DSP 441 Group Project

Mark Rogers, Audri Casey, Ryan Viti

4/20/2020

# Introduction

Our group decided to take a look at what factors play in college students receiving their GPA. In order to collect data, we created a Google Forms survey and shared it amongst our peers and within our campus organizations. We received a great deal of responses, and with real world data we hope to gain some insight as to what influences students receiving high, or low, GPAs, and other insight into college students as a whole. To do this, we will form different hypotheses, and perform respective analysis approaches to draw conclusions from our data. Our first question of interest is; can we create a classification tree that is able to accurately predict students’ GPA letter grade? Our second question is; can we use GPA and Hours\_Work to determine if students are in a study group? And finally, can a multi-class support vector machine accurately predict students’ GPA letter grade? # Descriptive Statistics

To start, we will show the names of our variables and the head of the data set.

## [1] "Gender" "Age" "Major" "Double\_Major"   
## [5] "Minor" "Academic\_Level" "Student\_Status" "Grade"   
## [9] "GPA" "Hours\_Studying" "Study\_Location" "Study\_Group"   
## [13] "Housing1" "Housing2" "Home\_State" "Hours\_Sleep"   
## [17] "Post\_Grad" "Campus\_Org" "Employed" "Hours\_Work"

## Gender Age Major Double\_Major Minor Academic\_Level Student\_Status Grade  
## 1 F 22 KIN N 0 4 Full-time B+  
## 2 F 21 PRS Y 1 4 Full-time B-  
## 3 M 22 CSC N 0 4 Full-time B+  
## 4 F 22 PHP N 0 4 Full-time B  
## 5 M 21 COM N 1 4 Full-time B-  
## 6 M 22 BUS N 0 4 Full-time B  
## GPA Hours\_Studying Study\_Location Study\_Group Housing1 Housing2  
## 1 3.50 4 Library Alone Off Friends  
## 2 2.70 10 Library Alone Off Friends  
## 3 3.50 10 Classroom Group Off Friends  
## 4 3.20 20 Home Alone Off Friends  
## 5 2.90 6 Library Group Off Friends  
## 6 3.21 5 Home Alone Off Friends  
## Home\_State Hours\_Sleep Post\_Grad Campus\_Org Employed Hours\_Work  
## 1 NJ 7.5 Grad school 3 Y 15  
## 2 PA 6.0 Work 1 Y 10  
## 3 RI 8.0 Work 1 Y 10  
## 4 RI 6.0 Work 3 Y 15  
## 5 RI 5.0 Work 1 Y 8  
## 6 RI 9.0 Work 1 Y 6

Before we could get to work on our analysis, we needed to do some data processing in order to perform our analysis approaches. The first thing we did was clean up our .csv file and remove the null values and cleaned up our survey answers. For example, we received different forms of the same answer on our survey, such as “RI” and “Rhode Island” as home state, so we normalized all these by hand in Excel. The second thing we needed to do was correct the variables that have factors of only two levels, which will give us an error when making our models. We corrected this by creating dummy variables, 0s and 1s, for variables with only two factor levels. These variables were “Double\_Major”, “Student\_Status”, “Study\_Group”, “Housing1”, and “Employed”. Below you can see the resulting tables of values for each variable after using the ifelse() function for the correction.

## [1] "Double\_Major -> 0 = Single Major ; 1 = Double Major"

##   
## 0 1   
## 82 8

## [1] "Student\_Status -> 0 = Full-time ; 1 = Part-time"

##   
## 0 1   
## 88 2

## [1] "Study\_Group -> 0 = Studies alone ; 1 = Studies in a group"

##   
## 0 1   
## 71 19

## [1] "Housing1 -> 0 = Lives off campus ; 1 = Lives on campus"

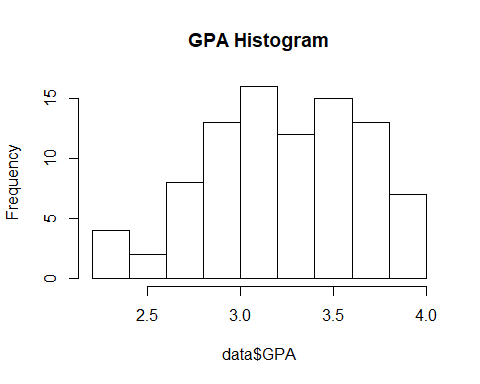
##   
## 0 1   
## 47 43

## [1] "Employed -> 0 = Employed ; 1 = Unemployed"

##   
## 0 1   
## 46 44

Here we take a look at the values and value counts of our main response variable, GPA. We will also plot these values of GPA with a histogram as shown below.

##   
## 2.2 2.3 2.51 2.6 2.7 2.76 2.8 2.83 2.85 2.87 2.9 2.94 2.95 2.99 3   
## 1 3 1 1 3 1 4 1 1 1 5 1 1 1 2   
## 3.01 3.02 3.06 3.1 3.12 3.19 3.2 3.21 3.27 3.3 3.34 3.35 3.38 3.4 3.41   
## 1 1 1 5 1 2 5 1 1 3 1 1 1 4 1   
## 3.43 3.49 3.5 3.53 3.55 3.6 3.63 3.65 3.67 3.7 3.72 3.75 3.78 3.8 3.85   
## 2 2 2 1 1 6 1 1 2 2 2 1 1 3 2   
## 3.87 3.89 3.91 3.94 3.95   
## 1 1 1 1 1



Now we output the maximum, minimum, and median of GPA.

max(data$GPA)

## [1] 3.95

min(data$GPA)

## [1] 2.2

median(data$GPA)

## [1] 3.285

Also, we will look at the mean value, variance, and standard deviation of GPA.

mean(data$GPA)

## [1] 3.254667

var(data$GPA)

## [1] 0.176742

sd(data$GPA)

## [1] 0.420407

## Analysis

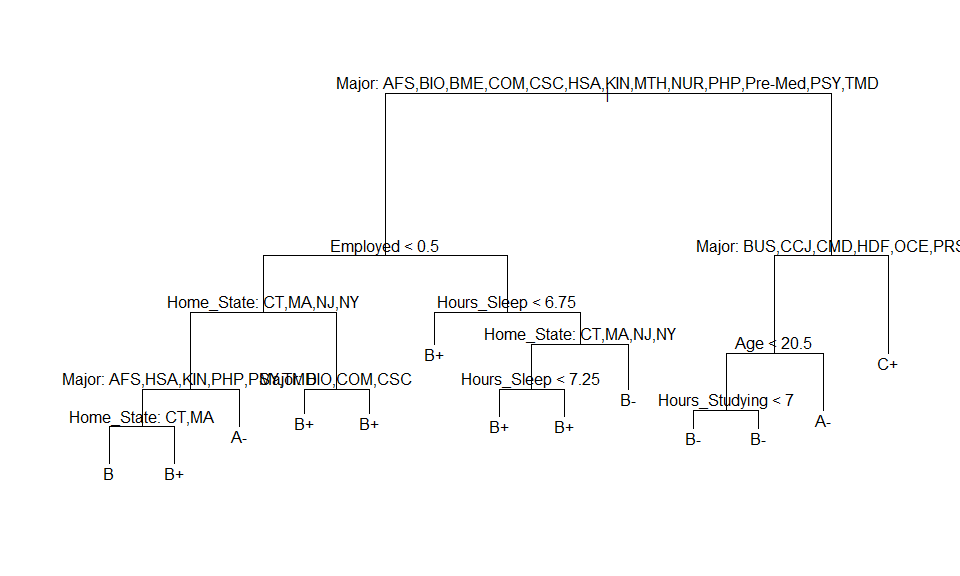
#### First Analysis: Classification Tree

The response variable in this case is the categorical variable “Grade” which equates students’ numerical GPA to a letter grade A, A-, B+, B, and so on). The variables used in the construction of the tree are the student’s major (categorical), number of hours spent studying (numerical). The first type of analysis we did is a classification tree which is used to predict categorical variables. We chose to use a classification tree because the relationship between many of the predictors is non-linear which was seen using the pairs() function on our data set.

We proceed with the analysis under the hypothesis that a classification tree can accurately predict GPA using at least one terminal node. We split the data into training and test data.

set.seed(2020)  
tree\_train = sample(90, 81)  
data.tree = tree(Grade ~ .-GPA, data, subset = tree\_train)  
summary(data.tree)

##   
## Classification tree:  
## tree(formula = Grade ~ . - GPA, data = data, subset = tree\_train)  
## Variables actually used in tree construction:  
## [1] "Major" "Employed" "Home\_State" "Hours\_Sleep"   
## [5] "Age" "Hours\_Studying"  
## Number of terminal nodes: 13   
## Residual mean deviance: 1.636 = 111.3 / 68   
## Misclassification error rate: 0.3333 = 27 / 81



set.seed(2020)  
tree\_test=data[-tree\_train,]  
Grade.test=data$Grade[-tree\_train]  
tree.pred=predict(data.tree,tree\_test,type="class")  
#table(tree.pred,Grade.test)  
table(tree.pred, data[-tree\_train, "Grade"])

##   
## tree.pred A- B B- B+ C C+  
## A- 0 1 0 1 0 0  
## B 0 0 0 0 0 0  
## B- 0 2 0 1 0 0  
## B+ 1 0 3 0 0 0  
## C 0 0 0 0 0 0  
## C+ 0 0 0 0 0 0

The tree correctly classifies only 22.22% of the observations. We pruned the tree once and performed cross-validation.

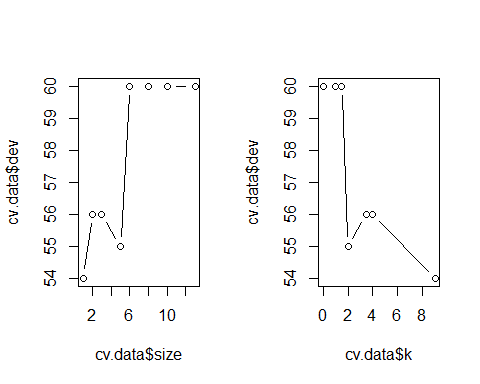
cv.data=cv.tree(data.tree, FUN=prune.misclass)  
names(cv.data)

## [1] "size" "dev" "k" "method"

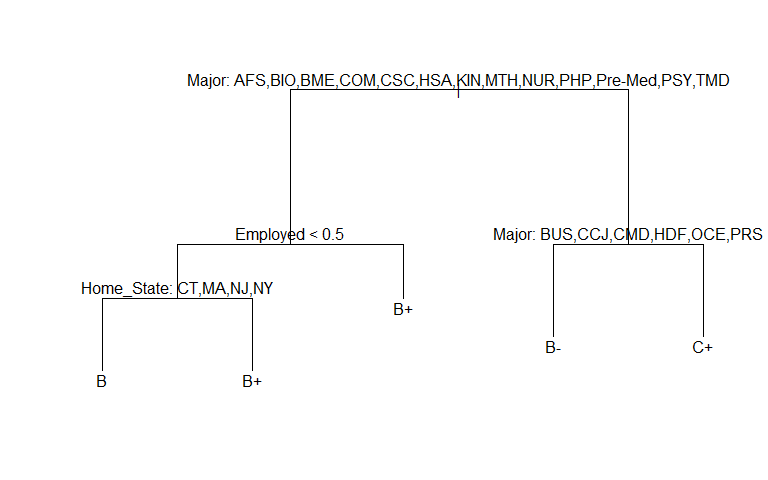
cv.data

## $size  
## [1] 13 10 8 6 5 3 2 1  
##   
## $dev  
## [1] 60 60 60 60 55 56 56 54  
##   
## $k  
## [1] -Inf 0.0 1.0 1.5 2.0 3.5 4.0 9.0  
##   
## $method  
## [1] "misclass"  
##   
## attr(,"class")  
## [1] "prune" "tree.sequence"

