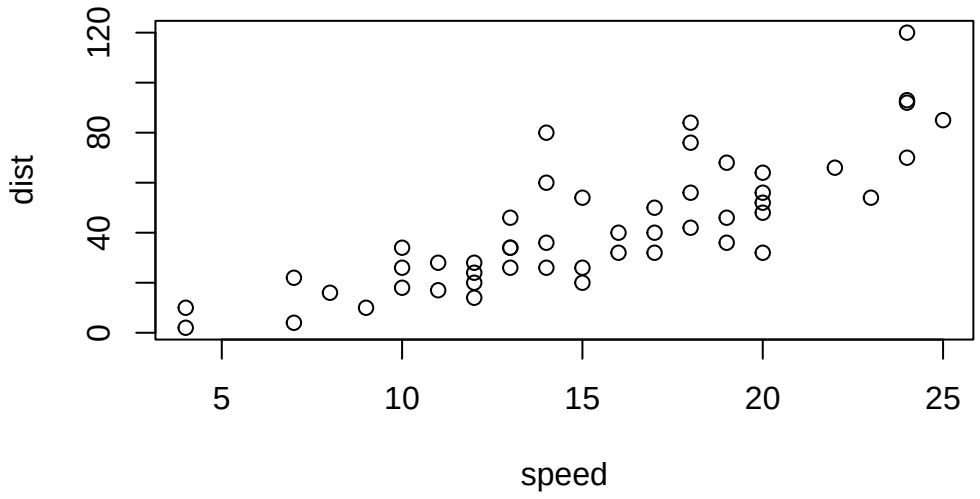
Class 5: Data Visualization with ggplot

Shreyas Sankaranarayanan (PID: A17077452)

Today we will have our first play with the **ggplot2** package - one of the most popular graphics packages on the planet.

There are many plotting systems in R. These include so-called *"base"* plotting/graphics.

```
plot(cars)
```



Base plot is generally rather short code and somewhat dull plots - but it is always there for you and is fast for big datasets.

If I want to use **ggplot2** it takes some more work.

```
#ggplot(cars)
```

I need to install the package first to my computer. To do this I can use the function `install.packages("ggplot2")`

Every time I want to use a package I need to load it up with a `library()` call.

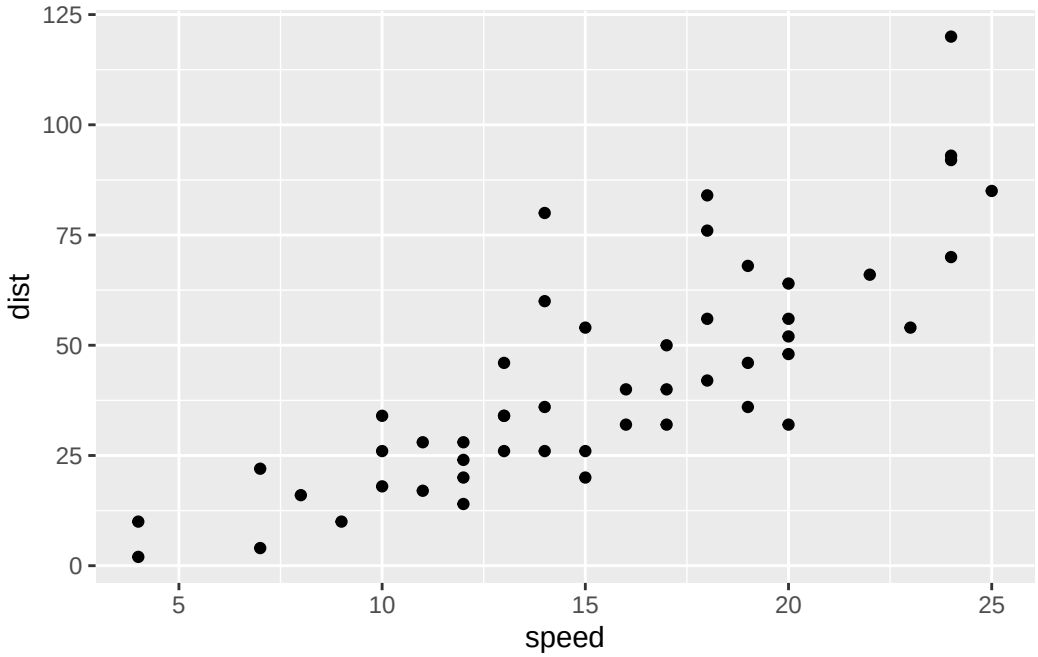
```
#Make sure to run in console: install.packages("ggplot2")
library(ggplot2)
ggplot(cars)
```


Every ggplot has at least 3 things:

- **data** (the data.frame that you want to plot)
- **aes** (the aesthetic mapping of the data to the plot)
- **geom** (how do you want the plot to look, points, lines, etc.)

Let us create a barebones scatterplot of the data

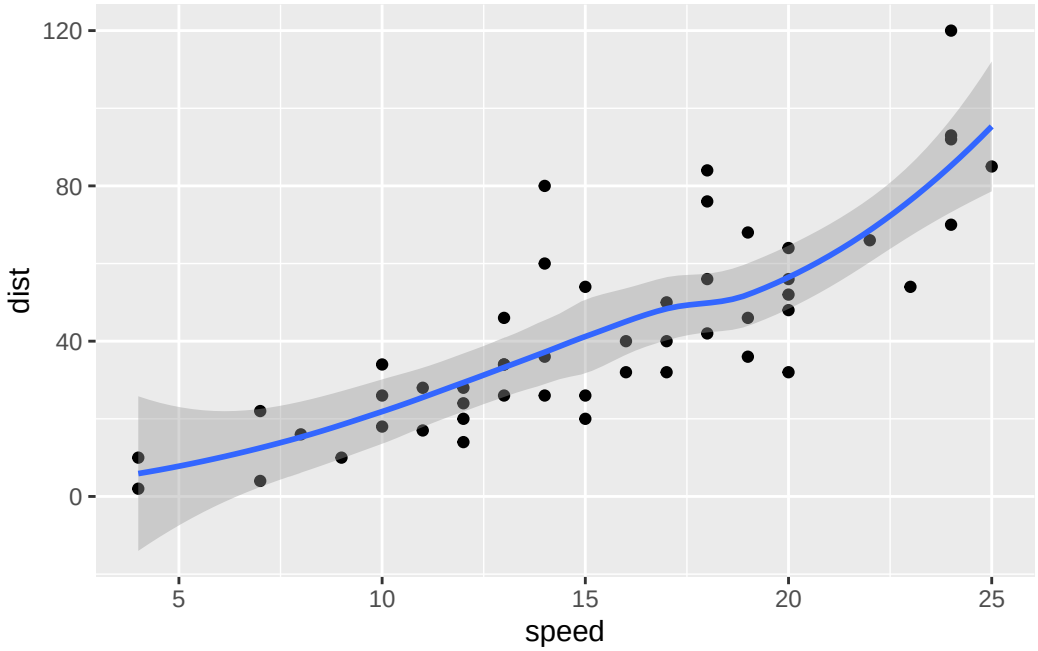
```
ggplot(data = cars, mapping = aes(speed, dist)) + geom_point()
```



Let us add an extra geom parameter:

```
ggplot(data = cars, mapping = aes(speed, dist)) + geom_point() + geom_smooth()
```

