Code Appendix

```
library(tidyverse)
library(MASS)
library(caret)
library(caretEnsemble)
library(glmnet)
library(corrplot)
library(lubridate)
```

```
set.seed(578)

raw_data <- read_csv("LengthOfStay.csv", col_types = "ccfffffffffffffnnnnnnnnnnncfn")
# glimpse(raw_data)

data <- raw_data %>%
  mutate(vdate_month = month(parse_date(vdate, "%m/%d/%Y"))) %>%
  sample_n(size = 8000) # actually, take 8k samples so we can save 4k for a
# final test of the CV models
# summary(data)
```

```
ggplot(data, aes(lengthofstay)) +
  geom_histogram(binwidth = 1) +
  theme_bw()
```

```
d1 <- data %>%
  dplyr::select(5:15) %>%
  pivot_longer(cols = everything()) %>%
  group_by(name, value) %>%
  summarize(count = n()) %>%
  pivot_wider(id_cols = name, names_from = value, values_from = count)
knitr::kable(d1, caption = "Summary of categorical predictors")
```

```
numdata <- data %>%
  dplyr::select(16:24) %>%
  pivot_longer(everything())
```

```
ggplot(numdata, aes(value)) +
  geom_histogram() +
  facet_wrap(~name, scales = "free") +
  theme_bw()
```

```
cmat <- cor(data[, c(16:24, 28)], method = "spearman")
corrplot(cmat, type = "upper")
```

```
# 1 - set up a for loop for 10-fold cross validation
# 2 - set up train and test indices from data (df); nested for loop?
# 3 - each model will use a different data structure, so just make sure to use
# the same indices to
```

```r
# subset test and train prior to creating the required data structures.
# 4 - apply models and record MSE.
# lm and lasso can use the same data struc (pg. 120 class notes)
# decision tree
# maybe use gams
# 5 - caretEnsemble will actually do this through model methods

set.seed(578)
# split data equally into train and test sets; cross validate using train.
# Apply final models to the test set.
# ToDo drop extra cols that aren't predictors; address rcount variable. Could
# try one-hot encoding. Need to think carefully about how to handle this. It
# is a strange mix of ordinal and numeric data. I think you can make the
# argument to convert the 5+ values to 5?
# first center and scale numeric predictors
los <- data$lengthofstay
data_cent_scale <- data %>%
  dplyr::select(-lengthofstay) %>%
  dplyr::mutate_if(is.numeric, scale) %>%
  dplyr::mutate_if(is.numeric, scale) %>%
  bind_cols(., lengthofstay = los)
train <- sample(1:nrow(data_cent_scale), nrow(data_cent_scale) / 2)
test <- (-train)
train_set <- data_cent_scale[train, -c(1, 2, 3, 13, 25, 26, 28)] # drop id, vdate, discharged,
# vdate_month, fibrosisandother
test_set <- data_cent_scale[test, -c(1, 2, 3, 13, 25, 26, 28)]

library(skimr)
skimmed <- skim_to_wide(data_cent_scale[, -c(1, 2, 3, 13, 25, 26, 28)])
# skimmed
```

```r
# There's a large list of model algos available in caret.
# To get the details, use
# modelLookup(<algorithm>)

knitr::kable(models <- bind_rows(
  modelLookup("rpart"),
  modelLookup("glm"),
  modelLookup("lasso"),
  modelLookup("glm.nb")
), caption = "List of models in caret")
```

```r
# set up for caretEnsemble to CV all models in one go.
library(caretEnsemble)
grid <- 10^seq(10, -2, length = 100)
trainControl_args <- trainControl(
  method = "cv",
  number = 10,
  savePredictions = "final",
  index = createFolds(
    test_set$lengthofstay,
    10
  )
)
```

```r
# createFolds sets up k folds to use across all models

algorithmList <- c(
  "lm",
  "lasso",
  "rpart",
  "glm.nb"
)
tune_list <- list(poisson = caretModelSpec(method = "glm", family = "poisson"))
# family is required for glm and must be passed via caretModelSpec (instead of algorithmList)

set.seed(578)

models <- caretList(lengthofstay ~ ., # model formula
  data = train_set, # training set
  trControl = trainControl_args, # cv params
  methodList = algorithmList, # which models
  tuneList = tune_list, # glm needs this
  continue_on_fail = FALSE # stop if something fails
)

out <- resamples(models)

options(digits = 4)
model_results <- data.frame(
  LM = min(models$lm$results$RMSE),
  POISSON = min(models$poisson$results$RMSE),
  LASSO = min(models$lasso$results$RMSE),
  RPART = min(models$rpart$results$RMSE),
  NB = min(models$glm.nb$results$RMSE)
) %>%
  pivot_longer(cols = everything(), names_to = "Model", values_to = "RMSE") %>%
  arrange(RMSE)
```

```r
NB_link_tune <- models[["glm.nb"]][["results"]]
knitr::kable(NB_link_tune, caption = "Negative binomial CV results for link function")
```

```r
# ?resamples
resamples <- resamples(models)
dotplot(resamples, metric = "RMSE")
```

```r
plot(varImp(models$rpart))
```

```r
rpart.plot::rpart.plot(models$rpart$finalModel)
```

```r
plot(varImp(models$glm.nb))
```

```r
# evaluate final models on test data
pred_lm <- predict.train(models$lm, newdata = test_set)
pred_poisson <- predict.train(models$poisson, newdata = test_set)
pred_lasso <- predict.train(models$lasso, newdata = test_set)
pred_rpart <- predict.train(models$rpart, newdata = test_set)
pred_glm <- predict.train(models$glm.nb, newdata = test_set)
pred_RMSE <- data.frame(
  LM = RMSE(pred_lm, test_set$lengthofstay),
```

```
  POISSON = RMSE(pred_poisson, test_set$lengthofstay),
  LASSO = RMSE(pred_lasso, test_set$lengthofstay),
  RPART = RMSE(pred_rpart, test_set$lengthofstay),
  NB = RMSE(pred_glm, test_set$lengthofstay)
) %>%
  pivot_longer(cols = everything(), names_to = "Model", values_to = "RMSE") %>%
  arrange(RMSE)
knitr::kable(pred_RMSE, caption="RMSE of final models across the test data")
```

```
nb_coeffs <- as.data.frame(models$glm.nb$finalModel$coefficients) %>%
  set_names("coefficient") %>%
  arrange(desc(abs(coefficient)))
knitr::kable(nb_coeffs, caption = "Negative binomial model coefficients")
```