EDUC 423B/SOC 302B: Assignment 1

**Caity McGinley**

**4/29/21 OAE Extension**

Hi Caity,

What was going on with the .50 AUC everywhere, right? Something was off there, but by reading the code I can’t quite figure out what might have been the problem. I am going to see if I can reproduce it and figure it out ☺. I don’t ding you for this result, because if I can’t see what’s the matter just now, I don’t expect you to have solved it.

This assignment shows you work through the tidymodels process: all the code looks good. I would have liked to see more reflection on the models besides model performance (coefficients, importance) and why a model with only demographic, largely “ascribed” models doesn’t work so well from performance and ethical reasons. In addition, I agree that simple / faster models are great, but in the end, we are looking for a model that predicts better. You probably had that in mind as well and making this point “all things equal” but the write up seemed to suggest efficiency was all you were looking for.

You’ve added an interesting new predictor to the mix and I would have thought that would make a bigger difference, but it didn’t . My hunch is that this was due to the na.omit() code where we probably lost a lot of learners that did not finish any assignments. Among them likely a bunch of learners who withdrew. OR this is related to the weird behavior that the models classify a 100% as complete and 0% as withdrawn.

Grade: A-

# Honor Code Statement

I strongly encourage students to form study groups and students may discuss and work on assignments in groups. I expect that each student understands their own submission. As such, students must write their submissions independently and clearly disclose the names of all other students who were part of their study group. Additionally, lifting code or solutions directly from the internet (e.g., Google, GitHub, Stack Overflow) is a violation of the [Stanford Honor Code](https://communitystandards.stanford.edu/policies-and-guidance/honor-code). I take academic honesty and Honor Code violations extremely seriously and expect the same of students. If you have questions about what may or may not constitute an Honor Code violation, please reach out to me and we will figure it out together.

**Email:** [caitym26@stanford.edu](mailto:caitym26@stanford.edu)

**SUID:** 06211809 **Study Group: Jon and Victoria**

I acknowledge and agree to abide by the Honor Code.

**Signed:**

Caity McGinley

# Setup and data manipulation

#Libraries

library(tidyverse)

library(tidymodels)

library(vip)

library(rpart.plot)

library(ranger)

library(skimr)

library(doParallel)

library(ggplot2)  
library(glmnet)

library(janitor)

library(plotROC)

#Path

setwd("C:/Users/cmcgi/Downloads/Soc 302A\_Lab1")  
  
#Data

assessments <- read\_csv("assessments.csv")

##   
## -- Column specification --------------------------------------------------------  
## cols(  
## code\_module = col\_character(),  
## code\_presentation = col\_character(),  
## id\_assessment = col\_double(),  
## assessment\_type = col\_character(),  
## date = col\_double(),  
## weight = col\_double()  
## )

courses <- read\_csv("courses.csv")

##   
## -- Column specification --------------------------------------------------------  
## cols(  
## code\_module = col\_character(),  
## code\_presentation = col\_character(),  
## module\_presentation\_length = col\_double()  
## )

studentassessment <- read\_csv("studentAssessment.csv")

##   
## -- Column specification --------------------------------------------------------  
## cols(  
## id\_assessment = col\_double(),  
## id\_student = col\_double(),  
## date\_submitted = col\_double(),  
## is\_banked = col\_double(),  
## score = col\_double()  
## )

studentinfo <- read\_csv("studentInfo.csv")

##   
## -- Column specification --------------------------------------------------------  
## cols(  
## code\_module = col\_character(),  
## code\_presentation = col\_character(),  
## id\_student = col\_double(),  
## gender = col\_character(),  
## region = col\_character(),  
## highest\_education = col\_character(),  
## imd\_band = col\_character(),  
## age\_band = col\_character(),  
## num\_of\_prev\_attempts = col\_double(),  
## studied\_credits = col\_double(),  
## disability = col\_character(),  
## final\_result = col\_character()  
## )

studentregistration <- read\_csv("studentRegistration.csv")

##   
## -- Column specification --------------------------------------------------------  
## cols(  
## code\_module = col\_character(),  
## code\_presentation = col\_character(),  
## id\_student = col\_double(),  
## date\_registration = col\_double(),  
## date\_unregistration = col\_double()  
## )

studentvle <- read\_csv("studentVle.csv")

##   
## -- Column specification --------------------------------------------------------  
## cols(  
## code\_module = col\_character(),  
## code\_presentation = col\_character(),  
## id\_student = col\_double(),  
## id\_site = col\_double(),  
## date = col\_double(),  
## sum\_click = col\_double()  
## )

vle <- read\_csv("vle.csv")

##   
## -- Column specification --------------------------------------------------------  
## cols(  
## id\_site = col\_double(),  
## code\_module = col\_character(),  
## code\_presentation = col\_character(),  
## activity\_type = col\_character(),  
## week\_from = col\_double(),  
## week\_to = col\_double()  
## )

#cleaning and joining from W6\_Restructuring   
  
d <- inner\_join(studentinfo, studentassessment)

## Joining, by = "id\_student"

n\_distinct(studentinfo$id\_student) == n\_distinct(d$id\_student)

## [1] FALSE

n\_distinct(studentinfo$id\_student) - n\_distinct(d$id\_student)

## [1] 5416

studentinfo %>%  
 filter(!id\_student %in% unique(d$id\_student)) %>%  
 select(id\_student) %>%  
 unique()

## # A tibble: 5,416 x 1  
## id\_student  
## <dbl>  
## 1 30268  
## 2 135335  
## 3 281589  
## 4 292923  
## 5 305539  
## 6 346843  
## 7 354858  
## 8 405961  
## 9 494424  
## 10 1763015  
## # ... with 5,406 more rows

#joining  
d <- left\_join(studentinfo, studentassessment)

## Joining, by = "id\_student"

n\_distinct(studentinfo$id\_student) == n\_distinct(d$id\_student)

## [1] TRUE

count(d, is.na(id\_assessment))

## # A tibble: 2 x 2  
## `is.na(id\_assessment)` n  
## <lgl> <int>  
## 1 FALSE 207319  
## 2 TRUE 5847

#cleaning  
studentinfo %>%  
 mutate(dup\_id = duplicated(id\_student) | duplicated(id\_student, fromLast = T)) %>%  
 filter(dup\_id == T) %>%  
 arrange(id\_student)

## # A tibble: 7,346 x 13  
## code\_module code\_presentation id\_student gender region highest\_education  
## <chr> <chr> <dbl> <chr> <chr> <chr>   
## 1 DDD 2013J 8462 M London Reg~ HE Qualification   
## 2 DDD 2014J 8462 M London Reg~ HE Qualification   
## 3 DDD 2013B 24213 F East Angli~ A Level or Equiv~  
## 4 DDD 2014B 24213 F East Angli~ A Level or Equiv~  
## 5 BBB 2013J 25629 F Scotland Lower Than A Lev~  
## 6 BBB 2014B 25629 F Scotland Lower Than A Lev~  
## 7 DDD 2013J 27417 M South West~ Lower Than A Lev~  
## 8 DDD 2014J 27417 M South West~ Lower Than A Lev~  
## 9 BBB 2013B 27891 M Scotland Lower Than A Lev~  
## 10 BBB 2014B 27891 M Scotland Lower Than A Lev~  
## # ... with 7,336 more rows, and 7 more variables: imd\_band <chr>,  
## # age\_band <chr>, num\_of\_prev\_attempts <dbl>, studied\_credits <dbl>,  
## # disability <chr>, final\_result <chr>, dup\_id <lgl>

#joining  
full\_assessments <- full\_join(assessments, studentassessment)

## Joining, by = "id\_assessment"

summary(full\_assessments)

## code\_module code\_presentation id\_assessment assessment\_type   
## Length:173930 Length:173930 Min. : 1752 Length:173930   
## Class :character Class :character 1st Qu.:15022 Class :character   
## Mode :character Mode :character Median :25359 Mode :character   
## Mean :26554   
## 3rd Qu.:34883   
## Max. :40088   
##   
## date weight id\_student date\_submitted  
## Min. : 12.0 Min. : 0.00 Min. : 6516 Min. :-11   
## 1st Qu.: 54.0 1st Qu.: 0.00 1st Qu.: 504429 1st Qu.: 51   
## Median :129.0 Median : 9.00 Median : 585208 Median :116   
## Mean :130.6 Mean : 12.75 Mean : 705151 Mean :116   
## 3rd Qu.:214.0 3rd Qu.: 18.00 3rd Qu.: 634498 3rd Qu.:173   
## Max. :261.0 Max. :100.00 Max. :2698588 Max. :608   
## NA's :2873 NA's :18 NA's :18   
## is\_banked score   
## Min. :0.00000 Min. : 0.0   
## 1st Qu.:0.00000 1st Qu.: 65.0   
## Median :0.00000 Median : 80.0   
## Mean :0.01098 Mean : 75.8   
## 3rd Qu.:0.00000 3rd Qu.: 90.0   
## Max. :1.00000 Max. :100.0   
## NA's :18 NA's :191

#joining  
d <- left\_join(studentinfo, full\_assessments) I would have just started here personally. Why do the wrong join first? This was in the lab for educational purposes.

## Joining, by = c("code\_module", "code\_presentation", "id\_student")

#cleaning, grouping  
d %>%  
 mutate(weighted\_score = score \* weight) %>%  
 group\_by(imd\_band) %>%  
 summarize(average\_weighted\_score = mean(score, na.rm = T),  
 sd\_weighted\_score = sd(score, na.rm = T)) %>%  
 na.omit()

## # A tibble: 10 x 3  
## imd\_band average\_weighted\_score sd\_weighted\_score  
## <chr> <dbl> <dbl>  
## 1 0-10% 72.7 20.0  
## 2 10-20 73.7 19.8  
## 3 20-30% 74.9 19.2  
## 4 30-40% 75.2 18.9  
## 5 40-50% 75.9 18.6  
## 6 50-60% 75.8 18.9  
## 7 60-70% 76.3 18.4  
## 8 70-80% 76.5 18.3  
## 9 80-90% 77.8 17.9  
## 10 90-100% 78.0 17.7

#data in d now

Kudos for careful merging of data and exploring the data

What happens with students who don’t have any valid scores? Will they be NA and will you be filtering them from the data if you na.omit() ? That might be problematic because withdrawn students are probably less likely to have an assignment finished than other students.   
  
#interested in gender, socioeconomic status, highest education, age, and disability status  
# gender (gender), highest ed (highest\_education), age (age\_band), disability status (disability), SES (imd\_band), final\_result  
  
  
glimpse(d) #checking

## Rows: 180,662  
## Columns: 19  
## $ code\_module <chr> “AAA”, “AAA”, “AAA”, “AAA”, “AAA”, “AAA”, “AAA”, ~  
## $ code\_presentation <chr> “2013J”, “2013J”, “2013J”, “2013J”, “2013J”, “201~  
## $ id\_student <dbl> 11391, 11391, 11391, 11391, 11391, 28400, 28400, ~  
## $ gender <chr> “M”, “M”, “M”, “M”, “M”, “F”, “F”, “F”, “F”, “F”,~  
## $ region <chr> “East Anglian Region”, “East Anglian Region”, “Ea~  
## $ highest\_education <chr> “HE Qualification”, “HE Qualification”, “HE Quali~  
## $ imd\_band <chr> “90-100%”, “90-100%”, “90-100%”, “90-100%”, “90-1~  
## $ age\_band <chr> “55<=”, “55<=”, “55<=”, “55<=”, “55<=”, “35-55”, ~  
## $ num\_of\_prev\_attempts <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~  
## $ studied\_credits <dbl> 240, 240, 240, 240, 240, 60, 60, 60, 60, 60, 60, ~  
## $ disability <chr> “N”, “N”, “N”, “N”, “N”, “N”, “N”, “N”, “N”, “N”,~  
## $ final\_result <chr> “Pass”, “Pass”, “Pass”, “Pass”, “Pass”, “Pass”, “~  
## $ id\_assessment <dbl> 1752, 1753, 1754, 1755, 1756, 1752, 1753, 1754, 1~  
## $ assessment\_type <chr> “TMA”, “TMA”, “TMA”, “TMA”, “TMA”, “TMA", "TMA", ~  
## $ date <dbl> 19, 54, 117, 166, 215, 19, 54, 117, 166, 215, NA,~  
## $ weight <dbl> 10, 20, 20, 20, 30, 10, 20, 20, 20, 30, NA, 10, 2~  
## $ date\_submitted <dbl> 18, 53, 115, 164, 212, 22, 52, 121, 164, 212, NA,~  
## $ is\_banked <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, NA, 0, 0, 0, 0, 0, ~  
## $ score <dbl> 78, 85, 80, 85, 82, 70, 68, 70, 64, 60, NA, 72, 7~

table(d$final\_result) #Not binary

##   
## Distinction Fail Pass Withdrawn   
## 26330 29750 106024 18558

table(d$gender) #binary

##   
## F M   
## 83131 97531

table(d$disability) #binary

##   
## N Y   
## 164733 15929

table(d$num\_of\_prev\_attempts) #could be interesting other variable to look at

##   
## 0 1 2 3 4 5 6   
## 159492 17040 3268 623 181 43 15

skim(d) what a nice function! I gotta remember that one myself!

Data summary

|  |  |
| --- | --- |
| Name | d |
| Number of rows | 180662 |
| Number of columns | 19 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Column type frequency: |  |
| character | 10 |
| numeric | 9 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Group variables | None |

**Variable type: character**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| skim\_variable | n\_missing | complete\_rate | min | max | empty | n\_unique | whitespace |
| code\_module | 0 | 1.00 | 3 | 3 | 0 | 7 | 0 |
| code\_presentation | 0 | 1.00 | 5 | 5 | 0 | 4 | 0 |
| gender | 0 | 1.00 | 1 | 1 | 0 | 2 | 0 |
| region | 0 | 1.00 | 5 | 20 | 0 | 13 | 0 |
| highest\_education | 0 | 1.00 | 15 | 27 | 0 | 5 | 0 |
| imd\_band | 7810 | 0.96 | 5 | 7 | 0 | 10 | 0 |
| age\_band | 0 | 1.00 | 4 | 5 | 0 | 3 | 0 |
| disability | 0 | 1.00 | 1 | 1 | 0 | 2 | 0 |
| final\_result | 0 | 1.00 | 4 | 11 | 0 | 4 | 0 |
| assessment\_type | 6750 | 0.96 | 3 | 4 | 0 | 3 | 0 |

**Variable type: numeric**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| skim\_variable | n\_missing | complete\_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
| id\_student | 0 | 1.00 | 704970.61 | 551538.73 | 3733 | 505199 | 585503 | 634646 | 2716795 | ▅▇▁▁▁ |
| num\_of\_prev\_attempts | 0 | 1.00 | 0.15 | 0.45 | 0 | 0 | 0 | 0 | 6 | ▇▁▁▁▁ |
| studied\_credits | 0 | 1.00 | 77.32 | 37.92 | 30 | 60 | 60 | 90 | 655 | ▇▁▁▁▁ |
| id\_assessment | 6750 | 0.96 | 26553.80 | 8829.78 | 1752 | 15022 | 25359 | 34883 | 37443 | ▁▅▁▅▇ |
| date | 9615 | 0.95 | 130.61 | 78.03 | 12 | 54 | 129 | 214 | 261 | ▇▅▅▃▇ |
| weight | 6750 | 0.96 | 12.74 | 17.88 | 0 | 0 | 9 | 18 | 100 | ▇▂▁▁▁ |
| date\_submitted | 6750 | 0.96 | 116.03 | 71.48 | -11 | 51 | 116 | 173 | 608 | ▇▇▁▁▁ |
| is\_banked | 6750 | 0.96 | 0.01 | 0.10 | 0 | 0 | 0 | 0 | 1 | ▇▁▁▁▁ |
| score | 6923 | 0.96 | 75.80 | 18.80 | 0 | 65 | 80 | 90 | 100 | ▁▁▂▇▇ |
| o |  |  |  |  |  |  |  |  |  |  |

Ok phew foks with missing data on score are still in the data. But what are you going to do with missing score values?

#Thoughts  
#age, highest\_ed, final\_result - categorical   
#Disability can be binary, gender can be binary (character)  
#final\_result might to just be completion or withdrawn   
#imd\_band - numeric   
  
sapply(d, class)

## code\_module code\_presentation id\_student   
## "character" "character" "numeric"   
## gender region highest\_education   
## "character" "character" "character"   
## imd\_band age\_band num\_of\_prev\_attempts   
## "character" "character" "numeric"   
## studied\_credits disability final\_result   
## "numeric" "character" "character"   
## id\_assessment assessment\_type date   
## "numeric" "character" "numeric"   
## weight date\_submitted is\_banked   
## "numeric" "numeric" "numeric"   
## score   
## "numeric"

Good, althpugh you could gather this from your glimpse() as well.

table(d$imd\_band)

##   
## 0-10% 10-20 20-30% 30-40% 40-50% 50-60% 60-70% 70-80% 80-90% 90-100%   
## 16161 17698 18405 19957 17751 17557 16679 16765 16344 15535

#Filtering for selected variables   
data <- d %>%   
 select (gender, final\_result, disability, imd\_band, highest\_education, age\_band)   
  
  
#Cleaning missingness from imd\_band

data <- na.omit(data)  
  
table(data$imd\_band)

##   
## 0-10% 10-20 20-30% 30-40% 40-50% 50-60% 60-70% 70-80% 80-90% 90-100%   
## 16161 17698 18405 19957 17751 17557 16679 16765 16344 15535

# Making binary factor outcome, making numeric

data <- data %>%  
 mutate(final\_result = case\_when(  
 final\_result == "Pass" ~ "Completed",  
 final\_result == "Fail" ~ "Completed",  
 final\_result == "Distinction" ~ "Completed",   
 final\_result == "Withdrawn" ~ "Withdrew")) %>%   
 mutate(imd\_band = case\_when(  
 imd\_band == "0-10%" ~ 1,   
 imd\_band == "10-20" ~ 2,  
 imd\_band == "20-30%" ~ 3,  
 imd\_band == "30-40%" ~ 4,  
 imd\_band == "40-50%" ~ 5,  
 imd\_band == "50-60%" ~ 6,  
 imd\_band == "60-70%" ~ 7,  
 imd\_band == "70-80%" ~ 8,  
 imd\_band == "80-90%" ~ 9,  
 imd\_band == "90-100%" ~ 10,  
 ))   
  
table(data$imd\_band)

##   
## 1 2 3 4 5 6 7 8 9 10   
## 16161 17698 18405 19957 17751 17557 16679 16765 16344 15535

Good work!

**Predicting MOOC completion with student demographic characteristics**

To what extent do student characteristics gender, socioeconomic status, highest education, age, and disability status predict MOOC completion using different machine learning algorithms? Use at least 2 supervised learning algorithms and use cross validation to tune hyperparamaters and assess model performance on the test data.

#create your training and testing data  
set.seed(1234)  
d\_split <-   
 initial\_split(data,   
 prop = .8,   
 strata = final\_result)  
  
#training  
d\_train <-   
 training(d\_split)  
  
#testing  
d\_test <-  
 testing(d\_split)  
  
  
#Algorithm  
  
# Decision tree  
dt\_model <-   
 decision\_tree(cost\_complexity = tune(),  
 tree\_depth = tune()) %>%   
 set\_engine("rpart") %>%   
 set\_mode("classification")  
  
  
# Logistic regression with regularization  
regularization\_model <-   
 logistic\_reg(penalty = tune(),   
 mixture = tune()) %>%  
 set\_engine('glmnet') %>%   
 set\_mode('classification')  
  
# Random forest – wanna try this one too for practice   
rf\_model <-   
 rand\_forest(mtry = tune(),  
 trees = tune(),  
 min\_n = 10) %>%  
 set\_engine("ranger",   
 importance = "impurity") %>%  
 set\_mode("classification")   
  
  
  
#Recipe   
#the thing we’re trying to predict is the variable to the left of the ~, and the predictor variables are the things to the right of it  
  
#completion is what we are trying to predict   
#predictors are gender, socioeconomic status, highest education, age, and disability status  
# gender (gender), highest ed (highest\_education), age (age\_band), disability status (disability), SES (can't find that) imd\_band is a proxy for ses, final\_result, imd\_band – specifies the Index of Multiple Depravation band of the place where the student lived during the module-presentation (assumes that holds true)  
  
d\_recipe <-   
 recipe(final\_result ~ .,   
 data = d\_train) %>%   
 step\_normalize(imd\_band) %>%   
 step\_string2factor(final\_result, age\_band, highest\_education, gender, disability) %>%   
 step\_dummy(age\_band, highest\_education, gender, disability)  
d\_recipe

Good, but it is a bit redundant to first recode variables to factors and then to dummies again. You can just pass the variables to step\_dummy right away

## Data Recipe  
##   
## Inputs:  
##   
## role #variables  
## outcome 1  
## predictor 5  
##   
## Operations:  
##   
## Centering and scaling for imd\_band  
## Factor variables from final\_result, age\_band, ...  
## Dummy variables from age\_band, highest\_education, gender, disability

#Cross-validation and grid   
  
# 3) Set up cross validation----------------------------------------------------  
# Set up our grids  
  
  
  
# Decision tree  
dt\_grid <-   
 grid\_regular(cost\_complexity(),  
 tree\_depth(),  
 levels = 5)  
dt\_grid

## # A tibble: 25 x 2  
## cost\_complexity tree\_depth  
## <dbl> <int>  
## 1 0.0000000001 1  
## 2 0.0000000178 1  
## 3 0.00000316 1  
## 4 0.000562 1  
## 5 0.1 1  
## 6 0.0000000001 4  
## 7 0.0000000178 4  
## 8 0.00000316 4  
## 9 0.000562 4  
## 10 0.1 4  
## # ... with 15 more rows

# Regularization  
reg\_grid <-   
 grid\_regular(penalty(),  
 mixture(),  
 levels = 5)  
reg\_grid

## # A tibble: 25 x 2  
## penalty mixture  
## <dbl> <dbl>  
## 1 0.0000000001 0   
## 2 0.0000000316 0   
## 3 0.00001 0   
## 4 0.00316 0   
## 5 1 0   
## 6 0.0000000001 0.25  
## 7 0.0000000316 0.25  
## 8 0.00001 0.25  
## 9 0.00316 0.25  
## 10 1 0.25  
## # ... with 15 more rows

# Random forest  
rf\_grid <- grid\_regular(mtry(range = c(25, 100)), does this value make sense for your model with 12 variables?  
 trees(range = c(100, 500)),  
 levels = 5)  
rf\_grid

## # A tibble: 25 x 2  
## mtry trees  
## <int> <int>  
## 1 25 100  
## 2 43 100  
## 3 62 100  
## 4 81 100  
## 5 100 100  
## 6 25 200  
## 7 43 200  
## 8 62 200  
## 9 81 200  
## 10 100 200  
## # ... with 15 more rows

# Set up 5-fold cross validation  
d\_cv <-   
 vfold\_cv(d\_train, v = 5)  
  
  
# 4) Tune the hyperparameters  
  
# Decision tree  
d\_dt\_wf <-   
 workflow() %>%  
 add\_model(dt\_model) %>%  
 add\_recipe(d\_recipe)  
  
  
set.seed(1234)  
dt\_results <-   
 d\_dt\_wf %>%   
 tune\_grid(resamples = d\_cv,  
 grid = dt\_grid)

## Warning: package 'rlang' was built under R version 4.0.4

## Warning: package 'vctrs' was built under R version 4.0.4

dt\_results

# Processing

## # Tuning results  
## # 5-fold cross-validation   
## # A tibble: 5 x 4  
## splits id .metrics .notes   
## <list> <chr> <list> <list>   
## 1 <split [110626/27657]> Fold1 <tibble [50 x 6]> <tibble [0 x 1]>  
## 2 <split [110626/27657]> Fold2 <tibble [50 x 6]> <tibble [0 x 1]>  
## 3 <split [110626/27657]> Fold3 <tibble [50 x 6]> <tibble [0 x 1]>  
## 4 <split [110627/27656]> Fold4 <tibble [50 x 6]> <tibble [0 x 1]>  
## 5 <split [110627/27656]> Fold5 <tibble [50 x 6]> <tibble [0 x 1]>

# Regularization  
d\_reg\_wf <-   
 workflow() %>%  
 add\_model(regularization\_model) %>%  
 add\_recipe(d\_recipe)  
  
set.seed(1234)  
reg\_results <-   
 d\_reg\_wf %>%   
 tune\_grid(resamples = d\_cv,  
 grid = reg\_grid)  
reg\_results

## # Tuning results  
## # 5-fold cross-validation   
## # A tibble: 5 x 4  
## splits id .metrics .notes   
## <list> <chr> <list> <list>   
## 1 <split [110626/27657]> Fold1 <tibble [50 x 6]> <tibble [0 x 1]>  
## 2 <split [110626/27657]> Fold2 <tibble [50 x 6]> <tibble [0 x 1]>  
## 3 <split [110626/27657]> Fold3 <tibble [50 x 6]> <tibble [0 x 1]>  
## 4 <split [110627/27656]> Fold4 <tibble [50 x 6]> <tibble [0 x 1]>  
## 5 <split [110627/27656]> Fold5 <tibble [50 x 6]> <tibble [0 x 1]>

# Random forest  
d\_rf\_wf <-   
 workflow() %>%  
 add\_model(rf\_model) %>%  
 add\_recipe(d\_recipe)  
  
  
set.seed(1234)  
rf\_results <-   
 d\_rf\_wf %>%   
 tune\_grid(resamples = d\_cv,  
 grid = rf\_grid)

# Processing

## # Tuning results  
## # 5-fold cross-validation   
## # A tibble: 5 x 4  
## splits id .metrics .notes   
## <list> <chr> <list> <list>   
## 1 <split [110626/27657]> Fold1 <tibble [50 x 6]> <tibble [25 x 1]>  
## 2 <split [110626/27657]> Fold2 <tibble [50 x 6]> <tibble [25 x 1]>  
## 3 <split [110626/27657]> Fold3 <tibble [50 x 6]> <tibble [25 x 1]>  
## 4 <split [110627/27656]> Fold4 <tibble [50 x 6]> <tibble [25 x 1]>  
## 5 <split [110627/27656]> Fold5 <tibble [50 x 6]> <tibble [25 x 1]>

This model has notes (25x1) that’s probably because the values for mtry were not relevant to this siuatuin. You did not have a hundred variables to sample, just 10 or so with 12 precictor variabls in total .

# 5) Pick the best model and evaluate performance  
# Decision tree  
final\_tree\_preds <-   
 d\_dt\_wf %>%   
 finalize\_workflow(select\_best(dt\_results,   
 "roc\_auc")) %>%  
 last\_fit(d\_split) %>%  
 collect\_predictions() %>%  
 mutate(Algorithm = "Decision tree")  
  
  
# Regularization  
final\_reg\_preds <-   
d\_reg\_wf %>%   
 finalize\_workflow(select\_best(reg\_results,   
 "roc\_auc")) %>%  
 last\_fit(d\_split) %>%  
 collect\_predictions() %>%  
 mutate(Algorithm = "Elastic net logistic regression")  
  
# Random forest  
final\_forest\_preds <-   
 d\_rf\_wf %>%   
 finalize\_workflow(select\_best(rf\_results,   
 "roc\_auc")) %>%  
 last\_fit(d\_split) %>%  
 collect\_predictions() %>%  
 mutate(Algorithm = "Random forest")

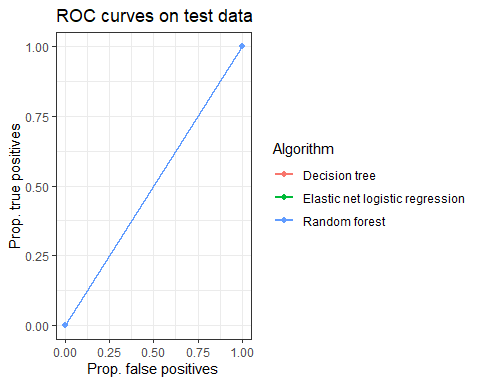
# Now we combine all our metrics and plot  
all\_predict <- final\_reg\_preds %>%  
 rbind(final\_tree\_preds) %>%   
 rbind(final\_forest\_preds)  
  
table(all\_predict$Algorithm)

