Getting Started with R

Part 2

Table of contents

1 The CSV & TSV File Formats

`CSV` (**Comma**-Separated Values) and `TSV` (**Tab**-Separated Values) are **plain text** formats used for storing data in a tabular structure. Both formats human-readable and easy to handle in many programming environments, including R.

- CSV (Comma-Separated Values):
  - Fields (columns) are separated by commas.
  - Lines (rows) are separated by line breaks.
  - Commonly used due to its simplicity and broad application in systems that handle tabular data.
  - Example:

```
Name,Age,Occupation
Alice,28,Engineer
Bob,35,Data Scientist
```

- TSV (Tab-Separated Values):
  - Fields are separated by tabs.
  - Often preferred when data values may contain commas, to avoid confusion.
  - Example:

```
Name    Age    Occupation
Alice    28    Engineer
Bob    35    Data Scientist
```

2 Loading Data in R

- There are different ways (functions) to read (or load, or import) data files into R.
- One simple and easy way is using the `read.csv()` function.
- Example:

```r
df = read.csv("filename.csv")
```

3 The Dataset: The Happiness Index 2019

The World Happiness Report 2019

4 Reading the Happines Dataset

```
df = read.csv("https://raw.githubusercontent.com/ahmedmoustafa/datasets/main/happiness/hap
head(df)
```

country	category	score	gdp__per__capita	social__support	healthy__life__expectancy	freedom__to__make__life__choices	generosity	perceptions__of__corruption
Afghanistan	Underdeveloped	3.203	0.350	0.517	0.361	0.000	0.158	0.025
Albania	Transitioning	4.719	0.947	0.848	0.874	0.383	0.178	0.027
Algeria	Developing	5.211	1.002	1.160	0.785	0.086	0.073	0.114
Argentina	Developing	6.086	1.092	1.432	0.881	0.471	0.066	0.050
Armenia	Transitioning	4.559	0.850	1.055	0.815	0.283	0.095	0.064
Australia	Developed	7.228	1.372	1.548	1.036	0.557	0.332	0.290

5 Exploring the Structure of the Dataset

- **Shape of the Data**: check the dimensions (number of rows and columns) of the dataset

```
dim(df)
```

```
[1] 155   9
```

```
paste("Number of rows (countries):", nrow(df))
```

```
[1] "Number of rows (countries): 155"
```

```
paste("Number of columns (attributes):", ncol(df))
```

```
[1] "Number of columns (attributes): 9"
```

- **Column Names**: generate a list of all the attributes/columns in the dataset

```
colnames(df)
```

```
[1] "country"                      "category"
[3] "score"                        "gdp_per_capita"
[5] "social_support"               "healthy_life_expectancy"
[7] "freedom_to_make_life_choices" "generosity"
[9] "perceptions_of_corruption"
```

- **Column Data Types**: understand the kind of data each column holds (numeric, character, factor, etc.).

```
sapply(df, class)
```

```
                     country                     category
                 "character"                  "character"
                       score                gdp_per_capita
                   "numeric"                    "numeric"
              social_support      healthy_life_expectancy
                   "numeric"                    "numeric"
freedom_to_make_life_choices                  generosity
                   "numeric"                    "numeric"
   perceptions_of_corruption
                   "numeric"
```

6 Categorical Variables

- Which columns make sense to be converted to `factor`? `category` is a **qualitative** variable.

```
df$category = factor(df$category)
levels(df$category)
```

```
[1] "Developed"      "Developing"     "Transitioning"  "Underdeveloped"
```

- It is actually an **ordinal** qualitative variable. So, instead of the default levels (alphabetical), let's provide a real order.

```
df$category = factor(df$category, levels = c("Developed", "Transitioning", "Developing", "
levels(df$category)
```

```
[1] "Developed"      "Transitioning"  "Developing"     "Underdeveloped"
```

7 The Happiest Country

- We need to determine the highest score using the `max()` function, then locate the index (position) of the country with that max score using the `which()` function.

```
df$country[which(df$score == max(df$score))]
```

```
[1] "Finland"
```

- Alternatively, there is also the 2-in-1 function `which.max()`

```
df$country[which.max(df$score)]
```

```
[1] "Finland"
```

8 The Least Happy Country

```r
df$country[which(df$score == min(df$score))]
```

```
[1] "South Sudan"
```

```r
df$country[which.min(df$score)]
```

```
[1] "South Sudan"
```

