Assignment for Class Computer Science for Big Data E1

Spark Example pi.py, Points: 6

Apache Spark is an open-source distributed cluster-computing framework that supports programming languages: Java, Scala, Python and R. *PySpark* is the Python language binding for Spark.

Spark exists in a variety of configurations from a single-system (laptop) configuration for developing and testing Spark code, to large cluster or cloud deployments.

A Unix-based OS (Linux, MacOS) is recommended for installation: https://spark.apache.org/downloads.html, (~220MB). If you have Docker, various Docker containers exist for *pySpark*, which simplify the setup, search for containers at: https://hub.docker.com/search?q=pyspark&type=image.

A setup for Windows is described here: https://phoenixnap.com/kb/install-spark-on-windows-10. *Cygwin* is not sufficient to install and run Spark. Other Alternatives for Windows are:

- *Windows Subsystem for Linux* (WSL), which is a Virtual Machine (VM) integrated in the kernel of Windows 10.
- Setting up a Linux Virtual Machine, e.g. with *Virtual Box* (https://www.virtualbox.org).
- Use of a *vagrant* "Spark-box" such as https://app.vagrantup.com/paulovn/boxes/spark-base64, which is a Virtual Box VM with Python Spark preconfigured (vagrant is a VM-management software that connects to VM Software such as Virtual Box and offers easier VM deployment).

Spark requires **Java 1.8** and *pySpark* requires **Python 3**. Install the Java 1.8 JRE run-time, if not present, and make sure that $JAVA_HOME and $PATH point to it for Spark.

1.) Install *pySpark* for your system and briefly document the major steps by few bullets describing the packages you have chosen and major installation steps, also for yourself to learn and remember. [1 Pts]

2.) Test your installation by running simple Python code and the *pySpark* code example *pi.py* that comes with the default distribution in: spark/examples/src/main/python. You also find it in Moodle. If *pySpark* is installed properly, the following executions should work:

a. **pyspark** – opens an interactive shell to execute pySpark/Python commands. [a-c: 3 Pts]

```
>>> list = [1,22,2,3,4]
>>> print(sorted(list))
[1, 2, 3, 4, 22]
```

b. **pyspark < pi.py** – executes the pySpark program pi.py **--> Pi is roughly 3.140360** .

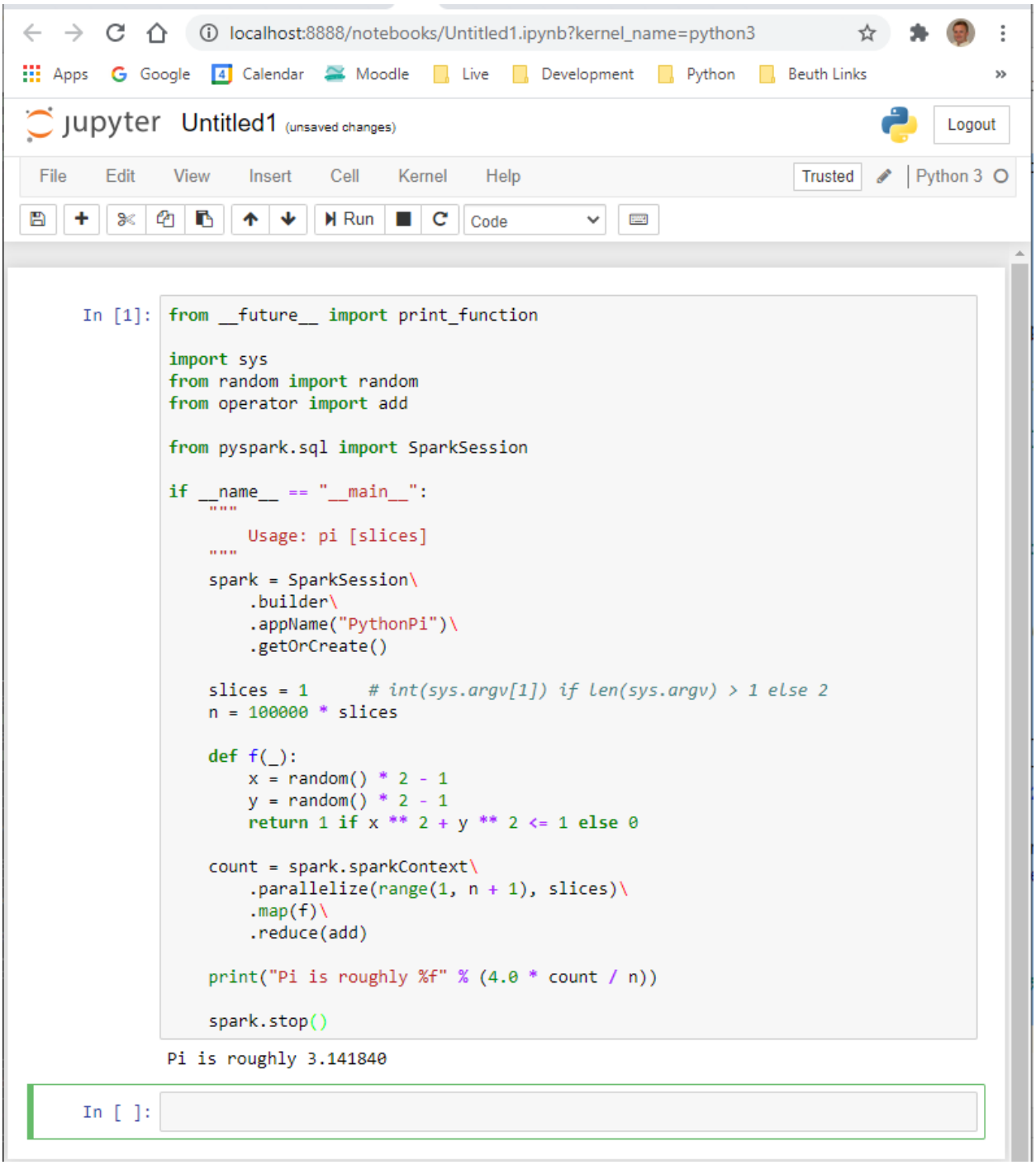
c. **spark-submit --master local[4] pi.py 2>session.log | tee result**
launches the *pySpark* master process with up to 4 "local" worker processes. The master process receives the submission and assigns the program for execution to one worker process.

d. When *Jupyter* is installed, *pySpark* can run a local Jupyter server by setting environment variables for executing notebooks:

PYSPARK_DRIVER_PYTHON="jupyter" PYSPARK_DRIVER_PYTHON_OPTS="notebook" pyspark

3.) Create a new notebook (upper right -> New -> Python3 kernel -> opens a new notebook). [d: 1 Pts]

Copy the code of *pi.py* into the notebook and execute. The notebook should run without error and produce the following result:



4.) Reengineer the program *pi.py* and briefly describe what it does and how it works. What happens when the value of variable 'slices' increases from 1 to 2 and 4? [1 Pts]