Portfolio 3

Kiri Koppelgaard

September 28, 2018

```
{r setup, include=FALSE} knitr::opts_chunk$set(echo = TRUE)
```

Welcome to the third exciting part of the Language Development in ASD exercise

In this exercise we will delve more in depth with different practices of model comparison and model selection, by first evaluating your models from last time, then learning how to cross-validate models and finally how to systematically compare models.

N.B. There are several datasets for this exercise, so pay attention to which one you are using!

1. The (training) dataset from last time (the awesome one you produced :-)).
2. The (test) datasets on which you can test the models from last time:

- Demographic and clinical data: https://www.dropbox.com/s/ra99bdvm6fzay3g/demo_test.csv?dl=1
- Utterance Length data: https://www.dropbox.com/s/uxtqqzl18nwxowq/LU_test.csv?dl=1
- Word data: https://www.dropbox.com/s/1ces4hv8kh0stov/token_test.csv?dl=1

```
{r, include = FALSE} #Loading packages library(pacman) p_load(tidyverse, stringr, Metrics,
caret, lme4) setwd("~/Cognitive Science/3. Semester/Experimental Methods 3/Portfolios")
```

Exercise 1) Testing model performance

How did your models from last time perform? In this exercise you have to compare the results on the training data () and on the test data. Report both of them. Compare them. Discuss why they are different.

- recreate the models you chose last time (just write the model code again and apply it to your training data (from the first assignment))

"'{r, include=FALSE} #The models predicting child MLU model_quadratic = lmer(CHI_MLU ~ VISIT + Diagnosis + I(VISITˆ2)+(1+VISIT+ I(VISITˆ2)|SUBJ), autism_data, REML = FALSE)

Buidling a baseline model

intercept_model =lmer(CHI_MLU ~ 1+ (1+VISIT+ I(VISITˆ2)|SUBJ), autism_data, REML=FALSE)# creating the intercept model

Buidling a null model

null_model =lmer(CHI_MLU ~ VISIT + (1+VISIT+ I(VISITˆ2)|SUBJ), autism_data, REML=FALSE)# creating the null model

Comparing the models using anova()

anova(null_model,intercept_model, model_quadratic)

“‘

calculate performance of the model on the training data: root mean square error is a good measure. (Tip: google the function rmse())

```
{r,  include=FALSE} rmse(autism_data$CHI_MLU[!is.na(autism_data$CHI_MLU)], predict(model_quadratic))
```

- create the test dataset (apply the code from assignment 1 part 1 to clean up the 3 test datasets)

“‘{r, include=FALSE}

Loading data

demo_test <- read.csv(“demo_test.csv”) LU_test <- read.csv(“LU_test.csv”) token_test <- read.csv(“token_test.csv”) autism_data <- read.csv((“autism_data.csv”))

colnames(autism_data)[2] <- “SUBJ”

Renaming columns

colnames(demo_test)[1] <- “SUBJ” colnames(demo_test)[2] <- “VISIT”

Extracting numbers out of strings

LU_test$VISIT$ < $-str_extract(LU_test$VISIT, “\d”) token_test$VISIT$ < $-str_extract(token_test$VISIT, “\d”)

removing special characters

LU_test$SUBJ$ < $-str_replace_all(LU_test$SUBJ, pattern = “[[:punct:]]”, replacement =“”) token_test$SUBJ$ < $-str_replace_all(token_test$SUBJ, pattern = “[[:punct:]]”, replacement =“”) demo_test$SUBJ$ < $-str_replace_all(demo_test$SUBJ, pattern = “[[:punct:]]”, replacement =“”)

merging the data

test <- full_join(token_test,LU_test) test$VISIT$ < $-as.numeric(test$VISIT) test <- full_join(test,demo_test)

selecting relevant columns

test <- test %>% as_data_frame %>% select(SUBJ, VISIT, Ethnicity, Diagnosis, Gender, Age, ADOS, MullenRaw, ExpressiveLangRaw, MOT_MLU, MOT_LUstd, CHI_MLU, CHI_LUstd, types_MOT, types_CHI, tokens_MOT, tokens_CHI)

Renaming columns

colnames(test)[8] <- “nonVerbalIQ” colnames(test)[9] <- “verbalIQ”

Creating a subset of data from visit 1

test_subset <- subset(test, VISIT == 1) test_filter <- filter(test, VISIT == “1”)

