**Python Pandas Tutorial : Learn Pandas for Data Analysis**

 Last updated on Nov 26,201929K Views



[**Aayushi Johari**](https://www.edureka.co/blog/author/aayushiedureka-co/) A technophile who likes writing about different technologies and spreading knowledge.

* [**2 Comments**](javascript:void(0))
* [**Bookmark**](https://www.edureka.co/blog/python-pandas-tutorial/)

**4 / 17 Blog from Python Libraries**

In this blog, we will be discussing data analysis using Pandas in Python. Today, [***Python Certification***](https://www.edureka.co/data-science-python-certification-course) is a hot skill in the industry that surpassed PHP in 2017 and C# in 2018 in terms of overall popularity and use. Before talking about Pandas, one must understand the concept of Numpy arrays. Why? Because Pandas is an open source software library which is built on top of [NumPy](https://www.edureka.co/blog/python-numpy-tutorial/). In this Python Pandas Tutorial, I will take you through the following topics, which will serve as fundamentals for the upcoming blogs:

* [What is Pandas?](https://www.edureka.co/blog/python-pandas-tutorial/#WhatIsPandas)
* [Pandas Operation](https://www.edureka.co/blog/python-pandas-tutorial/#PandasOperations)
  + [Slicing the data frame](https://www.edureka.co/blog/python-pandas-tutorial/#Slicing)
  + [Merging & Joining](https://www.edureka.co/blog/python-pandas-tutorial/#Merging&Joining)
  + [Concatenation](https://www.edureka.co/blog/python-pandas-tutorial/#Concatenation)
  + [Changing the index](https://www.edureka.co/blog/python-pandas-tutorial/#ChangeIndex)
  + [Change Column headers](https://www.edureka.co/blog/python-pandas-tutorial/#ChangeHeaders)
  + [Data munging](https://www.edureka.co/blog/python-pandas-tutorial/#Munging)
* [Use-Case: Analyze youth unemployment data](https://www.edureka.co/blog/python-pandas-tutorial/#Use-Case)

Let’s get started. :-)

**What is Python Pandas?**

Pandas is used for data manipulation, analysis and cleaning. Python pandas is well suited for different kinds of data, such as:

* Tabular data with heterogeneously-typed columns
* Ordered and unordered time series data
* Arbitrary matrix data with row & column labels
* Unlabelled data
* Any other form of observational or statistical data sets

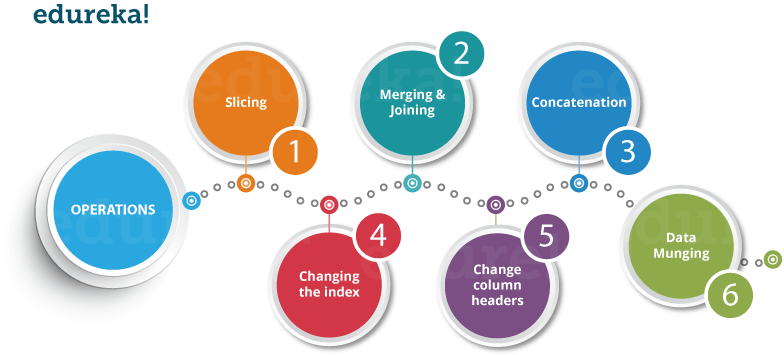
**How to install Pandas?**

To install Python Pandas, go to your command line/ terminal and type “pip install pandas” or else, if you have anaconda installed in your system, just type in “conda install pandas”. Once the installation is completed, go to your IDE (Jupyter, PyCharm etc.) and simply import it by typing: “import pandas as pd”

Moving ahead in Python pandas tutorial, let’s take a look at some of its operations:

**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. 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.

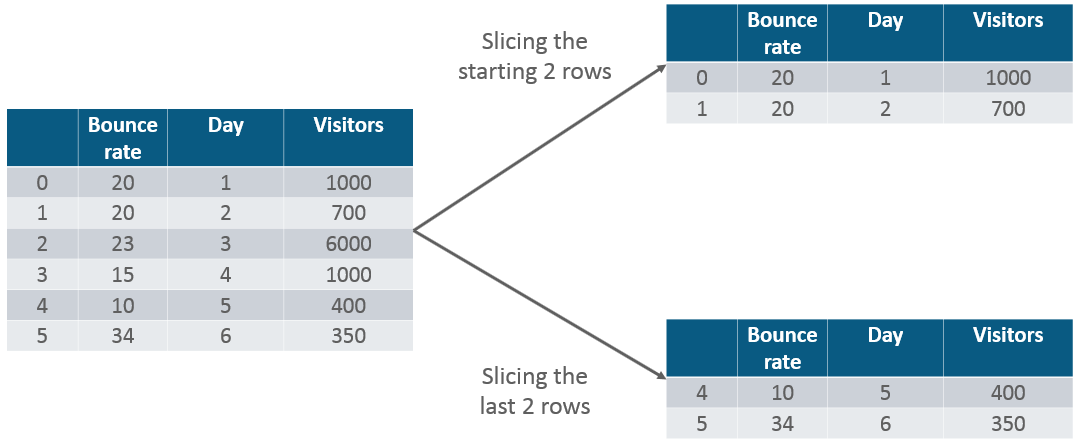
Refer the below code for its implementation in PyCharm:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | import pandas as pd    XYZ\_web= {'Day':[1,2,3,4,5,6], "Visitors":[1000, 700,6000,1000,400,350], "Bounce\_Rate":[20,20, 23,15,10,34]}    df= pd.DataFrame(XYZ\_web)    print(df) |

**Output:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | Bounce\_Rate Day Visitors  0     20          1   1000  1     20          2   700  2     23          3   6000  3     15          4   1000  4     10          5   400  5     34          6   350 |

The code above will convert a dictionary into a pandas Data Frame along with index to the left. Now, let us slice a particular column from this data frame. Refer the image below:



|  |  |
| --- | --- |
| 1 | print(df.head(2)) |

**Output:**

|  |  |
| --- | --- |
| 1  2  3 | Bounce\_Rate Day Visitors  0      20         1   1000  1      20         2    700 |

Similarly, if you want the last two rows of the data, type in the below command:

|  |  |
| --- | --- |
| 1 | print(df.tail(2)) |

**Output:**

[[](https://www.edureka.co/python-programming-certification-training)](https://www.edureka.co/python-programming-certification-training" \t "_blank)

**[Python Programming Certification Course](https://www.edureka.co/python-programming-certification-training" \t "_blank)**

[Explore Curriculum](https://www.edureka.co/python-programming-certification-training" \t "_blank)

|  |  |
| --- | --- |
| 1  2  3 | Bounce\_Rate Day Visitors  4      10      5    400  5      34      6    350 |

Next in Python Pandas tutorial, let us perform merging and joining.

**Merging & Joining**

In merging, you can merge two data frames to form a single data frame. 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:

|  |  |
| --- | --- |
| 1  2  3  4  5 | HPI   IND\_GDP Int\_Rate  0  80      50      2  1  90      45      1  2  70      45      2  3  60      67      3 |

**Output:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | import pandas as pd    df1= pd.DataFrame({ "HPI":[80,90,70,60],"Int\_Rate":[2,1,2,3],"IND\_GDP":[50,45,45,67]}, index=[2001, 2002,2003,2004])    df2=pd.DataFrame({ "HPI":[80,90,70,60],"Int\_Rate":[2,1,2,3],"IND\_GDP":[50,45,45,67]}, index=[2005, 2006,2007,2008])    merged= pd.merge(df1,df2)    print(merged) |

As you can see above, the two data frames has merged into a single data frame. Now, you can also specify the column which you want to make common. For example, I want the “HPI” column to be common and for everything else, I want separate columns. So, let me implement that practically:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | df1 = pd.DataFrame({"HPI":[80,90,70,60],"Int\_Rate":[2,1,2,3], "IND\_GDP":[50,45,45,67]}, index=[2001, 2002,2003,2004])    df2 = pd.DataFrame({"HPI":[80,90,70,60],"Int\_Rate":[2,1,2,3],"IND\_GDP":[50,45,45,67]}, index=[2005, 2006,2007,2008])    merged= pd.merge(df1,df2,on ="HPI")    print(merged) |

**Output:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | IND\_GDP  Int\_Rate  Low\_Tier\_HPI  Unemployment  2001     50      2         50.0            1.0  2002     45      1         NaN             NaN  2003     45      2         45.0            3.0  2004     67      3         67.0            5.0  2004     67      3         34.0            6.0 |

Next, let us understand **joining** in python pandas tutorial. It is yet another convenient method to combine two differently indexed dataframes into a single result dataframe. This is quite similar to the “merge” operation, except the joining operation will be on the “index” instead of  the “columns”. Let us implement it practically.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | df1 = pd.DataFrame({"Int\_Rate":[2,1,2,3], "IND\_GDP":[50,45,45,67]}, index=[2001, 2002,2003,2004])    df2 = pd.DataFrame({"Low\_Tier\_HPI":[50,45,67,34],"Unemployment":[1,3,5,6]}, index=[2001, 2003,2004,2004])    joined= df1.join(df2)  print(joined) |

**Output:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | IND\_GDP  Int\_Rate Low\_Tier\_HPI  Unemployment  2001     50       2         50.0           1.0  2002     45       1         NaN            NaN  2003     45       2         45.0           3.0  2004     67       3         67.0           5.0  2004     67       3         34.0           6.0 |

As you can notice in the above output, in year 2002(index), there is no value attached to columns “low\_tier\_HPI” and “unemployment”, therefore it has printed NaN (Not a Number). Later in 2004, both the values are available, therefore it has printed the respective values.

*You may go through this recording of Python Pandas tutorial where our instructor has explained the topics in a detailed manner with examples that will help you to understand this concept better.*

**Python For Data Analysis | Python Pandas Tutorial | Python Training | Edureka**

Moving ahead in Python pandas tutorial, let us understand how to concatenate two data data frames.

**Concatenation**

Concatenation basically glues the dataframes together. You can select the dimension on which you want to concatenate. For that, just use “pd.concat” and pass in the list of dataframes to concatenate together. Consider the below example.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | df1 = pd.DataFrame({"HPI":[80,90,70,60],"Int\_Rate":[2,1,2,3], "IND\_GDP":[50,45,45,67]}, index=[2001, 2002,2003,2004])    df2 = pd.DataFrame({"HPI":[80,90,70,60],"Int\_Rate":[2,1,2,3],"IND\_GDP":[50,45,45,67]}, index=[2005, 2006,2007,2008])    concat= pd.concat([df1,df2])    print(concat) |

**Output:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | HPI  IND\_GDP Int\_Rate  2001    80    50       2  2002    90    45       1  2003    70    45       2  2004    60    67       3  2005    80    50       2  2006    90    45       1  2007    70    45       2  2008    60    67       3 |

As you can see above, the two dataframes are glued together in a single dataframe, where the index starts from 2001 all the way upto 2008. Next, you can also specify axis=1 in order to join, merge or cancatenate along the columns. Refer the code below:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | df1 = pd.DataFrame({"HPI":[80,90,70,60],"Int\_Rate":[2,1,2,3], "IND\_GDP":[50,45,45,67]}, index=[2001, 2002,2003,2004])    df2 = pd.DataFrame({"HPI":[80,90,70,60],"Int\_Rate":[2,1,2,3],"IND\_GDP":[50,45,45,67]}, index=[2005, 2006,2007,2008])    concat= pd.concat([df1,df2],axis=1)    print(concat) |

**Output:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | HPI  IND\_GDP  Int\_Rate HPI  IND\_GDP Int\_Rate  2001   80.0  50.0       2.0   NaN    NaN     NaN  2002   90.0  45.0       1.0   NaN    NaN     NaN  2003   70.0  45.0       2.0   NaN    NaN     NaN  2004   60.0  67.0       3.0   NaN    NaN     NaN  2005   NaN   NaN        NaN   80.0   50.0    2.0  2006   NaN   NaN        NaN   90.0   45.0    1.0  2007   NaN   NaN        NaN   70.0   45.0    2.0  2008   NaN   NaN        NaN   60.0   67.0    3.0 |

As you can above, there are bunch of missing values. This happens because the dataframes didn’t have values for all the indexes you want to concatenate on. Therefore, you should make sure that you have all the information lining up correctly when you join or concatenate on the axis.

**Change the index**

Next in python pandas tutorial, we’ll understand how to change the index values in a dataframe. 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:

**Data Science Training**

**[PYTHON CERTIFICATION TRAINING FOR DATA SCIENCE](https://www.edureka.co/data-science-python-certification-course" \o "Python Certification Training for Data Science" \t "_blank)**

**[Python Certification Training for Data Science](https://www.edureka.co/data-science-python-certification-course" \o "Python Certification Training for Data Science" \t "_blank)**

*[Reviews](https://www.edureka.co/data-science-python-certification-course" \o "Python Certification Training for Data Science" \t "_blank)*

