ETC3250/5250 Tutorial 4 Instructions

Categorical response, and dimension reduction

prepared by Professor Di Cook

Week 4

🎯 Objective

The objectives for this week are to

- examine the mathematics involved in re-writing the logistic regression model and linear discriminant analysis rules
- compute and interpret dimension reductions using PCA
- apply PCA to examine temporal trends

🔧 Preparation

Make sure you have these packages installed:

```
install.packages(c("tidyverse", "kableExtra",  "viridisLite", "plotly", "mvtnorm", "GGally"))
```

📖 Reading

- Textbook section 4.3, 12.2

👋 Getting started

If you are in a zoom tutorial, say hello in the chat. If in person, do say hello to your tutor and to your neighbours.

⚙️ Exercises

1. Logistic regression

This question expects you to work through some equations for logistic regression, by hand, using the following data:

```
library(tidyverse)
library(kableExtra)
d <- tibble(x=c(1.5, 2.0, 2.1, 2.2, 2.5, 3, 3.1, 3.9, 4.1), y=c(0,0,0,1,0,1,0,1,1))
kable(d) %>%
  kable_styling(full_width=FALSE)
```

x	y
1.5	0

x	y
2.0	0
2.1	0
2.2	1
2.5	0
3.0	1
3.1	0
3.9	1
4.1	1

a.

Write out likelihood function, as function of β_0 and β_1. (The equation in lecture 3a, at the top of slide 9 is the one you need.)

b.

Show that the log likelihood

$$\sum_{i=1}^{9}\{y_i \log p(x_i) + (1 - y_i) \log (1 - p(x_i))\}$$

where $p(x_i) = P(Y = 1|x_i) = \frac{1}{e^{-z_i}+1} = \frac{e^{z_i}}{e^{z_i}+1}$ and $z_i = \beta_0 + \beta_1 x_i$, can be written as

$$\sum_{i=1}^{9}\{y_i(\beta_0 + \beta_1 x_i) - \log (1 + e^{\beta_0+\beta_1 x_i})\}$$

.

Justify each of the steps you make in the algebra. (You can fill in the gaps below.)

$$
\begin{aligned}
\log\ l(\beta_0, \beta_1) &= \sum_{i=1}^{9} \underline{\hspace{3cm}} && \text{first step and explanation} \\
&= \sum_{i=1}^{9} \ \{y_i(\log p(x_i) - \log(1 - p(x_i))) + \log(1 - p(x_i))\} && \text{explain} \\
&= \sum_{i=1}^{9} \ \underline{\hspace{4cm}} && \text{difference of logs is a log of quotient} \\
&= \sum_{i=1}^{9} \{y_i \log \frac{\frac{e^{z_i}}{e^{z_i}+1}}{(1 - \frac{e^{z_i}}{e^{z_i}+1})} + \log(1 - \frac{e^{z_i}}{e^{z_i} + 1})\} && \text{explain} \\
&= \sum_{i=1}^{9} \underline{\hspace{7cm}} && \text{reduce} \\
&= \sum_{i=1}^{9} \{y_i \log e^{z_i} + \log \frac{1}{1 + e^{z_i}}\} && \text{explain} \\
&= \sum_{i=1}^{9} \underline{\hspace{3.5cm}} && \text{log of exp, and invert quotient} \\
&= \sum_{i=1}^{9} \{y_i(\beta_0 + \beta_1 x_i) - \log{(1 + e^{\beta_0 + \beta_1 x_i})}\} && \text{substitute } z_i = \beta_0 + \beta_1 x_i
\end{aligned}
$$

c.

Plot the function for different values of β_0 and β_1, or if you prefer, you can solve the equation analytically, to find the maximum, and thus to provide parameter estimates.

```
likelihood <- function(x, y, b0, b1) {
  sum(y*(b0+b1*x)-log(1+exp(b0+b1*x)))
}
d_grid <- expand_grid(b0 = seq(-8, 4, 0.05),
                      b1 = seq(1, 4, 0.05))
d_grid <- d_grid %>%
  rowwise() %>%
  mutate(l = likelihood(d$x, d$y, b0, b1)) %>%
  ungroup()
estimates <- d_grid %>%
  arrange(desc(l)) %>%
  slice(1)
estimates
ggplot(d_grid) +
  geom_tile(aes(x=b0, y=b1, fill=l)) +
  scale_fill_viridis_c() +
  geom_point(data=estimates, aes(x=b0, y=b1),
             colour="black") +
  theme_bw() +
  theme(aspect.ratio=1)
```

d.

Check that you got it correct, by actually fitting the model.

```
# This code fits the model to check our calculations
library(tidymodels)
library(broom)
logistic_mod <- logistic_reg() %>%
  set_engine("glm") %>% #<<
  set_mode("classification") %>% #<<
  translate()

d <- d %>%
  mutate(y_f = factor(y))

d_fit <-
  logistic_mod %>%
  fit(y_f ~ x,
      data = d)

tidy(d_fit)
glance(d_fit)
```

e.

Write down the model equation using the parameter estimates.

f.

