STARTING APP

# ---------------------------------------------------------------------------- #

# ACTIVITY 1.4f TIMESERIES\_3 SERVER

# ---------------------------------------------------------------------------- #

# Load in libraries

library(shiny)

library(dplyr)

library(ggplot2)

library(RColorBrewer)

library(lubridate)

# Load in the raw data

raw\_data <- read.csv("data/raw\_data.csv", stringsAsFactors=FALSE)

# Create a colour palette

col\_palette <- brewer.pal(name="Dark2", n=8)

# Collect list of years

raw\_data$date <- as.Date(raw\_data$date) # Change the structure of 'date' to a date

yrs <- c(unique(year(raw\_data$date)), 2015) #Extract year and take only the unique years

# Set plot breaks

plot\_breaks = seq(from=0, to=12\*length(yrs)-1, by=12)

# Summarise for 'all' data - this will always be plotted

overall\_summary <- raw\_data %>%

group\_by(month) %>%

summarise(n = length(month)) %>%

mutate(region="All data",

species="All data")

# Summarise for 'all' sexes, divided by region data

region\_allspecies\_summary <- raw\_data %>%

group\_by(month, region, age) %>%

summarise(n = length(month)) %>%

mutate(species="All species")

# Summarise for 'all' region, divided by sex data

species\_allregions\_summary <- raw\_data %>%

group\_by(month, species, age) %>%

summarise(n = length(month)) %>%

mutate(region="All Regions")

# Summarise region and sex data

region\_species\_summary <- raw\_data %>%

group\_by(month, region, species, age) %>%

summarise(n = length(month))

# Join summary data together

summary\_data <- bind\_rows(overall\_summary, region\_allspecies\_summary, species\_allregions\_summary, region\_species\_summary)

#------------------------------------------------------------------------------#

# Begin server section

shinyServer(function(input, output) {

# Subset for the chosen region and sexes

data\_subset <- reactive({

# Subset for region

data\_sub = summary\_data %>%

filter(region==input$select\_region | region=="All data")

# Subset for sex

if(length(input$select\_species)>0){

data\_sub = data\_sub %>%

filter(species %in% input$select\_species | species=="All species" | species=="All data")

} else {

data\_sub = data\_sub %>%

filter(species=="All species" | species=="All data")

}

# Subset for age

data\_sub = data\_sub %>%

filter(age <= input$age\_slider | is.na(age))

# Summarise data using subsets created above

data\_sub = data\_sub %>%

group\_by(month, region, species) %>%

summarise(n = sum(n))

as.data.frame(data\_sub)

})

# Produce plot

output$tsPlot <- renderPlot({

ggplot() +

geom\_path(data=data\_subset(), aes(x=month, y=n, color=species), size=1) +

scale\_color\_manual(name="Species", values=col\_palette) +

labs(title=input$select\_region, x="Date (Month)", y="Number of records") +

# Extra plotting code to control appearence

scale\_x\_continuous(breaks=plot\_breaks, labels=yrs, limits=c(min(overall\_summary$month), max(overall\_summary$month))) +

theme\_classic() +

theme(axis.text = element\_text(size=14),

axis.title = element\_text(size=18),

plot.title = element\_text(size=20),

legend.title = element\_text(size=18),

legend.text = element\_text(size=14))

})

})

STARTING APP

# ---------------------------------------------------------------------------- #

# ACTIVITY 1.4f TIMESERIES\_3 UI

# ---------------------------------------------------------------------------- #

# Load in libraries

library(shiny)

library(dplyr)

library(ggplot2)

# Load in the raw data

raw\_data <- read.csv("data/raw\_data.csv", stringsAsFactors=FALSE)

# Collect a list of regions for the dropdown menu

options\_list <- c("All Regions", sort(unique(raw\_data$region)))

# Collect min and max ages for the slider

min\_age <- min(raw\_data$age)

max\_age <- max(raw\_data$age)

#------------------------------------------------------------------------------#

# Begin ui section

shinyUI(fluidPage(

# Application title

titlePanel("Day 1 - Timeseries\_3"),

# Add a line break

br(),

# Add text section

h4("This app is identical to the last, with a new widget: sliderInput"),

h4("Using these widgets together, we can change the region, the sex and the maximum age we want to view on the plot. The line showing 'all data' will always be visible!"),

# Add a line break

br(),

# Sidebar with a slider input for number of bins

sidebarLayout(

sidebarPanel(

# Add a dropdown menu widget

selectInput("select\_region", label = h3("Select a Region:"),

choices = options\_list,

selected = 1),

br(),

# Add a checkbox widget

checkboxGroupInput("select\_species", label = h3("Select a Species"),

choices = list("Cat" = "cat", "Dog" = "dog", "Human"="human", "Jackal"="jackal", "Lion"="lion"),

selected = c("cat", "dog", "human", "jackal", "lion")),

br(),

# Add a slider

sliderInput("age\_slider", label = h3("Select a maximum age"),

min = min\_age, max = max\_age, value = max\_age)

),

# Show a plot of the generated distribution

mainPanel(

plotOutput("tsPlot", height=700)

)

)

))

