Job Fiction - Indexing Jobs Data to Build a Training Model

Objective

Objective of this notebook is to build a training model based on JOBFICTION database, a collection of job posts, job titles, company, location, job post URL acquired from Indeed Web Services API. Using the training model, we will be able to predict right job title based on the job descriptions passed to the model. Output from the training model would include - a corpus based on vector space model, key words and phrases, skill identifiers, predicted job titles and corresponding scores. All the results will be persisted and updated with the new jobs being collected.

Based on the input from job seekers i.e. job descriptions submitted we will able to determine

- Job titles closest to the job description or keywords submitted (based on the weights associated)
- Recommended job posts
- Keywords to search for the right job posts

The first part of this notebook will explore how jobs in the JOBFICTION database can be classified.

Why do we have to classify the job posts?

A `truck driver` job post is way different from a `database administrator` job post. With the help of clustering algorithms we categorize similar jobs into same cluster based purely on the job description. Similar to movie genres this classifier is expected to create job categories based on similarity of job descriptions. We can then study the job titles under the same cluster to see how true clusters. Since there is no training data set available we resort to unsupervised clustering and the challenge is to define the number of clusters.

We focus only on the data related job posts i.e. job posts with the word "data" in either job title or job description.

Approach

- Export job descriptions, job title, company and job id from JOBFICTION database
- Remove stop words
- Tokenize and stem each job description
- Transforming the corpus into vector space using tf-idf
- Clustering the documents using the k-means algorithm
- Plot the clusters
- Using multidimensional scaling to reduce dimensionality within the corpus (LSI)
- Topic modeling using Latent Dirichlet Allocation (LDA)
- Named entity recognition against occupation skills and title taxonomies to identify skills

(Future Work)

- Hierarchical clustering on the corpus using Ward clustering (http://en.wikipedia.org/wiki/Ward%27s_method)
- Plot the clusters with hierarchial clustering

Imports

In [1]:

```
%matplotlib inline

from nltk.tokenize import RegexpTokenizer
from nltk.stem.porter import PorterStemmer
from nltk.stem.snowball import SnowballStemmer
from stop_words import get_stop_words
from nltk.corpus import stopwords
from gensim import corpora, models, similarities
from sklearn.cluster import KMeans, MiniBatchKMeans
from collections import Counter
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from wordcloud import WordCloud
import logging
import random
import gensim
import nltk
import re
import os
```

In [2]:

```
logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s', level=lo
gging.INFO)
```

Configuration

In [3]:

```
DATA_DIR = os.path.join("/home", "rt", "wrk", "jobs", "data")
MODEL_DIR = os.path.join("/home", "rt", "wrk", "jobs", "models")
```

1. Export data from JOBFICTION database

Let's extract jobs from JOBFICTION database

In the jobs table, job description is an array of sentences. In order to export job description, this mongo javasript will be run to combine array elements as a string. For traceback we will add __id field to every record.

In [7]:

```
%%writefile export_jobs_with_title.js
db.jobs.find({"summary": /data/}, { _id: 1, jobtitle: 1, company: 1, summary: 1}
).forEach( function (x)
    {
        var jobdesc = '';
        var s = ''
        x.summary.forEach( function (y) {
            s = y.replace(new RegExp('\r?\n','g'), ' ').replace(new RegExp('[|]'
,'g'), '');
            jobdesc += s + ' ';
        });
        print(x._id + "|" + x.jobtitle + "|" + x.company + "|" + jobdesc);
    });
```

```
Overwriting export_jobs_with_title.js
```

In [4]:

```
!mkdir ./data ./models
```

```
mkdir: cannot create directory ‘./data’: File exists
```

Run export script to dump data to text file

In [8]:

```
!time mongo JOBFICTION --quiet export_jobs_with_title.js > ./data/export_jobs_w_
title.txt
```

```
real    4m54.169s
user    0m46.439s
sys     0m3.300s
```

In [9]:

```
!wc -l ./data/export_jobs_w_title.txt
!head -1 ./data/export_jobs_w_title.txt
```

```
144554 ./data/export_jobs_w_title.txt
indeed_6ed966da9f33ffc1|Associate|Potbelly Sandwich Shop|Presidentia
l Towers!!!!!! A Potbelly Associateâs job is to make our customers r
eally happy. Since they are the primary point of customer contact, i
t is up to them to provide our customers and excellent experience by
providing fast, friendly and efficient service and by delivering a q
uality and consistent product every time, in a clean and inviting en
vironment. Essential ï§ Demonstrates and reinforces Potbellyâs Behav
iors and Valuesâ Integrity, Food Loving, Teamwork, Accountability, P
ositive Energy, Coaching, Delivering Results through Execution, Buil
ding and Inspiring Teams, Creating Potbelly âFansâ-- through all int
eractions. ï§ Ability to discuss Potbelly history with others. ï§ Pr
epare quality finished products (sandwiches, salads, soups, cookies,
```

ice cream, etc.) efficiently per Potbelly recipe manual standards. ï § Comply with health and safety standards for food, cleanliness and safety of shop. ï§ Maintain personal hygiene standards, including we aring clean Potbelly uniform. ï§ Comply with established food safety requirements and practices. ï§ Comply with shop security and safety standards. ï§ Be speedy and accurate in fulfilling orders. ï§ Handle raw and finished waste according to established procedures. ï§ Make customers really happy. ï§ Engage in friendly conversation with cust omers in line. ï§ Act with a sense of urgency toward all customers i n the shop. Other Key Functions ï§ Restock food line, chips and cool er. ï§ ï§ Work multiple stations (load, dress, shakes, cash, prep, f ront) as directed by Manager. ï§ Deliver catering orders as detailed in the Catering Driver and Delivery Agreement. ï§ Clean tables, coun ters, floors, bathrooms, kitchen and utensils; take out trash. ï§ Op erate cash register: handle, balance and follow all cash handling pr ocedures. ï§ Effectively handle customer complaints/issues. ï§ Take catering and delivery orders over the phone. ï§ PHYSICAL FUNCTIONS ï § Ability to stand/walk a minimum of 3 hours or as needed. ï§ Must b e able to exert well-paced and frequent mobility for periods of up t o 3 hours or as needed. ï§ Be able to lift up to 10 pounds frequentl y. ï§ Will frequently reach, feel, bend, stoop, carry, finely manipu late and key in data. ï§ Able to work in both warm and cool environm ents, indoors (95%) and outdoors (5%). ï§ Must be able to tolerate h igher levels of noise from music, customer and employee traffic. ï§ Must be able to tolerate potential allergens: peanut products, egg, dairy, gluten, soy, seafood and shellfish. EXPERIENCE, EDUCATION AND BEHAVIORS ï§ Must represent Potbelly Advantage and Our Values. ï§ Mu st be at least 16 years of age ï§ For Illinois employees, all employ ees are required to become food safety certified within 30 days of e mployment. Failure to do so will result in termination of employment . ï§ Must be friendly and customer service-oriented. ï§ Strong verba l communication skills. ï§ Must possess neat and clean hygiene. ï§ A bility to handle a knife confidently. ï§ Must be able to work in a f ast-paced environment and have a sense ofï§ Ability to work as a tea m-player. ï· Ability to comprehend and communicate in English via ve rbal and written communication, such that employee can perform his o r her job responsibilities. ï§ Must demonstrate leadership behaviors and values that align with Potbelly urgency. Potbelly.Com/Careers Jo b Type: Part-time Local candidates only: Chicago, IL 60661 Required education: High school or equivalent

2. Create training data set

We will export random 10K job descriptions as training data set. We will use unsupervised clustering to see how similar job descriptions are. based on clusters we can do topic modeling with LDA for each cluster. We can keep updating the model with new job posts.

Below sort to be optimized by randomized only job ids instead of entire text.

In [10]:

```
!time sort -t'|' -k1 -R ./data/export_jobs_w_title.txt | head -10000 > ./data/tr
ain_w_complete_text.txt
```

```
sort: write failed: standard output: Broken pipe
sort: write error

real    8m32.986s
user    8m30.838s
sys     0m2.053s
```

In [11]:

```
!time awk -F'|' 'BEGIN{OFS="|"}{print $1, $2, $3}'
./data/train_w_complete_text.txt > ./data/train_labels.txt
!time awk -F'|' 'BEGIN{OFS="|"}{print $4}' ./data/train_w_complete_text.txt > ./
data/train.txt
```

```
real    0m0.570s
user    0m0.068s
sys     0m0.021s

real    0m2.101s
user    0m1.167s
sys     0m0.125s
```

In [12]:

```
!head ./data/train_labels.txt
```

```
indeed_6d13e1749c444e23|Financial Examiner (EL)|GA Dept of Banking &
Finance
indeed_6d16914061219ee4|Analytics Payer/Provider Healthcare Analytic
s Manager|PRICE WATERHOUSE COOPERS
indeed_50c9ebbfb19f9ed7|Aircraft Maintenance Analyst|Ronkonkoma, NY
indeed_6d1fbfcd14cf79e9|Operations Center Representative - All Shift
s|Ascent LLC.
indeed_9a61d5c6de9dec4b|Administrator, Payroll|Community Action Proj
ect
indeed_53c5e81c18aa4202|Project Coordinator/Data Analyst|The Fund fo
r Public Health in New York, Inc.
indeed_bf4b755eadef6b10|Plant Manager|IEC Holden Inc.
indeed_e5ee1725b888eeb0|IT Infrastructure & Security Manager|Collibr
a
indeed_08b4c32dcb730ba2|Material Control Specialist l|PRIMUS
indeed_3aede0ed8048b044|Licensed Financial Advisor|Scient Federal Cr
edit Union
```

In [13]:

```
!tail -1 ./data/export_jobs_w_title.txt > ./data/test_w_complete_text.txt
!awk -F'|' 'BEGIN{OFS="|"}{print $1, $2, $3}' ./data/test_w_complete_text.txt >
./data/test_labels.txt
!awk -F'|' 'BEGIN{OFS="|"}{print $4}' ./data/test_w_complete_text.txt > ./data/t
est.txt
```

In [17]:

```
!head -2 ./data/train.txt | tail -1 > ./data/sample.txt
```

3. Cleansing Data - Stop words, Tokenizing and Stemming

Failing to cleanse and normalize the data properly can decrease the overall effectiveness of the model. Let's define few functions before we take off

In [4]:

```
# replace forward and back slash, hyphen, underscores and other characters
def preprocess(text):
    clean = text
    clean = re.sub("[/_-]", " ", clean)
    clean = re.sub("[^a-zA-Z.+3]", " ", clean) # get rid of any terms that aren'
t words
    return clean
```

In [5]:

```
# define a tokenizer and stemmer to returns the set of stems in the text passed

def tokenize_and_stem(text):
    # tokenize by sentence, then by word to catch any punctuations
    tokens = [word.lower() for sent in nltk.sent_tokenize(text) for word in nltk
.word_tokenize(sent)]
    filtered_tokens = []

    # remove stop words from tokens
    en_stop = set(get_stop_words('en') + stopwords.words("english"))
    stopped_tokens = [i for i in tokens if not i in en_stop]

    # filter out tokens not containing alphanumeric
    for token in stopped_tokens:
        if re.search('[a-zA-Z]', token):
            filtered_tokens.append(token)

    stems = [stemmer.stem(t) for t in filtered_tokens]

    return stems

def tokenize_only(text):
    # tokenize by sentence, then by word to catch any punctuations
    tokens = [word.lower() for sent in nltk.sent_tokenize(text) for word in nltk
.word_tokenize(sent)]
    filtered_tokens = []

    # remove stop words from tokens
    en_stop = set(get_stop_words('en') + stopwords.words("english"))
    stopped_tokens = [i for i in tokens if not i in en_stop]

    # filter out tokens not containing alphanumeric
    for token in stopped_tokens:
        if re.search('[a-zA-Z]', token):
            filtered_tokens.append(token)

    return filtered_tokens
```

In [6]:

```
# create p_stemmer of class SnowballStemmer
stemmer = SnowballStemmer("english")
```

Read training data

In [7]:

```
# compile training docs into a list
train = [ preprocess(line.decode('unicode_escape').encode('ascii', 'ignore')) fo
r line in open(os.path.join(DATA_DIR, 'train.txt'), 'r') ]
```

In [8]:

```
# compile training labels for tracking and debugging purposes only
train_labels = [ line.strip('\n').split('|') for line in open(os.path.join(DATA_
DIR, 'train_labels.txt'), 'r') ]
```

In [9]:

```
train_labels[0]
```

Out[9]:

```
['indeed_08b4c32dcb730ba2',
 'Material Control Specialist 1',
 'PRIMUS',
 'http://www.indeed.com/viewjob?jk=08b4c32dcb730ba2&qd=PuuFZTQAvQAUo
ZwXvwwydSD2Xbj1fmpX7gJmQg4hQRyeKBp7sm3CtfD1nezqvQufluU-vKaifQYZ4kZhP
nTR8bh_AUMzY0rkVCDEENdo8Gg&indpubnum=3869750015307590&atk=1ac2guif5b
qrpahl']
```

Creating persistent files with words (i) tokenized and stemmed and (ii) tokenized separetely.

In [10]:

```
FILE_STEM = os.path.join(DATA_DIR, 'train_stem.txt')
FILE_TOKEN = os.path.join(DATA_DIR, 'train_token.txt')
```

Calling tokenizer and stemmer functions on the training data

In [11]:

```
f_stem = open(FILE_STEM, 'w')
f_token = open(FILE_TOKEN, 'w')

for jobdesc in train:
    stemmed = tokenize_and_stem(jobdesc)
    f_stem.write(' '.join(stemmed).encode('utf-8').strip() + '\n')

    tokenized = tokenize_only(jobdesc)
    f_token.write(' '.join(tokenized).encode('utf-8').strip() + '\n')
```

4. Bag-of-Words (BoW) Corpus & Dictionary

Creating Dictionary

In [12]:

```
%time
dictionary = corpora.Dictionary([line.lower().split() for line in open(FILE_TOKE
N)])
dictionary.compactify()
dictionary.save(os.path.join(MODEL_DIR, "train_jobs.dict"))
print(dictionary)
```

```
CPU times: user 5 µs, sys: 0 ns, total: 5 µs
Wall time: 8.11 µs
Dictionary(22863 unique tokens: [u'nordisk', u'environments.investme
nt', u'circuitry', u'ebta', u'localized']...)
```

Corpus

For scalability reason, using iterator to stream job description one by one instead of reading all jobs at a time in memory

Each document in the tokenized file is converted to bag-of-words model before storing as a corpus

In [13]:

```
class jobCorpus(object):
    def __iter__(self):
        for line in open(FILE_TOKEN):
            # assume there's one document per line, tokens separated by whitespa
ce
            yield dictionary.doc2bow(line.lower().split())
```

In [14]:

```
jobs_corpus = jobCorpus()
corpora.MmCorpus.serialize(os.path.join(MODEL_DIR, "train_jobs.mm"), jobs_corpus
)
```

In [15]:

```
corpus = corpora.MmCorpus(os.path.join(MODEL_DIR, "train_jobs.mm"))
print corpus
```

```
MmCorpus(2713 documents, 22863 features, 529599 non-zero entries)
```

5. Dimensionality Reduction using Latent Semantic Indexing

Since we do not know how many topics this corpus should yield so we decided to compute this by reducing the features to n = 10 dimensions, then clustering the points for different values of K (number of clusters) to find an optimum value. Gensim offers various transforms that allow us to project the vectors in a corpus to a different coordinate space. One such transform is the Latent Semantic Indexing (LSI) transform, which we use to project the original data to 50D.

In [16]:

```
MAX_LSI_TOPICS = 10
```

In [17]:

```
%%time
dictionary = corpora.Dictionary.load(os.path.join(MODEL_DIR, "train_jobs.dict"))
corpus = corpora.MmCorpus(os.path.join(MODEL_DIR, "train_jobs.mm"))

tfidf = models.TfidfModel(corpus, normalize=True)
corpus_tfidf = tfidf[corpus]

# reduce the vector space by projecting  to 10 dimensions
lsi = gensim.models.LsiModel(corpus_tfidf, id2word=dictionary, num_topics = MAX_
LSI_TOPICS)
```

```
CPU times: user 9.04 s, sys: 79 ms, total: 9.12 s
Wall time: 9.58 s
```

In [18]:

```
# write coordinates to file
fcoords = open(os.path.join(MODEL_DIR, "train_jobs_lsi_coords.csv"), 'wb')
for vector in lsi[corpus]:
    if len(vector) != MAX_LSI_TOPICS:
        continue
    v = '\t'.join([ "{:6.6f}".format(x[1]) for x in vector ])
    fcoords.write(v + '\n')
fcoords.close()
```

In [19]:

```
!wc -l ./models/train_jobs_lsi_coords.csv
!head -2 ./models/train_jobs_lsi_coords.csv
```

```
wc: ./models/train_jobs_lsi_coords.csv: No such file or directory
head: cannot open ‘./models/train_jobs_lsi_coords.csv’ for reading:
No such file or directory
```

6. K-Means Clustering

Next we clustered the points in the reduced dimension LSI space using K-Means, varying the number of clusters (K) from 1 to 50. The objective function used is the Inertia of the cluster, defined (http://scikit-learn.org/stable/modules/clustering.html#k-means) as the sum of squared differences of each point to its cluster centroid. This value is fed from Scikit-Learn K-Means algorithm.

Reference:

- Stackoverflow (http://stackoverflow.com/questions/6645895/calculating-the-percentage-of-variance-measure-for-k-means)
- Data science central post by Vincent Granville (http://www.analyticbridge.com/profiles/blogs/identifying-the-number-of-clusters-finally-a-solution)

Determine Number of Topics

In [20]:

```
MAX_K = 100
```

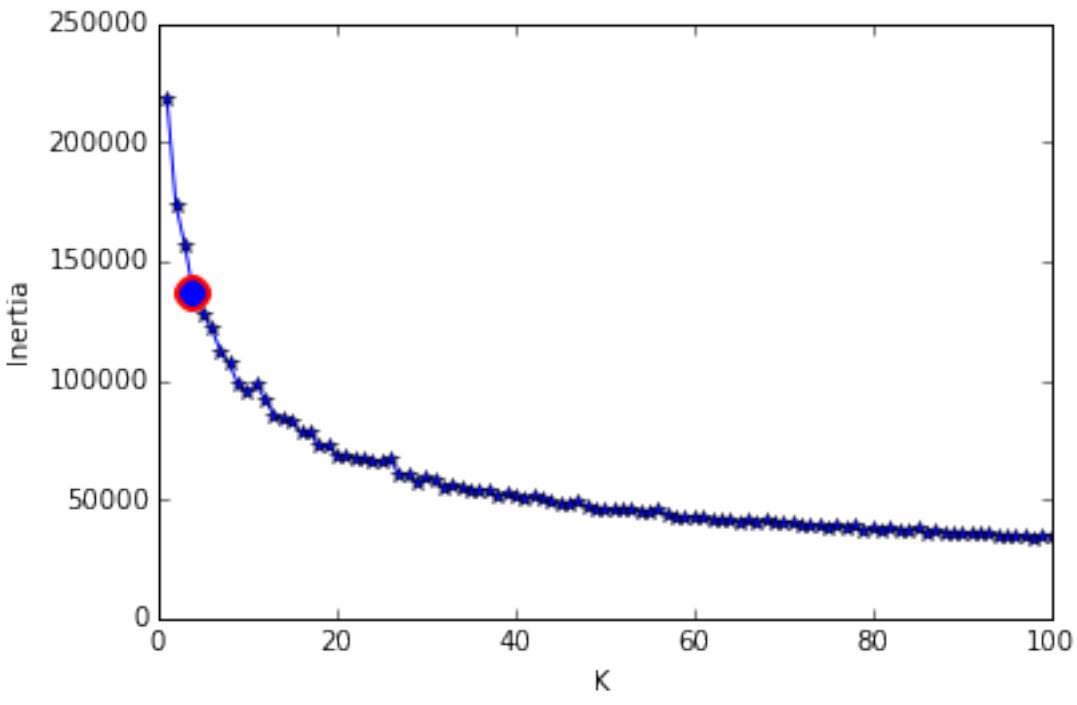
In [21]:

```
X = np.loadtxt(os.path.join(MODEL_DIR, "train_jobs_lsi_coords.csv"), delimiter="
\t")
ks = range(1, MAX_K + 1)

inertias = np.zeros(MAX_K)
diff = np.zeros(MAX_K)
diff2 = np.zeros(MAX_K)
diff3 = np.zeros(MAX_K)
```

In [22]:

```
for k in ks:
    #kmeans = KMeans(k).fit(X)
    kmeans = MiniBatchKMeans(n_clusters=k, init='k-means++', n_init=1, init_size
=1000, batch_size=1000).fit(X)
    inertias[k - 1] = kmeans.inertia_
    # first difference
    if k > 1:
        diff[k - 1] = inertias[k - 1] - inertias[k - 2]
    # second difference
    if k > 2:
        diff2[k - 1] = diff[k - 1] - diff[k - 2]
    # third difference
    if k > 3:
        diff3[k - 1] = diff2[k - 1] - diff2[k - 2]

elbow = np.argmin(diff3[3:]) + 3
print elbow

plt.plot(ks, inertias, "b*-")
plt.plot(ks[elbow], inertias[elbow], marker='o', markersize=12,
         markeredgewidth=2, markeredgecolor='r', markerfacecolor=None)
plt.ylabel("Inertia")
plt.xlabel("K")
plt.show()
```

3



We plotted the inertias for different values of K from 1 to 100. Using the approach of calculating the third differential to find an elbow point, the elbow point happens here for K=6 or 7 and is marked with a red dot

In [24]:

```
from pandas.tools.plotting import scatter_matrix
X = np.loadtxt(os.path.join(MODEL_DIR, "train_jobs_lsi_coords.csv"), delimiter="
\t")
df = pd.DataFrame(X, columns=range(10))
```

In [25]:

```
NUM_TOPICS = 3

X = np.loadtxt(os.path.join(MODEL_DIR, "train_jobs_lsi_coords.csv"), delimiter="
\t")
kmeans = MiniBatchKMeans(n_clusters=NUM_TOPICS, init='k-means++', n_init=1, init
_size=1000, batch_size=1000).fit(X)
y = kmeans.labels_

colors = [ "peru", "dodgerblue", "brown", "darkslategray", "lightsalmon", "orang
e", "springgreen", "orangered", "yellow", "firebrick" ]
```

In [26]:

```
Counter(y)
```

Out[26]:

```
Counter({0: 670, 1: 1869, 2: 174})
```

In [27]:

```
#Plotting

df = pd.DataFrame(X, columns=range(10))
scatter_matrix(df, figsize=(50,50), alpha=0.2, marker='.', c=colors, diagonal=No
ne, edgecolors='None')

#for j in range(10):
#    for k in range(10):
#        if j < k:
#            plt.figure(figsize=(10,10))
#            plt.title("Scatter plot for ({}, {})".format(j, k))
#            for i in range(X.shape[0]):
#                plt.scatter(X[i][j], X[i][k], c=colors[y[i]], s=10)
#            plt.show()
```

Out[27]:

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac648
d10>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac7f1
cd0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac9bb
b10>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac732
```

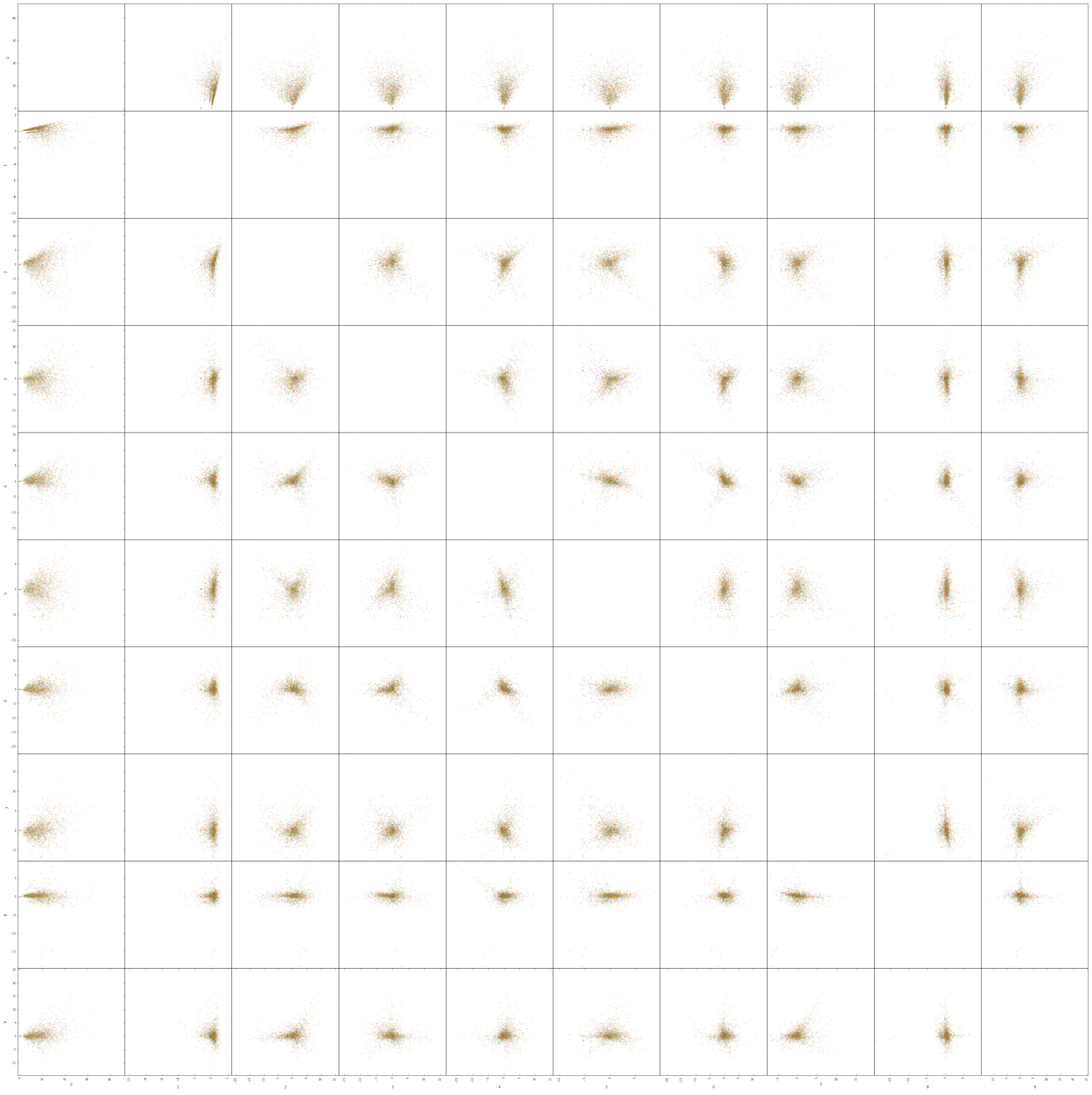
```
190>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac91b
5d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ae0e3
b10>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ae865
610>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ae90e
d90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ae964
1d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ae8c6
cd0>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ae8af
e10>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ae98f
810>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6aec36
5d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6aebba
550>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6aeb1d
a50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6aea21
790>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ae347
210>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ae2ca
250>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ae282
e50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ae1f1
a10>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ae174
990>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6adc42
6d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6adbc6
710>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6adbf0
e90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6adaae
ed0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ada31
e50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ada23
450>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad9a7
190>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad8f6
bd0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad884
a90>],
```

```
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad807
7d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad777
510>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad6fb
150>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad91a
810>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad639
a90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad5f2
910>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad52c
590>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad4b2
1d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad495
250>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad40a
f50>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad42e
4d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad2ff
b50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad284
790>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad269
dd0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad16d
b10>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad152
750>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad0d7
610>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ad05a
250>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6acfca
090>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6acf3f
c90>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7ff6acf27
b10>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6aceaa
850>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6aceca
690>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6acd9e
450>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6accba
090>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6acc1d
6d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6acba2
```

410>,

        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6acb0a
050>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6acafd
ed0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6aca82
b10>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac95a
8d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac760
510>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac587
190>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac57e
f50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac483
c90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac471
a90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac3f7
6d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac35b
550>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac2df
290>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac37f
090>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac1c7
e50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac14a
a90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac13b
110>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac0b2
e10>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac00c
a50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6abf9b
910>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6abf1f
550>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6abe90
390>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6abe07
f90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6abdeb
e10>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7ff6abd6f
b50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6abd91
190>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6abc63
750>,

```
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6abbe7
390>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6abb4b
9d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6abad0
710>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ac3a5
750>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6aba18
a10>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ab99d
650>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ab90f
110>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ab884
e10>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ab85e
a50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ab7ed
910>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ab772
550>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ab6e2
390>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ab659
f90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ab63f
e10>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ab544
b50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ab5e5
190>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7ff6ab4b7
750>]], dtype=object)

/usr/local/lib/python2.7/dist-packages/matplotlib/collections.py:590
: FutureWarning: elementwise comparison failed; returning scalar ins
tead, but in the future will perform elementwise comparison
  if self._edgecolors == str('face'):
```



7. Topic Modeling using LDA

In [28]:

```
%%time
dictionary = corpora.Dictionary.load(os.path.join(MODEL_DIR, "train_jobs.dict"))
corpus = corpora.MmCorpus(os.path.join(MODEL_DIR, "train_jobs.mm"))

# Project to LDA space
NUM_TOPICS = 3
lda = gensim.models.LdaModel(corpus, id2word=dictionary, num_topics=NUM_TOPICS,
                             chunksize=2000,
                             passes=20,
                             alpha='auto',
                             eval_every=10,
                             minimum_probability=0.01
                            )
```

```
CPU times: user 7min 6s, sys: 149 ms, total: 7min 6s
Wall time: 7min 7s
```

Topic Terms

In [29]:

```
lda.print_topics(NUM_TOPICS, 50)[0]
```

Out[29]:

```
(0,
 u'0.016*data + 0.016*business + 0.014*experience + 0.011*management
+ 0.009*project + 0.009*skills + 0.008*ability + 0.007*work + 0.006*
requirements + 0.006*team + 0.006*analysis + 0.005*knowledge + 0.005
*development + 0.005*required + 0.004*process + 0.004*information +
0.004*years + 0.004*strong + 0.004*support + 0.004*including + 0.004
*financial + 0.004*processes + 0.004*reporting + 0.004*projects + 0.
004*related + 0.003*job + 0.003*technical + 0.003*risk + 0.003*syste
ms + 0.003*working + 0.003*degree + 0.003*reports + 0.003*ensure + 0
.003*develop + 0.003*analyst + 0.003*client + 0.003*solutions + 0.00
3*internal + 0.003*preferred + 0.003*communication + 0.003*responsib
ilities + 0.003*issues + 0.003*functional + 0.003*provide + 0.003*po
sition + 0.003*quality + 0.003*manage + 0.003*responsible + 0.003*te
ams + 0.002*system')
```

In [30]:

```
ftopics = open(os.path.join(MODEL_DIR, "train_jobs_topics.txt"), 'wb')
for t in lda.print_topics(NUM_TOPICS, 50):
    ftopics.write(str(t[0]) + ':' + t[1] + '\n')
ftopics.close()
```

Job Topics

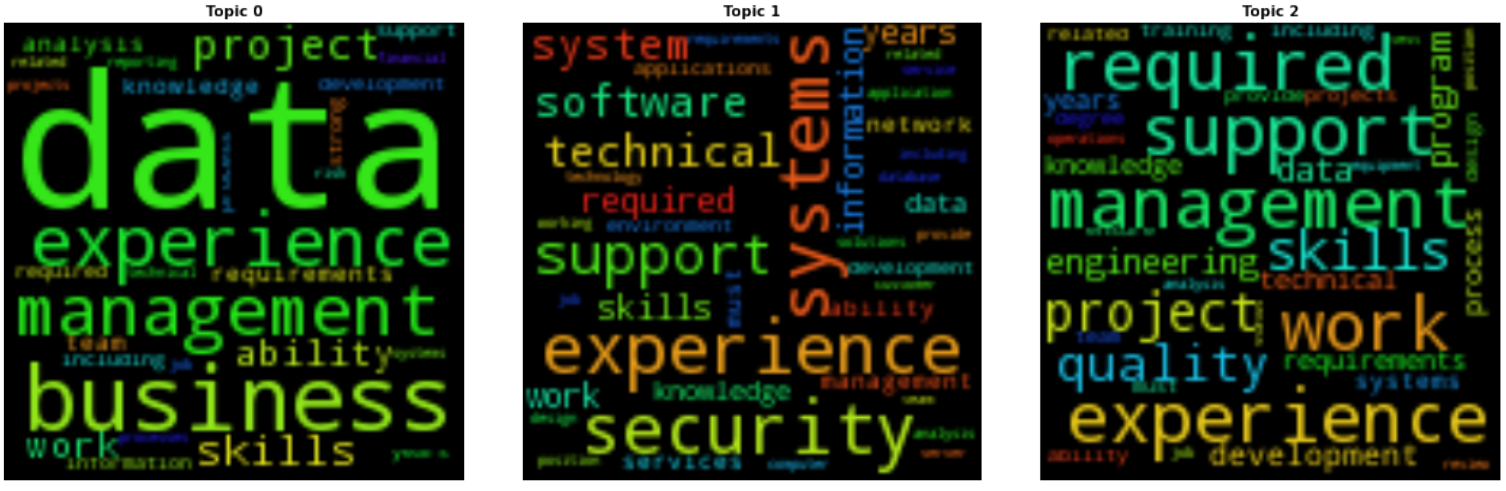
In [31]:

```
fjobtopics = open(os.path.join(MODEL_DIR, "train_jobs_topics.csv"), 'wb')
for doc_id in range(len(corpus)):
    docbow = corpus[doc_id]
    doc_topics = lda.get_document_topics(docbow)
    for topic_id, topic_prob in doc_topics:
        fjobtopics.write("%d\t%d\t%.3f\n" % (doc_id, topic_id, topic_prob))
fjobtopics.close()
```

Topic wordcloud representation for analysis

In [32]:

```
final_topics = open(os.path.join(MODEL_DIR, "train_jobs_topics.txt"), 'rb')
number_of_subplots=NUM_TOPICS
v = 0
fig = plt.figure(figsize=(15,15))
fig.subplots_adjust(left = 0.1, bottom=0.1, right=0.2, top=0.2)

for line in final_topics:
    line = line.strip('\n')
    curr_topic = line.split(':')[0]
    topic_scores = ''.join(line.split(':')[1:])

    scores = [float(x.split("*")[0]) for x in topic_scores.split(" + ")]
    words = [x.split("*")[1] for x in topic_scores.split(" + ")]

    freqs = []
    for word, score in zip(words, scores):
        freqs.append((word, score))

    elements = WordCloud(width=120, height=120).fit_words(freqs)

    v += 1
    ax1 = fig.add_subplot(int(NUM_TOPICS/3)+1, 3, v)
    ax1.set_title("Topic {}".format(curr_topic), fontsize=10, fontweight='bold')
    ax1.imshow(elements)
    ax1.axis("off")

fig.suptitle("Topics Word Cloud", fontsize=14, fontweight='bold')
plt.tight_layout()
plt.show()
final_topics.close()
```



Topic Probability Distribution for Given List of jobs

In [126]:

```
NUM_TOPICS = 3
```

In [33]:

```
topic_df = pd.read_csv(os.path.join(MODEL_DIR, "train_jobs_topics.csv"), sep="\t
",
                       names=["doc_id", "topic_id", "topic_prob"],
                       skiprows=0)

#doc_ids = []
#for i in range(6):
#    doc_ids.append(int(random.random() * max_doc_id))

def plot_job_distr(df, search_job_ids, train_labels):
    job_idx = [ x[0] for x in train_labels ]

    for job_id in search_job_ids:
        index = job_idx.index(job_id)
        filt = df[df["doc_id"] == index]
        topic_ids = filt["topic_id"].tolist()
        topic_probs = filt["topic_prob"].tolist()
        prob_dict = dict(zip(topic_ids, topic_probs))

        ys = []
        for i in range(NUM_TOPICS):
            if prob_dict.has_key(i):
                ys.append(prob_dict[i])
            else:
                ys.append(0.0)

        plt.title("Job ID: {}; Title: {}".format(train_labels[index][2], train_l
abels[index][0]))
        plt.ylabel("P(topic)")
        plt.ylim(0.0, 1.0)
        plt.xticks(range(NUM_TOPICS), ["Topic#%d" % (x) for x in range(NUM_TOPIC
S)])
        plt.grid(True)
        plt.bar(range(NUM_TOPICS), ys, align="center")
        plt.show()
```

In [34]:

```
topic_df.head()
```

Out[34]:

	doc_id	topic_id	topic_prob
0	0	2	0.989
1	1	0	0.297
2	1	1	0.246
3	1	2	0.458
4	2	0	0.999

In [35]:

```
search_job_ids = [
    'indeed_22bae41b37f33dac',
    'indeed_436a7d3058330c9d',
    'indeed_aa38b3c95efac92f',
    'indeed_ce3756994e61c0a8'
]
```

In [36]:

```
plot_job_distr(topic_df, search_job_ids, train_labels)
```

```
---------------------------------------------------------------------
-------
ValueError                                Traceback (most recent cal
l last)
<ipython-input-36-57ed17b85085> in <module>()
----> 1 plot_job_distr(topic_df, search_job_ids, train_labels)

<ipython-input-33-726606a00197> in plot_job_distr(df, search_job_ids
, train_labels)
     11
     12     for job_id in search_job_ids:
---> 13         index = job_idx.index(job_id)
     14         filt = df[df["doc_id"] == index]
     15         topic_ids = filt["topic_id"].tolist()

ValueError: 'indeed_22bae41b37f33dac' is not in list
```

Topic wise distribution

Particular job can be tagged in multiple topics. We will assign topic # to a job based on top score

In [37]:

```
topic_idx = topic_df.groupby(['doc_id'])['topic_prob'].transform(max) == topic_d
f['topic_prob']
top_topics = topic_df[topic_idx]
top_topics.groupby(['topic_id'])['topic_id'].agg(['count'])
```

Out[37]:

	count
topic_id	
0	1085
1	968
2	660

In [42]:

```
for i in range(NUM_TOPICS):
    topic_docs = [ train_labels[x] for x in top_topics[top_topics['topic_id'] ==
i]['doc_id'] ]
    topic_docs_df = pd.DataFrame.from_records(topic_docs, columns=["Job Id", "Jo
b Title", "Company", "URL"])
    topic_docs_df.to_csv(os.path.join(DATA_DIR, "topic_" + str(i) + ".csv"), sep
= "|", index = False)
    #print topic_docs_df.head()
```

8. Testing with Random Job Post

In [94]:

```
!tail -1 ~/wrk/jobs/data/export_jobs_w_title.txt | awk -F'|' '{print $5}' > ~/wr
k/jobs/data/test.txt
!tail -1 ~/wrk/jobs/data/export_jobs_w_title.txt | awk -F'|' '{print $1"|"$2"|"$
3"|"$4}' > ~/wrk/jobs/data/test_labels.txt
!cat ~/wrk/jobs/data/test.txt
!cat ~/wrk/jobs/data/test_labels.txt
```

```
McCoyâs Building Supply is looking for a strong candidate for a new
Pricing Analyst position based at our Headquarters facility in San M
arcos, Texas. This is an exempt-level position, and the final salary
for this position is to be determined.     Our ideal  Pricing Analys
t candidate will be responsible for driving price optimization and e
xecuting pricing strategies at McCoyâs.  This includes gathering com
petitor pricing, developing pricing scenarios that fit each category
âs overall strategy, and supporting your recommendations to McCoyâs
Merchants, with maximizing profitable market share growth for the bu
siness as the main goal. You need to be collaborative and persuasive
, have a technical eye, and be able to communicate with non-technica
```

l teammates. Fact based, data driven decision-making is a key part o f what youâll do to deliver the best pricing plans to our Merchandis ing and Operations Teams, and ultimately to our Born to Build Custom ers. SOME OF THE DUTIES AND RESPONSIBILITIES OF THIS POSITION I NCLUDE THE FOLLOWING : Â· Price Optimization : Incorporating competitive intelligence, develop pricing scenarios, and make recomm endations to Merchants in support of category strategies. Provide fi nancial analysis and analytical support to the Merchant community to assist group in making better pricing decisions. Â· Execute Cat egory Pricing Strategies : Present options, facilitate decisions, a nd implement pricing strategies, build and manage business rules and strategic pricing plan for all categories, and work collaboratively across the Merchandising organization Â· Deliver Competitive In telligence : Collect and Monitor competitorsâ prices and analyze re sults to drive changes to individual prices, and potential changes t o pricing strategies. Execute âwhat ifâ scenarios. Analyze and track progress on strategic pricing decisions and strategic pricing plans Â· General Responsibilities : Manage the pricing calendar to balan ce workload in the stores. Coordinate the day-to-day pricing activit ies within each merchandise category. Proactively communicate releva nt information as necessary to appropriate levels in the organizatio n, formally and informally, in both written and oral forms Requi rements SOME OF THE QUALIFICATIONS OF THIS POSITION INCLUDE : Â· Bachelor's degree from four-year college or university; or one to two years of applicable merchandising analysis experience; or equiva lent combination of education and experience Â· Ability to utili ze Microsoft Office (Word, Excel, Access and PowerPoint) and other s oftware programs at an intermediate level Â· Must be regularly a vailable and willing to work at least 8 hours per day, 40 hours per week or such other hours per day or hours per week as the employer d etermines are necessary or desirable to meet business needs Â· T his position requires occasional travel with overnight stays, so you must be able to meet the driverâs license and insurance requirements of the Company PREFERRED QUALIFICATIONS Â· Retail experienc e is strongly preferred Â· Experience with data warehousing and statistical analysis software packages (e.g., Cognos, SAS, SPSS, Sta ta) Â· Specific experience and proficiency with retail pricing s oftware packages Â· Experience with BI/Data Warehousing Tool (Co gnos, BI10+ or related tools) Â· Certified Pricing Professional (CPP) certification NOTE: A full job description will be provid ed to initially qualified candidates during the interview process.
indeed_f7b2b78d308b2e7b|Pricing Analyst|McCoy's Building Supply|http ://www.indeed.com/viewjob?jk=f7b2b78d308b2e7b&qd=PuuFZTQAvQAUoZwXvww yddUYJIifLepZz3H4vGYPJ2-_LiCPa505cRTtNIIqqYAPjqV6NiOfT96MeYswXFwOESu Hnh4d5TNqhbGUJLosmuM&indpubnum=3869750015307590&atk=1aeas3r7bb9fkfmm

In [99]:

```
!grep indeed_50bf5026f812b820 ~/wrk/jobs/data/export_jobs_w_title.txt | awk -F'|
' '{print $5}' > ~/wrk/jobs/data/test.txt
!grep indeed_50bf5026f812b820 ~/wrk/jobs/data/export_jobs_w_title.txt | awk -F'|
' '{print $1"|"$2"|"$3"|"$4}' > ~/wrk/jobs/data/test_labels.txt
!cat ~/wrk/jobs/data/test.txt
!cat ~/wrk/jobs/data/test_labels.txt
```

```
Teachers hold primary responsibility for the implementation and deve
lopment of Uncommonâs curriculum and the success of its students. Th
erefore, Uncommon Schools seeks teachers who are committed to contin
uously improving curriculum and instruction through collaboration as
part of a grade level team. Implement curricula and activities to me
et academic standards;Design and implement assessments that measure
progress towards academic standards;Use assessment data to refine cu
rriculum and inform ins o-o-o
indeed_50bf5026f812b820|High School Algebra 1 Teacher (2016-2017 Sch
ool Year)|Preparatory Charter Schools|http://www.indeed.com/viewjob?
jk=50bf5026f812b820&qd=PuuFZTQAvQAUoZwXvwwydVJX_fBthdM8Fvcy9hVLgMsm1
Jstv5h9RbSRH07keVMyhGW0PtQg12oEkmVRFhi1RJifobd018Nm_bbbb0NA9MI&indpu
bnum=3869750015307590&atk=1abet3edfbqnj8lk
```

In [100]:

```
# compile sample documents into a list
test_set = [ preprocess(line.decode('unicode_escape').encode('ascii', 'ignore'))
for line in open('/home/rt/wrk/jobs/data/test.txt', 'r') ]

# list for tokenized documents in loop
test_tokenized = tokenize_only(test_set[0])
test_dict = corpora.Dictionary([test_tokenized])
test_bow = dictionary.doc2bow(test_tokenized)
```

Let's see what topis test document belongs to

In [101]:

```
for topics in lda[test_bow]:
    print topics
```

```
(5, 0.98190453974814329)
```

So the test document belongs to topics 0, 2, 3, 7 and 9

In [58]:

```
print test_set
```

```
[u"Now Hiring Company Truck Drivers. At Transport America We Raised
Pay!  Company Truck Driver Benefits: Top 10% Industry Pay Year Round
Steady Freight Performance Pay   Experienced Drivers Earn Top Scale
in 2 Years Flexible Home Time, Including Get Home Certificates 24 7
Support, 365 Days A Year Pick Your Schedule Option Lease Purchase Op
tions Day 1 Medical Dental Vision Disability Benefits Package Transf
er Opportunities Available E Logs and an InCab Communication Hub Rol
l Stability and OnGuard System CSA Safe Carrier New Fleet of Equipme
nt  New Kenworths In Delivery  At Transport America, our goal is to
deliver excellence in all that we do. At a time when others are movi
ng to asset lite models, we are committed to running assets in netwo
rks, which gives you reliable capacity with an excellence of service
unsurpassed in the transportation industry. We are big enough to cre
ate meaningful solutions, but small enough to provide you the level
of customer service you deserve. We believe in hiring the best truck
drivers in the industry and empower them to create solutions for our
customers. Because of our asset intensity, we attract and retain the
best drivers in the trucking industry. The technology we employ is f
ocused on enhancing your service experience. Our experienced driver
base, with retention levels well above the industry average, sets us
apart from our competitors. Transport America's fleet of company tru
ck drivers is the best and most experienced on the road. We welcome
you to fill out the form above to be contacted by one of our recruit
ers! Call us for details at 877 957 3117\n"]
```

Appendix

1. Tokenizing and Stemming

In [209]:

```
vocab_stemmed = []
vocab_tokenized = []

for jobdesc in train:
    stemmed = tokenize_and_stem(jobdesc)
    vocab_stemmed.extend(stemmed)

    tokenized = tokenize_only(jobdesc)
    vocab_tokenized.extend(tokenized)
```

```
337 337
337
```

In [210]:

```
print "{}, {}".format(len(vocab_stemmed), len(vocab_tokenized))
```

```
337, 337
```

In [211]:

```
df_vocab = pd.DataFrame({'words': vocab_tokenized}, index = vocab_stemmed)
df_vocab = df_vocab.drop_duplicates()
print 'there are ' + str(df_vocab.shape[0]) + ' items in vocab_frame'
```

```
there are 235 items in vocab_frame
```

In [212]:

```
print df_vocab.head(20)
```

```
              words
potbelli    potbelly
associ    associates
job              job
make            make
custom     customers
realli        really
happi          happy
sinc           since
primari      primary
point          point
custom      customer
contact      contact
provid       provide
excel      excellent
experi    experience
provid     providing
fast            fast
friend      friendly
effici     efficient
servic       service
```

2. Adding URLs to Training Labels

In [ ]:

```
!awk -F"|" '{print $1"|"$4}' export_jobs_w_title.txt > urls.txt
```

In [ ]:

```
!awk -F"|" 'NR==FNR {a[$1]=$2;next} {if($1 in a) { print $0, a[$1] } }' OFS="|"
urls.txt train_labels.txt > train_labels_urls.txt
```