Technische Hochschule Rosenheim

Modul - Introduction to AI (AI1)

Bachelor Programme AAI

04 - AI Scenarios

Prof. Dr. Marcel Tilly

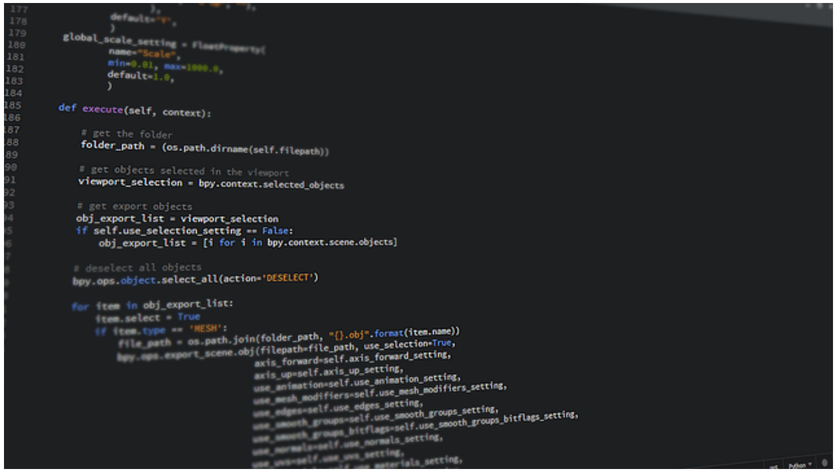
Faculty of Computer Science, Cloud Computing

Vers. 8.11.2021

Agenda

On the menu for today:

- AI Scenarios
- Python
  - functions
  - comprehensions



```
            default='Y',
            )
    global_scale_setting = FloatProperty(
            name="Scale",
            min=0.01, max=1000.0,
            default=1.0,
            )

    def execute(self, context):

        # get the folder
        folder_path = (os.path.dirname(self.filepath))

        # get objects selected in the viewport
        viewport_selection = bpy.context.selected_objects

        # get export objects
        obj_export_list = viewport_selection
        if self.use_selection_setting == False:
            obj_export_list = [i for i in bpy.context.scene.objects]

        # deselect all objects
        bpy.ops.object.select_all(action='DESELECT')

        for item in obj_export_list:
            item.select = True
            if item.type == 'MESH':
                file_path = os.path.join(folder_path, "{}.obj".format(item.name))
                bpy.ops.export_scene.obj(filepath=file_path, use_selection=True,
                                        axis_forward=self.axis_forward_setting,
                                        axis_up=self.axis_up_setting,
                                        use_animation=self.use_animation_setting,
                                        use_mesh_modifiers=self.use_mesh_modifiers_setting,
                                        use_edges=self.use_edges_setting,
                                        use_smooth_groups=self.use_smooth_groups_setting,
                                        use_smooth_groups_bitflags=self.use_smooth_groups_bitflags_setting,
                                        use_normals=self.use_normals_setting,
                                        use_uvs=self.use_uvs_setting,
```

Technische Hochschule Rosenheim

AI Scenarios

Task 1: AI Topics

- Break out in groups 3-4 students each
- Collect on https://zumpad.zum.de/
  - Create the group zumpad yourself (no registration required!)

Collect a list of AI related topics!

- At the end: Share the link to your group zumpad in the chat!

10 min.

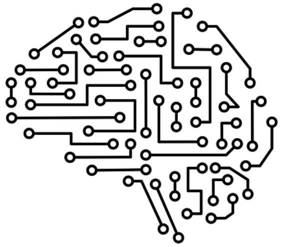
Task 2: AI Technologies

- Break out in groups 3-4 students each (new group!)
- Collect on https://zumpad.zum.de/
    - Create the group zumpad yourself (no registration required!)

Collect a list of AI technologies!

5 min.

- At the end: Share the link to your group zumpad in the chat!



Task 3: AI Domains

- Break out in groups 3-4 students each (new group!)
- Collect on https://zumpad.zum.de/
  - Create the group zumpad yourself (no registration required!)

Collect a list of AI domains and related problems!

10 min.

- At the end: Share the link to your group zumpad in the chat!

Homework

- Form teams of 3-4 students each until 16.11
- Pick a domain until 30.11
- Create a poster (DIN A0)
    - Motivate AI topics for this domain
    - List use cases which can be solved by KI
    - List technologies which are/ can be used
    - Find images at oen repos (e.g. *pixabay.com*)
- Deadline: 18.01.2022
- In presence paper presentation on 25.01.2022

Python (cont'd)

Functions

Technische Hochschule Rosenheim

- Functions are used to structure statements, which can then be conveniently executed any number of times by calling the function. The function can receive input arguments and return objects itself.
- Functions in Python are called with the keyword `def`, the function name and the passed parameter list as follows:

```
def function_name(a,b,...):
    <operations>

#  Define a function to sum numbers
def sum(a, b):
    return a+b

result = sum (1,2)
print(result)
```

https://repl.it/@marceltilly/TH-Rosenheim#lecture/10-functions.py

Functions

Technische Hochschule Rosenheim

Functions can be used as parameters:

```
#  product function
def prod(a, b):
 return a*b

 #  sum function
def sum(a, b):
 return a+b

#  Function:  using op on a and b
def execute(a, b, op):
    return op(a, b)

#  Functions can be used like parameters
print(execute(3, 5, prod))
print(execute(3, 5, sum))
```

https://repl.it/@marceltilly/TH-Rosenheim#lecture/11-lambda.py

Exercise 3

Write short Python programms that …

… writes all content of a given file into a new file by skipping every 5th line

List Comprehension

- List comprehensions provide a concise way to create lists
- The list comprehension always returns a result list
- The result will be a new list resulting from evaluating the expression in the context of the for and if clauses which follow it

The basic syntax is

```
list_variable = [expression for item in iterable if condition]
```

```
S = [x**2 for x in range(10)]
V = [2**i for i in range(13)]
M = [x for x in S if x % 2 == 0]
```

https://repl.it/@marceltilly/TH-Rosenheim#lecture/12-comprehensions.py

Exercise 3

Write short Python programms that ...

Let’s say I give you a list saved in a variable: a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]. Write one line of Python that takes this list a and makes a new list that has only the even elements of this list in it.

Hint: Use comprehensions!

> Note: This is a one liner!

Python Program

Technische Hochschule Rosenheim

- A **Python** program is a script that is executed from start to finish.
- Functions **def** are prepended and interpreted and executed as needed.
- The main body may appear as a sequence of commands at the end of the file.
  - The usual way is to mark it with `if __name__ == '_main__':`

```
<import>

<global variables>

<functions>
def <name>:
    <statements>

if __name__ == '_main__':
    <statement>
```

Import

External modules can be added via `import`.

```python
# import the modules
import sys
import random
import uuid

# import the module and assign alias
import numpy as np

# import of a module, subpackage or object from a module
# with alias assignment
from matplotlib import plot as plt
```

Library: Numpy

https://numpy.org/

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- useful linear algebra, Fourier transform, and random number capabilities

```
>>> import numpy as np
>>> a = np.arange(15).reshape(3, 5)
>>> a
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14]])
```

Numpy Example

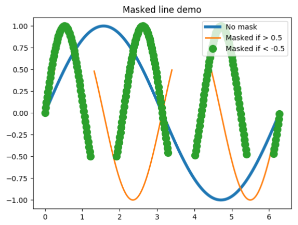
```python
import numpy as np
import time
size_of_vec = 1000

def pure_python_version():
    t1 = time.time()
    X = range(size_of_vec)
    Y = range(size_of_vec)
    Z = [X[i] + Y[i] for i in range(len(X))]
    return time.time() - t1


def numpy_version():
    t1 = time.time()
    X = np.arange(size_of_vec)
    Y = np.arange(size_of_vec)
    Z = X + Y
    return time.time() - t1

t1 = pure_python_version()
t2 = numpy_version()
print(t1, t2)
print("NumPy is in this example " + str(t1/t2) + " faster!")
```

code on https://repl.it/@marceltilly/TH-Rosenheim#lecture/13-numpy.py

Graph output: Matplotlib

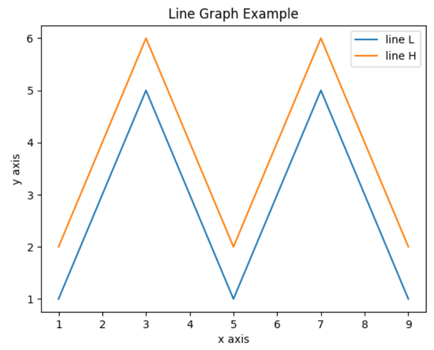
https://matplotlib.org/

- Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.
- Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits.



Matplotlib Example

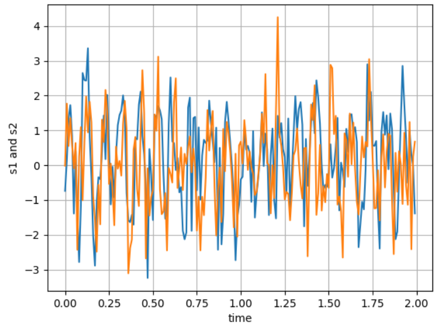
```
import matplotlib.pyplot as plt

x  = [1, 2, 3, 4, 5, 6, 7, 8, 9]
y1 = [1, 3, 5, 3, 1, 3, 5, 3, 1]
y2 = [2, 4, 6, 4, 2, 4, 6, 4, 2]
plt.plot(x, y1, label="line L")
plt.plot(x, y2, label="line H")
plt.plot()

plt.xlabel("x axis")
plt.ylabel("y axis")
plt.title("Line Graph Example")
plt.legend()
plt.show()
plt.savefig("simple.png")
```



code on https://repl.it/@marceltilly/TH-Rosenheim#lecture/14-matplotlib.py

Matplotlib Example

```python
import numpy as np
import matplotlib.pyplot as plt

dt = 0.01
t = np.arange(0, 2, dt)
# white noise 1
nse1 = np.random.randn(len(t))
# white noise 2
nse2 = np.random.randn(len(t))

# Two signals at 10Hz and a random part
s1 = np.sin(2 * np.pi * 10 * t) + nse1
s2 = np.sin(2 * np.pi * 10 * t) + nse2

fig, axs = plt.subplots()
axs.plot(t, s1, t, s2)
axs.set_xlabel('time')
axs.set_ylabel('s1 and s2')
axs.grid(True)

plt.show()
plt.savefig("test.png")
```



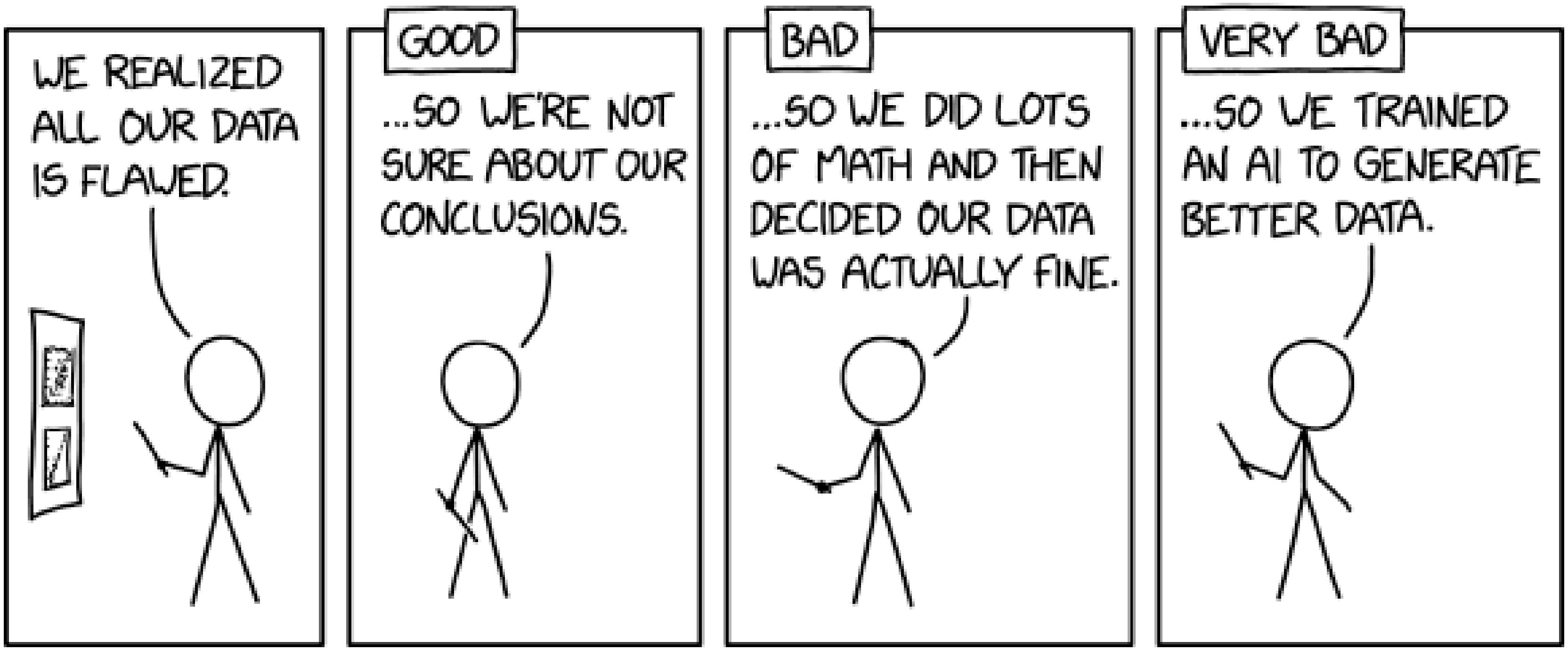
code on https://repl.it/@marceltilly/TH-Rosenheim#lecture/15-mat.py

Summary

Lessons learned today:

- Python Basics
    - comprehensions
    - numpy and matplotlib
- more to come...

Final remark


WE REALIZED ALL OUR DATA IS FLAWED.
GOOD
...SO WE'RE NOT SURE ABOUT OUR CONCLUSIONS.
BAD
...SO WE DID LOTS OF MATH AND THEN DECIDED OUR DATA WAS ACTUALLY FINE.
VERY BAD
...SO WE TRAINED AN AI TO GENERATE BETTER DATA.

taken from https://xkcd.com/2494/