refactor_wine_quality_solution

June 9, 2020

1 Refactor: Wine Quality Analysis

In this exercise, you'll refactor code that analyzes a wine quality dataset taken from the UCI Machine Learning Repository here. Each row contains data on a wine sample, including several physicochemical properties gathered from tests, as well as a quality rating evaluated by wine experts.

The code in this notebook first renames the columns of the dataset and then calculates some statistics on how some features may be related to quality ratings. Can you refactor this code to make it more clean and modular?

In [3]:
```
import pandas as pd
df = pd.read_csv('winequality-red.csv', sep=';')
df.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
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

1.0.1 Renaming Columns

You want to replace the spaces in the column labels with underscores to be able to reference columns with dot notation. Here's one way you could've done it.

```
In [2]: df.columns = [label.replace(' ', '_') for label in df.columns]
        df.head()
```

Out[2]:

	fixed_acidity	volatile_acidity	citric_acid	residual_sugar	chlorides	\
0	7.4	0.70	0.00	1.9	0.076	
1	7.8	0.88	0.00	2.6	0.098	
2	7.8	0.76	0.04	2.3	0.092	
3	11.2	0.28	0.56	1.9	0.075	
4	7.4	0.70	0.00	1.9	0.076	

	free_sulfur_dioxide	total_sulfur_dioxide	density	pH	sulphates	\
0	11.0	34.0	0.9978	3.51	0.56	
1	25.0	67.0	0.9968	3.20	0.68	
2	15.0	54.0	0.9970	3.26	0.65	
3	17.0	60.0	0.9980	3.16	0.58	
4	11.0	34.0	0.9978	3.51	0.56	

	alcohol	quality
0	9.4	5
1	9.8	5
2	9.8	5
3	9.8	6
4	9.4	5

1.0.2 Analyzing Features

Now that your columns are ready, you want to see how different features of this dataset relate to the quality rating of the wine. A very simple way you could do this is by observing the mean quality rating for the top and bottom half of each feature. The code below does this for four features. It looks pretty repetitive right now. Can you make this more concise?

You might challenge yourself to figure out how to make this code more efficient! But you don't need to worry too much about efficiency right now - we will cover that more in the next section.

```
In [3]: def numeric_to_buckets(df, column_name):
            median = df[column_name].median()
            for i, val in enumerate(df[column_name]):
                if val >= median:
                    df.loc[i, column_name] = 'high'
                else:
                    df.loc[i, column_name] = 'low'

In [4]: for feature in df.columns[:-1]:
            numeric_to_buckets(df, feature)
            print(df.groupby(feature).quality.mean(), '\n')
```

```
fixed_acidity
high    5.726061
low     5.540052
Name: quality, dtype: float64

volatile_acidity
high    5.392157
low     5.890166
Name: quality, dtype: float64

citric_acid
high    5.822360
low     5.447103
Name: quality, dtype: float64

residual_sugar
high    5.665880
low     5.602394
Name: quality, dtype: float64

chlorides
high    5.507194
low     5.776471
Name: quality, dtype: float64

free_sulfur_dioxide
high    5.595268
low     5.677136
Name: quality, dtype: float64

total_sulfur_dioxide
high    5.522981
low     5.750630
Name: quality, dtype: float64

density
high    5.540574
low     5.731830
Name: quality, dtype: float64

pH
high    5.598039
low     5.675607
Name: quality, dtype: float64

sulphates
high    5.898917
low     5.351562
```

```
Name: quality, dtype: float64
```

```
alcohol
high    5.958904
low     5.310302
Name: quality, dtype: float64
```

In [ ]: