DATA SCIENCE EAST AFRICA

INTRODUCTIONTO PANDAS

Day 6/20

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license.

The pandas package is the most important tool at the disposal of Data Scientists and Analysts working in Python today. The powerful machine learning and glamorous visualization tools may get all the attention, but pandas is the backbone of most data projects.

This tool is essentially your data’s home. Through pandas, you get acquainted with your data by cleaning, transforming, and analyzing it.

For example, say you want to explore a dataset stored in a CSV on your computer.Pandas will extract the data from that CSV into a DataFrame — a table, basically — then let you do things like:

- Calculate statistics and answer questions about the data, like:
  - What's the average, median, max, or min of each column?
  - Does column A correlate with column B?
  - What does the distribution of data in column C look like?

- Clean the data by doing things like removing missing values and filtering rows or columns by some criteria.

- Visualize the data with help from Matplotlib. Plot bars, lines, histograms, bubbles, and more.

- Store the cleaned, transformed data back into a CSV, other file or database.

Before you jump into the modeling or the complex visualizations you need to have a good understanding of the nature of your dataset and pandas is the best avenue through which to do that.

How does pandas fit into the data science toolkit?

Not only is the pandas library a central component of the data science toolkit but it is used in conjunction with other libraries in that collection. Pandas is built on top of the NumPy package, meaning a lot of the structure of NumPy is used or replicated in Pandas. Data in pandas is often used to feed statistical analysis in SciPy, plotting functions from Matplotlib, and machine learning algorithms in Scikit–learn.

Jupyter Notebooks offer a good environment for using pandas to do data exploration and modeling, but pandas can also be used in text editors just as easily.

Jupyter Notebooks give us the ability to execute code in a particular cell as opposed to running the entire file. This saves a lot of time when working with large datasets and complex transformations. Notebooks also provide an easy way to visualize pandas' DataFrames and plots. As a matter of fact, this article was created entirely in a Jupyter Notebook.

When should you start using pandas?

If you do not have any experience coding in Python, then you should stay away from learning pandas until you do. You don't have to be at the level of the software engineer, but you should be adept at the basics, such as lists, tuples, dictionaries, functions, and iterations. Also, I'd also recommend familiarizing yourself with NumPy due to the similarities mentioned above. Even though accelerated programs teach you pandas, better skills beforehand means you'll be able to maximize time for learning and mastering the more complicated material.

Pandas First Steps

Install and import . . .

Pandas is an easy package to install. Open up your terminal program (for Mac users) or command line (for PC users) and install it using either of the following commands:

```
conda install pandas
```

OR

```
pip install pandas
```

Alternatively, if you're currently viewing this article in a Jupyter notebook you can run this cell :

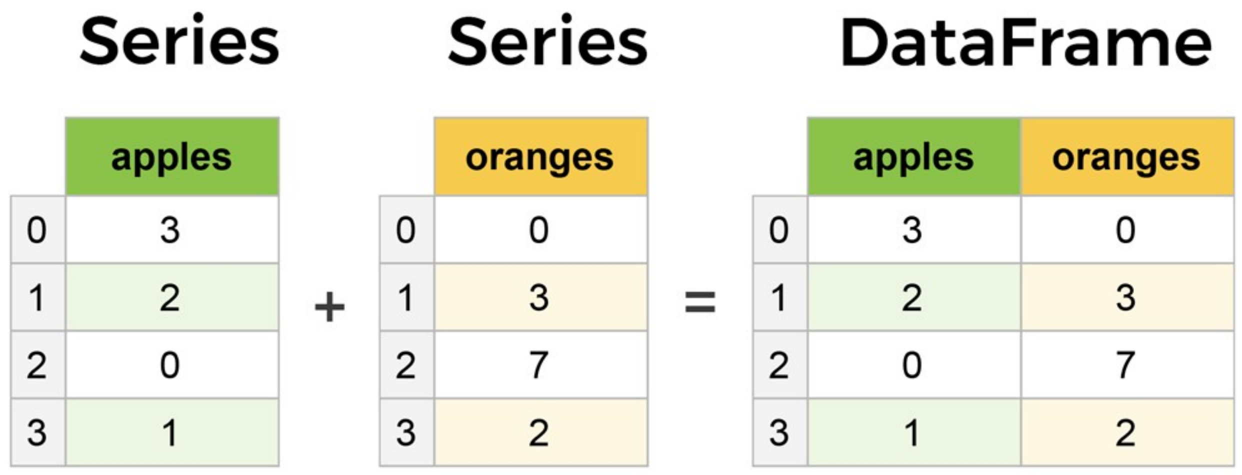
```
!pip install pandas
```

The ! at the beginning runs cells as if they were in a terminal. To import pandas we usually import it with a shorter name since it's used so much :

```
Import pandas as pd
```

Core components of pandas: Series and DataFrames

The primary two components of pandas are series and DataFrame. A series is essentially a column, and a DataFrame is a multi-dimensional table made up of a collection of Series.



	apples
0	3
1	2
2	0
3	1

	oranges
0	0
1	3
2	7
3	2

	apples	oranges
0	3	0
1	2	3
2	0	7
3	1	2

DataFrames and Series are quite similar in that many operations that you can do with one you can do with the other, such as filling in null values and calculating the mean.

Creating DataFrames from scratch

Creating DataFrames right in Python is good to know and quite useful when testing new methods and functions you find in the pandas docs. There are *many* ways to create a DataFrame from scratch, but a great option is to just use a simple dict.

Let’s say we have a fruit stand that sells apples and oranges. We want to have a column for each fruit and a row for each customer purchase.

To organize this as a dictionary for pandas we could do something like:

```
data = {
   "apples' : [3,2,0,1]
   "oranges" :[0,3,7,2]
}
```

And then pass it to the pandas DataFrame constructor :

```
purchases = pd.DataFrame(data)
purchases
```

Run the above code and see what will be your out put.

Exercise:

- Learn how to install Anaconda

- Learn how to Read json file and csv file pandas

- Read more on series and DataFrame.

You should be able to:

1). Know what kind of Data pandas handle.
2). Know how to read and write tabular data.
3). Know how to select a subset of a table.
4). Know how to create plots in Pandas.
5 Know how to create new columns derived from existing columns?
6). How to calculate summary statistics.
7). How to reshape the layout of tables.
8). Know how to combine data from multiple tables.
9). Know how to handle time series data.
10). Know how to manipulate textual data.

Best Wishes,
Regards Data Science East Africa.