Import necessary dependencies

In [2]:
```
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

%matplotlib inline
```

Load and merge datasets

TODO

- Read about the dataset structure (winequality.names)
- Load the winequality-white.csv and winequality-red.csv into Pandas DataFrames
- Add `wine_type` as an attribute
- Use `apply` method (and then pd.Categorical) to encode `quality_label` attribute (feature) with categorical values -> `low`, `medium`, `high`
  - Assume:
    'low' : $quality \in [0, 5)$,
    'medium' : $quality \in [5, 7)$,
    'high' : $quality \in [7, 10]$,
- merge red and white wine datasets and assign it to `wines` variable
- re-shuffle records to randomize data points

In [3]:
```
dataset_red = pd.read_csv('./winequality-red.csv', sep=';')
dataset_white = pd.read_csv('./winequality-white.csv', sep=';')

dataset_red['wine_type'] = 'red'
dataset_white['wine_type'] = 'white'

def give_label(number: int) -> str:
    if number < 5:
        return 'low'
    elif number < 7 and number >= 5:
        return 'medium'
    else:
        return 'high'

dataset_red['quality_label'] = dataset_red.apply(lambda row: give_label(row['qualit
dataset_white['quality_label'] = dataset_white.apply(lambda row: give_label(row['qu

dataset_red['quality_label'] = pd.Categorical(dataset_red.quality_label)
dataset_white['quality_label'] = pd.Categorical(dataset_white.quality_label)

wines = pd.concat([dataset_red, dataset_white])
wines = wines.sample(frac=1).reset_index(drop=True)  #dodanie reset_index
```

Understand dataset features and values

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

Out[4]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
0	6.1	0.34	0.27	2.6	0.024	20.0	105.0	0.99060	3.40	0.67	12.2
1	7.3	0.43	0.24	2.5	0.078	27.0	67.0	0.99648	3.60	0.59	11.1
2	6.9	0.23	0.35	6.9	0.030	45.0	116.0	0.99244	2.80	0.54	11.0
3	7.2	0.34	0.20	5.8	0.062	52.0	203.0	0.99461	3.17	0.44	9.8
4	7.5	0.19	0.40	7.1	0.056	50.0	110.0	0.99540	3.06	0.52	9.9

Understanding Wine and Types

Wine is an alcoholic beverage made from grapes which is fermented without the addition of sugars, acids, enzymes, water, or other nutrients

Red wine is made from dark red and black grapes. The color usually ranges from various shades of red, brown and violet. This is produced with whole grapes including the skin which adds to the color and flavor of red wines, giving it a rich flavor.

White wine is made from white grapes with no skins or seeds. The color is usually straw-yellow, yellow-green, or yellow-gold. Most white wines have a light and fruity flavor as compared to richer red wines.

Understanding Wine Attributes and Properties

- **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.

Exploratory Data Analysis and Visualizations

Descriptive Statistics

Let's do a quick basic descriptive summary statistics on some of these attributes of interest.

In [7]:
```
subset_attributes = ['residual sugar', 'total sulfur dioxide', 'sulphates', 'alcoho
rs = round(dataset_red[subset_attributes].describe(),2)
ws = round(dataset_white[subset_attributes].describe(),2)
pd.concat([rs, ws], axis=1, keys=['Red Wine Statistics', 'White Wine Statistics'])
```

Out[7]:

	Red Wine Statistics						W			
	residual sugar	total sulfur dioxide	sulphates	alcohol	volatile acidity	quality	residual sugar	total sulfur dioxide	sulphates	alcoho
count	1599.00	1599.00	1599.00	1599.00	1599.00	1599.00	4898.00	4898.00	4898.00	4898.0
mean	2.54	46.47	0.66	10.42	0.53	5.64	6.39	138.36	0.49	10.5
std	1.41	32.90	0.17	1.07	0.18	0.81	5.07	42.50	0.11	1.2
min	0.90	6.00	0.33	8.40	0.12	3.00	0.60	9.00	0.22	8.0
25%	1.90	22.00	0.55	9.50	0.39	5.00	1.70	108.00	0.41	9.5
50%	2.20	38.00	0.62	10.20	0.52	6.00	5.20	134.00	0.47	10.4
75%	2.60	62.00	0.73	11.10	0.64	6.00	9.90	167.00	0.55	11.4
max	15.50	289.00	2.00	14.90	1.58	8.00	65.80	440.00	1.08	14.2

In [8]:
```
subset_attributes = ['alcohol', 'volatile acidity', 'pH', 'quality']
ls = round(wines[wines['quality_label'] == 'low'][subset_attributes].describe(),2)
ms = round(wines[wines['quality_label'] == 'medium'][subset_attributes].describe(),
hs = round(wines[wines['quality_label'] == 'high'][subset_attributes].describe(),2)
pd.concat([ls, ms, hs], axis=1, keys=['Low Quality Wine', 'Medium Quality Wine', 'H
```

Out[8]:

	Low Quality Wine				Medium Quality Wine				High		
	alcohol	volatile acidity	pH	quality	alcohol	volatile acidity	pH	quality	alcohol	volatile acidity	
count	246.00	246.00	246.00	246.00	4974.00	4974.00	4974.00	4974.00	1277.00	1277.00	127
mean	10.18	0.47	3.23	3.88	10.27	0.35	3.22	5.57	11.43	0.29	
std	1.00	0.25	0.19	0.33	1.07	0.17	0.16	0.50	1.22	0.12	
min	8.00	0.11	2.74	3.00	8.00	0.08	2.72	5.00	8.50	0.08	
25%	9.40	0.28	3.09	4.00	9.40	0.23	3.11	5.00	10.70	0.20	
50%	10.05	0.38	3.22	4.00	10.00	0.30	3.20	6.00	11.50	0.27	
75%	10.90	0.61	3.36	4.00	11.00	0.42	3.32	6.00	12.40	0.34	
max	13.50	1.58	3.90	4.00	14.90	1.33	4.01	6.00	14.20	0.92	

Univariate Analysis

Univariate analysis is basically the simplest form of data analysis or visualization where we are only concerned with analyzing one data attribute or variable and visualizing the same (one dimension).

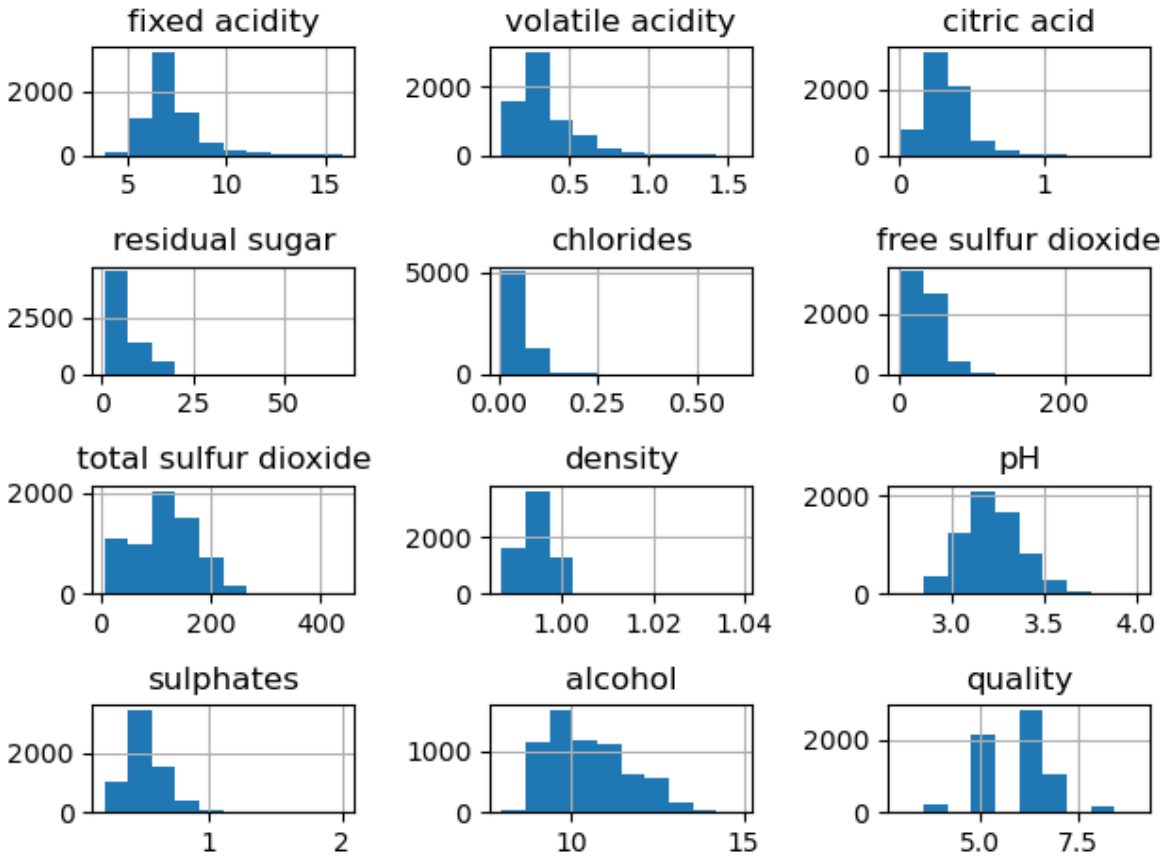
Visualizing one dimension

TODO

One of the quickest and most effective ways to visualize all numeric data and their distributions, is to leverage histograms using pandas. Plot histograms, which will give you a good idea about the basic data distribution of any of the attributes.

In [8]:
```
plt.figure(figsize=(10, 10))
wines.hist()
plt.tight_layout()
plt.show()
```

```
<Figure size 1000x1000 with 0 Axes>
```



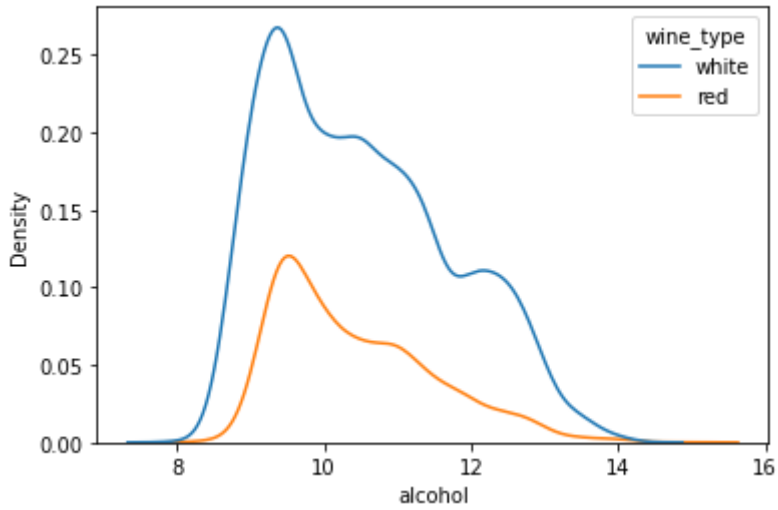
Continuous, numeric attribute in 1-D

TODO

Choose one attribute and plot it in continuous space side-by-side with its histogram. You can use `seaborn kdeplot`.

