Actvidad 5

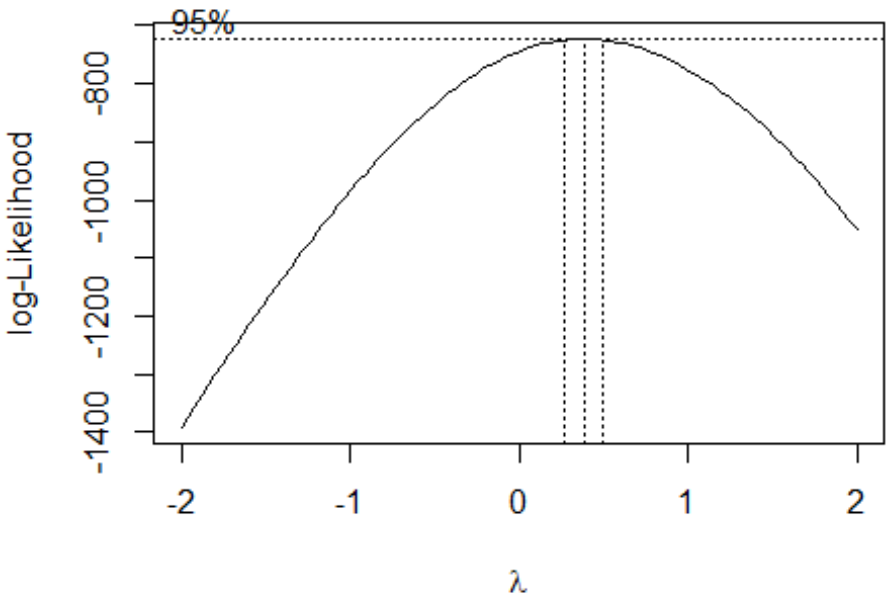
Saúl Francisco Vázquez del Río

2024-08-14

```
df=read.csv("C:\\Users\\saulv\\OneDrive\\Escritorio\\Septimo
semestre\\mc-donalds-menu.csv") #Leer la base de datos
```

1. Utiliza la transformación Box-Cox. Utiliza el modelo exacto y el aproximado de acuerdo con las sugerencias de Box y Cox para la transformación

```
Prote = df$Protein
library(MASS)
bc<-boxcox((Prote+1)~1)
```

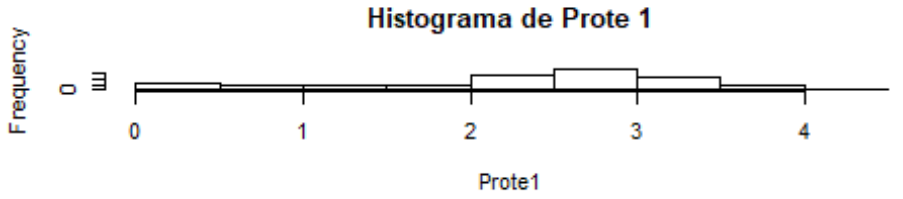


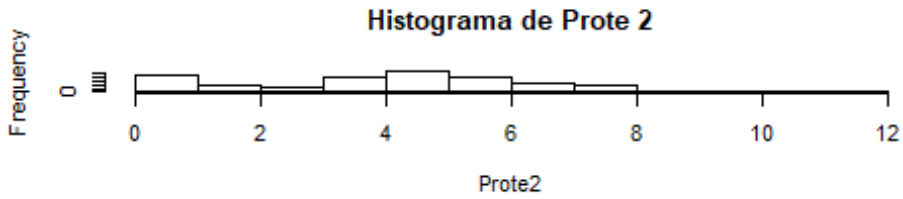

```
l=bc$x[which.max(bc$y)]
print(l)

## [1] 0.3838384
```

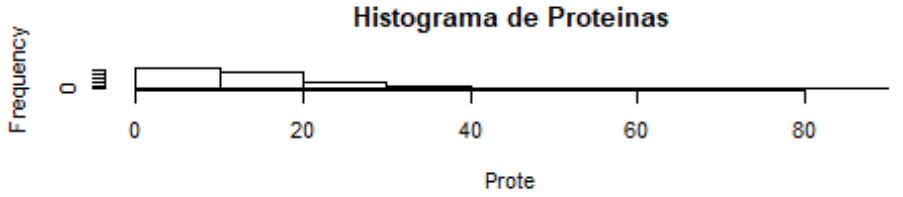
2.Escribe las ecuaciones de los modelos encontrados. $\log(x+1)$

$$\frac{x^{\lambda}-1}{\lambda}$$

```r
Prote1=log(Prote + 1)
Prote2=((Prote+1)^l-1)/l
par(mfrow=c(3,1))
hist(Prote1,col=0,main="Histograma de Prote 1")
hist(Prote2,col=0,main="Histograma de Prote 2")
hist(Prote,col=0,main="Histograma de Proteinas")
```








```r
library(nortest)
D=ad.test(Prote)
D$p.value
```

```
## [1] 8.515383e-12
```

```r
D=ad.test(Prote1)
D$p.value
```

```
## [1] 3.7e-24
```

```r
D=ad.test(Prote2)
D$p.value
```

```
## [1] 1.831193e-08
```

```r
library(e1071)
print("Original")
```

```
## [1] "Original"
```

```r
summary(Prote)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    0.00    4.00   12.00   13.34   19.00   87.00
```

```
print("Curtosis")
```

```
## [1] "Curtosis"
```

```
kurtosis(Prote)
```

```
## [1] 5.7955
```

```
print("Sesgo")
```

```
## [1] "Sesgo"
```

```
skewness(Prote)
```

```
## [1] 1.561741
```

```
library(e1071)
print("Transformacion 1")
```

```
## [1] "Transformacion 1"
```

```
summary(Prote1)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   0.000   1.609   2.565   2.230   2.996   4.477
```

```
print("Curtosis")
```

```
## [1] "Curtosis"
```

```
kurtosis(Prote1)
```

```
## [1] -0.4185221
```

```
print("Sesgo")
```

```
## [1] "Sesgo"
```

```
skewness(Prote1)
```

```
## [1] -0.7992368
```

```
library(e1071)
print("Transformacion 2")
```

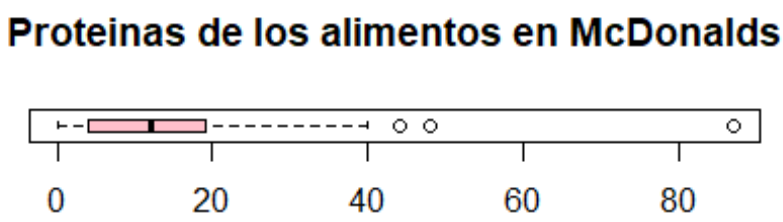
```
## [1] "Transformacion 2"
```

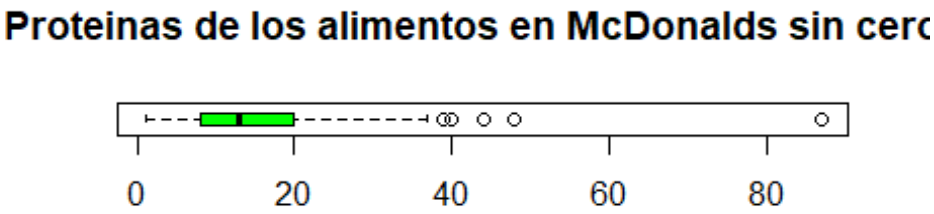
```
summary(Prote2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   0.000   2.227   4.368   4.021   5.622  11.923
```

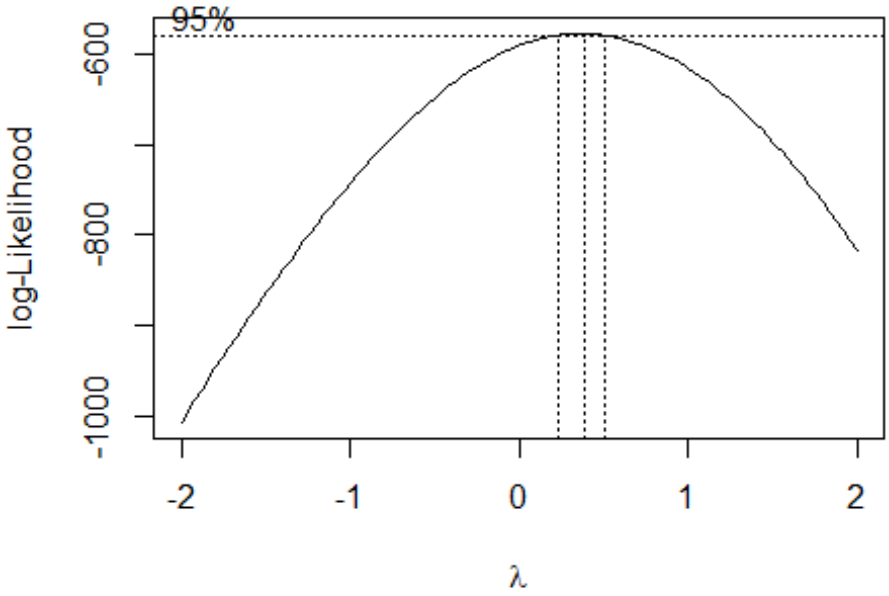
```
print("Curtosis")
```

```
## [1] "Curtosis"

kurtosis(Prote2)

## [1] -0.5104494

print("Sesgo")

## [1] "Sesgo"

skewness(Prote2)

## [1] -0.1145447

M2 <- Prote[Prote > 0]
par(mfrow=c(2,1))
boxplot(Prote, horizontal = TRUE, col="pink", main="Proteinas de los
alimentos en McDonalds")
boxplot(M2, horizontal = TRUE, col="green", main="Proteinas de los
alimentos en McDonalds sin ceros")
```

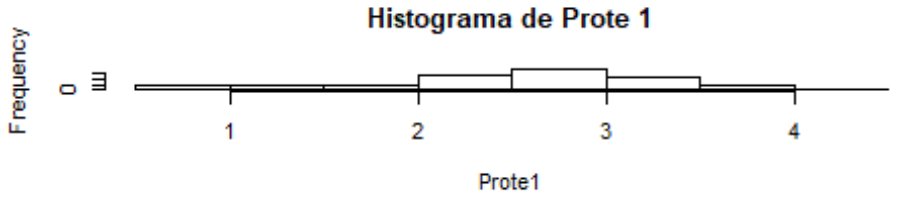


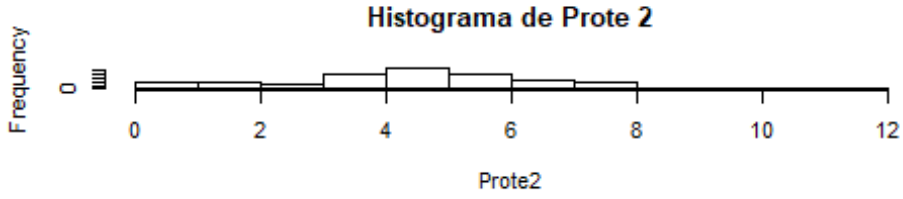


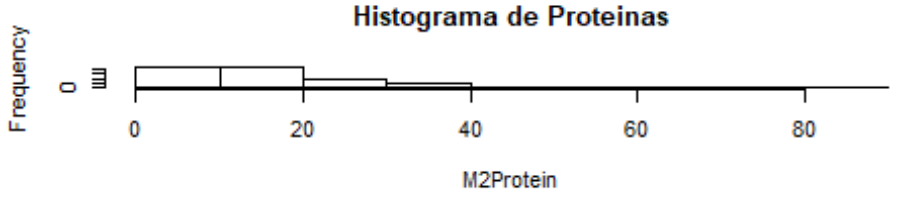

```
library(MASS)
M2 = subset(df, Protein>0)
M2Protein <- M2$Protein
bc<-boxcox((M2Protein+1)~1)
```




```
l=bc$x[which.max(bc$y)]

print(l)

## [1] 0.3838384

Prote1=log(M2Protein+1)
Prote2=((M2Protein+1)^l-1)/l
par(mfrow=c(3,1))
hist(Prote1,col=0,main="Histograma de Prote 1")
hist(Prote2,col=0,main="Histograma de Prote 2")
hist(M2Protein,col=0,main="Histograma de Proteinas")
```








```
library(nortest)
D0=ad.test(M2Protein)
D1=ad.test(Prote1)
D2=ad.test(Prote2)

print("D0")

## [1] "D0"

print(D0)

##
##  Anderson-Darling normality test
##
## data:  M2Protein
## A = 4.2129, p-value = 1.685e-10

print("D1")

## [1] "D1"

print(D1)

##
##  Anderson-Darling normality test
##
## data:  Prote1
## A = 5.9646, p-value = 1.034e-14
```

```r
print("D2")
```

```
## [1] "D2"
```

```r
print(D2)
```

```
##
##  Anderson-Darling normality test
##
## data:  Prote2
## A = 1.6613, p-value = 0.000283
```

```r
library(e1071)
m0=round(c(as.numeric(summary(M2Protein)),kurtosis(M2Protein),skewness(M2
Protein),D0$p.value),3)
m1=round(c(as.numeric(summary(Prote1)),kurtosis(Prote1),skewness(Prote1),
D1$p.value),3)
m2=round(c(as.numeric(summary(Prote2)),kurtosis(Prote2),skewness(Prote2),
D2$p.value),3)

m<-as.data.frame(rbind(m0,m1,m2))
row.names(m)=c("Original","Primer modelo","Segundo Modelo")
names(m)=c("Minimo","Q1","Mediana","Media","Q3","Máximo","Curtosis","Sesg
o","Valor p")

print(m)
```

```
##                Minimo    Q1 Mediana  Media     Q3 Máximo Curtosis
Sesgo
## Original        1.000 8.000  13.000 14.884 20.000 87.000    6.814
1.697
## Primer modelo   0.693 2.197   2.639  2.489  3.045  4.477   -0.092 -
0.733
## Segundo Modelo  0.794 3.450   4.569  4.486  5.777 11.923   -0.027
0.004
##                Valor p
## Original             0
## Primer modelo        0
## Segundo Modelo       0
```

```r
library(VGAM)
```

```
## Cargando paquete requerido: stats4
```

```
## Cargando paquete requerido: splines
```

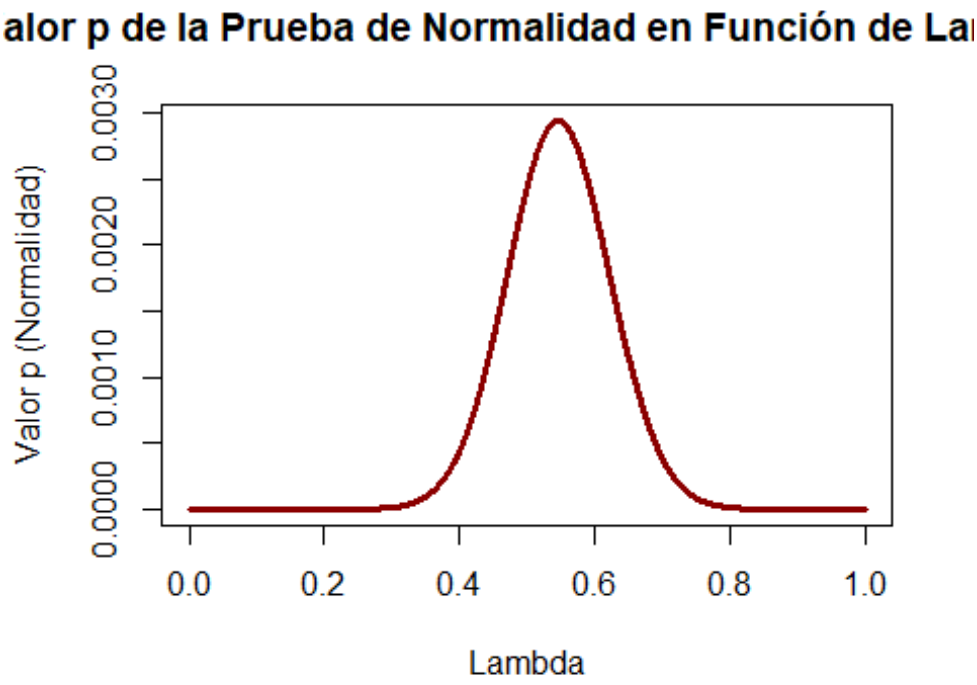
```r
Prote3<- yeo.johnson(M2Protein, lambda = 1)
print(Prote3)
```

```
##   [1]  5.2955787  5.4612587  4.7615655  5.9281928  5.9281928
6.6260599
##   [7]  5.6216488  5.6216488  5.7771703  5.7771703  4.1568554
```

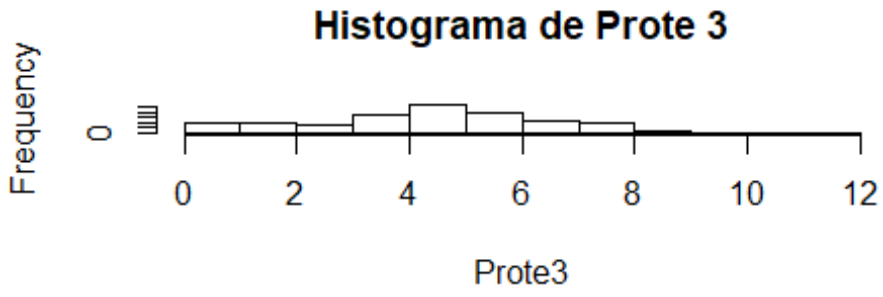
```
4.1568554
##  [13]  5.4612587  5.4612587  5.4612587  5.4612587  5.2955787
5.2955787
##  [19]  6.4932972  5.6216488  5.7771703  4.1568554  5.7771703
5.9281928
##  [25]  7.1287843  7.1287843  7.4801104  6.8827676  6.8827676
6.6260599
##  [31]  6.6260599  7.8128150  7.8128150  7.7038246  7.7038246
3.4499091
##  [37]  4.9463384  4.3678356  0.7941071  2.8930904  2.5771963
2.5771963
##  [43]  6.3573496  7.1287843  7.9200053  7.9200053  7.0070394
8.9988974
##  [49]  4.3678356  4.9463384  6.3573496  8.1292837  6.0750423
6.7558264
##  [55]  6.0750423  6.0750423  6.0750423  6.3573496  6.8827676
7.8128150
##  [61]  8.2315091  7.3652046  7.8128150  7.8128150  8.2315091
5.9281928
##  [67]  4.7615655  6.0750423  5.7771703  4.7615655  7.3652046
7.8128150
##  [73]  6.7558264  7.1287843  6.7558264  7.2481328  6.2180085
6.7558264
##  [79]  3.6998078  4.5690368  6.0750423  8.6257262 11.9231668
4.9463384
##  [85]  3.6998078  6.4932972  7.0070394  2.8930904  6.2180085
6.7558264
##  [91]  4.7615655  5.1241251  4.7615655  5.1241251  4.9463384
5.1241251
##  [97]  1.3665522  2.2269170  2.8930904  0.7941071  0.7941071
2.2269170
## [103]  1.3665522  1.3665522  1.3665522  0.7941071  3.4499091
3.1822528
## [109]  2.8930904  1.3665522  1.8302649  2.2269170  0.7941071
3.4499091
## [115]  3.6998078  1.3665522  1.8302649  2.2269170  0.7941071
0.7941071
## [121]  0.7941071  3.6998078  4.1568554  4.9463384  3.6998078
4.1568554
## [127]  4.9463384  3.6998078  4.1568554  4.9463384  3.6998078
4.1568554
## [133]  4.9463384  3.6998078  4.3678356  4.9463384  3.9347419
4.3678356
## [139]  5.1241251  3.9347419  4.3678356  5.1241251  3.9347419
4.3678356
## [145]  5.1241251  3.9347419  4.3678356  5.1241251  3.9347419
4.3678356
## [151]  5.1241251  3.9347419  4.5690368  5.1241251  4.1568554
4.5690368
## [157]  5.2955787  3.9347419  4.3678356  5.1241251  3.9347419
```

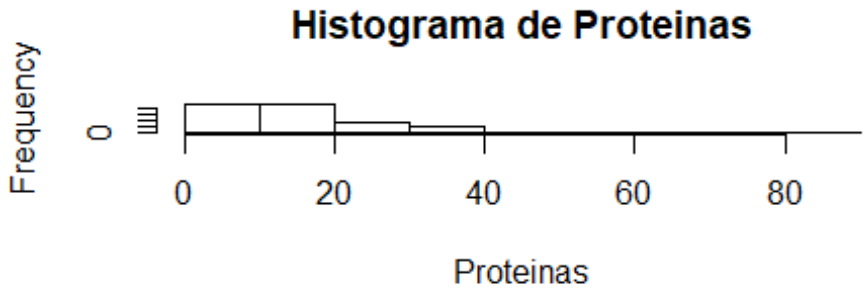
```
4.5690368
## [163]  5.2955787  4.1568554  4.7615655  5.2955787  4.3678356
4.7615655
## [169]  5.6216488  0.7941071  0.7941071  1.3665522  0.7941071
0.7941071
## [175]  1.3665522  0.7941071  0.7941071  1.3665522  0.7941071
0.7941071
## [181]  1.3665522  0.7941071  0.7941071  1.3665522  3.4499091
3.6998078
## [187]  4.7615655  3.4499091  3.9347419  4.7615655  3.4499091
3.6998078
## [193]  4.5690368  3.4499091  3.9347419  4.7615655  3.1822528
3.6998078
## [199]  4.1568554  3.1822528  3.6998078  4.1568554  3.4499091
3.6998078
## [205]  4.3678356  1.3665522  1.8302649  2.2269170  1.8302649
2.2269170
## [211]  2.5771963  1.3665522  1.8302649  2.2269170  4.1568554
4.7615655
## [217]  5.4612587  4.3678356  4.9463384  5.4612587  4.3678356
4.9463384
## [223]  5.6216488  4.7615655  5.4612587  4.5690368  5.7771703
3.6998078
## [229]  4.3678356  4.9463384  3.4499091  5.9281928  3.9347419
```

```
library(VGAM)
lp <- seq(0,1,0.001) # Valores de lambda propuestos
nlp <- length(lp)
n=length(M2Protein)
D <- matrix(as.numeric(NA),ncol=2,nrow=nlp)
d <-NA
for (i in 1:nlp){
d= yeo.johnson(M2Protein, lambda = lp[i])
p=ad.test(d)
D[i,]=c(lp[i],p$p.value)}

N <- as.data.frame(D)
colnames(N) <- c("Lambda", "Valor-p")

plot(N$Lambda, N$`Valor-p`, type="l", col="darkred", lwd=3,
     xlab="Lambda", ylab="Valor p (Normalidad)",
     main="Valor p de la Prueba de Normalidad en Función de Lambda")
```




```
G=data.frame(subset(N,N$`Valor-p`==max(N$`Valor-p`)))
print(G)

##     Lambda   Valor.p
## 547  0.546 0.0029409

par(mfrow=c(2,1))
hist(Prote3,col=0,main="Histograma de Prote 3")
hist(M2Protein,col=0,main="Histograma de Proteinas",xlab="Proteinas")
```






```
library(e1071)
summary(Prote3)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.7941  3.4499  4.5690  4.4864  5.7772 11.9232
```

```
print("Curtosis")
```

```
## [1] "Curtosis"
```

```
kurtosis(Prote3)
```

```
## [1] -0.02734962
```

```
print("Sesgo")
```

```
## [1] "Sesgo"
```

```
skewness(Prote3)
```

```
## [1] 0.004282062
```

Define la mejor transformación de los datos de acuerdo a las características de los modelos que encontraste. Toma en cuenta los criterios del inciso anterior para analizar normalidad y la economía del modelo.

La mejor transformación en esta actividad para mi fue la de Yeo Johnson porque la p tiene un valor mayor en la prueba normalidad

Concluye sobre las ventajas y desventajas de los modelos de Box Cox y de Yeo Johnson.

Box Cox

Desventajas: No puede manejar datos que sean ceros o negativos y no puede ser considerada si los datos no son simétricos.

Ventajas: Es ideal si los datos son positivos y si esta puede a llegar a ser más simple si sus datos no nos son negativos.

Yeo Johnson

Desventajas: Es más compleja que Box Cox y podría necesitar ajustes si los datos no son uniformes

Ventajas: Puede manejar datos positivos y negativos, y es más fácil de usar cuando los datos son uniformes

Analiza las diferencias entre la transformación y el escalamiento de los datos: Escribe al menos 3 diferencias entre lo que es la transformación y el escalamiento de los datos

1. Transformación: Esta modifica la distribución de los datos para hacer a esta más cercana a una distribución normal.

Escalamiento: Esta escala los datos para que estos se encuentren en un rango específico.

2. Transformación: Ésta puede estabilizar la varianza de los datos

Escalamiento: Ajusta la magnitud de los datos y no afecta a la varianza

3. Transformación: Esta se le dificulta trabajar con datos negativos

Escalamiento: Esta se puede aplicar a cualquier rango de datos incluso si estos son negativos

Indica cuándo es necesario utilizar cada uno

Transformacion: Se usa cuando los datos no tiene una distribución normal

Escalamiento: Se usa cuando se quieren normalizar los datos