PART-1: Standardization, Correlation and Handling Missing Values

Standardization

From the book, we were introduced to the concept of z-scores or standard scores. Mathematically, the z-score stands for the following value -

z = (value - mean)/standard deviation

The main advantage given by standardization is interpretability: Sometimes, there are no defined metrics to assess the meaning of a statistic. For example, in a survey that assesses a person’s grumpiness level based on the number of questions answered in a grumpy manner, it is difficult to interpret what a particular grumpiness score means. What does it mean to answer 35 out of 50 questions in a grumpy manner? Z-score gives us a solution to this problem. Now, based on your z-score, we can see how many standard deviations above or below the mean value your response lies.

Let us now try finding the standard scores of an arbitrary data set -

```
X = c(15, 16, 17, 18, 18, 19, 20)
z = (X - mean(X)) / sd(X)
z
```

```
## [1] -1.4965398 -0.9145521 -0.3325644  0.2494233  0.2494233  0.8314110  1.4133987
```

Correlation

Correlation helps us describe relationships between two variables. Understanding this through an example. Loading data -

```
library(psych)
setwd("C:/Personal/Academics/IITK Resources/Sem-7/BSE658/GitHub/BSE658/Module 3/Notebooks/")
load( "parenthood.Rdata" )
head(parenthood)
```

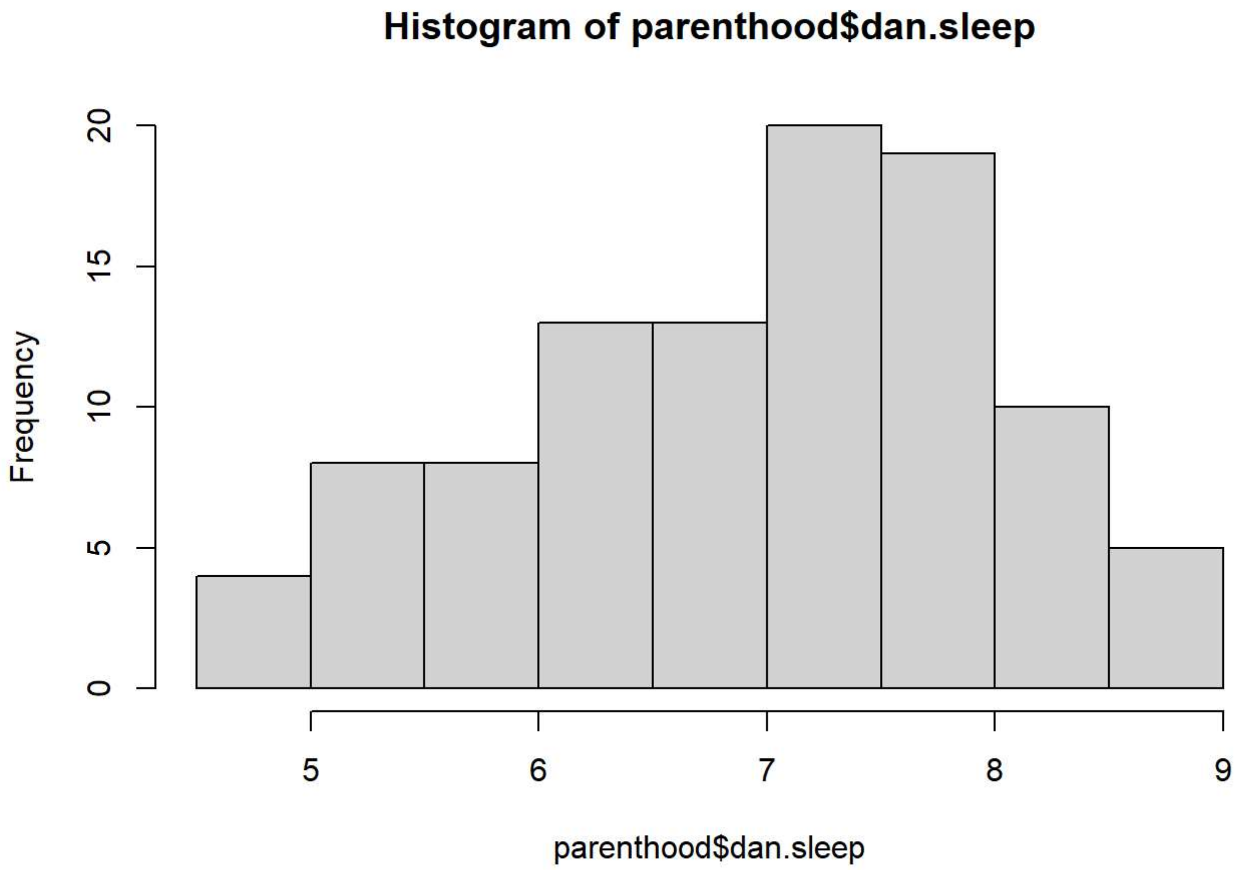
```
##   dan.sleep baby.sleep dan.grump day
## 1      7.59      10.18        56   1
## 2      7.91      11.66        60   2
## 3      5.14       7.92        82   3
## 4      7.71       9.61        55   4
## 5      6.68       9.75        67   5
## 6      5.99       5.04        72   6
```

```
describe(parenthood)
```

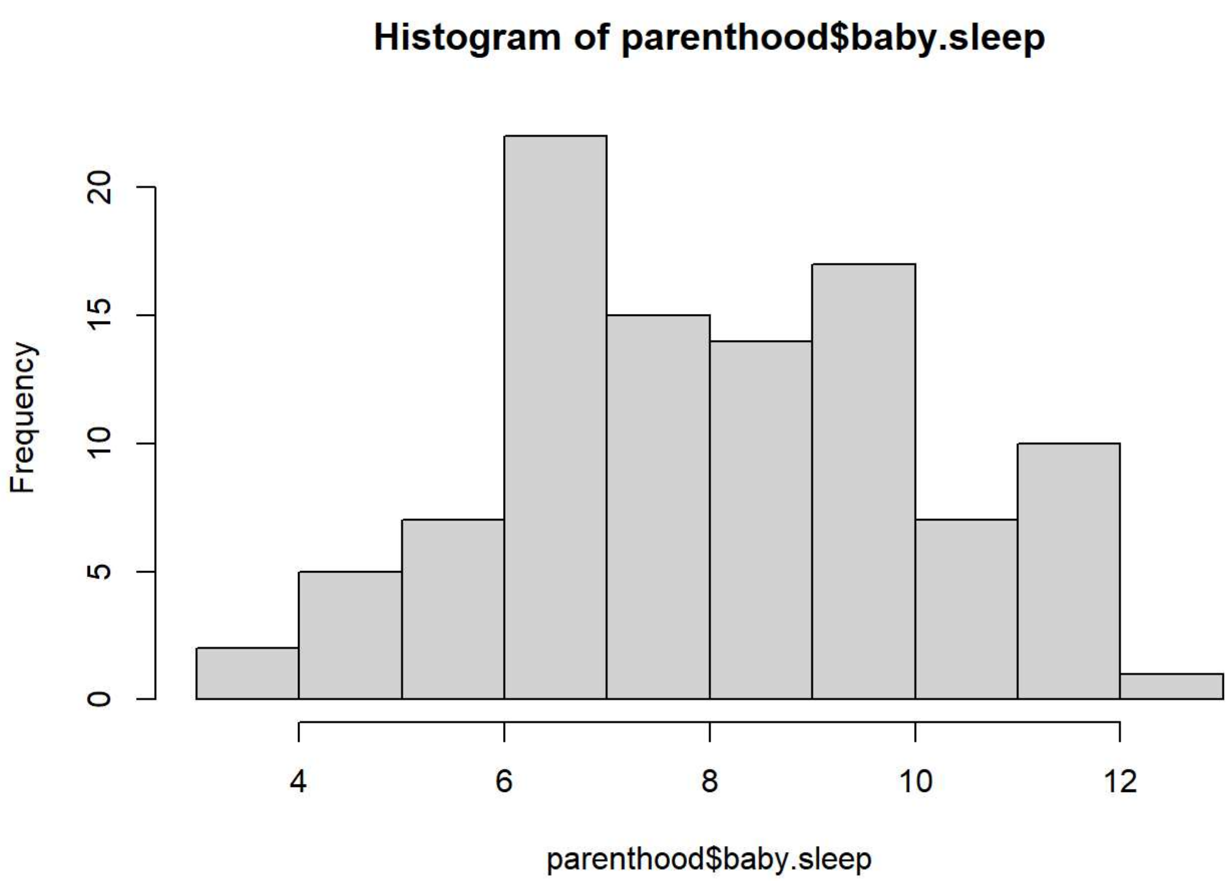
```
##            vars   n  mean    sd median trimmed   mad   min    max range  skew
## dan.sleep     1 100  6.97  1.02   7.03    7.00  1.09  4.84   9.00  4.16 -0.29
## baby.sleep    2 100  8.05  2.07   7.95    8.05  2.33  3.25  12.07  8.82 -0.02
## dan.grump     3 100 63.71 10.05  62.00   63.16  9.64 41.00  91.00 50.00  0.43
## day           4 100 50.50 29.01  50.50   50.50 37.06  1.00 100.00 99.00  0.00
##            kurtosis   se
## dan.sleep     -0.72 0.10
## baby.sleep    -0.69 0.21
## dan.grump     -0.16 1.00
## day           -1.24 2.90
```

Visualizing the data -

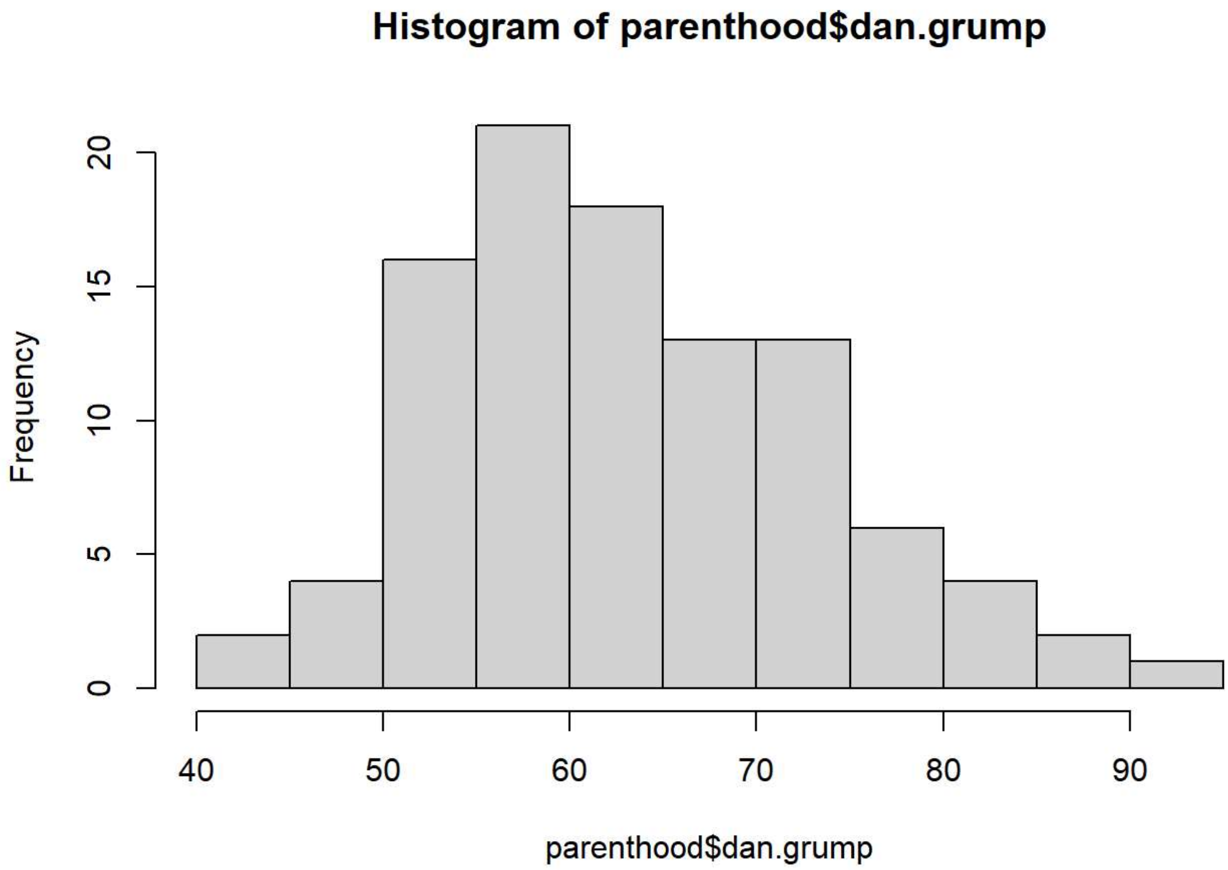
```
hist(parenthood$dan.sleep)
```




```
hist(parenthood$baby.sleep)
```




```
hist(parenthood$dan.grump)
```



Creating scatterplots using car package.

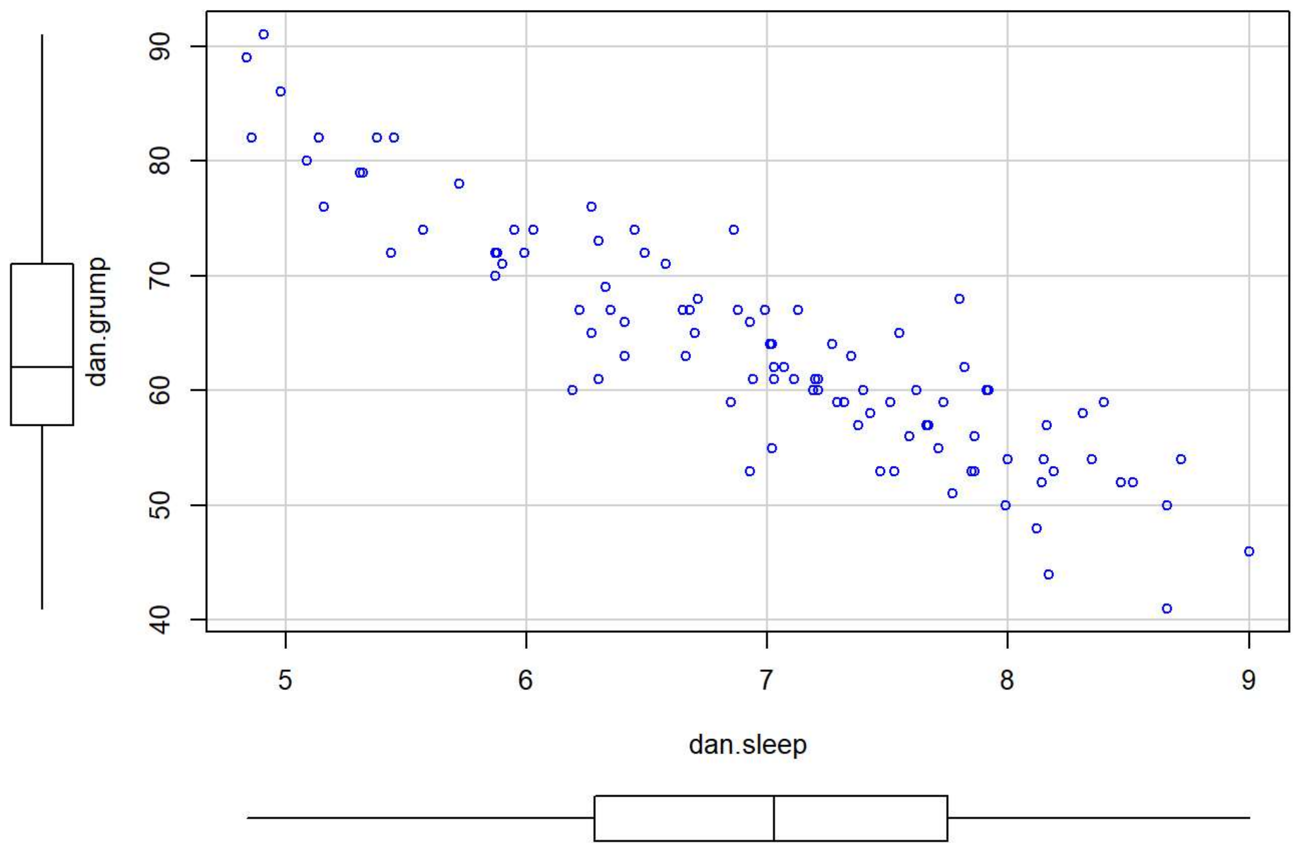
```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:psych':
##
##     logit
```

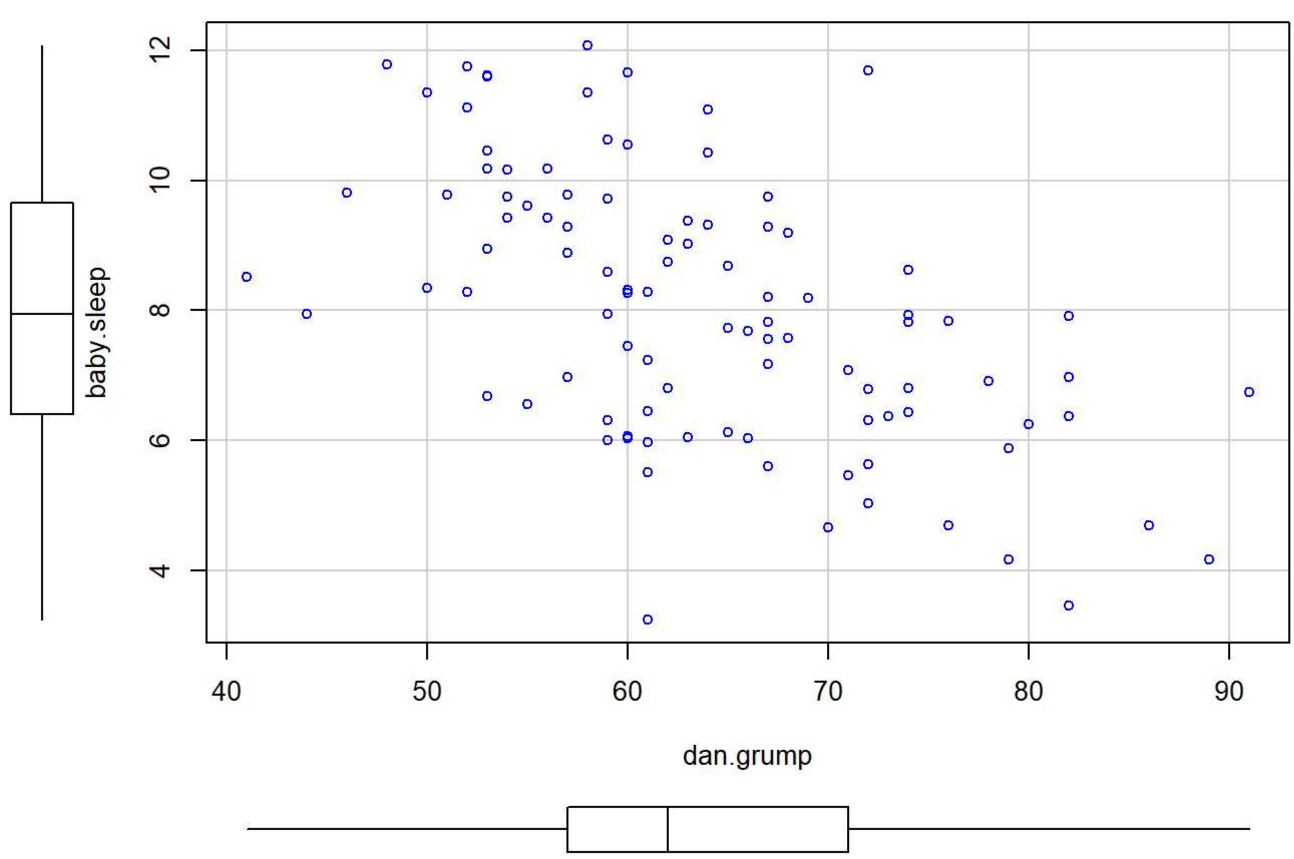
```
scatterplot( dan.grump ~ dan.sleep, data = parenthood, regLine = FALSE, smooth = FALSE)
```




```
scatterplot
```

```
## function (x, ...)
## {
##     UseMethod("scatterplot")
## }
## <bytecode: 0x000001501549b4c0>
## <environment: namespace:car>
```

