Assessment of COMP 170

Annemarie Andaleon & Isabel Heard

5/4/2023

Abstract

The purpose of this study was to help the computer science department at Loyola University Chicago assess how to best understand how students are learning in their class and what factors most impact their performance. We based their performance on assigned Bloom's Taxonomy Scores. We explored their performance using a mixed-effects model with fixed effects of topic scores, semester, their interaction, and random effects of individual students, professors, and class sections. We ultimately found that the strongest year for students was in Fall 2020, and their weakest year was Fall 2022. All topics are positively correlated with each other. The greatest source of variance within the data comes from students, not section or professor.

Introduction

The goal of this project is to help the computer science department at Loyola University Chicago find a consistent way to assess students across all levels of the curriculum. We want to provide them with a way to standardize and store student's test scores across all years and sections. Within the scope of this project, we will be looking at the class COMP 170. COMP 170 is an introduction to object-oriented programming class taught in Java. Our objective is to see how students performed semester to semester. This will be done by assigning each student a Bloom's score for how well they performed in each of the top ten topic knowledge areas provided.

These top ten knowledge areas and learning outcomes were created by the computer science department for the COMP 170 class. Some of the topics include knowledge areas such as, "Use types and operators appropriately in expressions", or "Use arrays to organize information." All of these knowledge areas are associated with one or more questions on the

final exam. With this information we can use a student's score on a question to see how well they understand each top ten knowledge area.

We will be basing the students' understanding of these top ten topics by Bloom's Taxonomy. Bloom's Taxonomy was created by Benjamin Bloom in 1956, which was then later revised in 2001.[1] It works by categorizing the cognitive domain of learning into distinct levels according to "complexity and richness". [1] These domains include remembering, understanding, application, analysis, evaluation, and creating (being the most complex). With these domains in mind, the computer science department provided a numeric rating that applies to each skill category ranging from 1 to 5, seen in Appendix A, Table 1. A rating of 3 or higher indicates that a student successfully achieved a learning outcome. Bloom's framework is important because it allows professors to outline what they want to teach, which in turn, helps students get the most out of the class.

With Bloom's scores within each top ten knowledge area, we will use a mixed effects model in R to run analysis on the students, along with their section, semester, and professor. The hope is to give the computer science department a better understanding of which topics students are struggling with and which topics they are succeeding in.

Methods

Data Compilation and Cleaning

Description of data

The data assembly process began by collecting grades from the final exams in the semesters of Fall 2020, Spring 2021, Spring 2022, and Fall 2022. The data in these excel sheets gave us an overall score, what section they were in, how many points they received on each question, whether the code was defect free, and any comments the professor had about the test. They also included which top ten topic each question related to, and how many total points each question was worth. We were also given the Bloom's Taxonomy Scores and the top ten knowledge areas as a reference point during our work. Additionally, we thought it would be

[1] https://citt.ufl.edu/resources/the-learning-process/designing-the-learning-experience/blooms-taxonomy/

useful to know what professor taught each section, so we were able to find this information on the computer science department's website.[2]

Data Cleaning Procedures

With the data assembled, we were able to begin cleaning. Our first task was to create two different types of data sets. One was for each semester containing a unique ID for each student, what section they were in, their score for each question, and what professor they had (f20, s21, f22, s22). Within these data sets, we decided to remove anyone who ended with a zero in the class. We did this because all of the students who received a zero either cheated or withdrew, so we did not deem it necessary to assign them Bloom's score. We then created a second dataset named 'Key', containing the question number, the top ten knowledge area associated with that question, and the total amount of points possible for each question (f20Key, s21Key, f22Key, s22Key).

With these data sets, we were able to create a loop that matched each top ten knowledge area to a question. This was necessary to do as for many questions, there was more than one topic area referenced, or a knowledge area was repeated across multiple questions. For instance, say knowledge are 5 is seen in questions 2 and 7, which are both 10 points apiece. Then, knowledge is ten would be scored out of 20 points, and divided by the scores a student got on questions 2 and 7.

We also ran a PCA model to see if we needed to reduce the dimensionality of our data. We found that all our variables are statistically significant and explain a large amount of variance in the data. So, it is important that we keep all variables in our analysis so we can create a suitable model.

Assigning a Bloom's Score

Our next step was to assign a Bloom's score to each top ten knowledge area. For a given topic, if a student scored between a 100%- 90%, they received a 4, 90%- 80% a 3, 80% - 10% a 2, and 10% - 0% a 1. In this case we did not assign any students a 5, as this would come from their homework's, projects, and class participation, which we did not have access to. Within this data set we have the student's unique anonymous ID, the semester and section the professors that

[2] https://drive.google.com/drive/folders/1Xuhepv0Fh5kTruPc4lB-ECf55Ssaz43C

taught each section as well. We also included a unique key called SSP. For example, one instance of this key looks like “F20_s004_WH”, which has information about the semester, year, section, and professor. With this we were able to create a new data set where we pivoted longer. This meant that each student would each have 10 rows, one for each top ten knowledge are score they received. A long data set will make It easier for us to run our mixed effects model. The variables in this dataset “df_long2” can be seen in Table 2 in Appendix A.

Graphical Summaries

The first graphical summary we made after creating our long data frame was looking at the distribution of scores in a certain top ten topic by semester (Figure #1). With this figure, we are able to see those questions pertaining to knowledge area #1 was scored differently throughout the semesters. It looks like sometimes they were either given full points or none at all or given a wide array of points.

We were then curious to see how many students got an 80% or above (a score of 3 or higher). With this, we are able to see that the most students got an 80% or above in the top ten knowledge areas of 3, 4, 9 and 10 (Figure #2). Next, we wanted to see if this held true for the means of all the students’ scores (Figure #3). In the bar graph, we were again able to see that the highest average scores were in topics 3, 4, 9 and 10.

Next, we created a mosaic plot to visualize the relationship between top ten topic and score (Figure #4). With this, we are able to see that most of the scores give were either a 2 or a 4. It shows that topics 2, 3, and 8 had the most people scoring a 1. We then have a bar chart of the scores by top ten topic, which gives us the same information as the mosaic plot, just in a different format (Figure #5).

Lastly, we have a correlation matrix of all our scores, which allows us to see which topics are high correlated with one another (Figure #6). It makes sense that they are all positively correlated with each other, as if you know one topic, you are more than likely to know a similar topic. Based on the graph, topics 5, 6, and 8 have very high correlation.

Model Selection

Our next step was to create a mixed effects model. We decided to use a mixed effects model because our data was nested, and it had a hierarchical structure.[3] We used our model to predict on the score of each top ten topic by having our fixed effects be semester and the top ten knowledge area, and our random effects be UserID, professor, and SSP. We also included an interaction term between top ten knowledge area and semester year.

```
model_interaction2 <- lmer(Score ~ topFactor +  semesterYear + topFactor:seme
sterYear + (1 | UserID) + (1 | Professor) + (1 | SSP), data = df_long2)
```

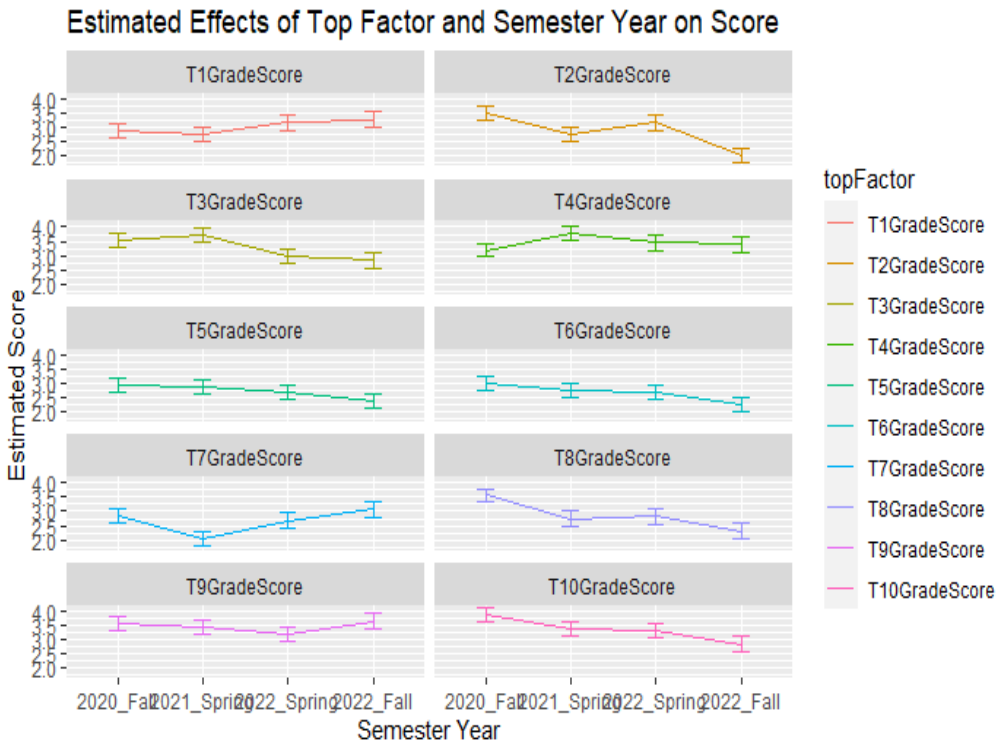
Results

In the output the mixed effects models gave us, we were able to find that there was substantial variability in scores across the different students, with a variance of 2.964e-01. Professors had less variance, with 1.408e-10, and section had little variance in score with 1.659e-02. We also ran a function to get the Bayes estimates of the random effects from our mixed model. It validates our conclusions from our model that Professor and Section does not have a big influence on Score beyond the fixed effects in the model, while the students provide the most variance in the model.

Analysis

With our coefficients and predictors from the mixed effects model, we were able to produce a lattice plot that shows how each semester did in each knowledge area by their overall Bloom's score. In this estimated effects plot, it allowed us to easily visualize what semesters did the best for each topic. For instance, the semester of fall 2020 did the best in topics 2, 5, 6, 8, and 10, but the worst in topic 4. Fall 2022 did the worst in topics 2, 3, 5, 6, 8, and 10, while doing the best in topics 7 and 9. Both spring 2021 and spring 2022 were fairly average across this model, not being the best or worst in more than two topics at time.

