204789: Machine Learning and Neural Network


julia

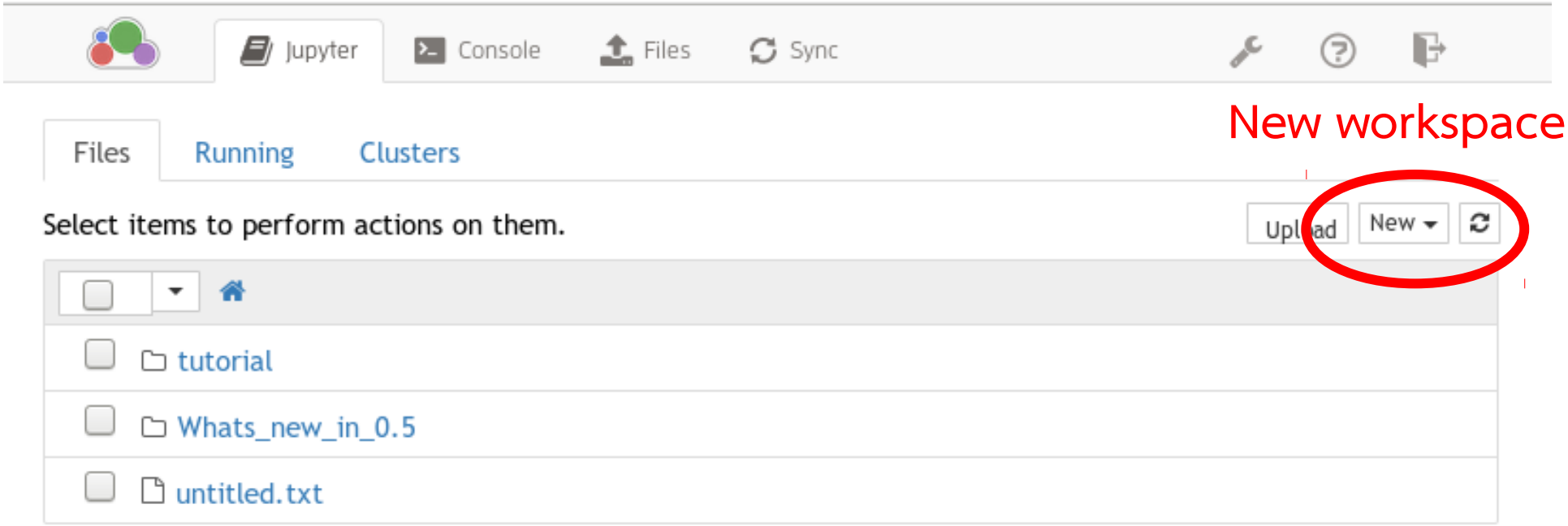
What is Julia ?

- a new programming language for scientific computing
- developed by a group mostly from MIT
- fully open source,
- convenient syntax for building math constructs like vectors, matrices, etc.
- super fast

Setting up Julia

- Two modes of execution
- JuliaBox.com: code in Julia in the cloud
  - automatically works
- Run Julia locally
  - faster and more flexible
  - Download binary from www.julialang.org

JuliaBox


Jupyter
Console
Files
Sync
New workspace
Files
Running
Clusters
Select items to perform actions on them.
Upload
New
tutorial
Whats_new_in_0.5
untitled.txt

General usage

- สามารถกด Tab เพื่อดูคำแนะนำได้
- สามารถใช้ลูกศรขึ้นลง เพื่อเรียกดูคำสั่งก่อนหน้า
- พิมพ์ ? เพื่อเปลี่ยนเข้าโหมดช่วยเหลือ
- ลง Package เพิ่มโดยใช้คำสั่ง
  - Pkg.add(“package_name”)
- เรียกใช้ package โดยใช้คำสั่ง
  - Using package_name

Basic operations

- +, -, *, /
- ยกกำลังด้วยเครื่องหมาย ^
- ฟังก์ชันทางคณิตศาสตร์อื่นๆ
  - exp(), sin(), cos(), tan(), log()

- ลองทำดู

```
result = exp(-2.33) * cos(22 * 180 / pi)
```

Boolean expression

- Evaluate to true or false
- Use ==, <=, >=, <, >, !=
- กลับ True เป็น false โดยใช้เครื่องหมาย !

