INTELLIGENT DATA ANALYSIS II

Introduction to Python

Overview

- What is Python?
  - Python is an open general purpose language that is widely used in scientific computing and machine learning.
  - Rich ecosystem of libraries for scientific computation. NumPy for linear algebra, scikit-learn for general machine learning, Apache Spark for distributed ML…

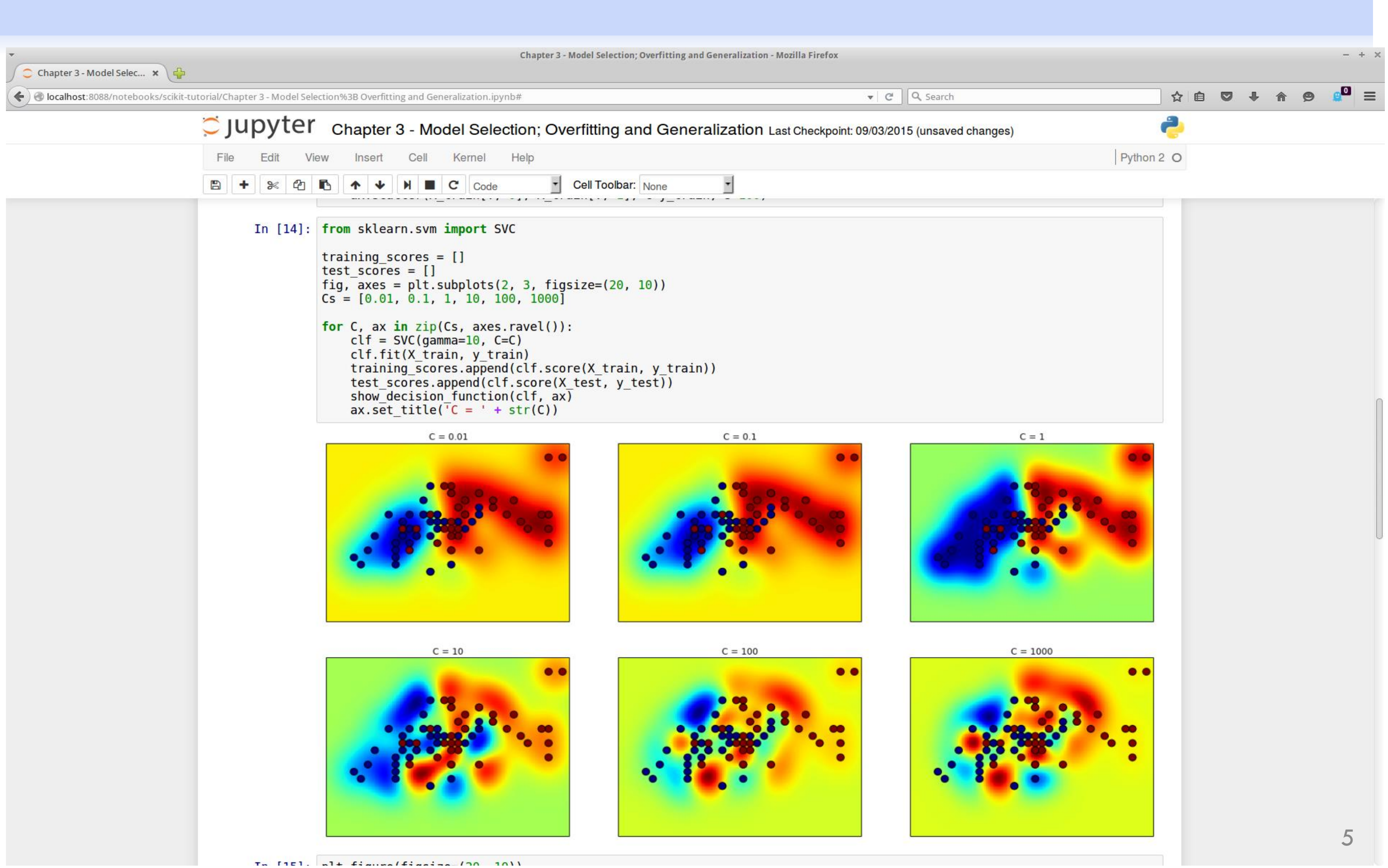
Overview

- Why switch from Matlab?
  - Better suited for recent developments (e.g. parallel/distributed computation);
  - Supposedly better career opportunities for you;
  - Possibly saving tons of license costs for the department.

Plan for this lecture

- □ Today: Live coding in a Python REPL (read-eval-print-loop) with IPython.
- □ Labs are being done in web-based notebooks. You can run snippets of Python code via your web browser (and even output fancy plots!).
- □ In the first exercise meeting we set up this environment. The first lab is an intro to Python and a small ML demonstration.
- □ Note: Python can also be compiled like other languages.

Your notebook server

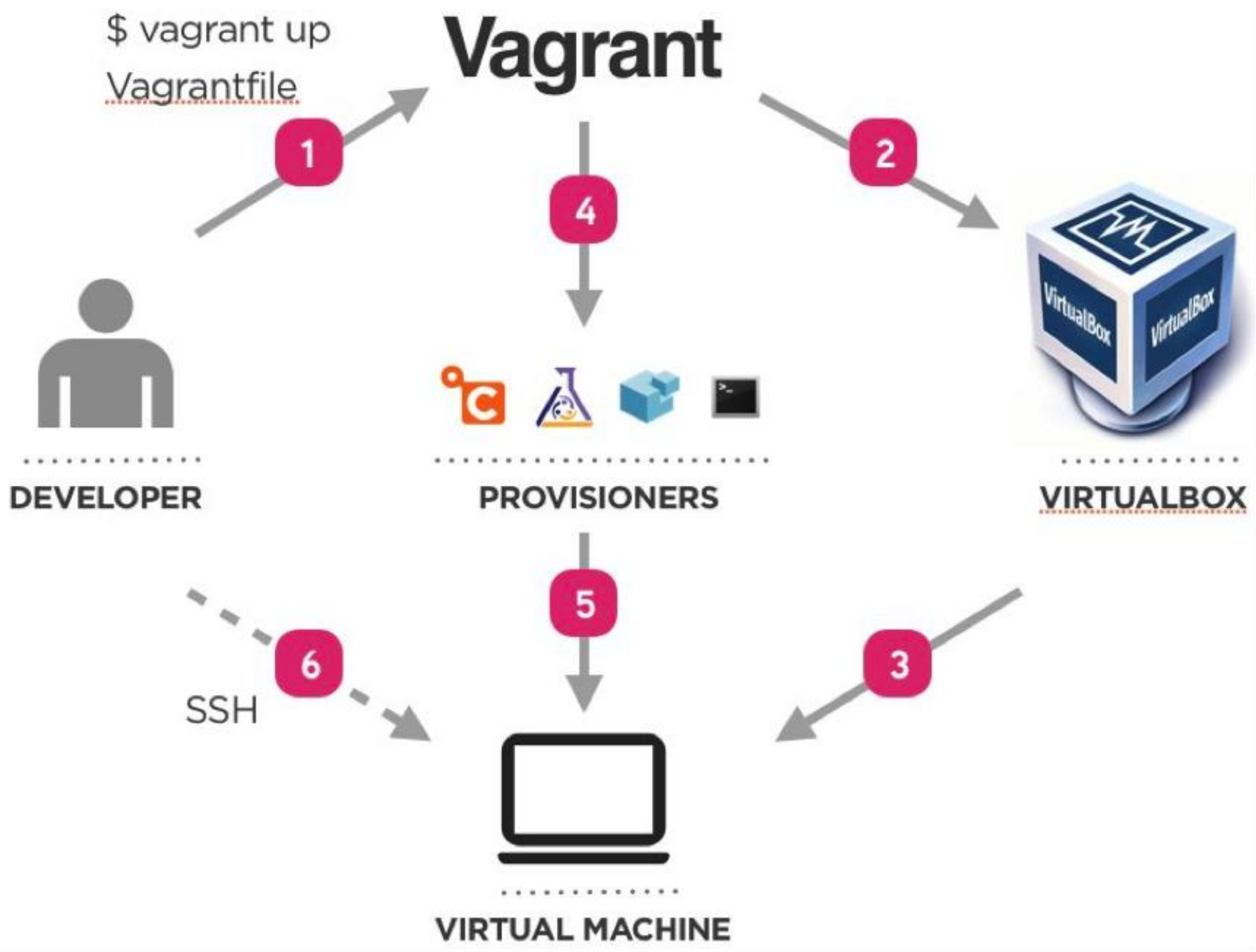

Chapter 3 - Model Selection; Overfitting and Generalization - Mozilla Firefox
Chapter 3 - Model Selec...
localhost:8088/notebooks/scikit-tutorial/Chapter 3 - Model Selection%3B Overfitting and Generalization.ipynb#
Search
jupyter
Chapter 3 - Model Selection; Overfitting and Generalization Last Checkpoint: 09/03/2015 (unsaved changes)
File
Edit
View
Insert
Cell
Kernel
Help
Python 2
Code
Cell Toolbar: None

```
In [14]: from sklearn.svm import SVC

         training_scores = []
         test_scores = []
         fig, axes = plt.subplots(2, 3, figsize=(20, 10))
         Cs = [0.01, 0.1, 1, 10, 100, 1000]

         for C, ax in zip(Cs, axes.ravel()):
             clf = SVC(gamma=10, C=C)
             clf.fit(X_train, y_train)
             training_scores.append(clf.score(X_train, y_train))
             test_scores.append(clf.score(X_test, y_test))
             show_decision_function(clf, ax)
             ax.set_title('C = ' + str(C))
```

C = 0.01
C = 0.1
C = 1
C = 10
C = 100
C = 1000

```
In [15]: plt.figure(figsize=(20, 10))
```

How your setup works



Python

- □ Python is dynamically typed, that means that the type of an expression is unknown before evaluation time. (but there are types!).
- □ Weirdest thing: blocks are given by the indentation (usually TAB).
- □ Supports basic notions of object-orientation and functional programming “well enough”.
- □ We use Python 2.7 in the lecture. Python 3.5 is the latest version, but not every library supports Python 3+.

It’s dangerous to go alone, take these:

- help: opens documentation.
- who, whos: lists all currently available identifiers, latter with more detail.
