Class 5: Data Visualization with GGPLOT

Nicholas Yousefi

Our first ggplot

To use the ggplot2 package I first need to have it installed on my computer.

To install any package we use the `install.packages()` command.

(I already did this when I saw the videos before this class, but it is commented out in the code below).

Now, can I use it???

No! You first must run `library(ggplot2)` every time you want to use it. (It is sort of like import statements in Python; you must run it every time you want to use ggplot).

```
# install.packages("ggplot2")
library(ggplot2)
ggplot() # this function alone creates a blank canvas
```

We will be using the following built-in data to make our graphs:

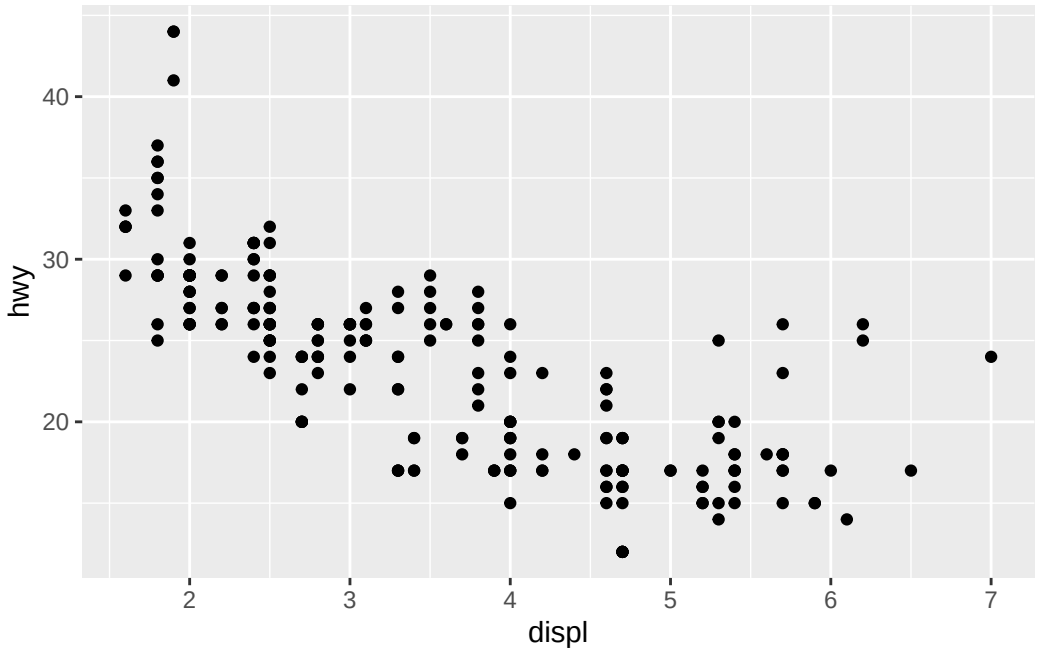
```
mpg
```

```
# A tibble: 234 x 11
   manufacturer model      displ  year   cyl trans drv     cty   hwy fl    class
   <chr>        <chr>      <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
 1 audi         a4           1.8  1999     4 auto~ f        18    29 p     comp~
 2 audi         a4           1.8  1999     4 manu~ f        21    29 p     comp~
 3 audi         a4           2    2008     4 manu~ f        20    31 p     comp~
 4 audi         a4           2    2008     4 auto~ f        21    30 p     comp~
 5 audi         a4           2.8  1999     6 auto~ f        16    26 p     comp~
 6 audi         a4           2.8  1999     6 manu~ f        18    26 p     comp~
 7 audi         a4           3.1  2008     6 auto~ f        18    27 p     comp~
 8 audi         a4 quattro   1.8  1999     4 manu~ 4        18    26 p     comp~
 9 audi         a4 quattro   1.8  1999     4 auto~ 4        16    25 p     comp~
10 audi         a4 quattro   2    2008     4 manu~ 4        20    28 p     comp~
# ... with 224 more rows
```

Let's plot displacement (displ) vs. highway miles per gallon (hwy):

All `ggplot()` graphs are made in the same way: - data + aes + geoms

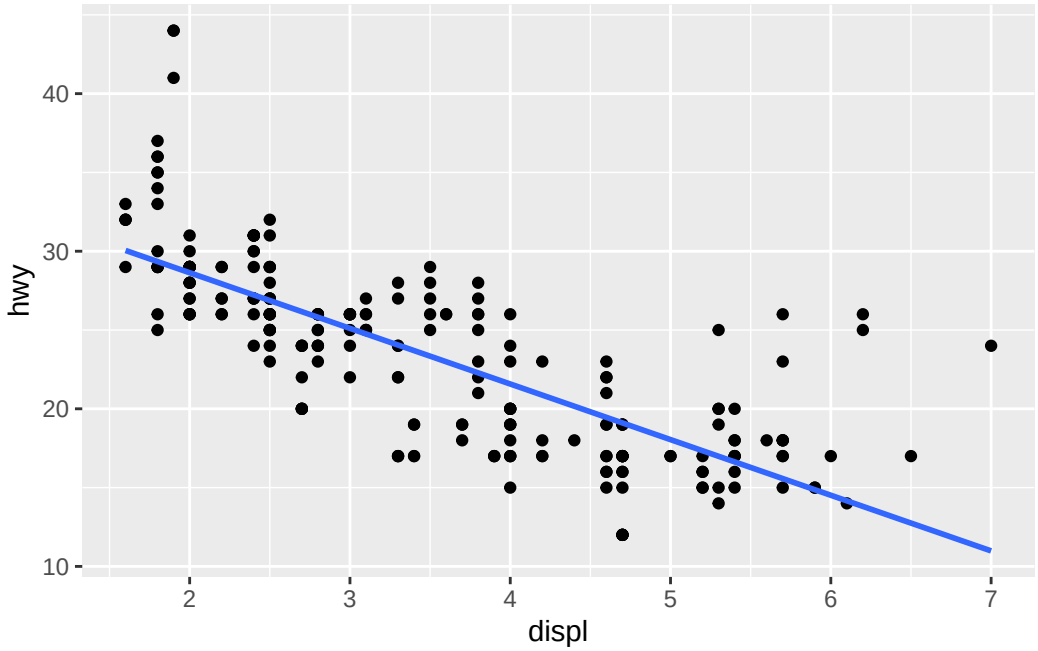
```
ggplot(data=mpg) +
  aes(x=displ, y=hwy) +
  geom_point()
```



I can add more layers:

```
ggplot(data=mpg) +
  aes(x=displ, y=hwy) +
  geom_point() +
  geom_smooth(se=FALSE, method=lm)
```

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



Plot of gene expression data

First read the data from online.

```
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
```

Q. How many genes are in this dataset?

```
nrow(genes)
```

```
[1] 5196
```

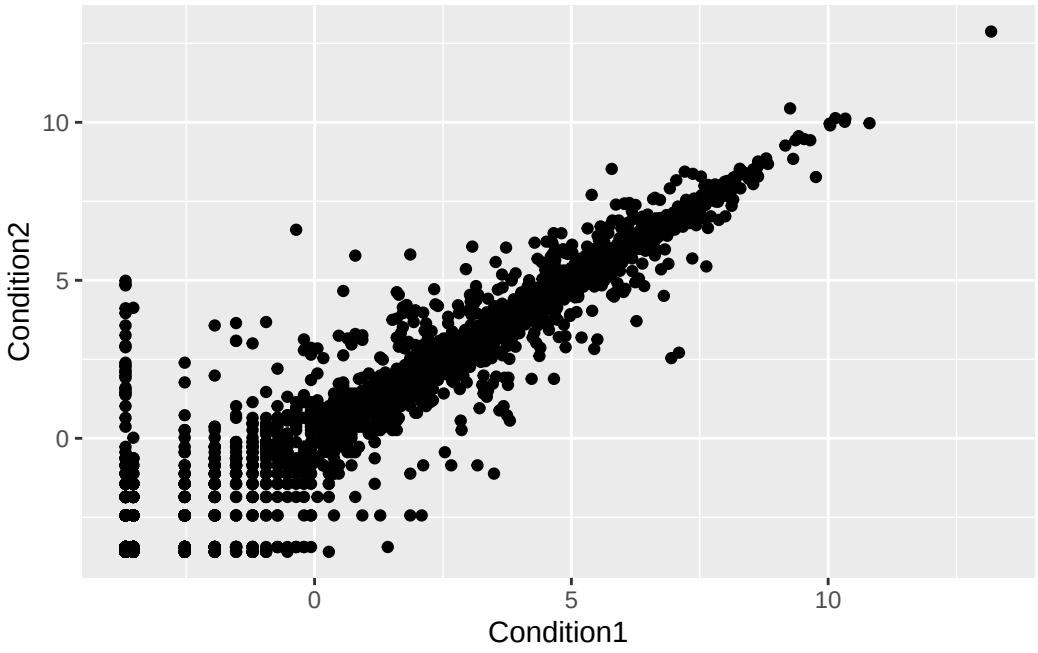
What are the colnames?

```
colnames(genes)
```

```
[1] "Gene"       "Condition1" "Condition2" "State"
```

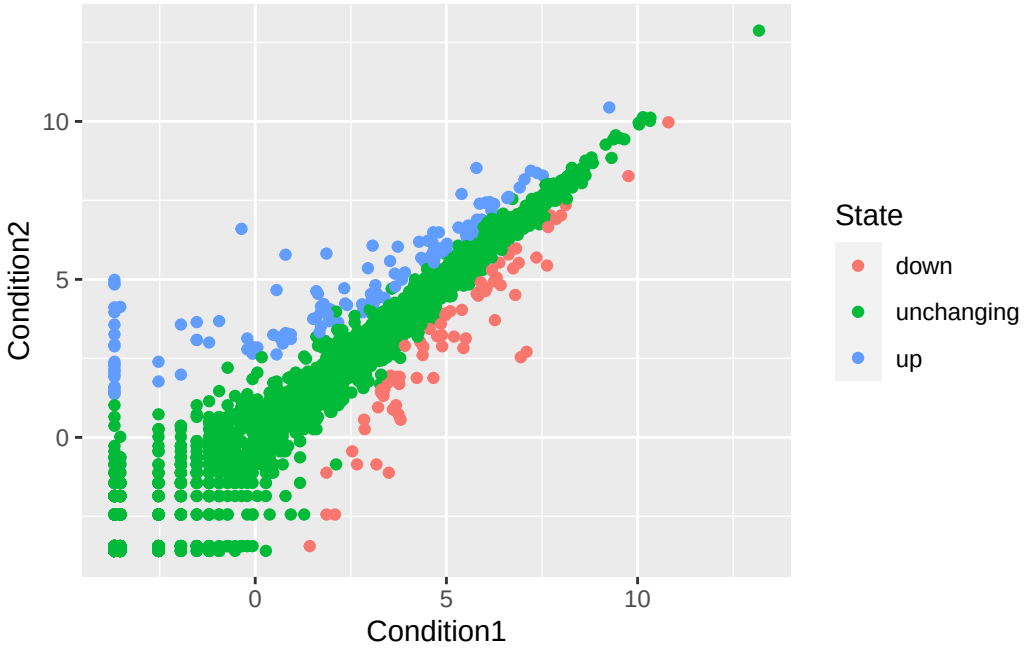
A first version plot of this data Condition1 vs. Condition2:

```
ggplot(data=genes) +
  aes(x=Condition1, y=Condition2) +
  geom_point()
```



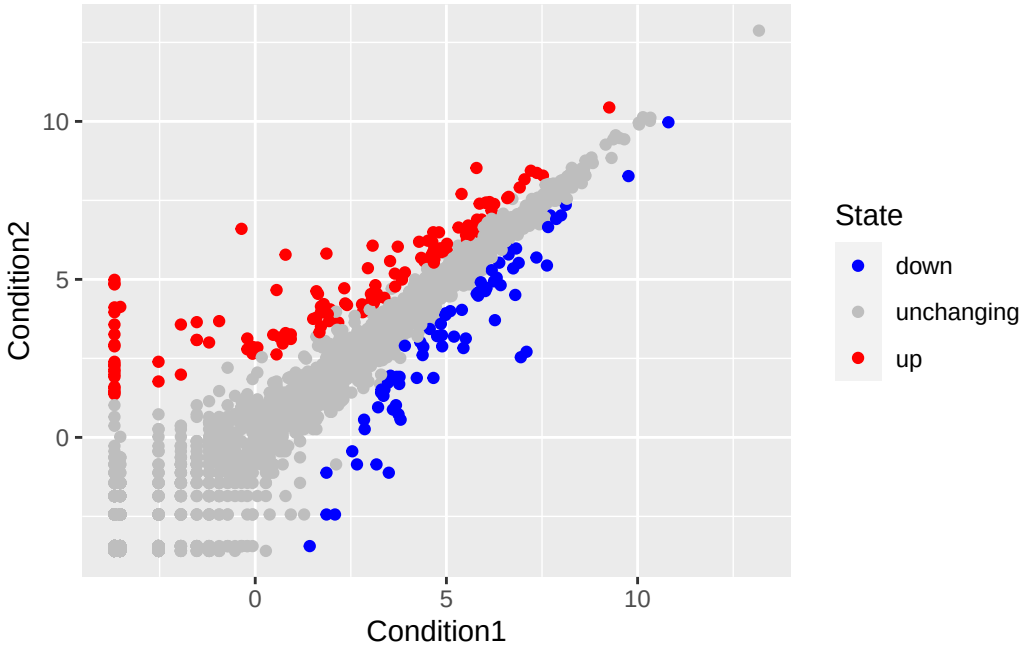
Adding Color:

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



Changing the color:

```
p <- p + scale_color_manual(values=c("blue", "gray", "red"))
p
```



Q. How many genes are up regulated and down regulated?

```
sum(genes$State == "up") # Number of up-regulated genes
```

```
[1] 127
```

```
sum(genes$State == "down") # Number of down-regulated genes
```

```
[1] 72
```

```
table(genes$State) # A faster way to tell you # of up/down regulated genes
```

```
      down unchanging         up
        72       4997        127
```

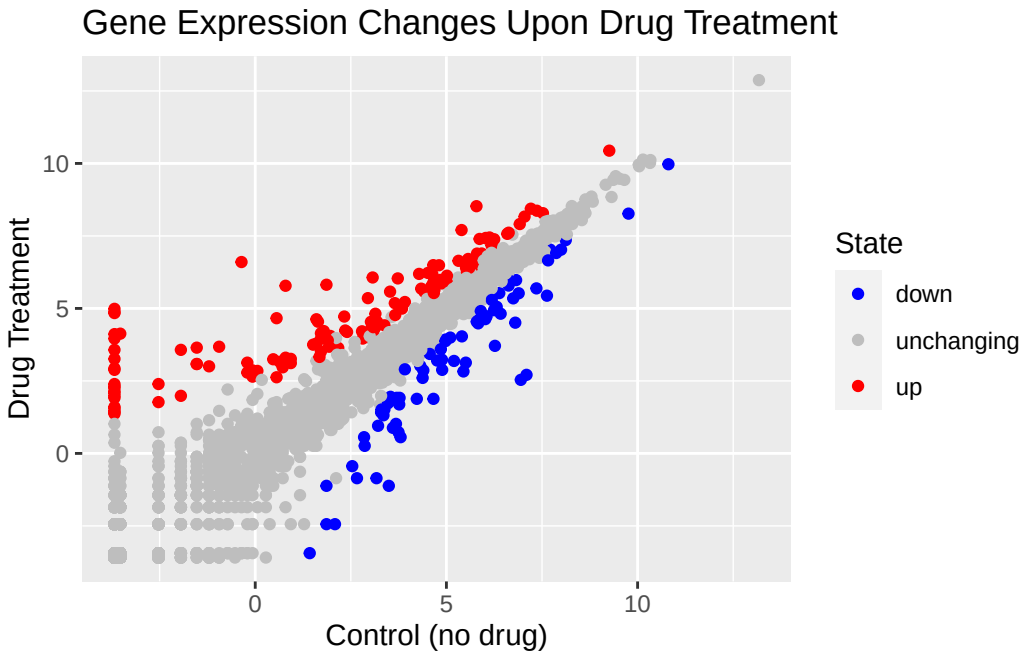
The Table function counts up the number of appearances of each item in a vector:

```
table(c("bimm143", "help", "me", "bimm143"))
```

```
bimm143    help      me
      2       1       1
```

