National University of Singapore
School of Computing
CS3244: Machine Learning
Tutorial 1

The Learning Problem

Readings: *Learning From Data*, Chapter 1.

Reinforcement learning the outcome of each input set is not linked to a specific outcome, but supervised learning has a clear one-to-one link

1. **Learning Paradigms.** Describe instances of learning problems for the following scenarios. For each scenario, describe a *supervised*, *unsupervised* and *reinforcement* learning problem. For one of the problems, formalize the given components in the learning problem: *input*, *output*, *data*. You need not describe the *hypothesis* nor the *target function* (to think about: why?).

   (a) In NUS (or other university) student domain. Students have problems, many of them, and you are the target audience. Describe problems that you encounter on a daily, weekly or semesterly basis.

   (b) Transshipment Logistics. One of Singapore's mainstay sources of income for decades[1] has been transshipment and the logistics associated with this. Hypothesize problems that occur in this scenario.

2. **Crash course for iPython Notebook[2].** In this part we will set up the interactive Python (iPython) environment on your machine and run an Jupyter notebook supporting iPython, that we will be using occasionally in homeworks and tutorials.

   (a) **Setup.** This can be easy or hard, depending on your platform and willingness to tinker with your machine (desktops and laptops on the three major OSes should be ok; but Droid and iOS is currently not very well supported). You can download and install `Anaconda`[3] which should install all the prerequisites for our course's use.

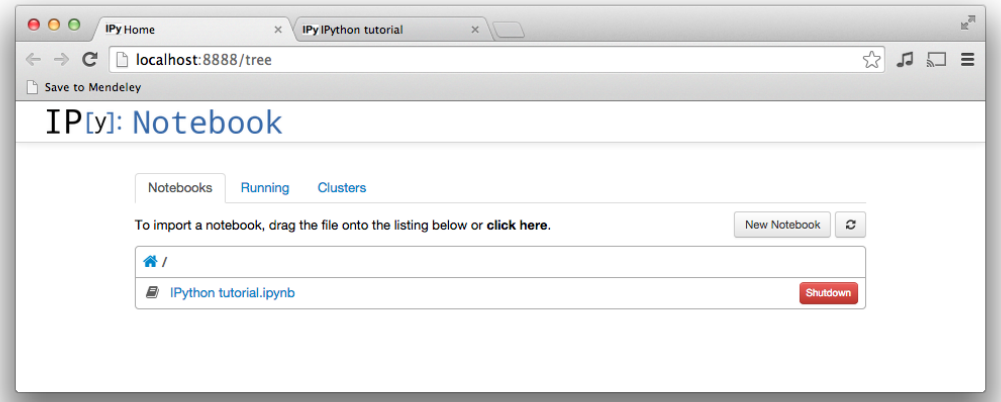
   (b) **Running.** For Mac and Linux users, open up your terminal. Windows users need to open up their Command Prompt. Change directories in the terminal (using the `cd` command) to the working directory where you want to store your IPython Notebook data.

   To run IPython Notebook, enter the following command:

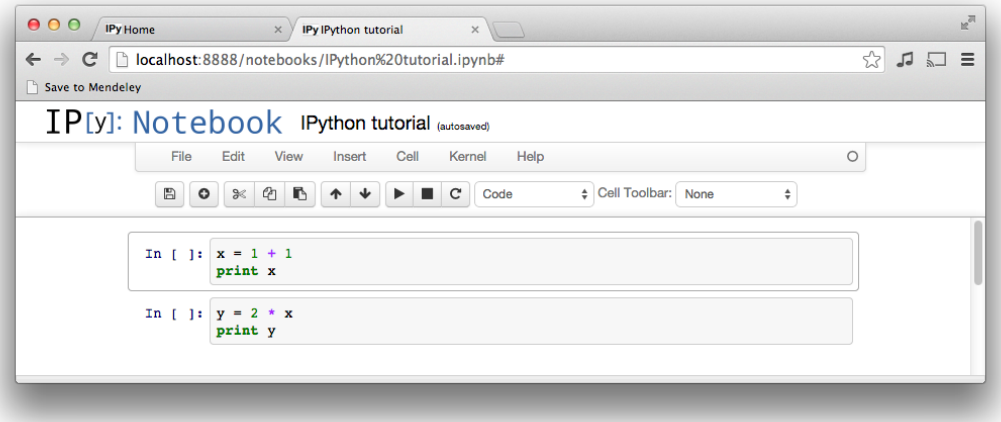
   ```
   1 ipython notebook
   ```

   It may take a minute or two to set itself up, but once IPython is running, it will point an instance of your web browser at http://localhost:8888, showing all available IPython notebooks in the current directory:

[1] Accordingly to Wikipedia, as of 2016, Singapore remains the world's busiest transshipment port.
[2] This section borrows documentation from the webpages http://cs231n.github.io/ipython-tutorial/.
[3] https://www.continuum.io/downloads



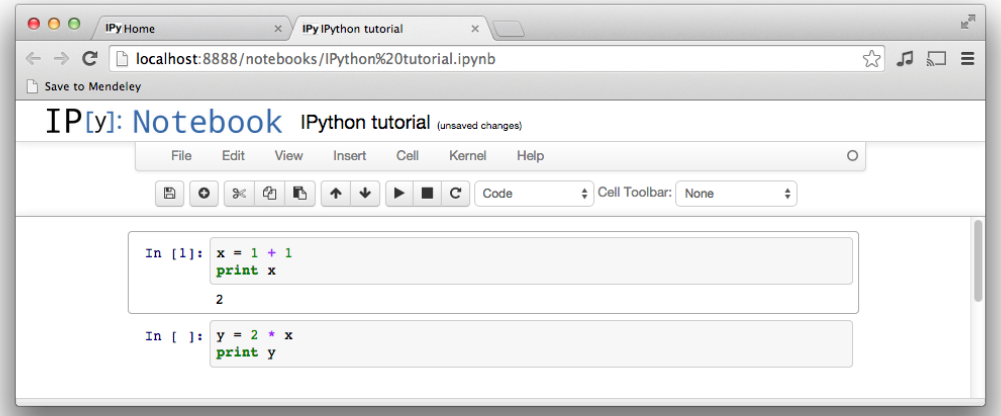
(c) **Executing.** Load the file `tut01-sample.ipynb`, which is a Python 3.x notebook. If you click through to a notebook file, you will see a screen like this[4]:



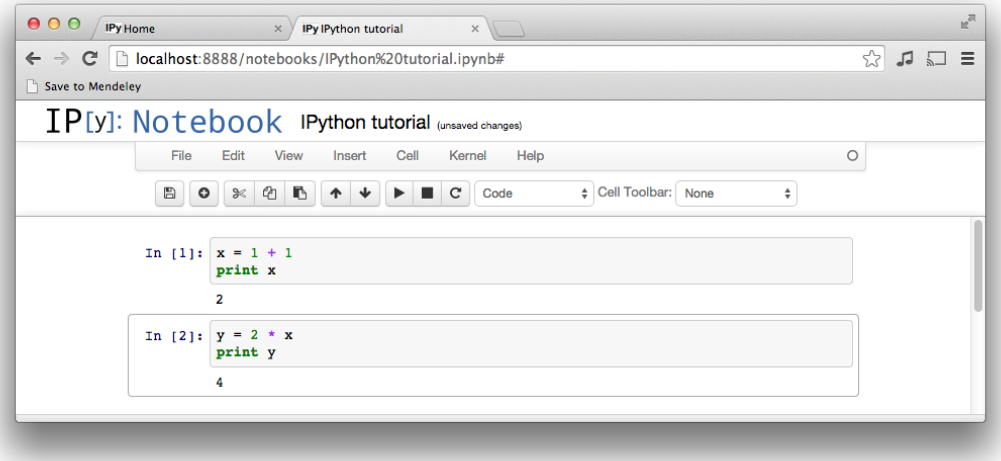
An IPython notebook is made up of a number of **cells**. Each cell can contain Python code. You can execute a cell by clicking on it and pressing Shift – Enter . When you do so, the code in the cell will run, and the output of the cell will be displayed beneath the cell.

[4]Just FYI, the screenshots are from an equivalent sample Python 2.7 notebook, not 3.x.

For example, after running the first cell the notebook looks like this:

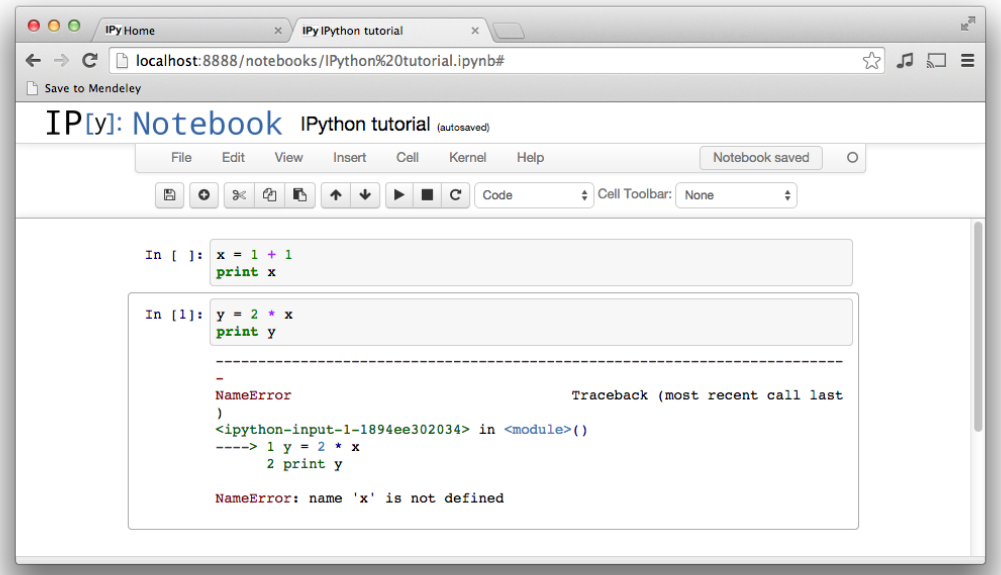


Global variables are shared between cells. Executing the second cell thus gives the following result:

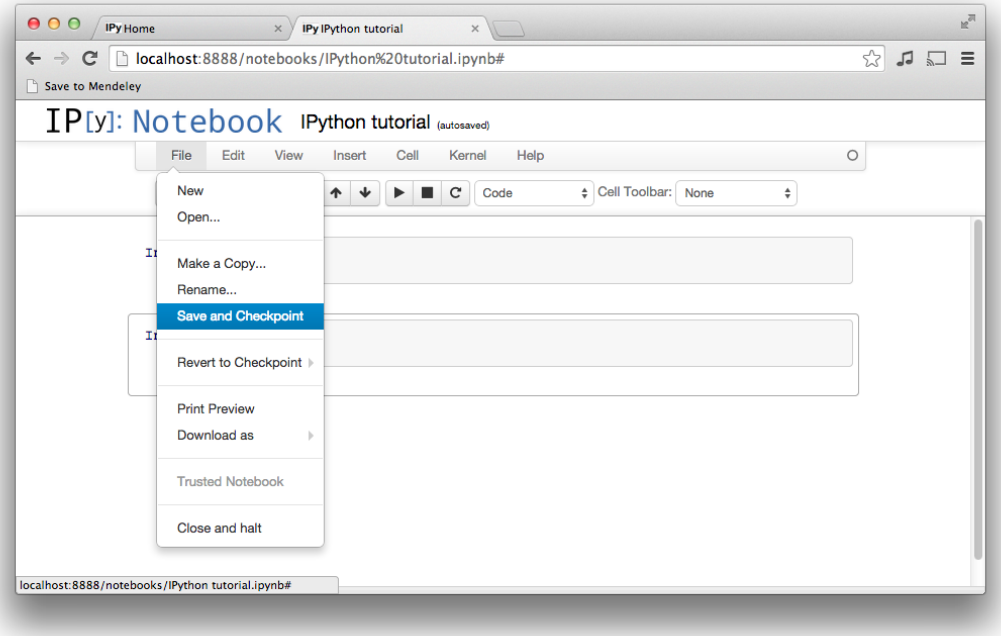


By convention, IPython notebooks are expected to be run from top to bottom.

Failing to execute some cells or executing cells out of order can result in errors:



After you have modified an IPython notebook for one of the assignments by modifying or executing some of its cells, remember to **save your changes!**



This has only been a brief introduction to IPython notebooks, but it should be enough to get you up and running on the assignments for this course.

3. **Perceptron Learning.** From IVLE, load the iPython notebook `tut01-pla.ipynb`. The code has four tasks; the latter two of which are not going to be covered in the scope of this tutorial. You are encouraged to discuss these two later optional exercises on the IVLE forum and attempt all four. The code implements a block of cells to create functions and sample data points from them, and has an incomplete perceptron learning algorithm. For completeness' sake, we rephrase the tasks that you should attempt in the code with respect to the first two tasks.

   (a) In the code, a skeleton of a working (but corrupted) PLA algorithm is given. It does the iteration but does not do the weight update. Modify the algorithm so that it correctly updates the weight. Here is the relevant code snippet, towards the bottom of the `pla()` definition.

```
    for _ in range(maxIter):
        i = nr.randint(N)
        if(yn[i] != g(xn[i,:])):
# If not classified correctly, adjust the line to account
# for that point.
## Begin - Insert your code here
## End - Insert your code here
return w
```

   (b) Identify what other difference this implemented version of PLA has from the version introduced in lecture.

4. **Homework submission.** With any homework submission, you will need to either print or submit in softcopy, an independent work declaration. The iPython notebook has a Markdown cell that simulates this for your practice. Please pay attention to it and attend to the instructions on how to complete it during tutorial.