CSC 458 Data Mining and Predictive Analytics I, Fall 2017, DEMO FALL 2020 csc523

The solution to this fall 2017 assignment 1 is on acad in ~parson/DataMine/ csc458fall2017_csc458fall2017_prepdata1.zip. We will go over it as our first example of Python regular expressions.
This is NOT an assignment for csc523 fall 2020.

Dr. Dale E. Parson, Assignment 1, Using Python 3.x scripting constructs and regular expressions to read and parse loosely structured textual data and to write an ARFF (attribute-relation file format) table of data for possible analysis. Due by 11:59 PM on Thursday October 5 via make turnitin.
You can stay on **acad** or **ssh mcgonagall** to run **make test**. Use the **python** or **ipython** 3.x interactive interpreters on mcgonagall, or install python 3.x on your machine per the course web page.

The goals of this assignment are to practice using Python programming constructs, data types, and its regular expression library to crack apart a textual data file and create an ARFF file amenable to analysis with the Weka data mining tool. This is the only programming (scripting) assignment this semester. Note that writing data cleaning scripts can account for as much as 50% of a data analyst's workload.

Perform the following steps to set up for this semester's projects and to get my handout. Start out in your login directory on csit (a.k.a. acad).

cd $HOME
mkdir DataMine
cp ~parson/DataMine/csc458fall2017_prepdata1.solution.zip DataMine/csc458fall2017_prepdata1.solution.zip
cd ./DataMine
unzip csc458fall2017_prepdata1.solution.zip
cd ./csc458fall2017_prepdata1
make testls
make test

Running **make testls** succeeds in running my **example script lsTOarff.py** that reads input file lsTOarff.rawtestdata.txt[1] that contains data from a recursive call to the **ls** command on my /home/kutztown.edu/parson/JavaLang directory; this script creates output file lsTOarff.arff[2] that contains data extracted from lsTOarff.rawtestdata.txt by lsTOarff.py and formatted into the ARFF file for use by the Weka data mining tool. Script lsTOarff.py is a correct, completed script that you must understand and use an example to complete your assignment. Do **not** change lsTOarff.py or any of the lsTOarff files.

$ **make testls**
'# Install low-space symbolic links for input & reference files.
/bin/rm -f ./lsTOarff.rawtestdata.txt ./lsTOarff.arff.ref
/bin/ln -s /home/kutztown.edu/parson/DataMine/lsTOarff.rawtestdata.txt ./lsTOarff.rawtestdata.txt
/bin/ln -s /home/kutztown.edu/parson/DataMine/lsTOarff.arff.ref ./lsTOarff.arff.ref
/usr/bin/python[3] ./lsTOarff.py /home/kutztown.edu/parson/DataMine/lsTOarff.rawtestdata.txt lsTOarff.arff
grep -v "created at" < lsTOarff.arff > lsTOarff.out

[1] /home/kutztown.edu/parson/DataMine/lsTOarff.rawtestdata.txt is the path to the input data file.
[2] /home/kutztown.edu/parson/DataMine/lsTOarff.arff.ref holds the test reference file.
[3] The makefile uses Python version 3.x instead of 2.x, although my example lsTOarff.py now works in both versions after I migrated it to 3.x. The path to python will differ from /usr/bin/python on acad and mcgonagall. The makefile selects a Python 3.x.

```
diff lsTOarff.out /home/kutztown.edu/parson/DataMine/lsTOarff.arff.ref > lsTOarff.dif
# Running weka from the command line.
# See http://www.cs.waikato.ac.nz/~remco/weka_bn/node13.html
# http://weka.wikispaces.com/Primer also useful for command line.
java -cp /home/kutztown.edu/parson/weka/weka-3-8-1/weka.jar
"weka.filters.unsupervised.attribute.StringToNominal" -R 1,12,13,17 -i lsTOarff.arff -o
lsTOarffNominals.arff
```

Running **make test** fails initially because you must complete the definition of file psTOarff.py that I have started. Script psTOarff.py when completed will analyze file psTOarff.rawtestdata.txt[4] created by the Unix **ps** command for examining status of running processes, and will create output file psTOarff.arff[5]. Please look for **STUDENT** comments in file psTOarff.py, and do **not** remove the parts that I have already solved. Script file psTOarff.py was originally a working solution from which I removed portions that you must now complete, starting with your name near the top. **Make sure to indent Python using only spaces (no tabs). My handout code uses 4 spaces per indentation level. Use that**.

I will be spending extensive class time going over the workings of example Python script lsTOarff.py and of the requirements for you to complete script file psTOarff.py. Plan to attend. If you must miss a class due to illness or an emergency, you can use the Blackboard Collaborate Ultra archive to view a recording of a class session. Your goals are to understand as completely as possible, A) the requirements and workings of example script lsTOarff.py, B) the requirements for completion of your script psTOarff.py, and then C) the successful completion of your script. I will go over the logistics of my makefile-driven testing approach in class.

Run **make turnitin** on acad by the due date. The late penalty is 10% per day, and I will not accept solutions after I go over an assignment. Plan to attend all classes, either in person (sections 010 and 101), or via Ultra (all class sections), and ask questions. Running **make turnitin** does not send you email. A successful run looks roughly like the following. It prints an error message when it does not work, usually due to running out of file space in a student account. If that happens, let me know.

```
$ make turnitin
/bin/rm -f *.o *.class .jar core *.exe *.obj *.pyc
/bin/rm -f *.out *.o *.arff *.dif *.out
# Remove the symbolic links
/bin/rm -f lsTOarff.arff.ref
/bin/rm -f lsTOarff.rawtestdata.txt
/bin/rm -f psTOarff.arff.ref
/bin/rm -f psTOarff.rawtestdata.txt
Do you really want to send csc458fall2017_prepdata1 to Professor Parson?
Hit Enter to continue, control-C to abort.
/bin/bash -c "cd .. ; /bin/chmod 700 .                    ; \ /bin/tar cvf ./csc458fall2017_prepdata1_parson.tar
csc458fall2017_prepdata1      ; \
                    /bin/gzip ./csc458fall2017_prepdata1_parson.tar                ; \
                    /bin/chmod 666 ./csc458fall2017_prepdata1_parson.tar.gz        ; \
                    /bin/mv ./csc458fall2017_prepdata1_parson.tar.gz ~parson/incoming"
```

[4] /home/kutztown.edu/parson/DataMine/psTOarff.rawtestdata.txt

[5] /home/kutztown.edu/parson/DataMine/psTOarff.arff.ref

csc458fall2017_prepdata1/
csc458fall2017_prepdata1/makelib
csc458fall2017_prepdata1/makefile
csc458fall2017_prepdata1/psTOarff.py
csc458fall2017_prepdata1/lsTOarff.py
csc458fall2017_prepdata1/makewho.sh