Dropping Features

Drop features from the dataset that have too many missing data values.

Chapter Goals:

- Figure out exactly how many missing values are in each feature
- Drop the features that contain too many missing values

A. Counting the missing values

In the previous chapter, we figured out that each of the `'MarkDown'` features, along with the `'CPI'` and `'Unemployment'` features contained missing values. We now want to figure out how many missing values each of these features has, i.e. how many rows of the combined feature DataFrame don't contain a value for the particular feature.

This can be done by counting the number of `True` values for each feature's column in the boolean DataFrame.

```
print(len(na_values))
print(sum(na_values['MarkDown1']))
print(sum(na_values['CPI']))
```

Counting the number of missing values in the 'MarkDown1' and 'CPI' features. The na_values boolean DataFrame represents the locations of missing data.

Since each feature's column contains `True` (equivalent to 1) or `False` (equivalent to 0), we just take the column's sum to count the number of `True`, i.e. missing values.

B. Dropping unusable features

The number of missing values in the `'MarkDown'` features are 4158, 5269, 4577, 4726, and 4140 respectively. Since each of the `'MarkDown'` feature values is missing in over half DataFrame's rows, we'll consider these features unusable and therefore drop them from the dataset.

```
markdowns = [
    'MarkDown1',
    'MarkDown2',
    'MarkDown3',
    'MarkDown4',
    'MarkDown5'
]
merged_features = merged_features.drop(columns=markdowns)
print(merged_features.columns.tolist())
```

Dropping the 'MarkDown' features from the DataFrame containing all the features.

Both the `'CPI'` and `'Unemployment'` features contain only 585 missing values. This is significantly less than the total number of rows in the DataFrame (8190), so we can still use these features. We'll discuss how to deal with the missing values in the next chapter.