Dashboards for Clicker Data

INFO 5200 Learning Analytics Group Project

[[Kimberly Williamson, khw44]]

This project is about developing a learning analytics dashboard based on clicker data. You will work in a team (see group assignments online) to learn how to make a dashboard using R Shiny.

Learning Objectives

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

Here is the official page with several tutorials (https://shiny.rstudio.com/tutorial/). 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 either a dashboard for the instructor which offers an overview of the class characteristics, engagement, and performance; or 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 Blackboard) 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

Attribute Name	Data Type	Definition
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 (Monday in class)

Go through the planning process described in the reading about dashboards. While some dashboards are certainly better than others, there is not one correct solution here.

Question 1: Decide as a group whether you want to make a student dashboard or a teacher dashboard. Carefully consider the impliations of this choice for the design and content. To plan, answer the following prompts for either the teacher and student 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 may find it helpful to explore the dataset in R for 5-10 minutes to get a good sense of what the dataset has to offer.

* Why? What is the goal? What questions to answer?

The overall goal is to:

(1) provide a summary of course statistics over the years (e.g. total number of students, attendance rate, quiz score distribution, etc.) and give the instructor/students insights into the class composition and student performance by session and year.

(2) Identify what prior skills the instructor should spend more time reviewing and whether pre-req is required for the course.

We try to answer the questions below:

1. What is the correlation between students' prior knowledge / skill set and their performance on the class? -&gt; We can recommend the professor to set a pre-req if necessary
2. What is the correlation between attendance rate and student performance?
3. How do students perform in each session each year?
4. How do the course statistics (e.g. quiz score distribution) differ by years?

* For whom? Who will use it and what is their background?

We decided to create a dashboard for the instructor.

Instructor Background: professor in computer science and business intelligence

* What? What data to show and what is its structure?

We use both the quiz dataset and experience dataset to create our visualizations.

1. The experience dataset contains the survey data on students' prior knowledge/skill set. It consists of the overall score students received in the prerequisite survey and the score for each skill per student. The dataset allows us to aggregate the overall survey score / individual skill score by year.
2. The quiz dataset contains information about student performance and attendance. It consists of the session data (e.g. attendance, clicker question data) and quiz performance data for each student from different years. The dataset allows us to aggregate statistics of attendance, clicker question data, quiz scores, etc. by session / year.

* How? How will visualizations support the goal?

To satisfy the goal of providing an overview of the course statistics, we will visualize the items below:

(1) Total number of students

(2) Average Attendance Rate

We will create graphs answering the questions below:

1. What is the correlation between students' prior knowledge / skill set and their performance on the class?

(1) Use a scatter plot and linear regression to show the correlation between students' skill set survey scores and quiz scores

(2) Use a box plot to identify which specific skills are highly correlated to the quiz scores

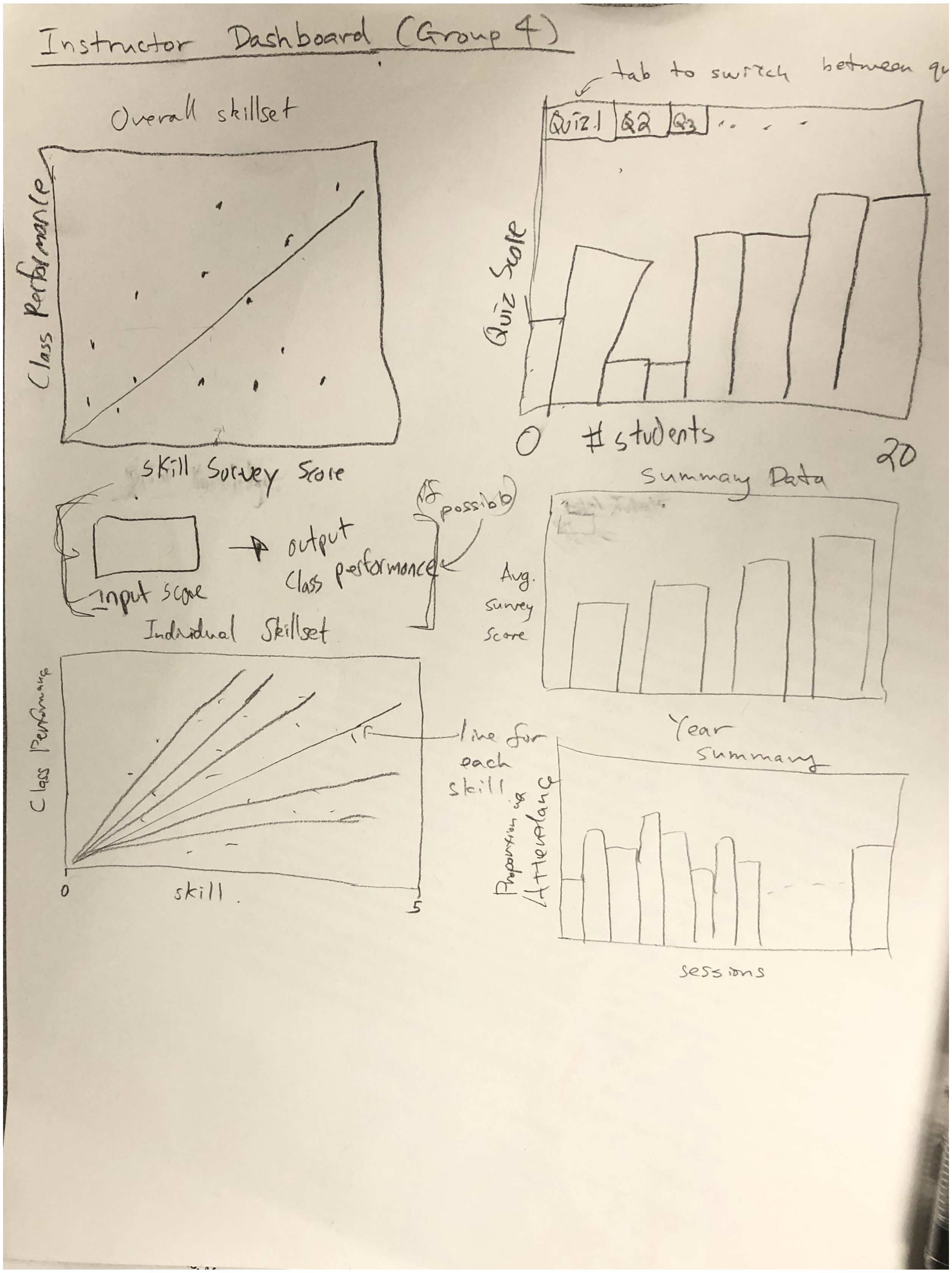
1. What is the correlation between attendance rate and student performance?

(1)Use a scatter plot and linear regression to show the correlation between students' attendance rate and quiz scores

1. How do students perform in each session each year?
2. (1)Create interactive graphs (e.g. histograms) which allows the instructor to check out and compare the session data, such as attendance rate, clicker question accuracy, average temperature, from different years.
3. How do the course statistics (e.g. quiz score distribution) differ by years?
4. (1)Create interactive graphs which allows the instructor to check out and compare statistics from different years (e.g. the instructor can select a specific year and see the histogram of score distribution for each quiz)

Question 2: Based on your plan above, make a sketch of what the dashboard would look like. See this week's reading for an example. 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, becasue 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 sketch to the TAs on Monday. By the end of class, you should have a robust plan to start implementing on Wednesday during class.

- UX:
  - We need to label the x axis and y axis for each graph, and other objects such as titles etc. Avoid inaccessible color combination
- Main page:
  - We want to see overall trends of the students' performance
  - First, we want to see how skill survey scores and average quiz scores are correlated, and then how each skill set correlates with the average quiz scores.
  - Second, we want to see if there is any correlation between attendance and quiz performance.
- By year:
  - The sum of skill set survey scores
  - Each year, how students perform on each quiz
  - Attendance rate by year
- By session:
  - We want to show clicker activities, attendance data using bar chart per session per year.
  - To have a better understanding of student's performance, we also add deviation from average score.



# 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.

If you like, your team can split up so that some of you 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 instructor dashboard tab. Here you are just specifying the wire frame i.e. **what goes where on the page**.

```
# Instructor Dashboard Tab
my_tab = tabItem(
    tabName = "mytab",
    h2("Instructor Dashboard"),

#######################################
####### BEGIN INPUT: Question 3 #######
#######################################
  h3("Overview"),
    # Dynamic infoBoxes
    fluidRow(
      infoBoxOutput("inst.info1"),
      infoBoxOutput("inst.info2")
    ),
    # Any visualization

     fluidRow(
        box(
            title = "Skills VS Quiz Performance",
            plotOutput("inst.plot1", height = 250)
        ),
        box(
          title = "Attendance vs Quiz Performance",
          plotOutput("inst.attendance_performance", height = 250)
      )
    ),
  fluidRow(
    box(
     title = "Quiz Performance By Each Skill Set",
            tabsetPanel(

              type="tabs",
              tabPanel("Database",plotOutput("tab.1")),
              tabPanel("SQL",plotOutput("tab.2")),
              tabPanel("Programming",plotOutput("tab.3")),
              tabPanel("Stored Procedure",plotOutput("tab.4")),
              tabPanel("ETL",plotOutput("tab.5")),
              tabPanel("Data Vis",plotOutput("tab.6"))
            )
     )
  )
#######################################
```

```
#######################################

)

by_year = tabItem(
    tabName = "By Year",
    h2("Instructor Dashboard"),

    # Any visualization
h3("Year Overview"),
  p("Select the Years"),
    checkboxGroupInput("inCheckboxGroup", "Input checkbox",
    c(2016, 2017, 2018), selected = c(2016, 2017, 2018)),
    fluidRow(
        box(
            title = "Skills",
            plotOutput("year.plot1", height = 300)
        ),
         box(
        title = "Quiz Score Distribution",
        selectInput("select_quiz", "Quiz Number",
                c("Quiz 1" = 1,
                  "Quiz 2" = 2,
                  "Quiz 3" = 3,
                  "Quiz 4" = 4,
                  "Quiz 5" = 5,
                  "Quiz 6" = 6,
                  "Quiz 7" = 7,
                  "Quiz 8" = 8,
                  "Quiz 9" = 9,
                  "Quiz 10" = 10)
                  ),
        plotOutput("quizScore.quizScoreDistribution", height = 300))
    ),
fluidRow(
     box(
            title = "Attendance",
            plotOutput("year.plot4", height = 250)
        )
)
)

by_session = tabItem(
    tabName = "bysession",
    h2("Instructor Dashboard"),

#######################################
####### BEGIN INPUT: Question 3 #######
#######################################
    # Dynamic infoBoxes
    h3("Session Overview"),

    selectInput(
      "inSelectInput",
      p("Select Year"),
      choices = list(
        "2016" = 2016,
        "2017" = 2017,
        "2018" = 2018),
      selected = 2016),

    # Any visualization
    fluidRow(
      box(
            title="attendance percentage per session",
            plotOutput("session.plot3", height = 300)
        ),
      box(
            title="Deviation from Average Attendance Percentage",
            plotOutput("session.plot4", height = 300)
        )
    ),
    fluidRow(
      box(
            title="Clicker Answer Rate (%) per session",
            plotOutput("session.plot5", height = 300)
        ),
      box(
            title="Deviation from Average Clicker Answer Rate (%)",
            plotOutput("session.plot6", height = 300)
        )
    ),
    fluidRow(
      box(
            title="Clicker Question Accuracy Rate (%) per session",
            plotOutput("session.plot7", height = 300)
        ),
      box(
            title="Deviation from Average Clicker Accuracy Rate (%)",
            plotOutput("session.plot8", height = 300)
        )
    ),
    fluidRow(
      box(
        title="Average Temperature Score per session",
        plotOutput("session.plot9", height = 300)
      ),
      box(
        title="Deviation from Average Temperature Score",
        plotOutput("session.plot10", height = 300)
      )
    )
########################################
########################################
)
```

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 student dashboard is typically for focused on an individual student. You can either pick a student (at random or intentionally) up here 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             #######
########################################
quiz %>% filter(STUDENT_KEY==1)
```

```
##    ACAD_DATE_KEY STUDENT_KEY YEAR SESSION_NUMBER QUIZ_NUMBER ATTENDED
## 1       20181106           1 2018             23          10        0
## 2       20181018           1 2018             19           8        1
## 3       20181016           1 2018             18           8        0
## 4       20180906           1 2018              5           4        1
## 5       20180927           1 2018             12           7        0
## 6       20180904           1 2018              4           3        1
## 7       20180920           1 2018              9           6        1
## 8       20180911           1 2018              6           5        0
## 9       20181002           1 2018             13          NA        1
## 10      20181004           1 2018             16          NA        0
## 11      20180830           1 2018              3           2        1
## 12      20180918           1 2018              8           6        1
## 13      20180925           1 2018             11           7        1
## 14      20181023           1 2018             20           9        1
## 15      20181025           1 2018             21           9        0
## 16      20180828           1 2018              2           1        1
##    TOTAL_POSSIBLE_CLICKER TOTAL_COMPLETED_CLICKER COMPLETED_Q_CLICKER
## 1                       3                       0                   0
## 2                       4                       3                   2
## 3                       1                       0                   0
## 4                       6                       3                   1
## 5                       2                       0                   0
## 6                       5                       6                  10
## 7                       1                       1                   0
## 8                       6                       0                   0
## 9                       4                       2                   1
## 10                      1                       0                   0
## 11                      3                       3                   3
## 12                      5                       3                   3
## 13                      4                       2                   2
## 14                      5                       5                   6
## 15                      4                       0                   0
## 16                      3                       3                   1
##    CORRECT_Q_CLICKER COMPLETED_T_CLICKER AVG_T_CLICKER QUIZ_SCORE
## 1                     0                    0              0          17
## 2                     1                    1              3          20
## 3                     0                    0              0          20
## 4                     0                    2              3          20
## 5                     0                    0              0          17
## 6                     4                    0              0          18
## 7                     0                    1              3          15
## 8                     0                    0              0          20
## 9                     1                    1              4          NA
## 10                    0                    0              0          NA
## 11                    2                    0              0          20
## 12                    0                    1              3          15
## 13                    1                    0              0          17
## 14                    1                    2              2          20
## 15                    0                    0              0          20
## 16                    1                    2              4          20
```

```
joinTable = quiz %>% group_by(STUDENT_KEY) %>%
        summarise(totalScore=sum(QUIZ_SCORE, na.rm = TRUE),
                  mean_quiz = mean(QUIZ_SCORE, na.rm = TRUE)) %>%
        inner_join(experience,by=c("STUDENT_KEY")) %>%
        gather("Test","PrevScore",
               -c("YEAR","STUDENT_KEY","PROG","totalScore","REQUIREMENT_GATHER_SCORE",
                  "SKILL_SURVEY_SCORE","mean_quiz"))

quiz_clicker = quiz %>%
  filter(TOTAL_COMPLETED_CLICKER == COMPLETED_Q_CLICKER + COMPLETED_T_CLICKER) %>% #completed quiz numb
  filter(TOTAL_POSSIBLE_CLICKER >= TOTAL_COMPLETED_CLICKER) #completed questions cannot be larger than

#Session data
#functions
session_data_manipulaiton_by_year = function(year) {
  session = quiz %>%
  group_by(SESSION_NUMBER) %>%
  filter(YEAR == year) %>%
  summarise(
    attendance_percentage = sum(ATTENDED) / length(ATTENDED) * 100, #attendance for each session
  )
  return(session)
}

#Clicker Overview
clicker_overview_manipulation_by_year = function(year) {
  clicker_overview = quiz_clicker %>%
  group_by(SESSION_NUMBER) %>%
  filter(YEAR == year) %>%
  summarise(
    overall_clicker_answer_rate = mean(TOTAL_COMPLETED_CLICKER / TOTAL_POSSIBLE_CLICKER)
  )
  return(clicker_overview)
}

#Clicker Data -> Questions
clicker_question_manipulation_by_year = function(year) {
  clicker_question = quiz_clicker %>%
  group_by(SESSION_NUMBER) %>% #group by session number
  filter(YEAR == year) %>%
  filter(ATTENDED == 1) %>% #only students who attended
  filter(COMPLETED_Q_CLICKER != 0) %>% #answered at least one question
  summarise(
    avg_clicker_correct_percentage = mean(CORRECT_Q_CLICKER/COMPLETED_Q_CLICKER) * 100 #student correct
  )
  return(clicker_question)
}

#Clicker Questions -> Temperature
clicker_temperature_manipulation_by_year = function(year) {
  clicker_temperature = quiz_clicker %>%
  filter(YEAR == year) %>%
  group_by(SESSION_NUMBER) %>%
  filter(ATTENDED == 1) %>%
  filter(COMPLETED_T_CLICKER != 0) %>% #answered at least one question
  summarise(
    avg_t = mean(AVG_T_CLICKER) #average temperature score per session
  )
  return(clicker_temperature)
}
#######################################
#######################################
```

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 that 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`.

```
# Instructor Dashboard Tab
server = function(input, output) {

#######################################
####### BEGIN INPUT: Question 4 #######
#######################################

    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 = "yellow")
    })
```

```
  output$inst.plot1 = renderPlot({
    plot(joinTable$SKILL_SURVEY_SCORE, joinTable$mean_quiz, main="Scatterplot",
       xlab="skill survey score ", ylab="average quiz score")
    # fit line
    abline(lm(joinTable$mean_quiz~joinTable$SKILL_SURVEY_SCORE), col="red") # regression line (y~x)
  })

  output$inst.attendance_performance = renderPlot({
  attendance_rate = quiz %>%
  group_by(STUDENT_KEY) %>%
  summarise(
    performance = mean(QUIZ_SCORE),
    attendance = sum(ATTENDED)
  ) %>%
  filter(!is.na(performance))
  ggplot(attendance_rate, aes(x=attendance, y=performance)) +
    geom_point(color = "black") + geom_smooth(method=lm, se=FALSE) +
    labs(x = "Total Number of Attendance", y="Average Quiz Score")
})

  output$tab.1 = renderPlot({
    boxplot(mean_quiz ~ PrevScore, data = joinTable %>% filter(Test=="DATABASE_SCORE"),
            xlab = "Survey Score", ylab = "Average Quiz Score")
  })
  output$tab.2 = renderPlot({
    boxplot(mean_quiz ~ PrevScore, data = joinTable %>% filter(Test=="SQL_SCORE"),
            xlab = "Survey Score", ylab = "Average Quiz Score")
  })
  output$tab.3 = renderPlot({
    boxplot(mean_quiz ~ PrevScore, data = joinTable %>% filter(Test=="PROGRAMING_SCORE"),
            xlab = "Survey Score", ylab = "Average Quiz Score")
  })
  output$tab.4 = renderPlot({
    boxplot(mean_quiz ~ PrevScore, data = joinTable %>% filter(Test=="STORED_PROC_SCORE"),
            xlab = "Survey Score", ylab = "Average Quiz Score")
  })
  output$tab.5 = renderPlot({
    boxplot(mean_quiz ~ PrevScore, data = joinTable %>% filter(Test=="ETL_SCORE"),
            xlab = "Survey Score", ylab = "Average Quiz Score")
  })
  output$tab.6 = renderPlot({
    boxplot(mean_quiz ~ PrevScore, data = joinTable %>% filter(Test=="DATA_VIS_SCORE"),
            xlab = "Survey Score", ylab = "Average Quiz Score")
  })
  output$year.plot1 = renderPlot({
    filterYear = experience %>% dplyr::filter(YEAR %in% input$inCheckboxGroup)
    hist(filterYear$SKILL_SURVEY_SCORE,
         main = "Histogram of Total Skill Survey Score",
         ylab="Number of Students",
         xlab="Total Skill Survey Score",
         border="gold",
         col="blue")
  })
  output$year.plot4 = renderPlot({
    ggplot(quiz,aes(as.factor(YEAR), fill=as.factor(ATTENDED))) +
      geom_bar(position = "fill") +
      scale_fill_discrete(
        name = "Attendance",
        labels= c("Absent", "Present"),h = c(80, 260)) +
      xlab("Year") +
      ylab("%")
  })
output$quizScore.quizScoreDistribution = renderPlot({
      selected_quiz = quiz %>% filter(QUIZ_NUMBER == input$select_quiz, YEAR %in% input$inCheckboxGrou
      hist(
        selected_quiz$QUIZ_SCORE,
        main = paste("QUIZ ", input$select_quiz),
        xlab="QUIZ SCORE",
        ylab = "NUMBER OF STUDENTS",
        border="gold",
        col="blue")
})

  #attendance percentage
  output$session.plot3 = renderPlot({

    #filter by year
    session = session_data_manipulaiton_by_year(input$inSelectInput)
    print(session)
    #graph
    ggplot(session, aes(x=SESSION_NUMBER, y=attendance_percentage)) +
      geom_bar(stat = "identity", width = 0.5, fill="blue") +
      theme(axis.text.x = element_text(angle = 90, vjust = 0.5)) +
      scale_x_continuous("Session Number", labels = as.character(session$SESSION_NUMBER), breaks = se
      ylab("Attendance Percentage (%)")
  })

  output$session.plot4 = renderPlot({

    #filter by year
    session_d = session_data_manipulaiton_by_year(input$inSelectInput)
    session_d$ap_z = round((session_d$attendance_percentage - mean(session_d$attendance_percentage))/
    session_d$ap_type = ifelse(session_d$ap_z < 0, "below", "above")
    session_d = session_d[order(session_d$ap_z), ]
    session_d$SESSION_NUMBER = factor(session_d$SESSION_NUMBER, levels = session_d$SESSION_NUMBER)
    ggplot(session_d, aes(x=SESSION_NUMBER, y=ap_z, label=ap_z)) +
        geom_bar(stat="identity", aes(fill=ap_type), width = 0.5) +
        scale_fill_manual(name="Quiz Score",
                          labels = c("Above Average", "Below Average"),
                          values = c("above" = "#d4af37", "below" = "#0000ff")) +
        coord_flip() +
        xlab("Session Number") +
        ylab("Attendance Percentage: z-score")
  })

  #Clicker answer rate
```

```
output$session.plot5 = renderPlot({
  #filter by year
  clicker_overview = clicker_overview_manipulation_by_year(input$inSelectInput)
  #plot graph
  ggplot(clicker_overview, aes(x=SESSION_NUMBER, y=overall_clicker_answer_rate)) +
    geom_bar(stat = "identity", width = 0.5, fill="blue") +
    theme(axis.text.x = element_text(angle = 90, vjust = 0.5)) +
    scale_x_continuous("Session Number",
                       labels = as.character(clicker_overview$SESSION_NUMBER),
                       breaks = clicker_overview$SESSION_NUMBER) +
    ylab("Clicker Answer Rate (%)")
})

output$session.plot6 = renderPlot({
  #filter by year
  clicker_overview_d = clicker_overview_manipulation_by_year(input$inSelectInput)
  clicker_overview_d$ca_z = round(
    (clicker_overview_d$overall_clicker_answer_rate - mean(clicker_overview_d$overall_clicker_answe
  clicker_overview_d$ca_type = ifelse(clicker_overview_d$ca_z < 0, "below", "above")
  clicker_overview_d = clicker_overview_d[order(clicker_overview_d$ca_z), ]
    clicker_overview_d$SESSION_NUMBER = factor(clicker_overview_d$SESSION_NUMBER, levels = clicker_
    ggplot(clicker_overview_d, aes(x=SESSION_NUMBER, y=ca_z, label=ca_z)) +
        geom_bar(stat="identity", aes(fill=ca_type), width = 0.5) +
        scale_fill_manual(name="Clicker Answer Rate (%)",
                          labels = c("Above Average", "Below Average"),
                          values = c("above" = "#d4af37", "below" = "#0000ff")) +
        coord_flip() +
        xlab("Session Number") +
        ylab("Clicker Answer Rate (%) : z-score")
})

#Clicker Question Rate
output$session.plot7 = renderPlot({
  #filter by year
  clicker_question = clicker_question_manipulation_by_year(input$inSelectInput)
  #plot graph
  ggplot(clicker_question, aes(x=SESSION_NUMBER, y=avg_clicker_correct_percentage)) +
    geom_bar(stat = "identity", width = 0.5, fill="blue") +
    theme(axis.text.x = element_text(angle = 90, vjust = 0.5)) +
    scale_x_continuous("Session Number",
                       labels = as.character(clicker_question$SESSION_NUMBER),
                       breaks = clicker_question$SESSION_NUMBER) +
    ylab("Clicker Question Correct (%)")
})

output$session.plot8 = renderPlot({
  #filter by year
  clicker_question = clicker_question_manipulation_by_year(input$inSelectInput)
  #plot graph
  clicker_question_d = clicker_question
  clicker_question_d$cp_z = round(
    (clicker_question_d$avg_clicker_correct_percentage - mean(clicker_question_d$avg_clicker_correc
  clicker_question_d$cp_type = ifelse(clicker_question_d$cp_z < 0, "below", "above")
```

```
      clicker_question_d = clicker_question_d[order(clicker_question_d$cp_z), ]
        clicker_question_d$SESSION_NUMBER = factor(clicker_question_d$SESSION_NUMBER, levels = clicker_
        ggplot(clicker_question_d, aes(x=SESSION_NUMBER, y=cp_z, label=cp_z)) +
            geom_bar(stat="identity", aes(fill=cp_type), width = 0.5) +
            scale_fill_manual(name="Accuracy Rate (%)",
                              labels = c("Above Average", "Below Average"),
                              values = c("above" = "#d4af37", "below" = "#0000ff")) +
            coord_flip() +
            xlab("Session Number") +
            ylab("Clicker Answer Rate (%) : z-score")
    })

    #Temperature
    output$session.plot9 = renderPlot({
      #filter by year
      clicker_temperature = clicker_temperature_manipulation_by_year(input$inSelectInput)
      #plot graph
      ggplot(clicker_temperature, aes(x=SESSION_NUMBER, y=avg_t)) +
        geom_bar(stat = "identity", width = 0.5, fill="blue") +
        theme(axis.text.x = element_text(angle = 90, vjust = 0.5)) +
        scale_x_continuous("Session Number",
                           labels = as.character(clicker_temperature$SESSION_NUMBER),
                           breaks = clicker_temperature$SESSION_NUMBER) +
        ylab("Average Temperature Score")
    })

    output$session.plot10 = renderPlot({
      #filter by year
      clicker_temperature = clicker_temperature_manipulation_by_year(input$inSelectInput)
      #plot graph
      clicker_temperature_d = clicker_temperature
      clicker_temperature_d$at_z = round((clicker_temperature_d$avg_t - mean(clicker_temperature_d$avg_
      clicker_temperature_d$at_type = ifelse(clicker_temperature_d$at_z < 0, "below", "above")
      clicker_temperature_d = clicker_temperature_d[order(clicker_temperature_d$at_z), ]
      clicker_temperature_d$SESSION_NUMBER = factor(clicker_temperature_d$SESSION_NUMBER, levels = clic
      ggplot(clicker_temperature_d, aes(x=SESSION_NUMBER, y=at_z, label=at_z)) +
          geom_bar(stat="identity", aes(fill=at_type), width = 0.5) +
          scale_fill_manual(name="Temperature Score",
                            labels = c("Above Average", "Below Average"),
                            values = c("above" = "#d4af37", "below" = "#0000ff")) +
          coord_flip() +
          xlab("Session Number") +
          ylab("Average Temperature Score: z-score")
    })



#######################################
#######################################

}
```

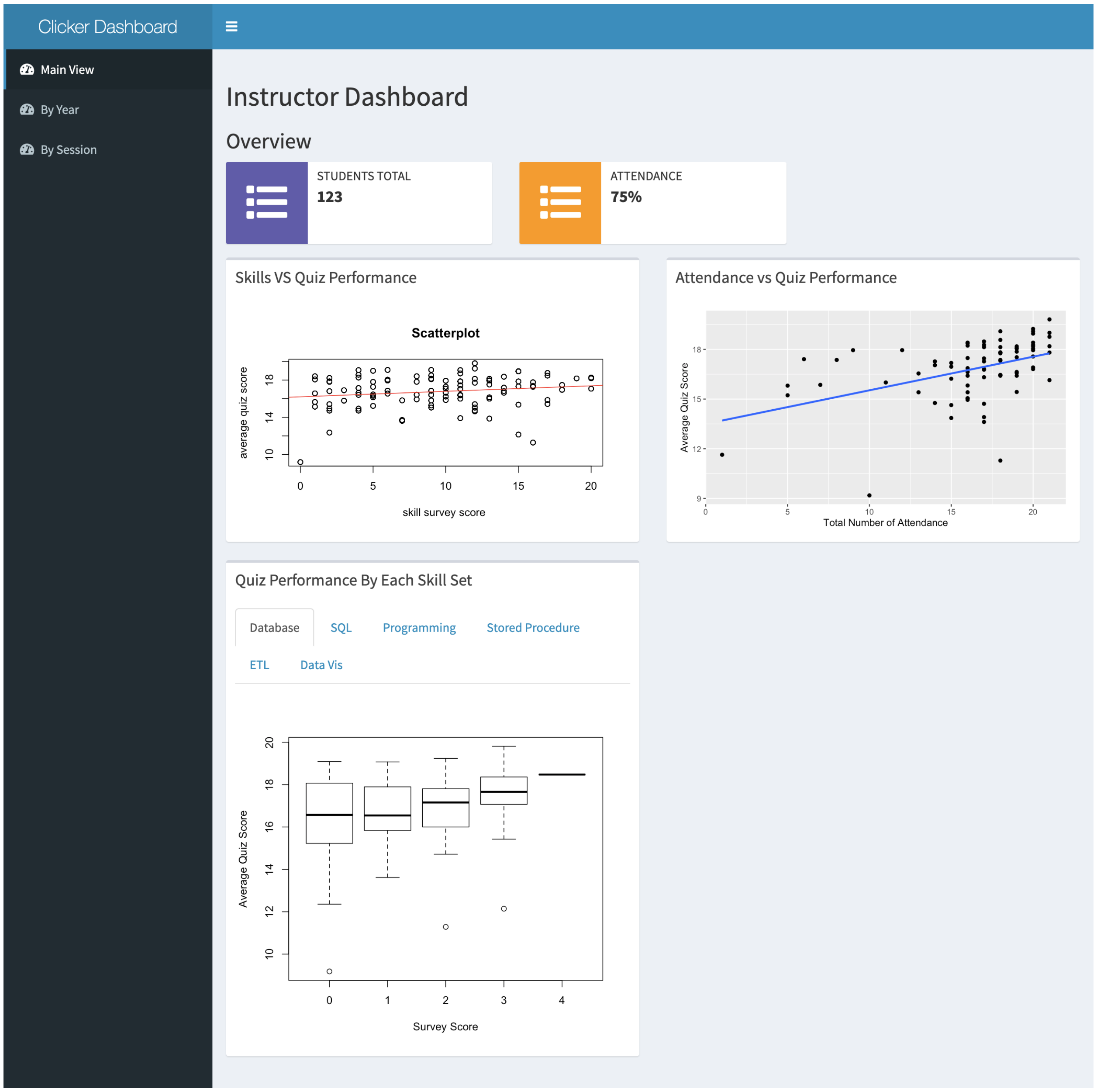
Part 5: Produce Dashboard and Reflect

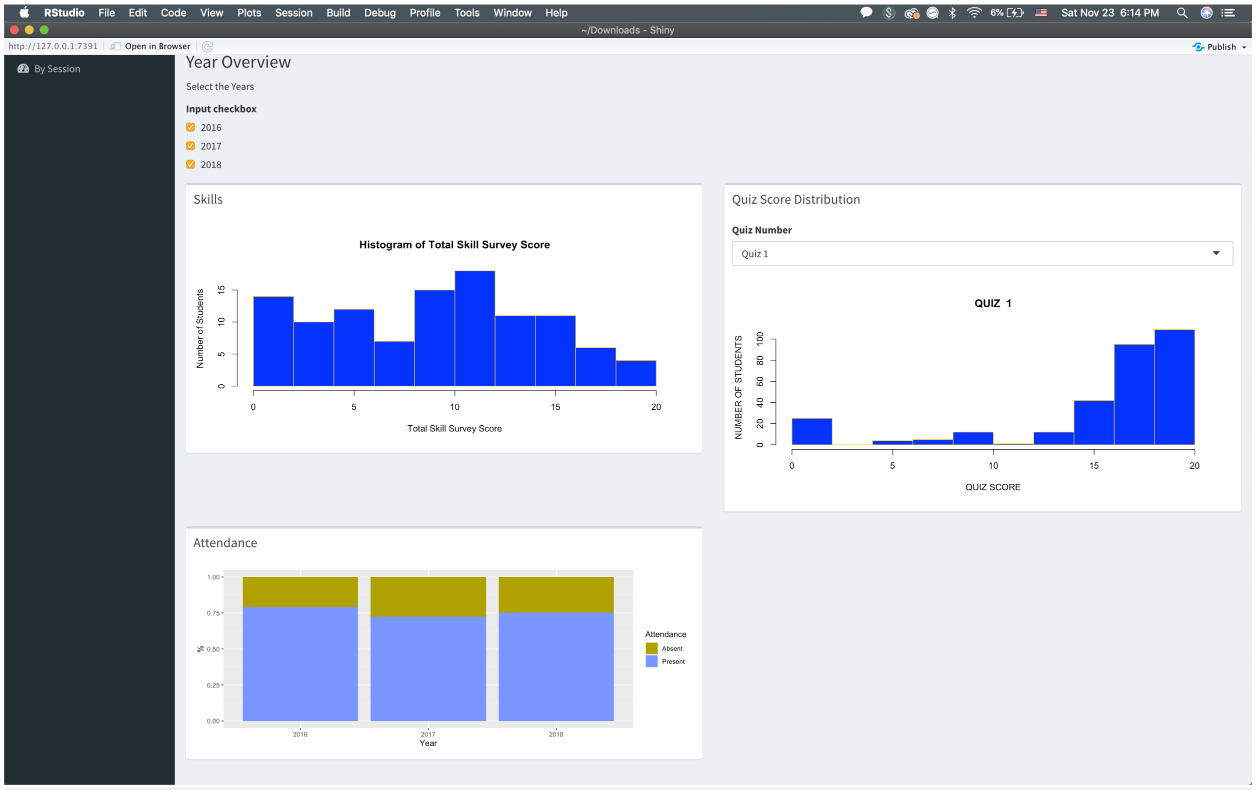
You should be able to simply run the code below **as is** to see your dashboard.

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.

```r
#######################################
### 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 one page
dside = dashboardSidebar(sidebarMenu(
    menuItem("Main View", tabName = "mytab", icon = icon("dashboard")),
    menuItem("By Year", tabName = "by_year", icon = icon("dashboard")),
    menuItem("By Session", tabName = "by_session", icon = icon("dashboard"))
))

# Here we set up the body of the dashboard
dbody = dashboardBody(
    tabItems(
      tabItem(tabName = "mytab", my_tab),
      tabItem(tabName = "by_year", by_year),
      tabItem(tabName = "by_session", by_session)
    )
)

# Combining header, sidebar, and body
ui = dashboardPage(dhead, dside, dbody)

# Generating a local instance of your dashboard
shinyApp(ui, server)
```

Question 5: Add a screenshot of your group's dahsboard here:


Clicker Dashboard
Main View
By Year
By Session
Instructor Dashboard
Overview
STUDENTS TOTAL
123
ATTENDANCE
75%
Skills VS Quiz Performance
Scatterplot
average quiz score
skill survey score
Attendance vs Quiz Performance
Average Quiz Score
Total Number of Attendance
Quiz Performance By Each Skill Set
Database
SQL
Programming
Stored Procedure
ETL
Data Vis
Average Quiz Score
Survey Score



Year Overview

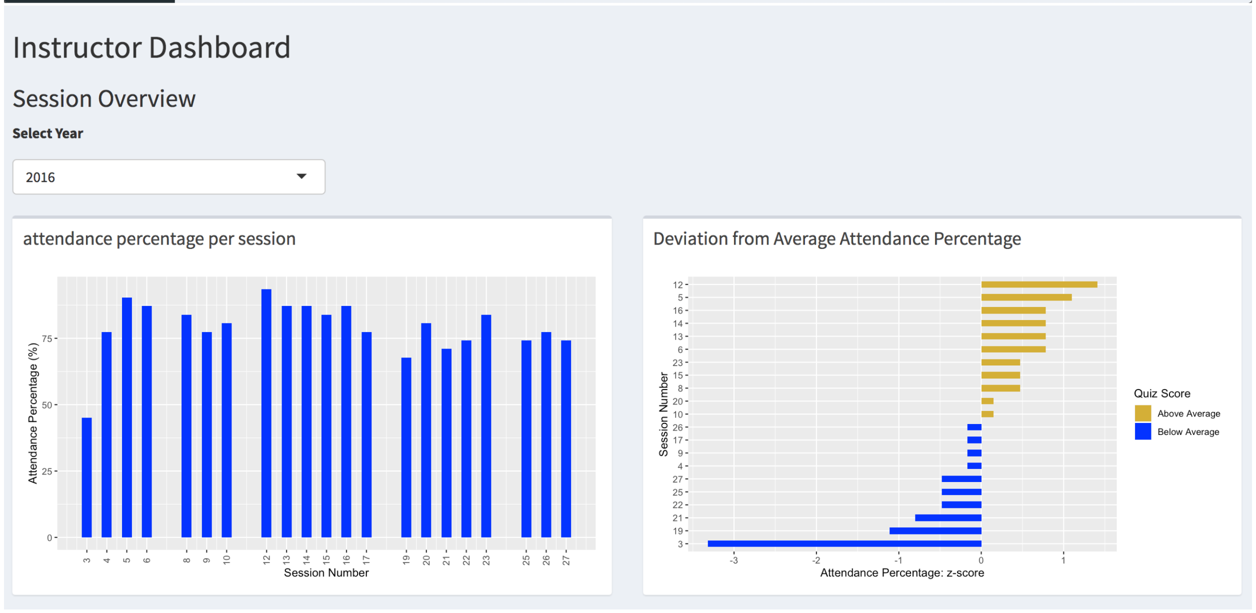
Select the Years

Input checkbox

- ☑ 2016
- ☑ 2017
- ☑ 2018

Quiz Number

Quiz 1

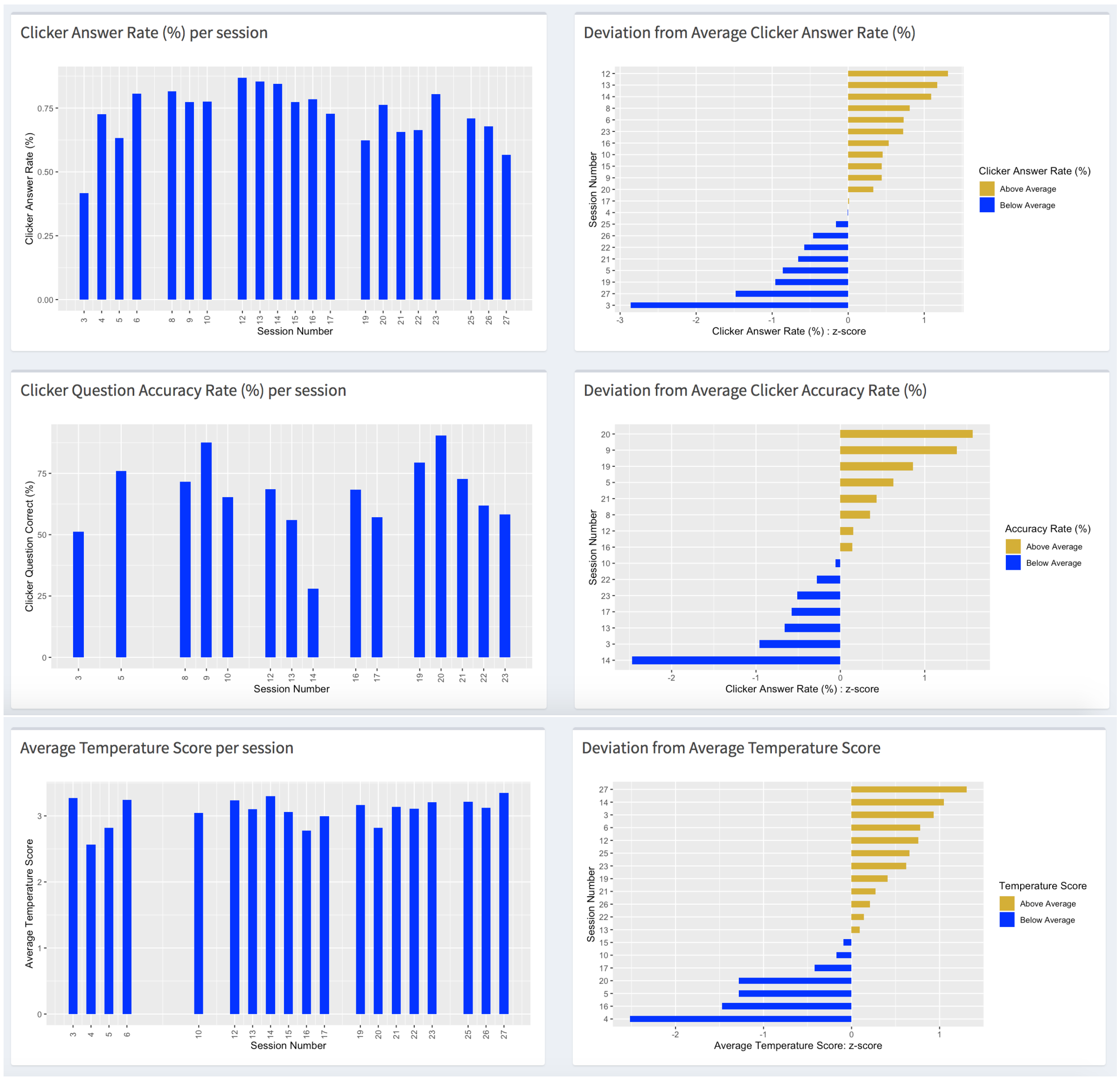

attendance percentage per session
Attendance Percentage (%)
Session Number
Deviation from Average Attendance Percentage
Session Number
Attendance Percentage: z-score
Quiz Score
Above Average
Below Average

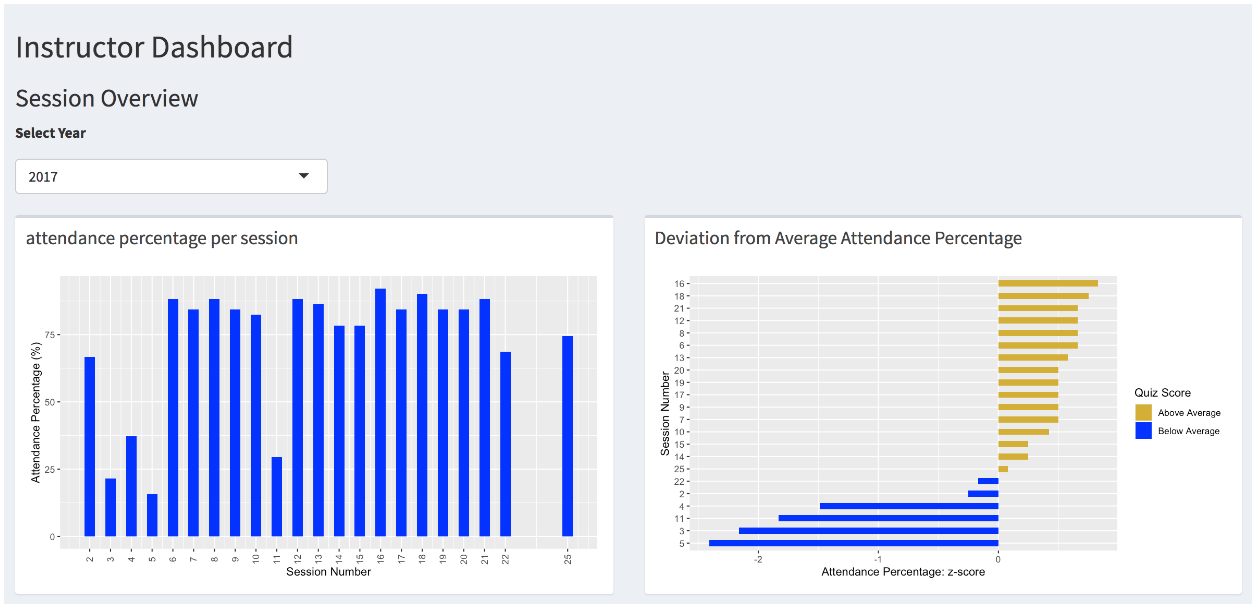
Instructor Dashboard

Session Overview

Select Year

2016


Clicker Answer Rate (%) per session
Clicker Answer Rate (%)
Session Number
Deviation from Average Clicker Answer Rate (%)
Session Number
Clicker Answer Rate (%) : z-score
Clicker Answer Rate (%)
Above Average
Below Average
Clicker Question Accuracy Rate (%) per session
Clicker Question Correct (%)
Session Number
Deviation from Average Clicker Accuracy Rate (%)
Session Number
Clicker Answer Rate (%) : z-score
Accuracy Rate (%)
Above Average
Below Average
Average Temperature Score per session
Average Temperature Score
Session Number
Deviation from Average Temperature Score
Session Number
Average Temperature Score: z-score
Temperature Score
Above Average
Below Average


Instructor Dashboard
Session Overview
Select Year
2017
attendance percentage per session
Attendance Percentage (%)
Session Number
Deviation from Average Attendance Percentage
Session Number
Attendance Percentage: z-score
Quiz Score
Above Average
Below Average

Question 6: Evaluate your group dashboard from the perspective of the instructor (instructor dashboard) or from the perspective of the student (student dashboard). What do you like, what would you change, what else would you like to see? *Each student needs to write their own evaluation here.*

I like how we broke down the class by years and session. This allows the instructor to see changes over time and nail down what sessions seems to cause students more problems. I think we had a problem matching the sessions up year to year, because not all sessions appeared for all years. I think if we had more time, we would have added more interaction.

Submit Project

This is the end of the homework. Please **Knit a PDF report** that shows both the R code and R output and upload it on the EdX platform. Alternatively, you can Knit it as a "doc", open it in Word, and save that as a PDF.

Important: Be sure that all your code is visible. If the line is too long, it gets cut off. If that happens, organize your code on several lines.