Limpieza y análisis de datos

David Sun

12/5/2022

Índice

1. Descripción del dataset

Se elige un dataset del repositorio Kaggle, en especial llamado **Avocado Prices** (link).

Este dataset nos proviene información sobre la venta minorista en las diferentes regiones de EE.UU de los aguacates de la variedad Hass, sin incluir las otras variedades entre el 4 de enero de 2015 hasta el 25 de marzo de 2018.

```
avocado <- read.csv("avocado.csv")
head(avocado, n = 3)
```

```
##   X       Date AveragePrice Total.Volume   X4046     X4225  X4770 Total.Bags
## 1 0 2015-12-27         1.33     64236.62 1036.74  54454.85  48.16    8696.87
## 2 1 2015-12-20         1.35     54876.98  674.28  44638.81  58.33    9505.56
## 3 2 2015-12-13         0.93    118220.22  794.70 109149.67 130.50    8145.35
##   Small.Bags Large.Bags XLarge.Bags         type year region
## 1    8603.62      93.25           0 conventional 2015 Albany
## 2    9408.07      97.49           0 conventional 2015 Albany
## 3    8042.21     103.14           0 conventional 2015 Albany
```

Revisamos los tipos de datos:

```
sapply(avocado, class)
```

```
##            X         Date AveragePrice Total.Volume        X4046        X4225
##    "integer"  "character"    "numeric"    "numeric"    "numeric"    "numeric"
##        X4770   Total.Bags   Small.Bags   Large.Bags  XLarge.Bags         type
##    "numeric"    "numeric"    "numeric"    "numeric"    "numeric"  "character"
##         year       region
##    "integer"  "character"
```

Vemos un resumen de las variables numéricas:

```
summary(avocado[, c("AveragePrice", "Total.Volume", "X4046", "X4225",
                    "X4770", "Total.Bags", "Small.Bags", "Large.Bags",
                    "XLarge.Bags")])
```

```
##   AveragePrice    Total.Volume          X4046             X4225
##  Min.   :0.440   Min.   :      85   Min.   :       0   Min.   :       0
##  1st Qu.:1.100   1st Qu.:   10839   1st Qu.:     854   1st Qu.:    3009
##  Median :1.370   Median :  107377   Median :    8645   Median :   29061
##  Mean   :1.406   Mean   :  850644   Mean   :  293008   Mean   :  295155
##  3rd Qu.:1.660   3rd Qu.:  432962   3rd Qu.:  111020   3rd Qu.:  150207
##  Max.   :3.250   Max.   :62505647   Max.   :22743616   Max.   :20470573
##      X4770           Total.Bags         Small.Bags         Large.Bags
##  Min.   :      0   Min.   :       0   Min.   :       0   Min.   :      0
##  1st Qu.:      0   1st Qu.:    5089   1st Qu.:    2849   1st Qu.:    127
##  Median :    185   Median :   39744   Median :   26363   Median :   2648
##  Mean   :  22840   Mean   :  239639   Mean   :  182195   Mean   :  54338
##  3rd Qu.:   6243   3rd Qu.:  110783   3rd Qu.:   83338   3rd Qu.:  22029
##  Max.   :2546439   Max.   :19373134   Max.   :13384587   Max.   :5719097
##   XLarge.Bags
```

```
##  Min.   :     0.0
##  1st Qu.:     0.0
##  Median :     0.0
##  Mean   :  3106.4
##  3rd Qu.:   132.5
##  Max.   :551693.7
```

Vamos a repasar las diferentes columnas:

- X: índice que se repite cada 51 observaciones por fecha, región, creo que no nos aporta nada y se borrará en este análisis.
- Date: la fecha de la observación.
- AveragePrice: el precio medio de venta de un aguacate.
- Total.Volume: número total de aguacates vendidos.
- X4046: número de ventas de aguacates pequeños clasificados como 4046 según el Product Lookup codes (PLU).
- X4225: número de ventas de aguacates grandes clasificados como 4225 según el PLU.
- X4770: número de ventas de aguacates extra grandes clasificados como 4770 según el PLU.
- Total.Bags: Número total de bolsas vendidas (cada bolsa hay múltiples aguacates).
- Small.Bags: bolsas vendidas pequeñas.
- Large.Bags: bolsas vendidas grandes.
- XLarge.Bags: bolsas vendidas extra grandes.
- type: orgánico o convencional.
- year: el año.
- region: la región que que se ha hecho la observación.

Para más información sobre los criterios de clasificación de los tamaños de los productos podemos entrar en **International Federation for Produce Standards (IFPS)** link.

Con este dataset podemos obtener información sobre la evolución de los precios de los aguacates y también el patrón de consumo en las diferentes regiones de EEUU.

2. Integración y selección de los datos de interés a analizar

Vamos a eliminar la primera columna ya que no podemos extraer ninguna información de ella:

```
avocado = subset(avocado, select = -c(X) )
```

Podriamos eliminar la columna del año y extraer la información de la fecha pero creo que para hacer consultas luego es más fácil, así que vamos a dejar esta columna.

3. Limpieza de los datos

3.1. Valores nulos

Miramos si el dataset tiene algún valor nulo:

```
table(is.na(avocado))
```

```
##
##   FALSE
## 237237
```

3.2. Cambio formato datos

Vamos a cambiar de "character" a "Date" la fecha:

```
avocado$Date <-as.Date(avocado$Date, "%Y-%m-%d")

# Print result
print(class(avocado$Date))
```

```
## [1] "Date"
```

```
print(avocado$Date[0:10])
```

```
##  [1] "2015-12-27" "2015-12-20" "2015-12-13" "2015-12-06" "2015-11-29"
##  [6] "2015-11-22" "2015-11-15" "2015-11-08" "2015-11-01" "2015-10-25"
```

Pasamos a factor la variable type a factor:

```
avocado$type = as.factor(avocado$type)
print(class(avocado$type))
```

```
## [1] "factor"
```

Miramos cuantos factores tiene:

```
levels(avocado$type)
```

```
## [1] "conventional" "organic"
```

3.3. Identifica y gestiona los valores extremos

Hay dos tipo de outliers:

- **Valor centinela**: este tipo de outlier suele ser representación de un valor nulo o desconocido y suele ser un valor extraño en el rango de valores que puede tener la variable por ejemplo -1 como las horas que pasan la gente en redes sociales.
- **Valor atípico propiamente**: es el valor atípico propiamente dicho, puede ser que sea un valor legítimo extremo o un valor extremo que pued ser un error, este valor hay que tratarla o no dependiendo de la situación, ya que puede afectar a los estimadores de las tendencias centrales y de dispersión como la media o la desviación estándar. Una también podemos usar estimadores resistentes a estos valores extremos, como la media winsorizada, media recortada o RIC para sustituir a la desviación estándar.

Hay muchas formas de detectar los valores extremos o outliers, en este caso usamos el médodo del Rango Intercuartílico (RIC), el RIc es la diferencia entre el cuartil 3 (Q3) y el cuartil 1 (Q1). clasificamos los outlier con los siguientes criterios:

- Q3 + RIC * 1.5 > outlier
- Q1 - RIC *1.5 < outlier

Vamos a ver algunas graficas y los outliers:

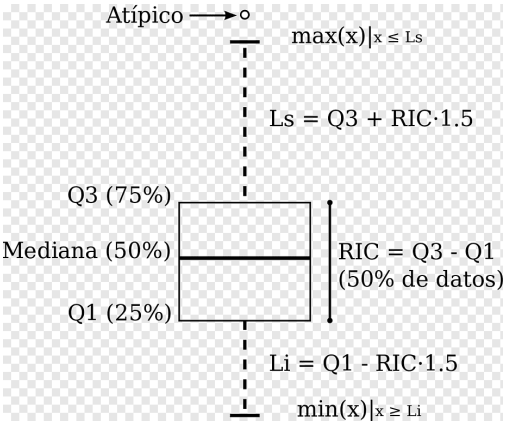
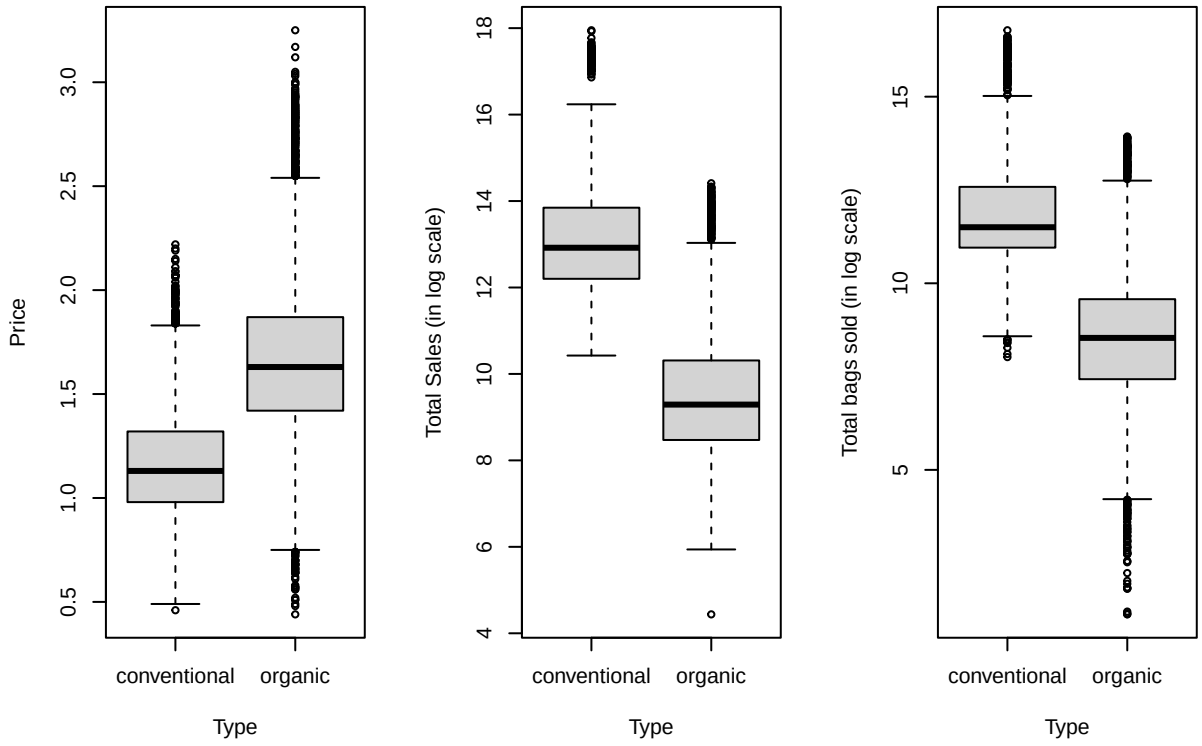


Figura 1: RIC

```r
# subset por tipo
avocado_conv <- avocado[avocado$type == "conventional", ]
avocado_org <- avocado[avocado$type == "organic", ]

par(mfrow=c(1,3))
boxplot(avocado$AveragePrice ~ avocado$type, horizontal=FALSE, xlab="Type", ylab="Price")
boxplot(log(avocado$Total.Volume) ~ avocado$type, horizontal=FALSE,
        xlab="Type", ylab="Total Sales (in log scale)")
boxplot(log(avocado$Total.Bags) ~ avocado$type ,horizontal=FALSE,
        xlab="Type", ylab="Total bags sold (in log scale)")
```




```
# outliers de aguacate convencional
boxplot.stats(avocado_conv$AveragePrice)$out[1:5]
```

```
## [1] 1.86 1.85 2.07 2.07 1.84
```

```
boxplot.stats(avocado_conv$Total.Bags)$out[1:5]
```

```
## [1] 1212707 1209165 1592438 1510267 1081497
```

```
# outliers de aguacate orgánico
boxplot.stats(avocado_org$AveragePrice)$out[1:5]
```

```
## [1] 2.58 2.79 2.66 2.59 2.74
```

```
boxplot.stats(avocado_org$Total.Bags)$out[1:5]
```

```
## [1] 38849.61 38474.55 39813.19 67348.67 38767.30
```

Aunque vemos que hay bastantes outliers, en realidad son valores realistas que realmente son los precios que se ha vendido y las cantidades de estas y no se han de eliminar.

Podemos ver ya que el precio de los aguacates ecológicos es mayor al de los convencionales y que hay menos ventas totales de las orgánicas.

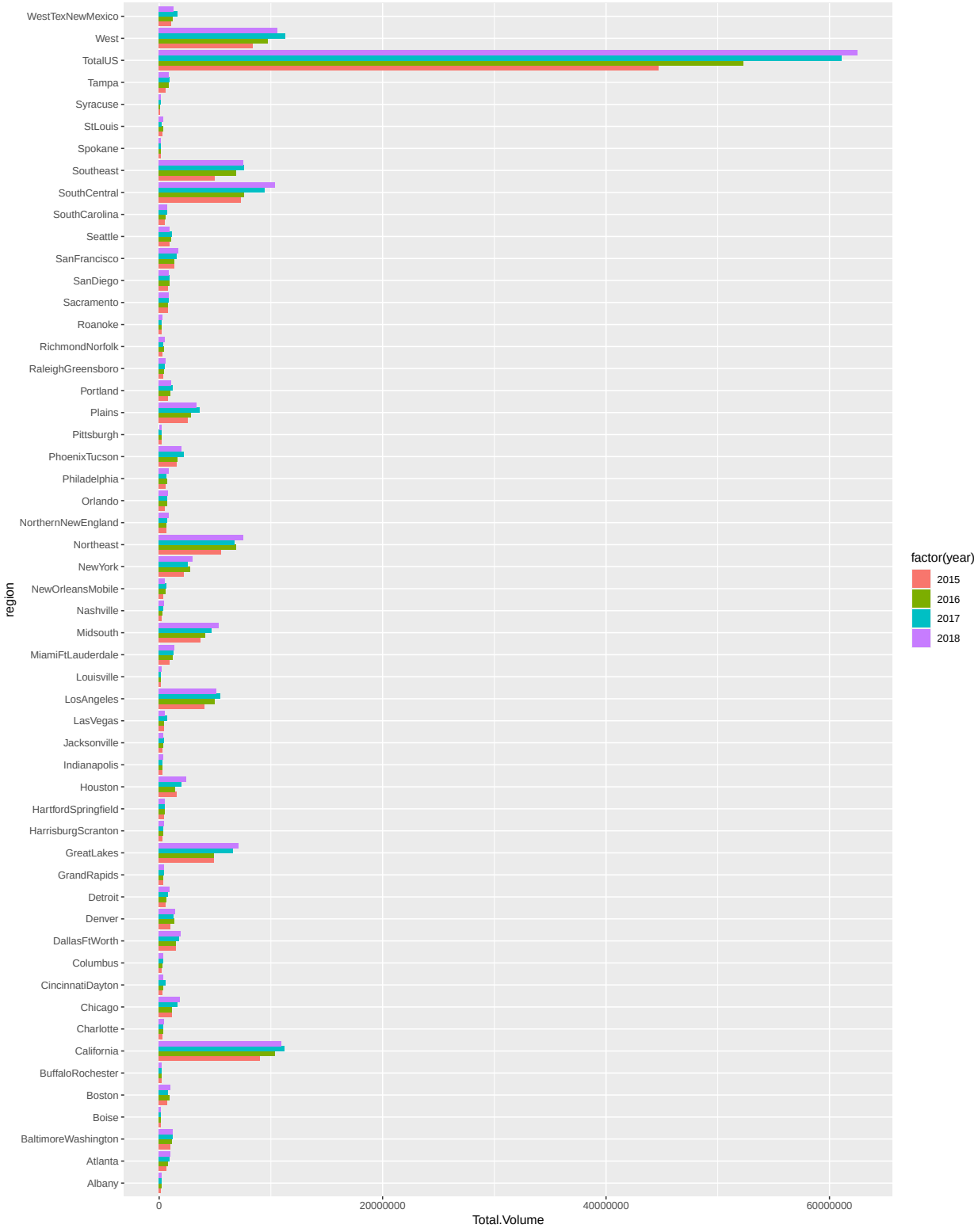
4. Análisis de los datos

Vamos a ver como se ha comportado las ventas y los precios de los aguacates convencionales en las diferentes regiones de los EEUU:

Podemos ver en la siguiente gráfica un aumento del volumen total de vental en todas las regiones en general desde 2015 hasta 2018 aunque el crecimiento desde 2017 hasta 2018 ha sido menor que los años anteriores.

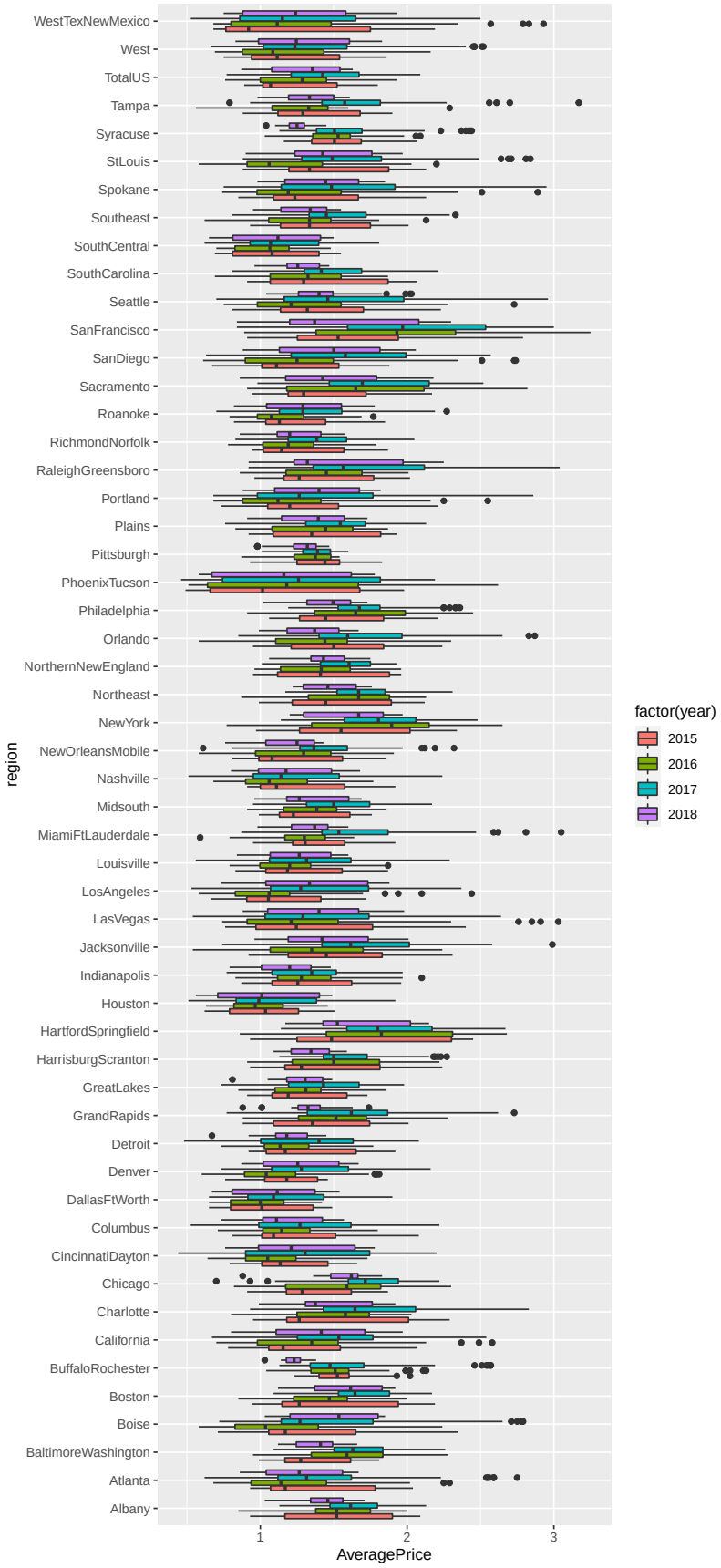
También podemos observar las regiones donde se han consumido más aguacate y los que menos.

```
ggplot(avocado_conv, aes(x = region, y = Total.Volume, fill = factor(year))) +
geom_bar(stat = "identity",position = "dodge") +
coord_flip()
```



Ahora miramos la evolucion de los precios de los aguacates convencionales:

```
ggplot(avocado, aes(y=region, x=AveragePrice, fill = factor(year))) + geom_boxplot()
```

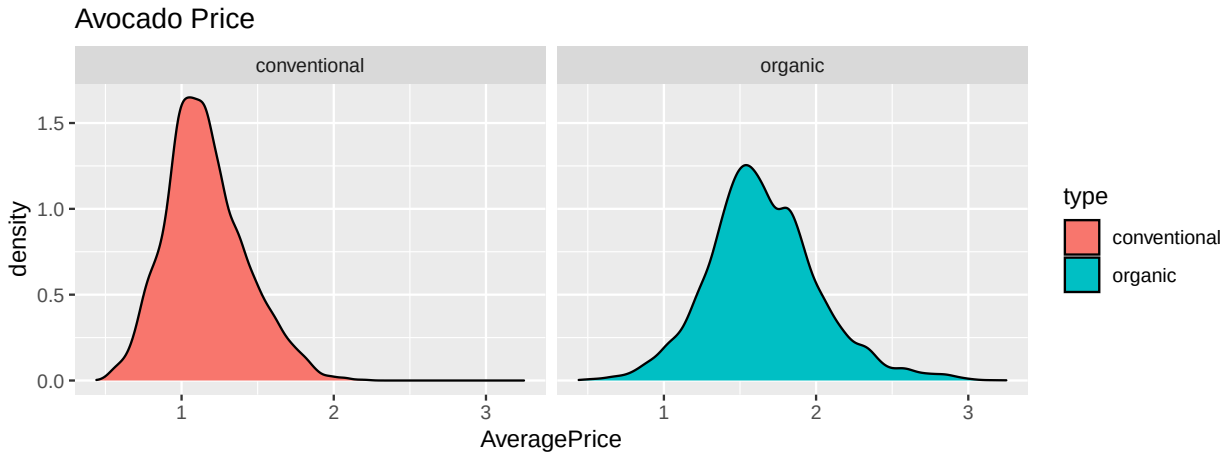

WestTexNewMexico
West
TotalUS
Tampa
Syracuse
StLouis
Spokane
Southeast
SouthCentral
SouthCarolina
Seattle
SanFrancisco
SanDiego
Sacramento
Roanoke
RichmondNorfolk
RaleighGreensboro
Portland
Plains
Pittsburgh
PhoenixTucson
Philadelphia
Orlando
NorthernNewEngland
Northeast
NewYork
NewOrleansMobile
Nashville
Midsouth
MiamiFtLauderdale
Louisville
LosAngeles
LasVegas
Jacksonville
Indianapolis
Houston
HartfordSpringfield
HarrisburgScranton
GreatLakes
GrandRapids
Detroit
Denver
DallasFtWorth
Columbus
CincinnatiDayton
Chicago
Charlotte
California
BuffaloRochester
Boston
Boise
BaltimoreWashington
Atlanta
Albany
region
1
2
3
AveragePrice
factor(year)
2015
2016
2017
2018

Podemos ver que en general en el país no ha subido los precios sino que ha ido fructuando. Vemos que una de las regiones donde el precio subió de forma abrupta fue en San Francisco y en RaleighGreensboro en el año 2017.

4.1. Normalidad y homogeneidad de la varianza

Vamos a estudiar la normalidad de los precios:

```
ggplot(avocado, aes(x=AveragePrice, fill=type)) + geom_density() + facet_wrap(~type) +
ggtitle("Avocado Price")
```




```
# Test normalidad de Kolmogorov-Smirnov aguacate convencional
lillie.test(avocado_conv$AveragePrice)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  avocado_conv$AveragePrice
## D = 0.057243, p-value < 0.00000000000000022
```

El test de Kolmogorov-Smirnov es un contraste bilateral donde:

- H0: los datos proceden de una distribución normal.
- H1: los datos no proceden de una distribución normal.

