Classification code

```
library(tidyverse)
```

```
## Loading tidyverse: ggplot2
## Loading tidyverse: tibble
## Loading tidyverse: tidyr
## Loading tidyverse: readr
## Loading tidyverse: purrr
## Loading tidyverse: dplyr
```

```
## Warning: package 'dplyr' was built under R version 3.4.2
```

```
## Conflicts with tidy packages ----------------------------------------------
```

```
## filter(): dplyr, stats
## lag():    dplyr, stats
```

```
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 3.4.2
```

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

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

```
## The following object is masked from 'package:tidyr':
##
##     expand
```

```
## Loading required package: foreach
```

```
##
## Attaching package: 'foreach'
```

```
## The following objects are masked from 'package:purrr':
##
##     accumulate, when
```

```
## Loaded glmnet 2.0-13
```

```
library(modelr)
library(plotROC)
```

```
#got approval to use entire dataset, so we're not splitting into test and train
by_subj <- read_csv("~/GitHub/skaltman/MSE226/data/relapse_subjects.csv", na = c("", "NaN"))
```

Functions used to calculate errors

```
test_err <- function(test, mod, threshold) {
  test <- as_tibble(test)

  errors <-
    test %>%
    mutate(
      pred = predict(mod, newdata = test, type = "response"),
      pred_group = ifelse(pred > threshold, 1, 0),
      pred_error = ifelse(relIn6Mos != pred_group, 1, 0)
    )
```

```
  return(mean(errors$pred_error))
}

test_err_fn <- function(test, mod, threshold) {
  test <- as_tibble(test)

  errors <-
    test %>%
    mutate(
      pred = predict(mod, newdata = test, type = "response"),
      pred_group = ifelse(pred > threshold, 1, 0),
      false_neg = ifelse(relIn6Mos > pred_group, 1, 0)
    )

  return(mean(errors$false_neg))
}

test_err_fp <- function(test, mod, threshold) {
  test <- as_tibble(test)

  errors <-
    test %>%
    mutate(
      pred = predict(mod, newdata = test, type = "response"),
      pred_group = ifelse(pred > threshold, 1, 0),
      false_pos = ifelse(relIn6Mos < pred_group, 1, 0)
    )

  return(mean(errors$false_pos))
}
```

Cross-validated logistic regression (LOOCV)

```
set.seed(1)

by_subj_cv <-
  by_subj %>%
  crossv_kfold(31)

#for threshold = .5
by_subj_cv %>%
  mutate(train = map(train, as_tibble),
         model = map(train, ~glm(relIn6Mos ~ nacc_neutral_beta,
                                 family = "binomial",
                                 data = .)),
         error = map2_dbl(test, model, test_err, threshold = .45),
         false_neg_rate = map2_dbl(test, model, test_err_fn, threshold = .45),
         false_pos_rate = map2_dbl(test, model, test_err_fp, threshold = .45)) %>%
  summarise(mean_error = mean(error),
            mean_fn_rate = mean(false_neg_rate),
            mean_fp_rate = mean(false_pos_rate)) %>%
  knitr::kable()
```

mean_error	mean_fn_rate	mean_fp_rate
0.1774194	0.1129032	0.0645161

The following code fits the model on all the data:

```
fit <- glm(relIn6Mos ~ nacc_neutral_beta, family = "binomial", data = by_subj)
fit
```

```
##
## Call:  glm(formula = relIn6Mos ~ nacc_neutral_beta, family = "binomial",
##     data = by_subj)
##
## Coefficients:
##       (Intercept)  nacc_neutral_beta
##           -0.3872            21.0739
##
## Degrees of Freedom: 38 Total (i.e. Null);  37 Residual
## Null Deviance:       50.92
## Residual Deviance: 38.64     AIC: 42.64
```

Error rates for different thresholds:

```
set.seed(1)

false_neg_rates <- rep(0, 20)
false_pos_rates <- rep(0, 20)
error_rates <- rep(0, 20)

for (t in seq(.05, 1, .05)) {
  curr_est <-
    by_subj_cv %>%
      mutate(train = map(train, as_tibble),
             model = map(train, ~glm(relIn6Mos ~ nacc_neutral_beta,
                                     family = "binomial",
                                     data = .)),
              false_neg_rate = map2_dbl(test,
                                        model,
                                        test_err_fn,
                                        threshold = t),
              false_pos_rate = map2_dbl(test,
                                        model,
                                        test_err_fp,
                                        threshold = t),
              error = map2_dbl(test,
                               model,
                               test_err,
                               threshold = t)) %>%
      summarise(mean_fn_rate = mean(false_neg_rate),
                mean_fp_rate = mean(false_pos_rate),
                mean_err_rate = mean(error))

  false_pos_rates[t*20] = curr_est$mean_fp_rate
  false_neg_rates[t*20] = curr_est$mean_fn_rate
  error_rates[t*20] = curr_est$mean_err_rate
```

