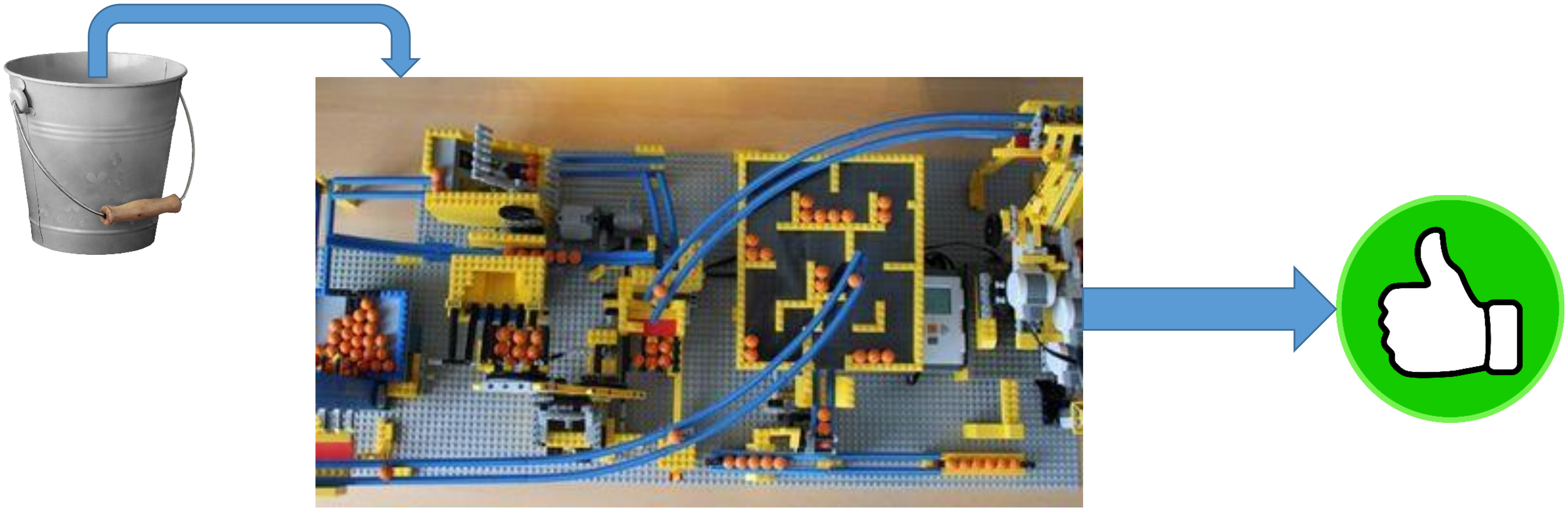
Introduction to Data Analysis and Cleaning

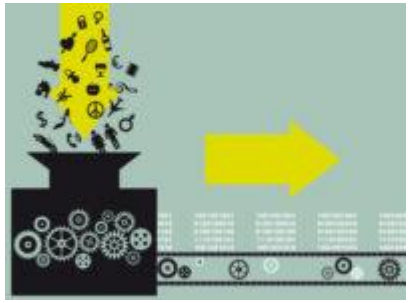
Mark Bell, Digital Researcher at The National Archives

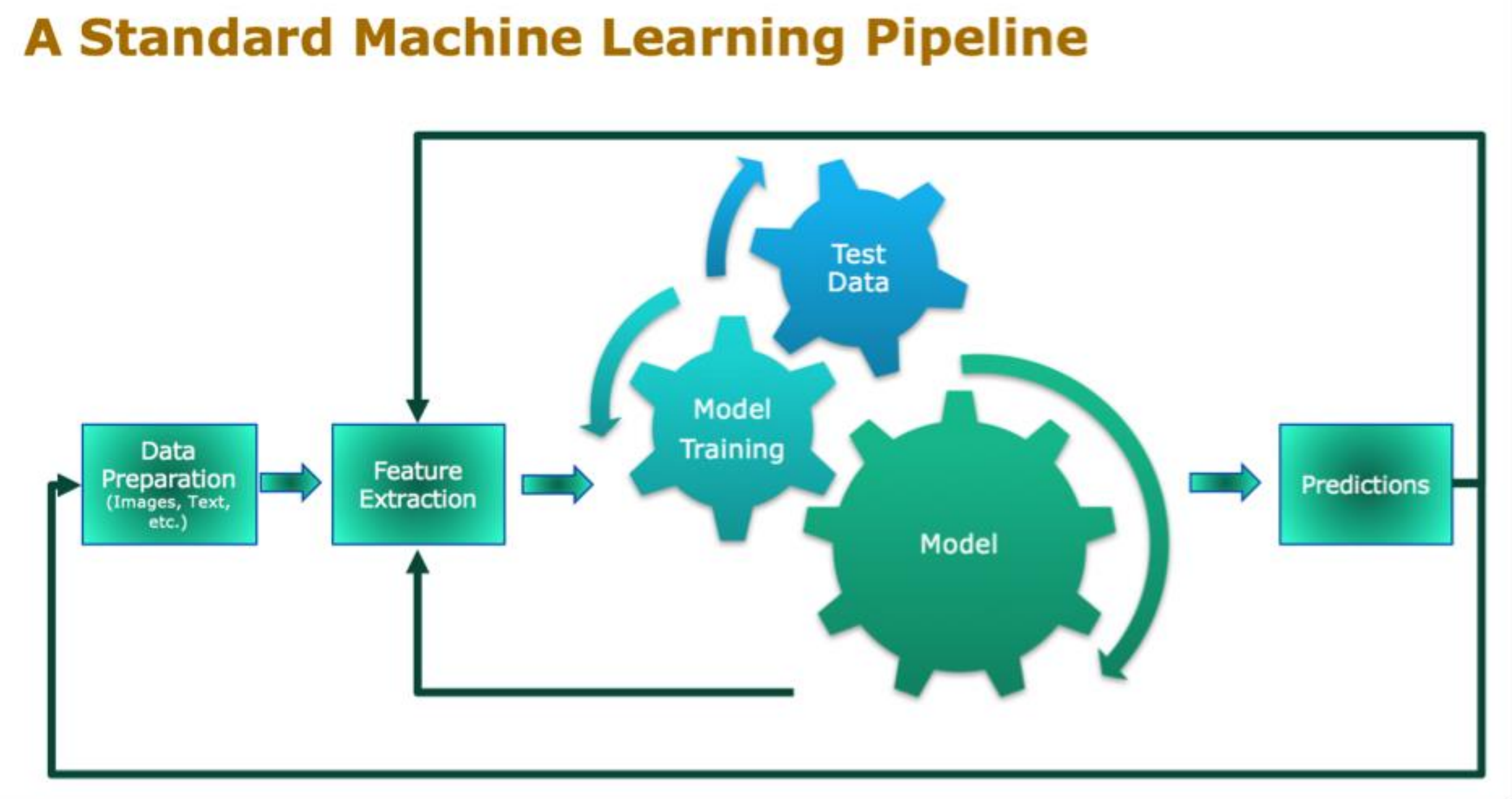
Why is this important?

The perception of machine learning



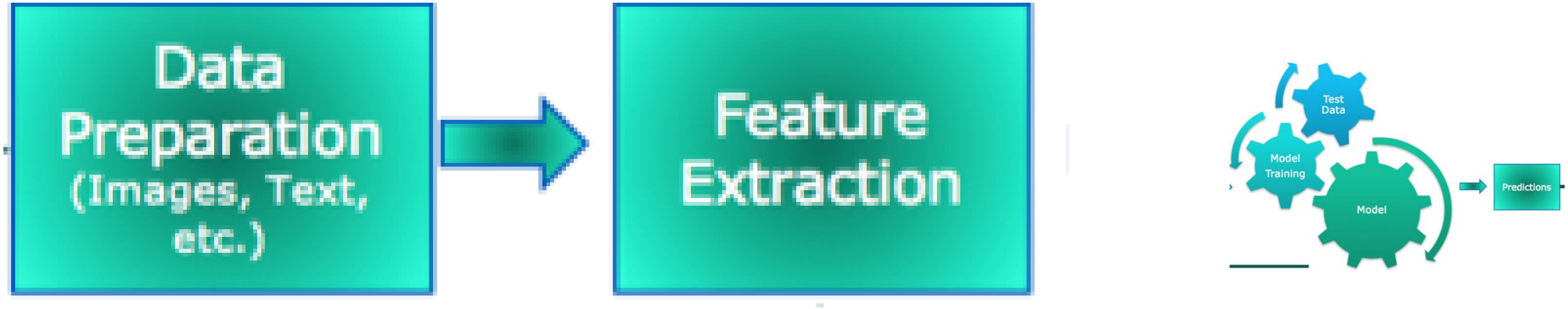
The machine learning pipeline




A Standard Machine Learning Pipeline
Data Preparation (Images, Text, etc.)
Feature Extraction
Model Training
Test Data
Model
Predictions

https://www.datanami.com/2018/09/05/how-to-build-a-better-machine-learning-pipeline/

Pipeline Effort

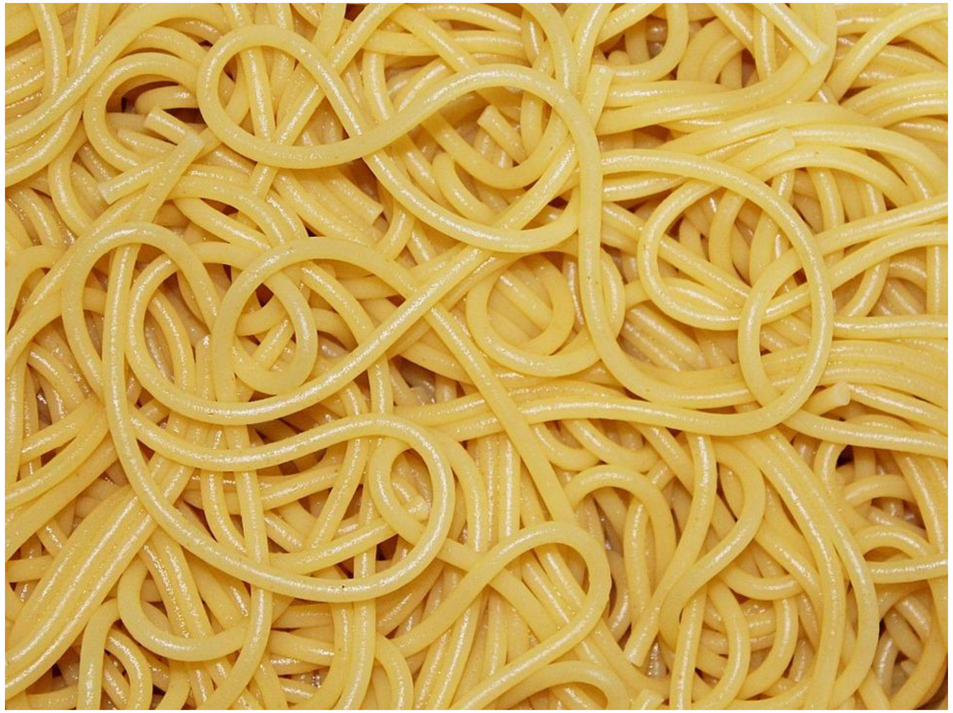


Data Preparation (Images, Text, etc.)

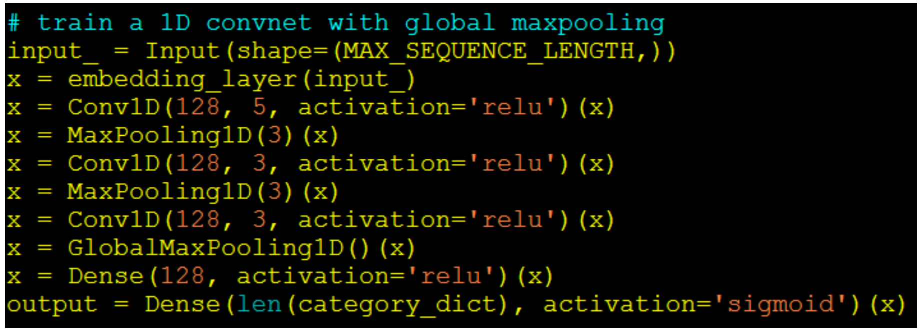
Feature Extraction

https://www.datanami.com/2018/09/05/how-to-build-a-better-machine-learning-pipeline/

Machine Learning made easy

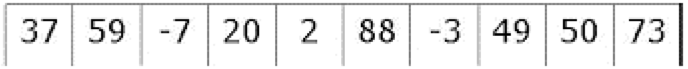


37	59	-7	20	2	88	-3	49	50	73

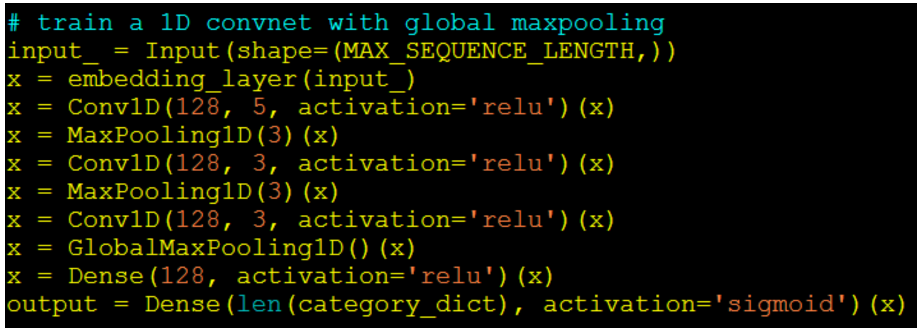


```
# train a 1D convnet with global maxpooling
input_ = Input(shape=(MAX_SEQUENCE_LENGTH,))
x = embedding_layer(input_)
x = Conv1D(128, 5, activation='relu')(x)
x = MaxPooling1D(3)(x)
x = Conv1D(128, 3, activation='relu')(x)
x = MaxPooling1D(3)(x)
x = Conv1D(128, 3, activation='relu')(x)
x = GlobalMaxPooling1D()(x)
x = Dense(128, activation='relu')(x)
output = Dense(len(category_dict), activation='sigmoid')(x)
```

Why you need to understand your data

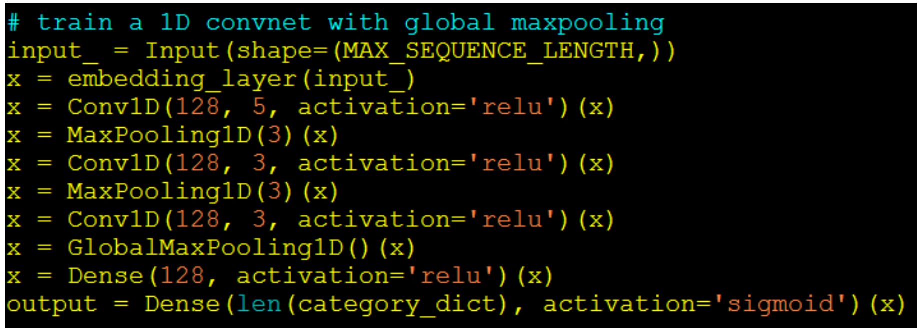


37	59	-7	20	2	88	-3	49	50	73



```
# train a 1D convnet with global maxpooling
input_ = Input(shape=(MAX_SEQUENCE_LENGTH,))
x = embedding_layer(input_)
x = Conv1D(128, 5, activation='relu')(x)
x = MaxPooling1D(3)(x)
x = Conv1D(128, 3, activation='relu')(x)
x = MaxPooling1D(3)(x)
x = Conv1D(128, 3, activation='relu')(x)
x = GlobalMaxPooling1D()(x)
x = Dense(128, activation='relu')(x)
output = Dense(len(category_dict), activation='sigmoid')(x)
```

👎

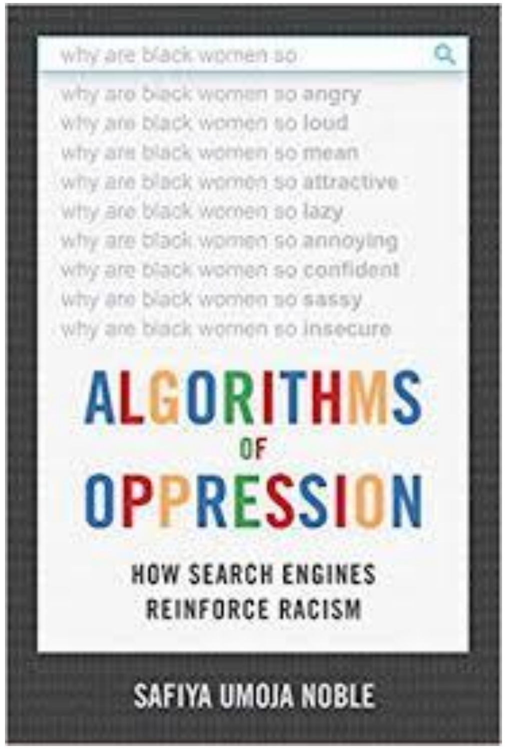


```
# train a 1D convnet with global maxpooling
input_ = Input(shape=(MAX_SEQUENCE_LENGTH,))
x = embedding_layer(input_)
x = Conv1D(128, 5, activation='relu')(x)
x = MaxPooling1D(3)(x)
x = Conv1D(128, 3, activation='relu')(x)
x = MaxPooling1D(3)(x)
x = Conv1D(128, 3, activation='relu')(x)
x = GlobalMaxPooling1D()(x)
x = Dense(128, activation='relu')(x)
output = Dense(len(category_dict), activation='sigmoid')(x)
```

👍

Why you should really understand your data!

Google faulted for racial bias in image search results for black teenagers



why are black women so

why are black women so angry
why are black women so loud
why are black women so mean
why are black women so attractive
why are black women so lazy
why are black women so annoying
why are black women so confident
why are black women so sassy
why are black women so insecure

ALGORITHMS OF OPPRESSION

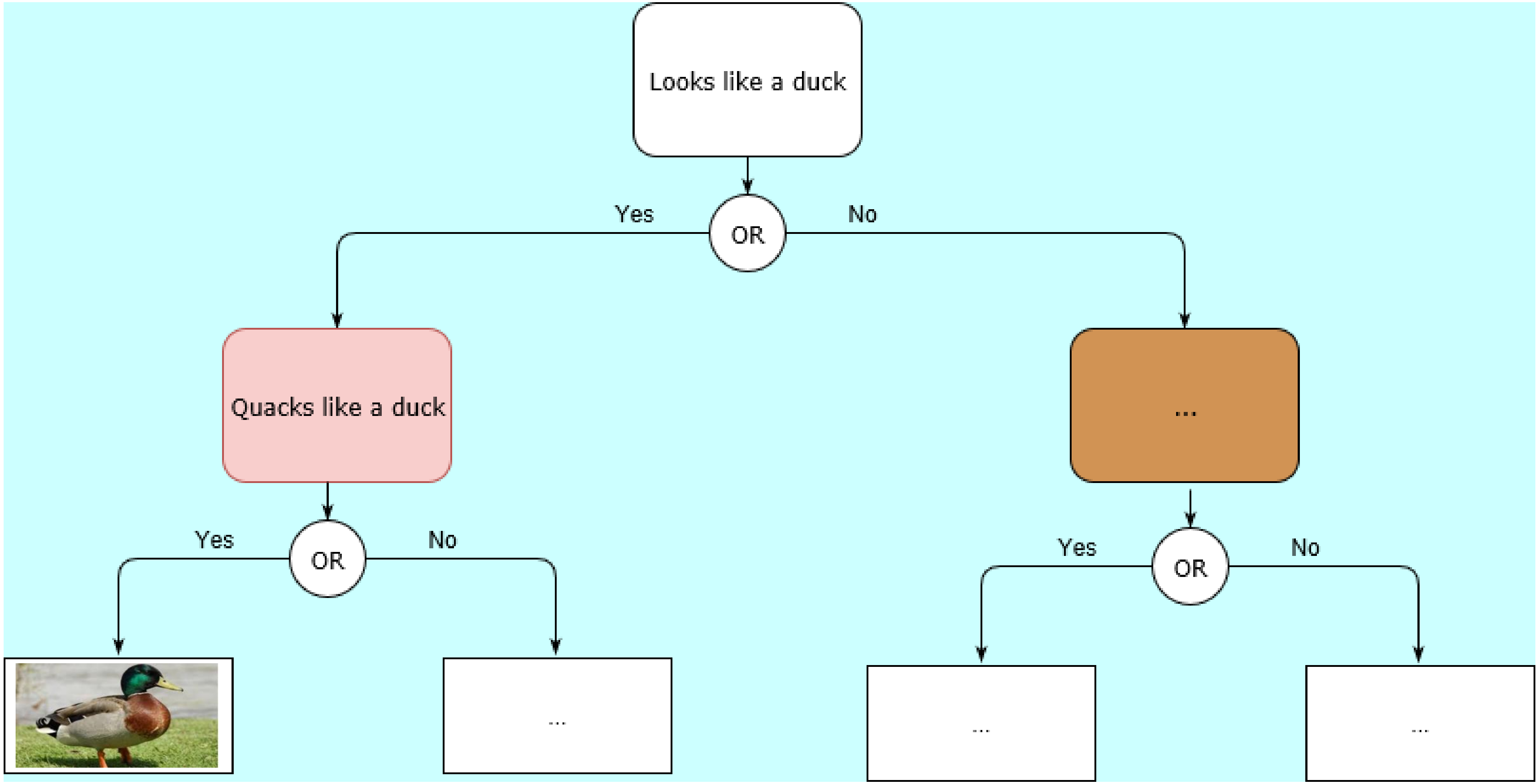
HOW SEARCH ENGINES REINFORCE RACISM

SAFIYA UMOJA NOBLE

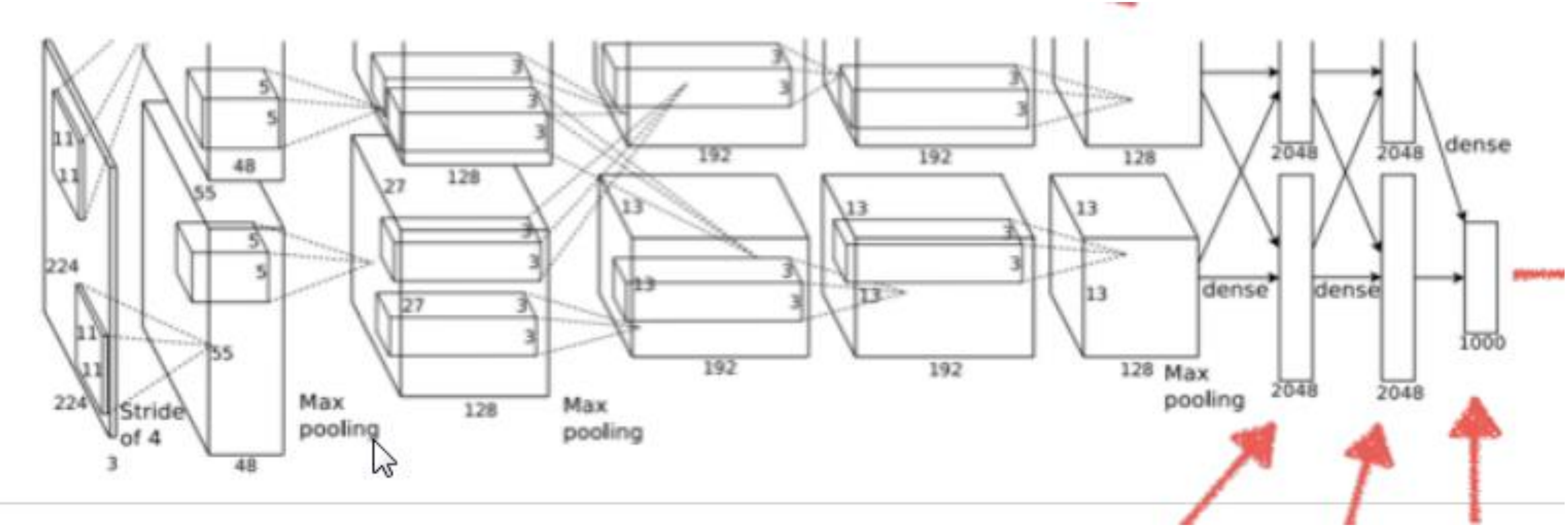
https://www.washingtonpost.com/news/morning-mix/wp/2016/06/10/google-faulted-for-racial-bias-in-image-search-results-for-black-teenagers/?noredirect=on&utm_term=.28b3b1204a19

GDPR: Right to an explanation

GDPR: Right to an explanation

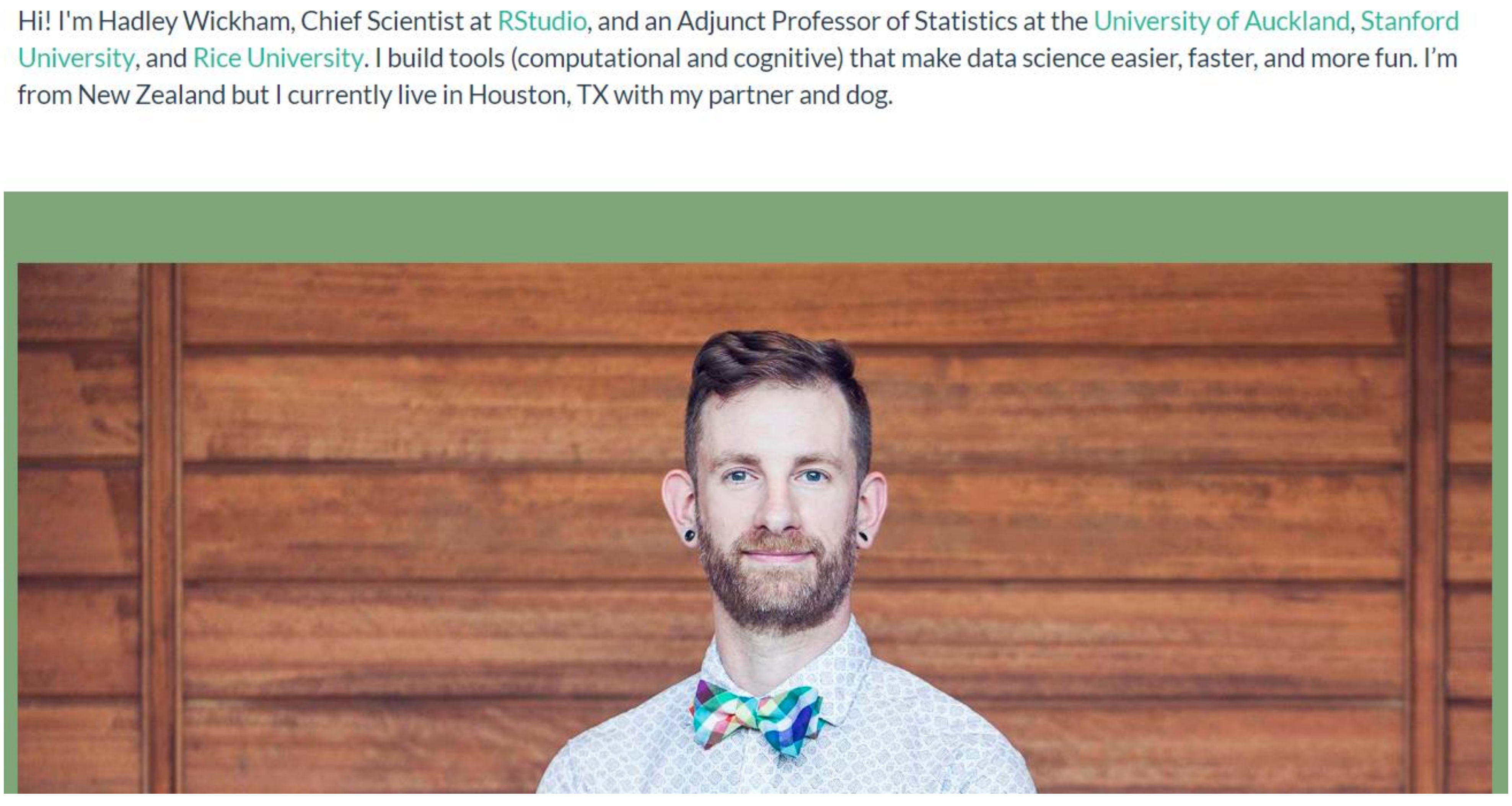


GDPR: Explain this



Introducing Tidy Data

hadley.nz

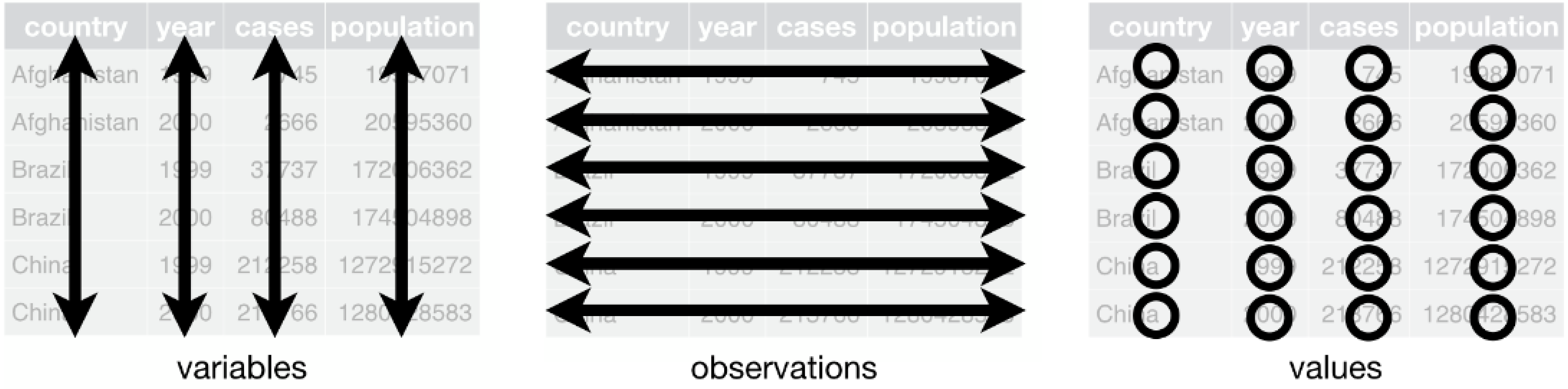


Hi! I'm Hadley Wickham, Chief Scientist at RStudio, and an Adjunct Professor of Statistics at the University of Auckland, Stanford University, and Rice University. I build tools (computational and cognitive) that make data science easier, faster, and more fun. I’m from New Zealand but I currently live in Houston, TX with my partner and dog.

Principles of Tidy Data

There are three interrelated rules which make a dataset tidy:

1. Each variable must have its own column.
2. Each observation must have its own row.
3. Each value must have its own cell.

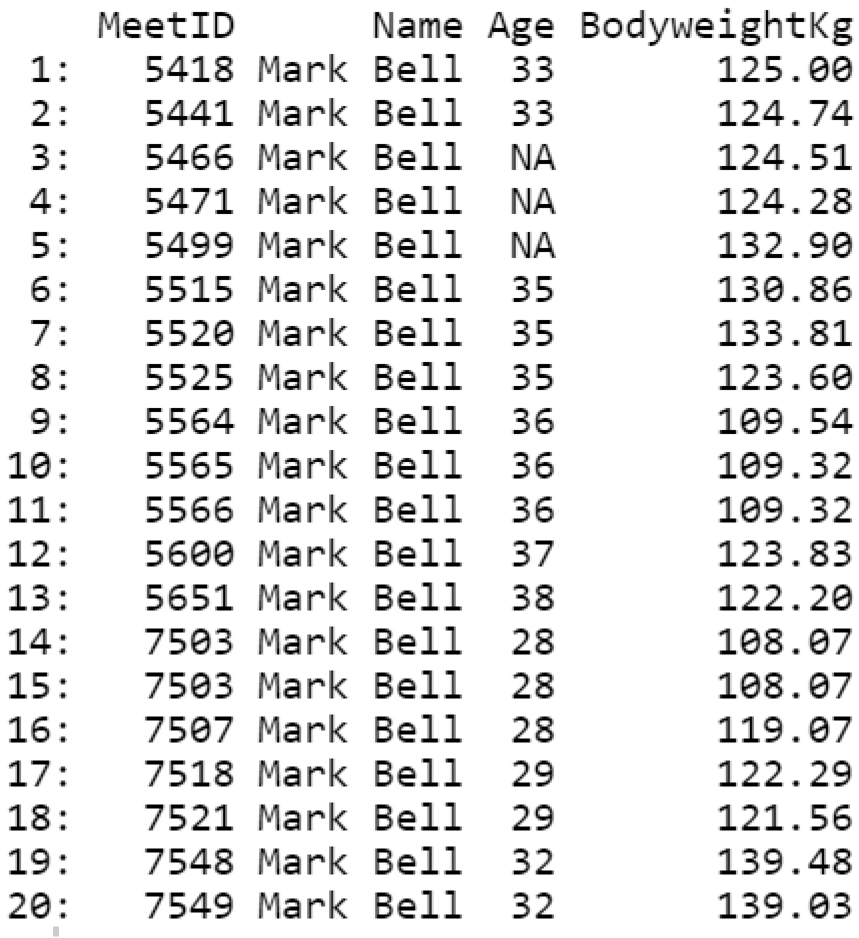


https://r4ds.had.co.nz/tidy-data.html

Exploratory Analysis

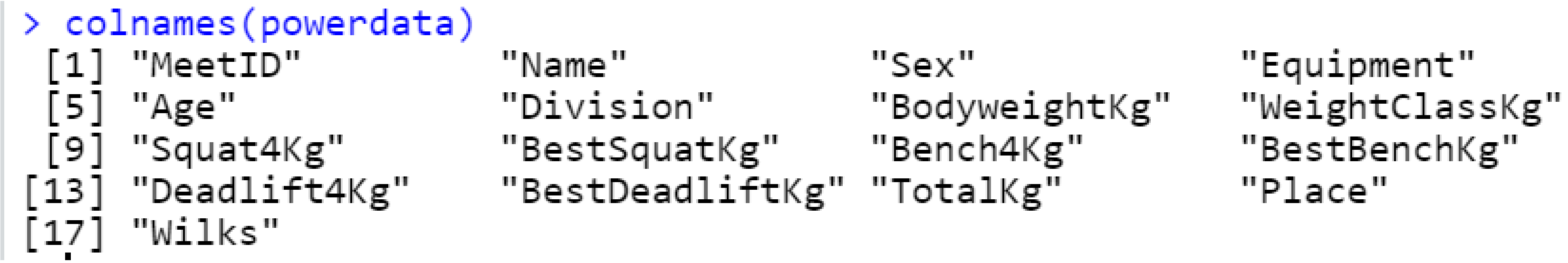
Powerlifting Data

https://www.kaggle.com/open-powerlifting/powerlifting-database

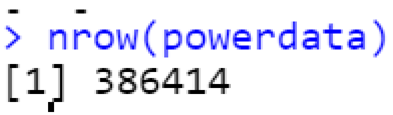


```
    MeetID      Name Age BodyweightKg
 1:   5418 Mark Bell  33       125.00
 2:   5441 Mark Bell  33       124.74
 3:   5466 Mark Bell  NA       124.51
 4:   5471 Mark Bell  NA       124.28
 5:   5499 Mark Bell  NA       132.90
 6:   5515 Mark Bell  35       130.86
 7:   5520 Mark Bell  35       133.81
 8:   5525 Mark Bell  35       123.60
 9:   5564 Mark Bell  36       109.54
10:   5565 Mark Bell  36       109.32
11:   5566 Mark Bell  36       109.32
12:   5600 Mark Bell  37       123.83
13:   5651 Mark Bell  38       122.20
14:   7503 Mark Bell  28       108.07
15:   7503 Mark Bell  28       108.07
16:   7507 Mark Bell  28       119.07
17:   7518 Mark Bell  29       122.29
18:   7521 Mark Bell  29       121.56
19:   7548 Mark Bell  32       139.48
20:   7549 Mark Bell  32       139.03
```

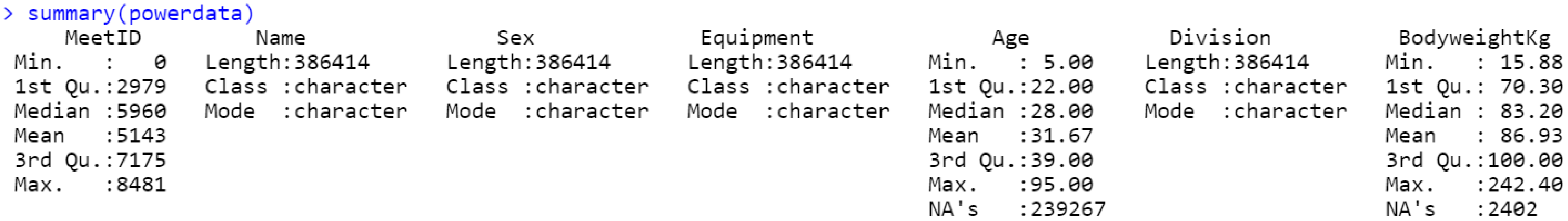
What have we got?



```
> colnames(powerdata)
 [1] "MeetID"         "Name"           "Sex"            "Equipment"
 [5] "Age"            "Division"       "BodyweightKg"   "WeightClassKg"
 [9] "Squat4Kg"       "BestSquatKg"    "Bench4Kg"       "BestBenchKg"
[13] "Deadlift4Kg"    "BestDeadliftKg" "TotalKg"        "Place"
[17] "Wilks"
```

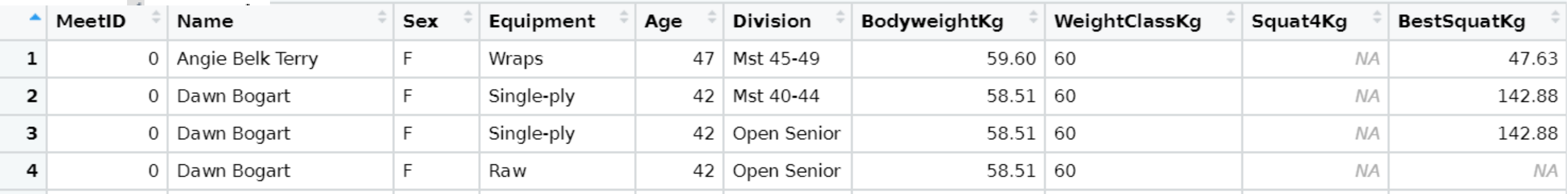


```
> nrow(powerdata)
[1] 386414
```



```
> summary(powerdata)
     MeetID          Name               Sex              Equipment              Age           Division         BodyweightKg
 Min.   :   0   Length:386414      Length:386414      Length:386414      Min.   : 5.00    Length:386414      Min.   : 15.88
 1st Qu.:2979   Class :character   Class :character   Class :character   1st Qu.:22.00    Class :character   1st Qu.: 70.30
 Median :5960   Mode  :character   Mode  :character   Mode  :character   Median :28.00    Mode  :character   Median : 83.20
 Mean   :5143                                                            Mean   :31.67                       Mean   : 86.93
 3rd Qu.:7175                                                            3rd Qu.:39.00                       3rd Qu.:100.00
 Max.   :8481                                                            Max.   :95.00                       Max.   :242.40
                                                                         NA's   :239267                      NA's   :2402
```

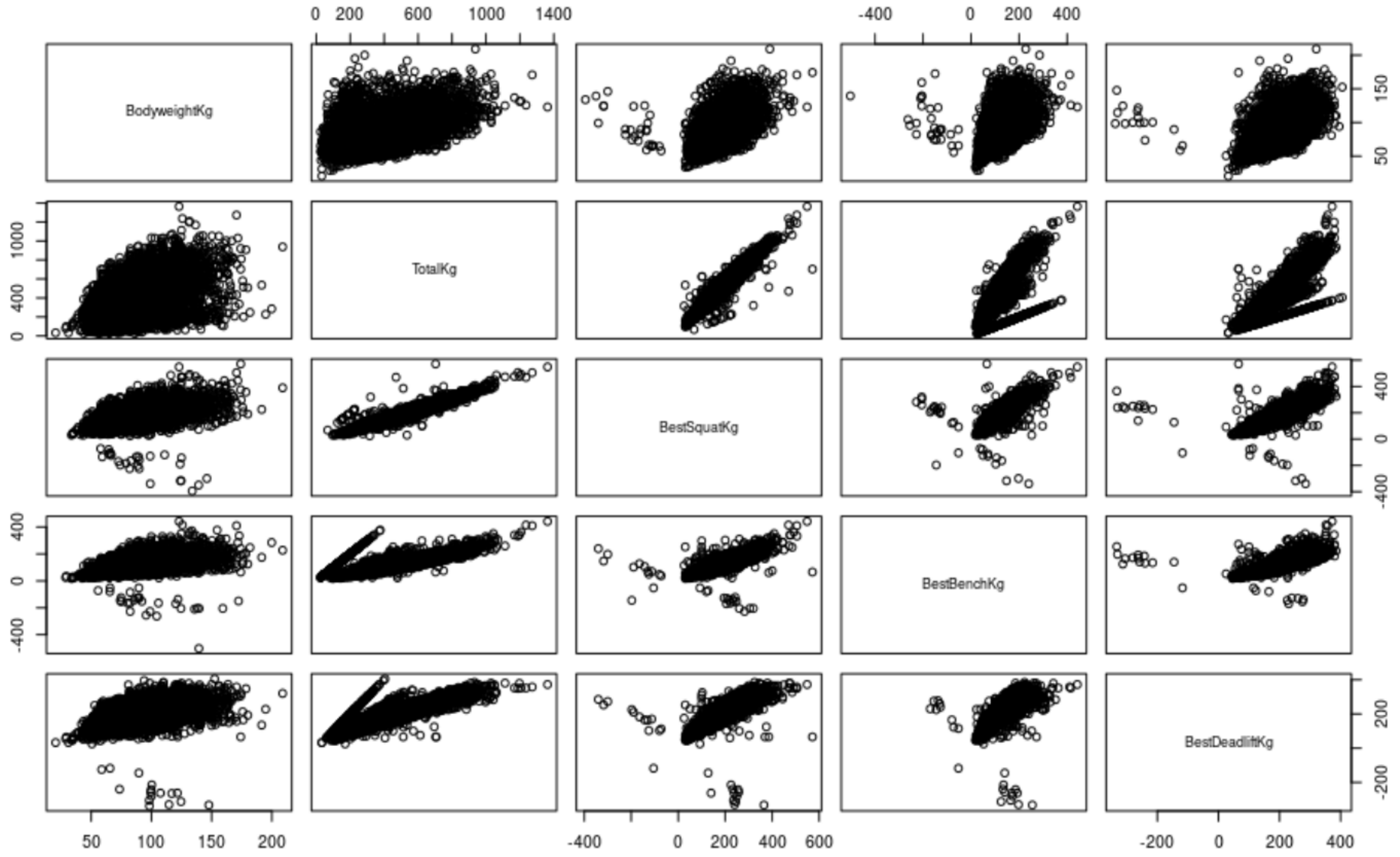
View(powerdata)



	MeetID	Name	Sex	Equipment	Age	Division	BodyweightKg	WeightClassKg	Squat4Kg	BestSquatKg
1	0	Angie Belk Terry	F	Wraps	47	Mst 45-49	59.60	60	NA	47.63
2	0	Dawn Bogart	F	Single-ply	42	Mst 40-44	58.51	60	NA	142.88
3	0	Dawn Bogart	F	Single-ply	42	Open Senior	58.51	60	NA	142.88
4	0	Dawn Bogart	F	Raw	42	Open Senior	58.51	60	NA	NA

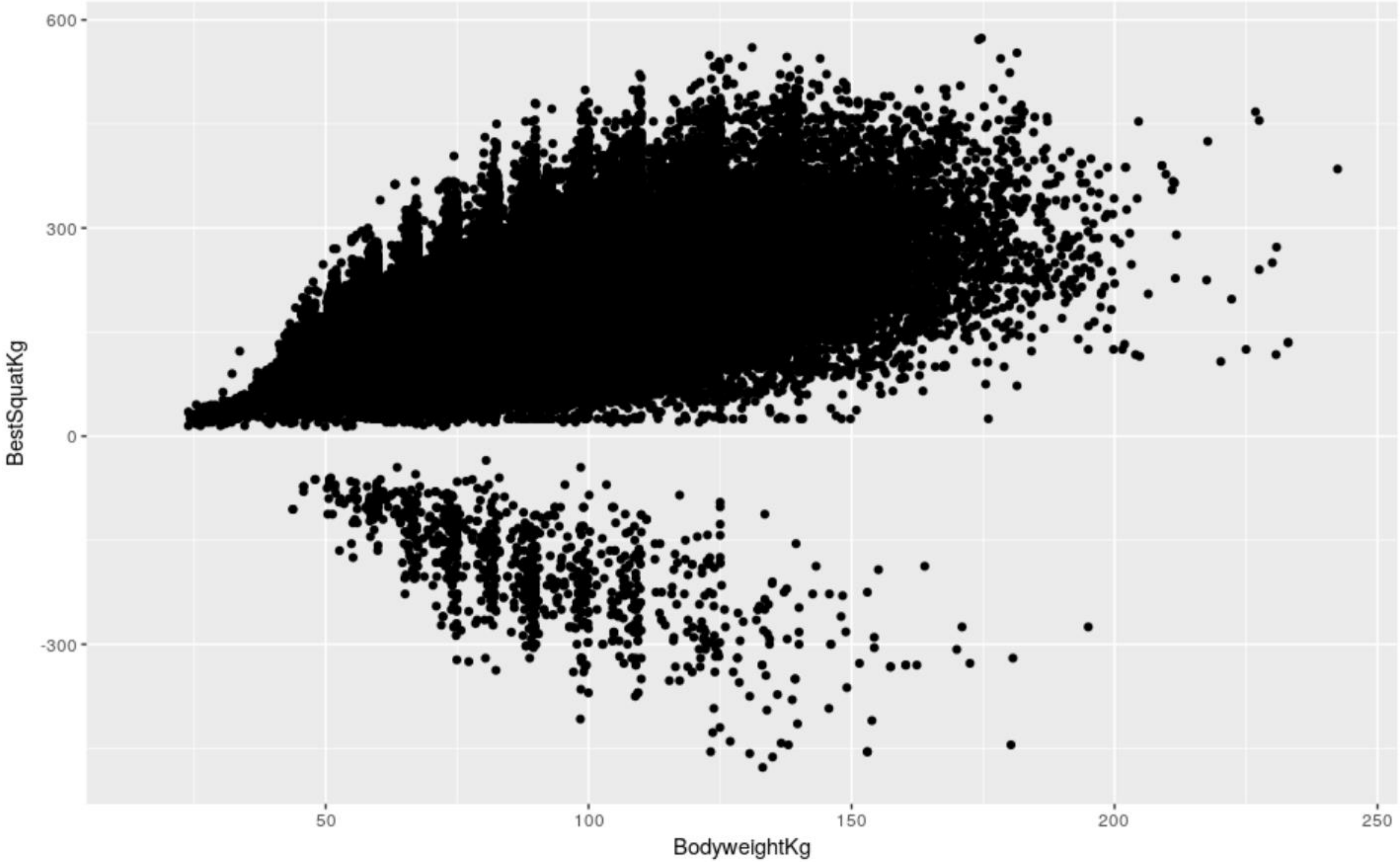
Plotting

```
powersample <- powerdata[sample(nrow(powerdata), 10000), ]
pairs(powersample[,c("BodyweightKg","TotalKg","BestSquatKg","BestBenchKg","BestDeadliftKg")])
```

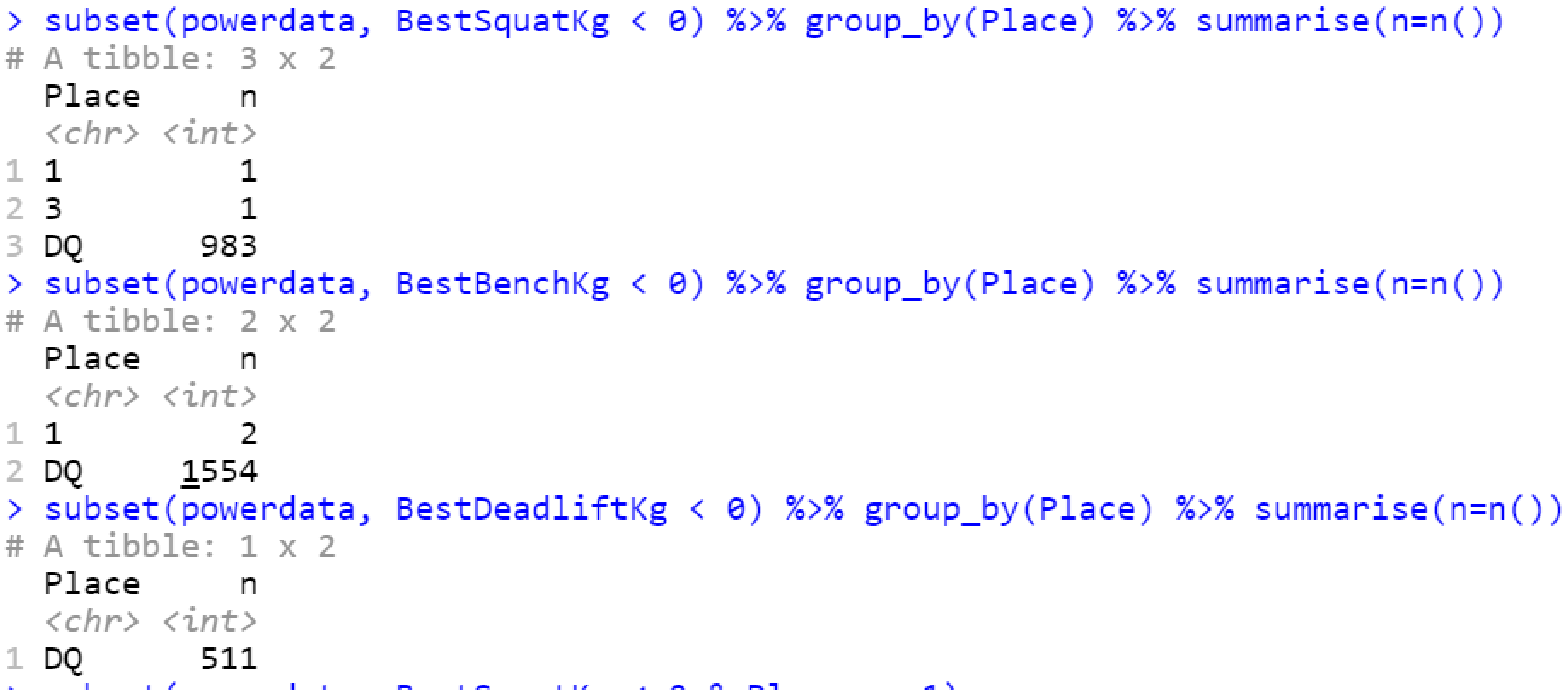


Negative reps

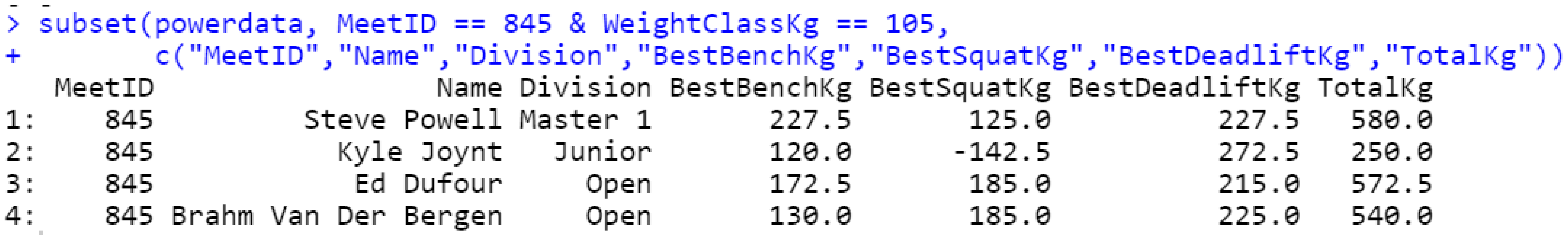
```
ggplot(powerdata, aes(x=BodyweightKg, y = BestSquatKg)) + geom_point()
```



Explaining the negatives



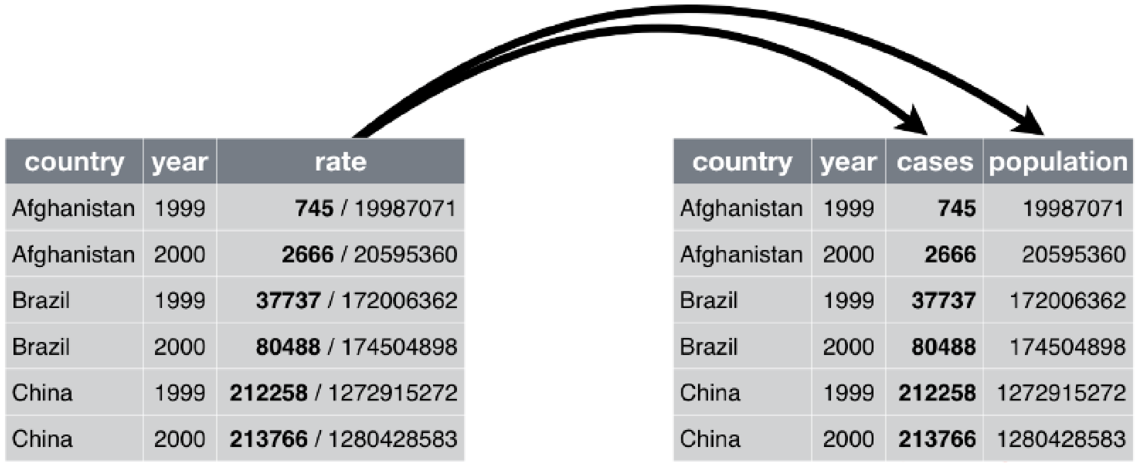
```
> subset(powerdata, BestSquatKg < 0) %>% group_by(Place) %>% summarise(n=n())
# A tibble: 3 x 2
  Place     n
  <chr> <int>
1 1         1
2 3         1
3 DQ      983
> subset(powerdata, BestBenchKg < 0) %>% group_by(Place) %>% summarise(n=n())
# A tibble: 2 x 2
  Place     n
  <chr> <int>
1 1         2
2 DQ     1554
> subset(powerdata, BestDeadliftKg < 0) %>% group_by(Place) %>% summarise(n=n())
# A tibble: 1 x 2
  Place     n
  <chr> <int>
1 DQ      511
```



```
> subset(powerdata, MeetID == 845 & WeightClassKg == 105,
+        c("MeetID","Name","Division","BestBenchKg","BestSquatKg","BestDeadliftKg","TotalKg"))
   MeetID                 Name Division BestBenchKg BestSquatKg BestDeadliftKg TotalKg
1:    845         Steve Powell Master 1       227.5       125.0          227.5   580.0
2:    845           Kyle Joynt   Junior       120.0      -142.5          272.5   250.0
3:    845            Ed Dufour     Open       172.5       185.0          215.0   572.5
4:    845 Brahm Van Der Bergen     Open       130.0       185.0          225.0   540.0
```

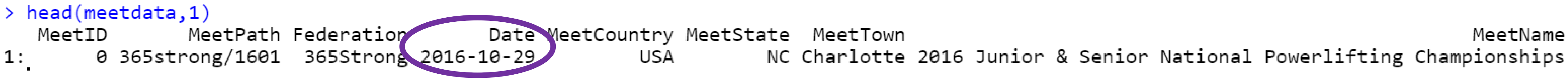
Tidying Data

Multiple variables stored in one column



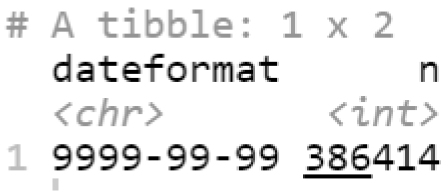
country	year	rate
Afghanistan	1999	**745** / 19987071
Afghanistan	2000	**2666** / 20595360
Brazil	1999	**37737** / 172006362
Brazil	2000	**80488** / 174504898
China	1999	**212258** / 1272915272
China	2000	**213766** / 1280428583

country	year	cases	population
Afghanistan	1999	**745**	19987071
Afghanistan	2000	**2666**	20595360
Brazil	1999	**37737**	172006362
Brazil	2000	**80488**	174504898
China	1999	**212258**	1272915272
China	2000	**213766**	1280428583



```
> head(meetdata,1)
   MeetID        MeetPath  Federation       Date MeetCountry MeetState   MeetTown                                          MeetName
1:      0 365strong/1601   365Strong 2016-10-29         USA        NC Charlotte 2016 Junior & Senior National Powerlifting Championships
```

```
meetdata$dateformat <- gsub("[0-9]","9", meetdata$Date)
meetdata %>% group_by(dateformat) %>% summarise(n=n()) %>% arrange(desc(n))
```

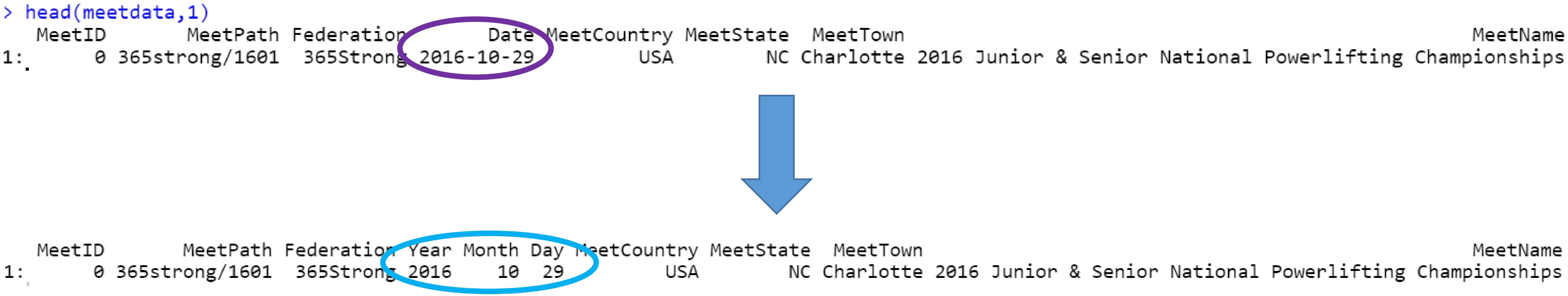


```
# A tibble: 1 x 2
  dateformat      n
  <chr>       <int>
1 9999-99-99 386414
```

https://r4ds.had.co.nz/tidy-data.html

Separating Dates

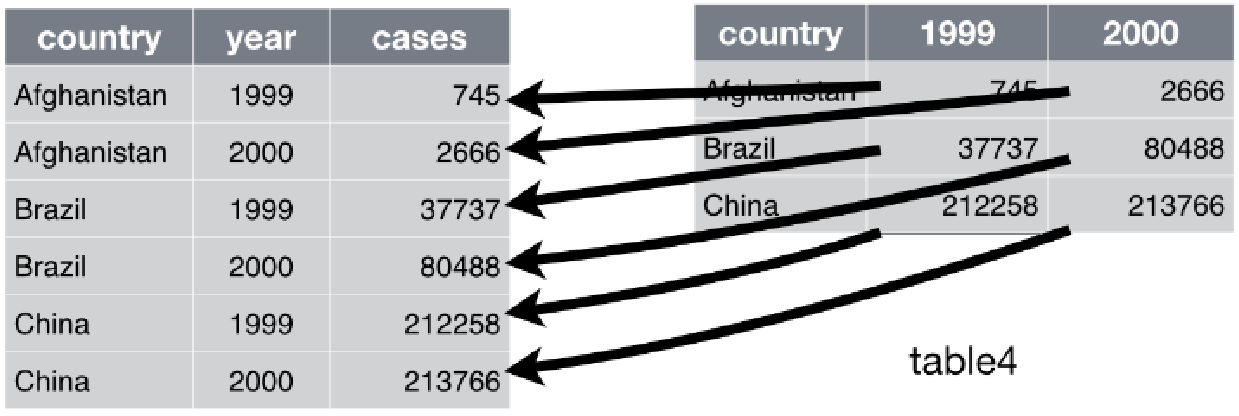
```
meetdata <- meetdata %>%   separate(Date, into = c("Year", "Month", "Day"))
meetdata $Year <- as.numeric(meetdata $Year)
meetdata $Month <- as.numeric(meetdata $Month)
meetdata $Day <- as.numeric(meetdata $Day)
```



```
> head(meetdata,1)
   MeetID       MeetPath Federation       Date MeetCountry MeetState  MeetTown                                                 MeetName
1:      0 365strong/1601  365Strong 2016-10-29         USA        NC Charlotte 2016 Junior & Senior National Powerlifting Championships
```

```
   MeetID       MeetPath Federation Year Month Day MeetCountry MeetState  MeetTown                                                 MeetName
1:      0 365strong/1601  365Strong 2016    10  29         USA        NC Charlotte 2016 Junior & Senior National Powerlifting Championships
```

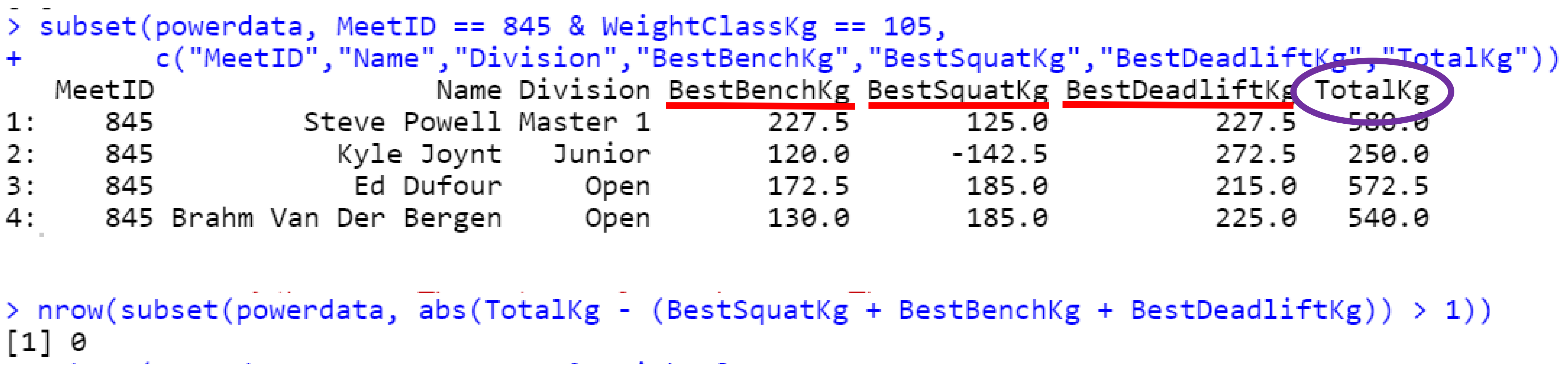
Column headers are values, not variable names



country	year	cases
Afghanistan	1999	745
Afghanistan	2000	2666
Brazil	1999	37737
Brazil	2000	80488
China	1999	212258
China	2000	213766

country	1999	2000
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

table4



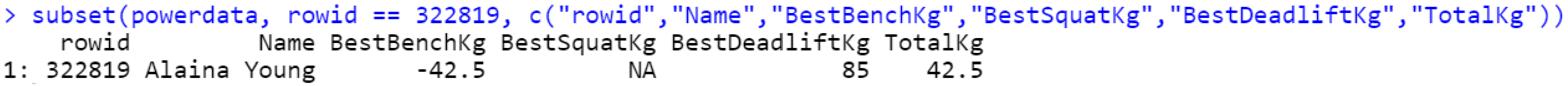
```
> subset(powerdata, MeetID == 845 & WeightClassKg == 105,
+        c("MeetID","Name","Division","BestBenchKg","BestSquatKg","BestDeadliftKg","TotalKg"))
   MeetID                 Name Division BestBenchKg BestSquatKg BestDeadliftKg TotalKg
1:    845         Steve Powell Master 1       227.5       125.0          227.5   580.0
2:    845           Kyle Joynt   Junior       120.0      -142.5          272.5   250.0
3:    845            Ed Dufour     Open       172.5       185.0          215.0   572.5
4:    845  Brahm Van Der Bergen     Open       130.0       185.0          225.0   540.0

> nrow(subset(powerdata, abs(TotalKg - (BestSquatKg + BestBenchKg + BestDeadliftKg)) > 1))
[1] 0
```

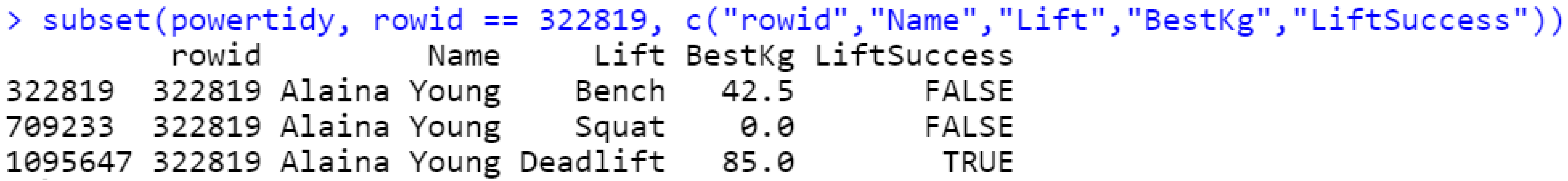
https://r4ds.had.co.nz/tidy-data.html

Turning headings into variables

```
powerdata$rowid <- rownames(powerdata)head(powerdata)
powerdata <- powerdata[,-c("Squat4Kg","Bench4Kg","Deadlift4Kg","TotalKg","Wilks")] %>%
  gather(`BestBenchKg`, `BestSquatKg`, `BestDeadliftKg`, key = "Lift", value = "BestKg") %>%
  mutate(Lift = sub("Kg","",sub("Best", "", Lift)))
```



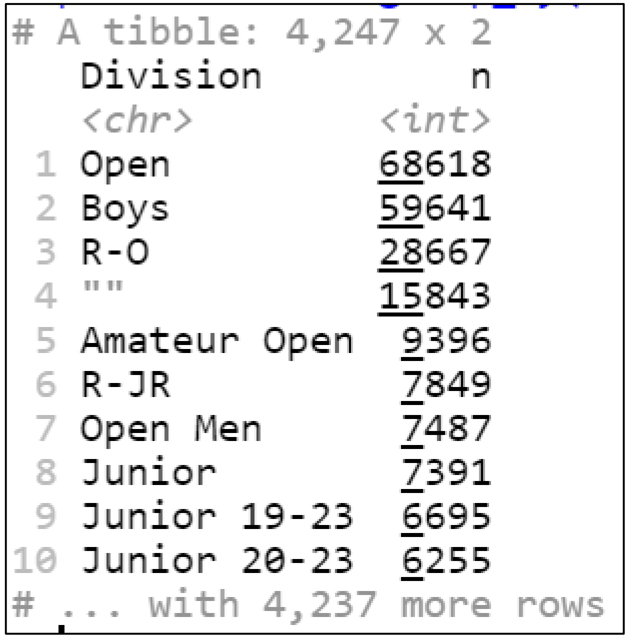
```
> subset(powerdata, rowid == 322819, c("rowid","Name","BestBenchKg","BestSquatKg","BestDeadliftKg","TotalKg"))
    rowid         Name BestBenchKg BestSquatKg BestDeadliftKg TotalKg
1: 322819 Alaina Young       -42.5          NA             85    42.5
```



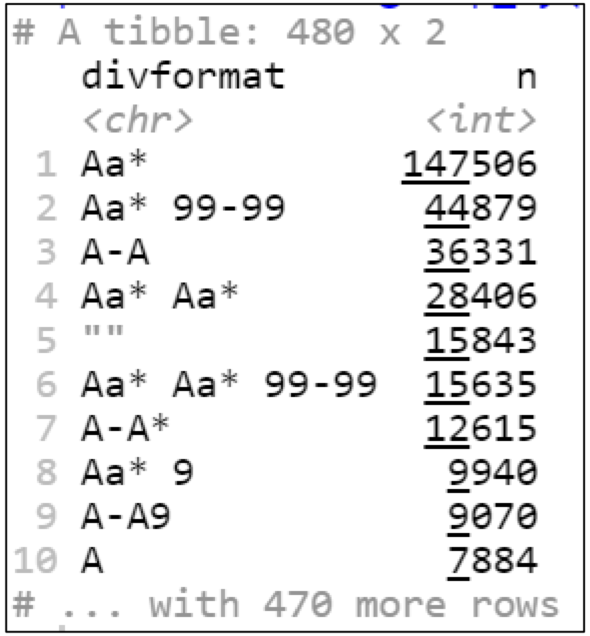
```
> subset(powertidy, rowid == 322819, c("rowid","Name","Lift","BestKg","LiftSuccess"))
         rowid         Name     Lift BestKg LiftSuccess
322819  322819 Alaina Young    Bench   42.5       FALSE
709233  322819 Alaina Young    Squat    0.0       FALSE
1095647 322819 Alaina Young Deadlift   85.0        TRUE
```

Tidying Text Columns

```
powerdata$divformat <- gsub("[[:upper:]]","A",powerdata$Division)
powerdata$divformat <- gsub("[[:lower:]]","a",powerdata$divformat)
powerdata$divformat <- gsub("[0-9]","9",powerdata$divformat)
powerdata$divformat <- gsub("[A]{2,}","A*",powerdata$divformat)
powerdata$divformat <- gsub("[a]{2,}","a*",powerdata$divformat)
powerdata %>% group_by(divformat) %>% summarise(n=n()) %>% arrange(desc(n))
```



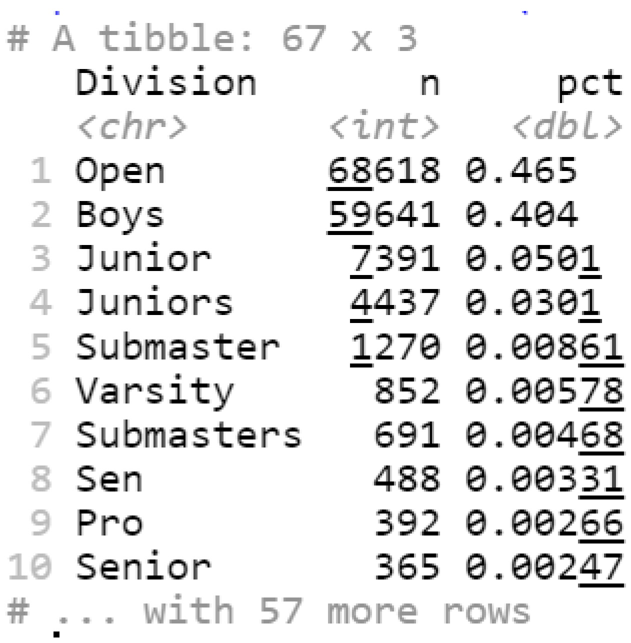
```
# A tibble: 4,247 x 2
   Division           n
   <chr>          <int>
 1 Open           68618
 2 Boys           59641
 3 R-O            28667
 4 ""             15843
 5 Amateur Open    9396
 6 R-JR            7849
 7 Open Men        7487
 8 Junior          7391
 9 Junior 19-23    6695
10 Junior 20-23    6255
# ... with 4,237 more rows
```



```
# A tibble: 480 x 2
   divformat          n
   <chr>          <int>
 1 Aa*           147506
 2 Aa* 99-99      44879
 3 A-A            36331
 4 Aa* Aa*        28406
 5 ""             15843
 6 Aa* Aa* 99-99  15635
 7 A-A*           12615
 8 Aa* 9           9940
 9 A-A9            9070
10 A               7884
# ... with 470 more rows
```

Tidying Text Columns

```
powerdata %>% filter(divformat == "Aa*") %>% group_by(Division) %>%
  summarise(n=n(),pct=n()/147506) %>% arrange(desc(n)) %>% print(n=10)
```

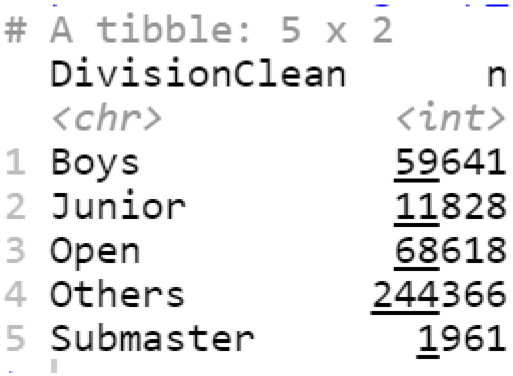


```
# A tibble: 67 x 3
   Division       n     pct
   <chr>      <int>   <dbl>
 1 Open       68618 0.465
 2 Boys       59641 0.404
 3 Junior      7391 0.0501
 4 Juniors     4437 0.0301
 5 Submaster   1270 0.00861
 6 Varsity      852 0.00578
 7 Submasters   691 0.00468
 8 Sen          488 0.00331
 9 Pro          392 0.00266
10 Senior       365 0.00247
# ... with 57 more rows
```

Tidying Text Columns

```
powerdata$DivisionClean <- "Others‟
powerdata$DivisionClean[powerdata$Division == 'Open'] <- 'Open‛
powerdata$DivisionClean[powerdata$Division == 'Boys'] <- 'Boys‛
powerdata$DivisionClean[powerdata$Division %in% c('Junior','Juniors')] <- 'Junior‛
powerdata$DivisionClean[powerdata$Division %in% c('Submaster','Submasters')] <- 'Submaster'
```

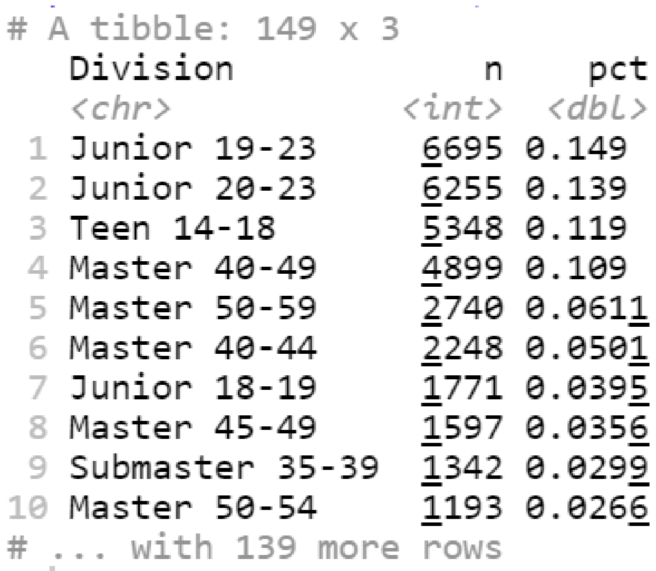
```
powerdata %>%
 group_by(DivisionClean) %>%
 summarise(n = n())
```



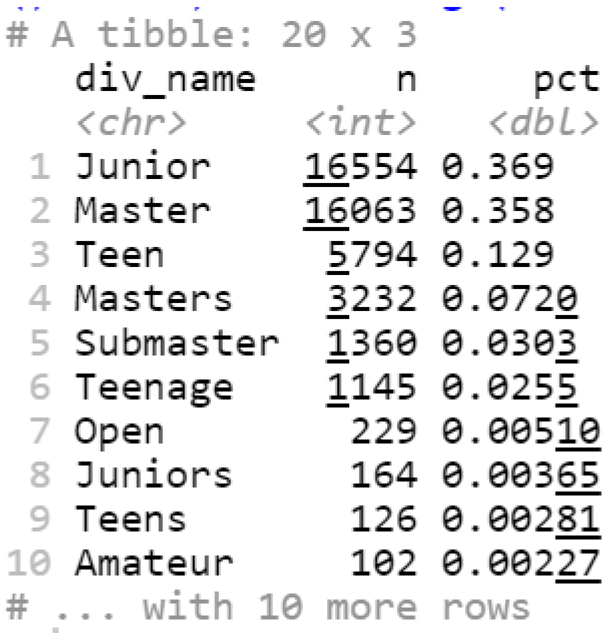
```
# A tibble: 5 x 2
  DivisionClean      n
  <chr>          <int>
1 Boys           59641
2 Junior         11828
3 Open           68618
4 Others        244366
5 Submaster       1961
```

Tidying Text Columns

```
powerdata %>% filter(divformat == "Aa* 99-99") %>%
  group_by(Division) %>%
  summarise(n=n(),pct=n()/44879) %>%
  arrange(desc(n)) %>%
  print(n=10)
```



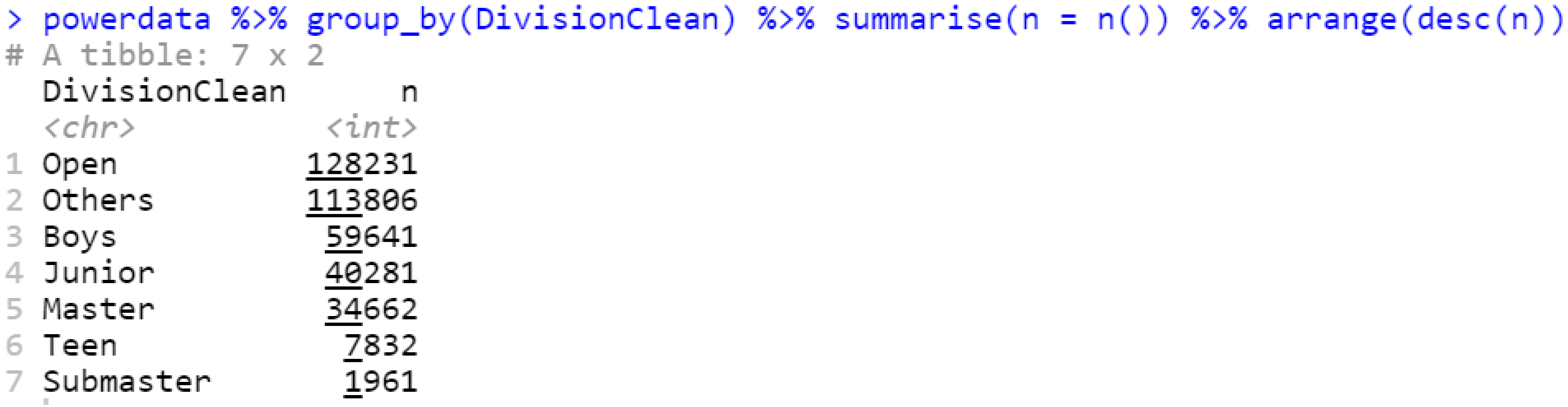
```
# A tibble: 149 x 3
   Division            n    pct
   <chr>           <int>  <dbl>
 1 Junior 19-23     6695 0.149
 2 Junior 20-23     6255 0.139
 3 Teen 14-18       5348 0.119
 4 Master 40-49     4899 0.109
 5 Master 50-59     2740 0.0611
 6 Master 40-44     2248 0.0501
 7 Junior 18-19     1771 0.0395
 8 Master 45-49     1597 0.0356
 9 Submaster 35-39  1342 0.0299
10 Master 50-54     1193 0.0266
# ... with 139 more rows
```



```
# A tibble: 20 x 3
   div_name      n     pct
   <chr>     <int>   <dbl>
 1 Junior    16554 0.369
 2 Master    16063 0.358
 3 Teen       5794 0.129
 4 Masters    3232 0.0720
 5 Submaster  1360 0.0303
 6 Teenage    1145 0.0255
 7 Open        229 0.00510
 8 Juniors     164 0.00365
 9 Teens       126 0.00281
10 Amateur     102 0.00227
# ... with 10 more rows
```

```
powerdata %>%
 filter(divformat == "Aa* 99-99") %>%
 separate(Division,c("div_name","age_group"),sep=" ") %>%
 group_by(div_name) %>% summarise(n=n(),pct=n()/44879) %>%
 arrange(desc(n)) %>% print(n=10)
```

Tidying Text Columns

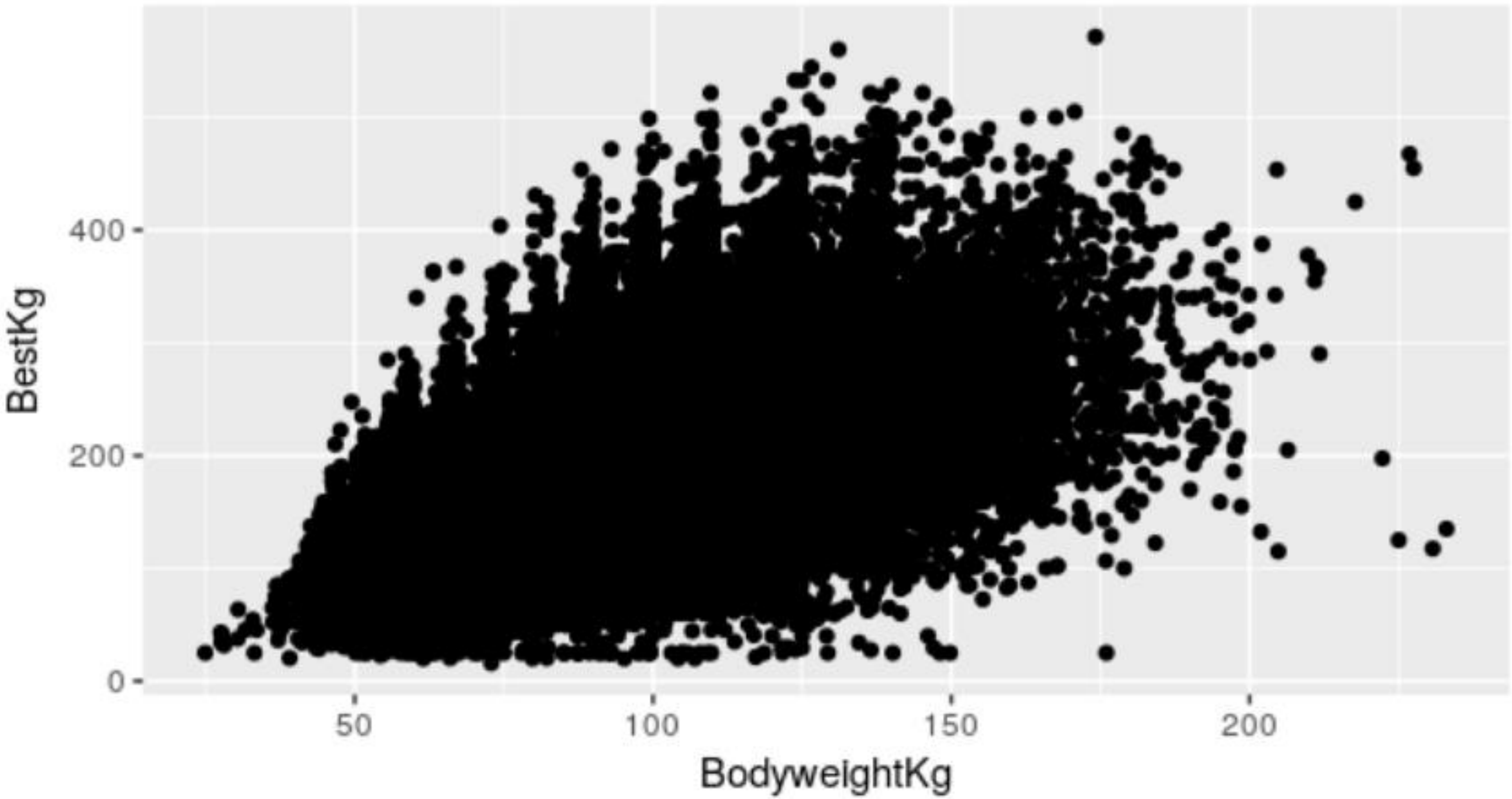


```
> powerdata %>% group_by(DivisionClean) %>% summarise(n = n()) %>% arrange(desc(n))
# A tibble: 7 x 2
  DivisionClean      n
  <chr>          <int>
1 Open          128231
2 Others        113806
3 Boys           59641
4 Junior         40281
5 Master         34662
6 Teen            7832
7 Submaster       1961
```

Back to the plot

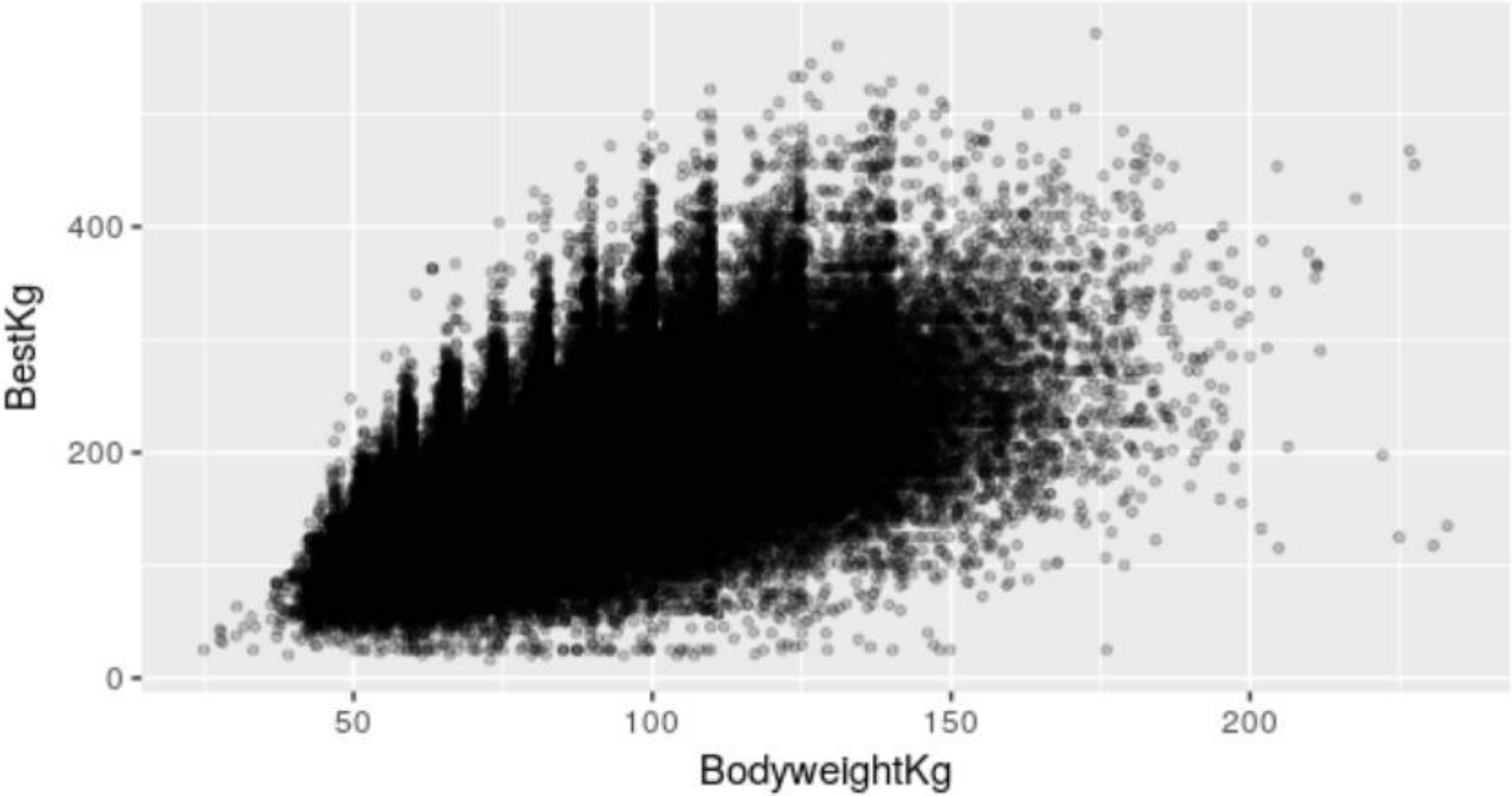
Squats again

```
ggplot(subset(powertidy,LiftSuccess == TRUE &
Lift == "Squat" & DivisionClean != "Others"),
aes(x = BodyweightKg, y = BestKg)) + geom_point()
```



