Apuntes Tidyverse - Gapminder example

Pilar Amat Rodrigo

9/6/2018

Contents

##Data wrangling

We are gonna work with a dataset call "gapminder" during all this course and we should install first those libraries and then call them.

```
# Load the gapminder package
library(gapminder)
# Load the dplyr package
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# Load the dplyr package
library(ggplot2)
# Look at the gapminder dataset
gapminder
```

```
## # A tibble: 1,704 x 6
##    country     continent  year lifeExp      pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.
##  2 Afghanistan Asia       1957    30.3  9240934      821.
##  3 Afghanistan Asia       1962    32.0 10267083      853.
##  4 Afghanistan Asia       1967    34.0 11537966      836.
##  5 Afghanistan Asia       1972    36.1 13079460      740.
##  6 Afghanistan Asia       1977    38.4 14880372      786.
##  7 Afghanistan Asia       1982    39.9 12881816      978.
##  8 Afghanistan Asia       1987    40.8 13867957      852.
##  9 Afghanistan Asia       1992    41.7 16317921      649.
```

```
## 10 Afghanistan Asia       1997    41.8 22227415      635.
## # ... with 1,694 more rows
```

USING VERBS TO SURF DATA

filter() : To filter subsets of observations based on a condition. It needs pipes to pass the concepts %>%. Will return a new dataset not affecting the original.

```
# Extract data from 2007 and the United States.
gapminder %>%
  filter(country== 'United States', year == 2007)
```

```
## # A tibble: 1 x 6
##   country       continent  year lifeExp       pop gdpPercap
##   <fct>         <fct>     <int>   <dbl>     <int>     <dbl>
## 1 United States Americas   2007    78.2 301139947    42952.
```

arrange(): sorts a table based on a condition. It needs pipes to pass the concepts %>%. Will return a new dataset not affecting the original. Use desc(*condition*) to sort it descendent.

```
# Order the data according to gdpPercap and then population in descencing order.
gapminder %>%
  arrange(gdpPercap)
```

```
## # A tibble: 1,704 x 6
##    country          continent  year lifeExp      pop gdpPercap
##    <fct>            <fct>     <int>   <dbl>    <int>     <dbl>
##  1 Congo, Dem. Rep. Africa     2002    45.0 55379852      241.
##  2 Congo, Dem. Rep. Africa     2007    46.5 64606759      278.
##  3 Lesotho          Africa     1952    42.1   748747      299.
##  4 Guinea-Bissau    Africa     1952    32.5   580653      300.
##  5 Congo, Dem. Rep. Africa     1997    42.6 47798986      312.
##  6 Eritrea          Africa     1952    35.9  1438760      329.
##  7 Myanmar          Asia       1952    36.3 20092996      331
##  8 Lesotho          Africa     1957    45.0   813338      336.
##  9 Burundi          Africa     1952    39.0  2445618      339.
## 10 Eritrea          Africa     1957    38.0  1542611      344.
## # ... with 1,694 more rows
```

```
gapminder %>%
  arrange(desc(pop))
```

```
## # A tibble: 1,704 x 6
##    country continent  year lifeExp        pop gdpPercap
##    <fct>   <fct>     <int>   <dbl>      <int>     <dbl>
##  1 China   Asia       2007    73.0 1318683096     4959.
##  2 China   Asia       2002    72.0 1280400000     3119.
##  3 China   Asia       1997    70.4 1230075000     2289.
##  4 China   Asia       1992    68.7 1164970000     1656.
##  5 India   Asia       2007    64.7 1110396331     2452.
##  6 China   Asia       1987    67.3 1084035000     1379.
##  7 India   Asia       2002    62.9 1034172547     1747.
##  8 China   Asia       1982    65.5 1000281000      962.
##  9 India   Asia       1997    61.8  959000000     1459.
## 10 China   Asia       1977    64.0  943455000      741.
## # ... with 1,694 more rows
```

mutate() : used to change a variable or adding a new variable. It needs pipes to pass the concepts %>%.

Will return a new dataset not affecting the original.

```
# Create and modify a variable. GDP is a new variable (gdpPercap*pop)
gapminder %>%
  mutate(gdp = gdpPercap*pop)
```

```
## # A tibble: 1,704 x 7
##    country     continent  year lifeExp      pop gdpPercap          gdp
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>        <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.  6567086330.
##  2 Afghanistan Asia       1957    30.3  9240934      821.  7585448670.
##  3 Afghanistan Asia       1962    32.0 10267083      853.  8758855797.
##  4 Afghanistan Asia       1967    34.0 11537966      836.  9648014150.
##  5 Afghanistan Asia       1972    36.1 13079460      740.  9678553274.
##  6 Afghanistan Asia       1977    38.4 14880372      786. 11697659231.
##  7 Afghanistan Asia       1982    39.9 12881816      978. 12598563401.
##  8 Afghanistan Asia       1987    40.8 13867957      852. 11820990309.
##  9 Afghanistan Asia       1992    41.7 16317921      649. 10595901589.
## 10 Afghanistan Asia       1997    41.8 22227415      635. 14121995875.
## # ... with 1,694 more rows
```

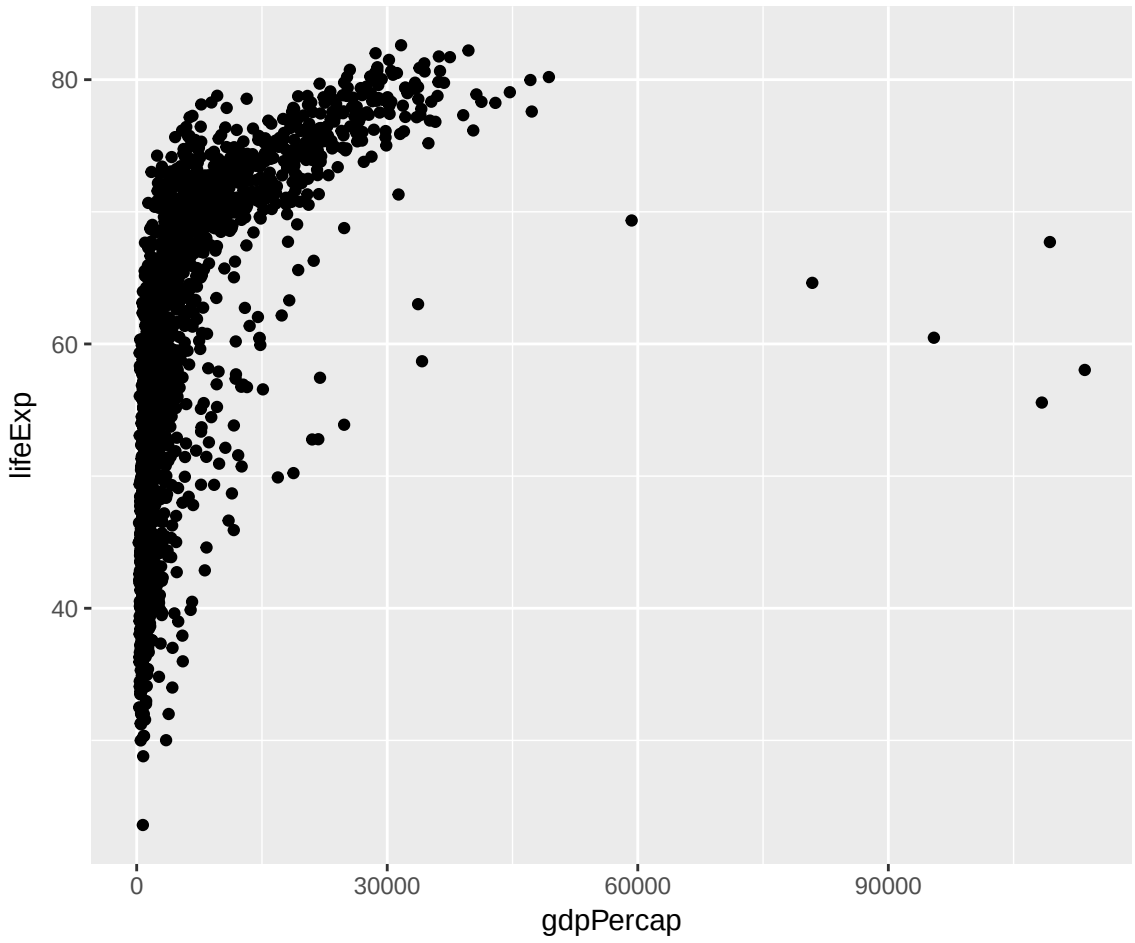
##Data visualization

We will use ggplot2 for visualization. So install the pack and then call the library

```
library(ggplot2)
```

Plotting : ggplot("dataset", aes(x="datacolumn1", y = "datacolumn2")) + typeofgraph. This last part will depend on the type of graph we need: geom_point() for scatter plots.

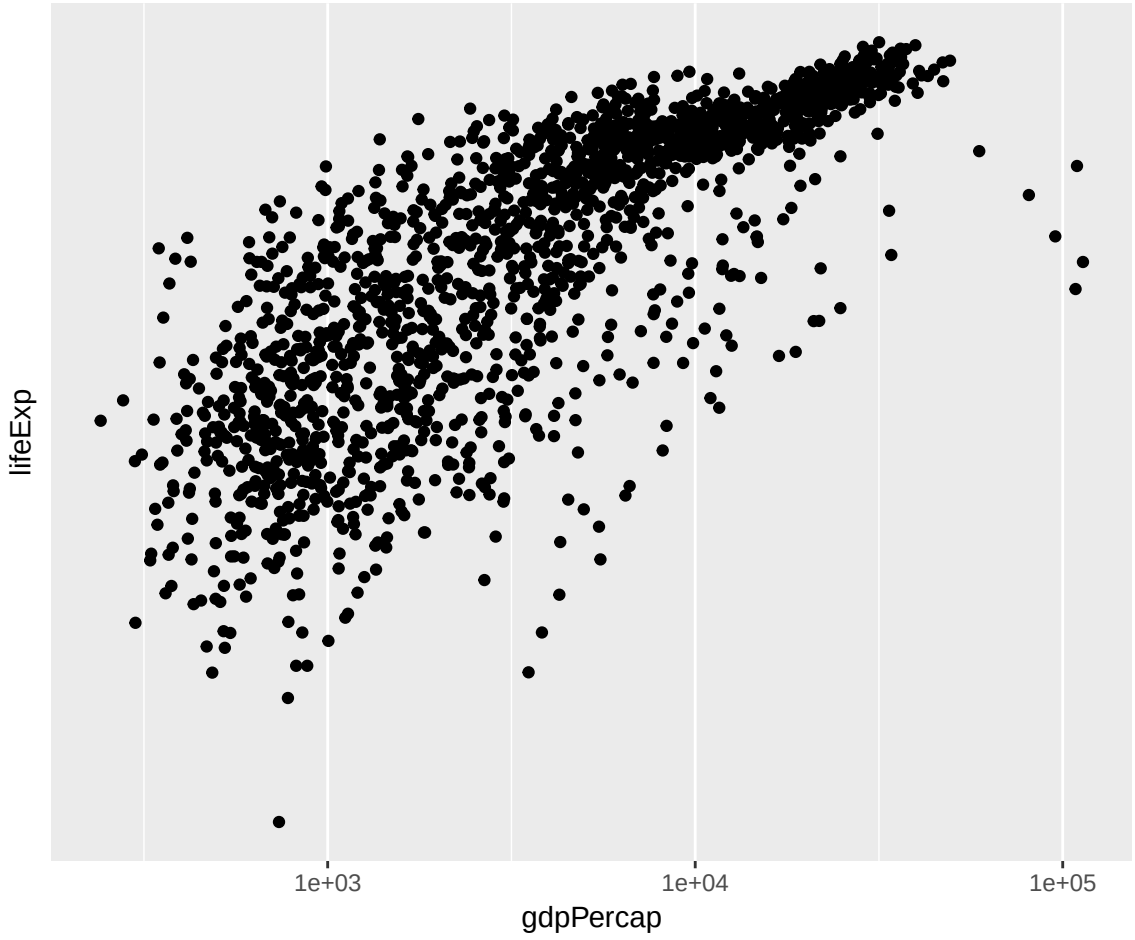
```
# Plot gapminder
ggplot(gapminder, aes(x= gdpPercap, y= lifeExp)) + geom_point()
```



When points are very close, we can use logarithmic scales help to visualize Adding a new part to the last part of the code.

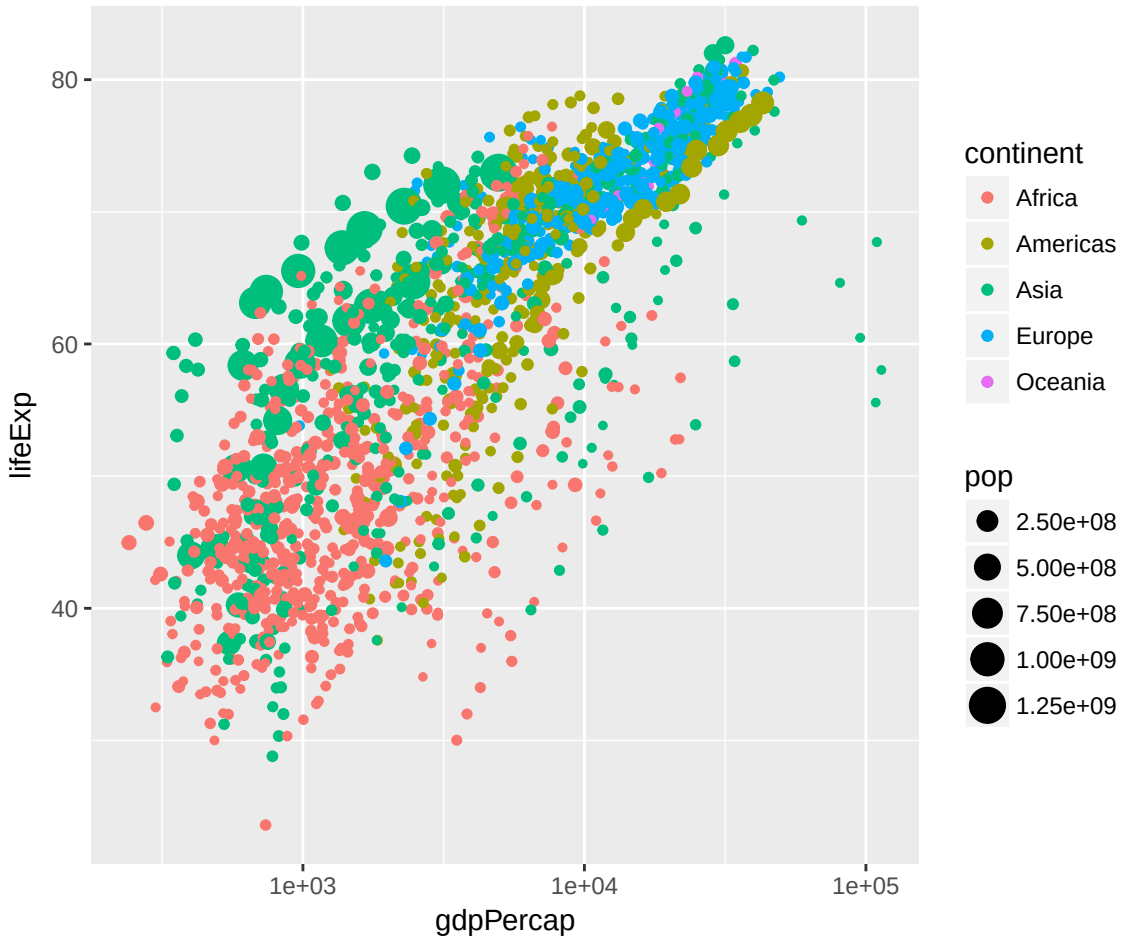
Scaling: scale_x_log10(), scale_y_log10()**

```
# Change this plot to put the x-axis, y-axis on a log scale to see the points more spread.
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp)) +
  geom_point() + scale_x_log10() + scale_y_log10()
```



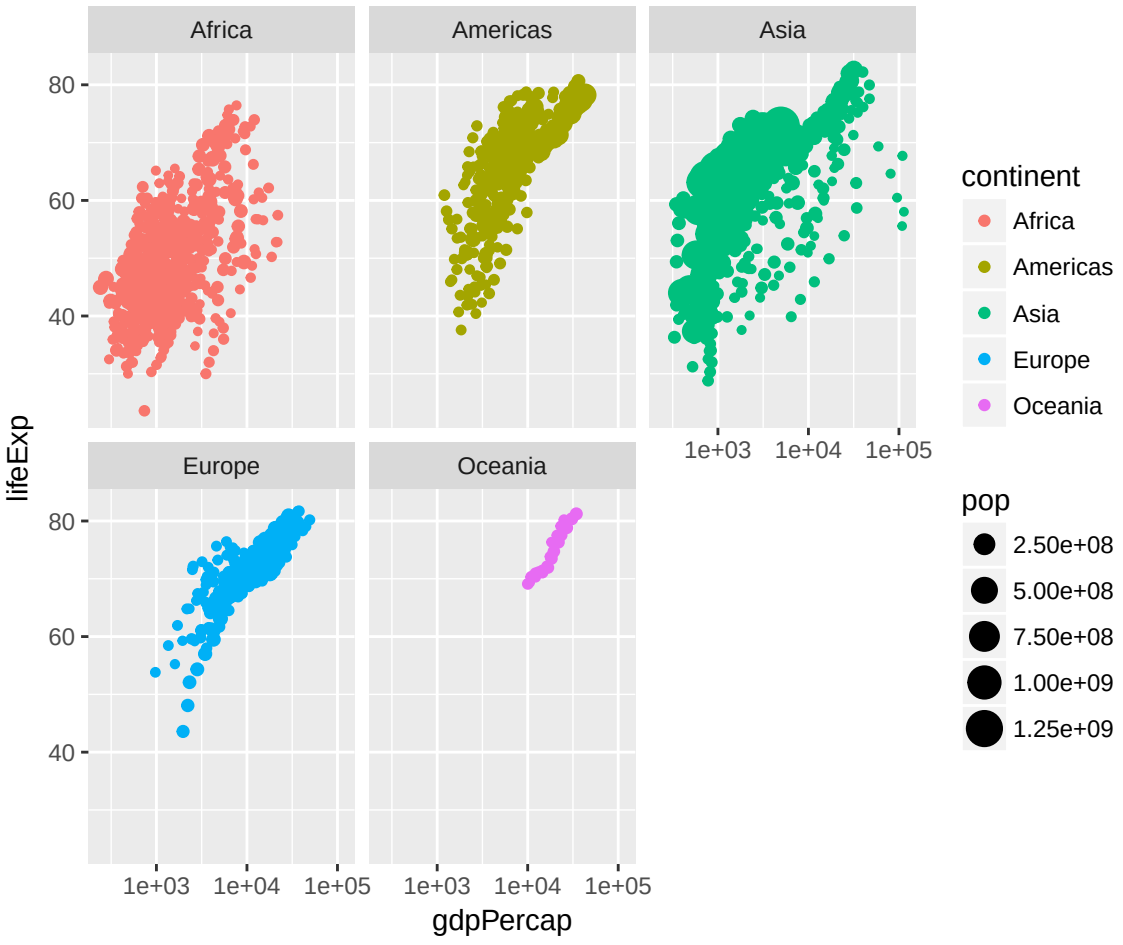
Aditional aesthetics: used mostly for categorical variables. **COLOR = categoricalvariable. SIZE = variable**

```
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp, color = continent, size = pop)) +
  geom_point() +
  scale_x_log10()
```



Faceting: showing smaller graphs to give more informartion. Use facet_wrap(~ variable)

```
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp, color = continent, size = pop)) +
  geom_point() +
  scale_x_log10()+
  facet_wrap(~ continent)
```



##Grouping and summarizing

Summarize(): getting a table of results in a new table calling for functions such as mean, sum, median, min, max... normaly the output it's a number/ answer.

```
# Summarize the average lifeExp

gapminder %>%
summarize(meanLifeExp= mean(lifeExp))
```

```
## # A tibble: 1 x 1
##   meanLifeExp
##         <dbl>
## 1        59.5
```

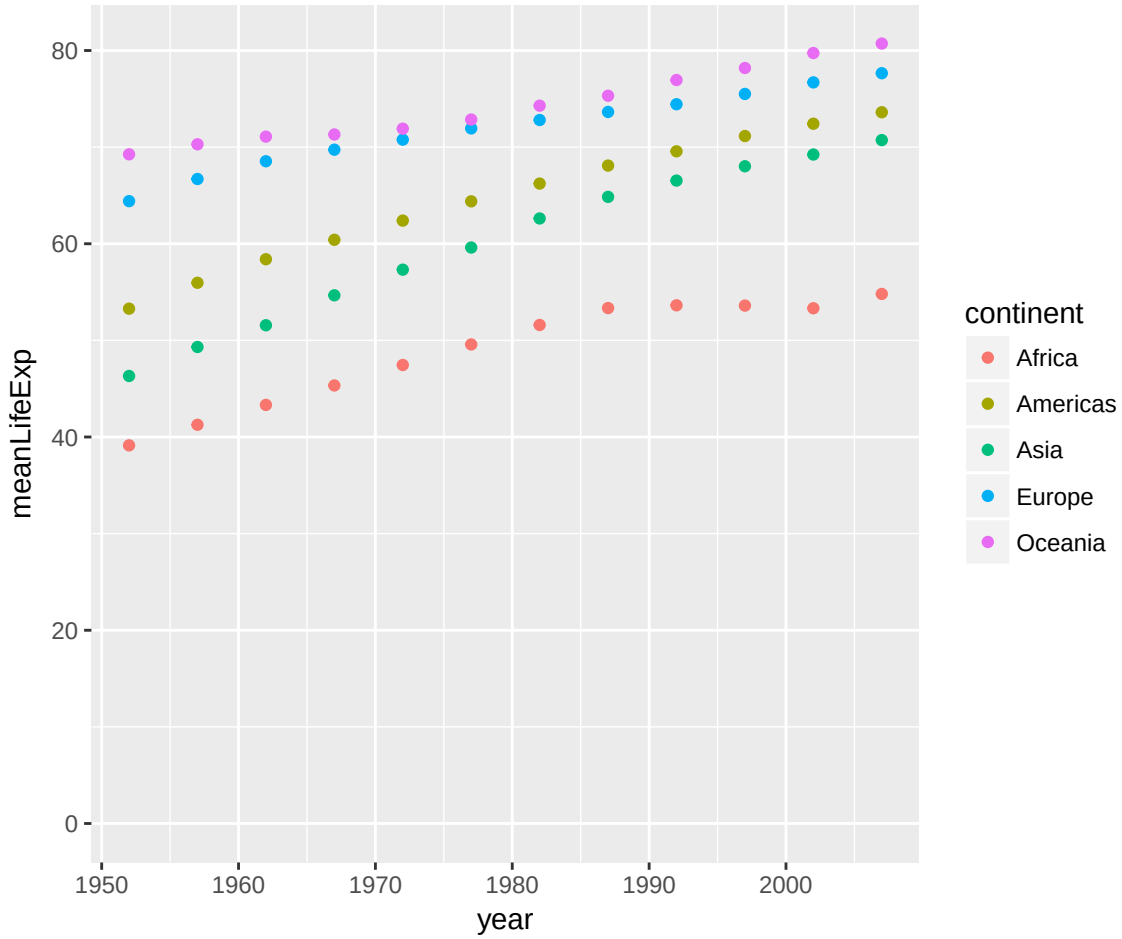
group_by(): before summarize() turns groups into one row each. Giving several arguments nestes the groups.

```
by_year_continent<-gapminder %>%
  group_by(year,continent)%>%
  summarize(meanLifeExp=mean(lifeExp),maxGDP=max(gdpPercap))
```

##Types of visualizations

expand_limits(y=0) -> to see gride from point 0.

```
ggplot(by_year_continent, aes(x=year, y= meanLifeExp, color= continent)) +
  geom_point()+
  expand_limits(y=0)
```



Type of charts Line Plot : geom_line()

Bar Plot : geom_col()

Histogram : geom_histogram()

Box Plot : geom_boxplot()

Title : ggtitle("xxxx")