Resuminer – A Résumé Recommender System for Job Seekers using Cluster Analysis

Orlando Ferrer Hernández
Master of Engineering in Computer Engineering
Dr. Jeffrey Duffany
Electrical & Computer Engineering and Computer Science Department
Polytechnic University of Puerto Rico

Abstract *— Résumés or curriculum vitae are an essential requirement for job seekers in any industry and are usually the first interactions with third parties that are actively hiring. For a single person, there is an infinite combination of résumés possible. They can vary not only in style, but in the content and the overall presentation. Resuminer is a system for comparing and tracking résumés and their performance against jobseekers. By tracking how a third party interact online with a résumé, and by gathering a large quantity of résumés, the system can analyze new résumés as input and make recommendations as to what elements can improve the chances of getting a callback. The system will assign "interaction points" with different values to the various elements in a résumé.*

Key Terms *— Bag of Words, Clustering, Corpus, Natural Language Processing (NLP).*

INTRODUCTION

In order to measure what are the qualities of a desirable résumé, we need to be able to collect data. In order to collect the desired data on what is usually a physical or electronical document we cannot track, we need to be able to move all documents to the Web. In every résumé created, physical or electronic, there will be a link to an electronic version of the résumé using a short URL and instructions inviting the reader to visit said URL for additional information and examples of work. Every different document will have a unique URL. Documents can also incorporate a QR code for quick access. By logging all visits each URL, we can identify how many times any version of a specific document is been looked at, and create a scoring system that will track all interactions. All third party interactions with a résumé will be referred to as "interactions", and the scoring system will be referred to as "interaction points". Third party interactions with the résumé will be translated into "interaction points".

Actions that will increase the "interaction points" of a particular résumé are:

- Visiting a résumé via short URL
- Visiting a résumé via QR code scan
- Clicking on links in the résumé website
- Receiving a callback and/or interview

Callback data will be entered manually into the system.

BACKGROUND

Machine learning can be described as a "Field of study that gives computers the ability to learn without being explicitly programmed" [1]. It is currently at the forefront of many recent and upcoming new technologies. Data can be considered the new currency of technology [2], and Big Data holds the answer to many interesting questions, some of them yet unknown. Machine learning is not a new topic, but with growing processing power and storage capabilities of modern computing, new applications are being found. It can be used from a simple classification, to complex computer vision used in self driving cars. At its core, machine learning consists of a set of inputs that generate a set of outputs using a model [3]. Once we have a model, we can make predictions. The model can evolve as more data is gathered.

Natural language processing is a way for a computer to analyze, understand and manipulate text to do useful things. At its core it extracts semantic information from human language, and can be used for simple tasks such as word frequency analysis, but can be used for complex tasks such as sentiment analysis, language translation, automatic summarization, and many others. In this work, some simple natural language processing will be used.

Database Schema

The back end of the online web application will need a database to save and display data. Below is the database schema.

Table 1
Page_visits

Field	Description
Resume_id	Unique id
Incoming_URL	Referrer URL
timestamp	Date and time

Table 2
Interactions

Field	Description
Resume_id	Primary id
timestamp	Date and time
Action_performed	Clicks

Table 3
Interaction_points_value

Field	Description
Visit_resume	Points for visiting
Scan_QR	Points for scanning QR
Click on link	Points for clicking
Click on Github	Points for clicking
Points	Points

Table 4
Job_requirements

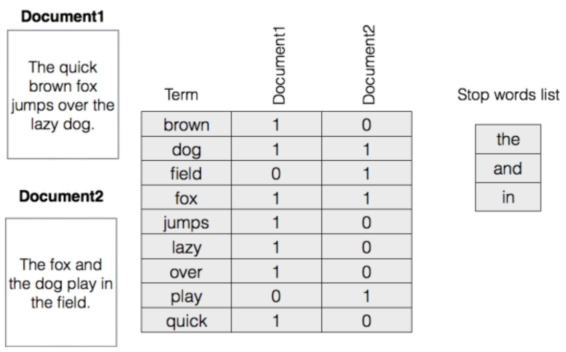
Field	Description
Resume_id	Unique id
job_title	Name of the position
Company_name	Company name
Position_description	Description