par(mfrow=c(1,2))  
plot(cv.data$size, cv.data$dev, type="b")  
plot(cv.data$k, cv.data$dev, type="b")



prune.data=prune.misclass(data.tree,best=5)  
plot(prune.data)  
text(prune.data,pretty=0)



tree.pred=predict(prune.data, tree\_test, type="class")  
table(tree.pred, Grade.test)

## Grade.test  
## tree.pred A- B B- B+ C C+  
## A- 0 0 0 0 0 0  
## B 0 1 0 1 0 0  
## B- 0 2 0 1 0 0  
## B+ 1 0 3 0 0 0  
## C 0 0 0 0 0 0  
## C+ 0 0 0 0 0 0

(1+3+1+3+4+4)/18

## [1] 0.8888889

Proceeding with a value of best=3, we correctly classify 31.11% of the observations. While the pruned tree provides more significant results than the original tree, we can conclude that a classification tree does not perform the best in accurately predicting a student’s Grade based on major and the number of hours spent studying on average per week.

##### Random Forest

library(randomForest)

## Warning: package 'randomForest' was built under R version 3.6.3

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

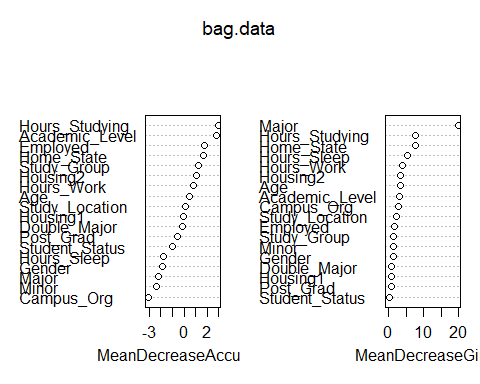
set.seed(2020)  
bag.data = randomForest(Grade ~ .-GPA, data = data, mtry = 9, importance = TRUE)  
bag.data

##   
## Call:  
## randomForest(formula = Grade ~ . - GPA, data = data, mtry = 9, importance = TRUE)   
## Type of random forest: classification  
## Number of trees: 500  
## No. of variables tried at each split: 9  
##   
## OOB estimate of error rate: 71.11%  
## Confusion matrix:  
## A- B B- B+ C C+ class.error  
## A- 1 4 4 7 0 0 0.9375000  
## B 2 4 4 10 0 0 0.8000000  
## B- 2 3 6 8 0 0 0.6842105  
## B+ 4 6 4 15 0 0 0.4827586  
## C 0 1 0 0 0 0 1.0000000  
## C+ 1 2 1 1 0 0 1.0000000

importance(bag.data)

## A- B B- B+ C C+  
## Gender -2.2733651 -2.6306592 1.0389946 -0.67064429 0 -1.1182766  
## Age -2.3265702 0.3752057 1.6484769 0.35115783 0 0.5088610  
## Major -2.7104178 -2.8591326 0.2158087 -0.50536911 0 1.8471197  
## Double\_Major 0.9549125 0.9862434 -1.3832011 0.03734252 0 1.0010015  
## Minor -2.2607265 -1.2160230 0.4146555 -0.77139960 0 0.0000000  
## Academic\_Level -1.9083909 0.3476794 1.7875676 3.75709919 0 0.7929038  
## Student\_Status -1.0010015 0.0000000 0.0000000 0.00000000 0 0.0000000  
## Hours\_Studying 2.6585268 -2.8471199 2.6832398 2.07342159 0 0.7614836  
## Study\_Location -2.3156687 -2.6299538 1.5655113 2.04729699 0 1.7105892  
## Study\_Group -0.5806842 0.8942954 0.8955485 0.56960107 0 1.0010015  
## Housing1 -2.2968159 -0.2448133 -0.1491637 0.43381805 0 1.5109947  
## Housing2 0.1295584 0.1294111 -0.6184879 1.70964392 0 0.7824408  
## Home\_State -0.1142037 -0.2931106 -1.2979300 3.24384526 0 -0.3615324  
## Hours\_Sleep 0.2090927 -0.4286311 1.6223278 -3.34448196 0 -0.5890057  
## Post\_Grad -1.0956529 1.3294657 -1.0513443 -0.31082375 0 1.0010015  
## Campus\_Org -1.3730530 1.6625320 -1.4171810 -3.32932203 0 -0.8170415  
## Employed -3.7981545 4.8390961 2.3447101 -0.25918846 0 -1.5499079  
## Hours\_Work -3.3340846 3.7810094 2.8387800 -1.19473200 0 -0.3562805  
## MeanDecreaseAccuracy MeanDecreaseGini  
## Gender -1.82890953 1.2725533  
## Age 0.46744516 3.4595616  
## Major -2.21549931 19.7562488  
## Double\_Major -0.10711309 0.8940008  
## Minor -2.36089735 1.3314399  
## Academic\_Level 2.75718773 3.1423006  
## Student\_Status -1.00100150 0.2662428  
## Hours\_Studying 2.91715692 7.7837331  
## Study\_Location 0.12005839 2.3700114  
## Study\_Group 1.19755425 1.3749819  
## Housing1 -0.05042885 0.8398075  
## Housing2 1.09505966 3.5077435  
## Home\_State 1.65262647 7.7404115  
## Hours\_Sleep -1.79539379 5.3637108  
## Post\_Grad -0.61159305 0.7395331  
## Campus\_Org -3.06299189 2.9547126  
## Employed 1.76487043 1.6475664  
## Hours\_Work 0.83116617 3.8756405

varImpPlot(bag.data)

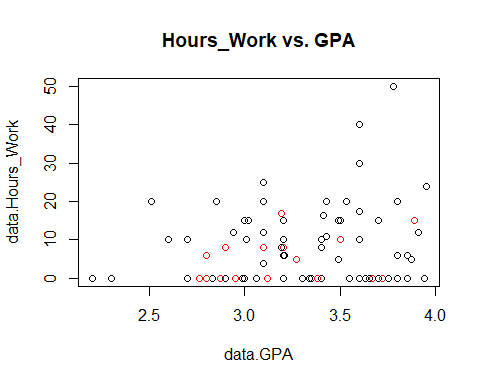


#### Second Analysis: Two-Class Support Vector Machine

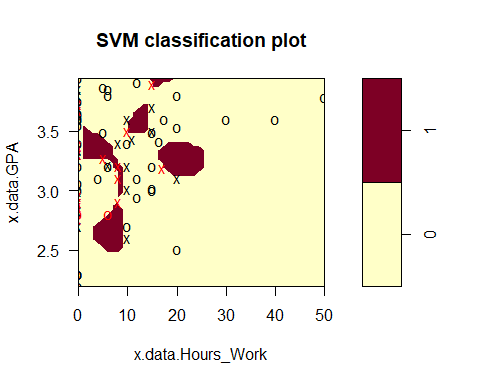
For our initial SVM, we use our binary variable, Study\_Group as our response, where a 1 indicates a given student is part of a study group, and a 0 indicates otherwise. Our explanatory variables for this analysis are GPA and Hours\_Work. GPA is a continuous, numeric variable indicating a student’s GPA as of Fall 2019. Hours\_Work indicates on average how many hours a student works at a job (full-time or part-time), which is a discrete, numeric variable. The Support Vector Machine is our approach to this classification problem. This is a relevant approach because our hypothesis looks to determine a binary response variable based on two explanatory variables related to that response variable. Some possible limitations to this approach include the fact that one of our explanatory variables is not continuous, which creates a distorted picture of the traditional SVM plot, noting clear gaps in one dimension of our explanatory variable plot. A small sample of respondents is another potential issue, but we continue since there are only so many resources available to us. Our hypothesis is stated as: A student’s GPA and the number of hours of work they get each night can determine if they are likely to be a part of a study group on campus. Since our classification boundaries were not linear, we are inclined to utilize a radial kernel for our SVM as opposed to the default, linear kernel. This will most likely give us a more appropriate output and more accurate classification rate. We could have chosen Logistic Regression as our model of choice here as well, but as was discussed in class, we should utilize SVM when our decision boundaries appear to be nonlinear. Also, the fact that we have classes that are nearly separable, using SVM is the optimal choice.

## [1] "data.frame"

## [1] "numeric"



##   
## Call:  
## svm(formula = y ~ ., data = svm\_dat[svm\_train, ], kernel = "radial",   
## gamma = 1, cost = 1e+05)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: radial   
## cost: 1e+05   
##   
## Number of Support Vectors: 41  
##   
## ( 25 16 )  
##   
##   
## Number of Classes: 2   
##   
## Levels:   
## 0 1



##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost gamma  
## 0.1 0.01  
##   
## - best performance: 0.225   
##   
## - Detailed performance results:  
## cost gamma error dispersion  
## 1 1e-01 0.01 0.2250000 0.1645701  
## 2 1e+00 0.01 0.2250000 0.1645701  
## 3 1e+01 0.01 0.2250000 0.1645701  
## 4 1e+02 0.01 0.2250000 0.1645701  
## 5 1e+03 0.01 0.2250000 0.1645701  
## 6 1e+04 0.01 0.2250000 0.1645701  
## 7 1e-01 0.10 0.2250000 0.1645701  
## 8 1e+00 0.10 0.2250000 0.1645701  
## 9 1e+01 0.10 0.2250000 0.1645701  
## 10 1e+02 0.10 0.2250000 0.1645701  
## 11 1e+03 0.10 0.2500000 0.1666667  
## 12 1e+04 0.10 0.2986111 0.1598731  
## 13 1e-01 1.00 0.2250000 0.1645701  
## 14 1e+00 1.00 0.2250000 0.1645701  
## 15 1e+01 1.00 0.3111111 0.1605354  
## 16 1e+02 1.00 0.2875000 0.1671867  
## 17 1e+03 1.00 0.2361111 0.1723566  
## 18 1e+04 1.00 0.2722222 0.1158703  
## 19 1e-01 10.00 0.2250000 0.1645701  
## 20 1e+00 10.00 0.2500000 0.1666667  
## 21 1e+01 10.00 0.2847222 0.1459986  
## 22 1e+02 10.00 0.3097222 0.1805021  
## 23 1e+03 10.00 0.3847222 0.2260521  
## 24 1e+04 10.00 0.3625000 0.2531057

## cost gamma  
## 1 0.1 0.01

table(true=svm\_dat[-svm\_train,"y"], pred=predict(tune.out$best.model,newdata = svm\_dat[-svm\_train,]))

## pred  
## true 0 1  
## 0 8 0  
## 1 1 0

1/9 #based on table above

## [1] 0.1111111

11.11% of test observations are misclassified by our SVM.

Therefore, we see that based on our dataset training and testing data for the SVM, our model incurs a 22.22% misclassification rate, meaning we classify correctly 87.88% of the time. Our optimal cost and gamma parameters came out to be 0.1 and 0.01, according to tuning function. In general, this seems like a strong model to use for future prediction.

#### Analysis Three: Multi-Class Support Vector Machine

For our third analysis, we will use a multi-class support vector machine to look into classifying students’ GPAs. Since the value of k would be too large if we used each individual value of GPA as a class, we decided to create a new column “Grade” that is the letter grade corresponding to the GPA value. This means “Grade” is our response variable for this analysis, with k equalling 6 since the column covers grades A-, A, B+, B, B-, and C+ (no one who took our survey had a 4.0 GPA or less than a 2.3 GPA). The explanatory variables are Age (in years), Double\_Major (yes/no), Minor (number of minors), Academic\_Level (grade year), Hours\_Studying (in hours), Study\_Group (yes/no), Housing1 (on-campus/off-campus), Hours\_Sleep (in hours), Campus\_Org (yes/no), Employed (yes/no), and Hours\_Work (in hours). As stated in the descriptive statistics section, we encoded yes/no answers into 0s and 1s. When performing the svm() function, we will use a one-versus-one approach, which is the default for the function. Here we decided to use SVM and not Logistic regression because our classes are nearly separable, and contain nonlinear boundaries. Our hypothesis is as follows: Using the predictors above, we can accurately predict a student’s grade.

set.seed(2020)  
msvm\_train = sample(90, 81)  
svm.fit = svm(b ~ ., data = df[msvm\_train,], kernel = "radial", cost = 1e5, gamma = 1)  
summary(svm.fit)

##   
## Call:  
## svm(formula = b ~ ., data = df[msvm\_train, ], kernel = "radial",   
## cost = 1e+05, gamma = 1)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: radial   
## cost: 1e+05   
##   
## Number of Support Vectors: 81  
##   
## ( 27 17 15 16 5 1 )  
##   
##   
## Number of Classes: 6   
##   
## Levels:   
## A- B B- B+ C C+

##   
## A- B B- B+ C C+  
## A- 15 0 0 0 0 0  
## B 0 17 0 0 0 0  
## B- 0 0 16 0 0 0  
## B+ 0 0 0 27 0 0  
## C 0 0 0 0 1 0  
## C+ 0 0 0 0 0 5

Here we see our SVM model performs well with the training data. Now we perform cross validation to find the optimal values for cost and gamma.

set.seed(2020)  
tune.out = tune(svm, b ~ ., data = df[msvm\_train,], kernel = "radial",  
 ranges = list(cost = c(0.1,1,10,100,1000,10000), gamma = c(0.5,1,2,3,4,5,6)))  
summary(tune.out)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost gamma  
## 1 0.5  
##   
## - best performance: 0.6444444   
##   
## - Detailed performance results:  
## cost gamma error dispersion  
## 1 1e-01 0.5 0.6708333 0.1958777  
## 2 1e+00 0.5 0.6444444 0.2314074  
## 3 1e+01 0.5 0.6583333 0.2113668  
## 4 1e+02 0.5 0.6583333 0.2113668  
## 5 1e+03 0.5 0.6583333 0.2113668  
## 6 1e+04 0.5 0.6583333 0.2113668  
## 7 1e-01 1.0 0.6708333 0.1958777  
## 8 1e+00 1.0 0.6458333 0.1916969  
## 9 1e+01 1.0 0.6472222 0.2670137  
## 10 1e+02 1.0 0.6472222 0.2670137  
## 11 1e+03 1.0 0.6472222 0.2670137  
## 12 1e+04 1.0 0.6472222 0.2670137  
## 13 1e-01 2.0 0.6708333 0.1958777  
## 14 1e+00 2.0 0.6708333 0.1958777  
## 15 1e+01 2.0 0.6458333 0.1916969  
## 16 1e+02 2.0 0.6458333 0.1916969  
## 17 1e+03 2.0 0.6458333 0.1916969  
## 18 1e+04 2.0 0.6458333 0.1916969  
## 19 1e-01 3.0 0.6708333 0.1958777  
## 20 1e+00 3.0 0.6708333 0.1958777  
## 21 1e+01 3.0 0.6708333 0.1958777  
## 22 1e+02 3.0 0.6708333 0.1958777  
## 23 1e+03 3.0 0.6708333 0.1958777  
## 24 1e+04 3.0 0.6708333 0.1958777  
## 25 1e-01 4.0 0.6708333 0.1958777  
## 26 1e+00 4.0 0.6708333 0.1958777  
## 27 1e+01 4.0 0.6708333 0.1958777  
## 28 1e+02 4.0 0.6708333 0.1958777  
## 29 1e+03 4.0 0.6708333 0.1958777  
## 30 1e+04 4.0 0.6708333 0.1958777  
## 31 1e-01 5.0 0.6708333 0.1958777  
## 32 1e+00 5.0 0.6708333 0.1958777  
## 33 1e+01 5.0 0.6708333 0.1958777  
## 34 1e+02 5.0 0.6708333 0.1958777  
## 35 1e+03 5.0 0.6708333 0.1958777  
## 36 1e+04 5.0 0.6708333 0.1958777  
## 37 1e-01 6.0 0.6708333 0.1958777  
## 38 1e+00 6.0 0.6708333 0.1958777  
## 39 1e+01 6.0 0.6708333 0.1958777  
## 40 1e+02 6.0 0.6708333 0.1958777  
## 41 1e+03 6.0 0.6708333 0.1958777  
## 42 1e+04 6.0 0.6708333 0.1958777

With the optimal values of cost and gamma, we find the predictions with our test data.

## pred  
## true A- B B- B+ C C+  
## A- 0 0 0 1 0 0  
## B 0 0 0 3 0 0  
## B- 1 0 0 2 0 0  
## B+ 0 0 0 2 0 0  
## C 0 0 0 0 0 0  
## C+ 0 0 0 0 0 0

(1+3+2+1)/9

## [1] 0.7777778

66.67% of test observations are misclassified.

(1+1+1+12)/45

## [1] 0.3333333

33.33% of test observations are correctly classified. Based on our cross-validation tuning of our model, we found that the most optimal value of cost was 10 and gamma was 0.5. 66.67% error rate is rather high, which could hinder our hypothesis. However, if we were to pick one student at random, and try to guess their Grade value for their GPA, we would have 1/6 (16.67%) chance of getting it right. So our SVM model predicts a student’s Grade value for their GPA more accurately than picking at random. Given this understanding of our model’s performance, we say that our hypothesis is correct.

# Conclusion

In our project we shared a survey with our fellow students on the URI campus, and asked them to give us information about their education, work, living, and study habits. Our goal was to see what variables in students’ lives may determine what GPA they achieve. To do this we performed three analyses. Our primary response variables were GPA, the students’ numerical GPA, and Grade, the students’ letter value for their GPA. Our first analysis of a Classification Tree performed with 31.11% accuracy, which was a similar result to our third analysis. Our second analysis was an experimental hypothesis, in which we used a two-class Support Vector Machine to determine if we could classify students as being a part of a study group, or students who study alone, based on their GPA and weekly work hours. This analysis allowed us to view GPA as a predictor variable rather than a response, and provided great results with a 22.22% misclassification error rate. Our final analysis was a k=6 class Support Vector Machine to determine if we can classify students’ letter GPA grade based on an assortment of predictors. This analysis yielded similar results to our Classification Tree, in that our model correctly classified 33.33% of the test observations. We determined that our models were significant, in that randomly selecting a student and guessing their letter GPA grade has a 16.67% chance of being accurate, so our models performed better than randomly guessing. Of course, we could always have made changes in hindsight that would help us in further analysis.

Exploring our dataset for trends that we could isolate using the techniques we learned in class, we found that there might be other interesting relationships we might like to explore. We did observe that our dataset did not have perhaps the widest range of GPA’s, noting that between 3.1-3.6 was around the most prominent based on the GPA’s of our respondents. Sampling a larger number of college students could give us a better picture of how these variables interact. Additionally, if we could find ways of measuring intrinsic values of a given student via predetermined metrics of some sort, such as self-motivation, situational awareness, conscientiousness, grit, etc., we could perhaps uncover some interesting relationships between them and overall academic performance, which we measured using an interval of GPAs and assigning them letter grades, as mentioned previously. In summary, a larger sample size with more variables to choose from could improve the quality and accuracy of some of our models in predicting future academic performance.