##   
## Decision tree Elastic net logistic regression   
## 34569 34569   
## Random forest   
## 34569

# Only shows one algorithm at a time

roc\_plot1 <- ggplot(all\_predict,   
 aes(d = final\_result,  
 m = .pred\_class,   
 color = Algorithm)) +   
 geom\_roc(labels = FALSE) +  
 labs(title = "ROC curves on test data",  
 y = "Prop. true positives",  
 x = "Prop. false positives") +  
 theme\_bw()   
roc\_plot1

## Warning in verify\_d(data$d): D not labeled 0/1, assuming Completed = 0 and  
## Withdrew = 1!



Hmmm that doesn’t look right

roc\_plot1$data

## # A tibble: 103,707 x 8  
## id .pred\_Completed .pred\_Withdrew .row .pred\_class final\_result .config   
## <chr> <dbl> <dbl> <int> <fct> <fct> <chr>   
## 1 train~ 0.922 0.0781 15 Completed Completed Preproc~  
## 2 train~ 0.887 0.113 18 Completed Completed Preproc~  
## 3 train~ 0.927 0.0735 26 Completed Completed Preproc~  
## 4 train~ 0.927 0.0726 33 Completed Completed Preproc~  
## 5 train~ 0.921 0.0793 41 Completed Completed Preproc~  
## 6 train~ 0.899 0.101 44 Completed Completed Preproc~  
## 7 train~ 0.899 0.101 46 Completed Completed Preproc~  
## 8 train~ 0.895 0.105 54 Completed Completed Preproc~  
## 9 train~ 0.895 0.105 55 Completed Completed Preproc~  
## 10 train~ 0.885 0.115 62 Completed Completed Preproc~  
## # ... with 103,697 more rows, and 1 more variable: Algorithm <chr>

# Hmmm not super great  
  
roc\_plot1$Algorithm <- na.omit(roc\_plot1$Algorithm)  
  
calc\_auc(roc\_plot1) #Path all are the same?

## Warning in verify\_d(data$d): D not labeled 0/1, assuming Completed = 0 and  
## Withdrew = 1!

## PANEL group AUC  
## 1 1 1 0.5  
## 2 1 2 0.5  
## 3 1 3 0.5  
## 4 1 4 0.5  
## 5 1 5 0.5

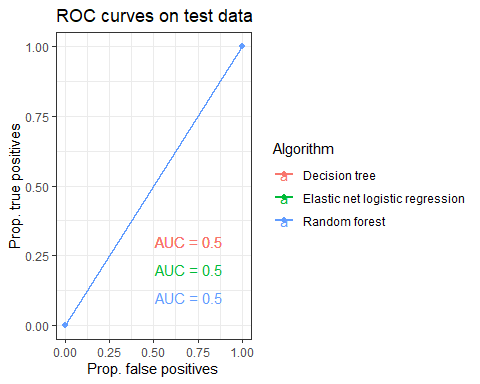
#Had some trouble with this one, it looks like roc\_plot1 produces five groups, but I only have three algorithms  
#Tried to just override this by putting the decision tree info in the same spot   
alc <- tibble(  
 Algorithm = case\_when(calc\_auc(roc\_plot1)$group == 1 ~ "Decision tree",  
 calc\_auc(roc\_plot1)$group == 2 ~ "Elastic net logistic regression",  
 calc\_auc(roc\_plot1)$group == 3 ~ "Random forest",   
 calc\_auc(roc\_plot1)$group == 4 ~ "Decision tree",  
 calc\_auc(roc\_plot1)$group == 5 ~ "Decision tree",),   
 AUC = str\_c("AUC = ", as.character(round(calc\_auc(roc\_plot1)$AUC, 3), sep = "")),  
 x = rep(.7, n = 5),  
 y = rep(c(.30, .20, .10, .30, .30)))

## Warning in verify\_d(data$d): D not labeled 0/1, assuming Completed = 0 and  
## Withdrew = 1!

## Warning in verify\_d(data$d): D not labeled 0/1, assuming Completed = 0 and  
## Withdrew = 1!  
  
## Warning in verify\_d(data$d): D not labeled 0/1, assuming Completed = 0 and  
## Withdrew = 1!  
  
## Warning in verify\_d(data$d): D not labeled 0/1, assuming Completed = 0 and  
## Withdrew = 1!  
  
## Warning in verify\_d(data$d): D not labeled 0/1, assuming Completed = 0 and  
## Withdrew = 1!  
  
## Warning in verify\_d(data$d): D not labeled 0/1, assuming Completed = 0 and  
## Withdrew = 1!

roc\_plot1 +   
 geom\_text(data = alc,   
 mapping = aes(d = NULL,  
 m = NULL,  
 x = x,   
 y = y,   
 color = Algorithm,  
 label = AUC))

## Warning in verify\_d(data$d): D not labeled 0/1, assuming Completed = 0 and  
## Withdrew = 1!



#all are the same, linear is not that great

Something is not right here. You should indeed have 3 models only. I see you are following the labs code so I am not sure what might have been causing this. Attaching example case for you

It is noticeable that the models returned predicted classes are all Complete. The models do not classify any case as Withdrawn. This is weird.

I went back to do a good ol fashioned glm and didn’t end up with 100% complete predictions. Would love to figure this out, but am in a bit of a time crunch right now, so it may take a bit

m <- glm(result ~ as.factor(gender) + as.factor(highest\_education) + as.factor(imd\_band) + as.factor(age\_band) + as.factor(disability), data = d, family = "binomial")

summary(m)

mypreds <- predict(m)

myclass <- ifelse(mypreds < 0.5, 0, 1)

gplot(myclass)



# Report to management

**Write a report to management about your findings. Shortly describe the dataset, the variables, and your modeling strategy (explain the algorithms used in your own words), the results, and model performance. Add your opinion of using one of the algorithms for a early warning system for learners at risk and suggest what might improve an early warning system.**

The MOOC data set consisted of +172k objects of 6 variables of interest: gender, completion status, disability SES, highest education level, and age. All variables were factors of multiple levels except for the variable measuring SES, which was numeric. (well, you assumed it to be numeric… technically a discrete variable) Three different supervised learning algorithms were utilized to explore the data. The algorithms used were decision tree, elastic net logistical regression, and random forest.