  !(value == 4)

- รวม expression โดยใช้ && หรือ ||

if / else

```
if (value < 5)
    value = 10
elseif (value == 10)
    value = 15
else
    value = 20
end
```

Ranges

- สามารถสร้างลำดับของตัวเลขได้โดย

  1:10

- ตัวอย่างข้างต้นสร้าง เลข 1 ถึง 10 โดยเพิ่มทีละ 1
- สามารถกำหนดการเพิ่มของเลขได้โดย

  1:0.1:10

- สามารถดูลำดับได้โดยฟังก์ชัน collect(1:10)

Arrays, Vector, Matrices

- สร้าง array โดยใช้ []

  A = [1,2,3,4]

- ดูมิติของ array โดยใช้ฟังก์ชัน size()
  - Vector คือชื่อเรียก 1-dim array
  - Matrix คือชื่อเรียก 2-dim array
- แปลงรูปร่าง array โดยฟังก์ชัน reshape()

  reshape(a,2,2)

Useful array creating functions

- ones(n,m), zeros(n,m) สร้างเมทริกซ์ขนาด n x m ที่มีสมาชิกเป็น 1 หรือ 0 ทั้งหมด
- eye(m) สร้าง identity matrix
- fill(k, n, m) สร้างเมทริกซ์ขนาด n x m ที่มีสมาชิกทุกตัวมีค่าเท่ากับ k

Array indexing

- a = randn(2, 2)
- a[1, 1] # สมาชิกแถวที่ 1 คอลัมน์ 1
- a[1, :] # ทุกตัวของแถวแรก
- a[:, 1] # ทุกตัวของคอลัมน์แรก

Array methods

- a = [-1, 0, 1]
- length(a)
- sum(a)
- mean(a)
- std(a)
- var(a)
- maximum(a) , minimum(a)
- sort(a)

Element-wise operation

- ใช้ . นำหน้า operator ทางคณิตศาสตร์

- ones(2, 2) * ones(2, 2) # Matrix multiplication

- ones(2, 2) .* ones(2, 2) # Element by element multiplication

List (1-dim array)

- สร้างโดยใช้เครื่องหมาย []
  - my_list = [“dog”, 1, -3.14]
- เข้าถึงสมาชิกลำดับที่ i ของลิสต์โดยใช้ [i]
  - my_list[1]
  - my_list[0] : error, list start at 1
- สมาชิกตัวสุดท้ายของ list
  - my_list[end]

Dictionary

- Dictionary คล้ายกับ List แต่ว่าเก็บข้อมูลเป็นลักษณะคู่อันดับ key, value
- สร้างได้โดยเรียก Constructor Dict()

  myCollection = Dict()

  myCollection = Dict(“A”=>1, “B” => 2, “C” => “three”)

- Key must be unique
- Looking for elements

  myCollection[key] for example myCollection[“A”]

- Useful function: haskey(dict, key),

Dictionary

- Modifying value

  myCollection[“A”] = 55

- Deleting key

  delete!(myCollection, “A”)

- Useful function:
  - haskey(dict, key)
  - Getting all keys => collect(keys(dict))
  - Getting all value => collect(values(dict))

Ref: https://en.wikibooks.org/wiki/Introducing_Julia/Dictionaries_and_sets

For loop

```
value = 0

for I in 1:10

    value += I

end


for r in [1,2,3,4]

    value += r

end
```

Functions

- A chuck of code that can be run over and over
- Useful functions
  - print(“put this on the screen”)
  - println(“with new line”)
  - help(“function_name”)
  - quit()
  - rand() #generates random number between 0 and 1
  - sort()

Exercise

- ให้เขียนโปรแกรมเพื่อเปรียบเทียบความเร็วของ sorting algorithms สองตัว คือ
  - Merge sort: sort(x, alg=MergeSort)
  - Insertion sort: sort(x, alg=InsertionSort)
- วัดผลโดยการจับเวลาเรียกฟังก์ชัน

  tic()

  do_something

  toc()

File I/O

- เขียนลงไฟล์ โดยใช้ delimiter (default=whitespace)
  - writedlm(‘yourfilename’, variable, [delimiter])
- อ่านไฟล์เข้ามา
  - readdlm(‘yourfilename’,[delimiter])
- สำหรับการอ่านและเขียนไฟล์ csv (comma separated value) มีฟังก์ชันพิเศษ
  - writecsv() และ readcsv()

MATLAB File I/O

- Install the “MAT” package
  - Pkg.add(“MAT”)
- Reading from .mat file as a Dict
  - Vars = matread(“filename.mat”)
- Accessing value
  - Vars[“dict_key”]
- Writing to .mat file
  - matwrite(“filename.mat”, {myvar=>value....})

Neural network package

- ใช้แพคเกจที่ชื่อว่า BackpropNeuralNet
  - Pkg.add("BackpropNeuralNet")
- วิธีการใช้งาน

```
using BackpropNeuralNet

net = init_network([2, 3, 2]) # Initialise input,hidden,output

train(net, [0.15, 0.7], [0.1, 0.9])  # net, input, output

net_eval(net, [0.15, 0.7])
```

Hypothesis Testing

- แพคเกจสำหรับทดสอบทางสถิติชื่อ HypothesisTests
  - https://github.com/JuliaStats/HypothesisTests.jl
- การเพิ่มแพคเกจ
  - Pkg.add("HypothesisTests")
- การเรียกใช้แพคเกจ
  - using HypothesisTests
- การตรวจสอบโดยใช้ Wilcoxon Rank sum test
  - pvalue(SignedRankTest(x, y))

Plotting some results

- เรียกใช้แพคเกจชื่อว่า Winston
- วาดกราฟโดยสร้าง plot ขึ้นมาก่อน

  p = FramedPlot()

  p = FramedPlot(title=”my plot”, xlabel=”x axis”, ylabel=”y axis”)

  p ถือเป็น plot object ที่เราสามารถใส่ (add) วัตถุเข้าไปได้
- วัตถุที่ได้ใช้บ่อยคือ Points() ซึ่งมีวิธีการสร้างคือ

  Points(arrayOfXCoor, arrayOfYCoor, kind, color)

  EX: Points(x, y, kind=”plus”, color=”red”)

Plotting some results (cont.)

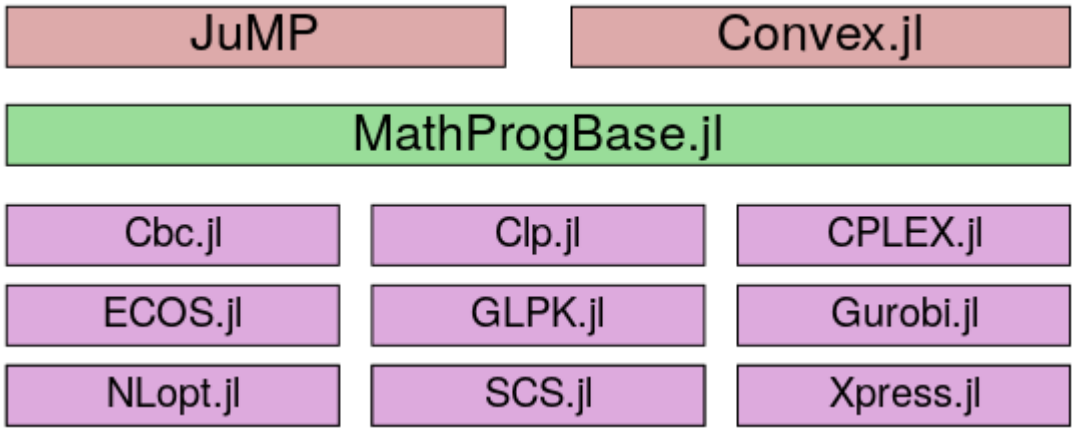
- จากนั้นเราสามารถ add วัตถุลง FramedPlot ได้

  add(p, Points(x, y, kind=”plus”, color=”red”))

- Kind มีให้เลือกดังนี้
  - Solid, dotted, dotdashed, plus, circle, asterisk, dot, cross, square, diamond, triangle, down-triangle, right-triangle, left-triangle
- Color มีให้เลือกดังนี้
  - Yellow, magenta, cyan, red, green, blue, white, black

JUMP: Julia Optimization Package

- A modeling language for mathematical optimization embedded in Julia
- Syntax that mimics natural mathematical expressions.
- Supports many solvers: open-source and commercial.
- Fast



Installing Jump

- ติดตั้ง package JuMP

  Pkg.add(“JuMP”)

- ติดตั้ง Solver

  Pkg.add(“Ipopt”) หรือ

  Pkg.add(“Clp”)

Example

- เรียกใช้งาน package

  ```
  using JuMP
  ```

- สร้างโมเดล (ใช้ชื่ออื่นนอกจาก m ได้)

  ```
  m = Model()
  ```

- กำหนดตัวแปร

  ```
  @variable(m, 0 <= x <= 2)
  ```

  ```
  @variable(m, 0 <= y <= 30)
  ```

Example

- กำหนด objective

  ```
  @objective(m, Max, 5x + 3y)
  ```

- กำหนด constraint

  ```
  @constraint(m, 1x + 5y <= 3.0)
  ```

- ตรวจดูโมเดลก่อนทำการ solve

  ```
  print(m)
  ```

Example

- ทำการ solve

```
status = solve(m)
```

- แสดงค่าของ objective function หลังจาก solve แล้ว

```
println("Objective value: ", getobjectivevalue(m))
```

- แสดงค่าของ optimal variables ที่หาได้

```
println("x = ", getvalue(x))
println("y = ", getvalue(y))
```

Distributions Package

- ติดตั้ง Package

```
Pkg.add("Distributions")
```

- สร้าง distribution ขึ้นมา

```
using Distributions

myDist = normal()
```

- สุ่มค่า 100 ค่า จาก distribution ที่สร้างขึ้นมา

```
samples = rand(myDist, 100)
```

Interesting packages

- Mocha : Deep Learning package
- GaussianMixtures : Large scale Gaussian Mixture Models
- DecisionTree: Decision Tree Classifier and Regressor

For more information see

- http://julialang.org/ - Julia Language
- https://jump.readthedocs.io/en/latest/ - Jump optimization
- http://www.juliaopt.org/ - Julia optimization
- http://pkg.julialang.org/ - List of available packages