**5 Methods for Filtering Strings with Python Pandas**

**Explained with examples**



(Photo by [Neven Krcmarek](https://unsplash.com/@nevenkrcmarek?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText) on [Unsplash](https://unsplash.com/s/photos/hand-writing?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText))

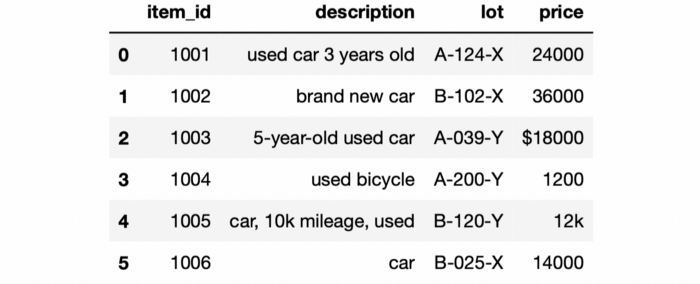
We often deal with textual data, which requires much more cleaning than numeric data. In order to extract usable and informative insights from textual data, we usually need to perform several steps of preprocessing and filtering.

Pandas library has lots of functions and methods that make working with textual data…

1. Containing a sequence of characters
2. The length of a string
3. Starting or ending with a specific sequence of characters
4. Characters being numeric or alphanumeric
5. Number of occurrences of a specific sequence of characters

Let’s start with creating a DataFrame from a csv file.

import pandas as pddf = pd.read\_csv("example.csv")df

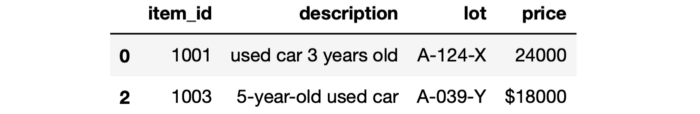


df (image by author)

The DataFrame contains 6 rows and 4 columns. We will use different methods to filter rows in this DataFrame. These methods are available via the str accessor of Pandas.

The first filtering operation we will do is to check if a string contains a particular word or a sequence of characters. We can use the contains method for this task. The following line of code finds the rows in which the description field contains “used car”.

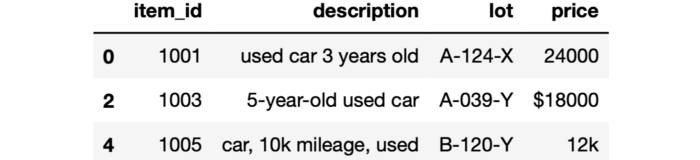
df[df["description"].str.contains("used car")]



(image by author)

There is actually another row for a used car but the description does not contain the “used car” expression. In order to find all the used cars in this DataFrame, we can look for the words “used” and “car” separately.

df[df["description"].str.contains("used") &  
 df["description"].str.contains("car")]



(image by author)

The last row contains both “car” and “used” but not together.

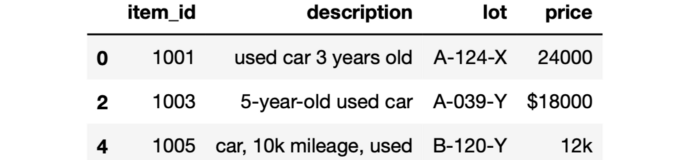
We can also filter based on the length of a string. Let’s say we are only interested in the description that are longer than 15 characters. We can do this operation by using the built-in len function as follows:

df[df["description"].apply(lambda x: len(x) > 15)]

We write a lambda expression that includes checking the length with the len function and apply it to each row in the description column. A more practical and efficient way of doing this operation is to use the len method via the str accessor.

df[df["description"].str.len() > 15]

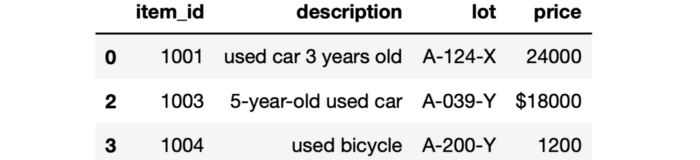
The output will be the same:



(image by author)

We can filter based on the first or last letter of a string using the startswith and endswith methods, respectively.

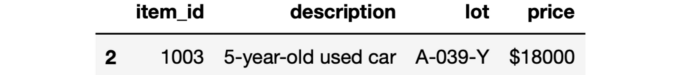
df[df["lot"].str.startswith("A")]



(image by author)

These methods are able to check the first n characters as well. For instance, we can select rows in which the lot value starts with ‘A-0’:

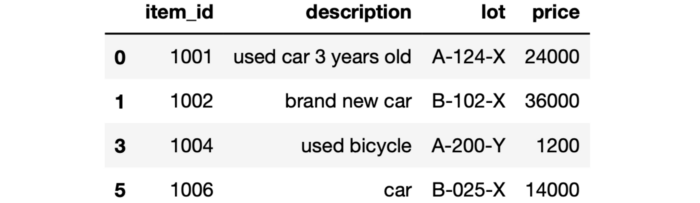
df[df["lot"].str.startswith("A-0")]



(image by author)

Python has some built-in string functions, which can be used for filtering string values in Pandas DataFrames. For instance, in the price column, there are some non-numeric characters such as $ and k. We can filter these out using the isnumeric function.

df[df["price"].apply(lambda x: x.isnumeric()==True)]



(image by author)

If we are looking for alphanumeric characters (i.e. only letters and numbers), we can use the isalphanum function.

The count method counts the number of occurrences of a single character or a sequence of characters. This might come in handy for filtering in some cases. For instance, you may be looking for strings that do not include a sequence of characters more than once.

We can count the word “used” in the description column as follows:

df["description"].str.count("used")**# output**  
0 1  
1 0  
2 1  
3 1  
4 1  
5 0  
Name: description, dtype: int64

If you want to use this for filtering, just compare it to a value as follows:

df[df["description"].str.count("used") < 1]

We have covered 5 different ways of Pandas DataFrames based on string values. We tend to give more importance to numeric data but textual data is just as important and carries lots of valuable information. Therefore, being able to clean and preprocess textual data is of crucial importance for data analysis and modeling.