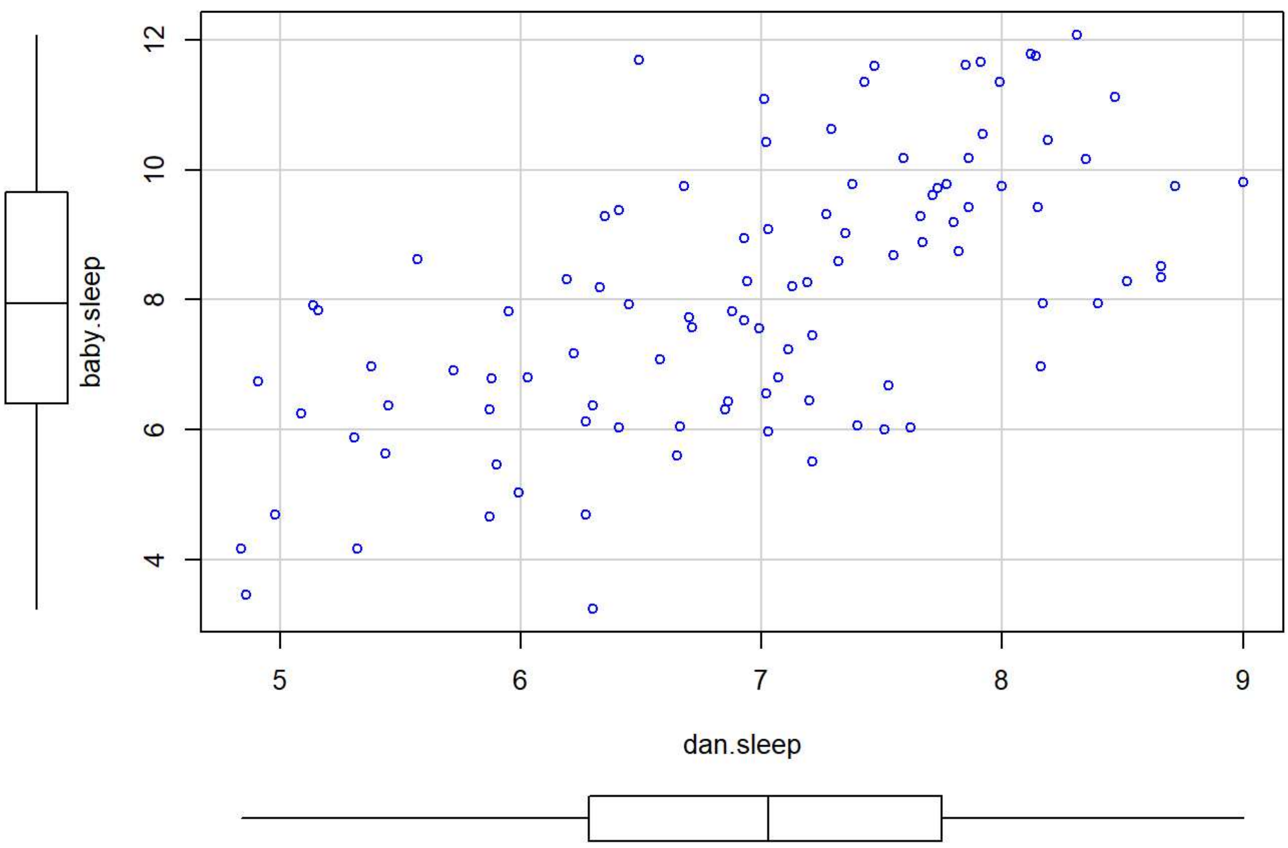
```
scatterplot( baby.sleep ~ dan.grump, data = parenthood, regLine = FALSE, smooth = FALSE)
```




```
scatterplot
```

```
## function (x, ...)
## {
##     UseMethod("scatterplot")
## }
## <bytecode: 0x000001501549b4c0>
## <environment: namespace:car>
```

```
scatterplot( baby.sleep ~ dan.sleep, data = parenthood, regLine = FALSE, smooth = FALSE)
```




```
scatterplot
```

```
## function (x, ...)
## {
##     UseMethod("scatterplot")
## }
## <bytecode: 0x000001501549b4c0>
## <environment: namespace:car>
```

From above plots, it is visible that dan's sleep and his grumpiness are related to each other and even his baby's sleep and his grump are related. However, it is also clear that dan's sleep and his grumpiness are better related than the baby's sleep and dan's grump. This can be quantified using the correlation coefficient (r). This is closely related to the notion of covariance. Let us try finding correlation coefficient values too now.

```
cor(x = parenthood$dan.sleep, y = parenthood$dan.grump)
```

```
## [1] -0.903384
```

Hence, Dan's sleep and Dan's grumpiness have a strong negative correlation. There is a better way to see the correlation between all types of values in a dataframe.

```
cor(parenthood)
```

```
##              dan.sleep  baby.sleep   dan.grump         day
## dan.sleep   1.00000000  0.62794934 -0.90338404 -0.09840768
## baby.sleep  0.62794934  1.00000000 -0.56596373 -0.01043394
## dan.grump  -0.90338404 -0.56596373  1.00000000  0.07647926
## day        -0.09840768 -0.01043394  0.07647926  1.00000000
```

Now let's take a look at this data called "Anscombe's Quartet"

```
load( "anscombesquartet.Rdata" )
cor( X1, Y1 )
```

```
## [1] 0.8164205
```

```
cor( X2, Y2 )
```

```
## [1] 0.8162365
```

```
cor (X3, Y3)
```

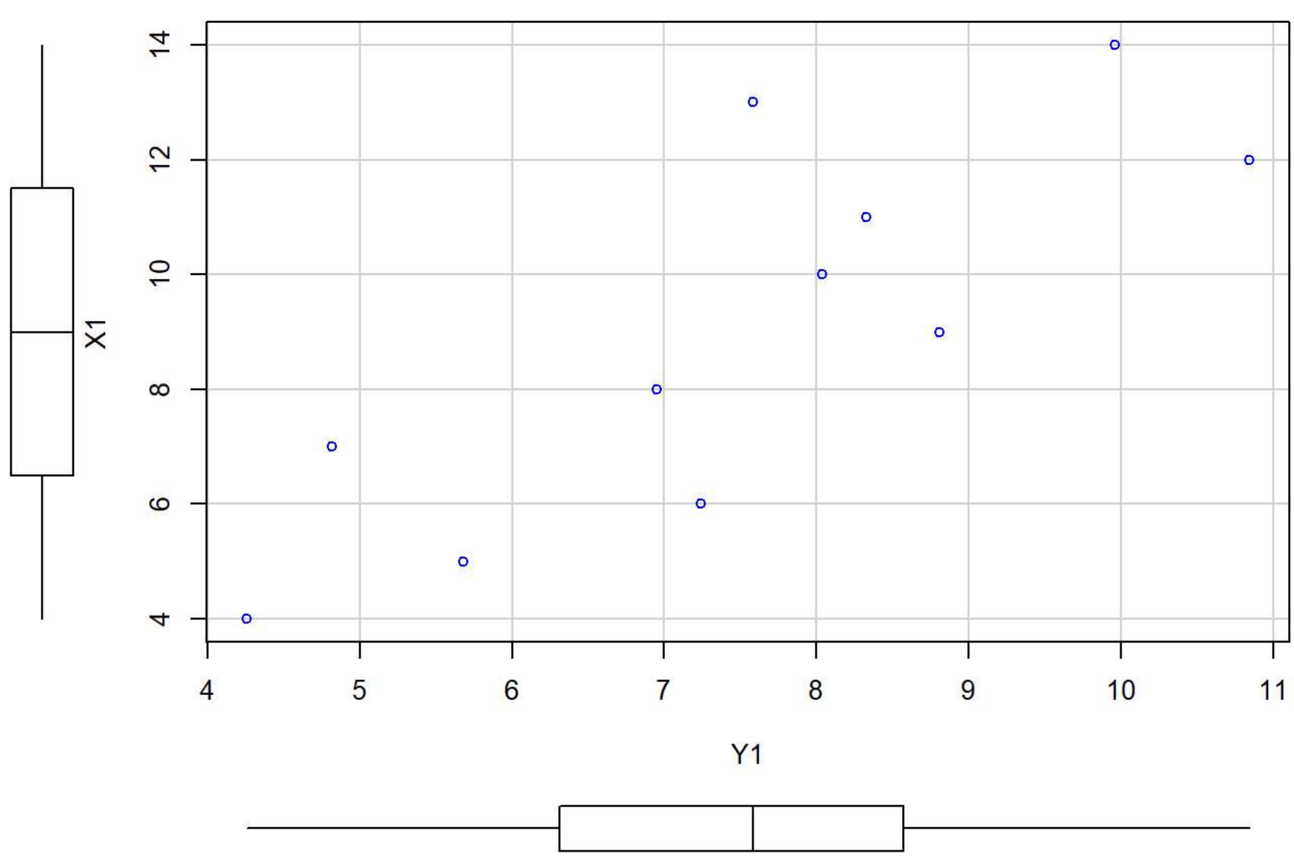
```
## [1] 0.8162867
```

```
cor (X4, Y4)
```

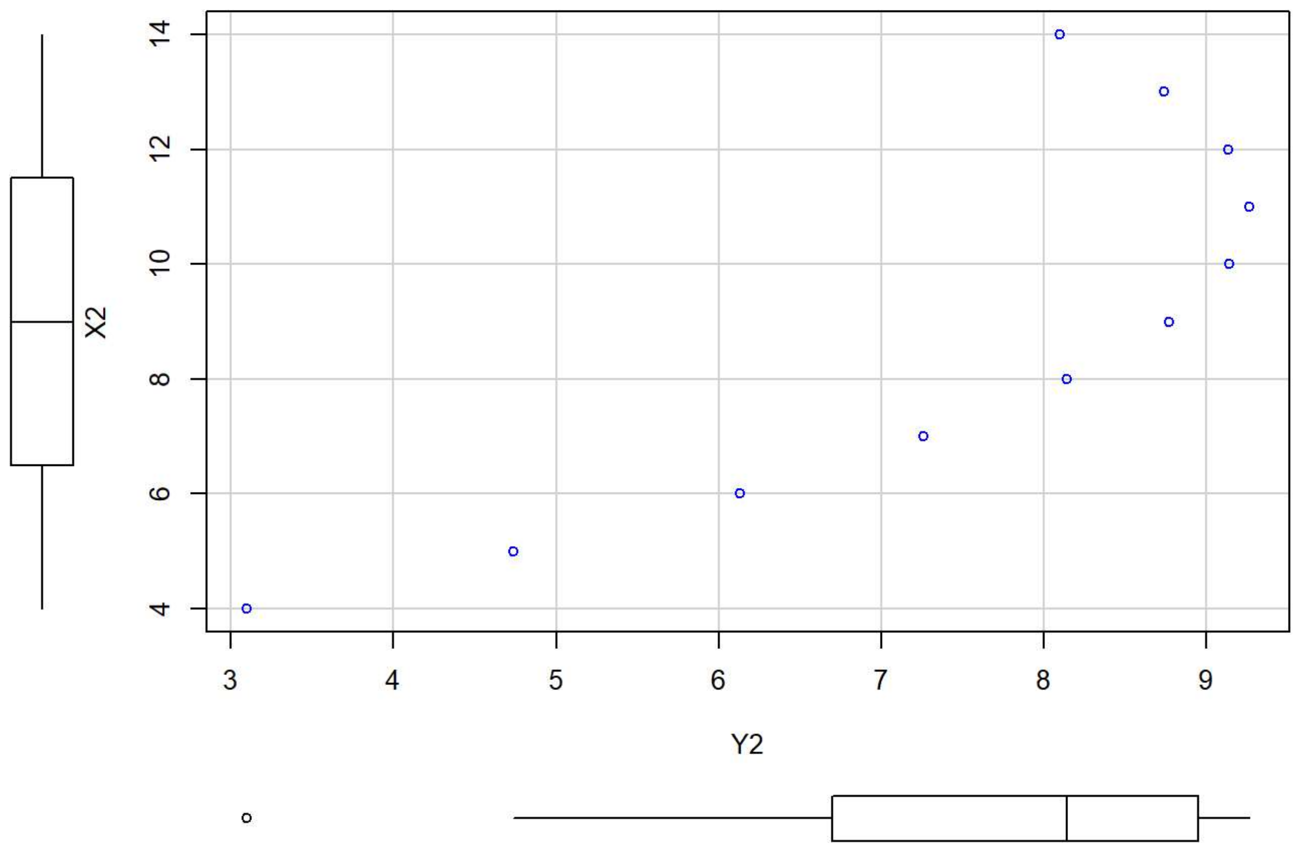
```
## [1] 0.8165214
```

All have nearly the same correlation. Let us now visualize this

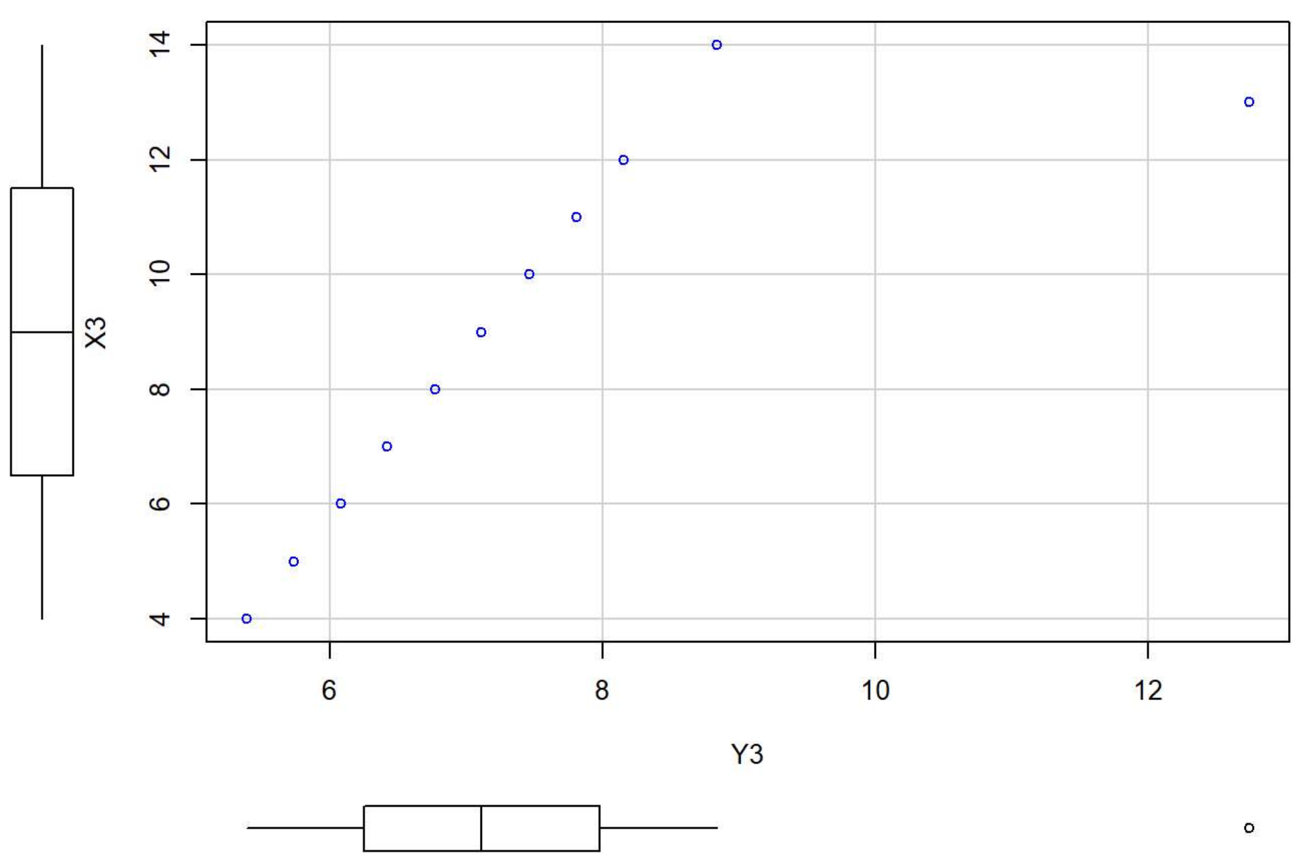
```
scatterplot(X1~Y1, regLine = FALSE, smooth = FALSE)
```



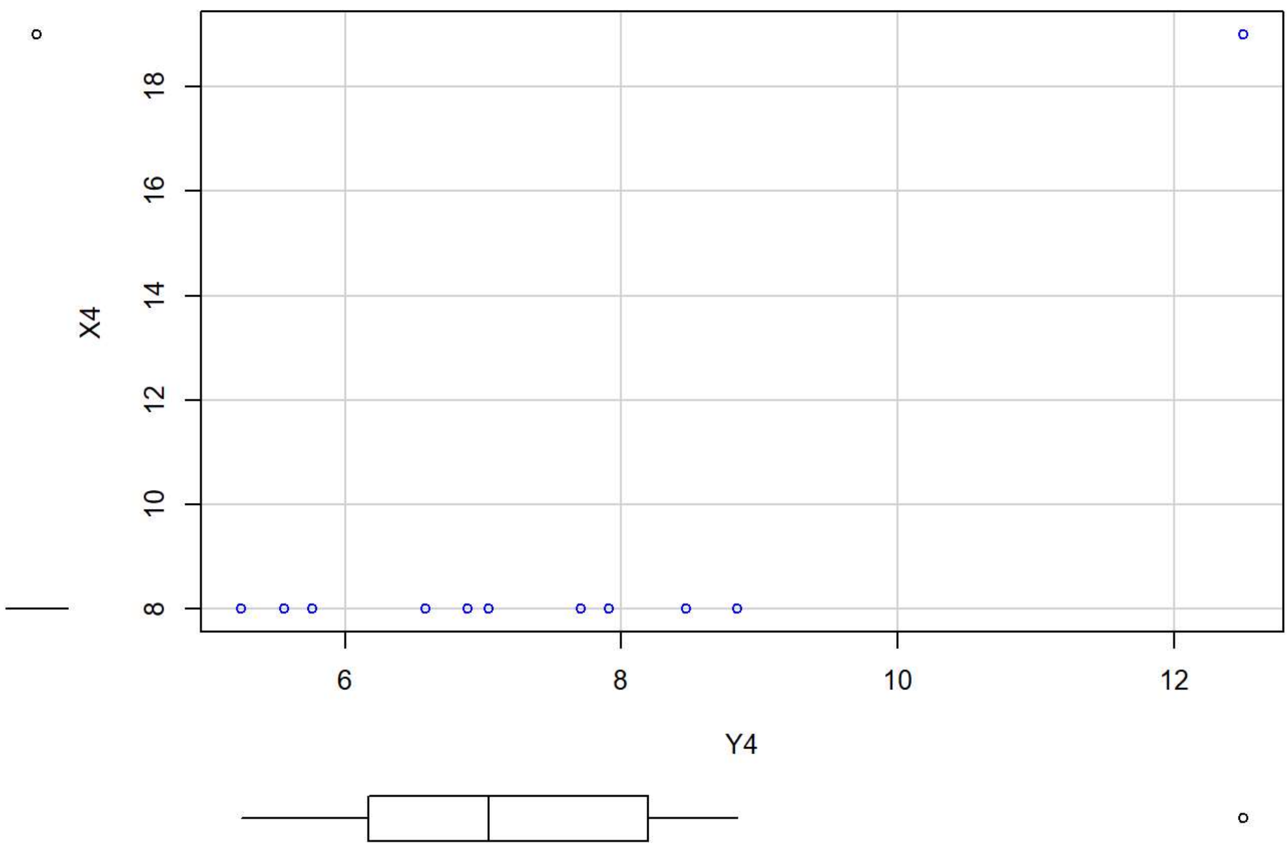

```
scatterplot(X2~Y2, regLine = FALSE, smooth = FALSE)
```




```
scatterplot(X3~Y3, regLine = FALSE, smooth = FALSE)
```




```
scatterplot(X4~Y4, regLine = FALSE, smooth = FALSE)
```



These plots are visibly different. Hence, correlation values by themselves do not reveal enough information.

Let us now understand the Spearman's Rank Order Correlation Coefficient. Loading data.

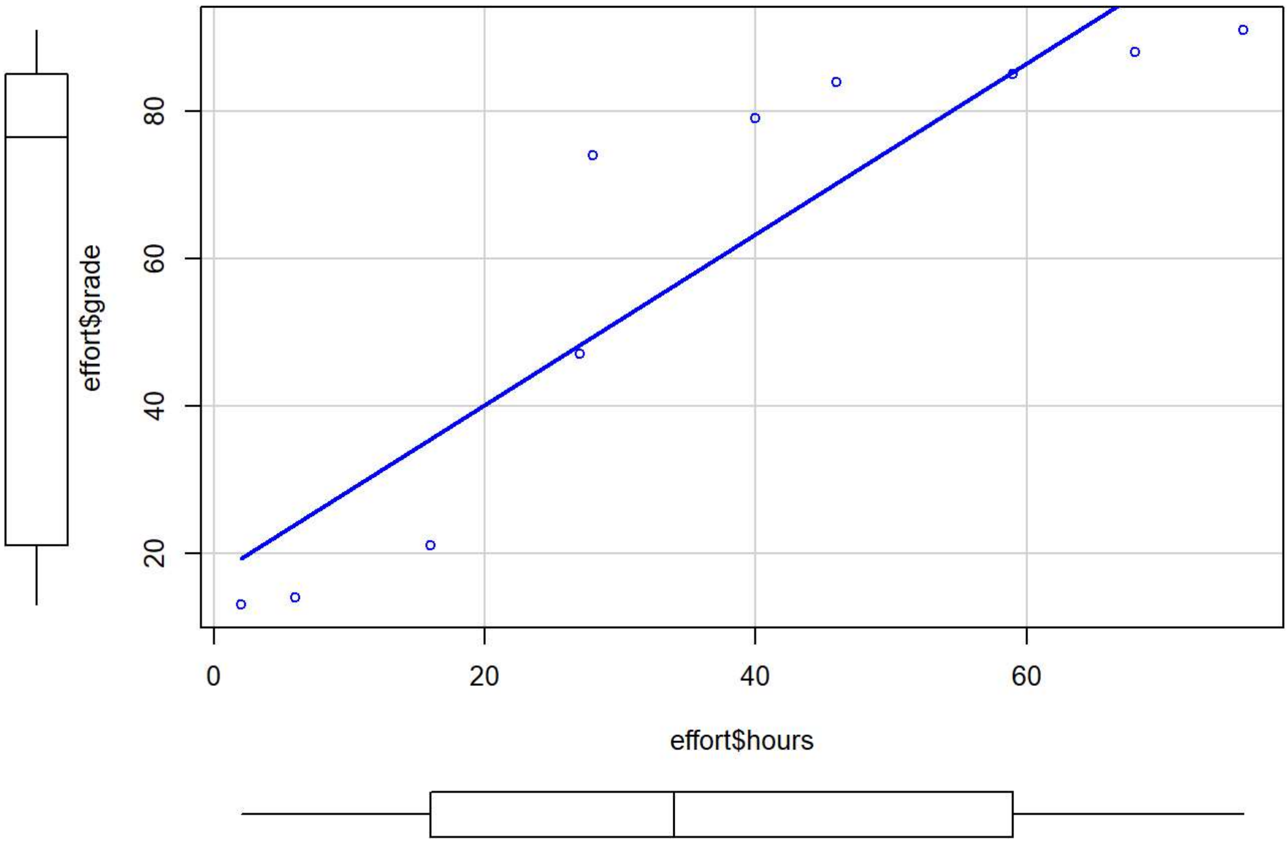
```
load( "effort.Rdata" )
effort
```

```
##    hours grade
## 1      2    13
## 2     76    91
## 3     40    79
## 4      6    14
## 5     16    21
## 6     28    74
## 7     27    47
## 8     59    85
## 9     46    84
## 10    68    88
```

```
cor(effort)
```

```
##          hours    grade
## hours 1.000000 0.909402
## grade 0.909402 1.000000
```

```
scatterplot(effort$hours, effort$grade, regLine = TRUE, smooth = FALSE)
```



When we want to understand ordinal relationships, the Spearman's correlation may be a much more useful tool.

```
hours.rank <- rank( effort$hours )   # rank students by hours worked
grade.rank <- rank( effort$grade )   # rank students by grade received


cor( hours.rank, grade.rank )
```

```
## [1] 1
```

```
#Execute this and compare with the correlation coefficient we got above
cor( effort$hours, effort$grade, method = "spearman")
```

```
## [1] 1
```

This essentially implies that the more hours a student studies, it is guaranteed that they will score better.

Handling Missing Values

Loading a data set with missing values

```
load( "parenthood2.Rdata" )
print( parenthood2 )
```

```
##     dan.sleep baby.sleep dan.grump day
## 1       7.59         NA        56   1
## 2       7.91      11.66        60   2
## 3       5.14       7.92        82   3
## 4       7.71       9.61        55   4
## 5       6.68       9.75        NA   5
## 6       5.99       5.04        72   6
## 7       8.19      10.45        53   7
## 8       7.19       8.27        60   8
## 9       7.40       6.06        NA   9
## 10      6.58       7.09        71  10
## 11      6.49      11.68        72  11
## 12      6.27       6.13        65  12
## 13      5.95       7.83        74  13
## 14      6.65       5.60        67  14
## 15      6.41       6.03        66  15
## 16      6.33       8.19        69  16
## 17      6.30       6.38        73  17
## 18      8.47      11.11        52  18
## 19      7.21       5.51        61  19
## 20      7.53       6.69        53  20
## 21      8.00       9.74        54  21
## 22      7.35       9.02        63  22
## 23      6.86       6.44        74  23
## 24      7.86       9.43        56  24
## 25      4.86       3.46        82  25
## 26      5.87       6.32        72  26
## 27      8.40       7.95        NA  27
## 28        NA       7.69        66  28
## 29      7.21       7.45        60  29
## 30      6.99         NA        67  30
## 31      8.17       7.95        44  31
## 32      7.85         NA        53  32
## 33      6.27       4.70        76  33
## 34      8.66       8.52        41  34
## 35      4.98       4.70        86  35
## 36      6.19       8.32        60  36
## 37      6.41       9.38        63  37
## 38      4.84       4.18        89  38
## 39      7.03       5.98        61  39
## 40      7.66       9.29        57  40
## 41      7.51         NA        59  41
## 42      7.92      10.54        60  42
## 43      8.12      11.78        48  43
## 44      7.47      11.60        53  44
## 45      7.99      11.35        50  45
## 46      5.44       5.63        72  46
## 47      8.16       6.98        57  47
## 48      7.62       6.03        60  48
## 49      5.87       4.66        70  49
## 50      9.00       9.81        46  50
## 51      8.31      12.07        58  51
## 52      6.71       7.57        68  52
## 53      7.43      11.35        58  53
## 54      5.90         NA        71  54
## 55      8.52       8.29        52  55
## 56      6.03         NA        74  56
## 57      7.29         NA        59  57
## 58      7.32       8.59        59  58
## 59      6.88       7.82        67  59
## 60      6.22       7.18        67  60
## 61      6.94       8.29        61  61
## 62      7.01      11.08        64  62
## 63        NA       6.46        61  63
## 64        NA       3.25        61  64
## 65        NA       9.74        54  65
## 66      7.82       8.75        62  66
```

```
## 67       8.14      11.75        52  67
## 68       7.27       9.31        64  68
## 69         NA       7.73        65  69
## 70       7.55       8.68        NA  70
## 71       7.38       9.77        57  71
## 72       7.73       9.71        59  72
## 73       5.32         NA        79  73
## 74       7.86      10.18        53  74
## 75       6.35       9.28        NA  75
## 76       7.11         NA        61  76
## 77       5.45       6.38        82  77
## 78       7.80       9.20        68  78
## 79       7.13       8.20        67  79
## 80       8.35      10.16        54  80
## 81       6.93       8.95        53  81
## 82         NA       6.80        62  82
## 83       8.66       8.34        50  83
## 84       5.09       6.25        NA  84
## 85       4.91         NA        NA  85
## 86       7.03       9.09        62  86
## 87       7.02      10.42        64  87
## 88         NA       8.89        57  88
## 89       8.15       9.43        54  89
## 90       5.88       6.79        NA  90
## 91         NA       6.91        78  91
## 92       6.66       6.05        63  92
## 93       6.85         NA        59  93
## 94       5.57       8.62        74  94
## 95       5.16       7.84        76  95
## 96         NA       5.89        79  96
## 97       7.77       9.77        51  97
## 98       5.38       6.97        82  98
## 99       7.02       6.56        55  99
## 100      6.45       7.93        74 100
```

```
describe( parenthood2 )
```

```
##            vars   n  mean    sd median trimmed   mad   min    max range  skew
## dan.sleep     1  91  6.98  1.02   7.03    7.02  1.13  4.84   9.00  4.16 -0.33
## baby.sleep    2  89  8.11  2.05   8.20    8.13  2.28  3.25  12.07  8.82 -0.09
## dan.grump     3  92 63.15  9.85  61.00   62.66 10.38 41.00  89.00 48.00  0.38
## day           4 100 50.50 29.01  50.50   50.50 37.06  1.00 100.00 99.00  0.00
##            kurtosis   se
## dan.sleep     -0.73 0.11
## baby.sleep    -0.59 0.22
## dan.grump     -0.31 1.03
## day           -1.24 2.90
```

Finding correlations

```
cor(parenthood2)
```

```
##            dan.sleep baby.sleep dan.grump day
## dan.sleep          1         NA        NA  NA
## baby.sleep        NA          1        NA  NA
## dan.grump         NA         NA         1  NA
## day               NA         NA        NA   1
```

Why are we getting these values? Because there are values we don’t know in our data set. It is not possible to compute means if we don’t know certain values. There are two ways we can tackle this - 1. By omitting the entire row whenever NA is present

```
cor(parenthood2, use = "complete.obs")
```

```
##              dan.sleep baby.sleep   dan.grump         day
## dan.sleep   1.00000000  0.6394985 -0.89951468  0.06132891
## baby.sleep  0.63949845  1.0000000 -0.58656066  0.14555814
## dan.grump  -0.89951468 -0.5865607  1.00000000 -0.06816586
## day         0.06132891  0.1455581 -0.06816586  1.00000000
```

2. When calculating correlation, omitting the specific row only in a pairwise manner. That is, if we have a missing value in baby sleep, the row may still be included while calculating correlation between dan.grump and dan.sleep.

```
cor(parenthood2, use = "pairwise.complete.obs")
```

```
##              dan.sleep  baby.sleep    dan.grump          day
## dan.sleep   1.00000000  0.61472303 -0.903442442 -0.076796665
## baby.sleep  0.61472303  1.00000000 -0.567802669  0.058309485
## dan.grump  -0.90344244 -0.56780267  1.000000000  0.005833399
## day        -0.07679667  0.05830949  0.005833399  1.000000000
```

This can also be calculated using the lsr package.

```
library(lsr)
correlate(parenthood2)
```

```
## 
## CORRELATIONS
## ============
## - correlation type:  pearson 
## - correlations shown only when both variables are numeric
## 
##            dan.sleep baby.sleep dan.grump    day
## dan.sleep          .      0.615    -0.903 -0.077
## baby.sleep     0.615          .    -0.568  0.058
## dan.grump     -0.903     -0.568         .  0.006
## day           -0.077      0.058     0.006      .
```

PART-2: Correlation Plots

Loading the package

```
library(ggcorrplot)
```

```
## Loading required package: ggplot2
```

```
## 
## Attaching package: 'ggplot2'
```

```
## The following objects are masked from 'package:psych':
## 
##     %+%, alpha
```

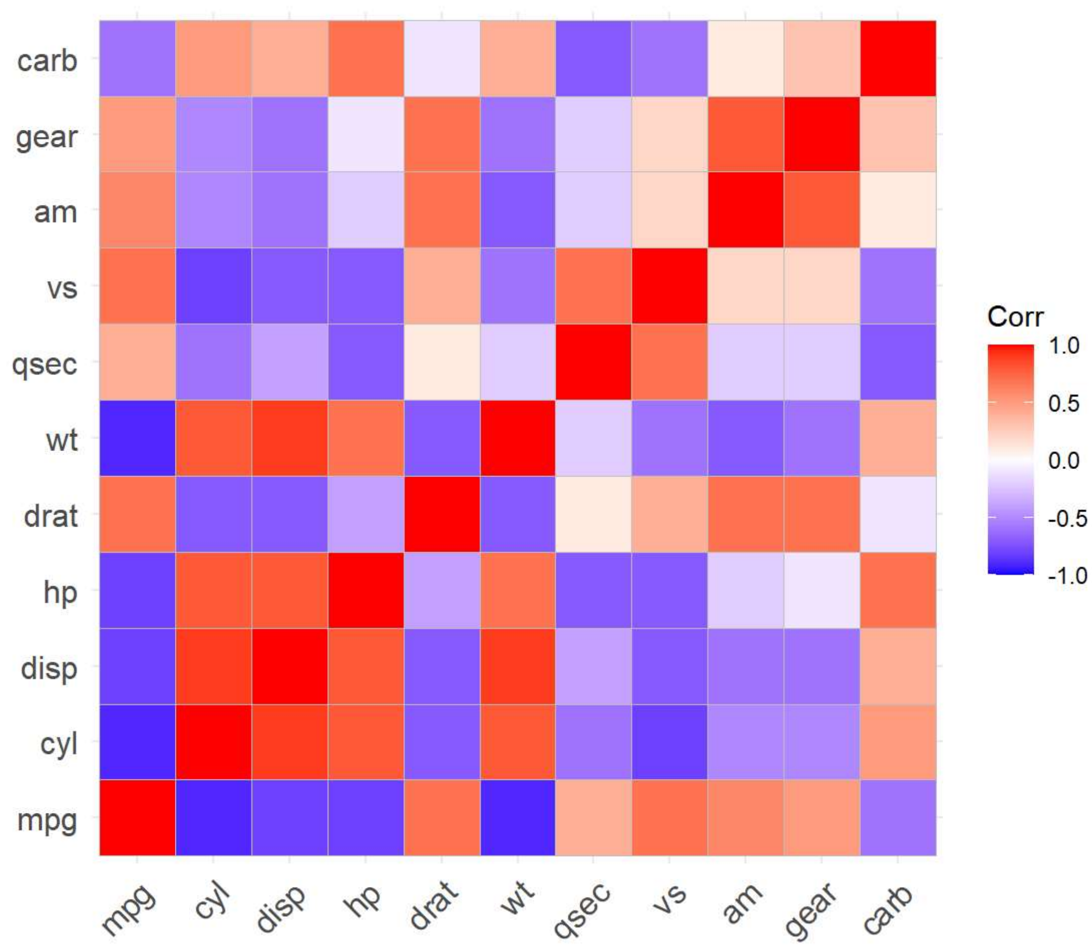
```
data(mtcars) #loading data
corr <- round(cor(mtcars), 1) #Rounding correlation data to one decimal place
head(corr[, 1:6]) #Displaying the correlation matrix
```

```
##       mpg  cyl disp   hp drat   wt
## mpg   1.0 -0.9 -0.8 -0.8  0.7 -0.9
## cyl  -0.9  1.0  0.9  0.8 -0.7  0.8
## disp -0.8  0.9  1.0  0.8 -0.7  0.9
## hp   -0.8  0.8  0.8  1.0 -0.4  0.7
## drat  0.7 -0.7 -0.7 -0.4  1.0 -0.7
## wt   -0.9  0.8  0.9  0.7 -0.7  1.0
```

```
# Compute a matrix of correlation p-values
p.mat <- cor_pmat(mtcars)
head(p.mat[, 1:4])
```

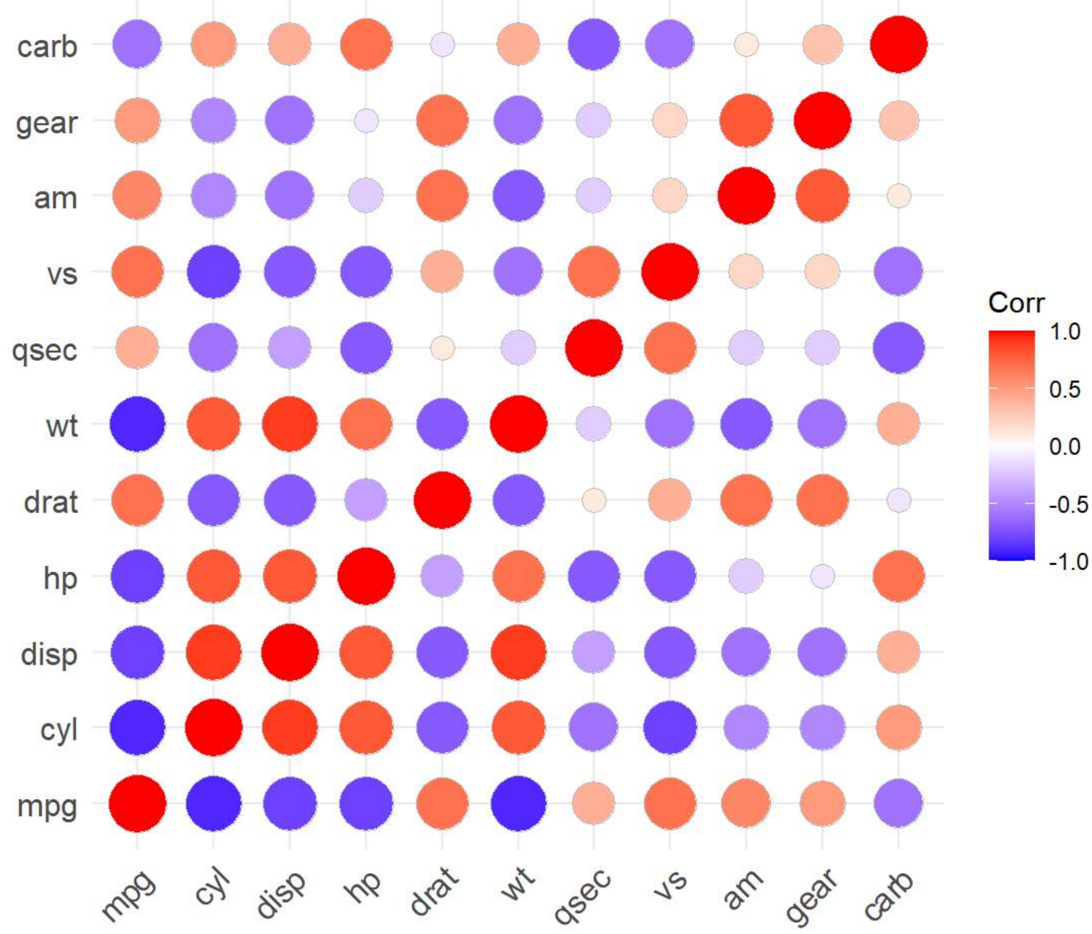
```
##               mpg          cyl         disp           hp
## mpg  0.000000e+00 6.112687e-10 9.380327e-10 1.787835e-07
## cyl  6.112687e-10 0.000000e+00 1.802838e-12 3.477861e-09
## disp 9.380327e-10 1.802838e-12 0.000000e+00 7.142679e-08
## hp   1.787835e-07 3.477861e-09 7.142679e-08 0.000000e+00
## drat 1.776240e-05 8.244636e-06 5.282022e-06 9.988772e-03
## wt   1.293959e-10 1.217567e-07 1.222320e-11 4.145827e-05
```

```
ggcorrplot(corr)
```



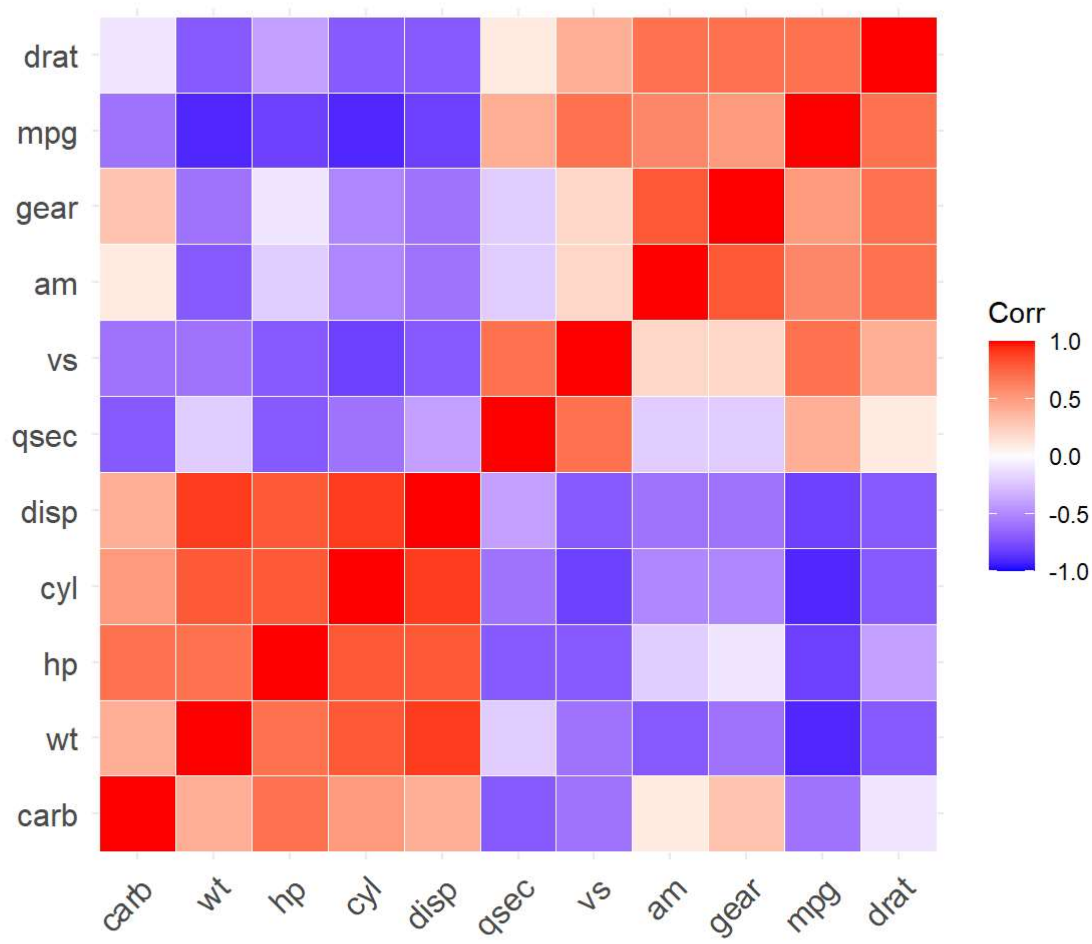
Above is a correlation matrix. In this manner we can visualize the correlation between different value pairs in high dimensional data sets. This uses the default method as square. Let us now try circle.

```
ggcorrplot(corr, method = "circle")
```



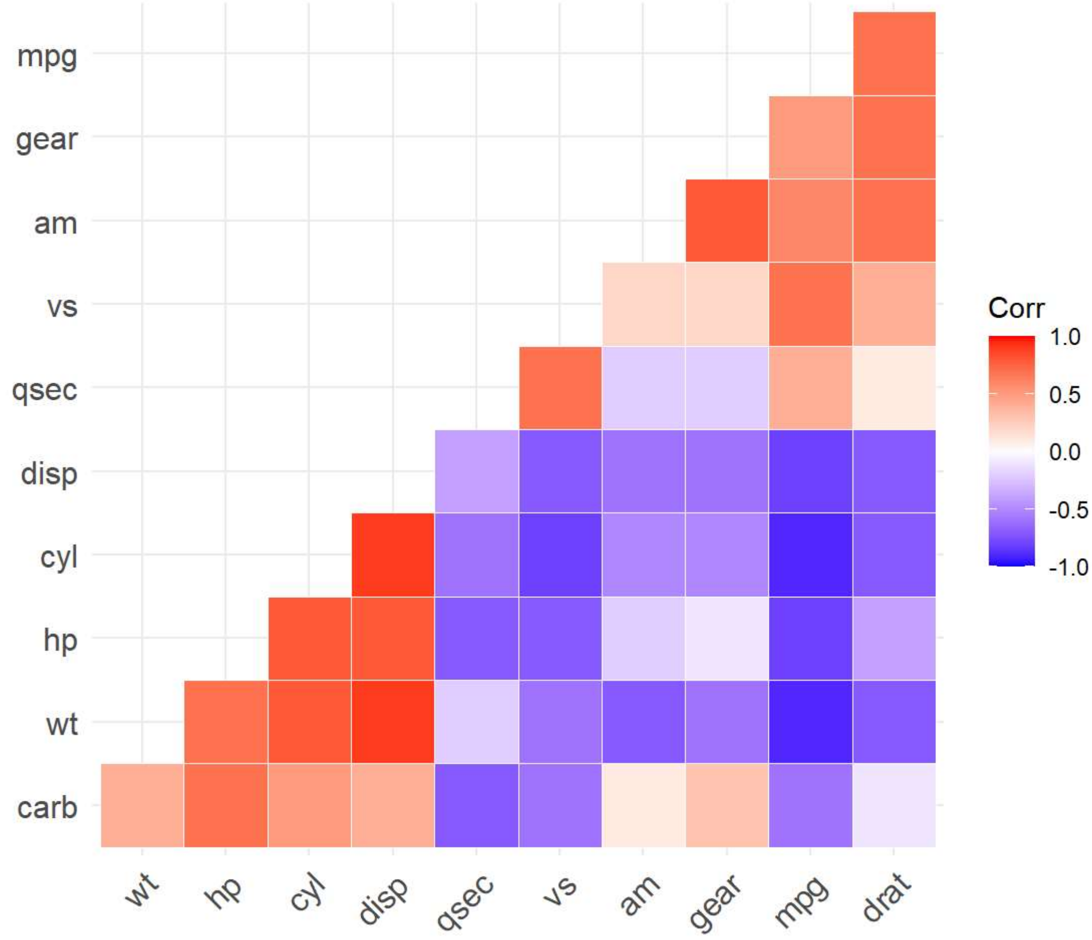
This gives you circle plots. Another interesting feature, is that ggcorrplot can perform a heirarchical clustering in the dataset. Values in the same cluster have higher correlations with each other and low correlations with values in other clusters.

```
ggcorrplot(corr, hc.order = TRUE, outline.col = "white")
```

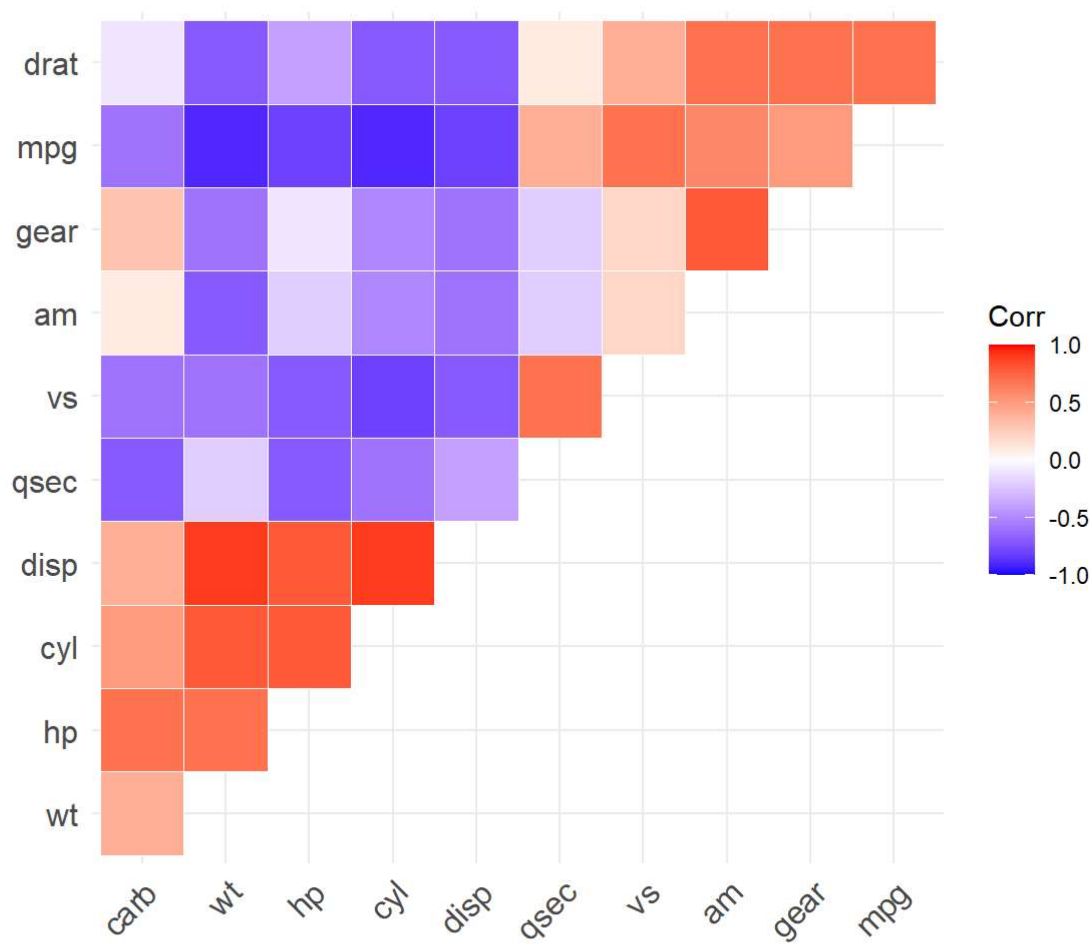


There is redundancy in this plot, since it is a mirror image across the diagonal. Hence, it can be more compactly represented as follows -

```
ggcorrplot(corr, hc.order = TRUE, type = "lower",
     outline.col = "white")
```

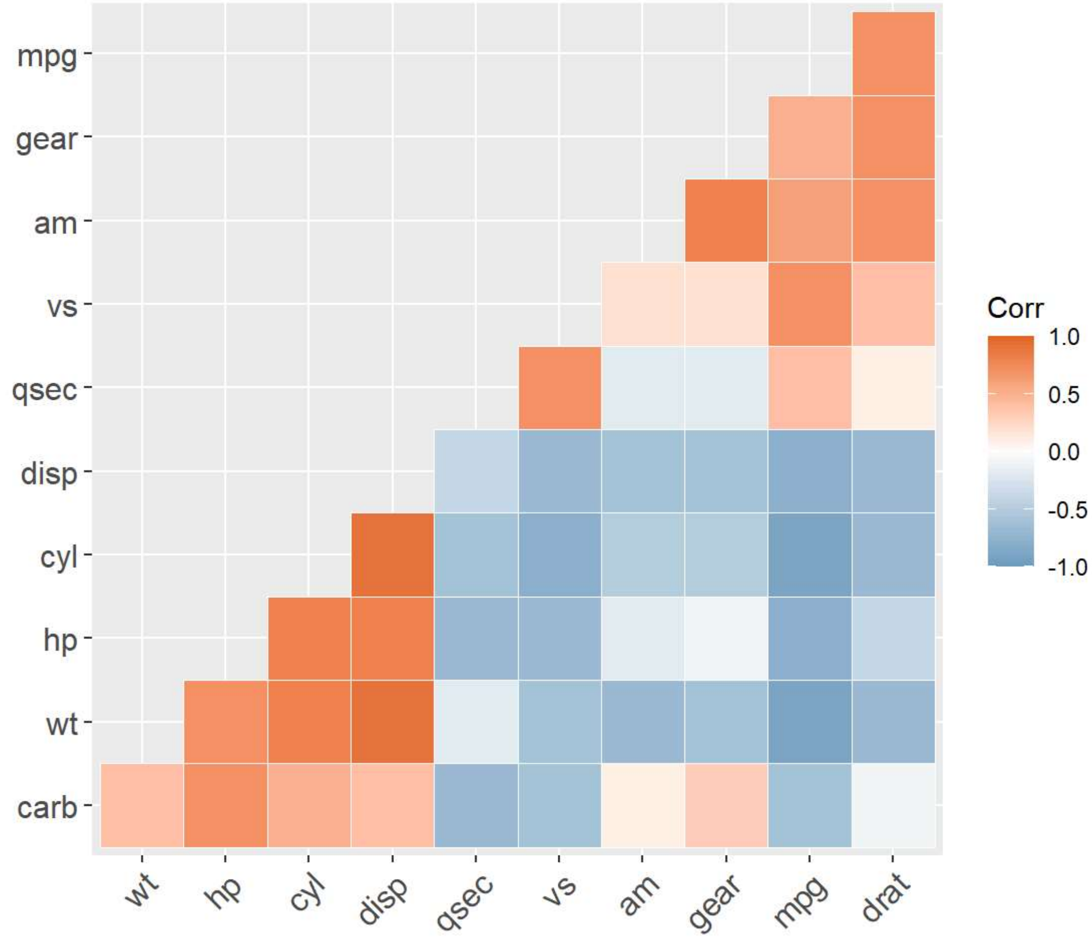



```
ggcorrplot(corr, hc.order = TRUE, type = "upper",
     outline.col = "white")
```



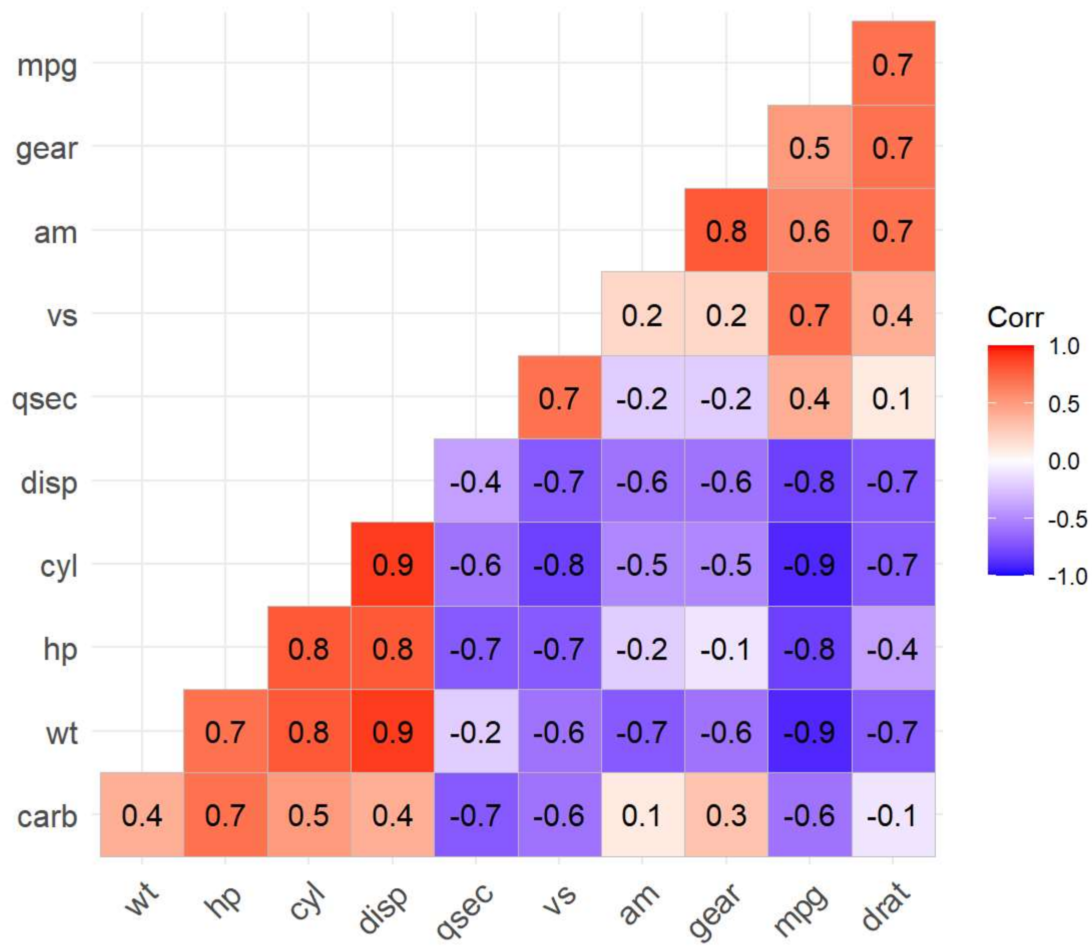
We can also change the colours and theme of the plots.

```
ggcorrplot(corr, hc.order = TRUE, type = "lower",
   outline.col = "white",
   ggtheme = ggplot2::theme_gray,
   colors = c("#6D9EC1", "white", "#E46726"))
```



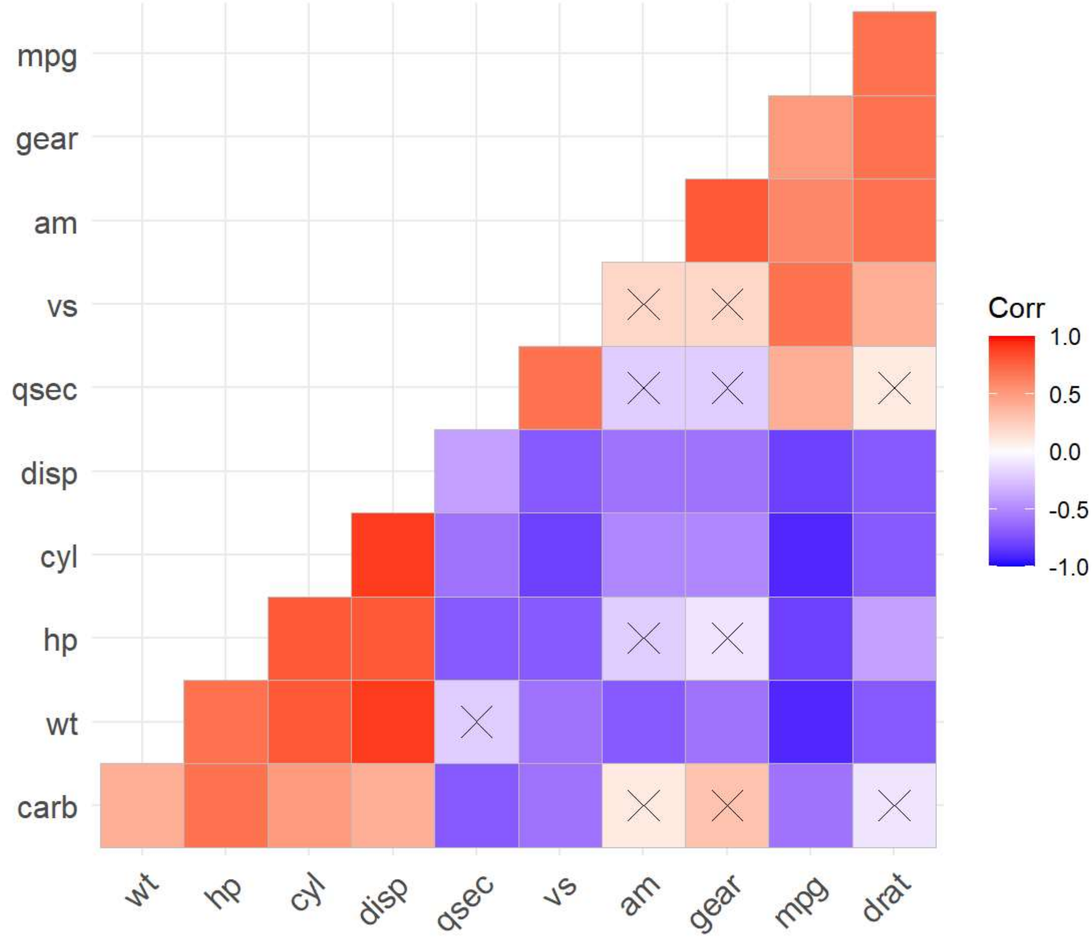
We can also visualize the specific correlation coefficients.

```
ggcorrplot(corr, hc.order = TRUE, type = "lower",
   lab = TRUE)
```



It may be even more meaningful to include p-values, The following code, when executed, crosses out those values with statistically insignificant correlations.

```
ggcorrplot(corr, hc.order = TRUE,
    type = "lower", p.mat = p.mat)
```



We may also leave them blank.

```
ggcorrplot(corr, p.mat = p.mat, hc.order = TRUE,
    type = "lower", insig = "blank")
```