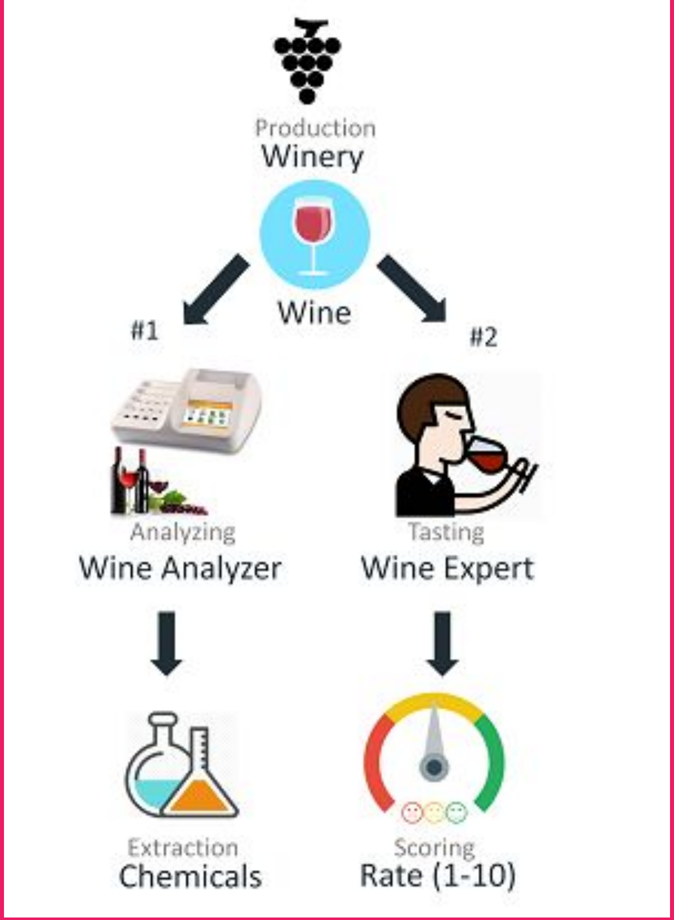
Wine Quality

-Sonya @AUG2021



Wine is an alcoholic drink that is made up of fermented grapes. How can you differentiate the wine according to their quality?

According to experts, the wine is differentiated according to its smell, flavor, and color, but how the ML to do this job?

About the project

---

1. Import & process dataset
2. Data Exploration
3. Model Test & Predict
   - Analysis using Linear Regression
   - Analysis using Classification

--- Endnote

Import & Process Data

What I do

- Get info
- Checking that missing values
- Get describe
- Check shape
- Drop duplicate
- Check unique values

```python
# raw data import

labels = ('fixed_acidity', 'volatile_acidity', 'citric_acid', 'residual_sugar', 'chlorides',
          'free_sulfur_dioxide', 'total_sulfur_dioxide', 'density', 'pH', 'sulphates', 'alcohol', 'quality')

df_ice = pd.read_csv('winequality-ice.csv', header=0, names=labels, sep=';')
df_fruit = pd.read_csv('winequality-fruit.csv',
                       header=0, names=labels, sep=';')
```

executed in 34ms, finished 15:36:51 2021-08-12

```python
df_ice.head()
```

executed in 33ms, finished 15:36:55 2021-08-12

	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
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

```python
df_fruit.head()
```

executed in 26ms, finished 15:37:00 2021-08-12

	fixed_acidity	volatile_acidity	citric_acid	residual_sugar	chlorides	free_sulfur_dioxide	total_sulfur_dioxide	density	pH	sulphates	alcohol	quality
0	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.0010	3.00	0.45	8.8	6
1	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.9940	3.30	0.49	9.5	6
2	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.9951	3.26	0.44	10.1	6
3	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6
4	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6

Desired dataframe

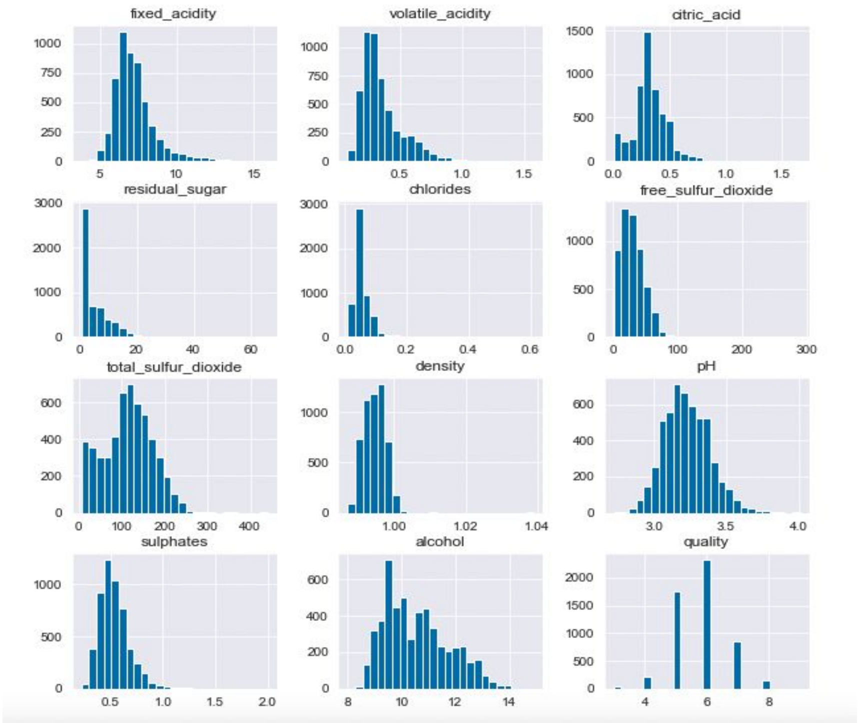
- Append the category it needed
- Append datasets
- Output for worklog

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5320 entries, 0 to 4897
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   fixed_acidity         5320 non-null   float64
 1   volatile_acidity      5320 non-null   float64
 2   citric_acid           5320 non-null   float64
 3   residual_sugar        5320 non-null   float64
 4   chlorides             5320 non-null   float64
 5   free_sulfur_dioxide   5320 non-null   float64
 6   total_sulfur_dioxide  5320 non-null   float64
 7   density               5320 non-null   float64
 8   pH                    5320 non-null   float64
 9   sulphates             5320 non-null   float64
 10  alcohol               5320 non-null   float64
 11  quality               5320 non-null   int64
 12  category              5320 non-null   object
dtypes: float64(11), int64(1), object(1)
memory usage: 741.9+ KB
```

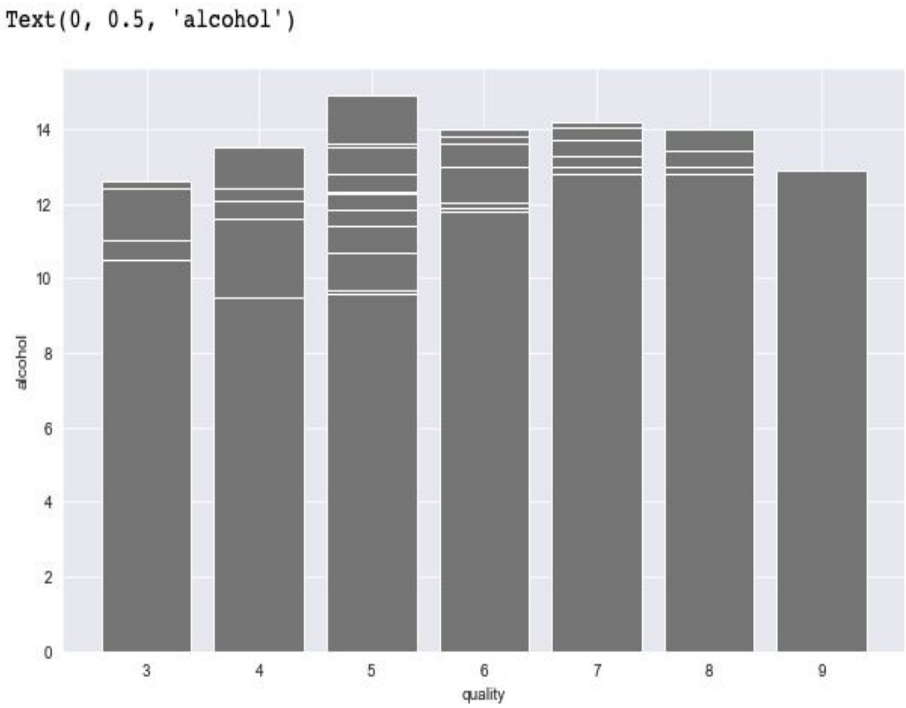
```python
# output the edited dataset and save for worklog

outputpath='/Users/sonyaWU/Desktop/wine_quality_1.xlsx'
wine_df.to_excel(outputpath,index=False,header=True)
```

executed in 839ms, finished 15:08:30 2021-08-11



Distribute on features

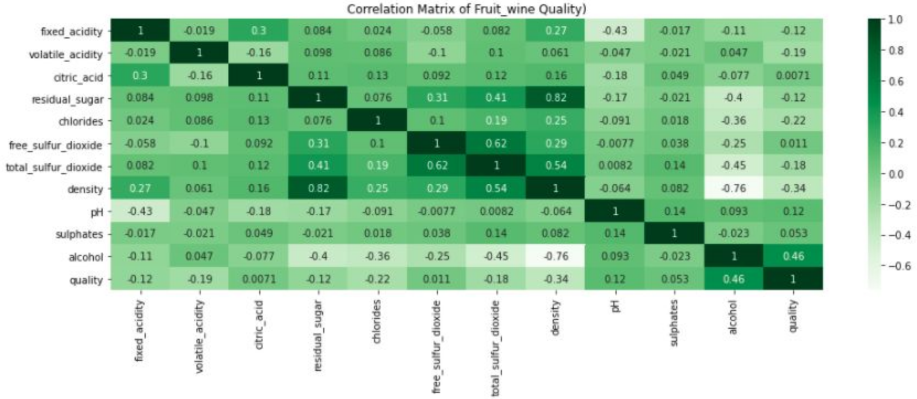


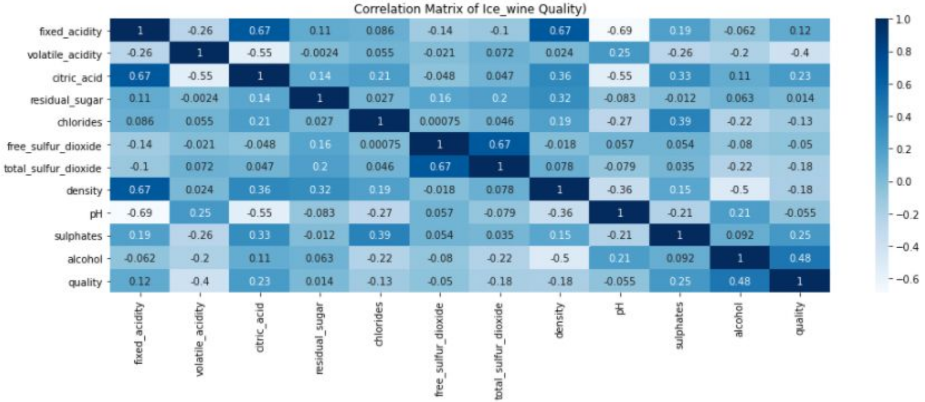
Value of alcohol make change

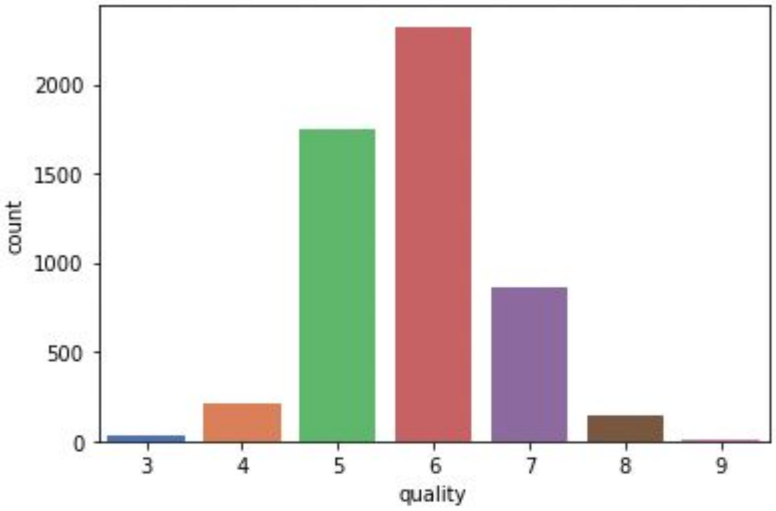
Exploration

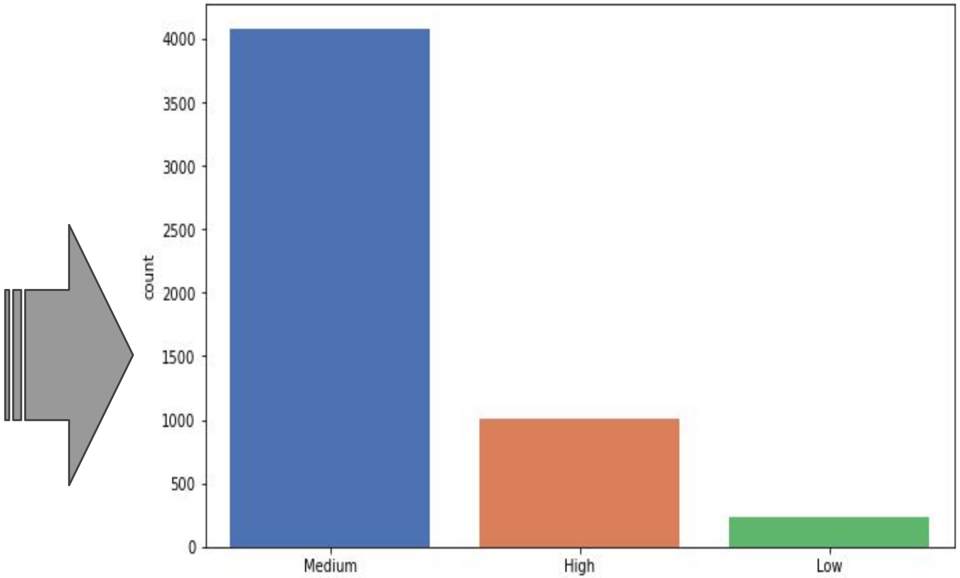
Correlation

Compare these three charts, we can see **Sulfur dioxide** is highly correlated with the Quality of a wine. And followed by **Density & Alcohol**.









Set quality standards

3,4 = Low

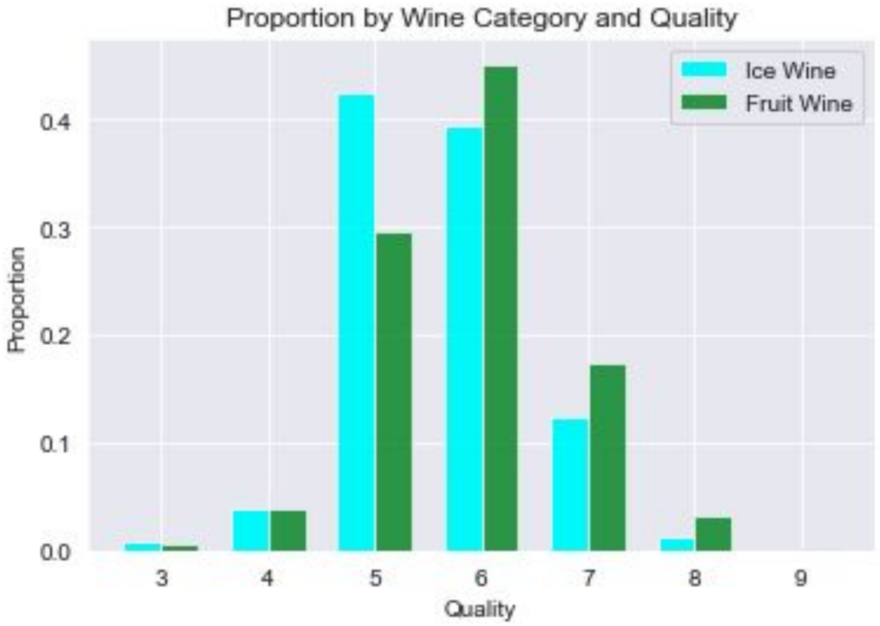
5,6 = Medium

7,8,9 = High

High:1009
Medium:4075
Low:236

Category & Quality Proportion

- Counts for each rating and category
- Total counts for each category
- Proportions by dividing rating counts by total of samples



Model Test & Predict

Linear Regression

Try in datasets for df_ice, or df_fruit, result comes difference.

The coefficients denote the impact of each on the quality of wine.

1. Every Alcohol measure increase will lead to increase of 0.33 in quality.
2. The Chlorides increasing will be decrease the quality.

In [190]:

```
print('Mean Absolute Error:', metrics.mean_absolute_error(test_pred, y_test))
print('Mean Squared Error:', metrics.mean_squared_error(test_pred, y_test))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(test_pred, y_test)))
```

executed in 4ms, finished 07:55:12 2021-08-13

```
Mean Absolute Error: 0.5327418793324455
Mean Squared Error: 0.44458329185943474
Root Mean Squared Error: 0.6667707940960182
```

In [191]:

```
coefficients = pd.DataFrame(lr.coef_, features)
coefficients.columns = ['Coefficient']
print(coefficients)
```

executed in 9ms, finished 07:55:14 2021-08-13

```
                      Coefficient
fixed_acidity            0.053091
volatile_acidity        -1.147530
citric_acid             -0.211109
chlorides               -1.292045
total_sulfur_dioxide    -0.001856
density                -21.321908
sulphates                0.876361
alcohol                  0.273638
```

Random Forest Classifier

It shows a massive increase in accuracy when we reduced the number of classes while still managing to retain meaning of the output.

```python
from sklearn.preprocessing import MinMaxScaler

# creating normalization object
norm = MinMaxScaler()

norm_fit = norm.fit(X_train)
new_xtrain = norm_fit.transform(X_train)
new_xtest = norm_fit.transform(X_test)


print(new_xtrain)
```

executed in 25ms, finished 17:25:44 2021-08-12

```
[[0.48672566 0.17948718 0.34       ... 0.49632893 0.19161677 0.33846154]
 [0.37168142 0.12820513 0.4        ... 0.44566814 0.2754491  0.52307692]
 [0.48672566 0.12820513 0.35       ... 0.48825257 0.2994012  0.43076923]
 ...
 [0.28318584 0.35897436 0.02       ... 0.49412628 0.14371257 0.16923077]
 [0.54867257 0.13675214 0.44       ... 0.61894273 0.26946108 0.24615385]
 [0.17699115 0.48290598 0.2        ... 0.56020558 0.1257485  0.12307692]]
```

4.2.2 Apply model

```python
# importing modules
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report
```

executed in 5ms, finished 17:34:12 2021-08-12

```python
f = RandomForestClassifier()
f.fit(X_train, y_train)
g = f.predict(X_test)
print("Accuracy for RandomForestClassifier:",
      metrics.accuracy_score(y_test, g))
```

executed in 213ms, finished 07:46:43 2021-08-13

```
Accuracy for RandomForestClassifier: 0.5588235294117647
```

Coming up later:

- Follow up using multiple modelling for classifications
- Random Forest Classifier gives the 88% accuracy

Thank you Mr.Terresiu for sharing data information in Lulu Island Winery.