[3] https://www.statstest.com/mixed-effects-model/



Conclusion/Recommendations

Ultimately, the greatest effect on score in COMP 170 is semester and students. Fall 2020 is the strongest semester, while Fall 2022 is the weakest semester for the students. What may have impacted this is COVID-19; students may have had an easier time with COMP 170 when it was fully online, whereas when they came back in person they struggled more. According to the mixed-effect model, the most variation in score comes from students, and not from section or professor. This is a good sign; this means that it seems that grading for professors and that different classes within a semester seem to be pretty consistent. There does seem to be high correlation between different topics, but they are all positively correlated. Therefore, we can confirm that the topics do build on each other and do not contradict. We would suggest reexamining the different topics to see if you can make them more independent and get rid of possible redundancy.

As for future recommendations, we would suggest a better way to store data. One way would be keeping it all in one document, but the best way of storing all this data would be through a database. We would also recommend standardizing how professors input scores,

comments, and grades. To make sure that grading is consistent across all professors, we would also recommend having multiple professors grade the same test to see how similar grading is.

For future research, we would also recommend comparing the scores of students who took COMP 170 to their scores in COMP 271, which typically follows COMP 170. That way you can confirm that the skills they are taught in COMP 170 lead into COMP 271.

References

https://citt.ufl.edu/resources/the-learning-process/designing-the-learning-experience/blooms-taxonomy/

https://drive.google.com/drive/folders/1Xuhepv0Fh5kTruPc4lB-ECf55Ssaz43C

https://www.statstest.com/mixed-effects-model/

Appendix A

This appendix contains all tables and figures referenced within this report.

Table 1: Bloom's Taxonomy

Rating	Description
1	No or limited evidence or learning
2	Less than workable knowledge
3	Adequate knowledge of desired learning
4	Full mastery of all basic skills
5	Advanced skills

Table 2: Description of Variables in Dataset

Variable	Description
UserID	Unique ID for each student
Section	Section number student was in
Year	Year student took class
Professor	Semester the class was held
SSP	Professor that taught that class
Top_Ten_Topic	Unique ID combining semester, year, section, and professor
Score	Top ten score category

