Te Ipu o te Mahara
Artificial Intelligence
Institute
THE UNIVERSITY OF WAIKATO

Bridging the gap: Scripting Weka from Python

When scripting is life

Peter Reutemann

Outline

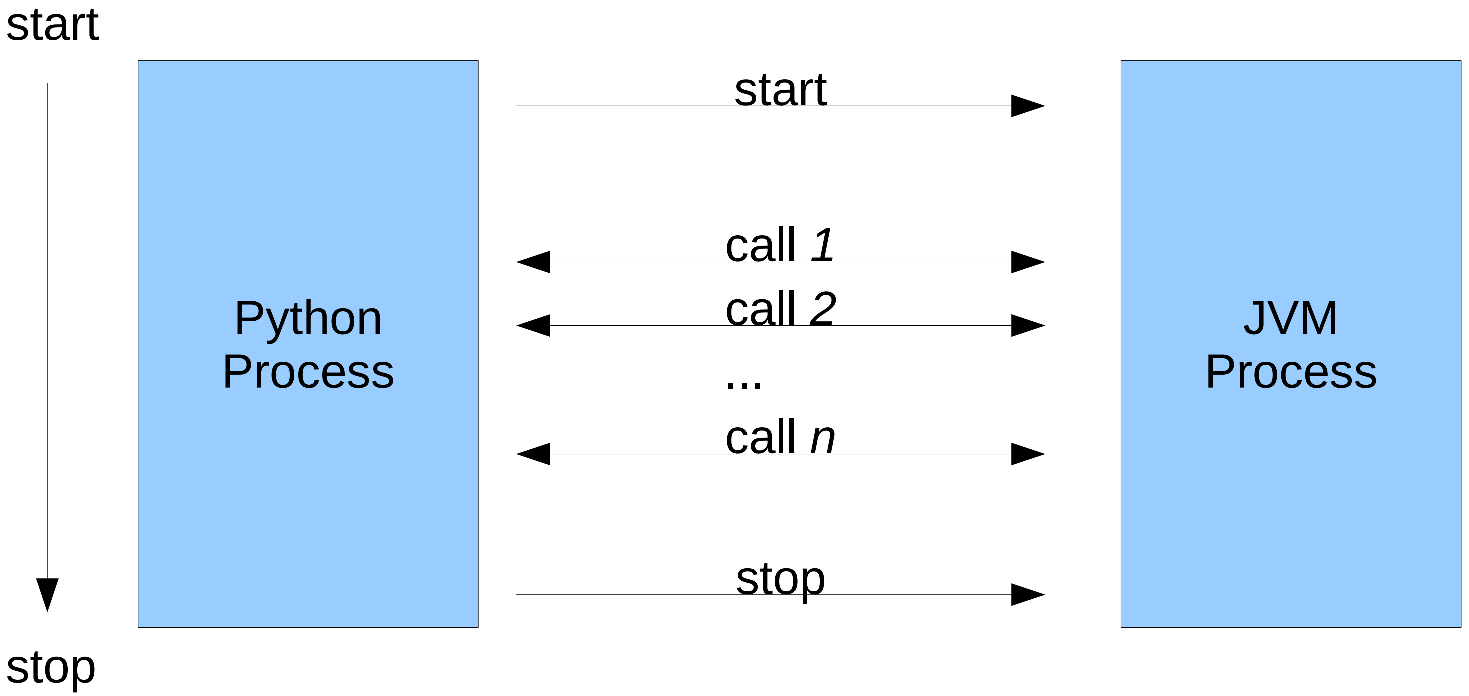
- Motivation
- javabridge
- python-weka-wrapper3
- sklearn-weka-plugin
- Demo

Why?

- Weka's Explorer and KnowledgeFlow applications are too Weka-centric
- Python has gazillions of libraries for
  - loading various data types/sources
  - generating data that Weka can handle (i.e., tabular)
  - visualizing results
- Scripts encapsulate all steps – reproducible results!
- But how to bridge the Java/Python divide?
  - Python library weka launches Java processes - inefficient/limited

javabridge

- Allows starting JVM from Python
- Interacts with JVM via JNI method lookup APIs



Caveat: once the JVM gets stopped, the Python process needs to be restarted

Java Native Interface

- Operations
  - instantiating objects
  - calling object methods
  - calling static methods
  - create call objects (to speed up repeated calls)
- Determine JNI signatures
  - javap – disassembles one or more class files

Example

- Print public JNI signatures of Instances class
  - Command: javap -public -s -cp weka.jar weka.core.Instances

```
public class weka.core.Instances extends java.util.AbstractList<weka.core.Instance> implements java.io.Serializable,
weka.core.RevisionHandler {
  public static final java.lang.String FILE_EXTENSION;
    descriptor: Ljava/lang/String;
...
  public weka.core.Instances(java.io.Reader, int) throws java.io.IOException;
    descriptor: (Ljava/io/Reader;I)V
  public weka.core.Instances(weka.core.Instances);
    descriptor: (Lweka/core/Instances;)V
...
  public weka.core.Instances stringFreeStructure();
    descriptor: ()Lweka/core/Instances;
  public boolean add(weka.core.Instance);
    descriptor: (Lweka/core/Instance;)Z
…
```

- Ugly? You bet… Best to write wrapper code only once!

python-weka-wrapper3

- Wraps the major class hierarchies in Weka

  data generators, I/O converters, stopwords, stemmers, tokenizers, filters, associators, classifiers, clusterers, attribute selection

- Furthermore

  datasets, tags, index, range, package management, database access, experiments, timeseries support, visualization via matplotlib and pygraphviz, basic workflow system

- Instantiation
  - Java class name
  - command-line options if weka.core.OptionHandler
- Low-level Java access (property of JavaObject class):

  jwrapper – returns Python object making methods available as Python attributes

https://github.com/fracpete/python-weka-wrapper3

Example

- Build a classifier and output model

```
import weka.core.jvm as jvm
import weka.core.converters as converters
from weka.classifiers import Classifier

jvm.start(packages=True)

data = converters.load_any_file("/some/where/iris.arff")
data.class_is_last()
cls = Classifier(classname="weka.classifiers.trees.J48",
                 options=["-C", "0.3"])
cls.build_classifier(data)
print(cls)

jvm.stop()
```

sklearn-weka-plugin

- Based on python-weka-wrapper3
- Makes Weka algorithms available in scikit-learn

https://scikit-learn.org/

https://github.com/fracpete/sklearn-weka-plugin

Example

- 10-fold cross-validation of linear regression

```
import sklweka.jvm as jvm
from sklweka.dataset import load_arff
from sklweka.classifiers import WekaEstimator
from sklearn.model_selection import cross_val_score

jvm.start(packages=True)

X, y, meta = load_arff("/some/where/bolts.arff", class_index="last")
lr = WekaEstimator(classname="weka.classifiers.functions.LinearRegression")
scores = cross_val_score(lr, X, y, cv=10,
                         scoring='neg_root_mean_squared_error')
print("Cross-validating LR on bolts (negRMSE)\n", scores)

jvm.stop()
```

Demo

Let's see some live examples!

Questions?

Py{WEKA}3

https://github.com/fracpete/python-weka-wrapper3

https://github.com/fracpete/sklearn-weka-plugin

https://github.com/fracpete/weka-user-conference-2021