- del x: deletes x from memory.
- clear: clears output if you run Python in a terminal.

Live Coding: Fundamentals

1. Hello world, variables.
2. Functions.
3. Control flow.
4. Lists.

Live Coding: Math & ML

1. Math module.
2. NumPy.
3. Matplotlib.
4. Scikit-learn.

Cheat Sheet I: Matrix creation

```
import numpy as np
```

Prefer np.ndarray to np.matrix (np.asarray)

```
np.matrix(‘1 2; 3 4’)
np.array([[1, 2], [3, 4]])
np.eye(3)
np.ones((3, 3))
np.zeros((2, 2))
np.empty((3, 4))
np.diag([1, 2, 3])
```

Cheat Sheet II: Selection

```
A = np.random.rand(5, 5) # Without brackets!
A[0, 0] # first element (starts at 0)
A[0, 4] # first row, fifth column
A[0] # returns first row
A[:, 0] # first column
A[:3, 0] # first three columns
A[ [0, 2, 1] ] # select first, third, second row in order
I = A >= 0.5 # matrix of Booleans (true if >= 0.5)
A[I] # selects values >= 0.5 from A as a 1-dim ndarray
```

Cheat Sheet III: Operations on Data

```
A = np.random.rand(5, 4)
A.shape # (number of rows, columns)
A.reshape(20), A.reshape(5, 2, 2) # change of dims
A.flatten() # flatten to row vector, flatten(1) to columns
B = np.random.rand(4, 5)
A.dot(B) # np.dot(A, B) – matrixmultiplication
A * A # element-wise multiplication
A**5 # element-wise power of
A – 3*A + A # scala-mult, addition, substraction
A.T # transpose of A
```

More sources

- https://continuum.io – Anaconda distribution, easy to use installation of Python. Works well under Windows.
- http://learnpythonthehardway.org – A gentle introduction to Python as a general-purpose language.
- https://www.edx.org – Decent (and free) online classes for Python.
  - 6.00.2x: Python intro with scientific/statistical approach. If you lack CS fundamentals start with 6.00.1x.
  - CS190-1x: Large scale ML with Python and Spark. Labs very cool (e.g. visualization of neuroimage data of Jellyfishes).

More sources

- □ https://github.com/amueller – Wonderful collection of tutorials for ML with Python with notebooks, you can find accompanying videos often.
- □ https://github.com/parallel_ml_tutorial -- Parallel ML with Python. Useful for quicker prototyping.