Renaming columns

colnames(test_subset)[7] <- “ADOS1” colnames(test_subset)[8] <- “nonverbalIQ1” colnames(test_subset)[9] <- “verbalIQ1”

Selecting relevant columns

test_subset <- test_subset %>% as_data_frame %>% select(SUBJ, ADOS1, nonverbalIQ1, verbalIQ1)

Merging subset with final data set

test <- merge(test,test_subset, by = “SUBJ”)

Selecting relevant columns and arranging in correct order

test <- test %>% as_data_frame %>% select(SUBJ, Diagnosis, Gender, Age, Ethnicity, VISIT, ADOS1, nonverbalIQ1, verbalIQ1, MOT_MLU, MOT_LUstd, CHI_MLU, CHI_LUstd, types_MOT, types_CHI, tokens_MOT, tokens_CHI)

Alternative

test <- as.integer(as.factor(test$SUBJ))

Selecting relevant columns and arranging in correct order

test <- test %>% as_data_frame %>% select(SUBJ, Gender, Age, Diagnosis, Ethnicity, VISIT, ADOS1, nonverbalIQ1, verbalIQ1, MOT_MLU, MOT_LUstd, CHI_MLU, CHI_LUstd, types_MOT, types_CHI, tokens_MOT, tokens_CHI)

Gender specification

test <- mutate(test, Gender = ifelse(Gender %in% c(“1”), “M”, “F”))

levels(test$Gender) <- c("F", "M")

Diagonosis specification

test <- mutate(test, Diagnosis = ifelse(Diagnosis %in% c("A"), "ASD", "TD"))

```

- test the performance of the models on the test data (Tips: google the functions "predict()")

```{r, include=FALSE} rmse(test$CHI_MLU[!is.na(test$CHI_MLU)], predict(model_quadratic, test[!is.na(test$CHI_MLU),], allow.new.levels = TRUE)) #RMSE(autism_data$CHI_MLU, predictions)

- optional: predictions are never certain, can you identify the uncertainty of the predictions? (e.g. google predictinterval())

formatting tip: If you write code in this document and plan to hand it in, remember to put include=FALSE in the code chunks before handing in.

REPONSE: The quadratic model (pseudocode: mean length of utterance ~ visit + visit^2 + diagnosis + (VISIT+ VISIT^2|SUBJ)) trained on the train data produces a root mean square error 0.289. The error increases when applying the model on the test set. The mean squared error goes from 0.289 to 0.773, when applying it on a new dataset - the vaidation test set. The model does not do well, when applied to new data, and is thus not able to predict and generalize to the population. This could be due to overfitting of the data in the training set.

#### Exercise 2) Model Selection via Cross-validation (N.B: ChildMLU!)

One way to reduce bad surprises when testing a model on new data is to train the model via cross-validation.

In this exercise you have to use cross-validation to calculate the predictive error of your models and use this predictive error to select the best possible model.

- Use cross-validation to compare your model from last week with the basic model (Child MLU as a function of Time and Diagnosis, and don't forget the random effects!)
- (Tips): google the function "createFolds"; loop through each fold, train both models on the other folds and test them on the fold)

```{r, include = FALSE}

folds <- createFolds(unique(autism_data$SUBJ),10) folds

rmse_list=data.frame()

# simple model

n=1 rmsetrain = NULL rmsetest = NULL

for (f in folds){ train_data <- subset(autism_data, !(SUBJ %in% f)) test_data <- subset(autism_data, SUBJ %in% f) model_quadratic = lmer(CHI_MLU ~ VISIT + Diagnosis + I(VISIT^2)+(1+VISIT+ I(VISIT^2)|SUBJ), data = train_data, REML = FALSE) rmsetrain[n] = rmse(train_data$CHI_MLU[!is.na(train_data$CHI_MLU)], predict(model_quadratic)) rmsetest[n] = rmse(test_data$CHI_MLU[!is.na(test_data$CHI_MLU)], predict(model_quadratic, test_data[!is.na(test_data$CHI_MLU),], allow.new.levels = TRUE)) n = n +1 }

rmse <- data.frame(rmsetest, rmsetrain) max(rmse$rmsetrain) - min(rmse$rmsetrain) max(rmse$rmsetest) - min(rmse$rmsetest) mean(rmse$rmsetest)
```

model 11 (fra sidste portfolio)

n=1 rmsetrain_1 = NULL rmsetest_1 = NULL

for (f in folds){ train_data <- subset(autism_data, !(SUBJ %in% f)) test_data <- subset(autism_data, SUBJ %in% f) model11 = lmer(CHI_MLU ~ VISIT + Diagnosis + ADOS1 + verbalIQ1 + nonverbalIQ1 + I(VISITˆ2)+(1+VISIT+ I(VISITˆ2)|SUBJ)+ (1|Gender)+ (1|Ethnicity), data = train_data, REML = FALSE) rmsetrain_1[n] = rmse(train_data$CHI_MLU[!is.na(train_data$CHI_MLU)], predict(model11)) rmsetest_1[n] = rmse(test_data$CHI_MLU[!is.na(test_data$CHI_MLU)], predict(model11, test_data[!is.na(test_data$CHI_MLU),], allow.new.levels = TRUE)) n = n +1 } rmse1 <- data.frame(rmsetest_1, rmsetrain_1) max(rmse1$rmsetrain_1) − min(rmse1rmsetrain_1) max(rmse1$rmsetest_1) − min(rmse1rmsetest_1) mean(rmse1$rmsetest_1)

model 2

n=1 rmsetrain_2 = NULL rmsetest_2 = NULL

for (f in folds){ train_data <- subset(autism_data, !(SUBJ %in% f)) test_data <- subset(autism_data, SUBJ %in% f) model2 = lmer(CHI_MLU ~ VISIT + Diagnosis + verbalIQ1 + nonverbalIQ1 + I(VISITˆ2)+(1+VISIT+ I(VISITˆ2)|SUBJ)+ (1|Gender)+ (1|Ethnicity), data = train_data, REML = FALSE) rmsetrain_2[n] = rmse(train_data$CHI_MLU[!is.na(train_data$CHI_MLU)], predict(model2)) rmsetest_2[n] = rmse(test_data$CHI_MLU[!is.na(test_data$CHI_MLU)], predict(model2, test_data[!is.na(test_data$CHI_MLU),], allow.new.levels = TRUE)) n = n +1 } rmse2 <- data.frame(rmsetest_2, rmsetrain_2) max(rmse2$rmsetrain_2) − min(rmse2rmsetrain_2) max(rmse2$rmsetest_2) − min(rmse2rmsetest_2)

mean(rmse2$rmsetest_2)

model 3

n=1 rmsetrain_3 = NULL rmsetest_3 = NULL

for (f in folds){ train_data <- subset(autism_data, !(SUBJ %in% f)) test_data <- subset(autism_data, SUBJ %in% f) model3 = lmer(CHI_MLU ~ VISIT + Diagnosis + verbalIQ1 + I(VISITˆ2)+(1+VISIT+ I(VISITˆ2)|SUBJ)+ (1|Gender)+ (1|Ethnicity), data = train_data, REML = FALSE) rmsetrain_3[n] = rmse(train_data$CHI_MLU[!is.na(train_data$CHI_MLU)], predict(model3)) rmsetest_3[n] = rmse(test_data$CHI_MLU[!is.na(test_data$CHI_MLU)], predict(model3, test_data[!is.na(test_data$CHI_MLU),], allow.new.levels = TRUE)) n = n +1 } rmse3 <- data.frame(rmsetest_3, rmsetrain_3) max(rmse3$rmsetrain_3) − min(rmse3rmsetrain_3) max(rmse3$rmsetest_3) − min(rmse3rmsetest_3) mean(rmse3$rmsetest_3)

model 4

n=1 rmsetrain_4 = NULL rmsetest_4 = NULL

for (f in folds){ train_data <- subset(autism_data, !(SUBJ %in% f)) test_data <- subset(autism_data, SUBJ %in% f) model4 = lmer(CHI_MLU ~ VISIT + Diagnosis + ADOS1 + verbalIQ1 + I(VISITˆ2)+(1+VISIT+ I(VISITˆ2)|SUBJ)+ (1|Gender)+ (1|Ethnicity), data = train_data, REML = FALSE) rmsetrain_4[n] = rmse(train_data$CHI_MLU[!is.na(train_data$CHI_MLU)], predict(model4)) rmsetest_4[n] = rmse(test_data$CHI_MLU[!is.na(test_data$CHI_MLU)], predict(model4, test_data[!is.na(test_data$CHI_MLU),], allow.new.levels = TRUE)) n = n +1 } rmse4 <- data.frame(rmsetest_4, rmsetrain_4) max(rmse4$rmsetrain_4) − min(rmse4rmsetrain_4) max(rmse4$rmsetest_4) − min(rmse4rmsetest_4) mean(rmse4$rmsetest_4)

