Project 2 – Exploratory Data Analysis (EDA) of Two Data Sets
ALY 6000

Project Instructions

In this two-part project, you will explore operations required to analyze a data set. One package that supports data analysis is the tidyverse package. You may use the tidyverse to complete this project or make use of the built in functionality of R. Displayed answers reflect a view of the data using operations from the tidyverse packages.

Note: Utilize the file **project2_tests.R** with the code below to run a series of tests (not comprehensive) on your code. Any failed test signals that something is wrong with the results or that you have not utilized the specified variable names. These tests will only run correctly if you match the variables names as stated in each problem.

```
p_load(testthat)
#testthat::test_file("project2_tests.R")
```

Setting Up Your Project

Complete the following steps to create and organize your initial R project.

1. Create a new R Project called **Lastname_Project2**.
2. Create a new R Script and save it into the R folder of your project as **Lastname_Project2.R**.
3. Download the data set **2015.csv** from Canvas and save it into the project folder.
4. Download the data set **baseball.csv** from Canvas and save it into the project folder.
5. Load any libraries/packages that you will use as the first lines of the code. For example, if you choose to use the tidyverse package then you would include the following line of code.

```
    library(tidyverse)
```

Assignment Part 1

Data can measure many things. Countries, for example, can be assessed against a variety of metrics. In addition to the gross domestic product (GDP) of a given country, researchers consider other data points in assessing the quality of life across the globe. To understand

how data can be wrangled to measure freedom, trust, and other measures of human life, complete the following steps. The assignment displays the expected outcome after each step.

1. Read the data set **2015.csv** and store it in a variable called **data_2015**. You can test that you loaded it correctly with the code utilizing the head function below.

```
head(data_2015)
```

```
# A tibble: 6 × 12
  Country  Region Happi…¹ Happi…² Stand…³ Econo…⁴ Family Healt…⁵
Freedom Trust…⁶
  <chr>    <chr>    <dbl>   <dbl>   <dbl>   <dbl>  <dbl>   <dbl>
<dbl>   <dbl>
1 Switzer… Weste…       1    7.59  0.0341    1.40   1.35   0.941
0.666   0.420
2 Iceland  Weste…       2    7.56  0.0488    1.30   1.40   0.948
0.629   0.141
3 Denmark  Weste…       3    7.53  0.0333    1.33   1.36   0.875
0.649   0.484
4 Norway   Weste…       4    7.52  0.0388    1.46   1.33   0.885
0.670   0.365
5 Canada   North…       5    7.43  0.0355    1.33   1.32   0.906
0.633   0.330
6 Finland  Weste…       6    7.41  0.0314    1.29   1.32   0.889
0.642   0.414
# … with 2 more variables: Generosity <dbl>, `Dystopia Residual` <dbl>,
and
#   abbreviated variable names ¹`Happiness Rank`, ²`Happiness Score`,
#   ³`Standard Error`, ⁴`Economy (GDP per Capita)`,
#   ⁵`Health (Life Expectancy)`, ⁶`Trust (Government Corruption)`
```

2. Use the function **names** to produce the column names for your data set.

```
names(data_2015)
```

```
 [1] "Country"                       "Region"
 [3] "Happiness Rank"                "Happiness Score"
 [5] "Standard Error"                "Economy (GDP per Capita)"
 [7] "Family"                        "Health (Life Expectancy)"
 [9] "Freedom"                       "Trust (Government Corruption)"
[11] "Generosity"                    "Dystopia Residual"
```

3. Use the **view** function to view the data set in a separate tab.

4. Use the **glimpse** function to view your data set in another configuration.

```
glimpse(data_2015)
```

5. Install and load the **janitor** package. Janitor has a function called **clean_names** that can be given a data frame to make the names more R friendly. Be sure to store the resulting converted data frame in a variable.

```
library(janitor)
data_2015 <- clean_names(data_2015)
data_2015
```

6. Select from the data set the **country**, **region**, **happiness_score**, and **freedom columns**. Store this new table as **happy_df**.

```
# A tibble: 158 × 4
   country     region                    happiness_score freedom
   <chr>       <chr>                               <dbl>   <dbl>
 1 Switzerland Western Europe                       7.59   0.666
 2 Iceland     Western Europe                       7.56   0.629
 3 Denmark     Western Europe                       7.53   0.649
 4 Norway      Western Europe                       7.52   0.670
 5 Canada      North America                        7.43   0.633
 6 Finland     Western Europe                       7.41   0.642
 7 Netherlands Western Europe                       7.38   0.616
 8 Sweden      Western Europe                       7.36   0.660
 9 New Zealand Australia and New Zealand            7.29   0.639
10 Australia   Australia and New Zealand            7.28   0.651
# … with 148 more rows
```

7. Slice the first 10 rows from **happy_df** and store it as **top_ten_df**.

```
# A tibble: 10 × 4
   country     region                    happiness_score freedom
   <chr>       <chr>                               <dbl>   <dbl>
 1 Switzerland Western Europe                       7.59   0.666
 2 Iceland     Western Europe                       7.56   0.629
 3 Denmark     Western Europe                       7.53   0.649
 4 Norway      Western Europe                       7.52   0.670
 5 Canada      North America                        7.43   0.633
 6 Finland     Western Europe                       7.41   0.642
 7 Netherlands Western Europe                       7.38   0.616
 8 Sweden      Western Europe                       7.36   0.660
 9 New Zealand Australia and New Zealand            7.29   0.639
10 Australia   Australia and New Zealand            7.28   0.651
```

8. From **happy_df** filter the table for freedom values under 0.20. Store this new table as **no_freedom_df.**

```
# A tibble: 12 × 4
   country               region
happiness_sc…¹ freedom
   <chr>                 <chr>
<dbl>   <dbl>
```

```
 1 Pakistan                Southern Asia
5.19  0.121
 2 Montenegro              Central and Eastern Europe
5.19  0.183
 3 Bosnia and Herzegovina Central and Eastern Europe
4.95  0.0924
 4 Greece                  Western Europe
4.86  0.0770
 5 Iraq                    Middle East and Northern Africa
4.68  0
 6 Sudan                   Sub-Saharan Africa
4.55  0.101
 7 Armenia                 Central and Eastern Europe
4.35  0.198
 8 Egypt                   Middle East and Northern Africa
4.19  0.173
 9 Angola                  Sub-Saharan Africa
4.03  0.104
10 Madagascar              Sub-Saharan Africa
3.68  0.192
11 Syria                   Middle East and Northern Africa
3.01  0.157
12 Burundi                 Sub-Saharan Africa
2.90  0.118
# … with abbreviated variable name ¹happiness_score
```

9. Arrange the values in **happy_df** in descending order by their freedom values. Store this new table as **best_freedom_df**.

```
# A tibble: 158 × 4
   country              region                          happiness_score
freedom
   <chr>                <chr>                                     <dbl>
<dbl>
 1 Norway               Western Europe                             7.52
0.670
 2 Switzerland          Western Europe                             7.59
0.666
 3 Cambodia             Southeastern Asia                          3.82
0.662
 4 Sweden               Western Europe                             7.36
0.660
 5 Uzbekistan           Central and Eastern Europe                 6.00
0.658
 6 Australia            Australia and New Zealand                  7.28
0.651
 7 Denmark              Western Europe                             7.53
0.649
 8 Finland              Western Europe                             7.41
0.642
```

```
 9 United Arab Emirates Middle East and Northern Africa            6.90
0.642
10 Qatar                Middle East and Northern Africa            6.61
0.640
# … with 148 more rows
```

10. Create a new column **data_2015** called **gff_stat.** For each row, the **gff_stat** is the sum of the family, freedom, and generosity values. Store the resulting table back into the **data_2015** variable.

```
# A tibble: 158 × 13
   country region happi…¹ happi…² stand…³ econo…⁴ family healt…⁵
freedom trust…⁶
   <chr>   <chr>    <dbl>   <dbl>   <dbl>   <dbl>  <dbl>   <dbl>
<dbl>   <dbl>
 1 Switze… Weste…       1    7.59  0.0341    1.40   1.35   0.941
0.666   0.420
 2 Iceland Weste…       2    7.56  0.0488    1.30   1.40   0.948
0.629   0.141
 3 Denmark Weste…       3    7.53  0.0333    1.33   1.36   0.875
0.649   0.484
 4 Norway  Weste…       4    7.52  0.0388    1.46   1.33   0.885
0.670   0.365
 5 Canada  North…       5    7.43  0.0355    1.33   1.32   0.906
0.633   0.330
 6 Finland Weste…       6    7.41  0.0314    1.29   1.32   0.889
0.642   0.414
 7 Nether… Weste…       7    7.38  0.0280    1.33   1.28   0.893
0.616   0.318
 8 Sweden  Weste…       8    7.36  0.0316    1.33   1.29   0.911
0.660   0.438
 9 New Ze… Austr…       9    7.29  0.0337    1.25   1.32   0.908
0.639   0.429
10 Austra… Austr…      10    7.28  0.0408    1.33   1.31   0.932
0.651   0.356
# … with 148 more rows, 3 more variables: generosity <dbl>,
#   dystopia_residual <dbl>, gff_stat <dbl>, and abbreviated variable
names
#   ¹happiness_rank, ²happiness_score, ³standard_error,
#   ⁴economy_gdp_per_capita, ⁵health_life_expectancy,
#   ⁶trust_government_corruption
```

