In [1]:

```
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.svm import SVC
import matplotlib.pyplot as plt
from sklearn.linear_model import SGDClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report,confusion_matrix,accuracy_score

%matplotlib inline
```

In [2]:

```
wine = pd.read_csv('winequality-red.csv')
wine.head()
```

Out[2]:

| | fixed acidity | volatile acidity | citric acid | residual sugar | chlorides | free sulfur dioxide | total sulfur dioxide | density | pH | sulphates | alcohol |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 7.4 | 0.70 | 0.00 | 1.9 | 0.076 | 11.0 | 34.0 | 0.9978 | 3.51 | 0.56 | 9.4 |
| **1** | 7.8 | 0.88 | 0.00 | 2.6 | 0.098 | 25.0 | 67.0 | 0.9968 | 3.20 | 0.68 | 9.8 |
| **2** | 7.8 | 0.76 | 0.04 | 2.3 | 0.092 | 15.0 | 54.0 | 0.9970 | 3.26 | 0.65 | 9.8 |
| **3** | 11.2 | 0.28 | 0.56 | 1.9 | 0.075 | 17.0 | 60.0 | 0.9980 | 3.16 | 0.58 | 9.8 |
| **4** | 7.4 | 0.70 | 0.00 | 1.9 | 0.076 | 11.0 | 34.0 | 0.9978 | 3.51 | 0.56 | 9.4 |

In [3]:

```
wine.describe()
```

Out[3]:

| | fixed acidity | volatile acidity | citric acid | residual sugar | chlorides | free sulfur dioxide | total sulfur dioxide |
|---|---|---|---|---|---|---|---|
| **count** | 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 | 46.467792 |
| **std** | 1.741096 | 0.179060 | 0.194801 | 1.409928 | 0.047065 | 10.460157 | 32.895324 |
| **min** | 4.600000 | 0.120000 | 0.000000 | 0.900000 | 0.012000 | 1.000000 | 6.000000 |
| **25%** | 7.100000 | 0.390000 | 0.090000 | 1.900000 | 0.070000 | 7.000000 | 22.000000 |
| **50%** | 7.900000 | 0.520000 | 0.260000 | 2.200000 | 0.079000 | 14.000000 | 38.000000 |
| **75%** | 9.200000 | 0.640000 | 0.420000 | 2.600000 | 0.090000 | 21.000000 | 62.000000 |
| **max** | 15.900000 | 1.580000 | 1.000000 | 15.500000 | 0.611000 | 72.000000 | 289.000000 |

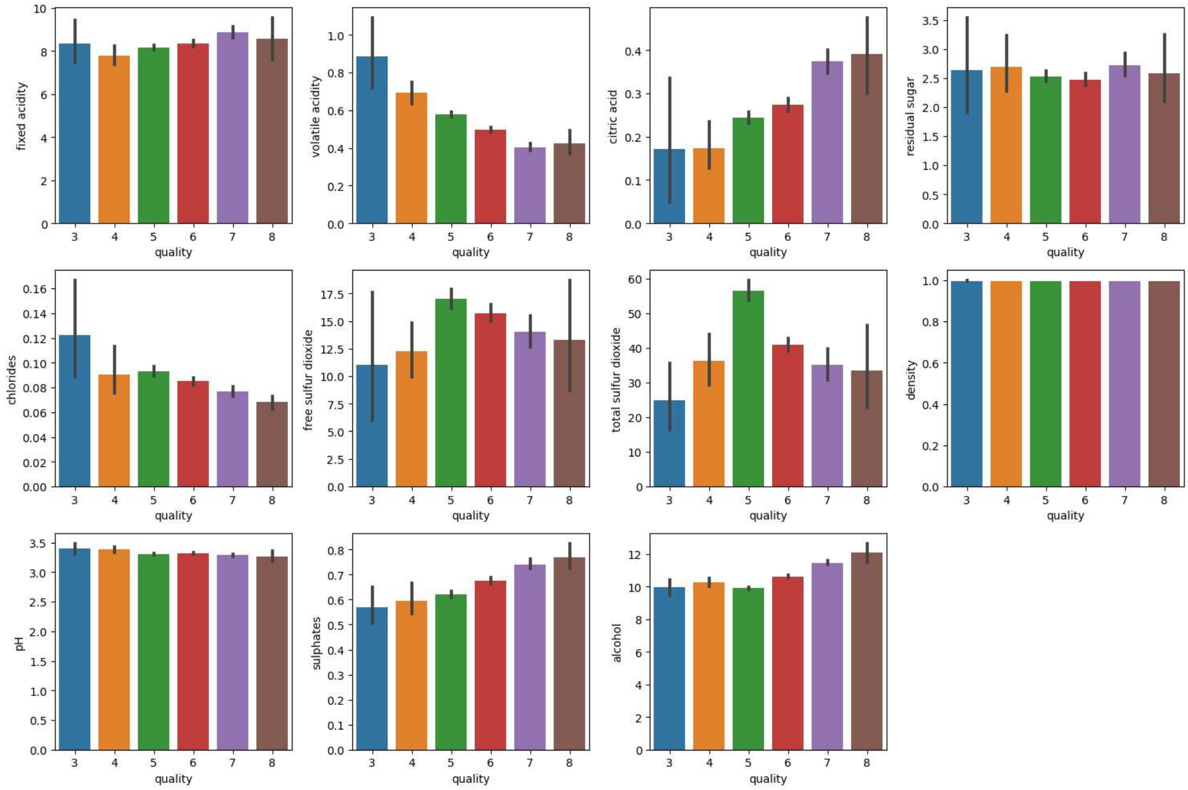
In [4]:

```
wine.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   int64  
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

In [5]:

```
fig = plt.figure(figsize=(15,10))

plt.subplot(3,4,1)
sns.barplot(x='quality',y='fixed acidity',data=wine)

plt.subplot(3,4,2)
sns.barplot(x='quality',y='volatile acidity',data=wine)

plt.subplot(3,4,3)
sns.barplot(x='quality',y='citric acid',data=wine)

plt.subplot(3,4,4)
sns.barplot(x='quality',y='residual sugar',data=wine)

plt.subplot(3,4,5)
sns.barplot(x='quality',y='chlorides',data=wine)

plt.subplot(3,4,6)
sns.barplot(x='quality',y='free sulfur dioxide',data=wine)

plt.subplot(3,4,7)
sns.barplot(x='quality',y='total sulfur dioxide',data=wine)

plt.subplot(3,4,8)
sns.barplot(x='quality',y='density',data=wine)

plt.subplot(3,4,9)
sns.barplot(x='quality',y='pH',data=wine)

plt.subplot(3,4,10)
sns.barplot(x='quality',y='sulphates',data=wine)

plt.subplot(3,4,11)
sns.barplot(x='quality',y='alcohol',data=wine)

plt.tight_layout()
```




In [6]:
```
wine['quality'].value_counts()
```

Out[6]:
```
5    681
6    638
7    199
4     53
8     18
3     10
Name: quality, dtype: int64
```

In [7]:
```
ranges = (2,6.5,8)
groups = ['bad','good']
wine['quality'] = pd.cut(wine['quality'],bins=ranges,labels=groups)
le = LabelEncoder()
wine['quality'] = le.fit_transform(wine['quality'])
wine.head()
```

Out[7]:

| | fixed acidity | volatile acidity | citric acid | residual sugar | chlorides | free sulfur dioxide | total sulfur dioxide | density | pH | sulphates | alcohol |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 7.4 | 0.70 | 0.00 | 1.9 | 0.076 | 11.0 | 34.0 | 0.9978 | 3.51 | 0.56 | 9.4 |
| **1** | 7.8 | 0.88 | 0.00 | 2.6 | 0.098 | 25.0 | 67.0 | 0.9968 | 3.20 | 0.68 | 9.8 |
| **2** | 7.8 | 0.76 | 0.04 | 2.3 | 0.092 | 15.0 | 54.0 | 0.9970 | 3.26 | 0.65 | 9.8 |
| **3** | 11.2 | 0.28 | 0.56 | 1.9 | 0.075 | 17.0 | 60.0 | 0.9980 | 3.16 | 0.58 | 9.8 |
| **4** | 7.4 | 0.70 | 0.00 | 1.9 | 0.076 | 11.0 | 34.0 | 0.9978 | 3.51 | 0.56 | 9.4 |

In [8]:
```
wine['quality'].value_counts()
```

Out[8]:
```
0    1382
1     217
Name: quality, dtype: int64
```

In [9]:
```
good_quality = wine[wine['quality']==1]
bad_quality = wine[wine['quality']==0]

bad_quality = bad_quality.sample(frac=1)
bad_quality = bad_quality[:217]

new_df = pd.concat([good_quality,bad_quality])
new_df = new_df.sample(frac=1)
new_df
```

Out[9]:

| | fixed acidity | volatile acidity | citric acid | residual sugar | chlorides | free sulfur dioxide | total sulfur dioxide | density | pH | sulphates | alcol |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **1421** | 7.5 | 0.400 | 0.18 | 1.6 | 0.079 | 24.0 | 58.0 | 0.99650 | 3.34 | 0.58 | |
| **549** | 9.0 | 0.530 | 0.49 | 1.9 | 0.171 | 6.0 | 25.0 | 0.99750 | 3.27 | 0.61 | |
| **1193** | 6.4 | 0.885 | 0.00 | 2.3 | 0.166 | 6.0 | 12.0 | 0.99551 | 3.56 | 0.51 | 1 |
| **124** | 7.8 | 0.500 | 0.17 | 1.6 | 0.082 | 21.0 | 102.0 | 0.99600 | 3.39 | 0.48 | |
| **640** | 9.9 | 0.540 | 0.45 | 2.3 | 0.071 | 16.0 | 40.0 | 0.99910 | 3.39 | 0.62 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| **1204** | 7.2 | 0.360 | 0.46 | 2.1 | 0.074 | 24.0 | 44.0 | 0.99534 | 3.40 | 0.85 | 1 |
| **842** | 10.6 | 0.500 | 0.45 | 2.6 | 0.119 | 34.0 | 68.0 | 0.99708 | 3.23 | 0.72 | 1 |
| **290** | 8.7 | 0.520 | 0.09 | 2.5 | 0.091 | 20.0 | 49.0 | 0.99760 | 3.34 | 0.86 | 1 |
| **1440** | 7.2 | 0.370 | 0.32 | 2.0 | 0.062 | 15.0 | 28.0 | 0.99470 | 3.23 | 0.73 | 1 |
| **455** | 11.3 | 0.620 | 0.67 | 5.2 | 0.086 | 6.0 | 19.0 | 0.99880 | 3.22 | 0.69 | 1 |

434 rows × 12 columns

In [10]:
```
new_df['quality'].value_counts()
```

Out[10]:
```
0    217
1    217
Name: quality, dtype: int64
```

In [11]:
```
new_df.corr()['quality'].sort_values(ascending=False)
```

Out[11]:
```
quality                 1.000000
alcohol                 0.577559
sulphates               0.355054
citric acid             0.319310
fixed acidity           0.174802
residual sugar         -0.001258
pH                     -0.069247
free sulfur dioxide    -0.088221
chlorides              -0.209602
total sulfur dioxide   -0.213407
density                -0.245622
volatile acidity       -0.457369
Name: quality, dtype: float64
```

In [18]:
```
from sklearn.model_selection import train_test_split

X = new_df.drop('quality',axis=1)
y = new_df['quality']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_sta
```

In [19]:
```
param = {'n_estimators':[100,200,300,400,500,600,700,800,900,1000]}

grid_rf = GridSearchCV(RandomForestClassifier(),param,scoring='accuracy',cv=10,)
grid_rf.fit(X_train, y_train)

print('Best parameters --> ', grid_rf.best_params_)

# Wine Quality Prediction
pred = grid_rf.predict(X_test)

print(confusion_matrix(y_test,pred))
print('\n')
print(classification_report(y_test,pred))
print('\n')
print(accuracy_score(y_test,pred))
```

```
Best parameters -->  {'n_estimators': 900}
[[55 19]
 [ 3 54]]


              precision    recall  f1-score   support

           0       0.95      0.74      0.83        74
           1       0.74      0.95      0.83        57

    accuracy                           0.83       131
   macro avg       0.84      0.85      0.83       131
weighted avg       0.86      0.83      0.83       131



0.8320610687022901
```

In [ ]: