Project Example

Final Project

Data

The data that I am using for my project is from the World Happiness Report.

I obtained the data from Kaggle.com at this link: https://www.kaggle.com/ajaypalsinghlo/world-happiness-report-2021/download (https://www.kaggle.com/ajaypalsinghlo/world-happiness-report-2021/download).

The data comes from an annual report from the Gallup World Poll. Here is a link for more information about this poll: https://www.gallup.com/analytics/247355/gallup-world-happiness-report.aspx (https://www.gallup.com/analytics/247355/gallup-world-happiness-report.aspx). This report has gained global recognition and is used to inform policy making decisions.

The variables in the data include score estimates for various factors that have been shown to be related to happiness. The estimate for happiness variable is called “Ladder Score”. There is data related to a fake country called “Dystopia” which simulates a country that has the worlds lowest values for all the factors related to happiness. The factors related to happiness are sometimes used to explain why some countries report higher levels of happiness.

I attempted to find more about how people were surveyed for the poll, but did not find much information. The poll however occurs annually through a survey of adults around the world.

Now I will load packages that I may need in my analysis:

```
library(tidyverse)
```

Data Import

I will then import my data by using the `read_csv()` function of the readr package as the data is in csv format. I will name the data Happiness.

```
Happiness <- read_csv("world-happiness-report-2021.csv")
```

```
## Rows: 149 Columns: 20
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr  (2): Country name, Regional indicator
## dbl (18): Ladder score, Standard error of ladder score, upperwhisker, lowerw...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

I then take a look at my data using the `head()` function. I use the `dim()` function to see how many rows/columns there are.

```
head(Happiness)
```

```
## # A tibble: 6 × 20
##   `Country name` `Regional indicator` `Ladder score` Standard error of ladder …¹
##   <chr>          <chr>                         <dbl>                       <dbl>
## 1 Finland        Western Europe                 7.84                       0.032
## 2 Denmark        Western Europe                 7.62                       0.035
## 3 Switzerland    Western Europe                 7.57                       0.036
## 4 Iceland        Western Europe                 7.55                       0.059
## 5 Netherlands    Western Europe                 7.46                       0.027
## 6 Norway         Western Europe                 7.39                       0.035
## # ℹ abbreviated name: ¹`Standard error of ladder score`
## # ℹ 16 more variables: upperwhisker <dbl>, lowerwhisker <dbl>,
## #   `Logged GDP per capita` <dbl>, `Social support` <dbl>,
## #   `Healthy life expectancy` <dbl>, `Freedom to make life choices` <dbl>,
## #   Generosity <dbl>, `Perceptions of corruption` <dbl>,
## #   `Ladder score in Dystopia` <dbl>, `Explained by: Log GDP per capita` <dbl>,
## #   `Explained by: Social support` <dbl>, …
```

```
dim(Happiness)
```

```
## [1] 149  20
```

To determine how many unique countries there are I used the `unqiue()` function and the `length()` function.

```
pull(Happiness, "Country name") %>%
  unique() %>%
  length()
```

```
## [1] 149
```

This shows that there are 149 Countries.

Data Cleaning/Wrangling

I decided to rename the first variable called `Country name` so that it would not have a space - this makes it easier to work with as I will not need to use quotes around it for most functions. I used the `rename()` function to do this, listing the new name first.

```
Happiness <- Happiness %>%
  rename("Country_name" = "Country name")
colnames(Happiness)
```

```
##  [1] "Country_name"
##  [2] "Regional indicator"
##  [3] "Ladder score"
##  [4] "Standard error of ladder score"
##  [5] "upperwhisker"
##  [6] "lowerwhisker"
##  [7] "Logged GDP per capita"
##  [8] "Social support"
##  [9] "Healthy life expectancy"
## [10] "Freedom to make life choices"
## [11] "Generosity"
## [12] "Perceptions of corruption"
## [13] "Ladder score in Dystopia"
## [14] "Explained by: Log GDP per capita"
## [15] "Explained by: Social support"
## [16] "Explained by: Healthy life expectancy"
## [17] "Explained by: Freedom to make life choices"
## [18] "Explained by: Generosity"
## [19] "Explained by: Perceptions of corruption"
## [20] "Dystopia + residual"
```

I can see that this column is renamed, but I then realized that many columns have names with spaces. I decided to do the same thing for the `"Ladder score"` variable. I renamed it `"Happiness_score"`, as it is easier to tell what this means.

```
Happiness <- Happiness %>%
  rename("Happiness_score" = "Ladder score")
colnames(Happiness)
```

```
##  [1] "Country_name"
##  [2] "Regional indicator"
##  [3] "Happiness_score"
##  [4] "Standard error of ladder score"
##  [5] "upperwhisker"
##  [6] "lowerwhisker"
##  [7] "Logged GDP per capita"
##  [8] "Social support"
##  [9] "Healthy life expectancy"
## [10] "Freedom to make life choices"
## [11] "Generosity"
## [12] "Perceptions of corruption"
## [13] "Ladder score in Dystopia"
## [14] "Explained by: Log GDP per capita"
## [15] "Explained by: Social support"
## [16] "Explained by: Healthy life expectancy"
## [17] "Explained by: Freedom to make life choices"
## [18] "Explained by: Generosity"
## [19] "Explained by: Perceptions of corruption"
## [20] "Dystopia + residual"
```

I then decided to arrange the data by `Happiness_score` using the `arrange()` function with the largest score shown first, thus I needed to use `desc()`.

```
Happiness <- Happiness %>% arrange(desc(Happiness_score))
head(Happiness)
```

```
## # A tibble: 6 × 20
##   Country_name `Regional indicator` Happiness_score Standard error of ladder s…¹
##   <chr>        <chr>                          <dbl>                        <dbl>
## 1 Finland      Western Europe                  7.84                        0.032
## 2 Denmark      Western Europe                  7.62                        0.035
## 3 Switzerland  Western Europe                  7.57                        0.036
## 4 Iceland      Western Europe                  7.55                        0.059
## 5 Netherlands  Western Europe                  7.46                        0.027
## 6 Norway       Western Europe                  7.39                        0.035
## # ℹ  abbreviated name: ¹`Standard error of ladder score`
## # ℹ  16 more variables: upperwhisker <dbl>, lowerwhisker <dbl>,
## #   `Logged GDP per capita` <dbl>, `Social support` <dbl>,
## #   `Healthy life expectancy` <dbl>, `Freedom to make life choices` <dbl>,
## #   Generosity <dbl>, `Perceptions of corruption` <dbl>,
## #   `Ladder score in Dystopia` <dbl>, `Explained by: Log GDP per capita` <dbl>,
## #   `Explained by: Social support` <dbl>, …
```

I could see from this that Finland is the top scoring country and that all of the top 6 countries are in Western Europe.

I then wanted to know about the US so I filtered for `United States` for the `Country_name`.

```
Happiness %>% filter(Country_name == "United States")
```

```
## # A tibble: 1 × 20
##   Country_name  `Regional indicator`  Happiness_score Standard error of ladder…¹
##   <chr>         <chr>                           <dbl>                      <dbl>
## 1 United States North America and ANZ            6.95                      0.049
## # ℹ  abbreviated name: ¹`Standard error of ladder score`
## # ℹ  16 more variables: upperwhisker <dbl>, lowerwhisker <dbl>,
## #   `Logged GDP per capita` <dbl>, `Social support` <dbl>,
## #   `Healthy life expectancy` <dbl>, `Freedom to make life choices` <dbl>,
## #   Generosity <dbl>, `Perceptions of corruption` <dbl>,
## #   `Ladder score in Dystopia` <dbl>, `Explained by: Log GDP per capita` <dbl>,
## #   `Explained by: Social support` <dbl>, …
```

Looks like the score was fairly high for the United States - much higher than Dystopia.

I wondered how the US ranked, so I decided to make a variable for rank. To do so I used the `seq()` function to create a sequence of numbers from 1 to 149 the number of countries or rows in the data.

```
Happiness <- Happiness %>% mutate(rank = seq(1:149))
Happiness$rank
```

```
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##  [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##  [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##  [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##  [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##  [91]  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
## [109] 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
## [127] 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
## [145] 145 146 147 148 149
```

I then looked again at the US using `filter()`. I used the `pull()` function to get the rank for the US.

```
Happiness %>%
  filter(Country_name == "United States") %>%
  pull(rank)
```

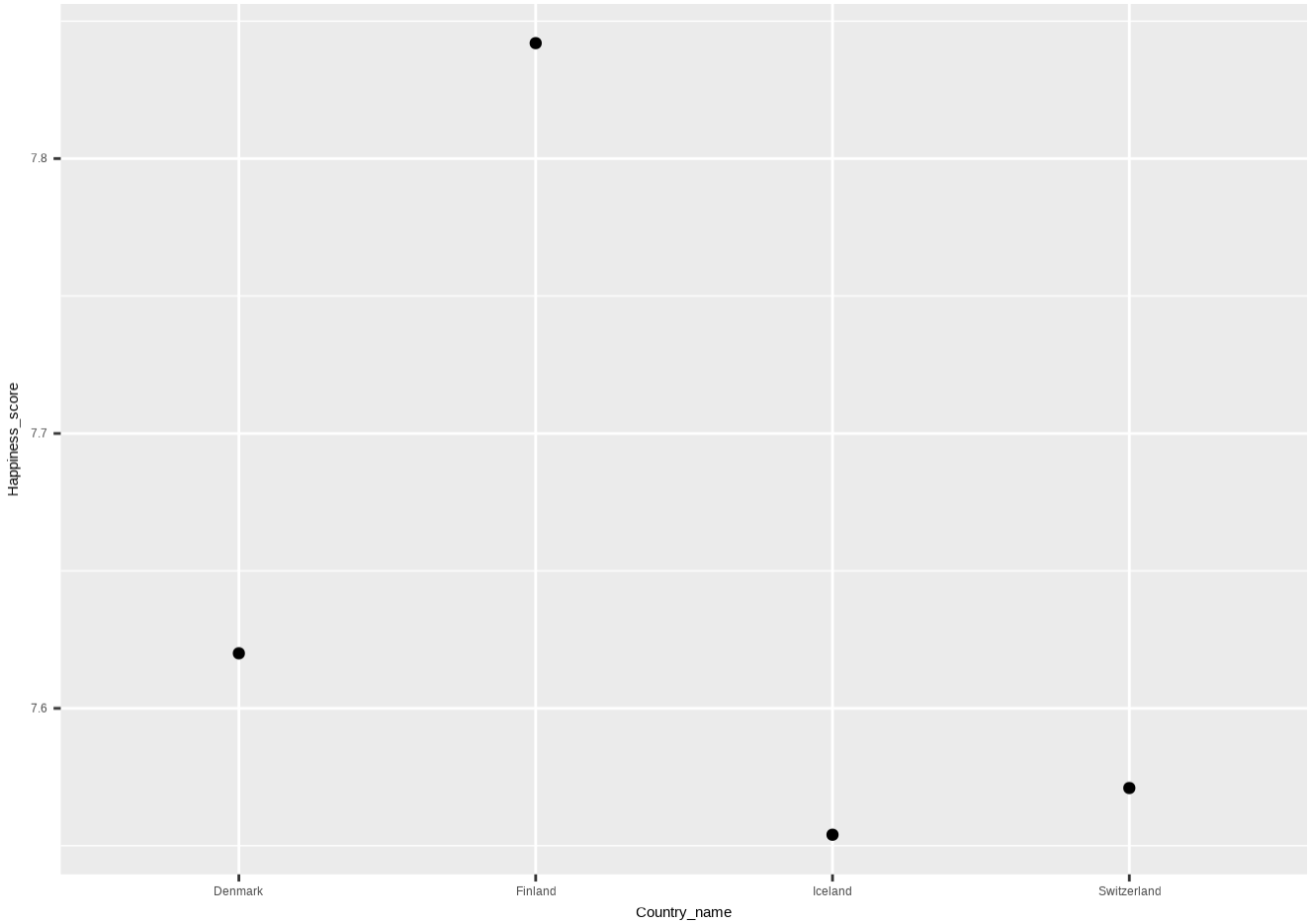
```
## [1] 19
```

The US ranked 19 in 2021 for happiness out of the 149 countries.

Data Visualization

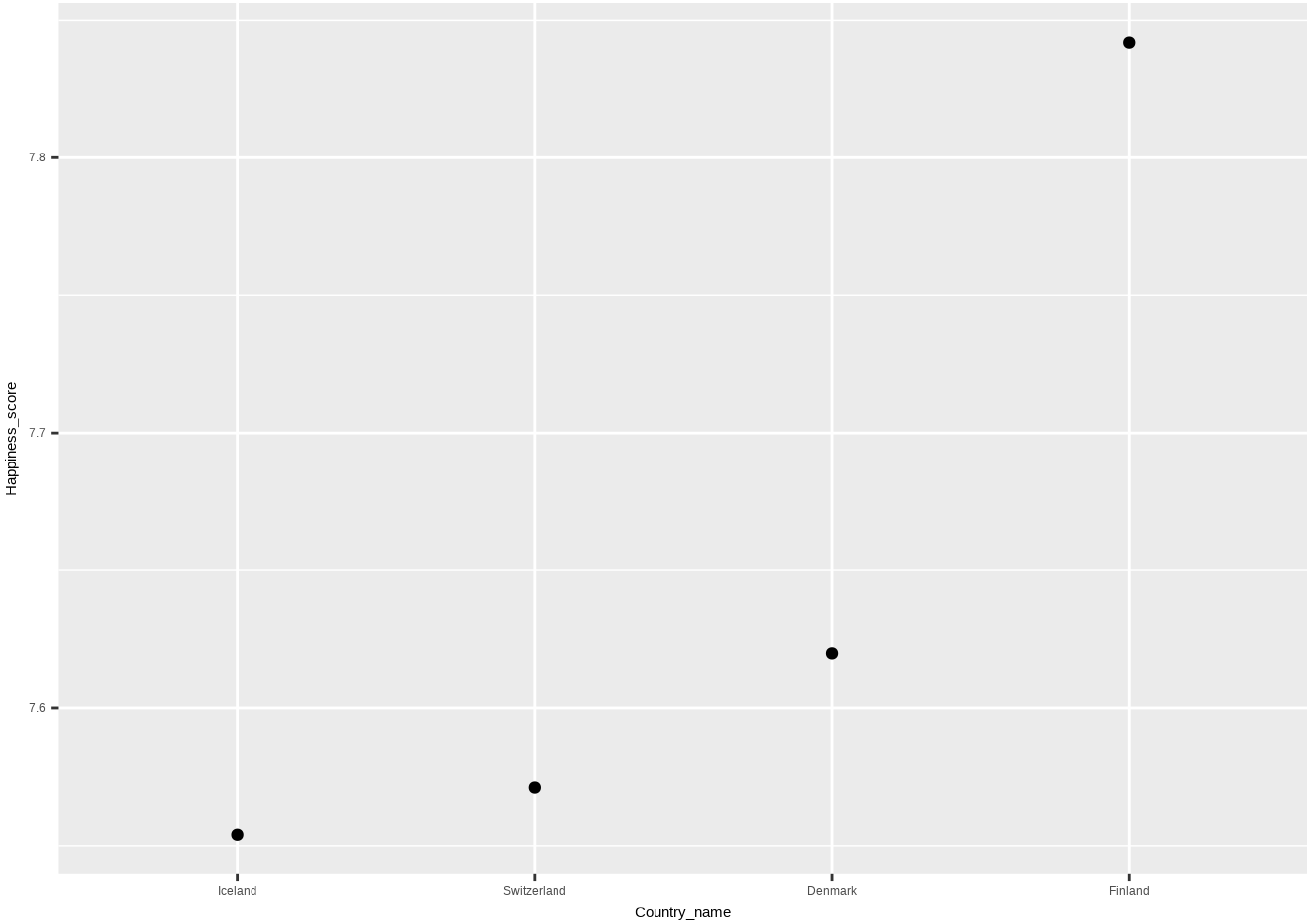
I decided to make a plot of the top 5 countries using `geom_point()` and `ggplot()`. First I created a new dataset of just the top 5 ranking countries using the `filter()` function to filter for countries with a `rank` of less than 5.

```
top5 <- Happiness %>% filter(rank < 5)

ggplot(
  data = top5,
  aes(
    x = Country_name,
    y = Happiness_score
  )
) +
  geom_point()
```



I felt that this plot could be improved. So first I used I made `Country_name` a factor and then reordered it by the `Happiness_score` variable.

```
top5 <- top5 %>%
  mutate(
    Country_name = as_factor(Country_name),
    Country_name = fct_reorder(Country_name, Happiness_score)
  )

ggplot(
  data = top5,
  aes(
    x = Country_name,
    y = Happiness_score
  )
) +
  geom_point()
```

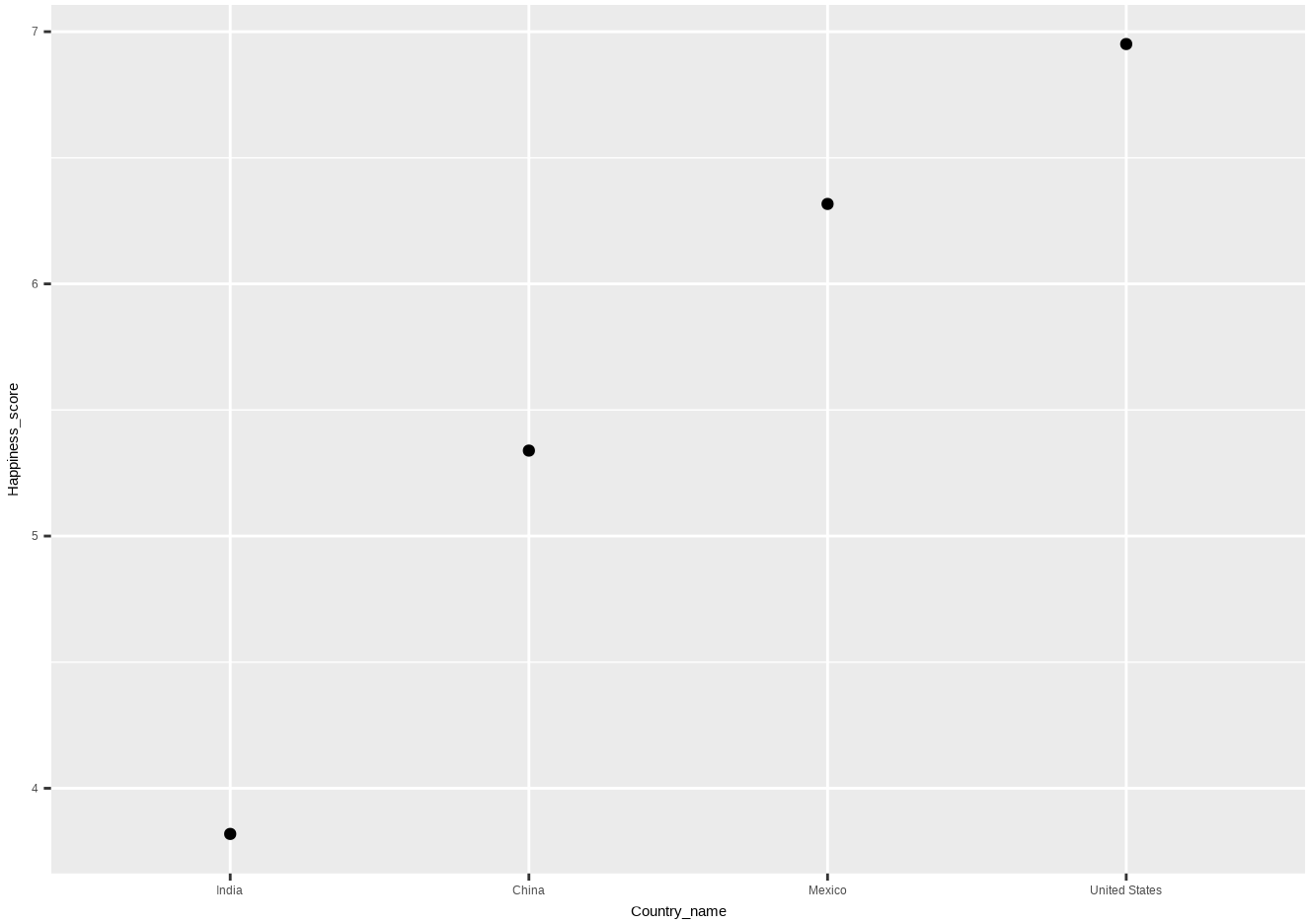


I then decided to make a plot of a few countries of interest. Since I again wanted to have countries ordered by the `Happiness_score`, I figured I might look at additional countries, so I decided to use the full dataset `Happiness`.

```
Happiness <- Happiness %>%
  mutate(
    Country_name = as_factor(Country_name),
    Country_name = fct_reorder(Country_name, Happiness_score)
  )
```

Now I check out these other countries that I was interested in by filtering for them from the `Happiness` data set and creating another plot.

```
int_Country <- Happiness %>% filter(Country_name %in% c("China", "United States", "India", "Mexico"))

ggplot(
  data = int_Country,
  aes(
    x = Country_name,
    y = Happiness_score
  ),
  group = Country_name
) +
  geom_point()
```



I can see from this plot that the United States ranked highest. Mexico ranked second, China third, and India fourth.

I then decided to compare different regions. Using `count()` and `tally()` I determined that there were 10 different regions.

```
count(Happiness, `Regional indicator`)
```

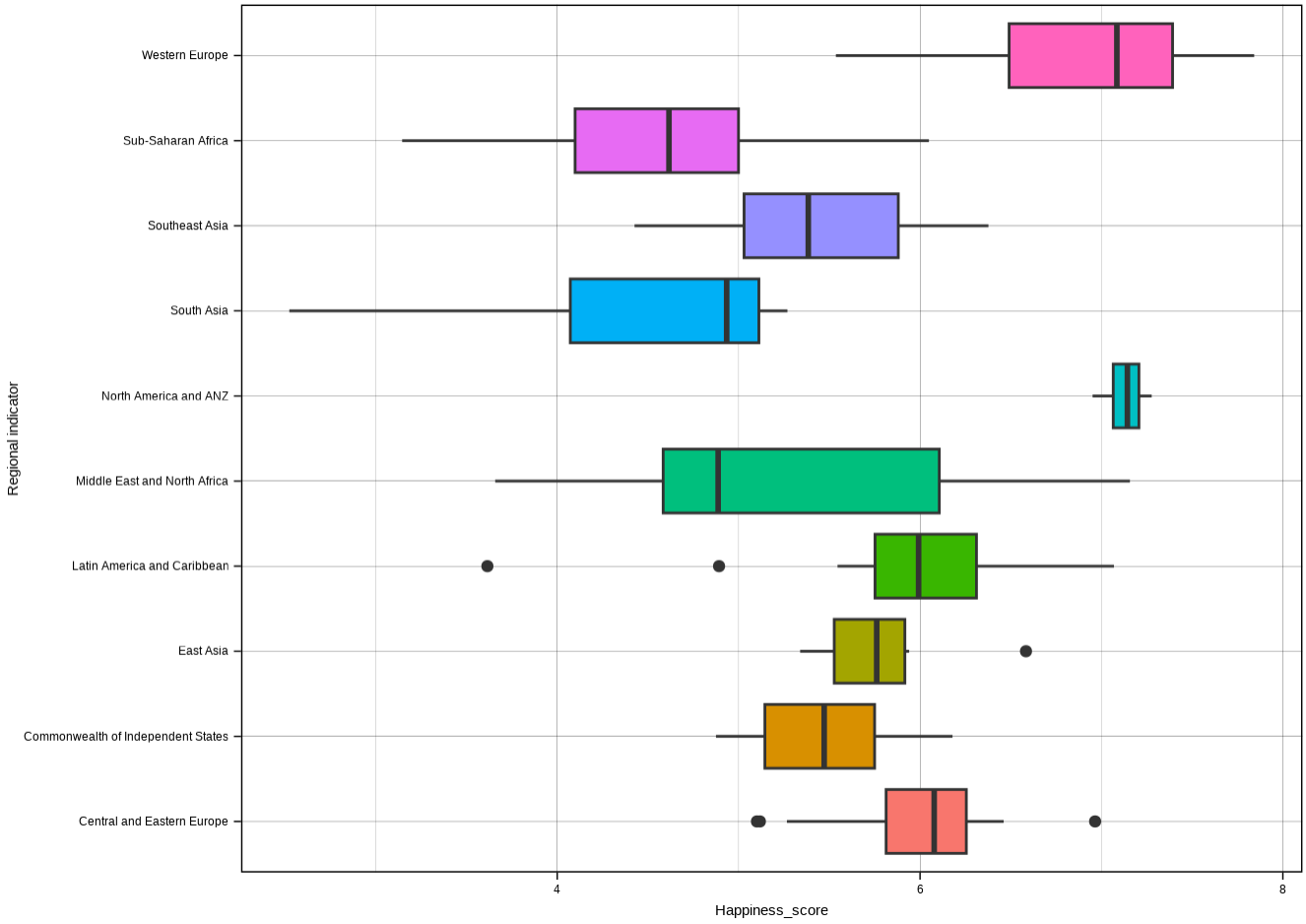
```
## # A tibble: 10 × 2
##    `Regional indicator`                n
##    <chr>                           <int>
##  1 Central and Eastern Europe         17
##  2 Commonwealth of Independent States    12
##  3 East Asia                           6
##  4 Latin America and Caribbean        20
##  5 Middle East and North Africa       17
##  6 North America and ANZ               4
##  7 South Asia                          7
##  8 Southeast Asia                      9
##  9 Sub-Saharan Africa                 36
## 10 Western Europe                     21
```

```
count(Happiness, `Regional indicator`) %>% tally()
```

```
## # A tibble: 1 × 1
##       n
##   <int>
## 1    10
```

I then used `ggplot()` and `geom_boxplot()` to create a boxplot of the happiness scores for the countries in these different regions. I decided to use `coord_flip()` to flip the coordinates to make it easier to read the different region names. I colored the boxplots but using the `fill` argument within the `aes` specifications. I changed the style of the plot with `theme_linedraw()` and I removed the legend using `theme(legend.position = "none")`.

```
ggplot(
  data = Happiness,
  aes(
    x = `Regional indicator`,
    y = Happiness_score,
    fill = `Regional indicator`
  )
) +
  geom_boxplot() +
  theme_linedraw() +
  theme(legend.position = "none") +
  coord_flip()
```



Data Analysis

I then decided to perform a simple analysis to compare countries in different regions. I decided to compare countries in `Western Europe` with those of `East Asia`.

I first filtered the data to create two new data sets for these regions only and then performed a t test without assuming equal variance between the scores for the countries for each region.

```
West_Eur <- Happiness %>%
  filter(`Regional indicator` == "Western Europe")
NAmer <- Happiness %>%
  filter(`Regional indicator` == "North America and ANZ")

t.test(
  x = pull(West_Eur, Happiness_score),
  y = pull(NAmer, Happiness_score),
  var.equal = FALSE
)
```

```
##
##  Welch Two Sample t-test
##
## data:  pull(West_Eur, Happiness_score) and pull(NAmer, Happiness_score)
## t = -1.3431, df = 22.338, p-value = 0.1927
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.5431079  0.1159174
## sample estimates:
## mean of x mean of y
##  6.914905  7.128500
```

The p-value was greater than 0.05, suggesting that there might not be a difference in happiness scores between Western Europe and North America.

Disclaimer - this was intended to demonstrate how one might perform such an analysis. We are not claiming that there is a difference or is not a difference in happiness scores between the countries in these two regions.

Bonus:

If I wanted to perform this analysis on multiple aspects about my data more simply, I might think about making a function like the following:

```
test_reg_diff <- function(aspect) {
  t.test(
    x = pull(West_Eur, aspect),
    y = pull(NAmer, aspect),
    var.equal = FALSE
  )
}
```

This allows me to just specify what aspect (or variable) I would like to compare between the two regions. I can apply it by supplying the name of the column to pull from the datasets.

```
test_reg_diff(aspect = c("Social support"))
```

```
##
##  Welch Two Sample t-test
##
## data:  pull(West_Eur, aspect) and pull(NAmer, aspect)
## t = -1.538, df = 19.734, p-value = 0.1399
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.044847688  0.006800069
## sample estimates:
## mean of x mean of y
## 0.9144762 0.9335000
```

```
test_reg_diff(aspect = c("Healthy life expectancy"))
```

```
##
##  Welch Two Sample t-test
##
## data:  pull(West_Eur, aspect) and pull(NAmer, aspect)
## t = 0.50945, df = 3.0939, p-value = 0.6446
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -3.640256  5.056447
## sample estimates:
## mean of x mean of y
##   73.0331   72.3250
```

This tests if there is a column name that is the same as what is supplied as `aspect`. This then gets pulled from each dataset for the t test.

There appears to be no significant difference according to the p-value for `Social Support` or `Healthy life expectancy`, as they are less than 0.5.

Disclaimer - this was intended to demonstrate how one might perform such an analysis. We are not claiming that there is a difference or is not a difference in social support or healthy life expectancy between the countries in these two regions.

```
sessionInfo()
```

```
## R version 4.5.0 (2025-04-11)
## Platform: x86_64-pc-linux-gnu
## Running under: Ubuntu 24.04.2 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
## LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblasp-r0.3.26.so;  LAPACK version
3.12.0
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## time zone: Etc/UTC
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] readxl_1.4.5           plotly_4.10.4          directlabels_2024.1.21
##  [4] patchwork_1.3.0        emo_0.0.0.9000         scales_1.4.0
##  [7] emoji_16.0.0           knitr_1.50             broom_1.0.8
## [10] esquisse_2.1.0         ThemePark_0.0.1        naniar_1.1.0
## [13] lubridate_1.9.4        forcats_1.0.0          stringr_1.5.1
## [16] dplyr_1.1.4            purrr_1.0.4            readr_2.1.5
## [19] tidyr_1.3.1            tibble_3.2.1           ggplot2_3.5.2
## [22] tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
##  [1] writexl_1.5.4       datamods_1.5.3      rlang_1.1.6
##  [4] magrittr_2.0.3      rio_1.2.3           e1071_1.7-16
##  [7] compiler_4.5.0      systemfonts_1.2.2   vctrs_0.6.5
## [10] quadprog_1.5-8      sysfonts_0.8.9      pkgconfig_2.0.3
## [13] crayon_1.5.3        fastmap_1.2.0       backports_1.5.0
## [16] labeling_0.4.3      utf8_1.2.4          promises_1.3.2
## [19] phosphoricons_0.2.1 rmarkdown_2.29      tzdb_0.5.0
## [22] visdat_0.6.0        ragg_1.4.0          bit_4.6.0
## [25] xfun_0.52           showtext_0.9-7      cachem_1.1.0
## [28] jsonlite_2.0.0      later_1.4.2         parallel_4.5.0
## [31] R6_2.6.1            bslib_0.9.0         stringi_1.8.7
## [34] RColorBrewer_1.1-3  shinybusy_0.3.3     jquerylib_0.1.4
## [37] cellranger_1.1.0    Rcpp_1.0.14         assertthat_0.2.1
## [40] clisymbols_1.2.0    httpuv_1.6.15       timechange_0.3.0
## [43] tidyselect_1.2.1    rstudioapi_0.17.1   yaml_2.3.10
## [46] curl_6.2.2          shiny_1.10.0        withr_3.0.2
## [49] evaluate_1.0.3      proxy_0.4-27        getopt_1.20.4
## [52] zip_2.3.2           pillar_1.10.2       KernSmooth_2.23-26
## [55] generics_0.1.3      vroom_1.6.5         rprojroot_2.0.4
## [58] hms_1.1.3           xtable_1.8-4        class_7.3-23
## [61] glue_1.8.0          toastui_0.4.0       lazyeval_0.2.2
## [64] tools_4.5.0         data.table_1.17.0   reactable_0.4.4
## [67] grid_4.5.0          optparse_1.7.5      crosstalk_1.2.1
## [70] showtextdb_3.0      cli_3.6.5           config_0.3.2
```

```
## [73] textshaping_1.0.0   viridisLite_0.4.2   gtable_0.3.6
## [76] sass_0.4.10         digest_0.6.37       classInt_0.4-11
## [79] htmlwidgets_1.6.4   farver_2.1.2        htmltools_0.5.8.1
## [82] lifecycle_1.0.4     httr_1.4.7          shinyWidgets_0.9.0
## [85] here_1.0.1          mime_0.13           bit64_4.6.0-1
```

How this might get evaluated

According to the guidelines:

1. The student described **where** the data came from including how the instructor could access the data. (0.25 / 0.25 points)
2. The student described information about the data (what the variables mean) and how the data was originally created. (0.25 / 0.25 points)
3. The student performed more than 3 data wrangling methods described in the course, such as arranging data, filtering data, renaming variables, and creating a new variable (`rank`). (3 / 3 points)
4. The student described what steps they did to wrangle the data and why. (1/1 points)
5. The student made two different kinds of visualizations (scatter/point and boxplot). (2 / 2 points)
6. The student performed a simple analysis on the data to compare two regions with a t-test. (2 / 2 points)
7. The student described the analysis and provided some interpretation of the results. (1 / 1 points)
8. The student had `session_info()` print at the end of the analysis (0.5 / 0.5 points)
9. The student made sure that the document knit to create an html file. Bonus: A function was also created worth 1.5 points, but this is not required.

Thus student would get full points (10 out of 10). The extra points from the bonus would not be used to account for less points for other aspects of the project, as the student fulfilled all the other aspects required.