Data Visualization

Digital Assignment - 2

19BCE1612 - Devesh Tibarewal

11 April 2022

Define Multivariate Data.

Multivariate data analysis is a type of statistical analysis that involves more than two dependent variables, resulting in a single outcome. Many problems in the world can be practical examples of multivariate equations as whatever happens in the world happens due to multiple reasons.

Example of multivariate data with a case study

All the datasets that contains more than 2 attributes can be considered as Multivariate data like the dataset used in my notebook below.

My Notebook

Importing Necessary Dependencies.

```
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
```

Loading and merging Dataset.

```
white_wine = pd.read_csv('winequality-white.csv', sep=';')
red_wine = pd.read_csv('winequality-red.csv', sep=';')

# store wine type as an attribute
red_wine['wine_type'] = 'red'
white_wine['wine_type'] = 'white'

# bucket wine quality scores into qualitative quality labels
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)
```

Understand dataset features and values.

```
wines.head()
```

```
##    fixed acidity  volatile acidity  ...  wine_type  quality_label
## 0            7.0              0.17  ...      white           high
## 1            7.7              0.64  ...        red            low
## 2            6.8              0.39  ...      white         medium
## 3            6.3              0.28  ...      white         medium
## 4            7.4              0.35  ...      white         medium
##
## [5 rows x 14 columns]
```

Exploratory Data Analysis and Visualizations

Descriptive Statistics

```
subset_attributes = ['residual sugar', 'total sulfur dioxide',
'sulphates', 'alcohol', 'volatile acidity', 'quality']
rs = round(red_wine[subset_attributes].describe(),2)
ws = round(white_wine[subset_attributes].describe(),2)
pd.concat([rs, ws], axis=1, keys=['Red Wine Statistics', 'White Wine
Statistics'])
```

```
##       Red Wine Statistics                      ... White Wine
Statistics
##            residual sugar total sulfur dioxide  ...      volatile
acidity  quality
## count             1599.00             1599.00  ...
4898.00  4898.00
## mean                 2.54               46.47  ...
0.28     5.88
## std                  1.41               32.90  ...
0.10     0.89
## min                  0.90                6.00  ...
0.08     3.00
## 25%                  1.90               22.00  ...
0.21     5.00
## 50%                  2.20               38.00  ...
0.26     6.00
## 75%                  2.60               62.00  ...
0.32     6.00
## max                 15.50              289.00  ...
1.10     9.00
```

```
##
## [8 rows x 12 columns]
```

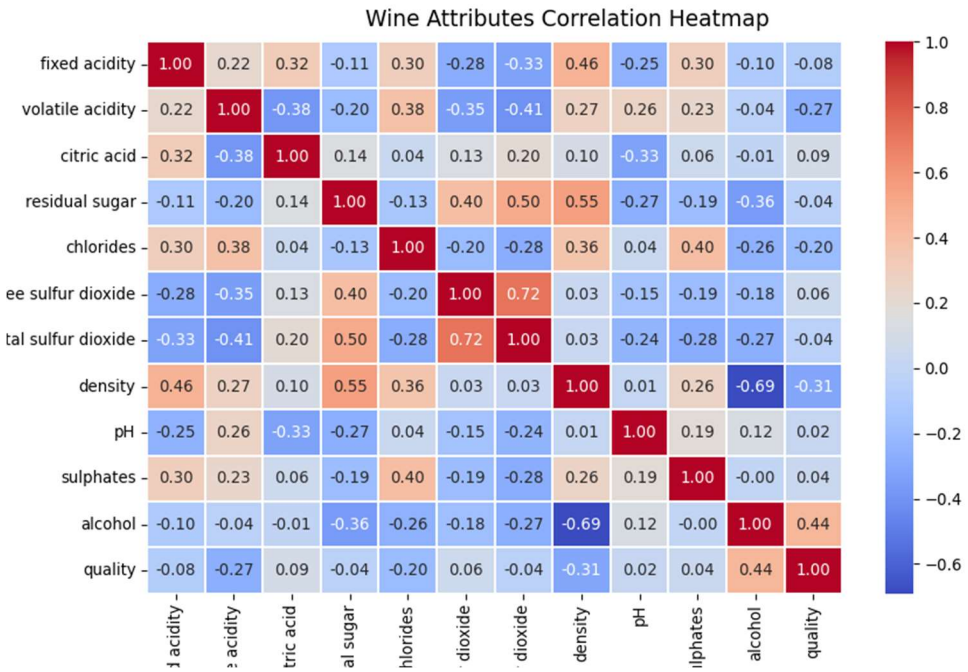
```
subset_attributes = ['alcohol', 'volatile acidity', 'pH', 'quality']
ls = round(wines[wines['quality_label'] ==
'low'][subset_attributes].describe(),2)
ms = round(wines[wines['quality_label'] ==
'medium'][subset_attributes].describe(),2)
hs = round(wines[wines['quality_label'] ==
'high'][subset_attributes].describe(),2)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

```
##       Low Quality Wine                    ... High Quality Wine
##                alcohol volatile acidity  ...                pH quality
## count          2384.00          2384.00  ...            198.00  198.00
## mean              9.87             0.40  ...              3.23    8.03
## std               0.84             0.19  ...              0.16    0.16
## min               8.00             0.10  ...              2.88    8.00
## 25%               9.30             0.26  ...              3.13    8.00
## 50%               9.60             0.34  ...              3.23    8.00
## 75%              10.40             0.50  ...              3.33    8.00
## max              14.90             1.58  ...              3.72    9.00
##
## [8 rows x 12 columns]
```

Multivariate Analysis

Visualizing two dimensions

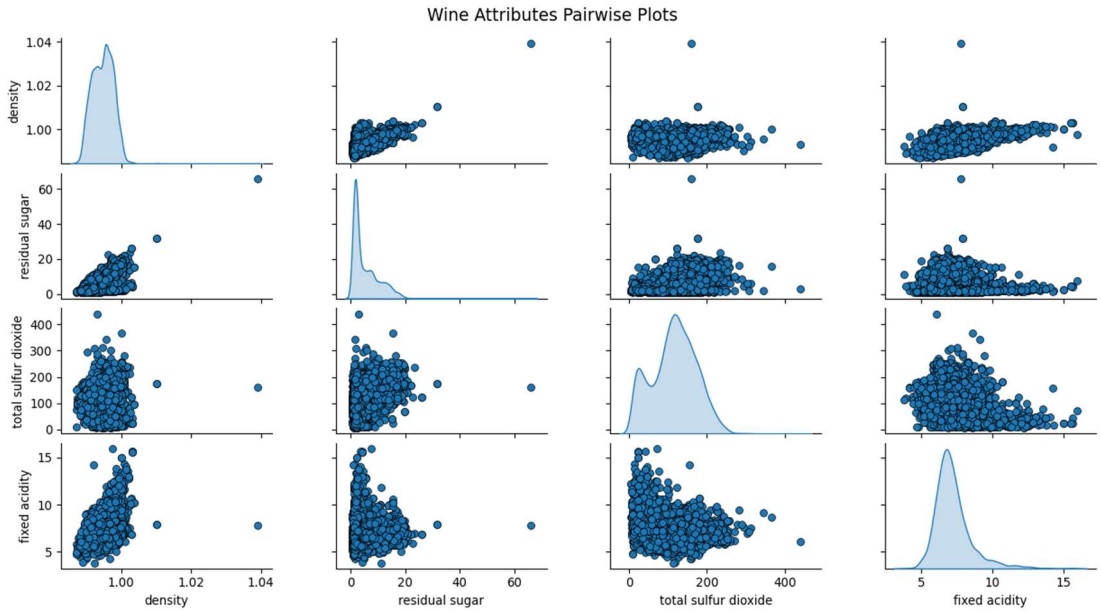
```
f, ax = plt.subplots(figsize=(10, 6))
corr = wines.corr()
hm = sns.heatmap(round(corr,2), annot=True, ax=ax,
cmap="coolwarm",fmt='.2f',
            linewidths=.05)
f.subplots_adjust(top=0.93)
t = f.suptitle('Wine Attributes Correlation Heatmap', fontsize=14)
plt.show()
```



Inference:

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.

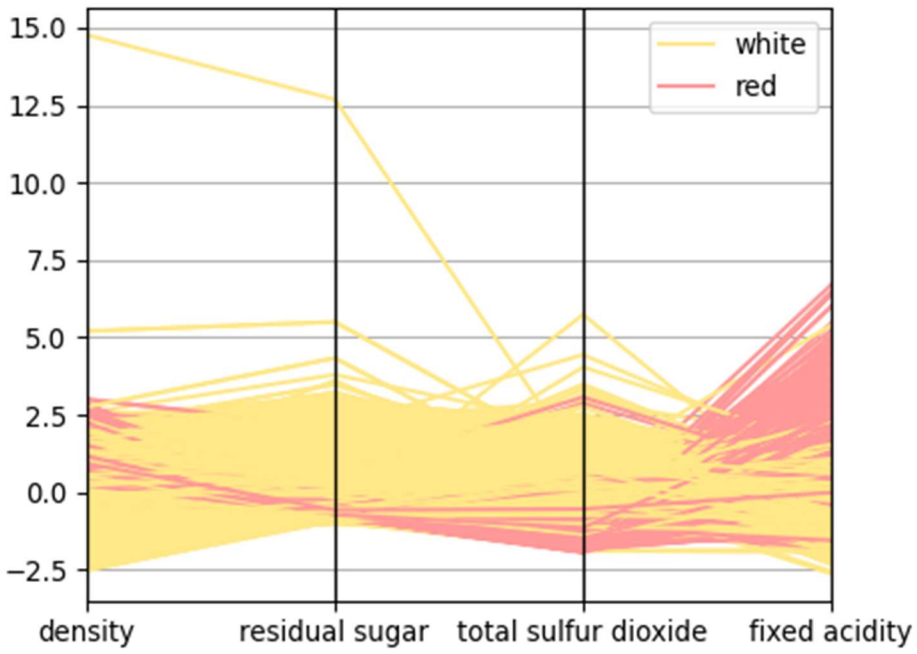
```
cols = ['density', 'residual sugar', 'total sulfur dioxide', 'fixed
acidity']
pp = sns.pairplot(wines[cols], height=1.8, aspect=1.8,
                  plot_kws=dict(edgecolor="k", linewidth=0.5),
                  diag_kind="kde", diag_kws=dict(shade=True))

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



Inference:

Based on the above plot, you can see that scatter plots are also a decent way of observing potential relationships or patterns in two-dimensions for data attributes.

```
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)
final_df.head()

##     density  residual sugar  total sulfur dioxide  fixed acidity
wine_type
## 0 -0.165631        1.546371              0.181456      -0.166089
white
## 1  0.301278       -0.681719              0.305311       0.373895
red
## 2 -0.859324        0.411306              0.305311      -0.320370
white
## 3  0.408001        1.210056              1.189993      -0.706073
white
## 4  1.395180        1.777588              2.003900       0.142473
white

from pandas.plotting import parallel_coordinates
f = plt.figure()
pc = parallel_coordinates(final_df, 'wine_type', color=('#FFE888',
'#FF9999'))
plt.show()
```

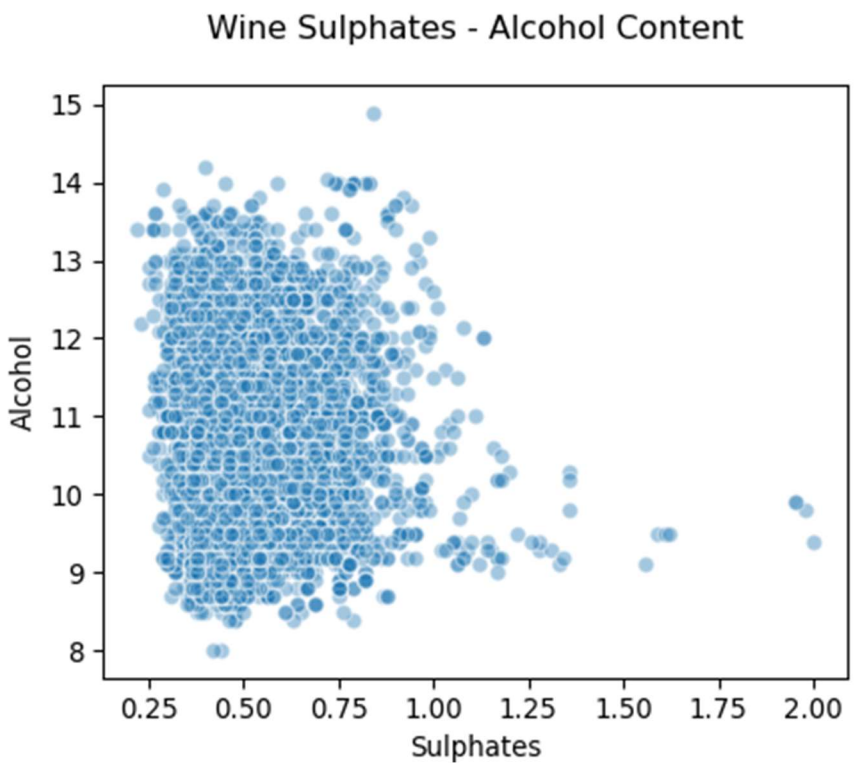


Inference:

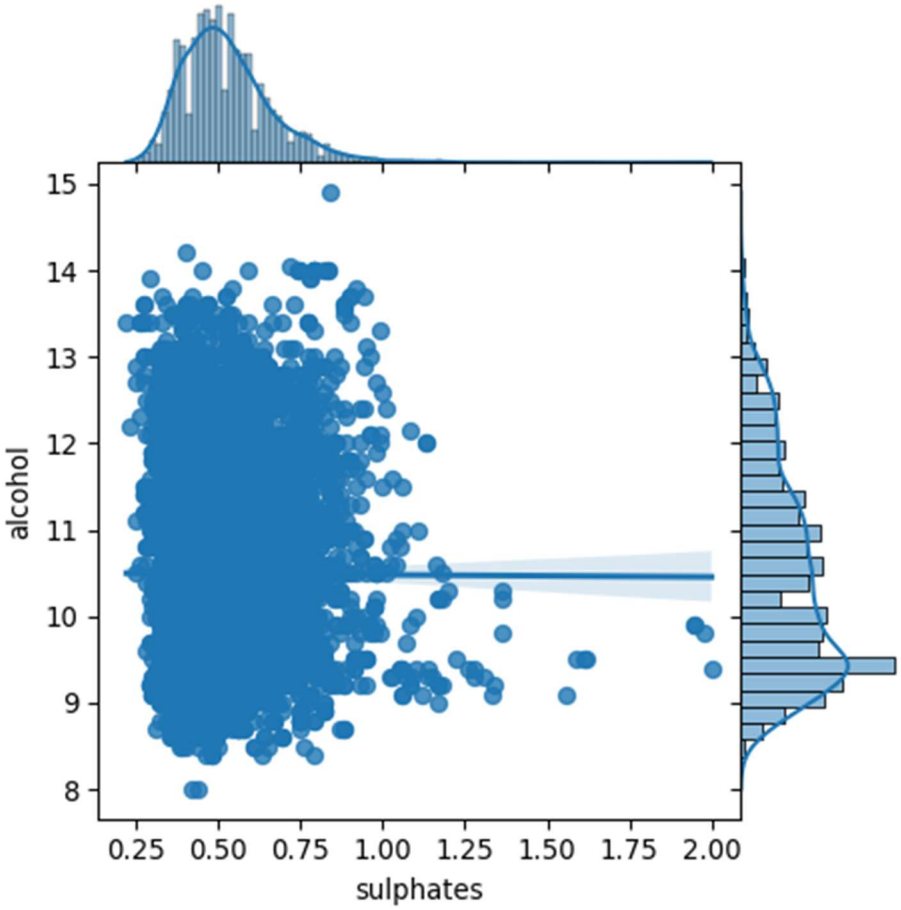
Basically, in this visualization as depicted above, points are represented as connected line segments. Each vertical line represents one data attribute. One complete set of connected line segments across all the attributes represents one data point. Hence points that tend to cluster will appear closer together. Just by looking at it, we can clearly see that density is slightly more for red wines as compared to white wines. Also residual sugar and total sulfur dioxide is higher for white wines as compared to red and fixed acidity is higher for red wines as compared to white wines.

Two Continuous Numeric attributes

```
f = plt.figure()
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)
plt.show()
```




```
f = plt.figure()
jp = sns.jointplot(x='sulphates', y='alcohol', data=wines, kind='reg',
space=0, height=5, ratio=4)
plt.show()
```



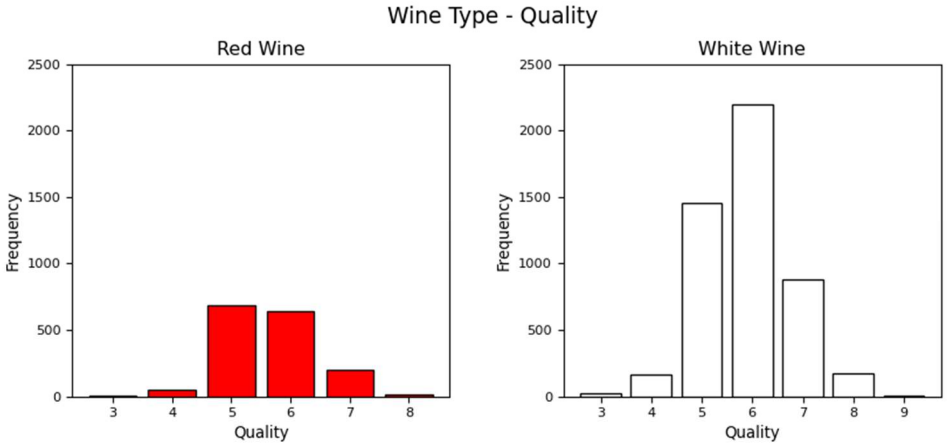
Inference:

The scatter plot is depicted on the left side and the joint plot on the right in the above figure. Like we mentioned, you can check out correlations, relationships as well as individual distributions in the joint plot.

Two Discrete Categorical attributes

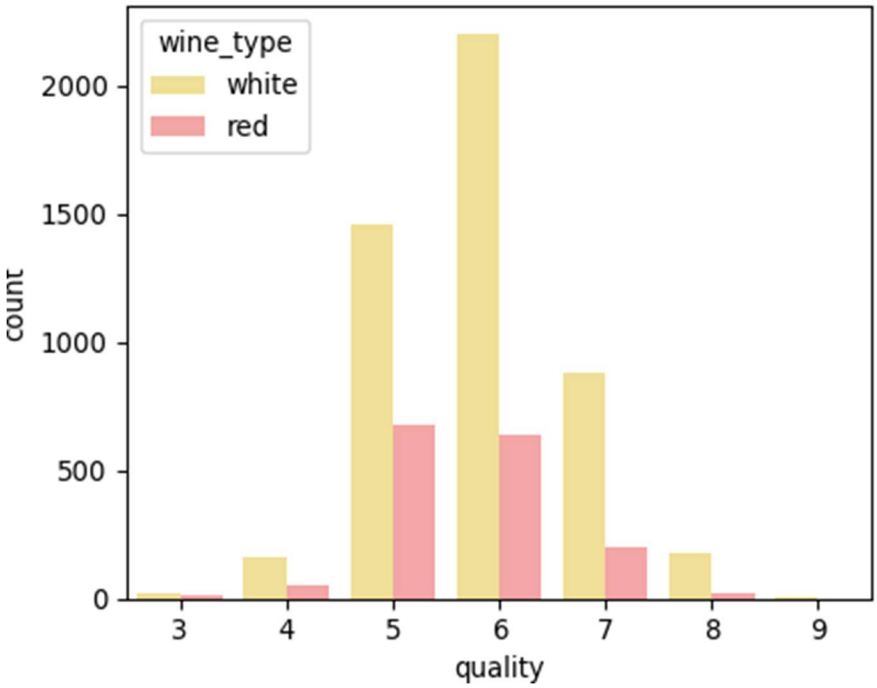
```
fig = plt.figure(figsize = (10, 4))
title = fig.suptitle("Wine Type - Quality", fontsize=14)
fig.subplots_adjust(top=0.85, wspace=0.3)

ax1 = fig.add_subplot(1,2, 1)
ax1.set_title("Red Wine")
ax1.set_xlabel("Quality")
ax1.set_ylabel("Frequency")
rw_q = red_wine['quality'].value_counts()
rw_q = (list(rw_q.index), list(rw_q.values))
ax1.set_ylim([0, 2500])

## (0.0, 2500.0)

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 = white_wine['quality'].value_counts()
ww_q = (list(ww_q.index), list(ww_q.values))
ax2.set_ylim([0, 2500])

## (0.0, 2500.0)

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)
plt.show()
```