9 The Top 10 Happiest Countries

We need to obtain the **descending** `order()` of the countries according to the `score` column then obtain the first 10:

- Using the `decreasing` parameter:

```r
df$country[order(df$score, decreasing = TRUE)][1:10]
```

```
 [1] "Finland"     "Denmark"     "Norway"      "Iceland"     "Netherlands"
 [6] "Switzerland" "Sweden"      "New Zealand" "Canada"      "Austria"
```

- Using the **negative** (`-`) scores:

```r
df$country[order(-df$score)][1:10]
```

```
 [1] "Finland"     "Denmark"     "Norway"      "Iceland"     "Netherlands"
 [6] "Switzerland" "Sweden"      "New Zealand" "Canada"      "Austria"
```

10 The Top 10 Happiest Countries

- The full records of the top 10 happiest countries:

```
df[order(-df$score)[1:10], ]
```

	country	category	score	gdp_per_capita	social_support	healthy_life_expectancy	freedom_to_make_life_choices	generosity	perceptions_of_corruption
44	Finland	Developed	7.769	1.340	1.587	0.986	0.596	0.153	0.393
37	Denmark	Developed	7.600	1.383	1.573	0.996	0.592	0.252	0.410
105	Norway	Developed	7.554	1.488	1.582	1.028	0.603	0.271	0.341
58	Iceland	Developed	7.494	1.380	1.624	1.026	0.591	0.354	0.118
99	Netherlands	Developed	7.488	1.396	1.522	0.999	0.557	0.322	0.298
133	Switzerland	Developed	7.480	1.452	1.526	1.052	0.572	0.263	0.343
132	Sweden	Developed	7.343	1.387	1.487	1.009	0.574	0.267	0.373
100	New Zealand	Developed	7.307	1.303	1.557	1.026	0.585	0.330	0.380
24	Canada	Developed	7.278	1.365	1.505	1.039	0.584	0.285	0.308
7	Austria	Developed	7.246	1.376	1.475	1.016	0.532	0.244	0.226

11 Egypt's Happy Score & Rank

- Find the row number (index) with the country **equals** (==) "Egypt" to obtain the score in that row (at that index)

```
df$score[which(df$country == "Egypt")]
```

```
[1] 4.166
```

- Similarly, we obtain the row number of Egypt then use that index to obtain the corresponding rank of the score, after **ranking** the scores

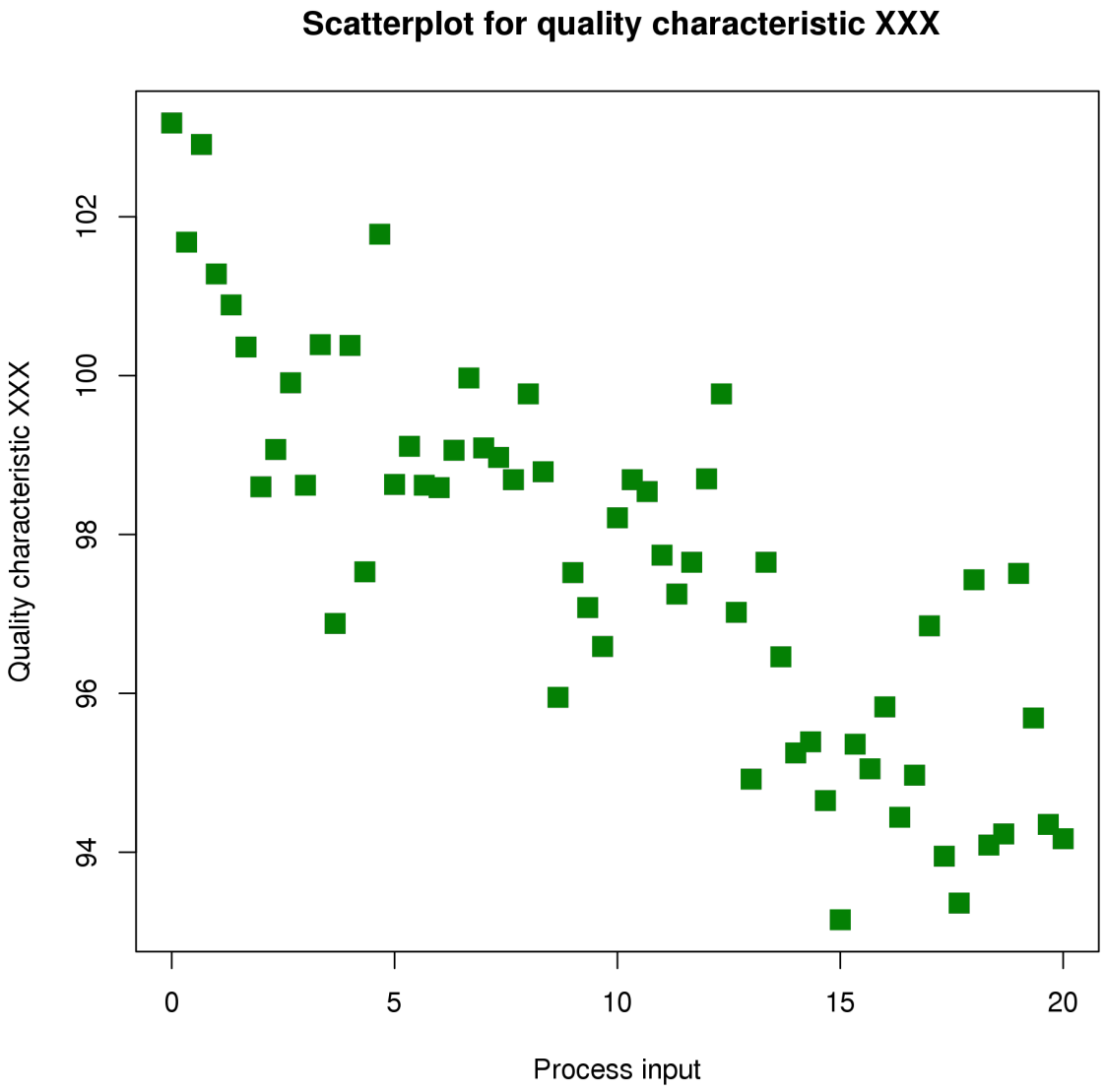
```
rank(-df$score)[which(df$country == "Egypt")]
```

```
[1] 136
```

- Note the use of the **negative sign** (-) above with the score to switch the direction of ranking from ascending (which is the default) to descending

12 A Glimpse of Data Visualization in R

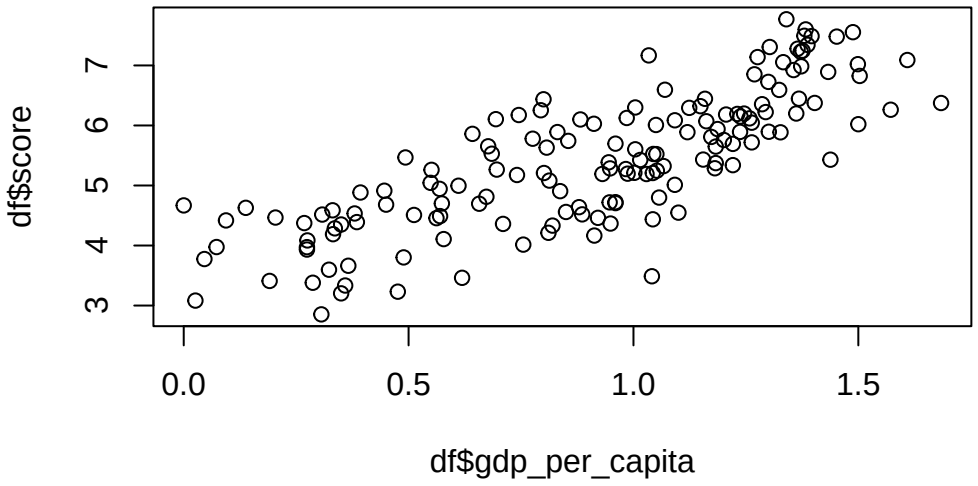
- Using the basic `plot` function in R, we can visualize the relationship between two variables as a scatter plot.



- For example, let's investigate the relationship between the `score` (on the y-axis) and the `gdp_per_capita`

13 Relationship Between Happiness and GDP, Visually

```
plot(df$score ~ df$gdp_per_capita)
```



14 Relationship Between Happiness and GDP, Quantitatively

```
cor(df$score, df$gdp_per_capita)
```

```
[1] 0.7937202
```

Both the graph and the correlation coefficient suggest a strong association between population happiness and the country's GDP.