Dashboards for Clicker Data

INFO 4100 Learning Analytics

[Gabriela Castro Martinez 5259723, Vanessa Rosero 5246624, Hermon Beluts 5291676, Hermione Bossolina 5245124, Hannah Yeh hmy6, Aishwarya Khubchandani 5248270]

This project is about developing a learning analytics dashboard based on clicker data. You will work as a team to learn how to make a dashboard using R Shiny (official page with several tutorials: <https://shiny.rstudio.com/tutorial/>).

**Learning Objectives**

1. Understand the structure of clicker data
2. Create multiple different visualizations
3. Design and implement an instructor and student dashboard
4. Critically evaluate your own dashboard design

You are given aggregated clicker records for a CS course taught at Cornell. There are two datasets: the experience dataset and the quiz dataset.

**Scenario**

You are approached by a college instructor who uses iClickers in her CS class on Business Intelligence. She would like to gain insights about her students and how they are engaging/performing in order to better help them in class. She would also like to better support students by giving them feedback at scale about where they stand and perhaps how they compare to others in the class.

You offer to build a prototype of a dashboard using her clicker data: this is a dashboard for the instructor which offers an overview of the class characteristics, engagement, and performance; and it is a dashboard for students which offers a specific student an overview of their engagement and performance (and how it compares to others).

**Data**

The **experience dataset** contains one record per student who completed the CS course between 2016-2018. There are two sources to this dataset: Faculty Center and a Skills Survey (administered via the Blackboard LMS) where students self reported their skill level for various skills the first week of class. This data has been de-identified. Name, netid, emplid, major have all been removed and replaced with a unique numeric identifier. Note that not all students completed the skills survey, they will have null values for the survey result fields.

| Attribute Name | Data Type | Definition |
| --- | --- | --- |
| student\_key | numeric Unique key | Assigned as part of de-identification process. Uniquely identifies student records for this data set only. |
| year | numeric | Four digit year student was enrolled in BI Class. |
| prog | character Values (GRAD, UGRAD) | Indicates whether the student was a graduate or undergraduate student when they were enrolled in BI course. |
| database\_score | numeric (0-5) | Self reported experience level with database technology prior to taking course. 0= no experience, 5= expertise |
| sql\_score | numeric (0-5) | Self reported experience level with SQL prior to taking course. 0= no experience, 5=expertise |
| programing\_score | numeric (0-5) | Self reported experience level with Any Programing language prior to taking course. 0=no experience, 5=expertise |
| stored\_proc\_score | numeric (0-5) | Self reported experience level with stored procedure languages prior to taking course. 0=no experience, 5=expertise |
| etl\_score | numeric (0-5) | Self reported experience level with Extract Transform Load (ETL) development prior to taking course. 0=no experience, 5=expertise |
| data\_vis\_score | numeric (0-5) | Self reported experience level using data visualization tools prior to taking course. 0=no experience, 5=expertise |
| requirement\_gather\_score | numeric (0-5) | Self reported experience level gathering customer requirements prior to taking course. 0=no experience, 5=expertise |
| skill\_survey\_score | numeric | Sum of the self reported skill level scores. |

The **quiz dataset** contains one record per student per class session held where iClickers were used. Sources used in the creation of this data set include: iClicker session xml files, Blackboard gradebook (for quiz scores), and the Blackboard class schedule (used to map iClicker session to related quiz scores). Note that in some cases there are multiple iClicker sessions / lectures associated with a single quiz. This dataset may be joined to the experience dataset by the student\_key field.

| Attribute Name | Data Type | Definition |
| --- | --- | --- |
| Acad\_date\_key | numeric | Date key in the form of YYYYMMDD indicating the date the class session was held. |
| student\_key | numeric | Unique identifier for students who took BI class 2016-2018. This key is the primary key for the experience\_data file. |
| year | numeric | Four digit year class session was held. |
| session\_number | numeric | Identifies the session number for a particular semester. Session number is assigned by iClicker. |
| quiz\_number | numeric | There are 10 quizzes throughout the BI course. This attribute indicates which quiz is associated with the iClicker session(s). |
| attended | numeric (0,1) | Binary indicating whether the student attended that particular class session / lecture. 0=no, 1=yes. |
| total\_possible\_clicker | numeric | The total number of iClicker questions asked that session. |
| total\_completed\_clicker | numeric | The number of iClicker questions answered by student that session. |
| completed\_q\_clicker | numeric | The number of completed Quiz iClicker questions |
| correct\_q\_clicker | numeric | How many correct Quiz answers by student that session. |
| completed\_t\_clicker | number | How many Temperature questions answered by student that session. Temperature questions are 0-5, 0= bad, 5=great. There is no correct answer to Temperature questions, they are used to guage how students are feeling about a particular subject, assignment, etc. |
| avg\_t\_clicker | number | The average temperature answer by student for that session. An average of 1 or 2 would be generally negative, while 4 or 5 would be generally positive responses. |
| quiz\_score | numeric | Quiz score out of 20 points possible. |

# Part 1: Planning / Sketching

Go through the planning / sketching process described in the reading about dashboards. While some dashboards are certainly better than others, there is not one correct solution here. However, spending enough time to make a concrete plan is essential for the success of your project. Everything you do to make the dashboards will be easier if you have a clear plan, especially because you will be splitting up the work and everyone needs to know what they should work on.

**Question 1:** You will make a student dashboard and a teacher dashboard. Carefully consider the implications of this for design and content. To plan, answer the following prompts once for the student dashboard and then for the teacher dashboard. The more concrete you are here the easier it will be later. Focus on the concrete ideas that you will implement in the next steps. You can iterate on this step and modify your responses as your ideas for the dashboard become clearer. You should explore the dataset in R for 5-10 minutes to get a good sense of what the dataset has to offer.

*Planning for the student dashboard*

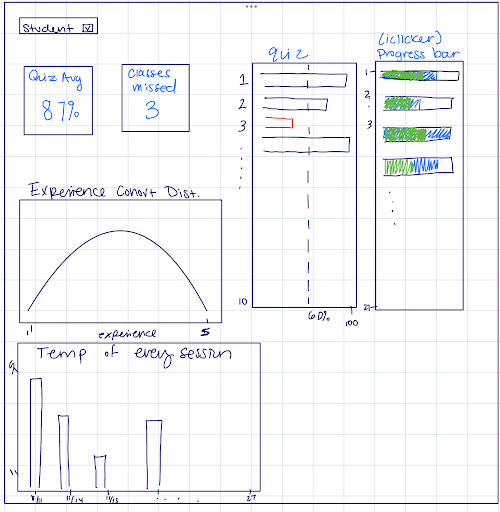
* For whom? Who will use it and what is their background?
  + Individual students in the course who want to monitor their performance.
  + There is an expected level of digital literacy - we expect people to know how to navigate the webpage and understand the graphs we are utilizing.
* Why? What is the goal? What questions to answer?
  + Tracking student attendance and participation to assist students in monitoring their attendance.
  + How am I doing compared to students who have similar experiences to me?
  + What is the student’s participation, knowledge, and excitement in the course.
* What? What data to show and what is its structure?
  + The total number of classes missed which we can show by doing sum(quiz$ATTENDED == 0) for each student.
  + We can use quiz$CORRECT\_Q\_CLICKER, quiz$TOTAL\_COMPLETED\_CLICKER, quiz$TOTAL\_POSSIBLE\_CLICKER to compare how many clicker questions they correctly answered, how many clicker questions they completed, and how many total possible clickers there were.
  + Monitoring the 10 quizzes that occur throughout the year. Having a bar graph of quiz scores having the ones under 60% be red. Having an average of the quiz score.
  + We can compare the Experience Cohort by matching students with different cohorts to show how they are doing compared to others.
  + The temperature of every session that the student is in.
* How? How will visualizations support the goal?
  + Showing the total number of iclickers answered on a bar: visualizing how much you answered, how many you got correct, and total number of iclickers. It will allow students to see how their performance is going supporting our goal of tracking student attendance and participation in the course.
  + Compare the Experience Cohort by matching
    - Look at the experience data (skill\_survey\_score and prog) and then make 3 ranges (arbitrary categories). Students will fall into one of these categories and will be either an UGRAD or a GRAD and then they will see how they’re doing among student with similar prior knowledge / experience.
    - This will support the goal of “How am I doing compared to students who have similar experiences to me?”
  + Having a bar graph of quiz scores having the ones under 60% be red. This will support the goal of updating the students on their quiz scores and the average of their quizzes.
  + Having a progress bar of how they are doing with correct quiz questions and total completed will help them see how well they are actually doing in certain topics. In case they need a certain number of correct answers.

*Planning for the teacher dashboard*

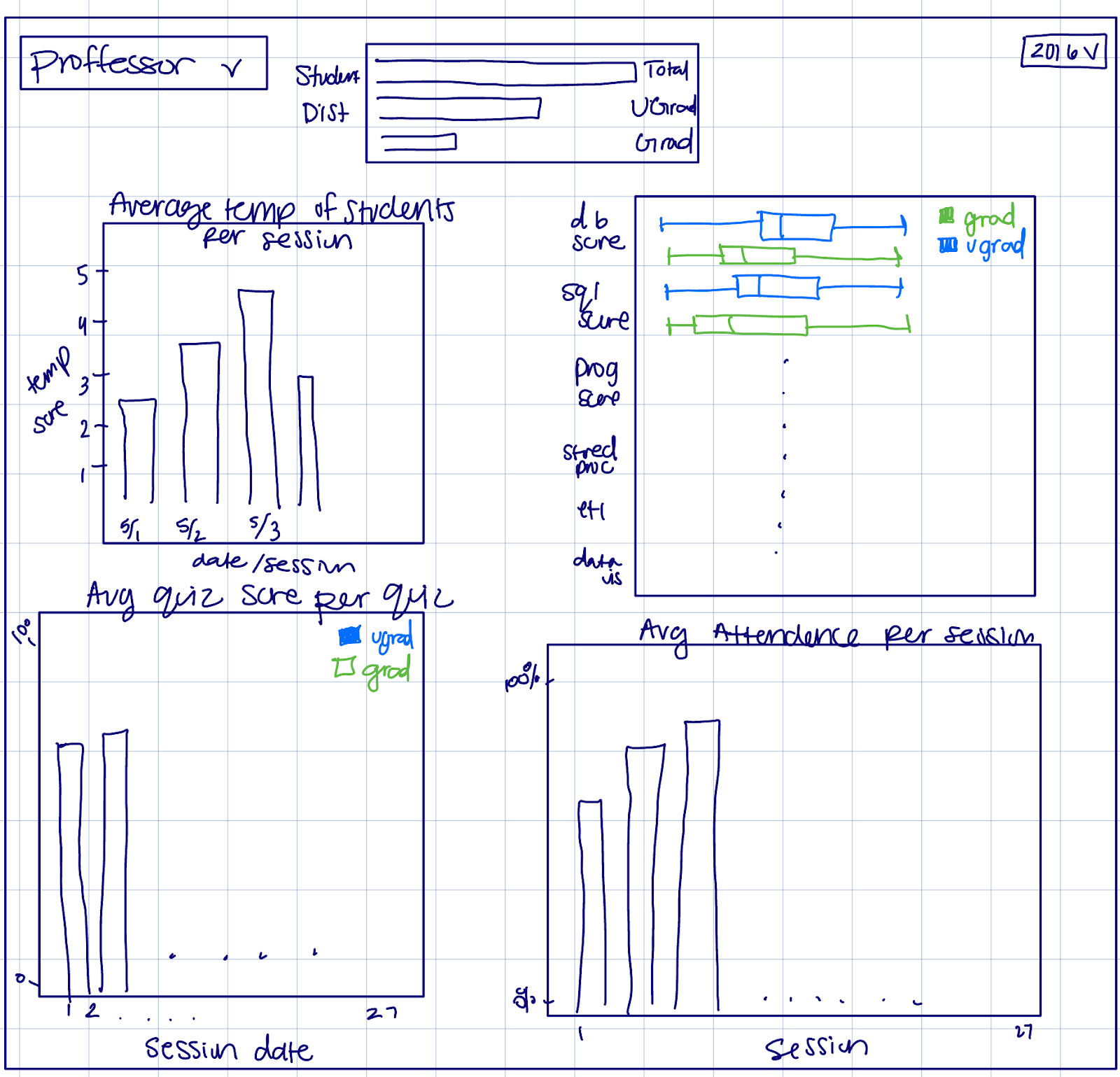
* For whom? Who will use it and what is their background?
  + An instructor of a class interested in understanding their students’ progress in realtime.
  + An instructor who is interested in understanding student’s participation and their current temperature in the course.
  + An instructor who has enough digital literacy to read basic bar graphs and navigate the dashboard.
* Why? What is the goal? What questions to answer?
  + Determining student expertise and prior knowledge in different course related categories (sql\_score, programming\_score, data\_vis\_score, etc.)
  + What is the student’s prior knowledge in the class and how is that assisting them in the course?
* What? What data to show and what is its structure?
  + The average temperature of all students
  + Distribution of average student skill level in each of the different categories (sql\_score, programming\_score, data\_vis\_score, store\_proc\_score, etl\_score).
  + How many students are GRAD and UGRAD.
  + Average attendance per session of the students split on UGRAD and GRAD.
  + Average quiz scores for the 10 quizzes on all students separated on UGRAD and GRAD.
* How? How will visualizations support the goal?
  + Bar graph of average temperature each day to see how students are feeling throughout the semester.
  + Each category will have a box plot to describe the distribution of skill levels among all students so that the professor can gain an understanding of which category students score the highest in.

**Question 2:** Based on your plan above, make a sketch of what the dashboard would look like. See this week’s readings for examples. Be detailed about what kinds of data points and visualizations you want to see in different parts of the page. Consider the user experience and how you should position more general information compared to more specific information, and where you may need some additional explanation to help the viewer understand a graphic, for example. In your sketch, it is useful to give labels to different objects, because in the steps below you can split up work between team members and the labels will help you connect the UI with the data objects. Show your sketches in section to get feedback from the teaching team.

Each dashboard should contain at least 4 data visualizations. You may include any additional summary statistics (e.g. key percentages or tables).



student sketch



Teacher sketch

# Part 2: Dashboard Wire-frame Implementation

This is where you generate the dashboard layout. You are given a very basic wire frame example for the dashboard below. For more information on how R Shiny Dashboards work, look at <https://rstudio.github.io/shinydashboard/get_started.html> and <https://rstudio.github.io/shinydashboard/structure.html>. You can add different types of content into a fuidRow(). In the starter code, there are 2 rows of content: the first has two little info boxes; the second has two larger viz boxes. You can add more rows and change what is in them as you wish. Follow the naming convention, e.g. inst.info1 is the first info box for instructors.

Your team can split up the tasks. Some work on creating the UI (this part), while others work on pre-processing the data and creating the statistics and visualizations that will populate the UI (next part).

**Question 3:** Create the layout for the dashboard tabs. You can have as many “tabs” as you like. Each tab is the content displayed when the user clicks on one of the menu items (so it is the page content). Here you are just specifying the wire frame i.e. **what goes where on the pages**, not what goes into it.

#######################################  
####### BEGIN INPUT: Question 3 #######  
#######################################  
# Example of a tab (i.e. page)  
instructor\_dash = tabItem(  
 tabName = "instructor",  
 h2("Instructor Dashboard"),  
   
 # Dynamic infoBoxes  
 fluidRow(  
 infoBoxOutput("inst.info1"),  
 infoBoxOutput("inst.info2")  
 ),  
 # Visualizations  
 fluidRow(  
 box(  
 title = "Student Distribution",  
 plotOutput("student\_distribution.plot", height = 400)  
 ),  
 box(  
 title = "Experience Distribution",  
 plotOutput("experience\_dist.plot", height = 400)  
 ),  
 box(  
 title = "Average Quiz Score Per Quiz",  
 plotOutput("avg\_quiz.plot", height = 300)  
 ),  
 box(  
 title = "Average Temperature Per Session",  
 plotOutput("average\_temp.plot", height = 300)  
 ),  
 box(  
 title = "Average Attendance Per Session",  
 plotOutput("average\_attendance.plot", height = 300)  
 )  
 )  
 )  
   
  
student\_dash = tabItem(  
 tabName = "student",  
 h2("Student Dashboard"),  
 # Dynamic infoBoxes  
 fluidRow(  
 infoBoxOutput("small.box1"),  
 infoBoxOutput("small.box2")  
 ),  
 # Vvisualizations  
 fluidRow(  
 box(  
 title = "Progress Bar",  
 plotOutput("clicker\_prog\_student.plot", height = 400)  
 ),  
 box(  
 title = "Temperature of Every Session",  
 plotOutput("temp\_student.plot", height = 400)  
 ),  
 box(  
 title = "Quiz Score",  
 plotOutput("quiz\_score.plot", height = 400)  
 ),   
 box(  
 title = "Histogram of Quiz Scores for Your Experience Cohort",  
 plotOutput("cohortHistogram", height = 400)  
 )  
   
 )  
)  
  
#######################################  
#######################################

# Part 3: Data Pre-processing

Get the data ready for use in the dashboard. Before the next stage, you want to have the data ready in the right format for simple computations and plotting. To do this effectively, you need to know by now what you want to display in each dashboard. However, this is also an iterative process. Once you have completed a first iteration of the design, you can come back to this step and add further pre-processing for more visualizations you like to add. This step is also an opportunity to better understand the structure of the datasets.

The instructor dashboard should show information for all students. The student dashboard is typically focused on an individual student. You can either pick a student (at random or intentionally) and use them as the “reference student” for the student dashboard. Or, a bit more ambitious but also more rewarding to try out, you can create an interactive dashboard in which you select the student and then the dashboard updates to show the information for that student. I would recommend you start with the simpler version and get that to work before you try to make it dynamic.

Use the space below to be ready for your information visualizations in the dashboards.

#######################################  
####### BEGIN INPUT #######  
#######################################  
  
selected\_student <- subset(quiz, STUDENT\_KEY == '105')  
  
selected\_year <- subset(experience, YEAR == '2017')  
  
  
#######################################  
#######################################

# Part 4: Prepare All Data Visualizations

This is where you create the content for the wire frames you created above. Again, you can refer to the examples and documentation in <https://rstudio.github.io/shinydashboard/get_started.html> and <https://rstudio.github.io/shinydashboard/structure.html> for guidance. You can also find many examples online just by searching with Google.

**Question 4:** For each of the pieces of content you planned for in the wire frames above, generate the relevant content. You need to assign them all to the output variable by referencing the name of the wire frame element you chose above like this output$name.of.element.

server = function(input, output) {  
 #######################################  
 ####### BEGIN INPUT: Question 4 #######  
 #######################################  
   
 #info boxes  
 output$inst.info1 = renderInfoBox({  
 infoBox(  
 "Students total",  
 length(unique(quiz$STUDENT\_KEY)),  
 icon = icon("list"),  
 color = "purple"  
 )  
 })  
   
 output$inst.info2 = renderInfoBox({  
 infoBox(  
 "Attendance",  
 paste0(round(100 \* mean(quiz$ATTENDED)), "%"),  
 icon = icon("list"),  
 color = "fuchsia"  
 )  
 })  
   
 output$small.box1 = renderInfoBox({  
 infoBox(  
 "Quiz Average",  
 paste0(round(100 \* mean(selected\_student$QUIZ\_SCORE/20)), "%"),  
 icon = icon("list"),  
 color = "purple"  
 )  
 })  
   
 output$small.box2 = renderInfoBox({  
 infoBox(  
 "Classes Missed",  
 paste0(sum(selected\_student$ATTENDED == 0)),  
 icon = icon("list"),  
 color = "fuchsia"  
 )  
 })  
   
 output$inst.plot1 = renderPlot({  
 hist(quiz$QUIZ\_SCORE)  
 })  
   
 #output$inst.plot2 = renderPlot({  
 # I'm giving you three examples here  
 # only the last one will be displayed  
 # but you can try out each one.  
 # As you can see, using ggplot allows you to  
 # make much better-looking visualizations.  
   
 # simple  
 #hist(quiz$TOTAL\_COMPLETED\_CLICKER)  
   
 # using ggplot  
 #ggplot(quiz, aes(TOTAL\_COMPLETED\_CLICKER)) +  
 # geom\_histogram()  
   
 # using ggplot to make it clearer  
 #ggplot(quiz, aes(x = TOTAL\_COMPLETED\_CLICKER)) +  
 # geom\_density(fill = "gray") + # makes the density line and gray fill  
 # labs(x = "Completed Clicker Questions", y = "Density") + # change axis labels  
 # theme\_classic() # to use white background  
 #})  
   
   
 #INSTRUCTOR PLOTS  
   
 output$student\_distribution.plot = renderPlot({  
 student\_dist\_data <- experience %>%  
 group\_by(PROG) %>%  
 summarise(count = n(), .groups = 'drop') %>%  
 mutate(PROGRAM = factor(PROG, levels = c("UGRAD", "GRAD")))  
  
 # plot distribution of students across programs  
 ggplot(student\_dist\_data, aes(x = PROGRAM, y = count, fill = PROG)) +  
 geom\_bar(stat = "identity", alpha = .3) +  
 labs(title = "Distribution of Students by Program",  
 x = "Program",  
 y = "Number of Students") +  
 scale\_fill\_manual(values = c("blue", "purple")) +  
 theme\_minimal() +  
 coord\_flip()  
 })  
   
   
 output$experience\_dist.plot = renderPlot({  
 cleaned\_data <- tidyr::pivot\_longer(  
 selected\_year,  
 cols = c(  
 DATABASE\_SCORE,  
 SQL\_SCORE,  
 PROGRAMING\_SCORE,  
 STORED\_PROC\_SCORE,  
 ETL\_SCORE,  
 DATA\_VIS\_SCORE  
 ),  
 names\_to = "score\_type",  
 values\_to = "score"  
 )  
 ggplot(cleaned\_data, aes(y = PROG, x = score, fill = PROG)) +  
 geom\_boxplot() +  
 facet\_wrap( ~ score\_type, ncol = 1) +  
 labs(  
 title = "Boxplot of Scores by Category and Program",  
 x = NULL,  
 y = NULL,  
 fill = "Program"  
 ) +  
 theme\_minimal() +  
 theme(axis.text.y = element\_blank(), legend.position = "bottom") +  
 scale\_fill\_manual(values = alpha(c("blue", "purple"), 0.3))  
   
 })  
   
 # joining the two datasets so I can separate quiz dataset by prog  
 joined\_data = reactive({  
 merge(quiz, experience, by = "STUDENT\_KEY", all.x = TRUE)  
 })  
   
 # toggle between years  
 selected\_year\_data = reactive({  
 subset(joined\_data(), year = 2017)  
 })  
   
   
 # grouping avg quiz scores by prog and quiz number  
 avg\_scores = reactive({  
 aggregate(QUIZ\_SCORE ~ PROG + QUIZ\_NUMBER,  
 data = selected\_year\_data(),  
 FUN = mean)  
 })  
   
 output$avg\_quiz.plot = renderPlot({  
 ggplot(avg\_scores(), aes(  
 x = factor(QUIZ\_NUMBER),  
 y = QUIZ\_SCORE,  
 fill = PROG  
 )) +  
 geom\_bar(stat = "identity", position = "dodge", alpha = .3) +  
 labs(x = "Quiz Number", y = "Average Quiz Score", title = "Average Quiz Scores by Quiz Number") +  
 scale\_fill\_discrete(name = "Student Level") +  
 scale\_fill\_manual(values = c("blue", "purple")) +  
 theme\_minimal()  
 })  
   
   
 # AVERAGE TEMP PER SESH  
 temp\_data <- quiz %>%  
 filter(YEAR == 2017) %>%  
 group\_by(ACAD\_DATE\_KEY) %>%  
 summarise(average\_temp = mean(AVG\_T\_CLICKER, na.rm = TRUE), .groups = 'drop') %>%  
 mutate(ACAD\_DATE\_KEY = as.Date(as.character(ACAD\_DATE\_KEY), "%Y%m%d"))  
   
   
 # Plot average temperature  
 output$average\_temp.plot <- renderPlot({  
 ggplot(temp\_data, aes(x = factor(ACAD\_DATE\_KEY), y = average\_temp)) +  
 geom\_bar(stat = "identity", fill = "blue", alpha = .3) +  
 labs(title = "Average Temperature of Students Per Session",  
 x = "Session Date",  
 y = "Average Temperature") +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))  
 })  
   
   
 # AVERAGE ATTENDANCE PER SESSION  
 attendance\_data <- quiz %>%  
 filter(YEAR == 2017) %>%  
 group\_by(ACAD\_DATE\_KEY) %>%  
 summarise(average\_attendance = mean(ATTENDED, na.rm = TRUE), .groups = 'drop') %>%  
 mutate(ACAD\_DATE\_KEY = as.Date(as.character(ACAD\_DATE\_KEY), "%Y%m%d"))  
   
 # Plot average attendance  
 output$average\_attendance.plot <- renderPlot({  
 ggplot(attendance\_data, aes(x = factor(ACAD\_DATE\_KEY), y = average\_attendance)) +  
 geom\_bar(stat = "identity", fill = "blue", alpha = .3) +  
 labs(title = "% of Students in Attendence",  
 x = "Session Date",  
 y = "Average Attendance") +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))  
 })  
   
   
   
 #STUDENT PLOTS  
   
 output$clicker\_prog\_student.plot = renderPlot({  
 #get the data of one student ^  
 plot <-  
 data.frame(  
 x = selected\_student$QUIZ\_NUMBER,  
 y1 = selected\_student$TOTAL\_POSSIBLE\_CLICKER,  
 y2 = selected\_student$TOTAL\_COMPLETED\_CLICKER,  
 y3 = selected\_student$CORRECT\_Q\_CLICKER  
 )  
 melted = melt(plot, id = 'x')  
 ggplot(melted, aes(x = x, y = value, fill = variable)) +  
 geom\_bar(width = 0.5,  
 stat = 'identity',  
 alpha = 0.3) +  
 scale\_fill\_manual(  
 name = 'Legend',  
 values = c('gray', 'blue', 'green'),  
 labels = c('Total', 'Total Completed', 'Total Correct')  
 ) +  
 labs(title = "Clicker Progress Bar",  
 x = "Quizes",  
 y = "Number of Questions") +  
 coord\_flip() +  
 theme\_minimal()  
 })  
   
   
 output$temp\_student.plot = renderPlot({  
 student\_data <- selected\_student  
   
 # Convert to character before converting to Date  
 date\_character <- as.character(student\_data$ACAD\_DATE\_KEY)  
   
 # Convert character to Date object with specified format  
 student\_data$ACAD\_DATE\_KEY <-  
 as.Date(date\_character, format = "%Y%m%d")  
   
 # shortened date  
 student\_data$shorten\_date <-  
 format(student\_data$ACAD\_DATE\_KEY, "%m/%d")  
   
 ggplot(student\_data, aes(x = shorten\_date, y = AVG\_T\_CLICKER)) +  
 geom\_bar(stat = "identity", fill = "blue", alpha = .3) +  
 labs(x = "Session Date", y = "Average Temperature Score") +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))  
 })  
   
   
 output$quiz\_score.plot = renderPlot({  
 student\_data <- selected\_student  
   
 # cut-off set to 60%, which is 12  
 cutoff <- 12  
   
 # Create a new column to indicate whether QUIZ\_SCORE is below the cutoff  
 student\_data$colorful <-  
 ifelse(student\_data$QUIZ\_SCORE > cutoff, "green", "blue")  
   
 # using ggplot  
 ggplot(student\_data, aes(  
 x = factor(SESSION\_NUMBER),  
 y = QUIZ\_SCORE,  
 fill = colorful  
 )) +  
 geom\_bar(stat = "identity", alpha = .3) + # makes the density line and gray fill  
 scale\_fill\_manual(  
 name = "Passing or Not",  
 values = c("green", "blue"),  
 labels = c("Pass", "Below Passing")  
 ) + # specify colors and legend title  
 # scale\_fill\_manual(values = c("blue", "red")) + # specify colors manually  
 labs(y = "Quiz Score", x = "Session") + # change axis labels  
 theme\_classic() +  
 coord\_flip()  
 })  
   
 specific\_student <- 105  
 experience[is.na(experience)] <- 0  
   
 # filter experience dataset for the specified years (2016, 2017, 2018)  
 exp\_2016 <- filter(experience, YEAR == 2016)  
 exp\_2017 <- filter(experience, YEAR == 2017)  
 exp\_2018 <- filter(experience, YEAR == 2018)  
   
