Math_574M_Hw7

Saheed Adisa Ganiyu

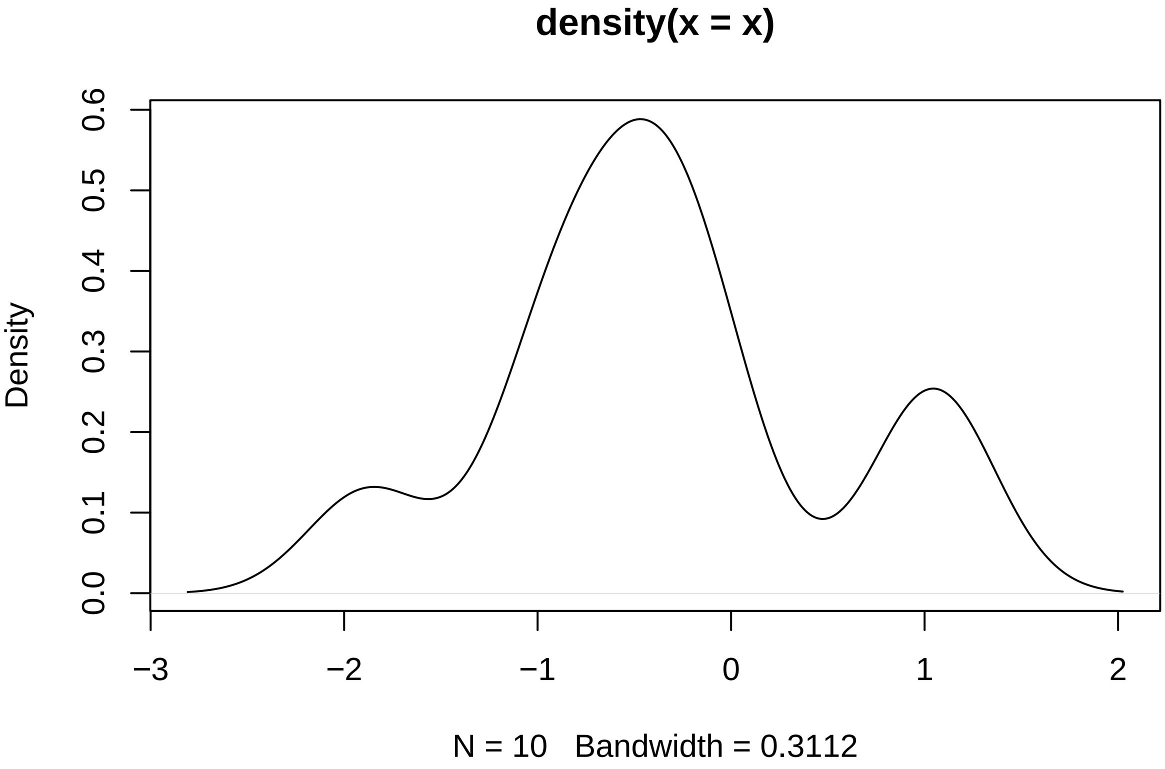
2023-08-30

10. Generate random numbers from Normal distribution

```
help(set.seed)
set.seed(2023)     # for reproducible of random generated number
x = rnorm(10, mean=0, sd=1)     # generating a random sample of 10 numbers from a normal distribution
print(round(x,4))
```

```
##  [1] -0.0838 -0.9829 -1.8751 -0.1861 -0.6335  1.0908 -0.9137  1.0016 -0.3993
## [10] -0.4681
```

```
plot(density(x))     # creating a density plot
```

