Unix data tools project

By: Joel Swift

Generating the list of bands

For this task I visited wikipeida and found a list of all punk rock bands (at least most of them). I downloaded the source code behind the wikipedia page and then proceeded to format it

0-K

https://en.wikipedia.org/wiki/List_of_punk_rock_bands,_0%E2%80%93K

L-Z

https://en.wikipedia.org/wiki/List_of_punk_rock_bands,_L%E2%80%93Z

Rough formatting

I preformed formatting in sublime text as I needed to see the changes happening since the band names can be filled with symbols (e.g. ’“/).

```
search term : ^\| \[\[(.*)\]\] \|\| \[\[(.*)\]\], [A-Z]+
replacement : $1\t$2

----------------------------------------------------------------------

search term : ^\|-\n
replacement :

----------------------------------------------------------------------

search term : ^\|\}\n
replacement :

----------------------------------------------------------------------

search term : ^==\w==\n
replacement :

----------------------------------------------------------------------

search term : \(band\)
```

```
replacement :

--------------------------------------------------------------------

search term : ^\|\s\[\[(.*)\]\] \|\| \[\[([A-Z ,.]*)\]\]
replacement : $1\t$2

--------------------------------------------------------------------

search term : ^\|\s\[\[([A-Z|! ]+)\]\]\s\|\|\s
replacement : $1\t

--------------------------------------------------------------------

search term : ^\|\s\[\[([A-Z,0-9|! ]+)\]\]\s\|\|\s
replacement : $1\t

--------------------------------------------------------------------

search term : ^([+A-Z,0-9 .]+) \|\1
replacement : $1

--------------------------------------------------------------------

search term : \|\|\s([0-9,-]*)[A-Z,0-9]*.*\|\|.*$
replacement : $1

--------------------------------------------------------------------

search term : -.*$
replacement :

--------------------------------------------------------------------

search term : ([A-Z ]+), ([A-z]+)
replacement : $1\t$2

--------------------------------------------------------------------

search term : \s([0-9]{2,4})$
replacement : \t$1

--------------------------------------------------------------------
```

After these search and replace searches in sublime text 3, I manually editted the bits in order to clean up the dataset. This was done in libre office calc the process involved swaping country names for 3 digit codes (http://www.nationsonline.org/oneworld/country_code_list.htm) and clean up any missing information such as year of origin. The list was then cut to only bands from the USA, mostly because this is the music I like and because the dataset was already massive with the bands from the USA alone. I have included these files in the tar.

Generating the song lists files

Songs and lyrics obtained from lyricWiki, all credit to the thousands of supporters of the website!

Saved the USA list as a .csv file and used it with the follwing shell script.

Example band list file structure

```
more bands_A-J_USA.csv

BAND,LOWER_POLITICAL,UPPER_POLITICAL,COUNTRY,ORIGIN_YEAR
Blanks 77,Hillside,New Jersey,USA,1990
Bl\'ast,Santa Cruz,California,USA,1984
Blatz,Berkeley,California,USA,1989
Blink 182,Poway,California,USA,1992
Dr. Know,Oxnard,California,USA,1981
Dropdead,Providence,Rhode Island,USA,1990
Dropkick Murphys,Quincy,Massachusetts,USA,1996
Drunk Injuns,San Jose,California,USA,1983
The Ducky Boys,Boston,Massachusetts,USA,1995
Dwarves,Chicago,Illinois,USA,1986
DYS,Boston,Massachusetts,USA,1983
Dystopia,Orange County,California,USA,1991
Early Graves,San Francisco,California,USA,2007
Earth Crisis,Syracuse,New York,USA,1989
The Faith,Washington,D.C.,USA,1981
Fang,Berkeley,California,USA,1981
...
```

This script was made for both A-J and L-Z bands

```
!/usr/bin/env bash

for i in $(cut -d ',' -f 1 bands_A-J_USA.csv | sed 's/ /_/g'); do

    echo $i
    wget http://lyrics.wikia.com/wiki/$i > $i

    #sleep to avoid pissing off lyricwikis server
    sleep 2s
done


for i in $(cut -d ',' -f 1 bands_A-J_USA.csv | sed 's/ /_/g'); do

        echo $i
        egrep -o '<b><a href="/wiki/'$i':[A-Z,a-z,0-9% _]+"' /
        "$i".1 | egrep -o '".*"' | cut -c 2- | rev | /
        cut -c 2- | rev > "$i"_songlist.txt

done
```

Example songlist file structure

```
more Fugazi_songlist.txt
...
/wiki/Fugazi:Five_Corporations
/wiki/Fugazi:Caustic_Acrostic
/wiki/Fugazi:Closed_Captioned
/wiki/Fugazi:Floating_Boy
/wiki/Fugazi:Foreman%27s_Dog
/wiki/Fugazi:Arpeggiator
/wiki/Fugazi:Guilford_Fall
/wiki/Fugazi:Pink_Frosty
...
```

Generating lyric files

Shell script to get lyrics for each song in a song list file.

```
#!/usr/bin/env bash

for band in $(ls -1 *txt); do

    #echo $band

    ###Shout outs to stdout to signify progress###
    echo "starting band $band"

    for i in $(cat $band); do

    ###Fixing varibles###
    bandonly=${band::-13}
    songonly=`echo "$i" | egrep -o ":.*$" | tr -d ":"`

    lyrics $bandonly $songonly --separator " " >> /
    lyricsdir/$bandonly.txt

    ###TESTING###
    #echo $i
    #echo $bandonly
    #echo $songonly

    done

    egrep -v "ERROR: Cannot download lyrics" /
    lyricsdir/$bandonly.txt | sed 's/ /\n/g' | /
    tr -d ',()?' | sed '/^$/d' > $$
    mv $$ lyricsdir/$bandonly.txt

    echo "$band done"
done
```

Below is a sample of just a single file, basically each song was split at space characters to produce a word list. This was appended to a list for the entire discography of the band, thus producing a single file with one word per line for every song of the band.

```
cat Bad_Religion.txt
...
The
lonely
quest
for
meaning
and
the
universe
is
dreaming
...
```

Cleaning up the lyric files

After looking at the resulting lyric files (e.g. Bad_Religion.txt), I noticed that there were many characters that are not words (e.g. -,.,") and words were counted incorrectly due to a mix of both upper and lower case letters. I decided the best way to deal with this was to run a few one liners to remove most of the junk charcters and the use tr (translate) to convert to all lowercase letters.

```
#Find empty files and delete
find . -empty -delete

#Remove lone '-' in word lists
for i in "*.txt"; do sed -i 's/^-$//g' $i; done

#remove begining and ending '-' in word lists
for i in "*.txt"; do sed -i 's/^\-//g' $i; done

#and ending '-'
for i in "*.txt"; do sed -i 's/\-*$//g' $i ; done

#and a few more for small clean ups
for i in "*.txt"; do sed -i 's/^\-*$//g' $i; done
for i in "*.txt"; do sed -i 's/^\-*//g' $i; done

#Remove "'s
for i in "*.txt"; do sed -i 's/"//g' $i; done

#Remove .'s ! and []

for i in $(ls -1 *.txt); do tr -d '\/\.\!\]\[' /
< $i > $$; mv $$ $i; done

#convert to lower case

for i in $(ls -1 *.txt); do tr A-Z a-z < $i > $$; mv $$ $i; done
```

Tanget 1 (Zipf's Law)

I choose to investigate if the "punk rock corpus" fit zipfs law (clip from wikipdeia below).

"Zipf's law states that given a large sample of words used, the frequency of any word is inversely proportional to its rank in the frequency table."

Bash bit

```
cat *.txt | sort | uniq -c | sort -rn | sed -r "s/^\s+//g" | /
sed -r "s/\s+/\t/g" > Zipfs_law.tsv
```

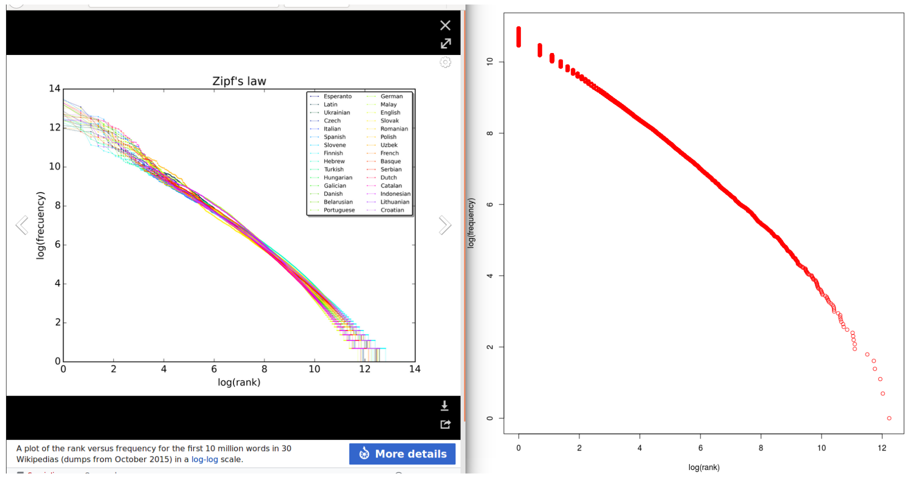
Plotting the graph

```
getwd()
setwd("/home/kenizzer/Downloads")
```

```
x <- read.csv("Zipfs_law.tsv", header = FALSE, sep = "\t")

x2 <- cbind(x, 1:nrow(x))

plot(log(x2[,1]), log(x2[,3]), xlab = "log(rank)", /
ylab = "log(frequency)", col = "red")
```



A plot of the rank versus frequency for the first 10 million words in 30 Wikipedias (dumps from October 2015) in a log-log scale.

Figure 1: Zipf's law plot: Log(rank) plotted against Log(frequency), at left is graph from wikicommons and right is my graph.

Generating data matrix for machine learning applications

There is more coming to this at a later time, but I need to turn in what I got to. I plan to use this matrix to play around with in scikitlearn.

Generating word list and band count files

```
#wordlist
#uniq -d removes singletons
cat *.txt | sort | tr A-Z a-z | uniq -d | tr '\n' ',' > /
wordlist_allbands.csv

#Count files
for i in *.txt ; do cat $i | sort | uniq -c | /
awk {'print $2" "$1'} > ${i%.txt}.count ; done
```

Generating data matrix

```python
### Got help from a python magician to get this code working
###(Thanks Zach)!

import os
import glob
import pandas as pd
import numpy as np
import csv

X = glob.glob('*.count')

with open('wordlist_allbands.csv', 'rb') as file:
    Wordlist = file.read()

Wordlist = Wordlist.split(',')

lodf = []
for t in X:
    df = pd.read_table(t, header=None, index_col=False, sep=' ')
    df.columns = ['word', 'count']
    lodf.append(df)


print(len(lodf))
print(lodf[0].head())

#df.to_csv("temp.csv", sep='\t')

df = pd.DataFrame()
df['word'] = Wordlist
#print(df.head())

for i in lodf:
    df = df.merge(i, how='left', on='word')

#print(df.head())

colnames = ['word'] + X
df.columns = colnames
df.to_csv('df.csv')
```