Group Project Part I: Data Wrangling

Wranglers

Wednesday, October 7, 2015

## Team Members

* Nisha Balani
* Chanakya Daparthy
* Sri Dakshinya Gudipati
* Sai Aditya Immaneni
* Anusha Vanam
* Keith Williams

# Data Preparation

The data for patent bibliographic data exists in 533 zipped files. The format for the relevant data is pseudo-xml. The following process is used to extract and clean the data:

#### 1. Unzip and Transform Raw ipgbXXXXXXXX\_wkXX.xml Files

Each \*.zip file is iteratively extracted, and, using regular expressions, the .xml transformed, so that each patent is on a single line. These patent per line files are filtered, so that only patents belonging to our company and its competitors are kept. See *Figure 1* for the shell code.

#### 2. Prepare Each Company's Patents for Feature Extraction

The one patent per line text file is split into 4 files: one for the patents belonging to our company, and three for each of our closest competitors. See *Figure 2* for the shell code.

#### 3. Use Regex to Extract Fields and Write .csv

Each of the four one patent per line patent files are processed using a Python script (*Figure 3*). The resulting file is a .csv, where each row is a single patent and each column is a feature of that patent. The features are:  
- Company Name  
- Patent Assignee  
- Year Granted  
- Year Applied  
- Patent Class  
- Patent Number  
- Patent Title  
- Patent Abstract

*Figures 4 & 5* visualize word frequencies found in patent titles and patent abstracts for our company, Medtronic.

# Patent Ratios

The USPTO website was used to query the number of patents per year over the last 10 years for Medtronic and its three closest competitors: Stryker, Boston Scientific, and Abbott. These numbers were compared to the number of patents for each company mined from the patent bibliographic data (*Figures 6, 7, & 8*). Nearly 100% of the reported patents were mined from the bibliographic data.

# Other Relevant Data

* Compare patents utilized vs. non utilized (utilization = products implemented)
* Connect patent(s) with products and products to revenue data
  + Product line information
* Revenue information by product (SEC Filings 10K/10Q)
* Macro economic scenario for medical industry
* Patent classification is alpha numeric- dictionary of these codes and translate them to business information
* Medical eco system: Wearable technology (FitBit, Jawbone, Google Lens etc.)

# Appendix

#### Figure 1

Shell script for extracting, transforming, and filtering patent .xml files.

shopt -s extglob #set shell options  
  
cat >> oppl.txt #create empty text file for appending   
^D  
  
# loop over each zip, extract and transform  
for file in \*.zip  
do  
unzip "$file"  
cat \*.xml | python ../one-patent-per-line.py | grep -i -E "medtronic|covidien|stryker|boston scientfic|abbott" >> ../oppl.txt  
rm \*.!(zip)  
done

#### Figure 2

Shell script for splitting one patent per line file into 4 files, one for each company.

grep -i -E "<assignee>.\*?medtronic.\*?</orgname|<assignee>.\*?covidien.\*?</orgname" oppl.txt > oppl\_medtronic.txt  
grep -i -E "<assignee>.\*?stryker.\*?</orgname" oppl.txt > oppl\_stryker.txt  
grep -i -E "<assignee>.\*?boston scientific.\*?</orgname" oppl\_combined.txt > oppl\_boston\_scientific.txt  
grep -i -E "<assignee>.\*?abbott.\*?</orgname" oppl\_combined.txt > oppl\_abbott.txt

#### Figure 3

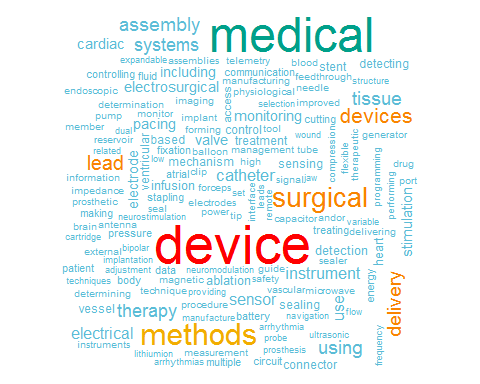
Python script for creating structured .csv.

# Takes as an input a list of .txt files in which each patent is on its own   
# line: an output of one-patent-per-line.py | grep -i "COMPANY" > out.txt  
# Extracts fields for output as a tidy .csv  
  
import sys as sys # for unix pipes  
import re as re # for regular expressions  
import pandas as pd # for structured data tables  
  
def oppl\_to\_df(company, patent\_file):  
 """Takes a wide patent file, and returns a pandas DataFrame  
 Inputs:  
 company: string of company name  
 patent\_file: .txt file where each patent owned by company is on a single line  
 Ouput:  
 pandas DataFrame, where rows are patents, and columns are the features:  
 company name  
 patent assignee  
 year granted  
 year applied  
 patent class  
 patent number  
 patent title  
 patent abstract  
 """  
   
 # initialize fields as a dictionary  
 fields = {'company\_name' : [],  
 'patent\_assignee' : [],  
 'year\_granted' : [],  
 'year\_applied' : [],  
 'patent\_class' : [],  
 'patent\_number' : [],  
 'patent\_title' : [],  
 'patent\_abstract' : []}  
   
 # specify regular expressions  
 assignee\_re = '<assignee>.\*?</orgname>'  
 pub\_ref\_re = '<publication-reference>.\*?</publication-reference>'  
 date\_tags = '<publication-reference>.\*<date>|</date>.\*'  
 id\_tags = '<publication-reference>.\*<doc-number>|</doc-number>.\*'  
 app\_ref\_re = '<application-reference.\*?</application-reference>'  
 class\_re = '</classification-locarno>.\*?</main-classification>'  
 title\_re = '<invention-title.\*?</invention-title>'  
 abstract\_re = '<abstract.\*?</abstract>'  
   
 for line in patent\_file.readlines():  
 # Use regex to find fields,   
 # and append them to appropriate dictionary value  
   
 # attach company name  
 fields['company\_name'].append(company)  
   
 # extract patent assignee  
 assignee = re.search(assignee\_re, line)  
 if assignee:  
 patent\_assignee = re.sub('<assignee>.\*<orgname>|</orgname>',   
 '',   
 assignee.group())  
 fields['patent\_assignee'].append(patent\_assignee)  
 else:  
 fields['patent\_assignee'].append('None')  
   
 # extract year granted and patent number,   
 # both in <publication-reference>  
 pub\_ref = re.search(pub\_ref\_re, line)  
 if pub\_ref:  
 year\_granted = re.sub(date\_tags, '', pub\_ref.group())  
 patent\_number = re.sub(id\_tags, '', pub\_ref.group())  
 fields['year\_granted'].append(year\_granted[0:4])  
 fields['patent\_number'].append(patent\_number)  
 else:  
 fields['year\_granted'].append('None')  
 fields['patent\_number'].append('None')  
   
 # extract year applied  
 app\_ref = re.search(app\_ref\_re, line)  
 if app\_ref:  
 year\_applied = re.sub('<application.\*<date>|</date>.\*',  
 '',  
 app\_ref.group())  
 fields['year\_applied'].append(year\_applied[0:4])  
 else:  
 fields['year\_applied'].append('None')  
   
 # extract patent classification  
 classification = re.search(class\_re, line)  
 if classification:  
 patent\_class = re.sub('</class.\*<main-classification>|</main.\*',  
 '',  
 classification.group())  
 fields['patent\_class'].append(patent\_class)  
 else:  
 fields['patent\_class'].append('None')  
   
 # extract patent title  
 title = re.search(title\_re, line)  
 if title:  
 patent\_title = re.sub('<invention.\*?>|</invention.\*',  
 '',  
 title.group())  
 fields['patent\_title'].append(patent\_title)  
 else:  
 fields['patent\_title'].append('None')  
   
 # extract patent abstract  
 abstract\_field = re.search(abstract\_re, line)  
 if abstract\_field:  
 abstract = re.sub('<abstract.\*<p.\*?>|</abstract>',  
 '',  
 abstract\_field.group())  
 fields['patent\_abstract'].append(abstract)  
 else:  
 fields['patent\_abstract'].append('None')  
   
 # transform dictionary to pandas DataFrame  
 return pd.DataFrame(fields)  
   
def run():  
 # paths to patent files for each company. Each patent is on a single line.  
 oppl\_medtronic = open('../oppl\_medtronic.txt')  
 oppl\_stryker = open('../oppl\_stryker.txt')  
 oppl\_bs = open('../oppl\_boston\_scientific.txt')  
 oppl\_abbott = open('../oppl\_abbott.txt')  
   
 # define lists of companies, and the patent files for feature extraction loop  
 oppl\_files = [oppl\_medtronic, oppl\_stryker, oppl\_bs, oppl\_abbott]  
 companies = ['Medtronic', 'Stryker', 'Boston Scientific', 'Abbott']  
   
 # convert each patent file into a tidy pandas data frame  
 dfs = [oppl\_to\_df(companies[i], oppl\_files[i]) for i in range(len(companies))]  
   
 # concatenate the data frames, and write to .csv  
 df\_combined = pd.concat(dfs, ignore\_index=True)  
 df\_combined.to\_csv('medtronic\_and\_competitor\_patents\_05-15.csv', index=False)  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 run()

#### Figure 4

Visualization of word freqencies in Medtronic patent titles. R code provided.

library(wordcloud) # Package for word clouds  
library(wesanderson) # color palettes  
  
# filter company equal to Medtronic  
med <- patents %>% filter(company\_name == 'Medtronic')  
  
# visualize patent titles  
palette <- wes\_palette("Darjeeling", 5, "discrete")  
palette <- palette[5:1]  
wordcloud(med$patent\_title, max.words=150, colors=palette)



#### Figure 5

Visualization of word frequencies in Medtronic patent abstracts. R code provied.

# visualize patent abstracts  
wordcloud(med$patent\_abstract, max.words=150, colors=palette)

## Warning in wordcloud(med$patent\_abstract, max.words = 150, colors =  
## palette): endline could not be fit on page. It will not be plotted.

## Warning in wordcloud(med$patent\_abstract, max.words = 150, colors =  
## palette): therapy could not be fit on page. It will not be plotted.

## Warning in wordcloud(med$patent\_abstract, max.words = 150, colors =  
## palette): extending could not be fit on page. It will not be plotted.

## Warning in wordcloud(med$patent\_abstract, max.words = 150, colors =  
## palette): implanted could not be fit on page. It will not be plotted.

## Warning in wordcloud(med$patent\_abstract, max.words = 150, colors =  
## palette): positioned could not be fit on page. It will not be plotted.

## Warning in wordcloud(med$patent\_abstract, max.words = 150, colors =  
## palette): electrodes could not be fit on page. It will not be plotted.

## Warning in wordcloud(med$patent\_abstract, max.words = 150, colors =  
## palette): distal could not be fit on page. It will not be plotted.

## Warning in wordcloud(med$patent\_abstract, max.words = 150, colors =  
## palette): longitudinal could not be fit on page. It will not be plotted.

## Warning in wordcloud(med$patent\_abstract, max.words = 150, colors =  
## palette): embodiments could not be fit on page. It will not be plotted.

## Warning in wordcloud(med$patent\_abstract, max.words = 150, colors =  
## palette): portion could not be fit on page. It will not be plotted.

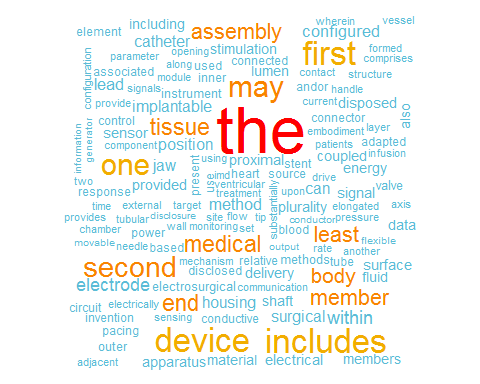
## Warning in wordcloud(med$patent\_abstract, max.words = 150, colors =  
## palette): cardiac could not be fit on page. It will not be plotted.

## Warning in wordcloud(med$patent\_abstract, max.words = 150, colors =  
## palette): patient could not be fit on page. It will not be plotted.

## Warning in wordcloud(med$patent\_abstract, max.words = 150, colors =  
## palette): devices could not be fit on page. It will not be plotted.

## Warning in wordcloud(med$patent\_abstract, max.words = 150, colors =  
## palette): system could not be fit on page. It will not be plotted.

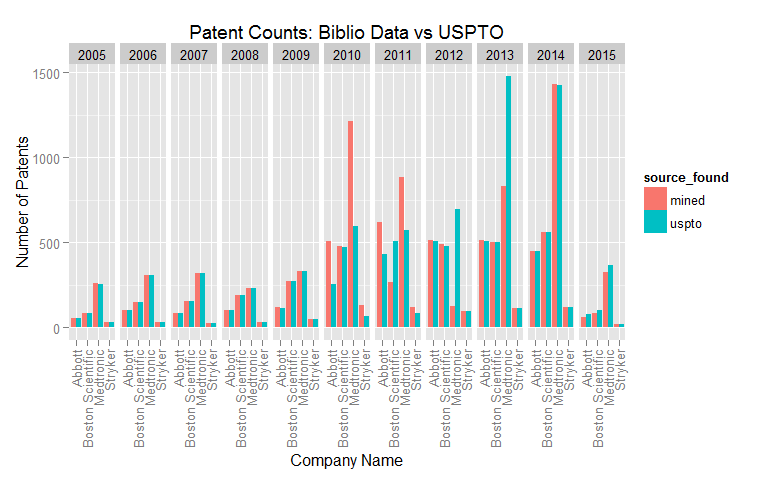
## Warning in wordcloud(med$patent\_abstract, max.words = 150, colors =  
## palette): include could not be fit on page. It will not be plotted.



#### Figure 6

By year comparison of mined patents to patents reported on the USPTO website. R code for creating visualization provided.

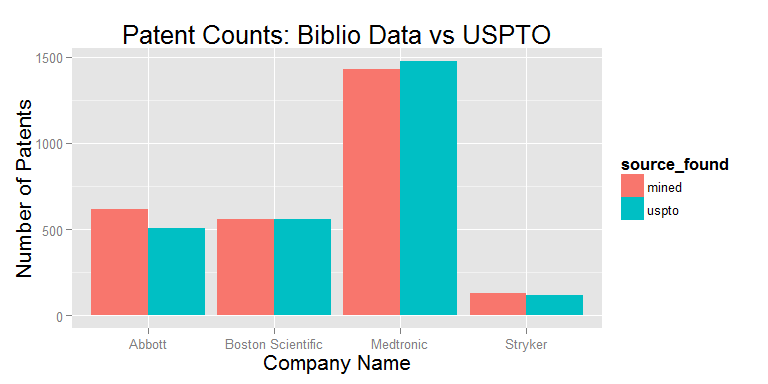
# visualize ratios by year  
plot\_ratios <- ggplot(patent\_ratios, aes(company\_name, count, fill=source\_found)) +  
 geom\_bar(stat='identity', position='dodge') +  
 facet\_grid(.~year\_granted, scales="free\_y") +  
 labs(list(title = "Patent Counts: Biblio Data vs USPTO",  
 x = "Company Name",  
 y = "Number of Patents")) +  
 theme(axis.text.x = element\_text(angle=90, hjust=1, vjust=0.5))  
plot\_ratios



#### Figure 7

Comparison of total mined patents to patents reported on the USPTO website from 2005 to March 2015. R code for creating visualization provided.

# visualize ratios: total by company  
plot\_totals <- ggplot(patent\_ratios, aes(company\_name, count, fill=source\_found)) +  
 geom\_bar(stat='identity', position='dodge') +  
 labs(list(title = "Patent Counts: Biblio Data vs USPTO",  
 x = "Company Name",  
 y = "Number of Patents")) +  
 theme(title = element\_text(size = 16))  
plot\_totals



#### Figure 8

Table of mined *vs.* reported patents.

|  |  |  |  |
| --- | --- | --- | --- |
| company\_name | year\_granted | mined | uspto |
| Abbott | 2005 | 54 | 54 |
| Abbott | 2006 | 104 | 101 |
| Abbott | 2007 | 85 | 85 |
| Abbott | 2008 | 101 | 101 |
| Abbott | 2009 | 117 | 115 |
| Abbott | 2010 | 510 | 253 |
| Abbott | 2011 | 619 | 433 |
| Abbott | 2012 | 515 | 505 |
| Abbott | 2013 | 512 | 508 |
| Abbott | 2014 | 450 | 449 |
| Abbott | 2015 | 61 | 76 |
| Boston Scientific | 2005 | 87 | 86 |
| Boston Scientific | 2006 | 146 | 146 |
| Boston Scientific | 2007 | 156 | 156 |
| Boston Scientific | 2008 | 189 | 192 |
| Boston Scientific | 2009 | 275 | 274 |
| Boston Scientific | 2010 | 478 | 474 |
| Boston Scientific | 2011 | 266 | 507 |
| Boston Scientific | 2012 | 488 | 477 |
| Boston Scientific | 2013 | 504 | 504 |
| Boston Scientific | 2014 | 561 | 561 |
| Boston Scientific | 2015 | 83 | 100 |
| Medtronic | 2005 | 260 | 257 |
| Medtronic | 2006 | 310 | 306 |
| Medtronic | 2007 | 320 | 317 |
| Medtronic | 2008 | 229 | 232 |
| Medtronic | 2009 | 333 | 332 |
| Medtronic | 2010 | 1211 | 599 |
| Medtronic | 2011 | 884 | 571 |
| Medtronic | 2012 | 126 | 698 |
| Medtronic | 2013 | 829 | 1476 |
| Medtronic | 2014 | 1430 | 1423 |
| Medtronic | 2015 | 327 | 367 |
| Stryker | 2005 | 33 | 33 |
| Stryker | 2006 | 30 | 30 |
| Stryker | 2007 | 27 | 27 |
| Stryker | 2008 | 31 | 31 |
| Stryker | 2009 | 48 | 48 |
| Stryker | 2010 | 130 | 65 |
| Stryker | 2011 | 119 | 82 |
| Stryker | 2012 | 96 | 94 |
| Stryker | 2013 | 116 | 116 |
| Stryker | 2014 | 118 | 117 |
| Stryker | 2015 | 17 | 21 |