11. Group the **happy_df** data set by region. Run a summary that provides the number of countries in each region in a column called **country_count**, the **mean** happiness for each region in a column called **mean_happiness**, and the **mean** freedom of each region in a column called **mean_freedom.** Store your resulting table in a variable called **regional_stats_df.**

```
# A tibble: 10 × 4
   region                          country_count mean_happiness
```

```
   mean_freedom
      <chr>                                <int>          <dbl>
   <dbl>
    1 Australia and New Zealand                2          7.28
   0.645
    2 Central and Eastern Europe              29          5.33
   0.358
    3 Eastern Asia                             6          5.63
   0.462
    4 Latin America and Caribbean             22          6.14
   0.502
    5 Middle East and Northern Africa         20          5.41
   0.362
    6 North America                            2          7.27
   0.590
    7 Southeastern Asia                        9          5.32
   0.557
    8 Southern Asia                            7          4.58
   0.373
    9 Sub-Saharan Africa                      40          4.20
   0.366
   10 Western Europe                          21          6.69
   0.550
```

Assignment Part 2

In Part Two of this R Project, you will analyze a data set of batting statistics from the 1986 Major League Baseball season. You will then draft a brief executive summary that corresponds to the data analysis. Details for both the data analysis and executive summary follow below.

12. Download the **baseball.csv** data set that represents batting statistics from the 1986 Major League Baseball season. Read this data set in a **variable** called **baseball.**
13. Spend time with the data using various exploration functions to get a general feel for what you are working with. For more information on this data set and its various columns, see Baseball Reference's 1986 Major League Standard Batting.

14. Remove (**filter)** from **baseball** any player with 0 at bats (AB). Store the result in **baseball.**

```
   # A tibble: 726 × 16
      Last  First   Age     G    PA    AB     R     H  `2B`  `3B`    HR
   RBI    SB
      <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
   <dbl> <dbl>
    1 Acker Jim      27    21    28    28     1     3     1     0     0
   0     0
```

```
 2 Addu… Jim      26     3    13    11     2     1     1     0     0
0     0
 3 Agua… Luis     27    62   146   133    17    28     6     1     4
13     1
 4 Agui… Rick     24    32    57    51     4     8     0     0     2
6     0
 5 Aldr… Mike     25    84   256   216    27    54    18     3     2
25     1
 6 Alex… Doyle    35    18    45    38     2     8     1     0     0
5     0
 7 Alla… Andy     24   101   324   293    30    66     7     3     1
29    10
 8 Almon Bill     33   102   230   196    29    43     7     2     7
27    11
 9 Amel… Ed       27     8    11    11     0     1     0     0     0
0     0
10 Ande… Larry    33    48     7     6     0     0     0     0     0
0     0
# … with 716 more rows, and 3 more variables: CS <dbl>, BB <dbl>, SO
<dbl>
```

15. Add a new column batting average called **BA.** Batting average is computed by the number of hits (H) divided by the number of at bats (AB). Store the result in **baseball.**

```
# A tibble: 726 × 17
   Last  First   Age     G    PA    AB     R     H  `2B`  `3B`    HR
RBI    SB
   <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
<dbl> <dbl>
 1 Acker Jim      27    21    28    28     1     3     1     0     0
0     0
 2 Addu… Jim      26     3    13    11     2     1     1     0     0
0     0
 3 Agua… Luis     27    62   146   133    17    28     6     1     4
13     1
 4 Agui… Rick     24    32    57    51     4     8     0     0     2
6     0
 5 Aldr… Mike     25    84   256   216    27    54    18     3     2
25     1
 6 Alex… Doyle    35    18    45    38     2     8     1     0     0
5     0
 7 Alla… Andy     24   101   324   293    30    66     7     3     1
29    10
 8 Almon Bill     33   102   230   196    29    43     7     2     7
27    11
 9 Amel… Ed       27     8    11    11     0     1     0     0     0
0     0
10 Ande… Larry    33    48     7     6     0     0     0     0     0
0     0
# … with 716 more rows, and 4 more variables: CS <dbl>, BB <dbl>, SO
```

```
<dbl>,
#   BA <dbl>
```

16. On-base percentage (OBP) is arguably a better statistic than batting average. Create a column called **OBP** that computes this stat as (H + BB) / (AB + BB). Store the result in **baseball.**

```
# A tibble: 726 × 18
   Last  First   Age     G    PA    AB     R     H  `2B`  `3B`    HR
RBI    SB
   <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
<dbl> <dbl>
 1 Acker Jim      27    21    28    28     1     3     1     0     0
0     0
 2 Addu… Jim      26     3    13    11     2     1     1     0     0
0     0
 3 Agua… Luis     27    62   146   133    17    28     6     1     4
13     1
 4 Agui… Rick     24    32    57    51     4     8     0     0     2
6     0
 5 Aldr… Mike     25    84   256   216    27    54    18     3     2
25     1
 6 Alex… Doyle    35    18    45    38     2     8     1     0     0
5     0
 7 Alla… Andy     24   101   324   293    30    66     7     3     1
29    10
 8 Almon Bill     33   102   230   196    29    43     7     2     7
27    11
 9 Amel… Ed       27     8    11    11     0     1     0     0     0
0     0
10 Ande… Larry    33    48     7     6     0     0     0     0     0
0     0
# … with 716 more rows, and 5 more variables: CS <dbl>, BB <dbl>, SO
<dbl>,
#   BA <dbl>, OBP <dbl>
```

17. Determine the 10 players who struck out the most this season. Store these results as **strikeout_artist.**

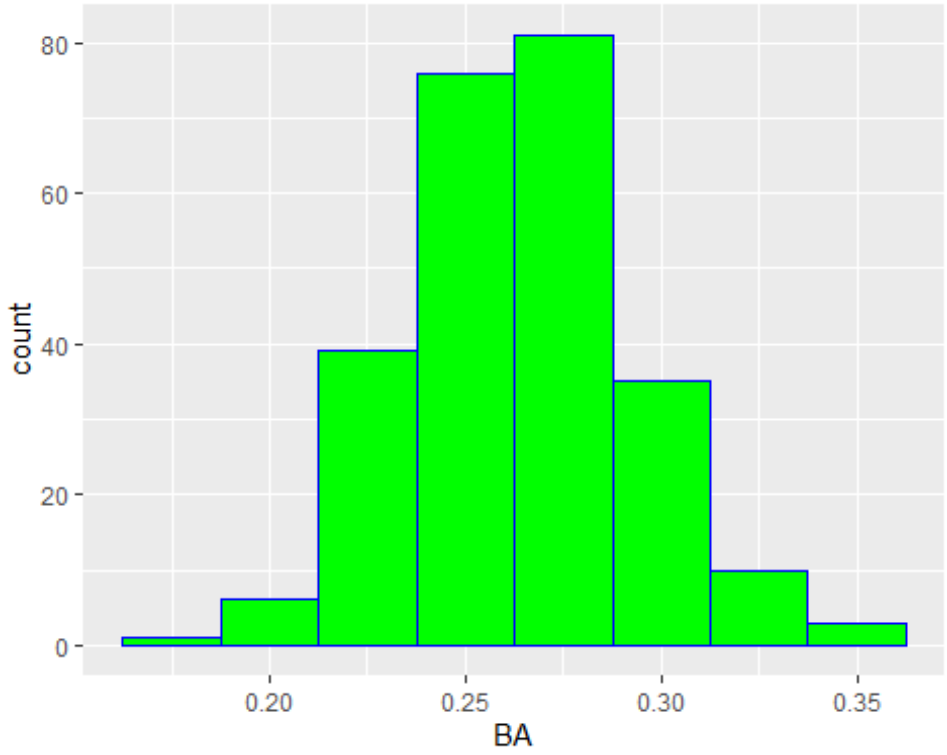
```
# A tibble: 10 × 18
   Last  First   Age     G    PA    AB     R     H  `2B`  `3B`    HR
RBI    SB
   <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
<dbl> <dbl>
 1 Inca… Pete     22   153   606   540    82   135    21     2    30
88     3
 2 Deer  Rob      25   134   546   466    75   108    17     3    33
86     5
 3 Cans… Jose     21   157   682   600    85   144    29     1    33
117    15
 4 Pres… Jim      24   155   660   616    83   163    33     4    27
107     0
```

```
 5 Tart… Danny    23   137   578   511    76   138    25     6    25
96     4
 6 Balb… Steve    29   138   562   512    54   117    25     1    29
88     0
 7 Barf… Jesse    26   158   671   589   107   170    35     2    40
108     8
 8 Samu… Juan     25   145   633   591    90   157    36    12    16
78    42
 9 Murp… Dale     30   160   692   614    89   163    29     7    29
83     7
10 Stra… Darr…    24   136   562   475    76   123    27     5    27
93    28
# … with 5 more variables: CS <dbl>, BB <dbl>, SO <dbl>, BA <dbl>, OBP
<dbl>
```

18. To be eligible for end-of-season awards, a player must have either at least 300 at bats or appear in at least 100 games. Keep only the players who are eligible to be considered and store them in a variable called **eligible_df.**

```
# A tibble: 251 × 18
   Last  First   Age     G    PA    AB     R     H  `2B`  `3B`    HR
RBI    SB
   <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
<dbl> <dbl>
 1 Alla… Andy     24   101   324   293    30    66     7     3     1
29    10
 2 Almon Bill     33   102   230   196    29    43     7     2     7
27    11
 3 Armas Tony     32   121   453   425    40   112    21     4    11
58     0
 4 Ashby Alan     34   120   361   315    24    81    15     0     7
38     1
 5 Back… Wally    26   124   440   387    67   124    18     2     1
27    13
 6 Bain… Haro…    27   145   618   570    72   169    29     2    21
88     2
 7 Balb… Steve    29   138   562   512    54   117    25     1    29
88     0
 8 Barf… Jesse    26   158   671   589   107   170    35     2    40
108     8
 9 Barr… Marty    28   158   713   625    94   179    39     4     4
60    15
10 Bass  Kevin    27   157   640   591    83   184    33     5    20
79    22
# … with 241 more rows, and 5 more variables: CS <dbl>, BB <dbl>, SO
<dbl>,
#   BA <dbl>, OBP <dbl>
```

19. For eligible players, create a histogram of batting average.



20. Important statistics for baseball players include the on-base percentage (OBP), the number of home runs (HR), the number of runs batted-in (RBI) among others. Analyze the eligible players and select a player that in your opinion is deserving of the Most Valuable Player (MVP) award. This choice must be supported by your data. In your report, you should present your data analysis supported by relevant data points and statistics that supports your recommendation. Produce a concise, written executive summary that focuses on the baseball data analysis. In addition to the title page and citations, it contains an introduction, presentation of written key findings, and a conclusion that contains your recommendations as supported by the data. Your executive summary should adhere to basic APA guidelines.

Submitting to Canvas

When you are satisfied with your solution, take the following steps:

1. **Remove** any lines in your code with "install.packages."
2. **Remove** any lines in your code that use the **view** function.
3. Submit two (2) files under the appropriate assignment in Canvas:
    1. Your R script named **Lastname_Project2.R**.
    2. A PDF file of your four-page report titled **Lastname_Project2_Report.pdf**.

Your report (on the baseball analysis only) should contain the following information formatted as specified below:

Title Page
Include your name, assignment title, and submission date

Introduction and Key Findings
Include an overview of the assignment and any findings

Conclusion/Recommendations
Include evidence-based recommendations and visualizations or direct presentation of tabular data

Works Cited
Include all sources, including YouTube videos, instruction materials, Google search results, and texts that informed your study of statistics and R

Your report should be as concise as possible while maintaining fluency. Your key findings will be strongest if supported by visualizations or direct presentation of tabular data.

Your summary must adhere to APA guidelines, including page numbers on each page (including the title page) in the upper right corner. See the following examples for title pages, citations, and general APA formatting.

Congratulations on completing your second project!