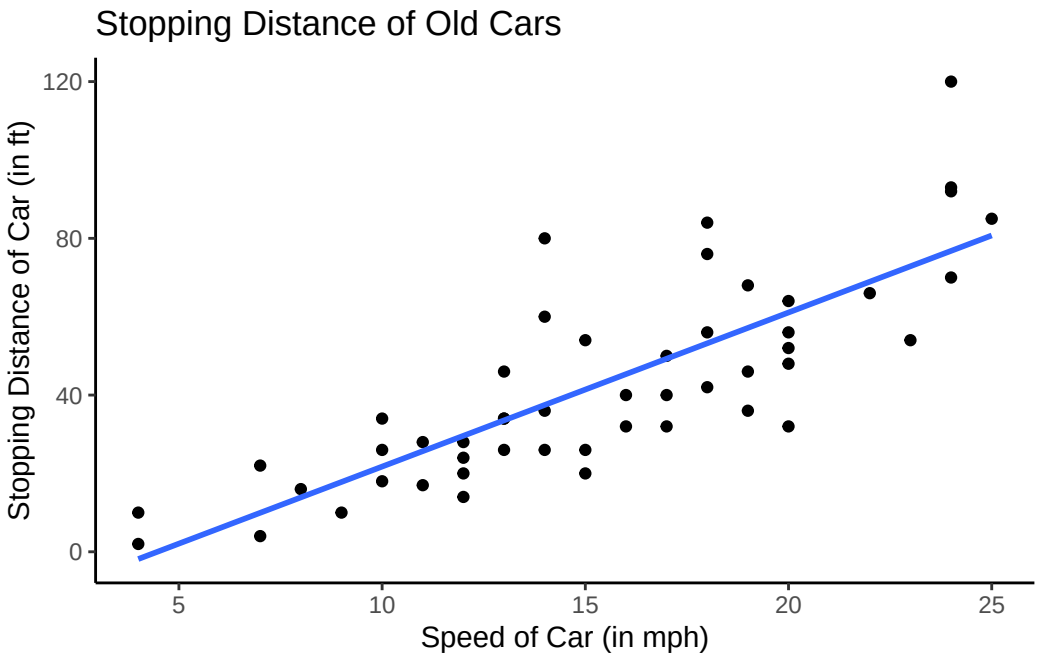
```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



I want a linear model and no standard error bounds shown on my plot. I also want nicer axi labels and a title etc.

```
ggplot(data = cars, mapping = aes(speed, dist)) + geom_point() + geom_smooth(method = "lm"
```

```
`geom_smooth()` using formula = 'y ~ x'
```



A mroe complicated scatterplot

Here we make a plot of gene expression data:

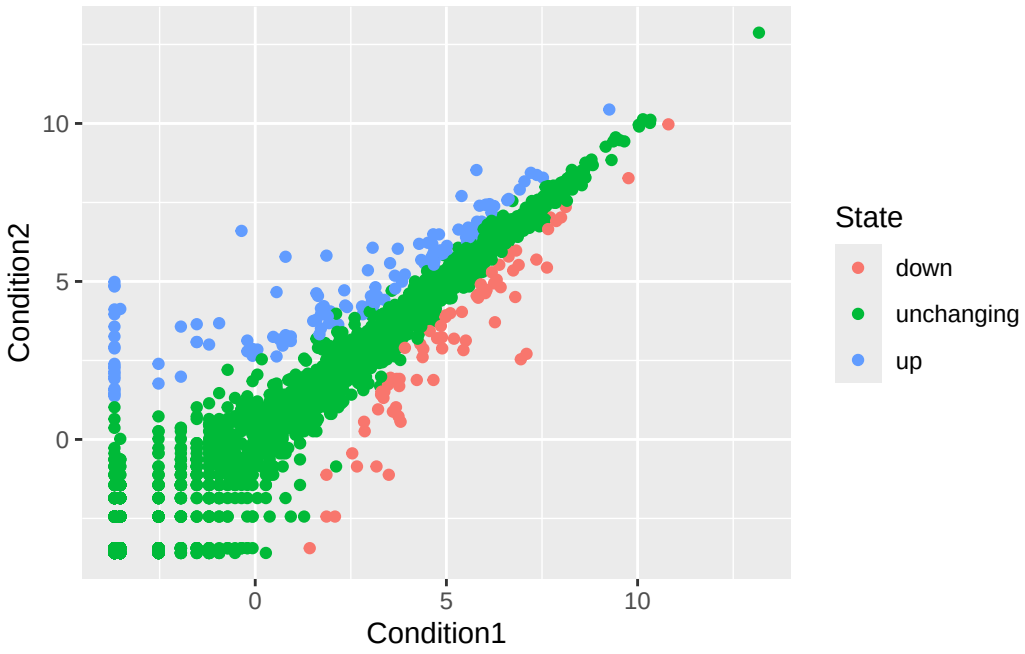
```
url <- "https://bioboot.github.io/bimm143_S20/class-material/up_down_expression.txt"
genes <- read.delim(url)
head(genes);
```