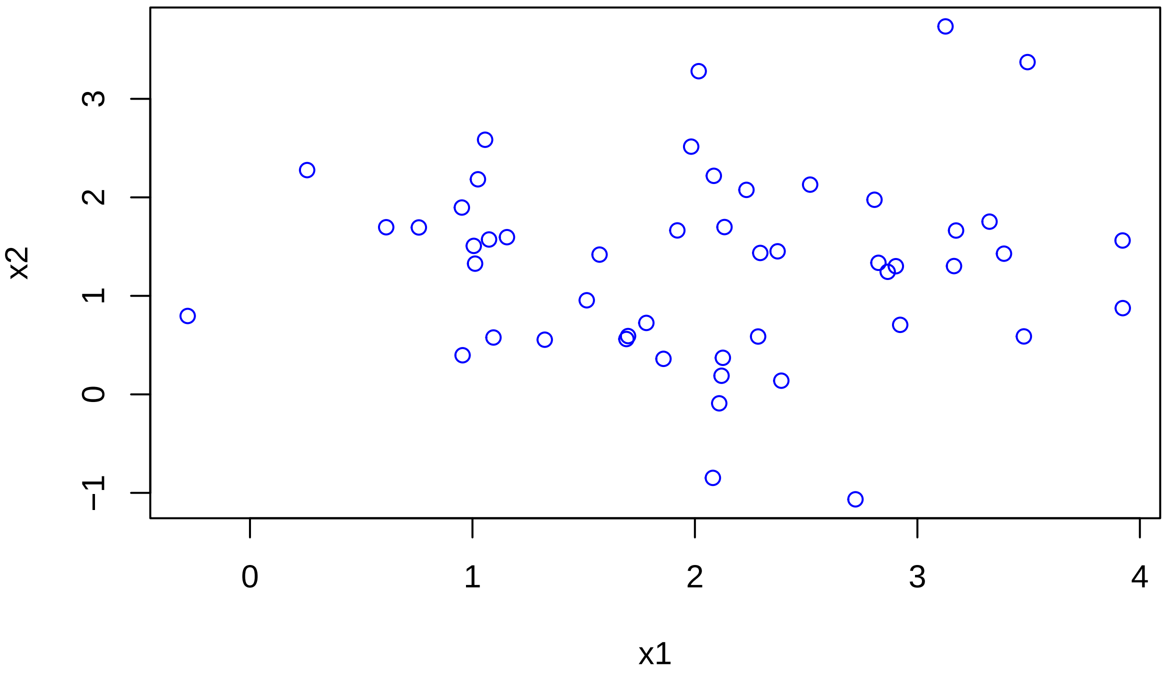


11. Generate data from multivariate normal distribution.

```r
library(MASS)     # loading the MASS package
help(mvrnorm)
mean1 = c(2,1)     # creating a vector of two element
cov1 = matrix(c(1,0,0,1),nrow=2)     # creating a matrix of size 2x2
data1 = mvrnorm(50,mean1,cov1)     # generating a random sample of 50 data points from a multivariate n
print(data1)
```

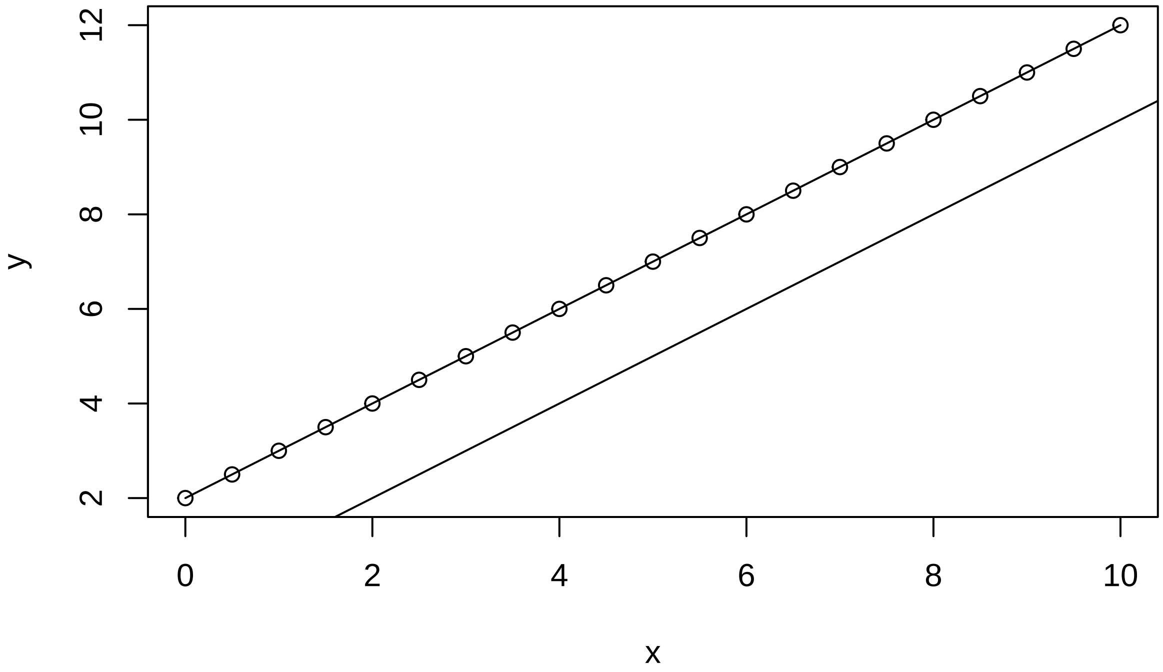
```
##             [,1]        [,2]
##  [1,]  1.0118161  1.32696208
##  [2,]  2.2840632  0.58725310
##  [3,]  3.9222675  1.56203647
##  [4,]  3.1737198  1.66335826
##  [5,]  0.9557173  0.39710272
##  [6,]  2.1329281  1.69837769
##  [7,]  1.1551230  1.59584645
##  [8,]  2.3715925  1.45209183
##  [9,]  0.9524161  1.89674396
## [10,]  1.0745843  1.57221651
## [11,]  3.4786053  0.58834699
## [12,]  2.9226349  0.70567285
## [13,]  2.0849024  2.21857396
## [14,]  2.8665100  1.24411143
## [15,]  1.3249576  0.55484804
## [16,]  2.0807049 -0.84780364
## [17,]  2.1255926  0.37117469
## [18,]  2.3883311  0.13891931
## [19,]  1.9831726  2.51492030
## [20,]  3.1263066  3.73523893
## [21,]  1.7816482  0.72512294
## [22,]  0.2569631  2.27665407
## [23,]  2.1195785  0.18901966
## [24,]  1.5138200  0.95507722
## [25,]  1.8586315  0.36058762
## [26,]  1.6920324  0.56164400
## [27,]  1.0059187  1.50720220
## [28,]  2.5178118  2.12921920
## [29,]  2.8071364  1.97563765
## [30,]  3.9231790  0.87539621
## [31,]  3.3891385  1.42833658
## [32,]  1.5713300  1.41872636
## [33,]  2.2937814  1.43546649
## [34,] -0.2799988  0.79573791
## [35,]  3.1645108  1.30235349
## [36,]  0.7591324  1.69435780
## [37,]  3.4950839  3.37306566
## [38,]  2.2316363  2.07597029
## [39,]  2.8248667  1.33538478
## [40,]  3.3241731  1.75342823
## [41,]  1.9209863  1.66438628
## [42,]  2.1092416 -0.09088266
## [43,]  1.0944751  0.57722915
## [44,]  1.0246254  2.18340204
## [45,]  1.0569171  2.58482679
## [46,]  2.0169729  3.28085537
## [47,]  2.7215976 -1.06545705
## [48,]  1.6994586  0.59188803
## [49,]  2.9031171  1.30101457
## [50,]  0.6123056  1.69661308
```

```
plot(data1,xlab="x1",ylab="x2",col="blue")      # creating a scatter plot of data1
```



12. Draw a scatter plot

```
x = seq(0,10,0.5)      # generating a sequence of numbers from 0 to 10 with step of 0.5
y = seq(2,12,0.5)      # generating a sequence of numbers from 2 to 12 with step of 0.5
plot(x,y)      # creating the plot of x, y
lines(x,y)
abline(a=0,b=1)      # adding a straight line of intercept "a" with slope "b"
```



13. Load data and get basic statistics summary

```
data(cars)     # loading the car dataset
cars
```

```
##    speed dist
## 1      4    2
## 2      4   10
## 3      7    4
## 4      7   22
## 5      8   16
## 6      9   10
## 7     10   18
## 8     10   26
## 9     10   34
## 10    11   17
## 11    11   28
## 12    12   14
## 13    12   20
## 14    12   24
## 15    12   28
## 16    13   26
## 17    13   34
## 18    13   34
## 19    13   46
## 20    14   26
## 21    14   36
## 22    14   60
## 23    14   80
## 24    15   20
## 25    15   26
## 26    15   54
## 27    16   32
## 28    16   40
## 29    17   32
## 30    17   40
## 31    17   50
## 32    18   42
## 33    18   56
## 34    18   76
## 35    18   84
## 36    19   36
## 37    19   46
## 38    19   68
## 39    20   32
## 40    20   48
## 41    20   52
## 42    20   56
## 43    20   64
## 44    22   66
## 45    23   54
## 46    24   70
## 47    24   92
## 48    24   93
## 49    24  120
## 50    25   85
```