model 5

n=1 rmsetrain_5 = NULL rmsetest_5 = NULL

for (f in folds){ train_data <- subset(autism_data, !(SUBJ %in% f)) test_data <- subset(autism_data, SUBJ %in% f) model5 = lmer(CHI_MLU ~ VISIT + Diagnosis + nonverbalIQ1 + I(VISITˆ2)+(1+VISIT+ I(VISITˆ2)|SUBJ)+ (1|Gender)+ (1|Ethnicity), data = train_data, REML = FALSE) rmse-train_5[n] = rmse(train_data$CHI_MLU[!is.na(train_data$CHI_MLU)], predict(model5)) rmsetest_5[n] = rmse(test_data$CHI_MLU[!is.na(test_data$CHI_MLU)], predict(model5, test_data[!is.na(test_data$CHI_MLU),], allow.new.levels = TRUE)) n = n +1 } rmse5 <- data.frame(rmsetest_5, rmsetrain_5) max(rmse5$rmsetrain_5)− min(rmse5rmsetrain_5) max(rmse5$rmsetest_5) − min(rmse5rmsetest_5) mean(rmse5$rmsetest_5)

model Signe

n=1 rmsetrain_signe = NULL rmsetest_signe = NULL

for (f in folds){ train_data <- subset(autism_data, !(SUBJ %in% f)) test_data <- subset(autism_data, SUBJ %in% f) model_signe = lmer(CHI_MLU ~ VISIT + Diagnosis + MOT_MLU +ADOS1 + types_CHI + tokens_CHI + I(VISITˆ2)+(1+VISIT+ I(VISITˆ2)|SUBJ)+ (1|Gender), train_data, REML = FALSE) rmsetrain_signe[n] = rmse(train_data$CHI_MLU[!is.na(train_data$CHI_MLU)], predict(model_signe)) rmsetest_signe[n] = rmse(test_data$CHI_MLU[!is.na(test_data$CHI_MLU)], predict(model_signe, test_data[!is.na(test_data$CHI_MLU),], allow.new.levels = TRUE)) n = n +1 }

rmse_signe <- data.frame(rmsetest_signe, rmsetrain_signe) mean(rmse_signe$rmsetrain_signe)mean(rmse_signe$rmsetest_sign

‘‘‘

Which model is better at predicting new data: your model from last week or the basic model (Child MLU as a function of Time and Diagnosis, and don't forget the random effects!)?

- Test both of them on the test data. “‘{r, include = FALSE} #model 3 tested on real test data rmse(test$CHI_MLU[!is.na(test$CHI_MLU)], predict(model3, test[!is.na(test$CHI_MLU),], allow.new.levels = TRUE))

model 11

model11 = lmer(CHI_MLU ~ VISIT + Diagnosis + ADOS1 + verbalIQ1 + nonverbalIQ1 + I(VISITˆ2)+(1+VISIT+ I(VISITˆ2)|SUBJ)+ (1|Gender)+ (1|Ethnicity), data = train_data, REML = FALSE)

rmse(test$CHI_MLU[!is.na(test$CHI_MLU)], predict(model11, test[!is.na(test$CHI_MLU),], allow.new.levels = TRUE))

RMSE(autism_data$CHI_MLU, predictions)

‘‘‘

- Report the results and comment on them.

The quadratic model (pseudocode: mean length of utterance ~ visit + visitˆ2 + diagnosis + (VISIT+ VISITˆ2|SUBJ)) produces a mean squared error of 0.773, when applied to the test data compared to 0.658 produced by the best explaining model from last time (mean length of utterance ~ Diagnosis + visit+ visitˆ2

+ ADOS + verbal IQ + nonverbal IQ + (VISIT+ I(VISIT^2)|SUBJ)).Thus, the best model is still the last mentioned, which is able to predict the new data the best.

- Now try to find the best possible predictive model of ChildMLU, that is, the one that produces the best cross-validated results.

Based on the mean of the mean squared errors, when applied to the cross-validated test data, the best predictive model is mmean length of utterance ~ Diagnosis + verbalIQ + visit + visit^2+(1+VISIT+ VISIT^2|SUBJ)+ (1|Gender)+ (1|Ethnicity)

- Which model is better at predicting new data: the one you selected last week or the one chosen via cross-validation this week?

Based on the mean squared errors for the model applied to the 'true' test data, the best predictive model is the one found the one found by cross-vaildation (rmse = 0.64) compared to the best explainging model from last week (rmse = 0.66).

- Bonus Question 1: What is the effect of changing the number of folds? Can you plot RMSE as a function of number of folds?
- Bonus Question 2: compare the cross-validated predictive error against the actual predictive error on the test data

Exercise 3) Assessing the single child

Let's get to business. This new kiddo - Bernie - has entered your clinic. This child has to be assessed according to his group's average and his expected development.

Bernie is one of the six kids in the test dataset, so make sure to extract that child alone for the following analysis.

```
{r, include = FALSE} bernie <- filter(test, SUBJ == "Bernie")
```

You want to evaluate:

- how does the child fare in ChildMLU compared to the average TD child at each visit? Define the distance in terms of absolute difference between this Child and the average TD. (Tip: recreate the equation of the model: Y=Intercept+BetaX1+BetaX2, etc; input the average of the TD group for each parameter in the model as X1, X2, etc.).

"'{r, include = FALSE} summary(model_signe)

TD1 = -0.0917431 + 0.0337088*1 + 0.0023032+ 0.2705685 +0.0049511 + 0.0077829 +0.0001158+ (-0.00030861)* TD1

TD2 = -0.0917431 + 0.0337088*2 + 0.0023032+ 0.2705685 +0.0049511 + 0.0077829 +0.0001158+ (-0.00030864)* TD2

TD3 = -0.0917431 + 0.0337088*3 + 0.0023032+ 0.2705685 +0.0049511 + 0.0077829 +0.0001158+ (-0.00030869)* TD3

TD4 = -0.0917431 + 0.0337088*4 + 0.0023032+ 0.2705685 +0.0049511 + 0.0077829 +0.0001158+ (-0.000308616)* TD4

TD5 = -0.0917431 + 0.0337088*5 + 0.0023032+ 0.2705685 +0.0049511 + 0.0077829 +0.0001158+ (-0.0003086225)* TD5

TD6 = -0.0917431 + 0.0337088*6 + 0.0023032+ 0.2705685 +0.0049511 + 0.0077829 +0.0001158+ (-0.0003086236)* TD6

TD2 = 0.722199 + 0.116477 + 0.541572*2 + -0.044256*4 TD2

TD3 = 0.722199 + 0.116477 + 0.541572*3 + -0.044256*9 TD3

TD4 = 0.722199 + 0.116477 + 0.541572*4* + *-0.04425616* TD4

TD5 = 0.722199 + 0.116477 + 0.541572*5* + *-0.04425625* TD5

TD6 = 0.722199 + 0.116477 + 0.541572*6* + *-0.04425636*

{r, echo = TRUE} data.frame(TD1, TD2, TD3, TD4, TD5, TD6) bernie$CHI_MLU "'

Based on the quadratic model the typically developed child starts of with a mean length of utterance on 1.34, whereas Bernie starts 2.54. Thus, Bernie makes longer utterances when entering the experiment. At the 6th visit Bernie has a mean length of utterance of 3.17, whereas a typically developed child has a mean length of utterance of 2.49. Thus, Bernie is still superior, however the increase is smaller for Bernie compared to typically developed children (0.75 (TD) vs. 0.63 (Bernie))

- how does the child fare compared to the model predictions at Visit 6? Is the child below or above expectations? (tip: use the predict() function on Bernie's data only and compare the prediction with the actual performance of the child) #Is this correct?

  `predict(model_quadratic, bernie, allow.new.levels = TRUE)`

  Based on the best predictive model found by cross-validation, Bernie is estimated to having a mean length of utterance of 2.99 at visit. In reality he has a mean length of utterance 3.17, and thus, the model has underestimated him and predicts he has a slower development than he in reality has.

OPTIONAL: Exercise 4) Model Selection via Information Criteria

Another way to reduce the bad surprises when testing a model on new data is to pay close attention to the relative information criteria between the models you are comparing. Let's learn how to do that!

Re-create a selection of possible models explaining ChildMLU (the ones you tested for exercise 2, but now trained on the full dataset and not cross-validated).

"'{r, include = FALSE}

simple model

model_quadratic = lmer(CHI_MLU ~ VISIT + Diagnosis + I(VISITˆ2)+(1+VISIT+ I(VISITˆ2)|SUBJ), data = autism_data, REML = FALSE)

rmsetrain[n] = rmse(autism_data$CHI_MLU[!is.na(autism_data$CHI_MLU)], predict(model_quadratic)) rmsetest[n] = rmse(test$CHI_MLU[!is.na(test$CHI_MLU)], predict(model_quadratic, test[!is.na(test$CHI_MLU),], allow.new.levels = TRUE))

data.frame(rmsetest, rmsetrain)

model 11 (fra sidste portfolio)

model_11 = lmer(CHI_MLU ~ VISIT + Diagnosis + ADOS1 + verbalIQ1 + nonverbalIQ1 + I(VISITˆ2)+(1+VISIT+ I(VISITˆ2)|SUBJ)+ (1|Gender)+ (1|Ethnicity), data = train_data, REML = FALSE)

rmsetrain[n] = rmse(autism_data$CHI_MLU[!is.na(autism_data$CHI_MLU)], predict(model_11)) rmsetest[n] = rmse(test$CHI_MLU[!is.na(test$CHI_MLU)], predict(model_11, test[!is.na(test$CHI_MLU),], allow.new.levels = TRUE))

data.frame(rmsetest, rmsetrain)

model 2

model2 = lmer(CHI_MLU ~ VISIT + Diagnosis + verbalIQ1 + nonverbalIQ1 + I(VISITˆ2)+(1+VISIT+ I(VISITˆ2)|SUBJ)+ (1|Gender)+ (1|Ethnicity), data = train_data, REML = FALSE)

rmsetrain[n] = rmse(autism_data$CHI_MLU[!is.na(autism_data$CHI_MLU)], predict(model_2)) rmsetest[n] = rmse(test$CHI_MLU[!is.na(test$CHI_MLU)], predict(model_2, test[!is.na(test$CHI_MLU),], allow.new.levels = TRUE))

data.frame(rmsetest, rmsetrain)

model 3

model3 = lmer(CHI_MLU ~ VISIT + Diagnosis + verbalIQ1 + I(VISITˆ2)+(1+VISIT+ I(VISITˆ2)|SUBJ)+ (1|Gender)+ (1|Ethnicity), data = train_data, REML = FALSE) rmsetrain[n] = rmse(autism_data$CHI_MLU[!is.na(autism_data$CHI_MLU)], predict(model_3)) rmsetest[n] = rmse(test$CHI_MLU[!is.na(test$CHI_MLU)], predict(model_3, test[!is.na(test$CHI_MLU),], allow.new.levels = TRUE))

data.frame(rmsetest, rmsetrain)

model 4

model4 = lmer(CHI_MLU ~ VISIT + Diagnosis + ADOS1 + verbalIQ1 + I(VISITˆ2)+(1+VISIT+ I(VISITˆ2)|SUBJ)+ (1|Gender)+ (1|Ethnicity), data = train_data, REML = FALSE)

rmsetrain[n] = rmse(autism_data$CHI_MLU[!is.na(autism_data$CHI_MLU)], predict(model_4)) rmsetest[n] = rmse(test$CHI_MLU[!is.na(test$CHI_MLU)], predict(model_4, test[!is.na(test$CHI_MLU),], allow.new.levels = TRUE))

data.frame(rmsetest, rmsetrain)

model 5

model5 = lmer(CHI_MLU ~ VISIT + Diagnosis + nonverbalIQ1 + I(VISITˆ2)+(1+VISIT+ I(VISITˆ2)|SUBJ)+ (1|Gender)+ (1|Ethnicity), data = train_data, REML = FALSE)

rmsetrain[n] = rmse(autism_data$CHI_MLU[!is.na(autism_data$CHI_MLU)], predict(model_5)) rmsetest[n] = rmse(test$CHI_MLU[!is.na(test$CHI_MLU)], predict(model_5, test[!is.na(test$CHI_MLU),], allow.new.levels = TRUE))

data.frame(rmsetest, rmsetrain)

“‘

Then try to find the best possible predictive model of ChildMLU, that is, the one that produces the lowest information criterion.

- Bonus question for the optional exercise: are information criteria correlated with cross-validated RMSE? That is, if you take AIC for Model 1, Model 2 and Model 3, do they co-vary with their cross-validated RMSE?

OPTIONAL: Exercise 5): Using Lasso for model selection

Welcome to the last secret exercise. If you have already solved the previous exercises, and still there's not enough for you, you can expand your expertise by learning about penalizations. Check out this tutorial: http://machinelearningmastery.com/penalized-regression-in-r/ and make sure to google what penalization is, with a focus on L1 and L2-norms. Then try them on your data!