

به نام خدا

کلاس کاربرد هوش مصنوعی در مدیریت | دکتر خادمی

مروری بر مطالب:

۱- کلیات کار با پایتون

۲- شبکه عصبی در پایتون

۳- الگوریتم ژنتیک

۴- یک نمونه الگوریتم PSO

اول: کلیات کار با پایتون

ویژگی های زبان پایتون

۱- زبان تفسیری

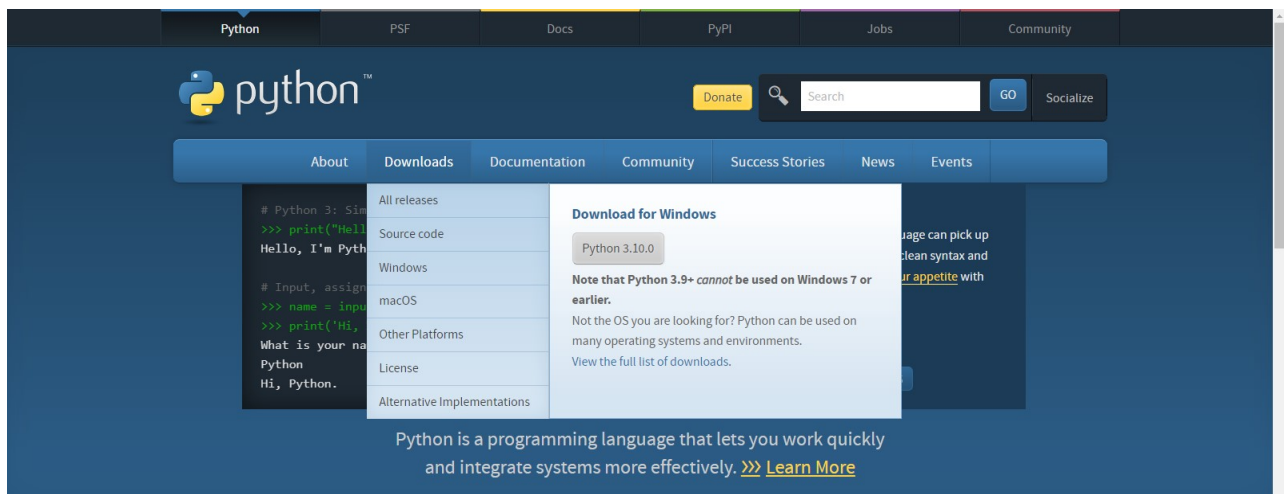
۲- نوشتار آسان

۳- خوانایی بالا

۴- زبان شیء گرا

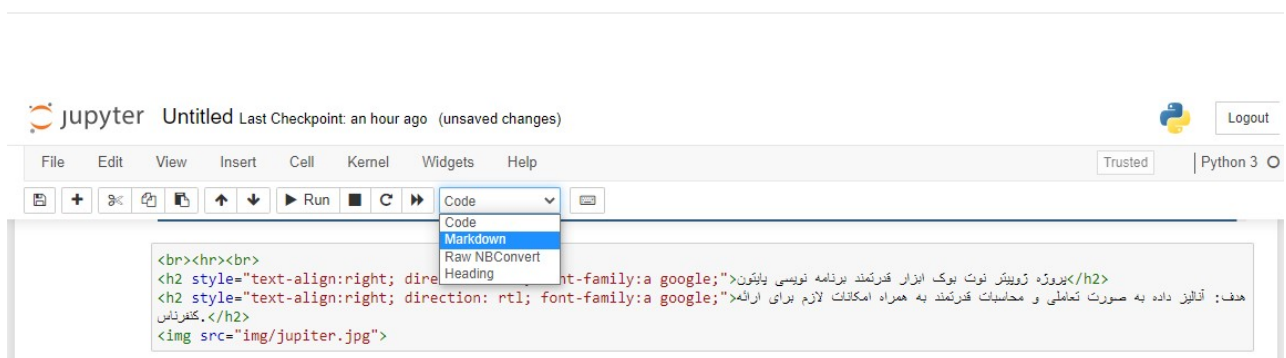
۵- کاربردی ترین ابزارهای هوش مصنوعی

شیوه نصب پایتون بر روی ویندوز - python.org



پروژه ژوپیتر نوت بوک ابزار قدرتمند برنامه نویسی پایتون

هدف: آنالیز داده به صورت تعاملی و محاسبات قدرتمند به همراه امکانات لازم برای ارائه کنفرانس.



شیوه نصب ژوپیتر نوت بوک در ویندوز - anaconda.com



Individual Edition

# Your data science toolkit

With over 25 million users worldwide, the open-source Individual Edition (Distribution) is the easiest way to perform Python/R data science and machine learning on a single machine. Developed for solo practitioners, it is the toolkit that equips you to work with thousands of open-source packages and libraries.



Anaconda Navigator  
File Help

ANACONDA NAVIGATOR

Sign in

Home

Environments

Learning

Community



Applications on base (root)

Channels

Refresh

<p>CMD.exe Prompt 0.1.1</p> <p>Run a cmd.exe terminal with your current environment from Navigator activated</p> <p>Launch</p>	<p>Datalore</p> <p>Online Data Analysis Tool with smart coding assistance by JetBrains. Edit and run your Python notebooks in the cloud and share them with your team.</p> <p>Launch</p>	<p>IBM Watson Studio Cloud</p> <p>IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build models, using open source data science tools or visual modeling.</p> <p>Launch</p>	<p>JupyterLab 2.2.6</p> <p>An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.</p> <p>Launch</p>	<p>Jupyter Notebook 6.1.4</p> <p>Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.</p> <p>Launch</p>
<p>Powershell Prompt 0.0.1</p> <p>Launch</p>	<p>Qt Console 4.7.7</p> <p>Launch</p>	<p>Spyder 4.1.5</p> <p>Launch</p>	<p>VS Code 1.62.3</p> <p>Launch</p>	<p>Glueviz 1.0.0</p> <p>Launch</p>

فراگیری سریع پایتون در W3SCHOOLS.COM/PYTHON

[HOME](#)[HTML](#)[CSS](#)[JAVASCRIPT](#)[SQL](#)[PYTHON](#)[PHP](#)[BOOTSTRAP](#)[HOW TO](#)[W3.CSS](#)[JAVA](#)[JQUERY](#)[C++](#)[C#](#)[R](#)[React](#)[Kotlin](#)

Python Tutorial

Python HOME

Python Intro

Python Get Started

Python Syntax

Python Comments

Python Variables

Python Data Types

Python Numbers

Python Casting

Python Strings

Python Booleans

Python Operators

Python Lists

Python Tuples

Python Sets

Python Dictionaries

**Python If...Else**

Python While Loops

Python For Loops

Python Functions

Python Lambda

Python Arrays

Python Classes/Objects

Python Inheritance

Python Iterators

## Python Conditions and If statements

Python supports the usual logical conditions from mathematics:

- Equals: `a == b`
- Not Equals: `a != b`
- Less than: `a < b`
- Less than or equal to: `a <= b`
- Greater than: `a > b`
- Greater than or equal to: `a >= b`

These conditions can be used in several ways, most commonly in "if statements" and loops.

An "if statement" is written by using the `if` keyword.

### Example

If statement:

```
a = 33
b = 200
if b > a:
    print("b is greater than a")
```

Try it Yourself »

تنظیم

آسان

Unmetered Connections

Best Encryption

کهولیس

NEW

We just launched W3Schools videos

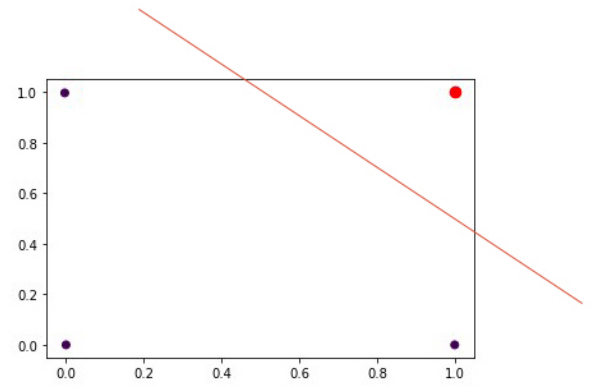
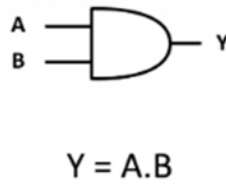
## دوم: شبکه عصبی در پایتون

فلسفه شبکه عصبی و سایر مدل‌های هوش مصنوعی



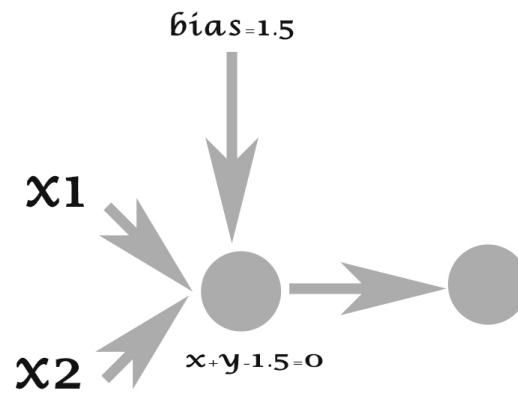
پیاده سازی گیت اند با شبکه پرسپترون

Inputs		Output
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



معادله خط

$$y - y_1 = m(x - x_1) \Rightarrow y + x - 1.5 = 0$$



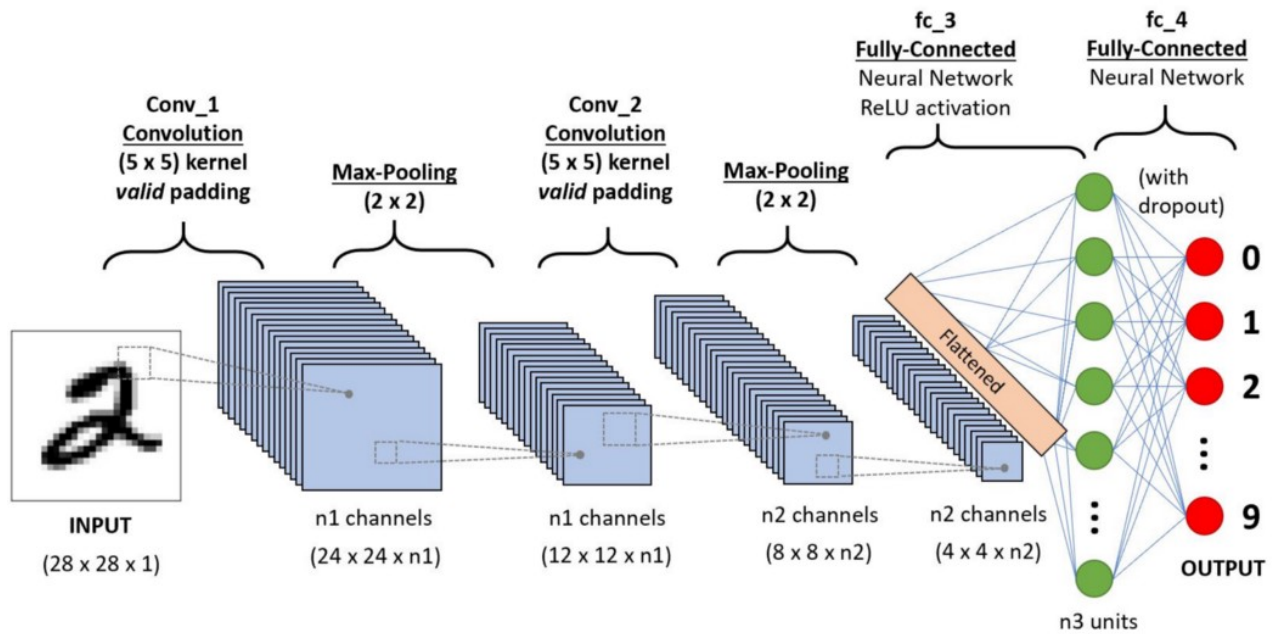
```
In [31]: from sklearn.linear_model import Perceptron
import matplotlib.pyplot as plt
import numpy as np
from itertools import product

data = [[0, 0], [0, 1], [1, 0], [1, 1]]
labels = [0, 0, 0, 1]

classifier = Perceptron(max_iter = 40)
classifier.fit(data, labels)
print("Accuracy = "+str(classifier.score(data, labels)))

Accuracy = 1.0
```

## نمونه ساخت شبکه عصبی به صورت دستی



### Model Building

In [16]:

```
from tensorflow.keras import Sequential
model = Sequential()

# Conv2D(number of filters, (size of kernel), padding_same="28*28 output
# , activation function="")
model.add(Conv2D(32, (3, 3), padding = 'same', input_shape=(28, 28, 1), activation = 'relu'))
model.add(MaxPool2D((2, 2)))


model.add(Conv2D(64, (3, 3), padding = 'same', activation = 'relu'))
model.add(MaxPool2D((2, 2)))

model.add(Conv2D(128, (3, 3), padding = 'same', activation = 'relu'))
model.add(MaxPool2D((2, 2)))

model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dense(24, activation="softmax"))
```


سوم: الگوریتم ژنتیک در پایتون

استفاده از کتابخانه ژنتیک الگوریتم



HelpSponsorsLog inRegister

# geneticalgorithm 1.0.2

`pip install geneticalgorithm`

Released: Dec 28, 2020

An easy implementation of genetic-algorithm (GA) to solve continuous and combinatorial optimization problems with real, integer, and mixed variables in Python

Navigation

Project description

Release history

Download files

Project links

Homepage

Statistics

Project description

## geneticalgorithm

geneticalgorithm is a Python library distributed on [Pypi](#) for implementing standard and elitist [genetic-algorithm](#) (GA). This package solves continuous, [combinatorial](#) and mixed [optimization](#) problems with continuous, discrete, and mixed variables. It provides an easy implementation of genetic-algorithm (GA) in Python.

Installation

Use the package manager [pip](#) to install geneticalgorithm in Python.

```
pip install geneticalgorithm
```

## یک مثال ساده بهینه سازی

فرض کنید می خواهیم مجموعه ای از  $X=(x_1,x_2,x_3)$  را پیدا کنیم که تابع  $f(X)=x_1+x_2+x_3$  را کمینه می کند و در آن  $X$  می تواند هر عدد طبیعی در بازه  $[0,10]$  باشد.

به راحتی می توان دریافت که پاسخ آن  $X=(0,0,0)$  است.

```
In [35]: import numpy as np
from geneticalgorithm import geneticalgorithm as ga

def f(X):
    return np.sum(X)

varbound=np.array([[0,10]]*3)

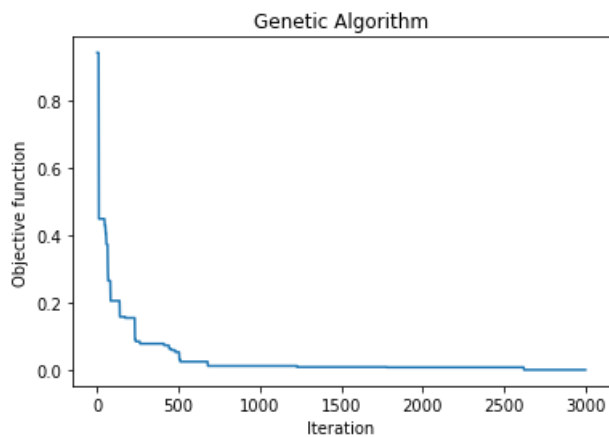
algorithm_param = {'max_num_iteration': 3000,\
                    'population_size':100,\
                    'mutation_probability':0.1,\
                    'elit_ratio': 0.01,\
                    'crossover_probability': 0.5,\
                    'parents_portion': 0.3,\
                    'crossover_type':'uniform',\
                    'max_iteration_without_improv':None}

model=ga(function=f,\
          dimension=3,\
          variable_type='real',\
          variable_boundaries=varbound,\
          algorithm_parameters=algorithm_param)

model.run()
```

The best solution found:  
[0.00147485 0.00021725 0.00060048]

Objective function:  
0.0022925769423254305



## چهارم: یک نمونه الگوریتم PSO

استفاده از کتابخانه pyswarm برای مسئله PSO



```

In [42]: import numpy as np
from pyswarm import pso

# Define the objective (to be minimize)
def weight(x, *args):
    H, d, t = x
    B, rho, E, P = args
    return rho*2*np.pi*d*t*np.sqrt((B/2)**2 + H**2)

# Setup the constraint functions
def yield_stress(x, *args):
    H, d, t = x
    B, rho, E, P = args
    return (P*np.sqrt((B/2)**2 + H**2))/(2*t*np.pi*d*H)

def buckling_stress(x, *args):
    H, d, t = x
    B, rho, E, P = args
    return (np.pi**2*E*(d**2 + t**2))/(8*((B/2)**2 + H**2))

def deflection(x, *args):
    H, d, t = x
    B, rho, E, P = args
    return (P*np.sqrt((B/2)**2 + H**2)**3)/(2*t*np.pi*d*H**2*E)

def constraints(x, *args):
    strs = yield_stress(x, *args)
    buck = buckling_stress(x, *args)
    defl = deflection(x, *args)
    return [100 - strs, buck - strs, 0.25 - defl]

# Define the other parameters
B = 60 # inches
rho = 0.3 # lb/in^3
E = 30000 # kpsi (1000-psi)
P = 66 # kip (1000-lbs, force)
args = (B, rho, E, P)

# Define the lower and upper bounds for H, d, t, respectively
lb = [10, 1, 0.01]
ub = [30, 3, 0.25]

xopt, fopt = pso(weight, lb, ub, f_ieqcons=constraints, args=args)

# The optimal input values are approximately
# xopt = [29, 2.4, 0.06]
# with function values approximately
# weight = 12 lbs
# yield stress = 100 kpsi (binding constraint)
# buckling stress = 150 kpsi
# deflection = 0.2 in

```

Stopping search: Swarm best objective change less than 1e-08