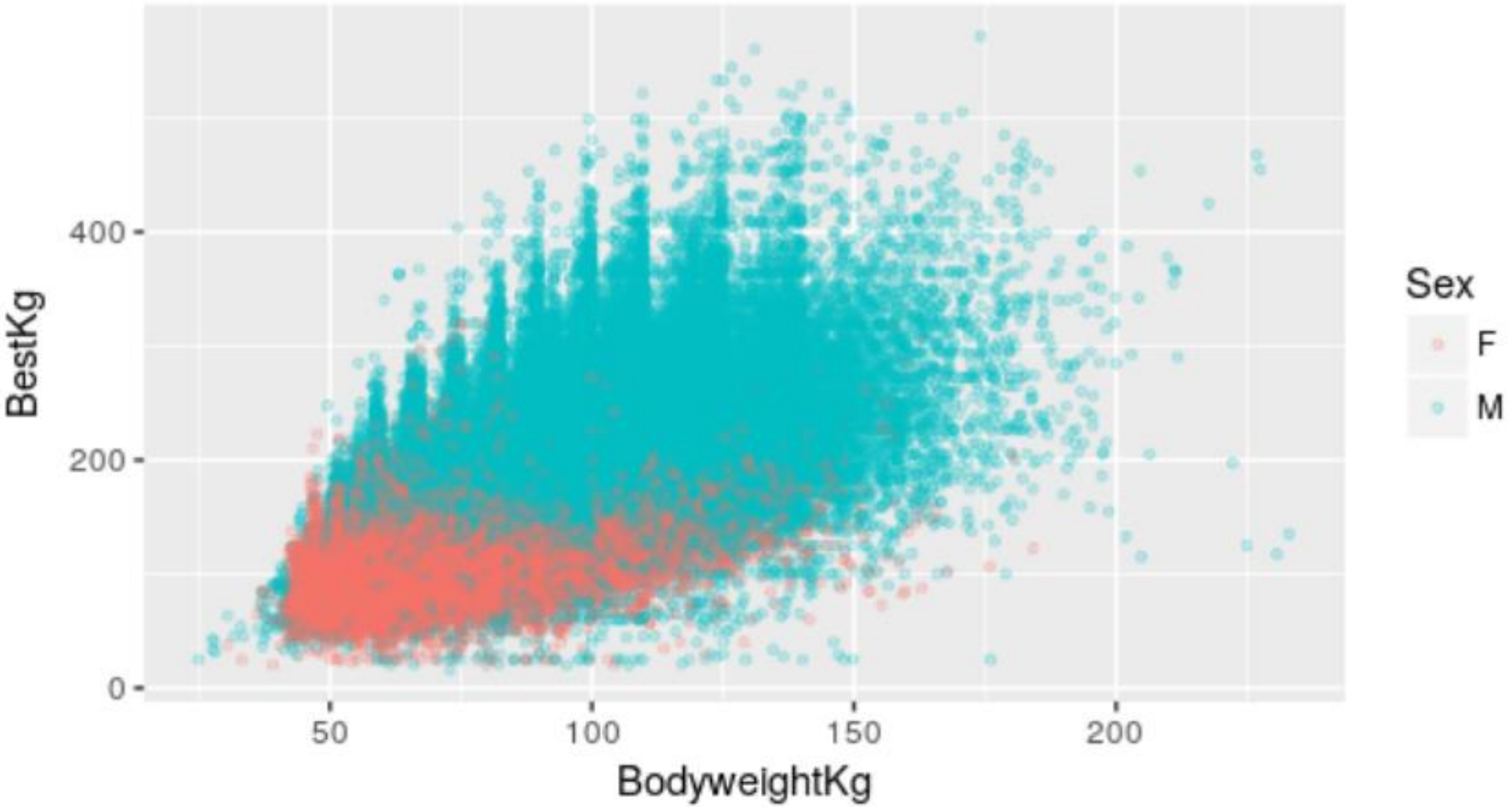
Alpha channel

```
ggplot(subset(powertidy,LiftSuccess == TRUE & Lift == "Squat"
& DivisionClean != "Others"), aes(x = BodyweightKg, y =
BestKg)) + geom_point(alpha = 0.2, size = 1)
```



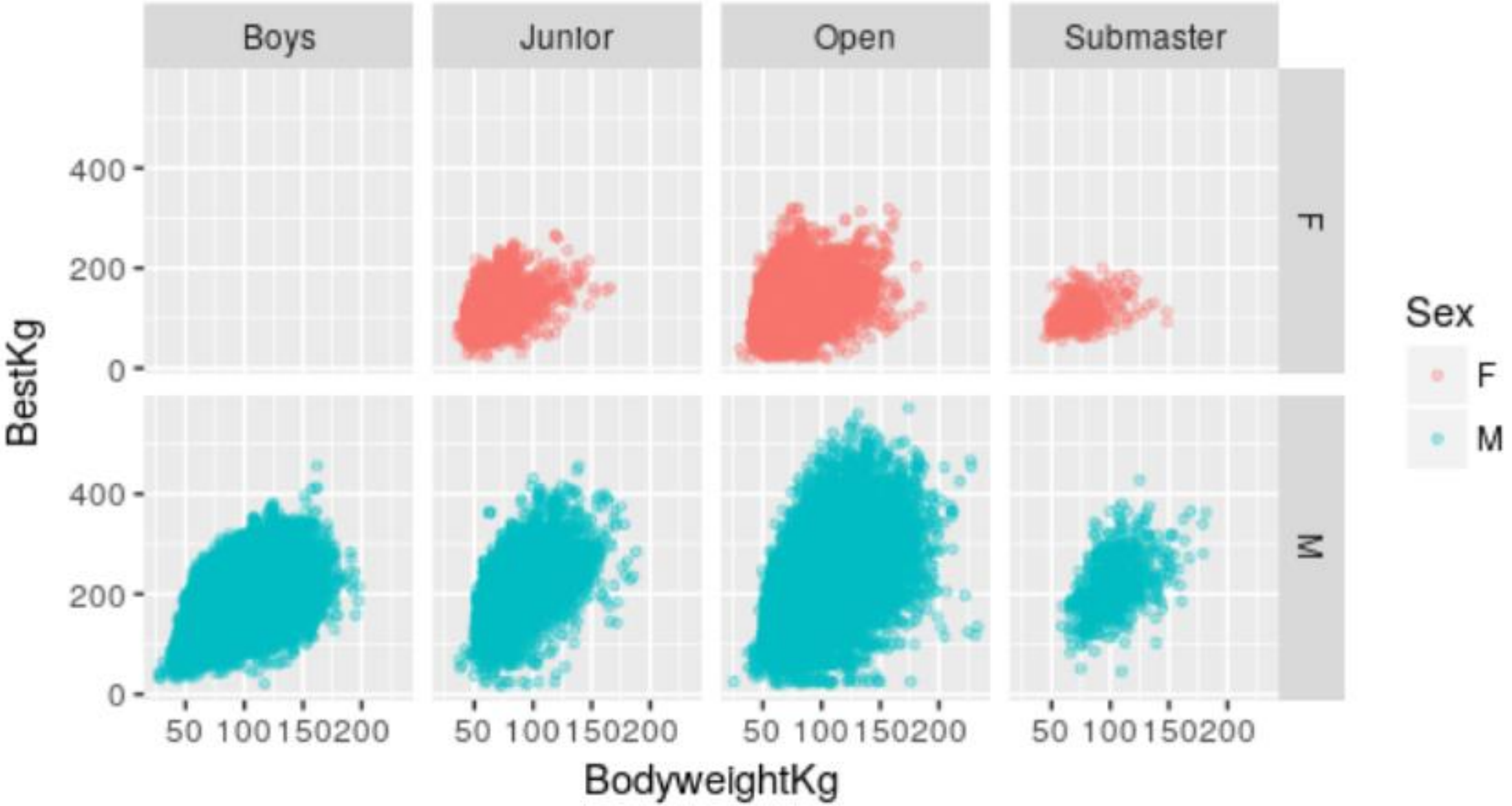
Colour by variable

```
ggplot(subset(powertidy,LiftSuccess == TRUE & Lift == "Squat"
& DivisionClean != "Others"), aes(x = BodyweightKg, y = BestKg,
colour = Sex)) + geom_point(alpha = 0.2, size = 1)
```



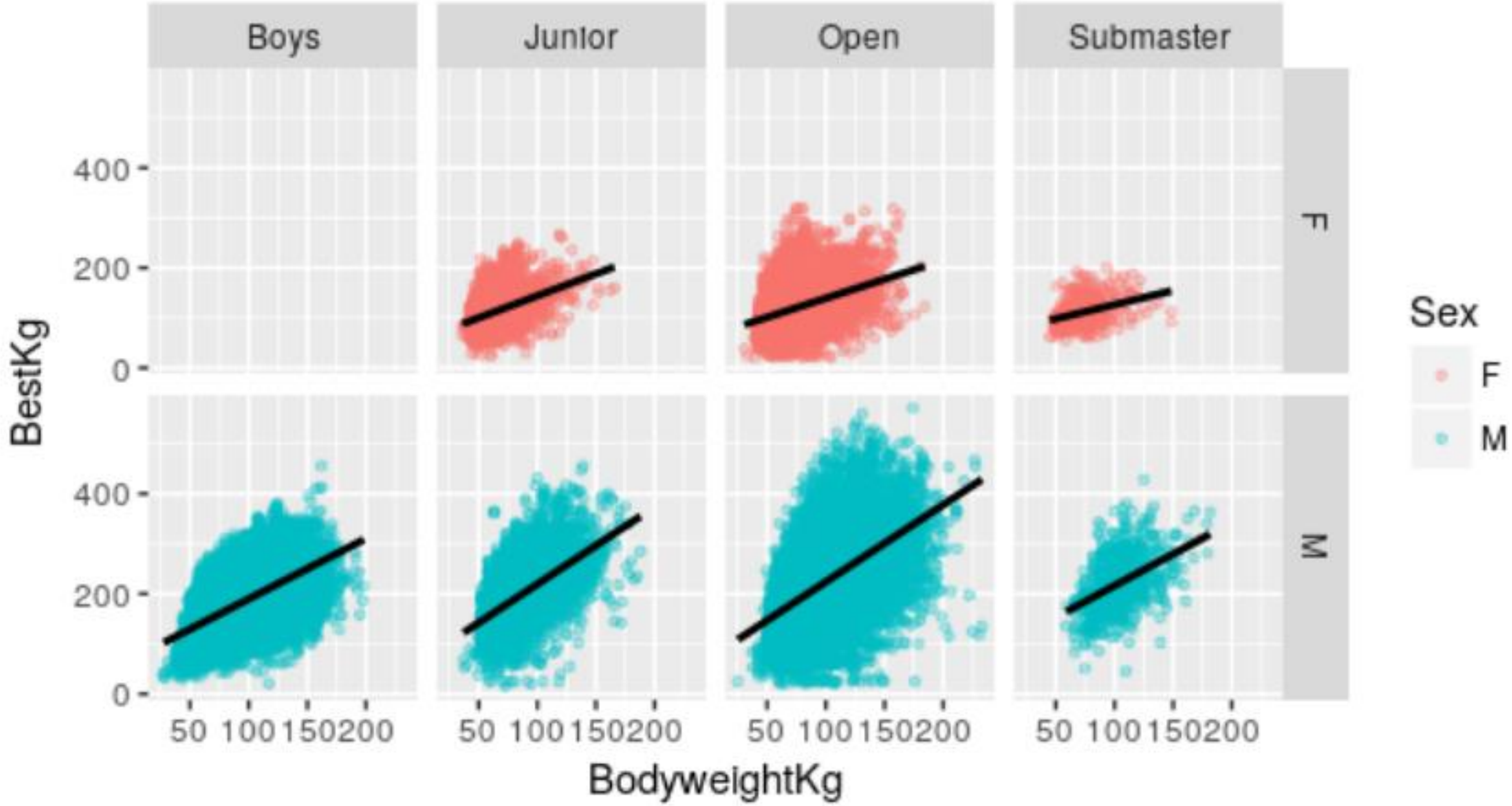
Facet Plots

```
ggplot(subset(powertidy,LiftSuccess == TRUE & Lift == "Squat"
& DivisionClean != "Others"), aes(x = BodyweightKg, y = BestKg,
colour = Sex)) + geom_point(alpha = 0.2, size = 1) +
facet_grid(Sex ~ DivisionClean)
```



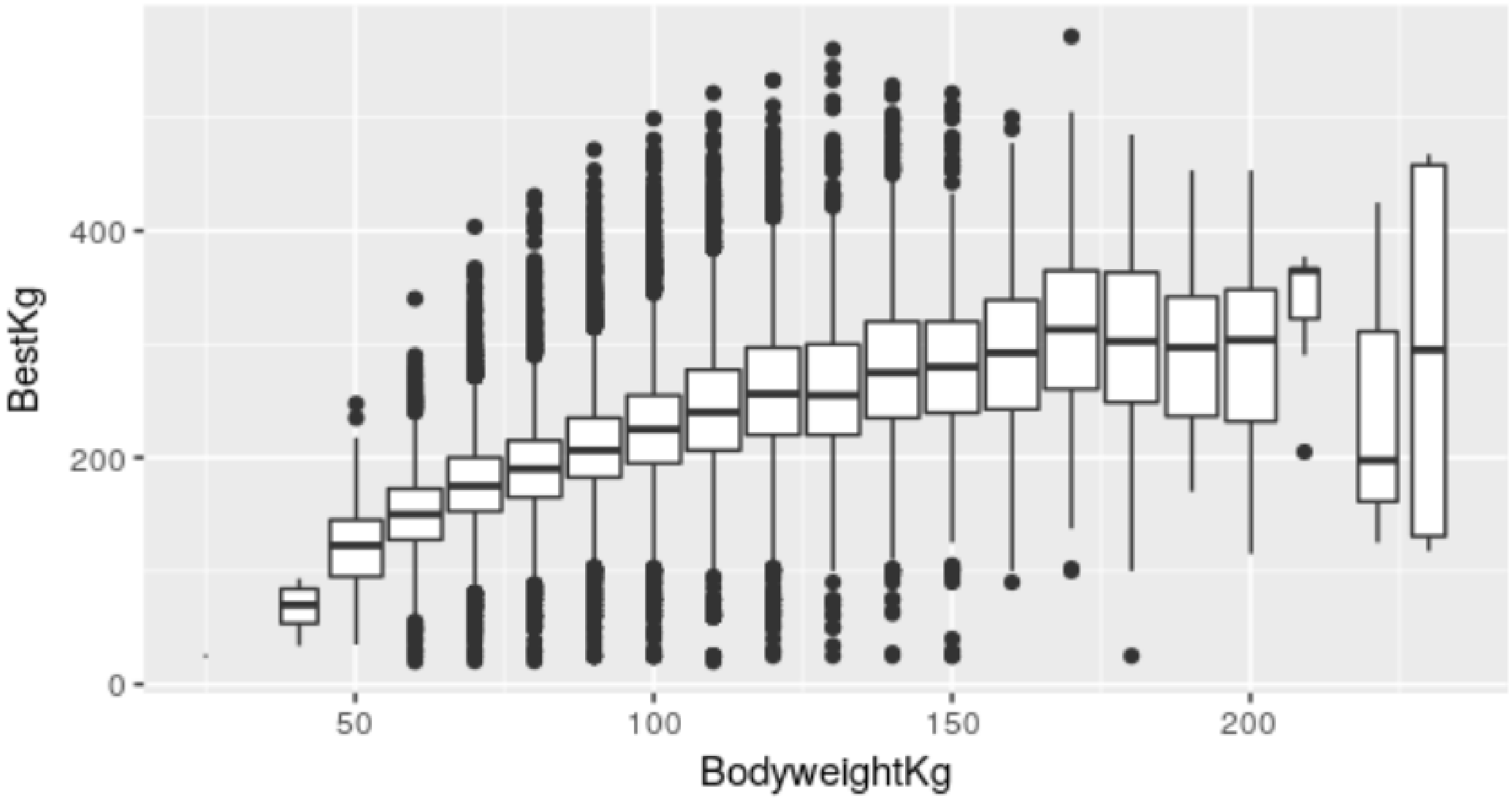
Regression lines

```
ggplot(subset(powertidy,LiftSuccess == TRUE & Lift == "Squat"
& DivisionClean != "Others"), aes(x = BodyweightKg, y = BestKg,
colour = Sex)) + geom_point(alpha = 0.2, size = 1) +
facet_grid(Sex ~ DivisionClean) + stat_smooth(method='lm',
colour='black')
```



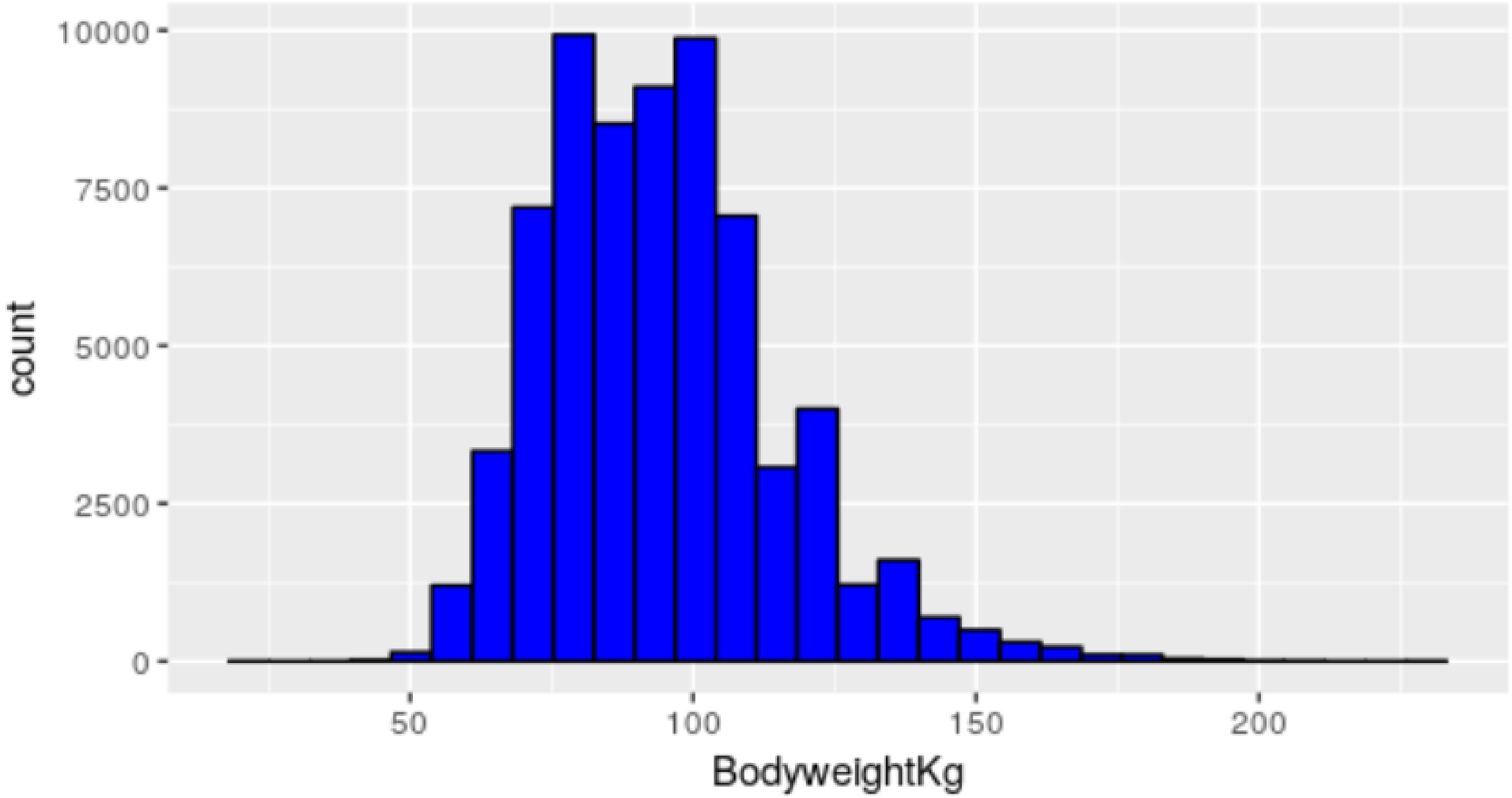
Boxplots

```
ggplot(subset(powertidy,LiftSuccess == TRUE &
Lift == "Squat" &                DivisionClean ==
"Open" & Sex == 'M'),       aes(x = BodyweightKg,
y = BestKg)) +  geom_boxplot(mapping = aes(group
= cut_width(BodyweightKg, 10)))
```



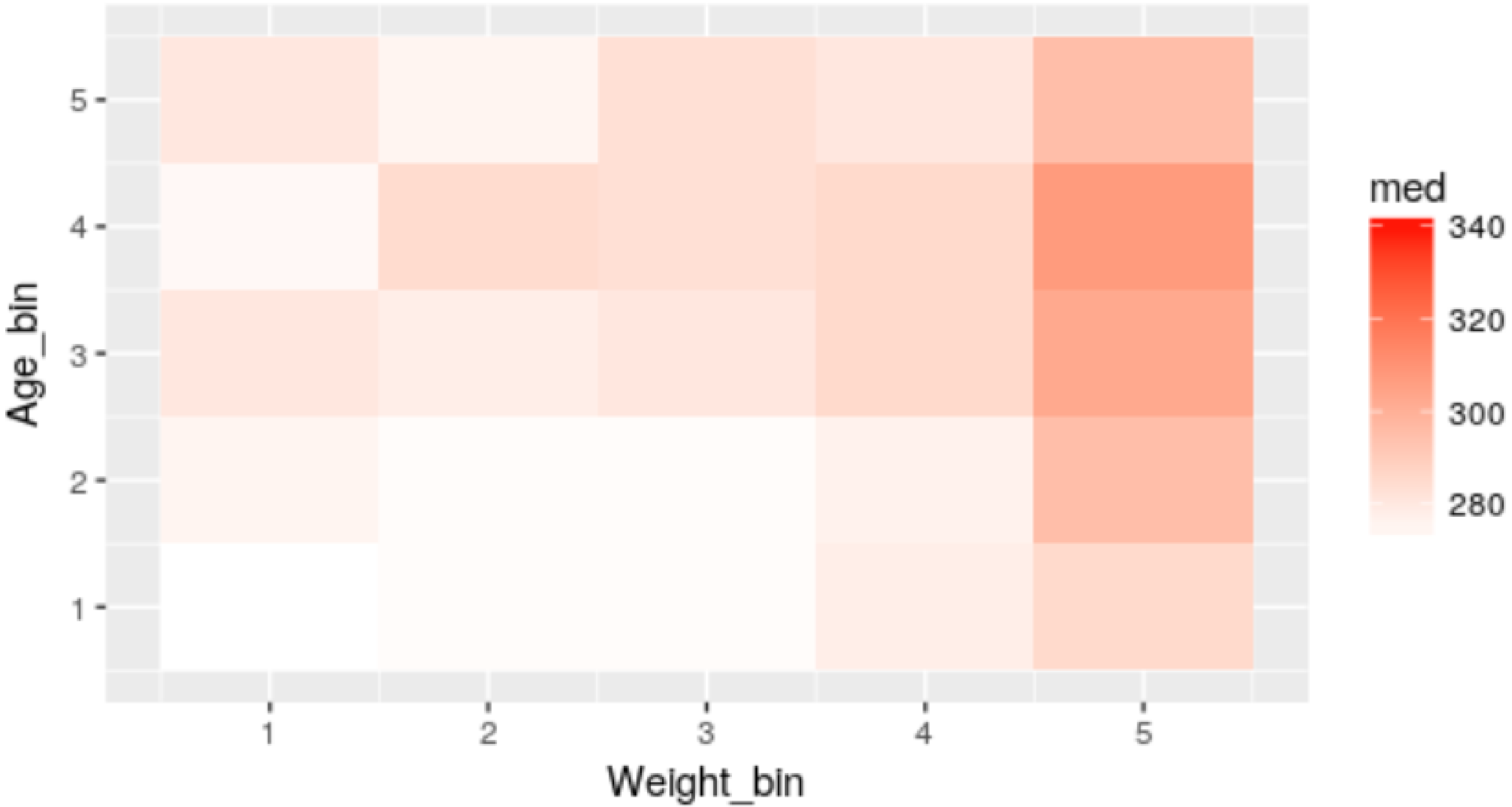
Histograms

```
ggplot(subset(powertidy,LiftSuccess == TRUE &
Lift == "Squat" &                DivisionClean ==
"Open" & Sex == 'M')) +  geom_histogram(mapping =
aes(x = BodyweightKg), stat = "bin", fill =
"blue", colour = "black")
```



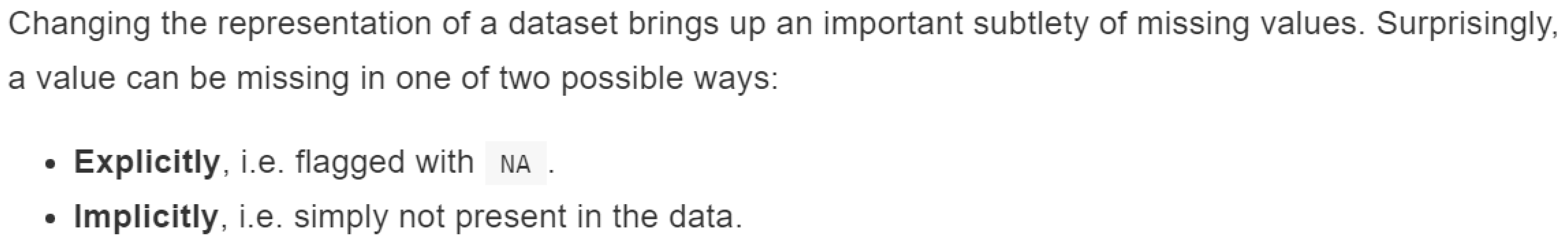
Heatmaps

```
powertidy %>% filter(Lift == "Squat" & LiftSuccess ==
TRUE & Age > 0 & Sex == 'M') %>%  mutate(Age_bin =
ntile(Age, 5), Weight_bin = ntile(BodyweightKg, 5),
Lift_bin = ntile(BestKg, 5)) %>%  filter(Lift_bin == 5)
%>%  group_by(Age_bin, Weight_bin) %>% summarise(n = n(),
med = median(BestKg)) %>%  ggplot(aes(x = Weight_bin, y =
Age_bin)) + geom_tile(aes(fill = med)) +
scale_fill_gradient(low = "white",high = "red")
```



Timeseries data

Missing Values



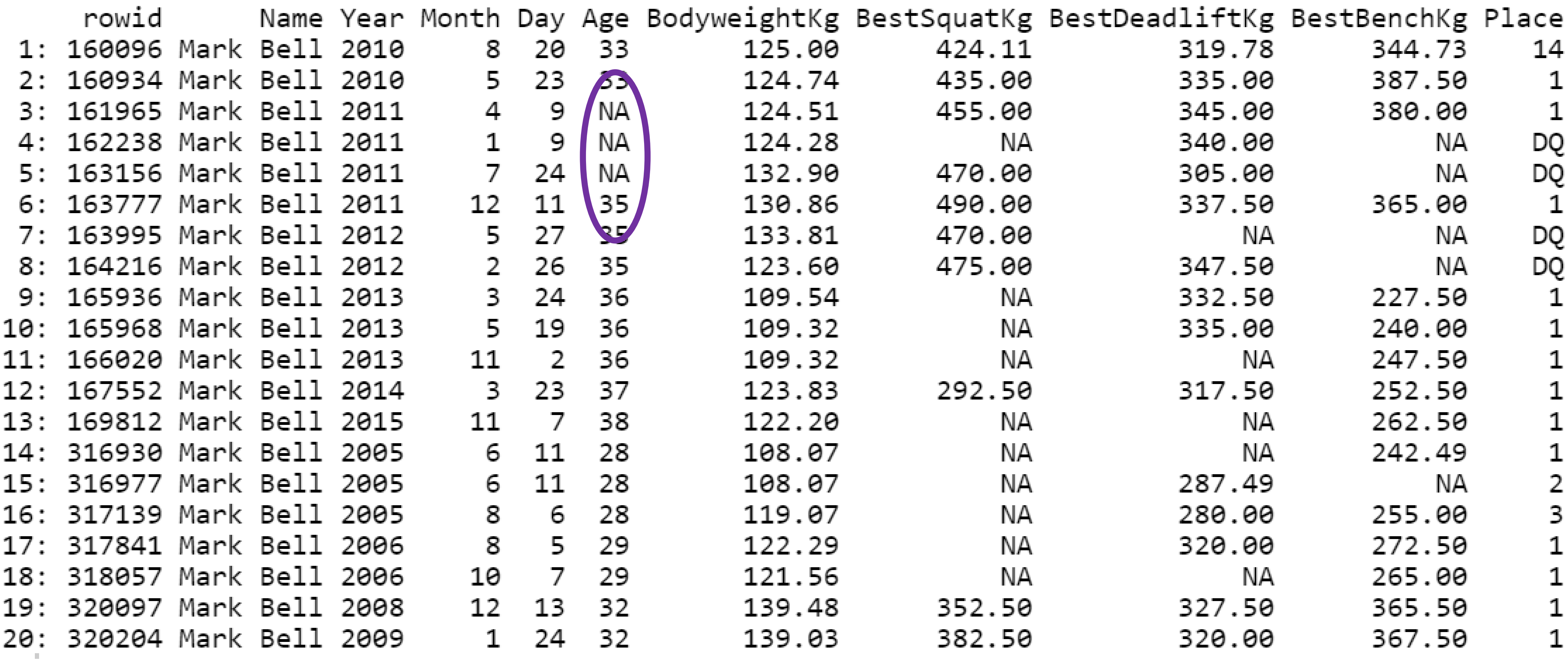
Changing the representation of a dataset brings up an important subtlety of missing values. Surprisingly, a value can be missing in one of two possible ways:

- **Explicitly**, i.e. flagged with `NA`.
- **Implicitly**, i.e. simply not present in the data.

https://r4ds.had.co.nz/tidy-data.html

Back to me

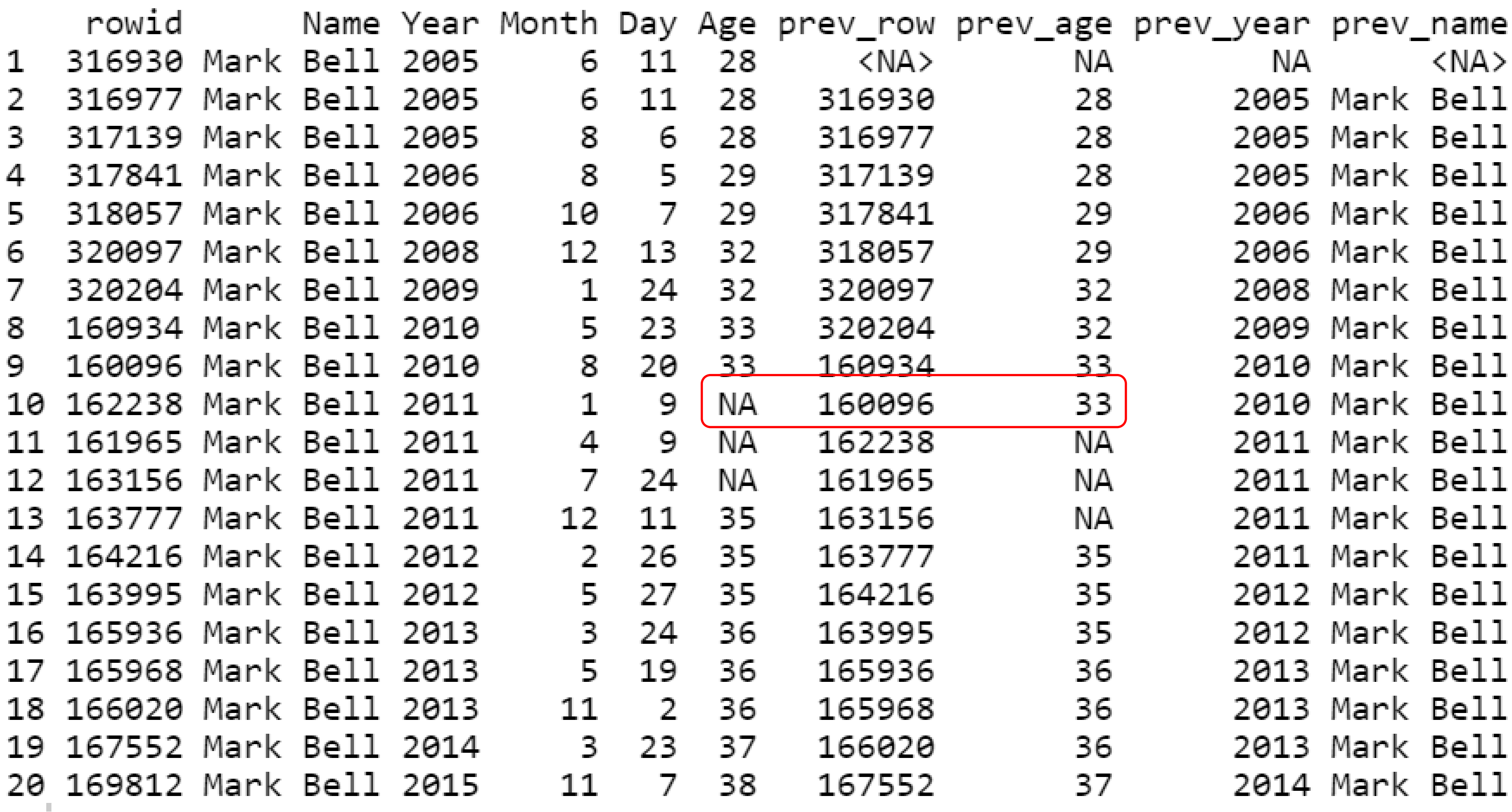
```
mb <- subset(power_meet_data, Name == "Mark Bell",
c("rowid", "Name",
"Year","Month","Day","Age","BodyweightKg",
"BestSquatKg","BestDeadliftKg",  "BestBenchKg","Place"))
```



```
     rowid      Name Year Month Day Age BodyweightKg BestSquatKg BestDeadliftKg BestBenchKg Place
 1: 160096 Mark Bell 2010     8  20  33       125.00      424.11         319.78      344.73    14
 2: 160934 Mark Bell 2010     5  23  33       124.74      435.00         335.00      387.50     1
 3: 161965 Mark Bell 2011     4   9  NA       124.51      455.00         345.00      380.00     1
 4: 162238 Mark Bell 2011     1   9  NA       124.28          NA         340.00          NA    DQ
 5: 163156 Mark Bell 2011     7  24  NA       132.90      470.00         305.00          NA    DQ
 6: 163777 Mark Bell 2011    12  11  35       130.86      490.00         337.50      365.00     1
 7: 163995 Mark Bell 2012     5  27  35       133.81      470.00             NA          NA    DQ
 8: 164216 Mark Bell 2012     2  26  35       123.60      475.00         347.50          NA    DQ
 9: 165936 Mark Bell 2013     3  24  36       109.54          NA         332.50      227.50     1
10: 165968 Mark Bell 2013     5  19  36       109.32          NA         335.00      240.00     1
11: 166020 Mark Bell 2013    11   2  36       109.32          NA             NA      247.50     1
12: 167552 Mark Bell 2014     3  23  37       123.83      292.50         317.50      252.50     1
13: 169812 Mark Bell 2015    11   7  38       122.20          NA             NA      262.50     1
14: 316930 Mark Bell 2005     6  11  28       108.07          NA             NA      242.49     1
15: 316977 Mark Bell 2005     6  11  28       108.07          NA         287.49          NA     2
16: 317139 Mark Bell 2005     8   6  28       119.07          NA         280.00      255.00     3
17: 317841 Mark Bell 2006     8   5  29       122.29          NA         320.00      272.50     1
18: 318057 Mark Bell 2006    10   7  29       121.56          NA             NA      265.00     1
19: 320097 Mark Bell 2008    12  13  32       139.48      352.50         327.50      365.50     1
20: 320204 Mark Bell 2009     1  24  32       139.03      382.50         320.00      367.50     1
```

Back to the previous

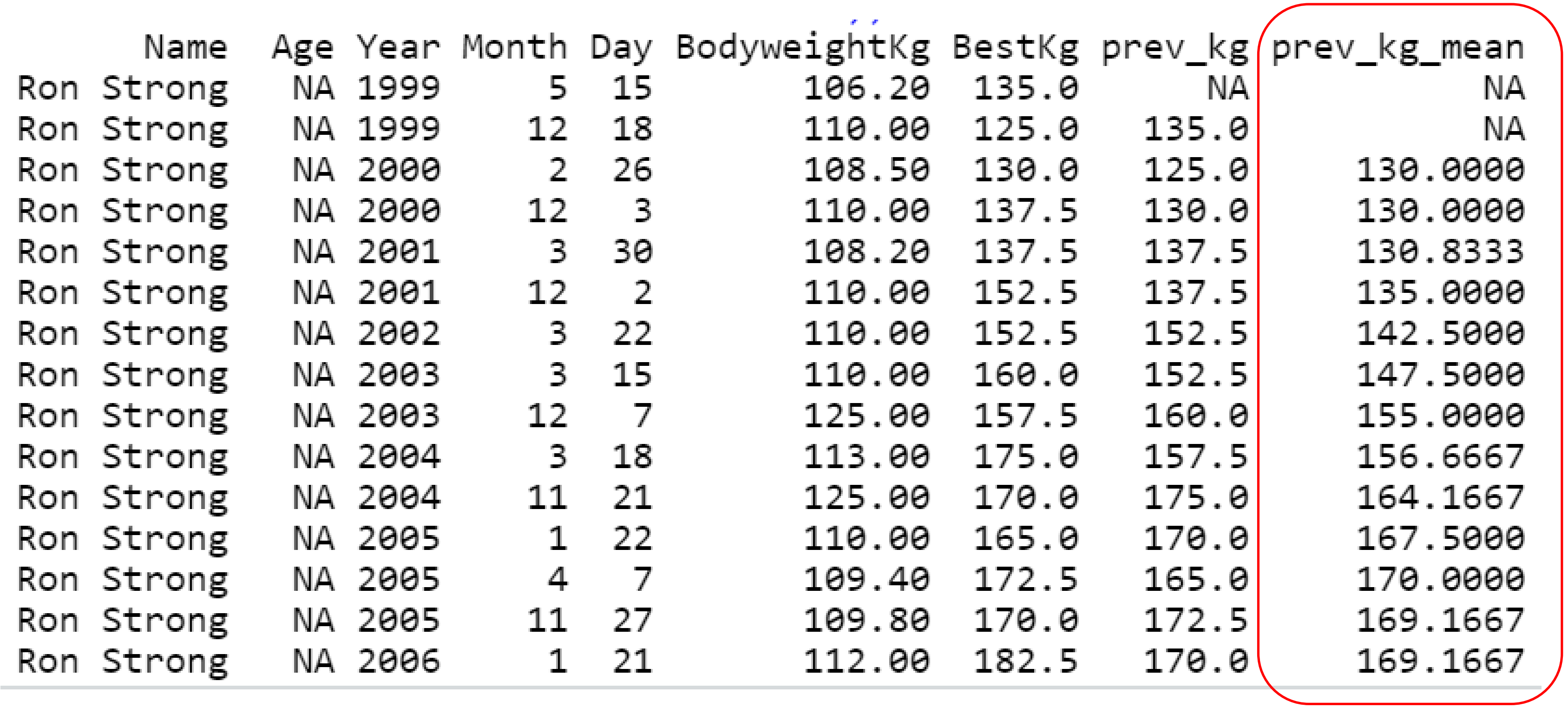
```
mb %>% select(rowid, Name, Year, Month, Day, Age) %>%
arrange( Name, Year, Month, Day) %>%  mutate(prev_row =
lag(rowid), prev_age = lag(Age), prev_year = lag(Year),
prev_name = lag(Name))
```



```
    rowid      Name Year Month Day Age prev_row prev_age prev_year prev_name
1  316930 Mark Bell 2005     6  11  28     <NA>       NA        NA      <NA>
2  316977 Mark Bell 2005     6  11  28   316930       28      2005 Mark Bell
3  317139 Mark Bell 2005     8   6  28   316977       28      2005 Mark Bell
4  317841 Mark Bell 2006     8   5  29   317139       28      2005 Mark Bell
5  318057 Mark Bell 2006    10   7  29   317841       29      2006 Mark Bell
6  320097 Mark Bell 2008    12  13  32   318057       29      2006 Mark Bell
7  320204 Mark Bell 2009     1  24  32   320097       32      2008 Mark Bell
8  160934 Mark Bell 2010     5  23  33   320204       32      2009 Mark Bell
9  160096 Mark Bell 2010     8  20  33   160934       33      2010 Mark Bell
10 162238 Mark Bell 2011     1   9  NA   160096       33      2010 Mark Bell
11 161965 Mark Bell 2011     4   9  NA   162238       NA      2011 Mark Bell
12 163156 Mark Bell 2011     7  24  NA   161965       NA      2011 Mark Bell
13 163777 Mark Bell 2011    12  11  35   163156       NA      2011 Mark Bell
14 164216 Mark Bell 2012     2  26  35   163777       35      2011 Mark Bell
15 163995 Mark Bell 2012     5  27  35   164216       35      2012 Mark Bell
16 165936 Mark Bell 2013     3  24  36   163995       35      2012 Mark Bell
17 165968 Mark Bell 2013     5  19  36   165936       36      2013 Mark Bell
18 166020 Mark Bell 2013    11   2  36   165968       36      2013 Mark Bell
19 167552 Mark Bell 2014     3  23  37   166020       36      2013 Mark Bell
20 169812 Mark Bell 2015    11   7  38   167552       37      2014 Mark Bell
```

Strong by name…

```
powertidy %>% filter(Name == "Ron Strong" & Lift ==
"Bench" & LiftSuccess == TRUE) %>%  select(rowid, Name,
Age, Year, Month, Day, BodyweightKg, BestKg) %>%
arrange( Name, Year, Month, Day) %>%  mutate(prev_kg =
lag(BestKg, 1)) %>%  mutate(prev_kg_mean = rollapply(data
= prev_kg,                                         width =
3,                                         FUN = mean,
align = "right",
fill = NA,                                         na.rm =
T))
```

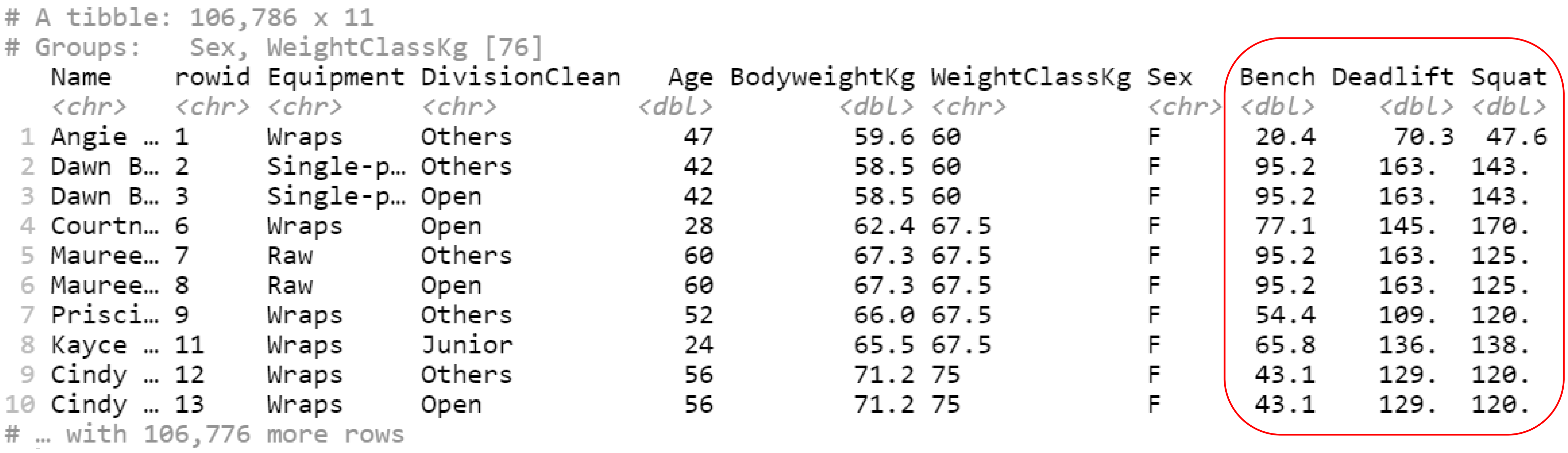


```
       Name   Age Year Month Day BodyweightKg BestKg prev_kg prev_kg_mean
Ron Strong     NA 1999     5  15       106.20  135.0      NA           NA
Ron Strong     NA 1999    12  18       110.00  125.0   135.0           NA
Ron Strong     NA 2000     2  26       108.50  130.0   125.0     130.0000
Ron Strong     NA 2000    12   3       110.00  137.5   130.0     130.0000
Ron Strong     NA 2001     3  30       108.20  137.5   137.5     130.8333
Ron Strong     NA 2001    12   2       110.00  152.5   137.5     135.0000
Ron Strong     NA 2002     3  22       110.00  152.5   152.5     142.5000
Ron Strong     NA 2003     3  15       110.00  160.0   152.5     147.5000
Ron Strong     NA 2003    12   7       125.00  157.5   160.0     155.0000
Ron Strong     NA 2004     3  18       113.00  175.0   157.5     156.6667
Ron Strong     NA 2004    11  21       125.00  170.0   175.0     164.1667
Ron Strong     NA 2005     1  22       110.00  165.0   170.0     167.5000
Ron Strong     NA 2005     4   7       109.40  172.5   165.0     170.0000
Ron Strong     NA 2005    11  27       109.80  170.0   172.5     169.1667
Ron Strong     NA 2006     1  21       112.00  182.5   170.0     169.1667
```

Preparing for machine learning

Untidy data

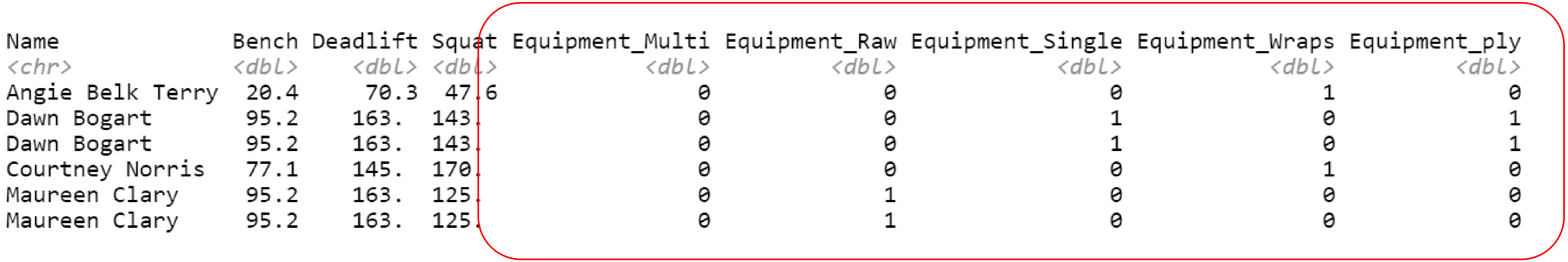
```
power_untidy <- spread(subset(powertidy_norm, LiftSuccess
== TRUE,
c("Name","rowid","Equipment","DivisionClean", "Age",
"BodyweightKg", "WeightClassKg","Sex", "Lift",
"BestKg")),                              key = Lift, value =
BestKg)power_untidy <-
power_untidy[complete.cases(power_untidy),]
```



```
# A tibble: 106,786 x 11
# Groups:   Sex, WeightClassKg [76]
   Name      rowid Equipment DivisionClean   Age BodyweightKg WeightClassKg Sex   Bench Deadlift Squat
   <chr>     <chr> <chr>     <chr>         <dbl>        <dbl> <chr>         <chr> <dbl>    <dbl> <dbl>
 1 Angie …   1     Wraps     Others           47         59.6 60            F      20.4     70.3  47.6
 2 Dawn B…   2     Single-p… Others           42         58.5 60            F      95.2    163.  143. 
 3 Dawn B…   3     Single-p… Open             42         58.5 60            F      95.2    163.  143. 
 4 Courtn…   6     Wraps     Open             28         62.4 67.5          F      77.1    145.  170. 
 5 Mauree…   7     Raw       Others           60         67.3 67.5          F      95.2    163.  125. 
 6 Mauree…   8     Raw       Open             60         67.3 67.5          F      95.2    163.  125. 
 7 Prisci…   9     Wraps     Others           52         66.0 67.5          F      54.4    109.  120. 
 8 Kayce …   11    Wraps     Junior           24         65.5 67.5          F      65.8    136.  138. 
 9 Cindy …   12    Wraps     Others           56         71.2 75            F      43.1    129.  120. 
10 Cindy …   13    Wraps     Open             56         71.2 75            F      43.1    129.  120. 
# … with 106,776 more rows
```

Categorical variables: One hot encoding

```
power_untidy <- power_untidy %>%
separate_rows(Equipment) %>% mutate(count = 1) %>%
spread(Equipment, count, fill = 0, sep = "_")
```

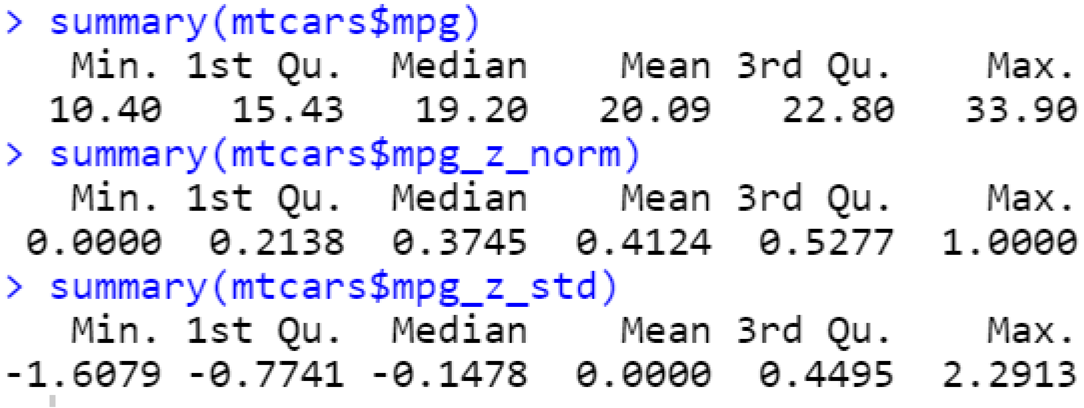


```
Name                Bench Deadlift Squat Equipment_Multi Equipment_Raw Equipment_Single Equipment_Wraps Equipment_ply
<chr>               <dbl>    <dbl> <dbl>           <dbl>         <dbl>            <dbl>           <dbl>         <dbl>
Angie Belk Terry     20.4     70.3  47.6               0             0                0               1             0
Dawn Bogart          95.2    163.  143.                0             0                1               0             1
Dawn Bogart          95.2    163.  143.                0             0                1               0             1
Courtney Norris      77.1    145.  170.                0             0                0               1             0
Maureen Clary        95.2    163.  125.                0             1                0               0             0
Maureen Clary        95.2    163.  125.                0             1                0               0             0
```

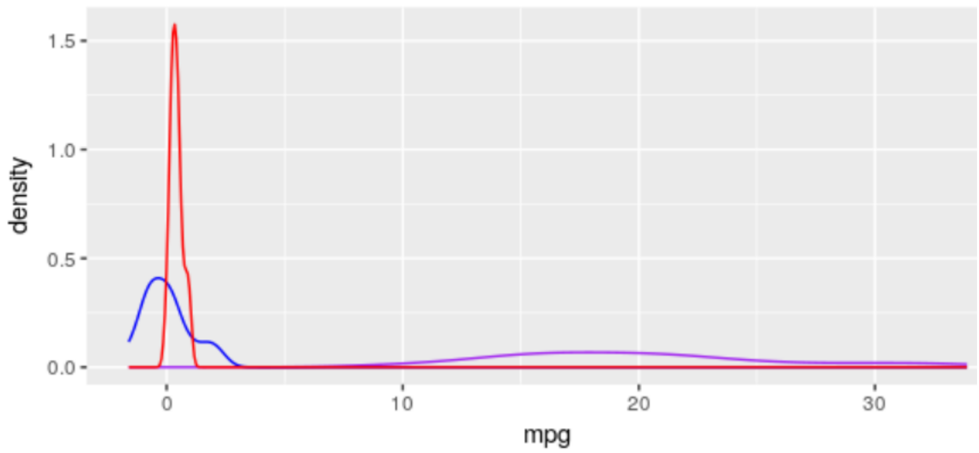
Normalisation vs. Standardisation (Subtle change of data!)

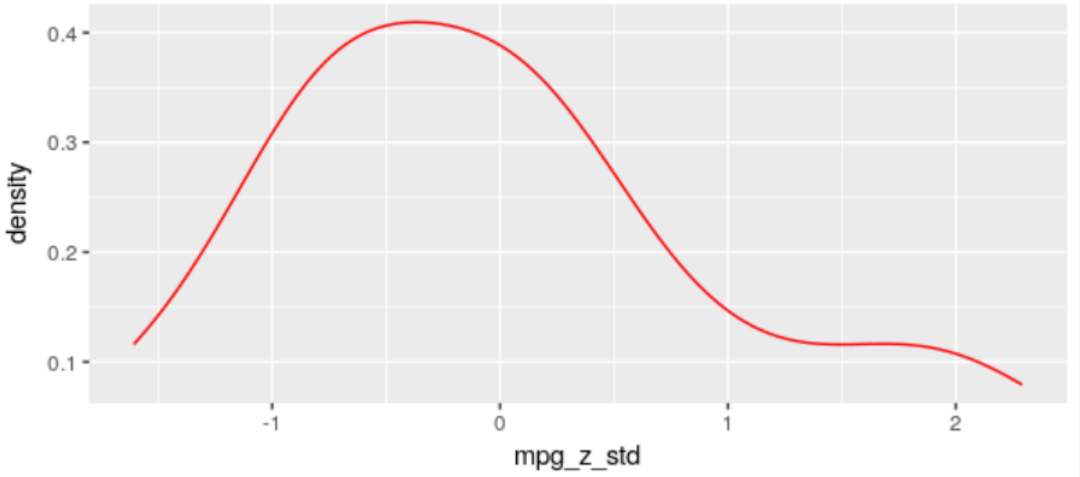
```
# Standardisation
mtcars <- mtcars %>% mutate_each(funs(z_std = (. -
mean(.))/sd(.)))

# Normalisation
mtcars <- mtcars %>% mutate_each(funs(z_norm = (. -
min(.))/(max(.)-min(.))))
```



```
> summary(mtcars$mpg)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  10.40   15.43   19.20   20.09   22.80   33.90 
> summary(mtcars$mpg_z_norm)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.0000  0.2138  0.3745  0.4124  0.5277  1.0000 
> summary(mtcars$mpg_z_std)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
-1.6079 -0.7741 -0.1478  0.0000  0.4495  2.2913 
```

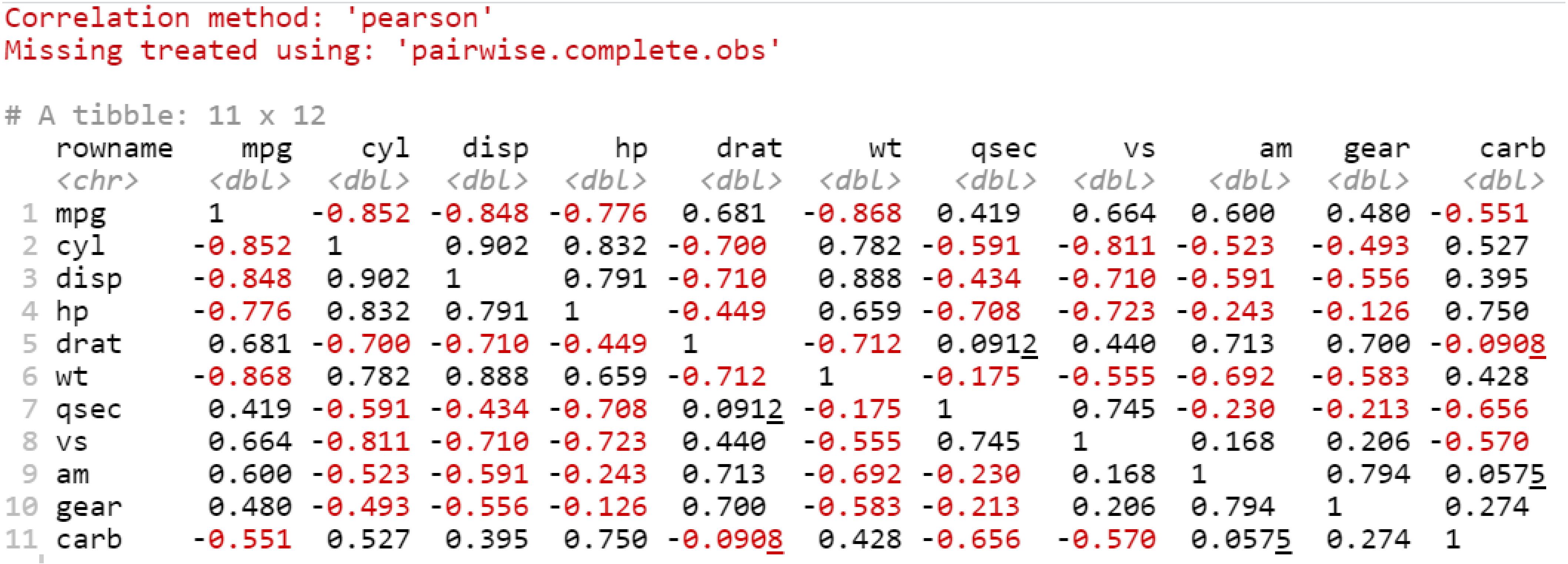




The stats bit

Correlations

```
mtcars %>% correlate(diagonal = 1)
```

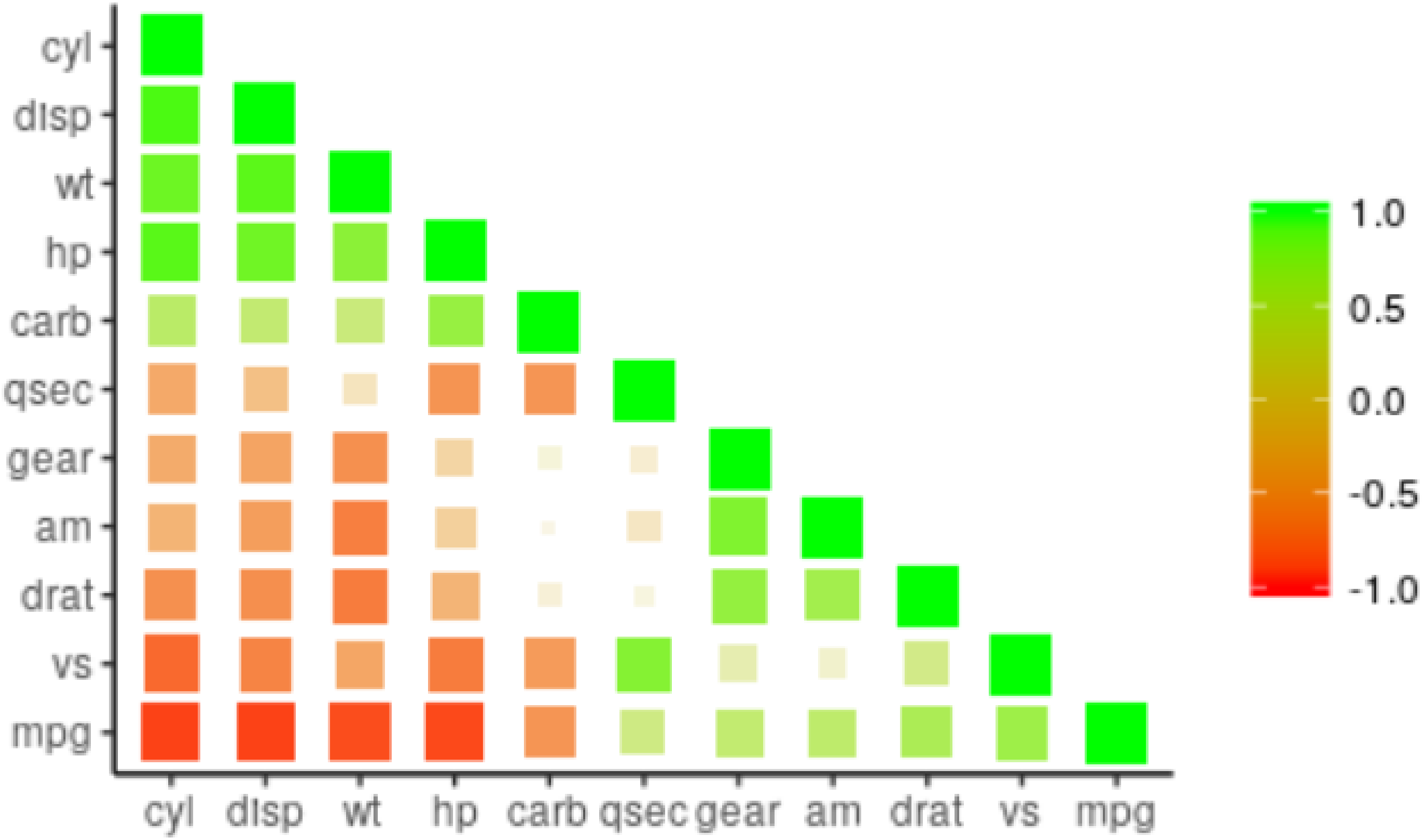


```
Correlation method: 'pearson'
Missing treated using: 'pairwise.complete.obs'

# A tibble: 11 x 12
   rowname    mpg    cyl   disp     hp    drat      wt    qsec     vs      am   gear    carb
   <chr>    <dbl>  <dbl>  <dbl>  <dbl>   <dbl>   <dbl>   <dbl>  <dbl>   <dbl>  <dbl>   <dbl>
 1 mpg      1     -0.852 -0.848 -0.776  0.681  -0.868   0.419   0.664  0.600   0.480 -0.551 
 2 cyl     -0.852  1      0.902  0.832 -0.700   0.782  -0.591  -0.811 -0.523  -0.493  0.527 
 3 disp    -0.848  0.902  1      0.791 -0.710   0.888  -0.434  -0.710 -0.591  -0.556  0.395 
 4 hp      -0.776  0.832  0.791  1     -0.449   0.659  -0.708  -0.723 -0.243  -0.126  0.750 
 5 drat     0.681 -0.700 -0.710 -0.449  1      -0.712   0.0912  0.440  0.713   0.700 -0.0908
 6 wt      -0.868  0.782  0.888  0.659 -0.712   1      -0.175  -0.555 -0.692  -0.583  0.428 
 7 qsec     0.419 -0.591 -0.434 -0.708  0.0912 -0.175   1       0.745 -0.230  -0.213 -0.656 
 8 vs       0.664 -0.811 -0.710 -0.723  0.440  -0.555   0.745   1      0.168   0.206 -0.570 
 9 am       0.600 -0.523 -0.591 -0.243  0.713  -0.692  -0.230   0.168  1       0.794  0.0575
10 gear     0.480 -0.493 -0.556 -0.126  0.700  -0.583  -0.213   0.206  0.794   1      0.274 
11 carb    -0.551  0.527  0.395  0.750 -0.0908  0.428  -0.656  -0.570  0.0575  0.274  1     
```

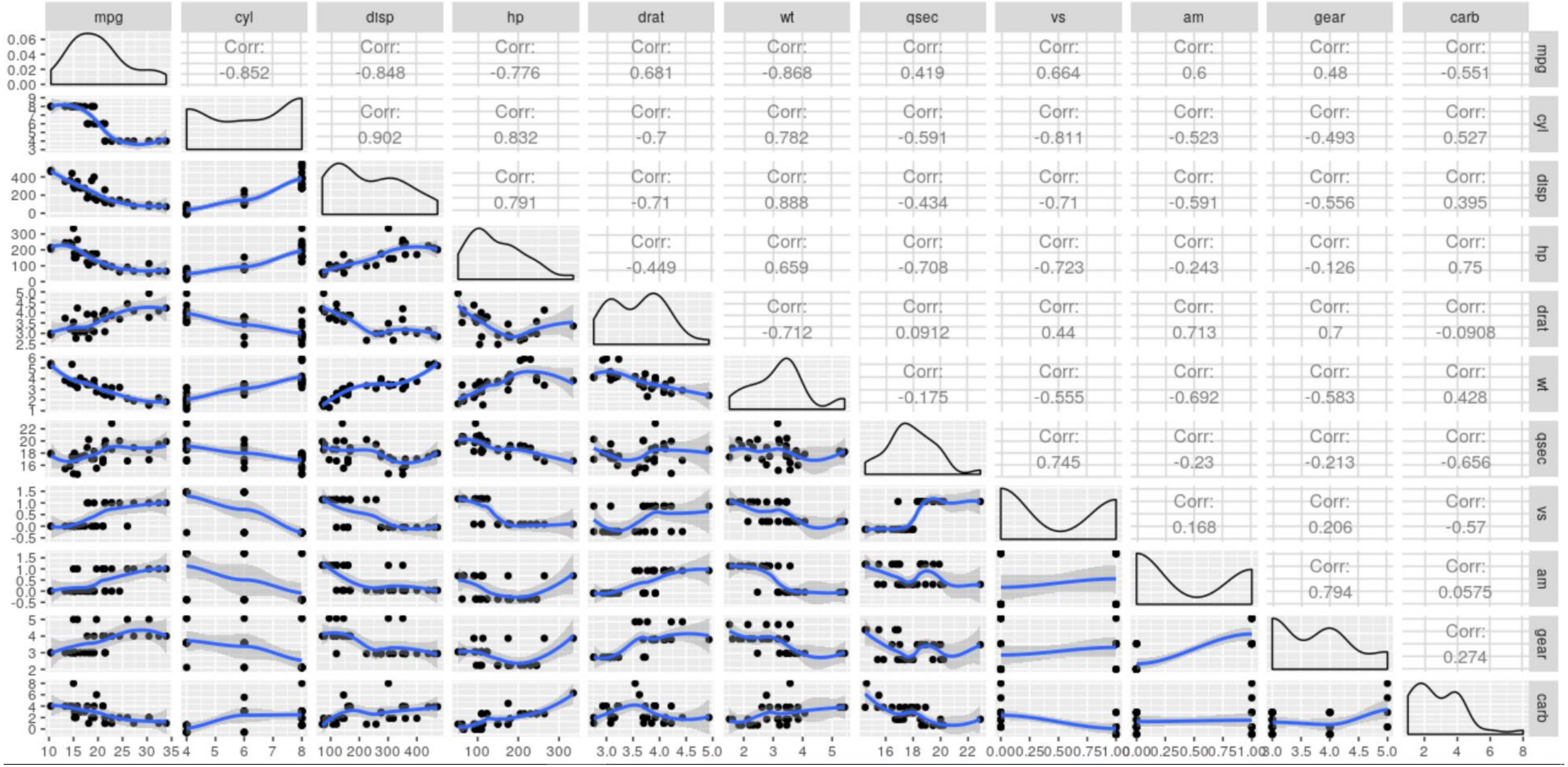
Correlations

```
mtcars %>%   correlate(method = 'spearman', diagonal = 1)
%>%rearrange(method = "MDS", absolute = FALSE) %>%
shave() %>%   rplot(shape = 15, colors = c("red",
"green"))
```



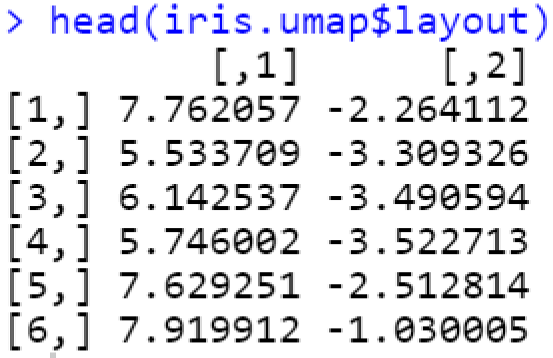
Correlations

```
ggpairs(mtcars)
```



Dimensionality Reduction

```
cars.data = mtcars[,names(mtcars) != "cyl"]cars.labels =
mtcars[,"cyl"]cars.umap =
umap(cars.data)head(iris.umap$layout)
```



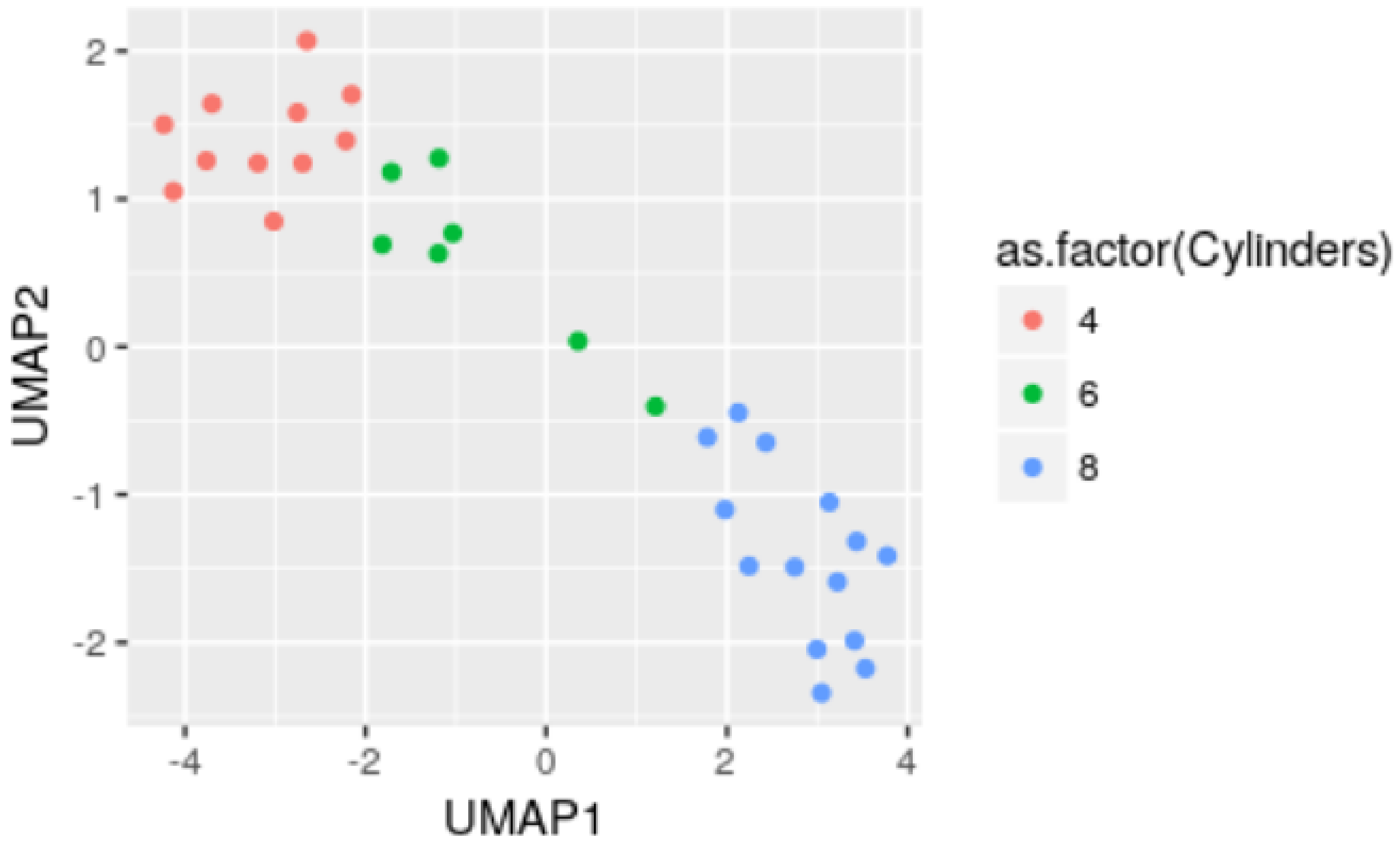
```
> head(iris.umap$layout)
         [,1]      [,2]
[1,] 7.762057 -2.264112
[2,] 5.533709 -3.309326
[3,] 6.142537 -3.490594
[4,] 5.746002 -3.522713
[5,] 7.629251 -2.512814
[6,] 7.919912 -1.030005
```

Other algorithms are available:

- PCA
- T-SNE
- UMAP

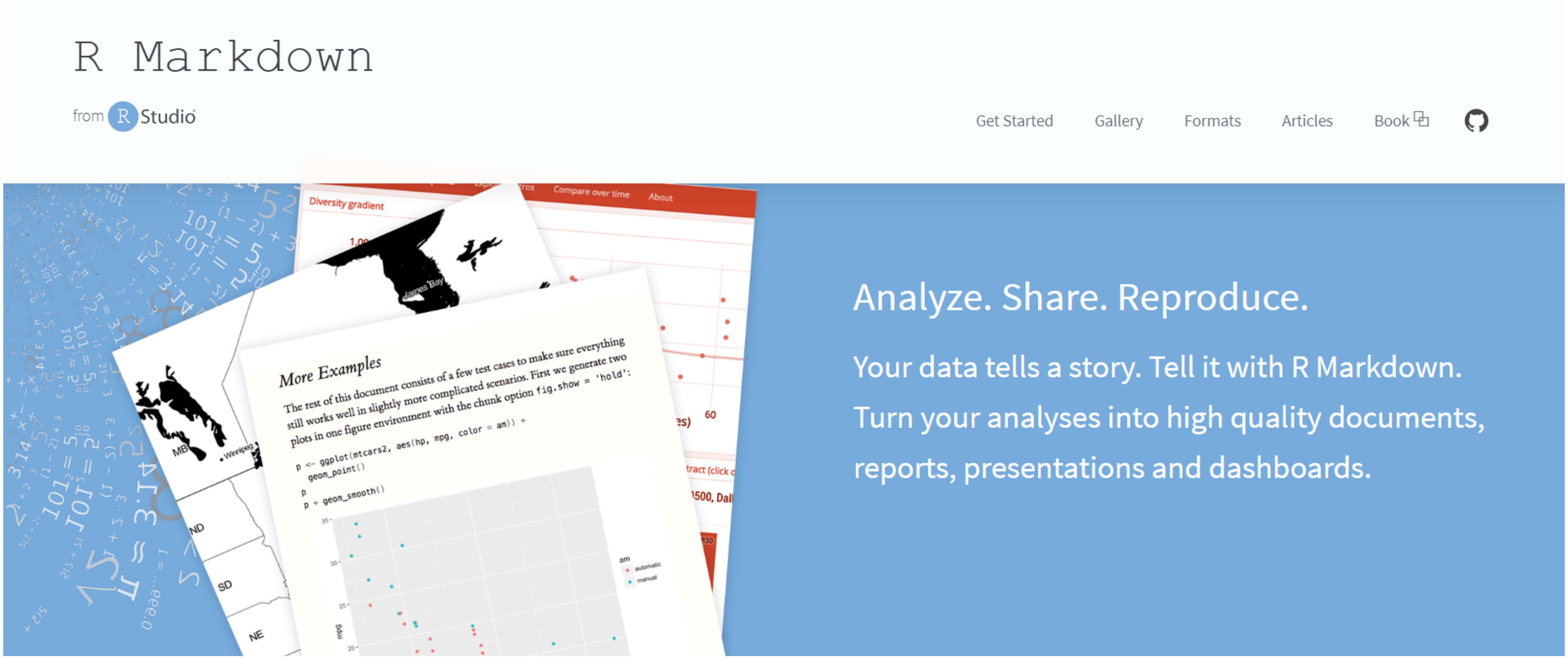
Dimensionality Reduction

```
as_tibble(cars.umap$layout, .name_repair = "universal")
%>% rename(UMAP1 = 1, UMAP2 = 2) %>%   mutate(Cylinders =
mtcars$cyl) %>%  ggplot(aes(UMAP1, UMAP2, color =
as.factor(Cylinders))) + geom_point()
```



Almost finished

Sharing your analysis



Analyze. Share. Reproduce.

Your data tells a story. Tell it with R Markdown.

Turn your analyses into high quality documents, reports, presentations and dashboards.

More Examples

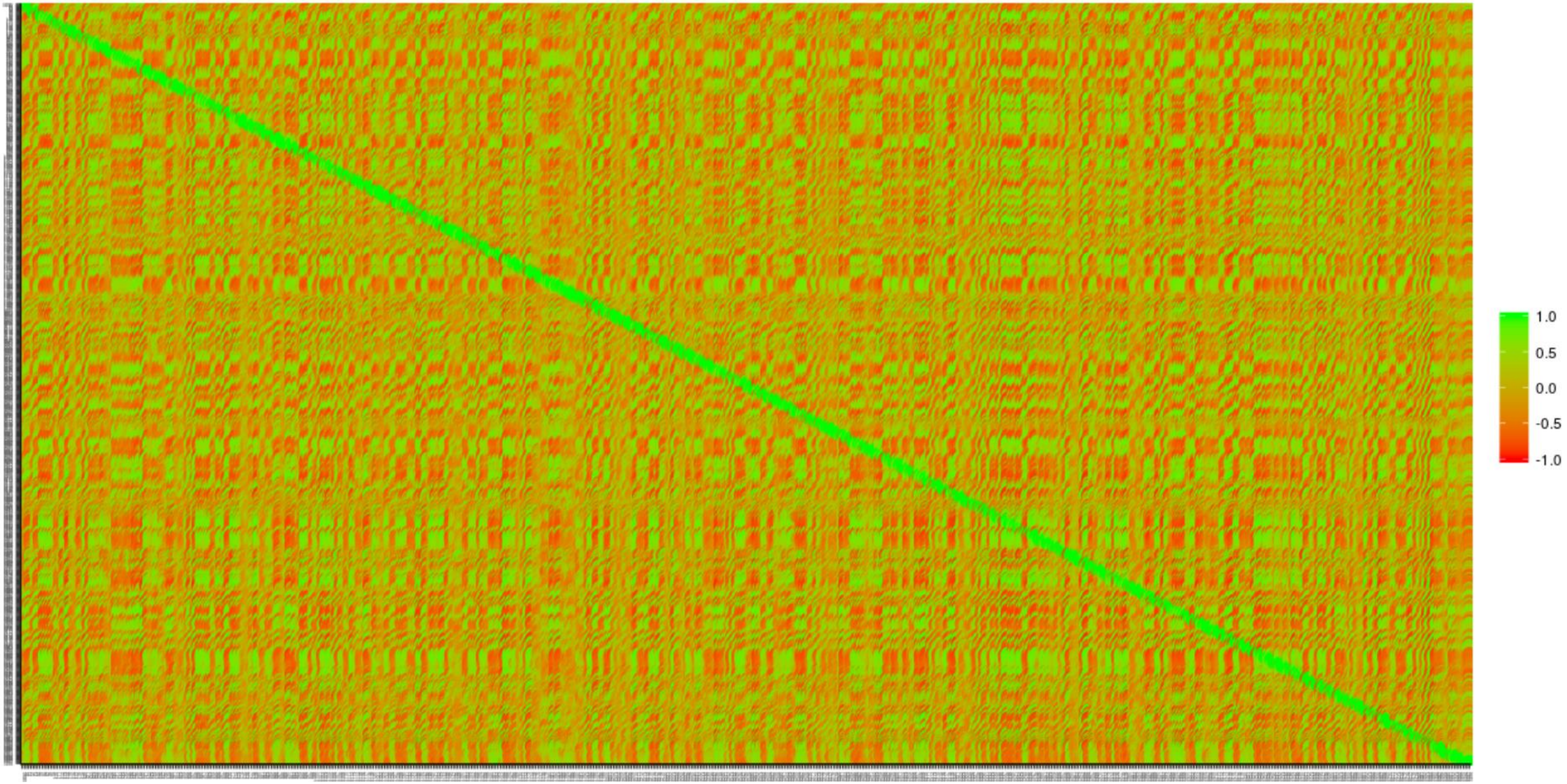
The rest of this document consists of a few test cases to make sure everything still works well in slightly more complicated scenarios. First we generate two plots in one figure environment with the chunk option `fig.show = 'hold'`:

```
p <- ggplot(mtcars2, aes(hp, mpg, color = am)) +
  geom_point()
p
p + geom_smooth()
```

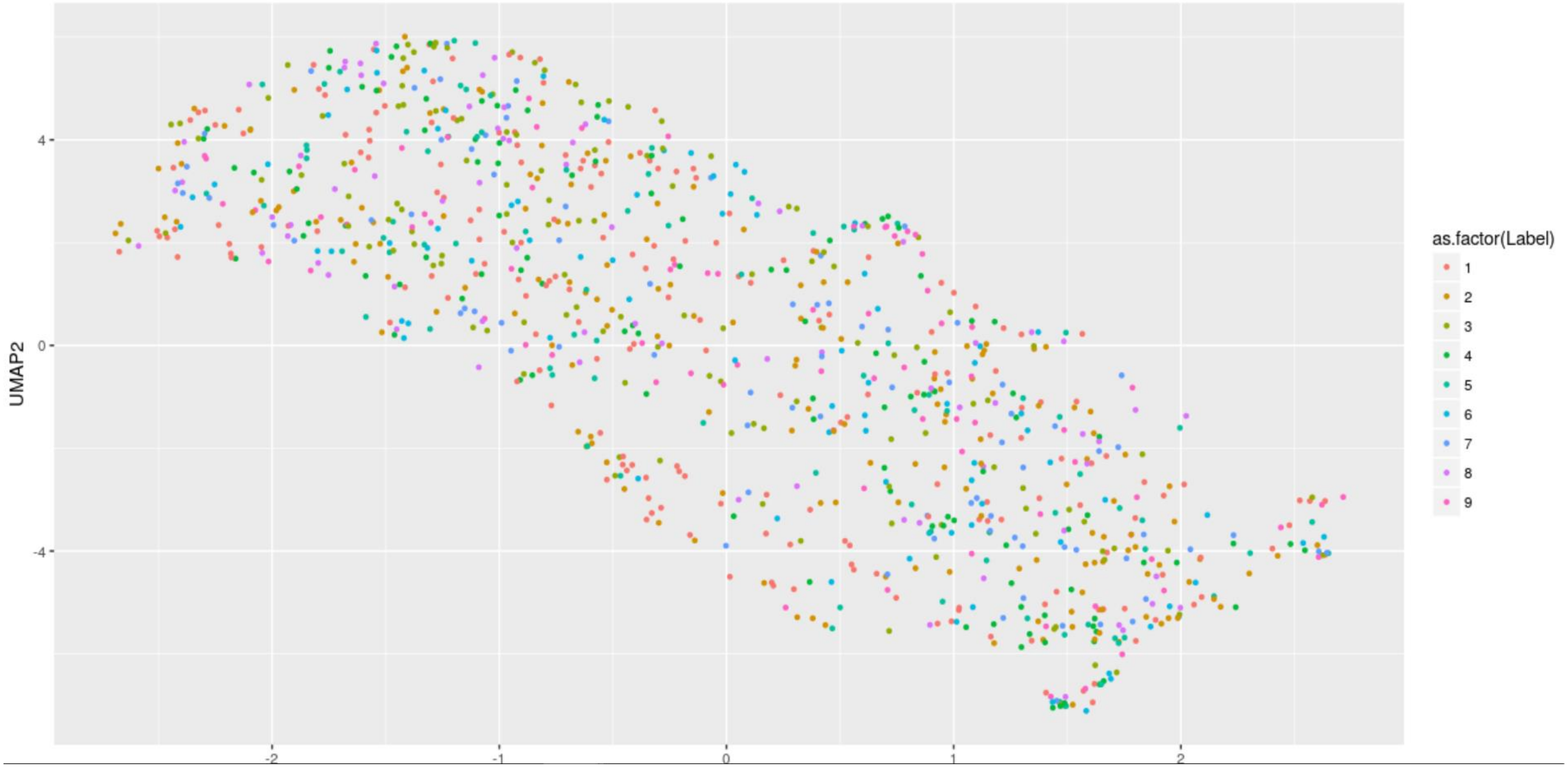
Summary

- **Data is rarely clean**
- **Tidy your data**
- **Visualise your data**
- **Know your numbers**
  - **High values; Low values; Missing values**
  - **Quartiles**
  - **Mean; Medians**
  - **Correlations**
- **Create your own features**
- **Go to Kaggle!**

Be thankful… 1000 rows x 1875 columns



UMAP not to the rescue



Thank You