Netflix Price Distribution Around Globe

CONTENTS

1. INTRODUCTION

Data analysis is the process of making raw data usable and actionable on the results obtained.

Turning data into actionable knowledge is the difference between faltering and success for businesses and organizations of all types.

Maximizing the value of information requires data analysis. Data analysis is the process of analysing raw data to reach a conclusion.

Providing an advanced data analysis approach is a development process that takes time and determination.

This report will include machine learning application on a given dataset along with basic statistical tests. Later, the elaboration of the model and how it is realized will be discussed later.

2.DESCRIPTIVE STATISTICS

2.1. What is Descriptive Statistic?

Descriptive Analytics can be used descriptively. As can be seen in the figure on the side, Descriptive Analytics is the most basic concept that gives information about the data. The information we define data (specifically business) is the analytics we obtain. Uncovering the truth about the data. What happened? The answer to the question (What?) is the concept.

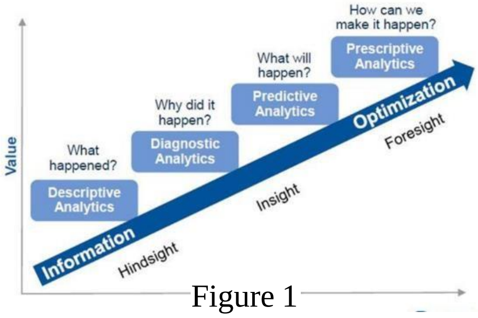


Figure 1

2.2. Random Variables

Random events are based on random variables.

Quantitative Variables

- Discrete random variables
  If a random appears values, the segment is randomly estimated.

- Continuous random variables
  If we cannot obtain a value by counting, if we need to measure, we can define these values as continuous random variables. Since a certain range will be mentioned, there are infinitely different options that can be taken between 0 and 1. Therefore, we must express a value in this range with continuous random variables.

Qualitative Variables

- Nominal random variables
- Ordinal random variables

2.3. Numerical Measures

Mean: is the average of a data set.

Median: mode is the most common number in a data set.

Mode: median is the middle of the set of numbers.

Percentiles: is a number where a certain percentage of scores fall below that number.

Range: is the difference between the lowest and highest values.

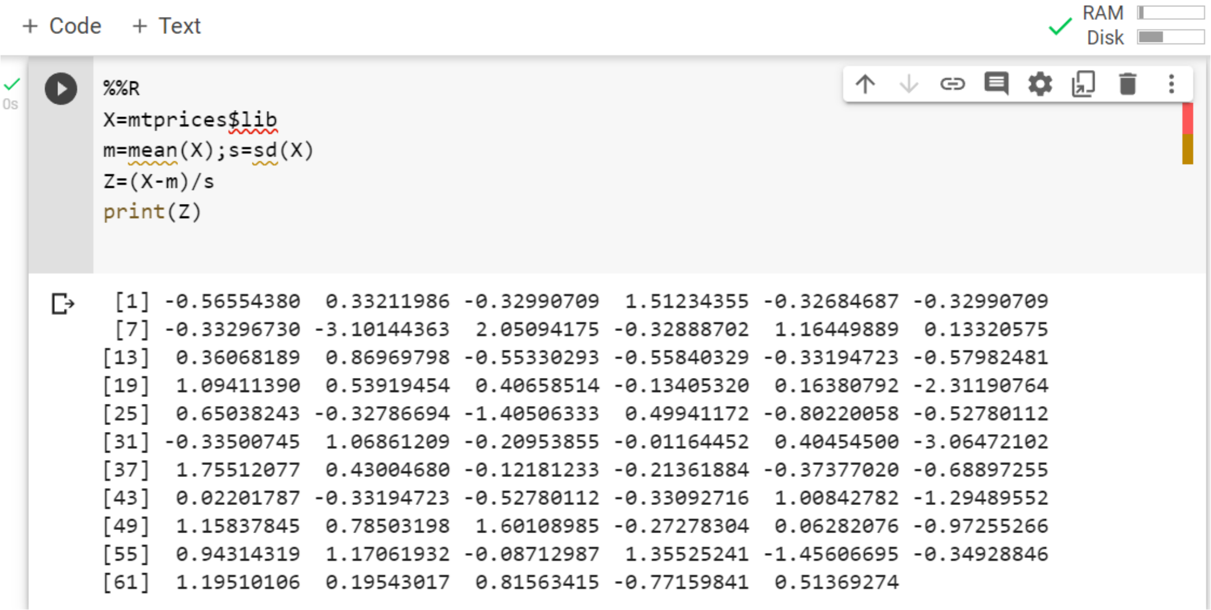
Variance: is a measure of variability. It is calculated by taking the average of squared deviations from the mean.

Standard deviation: is a measure of how dispersed the data is in relation to the mean.

Countries	Total Library Size	No. of TV Shows	No. of Movies	Cost Per Month - Basic ($)	Cost Per Month - Standard ($)	Cost Per Month - Premium ($)
Mean	5314.415	3518.954	1795.462	8.368462	11.99	15.61292
Median	5195	3512	1841	8.99	11.49	14.45
Mode	"4797" "4989" "4991"	3154	"1835" "1971"	"9.03"	"11.29" "14.67"	"20.32"
Range	5051	3559	2014	10.91	17.46	22.94

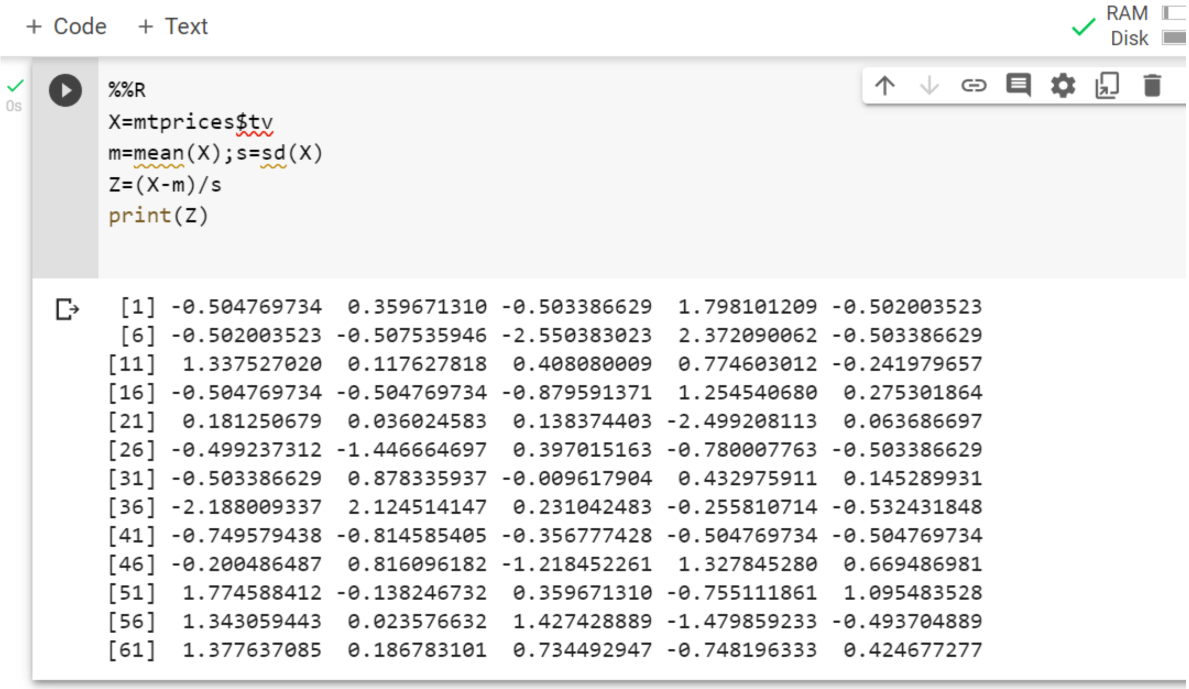
2.4. Applications of central and variational measures

Data normalization: is adjusting values measured on different scales to a notionally common scale, often prior to averaging.



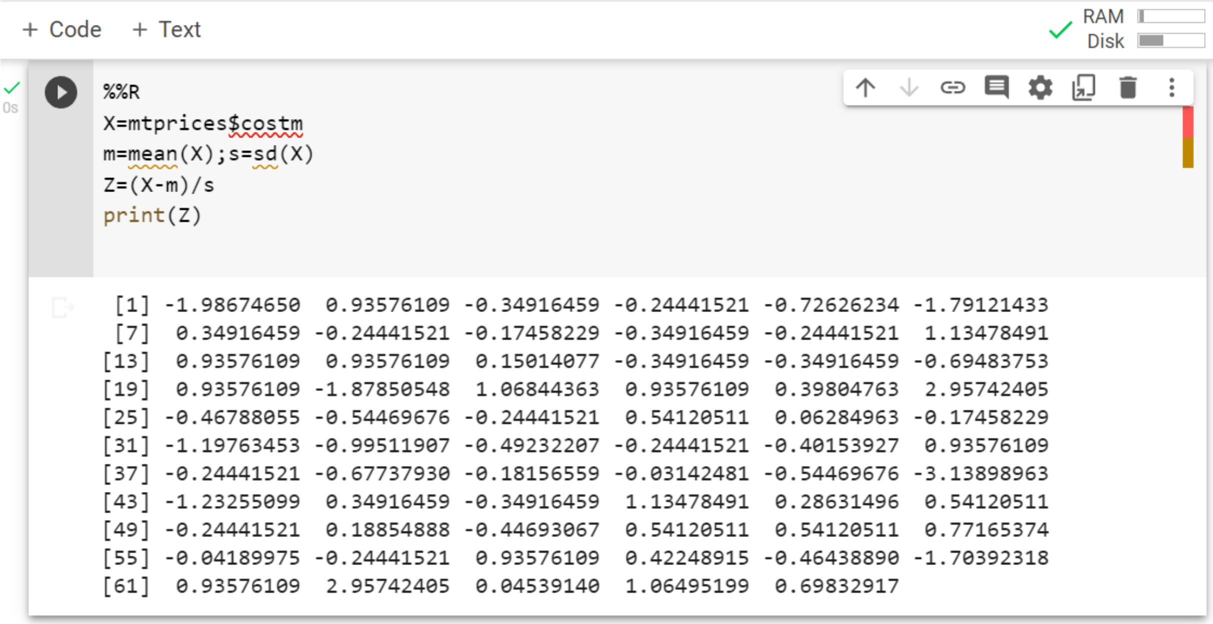
```
%%R
X=mtprices$lib
m=mean(X);s=sd(X)
Z=(X-m)/s
print(Z)
```

```
 [1] -0.56554380  0.33211986 -0.32990709  1.51234355 -0.32684687 -0.32990709
 [7] -0.33296730 -3.10144363  2.05094175 -0.32888702  1.16449889  0.13320575
[13]  0.36068189  0.86969798 -0.55330293 -0.55840329 -0.33194723 -0.57982481
[19]  1.09411390  0.53919454  0.40658514 -0.13405320  0.16380792 -2.31190764
[25]  0.65038243 -0.32786694 -1.40506333  0.49941172 -0.80220058 -0.52780112
[31] -0.33500745  1.06861209 -0.20953855 -0.01164452  0.40454500 -3.06472102
[37]  1.75512077  0.43004680 -0.12181233 -0.21361884 -0.37377020 -0.68897255
[43]  0.02201787 -0.33194723 -0.52780112 -0.33092716  1.00842782 -1.29489552
[49]  1.15837845  0.78503198  1.60108985 -0.27278304  0.06282076 -0.97255266
[55]  0.94314319  1.17061932 -0.08712987  1.35525241 -1.45606695 -0.34928846
[61]  1.19510106  0.19543017  0.81563415 -0.77159841  0.51369274
```



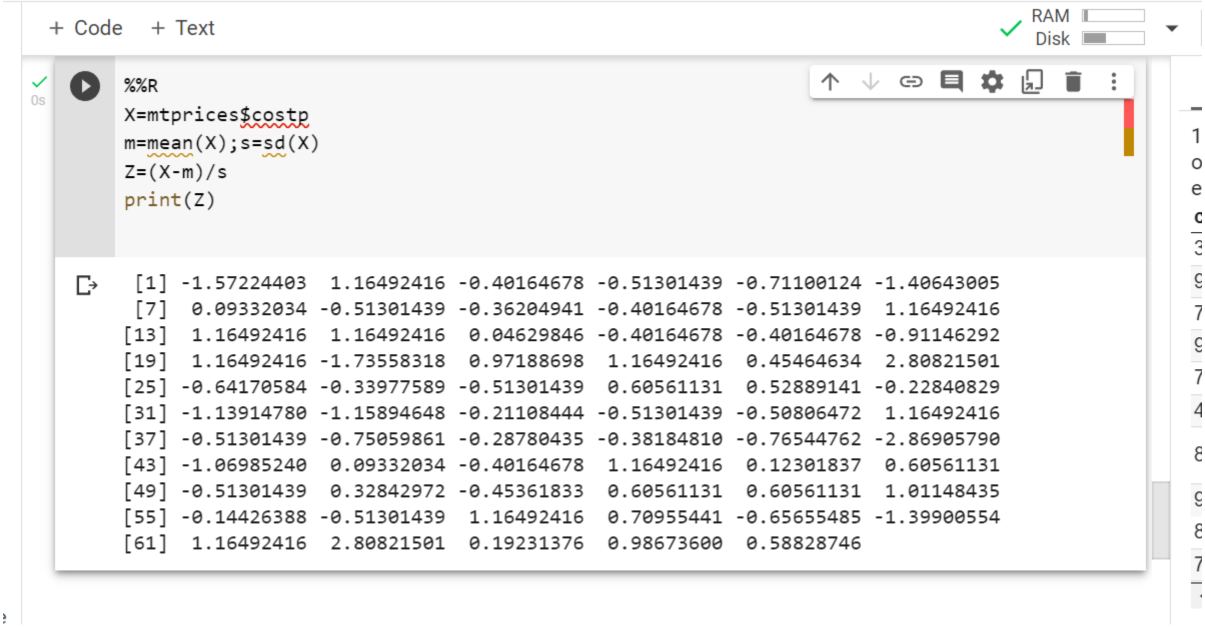
```
%%R
X=mtprices$tv
m=mean(X);s=sd(X)
Z=(X-m)/s
print(Z)
```

```
 [1] -0.504769734  0.359671310 -0.503386629  1.798101209 -0.502003523
 [6] -0.502003523 -0.507535946 -2.550383023  2.372090062 -0.503386629
[11]  1.337527020  0.117627818  0.408080009  0.774603012 -0.241979657
[16] -0.504769734 -0.504769734 -0.879591371  1.254540680  0.275301864
[21]  0.181250679  0.036024583  0.138374403 -2.499208113  0.063686697
[26] -0.499237312 -1.446664697  0.397015163 -0.780007763 -0.503386629
[31] -0.503386629  0.878335937 -0.009617904  0.432975911  0.145289931
[36] -2.188009337  2.124514147  0.231042483 -0.255810714 -0.532431848
[41] -0.749579438 -0.814585405 -0.356777428 -0.504769734 -0.504769734
[46] -0.200486487  0.816096182 -1.218452261  1.327845280  0.669486981
[51]  1.774588412 -0.138246732  0.359671310 -0.755111861  1.095483528
[56]  1.343059443  0.023576632  1.427428889 -1.479859233 -0.493704889
[61]  1.377637085  0.186783101  0.734492947 -0.748196333  0.424677277
```



```
%%R
X=mtprices$costm
m=mean(X);s=sd(X)
Z=(X-m)/s
print(Z)
```

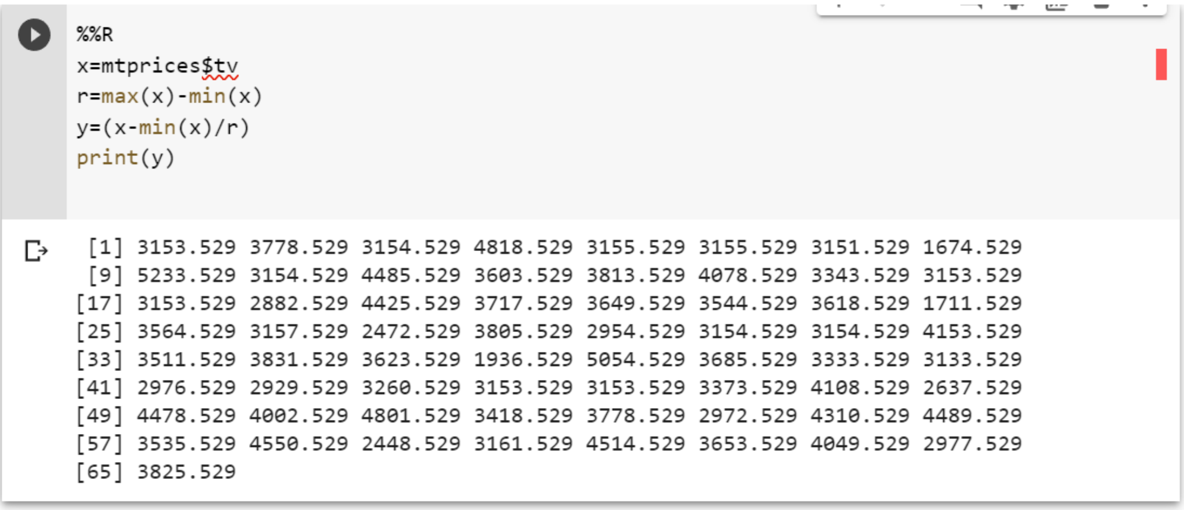
```
 [1] -1.98674650  0.93576109 -0.34916459 -0.24441521 -0.72626234 -1.79121433
 [7]  0.34916459 -0.24441521 -0.17458229 -0.34916459 -0.24441521  1.13478491
[13]  0.93576109  0.93576109  0.15014077 -0.34916459 -0.34916459 -0.69483753
[19]  0.93576109 -1.87850548  1.06844363  0.93576109  0.39804763  2.95742405
[25] -0.46788055 -0.54469676 -0.24441521  0.54120511  0.06284963 -0.17458229
[31] -1.19763453 -0.99511907 -0.49232207 -0.24441521 -0.40153927  0.93576109
[37] -0.24441521 -0.67737930 -0.18156559 -0.03142481 -0.54469676 -3.13898963
[43] -1.23255099  0.34916459 -0.34916459  1.13478491  0.28631496  0.54120511
[49] -0.24441521  0.18854888 -0.44693067  0.54120511  0.54120511  0.77165374
[55] -0.04189975 -0.24441521  0.93576109  0.42248915 -0.46438890 -1.70392318
[61]  0.93576109  2.95742405  0.04539140  1.06495199  0.69832917
```



```
%%R
X=mtprices$costp
m=mean(X);s=sd(X)
Z=(X-m)/s
print(Z)
```

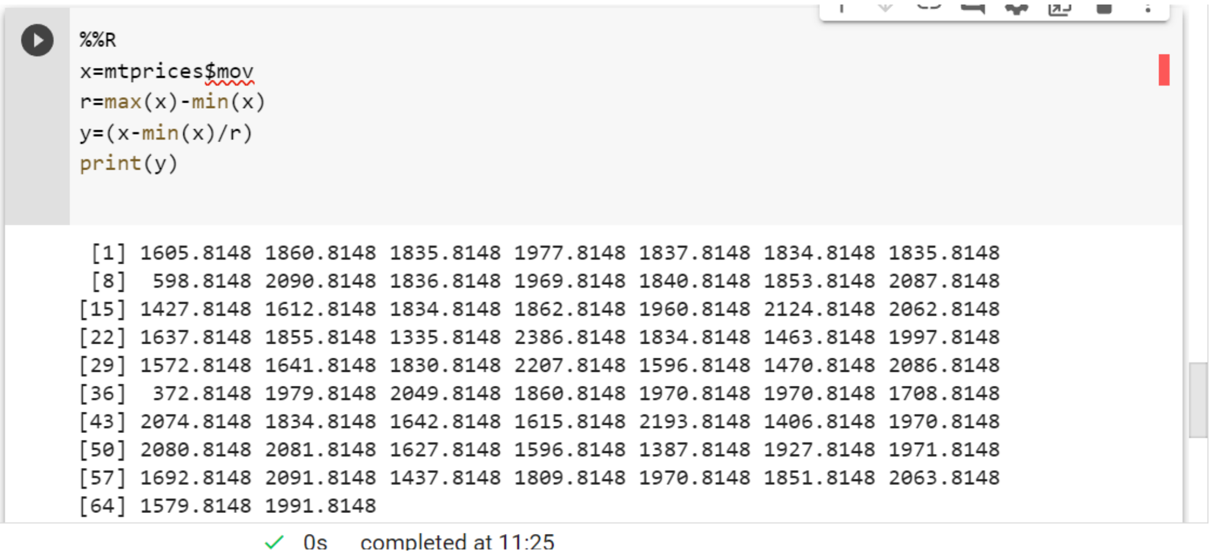
```
 [1] -1.57224403  1.16492416 -0.40164678 -0.51301439 -0.71100124 -1.40643005
 [7]  0.09332034 -0.51301439 -0.36204941 -0.40164678 -0.51301439  1.16492416
[13]  1.16492416  1.16492416  0.04629846 -0.40164678 -0.40164678 -0.91146292
[19]  1.16492416 -1.73558318  0.97188698  1.16492416  0.45464634  2.80821501
[25] -0.64170584 -0.33977589 -0.51301439  0.60561131  0.52889141 -0.22840829
[31] -1.13914780 -1.15894648 -0.21108444 -0.51301439 -0.50806472  1.16492416
[37] -0.51301439 -0.75059861 -0.28780435 -0.38184810 -0.76544762 -2.86905790
[43] -1.06985240  0.09332034 -0.40164678  1.16492416  0.12301837  0.60561131
[49] -0.51301439  0.32842972 -0.45361833  0.60561131  0.60561131  1.01148435
[55] -0.14426388 -0.51301439  1.16492416  0.70955441 -0.65655485 -1.39900554
[61]  1.16492416  2.80821501  0.19231376  0.98673600  0.58828746
```

Data scaling: is usually means a linear transformation of the form f(x)=ax+b.



```
%%R
x=mtprices$tv
r=max(x)-min(x)
y=(x-min(x)/r)
print(y)
```

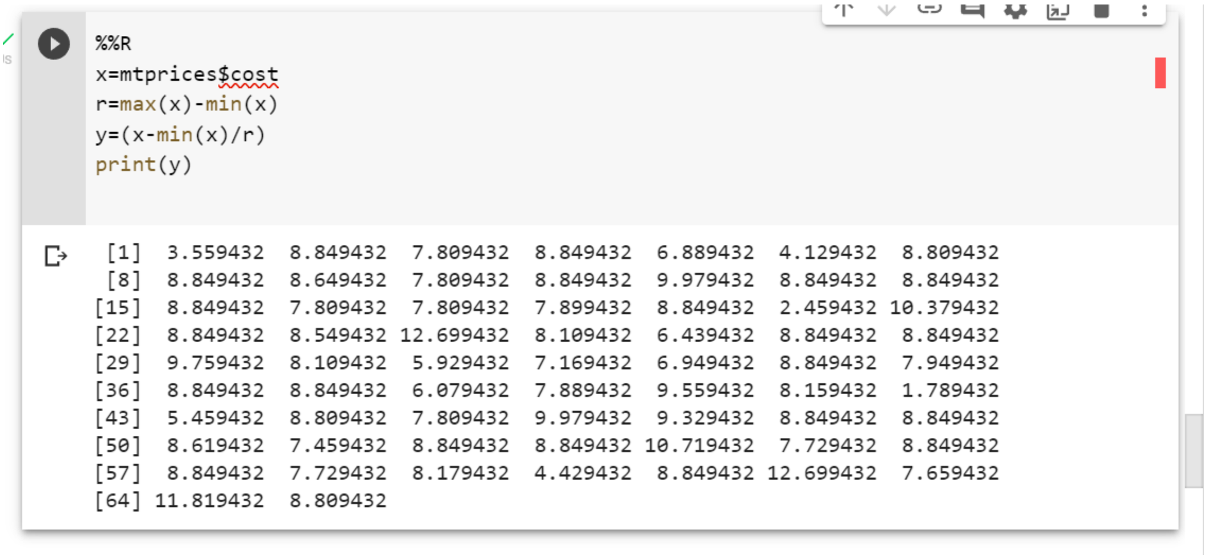
```
 [1] 3153.529 3778.529 3154.529 4818.529 3155.529 3155.529 3151.529 1674.529
 [9] 5233.529 3154.529 4485.529 3603.529 3813.529 4078.529 3343.529 3153.529
[17] 3153.529 2882.529 4425.529 3717.529 3649.529 3544.529 3618.529 1711.529
[25] 3564.529 3157.529 2472.529 3805.529 2954.529 3154.529 3154.529 4153.529
[33] 3511.529 3831.529 3623.529 1936.529 5054.529 3685.529 3333.529 3133.529
[41] 2976.529 2929.529 3260.529 3153.529 3153.529 3373.529 4108.529 2637.529
[49] 4478.529 4002.529 4801.529 3418.529 3778.529 2972.529 4310.529 4489.529
[57] 3535.529 4550.529 2448.529 3161.529 4514.529 3653.529 4049.529 2977.529
[65] 3825.529
```



```
%%R
x=mtprices$mov
r=max(x)-min(x)
y=(x-min(x)/r)
print(y)
```

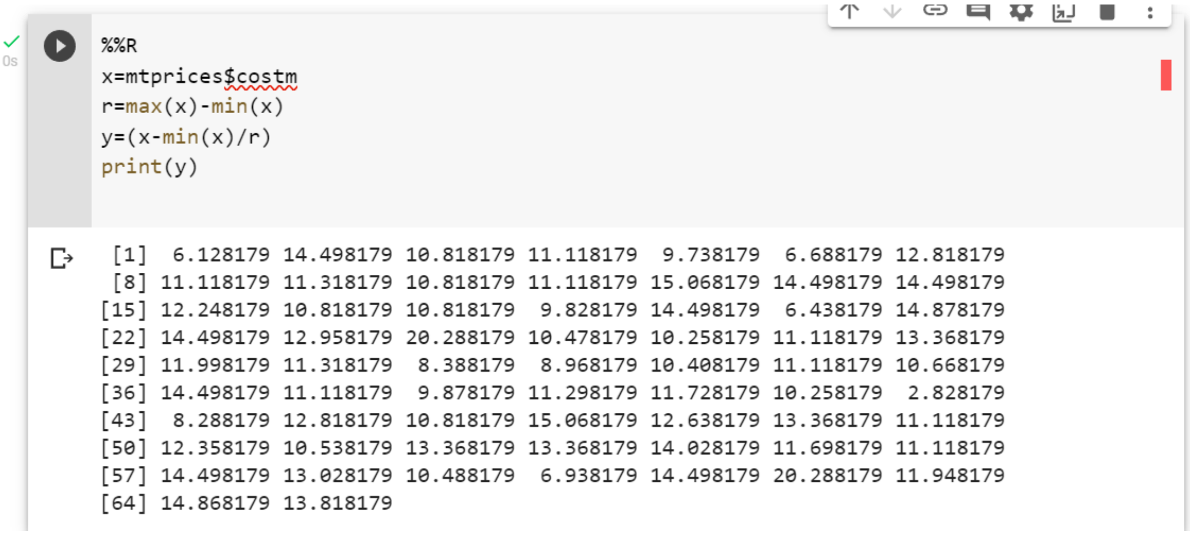
```
 [1] 1605.8148 1860.8148 1835.8148 1977.8148 1837.8148 1834.8148 1835.8148
 [8]  598.8148 2090.8148 1836.8148 1969.8148 1840.8148 1853.8148 2087.8148
[15] 1427.8148 1612.8148 1834.8148 1862.8148 1960.8148 2124.8148 2062.8148
[22] 1637.8148 1855.8148 1335.8148 2386.8148 1834.8148 1463.8148 1997.8148
[29] 1572.8148 1641.8148 1830.8148 2207.8148 1596.8148 1470.8148 2086.8148
[36]  372.8148 1979.8148 2049.8148 1860.8148 1970.8148 1970.8148 1708.8148
[43] 2074.8148 1834.8148 1642.8148 1615.8148 2193.8148 1406.8148 1970.8148
[50] 2080.8148 2081.8148 1627.8148 1596.8148 1387.8148 1927.8148 1971.8148
[57] 1692.8148 2091.8148 1437.8148 1809.8148 1970.8148 1851.8148 2063.8148
[64] 1579.8148 1991.8148
```

✓ 0s completed at 11:25



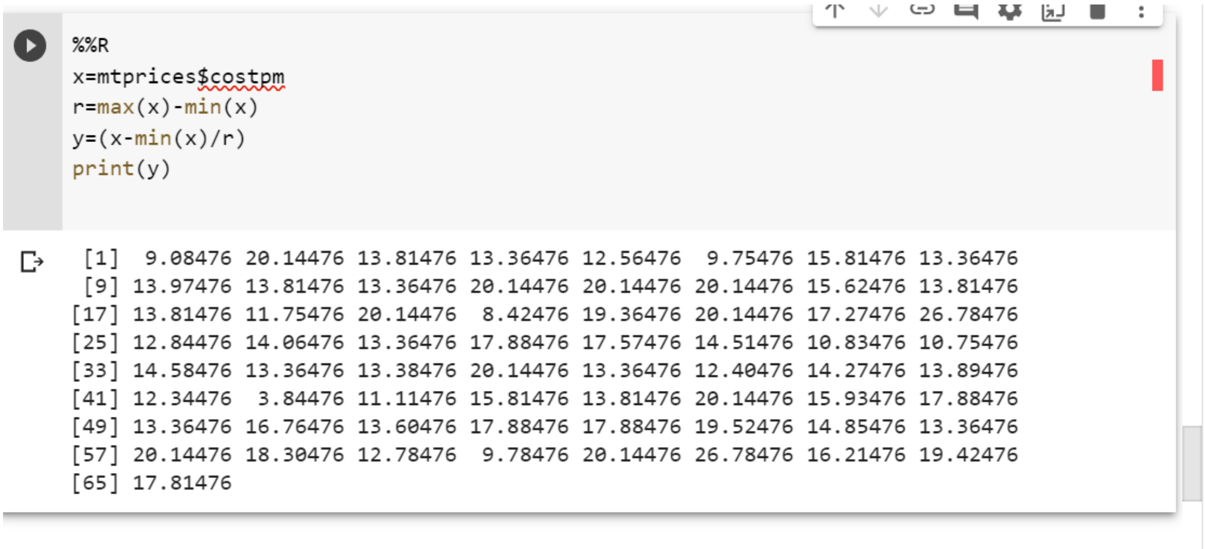
```
%%R
x=mtprices$cost
r=max(x)-min(x)
y=(x-min(x)/r)
print(y)
```

```
 [1]  3.559432  8.849432  7.809432  8.849432  6.889432  4.129432  8.809432
 [8]  8.849432  8.649432  7.809432  8.849432  9.979432  8.849432  8.849432
[15]  8.849432  7.809432  7.809432  7.899432  8.849432  2.459432 10.379432
[22]  8.849432  8.549432 12.699432  8.109432  6.439432  8.849432  8.849432
[29]  9.759432  8.109432  5.929432  7.169432  6.949432  8.849432  7.949432
[36]  8.849432  8.849432  6.079432  7.889432  9.559432  8.159432  1.789432
[43]  5.459432  8.809432  7.809432  9.979432  9.329432  8.849432  8.849432
[50]  8.619432  7.459432  8.849432  8.849432 10.719432  7.729432  8.849432
[57]  8.849432  7.729432  8.179432  4.429432  8.849432 12.699432  7.659432
[64] 11.819432  8.809432
```



```
%%R
x=mtprices$costm
r=max(x)-min(x)
y=(x-min(x)/r)
print(y)
```

```
 [1]  6.128179 14.498179 10.818179 11.118179  9.738179  6.688179 12.818179
 [8] 11.118179 11.318179 10.818179 11.118179 15.068179 14.498179 14.498179
[15] 12.248179 10.818179 10.818179  9.828179 14.498179  6.438179 14.878179
[22] 14.498179 12.958179 20.288179 10.478179 10.258179 11.118179 13.368179
[29] 11.998179 11.318179  8.388179  8.968179 10.408179 11.118179 10.668179
[36] 14.498179 11.118179  9.878179 11.298179 11.728179 10.258179  2.828179
[43]  8.288179 12.818179 10.818179 15.068179 12.638179 13.368179 11.118179
[50] 12.358179 10.538179 13.368179 13.368179 14.028179 11.698179 11.118179
[57] 14.498179 13.028179 10.488179  6.938179 14.498179 20.288179 11.948179
[64] 14.868179 13.818179
```

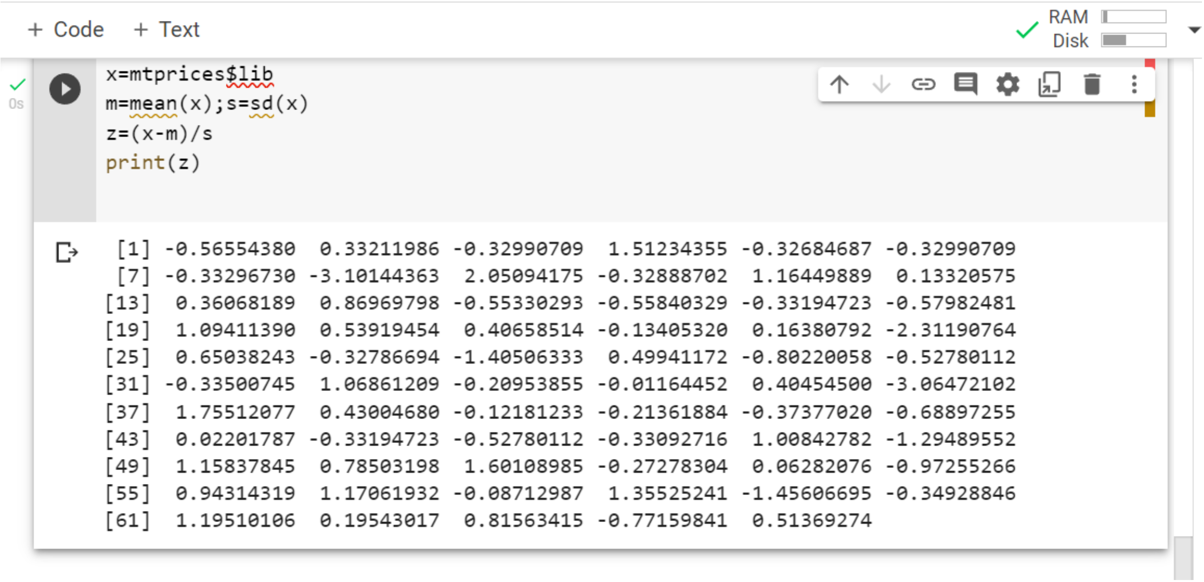


```
%%R
x=mtprices$costpm
r=max(x)-min(x)
y=(x-min(x)/r)
print(y)
```

```
 [1]  9.08476 20.14476 13.81476 13.36476 12.56476  9.75476 15.81476 13.36476
 [9] 13.97476 13.81476 13.36476 20.14476 20.14476 20.14476 15.62476 13.81476
[17] 13.81476 11.75476 20.14476  8.42476 19.36476 20.14476 17.27476 26.78476
[25] 12.84476 14.06476 13.36476 17.88476 17.57476 14.51476 10.83476 10.75476
[33] 14.58476 13.36476 13.38476 20.14476 13.36476 12.40476 14.27476 13.89476
[41] 12.34476  3.84476 11.11476 15.81476 13.81476 20.14476 15.93476 17.88476
[49] 13.36476 16.76476 13.60476 17.88476 17.88476 19.52476 14.85476 13.36476
[57] 20.14476 18.30476 12.78476  9.78476 20.14476 26.78476 16.21476 19.42476
[65] 17.81476
```

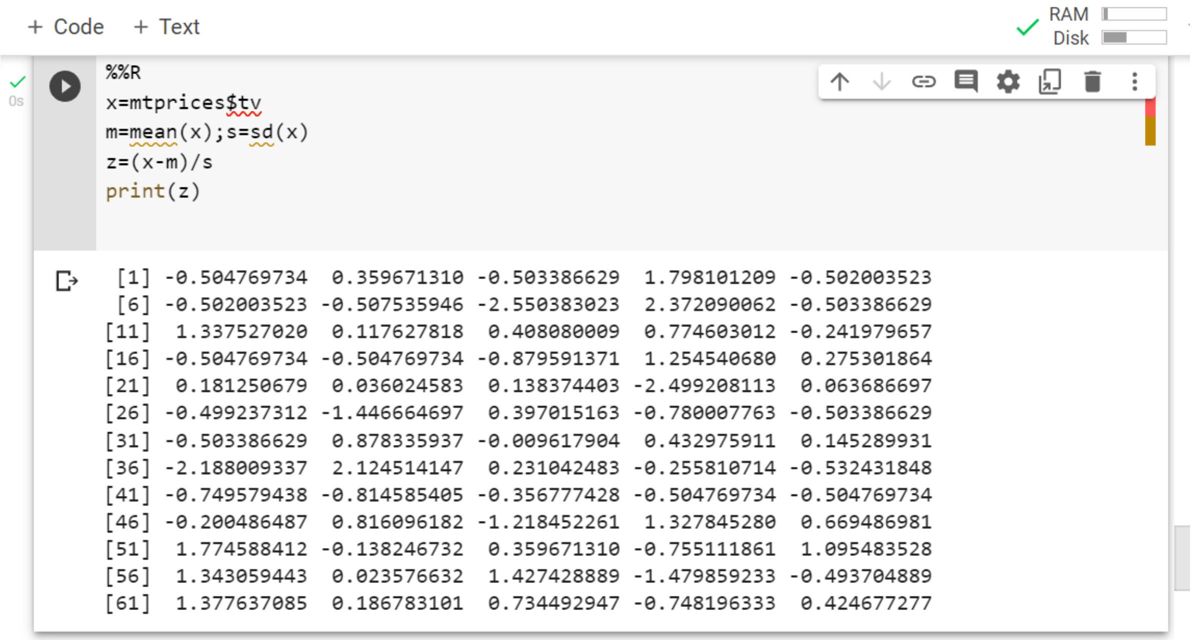
Outlier detection: is an observation that appears to deviate markedly from other observations in the sample.

Missing values imputation: a simple and popular approach to data imputation involves using statistical methods to estimate a value for a column from those values that are present, then replace all missing values in the column with the calculated statistic.



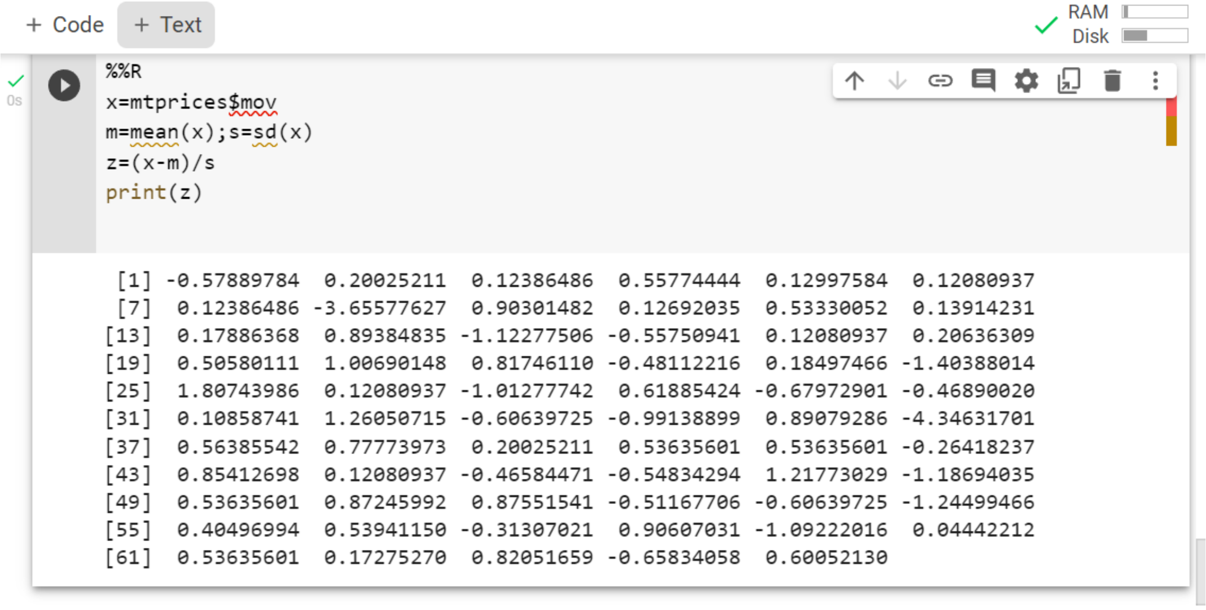
```
x=mtprices$lib
m=mean(x);s=sd(x)
z=(x-m)/s
print(z)
```

```
 [1] -0.56554380  0.33211986 -0.32990709  1.51234355 -0.32684687 -0.32990709
 [7] -0.33296730 -3.10144363  2.05094175 -0.32888702  1.16449889  0.13320575
[13]  0.36068189  0.86969798 -0.55330293 -0.55840329 -0.33194723 -0.57982481
[19]  1.09411390  0.53919454  0.40658514 -0.13405320  0.16380792 -2.31190764
[25]  0.65038243 -0.32786694 -1.40506333  0.49941172 -0.80220058 -0.52780112
[31] -0.33500745  1.06861209 -0.20953855 -0.01164452  0.40454500 -3.06472102
[37]  1.75512077  0.43004680 -0.12181233 -0.21361884 -0.37377020 -0.68897255
[43]  0.02201787 -0.33194723 -0.52780112 -0.33092716  1.00842782 -1.29489552
[49]  1.15837845  0.78503198  1.60108985 -0.27278304  0.06282076 -0.97255266
[55]  0.94314319  1.17061932 -0.08712987  1.35525241 -1.45606695 -0.34928846
[61]  1.19510106  0.19543017  0.81563415 -0.77159841  0.51369274
```



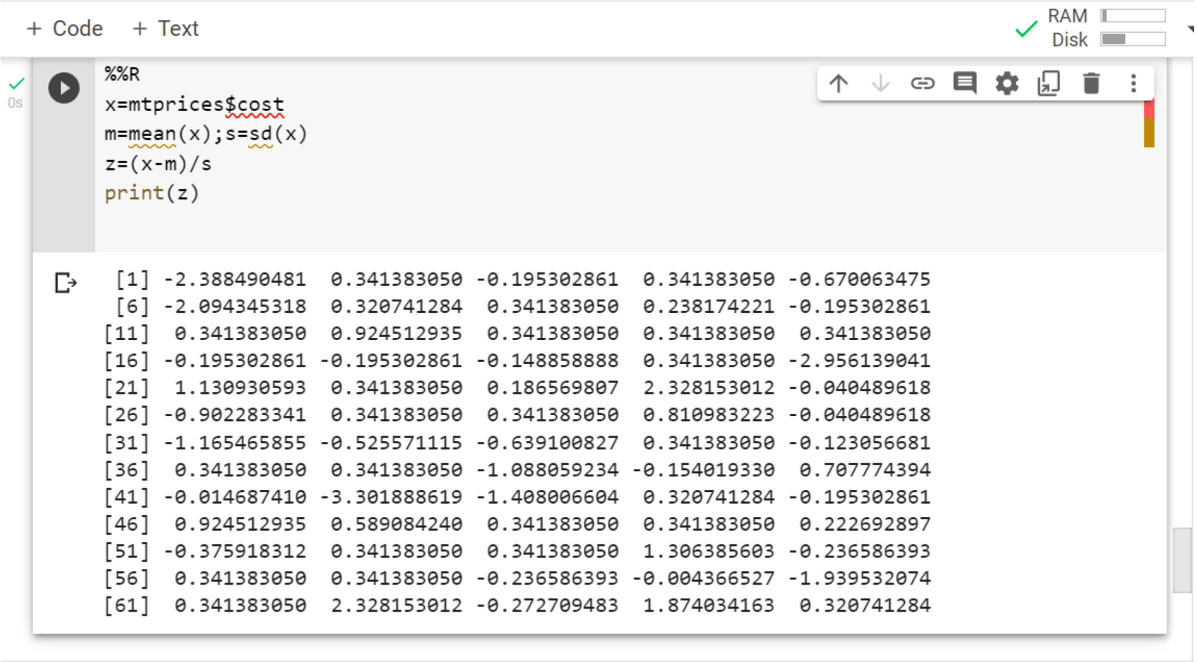
```
%%R
x=mtprices$tv
m=mean(x);s=sd(x)
z=(x-m)/s
print(z)
```

```
 [1] -0.504769734  0.359671310 -0.503386629  1.798101209 -0.502003523
 [6] -0.502003523 -0.507535946 -2.550383023  2.372090062 -0.503386629
[11]  1.337527020  0.117627818  0.408080009  0.774603012 -0.241979657
[16] -0.504769734 -0.504769734 -0.879591371  1.254540680  0.275301864
[21]  0.181250679  0.036024583  0.138374403 -2.499208113  0.063686697
[26] -0.499237312 -1.446664697  0.397015163 -0.780007763 -0.503386629
[31] -0.503386629  0.878335937 -0.009617904  0.432975911  0.145289931
[36] -2.188009337  2.124514147  0.231042483 -0.255810714 -0.532431848
[41] -0.749579438 -0.814585405 -0.356777428 -0.504769734 -0.504769734
[46] -0.200486487  0.816096182 -1.218452261  1.327845280  0.669486981
[51]  1.774588412 -0.138246732  0.359671310 -0.755111861  1.095483528
[56]  1.343059443  0.023576632  1.427428889 -1.479859233 -0.493704889
[61]  1.377637085  0.186783101  0.734492947 -0.748196333  0.424677277
```



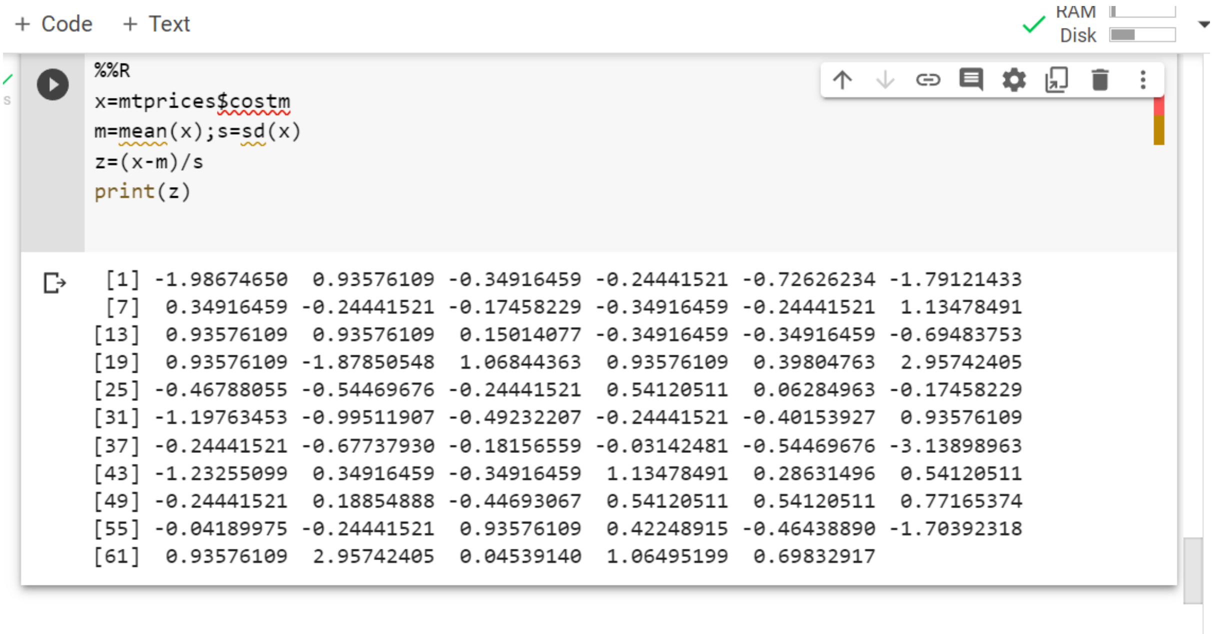
```
%%R
x=mtprices$mov
m=mean(x);s=sd(x)
z=(x-m)/s
print(z)
```

```
 [1] -0.57889784  0.20025211  0.12386486  0.55774444  0.12997584  0.12080937
 [7]  0.12386486 -3.65577627  0.90301482  0.12692035  0.53330052  0.13914231
[13]  0.17886368  0.89384835 -1.12277506 -0.55750941  0.12080937  0.20636309
[19]  0.50580111  1.00690148  0.81746110 -0.48112216  0.18497466 -1.40388014
[25]  1.80743986  0.12080937 -1.01277742  0.61885424 -0.67972901 -0.46890020
[31]  0.10858741  1.26050715 -0.60639725 -0.99138899  0.89079286 -4.34631701
[37]  0.56385542  0.77773973  0.20025211  0.53635601  0.53635601 -0.26418237
[43]  0.85412698  0.12080937 -0.46584471 -0.54834294  1.21773029 -1.18694035
[49]  0.53635601  0.87245992  0.87551541 -0.51167706 -0.60639725 -1.24499466
[55]  0.40496994  0.53941150 -0.31307021  0.90607031 -1.09222016  0.04442212
[61]  0.53635601  0.17275270  0.82051659 -0.65834058  0.60052130
```



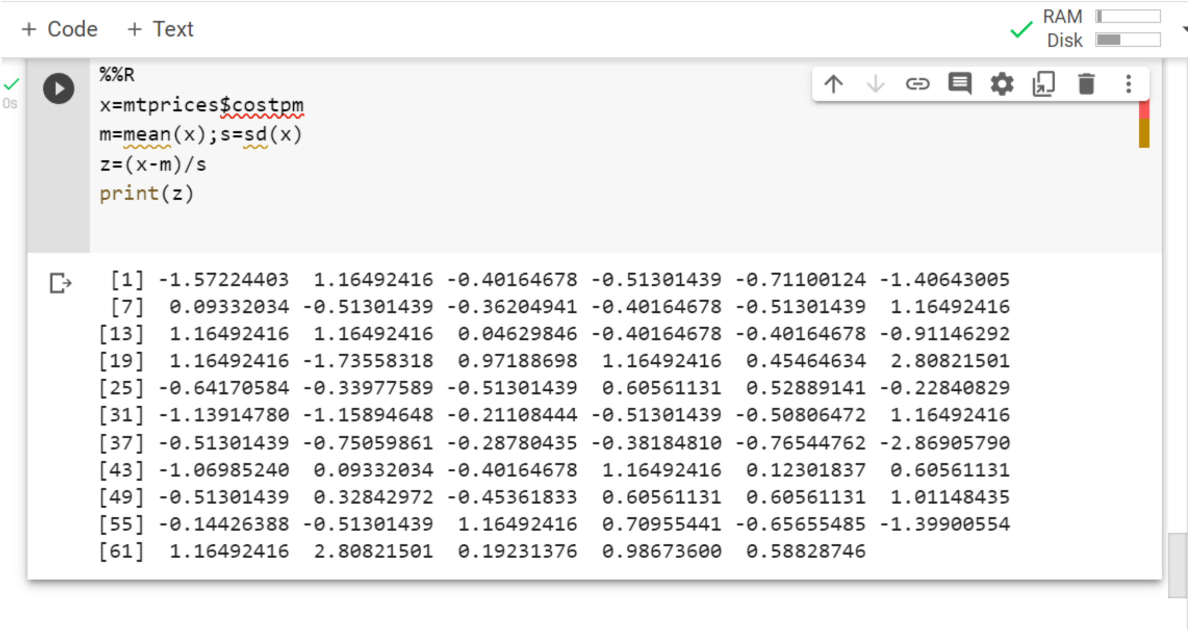
```
%%R
x=mtprices$cost
m=mean(x);s=sd(x)
z=(x-m)/s
print(z)
```

```
 [1] -2.388490481  0.341383050 -0.195302861  0.341383050 -0.670063475
 [6] -2.094345318  0.320741284  0.341383050  0.238174221 -0.195302861
[11]  0.341383050  0.924512935  0.341383050  0.341383050  0.341383050
[16] -0.195302861 -0.195302861 -0.148858888  0.341383050 -2.956139041
[21]  1.130930593  0.341383050  0.186569807  2.328153012 -0.040489618
[26] -0.902283341  0.341383050  0.341383050  0.810983223 -0.040489618
[31] -1.165465855 -0.525571115 -0.639100827  0.341383050 -0.123056681
[36]  0.341383050  0.341383050 -1.088059234 -0.154019330  0.707774394
[41] -0.014687410 -3.301888619 -1.408006604  0.320741284 -0.195302861
[46]  0.924512935  0.589084240  0.341383050  0.341383050  0.222692897
[51] -0.375918312  0.341383050  0.341383050  1.306385603 -0.236586393
[56]  0.341383050  0.341383050 -0.236586393 -0.004366527 -1.939532074
[61]  0.341383050  2.328153012 -0.272709483  1.874034163  0.320741284
```



```
%%R
x=mtprices$costm
m=mean(x);s=sd(x)
z=(x-m)/s
print(z)
```

```
 [1] -1.98674650  0.93576109 -0.34916459 -0.24441521 -0.72626234 -1.79121433
 [7]  0.34916459 -0.24441521 -0.17458229 -0.34916459 -0.24441521  1.13478491
[13]  0.93576109  0.93576109  0.15014077 -0.34916459 -0.34916459 -0.69483753
[19]  0.93576109 -1.87850548  1.06844363  0.93576109  0.39804763  2.95742405
[25] -0.46788055 -0.54469676 -0.24441521  0.54120511  0.06284963 -0.17458229
[31] -1.19763453 -0.99511907 -0.49232207 -0.24441521 -0.40153927  0.93576109
[37] -0.24441521 -0.67737930 -0.18156559 -0.03142481 -0.54469676 -3.13898963
[43] -1.23255099  0.34916459 -0.34916459  1.13478491  0.28631496  0.54120511
[49] -0.24441521  0.18854888 -0.44693067  0.54120511  0.54120511  0.77165374
[55] -0.04189975 -0.24441521  0.93576109  0.42248915 -0.46438890 -1.70392318
[61]  0.93576109  2.95742405  0.04539140  1.06495199  0.69832917
```



```
%%R
x=mtprices$costpm
m=mean(x);s=sd(x)
z=(x-m)/s
print(z)
```

```
 [1] -1.57224403  1.16492416 -0.40164678 -0.51301439 -0.71100124 -1.40643005
 [7]  0.09332034 -0.51301439 -0.36204941 -0.40164678 -0.51301439  1.16492416
[13]  1.16492416  1.16492416  0.04629846 -0.40164678 -0.40164678 -0.91146292
[19]  1.16492416 -1.73558318  0.97188698  1.16492416  0.45464634  2.80821501
[25] -0.64170584 -0.33977589 -0.51301439  0.60561131  0.52889141 -0.22840829
[31] -1.13914780 -1.15894648 -0.21108444 -0.51301439 -0.50806472  1.16492416
[37] -0.51301439 -0.75059861 -0.28780435 -0.38184810 -0.76544762 -2.86905790
[43] -1.06985240  0.09332034 -0.40164678  1.16492416  0.12301837  0.60561131
[49] -0.51301439  0.32842972 -0.45361833  0.60561131  0.60561131  1.01148435
[55] -0.14426388 -0.51301439  1.16492416  0.70955441 -0.65655485 -1.39900554
[61]  1.16492416  2.80821501  0.19231376  0.98673600  0.58828746
```

3. PROBABILITY MODELS

3.1 What is probability model ?

Probability model is a model or technique that can predict the likelihood or chances of an event/events to occur. Probability models are used to predict the possible outcome of data by analysing the dataset for past and present event. Depending on the type of dataset there are different models that can be applied on the dataset.

3.2 List of probability models used.

In the assessment we have used three different types of probability models

1. **Bernoulli model**
2. **Poisson model**
3. **Uniform model**

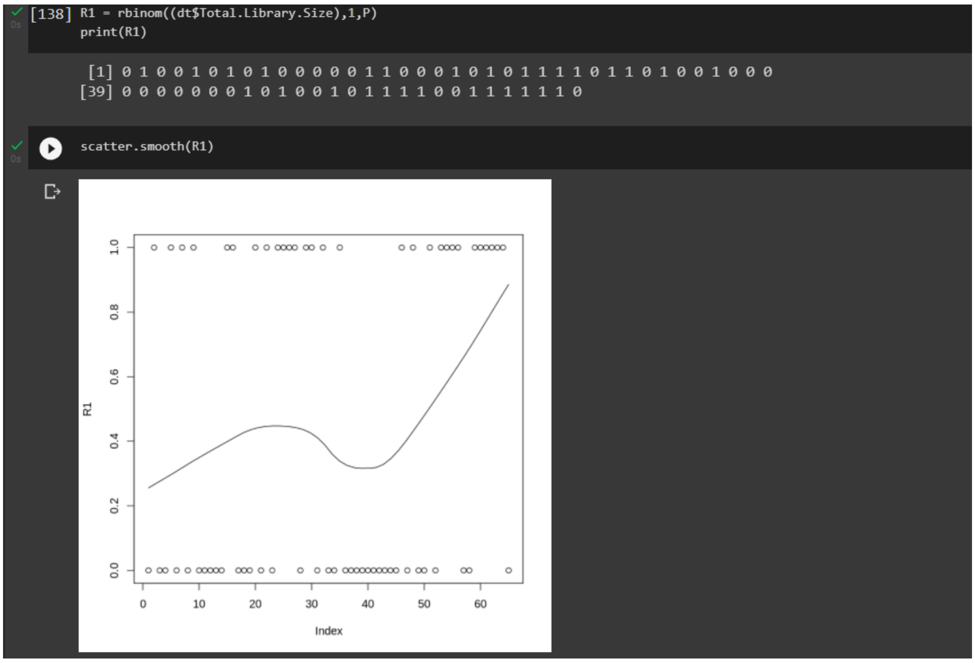
3.2.1

1. Bernoulli model: Bernoulli model is the simplest type of discrete probability model because it deals with only binary variables (i.e., 1’s and 0’s). This model is used in coin flip experiments where the possible outcome is head or tail. Bernoulli model is used to separate the data in the groups of two.

1.1 Weakness: - Bernoulli model can only work with binary dataset which sometime arises a problem in segregating the data.

1.2 Formula -

$$f(k;p) = \begin{cases} p & \text{if } k = 1, \\ q = 1 - p & \text{if } k = 0. \end{cases}$$



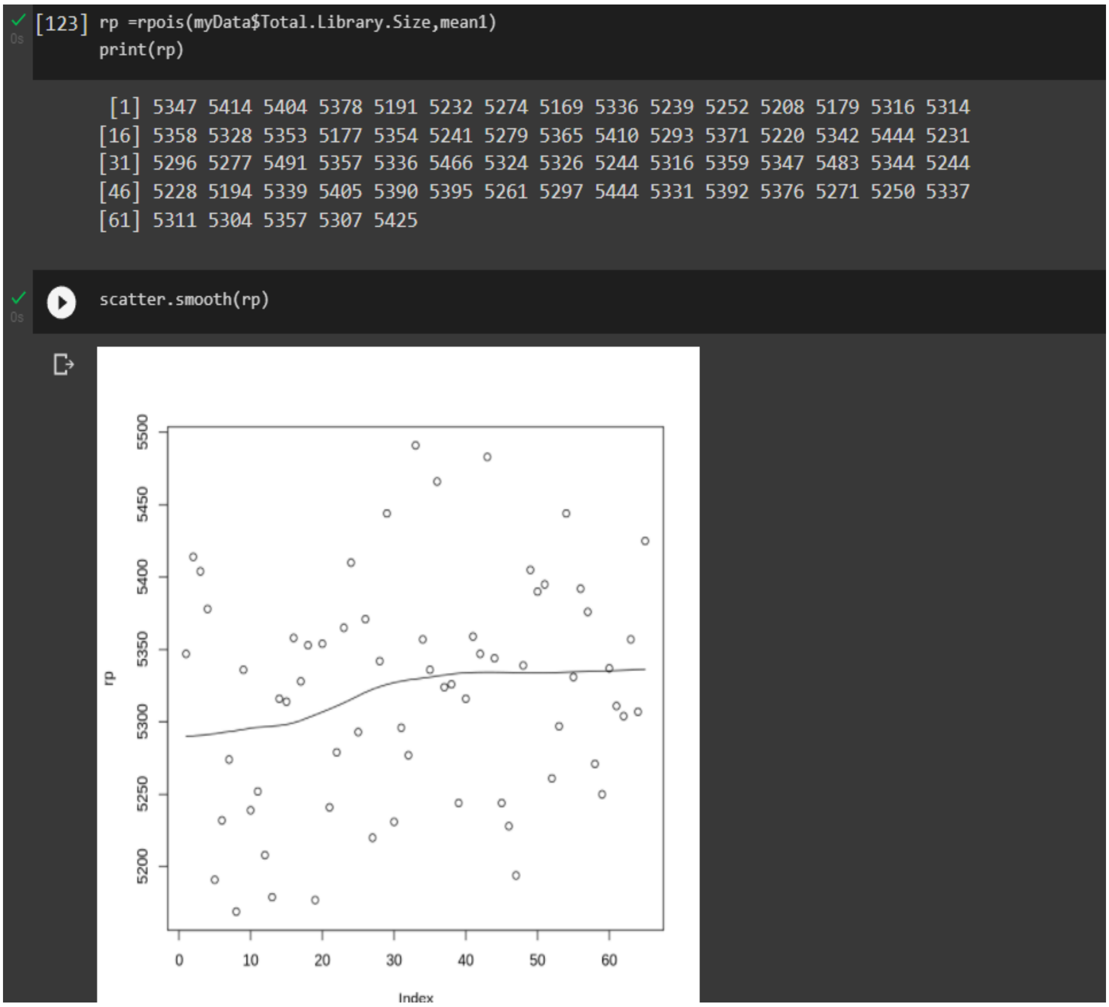
```
[138] R1 = rbinom((dt$Total.Library.Size),1,P)
      print(R1)

 [1] 0 1 0 0 1 0 1 0 1 0 0 0 0 0 1 1 0 0 0 1 0 1 0 1 1 1 1 0 1 1 0 1 0 0 1 0 0 0
[39] 0 0 0 0 0 0 0 1 0 1 0 0 1 0 1 1 1 1 0 0 1 1 1 1 1 1 0
```

```
scatter.smooth(R1)
```

2. Poisson model: Poisson model is used for regression type analysis. Poisson model is extensively used for predictive analysis. This model can be used where there are multiple possible outcomes. This model comes under discrete probability model. Fractional numbers cannot be used in regression model as a response variable. Examples, Number of lectures in a month, Basketball match wins.

$$f(x) = \frac{\lambda^x e^{-\lambda}}{x!} \quad where\ x = 0, 1, 2, 3, \ldots$$

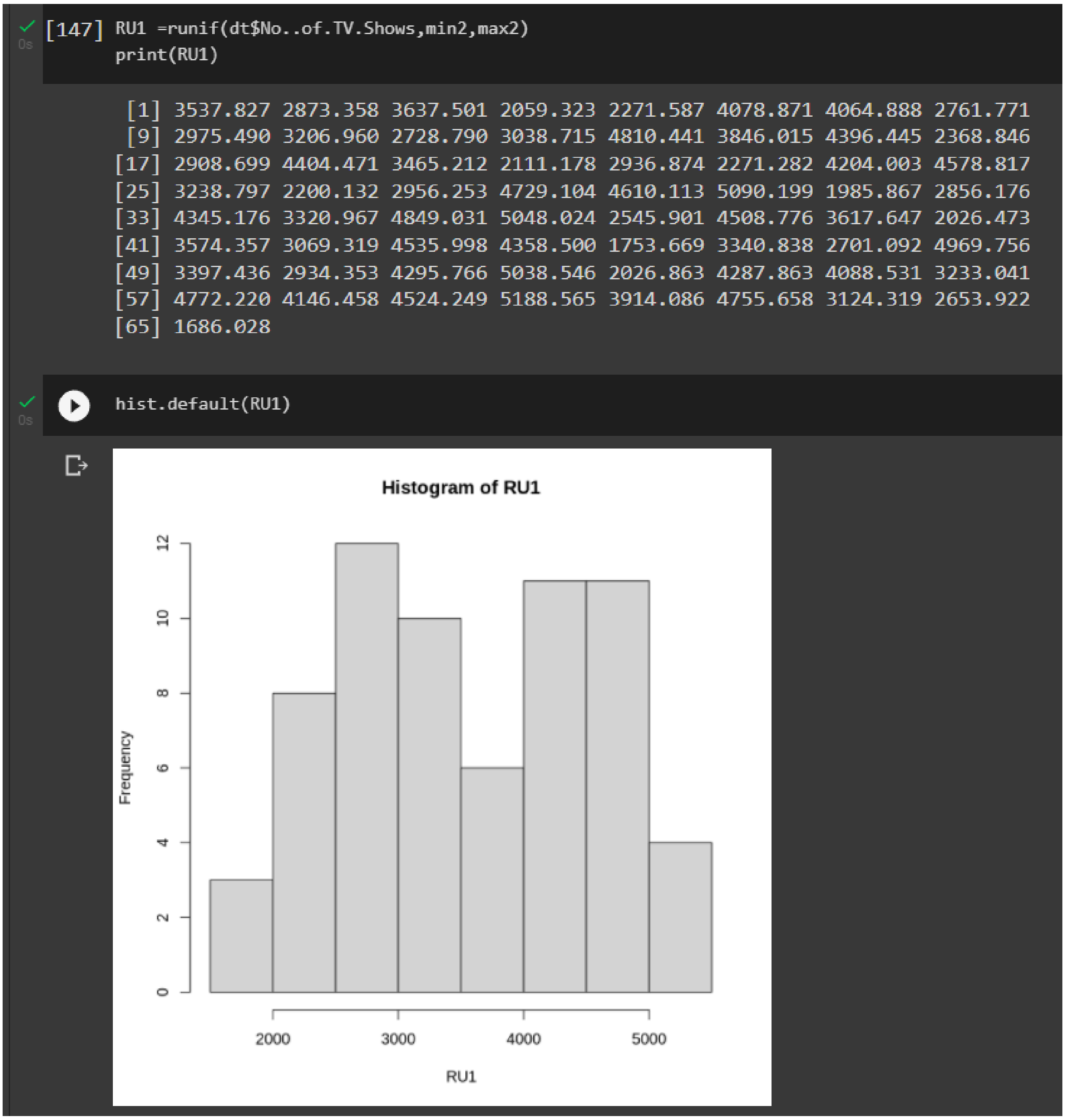


```
[123] rp =rpois(myData$Total.Library.Size,mean1)
      print(rp)

 [1] 5347 5414 5404 5378 5191 5232 5274 5169 5336 5239 5252 5208 5179 5316 5314
[16] 5358 5328 5353 5177 5354 5241 5279 5365 5410 5293 5371 5220 5342 5444 5231
[31] 5296 5277 5491 5357 5336 5466 5324 5326 5244 5316 5359 5347 5483 5344 5244
[46] 5228 5194 5339 5405 5390 5395 5261 5297 5444 5331 5392 5376 5271 5250 5337
[61] 5311 5304 5357 5307 5425
```

```
scatter.smooth(rp)
```

3. Uniform model: In this type of model all outcomes are equally likely to occur. For example, a coin tossed, a deck of cards both have uniform model in them. A coin tossed has equal probability of getting heads and tails. A deck of cards has equal probability of drawing a spade, a diamond, a heart, a club. Uniform distribution can be visualised as a straight line since, all the events are equally likely to occur. Uniform distribution is used as null hypothesis in hypothesis testing.



```
[147] RU1 =runif(dt$No..of.TV.Shows,min2,max2)
      print(RU1)
```

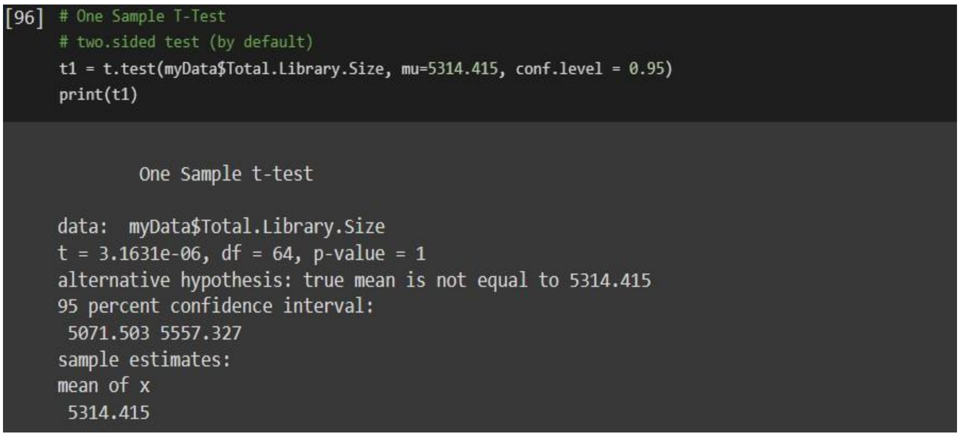
```
 [1] 3537.827 2873.358 3637.501 2059.323 2271.587 4078.871 4064.888 2761.771
 [9] 2975.490 3206.960 2728.790 3038.715 4810.441 3846.015 4396.445 2368.846
[17] 2908.699 4404.471 3465.212 2111.178 2936.874 2271.282 4204.003 4578.817
[25] 3238.797 2200.132 2956.253 4729.104 4610.113 5090.199 1985.867 2856.176
[33] 4345.176 3320.967 4849.031 5048.024 2545.901 4508.776 3617.647 2026.473
[41] 3574.357 3069.319 4535.998 4358.500 1753.669 3340.838 2701.092 4969.756
[49] 3397.436 2934.353 4295.766 5038.546 2026.863 4287.863 4088.531 3233.041
[57] 4772.220 4146.458 4524.249 5188.565 3914.086 4755.658 3124.319 2653.922
[65] 1686.028
```

```
hist.default(RU1)
```

4. HYPOTHESIS TESTING: hypothesis testing is a technique by which data analyst test assumption for chosen parameter or variable. Analyst can use different techniques to analyse the data. These techniques depend on type and structure of the data. Hypothesis testing is used to assess plausibility of sample data. The test provides evidence concerning le T-Test is used to test the statistical difference between a sample mean and a known or assumed/hypothesized value of the mean in the population.

- Syntax by which we can find one-sample t-test is t.test(y, mu = 0). Here, x = variable name.
- There are three distinct types of hypotheses testing we have applied on this data.

1. **one sample t-test :** The One-Sample T-Test is used to test the statistical difference between a sample mean and a known or assumed/hypothesized value of the mean in the population.

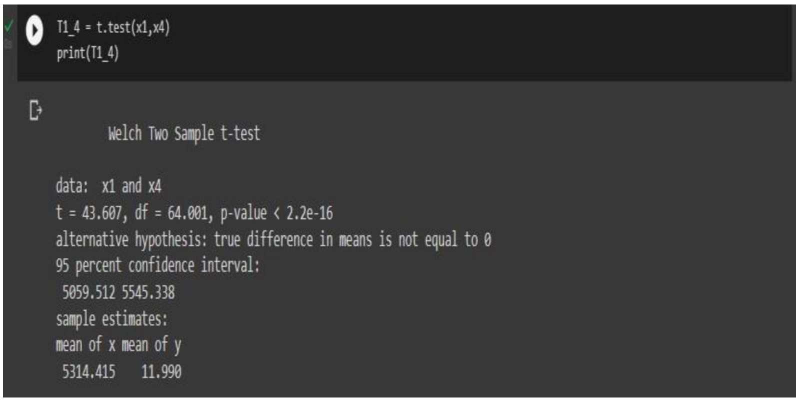


```
[96] # One Sample T-Test
     # two.sided test (by default)
     t1 = t.test(myData$Total.Library.Size, mu=5314.415, conf.level = 0.95)
     print(t1)


        One Sample t-test

data:  myData$Total.Library.Size
t = 3.1631e-06, df = 64, p-value = 1
alternative hypothesis: true mean is not equal to 5314.415
95 percent confidence interval:
 5071.503 5557.327
sample estimates:
mean of x 
 5314.415 
```

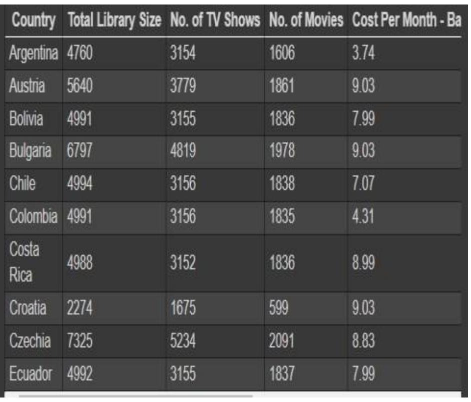
2. **two sampled t-test:** The One-Samp and mu is set equal to the mean specified by the null hypothesis.



```
T1_4 = t.test(x1,x4)
print(T1_4)


        Welch Two Sample t-test

data:  x1 and x4
t = 43.607, df = 64.001, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 5059.512 5545.338
sample estimates:
mean of x mean of y 
 5314.415    11.990 
```



Country	Total Library Size	No. of TV Shows	No. of Movies	Cost Per Month - Ba
Argentina	4760	3154	1606	3.74
Austria	5640	3779	1861	9.03
Bolivia	4991	3155	1836	7.99
Bulgaria	6797	4819	1978	9.03
Chile	4994	3156	1838	7.07
Colombia	4991	3156	1835	4.31
Costa Rica	4988	3152	1836	8.99
Croatia	2274	1675	599	9.03
Czechia	7325	5234	2091	8.83
Ecuador	4992	3155	1837	7.99

3. correlation test: Correlation can help us to figure out how much 2 or multiple variables are related to each other. For example, correlation test can be useful to figure out relation between type of credit card and amount in the bank account, or the electricity bill is corelated with the amount you use. The correlation coefficient is generally between –1 & +1.

Pearson Correlation Testing in R

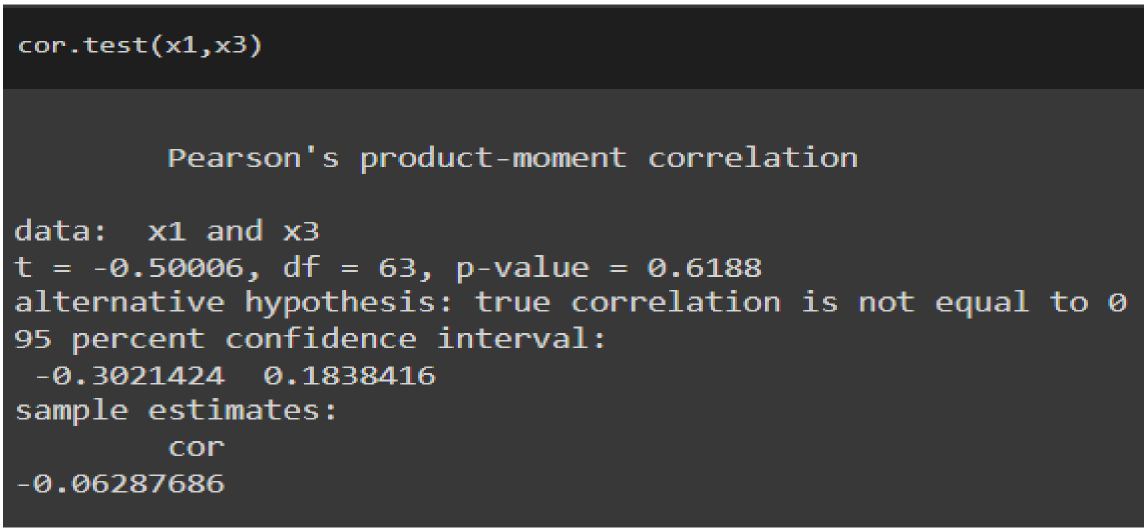
There 2 Type correlation:

1.1 Parametric Correlation

Pearson correlation (r) : Pearson correlation deals with linear dependencies. Two dependencies can be measured using Pearson test (X and Y).

1.2 Non-Parametric Correlation:

Kendall(tau) and Spearman(rho): Kendall(tau) and Spearman(rho) are rank based coefficient.



```
cor.test(x1,x3)
```

```

        Pearson's product-moment correlation

data:  x1 and x3
t = -0.50006, df = 63, p-value = 0.6188
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.3021424  0.1838416
sample estimates:
        cor 
-0.06287686 
```

5. Conclusion and Observations

By conducting research on this dataset, we come no. of conclusions

- This correlation graph is quite obvious the No of Library Size, TV Shows and Movies are mostly correlated.
- The Cost Per month of Basic, Standard and Premium is mostly correlated. This is because these feature columns relate to each other
- The average library size is **5314** with an average of **3519 TV Shows** and **1795 Movies**
- The content library size varies between **2274 and 7425**, which indicates that there is a significant difference in what content a Netflix customer can see based on where they are accessing Netflix from.
- **75% of customers** have access to a library of at least **4948 TV Shows and Movies**.
- Based on the data we can say that the majority of Netflix's content in any given country consists of TV Shows.
- The standard deviation in TV Shows available is 723, while the standard deviation in Movies available is 327
- Therefore, Movies available to users in different countries vary by a smaller margin than TV Shows available to users in different countries.
- TV Shows are the main differentiator in terms of content between each country.
- Costliest countries are mostly European countries - Switzerland, Liechtenstein, Italy, Ireland, Iceland, and France.
- Least Subscription fee countries are - Turkey, India, Argentia, Columbia, Brail.

REFERENCES

[1] https://www.simplilearn.com/data-analysis-methods-process-types-article

[2] https://study.com/learn/lesson/what-is-mode.html

[3]https://www.statisticshowto.com/probability-and-statistics/statistics-definitions/mean-median-mode/

[4] https://monkeylearn.com/blog/data-analysis-examples/

[5] https://www.coursera.org/articles/what-is-data-analysis-with-examples

[6] https://www.investopedia.com/terms/h/hypothesistesting.asp

[7] https://study.dbs.ie/2122/msc-data/B9DA101/u3/index.html#/

[8] https://www.geeksforgeeks.org/

[9] https://www.kaggle.com/prasertk/netflix-subscription-price-in-different-countries

[10] https://www.geeksforgeeks.org/t-test-approach-in-r-programming/