## Dana Miller

### Random Forest Assignment

drug = read\_csv("drug\_data-1.csv")

##   
## -- Column specification --------------------------------------------------------  
## cols(  
## .default = col\_character(),  
## Column1 = col\_double(),  
## Column2 = col\_double(),  
## Column3 = col\_double(),  
## Column4 = col\_double(),  
## Column5 = col\_double(),  
## Column6 = col\_double(),  
## Column7 = col\_double(),  
## Column8 = col\_double(),  
## Column9 = col\_double(),  
## Column10 = col\_double(),  
## Column11 = col\_double(),  
## Column12 = col\_double(),  
## Column13 = col\_double()  
## )  
## i Use `spec()` for the full column specifications.

names(drug) =c("ID", "Age", "Gender", "Education", "Country", "Ethnicity","Nscore", "Escore", "Oscore", "Ascore", "Cscore", "Impulsive","SS", "Alcohol", "Amphet", "Amyl", "Benzos", "Caff",  
 "Cannabis","Choc", "Coke", "Crack", "Ecstasy", "Heroin", "Ketamine", "Legalh","LSD", "Meth", "Mushrooms", "Nicotine", "Semer", "VSA")  
# str(drug)

drug[drug=="CL0"] = "No"  
drug[drug=="CL1"] = "No"  
drug[drug=="CL2"] = "Yes"  
drug[drug=="CL3"] = "Yes"  
drug[drug=="CL4"] = "Yes"  
drug[drug=="CL5"] = "Yes"  
drug[drug=="CL6"] = "Yes"

drug\_clean = drug%>% mutate\_at(vars(Age:Ethnicity),funs(as\_factor))%>%  
 mutate(Age =factor(Age, labels =c("18\_24", "25\_34", "35\_44",  
 "45\_54", "55\_64", "65\_")))%>%  
 mutate(Gender =factor(Gender, labels =c("Male", "Female")))%>%  
 mutate(Education =factor(Education, labels =  
 c("Under16", "At16", "At17", "At18", "SomeCollege",  
 "ProfessionalCert", "Bachelors", "Masters", "Doctorate")))%>%  
 mutate(Country =factor(Country,  
 labels =c("USA", "NewZealand", "Other", "Australia",  
 "Ireland","Canada","UK")))%>%  
 mutate(Ethnicity =factor(Ethnicity,  
 labels =c("Black", "Asian", "White", "White/Black", "Other",  
 "White/Asian", "Black/Asian")))%>%  
 mutate\_at(vars(Alcohol:VSA),funs(as\_factor))%>%  
 select(-ID)

## Warning: `funs()` is deprecated as of dplyr 0.8.0.  
## Please use a list of either functions or lambdas:   
##   
## # Simple named list:   
## list(mean = mean, median = median)  
##   
## # Auto named with `tibble::lst()`:   
## tibble::lst(mean, median)  
##   
## # Using lambdas  
## list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_warnings()` to see where this warning was generated.

# str(drug\_clean)

drug\_clean = drug\_clean%>% select(!(Alcohol:Mushrooms))%>% select(!(Semer:VSA))  
names(drug\_clean)

## [1] "Age" "Gender" "Education" "Country" "Ethnicity" "Nscore"   
## [7] "Escore" "Oscore" "Ascore" "Cscore" "Impulsive" "SS"   
## [13] "Nicotine"

**Task 1**

skim(drug\_clean)

Data summary

|  |  |
| --- | --- |
| Name | drug\_clean |
| Number of rows | 1885 |
| Number of columns | 13 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Column type frequency: |  |
| factor | 6 |
| numeric | 7 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Group variables | None |

**Variable type: factor**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| skim\_variable | n\_missing | complete\_rate | ordered | n\_unique | top\_counts |
| Age | 0 | 1 | FALSE | 6 | 18\_: 643, 25\_: 481, 35\_: 356, 45\_: 294 |
| Gender | 0 | 1 | FALSE | 2 | Mal: 943, Fem: 942 |
| Education | 0 | 1 | FALSE | 9 | Som: 506, Bac: 480, Mas: 283, Pro: 270 |
| Country | 0 | 1 | FALSE | 7 | UK: 1044, USA: 557, Oth: 118, Can: 87 |
| Ethnicity | 0 | 1 | FALSE | 7 | Whi: 1720, Oth: 63, Bla: 33, Asi: 26 |
| Nicotine | 0 | 1 | FALSE | 2 | Yes: 1264, No: 621 |

**Variable type: numeric**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| skim\_variable | n\_missing | complete\_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
| Nscore | 0 | 1 | 0.00 | 1.00 | -3.46 | -0.68 | 0.04 | 0.63 | 3.27 | ▁▃▇▅▁ |
| Escore | 0 | 1 | 0.00 | 1.00 | -3.27 | -0.70 | 0.00 | 0.64 | 3.27 | ▁▃▇▃▁ |
| Oscore | 0 | 1 | 0.00 | 1.00 | -3.27 | -0.72 | -0.02 | 0.72 | 2.90 | ▁▃▇▆▁ |
| Ascore | 0 | 1 | 0.00 | 1.00 | -3.46 | -0.61 | -0.02 | 0.76 | 3.46 | ▁▃▇▃▁ |
| Cscore | 0 | 1 | 0.00 | 1.00 | -3.46 | -0.65 | -0.01 | 0.58 | 3.46 | ▁▃▇▃▁ |
| Impulsive | 0 | 1 | 0.01 | 0.95 | -2.56 | -0.71 | -0.22 | 0.53 | 2.90 | ▁▆▇▃▁ |
| SS | 0 | 1 | 0.00 | 0.96 | -2.08 | -0.53 | 0.08 | 0.77 | 1.92 | ▂▇▇▇▅ |

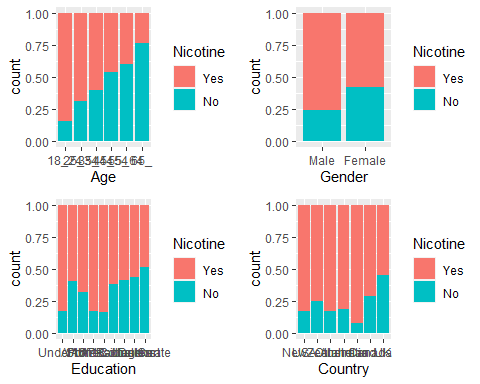
**There are no missing data in the drug\_clean dataframe.**

**Task 2**

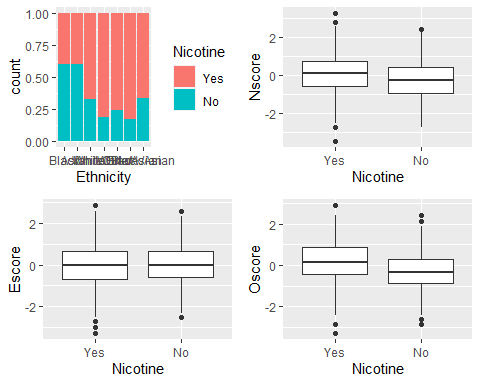
set.seed(1234)  
drug\_split = initial\_split(drug\_clean, prop = 0.7, strata = Nicotine)  
train = training(drug\_split)  
test = testing(drug\_split)

**Task 3**

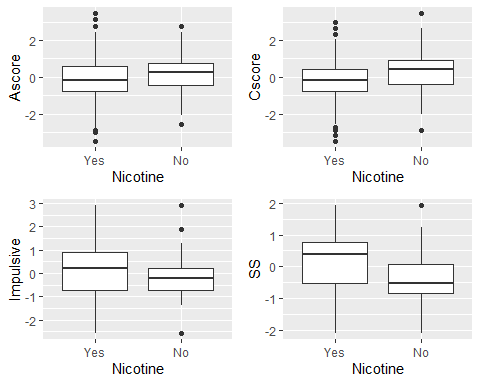
p1 = ggplot(train, aes(x= Age, fill= Nicotine)) + geom\_bar(position = "fill")  
p2 = ggplot(train, aes(x= Gender, fill= Nicotine)) + geom\_bar(position = "fill")  
p3 = ggplot(train, aes(x= Education, fill= Nicotine)) + geom\_bar(position = "fill")  
p4 = ggplot(train, aes(x= Country, fill= Nicotine)) + geom\_bar(position = "fill")  
grid.arrange(p1,p2,p3,p4)



