# CHAPTER 1: Python Ecosystem for Machine Learning

## Introduction

This chapter is intended to provide a comprehensive introduction to Python Ecosystem. After completing this chapter, you will know about python, the strengths, & weaknesses of python, installing python, and finally python packages.

## An Introduction to Python

Python is a popular object-oriented programming language having the capabilities of a high-level programming language. Its easy-to-learn syntax and portability capability make it popular these days. The following facts give us an introduction to Python.

* Python was developed by Guido van Rossum at Stichting Mathematisch Centrum in the Netherlands.
* It was written as the successor of a programming language named ‘ABC’
* Its first version was released in 1991.
* The name python was picked by Guido van Rossum from a TV show named Monty Python’s Flying Circus.
* It is an open-source programming language which means that we can freely download it and use it to develop programs. It can be downloaded from [www.python.org](http://www.python.org)
* Python programming language is having the features of Java and C both. It is having the elegant ‘C’ code and on the other hand, it is having classes and objects like Java for object-oriented programming.
* It is an interpreted language, which means the source code of the python program would be first converted into bytecode and then executed by the Python virtual machine.

**Strengths and Weaknesses of Python**

every programming language has some strengths as well as weaknesses so does Python too.

**Strengths**

According to students and surveys, Python is the fifth most important language as well as the most popular language for machine learning and data science. it is because of the following strength’s that Python has-

**Easy to learn and understand** - the syntax of Python is simpler; hence it is relatively easy, even for beginners also, to learn and understand the language.

**Multi-purpose language** - Python is a multi-purpose programming language because it supports structured programming, object-oriented programming as well as functional programming.

**A huge number of modules** - Python has a huge number of modules covering every aspect of programming. These modules are easily available for use hence making Python an extensible language.

**Support of open-source community** - as being an open-source programming language, Python is supported by a very large developer community. due to this, the bugs are easily fixed by the Python community. This characteristic makes Python very robust and adaptive.

**Scalability** - Python is a scalable programming language because it provides an improved structure for supporting large programs than shell scripts.

**Weakness**

Although Python is a popular and powerful programming language, it has its own weakness of slow execution speed.

The execution speed of Python is slow as compared to compiled languages because Python is an interpreted language. this can be the major area of improvement for the Python community.

## Installing Python

For working in Python, you must first have to install it, you can perform the installation of Python in any of the following two ways –

* Installing Python individually
* using pre-packaged Python distribution - Anaconda.

Let us discuss this each in detail.

If you want to install Python on your computer, then you need to download only the binary code applicable to your platform. Python distribution is available for Windows, Linux, and Mac platforms.

The following is a quick overview of installing Python on the above-mentioned platforms –

**On Unix and Linux platforms**

With help of the following steps, we can install Python on Unix and Linux platforms –

* First, go to [www.python.org/downloads/](http://www.python.org/downloads/)
* Next, click on the link to download the zipped source code available for Unix/Linux.
* Now, download and extract files.
* Next, we can edit the Modules/Setup file if we want to customize some options.
  + Next, write the command **run ./configure script**
  + Make
  + Make install

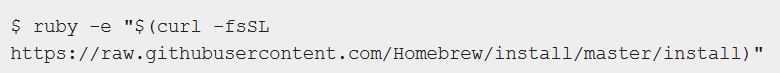
**On Windows platform**

With the help of the following steps, we can install Python on the Windows platform –

* First, go to [www.python.org/downloads/](http://www.python.org/downloads/)
* Next, click on the link for the windows installer python-XYZ.msi file. Here XYZ is the version we wish to install.
* Now, we must run the file that is downloaded. It will take us to the Python install wizard, which is easy to use. Now, accept the default settings and wait until the installation is finished.

**On Macintosh platform**

For Mac OS X, Homebrew, a great and easy-to-use package installer is recommended to install Python 3. In case If you don't have Homebrew, You can install it with the help of the following command –



It can be updated with the command below –



Now, to install Python3 on your system, we need to run the following command –

****

**Using Pre-packaged Python distribution: Anaconda**

Anaconda is packaged compilation of Python which have all the libraries widely used in data science. We can follow the following steps setup Python environment using Anaconda –

* Step 1 – First, we need to download the required installation package from Anaconda distribution. the link for the same is [www.anaconda.com/distribution](http://www.anaconda.com/distribution). You can choose from Windows, Mac, and Linux OS as per your requirement.
* Step 2 – Next, select the Python version you want to install on your machine. the latest Python version is 3.10. There you will get the options for both 64-bit and 32-bit graphical installers both.
* Step 3 - After selecting the OS and Python version, it will download the Anaconda installer on your computer. Now, double-click the file, and the installer will install the Anaconda package.
* Step 4 - for checking whether it is installed or not, open a command prompt and type Python as follows –

Text

Description automatically generated

## Why Python for Machine Learning?

Python is the fifth most important language as well as a most popular language for machine learning and data science. the following are the features of Python that makes it the preferred choice of language for data science –

**An extensive set of packages** - Python has an extensive and powerful set of packages that are ready to use in various domains. it also has packages like NumPy, Scipy, pandas, scikit-learn, etc. which are required for machine learning and data science.

