Learn How to (easily!!) do 3 Advanced Excel Tasks in Python

An Introduction to Pandas for Excel Power Users

Nik Piepenbreier Mar 15, 2020 · 5 min read ★

The face-off begins! (Source: Nik Piepenbreier)

regarded as a bit more challenging to pick up, but as having limitless potential. In this post, we'll explore three things you can easily do in Python that you normally do in Excel!

We'll begin by import pandas and loading two dataframes based on the sheets available in our workbook. We'll call them *sales* and *states*.

```
1  import pandas as pd
2  sales = pd.read_excel('https://github.com/datagy/mediumdata/raw/master/pythonexcel.xlsx', sheet_n
3  states = pd.read_excel('https://github.com/datagy/mediumdata/raw/master/pythonexcel.xlsx', sheet_
```

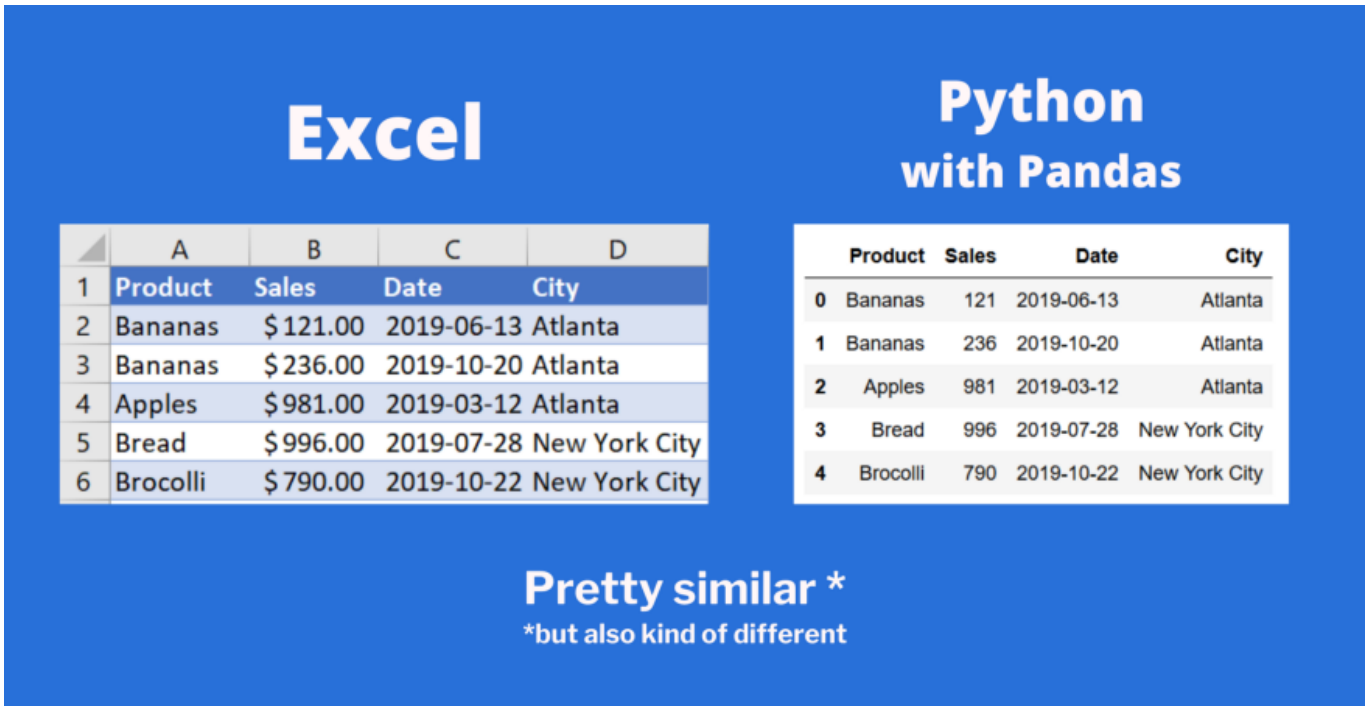
datagy-python-excel-01.py hosted with ♡ by GitHub view raw

Importing our datasets into Pandas dataframes.

Let's image we ran the .head() method on the dataframes, as sampled below:

```
print(sales.head())
```

We can compare this to how the data would look in Excel:



We can see the data that's displayed is relatively similar to how Excel would display the data, but there are some key differences:

- Excel starts at Row 1, while Pandas starts at row ('index') 0,
- Excel labels columns with letters starting at A, while Pandas labels the columns with variable names

Let's begin to dive into how to work with Pandas to complete Excel tasks.

. . .

IF Functions in Python

Using IF functions in Excel is pretty handy and allows us to apply a certain label based on a condition in another cell. Let's say we wanted to create a new column that lets us know if the value in a cell in column B is greater than 500. In Excel, we'd label Column E MoreThan500 and we'd go into Cell E2 and write in:

```
=IF([@Sales]>500, "Yes", "No")
```

	A	B	C	D	E	F	G
1	Product	Sales	Date	City	MoreThan500		
2	Bananas	$121.00	2019-06-13	Atlanta			
3	Bananas	$236.00	2019-10-20	Atlanta			
4	Apples	$981.00	2019-03-12	Atlanta			
5	Bread	$996.00	2019-07-28	New York City			
6	Brocolli	$790.00	2019-10-22	New York City			
7	Apples	$762.00	2019-11-03	Toronto			
8	Bananas	$870.00	2019-11-18	New York City			

Applying an IF Function in Excel (Source: Nik Piepenbreier)

If we wanted to do this in Pandas, we could use list comprehensions to easily apply the same if statement:

List comprehensions explained (Source: Nik Piepenbreier)

List comprehensions are great tools for this kind of work, which reduces the need to write complex if/else statements. You could accomplish the same thing with an if/else statement, but this saves time and make the code a little cleaner. You can learn more about list comprehensions in detail by checking out this article.

Want to learn more about list comprehensions? Check out my video!

Python List Comprehensions | Python Tutorial | Why and How to Use The...

List Comprehensions in Python. Source: Nik Piepenbreier

. . .

VLOOKUP in Pandas

In our dataset we have cities on one sheet and states/provinces on another. This isn't ideal, but we can use VLOOKUP in Excel to link the data. A VLOOKUP works similarly to a left join, where every record in the left dataset is retained. We tell Excel to look vertically up and down a column for a specific value in a lookup table and then return a value that sits a certain number of columns to the right of it.

Let's add a column called "State" and use VLOOKUP to return the corresponding state from the *states* table.

	A	B	C	D	E	F	G	H	I
1	Product	Sales	Date	City	MoreThan500	State			
2	Bananas	$121.00	2019-06-13	Atlanta	No				
3	Bananas	$236.00	2019-10-20	Atlanta	No				
4	Apples	$981.00	2019-03-12	Atlanta	Yes				
5	Bread	$996.00	2019-07-28	New York City	Yes				
6	Brocolli	$790.00	2019-10-22	New York City	Yes				
7	Apples	$762.00	2019-11-03	Toronto	Yes				
8	Bananas	$870.00	2019-11-18	New York City	Yes				
9	Bananas	$852.00	2019-03-21	Atlanta	Yes				
10	Apples	$427.00	2019-05-11	Toronto	No				
11	Bread	$576.00	2019-09-14	Atlanta	Yes				

Using VLOOKUP to add state/province information (Source: Nik Piepenbreier)

In Python, we can accomplish the same thing using the Pandas *merge* function. Merge takes two dataframes and merges them. To accomplish this, we would write the following code:

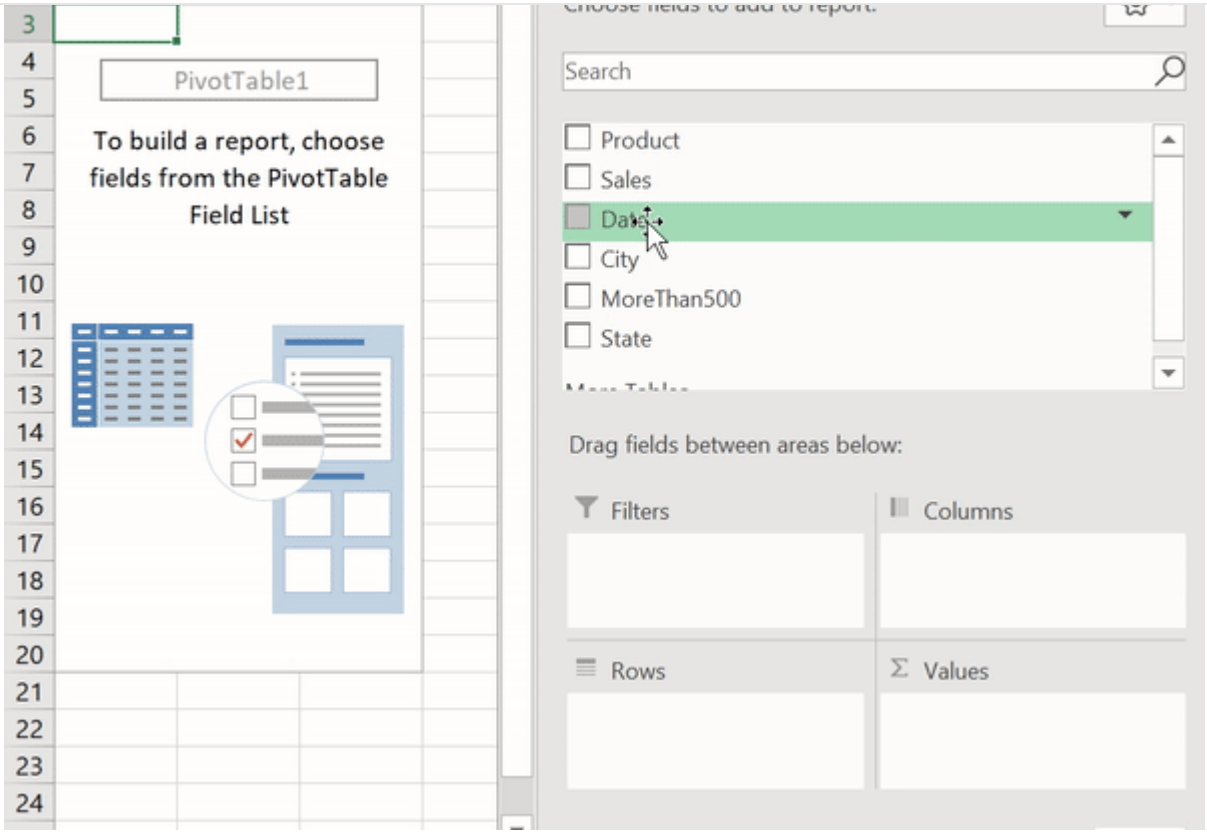
Let's break this down argument by argument:

1. The first argument is the original dataframe
2. The second argument is the dataframe we're looking up values in
3. How specifies the type of join we want to make
4. On specifies the variable that we want to merge on (there's also left_on and right_on if the variables are called different things in each dataframe).

. . .

Pivot Tables in Pandas

Pivot tables are one of Excel's most powerful features — they allow us to extract meaningful data about large datasets incredibly quickly. Let's create a pivot table on the sum of sales per city.



Generating a pivot table in Excel (Source: Nik Piepenbreier)

To do this, we simply drag the City field into the Rows section and the Sales field into the Values section. Automatically, Excel totals up the sales for each of the cities in our dataset.

To generate the same pivot table in Pandas, we would write the following code:

```
sales.pivot_table(index = 'City', values = 'Sales', aggfunc = 'sum')
```

Let's break this down again:

1. We use sales.pivot_table to let Pandas know we want to create a pivot table based on the sales dataframe
2. Index specifies the values we want to aggregate by
3. Values specifies the values we want to aggregate

. . .

Wrapping Up

In this article, we learned how to import Excel data into Pandas, how to complete IF and VLOOKUP functions, and how to generate pivot tables. But you might be asking yourself, why use Pandas, if you can do everything listed here in Excel? There isn't a clear answer here. Python lets us generate re-usable, traceable code that lets us easily replicate our analytical design. Excel might just be enough for smaller analyses. I encourage you to give Pandas a shot to see if it grows on you.

Python Data Analysis Excel Pandas Data Science