Support Vector Machines

Fraud in Wine

Wine fraud relates to the commercial aspects of wine. The most prevalent type of fraud is one where wines are adulterated, usually with the addition of cheaper products (e.g. juices) and sometimes with harmful chemicals and sweeteners (compensating for color or flavor).

Counterfeiting and the relabelling of inferior and cheaper wines to more expensive brands is another common type of wine fraud.

Project Goals

A distribution company that was recently a victim of fraud has completed an audit of various samples of wine through the use of chemical analysis on samples. The distribution company specializes in exporting extremely high quality, expensive wines, but was defrauded by a supplier who was attempting to pass off cheap, low quality wine as higher grade wine. The distribution company has hired you to attempt to create a machine learning model that can help detect low quality (a.k.a "fraud") wine samples. They want to know if it is even possible to detect such a difference.

Data Source: 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.

In [1]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:

```
df = pd.read_csv("../DATA/wine_fraud.csv")
```

In [3]:

```
df.head()
```

Out[3]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcoh
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9

In [4]:

```
df.shape
```

Out[4]:

```
(6497, 13)
```

In [5]:

```
df['quality'].unique()
```

Out[5]:

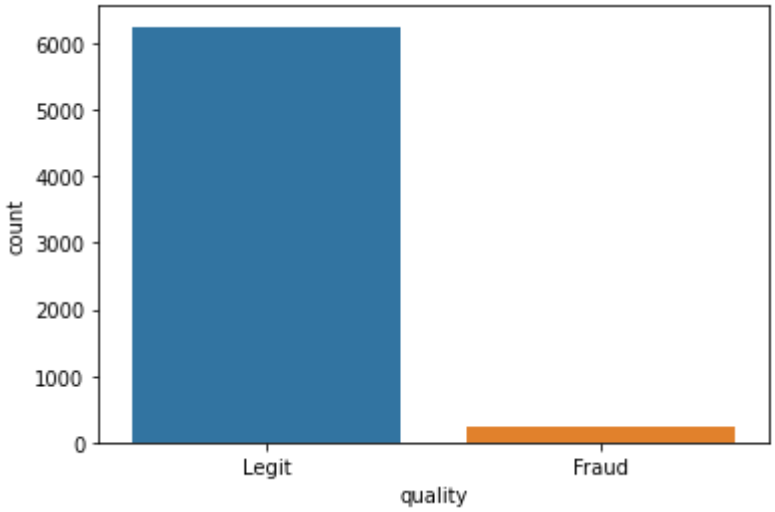
```
array(['Legit', 'Fraud'], dtype=object)
```

In [6]:

```
1 sns.countplot(x = 'quality',data = df)
```

Out[6]:

<AxesSubplot:xlabel='quality', ylabel='count'>

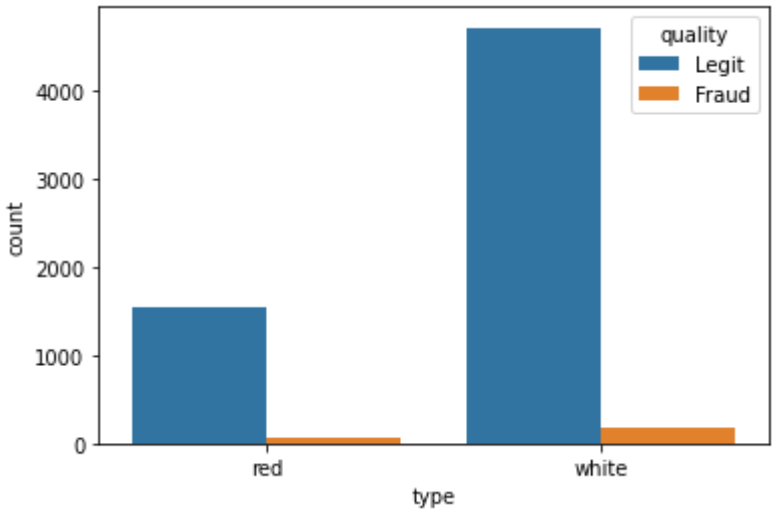


In [7]:

```
1 sns.countplot(x = 'type',data = df,hue='quality')
```

Out[7]:

<AxesSubplot:xlabel='type', ylabel='count'>



In [8]:

```
reds = df[df['type'] == 'red']
```

In [9]:

```
whites = df[df['type'] == 'white']
```

In [10]:

```
len(reds[reds['quality'] == 'Fraud'])/len(reds)*100
```

Out[10]:

```
3.9399624765478425
```

In [11]:

```
len(whites[whites['quality'] == 'Fraud'])/len(whites)*100
```

Out[11]:

```
3.7362188648427925
```

In [12]:

```
df['Fraud'] = df['quality'].map({'Legit':0 , 'Fraud':1})
```

In [13]:

```
df.corr()['Fraud'].sort_values()
```

Out[13]:

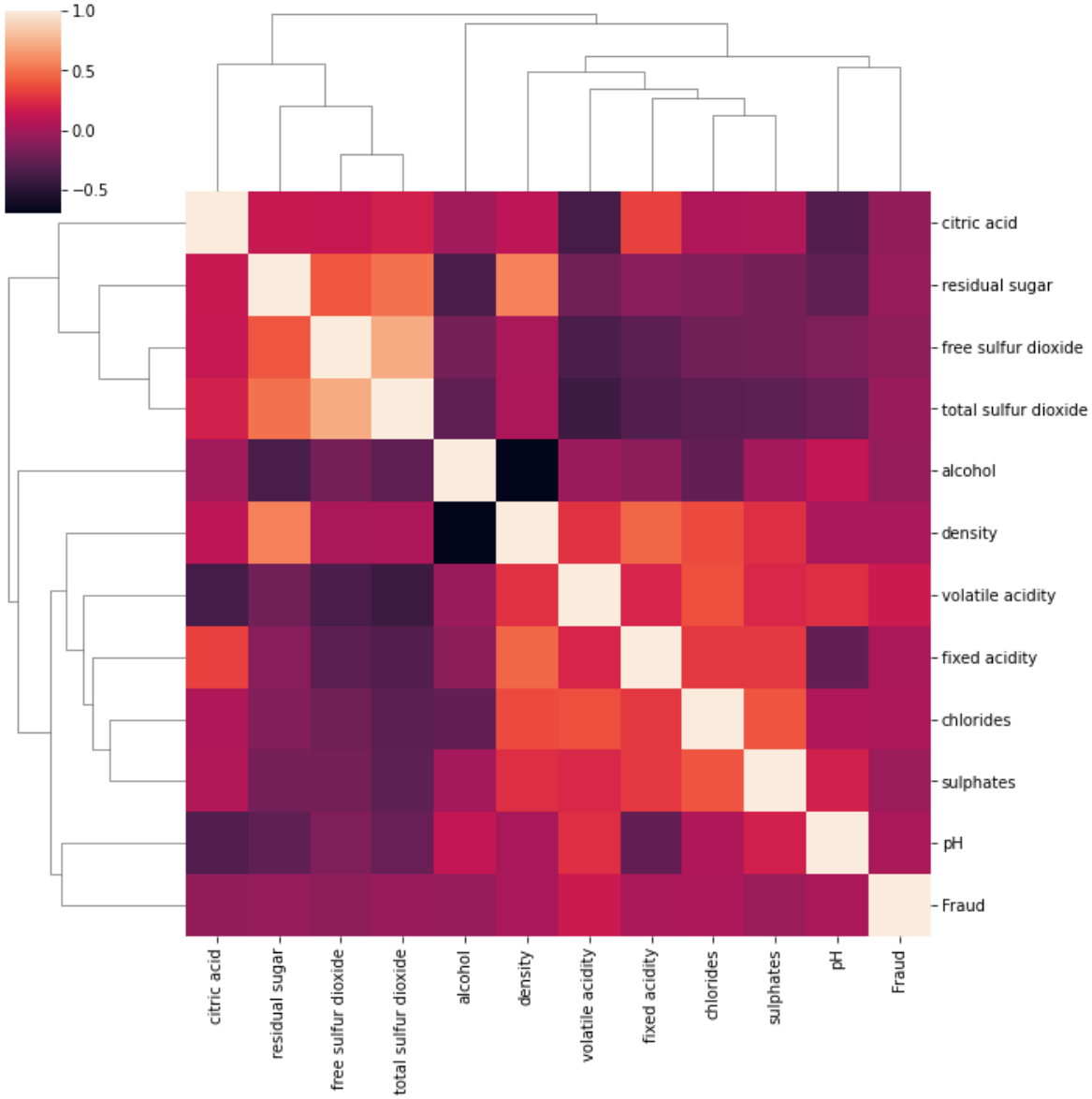
```
free sulfur dioxide    -0.085204
citric acid            -0.061789
alcohol                -0.051141
residual sugar         -0.048756
total sulfur dioxide   -0.035252
sulphates              -0.034046
density                 0.016351
pH                      0.020107
fixed acidity           0.021794
chlorides               0.034499
volatile acidity        0.151228
Fraud                   1.000000
Name: Fraud, dtype: float64
```

In [14]:

```
1 sns.clustermap(df.corr())
```

Out[14]:

```
<seaborn.matrix.ClusterGrid at 0x1dbd62f7e80>
```



In [15]:

```
1 df = df.drop('Fraud',axis = 1)
```

In [16]:

```
df['type'] = pd.get_dummies(df['type'],drop_first=True)
```

In [17]:

```
X = df.drop('quality',axis = 1)
```

In [18]:

```
y = df['quality']
```

In [19]:

```
from sklearn.model_selection import train_test_split
```

In [20]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.10, random_state=
```

In [21]:

```
from sklearn.preprocessing import StandardScaler
```

In [22]:

```
scaler = StandardScaler()
```

In [23]:

```
scale_X_train = scaler.fit_transform(X_train)
```

In [24]:

```
scale_X_test = scaler.transform(X_test)
```

In [25]:

```
from sklearn.svm import SVC
```

In [26]:

```
svc = SVC(class_weight='balanced')
```

In [27]:

```
from sklearn.model_selection import GridSearchCV
```

In [28]:

```
param_grid = {'C':[0.01,0.1,0.2,0.5,1,2,5],'kernel':['linear','rbf','sigmoid'],'degree'
```

In [29]:

```
grid = GridSearchCV(svc,param_grid)
```

In [30]:

```
grid.fit(scale_X_train,y_train)
```

Out[30]:

```
GridSearchCV(estimator=SVC(class_weight='balanced'),
             param_grid={'C': [0.01, 0.1, 0.2, 0.5, 1, 2, 5],
                         'degree': [1, 2, 3, 4, 5], 'gamma': ['scale', 'aut
o'],
                         'kernel': ['linear', 'rbf', 'sigmoid']})
```

In [31]:

```
grid.best_params_
```

Out[31]:

```
{'C': 5, 'degree': 1, 'gamma': 'auto', 'kernel': 'rbf'}
```

In [32]:

```
from sklearn.metrics import confusion_matrix,classification_report,plot_confusion_matri
```

In [33]:

```
preds = grid.predict(scale_X_test)
```

In [34]:

```
confusion_matrix(y_test,preds)
```

Out[34]:

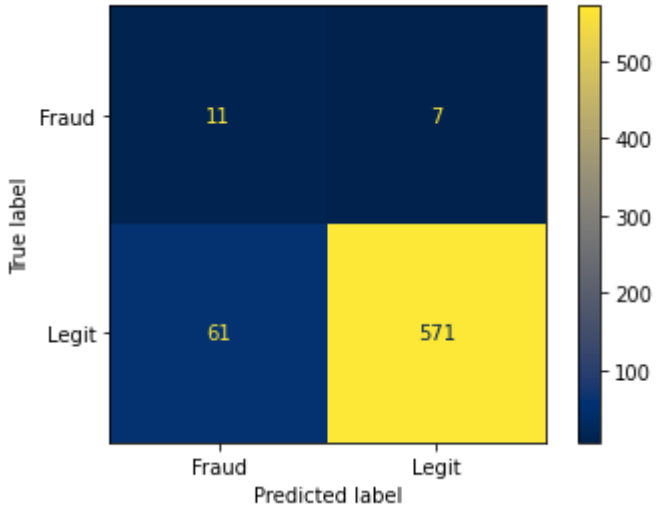
```
array([[ 11,   7],
       [ 61, 571]], dtype=int64)
```

In [35]:

```
1 plot_confusion_matrix(grid,scale_X_test,y_test,cmap='cividis')
```

Out[35]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x1dbd6348
f40>
```



In [ ]:

```
1
```