Wine Quality: Determining the Physicochemical Effect

Kaitlin Onthank

May 9, 2014

This report and IPython notebook are available at http://www.github.com/konthank/Capstone.

Introduction

Overview

Wine-making, or viniculture, has been an art all over the world for thousands of years. A perfect balance of acids, sugars, and alcohol paired with the right grape can make for an exceptional glass of wine. Sounds simple enough, right? If you have ever toured a vineyard, it may appear easy: grow the grapes, crush them, put them in a barrel, then, enjoy! But, not so fast...

There are many factors that go into making wine and just as many considerations while tasting the wine. Because the focus of this report is on the characteristics and taste of completed wines, I will only quickly glaze over the complexity of the wine-making process. First, the grapes are harvested from the plants. These grapes are then separated into two groups; one for red wines and one for white wines. Next, the fermentation process begins where yeast may be added in order to convert sugars into alcohol. From here, there may be a secondary fermentation, a maturation period, or clarifications. Additionally, wines are blended with other wines to make the perfect combination. As you can see, there could be a few steps, or many steps. These steps could take a couple months, or several years! With all of the little tweaks that could be made in the process, there is no doubt every wine is unique.

When a wine is complete, it has several measurable physicochemical characteristics (acids, sugars, alcohol, etc.). I would like to explore the differences in these characteristics between red and white wines to see if they are significantly different. I would also like to determine whether these characteristics can be used to predict the quality of the wine. If so, wine-makers could better tailor their wine-making process to produce better quality wines. Finally, I want to see if it is possible to divide wines into 4 distinct categories (excellent, good, fair, and poor) by their qualities and physicochemical characteristics. To do this, we need relevant data...

Data

The data set consists of 1,599 red wines and 4,898 white wines from the Vinho Verde region in the northwest of Portugal (accessible at http://archive.ics.uci.edu/ml/datasets/Wine+Quality). These vineyards are located in a mild climate with similar conditions, harvest seasons, and processes as California wines (more specifically, Napa and Sonoma Valley wines). Because these data will be considered "typical" instances for this report, the results could possibly be generalized to all wines. Below is a list of the variables and their definitions.

Physicochemical Characteristics (Explanatory):

Fixed acidity (g/dm^3) - measure of tartaric acid, an indication of tartness
Volatile acidity (g/dm^3) - measure of acetic acid, an indication of spoilage
Citric acid (g/dm^3) - removes excess iron and copper from wine
Residual sugar (g/dm^3) - remains after fermentation or added to process
Chlorides (g/dm^3) - measure of sodium chloride, softens acidity
Free sulfur dioxide (mg/dm^3) - acts as germicidal and antioxidant
Total sulfur dioxide (mg/dm^3) - protects from spoilage and oxidation
Density (g/cm^3) - weighted measure of alcohol, sugar, and water pH (*0-14*) - measure of acid strength
Sulphates (g/dm^3) - stops fermentation
Alcohol ($vol.\%$) - results from fermentation process

Response:
Quality (*0-10*) - median of 3 wine-tasters' scores (0-very bad, 10-excellent)

Note: Other elements such as grape type, brand, or price were not available for privacy reasons.

Hypotheses

As a wine enthusiast, I am well aware the inconsistencies and uniqueness across wines may be an issue for modeling quality. However, through research and personal experience, I can make a few general hypotheses about the results:

- Red wines will be significantly different from white wines. For example,
  - White wines tend to have a higher fixed acidity
  - White wines tend to have a higher sugar content
  - Red wines tend to have a higher alcohol percentage
- Modeling will show some of these characteristics have an effect on quality
- Grouping into 4 distinct categories will work, but may result in overlap or poorly defined partitions

The following analysis attempts to reach the previously stated objectives and prove (or disprove) my hypotheses.

Analysis

Setup

In [1]:
```
# Import modules:

import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['axes.unicode_minus']=False
import numpy as np
import scipy as sp
import statsmodels.api as sm
from scipy import stats
from mpl_toolkits.mplot3d import Axes3D
import seaborn as sb
from scipy.cluster.vq import *
%pylab inline
```

In [2]:
```
# Pull csv files from URL:

!wget "http://archive.ics.uci.edu/ml/machine-learning-databases/wine- qual
```

```
!wget "http://archive.ics.uci.edu/ml/machine-learning-databases/wine- qual
```

In [3]:
```
# Bring downloaded data into notebook:

reds = pd.read_csv('winequality-red.csv', sep=";")
whites = pd.read_csv('winequality-white.csv', sep=";")
```

In [4]:
```
# Create design matrices and response vectors for ease and regression
purposes:

Xr = reds.ix[:, 0:11]
Yr = reds.ix[:, 11]
Xw = whites.ix[:,0:11]
Yw = whites.ix[:, 11]
```

Reds vs. Whites

The first objective is finding differences between red and white wines. To do this, we will begin by visually examining the distributions of each characteristic to see how they may differ. Figure 1, below, shows side-by-side boxplots of each of the 11 characteristics, plotting red wines adjacent to white wines.

In [6]:
```
# Set figure size:

a = pylab.rcParams['figure.figsize']
pylab.rcParams['figure.figsize'] = (8, 10.5)

# Boxplot for fixed acidity:
subplot(4,2,1)
sb.boxplot([Xr['fixed acidity'], Xw['fixed acidity']], names=["Reds","Whit
plt.title("Fixed Acidity ($g/dm^3$)", fontsize=18)
sb.set(style="whitegrid", context = "talk")
grid(b=None)
xticks(fontsize=15)
yticks(fontsize=15)


# Boxplot for volatile acidity:
subplot(4,2,2)
sb.boxplot([Xr['volatile acidity'], Xw['volatile acidity']], names=["Reds'
plt.title("Volatile Acidity ($g/dm^3$)", fontsize=18)
sb.set(style="whitegrid", context="talk")
grid(b=None)
xticks(fontsize=15)
yticks(fontsize=15)


# Boxplot for citric acid:
subplot(4,2,3)
sb.boxplot([Xr['citric acid'], Xw['citric acid']], names=["Reds","Whites"]
plt.title("Citric Acid ($g/dm^3$)", fontsize=18)
sb.set(style="whitegrid", context="talk")
grid(b=None)
xticks(fontsize=15)
yticks(fontsize=15)


# Boxplot for residual sugar:
subplot(4,2,4)
sb.boxplot([Xr['residual sugar'], Xw['residual sugar']], names=["Reds","Wh
plt.title("Residual Sugar ($g/dm^3$)", fontsize=18)
sb.set(style="whitegrid", context="talk")
```

```
grid(b=None)
xticks(fontsize=15)
yticks(fontsize=15)


# Boxplot for chlorides:
subplot(4,2,5)
sb.boxplot([Xr['chlorides'], Xw['chlorides']], names=["Reds","Whites"], co
plt.title("Chlorides ($g/dm^3$)", fontsize=18)
sb.set(style="whitegrid", context="talk")
grid(b=None)
xticks(fontsize=15)
yticks(fontsize=15)


# Boxplot for free sulfur dioxide:
subplot(4,2,6)
sb.boxplot([Xr['free sulfur dioxide'], Xw['free sulfur dioxide']], names=
plt.title("Free Sulfur Diox. ($mg/dm^3$)", fontsize=18)
sb.set(style="whitegrid", context="talk")
grid(b=None)
xticks(fontsize=15)
yticks(fontsize=15)


# Add caption:

subplot(4,2,7)
plt.gca().axison=False
text(0, .8, '''Figure 1. Side-by-side boxplots compare reds and whites in
characteristic.''', fontsize=12)

tight_layout()
subplots_adjust(hspace=.4)

# Reset figure size:

pylab.rcParams['figure.figsize'] = a
```

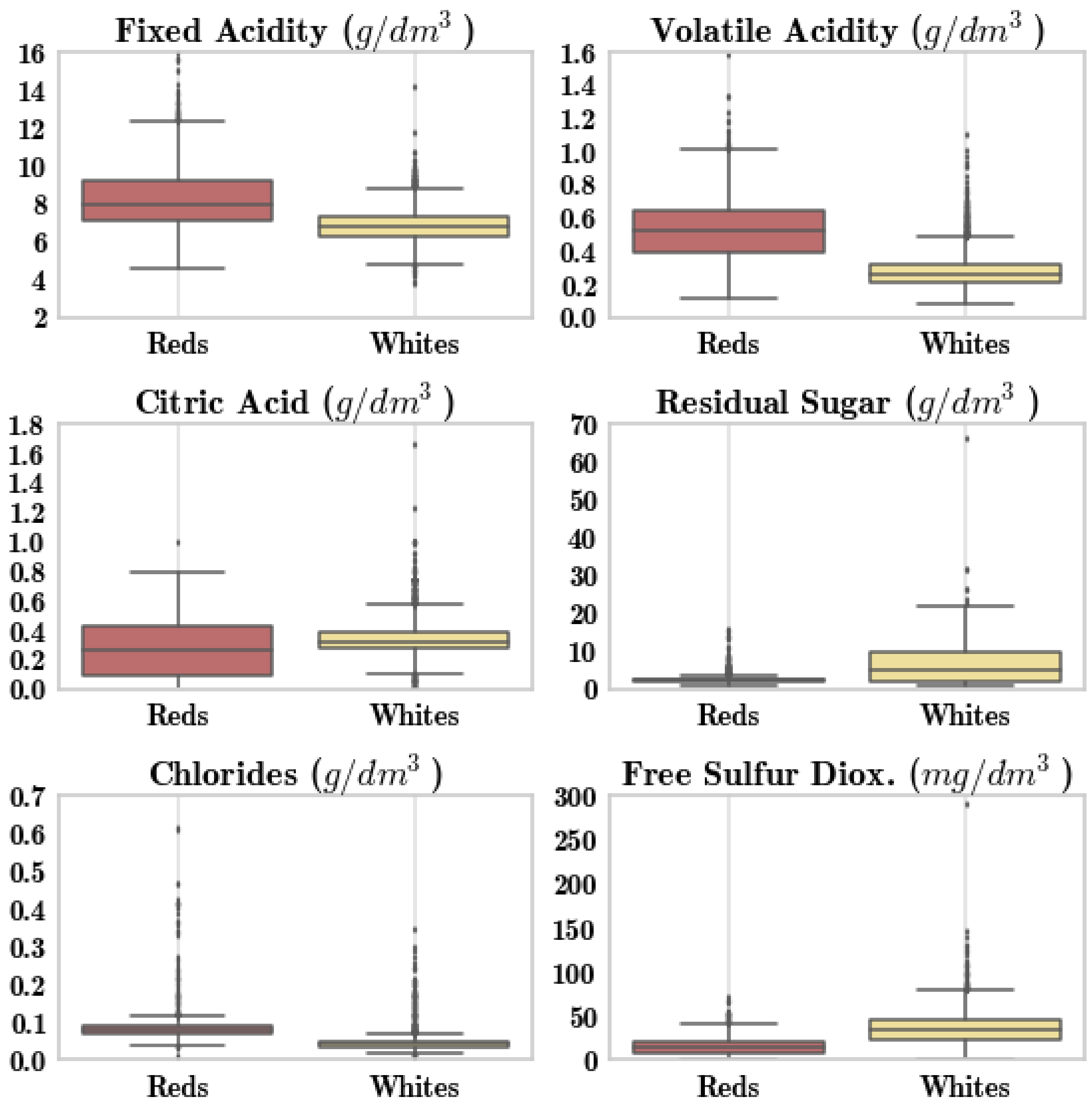


Figure 1. Side-by-side boxplots compare reds and whites in each physicochemical characteristic.

```
In [7]: # Set figure size:

        a = pylab.rcParams['figure.figsize']
        pylab.rcParams['figure.figsize'] = (8, 10.5)
```

```
# Boxplot for total sulfur dioxide:
subplot(4,2,1)
sb.boxplot([Xr['total sulfur dioxide'], Xw['total sulfur dioxide']], names
plt.title("Total Sulfur Diox. ($mg/dm^3$)", fontsize=18)
sb.set(style="whitegrid", context="talk")
grid(b=None)
xticks(fontsize=15)
yticks(fontsize=15)


# Boxplot for density:
subplot(4,2,2)
sb.boxplot([Xr['density'], Xw['density']], names=["Reds","Whites"], color=
plt.title("Density ($g/cm^3$)", fontsize=18)
sb.set(style="whitegrid", context="talk")
grid(b=None)
xticks(fontsize=15)
yticks(fontsize=15)


# Boxplot for pH:
subplot(4,2,3)
sb.boxplot([Xr['pH'], Xw['pH']], names=["Reds","Whites"], color=["indianre
plt.title("pH", fontsize=18)
sb.set(style="whitegrid", context="talk")
grid(b=None)
xticks(fontsize=15)
yticks(fontsize=15)


# Boxplot for sulphates:
subplot(4,2,4)
sb.boxplot([Xr['sulphates'], Xw['sulphates']], names=["Reds","Whites"], co
plt.title("Sulphates ($g/dm^3$)", fontsize=18)
sb.set(style="whitegrid", context="talk")
grid(b=None)
xticks(fontsize=15)
yticks(fontsize=15)


# Boxplot for alcohol:
subplot(4,2,5)
sb.boxplot([Xr['alcohol'], Xw['alcohol']], names=["Reds","Whites"], color=
plt.title("Alcohol (%)", fontsize=18)
sb.set(style="whitegrid", context="talk")
grid(b=None)
xticks(fontsize=15)
yticks(fontsize=15)


# Add caption:

subplot(4,2,7)
plt.gca().axison=False
text(0, .8, '''Figure 1 cont. Side-by-side boxplots compare reds and white
characteristic.''', fontsize=12)

tight_layout()
subplots_adjust(hspace=.4)


# Reset figure size:

pylab.rcParams['figure.figsize'] = a
```

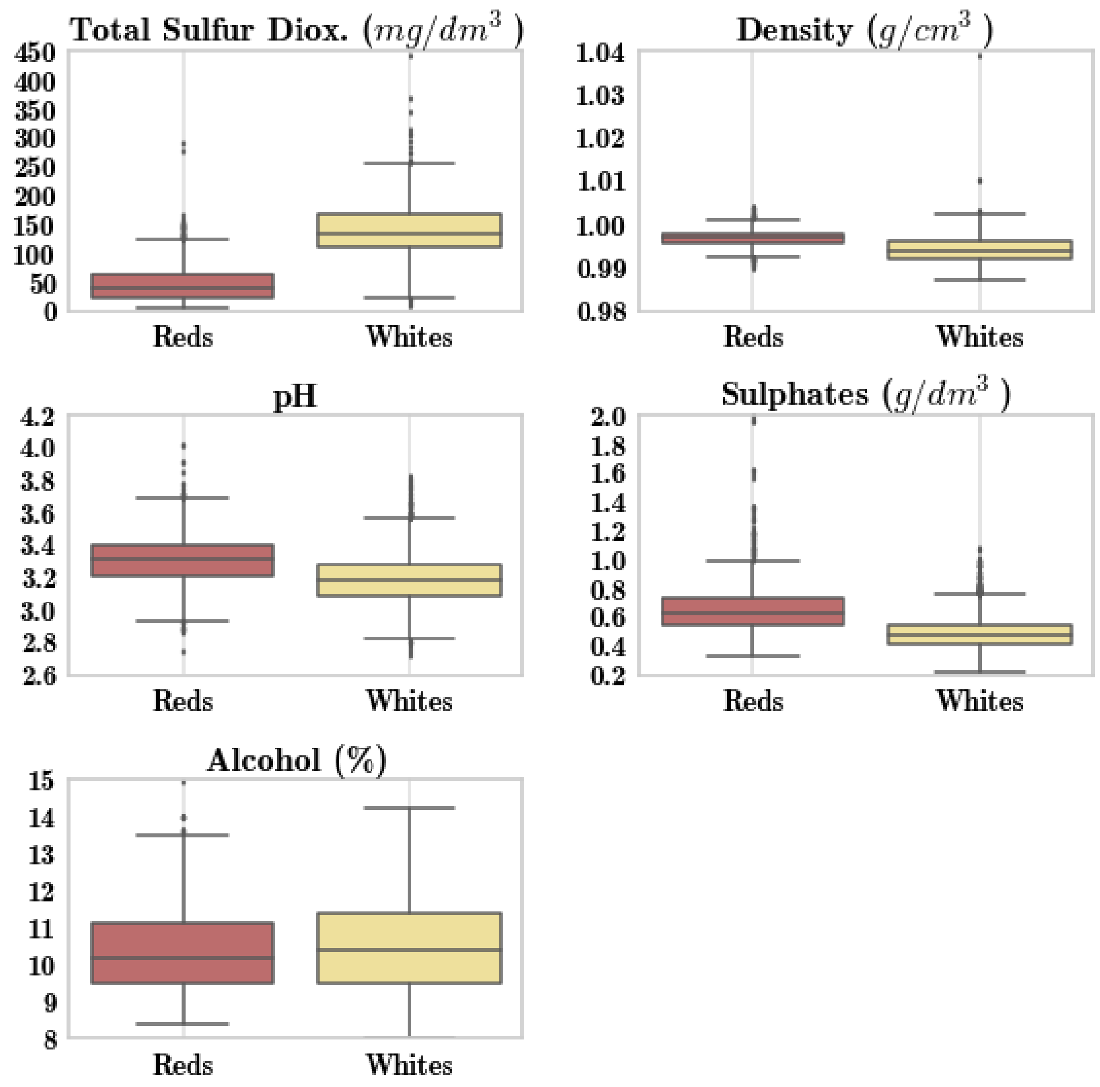


Figure 1 cont. Side-by-side boxplots compare reds and whites in each physicochemical characteristic.

As you can see, many of the boxplots show an obvious contrast between red and white wines.

- For **fixed acidity**, the red wine average appears to clearly dominate the white wine average, indicating red wines are generally more tart. This definitely comes as a surprise given resources that suggested otherwise. However, it is important to look at the distribution as well, noting the reds' wide range, the numerous outliers in both colors, and the similar minimums.
- Red wines also appear to have a much higher average **volatile acidity** than white wines. Again, there is quite a

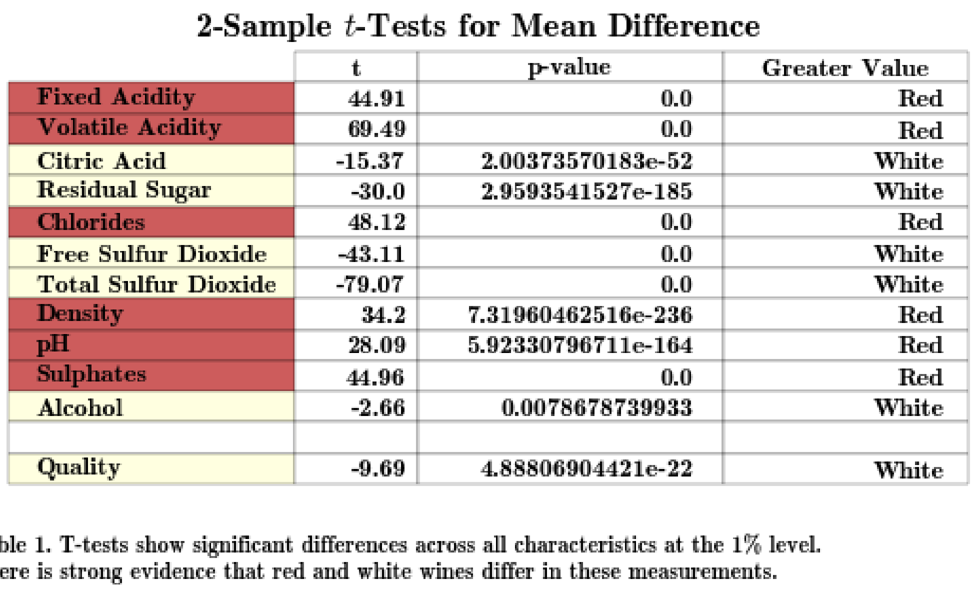
bit of overlap and further analysis will determine significance. Note the very low values here and remember that this characteristic measures spoilage, which is known to be more likely for a red wine.

- **Citric acid** averages appear similar, but the disparity in the quartiles and outliers is clear.
- The boxplots of **residual sugars** support my original hypothesis and display the truly small smount of sugars present in red wines compared to white wines. A seemingly uncharacteristic outlier of a white wine with residual sugars measuring $\sim 65\ g/dm^3$ skews the plot immensely and may need to be further addressed.
- An examination of **chlorides** is not very descriptive in its high number of outliers and miniscule interquartile range, but whites do appear to have a slightly smaller amount of sodium chloride than reds.
- White wines clearly dominate red wines in both **free sulfur dioxide** and **total sulfur dioxide**. This may seem intuitive given the latter is a function of the former. Why white wines dominate reds may be an indication that white wines have a greater need for antibacterial components.
- It appears red wines are slightly more dense than white wines. This is curious, given **density** is a function of alcohol, water, and sugars; two of which appear to be greater in white wines.
- The distributions for **pH** are very similar, with red wines shifted up about 0.1. It makes sense that red wines would have a higher pH because it is a measure of acidity.
- Red wines are showing a higher average **sulphates** measure than white wines, but an overlap suggests there are many white wines that are greater than many of the red wines.
- Finally, red wines and white wines are similar in **alcohol** percentage, with only a slight lead going to the whites.

A more concrete way to determine if the characteristics are different is through the use of 2-sample t-tests for significant difference in means. These tests will take deviations into account, which could not be easily considered in Figure 1. Table 1 provides each physicochemical characteristic, the t-statistic when testing $H_0 : \mu_{Red} - \mu_{White} = 0$ vs. $H_1 : \mu_{Red} - \mu_{White} \neq 0$, the corresponding p-value, and a summary of which wine type has a greater value in that category. The response variable, quality, has also been tested, included in the table, and will be analyzed later on.

In [8]:
```
# 2-sample t-tests for significant difference in means:

t1, p1 = stats.ttest_ind(reds['fixed acidity'], whites['fixed acidity'])
t2, p2 = stats.ttest_ind(reds['volatile acidity'], whites['volatile acidit
t3, p3 = stats.ttest_ind(reds['citric acid'], whites['citric acid'])
t4, p4 = stats.ttest_ind(reds['residual sugar'], whites['residual sugar'])
t5, p5 = stats.ttest_ind(reds['chlorides'], whites['chlorides'])
t6, p6 = stats.ttest_ind(reds['free sulfur dioxide'], whites['free sulfur
t7, p7 = stats.ttest_ind(reds['total sulfur dioxide'], whites['total sulfu
t8, p8 = stats.ttest_ind(reds['density'], whites['density'])
t9, p9 = stats.ttest_ind(reds['pH'], whites['pH'])
t10, p10 = stats.ttest_ind(reds['sulphates'], whites['sulphates'])
t11, p11 = stats.ttest_ind(reds['alcohol'], whites['alcohol'])
t12, p12 = stats.ttest_ind(reds['quality'], whites['quality'])


# Set figure size:

a = pylab.rcParams['figure.figsize']
pylab.rcParams['figure.figsize'] = (8.2, 6.0)


# Create table of variables, t-statistics, and p-values:

collabels = ['t', 'p-value', 'Greater Value']
rowlabels = ['Fixed Acidity','Volatile Acidity', 'Citric Acid', 'Residual
             'Total Sulfur Dioxide','Density','pH','Sulphates','Alcohol','
tablevals = [[round(t1,2),p1,'Red'],[round(t2,2),p2,'Red'],[round(t3,2),p
             [round(t4,2),p4,'White'],[round(t5,2),round(p5,4),'Red'],[rou
             [round(t7,2),p7,'White'],[round(t8,2),p8,'Red'],[round(t9,2),
             [round(t10,2),p10,'Red'],[round(t11,2),p11,'White'],['','',''
```

```
rowColours = ['indianred','indianred', 'lightyellow','lightyellow','indiar
              'indianred','lightyellow','white','lightyellow']

subplot(2,1,1)
plt.gca().axison=False
sb.set(style="whitegrid", context="talk")
the_table = plt.table(cellText=tablevals, colWidths=[.1,.25,.2],rowColours
                      colLabels=collabels, loc="upper right")
the_table.auto_set_font_size(False)
the_table.set_fontsize(14)
the_table.scale(1.2,1.1)
title("2-Sample $t$-Tests for Mean Difference", fontsize=18, ha='center')


# Add caption:

subplot(2,1,2)
plt.gca().axison=False
text(0, .5, '''Table 1. T-tests show significant differences across all ch
There is strong evidence that red and white wines differ in these measurer

tight_layout()
subplots_adjust(hspace=0)

# Reset figure size:

pylab.rcParams['figure.figsize'] = a
```



2-Sample $t$-Tests for Mean Difference

	t	p-value	Greater Value
Fixed Acidity	44.91	0.0	Red
Volatile Acidity	69.49	0.0	Red
Citric Acid	-15.37	2.00373570183e-52	White
Residual Sugar	-30.0	2.9593541527e-185	White
Chlorides	48.12	0.0	Red
Free Sulfur Dioxide	-43.11	0.0	White
Total Sulfur Dioxide	-79.07	0.0	White
Density	34.2	7.31960462516e-236	Red
pH	28.09	5.92330796711e-164	Red
Sulphates	44.96	0.0	Red
Alcohol	-2.66	0.0078678739933	White
Quality	-9.69	4.88806904421e-22	White

Table 1. T-tests show significant differences across all characteristics at the 1% level.
There is strong evidence that red and white wines differ in these measurements.

Table 1 shows that all of the mean differences in characteristics are significant at the 1% level. This confirms the boxplots in Figure 1 do indeed show true disparities in the averages. This also definitively disproves the hypotheses that white wines would have a higher fixed acidity than red wines, and red wines would have a higher alcohol percentage than white wines. My inkling about white wines having higher residual sugars than red wines was confirmed. Now, we will explore possible correlations among these characteristics.

Correlating Variables

Before attempting to use physicochemical characteristics to predict quality, it is important to check whether any of the explanatory variables are affecting one another. This multicollinearity could pose a problem during a regression analysis and should be addressed prior to any modeling. First, we will look at a correlation matrix for the red wine characteristics. It can be speculated that there may be possible correlations between free sulfur dioxide and total sulfur dioxide, or density and residual sugars.

```
In [17]: # Set figure size:

         a = pylab.rcParams['figure.figsize']
         pylab.rcParams['figure.figsize'] = (4.5, 5.0)


         # Correlogram displaying correlations between red wine characteristics:

         subplot(2,1,1)
         cmap = sb.blend_palette(["#0060c7","#70b5ff", "#d6eaff", "#cdfed5", "#21fc
         sb.corrplot(Xr, annot=False, diag_names=False, cbar=True, cmap=cmap)
         plt.title("Red Correlogram", fontsize=18)
         sb.set(style="whitegrid", context="talk")
         xticks(fontsize=11, rotation=70, ha='right')
         yticks(fontsize=11)


         # Add caption:

         subplot(2,1,2)
         plt.gca().axison=False
         text(0, 0,'''Figure 2. A visual correlation matrix shows possible
         issues with correlated explanatory variables for red
         wines.''', fontsize=12)


         # Reset figure size:

         pylab.rcParams['figure.figsize'] = a
```

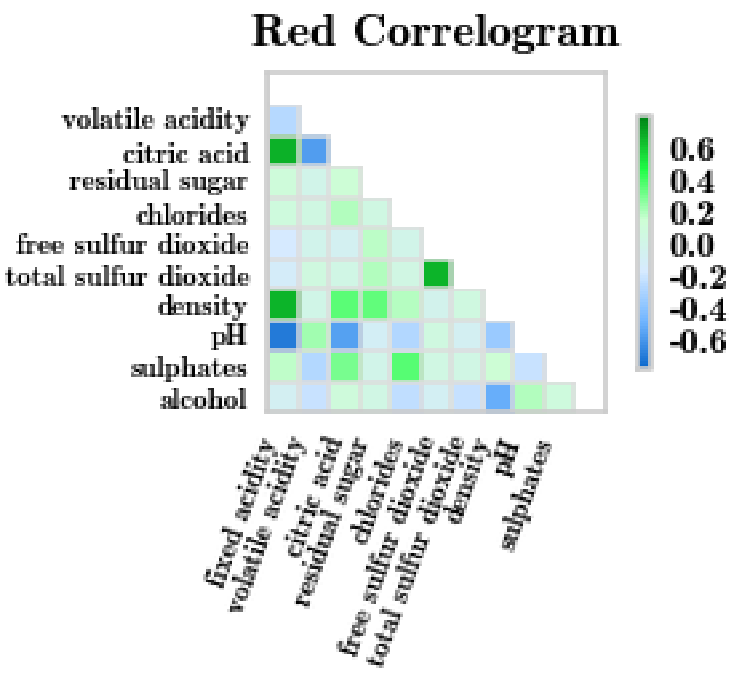


Figure 2. A visual correlation matrix shows possible issues with correlated explanatory variables for red wines.

By examining the darkest squares in the matrix in Figure 2, a few possible issues arise. There seems to be somewhat strong positive correlations between citric acid & fixed acidity, density & fixed acidity, and total sulfur dioxide & free sulfur dioxide. The first and third pairs are not surprising, as they are measuring similar things. There also appear to be somewhat strong negative correlations between citric acid & volatile acidity, and pH & fixed acidity. Again, these were expected.

In [18]:
```
# Set figure size:

a = pylab.rcParams['figure.figsize']
pylab.rcParams['figure.figsize'] = (4.5, 5.0)


# Correlogram displaying correlations between white wine characteristics:
```

```
subplot(2,1,1)
cmap = sb.blend_palette(["#0060c7","#70b5ff", "#d6eaff", "#cdfed5", "#21fc
sb.corrplot(Xw, annot=False, diag_names=False, cbar=True, cmap=cmap)
plt.title("White Correlogram", fontsize=18)
sb.set(style="whitegrid", context="talk")
xticks(fontsize=11, rotation=70, ha='right')
yticks(fontsize=11)


# Add caption:

subplot(2,1,2)
plt.gca().axison=False
text(0,0,'''Figure 3. A visual correlation matrix shows possible
issues with correlated explanatory variables for white
wines.''', fontsize=12)


# Reset figure size:

pylab.rcParams['figure.figsize'] = a
```

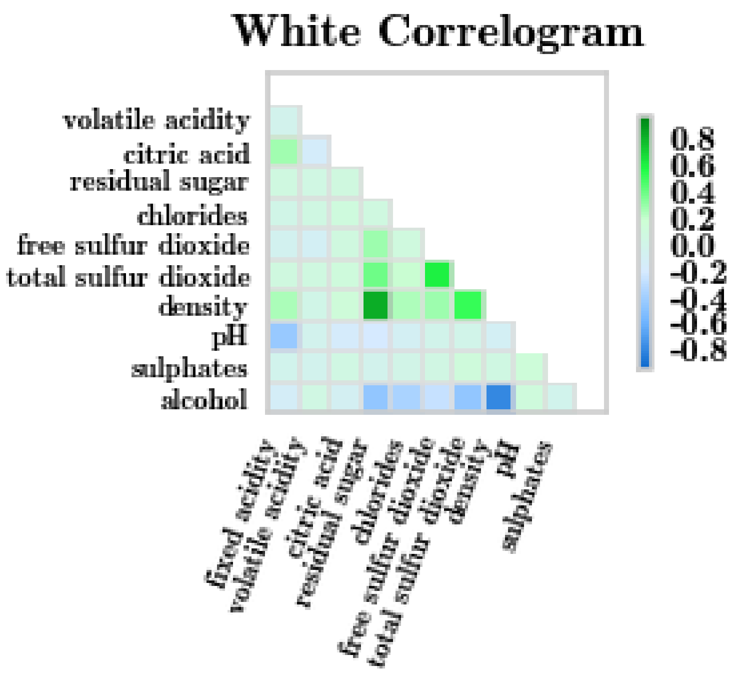


Figure 3. A visual correlation matrix shows possible issues with correlated explanatory variables for white wines.

In Figure 3, a strong positive correlation is present between density and residual sugar, while a strong negative correlation is seen between density and alcohol. Other possible, yet milder, problems could occur between the two sulfur dioxides and between density and total sulfur dioxide. Steps may need to be taken in the modeling process to handle all or some of these collinearities.

Quality Spread

Finally, the remaining variable, quality, must be studied before fitting a model. From Table 1, we can see there is a significant difference between the quality of these red and white wines. Additionally, we know there are more than 3 times as many white wine observations as there are for red wines. Knowing these differences, we should check each distribution to get a rough idea of the numbers we are working with.

In [19]:
```
# Set figure size:

a = pylab.rcParams['figure.figsize']
#pylab.rcParams['figure.figsize'] = (10.0, 7.0)
pylab.rcParams['figure.figsize'] = (8.2, 5.5)

# Histogram of red wine qualities:

subplot(2,2,1)
redhist = plt.hist(Yr, 6, color="indianred")
plt.title("Red Wine Quality",fontsize=18)
sb.axlabel("Score", "", fontsize=15)
sb.set(style="whitegrid", context = "talk")
grid(b=None)
xlim(xmin=0,xmax=10)
ylim(ymax=2500)
xticks(fontsize=15)
yticks(fontsize=15)


# Histogram of white wine qualities:

subplot(2,2,2)
plt.hist(Yw, 7, color='#FFEC8B', align='mid')
plt.title("White Wine Quality", fontsize=18)
sb.axlabel("Score", "", fontsize=15)
sb.set(style="whitegrid", context = "talk")
grid(b=None)
xlim(xmin=0,xmax=10)
xticks(fontsize=15)
yticks(fontsize=15)


# Add caption:

subplot(2,2,3)
plt.gca().axison=False
text(0, .6, '''Figure 4. These histograms display the distributions of the
of the 1,599 red wines and 4,898 white wines. Scores are between 3 and 9 w
very few 'very bad' or 'excellent' wines.''', fontsize=12)

tight_layout()


# Reset figure size:

pylab.rcParams['figure.figsize'] = a
```

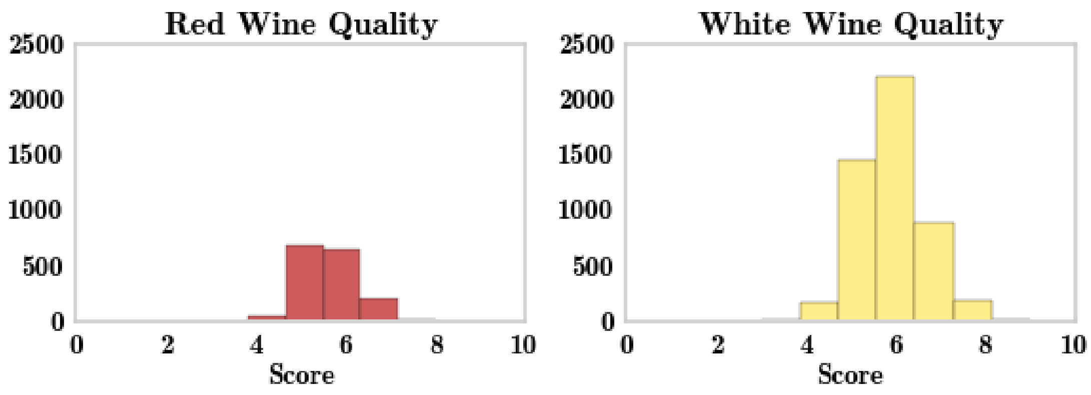


Figure 4. These histograms display the distributions of the response variable, quality, of the 1,599 red wines and 4,898 white wines. Scores are between 3 and 9 with very few 'very bad' or 'excellent' wines.

Through Figure 2, we can visually compare the two distributions. Both have obvious peaks in the middle at scores 5 and 6. Tapering down each side makes for several 4, 7, and 8 scores, and very few 0, 1, 2, 3, 9, or 10 scores. While sample size and means proved dissimilar, the overall spread of quality is similar between the two types. This shows that even with two numerically unique sets of characteristics per color, red and white wine quality scores can both fall in the same neighborhood.

Regression

After exploring all variables and determining differences between red and white wines, we can now attempt to predict quality using the characteristics. A regression analysis can provide us with details on how well the model will predict as well as which characteristics are increasing (or decreasing) the quality score. Beginning with ordinary least squares (OLS), we include all of the variables in the model, including an intercept term. One by one, insignificant variables (at the 5% level) are removed through a backwards selection.

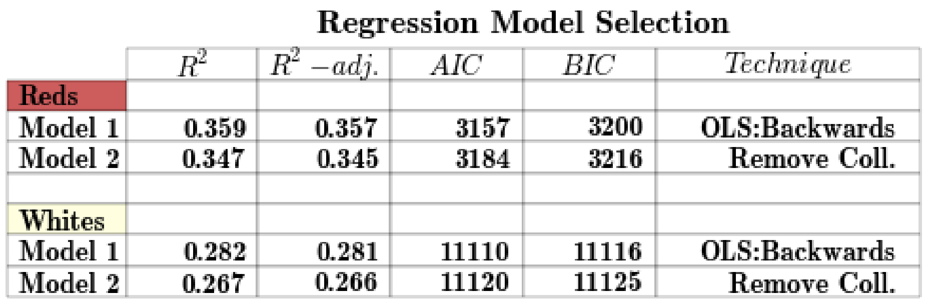
For the red wines OLS, only density, fixed acidity, residual sugars, and citric acid were removed from the model. This leaves quite a few of the characteristics in the model. Additionally, model-fit statistics (R^2, $R^2 - adj.$, AIC and BIC) were unsatisfactory (see Table 2). In an attempt to remove potential multicollinearity, free sulfur dioxide and chlorides were removed from the model in a second OLS. However, these attempts did not improve the model.

For the white wines OLS, only citric acid, chlorides, and total sulfur dioxide were removed from the model. Model-fit statistics were, once again, unsatisfactory (see Table 2). To remove potential multicollinearity, free sulfur dioxide was removed from the model in a second OLS. These attempts proved futile again.

Table 2 provides a summary of the four models. For full regression output, see Appendix.

In [21]:
```
# Set figure size:

a = pylab.rcParams['figure.figsize']
#pylab.rcParams['figure.figsize'] = (20.0, 6.5)
pylab.rcParams['figure.figsize'] = (8.2, 6)
```

```
# Create table with models and statistics:

collabels = ['$R^2$', '$R^2-adj.$', '$AIC$', '$BIC$','$Technique$']
rowlabels = ['Reds', 'Model 1', 'Model 2', ' ', 'Whites', 'Model 1', 'Mode
tablevals = [[' ',' ',' ',' ',' '],[.359,.357,3157,3200,'OLS:Backwards'],
              [' ',' ',' ',' ',' '],[' ',' ',' ',' ',' '],[.282,.281,11110,
              [.267,.266,11120,11125,'Remove Coll.']]
rowColours = ['indianred','white','white','white','lightyellow','white','w

models = plt.figure()
models.add_subplot(211, frameon=False)
plt.gca().axison=False
sb.set(style="whitegrid", context="talk")
models = plt.table(cellText=tablevals, rowLabels=rowlabels, rowColours=row
                   colLabels=collabels, loc='upper center', colWidths=[.1,
models.auto_set_font_size(False)
models.set_fontsize(15)
models.scale(1.3,1.1)
title("Regression Model Selection", fontsize=18, ha='center')


# Add caption:

subplot(2,1,2)
plt.gca().axison=False
text(0,.75,'''Table 2. Quality for red and white wines was modeled using O
selection. The same technique was used again after removing possible corre
variables identified in the previous correlograms. Results were unsatisfac
did not directly support hypotheses. Full OLS Regression results are provi
the appendix.''', fontsize=12)

tight_layout()
subplots_adjust(hspace=0)


# Reset figure size:

pylab.rcParams['figure.figsize'] = a
```



Regression Model Selection

	R^2	R^2 $-adj.$	AIC	BIC	$Technique$
Reds					
Model 1	0.359	0.357	3157	3200	OLS:Backwards
Model 2	0.347	0.345	3184	3216	Remove Coll.
Whites					
Model 1	0.282	0.281	11110	11116	OLS:Backwards
Model 2	0.267	0.266	11120	11125	Remove Coll.

Table 2. Quality for red and white wines was modeled using OLS with backwards selection. The same technique was used again after removing possible correlating variables identified in the previous correlograms. Results were unsatisfactory and did not directly support hypotheses. Full OLS Regression results are provided in the appendix.

One purpose of this report is to see whether quality can be modeled from a wine's physicochemical characteristics. From what we see in Table 2 above, the modeling is not completely useless but will be considered mediocre. Other pursuits to find a good model included a multinomial logit regression and a k-nearest-neighbors regression. Both were unsuccessful and not up to the standards for this report and the hypotheses.

Upon further pondering of this problem, it became apparent that the inability to model quality may be obvious. While all of these wines originate from the same region, they are separate observations for a reason: they are unique! The characteristics of a single red wine can vary dramatically when compared to another red wine, but both may taste great. A white wine with high sugar content may get a quality score identical to a white wine with low sugar content. The wine-making process is so complex and unique from wine to wine; maybe we should not expect to be able to predict how good it will be. Subsequently, our response variable is based upon the taste preferences of a human being, which are commonly known to be dramatically different from human to human. Pairing objective measures of chemicals with subjective measures of taste is a shaky road to go down. However, we still have one more objective to cover…

Classification

The next approach of this analysis is to use a clustering algorithm to sort the wines into 4 groups: excellent, good, fair, and poor. With a scale from 0 to 10, the quality variable may have too big a range. Who can really discern the difference between a 4 and a 5? Four groups is much easier to analyze and interpret for our purposes.

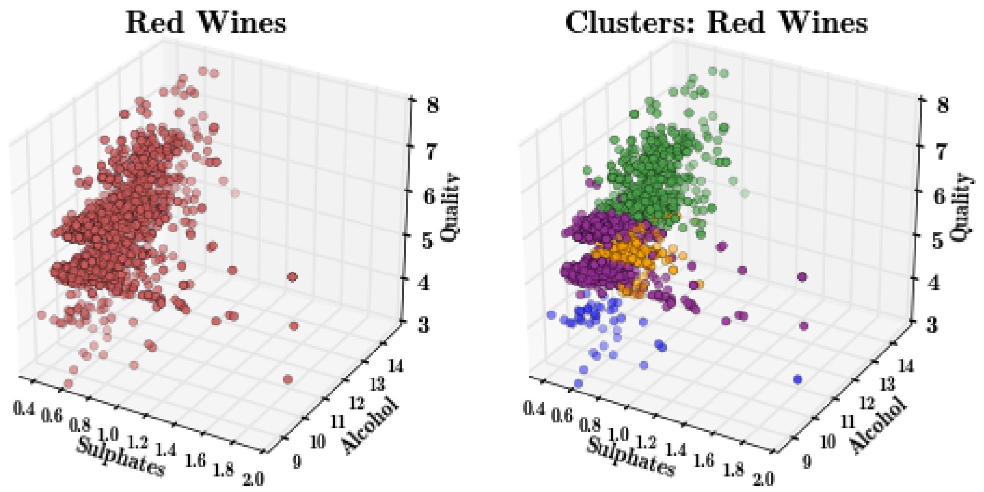
On another note, the composition of these wines is also difficult to interpret, especially when we are taking 11 characteristics into account. The goal here is to tailor the analysis not only in the quality variable, but in the others as well.

As a solution, we will choose two characteristics per wine type to aid in the clustering process. This way, we will have descriptive statistics on 3 aspects of each grouping. But, how to choose these two characteristics?

For the red wines, I was particularly interested in elements that made a positive contribution to quality. Considering previously obtained regression coefficients, input from resources, and personal intrigue, sulphates and alcohol percentage were chosen. In the same manner, pH and sulphates were chosen for the white wines.

In [25]:
```python
# Set figure size:

a = pylab.rcParams['figure.figsize']
pylab.rcParams['figure.figsize'] = (8.35, 8.35)


# 3D scatter plot of red wines:

pylab.close()
figR = plt.figure()
axR = figR.add_subplot(221, projection='3d')
axR.scatter(xs=Xr['sulphates'], ys=Xr['alcohol'], zs=Yr, c='indianred')
axR.set_xlabel('Sulphates', fontsize=12)
axR.set_ylabel('Alcohol', fontsize=12)
axR.set_zlabel('Quality', fontsize=12)
xticks(fontsize=10)
yticks(fontsize=10)
title("Red Wines", fontsize=18)


# 3D scatter plot of clusters:

axRC = figR.add_subplot(222, projection='3d')
resR, idxR = kmeans2(np.column_stack([Xr['sulphates'], Xr['alcohol'], whit
                     iter=100)
colorsR = ([('#993299','#3232ff','#4ca64c','#ffa500')[i] for i in idxR])
axRC.scatter(Xr['sulphates'], Xr['alcohol'], Yr, c=colorsR)
axRC.set_xlabel('Sulphates', fontsize=12)
axRC.set_ylabel('Alcohol', fontsize=12)
axRC.set_zlabel('Quality', fontsize=12)
xticks(fontsize=10)
yticks(fontsize=10)
title("Clusters: Red Wines", fontsize=18)


# Store means:

resRsort = sorted(resR, key=lambda resR:resR[2])

PoorSulphatesR = round(resRsort[0][0],4)
PoorAlcoholR = round(resRsort[0][1],4)
PoorQualityR = round(resRsort[0][2],4)

FairSulphatesR = round(resRsort[1][0],4)
FairAlcoholR = round(resRsort[1][1],4)
FairQualityR = round(resRsort[1][2],4)

GoodSulphatesR = round(resRsort[2][0],4)
GoodAlcoholR = round(resRsort[2][1],4)
GoodQualityR = round(resRsort[2][2],4)

ExcellentSulphatesR = round(resRsort[3][0],4)
ExcellentAlcoholR = round(resRsort[3][1],4)
ExcellentQualityR = round(resRsort[3][2],4)


# Find category colors:

test=pd.DataFrame(colorsR,idxR)
col = [0,0,0,0]
quals = [PoorQualityR, FairQualityR, GoodQualityR, ExcellentQualityR]
ind = idxR[where(test=='#3232ff')[0][0]]
val = round(resR[ind,2],4)
ind2 = quals.index(val)
```

```
col[ind2]='#3232ff'
ind = idxR[where(test=='#4ca64c')[0][0]]
val = round(resR[ind,2],4)
ind2 = quals.index(val)
col[ind2]='#4ca64c'
ind = idxR[where(test=='#ffa500')[0][0]]
val = round(resR[ind,2],4)
ind2 = quals.index(val)
col[ind2]='#ffa500'
ind = idxR[where(test=='#993299')[0][0]]
val = round(resR[ind,2],4)
ind2 = quals.index(val)
col[ind2]='#993299'

# Create table of means:

subplot2grid((2,2), (1,0), colspan=2)
plt.gca().axison=False
tablevalsR = [[PoorSulphatesR,PoorAlcoholR,PoorQualityR],[FairSulphatesR,F
             [GoodSulphatesR,GoodAlcoholR,GoodQualityR],[ExcellentSulphate
rowlabels = ['Poor', 'Fair', 'Good', 'Excellent']
collabels = ['Sulphates', 'Alcohol', 'Quality']
rowcolors = col
redtable = plt.table(cellText=tablevalsR, colWidths=[.18,.18,.18],rowLabel
                     loc='upper center', rowColours=rowcolors)
redtable.auto_set_font_size(False)
redtable.set_fontsize(12)
redtable.scale(1.2,1)
title("Red Cluster Means", fontsize=18, ha='center')


# Add caption:

text(0,.3,'''Figure 5. The left-hand graph shows a scatterplot of sulphate
wines data. Outliers push the plot to the far left. The right-hand graph c
using a k-means algorithm. Four distinct groupings are seen among the data
displays group means for each cluster of red wines created by the algorith
comparisons easy and descriptive.''',fontsize=12)

tight_layout()
subplots_adjust(hspace=.2, wspace=0)


# Reset figure size:

pylab.rcParams['figure.figsize'] = a
```



Red Cluster Means

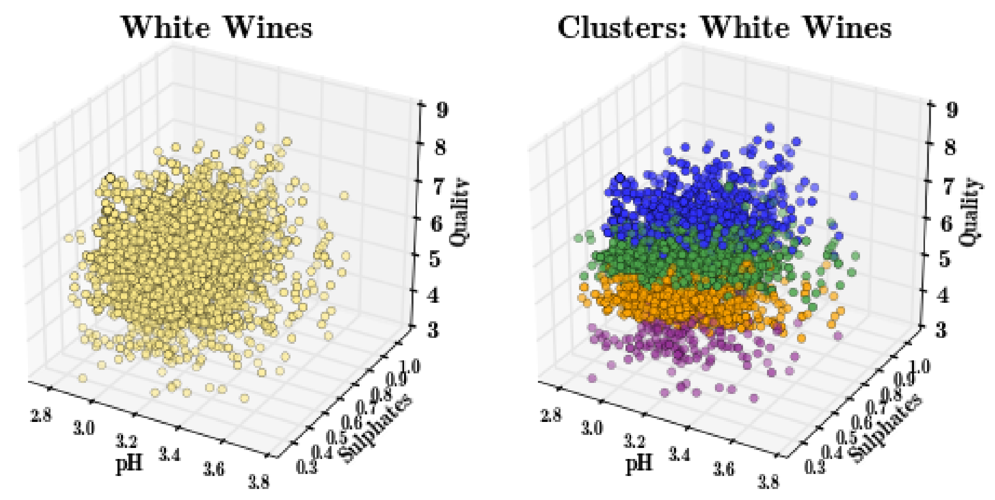
	Sulphates	Alcohol	Quality
Poor	0.6016	9.7318	4.6732
Fair	0.6171	10.9908	6.0626
Good	0.6415	9.6264	6.6597
Excellent	0.7011	11.4907	7.951

Figure 5. The left-hand graph shows a scatterplot of sulphates and alcohol against quality in the red wines data. Outliers push the plot to the far left. The right-hand graph demonstrates a cluster analysis using a k-means algorithm. Four distinct groupings are seen among the data points. The bottom table displays group means for each cluster of red wines created by the algorithm. This makes across-group comparisons easy and descriptive.

In Figure 5, the left-hand plot shows red wine scatter of sulphates and alcohol against quality. Then, a ‘k-means’ algorithm loops through iterations of different clusterings until the optimal combination is reached. The right-hand plot now shows the distinct groupings in the scatter. The attached table provides the means (or centroids) retrieved from the algorithm. As you can see, the quality score in the third column increases as you go from poor to excellent. This table is extremely useful if one wanted to see where a wine may fall based on sulphates and alcohol percentage. Hypothetically, say a new wine is created in the Vinho Verde region, but wine-makers do not have the resources to put it through taste-testing. This new wine has a sulphate measure of 0.6 and 10% alcohol. It is easy to see this wine would probably test poorly. Another advantage to this type of “modeling” is that the inquiring person could easily choose any 2 characteristics of interest to make a conclusion.

In [28]:
```
# Set figure size:

a = pylab.rcParams['figure.figsize']
pylab.rcParams['figure.figsize'] = (8.35, 8.35)

# 3D scatter plot of white wines:

pylab.close()
figW = plt.figure()
axW = figW.add_subplot(221, projection='3d')
axW.scatter(xs=Xw['pH'], ys=Xw['sulphates'], zs=Yw, c='#FFEC8B')
axW.set_xlabel('pH', fontsize=12)
axW.set_ylabel('Sulphates', fontsize=12)
axW.set_zlabel('Quality', fontsize=12)
xticks(fontsize=10)
yticks(fontsize=10)
plt.title("White Wines", fontsize=18)


# 3D scatter plot of white clusters:

axWC = figW.add_subplot(222, projection='3d')
resW, idxW = kmeans2(np.column_stack([Xw['pH'], Xw['sulphates'], whiten(Yw
colorsW = ([('#993299','#3232ff','#4ca64c','#ffa500')[i] for i in idxW])
axWC.scatter(Xw['pH'], Xw['sulphates'], Yw, c=colorsW)
axWC.set_xlabel('pH', fontsize=12)
axWC.set_ylabel('Sulphates', fontsize=12)
axWC.set_zlabel('Quality', fontsize=12)
xticks(fontsize=10)
yticks(fontsize=10)
plt.title("Clusters: White Wines", fontsize=18)


# Store means:

resWsort = sorted(resW, key=lambda resW:resW[2])

PoorPhW = round(resWsort[0][0],4)
PoorSulphatesW = round(resWsort[0][1],4)
PoorQualityW = round(resWsort[0][2],4)

FairPhW = round(resWsort[1][0],4)
FairSulphatesW = round(resWsort[1][1],4)
FairQualityW = round(resWsort[1][2],4)

GoodPhW = round(resWsort[2][0],4)
GoodSulphatesW = round(resWsort[2][1],4)
GoodQualityW = round(resWsort[2][2],4)

ExcellentPhW = round(resWsort[3][0],4)
ExcellentSulphatesW = round(resWsort[3][1],4)
ExcellentQualityW = round(resWsort[3][2],4)


# Find category colors:

test=pd.DataFrame(colorsW,idxW)
col = [0,0,0,0]
quals = [PoorQualityW, FairQualityW, GoodQualityW, ExcellentQualityW]
ind = idxW[where(test=='#3232ff')[0][0]]
val = round(resW[ind,2],4)
ind2 = quals.index(val)
col[ind2]='#3232ff'
ind = idxW[where(test=='#4ca64c')[0][0]]
val = round(resW[ind,2],4)
```

```
ind2 = quals.index(val)
col[ind2]='#4ca64c'
ind = idxW[where(test=='#ffa500')[0][0]]
val = round(resW[ind,2],4)
ind2 = quals.index(val)
col[ind2]='#ffa500'
ind = idxW[where(test=='#993299')[0][0]]
val = round(resW[ind,2],4)
ind2 = quals.index(val)
col[ind2]='#993299'


# Create table of means:

subplot2grid((2,2),(1,0),colspan=2)
plt.gca().axison=False
tablevalsW = [[PoorPhW,PoorSulphatesW,PoorQualityW],[FairPhW,FairSulphates
             [GoodPhW,GoodSulphatesW,GoodQualityW],[ExcellentPhW,Excellent
rowlabels = ['Poor', 'Fair', 'Good', 'Excellent']
collabels = ['pH', 'Sulphates', 'Quality']
rowcolors = col
whitetable = plt.table(cellText=tablevalsW, colWidths=[.18,.18,.18], rowLa
whitetable.auto_set_font_size(False)
whitetable.set_fontsize(12)
whitetable.scale(1.2,1)
title("White Cluster Means", fontsize=18, ha='center')


# Add caption:

text(0,.3,'''Figure 6. The left-hand graph shows a scatterplot of ph and s
wines data. Outliers push the plot to the far left. The right-hand graph c
using a k-means algorithm. Four distinct groupings are seen among the data
displays group means for each cluster of white wines created by the algori
comparisons easy and descriptive.''',fontsize=12)

tight_layout()
subplots_adjust(hspace=.2,wspace=0)


# Reset figure size:

pylab.rcParams['figure.figsize']=a
```



White Cluster Means

	pH	Sulphates	Quality
Poor	3.1834	0.476	4.3936
Fair	3.1688	0.4822	5.6462
Good	3.1886	0.4911	6.7755
Excellent	3.2151	0.5001	8.1018

Figure 6. The left-hand graph shows a scatterplot of ph and sulphates against quality in the white wines data. Outliers push the plot to the far left. The right-hand graph demonstrates a cluster analysis using a k-means algorithm. Four distinct groupings are seen among the data points. The bottom table displays group means for each cluster of white wines created by the algorithm, making across-group comparisons easy and descriptive.

In Figure 6, the left-hand plot shows white wine scatter of pH and sulphates against quality. The ‘k-means’ algorithm loops through the different clusterings until the optimal combination is reached. The right-hand plot shows the distinct groupings in the scatter. The attached table provides the means. The quality score in the third column increases as you go from poor to excellent. Hypothetically, a new wine with a pH of 3.0 and sulphate measure of 0.5 may fall in the fair or good category.

Conclusion

I can't say it enough: wine-making and wine-tasting are tricky concepts to nail down. This analysis has definitely supported that conclusion above all others. However, just because a phenomenon is difficult to analyze doesn't mean we can't do our best to visualize and interpret the data we are given. Adding to the body of knowledge is just as important whether the outcome is so-so results or amazing results. With that said, I will recap everything we learned here…

Objective #1: Prove red wines are different from white wines.
This ended up being the simplest and most successful part of the analysis. There is no doubt every characteristic was significantly different between the two wine types. Some differences were expected; others came as a surprise, but could potentially be attributed to the specific region the wines came from. This again raises the issue of ability to extend the results of this analysis to wines outside of the Vinho Verde region.

Objective #2: Predict quality from physicochemical characteristics.
While modeling quality did not yield useful or satisfying results, the process forced me to think about why this was happening. An important conclusion was made that the uniqueness of wine-making coupled with the subjectivity of a person's palate is far too complicated to be written in stone. Luckily, objective #3 was able to partially encompass this objective.

Objective #3: Categorize wines as excellent, good, fair, or poor.
While coefficients of a regression model are occasionally hard to grasp, I find a visual aid with descriptive statistics to be much more helpful in understanding a problem. For these data, a cluster analysis helps prediction as well as categorizing. We were able to successfully group our wines and compare group properties.

References

C. Kissack. The Components of Wine. Wine Doctor [http://www.thewinedoctor.com/advisory/tastecomponents.shtml].

Comissao de Viticultura da Regiao dos Vinhos Verdes. Vinho Verde [http://www.vinhoverde.pt].

K. Bache and M. Lichman (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

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, 2009.

Winemaking. Wikipedia [http://en.wikipedia.org/wiki/Winemaking].

