Tipología y ciclo de vida de los datos
Práctica 2
Yaiza Santana Santana

ÍNDICE

Ejercicio 1.

1.1 Descripción del dataset.

Para esta práctica se ha escogido el *dataset* propuesto sobre la calidad de las variantes rojas del vino portugués [1].

Esta fuente de datos, única, consta de 12 variables y 1599 registros. Los parámetros que definen cada fila del conjunto de datos son:

- **acidez fija**: la mayoría de los ácidos involucrados con el vino o fijos o no volátiles (no se evaporan fácilmente)
- **acidez volátil**: la cantidad de ácido acético en el vino, que en niveles demasiado altos puede llevar a un sabor desagradable a vinagre.
- **ácido cítrico**: dará información sobre la 'frescura' y sabor a los vinos
- **azúcar residual**: la cantidad de azúcar restante después de que se detiene la fermentación, es raro encontrar vinos con menos de 1 gramo / litro y los vinos con más de 45 gramos / litro se consideran dulces.
- **cloruros**: la cantidad de sal en el vino.
- **dióxido de azufre libre**: datos sobre el crecimiento microbiano y la oxidación del vino.
- **dióxido de azufre total**: en bajas concentraciones, es mayormente indetectable en el vino, pero a concentraciones de SO2 libres superiores a 50 ppm, el SO2 se hace evidente en la nariz y el sabor del vino.
- **densidad**: la densidad del agua es cercana a la del agua según el porcentaje de alcohol y contenido de azúcar
- **pH**: describe qué tan ácido o básico es un vino en una escala de 0 (muy ácido) a 14 (muy básico); La mayoría de los vinos están entre 3-4 en la escala de pH.
- **sulfatos**: un aditivo para el vino que puede contribuir a los niveles de gas de dióxido de azufre (S02), que actúa como un antimicrobiano y antioxidante.
- **alcohol**: el porcentaje de alcohol del vino.
- **calidad**: variable de salida (basada en datos sensoriales, puntuación entre 0 y 10)

1.2 ¿Qué pregunta/problema se pretende responder?

Para esta práctica se pretende resolver preguntas sobre qué parámetros influyen más en la calidad del vino según los usuarios que han clasificado los registros a través de sus sentidos del sabor y olfato.

Por ejemplo, podemos averiguar si para la calificación de la calidad del vino influye que al comensal le sea agradable de sabor (estudio con la acidez volátil), o si tiene que ver con la cantidad de alcohol que tiene el vino.

Ejercicio 2.

2.1 Fuente de datos.

En este caso no hemos tenido que fusionar diferentes fuentes de datos. A continuación, se muestra con detalle las características del *dataset*.

Estos datos los descargamos en formato .csv, y lo transformamos en dataframe para poder usar la potencialidad de las amplias librerías que disponemos con Python para el análisis de datos y resolver los enunciados propuestos.

El lenguaje de programación escogido ha sigo Python, y se ha ejecutado el script con el entorno de desarrollo ***Pycharm***, con la versión **3.7.1 de Python**.

2.2 Exploración de los datos.

Conocer el número de muestras con el que cuenta nuestro conjunto de datos es muy importante, por ejemplo para saber qué test de contraste de hipótesis usar, o qué relevancia podrán tener los *outliers* encontrados.

Como se comentó anteriormente, el dataset contiene 1599 registros:

```
# Controlando la cantidad de registros (1599)
print(df['quality'].count())
```

Por consola podemos comprobarlo:

```
[5 rows x 12 columns]
1599
```

Además, comprobamos que las propiedades son las mismas que las explicadas en la web:

```
# para conocer las propiedades
print(df.columns)
```

En consola:

```
Index(['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH', 'sulphates', 'alcohol', 'quality'],
      dtype='object')
```

Efectivamente, son las 12 variables descritas en el comienzo de esta práctica.

Es importante saber el tipo de las variables que ha asignado Python, ya que según éste, podremos operar de una forma u otra, realizar transformación de los tipos de los datos, etc:

```
# para conocer el tipo de las columnas
print(df.dtypes)
```

En consola:

```
[8 rows x 12 columns]
fixed acidity           float64
volatile acidity        float64
citric acid             float64
residual sugar          float64
chlorides               float64
free sulfur dioxide     float64
total sulfur dioxide    float64
density                 float64
pH                      float64
sulphates               float64
alcohol                 float64
quality                   int64
dtype: object
```

Vemos que todas las variables son flotantes, excepto la puntuación que es un número entero (del 0 al 10), tal y como esperábamos.

Por último, es interesante también saber medidas descriptivas de las variables de la siguiente manera:

```
# Ahora veamos algunas estadísticas de nuestros datos
print(df.describe())
```

En consola:

```
       fixed acidity  volatile acidity  ...      alcohol      quality
count    1599.000000       1599.000000  ...  1599.000000  1599.000000
mean        8.319637          0.527821  ...    10.422983     5.636023
std         1.741096          0.179060  ...     1.065668     0.807569
min         4.600000          0.120000  ...     8.400000     3.000000
25%         7.100000          0.390000  ...     9.500000     5.000000
50%         7.900000          0.520000  ...    10.200000     6.000000
75%         9.200000          0.640000  ...    11.100000     6.000000
max        15.900000          1.580000  ...    14.900000     8.000000
```

Con estos datos podemos hacernos una idea de valores como máximos, la media, mediana, o rango intercuartílico de cada variable.

2.3 Selección de los datos.

Tal como sugiere la web de la fuente de datos, se ha dividido el conjunto en dos grupos; el vino bueno que tiene puntuación de calidad igual o mayor a 7, y el resto, que es el vino no bueno:

```
# -------------------- SELECCIÓN DE DATOS -------------------- #
# vamos a hacer grupos de datos en tres clases dependiendo de la calidad del vino
vino_bueno = df[df['quality'] >= 7.0]
vino_no_bueno = df[df['quality'] < 7.0]

print('Hay ' + str(len(vino_no_bueno)) + ' vinos no buenos')
print('Hay ' + str(len(vino_bueno)) + ' vinos buenos')
```

En consola:

```
Hay 1382 vinos no buenos
Hay 217 vinos buenos
```

Se puede comprobar que el total de la suma de los dos grupos hacen 1599 registros.

Ejercicio 3.

3.1.1 ¿Los datos contienen ceros o elementos vacíos?

Se ha comprobado que este conjunto de datos no contenía elementos vacíos. Para ello hemos usado la función ***isnull*** para todos los parámetros. Si esta función devuelve True, es que existen algunos datos vacíos, si devuelve False, es que no:

```
# Apartado 3. LIMPIEZA DE DATOS
# ------------------- SABER SI HAY QUE RELLENAR DATOS ---------------- #
# Controlando valores nulos; si resulta False el dataset esta limpio de valores faltantes
print(df.isnull().any().any())
```

Nos ha devuelto en consola:

```
False
```

Por lo tanto, el *dataset* está limpio de valores faltantes.

3.1.2 ¿Cómo gestionarías cada uno de estos casos?

En el caso de que hubieran existido, se hubieran sustituido por la media (para variables simétricas), mediana (para variables asimétricas), o medias winsorizadas si hiciera falta recurrir a medidas robustas [2].

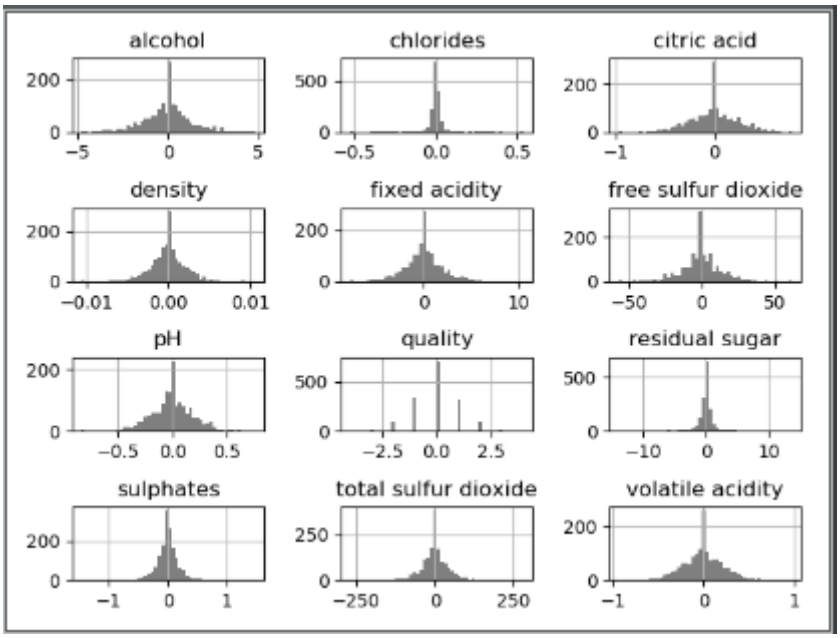
3.2 Identificación y tratamiento de valores extremos.

Recordemos de la PEC anterior que, un valor extremo Es un dato que corresponde a un valor mínimo o máximo de una variable del conjunto de datos o del mismo conjunto de datos. Detenernos en el estudio de estos es vital para el proyecto de datos, ya que pueden afectar al análisis de los datos, o incluso ser un *outlier,* y según el caso al que nos enfrentemos, pueden verse afectadas medidas tan importantes como la correlación o regresión, entre otros.

Se muestra a continuación como para cada variable se visualiza el histograma, los outliers para cada variable, y el diagrama de caja que nos dice si la variable será simétrica o no:

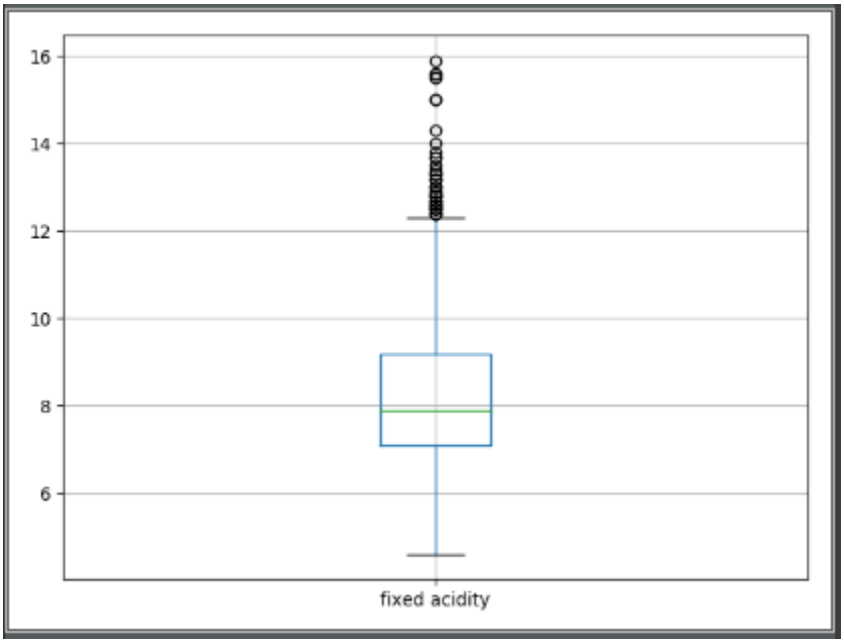
```
# ---------------------------  SABER SI HAY RUIDO ---------------------- #
# histograma [2]
plt.figure();
df.diff().hist(color='k', alpha=0.5, bins=50)
plt.show()

# outliers
for i in range(len(df.columns)):
    # descubrimiento de los outliers
    outliers = outliers_modified_z_score(df[df.columns[i]])
    print('Outliers de la columna ' + str(df.columns[i]) + ':')
    print(df.iloc[list(np.asarray(outliers)[0]),[i]])
    print(list(np.asarray(outliers)[0]))

    # diagramas de cajas para ver si son datos simétricos [3]
    boxplot = df.boxplot(column=[df.columns[i]])
    plt.show()
```

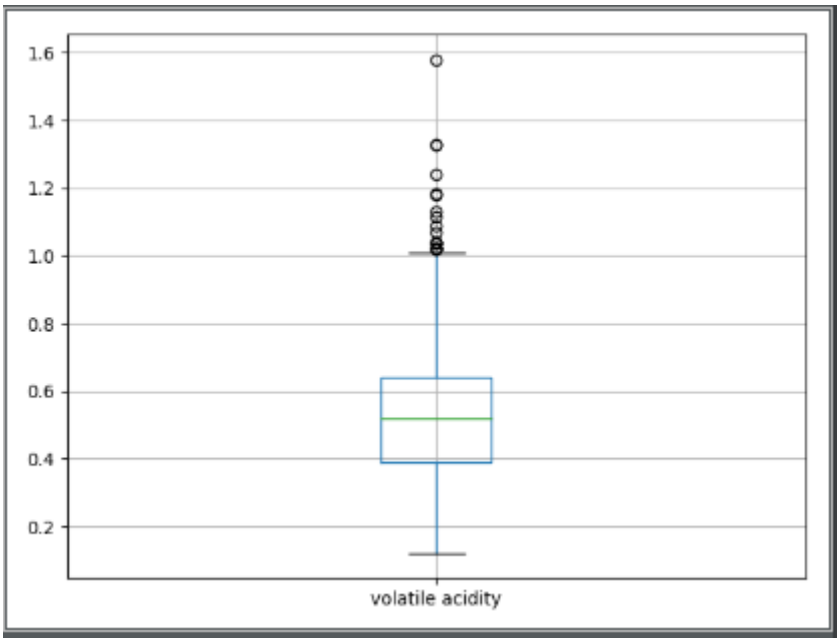
En SciView (herramienta de visualización de Pycharm):

