Problem Set 7

Matt Krasnow!!!!! (the goat)

Due Friday, November 15, 2024 at 11:59pm

Problem set policies. *Please provide concise, clear answers for each question while making sure to fully explain your reasoning. For problems asking for calculations, show your work in addition to clearly indicating the final answer. For problems involving* `R`*, be sure to include the code and output in your solution.*

Please submit the PDF of your knit solutions to Gradescope and be sure to assign which pages of your solution correspond to each problem. Make sure that the PDF is fully readable to the graders; e.g., make sure that lines don't run off the page margin.

We encourage you to discuss problems with other students (and, of course, with the teaching team), but you must write your final answer in your own words. Solutions prepared "in committee" are not acceptable. If you do collaborate with classmates on a problem, please list your collaborators on your solution. Be aware that simply copying answers found online, whether human-generated or machine-generated, is a violation of the Honor Code.

Question 1

There are two datasets included that will be used for this problem:

- a training dataset called `bosflights18.csv` which includes data on a subset of flights ($n = 10,000$) in and out of Boston's Logan Airport in the year 2018.
- a test dataset called `bosflights18_test.csv` (note it is actually larger in size than `bosflights18.csv`)

The important variables in these datasets are:

`flight_time`: the total amount of time the flight takes from the time the plane takes off until the time it arrives at the destination gate.

`year`: year of flight (they are all from 2018)

`month`: month: 1 = January, 2 = February, etc.

`dayofmonth`: the calendar day of the month, from 1 to 31.

`weekday`: day of the week: 1 = Monday, 2 = Tuesday, etc.

`carrier`: the unique 2-digit carrier code of the flight. For details, see the list here: https://en.wikipedia.org/wiki/List_of_airlines_of_the_United_States

`tailnum`: the unique tail number of the aircraft

`flightnum`: the carrier's specific flight number

`origin`: the originating airport. See http://www.leonardsguide.com/us-airport-codes.shtml.

`dest`: the destination airport.

`bos_depart`: an indicator if the flight departed out of Boston.

`schedule_depart`: the scheduled departure time in minutes across the day ranging from 0 to 1439. 7pm is 1140, for example.

`depart`: the actual departure time (in minutes)

`wheels_off`: the time of day the plane took off (in minutes)

`distance`: the distance of the flight, in miles.

`weather_delay`: an indicator if the delay is due to extreme weather.

`nas_delay`: an indicator if the delay is due to the national aviation system (air traffic control, for example).

`security_delay`: an indicator if the delay is due to a security issue at the terminal.

`late_aircraft_delay`: an indicator if the delay is due to a late arrival of a previous flight with the same aircraft.

`carrier_delay`: an indicator if the delay is due to a carrier (kind of a catch all if not the others).

More info on the delay indicators can be found at the Bureau of Transportation Statistics (BTS).

We want to predict `flight_time` (untransformed) based on all of the other predictors in the data set (all other variables could be measured at some point before the flight takes off).

(a) Fit the following three linear models and for each report (1) R^2 on the training data, (2) the number of non-`NA` β estimates in the training model, and (3) the number of `NA` β estimates in the training model:

- **lm1** that predicts flight time from the main effects of all of the included predictors (untransformed quantitative predictors, but be sure to handle categorical predictors appropriately).
- **lm2** that predicts flight time from the main effects of all of the included predictors but treats `distance` with a 15^{th} order polynomial function (in this case, do NOT use the `raw=T` argument in `poly`).
- **lm3** that predicts flight time from the main effects (treating `distance` with a 15^{th} order polynomial function) and the interactions of `bos_depart` with all the other predictors (including all polynomial terms of `distance`) [ignore other interactions].

Note: you should completely ignore 4 variables here: `year`, `day_of_month`, `flightnum`, and `tailnum`.

```
library(tidyverse)


flights_train <- read.csv("data/bosflights18.csv")

# Remove excluded variables and prepare the data
flights_clean <- flights_train %>%
  select(-year, -dayofmonth, -flightnum, -tailnum) %>%
  mutate(
    month = as.factor(month),
    weekday = as.factor(weekday),
    carrier = as.factor(carrier),
    origin = as.factor(origin),
    dest = as.factor(dest),
    bos_depart = as.factor(bos_depart),
    weather_delay = as.factor(weather_delay),
    nas_delay = as.factor(nas_delay),
    security_delay = as.factor(security_delay),
    late_aircraft_delay = as.factor(late_aircraft_delay),
    carrier_delay = as.factor(carrier_delay)
  )
```

```
# one

lm1 <- lm(flight_time ~ ., data = flights_clean)

# Extract results for lm1
r2_lm1 <- summary(lm1)$r.squared
coef_lm1 <- summary(lm1)$coefficients
na_count_lm1 <- sum(is.na(coef_lm1[,1]))
nonna_count_lm1 <- sum(!is.na(coef_lm1[,1]))
```

```
# two

lm2 <- lm(flight_time ~ . - distance + poly(distance, 15),
          data = flights_clean)


r2_lm2 <- summary(lm2)$r.squared
coef_lm2 <- summary(lm2)$coefficients
na_count_lm2 <- sum(is.na(coef_lm2[,1]))
nonna_count_lm2 <- sum(!is.na(coef_lm2[,1]))
```

```r
# three

poly_dist <- poly(flights_clean$distance, 15)
# Create the formula with interactions
lm3 <- lm(flight_time ~ (. - distance + poly(distance, 15)) * bos_depart,
          data = flights_clean)

r2_lm3 <- summary(lm3)$r.squared
coef_lm3 <- summary(lm3)$coefficients
na_count_lm3 <- sum(is.na(coef_lm3[,1]))
nonna_count_lm3 <- sum(!is.na(coef_lm3[,1]))
```

```r
results <- data.frame(
  Model = c("lm1", "lm2", "lm3"),
  R_squared = c(r2_lm1, r2_lm2, r2_lm3),
  Non_NA_betas = c(nonna_count_lm1, nonna_count_lm2, nonna_count_lm3),
  NA_betas = c(na_count_lm1, na_count_lm2, na_count_lm3)
)


print(results)
```

```
##   Model R_squared Non_NA_betas NA_betas
## 1   lm1 0.9511597           79        0
## 2   lm2 0.9560594           93        0
## 3   lm3 0.9606136          146        0
```

INTERPRETATOIN:

- all are good fit to the training data with high r^2 values
- progressive improbvement in r^2 (but small)

Stability - all of them have no NA coefficients indicating stable estimation

Implications - they are all good fits, with lm3 being the best because the r^2 is highest (tentatively)

(b) Why are there `NA` estimates (be specific to this dataset)?

There are not any NA estimates in these models! Except if year day of the month flight num and tail num are – these should not have any relation to the response variable so we excluded them. the rest of the variables are non NA.

(c) Evaluate the three models in part (a) based on RMSE on both the train and test sets. Interpret the results as to which model is best for prediction and which models may be overfit.

```r
rmse <- function(actual, predicted) {
  sqrt(mean((actual - predicted)^2, na.rm = TRUE))
}


train_rmse_lm1 <- rmse(flights_clean$flight_time, predict(lm1))
train_rmse_lm2 <- rmse(flights_clean$flight_time, predict(lm2))
train_rmse_lm3 <- rmse(flights_clean$flight_time, predict(lm3))


flights_test <- read.csv("data/bosflights18_test.csv")
```

```r
flights_test_clean <- flights_test %>%
  select(-year, -dayofmonth, -flightnum, -tailnum) %>%
  mutate(
    month = as.factor(month),
    weekday = as.factor(weekday),
    carrier = as.factor(carrier),
    origin = as.factor(origin),
    dest = as.factor(dest),
    bos_depart = as.factor(bos_depart),
    weather_delay = as.factor(weather_delay),
    nas_delay = as.factor(nas_delay),
    security_delay = as.factor(security_delay),
    late_aircraft_delay = as.factor(late_aircraft_delay),
    carrier_delay = as.factor(carrier_delay)
  )


test_rmse_lm1 <- rmse(flights_test_clean$flight_time, predict(lm1, newdata = flights_test_clean))
test_rmse_lm2 <- rmse(flights_test_clean$flight_time, predict(lm2, newdata = flights_test_clean))
```

```
## Warning in predict.lm(lm2, newdata = flights_test_clean): prediction from
## rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```r
test_rmse_lm3 <- rmse(flights_test_clean$flight_time, predict(lm3, newdata = flights_test_clean))
```

```
## Warning in predict.lm(lm3, newdata = flights_test_clean): prediction from
## rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```r
results <- data.frame(
  Model = c("lm1", "lm2", "lm3"),
  Train_RMSE = c(train_rmse_lm1, train_rmse_lm2, train_rmse_lm3),
  Test_RMSE = c(test_rmse_lm1, test_rmse_lm2, test_rmse_lm3)
)

print(results)
```

```
##   Model Train_RMSE Test_RMSE
## 1   lm1   12.14254  12.07446
## 2   lm2   11.51737  11.56716
## 3   lm3   10.90419  10.98704
```

INTERPRETATION - all are very similar in respect to RMSE - slighht progressive improvement in RMSE (decreaasing test error) -

overfitting analysis - very slight increase between train and test RMSE in LM2 and LM3 - this indicates that there is probably limited overfitting, but expectdely as model complexity increases, overfitting may occur

- lm3 is the best probably because it has the lowest RMSE without evidence of signficant overfitting in comparison to the other models. however, it is more complex which may not be favorable

Question 2

(a) Fit well-tuned Ridge and LASSO regression models using `cv.glmnet` based on the predictors used in the **lm3** model from the previous problem. Hint: the R command `model.matrix` may be helpful to get you started.

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.1-8
```

```
# First, create the formula string that matches lm3
formula_lm3 <- as.formula("flight_time ~ (. - distance + poly(distance, 15)) * bos_depart")


X <- model.matrix(formula_lm3, data = flights_clean)[,-1]
y <- flights_clean$flight_time

set.seed(139)


ridge_cv <- cv.glmnet(X, y, alpha = 0)


lasso_cv <- cv.glmnet(X, y, alpha = 1)

# optimal lambda values
ridge_lambda_min <- ridge_cv$lambda.min
ridge_lambda_1se <- ridge_cv$lambda.1se
lasso_lambda_min <- lasso_cv$lambda.min
lasso_lambda_1se <- lasso_cv$lambda.1se


cat("Ridge Regression:\n")
```

```
## Ridge Regression:
```

```
cat("Optimal lambda (minimum MSE):", ridge_lambda_min, "\n")
```

```
## Optimal lambda (minimum MSE): 5.261206
```

```
cat("Optimal lambda (1se rule):", ridge_lambda_1se, "\n\n")
```

```
## Optimal lambda (1se rule): 5.261206
```

```
cat("LASSO Regression:\n")
```

```
## LASSO Regression:
```

```
cat("Optimal lambda (minimum MSE):", lasso_lambda_min, "\n")
```

```
## Optimal lambda (minimum MSE): 0.01606696
```

```r
cat("Optimal lambda (1se rule):", lasso_lambda_1se, "\n")
```

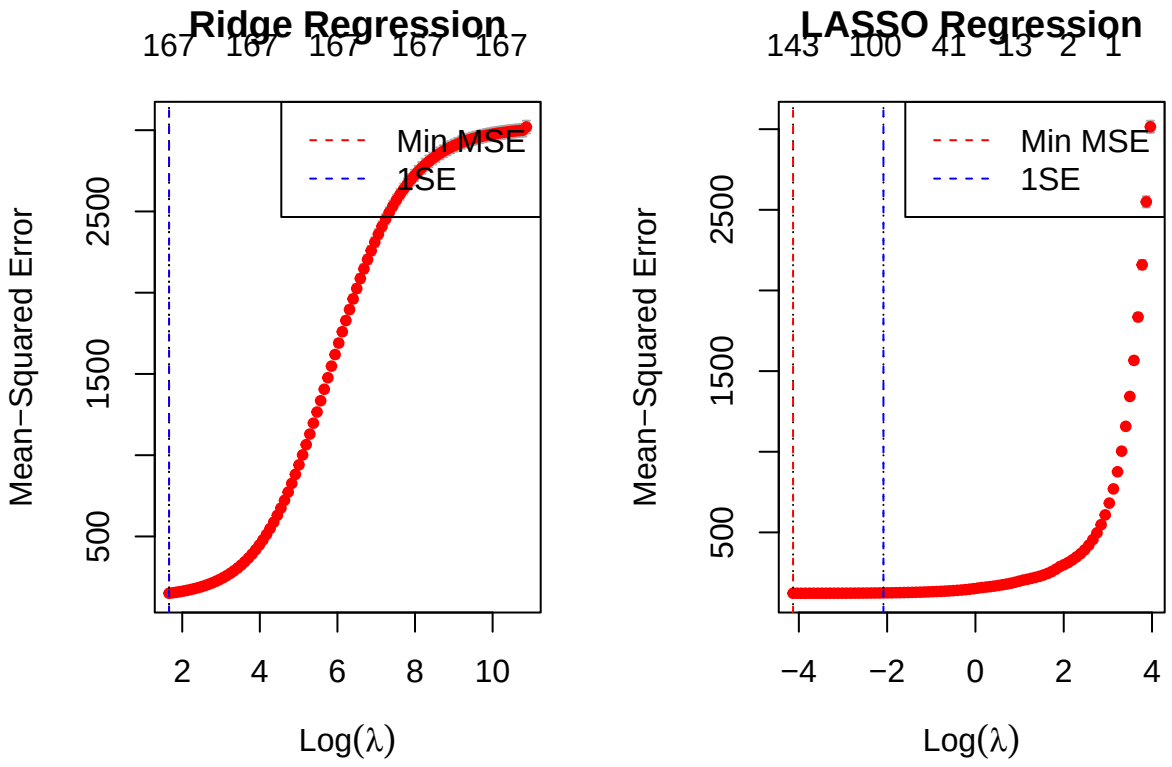
```
## Optimal lambda (1se rule): 0.1244007
```

(b) For both the Ridge and LASSO models, plot the average MSE on the validation sets against the λ's you considered in the previous part. Report the best λ's. (This part should require almost no work if you did part (a)).

```r
par(mfrow=c(1,2))

plot(ridge_cv, main="Ridge Regression")
abline(v=log(ridge_lambda_min), col="red", lty=2)
abline(v=log(ridge_lambda_1se), col="blue", lty=2)
legend("topright",
       legend=c("Min MSE", "1SE"),
       col=c("red", "blue"),
       lty=2)

plot(lasso_cv, main="LASSO Regression")
abline(v=log(lasso_lambda_min), col="red", lty=2)
abline(v=log(lasso_lambda_1se), col="blue", lty=2)
legend("topright",
       legend=c("Min MSE", "1SE"),
       col=c("red", "blue"),
       lty=2)
```




```r
par(mfrow=c(1,1))
```

Ridge:

lambda_min (minimum MSE) = 5.261206 lambda_1se (one standard error rule) = 5.261206

LASSO:

lambda_min (minimum MSE) = 0.01606696 lambda_1se (one standard error rule) = 0.07118664

from A

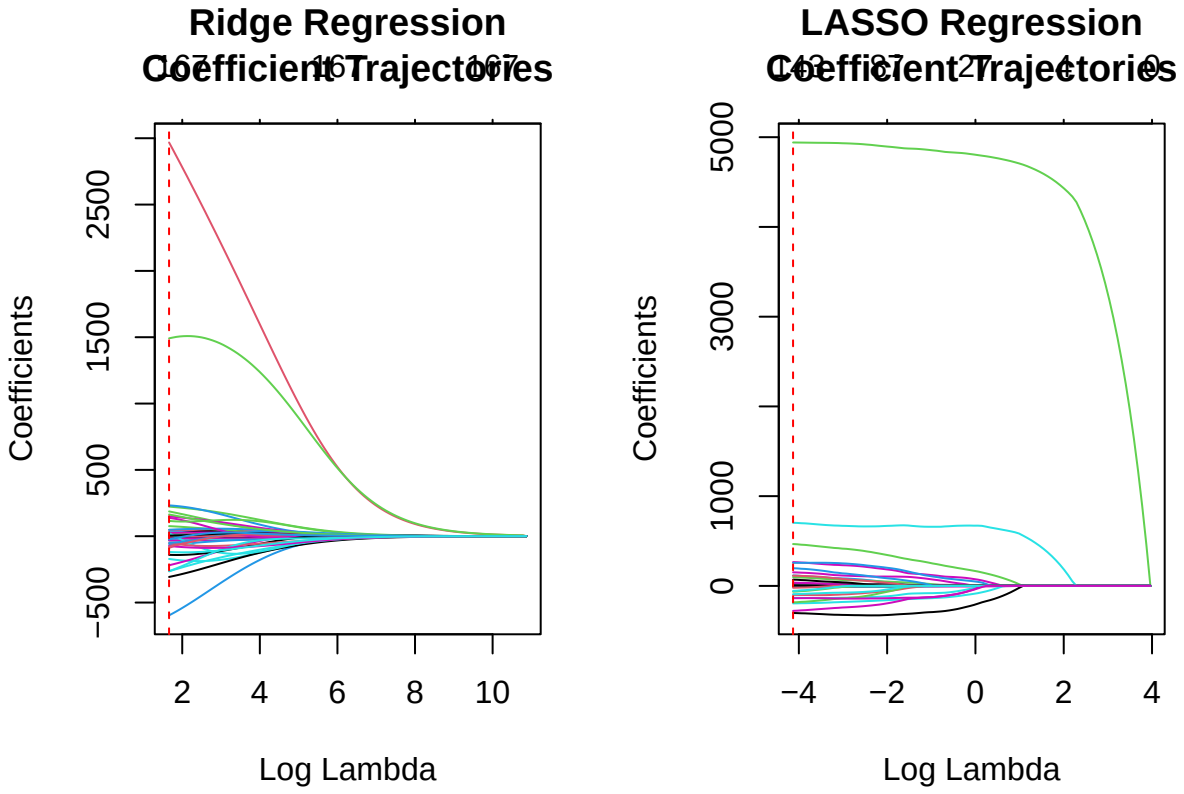
(c) Provide the "$\widehat{\beta}$ trajectory" plots of the main effects from these models (plot each β_j as a function of λ as a curve, and do this for all main/linear effects). Interpret what you see in 2-3 sentences.

```
# (without cross-validation this time)
ridge_fit <- glmnet(X, y, alpha = 0)
lasso_fit <- glmnet(X, y, alpha = 1)

par(mfrow=c(1,2))

plot(ridge_fit, xvar="lambda", main="Ridge Regression\nCoefficient Trajectories",
     label=TRUE)
abline(v=log(ridge_lambda_min), col="red", lty=2)

plot(lasso_fit, xvar="lambda", main="LASSO Regression\nCoefficient Trajectories",
     label=TRUE)
abline(v=log(lasso_lambda_min), col="red", lty=2)
```




```
par(mfrow=c(1,1))
```

In the Ridge, we see that coefficients are shrunk gradually towards zero as lambda increases, but never reach exactly zero. In contrast, the LASSO plot shows coefficients being shrunk to exactly zero as lambda increase. Some coefficients are eliminated entirely at larger lambda values, indicating these variables are considered less important for prediction by the LASSO model.

More interestingly, there are two coefficients which seem to be the most important in both – further analysis would idenitfy which these are

(d) Choose a best regularized/penalized regression model and briefly justify your choice. Revisit the grid of λ's that were used (either explicitly by you, or automatically by `R`) and comment on whether it's obvious that these penalized models will predict better than the original model.

LASSO IS BETTER - use lambda min or = 0.016 - good variable selection – less important variables go to zero - stable MSE across range of lambda valus - slightly better performance than RIDGE

Its not obvious if using LASSO or RIDGE will be better than the lm3. there isn't much improvement becuase - relatively flat MSE curves around optimal lmabda - lm3 already had good performance - no dramatic shift in the selected lmabda values

Question 3: Work on your EDA

Question 3 allows you to start working on your final project EDA. Thus, if you find any issues with your data, you will be aware early! Evaluate the quality of your data by creating a table which, for each important continuous variable in your dataset reports:

- The number of non-missing observations
- The number of missing observations
- A measure(s) of the central tendency (i.e., mean, media)
- A measure(s) of variability (i.e, sd, IQR)

and for each important categorical variable in your dataset reports:

- The levels of the variable
- For each level:
  - The number of non-missing observations
  - The number of missing observations

```
data <- read.csv("data/NSDUH_data.csv")


summary(data)
```

```
##       AGE3            irsex        IREDUHIGHST2      WRKSTATWK2
##  Min.   : 1.000   Min.   :1.000   Min.   : 1.000   Min.   : 1.00
##  1st Qu.: 4.000   1st Qu.:1.000   1st Qu.: 7.000   1st Qu.: 1.00
##  Median : 7.000   Median :2.000   Median : 9.000   Median : 3.00
##  Mean   : 6.625   Mean   :1.542   Mean   : 8.193   Mean   :16.19
##  3rd Qu.: 9.000   3rd Qu.:2.000   3rd Qu.:11.000   3rd Qu.: 8.00
##  Max.   :11.000   Max.   :2.000   Max.   :11.000   Max.   :99.00
##
##      income         NOMARR2         addprev         yodprev
##  Min.   :1.000   Min.   : 1.00   Min.   : 1.00   Min.   : 1.00
##  1st Qu.:2.000   1st Qu.: 1.00   1st Qu.: 1.00   1st Qu.:99.00
##  Median :3.000   Median :99.00   Median : 2.00   Median :99.00
##  Mean   :2.824   Mean   :56.78   Mean   :23.44   Mean   :79.59
##  3rd Qu.:4.000   3rd Qu.:99.00   3rd Qu.: 2.00   3rd Qu.:99.00
##  Max.   :4.000   Max.   :99.00   Max.   :99.00   Max.   :99.00
##
##     yorelig         adrelig         HEALTH2         snrldcsn
##  Min.   : 1.00   Min.   : 1.00   Min.   :1.000   Min.   : 1.00
##  1st Qu.:99.00   1st Qu.:99.00   1st Qu.:2.000   1st Qu.: 2.00
##  Median :99.00   Median :99.00   Median :2.000   Median : 3.00
##  Mean   :97.05   Mean   :91.83   Mean   :2.305   Mean   :24.51
##  3rd Qu.:99.00   3rd Qu.:99.00   3rd Qu.:3.000   3rd Qu.: 4.00
##  Max.   :99.00   Max.   :99.00   Max.   :4.000   Max.   :99.00
##                                  NA's   :13
##     hltinmnt        snrlgsvc         cigtry         COUTYP4
##  Min.   : 1.00   Min.   : 1.00   Min.   :  1.0   Min.   :1.000
##  1st Qu.: 1.00   1st Qu.: 1.00   1st Qu.: 17.0   1st Qu.:1.000
##  Median : 2.00   Median : 2.00   Median :991.0   Median :2.000
##  Mean   :48.97   Mean   :23.76   Mean   :582.7   Mean   :1.686
##  3rd Qu.:99.00   3rd Qu.: 6.00   3rd Qu.:991.0   3rd Qu.:2.000
##  Max.   :99.00   Max.   :99.00   Max.   :997.0   Max.   :3.000
```