Plot your data and the fitted model.

```
# Change the beta values as necessary
d <- d %>%
  mutate(pred = (exp(YOUR_BETAHAT_0 + YOUR_BETAHAT_1*x))/(1+exp(YOUR_BETAHAT_0 + YOUR_
BETAHAT_1*x)))
ggplot(d, aes(x=x, y=y)) +
  geom_point() +
  geom_line(aes(y=pred), colour = "#ff7f00")
```

2. Principal Component Analysis

Here we are going to examine cross-rates for different currencies relative to the US dollar, to examine how the currencies changed as COVID-19 appeared. Some currencies moved in similar directions, and some opposite, some reacted strongly, and others not at all. PCA can help you to extract these differences.

A cross-rate is *an exchange rate between two currencies computed by reference to a third currency, usually the US dollar.*

The data file `rates_Nov19_Mar20.csv` was extracted from https://openexchangerates.org (https://openexchangerates.org).

a.

What's the data? Make a plot of the Australian dollar against date. Explain how the Australian dollar has changed relative to the US dollar over the 5 month period.

```
library(tidyverse)
rates <- read_csv(here::here("data/rates_Nov19_Mar20.csv"))
ggplot(rates, aes(x=date, y=AUD)) + geom_line()
```

b.

You are going to work with these currencies: AUD, CAD, CHF, CNY, EUR, GBP, INR, JPY, KRW, MXN, NZD, RUB, SEK, SGD, ZAR. List the names of the countries and currency name that these codes refer to. Secondary question: why is the USD a constant 1 in this data.

c.

The goal of the principal component analysis is to examine the relative movement of this subset of currencies, especially since coronavirus emerged until the end of March. PCA is used to summarise the volatility (variance) in the currencies, relative to each other. To do this you need to:

- Standardise all the currencies, individually. The resulting values will have a mean 0 and standard deviation equal to 1.
- Flip the sign so that high means the currency strengthened against the USD, and low means that it weakened. Its easier to explain trends, if you don't need to talk with double-negatives.
- Make a plot of all the currencies to check the result.

```
library(viridisLite)
library(plotly)
rates_sub <- rates %>%
  select(date, AUD, CAD, CHF, CNY, EUR, GBP, INR, JPY, KRW, MXN, NZD, RUB, SEK, SGD,
 ZAR) %>%
  mutate_if(is.numeric, function(x) -1*(x-mean(x))/sd(x))
rates_sub_long <- rates_sub %>%
  pivot_longer(cols=AUD:ZAR, names_to="currency", values_to="crossrate")
ggplot(rates_sub_long, aes(x=date, y=crossrate, colour=currency)) + geom_line() +
  scale_colour_viridis_d("")
# ggplotly() Make an interactive plot to browse the currencies
```

d.

Conduct a principal component analysis on the subset of currencies. The base function `prcomp` can be used for this. You need to work from a wide format of the data, where dates are in the columns, and currencies are in the rows. Normally, PCA operates on standardised variables but for this data, you need to NOT standardise each date. Think about why this is best.

- Why is this data considered to be high-dimensional?
- Make a scree plot to summarise the variance explained by cumulative principal components. How much of the total variation do two PCs explain?
- Plot the first two principal components. Write a summary of what you learn about the similarity and difference between the currencies.
- Plot the loadings for PC1. Add a base line set at $1/\sqrt{15}$. Why use this as a guide? What time frame generated a big movement (or divergence) in the currencies? Which currencies strengthened relative to the USD in that period? What happened to the Australian dollar? Answer these questions in a paragraph, written in your own words.
- Do the same analysis for PC2. In what time frame was there another movement of currencies? Which currencies primarily strengthened, and which weakened during this period?
- Finish with a paragraph summarising what variability the principal components analysis is summarising. What dimension reduction is being done?

```
library(ggrepel)
library(kableExtra)
rates_sub_wide <- rates_sub_long %>%
  pivot_wider(id_cols=currency, names_from=date, values_from = crossrate)
rates_pca <- prcomp(rates_sub_wide[,-1], scale=FALSE)
screeplot(rates_pca, type="l")
summary(rates_pca)
rates_pca$x %>%
  as_tibble() %>%
  mutate(currency = rates_sub_wide$currency) %>%
  ggplot(aes(x=PC1, y=PC2)) +
    geom_point() +
    geom_text_repel(aes(x=PC1, y=PC2, label=currency)) +
  theme(aspect.ratio=1)
rates_pc_loadings <- as_tibble(rates_pca$rotation[,1:2]) %>%
  mutate(date = rownames(rates_pca$rotation),
         indx = 1:nrow(rates_pca$rotation),
         ymin=rep(0, nrow(rates_pca$rotation)))
ggplot(rates_pc_loadings) +
  geom_hline(yintercept=c(-1/sqrt(nrow(rates_pca$rotation)),
                          1/sqrt(nrow(rates_pca$rotation))), colour="red") +
  geom_errorbar(aes(x=indx, ymin=ymin, ymax=PC1)) +
  geom_point(aes(x=indx, y=PC1))
ggplot(rates_pc_loadings) +
  geom_hline(yintercept=c(-1/sqrt(nrow(rates_pca$rotation)),
                          1/sqrt(nrow(rates_pca$rotation))), colour="red") +
  geom_errorbar(aes(x=indx, ymin=ymin, ymax=PC2)) +
  geom_point(aes(x=indx, y=PC2))
```