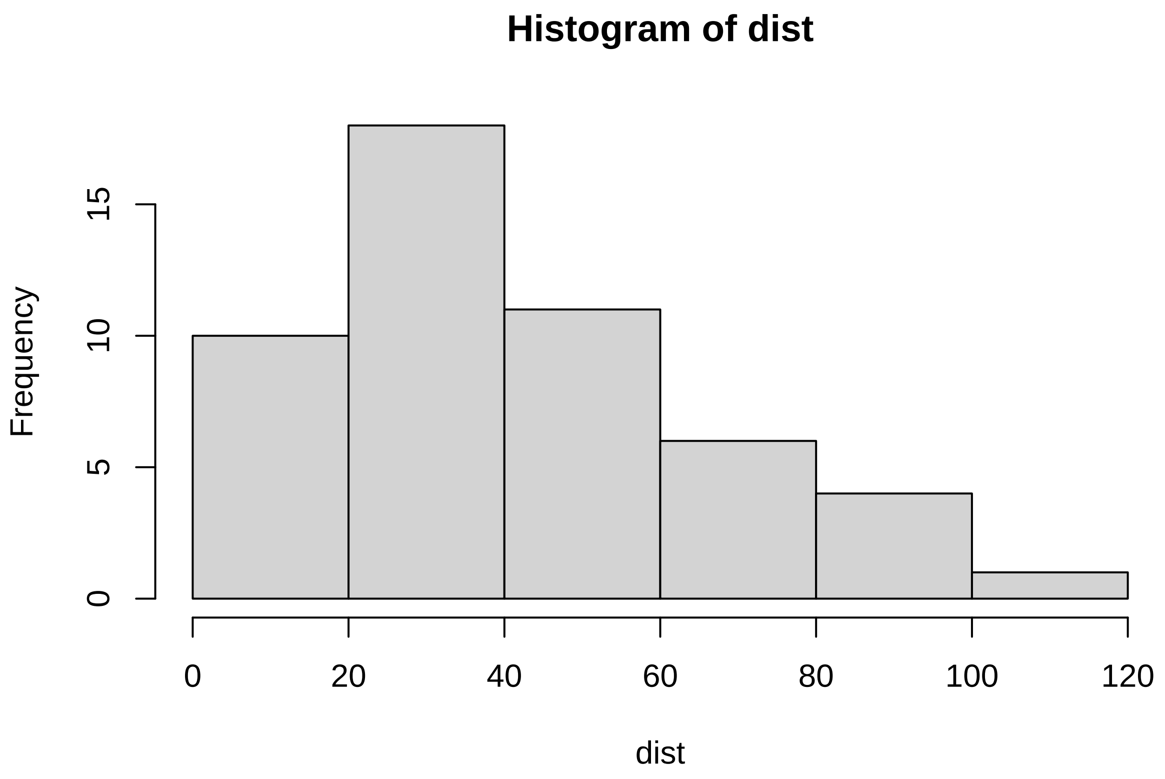
```
attach(cars)     # attaching the cars dataset to the search path
mean(dist)     # computing the mean of columns "dist"
```

```
## [1] 42.98
```

```
var(dist)     # computing the variance of columns "dist"
```

```
## [1] 664.0608
```

```
hist(dist)     # crating the histogram of "dist"
```



14. Matrix and submatrix

```
a = matrix(rnorm(20), 5, 4)      # creating a 5x4 matrix with 20 random numbers from a standard normal d
a
```

```
##              [,1]         [,2]       [,3]       [,4]
## [1,]  0.669886090  0.009423151 -0.4750617  1.0991389
## [2,]  0.006142371 -1.004842034  0.3680799 -0.6944905
## [3,]  0.265166372  2.579247774  0.1035406 -0.2039437
## [4,] -2.061095747  0.701311418  0.3463417 -0.3297715
## [5,]  0.752977196  1.879238238 -0.7683650  1.2507176
```

```
b = a[1,]      # extractsing the first row of the matrix "a"
b
```

```
## [1]  0.669886090  0.009423151 -0.475061689  1.099138929
```

```
c = a[, 2:4]      # extracting columns 2 through 4
c
```

```
##              [,1]       [,2]       [,3]
## [1,]  0.009423151 -0.4750617  1.0991389
## [2,] -1.004842034  0.3680799 -0.6944905
## [3,]  2.579247774  0.1035406 -0.2039437
## [4,]  0.701311418  0.3463417 -0.3297715
## [5,]  1.879238238 -0.7683650  1.2507176
```

```
d = a[-c(1:2),]      # excluding rows 1 and 2 from the matrix "a"
d
```

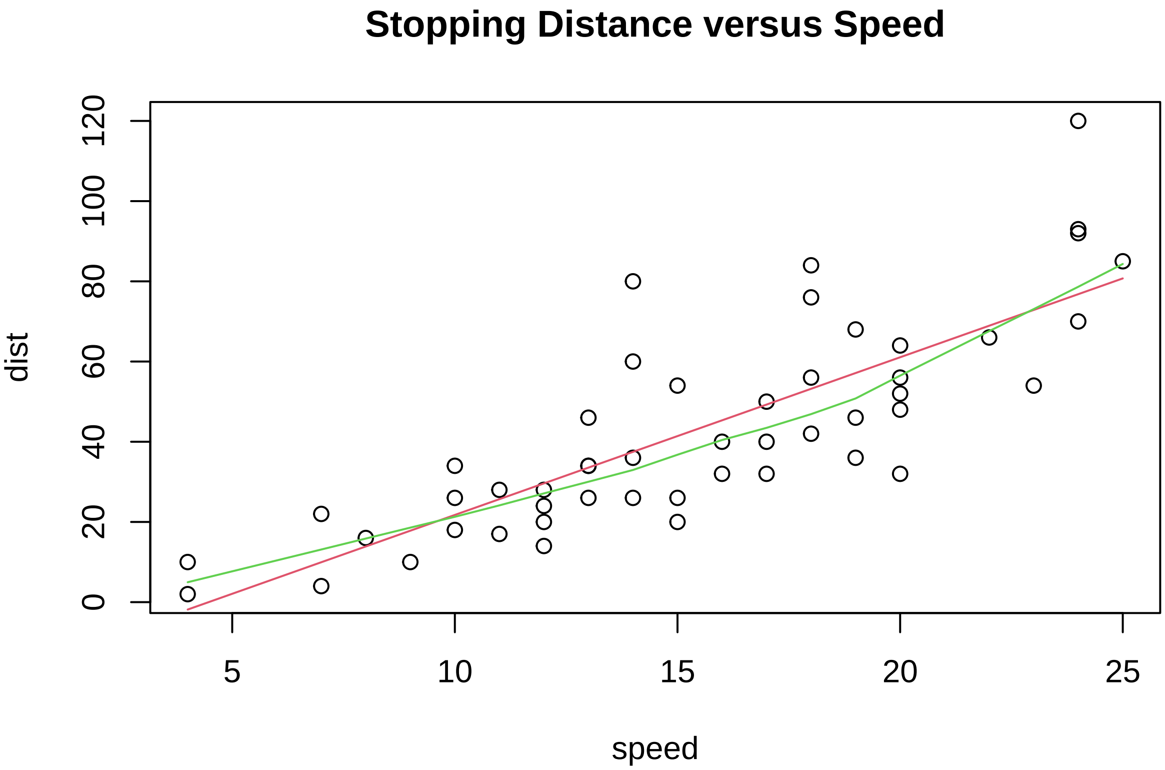
```
##            [,1]      [,2]       [,3]       [,4]
## [1,]  0.2651664 2.5792478  0.1035406 -0.2039437
## [2,] -2.0610957 0.7013114  0.3463417 -0.3297715
## [3,]  0.7529772 1.8792382 -0.7683650  1.2507176
```

15. Fit a simple linear regression model

```
 mymodel = lm(dist ~ speed)     # fitting the linear model by regressing column "dist" on column "speed
summary(mymodel)
```

```
##
## Call:
## lm(formula = dist ~ speed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.069  -9.525  -2.272   9.215  43.201
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -17.5791     6.7584  -2.601   0.0123 *
## speed         3.9324     0.4155   9.464 1.49e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.38 on 48 degrees of freedom
## Multiple R-squared:  0.6511, Adjusted R-squared:  0.6438
## F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

```
### draw the scatter plot
plot(cars, main="Stopping Distance versus Speed")
### draw the fitted regression line (red)
lines(speed, fitted(mymodel), type="l", lty=1, col=2)
### draw a smooth line through a scatter plot (green)
lines(stats::lowess(cars), type="l", lty=1, col=3)
```



```r
detach(cars)
```

16. Data manipulation using tidyr and dplyr. Make plots using ggplot2.