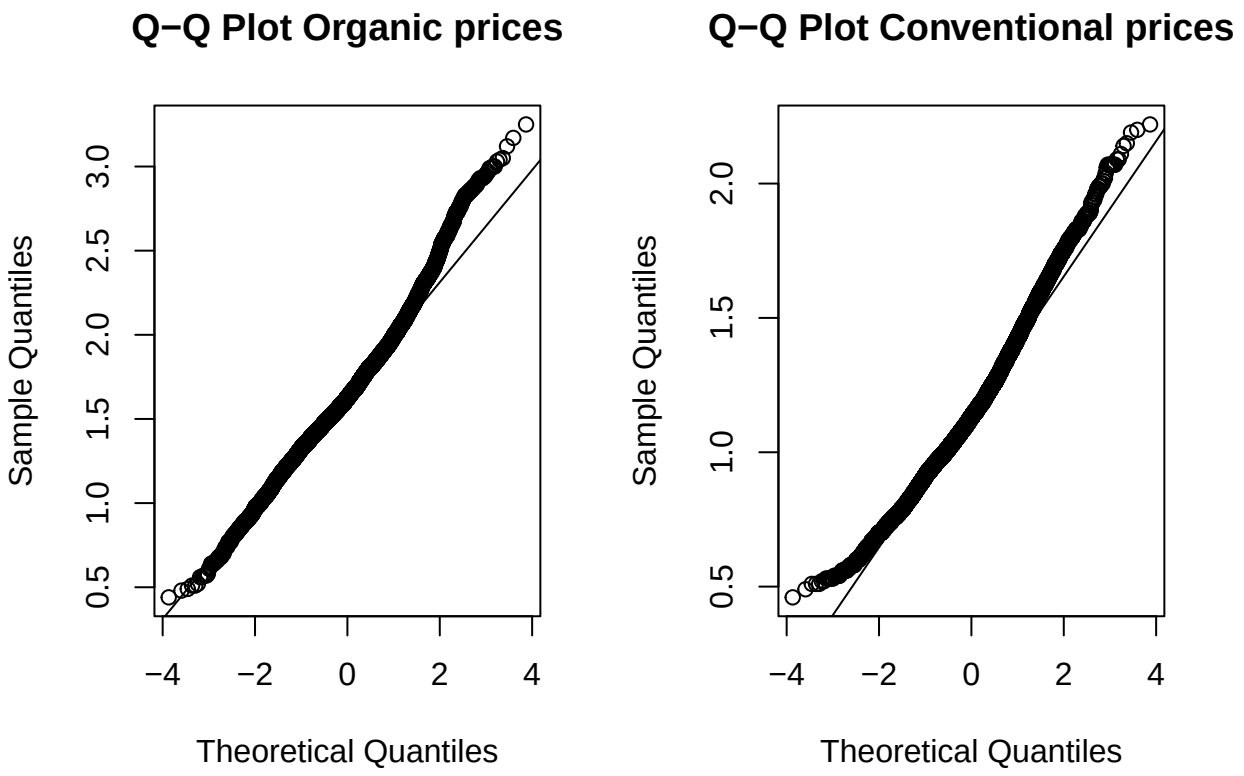
El P valor es la probabilidad de haber obtenido el resultado que hemos obtenido suponiendo que la hipótesis nula H0 es cierta. El valor de alfa predeterminado es de 0.05, si P valor es menor a alfa podemos descartar ya la hipótesis nula, es decir los datos no tienen una distribución normal.

```
# Test normalidad de Kolmogorov-Smirnov aguacate orgánico
lillie.test(avocado_org$AveragePrice)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  avocado_org$AveragePrice
## D = 0.042489, p-value < 0.00000000000000022
```

Podemos ver que el precio de los aguacates orgánicos tampoco son normales. Podemos también complementarlo con la visualización con los QQ plots, donde la linea sería la representación de una distribución normal, como vemos los datos no encajan del todo.

```
par(mfrow=c(1,2))
qqnorm(avocado_org$AveragePrice, main=NULL)
qqline(avocado_org$AveragePrice)
title("Q-Q Plot Organic prices")
qqnorm(avocado_conv$AveragePrice, main=NULL)
qqline(avocado_conv$AveragePrice)
title("Q-Q Plot Conventional prices")
```



Tendremos que usar el Test Fligner-Killeen para el análisis de la homocedasticidad ya que no cumplen con la normalidad:

```
fligner.test(AveragePrice  ~ type, data = avocado)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  AveragePrice by type
## Fligner-Killeen:med chi-squared = 554.62, df = 1, p-value <
## 0.00000000000000022
```

Vemos que la p-valor es estadísticamente significativa, rechazando la hipótesis nula de la igualdad de varianzas.

4.2. Test de rangos y signos

Vamos a emplear el Test de rangos y signos como la prueba no paramétrica de dos muestras independientes:

```
wilcox.test(avocado_org$AveragePrice, avocado_conv$AveragePrice, alternative="greater")
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  avocado_org$AveragePrice and avocado_conv$AveragePrice
## W = 72573720, p-value < 0.00000000000000022
## alternative hypothesis: true location shift is greater than 0
```

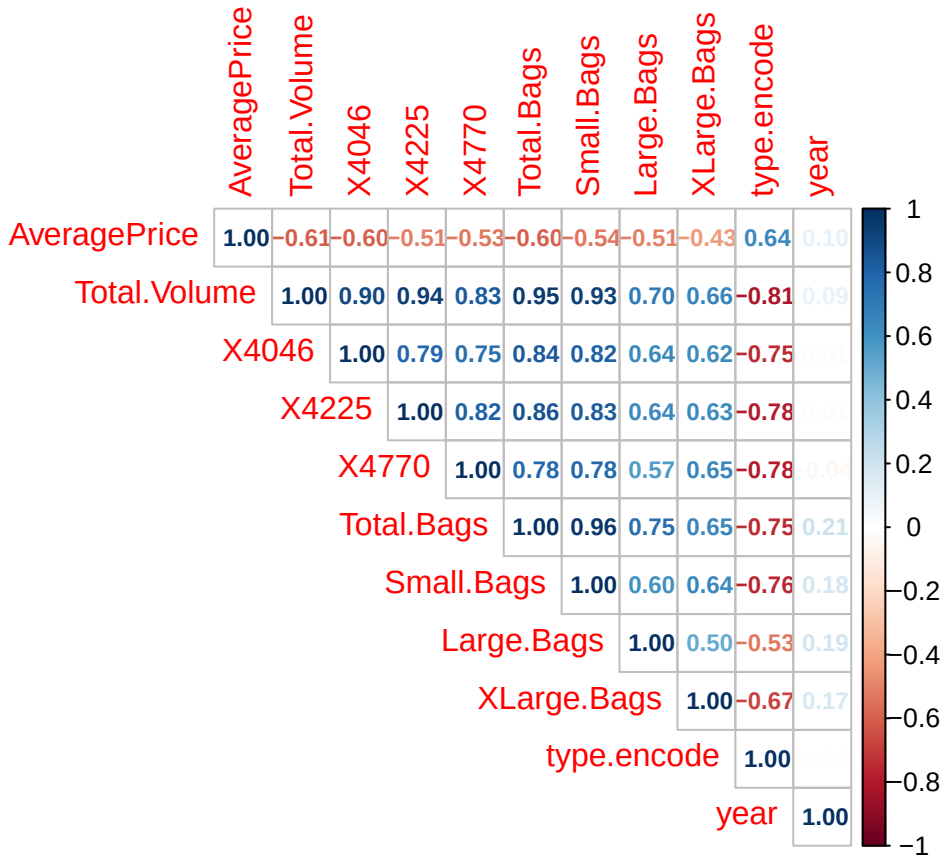
Vemos que también el p valor es menor a alfa=0.05, descartando la hipótesis nula de igualdad de mediana, aceptando el precio mayor de los aguacates orgánicos.

4.3. Correlación de Spearman

```
# encoding variable categóricas 1 como organicas 0 convencionales
avocado$type.encode <- ifelse(avocado$type == "organic",1,0)
```

Vamos a representar una correlación de Spearman:

```
correlation = round(cor(avocado[,c("AveragePrice", "Total.Volume", "X4046", "X4225",
                    "X4770", "Total.Bags", "Small.Bags", "Large.Bags",
                    "XLarge.Bags", "type.encode", "year")], method = "spearman"),2)
corrplot(correlation, method="number", type="upper", number.cex=0.75)
```



Vemos que el tipo de aguacate tiene una correlación positiva de 0.64 con el precio mientras todas las otras variables tiene correlación negativa con el precio medio, siendo el año una variable que casi no tiene correlación con el precio.

4.4 Modelos de regresión

Primero vamos a dividir el dataset en conjunto de entrenamiento/training y de test.

```
set.seed(222222) # un random seed cualquiera
split = sample.split(avocado$AveragePrice, SplitRatio = 0.8)
training_set = subset(avocado, split == TRUE)
testing_set = subset(avocado, split == FALSE)
```

4.4.1. Regresión lineal múltiple

Modelo de regresión linear múltiple:

En la regresión linear múltiple no siempre el modelo con todas las variables es la mejor, eso es porque no todas las variables aportan información útil al modelo, usaremos el método de: Eliminación hacia atras, que consiste en:

- 1. Elegir un nivel de significación (SL) para quedar con la variable (0.05 normalmente)
- 2. meter todas las variables modelos a la vez: *All in*
- 3. elegir la variable predictora con p valor más grande, si p valor > SL al paso 4 sino al FIN

- 4. Eliminar variable predictora
- 5. Ajustar el nuevo modelo sin esa variable y repetir paso 3
- FIN: cuando todas las variables tenga que p valor < SL

```
# la función ya se encarga de crear las variables dummies de las variables categóricas
model = lm(AveragePrice ~ Total.Volume + X4046 + X4225 + X4770 + Total.Bags + Small.Bags +
                      Large.Bags + XLarge.Bags + type + year + region, data = training_set)
summary(model)
```

```
##
## Call:
## lm(formula = AveragePrice ~ Total.Volume + X4046 + X4225 + X4770 +
##     Total.Bags + Small.Bags + Large.Bags + XLarge.Bags + type +
##     year + region, data = training_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.05765 -0.16343 -0.01109  0.14560  1.49277
##
## Coefficients:
##                                Estimate    Std. Error t value
## (Intercept)                -77.570678279   4.890996771 -15.860
## Total.Volume                -0.000006597   0.000042451  -0.155
## X4046                        0.000006603   0.000042451   0.156
## X4225                        0.000006587   0.000042451   0.155
## X4770                        0.000006522   0.000042450   0.154
## Total.Bags                  -0.015202776   0.033534104  -0.453
## Small.Bags                   0.015209363   0.033534236   0.454
## Large.Bags                   0.015209313   0.033534236   0.454
## XLarge.Bags                  0.015210795   0.033534237   0.454
## typeorganic                  0.490174905   0.004691741 104.476
## year                         0.039126189   0.002425884  16.129
## regionAtlanta               -0.213493050   0.022841158  -9.347
## regionBaltimoreWashington   -0.014088953   0.023155668  -0.608
## regionBoise                 -0.207095818   0.022988729  -9.009
## regionBoston                -0.014049653   0.023162140  -0.607
## regionBuffaloRochester      -0.046441524   0.023005999  -2.019
## regionCalifornia            -0.166912601   0.023221809  -7.188
## regionCharlotte              0.050229042   0.023252394   2.160
## regionChicago                0.012990835   0.023293680   0.558
## regionCincinnatiDayton      -0.343333907   0.022864931 -15.016
## regionColumbus              -0.311896498   0.023202148 -13.443
## regionDallasFtWorth         -0.472388475   0.023267341 -20.303
## regionDenver                -0.314254762   0.023271644 -13.504
## regionDetroit               -0.296633387   0.023199501 -12.786
## regionGrandRapids           -0.058904016   0.023294947  -2.529
## regionGreatLakes            -0.223928670   0.023901118  -9.369
## regionHarrisburgScranton    -0.045970115   0.023180683  -1.983
## regionHartfordSpringfield    0.260583514   0.022991632  11.334
## regionHouston               -0.503832166   0.023214095 -21.704
## regionIndianapolis          -0.240722185   0.023071570 -10.434
## regionJacksonville          -0.035869517   0.023339338  -1.537
```

```
## regionLasVegas             -0.170876687   0.022949585  -7.446
## regionLosAngeles           -0.361879474   0.023341972 -15.503
## regionLouisville           -0.259555183   0.022882097 -11.343
## regionMiamiFtLauderdale    -0.128894773   0.023204830  -5.555
## regionMidsouth             -0.142321013   0.023362791  -6.092
## regionNashville            -0.347833362   0.022846280 -15.225
## regionNewOrleansMobile     -0.248649592   0.023098095 -10.765
## regionNewYork               0.176915795   0.023047508   7.676
## regionNortheast             0.058161469   0.024879640   2.338
## regionNorthernNewEngland   -0.088159748   0.023060338  -3.823
## regionOrlando              -0.049482826   0.023143870  -2.138
## regionPhiladelphia          0.074073452   0.022773418   3.253
## regionPhoenixTucson        -0.338992052   0.023204633 -14.609
## regionPittsburgh           -0.196548964   0.023427691  -8.390
## regionPlains               -0.119872711   0.023185209  -5.170
## regionPortland             -0.253872088   0.023116551 -10.982
## regionRaleighGreensboro    -0.001830474   0.022973566  -0.080
## regionRichmondNorfolk      -0.267469636   0.022863322 -11.699
## regionRoanoke              -0.306442049   0.023027301 -13.308
## regionSacramento            0.057893089   0.023268120   2.488
## regionSanDiego             -0.168597992   0.023159275  -7.280
## regionSanFrancisco          0.258445112   0.023024304  11.225
## regionSeattle              -0.130963378   0.023030208  -5.687
## regionSouthCarolina        -0.149985266   0.022907380  -6.547
## regionSouthCentral         -0.452901778   0.023859770 -18.982
## regionSoutheast            -0.159095022   0.023595144  -6.743
## regionSpokane              -0.118571549   0.023068956  -5.140
## regionStLouis              -0.129472643   0.023201761  -5.580
## regionSyracuse             -0.040723578   0.023198943  -1.755
## regionTampa                -0.159769930   0.023038326  -6.935
## regionTotalUS              -0.163839508   0.029321255  -5.588
## regionWest                 -0.250769101   0.023982399 -10.456
## regionWestTexNewMexico     -0.288930337   0.023228749 -12.438
##                                        Pr(>|t|)
## (Intercept)               < 0.0000000000000002 ***
## Total.Volume                           0.876504
## X4046                                  0.876389
## X4225                                  0.876697
## X4770                                  0.877888
## Total.Bags                             0.650302
## Small.Bags                             0.650161
## Large.Bags                             0.650163
## XLarge.Bags                            0.650131
## typeorganic               < 0.0000000000000002 ***
## year                      < 0.0000000000000002 ***
## regionAtlanta             < 0.0000000000000002 ***
## regionBaltimoreWashington              0.542902
## regionBoise               < 0.0000000000000002 ***
## regionBoston                           0.544140
## regionBuffaloRochester                 0.043540 *
## regionCalifornia            0.0000000000006908 ***
## regionCharlotte                        0.030776 *
## regionChicago                          0.577059
## regionCincinnatiDayton    < 0.0000000000000002 ***
```

```
## regionColumbus            < 0.0000000000000002 ***
## regionDallasFtWorth       < 0.0000000000000002 ***
## regionDenver              < 0.0000000000000002 ***
## regionDetroit             < 0.0000000000000002 ***
## regionGrandRapids                    0.011462 *
## regionGreatLakes          < 0.0000000000000002 ***
## regionHarrisburgScranton             0.047373 *
## regionHartfordSpringfield < 0.0000000000000002 ***
## regionHouston             < 0.0000000000000002 ***
## regionIndianapolis        < 0.0000000000000002 ***
## regionJacksonville                   0.124347
## regionLasVegas              0.0000000000001018 ***
## regionLosAngeles          < 0.0000000000000002 ***
## regionLouisville          < 0.0000000000000002 ***
## regionMiamiFtLauderdale     0.0000000283047729 ***
## regionMidsouth              0.0000000011447843 ***
## regionNashville           < 0.0000000000000002 ***
## regionNewOrleansMobile    < 0.0000000000000002 ***
## regionNewYork               0.0000000000000174 ***
## regionNortheast                      0.019416 *
## regionNorthernNewEngland             0.000132 ***
## regionOrlando                        0.032529 *
## regionPhiladelphia                   0.001146 **
## regionPhoenixTucson       < 0.0000000000000002 ***
## regionPittsburgh          < 0.0000000000000002 ***
## regionPlains                0.0000002369177511 ***
## regionPortland            < 0.0000000000000002 ***
## regionRaleighGreensboro              0.936495
## regionRichmondNorfolk     < 0.0000000000000002 ***
## regionRoanoke             < 0.0000000000000002 ***
## regionSacramento                     0.012854 *
## regionSanDiego              0.0000000000003511 ***
## regionSanFrancisco        < 0.0000000000000002 ***
## regionSeattle               0.0000000132094656 ***
## regionSouthCarolina         0.0000000000604866 ***
## regionSouthCentral        < 0.0000000000000002 ***
## regionSoutheast             0.0000000000161335 ***
## regionSpokane               0.0000002784881035 ***
## regionStLouis               0.0000000244404962 ***
## regionSyracuse                       0.079211 .
## regionTampa                 0.0000000000042348 ***
## regionTotalUS               0.0000000234179349 ***
## regionWest                < 0.0000000000000002 ***
## regionWestTexNewMexico    < 0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2678 on 14529 degrees of freedom
## Multiple R-squared:  0.5598, Adjusted R-squared:  0.5579
## F-statistic: 293.3 on 63 and 14529 DF,  p-value: < 0.00000000000000022
```

los pasos son muy repetitivos nos lo saltamos y obtenemos el modelo final al aplicar el algoritmo anterior:

```r
# se elimina ya de una sola pasada todas las regiones que no tiene poder predictor del precio
region_no_sig = c('Syracuse', 'RaleighGreensboro', 'Chicago','Boston', 'BaltimoreWashington')
training_set_1 = subset(training_set, ! region %in% region_no_sig)
testing_set_1 = subset(testing_set, ! region %in% region_no_sig)
```

```r
linear_model = lm(AveragePrice ~ type + year + region, data = training_set_1)
summary(linear_model)$adj.r.squared
```

```
## [1] 0.5583716
```

Vemos que tiene casi el mismo R ajustado (0.558) que el primer modelo pero con menos variables.

Puede ser que el R que nos ha dado sea bajo porque los datos no tenga linealidad, homocedasticidad ni tampoco normalidad de las variables.

Un R ajustado de 0.558 no es un modelo muy bueno. podemos probar con otro modelo de regresión como puede ser como el Random Forest o el Support Vector Maquine.

4.4.2. Regresión Random Forest

```r
# elegimos 500 arboles
rf_model <- randomForest(AveragePrice ~ Total.Volume + X4046 + X4225 + X4770 +
                         Total.Bags + Small.Bags + Large.Bags + XLarge.Bags + type +
                         year + region,
                         data = training_set,
                         ntree=500,
                         keep.forest=TRUE,
                         importance=FALSE)
rf_model
```

```
##
## Call:
##  randomForest(formula = AveragePrice ~ Total.Volume + X4046 +      X4225 + X4770 + Total.Bags + Small
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##           Mean of squared residuals: 0.0247138
##                     % Var explained: 84.76
```

Vemos que el modelo es bastante bueno, donde las variables independientes explica un 85% de la varianza de la variable precio.

4.4.3. Support Vector Maquine

En el Suport vector maquine podemos elegir diferentes *kernels* o núcleos para poder clasificar los datos , se ha elegido el que ha tenido la mejor evaluación:

```
svm_model = svm(formula = AveragePrice ~ Total.Volume + X4046 + X4225 + X4770 +
                          Total.Bags + Small.Bags + Large.Bags + XLarge.Bags +
                          type + year + region,
                data = training_set,
                type = "eps-regression",
                kernel = "radial")
svm_model
```

```
##
## Call:
## svm(formula = AveragePrice ~ Total.Volume + X4046 + X4225 + X4770 +
##     Total.Bags + Small.Bags + Large.Bags + XLarge.Bags + type + year +
##     region, data = training_set, type = "eps-regression", kernel = "radial")
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  radial
##        cost:  1
##       gamma:  0.015625
##     epsilon:  0.1
##
##
## Number of Support Vectors:  12236
```

4.4.4. Evaluación de modelos

Comparación de los modelo con el RMSE (Raíz del error cuadrático medio o Root Mean Square Error)

```
# lineal
pred_linear <- predict(linear_model, testing_set_1[, c('type', 'year', 'region')])
RMSE_linear=rmse(pred_linear,testing_set_1$AveragePrice)
RMSE_linear
```

```
## [1] 0.2734394
```

```
# forest
pred_forest <- predict(rf_model, testing_set[, c("Total.Volume", "X4046", "X4225",
                    "X4770", "Total.Bags", "Small.Bags", "Large.Bags",
                    "XLarge.Bags",'type', 'year', 'region')])
RMSE_rf=rmse(pred_forest,testing_set$AveragePrice)
RMSE_rf
```

```
## [1] 0.1546469
```

```
# svm
pred_svm <- predict(svm_model, testing_set[, c("Total.Volume", "X4046", "X4225",
                    "X4770", "Total.Bags", "Small.Bags", "Large.Bags",
                    "XLarge.Bags",'type', 'year', 'region')])
RMSE_svm=rmse(pred_svm,testing_set$AveragePrice)
RMSE_svm
```

```
## [1] 0.2601685
```

Ya podemos ver que el Random Forest nos ha dado el mejor RMSE y la R^2 ajustado deberia ser también la mejor:

```
# lineal
r2 = R2(testing_set_1$AveragePrice, pred_linear)
r2
```

```
## [1] 0.554322
```

```
# random forest
r2 = R2(testing_set$AveragePrice, pred_forest)
r2
```

```
## [1] 0.8565134
```

```
# SVM
r2 = R2(testing_set$AveragePrice, pred_svm)
r2
```

```
## [1] 0.5866428
```

Efectivamente el modelo Random forest nos ha dado un R^{2} ajustado de 0.85, un poder de predicción bastante bueno.

4.5. Exportar datos

```
# borrar type ya que tenemos el type.encoded
avocado = subset(avocado, select = -c(type) )

write.csv(avocado,"avocado_finish.csv", row.names = TRUE)
```

5. Conclusiones

Al estudiar la normalidad y la homocedasticidad, vemos que la distribución de precios ni es normal ni homocedástico por lo tanto solo podemos emplear pruebas no paramétricas que cumplan con estos requisitos, se ha usado el Test de rangos y signos para comparar los precios entre los aguacates orgánicos y no orgánicos y podemos afirmar que estadísticamente el precio de los aguacate orgánicos es mayor a los convencionales, y por lo que se ve también se vende más más cantidad de estas últimas.

En el punto 4.4 cuando evaluamos los diferentes modelos de regresión podemos ver que la regresión lineal en este caso ha dado unos resultados muy pobres dando a entender que nos falta datos relevantes o que el problema no es lineal, mientra que se mejora un poco la RMSE con SVM, la mejora realmente importante viene con el modelo de Random Forest donde también tenemos un R^{2} de 0.85 que nos podrá dar unos resultados bastante fidedignos.

Hay que estacar estos dos últimos modelos son computacionalmente mucho más pesados que la regresión lineal.