```
        Gene Condition1 Condition2      State
1      A4GNT -3.6808610 -3.4401355 unchanging
2       AAAS  4.5479580  4.3864126 unchanging
3      AASDH  3.7190695  3.4787276 unchanging
4       AATF  5.0784720  5.0151916 unchanging
5       AATK  0.4711421  0.5598642 unchanging
6 AB015752.4 -3.6808610 -3.5921390 unchanging
```

Here is a scatterplot plot of condition 1 vs condition 2 which highlights which genes are upregulated (red) or downregulated (blue).

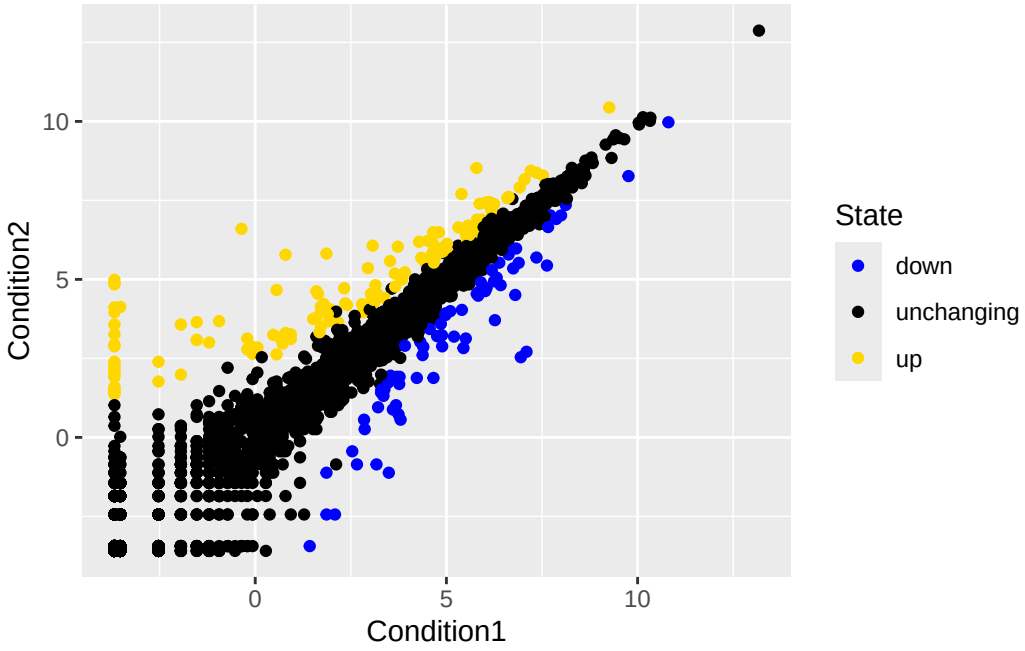
```
p <- ggplot(data = genes, mapping = aes(Condition1, Condition2, col = State)) + geom_point
```

```
p
```



To modfiy the color scheme the function `scale_color_manual()` can be used in this case we want a UCSD themed color scheme (blue, gold, and black). So to implement this you do the following:

```
p + scale_color_manual(values = c("blue", "black", "gold"))
```



Exploring the gapminder dataset

Let us first read the dataset from online:

```
url <- "https://raw.githubusercontent.com/jennybc/gapminder/master/inst/extdata/gapminder.
gapminder <- read.delim(url)
```

What number of rows?

```
nrow(gapminder)
```

```
[1] 1704
```

What number of columns?

```
ncol(gapminder)
```

```
[1] 6
```

Let us analyze the contents of this dataset: A table of all the years in the dataset

```
table(gapminder$year)
```

```
1952 1957 1962 1967 1972 1977 1982 1987 1992 1997 2002 2007
 142  142  142  142  142  142  142  142  142  142  142  142
```

Table of the different continents represented in the dataset

```
table(gapminder$continent)
```

```
  Africa Americas     Asia   Europe  Oceania
     624      300      396      360       24
```

Unique number of continents

