Final Project

Estimated time needed: **60** minutes

Objectives

You work for a company who provides career services for those with lower incomes. The marketing team has a small budget and wants your help to find out which demographics (age, education, gender, etc) to target advertise to in the US. You will create an interactive dashboard with Shiny to explore census data to figure out what trends in different demographics you can find and tell the marketing team your findings.

Dataset Used

The dataset you will use is the UCI Adult dataset. The dataset was taken from the Census database and contains demographic information on people. The variables you will focus on are:

- `native_country`: the person's native country
- `age`: age of person
- `hours_per_week`: number of hours the person works per week
- `education`: education level
- `workclass`: class of work like federal government or private etc.
- `sex`: male or female
- `prediction`: the income level, either `"<=50K"` or `">50K"`, you will facet your graphs by this variable

RStudio with Watson Studio

This project will use Watson Studio to run RStudio so that you do not need to install anything locally. Before starting the project, there will be instructions on how to set up your Watson Studio project to run RStudio.

Let's start creating the Shiny dashboard

Goal

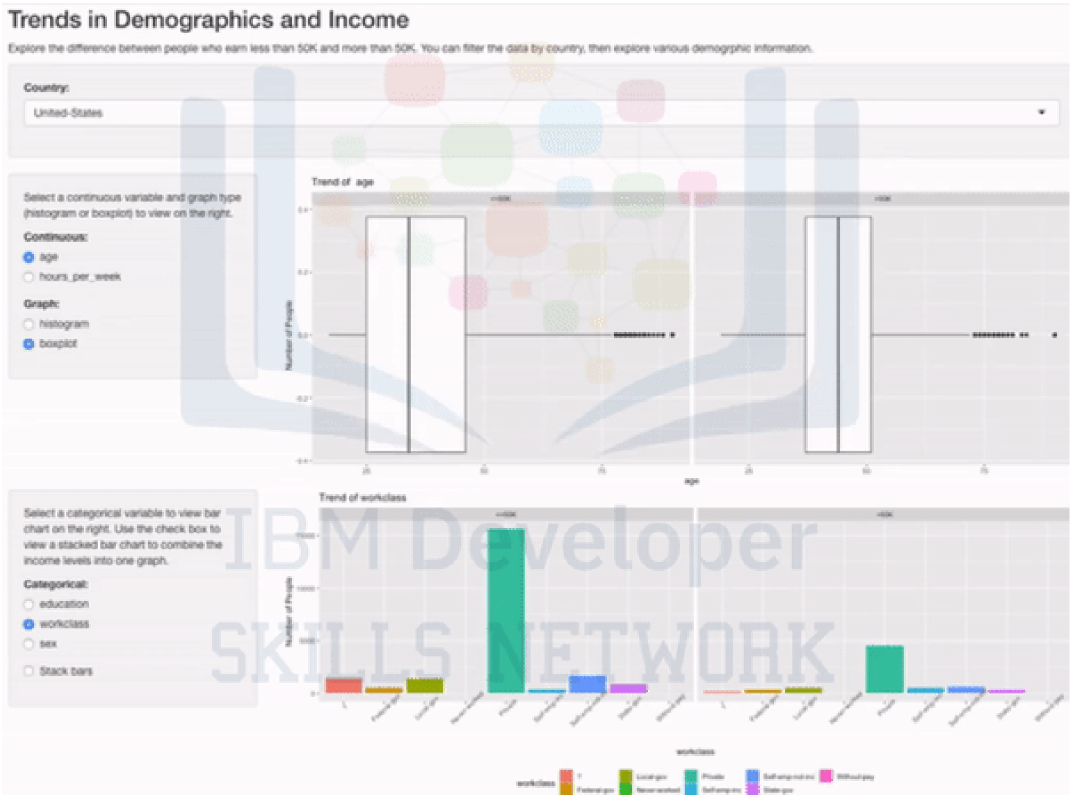
Create a dashboard with the UCI Adult dataset that can give insight on the following questions:

- Does education level impact the salary?
- Impact of age and sex on salary?
- People from which native countries are given less income in spite of having good education?
- Does number of hours working contribute to less income?
- Which workclass faces less income?

Expected Output

Below is the expected result of the project. The dashboard will allow users to select the native country of people to look at. Then users can select which continuous variables to display, age or hours per week. For the continuous variables, the user can choose to display a histogram or boxplot. Both types of graphs are faceted by the income level (`"<=50K"` or `">50K"`)

In the next section, the user can choose which categorical variables to plot with a bar chart. There is also a checkbox to stack the bar charts or facet them by income level (`"<=50K"` or `">50K"`).



Trends in Demographics and Income

Explore the difference between people who earn less than 50K and more than 50K. You can filter the data by country, then explore various demogrphic information.

Country: United-States

Select a continuous variable and graph type (histogram or boxplot) to view on the right.

Continuous: [✓ age] hours_per_week

Graph: histogram [✓ boxplot]

Select a categorical variable to view bar chart on the right. Use the check box to view a stacked bar chart to combine the income levels into one graph.

Categorical: education [✓ workclass] sex

Stack bars

This project includes the following tasks:

1. Add application title in the UI
2. Add first fluidRow to select input for country in UI
3. Add second fluidRow to control how to plot the continuous variables in UI
4. Add third fluidRow to control how to plot the categorical variables in UI
5. Create logic to plot histogram or boxplot in server
6. Create logic to plot faceted bar chart or stacked bar chart in server
7. Optionally change the themes to the graphs

Before starting these tasks, you will first set up Watson Studio to run RStudio. Then, you will download and become familiar with the starter code. The next two sections will guide you with these.

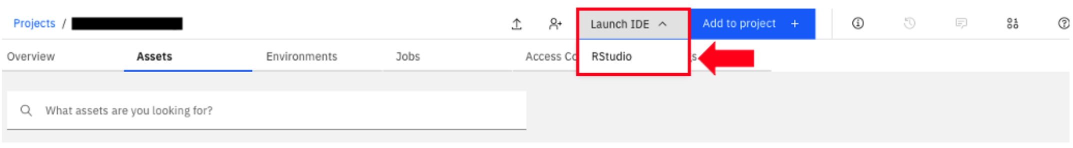
Launch RStudio in IBM Watson Studio

RStudio is the perfect development tool for R Shiny. Thus, in this project, you will be building the R Shiny app using RStudio, hosted by IBM Watson Studio.

1. First, make sure you have IBM Watson Studio setup and create a project, follow this guide for help.

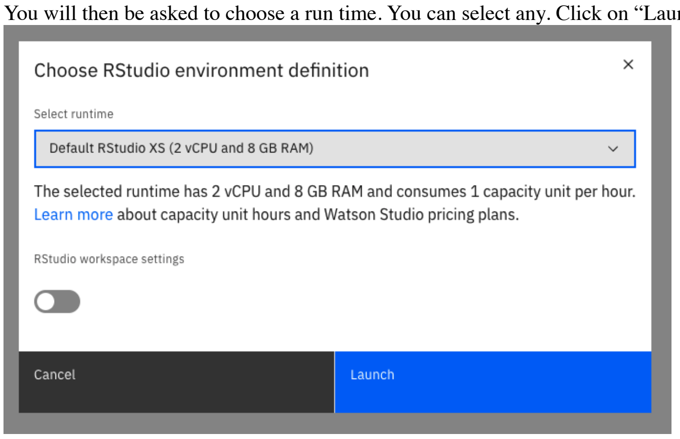
2. In the Watson Studio project, click on "New Data asset" to add the two starter code files (ui.R, server.R) and the data file (adult.csv) to the project.

3. To launch RStudio, click on the `Launch IDE` drop down menu, then click on `RStudio`.



Projects / [redacted]

4. You will then be asked to choose a run time. You can select any. Click on "Launch".



Choose RStudio environment definition

Select runtime

Default RStudio XS (2 vCPU and 8 GB RAM)

The selected runtime has 2 vCPU and 8 GB RAM and consumes 1 capacity unit per hour. Learn more about capacity unit hours and Watson Studio pricing plans.

RStudio workspace settings

You will be directed to the RStudio IDE. Now you are all set up in Watson Studio to get started on the project.

Download and understand the starter code

As you learned, a Shiny app contains two parts: the UI and the server. You are given these two files with some starer code to begin. Throughout the code, you will see comments such as “TASK 1” to help guide you.

Download starter code

In the RStudio console, run the commands to download the two R files and the dataset to start.

```
# UI starter code
url <- "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DV0151EN-SkillsNetwork/labs/module_4/starter_code/ui.
download.file(url, destfile = "ui.R")

# Server starter code
url <- "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DV0151EN-SkillsNetwork/labs/module_4/starter_code/serv
download.file(url, destfile = "server.R")

# Dataset
url <- "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DV0151EN-SkillsNetwork/labs/module_4/starter_code/adul
download.file(url, destfile = "adult.csv")
```

Copied!

Once these files have downloaded, click on these files to open them.

ui.R

You will work on the UI code in the first four tasks, then move on to the server code.
First, it loads the shiny library, if you want to run this locally make sure to install first:

```
install.packages("shiny")
```

Copied!

Then, the UI uses a fluid row layout, which you have learned about but not used in the labs so far. The base code will give a skeleton of the layout. To help you understand it more, see the below section.

Fluid Row Layouts

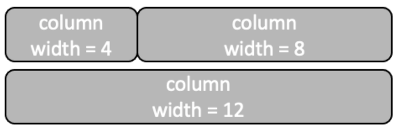
In the past lessons you learned more about using the sidebar layout and the vertical layout. In this project, you will use fluid rows which allows for more customization.

You can think of each fluidRow as having 12 columns, so all the widths should add up to 12. For example:

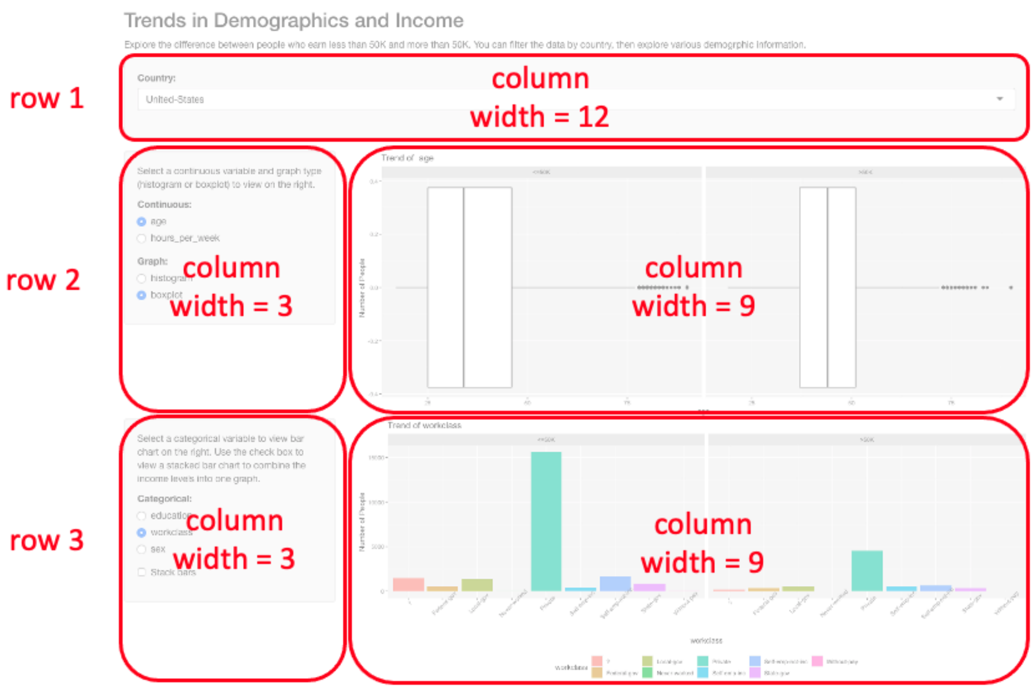
```
ui <- fluidPage(
 fluidRow(
     column(width = 4),
     column(width = 8)
 ),
 fluidRow(column(width = 12))
)
```

Copied!

Which would look like:



Now, for this application, we want the layout to look like:


Trends in Demographics and Income

Explore the difference between people who earn less than 50K and more than 50K. You can filter the data by country, then explore various demogrphic information.

Select a continuous variable and graph type (histogram or boxplot) to view on the right.

Select a categorical variable to view bar chart on the right. Use the check box to view a stacked bar chart to combine the income levels into one graph.

The first four tasks will walk you through the title panel and then each **row** of the layout.

Additionally in the starter code, you will see `wellPanel()` used. Well panels simply just hold other objects, like a group of input widgets. They create a panel with a border and gray background. For example, you could put a selectInput and radioButtons inside a well panel like so:

```
fluidRow(
    column(width = 4,
      wellPanel(
        selectInput(...),
        radioButtons(...))
      ),
    column(width = 8, plotOutput('p1'))
)
```

Copied!

server.R

After completing the UI, you will work on the server code.
Inside the `shinyServer()`, there is first this given code

```
df_country <- reactive({
  adult %>% filter(native_country == input$country)
})
```

Copied!

This uses the input value `country` to filter the dataset. The new data is stored as `df_country` and can be accessed using `df_country()`.

There are two parts for you to complete

- The histogram and boxplot section. Based on the input graph type, you will create a histogram or boxplot, both will be faceted by the `prediction` (which is either `"<=50K"` or `">50K"`).
- The bar chart. Based on the check box `is_stacked`, you will create a stacked bar chart or a bar chart faceted by `prediction`.

The `...` will be where you modify the code.

Now that you understand the UI and server starter code, let's get started.

TASK 1 - Add application title in the UI

In the ui.R file, add a title to the application with `titlePanel()`.

TASK 2 - Add first fluidRow to select input for country in UI

In the first fluidRow, add a `selectInput()` in the `wellPanel()`. This select input will be used to filter the data by the person's native country. Use the ID `country`, give it any label you think fits, and as `choices` use the five countries: `"United-States", "Canada", "Mexico", "Germany", "Philippines"`

TASK 3 - Add second fluidRow to control how to plot the continuous variables in UI

In the second fluidRow, add two `radioButtons` and the output plot.

- The first radio buttons are to choose `"continuous_variable"` (input ID), the choices are `"age"` and `"hours_per_week"`. Add a label.
- The second radio buttons are to choose the "`graph_type`" (input ID), the choices are `"histogram"` and `"boxplot"`. Add a label.
- Add a plot output, `"p1"`, with `plotOutput()`

Hint: You can always find more information about functions by going in the RStudio console and using for example

```
?radioButtons
```

Copied!

TASK 4 - Add third fluidRow to control how to plot the categorical variables in UI

In the last fluidRow, you will add `radioButtons()`, `checkboxInput()`, and the output plot.

- Add the radio buttons to select the `"categorical_variable"` (input ID) with choices `"education"`, "`workclass`", and "`sex`". Add a label as well.
- Add a checkbox to check if the bars `"is_stacked"` (input ID), that is, if the checkbox is checked then the bars will be stacked. Otherwise, they will be faceted and unstacked. Add a label. The initial `value` can be either `FALSE` or `TRUE`.
- Add a plot output, `"p2"`, with `plotOutput()`

Test UI code

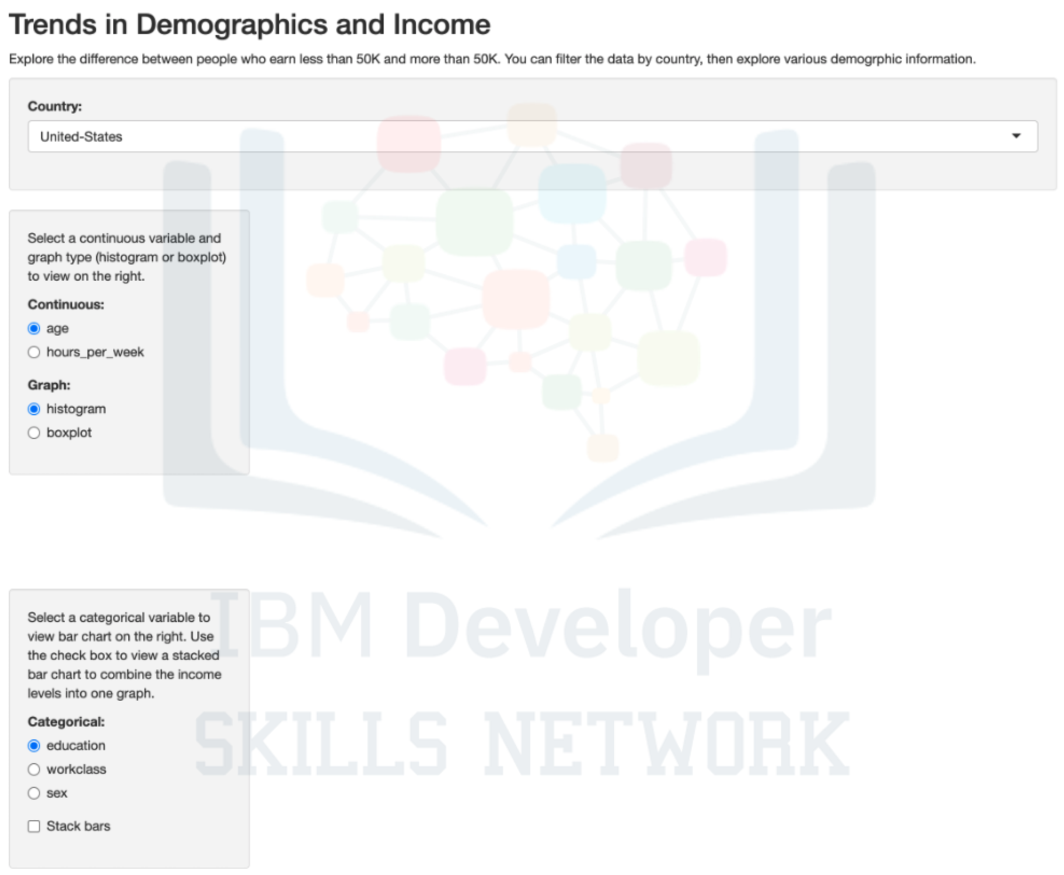
You have completed the UI code. If you want to test that it is working, in the server.R file you can comment out everything in `shinyServer()` so that the server side functions doesn't break. Or you could clear everything in `shinyServer()` like:

```
shinyServer(function(input, output) {

})
```

Copied!

Then if you click `Run App`, you should see all the components and input widgets like the image below


Trends in Demographics and Income

Explore the difference between people who earn less than 50K and more than 50K. You can filter the data by country, then explore various demogrphic information.

Country: United-States

Select a continuous variable and graph type (histogram or boxplot) to view on the right.

Continuous: ● age ○ hours_per_week

Graph: ● histogram ○ boxplot

Select a categorical variable to view bar chart on the right. Use the check box to view a stacked bar chart to combine the income levels into one graph.

Categorical: ● education ○ workclass ○ sex

☐ Stack bars

You will notice that the graphs are not there, this logic will be handled in the server code.

TASK 5 - Create logic to plot histogram or boxplot in server

You can now go to the server.R file. Follow the comments and fill in the `....` The input variable you want to use for the histogram and boxplot is the continuous variable.

The first part will be to complete the histogram and boxplot logic. There is a conditional statement, so if the graph type the user input with the radio button is “histogram”, then you will plot a histogram. You will

- add the continuous variable to use for the x-axis,
- add the labels for the y axis and title, and
- facet the graph by `prediction`

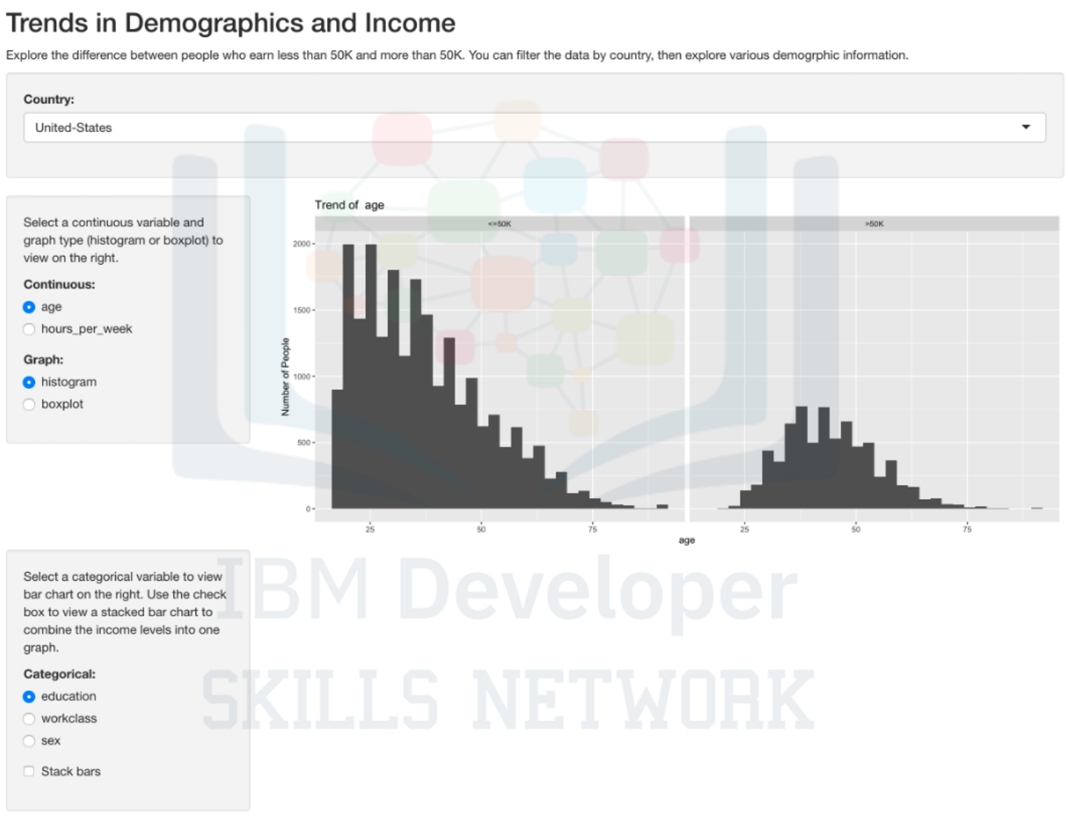
Otherwise, create a boxplot. You will:

- add the continuous variable to use for the y-axis,
- flip the coordinates
- add the labels for the y axis and title, and
- facet the graph by `prediction`

Hints:

- You can use something like `input$continuous_variable` to get the input IDs you set in the UI.
- In ggplot, you used `aes()` before. There is another similar function called `aes_string()`. The difference is that it takes as inputs strings. For example: `ggplot(mtcars, aes(mpg))` is equivalent to `ggplot(mtcars, aes_string("mpg"))`. When you call `input$continuous_variable`, it will give you back a string so you can use `aes_string(x = input$continuous_variable)`.
- Facet your plots using `facet_warp(~variable)`
- You can use `paste()` to paste together strings

To test your work, you can comment out all of the “TASK 6” code and select `Run App`. Try changing the country and try the different options for radio buttons “Continuous” and “Graphs”. Your dashboard should look similar to:


Trends in Demographics and Income

Explore the difference between people who earn less than 50K and more than 50K. You can filter the data by country, then explore various demogrphic information.

Country: United-States

Select a continuous variable and graph type (histogram or boxplot) to view on the right.

Continuous: [✓ age] hours_per_week

Graph: [✓ histogram] boxplot

Select a categorical variable to view bar chart on the right. Use the check box to view a stacked bar chart to combine the income levels into one graph.

Categorical: [✓ education] workclass sex

☐ Stack bars

TASK 6 - Create logic to plot faceted bar chart or stacked bar chart in server

The final section of the dashboard is the bar chart. Follow the comments and fill in the `....` The input variable you want to use for the bar chart is the categorical variable.

The first part is a base ggplot object `p`. You will have to:

- add the categorical variable to use for the x-axis,
- add the labels for the y axis and title, and
- change the theme so that
  - the x axis text labels are at a 45 degree angle
  - the legend position is the bottom

The geometry object (bar) will be different based on a condition. If the check box is checked, then you will:

- create a stacked bar chart that uses `prediction` as the fill

Otherwise, you will:

- create a bar chart (not stacked) faceted by `prediction`, the fill should be the input categorical variable

Hints:

- You can use something like `input$categorical_variable` to get the input IDs you set in the UI
- You can use `aes_string()` in `geom_bar` to change the `fill`
- To change the x-axis text angle, use parameter `axis.text.x` in `theme()`
  - since you are changing the text, use `element_text()` and modify the `angle` parameter
- To change the legend position, use parameter `legend.position` in `theme()`
- Facet your plots using `facet_warp(~variable)`
- You can use `paste()` to paste together strings

TASK 7 - Optionally change the themes to the graphs

Feel free to add any more customizations to the graphs. You can add a different theme or color palettes to any of the plots. If you want to use `ggthemes` in RStudio in Watson Studio, install the package first:

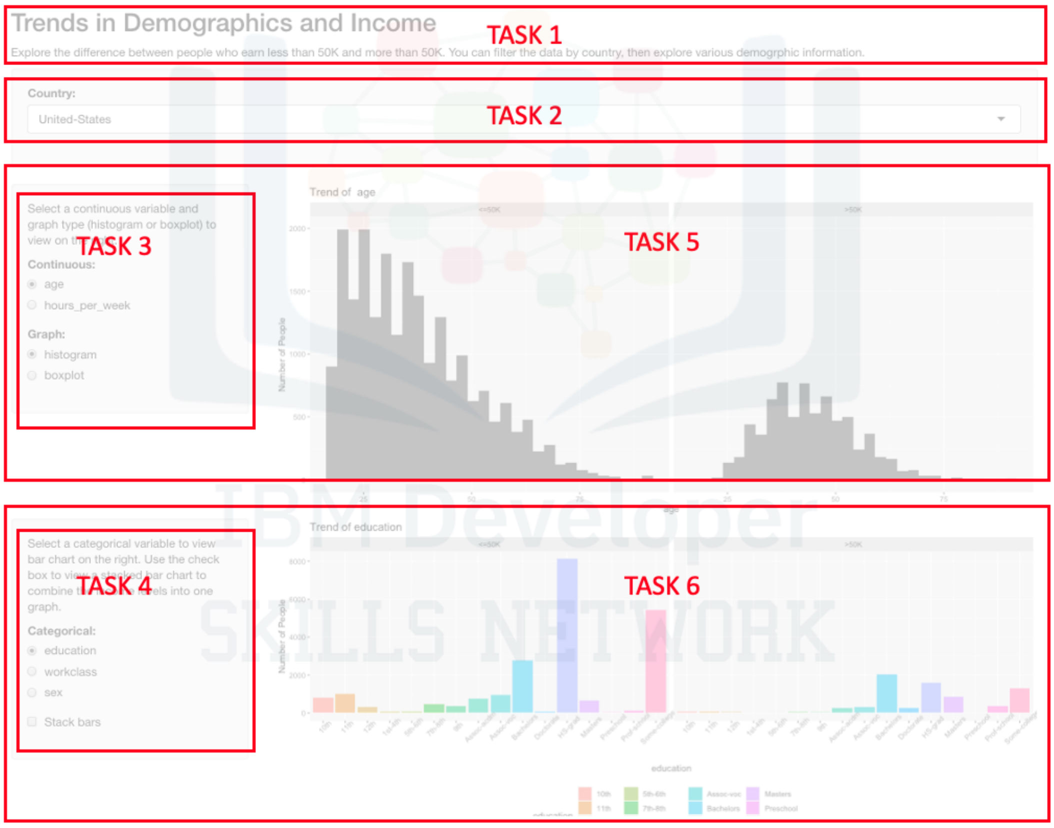
```
install.packages("ggthemes")
library(ggthemes)
```

Copied!

Submitting

Congratulations, you have completed the Shiny dashboard! To test that it's working, click `Run App`. It may take a few seconds for the graphs to load initially. You can play around with the dashboard and see what kinds of insights you can find.

To submit your project, you will take screen shots of components of your dashboard and a peer will review your work. Below is an image that breaks down what area of the dashboard you should submit for each task for the peer review.


Trends in Demographics and Income

Explore the difference between people who earn less than 50K and more than 50K. You can filter the data by country, then explore various demogrphic information.

Country: United-States

Select a continuous variable and graph type (histogram or boxplot) to view on the right.

Continuous: ● age ○ hours_per_week

Graph: ● histogram ○ boxplot

Select a categorical variable to view bar chart on the right. Use the check box to view a stacked bar chart to combine the levels into one graph.

Categorical: ● education ○ workclass ○ sex

☐ Stack bars

Author(s)

[Tiffany Zhu]

[Saishruthi Swaminathan]

Changelog

Date	Version	Changed by	Change Description
2021-05-11	1.0	Tiffany Zhu	Created the initial version