Dashboards for Clicker Data

INFO 4100 Learning Analytics

Geoffrey Miller, gom6

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?

The dashboard is for college students in a CS class on business intelligence. They have likely been in several college classes before, they participate in clicker questions and quizzes within this specific class, and are expected to attend each class.

* Why? What is the goal? What questions to answer? Enable students to succeed in the class and improve their performance Make it easy and accessible for students to track their progress within the class, predict their future performance, and gain insights on how to succeed in the class.
* What? What data to show and what is its structure? The data contains a wide variety of insights into the class, including the quiz number and corresponding score, total number of completed clicker questions, etc for each student The data includes insights from many different years and sessions. Each row represents a student.
* How? How will visualizations support the goal? The visualizations will identify trends/patterns within the data. For example, seeing correlation between the number of clicker questions attempted and quiz score encourages the student to participate in class in order to receive a better grade Box plots, scatter plots, pie charts, etc can all be effective visualizations that are easy for instructor & students to understand

*Planning for the teacher dashboard*

* For whom? Who will use it and what is their background? A college instructor who is teaching a CS class on business intelligence. She likely has taught several college classes before and needs a way to effectively monitor overall student progress, performance, engagement, etc in an easy, reliable way.
* Why? What is the goal? What questions to answer? The goal is for the instructor to be able to monitor student progress and gain a solid understanding of the activities of students within her class Another goal is for the dashboard to be easy to understand and accurate so that the instructor can make any changes to the class/keep on doing what improves student performance.
* What? What data to show and what is its structure? The data includes a wide variety of information about student progress and performance, including the quiz number and corresponding score, total number of completed clicker questions, etc for each student The data is organized into a table and each row represents a student
* How? How will visualizations support the goal? The visualizations will condense, analyze, and then present the data in an organized manner in order for the instructor to gain a better understanding of the overall student progress and performance within her class The visualizations will include a wide variety of graphs that provide different types of insights for the instructor, including student attendance, percentage of clicker questions answered correctly, etc.

**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).

# 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"),  
   
fluidRow(  
 ),  
 # Any visualization  
 fluidRow(  
 box(  
 title = "Box 1",  
 plotOutput("inst.plot1", height = 250)  
 ),  
 box(  
 title = "Box 2",  
 plotOutput("inst.plot2", height = 250)  
 ),  
 box(  
 title = "Box 3",  
 plotOutput("inst.plot3", height = 250)  
 ),  
 box(  
 title = "Box 4",  
 plotOutput("inst.plot4", height = 250)  
 )  
 )  
)  
  
# Another empty tab  
student\_dash = tabItem(  
 tabName = "student",  
 h2("Student Dashboard"),  
   
 # Dynamic infoBoxes  
 fluidRow(  
 ),  
 # Any visualization  
 fluidRow(  
 box(  
 title = "Box 1",  
 plotOutput("stud.plot1", height = 250)  
 ),  
 box(  
 title = "Box 2",  
 plotOutput("stud.plot2", height = 250)  
 ),  
 box(  
 title = "Box 3",  
 plotOutput("stud.plot3", height = 250)  
 ),  
 box(  
 title = "Box 4",  
 plotOutput("stud.plot4", height = 250)  
 )  
 )  
)  
#######################################  
#######################################

# 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 #######  
#######################################  
  
#Student pre-processing  
  
# iClicker completion percentage for each session  
# Nothing needed  
  
# iClicker completion percentage per session  
# Nothing needed  
  
# percent of classes attended pie chart on the student dashboard  
quiz |>  
 group\_by(STUDENT\_KEY) |>  
 filter(STUDENT\_KEY == 1) |>  
 summarise(  
 total = n(),  
 attend\_pct = sum(ATTENDED == 1) / total,  
 n\_attend\_pct = 1 - attend\_pct,  
 ) -> attend\_pcts  
  
col2 <- c(attend\_pcts$attend\_pct, attend\_pcts$n\_attend\_pct)  
  
col1 <- c("Attended", "Did not attend")  
  
class\_pct <- data.frame(col1,col2)  
  
# total possible clicker, quiz score scatter plot on the student dashboard  
# No processing was needed for this  
  
# iClicker completion percentage for each session  
quiz$completion\_percentage <- quiz$TOTAL\_COMPLETED\_CLICKER / quiz$TOTAL\_POSSIBLE\_CLICKER \* 100  
  
# iClicker completion percentage per session  
attendance\_rate <- quiz %>%  
 group\_by(SESSION\_NUMBER) %>%  
 summarise(attendance\_rate = mean(ATTENDED, na.rm = TRUE))  
  
  
#Inst Plot 1  
attendance\_rate1 <- quiz %>%  
 group\_by(YEAR, QUIZ\_NUMBER) %>%  
 summarise(Attendance\_Rate = mean(ATTENDED))

## `summarise()` has grouped output by 'YEAR'. You can override using the `.groups`  
## argument.

#INST PLOT 2  
vg\_scores <- experience %>%  
 summarise(across(contains("\_SCORE"), mean, na.rm = TRUE)) %>%  
 pivot\_longer(cols = contains("\_SCORE"), names\_to = "Score\_Type", values\_to = "Average\_Score")

## Warning: There was 1 warning in `summarise()`.  
## ℹ In argument: `across(contains("\_SCORE"), mean, na.rm = TRUE)`.  
## Caused by warning:  
## ! The `...` argument of `across()` is deprecated as of dplyr 1.1.0.  
## Supply arguments directly to `.fns` through an anonymous function instead.  
##   
## # Previously  
## across(a:b, mean, na.rm = TRUE)  
##   
## # Now  
## across(a:b, \(x) mean(x, na.rm = TRUE))

custom\_labels <- c("DATABASE\_SCORE", "SQL\_SCORE", "PROGRAMING\_SCORE",  
 "STORED\_PROC\_SCORE", "ETL\_SCORE", "DATA\_VIS\_SCORE",  
 "REQUIREMENT\_GATHER\_SCORE", "SKILL\_SURVEY\_SCORE")  
  
#INST PLOT 3  
#Distribution of Quiz Scores by Year and Quiz Number:  
quiz$YEAR <- as.factor(quiz$YEAR)  
quiz$QUIZ\_NUMBER <- as.factor(quiz$QUIZ\_NUMBER)  
  
#INST PLOT 4  
  
 ui <- fluidPage(  
 titlePanel("Quiz Data Analysis"),  
 sidebarLayout(  
 sidebarPanel(  
 selectInput("year", "Select Year", choices = unique(quiz$YEAR)),  
 selectInput("session", "Select Session Number", choices = unique(quiz$SESSION\_NUMBER))  
 ),  
 mainPanel(  
 plotOutput("quizPlot")  
 )  
 )  
)  
   
 #INST 4  
quiz$QUIZ\_NUMBER <- as.factor(quiz$QUIZ\_NUMBER)  
quiz$YEAR <- as.factor(quiz$YEAR)  
#######################################  
#######################################  
  
#######################################  
#######################################

# 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 #######  
#######################################  
   
 output$inst.plot1 = renderPlot({  
 ggplot(attendance\_rate1, aes(x = QUIZ\_NUMBER, y = Attendance\_Rate, group = as.factor(YEAR), color = as.factor(YEAR))) +  
 geom\_line() +  
 geom\_point() +  
 labs(x = "Session Number", y = "Attendance Rate", title = "Attendance Rate by Session Number for Each Year") +  
 scale\_color\_discrete(name = "Year", labels = c("2016", "2017", "2018")) +  
 theme\_minimal()  
 })  
   
   
 output$inst.plot2 = renderPlot({  
ggplot(avg\_scores, aes(x = factor(Score\_Type, levels = custom\_labels), y = Average\_Score)) +  
 geom\_bar(stat = "identity", fill = "orange", color = "black") +  
 labs(x = "Score Type", y = "Average Score", title = "Average Score by Score Type") +  
 scale\_x\_discrete(labels = c("Database Score", "SQL Score", "Programming Score",  
 "Stored Proc Score", "ETL Score", "Data Vis Score",  
 "Requirement Gather Score", "Skill Survey Score")) +  
 theme\_minimal()+  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))  
 })  
   
  
   
 output$inst.plot3 = renderPlot({  
ggplot(quiz, aes(x = QUIZ\_NUMBER, y = QUIZ\_SCORE, fill = YEAR)) +  
 geom\_boxplot() +  
 labs(x = "Quiz Number", y = "Quiz Score", title = "Distribution of Quiz Scores by Year and Quiz Number") +  
 theme\_minimal()  
 })  
   
   
 output$inst.plot4 = renderPlot({  
ggplot(quiz, aes(x = QUIZ\_SCORE, y = TOTAL\_COMPLETED\_CLICKER, color = YEAR)) +  
 geom\_point() +  
 facet\_wrap(~ QUIZ\_NUMBER, scales = "free") +  
 labs(title = "Quiz Scores vs. Completed Clickers",  
 x = "Quiz Score", y = "Total Completed Clicker Questions",  
 color = "Year") +  
 theme\_minimal()  
  
 })  
   
   
 output$stud.plot1 = renderPlot({  
 ggplot(quiz, aes(x = factor(SESSION\_NUMBER), y = completion\_percentage)) +  
 geom\_boxplot(fill = "skyblue", color = "black") +  
 labs(title = "iClicker Completion Percentage Across Sessions",  
 x = "Session Number",  
 y = "Completion Percentage") +  
 theme\_minimal()  
 })  
  
   
 output$stud.plot2 = renderPlot({  
 ggplot(attendance\_rate, aes(x = factor(SESSION\_NUMBER), y = attendance\_rate)) +  
 geom\_bar(stat = "identity", fill = "skyblue", color = "black") +  
 labs(title = "Attendance Rate",  
 x = "Session Number",  
 y = "Attendance Rate") +  
 theme\_minimal()  
 })  
 output$stud.plot3 = renderPlot({  
 quiz |>  
 group\_by(STUDENT\_KEY) |>  
 filter(STUDENT\_KEY == 1) |>  
 ggplot(aes(x = SESSION\_NUMBER, y = TOTAL\_COMPLETED\_CLICKER)) +  
 labs(  
 title = "Completed Clicker Questions",  
 subtitle = "By Class Session",  
 x = "Class Session",  
 y = "Number of Click questions Completed"  
 ) +  
 geom\_point()  
 })  
   
 output$stud.plot4 = renderPlot({  
 class\_pct |>  
 ggplot(aes(x = "", y = col2, fill = col1)) +  
 geom\_bar(stat = "identity") +  
 coord\_polar("y", start=0) +  
 labs(title = "Attendance Percentage by Student",  
 fill = "Status",  
 ) +  
 geom\_text(aes(label = col2)) +   
 scale\_color\_discrete(name = col1) +  
 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)  
  
# 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 computer

Description automatically generated

A screenshot of a computer

Description automatically generated

‘

**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?
  + I like how students can view their attendance and clicker information in an easy, visual format. After a long semester, it is easy to loose track of these important metrics, and the dashboard puts all the information in one place. In addition, students can see their progress in terms of different sections, which can be important as they try to holistically evaluate their progress throughout the semester.
* What would you change or add to it if you had more time?
  + If I had more time, I would want to add interactive features where students can view different metrics filtered by each session. In addition, I would want students to have the ability to predict quiz scores or other information based on their current progress.
* What was the biggest challenge you faced? How did you address it?
  + The biggest challenge we faced was working to find different visualizations to show the complexities of the data.

*Reflection for the teacher dashboard*

* What do you like about it?

I like that it has a wide variety of graphs and that the graphs give information about the different years and session numbers so that the instructor can have a deeper understanding about how the course has progressed over time.

* What would you change or add to it if you had more time?

I would probably add more graphs to it so that the teacher could access even more information about the course. There is potential to even use drop down menus to organize the graphs by different filters.

* What was the biggest challenge you faced? How did you address it?

The biggest challenge we faced was putting ourself in the shoes of the teacher to understand what graphs would be beneficial for them to view as they work on planning the course.