Hillsboro Python Machine Learning Meetup

Mar/2017

Ernest Bonat, Ph.D.
Senior Software Engineer
Senior Data Scientist

PC Wi-Fi:
un:ps: PSUSUMMER2014):

- 6:00 – 6:40 pm: Pizza, **water only** and networking.

- 6:40 – 6:45 pm: Welcome message by Ernest Bonat, Ph.D.

- 6:45 – 8:00 pm: Presentation and open discussions.

- 8.00 pm – 9.00 pm: Coding and learning session. Bring your Python development laptop!

Why did I create this meetup?

1. Bad traffic to Portland downtown.
2. Hard to find a parking.
3. Bad Python presentation code.
4. No time at all to review the presentation and learn something after the meeting.

We need your support:

1. Need 2 Senior Python Developers for presentation and code review every month (Co-organizers, 4-6 hours a month).

2. Every meeting cost about $200. We need companies to sponsor our meetings.

3. Email Ernest at ebonat@15itresources.com

Our Meetup Mission:

1. *"Come, Listen, Code and Learn".*

2. Finding and presenting best practices of Machine Learning using Python Data Stack.

3. Create great networking place for Hillsboro-Beaverton Data Scientists.

Today Presentation

“High Performance Data Analytics Using Python Asynchronous Programming”

Training

"Business Statistics Course for Python Programmers"

(http://15itresources.com/training/)

What do we do different?

1. Corporate/in-person/hands-on training.
2. Direct business data analysis for your company needs.
3. Own and use the Ernest's Python Data Science libraries which offer full proof programming recipes.

Multithreading vs Asynchronous Programming

- Multithreading – run on many threads
- Asynchronous – run on a single thread

Python coroutines are all run on a single thread (application main thread), and don’t require extra sockets or memory, it would be a lot harder to run out of resources.

Task Planning

Sync	Task 1	Task 2	Task 3	Task 4	Task 5
Parallel	Task 1				
	Task 2				
	Task 3				
	Task 4				
	Task 5				
Async	Task 1	Task 2	Task 3		
				Task 4	
	Task 5				

Table 1. Sync vs. Parallel vs. Async

asyncio/await Python Code

```
# in python 3.4 (it works in 3.5)
@asyncio.coroutine
def py34_coroutine():
    yield from do_stuff()
```

```
# in python 3.5
async def py35_coroutine():
    await do_stuff()
```

Main running asyncio APIs

```
# create even loop object
ioloop = asyncio.get_event_loop()

# set list of task to run
tasks = [ioloop.create_task(Task1), ioloop.create_task(Task2)),

# create the wait talk object
wait_tasks = asyncio.wait(tasks)

# run all the talks until all complete
ioloop.run_until_complete(wait_tasks)


# release from memory the event loop object
ioloop.close()
```

Online Papers

- async/await in Python 3.5 and why it is awesome https://www.youtube.com/watch?v=m28fiN9y_r8&t=178s

- How the heck does async/await work in Python 3.5? https://snarky.ca/how-the-heck-does-async-await-work-in-python-3-5/

Numba (http://numba.pydata.org/)

The Numba project is supported by Continuum Analytics – creator of Anaconda Python Package!

Python code can be just-in-time compiled to native machine instructions, similar in performance to C, C++ and FORTRAN, without having to switch languages or Python interpreters.

Install Numba in your laptops: ‘conda install numba’ and run the first example

Coding Session

Async	Task 1	Task 2	Task 5		
			Task 4		
	Task 3	Task 7			
	Task 6	Task 8			

White an asyncio Python program based on the table above for Taks1….Task6.

Presentation Source Code

(https://github.com/ebonat/hillsboro_machine_learning_03_2017)