R ile Veri Ön İşleme

Muhammed Fatih TÜZEN

15 01 2022

İÇİNDEKİLER

1 Veri Ön İşleme

Veri ön işleme; istatistiksel modeller kurulmadan önce veri seti üzerinde yapılan bir takım düzeltme, eksik veriyi tamamlama, tekrarlanan verileri kaldırma, dönüştürme, bütünleştirme, temizleme, normalleştirme, boyut indirgeme vb. işlemlerdir. Bu aşamada ister istemez veri üzerinde bilgi keşfi yapılmış olur. Veri önişleme istatistiksel bir modelleme sürecinin büyük kısmını oluşturmaktadır. Kesin bir rakam olmamakla birlikte modelleme sürecinin yarısından fazlasının bu aşamada harcandığını ifade edebiliriz. Veri ön işleme temel anlamda 4 aşamadan oluşmaktadır. Bunlar sırasıyla şu şekildedir:

1. **Veri Temizleme :** Eksik verilerin tamamlanması, aykırı değerlerin teşhis edilmesi ve verilerdeki tutarsızlıkların giderilmesi gibi işlemler yapılmaktadır.
2. **Veri Birleştirme:** Farklı farklı veri tabanlarında bulunan veri setlerinin tek bir yerde toplanması aşamasının düzenli bir şekilde yürütülmesi sağlanır.
3. **Veri Dönüştürme :** Bu aşamada veriler, modelleme için uygun formlara dönüştürülürler. Veri dönüştürme; düzeltme, birleştirme, genelleştirme ve normalleştirme gibi değişik işlemlerden biri veya bir kaçını içerebilir. Veri normalleştirme , min-max dönüşümü, z standartlaştırması gibi yöntemler en sık kullanılan veri dönüştürme işlemlerinden bazılarıdır.
4. **Veri İndirgeme :** Daha küçük hacimli olarak veri kümesinin indirgenmiş bir örneğinin elde edilmesi amacıyla uygulanır. Bu sayede elde edilen indirgenmiş veri kümesine modelleme teknikleri uygulanarak daha etkin sonuçlar elde edilebilir. Veri Birleştirme (Data Aggregation), Boyut indirgeme (Dimension Reduction), Veri Sıkıştırma (Data Compression), Kesikli hale getirme (Discretization), Özellik Seçimi (Feature Selection) sık kullanılan veri indirgeme işlemlerindendir.

Bu dokümanda eksik veriler (missing values), aykırı değerler (outliers) ve veri normalleştirme işlemleri R uygulamları ile anlatılacaktır.

2 Eksik Veriler

Eksik veriler (kayıp gözlem), veri toplamada kaçınılmaz bir durumdur ve üzerinde dikkatle durulmalıdır. Sistematik bir kayıp gözlem durumu yoksa ortada ciddi bir sorun yoktur. Ama rastgele olmayan bir hata varsa tüm kitleye dair yanlılık olacağı için bu durum göz ardı edilemez.

```
df <- data.frame(weight=c(rnorm(15,70,10),rep(NA,5)),
height=c(rnorm(17,165,20),rep(NA,3)))

set.seed(12345)
```

```r
rows <- sample(nrow(df))
df2 <- df[rows, ]

# eksik verilerin sorgulanması

is.na(df2) # sorgulanma
```

```
##    weight height
## 14  FALSE  FALSE
## 19   TRUE   TRUE
## 16   TRUE  FALSE
## 11  FALSE  FALSE
## 18   TRUE   TRUE
## 8   FALSE  FALSE
## 2   FALSE  FALSE
## 6   FALSE  FALSE
## 17   TRUE  FALSE
## 13  FALSE  FALSE
## 7   FALSE  FALSE
## 1   FALSE  FALSE
## 15  FALSE  FALSE
## 10  FALSE  FALSE
## 12  FALSE  FALSE
## 9   FALSE  FALSE
## 4   FALSE  FALSE
## 20   TRUE   TRUE
## 3   FALSE  FALSE
## 5   FALSE  FALSE
```

```r
which(is.na(df2)) #konum
```

```
## [1]  2  3  5  9 18 22 25 38
```

```r
sum(is.na(df2)) # toplam eksik veri sayısı
```

```
## [1] 8
```

```r
colSums(is.na(df2)) # değişken düzeyinde eksik veri sayısı
```

```
## weight height
##      5      3
```

```
df2[!complete.cases(df2), ] #en az bir tane eksik olan satırlar
```

```
##    weight   height
## 19     NA       NA
## 16     NA 140.4876
## 18     NA       NA
## 17     NA 160.6821
## 20     NA       NA
```

```
df2[complete.cases(df2), ]$weight
```

```
##  [1] 64.39677 73.55776 55.03720 59.83653 79.88417 55.77330 64.42455 65.90209
##  [9] 79.48667 69.71473 52.08442 71.00339 69.64758 60.06650 74.10770
```

2.1 naniar Paketi ile Eksik Veri İnceleme

```
library(naniar)
```

```
## Warning: package 'naniar' was built under R version 4.0.5
```

```
library(dplyr)

n_miss(df2) # saydırma
```

```
## [1] 8
```

```
n_miss(df2$weight)
```

```
## [1] 5
```

```
n_complete(df2) # toplam sayı
```

```
## [1] 32
```

```
prop_miss(df2) # eksik veri oranı
```

```
## [1] 0.2
```

```
prop_complete(df2) # dolu veri oranı
```

```
## [1] 0.8
```

```
prop_complete(df2$height)
```

```
## [1] 0.85
```

```
# airquality verisi
df_air <- as_tibble(airquality)
df_air
```

```
## # A tibble: 153 x 6
##    Ozone Solar.R  Wind  Temp Month   Day
##    <int>   <int> <dbl> <int> <int> <int>
##  1    41     190   7.4    67     5     1
##  2    36     118   8      72     5     2
##  3    12     149  12.6    74     5     3
##  4    18     313  11.5    62     5     4
##  5    NA      NA  14.3    56     5     5
##  6    28      NA  14.9    66     5     6
##  7    23     299   8.6    65     5     7
##  8    19      99  13.8    59     5     8
##  9     8      19  20.1    61     5     9
## 10    NA     194   8.6    69     5    10
## # ... with 143 more rows
```

```
# eksik verileri özetleme
miss_var_summary(df_air) # değişken düzeyinde
```

```
## # A tibble: 6 x 3
##   variable n_miss pct_miss
##   <chr>     <int>    <dbl>
## 1 Ozone        37    24.2
## 2 Solar.R       7     4.58
## 3 Wind          0     0
## 4 Temp          0     0
## 5 Month         0     0
## 6 Day           0     0
```

```
miss_case_summary(df_air) # satır düzeyinde
```

```
## # A tibble: 153 x 3
##     case n_miss pct_miss
##    <int>  <int>    <dbl>
##  1     5      2     33.3
##  2    27      2     33.3
##  3     6      1     16.7
##  4    10      1     16.7
##  5    11      1     16.7
##  6    25      1     16.7
##  7    26      1     16.7
##  8    32      1     16.7
##  9    33      1     16.7
## 10    34      1     16.7
## # ... with 143 more rows
```

```
df_air %>% group_by(Month) %>% miss_var_summary() # grup düzeyinde
```

```
## # A tibble: 25 x 4
## # Groups:   Month [5]
##    Month variable n_miss pct_miss
##    <int> <chr>     <int>    <dbl>
##  1     5 Ozone         5     16.1
##  2     5 Solar.R       4     12.9
##  3     5 Wind          0      0
##  4     5 Temp          0      0
##  5     5 Day           0      0
##  6     6 Ozone        21     70
##  7     6 Solar.R       0      0
##  8     6 Wind          0      0
##  9     6 Temp          0      0
## 10     6 Day           0      0
## # ... with 15 more rows
```

```
df_air %>% group_by(Month) %>% miss_case_summary()
```

```
## # A tibble: 153 x 4
## # Groups:   Month [5]
##    Month  case n_miss pct_miss
##    <int> <int>  <int>    <dbl>
##  1     5     5      2       40
```

```
##  2     5    27      2       40
##  3     5     6      1       20
##  4     5    10      1       20
##  5     5    11      1       20
##  6     5    25      1       20
##  7     5    26      1       20
##  8     5     1      0        0
##  9     5     2      0        0
## 10     5     3      0        0
## # ... with 143 more rows
```

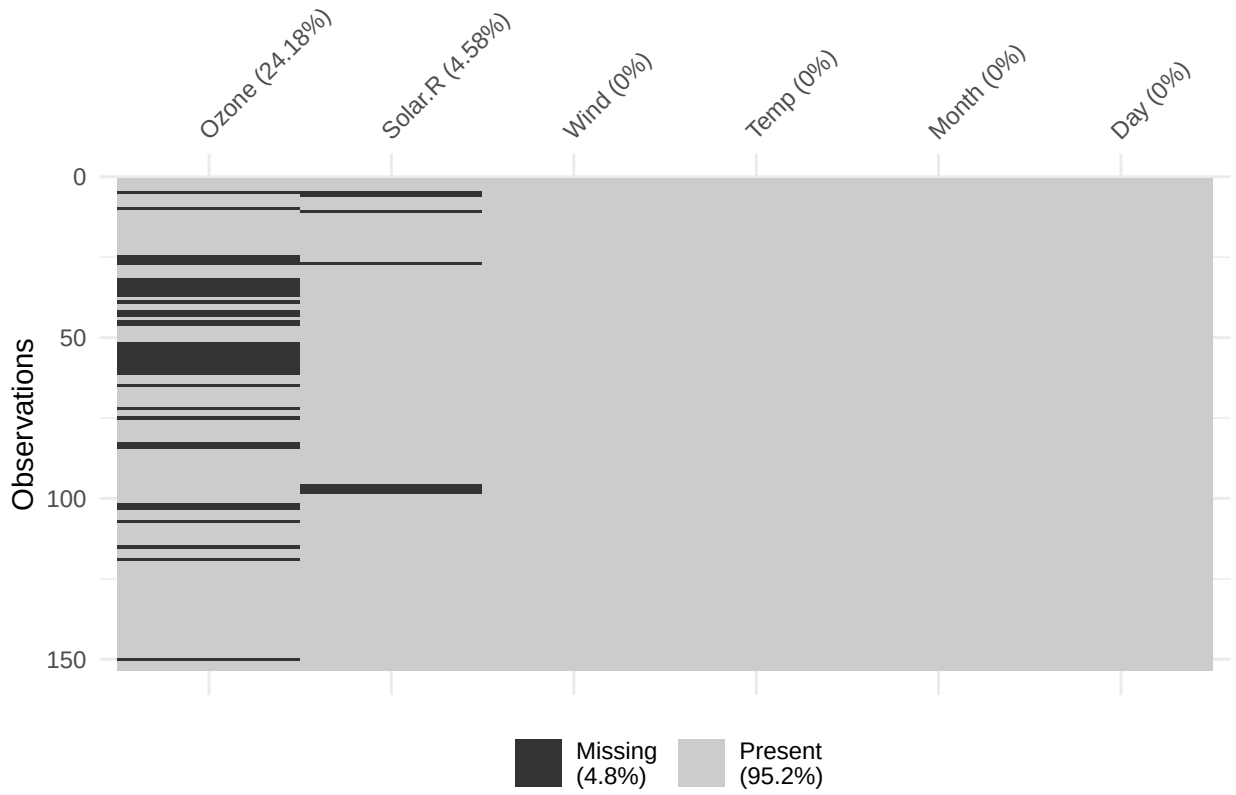
```
# eksik verileri tablolama
miss_var_table(df_air)
```

```
## # A tibble: 3 x 3
##   n_miss_in_var n_vars pct_vars
##           <int>  <int>    <dbl>
## 1             0      4     66.7
## 2             7      1     16.7
## 3            37      1     16.7
```

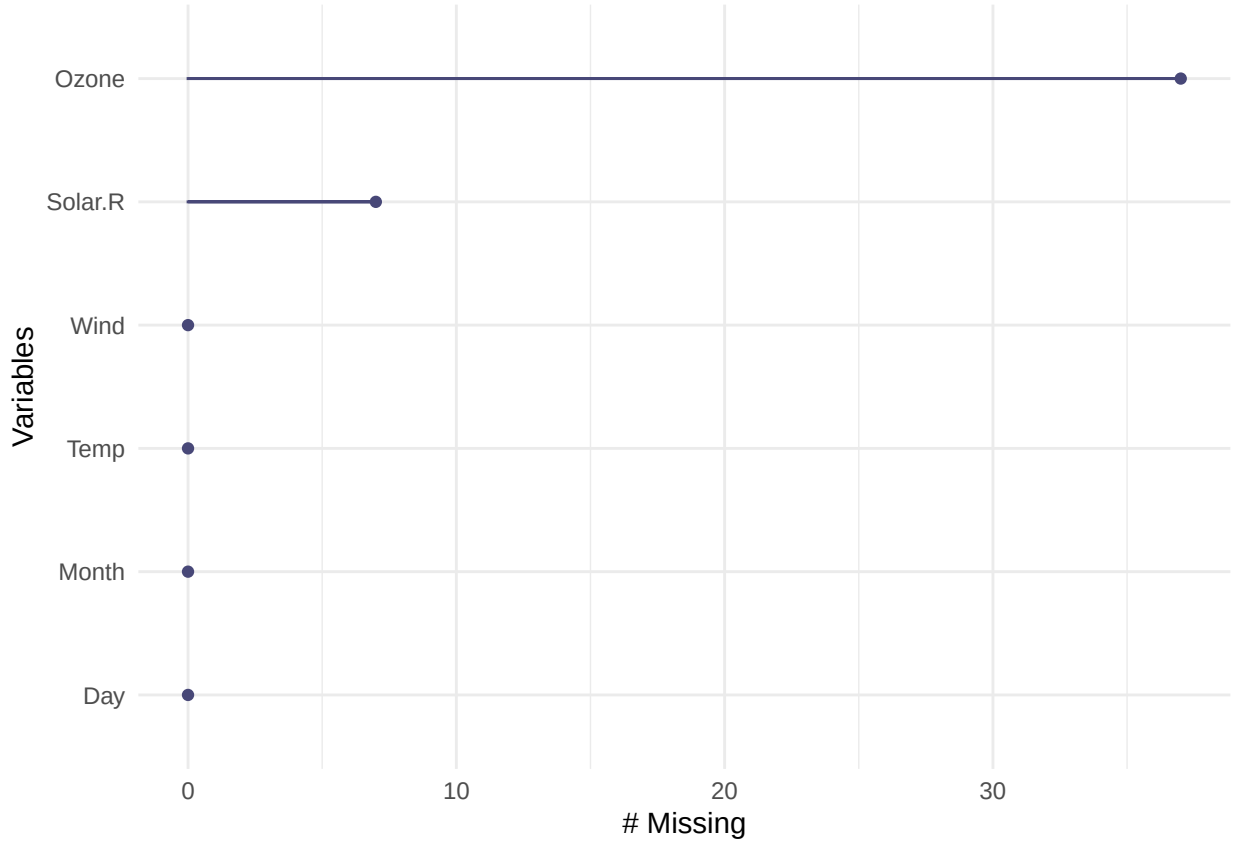
```
miss_case_table(df_air)
```

```
## # A tibble: 3 x 3
##   n_miss_in_case n_cases pct_cases
##            <int>   <int>     <dbl>
## 1              0     111     72.5
## 2              1      40     26.1
## 3              2       2      1.31
```

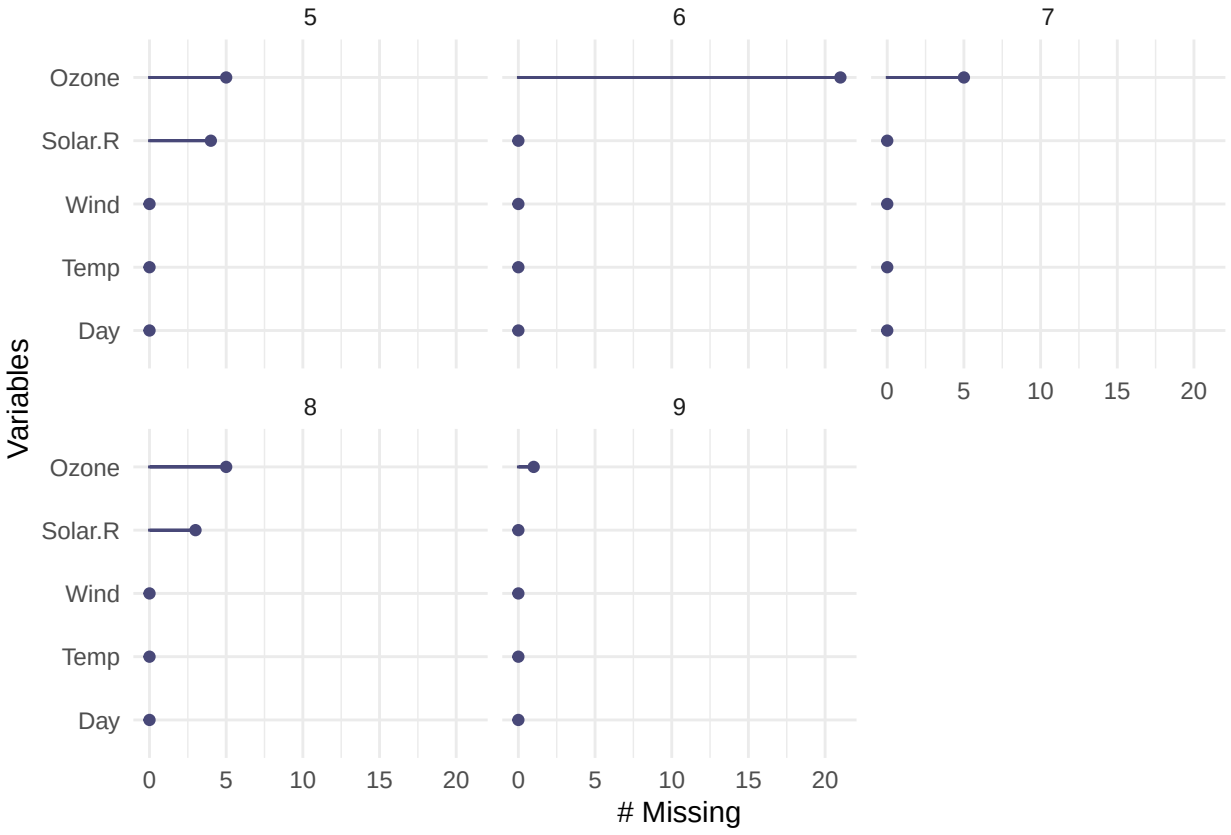
```
# eksik verileri görselleştirme
vis_miss(df_air)
```



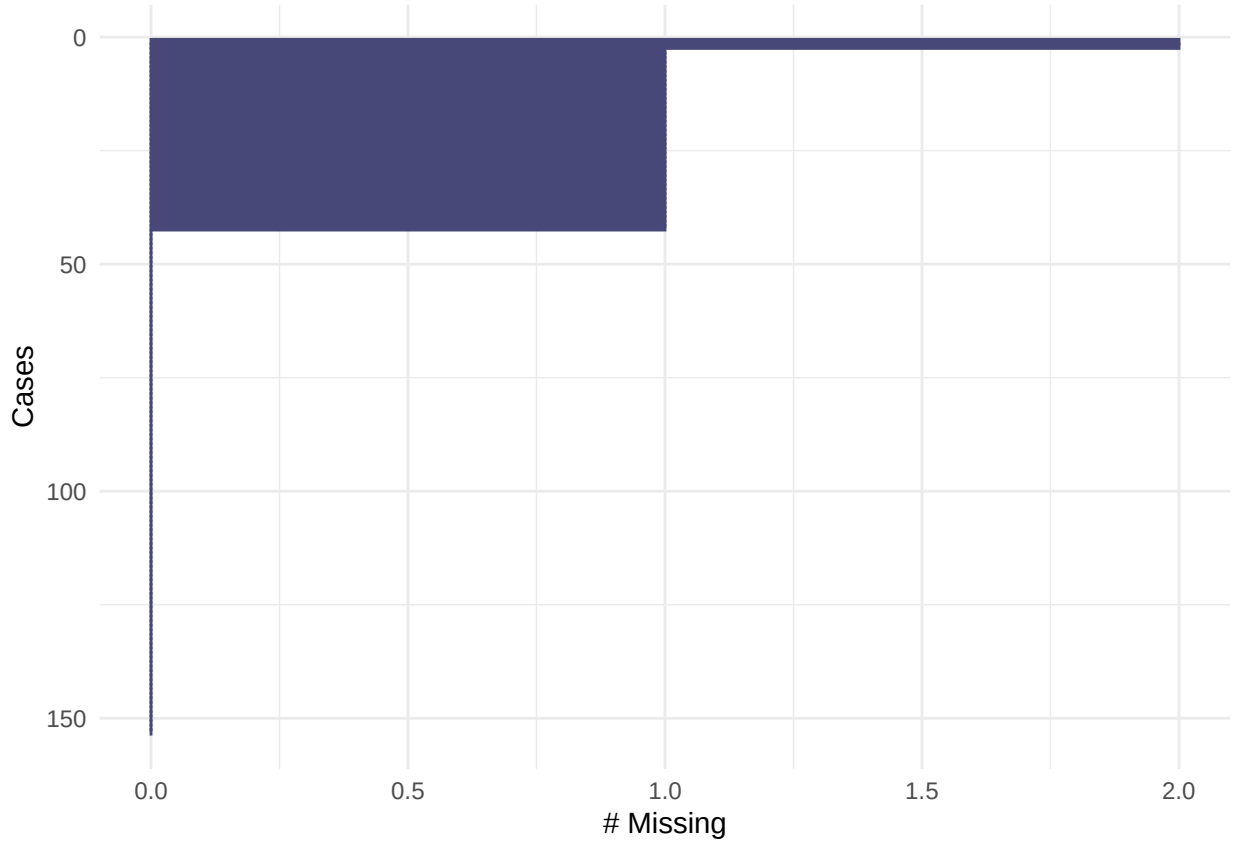

```
gg_miss_var(df_air)
```



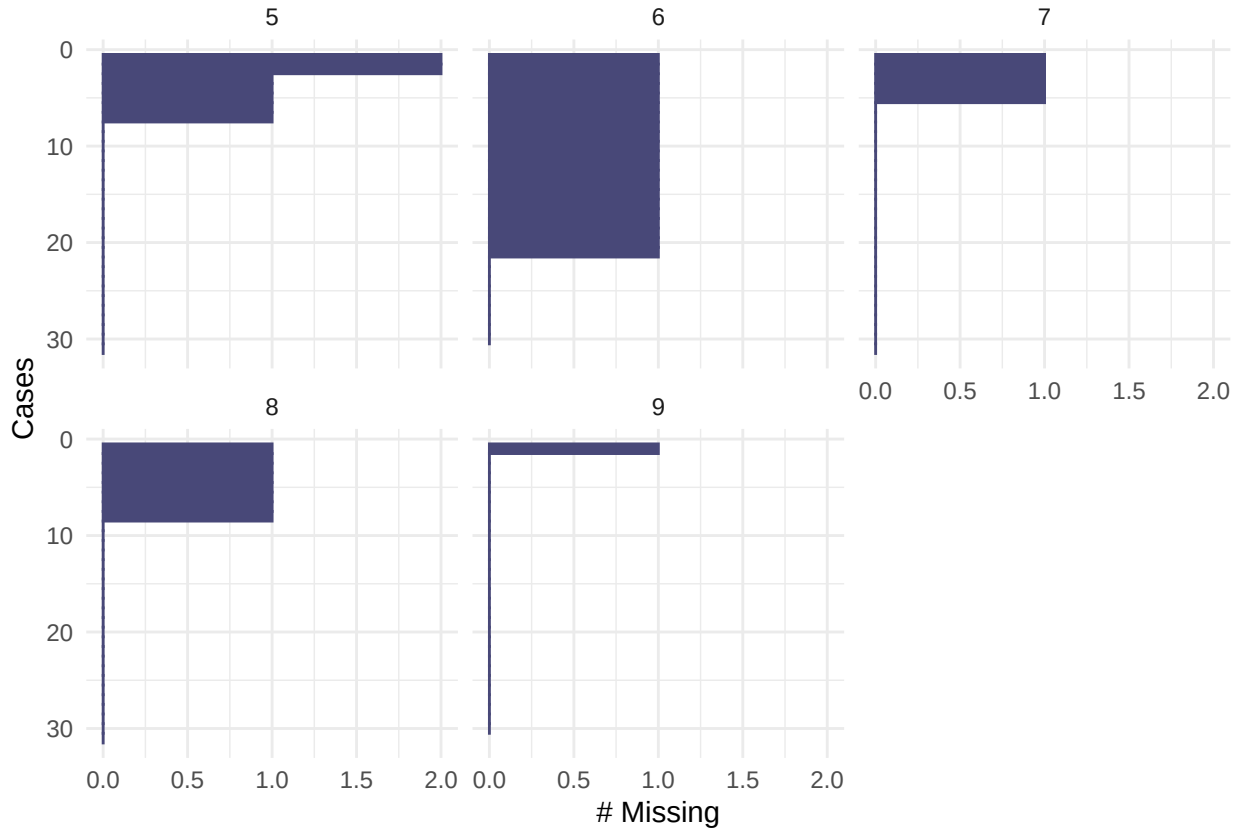

```
gg_miss_var(df_air, facet= Month)
```




```
gg_miss_case(df_air)
```




```
gg_miss_case(df_air,facet = Month)
```



2.2 Eksik Verileri Silme

```
# eksik veriden tamamen kurtulma
na.omit(df2)
```

```
##       weight   height
## 14 64.39677 154.3084
## 11 73.55776 145.8210
## 8  55.03720 155.3645
## 2  59.83653 149.3256
## 6  79.88417 169.3436
## 13 55.77330 167.4628
## 7  64.42455 197.2140
## 1  65.90209 193.8570
## 15 79.48667 157.3351
## 10 69.71473 153.9186
## 12 52.08442 170.8717
## 9  71.00339 163.6309
## 4  69.64758 181.8757
```

```
## 3  60.06650 156.7995
## 5  74.10770 155.6537
```

```
complete.cases(df2)
```

```
##  [1]  TRUE FALSE FALSE  TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
## [13]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE
```

```
df2[complete.cases(df2), ] # dolu olanlar satırlar
```

```
##      weight   height
## 14 64.39677 154.3084
## 11 73.55776 145.8210
## 8  55.03720 155.3645
## 2  59.83653 149.3256
## 6  79.88417 169.3436
## 13 55.77330 167.4628
## 7  64.42455 197.2140
## 1  65.90209 193.8570
## 15 79.48667 157.3351
## 10 69.71473 153.9186
## 12 52.08442 170.8717
## 9  71.00339 163.6309
## 4  69.64758 181.8757
## 3  60.06650 156.7995
## 5  74.10770 155.6537
```

```
df2[complete.cases(df2), ]$weight # değişken bazında dolu olan satırlar
```

```
##  [1] 64.39677 73.55776 55.03720 59.83653 79.88417 55.77330 64.42455 65.90209
##  [9] 79.48667 69.71473 52.08442 71.00339 69.64758 60.06650 74.10770
```

2.3 İmputasyon

```
# eksik verilere basit değer atama
df2$weight2 <- ifelse(is.na(df2$weight),mean(df2$weight, na.rm = TRUE),df2$weight)
sapply(df2, function(x) ifelse(is.na(x), mean(x, na.rm = TRUE), x ))
```

```
##         weight   height  weight2
##  [1,] 64.39677 154.3084 64.39677
##  [2,] 66.32822 163.1736 66.32822
##  [3,] 66.32822 140.4876 66.32822
##  [4,] 73.55776 145.8210 73.55776
##  [5,] 66.32822 163.1736 66.32822
##  [6,] 55.03720 155.3645 55.03720
##  [7,] 59.83653 149.3256 59.83653
##  [8,] 79.88417 169.3436 79.88417
##  [9,] 66.32822 160.6821 66.32822
## [10,] 55.77330 167.4628 55.77330
## [11,] 64.42455 197.2140 64.42455
## [12,] 65.90209 193.8570 65.90209
## [13,] 79.48667 157.3351 79.48667
## [14,] 69.71473 153.9186 69.71473
## [15,] 52.08442 170.8717 52.08442
## [16,] 71.00339 163.6309 71.00339
## [17,] 69.64758 181.8757 69.64758
## [18,] 66.32822 163.1736 66.32822
## [19,] 60.06650 156.7995 60.06650
## [20,] 74.10770 155.6537 74.10770
```

```
library(zoo)
sapply(df2, function(x) ifelse(is.na(x), na.locf(x), x )) # carry forward
```

```
##         weight   height  weight2
##  [1,] 64.39677 154.3084 64.39677
##  [2,] 64.39677 154.3084 66.32822
##  [3,] 64.39677 140.4876 66.32822
##  [4,] 73.55776 145.8210 73.55776
##  [5,] 73.55776 145.8210 66.32822
##  [6,] 55.03720 155.3645 55.03720
##  [7,] 59.83653 149.3256 59.83653
##  [8,] 79.88417 169.3436 79.88417
##  [9,] 79.88417 160.6821 66.32822
## [10,] 55.77330 167.4628 55.77330
## [11,] 64.42455 197.2140 64.42455
## [12,] 65.90209 193.8570 65.90209
## [13,] 79.48667 157.3351 79.48667
## [14,] 69.71473 153.9186 69.71473
## [15,] 52.08442 170.8717 52.08442
## [16,] 71.00339 163.6309 71.00339
## [17,] 69.64758 181.8757 69.64758
## [18,] 69.64758 181.8757 66.32822
```

```
## [19,] 60.06650 156.7995 60.06650
## [20,] 74.10770 155.6537 74.10770
```

```
sapply(df2, function(x) ifelse(is.na(x), na.locf(x,fromlast=TRUE), x ))
```

```
##         weight   height  weight2
##  [1,] 64.39677 154.3084 64.39677
##  [2,] 64.39677 154.3084 66.32822
##  [3,] 64.39677 140.4876 66.32822
##  [4,] 73.55776 145.8210 73.55776
##  [5,] 73.55776 145.8210 66.32822
##  [6,] 55.03720 155.3645 55.03720
##  [7,] 59.83653 149.3256 59.83653
##  [8,] 79.88417 169.3436 79.88417
##  [9,] 79.88417 160.6821 66.32822
## [10,] 55.77330 167.4628 55.77330
## [11,] 64.42455 197.2140 64.42455
## [12,] 65.90209 193.8570 65.90209
## [13,] 79.48667 157.3351 79.48667
## [14,] 69.71473 153.9186 69.71473
## [15,] 52.08442 170.8717 52.08442
## [16,] 71.00339 163.6309 71.00339
## [17,] 69.64758 181.8757 69.64758
## [18,] 69.64758 181.8757 66.32822
## [19,] 60.06650 156.7995 60.06650
## [20,] 74.10770 155.6537 74.10770
```

```
sapply(df2, function(x) ifelse(is.na(x), na.approx(x), x )) # linear interpolation
```

```
##         weight   height  weight2
##  [1,] 64.39677 154.3084 64.39677
##  [2,] 67.45043 147.3980 66.32822
##  [3,] 70.50410 140.4876 66.32822
##  [4,] 73.55776 145.8210 73.55776
##  [5,] 64.29748 150.5927 66.32822
##  [6,] 55.03720 155.3645 55.03720
##  [7,] 59.83653 149.3256 59.83653
##  [8,] 79.88417 169.3436 79.88417
##  [9,] 67.82873 160.6821 66.32822
## [10,] 55.77330 167.4628 55.77330
## [11,] 64.42455 197.2140 64.42455
## [12,] 65.90209 193.8570 65.90209
## [13,] 79.48667 157.3351 79.48667
```

```
## [14,] 69.71473 153.9186 69.71473
## [15,] 52.08442 170.8717 52.08442
## [16,] 71.00339 163.6309 71.00339
## [17,] 69.64758 181.8757 69.64758
## [18,] 64.85704 169.3376 66.32822
## [19,] 60.06650 156.7995 60.06650
## [20,] 74.10770 155.6537 74.10770
```

```
sapply(df2, function(x) ifelse(is.na(x), na.approx(x), x )) # cubic interpolation
```

```
##         weight   height  weight2
##  [1,] 64.39677 154.3084 64.39677
##  [2,] 67.45043 147.3980 66.32822
##  [3,] 70.50410 140.4876 66.32822
##  [4,] 73.55776 145.8210 73.55776
##  [5,] 64.29748 150.5927 66.32822
##  [6,] 55.03720 155.3645 55.03720
##  [7,] 59.83653 149.3256 59.83653
##  [8,] 79.88417 169.3436 79.88417
##  [9,] 67.82873 160.6821 66.32822
## [10,] 55.77330 167.4628 55.77330
## [11,] 64.42455 197.2140 64.42455
## [12,] 65.90209 193.8570 65.90209
## [13,] 79.48667 157.3351 79.48667
## [14,] 69.71473 153.9186 69.71473
## [15,] 52.08442 170.8717 52.08442
## [16,] 71.00339 163.6309 71.00339
## [17,] 69.64758 181.8757 69.64758
## [18,] 64.85704 169.3376 66.32822
## [19,] 60.06650 156.7995 60.06650
## [20,] 74.10770 155.6537 74.10770
```

```
# Hmisc paketi ile değer atama

library(Hmisc)
```

```
## Warning: package 'Hmisc' was built under R version 4.0.5
```

```
impute(df2$weight,mean)
```

```
##         1         2         3         4         5         6         7         8
##  64.39677 66.32822* 66.32822*  73.55776 66.32822*  55.03720  59.83653  79.88417
```

```
##         9        10        11        12        13        14        15        16
## 66.32822*  55.77330  64.42455  65.90209  79.48667  69.71473  52.08442  71.00339
##        17        18        19        20
##  69.64758 66.32822*  60.06650  74.10770
```

```
impute(df2$weight,median)
```

```
##         1         2         3         4         5         6         7         8
##  64.39677 65.90209* 65.90209*  73.55776 65.90209*  55.03720  59.83653  79.88417
##         9        10        11        12        13        14        15        16
## 65.90209*  55.77330  64.42455  65.90209  79.48667  69.71473  52.08442  71.00339
##        17        18        19        20
##  69.64758 65.90209*  60.06650  74.10770
```

```
sapply(df2, function(x) ifelse(is.na(x), impute(x,median), x ))
```

```
##         weight   height  weight2
##  [1,] 64.39677 154.3084 64.39677
##  [2,] 65.90209 157.3351 66.32822
##  [3,] 65.90209 140.4876 66.32822
##  [4,] 73.55776 145.8210 73.55776
##  [5,] 65.90209 157.3351 66.32822
##  [6,] 55.03720 155.3645 55.03720
##  [7,] 59.83653 149.3256 59.83653
##  [8,] 79.88417 169.3436 79.88417
##  [9,] 65.90209 160.6821 66.32822
## [10,] 55.77330 167.4628 55.77330
## [11,] 64.42455 197.2140 64.42455
## [12,] 65.90209 193.8570 65.90209
## [13,] 79.48667 157.3351 79.48667
## [14,] 69.71473 153.9186 69.71473
## [15,] 52.08442 170.8717 52.08442
## [16,] 71.00339 163.6309 71.00339
## [17,] 69.64758 181.8757 69.64758
## [18,] 65.90209 157.3351 66.32822
## [19,] 60.06650 156.7995 60.06650
## [20,] 74.10770 155.6537 74.10770
```

```
impute(df2$weight, 70) # özel değer atama
```

```
##         1         2         3         4         5         6         7         8
##  64.39677 70.00000* 70.00000*  73.55776 70.00000*  55.03720  59.83653  79.88417
```

```
##         9        10        11        12        13        14        15        16
## 70.00000*  55.77330  64.42455  65.90209  79.48667  69.71473  52.08442  71.00339
##        17        18        19        20
##  69.64758 70.00000*  60.06650  74.10770
```

```
# KNN (k-nearest neighbor) ile Değer Atama

library(DMwR2)
```

```
## Warning: package 'DMwR2' was built under R version 4.0.5
```

```
anyNA(df_air)
```

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

```
# airquality verisindeki Wind değişkeninin bazı değerlerini NA yapalım
set.seed(1234)
row_num <- sample(1:nrow(airquality),5)
row_num # bu satırdaki değerlere NA atanacak
```

```
## [1]  28  80 150 101 111
```

```
airquality_2 <- airquality
airquality_2[row_num,"Wind"] <- NA
airquality_2[row_num,"Wind"]
```

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

```
head(airquality_2,20)
```

```
##    Ozone Solar.R Wind Temp Month Day
## 1     41     190  7.4   67     5   1
## 2     36     118  8.0   72     5   2
## 3     12     149 12.6   74     5   3
## 4     18     313 11.5   62     5   4
## 5     NA      NA 14.3   56     5   5
## 6     28      NA 14.9   66     5   6
## 7     23     299  8.6   65     5   7
## 8     19      99 13.8   59     5   8
## 9      8      19 20.1   61     5   9
## 10    NA     194  8.6   69     5  10
```

```
## 11     7      NA  6.9   74     5  11
## 12    16     256  9.7   69     5  12
## 13    11     290  9.2   66     5  13
## 14    14     274 10.9   68     5  14
## 15    18      65 13.2   58     5  15
## 16    14     334 11.5   64     5  16
## 17    34     307 12.0   66     5  17
## 18     6      78 18.4   57     5  18
## 19    30     322 11.5   68     5  19
## 20    11      44  9.7   62     5  20
```

```
# k parametresi, verilen bir noktaya en yakın komşuların sayısıdır.
# Örneğin: k=5 olsun. Bu durumda mesafeye (öklit) göre en yakın 5 komşu belirlenir
# ve mesafenin ağırlıklı ortalaması hesaplanır.
# ağırlıklandırma, her komşuya 1 / d ağırlığının verilmesini içerir.
# burada d komşuya olan uzaklıktır.
knn_df_air <- knnImputation(airquality_2, k = 5) # k komşu sayısı

result <- data.frame(row=row_num,
                     orig=airquality[row_num,"Wind"],
                     knn=knn_df_air[row_num,"Wind"])
result
```

```
##   row orig       knn
## 1  28 12.0 10.079819
## 2  80  5.1  8.765250
## 3 150 13.2  9.914454
## 4 101  8.0  6.807361
## 5 111 10.9 11.237192
```

```
mean(result$orig-result$knn)
```

```
## [1] 0.4791848
```

Eksik verilerin analiz edilmesi ve imputasyon konusunda R içerisinde çeşitli kütühaneler bulunmaktadır. Bunlardan en çok bilinenleri **`mice`, `VIM`, `missForest`, `imputation`, `mi`, `Amelia`** paketleridir. Ayrıca **Sosyal Bilimler konuları içerisindeki eksik veriler bölümünden de** yararlanılabilir.

3 Aykırı Değer Analizi

Aykırı değer, diğer gözlemlerden uzak olan, yani diğer veri noktalarından önemli ölçüde farklı olan bir veri noktası olan bir değer veya gözlemdir. Bu dokümanda, tanımlayıcı

istatistikler (minimum, maksimum, histogram, kutu grafiği ve yüzdelikler dahil) gibi basit teknikler ve Hampel filtresi, Z-Skoru ile aykırı değer analizi anlatılacaktır.

3.1 Minumum ve Maximum

```
library(ggplot2)

# mpg verisindeki hwy değişkeni üzerinden inceleyelim
summary(mpg$hwy)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   12.00   18.00   24.00   23.44   27.00   44.00
```

```
min(mpg$hwy)
```

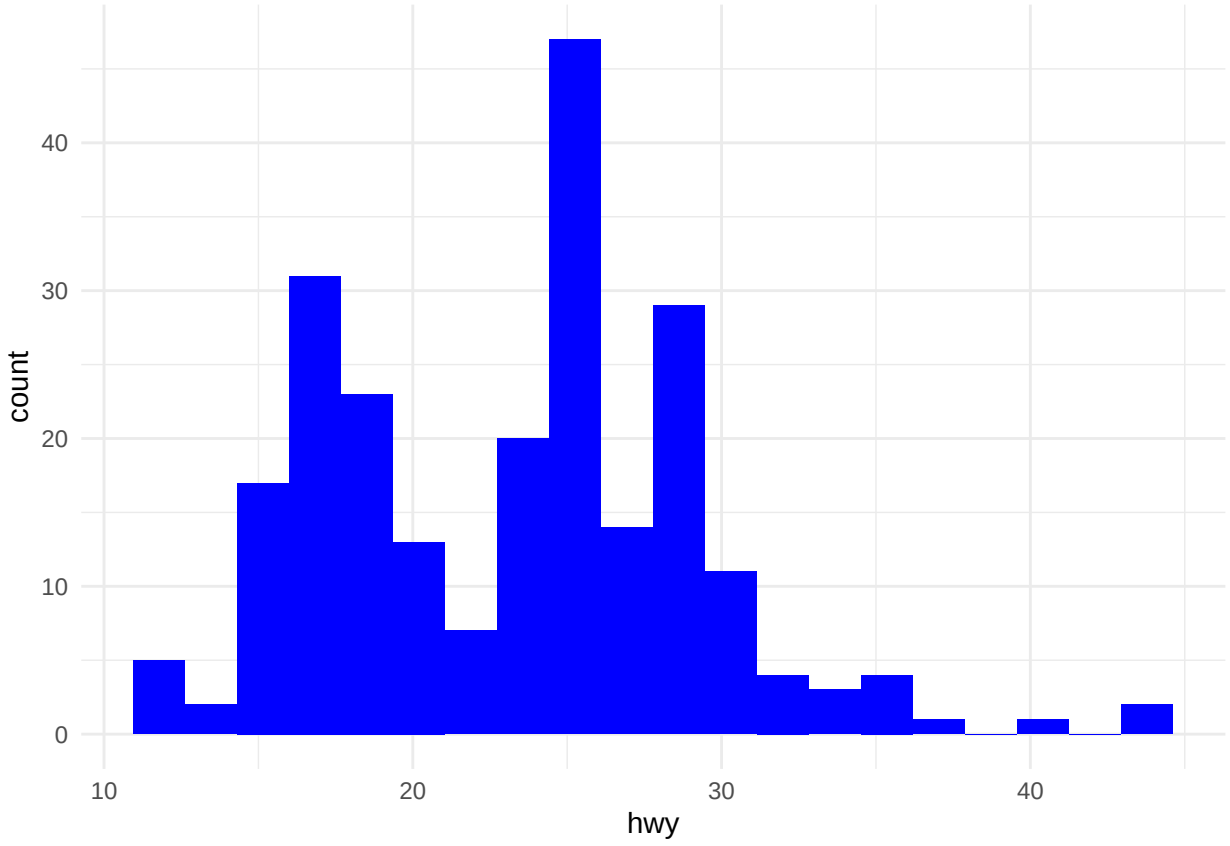
```
## [1] 12
```

```
max(mpg$hwy)
```

```
## [1] 44
```

3.2 Histogram

```
ggplot(mpg) +
  aes(x = hwy) +
  geom_histogram(bins = 20, fill = "blue") +
  theme_minimal()
```




```
# grafiğiin sağ tarafında kalan gözlemler şüpheli görünüyor.
```

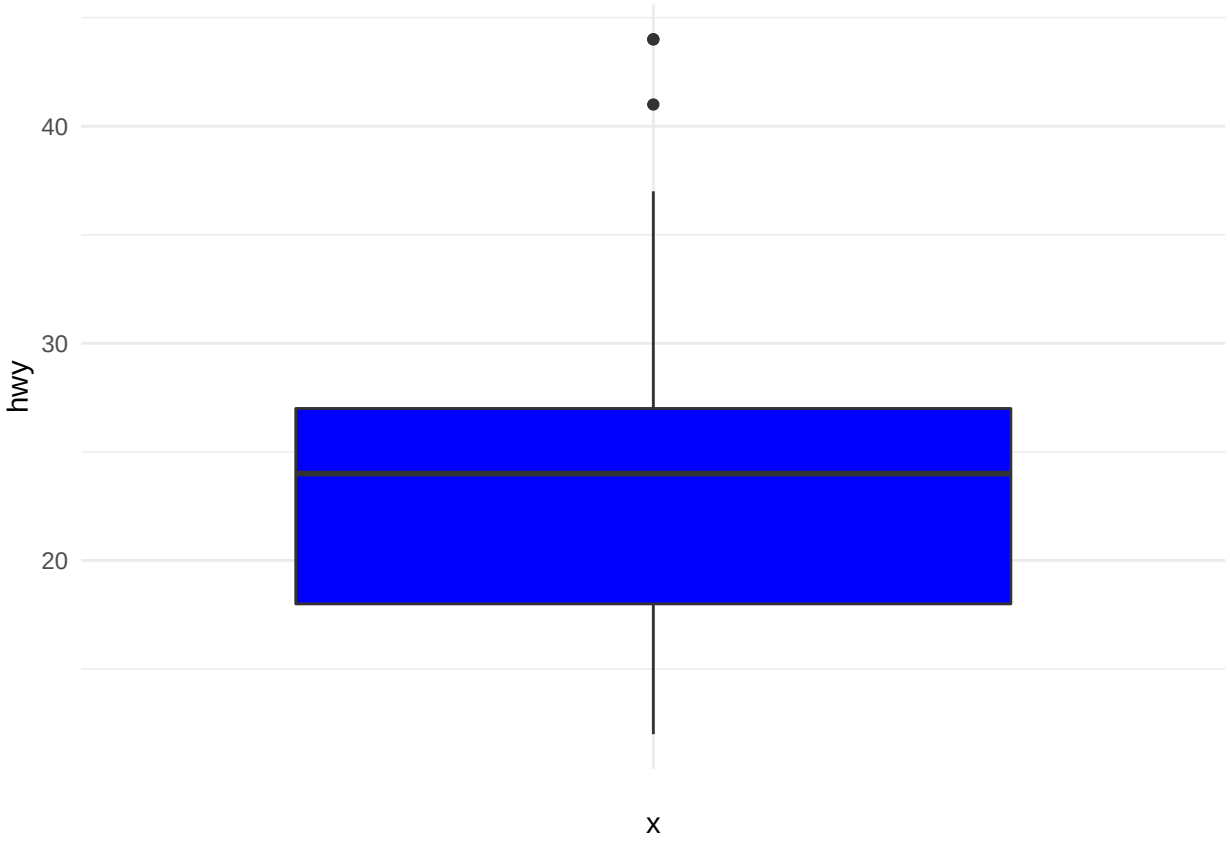
3.3 Boxplot

Bir boxplot grafiği, beş konum özetini (minimum, ortanca, birinci ve üçüncü çeyrekler ve maksimum) ve çeyrekler arası aralık (IQR) kriteri kullanılarak şüpheli bir aykırı değer olarak sınıflandırılan herhangi bir gözlemi görüntüleyerek nicel bir değişkeni görselleştirmeye yardımcı olur.

$I = [Q_1 - 1.5 * IQR; Q_3 + 1.5 * IQR]$

IQR ise üçüncü ve birinci çeyrek arasındaki farktır. R içerisindeki **`IQR()`** fonksiyonu bu amaçla kullanılabilir.

```
ggplot(mpg) +
  aes(x = "", y = hwy) +
  geom_boxplot(fill = "blue") +
  theme_minimal()
```




```
# outlier değrlerine erişim
boxplot.stats(mpg$hwy)$out
```

```
## [1] 44 44 41
```

```
# outier olarak görülen değerlerin konumları
hwy_out <- boxplot.stats(mpg$hwy)$out
hwy_out_sira <- which(mpg$hwy %in% c(hwy_out))
hwy_out_sira
```

```
## [1] 213 222 223
```

```
# outlier olarak görülen satırlar
mpg[hwy_out_sira, ]
```

```
## # A tibble: 3 x 11
##   manufacturer model   displ  year   cyl trans   drv     cty   hwy fl    class
##   <chr>        <chr>   <dbl> <int> <int> <chr>   <chr> <int> <int> <chr> <chr>
## 1 volkswagen   jetta     1.9  1999     4 manual~ f        33    44 d     compact
## 2 volkswagen   new be~   1.9  1999     4 manual~ f        35    44 d     subcom~
## 3 volkswagen   new be~   1.9  1999     4 auto(l~ f        29    41 d     subcom~
```

3.4 Yüzdelikler (Percentiles)

Bu aykırı değer tespiti yöntemi, yüzdelik dilimlere dayalıdır. Yüzdelikler yöntemiyle, 2,5 ve 97,5 yüzdelik dilimlerin oluşturduğu aralığın dışında kalan tüm gözlemler potansiyel aykırı değerler olarak kabul edilecektir. Aralığı oluşturmak için 1 ve 99 veya 5 ve 95 yüzdelikler gibi diğer yüzdelikler de düşünülebilir.

```
alt_sinir <- quantile(mpg$hwy, 0.025)
alt_sinir
```

```
## 2.5%
##   14
```

```
ust_sinir <- quantile(mpg$hwy, 0.975)
ust_sinir
```

```
##  97.5%
## 35.175
```

```
# Bu yönteme göre, 14'ün altındaki ve 35.175'in üzerindeki tüm gözlemler,
# potansiyel aykırı değerler olarak kabul edilecektir.

outlier_sira <- which(mpg$hwy < alt_sinir | mpg$hwy > ust_sinir)
outlier_sira
```

```
##  [1]  55  60  66  70 106 107 127 197 213 222 223
```

```
# Bu yönteme göre 11 adet outlier bulunmuştur.
mpg[outlier_sira,]
```

```
## # A tibble: 11 x 11
##    manufacturer model    displ  year   cyl trans  drv     cty   hwy fl    class
##    <chr>        <chr>    <dbl> <int> <int> <chr>  <chr> <int> <int> <chr> <chr>
##  1 dodge        dakota ~   4.7  2008     8 auto(~ 4         9    12 e     pickup
##  2 dodge        durango~   4.7  2008     8 auto(~ 4         9    12 e     suv
##  3 dodge        ram 150~   4.7  2008     8 auto(~ 4         9    12 e     pickup
##  4 dodge        ram 150~   4.7  2008     8 manua~ 4         9    12 e     pickup
##  5 honda        civic      1.8  2008     4 auto(~ f        25    36 r     subco~
##  6 honda        civic      1.8  2008     4 auto(~ f        24    36 c     subco~
##  7 jeep         grand c~   4.7  2008     8 auto(~ 4         9    12 e     suv
##  8 toyota       corolla    1.8  2008     4 manua~ f        28    37 r     compa~
##  9 volkswagen   jetta      1.9  1999     4 manua~ f        33    44 d     compa~
## 10 volkswagen   new bee~   1.9  1999     4 manua~ f        35    44 d     subco~
## 11 volkswagen   new bee~   1.9  1999     4 auto(~ f        29    41 d     subco~
```

```r
# Sınırları biraz daha küçültelim
alt_sinir <- quantile(mpg$hwy, 0.01)
ust_sinir <- quantile(mpg$hwy, 0.99)

outlier_sira <- which(mpg$hwy < alt_sinir | mpg$hwy > ust_sinir)

mpg[outlier_sira, ]
```

```
## # A tibble: 3 x 11
##   manufacturer model   displ  year   cyl trans   drv     cty   hwy fl    class
##   <chr>        <chr>   <dbl> <int> <int> <chr>   <chr> <int> <int> <chr> <chr>
## 1 volkswagen   jetta     1.9  1999     4 manual~ f        33    44 d     compact
## 2 volkswagen   new be~   1.9  1999     4 manual~ f        35    44 d     subcom~
## 3 volkswagen   new be~   1.9  1999     4 auto(l~ f        29    41 d     subcom~
```

```r
# Buna göre IQR ile elde edildiği gibi 3 adet outlier bulundu.
```

3.5 Hampel Filtresi

Hampel filtresi olarak bilinen başka bir yöntem, medyan, artı veya eksi 3 medyan mutlak sapma tarafından oluşturulan aralığın (I) dışındaki değerleri aykırı değer olarak değerlendirmekten oluşur.

$I = [median - 3 * MAD; median + 3 * MAD]$

MAD, medyan mutlak sapmadır ve verilerin medyanından mutlak sapmaların medyanı olarak tanımlanır. R içerisindeki **mad()** fonksiyonu bu amaçla kullanılabilir.

$MAD = median(|Xi - \tilde{X}|)$

```r
alt <- median(mpg$hwy) - 3 * mad(mpg$hwy, constant = 1)
alt
```

```
## [1] 9
```

```r
ust <- median(mpg$hwy) + 3 * mad(mpg$hwy, constant = 1)
ust
```

```
## [1] 39
```

```r
outliers_hampel <- which(mpg$hwy < alt | mpg$hwy > ust)
outliers_hampel
```

```
## [1] 213 222 223
```

```
mpg[outliers_hampel, ]
```

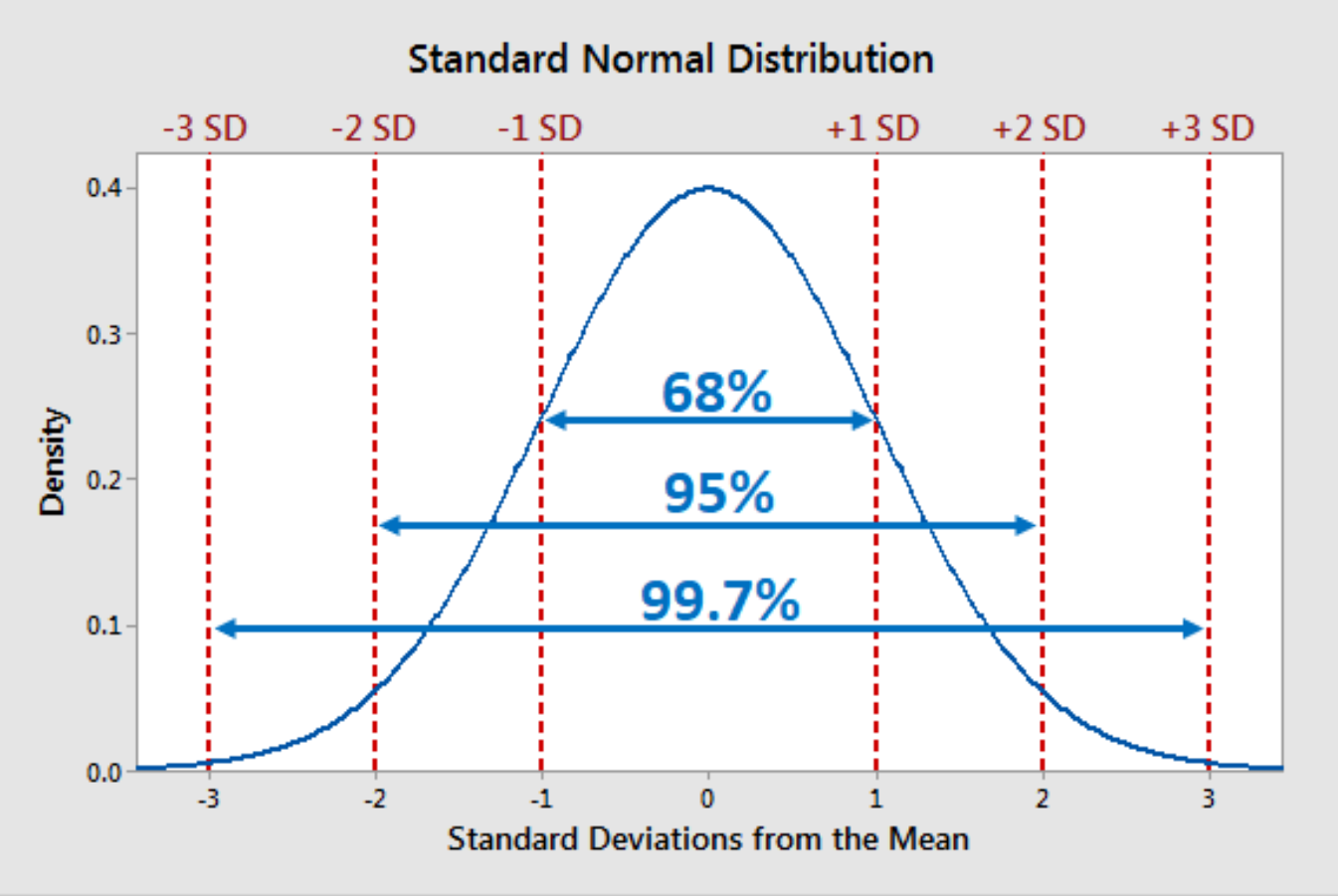
```
## # A tibble: 3 x 11
##   manufacturer model   displ  year   cyl trans   drv     cty   hwy fl    class
##   <chr>        <chr>   <dbl> <int> <int> <chr>   <chr> <int> <int> <chr> <chr>
## 1 volkswagen   jetta     1.9  1999     4 manual~ f        33    44 d     compact
## 2 volkswagen   new be~   1.9  1999     4 manual~ f        35    44 d     subcom~
## 3 volkswagen   new be~   1.9  1999     4 auto(l~ f        29    41 d     subcom~
```

```
# Hampel filtresine göre 3 adet outlier bulunmuştur.
```

3.6 Z-Skor Yöntemi

Aykırı değerlerin tespitinde ortalama ve standart sapmanın kulllanıldığı en bilinen yöntemlerdendir ve aşağıdaki şekilde hesaplanır.

$Z_i = \frac{(X_i - \mu)}{\sigma}$



```
std_z <- function(x){

  z=(x-mean(x))/sd(x)
  return(z)
}

mpg$hwy_std <- std_z(mpg$hwy)
mpg[,c("hwy","hwy_std")]
```

```
## # A tibble: 234 x 2
##      hwy hwy_std
##    <int>   <dbl>
##  1    29   0.934
##  2    29   0.934
##  3    31   1.27
##  4    30   1.10
##  5    26   0.430
##  6    26   0.430
##  7    27   0.598
##  8    26   0.430
##  9    25   0.262
## 10    28   0.766
## # ... with 224 more rows
```

```
# -3 ve +3 sapma dışında kalanları aykırı değer olarak kabul ediyoruz.
outliers_zskor <- which(mpg$hwy_std < -3 | mpg$hwy_std > +3)
outliers_zskor
```

```
## [1] 213 222
```

```
mpg[outliers_zskor,c() ]
```

```
## # A tibble: 2 x 0
```

```
# bu yönteme göre 2 adet aykırı değer bulunmuştur.
```

4 Veri Normalleştirme

Değişkenler farklı ölçeklerde ölçüldüğünde, genellikle analize eşit katkıda bulunmazlar. Örneğin, bir değişkenin değerleri 0 ile 100.000 arasında ve başka bir değişkenin değerleri 0

ile 100 arasında değişiyorsa, daha büyük aralığa sahip değişkene analizde daha büyük bir ağırlık verilecektir. Değişkenleri normalleştirerek, her bir değişkenin analize eşit katkı sağladığından emin olabiliriz. Değişkenleri normalleştirmek için (veya ölçeklendirmek) genellikle min-max ya da z dönüşümü yöntemleri kullanılır.

```
# min-max dönüşümleri

# 0 ile 1 arasi dönüşüm
std_0_1 <- function(x) {
  (x - min(x)) / (max(x) - min(x))
}

#-1 ile +1 arası dönüşüm
std_1_1 <- function(x) {
  ((x - mean(x)) / max(abs(x - mean(x))))
}

# a ile b arası dönüşüm
std_min_max <- function(x,a,b) {
  # a min değer
  # b max değer
  (a + ((x - min(x)) * (b - a)) / (max(x) - min(x)))
}

set.seed(12345)
dat <- data.frame(x = rnorm(20, 10, 3),
                  y = rnorm(20, 30, 8),
                  z = rnorm(20, 25, 5))
dat
```

```
##            x        y        z
## 1  11.756586 36.23698 30.64255
## 2  12.128398 41.64628 13.09821
## 3   9.672090 24.84537 19.69867
## 4   8.639508 17.57490 29.68570
## 5  11.817662 17.21832 29.27226
## 6   4.546132 44.44078 32.30365
## 7  11.890296 26.14682 17.93451
## 8   9.171448 34.96304 27.83702
## 9   9.147521 34.89699 27.91594
## 10  7.242034 28.70151 18.46601
## 11  9.651257 36.49499 22.29807
## 12 15.451936 47.57467 34.73846
## 13 11.111884 46.39352 25.26795
## 14 11.560649 43.05957 26.75831
```

```
## 15  7.748404 32.03417 21.64512
## 16 12.450700 33.92951 26.38977
## 17  7.340927 27.40731 28.45586
## 18  9.005267 16.70360 29.11898
## 19 13.362138 44.14187 35.72533
## 20 10.896171 30.20641 13.26528
```

```
summary(dat)
```

```
##        x                y               z        
##  Min.   : 4.546   Min.   :16.70   Min.   :13.10  
##  1st Qu.: 8.914   1st Qu.:27.09   1st Qu.:21.16  
##  Median :10.284   Median :34.41   Median :27.30  
##  Mean   :10.230   Mean   :33.23   Mean   :25.53  
##  3rd Qu.:11.836   3rd Qu.:42.00   3rd Qu.:29.38  
##  Max.   :15.452   Max.   :47.57   Max.   :35.73
```

```
apply(dat, 2, std_0_1)
```

```
##                x          y           z
##  [1,] 0.6611575 0.63274053 0.775368144
##  [2,] 0.6952505 0.80796300 0.000000000
##  [3,] 0.4700211 0.26373477 0.291705877
##  [4,] 0.3753393 0.02822392 0.733080320
##  [5,] 0.6667578 0.01667340 0.714808256
##  [6,] 0.0000000 0.89848463 0.848779748
##  [7,] 0.6734179 0.30589231 0.213738973
##  [8,] 0.4241150 0.59147416 0.651378062
##  [9,] 0.4219211 0.58933460 0.654866001
## [10,] 0.2471988 0.38864587 0.237228478
## [11,] 0.4681108 0.64109819 0.406585628
## [12,] 1.0000000 1.00000000 0.956385878
## [13,] 0.6020419 0.96173940 0.537838847
## [14,] 0.6431912 0.85374322 0.603705080
## [15,] 0.2936301 0.49659993 0.377728555
## [16,] 0.7248037 0.55799517 0.587417289
## [17,] 0.2562668 0.34672297 0.678727553
## [18,] 0.4088772 0.00000000 0.708033996
## [19,] 0.8083774 0.88880212 1.000000000
## [20,] 0.5822623 0.43739366 0.007383637
```

```
library(dplyr)

dat %>% mutate_all(std_0_1) %>% summary()
```

```
##        x                y                z         
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000  
##  1st Qu.:0.4005   1st Qu.:0.3365   1st Qu.:0.3562  
##  Median :0.5261   Median :0.5737   Median :0.6275  
##  Mean   :0.5211   Mean   :0.5354   Mean   :0.5492  
##  3rd Qu.:0.6684   3rd Qu.:0.8194   3rd Qu.:0.7194  
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
```

```
dat %>% mutate_all(std_1_1) %>% summary()
```

```
##        x                   y                   z          
##  Min.   :-1.000000   Min.   :-1.00000   Min.   :-1.0000  
##  1st Qu.:-0.231502   1st Qu.:-0.37143   1st Qu.:-0.3514  
##  Median : 0.009603   Median : 0.07154   Median : 0.1426  
##  Mean   : 0.000000   Mean   : 0.00000   Mean   : 0.0000  
##  3rd Qu.: 0.282624   3rd Qu.: 0.53057   3rd Qu.: 0.3098  
##  Max.   : 0.918881   Max.   : 0.86789   Max.   : 0.8207
```

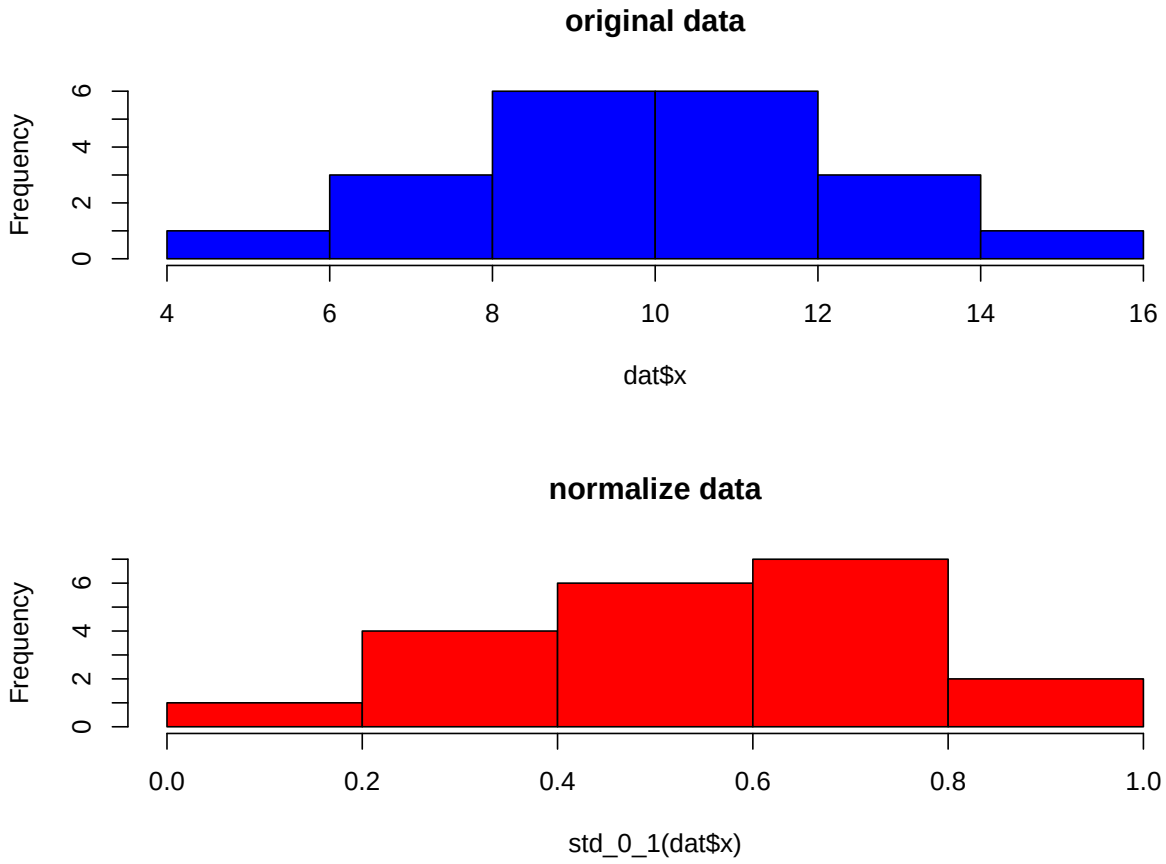
```
dat %>% mutate_all(std_min_max, a = -2, b = 2) %>% summary()
```

```
##        x                   y                  z          
##  Min.   :-2.00000   Min.   :-2.0000   Min.   :-2.0000  
##  1st Qu.:-0.39803   1st Qu.:-0.6539   1st Qu.:-0.5751  
##  Median : 0.10457   Median : 0.2947   Median : 0.5102  
##  Mean   : 0.08455   Mean   : 0.1415   Mean   : 0.1970  
##  3rd Qu.: 0.67369   3rd Qu.: 1.2776   3rd Qu.: 0.8775  
##  Max.   : 2.00000   Max.   : 2.0000   Max.   : 2.0000
```

```
dat %>% mutate_all(std_z) %>% summary()
```

```
##        x                   y                   z          
##  Min.   :-2.27173   Min.   :-1.7088   Min.   :-1.9165  
##  1st Qu.:-0.52591   1st Qu.:-0.6347   1st Qu.:-0.6735  
##  Median : 0.02182   Median : 0.1223   Median : 0.2732  
##  Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.0000  
##  3rd Qu.: 0.64204   3rd Qu.: 0.9067   3rd Qu.: 0.5937  
##  Max.   : 2.08745   Max.   : 1.4831   Max.   : 1.5729
```

```
par(mfrow=c(2,1))
hist(dat$x,main="original data",col="blue")
hist(std_0_1(dat$x),main="normalize data",col="red")
```




```
# bu dönüşümler verinin dağılımını değiştirmemektedir.
```