Univarite Statistics

In general the data you deal with in industry is too big to go through each and every observation. To understand the data you need to come up with a way to summarise this so that you can quickly go through the summary and understand your data faster.

In this module we are going to learn what kind of summaries to consider to understand your data in a wholesome manner. There are three facets of the data that we need to summarize in order to capture all aspects of the data:

- Central Tendency
- Variability
- Shape

We can create these summaries in two ways:

- Numeric Summary
- Data Visualisation

We'll learn how to summarise each aspect of the data numerically first. Also here onwards in this module , when we say data, we are talking about just one variable. [**Uni**variate Statistics]

Central Tendency

Central tendency is average behaviour of the data. For example if somebody asks a student , "what is the age of a typical student in their class?", best bet is the average age of students in the class.

Central tendency is the value of your data which is most expected outcome. There are three measures for central tendency:

- Mean
- Median
- Mode

Mean

Mean is defined as follows:

$$\bar{x} = \frac{\sum\limits_{i=1}^{n} x_i}{n}$$

This is simply average of the values. Consider the average of these numbers: {1,2,3,4,5,6,7} . It is 4. Now if I include another value to this set of values say 10000, then the average becomes 1257, which is no where close to majority of the data {1,2,3,4,5,6,7,10000}.

This tells us that mean is not a good measure of central tendency if data contains extreme values or outliers.

Median

Median is defined as the middle most value when your data is sorted. For example is your data was {1,10,8,7,9,12,3}, you'd sort it : {1,3,7,8,9,10,12} , your data has 7 elements, the middle most term will be $\frac{(7+1)}{2}^{th}$ term which is 8.

If your data has even number of elements then you'll have two middle most values. For example lets consider this data : {1,3,7,8,9,10,12,14000}. two middle most terms are : 8 & 9, median is taken as average of these two middle most values. which is 8.5

You might have noticed by now that median doesnt not depend on the values of elements , only on their order in the data which makes it not so sensitive to extreme values present in the data.

Mode

Mode is simply defined as the value which occurs most frequently in the data. It is mostly used for categorical variables because numerical summaries do not make sense in context of categorical variables.

A dataset can have multiple modes , since highest frequency can be equal for many categories of the categorical variable. As opposed to this mean and median for a dataset take a single value.

Variability

datasets having same average do not neccessarily behave in a same fashion. They might differ in terms of spread. Here is an example, both the data sets have their average as 10.

{7,8,9,10,11,12,13} : 10

{-100,-50,10,90,100} : 10

you can see that the second data is much more spread out in comparison to the first one, inspite of them having same mean. There are many ways in which this spread can be measured.

- Range
- Mean Absolute Deviation (MAD)
- Standard Deviation / Variance
- Inter Quartile Range

Range

Range is simply defined as difference between minimum and maximum values in the data. It's a very primitive measure and is rarely used. You can observe that since it relies on extreme values in your data to start with, it is very sensitive to presence of extreme values in your data

Mean Absolute Deviation [MAD]

Spread is defined as deviation of individual values from the mean of the data. If you simply try to add these deviations and take average, positive and negative deviations will cancel each other out . Average absolute values of these deviation is one way to avoid effect of signs in the deviations. MAD is defined as :

$$MAD \ = \frac{\sum_{i=1}^{n} |x_i - \bar{x}|}{n}$$

MAD is rarely used in practice because it is very tedious to manipulate it algebraically hence it hasnt been considered extensivley in theory.

Standard Deviation and Variance

Standard Deviation is defined as :

$$\sigma \ = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n}}$$

variance is simply square of the standard deviation: σ^2

Inter Quartile Range [IQR]

All of the measures that we discussed for spread are sensitive to extreme values. IQR is one measure which is not sensitive to outliers. It is defined as difference between q_1 and q_3 where q_1 , q_3 are first and third quartiles respectively.

quartiles are the values which divide your data into 4 parts. Median is second quartile or q_2 , it divides into two equal parts. You can consider q_1 to be median of the first part and q_3 to be median of the second part.

IQR is not sensitive to outliers. why?

Shape

Measures of central tendency and variability still do not tell how frequently a value occurs in your data. Or in other words , given a value what are the chances that it will be in the data.

To get an idea regarding the same we can plot frequency bar charts or histograms.The Shape depicted by these histograms can be of three types:

- Symmetric
- Positively Skewed
- Negatively Skewed

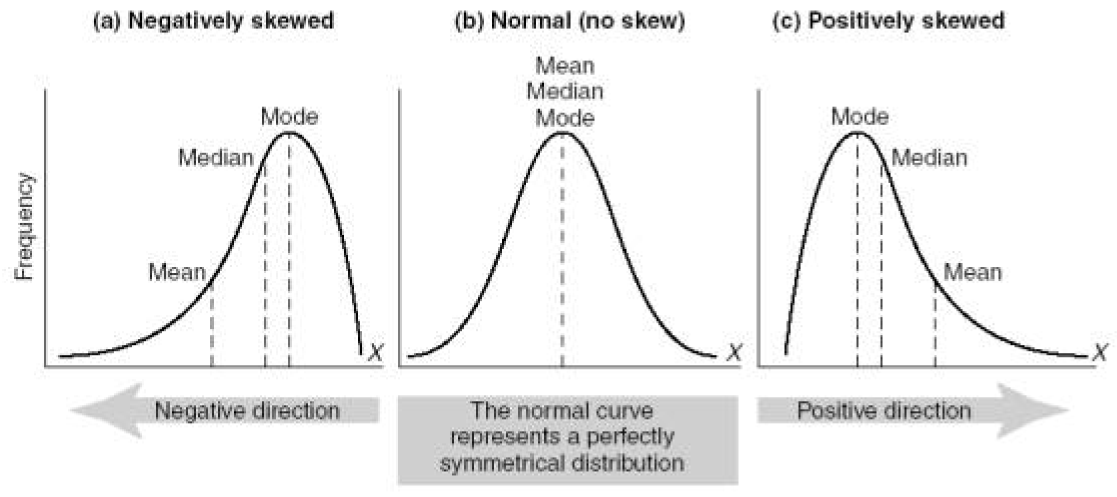


FIGURE 15.6 Examples of normal and skewed distributions

Figure 1: shape

Shape is also represented in terms of distributions which we discuss in detail in hypothesis testing module. For now you can understand symmetric shape to be where data values on either side of the mean are equally probable where as for positively and negatively skewed data has unbalanced properties.

For a symmetric shape

$$q_2 - q_1 = q_3 - q_2$$

where as for positively skewed shape

$$q_2 - q_1 < q_3 - q_2$$

and for negatively skewed shape

$$q_2 - q_1 > q_3 - q_2$$

Outliers

Outliers are the values which are too extreme w.r.t. to majority of the data. In order to find out if some value is an outlier , you first need to define a limit for your data beyond which you will consider any value to be an outlier. Here are two standard limits which are often used:

$$[\mu - k * \sigma, \mu + k * \sigma]$$

where μ and σ are mean and standard deviation of the data . k is a real number. Most used values of k in industry are 2 and 3. Higher the value k, more linient you are towards extreme values in the data.

$$[q_1 - k * IQR, q_3 + k * IQR]$$

k here again is a real number , most used value of k in industry is 1.5.

You can decide to chose either of these numbers, the first one is most used in industry though. What are value lies beyond these limits in the data is considered to be an outlier

Summary Stats in R

We have already seen few of individual summary functions. There is one for any kind of summary statistics that you have heard of. I amd going to put down some examples below. For these examples we are going to few inbuilt datasets in R.

```
data(mtcars)
```

Lets start with calculating mean, median, minimum, maximum, variance and standard deviation values of variable `mpg` in the data.

```
mean(mtcars$mpg)
```

```
## [1] 20.09062
```

```
median(mtcars$mpg)
```

```
## [1] 19.2
```

```
min(mtcars$mpg)
```

```
## [1] 10.4
```

```
max(mtcars$mpg)
```

```
## [1] 33.9
```

```
var(mtcars$mpg)
```

```
## [1] 36.3241
```

```
sd(mtcars$mpg)
```

```
## [1] 6.026948
```

On the same lines you'll find many other functions for various summary stats. You can use `summary` function to generate a standard 6 point summary from a variable or entire data.

```
summary(mtcars$mpg)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   10.40   15.42   19.20   20.09   22.80   33.90
```

```
summary(mtcars)
```

```
##       mpg             cyl             disp             hp
##  Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0
##  1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5
##  Median :19.20   Median :6.000   Median :196.3   Median :123.0
##  Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7
##  3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0
##  Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0
##       drat             wt             qsec             vs
##  Min.   :2.760   Min.   :1.513   Min.   :14.50   Min.   :0.0000
##  1st Qu.:3.080   1st Qu.:2.581   1st Qu.:16.89   1st Qu.:0.0000
##  Median :3.695   Median :3.325   Median :17.71   Median :0.0000
##  Mean   :3.597   Mean   :3.217   Mean   :17.85   Mean   :0.4375
##  3rd Qu.:3.920   3rd Qu.:3.610   3rd Qu.:18.90   3rd Qu.:1.0000
##  Max.   :4.930   Max.   :5.424   Max.   :22.90   Max.   :1.0000
##        am              gear            carb
##  Min.   :0.0000   Min.   :3.000   Min.   :1.000
##  1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:2.000
##  Median :0.0000   Median :4.000   Median :2.000
##  Mean   :0.4062   Mean   :3.688   Mean   :2.812
##  3rd Qu.:1.0000   3rd Qu.:4.000   3rd Qu.:4.000
##  Max.   :1.0000   Max.   :5.000   Max.   :8.000
```

Your next question should be , how to access individual values of these outputs. Lets start with looking at class of these outputs. If the `class` is something that we are not familiar with, we'll next look for `names` of elements within the output in their `class`.

```
s=summary(mtcars$mpg)
class(s)
```

```
## [1] "summaryDefault" "table"
```

```
names(s)
```

```
## [1] "Min."    "1st Qu." "Median"  "Mean"    "3rd Qu." "Max."
```

Now we know names of each components, we can use them to fetch the values:

```
s["Max."]
```

```
## Max.
## 33.9
```

Now lets try the same with summary of entire data.

```
s=summary(mtcars)
class(s)
```

```
## [1] "table"
```

```
names(s)
```

```
## NULL
```

This gets a little confusing. If output `s` doesnt have any names, how can we access any of the output? Lets try with numeric indices.

```
#this outputs element in 1st row and 2 column in tables s, which will be min value of variable "cyl"
s[1,2]
```

```
## [1] "Min.   :4.000  "
```

There are other summary functions as well coming from different packages in R. We'll look at one, that doesnt mean there are not others.

```
library(psych)
describe(mtcars)
```

```
##      vars  n   mean     sd median trimmed    mad   min    max  range  skew
## mpg     1 32  20.09   6.03  19.20   19.70   5.41 10.40  33.90  23.50  0.61
## cyl     2 32   6.19   1.79   6.00    6.23   2.97  4.00   8.00   4.00 -0.17
## disp    3 32 230.72 123.94 196.30  222.52 140.48 71.10 472.00 400.90  0.38
## hp      4 32 146.69  68.56 123.00  141.19  77.10 52.00 335.00 283.00  0.73
## drat    5 32   3.60   0.53   3.70    3.58   0.70  2.76   4.93   2.17  0.27
## wt      6 32   3.22   0.98   3.33    3.15   0.77  1.51   5.42   3.91  0.42
## qsec    7 32  17.85   1.79  17.71   17.83   1.42 14.50  22.90   8.40  0.37
## vs      8 32   0.44   0.50   0.00    0.42   0.00  0.00   1.00   1.00  0.24
## am      9 32   0.41   0.50   0.00    0.38   0.00  0.00   1.00   1.00  0.36
## gear   10 32   3.69   0.74   4.00    3.62   1.48  3.00   5.00   2.00  0.53
## carb   11 32   2.81   1.62   2.00    2.65   1.48  1.00   8.00   7.00  1.05
##      kurtosis    se
## mpg     -0.37  1.07
## cyl     -1.76  0.32
## disp    -1.21 21.91
## hp      -0.14 12.12
## drat    -0.71  0.09
## wt      -0.02  0.17
## qsec     0.34  0.32
## vs      -2.00  0.09
## am      -1.92  0.09
## gear    -1.07  0.13
## carb     1.26  0.29
```

you can see that this gives much more detailed summary of the dataset. Lets try to acces these values as well.

```
s=describe(mtcars)
class(s)
```

```
## [1] "psych"      "describe"   "data.frame"
```

you can see that the underlying class of the output object is data.frame. Good news , we know how to access ths output. here rownames are variables names of the dataset mtcars, column names are summary stats name. So if i wanted to look at max value of variable gear:

```
s["gear","max"]
```

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

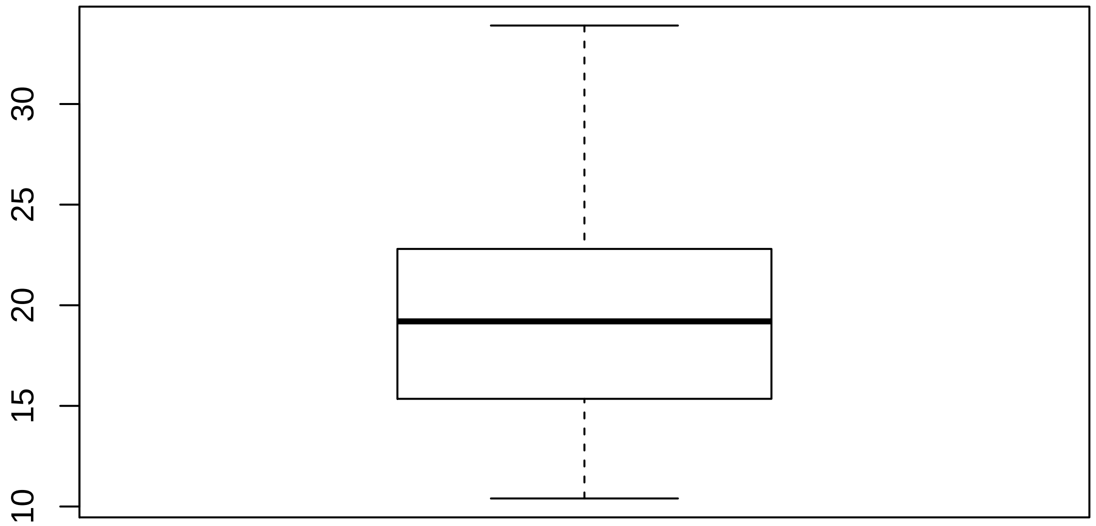
or multiple summaries of multiple variables [basically you can get any kind of subset from this data.frame]

```
s[c("gear","mpg","wt"),c("max","sd","skew")]
```

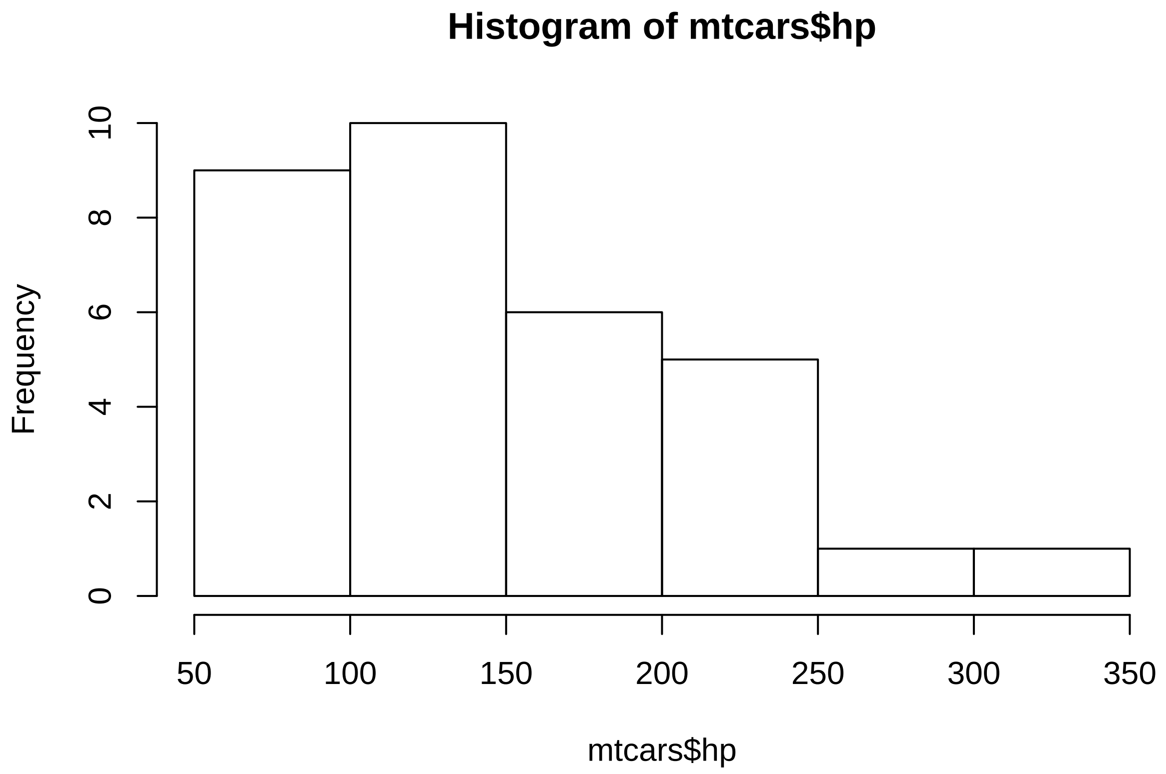
```
##         max   sd skew
## gear  5.00 0.74 0.53
## mpg  33.90 6.03 0.61
## wt    5.42 0.98 0.42
```

To get an idea about shape of a variable , we can plot boxplot and histograms.

```
boxplot(mtcars$mpg)
```



```
hist(mtcars$hp,breaks=10)
```



We'll look at better ways to visualize data in the next module dedicated specifically to this.

So far we have been looking at summaries of numeric variables only. How do we start to summaries categorical variables? Maximum that we can do is to look at what distinct values a categorical variable takes and what are the frequencies of these values in the data. Cross tables between two or more categorical variable is also a great source of insights. To learn how to do that in R, we'll be using datasets `Arthritis` from package `vcd`.

```
library(vcd)
```

```
## Loading required package: grid
```

```
data(Arthritis)
```

You can get details of the data set by running `?Arthritis`. To get simple frequency table of single variable you can use function `table`.

```
tab1=table(Arthritis$Improved)
tab1
```

```
##
##   None   Some Marked
##     42     14     28
```

You can convert these counts to percentage by apply function `prop.table` on the output of function table.

```
prop.table(tab1)
```

```
##
##      None      Some    Marked
## 0.5000000 0.1666667 0.3333333
```

You can apply round off to this output or convert it to proper percentages by multiplying by 100

```
round(prop.table(tab1),2)
```

```
##
##   None   Some Marked
##   0.50   0.17   0.33
```

```
prop.table(tab1)*100
```

```
##
##     None     Some   Marked
## 50.00000 16.66667 33.33333
```

For creating cross tables between two or more variables , you can either use function `table` or `xtabs`.

```
table(Arthritis$Improved,Arthritis$Sex)
```

```
##
##          Female Male
##   None       25   17
##   Some       12    2
##   Marked     22    6
```

```
xtabs(~Improved+Sex,data=Arthritis)
```

```
##         Sex
## Improved Female Male
##   None       25   17
##   Some       12    2
##   Marked     22    6
```

We'll be using function `xtabs` here.

```r
tab2=xtabs(~Treatment+Improved,data=Arthritis)
```

Applying `prop.table` on this count table will give you overall percentages, but you might interested in percetnage across a particular dimension. You can achieve that by supplying dimension number as second argument

```r
prop.table(tab2)
```

```
##          Improved
## Treatment       None       Some     Marked
##   Placebo 0.34523810 0.08333333 0.08333333
##   Treated 0.15476190 0.08333333 0.25000000
```

```r
#Percentages total to 1 across values of Treatment
prop.table(tab2,1)
```

```
##          Improved
## Treatment      None      Some    Marked
##   Placebo 0.6744186 0.1627907 0.1627907
##   Treated 0.3170732 0.1707317 0.5121951
```

```r
#Percentages total to 1 across values of Improved
prop.table(tab2,2)
```

```
##          Improved
## Treatment      None      Some    Marked
##   Placebo 0.6904762 0.5000000 0.2500000
##   Treated 0.3095238 0.5000000 0.7500000
```

function `margin.table` collapses your count table across the given dimension [passed as second argument]. If second argument is not present, result is simple count of total elements.

```r
margin.table(tab2)
```

```
## [1] 84
```

```r
margin.table(tab2,1)
```

```
## Treatment
## Placebo Treated
##      43      41
```

```r
margin.table(tab2,2)
```

```
## Improved
##   None   Some Marked
##     42     14     28
```

`margin.tables` does not look like a very useful function at this point of time when we are looking at two way tables only, it will have more utility if used with higher dimension cross tables. We'll look at that example as well very soon. For now, the next thing that we are going to discuss is , adding sum across a dimension without collapsing the table. This can be done using `addmargins` function.

```r
addmargins(tab2)
```

```
##          Improved
## Treatment None Some Marked Sum
##   Placebo   29    7      7  43
##   Treated   13    7     21  41
##   Sum       42   14     28  84
```

```
addmargins(tab2,2)
```

```
##          Improved
## Treatment None Some Marked Sum
##   Placebo   29    7      7  43
##   Treated   13    7     21  41
```

```
addmargins(tab2,1)
```

```
##          Improved
## Treatment None Some Marked
##   Placebo   29    7      7
##   Treated   13    7     21
##   Sum       42   14     28
```

Now we'll look at a 3 way cross table. To make the display more intuitive we'll use function `ftable`. Keep in mind that this function doesn't change counts in any manner, it just changes the way count table is being displayed.

```
tab3=xtabs(~Treatment+Sex+Improved,data=Arthritis)
ftable(tab3)
```

```
##                  Improved None Some Marked
## Treatment Sex
## Placebo   Female            19    7      6
##           Male              10    0      1
## Treated   Female             6    5     16
##           Male               7    2      5
```

Applying `margin.table` on this with individul dimension index or a vector index produces appropriate collpased tables.

```
margin.table(tab3,1)
```

```
## Treatment
## Placebo Treated
##      43      41
```

```
margin.table(tab3,c(1,3))
```

```
##          Improved
## Treatment None Some Marked
##   Placebo   29    7      7
##   Treated   13    7     21
```

Here is an example of applying prop.table with dimension vector.

```
ftable(prop.table(tab3,c(1,3)))
```

```
##                  Improved      None      Some    Marked
## Treatment Sex
## Placebo   Female          0.6551724 1.0000000 0.8571429
##           Male            0.3448276 0.0000000 0.1428571
## Treated   Female          0.4615385 0.7142857 0.7619048
##           Male            0.5384615 0.2857143 0.2380952
```

To understand the output better here, we are going to add margins to the output as well

```
ftable(addmargins(prop.table(tab3,c(1,2)),3))
```

```
##                    Improved       None       Some     Marked        Sum
## Treatment Sex
## Placebo   Female          0.59375000 0.21875000 0.18750000 1.00000000
##           Male            0.90909091 0.00000000 0.09090909 1.00000000
## Treated   Female          0.22222222 0.18518519 0.59259259 1.00000000
##           Male            0.50000000 0.14285714 0.35714286 1.00000000
```

Note: dimension index are assigned to different variable in the order in which they appear in `xtabs` call. In the example that we have shared here 1:Treatment , 2:Sex , 3:Improved.

We'll conclude here.

Prepared By: Lalit Sachan

Contact: lalit.sachan@edvancer.in

In case of any doubts/question regarding content of reading material, please post on QA forum in LMS.