DOWNLOAD PDF PANDAS FOR DATA ANALYSIS

Chapter 1 : Python Data Analysis Library â€” pandas: Python Data Analysis Library

Python Data Analysis LibraryÂ¶ pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. pandas is a NumFOCUS sponsored project.

Data manipulation, visualization, and analysis with for Python You should now be able to follow along with this series using either Python 2 or Python 3. If you are having any trouble, comment on the video or shoot me an email for help. The Pandas module is a massive collaboration of many modules along with some unique features to make a very powerful module. Pandas is great for data manipulation, data analysis, and data visualization. The Pandas modules uses objects to allow for data analysis at a fairly high performance rate in comparison to typical Python procedures. With it, we can easily read and write from and to CSV files, or even databases. From there, we can manipulate the data by columns, create new columns, and even base the new columns on other column data. Next, we can progress into data visualization using Matplotlib. Matplotlib is a great module even without the teamwork of Pandas, but Pandas comes in and makes intuitive graphing with Matplotlib a breeze. In this series, I would like to walk you through the basics of the Pandas module, to show you all of the things you can do with it. At the end, I will even show you how to overcome any instance where Pandas does not do the operation you are looking for, using function mapping. The series is both text and video, with example code to go along with it. You can choose to only use one form, or both. Before we begin, you will need to not only download Pandas, but you will also need all the many dependencies. As I said before, Pandas makes use of a sort of collaboration between many modules, as well as adding some of its own code into the mix. This means we need quite a few others. I find the easiest set up to be the following website: If you are not using Windows, then that website is of less use to you, but, then again, installing modules on Mac OS and Linux is usually quite simple! When on the website, download all of the dependencies listed. If you are not using that website to download Pandas or the dependencies, here is a list of the packages that you will definitely need: It should be as easy as: It would be wise to get a general understanding about the terms used in Pandas, like what is a series, dataframe, indexing, slicing Series - A series is a one-dimensional NumPy-like array. You can put any data type in here, and perform vectorized operations on it. A series is also dictionary - like in many ways. Usually this is denoted as "s. Again, any data type can be stuffed in here. Usually this is denoted as "df". Index - This is what the data is "associated" by. So if you have time series data, like stock price information, generally the "index" is the date. Slicing - Selecting specific batches of data. With that, what all can we further do? Since Pandas uses objects, we quickly can sort our data in just about any way we please, and do it fast. We can move columns around, add new ones, and remove others. Along with that, we can do basic and complex mathematics on our data, either with our own code or with one of the many built-in functions for Pandas, like standard deviation, correlation, or moving averages for example. Pandas works seemlessly with Matplotlib including data sets that have dates. One of the most popular types of files to handle for data analysis in general is the CSV, or comma separated variable, file type. This is because the spreadsheet-style is popularly used for data analysis and lends itself to it. Reading CSV files into Python natively is actually fairly simplistic, but going from there can be a tedious challenge. With Pandas, we can of course read into and write to CSV files just like we can with Python already, but where Pandas shines is with any sort of manipulation of the data. Sometimes, you might only be concerned with a few columns out of many, or maybe you want to do some simple operations on the rows and columns. Maybe you want to add new columns, or move them around. All of this gets slightly more challenging natively with Python, but is quite simple with Pandas. First, we need to import the proper modules to help us in our task: For some of the indexes, the label might look funny. For stocks, however, the label is just the ticker. Instead of printing out everything, there is a quick and easy way for us to print out the first few lines of our data set: We can do this with one simple line: Since we just saved one, how about we just read from that?

Chapter 2 : Top 8 resources for learning data analysis with pandas

Python is a great language for doing data analysis, primarily because of the fantastic ecosystem of data-centric Python packages. Pandas is one of those packages, and makes importing and analyzing data much easier.

Install numpy, matplotlib, pandas, pandas-datareader, quandl, and sklearn Need help installing packages with pip? The Pandas module is a high performance, highly efficient, and high level data analysis library. At its core, it is very much like operating a headless version of a spreadsheet, like Excel. Most of the datasets you work with will be what are called dataframes. You may be familiar with this term already, it is used across other languages, but, if not, a dataframe is most often just like a spreadsheet. From here, we can utilize Pandas to perform operations on our data sets at lightning speeds. Pandas is also compatible with many of the other data analysis libraries, like Scikit-Learn for machine learning, Matplotlib for Graphing, NumPy, since it uses NumPy, and more. Another bit of good news? You can easily load in, and output out in the xls or xlsx format quickly, so, even if your boss wants to view things the old way, they can. Pandas is also compatible with text files, csv, hdf files, xml, html, and more with its incredibly powerful IO. Most importantly, if you have questions, ask them! If you seek out answers for each of the areas of confusion, and do this for everything, eventually you will have a full picture. Most of your questions will be Google-able as well. If I have not sold you yet on Pandas, the elevator pitch is: Next, go to your terminal or cmd. Did you get a "pip is not a recognized command" or something similar? No problem, this means pip is not on your PATH. You can look up how to add something to your path if you like, but you can always just explicitly give the path to the program you want to execute. Python34 means Python 3. If you have Python 3. The editor really does not matter in the grand scheme of things. You should try multiple editors, and go with the one that suits you best. Whatever you feel comfortable with, and you are productive with. Generally, a DataFrame is closest to the Dictionary Python data structure. This is just a common standard used when importing the Pandas module. Finally, we import pandas. Now, we can create a dataframe like so: Naming your dataframe df is not required, but again, is pretty popular standard for working with Pandas. It just helps people immediately identify the working dataframe without needing to trace the code back. So this gives us a dataframe, how do we see it? Well, can can just print it, like: Instead, most people will just do: We can clean this up to be just rows and columns like a spreadsheet might be with: As you perform analysis and such, this will be useful to see if what you intended actually happened or not. We could stop here with the intro, but one more thing: Like I said earlier, Pandas works great with other modules, Matplotlib being one of them. Open your terminal or cmd. You should already have got it I am prety sure with your pandas installation, but we want to make sure. Now, at the top of your script with the other imports, add: Style helps us quickly make our graphs look good, and style. Interested in learning more about Matplotlib? Check out the in-depth Matplotlib tutorial series! Next, below our print df. DataReader "XOM", "morningstar", start, end df.

Chapter 3 : Python Pandas Tutorial | Learn Pandas For Data Analysis | Edureka

Pandas is an open source Python library for data analysis. It gives Python the ability to work with spreadsheet-like data for fast data loading, manipulating, aligning, and merging, among other.

Data analysis with Pandas 13 Aug. This library is a high-level abstraction over low-level NumPy which is written in pure C. I use pandas on a daily basis and really enjoy it because of its eloquent syntax and rich functionality. Hope this note will help you dive into data analysis realm with Python and pandas. DataFrame and Series In order to master pandas you have to start from scratch with two main data structures: Series Series is an object which is similar to Python built-in list data structure but differs from it because it has associated label with each element or so-called index. This distinctive feature makes it look like associated array or dictionary hashmap representation. If index is not provided explicitly, then pandas creates RangeIndex starting from 0 to N-1, where N is a total number of elements. Moreover, each Series object has data type dtype , in our case data type is int Series has attributes to extract its values and index: DataFrame Simply said, DataFrame is a table. It has rows and columns. Each column in a DataFrame is a Series object, rows consist of elements inside Series. DataFrame can be constructed using built-in Python dicts: If you do not provide row index explicitly, pandas will create RangeIndex from 0 to N-1, where N is a number of rows inside DataFrame. Accessing elements by Index: Series object will also have the same index as DataFrame has: If you want to reset index, no problem: But usually in my practice I use CSV files. For example, if you want to save our previous DataFrame run this: We have saved our DataFrame, but what about reading data? Aggregating and Grouping in Pandas Grouping is probably one of the most popular methods in data analysis. If you want to group data in pandas you have to use. In order to demonstrate aggregates and grouping in pandas I decided to choose popular Titanic dataset which you can download using this link. In order to pivot a table in pandas you have to use. I will demonstrate how to use it on our Titanic dataset. Time Series analysis using pandas Pandas was created to analyze time series data. In order to illustrate how easy it is, I prepared sample dataset with Apple stock prices 5 year period. You can download it here. Open non-null float64 High non-null float64 Low non-null float64 Close non-null float64 Volume non-null int64 Adj Close non-null float64 dtypes: If datetime column is different from ISO format, then you have to use built-in pandas function pandas. If you want to know more about resampling go ahead and dive into official docs. Visualization in pandas At the very beginning of this post I said that pandas is built on numpy, when it comes to visualization pandas uses library called matplotlib. Taking Closing price between Feb, and Feb, Take a look at , there is a drop because of 7 to 1 split held by Apple. Hope you like it, if so please leave a comment below.

Chapter 4 : A Complete Tutorial to Learn Data Science with Python from Scratch

pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python.

Here is the description of variables: Also, you will be able to plot your data inline, which makes this a really good environment for interactive data analysis. You can check whether the environment has loaded correctly, by typing the following command and getting the output as seen in the figure below: Following are the libraries we will use during this tutorial: I have still kept them in the code, in case you use the code in a different environment. This is how the code looks like till this stage: Alternately, you can also look at more rows by printing the dataset. Next, you can look at summary of numerical fields by using describe function df. LoanAmount has â€" 22 missing values. Same with CoapplicantIncome Please note that we can get an idea of a possible skew in the data by comparing the mean to the median, i. For the non-numerical values e. The frequency table can be printed by following command: It can be a list of columns as well. Distribution analysis Now that we are familiar with basic data characteristics, let us study distribution of various variables. Let us start with numeric variables â€" namely ApplicantIncome and LoanAmount Lets start by plotting the histogram of ApplicantIncome using the following commands: This is also the reason why 50 bins are required to depict the distribution clearly. Next, we look at box plots to understand the distributions. Box plot for fare can be plotted by: This can be attributed to the income disparity in the society. Part of this can be driven by the fact that we are looking at people with different education levels. Let us segregate them by Education: But there are a higher number of graduates with very high incomes, which are appearing to be the outliers. Clearly, both ApplicantIncome and LoanAmount require some amount of data munging. LoanAmount has missing and well as extreme values values, while ApplicantIncome has a few extreme values, which demand deeper understanding. We will take this up in coming sections. Categorical variable analysis Now that we understand distributions for ApplicantIncome and LoanIncome, let us understand categorical variables in more details. We will use Excel style pivot table and cross-tabulation. For instance, let us look at the chances of getting a loan based on credit history. This can be achieved in MS Excel using a pivot table as: So the mean represents the probability of getting loan. Now we will look at the steps required to generate a similar insight using Python. Please refer to this article for getting a hang of the different data manipulation techniques in Pandas. Alternately, these two plots can also be visualized by combining them in a stacked chart:: You can quickly code this to create your first submission on AV Datahacks. I hope your love for pandas the animal would have increased by now â€" given the amount of help, the library can provide you in analyzing datasets. Data Munging in Python: Data munging â€" recap of the need While our exploration of the data, we found a few problems in the data set, which needs to be solved before the data is ready for a good model. Here are the problems, we are already aware of: We should estimate those values wisely depending on the amount of missing values and the expected importance of variables. In addition to these problems with numerical fields, we should also look at the non-numerical fields i. Though the missing values are not very high in number, but many variables have them and each one of these should be estimated and added in the data. Get a detailed view on different imputation techniques through this article. Remember that missing values may not always be NaNs. So we should check for values which are unpractical. How to fill missing values in LoanAmount? Thus we see some variations in the median of loan amount for each group and this can be used to impute the values. This can be done using the following code: Since the extreme values are practically possible, i. Now the distribution looks much closer to normal and effect of extreme values has been significantly subsided. One intuition can be that some applicants have lower income but strong support Co-applicants. So it might be a good idea to combine both incomes as total income and take a log transformation of the same. Also, I encourage you to think about possible additional information which can be derived from the data. Next, we will look at making predictive models. Skicit-Learn sklearn is the most commonly used library in Python for this purpose and we will follow the trail. I encourage you to get a

refresher on sklearn through this article. Since, sklearn requires all inputs to be numeric, we should convert all our categorical variables into numeric by encoding the categories. Before that we will fill all the missing values in the dataset. Then we will define a generic classification function, which takes a model as input and determines the Accuracy and Cross-Validation scores. Since this is an introductory article, I will not go into the details of coding. Please refer to this article for getting details of the algorithms with R and Python codes. Import models from scikit learn module: In simple words, taking all variables might result in the model understanding complex relations specific to the data and will not generalize well. Read more about Logistic Regression. We can easily make some intuitive hypothesis to set the ball rolling. The chances of getting a loan will be higher for: Applicants having a credit history remember we observed this in exploration? But this is a more challenging case. The accuracy and cross-validation score are not getting impacted by less important variables. We have two options now: I will leave this to your creativity. Decision Tree Decision tree is another method for making a predictive model. It is known to provide higher accuracy than logistic regression model. Read more about Decision Trees. We can try different combination of variables: This is the result of model over-fitting the data. Random Forest Random forest is another algorithm for solving the classification problem. Read more about Random Forest. An advantage with Random Forest is that we can make it work with all the features and it returns a feature importance matrix which can be used to select features. This is the ultimate case of overfitting and can be resolved in two ways: Create a series with feature importances: Also, we will modify the parameters of random forest model a little bit: Remember that random forest models are not exactly repeatable. Different runs will result in slight variations because of randomization. But the output should stay in the ballpark. You would have noticed that even after some basic parameter tuning on random forest, we have reached a cross-validation accuracy only slightly better than the original logistic regression model. This exercise gives us some very interesting and unique learning: Using a more sophisticated model does not guarantee better results. Avoid using complex modeling techniques as a black box without understanding the underlying concepts. Doing so would increase the tendency of overfitting thus making your models less interpretable Feature Engineering is the key to success. Everyone can use an Xgboost models but the real art and creativity lies in enhancing your features to better suit the model. So are you ready to take on the challenge? Majorly, it has great computational intensity and has powerful data analytics libraries.

DOWNLOAD PDF PANDAS FOR DATA ANALYSIS

Chapter 5 : pandas: powerful Python data analysis toolkit â€” pandas documentation

Please enter a valid input. Please enter a valid email id or comma separated email id's. In order to perform slicing on data, you need a data frame. Don't worry, data frame is a 2-dimensional data structure and a most common pandas object. So first, let's create a data frame. Refer the below.

Pandas is used for data manipulation, analysis and cleaning. Python pandas is well suited for different kinds of data, such as: Python Pandas Operations Using Python pandas, you can perform a lot of operations with series, data frames, missing data, group by etc. Some of the common operations for data manipulation are listed below: Now, let us understand all these operations one by one. Slicing the Data Frame In order to perform slicing on data, you need a data frame. Now, let us slice a particular column from this data frame. Refer the image below: Get Python Certified Now! You can also decide which columns you want to make common. Let me implement that practically, first I will create three data frames, which has some key-value pairs and then merge the data frames together. Refer the code below: Now, you can also specify the column which you want to make common. So, let me implement that practically: It is yet another convenient method to combine two differently indexed dataframes into a single result dataframe. Let us implement it practically. Later in , both the values are available, therefore it has printed the respective values. Concatenation Concatenation basically glues the dataframes together. You can select the dimension on which you want to concatenate. Consider the below example. Therefore, you should make sure that you have all the information lining up correctly when you join or cancatenate on the axis. For example, let us create a dataframe with some key value pairs in a dictionary and change the index values. Consider the example below: Let us see how it actually happens: Change the Column Headers Let us now change the headers of column in this python pandas tutorial. So, let me implement it practically. Next in python pandas tutorial, let us perform data munging. Data Munging In Data munging, you can convert a particular data into a different format. For example, if you have a. So, let me implement this practically. You can directly copy the path of the file and paste it in your browser which displays the data in a HTML format. Refer the below screenshot: You have to use this dataset and find the change in the percentage of youth for every country from First, let us understand the dataset which contains the columns as Country Name, Country Code and the year from to Refer the screenshot below:

Chapter 6 : Data analysis with Pandas

Data Analysis in Python with Pandas (ratings) Course Ratings are calculated from individual students' ratings and a variety of other signals, like age of rating and reliability, to ensure that they reflect course quality fairly and accurately.

How to get frequency counts of unique items of a series? L1 Calculte the frequency counts of each unique value ser. How to bin a numeric series to 10 groups of equal size? L2 Bin the series ser into 10 equal deciles and replace the values with the bin name. How to convert a numpy array to a dataframe of given shape? How to find the positions of numbers that are multiples of 3 from a series? L2 Find the positions of numbers that are multiples of 3 from ser. How to extract items at given positions from a series Difficulty Level: L1 From ser, extract the items at positions in list pos. How to stack two series vertically and horizontally? L1 Stack ser1 and ser2 vertically and horizontally to form a dataframe. How to get the positions of items of series A in another series B? L2 Get the positions of items of ser2 in ser1 as a list. Series [1, 3, 10, 13] Solution 1 [np. How to compute the mean squared error on a truth and predicted series? L2 Compute the mean squared error of truth and pred series. How to convert the first character of each element in a series to uppercase? L2 Change the first character of each word to upper case in each word of ser. How to calculate the number of characters in each word in a series? How to compute difference of differences between consequtive numbers of a series? L1 Difference of differences between the consequtive numbers of ser. Series [1, 3, 6, 10, 15, 21, 27, 35] Desired Output [nan, 2. Series [1, 3, 6, 10, 15, 21, 27, 35] Solution print ser. How to convert a series of date-strings to a timeseries? How to get the day of month, week number, day of year and day of week from a series of date strings? L2 Get the day of month, week number, day of year and day of week from ser. How to convert year-month string to dates corresponding to the 4th day of the month? L2 Change ser to dates that start with 4th of the respective months. How to filter words that contain atleast 2 vowels from a series? L3 From ser, extract words that contain atleast 2 vowels. How to filter valid emails from a series? L3 Extract the valid emails from the series emails. The regex pattern for valid emails is provided as reference. How to get the mean of a series grouped by another series? L2 Compute the mean of weights of each fruit. How to compute the euclidean distance between two series? L2 Compute the euclidean distance between series points p and q, without using a packaged formula. Series [10, 9, 8, 7, 6, 5, 4, 3, 2, 1] Desired Output How to find all the local maxima or peaks in a numeric series? L3 Get the positions of peaks values surrounded by smaller values on both sides in ser. How to replace missing spaces in a string with the least frequent character? How to fill an intermittent time series so all missing dates show up with values of previous non-missing date? L2 ser has missing dates and values. Make all missing dates appear and fill up with value from previous date. How to compute the autocorrelations of a numeric series? L3 Compute autocorrelations for the first 10 lags of ser. Find out which lag has the largest correlation. How to import only every nth row from a csv file to create a dataframe? L2 Import every 50th row of BostonHousing dataset as a dataframe. Show Solution Solution 1: DataFrame for chunk in df: How to change column values when importing csv to a dataframe? How to create a dataframe with rows as strides from a given series? How to import only specified columns from a csv file? How to get the nrows, ncolumns, datatype, summary stats of each column of a dataframe? Also get the array and list equivalent. L2 Get the number of rows, columns, datatype and summary statistics of each column of the Cars93 dataset. Also get the numpy array and list equivalent of the dataframe. Price float64 Price float64 Max. How to extract the row and column number of a particular cell with given criterion? What is the row and column number of the cell with the highest Price value? Price Get the value df. How to rename a specific columns in a dataframe? How to check if a dataframe has any missing values? L1 Check if df has any missing values.

Chapter 7 : pandas (software) - Wikipedia

Explores sample Jupyter Notebooks to showcase the power of pandas for data analysis The racedaydvl.com attachment with the working files for this course is attached to this lesson. Download and unpack the racedaydvl.com file in the

directory of your choice.

Chapter 8 : Python Programming Tutorials

python pandas exercises are designed to challenge your logical muscle and to help internalize data manipulation with python's favorite package for data analysis. The questions are of 3 levels of difficulties with L1 being the easiest to L3 being the hardest.

Chapter 9 : Pandas Exercises for Data Analysis â€“ Machine Learning Plus

There are audio issues with this video that cannot be fixed. We recommend listening to the tutorial without headphones to minimize the buzzing sound. Tutorial information may be found at https.