Firstly, Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. In our case, I utilized a classification model where the decision variable is Categorical. Yes otherwise it would have been called a regression tree The tree is constructed through a process known as binary recursive partitioning which is an iterative process of splitting the data into partitions, and then splitting it up further on each of the branches. A pro of DTs is that, compared to other algorithms, they require less effort for data preparation during pre-processing. It was a fairly quick process! It’s also easy to interpret. A con is that a small change in the data can cause a large change in the structure of the decision tree causing instability and overfitting of the data. Very nice to give this intuitive explanation to management.

Second, elastic net logistic regression is a type of regression model. The elastic net regression combines L1 norms (LASSO) and L2 norms (ridge regression) into a penalized model for generalized linear regression. This gives it sparsity (L1) and robustness (L2) properties. Might be good to be add a bit more explanation here It is a reasonable algorithm— the logistic regression doesn’t assume a linear relationship between independent and dependent variables, and the latter doesn’t need to be normally distributed. However, the elastic net is computationally more expensive than LASSO or Ridge. I also find the elastic net logistic regression harder to interpret than a decision tree.

Lastly, instead of relying upon one decision tree, the random forest takes the figure ( a subset) from each tree (the data) and resamples over and over again. Therefore, it is a collection of decision trees that is seen as a whole, rather than individually. The more noticeable number of trees in the forest prompts higher exactness and forestalls the issue of overfitting, which is a downfall of the decision tree. What does this mean in your own words? Nonetheless, aRandom Forest with large number of trees can make the processing slow and ineffective for real-time predictions (mine was particularly slow!).

It appears in all the models that highest level of education was the most important predictor in predicting whether someone would withdraw or complete the MOOC (vip code not included here, was run but not included in this Rmd). Why not included? Using the ROC plot, which is a plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied what doe sthat mean in your own words? , I found that all three algorithms had the same AUC (area under curve) value of ~.5. When AUC is approximately 0.5, the model has no discrimination capacity to distinguish between positive class and negative class. Therefore, none of the models were great. However, the operating times were widely different. Given the difference in operating time, I suggest the decision tree, because due to their branching structure, decision trees can easily model nonlinear relationships, as regressions cannot easily express non-linear relationships. Decision Trees are non-linear classifiers; they do not require data to be linearly separable. Logistic Regression wants data to be divided into two separate parts. I wasn’t sure, so therefore, I think decision tree is the better option. Furthermore, even though the DT can be prone to overfitting, the random forest took ~30 minutes to run. The logistic regression was even quicker than the DT, but I worried I might not be capturing something I could be with the decision tree. Like what? The resampling nature of the random forest made it difficult for my computer to handle. The decision tree is a more time effective option. Conclusively, I think the best early warning system would be a model that is time effective. I appreciate that you’ve thought about comparing the models in terms of their strengths and weaknesses, but I would disagree with you that it is time effectiveness we’re after. Yes, in case all models perform equally, then pick the easier and fastest one. But if there are differences in performance, I’d go for the best performance model ☺

You’ve looked at the importance (and perhaps also the coefficients for the direction of effects) but I don’t see them. This would have been nice to include

# Your suggestion

**Pick one suggestion to improve the algorithm for an early warning system and incorporate it in your analysis. Evaluate if your suggestion was an improvement.**

#new vairables I want to add: score   
data2 <- d %>%   
 select (gender, final\_result, disability, imd\_band, highest\_education, age\_band, score)   
  
  
  
data2 <- na.omit(data2) Yikes, you are throwing out people without a score! Those are likely a lot of withdrawn people.   
  
table(data2$imd\_band)

##   
## 0-10% 10-20 20-30% 30-40% 40-50% 50-60% 60-70% 70-80% 80-90% 90-100%   
## 15249 16763 17482 19183 17037 16964 16148 16251 15837 15129

data2 <- data2 %>%  
 mutate(final\_result = case\_when(  
 final\_result == "Pass" ~ "Completed",  
 final\_result == "Fail" ~ "Completed",  
 final\_result == "Distinction" ~ "Completed",   
 final\_result == "Withdrawn" ~ "Withdrew")) %>%   
 mutate(imd\_band = case\_when(  
 imd\_band == "0-10%" ~ 1,   
 imd\_band == "10-20" ~ 2,  
 imd\_band == "20-30%" ~ 3,  
 imd\_band == "30-40%" ~ 4,  
 imd\_band == "40-50%" ~ 5,  
 imd\_band == "50-60%" ~ 6,  
 imd\_band == "60-70%" ~ 7,  
 imd\_band == "70-80%" ~ 8,  
 imd\_band == "80-90%" ~ 9,  
 imd\_band == "90-100%" ~ 10,  
 ))   
  
  
#create your training and testing data  
set.seed(1234)  
d\_split2 <-   
 initial\_split(data2,   
 prop = .8,   
 strata = final\_result)  
  
#training  
d\_train2 <-   
 training(d\_split2)  
  
#testing  
d\_test2 <-  
 testing(d\_split2)  
  
# Logistic regression with regularization  
regularization\_model2 <-   
 logistic\_reg(penalty = tune(),   
 mixture = tune()) %>%  
 set\_engine('glmnet') %>%   
 set\_mode('classification')  
  
class(data2$num\_of\_prev\_attempts)

## Warning: Unknown or uninitialised column: `num\_of\_prev\_attempts`.

## [1] "NULL"

table(data2$num\_of\_prev\_attempts)

## Warning: Unknown or uninitialised column: `num\_of\_prev\_attempts`.

## < table of extent 0 >

d\_recipe2 <-   
 recipe(final\_result ~ .,   
 data = d\_train2) %>%   
 step\_normalize(imd\_band, score) %>%   
 step\_string2factor(final\_result, age\_band, highest\_education, gender, disability) %>%   
 step\_dummy(age\_band, highest\_education, gender, disability)  
d\_recipe2

## Data Recipe  
##   
## Inputs:  
##   
## role #variables  
## outcome 1  
## predictor 6  
##   
## Operations:  
##   
## Centering and scaling for imd\_band, score  
## Factor variables from final\_result, age\_band, ...  
## Dummy variables from age\_band, highest\_education, gender, disability

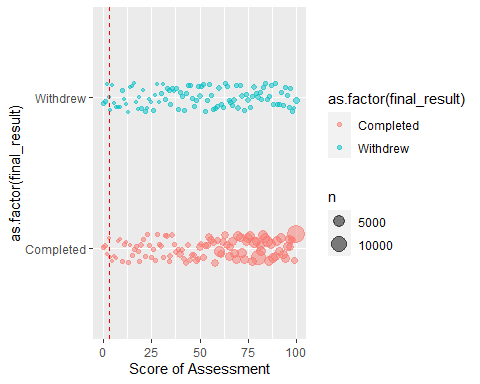
# Regularization  
reg\_grid2 <-   
 grid\_regular(penalty(),  
 mixture(),  
 levels = 5)  
reg\_grid2

## # A tibble: 25 x 2  
## penalty mixture  
## <dbl> <dbl>  
## 1 0.0000000001 0   
## 2 0.0000000316 0   
## 3 0.00001 0   
## 4 0.00316 0   
## 5 1 0   
## 6 0.0000000001 0.25  
## 7 0.0000000316 0.25  
## 8 0.00001 0.25  
## 9 0.00316 0.25  
## 10 1 0.25  
## # ... with 15 more rows

# Set up 5-fold cross validation  
d\_cv2 <-   
 vfold\_cv(d\_train2, v = 5)  
  
# Regularization  
d\_reg\_wf2 <-   
 workflow() %>%  
 add\_model(regularization\_model2) %>%  
 add\_recipe(d\_recipe2)  
  
set.seed(1234)  
reg\_results2 <-   
 d\_reg\_wf2 %>%   
 tune\_grid(resamples = d\_cv2,  
 grid = reg\_grid2)  
reg\_results2

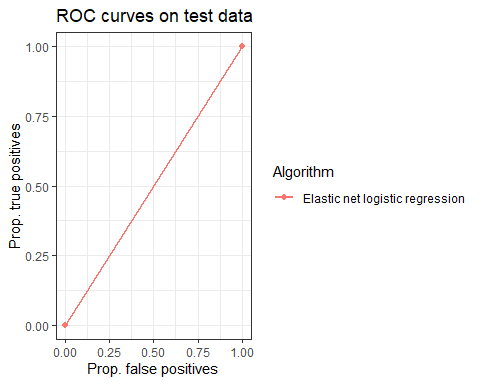
## # Tuning results  
## # 5-fold cross-validation   
## # A tibble: 5 x 4  
## splits id .metrics .notes   
## <list> <chr> <list> <list>   
## 1 <split [106268/26567]> Fold1 <tibble [50 x 6]> <tibble [0 x 1]>  
## 2 <split [106268/26567]> Fold2 <tibble [50 x 6]> <tibble [0 x 1]>  
## 3 <split [106268/26567]> Fold3 <tibble [50 x 6]> <tibble [0 x 1]>  
## 4 <split [106268/26567]> Fold4 <tibble [50 x 6]> <tibble [0 x 1]>  
## 5 <split [106268/26567]> Fold5 <tibble [50 x 6]> <tibble [0 x 1]>

# Regularization  
final\_reg\_preds2 <-   
 d\_reg\_wf2 %>%   
 finalize\_workflow(select\_best(reg\_results2,   
 "roc\_auc")) %>%  
 last\_fit(d\_split2) %>%  
 collect\_predictions() %>%  
 mutate(Algorithm = "Elastic net logistic regression")  
  
  
  
#Plot to see   
ggplot(data = d\_train2,   
 aes(x = score,   
 y = as.factor(final\_result))) +  
 geom\_count(aes(color = as.factor(final\_result)),  
 position = position\_jitter(width = 0,   
 height = 0.1),   
 alpha = 0.5) +  
 geom\_vline(xintercept = 3,   
 color = "red",   
 lty = 2) +  
 xlab("Score of Assessment")

Wow I would thought I’d see more of a difference between the withdrew and other cases.

#plot for AUC  
roc\_plot2 <- ggplot(final\_reg\_preds2,   
 aes(d = final\_result,  
 m = .pred\_class,   
 color = Algorithm)) +   
 geom\_roc(labels = FALSE) +  
 labs(title = "ROC curves on test data",  
 y = "Prop. true positives",  
 x = "Prop. false positives") +  
 theme\_bw()   
roc\_plot2

## Warning in verify\_d(data$d): D not labeled 0/1, assuming Completed = 0 and  
## Withdrew = 1!



#Cauclating auc. Same as the other graph for LR. No improvement.  
calc\_auc(roc\_plot2)

## Warning in verify\_d(data$d): D not labeled 0/1, assuming Completed = 0 and  
## Withdrew = 1!

## PANEL group AUC  
## 1 1 1 0.5

I decided to add score, which is the student’s score in this assessment to my predictors. My logic behind this selection was that a user’s score might influence their decision to drop or remain in a course. For example, a student who failed or was close to failing might decide to drop out preemptively. However, it does not appear that score is a predictor that improves the model as the AUC is still .5. That can’t be right. I’m looking at your code and it looks good, so I’m going to try to reproduce your code and figure out what might be the problem I decided to use an elastic net logistic regression because it was the most time effective and I wanted to test if it produced different results compared to my preferred decision tree. However, it did not.