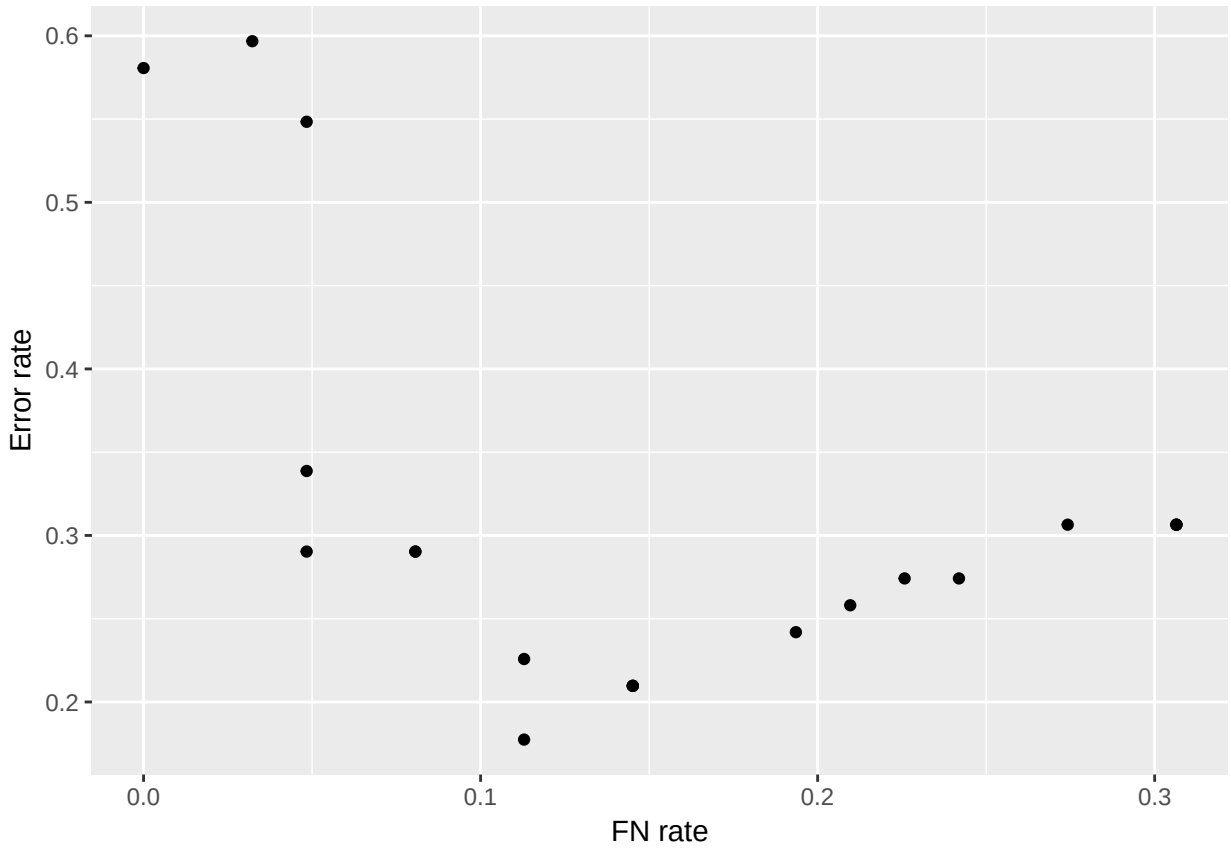
```
}

error_tibble <-
  tibble(`FN rate` = false_neg_rates,
         `Error rate` = error_rates,
         Threshold = seq(.05, 1, .05))

error_tibble %>%
  filter(Threshold < .6) %>%
  knitr::kable()
```

FN rate	Error rate	Threshold
0.0000000	0.5806452	0.05
0.0322581	0.5967742	0.10
0.0483871	0.5483871	0.15
0.0483871	0.3387097	0.20
0.0483871	0.2903226	0.25
0.0806452	0.2903226	0.30
0.0806452	0.2903226	0.35
0.1129032	0.2258065	0.40
0.1129032	0.1774194	0.45
0.1451613	0.2096774	0.50
0.1451613	0.2096774	0.55

```
error_tibble %>%
  ggplot(aes(`FN rate`, `Error rate`)) +
  geom_point()
```



ROC curve

```
set.seed(1)

for_roc <-
  by_subj_cv %>%
  mutate(train = map(train, as_tibble),
         model = map(train, ~glm(relIn6Mos ~ nacc_neutral_beta,
                                 family = "binomial",
                                 data = .)))

false_pos_rates <- rep(0, 20)
true_pos_rates <- rep(0, 20)

for (t in seq(0.05, 1, .05)) {
  curr_est <-
    for_roc %>%
      mutate(true_pos_rate = 1 - map2_dbl(test,
                                          model,
                                          test_err_fn,
                                          threshold = t),
             false_pos_rate = map2_dbl(test,
                                       model,
                                       test_err_fp,
                                       threshold = t)) %>%
      summarise(mean_tp_rate = mean(true_pos_rate),
                mean_fp_rate = mean(false_pos_rate))
```

```
  false_pos_rates[t*20] = curr_est$mean_fp_rate
  true_pos_rates[t*20] = curr_est$mean_tp_rate
}


tibble(true_pos_rates, false_pos_rates, t = seq(0.05, 1, .05)) %>%
  ggplot(aes(false_pos_rates, true_pos_rates)) +
  geom_point(aes(size = t)) +
  geom_line() +
  labs(x = "False positive rate",
       y = "True positive rate",
       title = "ROC")
```