Appendix

Regression Output

In [61]:
```
# Add column of 1's for intercept in design matrix:

Xr['intercept'] = 1
Xw['intercept'] = 1
```

In [64]:
```
# Full model OLS for red wines:

sm.OLS(Yr, Xr).fit().summary()
```

```
## 1. Remove density:
sm.OLS(Yr, Xr[['intercept', 'fixed acidity', 'volatile acidity', 'citric a
## 2. Remove fixed acidity:
sm.OLS(Yr, Xr[['intercept', 'volatile acidity', 'citric acid', 'residual s
## 3. Remove residual sugar:
sm.OLS(Yr, Xr[['intercept', 'volatile acidity', 'citric acid', 'chlorides'
## 4. Remove citric acid:
sm.OLS(Yr, Xr[['intercept', 'volatile acidity', 'chlorides', 'free sulfur
```

Out [64]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
========
Dep. Variable:                 quality   R-squared:
0.359
Model:                             OLS   Adj. R-squared:
0.357
Method:                  Least Squares   F-statistic:
127.6
Date:                 Mon, 28 Apr 2014   Prob (F-statistic):
5.32e-149
Time:                         08:34:22   Log-Likelihood:
-1570.5
No. Observations:                 1599   AIC:
3157.
Df Residuals:                     1591   BIC:
3200.
Df Model:                            7
==============================================================================
==================
                          coef    std err          t      P>|t|
[95.0% Conf. Int.]
------------------------------------------------------------------------------
------------------
intercept               4.4301      0.403     10.995      0.000
3.640     5.220
volatile acidity       -1.0128      0.101    -10.043      0.000
-1.211    -0.815
chlorides              -2.0178      0.398     -5.076      0.000
-2.798    -1.238
free sulfur dioxide     0.0051      0.002      2.389      0.017
0.001     0.009
total sulfur dioxide   -0.0035      0.001     -5.070      0.000
-0.005    -0.002
pH                     -0.4827      0.118     -4.106      0.000
-0.713    -0.252
sulphates               0.8827      0.110      8.031      0.000
0.667     1.098
alcohol                 0.2893      0.017     17.225      0.000
```

```
0.256     0.322
==============================================================================
========
Omnibus:                       24.204   Durbin-Watson:
1.750
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
35.245
Skew:                          -0.156   Prob(JB):
2.22e-08
Kurtosis:                       3.657   Cond. No.
1.71e+03
==============================================================================
========

Warnings:
[1] The condition number is large, 1.71e+03. This might indicate that
there are
strong multicollinearity or other numerical problems.
"""
```

In [34]:

```
# Full model OLS for white wines:
sm.OLS(Yw, Xw).fit().summary()
## 1. Remove citric acid:
sm.OLS(Yw, Xw[['intercept', 'fixed acidity', 'volatile acidity', 'residual
## 2. Remove chlorides:
sm.OLS(Yw, Xw[['intercept', 'fixed acidity', 'volatile acidity', 'residual
## 3. Remove total sulfur dioxide:
sm.OLS(Yw, Xw[['intercept', 'fixed acidity', 'volatile acidity', 'residual
```

Out [34]:

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
========
Dep. Variable:                quality   R-squared:
0.282
Model:                            OLS   Adj. R-squared:
0.281
Method:                 Least Squares   F-statistic:
239.7
Date:                Fri, 14 Mar 2014   Prob (F-statistic):
0.00
Time:                        10:24:09   Log-Likelihood:
-5544.1
No. Observations:                4898   AIC:
1.111e+04
Df Residuals:                    4889   BIC:
1.116e+04
Df Model:                           8
==============================================================================
=================
                           coef    std err          t      P>|t|
[95.0% Conf. Int.]
----------------------------------------------------------------------
-----------------
intercept               154.1062     18.100      8.514      0.000
118.622   189.591
fixed acidity             0.0681      0.020      3.333      0.001
0.028     0.108
volatile acidity         -1.8881      0.110    -17.242      0.000
-2.103    -1.673
residual sugar            0.0828      0.007     11.370      0.000
0.069     0.097
free sulfur dioxide       0.0033      0.001      4.950      0.000
0.002     0.005
density                -154.2913     18.344     -8.411      0.000
-190.254  -118.329
pH                        0.6942      0.103      6.717      0.000
0.492     0.897
sulphates                 0.6285      0.100      6.287      0.000
0.433     0.824
alcohol                   0.1932      0.024      8.021      0.000
0.146     0.240
======================================================================
========
Omnibus:                      114.194   Durbin-Watson:
1.621
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
251.255
Skew:                           0.075   Prob(JB):
2.76e-55
Kurtosis:                       4.099   Cond. No.
9.95e+04
======================================================================
========

Warnings:
[1] The condition number is large, 9.95e+04. This might indicate that
there are
strong multicollinearity or other numerical problems.
"""
```

In [15]:
```
# OLS with removed collinearity for red wines:

sm.OLS(Yr, Xr[['intercept', 'volatile acidity', 'total sulfur dioxide', 'p
```

Out [15]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
======================================================================
========
Dep. Variable:                quality   R-squared:
0.347
Model:                            OLS   Adj. R-squared:
0.345
```

```
Method:                 Least Squares   F-statistic:
169.3
Date:                Mon, 17 Mar 2014   Prob (F-statistic):
1.39e-144
Time:                        15:03:49   Log-Likelihood:
-1586.0
No. Observations:                1599   AIC:
3184.
Df Residuals:                    1593   BIC:
3216.
Df Model:                           5
============================================================================
==================
                          coef    std err          t      P>|t|
[95.0% Conf. Int.]
----------------------------------------------------------------------------
------------------
intercept               3.7488      0.387      9.677      0.000
2.989     4.509
volatile acidity       -1.1298      0.100    -11.354      0.000
-1.325    -0.935
total sulfur dioxide   -0.0023      0.001     -4.456      0.000
-0.003    -0.001
pH                     -0.3189      0.115     -2.783      0.005
-0.544    -0.094
sulphates               0.6671      0.102      6.565      0.000
0.468     0.866
alcohol                 0.3076      0.017     18.534      0.000
0.275     0.340
============================================================================
========
Omnibus:                       28.204   Durbin-Watson:
1.753
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
42.779
Skew:                          -0.169   Prob(JB):
5.14e-10
Kurtosis:                       3.727   Cond. No.
1.41e+03
============================================================================
========

Warnings:
[1] The condition number is large, 1.41e+03. This might indicate that
there are
strong multicollinearity or other numerical problems.
"""
```

In [16]:
```
# OLS with removed collinearity for white wines:

sm.OLS(Yw, Xw[['intercept', 'fixed acidity', 'volatile acidity', 'residual
Out [16]: <class 'statsmodels.iolib.summary.Summary'>
          """
                                      OLS Regression Results
          ==============================================================================
          ========
          Dep. Variable:                  quality   R-squared:
          0.267
          Model:                              OLS   Adj. R-squared:
          0.266
          Method:                   Least Squares   F-statistic:
          297.4
          Date:                  Mon, 17 Mar 2014   Prob (F-statistic):
          0.00
          Time:                          15:06:27   Log-Likelihood:
          -5592.9
          No. Observations:                  4898   AIC:
          1.120e+04
          Df Residuals:                      4891   BIC:
          1.125e+04
          Df Model:                             6
          ==============================================================================
          ==============
                                 coef    std err          t      P>|t|
          [95.0% Conf. Int.]
          ------------------------------------------------------------------------------
          --------------
          intercept            2.0890      0.336      6.218      0.000
          1.430     2.748
          fixed acidity       -0.0615      0.014     -4.311      0.000
          -0.090    -0.034
          volatile acidity    -2.0962      0.109    -19.315      0.000
          -2.309    -1.883
          residual sugar       0.0283      0.002     11.595      0.000
          0.023     0.033
          pH                   0.1608      0.082      1.969      0.049
          0.001     0.321
          sulphates            0.4126      0.096      4.280      0.000
          0.224     0.602
          alcohol              0.3708      0.010     37.189      0.000
          0.351     0.390
          ==============================================================================
          ========
          Omnibus:                       95.219   Durbin-Watson:
          1.621
          Prob(Omnibus):                  0.000   Jarque-Bera (JB):
          192.246
          Skew:                           0.073   Prob(JB):
          1.80e-42
          Kurtosis:                       3.960   Cond. No.
          465.
          ==============================================================================
          ========
          """
```

Testing Suite

In [89]:
```
import nose

# Check missing values in imported data:

def test_csvRed():
    assert pd.isnull(reds[0:11]) == False

def test_csvWhite():
    assert pd.isnull(whites[0:11]) == False


# Check that calculated t values are numbers and not missing:

def test_ttests():
    assert [t1,t2,t3,t4,t5,t6,t7,t8,t9,t10,t11,t12] <> NaN


# Check that p-values are non-negative:

def test_pvals():
    assert [p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,p11,p12] >= 0


# Check that intercept is included in design matrix:

def test_Xred():
    assert Xr['intercept']==1

def test_Xwhite():
    assert Xw['intercept']==1


# Check that created design matrices and response vectors are correct dime

def test_Xredlength():
    assert len(Xr)==len(reds)

def test_Xwhitelength():
    assert len(Xw)==len(whites)

def test_Yredlength():
    assert len(Yr)==len(reds)

def test_Ywhitelength():
    assert len(Yw)==len(whites)


# Check cluster means are numbers and positive:

def test_clustermeansRed():
    assert resR <= 0

def test_clustermeansWhite():
    assert resW <= 0


# Check table means match stored means:

def test_clustertableRed():
    assert resR == tablevalsR

def test_clustertableWhite():
    assert resW == tablevalsW
```