```
length(unique(gapminder$continent))
```

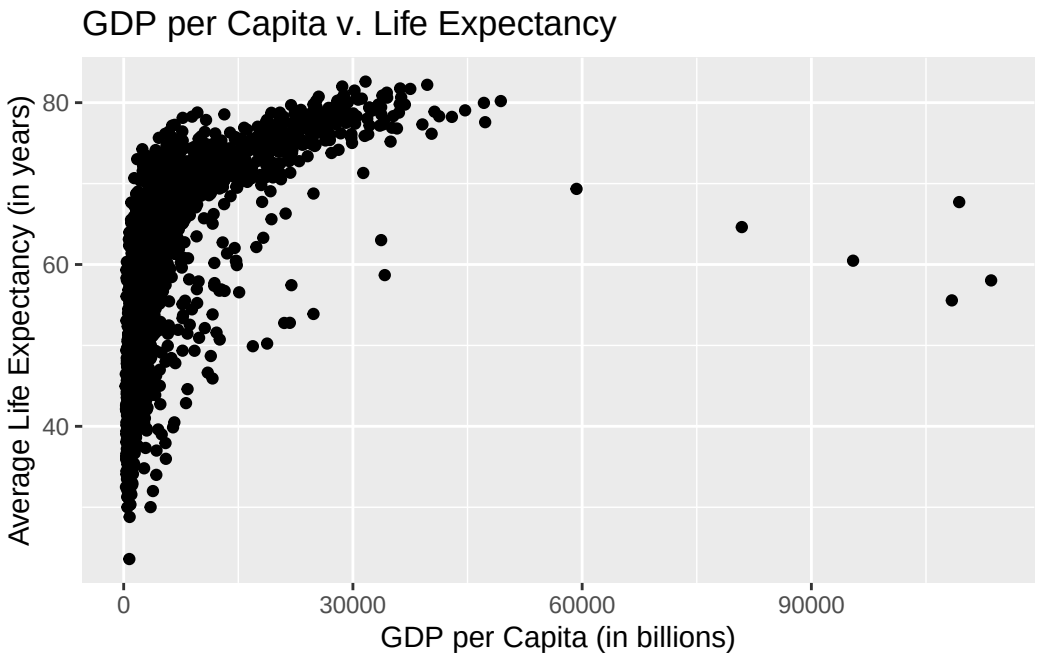
```
[1] 5
```

Unique number of countries

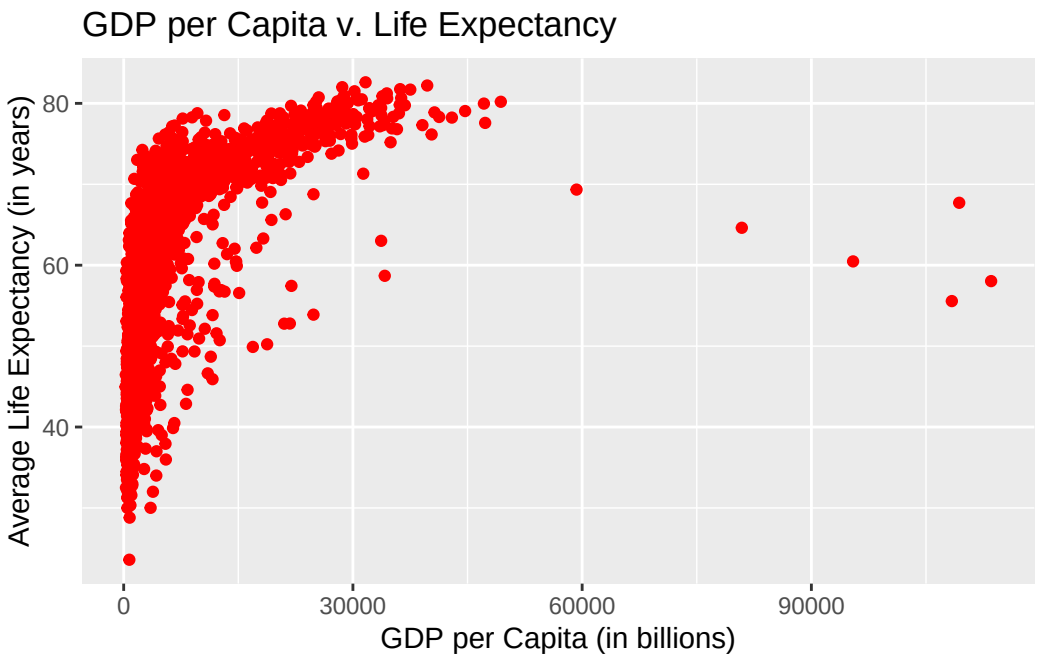
```
length(unique(gapminder$countries))
```

```
[1] 0
```

```
ggplot(data = gapminder, mapping = aes(gdpPercap, lifeExp)) + geom_point() + labs(title =
```




```
ggplot(data = gapminder, mapping = aes(gdpPercap, lifeExp)) + geom_point(col = "red") + la
```



##Using dplyr package to isolate and analyze gapminder data from 2007 We will then utilize the **dplyr** package to analyze this data set by single year. If need be run `install.packages("dplyr")`:

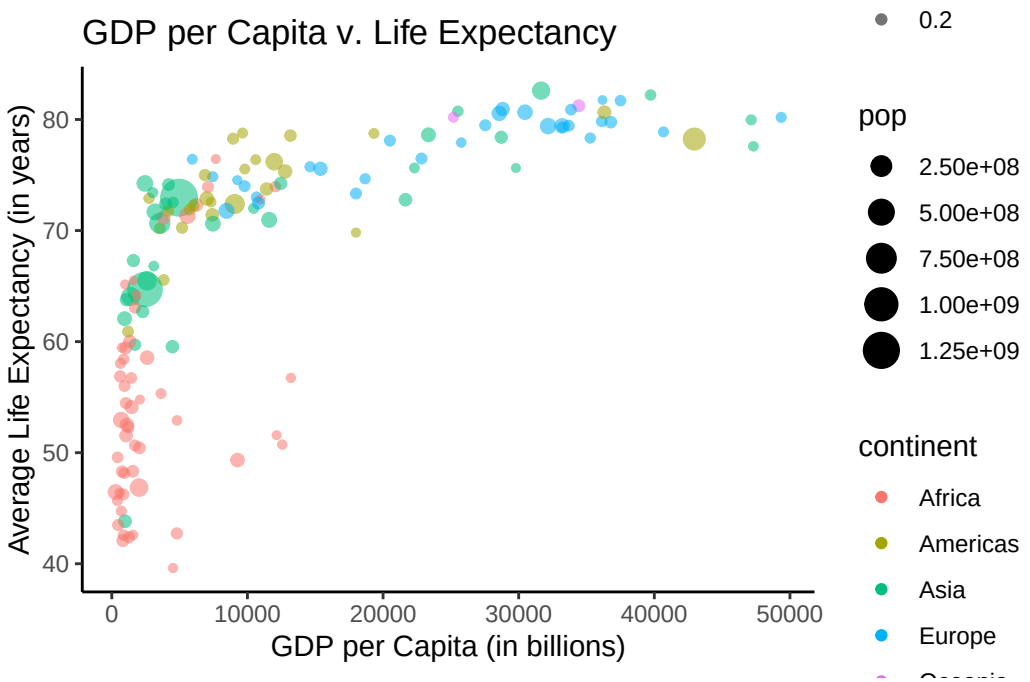
```
#Make sure to run in console: install.packages("dplyr")
library(dplyr)
```

```
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
```

```
gapminder_2007 <- gapminder %>% filter(year==2007)
```

Lets create a plot for life expectancy in 2007 based on GDP per capita:

```
ggplot(data = gapminder_2007, mapping = aes(gdpPercap, lifeExp, col = continent, size = po
```

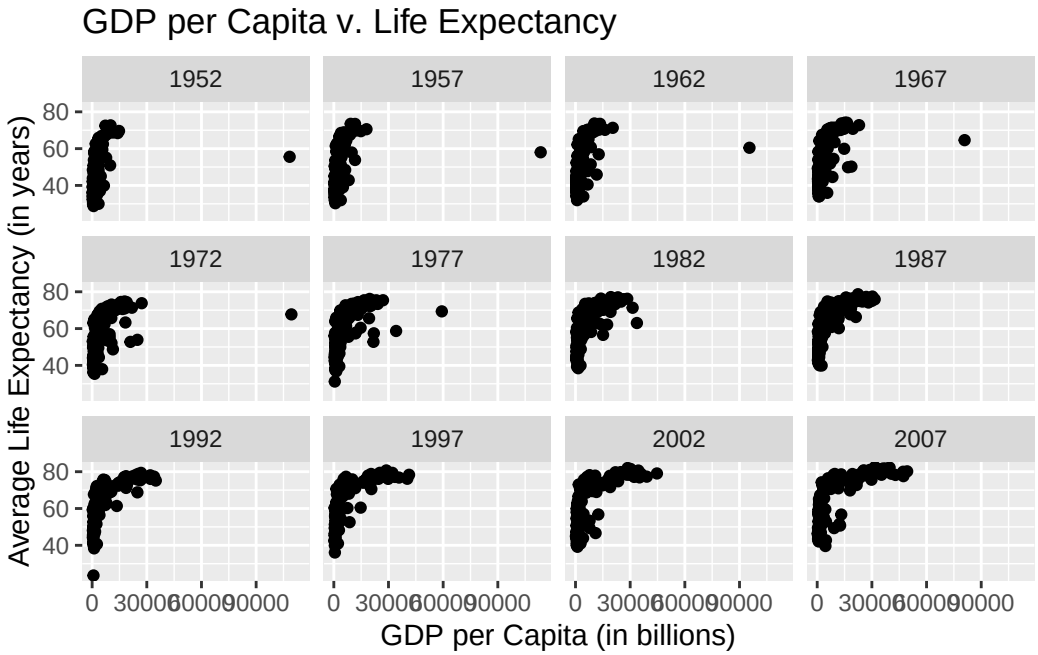


Introducing `facet_wrap`

The facet_wrap function merely produces different graphs using the unique entries in a column of a data frame as a qualitative variable and creates different plots based on such.

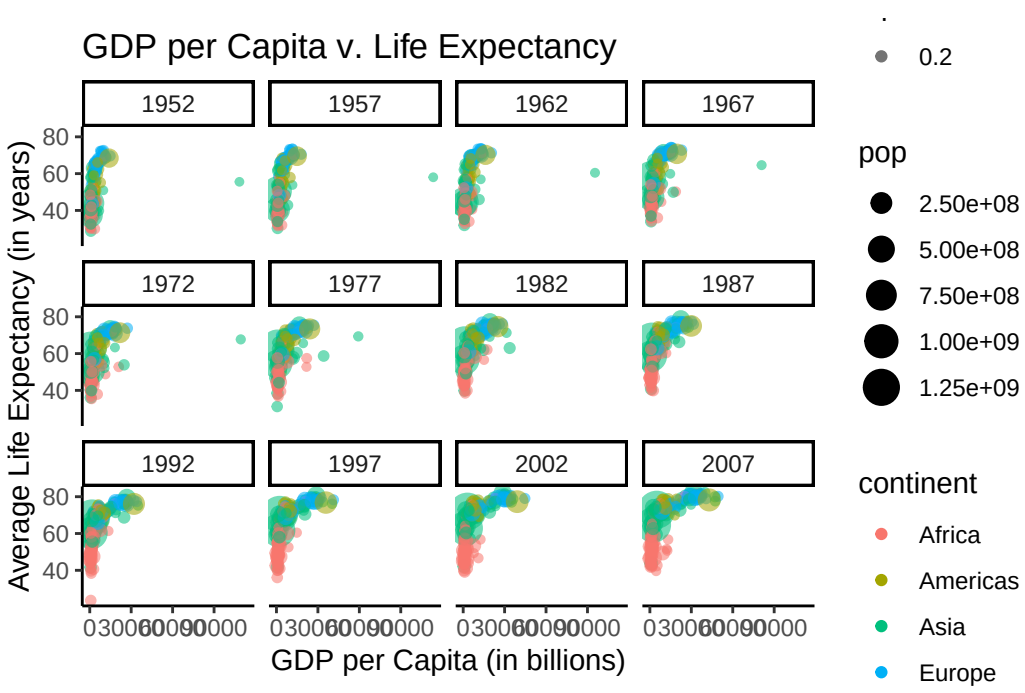
The graph below uses year as a classifier to create different graphs from:

```
ggplot(data = gapminder, mapping = aes(gdpPercap, lifeExp)) + geom_point() + labs(title =
```



The plot below are the same graphs but with the extra aesthetic elements:

```
ggplot(data = gapminder, mapping = aes(gdpPercap, lifeExp, col = continent, size = pop)) +
```


GDP per Capita v. Life Expectancy
1952
1957
1962
1967
1972
1977
1982
1987
1992
1997
2002
2007
Average Life Expectancy (in years)
80
60
40
0
30000
60000
90000
GDP per Capita (in billions)
.
0.2
pop
2.50e+08
5.00e+08
7.50e+08
1.00e+09
1.25e+09
continent
Africa
Americas
Asia
Europe