```r
# pretty table
kable(res, digits = c(0, 0, 1),
            caption = "Store customers \\& \\% active") |>
  kable_styling(full_width = FALSE, latex_options = "hold_position")
```

Table 6: Store customers & % active

store_id	n_customers	pct_active
1	326	97.5
2	273	97.4

b. Generate a table identifying the names and country of each staff member.

```r
# staff name + country via staff -> address -> city -> country
res_staff <- dbGetQuery(con, "
  SELECT
      s.staff_id,
      (s.first_name || ' ' || s.last_name) AS staff_name,
      co.country AS country
  FROM staff AS s
  JOIN address AS a   ON s.address_id = a.address_id
  JOIN city    AS ci  ON a.city_id    = ci.city_id
  JOIN country AS co  ON ci.country_id = co.country_id
  ORDER BY s.staff_id;
")

kable(res_staff,
      caption = "Staff names and country") |>
  kable_styling(full_width = FALSE, latex_options = 'hold_position')
```

Table 7: Staff names and country

staff_id	staff_name	country
1	Mike Hillyer	Canada
2	Jon Stephens	Australia

c. Identify the name(s) of the film(s) which was/were rented for the highest dollar value. (Assume all costs are in USD regardless of country.) (Hint: You can merge a table more than once.)

```
# total revenue per film = sum of payments joined via rental -> inventory -> film
best_films <- dbGetQuery(con, "
  WITH film_rev AS (
    SELECT
      f.film_id,
      f.title,
      ROUND(SUM(p.amount), 2) AS revenue
    FROM film AS f
    JOIN inventory AS i ON i.film_id = f.film_id
    JOIN rental   AS r ON r.inventory_id = i.inventory_id
    JOIN payment  AS p ON p.rental_id   = r.rental_id
    GROUP BY f.film_id, f.title
  )
  SELECT film_id, title, revenue
  FROM film_rev
  WHERE revenue = (SELECT MAX(revenue) FROM film_rev)
  ORDER BY title;
")

kable(best_films,
      caption = "Film(s) with the highest total rental revenue (USD)") |>
  kable_styling(full_width = FALSE, latex_options = "hold_position")
```

Table 8: Film(s) with the highest total rental revenue (USD)

film_id	title	revenue
879	TELEGRAPH VOYAGE	231.73

```
if (exists("con") && DBI::dbIsValid(con)) DBI::dbDisconnect(con)
```

Problem 3 - Australian Records

Download the "Australia - 500 Records" data from https://www.briandunning.com/sample-data/ and import it into R. This is entirely fake data; you can read the website for details. Use it to answer the following questions.

```
# download + read the AU-500 ZIP
zip_url <- "https://www.briandunning.com/sample-data/au-500.zip"
zip_path <- tempfile(fileext = ".zip")
curl::curl_download(zip_url, zip_path, mode = "wb")
```

```r
# inspect zip; pick the first CSV inside
zf <- unzip(zip_path, list = TRUE)
stopifnot(any(grepl("\\.csv$", zf$Name, ignore.case = TRUE)))
csv_name <- zf$Name[grep("\\.csv$", zf$Name, ignore.case = TRUE)][1]

# extract and read
csv_path <- unzip(zip_path, files = csv_name, exdir = tempdir(), overwrite = TRUE)
au <- read.csv(csv_path, stringsAsFactors = FALSE)
```

a. What percentage of the websites are .com's (as opposed to .net, .com.au, etc)?

```r
# find the website column (case-insensitive match on 'web')
web_col <- grep("^web|website", names(au), ignore.case = TRUE, value = TRUE)
stopifnot(length(web_col) >= 1)
web_col <- web_col[1]

# normalize and extract host
web <- tolower(trimws(au[[web_col]]))
web <- sub("^https?://", "", web)
web <- sub("^www\\.", "", web)
host <- sub("/.*$", "", web)          # drop path/query
host <- sub("\\.$", "", host)         # drop trailing dot

# keep non-missing, non-empty hosts
keep <- nzchar(host) & !is.na(host)
host_kept <- host[keep]

# plain .com (NOT .com.au, etc.): host ends with ".com"
is_com <- grepl("\\.com$", host_kept)

pct_com <- mean(is_com) * 100
out <- data.frame(
  n_total_with_website = length(host_kept),
  n_com = sum(is_com),
  pct_com = round(pct_com, 1)
)

kable(out, caption = "Share of websites that are plain .com (not .com.au, etc.)") |>
  kable_styling(full_width = FALSE, latex_options = "hold_position")
```

Table 9: Share of websites that are plain .com (not .com.au, etc.)

n_total_with_website	n_com	pct_com
500	0	0

b. What is the most common domain name amongst the email addresses? (In the email "statistics@umich.edu", "umich.edu" is the domain name.)

```
# locate the email column
email_col <- grep("email", names(au), ignore.case = TRUE, value = TRUE)[1]
stopifnot(length(email_col) == 1)

# extract domains (robust to multiple emails per cell separated by , or ;)
emails_raw <- au[[email_col]]

# split cells with multiple emails, flatten
emails_vec <- unlist(strsplit(emails_raw, "[,;]", fixed = FALSE), use.names = FALSE)

# clean + keep valid-looking emails
emails_vec <- tolower(trimws(emails_vec))
emails_vec <- emails_vec[nzchar(emails_vec) & grepl(".+@.+\\..+", emails_vec)]

# domain = text after '@', strip trailing punctuation/spaces
domains <- sub(".*@", "", emails_vec)
domains <- sub("[\\s\\p{P}]+$", "", domains, perl = TRUE)

# count
tab <- sort(table(domains), decreasing = TRUE)
top_domains <- head(tab, 10)

# most common domain(s) (handles ties)
max_n <- as.integer(top_domains[1])
most_common <- names(tab)[tab == max_n]

# output
kable(
  data.frame(domain = names(top_domains), n = as.integer(top_domains)),
  caption = "Top email domains"
) |>
  kable_styling(full_width = FALSE, latex_options = "hold_position")
```

Table 10: Top email domains

domain	n
hotmail.com	114
gmail.com	102
yahoo.com	84
agar.net.au	1
agney.net.au	1
ahlborn.com.au	1
albrough.com.au	1
alerte.com.au	1
amedro.net.au	1
andrion.com.au	1

```
most_common
```

```
[1] "hotmail.com"
```

c. What proportion of company names contain a non-alphabetic character, excluding commas and whitespace. (E.g. "Jane Doe, LLC" would not contain an eligible non-alphabetic character; "Plumber 247" would.) What about if you also exclude ampersands ("&")?

```
# find the company-name column
co_col <- grep("company|business|firm|employer", names(au), ignore.case = TRUE, value = TRUE)
stopifnot(length(co_col) == 1)

co <- au[[co_col]]
co <- trimws(co)
keep <- nzchar(co) & !is.na(co)              # use only non-missing company names
co_kept <- co[keep]

# 1) Non-alphabetic, excluding commas and whitespace
#    → flag any character NOT A-Z/a-z, comma, or whitespace
has_nonalpha_excl_comma_space <- grepl("[^A-Za-z,\\s]", co_kept)

# 2) Same, but also exclude ampersand (&)
has_nonalpha_excl_comma_space_amp <- grepl("[^A-Za-z,\\s&]", co_kept)

n <- length(co_kept)
out <- data.frame(
```

```
  metric = c(
    "Contains non-alphabetic (excluding commas & whitespace)",
    "Contains non-alphabetic (excluding commas, whitespace, and &)"
  ),
  count = c(sum(has_nonalpha_excl_comma_space),
            sum(has_nonalpha_excl_comma_space_amp)),
  proportion = c(mean(has_nonalpha_excl_comma_space),
                 mean(has_nonalpha_excl_comma_space_amp))
)

# pretty table
knitr::kable(transform(out, proportion = sprintf("%.1f%%", 100*proportion)),
             caption = "Proportion of company names with non-alphabetic characters") |>
  kableExtra::kable_styling(full_width = FALSE, latex_options = "hold_position")
```

Table 11: Proportion of company names with non-alphabetic characters

metric	count	proportion
Contains non-alphabetic (excluding commas & whitespace)	494	98.8%
Contains non-alphabetic (excluding commas, whitespace, and &)	494	98.8%

d. In Australia, phone have 10 digits - but unlike in the US where we write all numbers as "123-456-7890", they write land lines and cell phones differently[1]:

- Landlines: 12-3456-7890
- Cell phones: 1234-567-890

There are two different phones listed for each record. Make all phone numbers written like cell phones. Show it works by printing the first 10 phone numbers of each column.

```
# helper
fmt_cell <- function(x) {
  d <- gsub("\\D", "", x)                  # keep digits only

  d <- ifelse(grepl("^61\\d{9}$", d), paste0("0", substr(d, 3, 11)), d)

  # only format if we have exactly 10 digits
  ok <- nchar(d) == 10
  out <- rep(NA_character_, length(d))
  out[ok] <- paste0(substr(d[ok], 1, 4), "-", substr(d[ok], 5, 7), "-", substr(d[ok], 8, 10))
  out
```

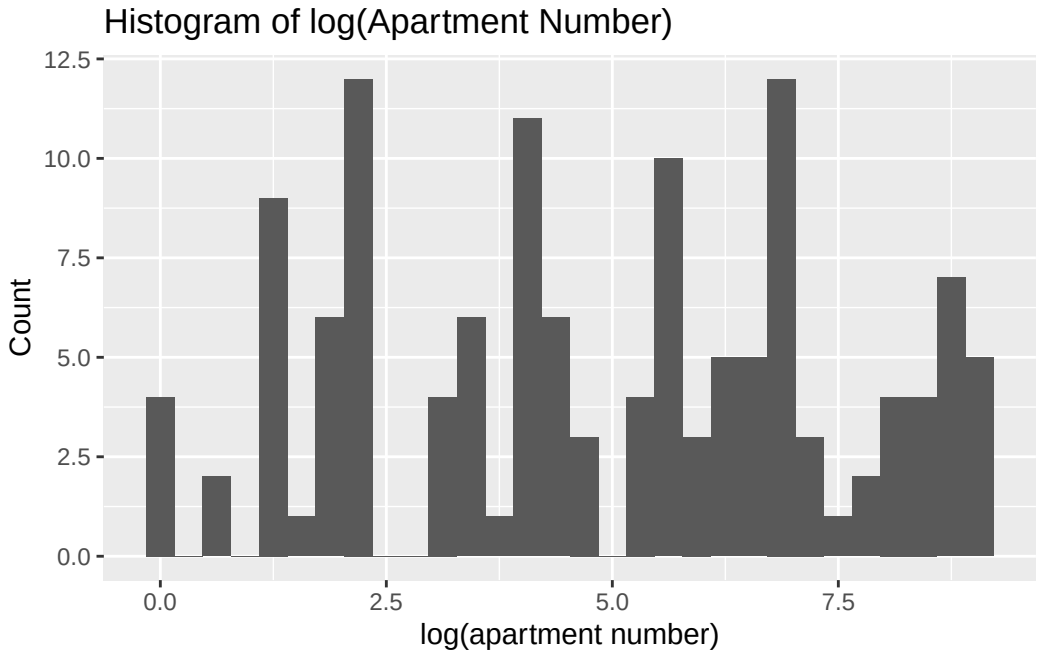
```
}

# identify phone columns
phone_cols <- grep("phone|mobile|tel", names(au), ignore.case = TRUE, value = TRUE)

stopifnot(length(phone_cols) >= 2)  # require at least two phone columns

# reformat in place to cell style
au[phone_cols] <- lapply(au[phone_cols], fmt_cell)

# show first 10 numbers of each phone column to verify
verif <- lapply(au[phone_cols], function(v) head(v, 10))
names(verif) <- phone_cols

# print a compact table
print(do.call(cbind, verif), quote = FALSE)
```

```
      phone1       phone2
 [1,] 0381-749-123 0458-665-290
 [2,] 0799-973-366 0497-622-620
 [3,] 0855-589-019 0427-885-282
 [4,] 0260-444-682 0443-795-912
 [5,] 0214-556-085 0453-666-885
 [6,] 0878-681-355 0451-966-921
 [7,] 0865-228-931 0427-991-688
 [8,] 0252-269-402 0415-961-606
 [9,] 0731-849-989 0411-732-965
[10,] 0868-904-661 0461-862-457
```

e. Produce a histogram of the log of the apartment numbers for all addresses. (You may assume any number at the end of the an address is an apartment number.)

```
# locate an address column
addr_col <- grep("address|street", names(au), ignore.case = TRUE, value = TRUE)[1]
stopifnot(length(addr_col) == 1)

addr <- au[[addr_col]]

# extract trailing number
apt_str <- stringr::str_extract(addr, "\\d+\\s*$")
apt_num <- suppressWarnings(as.numeric(trimws(apt_str)))
```

```r
apt_num <- apt_num[!is.na(apt_num) & apt_num > 0]

apt_df <- data.frame(log_apt = log(apt_num))

# plot
if (!"ggplot2" %in% loadedNamespaces()) library(ggplot2)

ggplot(apt_df, aes(x = log_apt)) +
  geom_histogram(bins = 30) +
  labs(
    title = "Histogram of log(Apartment Number)",
    x = "log(apartment number)",
    y = "Count"
  )
```



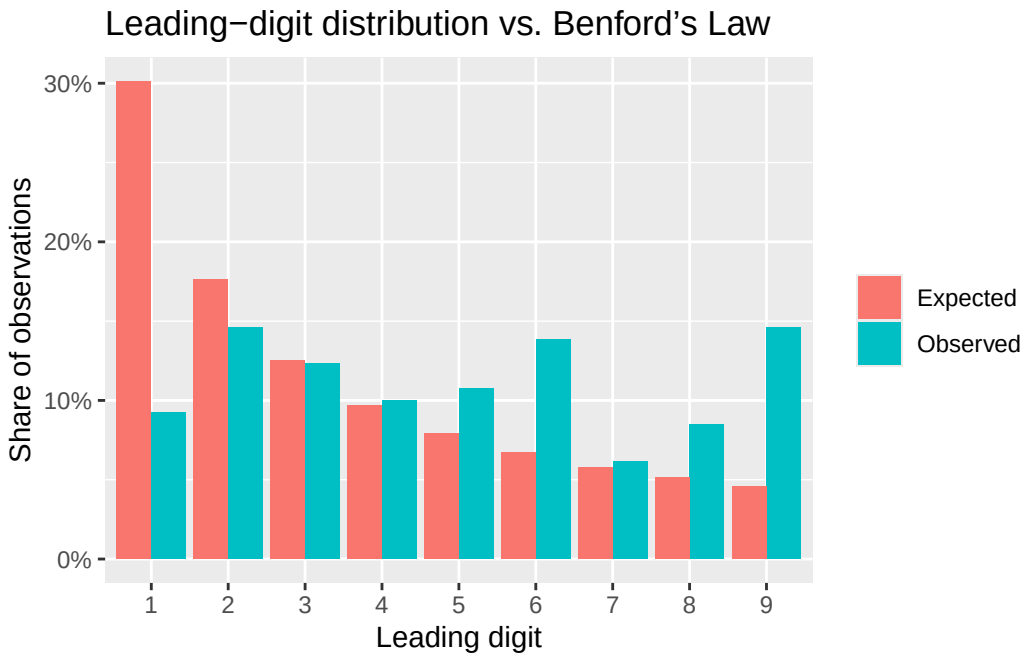
f. Benford's law is an observation about the distribution of the leading digit of real numerical data. Examine whether the apartment numbers appear to follow Benford's law. Do you think the apartment numbers would pass as real data?

```r
# reuse the same extraction
addr_col <- grep("address|street", names(au), ignore.case = TRUE, value = TRUE)[1]
stopifnot(length(addr_col) == 1)

addr <- au[[addr_col]]
apt_str <- stringr::str_extract(addr, "\\d+\\s*$")
apt_num <- suppressWarnings(as.numeric(trimws(apt_str)))
apt_num <- apt_num[!is.na(apt_num) & apt_num > 0]

# leading digit 1-9
leading_digit <- as.integer(substr(as.character(apt_num), 1, 1))
leading_digit <- leading_digit[leading_digit %in% 1:9]

# observed counts & proportions
obs_counts <- tabulate(leading_digit, nbins = 9)
names(obs_counts) <- as.character(1:9)
n <- sum(obs_counts)
obs_prop <- obs_counts / n

# benford expected proportions
benford_p <- log10(1 + 1 / (1:9))

# chi-square
chisq <- chisq.test(x = obs_counts, p = benford_p, rescale.p = TRUE, simulate.p.value = FALSE

# observed vs expected table
benford_tbl <- data.frame(
  digit = 1:9,
  observed_n = as.integer(obs_counts),
  observed_pct = round(100 * obs_prop, 1),
  expected_pct = round(100 * benford_p, 1),
  diff_pct_pt = round(100 * (obs_prop - benford_p), 1)
)

knitr::kable(
  benford_tbl,
  caption = "Leading-digit distribution of apartment numbers vs. Benford expectation (\\%)"
) |>
  kableExtra::kable_styling(full_width = FALSE, latex_options = "hold_position")
```

Table 12: Leading-digit distribution of apartment numbers vs. Benford expectation (%)

digit	observed_n	observed_pct	expected_pct	diff_pct_pt
1	12	9.2	30.1	-20.9
2	19	14.6	17.6	-3.0
3	16	12.3	12.5	-0.2
4	13	10.0	9.7	0.3
5	14	10.8	7.9	2.9
6	18	13.8	6.7	7.2
7	8	6.2	5.8	0.4
8	11	8.5	5.1	3.3
9	19	14.6	4.6	10.0

```r
# plot observed vs expected
if (!"ggplot2" %in% loadedNamespaces()) library(ggplot2)

plot_df <- tidyr::pivot_longer(
  data.frame(
    digit = factor(1:9),
    Observed = obs_prop,
    Expected = benford_p
  ),
  cols = c("Observed", "Expected"),
  names_to = "series",
  values_to = "prop"
)

ggplot(plot_df, aes(x = digit, y = prop, fill = series)) +
  geom_col(position = "dodge") +
  scale_y_continuous(labels = scales::percent_format(accuracy = 1)) +
  labs(
    title = "Leading-digit distribution vs. Benford's Law",
    x = "Leading digit",
    y = "Share of observations",
    fill = NULL
  )
```




```
# report test result and a simple interpretation
decision <- if (chisq$p.value < 0.05) "reject" else "do not reject"
cat(sprintf(
  "Chi-square GOF: X^2 = %.2f (df = %d), p = %.4f → We %s Benford's Law at   = 0.05.\n",
  unname(chisq$statistic), unname(chisq$parameter), chisq$p.value, decision
))
```

```
Chi-square GOF: X^2 = 62.27 (df = 8), p = 0.0000 → We reject Benford's Law at   = 0.05.
```

Footnotes

[1] Technically phone numbers starting in 04 are cell phones and all others are landlines, but we'll ignore that detail for this problem. "'

GitHub Link

- Repo: https://github.com/brynnwoolley/STATS-506#
