www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 6 Issue 10 October 2017, Page No. 22823-22829
Index Copernicus value (2015): 58.10 DOI: 10.18535/ijecs/v6i10.20

Scripting Languages for Research in Data Mining

T Sandeep[1], G.Dhana Laxmi[2], Dr. A Kanaka Durga[3]

[1, 2, 3](IT Department, Stanley College of Engineering and Technology for Women, Hyderabad, India)

Abstract
A number of scripting languages are available today. Based on the characteristics and features they have, their usage is defined. Usage of scripting languages in the field of data mining and other research areas is increasing as they are simple, powerful and more importantly they are open source. This paper discusses about a few scripting languages and their applications in data mining. We have used one of the scripting languages i.e. Python and demonstrated how it is useful for the research.

Keywords—open source; scripting languages, Data Mining.

I. Introduction

Scripting languages play an important role in current research applications. There are many reasons why scripting languages are popular especially in data mining research. Scripting languages are object-oriented, easy to learn and apply. They have powerful string-handling abilities and flexible syntax. They are portable, embeddable, and extensible. They have rich sets of libraries and some of them also provide support for concurrent programming [1].

Scripting languages have applications in different research areas such as data mining, image processing and other areas of computing. Data mining being a field that concerns itself with extracting information from large repositories of data has many phases. Scripting languages come in handy in the implementation of many phases of Knowledge Discovery using Data Mining (KDD) like preprocessing, transformation and data mining. Also, scripting languages find applications in image processing.

II. Background of Scripting Languages

In this paper, the limitation is that, we are confining our study to the applications of scripting languages to data mining. We also provide insight on the various attributes/features of specific scripting languages considered in this study.

The term ‘scripting language’ has been defined from two perspectives namely: the pragmatic perspective and the philosophical perspective [1]. The two perspectives however agree on the fact that scripting languages are interpreted [2] possess automatic memory management and powerful operations tightly built in, rather than relying on libraries [1]. In recent times, they also possess dynamic and strong typing as seen in Python, Perl and a host of others.

Scripting languages play an important role in data mining research as well as image processing. Their significance cannot be underestimated. Scripting languages originated as a result of the development of the internet as a tool of communication. Rather than being compiled, scripting languages are usually being interpreted. In the next section we review ten popular scripting languages by examining their features, advantages, and limitations.

III.Features, Advantages And Limitations Of Popular Scripting Languages

This section describes the features, advantages and limitations of Python, Haskell, Lua, Perl, Scala, PHP, JavaScript, Erlang, R and Ruby as ten popular scripting languages. They are popular in the sense that they rank higher in comparison to other known scripting languages in the TIOBE programming community index [4]:

A. Python

Python is a general-purpose, high-level programming language that also provides scripting capability [3]. It first appeared in 1991 and was designed by Guido van Rossum. The language was influenced by ABC, ALGOL 68, C, Haskell, Lisp, Modula-3, Perl, and Java. It has also influenced the design of other languages namely: Boo, Cobra, D, Falcon, Groovy, Ruby, and JavaScript.

B. Haskell

It is an advanced, purely-functional programming language that supports scripting capabilities [5]. It first appeared in 1990 and is an open-source product of more than twenty years of cutting-edge research which allows rapid development of robust, concise, correct software. The language was influenced by languages like: Standard ML, Lisp, and Scheme. It has in turn also influenced several other languages like: Python and Scala

C. Lua

Lua is a powerful, fast, lightweight, embeddable language that first appeared in 1993. "Lua" (pronounced LOO-ah) means "Moon" in Portuguese [6]. Roberto et al. [7] designed the language. The language was inspired by C++, CLU, Modula, Scheme and SNOBOL. It has in turn inspired languages like: Io, GameMonkey, Squirrel, Falcon and MiniD.

D. Perl

Perl is a highly capable, feature-rich programming language that first appeared in 1987 [8]. It was developed by Larry Wall and can be used in mission critical projects. The language was influenced by languages like: AWK, Smalltalk 80, Lisp, C, C++, sed, UNIX shell, and Pascal. It has in turn influenced the creation of Python, PHP, Ruby, JavaScript, and Falcon.

E. Scala

It is a general purpose programming language designed to express common programming patterns in a concise, elegant, and type-safe way. It was designed by Martin Odersky and first appeared in 2003 [9]. The language was inspired by languages like Eiffel, Erlang, Haskell, Java, Lisp, Pizza, Standard ML, OCaml, Scheme and Smalltalk. It has in turn influenced the following languages namely: Fantom, Ceylon, and Kotlin.

F. PHP

It is a widely used general purpose scripting language that is especially suited for Web development and can be embedded into HTML. It was designed by Rasmus Lerdorf [10] using the C programming language and first appeared in 1995. PHP was influenced by Perl, C, C++, Java and Tcl.

G. JavaScript

JavaScript is a lightweight programming language that first appeared in 1994 and was designed by Brendan Eich [11]. The language was influenced by C, Java, Perl, Python, Scheme, Self. It has in turn influenced ActionScript, CoffeeScript, Dart, Jscript .NET, Objective-J, QML, TIScript, and TypeScript.

H. Erlang

Erlang is a programming language designed at the Ericsson Computer Science Laboratory. It first appeared in 1986. The language was influenced by Prolog and ML. It has in turn influenced F#, Clojure, Rust, Scala, Opa and Reia [12].

I. R

R is a language and environment for statistical computing and graphics. It was designed by Ihaka and Gentleman and first appeared in 1993 [13]. It was influenced by S, Scheme, and XLispStat.

J. Ruby

It is a dynamic open source programming language with a focus on simplicity and productivity. It was designed by Yukihiro Matsumoto [14] and first appeared in 1995. The language was influenced by Ada, C++, CLU, Dylan, Eiffel, Lisp, Perl, Python, and Smalltalk. It has also in turn influenced Falcon, Fancy, Groovy, loke, Mirah, Nu, and Reia.

IV. Python Packages for Data Mining

i. Numpy

NumPy is the fundamental package for scientific computing with Python. NumPy is an extension to the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large library of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications.

ii.Scipy

SciPy (pronounced "Sigh Pie") is open-source software for mathematics, science, and engineering. The SciPy library depends on NumPy, which provides convenient and fast N-dimensional array manipulation. The SciPy library is built to work with NumPy arrays, and provides many user-friendly and efficient numerical routines such as routines for numerical integration and optimization. Together, they run on all popular operating systems, are quick to install, and are free of charge. NumPy and SciPy are easy to use, but powerful enough to be depended upon by some of the world's leading scientists and engineers. If you need to manipulate numbers on a computer and display or publish the results, Scipy is the tool for the job.

iii. Pandas

Pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis / manipulation tool available in any language. It is already well on its way toward this goal.

Pandas is well suited for many different kinds of data:

Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet.
Ordered and unordered (not necessarily fixed-frequency) time series data.
Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels.
Any other form of observational / statistical data sets. The data actually need not be labeled at all to be placed into a pandas data structure.

iv.Matplotlib

Matplotlib is a plotting library for the Python programming language and its NumPy numerical mathematics extension. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like wxPython, Qt, or GTK+. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB. SciPy makes use of matplotlib.

v.Ipython

IPython is a command shell for interactive computing in multiple programming languages, originally developed for the Python programming language, that offers enhanced introspection, rich media, additional shell syntax, tab completion, and rich history. IPython currently provides the following features:
Powerful interactive shells (terminal and Qt-based).
A browser-based notebook with support for code, text, mathematical expressions, inline plots and other rich media.
Support for interactive data visualization and use of GUI toolkits.
Flexible, embeddable interpreters to load into one's own projects.
Easy to use, high performance tools for parallel computing.

V. Using Python Packages- An Example

i. **How to install Python**

- Install Python runtime and pip package manager
  a. Go to python.org.
  b. Click on the appropriate Windows installer msi link.
  c. Once downloaded run the msi to install Python runtime
- Download pymssql module.
  Make sure you choose the correct whl file. For example : If you are using Python 2.7 on a 64 bit machine choose : pymssql-2.1.1-cp27-none-win_amd64.whl. Once you download the .whl file place it in the the C:/Python27 folder.

.

- Open cmd.exe
- Install pymssql module
  For example, if you are using Python 2.7 on a 64 bit machine:
  > cd c:\Python27
  > pip install pymssql‑2.1.1‑cp27‑none‑win_amd64.whl

For Ubuntu Linux

- Install Python runtime and pip package manager Python comes pre-installed on most distributions of Ubuntu. If your machine does not have python installed, you can get either download the source tarball from python.org and build locally, or you can use the package manager:
  > sudo apt-get install python
- Open terminal
- Install pymssql module and dependencies
  > sudo apt-get --assume-yes update
  > sudo apt-get --assume-yes install freetds-dev freetds-bin
  > sudo apt-get --assume-yes install python-dev python-pip
  > sudo pip install pymssql

ii. After installing Python install the appropriate packages for the application using pip installer.

iii. After the installation of appropriate package, we can use the functions supported by the package.

iv. We can not only use the existing package libraries but also create our own package libraries and make it available to the Python user group.

v. Reading in a dataset from a CSV file[15]

The code corresponding to reading a csv file is shown below.

```
import numpy as np

# reading in all data into a NumPy array
all_data = np.loadtxt(open("./wine_data.csv","r"),
delimiter=",",
skiprows=0,
dtype=np.float64
)

# load class labels from column 1
y_wine = all_data[:,0]

# conversion of the class labels to integer-type array
y_wine = y_wine.astype(np.int64, copy=False)
```

```
# load the 14 features
X_wine = all_data[:,1:]

# printing some general information about the data
print('\ntotal number of samples (rows):', X_wine.shape[0])
print('total number of features (columns):', X_wine.shape[1])

# printing the 1st wine sample
float_formatter = lambda x: '{:.2f}'.format(x)
np.set_printoptions(formatter={'float_kind':float_formatter})
print('\n1st sample (i.e., 1st row):\nClass label: {:d}\n{:}\n'
.format(int(y_wine[0]), X_wine[0]))

# printing the rel.frequency of the class labels
print('Class label frequencies')
print('Class 1 samples:
{:.2%}'.format(list(y_wine).count(1)/y_wine.shape[0]))
print('Class 2 samples:
{:.2%}'.format(list(y_wine).count(2)/y_wine.shape[0]))
print('Class 3 samples:
{:.2%}'.format(list(y_wine).count(3)/y_wine.shape[0]))
```

The output of the above code is as shown next.

```
total number of samples (rows): 178
total number of features (columns): 13

  1st sample (i.e., 1st row):
  Class label: 1
[14.23 1.71 2.43 15.60 127.00 2.80 3.06 0.28 2.29 5.64 1.04 3.92 1065.00]

Class label frequencies
Class 1 samples: 33.15%
Class 2 samples: 39.89%
Class 3 samples: 26.97%
```

vi. Visualization of a dataset

There are endless ways to visualize datasets for get an initial idea of how the data looks like. The most common ones are probably histograms and scatter plots.

Histograms are a useful data to explore the distribution of each feature across the different classes. This could provide us with intuitive insights which features have a good and not-so-good inter-class separation. Below, we will plot a sample histogram for the "Alcohol content" feature for the three wine classes.

```
from matplotlib import pyplot as plt
from math import floor, ceil # for rounding up and down
plt.figure(figsize=(10,8))
# bin width of the histogram in steps of 0.15
bins = np.arange(floor(min(X_wine[:,0])), ceil(max(X_wine[:,0])), 0.15)
# get the max count for a particular bin for all classes combined
max_bin = max(np.histogram(X_wine[:,0], bins=bins)[0])
# the order of the colors for each histogram
```

```
colors = ('blue', 'red', 'green')
for label,color in zip(
        range(1,4), colors):
    mean = np.mean(X_wine[:,0][y_wine == label]) # class sample mean
    stdev = np.std(X_wine[:,0][y_wine == label]) # class standard deviation
    plt.hist(X_wine[:,0][y_wine == label],
             bins=bins,
 alpha=0.3, # opacity level
             label='class {} ($$\mu={:.2f}$$, $$\sigma={:.2f}$$)'.format(label, mean, stdev),
             color=color)
plt.ylim([0, max_bin*1.3])
plt.title('Wine data set - Distribution of alocohol contents')
plt.xlabel('alcohol by volume', fontsize=14)
plt.ylabel('count', fontsize=14)
plt.legend(loc='upper right')
plt.show()
```

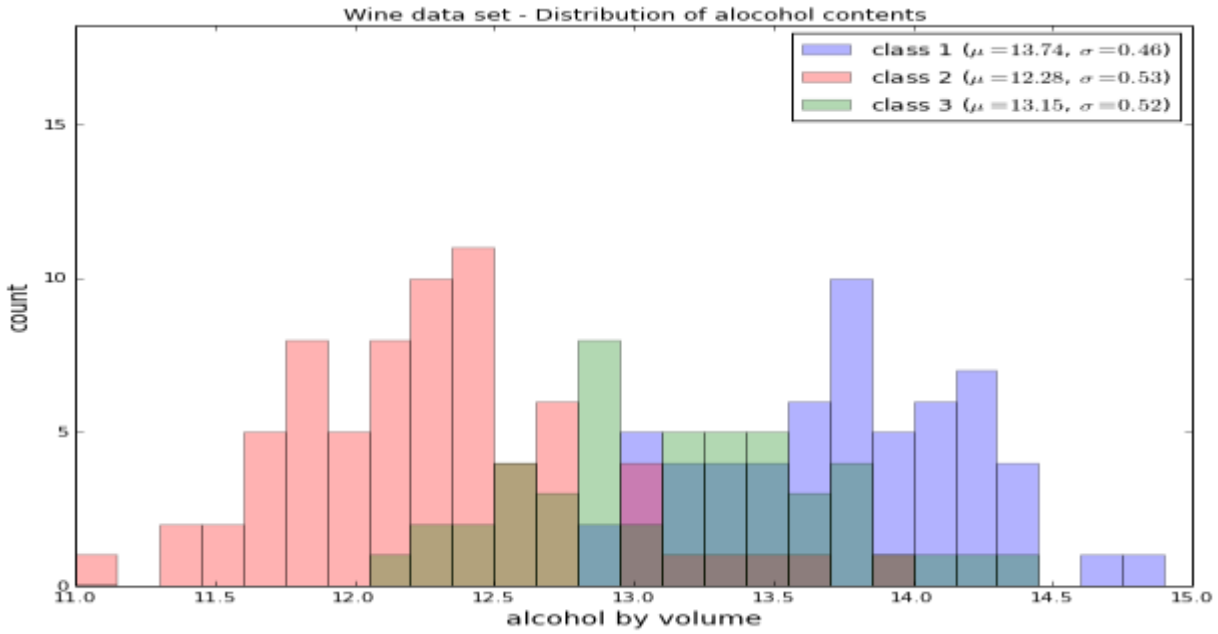

Fig 1: Plotting wine data set using Matplotlib

VI. Conclusion

We have used Python Packages numpy, math, matplotlib and demonstrated how to read data from a data set and demonstrate the data by plotting using the matplotlib package. This package is similar to matlab package and supports almost all the operations supported by Matlab. The main advantage of using Python is it is a free and open source software easily available for all kinds of uses and it supports a lot of packages related to different areas. We conclude here with the application related to data mining and Python can also be used in other areas of research like Image Processing, Software Engineering, Biotechnological applications, Aerospace applications etc,.

Acknowledgement

We would like to express our heartfelt thanks to the Principal and management of Stanley College of Engineering and Technology for Women, Hyderabad for their valuable help and support in this work.

References

1. L. Prechelt, "Are scripting languages any good? A validation of Perl, Python, Rexx, and Tcl against C, C++ and Java," Advances in Computers, vol. 57, pp. 205-270, 2003.

2. J. K. Ousterhout, "Scripting: Higher level programming for the 21st century," Computer, vol. 31, pp. 23-30, 1998.
3. J. Python, "Python programming language," USENIX Annual Technical Conference, 2007.
4. http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html.
5. http://www.haskell.org/haskellwiki/Haskell.
6. http://www.lua.org/about.html
7. http://www.lua.org/authors.html
8. http://www.perl.org/about.html
9. http://www.scala-lang.org/node/241
10. R. Lerdorf and K. Tatroe, "Programming PHP", O' Reilly, pp. 5, 2002
11. http://www.aminutewithbrendan.com/ .
12. www.erlang.org/about.html
13. http://en.wikipedia.org/wiki/R_%28programming_language%29
14. http://www.ruby-lang.org/en/about/
15. http://sebastianraschka.com/Articles/2014_scikit_dataprocessing.html#reading-in-a-dataset-from-a-csv-file