Problem Set 3: Predicting bullying

Group Theta

2025-04-21

An Overview

In this assignment, we built the **random forest** regression and classifier to predict the students at risk for bullying. Specifically, we divided the tasks into three parts:

1. Data Cleaning and Feature Engineering

The pipeline begins with a custom function that processes the raw student survey data. Using the survey's codebook, I grouped related questions and transformed them into summary variables. For example, questions about school safety (e.g., `psafe` and `esafe` items) were averaged into a single `safe_mean` score, while binary indicators such as the "feel safer" questions were summed into variables like `feel_safer_sum`. This not only reduced dimensionality from **195** to around **45** variables but also made the data more interpretable for modeling. All character variables were converted to factors, and missing data was handled using multiple imputation through the mice package. Training and test data were combined during imputation to maintain consistent structure, and the final datasets were re-separated and written out for modeling.

2. Regression Task : Predicting Bully Scores

For predicting continuous bullying levels, I trained a random forest regression model using 10-fold cross-validation. The caret package was used to tune the model, and the best performance was achieved at `mtry = 47` with an RMSE of **0.404**, which reflects realistic generalization error. A much lower RMSE of **0.172** was found when predicting on the training data directly, showing expected overfitting on seen data. To understand which variables were driving predictions, I extracted the top 15 most important features. Variables like `safe_mean`, `disc_mean`, and `belong_mean` appeared at the top, showing that students' feelings of safety, experiences with discrimination, and sense of belonging were the strongest predictors of bullying scores.

3. Classification Task : Predicting High Bully Risk

I then created a classification task by labeling students as `bully_high` if their score was **2.5** or above. A second random forest model was trained to classify this binary target, again using cross-validation and optimizing for ROC AUC. I evaluated the model using ROC curves and calculated sensitivity and specificity across thresholds. The chosen threshold of **0.5** provided a balance between

catching most true positives and avoiding too many false alarms. A confusion matrix showed the classifier's accuracy, false positives, and false negatives. The final test set predictions included each student's bullying score, a high-risk label, and their predicted risk probability. This allowed for both continuous risk tracking and clear flags for intervention.

Below are the codes for each section, the write-ups and discussion questions are at the end.

1. Data Cleaning

```
#load the data
train <- read.csv("/Users/le/Desktop/pset3_predict_bully/data/student_survey_data.csv")
test <- read.csv("/Users/le/Desktop/pset3_predict_bully/data/student_test_data.csv")


# Define a cleaning + feature engineering function
process_survey_data <- function(df, is_train = TRUE) {

    # Remove rows with missing 'bully' ONLY for training
    if (is_train) {
        df <- df %>% filter(!is.na(bully))
    }

    # 3. Convert characters to factors
    df <- df %>%
        mutate(across(where(is.character), as.factor))

    # Aggregate similar questions together, if it is a likert scale, then find the avg, if it
    # Q12
    Q12_safe_mean <- df %>%
        select(student_id, psafe1:psafe7, esafe1:esafe7) %>%
        pivot_longer(-student_id) %>%
        group_by(student_id) %>%
        summarise(safe_mean = mean(value, na.rm = TRUE))

    df <- df %>% left_join(Q12_safe_mean, by = "student_id") %>%
        select(-c(psafe1:psafe7, esafe1:esafe7))


    # Q13
    Q13_feel_safer_sum <- df %>%
        select(student_id, feel_safer_clear : feel_safer_training) %>%
        pivot_longer(-c(student_id, feel_safer_text)) %>%
        group_by(student_id) %>%
        summarise(feel_safer_sum = sum(value))
    df <- df %>%
        select(-starts_with("feel_safer_")) %>%
        left_join(Q13_feel_safer_sum, by = "student_id")
```

```r
# Q16
Q16_disc_mean <- df %>%
    select(student_id, disc_race:disc_country) %>%
    pivot_longer(-student_id) %>%
    group_by(student_id) %>%
    summarise(disc_mean = mean(value, na.rm = TRUE))

df <- df %>%
    left_join(Q16_disc_mean, by = "student_id") %>%
    select(-c(disc_race:disc_country))

# Q17: support
Q17_support_mean <- df %>%
    select(student_id, support1:support8) %>%
    pivot_longer(-student_id) %>%
    group_by(student_id) %>%
    summarise(support_mean = mean(value, na.rm = TRUE))

df <- df %>%
    select(-c(support1:support8)) %>%
    left_join(Q17_support_mean, by = "student_id")

# Q18: belong
Q18_belong_mean <- df %>%
    select(student_id, belong1:belong11) %>%
    pivot_longer(-student_id) %>%
    group_by(student_id) %>%
    summarise(belong_mean = mean(value, na.rm = TRUE))

df <- df %>%
    select(-c(belong1:belong11)) %>%
    left_join(Q18_belong_mean, by = "student_id")

# Q21: rules
Q21_rules_mean <- df %>%
    select(student_id, rules1:rules9) %>%
    pivot_longer(-student_id) %>%
    group_by(student_id) %>%
    summarise(rules_mean = mean(value, na.rm = TRUE))

df <- df %>%
    select(-c(rules1:rules9)) %>%
    left_join(Q21_rules_mean, by = "student_id")

# Q22: sm_
Q22_sm_sum <- df %>%
    select(student_id, sm_facebook:sm_none) %>%
```

```
    pivot_longer(-c(student_id, sm_text)) %>%
    group_by(student_id) %>%
    summarise(sm_sum = sum(value, na.rm = TRUE))

df <- df %>%
    select(-c(sm_facebook:sm_none)) %>%
    left_join(Q22_sm_sum, by = "student_id")

# Q26: talk
Q26_talk_mean <- df %>%
    select(student_id, talk_appropriate:talk_connect) %>%
    pivot_longer(-student_id) %>%
    group_by(student_id) %>%
    summarise(talk_mean = mean(value, na.rm = TRUE))

df <- df %>%
    select(-c(talk_appropriate:talk_connect)) %>%
    left_join(Q26_talk_mean, by = "student_id")

# Q32: use_sm
Q32_use_sm_sum <- df %>%
    select(student_id, use_sm_close_friends:use_sm_give_help) %>%
    pivot_longer(-c(student_id, use_sm_text)) %>%
    group_by(student_id) %>%
    summarise(use_sm_sum = sum(value, na.rm = TRUE))

df <- df %>%
    select(-c(use_sm_close_friends:use_sm_give_help)) %>%
    left_join(Q32_use_sm_sum, by = "student_id")

# Q33: sm_help
Q33_sm_help_mean <- df %>%
    select(student_id, sm_help_connect:sm_help_reach_out) %>%
    pivot_longer(-student_id) %>%
    group_by(student_id) %>%
    summarise(sm_help_mean = mean(value, na.rm = TRUE))

df <- df %>%
    select(-c(sm_help_connect:sm_help_reach_out)) %>%
    left_join(Q33_sm_help_mean, by = "student_id")

# Q34: sm_concern
Q34_sm_concern_mean <- df %>%
    select(student_id, sm_concern_left_out:sm_concern_stalk) %>%
    pivot_longer(-student_id) %>%
    group_by(student_id) %>%
    summarise(sm_concern_mean = mean(value, na.rm = TRUE))
```

```r
df <- df %>%
    select(-c(sm_concern_left_out:sm_concern_stalk)) %>%
    left_join(Q34_sm_concern_mean, by = "student_id")

# Q35: sm_ever
Q35_sm_ever_sum <- df %>%
    select(student_id, sm_ever_made_mean:sm_ever_stalked) %>%
    pivot_longer(-student_id) %>%
    group_by(student_id) %>%
    summarise(sm_ever_sum = sum(value, na.rm = TRUE))

df <- df %>%
    select(-c(sm_ever_made_mean:sm_ever_stalked)) %>%
    left_join(Q35_sm_ever_sum, by = "student_id")

# Q36: sm_relation
Q36_sm_relation_mean <- df %>%
    select(student_id, sm_less_real:sm_more_open) %>%
    pivot_longer(-student_id) %>%
    group_by(student_id) %>%
    summarise(sm_relation_mean = mean(value, na.rm = TRUE))

df <- df %>%
    select(-c(sm_less_real:sm_more_open)) %>%
    left_join(Q36_sm_relation_mean, by = "student_id")

# Q39: school
Q39_school_mean <- df %>%
    select(student_id, school_appropriate:school_connect) %>%
    pivot_longer(-student_id) %>%
    group_by(student_id) %>%
    summarise(school_mean = mean(value, na.rm = TRUE))

df <- df %>%
    select(-c(school_appropriate:school_connect)) %>%
    left_join(Q39_school_mean, by = "student_id")

# Q40: tell
Q40_tell_sum <- df %>%
    select(student_id, tell_friend:tell_text) %>%
    pivot_longer(-student_id) %>%
    group_by(student_id) %>%
    summarise(tell_sum = sum(value, na.rm = TRUE))

df <- df %>%
    select(-c(tell_friend:tell_text)) %>%
    left_join(Q40_tell_sum, by = "student_id")
```

```r
    # Remove *_text columns
    df <- df %>% select(-ends_with("_text"))

    return(df)
}


test_cleaned <- process_survey_data(test, is_train = FALSE)
train_cleaned <- process_survey_data(train, is_train = TRUE)



# Convert sm_age to numeric
train_cleaned$sm_age <- as.numeric(str_extract(train_cleaned$sm_age, "\\d+\\.?\\d*"))
test_cleaned$sm_age  <- as.numeric(str_extract(test_cleaned$sm_age, "\\d+\\.?\\d*"))
summary(train_cleaned$sm_age)
summary(test_cleaned$sm_age)




# Save and remove bully label from training data
bully <- train_cleaned$bully

train_cleaned <- train_cleaned %>% select(-bully)
train_cleaned$source <- "train"
test_cleaned$source <- "test"


# Combine
combined <- bind_rows(train_cleaned, test_cleaned)

#Imputation: no need to run, already saved

# # Initialize mice to get the method template
# ini <- mice(combined, maxit = 0)
# meth <- ini$method
#
#
# #set method for school_value_red
# meth["school_values_red"] <- "logreg"
# meth
#
# imp <- mice(combined, method = meth, m = 5, seed = 666)
#
# # Extract the completed data
# combined_imputed <- complete(imp, 1)
```

```r
#check missingness
# map_int(combined_imputed, ~sum(is.na(.x)))

# #remove the one missing variable
# combined_imputed <- combined_imputed %>% select(-school_values_red)
#
# write.csv(combined_imputed, "/Users/le/Desktop/pset3_predict_bully/data/combined_imputed.csv

combined_imputed <- read.csv("/Users/le/Desktop/pset3_predict_bully/data/combined_imputed.csv")

# Re-split
train_imputed <- combined_imputed %>% filter(source == "train") %>% select(-source)
test_imputed <- combined_imputed %>% filter(source == "test") %>% select(-source)

#put bully back
train_imputed$bully <- bully


#check for missingness in imputed data (no missing)
train_imputed%>%
    summarise(across(everything(), ~ sum(is.na(.)))) %>%
    pivot_longer(
        everything(),
        names_to = "variable",
        values_to = "missing_count"
    ) %>%
    arrange(desc(missing_count)) %>%
    print(n = Inf)

test_imputed%>%
    summarise(across(everything(), ~ sum(is.na(.)))) %>%
    pivot_longer(
        everything(),
        names_to = "variable",
        values_to = "missing_count"
    ) %>%
    arrange(desc(missing_count)) %>%
    print(n = Inf)


#save the imputed data for later use
# write.csv(train_imputed, "/Users/le/Desktop/pset3_predict_bully/data/train_imputed.csv", row
# write.csv(test_imputed, "/Users/le/Desktop/pset3_predict_bully/data/test_imputed.csv", row.n
```

```r
#calculate the percentage of bully >= 2.5
train_imputed %>%
    summarise(bully = mean(bully >= 2.5)) %>%
    mutate(percentage = bully * 100)
train_imputed %>% filter(bully >= 2.5) %>% count()
summary(train_imputed$bully)
```

2. Random Forest Regression

```
#load the imputed data
train_imputed <- read.csv("/Users/le/Desktop/pset3_predict_bully/data/train_imputed.csv")
test_imputed <- read.csv("/Users/le/Desktop/pset3_predict_bully/data/test_imputed.csv")

#remove student id
student_id_train <- train_imputed$student_id

train_imputed <- train_imputed %>% select(-student_id)



#random forest
#
# #1. regression
# #TRAIN
# set.seed(23123)
#
# ctrl <- trainControl(
#     method = "cv",
#     number = 10,
#     verboseIter = FALSE
# )
#
# rf_bully_reg <- train(
#     bully ~ .,
#     data = train_imputed,
#     method = "rf",
#     ntree = 100,
#     tuneLength = 5,
#     trControl = ctrl,
# )
#
#
#
# saveRDS(rf_bully_reg, file = "/Users/le/Desktop/pset3_predict_bully/data/rf_bully_reg.rds")


rf_bully_reg = readRDS("/Users/le/Desktop/pset3_predict_bully/data/rf_bully_reg.rds")


summary(rf_bully_reg)
```

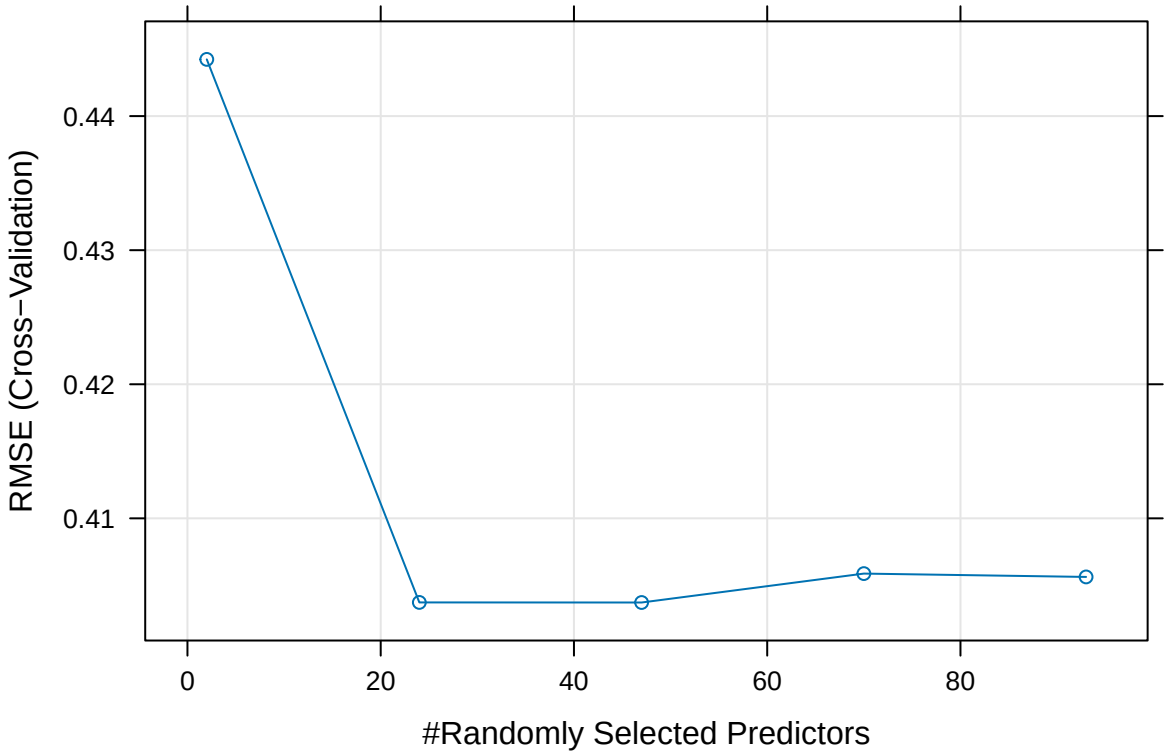
```
##                 Length Class      Mode
## call               5   -none-     call
## type               1   -none-     character
```

```
## predicted       7745   -none-     numeric
## mse              100   -none-     numeric
## rsq              100   -none-     numeric
## oob.times       7745   -none-     numeric
## importance        93   -none-     numeric
## importanceSD       0   -none-     NULL
## localImportance    0   -none-     NULL
## proximity          0   -none-     NULL
## ntree              1   -none-     numeric
## mtry               1   -none-     numeric
## forest            11   -none-     list
## coefs              0   -none-     NULL
## y               7745   -none-     numeric
## test               0   -none-     NULL
## inbag              0   -none-     NULL
## xNames            93   -none-     character
## problemType        1   -none-     character
## tuneValue          1   data.frame list
## obsLevels          1   -none-     logical
## param              1   -none-     list
```

```
rf_bully_reg$results
```

```
##   mtry      RMSE  Rsquared       MAE     RMSESD RsquaredSD       MAESD
## 1    2 0.4442311 0.3254797 0.3164568 0.01536735 0.02444093 0.006427885
## 2   24 0.4037290 0.3790138 0.2827091 0.01239122 0.03662380 0.004287604
## 3   47 0.4037242 0.3762482 0.2810594 0.01383786 0.03882767 0.004970418
## 4   70 0.4058790 0.3690354 0.2830887 0.01358646 0.04050166 0.004978766
## 5   93 0.4056306 0.3701797 0.2817759 0.01455655 0.03912917 0.006190498
```

```
plot(rf_bully_reg)
```



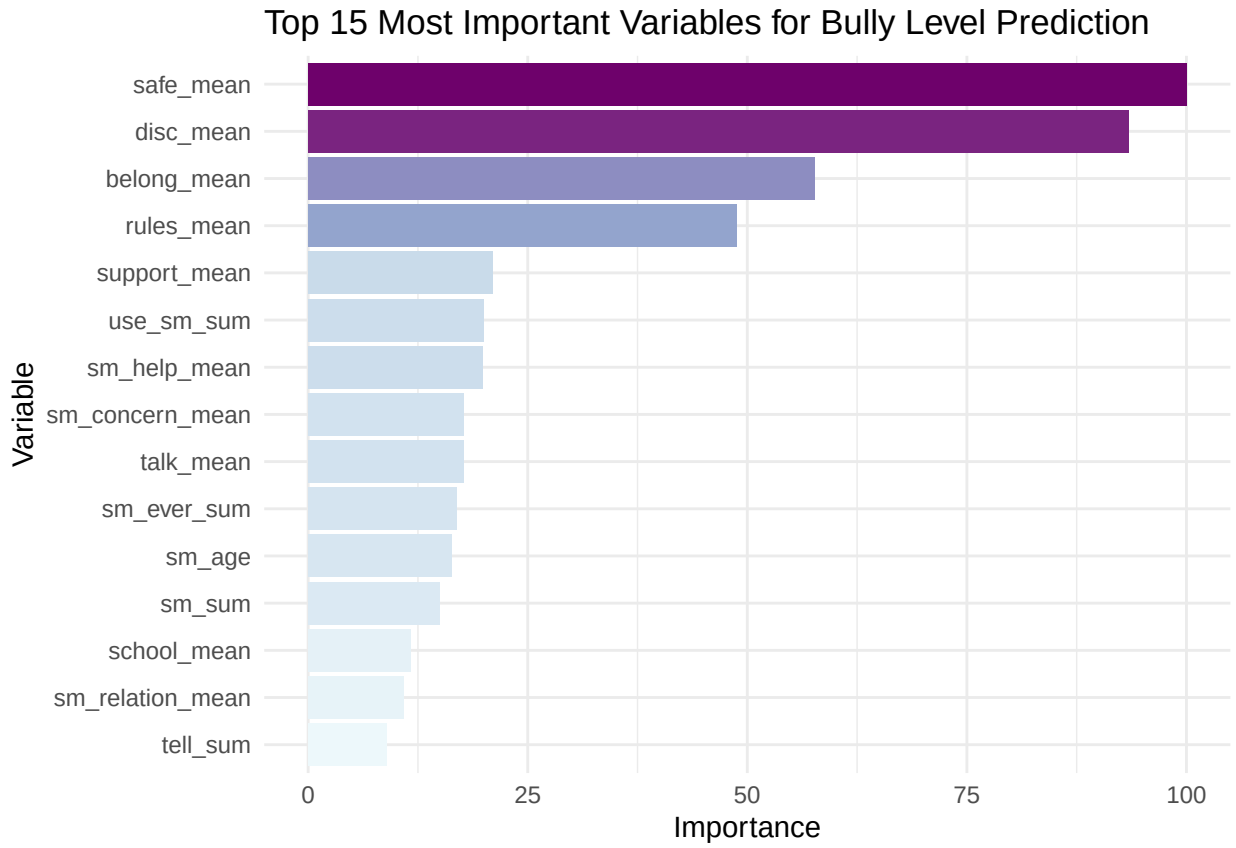
The best performance was achieved at mtry = 47 with an RMSE of 0.404

```
#find the lowest train rmse, we know that mtry = 47, gives a minimum rmse of 0.404
rf_bully_reg$results %>%
    filter(RMSE == min(RMSE)) %>%
    select(mtry, RMSE)
```

```
##   mtry      RMSE
## 1   47 0.4037242
```

```
#retrieve importance
reg_imp <- caret::varImp(rf_bully_reg)
reg_imp <- reg_imp$importance
reg_imp <- reg_imp %>%
    rownames_to_column("feature") %>%
    arrange(desc(Overall)) %>%
    slice_head(n = 15)


library(RColorBrewer)

ggplot(reg_imp, aes(x = Overall, y = reorder(feature, Overall), fill = Overall)) +
```

```
    geom_col() +
    scale_fill_distiller(palette = "BuPu", direction = 1) +
    labs(
        title = "Top 15 Most Important Variables for Bully Level Prediction",
        x = "Importance",
        y = "Variable"
    ) +
    theme_minimal() +
    theme(legend.position = "none")
```



By plotting the top 15 important variables, we know that questions begin with `safe`, `disc`, `belong` and `rules` are playing crucial rules in our predictions. We should look into these questions. The code below shows TRAIN rmse is 0.172, potentially indicating overfits.

```
str(train_imputed$sm_age)
```

```
##  num [1:7745] 9 12 9 10 10 15 13 13 13 12 ...
```

```
summary(train_imputed$sm_age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   10.00   11.00   10.83   13.00  112.00
```

```r
train_imputed <- train_imputed %>% mutate(rf_pred = predict(rf_bully_reg, newdata = train_imput
RMSE(train_imputed$rf_pred, train_imputed$bully) #rmse = 0.172
```

```
## [1] 0.1718539
```

```r
#TEST rmse, predict bully level

test_imputed <- test_imputed %>%
    mutate(predicted_bully_level = predict(rf_bully_reg, newdata = test_imputed))

#For test rmse calculations:
#RMSE(test_imputed$predicted_bully_level, test_imputed$bully)

# #save the prediction into csv
# write.csv(test_imputed, "/Users/le/Desktop/pset3_predict_bully/data/test_bully.csv", row.nam
```

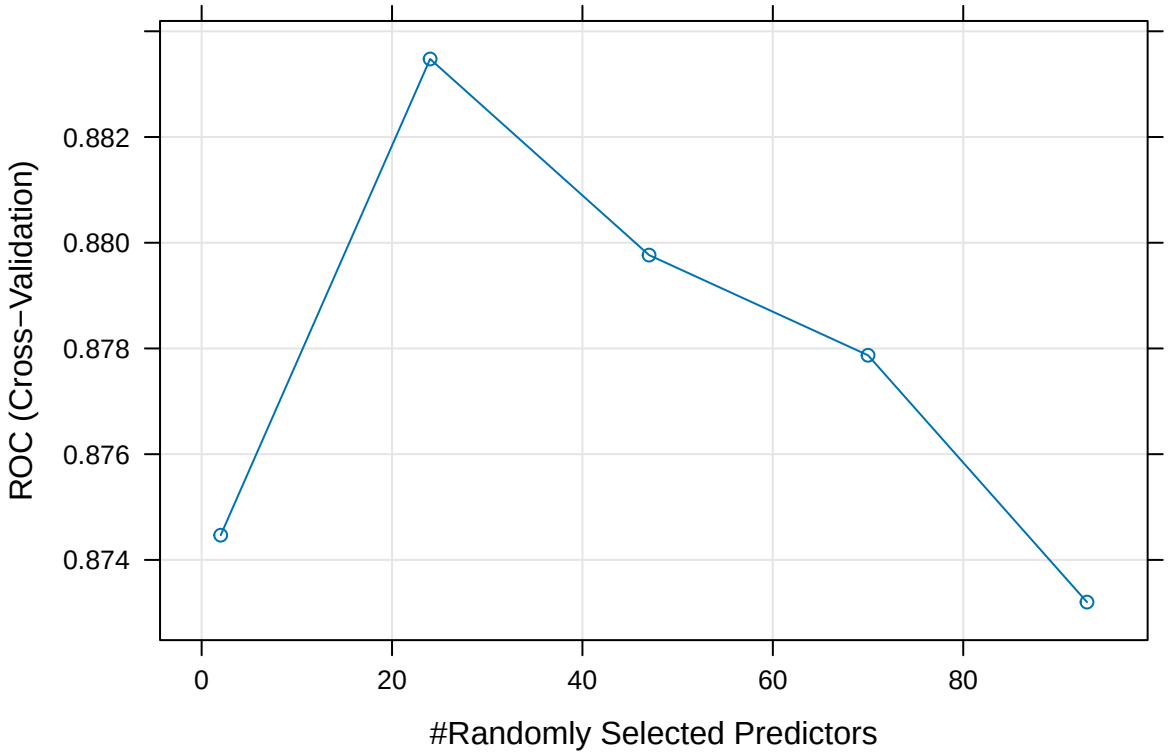
3. Random Forest Classification

```
train_imputed <- read.csv("/Users/le/Desktop/pset3_predict_bully/data/train_imputed.csv")
test_imputed <- read.csv("/Users/le/Desktop/pset3_predict_bully/data/test_bully.csv")
#note: here the test imputed we used another test imputed data, not the one we used for regres


#make a binary variable of bully scored 2.5 or above
train_imputed$bully_high <- ifelse(train_imputed$bully >= 2.5, 1, 0)
train_imputed$bully_high <- factor(train_imputed$bully_high, levels = c(0, 1), labels = c("No"

#
# test_imputed$predicted_bully_level <- predicted_bully_level
# test_imputed$bully_high <- ifelse(test_imputed$predicted_bully_level >= 2.5, 1, 0)
# test_imputed$bully_high <- factor(test_imputed$bully_high, levels = c(0, 1), labels = c("No"


#rf classifier, bully and bully risk
#
# ctrl <- trainControl(
#     method = "cv",           # cross-validation
#     number = 5,              # 5-fold
#     classProbs = TRUE,       # get probabilities for AUC
#     summaryFunction = twoClassSummary,  # for ROC
#     savePredictions = "final"
# )
#
#
# # Fit the model
# rf_bully_class <- train(
#     bully_high ~ .,
#     data = train_imputed %>% select(-c(bully, student_id)),     # your cleaned dataset
#     method = "rf",
#     trControl = ctrl,
#     tuneLength = 5,
#     metric = "ROC"           # maximize AUC
# )

# saveRDS(rf_bully_class, file = "/Users/le/Desktop/pset3_predict_bully/data/rf_bully_class.rd



#plotting ROC and AUC

# probs = predicted probabilities for the positive class ("Yes")
# actual = actual binary outcome ("No"/"Yes")
```

```
rf_bully_class <- read_rds("/Users/le/Desktop/pset3_predict_bully/data/rf_bully_class.rds")
rf_bully_class$results
```

```
##   mtry       ROC      Sens        Spec      ROCSD      SensSD     SpecSD
## 1    2 0.8744678 1.0000000 0.002777778 0.02323757 0.000000000 0.00621130
## 2   24 0.8834762 0.9978331 0.107914764 0.01741868 0.002053710 0.05000665
## 3   47 0.8797662 0.9975621 0.132952816 0.01788477 0.001951130 0.05060657
## 4   70 0.8778713 0.9967498 0.152321157 0.02094268 0.002108537 0.06041780
## 5   93 0.8732019 0.9963434 0.152321157 0.02223903 0.002374617 0.06041780
```

```
plot(rf_bully_class)
```

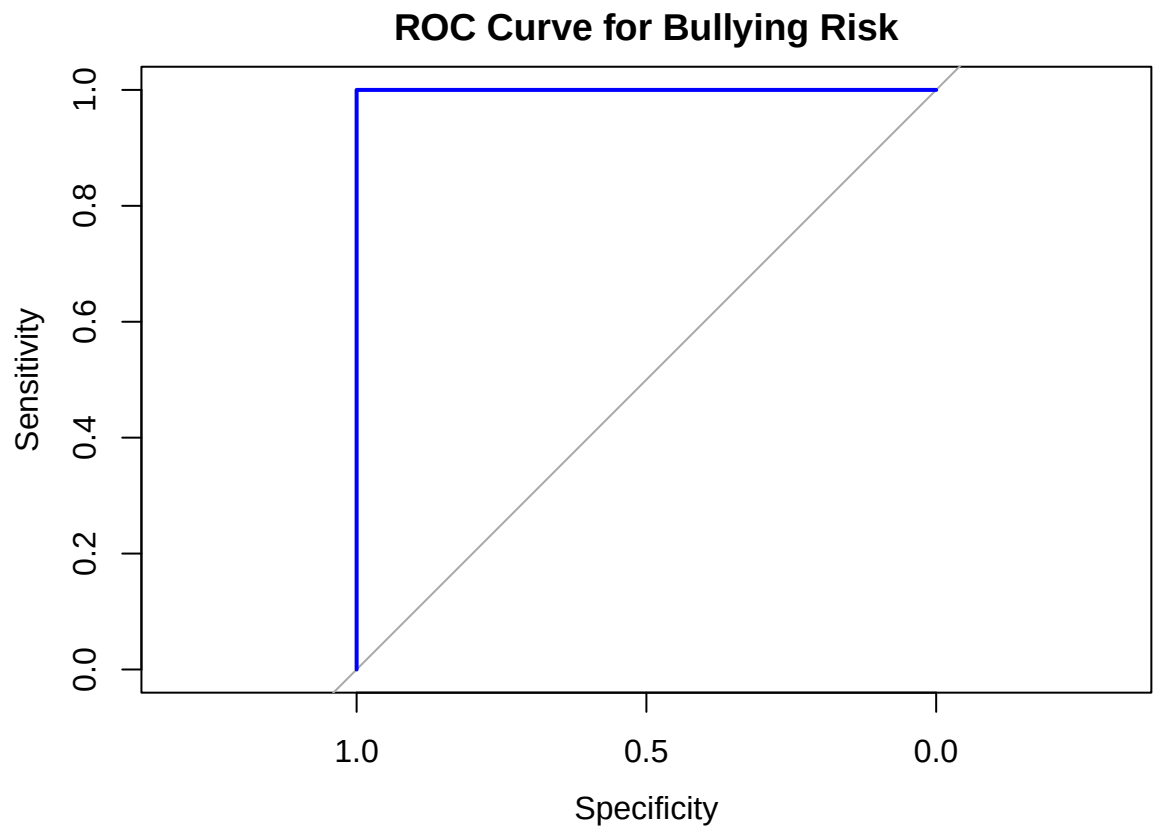


By looking at the ROC plot, we can see that the optimal AUC is around 0.883, which is a good performance. The sensitivity is `0.997` but specificity is only around `0.1`, this will be discussed later in dicussion questions.

```
# Get predicted probabilities
predicted_bully_probs <- predict(rf_bully_class, newdata = train_imputed, type = "prob")[, "Yes

# ROC Curve and AUC
roc_obj <- roc(train_imputed$bully_high, predicted_bully_probs)
```

```
## Setting levels: control = No, case = Yes

## Setting direction: controls < cases
```

```
plot(roc_obj, col = "blue", main = "ROC Curve for Bullying Risk")
```

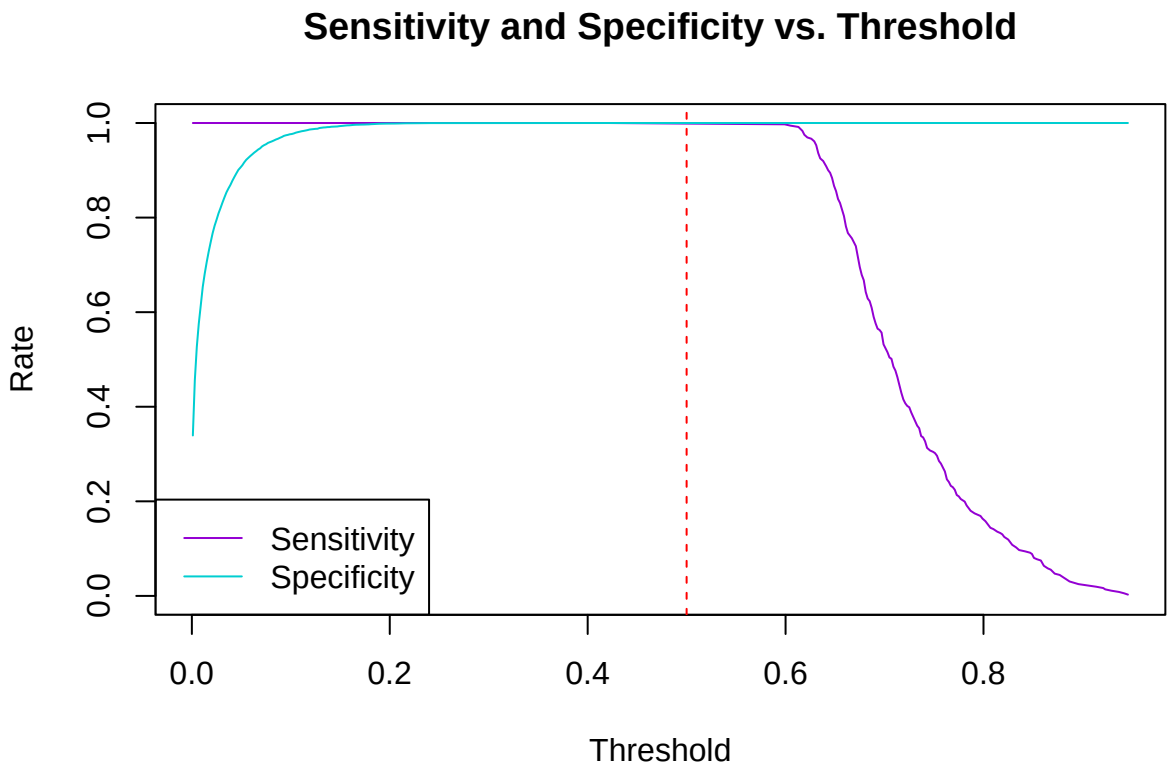



```
# abline(a = 0, b = 1, lty = 2, col = "gray")  # diagonal line

# Print AUC
auc(roc_obj)
```

```
## Area under the curve: 1
```

```
#Tuning the threshold manually, to get the best sensitivity and specificity cut off
# Get threshold performance stats
roc_coords <- coords(roc_obj, x = "all", input = "threshold", ret = c("threshold", "sensitivity

# Plot Sensitivity vs. Specificity
plot(roc_coords$threshold, roc_coords$sensitivity, type = "l", col = "darkviolet", ylim = c(0,1
     xlab = "Threshold", ylab = "Rate", main = "Sensitivity and Specificity vs. Threshold")
lines(roc_coords$threshold, roc_coords$specificity, col = "darkturquoise")
```

```r
legend("bottomleft", legend = c("Sensitivity", "Specificity"), col = c("darkviolet", "darkturqu
# Add a vertical line at the optimal threshold, make it bold
abline(v = 0.5, col = "red", lty = 2 )
```



By plotting the specificity against sensitivity, we can see that anywhere between 0.2 and 0,6 will give both = 1, we therefore will just use 0.5 as the threshold for classification. Next let's do a confusion matrix.

```r
# Get predicted classes on train data (using 0.5 threshold)
predicted_bully_probs <- predict(rf_bully_class, newdata = train_imputed, type = "prob")[, "Yes

pred_class <- ifelse(predicted_bully_probs >= 0.5, "Yes", "No")

# Actual classes (make sure they're factors with the same levels)
actual <- factor(train_imputed$bully_high, levels = c("No", "Yes"))

# Confusion matrix
conf <- confusionMatrix(factor(pred_class, levels = c("No", "Yes")), actual, positive = "Yes")
print(conf)
```

```
## Confusion Matrix and Statistics
##
```

```
##           Reference
## Prediction   No  Yes
##        No  7384    0
##        Yes    0  361
##
##                Accuracy : 1
##                  95% CI : (0.9995, 1)
##     No Information Rate : 0.9534
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.00000
##             Specificity : 1.00000
##          Pos Pred Value : 1.00000
##          Neg Pred Value : 1.00000
##              Prevalence : 0.04661
##          Detection Rate : 0.04661
##    Detection Prevalence : 0.04661
##       Balanced Accuracy : 1.00000
##
##        'Positive' Class : Yes
##
```

Finally, we can wrap it up into a csv file, with four variables, `student_id`, `predicted_bully_level`, `predicted_bully_high` and `predicted_bully_risk`.

```
#assemble for final predictions
final_predictions <- data.frame(
    student_id = test_imputed$student_id,
    predicted_bully_level = round(test_imputed$predicted_bully_level, 2),
    predicted_bully_high = test_imputed$predicted_bully_high,
    predicted_bully_risk = round(test_imputed$predicted_bully_probs, 2)
)

write.csv(final_predictions, "/Users/le/Desktop/pset3_predict_bully/data/Theta_Student_Predicti
```

Writeup

1) Give a quick overview of your data. In this overview be sure to answer at least the following:

```
a) How much data do you have?
b) What is the range of bully scores?
c) How common is the high level of bullying?

 The original train data has 195 variables and 8366 observations. The original test data has
 See line 325 - 330 for summary statistics: After cleaning, the train data has 44 variables a
```

2) Tell us what machine learning tool you used, and why/how you ended up selecting that model.

I eventually chose random forest for regression and classifications. First, I made a random forest for regression predictions, where I predicted students' continuous bullying scores. The training RMSE was 0.172, and the best-tuned model had an RMSE of 0.404 using 10-fold cross-validation. Then, I did a classification task, where I labeled students with a bully score of 2.5 or above as "high risk" and used a random forest classifier to predict this binary outcome. I evaluated model performance using ROC curves, AUC, and confusion matrices, and tuned the threshold to balance sensitivity and specificity. The final predictions included both the predicted score and bullying risk category for each student.

I also tried ridge and lasso but obtained inconsistent lambda values. The reason for choosing ridge and lasso is because the bully survey has many questions and many of which are similar (potentially correlated). This makes elastic net model a good fit to make predictions, since it helps to perform feature selections better and shrink the unimportant coefficients, such that we can include all the coefficients instead of selecting a few that may cause bias. But, I got stuck the halfway, so I gave up and chose random forest instead.

3) Estimate how well you *think* you are going to do on the set-aside data we have. In particular, report your estimate of what RMSE you will have on the continuous predictor, and what false positive rate and false negative rate you will have on your categorical predictions.

I expect the RMSE for the continuous predictor to be slightly higher than on the training data, though not by much, assuming the model generalizes well. If I used cross-validation, the reported training RMSE may actually reflect average performance on unseen folds, so a slightly lower test RMSE wouldn't be surprising. This is an indication of improved accuracy of predictions, since it is not over fitting the training set. The model is flexible and can be implemented onto other sets of data. Therefore, I expect reasonable generalization with test RMSE close to the training set, indicating the model hasn't overfit and is likely to perform well.

For the categorical predictions, I anticipate a modest increase in both the false positive rate (FPR) and false negative rate (FNR) on the test set compared to training. FPR means we are falsely labeling students as bullied when they're not being bullied, while a FNR means among those who are being bullied, we are mislabeling them as not being bullied. That said, minimizing false negatives is more important, when predicting students who are at risk for bullying. We want a low FNR because we don't want to miss any students who really need help. It's better to catch

true positives, even at the cost of a few extra false alarms. In that sense, a good model prioritizes reducing FNR, even at the cost of a slightly higher FPR.

4) Identify which variables were generally the most helpful for prediction?

As variables of importance plot shows, when predicting the bully scores, the most important variables for predicting bullying levels in students are those related to students' perceived safety, discrimination experiences, and sense of belonging (Q12 and Q13). The variable safe_mean, constructed from items like "How physically safe do you feel..." and "How safe do you feel from teasing and exclusion..." in places such as classrooms, hallways, and buses, had the highest predictive power. This highlights that perceived safety in school environments is the strongest indicator of bullying risk. The next most important predictor, disc_mean, was derived from Question 16, responses about perceived discrimination (e.g., based on race, gender, religion), suggesting that students who report being targeted for their identity traits are more likely to report higher bullying levels. The third variable, belong_mean, is from Question 18, reflecting how connected and respected students feel at school. Items like "Students care about one another" and "Teachers and students treat each other with respect" indicate that a strong sense of belonging is protective against bullying. Overall, this feature compression highlighted that bullying is most closely associated with how safe and included students feel, and whether they experience or witness bias or exclusion in daily school life. These could guide targeted interventions in addressing students of bully at risks.

Discussion Questions

The following questions should be answered and turned in as a pdf.

1) Give a reason for preferring a binary classification vs. the continuous classification, or vice-versa.

Using binary classification, like predicting whether a student is "bully_high" or not, is useful when the goal is to clearly flag students at risk and take action. It's simple and easier to explain, especially in real-world settings like schools. On the other hand, using the continuous bully score gives more detailed information about where a student falls on the bullying spectrum. This helps if we want to understand different levels of risk, not just high or low. So if the goal is to identify and support students at the highest risk, binary classification makes sense. But if we want a more nuanced understanding of bullying experiences across all students, the continuous score might be better.

2) Is it more important to have a good false positive rate or good false negative rate in this context? Tune your classifier to take this into account, if you can.

It is more important to have a low false negative rate. That means we'd rather catch as many students who are at risk of bullying as possible, even if we accidentally include some who aren't. Missing someone who is truly being bullied could mean they don't get help, which has serious consequences. On the other hand, a false positive might lead to checking in with a student who turns out to be fine, which is still okay. From the plots, sensitivity stays high across a wide range of thresholds, so I would tune the classifier to favor higher sensitivity, even if it slightly lowers specificity. That way, we reduce the chance of missing someone who actually needs support.

From the sensitivity and specificity plot we built on the training data, we want to choose a level at which it optimize the sensitivity and specificity, we therefore chose a classification threshold of 0.5, to prioritize identifying students at high risk of bullying. At this threshold, the model maintains

very high sensitivity, meaning it successfully flags nearly all students who are actually experiencing elevated bullying, which aligns with our goal of minimizing false negatives. If (or not) this choice does reduce specificity, leading to more false positives, we believe this is an acceptable trade-off in a context where missing a truly at-risk student may have serious consequences. This threshold allows the model to serve as a screening tool, casting a wider net to ensure that students in need of support are not overlooked.

3) Do you think your estimate of RMSE is too high or too low? Why?

For predicting continuous bullying levels, I trained a random forest regression model using 10-fold cross-validation. The caret package was used to tune the model, and the best performance was achieved at mtry = 47 with an RMSE of 0.404, which reflects realistic generalization error. A much lower RMSE of 0.172 was found when predicting on the training data directly, showing expected overfitting on seen data. To understand which variables were driving predictions, I extracted the top 15 most important features as shown earlier.

In terms of tuning the model, I don't think the 0.404 RMSE is too high, it actually gives a better sense of how the model will generalize. The 0.172 value is too low because it reflects overfitting. In short, the tuning RMSE is more trustworthy for evaluating real-world performance. We should use the RMSE from model tuning. In this case, the 0.404, to evaluate model performance. That value comes from cross-validation, which tests the model on data it hasn't seen, making it a much better estimate of how the model will perform on new or real-world data. The training RMSE (0.172) is useful to check for overfitting, but it's too optimistic and doesn't reflect generalization. So for reporting and comparing model quality, we should rely on the cross-validated RMSE.

4) Do you think your model is predictive enough to be useful? Why or why not? For the random forest regression, yes, I think the model is predictive enough to be useful. An RMSE of around 0.4 means the model can estimate bullying scores with reasonable accuracy on a scale that likely ranges from 1 to 5. It can still help identify students who are more at risk compared to others although imperfections are unavoidable.

For the classifier, the cross-validated results show that the model performs well in identifying students at risk of bullying, with a strong ROC AUC around 0.883 at its best (mtry = 24). Sensitivity is extremely high across all models, meaning it successfully captures nearly all students who are truly at risk. However, specificity is quite low, which indicates a high false positive rate, meaning the model often predicts bullying risk even when it's not present. This trade-off suggests the model prioritizes not missing any at-risk students, which makes sense in a school setting where the cost of missing someone in need is greater than mistakenly flagging someone. While the model is useful for early detection, it should be only used as a tool to guide further support or screening rather than making final decisions on its own.

Also, since the model compressed number of predictors, This likely improved both performance and interpretability. It also helped avoid overfitting and reduce noise, since fewer variables mean the model is less likely to memorize random patterns. Overall, this kind of thoughtful feature engineering made your pipeline stronger, more efficient, and easier to understand.

5) Do some of the variables seem "unfair" to you as variables that one would reasonably have for this kind of risk prediction?

Some variables do raise fairness concerns. For example, variables like disc_race, disc_gender, or disc_sex_or are important for understanding bullying, but using them in prediction could feel sensitive. These reflect personal identity and lived experience, so using them to label someone as

"at risk" might feel invasive or even stigmatizing. Also, if a school used such predictions without context, it could unintentionally reinforce bias. At the same time, ignoring these variables might hide important patterns.

6) Are there any ethical concerns you have for this sort of activity? What might those be? (This discussion can make reference to your findings, e.g., your estimated error rates.)

Yes, a key ethical concern is the high false positive rate seen in the model. While it's good at catching students truly at risk (high sensitivity), it also flags many who may not need help. This could lead to unnecessary attention or stigma that may cause troubles to the students. Since the predictions are based on personal survey responses, there is also a risk of misusing sensitive data. Therefore, the model should not be used to label students without context, also crucial to use de-identification techniques for other uses.