Jasen_Regularization

Jasen Zhang

11/21/2020

1 Loading the Data

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

##
## Attaching package: 'gridExtra'

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

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

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

## Loaded glmnet 4.0-2
```

2 Splitting Test and Training Set

Arbitrarily chose 10% to be test set

```
set.seed(221)
test_set_index <- sample(1:nrow(df), floor(nrow(df))/10)
train_set_index <- setdiff(1:nrow(df), test_set_index)

test <- df[test_set_index,]
train <- df[train_set_index,]

test_y <- test %>% select(response_vars)
test_x <- test %>% select(vars_1)
train_y <- train %>% select(response_vars)
train_x <- train %>% select(vars_1)
```

3 Lasso + Leave one out Cross Validation

10 folds since that's default

```
lambda_seq <- 10^seq(-2, -1, by = .05)

cv_output <- cv.glmnet(as.matrix(train_x), as.matrix(train_y),
                       alpha = 1, lambda = lambda_seq)

best_lambda <- cv_output$lambda.min

lasso_best <- glmnet(as.matrix(train_x), as.matrix(train_y), alpha = 1, lambda = best_lambda)

pred_test_y <- predict(lasso_best, s = best_lambda, newx = as.matrix(test_x))
pred_train_y <- predict(lasso_best, s = best_lambda, newx = as.matrix(train_x))
```

3.5 Plotting Graph of Hyperparameter

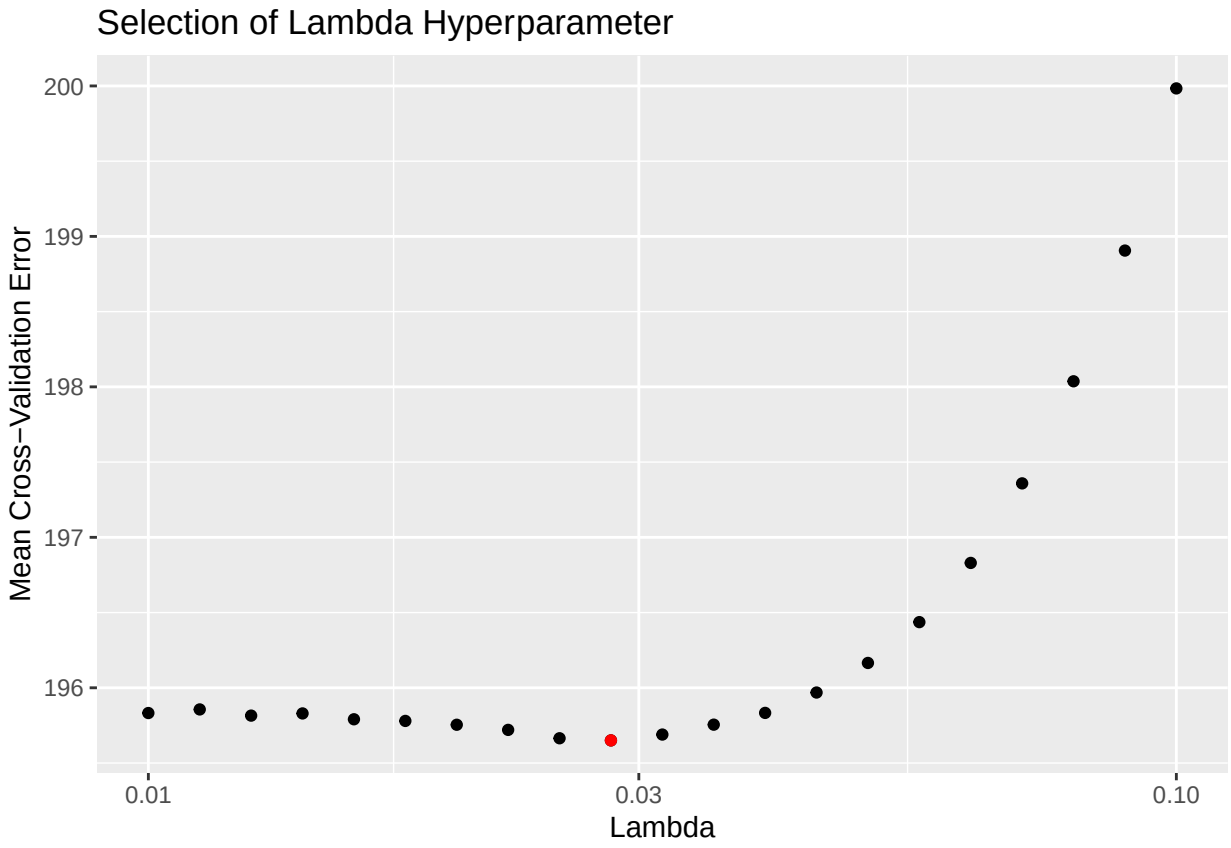
```
cv_error <- cv_output$cvm
lambdas <- cv_output$lambda
hyper <- data.frame(lambdas, cv_error)

# taking the minimum
min_cv_error <- min(hyper$cv_error)
min_df <- hyper %>% filter(cv_error == min_cv_error)

g_hyper <- ggplot() + geom_point(data = hyper, aes(x = lambdas, y = cv_error)) +
  geom_point(data = min_df, aes(x = lambdas, y = cv_error), color = 'red') +
  scale_x_continuous(trans = 'log10') +
  ylab('Mean Cross-Validation Error') +
```

```
  xlab('Lambda') +
  ggtitle('Selection of Lambda Hyperparameter')

g_hyper
```

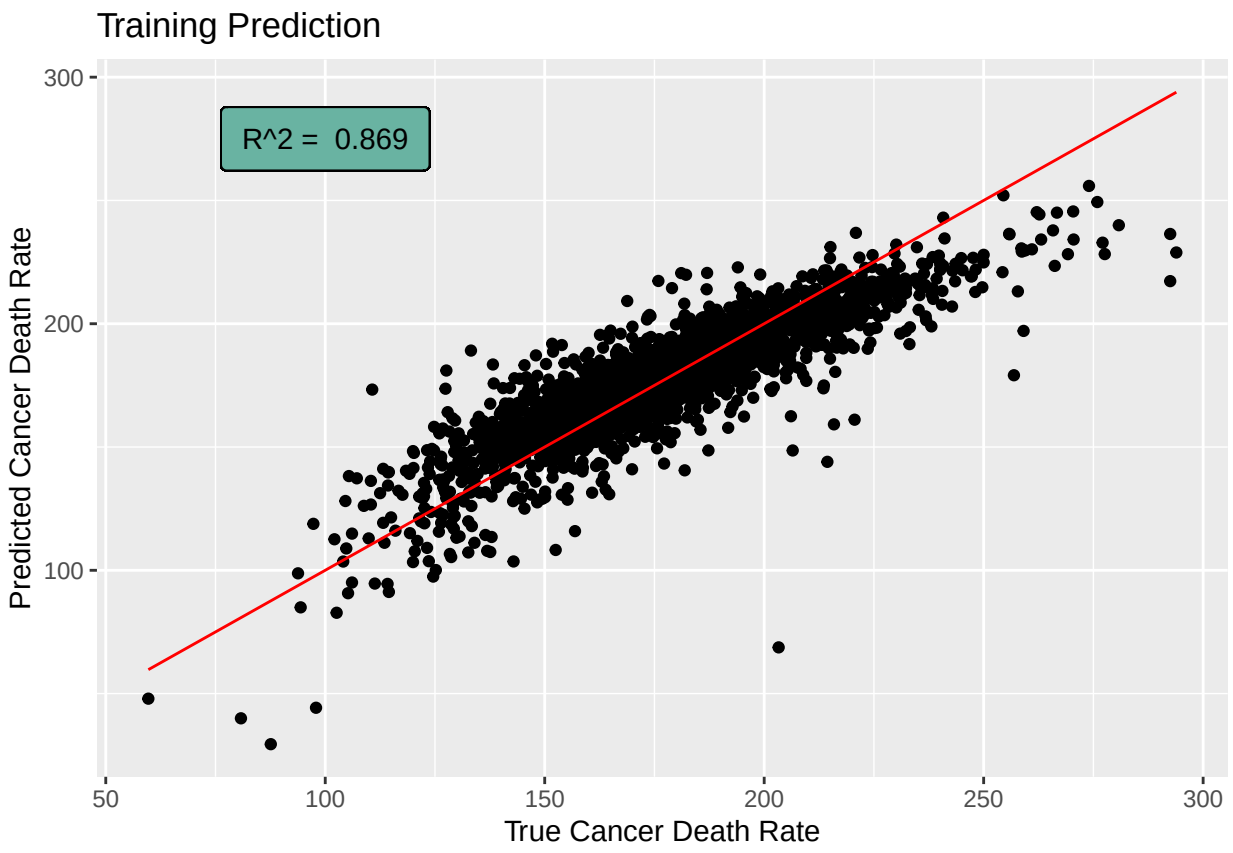


4 Plotting Y_hat vs Y for training set

```
train_results <- data.frame(train_y, pred_train_y)
colnames(train_results) <- c('y_true', 'y_hat')
R_squared <- as.numeric(unname(cor(train_y, pred_train_y)))
R_squared <- sprintf("%.3f", round(R_squared,3))
R_squared_label <- paste('R^2 = ', R_squared)

g <- ggplot(train_results) +
  geom_point(aes(x = y_true, y= y_hat)) +
  geom_line(aes(x = y_true, y = y_true), color = 'red')  +
  geom_label(label = R_squared_label, x = 100, y = 275, label.padding = unit(0.55, "lines"),
    label.size = 0.35,
    color = "black",
    fill="#69b3a2") +
  ylab('Predicted Cancer Death Rate') +
  xlab('True Cancer Death Rate') +
  ggtitle('Training Prediction')
```

```
g
```




```
rmse_lasso_train <- sqrt(sum((train_results$y_true - train_results$y_hat)^2)/nrow(train_results))
r_squared_lasso_train <- R_squared
```

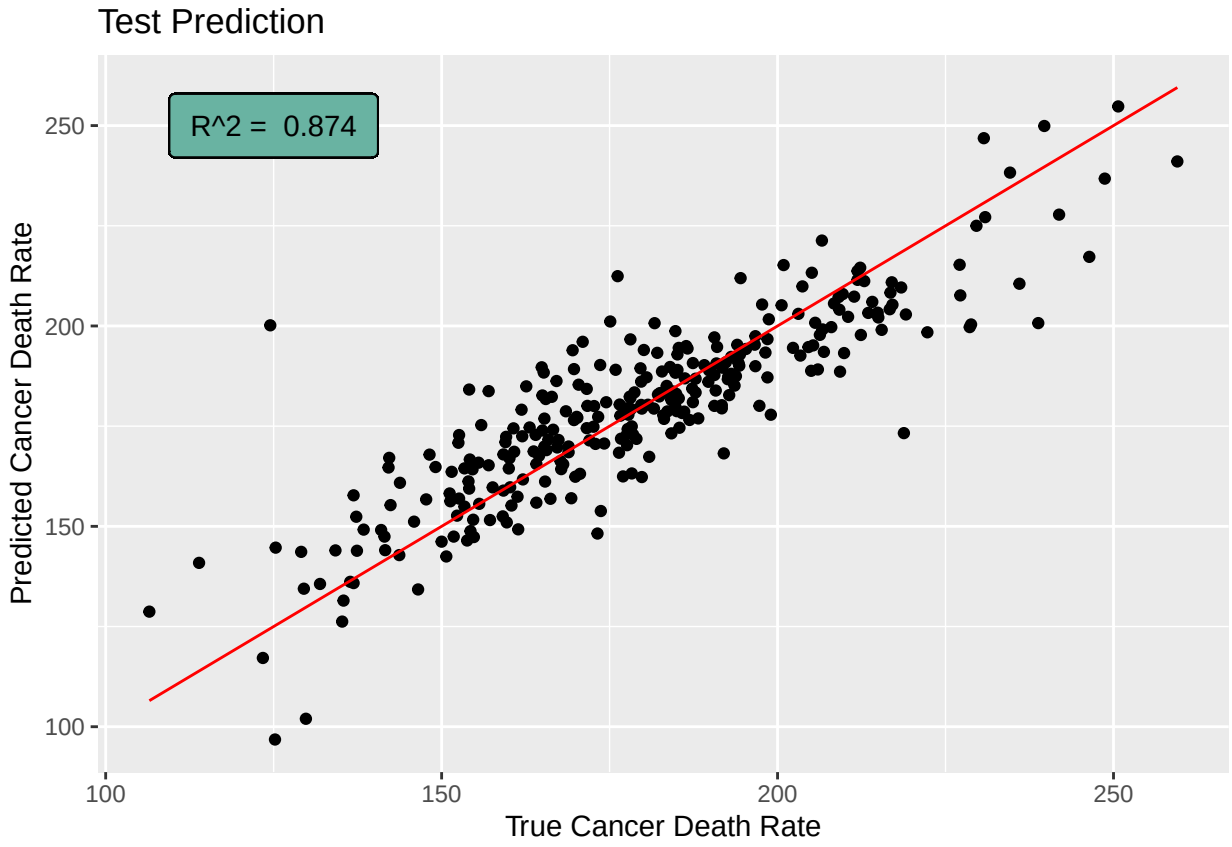
5 Plotting Y_hat vs Y for test set

```
test_results <- data.frame(test_y, pred_test_y)
colnames(test_results) <- c('y_true', 'y_hat')
R_squared <- as.numeric(unname(cor(test_y, pred_test_y)))
R_squared <- sprintf("%.3f", round(R_squared,3))
R_squared_label <- paste('R^2 = ', R_squared)

g <- ggplot(test_results) +
  geom_point(aes(x = y_true, y= y_hat)) +
  geom_line(aes(x = y_true, y = y_true), color = 'red')  +
  geom_label(label = R_squared_label, x = 125, y = 250, label.padding = unit(0.55, "lines"),
    label.size = 0.35,
    color = "black",
    fill="#69b3a2") +
  ylab('Predicted Cancer Death Rate') +
  xlab('True Cancer Death Rate') +
```

```
  ggtitle('Test Prediction')

g
```




```
rmse_lasso_test <- sqrt(sum((test_results$y_true - test_results$y_hat)^2)/nrow(test_results))
r_squared_lasso_test <- R_squared
```