COMP9414 Artificial Intelligence

Tutorial week 1 – Python introduction

1 Python Introduction

Python was conceived in the late 1980s by Guido van Rossum at Centrum Wiskunde & Informatica (CWI) in the Netherlands [1]. During the past decade, Python turned into a popular programming language known for its simplicity and readability. It is widely used in various domains, including artificial intelligence (AI). Python's versatility and extensive libraries make it an excellent choice for AI development.

- **What can Python do [2]?**
  1. Python can be used on a server to create web applications.
  2. Python can be used alongside software to create workflows.
  3. Python can connect to database systems. It can also read and modify files.
  4. Python can be used to handle big data and perform complex mathematics.
  5. Python can be used for algorithm development, optimization, and encoding.
- **Why Python?**
  1. Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc) (flexibility).
  2. Python has a simple syntax similar to the English language (readability).
  3. Python has a syntax that allows developers to write programs with fewer lines than other programming languages.
  4. Python runs on an interpreter system, meaning that code can be executed as soon as it is written.
  5. Python can be treated in a procedural way, an object-oriented way or a functional way.

- **Python syntax compared to other programming languages**

  Python was designed for readability and has some similarities to the English language with influence from mathematics. For example, Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses. Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly brackets for this purpose.

  Remember whitespaces means something in your code, before running your code, always remove redundant spaces.

During this tutorial, we will mention and use examples of the basic concepts in programming such as loops, functions, and classes. After that, we will introduce two key libraries in Python (not only for AI), Numpy and Matplotlib.

2 Setup

The most recent major version of Python is Python 3, we will use this version. Python 2, although is not up-to-date, is still quite popular. However, Python 2 might not be compatible with other libraries that we will use in the future.

2.1 Python

To do this tutorial and your future ones, you need to run Python. There are different alternatives, if you prefer to install Python on your laptop, we recommend Anaconda. However, if you have previous experience with Python and prefer something else, it should not be a problem. Anaconda is an open-source Python distribution that includes a full programming environment, therefore, you can code directly on your terminal or use platforms such as Spyder or Jupyter. Depending on the operative system you have, you should follow the installation directions:

- Windows
- MacOS
- Linux

Alternatively, Google Colab allows writing and executing Python code through the browser. What is Google Colab? See this link.

2.2 NumPy

NumPy is a powerful library for numerical computing in Python. It provides support for large, multi-dimensional arrays and a collection of functions to operate on these arrays efficiently.

To use NumPy on your laptop, first you need to install it (if it is not pre-installed). How to install NumPy? See this link. Both Anaconda and Google Colab have NumPy preinstalled, therefore you need simply import it in your code before use.

2.3 Matplotlib

Matplotlib is a widely-used data visualization library in Python. It allows us to create various types of plots, such as line plots, bar plots, and scatter plots. During our tutorials, you will see how this library helps us to show the results, and track the performance of our AI algorithms.

To use Matplotlib on your laptop, first you need to install it (if it is not pre-installed). How to install Matplotlib? See this link. Both Anaconda and Google Colab have Matplotlib preinstalled, therefore you need simply import it in your code before use.

3 Basics of Python

3.1 Loops

- **For loop:** Write a Python program that finds all the prime numbers between 1 and 1000 (inclusive) and stores them in a list.
- **While loop:** Write a Python program that counts the number of digits of a natural number n, i.e., n is a positive, integer number.

3.2 If - else statement

Write a Python program that takes a user's age as input and determines the ticket price for a theme park based on the following criteria:

- Children aged 3 and below enter for free.
- Children aged 4 to 10 pay $10.
- Adults aged 11 to 17 pay $15.
- Adults aged 18 to 59 pay $20.
- Seniors aged 60 and above pay $12.

The program should also consider whether the user has a membership. If a user has a membership, they receive a 20% discount on the ticket price. The program should display the appropriate ticket price based on the age and membership status entered by the user. Consider user only enters yes/no to declare their membership status.

3.3 Functions

Functions in Python enable us to break down our code into reusable blocks. They can take inputs (arguments) and return outputs (return values). To get more familiar with functions in Python, let's see the following example.

3.3.1 Example: Factorial

The factorial of a non-negative integer n is the product of all positive integers less than or equal to n. The factorial of 0 is defined as 1. Let's write a recursive function called `factorial` that takes an integer as input and returns its factorial.

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)

num = int(input("Enter a number: "))
result = factorial(num)
print("Factorial of", num, "is", result)
```

3.3.2 Exercise:

Write a Python program that checks if a given string is a palindrome or not. A palindrome is a word, phrase, number, or sequence of characters that reads the same forward and backward, ignoring spaces, punctuation, and letter casing. Your task is to write a function called `is_palindrome` that takes a string as input and returns True if the string is a palindrome, and False otherwise.

3.4 Classes and objects

Python supports object-oriented programming, allowing us to define classes and create objects based on them. Classes represent objects, and they consist of attributes (variables) and methods (functions).

3.4.1 Exercise:

Consider a simple banking system. Write a Python program that models a Bank Account class. The BankAccount class should have the following functionalities:

- Initialize the account with an account number and an initial balance.
- Allow deposits and withdrawals.
- Display the current balance.

What are other features that we can add to the bank account class?

3.4.2 Exercise:

Consider a library management system. Write a Python program that models the following entities: Library, Book, and Member.

The Library class should have the following functionalities:

- Keep track of a collection of Book objects.
- Allow members to borrow books if they are available.
- Keep a record of borrowed books and their due dates (members can borrow a book for 14 days).
- Allow members to return books.

The Book class should have the following attributes:

- Title.
- Author.
- Availability status.

The Member class should have the following attributes:

- Full name.
- Member ID.

Your task is to design and implement these classes to represent the library management system (**Hint:** you probably will need to import datetime package).

4 Numpy

The first step is calling the library in your code. To ease coding, we usually abbreviate Numpy into np.

```
import numpy as np
```

Now let's see a few simple examples of Numpy:

- Creating an array:

  ```
  arr = np.array([1, 2, 3, 4, 5, 6])
  print(arr)
  ```

- Accessing array elements:

```
    print(arr[0])  (Output: 1)
    print(arr[2])  (Output: 3)
```

- Performing arithmetic operations

```
    arr2 = arr + 2
    print(arr2)  (Output: [3 4 5 6 7 8])
```

- Reshaping an array:

```
    arr3 = np.reshape(arr2, (2, 3))
    print(arr3)
```

4.1 Exercise:

Write a Python program that generates a random matrix of size N×N, where N is a user-specified positive integer. The program should then find the sum of each row and each column of the matrix using the numpy library.

5 Matplotlib

The first step is calling the library in your code. To ease coding, we usually abbreviate Matplotlib into plt.

```
import matplotlib.pyplot as plt
```

Now let's see a few simple examples of Matplotlib:

- Line plot:

```
x = [1, 2, 3, 4, 5]
y = [2, 4, 6, 8, 10]
plt.plot(x, y)
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.title('Line Plot')
plt.show()
```

- Bar plot:

```
x = ['A', 'B', 'C', 'D', 'E']
y = [3, 7, 2, 9, 5]
plt.bar(x, y)
plt.xlabel('Categories')
plt.ylabel('Values')
plt.title('Bar Plot')
plt.show()
```

5.1 Exercise:

Consider a dataset containing the performance metrics (accuracy, precision, recall) of multiple machine learning models on a classification task given to you. Write a Python program that performs the following tasks:

- Calculate the mean and standard deviation of each metric across all models.
- Visualize the mean performance of each metric using a bar chart, with the x-axis representing the metrics and the y-axis representing the mean values.
- Plot error bars on the bar chart to represent the standard deviation of each metric.
- Compare the performance of each model on a specific metric using a line plot, with the x-axis representing the models and the y-axis representing the metric value.

Remember there is always a trade-off between simplicity and flexibility. It is simple to do a task by only few lines of the code, but it may lack of flexibility and may not be suitable for handling more complex scenarios or accommodating future changes. On the other hand, a more flexible and robust solution may require additional code and complexity. Some libraries help us to write a code too easy and it takes the opportunity from us to understand what is exactly happening behind the library and takes the flexibility of our code. This is your responsibility to consider all aspects of this trade-off before using any library.

References

[1] "Wikipedia, python (programming language)." `https://en.wikipedia.org/wiki/Python_(programming_language)`. Accessed: 2023-05-25.

[2] "W3school, python introduction." `https://www.w3schools.com/python/python_intro.asp`. Accessed: 2023-05-25.