Figure 1: Knowledge Area #1 Across All Semesters

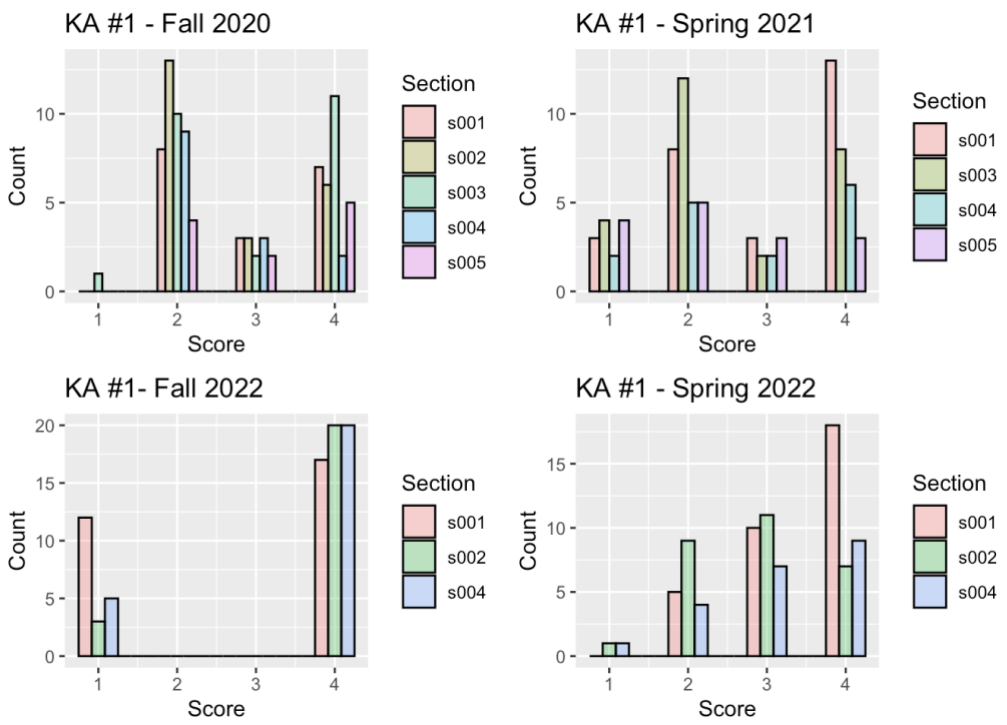


Figure 2: Students Who Received an 80% or Higher in Each Topic Area

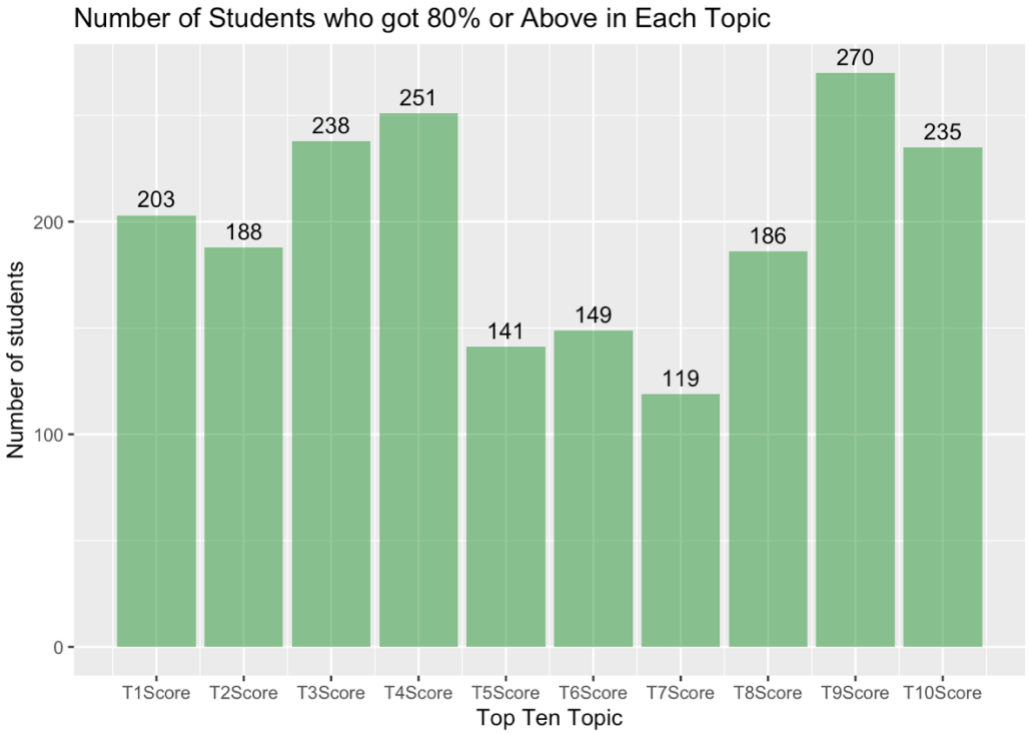


Figure 3: Average Scores for All Knowledge Areas

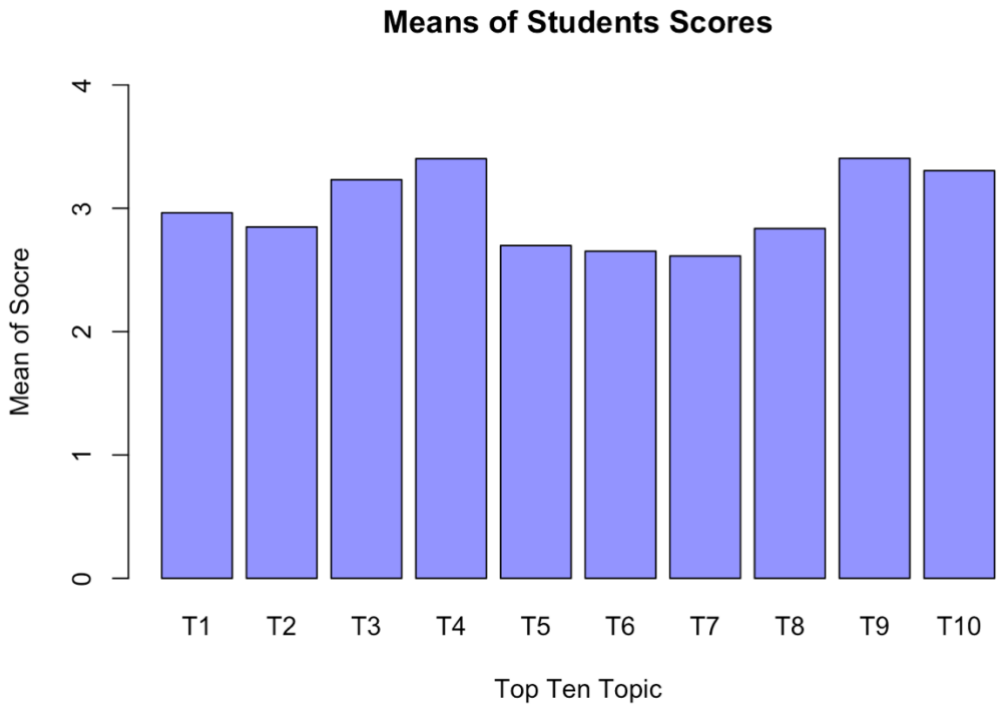


Figure 4: Mosaic Plot - Knowledge Areas vs Score

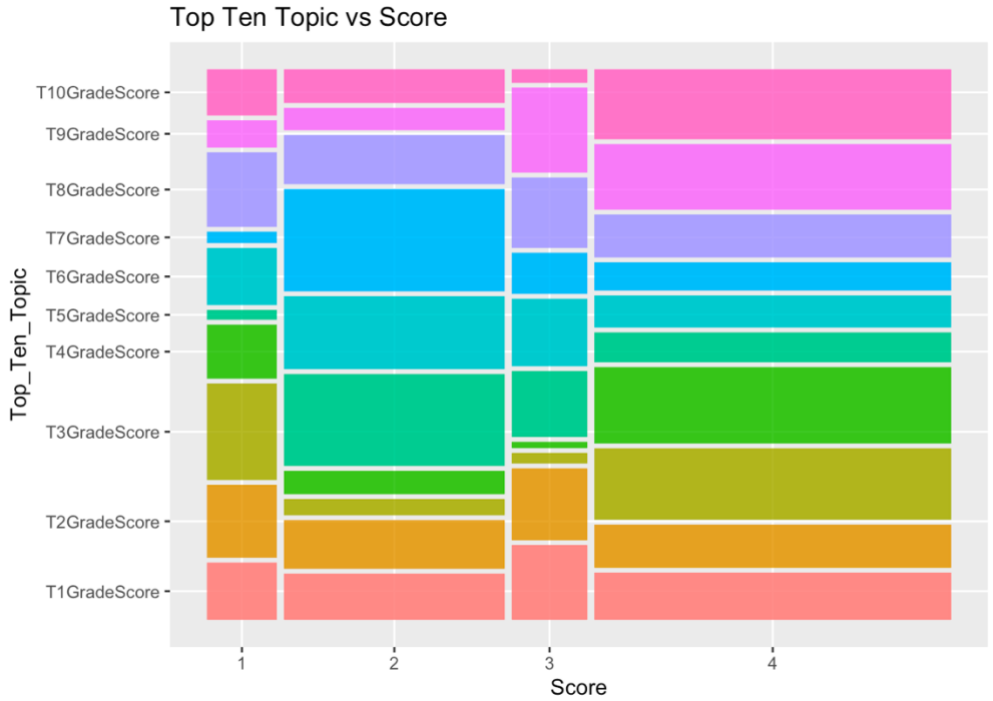


Figure 5: Bar Chart of Scores by Top Ten Topic

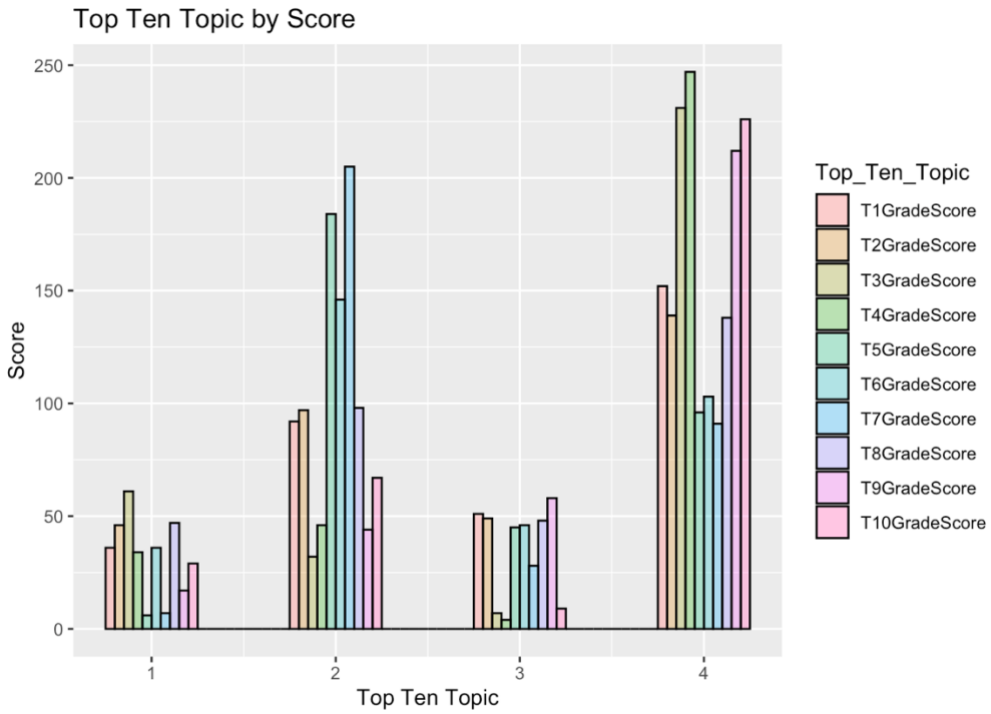
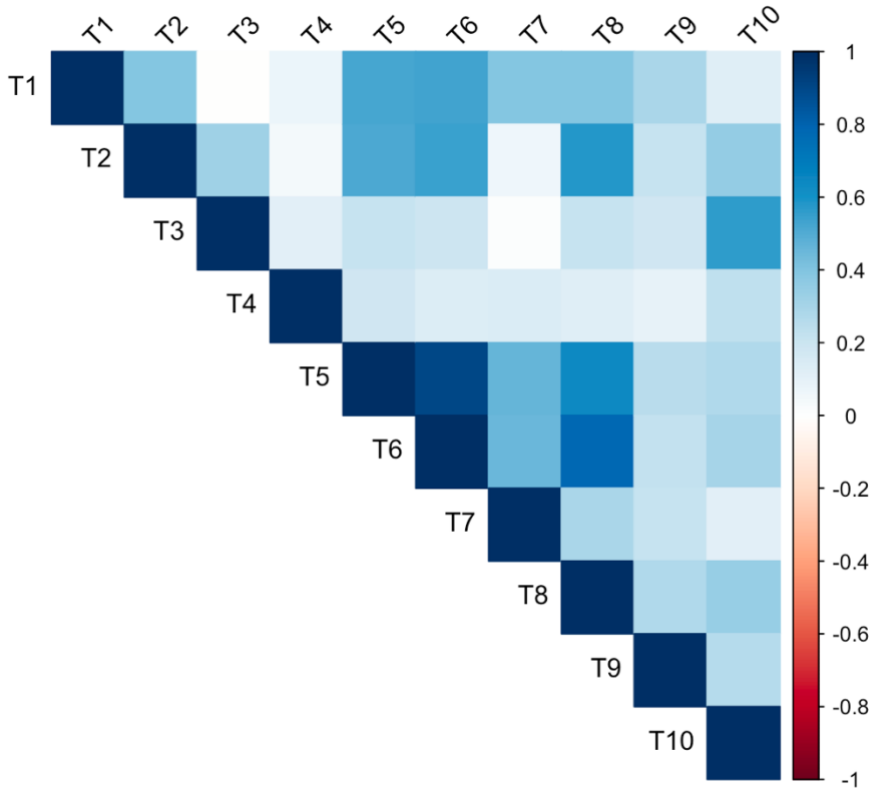


Figure 6: Correlation Matrix of Top Ten Topics



Appendix B

This appendix contains all code used for this report.

Installing Packages

```
#install.packages("readxl")
library("readxl")
#install.packages("tidyverse")
library(tidyverse)
#install.packages("sqldf")
library(sqldf)
#install.packages("ggeffects")
library(ggeffects)
#install.packages("lme4")
library(lme4)
#install.packages("pbkrtest")
library(pbkrtest)
#install.packages("effects")
library(effects)
#install.packages("ggplot2")
library(ggplot2)
#install.packages("corrplot")
library(corrplot)
#install.packages("tidyr")
library(tidyr)
#install.packages("gridExtra")
library(gridExtra)
#install.packages("ggmosaic")
library(ggmosaic)
#install.packages("ggcorrplot")
library(ggcorrplot)
```

Input Data

```
f20 <- read_excel("Clean Data/Cleaned Comp170F2020.xlsx", sheet = "RawExamTHR
EEScores")
f20Key <- read_excel("Clean Data/Cleaned Comp170F2020.xlsx", sheet = "Skills"
)
s21 <- read_excel("Clean Data/Cleaned Comp170S2021.xlsx", sheet = "RawExamTHR
EEScores")
s21Key <- read_excel("Clean Data/Cleaned Comp170S2021.xlsx", sheet = "Skills"
)
f22 <- read_excel("Clean Data/Cleaned Comp170F2022.xlsx", sheet = "RawExamTHR
EEScores")
f22Key <- read_excel("Clean Data/Cleaned Comp170F2022.xlsx", sheet = "Skills"
)
```

```
s22 <- read_excel("Clean Data/Cleaned Comp170Sp2022.xlsx", sheet = "RawExamTH
REEScores")
s22Key <- read_excel("Clean Data/Cleaned Comp170Sp2022.xlsx", sheet = "Skills
")
professors <- read_excel("Clean Data/Professors.xlsx", sheet = "Professors")
```

Data Cleaning

```
#Removing the name column
f20 <- within(f20, rm("Name"))
head(f20)
s21 <- within(s21, rm("Name"))
head(s21)
f22 <- within(f22, rm("Name"))
head(f22)
s22 <- within(s22, rm("Name"))
head(s22)
```

Assigning Student's a Random ID Number

```
set.seed(1234)
userIDs <- sample(1:400, replace=FALSE)
f20[100,]$Section <- "s005"
f20$'User ID' <- userIDs[1:100]
s21$'User ID' <- userIDs[101:195]
s22$'UserID' <- userIDs[196:282]
f22$'User ID' <- userIDs[283:369]
#Fix a column name in spring '22
colnames(s22)[3] <- '1.1'
```

Removing Cheaters/People who Withdrew/etc.

```
f20 <- subset(f20, !is.na(Total))
f20 <- f20[f20$Total != 0, ]
s21 <- subset(s21, !is.na(Total))
s21 <- s21[s21$Total != 0, ]
s22 <- subset(s22, !is.na(Total))
s22 <- s22[s22$Total != 0, ]
f22 <- subset(f22, !is.na(Total))
f22 <- f22[f22$Total != 0, ]
```