While this is a good way to visualize categorical data, as you can see, leveraging matplotlib has resulted in writing a lot of code. Another good way is to use stacked bars or multiple bars for the different attributes in a single plot. We can leverage seaborn for the same easily.

```
f = plt.figure()
cp = sns.countplot(x="quality", hue="wine_type", data=wines,
palette={"red": "#FF9999", "white": "#FFE888"})
plt.show()
```

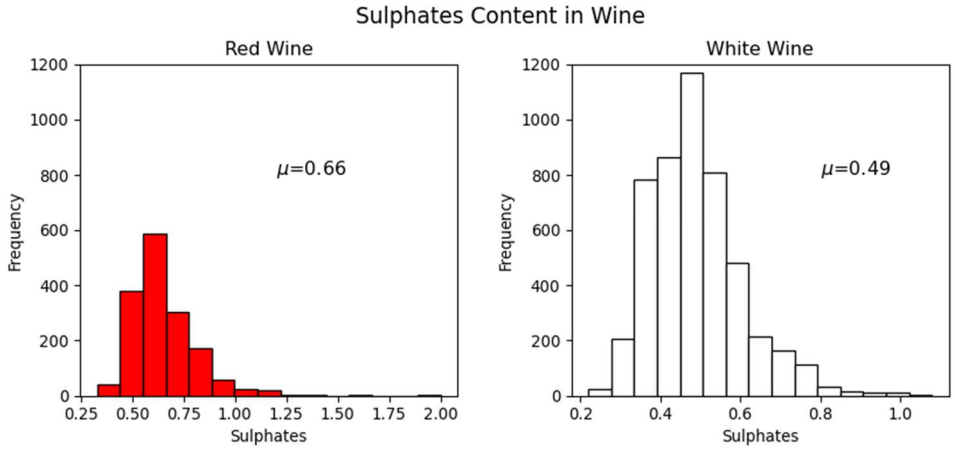


This definitely looks cleaner and you can also effectively compare the different categories easily from this single plot.

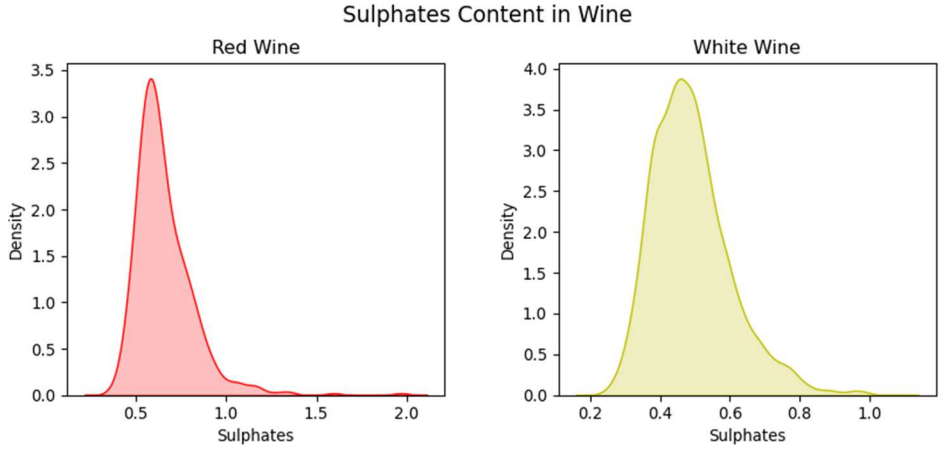
Mixed attributes (numeric & categorical)

```
fig = plt.figure(figsize = (10,4))
title = fig.suptitle("Sulphates Content in Wine", 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("Sulphates")
ax1.set_ylabel("Frequency")
ax1.set_ylim([0, 1200])

## (0.0, 1200.0)

ax1.text(1.2, 800, r'$\mu$='+str(round(red_wine['sulphates'].mean(),2)),
         fontsize=12)
r_freq, r_bins, r_patches = ax1.hist(red_wine['sulphates'], color='red',
bins=15,
                                     edgecolor='black', linewidth=1)

ax2 = fig.add_subplot(1,2, 2)
ax2.set_title("White Wine")
ax2.set_xlabel("Sulphates")
ax2.set_ylabel("Frequency")
ax2.set_ylim([0, 1200])

## (0.0, 1200.0)
```

```
ax2.text(0.8, 800, r'$\mu$='+str(round(white_wine['sulphates'].mean(),2)),
         fontsize=12)
w_freq, w_bins, w_patches = ax2.hist(white_wine['sulphates'],
color='white', bins=15,
                                     edgecolor='black', linewidth=1)
plt.show()
```

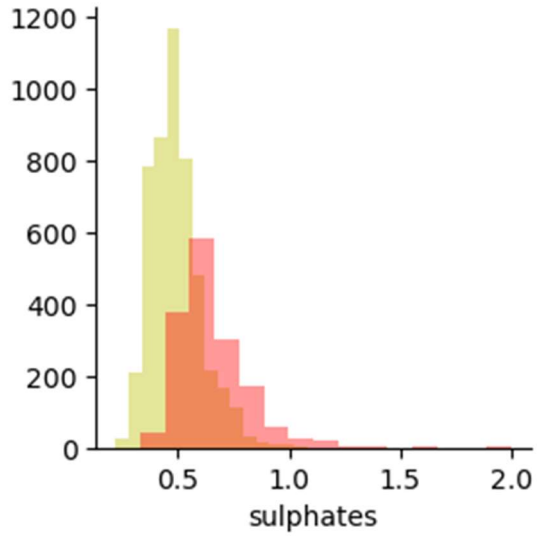



```
fig = plt.figure(figsize = (10, 4))
title = fig.suptitle("Sulphates Content in Wine", fontsize=14)
fig.subplots_adjust(top=0.85, wspace=0.3)

ax1 = fig.add_subplot(1,2, 1)
ax1.set_title("Red Wine")
ax1.set_xlabel("Sulphates")
ax1.set_ylabel("Density")
sns.kdeplot(red_wine['sulphates'], ax=ax1, shade=True, color='r')

ax2 = fig.add_subplot(1,2, 2)
ax2.set_title("White Wine")
ax2.set_xlabel("Sulphates")
ax2.set_ylabel("Density")
sns.kdeplot(white_wine['sulphates'], ax=ax2, shade=True, color='y')
plt.show()
```

