Forward Selection

Jasen

12/3/2020

1. Load 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

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

3 Forward and Backwards Selection

3.5 Important features

```r
features <- 'incidenceRate + povertyPercent + PctHS18_24 + avgDeathsPerYear_log + popEst2015_log + Medi
features2 <- unlist(strsplit(features, split = ' + ', fixed = T))
```

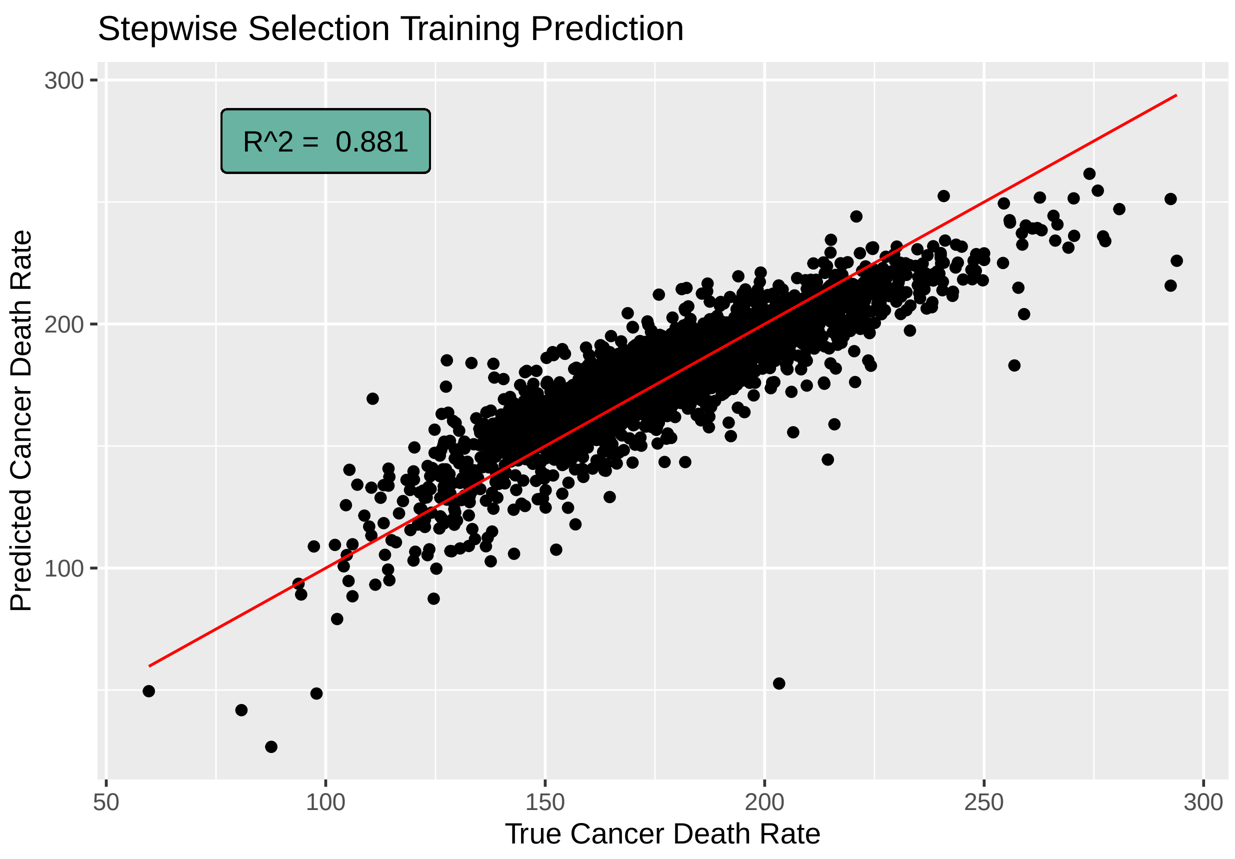
4 Train Statistics

```r
train_x <- train_x %>% select(features2)
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use `all_of(features2)` instead of `features2` to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```r
pred_train_y <- predict(step_model, data = train_x)

RMSE_step_train <- sqrt(sum((pred_train_y - train_y)^2)/length(train_y))

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_step_train <- R_squared
R_squared <- sprintf("%.3f", round(R_squared,3))
R_squared_label <- paste('R^2 = ', R_squared)
```

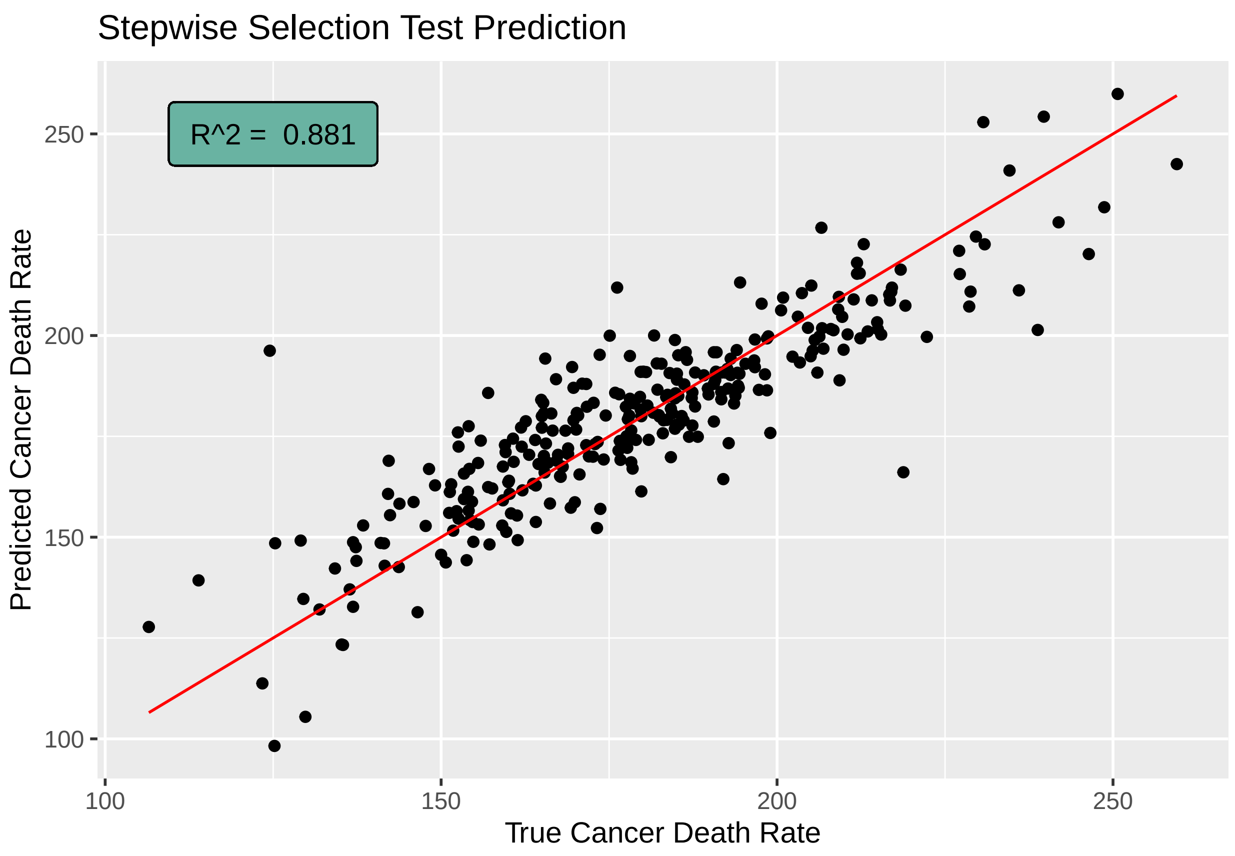
```r
g <- ggplot(train_results) +
  geom_point(aes(x = y_true, y= y_hat)) +
  geom_line(aes(x = y_true, y = y_true), color = 'red')  +
  geom_label(label = R_squared_label, x = 100, y = 275, label.padding = unit(0.55, "lines"),
    label.size = 0.35,
    color = "black",
    fill="#69b3a2") +
  ylab('Predicted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

g
```



4 Test Statistics

```r
test_x <- test_x %>% select(features2)
pred_test_y <- unname(predict(step_model, test_x))

RMSE_step_test <- sqrt(sum((pred_test_y - test_y)^2)/length(test_y))

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
R_squared_step_test <- R_squared
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('Stepwise Selection Test Prediction')

g
```



```r
RMSE_step_train
```

```
## [1] 13.09664
```

```r
R_squared_step_train
```

```
## [1] 0.8805349
```

```r
RMSE_step_test
```

```
## [1] 12.4312
```

```r
R_squared_step_test
```

```
## [1] 0.8809609
```