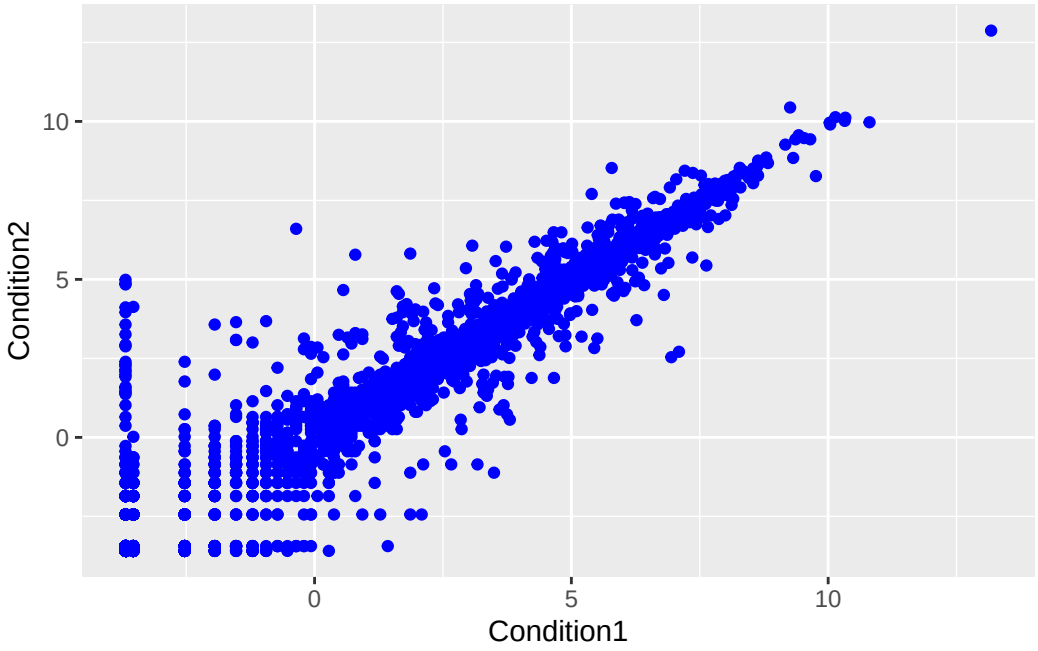
Now, let's make the graph have a nice title, axes, etc.

```
p <- p + labs(title="Gene Expression Changes Upon Drug Treatment",
         x="Control (no drug)",
         y="Drug Treatment",
         )
p
```



What if you add argument to `geom_plot(col="blue")`?

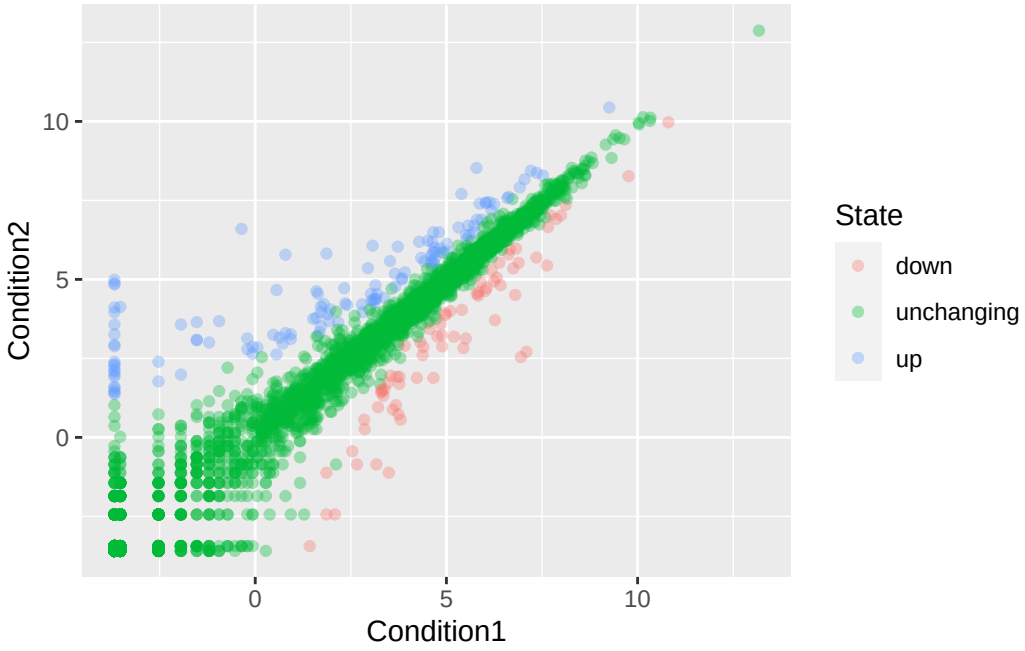
```
ggplot(data=genes) +
  aes(x=Condition1, y=Condition2, col=State) +
  geom_point(col="blue")
```



That's not what we want! Whatever you do in the `geom` layer is applied to every point equally.

You can also set the transparency by putting arguments into `geom_point()`.

```
ggplot(data=genes) +
  aes(x=Condition1, y=Condition2, col=State) +
  geom_point(alpha=0.35)
```


10
5
0
Condition2
0
5
10
Condition1
State
down
unchanging
up