COMPLETED APP

# ---------------------------------------------------------------------------- #

# ACTIVITY 1.4f TIMESERIES\_3 SERVER

# ---------------------------------------------------------------------------- #

# Load in libraries

library(shiny)

library(dplyr)

library(ggplot2)

library(RColorBrewer)

library(lubridate)

# Load in the raw data

raw\_data <- read.csv("data/raw\_data.csv", stringsAsFactors=FALSE)

# Create a colour palette

col\_palette <- brewer.pal(name="Dark2", n=8)

# Collect list of years

raw\_data$date <- as.Date(raw\_data$date) # Change the structure of 'date' to a date

yrs <- c(unique(year(raw\_data$date)), 2015) #Extract year and take only the unique years

# Set plot breaks

plot\_breaks = seq(from=0, to=12\*length(yrs)-1, by=12)

# Summarise for 'all' data - this will always be plotted

overall\_summary <- raw\_data %>%

group\_by(month) %>%

summarise(n = length(month)) %>%

mutate(region="All data",

species="All data")

# Summarise for 'all' sexes, divided by region data

region\_allspecies\_summary <- raw\_data %>%

group\_by(month, region, age) %>%

summarise(n = length(month)) %>%

mutate(species="All species")

# Summarise for 'all' region, divided by sex data

species\_allregions\_summary <- raw\_data %>%

group\_by(month, species, age) %>%

summarise(n = length(month)) %>%

mutate(region="All Regions")

# Summarise region and sex data

region\_species\_summary <- raw\_data %>%

group\_by(month, region, species, age) %>%

summarise(n = length(month))

# Join summary data together

summary\_data <- bind\_rows(overall\_summary, region\_allspecies\_summary, species\_allregions\_summary, region\_species\_summary)

#------------------------------------------------------------------------------#

# Begin server section

shinyServer(function(input, output) {

# Subset for the chosen region and sexes

data\_subset <- reactive({

# Subset for region

data\_sub = summary\_data %>%

filter(region==input$select\_region | region=="All data")

# Subset for sex

if(length(input$select\_species)>0){

data\_sub = data\_sub %>%

filter(species %in% input$select\_species | species=="All species" | species=="All data")

} else {

data\_sub = data\_sub %>%

filter(species=="All species" | species=="All data")

}

# Subset for age

data\_sub = data\_sub %>%

filter(age >= input$age\_slider[1] & age <= input$age\_slider[2] | is.na(age))

# Summarise data using subsets created above

data\_sub = data\_sub %>%

group\_by(month, region, species) %>%

summarise(n = sum(n))

as.data.frame(data\_sub)

})

# Produce plot

output$tsPlot <- renderPlot({

ggplot() +

geom\_path(data=data\_subset(), aes(x=month, y=n, color=species), size=1) +

scale\_color\_manual(name="Species", values=col\_palette) +

labs(title=input$select\_region, x="Date (Month)", y="Number of records") +

# Extra plotting code to control appearence

scale\_x\_continuous(breaks=plot\_breaks, labels=yrs, limits=c(min(overall\_summary$month), max(overall\_summary$month))) +

theme\_classic() +

theme(axis.text = element\_text(size=14),

axis.title = element\_text(size=18),

plot.title = element\_text(size=20),

legend.title = element\_text(size=18),

legend.text = element\_text(size=14))

})

})

COMPLETED APP

# ---------------------------------------------------------------------------- #

# ACTIVITY 1.4f TIMESERIES\_3 UI

# ---------------------------------------------------------------------------- #

# Load in libraries

library(shiny)

library(dplyr)

library(ggplot2)

# Load in the raw data

raw\_data <- read.csv("data/raw\_data.csv", stringsAsFactors=FALSE)

# Collect a list of regions for the dropdown menu

options\_list <- c("All Regions", sort(unique(raw\_data$region)))

# Collect min and max ages for the slider

min\_age <- min(raw\_data$age)

max\_age <- max(raw\_data$age)

#------------------------------------------------------------------------------#

# Begin ui section

shinyUI(fluidPage(

# Application title

titlePanel("Exploratory plots: Timeseries\_3 (Master)"),

# Add a line break

br(),

# Add text section

h4("This app is identical to the last, with a new widget: checkboxGroupInput"),

h4("Using these widgets, we can change the region and the sex we want to view on the plot. The line showing 'all data' will always be visible!"),

# Add a line break

br(),

# Sidebar with a slider input for number of bins

sidebarLayout(

sidebarPanel(

# Add a dropdown menu widget

selectInput("select\_region", label = h3("Select a Region:"),

choices = options\_list,

selected = 1),

br(),

# Add a checkbox widget

checkboxGroupInput("select\_species", label = h3("Select a Species"),

choices = list("Cat" = "cat", "Dog" = "dog", "Human"="human", "Jackal"="jackal", "Lion"="lion"),

selected = c("cat", "dog", "human", "jackal", "lion")),

br(),

# Add a slider

sliderInput("age\_slider", label = h3("Select a maximum age"),

min = min\_age, max = max\_age, value = c(min\_age, max\_age))

),

# Show a plot of the generated distribution

mainPanel(

plotOutput("tsPlot", height=700)

)

)

))