 # calculate quantiles for SKILL\_SURVEY\_SCORE to define cohorts  
 quantiles\_2016 <- quantile(exp\_2016$SKILL\_SURVEY\_SCORE, probs = c(0.33, 0.66, 1))  
 quantiles\_2017 <- quantile(exp\_2017$SKILL\_SURVEY\_SCORE, probs = c(0.33, 0.66, 1))  
 quantiles\_2018 <- quantile(exp\_2018$SKILL\_SURVEY\_SCORE, probs = c(0.33, 0.66, 1))  
   
 # define cohorts based on quantiles and list students in each cohort  
 cohort\_2016\_bot <- exp\_2016 %>%  
 filter(SKILL\_SURVEY\_SCORE <= quantiles\_2016[1]) %>%  
 pull(STUDENT\_KEY)  
   
 cohort\_2016\_mid <- exp\_2016 %>%  
 filter(SKILL\_SURVEY\_SCORE > quantiles\_2016[1] & SKILL\_SURVEY\_SCORE <= quantiles\_2016[2]) %>%  
 pull(STUDENT\_KEY)  
   
 cohort\_2016\_top <- exp\_2016 %>%  
 filter(SKILL\_SURVEY\_SCORE > quantiles\_2016[2]) %>%  
 pull(STUDENT\_KEY)  
   
 cohort\_2017\_bot <- exp\_2017 %>%  
 filter(SKILL\_SURVEY\_SCORE <= quantiles\_2017[1]) %>%  
 pull(STUDENT\_KEY)  
   
 cohort\_2017\_mid <- exp\_2017 %>%  
 filter(SKILL\_SURVEY\_SCORE > quantiles\_2017[1] & SKILL\_SURVEY\_SCORE <= quantiles\_2017[2]) %>%  
 pull(STUDENT\_KEY)  
   
 cohort\_2017\_top <- exp\_2017 %>%  
 filter(SKILL\_SURVEY\_SCORE > quantiles\_2017[2]) %>%  
 pull(STUDENT\_KEY)  
   
 cohort\_2018\_bot <- exp\_2018 %>%  
 filter(SKILL\_SURVEY\_SCORE <= quantiles\_2018[1]) %>%  
 pull(STUDENT\_KEY)  
   
 cohort\_2018\_mid <- exp\_2018 %>%  
 filter(SKILL\_SURVEY\_SCORE > quantiles\_2018[1] & SKILL\_SURVEY\_SCORE <= quantiles\_2018[2]) %>%  
 pull(STUDENT\_KEY)  
   
 cohort\_2018\_top <- exp\_2018 %>%  
 filter(SKILL\_SURVEY\_SCORE > quantiles\_2018[2]) %>%  
 pull(STUDENT\_KEY)  
   
 # find the appropriate cohort based on specific\_student ID  
 if (specific\_student %in% cohort\_2016\_bot) {  
 cohort\_students <- cohort\_2016\_bot  
 } else if (specific\_student %in% cohort\_2016\_mid) {  
 cohort\_students <- cohort\_2016\_mid  
 } else if (specific\_student %in% cohort\_2016\_top) {  
 cohort\_students <- cohort\_2016\_top  
 } else if (specific\_student %in% cohort\_2017\_bot) {  
 cohort\_students <- cohort\_2017\_bot  
 } else if (specific\_student %in% cohort\_2017\_mid) {  
 cohort\_students <- cohort\_2017\_mid  
 } else if (specific\_student %in% cohort\_2017\_top) {  
 cohort\_students <- cohort\_2017\_top  
 } else if (specific\_student %in% cohort\_2018\_bot) {  
 cohort\_students <- cohort\_2018\_bot  
 } else if (specific\_student %in% cohort\_2018\_mid) {  
 cohort\_students <- cohort\_2018\_mid  
 } else if (specific\_student %in% cohort\_2018\_top) {  
 cohort\_students <- cohort\_2018\_top  
 } else {  
 # handle case where specific\_student does not match any cohort  
 # this should never happen though  
 stop("Specific student ID not found in any cohort.")  
 }  
   
 # specific\_student's quiz data  
 quiz\_data <- reactive({  
 filter(quiz, STUDENT\_KEY == specific\_student & YEAR %in% c(2016, 2017, 2018))  
 })  
   
 # specific\_student's cohort data  
 cohort\_data <- reactive({  
 quiz %>%  
 filter(STUDENT\_KEY %in% cohort\_students)  
 })  
   
 # specific\_student's average quiz score  
 avg\_quiz\_score <- reactive({  
 mean(quiz\_data()$QUIZ\_SCORE, na.rm = TRUE)  
 })  
   
 # the histogram plot  
 output$cohortHistogram <- renderPlot({  
 ggplot(cohort\_data(), aes(x = QUIZ\_SCORE)) +  
 geom\_histogram(binwidth = 2, color = "black", fill = "blue", alpha = .3) +  
 geom\_vline(xintercept = avg\_quiz\_score(), color = "blue", linetype = "dashed", size = 1) +  
 labs(x = "Quiz Score", y = "Frequency") +  
 theme\_minimal()  
 })  
   
   
  
   
#######################################  
#######################################  
   
}

# Part 5: Produce Dashboard and Reflect

You should be able to simply run the code below **as is** to see your dashboard.

