Data Transformation and Discretization

Hazim Fitri

2024-12-18

Contents

Data Transformation Techniques

Normalization

Min-Max Normalization

$$V = \frac{[X - min(X)] \times [new.max(X) - new.min(X)]}{max(X) - min(X)} + new.min(X)$$

```
dataAP3 <- read.csv('./Data/dataAP3.csv', header = T)
head(dataAP3)
```

```
##   X Month Day_of_month Day_of_week ozone_ppm pressure_height.hPA Wind_speed.mph
## 1 1     1            1           4      3.01                5480              8
## 2 2     1            2           5      3.20                5660              6
## 3 3     1            3           6      2.70                5710              4
## 4 4     1            4           7      5.18                5700              3
## 5 5     1            5           1      5.34                5760              3
## 6 6     1            6           2      5.77                5720              4
##   Temperature_Celcius Inversion_base_height.IBH Pressure_gradient.Psi.ft
## 1                  30                      5000                      -15
## 2                  38                      1601                      -14
## 3                  40                      2693                      -25
## 4                  45                       590                      -24
## 5                  54                      1450                       25
## 6                  35                      1568                       15
##   Inversion_temperature.ivC Visibility_pAerosol
## 1                     30.56                 200
## 2                     46.94                 300
## 3                     47.66                 250
## 4                     55.04                 100
## 5                     57.02                  60
## 6                     53.78                  60
```

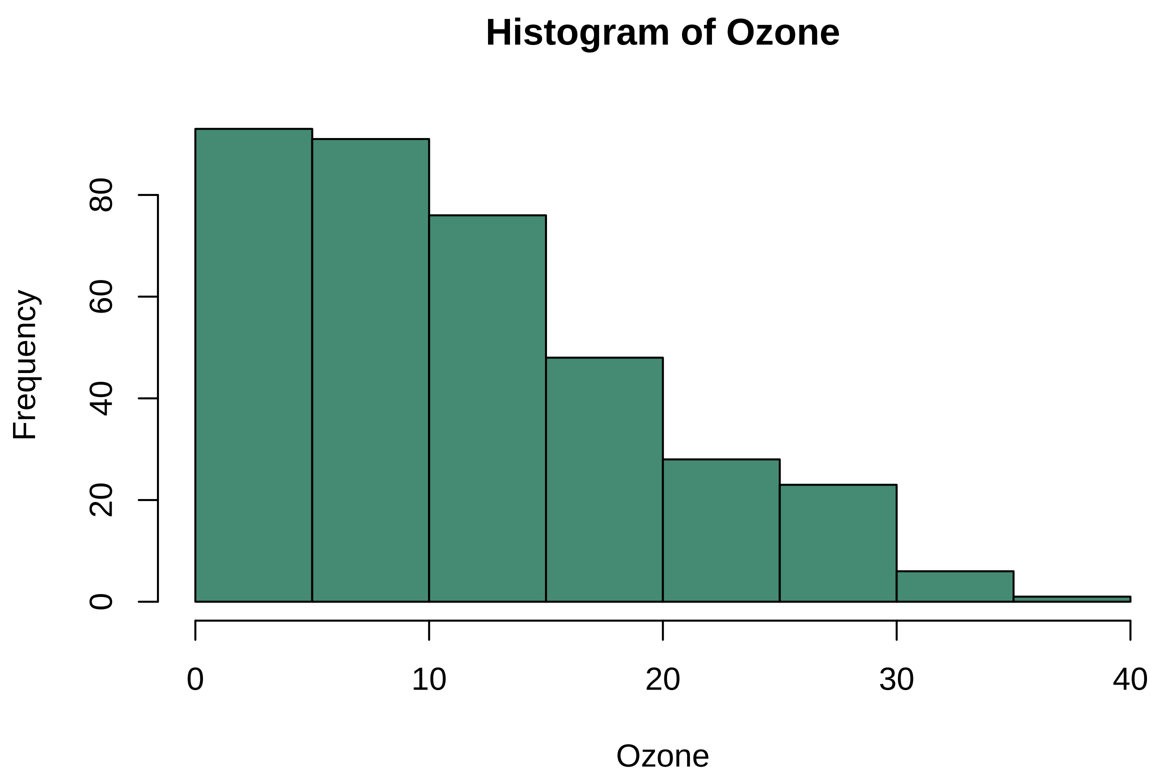
```
dataAP3 = dataAP3[,-c(1)]
head(dataAP3)
```

```
##   Month Day_of_month Day_of_week ozone_ppm pressure_height.hPA Wind_speed.mph
## 1     1            1           4      3.01                5480              8
## 2     1            2           5      3.20                5660              6
## 3     1            3           6      2.70                5710              4
## 4     1            4           7      5.18                5700              3
## 5     1            5           1      5.34                5760              3
## 6     1            6           2      5.77                5720              4
##   Temperature_Celcius Inversion_base_height.IBH Pressure_gradient.Psi.ft
## 1                  30                      5000                      -15
## 2                  38                      1601                      -14
## 3                  40                      2693                      -25
## 4                  45                       590                      -24
## 5                  54                      1450                       25
## 6                  35                      1568                       15
##   Inversion_temperature.ivC Visibility_pAerosol
## 1                     30.56                 200
```

```
## 2                    46.94                300
## 3                    47.66                250
## 4                    55.04                100
## 5                    57.02                 60
## 6                    53.78                 60
```

We can plot histogram to see whether the data is normal or not.

```r
hist(dataAP3$ozone_ppm, main = 'Histogram of Ozone', col = 'aquamarine4',
     xlab = 'Ozone')
```

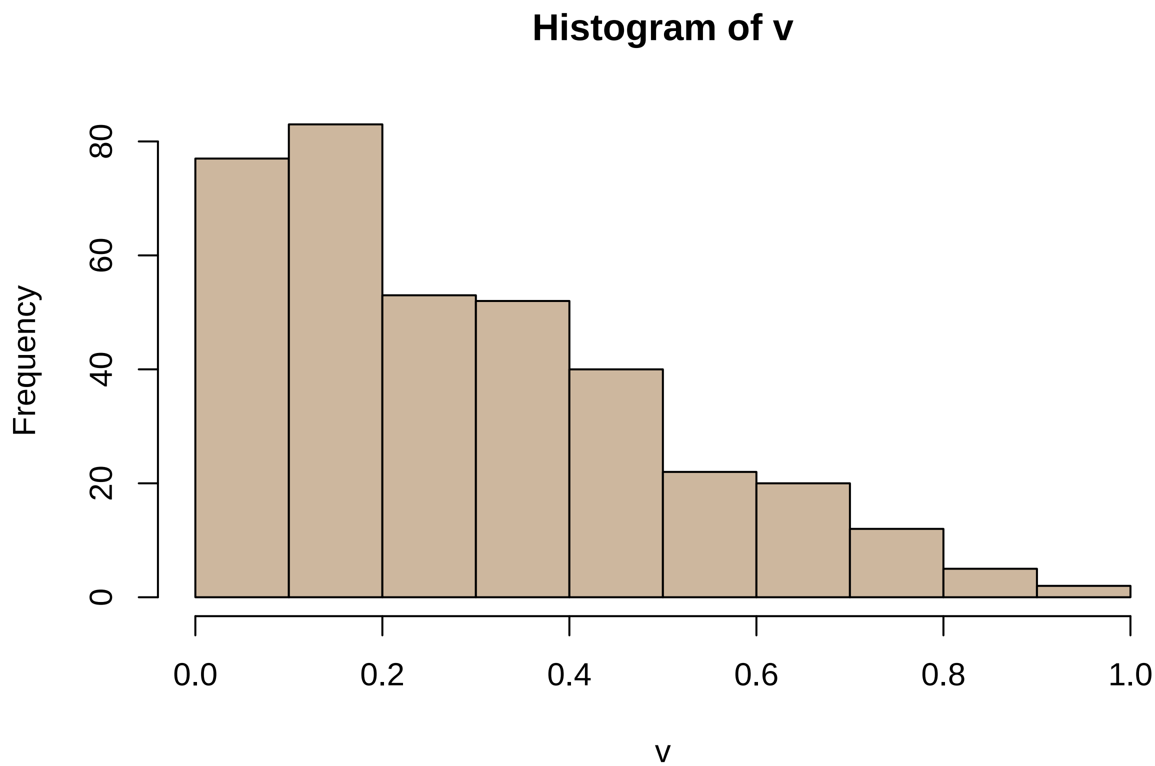


now that we've seen that the data is not normal, we can try to normalize it using min-max technique.

```r
min_ozone = min(dataAP3$ozone_ppm)
max_ozone = max(dataAP3$ozone_ppm)
v = ((dataAP3$ozone_ppm - min_ozone) * (1 - 0)) / (max_ozone-min_ozone)
head(v)
```

```
## [1] 0.06146001 0.06655931 0.05314010 0.11969941 0.12399356 0.13553408
```

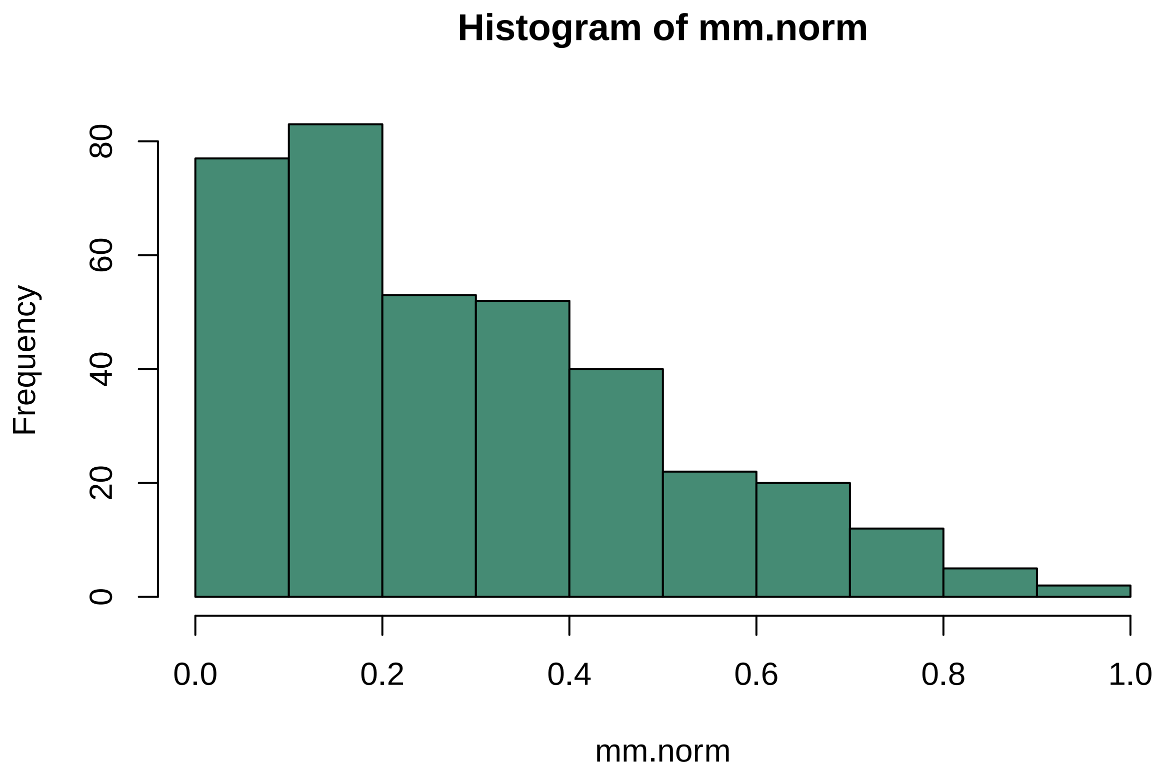
```r
hist(v, col = 'bisque3')
```



Try to make it in a function

```
mm = function (x, n.min, n.max) {
  min = min(x)
  max = max(x)
  mm.norm = ((x-min) * (n.max-n.min)/(max-min)) + n.min
  hist(mm.norm, col = 'aquamarine4')
  return(mm.norm)
}

new.dataAP3 = mm(dataAP3$ozone_ppm, 0, 1)
```

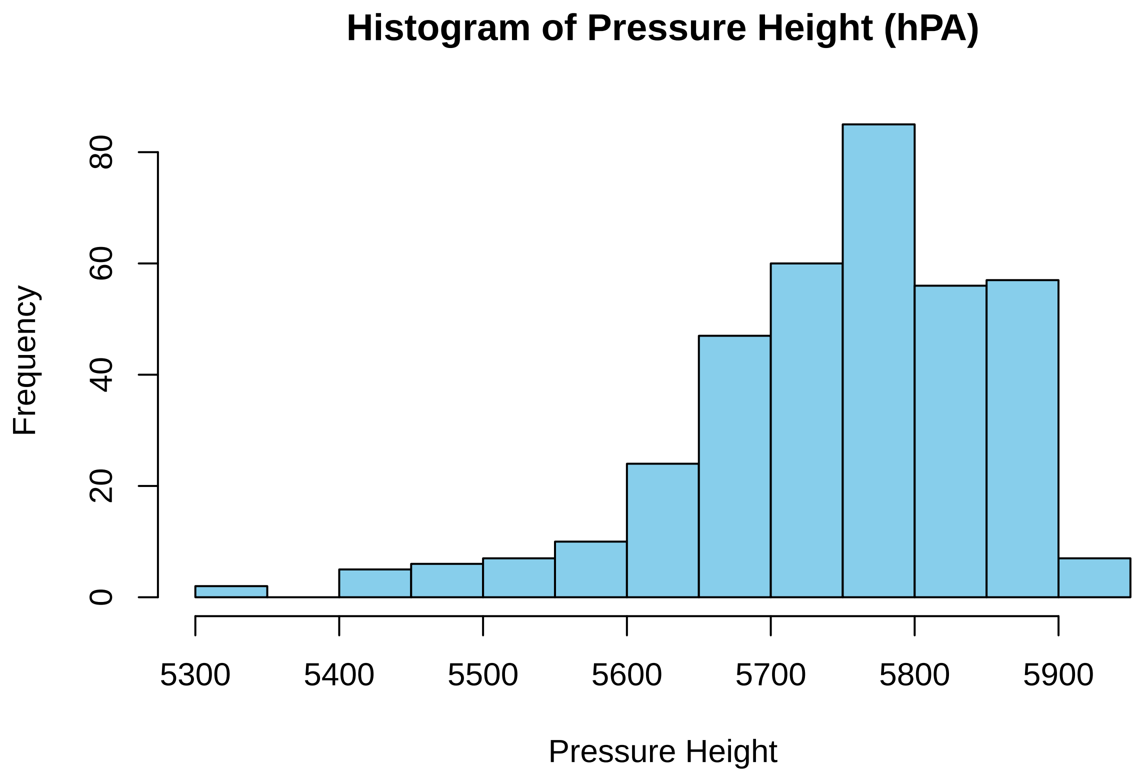


Z-score Normalization

Zero-mean normalization

$$Z = \frac{X - \mu_x}{\sigma_x}$$

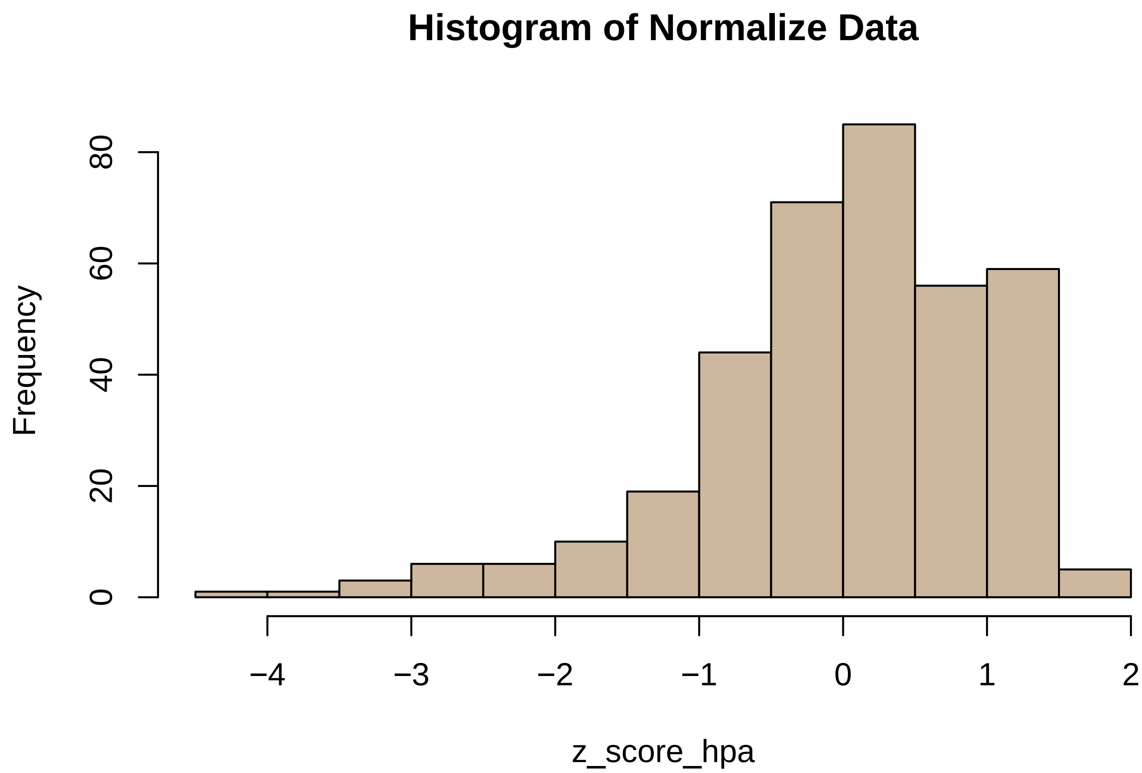
```
hist(dataAP3$pressure_height.hPA, col = 'skyblue', main = 'Histogram of Pressure Height (hPA)', xlab =
```




```r
mean_hpa = mean(dataAP3$pressure_height.hPA)
sd_hpa = sd(dataAP3$pressure_height.hPA)
z_score_hpa = (dataAP3$pressure_height.hPA - mean_hpa) / sd_hpa
head(z_score_hpa)
```

```
## [1] -2.58185122 -0.87803114 -0.40474779 -0.49940446  0.06853557 -0.31009112
```

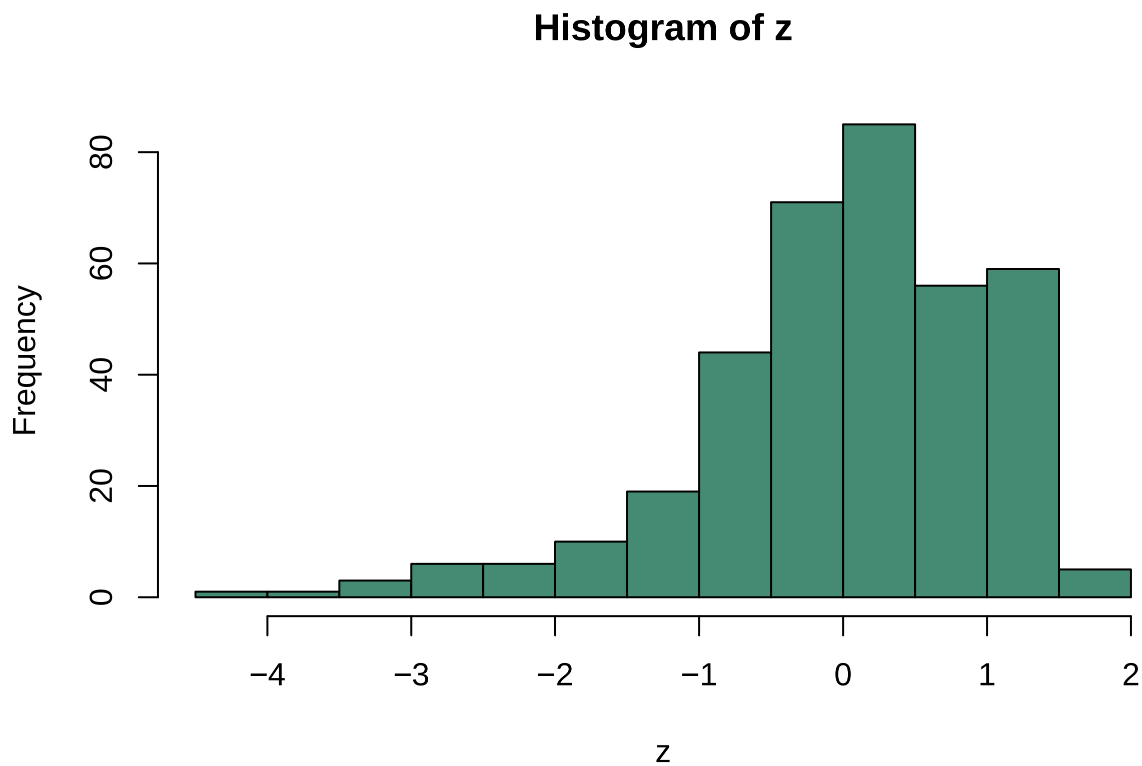
```r
hist(z_score_hpa, col = 'bisque3', main = 'Histogram of Normalize Data')
```



Try to make in in a function

```
z = function (x) {
  mean = mean(x)
  sd = sd(x)
  z = (x - mean)/sd
  hist(z, col = 'aquamarine4')
  return(z)
}

z(dataAP3$pressure_height.hPA)
```




```
##   [1] -2.58185122 -0.87803114 -0.40474779 -0.49940446  0.06853557 -0.31009112
##   [7]  0.35250558  0.35250558 -0.49940446 -0.49940446  0.16319224 -0.31009112
##  [13]  0.06853557  0.25784891  0.73113226  1.10975895  0.82578894  0.25784891
##  [19] -0.68871780 -0.31009112  0.06853557 -0.21543444 -0.49940446 -0.97268781
##  [25] -0.68871780  0.25784891  0.63647559  0.73113226  0.54181892  0.35250558
##  [31]  0.44716225  0.63647559  0.16319224 -0.78337447 -1.54062784 -3.24444791
##  [37] -3.81238794 -2.58185122 -1.44597117 -2.48719455 -1.82459785 -0.49940446
##  [43] -0.68871780 -0.49940446 -0.97268781 -1.35131450 -0.21543444  0.16319224
##  [49]  0.16319224 -0.49940446 -0.59406113 -0.49940446 -0.21543444 -0.59406113
##  [55] -1.06734448 -0.31009112 -0.12077777 -0.12077777 -0.12077777 -0.78337447
##  [61] -1.91925452 -2.67650789 -4.09635795 -3.14979124 -2.10856786 -1.44597117
##  [67] -0.87803114 -1.63528451 -2.29788121 -2.10856786 -1.25665783 -0.59406113
##  [73]  0.06853557 -0.12077777  0.25784891  0.35250558  0.06853557 -0.49940446
##  [79] -0.68871780 -0.31009112 -0.31009112 -0.21543444 -0.40474779 -0.02612110
##  [85] -0.31009112 -0.40474779 -1.44597117 -1.16200115 -0.59406113 -0.21543444
##  [91] -0.40474779 -1.35131450 -0.68871780 -1.25665783 -3.14979124 -2.01391119
##  [97] -1.54062784 -0.59406113 -1.91925452 -1.25665783 -1.16200115 -1.63528451
## [103] -1.82459785 -2.96047790 -2.58185122 -1.25665783 -2.86582123 -0.87803114
## [109] -0.68871780  0.06853557  0.35250558 -0.31009112 -0.87803114 -0.40474779
## [115]  0.25784891 -0.02612110 -0.49940446 -1.25665783 -0.97268781 -0.21543444
## [121]  0.54181892  0.35250558 -0.12077777 -0.87803114 -0.59406113 -0.02612110
## [127] -0.68871780 -0.97268781 -0.21543444 -0.21543444  0.06853557  0.73113226
## [133]  1.20441562  1.29907229  0.92044561  0.63647559  0.73113226  0.44716225
## [139] -0.12077777 -0.40474779 -0.31009112 -0.40474779 -0.12077777 -0.31009112
## [145] -0.59406113 -0.21543444  0.25784891  0.35250558 -0.02612110 -0.68871780
## [151] -0.31009112  0.16319224  0.44716225  0.25784891 -0.12077777 -0.21543444
```

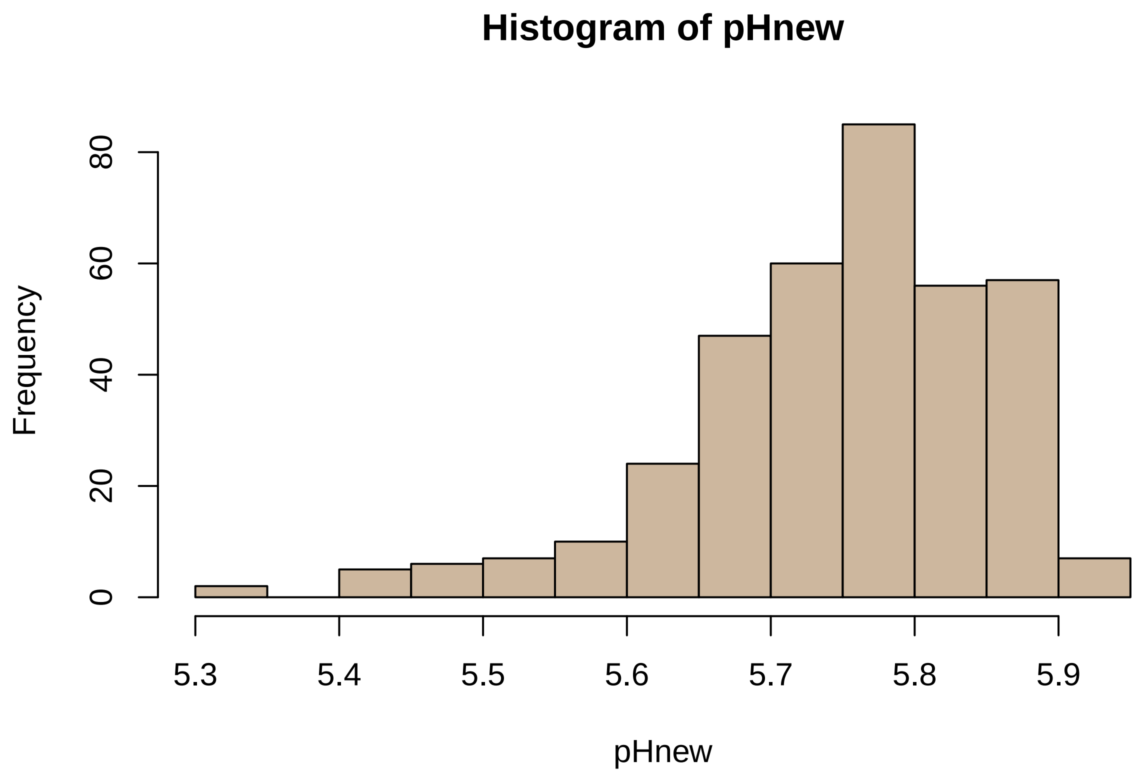
```
## [157]  0.06853557  0.16319224 -0.59406113 -0.97268781 -1.35131450 -1.72994118
## [163] -0.59406113  0.06853557  0.54181892  0.73113226  1.20441562  1.01510228
## [169]  0.73113226  1.10975895  1.01510228  0.44716225  0.44716225  0.16319224
## [175]  1.01510228  1.10975895  1.10975895  1.01510228  1.20441562  1.10975895
## [181]  1.01510228  0.73113226  0.63647559  0.63647559  1.01510228  1.10975895
## [187]  1.29907229  1.39372896  1.29907229  1.29907229  1.48838563  1.39372896
## [193]  1.01510228  0.73113226  0.92044561  0.73113226  0.44716225  0.54181892
## [199]  0.73113226  0.73113226  0.82578894  1.10975895  1.10975895  0.92044561
## [205]  1.01510228  1.39372896  1.29907229  1.20441562  1.29907229  1.10975895
## [211]  1.20441562 -0.21543444  0.63647559  0.25784891  0.16319224  0.35250558
## [217]  0.63647559  0.82578894  0.44716225  0.73113226  0.44716225  0.82578894
## [223]  0.44716225  1.48838563  1.29907229  1.10975895  0.25784891 -0.21543444
## [229]  0.25784891 -0.02612110  0.06853557 -0.21543444 -0.21543444  0.35250558
## [235] -0.02612110  1.20441562  1.29907229  1.20441562  1.10975895  1.39372896
## [241]  1.58304230  1.67769897  1.86701232  1.86701232  1.39372896  1.29907229
## [247]  1.01510228  0.82578894  0.44716225  0.06853557  0.54181892  0.92044561
## [253]  1.01510228  1.10975895  0.06853557  1.01510228  0.73113226  0.82578894
## [259]  0.44716225 -0.68871780  0.35250558  0.73113226  0.54181892  0.16319224
## [265]  0.25784891  0.44716225  0.16319224  0.44716225  0.25784891  0.35250558
## [271]  0.16319224 -0.02612110 -1.06734448 -1.06734448 -0.97268781 -0.40474779
## [277]  0.06853557  0.82578894  1.20441562  1.29907229  1.29907229  1.29907229
## [283]  1.29907229  0.92044561  0.73113226  0.73113226  1.01510228  0.73113226
## [289]  0.44716225  0.73113226  0.35250558 -0.21543444  0.25784891 -0.12077777
## [295] -0.40474779 -0.59406113 -0.78337447  0.06853557  0.63647559  0.35250558
## [301]  0.06853557  0.44716225  0.54181892 -0.02612110  0.82578894  1.01510228
## [307]  1.10975895  1.58304230  1.39372896  1.01510228  0.82578894  0.82578894
## [313]  0.92044561  0.54181892  0.16319224 -0.40474779 -2.39253788 -0.87803114
## [319] -0.49940446  0.54181892  1.01510228  1.39372896  0.92044561  0.25784891
## [325]  0.35250558  0.25784891  0.16319224 -0.02612110  0.25784891  0.35250558
## [331] -0.02612110 -0.78337447  0.06853557  0.16319224  0.54181892  0.54181892
## [337]  1.10975895  0.73113226  0.06853557 -0.68871780  0.25784891  0.54181892
## [343]  0.06853557 -0.68871780 -0.02612110  0.35250558  0.16319224 -0.02612110
## [349] -0.31009112  0.06853557  0.25784891 -0.87803114 -1.35131450 -1.06734448
## [355] -0.68871780 -0.97268781 -0.40474779 -0.68871780 -1.16200115  0.16319224
## [361]  0.44716225 -0.21543444 -0.59406113 -0.97268781 -1.91925452 -0.68871780
```

Decimal Scaling

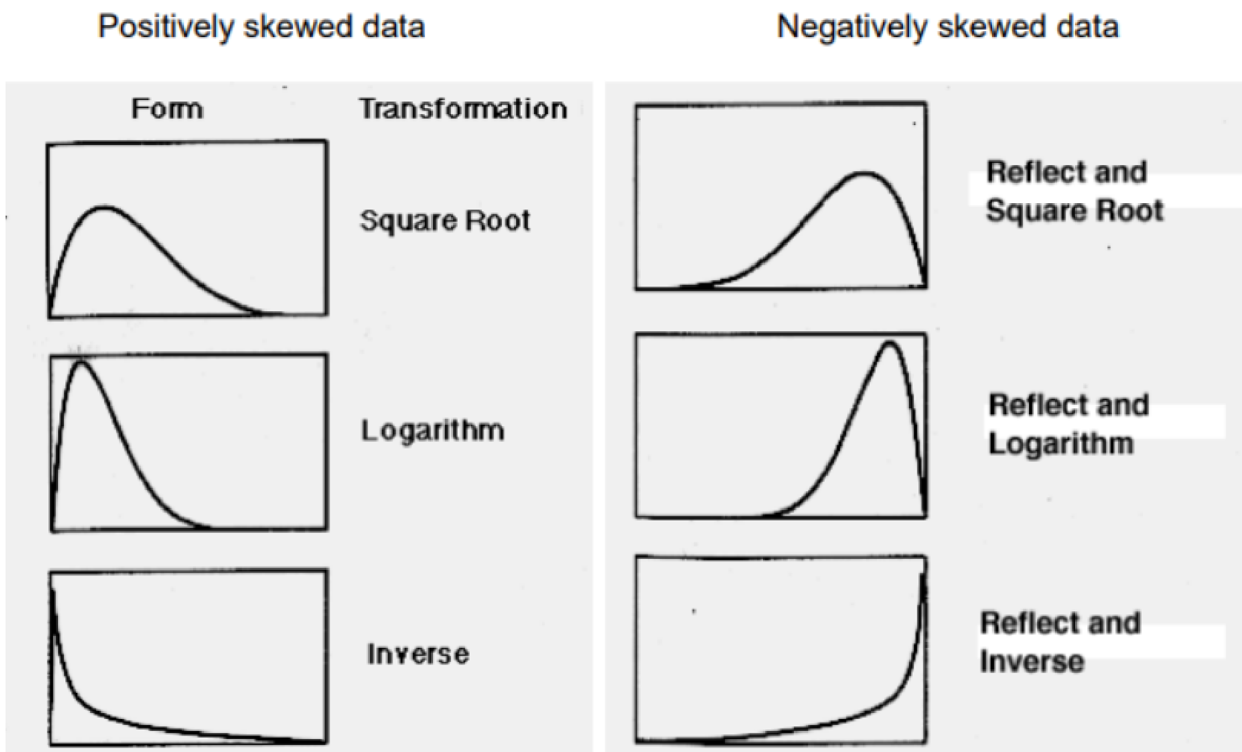
```r
pHnew = dataAP3$pressure_height.hPA/1000
head(pHnew)
```

```
## [1] 5.48 5.66 5.71 5.70 5.76 5.72
```

```r
hist(pHnew, col = 'bisque3')
```


Histogram of pHnew
Frequency
0
20
40
60
80
5.3
5.4
5.5
5.6
5.7
5.8
5.9
pHnew

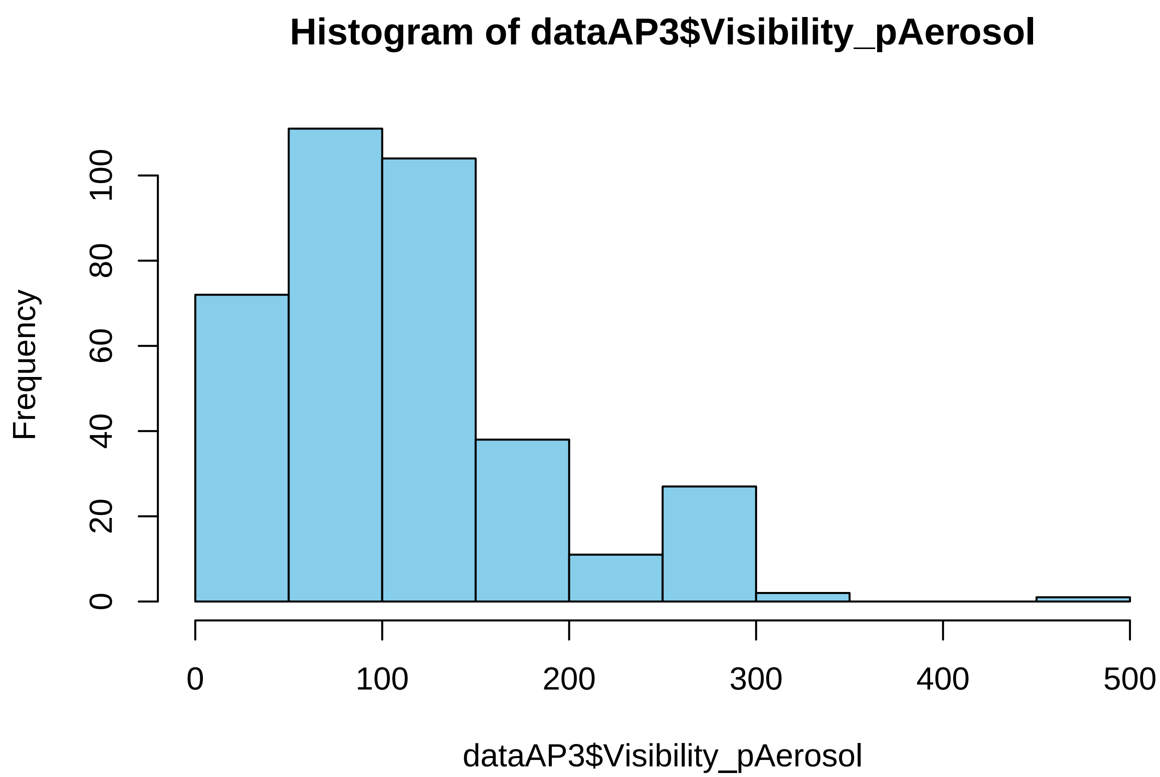
Normaling Data Distribution



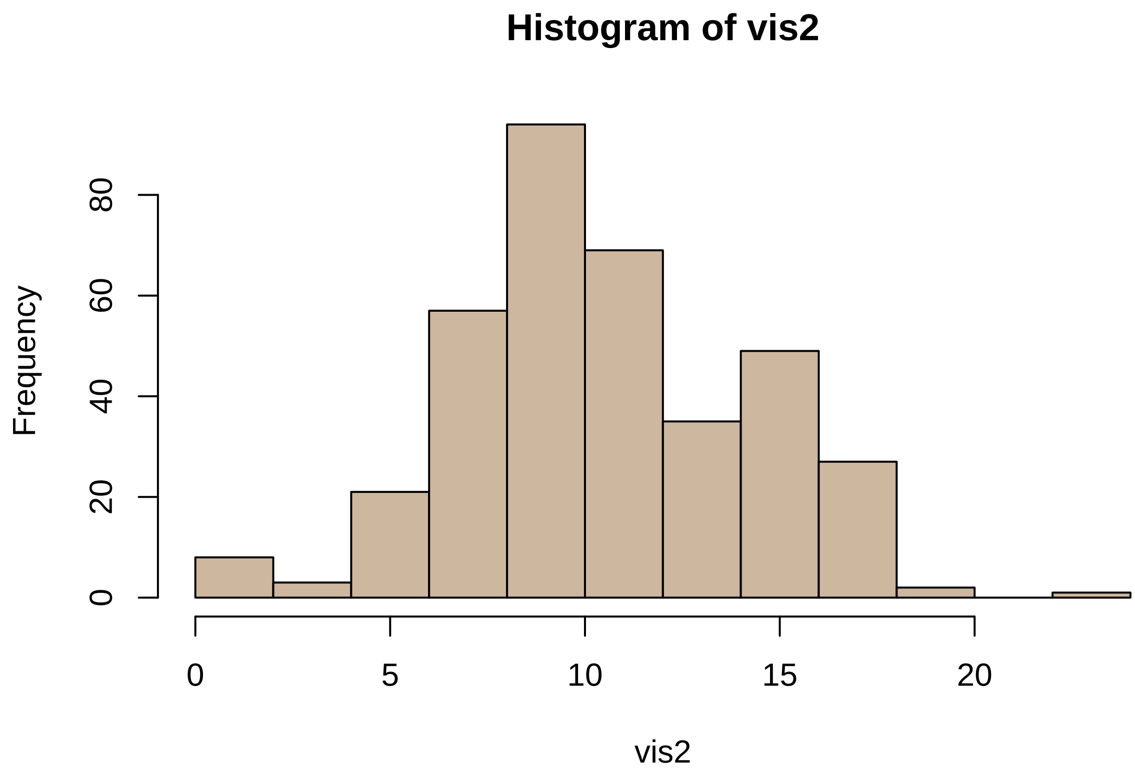

```r
dataAP3$Visibility_pAerosol
```

```
##   [1] 200 300 250 100  60  60 100 250 120 120 120 150  40 200 250 200 200 150
##  [19]  10 140 250 200 150 140  50   0  70 150 150 120  40 120   6  30 100 200
##  [37]  60 350 250 350 300 300 300 200 100 250 200 200  40   2 300 300 300 300
##  [55] 300 150 150  80  40  40  80 300 200 500 140 140 140 100 140 200 120 300
##  [73] 300 150   2  50  70  17 140 140 300 200 250  80  60 100 150 150 200 100
##  [91] 300 120 100 200 200 200 300 300 250 120 140 200 140  80 300 100 300 200
## [109] 120 100 120  60 120 100 100  27  40 140 150 100 100 120 150 100 120  80
## [127] 120 140 120  70  80  70  40  20  17  40  50  50  70  80 120 120 100 120
## [145] 120 200 120  40  70 100 120 100 120  70  80 100 100 120 120 120 150 140
## [163] 140 140 140  60  30  17  80  60 100 120 150 120 140 140 120 120  80 140
## [181] 140 150 120 120 140 100  50  40 100  80 100  60  50  70  80  80  80  90
## [199] 120 120 100  60  40  50  40  70  80  80  80  80  80 100 120 150 200 150
## [217] 150 150 150 100 100 100  30  80  70  60 150 200 200 200 250 300  70 300
## [235] 150 300  30 100 100  17  20   4  70  30  70  60  40  50  70 140 100 120
## [253] 100  70 150  50  70  40  70 120 140 140 100  50  70  40  40 100 120 120
## [271] 140 120  70 150 200 200 200  70  40  50  17  80 250 200   2  20   7  30
## [289]  50  70  17  80  50  60  60  80  50  50  40  40 300 200 150 100 100  60
## [307] 150 150 200 300 120  30 100  50  20 200 120 300 200  70 140 150 200   4
## [325]  40  30  30   2   0  30  60 150 100 250 150 200 200 200  80  60 300 200
## [343] 300  50  40  70 150 150  70 200 120 150 150  60  70 150 300 100  70  40
## [361] 140 200  70  40 100  70
```

```r
hist(dataAP3$Visibility_pAerosol, col = 'skyblue')
```



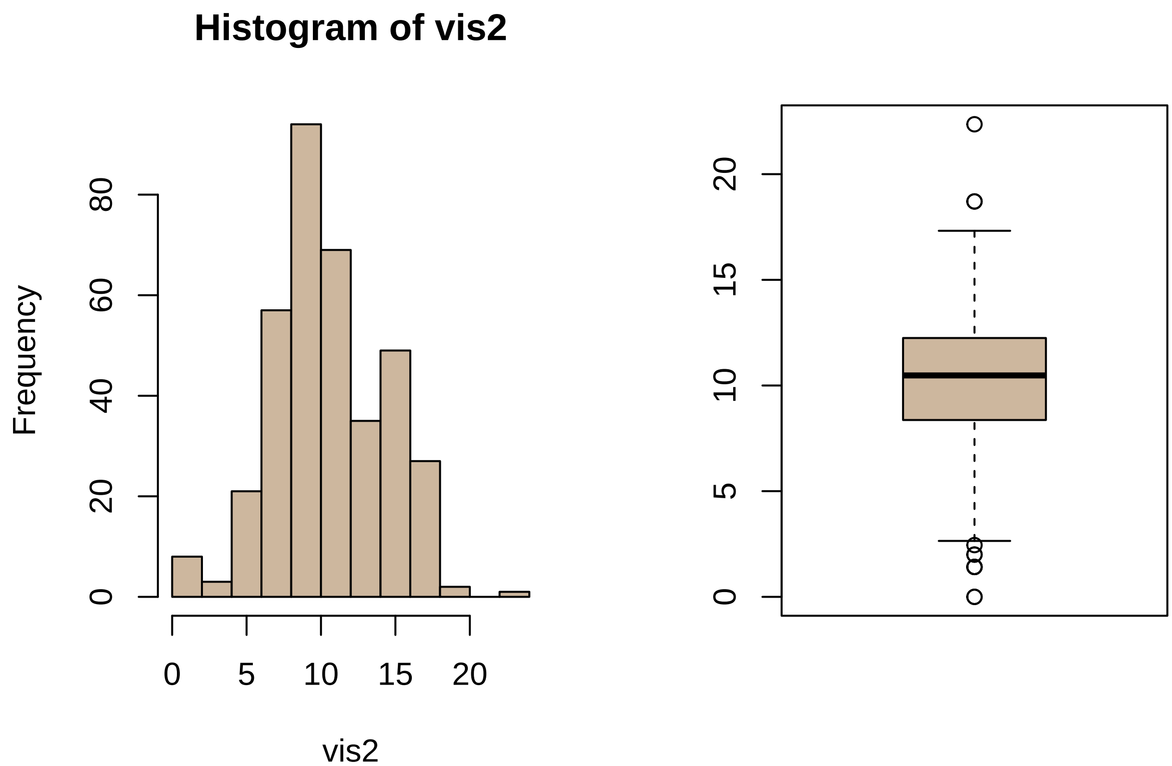

```r
vis2 = sqrt(dataAP3$Visibility_pAerosol)
hist(vis2, col = 'bisque3')
```



Assessing Normality

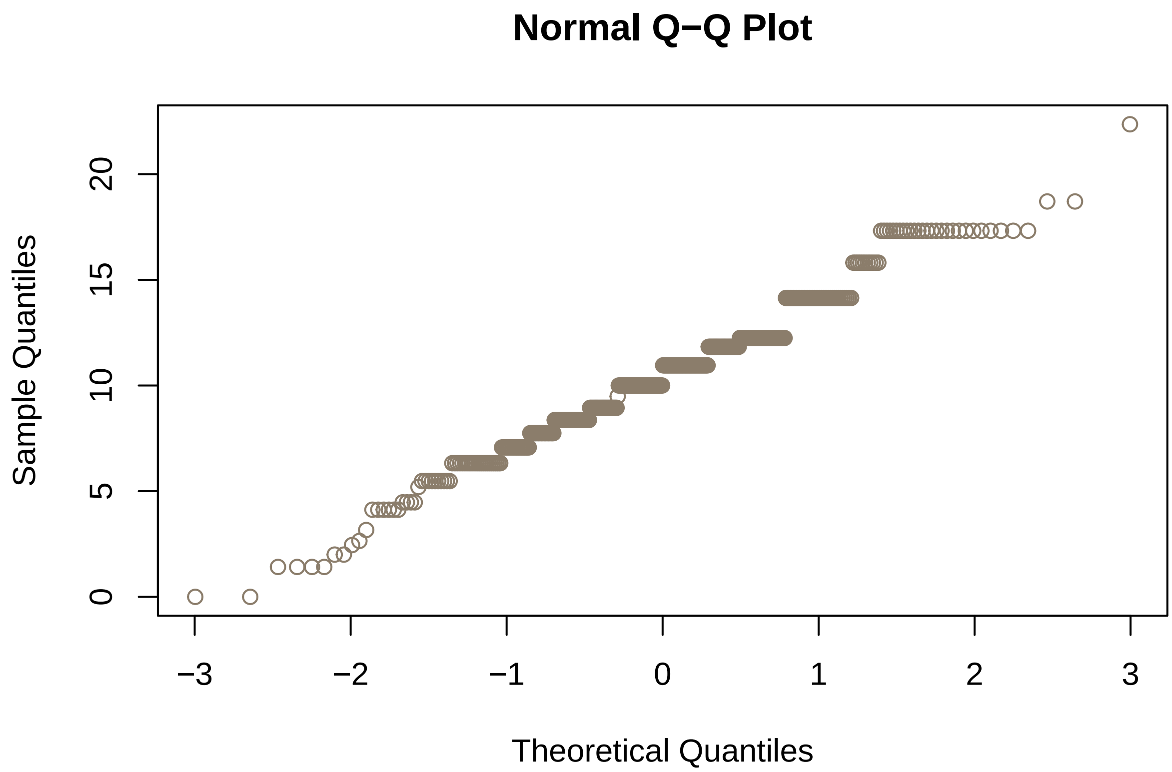
Histogram & Boxplot

```
par(mfrow=c(1,2))
hist(vis2, col = 'bisque3')
boxplot(vis2, col = 'bisque3')
```



Normal Quantile Plot (Q-Q Plot)

```
qqnorm(vis2, col = 'bisque4')
```



Goodnes-of-fit test

Kolmogorov-Smirnov

$H_0 =$ The sample data follows a normal distribution$H_1 =$ The sample data does not follow a normal distribution

```r
ks.test(vis2, 'pnorm', mean = mean(vis2), sd = sd(vis2))
```

```
## Warning in ks.test.default(vis2, "pnorm", mean = mean(vis2), sd = sd(vis2)):
## ties should not be present for the one-sample Kolmogorov-Smirnov test
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  vis2
## D = 0.099679, p-value = 0.001388
## alternative hypothesis: two-sided
```

Shapiro-Wilk

$H_0 =$ The sample data is drawn from a normal distribution$H_1 =$ The sample data is not normally distribution

```r
shapiro.test(vis2)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  vis2
## W = 0.983, p-value = 0.0002616
```

Anderson-Darling

$H_0 =$ The sample data follows a normal distribution$H_1 =$ The sample data does not follow a normal distribution

```r
library(nortest)
ad.test(vis2)
```

```
##
##  Anderson-Darling normality test
##
## data:  vis2
## A = 2.177, p-value = 1.554e-05
```

Discretization

Unsupervised Learning

This method need the knowledge of the industry and can be made manually for example like the financial class (B40, M40, T20)

```r
library(infotheo)
data("USArrests")
attach(USArrests)
USArrests
```

2 category

```
##                Murder Assault UrbanPop Rape
## Alabama          13.2     236       58 21.2
## Alaska           10.0     263       48 44.5
## Arizona           8.1     294       80 31.0
## Arkansas          8.8     190       50 19.5
## California        9.0     276       91 40.6
## Colorado          7.9     204       78 38.7
## Connecticut       3.3     110       77 11.1
## Delaware          5.9     238       72 15.8
## Florida          15.4     335       80 31.9
## Georgia          17.4     211       60 25.8
## Hawaii            5.3      46       83 20.2
## Idaho             2.6     120       54 14.2
```

```
## Illinois         10.4     249       83 24.0
## Indiana           7.2     113       65 21.0
## Iowa              2.2      56       57 11.3
## Kansas            6.0     115       66 18.0
## Kentucky          9.7     109       52 16.3
## Louisiana        15.4     249       66 22.2
## Maine             2.1      83       51  7.8
## Maryland         11.3     300       67 27.8
## Massachusetts     4.4     149       85 16.3
## Michigan         12.1     255       74 35.1
## Minnesota         2.7      72       66 14.9
## Mississippi      16.1     259       44 17.1
## Missouri          9.0     178       70 28.2
## Montana           6.0     109       53 16.4
## Nebraska          4.3     102       62 16.5
## Nevada           12.2     252       81 46.0
## New Hampshire     2.1      57       56  9.5
## New Jersey        7.4     159       89 18.8
## New Mexico       11.4     285       70 32.1
## New York         11.1     254       86 26.1
## North Carolina   13.0     337       45 16.1
## North Dakota      0.8      45       44  7.3
## Ohio              7.3     120       75 21.4
## Oklahoma          6.6     151       68 20.0
## Oregon            4.9     159       67 29.3
## Pennsylvania      6.3     106       72 14.9
## Rhode Island      3.4     174       87  8.3
## South Carolina   14.4     279       48 22.5
## South Dakota      3.8      86       45 12.8
## Tennessee        13.2     188       59 26.9
## Texas            12.7     201       80 25.5
## Utah              3.2     120       80 22.9
## Vermont           2.2      48       32 11.2
## Virginia          8.5     156       63 20.7
## Washington        4.0     145       73 26.2
## West Virginia     5.7      81       39  9.3
## Wisconsin         2.6      53       66 10.8
## Wyoming           6.8     161       60 15.6
```

```r
cutoff = 10 # Need domain explanation
status_m = ifelse(Murder<10,'Low Risk','High Risk')
head(status_m)
```

```
## [1] "High Risk" "High Risk" "Low Risk"  "Low Risk"  "Low Risk"  "Low Risk"
```

```r
# alternative way
cut(USArrests$Murder,
     breaks = c(min(USArrests$Murder), 10, max(USArrests$Murder)),
     labels = c('Low Risk', 'High Risk'))
```

```
##  [1] High Risk Low Risk  Low Risk  Low Risk  Low Risk  Low Risk  Low Risk
##  [8] Low Risk  High Risk High Risk Low Risk  Low Risk  High Risk Low Risk
## [15] Low Risk  Low Risk  Low Risk  High Risk Low Risk  High Risk Low Risk
```

```
## [22] High Risk Low Risk  High Risk Low Risk  Low Risk  Low Risk  High Risk
## [29] Low Risk  Low Risk  High Risk High Risk High Risk <NA>      Low Risk
## [36] Low Risk  Low Risk  Low Risk  Low Risk  High Risk Low Risk  High Risk
## [43] High Risk Low Risk  Low Risk  Low Risk  Low Risk  Low Risk  Low Risk
## [50] Low Risk
## Levels: Low Risk High Risk
```

```r
library(car)
```

More than 2 category

```
## Warning: package 'car' was built under R version 4.4.2
```

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

```
## Warning: package 'carData' was built under R version 4.4.2
```

```r
status_den = Recode(UrbanPop, "0:50 = 'Low Density';
                               51:70 = 'Moderate Density';
                               else = 'High Density'")
head(status_den)
```

```
## [1] "Moderate Density" "Low Density"      "High Density"     "Low Density"
## [5] "High Density"     "High Density"
```

```r
# alternative way
```

```r
assault_status = discretize(Assault, 'equalwidth', 4)
unique(assault_status)
```

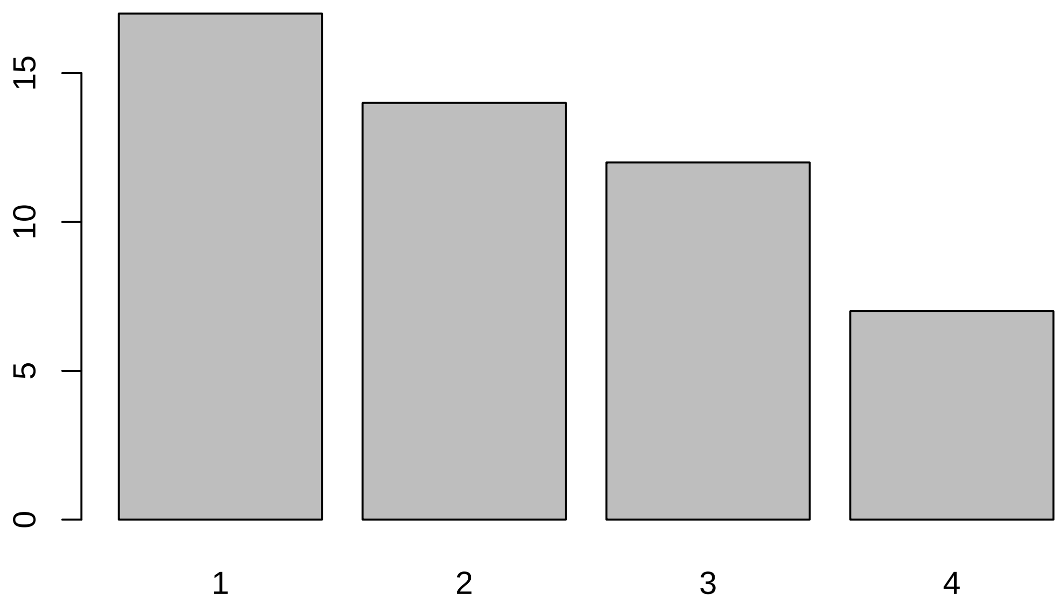
Equal-width

```
##   X
## 1 3
## 3 4
## 4 2
## 7 1
```

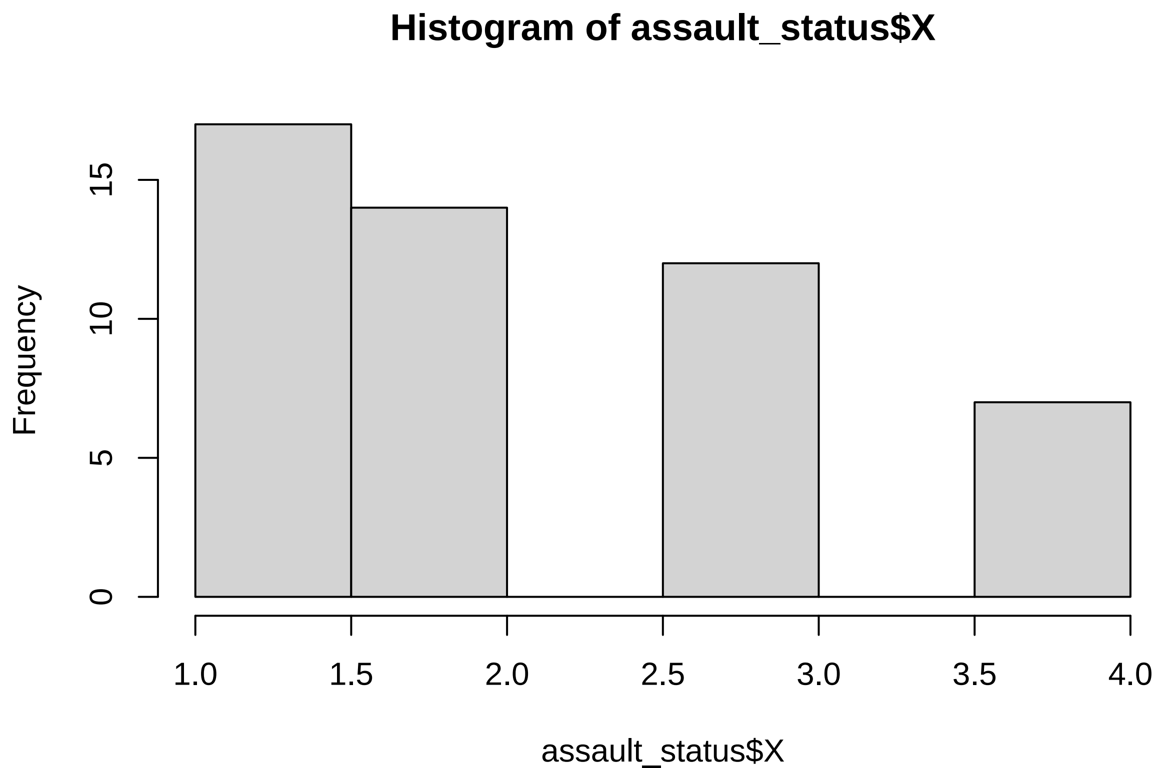
```r
head(assault_status)
```

```
##   X
## 1 3
## 2 3
## 3 4
## 4 2
## 5 4
## 6 3
```

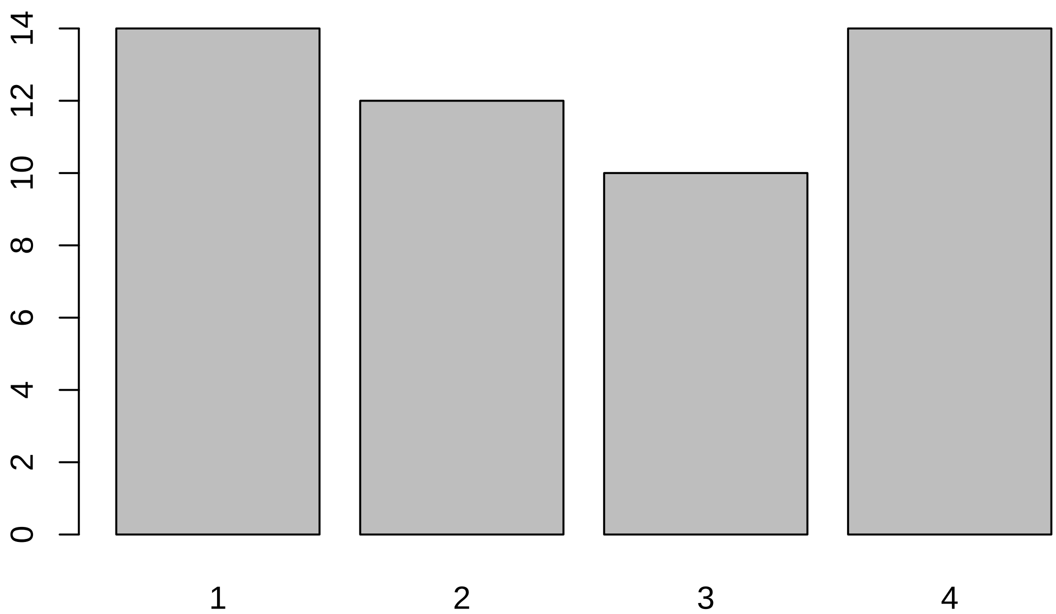
```r
barplot(table(assault_status$X))
```




```r
hist(assault_status$X)
```




```
barplot(table(discretize(Assault, 'equalfreq', 4)$X))
```



Equal Frequency

Supervised Learning

```r
library(discretization)
data(iris)
iris2 = chi2(iris, alp=0.05)$Disc.data
head(iris2)
```

Chi2

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1            1           3            1           1  setosa
## 2            1           2            1           1  setosa
## 3            1           2            1           1  setosa
## 4            1           2            1           1  setosa
## 5            1           3            1           1  setosa
## 6            1           3            1           1  setosa
```

ChiMerge Algorithm

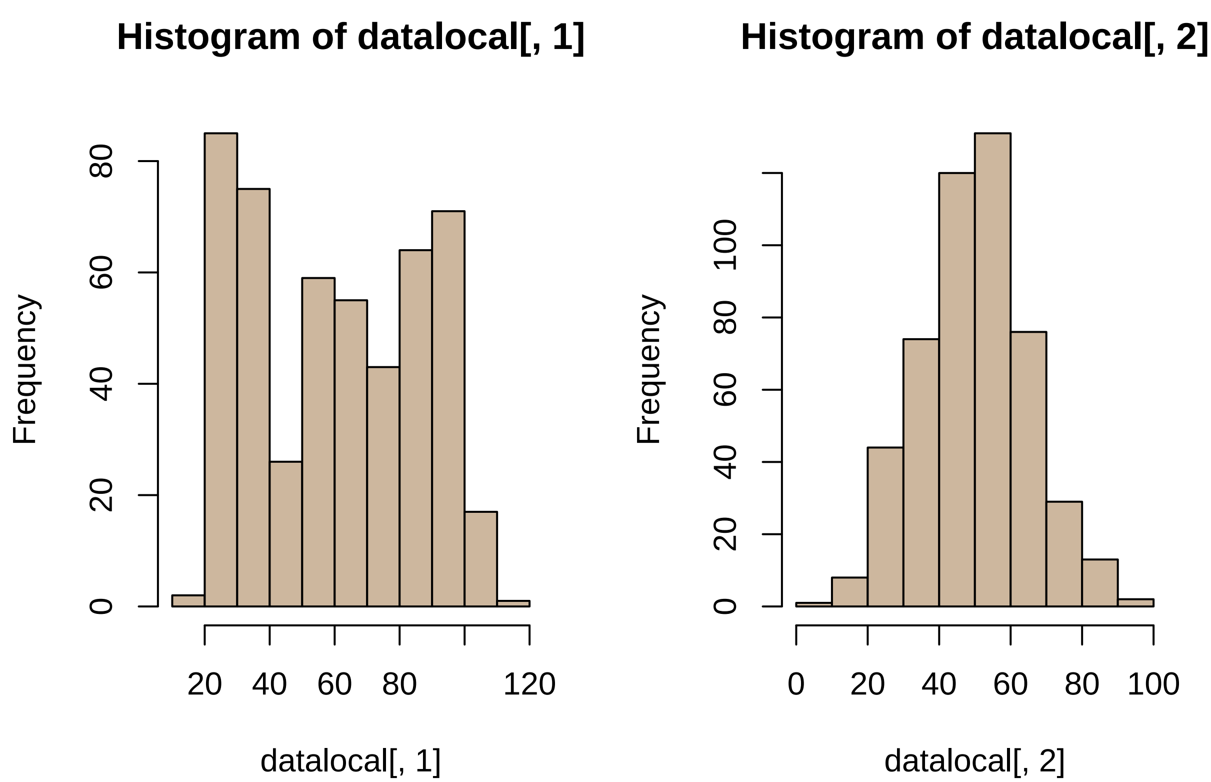
Top-down algorithm

Minimum Description Length Principle (MDLP) Add : kmeans

```
datalocal = read.csv('./Data/dataLocal.csv', sep = ';')
head(datalocal)
```

```
##   dataL1 dataL2
## 1   27.2   55.5
## 2   28.8   58.3
## 3   37.8   41.0
## 4   30.4   35.1
## 5   30.6   65.4
## 6   38.6   61.3
```

```
par(mfrow=c(1,2))
hist(datalocal[,1], col = 'bisque3')
hist(datalocal[,2], col = 'bisque3')
```




```
model = kmeans(datalocal[,1], 3)
km = as.factor(model$cluster)
hist(model$cluster, col = km)
```

Attribute formation

- Linear transformation
- Encoding
  - One-hot encoding
  - Ordinal encoding
  - Target encoding
  - Frequency encoding
- Rank transformation
- Box-cox transformation
- Polynomial approximation transformation
- Non-polynomial approximation transformation
- Wavelet transformation

Smoothing