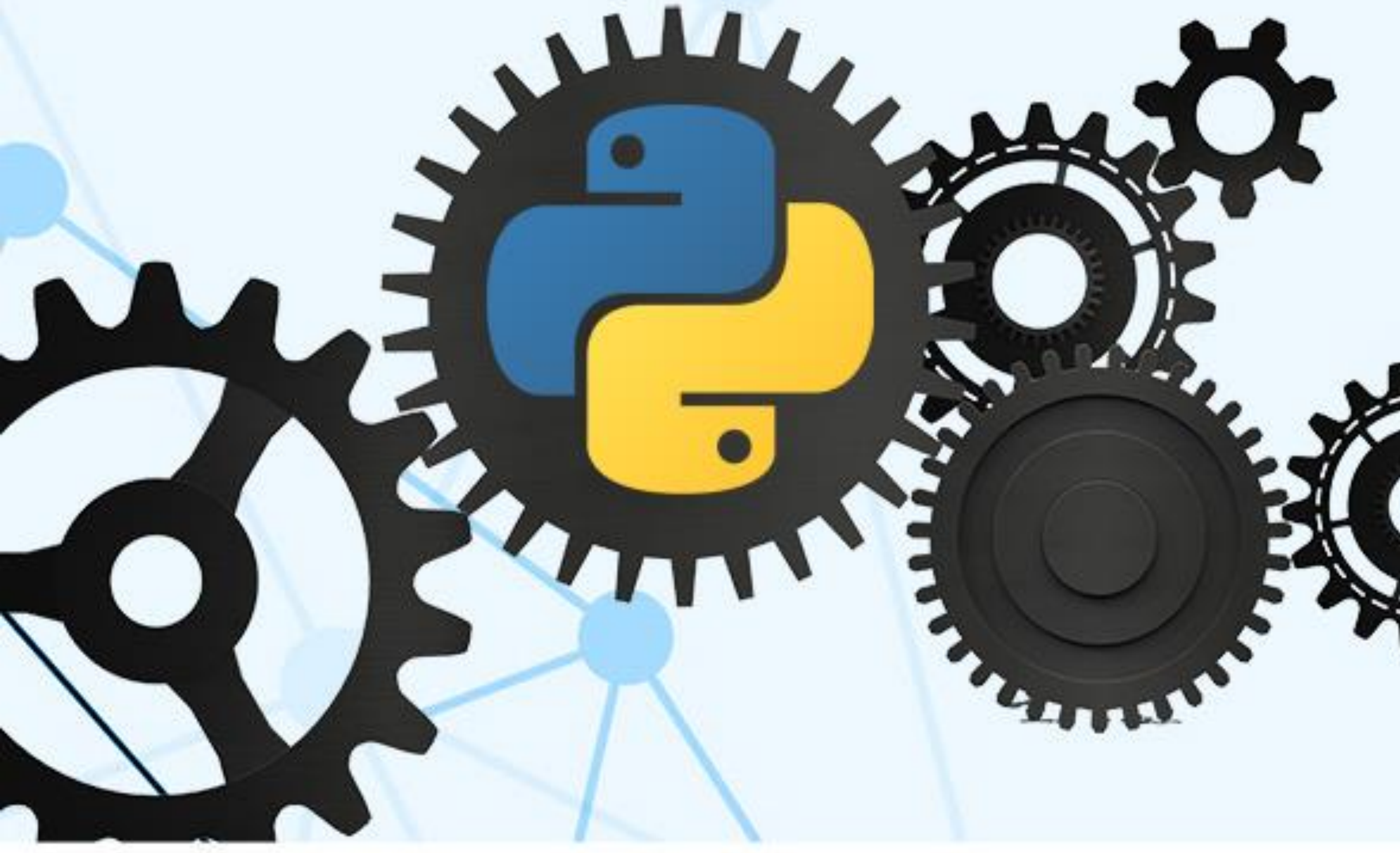
Machine Learning with Python



tutorialspoint

SIMPLY EASY LEARNING

www.tutorialspoint.com

https://www.facebook.com/tutorialspointindia https://twitter.com/tutorialspoint

About the Tutorial

Python is a general-purpose high level programming language that is being increasingly used in data science and in designing machine learning algorithms. This tutorial provides a quick introduction to Python and its libraries like numpy, scipy, pandas, matplotlib and explains how it can be applied to develop machine learning algorithms that solve real world problems.

This tutorial starts with an introduction to machine learning and the Python language and shows you how to setup Python and its packages. It further covers all important concepts such as exploratory data analysis, data preprocessing, feature extraction, data visualization and clustering, classification, regression and model performance evaluation.

This tutorial also provides various projects that teaches you the techniques and functionalities such as news topic classification, spam email detection, online ad click-through prediction, stock prices forecast and other several important machine learning algorithms.

Audience

This tutorial has been prepared for professionals aspiring to learn the basics of Python and develop applications involving machine learning techniques such as recommendation, classification, and clustering. Through this tutorial, you will learn to solve data-driven problems and implement your solutions using the powerful yet simple programming language, Python and its packages. After completing this tutorial, you will gain a broad picture of the machine learning environment and the best practices for machine learning techniques.

Prerequisites

Before you start proceeding with this tutorial, we assume that you have a prior exposure to Python, Numpy, pandas, scipy, matplotlib, Windows and any of the Linux operating system flavors. If you are new to any of these concepts, we recommend you to take up tutorials concerning these topics, before you dig further into this tutorial.

Copyright & Disclaimer

Table of Contents

1. Python Machine Learning – Introduction

Python is a popular platform used for research and development of production systems. It is a vast language with number of modules, packages and libraries that provides multiple ways of achieving a task.

Python and its libraries like NumPy, SciPy, Scikit-Learn, Matplotlib are used in data science and data analysis. They are also extensively used for creating scalable machine learning algorithms. Python implements popular machine learning techniques such as Classification, Regression, Recommendation, and Clustering.

Python offers ready-made framework for performing data mining tasks on large volumes of data effectively in lesser time. It includes several implementations achieved through algorithms such as linear regression, logistic regression, Naïve Bayes, k-means, K nearest neighbor, and Random Forest.

2. Python Machine Learning – Concepts

In this chapter, you will learn in detail about the concepts of Python in machine learning.

Python in Machine Learning

Python has libraries that enables developers to use optimized algorithms. It implements popular machine learning techniques such as recommendation, classification, and clustering. Therefore, it is necessary to have a brief introduction to machine learning before we move further.

What is Machine Learning?

Data science, machine learning and artificial intelligence are some of the top trending topics in the tech world today. Data mining and Bayesian analysis are trending and this is adding the demand for machine learning. This tutorial is your entry into the world of machine learning.

Machine learning is a discipline that deals with programming the systems so as to make them automatically learn and improve with experience. Here, learning implies recognizing and understanding the input data and taking informed decisions based on the supplied data. It is very difficult to consider all the decisions based on all possible inputs. To solve this problem, algorithms are developed that build knowledge from a specific data and past experience by applying the principles of statistical science, probability, logic, mathematical optimization, reinforcement learning, and control theory.

Applications of Machine Learning Algorithms

The developed machine learning algorithms are used in various applications such as:

- Vision processing
- Language processing
- Forecasting things like stock market trends, weather
- Pattern recognition
- Games
- Data mining
- Expert systems
- Robotics

Steps Involved in Machine Learning

A machine learning project involves the following steps:

- Defining a Problem
- Preparing Data
- Evaluating Algorithms
- Improving Results
- Presenting Results

The best way to get started using Python for machine learning is to work through a project end-to-end and cover the key steps like loading data, summarizing data, evaluating algorithms and making some predictions. This gives you a replicable method that can be used dataset after dataset. You can also add further data and improve the results.

3. Python Machine Learning – Environment Setup

In this chapter, you will learn how to setup the working environment for Python machine learning on your local computer.

Libraries and Packages

To understand machine learning, you need to have basic knowledge of Python programming. In addition, there are a number of libraries and packages generally used in performing various machine learning tasks as listed below:

- **numpy** - is used for its N-dimensional array objects
- **pandas** – is a data analysis library that includes dataframes
- **matplotlib** – is 2D plotting library for creating graphs and plots
- **scikit-learn** - the algorithms used for data analysis and data mining tasks
- **seaborn** – a data visualization library based on matplotlib

Installation

You can install software for machine learning in any of the two methods as discussed here:

Method 1

Download and install Python separately from **python.org** on various operating systems as explained below:

To install Python after downloading, double click the **.exe** (for Windows) or **.pkg** (for Mac) file and follow the instructions on the screen.

For Linux OS, check if Python is already installed by using the following command at the prompt:

```
$ python --version. ...
```

If Python 2.7 or later is not installed, install Python with the distribution's package manager. Note that the command and package name varies.

On Debian derivatives such as Ubuntu, you can use **apt**:

```
$ sudo apt-get install python3
```

Now, open the command prompt and run the following command to verify that Python is installed correctly:

```
$ python3 --version
```

```
Python 3.6.2
```

Similarly, we can download and install necessary libraries like numpy, matplotlib etc. individually using installers like **pip**. For this purpose, you can use the commands shown here:

```
$pip install numpy

$pip install matplotlib

$pip install pandas

$pip install seaborn
```

Method 2

Alternatively, to install Python and other scientific computing and machine learning packages simultaneously, we should install **Anaconda** distribution. It is a Python implementation for Linux, Windows and OSX, and comprises various machine learning packages like numpy, scikit-learn, and matplotlib. It also includes **Jupyter Notebook**, an interactive Python environment. We can install Python 2.7 or any 3.x version as per our requirement.

To download the free Anaconda Python distribution from Continuum Analytics, you can do the following:

Visit the official site of Continuum Analytics and its download page. Note that the installation process may take 15-20 minutes as the installer contains Python, associated packages, a code editor, and some other files. Depending on your operating system, choose the installation process as explained here:

For Windows: Select the **Anaconda for Windows** section and look in the column with Python 2.7 or 3.x. You can find that there are two versions of the installer, one for 32-bit Windows, and one for 64-bit Windows. Choose the relevant one.

For Mac OS: Scroll to the **Anaconda for OS X** section. Look in the column with Python 2.7 or 3.x. Note that here there is only one version of the installer: the 64-bit version.

For Linux OS: We select the "Anaconda for Linux" section. Look in the column with Python 2.7 or 3.x.

Note that you have to ensure that Anaconda's Python distribution installs into a single directory, and does not affect other Python installations, if any, on your system.

To work with graphs and plots, we will need these Python library packages: **matplotlib** and **seaborn**.

If you are using Anaconda Python, your system already has numpy, matplotlib, pandas, seaborn, etc. installed. We start the Anaconda Navigator to access either Jupyter Note book or Spyder IDE of python.

After opening either of them, type the following commands:

```
import numpy
import matplotlib
```

Now, we need to check if installation is successful. For this, go to the command line and type in the following command:

```
$ python
Python 3.6.3 |Anaconda custom (32-bit)| (default, Oct 13 2017, 14:21:34)
[GCC 7.2.0] on linux
```

Next, you can import the required libraries and print their versions as shown:

```
>>>import numpy
>>>print numpy.__version__
1.14.2
>>> import matplotlib
>>> print (matplotlib.__version__)
2.1.2
>> import pandas
>>> print (pandas.__version__)
0.22.0
>>> import seaborn
>>> print (seaborn.__version__)
0.8.1
```

4. Python Machine Learning – Types of Learning

Machine Learning (ML) is an automated learning with little or no human intervention. It involves programming computers so that they learn from the available inputs. The main purpose of machine learning is to explore and construct algorithms that can learn from the previous data and make predictions on new input data.

The **input** to a learning algorithm is training data, representing experience, and the **output** is any expertise, which usually takes the form of another algorithm that can perform a task. The input data to a machine learning system can be numerical, textual, audio, visual, or multimedia. The corresponding output data of the system can be a floating-point number, for instance, the velocity of a rocket, an integer representing a category or a class, for example, a pigeon or a sunflower from image recognition.

In this chapter, we will learn about the training data our programs will access and how learning process is automated and how the success and performance of such machine learning algorithms is evaluated.

Concepts of Learning

Learning is the process of converting experience into expertise or knowledge.

Learning can be broadly classified into three categories, as mentioned below, based on the nature of the learning data and interaction between the learner and the environment.

- Supervised Learning
- Unsupervised Learning
- Semi-supervised learning

Similarly, there are four categories of machine learning algorithms as shown below:

- Supervised learning algorithm
- Unsupervised learning algorithm
- Semi-supervised learning algorithm
- Reinforcement learning algorithm

However, the most commonly used ones are **supervised** and **unsupervised learning**.

Supervised Learning

Supervised learning is commonly used in real world applications, such as face and speech recognition, products or movie recommendations, and sales forecasting. Supervised learning can be further classified into two types: **Regression** and **Classification**.

Regression trains on and predicts a continuous-valued response, for example predicting real estate prices.

Classification attempts to find the appropriate class label, such as analyzing positive/negative sentiment, male and female persons, benign and malignant tumors, secure and unsecure loans etc.

In supervised learning, learning data comes with description, labels, targets or desired outputs and the objective is to find a general rule that maps inputs to outputs. This kind of learning data is called **labeled data**. The learned rule is then used to label new data with unknown outputs.

Supervised learning involves building a machine learning model that is based on **labeled samples**. For example, if we build a system to estimate the price of a plot of land or a house based on various features, such as size, location, and so on, we first need to create a database and label it. We need to teach the algorithm what features correspond to what prices. Based on this data, the algorithm will learn how to calculate the price of real estate using the values of the input features.

Supervised learning deals with learning a function from available training data. Here, a learning algorithm analyzes the training data and produces a derived function that can be used for mapping new examples. There are many **supervised learning algorithms** such as Logistic Regression, Neural networks, Support Vector Machines (SVMs), and Naive Bayes classifiers.

Common **examples** of supervised learning include classifying e-mails into spam and not-spam categories, labeling webpages based on their content, and voice recognition.

Unsupervised Learning

Unsupervised learning is used to detect anomalies, outliers, such as fraud or defective equipment, or to group customers with similar behaviors for a sales campaign. It is the opposite of supervised learning. There is no labeled data here.

When learning data contains only some indications without any description or labels, it is up to the coder or to the algorithm to find the structure of the underlying data, to discover hidden patterns, or to determine how to describe the data. This kind of learning data is called **unlabeled data**.

Suppose that we have a number of data points, and we want to classify them into several groups. We may not exactly know what the criteria of classification would be. So, an unsupervised learning algorithm tries to classify the given dataset into a certain number of groups in an optimum way.

Unsupervised learning algorithms are extremely powerful tools for analyzing data and for identifying patterns and trends. They are most commonly used for clustering similar input into logical groups. Unsupervised learning algorithms include Kmeans, Random Forests, Hierarchical clustering and so on.

Semi-supervised Learning

If some learning samples are labeled, but some other are not labeled, then it is semi-supervised learning. It makes use of a large amount of **unlabeled data for training** and a small amount of **labeled data for testing**. Semi-supervised learning is applied in cases where it is expensive to acquire a fully labeled dataset while more practical to label a small subset. For example, it often requires skilled experts to label certain remote sensing images, and lots of field experiments to locate oil at a particular location, while acquiring unlabeled data is relatively easy.

Reinforcement Learning

Here learning data gives feedback so that the system adjusts to dynamic conditions in order to achieve a certain objective. The system evaluates its performance based on the feedback responses and reacts accordingly. The best known instances include self-driving cars and chess master algorithm AlphaGo.

Purpose of Machine Learning

Machine learning can be seen as a branch of AI or Artificial Intelligence, since, the ability to change experience into expertise or to detect patterns in complex data is a mark of human or animal intelligence.

As a field of science, machine learning shares common concepts with other disciplines such as statistics, information theory, game theory, and optimization.

As a subfield of information technology, its objective is to program machines so that they will learn.

However, it is to be seen that, the purpose of machine learning is not building an automated duplication of intelligent behavior, but using the power of computers to complement and supplement human intelligence. For example, machine learning programs can scan and process huge databases detecting patterns that are beyond the scope of human perception.

5. Python Machine Learning – Data Preprocessing, Analysis and Visualization

In the real world, we usually come across lots of raw data which is not fit to be readily processed by machine learning algorithms. We need to preprocess the raw data before it is fed into various machine learning algorithms. This chapter discusses various techniques for preprocessing data in Python machine learning.

Data Preprocessing

In this section, let us understand how we preprocess data in Python.

Initially, open a file with a **.py** extension, for example **prefoo.py** file, in a text editor like notepad.

Then, add the following piece of code to this file:

```
import numpy as np

from sklearn import preprocessing

#We imported a couple of packages. Let's create some sample data and add the
line to this file:

input_data = np.array([[3, -1.5, 3, -6.4], [0, 3, -1.3, 4.1], [1, 2.3, -2.9, -
4.3]])
```

We are now ready to operate on this data.

Preprocessing Techniques

Data can be preprocessed using several techniques as discussed here:

Mean removal

It involves removing the mean from each feature so that it is centered on zero. Mean removal helps in removing any bias from the features.

You can use the following code for mean removal:

```
data_standardized = preprocessing.scale(input_data)

print "\nMean =", data_standardized.mean(axis=0)

print "Std deviation =", data_standardized.std(axis=0)
```

Now run the following command on the terminal:

```
$ python prefoo.py
```

You can observe the following output:

```
Mean = [  5.55111512e-17  -3.70074342e-17   0.00000000e+00  -1.85037171e-17]

Std deviation = [1.  1.  1.  1.]
```

Observe that in the output, mean is almost 0 and the standard deviation is 1.

Scaling

The values of every feature in a data point can vary between random values. So, it is important to scale them so that this matches specified rules.

You can use the following code for scaling:

```
data_scaler = preprocessing.MinMaxScaler(feature_range=(0, 1))

data_scaled = data_scaler.fit_transform(input_data)

print "\nMin max scaled data =", data_scaled
```

Now run the code and you can observe the following output:

```
Min max scaled data = [[ 1.          0.          1.          0.        ]

 [ 0.          1.          0.27118644  1.        ]

 [ 0.33333333  0.84444444  0.          0.2       ]]
```

Note that all the values have been scaled between the given range.

Normalization

Normalization involves adjusting the values in the feature vector so as to measure them on a common scale. Here, the values of a feature vector are adjusted so that they sum up to 1. We add the following lines to the prefoo.py file:

You can use the following code for normalization:

```
data_normalized = preprocessing.normalize(input_data, norm='l1')

print "\nL1 normalized data =", data_normalized
```

Now run the code and you can observe the following output:

```
L1 normalized data = [[ 0.21582734 -0.10791367  0.21582734 -0.46043165]

 [ 0.          0.35714286 -0.1547619   0.48809524]

 [ 0.0952381   0.21904762 -0.27619048 -0.40952381]]
```

Normalization is used to ensure that data points do not get boosted due to the nature of their features.

Binarization

Binarization is used to convert a numerical feature vector into a Boolean vector. You can use the following code for binarization:

```
data_binarized = preprocessing.Binarizer(threshold=1.4).transform(input_data)
print "\nBinarized data =", data_binarized
```

Now run the code and you can observe the following output:

```
Binarized data = [[ 1.  0.  1.  0.]
 [ 0.  1.  0.  1.]
 [ 0.  1.  0.  0.]]
```

This technique is helpful when we have prior knowledge of the data.