Overview

Input Variables:

fixed acidity: most acids involved with wine or fixed or nonvolatile

volatile acidity: the amount of acetic acid in wine

citric acid: found in small quantities, citric acid can add 'freshness' and flavor to wines

residual sugar: the amount of sugar remaining after fermentation stops

chlorides: the amount of salt in the wine

free sulfur dioxide: the free form of SO2 exists in equilibrium between molecular SO2 (as a dissolved gas) and bisulfite ion

total sulfur dioxide: amount of free and bound forms of S02

density: the density of water is close to that of water depending on the percent alcohol and sugar content

pH: describes how acidic or basic a wine is on a scale from 0 (very acidic) to 14 (very basic)

sulphates: a wine additive which can contribute to sulfur dioxide gas (S02) levels

alcohol: the percent alcohol content of the wine

Output Variable:

quality: output variable (based on sensory data, score between 0 and 10)

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import accuracy_score
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import cross_val_score
from sklearn.metrics import confusion_matrix
from collections import Counter
from IPython.core.display import display, HTML
sns.set_style('darkgrid')
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
wine = pd.read_csv('Red_wine.csv')
```

In [3]:
```
wine.head()
```

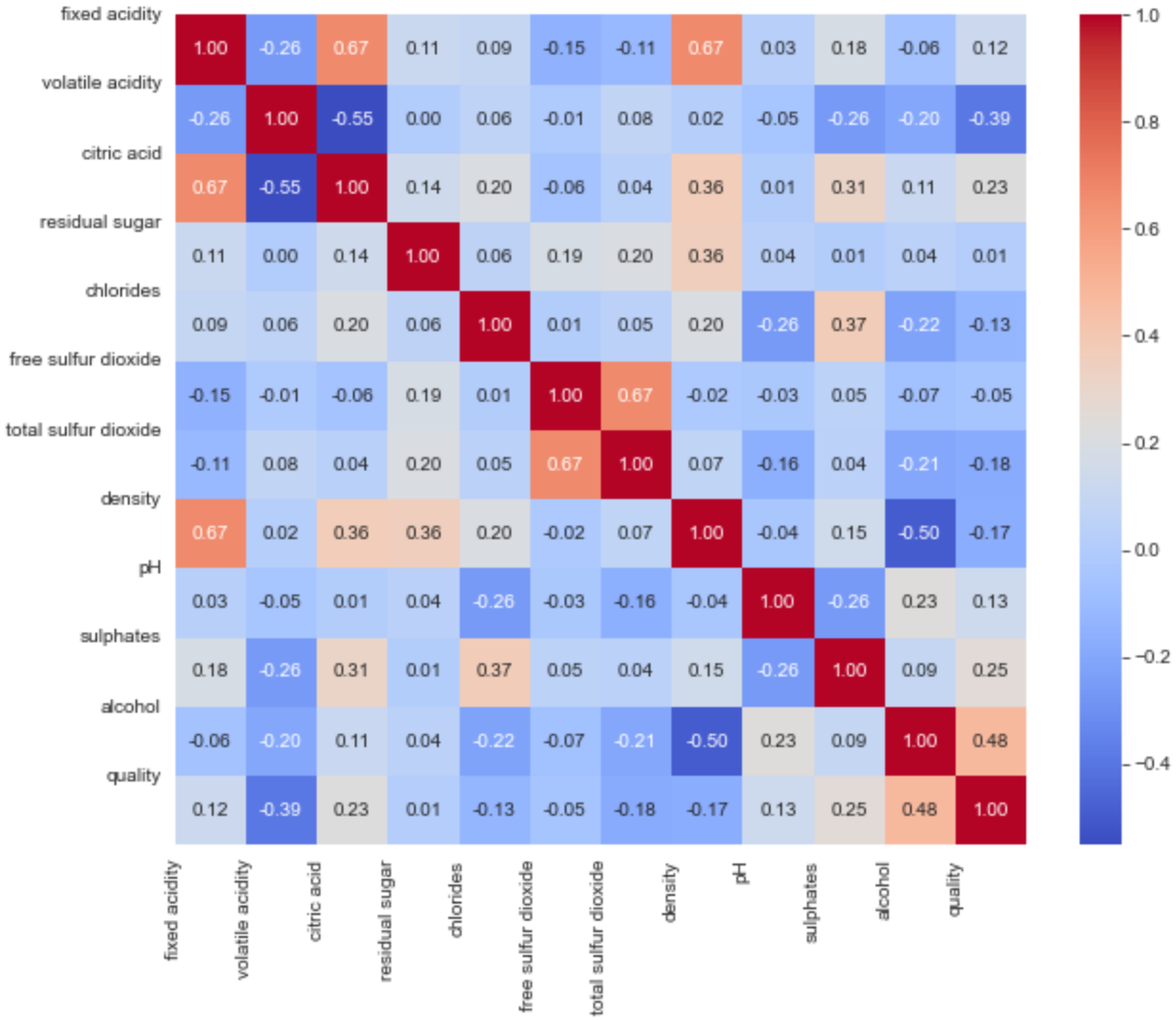
Out[3]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5.0

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5.0
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5.0
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6.0
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5.0

In [4]:

```
corr = wine.corr()
#Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
#Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap='coolwarm', annot=True, fmt=".2f")
#Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
#Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
#show plot
plt.show()
```



In [5]:

```
pred_test = wine.iloc[3]
```

In [6]:

```
pred_test['type'] = 1
```

```
pred_test.drop(['quality','total sulfur dioxide'],inplace=True)
#pred_test.drop('total_sulfur_dioxide',inplace=True)
pred_test
```

Out[6]:
```
fixed acidity          11.200
volatile acidity        0.280
citric acid             0.560
residual sugar          1.900
chlorides               0.075
free sulfur dioxide    17.000
density                 0.998
pH                      3.160
sulphates               0.580
alcohol                 9.800
type                    1.000
Name: 3, dtype: float64
```

In [7]:
```
wine.shape
```

Out[7]:
```
(1599, 12)
```

In [8]:
```
wine.isnull().sum()
```

Out[8]:
```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    1
density                 0
pH                      1
sulphates               0
alcohol                 0
quality                 1
dtype: int64
```

In [9]:
```
wine2 = wine.fillna(value=wine['total sulfur dioxide'].mean())
wine2 = wine.fillna(value=wine['pH'].mean())
wine2 = wine.fillna(value=wine['quality'].mean())
```

In [10]:
```
wine2.isnull().sum()
```

Out[10]:
```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64
```

In [11]:
```
wine2.describe()
```

Out[11]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	159
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.874922	46.407527	0.996747	
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.460157	32.881792	0.001887	
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	5.636421	0.990070	
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	22.000000	0.995600	
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	38.000000	0.996750	
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	62.000000	0.997835	
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	289.000000	1.003690	

In [12]:
```
wine2.info()
```

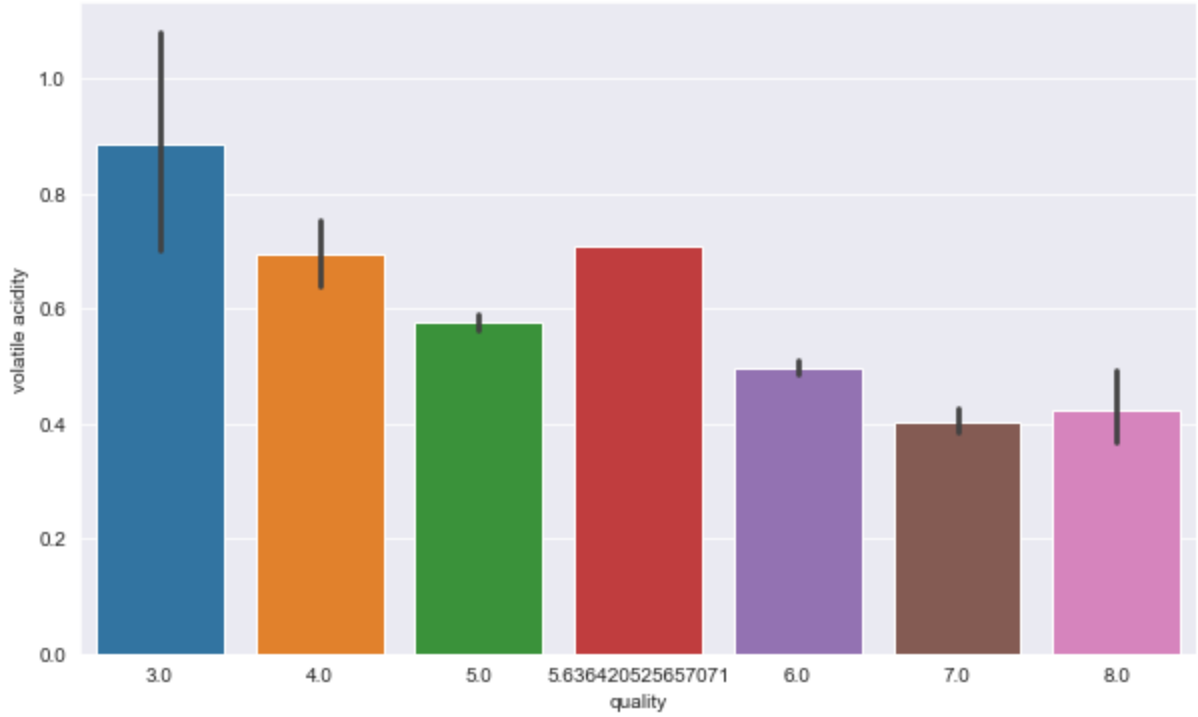
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         1599 non-null   float64
 1   volatile acidity      1599 non-null   float64
 2   citric acid           1599 non-null   float64
 3   residual sugar        1599 non-null   float64
 4   chlorides             1599 non-null   float64
 5   free sulfur dioxide   1599 non-null   float64
 6   total sulfur dioxide  1599 non-null   float64
 7   density               1599 non-null   float64
 8   pH                    1599 non-null   float64
 9   sulphates             1599 non-null   float64
 10  alcohol               1599 non-null   float64
 11  quality               1599 non-null   float64
dtypes: float64(12)
memory usage: 150.0 KB
```

In [13]:
```
wine2.isna().any()
```

Out[13]:
```
fixed acidity           False
volatile acidity        False
citric acid             False
residual sugar          False
chlorides               False
free sulfur dioxide     False
total sulfur dioxide    False
density                 False
pH                      False
sulphates               False
alcohol                 False
quality                 False
dtype: bool
```

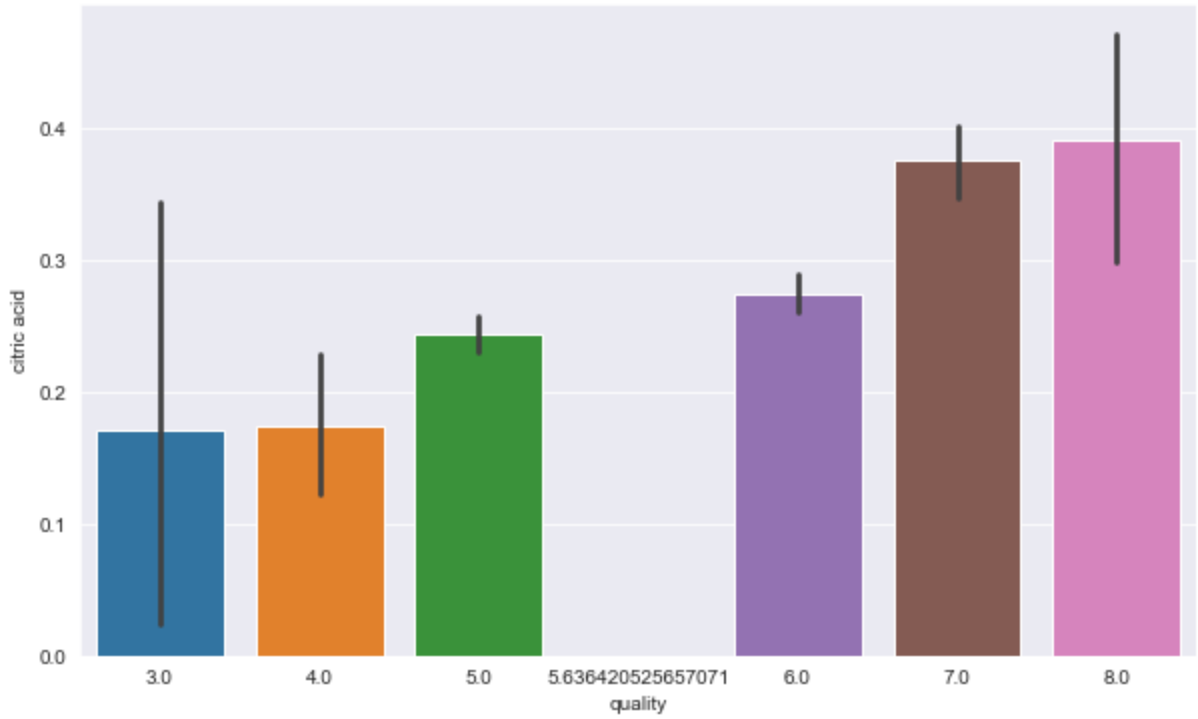
In [14]:
```
fig = plt.figure(figsize = (10,6))
sns.barplot(x = 'quality', y = 'volatile acidity', data = wine2)
```

Out[14]:
```
<AxesSubplot:xlabel='quality', ylabel='volatile acidity'>
```



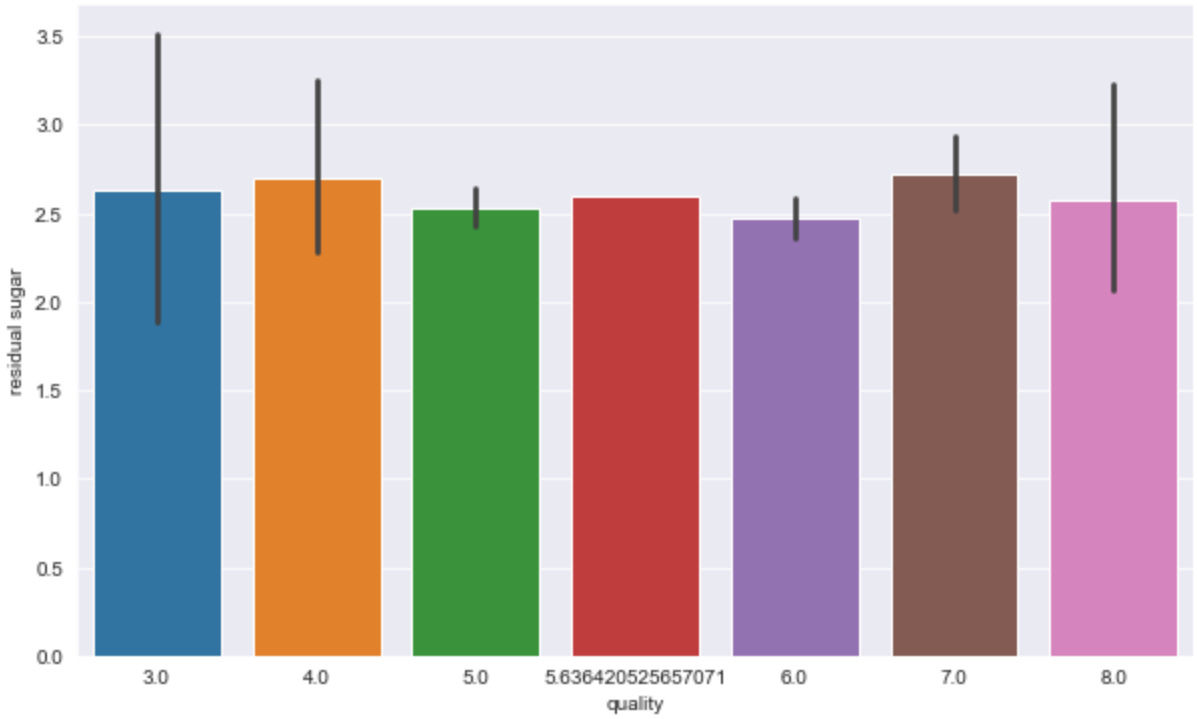
In [15]:
```
# composition of citric acid go higher as we go higher in the quality of the wine
fig = plt.figure(figsize = (10,6))
sns.barplot(x = 'quality', y = 'citric acid', data = wine2)
```

Out[15]:
```
<AxesSubplot:xlabel='quality', ylabel='citric acid'>
```



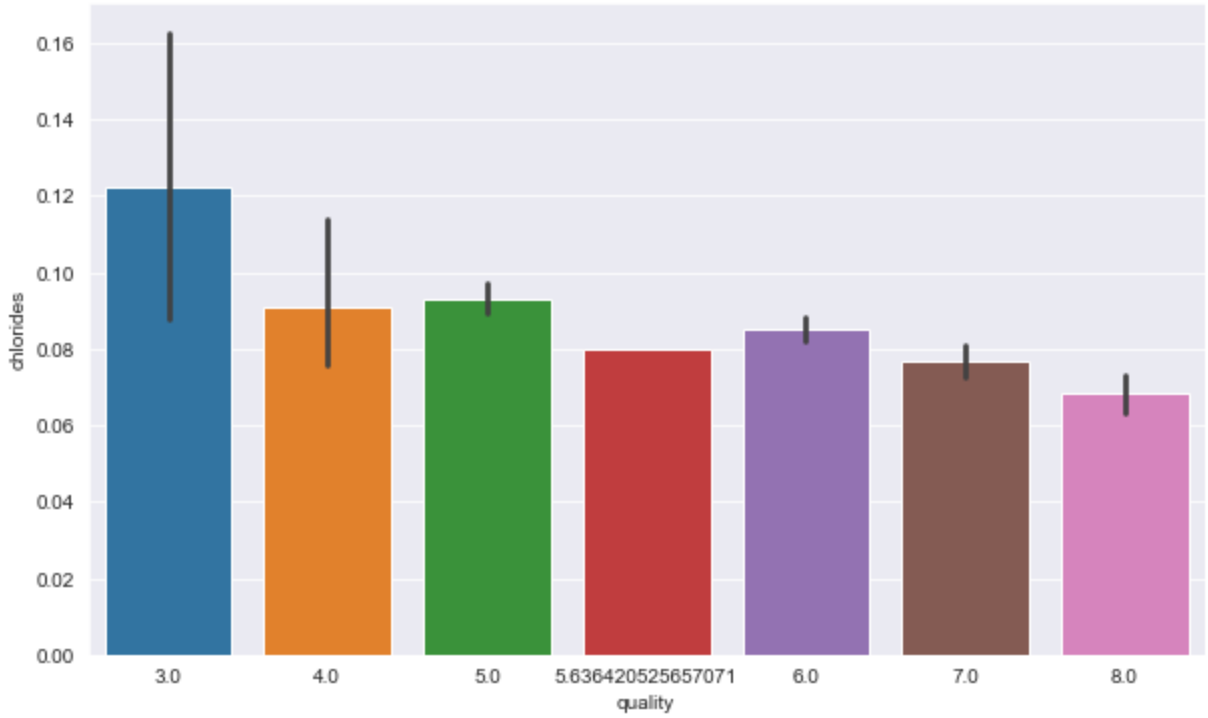
In [16]:
```
fig = plt.figure(figsize = (10,6))
sns.barplot(x = 'quality', y = 'residual sugar', data = wine2)
```

Out[16]:
```
<AxesSubplot:xlabel='quality', ylabel='residual sugar'>
```

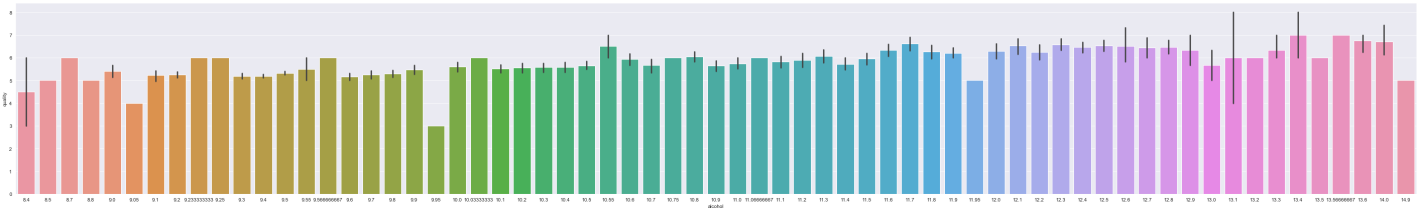


In [17]:
```
fig = plt.figure(figsize = (10,6))
sns.barplot(x = 'quality', y = 'chlorides', data = wine2)
```

Out[17]:
```
<AxesSubplot:xlabel='quality', ylabel='chlorides'>
```

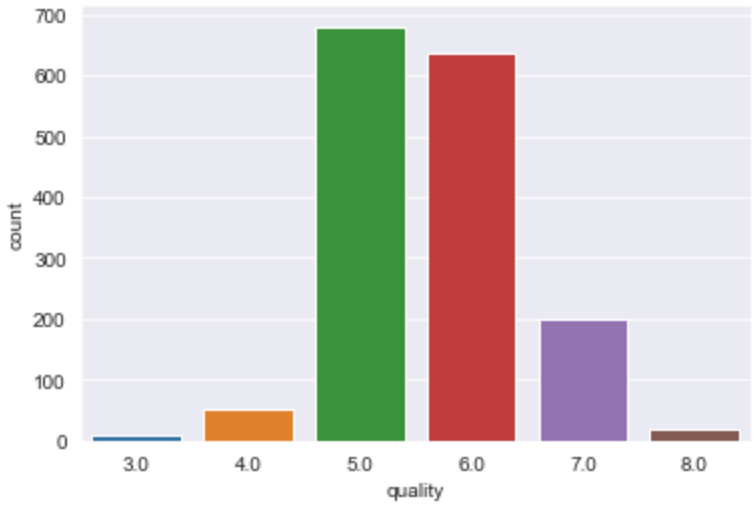


In [18]:
```
fig = plt.figure(figsize = (50,7))
ax = sns.barplot(x = 'alcohol', y = 'quality', data = wine2)
```



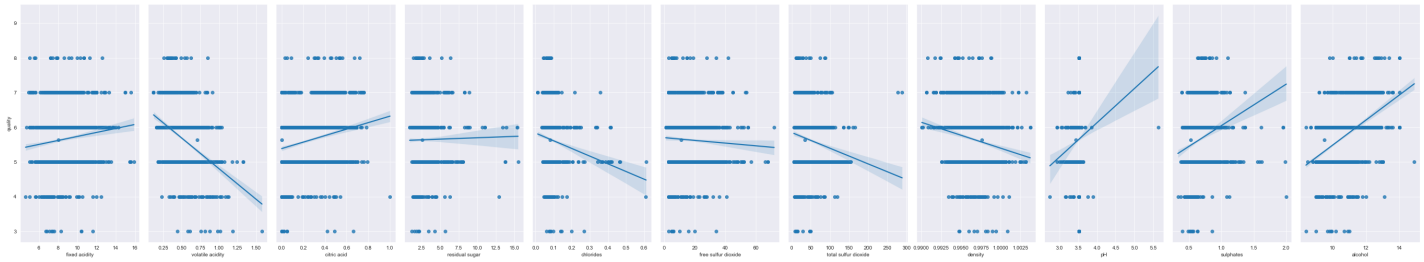
In [19]:
```
sns.countplot(wine['quality'])
```

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



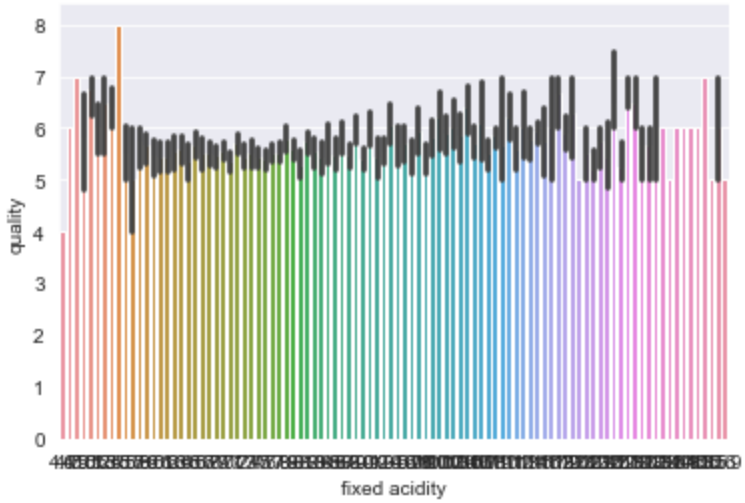
In [20]:
```
features = ['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar', 'chloric
x = wine2[features]
y = wine2['quality']
sns.pairplot(wine2, x_vars=features , y_vars='quality', kind='reg', size=7, aspect=0.5)
```

Out[20]:
```
<seaborn.axisgrid.PairGrid at 0x1ef5f693cd0>
```



In [21]:
```
sns.barplot(x = 'fixed acidity', y = 'quality', data = wine2)
```

Out[21]:
```
<AxesSubplot:xlabel='fixed acidity', ylabel='quality'>
```



In [22]:
```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, random_state = 3)
```

In [23]:
```
regressor = LinearRegression()
regressor.fit(x_train, y_train)
```

Out[23]:
```
▾ LinearRegression
LinearRegression()
```

In [24]:
```
accuracy = regressor.score(x_test, y_test)
"Accuracy : {}%".format(int(round(accuracy*100)))
```

Out[24]:
```
'Accuracy : 38%'
```