Type *Markdown* and LaTeX: α^2

Natural Language Processing of Text in Legal Contracts for Structured Database Construction

Project for *Orange Silicon Valley*

Rafael V. Baca, Esq.

SANDBOXING - TOY SET - Contract Pipeline

In [1]:
```
#reset -fs
```

Python package installs

In [1]:
```
# Standard Modules
import pandas as pd
import numpy as np
import re
```

In [2]:
```
# Misc Modules
import os
import sys
import getopt
```

In [3]:
```
# Package for converting Nation Names to estb Country Codes
# https://stackoverflow.com/questions/16253060/how-to-convert-country-names-
from textblob import TextBlob
from calendar import month_name
from collections import Counter
import sys

#reload(sys)
#sys.setdefaultencoding('utf8')
```

Estab local file paths for PDFs

In [4]:
```
for p in sys.path:
    print(p)
```

```
/Users/x/anaconda/envs/py27/lib/python27.zip
/Users/x/anaconda/envs/py27/lib/python2.7
/Users/x/anaconda/envs/py27/lib/python2.7/plat-darwin
/Users/x/anaconda/envs/py27/lib/python2.7/plat-mac
/Users/x/anaconda/envs/py27/lib/python2.7/plat-mac/lib-scriptpackages
/Users/x/anaconda/envs/py27/lib/python2.7/lib-tk
/Users/x/anaconda/envs/py27/lib/python2.7/lib-old
/Users/x/anaconda/envs/py27/lib/python2.7/lib-dynload
/Users/x/anaconda/envs/py27/lib/python2.7/site-packages
/Users/x/anaconda/envs/py27/lib/python2.7/site-packages/IPython/extension
s
/Users/x/.ipython
```

In [5]:
```
pwd
```

Out[5]:
```
u'/Users/x/Desktop/NLP/Project_Sandbox'
```

In [6]:
```
# path = '/Users/x/Desktop/NLP/Project_Sandbox/Baseline_MasterContract.pdf'
# df = open(path, 'rb')
```

Sandboxing PDFminer -

Extracting string text values - individual files

In [7]:
```
from pdfminer.pdfparser import PDFParser

from pdfminer.pdfdocument import PDFDocument

from pdfminer.layout import LAParams

from pdfminer.pdfpage import PDFPage

from pdfminer.pdfpage import PDFTextExtractionNotAllowed

from pdfminer.pdfinterp import PDFResourceManager

from pdfminer.pdfinterp import PDFPageInterpreter

from pdfminer.pdfdevice import PDFDevice

from pdfminer.converter import TextConverter


#This accounts for various discrepencies with IO package
# (as different versions of PDFMiner & .six important differently)
import io
from io import BytesIO
from io import StringIO
from cStringIO import StringIO
```

In [8]:
```
# THIS FUNCTION USES PDF MINER TO OCR RECEIVED PDFs
def convert_pdf_to_txt(path):

    rsrcmgr = PDFResourceManager()
    retstr = StringIO()
    codec = 'utf-8'
    laparams = LAParams()
    device = TextConverter(rsrcmgr, retstr, codec=codec, laparams=laparams)

    #device = TextConverter(rsrcmgr= PDFResourceManager(), retstr= BytesIO(),
    fp = file(path, 'rb')
    interpreter = PDFPageInterpreter(rsrcmgr, device)
    password = ""
    maxpages = 0
    caching = True
    pagenos=set()

    for page in PDFPage.get_pages(fp, pagenos, maxpages=maxpages, password=p
        interpreter.process_page(page)

    text = retstr.getvalue()


    fp.close()
    device.close()
    retstr.close()
    return text
```

Company Name Extraction

Tentative Flow/Pseudo Code:

OPTION (1) REG EX to CSV "Company Name" Column --

1. Step 1--

PARAGRAPH REGEX: Look to the principle place of business locations of each of the contracts using the terminology: "company incorporated" as the regular expression anchor such that the words at before and to the beginning of the sentence should have the company name.

Additionally, look to anchor a regular expression using a zip code in the same way as hte above described "company incorporated". US, FR, Spain, Germany have Zip codes with 5 numbers. Other European countries have 4 numbers or no zip, England uses an alphanumeric code. See https://en.wikipedia.org/wiki/List_of_postal_codes (https://en.wikipedia.org/wiki/List_of_postal_codes)

1. Step 2 --

CHUNKING EACH PARAGRAPH FROM STEP 1: It will thus tag noun words into PERSON vs ORGANIZATION using the Stanford tagging over the NLTK :

BUILD LISTS OF TAGS: Programatically make a list [] for tags of greater importance. Ex. ORG-Organization = [], PER - Person = [], LOCA[] - location.

1. Step 3 --

FREQUENCY COUNT OF ORGANIZATION TAGS: Each Chunk has a key & value (= word & tag). From the ORG list run a frequency count. Highest freq ct 1 should == Orange; All other frequency counts should be all contract party company tagged as ORG.

GENERATE KEY/COMPANY LIST for ORGS: List should have all company.

1. Step 4 --

PROGRAMATICALLY MATCH COMPANY LIST WITH SIC VALUES FOR COMPANY WITH UK COMPANY HOUSE API: Assigned Numeric Values should have the textual industry description for each company listed in the UK Company House API.

OPTION (2) TFIDF to CSV "Company Name" Column --

1. Step A --

PARAGRAPH REGEX: Repeat Step 1 from Option (1) above to obtain a list of paragraphs that have embedded company names for each contract document.

1. Step B --

Apply Sklearn Term Frequency, Inverse Document Fequency (tf-idf) to apply topic modeling to each entire contract. Because the corpus size is small, LDA may not be as accurate as stated in the literature.

1. Step C --

FREQUENCY COUNT TO SELECT TOP TOPICS: Apply Legal Expertise to hand-lable industry for each contract topic in the frequency count.

In [ ]:

Reading In Batch of Sample PDF Contracts

Batch of Contracts: Rough, non-listcomprehension way to extract each .pdf document in folder

In [9]:
```
import glob

text_lst = []
for filename in glob.glob('ContractSamples/*.pdf'):
    try:
        text_lst.append(convert_pdf_to_txt(filename).decode('utf-8'))
        print("Successful read file: ", filename)
    except:
        print("Can't read file: ", filename)
```

```
('Successful read file: ', 'ContractSamples/AT&T Master Agreement.pdf')
('Successful read file: ', 'ContractSamples/Orange Router Service Agreeme
nt.pdf')
('Successful read file: ', 'ContractSamples/Orange S.A._Google Search for
Docspdf.pdf')
('Successful read file: ', 'ContractSamples/Orange VPN Service Agreement.
pdf')
('Successful read file: ', 'ContractSamples/VerizonMSA.pdf')
```

In [10]:
```
len(text_lst)
```

Out[10]: 5

In [11]:
```
text_lst[0]
```

Out[11]:
```
u'9/27/2017\nEX-10.8 4 dex108.htm AT&T MASTER AGREEMENT\n\nAT&T Master Ag
reement\n\nExhibit\xa010.8\n\n\xa0\nCUSTOMER\xa0(\u201cCustomer\u201d)\nE
yeblaster,\xa0Inc\n\nCUSTOMER\xa0Address\n135\xa0West\xa018th\xa0Stree
t,\xa05th\xa0Floor\nNew\xa0York\nNY\xa010011\xa0USA\nCUSTOMER\xa0Contact
\nName:\xa0Nir\xa0Yaron\nTitle:\xa0COo\nTelephone:\xa0646\xad202\xad1334
\nFax:\nEmail:\xa0nir.yaron@eyeblaster.com\n\nAT&T\xa0MASTER\xa0AGREEMENT
\n\nMA\xa0Reference\xa0No.\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa0\xa
0\xa0\n\n\xa0\xa0 AT&T\xa0(\u201cAT&T\u201d)\n\nAT&T\xa0Corp.\xa0(If\xa0I
nternational,\xa0insert\xa0AT&T\xa0Legal\nEntity\xa0Signing\xa0Name)\n\n
\xa0\xa0\n\xa0\xa0 AT&T\xa0Address\n\n55\xa0Corporate\xa0Drive\xa0Bridgew
ater,\xa0NJ\xa008807\xa0(If\nInternational,\xa0insert\xa0AT&T\xa0Legal\xa
0Entity\xa0Information)\n\n\xa0\xa0\n\xa0\xa0\n\xa0\xa0 AT&T\xa0Contact\n
\xa0\xa0 Master\xa0Agreement\xa0Support\xa0Team\n\xa0\xa0 Email:\xa0mast@
att.com\n\xa0\xa0\n\xa0\xa0\n\xa0\xa0\n\nThis\xa0Agreement\xa0consists\xa
0of\xa0this\xa0Master\xa0Agreement\xa0and\xa0all\xa0schedules,\xa0exhibit
s\xa0and\xa0service\xa0order\xa0attachments\xa0(\u201cAttachments\u201d)
\xa0appended\nhereto\xa0or\xa0subsequently\xa0signed\xa0by\xa0the\xa0part
ies,\xa0and\xa0that\xa0reference\xa0this\xa0Master\xa0Agreement\xa0(colle
ctively,\xa0this\xa0\u201cAgreement\u201d).\xa0In\xa0the\xa0event\xa0of\x
```

Cleaning Function

In [12]:
```
def clean_file(text0):
    text0 = re.sub(r'\\n', "-", text0)  #this
    text0 = re.sub(r"\\xa0|\\xc2", " ", text0)  #this
    text0 = re.sub(r"\\", " ", text0)  # this
    return text0
```

File 1 - AT&T

In [13]:
```
import sys

reload(sys)
sys.setdefaultencoding('utf-8')
text0 = text_lst[0].decode('utf-8')#this
text0 = clean_file(text0)
text0
```

In [14]:
```
from textblob import TextBlob
sentence = TextBlob(text0).sentences
```

In [15]:
```
# sentence[0]
```

In [16]:
```
sentence0 = re.sub(r'[^\w+!\.,:&@\/]', " ", str(sentence[0]))
sentence0
```

Out[16]:
```
'9/27/2017 EX 10.8 4 dex108.htm AT&T MASTER AGREEMENT  AT&T Master Agreem
ent  Exhibit  10.8     CUSTOMER      Customer     Eyeblaster,  Inc  CUSTO
MER  Address 135  West  18th  Street,  5th  Floor New  York NY  10011  US
A CUSTOMER  Contact Name:  Nir  Yaron Title:  COo Telephone:  646  202  1
334 Fax: Email:  nir.yaron@eyeblaster.com  AT&T  MASTER  AGREEMENT  MA  R
eference  No.'
```

In [17]:
```
customer_email = re.findall(r'Email\:.*(?:\.com)', sentence0)
customer_phone = re.findall(r'Telephone\:.*(?:[\d+])', sentence0)
customer_address = re.findall(r'Address.*?(?=[A-Z]{3})', sentence0)
customer_name = re.findall(r'Name\:.*?(?=\w+:)', sentence0)
```

In [18]:
```
customer_email
```

Out[18]:
```
['Email:  nir.yaron@eyeblaster.com']
```

In [19]:
```
customer_address
```

Out[19]:
```
['Address 135  West  18th  Street,  5th  Floor New  York NY  10011  ']
```

In [20]:
```
customer_name
```

Out[20]:
```
['Name:  Nir  Yaron ']
```

In [21]:
```
customer_phone
```

Out[21]:
```
['Telephone:  646  202  1334']
```

In [22]:
```
#Will need to import nltk
# plus Stanford package will be downloaded as well
#it will be important to insert that package along where the source data res

#import nltk
#nltk.download()
```

In [23]:
```
from __future__ import print_function
import nltk
nltk.__version__
```

Out[23]:
```
'3.2.5'
```

In [24]:
```
if os.system('java -version') == 32512: # Value for ‘command not found’
    os.system('brew doctor')
    os.system('brew update')
    os.system('brew install cask')
    os.system('brew cask install java')
```

In [25]:
```
import nltk
from nltk.tag.stanford import StanfordNERTagger
```

In [26]:
```
base_path = './stanford-ner-2014-06-16'
st = StanfordNERTagger(base_path+'/classifiers/all.3class.distsim.crf.ser.gz
                       base_path+'/stanford-ner.jar')
```

In [27]:
```
st_tagged = st.tag(nltk.word_tokenize(sentence0))
org = []
per = []
for word, tag in st_tagged:
    if tag == 'ORGANIZATION':
        org.append(word)
    if tag == 'PERSON':
        per.append(word)
most_likely_org = " ".join([i for i,v in Counter(org).most_common(3)]).repla
```

In [28]:
```
most_likely_org
```

Out[28]:
```
u'& AT T'
```

File 5 - Verizon

In [29]:
```
from textblob import Word
text5=text_lst[4].encode('unicode-escape') # SAVE this
text5 = clean_file(text5)
text5
```

Out[29]:
```
'MASTER SERVICE AGREEMENT--BY AND BETWEEN--VERIZON SERVICES CORP., on beh
alf of the Verizon Companies--listed on Service Attachments herein, AND--
STATE OF OHIO, OFFICE OF INFORMATION TECHNOLOGY--THIS MASTER SERVICE AGRE
EMENT (the "Agreement"), is by and between Verizon-Services Corp. ("Vendo
r"), on behalf of all Verizon companies listed on all Service Attachments
-herein, with an address of 4020 Winnetka Avenue, Rolling Meadows IL 6000
8, and the State of-Ohio, Office of-Information Technology ("the State" o
r "OIT"), having its principal place of-business at 1320 Arthur E. Adams
 Drive, 3rdFloor, Columbus, OH 43221 Uointlyreferred hereto-as the "Parti
es") and is effective as of the date signed by the State. State Agencies,
Boards,-Commissions and Cooperative Purchasing members (including the Off
ice of-Information-Technology, collectively referred to as "Subscribing E
ntity") are eligible to use this Agreement.--VZ Case No.: 2004-292478-C>!
\'l1traclNe/.: 21)()!-/5J588--rZ Generaled By: JJC- J2// 2/2f!1i5--VZApfi
nJl\'ed10 /-urm. EJO12!J2iW-- x0c1--2-3--4-5--6-7--8--Table of Contents-
-...--,--oo--Services-Subscribing Entities-Cooperative Purchasing Members
...-Headings-Standard State Terms and Conditions--Service Specific Terms
 and Conditions-Relationship of Parties-Non-Exclusivity-Severability-Surv
ival--General Information-1.1-1.2-1.3-1.4-1.5-1.6 VendorAddedTermsandCond
```

In [30]:
```
#import sys

#reload(sys)
#sys.setdefaultencoding('utf-8')
#text5 = text_lst[4].decode('utf-8')#this
#text5 = clean_file(text0)
#text5
```

In [31]:
```
sentence5 = TextBlob(text5).sentences
```

In [32]:
```
text5 = str(sentence5[0:2])
```

In [33]:
```
text5
```

Out[33]:
```
'[Sentence("MASTER SERVICE AGREEMENT--BY AND BETWEEN--VERIZON SERVICES CO
RP., on behalf of the Verizon Companies--listed on Service Attachments he
rein, AND--STATE OF OHIO, OFFICE OF INFORMATION TECHNOLOGY--THIS MASTER S
ERVICE AGREEMENT (the "Agreement"), is by and between Verizon-Services Co
rp. ("Vendor"), on behalf of all Verizon companies listed on all Service
 Attachments-herein, with an address of 4020 Winnetka Avenue, Rolling Mea
dows IL 60008, and the State of-Ohio, Office of-Information Technology
 ("the State" or "OIT"), having its principal place of-business at 1320 A
rthur E. Adams Drive, 3rdFloor, Columbus, OH 43221 Uointlyreferred hereto
-as the "Parties") and is effective as of the date signed by the Stat
e."), Sentence("State Agencies, Boards,-Commissions and Cooperative Purch
asing members (including the Office of-Information-Technology, collective
ly referred to as "Subscribing Entity") are eligible to use this Agreemen
t.--VZ Case No.")]'
```

In [34]:
```
text_5_paragraph = re.findall(r'.*?(?:[A-Z][A-Z]\s\d{5})', text5)
# text_4_paragraph = re.findall(r'--.*?@\w+(?:\.com)', str(text_0_paragraph)
```

In [35]:
```
text_5_paragraph
```

Out[35]:
```
['[Sentence("MASTER SERVICE AGREEMENT--BY AND BETWEEN--VERIZON SERVICES C
ORP., on behalf of the Verizon Companies--listed on Service Attachments h
erein, AND--STATE OF OHIO, OFFICE OF INFORMATION TECHNOLOGY--THIS MASTER
 SERVICE AGREEMENT (the "Agreement"), is by and between Verizon-Services
 Corp. ("Vendor"), on behalf of all Verizon companies listed on all Servi
ce Attachments-herein, with an address of 4020 Winnetka Avenue, Rolling M
eadows IL 60008',
 ', and the State of-Ohio, Office of-Information Technology ("the State"
 or "OIT"), having its principal place of-business at 1320 Arthur E. Adam
s Drive, 3rdFloor, Columbus, OH 43221']
```

In [36]:
```
contractor_org = re.findall(r'State\s*of.*?[A-Z]\w+', text5)
```

In [37]:
```
contractor_org
```

Out[37]:
```
['State of-Ohio']
```

In [38]:
```
# tags = st.tag(nltk.word_tokenize(text_5_paragraph[0]))
tags = st.tag(nltk.word_tokenize(text_5_paragraph[0]))
ORG_tags = [Word(i).lemmatize() for i,v in tags if v == 'ORGANIZATION']


per = []
org = []
for word, tag in tags:
    if tag == 'ORGANIZATION':
        org.append(word)
    if tag == 'PERSON':
        per.append(word)
most_likely_org = " ".join([i for i,v in Counter(org).most_common(3)]).repla
```

In [39]:
```
most_likely_org
```

Out[39]:
```
u'Verizon Verizon-Services Corp.'
```

WEBSCRAPE SIC

In [40]:
```
contractor_name = [name for name in Counter(ORG_tags).keys()]
```

In [41]:
```
from textblob import TextBlob
sentence = TextBlob(" ".join(contractor_name))
sentence
```

Out[41]:
```
TextBlob("Verizon-Services Corp. VERIZON Verizon Companies SERVICES COR
P.")
```

In [42]:
```
customer_name = " ".join(re.findall(r'[A-Z][A-Z]+', str(sentence)))
```

In [43]:
```
customer_name = customer_name.replace(" ","+")
```

In [44]:
```
# customer_name =
```

In [45]:
```
# orgName = re.findall(r'(?<=u\'orgName\'\:\su\').*?(?=\',)', str(comp_info)
# print(orgName)
```

In [46]:
```
search_prefix = 'https://beta.companieshouse.gov.uk/search/companies?q='
```

In [47]:
```
full_search_link = search_prefix + customer_name
```

In [48]:
```
full_search_link
```

Out[48]:
```
'https://beta.companieshouse.gov.uk/search/companies?q=VERIZON+SERVICES+C
ORP'
```

In [49]:
```
import urllib
import urllib2
from bs4 import BeautifulSoup

company_pages = []
#Query the website and return the html to the variable 'page'
page = urllib2.urlopen(full_search_link)

#Parse the html in the 'page' variable, and store it in Beautiful Soup forma
soup = BeautifulSoup(page)

#get all links appear on page
all_links = soup.find_all('a', title="View company")
prefix = "https://beta.companieshouse.gov.uk"

for j,link in enumerate(all_links):
#     link.get("href")
    company_pages.append(prefix + link.get("href"))
```

In [50]:
```
company_pages
```

Out[50]:
```
[u'https://beta.companieshouse.gov.uk/company/08524398',
 u'https://beta.companieshouse.gov.uk/company/08500133',
 u'https://beta.companieshouse.gov.uk/company/09748527',
 u'https://beta.companieshouse.gov.uk/company/05932446',
 u'https://beta.companieshouse.gov.uk/company/03035660',
 u'https://beta.companieshouse.gov.uk/company/04489658',
 u'https://beta.companieshouse.gov.uk/company/05763520',
 u'https://beta.companieshouse.gov.uk/company/04636580',
 u'https://beta.companieshouse.gov.uk/company/02776038',
 u'https://beta.companieshouse.gov.uk/company/10337299',
 u'https://beta.companieshouse.gov.uk/company/10337292',
 u'https://beta.companieshouse.gov.uk/company/09329368',
 u'https://beta.companieshouse.gov.uk/company/OC363285',
 u'https://beta.companieshouse.gov.uk/company/09449064',
 u'https://beta.companieshouse.gov.uk/company/09959418',
 u'https://beta.companieshouse.gov.uk/company/08878804',
 u'https://beta.companieshouse.gov.uk/company/05545947',
 u'https://beta.companieshouse.gov.uk/company/04232904',
 u'https://beta.companieshouse.gov.uk/company/04818650',
 u'https://beta.companieshouse.gov.uk/company/BR008923']
```

In [51]:
```
def multiple_pages_data_extraction(company_pages):
    sic = []
    description = []
    company_name = []
    for company in company_pages:
        #extract page contain SIC number
        page = urllib2.urlopen(company)
        soup = BeautifulSoup(page)
        SIC = soup.find_all('span', id="sic0")

        #extract sic number
        s = re.findall(r'\d\d\d+', str(SIC))
        s = re.findall(r'\w+', str(s))
        s = " ".join(s)
        sic.append(s)

        #extract description
        d = re.findall(r'(?<=\d{5})[^\d]+(?=\\n)', str(SIC))
        d = re.findall(r'\w+', str(d))
        d = " ".join(d)
        description.append(d)

        #extract company name
        name = soup.find_all('p', id="company-name")
        c = re.findall(r'(?<=\>).*(?=\<)', str(name))
        c = re.findall(r'\w+', str(c))
        c = " ".join(c)
        company_name.append(c)

    df = pd.DataFrame({'Company Name': company_name, 'Description': descript
    return df
```

In [52]:
```
df = multiple_pages_data_extraction(company_pages)
df
```

Out[52]:

	Company Name	Description	SIC #
0	VERIZON DIGITAL MEDIA SERVICES UK LIMITED	Other telecommunications activities	61900
1	ODUENO FACILITY SERVICES LTD	General cleaning of buildings	81210
2	HS RAGHUVANSHI LTD	Information technology consultancy activities	62020
3	05932446 LIMITED		7482
4	SECURENETT SECURITY SYSTEMS LTD	Security systems service activities	80200
5	VERIZON EUROPEAN HOLDINGS LIMITED	Other telecommunications activities	61900
6	VERIZON FINANCING	Other telecommunications activities	61900
7	VERIZON INTERNATIONAL LIMITED	Other information technology service activities	62090
8	VERIZON UK LIMITED	Wired telecommunications activities	61100
9	VERIZON UK FINANCING LIMITED	Other service activities not elsewhere classified	96090
10	VERIZON UK HOLDING LIMITED	Other service activities not elsewhere classified	96090
11	MISBHA COMMUNICATIONS LIMITED	Information technology consultancy activities	62020
12	ARRINS LLP		
13	SECURENETT SECURITY INSTALLATIONS LTD	Security systems service activities	80200
14	RG MOBILE LTD		
15	AMEDEO SERVICES UK LIMITED	Other business support service activities not ...	82990
16	CORP SERVICES LIMITED	Development of building projects	41100
17	DYNAMIC CORP PERSONNEL LTD	Other activities of employment placement agencies	78109
18	HUTTON PROPERTY MANAGEMENT SERVICES LIMITED	Other business support service activities not ...	82990
19	INVENERGY SERVICES INTERNATIONAL CORP		

In [53]:
```
company_information = df.values[0]
```

In [54]:
```
company_information
```

Out[54]:
```
array(['VERIZON DIGITAL MEDIA SERVICES UK LIMITED',
       'Other telecommunications activities', '61900'], dtype=object)
```

In [55]:
```
writer = pd.ExcelWriter("company_info.xlsx")
df.to_excel(writer,'company_info')
writer.save()
```

In [ ]:
```

```

TOPIC MODELING - tf-idf

In [56]:
```
#! pip install sklearn
#! pip install scipy
```

In [57]:
```
import sklearn
from sklearn.feature_extraction.text import CountVectorizer
```

In [ ]:
```

```

In [58]:
```
vectorizer = CountVectorizer(max_features=1000,
                             max_df=0.95,
                             min_df=2,
                             stop_words='english')
```

In [77]:
```
test5 = text5.split()
```

In [78]:
```
vectorized = vectorizer.fit_transform(test5)
```

In [ ]:
```

```

In [83]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import NMF
```

In [80]:
```
TFIDvectorizer = TfidfVectorizer(max_features=1000,
                             max_df=0.95,
                             min_df=2,
                             stop_words='english')
```

In [81]:
```
TFIDized = TFIDvectorizer.fit_transform(test5)
```

In [84]:
```
model = NMF(init="nndsvd",
            n_components=4,
            max_iter=200)
```

In [90]:
```
W = model.fit_transform(TFIDized)
H = model.components_
```

In [87]:
```
terms = [""] * len(vectorizer.vocabulary_)
for term in vectorizer.vocabulary_.keys():
    terms[vectorizer.vocabulary_[term]] = term
```

In [89]:
```
terms[-5:]
```

Out[89]:
```
[u'service', u'services', u'state', u'technology', u'verizon']
```

In [113]:
```
import sys
# reload(sys)
# sys.setdefaultencoding("utf-8")
sys.stdout
```

Out[113]:
```
<open file '<stdout>', mode 'w' at 0x10d3e6150>
```

In [117]:
```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

In [131]:
```

for topic_index in range(H.shape[0]):
    top_indices = np.argsort(H[topic_index,:])[::-1][0:10]
    term_ranking = [terms[i] for i in top_indices]
#     dfs = pd.concat(df, axis=0)
    print("Topic ",topic_index,": ", ", ".join(term_ranking))
```

In [146]:
```
df = pd.DataFrame()
for topic_index in range(H.shape[0]):
    top_indices = np.argsort(H[topic_index,:])[::-1][0:10]
    df['Topic ' + str(topic_index)] = [terms[i] for i in top_indices]
```

In [145]:
```
df.T
```

Out[145]:

	0	1	2	3	4	5	6	7
Topic 0	state	sentence	master	information	technology	companies	listed	office
Topic 1	service	companies	listed	master	sentence	verizon	technology	state
Topic 2	agreement	technology	information	office	listed	companies	master	sentence
Topic 3	verizon	services	technology	information	office	master	sentence	companies

In [124]:
```
", ".join(term_ranking)
```

Out[124]:
```
u'verizon, services, technology, information, office, master, sentence, c
ompanies, listed, behalf'
```

In [ ]:
```
writer = pd.ExcelWriter("contract_topics.xlsx")
df.to_excel(writer,'contract_topics')
writer.save()
```


LIONEL HUTZ
ATTORNEY AT LAW
AS SEEN ON TV
KLondike 5-LAWW
CLOGGING OUR COURTS SINCE 1976

In [ ]: