Microsoft: DAT210x Programming with Python for Data Science

Bookmarks

Bookmark

Pandas will automatically attempt to figure out the best data type to use for each series in your dataset. Most of the time it does this flawlessly, but other times it fails horribly! Particularly the `.read_html()` method is notorious for defaulting all series data types to Python objects. You should check, and double-check the actual type of each column in your dataset to avoid unwanted surprises:

```
>>> df.dtypes

Date        object
Name        object
Gender      object
Height      object
Weight      object
Age         object
Job         object
```

If your data types don't look the way you expected them, explicitly convert them to the desired type using the `.to_datetime()`, `.to_numeric()`, and `.to_timedelta()` methods:

```
>>> df.Date = pd.to_datetime(df.Date, errors='coerce')
>>> df.Height = pd.to_numeric(df.Height, errors='coerce')
>>> df.Weight = pd.to_numeric(df.Weight, errors='coerce')
>>> df.Age = pd.to_numeric(df.Age, errors='coerce')
>>> df.dtypes

Date         datetime64
Name         object
Gender       object
Height       float64
Weight       float64
Age          int64
Job          object
```

Take note how to_numeric properly converts to decimal or integer depending on the data it finds. The `errors=`'coerce' parameter instructs Pandas to enter a NaN at any field where the conversion fails.

After fixing up your data types, let's say you want to see all the unique values present in a particular series. Call the `.unique()` method on it to view a list, or alternatively, if you'd like to know how many times each of those unique values are present, you can call `.value_counts()`. Either method works with series, but neither will function if called on a dataframe:

```
>>> df.Age.unique()

array([7, 33, 27, 40, 22], dtype=int64)
```

```
>>> df.Age.value_counts()

7       1
22      5
27      1
33      2
40      2
dtype: int64
```

There are many other possible data munging and wrangling tasks, many of which can be applied easily and generically to any dataset. We've referenced a site detailing almost 40 such operations for you to further explore in the Dive Deeper section. However, some wrangling tasks require you look closer at your data. For instance, if you survey users with a series of 1-10 ranked questions, and a user enters all 5's or all 1's, chances are they were not being completely honest. Another example would be a user entering in January 1, 1970 as their birthdate since you required they enter in *something* but they did not want to disclose the information. In order to further improve the accuracy of your datasets, always be on the lookout for these sorts of issues.