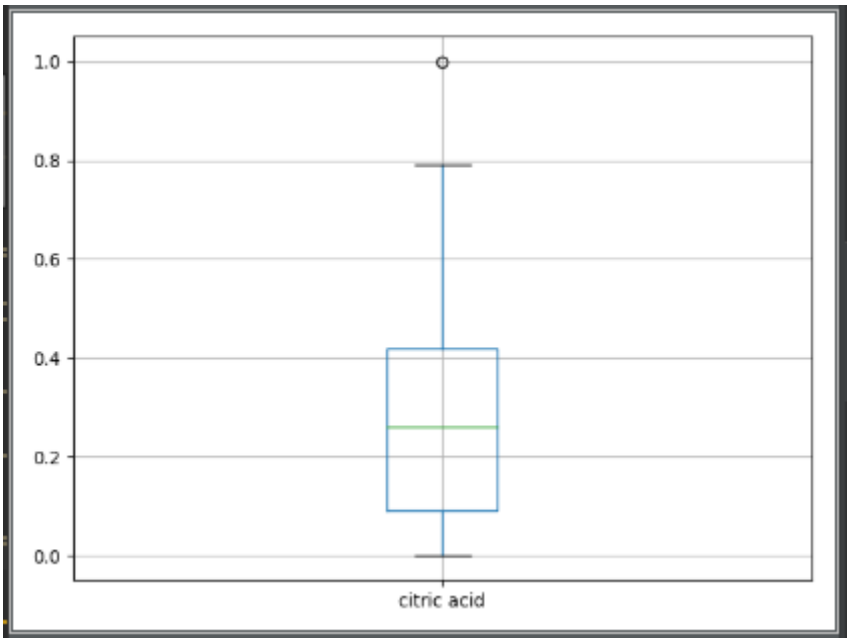


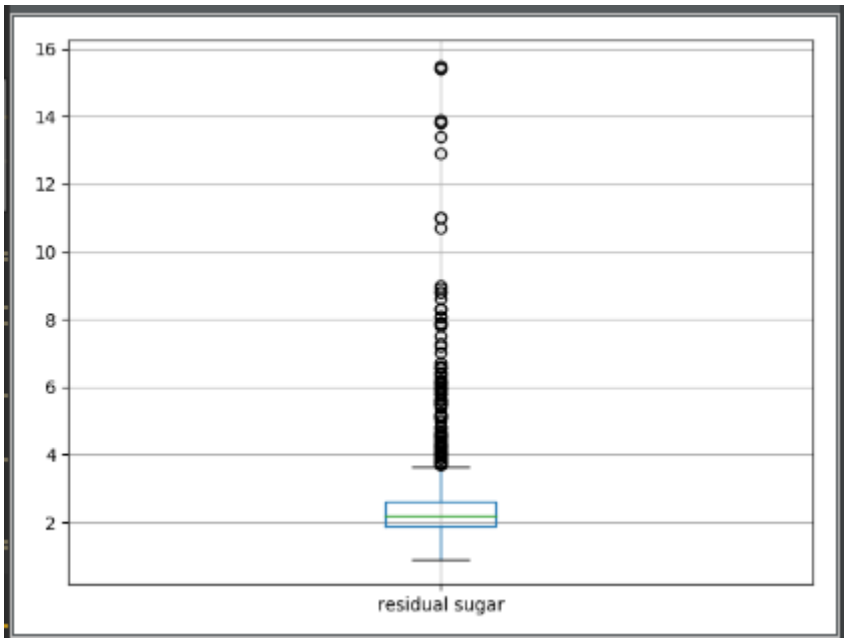
Histogramas para cada variable del conjunto de datos.

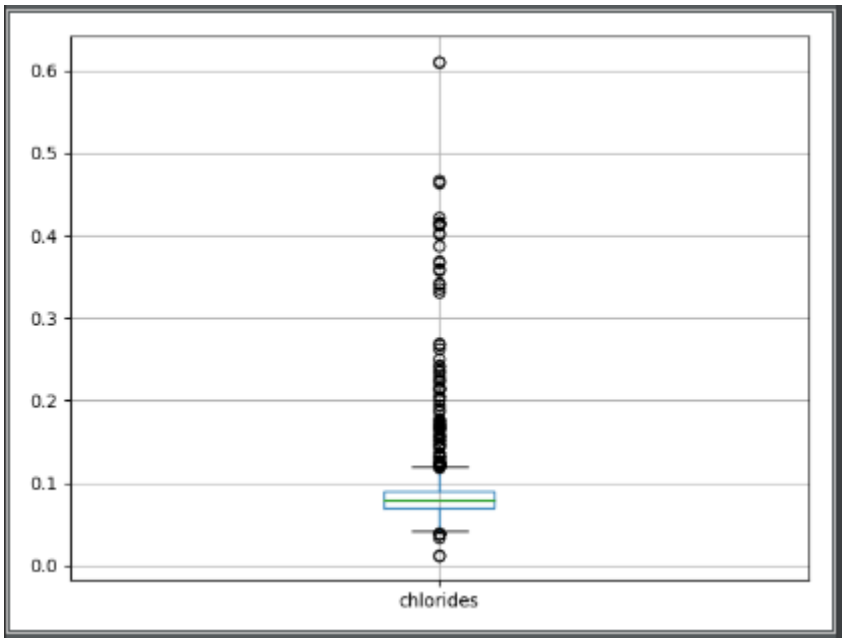
Los diagramas de cajas son los siguientes:

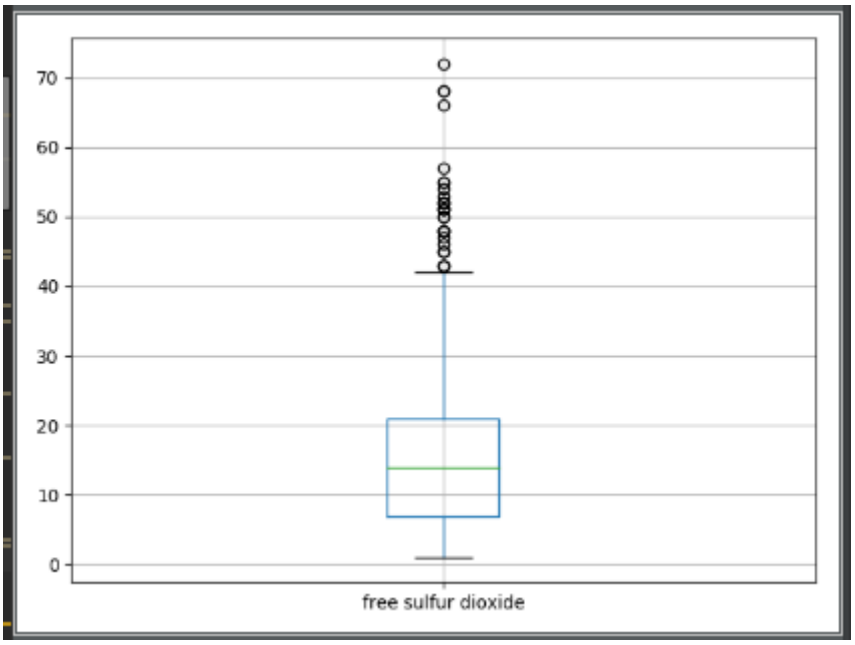


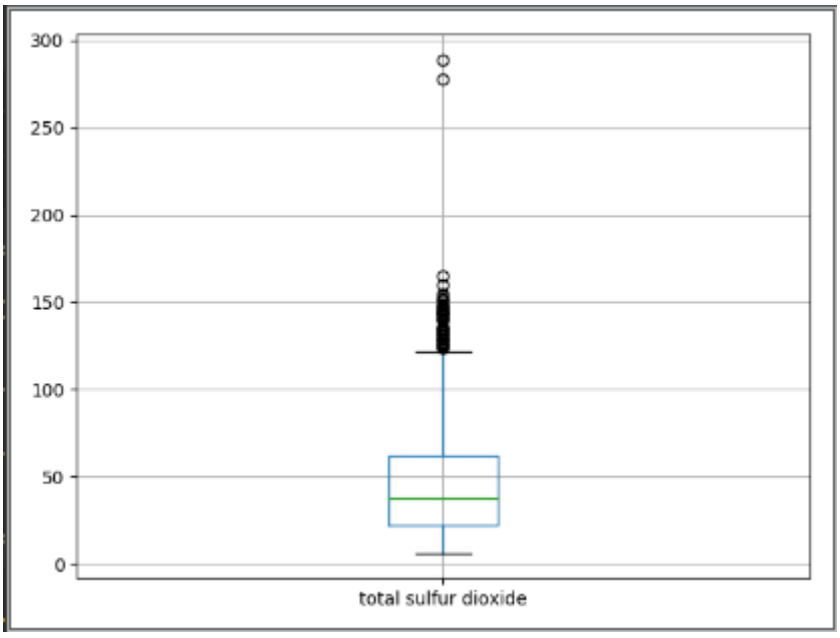

1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
volatile acidity

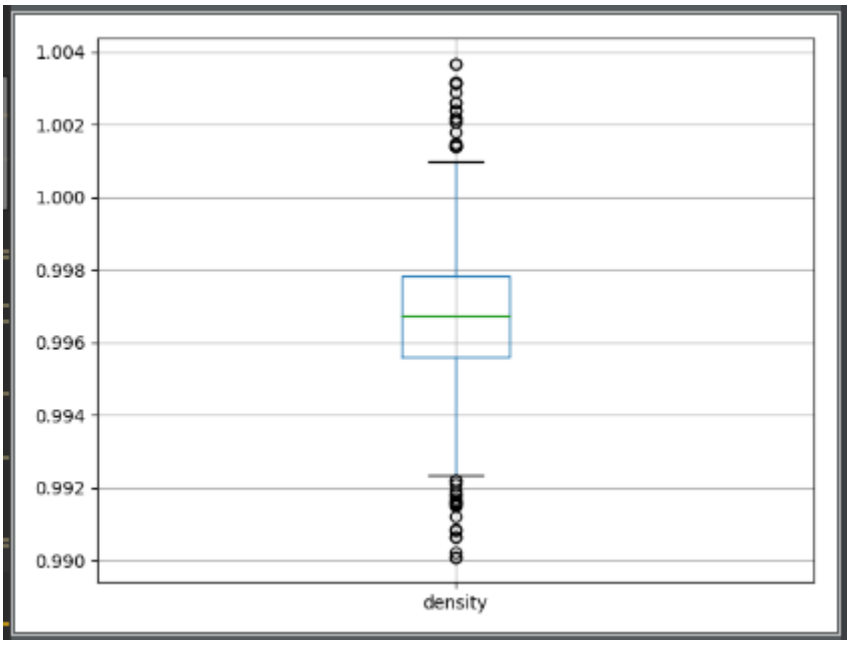

1.0
0.8
0.6
0.4
0.2
0.0
citric acid

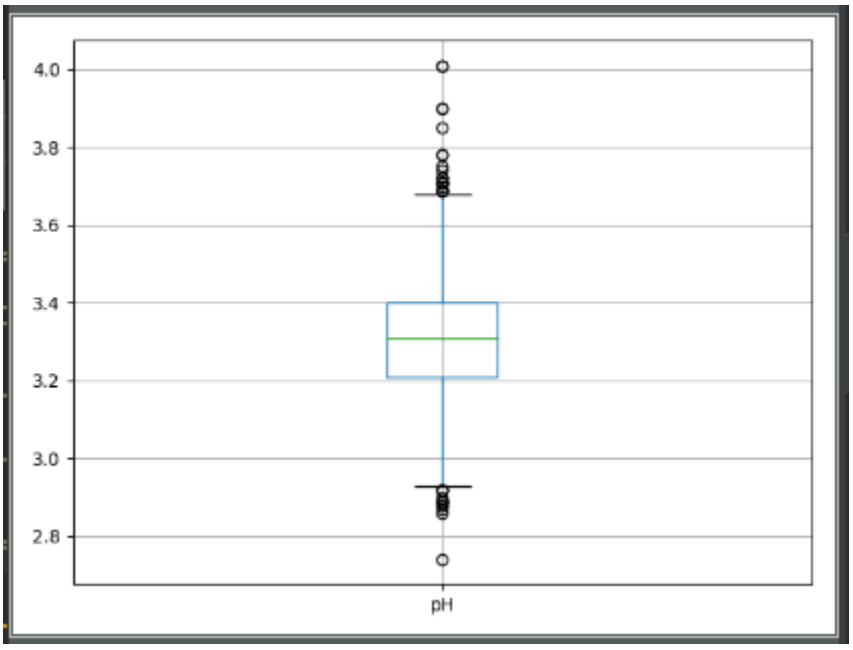

16
14
12
10
8
6
4
2
residual sugar

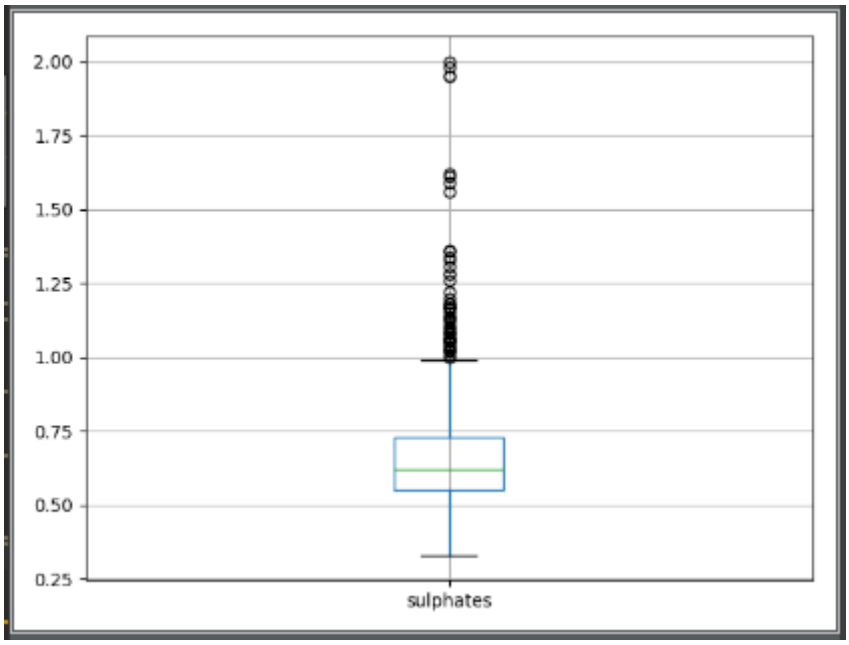

0.6
0.5
0.4
0.3
0.2
0.1
0.0
chlorides

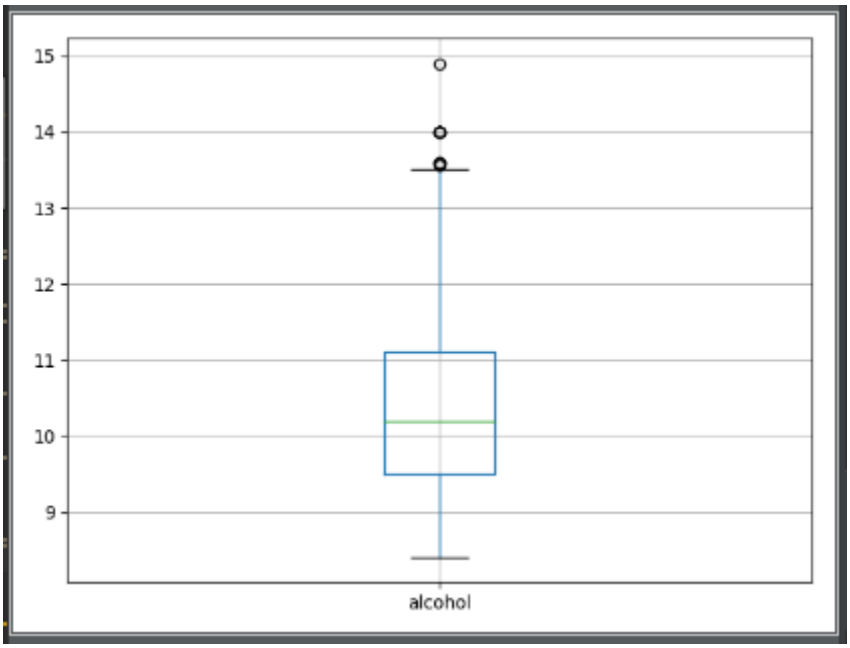

70
60
50
40
30
20
10
0
free sulfur dioxide

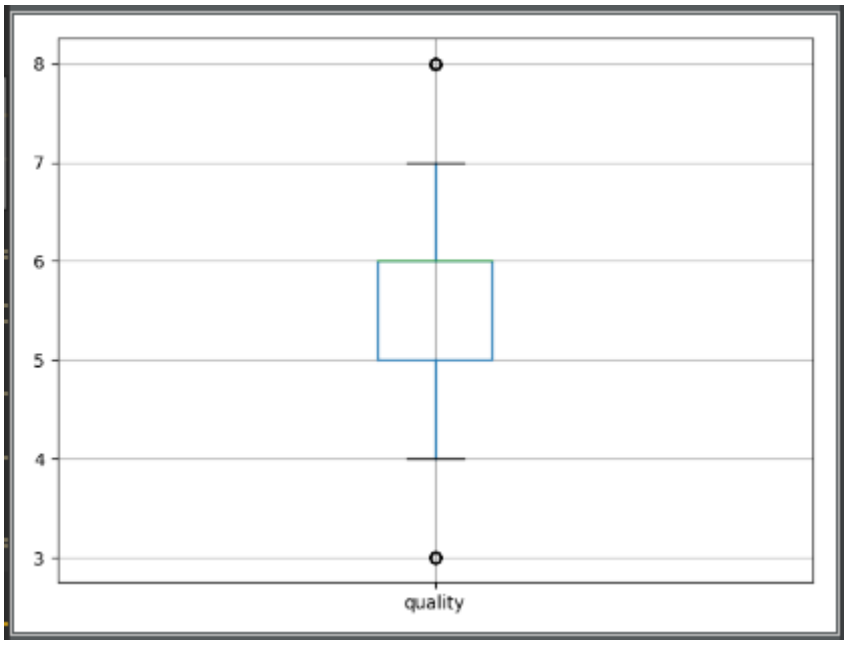

300
250
200
150
100
50
0
total sulfur dioxide


1.004
1.002
1.000
0.998
0.996
0.994
0.992
0.990
density


4.0
3.8
3.6
3.4
3.2
3.0
2.8
pH


2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
sulphates


15
14
13
12
11
10
9
alcohol



Según los diagramas de cajas que hemos visto, si hubiera que modificar los *outliers* por algún valor para que no modifique drásticamente los resultados, según la simetría de las variables sería:

Variable	Valor	Total outliers
Fixed acidity	Mediana	20
Volatile acidity	Media	6
Citric acid	Media	0
Residual sugar	Mediana	150
Chlorides	Media	80
Free sulfur dioxide	Media	16
Total sulfur dioxide	Mediana	38
Density	Media	14
pH	Media	6
Sulphates	Mediana	52
Alcohol	Media	8
Quality	Media	0

Han resultado los siguientes valores ***outliers***:

```
Outliers de la columna fixed acidity:
     fixed acidity
243            15.0
244            15.0
294            13.3
328            13.4
347            13.8
353            13.5
374            14.0
381            13.7
391            13.7
442            15.6
509            13.3
544            14.3
554            15.5
555            15.5
557            15.6
601            13.2
603            13.2
611            13.2
652            15.9
680            13.3
```

```
Outliers de la columna volatile acidity:
      volatile acidity
126              1.330
127              1.330
672              1.240
690              1.185
1299             1.580
1312             1.180
[126, 127, 672, 690, 1299, 1312]
```

```
Outliers de la columna residual sugar:
      residual sugar
9               6.10
11              6.10
14              3.80
15              3.90
18              4.40
33             10.70
35              5.50
39              5.90
40              5.90
55              3.80
57              5.10
64              4.65
65              4.65
154             5.50
155             5.50
156             5.50
157             5.50
163             7.30
164             7.20
192             3.80
215             5.60
1478            5.70
1501            4.30
1514            4.10
1515            4.10
1540            4.40
1558            6.70
1574           13.90
1577            5.10
1589            7.80

[150 rows x 1 columns]
```

```
Outliers de la columna chlorides:
      chlorides
14        0.176
15        0.170
17        0.368
19        0.341
38        0.172
42        0.332
81        0.464
83        0.401
106       0.467
120       0.178
125       0.146
147       0.236
151       0.610
169       0.360
181       0.270
226       0.337
240       0.263
```

```
1435      0.214
1436      0.169
1474      0.205
1476      0.205
1558      0.235
1570      0.230

[80 rows x 1 columns]
```

```
Outliers de la columna free sulfur dioxide:
      free sulfur dioxide
14                   52.0
15                   51.0
396                  68.0
400                  68.0
584                  54.0
925                  53.0
926                  52.0
982                  51.0
1131                 57.0
1244                 72.0
1295                 51.0
1296                 51.0
1358                 52.0
1434                 55.0
1435                 55.0
1558                 66.0
```

Outliers de la columna total sulfur dioxide:

	total sulfur dioxide
14	145.0
15	148.0
86	136.0
90	140.0
91	136.0
92	133.0
109	153.0
130	134.0
145	141.0
188	143.0
189	144.0
201	145.0
219	144.0
313	135.0
354	165.0
415	134.0
515	151.0
522	133.0
523	142.0
591	149.0
636	147.0
637	145.0
649	148.0
651	155.0
672	151.0
684	152.0
741	139.0
771	143.0
772	144.0
1079	278.0
1081	289.0
1131	135.0
1244	160.0
1400	141.0
1401	141.0
1419	133.0
1493	147.0
1496	147.0

```
Outliers de la columna density:
      density
442   1.00320
554   1.00315
555   1.00315
557   1.00315
836   0.99064
837   0.99064
889   1.00289
1017  0.99007
1018  0.99007
1114  0.99020
1269  0.99080
1270  0.99084
1434  1.00369
1435  1.00369
```

```
Outliers de la columna pH:
        pH
45    3.90
95    3.85
151   2.74
695   3.90
1316  4.01
1321  4.01
```

Outliers de la columna sulphates:

	sulphates
13	1.56
17	1.28
19	1.08
43	1.20
79	1.12
81	1.28
83	1.14
86	1.95
88	1.22
91	1.95
92	1.98
106	1.31
151	2.00
161	1.08
169	1.59
226	1.61
240	1.09
258	1.26
281	1.08
339	1.36
340	1.18
369	1.13
372	1.04
376	1.11
377	1.13
415	1.07
451	1.06
477	1.06
482	1.05
483	1.06
503	1.04
504	1.05
515	1.14
614	1.36
639	1.36
689	1.05
692	1.17
723	1.62
754	1.06
795	1.18
852	1.07
1051	1.34
1158	1.16
1165	1.10
1260	1.15
1288	1.17
1289	1.17
1319	1.33
1367	1.18
1370	1.17
1372	1.17
1403	1.10

```
Outliers de la columna alcohol:
      alcohol
142      14.0
144      14.0
467      14.0
588      14.0
652      14.9
821      14.0
1269     14.0
1270     14.0
```

```
Outliers de la columna quality:
Empty DataFrame
Columns: [quality]
Index: []
[]
```

Tras revisar los rangos de valores que deberían tener las variables, por lo pronto decidimos dejar los outliers. Más adelante veremos si no considerando estos registros, mejoran las correlaciones y podemos obtener un modelo mejor, y así también estudiar estos valores a parte [3]

Ejercicio 4.

4.1 Selección de grupos de datos que se quieren analizar.

Ya los hemos definido anteriormente (vinos buenos y no buenos).

4.2 Comprobación de la normalidad y homogeneidad de la varianza.

Para el cálculo de la correlación, contraste de hipótesis, regresión, etc, es necesario primeramente, saber si las variables provienen de una distribución normal, y si la varianza es igual entre variables.

Gracias a las librerías de Python, realizamos la prueba de normalidad de Anderson Darling, una de las técnicas que ofrecen mejores resultados:

```
# ---------- COMPROBACIÓN DE LA NORMALIDAD Y HOMOGENEIDAD DE LA VARIANZA---------- #

# pruebas de normalidad [4]
for i in range(len(df.columns)):

    print(df.columns[i])
    anderson_results = scipy.stats.anderson(df[df.columns[i]], dist='norm')
    print(anderson_results)
    if (anderson_results[1][2] > 0.05): # [5]
        print('No podemos rechazar la hipótesis nula de que la variable ' + str(df.columns[i])
              + ' proviene de una distribución normal')
    else:
        print('Podemos rechazar la hipótesis nula de que la variable ' + str(
            df.columns[i]) + ' proviene de una distribución normal')
```

En consola:

```
AndersonResult(statistic=28.142957509406642, critical_values=array([0.575, 0.654, 0.785, 0.916, 1.089]), significance_level=array([15. , 10. ,  5. ,  2.5,  1. ]))
No podemos rechazar la hipótesis nula de que la variable fixed acidity proviene de una distribución normal
volatile acidity
AndersonResult(statistic=5.6830748971976845, critical_values=array([0.575, 0.654, 0.785, 0.916, 1.089]), significance_level=array([15. , 10. ,  5. ,  2.5,  1. ]))
No podemos rechazar la hipótesis nula de que la variable volatile acidity proviene de una distribución normal
citric acid
AndersonResult(statistic=17.542087407024383, critical_values=array([0.575, 0.654, 0.785, 0.916, 1.089]), significance_level=array([15. , 10. ,  5. ,  2.5,  1. ]))
No podemos rechazar la hipótesis nula de que la variable citric acid proviene de una distribución normal
residual sugar
AndersonResult(statistic=188.06444866412653, critical_values=array([0.575, 0.654, 0.785, 0.916, 1.089]), significance_level=array([15. , 10. ,  5. ,  2.5,  1. ]))
No podemos rechazar la hipótesis nula de que la variable residual sugar proviene de una distribución normal
chlorides
AndersonResult(statistic=210.4491870499494, critical_values=array([0.575, 0.654, 0.785, 0.916, 1.089]), significance_level=array([15. , 10. ,  5. ,  2.5,  1. ]))
No podemos rechazar la hipótesis nula de que la variable chlorides proviene de una distribución normal
free sulfur dioxide
AndersonResult(statistic=38.60990999200476, critical_values=array([0.575, 0.654, 0.785, 0.916, 1.089]), significance_level=array([15. , 10. ,  5. ,  2.5,  1. ]))
No podemos rechazar la hipótesis nula de que la variable free sulfur dioxide proviene de una distribución normal
```

```
total sulfur dioxide
AndersonResult(statistic=52.48865143001012, critical_values=array([0.575, 0.654, 0.785, 0.916, 1.089]), significance_level=array([15. , 10. ,  5. ,  2.5,  1. ]))
No podemos rechazar la hipótesis nula de que la variable total sulfur dioxide proviene de una distribución normal
density
AndersonResult(statistic=3.867595192336921, critical_values=array([0.575, 0.654, 0.785, 0.916, 1.089]), significance_level=array([15. , 10. ,  5. ,  2.5,  1. ]))
No podemos rechazar la hipótesis nula de que la variable density proviene de una distribución normal
pH
AndersonResult(statistic=1.8641116106432492, critical_values=array([0.575, 0.654, 0.785, 0.916, 1.089]), significance_level=array([15. , 10. ,  5. ,  2.5,  1. ]))
No podemos rechazar la hipótesis nula de que la variable pH proviene de una distribución normal
sulphates
AndersonResult(statistic=46.93219549223704, critical_values=array([0.575, 0.654, 0.785, 0.916, 1.089]), significance_level=array([15. , 10. ,  5. ,  2.5,  1. ]))
No podemos rechazar la hipótesis nula de que la variable sulphates proviene de una distribución normal
alcohol
AndersonResult(statistic=34.91706402625891, critical_values=array([0.575, 0.654, 0.785, 0.916, 1.089]), significance_level=array([15. , 10. ,  5. ,  2.5,  1. ]))
No podemos rechazar la hipótesis nula de que la variable alcohol proviene de una distribución normal
quality
AndersonResult(statistic=110.63276775926852, critical_values=array([0.575, 0.654, 0.785, 0.916, 1.089]), significance_level=array([15. , 10. ,  5. ,  2.5,  1. ]))
No podemos rechazar la hipótesis nula de que la variable quality proviene de una distribución normal
```

Como podemos comprobar, para un nivel de significación alpha = 0.05, para todas las variables se devuelve un **p-value > alpha**, con lo cual para todas las variables no podemos rechazar la hipótesis nula de que cada variable proviene de una distribución normal.

Para verificar la homogeneidad de las varianzas, se ha usado el test de Fligner-killeen:

```
# prueba de varianzas homogéneas [6]
# print(stats.levene(df['fixed acidity'],df['quality'])) # otra forma

for i in range(len(df.columns)-1):
    fligner_results = scipy.stats.fligner(df['quality'],df[df.columns[i]])
    print(fligner_results)
    print(fligner_results[1])
    if (fligner_results[1] > 0.05):
        print('La varianza de la variable ' + str(df.columns[i]) + ' y la variable quality son iguales')
    else:
        print('La varianza de la variable ' + str(df.columns[i]) + ' y la variable quality no son iguales')


# como las varianzas son distintas se aplica una modificación del test de Student
# t, p = stats.ttest_ind(data.Control.dropna(), data.Treatment.dropna(), equal_var = False) # [7]
```

En consola obtenemos:

```
FlignerResult(statistic=370.45479435211206, pvalue=1.4899491299826965e-82)
1.4899491299826965e-82
La varianza de la variable fixed acidity y la variable quality no son iguales
FlignerResult(statistic=296.97394034581987, pvalue=1.5033455693734638e-66)
1.5033455693734638e-66
La varianza de la variable volatile acidity y la variable quality no son iguales
FlignerResult(statistic=301.0829403847896, pvalue=1.9135355234445877e-67)
1.9135355234445877e-67
La varianza de la variable citric acid y la variable quality no son iguales
FlignerResult(statistic=11.162133955066533, pvalue=0.0008348383976247281)
0.0008348383976247281
La varianza de la variable residual sugar y la variable quality no son iguales
FlignerResult(statistic=302.44982751188155, pvalue=9.639257409373958e-68)
9.639257409373958e-68
La varianza de la variable chlorides y la variable quality no son iguales
FlignerResult(statistic=1691.7688962516415, pvalue=0.0)
0.0
La varianza de la variable free sulfur dioxide y la variable quality no son iguales
FlignerResult(statistic=1862.4590457976583, pvalue=0.0)
0.0
La varianza de la variable total sulfur dioxide y la variable quality no son iguales
FlignerResult(statistic=291.6449381257884, pvalue=2.178425047547904e-65)
2.178425047547904e-65
La varianza de la variable density y la variable quality no son iguales
FlignerResult(statistic=301.31743212906724, pvalue=1.7011763779334925e-67)
1.7011763779334925e-67
La varianza de la variable pH y la variable quality no son iguales
FlignerResult(statistic=296.429289435054, pvalue=1.9757147217859962e-66)
1.9757147217859962e-66
La varianza de la variable sulphates y la variable quality no son iguales
FlignerResult(statistic=48.0868317747728, pvalue=4.077557388300544e-12)
4.077557388300544e-12
La varianza de la variable alcohol y la variable quality no son iguales
```

Hemos comprobado que la varianza de la calidad del vino comparada con las demás variables nunca son iguales.

Por lo tanto, para llevar a cabo un contraste de hipótesis, lo realizaremos con una modificación del test de Student [4], ya que hemos comprobado que:

- El tamaño de muestras es mayor que n > 30.
- Las variables se distribuyen normalmente.
- Las varianzas de las variables no son iguales.

4.3 Pruebas estadísticas.

4.3.1 Correlación.

El coeficiente de correlación de Pearson mide la relación lineal entre dos conjuntos de datos (X e Y) y requiere que cada uno de ellos esté distribuido normalmente. El valor que se obtiene estará entre un rango de [-1,1], donde 0 implica que no existe correlación.

Las correlaciones positivas significan que a medida que aumenta X, aumenta Y. Las correlaciones negativas, al revés, si aumenta X, disminuye Y.

Gracias a la librería ***scipy*** de Python:

```
# -------------------- ESTUDIO DE DEPENDENCIAS -------------------- #
# como las variables se han comprobado que siguen distribuciones normales[8]

for i in range(len(df.columns)-1):
    print('La correlación de la variable ' + str(df.columns[i]) + ' con la variable quality es')
    pearsonr_results = scipy.stats.pearsonr(df['quality'],df[df.columns[i]])
    print(pearsonr_results)
```

En consola:

```
La correlación de la variable fixed acidity con la variable quality es
(0.12405164911322429, 6.495635009281129e-07)
La correlación de la variable volatile acidity con la variable quality es
(-0.3905577802640072, 2.051714807013982e-59)
La correlación de la variable citric acid con la variable quality es
(0.22637251431804137, 4.9912952505096635e-20)
La correlación de la variable residual sugar con la variable quality es
(0.013731637340066275, 0.5832180131578858)
La correlación de la variable chlorides con la variable quality es
(-0.1289065599300527, 2.3133826540590474e-07)
La correlación de la variable free sulfur dioxide con la variable quality es
(-0.05065605724427634, 0.042833979508152646)
La correlación de la variable total sulfur dioxide con la variable quality es
(-0.18510028892653782, 8.621703423658214e-14)
La correlación de la variable density con la variable quality es
(-0.17491922778334873, 1.8749566520108486e-12)
La correlación de la variable pH con la variable quality es
(-0.057731391205382156, 0.02096277865147741)
La correlación de la variable sulphates con la variable quality es
(0.25139707906926134, 1.802088453452842e-24)
La correlación de la variable alcohol con la variable quality es
(0.47616632400113607, 2.8314769747769775e-91)
```

Vemos que la variable más relevante, con una correlación de 0.47 es el alcohol.

También podemos ver el otro valor, ***p-value***, que nos da información del peso estadístico de la correlación obtenida.

4.3.2. Contraste de hipótesis.

En este apartado tenemos dos muestras (de diferente tamaño); **vino bueno** y **vino no bueno**.

Como comentamos anteriormente, realizaremos un test paramétrico modificado de Student.

Se quiere realizar el siguiente contraste de hipótesis de dos muestras sobre la diferencia de medias, el cual es unilateral , y nos dice sobre la hipótesis alternativa que:

$$H_0 : \mu_1 - \mu_2 = 0$$
$$H_1 : \mu_1 - \mu_2 < 0$$

donde μ_1 es la media de la población de la que se extrae la primera muestra y μ_2 es la media de la población de la que se extrae la segunda. Además, el nivel de significación sigue siendo alpha=0.5.

```
# ---------------------CONTRASTE DE HIPÓTESIS [9] ------------------- #
t, p = scipy.stats.ttest_ind(vino_bueno['alcohol'], vino_no_bueno['alcohol'], equal_var = False)
print(t, p)
```

En consola:

```
17.450495229281344 7.7931668998382e-47
```

El segundo parámetro obtenido es el ***p-value*** [5]. El ser este más pequeño que alpha, rechazamos la hipótesis nula, por lo tanto se puede concluir que la media del alcohol del vino no bueno, es mayor que la del vino bueno.

4.3.3. Regresión.

En este último apartado, queremos obtener un modelo que sea capaz de predecir la calidad del vino en función algunas variables. Usaremos las variables que mejor correlación tenían calculadas anteriormente. Además, probaremos dos modelos para saber cuál ofrece mejores prestaciones:

```
# ------------- PREDICCIÓN CON REGRESIÓN MÚLTIPLE -------------- # [10]
df_lm = pd.DataFrame()
df_lm["alcohol"] = df["alcohol"]
df_lm["sulphates"] = df['sulphates']
df_lm['citric acid'] = df['citric acid']
df_lm['volatile acidity'] = df['volatile acidity']
df_lm['free sulfur dioxide'] = df['free sulfur dioxide']

# Se divide en training/testing sets
X_train = df_lm.iloc[:1280,]
X_test = df_lm.iloc[1280:,]


# La variable explicativa se divide en training/testing sets
y_train = df['quality'][:1280]
y_test = df['quality'][1280:]
print(len(y_train))
print(len(y_test))

# creamos el objeto para la regresión lineal
regr = linear_model.LinearRegression()

# Entrenamos el modelo
resultados_reg = regr.fit(X_train, y_train)

# Predecimos con el conjunto de test
y_pred = regr.predict(X_test)

# The coefficients
print('Coeficientes: \n', regr.coef_)
# The mean squared error [12]
print("Error cuadrático medio: %.2f"
      % mean_squared_error(y_test, y_pred))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % r2_score(y_test, y_pred))
```

En consola:

```
Coeficientes:
 [ 0.32102733  0.60178962 -0.12889901 -1.2534562  -0.00506523]
Error cuadrático medio: 0.44
Variance score: 0.26
```

Para el modelo 2:

```
# Modelo 2
df_lm2 = pd.DataFrame()
df_lm2["alcohol"] = df["alcohol"]
df_lm2["sulphates"] = df['sulphates']
df_lm2['citric acid'] = df['citric acid']
df_lm2['free sulfur dioxide'] = df['free sulfur dioxide']

# Se divide en training/testing sets
X_train2 = df_lm2.iloc[:1280,]
X_test2 = df_lm2.iloc[1280:,]


# La variable explicativa se divide en training/testing sets
y_train2 = df['quality'][:1280]
y_test2 = df['quality'][1280:]
print(len(y_train2))
print(len(y_test2))

# creamos el objeto para la regresión lineal
regr2 = linear_model.LinearRegression()

# Entrenamos el modelo
resultados_reg2 = regr2.fit(X_train2, y_train2)

# Predecimos con el conjunto de test
y_pred2 = regr2.predict(X_test2)

# The coefficients
print('Coeficientes: \n', regr2.coef_)
# The mean squared error [12]
print("Error cuadrático medio: %.2f"
      % mean_squared_error(y_test2, y_pred2))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % r2_score(y_test2, y_pred2))
```

En consola:

```
Coeficientes:
 [ 0.34804363  0.698464    0.45769991 -0.00401896]
Error cuadrático medio: 0.48
Variance score: 0.19
```

Podemos deducir que el modelo primero es mejor, puesto que el error cuadrático medio es menor; mientras más se acerque a 0, será mejor.

Ejercicio 5.

5.1. Representación de los resultados a partir de tablas y gráficas.

Se han añadido las gráficas correspondientes en cada apartado, para seguir un modelo de memoria más lógico.

Ejercicio 6.

6.1 Conclusiones.

Hemos comprobado con los pasos anteriores lo pasos a seguir en el pre-procesado de datos para su posterior análisis.

Hemos tenido un conjunto de datos original al cual no tuvimos que rellenar los datos vacíos porque eran inexistentes.

En cuanto a los valores extremos (*outliers*), no se eliminaron porque, se desconoce que hayan habido equipos mal calibrados para la toma de medidas, se sobreentiende que los datos pertenecen a la población de estudio, y que tenemos una muestra lo suficientemente grande para que disminuyan los posibles efectos adversos de éstos.

Se puede calcular la correlación de las variables sin tener en cuenta los *outliers*, y ver si éstas cambian. Si es el caso, se pueden estudiar los *outliers* a parte, para obtener un mejor modelo de los datos.

Además, se comprobó la normalidad y homogeneidad de las varianzas de las variables, para proceder a pruebas estadísticas como correlaciones, contraste de hipótesis y regresiones lineales para resolver el problema propuesto.

6.2 ¿Los resultados permiten resolver el problema?

La correlación nos ha ayudado a escoger las variables que mejor definen el modelo, y el contraste nos ha ayudado a saber que la media del alcohol de un vino bueno es menor que la de un vino no bueno.

Si vemos un fragmento del fichero de salida generado con los valores que se predijeron con los dos modelos, vemos como efectivamente, el primer valor se acerca más al valor real:

	A	B	C	D
1		y	ypred	ypred_2
2	1280	6	5,672317412	5,556831162
3	1281	6	5,672317412	5,556831162
4	1282	6	5,596617689	5,743611792
5	1283	6	5,434412105	5,561062347
6	1284	5	5,886767443	5,710588457
7	1285	5	5,776958067	5,712204697
8	1286	6	6,65501173	6,337128561
9	1287	5	6,311120025	6,318709329
10	1288	5	5,779208514	5,935330027
11	1289	5	5,779208514	5,935330027
12	1290	5	5,621091451	5,727611275
13	1291	6	5,707758689	5,773970907
14	1292	6	6,493836555	6,298098788
15	1293	4	5,124775122	5,205308154
16	1294	6	5,707758689	5,773970907
17	1295	5	4,965244752	4,926907677
18	1296	5	4,965244752	4,926907677
19	1297	6	6,185850174	6,152335898
20	1298	6	6,138428272	6,104214913
21	1299	3	4,426810714	5,564117711
22	1300	6	6,188159503	6,227475363
23	1301	6	5,705728129	6,051784973
24	1302	6	6,244334306	6,098399332
25	1303	5	6,13586142	5,959083326
26	1304	5	4,787253771	5,032195722
27	1305	5	5,238012708	5,314711777

Ejercicio 7.

7.1 Código.

El código generado para esta práctica se ha adjuntado en el proyecto Git.

8. Bibliografía.

Anotar que las referencias en el script y en este documento no coinciden.

[1] Fuente de datos de estudio [Última visita: 13/01/2019]
https://www.kaggle.com/uciml/red-wine-quality-cortez-et-al-2009

[2] Media winzorizada [Última visita: 9/01/2019]
https://www.cienciasinseso.com/tag/media-winsorizada/

[3] Best practices (Material UOC) [Última visita: 13/01/2019]

[4] http://pytolearn.csd.auth.gr/d1-hyptest/12/ttest-indep.html [Última visita: 11/01/2019]

[5] https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ttest_ind.html