Problem set 5

Dili Maduabum, Joshua Bailey

2024-02-16

Problem 1

1

Paper read!

2

Social utility refers to the benefit or satisfaction that individuals derive from their actions within a social context, not solely based on their own direct outcomes but also influenced by the actions and outcomes of others in their social network or community. It's a concept that encapsulates the notion that individuals' utility or happiness can be affected by the choices, behaviours, and welfare of others, in addition to their own personal outcomes.

Social learning is the process through which individuals acquire new information, behaviours, or strategies by observing or interacting with others. It involves the diffusion of knowledge and practices through a social network, where individuals can learn from the experiences and decisions of their peers.

The difference between social utility and social learning lies in their focus and mechanism. Social utility is about the derived satisfaction or benefit from being part of a social setting and how others' actions and outcomes affect an individual's own utility. Social learning, in contrast, is focused on the acquisition of knowledge, behaviours, or norms through the observation of or interaction with others. It is a mechanism for information diffusion and behavioural adaptation within a social network, emphasizing the cognitive aspect of learning from the environment and others' experiences.

3

Designing an ideal experiment to differentiate between social utility and social learning involves isolating the effects of social influence on individual decision-making from the effects of learning from others' experiences. The key distinction between these two mechanisms is the role of private versus public information and the subsequent effects on individual behavior. Social utility refers to the value individuals derive from conforming to the behaviors of their social group, often driven by network externalities, where the utility of a product or action increases as more people adopt it. On the other hand, social learning involves updating one's beliefs and decisions based on observing the actions and outcomes of others, under the assumption of imperfect information about the quality of an option.

This requires a setup where individuals make decisions based not on learning about the quality of options from others but on the intrinsic value they place on aligning with the behaviors of their social group. Conversely, to isolate social learning, the experimental design should allow individuals to update their beliefs about the quality of options based on the observed choices of others, without the influence of wanting to conform to a group norm for its own sake.

4

A regression of the number of viewers in later weeks on the number of viewers in the first week would not have a causal interpretation due to potential unobserved confounding variables. One such confounder could be the inherent attractiveness or quality of the movie, which simultaneously impacts viewership in the first week and viewership in later weeks. For example, a highly anticipated movie might attract a large audience in its first week due to its perceived quality, marketing, or star cast, and continue to attract viewers in subsequent weeks through word of mouth or sustained interest. This inherent quality affects both initial and later viewership but is not directly measured in a simple regression, leading to confounded estimates of the effect of first-week viewership on later viewership.

5

The authors exploit weather variations as an instrumental variable. This approach relies on the assumption that weather variations are random and thus not directly related to the movie's quality, but can significantly affect viewership numbers due to people's preference for indoor activities, like watching movies, during bad weather. By using weather as an instrument, the analysis can isolate the effect of early viewership on later viewership, effectively controlling for the unobserved movie quality that might simultaneously influence viewership across different weeks. This methodology allows for a cleaner estimation of the causal effect of social spillovers on movie consumption, sidestepping the confounding effect of inherent movie quality.

6

If Gilchrist and Sands had observed the confounding variable and implemented selection on observables under the conditional independence assumption, they could have potentially estimated the causal effect of first-week viewership on later-week viewership. This approach assumes that controlling for observed confounders—variables that affect both the treatment (first-week viewership) and the outcome (later-week viewership)—would allow for a causal interpretation of the relationship between the two.

However, merely observing the confounding variable and controlling for it would not necessarily help in distinguishing between social learning and social utility. The conditional independence assumption helps identify the causal effect of first-week viewership on later-week viewership, assuming all confounders are observed and correctly modeled. Still, this method alone does not differentiate whether the mechanism behind this effect is due to social learning (where individuals update their information based on others' actions) or social utility (where individuals derive utility from consuming a product in a social context, independent of information changes).

To disentangle social learning from social utility, one would need to identify mechanisms that specifically relate to information transmission or shared consumption experiences. For example, if the effect of first-week viewership on later-week viewership was found to be stronger for movies with more ambiguous quality (suggesting viewers are updating their beliefs based on others' actions), this would indicate social learning. Conversely, if the effect was uniform across movies of different quality levels, suggesting that the social aspect of viewing is driving the effect, this could indicate social utility. Gilchrist and Sands' approach using weather as an instrumental variable aims to capture the exogenous variation in first-week viewership unrelated to movie quality, potentially offering insights into the social spillover effects. However, observing the confounding variable and controlling for it through selection on observables might not provide clear evidence to distinguish between these two mechanisms without further assumptions or additional data on the nature of social interactions related to moviegoing.

7

1. **Variation in Responses Based on Information Precision**: Social learning theory suggests that the impact of observing others' actions (e.g., early viewership on later viewership) would vary depend-

ing on the precision of the information available to the observers. In contexts where there's greater uncertainty about a movie's quality, the influence of early viewers on later viewers should be more pronounced because potential viewers have more to learn from the actions of early viewers. This variation in response is consistent with social learning because it underlines the role of information acquisition in decision-making. In contrast, social utility, which derives from the shared experience of consumption, does not inherently vary with the precision of information about the product's quality.

2. **Differential Impact Based on Observable Characteristics of Consumers**: Social learning models also predict that the impact of early viewership on later viewership may differ across groups with different propensities for learning. For instance, if younger viewers are more influenced by peer behavior than older viewers, one might observe stronger social spillover effects among movies targeted at younger audiences. This implication is consistent with social learning as it reflects the idea that the propensity to update beliefs based on others' actions can vary across different segments of the population. Social utility does not offer a clear reason for why such differential impacts based on consumer characteristics should exist, as the enjoyment derived from shared consumption experiences does not inherently depend on the consumers' propensity to learn from others.

8

Gilchrist and Sands proxy for realized movie quality using ratings by expert reviewers, specifically from IMDb's top 1000 voters. This group is characterized by IMDb as the 1000 people who have voted for the most titles in their ratings polls. Movies are then categorized into quality terciles based on these ratings, capturing a significant dimension of demand as top tercile movies, on average, sell significantly more tickets over the first 6 weeks compared to bottom tercile movies.

They also use information about the movie's production budget. The rationale behind this is that advertising budgets, which influence the amount of information available to the public before a movie's release, are generally set as a fixed percentage of production budgets. Production budgets are thus positively correlated with the precision of prior knowledge about a movie.

While these proxies are insightful and useful for empirical analysis, they may not fully capture the nuances of movie quality or the exact precision of information. Movie quality, for instance, is subjective and can be influenced by factors beyond critic ratings, such as viewer preferences and cultural impacts, which are harder to quantify. Similarly, the precision of information might be affected by factors other than advertising, such as word-of-mouth or social media influence, which are not directly tied to the production budget.

9

Gilchrist and Sands do not expect the conditional expectation of a movie's total viewership with respect to weather to have a causal interpretation because of the strategic behavior of film studios in choosing release dates. Film studios tend to release movies at times when they expect higher demand, both in terms of the number and quality of movies available in theaters. This strategic timing means that the observed relationship between weather and movie viewership could be confounded by the studios' anticipation of demand variations across different times of the year.

The seasonality of movie viewership is driven by both underlying demand and supply-side considerations, where the supply side (i.e., the studios) takes into account expected demand in timing releases. This includes releasing certain types of movies during specific seasons to maximize viewership i.e. blockbuster movies are often released during summer and holiday seasons when audience turnout is expected to be higher. Thus, any observed correlation between weather and viewership could be influenced by these strategic release decisions, making it difficult to interpret the relationship causally.

10

The regression of opening weekend viewership on indicators for day of the week of the year, week, year, and holidays is not a saturated regression because it does not include every possible combination of these indicators as separate variables. A saturated model would include a unique indicator variable for every unique combination of the levels of these categorical variables, effectively allowing for a separate intercept for every combination. Given the vast number of combinations (especially when considering all weeks of all years, along with holidays), such a regression would be extremely high-dimensional and possibly exceed the number of observations available, leading to overfitting where the model perfectly fits the training data but performs poorly on new data.

They likely do not consider a saturated regression because it would involve an enormous number of parameters, leading to issues with overfitting and interpretability. High-dimensional models can be computationally challenging to estimate and may require regularization techniques to obtain stable estimates.That said, one advantage of such a model is that the coefficients of a saturated model can be straightforwardly interpreted.

11

11.a A movie should appear in the dataset at least 18 times. Each has a record for the weekend (Friday, Saturday and Sunday) from the opening weekend to at least 6 weekends later (for the ones kept). The ones dropped were not in theaters for more than 6 weekends.

11.b

```
#keeping films that aren't dropped
films_used <- films |>
  filter(dropped != 1)
```

11.c

```
# day when 12 Rounds came in
round_12_date <- as.Date("2009-03-27")

# Define the number of days to add
days_before <- 17984 #number under 12 Rounds "date" column

# Days prior to the
reference_date <- round_12_date - days_before + 1

# Print the new date
print(reference_date)
```

```
## [1] "1960-01-01"
```

11.d

```
films_used_d <- films_used |>
  mutate(movie_date = as.Date(reference_date + date)) |>
  #putting the release_date in the 4th column
  select(title, production_budget, release_yr,
         movie_date, sat_date, everything())

films_used_d[, c("title", "movie_date")]
```

```
## # A tibble: 24,855 x 2
##    title                movie_date
##    <chr>                <date>
##  1 (500) Days of Summer 2009-08-08
##  2 12 Rounds            2009-03-28
##  3 127 Hours            2010-11-26
##  4 13 Going on 30       2004-04-25
##  5 1408                 2007-06-24
##  6 16 Blocks            2006-03-05
##  7 17 Again             2009-04-19
##  8 2 Fast 2 Furious     2003-06-07
##  9 2012                 2009-11-15
## 10 21                   2008-03-28
## # i 24,845 more rows
```

11.e

```
#first using sat_date to get the date for each saturday
films_used_date <- films_used_d |>
  #getting the day for saturday
  mutate(sat_day = reference_date + sat_date) |>
  mutate(sat_day_of_week = wday(sat_day, label = TRUE)) |>
  mutate(
    fri_dummy = ifelse(movie_date == sat_day - 1, 1, 0),
    sat_dummy = ifelse(movie_date == sat_day , 1, 0),
    #reasoning... there was no movie released on Sunday....
    sun_dummy = ifelse(movie_date == sat_day + 1, 1, 0)
  ) |> arrange(title)

films_used_date[, c("title", "movie_date","sat_day" ,"fri_dummy", "sat_dummy", "sun_dummy")]
```

```
## # A tibble: 24,855 x 6
##    title                movie_date sat_day    fri_dummy sat_dummy sun_dummy
##    <chr>                <date>     <date>         <dbl>     <dbl>     <dbl>
##  1 (500) Days of Summer 2009-08-08 2009-08-08         0         1         0
##  2 (500) Days of Summer 2009-08-07 2009-08-08         1         0         0
##  3 (500) Days of Summer 2009-08-09 2009-08-08         0         0         1
##  4 (500) Days of Summer 2009-08-14 2009-08-15         1         0         0
##  5 (500) Days of Summer 2009-08-15 2009-08-15         0         1         0
##  6 (500) Days of Summer 2009-08-16 2009-08-15         0         0         1
##  7 (500) Days of Summer 2009-08-21 2009-08-22         1         0         0
##  8 (500) Days of Summer 2009-08-23 2009-08-22         0         0         1
##  9 (500) Days of Summer 2009-08-22 2009-08-22         0         1         0
## 10 (500) Days of Summer 2009-08-30 2009-08-29         0         0         1
## # i 24,845 more rows
```

11.f

```
#creating dummies for week using fastDummies
films_used_date <- films_used_date |>
  arrange(title, sat_day) |>
  group_by(title) |>
  # Assign numeric labels to unique elements of sat_date within each title
```

```
  mutate(week = as.integer(factor(sat_date)))


#Now using fast dummies...
films_used_date <- dummy_cols(films_used_date, select_columns = 'week')
films_used_date[, c("title", "movie_date" ,"week_1", "week_2", "week_3")]
```

```
## # A tibble: 24,855 x 5
##    title                movie_date week_1 week_2 week_3
##    <chr>                <date>      <int>  <int>  <int>
##  1 (500) Days of Summer 2009-08-08      1      0      0
##  2 (500) Days of Summer 2009-08-07      1      0      0
##  3 (500) Days of Summer 2009-08-09      1      0      0
##  4 (500) Days of Summer 2009-08-14      0      1      0
##  5 (500) Days of Summer 2009-08-15      0      1      0
##  6 (500) Days of Summer 2009-08-16      0      1      0
##  7 (500) Days of Summer 2009-08-21      0      0      1
##  8 (500) Days of Summer 2009-08-23      0      0      1
##  9 (500) Days of Summer 2009-08-22      0      0      1
## 10 (500) Days of Summer 2009-08-30      0      0      0
## # i 24,845 more rows
```

11.g

```
#using the "Fast Dummies" library... to automatically create dummies for year
film <- dummy_cols(films_used_date, select_columns = 'release_yr')

film[, c("title", "release_yr", "release_yr_2009", "release_yr_2010")]
```

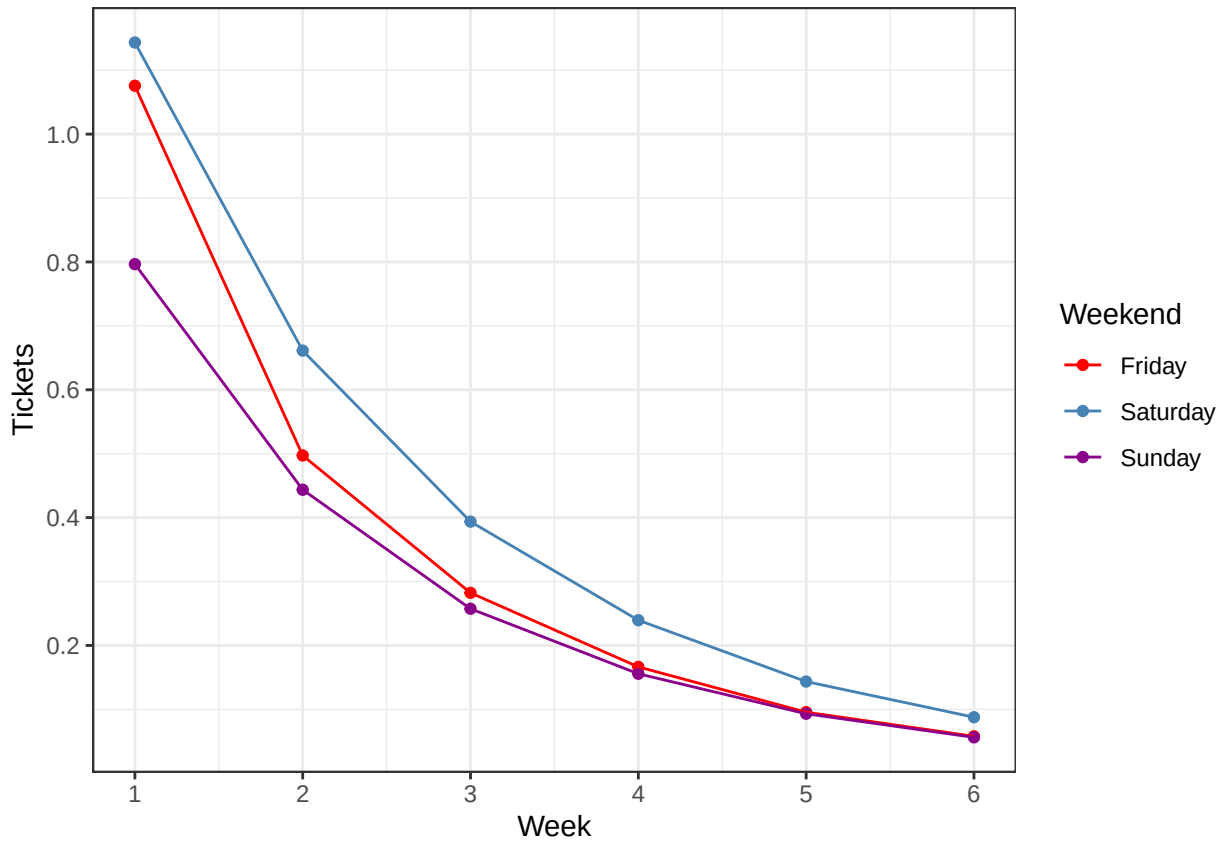
```
## # A tibble: 24,855 x 4
##    title                release_yr release_yr_2009 release_yr_2010
##    <chr>                     <dbl>           <int>           <int>
##  1 (500) Days of Summer       2009               1               0
##  2 (500) Days of Summer       2009               1               0
##  3 (500) Days of Summer       2009               1               0
##  4 (500) Days of Summer       2009               1               0
##  5 (500) Days of Summer       2009               1               0
##  6 (500) Days of Summer       2009               1               0
##  7 (500) Days of Summer       2009               1               0
##  8 (500) Days of Summer       2009               1               0
##  9 (500) Days of Summer       2009               1               0
## 10 (500) Days of Summer       2009               1               0
## # i 24,845 more rows
```

11.h

```
#combine the weekends
temp <- film |>
 mutate(weekend = case_when(
   sat_dummy == 1 ~ "Saturday",
   fri_dummy == 1 ~ "Friday",
   sun_dummy == 1 ~ "Sunday",
```

```
 )) |>
  group_by(week, weekend) |>
  summarize(mean = mean(tickets, na.rm = TRUE))

temp |>
  ggplot(aes(x = week, y = mean, color = as.factor(weekend))) +
  geom_point() +
  geom_line() +
  scale_color_manual(values = c("Saturday" = "#4682B4",
                                "Friday" = "red",
                                "Sunday" = "#8B008B")) +
  labs(color = "Weekend",
       y = "Tickets",
       x = "Week") +
  scale_x_continuous(breaks = scales::pretty_breaks(n = 6)) +  # Set x-axis ticks
  scale_y_continuous(breaks = scales::pretty_breaks(n = 6)) +  # Set y-axis ticks
  theme_bw()
```



12

NOT NEEDED

13

```
#subset colnames that have the hh in them
holiday <- str_subset(colnames(film), "hh")

#make the things in holiday "add"
holiday_dummy <- str_c(holiday, collapse = " + ")

#day of the week dummies
weekend_dummy <- str_c(str_subset(colnames(film), "dummy"), collapse = " + ")

#week of the year dummies
week_dummy <- str_c(str_subset(colnames(film), "week_"), collapse = " + ")

#year of the week dummy
year_dummy <- str_c(str_subset(colnames(film), "release_yr_"), collapse = " + ")

#combine
mod1 <- glue("tickets ~ {weekend_dummy} + {week_dummy} + {year_dummy} + {holiday_dummy}")

#fit a regression model
reg_mod1 <- lm(as.formula(mod1), data = film)


film <- film |>
  mutate(pred_tickets = predict(reg_mod1, film)) |>
  mutate(abnormal_viewership = tickets - pred_tickets)

film[, c("tickets","pred_tickets", "abnormal_viewership", "sat_day")]
```

```
## # A tibble: 24,855 x 4
##    tickets pred_tickets abnormal_viewership sat_day
##      <dbl>        <dbl>               <dbl> <date>
##  1  0.185         1.07              -0.890  2009-08-08
##  2  0.159         0.991             -0.833  2009-08-08
##  3  0.155         0.933             -0.777  2009-08-08
##  4  0.126         0.518             -0.393  2009-08-15
##  5  0.153         0.602             -0.449  2009-08-15
##  6  0.117         0.460             -0.343  2009-08-15
##  7  0.0981        0.296             -0.198  2009-08-22
##  8  0.0808        0.237             -0.156  2009-08-22
##  9  0.125         0.379             -0.254  2009-08-22
## 10  0.0660        0.114             -0.0478 2009-08-29
## # i 24,845 more rows
```

14

```
weather <- read_dta("data/weather_collapsed_day.dta")

#adding www to the column names
```

```r
original_cols <- colnames(weather)

# adding prefix using the paste
colnames(weather) <- paste("www", original_cols, sep = "_")

weather
```

```
## # A tibble: 4,932 x 39
##    www_date   www_sat_date www_snow www_rain www_mat5_10 www_mat5_15 www_mat5_20
##    <date>     <date>          <dbl>    <dbl>       <dbl>       <dbl>       <dbl>
##  1 1982-01-03 1982-01-02      0.165  0.533        0.0391      0.0276      0.0318
##  2 1982-01-01 1982-01-02      0.227  0.446        0.0476      0.0405      0.0472
##  3 1982-01-02 1982-01-02      0.233  0.408        0.0150      0.0157      0.0508
##  4 1982-01-10 1982-01-09      0.422  0.0916       0.464       0.0742      0.0529
##  5 1982-01-08 1982-01-09      0.222  0.0314       0.0709      0.0606      0.119
##  6 1982-01-09 1982-01-09      0.435  0.00428      0.156       0.0510      0.0857
##  7 1982-01-17 1982-01-16      0.255  0.0538       0.386       0.0455      0.0352
##  8 1982-01-15 1982-01-16      0.420  0.0533       0.0916      0.0940      0.102
##  9 1982-01-16 1982-01-16      0.494  0.135        0.174       0.0322      0.0455
## 10 1982-01-24 1982-01-23      0.216  0.0675       0.158       0.0589      0.0388
## # i 4,922 more rows
## # i 32 more variables: www_mat5_25 <dbl>, www_mat5_30 <dbl>, www_mat5_35 <dbl>,
## #   www_mat5_40 <dbl>, www_mat5_45 <dbl>, www_mat5_50 <dbl>, www_mat5_55 <dbl>,
## #   www_mat5_60 <dbl>, www_mat5_65 <dbl>, www_mat5_70 <dbl>, www_mat5_75 <dbl>,
## #   www_mat5_80 <dbl>, www_mat5_85 <dbl>, www_mat5_90 <dbl>, www_mat5_95 <dbl>,
## #   www_prec_0 <dbl>, www_prec_1 <dbl>, www_prec_2 <dbl>, www_prec_3 <dbl>,
## #   www_prec_4 <dbl>, www_prec_5 <dbl>, www_cloud_0 <dbl>, ...
```

```r
weather_film <- film |>
  left_join(weather |> #all variables except the saturday variable
              select(-c("www_sat_date")),
            #combine on dates, automatically filters out dates that don't match
            by = c("movie_date" = "www_date"))
```

15

```r
# Select columns with names containing "www_"
www_columns <- str_subset(colnames(weather_film), "www_")

# Create a copy of the original dataframe
df <- weather_film

# Define regression formula with dummy variables
regressors <-  glue("~ {weekend_dummy} + {week_dummy} + {year_dummy} + {holiday_dummy}")

# Iterate over columns with names containing "www_"
for (columns in www_columns) {
  # Construct regression formula
  model <- paste(columns, regressors)

  # Generate names for predicted values and residuals
```

```
  pred_name <- paste("pred", columns, sep = "_")
  resid_name <- paste("abnormal", columns, sep = "_")

  # Add predicted values and residuals to the dataframe
  df <- df |>
      mutate(!!pred_name := predict(lm(as.formula(model), data = df), df)) |>
    #residuals = column - predicted_value_for_column
      mutate(!!resid_name := eval(parse(text = columns)) - eval(parse(text = pred_name)))
}

#remove the predicted and original values, keeping only the residuals
new_weather <- df |>
  select(-c(contains("pred_www"), starts_with("www")))
```

16

```
#combine
#fit a regression model
week_2_data <- new_weather |>
  filter(week_2 == 1)

#using the same regression
reg_mod2 <- lm(as.formula(mod1), data = week_2_data)


new_weather_film_wk2 <- week_2_data |>
  mutate(pred_tickets_wk_2 = predict(reg_mod2, week_2_data)) |>
  mutate(abnormal_viewership_wk_2 = tickets - pred_tickets_wk_2)


new_weather_film_wk2[, c("tickets", "pred_tickets_wk_2", "week_2", "abnormal_viewership_wk_2")]
```

```
## # A tibble: 4,143 x 4
##    tickets pred_tickets_wk_2 week_2 abnormal_viewership_wk_2
##      <dbl>             <dbl>  <int>                    <dbl>
##  1  0.126              0.455      1                   -0.330
##  2  0.153              0.615      1                   -0.462
##  3  0.117              0.394      1                   -0.277
##  4  0.689              0.412      1                    0.277
##  5  0.671              0.350      1                    0.321
##  6  0.976              0.572      1                    0.404
##  7  0.0668             0.394      1                   -0.327
##  8  0.116              0.455      1                   -0.340
##  9  0.119              0.615      1                   -0.497
## 10  0.0554             0.394      1                   -0.339
## # i 4,133 more rows
```

17

```
#Mak
#subsetting the data to just be week 1
week_1_data <- new_weather |>
  filter(week_1 == 1)

#creating the "abnormal viewerships in week 1"------------
mod1 <- glue("tickets ~ {weekend_dummy} + {week_dummy} + {year_dummy} + {holiday_dummy}")

#fit a regression model
reg_mod1 <- lm(as.formula(mod1), data = week_1_data)

new_weather_film_wk1 <- week_1_data |>
  mutate(pred_tickets_wk_1 = predict(reg_mod1, week_1_data)) |>
  mutate(abnormal_viewership_wk1 = tickets - pred_tickets_wk_1)
```

17.a OLS;

```
abnormal_weather_wk1_names <-
  str_subset(colnames(new_weather_film_wk1), "abnormal_www")

abnormal_weather_wk1 <-
  str_c(abnormal_weather_wk1_names, collapse = "+")

ols_glue <- glue("abnormal_viewership_wk1 ~ {abnormal_weather_wk1}")
ols_mod <- lm(as.formula(ols_glue),
     new_weather_film_wk1)

#modelsummary(list(ols_mod), output = "gt")
```

17.b

```
#subset the data to include the variables of interest
leaps_data <- new_weather_film_wk1 |>
  select(c(abnormal_viewership_wk1, all_of(abnormal_weather_wk1_names)))

forward <- regsubsets(abnormal_viewership_wk1 ~ .,
           data = leaps_data, method = "forward")
```

```
## Reordering variables and trying again:
```

```
# Get summary of the models
summary_forward <- summary(forward)

# Find the index of the model with the highest R-squared Adjusted
best_model_index_fwd <- which.max(summary_forward$adjr2) #9th model has the highest

# Get the names of predictors (coef) in the best model (9), without the intercept([-1])
best_adjr_predictors <- names(coef(forward, id = best_model_index_fwd)[-1])

# Print the selected predictors and the corresponding R-squared Adjusted value
best_adjr_predictors
```

```
## [1] "abnormal_www_rain"    "abnormal_www_mat5_60" "abnormal_www_mat5_85"
## [4] "abnormal_www_mat5_90" "abnormal_www_prec_1"  "abnormal_www_cloud_0"
## [7] "abnormal_www_cloud_4" "abnormal_www_cloud_5" "abnormal_www_cloud_8"
```

```
#running regressions based on the model from foward (adj R^2)
regs_fwd <- str_c(best_adjr_predictors, collapse = " + ")

fwd_glue <- glue("abnormal_viewership_wk1 ~ {regs_fwd}")

fwd_mod <- lm(as.formula(fwd_glue), data = new_weather_film_wk1)
```

17.c

```
#only show the last steps (trace = 0)
backward <- step(ols_mod, direction = "backward",trace=0)
#name of the coefficients (without the intercept)
best_bkwd_predictors <- names(coefficients(backward)[-1])

best_bkwd_predictors
```

```
## [1] "abnormal_www_rain"    "abnormal_www_mat5_45" "abnormal_www_mat5_55"
## [4] "abnormal_www_mat5_70" "abnormal_www_mat5_75" "abnormal_www_prec_0"
## [7] "abnormal_www_cloud_3" "abnormal_www_cloud_4" "abnormal_www_mat_la"
```

```
#running regressions based on the model from backward
regs_bkwd <- str_c(best_bkwd_predictors, collapse = " + ")

bkwd_glue <- glue("abnormal_viewership_wk1 ~ {regs_bkwd}")

bkwd_mod <- lm(as.formula(bkwd_glue), data = new_weather_film_wk1)
```

17.d *i*

```
#making a matrix of the predictor variables
abnormal_vars <- as.matrix(new_weather_film_wk1 |>
                select(all_of(abnormal_weather_wk1_names)))
#making the ridge model
ridge_mod <- cv.glmnet(
  x = abnormal_vars,
  y = new_weather_film_wk1 |>
    pull(abnormal_viewership_wk1), #pull gets the numeric values
  alpha = 0, # Ridge penalty
  nfolds = 5 # 5 fold cross validation
)

#lambda that minimizes the cross validation error
nfold_ridge <- ridge_mod$lambda.min

#get the coefficients (no intercept)
no_inter_nfold_ridge <- coef(ridge_mod, s = nfold_ridge)[-1, ]

#coefs that aren't 0
coefs_nfold_ridge <- no_inter_nfold_ridge[no_inter_nfold_ridge != 0]
```

ii

```
# Find lambda within 1 standard deviation of the minimum lambda (storind estimates for now. Will predic
nfold_ridge_1se <- ridge_mod$lambda.1se

#get the coefficients (no intercept)
no_inter_nfold_ridge_1se <- coef(ridge_mod, s = nfold_ridge_1se)[-1, ]

#coefs that aren't 0
coefs_nfold_ridge_1se <- no_inter_nfold_ridge_1se[no_inter_nfold_ridge_1se != 0]
```

17.e *i*

```
lasso_mod <- cv.glmnet(
  x = abnormal_vars,
  y = new_weather_film_wk1 |>
    pull(abnormal_viewership_wk1), #pull gets the numeric values
  alpha = 1, # Lasso penalty
  nfolds = 5 # 5 fold cross validation
)

#lambda that minimizes the cross validation error
nfold_lasso = lasso_mod$lambda.min

#get the coefficients (no intercept)
no_inter_nfold_lasso <- coef(lasso_mod, s = nfold_lasso)[-1, ]

#coefs that aren't 0
coefs_nfold_lasso <- no_inter_nfold_lasso[no_inter_nfold_lasso != 0]
```

ii

```
# Find lambda within 1 standard deviation of the minimum lambda
nfold_lasso_1se <- lasso_mod$lambda.1se

#get the coefficients (no intercept)
no_inter_nfold_lasso_1se <- coef(lasso_mod, s = nfold_lasso_1se)[-1, ]

#coefs that aren't 0
coefs_nfold_lasso_1se <- no_inter_nfold_lasso_1se[no_inter_nfold_lasso_1se != 0]
```

iii Note: Could not find lambdas with 1 or 2, regressors. Finding 3, 4, 5 regressors for the remaining questions...

```
#making a function to find minimum lambda that has a certain number of regressors
min_lamba_regressors <- function(num_of_regressors) {
  # Create an empty list to store lambda
  lambda_list<- list()
  for (lasso_value in lasso_mod$lambda){
    #get the coefficients (no intercept)
    no_intercept_coefs <- coef(lasso_mod, s = lasso_value)[-1, ]
    #coefs that aren't 0
    coefs_lasso_value <- no_intercept_coefs[no_intercept_coefs != 0]
```

```
    #if the number of coeffiecients in a lambda value is ---, append
    if (length(coefs_lasso_value) == num_of_regressors) {
      lambda_list <- c(lambda_list, lasso_value)
    }
  }
  #find the minimum in the appended list
  minimum_lasso = min(unlist(lambda_list))
  return(minimum_lasso)
}

lasso_3regs <- min_lamba_regressors(3)

#get the coefficients (no intercept)
no_inter_lasso_3regs <- coef(lasso_mod, s = lasso_3regs)[-1, ]

#coefs that aren't 0
coefs_lasso_3regs <- no_inter_lasso_3regs[no_inter_lasso_3regs != 0]

lasso_3regs
```

```
## [1] 0.09405896
```

iv

```
lasso_4regs <- min_lamba_regressors(4)

#get the coefficients (no intercept)
no_inter_lasso_4regs <- coef(lasso_mod, s = lasso_4regs)[-1, ]

#coefs that aren't 0
coefs_lasso_4regs <- no_inter_lasso_3regs[no_inter_lasso_4regs != 0]
lasso_4regs
```

```
## [1] 0.07115216
```

v

```
lasso_5regs <- min_lamba_regressors(5)
#get the coefficients (no intercept)
no_inter_lasso_5regs <- coef(lasso_mod, s = lasso_5regs)[-1, ]

#coefs that aren't 0
coefs_lasso_5regs <- no_inter_lasso_3regs[no_inter_lasso_5regs != 0]
lasso_5regs
```

```
## [1] 0.05907177
```

17.f

```r
#making the fitted models
weather_film_wk1 <- new_weather_film_wk1 |>
  #add the new predicted columns
  mutate(
    #ols
    pred_ols = predict(ols_mod, new_weather_film_wk1),
    #prediction using coefficients obtained from the forward subset selection
    pred_fwd = predict(fwd_mod, new_weather_film_wk1),
    #prediction using coefficients obtained from the backward subset selection
    pred_bkwd = predict(bkwd_mod, new_weather_film_wk1),
    #the following model predictions are saved under [, "s1"]-----------
      #abnormal coefs is the predictor variables
    pred_ridge_5fold = predict(ridge_mod,
                               newx = abnormal_vars,
                               s = nfold_ridge)[, "s1"],
    #ridge 5 fold, 1se away
    pred_ridge_5fold_1se = predict(ridge_mod,
                               newx = abnormal_vars,
                               s = nfold_ridge_1se)[, "s1"],
    #lasso, 5 fold
    pred_lasso_5fold = predict(lasso_mod,
                               newx = abnormal_vars,
                               s = nfold_lasso)[, "s1"],
    #lasso 5 fold, 1se away
    pred_lasso_5fold_1se = predict(lasso_mod,
                               newx = abnormal_vars,
                               s = nfold_lasso_1se)[, "s1"],
    #lasso, min lambda with 3 regressors
    pred_lasso_3regs = predict(lasso_mod,
                               newx = abnormal_vars,
                               s = lasso_3regs)[, "s1"],
    #lasso, min lambda with 4 regressors
    pred_lasso_4regs = predict(lasso_mod,
                               newx = abnormal_vars,
                               s = lasso_4regs)[, "s1"],
    #lasso, min lambda with 5 regressors
    pred_lasso_5regs = predict(lasso_mod,
                               newx = abnormal_vars,
                               s = lasso_5regs)[, "s1"]
  )

#making a subset of just predictor values,
#...by selecting the last 10 columns
pred_mod_columns <- weather_film_wk1[tail(names(weather_film_wk1), 10)]

#making a correlation matrix
corrs <- round(cor(pred_mod_columns), 2)

# Print the correlation matrix as a LaTeX table
kable(corrs, format = "latex", booktabs = TRUE) |>
#scale (because of width) and keep the table at the position
  kable_styling(latex_options= c("scale_down","HOLD_position"))
```

	pred_ols	pred_fwd	pred_bkwd	pred_ridge_5fold	pred_ridge_5fold_1se	pred_lasso_5fold	pred_lasso_5fold_1se	pred_lasso_3regs	pred_lasso_4regs	pred_lasso_5regs
pred_ols	1.00	0.85	0.93	0.92	NA	0.90	NA	0.71	0.75	0.75
pred_fwd	0.85	1.00	0.86	0.90	NA	0.90	NA	0.77	0.80	0.80
pred_bkwd	0.93	0.86	1.00	0.93	NA	0.92	NA	0.74	0.78	0.78
pred_ridge_5fold	0.92	0.90	0.93	1.00	NA	0.98	NA	0.82	0.88	0.89
pred_ridge_5fold_1se	NA	NA	NA	NA	1	NA	NA	NA	NA	NA
pred_lasso_5fold	0.90	0.90	0.92	0.98	NA	1.00	NA	0.86	0.91	0.91
pred_lasso_5fold_1se	NA	NA	NA	NA	NA	NA	1	NA	NA	NA
pred_lasso_3regs	0.71	0.77	0.74	0.82	NA	0.86	NA	1.00	0.97	0.96
pred_lasso_4regs	0.75	0.80	0.78	0.88	NA	0.91	NA	0.97	1.00	1.00
pred_lasso_5regs	0.75	0.80	0.78	0.89	NA	0.91	NA	0.96	1.00	1.00

As we can see, they are very correlated with each other, except the with the model that returned zero regressors that has all NAs

17.g

```
cat(
  " OLS:", length(abnormal_weather_wk1_names), "\n",
  "Forward:", length(best_adjr_predictors), "\n",
  "Backward:", length(best_bkwd_predictors), "\n",
  "Ridge (5fold):", length(coefs_nfold_ridge), "\n",
  "Ridge (5fold, 1se):", length(coefs_nfold_ridge_1se), "\n",
  "Lasso (5fold):", length(coefs_nfold_lasso), "\n",
  "Lasso (5fold, 1se):", length(coefs_nfold_lasso_1se), "\n",
  "Lasso (3 regressors):", length(coefs_lasso_3regs), "\n",
  "Lasso (4 regressors):", length(coefs_lasso_4regs), "\n",
  "Lasso (5 regressors):", length(coefs_lasso_5regs)
  )
```

```
##  OLS: 37
##  Forward: 9
##  Backward: 9
##  Ridge (5fold): 37
##  Ridge (5fold, 1se): 37
##  Lasso (5fold): 16
##  Lasso (5fold, 1se): 0
##  Lasso (3 regressors): 3
##  Lasso (4 regressors): 4
##  Lasso (5 regressors): 5
```

17.h NOT NEEDED

17.i This is a low-dimensional dataset because the number of predictors is smaller than the number of observations.

18

```
wk2_abnormal_data <- new_weather_film_wk2 |>
  select(title, abnormal_viewership_wk_2) |>
  arrange(title) |>
  #creating a row id to join nicely...
  mutate(row_id = row_number())

wk1_abnormal_data <-weather_film_wk1 |>
  #remove unwanted pred variables
```

```r
  select(-c("pred_tickets", "pred_tickets_wk_1")) |>
  select(title, abnormal_viewership_wk1, contains("pred")) |>
  arrange(title) |>
  mutate(row_id = row_number())

wk1_2 <- left_join(wk1_abnormal_data, wk2_abnormal_data, by = c("title", "row_id"))
names_pred <- c(
    "OLS",
    "Forward",
    "Backward",
    "Ridge (5fold)",
    "Ridge (5fold, 1se)",
    "Lasso (5fold)",
    "Lasso (5fold, 1se)",
    "Lasso (3 regressors)",
    "Lasso (4 regressors)",
    "Lasso (5 regressors)"
)

fitted_columns <- str_subset(colnames(wk1_2), "pred")

# Create a list to store results
results <- list()

# Loop through the fitted columns and run regressions
for (fit_name in fitted_columns) {
  # Run the regression
  formula <- as.formula(paste("abnormal_viewership_wk_2 ~", fit_name))
  lm_model <- lm(formula, data = wk1_2)

  # Extract coefficient
  coefficient <- coef(lm_model)[2]  # Assuming the predictor variable is the second column

  # Store results
  results[[fit_name]] <- coefficient
}

#model no instruments
lm_model_no_inst <- lm(abnormal_viewership_wk_2 ~ abnormal_viewership_wk1, data = wk1_2)

  # Extract coefficient
coefficient_no_inst <- coef(lm_model_no_inst)[2]

names(results) <- names_pred

# Print results
for (fit_name in names(results)) {
  #Name, coefficient
  cat(fit_name, ":", results[[fit_name]], "\n")
}
```

```
## OLS : 0.3422299
## Forward : 0.2673496
```

```
## Backward : 0.3269672
## Ridge (5fold) : 0.4112934
## Ridge (5fold, 1se) : NA
## Lasso (5fold) : 0.3996168
## Lasso (5fold, 1se) : NA
## Lasso (3 regressors) : 3.559318
## Lasso (4 regressors) : 0.9532715
## Lasso (5 regressors) : 0.6779283
```

```
cat("No Instrument:", coefficient_no_inst)
```

```
## No Instrument: 0.418857
```

19

In the context of the paper, these results show how the estimate of opening viewership in subsequent weeks varies by method. These coefficients alone aren't sufficient to distinguish social learning from social utility. The authors went on a test for social learning by asking whether the momentum generated from an exogenous shock to opening weekend viewership is stronger for higher-quality movies than for lower-quality movies.

20

The absence of Lord of the Rings: The Return of the King, Lord of the Rings: The Two Towers, Toy Story 3, X2: X-Men United, and Finding Nemo from the upper decile made Kuehn and Lampe suspicious.

Kuehn and Lampe collect underlying user ratings data and reconstruct the quality categories for movies, addressing anomalies they found in the categorization of movies into high and low-quality categories by Gilchrist and Sands. While maintaining the use of Gilchrist and Sands' empirical framework, Kuehn and Lampe tested additional implications of social learning models, such as the impact of sequels versus original movies on sales momentum and the effect of audience age demographics (e.g., movies with a larger share of teenage viewers) on momentum.

Kuehn and Lampe find strong evidence of social learning, contradicting Gilchrist and Sands' findings which suggested a limited role for learning and emphasized network (or demand) externalities. Kuehn and Lampe's results indicate that movies in the top quality tercile exhibit significant momentum, which is not observed for movies in the bottom tercile. This pattern is consistent with social learning but does not support the presence of network externalities. Kuehn and Lampe's tests on the flexibility of prior beliefs (comparing sequels to originals) and the size of social networks (comparing movies with larger shares of teenage viewers to those with smaller shares) further support the presence of social learning. They find momentum to be significant for originals and movies with larger shares of teenage viewers, especially within the PG-13 category, reinforcing the idea that social learning mechanisms are at play.

22

The paper is an economics paper because it examines the impact of social interactions on consumer behavior, specifically how early viewership influences later movie attendance through social spillovers. This topic falls within the realm of economics as it deals with understanding how individuals make decisions based on information and experiences shared within their social networks, and how these decisions aggregate to broader market outcomes. This is of fundamental interest to the field of economics.

Problem 2

1

Given that $Y \mid X, \beta \sim \mathcal{N}\left(X\beta, \sigma^2 I_n\right)$, the density of Y conditional on X and β is given by the probability density function of the multivariate normal distribution. The pdf for a multivariate normal distribution with mean μ and covariance matrix Σ is given by:

$$f(y; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(y-\mu)^\top \Sigma^{-1}(y-\mu)\right)$$

For our specific case, $\mu = X\beta$ and $\Sigma = \sigma^2 I_n$, thus the conditional density of Y given X and β is:

$$f(Y \mid X, \beta) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2}(Y-X\beta)^\top (Y-X\beta)\right)$$

This expression gives the likelihood of observing Y given the predictors X and the regression coefficients β, under the assumption of normally distributed errors with variance σ^2.

2

Given that $\beta \sim \mathcal{N}\left(0, \tau^2 I_p\right)$, the density of β is given by the probability density function of the multivariate normal distribution, where $\mu = 0$ (a p-dimensional zero vector) and $\Sigma = \tau^2 I_p$ (a $p \times p$ covariance matrix where the variance of each component of β is τ^2 and all covariances are 0). The pdf for a multivariate normal distribution with mean μ and covariance matrix Σ is:

$$f(x; \mu, \Sigma) = \frac{1}{(2\pi)^{k/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right)$$

In our case, $k = p$, $\mu = 0$, and $\Sigma = \tau^2 I_p$, thus the density of β is:

$$f(\beta) = \frac{1}{(2\pi\tau^2)^{p/2}} \exp\left(-\frac{1}{2\tau^2}\beta^\top \beta\right)$$

This expression gives the prior belief about the distribution of the regression coefficients β, assuming they are drawn from a multivariate normal distribution centered at zero with covariance $\tau^2 I_p$, indicating a belief in the shrinkage towards zero of the coefficients (consistent with the principles of Ridge Regression).

3

To show that the posterior likelihood of β given Y and X satisfies the given expression, we'll use the densities for $f(Y \mid X, \beta)$ and $f(\beta)$ that we have previously derived and apply Bayes' theorem.

Given:

1. The conditional density of Y given X and β:

$$f(Y \mid X, \beta) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2}(Y-X\beta)^\top (Y-X\beta)\right)$$

2. The prior density of β:

$$f(\beta) = \frac{1}{(2\pi\tau^2)^{p/2}} \exp\left(-\frac{1}{2\tau^2}\beta^\top\beta\right)$$

3. Bayes' theorem for the posterior likelihood:

$$f(\beta \mid Y, X) = \frac{f(Y \mid X, \beta) f(\beta)}{f(Y)}$$

where $f(Y)$ does not depend on β and serves as a normalizing constant.

Substituting the expressions for $f(Y \mid X, \beta)$ and $f(\beta)$ into Bayes' theorem:

$$f(\beta \mid Y, X) = \frac{\frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2}(Y - X\beta)^\top(Y - X\beta)\right) \times \frac{1}{(2\pi\tau^2)^{p/2}} \exp\left(-\frac{1}{2\tau^2}\beta^\top\beta\right)}{f(Y)}$$

Since $f(Y)$ does not depend on β, and noting that the constants involving $(2\pi\sigma^2)^{n/2}$ and $(2\pi\tau^2)^{p/2}$ do not depend on β, we can denote all the terms that do not depend on β by a constant C'. Thus, the expression simplifies to:

$$f(\beta \mid Y, X) = C' \exp\left(-\frac{1}{2\sigma^2}(Y - X\beta)^\top(Y - X\beta) - \frac{1}{2\tau^2}\beta^\top\beta\right)$$

Here, C' is a constant that may involve factors from the normalizing constant $f(Y)$ and the constants pulled out of the exponential terms but crucially does not depend on β.

Given that C is defined as a constant that does not depend on β, we can express C' simply as C for clarity. Therefore, we show that:

$$f(\beta \mid Y, X) = C \exp\left(-\frac{1}{2\sigma^2}(Y - X\beta)^\top(Y - X\beta) - \frac{1}{2\tau^2}\beta^\top\beta\right)$$

4

To conclude that the β which maximizes the posterior likelihood (the mode of the posterior distribution) is given by the solution to the specified optimization problem, we start with the expression for the posterior likelihood derived in the previous question:

$$f(\beta \mid Y, X) = C \exp\left(-\frac{1}{2\sigma^2}(Y - X\beta)^t(Y - X\beta) - \frac{1}{2\tau^2}\beta^t\beta\right)$$

Maximizing the posterior likelihood $f(\beta \mid Y, X)$ with respect to β is equivalent to minimizing the exponent's argument because the exponential function is monotonically increasing. Thus, we focus on minimizing the negative exponent in the expression, which involves two terms:

1. The term $-\frac{1}{2\sigma^2}(Y - X\beta)^t(Y - X\beta)$ corresponds to the squared error between the observed and predicted values, scaled by the variance σ^2.

2. The term $-\frac{1}{2\tau^2}\beta^t\beta$ corresponds to the penalty on the size of β, scaled by the variance τ^2.

Dropping the constants and the negative sign, the problem of maximizing the posterior likelihood can be restated as minimizing the expression:

$$\frac{1}{2\sigma^2}(Y - X\beta)^t(Y - X\beta) + \frac{1}{2\tau^2}\beta^t\beta$$

Multiplying through by $2\sigma^2$ (which does not change the location of the minimum) to simplify, we get:

$$(Y - X\beta)^t(Y - X\beta) + \frac{\sigma^2}{\tau^2}\beta^t\beta$$

Recognizing that $\beta^t\beta = \|\beta\|_2^2$, the optimization problem becomes:

$$\min_\beta (Y - X\beta)^t(Y - X\beta) + \frac{\sigma^2}{\tau^2}\|\beta\|_2^2$$

This is precisely the ridge regression optimization problem, where $(Y - X\beta)^t(Y - X\beta)$ represents the least squares error term, and $\frac{\sigma^2}{\tau^2}\|\beta\|_2^2$ is the ridge penalty on the size of β, with the penalty strength controlled by the ratio $\frac{\sigma^2}{\tau^2}$. Thus, the β that maximizes the posterior likelihood under the Bayesian interpretation is indeed given by the solution to this optimization problem, highlighting the connection between Bayesian inference and ridge regression.

5

In Bayesian inference, we begin with a prior belief about the parameters (in this case, β) of our linear model. This belief is quantified by assuming that β follows a normal distribution centered around zero, with its dispersion controlled by τ^2. This assumption reflects our expectation that the true values of the regression coefficients are likely to be small or close to zero, enforcing a form of regularization even before observing the data.

When we observe the data (represented by Y and X), we update our beliefs about β by calculating the posterior distribution, which combines our prior belief and the likelihood of observing the data given β. The likelihood of the data given β is also modeled using a normal distribution, where the variance σ^2 captures the uncertainty or noise in the observations.

Maximizing the posterior likelihood of β, or equivalently, finding the mode of the posterior distribution, involves adjusting our estimate of β to best fit the observed data while also adhering to our initial regularization imposed by the prior. This balance is captured mathematically by an optimization problem that seeks to minimize the sum of two terms: one representing the fit of the model to the data (the least squares error term) and the other representing the size of the coefficients (the ridge penalty term), weighted by the ratio of the variance of the observation noise to the variance of the prior on β ($\frac{\sigma^2}{\tau^2}$).

This optimization problem is precisely that of ridge regression, which seeks to find regression coefficients that not only fit the data well but are also small in magnitude, to prevent overfitting and improve the generalizability of the model. The strength of the regularization is controlled by the ratio $\frac{\sigma^2}{\tau^2}$, linking the Bayesian interpretation directly to the regularization parameter in ridge regression. This parameter balance reflects the trade-off between fitting the data closely (minimizing prediction error) and keeping the model simple and robust to new data (minimizing the size of the coefficients).