Homework 10

2025-10-30

Boilerplate

Import packages

```r
library(printr)      # pretty print for Rmd
library(lubridate)   # dates
library(ggplot2)     # plots
library(dplyr)       # dataframes
library(tidyr)
library(tidyverse)
```

Question 14.1

> The breast cancer data set breast-cancer-wisconsin.data.txt from https://archive.ics.uci.edu/dataset/15/breast+cancer+wisconsin+original (description available at the same URL) has missing values.
>
> 1. Use the mean/mode imputation method to impute values for the missing data.
> 2. Use regression to impute values for the missing data.
> 3. Use regression with perturbation to impute values for the missing data.
> 4. (Optional) Compare the results and quality of classification models (e.g., SVM, KNN) build using
>   1. the data sets from questions 1,2,3;
>   2. the data that remains after data points with missing values are removed; and
>   3. the data set when a binary variable is introduced to indicate missing values.

Let's get the data, and add the column names in. In the description, it was stated that "?" is the unknown.

```r
raw <- read.csv(
  "data-14.1/breast-cancer-wisconsin.data.csv",
  col.names=c(
    "id", "thickness",
    "size_uniformity", "shape_uniformity",
    "adhesion", "epithelial_size",
    "bare_nuclei", "bland_chromatin",
    "normal_nucleoli", "mitoses",
    "class"
  ),
  na.strings="?"
)
```

Let's identify the rows that are NA because those are the ones we want to see. Note: all the NA are `bare_nuclei`, which makes this easier to track.

```
nan_rows = apply(is.na(raw), MARGIN = 1, any)
n_nan_rows <- nrow(raw[nan_rows,])
raw[nan_rows,c("id","bare_nuclei")]
```

	id	bare_nuclei
23	1057013	NA
40	1096800	NA
139	1183246	NA
145	1184840	NA
158	1193683	NA
164	1197510	NA
235	1241232	NA
249	169356	NA
275	432809	NA
292	563649	NA
294	606140	NA
297	61634	NA
315	704168	NA
321	733639	NA
411	1238464	NA
617	1057067	NA

14.1.A - Mean imputation

Since we know where our missing data is, doing a basic mean imputation is very easy.

```
mean_bare_nuclei <- raw %>%
  mutate(across(-id, as.double)) %>%
  replace_na(as.list(sapply(raw, mean, na.rm=TRUE)))

mean_bare_nuclei[nan_rows,c("id","bare_nuclei")]
```

	id	bare_nuclei
23	1057013	3.548387
40	1096800	3.548387
139	1183246	3.548387
145	1184840	3.548387
158	1193683	3.548387
164	1197510	3.548387
235	1241232	3.548387
249	169356	3.548387
275	432809	3.548387
292	563649	3.548387
294	606140	3.548387
297	61634	3.548387
315	704168	3.548387
321	733639	3.548387
411	1238464	3.548387
617	1057067	3.548387

14.1.B - Regression Imputation

In order to perform regression imputation we fit a model on non-nan rows excluding the identification column because it is arbitrary.

```
regression_imputation <- raw
regression_model <- lm(bare_nuclei ~ . - id, raw[-nan_rows,])
regression_imputation[nan_rows,"bare_nuclei"] <- predict(regression_model, raw[nan_rows,])

regression_imputation[nan_rows,c("id","bare_nuclei")]
```

	id	bare_nuclei
23	1057013	7.294519
40	1096800	3.250415
139	1183246	1.221394
145	1184840	1.580402
158	1193683	1.241581
164	1197510	1.425161
235	1241232	1.942271
249	169356	1.398817
275	432809	1.606746
292	563649	6.450250
294	606140	1.218099
297	61634	0.982795
315	704168	1.856060
321	733639	1.398817
411	1238464	1.221394
617	1057067	1.067020

14.1.C - Regression Imputation with perturbation

We will use the same regression model as the basis for the imputation, with the addition of a perturbation of the solution from a normal distribution using the standard error as the standard deviation.

```
regression_stats <- summary(regression_model)

perturbation_imputation <- raw
perturbation_imputation[
  nan_rows,"bare_nuclei"
] <- predict(regression_model, raw[nan_rows,]) + rnorm(
  n_nan_rows,
  0,
  sd = regression_stats$sigma
)

perturbation_imputation[nan_rows,c("id","bare_nuclei")]
```

	id	bare_nuclei
23	1057013	9.4162660
40	1096800	1.5930432
139	1183246	3.9198758
145	1184840	1.2005178
158	1193683	0.9096141
164	1197510	0.4095271
235	1241232	6.0398723
249	169356	-1.2719599
275	432809	1.3121804
292	563649	12.0279517
294	606140	0.4442407
297	61634	6.5890501
315	704168	2.4315598
321	733639	1.8494597
411	1238464	3.2709152
617	1057067	0.7494101

Question 15.1

> Describe a situation or problem from your job, everyday life, current events, etc., for which optimization would be appropriate. What data would you need?

A good example of optimization in everyday life is running errands. This is a generalization of the Traveling Salesman problem[1] although obviously everyday people use heuristics for their solution rather than using a formal solution. The problem stated formally is: what route minimizes total time or distance, subject to real-world constraints such as avoiding left turns, ordering stops, or respecting business hours. Data that would need to be collected are:

- The distance or time required to travel a road section at a specific time
- A preference against left hand turns[2]
- Preference constraints on road sections (for example there is a stretch of dirt road near me that I do not like to travel on)
- Any order constraints (e.g. opening/closing times, the stop to pick up a gift needs to occur before the stop to mail it, . . .)

[1] https://en.wikipedia.org/wiki/Travelling_salesman_problem
[2] https://www.cnn.com/2017/02/16/world/ups-trucks-no-left-turns