Veronica Vargas

Homework 6

Due date: 11/07/2024

October 31, 2024

1 Introduction

This exercise sheet contains a series of problems designed to test and enhance your understanding of the topics covered in the course. Please ensure that you attempt all problems and provide detailed solutions where necessary. If you have any questions or need clarification, feel free to reach out your TA.

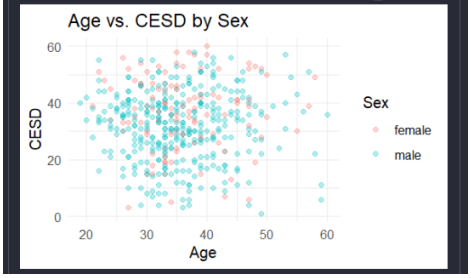
2 Exercises

Exercise 1: Cross-validating like there's no tomorrow

For this exercise, the following libraries might be useful: tidyverse, mosaicData, leaps, caret, ISLR2. You are going to analyze the HELPrct data set inside the mosaicData library.

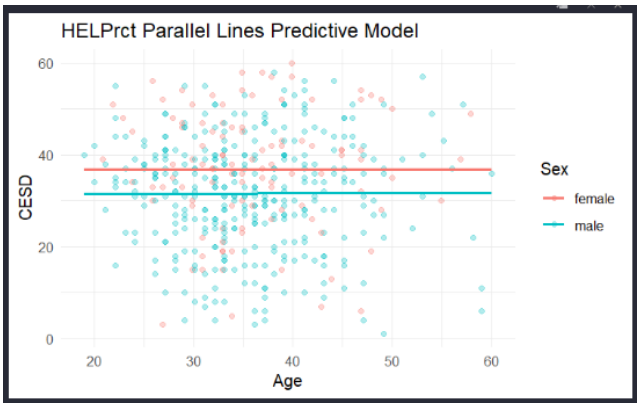
(a) Read the HELPrct data inside the mosaicData library (use the command data(HELPrct)). Generate a proper plot with cesd on the Y-axis, age on the X-axis, and a different color according to the sex. **[5 pts]**

```r
{r}
ggplot(HELPrct, aes(x = age, y = cesd)) +
  geom_point(aes(x = age, y = cesd,
                                color = factor(sex
)),
             alpha = 0.3) +
  labs(
    title = "Age vs. CESD by Sex",
    x = "Age",
    y = "CESD",
    color = "Sex") +
  theme_minimal()
```



(b) Fit a "parallel lines" model with a different intercept for each sex group. Plot your lines on ascatter plot with cesd on the Y-axis, age on the X-axis, and a different color line according to the sex. **[5 pts]**

```r
{r}
age_seq <- seq(min(HELPrct$age), max(HELPrct$age), by = 1)

pred_data <- expand.grid(
  age = age_seq,
  sex = c(0, 1))

pred_data$sex <- factor(pred_data$sex,
                        levels = c(0, 1),
                        labels = c("male", "female"))

pred_data$predicted_cesd <- predict(mod.coarsen1, newdata = pred_data)

ggplot() +
  geom_point(data = HELPrct, aes(x = age, y = cesd, color = sex),
             alpha = 0.3, position=position_dodge(width=0.5)) +
  geom_line(data = pred_data, aes(x = age, y = predicted_cesd,
                                  color = sex), size = 1) +
  labs(
    title = "HELPrct Parallel Lines Predictive Model",
    x = "Age",
    y = "CESD",
    color = "Sex"
  ) +
  theme_minimal()
```

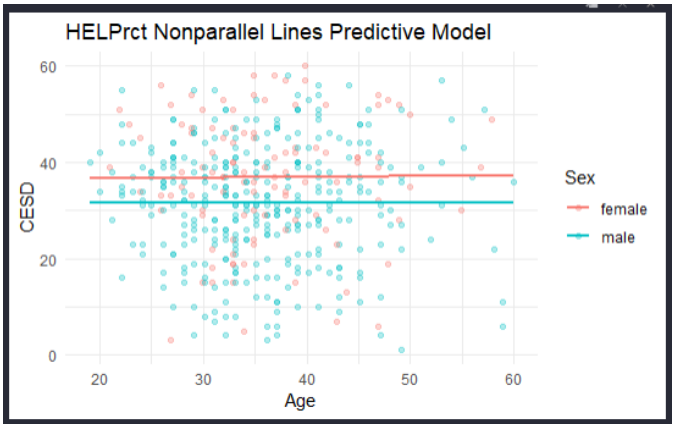


(c) Using the model defined above, what is the predicted cesd for a subject aged 33 and sex = "male"? **[5 pts]**

The predicted cesd for a subject aged 33 and sex = "male" using the parallel lines model is ~31.

(d) Now fit a "not necessarily parallel lines" model with a different intercept and slope for eachsex group. Plot your lines on a scatter plot with cesd on the Y-axis, age on the X-axis, and a different color line according to the sex. **[5 pts]**

```r
{r}
age_seq <- seq(min(HELPrct$age), max(HELPrct$age), by = 1)

pred_data <- expand.grid(
  age = age_seq,
  sex = c(0, 1))

pred_data$sex <- factor(pred_data$sex,
                        levels = c(0, 1),
                        labels = c("male", "female"))

pred_data$predicted_cesd <- predict(mod.coarsen2, newdata = pred_data)

ggplot() +
  geom_point(data = HELPrct, aes(x = age, y = cesd, color = sex),
             alpha = 0.3, position=position_dodge(width=0.5)) +
  geom_line(data = pred_data, aes(x = age, y = predicted_cesd,
                                  color = sex), size = 1) +
  labs(
    title = "HELPrct Nonparallel Lines Predictive Model",
    x = "Age",
    y = "CESD",
    color = "Sex"
  ) +
  theme_minimal()
```



(e) Using the model defined above, what is the predicted cesd for a subject aged 33 and sex = "male"? **[5 pts]**

The predicted cesd for a subject aged 33 and sex = "male" using the nonparallel lines model is ~31.

Now, you are asked to compare the two models using the validation set approach.

(f) Using the validation set approach, estimate the out-of-sample Residual Sum of Squares (RSS) of the model in (a) and the model in (c). Use the same training and test set for both the model. The

training set should present around 70% of the observations in your sample; the validation set should include the remaining 30%. **[10 pts]**

```r
set.seed(42)
train_indices <- sample(1:n, size = 0.7 * n)
train_data <- HELPrct[train_indices, ]
test_data <- HELPrct[-train_indices, ]
```

```r
# Model A: Parallel Lines Model
model_A_split <- lm(cesd ~ sex + age, data = train_data)
predictions_A_split <- predict(model_A_split, test_data)
rss_A_split <- sum((test_data$cesd - predictions_A_split)^2)
rss_A_split


# Model B: Nonparallel Lines Model
model_B_split <- lm(cesd ~ age + sex + age*sex, data = train_data)
predictions_B_split <- predict(model_B_split, test_data)
rss_B_split <- sum((test_data$cesd - predictions_B_split)^2)
rss_B_split
```

```
[1] 21798.25
[1] 21800.43
```

(g) Repeat the procedure in (f) three times. Every time you should have different training and validation (try setting a different seed each time to avoid to get the same sets) and estimate the test error (RSS) for both the models. Compute the average of the three RSS for model (a) and model (c). **[5 pts]**

```
[1] 18852.89
[1] 18913.66
```

```
[1] 9252.507
[1] 9254.149
```

```
[1] 1869.503
[1] 1868.867
```

Average Model (a) : 9991.63

Average Model (c) : 10012.23

(h) Based on the results of (g), which model is better? Explain why. **[5 pts]**

Based on the results of (g), the parallel lines model is better because it features a lower average RSS than the nonparallel lines model. In other words, there are less errors in the parallel lines model than there are in the nonparallel lines model.

Now, you are asked to compare the two models using the Leave-One-Out Cross-Validation (LOOCV) approach.

(i) Use the LOOCV to estimate the test error (RSS) of the model in (a) and the model in (c). **[10 pts]**

```r
rss_summary <- function(data, lev = NULL, model = NULL) {
  residuals <- data$obs - data$pred
  rss <- sum(residuals^2)
  rmse <- sqrt(mean(residuals^2))
  return(c(RMSE = rmse, RSS = rss))
}
```

```r
train_control_loocv <- trainControl(
  method = "LOOCV",
  summaryFunction = rss_summary,
  savePredictions = "all",
  classProbs = FALSE,
  allowParallel = FALSE
)

# Train Model A: Parallel Lines Model
set.seed(100)
model_A_caret_loocv <- train(
  cesd ~ sex + age,
  data = HELPrct,
  method = "lm",
  trControl = train_control_loocv,
  metric = "RMSE"
)

# Train Model B: Nonparallel Lines Model
set.seed(100)
model_B_caret_loocv <- train(
  cesd ~ age + sex + age*sex,
  data = HELPrct,
  method = "lm",
  trControl = train_control_loocv,
  metric = "RMSE"
)

model_A_caret_loocv$results
model_B_caret_loocv$results
```

Parallel Lines Model :

RMSE <dbl>	RSS <dbl>
12.38263	69458.28

Nonparallel Lines Model :

RMSE <dbl>	RSS <dbl>
12.41142	69781.69

(j) Does it make sense to repeat the estimate of the two RSS multiple times? If yes, do it three times (three for model (a) and three for model (c)). If you don't think it makes sense, explain why. **[5 pts]**

Since the LOOCV is saving the fitting process for each observation in the dataset, then it would not make sense to repeat the two RSS multiple times. This is built into the LOOCV.

(k) Based on the results of (j), which model is better? Explain why. **[5 pts]**

Based on the results of (j), the parallel lines model would be better because it features a lower RSS in comparison to the nonparallel lines model. This means that the parallel lines model contains less error than the nonparallel lines model.

Now, you are asked to compare the two models using *k*-folds cross-validation with $k = 10$

(l) Use 10-folds cross-validation to estimate the out-of-sample Residual Sum of Squares (RSS) of the model in (a) and the model in (c). Please, use the same 10 folds for both the models (*Hint: how do we prepare our folds? Set the folds only one time and use it for both the models*). **[10 pts]**

```r
{r}
train_control_kfold <- trainControl(
  method = "cv",
  number = 10,
  summaryFunction = rss_summary,
  savePredictions = "final",
  classProbs = FALSE,
  allowParallel = FALSE
)

# Train Model A: Parallel Lines Model
set.seed(123)
model_A_caret <- train(
  cesd ~ sex + age,
  data = HELPrct,
  method = "lm",
  trControl = train_control_kfold,
  metric = "RMSE"
)

# Train Model B: Nonparallel Lines Model
set.seed(123)
model_B_caret <- train(
  cesd ~ age + sex + age*sex,
  data = HELPrct,
  method = "lm",
  trControl = train_control_kfold,
  metric = "RMSE"
)

model_A_caret$results
model_B_caret$results
```

Parallel Lines Model :

RMSE <dbl>	RSS <dbl>
12.33696	6915.772

Nonparallel Lines Model :

RMSE <dbl>	RSS <dbl>
12.36503	6947.187

(m) Based on the results of (l), which model is better? Explain why. **[5 pts]**

Based on the results of (l), the parallel lines model would be better because it features a lower RSS than the nonparallel lines model. This means that there is less error in the parallel lines model than there is in the nonparallel lines model.

Now, you are asked to perform the validation set approach together with bootstrapping. (n)

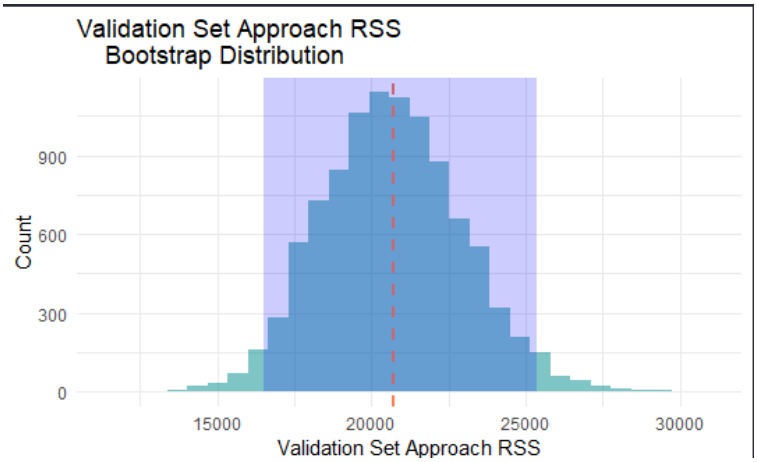
Code a function following the following steps: **[10 pts]**