Matching Each Question to a Top Ten Topic

```
#Fall 2020 - Top 10 Topic Score Table
f20Scores <- data.frame(UserID = f20$`User ID`)
for (topic in 1:10) {
  f20Topic <- sqldf(paste0("SELECT * FROM f20Key WHERE `Top Ten Topic` = ", t
opic))
  f20TopicSum <- sum(f20Topic$`Points Possible`)
  f20TopicData <- f20[append("User ID", f20Topic$Question)]

  if (length(f20TopicData) == 2) {
    f20TopicScore <- (f20TopicData[2]) / f20TopicSum
  } else if (length(f20TopicData) == 3) {
```

```
    f20TopicScore <- (f20TopicData[2] + f20TopicData[3]) / f20TopicSum
  } else {
    f20TopicScore <- (f20TopicData[2] + f20TopicData[3] + f20TopicData[4]) /
f20TopicSum
  }
  f20Scores[paste0("T", topic, "Grade")] <- f20TopicScore}

#Spring 2021 - Top 10 Topic Score Table
s21Scores <- data.frame(UserID = s21$`User ID`)
for (topic in 1:10) {
  s21Topic <- sqldf(paste0("SELECT * FROM s21Key WHERE `Top Ten Topic` = ", t
opic))
  s21TopicSum <- sum(s21Topic$`Points Possible`)
  s21TopicData <- s21[append("User ID", s21Topic$Question)]

  if (length(s21TopicData) == 2) {
    s21TopicScore <- (s21TopicData[2]) / s21TopicSum
  } else if (length(s21TopicData) == 3) {
    s21TopicScore <- (s21TopicData[2] + s21TopicData[3]) / s21TopicSum
  } else {
    s21TopicScore <- (s21TopicData[2] + s21TopicData[3] + s21TopicData[4]) /
s21TopicSum
  }
  s21Scores[paste0("T", topic, "Grade")] <- s21TopicScore}

#Fall 2022 - Top 10 Topic Score Table
f22Scores <- data.frame(UserID = f22$`User ID`)
for (topic in 1:10) {
  f22Topic <- sqldf(paste0("SELECT * FROM f22Key WHERE `Top Ten Topic` = ", t
opic))
  f22TopicSum <- sum(f22Topic$`Points Possible`)
  f22TopicData <- f22[append("User ID", f22Topic$Question)]

  if (length(f22TopicData) == 2) {
    f22TopicScore <- (f22TopicData[2]) / f22TopicSum
  } else if (length(f22TopicData) == 3) {
    f22TopicScore <- (f22TopicData[2] + f22TopicData[3]) / f22TopicSum
  } else {
    f22TopicScore <- (f22TopicData[2] + f22TopicData[3] + f22TopicData[4]) /
f22TopicSum
  }
  f22Scores[paste0("T", topic, "Grade")] <- f22TopicScore
}

#Spring 2022 - Top 10 Topic Score Table
s22Scores <- data.frame(UserID = s22$`UserID`)
for (topic in 1:10) {
  s22Topic <- sqldf(paste0("SELECT * FROM s22Key WHERE `Top Ten Topic` = ", t
opic))
  s22TopicSum <- sum(s22Topic$`Points Possible`)
```

```
  s22TopicData <- s22[append("UserID", s22Topic$Question)]

  if (length(s22TopicData) == 2) {
    s22TopicScore <- (s22TopicData[2]) / s22TopicSum
  } else if (length(s22TopicData) == 3) {
    s22TopicScore <- (s22TopicData[2] + s22TopicData[3]) / s22TopicSum
  } else {
    s22TopicScore <- (s22TopicData[2] + s22TopicData[3] + s22TopicData[4]) /
s22TopicSum
  }
  s22Scores[paste0("T", topic, "Grade")] <- s22TopicScore
}
```

Assigning a Bloom's Score to each top ten topic

```
#Assigning a Bloom's score to each top ten topic
# 90% - 100%  4
# 90% - 80%   3
# 80% - 10%   2
# 10% - 0%    1

# Define a function to assign scores based on grade ranges
assign_score <- function(grade) {
  if (!is.na(grade) && grade >= 0.9) {
    return(4)
  } else if (!is.na(grade) && grade >= 0.8) {
    return(3)
  } else if (!is.na(grade) && grade >= 0.1) {
    return(2)
  } else {
    return(1)}}

# FALL 2020
f20TopTenScores <- data.frame(UserID = f20Scores$UserID, Section = f20$Sectio
n, Year = 2020, Semester = "Fall")
f20TopTenScores$T1Score <- sapply(f20Scores$T1Grade, assign_score)

for (i in 2:10) {
  topic_name <- paste0("T", i, "Grade") # construct the topic name
  topic_scores <- f20Scores[, topic_name] # extract the scores for the topic
  topic_score <- sapply(topic_scores, assign_score) # assign scores using the
assign_score function
  col_name <- paste0(topic_name, "Score") # construct the column name for the
scores
  f20TopTenScores[, col_name] <- topic_score # assign the scores to the data
frame
}
#f20TopTenScores

#SPRING 2021
s21TopTenScores <- data.frame(UserID = s21Scores$UserID, Section = s21$Sectio
```

```
n, Year = 2021, Semester = "Spring")
s21TopTenScores$T1Score <- sapply(s21Scores$T1Grade, assign_score)

for (i in 2:10) {
  topic_name <- paste0("T", i, "Grade")
  topic_scores <- s21Scores[, topic_name]
  topic_score <- sapply(topic_scores, assign_score)
  col_name <- paste0(topic_name, "Score")
  s21TopTenScores[, col_name] <- topic_score
}
#s21TopTenScores

#FALL 2022
f22TopTenScores <- data.frame(UserID = f22Scores$UserID, Section = f22$Sectio
n, Year = 2022, Semester = "Fall")
f22TopTenScores$T1Score <- sapply(f22Scores$T1Grade, assign_score)

for (i in 2:10) {
  topic_name <- paste0("T", i, "Grade")
  topic_scores <- f22Scores[, topic_name]
  topic_score <- sapply(topic_scores, assign_score)
  col_name <- paste0(topic_name, "Score")
  f22TopTenScores[, col_name] <- topic_score
}
#f22TopTenScores

#SPRING 2022
s22TopTenScores <- data.frame(UserID = s22Scores$UserID, Section = s22$Sectio
n, Year = 2022, Semester = "Spring")
s22TopTenScores$T1Score <- sapply(s22Scores$T1Grade, assign_score)

for (i in 2:10) {
  topic_name <- paste0("T", i, "Grade")
  topic_scores <- s22Scores[, topic_name]
  topic_score <- sapply(topic_scores, assign_score)
  col_name <- paste0(topic_name, "Score")
  s22TopTenScores[, col_name] <- topic_score
}
#s22TopTenScores

#Combining all 4 semesters
BloomAll <- rbind(f20TopTenScores, s21TopTenScores, f22TopTenScores, s22TopTe
nScores)

#Adding Professors to each section
BloomA <- BloomAll %>%
  mutate(Professor = case_when(
    Year == 2020 & Section == "s001 (DIGH)" & Semester == "Fall" ~ "Chan-Tin,
Eric",
    Year == 2020 & Section == "s002" & Semester == "Fall" ~ "Yacobellis, Robe
```

```
rt",
    Year == 2020 & Section == "s003" & Semester == "Fall" ~ "Irakliotis, Leo"
,
    Year == 2020 & Section == "s004" & Semester == "Fall" ~ "Honig, William",
    Year == 2020 & Section == "s005" & Semester == "Fall" ~ "Yacobellis, Robe
rt",

    Year == 2021 & Section == "s001" & Semester == "Spring"  ~ "Yacobellis, R
obert",
    Year == 2021 & Section == "s003" & Semester == "Spring"  ~ "Yacobellis, R
obert",
    Year == 2021 & Section == "s004" & Semester == "Spring"  ~ "Honig, Willia
m",
    Year == 2021 & Section == "s005" & Semester == "Spring"  ~ "Irakliotis, L
eo",
    Year == 2021 & Section == "s006" & Semester == "Spring"  ~ "Irakliotis, L
eo",

    Year == 2022 & Section == "s001" & Semester == "Fall" ~ "Yacobellis, Robe
rt",
    Year == 2022 & Section == "s002" & Semester == "Fall" ~ "Honig, William",
    Year == 2022 & Section == "s004" & Semester == "Fall" ~ "Yacobellis, Robe
rt",

    Year == 2022 & Section == "S001" & Semester == "Spring"  ~ "Yacobellis, R
obert",
    Year == 2022 & Section == "s002" & Semester == "Spring"  ~ "Yacobellis, R
obert",
    Year == 2022 & Section == "s003" & Semester == "Spring"  ~ "Irakliotis, L
eo",
    Year == 2022 & Section == "s004" & Semester == "Spring"  ~ "Honig, Willia
m",
    Year == 2022 & Section == "s005" & Semester == "Spring"  ~ "Irakliotis, L
eo",
    TRUE ~ "Unknown Professor")) %>%
  mutate(SSP = case_when(
    Year == 2020 & Professor == "Chan-Tin, Eric" & Semester == "Fall" ~ "F20_
s001_ECT",
    Year == 2020 & Professor == "Yacobellis, Robert" & Semester == "Fall" ~ "
F20_s002_RY",
    Year == 2020 & Professor == "Irakliotis, Leo" & Semester == "Fall" ~ "F20
_s003_LI",
    Year == 2020 & Professor == "Honig, William" & Semester == "Fall" ~ "F20_
s004_WH",
    Year == 2020 & Professor == "Yacobellis, Robert" & Semester == "Fall" ~ "
F20_s005_RY",

    Year == 2021 & Professor == "Yacobellis, Robert" & Semester == "Spring" ~
"S21_s001_RY",
    Year == 2021 & Professor == "Honig, William" & Semester == "Spring" ~ "S2
```

```
1_s004_WH",
    Year == 2021 & Professor == "Irakliotis, Leo" & Semester == "Spring" ~ "S
21_s005_LI",
    Year == 2021 & Professor == "Irakliotis, Leo" & Semester == "Spring" ~ "S
21_s006_LI",

    Year == 2022 & Professor == "Yacobellis, Robert" & Semester == "Fall" ~ "
F22_s001_RY",
    Year == 2022 & Professor == "Honig, William" & Semester == "Fall" ~ "F22_
s002_WH",
    Year == 2022 & Professor == "Yacobellis, Robert" & Semester == "Fall" ~ "
F22_s004_RY",

    Year == 2022 & Professor == "Irakliotis, Leo" & Semester == "Spring" ~ "S
22_s001_RY",
    Year == 2022 & Professor == "Yacobellis, Robert" & Semester == "Spring" ~
"S22_s002_RY",
    Year == 2022 & Professor == "Honig, William" & Semester == "Spring" ~ "S2
2_s003_LI",
    Year == 2022 & Professor == "Irakliotis, Leo" & Semester == "Spring" ~ "S
22_s004_WH",
    Year == 2022 & Professor == "Irakliotis, Leo" & Semester == "Spring" ~ "S
22_s005_LI",
  ))
colnames(BloomA)[5] <- 'T1GradeScore' #Fixing name for Top Ten Topic #1

#Fixing section Numbers
BloomA$Section <- replace(BloomA$Section, BloomA$Section == "s001 (DIGH)", "s
001")
BloomA$Section <- replace(BloomA$Section, BloomA$Section == "S001", "s001")
head(BloomA)

##   UserID Section Year Semester T1GradeScore T2GradeScore T3GradeScore
## 1    284    s004 2020     Fall            2            4            4
## 2    336    s004 2020     Fall            2            4            4
## 3    101    s004 2020     Fall            3            4            4
## 4    111    s004 2020     Fall            3            4            4
## 5    393    s004 2020     Fall            2            4            4
## 6    133    s004 2020     Fall            2            1            1
##   T4GradeScore T5GradeScore T6GradeScore T7GradeScore T8GradeScore T9Grade
Score
## 1            4            2            2            2            4
4
## 2            1            2            2            2            3
3
## 3            4            4            4            3            4
4
## 4            4            4            4            3            4
4
## 5            4            2            2            2            3
```

```
3
## 6            1            2            2            2            4
4
##   T10GradeScore      Professor         SSP
## 1             4 Honig, William F20_s004_WH
## 2             4 Honig, William F20_s004_WH
## 3             4 Honig, William F20_s004_WH
## 4             4 Honig, William F20_s004_WH
## 5             4 Honig, William F20_s004_WH
## 6             2 Honig, William F20_s004_WH
```

Creating a long data frame

```
df_long2 <- pivot_longer(BloomA, cols = c(T1GradeScore, T2GradeScore, T3Grade
Score, T4GradeScore, T5GradeScore, T6GradeScore, T7GradeScore, T8GradeScore,
T9GradeScore ,T10GradeScore), names_to = "Top_Ten_Topic",  values_to = "Score
")
df_long2

df_long2$Top_Ten_Topic <- factor(df_long2$Top_Ten_Topic, levels = c("T1GradeS
core", "T2GradeScore", "T3GradeScore", "T4GradeScore", "T5GradeScore", "T6Gra
deScore", "T7GradeScore", "T8GradeScore", "T9GradeScore", "T10GradeScore"))
```

PCA

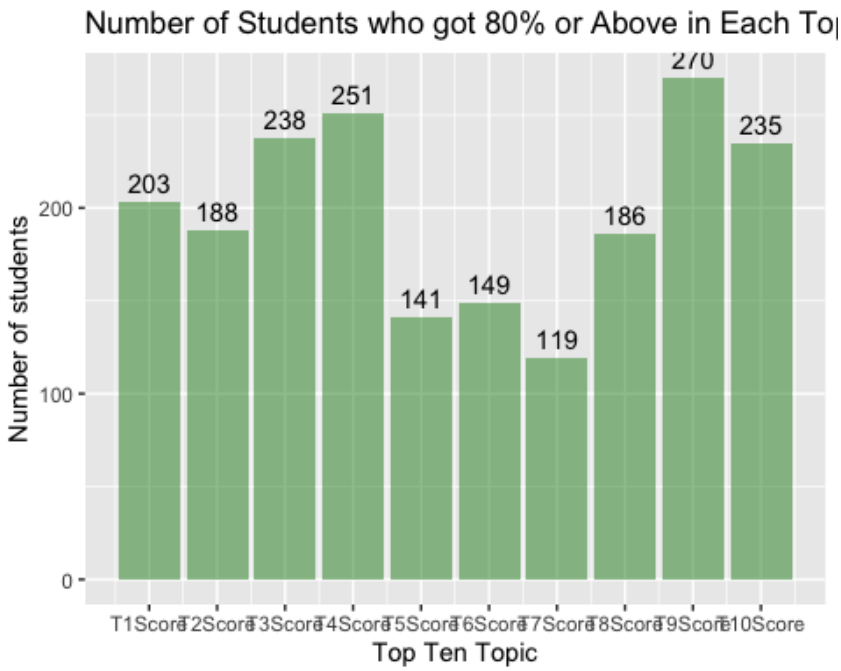
```
my_data_numeric <- df_long2 %>%
  select_if(is.numeric)
my_data_standardized <- scale(my_data_numeric)
pca <- prcomp(my_data_standardized)
corr_matrix <- cor(my_data_standardized)
#ggcorrplot(corr_matrix)
data.pca <- princomp(corr_matrix)
summary(data.pca)

## Importance of components:
##                           Comp.1    Comp.2       Comp.3
## Standard deviation     0.6637994 0.5742994 1.290478e-08
## Proportion of Variance 0.5719124 0.4280876 2.161511e-16
## Cumulative Proportion  0.5719124 1.0000000 1.000000e+00

data.pca$loadings[, 1:2]

##            Comp.1     Comp.2
## UserID  0.1508482  0.8317651
## Year    0.6413567 -0.4935938
## Score  -0.7522675 -0.2540312
```

Visualizations

How Many Students got 80% (3) or Above in each Topic

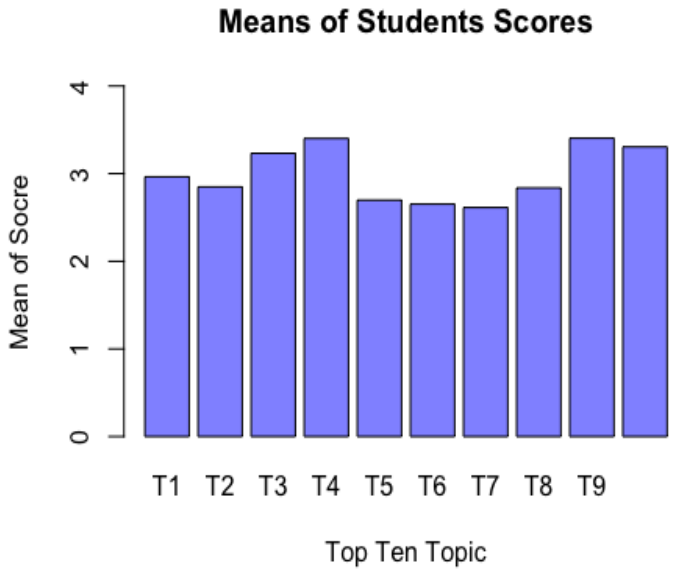
```
for (i in 1:10) {
  varname <- paste0("T", i, "GradeScore")
  passcount <- sum(BloomA[, varname] %in% c(3, 4))
```

```
  cat("Number of students who got an 80% or above in topic #", i, ":", passco
unt, "\n")}

## Number of students who got an 80% or above in topic # 1 : 203
## Number of students who got an 80% or above in topic # 2 : 188
## Number of students who got an 80% or above in topic # 3 : 238
## Number of students who got an 80% or above in topic # 4 : 251
## Number of students who got an 80% or above in topic # 5 : 141
## Number of students who got an 80% or above in topic # 6 : 149
## Number of students who got an 80% or above in topic # 7 : 119
## Number of students who got an 80% or above in topic # 8 : 186
## Number of students who got an 80% or above in topic # 9 : 270
## Number of students who got an 80% or above in topic # 10 : 235

#Creating a data frame with the pass counts for each topic
pass_counts <- numeric(10)
for (i in 1:10) {
  varname <- paste0("T", i, "GradeScore")
  passcount <- sum(BloomA[, varname] %in% c(3, 4))
  pass_counts[i] <- passcount
}
pass <- data.frame(Topic = 1:10, PassCount = pass_counts)
#Bar plot of passing students
ggplot(pass, aes(x = Topic, y = PassCount)) +
  geom_bar(stat = "identity", fill = alpha("forestgreen", 0.5)) +
  geom_text(aes(label = PassCount), vjust = -0.5) +
  scale_x_continuous(breaks = 1:10, labels = paste0("T", 1:10, "Score")) +
  labs(title = "Number of Students who got 80% or Above in Each Topic",
       x = "Top Ten Topic", y = "Number of students")
```



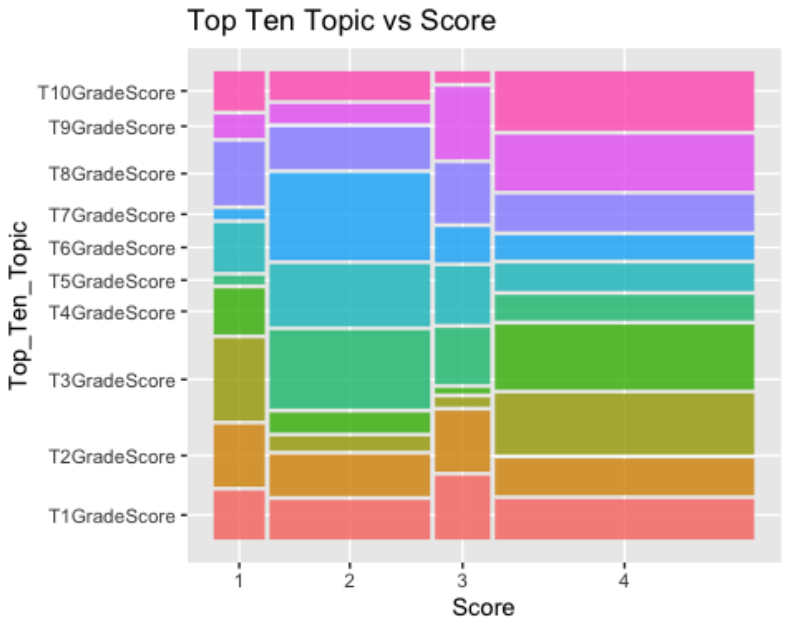
Mean Bar Chart for All Semesters

```
BloomAmean <- data.frame(T1 = BloomA$T1GradeScore,
                         T2 = BloomA$T2GradeScore,
                         T3 = BloomA$T3GradeScore,
                         T4 = BloomA$T4GradeScore,
                         T5 = BloomA$T5GradeScore,
                         T6 = BloomA$T6GradeScore,
                         T7 = BloomA$T7GradeScore,
                         T8 = BloomA$T8GradeScore,
                         T9 = BloomA$T9GradeScore,
                         T10 = BloomA$T10GradeScore)
means <- colMeans(BloomAmean)
barplot(means, main="Means of Students Scores", xlab="Top Ten Topic", ylab="M
ean of Socre", col = rgb(0, 0, 1, alpha = 0.5), ylim=c(0, length(means)-6))
```