```
##
##     NEWRACE2        service       LVLDIFCARE2       WRKDHRSWK2
##  Min.   :1.000   Min.   : 1.00   Min.   : 1.000   Min.   :  1.0
##  1st Qu.:1.000   1st Qu.: 2.00   1st Qu.: 1.000   1st Qu.: 40.0
##  Median :1.000   Median : 2.00   Median : 1.000   Median :998.0
##  Mean   :2.706   Mean   :18.56   Mean   : 3.661   Mean   :535.3
##  3rd Qu.:5.000   3rd Qu.: 2.00   3rd Qu.: 1.000   3rd Qu.:999.0
##  Max.   :7.000   Max.   :99.00   Max.   :98.000   Max.   :999.0
##
##     illflag
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.4873
##  3rd Qu.:1.0000
##  Max.   :1.0000
##
```

```
library(dplyr)
library(tidyr)
library(ggplot2)
library(knitr)
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.3.3
```

```
## corrplot 0.95 loaded
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
NSDUH_df <- read.csv("data/NSDUH_data.csv")

# Define missing value codes for each variable
NSDUH_df[NSDUH_df == 99] <- NA
NSDUH_df$LVLDIFCARE2[NSDUH_df$LVLDIFCARE2 %in% c(94, 97, 98)] <- NA
NSDUH_df$NOMARR2[NSDUH_df$NOMARR2 %in% c(94, 97, 99)] <- NA
NSDUH_df$WRKDHRSWK2[NSDUH_df$WRKDHRSWK2 %in% c(985, 994, 997, 998, 999)] <- NA
NSDUH_df$WRKSTATWK2[NSDUH_df$WRKSTATWK2 %in% c(98, 99)] <- NA
NSDUH_df$addprev[NSDUH_df$addprev %in% c(85, 94, 97, 98, 99)] <- NA
NSDUH_df$adrelig[NSDUH_df$adrelig %in% c(94, 97, 98, 99)] <- NA
NSDUH_df$cigtry[NSDUH_df$cigtry %in% c(985, 994, 997)] <- NA
NSDUH_df$hltinmnt[NSDUH_df$hltinmnt %in% c(85, 94, 97, 98, 99)] <- NA
NSDUH_df$service[NSDUH_df$service %in% c(85, 94, 97, 99)] <- NA
NSDUH_df$snrldcsn[NSDUH_df$snrldcsn %in% c(85, 94, 97, 98, 99)] <- NA
NSDUH_df$snrlgsvc[NSDUH_df$snrlgsvc %in% c(85, 94, 97, 98, 99)] <- NA
NSDUH_df$yodprev[NSDUH_df$yodprev %in% c(85, 94, 97, 98, 99)] <- NA
NSDUH_df$yorelig[NSDUH_df$yorelig %in% c(94, 97, 98, 99)] <- NA


summary_stats <- NSDUH_df %>%
```

```
  summarise(across(everything(), list(
    n_obs = ~sum(!is.na(.)),
    n_missing = ~sum(is.na(.)),
    mean = ~if(is.numeric(.)) mean(., na.rm = TRUE) else NA,
    median = ~if(is.numeric(.)) median(., na.rm = TRUE) else NA,
    sd = ~if(is.numeric(.)) sd(., na.rm = TRUE) else NA,
    min = ~if(is.numeric(.)) min(., na.rm = TRUE) else NA,
    max = ~if(is.numeric(.)) max(., na.rm = TRUE) else NA
  )))


summary_long <- summary_stats %>%
  pivot_longer(everything(),
               names_to = c("variable", "stat"),
               names_pattern = "(.*)_(.*)") %>%
  pivot_wider(names_from = stat, values_from = value) %>%
  arrange(variable)


print("Summary Statistics:")
```

```
## [1] "Summary Statistics:"
```

```
print(kable(summary_long, digits = 2))
```

```
##
##
## |variable       |   obs| missing|   mean| median|     sd| min| max|
## |:--------------|-----:|-------:|------:|------:|------:|---:|---:|
## |AGE3           |    NA|      NA|   6.62|      7|   3.07|   1|  11|
## |AGE3_n         | 59069|       0|     NA|     NA|     NA|  NA|  NA|
## |COUTYP4        |    NA|      NA|   1.69|      2|   0.71|   1|   3|
## |COUTYP4_n      | 59069|       0|     NA|     NA|     NA|  NA|  NA|
## |HEALTH2        |    NA|      NA|   2.30|      2|   0.93|   1|   4|
## |HEALTH2_n      | 59056|      13|     NA|     NA|     NA|  NA|  NA|
## |IREDUHIGHST2   |    NA|      NA|   8.19|      9|   2.66|   1|  11|
## |IREDUHIGHST2_n | 59069|       0|     NA|     NA|     NA|  NA|  NA|
## |LVLDIFCARE2    |    NA|      NA|   1.21|      1|   2.78|   1|  85|
## |LVLDIFCARE2_n  | 57564|    1505|     NA|     NA|     NA|  NA|  NA|
## |NEWRACE2       |    NA|      NA|   2.71|      1|   2.44|   1|   7|
## |NEWRACE2_n     | 59069|       0|     NA|     NA|     NA|  NA|  NA|
## |NOMARR2        |    NA|      NA|   1.20|      1|   0.40|   1|   2|
## |NOMARR2_n      | 25501|   33568|     NA|     NA|     NA|  NA|  NA|
## |WRKDHRSWK2     |    NA|      NA|  35.89|     40|  13.53|   1|  61|
## |WRKDHRSWK2_n   | 28437|   30632|     NA|     NA|     NA|  NA|  NA|
## |WRKSTATWK2     |    NA|      NA|   3.62|      2|   3.03|   1|   9|
## |WRKSTATWK2_n   | 51263|    7806|     NA|     NA|     NA|  NA|  NA|
## |addprev        |    NA|      NA|   1.65|      2|   0.48|   1|   2|
## |addprev_n      | 45827|   13242|     NA|     NA|     NA|  NA|  NA|
## |adrelig        |    NA|      NA|   5.56|      6|   1.41|   1|   6|
## |adrelig_n      |  4514|   54555|     NA|     NA|     NA|  NA|  NA|
## |cigtry         |    NA|      NA| 581.38|    991| 481.19|   1| 991|
## |cigtry_n       | 58873|     196|     NA|     NA|     NA|  NA|  NA|
## |hltinmnt       |    NA|      NA|   1.14|      1|   0.35|   1|   2|
```

