Tidyverse Intro: Travel and Weather - ggplot2

ETM 58D - Spring 2018

Mar 12, 2018

Introduction

This part of the exercise explains the basics of `ggplot2`. See the first part for preparations. We start directly from the data set.

```
travel_weather %>%
    tbl_df()
```

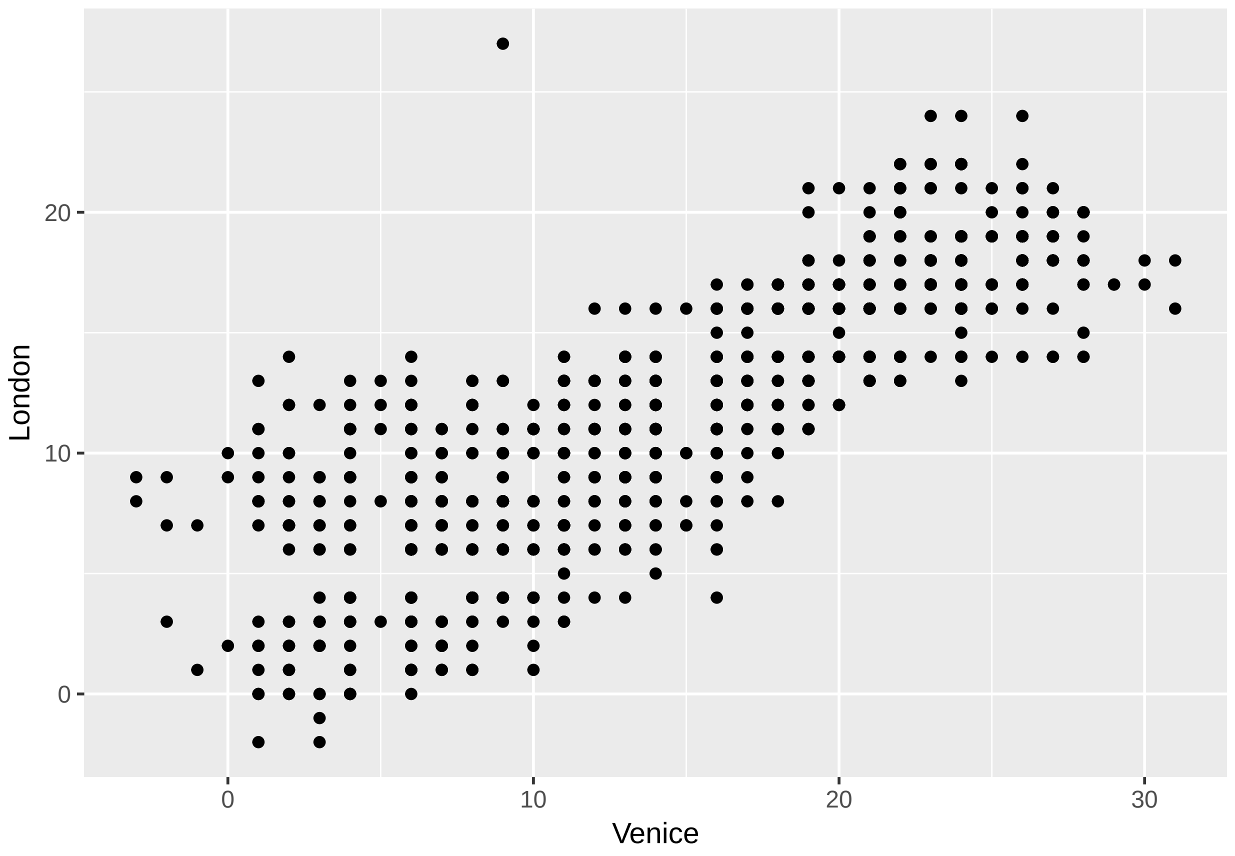
```
## # A tibble: 731 x 7
##      year month   day Amsterdam London   NYC Venice
##   * <dbl> <dbl> <dbl>     <dbl>  <dbl> <dbl>  <dbl>
##   1  2015  11.0  1.00      8.00   8.00  16.0  13.0
##   2  2015  11.0  2.00     10.0   11.0   15.0  10.0
##   3  2015  11.0  3.00      9.00  11.0   16.0   9.00
##   4  2015  11.0  4.00     12.0   11.0   17.0  10.0
##   5  2015  11.0  5.00     13.0   13.0   18.0  12.0
##   6  2015  11.0  6.00     16.0   14.0   21.0  13.0
##   7  2015  11.0  7.00     16.0   14.0   17.0  14.0
##   8  2015  11.0  8.00     12.0   12.0   11.0  13.0
##   9  2015  11.0  9.00     13.0   12.0   11.0  11.0
##  10  2015  11.0 10.0      14.0   14.0   12.0  11.0
## # ... with 721 more rows
```

You are used to pipe operator (`%>%`) for dplyr. While dplyr provides a chain of events, the method of thinking in `ggplot2` is similar to putting layers on top of each other, starting with a canvas. Therefore we use `+` operator to connect the functions.

Scatterplot

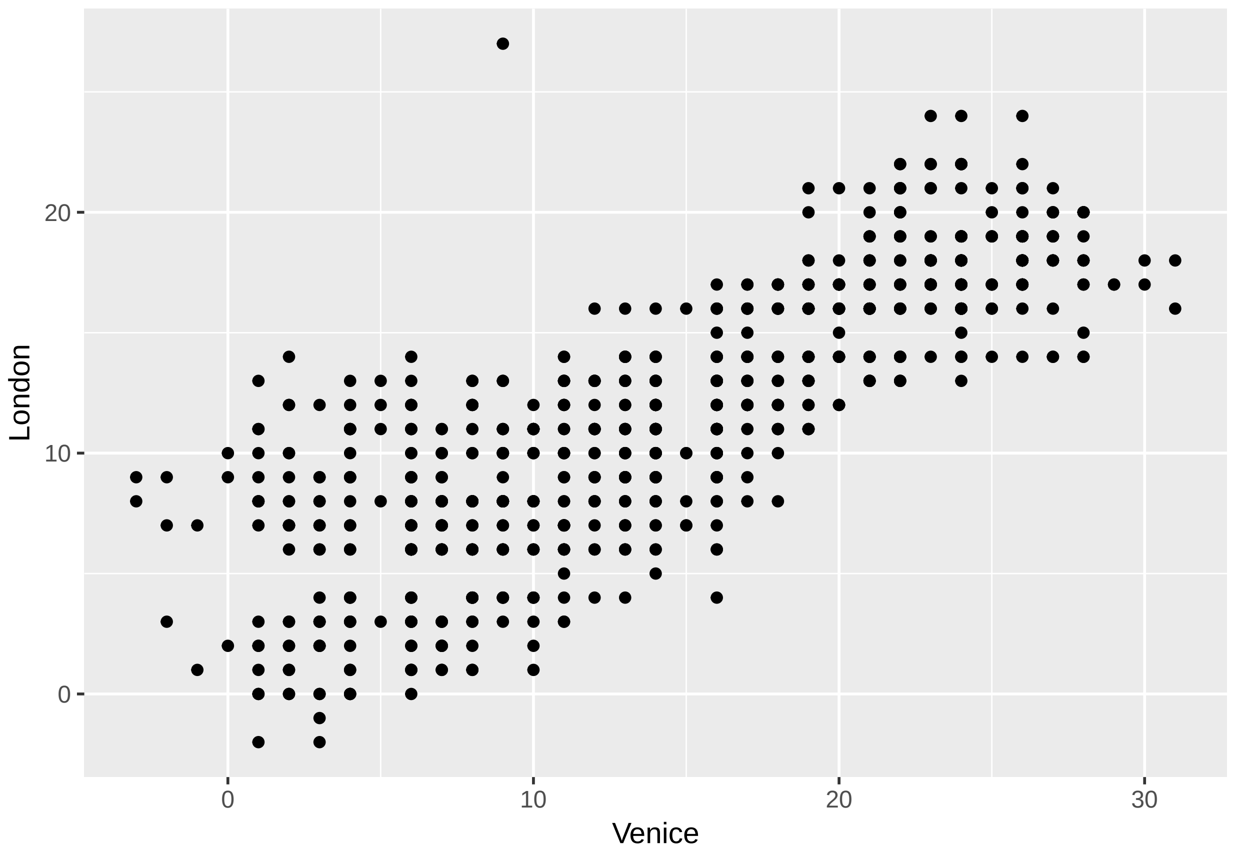
Scatterplot is the first chart we are going to learn, along with the basics of `ggplot2` anatomy. We start with the canvas function `ggplot`. `aes` is the aesthetics part where we define x and y axes along with other grouping variables (e.g. color, fill, alpha, shape, size). Once we set the data and aesthetics we declare the plot type we would like to show. For scatterplot the function is `geom_point`.

```
ggplot(data = travel_weather, aes(x = Venice, y = London)) +
    geom_point()
```



By the way, `ggplot2` is quite flexible in terms of placements of elements. Though, use them responsibly. Below is the same chart as above with different representation and some mixing with `dplyr`.

```
travel_weather %>% ggplot() + geom_point(aes(x = Venice, y = London))
```

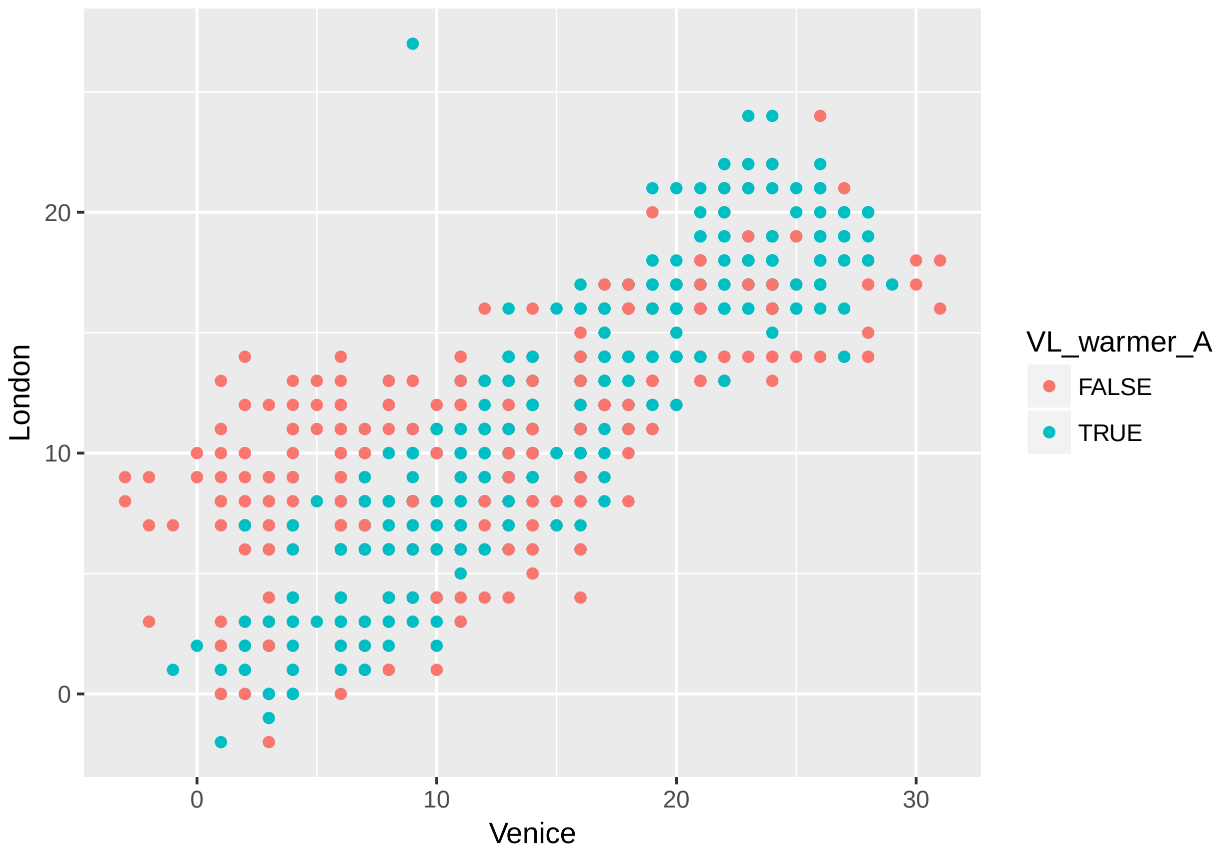


Let's color this chart a bit. If both Venice and London are warmer than Amsterdam, let's show it with some color.

```
travel_weather %>% mutate(VL_warmer_A = pmin(Venice, London) >=
    Amsterdam)
```

```
## # A tibble: 731 x 8
##     year month   day Amsterdam London   NYC Venice VL_warmer_A
##    <dbl> <dbl> <dbl>     <dbl>  <dbl> <dbl>  <dbl> <lgl>
##  1  2015  11.0  1.00      8.00   8.00  16.0  13.0  T
##  2  2015  11.0  2.00     10.0   11.0   15.0  10.0  T
##  3  2015  11.0  3.00      9.00  11.0   16.0   9.00 T
##  4  2015  11.0  4.00     12.0   11.0   17.0  10.0  F
##  5  2015  11.0  5.00     13.0   13.0   18.0  12.0  F
##  6  2015  11.0  6.00     16.0   14.0   21.0  13.0  F
##  7  2015  11.0  7.00     16.0   14.0   17.0  14.0  F
##  8  2015  11.0  8.00     12.0   12.0   11.0  13.0  T
##  9  2015  11.0  9.00     13.0   12.0   11.0  11.0  F
## 10  2015  11.0 10.0      14.0   14.0   12.0  11.0  F
## # ... with 721 more rows
```

```
travel_weather %>% mutate(VL_warmer_A = pmin(Venice, London) >=
    Amsterdam) %>% ggplot() + geom_point(aes(x = Venice, y = London,
    color = VL_warmer_A))
```

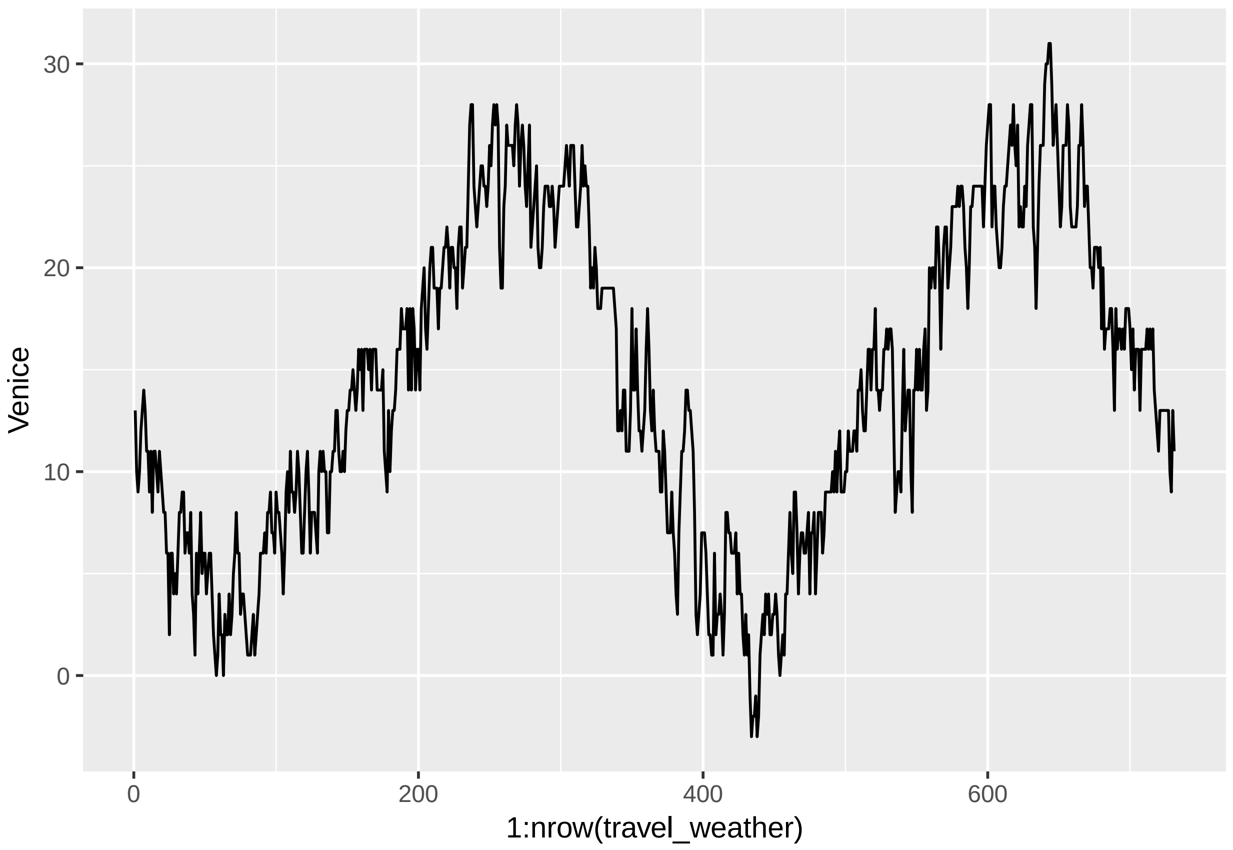


Refer to ggplot2 tutorials and cheat sheets for more "tricks".

Line Chart

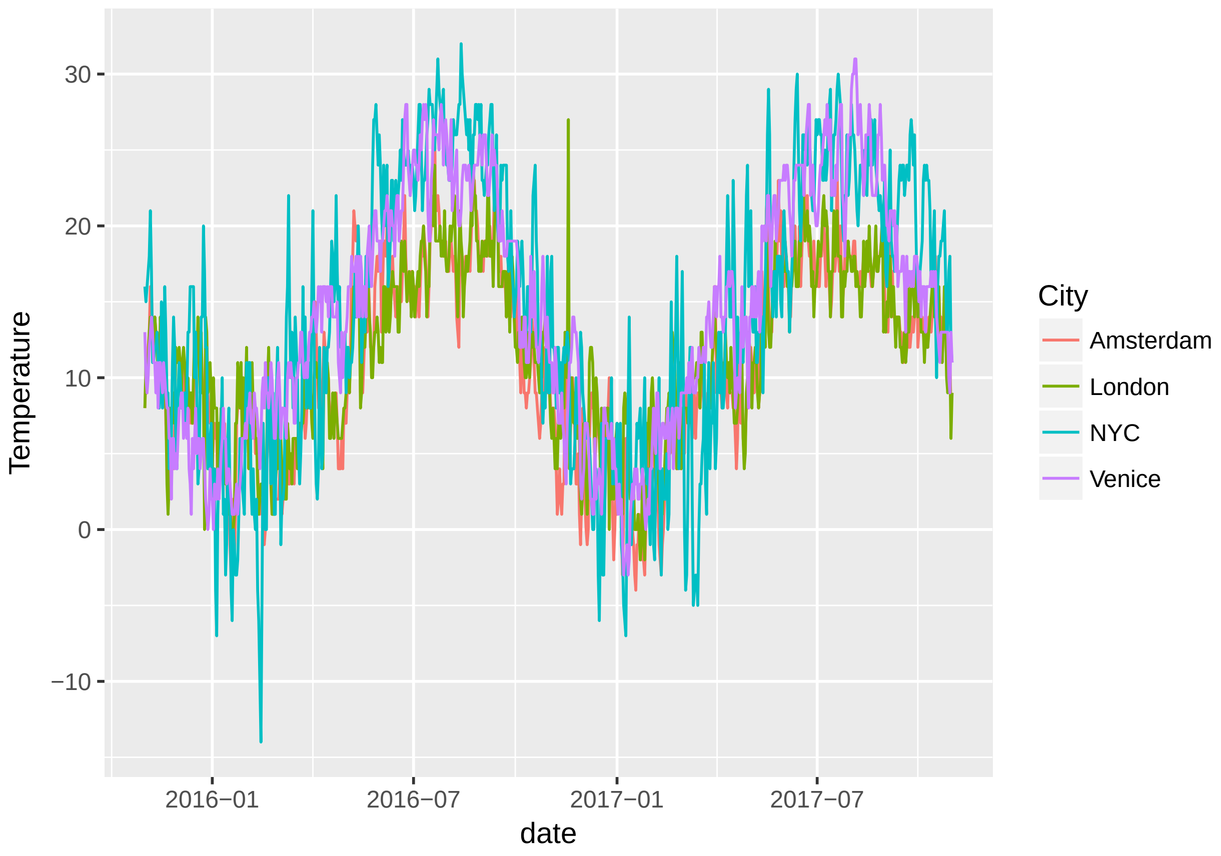
Unsurprisingly we are going to use `geom_line` here.

```r
ggplot(data = travel_weather, aes(x = 1:nrow(travel_weather),
    y = Venice)) + geom_line()
```



Let's make it more beautiful (kinda).

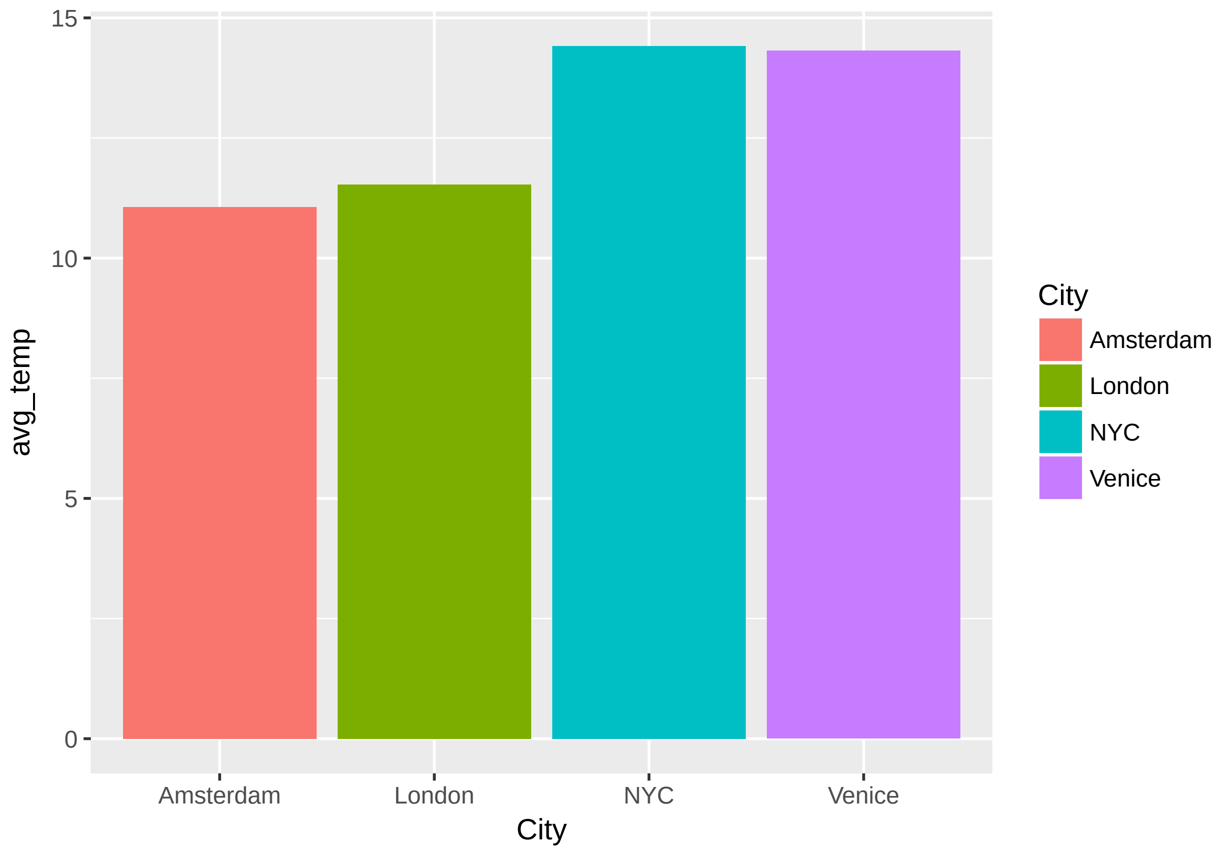
```
travel_weather %>% rowwise() %>% mutate(date = lubridate::as_date(paste(year,
    as.integer(month), as.integer(day), sep = "-"))) %>% ungroup() %>%
    select(-(year:day)) %>% gather(key = City, value = Temperature,
    -date) %>% ggplot(data = ., aes(x = date, y = Temperature,
    color = City)) + geom_line()
```



Bar Chart

Suppose you want to compare average temperatures of each city.

```
travel_weather %>% select(Amsterdam:Venice) %>% gather(key = City,
    value = Temperature) %>% group_by(City) %>% summarise(avg_temp = mean(Temperature,
    na.rm = T)) %>% ggplot(data = ., aes(x = City, y = avg_temp,
    fill = City)) + geom_bar(stat = "identity")
```



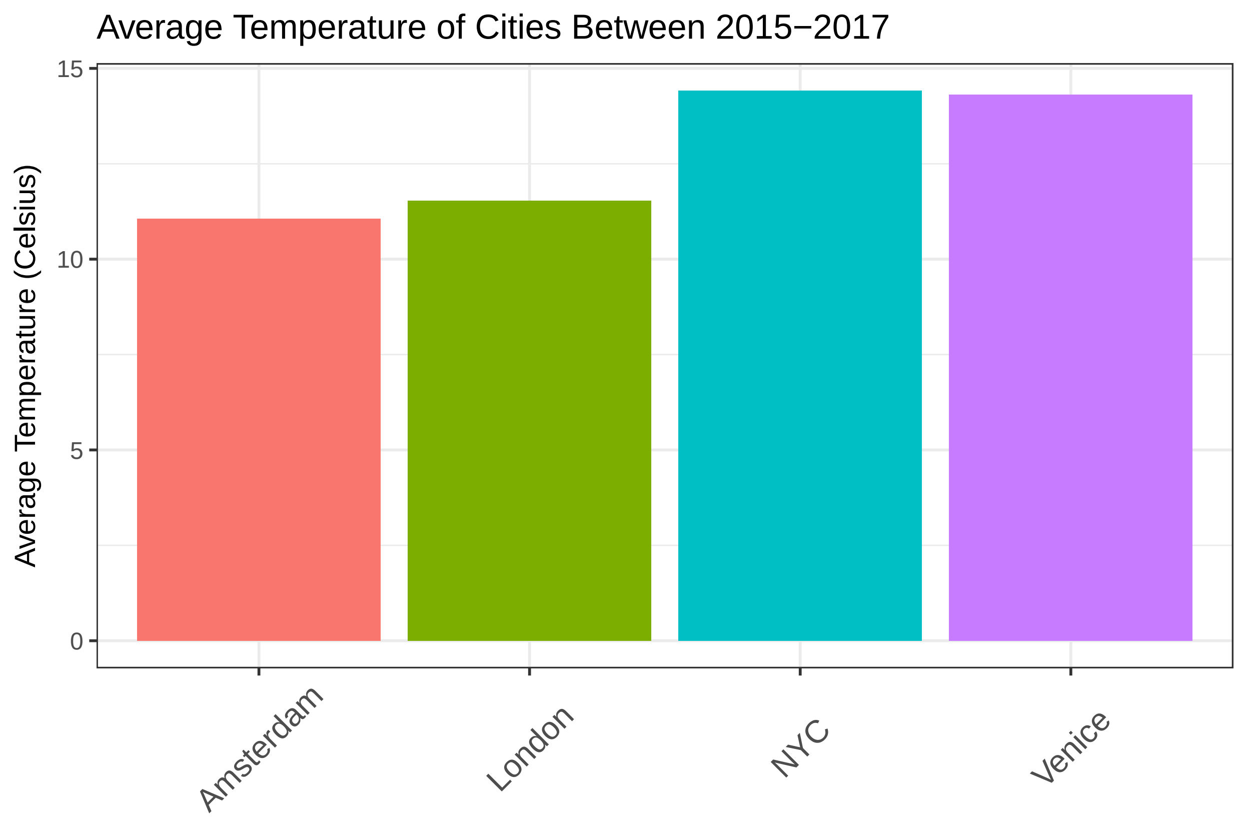
Modifications

It is possible to store ggplot2 in objects, modify the axes and other stuff easily. Continuing from the last example.

```
my_plot <- travel_weather %>% select(Amsterdam:Venice) %>% gather(key = City,
    value = Temperature) %>% group_by(City) %>% summarise(avg_temp = mean(Temperature,
    na.rm = T)) %>% ggplot(data = ., aes(x = City, y = avg_temp,
    fill = City)) + geom_bar(stat = "identity")
```

We store the whole plot in `my_plot`. Now, let's change the scene a bit.

```
my_plot + labs(x = "", y = "Average Temperature (Celsius)", title = "Average Temperature of Cities Betw
    theme_bw() + theme(legend.position = "none", axis.text.x = element_text(angle = 45,
    vjust = 0.5, hjust = 0.5, size = 12))
```



It is possible to do much more with ggplot2. It is left to your imagination and expertise.