In [10]:
```
sns.kdeplot(wines, x = 'alcohol',hue='wine_type')
```

Out[10]:
```
<AxesSubplot:xlabel='alcohol', ylabel='Density'>
```



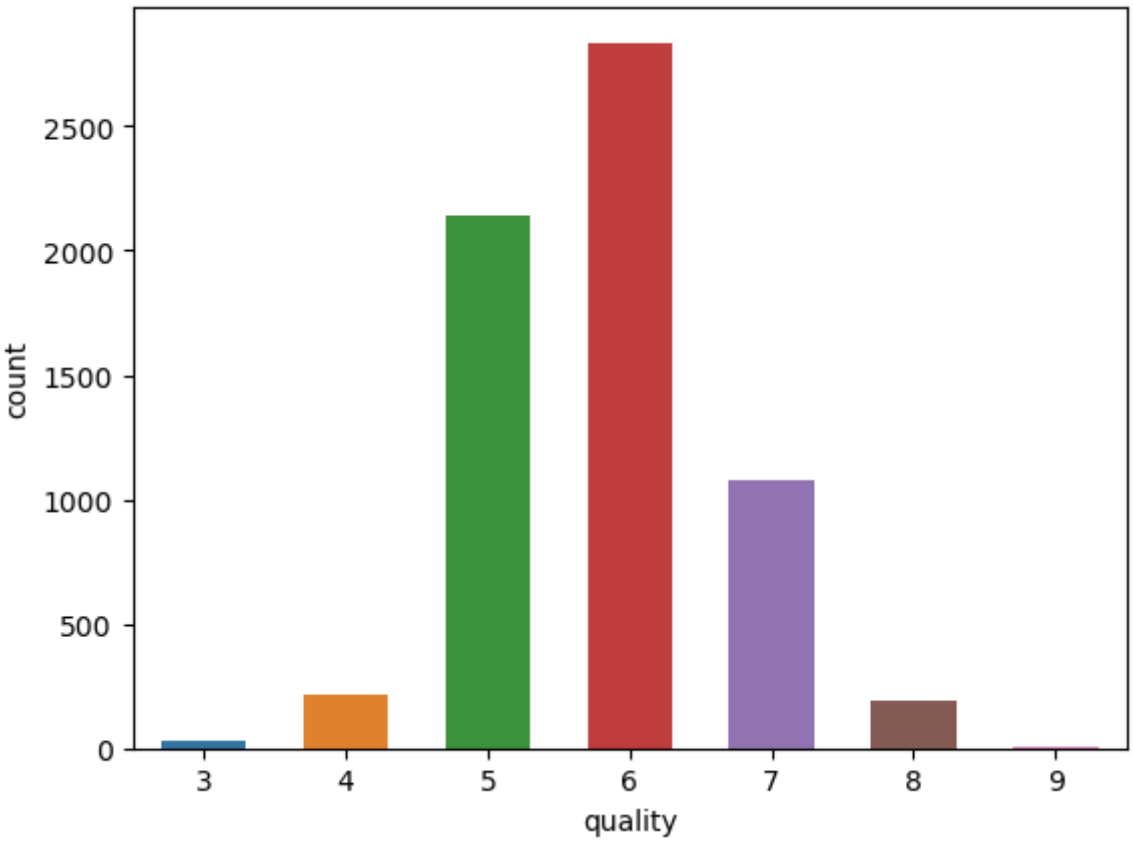
Discrete, categorical attribute in 1-D

TODO

Visualizing a discrete, categorical data attribute is slightly different and `bar plots` are one of the most effective ways to do the same. Use it, to visualiza `quality`.

In [9]:
```
sns.countplot(wines, x='quality', width=.6)
```

Out[9]:
```
<AxesSubplot: xlabel='quality', ylabel='count'>
```



Multivariate Analysis

Multivariate analysis is where the fun as well as the complexity begins. Here we analyze multiple data dimensions or attributes (2 or more). Multivariate analysis not only involves just checking out distributions but also potential relationships, patterns and correlations amongst these attributes.

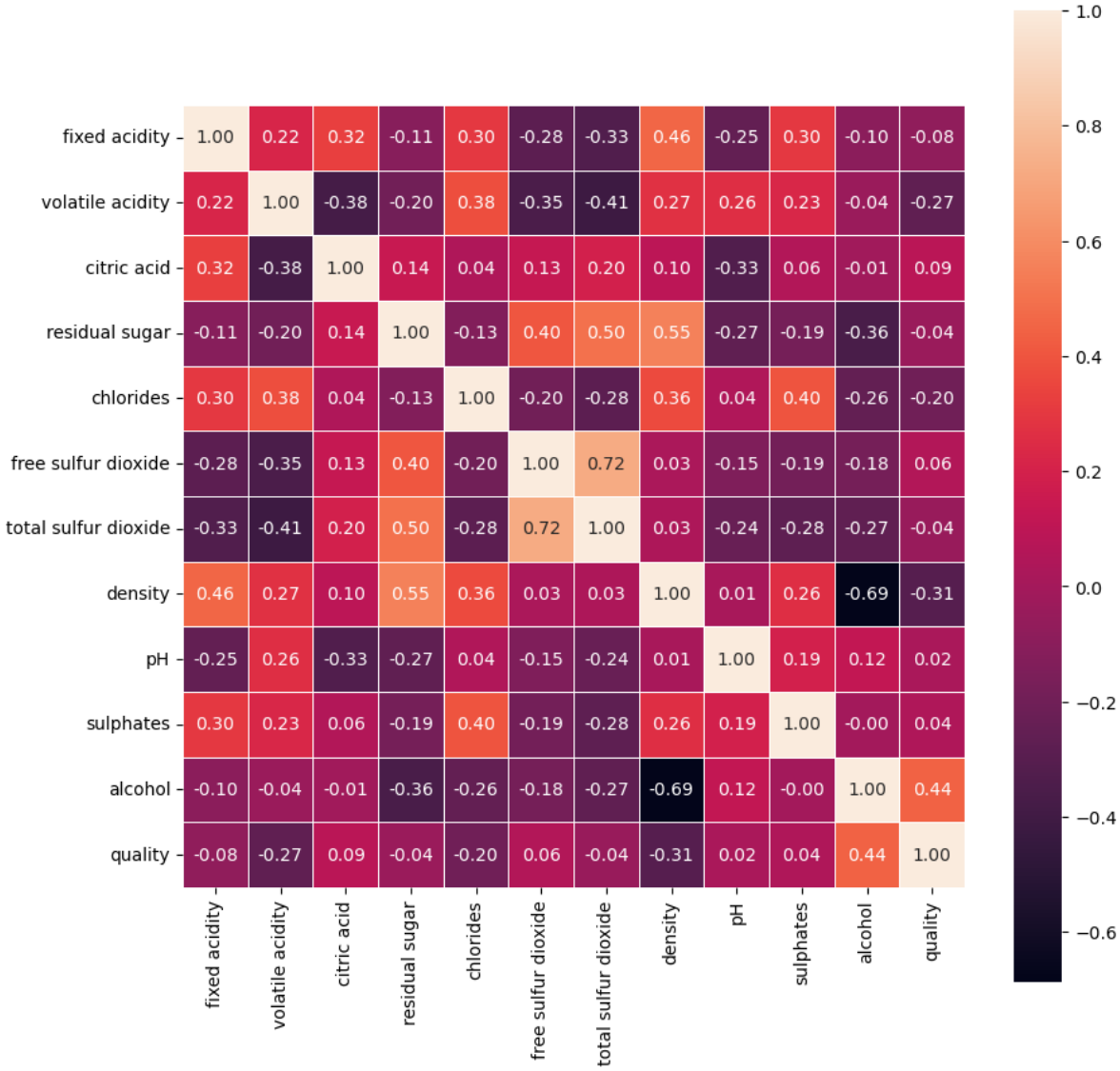
Visualizing two dimensions

TODO

One of the best ways to check out potential relationships or correlations amongst the different data attributes is to leverage a pair-wise correlation matrix (pandas `corr`) and depict it as a seaborn `heatmap`. Do it ;)

In [10]:
```
plt.figure(figsize=(10,10))
sns.heatmap(wines.corr(numeric_only=True), annot=True, fmt='.2f', linewidths=.5, sc
```

Out[10]: <AxesSubplot: >



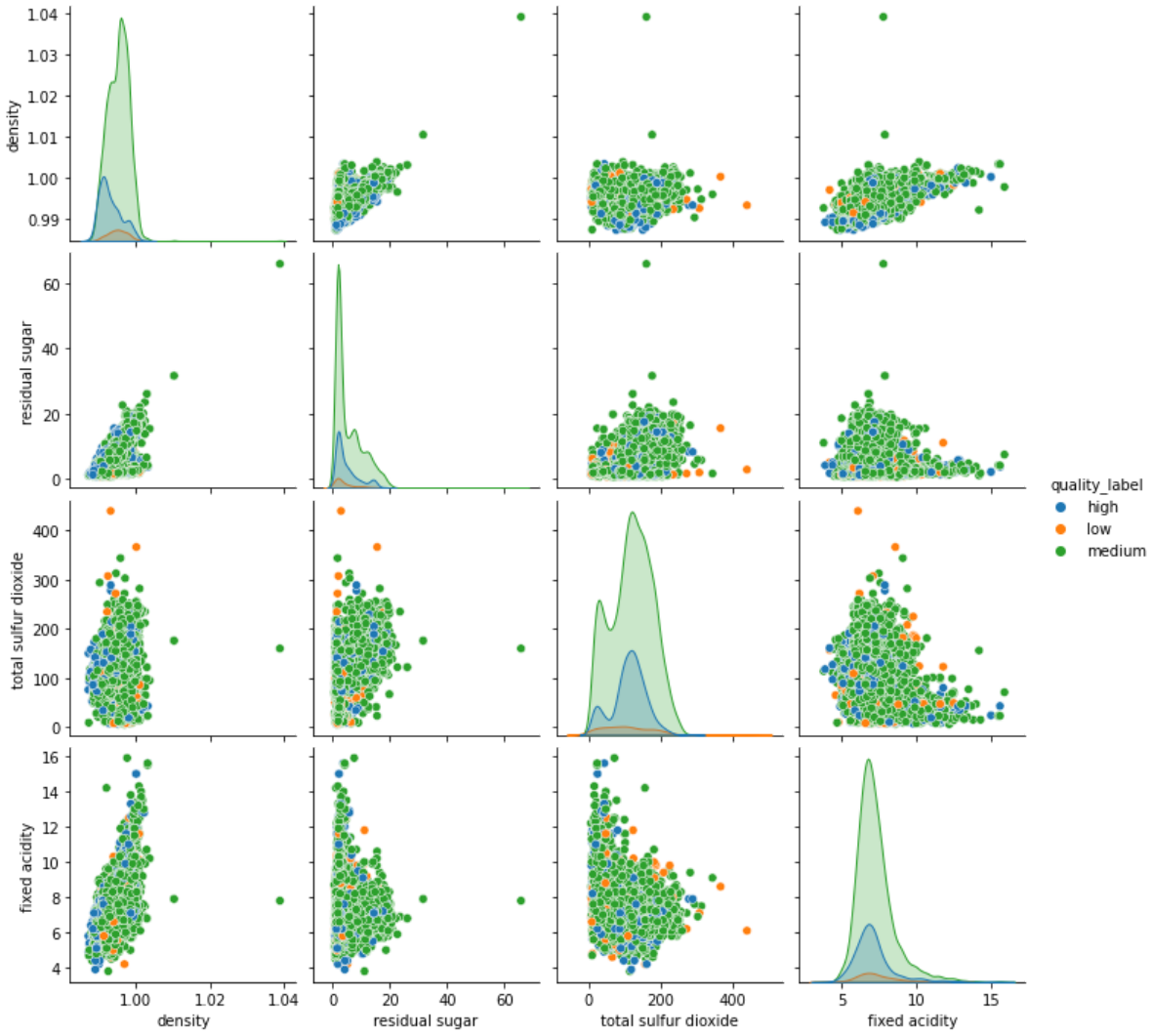
TODO

The gradients in the heatmap vary based on the strength of the correlation and you can clearly see it is very easy to spot potential attributes having strong correlations amongst themselves. Another way to visualize the same is to use pair-wise scatter plots amongst attributes of interest.

Use seaborn `pairplot` to plot correlation between 4 columns of your choosing (e.g. `cols = ['density', 'residual sugar', 'total sulfur dioxide', 'fixed acidity']`)

In [13]:
```
sns.pairplot(wines, vars =['density', 'residual sugar', 'total sulfur dioxide', 'fi
```

Out[13]: <seaborn.axisgrid.PairGrid at 0x1f8d40b9730>



TODO

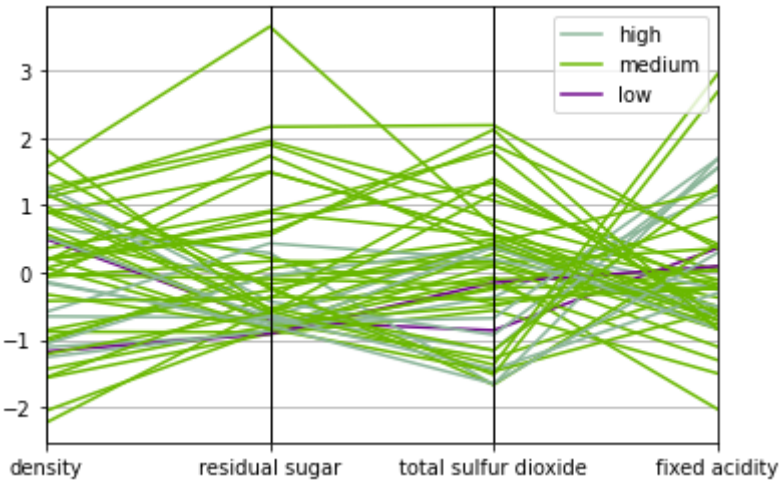
Yet another way of visualizing multivariate data for multiple attributes together is to use parallel coordinates.

`from pandas.plotting import parallel_coordinates`

To do it, it's recommended to scale your data accordingly before plotting. Use `StandardScaler` from `sklearn`.

In [14]:
```
from sklearn.preprocessing import StandardScaler
from pandas.plotting import parallel_coordinates
scaler = StandardScaler()

n = 50

scaled_data = scaler.fit_transform(wines[['density', 'residual sugar', 'total sulfu

scaled_data = pd.DataFrame(scaled_data, index=wines.index[:n], columns= ['density',
scaled_data['quality_label'] = wines['quality_label'][:n]

parallel_coordinates(scaled_data, class_column='quality_label')
```

Out[14]: <AxesSubplot:>



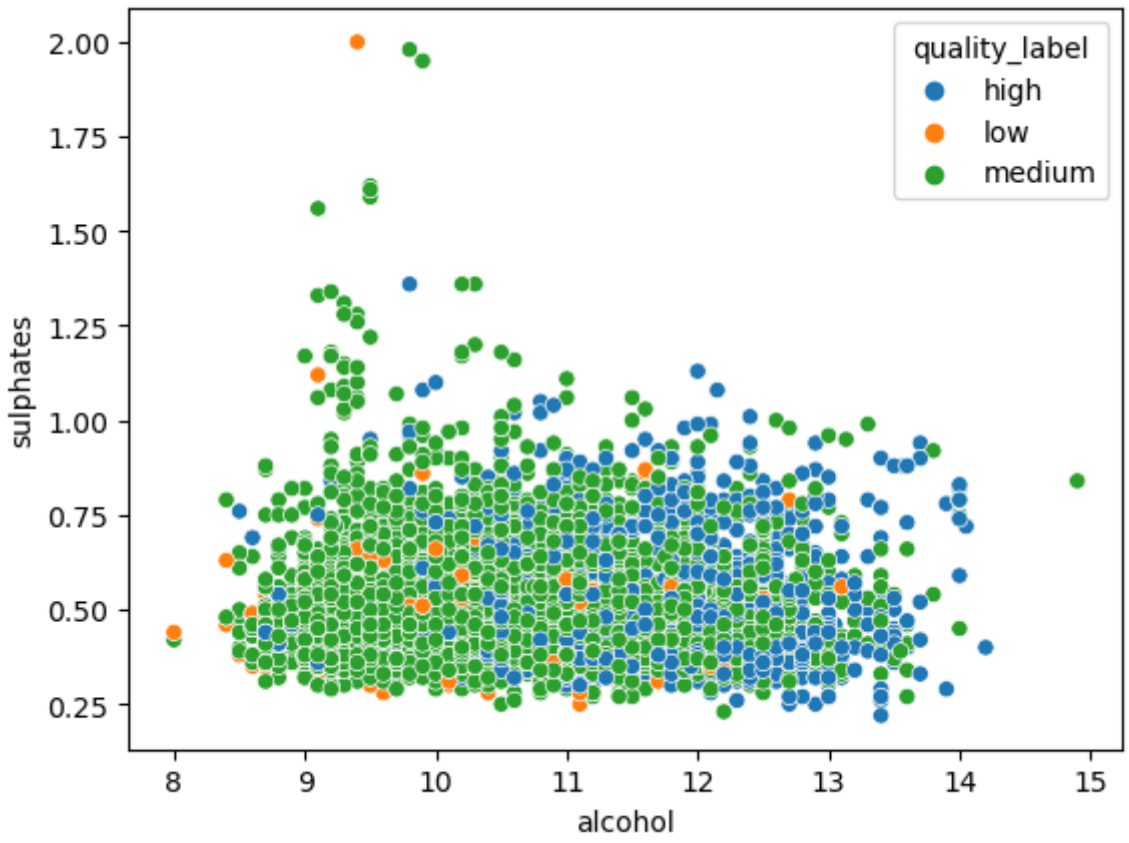
TODO

To visualize two continuous, numeric attributes, we can use scatter plots and joint plots. The second one in particular are good to not only check for patterns, relationships but also see the individual distributions for the attributes.

Choose two proper attributes (e.g. `sulphates` and `alcohol`) and plot it with scatter plot and jointplot.

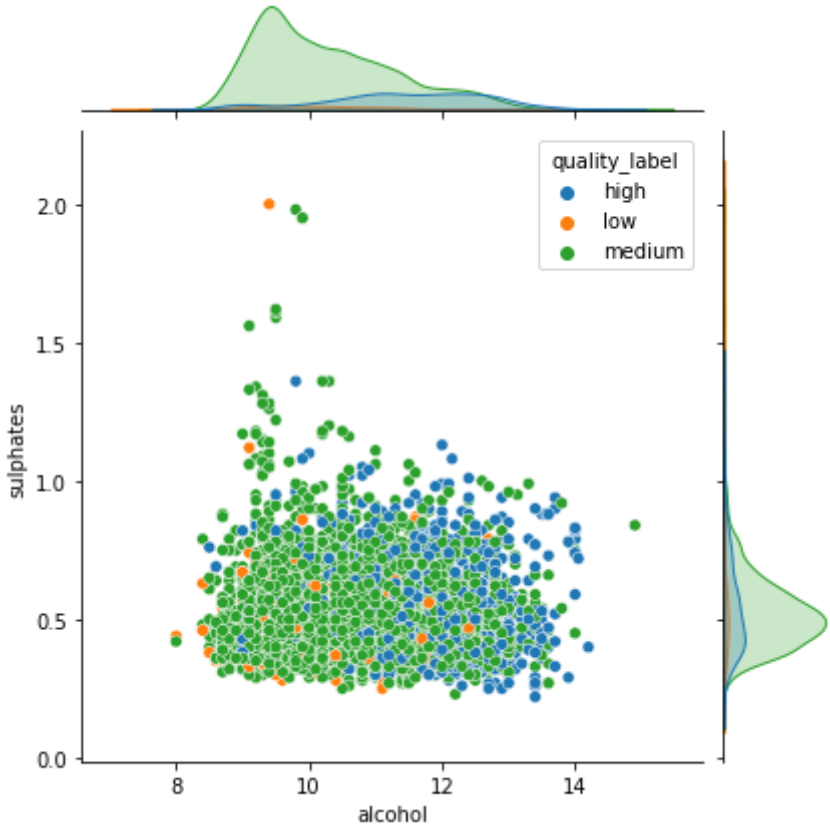
In [12]:
```
sns.scatterplot(wines, x='alcohol', y='sulphates', hue='quality_label')
```

Out[12]:
```
<AxesSubplot: xlabel='alcohol', ylabel='sulphates'>
```



In [15]:
```
sns.jointplot(wines, x='alcohol', y='sulphates', hue='quality_label')
```

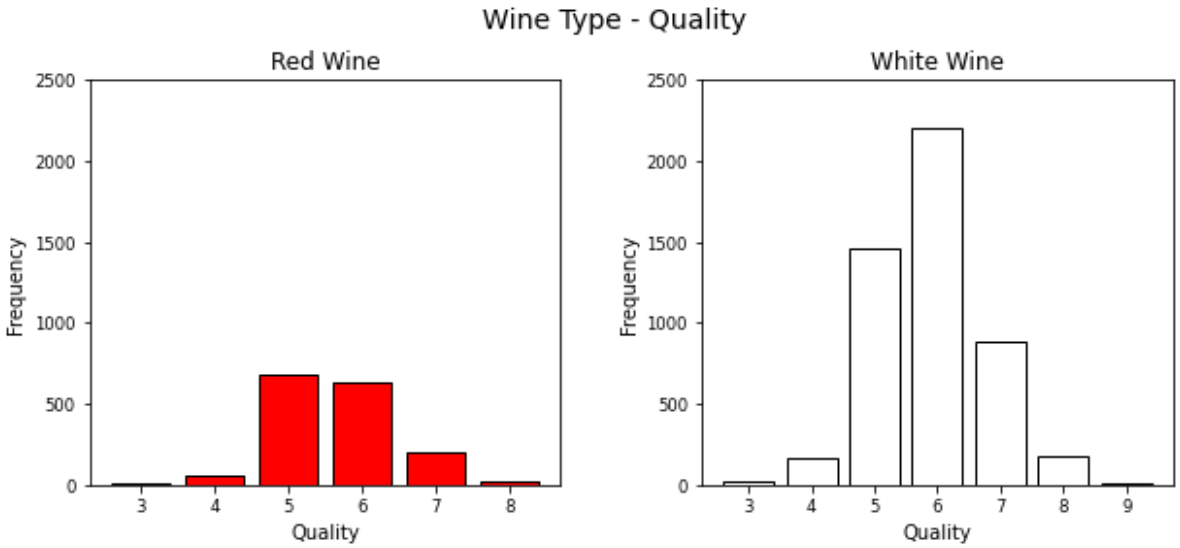
Out[15]:
```
<seaborn.axisgrid.JointGrid at 0x1f8d8d81ee0>
```



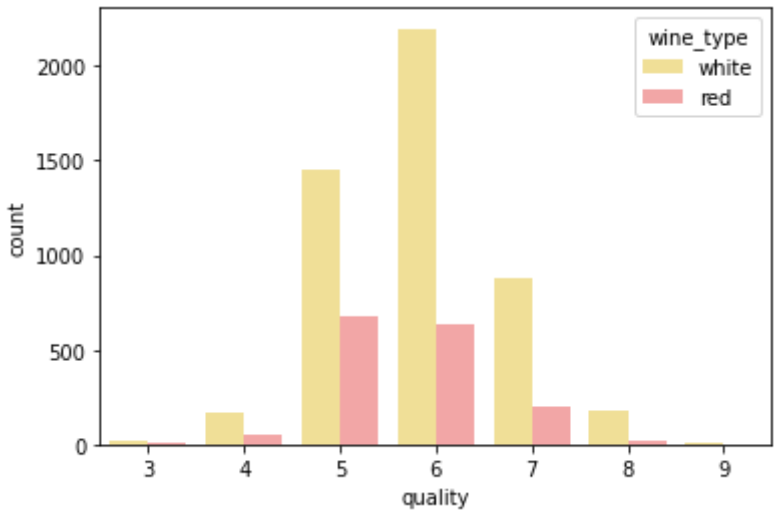
Two Discrete Categorical attributes

To visualize two discrete, categorical attributes we can leverage separate plots (subplots) or facets for one of the categorical dimensions.

In [16]:
```
fig = plt.figure(figsize = (10, 4))
title = fig.suptitle("Wine Type - Quality", fontsize=14)
fig.subplots_adjust(top=0.85, wspace=0.3)

ax1 = fig.add_subplot(1,2, 1)
ax1.set_title("Red Wine")
ax1.set_xlabel("Quality")
ax1.set_ylabel("Frequency")
rw_q = dataset_red['quality'].value_counts()
rw_q = (list(rw_q.index), list(rw_q.values))
ax1.set_ylim([0, 2500])
ax1.tick_params(axis='both', which='major', labelsize=8.5)
bar1 = ax1.bar(rw_q[0], rw_q[1], color='red',
        edgecolor='black', linewidth=1)


ax2 = fig.add_subplot(1,2, 2)
ax2.set_title("White Wine")
ax2.set_xlabel("Quality")
ax2.set_ylabel("Frequency")
ww_q = dataset_white['quality'].value_counts()
ww_q = (list(ww_q.index), list(ww_q.values))
ax2.set_ylim([0, 2500])
ax2.tick_params(axis='both', which='major', labelsize=8.5)
bar2 = ax2.bar(ww_q[0], ww_q[1], color='white',
        edgecolor='black', linewidth=1)
```



In [17]:
```
cp = sns.countplot(x="quality", hue="wine_type", data=wines,
                   palette={"red": "#FF9999", "white": "#FFE888"})
```

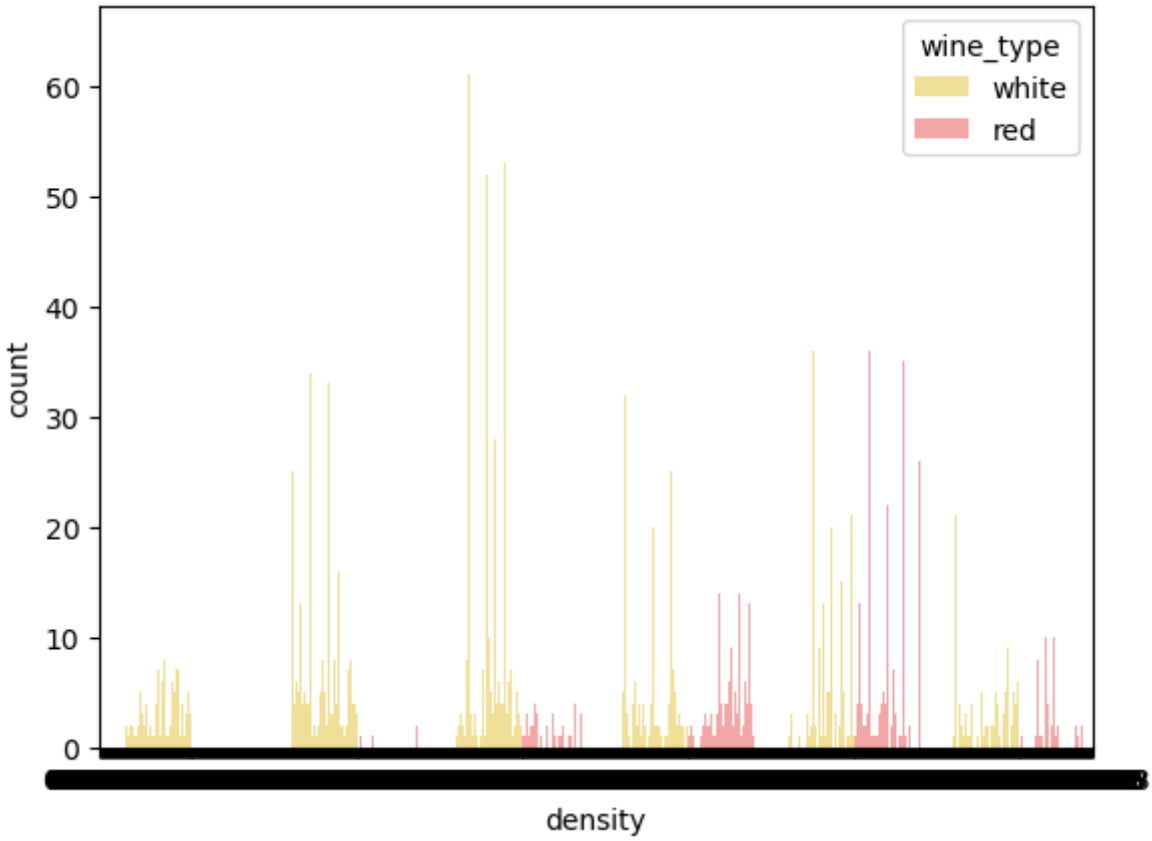


Mixed attributes (numeric & categorical)

To visualize mixed attributes in two-dimensions (essentially numeric and categorical together), we cab use faceting\subplots along with generic histograms or density plots.

Use example above to plot "Sulphates content in wine" histograms and facets (`Frequency` / `Density` and `Sulphates` as axis) for different types of wine.

In [17]:
```
cp = sns.countplot(x="density", hue="wine_type", data=wines,
                   palette={"red": "#FF9999", "white": "#FFE888"})
```



In [16]:
```
wines
```

Out[16]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcol
0	6.1	0.340	0.27	2.6	0.024	20.0	105.0	0.99060	3.40	0.67	1
1	7.3	0.430	0.24	2.5	0.078	27.0	67.0	0.99648	3.60	0.59	1
2	6.9	0.230	0.35	6.9	0.030	45.0	116.0	0.99244	2.80	0.54	1
3	7.2	0.340	0.20	5.8	0.062	52.0	203.0	0.99461	3.17	0.44	
4	7.5	0.190	0.40	7.1	0.056	50.0	110.0	0.99540	3.06	0.52	
...	...	...	...	...	...	...	...	...	...	...	
6492	7.2	0.650	0.02	2.3	0.094	5.0	31.0	0.99930	3.67	0.80	
6493	6.7	0.500	0.38	7.5	0.046	26.0	175.0	0.99662	3.32	0.54	
6494	6.3	0.400	0.32	10.6	0.049	38.0	209.0	0.99810	3.47	0.59	
6495	5.3	0.585	0.07	7.1	0.044	34.0	145.0	0.99450	3.34	0.57	
6496	7.6	0.510	0.24	2.4	0.091	8.0	38.0	0.99800	3.47	0.66	

6497 rows × 14 columns

TODO

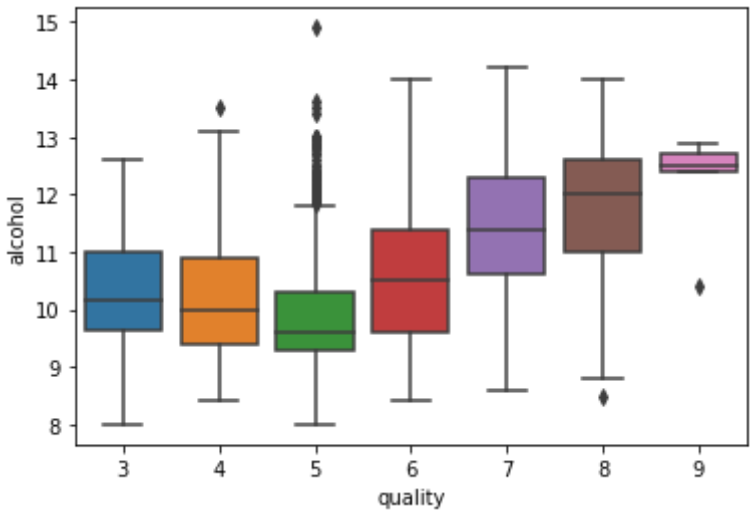
Box plots are effective in depicting groups of numeric data based on the different values in the categorical attribute. Additionally, they are a good way to know the quartile values in

the data and also potential outliers.

Plot a `boxplot` for `quality` and `alcohol` as axis. Then do the same for `violinplot` and verify the differences between the two.

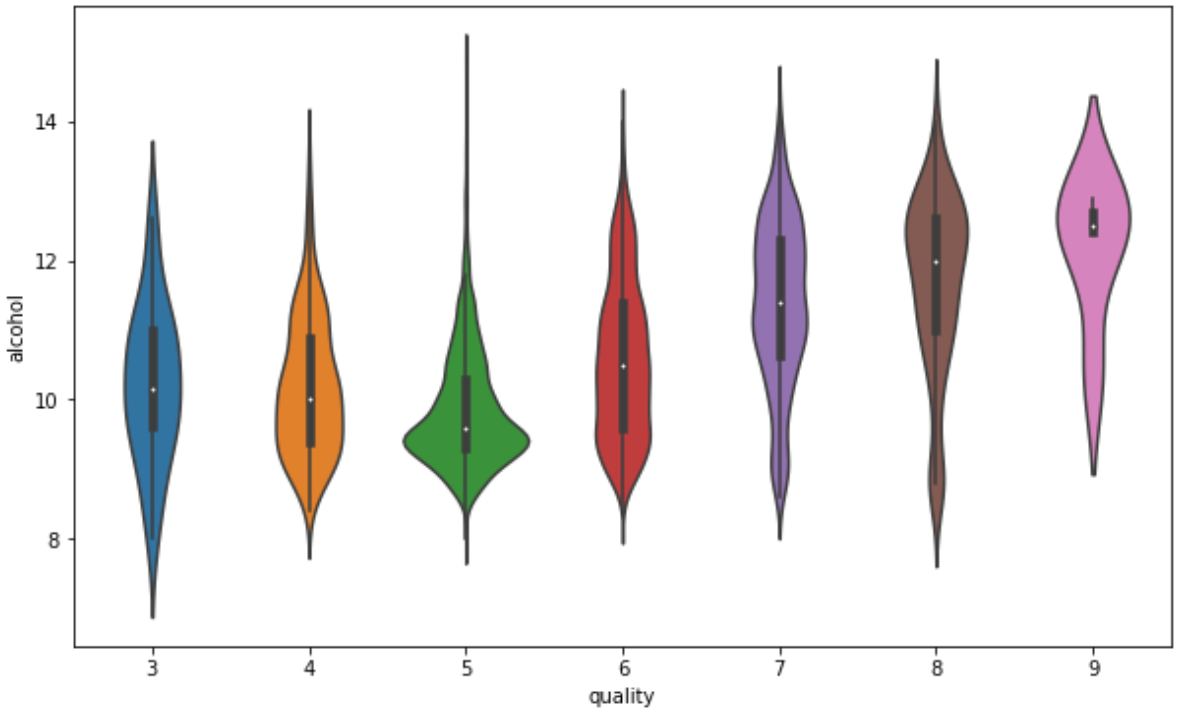
In [18]:
```
sns.boxplot(wines, y='alcohol',x='quality')
```

Out[18]:
```
<AxesSubplot:xlabel='quality', ylabel='alcohol'>
```



In [19]:
```
plt.figure(figsize=(10, 6))
sns.violinplot(wines, y='alcohol', x='quality')
```

Out[19]:
```
<AxesSubplot:xlabel='quality', ylabel='alcohol'>
```



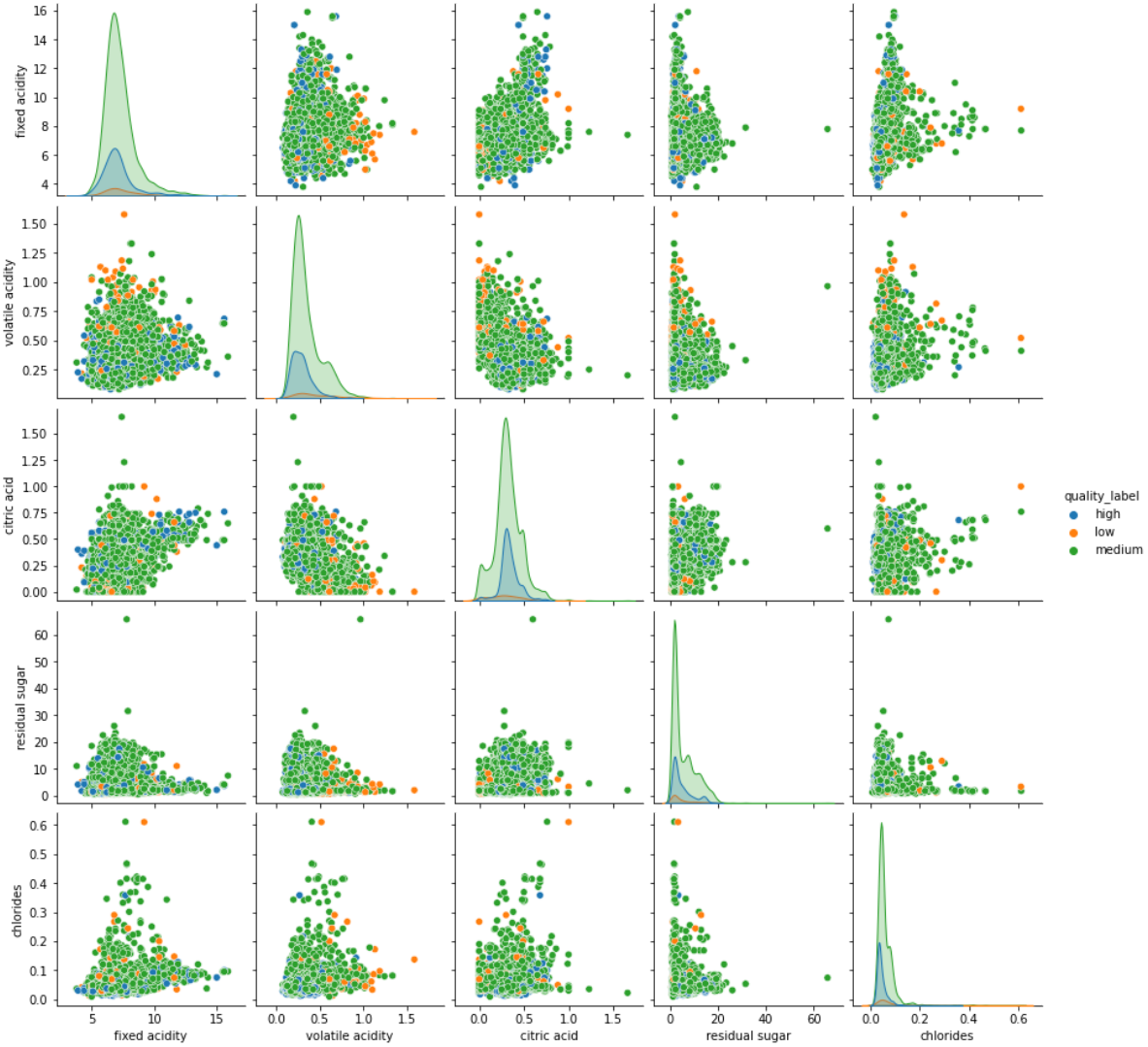
Visualizing three dimensions

We can visualize them by considering a pair-wise scatter plot and introducing the notion of color or hue to separate out values in a categorical dimension.

Plot 5 columns of your choosing with `wine_type` (one of columns) as 3rd dimension.

In [20]:
```
sns.pairplot(wines, vars=wines.columns[:5], hue='quality_label')
```

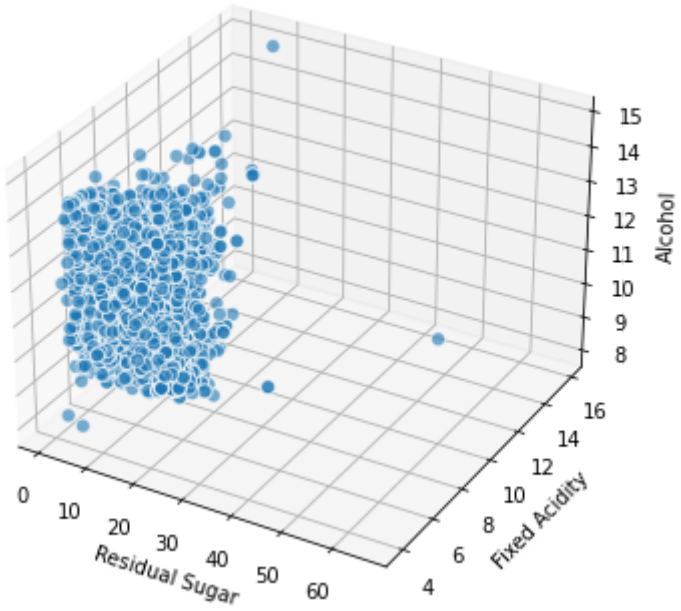
Out[20]:
```
<seaborn.axisgrid.PairGrid at 0x1f8da89b9d0>
```



Three Continuous Numeric attributes

In [21]:
```
fig = plt.figure(figsize=(8, 6))
ax = fig.add_subplot(111, projection='3d')

xs = wines['residual sugar']
ys = wines['fixed acidity']
zs = wines['alcohol']
ax.scatter(xs, ys, zs, s=50, alpha=0.6, edgecolors='w')

ax.set_xlabel('Residual Sugar')
ax.set_ylabel('Fixed Acidity')
ax.set_zlabel('Alcohol')
```

Out[21]: Text(0.5, 0, 'Alcohol')



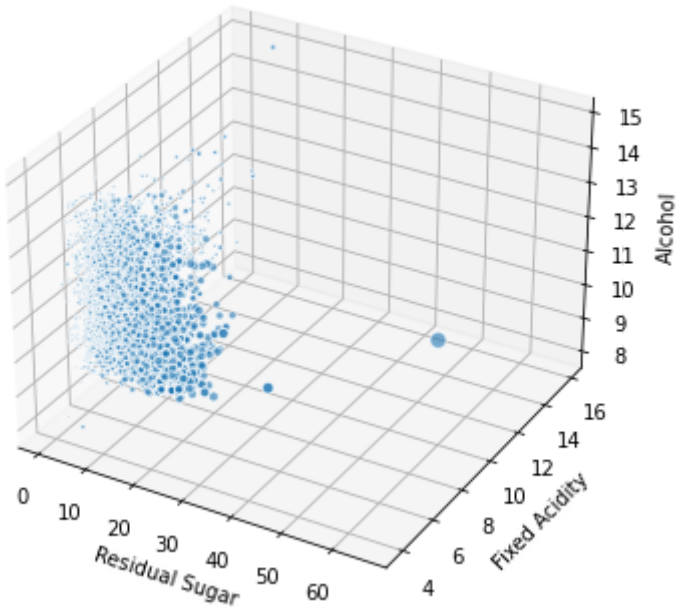
TODO

We can utilize size as the third dimension, where the size of the dots indicate the quantity of the third dimension.

Use `s` parameter of scatter plot to parametrize it by the size. Use `residual sugar` as the size indicator.

In [22]:
```
fig = plt.figure(figsize=(8, 6))
ax = fig.add_subplot(111, projection='3d')

xs = wines['residual sugar']
ys = wines['fixed acidity']
zs = wines['alcohol']
ax.scatter(xs, ys, zs, s=wines['residual sugar'], alpha=0.6, edgecolors='w')

ax.set_xlabel('Residual Sugar')
ax.set_ylabel('Fixed Acidity')
ax.set_zlabel('Alcohol')
```

Out[22]: Text(0.5, 0, 'Alcohol')



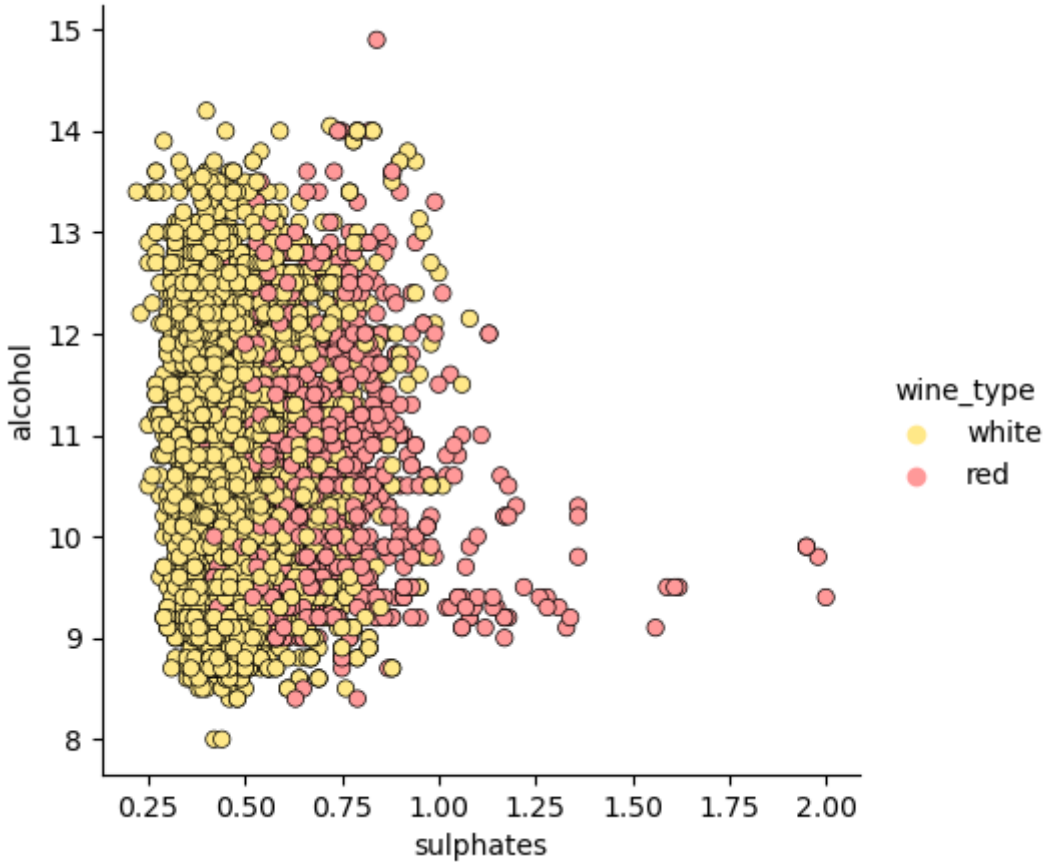
Three Discrete Categorical attributes

In [26]:
```
fc = sns.factorplot(x="quality", hue="wine_type", col="quality_label",
                    data=wines, kind="count",
                    palette={"red": "#FF9999", "white": "#FFE888"})
```

