R base

Keyboard symbols

: colon

; semicolon

~ tilde

& ampersand

- dash

_ underscore

\ backslash

R basic rules

- R is case sensitive
- R starts counting from one
- formula notation: $y \sim x_1 + x_2$
- [ ] is used for subsetting
- ( ) for the arguments of an R function

R basic rules

- The command line is shown as “>”
- A “+” substitutes “>” if a command in incomplete
- Assignments are defined using ← or =
- # is used for comments
- Missing values are shown as NA
- Impossible numbers as NAN
- Infinity is displayed as Inf

R data structures

- vector
- matrix
- array
- table
- dataframe
- list

R data types

- logical (TRUE, FALSE)
- numeric
- integer
- character (strings)
- factor (categorical variable)
- date

R objects

- data structures
- commands
- functions
- models
- data sets
- library objects

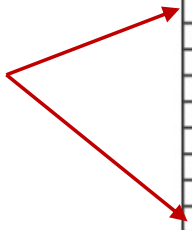
Dataframes

Variables are stored in columns

	Manufacturer	Model	Type	Min.Price	Price	Max.Price	MPG.city	MPG.highway	AirBags	DriveTrain
1	Acura	Integra	Small	12.9	15.9	18.8	25	31	None	Front
2	Acura	Legend	Midsize	29.2	33.9	38.7	18	25	Driver & Passenger	Front
3	Audi	90	Compact	25.9	29.1	32.3	20	26	Driver only	Front
4	Audi	100	Midsize	30.8	37.7	44.6	19	26	Driver & Passenger	Front
5	BMW	535i	Midsize	23.7	30	36.2	22	30	Driver only	Rear
6	Buick	Century	Midsize	14.2	15.7	17.3	22	31	Driver only	Front
7	Buick	LeSabre	Large	19.9	20.8	21.7	19	28	Driver only	Front
8	Buick	Roadmaster	Large	22.6	23.7	24.9	16	25	Driver only	Rear
9	Buick	Riviera	Midsize	26.3	26.3	26.3	19	27	Driver only	Front
10	Cadillac	DeVille	Large	33	34.7	36.3	16	25	Driver only	Front
11	Cadillac	Seville	Midsize	37.5	40.1	42.7	16	25	Driver & Passenger	Front
12	Chevrolet	Cavalier	Compact	8.5	13.4	18.3	25	36	None	Front
13	Chevrolet	Corsica	Compact	11.4	11.4	11.4	25	34	Driver only	Front
14	Chevrolet	Camaro	Sporty	13.4	15.1	16.8	19	28	Driver & Passenger	Rear
15	Chevrolet	Lumina	Midsize	13.4	15.9	18.4	21	29	None	Front
16	Chevrolet	Lumina_APV	Van	14.7	16.3	18	18	23	None	Front
17	Chevrolet	Astro	Van	14.7	16.6	18.6	15	20	None	4WD
18	Chevrolet	Caprice	Large	18	18.8	19.6	17	26	Driver only	Rear
19	Chevrolet	Corvette	Sporty	34.6	38	41.5	17	25	Driver only	Rear
20	Chrylser	Concorde	Large	18.4	18.4	18.4	20	28	Driver & Passenger	Front
21	Chrysler	LeBaron	Compact	14.5	15.8	17.1	23	28	Driver & Passenger	Front

R observations in a dataframe

observations are stored in rows



	Manufacturer	Model	Type	Min.Price	Price	Max.Price	MPG.city	MPG.highway
1	Acura	Integra	Small	12.9	15.9	18.8	25	31
2	Acura	Legend	Midsize	29.2	33.9	38.7	18	25
3	Audi	90	Compact	25.9	29.1	32.3	20	26
4	Audi	100	Midsize	30.8	37.7	44.6	19	26
5	BMW	535i	Midsize	23.7	30	36.2	22	30
6	Buick	Century	Midsize	14.2	15.7	17.3	22	31
7	Buick	LeSabre	Large	19.9	20.8	21.7	19	28
8	Buick	Roadmaster	Large	22.6	23.7	24.9	16	25
9	Buick	Riviera	Midsize	26.3	26.3	26.3	19	27
10	Cadillac	DeVille	Large	33	34.7	36.3	16	25
11	Cadillac	Seville	Midsize	37.5	40.1	42.7	16	25
12	Chevrolet	Cavalier	Compact	8.5	13.4	18.3	25	36
13	Chevrolet	Corsica	Compact	11.4	11.4	11.4	25	34
14	Chevrolet	Camaro	Sporty	13.4	15.1	16.8	19	28
15	Chevrolet	Lumina	Midsize	13.4	15.9	18.4	21	29
16	Chevrolet	Lumina_APV	Van	14.7	16.3	18	18	23
17	Chevrolet	Astro	Van	14.7	16.6	18.6	15	20
18	Chevrolet	Caprice	Large	18	18.8	19.6	17	26
19	Chevrolet	Corvette	Sporty	34.6	38	41.5	17	25
20	Chrylser	Concorde	Large	18.4	18.4	18.4	20	28
21	Chrysler	LeBaron	Compact	14.5	15.8	17.1	23	28
22	Chrysler	Imperial	Large	29.5	29.5	29.5	20	26
23	Dodge	Colt	Small	7.9	9.2	10.6	29	33
24	Dodge	Shadow	Small	8.4	11.3	14.2	23	29
25	Dodge	Spirit	Compact	11.9	13.3	14.7	22	27

R identifiers in a dataframe

colnames( )

rownames( )
or
index

	Manufacturer	Model	Type	Min.Price	Price	Max.Price	MPG.city	MPG.highway
1	Acura	Integra	Small	12.9	15.9	18.8	25	31
2	Acura	Legend	Midsize	29.2	33.9	38.7	18	25
3	Audi	90	Compact	25.9	29.1	32.3	20	26
4	Audi	100	Midsize	30.8	37.7	44.6	19	26
5	BMW	535i	Midsize	23.7	30	36.2	22	30
6	Buick	Century	Midsize	14.2	15.7	17.3	22	31
7	Buick	LeSabre	Large	19.9	20.8	21.7	19	28
8	Buick	Roadmaster	Large	22.6	23.7	24.9	16	25
9	Buick	Riviera	Midsize	26.3	26.3	26.3	19	27
10	Cadillac	DeVille	Large	33	34.7	36.3	16	25
11	Cadillac	Seville	Midsize	37.5	40.1	42.7	16	25
12	Chevrolet	Cavalier	Compact	8.5	13.4	18.3	25	36
13	Chevrolet	Corsica	Compact	11.4	11.4	11.4	25	34
14	Chevrolet	Camaro	Sporty	13.4	15.1	16.8	19	28
15	Chevrolet	Lumina	Midsize	13.4	15.9	18.4	21	29
16	Chevrolet	Lumina_APV	Van	14.7	16.3	18	18	23
17	Chevrolet	Astro	Van	14.7	16.6	18.6	15	20
18	Chevrolet	Caprice	Large	18	18.8	19.6	17	26
19	Chevrolet	Corvette	Sporty	34.6	38	41.5	17	25
20	Chrylser	Concorde	Large	18.4	18.4	18.4	20	28
21	Chrysler	LeBaron	Compact	14.5	15.8	17.1	23	28
22	Chrysler	Imperial	Large	29.5	29.5	29.5	20	26
23	Dodge	Colt	Small	7.9	9.2	10.6	29	33
24	Dodge	Shadow	Small	8.4	11.3	14.2	23	29
25	Dodge	Spirit	Compact	11.9	13.3	14.7	22	27

R variables in a dataframe

categorical vars | numerical vars | categorical vars

	Manufacturer	Model	Type	Min.Price	Price	Max.Price	MPG.city	MPG.highway	AirBags	DriveTrain
1	Acura	Integra	Small	12.9	15.9	18.8	25	31	None	Front
2	Acura	Legend	Midsize	29.2	33.9	38.7	18	25	Driver & Passenger	Front
3	Audi	90	Compact	25.9	29.1	32.3	20	26	Driver only	Front
4	Audi	100	Midsize	30.8	37.7	44.6	19	26	Driver & Passenger	Front
5	BMW	535i	Midsize	23.7	30	36.2	22	30	Driver only	Rear
6	Buick	Century	Midsize	14.2	15.7	17.3	22	31	Driver only	Front
7	Buick	LeSabre	Large	19.9	20.8	21.7	19	28	Driver only	Front
8	Buick	Roadmaster	Large	22.6	23.7	24.9	16	25	Driver only	Rear
9	Buick	Riviera	Midsize	26.3	26.3	26.3	19	27	Driver only	Front
10	Cadillac	DeVille	Large	33	34.7	36.3	16	25	Driver only	Front
11	Cadillac	Seville	Midsize	37.5	40.1	42.7	16	25	Driver & Passenger	Front
12	Chevrolet	Cavalier	Compact	8.5	13.4	18.3	25	36	None	Front
13	Chevrolet	Corsica	Compact	11.4	11.4	11.4	25	34	Driver only	Front
14	Chevrolet	Camaro	Sporty	13.4	15.1	16.8	19	28	Driver & Passenger	Rear
15	Chevrolet	Lumina	Midsize	13.4	15.9	18.4	21	29	None	Front
16	Chevrolet	Lumina_APV	Van	14.7	16.3	18	18	23	None	Front
17	Chevrolet	Astro	Van	14.7	16.6	18.6	15	20	None	4WD
18	Chevrolet	Caprice	Large	18	18.8	19.6	17	26	Driver only	Rear
19	Chevrolet	Corvette	Sporty	34.6	38	41.5	17	25	Driver only	Rear
20	Chrylser	Concorde	Large	18.4	18.4	18.4	20	28	Driver & Passenger	Front
21	Chrysler	LeBaron	Compact	14.5	15.8	17.1	23	28	Driver & Passenger	Front

R data visualization

- line plot
- bar plot
- scatterplot
- histogram
- boxplot

Example 1 – Cars93

Library MASS - Cars93

A dataset with 93 cars and 27 cols.

	Manufacturer	Model	Type	Min.Price	Price	Max.Price	MPG.city	MPG.highway	AirBags	DriveTrain	Cylinders	EngineSize	Horsepower	RPM
1	Acura	Integra	Small	12.9	15.9	18.8	25	31	None	Front	4	1.8	140	6300
2	Acura	Legend	Midsize	29.2	33.9	38.7	18	25	Driver & Passenger	Front	6	3.2	200	5500
3	Audi	90	Compact	25.9	29.1	32.3	20	26	Driver only	Front	6	2.8	172	5500
4	Audi	100	Midsize	30.8	37.7	44.6	19	26	Driver & Passenger	Front	6	2.8	172	5500
5	BMW	535i	Midsize	23.7	30	36.2	22	30	Driver only	Rear	4	3.5	208	5700
6	Buick	Century	Midsize	14.2	15.7	17.3	22	31	Driver only	Front	4	2.2	110	5200
7	Buick	LeSabre	Large	19.9	20.8	21.7	19	28	Driver only	Front	6	3.8	170	4800
8	Buick	Roadmaster	Large	22.6	23.7	24.9	16	25	Driver only	Rear	6	5.7	180	4000
9	Buick	Riviera	Midsize	26.3	26.3	26.3	19	27	Driver only	Front	6	3.8	170	4800
10	Cadillac	DeVille	Large	33	34.7	36.3	16	25	Driver only	Front	8	4.9	200	4100
11	Cadillac	Seville	Midsize	37.5	40.1	42.7	16	25	Driver & Passenger	Front	8	4.6	295	6000
12	Chevrolet	Cavalier	Compact	8.5	13.4	18.3	25	36	None	Front	4	2.2	110	5200
13	Chevrolet	Corsica	Compact	11.4	11.4	11.4	25	34	Driver only	Front	4	2.2	110	5200
14	Chevrolet	Camaro	Sporty	13.4	15.1	16.8	19	28	Driver & Passenger	Rear	6	3.4	160	4600
15	Chevrolet	Lumina	Midsize	13.4	15.9	18.4	21	29	None	Front	4	2.2	110	5200
16	Chevrolet	Lumina_APV	Van	14.7	16.3	18	18	23	None	Front	6	3.8	170	4800
17	Chevrolet	Astro	Van	14.7	16.6	18.6	15	20	None	4WD	6	4.3	165	4000
18	Chevrolet	Caprice	Large	18	18.8	19.6	17	26	Driver only	Rear	8	5	170	4200
19	Chevrolet	Corvette	Sporty	34.6	38	41.5	17	25	Driver only	Rear	8	5.7	300	5000
20	Chrylser	Concorde	Large	18.4	18.4	18.4	20	28	Driver & Passenger	Front	6	3.3	153	5300
21	Chrysler	LeBaron	Compact	14.5	15.8	17.1	23	28	Driver & Passenger	Front	4	3	141	5000
22	Chrysler	Imperial	Large	29.5	29.5	29.5	20	26	Driver only	Front	6	3.3	147	4800
23	Dodge	Colt	Small	7.9	9.2	10.6	29	33	None	Front	4	1.5	92	6000
24	Dodge	Shadow	Small	8.4	11.3	14.2	23	29	Driver only	Front	4	2.2	93	4800
25	Dodge	Spirit	Compact	11.9	13.3	14.7	22	27	Driver only	Front	4	2.5	100	4800
26	Dodge	Caravan	Van	13.6	19	24.4	17	21	Driver only	4WD	6	3	142	5000
27	Dodge	Dynasty	Midsize	14.8	15.6	16.4	21	27	Driver only	Front	4	2.5	100	4800
28	Dodge	Stealth	Sporty	18.5	25.8	33.1	18	24	Driver only	4WD	6	3	300	6000
29	Eagle	Summit	Small	7.9	12.2	16.5	29	33	None	Front	4	1.5	92	6000
30	Eagle	Vision	Large	17.5	19.3	21.2	20	28	Driver & Passenger	Front	6	3.5	214	5800
31	Ford	Festiva	Small	6.9	7.4	7.9	31	33	None	Front	4	1.3	63	5000
32	Ford	Escort	Small	8.4	10.1	11.9	23	30	None	Front	4	1.8	127	6500
33	Ford	Tempo	Compact	10.4	11.3	12.2	22	27	None	Front	4	2.3	96	4200

The structure function str( )

```
> library(MASS)
> d1=Cars93
> str(d1)
'data.frame':   93 obs. of  27 variables:
 $ Manufacturer      : Factor w/ 32 levels "Acura","Audi",..: 1 1 2 2 3 4 4 4 4 5 ...
 $ Model             : Factor w/ 93 levels "100","190E","240",..: 49 56 9 1 6 24 54 74 73 35 ...
 $ Type              : Factor w/ 6 levels "Compact","Large",..: 4 3 1 3 3 3 2 2 3 2 ...
 $ Min.Price         : num  12.9 29.2 25.9 30.8 23.7 14.2 19.9 22.6 26.3 33 ...
 $ Price             : num  15.9 33.9 29.1 37.7 30 15.7 20.8 23.7 26.3 34.7 ...
 $ Max.Price         : num  18.8 38.7 32.3 44.6 36.2 17.3 21.7 24.9 26.3 36.3 ...
 $ MPG.city          : int  25 18 20 19 22 22 19 16 19 16 ...
 $ MPG.highway       : int  31 25 26 26 30 31 28 25 27 25 ...
 $ AirBags           : Factor w/ 3 levels "Driver & Passenger",..: 3 1 2 1 2 2 2 2 2 2 ...
 $ DriveTrain        : Factor w/ 3 levels "4WD","Front",..: 2 2 2 2 3 2 2 3 2 2 ...
 $ Cylinders         : Factor w/ 6 levels "3","4","5","6",..: 2 4 4 4 2 2 4 4 4 5 ...
 $ EngineSize        : num  1.8 3.2 2.8 2.8 3.5 2.2 3.8 5.7 3.8 4.9 ...
 $ Horsepower        : int  140 200 172 172 208 110 170 180 170 200 ...
 $ RPM               : int  6300 5500 5500 5500 5700 5200 4800 4000 4800 4100 ...
 $ Rev.per.mile      : int  2890 2335 2280 2535 2545 2565 1570 1320 1690 1510 ...
 $ Man.trans.avail   : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 1 1 1 1 1 ...
 $ Fuel.tank.capacity: num  13.2 18 16.9 21.1 21.1 16.4 18 23 18.8 18 ...
 $ Passengers        : int  5 5 5 6 4 6 6 6 5 6 ...
 $ Length            : int  177 195 180 193 186 189 200 216 198 206 ...
 $ Wheelbase         : int  102 115 102 106 109 105 111 116 108 114 ...
 $ Width             : int  68 71 67 70 69 69 74 78 73 73 ...
 $ Turn.circle       : int  37 38 37 37 39 41 42 45 41 43 ...
 $ Rear.seat.room    : num  26.5 30 28 31 27 28 30.5 30.5 26.5 35 ...
 $ Luggage.room      : int  11 15 14 17 13 16 17 21 14 18 ...
 $ Weight            : int  2705 3560 3375 3405 3640 2880 3470 4105 3495 3620 ...
 $ Origin            : Factor w/ 2 levels "USA","non-USA": 2 2 2 2 2 1 1 1 1 1 ...
 $ Make              : Factor w/ 93 levels "Acura Integra",..: 1 2 4 3 5 6 7 9 8 10 ...
```

27 columns

dim( ), nrow( ), and head( ) functions

```
> dim(d1)
[1] 93 27
> n = nrow(d1)
> n
[1] 93
> head(d1)
```

displays first six rows

	Manufacturer	Model	Type	Min.Price	Price	Max.Price	MPG.city	MPG.highway
1	Acura	Integra	Small	12.9	15.9	18.8	25	31
2	Acura	Legend	Midsize	29.2	33.9	38.7	18	25
3	Audi	90	Compact	25.9	29.1	32.3	20	26
4	Audi	100	Midsize	30.8	37.7	44.6	19	26
5	BMW	535i	Midsize	23.7	30.0	36.2	22	30
6	Buick	Century	Midsize	14.2	15.7	17.3	22	31

	AirBags	DriveTrain	Cylinders	EngineSize	Horsepower	RPM	Rev.per.mile
1	None	Front	4	1.8	140	6300	2890
2	Driver & Passenger	Front	6	3.2	200	5500	2335
3	Driver only	Front	6	2.8	172	5500	2280
4	Driver & Passenger	Front	6	2.8	172	5500	2535
5	Driver only	Rear	4	3.5	208	5700	2545
6	Driver only	Front	4	2.2	110	5200	2565

The combine function c( )

- useful to create a vector

  c(1,3,4.5) gives [1] 1 3 4.5

- all elements in a vector must be of same type

  c(1,3,"a") gives [1] "1" "3" "a"

- to store vector in memory

  x = c(1,3,4.5)

- Vectors can be used for subsetting dataframes

Subsetting

selecting a portion of a dataset

Subsetting (querying) dataframes

- columns 2,4, and 9 (all rows)

```
d1[c(2,4,9)]
```

- rows 1 to 9 (all columns)

```
d1[c(1:9),]
```

- both

```
d1[c(1:9),c(2,4,9)]
```

Selecting columns of a dataframe

```
> d2 = d1[c("Model","Price","AirBags")]
> head(d2)
    Model Price             AirBags
1 Integra  15.9                None
2  Legend  33.9 Driver & Passenger
3      90  29.1         Driver only
4     100  37.7 Driver & Passenger
5    535i  30.0         Driver only
6 Century  15.7         Driver only
```

Selecting columns of a dataframe

```
> d2 = d1[c("Model","Price","AirBags")]
> head(d2)
    Model Price            AirBags
1 Integra  15.9               None
2  Legend  33.9 Driver & Passenger
3      90  29.1        Driver only
4     100  37.7 Driver & Passenger
5    535i  30.0        Driver only
6 Century  15.7        Driver only
```

```
> d2 = d1[c(2,5,9)]
> head(d2)
    Model Price            AirBags
1 Integra  15.9               None
2  Legend  33.9 Driver & Passenger
3      90  29.1        Driver only
4     100  37.7 Driver & Passenger
5    535i  30.0        Driver only
6 Century  15.7        Driver only
```

Selecting a row

```
> d1[10,]
   Manufacturer   Model  Type Min.Price Price Max.Price MPG.city MPG.highway     AirBags
10     Cadillac DeVille Large        33  34.7      36.3       16           25 Driver only
   DriveTrain Cylinders EngineSize Horsepower  RPM Rev.per.mile Man.trans.avail
10      Front         8        4.9        200 4100         1510              No
   Fuel.tank.capacity Passengers Length Wheelbase Width Turn.circle Rear.seat.room Luggage.room
10                 18          6    206       114    73          43             35           18
   Weight Origin             Make
10   3620    USA Cadillac DeVille
```

Not specifying columns, selects **all** columns

Selecting rows and columns of a dataframe

```
> d2 = d1[c(1:3),c("Model","Price","AirBags")]
> d2
    Model Price            AirBags
1 Integra  15.9               None
2  Legend  33.9 Driver & Passenger
3      90  29.1        Driver only
```

```
> d2 = d1[c(1:3),c(2,5,9)]
> d2
    Model Price            AirBags
1 Integra  15.9               None
2  Legend  33.9 Driver & Passenger
3      90  29.1        Driver only
```

Selecting columns

- using the $ operator

```
> d2 = data.frame(d1$Manufacturer,d1$Price)
> head(d2)
  d1.Manufacturer d1.Price
1           Acura     15.9
2           Acura     33.9
3            Audi     29.1
4            Audi     37.7
5             BMW     30.0
6           Buick     15.7
```

R function subset

```
> d2 = subset(d1,select=c(Manufacturer,Price))
> head(d2)
  Manufacturer Price
1       Acura  15.9
2       Acura  33.9
3        Audi  29.1
4        Audi  37.7
5         BMW  30.0
6       Buick  15.7
```

Selecting rows by condition

```
> # Name and Price of cars weighting > 4000
> #
> d2 = d1[d1$Weight>4000,c(1:3,5)]
> d2
   Manufacturer      Model  Type Price
8         Buick Roadmaster Large  23.7
17    Chevrolet      Astro   Van  16.6
52      Lincoln   Town_Car Large  36.1
66       Nissan      Quest   Van  19.1
```

Selecting rows by condition

```
> # Name and Price of cars weighting > 4000
> #
> d2 = d1[d1$Weight>4000,c(1:3,5)]
> d2
   Manufacturer      Model  Type Price
8         Buick Roadmaster Large  23.7
17    Chevrolet      Astro   Van  16.6
52      Lincoln   Town_Car Large  36.1
66       Nissan      Quest   Van  19.1
```

```
> # most expensive car
> #
> d1[d1$Price == max(d1$Price),1:3]
    Manufacturer Model    Type
59 Mercedes-Benz  300E Midsize
```

Selecting rows by condition

```
> # Name and Price of cars weighting > 4000
> #
> d2 = d1[d1$Weight>4000,c(1:3,5)]
> d2
   Manufacturer      Model  Type Price
8         Buick Roadmaster Large  23.7
17    Chevrolet      Astro   Van  16.6
52      Lincoln   Town_Car Large  36.1
66       Nissan      Quest   Van  19.1
```

```
> # most expensive car
> #
> d1[d1$Price == max(d1$Price),1:3]
    Manufacturer Model    Type
59 Mercedes-Benz  300E Midsize
```

```
> # how many exceeding 3000 lbs?
> aux = d1$Weight
> cars1=aux[aux>3000]
> length(cars1)
[1] 48
```

Selecting rows by condition – function subset( )

```
> # Ford cars -all cols
> d2 = subset(d1,subset = Manufacturer=="Ford")
> dim(d2)
[1]  8 27
```

```
> # Ford cars -Price col only
> d2 = subset(d1,Price,subset = Manufacturer=="Ford")
> d2
   Price
31   7.4
32  10.1
33  11.3
34  15.9
35  14.0
36  19.9
37  20.2
38  20.9
```

Selecting rows by condition – function subset

```
> # Ford cars -all cols
> d2 = subset(d1,subset = Manufacturer=="Ford")
> dim(d2)
[1]  8 27
```

```
> # Ford cars -Price col only
> d2 = subset(d1,Price,subset = Manufacturer=="Ford")
> d2
   Price
31   7.4
32  10.1
33  11.3
34  15.9
35  14.0
36  19.9
37  20.2
38  20.9
```

```
> # Ford cars -Manufacturer and Price cols only
> d2 = subset(d1,c(Manufacturer,Price),subset = Manufacturer=="Ford")
> d2
   Manufacturer Price
31         Ford   7.4
32         Ford  10.1
33         Ford  11.3
34         Ford  15.9
35         Ford  14.0
36         Ford  19.9
37         Ford  20.2
38         Ford  20.9
```

Selecting rows – multiple conditions

```
> # select Ford and Nissan cars (all columns)
> d2 = subset(d1,subset=Manufacturer=="Ford"|Manufacturer=="Nissan")
> dim(d2)
[1] 12 27
>
>
> # select Ford and Nissan cars (two columns only)
> d2 = subset(d1,c(Manufacturer,Price),subset=Manufacturer=="Ford"|Manufacturer=="Nissan")
> d2
   Manufacturer Price
31          Ford   7.4
32          Ford  10.1
33          Ford  11.3
34          Ford  15.9
35          Ford  14.0
36          Ford  19.9
37          Ford  20.2
38          Ford  20.9
64        Nissan  11.8
65        Nissan  15.7
66        Nissan  19.1
67        Nissan  21.5
```

Summary Table

numeric and categorical columns

Summary Table – categorical and numeric cols.

```
> d2 = subset(d1,select=c(Manufacturer,Type,Price,DriveTrain,Weight))
> summary(d2)
    Manufacturer      Type        Price       DriveTrain     Weight
 Chevrolet: 8    Compact:16   Min.   : 7.40   4WD  :10   Min.   :1695
 Ford     : 8    Large  :11   1st Qu.:12.20   Front:67   1st Qu.:2620
 Dodge    : 6    Midsize:22   Median :17.70   Rear :16   Median :3040
 Mazda    : 5    Small  :21   Mean   :19.51              Mean   :3073
 Pontiac  : 5    Sporty :14   3rd Qu.:23.30              3rd Qu.:3525
 Buick    : 4    Van    : 9   Max.   :61.90              Max.   :4105
 (Other)  :57
```

Summary Table – categorical and numeric cols.

```
> d2 = subset(d1,select=c(Manufacturer,Type,Price,DriveTrain,Weight))
> summary(d2)
    Manufacturer      Type         Price       DriveTrain     Weight
Chevrolet: 8    Compact:16   Min.   : 7.40   4WD  :10   Min.   :1695
Ford     : 8    Large  :11   1st Qu.:12.20   Front:67   1st Qu.:2620
Dodge    : 6    Midsize:22   Median :17.70   Rear :16   Median :3040
Mazda    : 5    Small  :21   Mean   :19.51              Mean   :3073
Pontiac  : 5    Sporty :14   3rd Qu.:23.30              3rd Qu.:3525
Buick    : 4    Van    : 9   Max.   :61.90              Max.   :4105
(Other)  :57
```

Summary of categorical columns ≠ Summary of numerical columns

Cross tabulation

number of rows by category

category = level

Cross tabulation

- Tables for counting rows in each category
- R functions

    summary()

    table( )

    prop.table( )

    xtabs( )

    ftable( )

Counting by categories

One categorical variable

- table( )
- prop.table(table( ))

Two categorical vars

- xtabs( )
- ftable( )

Cross tabulation – one factor (DriveTrain)

```
> # identify categories (factor levels) of DriveTrain
> levels(d1$DriveTrain)
[1] "4WD"   "Front" "Rear"
```

```
> # number of cars by DriveTrain
> table(d1$DriveTrain)

  4WD Front  Rear
   10    67    16
```

```
> # fraction of cars by DriveTrain
> prop.table(table(d1$DriveTrain))

      4WD     Front      Rear
0.1075269 0.7204301 0.1720430
```

Cross tabulation – 2 factors (AirBags, DriveTrain)

```
# number of cars by AirBags & DriveTrain
#
table(d1$AirBags,d1$DriveTrain)
```

```
##
##                       4WD Front Rear
##    Driver & Passenger   0    11    5
##    Driver only          5    28   10
##    None                 5    28    1
```

Cross tabulation – 2 factors (AirBags, DriveTrain)

```
# number of cars by AirBags & DriveTrain
#
table(d1$AirBags,d1$DriveTrain)
```

```
##
##                       4WD Front Rear
##   Driver & Passenger    0    11    5
##   Driver only           5    28   10
##   None                  5    28    1
```

```
t1 = table(d1$AirBags,d1$DriveTrain)
# add column with row totals
addmargins(t1,margin=2)
```

row totals

```
##
##                       4WD Front Rear Sum
##   Driver & Passenger    0    11    5  16
##   Driver only           5    28   10  43
##   None                  5    28    1  34
```

```
# add both row and col totals
addmargins(t1)
```

```
##
##                       4WD Front Rear Sum
##   Driver & Passenger    0    11    5  16
##   Driver only           5    28   10  43
##   None                  5    28    1  34
##   Sum                  10    67   16  93
```

Cross tabulation – 2 factors (AirBags, DriveTrain)

```
# number of cars by AirBags & DriveTrain
#
table(d1$AirBags,d1$DriveTrain)
```

```
##
##                       4WD Front Rear
##   Driver & Passenger    0    11    5
##   Driver only           5    28   10
##   None                  5    28    1
```

table data structure

```
t1 = table(d1$AirBags,d1$DriveTrain)
# add column with row totals
addmargins(t1,margin=2)
```

```
##
##                       4WD Front Rear Sum
##   Driver & Passenger    0    11    5  16
##   Driver only           5    28   10  43
##   None                  5    28    1  34
```

```
# add both row and col totals
addmargins(t1)
```

```
##
##                       4WD Front Rear Sum
##   Driver & Passenger    0    11    5  16
##   Driver only           5    28   10  43
##   None                  5    28    1  34
##   Sum                  10    67   16  93
```

Cross tabulation – 2 factors (table formats)

```
a = table(AirBags = d1$AirBags,DriveTrain=d1$DriveTrain)
data.frame(a)
```

```
##              AirBags DriveTrain Freq
## 1 Driver & Passenger        4WD    0
## 2        Driver only        4WD    5
## 3               None        4WD    5
## 4 Driver & Passenger      Front   11
## 5        Driver only      Front   28
## 6               None      Front   28
## 7 Driver & Passenger       Rear    5
## 8        Driver only       Rear   10
## 9               None       Rear    1
```

long table format

(convert table to a dataframe)

Cross tabulation – 2 factors (table formats)

```
a = table(AirBags = d1$AirBags,DriveTrain=d1$DriveTrain)
data.frame(a)
```

```
##              AirBags DriveTrain Freq
## 1 Driver & Passenger        4WD    0
## 2        Driver only        4WD    5
## 3               None        4WD    5
## 4 Driver & Passenger      Front   11
## 5        Driver only      Front   28
## 6               None      Front   28
## 7 Driver & Passenger       Rear    5
## 8        Driver only       Rear   10
## 9               None       Rear    1
```

long table format

(convert table to a dataframe)

short table format

```
# number of cars by AirBags & DriveTrain
#
table(d1$AirBags,d1$DriveTrain)
```

```
##
##                      4WD Front Rear
##   Driver & Passenger   0    11    5
##   Driver only          5    28   10
##   None                 5    28    1
```

Cross tabulation – 2 factors (table formats)

```
a = table(AirBags = d1$AirBags,DriveTrain=d1$DriveTrain)
data.frame(a)
```

```
##                AirBags DriveTrain Freq
## 1 Driver & Passenger         4WD    0
## 2         Driver only         4WD    5
## 3                None         4WD    5
## 4 Driver & Passenger       Front   11
## 5         Driver only       Front   28
## 6                None       Front   28
## 7 Driver & Passenger        Rear    5
## 8         Driver only        Rear   10
## 9                None        Rear    1
```

long table format

(convert table to a dataframe)

short table format

```
# number of cars by AirBags & DriveTrain
#
table(d1$AirBags,d1$DriveTrain)
```

```
##
##                      4WD Front Rear
##   Driver & Passenger   0    11    5
##   Driver only          5    28   10
##   None                 5    28    1
```

Cross tabulation – 3 factors

```
# or long-table format (naming the cols is important)
#
a = table(AirBags = d1$AirBags,DriveTrain=d1$DriveTrain,Origin = d1$Origin)
data.frame(a)
```

```
##                 AirBags DriveTrain  Origin Freq
## 1  Driver & Passenger          4WD     USA    0
## 2          Driver only          4WD     USA    3
## 3                 None          4WD     USA    2
## 4  Driver & Passenger        Front     USA    6
## 5          Driver only        Front     USA   15
## 6                 None        Front     USA   13
## 7  Driver & Passenger         Rear     USA    3
## 8          Driver only         Rear     USA    5
## 9                 None         Rear     USA    1
## 10 Driver & Passenger          4WD non-USA    0
## 11         Driver only          4WD non-USA    2
## 12                None          4WD non-USA    3
## 13 Driver & Passenger        Front non-USA    5
## 14         Driver only        Front non-USA   13
## 15                None        Front non-USA   15
## 16 Driver & Passenger         Rear non-USA    2
## 17         Driver only         Rear non-USA    5
## 18                None         Rear non-USA    0
```

short table format

```
> ftable(d1$AirBags,d1$DriveTrain,d1$Origin)
                                 USA non-USA ↵
Driver & Passenger 4WD             0       0
                   Front           6       5
                   Rear            3       2
Driver only        4WD             3       2
                   Front          15      13
                   Rear            5       5
None               4WD             2       3
                   Front          13      15
                   Rear            1       0
```

Cross tabulation – formula format

```
# xtabs works with formula
xtabs(~Passengers+AirBags,d1)
```

xtabs data structure

```
##                                AirBags
## Passengers | Driver & Passenger | Driver only | None
##          2 |                   0 |           2 |    0
##          4 |                   5 |           8 |   10
##          5 |                   4 |          20 |   17
##          6 |                   7 |          10 |    1
##          7 |                   0 |           3 |    5
##          8 |                   0 |           0 |    1
```

Cross tabulation – formula format

```
# xtabs works with formula
xtabs(~Passengers+AirBags,d1)
```

```
##                                AirBags
## Passengers | Driver & Passenger | Driver only | None
##          2 |                   0 |           2 |    0
##          4 |                   5 |           8 |   10
##          5 |                   4 |          20 |   17
##          6 |                   7 |          10 |    1
##          7 |                   0 |           3 |    5
##          8 |                   0 |           0 |    1
```

```
t2 = xtabs(~Passengers+AirBags,d1)
addmargins(t2)
```

```
##                                AirBags
## Passengers | Driver & Passenger | Driver only | None | Sum
##          2 |                   0 |           2 |    0 |   2
##          4 |                   5 |           8 |   10 |  23
##          5 |                   4 |          20 |   17 |  41
##          6 |                   7 |          10 |    1 |  18
##          7 |                   0 |           3 |    5 |   8
##          8 |                   0 |           0 |    1 |   1
## ---------------------------------------------------------
##        Sum |                  16 |          43 |   34 |  93
```

Cross tabulation – more than one factor

```
# xtabs works with formula
xtabs(~Passengers+AirBags,d1)
```

```
##            AirBags
## Passengers  Driver & Passenger  Driver only  None
##          2                   0            2     0
##          4                   5            8    10
##          5                   4           20    17
##          6                   7           10     1
##          7                   0            3     5
##          8                   0            0     1
```

long table format (converting xtabs to a dataframe)

```
t2 = xtabs(~Passengers+AirBags,d1)
d2 = data.frame(t2)
d2
```

```
##    Passengers            AirBags Freq
## 1           2 Driver & Passenger    0
## 2           4 Driver & Passenger    5
## 3           5 Driver & Passenger    4
## 4           6 Driver & Passenger    7
## 5           7 Driver & Passenger    0
## 6           8 Driver & Passenger    0
## 7           2        Driver only    2
## 8           4        Driver only    8
## 9           5        Driver only   20
## 10          6        Driver only   10
## 11          7        Driver only    3
## 12          8        Driver only    0
## 13          2               None    0
## 14          4               None   10
## 15          5               None   17
## 16          6               None    1
## 17          7               None    5
## 18          8               None    1
```

Cross tabulation – proportions

```
# proportion of cars by AirBags & DriveTrain
#
t1 = table(d1$AirBags,d1$DriveTrain)
prop.table(t1)
```

```
##
##                              4WD      Front       Rear
##   Driver & Passenger 0.00000000 0.11827957 0.05376344
##   Driver only        0.05376344 0.30107527 0.10752688
##   None               0.05376344 0.30107527 0.01075269
```

```
#
# proportion over rows
prop.table(t1,margin=1)
```

1

```
##
##                              4WD      Front       Rear
##   Driver & Passenger 0.00000000 0.68750000 0.31250000
##   Driver only        0.11627907 0.65116279 0.23255814
##   None               0.14705882 0.82352941 0.02941176
```

1
1
1

Pivot Tables

Pivot Tables

- R functions
  - apply( )
  - tapply( )
  - aggregate( )

Applying a function to numeric columns

- function apply( )

  apply(d1,2,mean) to find the mean of each column

Applying a function to numeric columns

- function apply( )

  apply(d1,2,mean)

- use 1 to apply the function by rows

Applying a function to numeric columns

- function apply( )

  apply(d1,2,mean)

- use 1 to apply the function by rows
- won't work apply(d1,2, mean – 3.5)
- will do apply(d1,2, function(x) mean(x) – 3.5)

Pivot table – tapply()

To find a summary measure of a numeric column classified by the levels of a factor

summary measure numeric factor

```
> # median weight per DriveTrain
```

Pivot table – tapply()

To find a summary measure of a numeric column classified by the levels of a factor

summary measure

numeric

factor

measure

```
> # median weight per DriveTrain
> tapply(d1$Weight,d1$DriveTrain,median)
```

Pivot table – tapply()

To find a summary measure of a numeric column classified by the levels of a factor

```
> # median weight per DriveTrain
> tapply(d1$Weight,d1$DriveTrain,median)
  4WD Front  Rear
 3720  2910  3520
```

- This is a pivot table using one factor (DriveTrain)
- The result is not a dataframe

Pivot table – **two** factors

summary measure numeric factor 1 factor 2

```
> # median weight per Airbags & DriveTrain
```

Pivot table – two factors

summary measure — numeric — factor 1 — factor 2

```
> # median weight per Airbags & DriveTrain
> aux = list(d1$AirBags,d1$DriveTrain)
> tapply(d1$Weight,aux,median)
                     4WD  Front Rear
Driver & Passenger    NA 3490.0 3515
Driver only         3735 2970.0 3510
None                2640 2552.5 3610
```

Pivot table – two factors

```
> # median weight per Airbags & DriveTrain
> aux = list(d1$AirBags,d1$DriveTrain)
> tapply(d1$Weight,aux,median)
                     4WD  Front Rear
Driver & Passenger    NA 3490.0 3515
Driver only         3735 2970.0 3510
None                2640 2552.5 3610
```

Pivot table – one factor

```
> # median weight per DriveTrain
> tapply(d1$Weight,d1$DriveTrain,median)
  4WD Front  Rear
 3720  2910  3520
```

aggregate( ) to get Pivot Table as a dataframe

summary
measure numeric columns factor

Average (Price, Weight and Length) of cars by DriveTrain

aggregate( ) to get Pivot Table as a dataframe

Average (Price, Weight and Length) of cars by DriveTrain

```
aggregate( cbind(Price,Weight,Length) ~ DriveTrain,data=d1,mean )
```

must use the formula syntax

```
##   DriveTrain    Price   Weight   Length
## 1        4WD 17.63000 3305.000 177.5000
## 2      Front 17.53582 2938.955 181.4478
## 3       Rear 28.95000 3488.750 194.1250
```

aggregate( ) to get Pivot Table as a dataframe

summary measure

numeric columns

factor

Average (Price, Weight and Length) of cars by DriveTrain

```
aggregate( cbind(Price,Weight,Length) ~ DriveTrain,data=d1,mean,na.rm = TRUE)
```

```
##   DriveTrain    Price   Weight   Length
## 1        4WD 17.63000 3305.000 177.5000
## 2      Front 17.53582 2938.955 181.4478
## 3       Rear 28.95000 3488.750 194.1250
```

to avoid getting errors if NA values exist

aggregate( ) to get Pivot Table as a dataframe

summary measure — numeric columns — two factors

Average (Price, Weight and Length) of cars by DriveTrain and AirBags

```
aggregate( cbind(Price,Weight,Length) ~ AirBags + DriveTrain,data=d1,mean,na.rm = TRUE)
```

```
##              AirBags DriveTrain    Price   Weight   Length
## 1        Driver only        4WD 21.38000 3623.000 179.4000
## 2               None        4WD 13.88000 2987.000 175.6000
## 3 Driver & Passenger      Front 26.17273 3393.636 191.4545
## 4        Driver only      Front 18.69286 2996.250 184.1071
## 5               None      Front 12.98571 2703.036 174.8571
## 6 Driver & Passenger       Rear 33.20000 3515.000 197.2000
## 7        Driver only       Rear 28.23000 3463.500 192.1000
## 8               None       Rear 14.90000 3610.000 199.0000
```

Split numeric columns from factor columns

Function sapply( )

Applies a function to a dataframe

sapply(dataframe, function)

Example with a simplified dataframe

```
d4 = subset(d1,select=c(AirBags,DriveTrain,Origin,Price,Weight,Width))
str(d4)
```

```
## 'data.frame':    93 obs. of  6 variables:
##  $ AirBags   : Factor w/ 3 levels "Driver & Passenger",..: 3 1 2 1 2 2
##  $ DriveTrain: Factor w/ 3 levels "4WD","Front",..: 2 2 2 2 3 2 2 3 2 2
##  $ Origin    : Factor w/ 2 levels "USA","non-USA": 2 2 2 2 2 1 1 1 1 1
##  $ Price     : num  15.9 33.9 29.1 37.7 30 15.7 20.8 23.7 26.3 34.7 ..
##  $ Weight    : int  2705 3560 3375 3405 3640 2880 3470 4105 3495 3620
##  $ Width     : int  68 71 67 70 69 69 74 78 73 73 ...
```

Example – dataframe with numeric columns only

```
d4 = subset(d1,select=c(AirBags,DriveTrain,Origin,Price,Weight,Width))
str(d4)
```

```
## 'data.frame':    93 obs. of  6 variables:
##  $ AirBags   : Factor w/ 3 levels "Driver & Passenger",..: 3 1 2 1 2 2
##  $ DriveTrain: Factor w/ 3 levels "4WD","Front",..: 2 2 2 2 3 2 2 3 2 2
##  $ Origin    : Factor w/ 2 levels "USA","non-USA": 2 2 2 2 2 1 1 1 1 1
##  $ Price     : num  15.9 33.9 29.1 37.7 30 15.7 20.8 23.7 26.3 34.7 ...
##  $ Weight    : int  2705 3560 3375 3405 3640 2880 3470 4105 3495 3620 ...
##  $ Width     : int  68 71 67 70 69 69 74 78 73 73 ...
```

```
# select numeric columns
#
dnum = d4[,sapply(d4, is.numeric)]
str(dnum)
```

```
## 'data.frame':    93 obs. of  3 variables:
##  $ Price : num  15.9 33.9 29.1 37.7 30 15.7 20.8 23.7 26.3 34.7 ...
##  $ Weight: int  2705 3560 3375 3405 3640 2880 3470 4105 3495 3620 ...
##  $ Width : int  68 71 67 70 69 69 74 78 73 73 ...
```

Example – dataframe with factor columns only

```
d4 = subset(d1,select=c(AirBags,DriveTrain,Origin,Price,Weight,Width))
str(d4)
```

```
## 'data.frame':    93 obs. of  6 variables:
##  $ AirBags    : Factor w/ 3 levels "Driver & Passenger",..: 3 1 2 1 2 2
##  $ DriveTrain: Factor w/ 3 levels "4WD","Front",..: 2 2 2 2 3 2 2 3 2 
##  $ Origin     : Factor w/ 2 levels "USA","non-USA": 2 2 2 2 2 1 1 1 1 1
##  $ Price      : num  15.9 33.9 29.1 37.7 30 15.7 20.8 23.7 26.3 34.7 ..
##  $ Weight     : int  2705 3560 3375 3405 3640 2880 3470 4105 3495 3620
##  $ Width      : int  68 71 67 70 69 69 74 78 73 73 ...
```

```
# select factor columns
#
dfactors = d4[,sapply(d4, is.factor)]
str(dfactors)
```

```
## 'data.frame':    93 obs. of  3 variables:
##  $ AirBags    : Factor w/ 3 levels "Driver & Passenger",..: 3 1 2 1
##  $ DriveTrain: Factor w/ 3 levels "4WD","Front",..: 2 2 2 2 3 2 2
##  $ Origin     : Factor w/ 2 levels "USA","non-USA": 2 2 2 2 2 1 1 1
```

Find levels from factor columns

```
# select factor columns
#
dfactors = d4[,sapply(d4, is.factor)]
str(dfactors)
```

```
## 'data.frame':    93 obs. of  3 variables:
##  $ AirBags    : Factor w/ 3 levels "Driver & Passenger",..: 3 1 2 1
##  $ DriveTrain: Factor w/ 3 levels "4WD","Front",..: 2 2 2 2 3 2 2
##  $ Origin     : Factor w/ 2 levels "USA","non-USA": 2 2 2 2 2 1 1 1
```

```
# find levels of all factor columns
#
sapply(dfactors,levels)
```

```
## $AirBags
## [1] "Driver & Passenger" "Driver only"        "None"
##
## $DriveTrain
## [1] "4WD"   "Front" "Rear"
##
## $Origin
## [1] "USA"     "non-USA"
```

Sorting

by columns

Sorting – one column

```
> d2 = subset(d1,select=c(Manufacturer,Price,Weight,Width))
> head(d2)
  Manufacturer Price Weight Width
1        Acura  15.9   2705    68
2        Acura  33.9   3560    71
3         Audi  29.1   3375    67
4         Audi  37.7   3405    70
5          BMW  30.0   3640    69
6        Buick  15.7   2880    69
```

Sorting – one column

```
> d2 = subset(d1,select=c(Manufacturer,Price,Weight,Width))
> head(d2)
  Manufacturer Price Weight Width
1        Acura  15.9   2705    68
2        Acura  33.9   3560    71
3         Audi  29.1   3375    67
4         Audi  37.7   3405    70
5          BMW  30.0   3640    69
6        Buick  15.7   2880    69


> # sort by Width
> d3 = d2[order(d2$Width),]
> head(d3)
   Manufacturer Price Weight Width
80       Subaru   8.4   2045    60
31         Ford   7.4   1845    63
39          Geo   8.4   1695    63
44      Hyundai   8.0   2345    63
83       Suzuki   8.6   1965    63
88   Volkswagen   9.1   2240    63
```

Sorting – one column

```
> d2 = subset(d1,select=c(Manufacturer,Price,Weight,Width))
> head(d2)
  Manufacturer Price Weight Width
1        Acura  15.9   2705    68
2        Acura  33.9   3560    71
3         Audi  29.1   3375    67
4         Audi  37.7   3405    70
5          BMW  30.0   3640    69
6        Buick  15.7   2880    69
```

```
> # sort by Width
> d3 = d2[order(d2$Width),]
> head(d3)
   Manufacturer Price Weight Width
80       Subaru   8.4   2045    60
31         Ford   7.4   1845    63
39          Geo   8.4   1695    63
44      Hyundai   8.0   2345    63
83       Suzuki   8.6   1965    63
88   Volkswagen   9.1   2240    63
```

```
> # sort DECREASING by Width
> d3 = d2[order(d2$Width,decreasing = T),]
> head(d3)
   Manufacturer Price Weight Width
8         Buick  23.7   4105    78
17    Chevrolet  16.6   4025    78
38         Ford  20.9   3950    78
18    Chevrolet  18.8   3910    77
52      Lincoln  36.1   4055    77
75      Pontiac  17.7   3240    75
```

Example 2 – Auto dataset

Example 2 – Auto dataset

A dataset of 392 cars with 9 numeric variables

```
library(ISLR)
d1=Auto
#
str(d1)
```

```
## 'data.frame':    392 obs. of  9 variables:
##  $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
##  $ cylinders   : num  8 8 8 8 8 8 8 8 8 8 ...
##  $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
##  $ horsepower  : num  130 165 150 150 140 198 220 215 225 190 ...
##  $ weight      : num  3504 3693 3436 3433 3449 ...
##  $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
##  $ year        : num  70 70 70 70 70 70 70 70 70 70 ...
##  $ origin      : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ name        : Factor w/ 304 levels "amc ambassador brougham",..: 49 36
```

```
nrow(d1)
```

```
## [1] 392
```

```
n = nrow(d1)
#
# remove factor last col
#
d2 = d1[,-9]
```

?Auto

Example 2 – Auto dataset

Format

A data frame with 392 observations on the following 9 variables.

`mpg`
: miles per gallon

`cylinders`
: Number of cylinders between 4 and 8

`displacement`
: Engine displacement (cu. inches)

`horsepower`
: Engine horsepower

`weight`
: Vehicle weight (lbs.)

`acceleration`
: Time to accelerate from 0 to 60 mph (sec.)

`year`
: Model year (modulo 100)

`origin`
: Origin of car (1. American, 2. European, 3. Japanese)

`name`
: Vehicle name

Example 2 – Auto dataset

Analyze these variables by performing the following

- Show a summary of all columns
- Describe mpg

  (1) by n. of cylinders (2) overall (3) by Origin
- Investigate the relation of horsepower and weight

  (1) by Origin and (2) overall
- Investigate the relation among all variables with

  (1) scatterplots (2) correlation values
- Show the distribution of mpg over the years

Example 2 – Auto dataset

Analyze these variables by performing the following

- Show a summary of all columns

```
> # remove last col
> #
> d2 = d1[,-9]
```

```
> summary(d2)
      mpg          cylinders      displacement     horsepower
 Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0
 1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0
 Median :22.75   Median :4.000   Median :151.0   Median : 93.5
 Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5
 3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0
 Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0

     weight      acceleration        year          origin
 Min.   :1613   Min.   : 8.00   Min.   :70.00   Min.   :1.000
 1st Qu.:2225   1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000
 Median :2804   Median :15.50   Median :76.00   Median :1.000
 Mean   :2978   Mean   :15.54   Mean   :75.98   Mean   :1.577
 3rd Qu.:3615   3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000
 Max.   :5140   Max.   :24.80   Max.   :82.00   Max.   :3.000
```

Example 2 – Auto dataset

Analyze these variables by performing the following

- Show a summary of all columns

```
> # remove last col
> #
> d2 = d1[,-9]
```

```
> summary(d2)
```

categorical

mpg	cylinders	displacement	horsepower
Min. : 9.00	Min. :3.000	Min. : 68.0	Min. : 46.0
1st Qu.:17.00	1st Qu.:4.000	1st Qu.:105.0	1st Qu.: 75.0
Median :22.75	Median :4.000	Median :151.0	Median : 93.5
Mean :23.45	Mean :5.472	Mean :194.4	Mean :104.5
3rd Qu.:29.00	3rd Qu.:8.000	3rd Qu.:275.8	3rd Qu.:126.0
Max. :46.60	Max. :8.000	Max. :455.0	Max. :230.0

categorical

weight	acceleration	year	origin
Min. :1613	Min. : 8.00	Min. :70.00	Min. :1.000
1st Qu.:2225	1st Qu.:13.78	1st Qu.:73.00	1st Qu.:1.000
Median :2804	Median :15.50	Median :76.00	Median :1.000
Mean :2978	Mean :15.54	Mean :75.98	Mean :1.577
3rd Qu.:3615	3rd Qu.:17.02	3rd Qu.:79.00	3rd Qu.:2.000
Max. :5140	Max. :24.80	Max. :82.00	Max. :3.000

Example 2 – Auto dataset

Convert some variables to categorical (factors)

```
> d3 = d2
> d3$cylinders = as.factor(d3$cylinders)
> d3$year = as.factor(d3$year)
> d3$origin = as.factor(d3$origin)
```

Example 2 – Auto dataset

Convert some variables to categorical (factors)

- Show a summary of all columns

```
> d3 = d2
> d3$cylinders = as.factor(d3$cylinders)
> d3$year = as.factor(d3$year)
> d3$origin = as.factor(d3$origin)
```

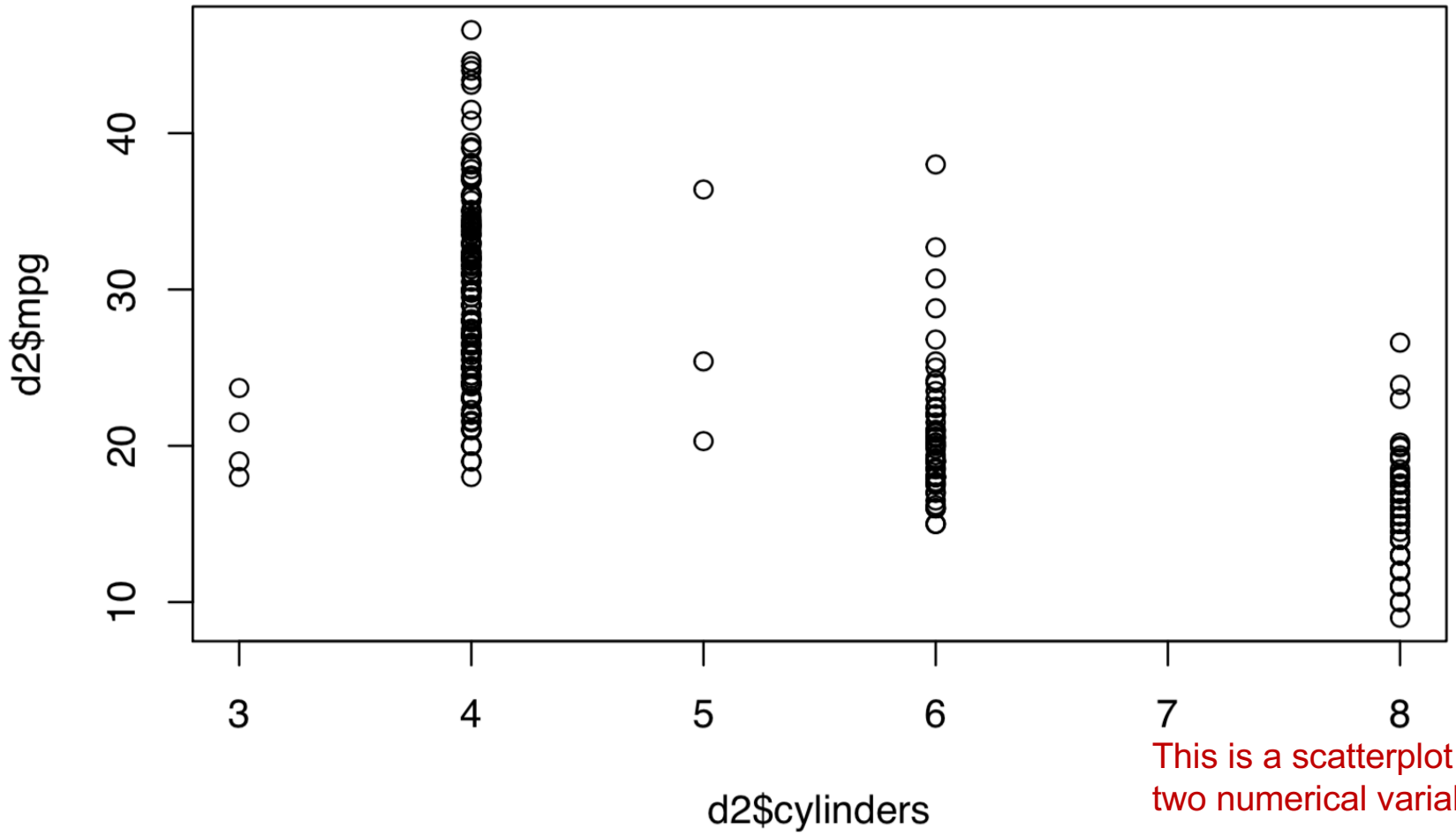
```
> summary(d3)
      mpg         cylinders  displacement     horsepower
 Min.   : 9.00   3:  4     Min.   : 68.0   Min.   : 46.0
 1st Qu.:17.00   4:199     1st Qu.:105.0   1st Qu.: 75.0
 Median :22.75   5:  3     Median :151.0   Median : 93.5
 Mean   :23.45   6: 83     Mean   :194.4   Mean   :104.5
 3rd Qu.:29.00   8:103     3rd Qu.:275.8   3rd Qu.:126.0
 Max.   :46.60             Max.   :455.0   Max.   :230.0

     weight      acceleration        year     origin
 Min.   :1613   Min.   : 8.00   73     : 40   1:245
 1st Qu.:2225   1st Qu.:13.78   78     : 36   2: 68
 Median :2804   Median :15.50   76     : 34   3: 79
 Mean   :2978   Mean   :15.54   75     : 30
 3rd Qu.:3615   3rd Qu.:17.02   82     : 30
 Max.   :5140   Max.   :24.80   70     : 29
                                (Other):193
 .
```

Example 2 – Auto dataset

- Describe mpg (1) by n. of cylinders (2) overall (3) by Origin

```
plot(d2$cylinders, d2$mpg)
```

here cylinders is numeric



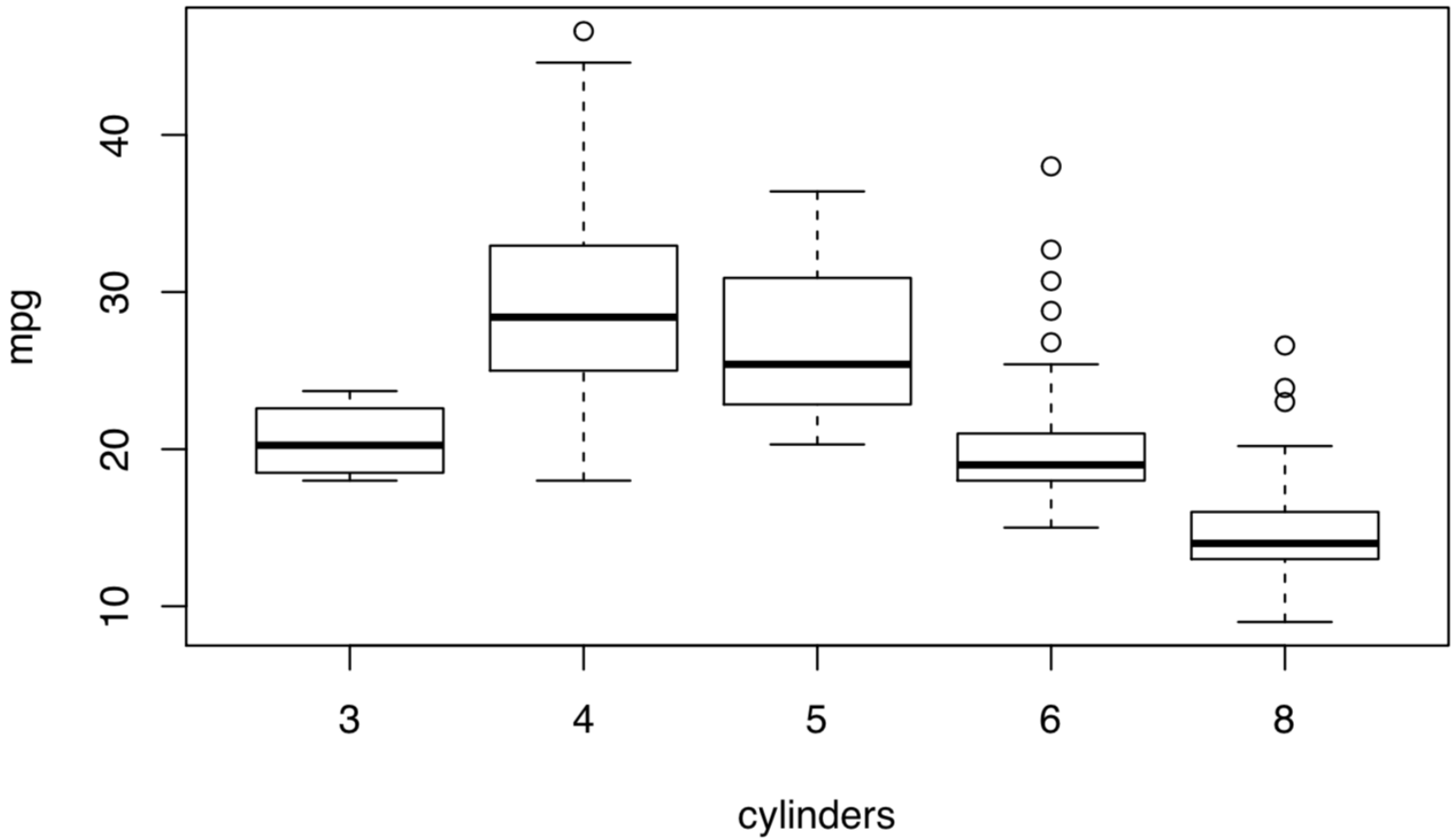
This is a scatterplot of two numerical variables

Example 2 – Auto dataset

```
# get boxplot when cylinders is factor
#
d2$cylinders=factor(d2$cylinders)
```

y ~ x, dataframe

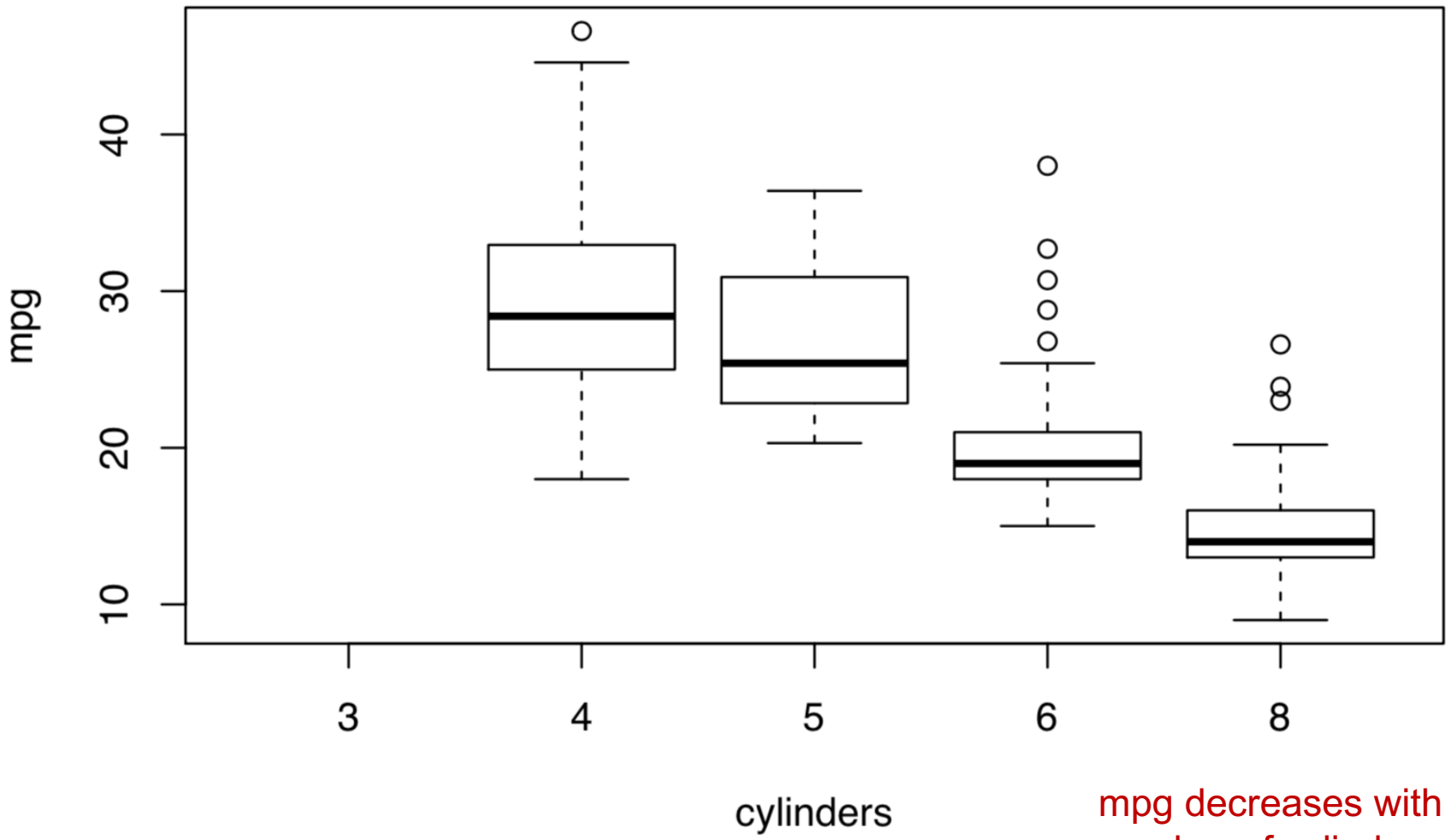
```
plot(mpg~cylinders,d3)
```



Example 2 – Auto dataset

```
# get boxplot when cylinders is factor
#
d2$cylinders=factor(d2$cylinders)
```

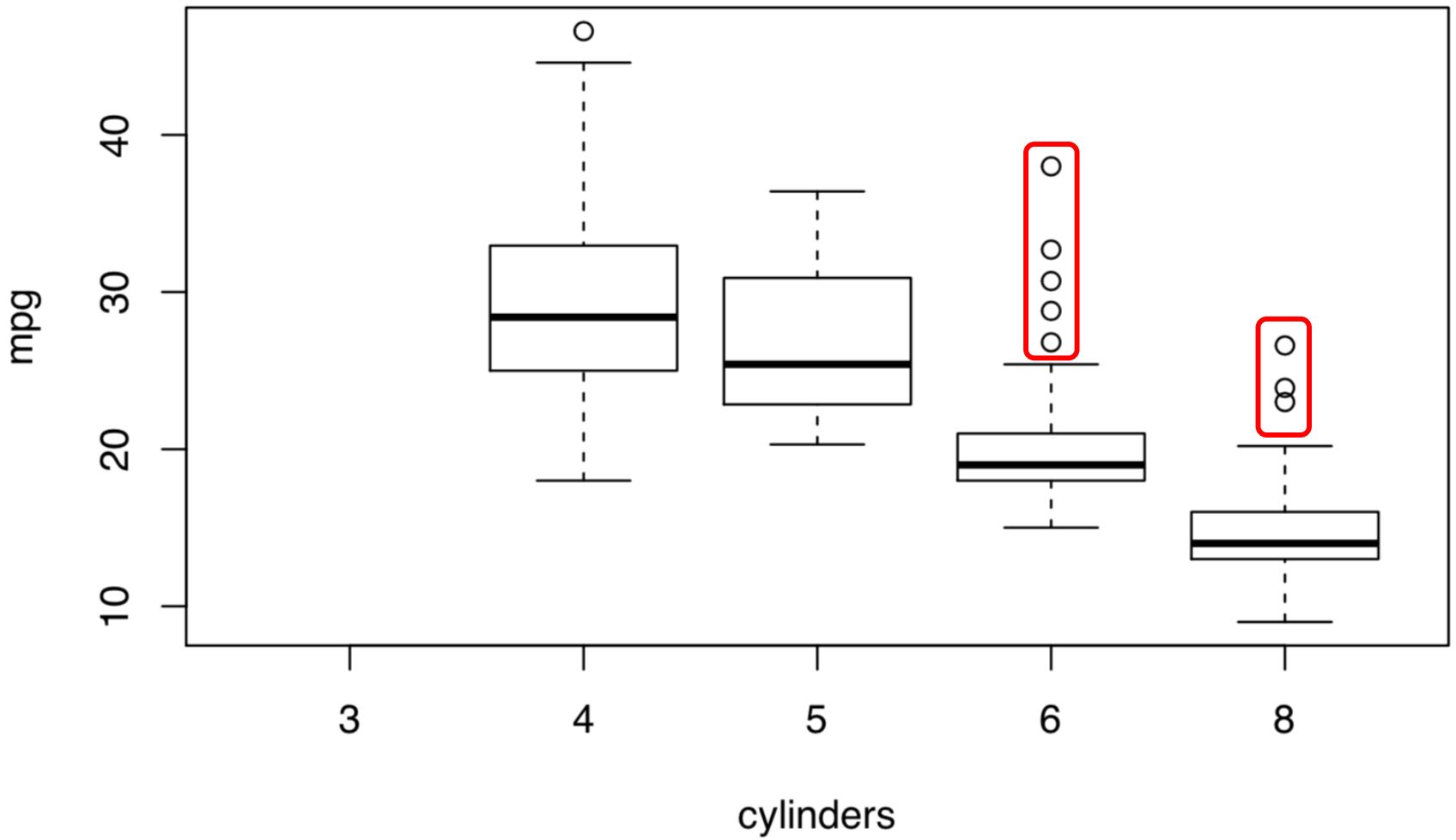
```
plot(mpg~cylinders,d3)
```



mpg decreases with number of cylinders

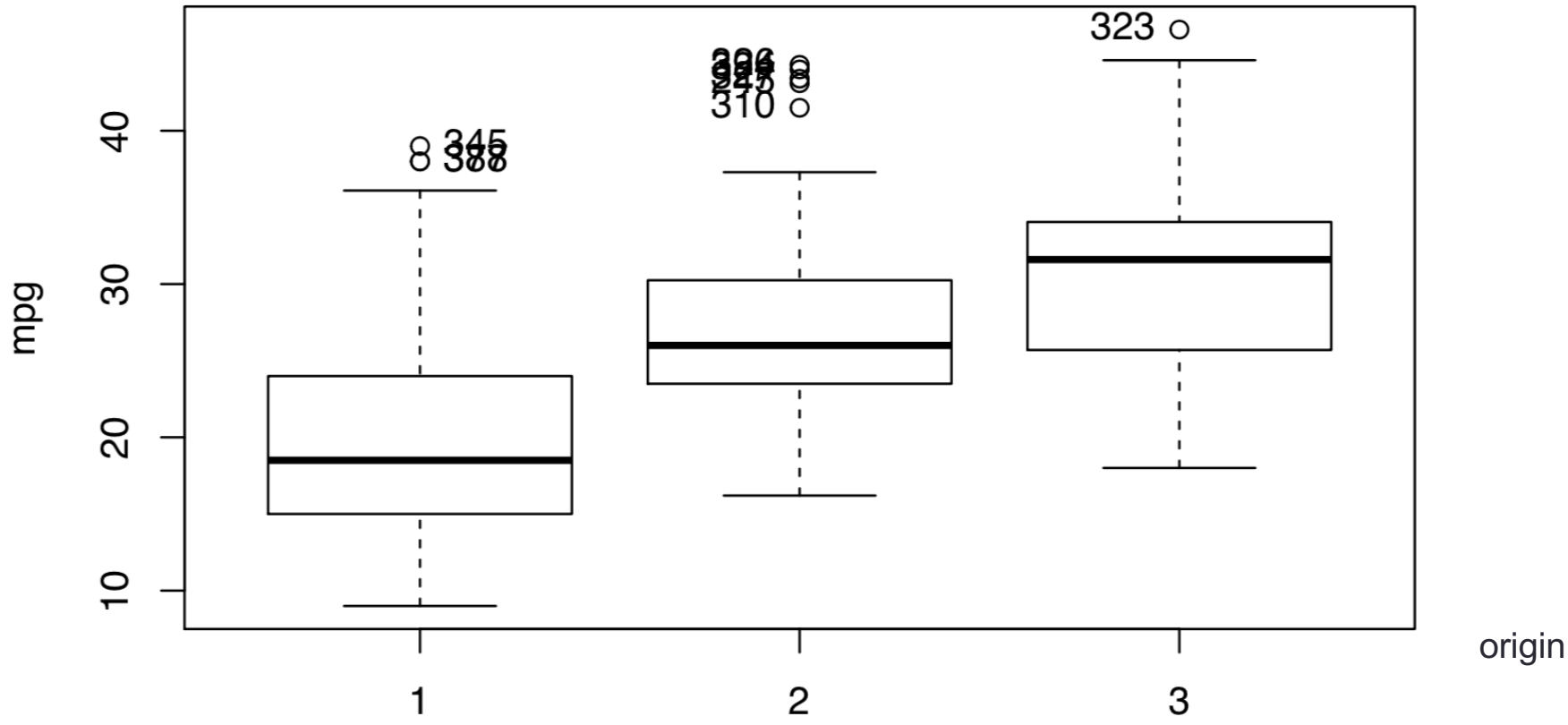
Example 2 – Auto dataset - outliers

```
plot(mpg~cylinders,d3)
```



Example 2 – Auto dataset

```
plot(mpg~origin,d3)          # same as
boxplot(mpg~origin,d3)
#
# to list outliers
#
Boxplot(mpg~origin,d3)       # library(car)
```




```
## [1] "345" "378" "387" "245" "310" "326" "327" "394" "323"
```

Auto dataset – list of outliers

```
plot(mpg~origin,d3)          # same as
boxplot(mpg~origin,d3)
#
# to list outliers
#
Boxplot(mpg~origin,d3)       # library(car)
```

```
## [1] "345" "378" "387" "245" "310" "326" "327" "394" "323"
```

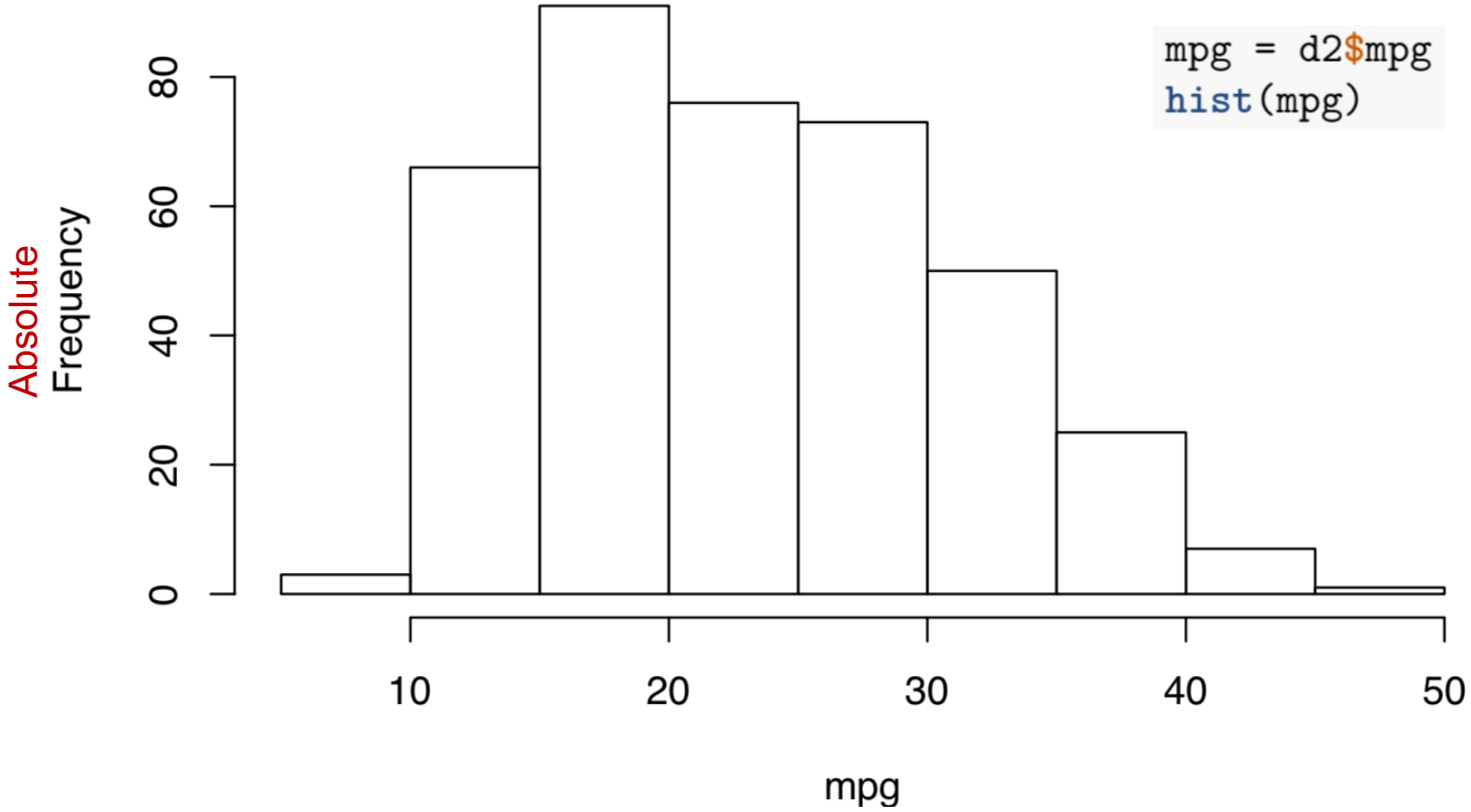
← row numbers are saved in object "a" as shown here ←

```
a=Boxplot(mpg~origin,d3)
d3[a,]
```

```
##      mpg cylinders displacement horsepower weight acceleration year origin
## 345 39.0         4           86         64   1875         16.4   81      1
## 378 38.0         4          105         63   2125         14.7   82      1
## 387 38.0         6          262         85   3015         17.0   82      1
## 245 43.1         4           90         48   1985         21.5   78      2
## 310 41.5         4           98         76   2144         14.7   80      2
## 326 44.3         4           90         48   2085         21.7   80      2
## 327 43.4         4           90         48   2335         23.7   80      2
## 394 44.0         4           97         52   2130         24.6   82      2
## 323 46.6         4           86         65   2110         17.9   80      3
```

Example 2 – Absolute frequency histogram

- Describe mpg (1) by n. of cylinders (2) overall (3) by Origin

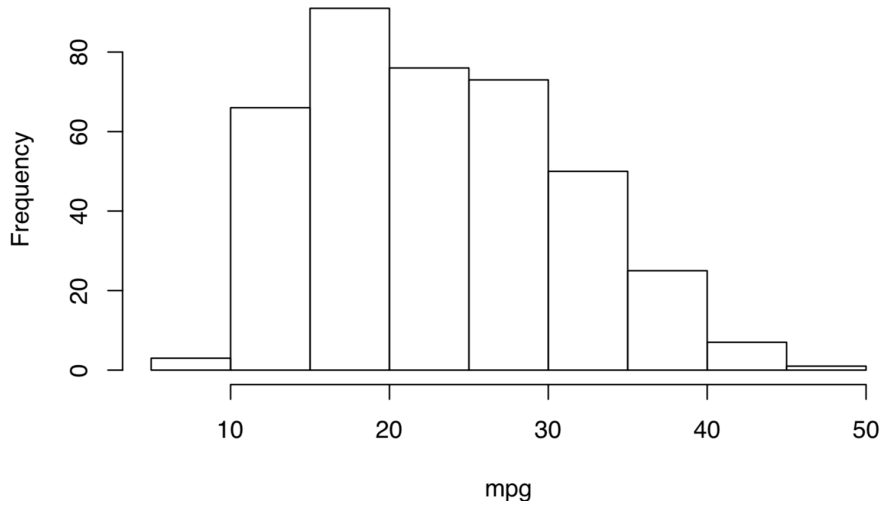


Histogram structure

```
h1 = hist(mpg)
str(h1)
```

```
## List of 6
##  $ breaks  : int [1:10] 5 10 15 20 25
##  $ counts  : int [1:9] 3 66 91 76 73
##  $ density : num [1:9] 0.00153 0.0336
##  $ mids    : num [1:9] 7.5 12.5 17.5
```

split interval (5,50) into 9 subintervals of width 5



Get the frequency table from a histogram

```
h1 = hist(mpg)
str(h1)
```

```
## List of 6
##  $ breaks  : int [1:10] 5 10 15 20 25
##  $ counts  : int [1:9] 3 66 91 76 73
##  $ density : num [1:9] 0.00153 0.0336
##  $ mids    : num [1:9] 7.5 12.5 17.5
```

```
breaks = h1$breaks
m = length(breaks)
rbounds = breaks[-1]
lbounds = breaks[-m]
```

m = 100

```
breaks
```

```
##  [1]  5 10 15 20 25 30 35 40 45 50
```

```
rbounds
```

```
## [1] 10 15 20 25 30 35 40 45 50
```

```
lbounds
```

```
## [1]  5 10 15 20 25 30 35 40 45
```

Histograms display the values from a frequency table

```
df4 = data.frame(from = lbounds,to=rbounds,
                 midpoint = h1$mids,
                 abs.frequency = h1$counts)
print(df4,row.names = F)
```

```
##  from to midpoint abs.frequency
##     5 10      7.5             3
##    10 15     12.5            66
##    15 20     17.5            91
##    20 25     22.5            76
##    25 30     27.5            73
##    30 35     32.5            50
##    35 40     37.5            25
##    40 45     42.5             7
##    45 50     47.5             1
```

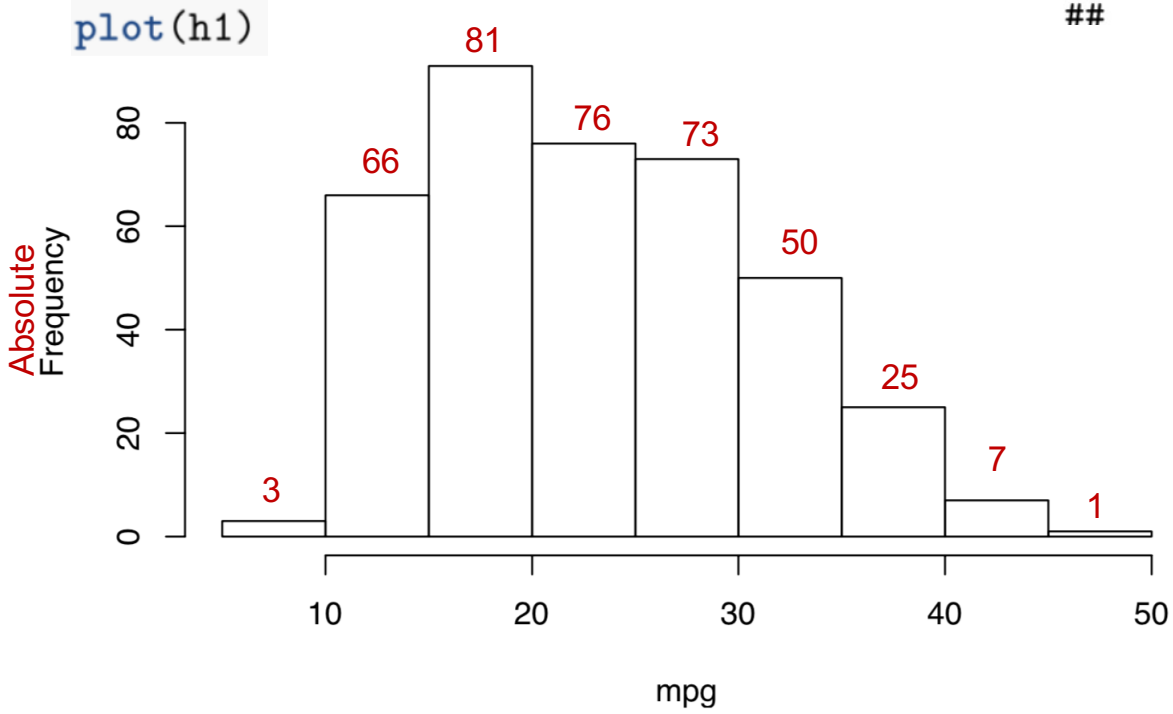
Histograms display the values from a frequency table

```
h1 = hist(mpg)
```

```
df4 = data.frame(from = lbounds,to=rbounds,
                 midpoint = h1$mids,
                 abs.frequency = h1$counts)
print(df4,row.names = F)
```

```
##  from to midpoint abs.frequency
##     5 10      7.5             3
##    10 15     12.5            66
##    15 20     17.5            91
##    20 25     22.5            76
##    25 30     27.5            73
##    30 35     32.5            50
##    35 40     37.5            25
##    40 45     42.5             7
##    45 50     47.5             1
```

```
plot(h1)
```

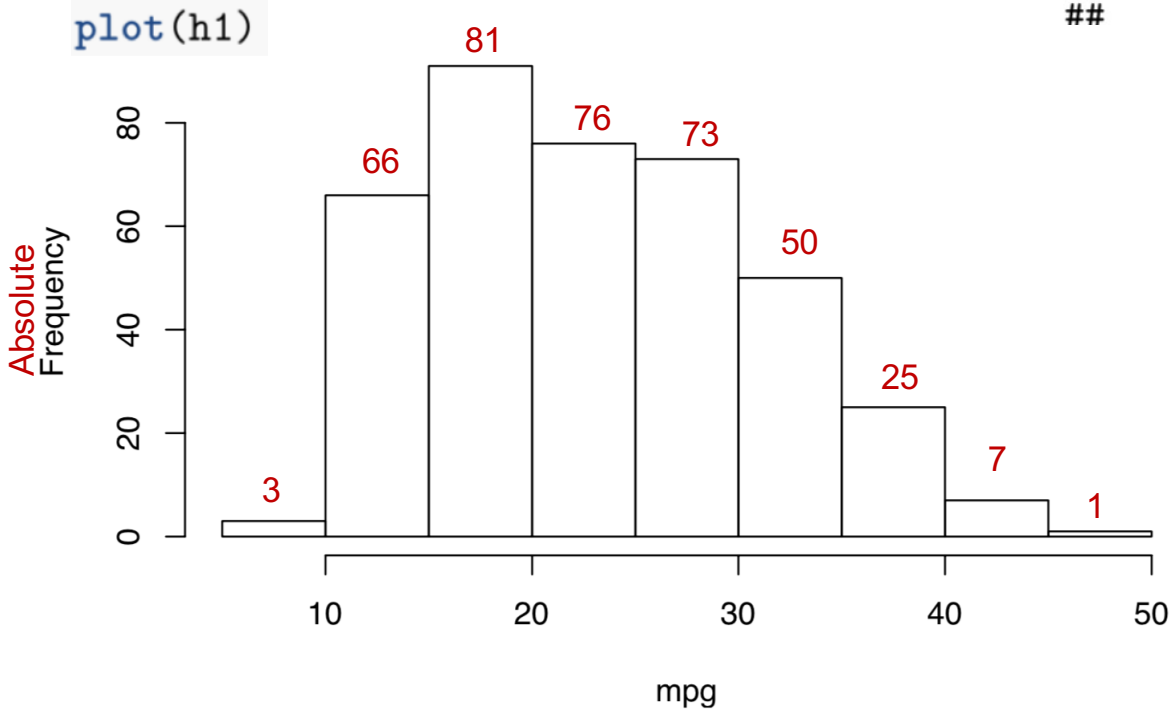


Histograms display the values from a frequency table

```
h1 = hist(mpg)
str(h1)
```

```
## List of 6
##  $ breaks  : int [1:10] 5 10 15 20 25
##  $ counts  : int [1:9] 3 66 91 76 73
##  $ density : num [1:9] 0.00153 0.0336
##  $ mids    : num [1:9] 7.5 12.5 17.5
```

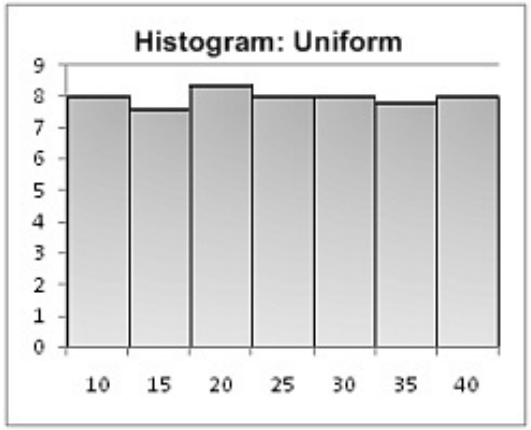
```
plot(h1)
```

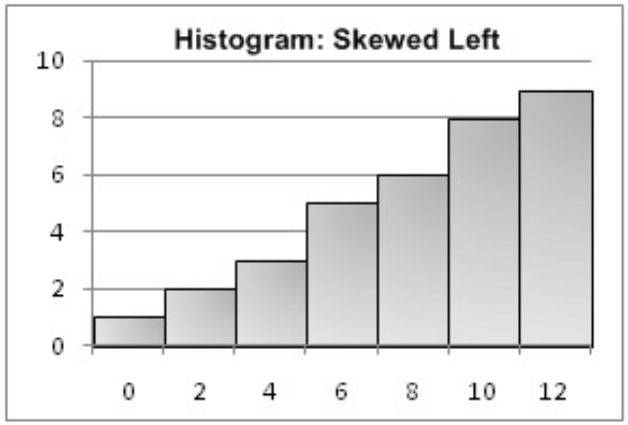


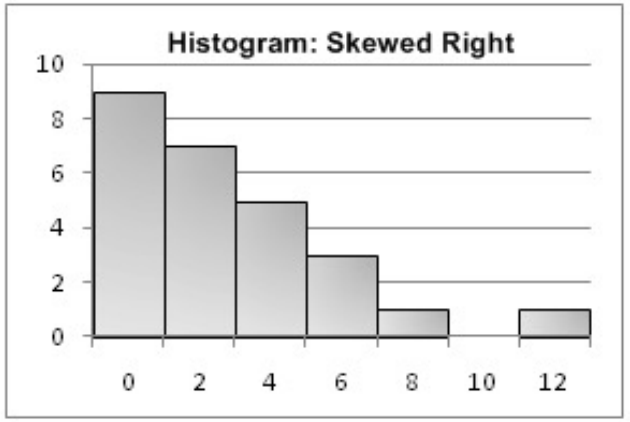

```
df4 = data.frame(from = lbounds,to=rbounds,
                 midpoint = h1$mids,
                 abs.frequency = h1$counts)
print(df4,row.names = F)
```

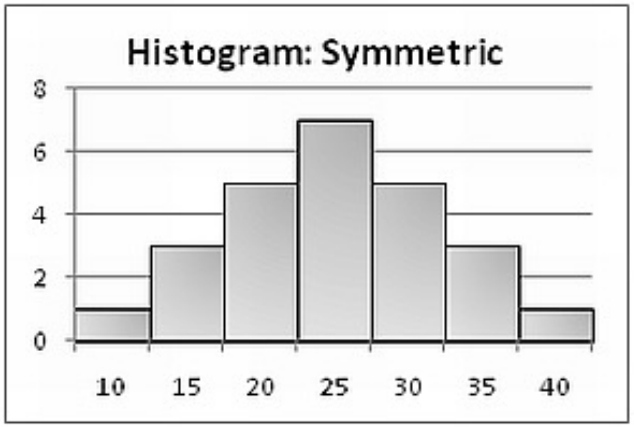
```
##  from to midpoint abs.frequency
##     5 10      7.5             3
##    10 15     12.5            66
##    15 20     17.5            91
##    20 25     22.5            76
##    25 30     27.5            73
##    30 35     32.5            50
##    35 40     37.5            25
##    40 45     42.5             7
##    45 50     47.5             1
```

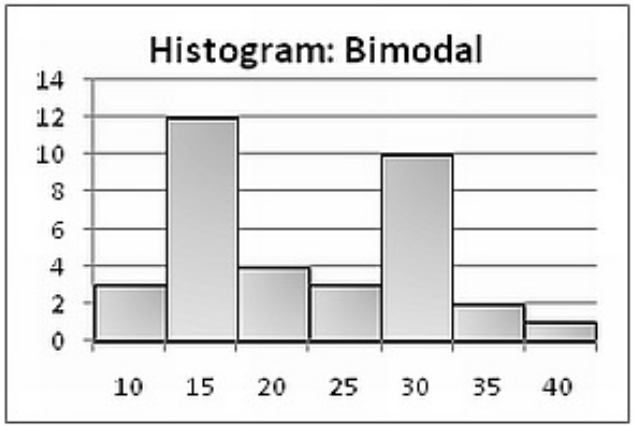
Histogram shapes









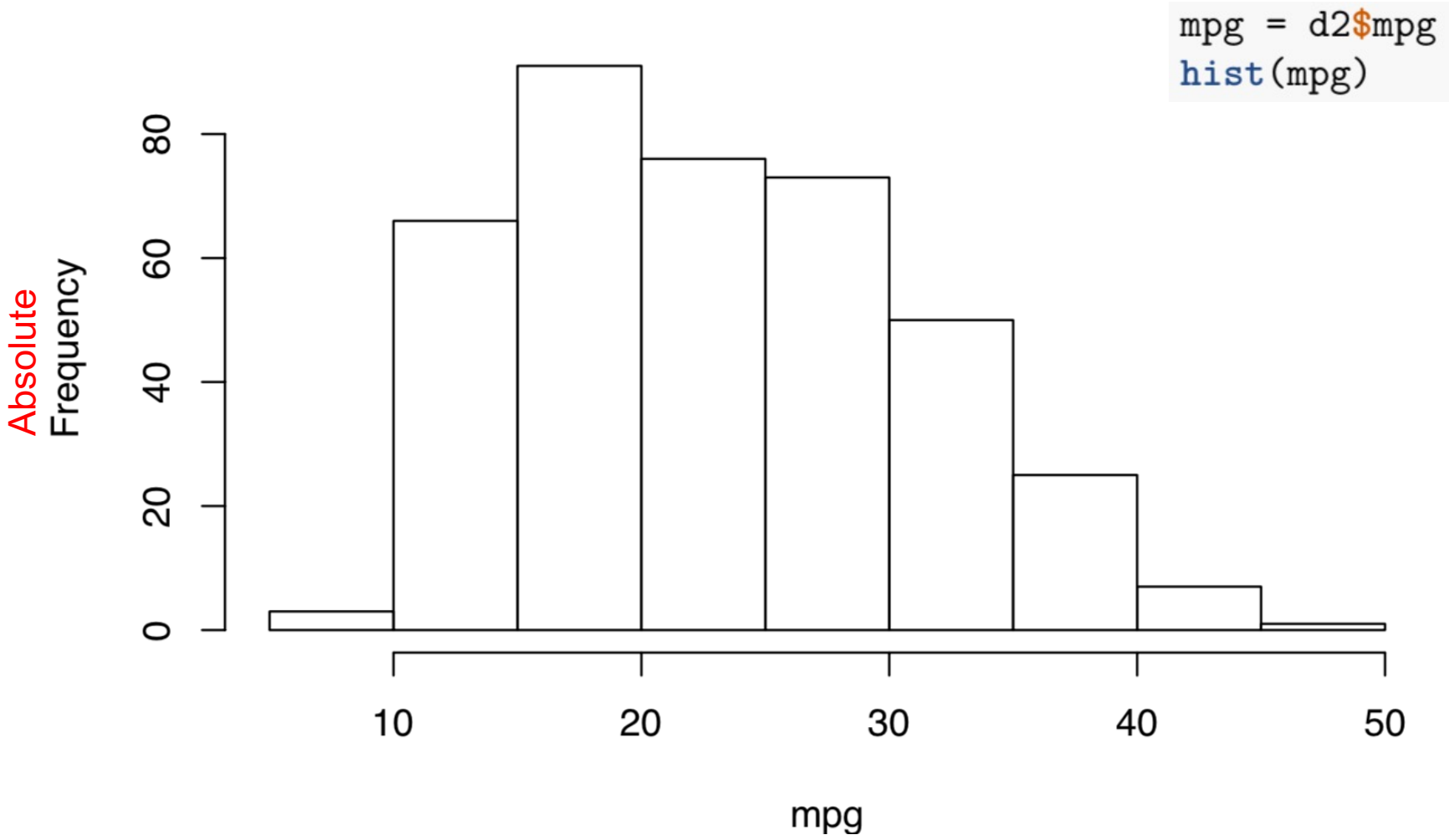


3 Types of Histograms

- Absolute frequency
- Relative frequency
- Density histogram

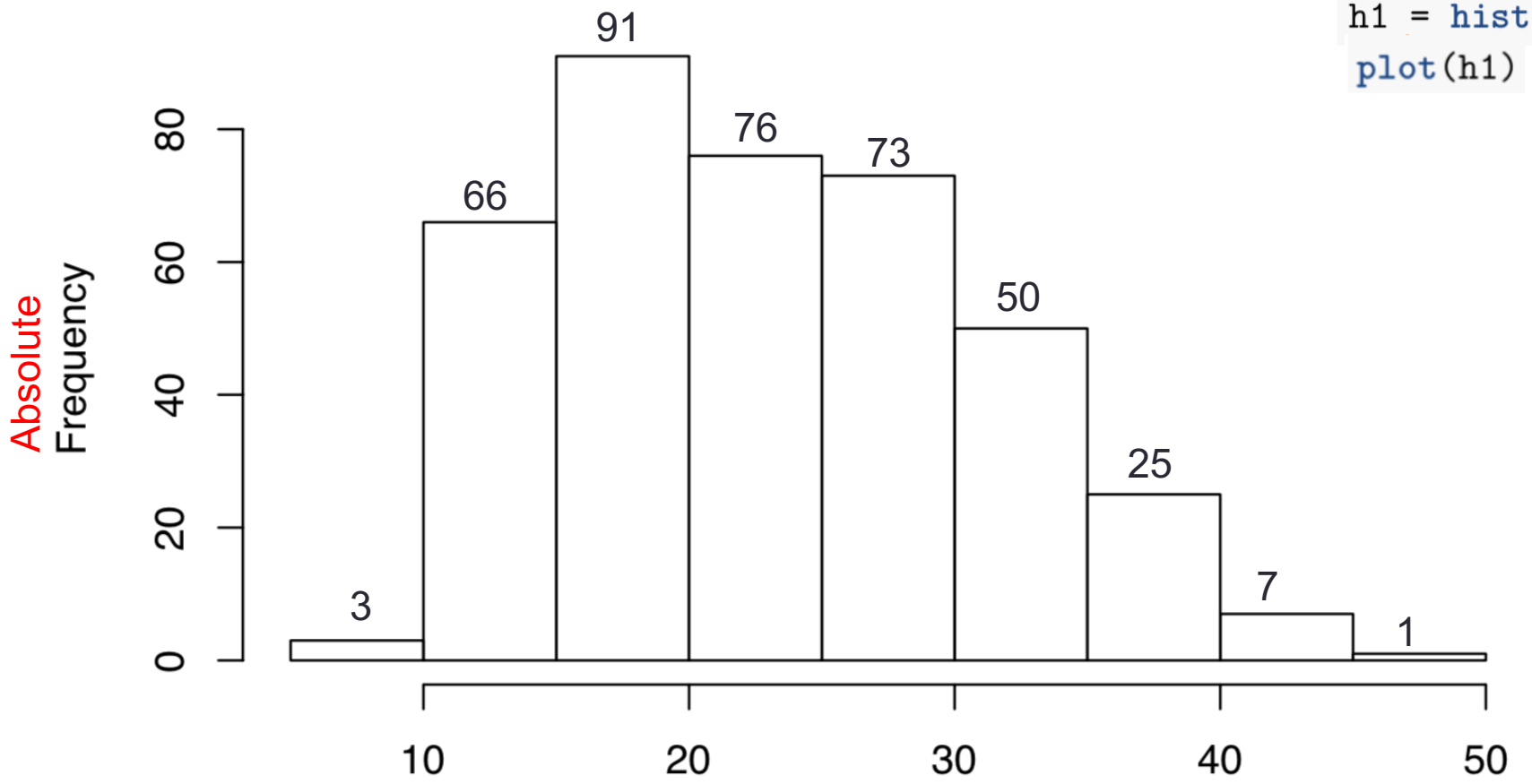
Example 2 – Absolute frequency histogram

```
mpg = d2$mpg
hist(mpg)
```



Example 2 – Absolute frequency histogram

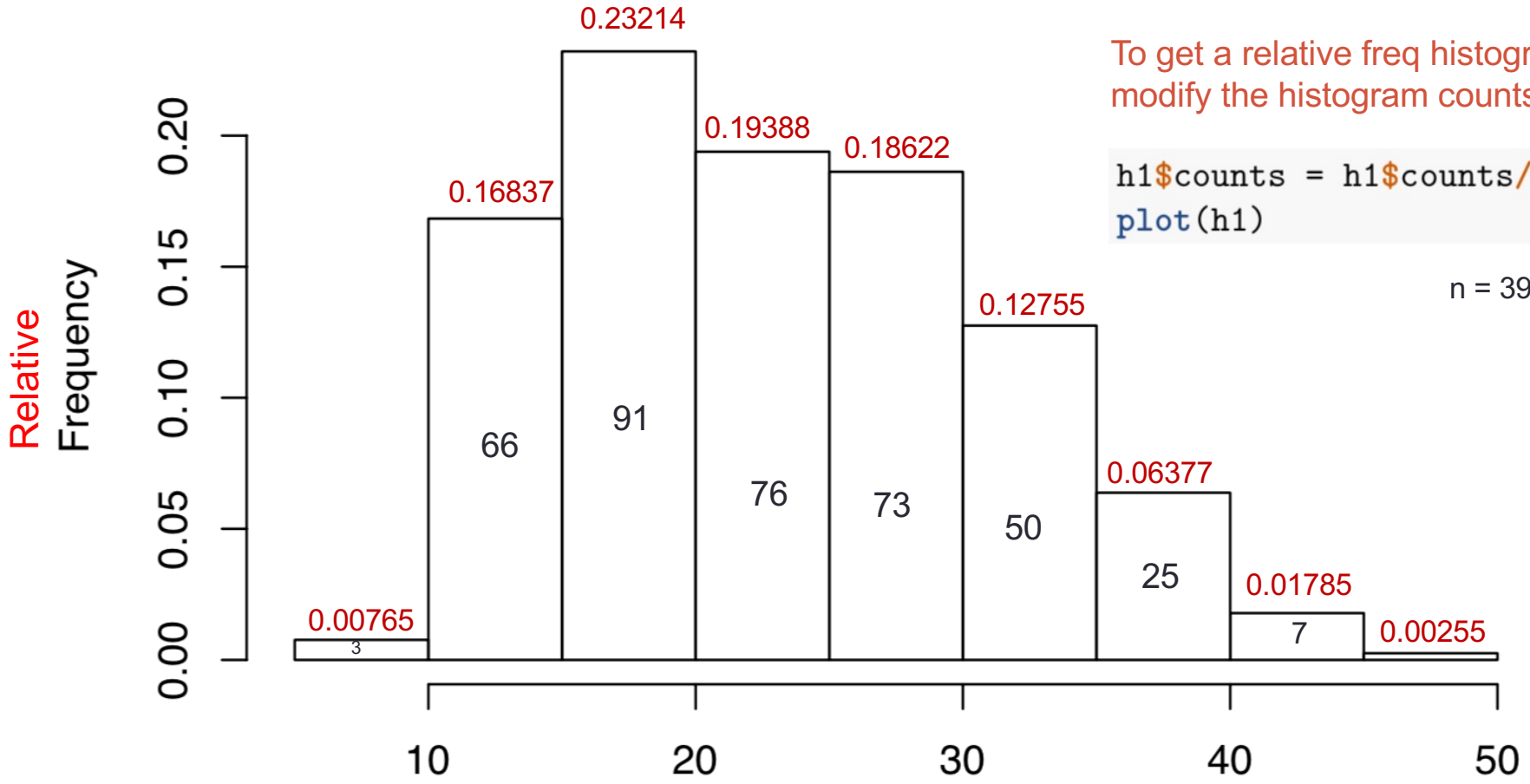
```
h1 = hist(mpg)
plot(h1)
```




```
h1$counts
```

```
## [1]  3 66 91 76 73 50 25  7  1
```

Example 2 – Relative frequency histogram



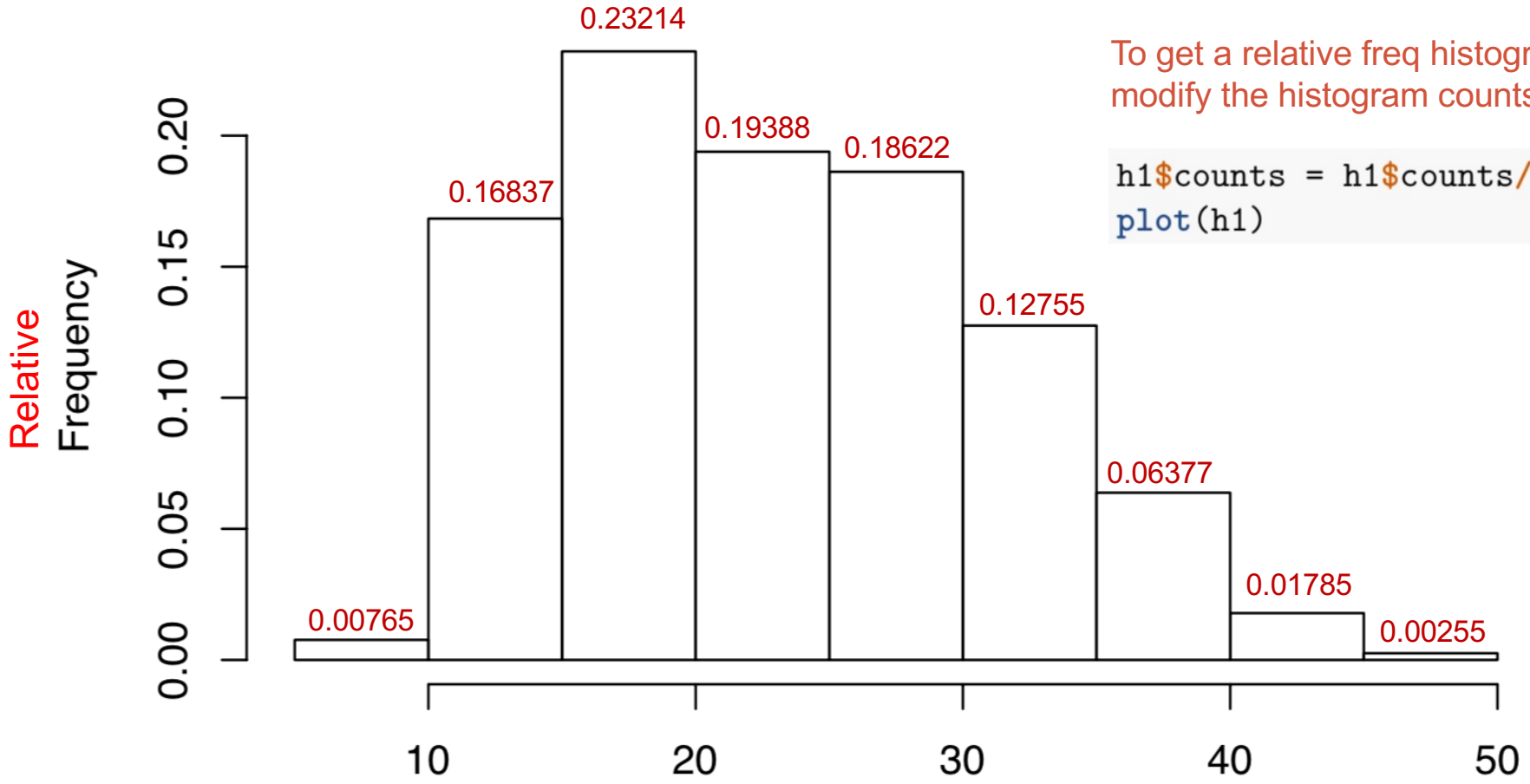
To get a relative freq histogram modify the histogram counts:

```
h1$counts = h1$counts/n
plot(h1)
```

n = 392

```
> h1$counts
[1] 0.007653061 0.168367347 0.232142857 0.193877551 0.186224490 0.127551020 0.063775510 0.017857143 0.00255102
```

Example 2 – Relative frequency histogram



To get a relative freq histogram modify the histogram counts:

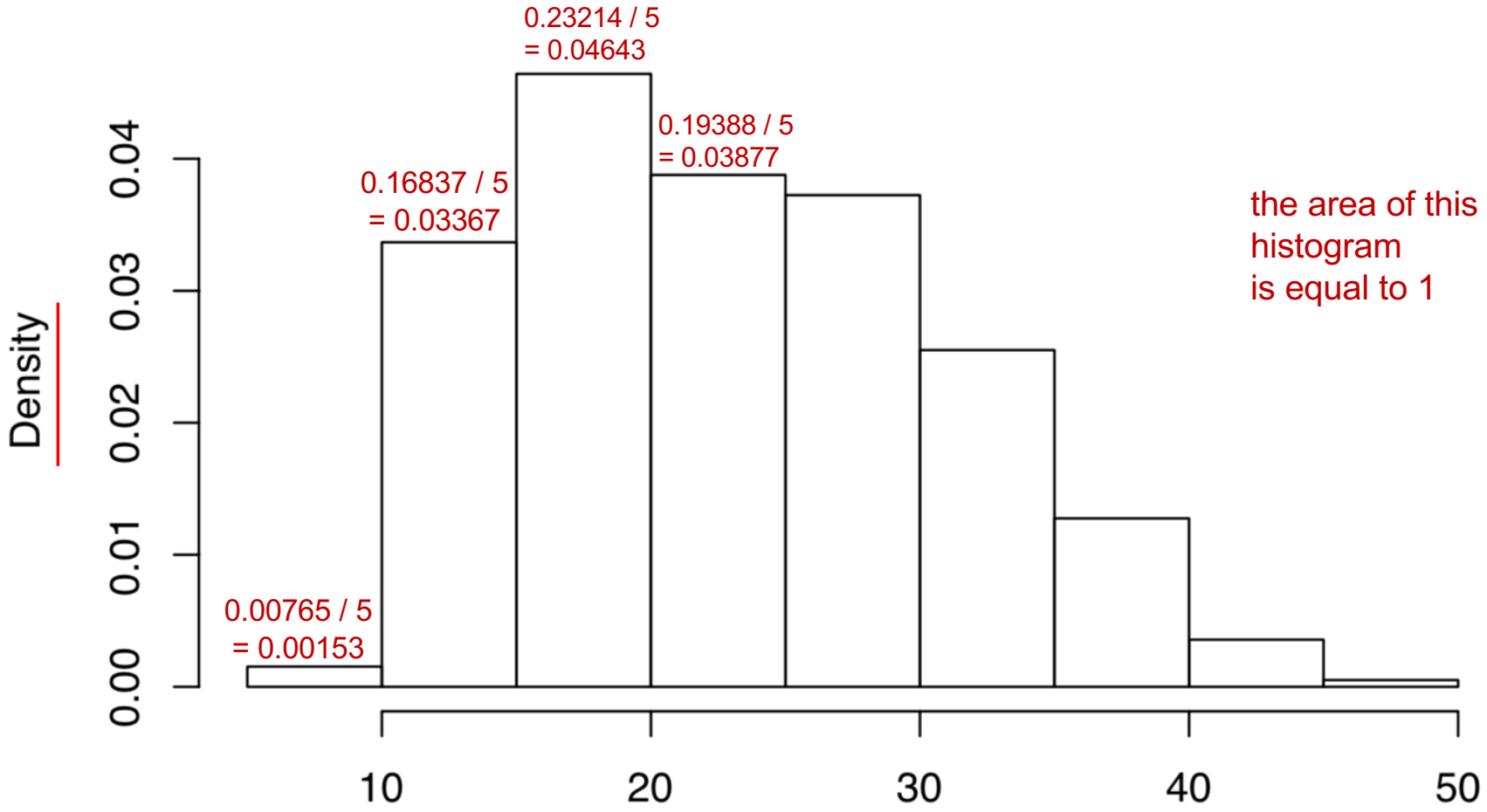
```
h1$counts = h1$counts/n
plot(h1)
```

```
> h1$counts
[1] 0.007653061 0.168367347 0.232142857 0.193877551 0.186224490 0.127551020 0.063775510 0.017857143 0.00255102
```

but the area of this histogram is equal to 5

Example 2 – Density histogram

```
hist(mpg,freq=F)
```



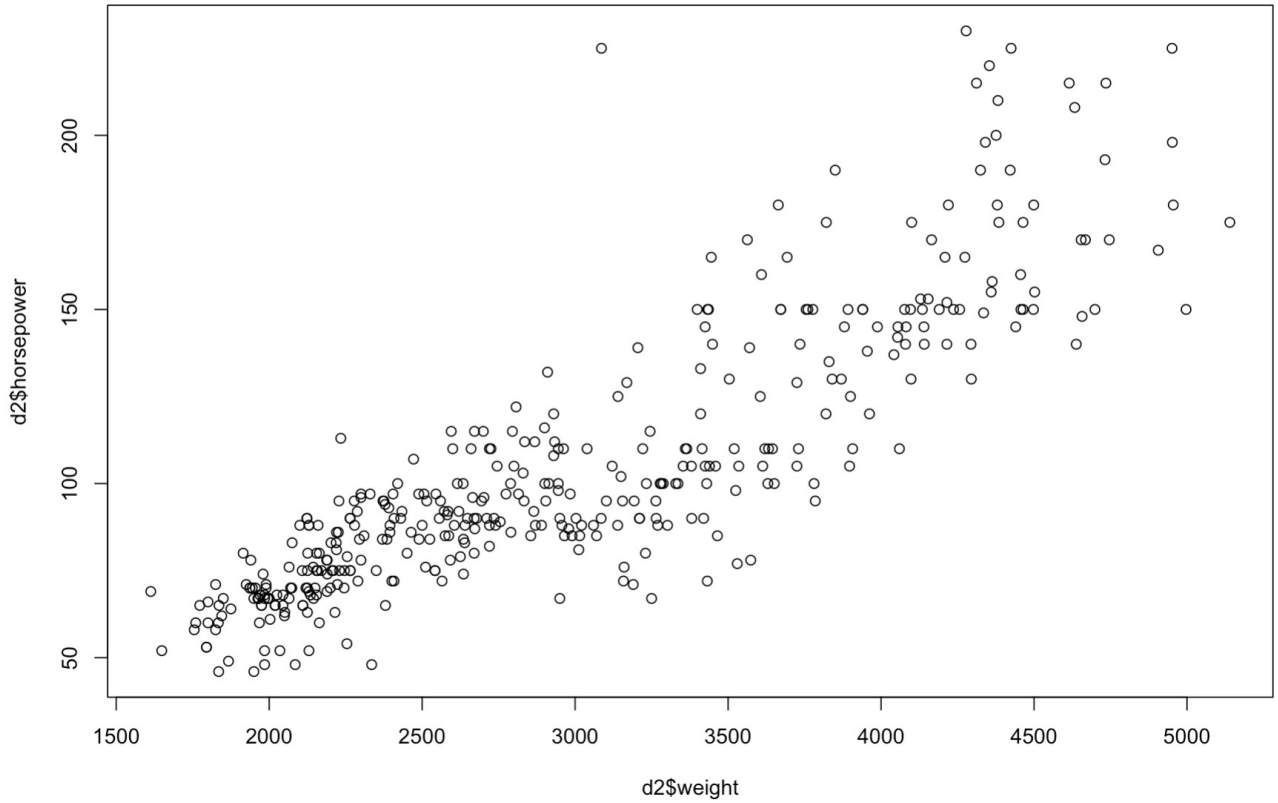
the area of this histogram is equal to 1

```
> h1$counts/5
[1] 0.0015306122 0.0336734694 0.0464285714 0.0387755102 0.0372448980 0.0255102041 0.0127551020 0.0035714286 0.0005102041
> h1$density
[1] 0.0015306122 0.0336734694 0.0464285714 0.0387755102 0.0372448980 0.0255102041 0.0127551020 0.0035714286 0.0005102041
```

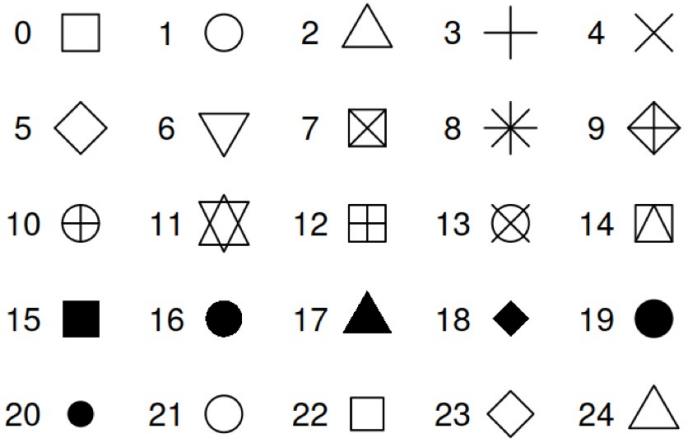
Example 2 – Auto dataset

The relation of horsepower and weight (1) overall (2) by Origin

```
plot(d2$weight,d2$horsepower)

plot(horsepower~weight,d2)
```



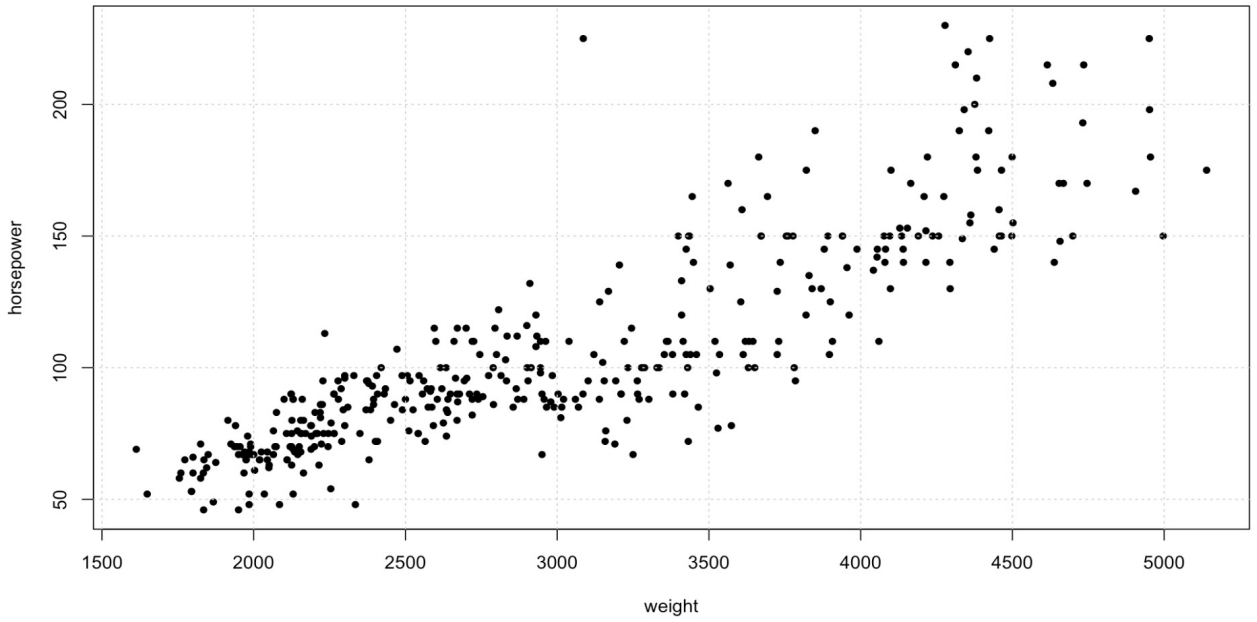
point characters



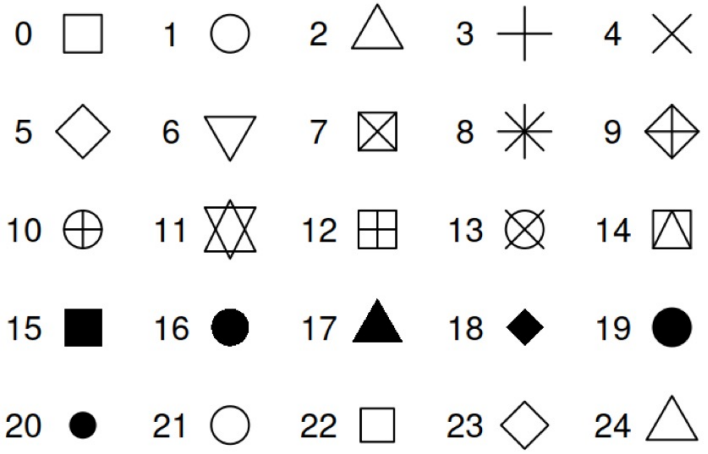
Example 2 – Auto dataset

The relation of horsepower and weight (1) overall

```
# change point character
plot(horsepower~weight,d2,pch=19,cex=0.75)
grid()
```



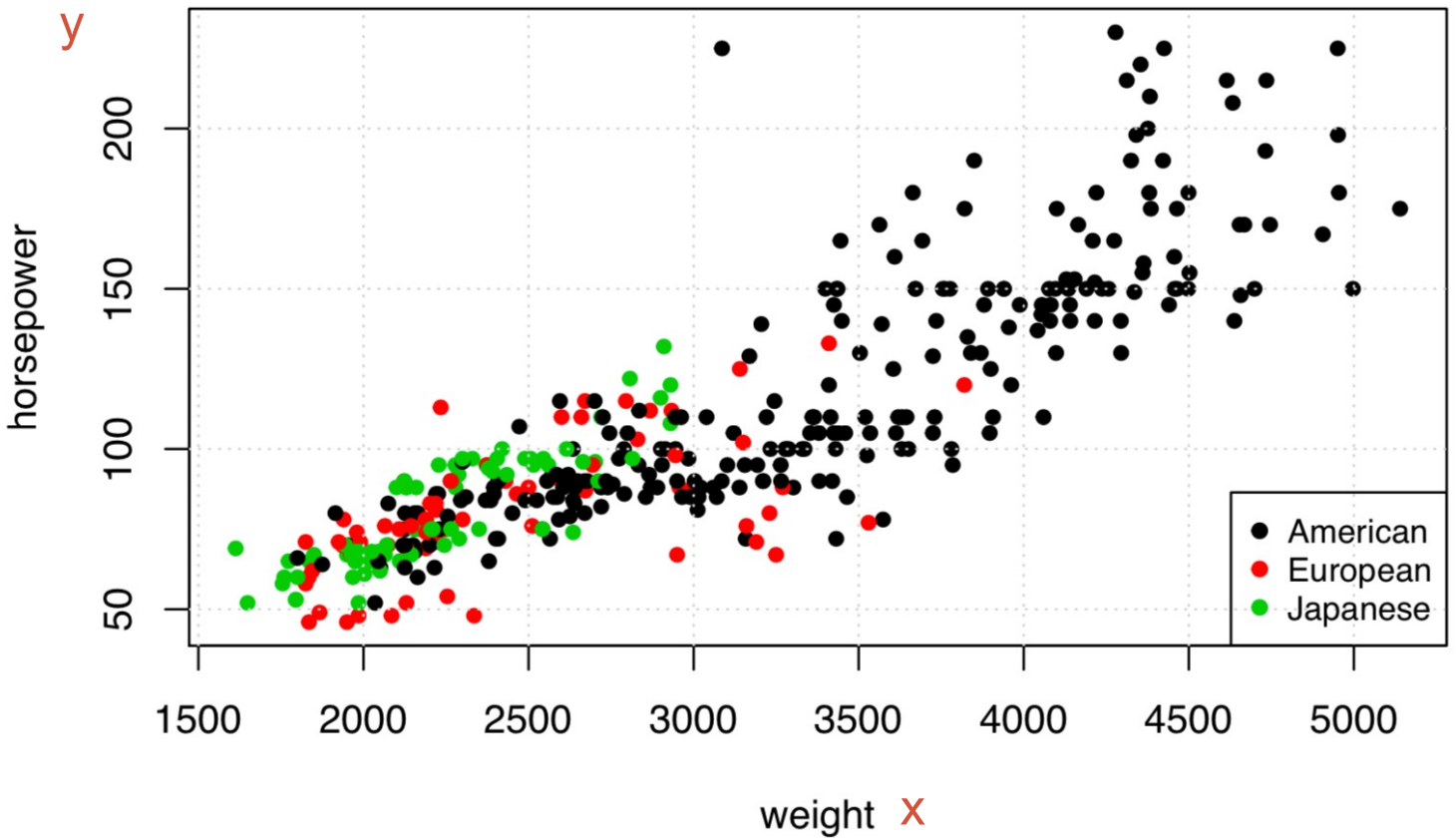
point characters



Auto dataset

```
plot(horsepower~weight,d2,pch=19,cex=0.75,col=origin)
grid()
```

The relation of horsepower and weight (2) by Origin



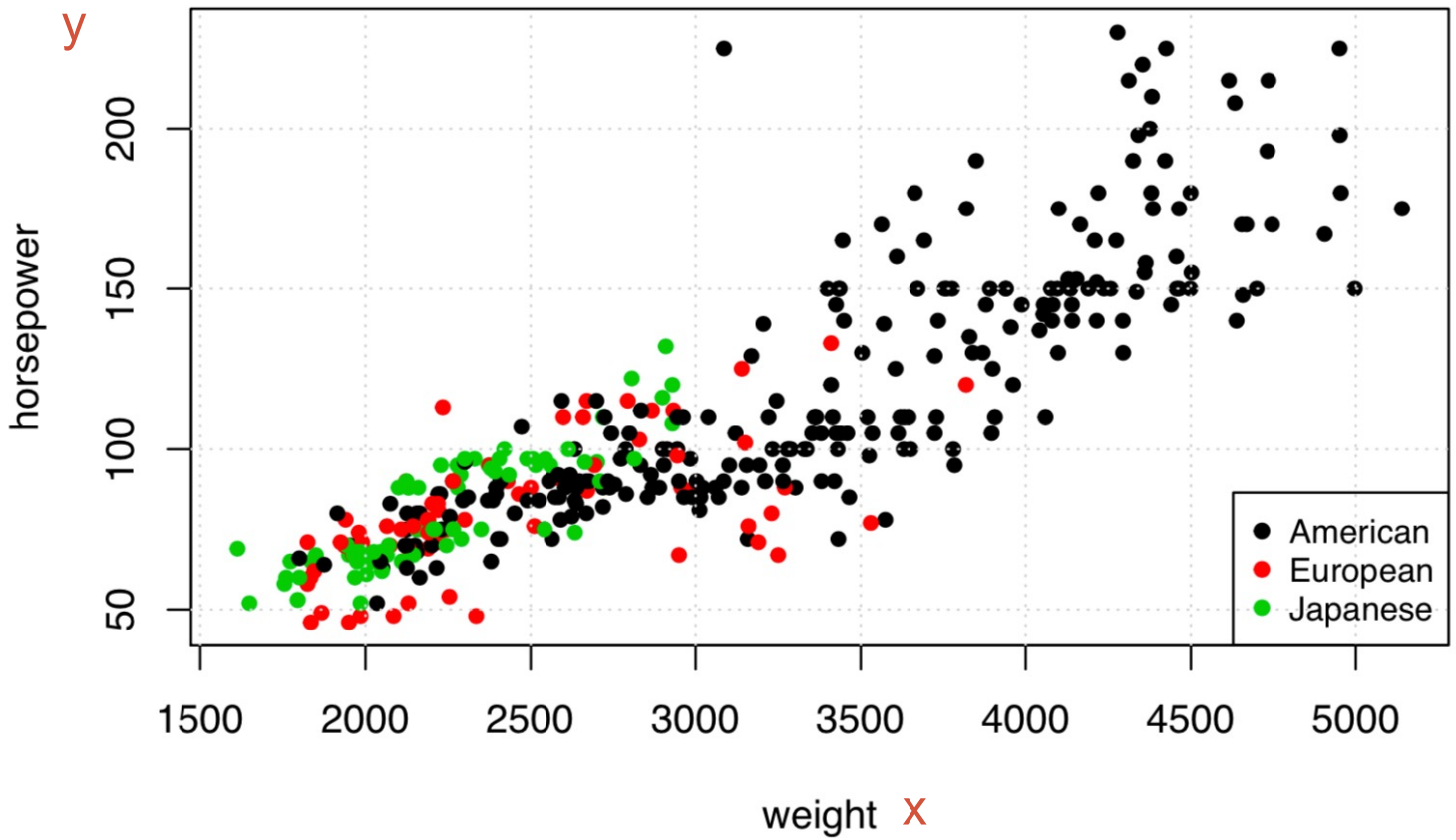

```
origin
1:245
2: 68
3: 79
```

Auto dataset

identify categories in column *origin*

```
plot(horsepower~weight,d2,pch=19,cex=0.75,col=origin)
grid()
```

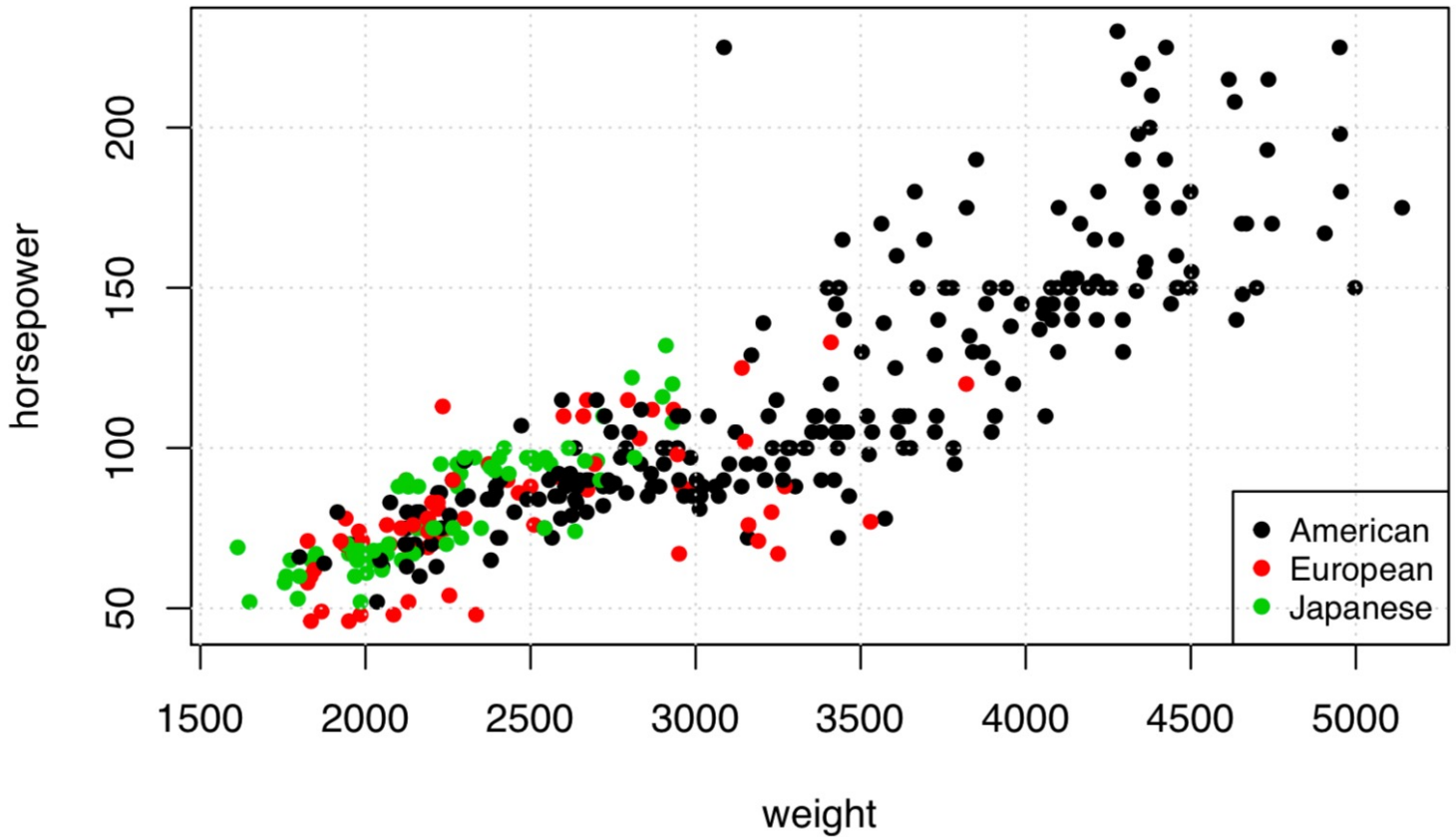
```
origin
1:245
2: 68
3: 79
```



Auto dataset

add a legend

```
plot(horsepower~weight,d2,pch=19,cex=0.75,col=origin)
grid()
#
# legend
#
label = c("American","European","Japanese")
color = c(1,2,3)
char = c(19,19,19)
legend("bottomright",label,pch=char,cex=0.8,col=color)
```



- **Japanese cars with the least weight**
- **European cars with less weight than Americans**

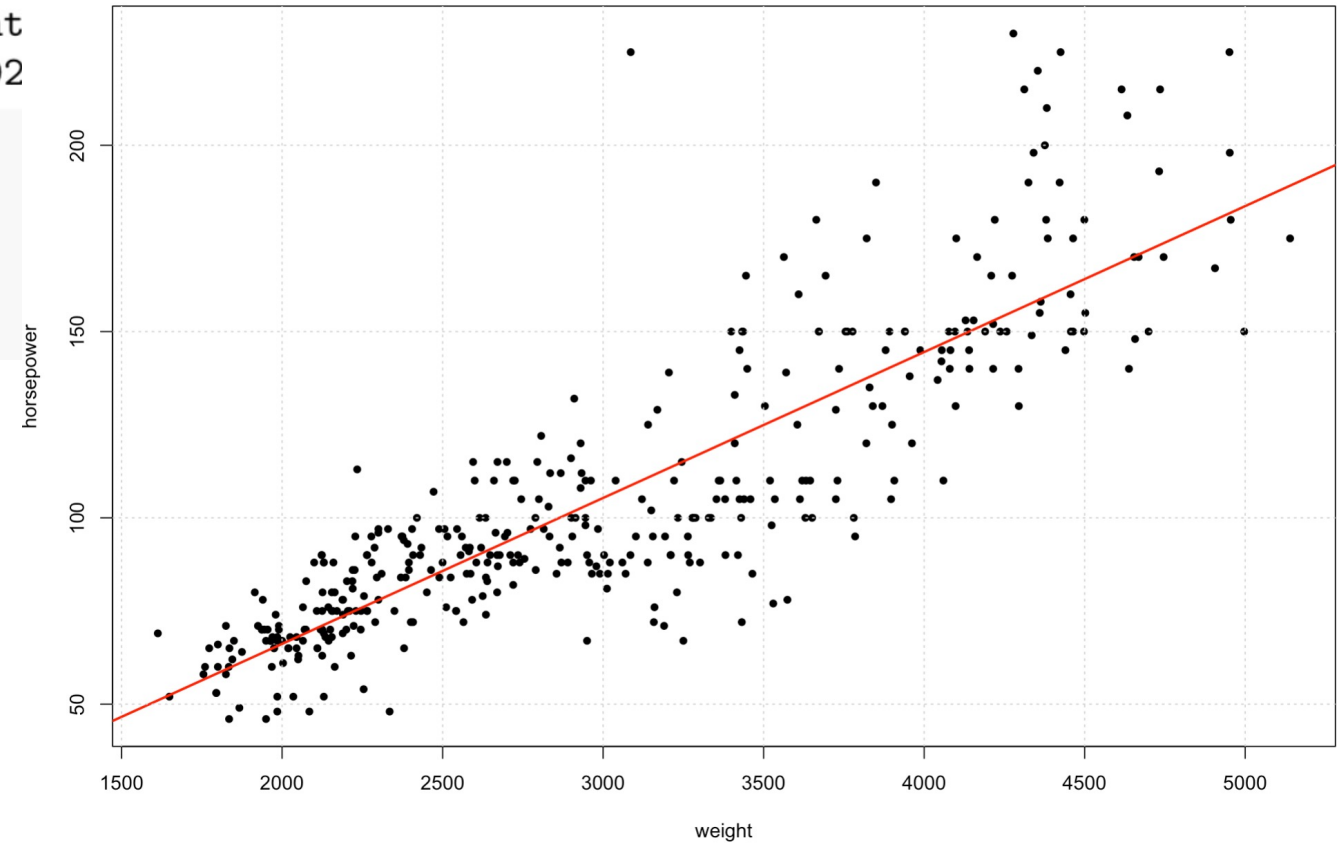
Example 2 – Auto dataset

```
plot(horsepower~weight,d2,pch=19,cex=0.75)
#
# linear model
#
m1=lm(horsepower~weight,d2)
coefficients(m1)
```

```
##   (Intercept)        weight
## -12.18348470   0.03917702
```

```
#
abline(m1)
abline(m1,col="red")
abline(m1,col="red",lwd=2)
grid()
```

add regression line to estimate the linear relation between horsepower and weight



Example 2 – Auto dataset

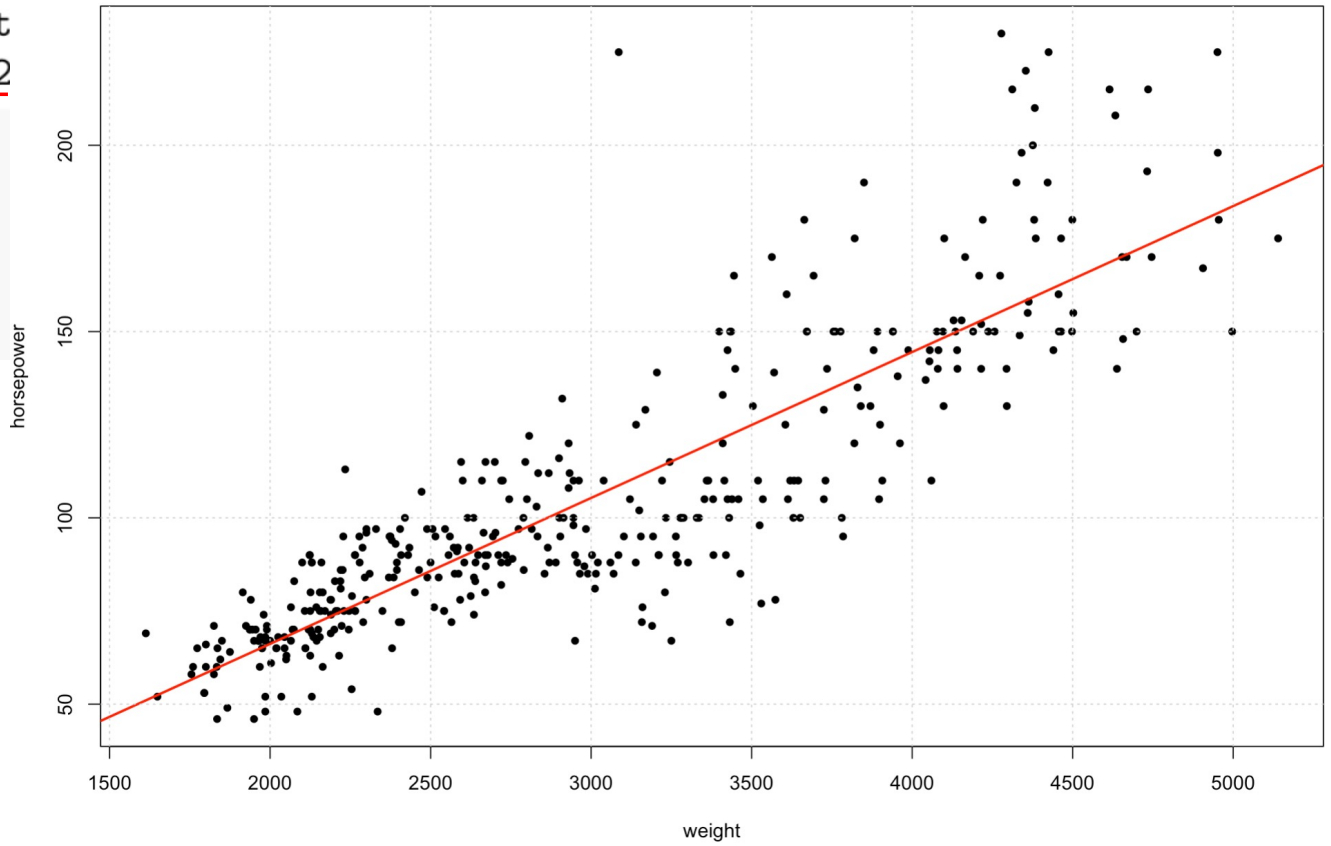
```
plot(horsepower~weight,d2,pch=19,cex=0.75)
#
# linear model
#
m1=lm(horsepower~weight,d2)
coefficients(m1)
```

```
##   (Intercept)        weight
## -12.18348470   0.03917702
```

```
#
abline(m1)
abline(m1,col="red")
abline(m1,col="red",lwd=2)
grid()
```

add regression line to estimate the linear relation between horsepower and weight

On average horsepower increases by 0.0391 for each additional pound



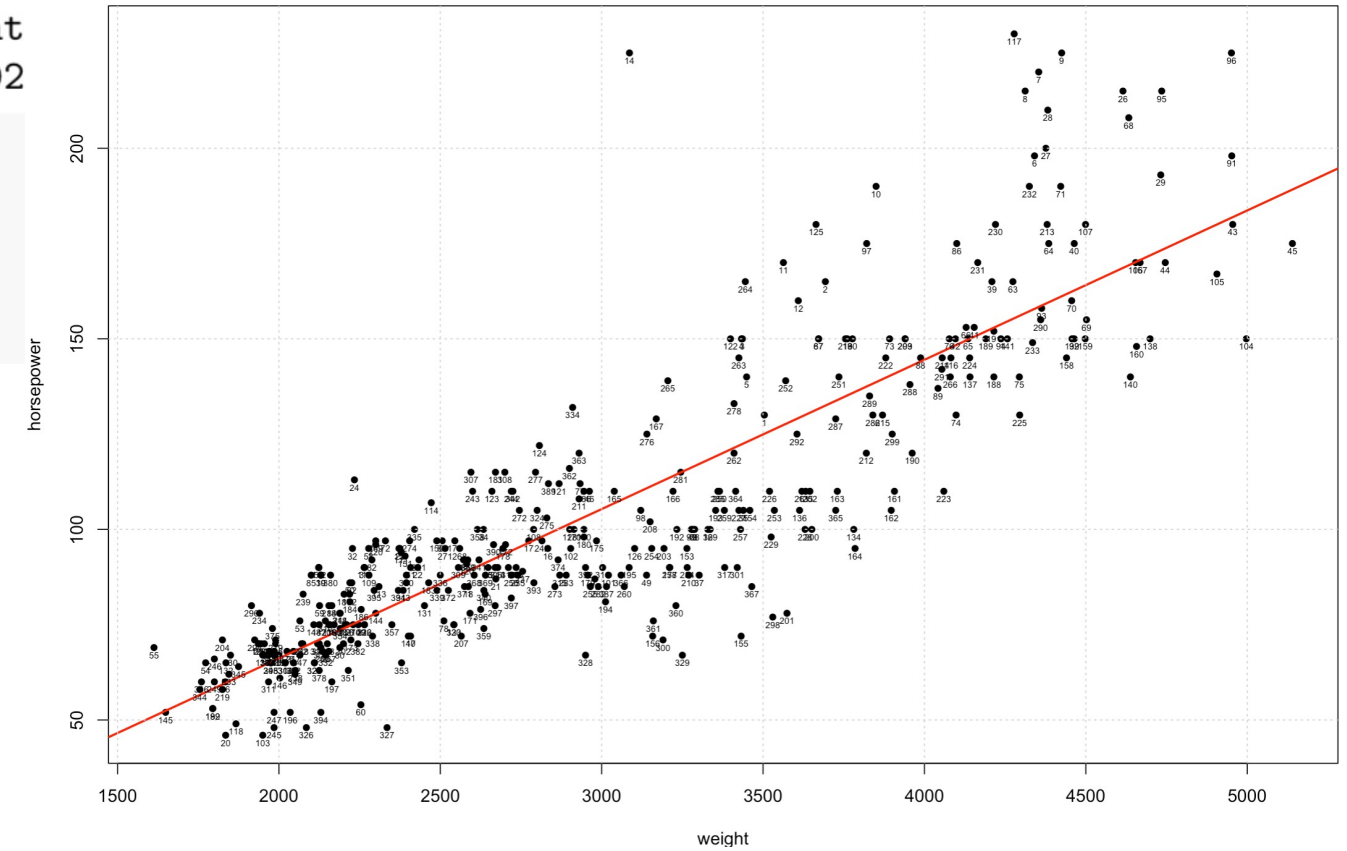
Example 2 – Auto dataset

```
plot(horsepower~weight,d2,pch=19,cex=0.75)
#
# linear model
#
m1=lm(horsepower~weight,d2)
coefficients(m1)
```

```
##   (Intercept)        weight
## -12.18348470   0.03917702
```

```
#
abline(m1)
abline(m1,col="red")
abline(m1,col="red",lwd=2)
grid()
```

```
# label all points (label is rownumber)
#
text(horsepower~weight,data=d2,
     labels=rownames(d2),pos=1,offset=0.25,
     cex=0.5)
```



Example 2 – Auto dataset

```
plot(horsepower~weight,d2,pch=19,cex=0.75)
#
# linear model
#
m1=lm(horsepower~weight,d2)
coefficients(m1)
```

```
##   (Intercept)        weight
## -12.18348470   0.03917702
```

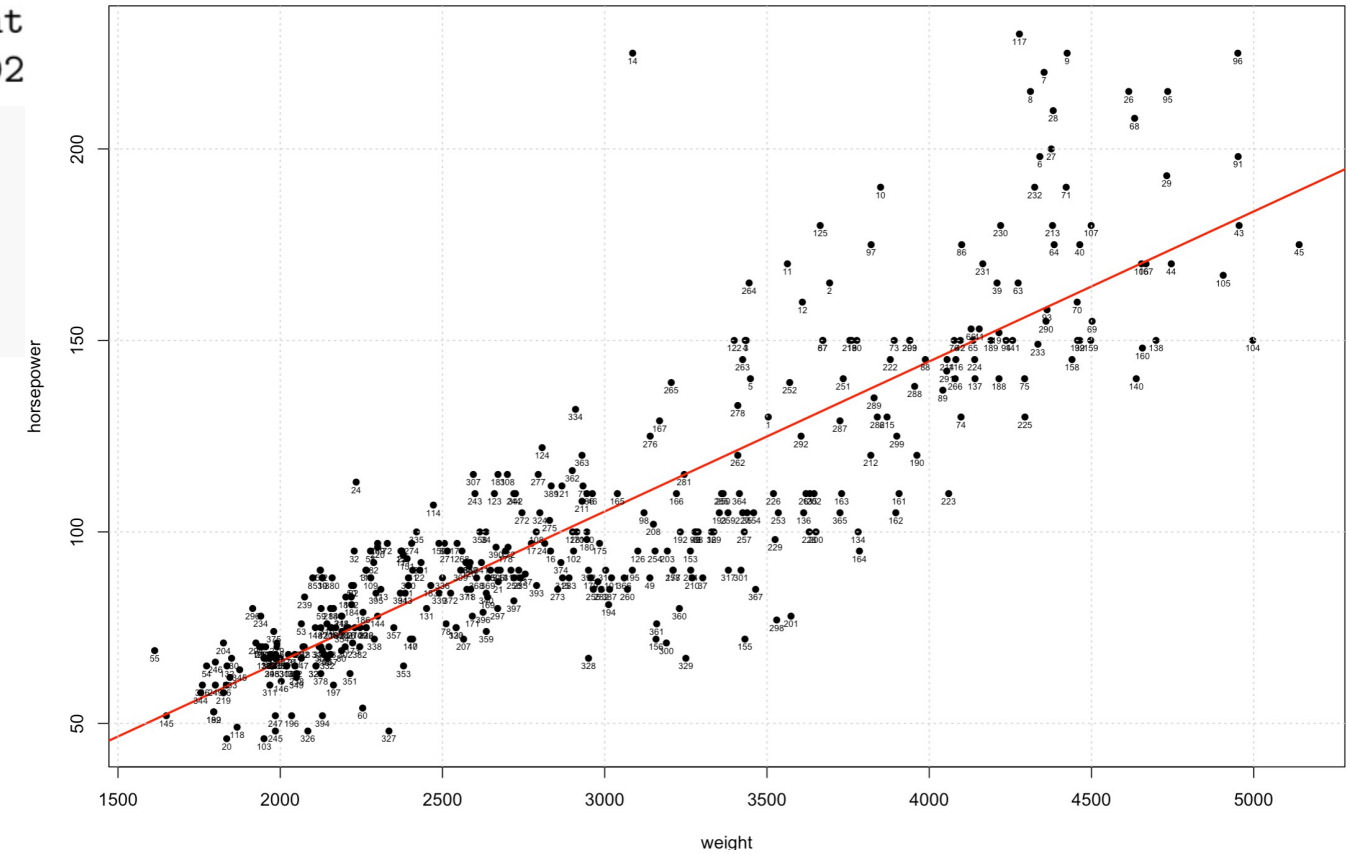
```
#
abline(m1)
abline(m1,col="red")
abline(m1,col="red",lwd=2)
grid()
```

```
# predict horsepower of a 3000-pound car
newval = data.frame(weight=3000)
newval
```

```
##   weight
## 1   3000
```

```
predict(m1,newval)
```

```
##        1
## 105.3476
```



Example 2 – Auto dataset

```
plot(horsepower~weight,d2,pch=19,cex=0.75)
#
# linear model
#
m1=lm(horsepower~weight,d2)
coefficients(m1)
```

```
##   (Intercept)        weight
## -12.18348470   0.03917702
```

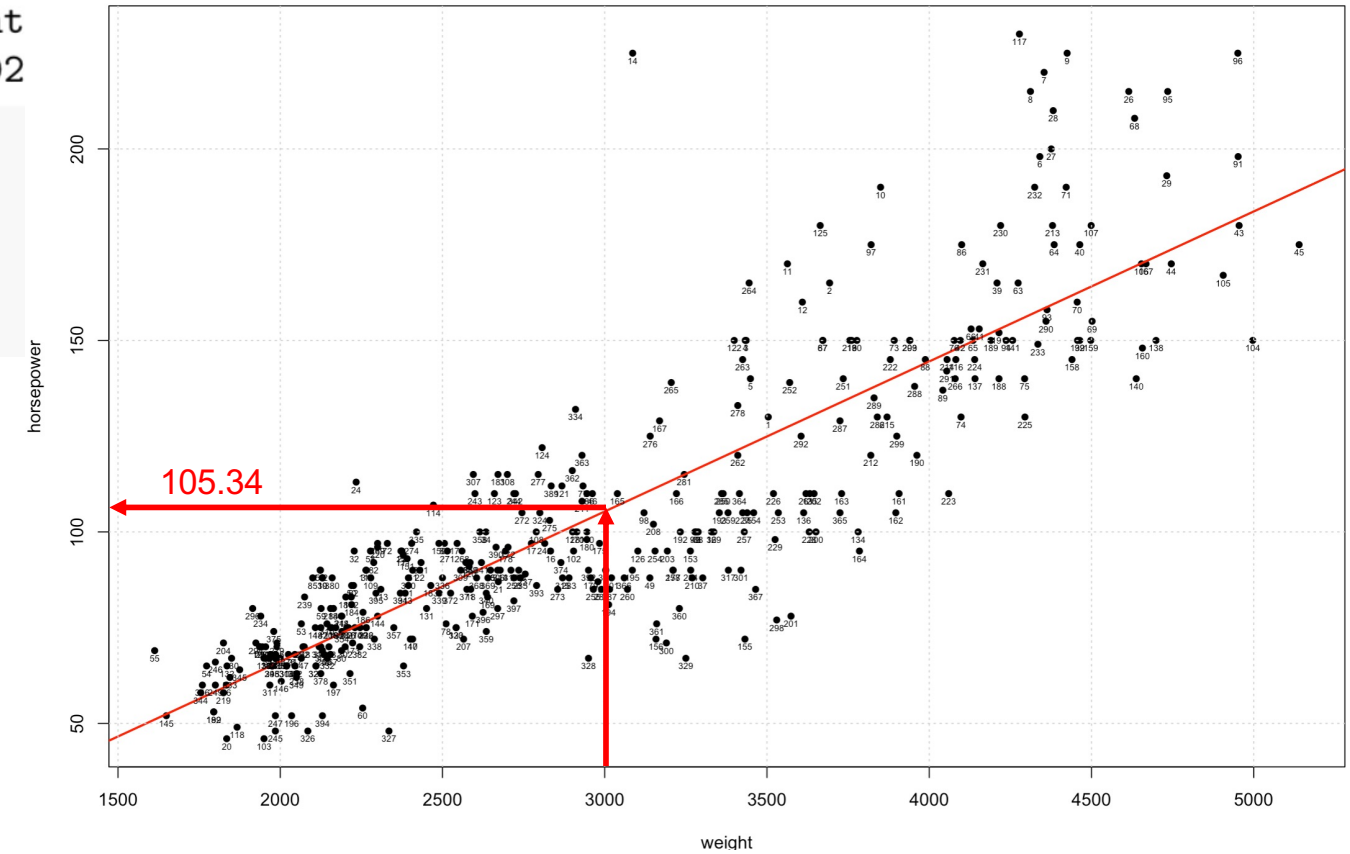
```
#
abline(m1)
abline(m1,col="red")
abline(m1,col="red",lwd=2)
grid()
```

```
# predict horsepower of a 3000-pound car
newval = data.frame(weight=3000)
newval
```

```
##   weight
## 1   3000
```

```
predict(m1,newval)
```

```
##        1
## 105.3476
```



Example 2 – Auto dataset

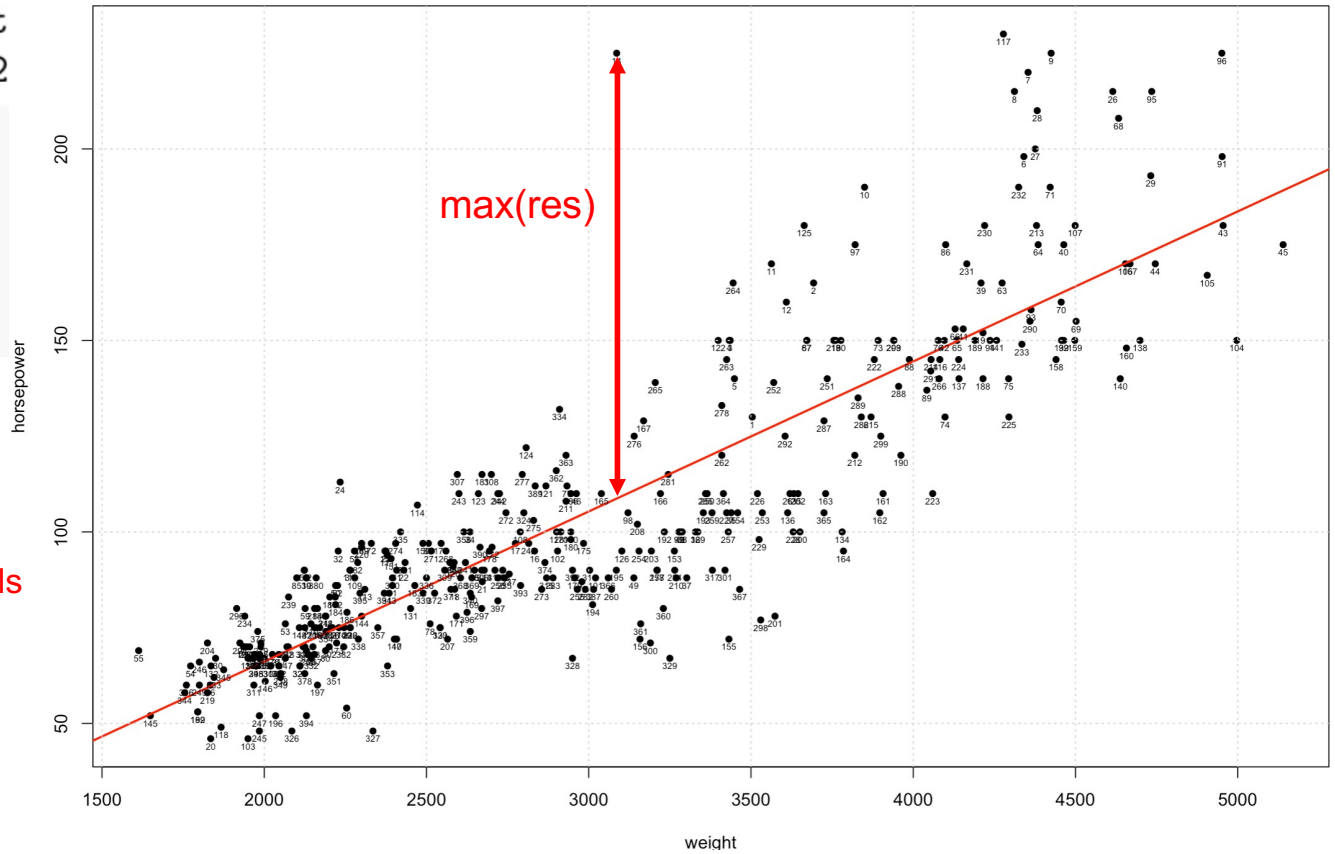
```
plot(horsepower~weight,d2,pch=19,cex=0.75)
#
# linear model
#
m1=lm(horsepower~weight,d2)
coefficients(m1)
```

```
##   (Intercept)        weight
## -12.18348470   0.03917702
```

```
#
abline(m1)
abline(m1,col="red")
abline(m1,col="red",lwd=2)
grid()
```

```
# largest outlier
#
res=resid(m1)
```

vector with 392 residuals

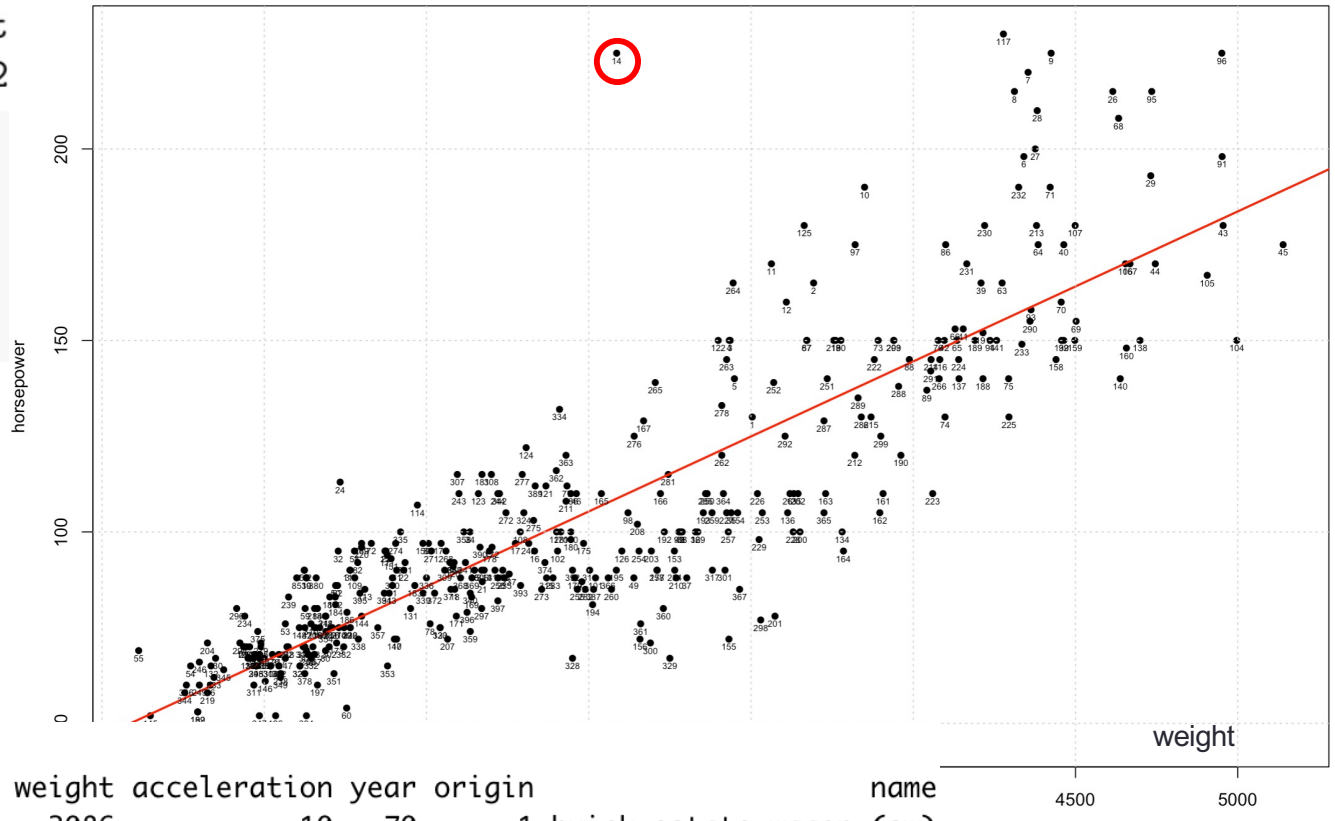


Example 2 – Auto dataset

```
plot(horsepower~weight,d2,pch=19,cex=0.75)
#
# linear model
#
m1=lm(horsepower~weight,d2)
coefficients(m1)
```

```
##   (Intercept)        weight
## -12.18348470   0.03917702
```

```
#
abline(m1)
abline(m1,col="red")
abline(m1,col="red",lwd=2)
grid()
```

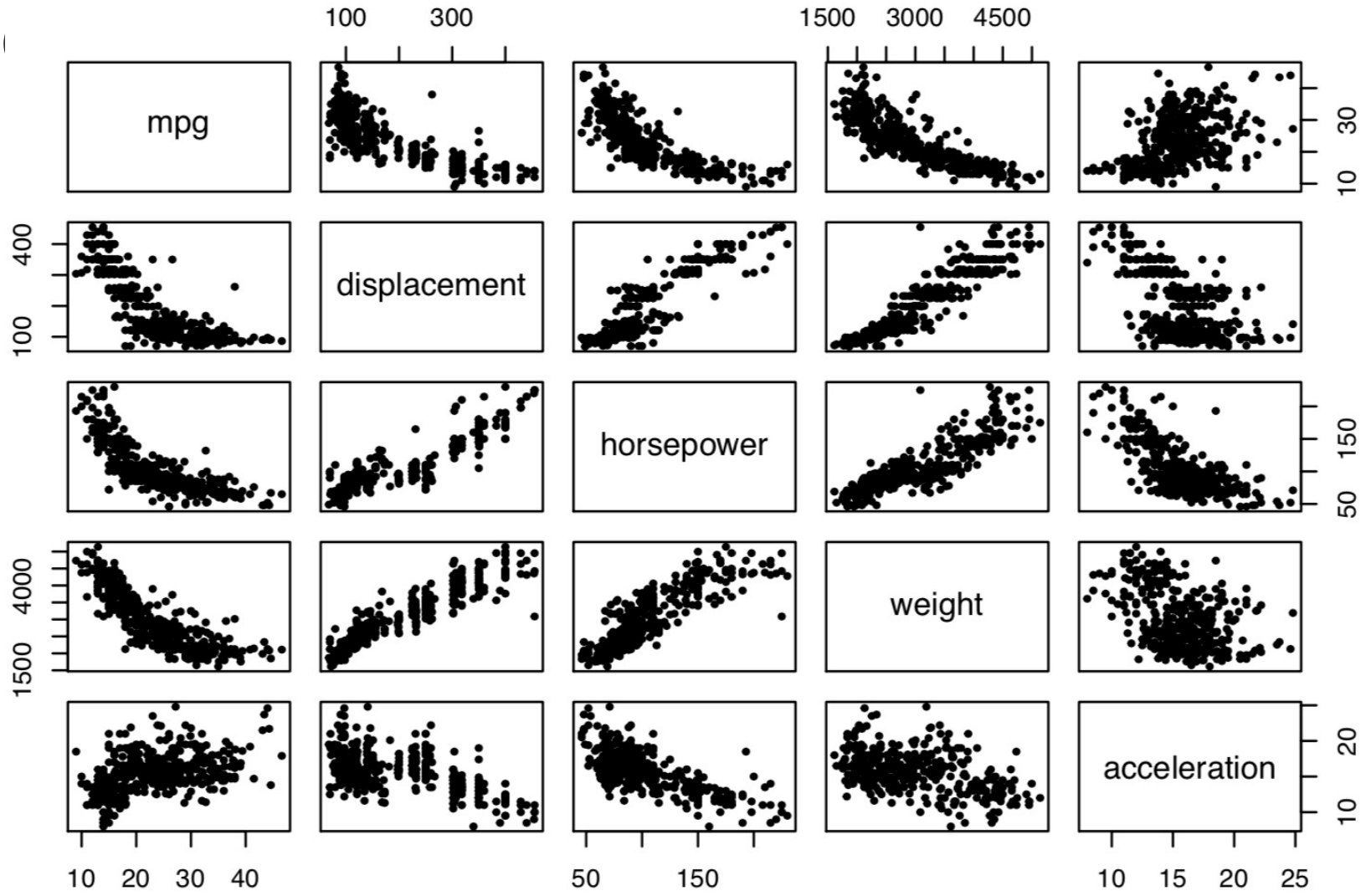



```
# largest outlier
#
res=resid(m1)
idx=which(res==max(res))
```

```
> d1[idx,]
   mpg cylinders displacement horsepower weight acceleration year origin                 name
14  14         8          455        225   3086           10   70      1 buick estate wagon (sw)
```

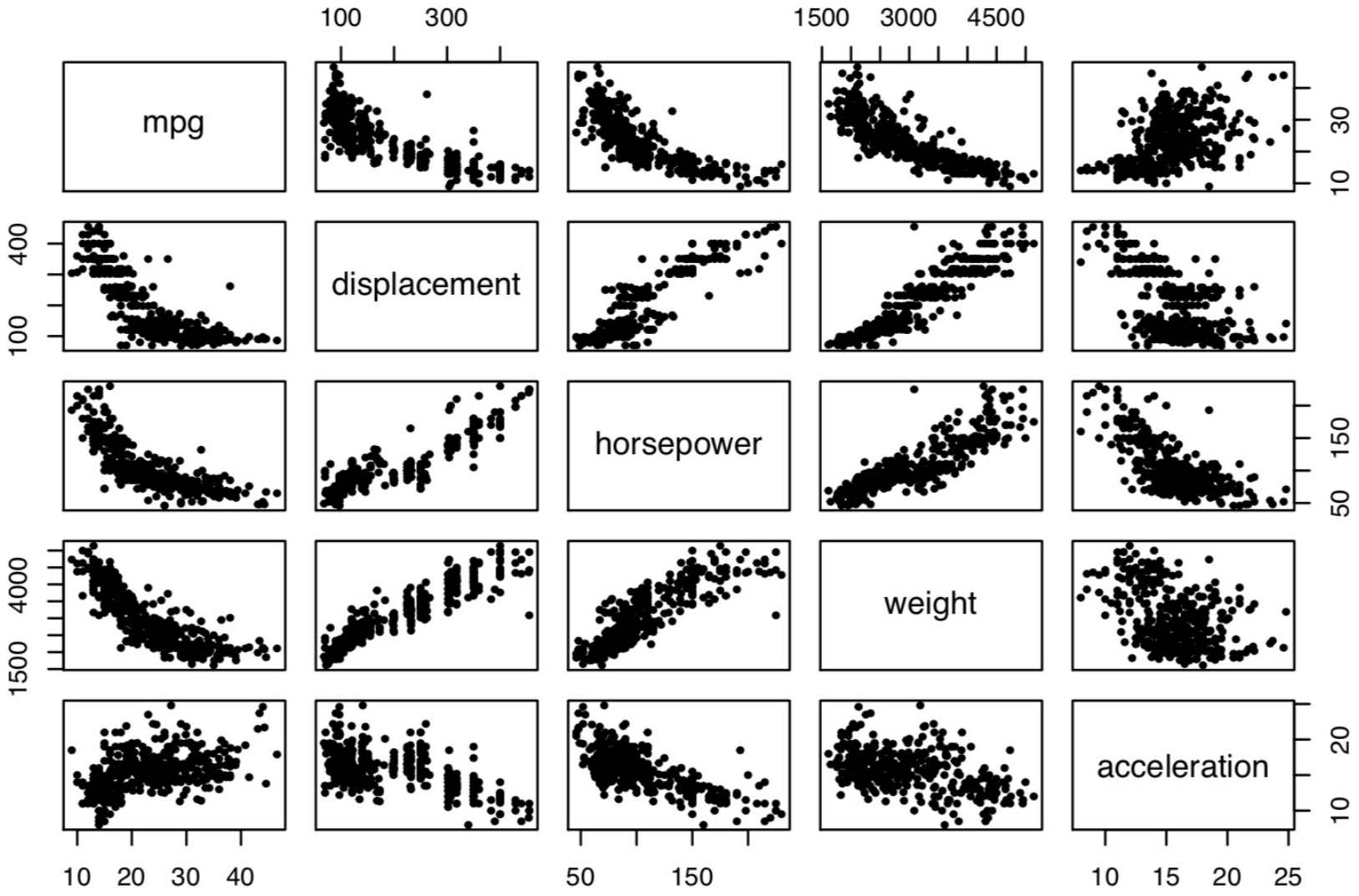
Example 2 – Auto dataset

The relation among all variables with (1) scatterplots



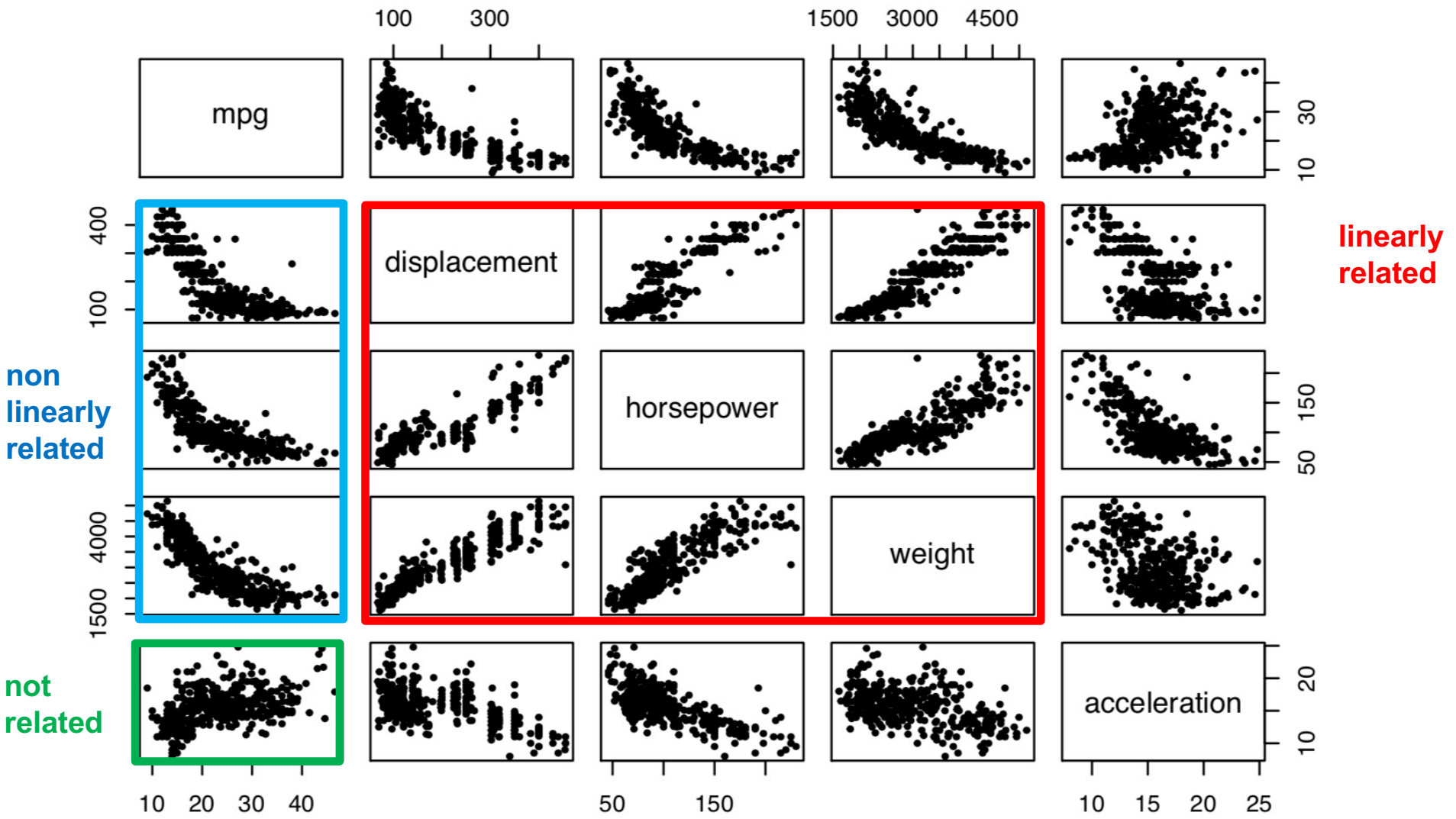
Example 2 – Auto dataset

```
pairs(~ mpg + displacement + horsepower + weight + acceleration,d2,
                                                    pch=19,cex=0.5)
```



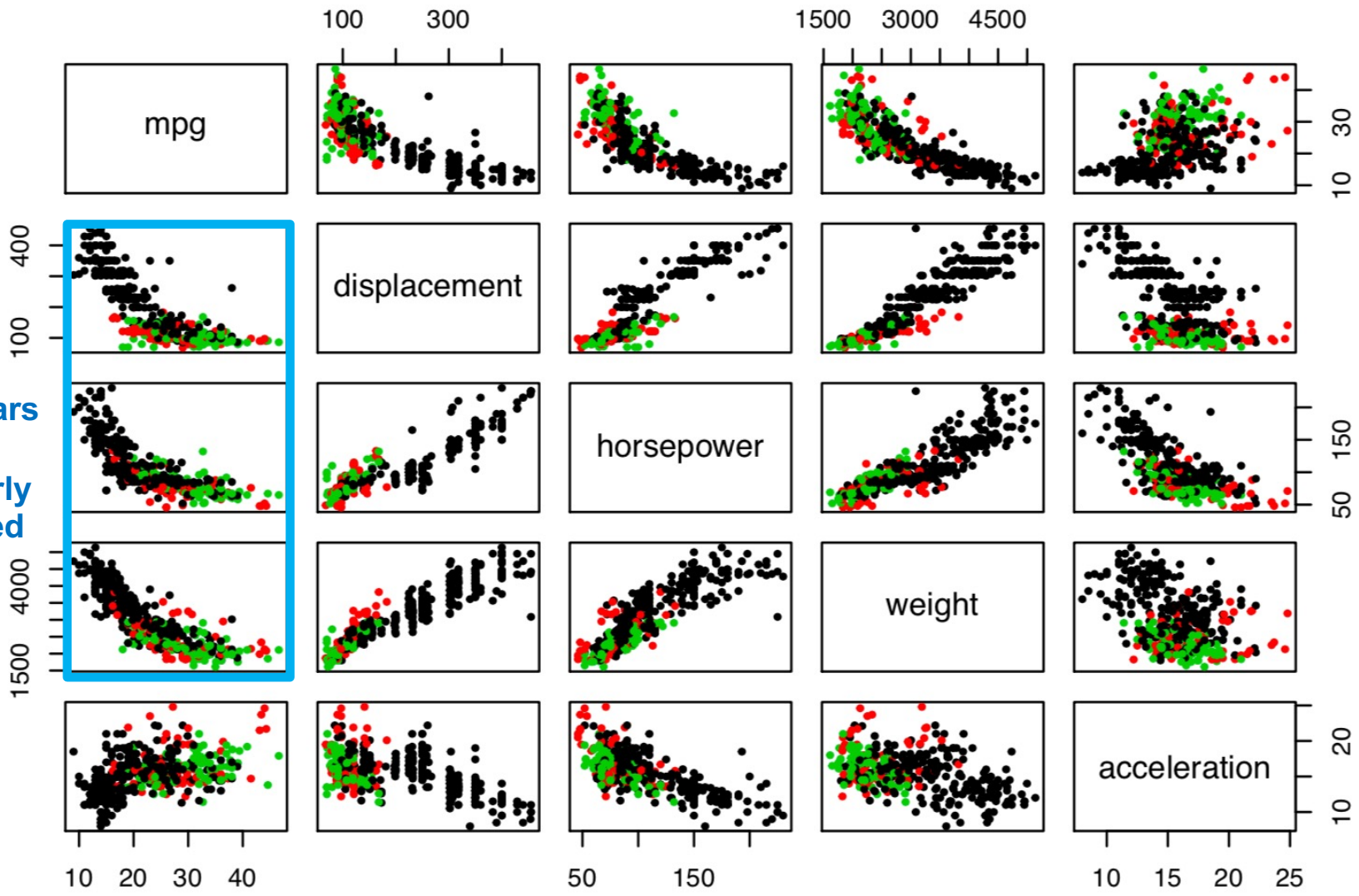
Example 2 – Auto dataset

```
pairs(~ mpg + displacement + horsepower + weight + acceleration,d2,
                                                    pch=19,cex=0.5)
```



Example 2 – Auto dataset

```
pairs(~ mpg + displacement + horsepower + weight + acceleration,d2,
      pch=19,cex=0.5,col=d1$origin)
```



only US cars non linearly related

Example 2 – Auto dataset

The relation among all variables with (2) correlations

```
d3=d2[,-c(2,7,8)]    ignore factors: cylinders, year and origin
cov(d3)
```

```
##                     mpg displacement  horsepower      weight acceleration
## mpg           60.918142    -657.5852  -233.85793  -5517.4407     9.115514
## displacement -657.585207   10950.3676  3614.03374  82929.1001  -156.994435
## horsepower   -233.857926    3614.0337  1481.56939  28265.6202   -73.186967
## weight      -5517.440704   82929.1001 28265.62023 721484.7090  -976.815253
## acceleration     9.115514    -156.9944   -73.18697   -976.8153     7.611331
```

```
cor(d3)
```

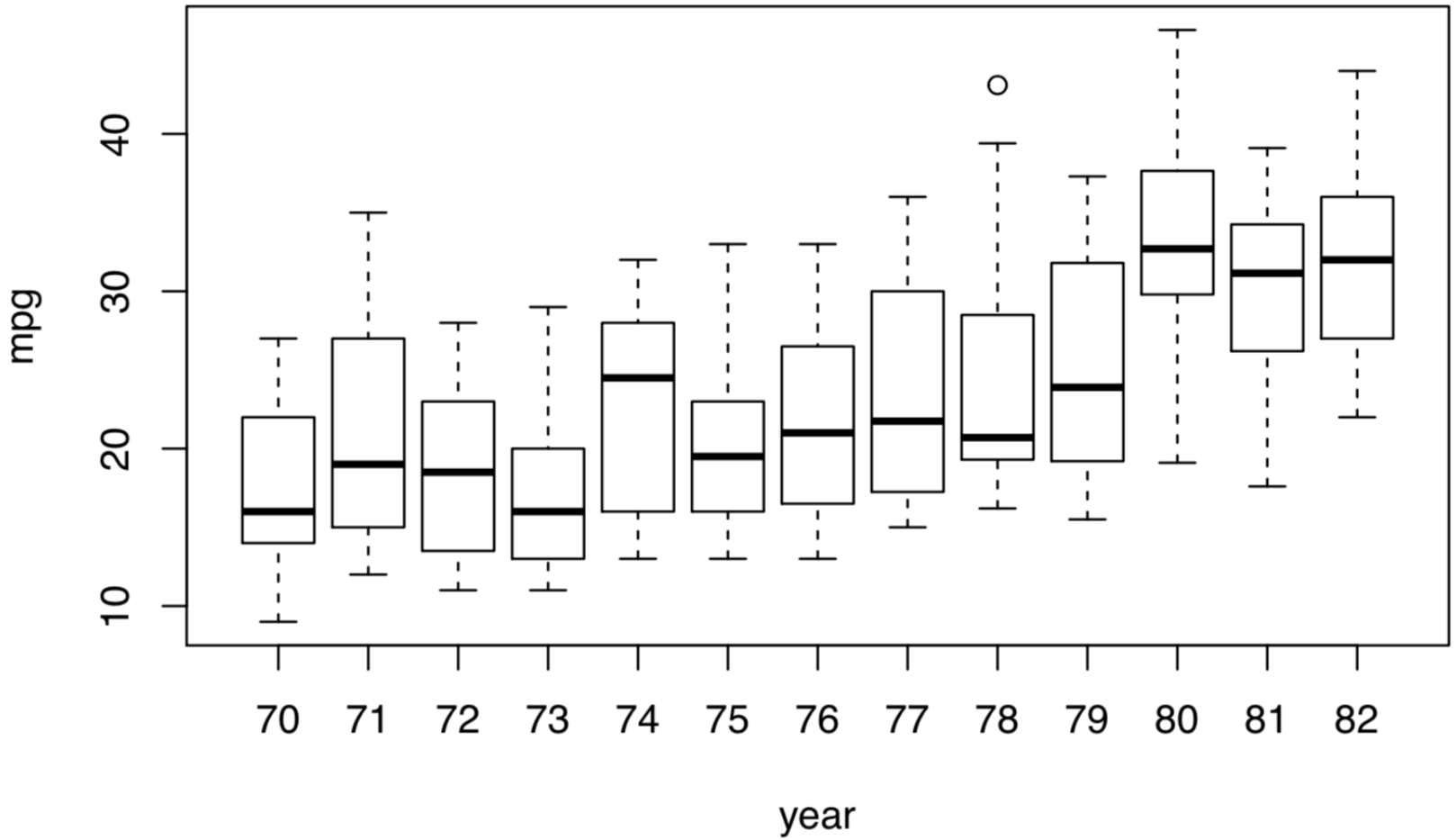
```
##                     mpg displacement horsepower     weight acceleration
## mpg           1.0000000   -0.8051269 -0.7784268 -0.8322442    0.4233285
## displacement -0.8051269    1.0000000  0.8972570  0.9329944   -0.5438005
## horsepower   -0.7784268    0.8972570  1.0000000  0.8645377   -0.6891955
## weight       -0.8322442    0.9329944  0.8645377  1.0000000   -0.4168392
## acceleration  0.4233285   -0.5438005 -0.6891955 -0.4168392    1.0000000
```

Example 2 – Auto dataset

The distribution of mpg over the years

```
plot(mpg~year,d3)
```

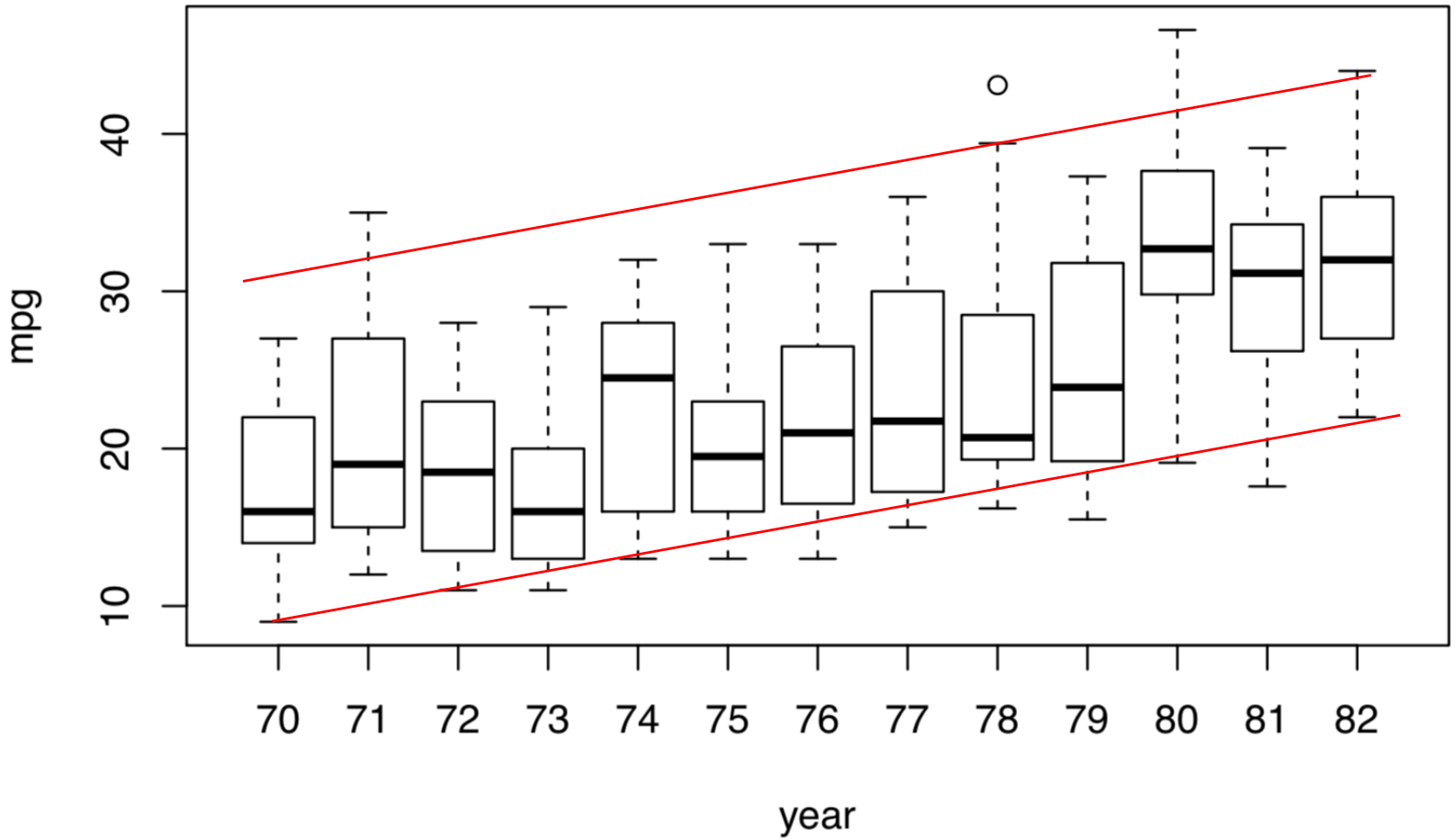
year is a factor



Example 2 – Auto dataset

The distribution of mpg shifts up over the years

```
plot(mpg~year,d3)
```



Box-and-whisker plot (boxplot)

- Graphical display showing key values of a variable distribution
- Key values: Center, spread, symmetry, and outliers
- Useful for comparing same variable on different categories