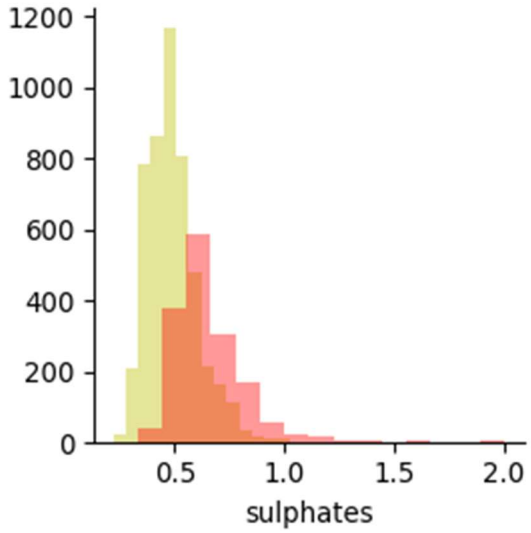



```
fig = plt.figure(figsize = (6, 4))
title = fig.suptitle("Sulphates Content in Wine", fontsize=14)
```

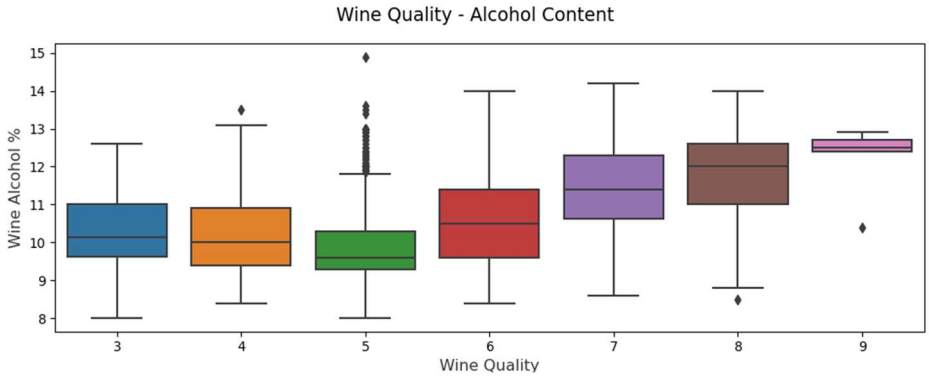
```
fig.subplots_adjust(top=0.85, wspace=0.3)
ax = fig.add_subplot(1,1, 1)
ax.set_xlabel("Sulphates")
ax.set_ylabel("Frequency")

g = sns.FacetGrid(wines, hue='wine_type', palette={"red": "r", "white":
"y"})
g.map(sns.distplot, 'sulphates', kde=False, bins=15, ax=ax)
```



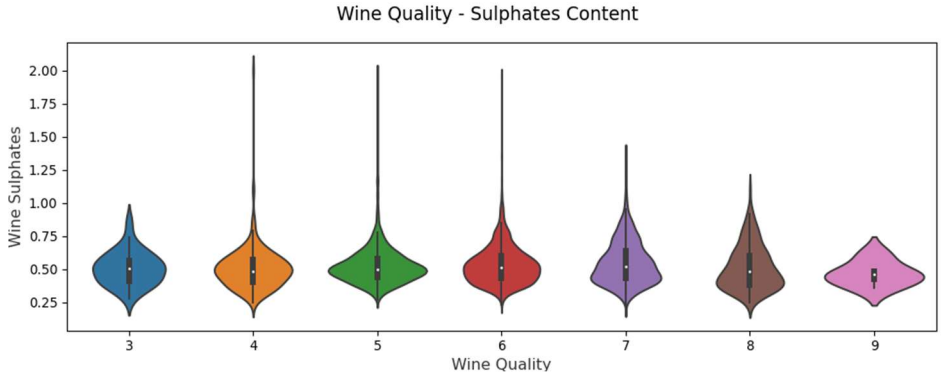

```
ax.legend(title='Wine Type')
plt.show()
```




```
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)
plt.show()
```




```
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)
plt.show()
```



Results

Above notebook are some types of graphs which are used to visualize multivariate dataset. The Notebook can be accessed this Github Link

Conclusion

This experiment enables us to understand and learn some effective strategies for visualizing data especially when the number of dimensions start to increase i.e. if we have a multivariate dataset.