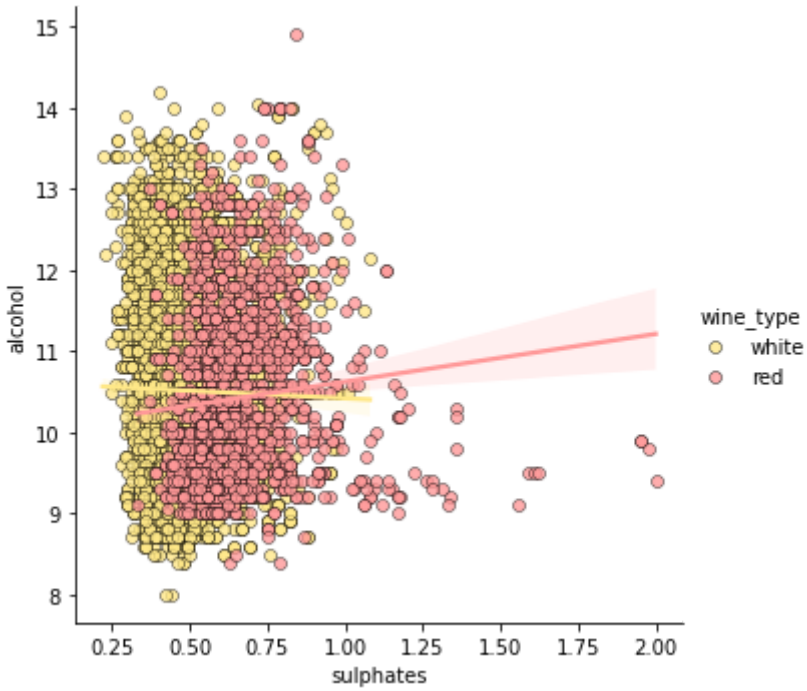
```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
c:\Users\Filif\Desktop\AGH_DataScience\Semestr1\wdzd\winequality-white\descriptive_analytics.ipynb Cell 47 in <cell line: 1>()
----> <a href='vscode-notebook-cell:/c%3A/Users/Filif/Desktop/AGH_DataScience/Semestr1/wdzd/winequality-white/descriptive_analytics.ipynb#X64sZmlsZQ%3D%3D?line=0'>1</a> fc = sns.factorplot(x="quality", hue="wine_type", col="quality_label",
      <a href='vscode-notebook-cell:/c%3A/Users/Filif/Desktop/AGH_DataScience/Semestr1/wdzd/winequality-white/descriptive_analytics.ipynb#X64sZmlsZQ%3D%3D?line=1'>2</a>                     data=wines, kind="count",
      <a href='vscode-notebook-cell:/c%3A/Users/Filif/Desktop/AGH_DataScience/Semestr1/wdzd/winequality-white/descriptive_analytics.ipynb#X64sZmlsZQ%3D%3D?line=2'>3</a>                     palette={"red": "#FF9999", "white": "#FFE888"})

AttributeError: module 'seaborn' has no attribute 'factorplot'
```

Mixed attributes (Numeric & Categorical)

In [14]:
```
jp = sns.pairplot(wines, x_vars=["sulphates"], y_vars=["alcohol"], height=4.5,
                  hue="wine_type", palette={"red": "#FF9999", "white": "#FFE888"},
                  plot_kws=dict(edgecolor="k", linewidth=0.5))
```



In [28]:
```
lp = sns.lmplot(x='sulphates', y='alcohol', hue='wine_type',
                palette={"red": "#FF9999", "white": "#FFE888"},
                data=wines, fit_reg=True, legend=True,
                scatter_kws=dict(edgecolor="k", linewidth=0.5))
```



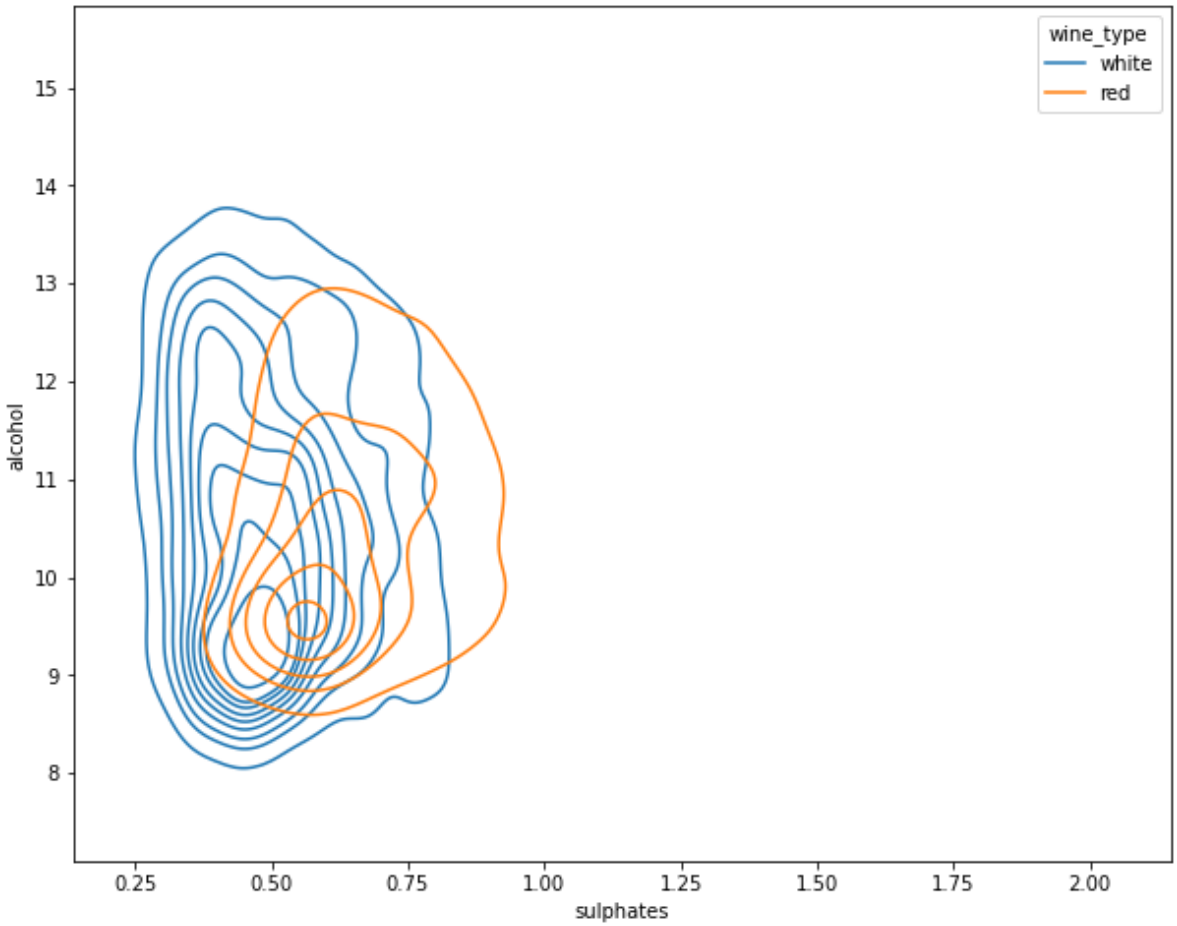
TODO

Thus hue acts as a good separator for the categories or groups and while there is no or very weak correlation as observed above, we can still understand from these plots that `sulphates` are slightly higher for red wines as compared to white.

Instead of a scatter plot, use a kernel density plot (`kdeplot`) to understand the data in three dimensions.

In [32]:
```
plt.figure(figsize=(10, 8))
sns.kdeplot(wines, x='sulphates', y ='alcohol', hue='wine_type')
```

Out[32]:
```
<AxesSubplot:xlabel='sulphates', ylabel='alcohol'>
```

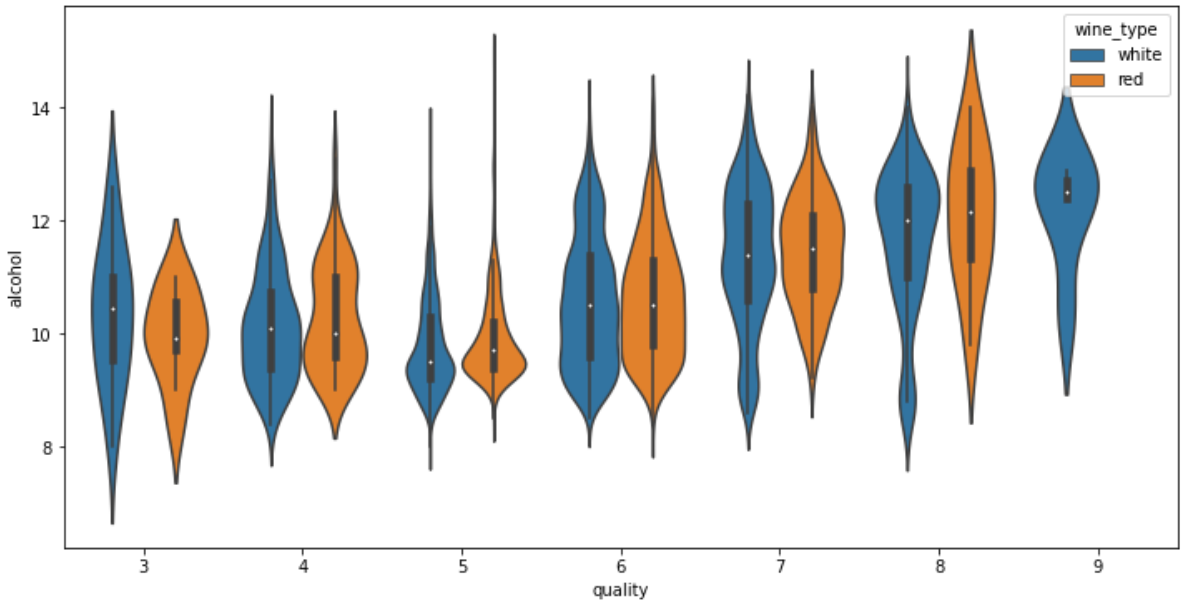


TODO

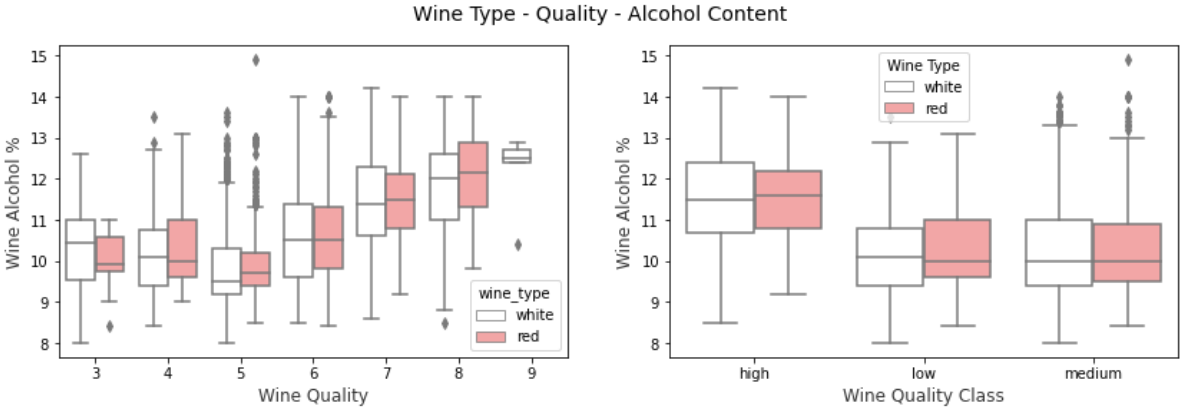
Use hue and one of the regular axes for visualizing data and use visualizations like box plots or violin plots to visualize the different groups of data.

In [34]:
```
plt.figure(figsize=(12, 6))
sns.violinplot(wines, y='alcohol', x='quality', hue='wine_type')
```

Out[34]:
```
<AxesSubplot:xlabel='quality', ylabel='alcohol'>
```



In [35]:
```
f, (ax1, ax2) = plt.subplots(1, 2, figsize=(14, 4))
f.suptitle('Wine Type - Quality - Alcohol Content', fontsize=14)

sns.boxplot(x="quality", y="alcohol", hue="wine_type",
               data=wines, palette={"red": "#FF9999", "white": "white"}, ax=ax1)
ax1.set_xlabel("Wine Quality",size = 12,alpha=0.8)
ax1.set_ylabel("Wine Alcohol %",size = 12,alpha=0.8)

sns.boxplot(x="quality_label", y="alcohol", hue="wine_type",
               data=wines, palette={"red": "#FF9999", "white": "white"}, ax=ax2)
ax2.set_xlabel("Wine Quality Class",size = 12,alpha=0.8)
ax2.set_ylabel("Wine Alcohol %",size = 12,alpha=0.8)
l = plt.legend(loc='best', title='Wine Type')
```



Visualizing four dimensions

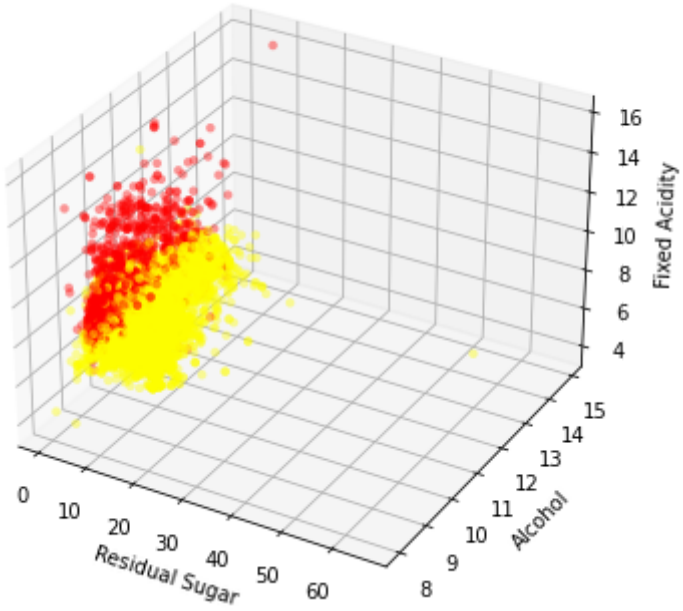
We will leverage various components of the charts visualize multiple dimensions. One way to visualize data in four dimensions is to use `depth` and `hue` as specific data dimensions in a conventional plot like a scatter plot.

Use: `xs` , `ys` , `zs` for plot (tips: use `zip`) and specify color by the `wine_type` .

In [42]:
```
fig = plt.figure(figsize=(8, 6))
ax = fig.add_subplot(111, projection='3d')

xs = list(wines['residual sugar'])
```

```
ys = list(wines['alcohol'])
zs = list(wines['fixed acidity'])
data_points = [(x, y, z) for x, y, z in zip(xs, ys, zs)]

colors = ['red' if wt == 'red' else 'yellow' for wt in list(wines['wine_type'])]

for data, color in zip(data_points, colors):
    x, y, z = data
    ax.scatter(x, y, z, alpha=0.4, c=color, edgecolors='none')

ax.set_xlabel('Residual Sugar')
ax.set_ylabel('Alcohol')
ax.set_zlabel('Fixed Acidity')
```

Out[42]: Text(0.5, 0, 'Fixed Acidity')

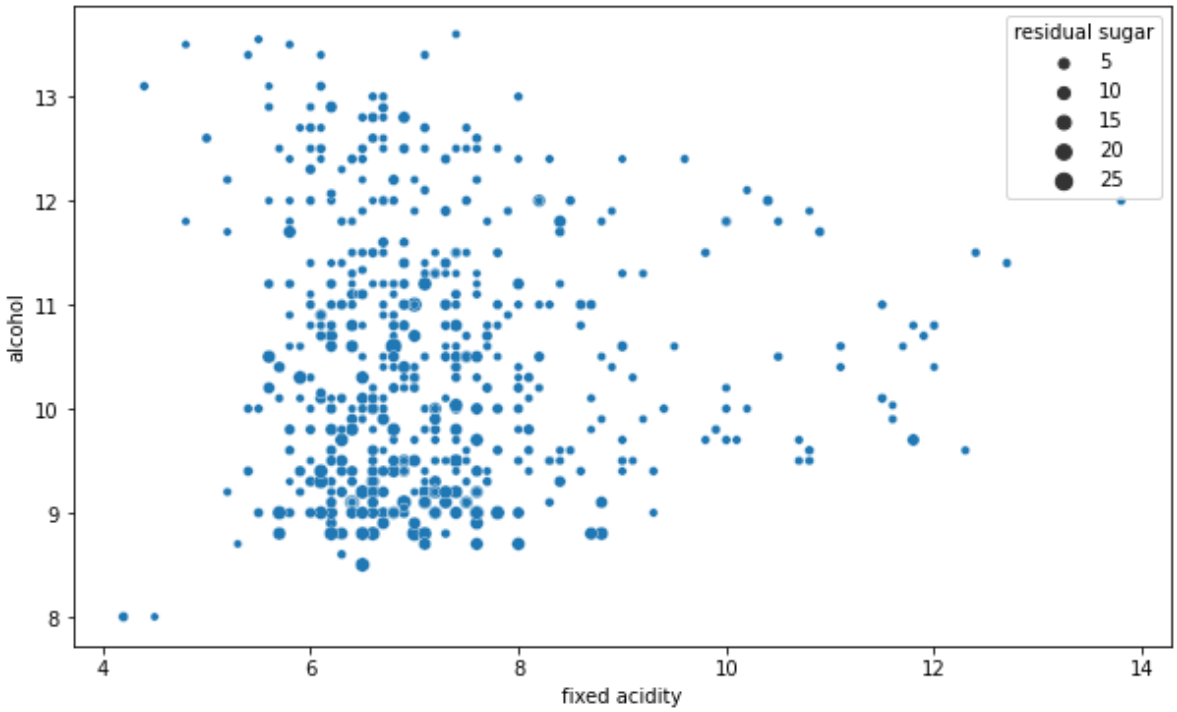


TODO

Another strategy is to keep a 2-D plot but use hue and data point size as data dimensions. Typically this would be a bubble chart similar to what we visualized earlier. Create bubble plot with specified size and color.

In [48]:
```
plt.figure(figsize=(10, 6))
# ograniczone do 500 lepiej wyglądа
sns.scatterplot(wines.iloc[:500], x='fixed acidity', y = 'alcohol', size='residual
```

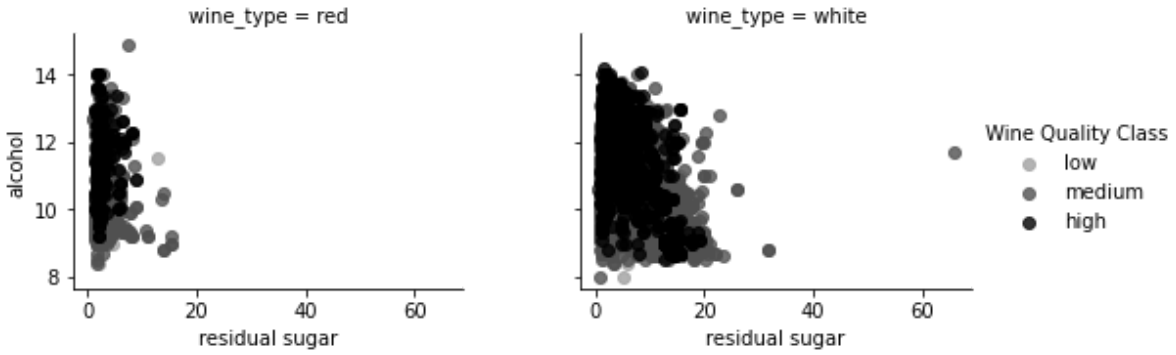
Out[48]: <AxesSubplot:xlabel='fixed acidity', ylabel='alcohol'>



TODO

If we have more that two categorical attributes to represent, we can reuse our concept of leveraging hue and facets to depict these attributes and regular plots like scatter plots to represent the numeric attributes. Use `FaceGrid` to plot a 4-D mix data using scatter plots leveraging the concepts of hue and facets for > 1 categorical attributes. Use `map` method of the `FaceGrid` object. Compare different attributes. For example "volatile acidity", "alcohol" and "volatile acidity", "total sulfur dioxide".

In [52]:
```
g = sns.FacetGrid(wines, col="wine_type", hue='quality_label',
                  col_order=['red', 'white'], hue_order=['low', 'medium', 'high'],
                  aspect=1.2, palette=sns.light_palette('black', 4)[1:])
g.map(plt.scatter, "residual sugar", "alcohol", alpha=0.8)
fig = g.fig
fig.subplots_adjust(top=0.8, wspace=0.3)
l = g.add_legend(title='Wine Quality Class')
```



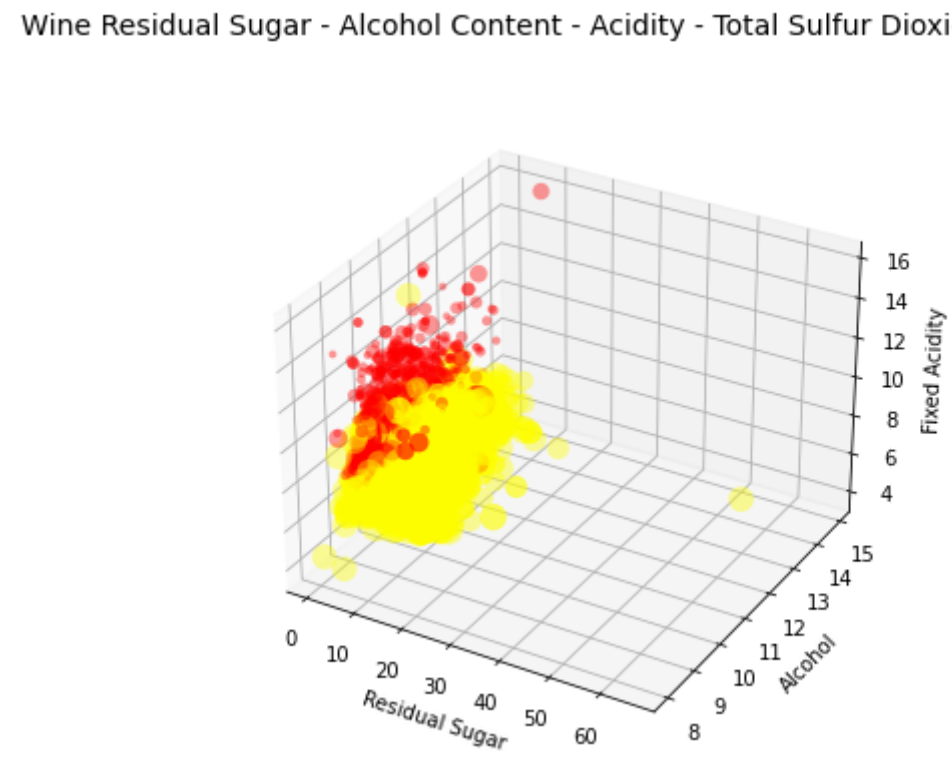
Visualizing five dimensions

Once again following a similar strategy as we followed in the previous section, to visualize data in five dimensions, we leverage various plotting components. Let's use depth, hue and

size to represent three of the data dimensions besides regular axes representing the other two dimensions. Since we use the notion of size, we will be basically plotting a three dimensional bubble chart.

In [36]:
```
fig = plt.figure(figsize=(8, 6))
ax = fig.add_subplot(111, projection='3d')
t = fig.suptitle('Wine Residual Sugar - Alcohol Content - Acidity - Total Sulfur Di

xs = list(wines['residual sugar'])
ys = list(wines['alcohol'])
zs = list(wines['fixed acidity'])
data_points = [(x, y, z) for x, y, z in zip(xs, ys, zs)]

ss = list(wines['total sulfur dioxide'])
colors = ['red' if wt == 'red' else 'yellow' for wt in list(wines['wine_type'])]

for data, color, size in zip(data_points, colors, ss):
    x, y, z = data
    ax.scatter(x, y, z, alpha=0.4, c=color, edgecolors='none', s=size)

ax.set_xlabel('Residual Sugar')
ax.set_ylabel('Alcohol')
ax.set_zlabel('Fixed Acidity')
```

Out[36]:
```
Text(0.5, 0, 'Fixed Acidity')
```



In [39]:
```
g = sns.FacetGrid(wines, col="wine_type", hue='quality_label',
                  col_order=['red', 'white'], hue_order=['low', 'medium', 'high'],
                  aspect=1.2, size=3.5, palette=sns.light_palette('black', 4)[1:])
g.map(plt.scatter, "residual sugar", "alcohol", alpha=0.8,
      edgecolor='white', linewidth=0.5, s=wines['total sulfur dioxide']*2)
fig = g.fig
fig.subplots_adjust(top=0.8, wspace=0.3)
fig.suptitle('Wine Type - Sulfur Dioxide - Residual Sugar - Alcohol - Quality', for
l = g.add_legend(title='Wine Quality Class')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
c:\Users\Filif\Desktop\AGH_DataScience\Semestr1\wdzd\winequality-white\descriptive
_analytics.ipynb Cell 66 in <cell line: 1>()
----> <a href='vscode-notebook-cell:/c%3A/Users/Filif/Desktop/AGH_DataScience/Seme
str1/wdzd/winequality-white/descriptive_analytics.ipynb#Y122sZmlsZQ%3D%3D?line=0'>
1</a> g = sns.FacetGrid(wines, col="wine_type", hue='quality_label',
      <a href='vscode-notebook-cell:/c%3A/Users/Filif/Desktop/AGH_DataScience/Seme
str1/wdzd/winequality-white/descriptive_analytics.ipynb#Y122sZmlsZQ%3D%3D?line=1'>
2</a>                 col_order=['red', 'white'], hue_order=['low', 'medium', 'h
igh'],
      <a href='vscode-notebook-cell:/c%3A/Users/Filif/Desktop/AGH_DataScience/Seme
str1/wdzd/winequality-white/descriptive_analytics.ipynb#Y122sZmlsZQ%3D%3D?line=2'>
3</a>                 aspect=1.2, size=3.5, palette=sns.light_palette('black',
4)[1:])
      <a href='vscode-notebook-cell:/c%3A/Users/Filif/Desktop/AGH_DataScience/Seme
str1/wdzd/winequality-white/descriptive_analytics.ipynb#Y122sZmlsZQ%3D%3D?line=3'>
4</a> g.map(plt.scatter, "residual sugar", "alcohol", alpha=0.8,
      <a href='vscode-notebook-cell:/c%3A/Users/Filif/Desktop/AGH_DataScience/Seme
str1/wdzd/winequality-white/descriptive_analytics.ipynb#Y122sZmlsZQ%3D%3D?line=4'>
5</a>       edgecolor='white', linewidth=0.5, s=wines['total sulfur dioxide']*2)
      <a href='vscode-notebook-cell:/c%3A/Users/Filif/Desktop/AGH_DataScience/Seme
str1/wdzd/winequality-white/descriptive_analytics.ipynb#Y122sZmlsZQ%3D%3D?line=5'>
6</a> fig = g.fig

TypeError: __init__() got an unexpected keyword argument 'size'
```

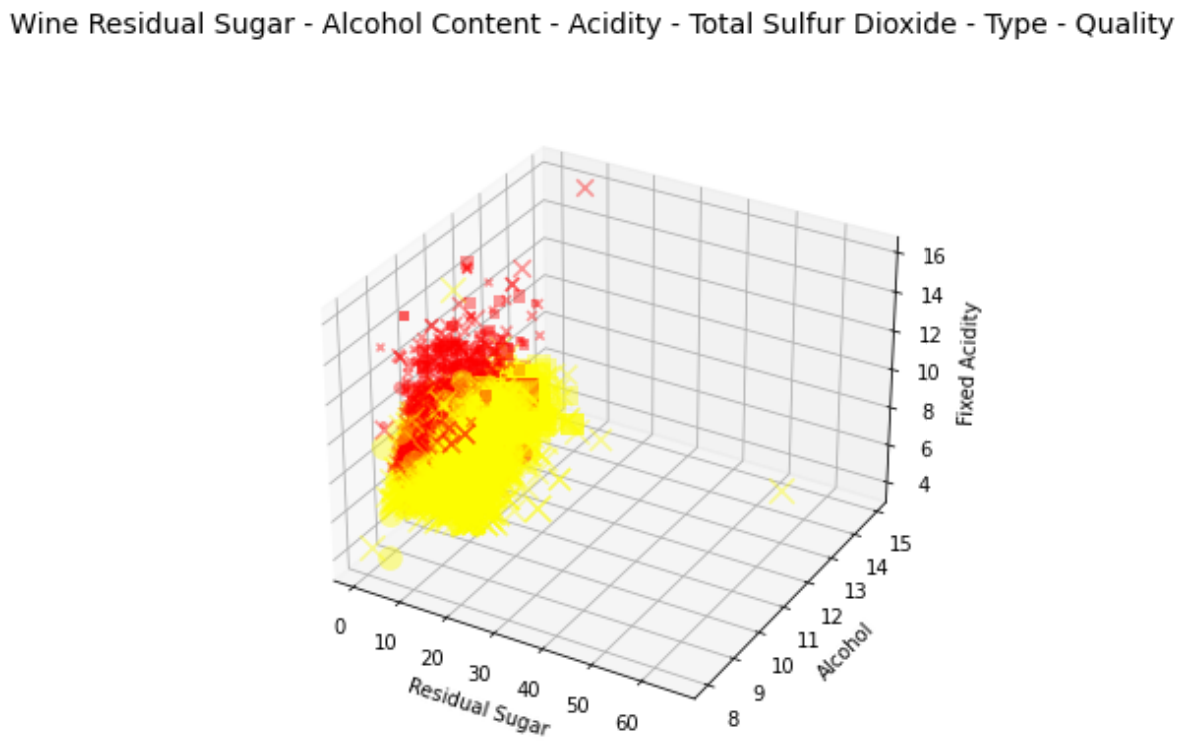
Visualizing six dimensions

Now, let's add another data dimension in our visualizations. We will leverage depth, hue, size and shape besides our regular two axes to depict all the six data dimensions.

In [40]:
```
fig = plt.figure(figsize=(8, 6))
t = fig.suptitle('Wine Residual Sugar - Alcohol Content - Acidity - Total Sulfur Di
ax = fig.add_subplot(111, projection='3d')

xs = list(wines['residual sugar'])
ys = list(wines['alcohol'])
zs = list(wines['fixed acidity'])
data_points = [(x, y, z) for x, y, z in zip(xs, ys, zs)]

ss = list(wines['total sulfur dioxide'])
colors = ['red' if wt == 'red' else 'yellow' for wt in list(wines['wine_type'])]
markers = [',' if q == 'high' else 'x' if q == 'medium' else 'o' for q in list(wine

for data, color, size, mark in zip(data_points, colors, ss, markers):
    x, y, z = data
    ax.scatter(x, y, z, alpha=0.4, c=color, edgecolors='none', s=size, marker=mark)

ax.set_xlabel('Residual Sugar')
ax.set_ylabel('Alcohol')
ax.set_zlabel('Fixed Acidity')
```

```
C:\Users\Filif\AppData\Local\Temp\ipykernel_10224\2127194094.py:16: UserWarning: Y
ou passed a edgecolor/edgecolors ('none') for an unfilled marker ('x').  Matplotli
b is ignoring the edgecolor in favor of the facecolor.  This behavior may change i
n the future.
  ax.scatter(x, y, z, alpha=0.4, c=color, edgecolors='none', s=size, marker=mark)
```

Out[40]: Text(0.5, 0, 'Fixed Acidity')



In [41]:
```
g = sns.FacetGrid(wines, row='wine_type', col="quality", hue='quality_label', size=
g.map(plt.scatter,  "residual sugar", "alcohol", alpha=0.5,
      edgecolor='k', linewidth=0.5, s=wines['total sulfur dioxide']*2)
fig = g.fig
fig.set_size_inches(18, 8)
fig.subplots_adjust(top=0.85, wspace=0.3)
fig.suptitle('Wine Type - Sulfur Dioxide - Residual Sugar - Alcohol - Quality Class
l = g.add_legend(title='Wine Quality Class')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
c:\Users\Filif\Desktop\AGH_DataScience\Semestr1\wdzd\winequality-white\descriptive
_analytics.ipynb Cell 70 in <cell line: 1>()
----> 1 g = sns.FacetGrid(wines, row='wine_type', col="quality", hue='quality_labe
l', size=4)
      2 g.map(plt.scatter,  "residual sugar", "alcohol", alpha=0.5,
      3       edgecolor='k', linewidth=0.5, s=wines['total sulfur dioxide']*2)
      4 fig = g.fig

TypeError: __init__() got an unexpected keyword argument 'size'
```

In [ ]: