4.3.3

Handling Duplicated Data

It's possible for data collection processes to introduce both missing and duplicated data. Although legitimate reasons might exist for keeping the duplicated data, it's more common to drop the duplicates. That's because keeping duplicated data might either skew data sums and averages by inflating numbers or create other problems during an analysis. So in this section, you'll learn how to identify and drop duplicated data by using Pandas functions.

Identify Any Duplicates

Pandas offers the `duplicated` function for identifying duplicated rows in a DataFrame.

Let's try it on a DataFrame that's named `best_actors_df` and that has duplicated rows, as the following code shows:

```
import pandas as pd
best_actors_df = pd.DataFrame({
    "Best Actors": ["Nic Cage", "Nic Cage", "Harrison Ford"]
})
best_actors_df

best_actors_df.duplicated()
```

In the preceding code, notice that the DataFrame has duplicated data: two "Nic Cage" rows. When we run the `duplicated` function, we get the output that the following image shows:

```
import pandas as pd

best_actors_df = pd.DataFrame({
    "Best Actors": ["Nic Cage", "Nic Cage", "Harrison Ford"]
})

best_actors_df
```

	Best Actors
0	Nic Cage
1	Nic Cage
2	Harrison Ford

```
best_actors_df.duplicated()
```

```
0    False
1     True
2    False
dtype: bool
```

In the preceding image, notice that the output is a DataFrame of three rows that has the following values: `False`, `True`, `False`. The `duplicated` function replaced one of the "Nic Cage" rows with a value of `True`—identifying that entry as duplicated.

IMPORTANT

To indicate duplicated entries, Pandas marks the first entry as `False` and any copies as `True`.

We can choose to drop either row, but only one is marked as `True`. This happens in case we want to use the `sum` function to count the duplicated data. Let's try that now with the following code:

```
best_actors_df.duplicated().sum()
```

Running the preceding code produces the number 1, because only one duplicated value exists.

```
best_actors_df.duplicated().sum()
```

```
1
```

Now that we've identified the duplicated values, the next step is to drop them.

Drop the Duplicates

The `drop_duplicates` function removes duplicated rows. We can run this function against either a DataFrame or a Series. The following code shows how to run this function on the `best_actors_df` DataFrame:

```
best_actors_df = best_actors_df.drop_duplicates()
best_actors_df
```

The preceding code results in the output that the following image shows:

```
best_actors_df = best_actors_df.drop_duplicates()
best_actors_df
```

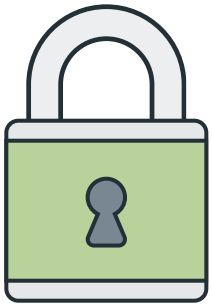
	Best Actors
0	Nic Cage
2	Harrison Ford

Nice! The code removed the duplicated “Nic Cage” row, making the data unique again and ensuring that calculations won’t be skewed.

IMPORTANT

Using `duplicated` and `drop_duplicates` provides a quick and straightforward way to handle duplicated data when preparing for analysis.

Now, before moving on, check your knowledge with the following assessment:



Access Denied

You don't have access to view this resource.

Your data keeps getting cleaner! And now that you've learned how to handle duplicated data, you'll next learn how to handle additional dirty data.