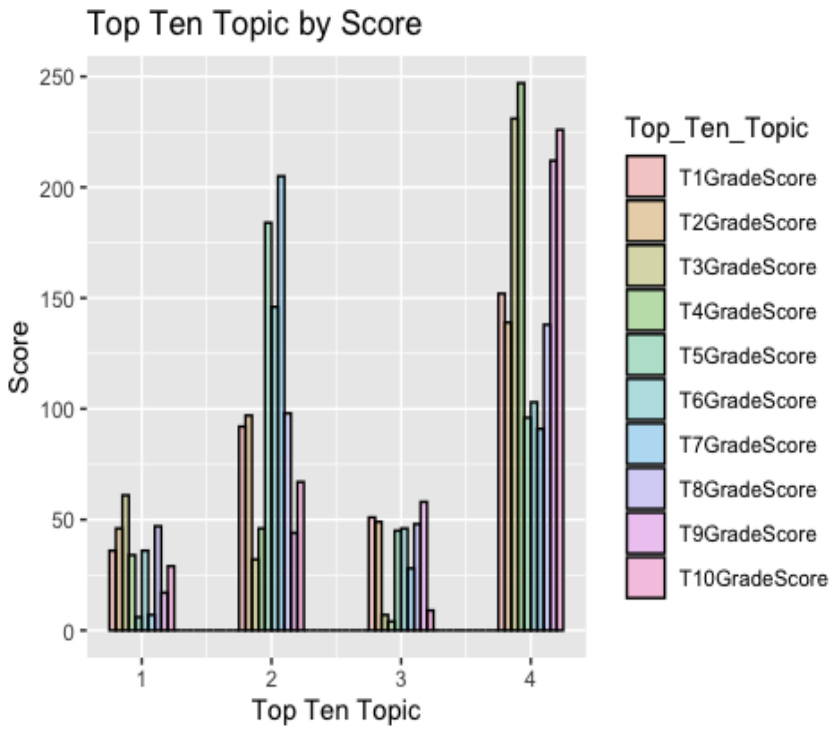
Mosaic Plot- Top Ten Topic vs Score

```
#Mosaic Plot
ggplot(data = df_long2) +
  geom_mosaic(aes(x = product(Top_Ten_Topic, Score), fill = Top_Ten_Topic)) +
  labs(x="Score", y="Top_Ten_Topic", title = "Top Ten Topic vs Score") +
  guides(fill = FALSE)
```



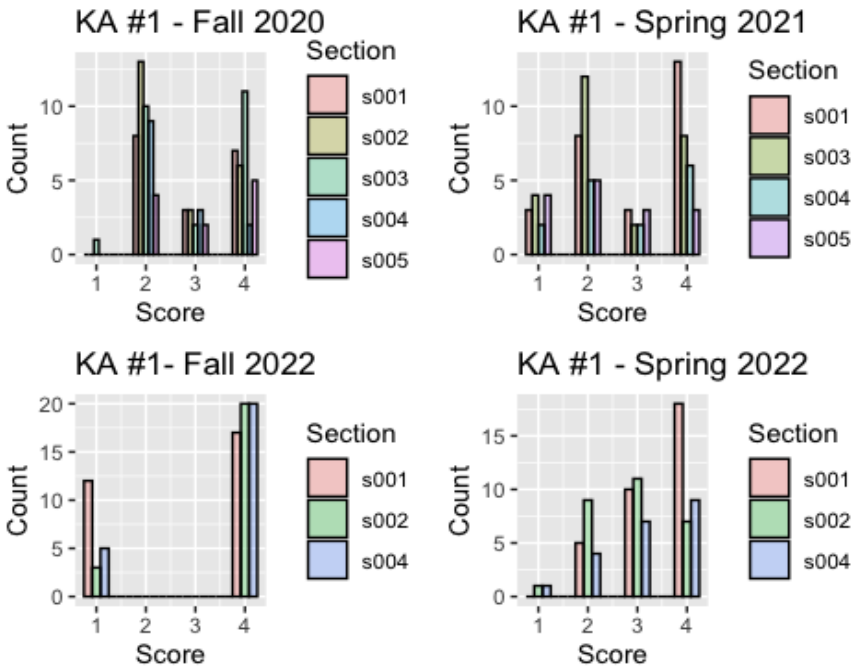
Bar chart of Scores by Top Ten Topic

```
#Bar chart of Scores by Top Ten Topic
ggplot(df_long2, aes(x = Score, fill = Top_Ten_Topic, color=Top_Ten_Topic)) +
  geom_histogram(binwidth = 0.5, position = "dodge", alpha=0.3, color = "blac
k") +
  labs(title = "Top Ten Topic by Score", x = "Top Ten Topic", y = "Score", fi
ll = "Top_Ten_Topic")
```



Knowledge Area #1 Side by Side bar charts

```
query1 <- sqldf("SELECT * FROM BloomA WHERE Year = 2020")
query3 <- sqldf("SELECT * FROM BloomA WHERE Year = 2022 AND Semester = 'Fall'
")
query2 <- sqldf("SELECT * FROM BloomA WHERE Year = 2021")
query4 <- sqldf("SELECT * FROM BloomA WHERE Year = 2022 AND Semester = 'Sprin
g' ")

graph1 <- ggplot(query1, aes(x = T1GradeScore, fill = Section, color=Section)
) +
  geom_histogram(binwidth = 0.5, position = "dodge", alpha=0.3, color = "blac
k") +
  labs(title = "KA #1 - Fall 2020", x = "Score", y = "Count", fill = "Section
")

graph2 <- ggplot(query2, aes(x = T1GradeScore, fill = Section, color=Section)
) +
  geom_histogram(binwidth = 0.5, position = "dodge", alpha=0.3, color = "blac
k") +
  labs(title = "KA #1 - Spring 2021", x = "Score", y = "Count", fill = "Secti
on")

graph3 <- ggplot(query3, aes(x = T1GradeScore, fill = Section, color=Section)
) +
  geom_histogram(binwidth = 0.5, position = "dodge", alpha=0.3, color = "blac
k") +
  labs(title = "KA #1- Fall 2022", x = "Score", y = "Count", fill = "Section"
)
graph4 <- ggplot(query4, aes(x = T1GradeScore, fill = Section, color=Section)
) +
  geom_histogram(binwidth = 0.5, position = "dodge", alpha=0.3, color = "blac
k") +
  labs(title = "KA #1 - Spring 2022", x = "Score", y = "Count", fill = "Secti
on")
library(ggplot2)
library(gridExtra)
grid.arrange(graph1, graph2, graph3, graph4, ncol = 2)
```

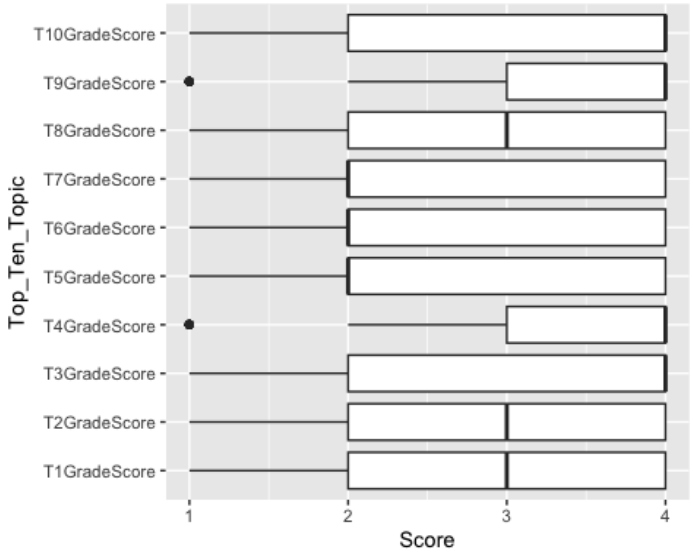


Violin plot of all Student's Scores

```
violin <- ggplot(df_long2, aes(x=Top_Ten_Topic, y=Score)) +
  geom_violin() + ggtitle("Violin Plot - All Student's Scores")
violin + coord_flip()
```

Box plot of all Student's Scores

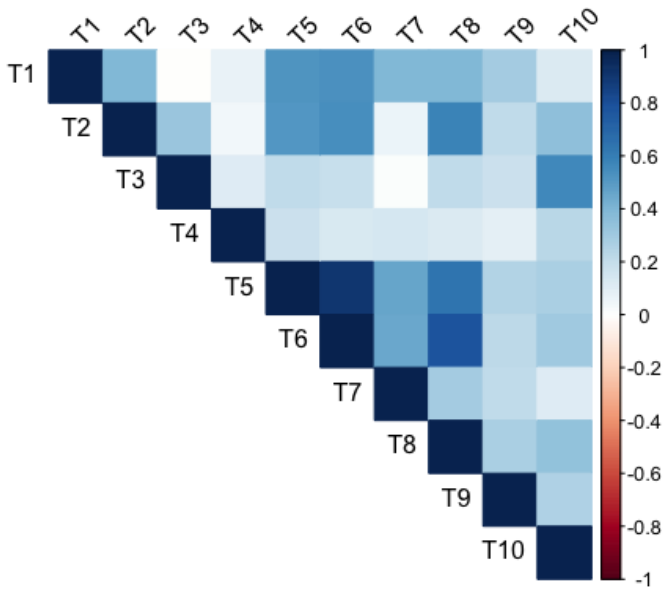
```
box_plot <- ggplot(df_long2, aes(x=Top_Ten_Topic, y=Score)) +
  geom_boxplot()
box_plot + coord_flip()
```



Analysis

Correlation Matrix

```
scores <- data.frame(T1 = BloomA$T1GradeScore,
                     T2 = BloomA$T2GradeScore,
                     T3 = BloomA$T3GradeScore,
                     T4 = BloomA$T4GradeScore,
                     T5 = BloomA$T5GradeScore,
                     T6 = BloomA$T6GradeScore,
                     T7 = BloomA$T7GradeScore,
                     T8 = BloomA$T8GradeScore,
                     T9 = BloomA$T9GradeScore,
                     T10 = BloomA$T10GradeScore)
corr_matrix <- cor(scores)
corrplot(corr_matrix, method = "color", type = "upper", tl.col = "black", tl.
srt = 45)
```




```
col <- colorRampPalette(c("#67001F", "#B2182B", "#D6604D", "#F4A582", "#FDDBC
7", "#FFFFFF", "#D1E5F0", "#92C5DE", "#4393C3", "#2166AC", "#053061"))(12)
#legend("bottomleft", c(-1, seq(0.2, 1, by = 0.2)), fill = col, title = "Corr
elation", cex = 0.8, box.col = NA)
```

Mixed Effects Model

```
#Factor the top ten scores
topFactor <- as.factor(df_long2$Top_Ten_Topic)
semesterYear <- paste(df_long2$Year, df_long2$Semester)
df_long2$semesterYear <- paste(df_long2$Year, df_long2$Semester, sep = "_")

#Mixed effect model with interaction - new section identifier SSP
model_interaction2 <- lmer(Score ~ topFactor +  semesterYear + topFactor:seme
sterYear + (1 | UserID) + (1 | Professor) + (1 | SSP), data = df_long2)

summary(model_interaction2)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: Score ~ topFactor + semesterYear + topFactor:semesterYear + (1 |
##     UserID) + (1 | Professor) + (1 | SSP)
##    Data: df_long2
##
## REML criterion at convergence: 8759.2
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.8073 -0.6066  0.0435  0.7213  2.9249
##
## Random effects:
##  Groups    Name        Variance  Std.Dev.
##  UserID    (Intercept) 2.964e-01 5.444e-01
##  SSP       (Intercept) 1.659e-02 1.288e-01
```

```
##  Professor (Intercept) 1.408e-10 1.187e-05
##  Residual              6.822e-01 8.260e-01
## Number of obs: 3310, groups:  UserID, 331; SSP, 11; Professor, 4
##
## Fixed effects:
##                                                 Estimate Std. Error t valu
e
## (Intercept)                                     2.838888   0.123920  22.90
9
## topFactorT2GradeScore                           0.662921   0.123817   5.35
4
## topFactorT3GradeScore                           0.662921   0.123817   5.35
4
## topFactorT4GradeScore                           0.325843   0.123817   2.63
2
## topFactorT5GradeScore                           0.089888   0.123817   0.72
6
## topFactorT6GradeScore                           0.157303   0.123817   1.27
0
## topFactorT7GradeScore                          -0.011236   0.123817  -0.09
1
## topFactorT8GradeScore                           0.696629   0.123817   5.62
6
## topFactorT9GradeScore                           0.696629   0.123817   5.62
6
## topFactorT10GradeScore                          0.977528   0.123817   7.89
5
## semesterYear2021_Spring                        -0.171240   0.183524  -0.93
3
## semesterYear2022_Fall                           0.442863   0.192106   2.30
5
## semesterYear2022_Spring                         0.311891   0.190788   1.63
5
## topFactorT2GradeScore:semesterYear2021_Spring  -0.662921   0.178240  -3.71
9
## topFactorT3GradeScore:semesterYear2021_Spring   0.325030   0.178240   1.82
4
## topFactorT4GradeScore:semesterYear2021_Spring   0.710302   0.178240   3.98
5
## topFactorT5GradeScore:semesterYear2021_Spring   0.042642   0.178240   0.23
9
## topFactorT6GradeScore:semesterYear2021_Spring  -0.157303   0.178240  -0.88
3
## topFactorT7GradeScore:semesterYear2021_Spring  -0.687559   0.178240  -3.85
7
## topFactorT8GradeScore:semesterYear2021_Spring  -0.696629   0.178240  -3.90
8
## topFactorT9GradeScore:semesterYear2021_Spring  -0.009882   0.178240  -0.05
5
## topFactorT10GradeScore:semesterYear2021_Spring -0.351022   0.178240  -1.96
```

```
9
## topFactorT2GradeScore:semesterYear2022_Fall    -1.922662   0.181798 -10.57
6
## topFactorT3GradeScore:semesterYear2022_Fall    -1.117467   0.181798  -6.14
7
## topFactorT4GradeScore:semesterYear2022_Fall    -0.221947   0.181798  -1.22
1
## topFactorT5GradeScore:semesterYear2022_Fall    -0.973005   0.181798  -5.35
2
## topFactorT6GradeScore:semesterYear2022_Fall    -1.170290   0.181798  -6.43
7
## topFactorT7GradeScore:semesterYear2022_Fall    -0.196556   0.181798  -1.08
1
## topFactorT8GradeScore:semesterYear2022_Fall    -1.657668   0.181798  -9.11
8
## topFactorT9GradeScore:semesterYear2022_Fall    -0.345980   0.181798  -1.90
3
## topFactorT10GradeScore:semesterYear2022_Fall   -1.432074   0.181798  -7.87
7
## topFactorT2GradeScore:semesterYear2022_Spring  -0.662921   0.178802  -3.70
8
## topFactorT3GradeScore:semesterYear2022_Spring  -0.870238   0.178802  -4.86
7
## topFactorT4GradeScore:semesterYear2022_Spring  -0.057550   0.178802  -0.32
2
## topFactorT5GradeScore:semesterYear2022_Spring  -0.565497   0.178802  -3.16
3
## topFactorT6GradeScore:semesterYear2022_Spring  -0.632913   0.178802  -3.54
0
## topFactorT7GradeScore:semesterYear2022_Spring  -0.488764   0.178802  -2.73
4
## topFactorT8GradeScore:semesterYear2022_Spring  -1.062483   0.178802  -5.94
2
## topFactorT9GradeScore:semesterYear2022_Spring  -0.696629   0.178802  -3.89
6
## topFactorT10GradeScore:semesterYear2022_Spring -0.867772   0.178802  -4.85
3

##
## Correlation matrix not shown by default, as p = 40 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it

## optimizer (nloptwrap) convergence code: 0 (OK)
## boundary (singular) fit: see ?isSingular
```

Extract the conditional estimates of the random effects

```
ranef(model_interaction2)
```

```
## $UserID
##      (Intercept)
## 2   -0.143922071
## 3    0.036357403
## 4    0.225065607
## 7   -0.395453074
## 10   0.648503606
## 11   0.119997438
## 12  -0.476742873
## 13   0.119997438
## 14   0.816766458
## 15  -0.387791468
## 16   0.526446433
## 17  -0.001814786
## 18  -0.126222195
## 19   0.411314484
## 20   0.181237125
## 21  -0.370091592
## 22  -0.164394384
## 23  -0.449031155
## 25   0.181237125
## 26   0.661314312
## 27   0.038707639
## 28  -0.314163275
## 30  -0.314163275
## 31  -0.151583677
## 33   0.668975919
## 34  -0.062632272
## 35   0.521025311
## 36   0.573894082
## 38   0.343816723
## 40   0.550224803
## 41   0.387645205
## 42  -0.698321674
## 43  -0.476742873
## 45   0.038707639
## 47   0.572897061
## 48  -0.611610753
## 49  -0.164394384
## 50  -0.288801793
## 51  -0.611610753
## 52   0.587686120
## 53   0.587686120
## 54  -0.774190351
## 55  -0.469081267
## 56   0.524096197
## 57  -0.001814786
## 58   0.114576316
## 59  -0.478587765
## 60  -0.645452903
```

```
## 61  -0.239003908
## 62  -0.425252784
## 63  -0.129293081
## 64   0.689026030
## 66   0.387645205
## 67   0.686675795
## 68   0.363866835
## 70  -0.234718368
## 71   0.086155288
## 72  -0.639322471
## 73   0.498734714
## 74   0.329027664
## 75  -0.143922071
## 76  -0.291872679
## 77  -0.245684183
## 78   0.363866835
## 79   0.455594238
## 80   0.661314312
## 81  -0.613960989
## 82  -0.370091592
## 83  -0.645452903
## 84   0.254865318
## 85  -0.326973982
## 86   0.580024513
## 87  -0.875530262
## 88   0.661314312
## 89   0.831555517
## 90  -0.276013953
## 91  -0.483870325
## 93  -0.100093588
## 94  -0.855480150
## 95   0.018657527
## 96  -0.326973982
## 97  -0.565160124
## 98  -0.113434355
## 99   0.166448066
## 100  0.587686120
## 101  0.455594238
## 102  0.573894082
## 103  0.618173835
## 104 -0.225211870
## 105  0.506396321
## 106  0.506396321
## 107 -0.232873476
## 108 -0.181383387
## 109  0.181237125
## 110 -0.286451557
## 111  0.455594238
## 113  0.849255393
## 114  0.343816723
```

```
## 115  0.550224803
## 116 -0.076424310
## 117 -0.164394384
## 118 -0.803746961
## 119 -0.367741356
## 120  0.770315829
## 121  0.181237125
## 122 -0.018803789
## 123 -0.164394384
## 124 -0.077421330
## 125 -0.530320954
## 126  0.573894082
## 128  0.816766458
## 129  0.849255393
## 130  0.648503606
## 132 -0.126222195
## 133 -0.926332345
## 134 -0.314163275
## 135  0.605385996
## 136 -0.245684183
## 137  0.573894082
## 138 -0.469081267
## 139 -0.698321674
## 140  0.445156634
## 141  0.334310225
## 142 -0.314163275
## 143 -0.151583677
## 144 -0.123871959
## 145  0.160764812
## 146  0.661314312
## 147  0.661314312
## 148  0.417444915
## 149  0.306355406
## 150  0.524096197
## 151 -0.151583677
## 152  0.363866835
## 153  0.573894082
## 154 -0.306501669
## 155  0.411314484
## 157 -0.153428569
## 158  0.573894082
## 159  0.099947326
## 160  0.648503606
## 161  0.740759220
## 164 -0.205161758
## 165  0.587686120
## 166  0.254865318
## 169 -0.889322299
## 170  0.683604909
## 171 -0.476742873
```

```
## 172 -0.072138770
## 173 -0.062632272
## 176  0.329027664
## 177  0.282577036
## 178  0.181237125
## 180  0.750265718
## 181 -1.465031167
## 182 -0.652133177
## 183 -0.205161758
## 184 -0.262673186
## 185 -0.232873476
## 186  0.099947326
## 188  0.498734714
## 189  0.580024513
## 190 -0.469081267
## 191  0.099947326
## 193  0.247737865
## 194  0.280226800
## 195  0.225065607
## 196  0.661314312
## 197 -0.205161758
## 198 -0.042582160
## 201 -0.469081267
## 202  0.445156634
## 203 -0.153428569
## 204  0.181237125
## 205 -0.286451557
## 206  0.831555517
## 207 -0.042582160
## 209 -0.449031155
## 210 -0.225211870
## 214 -0.601173149
## 215  0.573894082
## 216 -0.291872679
## 217  0.363866835
## 218  0.573894082
## 219 -1.296768315
## 221  0.160764812
## 222 -0.306501669
## 223  0.849255393
## 224 -0.164394384
## 225 -0.129293081
## 226 -0.692900552
## 227  0.181237125
## 228  0.225065607
## 229  0.334310225
## 231  0.038707639
## 232 -0.123871959
## 235  0.683604909
## 238 -0.855480150
```

```
## 239  0.038707639
## 240 -0.558032672
## 241 -0.321290727
## 242 -0.565160124
## 243 -0.692900552
## 244 -0.727739722
## 245 -1.295771294
## 246  0.445156634
## 248  0.573894082
## 249 -0.476742873
## 252  0.524096197
## 253 -0.956820061
## 254 -0.469081267
## 255 -0.469081267
## 256 -0.129293081
## 258 -0.489553580
## 259 -0.469081267
## 260  0.816766458
## 261 -0.530320954
## 262  0.160764812
## 263  0.282577036
## 265  0.173575519
## 266  0.683604909
## 267 -0.570843378
## 269 -0.712950664
## 271 -0.143922071
## 273 -0.077421330
## 275 -0.143922071
## 276  0.086155288
## 278  0.683604909
## 279  0.851605628
## 280 -0.126222195
## 281 -0.558032672
## 282 -0.164394384
## 283 -0.286451557
## 284 -0.032144556
## 285  0.661314312
## 286  0.262526924
## 287  0.661314312
## 288 -0.044932396
## 289 -0.631660865
## 290 -0.072138770
## 293  0.254865318
## 294  0.526446433
## 295  0.119997438
## 296 -0.402580526
## 297 -0.408263781
## 298 -0.343962985
## 299  0.648503606
## 300 -0.712950664
```

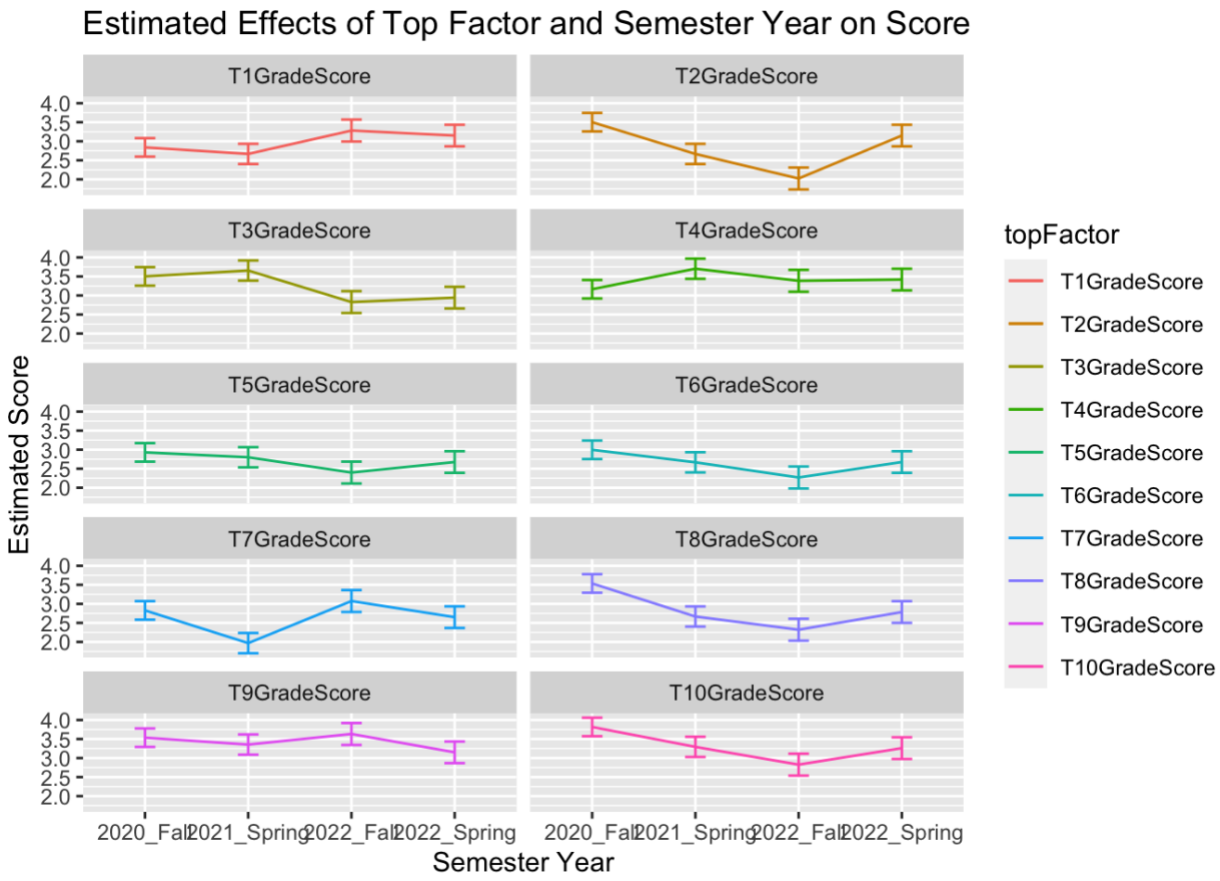
```
## 301 -0.314163275
## 302  0.496889823
## 303 -0.939120185
## 304 -0.587832382
## 305  0.404634209
## 306  0.506396321
## 307 -0.164394384
## 308  0.363866835
## 309 -0.306501669
## 311  0.648503606
## 312  0.282577036
## 314 -0.314163275
## 315 -0.454452277
## 316 -0.558032672
## 317  0.160764812
## 319 -0.314163275
## 320  0.740759220
## 321  0.343816723
## 322  0.018657527
## 323  0.343816723
## 324 -0.072138770
## 325  0.498734714
## 326 -0.601173149
## 327 -0.151583677
## 328 -0.018803789
## 329 -0.151583677
## 331 -0.890319320
## 332 -0.042582160
## 333  0.661314312
## 334 -0.722457162
## 335  0.036357403
## 336 -0.438593551
## 337 -0.151583677
## 338 -0.774190351
## 339 -0.564163104
## 340  0.343816723
## 342 -0.143922071
## 343  0.306355406
## 344  0.038707639
## 345  0.198937001
## 346 -0.936769949
## 347 -0.794240463
## 348 -0.320293707
## 349  0.036357403
## 350 -0.291872679
## 351  0.661314312
## 352 -0.314163275
## 356 -1.208351064
## 359 -0.326973982
## 360  0.816766458
```

```
## 361 -0.225211870
## 362  0.119997438
## 364 -0.314163275
## 365  0.181237125
## 366  0.816766458
## 368 -0.631660865
## 370 -0.476742873
## 371  0.485924008
## 372  0.130435042
## 373 -0.726742702
## 374 -0.070293878
## 375  0.181237125
## 376 -0.387791468
## 377 -0.449031155
## 378  0.417444915
## 379 -0.158711129
## 380 -0.070293878
## 381 -0.885036760
## 382 -0.181383387
## 384 -0.489553580
## 385  0.661314312
## 386 -0.205161758
## 388  0.618173835
## 389  0.248734886
## 390  0.201287237
## 391  0.323344410
## 392  0.689026030
## 393 -0.194724154
## 394  0.485924008
## 395  0.750265718
## 397 -0.288801793
## 398 -0.314163275
## 399 -0.482873305
##
## $SSP
##               (Intercept)
## F20_s001_ECT  0.058401274
## F20_s002_RY  -0.062498030
## F20_s003_LI   0.029284117
## F20_s004_WH  -0.025187361
## F22_s001_RY  -0.151445589
## F22_s002_WH   0.151445589
## S21_s001_RY   0.041717209
## S21_s004_WH   0.014296058
## S21_s005_LI  -0.056013268
## S22_s002_RY  -0.009096502
## S22_s003_LI   0.009096502
##
## $Professor
##                      (Intercept)
```

```
## Chan-Tin, Eric      4.958141e-10
## Honig, William      1.270503e-09
## Irakliotis, Leo    -2.269246e-10
## Yacobellis, Robert -1.539392e-09
##
## with conditional variances for "UserID" "SSP" "Professor"
```

Visualize Mixed Effects Model- Lattice Plot

```
#plot(model_interaction)
#plot(allEffects(model_interaction))
#plot(predictorEffect(predictor = "semesterYear", mod = model_interaction))
#Extract the predictor effects
pe <- predictorEffect(predictor = "semesterYear", mod = model_interaction2)

#Convert the data to a long format
pe_long <- reshape2::melt(pe, id.vars = "semseterYear")
colnames(pe_long) <- c("semesterYear", "topFactor", "estimatedScore", "se", "
lower", "upper")
pe_long$topFactor <- factor(pe_long$topFactor, levels = paste0("T", 1:10, "Gr
adeScore"))

ggplot(pe_long, aes(x = semesterYear, y = estimatedScore, group = 1, color =
topFactor)) +
  geom_line() +
  geom_errorbar(aes(ymin = lower, ymax = upper), width = 0.2) +
  xlab("Semester Year") +
  ylab("Estimated Score") +
  ggtitle("Estimated Effects of Top Factor and Semester Year on Score") +
  facet_wrap(~topFactor, ncol = 2)
```


Estimated Effects of Top Factor and Semester Year on Score
T1GradeScore
T2GradeScore
T3GradeScore
T4GradeScore
T5GradeScore
T6GradeScore
T7GradeScore
T8GradeScore
T9GradeScore
T10GradeScore
Estimated Score
4.0
3.5
3.0
2.5
2.0
2020_Fall
2021_Spring
2022_Fall
2022_Spring
Semester Year
topFactor
T1GradeScore
T2GradeScore
T3GradeScore
T4GradeScore
T5GradeScore
T6GradeScore
T7GradeScore
T8GradeScore
T9GradeScore
T10GradeScore