Table 5
Resume

Field	Description
Resume_id	Unique id
Short_url	Short URL for resume
Resume_text	Resume text
Company_name	Company name
Job_title	Job title

Algorithms & Models

The first level of text analysis will be a simple "bag of words" with frequency analysis. The "bag of words" is a model used in natural language processing where a document is represented as a bag (multiset) of unordered words, disregarding the grammatical structure but keeping duplicate words. After converting a document into this "bag of words" representation, it is possible to calculate various characteristics of the text, the most common being term frequency- the number of times each word appears in a document. After this is done, there are some techniques to normalize the text. Before normalization, we can remove some of the most common words such as prepositions and articles (the, on, at, in, which, on, and) and any other words that will not be relevant for the analysis- these are commonly referred to as stop words. Each individual file is called a "document", and the collection of all documents is called the "corpus".



Document1

The quick brown fox jumps over the lazy dog.

Document2

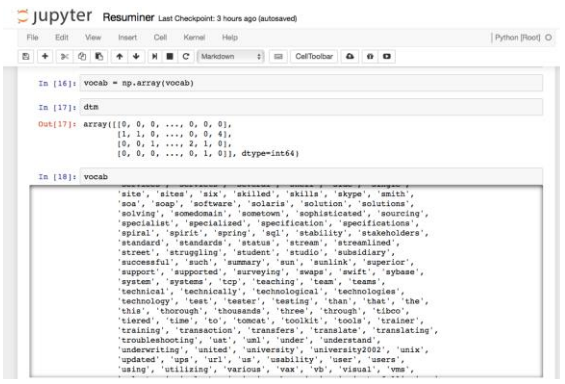
The fox and the dog play in the field.

Term	Document1	Document2
brown	1	0
dog	1	1
field	0	1
fox	1	1
jumps	1	0
lazy	1	0
over	1	0
play	0	1
quick	1	0

Stop words list
the
and
in

Figure 1
Bag of Words Representation

This information is saved internally in two separate matrixes, one for the terms and one for the term frequency.



```
In [16]: vocab = np.array(vocab)

In [17]: dtm

Out[17]: array([[0, 0, 0, ..., 0, 0, 0],
                [1, 1, 0, ..., 0, 0, 4],
                [0, 0, 1, ..., 2, 1, 0],
                [0, 0, 0, ..., 0, 1, 0]], dtype=int64)

In [18]: vocab

                'site', 'sites', 'six', 'skilled', 'skills', 'skype', 'smith',
                'soa', 'soap', 'software', 'solaris', 'solution', 'solutions',
                'solving', 'somedomain', 'sometown', 'sophisticated', 'sourcing',
                'specialist', 'specialized', 'specification', 'specifications',
                'spiral', 'spirit', 'spring', 'sql', 'stability', 'stakeholders',
                'standard', 'standards', 'status', 'stream', 'streamlined',
                'street', 'struggling', 'student', 'studio', 'subsidiary',
                'successful', 'such', 'summary', 'sun', 'sunlink', 'superior',
                'support', 'supported', 'surveying', 'swaps', 'swift', 'sybase',
                'system', 'systems', 'tcp', 'teaching', 'team', 'teams',
                'technical', 'technically', 'technological', 'technologies',
                'technology', 'test', 'tester', 'testing', 'than', 'that', 'the',
                'this', 'thorough', 'thousands', 'three', 'through', 'tibco',
                'tiered', 'time', 'to', 'tomcat', 'toolkit', 'tools', 'trainer',
                'training', 'transaction', 'transfers', 'translate', 'translating',
                'troubleshooting', 'uat', 'uml', 'under', 'understand',
                'underwriting', 'united', 'university', 'university2002', 'unix',
                'updated', 'ups', 'url', 'us', 'usability', 'user', 'users',
                'using', 'utilizing', 'various', 'vax', 'vb', 'visual', 'vms',
```

Figure 2

Runtime Terms and Term Frequency

After removing stop words, the next step towards normalization calculating the term frequency (tf). As the name suggests, the frequency of every word is calculated in every individual document. This information is relevant, but not enough when we have multiple document we want to compare. In order to analyze multiple documents, the inverse document frequency (idf) will be needed as well. The inverse document frequency will measure how important a term is. When computing the term frequency, all the term are considered equally important. The inverse document frequency searches for the terms through all the corpus, and measures the relevance of the word though all the documents. By calculating both of these, we can now calculate the term frequency-inverse document frequency (tf-idf), which is the product of both. This is a numerical statistic that reflects the relevance of a word to a document in a corpus. This value increases proportionally to the number of times it appears in a document, but is also offset by the frequency of the word in the corpus. Mathematically speaking, the term frequency tf (t, d) can be represented as "(1)"

$$tf(t,d) = 0.5 + 0.5\,\frac{f_{t,d}}{\max\{f_{t',d}: t' \in d\}} \qquad (1)$$

The inverse document frequency can be represented as "(2)".

$$idf(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|} \qquad (2)$$

The second level of analysis will involve clustering. After we have multiple résumés that have gone through the text analysis, and have been viewed online multiple times, we can do a k-means cluster analysis. The top key terms are selected for each cluster.

K-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean [4]. The way this is computed is by generating k initial means randomly. k clusters are created by associating observations to the nearest mean. The centroid of each of the k clusters now becomes the new mean. These steps are repeated until convergence. It is now possible to generate a cosine similarity matrix using the tf-idf matrix, then generate a distance matrix (1 – similarity matrix). After this we generate a plot. In the plot we should see the clusters of the different documents.

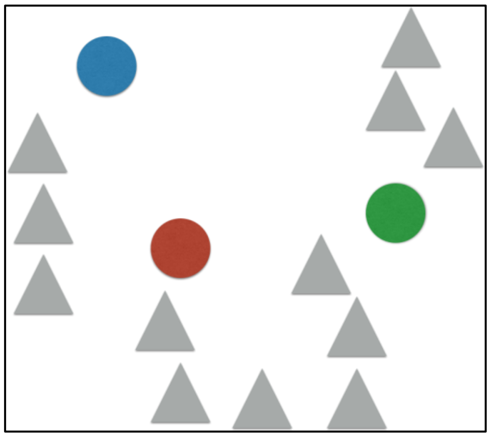


Figure 3

k Means Step 1- Random Generated k (shown in color)

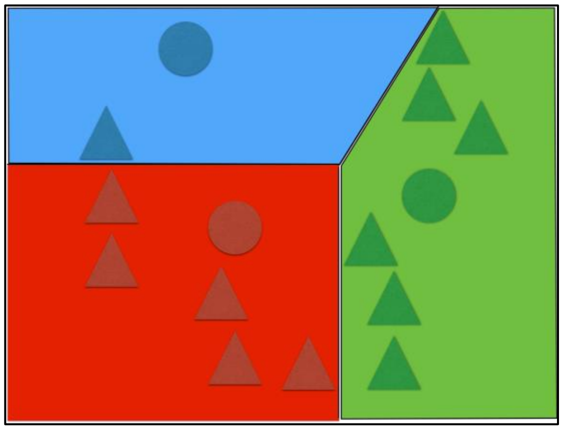


Figure 4

k Means Step 2 - Clusters Created by Associating with Nearest Mean

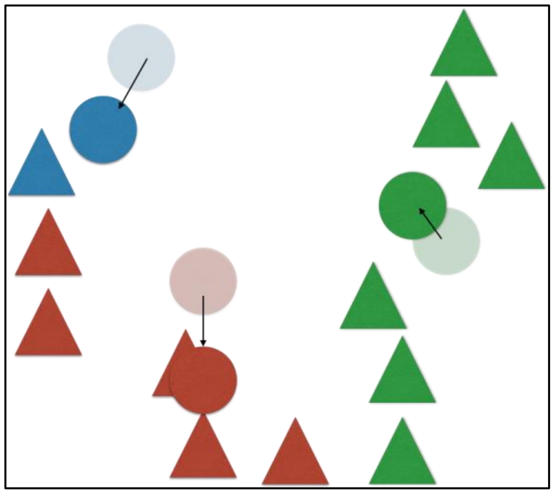


Figure 5

k Means Step 3 - Centroids become New Mean

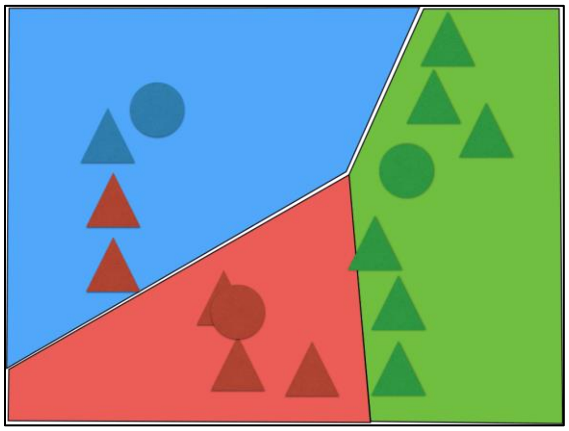


Figure 6

k Means Step 4 - Previous Steps Repeated until Convergence

The complete data analysis process, from reading the data to obtaining the cluster data can be summarized with figure 7. It can be summarized in six steps: reading the résumé data from text files, tokenizing each document into individual words, generating the term frequency-inverse document frequency (tf-idf) matrix, generating the clusters (using k-means), calculating the similarity, and plotting the results. These same steps can also be used when analyzing document style (such as fonts, font size, font style, document format). These data points can be entered into the system, to get a similar analysis about which document styling options make a document stand out. By comparing both of the resulting plots, it is possible to find what keywords attract more interest as well as which styling options seem to be preferred.

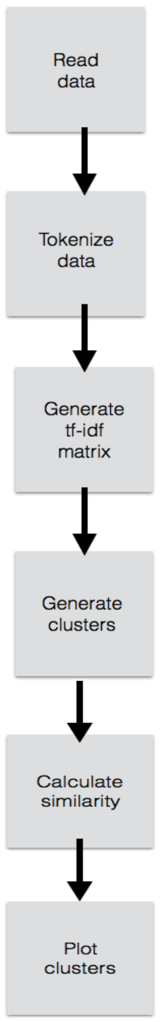


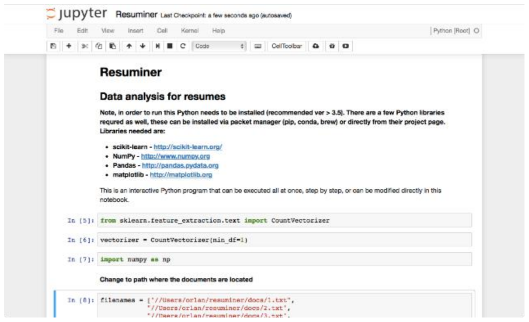
Figure 7

Resuminer Process

By feeding an untested résumé (without any "interaction points") to this system, and running it thought the same analysis, we'll be able to determine which cluster it is closest to and we can make an educated prediction on how it will perform. We can determine if this résumé is closer to the cluster with the desirable scores, or with a low score. As the dataset grows, the approximations to the correct cluster will improve.

TOOLS

The data analysis will be done using scikit-learn, a library for machine learning for Python. The scikit-learn libraries are well documented and available at http://scikit-learn.org. It is currently used extensively in the data science community for both academic research, as well as industry projects. Scikit-learn provides a range of supervised and unsupervised learning algorithms. In addition to scikit-learn, Jupyter Notebook (previously known as IPython) is being used as an interactive programming environment and documentation engine. It is available from http://jupyter.org. Jupyter Notebook runs as a server-client application in the web browser, it can be installed and executed locally or in a remote machine. Any notebooks created in this environment is easily portable to a different machine, and the code that is being executed can be documented.


jupyter Resuminer Last Checkpoint: a few seconds ago (autosaved)
File Edit View Insert Cell Kernel Help
Python [Root]
Code
CellToolbar

Resuminer

Data analysis for resumes

Note, in order to run this Python needs to be installed (recommended ver > 3.5). There are a few Python libraries requred as well, these can be installed via packet manager (pip, conda, brew) or directly from their project page. Libraries needed are:

- scikit-learn - http://scikit-learn.org/
- NumPy - http://www.numpy.org
- Pandas - http://pandas.pydata.org
- matplotlib - http://matplotlib.org

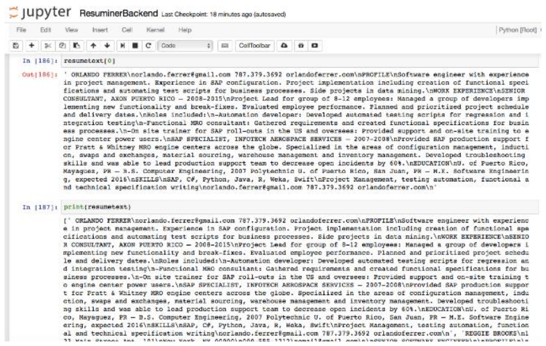
This is an interactive Python program that can be executed all at once, step by step, or can be modified directly in this notebook.

```
In [5]: from sklearn.feature_extraction.text import CountVectorizer

In [6]: vectorizer = CountVectorizer(min_df=1)

In [7]: import numpy as np
```

Change to path where the documents are located

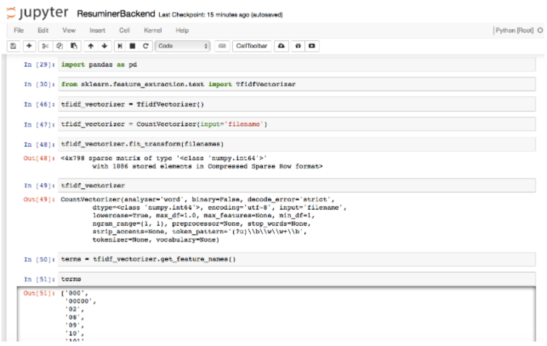
```
In [8]: filenames = ["//Users/orlan/resuminer/docs/1.txt",
                     "//Users/orlan/resuminer/docs/2.txt",
```

Figure 8
Resuminer Running in Jupyter Notebook


jupyter ResuminerBackend Last Checkpoint: 18 minutes ago (autosaved)
File Edit View Insert Cell Kernel Help
Python [Root]
Code
CellToolbar

```
In [186]: resumetext[0]

Out[186]: ' ORLANDO FERRER\norlando.ferrer@gmail.com 787.379.3692 orlandoferrer.com\nPROFILE\nSoftware engineer with experience
          in project management. Experience in SAP configuration. Project implementation including creation of functional speci
          fications and automating test scripts for business processes. Side projects in data mining.\nWORK EXPERIENCE\nSENIOR
          CONSULTANT, AXON PUERTO RICO — 2008-2015\nProject Lead for group of 8-12 employees: Managed a group of developers imp
          lementing new functionality and break-fixes. Evaluated employee performance. Planned and prioritized project schedule
          and delivery dates.\nRoles included:\n-Automation developer: Developed automated testing scripts for regression and i
          ntegration testing\n-Functional MRO consultant: Gathered requirements and created functional specifications for busin
          ess processes.\n-On site trainer for SAP roll-outs in the US and oversees: Provided support and on-site training to e
          ngine center power users.\nSAP SPECIALIST, INFOTECH AEROSPACE SERVICES — 2007-2008\nProvided SAP production support f
          or Pratt & Whitney MRO engine centers across the globe. Specialized in the areas of configuration management, inducti
          on, swaps and exchanges, material sourcing, warehouse management and inventory management. Developed troubleshooting
          skills and was able to lead production support team to decrease open incidents by 60%.\nEDUCATION\nU. of Puerto Rico,
          Mayaguez, PR — B.S. Computer Engineering, 2007 Polytechnic U. of Puerto Rico, San Juan, PR — M.E. Software Engineerin
          g, expected 2016\nSKILLS\nSAP, C#, Python, Java, R, Weka, Swift\nProject Management, testing automation, functional a
          nd technical specification writing\norlando.ferrer@gmail.com 787.379.3692 orlandoferrer.com\n'

In [187]: print(resumetext)

          [' ORLANDO FERRER\norlando.ferrer@gmail.com 787.379.3692 orlandoferrer.com\nPROFILE\nSoftware engineer with experienc
          e in project management. Experience in SAP configuration. Project implementation including creation of functional spe
          cifications and automating test scripts for business processes. Side projects in data mining.\nWORK EXPERIENCE\nSENIO
          R CONSULTANT, AXON PUERTO RICO — 2008-2015\nProject Lead for group of 8-12 employees: Managed a group of developers i
          mplementing new functionality and break-fixes. Evaluated employee performance. Planned and prioritized project schedu
          le and delivery dates.\nRoles included:\n-Automation developer: Developed automated testing scripts for regression an
          d integration testing\n-Functional MRO consultant: Gathered requirements and created functional specifications for bu
          siness processes.\n-On site trainer for SAP roll-outs in the US and oversees: Provided support and on-site training t
          o engine center power users.\nSAP SPECIALIST, INFOTECH AEROSPACE SERVICES — 2007-2008\nProvided SAP production suppor
          t for Pratt & Whitney MRO engine centers across the globe. Specialized in the areas of configuration management, indu
          ction, swaps and exchanges, material sourcing, warehouse management and inventory management. Developed troubleshooti
          ng skills and was able to lead production support team to decrease open incidents by 60%.\nEDUCATION\nU. of Puerto Ri
          co, Mayaguez, PR — B.S. Computer Engineering, 2007 Polytechnic U. of Puerto Rico, San Juan, PR — M.E. Software Engine
          ering, expected 2016\nSKILLS\nSAP, C#, Python, Java, R, Weka, Swift\nProject Management, testing automation, function
          al and technical specification writing\norlando.ferrer@gmail.com 787.379.3692 orlandoferrer.com\n', 'REGGIE BROOKS\n1
```

Figure 9
Full Document Loaded in Jupyter Notebook

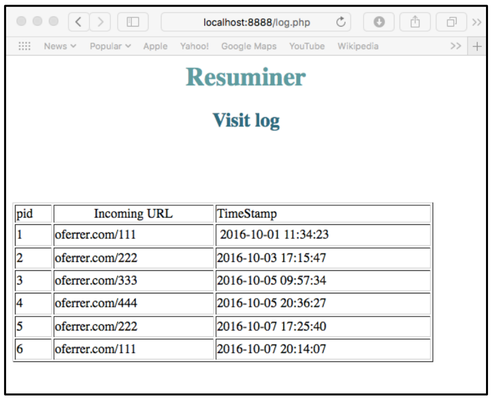

jupyter ResuminerBackend Last Checkpoint: 15 minutes ago (autosaved)
File Edit View Insert Cell Kernel Help
Python [Root]
Code
CellToolbar

```
In [29]: import pandas as pd

In [30]: from sklearn.feature_extraction.text import TfidfVectorizer

In [46]: tfidf_vectorizer = TfidfVectorizer()

In [47]: tfidf_vectorizer = CountVectorizer(input='filename')

In [48]: tfidf_vectorizer.fit_transform(filenames)

Out[48]: <4x798 sparse matrix of type '<class 'numpy.int64'>'
                 with 1086 stored elements in Compressed Sparse Row format>

In [49]: tfidf_vectorizer

Out[49]: CountVectorizer(analyzer='word', binary=False, decode_error='strict',
                 dtype=<class 'numpy.int64'>, encoding='utf-8', input='filename',
                 lowercase=True, max_df=1.0, max_features=None, min_df=1,
                 ngram_range=(1, 1), preprocessor=None, stop_words=None,
                 strip_accents=None, token_pattern='(?u)\\b\\w\\w+\\b',
                 tokenizer=None, vocabulary=None)

In [50]: terms = tfidf_vectorizer.get_feature_names()

In [51]: terms

Out[51]: ['000',
          '00000',
          '02',
          '08',
          '09',
          '10',
```

Figure 10
Performing tf-idf Analysis in Jupyter Notebook

User Interface

Users will be able to interface with the system through a regular web browser. The interface will be simple, it will display the database entries and will allow entering new data.



Resuminer

Visit log

pid	Incoming URL	TimeStamp
1	oferrer.com/111	2016-10-01 11:34:23
2	oferrer.com/222	2016-10-03 17:15:47
3	oferrer.com/333	2016-10-05 09:57:34
4	oferrer.com/444	2016-10-05 20:36:27
5	oferrer.com/222	2016-10-07 17:25:40
6	oferrer.com/111	2016-10-07 20:14:07

Figure 11
Display of the Database Entries in the Browser

Conclusion

All else being equal, there are infinite ways in which the same information can be conveyed. There are some qualities or styles of writing that make a document (a résumé, for the purpose of this work) stand out from the others. By tracking the interest in each unique document and applying data science techniques it is possible to have an understanding of which characteristics make a résumé stand out over other ones. Impact keywords could be a factor, but there are many others waiting to be discovered by applying other algorithms and gathering more training data.

Below is the result of plotting and analyzing all the points with the current test data in the Resuminer engine.

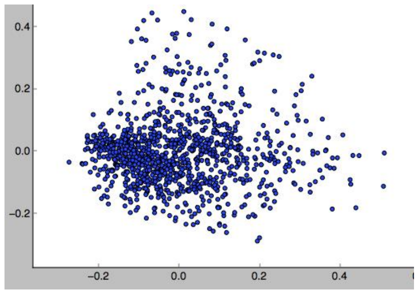


Figure 12
Resulting Clusters

FUTURE WORK

In its current state, only the backend engine has been implemented, so it is not easy to create new data, it needs to be loaded manually. In order to be able to easily add new résumés and track their performance, a web based user interfaced needs to be developed. It will interface with the existing database, and will allow for a simpler data input process. It will need to interface with the current Python backend to process the data. Another feature to be added is to not only analyze the text in résumés, but the styling information as well- such as font, font size, font style, and any others styling properties. Analyzing the style data in conjunction with the text, may yield interesting results as to which styles of résumés grab the attention of recruiters. It is also possible that the current implemented algorithm is not the most efficient one, so other algorithms and data processing techniques can be explored. As more data is collected, there will be more training data available to use and test the predictions made by the algorithm.

REFERENCES

[1] P. Tan, M. Steinbach and V. Kumar, *Introduction to data mining*, Boston: Pearson Addison Wesley, 2005.

[2] W. Eggers, R. Hamill and A. Ali, "Data as the new currency", *Deloitte Review*, no. 13, 2013, pp. 19-23.

[3] M. Fierro, "A Gentle Introduction to the Basics of Machine Learning", *SciBlog*, 2016.

[4] I. Witten, E. Frank and M. Hall, *Data mining*. Burlington, MA: Morgan Kaufmann, 2011.