refactor_wine_quality

November 23, 2019

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 [49]:
```
import pandas as pd
df = pd.read_csv('winequality-red.csv', sep=';')
df.head()
```

Out[49]:

	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 [50]: new_df = df.rename(columns={'fixed acidity': 'fixed_acidity',
                                     'volatile acidity': 'volatile_acidity',
                                     'citric acid': 'citric_acid',
                                     'residual sugar': 'residual_sugar',
                                     'free sulfur dioxide': 'free_sulfur_dioxide',
                                     'total sulfur dioxide': 'total_sulfur_dioxide'
                                    })
         new_df.head()
```

Out[50]:

	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

And here's a slightly better way you could do it. You can avoid making naming errors due to typos caused by manual typing. However, this looks a little repetitive. Can you make it better?

```
In [51]: labels = list(df.columns)
         i = 0
         for each in labels:
             labels[i] = each.replace(' ', '_')
             i+=1
         df.columns = labels

         df.head()
```

Out[51]:

	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 [52]: def groupby_quality(df_col,col_name):
             median = df_col.median()
             for i, col in enumerate(df_col):
                 if col >= median:
                     df.loc[i,col_name] = 'high'
                 else:
                     df.loc[i,col_name] = 'low'


In [53]: groupby_quality(df.alcohol,'alcohol')
         df.groupby('alcohol').quality.mean()

Out[53]: alcohol
         high    5.958904
         low     5.310302
         Name: quality, dtype: float64

In [54]: groupby_quality(df.pH,'pH')
         df.groupby('pH').quality.mean()

Out[54]: pH
         high    5.598039
```

```
low     5.675607
Name: quality, dtype: float64
```

In [55]:
```
groupby_quality(df.residual_sugar,'residual_sugar')
df.groupby('residual_sugar').quality.mean()
```

Out[55]:
```
residual_sugar
high    5.665880
low     5.602394
Name: quality, dtype: float64
```

In [56]:
```
groupby_quality(df.citric_acid,'citric_acid')
df.groupby('citric_acid').quality.mean()
```

Out[56]:
```
citric_acid
high    5.822360
low     5.447103
Name: quality, dtype: float64
```

In [ ]: