Homework 4

Context

This assignment reinforces ideas in Module 4: Constrained Optimization. We focus specifically on implementing quantile regression and LASSO.

Github link: https://github.com/xxou/bios731_hw4_ou

```
library(tidyverse)
library(corrplot)
library(quantreg)
library(glmnet)
library(LowRankQP)
library(gt)
```

Problem 1

```
cannabis_dt <- readRDS(here::here("data", "cannabis-2.rds"))
```

Q(1)

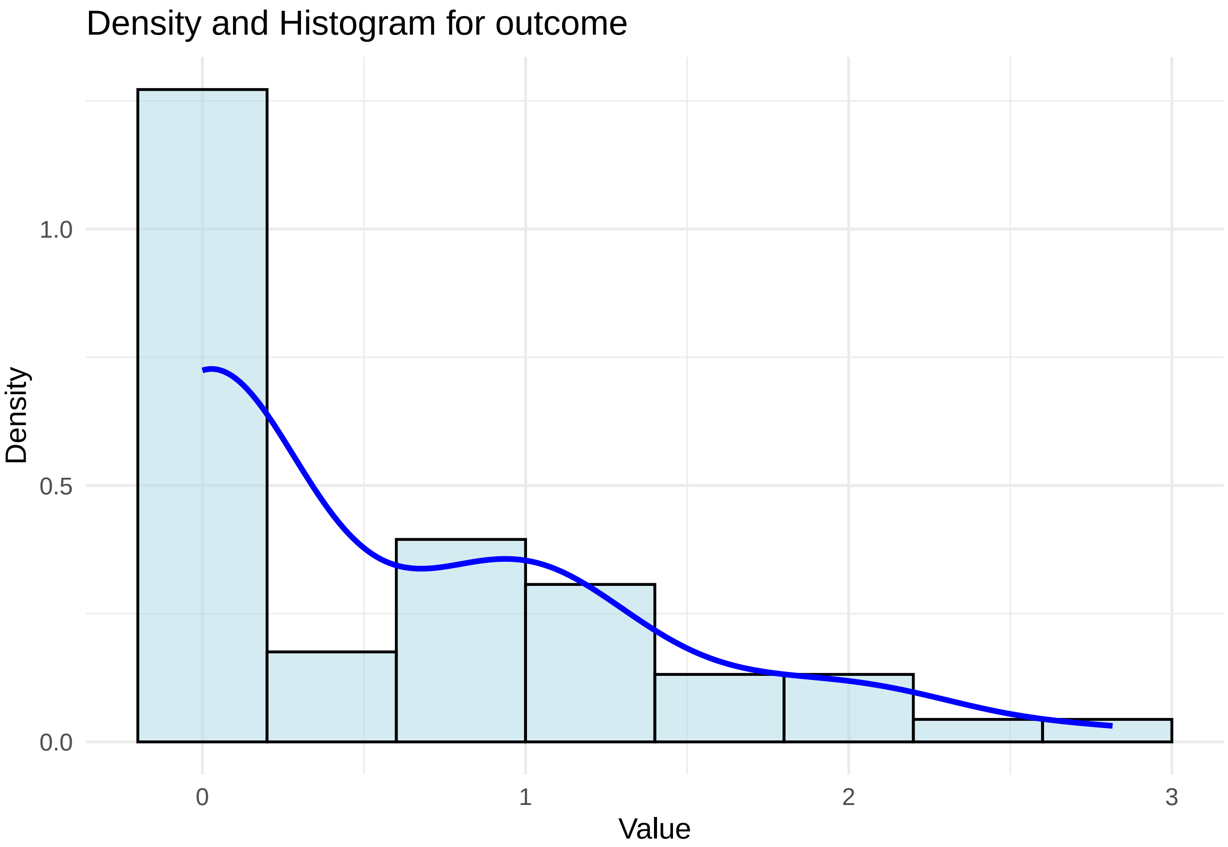
```
dim(cannabis_dt)
```

```
## [1] 57 29
```

n is 57 and p is 27 (29 - 1 - 1, id and t_mmr1)

Q(2) distribution of outcome

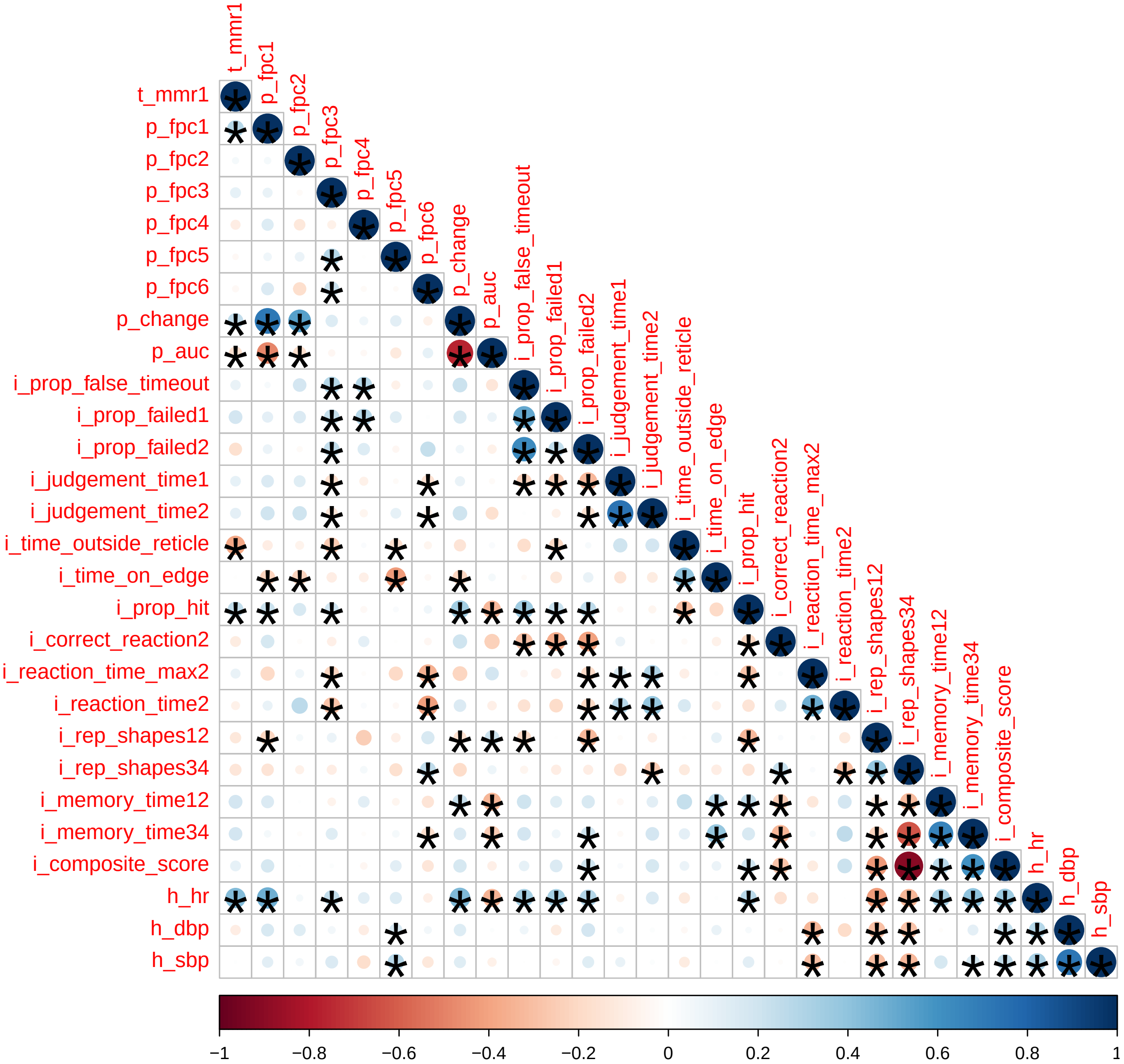
```
cannabis_dt |>
ggplot(aes(x = t_mmr1)) +
  geom_histogram(aes(y = ..density..), binwidth = 0.4,
                 fill = "lightblue", color = "black", alpha = 0.5) +
  geom_density(color = "blue", size = 1) +
  theme_minimal() +
  labs(title = "Density and Histogram for outcome",
       x = "Value",
       y = "Density")
```



It is asymmetric double exponential distribution.

Q(3) correlation

```
cor_dt <- cannabis_dt |>  select(-id)
M = cor(cor_dt)
res <- cor.mtest(M, conf.level = 0.95)
corrplot(M, insig = "label_sig", p.mat = res$p, type = "lower", sig.level = 0.05)
```



The outcome is asymmetric double exponential distribution. The outcome is correlated with `p_fpc1`, `p_change`, `p_auc`, `i_time_outside_reticle`, `i_prop_hit` and `h_hr`. The plot shows that many variables in the dataset are weakly to moderately correlated. A few variable pairs exhibit stronger correlations, such as `p_fpc1` with `p_auc`, and `i_correct_reaction2` with `i_prop_hit`, which are positively correlated. Strong negative correlations also appear, such as between `p_change` and `p_auc`, `i_rep_shapes` and `i_composite_score`.

Problem 2

I used the lp function from the lpSolve package to solve LP problem, employing the strategy of separating a number into its positive and negative components.

```
source(here::here("source", "my_rq.R"))
```

```r
# my
Y = cannabis_dt$t_mmr1
X = cannabis_dt |> select(p_change, h_hr, i_composite_score)
X = cbind(`(Intercept)` = 1, X)

my_rq(Y, X, tau = 0.25) -> beta_0.25
my_rq(Y, X, tau = 0.5) -> beta_0.5
my_rq(Y, X, tau = 0.75) -> beta_0.75
```

```r
# rq()
rq_0.25 <- rq(t_mmr1 ~ p_change + h_hr + i_composite_score, data = cannabis_dt, tau = 0.25)
rq_0.5 <- rq(t_mmr1 ~ p_change + h_hr + i_composite_score, data = cannabis_dt, tau = 0.5)
rq_0.75 <- rq(t_mmr1 ~ p_change + h_hr + i_composite_score, data = cannabis_dt, tau = 0.75)
```

```r
# mean  using linear regression
ols <- lm(t_mmr1 ~ p_change + h_hr + i_composite_score, data = cannabis_dt)
```

```r
# Extract coefficients from your function and rq
Variable = names(X)

my_results <- data.frame(
  Variable,
  my_rq_0.25 = beta_0.25,
  my_rq_0.5 = beta_0.5,
  my_rq_0.75 = beta_0.75
)

rq_results <- data.frame(
  Variable,
  rq_0.25 = coef(rq_0.25),
  rq_0.5 = coef(rq_0.5),
  rq_0.75 = coef(rq_0.75)
)

# Join both results
comparison_table <- full_join(my_results, rq_results, by = "Variable") |>
  mutate(ols_mean = ols$coefficients)

# Rearranged comparison table
comparison_table |>
  gt() |>
  tab_header(
    title = "Comparison of Coefficients from my_rq() and rq()"
  ) |>
  fmt_number(
    columns = -Variable,
    decimals = 4
  ) |>
  cols_label(
    my_rq_0.25 = "my_rq",
    rq_0.25 = "rq",
    my_rq_0.5 = "my_rq",
    rq_0.5 = "rq",
```

Comparison of Coefficients from my_rq() and rq()

	tau = 0.25		tau = 0.5		tau = 0.75		mean
Variable	my_rq	rq	my_rq	rq	my_rq	rq	ols
(Intercept)	-0.1501	-0.1501	-0.2834	-0.2834	-1.1140	-1.1140	-0.1788
p_change	0.0114	0.0114	0.0122	0.0122	0.0042	0.0042	0.0077
h_hr	0.0082	0.0082	0.0136	0.0136	0.0273	0.0273	0.0144
i_composite_score	0.3841	0.3841	0.1982	0.1982	-0.5809	-0.5809	-0.2382

```
    my_rq_0.75 = "my_rq",
    rq_0.75 = "rq",
    ols_mean = "ols"
  ) |>
  tab_spanner(label = "tau = 0.25", columns = c(my_rq_0.25, rq_0.25)) |>
  tab_spanner(label = "tau = 0.5", columns = c(my_rq_0.5, rq_0.5)) |>
  tab_spanner(label = "tau = 0.75", columns = c(my_rq_0.75, rq_0.75)) |>
  tab_spanner(label = "mean", columns = c(ols_mean))
```

The comparison shows that the coefficients estimated by the `my_rq()` function match those from `rq()`. While OLS provides average effects, quantile regression reveals more nuanced patterns—such as the strong positive effect of `i_composite_score` at lower quantiles that turns negative at higher quantiles, and a gradually increasing effect of `h_hr` across quantiles.

Problem 3

Q(1)

Let $\beta_j = \beta_j^+ - \beta_j^-$, where $\beta_j^+, \beta_j^- \geq 0$,

$$\tilde{X} = [X, -X] \in \mathbb{R}^{n*2p}, \quad B = \begin{bmatrix} \beta_j^+ \\ \beta_j^- \end{bmatrix}$$

$$\min \quad \{||y - X\beta||_2^2\} = \min \quad \{(y - \tilde{X}B)^T(y - \tilde{X}B)\}, \quad st \quad \mathbf{1}_{2p}^T B \leq \lambda$$

This problem becomes

$$\min \quad \{\frac{1}{2}B^T\tilde{X}^T\tilde{X}B + d^T B\}, \quad d = -\tilde{X}^T y \quad st \quad \mathbf{1}_{2p}^T B \leq \lambda$$

$$\tilde{X}^T\tilde{X} = \begin{bmatrix} X^T X & -X^T X \\ -X^T X & X^T X \end{bmatrix}$$

This matrix is not positive definite, thus QP solvers like solve.QP cannot be used.

Q(2)

```r
source(here::here("source", "my_lasso.R"))
```

```r
logY = log(cannabis_dt$t_mmr1 + 1e-06)
X = cannabis_dt |> select(-id, -t_mmr1) |> as.matrix()
```

```r
# Use glmnet with cross-validation
set.seed(234)

lasso_mod <- cv.glmnet(X, logY, alpha = 1, standardize = TRUE)
# plot(lasso_mod,"lambda")
lasso_mod$lambda.min
```

```
## [1] 0.8314611
```

```r
beta_glmnet <- coef(lasso_mod, s = "lambda.min")[-1]
```

Use the lambda.min from glmnet in my function:

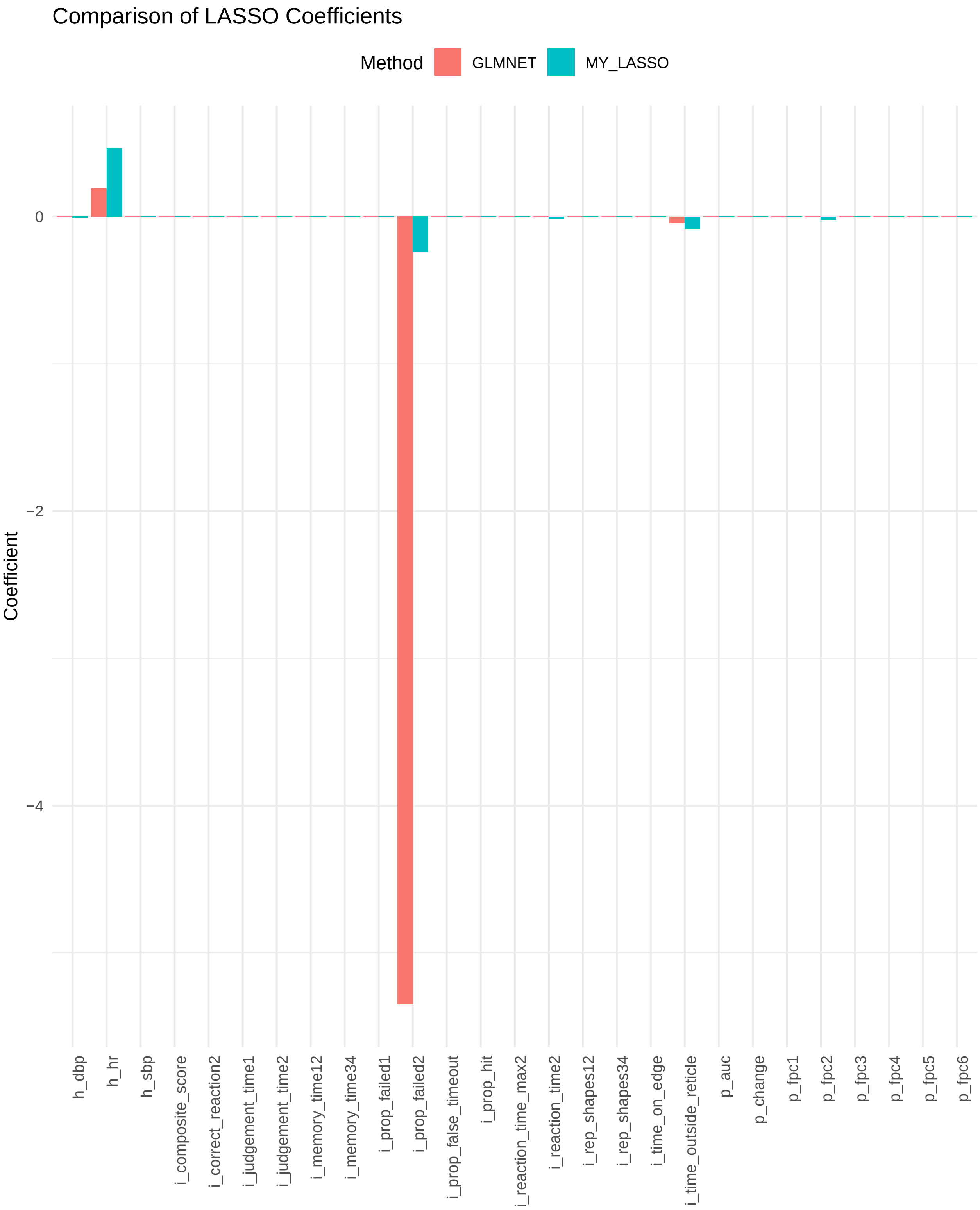
```r
beta_my <- my_lasso(logY, scale(X), lambda = lasso_mod$lambda.min)
```

Comparision:

```r
df <- data.frame(
  Variable = names(cannabis_dt)[-c(1,2)],
  GLMNET = beta_glmnet,
  MY_LASSO = beta_my
)
```

```r
df_long <- pivot_longer(df,
                        cols = c("GLMNET", "MY_LASSO"),
                        names_to = "Method",
                        values_to = "Coefficient")

ggplot(df_long, aes(x = Variable, y = Coefficient, fill = Method)) +
  geom_bar(stat = "identity", position = "dodge") +
  theme_minimal() +
  theme(legend.position = "top",
        axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "Comparison of LASSO Coefficients",
       x = "",
       y = "Coefficient")
```



Overall, both methods select similar variables with nonzero coefficients, which shows some consistency in variable selection. But there are some differences in the magnitude. For example, the coefficient of i__prop__failed2 in glmnet is much larger, while in my__lasso the value is smaller and more moderate. This may be caused by the different optimization algorithms or cross-validation process.