Assignment 4

Name: Keshav Goyal

Roll no: 21BEC2297

Importing libraries

[ ] ↳ *5 cells hidden*

Checking NULL values

[ ] ↳ *1 cell hidden*

Outlier detection and removal

↳ *3 cells hidden*

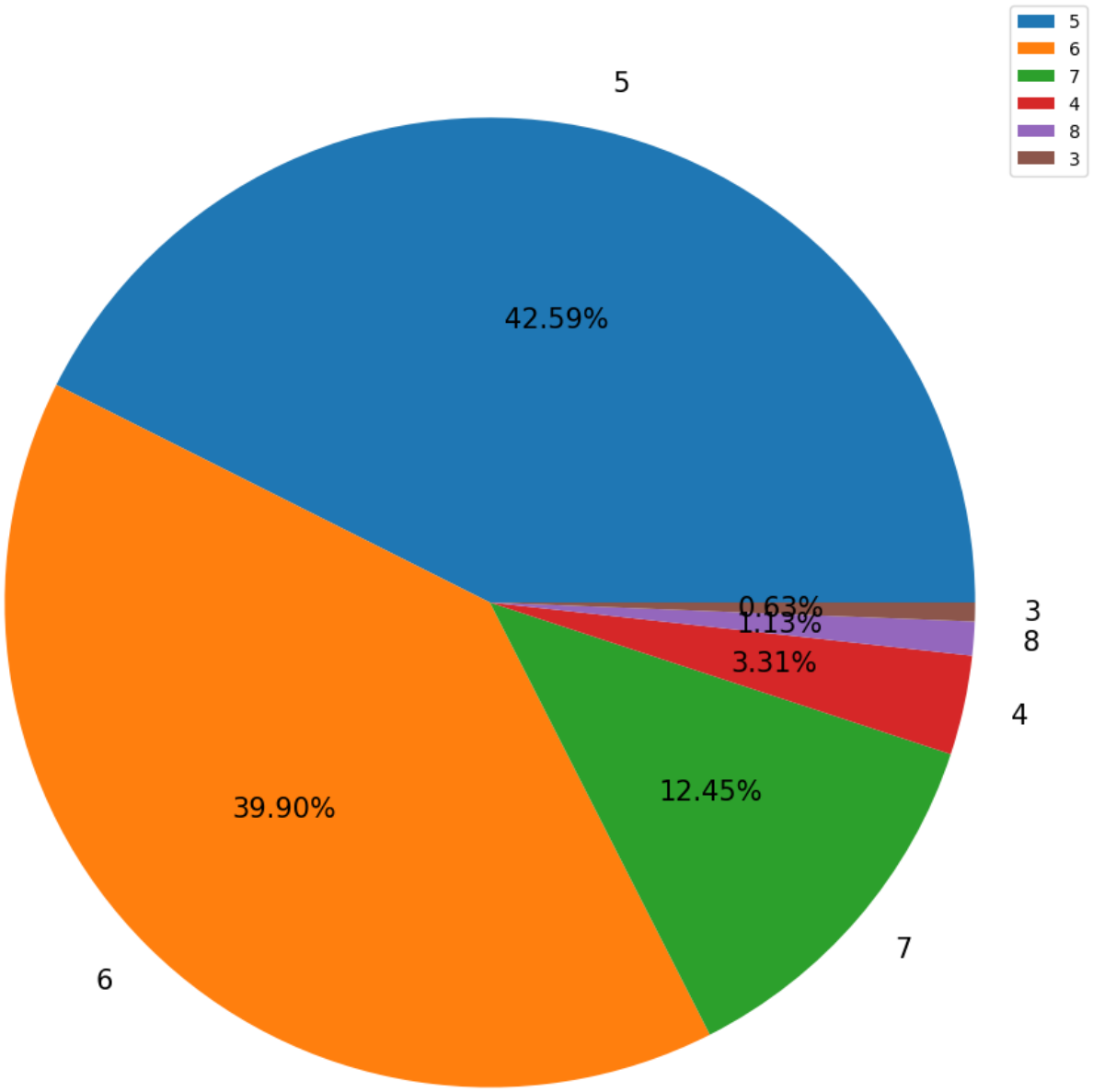
Correlation Heatmap

[ ] ↳ *2 cells hidden*

Visualizations

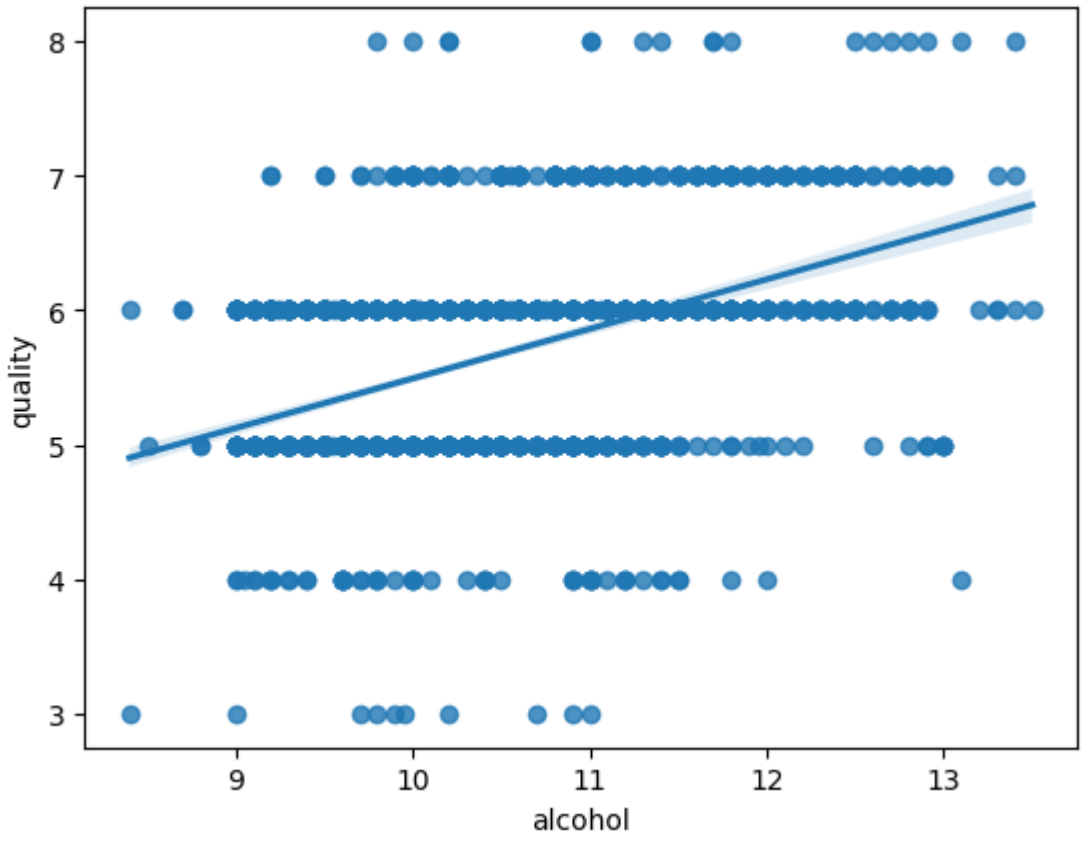
Univariate analysis

```
q_count=df['quality'].value_counts()
plt.figure(figsize=(12,12))
plt.title('Quality of Wine',fontdict={"fontsize":20})
plt.pie(q_count,labels=q_count.keys(),autopct='%.2f%%',textprops={"fontsize":15})
plt.legend()
plt.show()
```



Bivariate analysis

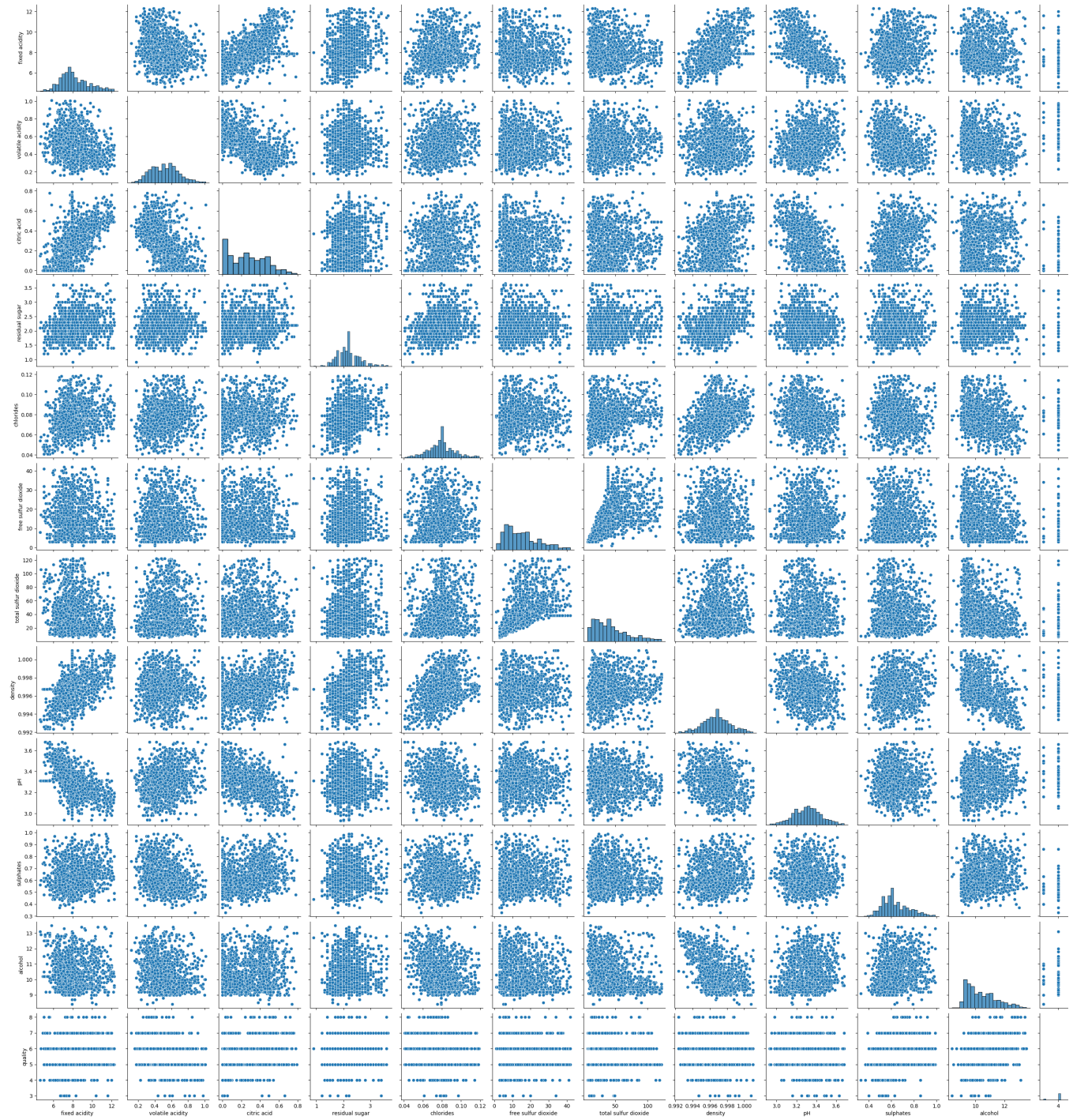
```
sns.regplot(x=df['alcohol'],y = df['quality'])
plt.show()
```



Multi-variate analysis

```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x7f08a0b8e800>
```



Label Encoding

```
def categorize(i):
    if(i>=3 and i<=5):
        return "Low"
    else:
        return "High"


df.quality=df.quality.apply(categorize)


from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
df.quality=le.fit_transform(df.quality)
df.head(10)
```

```
          fixed  volatile  citric  residual                    free    total
```

```
label_dict={0:le.inverse_transform([0])[0],1:le.inverse_transform([1])[0]}
label_dict
```

```
{0: 'High', 1: 'Low'}
```

```
df.quality.value_counts()
```

```
1    855
0    744
Name: quality, dtype: int64
```

Splitting scalling and balancing dataset

[ ] ↳ *7 cells hidden*

Model building

```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
```

```
model.fit(x_train, y_train)
```

```
RandomForestClassifier
RandomForestClassifier()
```

```
from sklearn.metrics import accuracy_score
x_test_prediction = model.predict(x_test)
test_data_accuracy = accuracy_score(x_test_prediction, y_test)
print('Accuracy : ', test_data_accuracy)
```

```
Accuracy :  0.85
```

```
input_data1 = (7.3,0.65,0.00,1.2,0.065,15.0,21.0,0.9946,3.39,0.47,10.0)
input_data2 = (7.3,0.98,0.05,2.1,0.061,20.0,49.0,0.99705,3.31,0.55,9.7)
# changing the input data to a numpy array
input_data_as_numpy_array1 = np.asarray(input_data1)
input_data_as_numpy_array2 = np.asarray(input_data2)
# reshape the data as we are predicting the label for only one instance
input_data_reshaped1 = input_data_as_numpy_array1.reshape(1,-1)
input_data_reshaped2 = input_data_as_numpy_array2.reshape(1,-1)
```

```
prediction1 = model.predict(input_data_reshaped1)
prediction2 = model.predict(input_data_reshaped2)
print(prediction1)
print(prediction2)

if (prediction1[0]==1):
  print('First Wine is a Good Quality Wine')
else:
  print('First Wine is a Bad Quality Wine')
if (prediction2[0]==1):
  print('Second Wine is a Good Quality Wine')
else:
  print('Second Wine is a Bad Quality Wine')
```

```
[1]
[0]
First Wine is a Good Quality Wine
Second Wine is a Bad Quality Wine
```

Test with Random observations

```
random_index=np.random.randint(0,x_test.shape[0])
random_index
```

```
64
```

```
random_parameters=x_test.iloc[random_index].values
random_parameters
```

```
array([0.35064935, 0.29213483, 0.26582278, 0.4       , 0.5       ,
       0.14634146, 0.25      , 0.43352601, 0.53333333, 0.21212121,
       0.21568627])
```

```
predicted_label=label_dict[model.predict([random_parameters])[0]]
true_label=label_dict[y_test.iloc[random_index]]
print('Predicted Label is:',predicted_label)
print('True Label is:',true_label)
```

```
Predicted Label is: High
True Label is: High
```