**[5](https://www.edureka.co/data-science-python-certification-course" \o "Python Certification Training for Data Science" \t "_blank)**[(58893)](https://www.edureka.co/data-science-python-certification-course" \o "Python Certification Training for Data Science" \t "_blank)

**[MACHINE LEARNING CERTIFICATION TRAINING USING PYTHON](https://www.edureka.co/machine-learning-certification-training" \o "Machine Learning Certification Training using Python" \t "_blank)**

**[Machine Learning Certification Training using Python](https://www.edureka.co/machine-learning-certification-training" \o "Machine Learning Certification Training using Python" \t "_blank)**

*[Reviews](https://www.edureka.co/machine-learning-certification-training" \o "Machine Learning Certification Training using Python" \t "_blank)*

**[5](https://www.edureka.co/machine-learning-certification-training" \o "Machine Learning Certification Training using Python" \t "_blank)**[(6903)](https://www.edureka.co/machine-learning-certification-training" \o "Machine Learning Certification Training using Python" \t "_blank)

**[DATA SCIENCE CERTIFICATION COURSE USING R](https://www.edureka.co/data-science-r-programming-certification-course" \o "Data Science Certification Course using R" \t "_blank)**

**[Data Science Certification Course using R](https://www.edureka.co/data-science-r-programming-certification-course" \o "Data Science Certification Course using R" \t "_blank)**

*[Reviews](https://www.edureka.co/data-science-r-programming-certification-course" \o "Data Science Certification Course using R" \t "_blank)*

**[5](https://www.edureka.co/data-science-r-programming-certification-course" \o "Data Science Certification Course using R" \t "_blank)**[(35573)](https://www.edureka.co/data-science-r-programming-certification-course" \o "Data Science Certification Course using R" \t "_blank)

**[DATA ANALYTICS WITH R CERTIFICATION TRAINING](https://www.edureka.co/data-analytics-with-r-certification-training" \o "Data Analytics with R Certification Training" \t "_blank)**

**[Data Analytics with R Certification Training](https://www.edureka.co/data-analytics-with-r-certification-training" \o "Data Analytics with R Certification Training" \t "_blank)**

*[Reviews](https://www.edureka.co/data-analytics-with-r-certification-training" \o "Data Analytics with R Certification Training" \t "_blank)*

**[5](https://www.edureka.co/data-analytics-with-r-certification-training" \o "Data Analytics with R Certification Training" \t "_blank)**[(22496)](https://www.edureka.co/data-analytics-with-r-certification-training" \o "Data Analytics with R Certification Training" \t "_blank)

**[PYTHON PROGRAMMING CERTIFICATION COURSE](https://www.edureka.co/python-programming-certification-training" \o "Python Programming Certification Course" \t "_blank)**

**[Python Programming Certification Course](https://www.edureka.co/python-programming-certification-training" \o "Python Programming Certification Course" \t "_blank)**

*[Reviews](https://www.edureka.co/python-programming-certification-training" \o "Python Programming Certification Course" \t "_blank)*

**[5](https://www.edureka.co/python-programming-certification-training" \o "Python Programming Certification Course" \t "_blank)**[(12384)](https://www.edureka.co/python-programming-certification-training" \o "Python Programming Certification Course" \t "_blank)

**[STATISTICS ESSENTIALS FOR ANALYTICS](https://www.edureka.co/statistics-essentials-for-analytics-self-paced" \o "Statistics Essentials for Analytics" \t "_blank)**

**[Statistics Essentials for Analytics](https://www.edureka.co/statistics-essentials-for-analytics-self-paced" \o "Statistics Essentials for Analytics" \t "_blank)**

*[Reviews](https://www.edureka.co/statistics-essentials-for-analytics-self-paced" \o "Statistics Essentials for Analytics" \t "_blank)*

**[5](https://www.edureka.co/statistics-essentials-for-analytics-self-paced" \o "Statistics Essentials for Analytics" \t "_blank)**[(5077)](https://www.edureka.co/statistics-essentials-for-analytics-self-paced" \o "Statistics Essentials for Analytics" \t "_blank)

**[SAS TRAINING AND CERTIFICATION](https://www.edureka.co/sas-training" \o "SAS Training and Certification" \t "_blank)**

**[SAS Training and Certification](https://www.edureka.co/sas-training" \o "SAS Training and Certification" \t "_blank)**

*[Reviews](https://www.edureka.co/sas-training" \o "SAS Training and Certification" \t "_blank)*

**[5](https://www.edureka.co/sas-training" \o "SAS Training and Certification" \t "_blank)**[(4116)](https://www.edureka.co/sas-training" \o "SAS Training and Certification" \t "_blank)

**[ANALYTICS FOR RETAIL BANKS](https://www.edureka.co/retail-banking" \o "Analytics for Retail Banks" \t "_blank)**

**[Analytics for Retail Banks](https://www.edureka.co/retail-banking" \o "Analytics for Retail Banks" \t "_blank)**

*[Reviews](https://www.edureka.co/retail-banking" \o "Analytics for Retail Banks" \t "_blank)*

**[5](https://www.edureka.co/retail-banking" \o "Analytics for Retail Banks" \t "_blank)**[(892)](https://www.edureka.co/retail-banking" \o "Analytics for Retail Banks" \t "_blank)

**[ADVANCED PREDICTIVE MODELLING IN R CERTIFICATION TRAINING](https://www.edureka.co/advanced-predictive-modeling-in-r-sp" \o "Advanced Predictive Modelling in R Certification Training" \t "_blank)**

**[Advanced Predictive Modelling in R Certification Training](https://www.edureka.co/advanced-predictive-modeling-in-r-sp" \o "Advanced Predictive Modelling in R Certification Training" \t "_blank)**

*[Reviews](https://www.edureka.co/advanced-predictive-modeling-in-r-sp" \o "Advanced Predictive Modelling in R Certification Training" \t "_blank)*

**[4](https://www.edureka.co/advanced-predictive-modeling-in-r-sp" \o "Advanced Predictive Modelling in R Certification Training" \t "_blank)**[(3525)](https://www.edureka.co/advanced-predictive-modeling-in-r-sp" \o "Advanced Predictive Modelling in R Certification Training" \t "_blank)

Next

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | import pandas as pd    df= pd.DataFrame({"Day":[1,2,3,4], "Visitors":[200, 100,230,300], "Bounce\_Rate":[20,45,60,10]})    df.set\_index("Day", inplace= True)    print(df) |

**Output:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | Bounce\_Rate  Visitors  Day  1      20           200  2      45           100  3      60           230  4      10           300 |

As you can notice in the output above, the index value has been changed with respect to the “Day” column.

**Change the Column Headers**

Let us now change the headers of column in this python pandas tutorial. Let us take the same example, where I will change the column header from “Visitors” to “Users”. So, let me implement it practically.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | import pandas as pd    df = pd.DataFrame({"Day":[1,2,3,4], "Visitors":[200, 100,230,300], "Bounce\_Rate":[20,45,60,10]})    df = df.rename(columns={"Visitors":"Users"})    print(df) |

**Output:**

|  |  |
| --- | --- |
| 1  2  3  4  5 | Bounce\_Rate  Day  Users  0    20         1    200  1    45         2    100  2    60         3    230  3    10         4    300 |

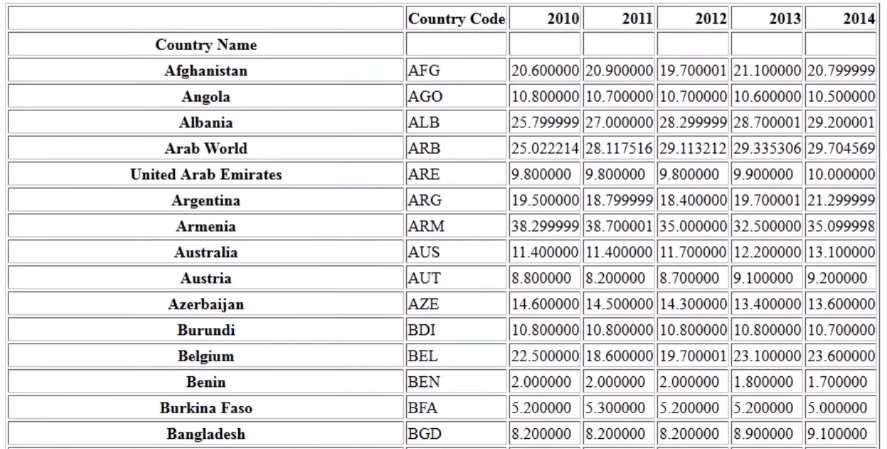
As you see above, column header “Visitors” has been changed to “Users”. Next in python pandas tutorial, let us perform data munging.

**Data Munging**

In Data munging, you can convert a particular data into a different format. For example, if you have a .csv file, you can convert it into .html or any other data format as well. So, let me implement this practically.

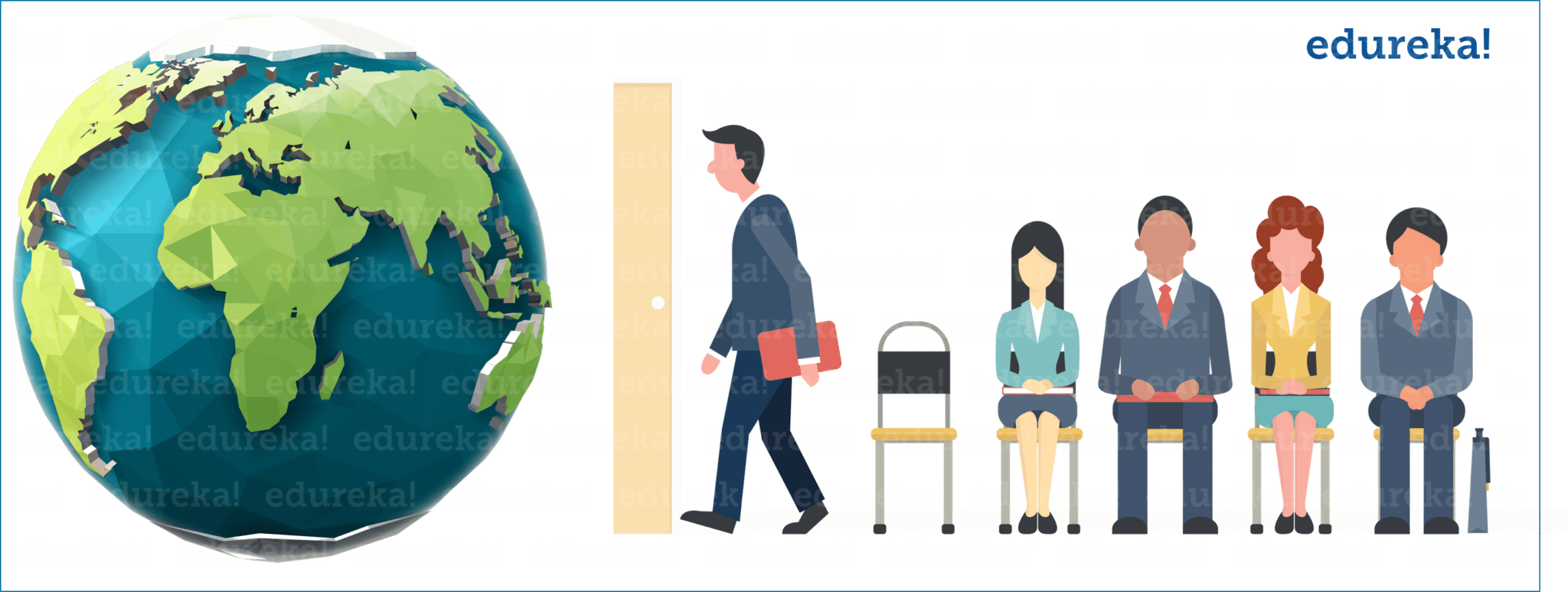
|  |  |
| --- | --- |
| 1  2  3  4  5 | import pandas as pd    country= pd.read\_csv("D:UsersAayushiDownloadsworld-bank-youth-unemploymentAPI\_ILO\_country\_YU.csv",index\_col=0)    country.to\_html('edu.html') |

Once you run this code, a HTML file will be created named “edu.html”. 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:

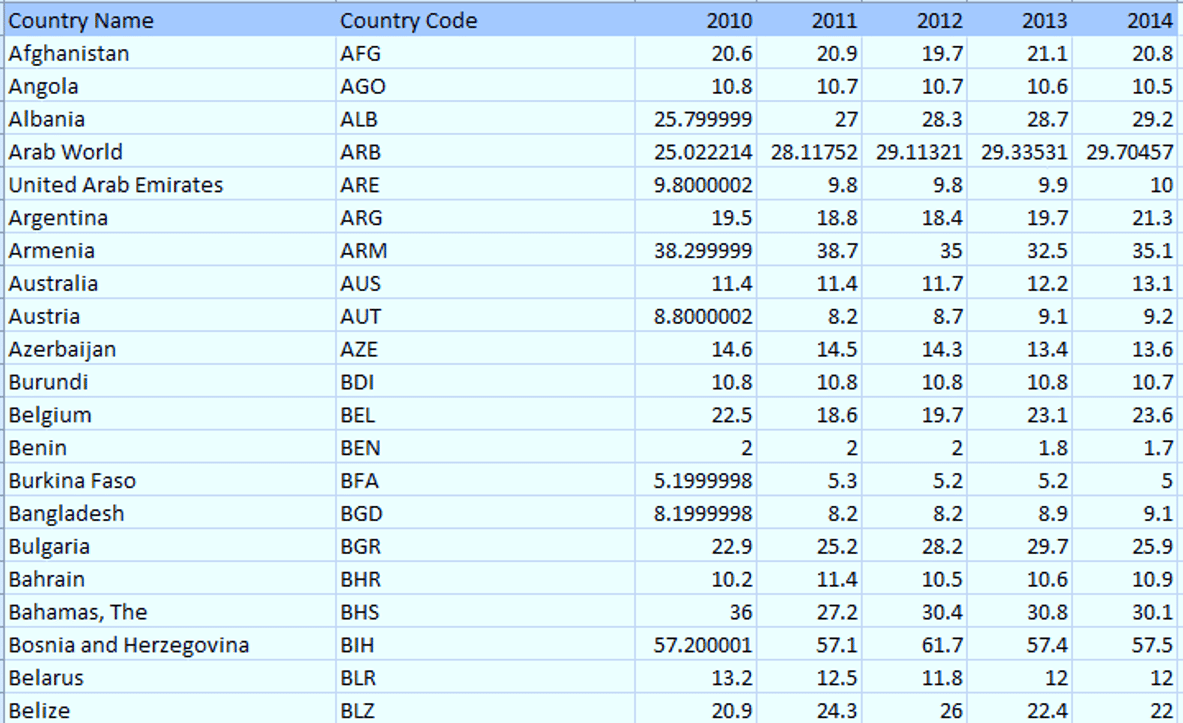
  
Next in python pandas tutorial, let’s have a look at a use-case which talks about the global youth unemployment.

**Python Pandas Tutorial: Use Case to Analyze Youth Unemployment Data**

**Problem Statement**: You are given a dataset whichcomprises of the percentage of unemployed youth globally from 2010 to 2014. You have to use this dataset and find the change in the percentage of youth for every country from 2010-2011.



First, let us understand the dataset which contains the columns as Country Name, Country Code and the year from 2010 to 2014.  Now using pandas, we will use “pd.read\_csv” to read the .csv file format file.   
Refer the screenshot below:



Let us move ahead and perform data analysis in which we are going to find out the percentage change in the unemployed youth between 2010 to 2011. Then we will visualize the same using [Matplotlib](https://www.edureka.co/blog/python-matplotlib-tutorial/#matplotlib) library, which is a powerful library for visualization in Python. It can be used in Python scripts, shell, web application servers and other GUI toolkits. You can use read more here: [Matplotlib Tutorial.](https://www.edureka.co/blog/python-matplotlib-tutorial/)

Now, let us implement the code in PyCharm:

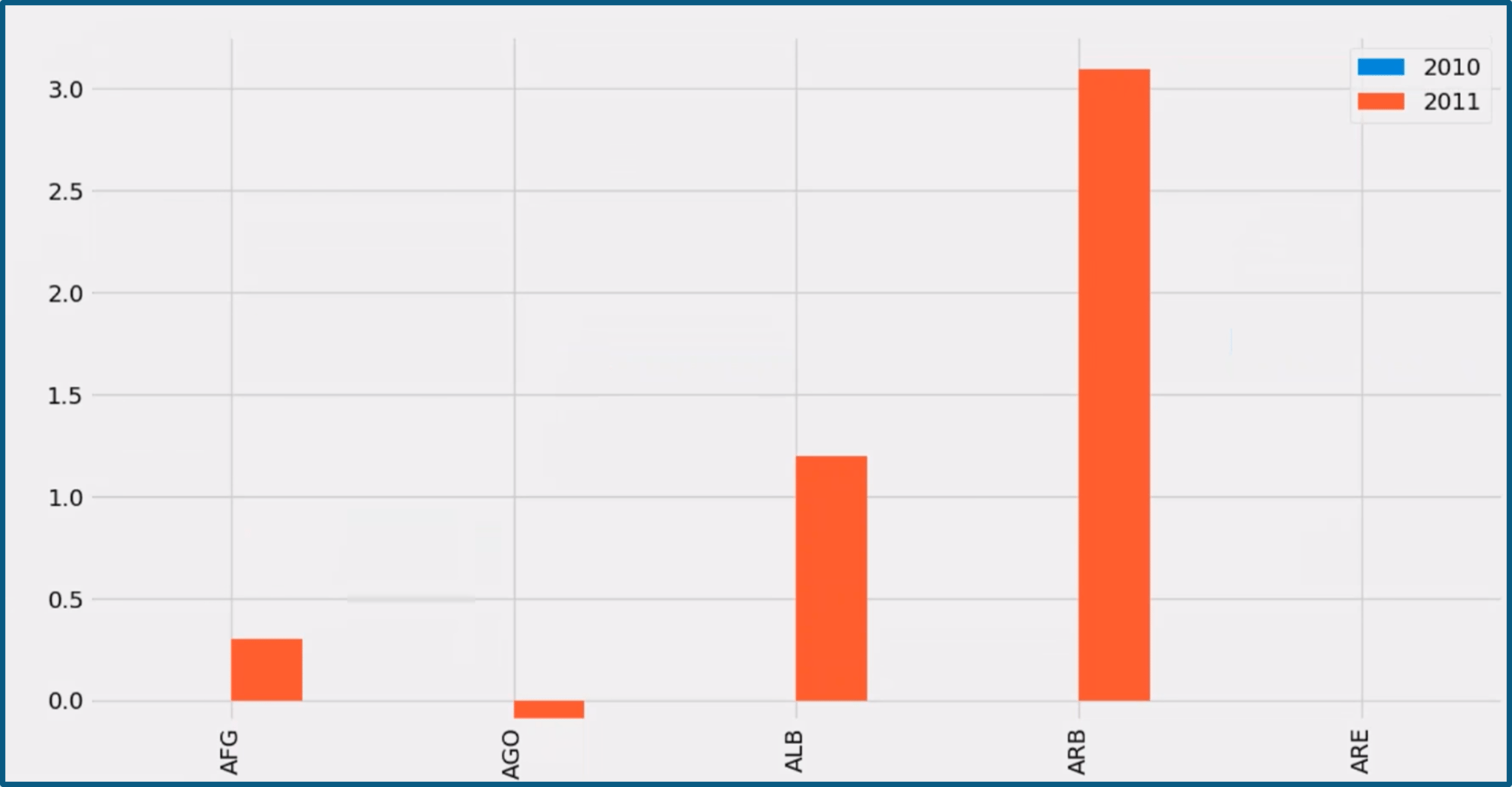
[[](https://www.edureka.co/python-programming-certification-training)](https://www.edureka.co/python-programming-certification-training" \t "_blank)

**[Python Programming Certification Course](https://www.edureka.co/python-programming-certification-training" \t "_blank)**

[Weekday / Weekend BatchesSee Batch Details](https://www.edureka.co/python-programming-certification-training" \t "_blank)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | import pandas as pd    import matplotlib.pyplot as plt    from matplotlib import style    style.use('fivethirtyeight')    country= pd.read\_csv("D:UsersAayushiDownloadsworld-bank-youth-unemploymentAPI\_ILO\_country\_YU.csv",index\_col=0)    df= country.head(5)    df= df.set\_index(["Country Code"])    sd = sd.reindex(columns=['2010','2011'])    db= sd.diff(axis=1)    db.plot(kind="bar")    plt.show() |

As you can see above, I have performed the analysis on the top 5 rows of the country dataframe. Next, I have defined a index value to be “Country Code” and then re-index the column to 2010 and 2011. Then, we have one more dataframe db, which prints the difference between the two columns or the percentage change of unemployed youth from 2010 to 2011. Finally, I have plotted a barplot using Matplotlib library in Python.



Now if you noticed in the above plot, in Afghanistan(AFG) between 2010 to 2011, there has been a rise in unemployed youth of approx. 0.25%. Then in Angola(AGO), there is a negative trend which means that the percentage of unemployed youth has been reduced. Similarly, you can perform analysis on different sets of data.

*I hope my blog on “Python Pandas Tutorial” was relevant for you. To get in-depth knowledge on python along with its various applications, you can enroll for live*[***Python online training***](https://www.edureka.co/data-science-python-certification-course)*by Edureka with 24/7 support and lifetime access.*

*Got a question for us? Please mention it in the comments section of this “Python Pandas tutorial” blog  and we will get back to you as soon as possible.*