**Note:** Unfortunately, you cannot knit this part into a pdf. So I added eval=FALSE to let the knitting run smoothly and you can submit your PDF.

#######################################  
### This code creates the dashboard ###  
#######################################  
  
# Here we set up the Header of the dashboard  
dhead = dashboardHeader(title = "Clicker Dashboard")  
  
# Here set up the sidebar which has links to two pages  
dside = dashboardSidebar(  
 sidebarMenu(  
 menuItem("Instructor View", tabName = "instructor", icon = icon("dashboard")),  
 menuItem("Student View", tabName = "student", icon = icon("th"))  
 )  
)  
  
# Here we set up the body of the dashboard  
dbody = dashboardBody(  
 tabItems(  
 student\_dash,  
 instructor\_dash  
 )  
)  
  
# Combining header, sidebar, and body  
ui = dashboardPage(dhead, dside, dbody, skin = "black")  
  
# Generating a local instance of your dashboard  
shinyApp(ui, server)

**Question 5:** Add screenshots of your group’s dahsboards below using this syntax or simply add them to the Word document after knitting:

A screenshot of a dashboard

Description automatically generated

|  |
| --- |
| Dashboard Screenshot Teacher |

A screenshot of a graph

Description automatically generated

|  |
| --- |
| Dashboard Screenshot Teacher pt.2 |

A screenshot of a dashboard

Description automatically generated

|  |
| --- |
| Dashboard Screenshot Student |

A screenshot of a graph

Description automatically generated

|  |
| --- |
| Dashboard Screenshot Student pt.2 |

**Question 6:** Evaluate your group dashboard from the perspective of the instructor (teacher dashboard) and from the perspective of the student (student dashboard). What do you like about it, what would you change or add to it if you had more time?

*Reflection for the student dashboard*

* What do you like about it?
  + The dashboard has a clear, intuitive,  student-friendly design that highlights essential information at a glance focusing on grades/performance and attendance  so far. The dashboard is personalized to every student this way.
  + The dashboard uses a variety of visualizations like bar charts and value boxes that effectively communicate student performance metrics.
* What would you change or add to it if you had more time?
  + More interactive features like filtering by quiz topics, sessions, or score ranges or seeing performance compared to the median.
* What was the biggest challenge you faced? How did you address it?
  + We had to learn how to use Rshiny and ggplot. Went through some videos to learn syntax and we also looked up the documentation. We also used hiTA to help with the Rshiny and ggplot.

*Reflection for the teacher dashboard*

* What would you change or add to it if you had more time?
  + Next time, we would try to make better predictive analysis tools to help teachers identify students who may need more help before the next quiz or bigger assignments.
  + We would try to make categorical breakdowns of weaker topics that students seem to have most problems with to maybe re-teach in class.
* What was the biggest challenge you faced? How did you address it?
  + We have trouble with our data transformation and the use of RShiny and ggplot for the box plot of scores of students.
* What do you like about it?
  + This project gave us a good overview of the whole class’ progress.
  + We liked how it gave us an opportunity to try and fix the problems we have in our classes.

# Self-reflection

**Briefly summarize your experience on this homework. What was easy, what was hard, what did you learn?**

* We enjoyed working as a group- everyone brings their own strengths, their own prior coding experience and knowledge to spearhead their work. Using Rshiny was a first for a lot of us and figuring it out took us a while.

# Estimate time spent

**We want to give students an estimate of how much time this homework will take. Please indicate how many hours you spent to complete this homework here.**

* We spent 10 hours.

# Generative AI usage

**As stated in the course syllabus, using generative AI is allowed to help you as you complete this homework. We are interested in how it is being used and whether it is helpful for you.**

* How much did you use generative AI (e.g., not at all, some, most, or all the questions) and which one did you use?
* We used HiTA to help us get assignment specific clarity on visualizing the engineered features like quiz score, attendance, and the student cohorts.
* If you used generative AI, how did you use it and was it helpful?
* We used it to understand RShiny documentation and it was very helpful in ntegrating knowledge of data transformation / engineered metrics into a visualization.

# Submit Homework

This is the end of the homework. Please **Knit to Word**. The resulting file has to show both the R code and R output. Upload it on the EdX platform before the due date.