```
## |hltinmnt_n     | 29986|   29083|     NA|     NA|     NA|  NA|  NA|
## |illflag        |    NA|      NA|   0.49|      0|   0.50|   0|   1|
## |illflag_n      | 59069|       0|     NA|     NA|     NA|  NA|  NA|
## |income         |    NA|      NA|   2.82|      3|   1.15|   1|   4|
## |income_n       | 59069|       0|     NA|     NA|     NA|  NA|  NA|
## |irsex          |    NA|      NA|   1.54|      2|   0.50|   1|   2|
## |irsex_n        | 59069|       0|     NA|     NA|     NA|  NA|  NA|
## |service        |    NA|      NA|   1.95|      2|   0.22|   1|   2|
## |service_n      | 48955|   10114|     NA|     NA|     NA|  NA|  NA|
## |snrldcsn       |    NA|      NA|   2.54|      3|   1.10|   1|   4|
## |snrldcsn_n     | 45584|   13485|     NA|     NA|     NA|  NA|  NA|
## |snrlgsvc       |    NA|      NA|   2.25|      1|   1.73|   1|   6|
## |snrlgsvc_n     | 45918|   13151|     NA|     NA|     NA|  NA|  NA|
## |yodprev        |    NA|      NA|   1.47|      1|   0.50|   1|   2|
## |yodprev_n      | 11746|   47323|     NA|     NA|     NA|  NA|  NA|
## |yorelig        |    NA|      NA|   5.71|      6|   1.16|   1|   6|
## |yorelig_n      |  1232|   57837|     NA|     NA|     NA|  NA|  NA|
```

```r
categorical_vars <- names(NSDUH_df)[sapply(NSDUH_df, function(x)
  length(unique(na.omit(x))) < 10)]

cat("\nFrequency Tables for Categorical Variables:\n")
```

```
##
## Frequency Tables for Categorical Variables:
```

```r
for(var in categorical_vars) {
  cat("\nFrequency table for", var, ":\n")
  print(table(NSDUH_df[[var]], useNA = "ifany"))
}
```

```
##
## Frequency table for irsex :
##
##     1     2
## 27047 32022
##
## Frequency table for WRKSTATWK2 :
##
##     1     2     3     4     5     6     7     8     9  <NA>
## 21486  7042  3020  2425  1707  2277  3196  4326  5784  7806
##
## Frequency table for income :
##
##     1     2     3     4
##  9849 15701  8533 24986
##
## Frequency table for NOMARR2 :
##
##     1     2  <NA>
## 20416  5085 33568
##
## Frequency table for addprev :
##
##     1     2  <NA>
```

```
## 16268 29559 13242
##
## Frequency table for yodprev :
##
##     1     2  <NA>
##  6201  5545 47323
##
## Frequency table for yorelig :
##
##     1     3     6  <NA>
##    70     1  1161 57837
##
## Frequency table for adrelig :
##
##     1     3     6  <NA>
##   387    10  4117 54555
##
## Frequency table for HEALTH2 :
##
##     1     2     3     4  <NA>
## 12666 22204 17714  6472    13
##
## Frequency table for snrldcsn :
##
##     1     2     3     4  <NA>
## 11318  8964 14485 10817 13485
##
## Frequency table for hltinmnt :
##
##     1     2  <NA>
## 25800  4186 29083
##
## Frequency table for snrlgsvc :
##
##     1     2     3     4     5     6  <NA>
## 26443  4538  3300  3901  4052  3684 13151
##
## Frequency table for COUTYP4 :
##
##     1     2     3
## 27015 23581  8473
##
## Frequency table for NEWRACE2 :
##
##     1     2     3     4     5     6     7
## 34169  7155   865   250  3066  2496 11068
##
## Frequency table for service :
##
##     1     2  <NA>
##  2522 46433 10114
##
## Frequency table for LVLDIFCARE2 :
##
```

```
##     1     2     3    85  <NA>
## 51978  4442  1082    62  1505
##
## Frequency table for illflag :
##
##     0     1
## 30287 28782
```

```r
numeric_vars <- names(NSDUH_df)[sapply(NSDUH_df, is.numeric)]

pdf("distribution_plots.pdf", width = 10, height = 8)
for(var in numeric_vars) {
  p <- ggplot(NSDUH_df, aes_string(x = var)) +
    geom_histogram(bins = 30, fill = "lightblue", color = "black") +
    theme_minimal() +
    labs(title = paste("Distribution of", var),
         x = var,
         y = "Count")
  print(p)
}
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
## Warning: Removed 7806 rows containing non-finite values (`stat_bin()`).
## Warning: Removed 33568 rows containing non-finite values (`stat_bin()`).
## Warning: Removed 13242 rows containing non-finite values (`stat_bin()`).
## Warning: Removed 47323 rows containing non-finite values (`stat_bin()`).
## Warning: Removed 57837 rows containing non-finite values (`stat_bin()`).
## Warning: Removed 54555 rows containing non-finite values (`stat_bin()`).
## Warning: Removed 13 rows containing non-finite values (`stat_bin()`).
## Warning: Removed 13485 rows containing non-finite values (`stat_bin()`).
## Warning: Removed 29083 rows containing non-finite values (`stat_bin()`).
## Warning: Removed 13151 rows containing non-finite values (`stat_bin()`).
## Warning: Removed 196 rows containing non-finite values (`stat_bin()`).
## Warning: Removed 10114 rows containing non-finite values (`stat_bin()`).
## Warning: Removed 1505 rows containing non-finite values (`stat_bin()`).
## Warning: Removed 30632 rows containing non-finite values (`stat_bin()`).
```

```r
dev.off()
```

```
## pdf
##   2
```

```r
# Calculate correlations for numeric variables with sufficient observations
valid_numeric_cols <- numeric_vars[sapply(NSDUH_df[numeric_vars], function(x)
```

```
  sum(!is.na(x)) > 30)]


if(length(valid_numeric_cols) > 1) {
  cor_matrix <- cor(NSDUH_df[valid_numeric_cols],
                    use = "pairwise.complete.obs")


  pdf("correlation_plot.pdf", width = 10, height = 8)
  corrplot(cor_matrix,
           method = "color",
           type = "upper",
           tl.col = "black",
           tl.srt = 45,
           addCoef.col = "black",
           number.cex = 0.7,
           diag = FALSE)
  dev.off()
}
```

```
## Warning in cor(NSDUH_df[valid_numeric_cols], use = "pairwise.complete.obs"):
## the standard deviation is zero
```

```
## pdf
##   2
```

```
missing_data <- data.frame(
  Variable = names(NSDUH_df),
  Missing_Percent = sapply(NSDUH_df, function(x) sum(is.na(x))/length(x) * 100)
)


cat("\nMissing Data Summary:\n")
```

```
##
## Missing Data Summary:
```

```
print(kable(missing_data[order(-missing_data$Missing_Percent), ],
           digits = 2))
```

```
##
##
## |             |Variable     | Missing_Percent|
## |:------------|:------------|---------------:|
## |yorelig      |yorelig      |           97.91|
## |adrelig      |adrelig      |           92.36|
## |yodprev      |yodprev      |           80.11|
## |NOMARR2      |NOMARR2      |           56.83|
## |WRKDHRSWK2   |WRKDHRSWK2   |           51.86|
## |hltinmnt     |hltinmnt     |           49.24|
## |snrldcsn     |snrldcsn     |           22.83|
## |addprev      |addprev      |           22.42|
## |snrlgsvc     |snrlgsvc     |           22.26|
## |service      |service      |           17.12|
## |WRKSTATWK2   |WRKSTATWK2   |           13.22|
## |LVLDIFCARE2  |LVLDIFCARE2  |            2.55|
```

```
## |cigtry       |cigtry       |           0.33|
## |HEALTH2      |HEALTH2      |           0.02|
## |AGE3         |AGE3         |           0.00|
## |irsex        |irsex        |           0.00|
## |IREDUHIGHST2 |IREDUHIGHST2 |           0.00|
## |income       |income       |           0.00|
## |COUTYP4      |COUTYP4      |           0.00|
## |NEWRACE2     |NEWRACE2     |           0.00|
## |illflag      |illflag      |           0.00|
```

```
# Save all results to a file
sink("eda_results.txt")
cat("EDA Results\n\n")
```

```
## EDA Results
```

```
cat("1. Summary Statistics:\n")
```

```
## 1. Summary Statistics:
```

```
print(kable(summary_long, digits = 2))
```

```
##
##
## |variable       |   obs| missing|   mean| median|     sd| min| max|
## |:--------------|-----:|-------:|------:|------:|------:|---:|---:|
## |AGE3           |    NA|      NA|   6.62|      7|   3.07|   1|  11|
## |AGE3_n         | 59069|       0|     NA|     NA|     NA|  NA|  NA|
## |COUTYP4        |    NA|      NA|   1.69|      2|   0.71|   1|   3|
## |COUTYP4_n      | 59069|       0|     NA|     NA|     NA|  NA|  NA|
## |HEALTH2        |    NA|      NA|   2.30|      2|   0.93|   1|   4|
## |HEALTH2_n      | 59056|      13|     NA|     NA|     NA|  NA|  NA|
## |IREDUHIGHST2   |    NA|      NA|   8.19|      9|   2.66|   1|  11|
## |IREDUHIGHST2_n | 59069|       0|     NA|     NA|     NA|  NA|  NA|
## |LVLDIFCARE2    |    NA|      NA|   1.21|      1|   2.78|   1|  85|
## |LVLDIFCARE2_n  | 57564|    1505|     NA|     NA|     NA|  NA|  NA|
## |NEWRACE2       |    NA|      NA|   2.71|      1|   2.44|   1|   7|
## |NEWRACE2_n     | 59069|       0|     NA|     NA|     NA|  NA|  NA|
## |NOMARR2        |    NA|      NA|   1.20|      1|   0.40|   1|   2|
## |NOMARR2_n      | 25501|   33568|     NA|     NA|     NA|  NA|  NA|
## |WRKDHRSWK2     |    NA|      NA|  35.89|     40|  13.53|   1|  61|
## |WRKDHRSWK2_n   | 28437|   30632|     NA|     NA|     NA|  NA|  NA|
## |WRKSTATWK2     |    NA|      NA|   3.62|      2|   3.03|   1|   9|
## |WRKSTATWK2_n   | 51263|    7806|     NA|     NA|     NA|  NA|  NA|
## |addprev        |    NA|      NA|   1.65|      2|   0.48|   1|   2|
## |addprev_n      | 45827|   13242|     NA|     NA|     NA|  NA|  NA|
## |adrelig        |    NA|      NA|   5.56|      6|   1.41|   1|   6|
## |adrelig_n      |  4514|   54555|     NA|     NA|     NA|  NA|  NA|
## |cigtry         |    NA|      NA| 581.38|    991| 481.19|   1| 991|
## |cigtry_n       | 58873|     196|     NA|     NA|     NA|  NA|  NA|
## |hltinmnt       |    NA|      NA|   1.14|      1|   0.35|   1|   2|
## |hltinmnt_n     | 29986|   29083|     NA|     NA|     NA|  NA|  NA|
## |illflag        |    NA|      NA|   0.49|      0|   0.50|   0|   1|
## |illflag_n      | 59069|       0|     NA|     NA|     NA|  NA|  NA|
## |income         |    NA|      NA|   2.82|      3|   1.15|   1|   4|
## |income_n       | 59069|       0|     NA|     NA|     NA|  NA|  NA|
```

```
## |irsex          |    NA|      NA|   1.54|      2|   0.50|   1|   2|
## |irsex_n        | 59069|       0|     NA|     NA|     NA|  NA|  NA|
## |service        |    NA|      NA|   1.95|      2|   0.22|   1|   2|
## |service_n      | 48955|   10114|     NA|     NA|     NA|  NA|  NA|
## |snrldcsn       |    NA|      NA|   2.54|      3|   1.10|   1|   4|
## |snrldcsn_n     | 45584|   13485|     NA|     NA|     NA|  NA|  NA|
## |snrlgsvc       |    NA|      NA|   2.25|      1|   1.73|   1|   6|
## |snrlgsvc_n     | 45918|   13151|     NA|     NA|     NA|  NA|  NA|
## |yodprev        |    NA|      NA|   1.47|      1|   0.50|   1|   2|
## |yodprev_n      | 11746|   47323|     NA|     NA|     NA|  NA|  NA|
## |yorelig        |    NA|      NA|   5.71|      6|   1.16|   1|   6|
## |yorelig_n      |  1232|   57837|     NA|     NA|     NA|  NA|  NA|
```

```
cat("\n\n2. Missing Data Summary:\n")
```

```
##
##
## 2. Missing Data Summary:
```

```
print(kable(missing_data[order(-missing_data$Missing_Percent), ],
            digits = 2))
```

```
##
##
## |             |Variable     | Missing_Percent|
## |:------------|:------------|---------------:|
## |yorelig      |yorelig      |           97.91|
## |adrelig      |adrelig      |           92.36|
## |yodprev      |yodprev      |           80.11|
## |NOMARR2      |NOMARR2      |           56.83|
## |WRKDHRSWK2   |WRKDHRSWK2   |           51.86|
## |hltinmnt     |hltinmnt     |           49.24|
## |snrldcsn     |snrldcsn     |           22.83|
## |addprev      |addprev      |           22.42|
## |snrlgsvc     |snrlgsvc     |           22.26|
## |service      |service      |           17.12|
## |WRKSTATWK2   |WRKSTATWK2   |           13.22|
## |LVLDIFCARE2  |LVLDIFCARE2  |            2.55|
## |cigtry       |cigtry       |            0.33|
## |HEALTH2      |HEALTH2      |            0.02|
## |AGE3         |AGE3         |            0.00|
## |irsex        |irsex        |            0.00|
## |IREDUHIGHST2 |IREDUHIGHST2 |            0.00|
## |income       |income       |            0.00|
## |COUTYP4      |COUTYP4      |            0.00|
## |NEWRACE2     |NEWRACE2     |            0.00|
## |illflag      |illflag      |            0.00|
```

```
cat("\n\n3. Categorical Variable Frequencies:\n")
```

```
##
##
## 3. Categorical Variable Frequencies:
```

```
for(var in categorical_vars) {
  cat("\nFrequency table for", var, ":\n")
```

```
  print(table(NSDUH_df[[var]], useNA = "ifany"))
}
```

```
##
## Frequency table for irsex :
##
##     1     2
## 27047 32022
##
## Frequency table for WRKSTATWK2 :
##
##     1     2     3     4     5     6     7     8     9  <NA>
## 21486  7042  3020  2425  1707  2277  3196  4326  5784  7806
##
## Frequency table for income :
##
##     1     2     3     4
##  9849 15701  8533 24986
##
## Frequency table for NOMARR2 :
##
##     1     2  <NA>
## 20416  5085 33568
##
## Frequency table for addprev :
##
##     1     2  <NA>
## 16268 29559 13242
##
## Frequency table for yodprev :
##
##     1     2  <NA>
##  6201  5545 47323
##
## Frequency table for yorelig :
##
##     1     3     6  <NA>
##    70     1  1161 57837
##
## Frequency table for adrelig :
##
##     1     3     6  <NA>
##   387    10  4117 54555
##
## Frequency table for HEALTH2 :
##
##     1     2     3     4  <NA>
## 12666 22204 17714  6472    13
##
## Frequency table for snrldcsn :
##
##     1     2     3     4  <NA>
## 11318  8964 14485 10817 13485
##
```

```
## Frequency table for hltinmnt :
##
##     1     2  <NA>
## 25800  4186 29083
##
## Frequency table for snrlgsvc :
##
##     1     2     3     4     5     6  <NA>
## 26443  4538  3300  3901  4052  3684 13151
##
## Frequency table for COUTYP4 :
##
##     1     2     3
## 27015 23581  8473
##
## Frequency table for NEWRACE2 :
##
##     1     2     3     4     5     6     7
## 34169  7155   865   250  3066  2496 11068
##
## Frequency table for service :
##
##     1     2  <NA>
##  2522 46433 10114
##
## Frequency table for LVLDIFCARE2 :
##
##     1     2     3    85  <NA>
## 51978  4442  1082    62  1505
##
## Frequency table for illflag :
##
##     0     1
## 30287 28782
sink()
```