**Easy prototyping** - another important feature of Python that makes it the choice of language for data science is easy and fast prototyping. this feature is useful for developing new algorithms.

**Collaboration features** - The field of Machine Learning basically needs good collaboration and Python provides many useful tools that make this extremely.

**One language for many tasks**- A typical machine learning project includes various tasks like data extraction, data manipulation, data analysis, feature extraction, modeling, evaluation, deployment, and updating the solution. as Python is a multi-purpose language, it allows machine learning to address all these tasks from a common platform.

## Components of Python ML Ecosystem

In this section, let us discuss some core machine learning libraries that form the components of the Python machine learning ecosystem. These useful components make Python an important language for machine learning and data science. Though there are many such components, let us discuss some of the important components of the Python ecosystem here –

**Jupyter notebook**

Jupyter notebooks basically provide an interactive computational environment for developing Python-based data science/machine learning applications. They are formerly known as iPython notebooks. The following are some of the features of Jupyter notebooks That makes it one of the best components of the Python ML ecosystem –

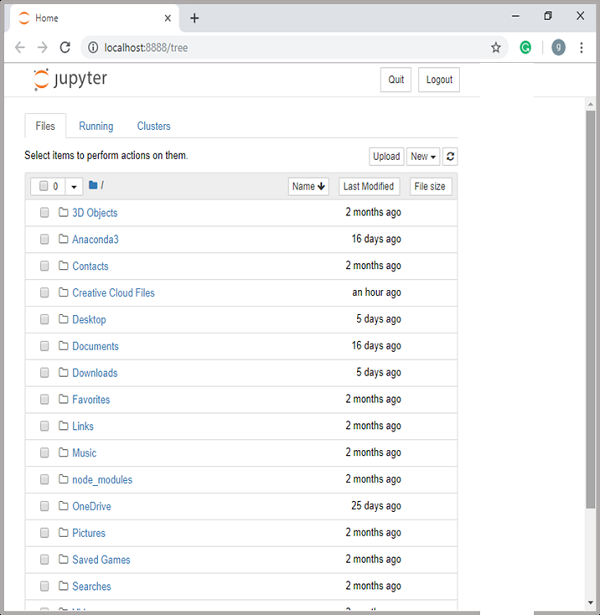
* Jupyter notebooks can illustrate the analysis process step by step by arranging the stuff like code, Images, text, output, etc. in a step-by-step manner.
* It helps data scientists to document the thought process while developing the analysis process.
* One can also capture the results as a part of the notebook.
* With the help of the Jupiter notebooks, we can share our work with a peer also.

**Installation and execution**

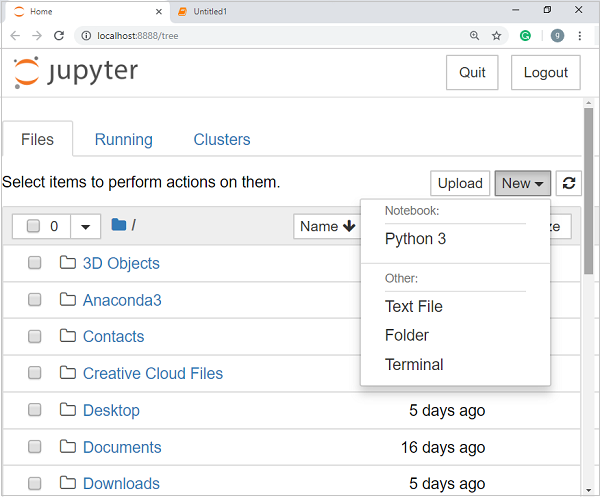
If you are using Anaconda distribution, then you need not install Jupiter notebook separately as it is already installed with it. You just need to go to the Anaconda prompt and type the following command –

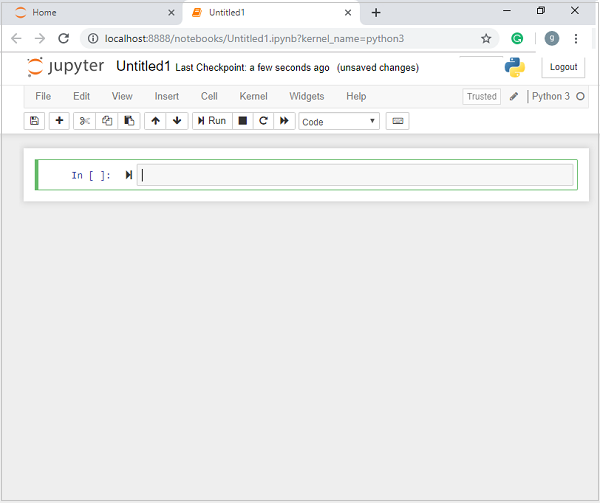


After pressing enter, it will start a notebook server at localhost:8888 of your computer. it is shown in the following screenshot –



Now, after clicking the new tab, you will get a list of options. select Python 3 and it will take you to the new notebook to start working in it. You will get a glimpse of it in the following screenshots





On the other hand, if you are using standard Python distribution then Jupyter notebook can be installed using the popular Python package installer, pip.



**Types of cells in Jupiter notebook**

The following are the three types of cells in the Jupyter notebook –

**Code cells** - As the name suggests, we can use these cells to write code. after writing the code/content, it will send it to the kernel that is associated with the notebook.

**Markdown cells** – We Can use the cells for notating the computation process. chicken contains stuff like text, images, Latex equations, HTML tags, etc.

**Row cells** - the text written in them is displayed as it is. these cells are basically used to add the text that we do not wish to be converted by the automatic conversion mechanism of Jupiter notebook.

**How to install python packages**

If you are using Anaconda distribution, there's no need to install most of the machine learning packages separately, as they are already installed with it. you just need to import the package into Python script. on the other hand, if you are using standard Python distribution then the packages can be installed using the popular Python package installer, pip.

For example:



We need to run the above code in the command prompt.

We can import the package into your python script with the help of the following –



Similarly, we can install all the required packages.

## Crash course in Python packages

Python is a high-level, dynamically typed multiparadigm programming language. Python code is often said to be almost like pseudo-code since it allows you to express very powerful ideas in very few lines of code while being very readable.

**Variables in Python**

In any programming language, the variable is nothing but just a name for identity. Like we all have our name and from our name, people will recognize oneself and identify oneself. So, Developers will use variables/ variable names to make the code more user-friendly, just like the English language so that no one will lose in the code. If any third person will see the code, they will be able to understand the code and the meaning behind it easily just by seeing useful variable names.

In a more technical term, we can say that variables in python are the values that will vary. A variable is used to store a value. According to the requirement, this value of a variable may change in the future.

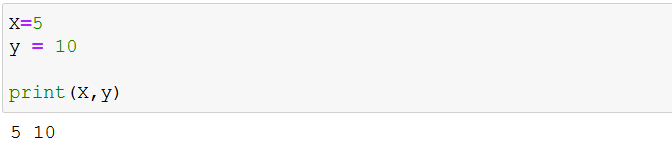
In python, there is no need of declaring a variable as soon as you assign something to a variable is called declared. explicitly we don't need any declaration.

**Defining**

A variable in Python is defined through the assignment. There is no concept of declaring a variable outside of that assignment.

Text

Description automatically generated with low confidence



For variable names, there are certain rules which we should follow as the best practice –

1. Variable name can't start with the number
2. Variable names are case-sensitive
3. don't use Python’s inbuilt/data type as a variable name.
4. Variable can only contain alphanumeric characters and underscores; no special characters are allowed.

### Fundamental data types in Python

**Integers**

Integer literals are created by any number without decimal or complex components. they can be represented as a whole number.

Graphical user interface, application, chat or text message

Description automatically generated

**Floats**

Float literals can be created by adding a decimal component to a number. when we need to display decimal points to a number, then we will use a float number.

Graphical user interface

Description automatically generated

**Boolean**

Boolean can be defined by typing True or False without quotes

A picture containing graphical user interface

Description automatically generated

**Strings**

String literals Can be defined with any single quotes, double quotes, or triple quotes. All give the same result with two important differences.

If you quote with single quotes, you don't have to escape double quotes and vice versa. if you could with triple quotes, your string can span multiple lines.

Graphical user interface, text, application, email

Description automatically generated

**Complex**

Complex literals can be created by using the notation x + yj where ‘x’ is the real component and y is the imaginary component.

Text

Description automatically generated with medium confidence

**Dynamic typing**

In Python, well the value that available points to has a type, available itself has no strict type in its definition. you can reuse the same variable to point to an object of a different type. it may be helpful to think of variables as “labels” associated with objects.

Chart

Description automatically generated with medium confidence

**Strong Typing**

Well, Python allows you to be very flexible with your types, you must still be aware of those types. The certain operation will require certain types as arguments.

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

This behavior is different from some other loosely typed language. In Python, however, it is possible to change the type of an object through built-in functions.

### Python Data Types

Variables are used to hold values of different data types. Python is a dynamically typed language; you don't need to define the type of variable while declaring it. the interpreter implicitly binds the value with its type. Python enables us to check the type of variable used in the program. with the help of the type() function, you can find out the type of the variable passed.

**Standard data types in Python**

Available you used to hold different types of values. for example, a person’s name must be stored as a string whereas an employee must be stored as an integer.

Python data types are broadly classified into two, Immutable and mutable.

**Immutable** - Immutable are the ones that cannot be changed once created. following are the immutable data types

1. **Numbers** - Number is here to store numeric values. Python creates a Number object when a number is assigned to a variable. There are four types of numeric data:
   1. **Int** - It is used for assigned integers like 12, 2, 7, etc.
   2. **Long** - This integer is used for a higher range of values like 908090800L, -0x1929292L, etc.
   3. **Float** - It is used to store floating-point numbers like 1.5, 701.89, 15.2, etc.
   4. **Complex** – This Is used for complex numbers like 2.14j, 2.0 + 2.3j, etc.
2. **String** - A string is defined as a sequence of characters represented in quotation marks. in Python, you can use single, double, or triple quotes to define a string.
3. **Tuple** – Tuple is an ordered collection of Python objects. A tuple is created by placing all the items inside the parenthesis (), separated by commas “,”. a tuple can have any number of items and they may be of different data types (int, float, list, string, etc)

Graphical user interface, text, application, email

Description automatically generated

A tuple Can be created without using parenthesis. this is known as tuple packing.

A picture containing graphical user interface

Description automatically generated

Tuple unpacking is also possible

Graphical user interface, text

Description automatically generated with medium confidence

Creating a tuple with one element is tricky.

Text

Description automatically generated

Having one element within parentheses is not enough. We need a trailing comma to indicate that it is in fact a tuple.

Text

Description automatically generated

Parenthesis is optional

Graphical user interface, text

Description automatically generated with medium confidence

Tuples are immutable, this means that elements of a tuple cannot be changed once it has been assigned. But, if the element is itself a mutable data type as a list, its nested items can be changed.

**Accessing elements of Tuples**

To access the tuple items, refer to the index number. Use the index operator [ ] to access an item in a tuple. The index must be an integer where the index starts from 0.

Text

Description automatically generated

**Mutable** – Mutable is which can be changed.

1. **List** – Lists are just like the arrays, declared in other languages which are an ordered collection of data. please be very flexible as items in a list do not need to be of the same type. Lists in Python can by just placing the sequence inside the square brackets [].

Let’s go ahead and see how we can construct a list!

A picture containing text

Description automatically generated

We just created a list of integers, but lists can hold different object types, different datatypes. For example:

Text

Description automatically generated with medium confidence

**Accessing an element of List**

In order to access the list items, refer to the index number. use index operator [ ] to access an element in a list. In Python, new to sequence indexes represent positions from the end of the array. Instead of having to compute the offset as in *List[len(list)-3]*, It is enough to just write *List[-3]*. Negative indexing means beginning from the end, -1 refers to the last element, -2 refers to the second-last element, etc.

Graphical user interface, application

Description automatically generated

1. **Dictionary** - Dictionary in Python is an unordered collection of data values, visit to store data values like a map, which, unlike other Data Types that hold only a single value as an element, dictionary holds *the key: value* pair. Key-Value is provided in the dictionary to make it more optimized. Each Key-Value pair in a dictionary is separated by a colon: whereas each key is separated by a “comma”.

**Creating Dictionary**

In Python, a dictionary can be created by placing a sequence of elements within curly { } braces, separated by “,”. values in the dictionary can be of any data type and can be duplicated., Whereas keys cannot be repeated and must be immutable. Dictionary can also be created by the built-in function *dist()*. An empty dictionary can be created by just placing it to braces {}.

Note - Dictionary keys are case sensitive, the same name but different cases of Key will be treated distinctly.

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

A picture containing text

Description automatically generated

1. **Sets** – Sets are an unordered collection of unique elements which can be constructed using the *set()* function.

**Graphical user interface, text, application

Description automatically generated**

Creating an empty set is a bit tricky. Empty curly braces {} will make an empty dictionary in python. To make a set without any elements we use the *set()* function without any arguments.

Graphical user interface, text, application

Description automatically generated

### Python libraries

Python libraries are a set of useful functions that eliminate the need for writing code from scratch. Python libraries play a vital role in developing machine learning, data science, data visualization, and more.

Learning machine learning can be old overwhelming. there are hundreds of tools and resources out there, and it's not always obvious what tools you should be focusing on or what should learn.

The short answer is that you should learn what you enjoy because machine learning offers a wide range of skills and tools. that being said, I wanted to share with you what I believe are most commonly used in machine learning/data science.

#### NumPy

NumPy Is mainly used for its support for N-dimensional arrays. These multi-dimensional arrays are 50 times more robust compared to python lists, making NumPy a favorite for data scientists.

NumPy is also used by other libraries such as TensorFlow for their internal computation on tensors. NumPy also provides fast precompiled functions for numerical routines, which can be hard to manually solve. To achieve better efficiency, NumPy uses array-oriented computations, so working with multiple classes becomes easy.

**1-Dimensional array**

Graphical user interface, application

Description automatically generated

**2-Dimensional array**

Graphical user interface, text, application, Word

Description automatically generated

**Initializing NumPy array of ‘x’\*’y’ dimension with zero**

Graphical user interface, application

Description automatically generated

**Arrange numbers between x and y with an interval of z**

Text

Description automatically generated

The above code returns the values between 1 and 10 in the interval of 2.



**Arrange z numbers between x and y**

Graphical user interface, text, application

Description automatically generated

The above code returns the 10 values between 5 and 15

**Filling the same number in an array of dimensions x and y**

Application

Description automatically generated with low confidence

The above code returns a NumPy array of dimension 2\*3 with all the values as 6.

**Filling random numbers in an array of dimensions x and y**

Text

Description automatically generated

The above code returned random values between 0 and 1.

**NumPy Array Inspection**

**Inspect the size of a NumPy array**

A picture containing graphical user interface

Description automatically generated

The size method returns the number of elements in an array.

**Inspect the shape of a NumPy array**

Graphical user interface

Description automatically generated with low confidence

The shaping method returns the dimension of the NumPy array.

**Reshape the NumPy array**

Graphical user interface, text, application, email

Description automatically generated

Note: We can not reshape the array randomly. The trick is x \* y = Total no. of elements of the array. We can only reshape the array based on the factors of Total no. of elements of the array.

Exp: the total no. of elements in an array is 6. The factors of 6 are (1x6, 6x1, 2x3, 3x2)

**Dimensions of the NumPy array**

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

**NumPy array Mathematical operations**

**Addition**

Graphical user interface, text, application, letter

Description automatically generated

What if I want to perform some operation row-wise or column-wise?

Graphical user interface, text, application, Word

Description automatically generated

**Subtraction**

Graphical user interface

Description automatically generated with medium confidence

**Multiplication**

Graphical user interface

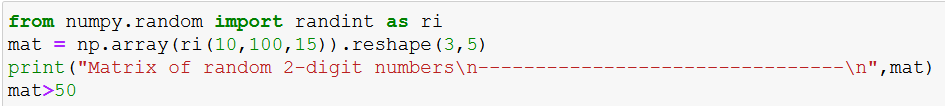
Description automatically generated with low confidence

**Division**

Graphical user interface, text

Description automatically generated with medium confidence

**Subsetting**



The 2nd line in the above code generates 15 random integers between 10 and 100 and reshape them into a 3x5 shape

After that, we used a condition which returns a True if the condition is met and a false if the condition is not met.

Text

Description automatically generated

Let’s subset the values which are greater than 50.

Graphical user interface, text

Description automatically generated

**Slicing**

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

#### Pandas

You’ve heard the saying. 70 to 80% of a data scientist’s job is understanding and cleaning the data, aka data exploration and data munging.

Pandas is primarily used for data analysis, and it is one of the most commonly used Python libraries. it provides you with some of the most useful set of tools to explore, clean, and analyze your data. with pandas, you can load, prepare, manipulate, and analyze all kinds of structured data. machine-learning libraries also revolve around pandas’ data frames as an input.

**Python Pandas Operations**

Using Python pandas can perform a lot of operations with series, data frames, missing data, group by, etc. some of the common operations for data manipulations are listed below

Diagram

Description automatically generated

Now, let us understand all these operations one by one.

**Slicing the data frame**

cannot perform slicing on data, you need a data frame. don't worry, data frame a two-dimensional data structure and a most common panda object. So first, let's create a data frame.

Text

Description automatically generated with low confidence

Table

Description automatically generated

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

Diagram

Description automatically generated

Table

Description automatically generated

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

Graphical user interface, application

Description automatically generated

Next in Python Pandas, let us perform merging and 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 have some key-value pairs, and then merge the data frames together. Refer to the code below:

Text

Description automatically generated

Table

Description automatically generated

As you can see above, the two data frames have 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 become common and for everything else, I want separate columns. so let me implement that practically:

Text

Description automatically generated with low confidence

Table

Description automatically generated

Next, let us understand joining in Python pandas. it is yet another convenient method to combine two differently indexed data frames into a single result date frame. this is quite similar to the “merge” operation, except the joint operation will be on the index instead of the “columns”. let us implement it practically.

Text

Description automatically generated

Table

Description automatically generated

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

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

**Concatenation**

Concatenation basically glues the data frames together. you can select the dimension on which you want to concatenate. for that, just use “pd.concat” and pass in the list of data frames to concatenate together. consider the below example.

Text

Description automatically generated

Table

Description automatically generated

As you can see above, the two data frames are glued together in a single data frame, where the index starts from 2001 all the way up to 2008. Next, you can also specify axis = 1 in order to join, merge or concatenate along with the columns, refer to the code below:

Text

Description automatically generated

Table

Description automatically generated

as you can see above, there are a bunch of missing values. this happens because the data frame didn't have values for all the indexes you want to concatenate on. Therefore, you should make sure that we have all the information lining up correctly when you join or concatenate on the axis.

**Change the index**

Next in the Python Pandas tutorial, we’ll understand how to change the index values in a data frame. For example, let us create a data frame with some key-value pairs in a dictionary and change the index values. Consider the below example:

let us see how it happens:

Text

Description automatically generated with low confidence

Table

Description automatically generated

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 the column in this Python pandas. let us take the same example, clear I will change the column headers from the “Visitors” to “Users”. So, let me implement it practically.

Text

Description automatically generated with low confidence

Table

Description automatically generated

As you see above, the column header “Visitors” has been changed to “Users”. Next in Python Pandas, 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.

Logo, company name

Description automatically generated

Table

Description automatically generated



Once you run this code, an HTML file will be created named “edu.html”. You can directly copy the path of the file and paste it into your browser which displays the data in an HTML format. Refer to the below screenshot:

Table

Description automatically generated

#### Matplotlib

Matplotlib Is a powerful tool for executing a variety of tasks. It is able to create different types of visualization reports like line plots, scatter plots, histograms, bar charts, line charts, box plots, and many more different plots. This library also supports three-dimensional plots.

**How to install Matplotlib**

You can install Matplotlib by using the PIP command.



To verify the installation you would have to write the following code chunk:

Graphical user interface, text, application

Description automatically generated

**Important types of plots in Matplotlib**

Now that you know what is Matplotlib and how you can install it in your system, let's discuss different kinds of plots that you can draw to analyze your data to present your findings.

**Sub Plots**-

Subplots() is a Matplotlib function that is used to display multiple plots in one figure. It contains various arguments such as a number of rows, columns, or sharex, sharey axis.

The layout is organized in rows and columns, which are represented by the first and second arguments.

The third argument represents the index of the current plot.

Code:





**Example:**

Draw 2 plots side by side

Graphical user interface, text, chat or text message

Description automatically generated

Chart, line chart

Description automatically generated

Draw 2 plots on top of each other:

Text, letter

Description automatically generated

Chart, line chart

Description automatically generated

So, if you want a figure with two rows in 1 column (meaning that the two plots will be displayed on top of each other instead of side by side), you can write the syntax like this.

Adding a title and Super tile to the figure

Graphical user interface, text, application, chat or text message

Description automatically generated

Chart, line chart

Description automatically generated

**Line Plot-**

To plot a line plot in Matplotlib, you use the generic plot() function from the PyPlot instance. there is no specific lineplot() function - the generic one automatically plots using lines or markers.

let's make our own small data set to work with:

A picture containing text

Description automatically generated

Chart, line chart

Description automatically generated

Let’s take another example to add labels and heading to the plot

Text

Description automatically generated

Chart, line chart

Description automatically generated

**Histogram-**

the most common graph for displaying frequency distributions is a histogram. To create a histogram the first step is to create a bin of ranges, Then distribute the whole range of values into a series of intervals, and count the value which will fall in the given interval. We can use plt.hist() function for plotting the histogram which will take various arguments like data, bins, color, etc.

Code:

Text, letter

Description automatically generated

Chart, bar chart

Description automatically generated

As you can see in the above plot, we got age groups with respect to the bins. our biggest age group is between 40 and 50.

**Bar plot-**

A bar graph uses bars to compare data among different categories. It is well suited when you want to measure the changes over a period of time. it can be represented horizontally or vertically. Also, the important thing to keep in mind is that the longer the bar, the greater is the value. Now, let us practical implement it using a Python Matplotlib.

Text

Description automatically generated

Chart, bar chart

Description automatically generated

In the above plot, I have displayed the comparison between the distance covered by two cars BMW and Audi over a period of five days. Next, let us move on to another kind of plot using Python Matplotlib.

**Scatter plot -**

Usually, we need scatter plots in order to compare continuous variables, for example, how much one variable is affected by another variable to build a relation out of it. The data is displayed as a collection of points, each having the value of one variable which determines the position on the horizontal axis and the value of the other variables determines the position on the vertical axis.

Example 1:

Text, letter

Description automatically generated

Chart, scatter chart

Description automatically generated

The plots show there is no relation between x and y variables.

Example 2-

**Text

Description automatically generated**

Chart, scatter chart

Description automatically generated

The relation between the x and y is negative. i.e., the values of y decrease as the values of x decrease.

**Box Plot -**

A Box plot is used to show the summary of the whole data set or all the numeric values in the data set. The summary contains minimum, first quartile, median, third quartile, and maximum. Also, the median is present between the first and the third quartile. Here x-axis contains the data values and y coordinates show the frequency distribution.

Example:

Text

Description automatically generated

Chart, box and whisker chart

Description automatically generated

**Contour Plot –**

Contour plots are away to show a three dimensional surface on a two-dimensional plane. it graphs 2 predictor variables x y on the y-axis and a response variable z as contours. These contours are sometimes called the z-slices or the iso-response values.

Text

Description automatically generated

Chart

Description automatically generated

**Importing data from different file types**

You must be able to load your data before you can start your machine learning project. The most common formats for machine learning data are CSV, TSV, Excel, URL, and Data Bases. Let’s look at it one by one.

**Reading CSV files**

Logo

Description automatically generated with low confidence

Graphical user interface, application

Description automatically generated

Another way of reading CSV file using “read\_table” and providing “sep” as a “,”

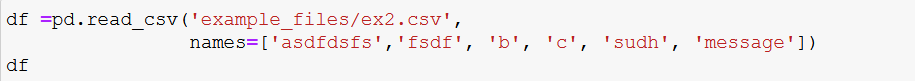


Reading CSV files without printing headers

Graphical user interface, text, application

Description automatically generated

Changing column names by providing names parameters



Graphical user interface, table

Description automatically generated

Use of index\_col to use as the row labels/indexes of the DataFrame

Graphical user interface

Description automatically generated with low confidence

**Reading TSV files (tab-separated values)**

Graphical user interface, text, application, email

Description automatically generated

**Reading Excel files**

Text

Description automatically generated with low confidence

Graphical user interface, application

Description automatically generated

**Reading data from URL**

A picture containing company name

Description automatically generated

A screenshot of a computer

Description automatically generated with low confidence

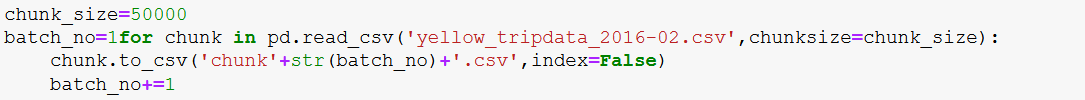
**Importing large files using chunking**

When we try to load the large data files system will throw the following Memory Error. (The memory error would depend upon the capacity of the system that you are using).

Text

Description automatically generated with low confidence

We shall look at the method called chunking, by which you can load out of memory datasets in pandas. This method can sometimes offer a healthy way out to manage the out-of-memory problem in pandas but may not work all the time.



We choose a chunk size of 50,000 which means at a time, only 50,000 rows of data will be imported.

**Interacting with Database**

We create a connection with the database, and we can interact with a database to retrieve and alter the database.

Text

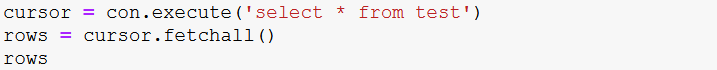
Description automatically generated

Text

Description automatically generated

Text

Description automatically generated



A picture containing letter

Description automatically generated

Text

Description automatically generated with medium confidence

Graphical user interface, application, Teams

Description automatically generated

Creating and the table and inserting the values into the table.

Table

Description automatically generated



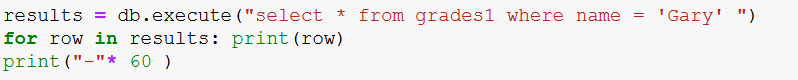
Retrieving the data

A picture containing chart

Description automatically generated

Background pattern

Description automatically generated with low confidence



A screenshot of a computer

Description automatically generated with medium confidence

A picture containing company name

Description automatically generated

Shape, rectangle

Description automatically generated

A picture containing text

Description automatically generated

Shape, rectangle

Description automatically generated

**Hands-on**

**Python practice**

**Step 1:** Import the necessary libraries

Text

Description automatically generated

**Step2:** Importing the dataset

Graphical user interface, text, application, email

Description automatically generated

**Step 3**: What is the number of observations in the dataset?

Solution1:

Graphical user interface, text, application, email

Description automatically generated

Solution2:

Text

Description automatically generated

**Step4: Print the name of all the columns**

**Graphical user interface, text, application

Description automatically generated**

**Step5: How is the dataset indexed?**

**Graphical user interface, application

Description automatically generated**

**Step6: Visualizing the data**

**Text

Description automatically generated**

**Chart, bar chart, histogram

Description automatically generated**

**Graphical user interface, text, application

Description automatically generated**

**Chart, scatter chart

Description automatically generated**

**SUMMARY**

In this chapter you discovered about the Python ecosystem for machine learning, you have covered a lot of ground about the basic syntax and usage of Python and three key python libraries used for machine learning and finally, we learned to import data from different files like CSV, TSV, Excel, and Databases.

**Assessment**

**Choose the appropriate option**

1. What is output for – b = [11, 13, 15, 17, 19, 21], print(b[::2])
   1. [19, 21]
   2. [11, 15]
   3. [11, 15, 19]
   4. [13, 17, 21]
2. Which of these is not a core data type?
   1. Lists
   2. Dictionary
   3. Tuples
   4. Class
3. What data type is the object below? L = [1, 23, ‘hello’, 1]
   1. List
   2. Dictionary
   3. Tuple
   4. Array
4. Which of the following function convert a string to a float in python?
   1. Int(x[,base])
   2. Long(x,[,base])
   3. Float(x)
   4. Str(x)
5. Guess the correct output of the following string operations

A picture containing shape

Description automatically generated

* 1. WelcomeWelcome
  2. Welcome2
  3. TypeError
  4. None of the above

**Fill in the spaces with appropriate answers**

1. Which method should I use to convert String “welcome to the beautiful world of python” to “Welcome To The Beautiful World Of Python”
2. NumPY stands for?
3. If a dimension is given as \_\_\_\_\_\_\_\_\_\_ in a reshaping operation, the other dimensions are automatically calculated.
4. All pandas data structures are \_\_\_\_\_\_\_\_\_\_\_\_ mutable but not always \_\_\_\_\_\_\_\_ mutable.
5. The plot which tells the correlation between two variables that may not be directly related is \_\_\_\_\_\_\_\_\_\_\_ graph/chart.

**True or False**

1. The “string” is immutable in Python, which means a string cannot be modified
   1. True
   2. False
2. What is the output of *print(type({ }) is set)*
   1. True
   2. False
3. It is possible to convert the NumPY array to a list in Python.
   1. Ture
   2. False
4. Series is a one-dimensional labeled array capable of holding any data type.
   1. True
   2. False
5. If data is a ndarray, an index must be the same length as the data.
   1. True
   2. False

**Programming Assessment**

Using the dataset for the below link. Perform the following tasks

<https://github.com/fenago/MLBook/blob/main/Chapter%201/code/Dataset/Occupation.txt>

1. Import the dataset
2. See the last 10 entries
3. What is the number of columns in the dataset?
4. What is the data type of each column?
5. Visualize the distribution of Occupation and age columns using histogram
6. Visualize gender using bar diagram

**Assessment Solutions**

**Choose the appropriate option**

1. C
2. D
3. A
4. C
5. A

**Fill in the spaces with appropriate answers**

1. Capitalize()
2. Numerical Python
3. Negative one
4. Value, Size
5. Scatter

**True or False**

1. True
2. False
3. True
4. True
5. True