



Anh-Tuan Nguyen
Phenikaa School of Computing

OUTLINE

- Computer/ computation
- Python basic
- Mathematical operation
- python variable and type

COMPUTER?



- Observe nature
- Imagine
- Learn
- Calculation?
- Remember important features?
- Interact with environment and action independently



- Perform calculator
- Remember (storage)
- Do what you tell it to do -> Programming -> Instruction

KNOWLEDGE?

Declarative knowledge is statements of fact.

- someone will win a Google Cardboard before class ends

Imperative knowledge is a recipe or “how-to”

- Open terminal
- Run “ipython” command
- Import numpy
- Create random numbers
- Do some calculations

EXAMPLE

Square root of a number x is y such that $y^*y = x$ (Declarative knowledge)

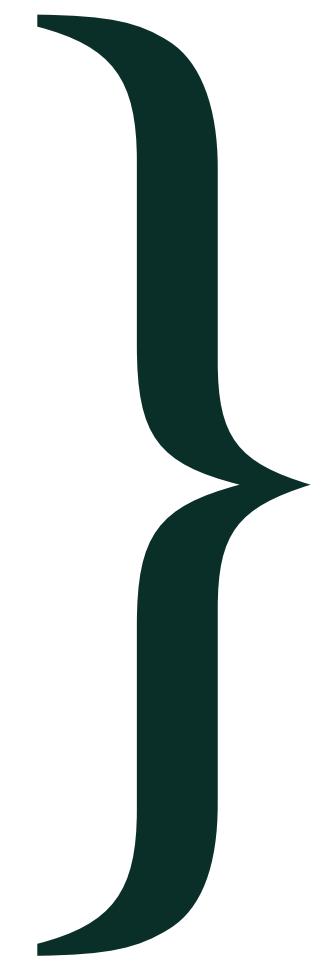
Recipe for deducing square root of a number x (16) (Imperative knowledge)

- Start with a **guess**, g
- If g^*g is **close enough** to x , stop and say g is the answer
- Otherwise make a **new guess** by averaging g and x/g
- Using the new guess, **repeat** process until close enough

g	g^*g	x/g	$(g+x/g)/2$
3	9	16/3	4.17
4.17	17.36	3.837	4.0035
4.0035	16.0277	3.997	4.000002

RECIPE

- Sequence of simple **steps**
- **Flow of control** process that specifies when each step is executed
- A means of determining **when to stop**

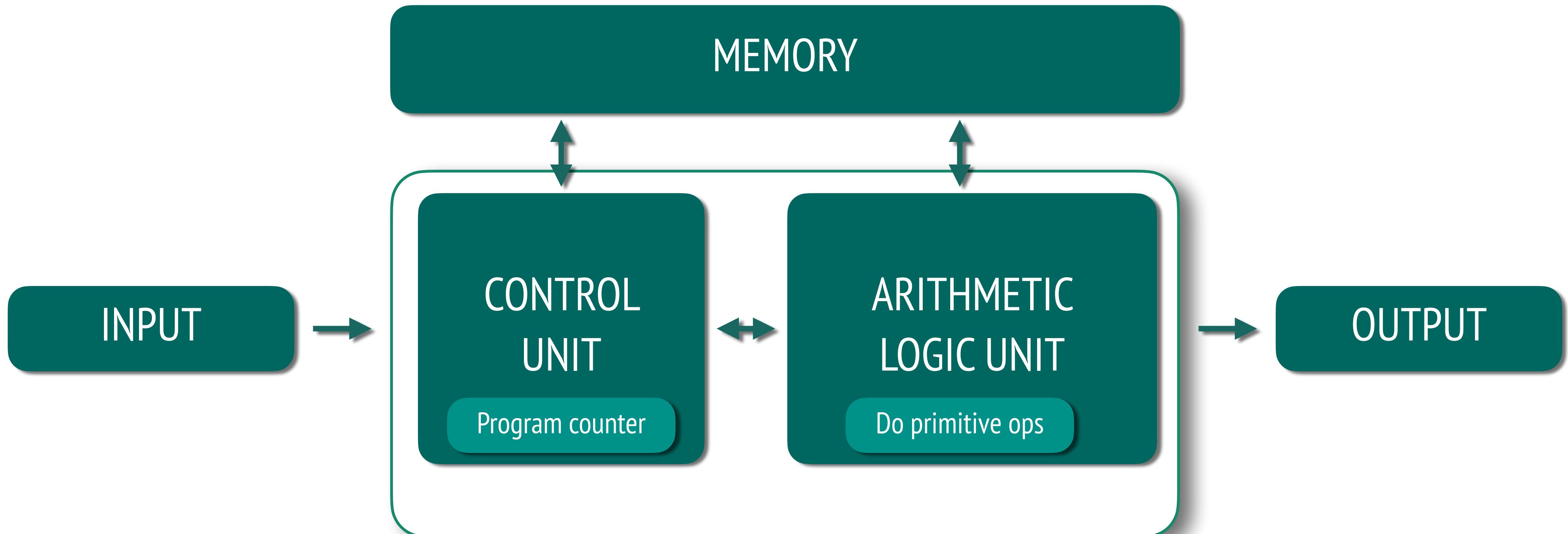


Algorithm

COMPUTER ARE MACHINES

- How to capture a recipe in a mechanical process
- Fixed program computer: calculators
- Stored program computer: Machine stores and executes instructions

BASIC COMPUTER ARCHITECTURE



STORED PROGRAM COMPUTE

- Sequence of **instructions stored** inside computer
 - built from predefined set of primitive instructions
 - 1) arithmetic and logic
 - 2) simple tests
 - 3) moving data
- Special program (interpreter) **executes each instruction in order**
 - use tests to change flow of control through sequence
 - stop when done

BASIC PRIMITIVES

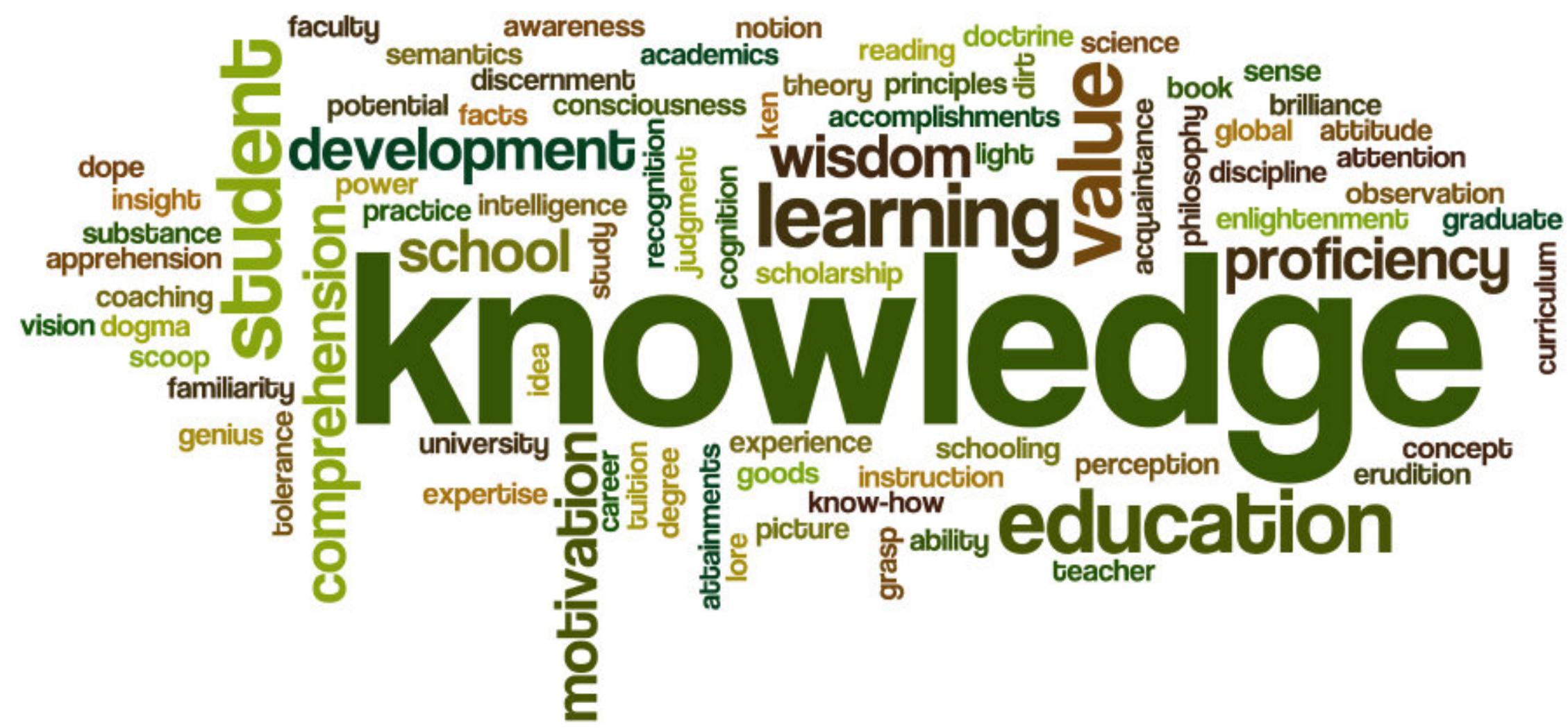
- Turing showed that you can **compute anything** using 6 primitives
- Modern programming languages have more convenient set of primitives
- Can abstract methods to **create new primitives**
- Anything computable in one language is computable in any other programming language

CREATING RECIPES/ALGORITHMS

- A programming language provides a set of primitive **operations**
- **Expressions** are complex but legal combinations of primitives in a programming language
- Expressions and computations have **values** and meanings in a programming language

ASPECTS OF LANGUAGES

- Primitive constructs
 - English: words
 - Programming language: numbers, strings, operators



```
float **
* < > bool
* <= <= string >= !=
int /
NoneType -
= = +
```

SYNTAX

- **Syntax**

- English:
 - ➡ “cat dog boy” -> not syntactically valid
 - ➡ “cat hugs boy” -> syntactically valid
- Programming language: numbers, strings, operators
 - ➡ “hi“5 -> not syntactically valid
 - ➡ 3.2*5. -> syntactically valid

STATIC SEMANTICS

- Static semantics is which syntactically valid strings have meaning
 - English:
 - ➡ “I are hungry“ -> syntactically valid but static semantic error
 - Programming language: numbers, strings, operators
 - ➡ $3.2 * 5$ -> syntactically valid
 - ➡ $3 + "hi"$ -> static semantic error

TYPE OF ERRORS

- Syntactic errors
 - common and easily caught
- Static semantics errors
 - some languages check for these before running program can cause unpredictable behavior
- No semantic error but different meaning than what programmer intended
 - program crashes, stops running
 - program runs forever
 - program gives an answer but different than expected

PYTHON PROGRAM

- A program is a sequence of definitions and commands
 - definitions evaluated
 - commands executed by Python interpreter in a shell
- Commands (statements) instruct interpreter to do something
- Can be typed directly in a shell or stored in a file that is read into the shell and evaluated

DATA OBJECTS

- Programs manipulate **data objects**
- objects have a **type** that defines the kinds of things programs can do to them
- objects are
 - ➔ scalar (can not be subdivided)
 - ➔ non-scalar (have internal structure that can be addressed)

SCALAR OBJECTS

- int - represent integers
- float - real numbers
- bool - Boolean values include True and False
- NoneType - special and has one value, None
- can use type () to see the type of an object

```
>>> type(5)
int
>>> type(3.0)
float
```

EXPRESSIONS

- Combine objects and operators to form expressions
- an expression has a **value**, which has a type
- syntax for a simple expression

<object> <operator> <object>

VARIABLE AND VALUES

- equal sign is an assignment of a value to a variable name

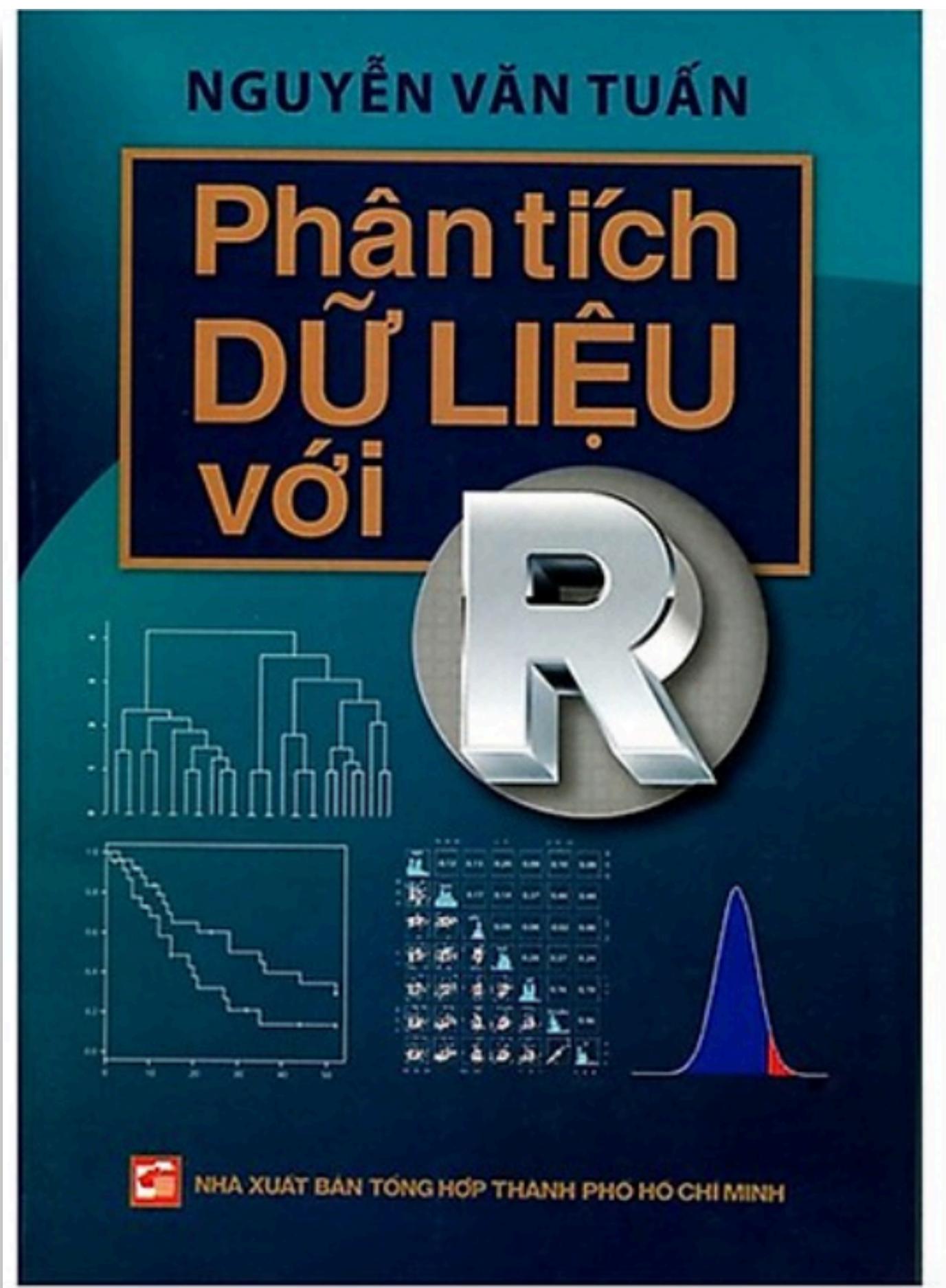
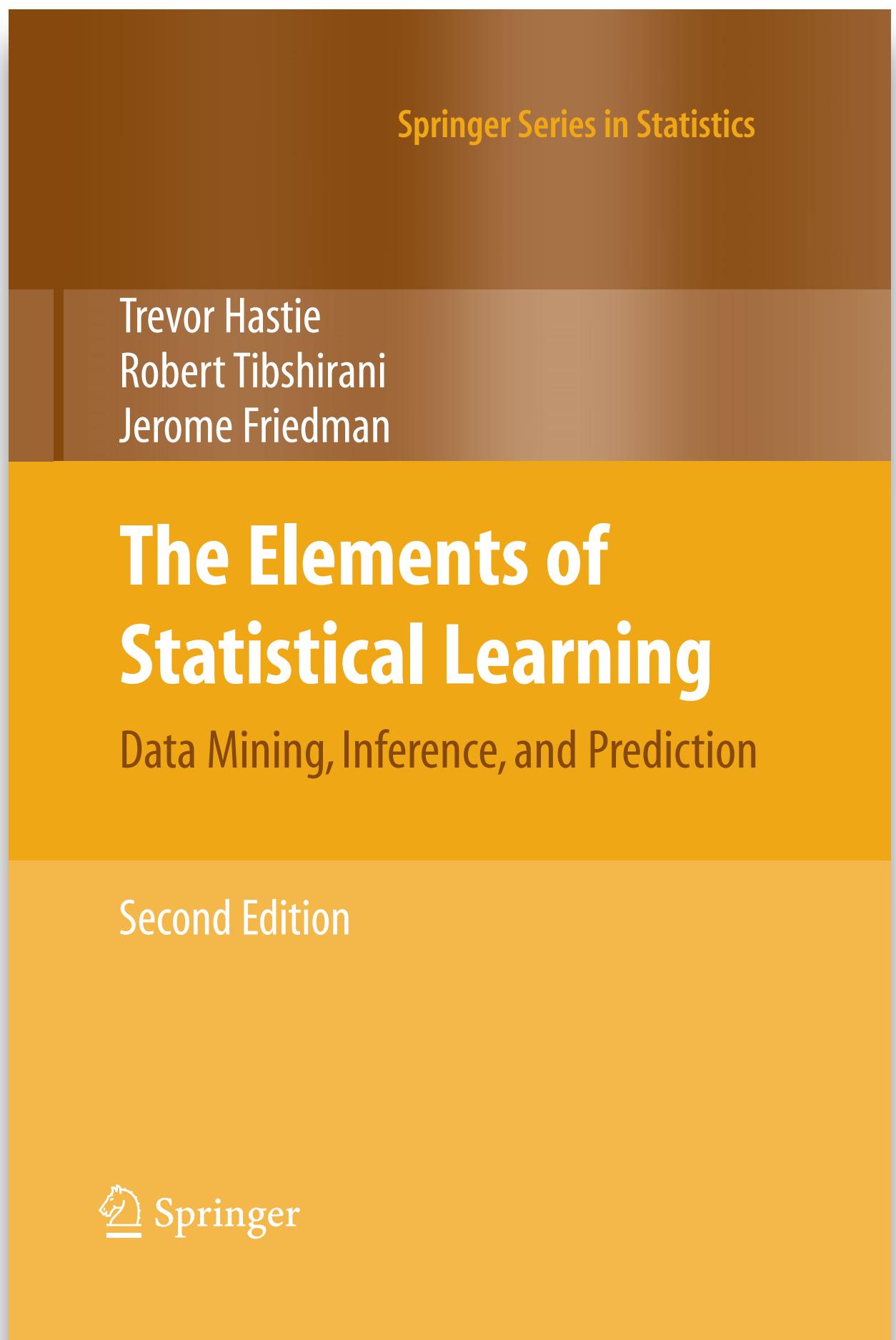
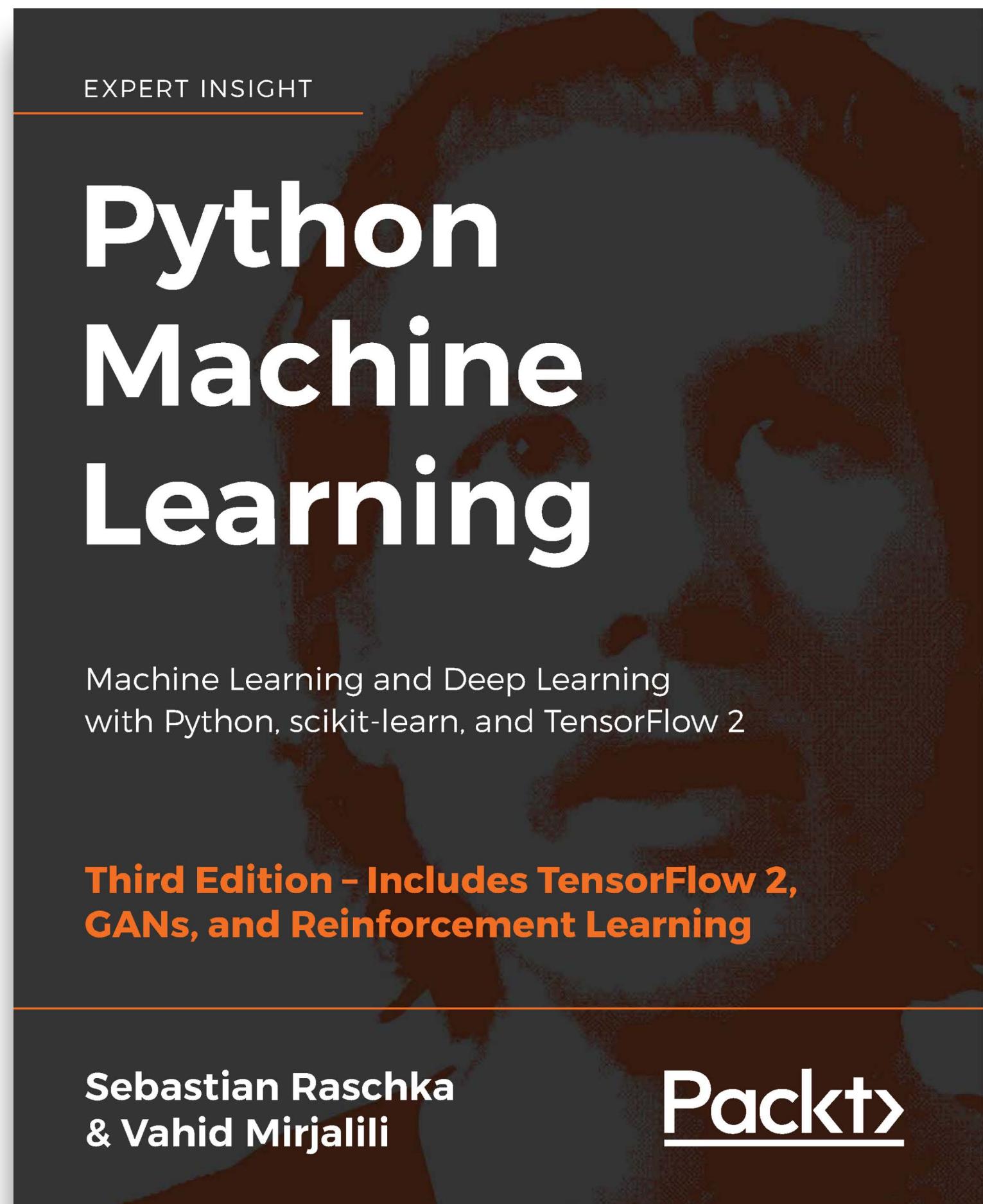
Variable

→ pi = 3.14159 ← Value

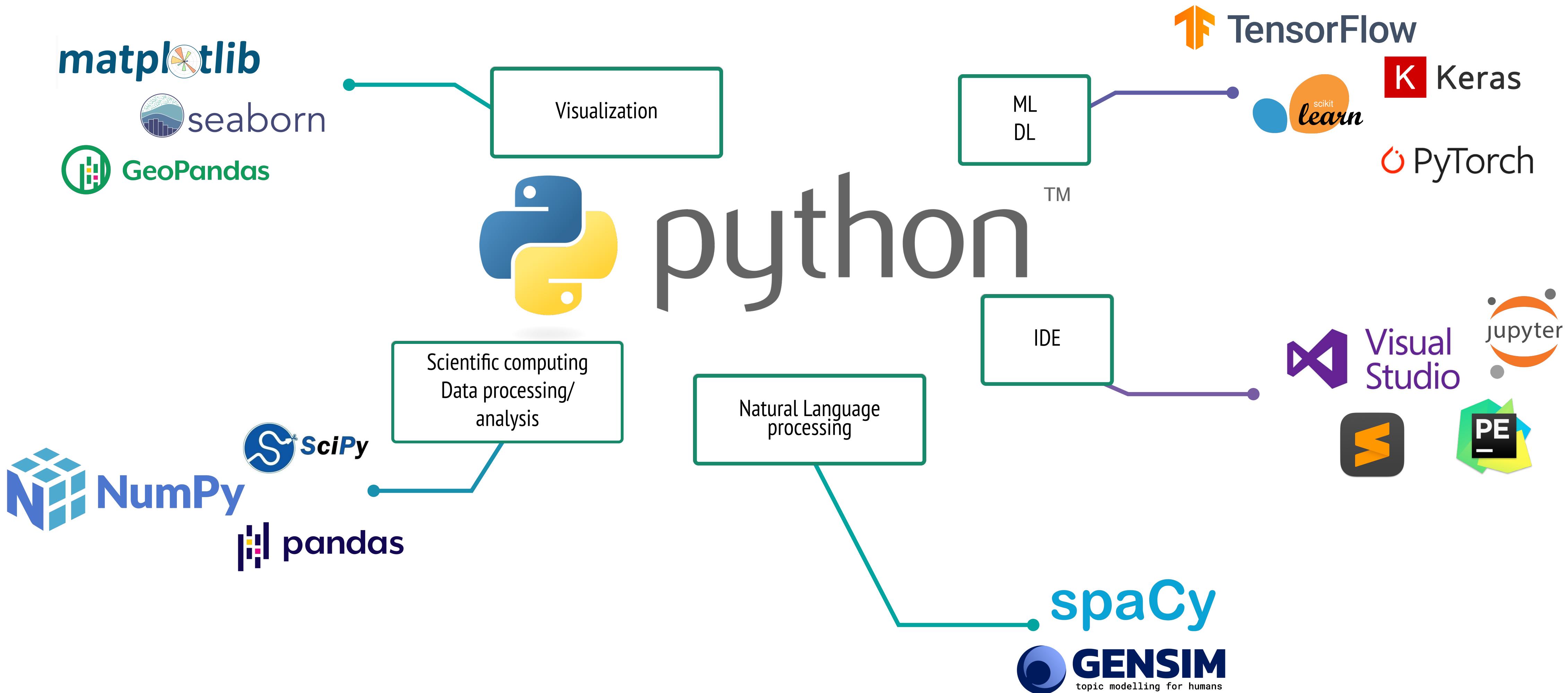
pi_approx = 22/7

- value stored in computer memory
- an assignment binds name to value
- retrieve value associated with name of variable by invoking the name, by typing pi

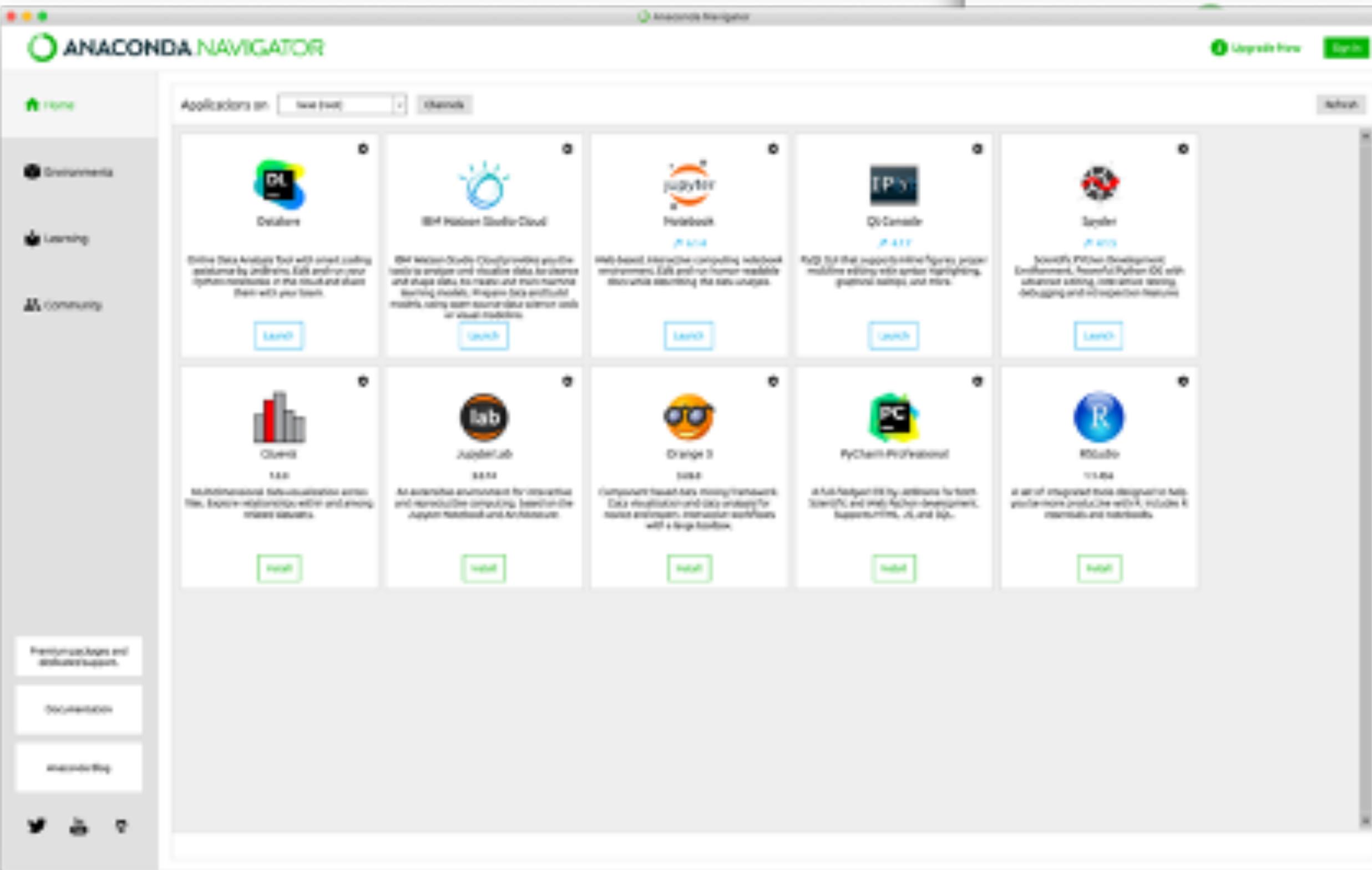
ATTACHMENTS



PROGRAMMING LANGUAGE/ PACKAGE REQUIREMENT

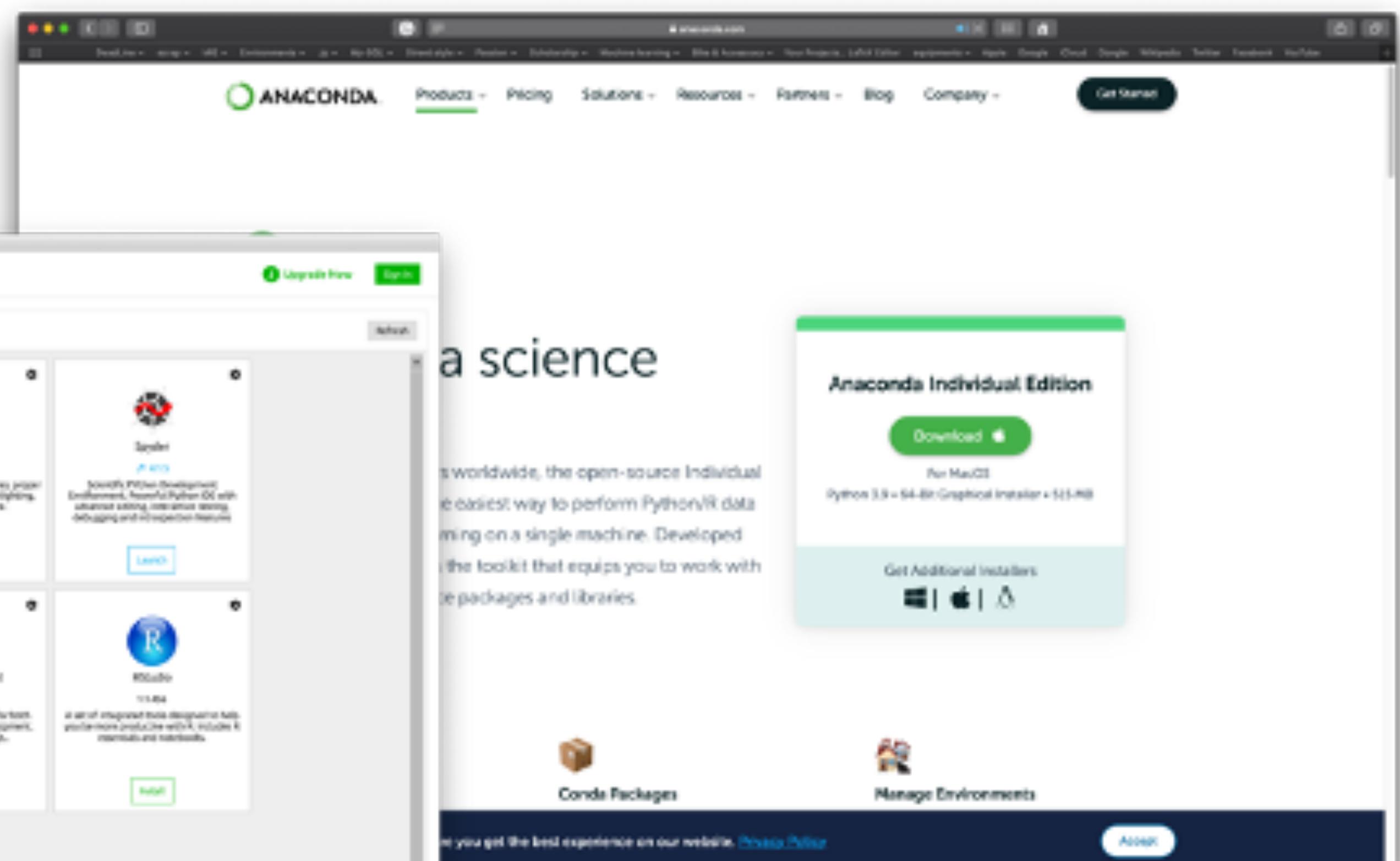


LEARNING BY CODING



The Anaconda Navigator interface displays a grid of data analysis tools:

- DataLab**: Multi-Tab Data Analysis Tool with smart coding completion by project. Edit environment specifications in the interface and share them with your team.
- IBH Holden Studio Cloud**: Multi-Tab Studio. Cloud-based platform to analyze and visualize data for science and shape ideas in this intuitive place. Share them with your team.
- Jupyter Notebook**: Multi-tab interface combining individual environments. Edit and run code in multiple notebooks simultaneously sharing the same workspace.
- QIConsole**: Python tool that integrates figures, poster, machine learning with semantic highlighting, graphical outputs, and more.
- Spyder**: Scientific Python Development Environment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features.
- Orange 3**: Component based data mining framework. Data visualization and data mining for machine learning, data mining and data analysis with a large toolbox.
- PyCharm Professional**: An IDE designed for Python developers. It includes support for scientific computing, data analysis, machine learning, and big data.
- RStudio**: A set of integrated tools designed to help you produce reproducible results with R, including R, R Markdown, and RStudio.



a science

Anaconda Individual Edition

Download   

Python 3.9 x 64-bit Graphical Installer - 513 MB

Get Additional Installers   

Conda Packages Manage Environments

Get the best experience on our website. [Privacy Policy](#) [Accept](#)