- Create a random sample with replacement from your data.
- Apply the validation set approach to the random sample to estimate the out-of-sample RSS for model (a). The split should assign 70% of the observations to the training set and the remaining 30% to the validation set.
- Return the out-of-sample RSS for model (a)

```r
{r}
rss_A_split_boot <- function(data, B = 10000, train_size = 0.7,
set_seed = 123) {

  set.seed(set_seed)
  rss_A_split_boot <- numeric(B)
  n <- nrow(data)

  for (b in 1:B) {
    boot_sample <- data[sample(1:n, size = n, replace = TRUE), ]

    train_indices <- sample(1:n, size = train_size * n)
    train_data <- boot_sample[train_indices, ]
    test_data <- boot_sample[-train_indices, ]

    model_A_split <- lm(cesd ~ sex + age, data = train_data)
    predictions_A_split <- predict(model_A_split, test_data)
    rss_A_split_boot[b] <- sum((test_data$cesd - predictions_A_split
)^2)
  }
  return(rss_A_split_boot)
}

modelA_boot_rss <- rss_A_split_boot(HELPrct, B = 10000)
```

(o) Use the function just created multiple times (10000 or less if your computer is not performative enough) to obtain the estimated sampling distribution of the out-of-sample RSS for model (a). Plot the distribution and describe it. **[5 pts]**

```r
{r}
ggplot(data = data.frame(rss_A_split = modelA_boot_rss), aes(x =
rss_A_split)) +
  geom_histogram(fill = "cyan4", alpha = 0.5, position = "identity") +
  geom_vline(xintercept = mean(modelA_boot_rss),
             linetype="dashed",
             color = "coral", linewidth=1) +
  annotate("rect", xmin = quantile(modelA_boot_rss, 0.025),
                   xmax = quantile(modelA_boot_rss, 0.975),
                   ymin = 0, ymax = Inf, fill = "blue",
                   alpha = 0.2) +
  labs(
    title = "Validation Set Approach RSS
    Bootstrap Distribution",
    x = "Validation Set Approach RSS",
    y = "Count")
  theme_minimal()
```



Here, the graph features a normal distribution where the bootstrap distribution is symmetrical along the mean.

(p) What is the difference between the approach of the previous question and repeating multiple times (10000, for instance) the validation set approach implemented in (f)? (*Hint: to answer this*

question you are not obliged to repeat multiple times the validation set approach implemented in (f): the question can be answered from an algorithmic and theory based perspective)
[5 pts]

The difference between the previous question and repeating the validation set approach in (f) is that the previous question is taking multiple samples with replacement whereas the validation set approach in (f) would entail drawing from the same data. By sampling with replacement in the previous question, we reduce sampling bias by drawing conclusions from multiple samples. Therefore, the error in each sample is reduced, while repeating the validation set approach in question (f) would yield the same results for every iteration.

Now, you are asked to answer theoretical questions.

(q) Comparing the LOOCV and k-fold Cross validation strategies, which one has the lowest bias? Explain why. Which one has the lowest variance in the test error? Explain Why. Which one is more computational intensive? Explain why. **[5 pts]**

The LOOCV has the lowest bias because it uses the most information to provide a conclusion. The k-fold cross validation has the lowest variance because it uses averages instead of placing a large amount of weight on any one observation like in the LOOCV. The LOOCV is more computationally intensive because it is training data on each observation in the dataset.

(r) Order the validation set approach, LOOCV, 10-folds CV, and the bootstrapped validation set approach (implemented in (o)) from least to most computationally intensive. Explain why. **[5 pts]**

Validation set approach; 10-folds CV; bootstrapped validation; LOOCV. The validation set approach uses only the training data and the validation data to form a conclusion, therefore it is the least computationally intensive. The 10-folds validation is dividing the data into 10 sections and training the data on 9 out of the 10 sections. Therefore, this is slightly more complicated than the validation set approach. The bootstrapped validation is repeating the same validation set approach (with the same % split) using random samples drawn from the population with replacement. It is not doing it multiple times like a k-fold model, but it has many iterations and is, therefore, more computationally intensive than the 10-folds CV. Finally, the LOOCV is the most computationally intensive because it repeats the training and validation process for every observation in the dataset.

3 Submission Instructions

Please submit your completed exercises by **October 31** through **gradescope**. Ensure that your solutions are well-organized, clearly written, and include all necessary calculations and explanations. Questions about submission should be directed to your TA.

4 Helpful Resources

To better assist you in the completion of this exercise sheet, we suggest you to review the following material:

- **Lecture 3** - bootstrapping estimated sampling distributions and confidence intervals
- **Lecture 11** - least squares regression;
- **Lecture 12** - bias in least squares regression;
- **Lecture 14** - bias and variance trade-off and cross-validation; • **Lab** - practicing all of the above