Práctica 2: Limpieza y análisis de datos

Autor: Victor Mascarell Ascó

1 de enero 2020

Descripción del dataset

En esta segunda práctica enmarcada dentro de la asignatura Tipología y Ciclo de Vida del Dato del Master en ciencia de datos de la UOC vamos a tratar el Dataset Cancer Breast que podemos encontrar en el siguiente enlace del repositorio de Kaggle: https://www.kaggle.com/uciml/breast-cancer-wisconsin-data.

El dataset contiene 569 registros de casos de cáncer de pecho con 30 atributos descriptivos de los mismos, 1 atributo clasificatorio en función de si el tumor es benigno o maligno y un atributo de ID. Los atributos descriptivos describen al media (mean), error estándar (se) y la media de los tres valores más grandes (worst) de las células del tumor a través de 10 de sus características. Así las 30 variables descriptivas sería:

1. radius_mean (Variable continúa)
2. texture_mean (Variable continúa)
3. perimeter_mean (Variable continúa)
4. area_mean (Variable continúa)
5. smoothness_mean (Variable continúa)
6. compactness_mean (Variable continúa)
7. concavity_mean (Variable continúa)
8. concave.points_mean (Variable continúa)
9. symmetry_mean (Variable continúa)
10. fractal_dimension_mean (Variable continúa)
11. radius_se (Variable continúa)
12. texture_se (Variable continúa)
13. perimeter_se (Variable continúa)

14. area_se (Variable continúa)
15. smoothness_se (Variable continúa)
16. compactness_se (Variable continúa)
17. concavity_se (Variable continúa)
18. concave.points_se (Variable continúa)
19. symmetry_se (Variable continúa)
20. fractal_dimension_se (Variable continúa)
21. radius_worst (Variable continúa)
22. texture_worst (Variable continúa)
23. perimeter_worst (Variable continúa)
24. area_worst (Variable continúa)
25. smoothness_worst (Variable continúa)
26. compactness_worst (Variable continúa)
27. concavity_worst (Variable continúa)
28. concave.points_worst (Variable continúa)
29. symmetry_worst (Variable continúa)
30. fractal_dimension_worst (Variable continúa):

Los otros campos serían:

31. id (Dato continúo): Identificador único.

32. diagnosis (Dato categórico): Diagnóstico.

Este dataset es importante ya que nos ofrece la información necesario de casos de cáncer de mama anteriores, lo que nos puede ayudar a construir un modelo predictivo para diagnosticar si los tumores de mama encontrados son malignos o benignos. Así, el objetivo final será crear un modelo que nos permita clasificar y predecir si el cáncer de mama es maligno o benigno.

Integración y selección de los datos de interés a analizar

Para realizar el modelo descartaremos la variable de ID, ya que no nos aporta información alguna sobre las posibles características del tumor. Y por otro lado dejaremos la variable clasificatoria para comprobar la fiabilidad del modelo. No obstante, en los siguientes apartados, también utilizaremos la variable clasificatoria para analizar los datos y concluir que variables descriptivas son las más adecuadas para construir el modelo.

Limpieza de los datos

Lectura y carga de datos

En primer lugar procedemos a cargar el fichero de datos. Observar si los datos se han cargado correctamente.

```
# Con read.csv carcamos el fichero
data <- read.csv('data.csv',stringsAsFactors = FALSE)

# Utilizamos dim para comprobar el número de registros
dim(data)
```

```
## [1] 569  33
```

```
# Utilizamos str para ver el tipo de datos
str(data)
```

```
## 'data.frame':    569 obs. of  33 variables:
##  $ id                      : int  842302 842517 84300903 84348301
84358402 843786 844359 84458202 844981 84501001 ...
##  $ diagnosis               : chr  "M" "M" "M" "M" ...
##  $ radius_mean             : num  18 20.6 19.7 11.4 20.3 ...
##  $ texture_mean            : num  10.4 17.8 21.2 20.4 14.3 ...
##  $ perimeter_mean          : num  122.8 132.9 130 77.6 135.1 ...
```

```
##  $ area_mean              : num  1001 1326 1203 386 1297 ...
##  $ smoothness_mean        : num  0.1184 0.0847 0.1096 0.1425 0.1003
...
##  $ compactness_mean       : num  0.2776 0.0786 0.1599 0.2839 0.1328
...
##  $ concavity_mean         : num  0.3001 0.0869 0.1974 0.2414
0.198 ...
##  $ concave.points_mean    : num  0.1471 0.0702 0.1279 0.1052 0.1043
...
##  $ symmetry_mean          : num  0.242 0.181 0.207 0.26 0.181 ...
##  $ fractal_dimension_mean : num  0.0787 0.0567 0.06 0.0974
0.0588 ...
##  $ radius_se              : num  1.095 0.543 0.746 0.496 0.757 ...
##  $ texture_se             : num  0.905 0.734 0.787 1.156 0.781 ...
##  $ perimeter_se           : num  8.59 3.4 4.58 3.44 5.44 ...
##  $ area_se                : num  153.4 74.1 94 27.2 94.4 ...
##  $ smoothness_se          : num  0.0064 0.00522 0.00615 0.00911
0.01149 ...
##  $ compactness_se         : num  0.049 0.0131 0.0401 0.0746
0.0246 ...
##  $ concavity_se           : num  0.0537 0.0186 0.0383 0.0566 0.0569
...
##  $ concave.points_se      : num  0.0159 0.0134 0.0206 0.0187 0.0188
...
##  $ symmetry_se            : num  0.03 0.0139 0.0225 0.0596
0.0176 ...
##  $ fractal_dimension_se   : num  0.00619 0.00353 0.00457 0.00921
0.00511 ...
##  $ radius_worst           : num  25.4 25 23.6 14.9 22.5 ...
##  $ texture_worst          : num  17.3 23.4 25.5 26.5 16.7 ...
##  $ perimeter_worst        : num  184.6 158.8 152.5 98.9 152.2 ...
##  $ area_worst             : num  2019 1956 1709 568 1575 ...
##  $ smoothness_worst       : num  0.162 0.124 0.144 0.21 0.137 ...
##  $ compactness_worst      : num  0.666 0.187 0.424 0.866 0.205 ...
##  $ concavity_worst        : num  0.712 0.242 0.45 0.687 0.4 ...
##  $ concave.points_worst   : num  0.265 0.186 0.243 0.258 0.163 ...
##  $ symmetry_worst         : num  0.46 0.275 0.361 0.664 0.236 ...
##  $ fractal_dimension_worst: num  0.1189 0.089 0.0876 0.173
```

```
0.0768 ...
##  $ X                       : logi  NA NA NA NA NA NA ...

# Imprimimos las primeras filas del dataset
head(data)

##         id diagnosis radius_mean texture_mean perimeter_mean
area_mean
## 1   842302         M       17.99        10.38         122.80
1001.0
## 2   842517         M       20.57        17.77         132.90
1326.0
## 3 84300903         M       19.69        21.25         130.00
1203.0
## 4 84348301         M       11.42        20.38          77.58
386.1
## 5 84358402         M       20.29        14.34         135.10
1297.0
## 6   843786         M       12.45        15.70          82.57
477.1
##   smoothness_mean compactness_mean concavity_mean
concave.points_mean
## 1         0.11840          0.27760         0.3001
0.14710
## 2         0.08474          0.07864         0.0869
0.07017
## 3         0.10960          0.15990         0.1974
0.12790
## 4         0.14250          0.28390         0.2414
0.10520
## 5         0.10030          0.13280         0.1980
0.10430
## 6         0.12780          0.17000         0.1578
0.08089
##   symmetry_mean fractal_dimension_mean radius_se texture_se
perimeter_se
## 1        0.2419                0.07871    1.0950     0.9053
8.589
## 2        0.1812                0.05667    0.5435     0.7339
3.398
## 3        0.2069                0.05999    0.7456     0.7869
```

```
4.585
## 4         0.2597                0.09744    0.4956     1.1560
3.445
## 5         0.1809                0.05883    0.7572     0.7813
5.438
## 6         0.2087                0.07613    0.3345     0.8902
2.217
##   area_se smoothness_se compactness_se concavity_se
concave.points_se
## 1  153.40      0.006399        0.04904      0.05373
0.01587
## 2   74.08      0.005225        0.01308      0.01860
0.01340
## 3   94.03      0.006150        0.04006      0.03832
0.02058
## 4   27.23      0.009110        0.07458      0.05661
0.01867
## 5   94.44      0.011490        0.02461      0.05688
0.01885
## 6   27.19      0.007510        0.03345      0.03672
0.01137
##   symmetry_se fractal_dimension_se radius_worst texture_worst
## 1     0.03003             0.006193        25.38         17.33
## 2     0.01389             0.003532        24.99         23.41
## 3     0.02250             0.004571        23.57         25.53
## 4     0.05963             0.009208        14.91         26.50
## 5     0.01756             0.005115        22.54         16.67
## 6     0.02165             0.005082        15.47         23.75
##   perimeter_worst area_worst smoothness_worst compactness_worst
## 1          184.60     2019.0           0.1622            0.6656
## 2          158.80     1956.0           0.1238            0.1866
## 3          152.50     1709.0           0.1444            0.4245
## 4           98.87      567.7           0.2098            0.8663
## 5          152.20     1575.0           0.1374            0.2050
## 6          103.40      741.6           0.1791            0.5249
##   concavity_worst concave.points_worst symmetry_worst
## 1          0.7119               0.2654         0.4601
## 2          0.2416               0.1860         0.2750
## 3          0.4504               0.2430         0.3613
```

```
## 4          0.6869                0.2575          0.6638
## 5          0.4000                0.1625          0.2364
## 6          0.5355                0.1741          0.3985
##   fractal_dimension_worst  X
## 1                 0.11890 NA
## 2                 0.08902 NA
## 3                 0.08758 NA
## 4                 0.17300 NA
## 5                 0.07678 NA
## 6                 0.12440 NA
```

Podemos comprobar que los datos se han cargado correctamente a excepción de la variable diagnosis que nos interesaría tenerla en factor y nos la carga con char . También nos ha cargado un atributo de más (X) que parece contener valores nulos. Este lo trataremos en el siguiente apartado. Ahora procederemos a cambiar el tipo de dato de diagnosis a factor

```
# Pasamos a factor diagnosis
data$diagnosis <- factor(data$diagnosis)

# Comprobamos el resultado
str(data$diagnosis)
```

```
##  Factor w/ 2 levels "B","M": 2 2 2 2 2 2 2 2 2 ...
```

¿Los datos contienen ceros o elementos vacíos?

Procedemos a calcular los valores vacíos de cada atributo

```
# Sumamos valores vacíos
colSums(is.na(data))
```

```
##                         id                  diagnosis
radius_mean
##                          0                          0
0
##            texture_mean          perimeter_mean
area_mean
##                          0                          0
0
##         smoothness_mean        compactness_mean
concavity_mean
```

```
##                         0                         0
0
##     concave.points_mean           symmetry_mean
fractal_dimension_mean
##                         0                         0
0
##               radius_se              texture_se
perimeter_se
##                         0                         0
0
##                 area_se           smoothness_se
compactness_se
##                         0                         0
0
##            concavity_se       concave.points_se
symmetry_se
##                         0                         0
0
##    fractal_dimension_se            radius_worst
texture_worst
##                         0                         0
0
##         perimeter_worst              area_worst
smoothness_worst
##                         0                         0
0
##       compactness_worst         concavity_worst
concave.points_worst
##                         0                         0
0
##          symmetry_worst fractal_dimension_worst
X
##                         0                         0
569
```

Podemos comprobar que la última columna es un error de carga y solo contiene valores vacíos por lo que procedemos a eliminarla, aprovechamos para eliminar también id. Si tuviéramos valores vacíos dentro de los campos de interés podríamos optar por sustituirlos por la media o eliminar el registro en función del interés.

```r
# Seleccionamos las columnas deseadas
data <- data[, 2:32]
```

De la misma forma vamos a comprobar si existen 0 en los atributos del dataset.

```r
# Sumamos valores = 0 por columnas
colSums(data==0)
```

```
##                diagnosis              radius_mean
texture_mean
##                        0                        0
0
##          perimeter_mean                area_mean
smoothness_mean
##                        0                        0
0
##         compactness_mean           concavity_mean
concave.points_mean
##                        0                       13
13
##           symmetry_mean  fractal_dimension_mean
radius_se
##                        0                        0
0
##              texture_se            perimeter_se
area_se
##                        0                        0
0
##           smoothness_se          compactness_se
concavity_se
##                        0                        0
13
##       concave.points_se             symmetry_se
fractal_dimension_se
##                       13                        0
0
##            radius_worst           texture_worst
perimeter_worst
##                        0                        0
0
##              area_worst        smoothness_worst
```

```
compactness_worst
##                         0                         0
0
##          concavity_worst    concave.points_worst
symmetry_worst
##                        13                        13
0
## fractal_dimension_worst
##                         0
```

Observamos que sí. Comprobamos si los registros que contienen 0 son los mismos.

```
# Imprimos las filas que contienen 0 en concave.points_se
print(data[data$concave.points_se==0,])
```

```
##     diagnosis radius_mean texture_mean perimeter_mean area_mean
## 102         B       6.981        13.43          43.79     143.5
## 141         B       9.738        11.97          61.24     288.5
## 175         B      10.660        15.15          67.49     349.6
## 176         B       8.671        14.45          54.42     227.2
## 193         B       9.720        18.22          60.73     288.1
## 315         B       8.597        18.60          54.09     221.2
## 392         B       8.734        16.84          55.27     234.3
## 474         B      12.270        29.97          77.42     465.4
## 539         B       7.729        25.49          47.98     178.8
## 551         B      10.860        21.48          68.51     360.5
## 558         B       9.423        27.88          59.26     271.3
## 562         B      11.200        29.37          70.67     386.0
## 569         B       7.760        24.54          47.92     181.0
##     smoothness_mean compactness_mean concavity_mean
concave.points_mean
## 102         0.11700          0.07568              0
0
## 141         0.09250          0.04102              0
0
## 175         0.08792          0.04302              0
0
## 176         0.09138          0.04276              0
0
## 193         0.06950          0.02344              0
```

```
0
## 315         0.10740          0.05847               0
0
## 392         0.10390          0.07428               0
0
## 474         0.07699          0.03398               0
0
## 539         0.08098          0.04878               0
0
## 551         0.07431          0.04227               0
0
## 558         0.08123          0.04971               0
0
## 562         0.07449          0.03558               0
0
## 569         0.05263          0.04362               0
0
##     symmetry_mean fractal_dimension_mean radius_se texture_se
perimeter_se
## 102        0.1930                0.07818    0.2241     1.5080
1.553
## 141        0.1903                0.06422    0.1988     0.4960
1.218
## 175        0.1928                0.05975    0.3309     1.9250
2.155
## 176        0.1722                0.06724    0.2204     0.7873
1.435
## 193        0.1653                0.06447    0.3539     4.8850
2.230
## 315        0.2163                0.07359    0.3368     2.7770
2.222
## 392        0.1985                0.07098    0.5169     2.0790
3.167
## 474        0.1701                0.05960    0.4455     3.6470
2.884
## 539        0.1870                0.07285    0.3777     1.4620
2.492
## 551        0.1661                0.05948    0.3163     1.3040
2.115
## 558        0.1742                0.06059    0.5375     2.9270
```

```
3.618
## 562        0.1060                0.05502    0.3141     3.8960
2.041
## 569        0.1587                0.05884    0.3857     1.4280
2.548
##     area_se smoothness_se compactness_se concavity_se
concave.points_se
## 102   9.833      0.010190       0.010840            0
0
## 141  12.260      0.006040       0.005656            0
0
## 175  21.980      0.008713       0.010170            0
0
## 176  11.360      0.009172       0.008007            0
0
## 193  21.690      0.001713       0.006736            0
0
## 315  17.810      0.020750       0.014030            0
0
## 392  28.850      0.015820       0.019660            0
0
## 474  35.130      0.007339       0.008243            0
0
## 539  19.140      0.012660       0.009692            0
0
## 551  20.670      0.009579       0.011040            0
0
## 558  29.110      0.011590       0.011240            0
0
## 562  22.810      0.007594       0.008878            0
0
## 569  19.150      0.007189       0.004660            0
0
##     symmetry_se fractal_dimension_se radius_worst texture_worst
## 102     0.02659              0.004100        7.930         19.54
## 141     0.02277              0.003220       10.620         14.10
## 175     0.03265              0.001002       11.540         19.20
## 176     0.02711              0.003399        9.262         17.04
## 193     0.03799              0.001688        9.968         20.83
```

```
## 315     0.06146              0.006820        8.952          22.44
## 392     0.01865              0.006736       10.170          22.80
## 474     0.03141              0.003136       13.450          38.05
## 539     0.02882              0.006872        9.077          30.92
## 551     0.03004              0.002228       11.660          24.77
## 558     0.03004              0.003324       10.490          34.24
## 562     0.01989              0.001773       11.920          38.30
## 569     0.02676              0.002783        9.456          30.37
##     perimeter_worst area_worst smoothness_worst compactness_worst
## 102           50.41      185.2           0.15840            0.12020
## 141           66.53      342.9           0.12340            0.07204
## 175           73.20      408.3           0.10760            0.06791
## 176           58.36      259.2           0.11620            0.07057
## 193           62.25      303.8           0.07117            0.02729
## 315           56.65      240.1           0.13470            0.07767
## 392           64.01      317.0           0.14600            0.13100
## 474           85.08      558.9           0.09422            0.05213
## 539           57.17      248.0           0.12560            0.08340
## 551           74.08      412.3           0.10010            0.07348
## 558           66.50      330.6           0.10730            0.07158
## 562           75.19      439.6           0.09267            0.05494
## 569           59.16      268.6           0.08996            0.06444
##     concavity_worst concave.points_worst symmetry_worst
## 102               0                    0         0.2932
## 141               0                    0         0.3105
## 175               0                    0         0.2710
## 176               0                    0         0.2592
## 193               0                    0         0.1909
## 315               0                    0         0.3142
## 392               0                    0         0.2445
## 474               0                    0         0.2409
## 539               0                    0         0.3058
## 551               0                    0         0.2458
## 558               0                    0         0.2475
## 562               0                    0         0.1566
## 569               0                    0         0.2871
##     fractal_dimension_worst
```

```
## 102                  0.09382
## 141                  0.08151
## 175                  0.06164
## 176                  0.07848
## 193                  0.06559
## 315                  0.08116
## 392                  0.08865
## 474                  0.06743
## 539                  0.09938
## 551                  0.06592
## 558                  0.06969
## 562                  0.05905
## 569                  0.07039
```

Son la mismas filas y como vemos la media, el error estándar y el "worst" dan 0, caso bastante improbable. Procedemos a eliminar estos registros del dataset ya que contiene más de 500.

```
# Eliminamos los valores
data <- data[data$concave.points_se!=0,]

# Comprobamos de nuevo con el sumatoria de "0" y la dimentsión.
colSums(data==0)
```

```
##               diagnosis             radius_mean
texture_mean
##                       0                       0
0
##          perimeter_mean               area_mean
smoothness_mean
##                       0                       0
0
##        compactness_mean          concavity_mean
concave.points_mean
##                       0                       0
0
##           symmetry_mean  fractal_dimension_mean
radius_se
##                       0                       0
0
##              texture_se            perimeter_se
```

```
area_se
##                         0                         0
0
##           smoothness_se          compactness_se
concavity_se
##                         0                         0
0
##       concave.points_se             symmetry_se
fractal_dimension_se
##                         0                         0
0
##             radius_worst           texture_worst
perimeter_worst
##                         0                         0
0
##               area_worst        smoothness_worst
compactness_worst
##                         0                         0
0
##         concavity_worst    concave.points_worst
symmetry_worst
##                         0                         0
0
## fractal_dimension_worst
##                         0
```

```
dim(data)
```

```
## [1] 556  31
```

Para la predicción utilizaremos la media de las características, ya que se considera más representativa en este caso. Por lo que vamos a proceder a eliminar las columnas relacionadas con contienen el error estándar y los peores valores o “worst”, quedándonos solo con 10 variables.

```
# Eliminamos las variables que contienen el SE y el "worst"
data <- data[, 1:11]
```

Tratamiento de valores extremos

A continuación procedemos a observar los valores extremos o outliers

```r
boxplot.stats(data$radius_mean)
```

```
## $stats
## [1]  7.691 11.760 13.455 16.050 22.270
##
## $n
## [1] 556
##
## $conf
## [1] 13.16754 13.74246
##
## $out
##  [1] 25.22 24.25 23.27 27.22 23.29 28.11 23.21 23.51 25.73 27.42
23.09
## [12] 24.63
```

```r
boxplot.stats(data$texture_mean)
```

```
## $stats
## [1]  9.710 16.175 18.855 21.750 29.810
##
## $n
## [1] 556
##
## $conf
## [1] 18.48144 19.22856
##
## $out
## [1] 32.47 33.81 39.28 33.56 31.12 30.72 30.62
```

```r
boxplot.stats(data$perimeter_mean)
```

```
## $stats
## [1]  48.34  75.80  87.09 105.40 147.30
##
## $n
## [1] 556
##
## $conf
## [1] 85.1066 89.0734
```

```
##
## $out
##  [1] 171.5 152.8 166.2 152.1 182.1 158.9 188.5 153.5 155.1 174.2
186.9
## [12] 152.1 165.5
```

```
boxplot.stats(data$area_mean)
```

```
## $stats
## [1]  170.40  427.60  557.65  798.30 1347.00
##
## $n
## [1] 556
##
## $conf
## [1] 532.8105 582.4895
##
## $out
##  [1] 1404 1878 1509 1761 1686 2250 1685 2499 1670 1364 1419 1491
1747 2010
## [15] 1546 1482 1386 1407 1384 2501 1682 1841 1479
```

```
boxplot.stats(data$smoothness_mean)
```

```
## $stats
## [1] 0.062510 0.086650 0.096035 0.105400 0.133500
##
## $n
## [1] 556
##
## $conf
## [1] 0.09477862 0.09729138
##
## $out
## [1] 0.1425 0.1398 0.1447 0.1634 0.1371
```

```
boxplot.stats(data$compactness_mean)
```

```
## $stats
## [1] 0.019380 0.066525 0.095090 0.130600 0.223900
```

```
##
## $n
## [1] 556
##
## $conf
## [1] 0.09079653 0.09938347
##
## $out
##  [1] 0.2776 0.2839 0.2396 0.2458 0.2293 0.2276 0.3454 0.2665 0.2768
0.2867
## [11] 0.2832 0.2413 0.2284 0.3114 0.2364 0.2363 0.2576 0.2770
```

```
boxplot.stats(data$concavity_mean)
```

```
## $stats
## [1] 0.000692 0.030740 0.064905 0.132350 0.281000
##
## $n
## [1] 556
##
## $conf
## [1] 0.05809643 0.07171357
##
## $out
##  [1] 0.3001 0.3130 0.3754 0.3339 0.4264 0.3003 0.4268 0.4108 0.2871
0.3523
## [11] 0.3201 0.3176 0.2914 0.3368 0.3189 0.3635 0.3174 0.3514
```

```
boxplot.stats(data$concave.points_mean)
```

```
## $stats
## [1] 0.001852 0.020890 0.034840 0.074855 0.152000
##
## $n
## [1] 556
##
## $conf
## [1] 0.03122397 0.03845603
##
```

```
## $out
##  [1] 0.1604 0.1845 0.1823 0.2012 0.1878 0.1620 0.1595 0.1913 0.1562
0.1689
```

```
boxplot.stats(data$symmetry_mean)
```

```
## $stats
## [1] 0.11670 0.16190 0.17925 0.19580 0.24590
##
## $n
## [1] 556
##
## $conf
## [1] 0.1769785 0.1815215
##
## $out
##  [1] 0.2597 0.2521 0.3040 0.2743 0.2906 0.2556 0.2655 0.2678 0.2540
0.2548
## [11] 0.2495 0.2595 0.2569 0.2538
```

```
boxplot.stats(data$fractal_dimension_mean)
```

```
## $stats
## [1] 0.049960 0.057670 0.061515 0.066100 0.078710
##
## $n
## [1] 556
##
## $conf
## [1] 0.06095013 0.06207987
##
## $out
##  [1] 0.09744 0.08243 0.08046 0.08980 0.08142 0.08261 0.09296
0.08116
##  [9] 0.08104 0.08743 0.08450 0.07950 0.09502 0.09575 0.07976
```

Observamos que aparecen algunos valores extremos, no obstante asumimos que estos valores son factibles y que se pueden dar casos que generen que el área de la célula, por ejemplo, sea más grande. Esto puede ser un factor influyente en el diagnóstico, por lo que inicialmente se

decide no tratar los valores extremos. Más adelante si resultará contraproducente para la predicción, se tomarían medidas.

Procedemos a guardar el dataset en un documento en formato .csv

```
write.csv(data,"clean_breast.csv")
```

Análisis de los datos

Selección de los grupos de datos a analizar

Antes que nada vamos a estandarizar las variables numéricas.

```
# Importamos la libreria dplyr y ggplot2
library(dplyr)
library(ggplot2)

# Utilizaremos mutate_if y luego is.numeric, ya que tenemos variables continuas
data_scaled <- mutate_if(data, is.numeric, scale)
```

A continuación sacaremos las correlaciones entre las variables numéricas:

```
cor(data[, 2:11])
```

```
##                         radius_mean texture_mean perimeter_mean
area_mean
## radius_mean               1.0000000  0.343924917      0.9977642
0.9880838
## texture_mean              0.3439249  1.000000000      0.3506461
0.3380214
## perimeter_mean            0.9977642  0.350646087      1.0000000
0.9871877
## area_mean                 0.9880838  0.338021366      0.9871877
1.0000000
## smoothness_mean           0.1572871  0.004953432      0.1949979
0.1659458
## compactness_mean          0.4922906  0.257085264      0.5446777
```

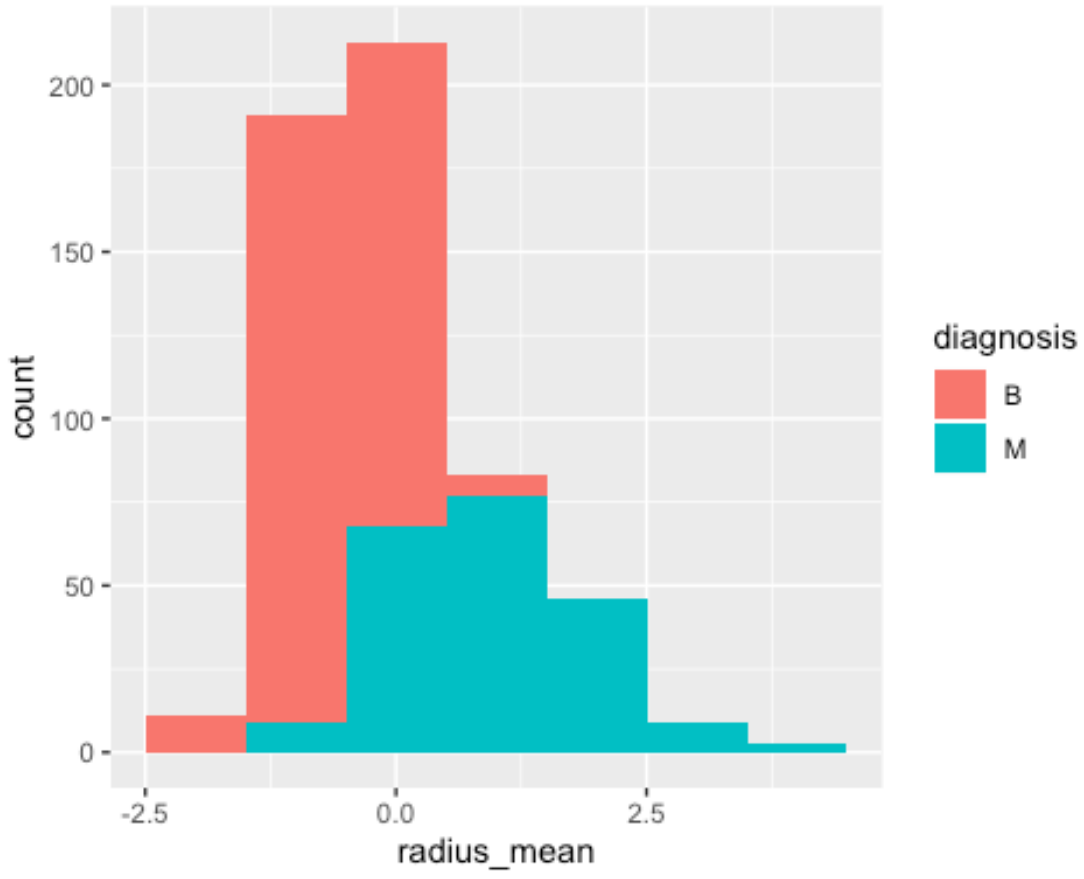
```
0.4862017
## concavity_mean           0.6668914  0.322500457      0.7075658
0.6774488
## concave.points_mean      0.8170052  0.315583845      0.8464296
0.8184491
## symmetry_mean            0.1509377  0.093134433      0.1870471
0.1528633
## fractal_dimension_mean  -0.3057228 -0.065109125     -0.2542666
-0.2784674
##                        smoothness_mean compactness_mean
concavity_mean
## radius_mean               0.157287059        0.4922906
0.6668914
## texture_mean              0.004953432        0.2570853
0.3225005
## perimeter_mean            0.194997902        0.5446777
0.7075658
## area_mean                 0.165945785        0.4862017
0.6774488
## smoothness_mean           1.000000000        0.6586596
0.5212938
## compactness_mean          0.658659596        1.0000000
0.8804934
## concavity_mean            0.521293833        0.8804934
1.0000000
## concave.points_mean       0.553796101        0.8265946
0.9189655
## symmetry_mean             0.556780020        0.6099932
0.5085856
## fractal_dimension_mean    0.591158775        0.5842977
0.3541342
##                        concave.points_mean symmetry_mean
## radius_mean                      0.8170052    0.15093771
## texture_mean                     0.3155838    0.09313443
## perimeter_mean                   0.8464296    0.18704712
## area_mean                        0.8184491    0.15286332
## smoothness_mean                  0.5537961    0.55678002
## compactness_mean                 0.8265946    0.60999323
## concavity_mean                   0.9189655    0.50858564
```

```
## concave.points_mean               1.0000000    0.47078756
## symmetry_mean                     0.4707876    1.00000000
## fractal_dimension_mean            0.1815634    0.47726915
##                          fractal_dimension_mean
## radius_mean                          -0.30572283
## texture_mean                         -0.06510913
## perimeter_mean                       -0.25426657
## area_mean                            -0.27846739
## smoothness_mean                       0.59115878
## compactness_mean                      0.58429773
## concavity_mean                        0.35413420
## concave.points_mean                   0.18156336
## symmetry_mean                         0.47726915
## fractal_dimension_mean                1.00000000
```

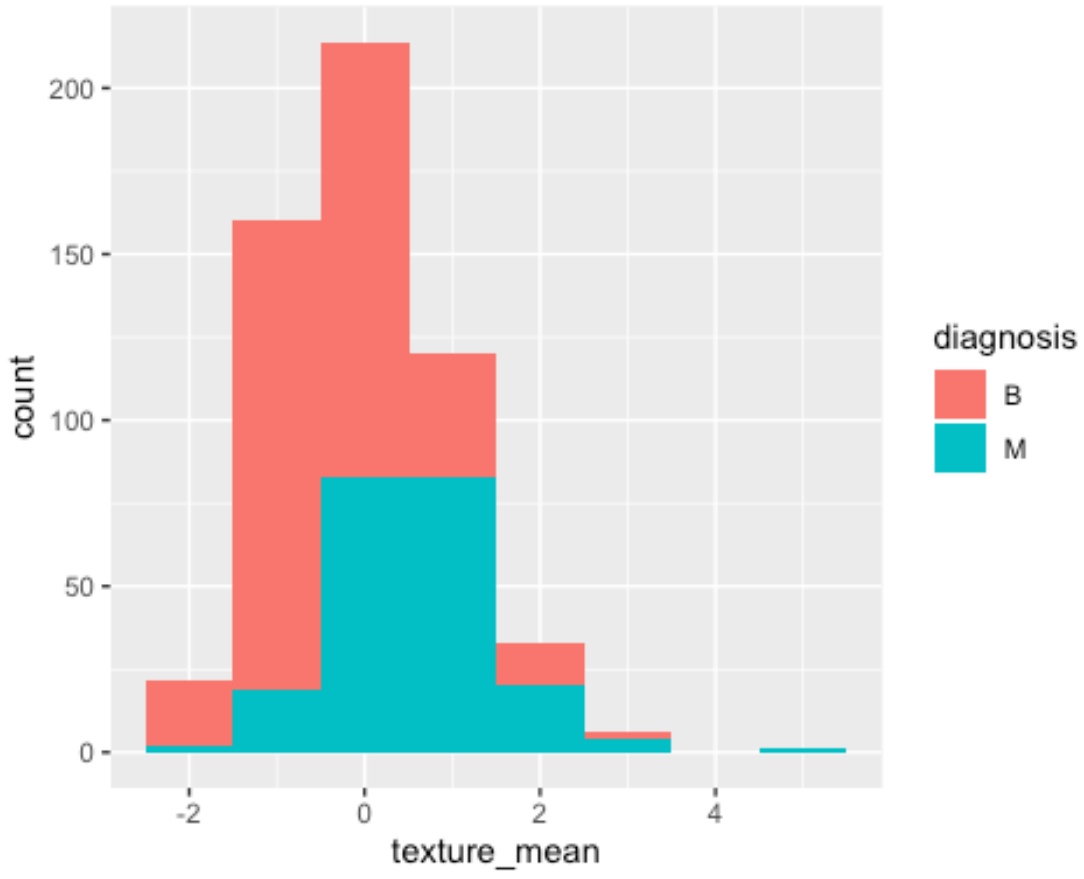
Observamos que radius_mean se relaciona fuertemente con perimeter_mean (0.997), area_mean (0.988) y concave.points_mean (0.817). concave.points_mean se relaciona fuertemente con concavity_mean (0.9189655), perimeter_mean (0.9189655) y compactness_mean (0.8265946).

Procedemos a graficar las distintas variables junto con la variable clasificatoria para hacer una primera estimación de cuales serán las variables con más peso.

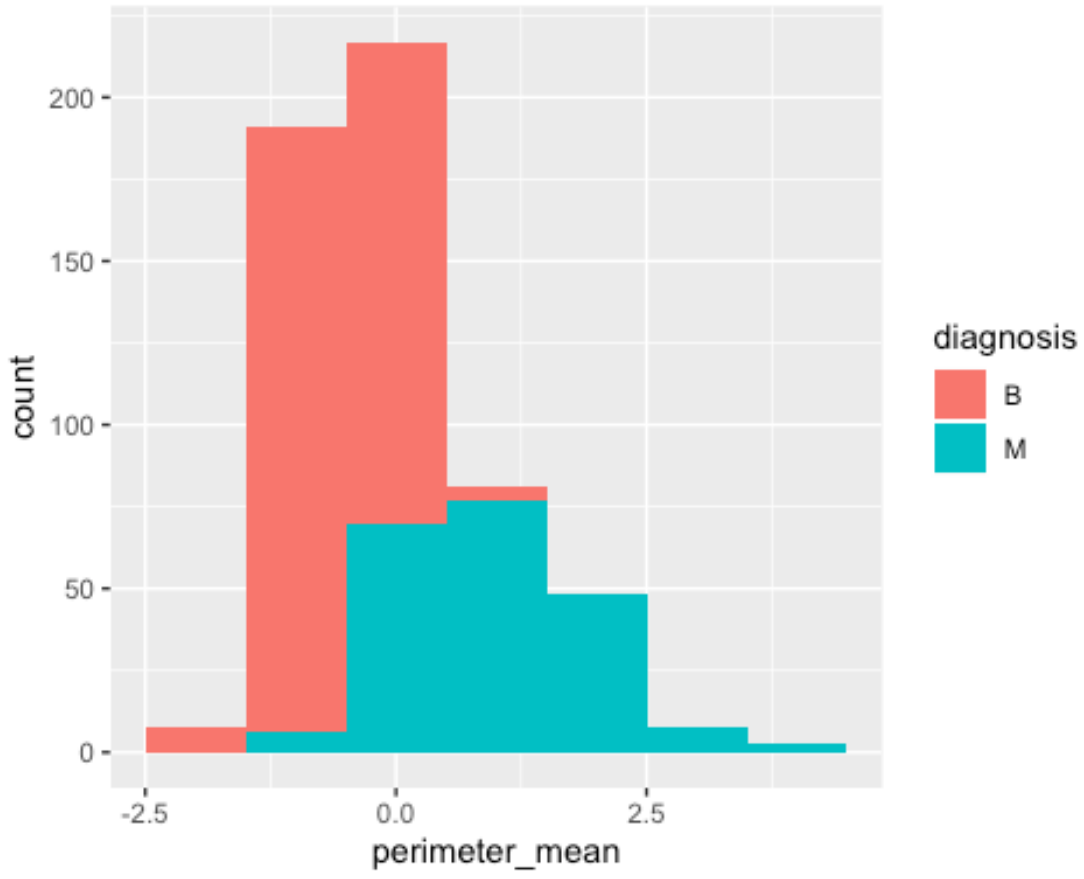
```
ggplot(data = data_scaled,aes(x=radius_mean,fill=diagnosis))
+geom_histogram(binwidth =1)
```



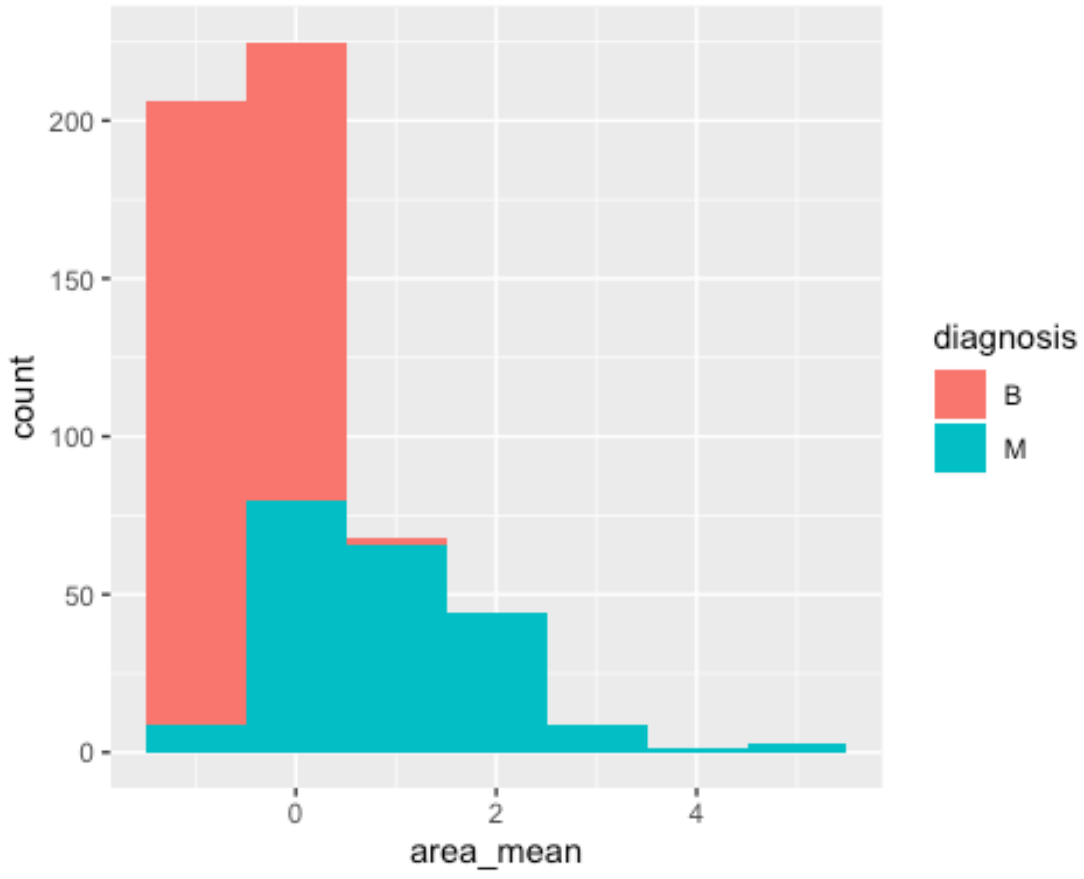

```
ggplot(data = data_scaled,aes(x=texture_mean,fill=diagnosis))
+geom_histogram(binwidth =1)
```



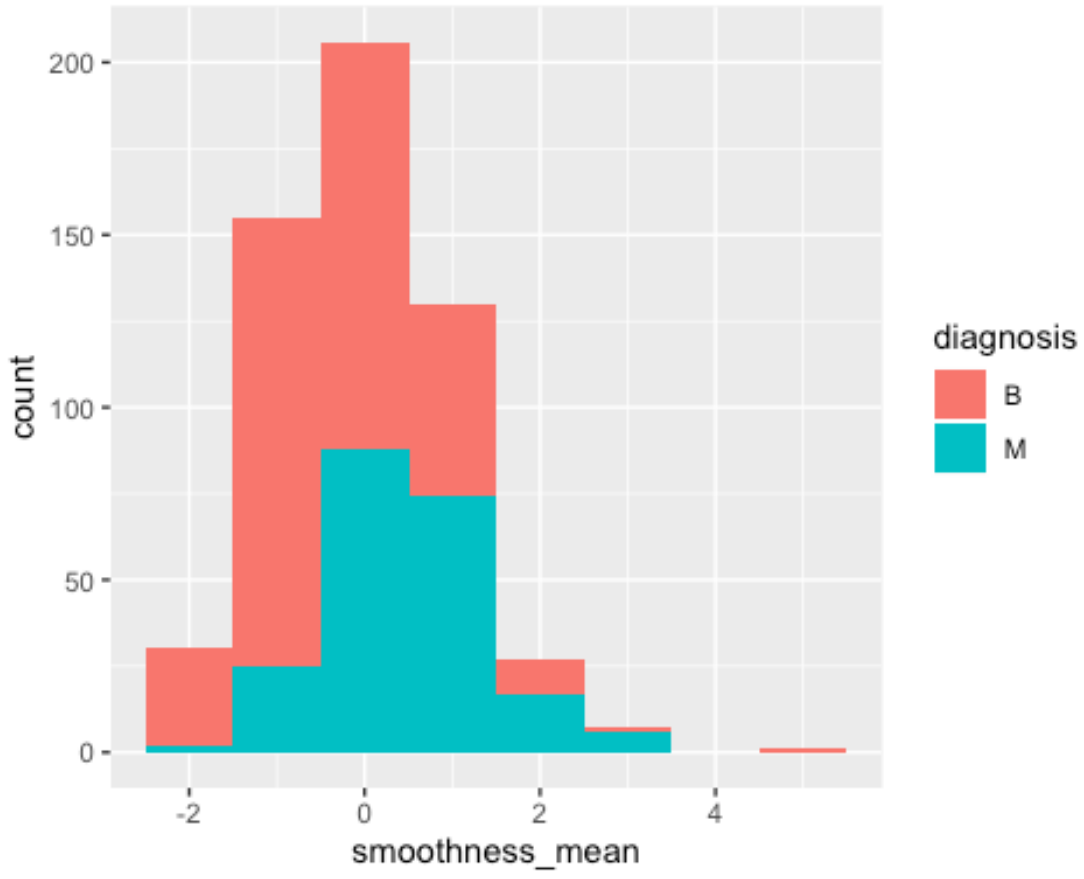

```
ggplot(data = data_scaled,aes(x=perimeter_mean,fill=diagnosis))
+geom_histogram(binwidth =1)
```



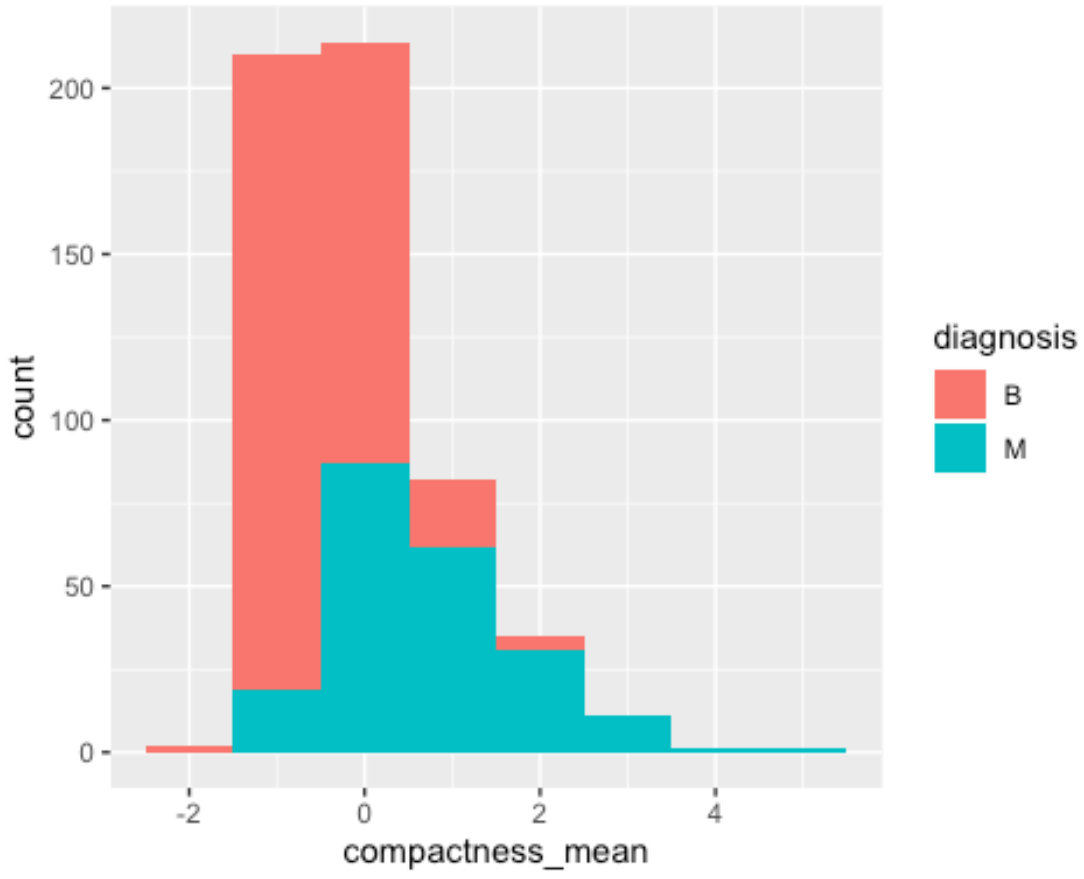

```
ggplot(data = data_scaled,aes(x=area_mean,fill=diagnosis))
+geom_histogram(binwidth =1)
```



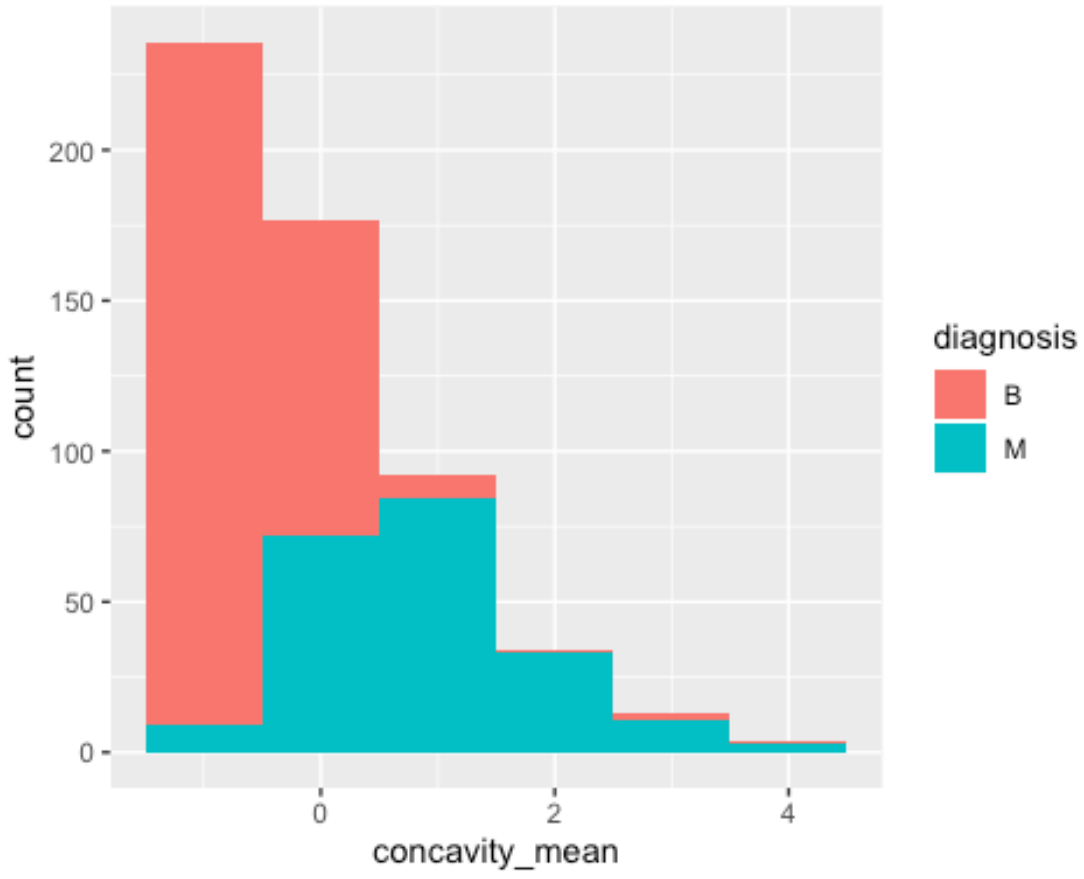

```
ggplot(data = data_scaled,aes(x=smoothness_mean,fill=diagnosis))
+geom_histogram(binwidth =1)
```



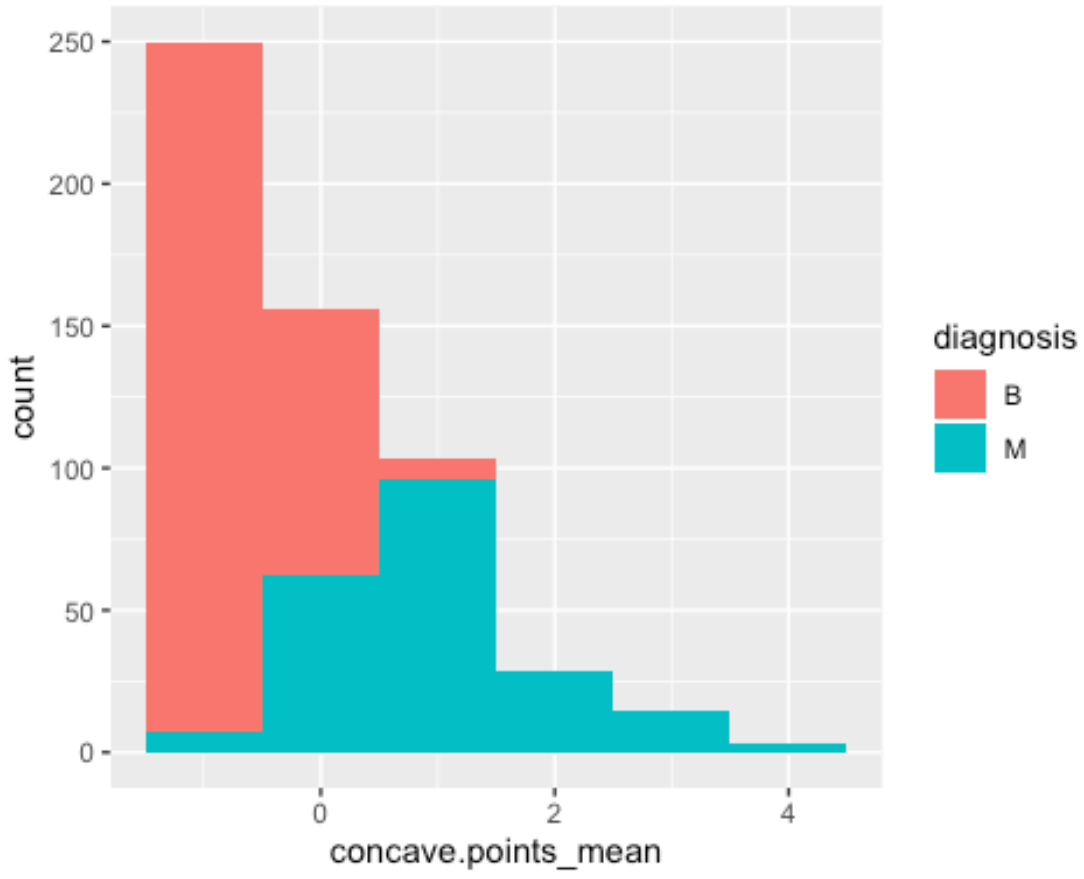

```
ggplot(data = data_scaled,aes(x=compactness_mean,fill=diagnosis))
+geom_histogram(binwidth =1)
```



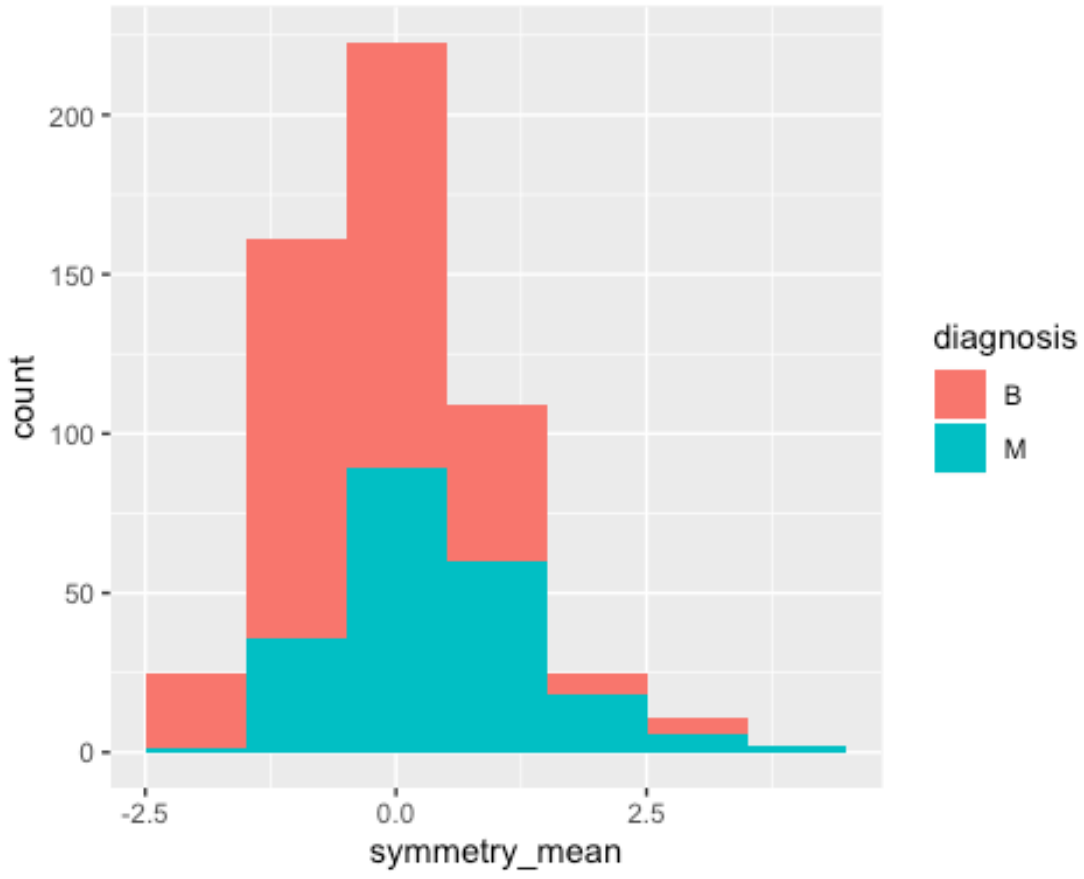

```
ggplot(data = data_scaled,aes(x=concavity_mean,fill=diagnosis))
+geom_histogram(binwidth =1)
```



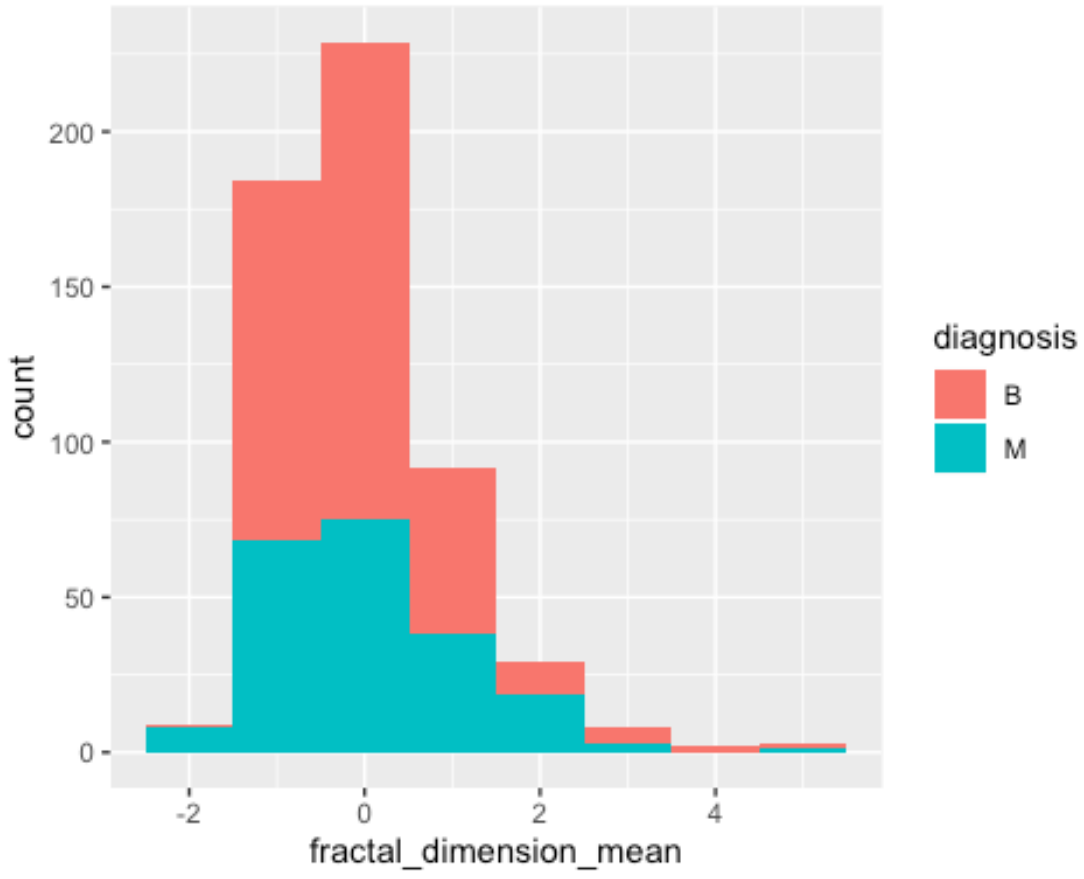

```
ggplot(data = data_scaled,aes(x=concave.points_mean,fill=diagnosis))
+geom_histogram(binwidth =1)
```




```
ggplot(data = data_scaled,aes(x=symmetry_mean,fill=diagnosis))
+geom_histogram(binwidth =1)
```




```
ggplot(data =
data_scaled,aes(x=fractal_dimension_mean,fill=diagnosis))
+geom_histogram(binwidth =1)
```



Se observa que radius_mean, perimeter_mean, area_mean, concave.points_mean y compactness_mean podrían ser relevantes para explicar el diagnóstico. Ya que a partir de un cierto nivel solo encontramos registros de un tipo u de otro.

Comprobación de la normalidad y homogeneidad de la varianza

Para comprobar la normalidad utilizaremos el test Saphiro

```
shapiro.test(data_scaled$radius_mean)

##
##  Shapiro-Wilk normality test
##
```

```
## data:  data_scaled$radius_mean
## W = 0.93534, p-value = 8.644e-15
```

```
shapiro.test(data_scaled$texture_mean)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data_scaled$texture_mean
## W = 0.97703, p-value = 1.169e-07
```

```
shapiro.test(data_scaled$perimeter_mean)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data_scaled$perimeter_mean
## W = 0.92972, p-value = 1.762e-15
```

```
shapiro.test(data_scaled$area_mean)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data_scaled$area_mean
## W = 0.85468, p-value < 2.2e-16
```

```
shapiro.test(data_scaled$smoothness_mean)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data_scaled$smoothness_mean
## W = 0.98488, p-value = 1.593e-05
```

```
shapiro.test(data_scaled$compactness_mean)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data_scaled$compactness_mean
## W = 0.91747, p-value < 2.2e-16
```

```
shapiro.test(data_scaled$concavity_mean)

##
##  Shapiro-Wilk normality test
##
## data:  data_scaled$concavity_mean
## W = 0.86574, p-value < 2.2e-16

shapiro.test(data_scaled$concave.points_mean)

##
##  Shapiro-Wilk normality test
##
## data:  data_scaled$concave.points_mean
## W = 0.8862, p-value < 2.2e-16

shapiro.test(data_scaled$symmetry_mean)

##
##  Shapiro-Wilk normality test
##
## data:  data_scaled$symmetry_mean
## W = 0.97019, p-value = 3.329e-09

shapiro.test(data_scaled$fractal_dimension_mean)

##
##  Shapiro-Wilk normality test
##
## data:  data_scaled$fractal_dimension_mean
## W = 0.92142, p-value < 2.2e-16
```

De acuerdo con el test de Shapiro, no podemos asumir normalidad en ninguna de las variables, ya que el p-valor es menor que 0.05. No obstante de acuerdo con el teorema del limite central, para muestras grandes (n > 30) podemos asumir normalidad.

Aplicación de pruebas estadísticas para comparar los grupos de datos

A continuación vamos a utilizar el algoritmo k-means para crear un modelo predictivo. Utilizaremos k = 2 ya que sabemos de antemano que k sería 2

```
data_clusters2 <- kmeans(data_scaled[, 2:11], 2)

table(data_clusters2$cluster,data_scaled$diagnosis)

##
##        B   M
##   1    4 162
##   2 340  50
```

Podemos ver que el modelo con k-means, ha clasificado correctamente 340 resultados benignos y 162 resultados malignos, con una precisión del modelo de cerca del 90%.

A continuación vamos a aplicar un modelo de regresión logística.

```
logit_model <-
glm(formula=diagnosis~radius_mean+texture_mean+perimeter_mean+area_mea
n+smoothness_mean+compactness_mean+concavity_mean+concave.points_mean+
symmetry_mean+fractal_dimension_mean, data=data_scaled,
family=binomial)

summary(logit_model)

##
## Call:
## glm(formula = diagnosis ~ radius_mean + texture_mean +
perimeter_mean +
##     area_mean + smoothness_mean + compactness_mean + concavity_mean
+
##     concave.points_mean + symmetry_mean + fractal_dimension_mean,
##     family = binomial, data = data_scaled)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.94745  -0.15493  -0.04012   0.00490   2.91079
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)               0.66975    0.56525   1.185   0.2361
## radius_mean              -7.18764   12.95109  -0.555   0.5789
## texture_mean              1.63427    0.27384   5.968  2.4e-09 ***
```

```
## perimeter_mean          -1.72113   12.11280  -0.142   0.8870
## area_mean               14.01933    5.90537   2.374   0.0176 *
## smoothness_mean          1.05575    0.44371   2.379   0.0173 *
## compactness_mean        -0.08063    1.06982  -0.075   0.9399
## concavity_mean           0.66370    0.64662   1.026   0.3047
## concave.points_mean      2.57527    1.09796   2.346   0.0190 *
## symmetry_mean            0.44706    0.29168   1.533   0.1253
## fractal_dimension_mean -0.47588    0.60437  -0.787   0.4310
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 739.14  on 555  degrees of freedom
## Residual deviance: 146.02  on 545  degrees of freedom
## AIC: 168.02
##
## Number of Fisher Scoring iterations: 9
```

```
library(caret)
confusionMatrix(table(predict(logit_model, type="response") >=
0.5,data_scaled$diagnosis == "M"))
```

```
## Confusion Matrix and Statistics
##
##
##         FALSE TRUE
##   FALSE   334   19
##   TRUE     10  193
##
##                Accuracy : 0.9478
##                  95% CI : (0.9259, 0.9648)
##     No Information Rate : 0.6187
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8885
##
##  Mcnemar's Test P-Value : 0.1374
```

```
##
##             Sensitivity : 0.9709
##             Specificity : 0.9104
##          Pos Pred Value : 0.9462
##          Neg Pred Value : 0.9507
##              Prevalence : 0.6187
##          Detection Rate : 0.6007
##    Detection Prevalence : 0.6349
##       Balanced Accuracy : 0.9407
##
##        'Positive' Class : FALSE
##
```

Conclusiones

Podemos concluir que el modelo de regresión logística clasifica de mejor forma los datos que el modelo basado en k-means. Con el primero obtenemos una precisión cercana al 95% mientras que con k-means nos quedamos en el 90%.

Contribuciones al trabajo

CONTRIBUCIONES	FIRMA
Investigación previa	Victor Mascarell Ascó
Redacción de las respuestas	Victor Mascarell Ascó
Desarrollo código	Victor Mascarell Ascó