p1 = ggplot(train, aes(x= Ethnicity, fill= Nicotine)) + geom\_bar(position = "fill")  
p2 = ggplot(train, aes(x= Nicotine, y= Nscore)) + geom\_boxplot()  
p3 = ggplot(train, aes(x= Nicotine, y= Escore)) + geom\_boxplot()  
p4 = ggplot(train, aes(x= Nicotine, y= Oscore)) + geom\_boxplot()  
grid.arrange(p1,p2,p3,p4)



p1 = ggplot(train, aes(x= Nicotine, y= Ascore)) + geom\_boxplot()  
p2 = ggplot(train, aes(x= Nicotine, y= Cscore)) + geom\_boxplot()  
p3 = ggplot(train, aes(x= Nicotine, y= Impulsive)) + geom\_boxplot()  
p4 = ggplot(train, aes(x= Nicotine, y= SS)) + geom\_boxplot()  
grid.arrange(p1,p2,p3,p4)



**All of the factor variables (Age, Gender, Education, Country and Ethnicity) seem to be predictos of Nicotine. Where Nscore, Escore, Oscore, Ascore and Cscore have a very similar variation and do not seem to be as stong of predictors. Impulsive and SS also seem to be predictors.**

**Task 4**

set.seed(123)  
rf\_folds = vfold\_cv(train, v=5)

drug\_recipe = recipe(Nicotine ~ ., train)%>%  
 step\_dummy(all\_nominal(), -all\_outcomes())  
  
rf\_model = rand\_forest(mtry = tune(), min\_n = tune(), trees = 100)%>%  
 set\_engine("ranger", importance = "permutation") %>%  
 set\_mode("classification")  
  
drug\_wflow =   
 workflow() %>%  
 add\_model(rf\_model)%>%  
 add\_recipe(drug\_recipe)  
  
rf\_grid = grid\_regular(  
 mtry(range = c(2,8)),  
 min\_n(range = c(2,20)),  
 levels = 10  
)  
  
set.seed(123)  
rf\_res\_tuned = tune\_grid(  
 drug\_wflow,  
 resamples = rf\_folds,  
 grid = rf\_grid  
)

##   
## Attaching package: 'rlang'

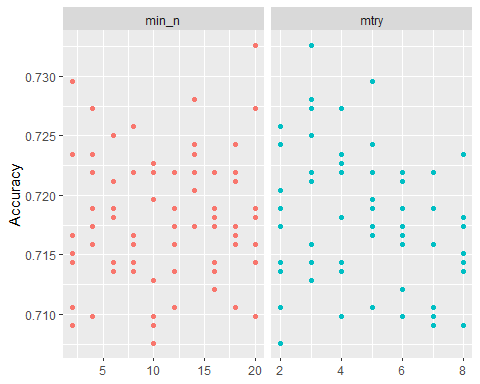
## The following objects are masked from 'package:purrr':  
##   
## %@%, as\_function, flatten, flatten\_chr, flatten\_dbl, flatten\_int,  
## flatten\_lgl, flatten\_raw, invoke, list\_along, modify, prepend,  
## splice

##   
## Attaching package: 'vctrs'

## The following object is masked from 'package:dplyr':  
##   
## data\_frame

## The following object is masked from 'package:tibble':  
##   
## data\_frame

rf\_res\_tuned %>%  
 collect\_metrics() %>%  
 filter(.metric == "accuracy") %>%  
 select(mean, min\_n, mtry) %>%  
 pivot\_longer(min\_n:mtry,  
 values\_to = "value",  
 names\_to = "parameter"  
 ) %>%  
 ggplot(aes(value, mean, color = parameter))+  
 geom\_point(show.legend = FALSE) +  
 facet\_wrap(~parameter, scales = "free\_x") +  
 labs(x=NULL, y= "Accuracy")



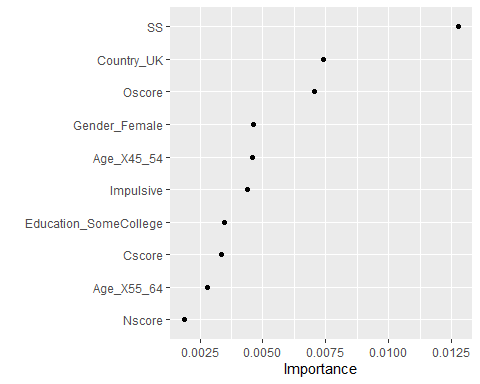
**Task 5**

best\_rf = select\_best(rf\_res\_tuned, "accuracy")  
  
final\_rf = finalize\_workflow(  
 drug\_wflow,  
 best\_rf  
)  
  
final\_rf

## == Workflow ====================================================================  
## Preprocessor: Recipe  
## Model: rand\_forest()  
##   
## -- Preprocessor ----------------------------------------------------------------  
## 1 Recipe Step  
##   
## \* step\_dummy()  
##   
## -- Model -----------------------------------------------------------------------  
## Random Forest Model Specification (classification)  
##   
## Main Arguments:  
## mtry = 3  
## trees = 100  
## min\_n = 20  
##   
## Engine-Specific Arguments:  
## importance = permutation  
##   
## Computational engine: ranger

final\_rf\_fit = fit(final\_rf, train)

final\_rf\_fit %>% pull\_workflow\_fit() %>% vip(geom = "point")



**The most important variables are SS (sensation seeking), the UK as described in the Country variable and Oscore (openness to experience). This shoes that someone who has a high sensation seeking score, lives in the UK and has a high openness to experience score will have a higherchance of using nicotine.**

**Task 6**

trainpredrf = predict(final\_rf\_fit, train)  
#head(trainpredrf)

confusionMatrix(trainpredrf$.pred\_class, train$Nicotine,positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 850 171  
## No 35 264  
##   
## Accuracy : 0.8439   
## 95% CI : (0.8232, 0.8631)  
## No Information Rate : 0.6705   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.6163   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.9605   
## Specificity : 0.6069   
## Pos Pred Value : 0.8325   
## Neg Pred Value : 0.8829   
## Prevalence : 0.6705   
## Detection Rate : 0.6439   
## Detection Prevalence : 0.7735   
## Balanced Accuracy : 0.7837   
##   
## 'Positive' Class : Yes   
##

testpredrf = predict(final\_rf\_fit, test)  
#head(testpredrf)

confusionMatrix(testpredrf$.pred\_class, test$Nicotine, positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 337 121  
## No 42 65  
##   
## Accuracy : 0.7115   
## 95% CI : (0.6722, 0.7486)  
## No Information Rate : 0.6708   
## P-Value [Acc > NIR] : 0.02111   
##   
## Kappa : 0.2676   
##   
## Mcnemar's Test P-Value : 9.999e-10   
##   
## Sensitivity : 0.8892   
## Specificity : 0.3495   
## Pos Pred Value : 0.7358   
## Neg Pred Value : 0.6075   
## Prevalence : 0.6708   
## Detection Rate : 0.5965   
## Detection Prevalence : 0.8106   
## Balanced Accuracy : 0.6193   
##   
## 'Positive' Class : Yes   
##

**The accuracy on the training set is about 84%, while the testing set has an accuracy of about 71%. Since there is big difference in between the two, the model might not be the best to use on data it hasn’t seen before.**

**Task 7**

**With this data and model, you could use it as a marketing tool for tobaco/vape companies. The model can help narrow down age, gender and what country of people to market to. The model could be used for marketing because ads can still be generalized, but geared towards individuals. If it was used for something more specific I might not use the model based off of the low accuracy on testing set. It might not be the best model to use since there was a significant difference between the training and testing.**