```r
# Loading the packages
library(dplyr)
```

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

## The following object is masked from 'package:MASS':
##
##     select

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

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

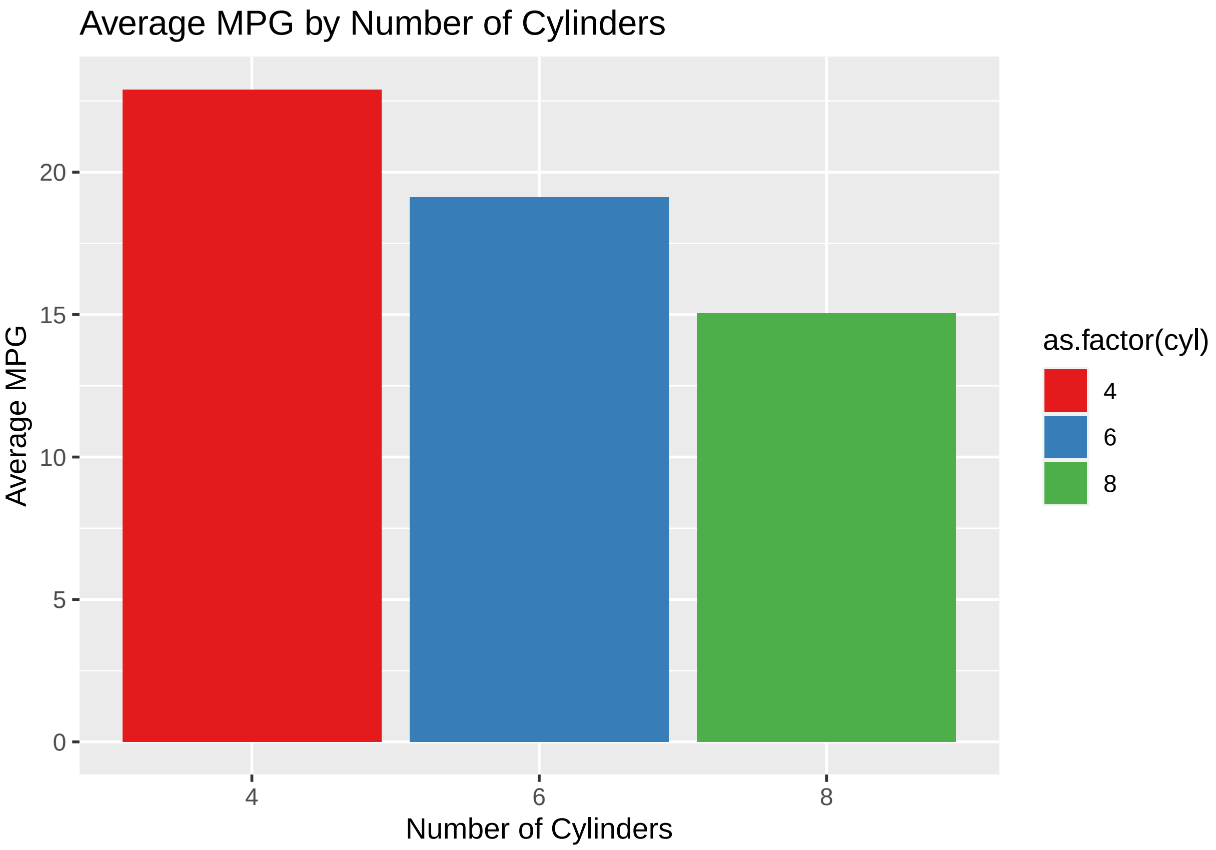
```r
library(dplyr)
library(ggplot2)
```

```
data(mtcars)     # loading mtcars dataset
help(mtcars)
### select all cars with automatic transmission and
### compute average mpg by number of cylinders
data1 <- mtcars %>%
  filter(am == 0) %>%     # the "%>%" is pipe operator which is use to chain together a series of data
  group_by(cyl) %>%
  summarise(avg_mpg = mean(mpg))

### Display the result
print(data1)
```

```
## # A tibble: 3 x 2
##     cyl avg_mpg
##   <dbl>   <dbl>
## 1     4    22.9
## 2     6    19.1
## 3     8    15.0
```

```
### draw a barplot
ggplot(data1, aes(x=as.factor(cyl), y=avg_mpg, fill=as.factor(cyl))) +
   geom_bar(stat="identity") +     # this adds a bar chart layer to the plot
   scale_fill_brewer(palette="Set1")+
   labs(x = "Number of Cylinders", y = "Average MPG", title = "Average MPG by Number of Cylinders")
```


Average MPG by Number of Cylinders
Average MPG
Number of Cylinders
as.factor(cyl)
🟥 4
🟦 6
🟩 8