2-visualizacion

January 25, 2021

#

Ciencia de Datos

Víctor Muñiz Sánchez

Maestría en Cómputo Estadístico

Enero a junio 2021

1 Visualización de datos multivariados

1.1 Descripción de los datos a usar

1.1.1 Tipos de vino y sus propiedades.

Usaremos éstos datos tomados de Cortez et al., 2009.

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import matplotlib as mpl
import numpy as np
import seaborn as sns
import os
os.chdir('/home/victor/cursos/ciencia_de_datos_2021/')

%matplotlib inline
```

[2]:
```
f = open('data/winequality.names.txt', 'r')
print(f.read())
```

```
Citation Request:
  This dataset is public available for research. The details are described in
[Cortez et al., 2009].
  Please include this citation if you plan to use this database:

  P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis.
  Modeling wine preferences by data mining from physicochemical properties.
  In Decision Support Systems, Elsevier, 47(4):547-553. ISSN: 0167-9236.
```

Available at: [@Elsevier] http://dx.doi.org/10.1016/j.dss.2009.05.016
[Pre-press (pdf)]
http://www3.dsi.uminho.pt/pcortez/winequality09.pdf
[bib] http://www3.dsi.uminho.pt/pcortez/dss09.bib

1. Title: Wine Quality

2. Sources
Created by: Paulo Cortez (Univ. Minho), Antonio Cerdeira, Fernando Almeida, Telmo Matos and Jose Reis (CVRVV) @ 2009

3. Past Usage:

P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis.
Modeling wine preferences by data mining from physicochemical properties.
In Decision Support Systems, Elsevier, 47(4):547-553. ISSN: 0167-9236.

In the above reference, two datasets were created, using red and white wine samples.
The inputs include objective tests (e.g. PH values) and the output is based on sensory data
(median of at least 3 evaluations made by wine experts). Each expert graded the wine quality
between 0 (very bad) and 10 (very excellent). Several data mining methods were applied to model
these datasets under a regression approach. The support vector machine model achieved the
best results. Several metrics were computed: MAD, confusion matrix for a fixed error tolerance (T),
etc. Also, we plot the relative importances of the input variables (as measured by a sensitivity
analysis procedure).

4. Relevant Information:

The two datasets are related to red and white variants of the Portuguese "Vinho Verde" wine.
For more details, consult: http://www.vinhoverde.pt/en/ or the reference [Cortez et al., 2009].
Due to privacy and logistic issues, only physicochemical (inputs) and sensory (the output) variables
are available (e.g. there is no data about grape types, wine brand, wine selling price, etc.).

These datasets can be viewed as classification or regression tasks.
The classes are ordered and not balanced (e.g. there are munch more normal wines than
excellent or poor ones). Outlier detection algorithms could be used to detect

```
the few excellent
   or poor wines. Also, we are not sure if all input variables are relevant. So
   it could be interesting to test feature selection methods.

5. Number of Instances: red wine - 1599; white wine - 4898.

6. Number of Attributes: 11 + output attribute

   Note: several of the attributes may be correlated, thus it makes sense to
apply some sort of
   feature selection.

7. Attribute information:

   For more information, read [Cortez et al., 2009].

   Input variables (based on physicochemical tests):
   1 - fixed acidity
   2 - volatile acidity
   3 - citric acid
   4 - residual sugar
   5 - chlorides
   6 - free sulfur dioxide
   7 - total sulfur dioxide
   8 - density
   9 - pH
   10 - sulphates
   11 - alcohol
   Output variable (based on sensory data):
   12 - quality (score between 0 and 10)

8. Missing Attribute Values: None
```

[3]:
```
white_wine = pd.read_csv('data/winequality-white.csv', sep=';')
red_wine = pd.read_csv('data/winequality-red.csv', sep=';')

# agregamos una nueva variable, el tipo de vino
red_wine['wine_type'] = 'red'
white_wine['wine_type'] = 'white'

# también se agrega la variable cualitativa 'quality', que se construye a␣
 ↪partir de
# los scores dados en 'quality'
red_wine['quality_label'] = red_wine['quality'].apply(lambda value: 'low'
                                                      if value <= 5 else␣
 ↪'medium'
```

```
                                                                    if value <= 7␣
↪else 'high')
red_wine['quality_label'] = pd.Categorical(red_wine['quality_label'],
                                           categories=['low', 'medium', 'high'])
white_wine['quality_label'] = white_wine['quality'].apply(lambda value: 'low'
                                                                    if value <= 5␣
↪else 'medium'
                                                                        if value <= 7␣
↪else 'high')
white_wine['quality_label'] = pd.Categorical(white_wine['quality_label'],
                                             categories=['low', 'medium',␣
↪'high'])

# merge red and white wine datasets
wines = pd.concat([red_wine, white_wine])

# re-shuffle records just to randomize data points
wines = wines.sample(frac=1, random_state=42).reset_index(drop=True)
```

[4]:
```
wines.head()
```

[4]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	wine_type	quality_label
0	7.0	0.17	0.74	12.8	0.045	24.0	126.0	0.99420	3.26	0.38	12.2	8	white	high
1	7.7	0.64	0.21	2.2	0.077	32.0	133.0	0.99560	3.27	0.45	9.9	5	red	low
2	6.8	0.39	0.34	7.4	0.020	38.0	133.0	0.99212	3.18	0.44	12.0	7	white	medium
3	6.3	0.28	0.47	11.2	0.040	61.0	183.0	0.99592	3.12	0.51	9.5	6	white	medium
4	7.4	0.35	0.20	13.9	0.054	63.0	229.0	0.99888	3.11	0.50	8.9	6	white	medium

[10]:
```
# si quieres guardar los datos...
wines.to_csv('data/all_wines.csv')
```

Variables

- **fixed acidity:** Acids are one of the fundamental properties of wine and contribute greatly

to the taste of the wine. Reducing acids significantly might lead to wines tasting flat. Fixed acids include tartaric, malic, citric, and succinic acids which are found in grapes (except succinic). This variable is usually expressed in $\frac{g(tartaricacid)}{dm^3}$ in the dataset.

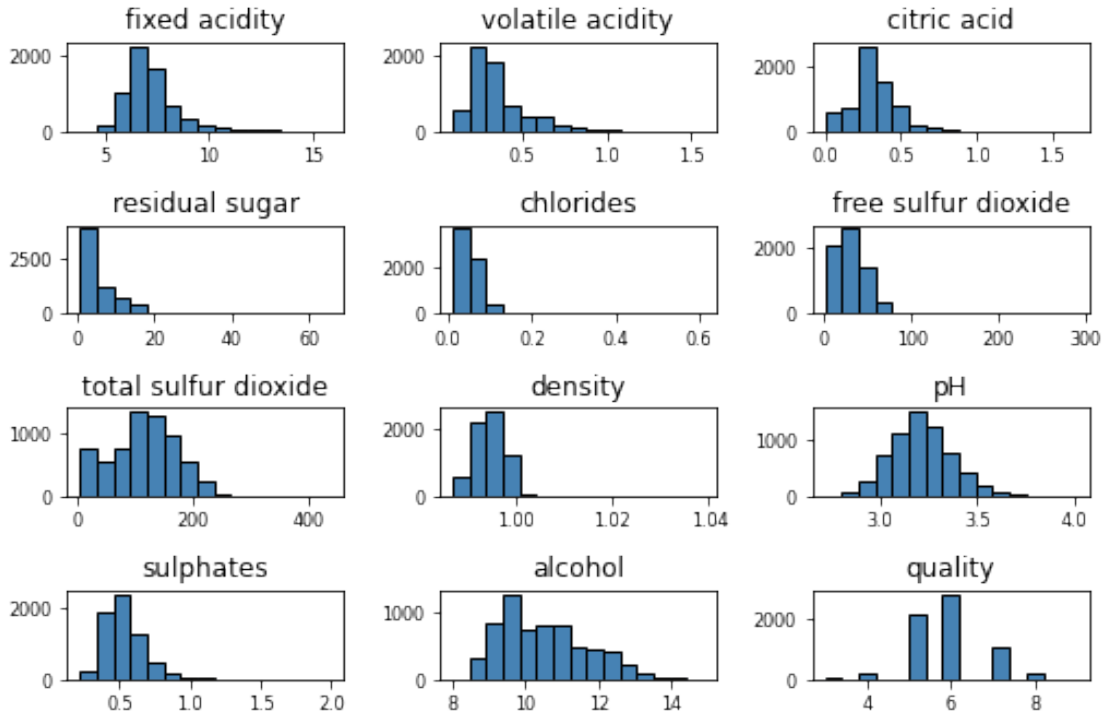
- **volatile acidity:** These acids are to be distilled out from the wine before completing the production process. It is primarily constituted of acetic acid though other acids like lactic, formic and butyric acids might also be present. Excess of volatile acids are undesirable and lead to unpleasant flavor. In the US, the legal limits of volatile acidity are 1.2 g/L for red table wine and 1.1 g/L for white table wine. The volatile acidity is expressed in $\frac{g(aceticacid)}{dm^3}$ in the dataset.
- **citric acid:** This is one of the fixed acids which gives a wine its freshness. Usually most of it is consumed during the fermentation process and sometimes it is added separately to give the wine more freshness. It's usually expressed in $\frac{g}{dm^3}$ in the dataset.
- **residual sugar:** This typically refers to the natural sugar from grapes which remains after the fermentation process stops, or is stopped. It's usually expressed in $\frac{g}{dm^3}$ in the dataset.
- **chlorides:** This is usually a major contributor to saltiness in wine. It's usually expressed in $\frac{g(sodiumchloride)}{dm^3}$ in the dataset.
- **free sulfur dioxide:** This is the part of the sulphur dioxide that when added to a wine is said to be free after the remaining part binds. Winemakers will always try to get the highest proportion of free sulphur to bind. They are also known as sulfites and too much of it is undesirable and gives a pungent odour. This variable is expressed in $\frac{mg}{dm^3}$ in the dataset.
- **total sulfur dioxide:** This is the sum total of the bound and the free sulfur dioxide (SO_2). Here, it's expressed in $\frac{mg}{dm^3}$. This is mainly added to kill harmful bacteria and preserve quality and freshness. There are usually legal limits for sulfur levels in wines and excess of it can even kill good yeast and give out undesirable odour.
- **density:** This can be represented as a comparison of the weight of a specific volume of wine to an equivalent volume of water. It is generally used as a measure of the conversion of sugar to alcohol. Here, it's expressed in $\frac{g}{cm^3}$.
- **pH:** Also known as the potential of hydrogen, this is a numeric scale to specify the acidity or basicity the wine. Fixed acidity contributes the most towards the pH of wines. You might know, solutions with a pH less than 7 are acidic, while solutions with a pH greater than 7 are basic. With a pH of 7, pure water is neutral. Most wines have a pH between 2.9 and 3.9 and are therefore acidic.
- **sulphates:** These are mineral salts containing sulfur. Sulphates are to wine as gluten is to food. They are a regular part of the winemaking around the world and are considered essential. They are connected to the fermentation process and affects the wine aroma and flavor. Here, it's expressed in $\frac{g(potassiumsulphate)}{dm^3}$ in the dataset.
- **alcohol:** Wine is an alcoholic beverage. Alcohol is formed as a result of yeast converting sugar during the fermentation process. The percentage of alcohol can vary from wine to wine. Hence it is not a surprise for this attribute to be a part of this dataset. It's usually measured in % vol or alcohol by volume (ABV).
- **quality:** Wine experts graded the wine quality between 0 (very bad) and 10 (very excellent). The eventual quality score is the median of at least three evaluations made by the same wine

experts.

- **wine_type:** Since we originally had two datasets for red and white wine, we introduced this attribute in the final merged dataset which indicates the type of wine for each data point. A wine can either be a 'red' or a 'white' wine. One of the predictive models we will build in this chapter would be such that we can predict the type of wine by looking at other wine attributes.
- **quality_label:** This is a derived attribute from the `quality` attribute. We bucket or group wine quality scores into three qualitative buckets namely low, medium and high. Wines with a quality score of 3, 4 & 5 are low quality, scores of 6 & 7 are medium quality and scores of 8 & 9 are high quality wines. We will also build another model in this chapter to predict this wine quality label based on other wine attributes.

1.2 Visualizaciones de variables en una dimensión

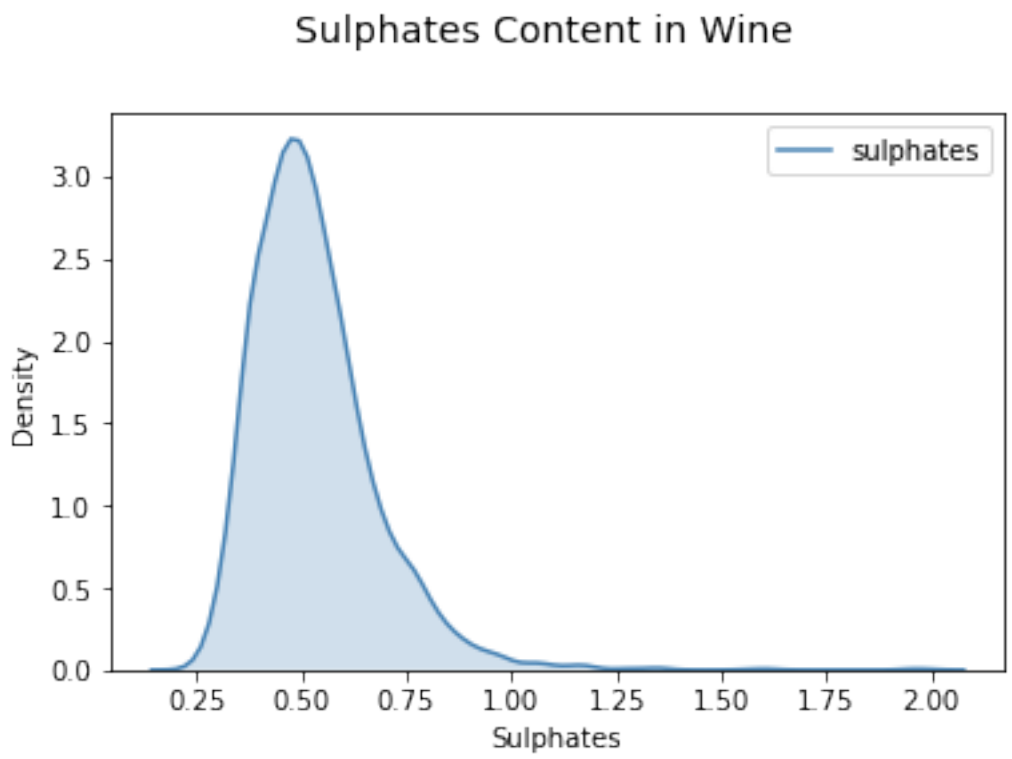
[12]:
```
wines.hist(bins=15, color='steelblue', edgecolor='black', linewidth=1.0,
           xlabelsize=8, ylabelsize=8, grid=False)
plt.tight_layout(rect=(0, 0, 1.2, 1.2))
```



1.2.1 Variables contínuas y categóricas

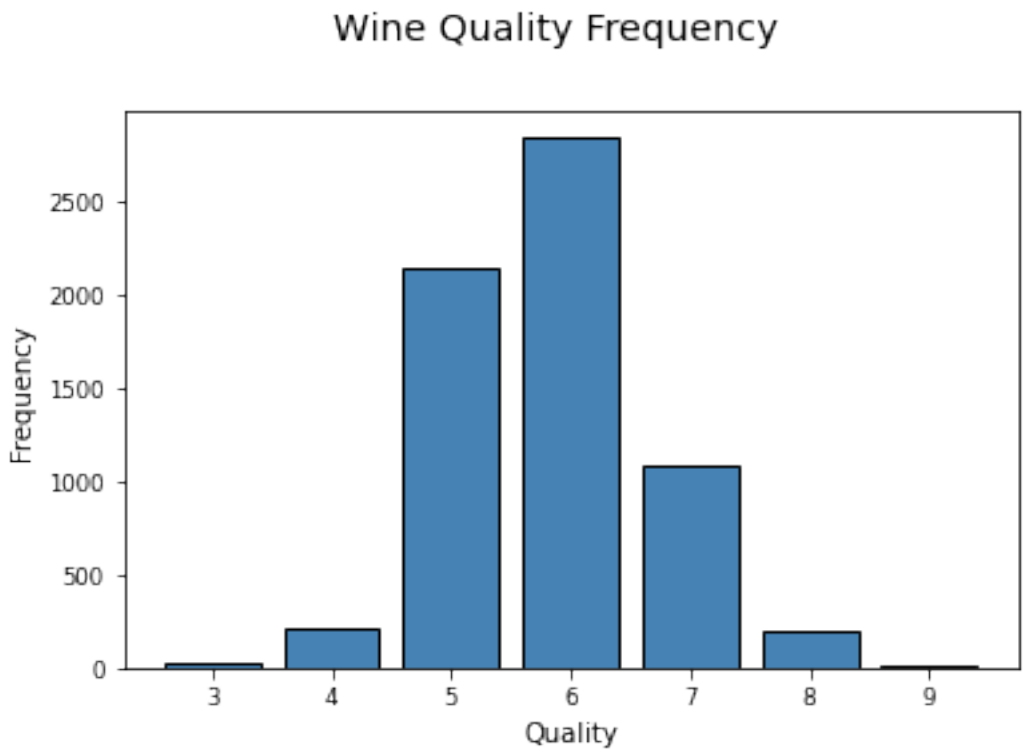
[13]:
```
fig = plt.figure(figsize = (6, 4))
title = fig.suptitle("Sulphates Content in Wine", fontsize=14)
fig.subplots_adjust(top=0.85, wspace=0.3)

ax1 = fig.add_subplot(1,1, 1)
ax1.set_xlabel("Sulphates")
ax1.set_ylabel("Density")
sns.kdeplot(wines['sulphates'], ax=ax1, shade=True, color='steelblue')
```

[13]:
```
<AxesSubplot:xlabel='Sulphates', ylabel='Density'>
```



[14]:
```
fig = plt.figure(figsize = (6, 4))
title = fig.suptitle("Wine Quality Frequency", fontsize=14)
fig.subplots_adjust(top=0.85, wspace=0.3)

ax = fig.add_subplot(1,1, 1)
ax.set_xlabel("Quality")
ax.set_ylabel("Frequency")
w_q = wines['quality'].value_counts()
w_q = (list(w_q.index), list(w_q.values))
ax.tick_params(axis='both', which='major', labelsize=8.5)
```

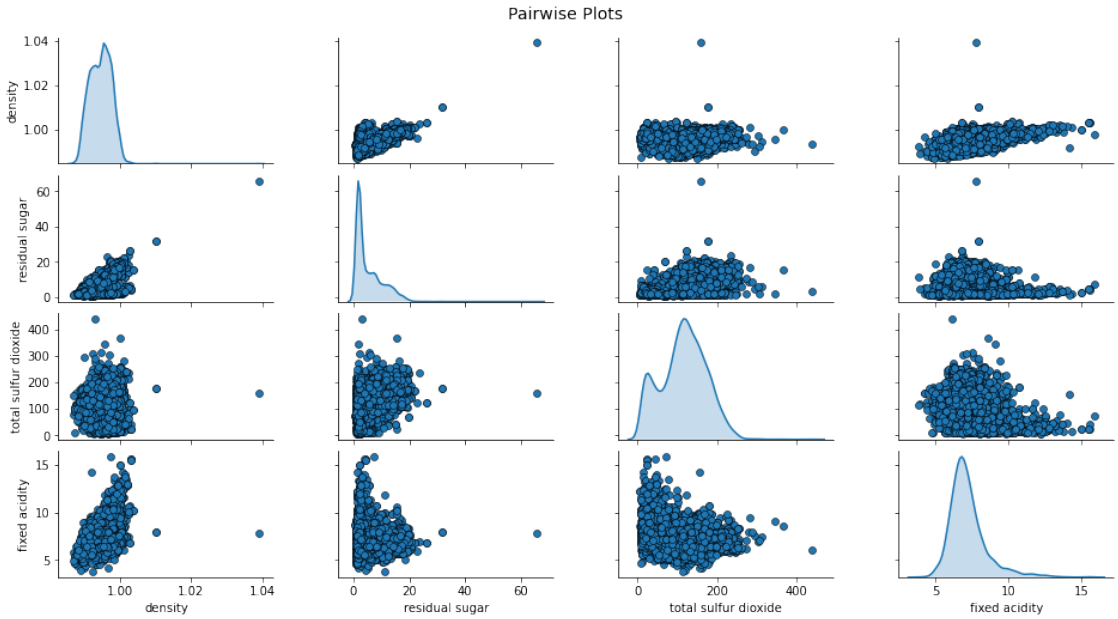
```
bar = ax.bar(w_q[0], w_q[1], color='steelblue',
        edgecolor='black', linewidth=1)
```



1.3 Visualizaciones de datos multivariados

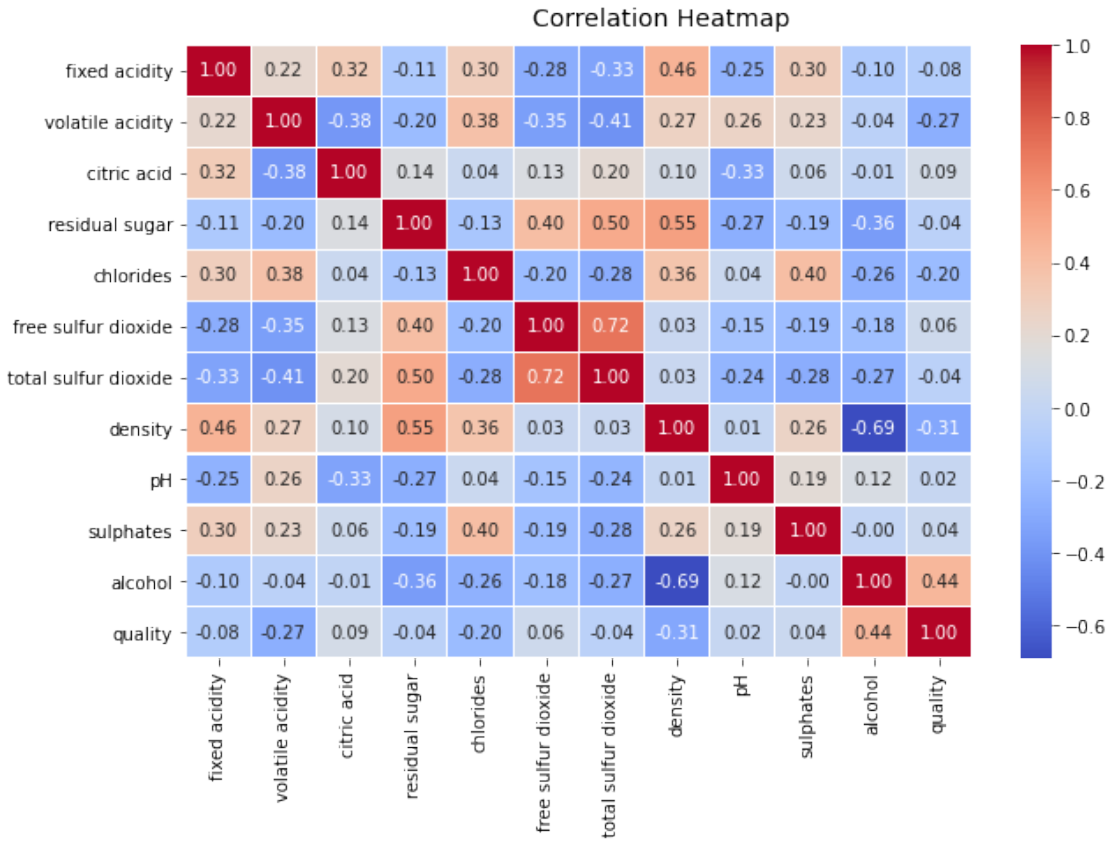
1.3.1 Gráficos de dispersión (scatterplots)

[15]:
```
cols = ['density', 'residual sugar', 'total sulfur dioxide', 'fixed acidity']
pp = sns.pairplot(wines[cols], height=1.8, aspect=1.8,
                  plot_kws=dict(edgecolor="k", linewidth=0.5),
                  diag_kind="kde", diag_kws=dict(shade=True))

fig = pp.fig
fig.subplots_adjust(top=0.93, wspace=0.3)
t = fig.suptitle('Pairwise Plots', fontsize=14)
```



1.3.2 Correlation plot

[16]:
```
f, ax = plt.subplots(figsize=(10, 6))
corr = wines.corr()
hm = sns.heatmap(round(corr,2), annot=True, ax=ax, cmap="coolwarm",fmt='.2f',
            linewidths=.05)
f.subplots_adjust(top=0.93)
t= f.suptitle('Correlation Heatmap', fontsize=14)
```

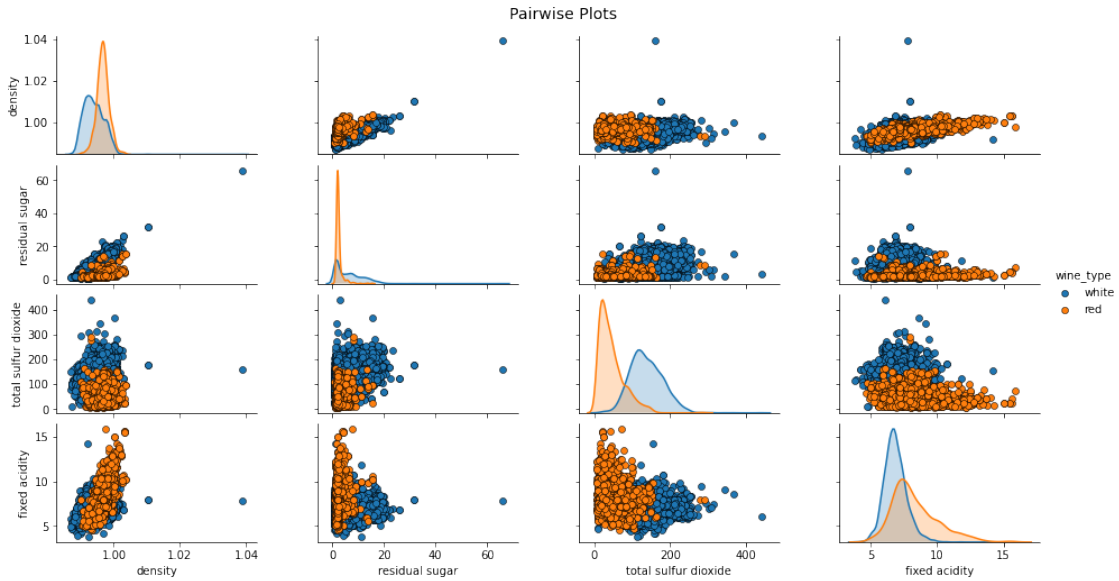


¿Estandarizar o no estandarizar?

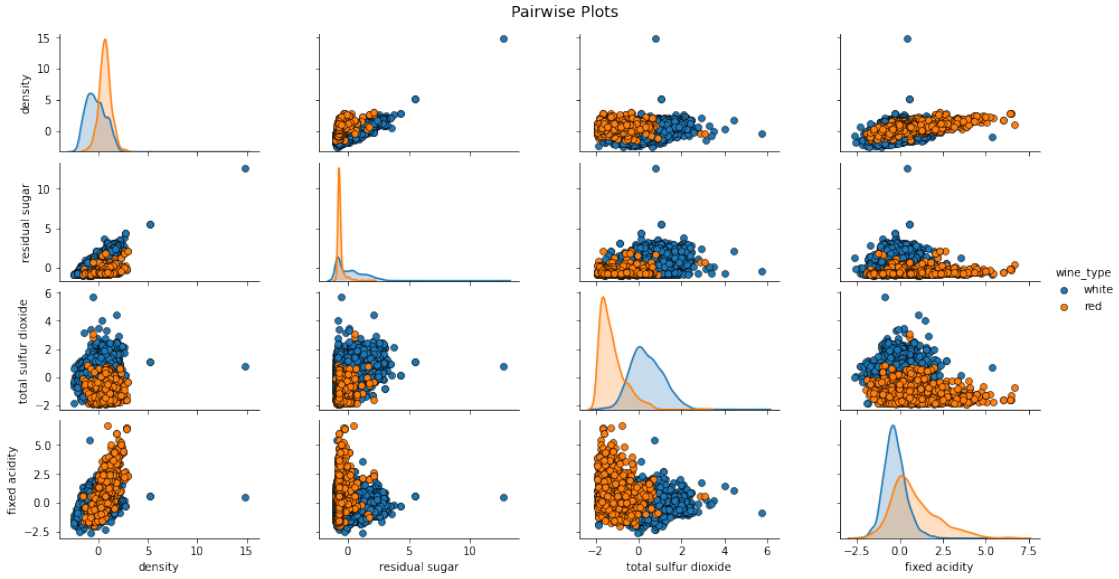
[24]:
```
cols = ['density', 'residual sugar', 'total sulfur dioxide', 'fixed acidity']
subset_df = wines[cols]

from sklearn.preprocessing import StandardScaler

ss = StandardScaler()
scaled_df = ss.fit_transform(subset_df)
scaled_df = pd.DataFrame(scaled_df, columns=cols)
final_df = pd.concat([scaled_df, wines['wine_type']], axis=1)

temp_df = pd.DataFrame(subset_df, columns=cols)
orig_df = pd.concat([temp_df, wines['wine_type']], axis=1)
```

[22]:
```
pp2 = sns.pairplot(orig_df, height=1.8, aspect=1.8, hue='wine_type',
                   plot_kws=dict(edgecolor="k", linewidth=0.5),
                   diag_kind="kde", diag_kws=dict(shade=True))

fig = pp2.fig
fig.subplots_adjust(top=0.93, wspace=0.3)
```

```
t = fig.suptitle('Pairwise Plots', fontsize=14)
```



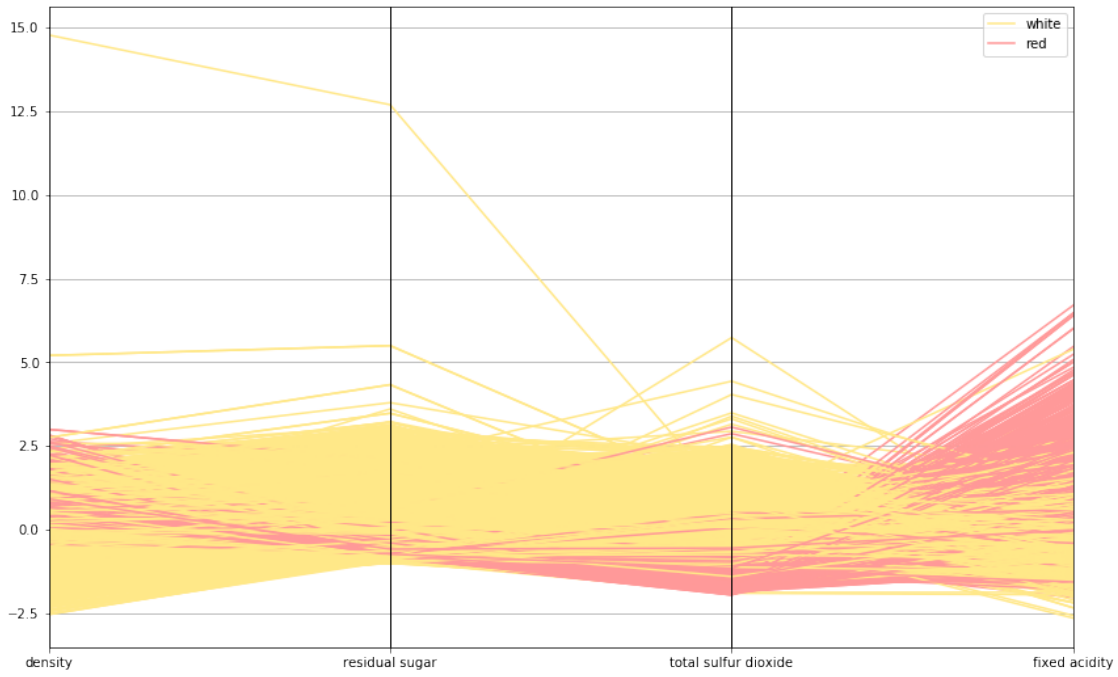
[23]:
```
pp2 = sns.pairplot(final_df, height=1.8, aspect=1.8, hue='wine_type',
                   plot_kws=dict(edgecolor="k", linewidth=0.5),
                   diag_kind="kde", diag_kws=dict(shade=True))

fig = pp2.fig
fig.subplots_adjust(top=0.93, wspace=0.3)
t = fig.suptitle('Pairwise Plots', fontsize=14)
```



1.3.3 Parallel coordinate plots (Inselberg 1985, Wegman 1990).

Usadas para examinar variables múltiples, correlaciones, agrupamientos en alta dimensión y otras relaciones. Se construye tomando los ejes paralelos en vez de la orientación ortogonal tradicional del sistema cartesiano. Cada observación está representada por una linea

[37]:
```
from pandas.plotting import parallel_coordinates

plt.figure(figsize = (14,9))
pc = parallel_coordinates(frame = final_df, class_column = 'wine_type',␣
 ↪color=('#FFE888', '#FF9999'))
```

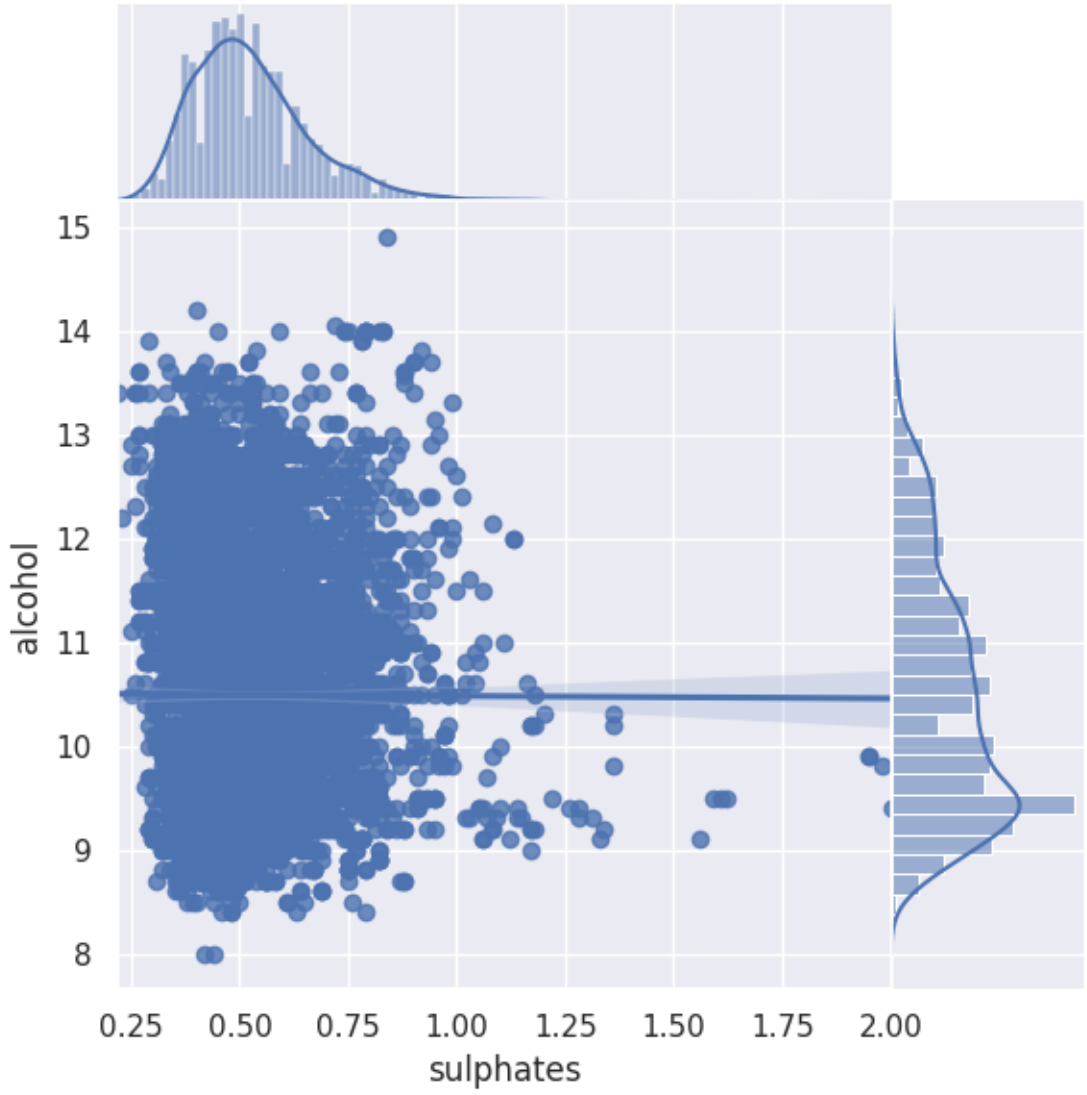


1.3.4 Scatterplots y distribuciones marginales

[41]:
```
plt.figure(figsize = (6,6))
plt.scatter(wines['sulphates'], wines['alcohol'],
            alpha=0.4, edgecolors='w')

plt.xlabel('Sulphates')
plt.ylabel('Alcohol')
plt.title('Wine Sulphates - Alcohol Content',y=1.05)
```

[41]:
```
Text(0.5, 1.05, 'Wine Sulphates - Alcohol Content')
```

[43]:
```
jp = sns.jointplot(x='sulphates', y='alcohol', data=wines, kind='reg', space=0,␣
↪height=6, ratio=4)
```

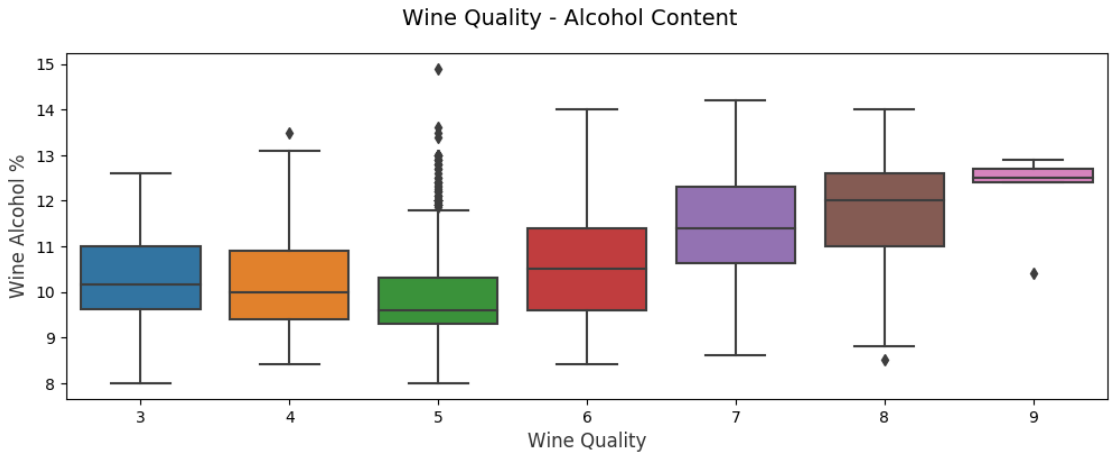


1.3.5 Visualización de variables numéricas y categóricas.

Box Plot

[57]:
```
f, (ax) = plt.subplots(1, 1, figsize=(12, 4))
f.suptitle('Wine Quality - Alcohol Content', fontsize=14)

sns.boxplot(x="quality", y="alcohol", data=wines,  ax=ax)
ax.set_xlabel("Wine Quality",size = 12,alpha=0.8)
ax.set_ylabel("Wine Alcohol %",size = 12,alpha=0.8)
```

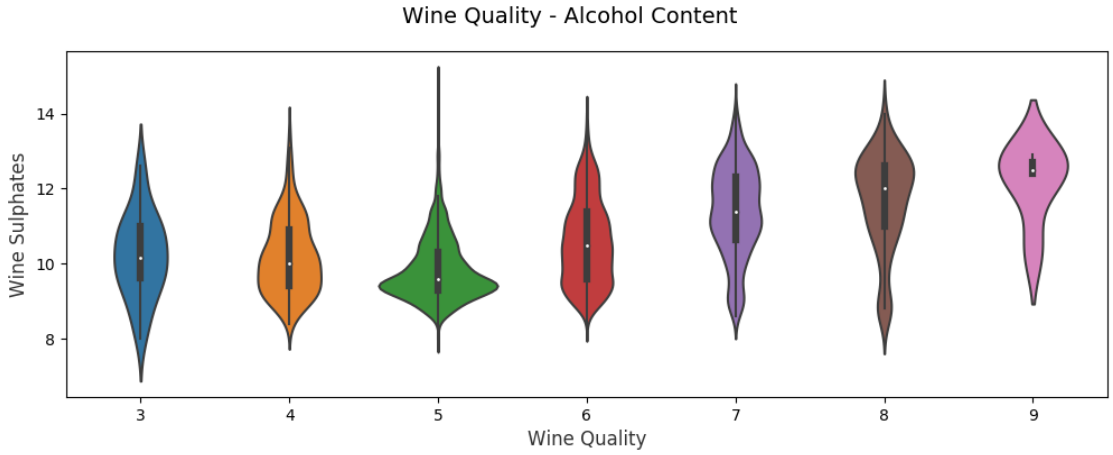
[57]:
```
Text(0, 0.5, 'Wine Alcohol %')
```



Violin Plot Parecido al boxplot, pero muestra la distribución estimada de la variable analizada

[61]:
```
f, (ax) = plt.subplots(1, 1, figsize=(12, 4))
f.suptitle('Wine Quality - Alcohol Content', fontsize=14)

sns.violinplot(x="quality", y="alcohol", data=wines,  ax=ax)
ax.set_xlabel("Wine Quality",size = 12,alpha=0.8)
ax.set_ylabel("Wine Sulphates",size = 12,alpha=0.8)
```

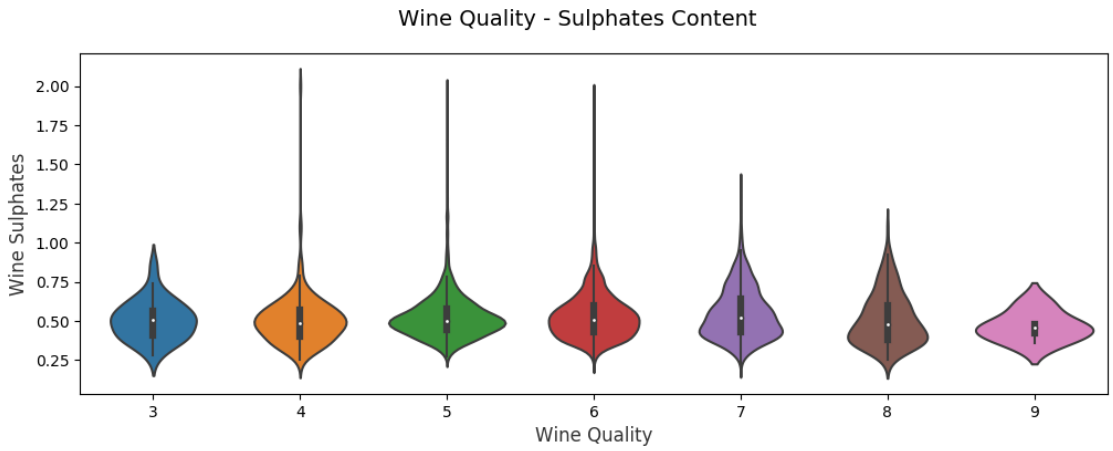
[61]:
```
Text(0, 0.5, 'Wine Sulphates')
```



[60]:
```
f, (ax) = plt.subplots(1, 1, figsize=(12, 4))
f.suptitle('Wine Quality - Sulphates Content', fontsize=14)

sns.violinplot(x="quality", y="sulphates", data=wines,  ax=ax)
```

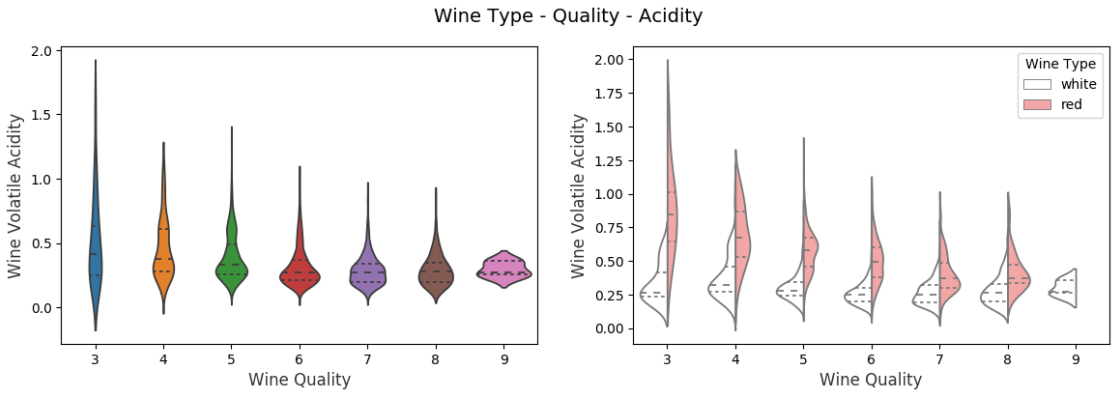
```
ax.set_xlabel("Wine Quality",size = 12,alpha=0.8)
ax.set_ylabel("Wine Sulphates",size = 12,alpha=0.8)
```

[60]: Text(0, 0.5, 'Wine Sulphates')



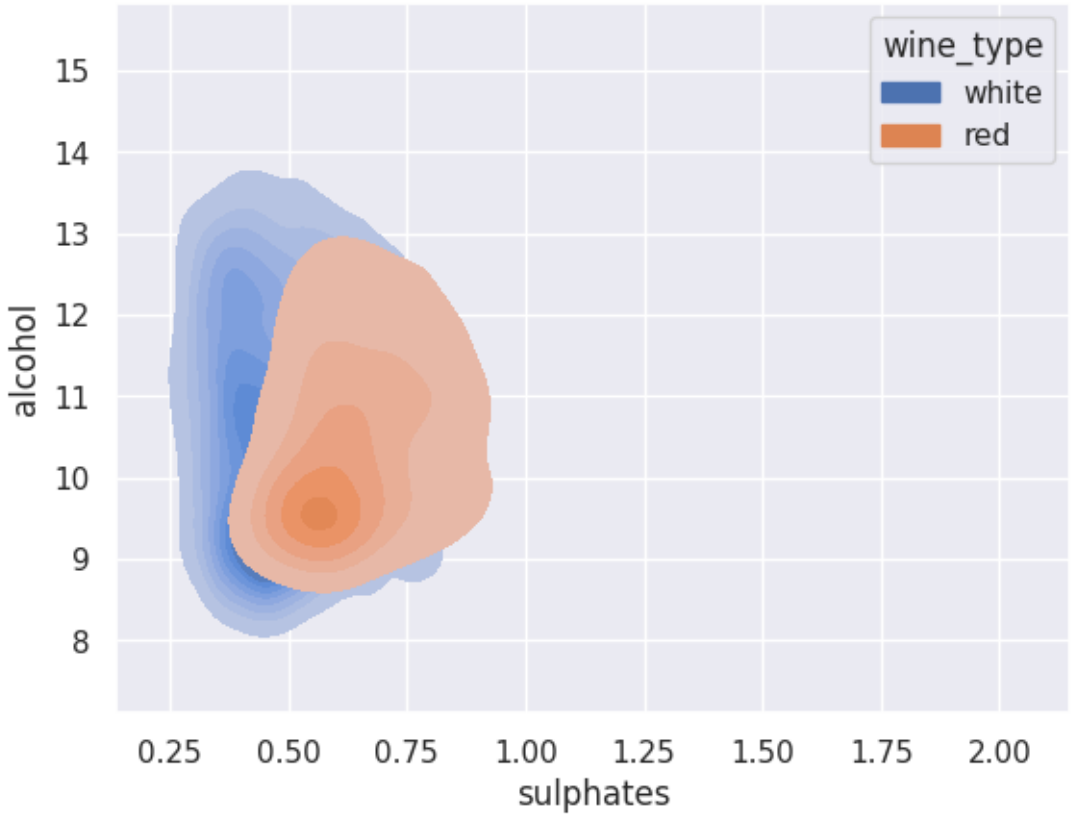
Podemos mostrar 2 variables categóricas

[62]:
```
f, (ax1, ax2) = plt.subplots(1, 2, figsize=(14, 4))
f.suptitle('Wine Type - Quality - Acidity', fontsize=14)

sns.violinplot(x="quality", y="volatile acidity",
               data=wines, inner="quart", linewidth=1.3,ax=ax1)
ax1.set_xlabel("Wine Quality",size = 12,alpha=0.8)
ax1.set_ylabel("Wine Volatile Acidity",size = 12,alpha=0.8)

sns.violinplot(x="quality", y="volatile acidity", hue="wine_type",
               data=wines, split=True, inner="quart", linewidth=1.3,
               palette={"red": "#FF9999", "white": "white"}, ax=ax2)
ax2.set_xlabel("Wine Quality",size = 12,alpha=0.8)
ax2.set_ylabel("Wine Volatile Acidity",size = 12,alpha=0.8)
l = plt.legend(loc='upper right', title='Wine Type')
```



Estimación de densidad. También, podemos mostrar estimaciones de la densidad en 2 dimensiones e incluir una variable categórica a través del color

[21]:
```
sns.reset_defaults()
sns.set_theme(style="darkgrid")
#plt.figure(figsize = (8,8))
ax = sns.kdeplot(x='sulphates', y='alcohol', data=wines, hue='wine_type',␣
↪fill=True)
```



OJO: fijarse que se tiene la última versión de Searborn para la visualización anterior (al menos la versión 0.10, da varios errores...)

[5]:
```
sns.__version__
```

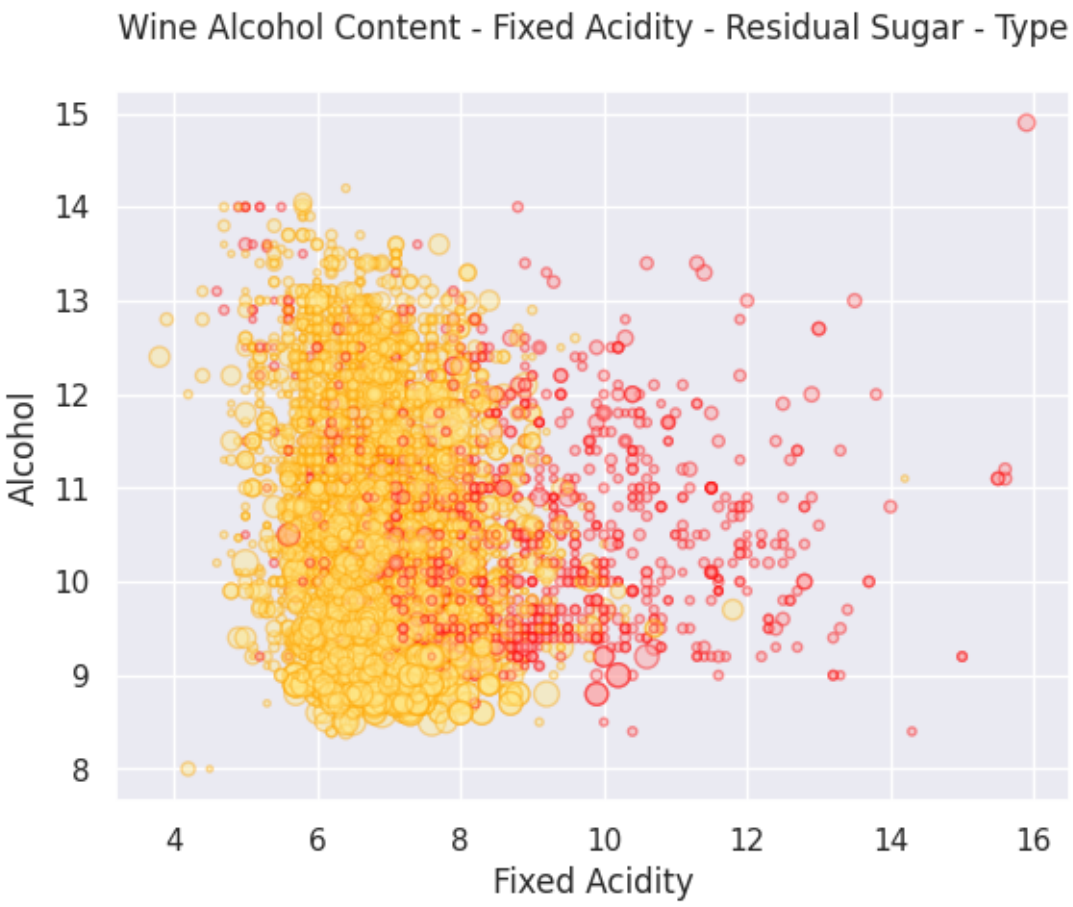
[5]:
```
'0.11.1'
```

Podemos generalizarlo para más de 2 variables mediante un Bubble Plot.

Por ejemplo, la variable ***residual sugar*** es proporcional al radio del círulo graficado en cada punto del scatter plot

[22]:
```
size = wines['residual sugar']*5
fill_colors = ['#FF9999' if wt=='red' else '#FFE888' for wt in␣
 ↪list(wines['wine_type'])]
edge_colors = ['red' if wt=='red' else 'orange' for wt in␣
 ↪list(wines['wine_type'])]

plt.scatter(wines['fixed acidity'], wines['alcohol'], s=size,
            alpha=0.4, color=fill_colors, edgecolors=edge_colors)

plt.xlabel('Fixed Acidity')
plt.ylabel('Alcohol')
plt.title('Wine Alcohol Content - Fixed Acidity - Residual Sugar - Type',y=1.05)
```

[22]:
```
Text(0.5, 1.05, 'Wine Alcohol Content - Fixed Acidity - Residual Sugar - Type')
```



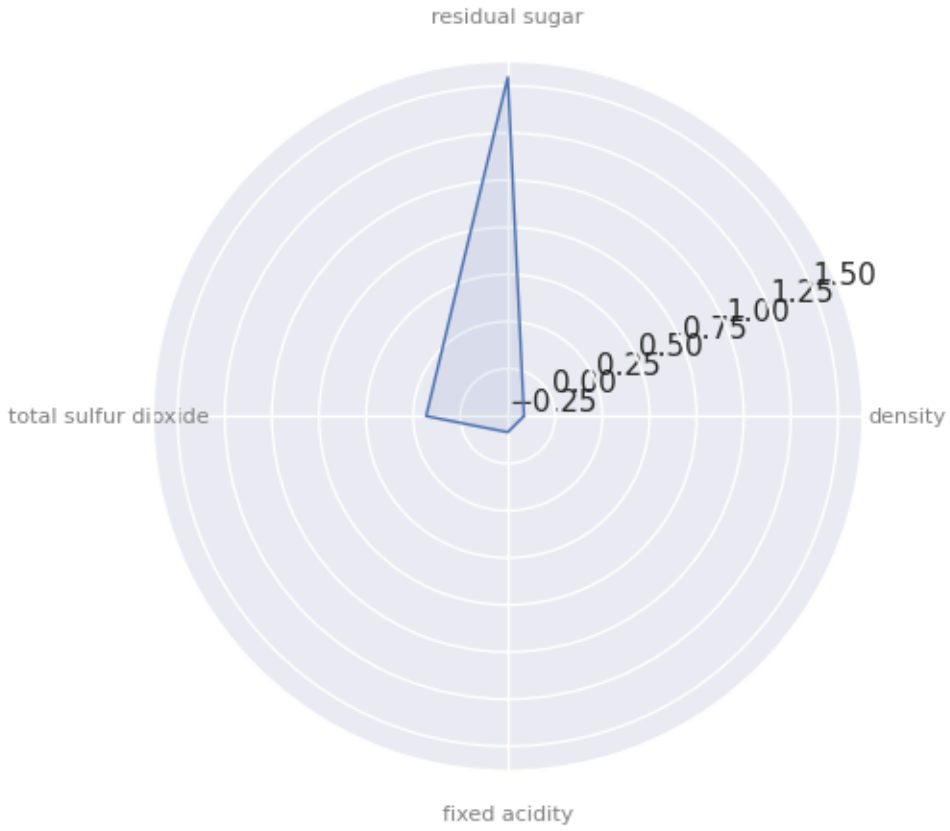
Radar plots, spider plots, star plots...

[44]:
```
from math import pi

scaled_df
variables = list(scaled_df)
N = len(variables)

# mostraremos los gráficos para 4 observaciones en particular, puedes cambiarlo␣
↪mediante el índice 'i'
i = 0
values = scaled_df.loc[i].values.flatten().tolist()
values += values[:1]
```

[45]:
```
# What will be the angle of each axis in the plot? (we divide the plot / number␣
↪of variable)
angles = [n / float(N) * 2 * pi for n in range(N)]
angles += angles[:1]
```

```
# Initialise the spider plot
ax = plt.subplot(111, polar=True)
# Draw one axe per variable + add labels labels yet
plt.xticks(angles[:-1], variables, color='grey', size=8)
# Plot data
ax.plot(angles, values, linewidth=1, linestyle='solid')
# Fill area
ax.fill(angles, values, 'b', alpha=0.1)
```

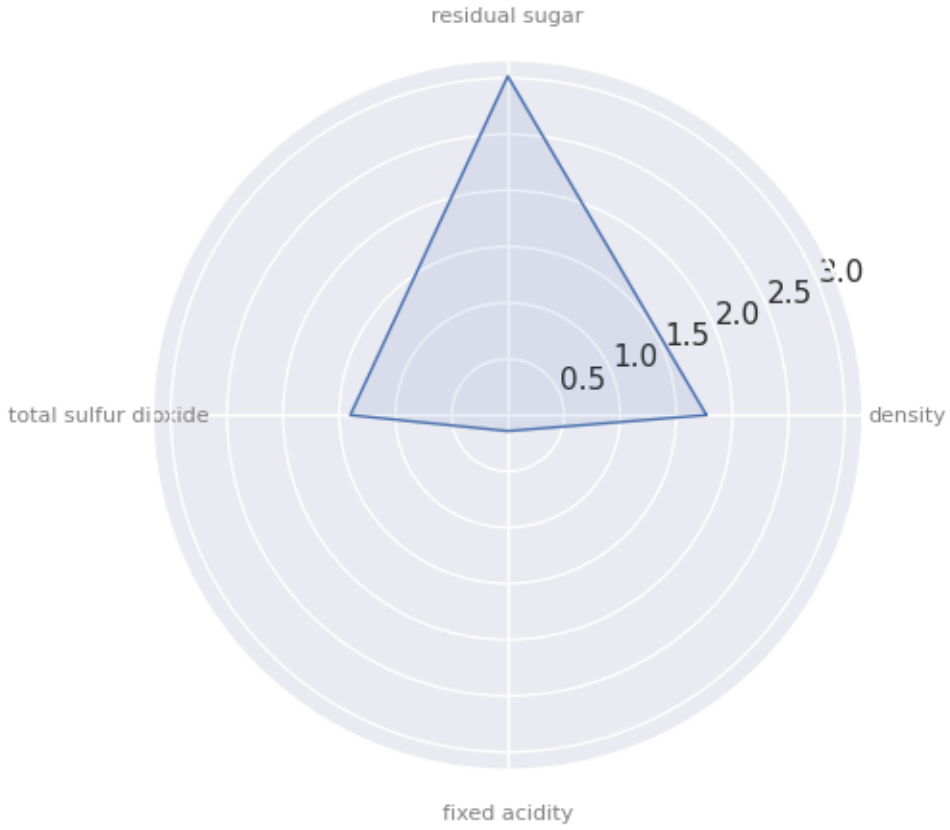
[45]: [<matplotlib.patches.Polygon at 0x7fad3b4a9850>]



[46]:
```
i = 10
values = scaled_df.loc[i].values.flatten().tolist()
values += values[:1]

# What will be the angle of each axis in the plot? (we divide the plot / number␣
↪of variable)
angles = [n / float(N) * 2 * pi for n in range(N)]
angles += angles[:1]
# Initialise the spider plot
```

```
ax = plt.subplot(111, polar=True)
# Draw one axe per variable + add labels labels yet
plt.xticks(angles[:-1], variables, color='grey', size=8)
# Plot data
ax.plot(angles, values, linewidth=1, linestyle='solid')
# Fill area
ax.fill(angles, values, 'b', alpha=0.1)
```

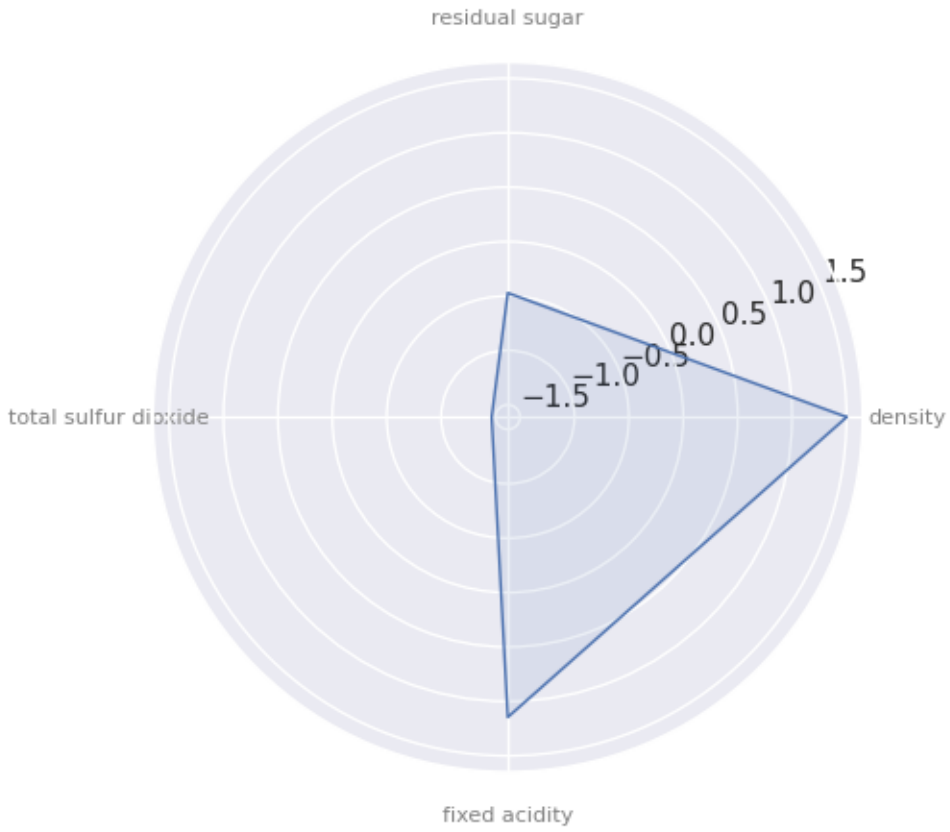
[46]: [<matplotlib.patches.Polygon at 0x7fad3b472550>]



[27]:
```
i = 50
values = scaled_df.loc[i].values.flatten().tolist()
values += values[:1]

# What will be the angle of each axis in the plot? (we divide the plot / number␣
 ↪of variable)
angles = [n / float(N) * 2 * pi for n in range(N)]
angles += angles[:1]
# Initialise the spider plot
ax = plt.subplot(111, polar=True)
```

```
# Draw one axe per variable + add labels labels yet
plt.xticks(angles[:-1], variables, color='grey', size=8)
# Plot data
ax.plot(angles, values, linewidth=1, linestyle='solid')
# Fill area
ax.fill(angles, values, 'b', alpha=0.1)
```

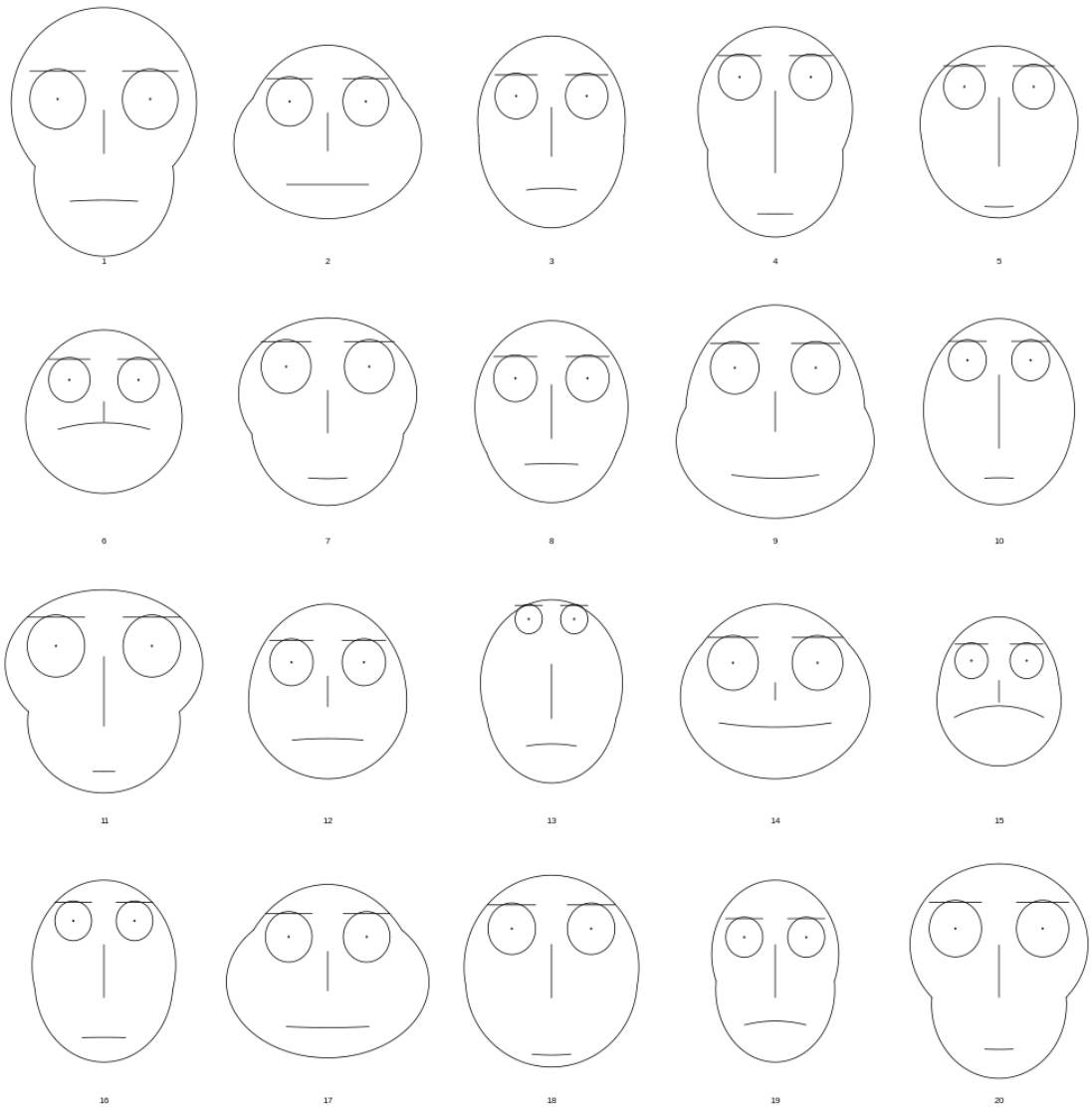
[27]: [<matplotlib.patches.Polygon at 0x7fad3b8e6520>]



Chernoff's faces: Muestra todas las variables como características de rostros. Cada variable corresponde a la forma y tamaño de alguna característica de la cara.

[39]:
```
from IPython.display import Image
Image(filename='notebooks/figs/chernoff1.png',width=600)
```

[39]:



effect of variables:

modified item Var
``height of face'' ``fixed.acidity''
``width of face'' ``volatile.acidity''
``structure of face'' ``citric.acid''
``height of mouth'' ``residual.sugar''
``width of mouth'' ``chlorides'' ``smiling'' ``free.sulfur.dioxide'' ``height of eyes'' ``total.sulfur.dioxide'' ``width of eyes'' ``density''
``height of hair'' ``pH''
``width of hair'' ``sulphates''
``style of hair'' ``fixed.acidity''
``height of nose'' ``volatile.acidity''
``width of nose'' ``citric.acid''

``width of ear'' ``residual.sugar''
``height of ear'' ``chlorides''

``People grow up studying and reacting to faces all of the time. Small and barely measurable differences are easily detected and evoke emotional reactions from a long catalogue buried in the memory… This implies that the human mind subconsciously operates as a high-speed computer, filtering out insignificant visual phenomena and focusing on the potentially important.''

Herman Chernoff (1973). ``*The Use of Faces to Represent Points in K-Dimensional Space Graphically*''. Journal of the American Statistical Association. American Statistical Association. 68 (342): 361--368.