Practicum Report for Attain, LLC

By Weihang Wen, Mengchen Xiao, Xuan Yang, Weichao Zhu, Yinian Lyu

Master of Science in Business Analytics, Sep 2018,

George Washington University School of Business

A Report submitted to

The Faculty of

The School of Business

Of The George Washington University

In partial fulfillment of the requirements

For the degree of Master of Science

In Business Analytics

Nov 25, 2018

Abstract

This practicum project has a holistic goal, to create analytics solutions which improve Attain's digital consulting platform. A number of projects will be undertaken to bring these improvements to different phases of the digital business cycle. The team on the one hand doing text mining in resumes which aims to use Machine Learning to identify workforce characteristics; on the other hand, the team develops the HAMS (Human Analytics Managed Service) to help client understand people's opinion of certain topics on the Internet, and in our specific case to target posts that not only contain hate speech on discussion board but also have strong intention of implementing attacking plans; additionally, the team analyzes project risks and design new risk tracking.

Table of Contents

Executive Summary

Improving the digital consulting platform with AI & Analytics

The team is experimenting with several areas of Attain's digital consulting platform to bring analytics to bear. The following mini-projects will combine to create an overall analytics upgrade to Attain's digital consulting platform:

1. Improve 'Service Selector'. After we have identified the customer & the customer's requirements, then we source those requirements to Attain capabilities. From the 'Service Categories' various services can be selected. The team is working on optimizing the 'association rules' which suggest complimentary service sets (to planners).

2. Improve 'Staffing' visibility. One of the capabilities Attain applies to jobs is skilled workers. These staffs are currently listed by name. The team is using machine learning to cull through Attain resumes by skill. Once completed planners will be able to select staff, for a job, by querying which staff possesses a given skillset.

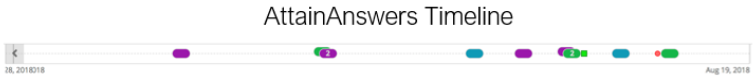
3. Design an Analytics Managed Service. AttainAnswers will be an online platform for the delivery of traditional consulting services & Asset-based consulting. An evolution beyond that would be 'Analytics Managed Services'. Based on a prior engagement pattern, the team is designing & building a prototype of an Analytics Managed Service to be exposed and offered on the AttainAnswers platform.

4. Consulting Risks Analysis. A key execution aspect in consulting engagements is risk management. The team took Attain data, detailing the risk profiles of engagements, and analyzed them for a cause. Insights culled from this analysis will inform how the 'risk features' of AttainAnswers will be designed to best assist engagement planners.

5. Bonus task. If time allows the team will examine the AttainAnswers lifecycle and make recommendations for how to analyze holistic engagement performance such that AttainAnswers will become smarter over time. With this insight, the platform can direct executives as to how the customer, partner or Attain fulfillment trends are changing over time.

Introduction

AttainAnswers Structure



Understand Customer Demand	Collaboratively Plan the Project	Execute & Oversee Work as it Occurs	Use Apps	Evaluate Human & AI Behavior
Document customer details, request & fulfillment options.	Customer & team work together to create the project plan & make assignments.	Team members use plan & free form collaboration. Managers track progress/risks.	Some Analytics Apps Available soon	During & after project execution, users will have opportunities to rate & improve the process

The goal of AttainAnswers is to provide an intelligent platform that both digitizes all customer interactions and learns from them. After a number of projects are run on the platform it begins to see trends across projects that most people wouldn't notice. Ultimately AttainAnswers will assist the company in directing its future efforts based on its knowledge of market trends & Attain's unique capabilities to execute to them.
Four areas were chosen to positively impact the business. Project 1 was developing a better algorithm methodology to assign services to jobs. Project 2 analyzed staffing, to index resumes by key skills and to train a model to recognize quality staff. Project 3 involved developing an analytics service to pulse context elements of engagements via custom internet search analysis. Project 4 involved an analysis of project risks, identifying key characteristics and time epochs of risks

The problem we are trying to solve is to help Attain. Inc, improve its digital consulting platform. Attain. Inc, a risk consulting company, now is planning to build a brand new consulting platform based on AttainAnswers which is a cloud-based collaborative work environment.

Association Rule Mining:

The first step for improving the customer digital journey on Attain Answers is to understand customer demand, which is the foundation stone of the whole of scheme. For this step, we improve Association Rule Mining Algorithms. Specifically, we optimize code of algorithms step by step and perfect the criteria selecting in case of meeting different demand in different dataset conditions.

Resume Mining:

Secondly, to better and faster execute the plan, the company needs to choose the right members. This is another place we will be part of it. We will index everyone's skills on their resumes and find the one with skills suitable for different plans, by utilizing the AI library.

Online Posts Mining:

By leveraging the AttainAnswers, some jobs will be executed by Artificial Intelligence library and the place we kick in is to build Human Analytics Managed Services used to analyze public attitude and reflection towards certain topics which may lead to risks. Thus, to identify risk, what we do is to build a pipeline connecting a web scraping platform called Webhose and social network analysis platform called Maltego. After finishing the pipeline, we will be working on Anti-semitism & Antifa topic and analyze how people who against Anti-semitism & Antifa connecting with each other or other organizations and what risk they might cause. The last part of the project is to evaluate the job execution. After the project finished, all these information will be stored in a database and used to analyze customers' demand trend by using unsupervised learning. And what we do is to improve Attain consulting services by using that trend.

Risk Data Analysis:

Another problem is that will risk factors e.g, schedule, budget, etc. influence the overall project risk and if they do, what is the relationship between them? The dataset Attain provides us with 8 projects and two aspects of risk in every project. One aspect focuses on the difference in risk assessment between project manager risk and Attain assessment. Another one shows the way risk assessment project manager obtained by timing the rates of probability and impact. With the help of risk data, we are able to see the upcoming risks and prevent risks with high impact and high probability during planning job execution phase and job execution phase.

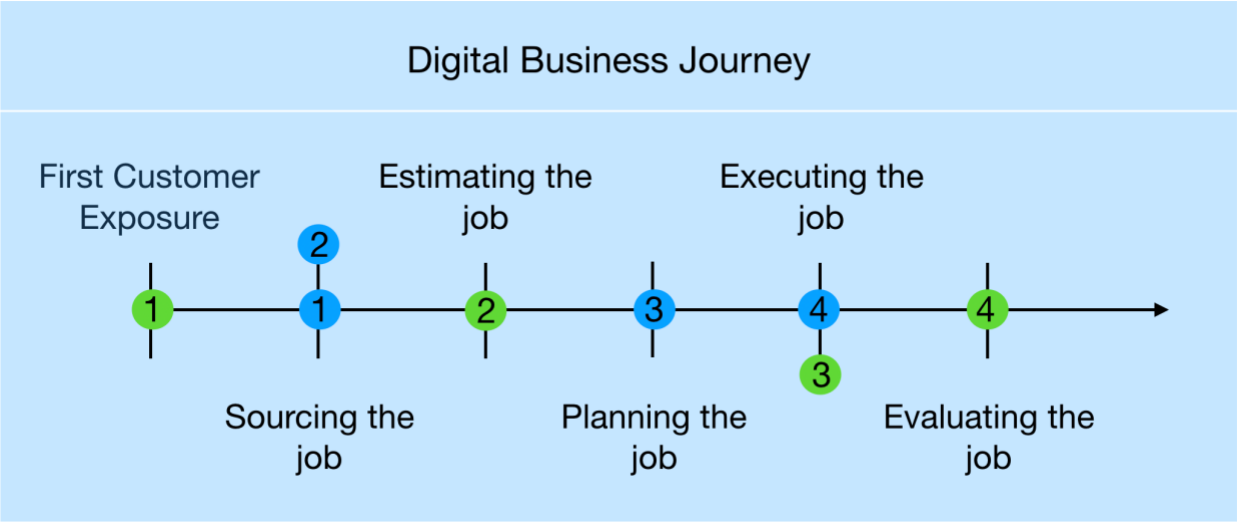
Background

Background of Attain

Attain is a management consulting company which specializes in technology implementations. The current business model involves customer interactions in a fairly traditional manner - phone calls, emails, exchanging paper & electronic documents, etc. As the consulting business model is under pressure & A.I. enters the scene, Attain undertook an effort to transition to a digital business platform. Attain understood that a proper business transformation, to digital, meant that all significant customer touchpoints would need to be converted to digital. Additionally, Attain knew that Artificial Intelligence would have to be a critical component of any digital business platform. By August 2018 Attain produced a working prototype of AttainAnswers that spanned the business lifecycle from identifying a customer, documenting customer needs, selecting the appropriate consulting+ services, planning a project & executing a project.

Workflow of AttainAnswer

The 4 blue numbers pertain to projects already underway & explained above. The green numbers pertain to future AI projects.



Number 1. After running a number of projects through AttainAnswers the following data will have been collected: Customer name, industry, line of business, mission, project goals, prior projects, and specific needs. Once enough data are collected Answers will begin to see trends in 'customer types served', 'customer demand signals', etc. These analyses will provide directional guidance as to where customer demand is headed.

Number 2. After a number of projects have been run through the system there will be estimation data, culled from prior similar estimates, to assist Attain planners. Similarly, since both risks & risk mitigations will be tracked, the system will also be able to advise the planner to include concomitant risks & their appropriate mitigations into their estimates.

Number 3. Once a number of jobs, of a given types, have been executed the system will begin to see trends, or partial execution patterns. These execution trends can be linked back to planned services such that later, when a planner selects a given job subset (of services) the system can recommend a plan to execute them.

Number 4. An important future improvement will be the evaluation of human and machine behavior. System events suggested by AI or generated by humans will be subject to evaluation such that later teams can benefit from knowing how that component of the human/machine ecosystem works. Kind of like a yelp for AttainAnswers capabilities.

Setting of the work

Back to 4 projects the team participated, project 1,2 and 3, were using python and jupyter notebook to conduct exploratory analysis and modeling part; plus, the visualization part in project 3 was used Maltego to visualize social network. Project 4 was modeled in SAS programing and visualized in Tableau software.

Methodology

Association Rule Mining

Association rule mining (ARM) is a useful data mining technique to detect patterns and rules. To provide better service for Attain's customers and understand better their demand patterns, it's significant to improve the current association ARM algorithm according to different dataset features and different technical requests.

To perform ARM, the collected information is analyzed and a number of relevant rule criteria are detected. One of the applied algorithm is Apriori. Apriori finds rules and relations among variables in a data set. The algorithm finds frequent patterns and calculates the rule's several indices. Support and confidence, are the two major indices, which have useful applications to evaluate the rules. Lift is the third but the most important index. Lift(A -> B) refers to the increase in the ratio of sale of B when A is sold. Lift(A –> B) can be calculated by dividing Confidence(A -> B) divided by Support(B).

For large sets of data, there can be hundreds of items in hundreds of thousands transactions. The Apriori algorithm tries to extract rules for each possible combination of items. This process can be extremely slow due to the number of combinations. To speed up the process, we need to perform the following steps:
1. Set a minimum value for support and confidence. This means that we are only interested in finding rules for the items that have certain default existence (e.g. support) and have a minimum value for co-occurrence with other items (e.g. confidence).
2. Extract all the subsets having higher value of support than minimum threshold.
3. Select all the rules from the subsets with confidence value higher than minimum threshold.
4. Order the rules by descending order of Lift.

Specifically, we respectively implement python code step by step. As first step, we use generate_rules to take dataframes of frequent itemsets as produced by the apriori function in mlxtend.association when we want to generate association rules from frequent itemsets. The generate_rules() function allows you to (1) specify your metric of interest and (2) the according threshold.

```
import pandas as pd
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori


dataset = [['Milk', 'Onion', 'Nutmeg', 'Kidney Beans', 'Eggs', 'Yogurt'],
           ['Dill', 'Onion', 'Nutmeg', 'Kidney Beans', 'Eggs', 'Yogurt'],
           ['Milk', 'Apple', 'Kidney Beans', 'Eggs'],
           ['Milk', 'Unicorn', 'Corn', 'Kidney Beans', 'Yogurt'],
           ['Corn', 'Onion', 'Onion', 'Kidney Beans', 'Ice cream', 'Eggs']]

te = TransactionEncoder()
te_ary = te.fit(dataset).transform(dataset)
df = pd.DataFrame(te_ary, columns=te.columns_)
frequent_itemsets = apriori(df, min_support=0.6, use_colnames=True)

from mlxtend.frequent_patterns import association_rules

association_rules(frequent_itemsets, metric="confidence", min_threshold=0.7)
```

Next, we discuss the condition that when we you are interested in rules according to a different metric of interest, you can simply adjust the metric and min_threshold arguments . E.g. if you are only interested in rules that have a lift score of >= 1.2, you would do the following:
rules = association_rules(frequent_itemsets, metric="lift", min_threshold=1.2)

And then, say, we are ony interested in rules that satisfy the following criteria:
rules["antecedent_len"] = rules["antecedents"].apply(lambda x: len(x))
rules[(rules['antecedent_len'] >= 2) &
    (rules['confidence'] > 0.75) &
    (rules['lift'] > 1.2)]

Thirdly, when we face with a condition where there are frequent itemsets with incomplete antecedent and consequent information. For example, the confidence is computed as confidence(A→C)=support(A→C)/support(A). But we do not have support(A). In these scenarios, where not all metrix can be computed, due to incomplete input DataFrames, you can use the support_only=True option, which will only compute the support column of a given rule that does not require as much info: support(A→C)=support(A∪C).

Resume Text Mining

1. Topic Modelling with LDA model

The first part of the resume mining project is utilizing the LDA (Latent Dirichlet Allocation) model which is frequently used to classify text in a resume to a particular topic with regarding to word frequency estimators like TF-IDF or Bag of Words to discover the abstract topics in the resume files such that we could better identify the features of workforce in Attain. (Li, 2018)

To build an LDA model, we used Gensim package. But first, we have to do the text preprocessing before running the model. We split each resume manually into CSV file with each row representing a resume.

```
In [7]: df = pd.read_csv('Resumetest.csv');
        df.head(10)
```

Out[7]:

	Index	Resume_text
0	1	Technical IT Agile Project Manager / Scrum Mas...
1	2	Certified Project Management Professional (PMP...
2	3	Mr. Agee serves as a Partner and Board Member ...
3	4	Ms. Morin-Kensicki brings over 20 years of pro...
4	5	Information Technology Professional with six y...
5	6	Mr. Thomas Pagurek\n\nTechnical Services Manag...
6	7	I'm looking to forward my career, which focuse...
7	8	Attain LLC\nChief Engineer & Infrastructure Op...
8	9	Chesapeake Bay Program | Annapolis, MD\t(05/20...
9	10	Mary Baldwin University Library Assistant, Gra...

We tokenized each sentence into a list of words, unnecessary characters and deleting all the punctuations.

Tokenize and data clean-up

```
In [9]: def sent_to_words(sentences):
            for sentence in sentences:
                yield(gensim.utils.simple_preprocess(str(sentence), deacc=True))  #

        data_words = list(sent_to_words(data))

        print(data_words[:1])
```

```
[['technical', 'it', 'agile', 'project', 'manager', 'scrum', 'master', 'w
ho', 'works', 'tirelessly', 'with', 'relevant', 'stakeholders', 'to', 'ma
nage', 'benefits', 'throughout', 'the', 'project', 'life', 'cycle', 'fo
r', 'benefit', 'realization', 'at', 'project', 'completion', 'grasps', 'o
rganizational', 'culture', 'and', 'project', 'environment', 'and', 'initi
atives', 'as', 'well', 'as', 'business', 'strategies', 'and', 'structur
e', 'quickly', 'strong', 'influencer', 'with', 'emotional', 'intelligenc
e', 'strong', 'business', 'acumen', 'situational', 'awareness', 'and', 't
uned', 'to', 'lead', 'diverse', 'project', 'teams', 'driving', 'performan
ce', 'through', 'coaching', 'and', 'mentoring', 'as', 'previous', 'princi
pal', 'with', 'solutionersnet', 'have', 'always', 'worked', 'diligently',
```

To transform the words to its root word, like transforming ‘managing’ to ‘manage’, we tried two approaches: Stemming and Lemmatization. We found that Lemmatization is better than stemming because, for lemmatization, it can successfully transform ‘managing’ to ‘manage’ while for stemming, it can only convert ‘managing’ to ‘manag’, which is not precise and may cause problems. Also, we deleted all the stopwords.

```
]: def lemmatization(texts, allowed_postags=['NOUN', 'ADJ', 'VERB', 'ADV']):
       """https://spacy.io/api/annotation"""
       texts_out = []
       for sent in texts:
           doc = nlp(" ".join(sent))
           texts_out.append([token.lemma_ if token.lemma_ not in ['-PRON-'] else '' for token in do
       return texts_out

   import en_core_web_sm
   nlp = en_core_web_sm.load(disable=['parser', 'ner'])

   # Do lemmatization keeping only Noun, Adj, Verb, Adverb
   data_lemmatized = lemmatization(data_words, allowed_postags=['NOUN', 'ADJ', 'VERB', 'ADV'])

   print(data_lemmatized[:2])
```

```
'project', 'execution', 'plan', 'review', 'approve', 'project', 'control', 'system', 'whic
h', 'measure', 'progress', 'performance', 'provide', 'early', 'warning', 'deviation', 'pla
n', 'identify', 'corrective', 'action', 'be', 'take', 'identifie', 'quality', 'requirement',
'ensure', 'proper', 'process', 'be', 'identify', 'implement', 'achieve', 'contractual', 'qua
lity', 'commitment', 'assess', 'customer', 'perception', 'quality', 'regular', 'periodic',
'basis', 'manage', 'day', 'day', 'approval', 'bid', 'tabulation', 'commitment', 'major', 'pu
rchase', 'order', 'contract', 'sub', 'contract', 'meet', 'air', 'traffic', 'organization',
'business', 'stakeholder', 'sm', 'pms', 'clarity', 'integrator', 'capture', 'document', 'spe
cify', 'requirement', 'need', 'launch', 'clarity', 'demand', 'manage', 'funding', 'approve',
'project', 'near', 'hand', 'advance', 'rd', 'obtain', 'approval', 'funding', 'loas', 'variou
s', 'region', 'headquarters', 'location', 'begin', 'design', 'effort', 'manage', 'outsourc
e', 'effort', 'various', 'contract', 'firm', 'solicit', 'project', 'scope', 'agreement', 'ta
sk', 'order', 'specified', 'design', 'service', 'behalf', 'faa', 'manage', 'deployment', 'fu
nding', 'cost', 'government', 'fund', 'equipment', 'gfe', 'terminal', 'project', 'as', 'wel
l', 'management', 'funding', 'sit', 'report', 'design', 'contract', 'award', 'construction',
'effort', 'perform', 'business', 'case', 'cost', 'benefit', 'analysis', 'new', 'existing',
'project', 'fy', 'assist', 'clarity', 'administrator', 'create', 'user', 'would', 'set', 're
source', 'rol', 'right', 'resetting', 'password', 'test', 'new', 'clarity', 'functionality',
'functionality', 'test', 'development', 'environment', 'prior', 'push', 'production', 'set',
'new', 'schedule', 'clarity', 'tool', 'uploaded', 'legacy', 'msp', 'schedule', 'system', 'wo
```

Then we created a dictionary from the data we processed, which have the frequency of the words.

```
dictionary = gensim.corpora.Dictionary(data_lemmatized)
```

```
count = 0
for k, v in dictionary.iteritems():
    print(k, v)
    count += 1
    if count > 10:
        break
```

```
0
3178 pool
1128 vary
3423 jenkin
4650 reconstruct
950 go
2839 prtg
1409 manipulation
5827 mortgage
458 drawing
774 ticket
```

We filtered out the tokens that appear in less than 15 documents, more than 0.5 documents and kept only first 100000 most frequent tokens. We then created a dictionary that contains the number of words and the frequency. We saved it to bow_corpus.

```
In [14]: dictionary.filter_extremes(no_below=15, no_above=0.5, keep_n=100000)
```

```
In [15]: bow_corpus = [dictionary.doc2bow(doc) for doc in data_lemmatized]
         bow_corpus[14]
```

```
Out[15]: [(6, 1),
          (9, 1),
          (11, 1),
          (13, 1),
          (18, 2),
          (19, 1),
          (31, 1),
          (36, 1),
          (38, 2),
          (47, 1),
          (50, 1),
          (53, 1),
          (54, 3),
          (56, 1),
          (64, 1),
          (69, 2),
```

So now, we completed our data preparation and began to run the data into LDA Mallet Model by using the Gensim package.

Building LDA Mallet Model

In [19]:
```
import os
from gensim.models.wrappers import LdaMallet

os.environ['MALLET_HOME'] = '/Users/weichaozhu/Desktop/Practicum/mallet-2.0.8'

mallet_path = '/Users/weichaozhu/Downloads/mallet-2.0.8/bin/mallet'
ldamallet = gensim.models.wrappers.LdaMallet(mallet_path, corpus=bow_corpus, num_topics=10, id2word=dictionary)
```

In [ ]:
```
```

In [21]:
```
# Show Topics
pprint(ldamallet.show_topics(formatted=False))

# Compute Coherence Score
coherence_model_ldamallet = CoherenceModel(model=ldamallet, texts=data_lemmatized, dictionary=dictionary, coherence='c_
coherence_ldamallet = coherence_model_ldamallet.get_coherence()
print('\nCoherence Score: ', coherence_ldamallet)
```

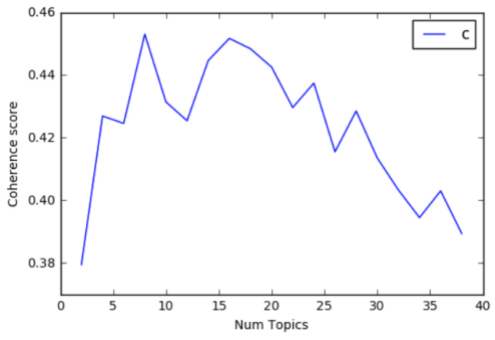
The coherence score we get from the LDA Mallet model is 0.424, which is a good way to judge the performance of the model. Basically, the higher the coherence score, the better the model.

```
[(0,
  [('research', 0.073031236251649798),
   ('ms', 0.045314562252529694),
   ('perform', 0.036955565332160142),
   ('federal', 0.036075670919489662),
   ('communication', 0.028156621205455347),
   ('associate', 0.024637043554773426),
   ('analyze', 0.023757149142102949),
   ('student', 0.022877254729432469),
   ('conduct', 0.021557413110426749),
   ('assistant', 0.021117465904091508)]),
 (1,
  [('schedule', 0.055711282410651715),
   ('risk', 0.044498948843728098),
   ('quality', 0.038542396636299929),
   ('update', 0.033286615276804488),
   ('review', 0.030133146461107218),
   ('change', 0.028731604765241767),
   ('meet', 0.0255781359495445),
   ('budget', 0.023475823405746322),
   ('track', 0.023125437981779958),
   ('document', 0.022424667133847231)]),
 (2,
  [('tool', 0.072430964237211404),
   ('web', 0.051607062019013127),
   ('analyst', 0.041647804436396561),
   ('build', 0.041195110909913991),
   ('engineering', 0.036215482118605702),
   ('model', 0.027161611588954276),
   ('developer', 0.026708918062471707),
   ('cloud', 0.026708918062471707),
   ('enterprise', 0.023087369850611137),
   ('java', 0.022181982797645994)]),
 (3,
  [('task', 0.054852320675105488),
   ('key', 0.035673187571921748),
   ('health', 0.034522439585730723),
   ('serve', 0.034138856923667048),
   ('leadership', 0.033371691599539698),
   ('planning', 0.029919447640966629),
   ('area', 0.027234369006520907),
   ('wide', 0.026850786344457232),
   ('state', 0.024549290372075181),
   ('activity', 0.024549290372075181)]),
 (4,
  [('delivery', 0.074744661095636031),
   ('national', 0.037140204271123488),
   ('association', 0.036211699164345405),
   ('community', 0.034354688950789226),
   ('case', 0.033426183844011144),
   ('defense', 0.027855153203342618),
   ('consultant', 0.025069637883008356),
   ('honor', 0.024605385329619311),
   ('good', 0.024605385329619311),
   ('state', 0.024605385329619311)]),
 (5,
  [('contract', 0.12790294627383014),
   ('proposal', 0.073483535528596183),
   ('government', 0.044367417677642983),
   ('federal', 0.023223570190641248),
   ('account', 0.022876949740034663),
   ('staff', 0.022530329289428077),
   ('agreement', 0.021143847487001734),
   ('order', 0.020103986135181974),
   ('review', 0.020103986135181974),
   ('administration', 0.020103986135181974)]),
```

```
(6,
 [('database', 0.094045368620037803),
  ('test', 0.086011342155009454),
  ('oracle', 0.070415879017013239),
  ('testing', 0.059546313799621928),
  ('production', 0.03780718336483932),
  ('upgrade', 0.030718336483931945),
  ('sql', 0.02835538752362949),
  ('multiple', 0.025992438563327031),
  ('issue', 0.024574669187145556),
  ('role', 0.024102079395085067)]),
(7,
 [('network', 0.11427566807313642),
  ('server', 0.068917018284106887),
  ('computer', 0.050632911392405063),
  ('microsoft', 0.047819971870604779),
  ('window', 0.038326300984528834),
  ('infrastructure', 0.036568213783403657),
  ('engineer', 0.024261603375527425),
  ('access', 0.021097046413502109),
  ('digital', 0.018635724331926864),
  ('configuration', 0.018635724331926864)]),
(8,
 [('issue', 0.046343537414965989),
  ('assist', 0.043792517006802721),
  ('office', 0.036564625850340135),
  ('problem', 0.029336734693877552),
  ('desk', 0.025935374149659865),
  ('procedure', 0.024659863945578231),
  ('account', 0.0233843537414966),
  ('installation', 0.0233843537414966),
  ('equipment', 0.022534013605442178),
  ('skill', 0.020833333333333332)]),
```

```
(9,
 [('product', 0.058802045288531772),
  ('strategy', 0.035427319211102995),
  ('enterprise', 0.034696859021183343),
  ('deliver', 0.030314097881665451),
  ('budget', 0.028487947406866325),
  ('sale', 0.028487947406866325),
  ('strategic', 0.027392257121986851),
  ('improve', 0.027027027027027029),
  ('director', 0.026296566837107377),
  ('improvement', 0.024470416362308255)])]

Coherence Score:  0.424103808543
```

To improve the model, we tried to find the optimal number of topics by trying different values of the number of topics and choose the one that has the highest coherence score. Form the output, we notice that the coherence value is highest(0.453) when the number of topics is 8.




```
Num Topics = 2  has Coherence Value of 0.3794
Num Topics = 4  has Coherence Value of 0.4269
Num Topics = 6  has Coherence Value of 0.4245
Num Topics = 8  has Coherence Value of 0.453
Num Topics = 10  has Coherence Value of 0.4314
Num Topics = 12  has Coherence Value of 0.4254
Num Topics = 14  has Coherence Value of 0.4446
Num Topics = 16  has Coherence Value of 0.4517
Num Topics = 18  has Coherence Value of 0.4484
Num Topics = 20  has Coherence Value of 0.4425
Num Topics = 22  has Coherence Value of 0.4296
Num Topics = 24  has Coherence Value of 0.4374
Num Topics = 26  has Coherence Value of 0.4155
Num Topics = 28  has Coherence Value of 0.4285
Num Topics = 30  has Coherence Value of 0.4135
Num Topics = 32  has Coherence Value of 0.4033
Num Topics = 34  has Coherence Value of 0.3944
Num Topics = 36  has Coherence Value of 0.403
Num Topics = 38  has Coherence Value of 0.3893
```

Therefore, we selected this optimal model and printed out the 8 topics below.

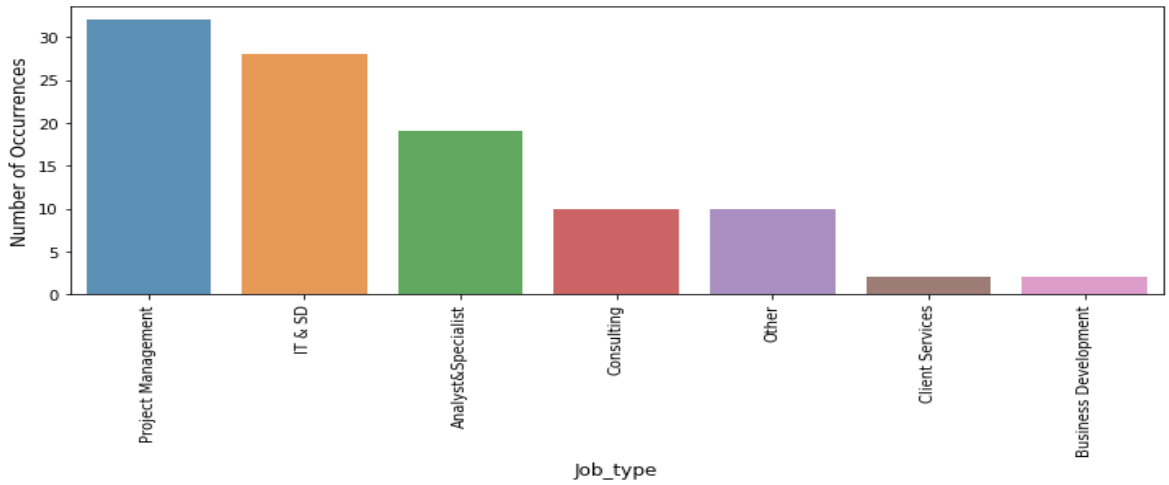
```
In [67]: # Select the model and print the topics
optimal_model = model_list[3]
model_topics = optimal_model.show_topics(formatted=False)
pprint(optimal_model.print_topics(num_words=10))
```

```
[(0,
  '0.065*"test" + 0.045*"testing" + 0.041*"web" + 0.038*"tool" + '
  '0.027*"functional" + 0.025*"engineer" + 0.024*"role" + 0.024*"build" + '
  '0.023*"developer" + 0.023*"enterprise"'),
 (1,
  '0.121*"contract" + 0.070*"proposal" + 0.042*"government" + 0.041*"task" + '
  '0.031*"order" + 0.029*"office" + 0.025*"staff" + 0.020*"agreement" + '
  '0.019*"administration" + 0.019*"corporate"'),
 (2,
  '0.097*"network" + 0.059*"server" + 0.041*"microsoft" + 0.039*"window" + '
  '0.035*"computer" + 0.022*"desktop" + 0.022*"hardware" + '
  '0.021*"infrastructure" + 0.019*"perform" + 0.019*"engineer"'),
 (3,
  '0.048*"schedule" + 0.038*"risk" + 0.032*"document" + 0.031*"change" + '
  '0.027*"review" + 0.027*"ms" + 0.025*"update" + 0.022*"quality" + '
  '0.022*"budget" + 0.021*"standard"'),
 (4,
  '0.068*"database" + 0.051*"oracle" + 0.048*"issue" + 0.035*"assist" + '
  '0.032*"production" + 0.030*"skill" + 0.026*"upgrade" + '
  '0.023*"administration" + 0.022*"set" + 0.019*"june"'),
 (5,
  '0.056*"research" + 0.054*"delivery" + 0.031*"agency" + 0.028*"case" + '
  '0.025*"institute" + 0.025*"community" + 0.024*"national" + 0.024*"college" '
  '+ 0.022*"state" + 0.019*"high"'),
 (6,
  '0.036*"cost" + 0.031*"financial" + 0.028*"account" + 0.028*"budget" + '
  '0.025*"sale" + 0.024*"director" + 0.021*"successful" + 0.020*"executive" + '
  '0.019*"reporting" + 0.018*"defense"'),
 (7,
  '0.052*"product" + 0.032*"strategy" + 0.032*"enterprise" + 0.027*"strategic" '
  '+ 0.025*"leadership" + 0.024*"company" + 0.023*"communication" + '
  '0.021*"deliver" + 0.021*"practice" + 0.018*"planning"')]
```

2. Resume Text Classification With Text Convolutional Neural Network

The second part of the resume text mining is implementing a Convolutional Neural Network model for the resume classification. The purpose of the classification is that planners in Attain could better allocate the project and tasks to the corresponding workforce regarding to their job categories.

Before building the model, same data preprocessing steps were employed here and raw resume text were cleaned through lemmatization and a new column named “Job_type” was manually created to represent the category of corresponding workforce.



The original job type included in the 103 resume files from Attain are distributed as above. For the purpose of accuracy and application. The job type in the final dataset used to train the model was simplified to three of them: Business Intelligence, IT&SD, Project Management. As for the text input for the model, the whole resume text was replaced by the most recent job history because the original resume files contain too much information that are not relevant with the job type so that it could be a noise rather than some useful information for the model to digest and produce the ideal outcome. Another thing we did to the training dataset is adding new "resume" to the job history column because there are only 70 resume files left after simplification for the three job types. 100 resume files for each category of job were collected as the final training data for the model.

```
texts = []
labels = []


for idx in range(df.Job_history.shape[0]):
    text = BeautifulSoup(df.Job_history[idx])
    texts.append(clean_str(str(text.get_text().encode())))

for idx in df['Job_type']:
    labels.append(idx)
```

```
tokenizer = Tokenizer(num_words=MAX_NB_WORDS)
tokenizer.fit_on_texts(texts)
sequences = tokenizer.texts_to_sequences(texts)

word_index = tokenizer.word_index
print('Number of Unique Tokens',len(word_index))
```

Firstly, utilizing the beautiful soup to remove some tags and unwanted characters in our pre-cleaned resume text. Then transferring resume text into unique tokens.

Embedding with Goolge Glove

```
embeddings_index = {}
f = open('glove.6B.100d.txt',encoding='utf8')
for line in f:
    values = line.split()
    word = values[0]
    coefs = np.asarray(values[1:], dtype='float32')
    embeddings_index[word] = coefs
f.close()

print('Total %s word vectors in Glove 6B 100d.' % len(embeddings_index))
```

```
Total 400000 word vectors in Glove 6B 100d.
```

```
embedding_matrix = np.random.random((len(word_index) + 1, EMBEDDING_DIM))
for word, i in word_index.items():
    embedding_vector = embeddings_index.get(word)
    if embedding_vector is not None:
        # words not found in embedding index will be all-zeros.
        embedding_matrix[i] = embedding_vector

embedding_layer = Embedding(len(word_index) + 1,
                            EMBEDDING_DIM,weights=[embedding_matrix],
                            input_length=MAX_SEQUENCE_LENGTH,trainable=True)
```

Next thing is word embedding with Google Glove which is a pre-trained unsupervised learning algorithm for obtaining vector representations for word. Here, we basically convert all resume text in the final dataset into sequences of word indices and prepare an “embedding matrix” containing all the vectors in the word index and load the embedding matrix into the keras embedding layer to build a 1D convolutional neural network ending with a output of 3 job categories. (Chollet, 2016)

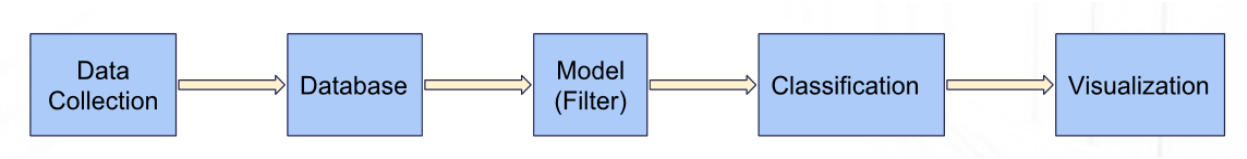
Convolutional Architecture

```
sequence_input = Input(shape=(MAX_SEQUENCE_LENGTH,), dtype='int32')
embedded_sequences = embedding_layer(sequence_input)
l_cov1= Conv1D(128, 5, activation='relu')(embedded_sequences)
l_pool1 = MaxPooling1D(5)(l_cov1)
l_cov2 = Conv1D(128, 5, activation='relu')(l_pool1)
l_pool2 = MaxPooling1D(5)(l_cov2)
l_cov3 = Conv1D(128, 5, activation='relu')(l_pool2)
l_pool3 = MaxPooling1D(35)(l_cov3)  # global max pooling
l_flat = Flatten()(l_pool3)
l_dense = Dense(128, activation='relu')(l_flat)
preds = Dense(len(macronum), activation='softmax')(l_dense)

model = Model(sequence_input, preds)
model.compile(loss='categorical_crossentropy',
              optimizer='rmsprop',
              metrics=['acc'])

print("Simplified convolutional neural network")
model.summary()
cp=ModelCheckpoint('model_cnn.hdf5',monitor='val_acc',verbose=1,save_best_only=True)
```

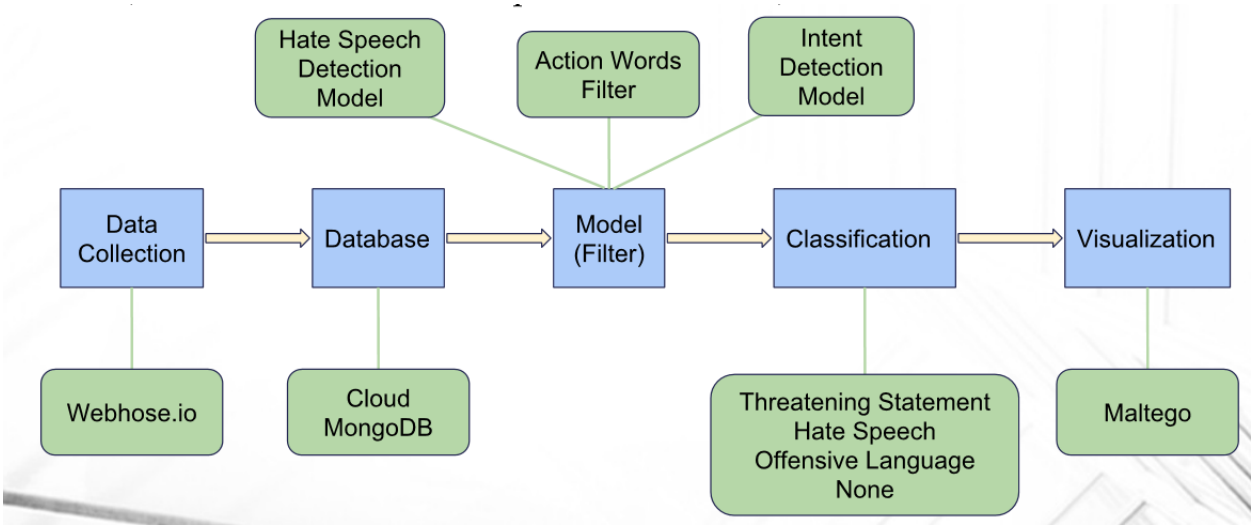
```
Simplified convolutional neural network
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 1000)              0
_________________________________________________________________
embedding_1 (Embedding)      (None, 1000, 100)         352800
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 996, 128)          64128
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 199, 128)          0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 195, 128)          82048
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 39, 128)           0
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 35, 128)           82048
_________________________________________________________________
max_pooling1d_3 (MaxPooling1 (None, 1, 128)            0
_________________________________________________________________
flatten_1 (Flatten)          (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_2 (Dense)              (None, 3)                 387
=================================================================
Total params: 597,923
Trainable params: 597,923
Non-trainable params: 0
```

Here is the architecture of the convolutional neural network, using a total of 128 filters with size 5 and set max pooling of 5 and 35.

HAMS (Human Analytics Managed Services)



In project 3, the team is developing an analytics services to classify and visualize people's opinions via custom internet search.The workflow is first crawling the data from the Internet, where client could define the search terms and data source, and then importing all data into a temporary database. Next, we build models to filter and classify the data queried from database. Finally, we would visualize the results in Maltego Application.



In our specific case, we would like to understand people's opinion about Anti-semitism and Antifa across sites and we followed the workflow of HAMS to conduct our analysis.

Model Building

Data Query and Storage

First we collected the data to build our model. We crawled posts that contained the words "Anti-semitism", "Antifa", or "jews" from different forums by using Webhose API (https://webhose.io/) in json format. After that we stored the data in the Cloud MongoDB and the database currently contains almost 40,000 posts.

Hate Speech Detection Model

We used two models to classify the posts. The first model was a hate speech detection library called HateSonar. Taking texts as input, HateSonar would provide the probability of the text being offensive language, hate speech or neither, and also provide the most possible category the text would fall into. For the categories of neither and offensive language, we just accepted the prediction provided by HateSonar. However, for hate speech, which is the category of interest,

we only took posts that had probability higher than 0.5 of being hate speech as hate speech and the rest as offensive language. At this stage, we had 4692 posts classified as hate speech.

Action Words Filter

To further look into the posts classified as hate speech, we splitted the posts into two subsets using a filter consists of 153 action words associated with crime or terrorism, such as abuse, arrest, burn, kill, etc. We believed that the posts that contained these action words would be more likely to contain threatening statements. After filtering, we had 717 posts with these action words, and 3975 posts without action words.

Threatening Statement Detection Model

Next, we would build a model that could detect a post that contains threatening statement. The 3975 posts without the actions words would be classified as hate speech and labeled as 0, and the 717 post with the actions words would need further investigation. We read all these 717 and labeled the ones with threatening statements as 1 and the rest as 0. We ended up having 29 posts labeled as 1 and 4663 posts labeled as 0, and all these 4692 posts would be used to build the Threatening Statement Detection Model. After splitting these posts into train and test sets with a ratio of 7:3, we have 3355 posts in train dataset and 1337 posts in test dataset.

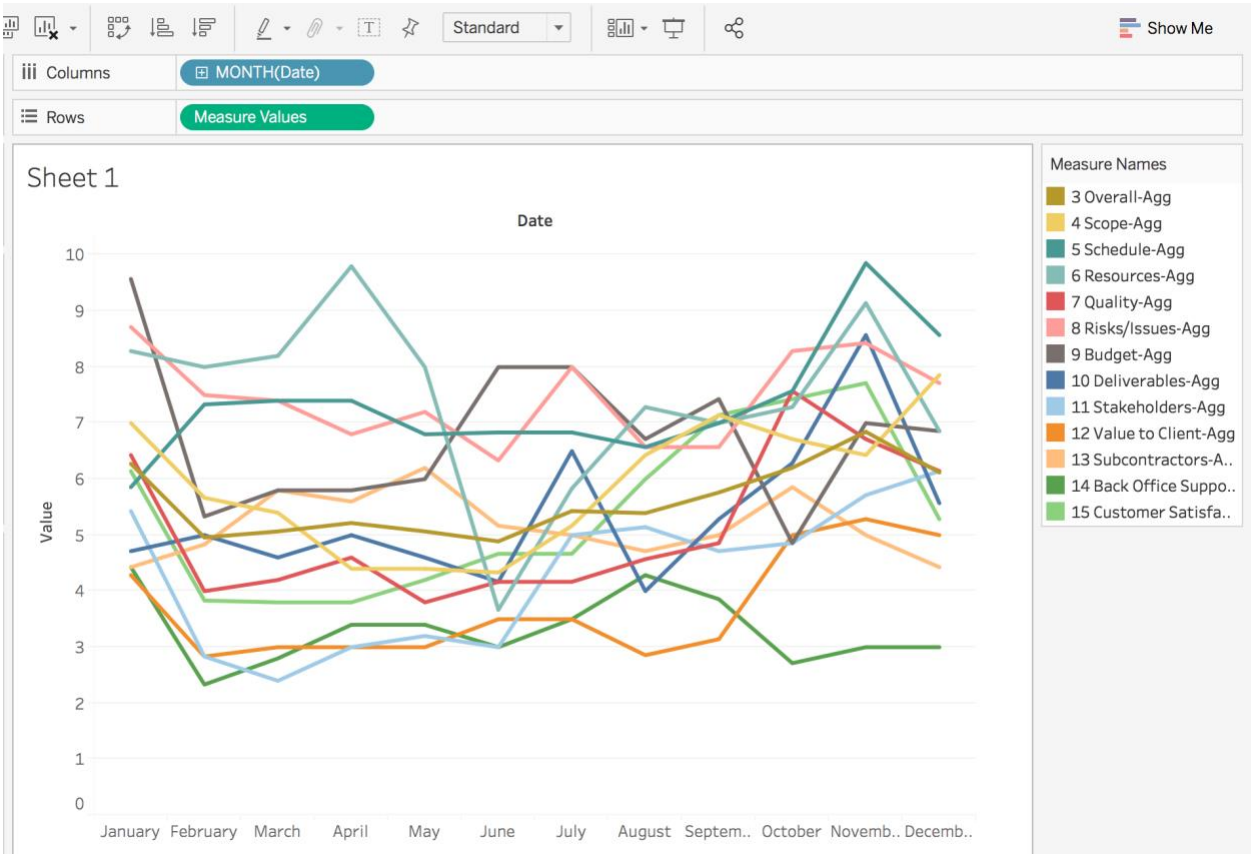
Then, we used TfidfVectorizer to transform text in posts into a sparse matrix of TF-IDF features. This transform consists two parts character vector and word vector, and ngram of character vector is from 1 to 4 which could see a set of word as a whole vector.
Last, we tried different models and the H2OAutoML performed the best in terms of AUC score, so we trained a model with H2OAutoML and so far the AUC of this model is 0.93.
We applied this model in a total new dataset and 3 out of 1854 posts are labeled as threatening statement; for instance, “we come in piece. We\'re here to help you k**l the Jews.” this text of post is correctly labeled as threatening statement; however, “the usa want to k**l all nativ americans and hitlers germany want to kill all jews i dont see a difference” this text of post is labeled as threatening statement falsely. Since we want to see a high precision which means we do not want to miss any threatening statement, although some non-threatening statements are classified as threatening statement incorrectly. With this filter, we can classify posts not only has hate speech but also contains intentions. The last part is visualization. The purpose of this part is to see whether there is a pattern among posts with hate speech and intentions.

Risk data

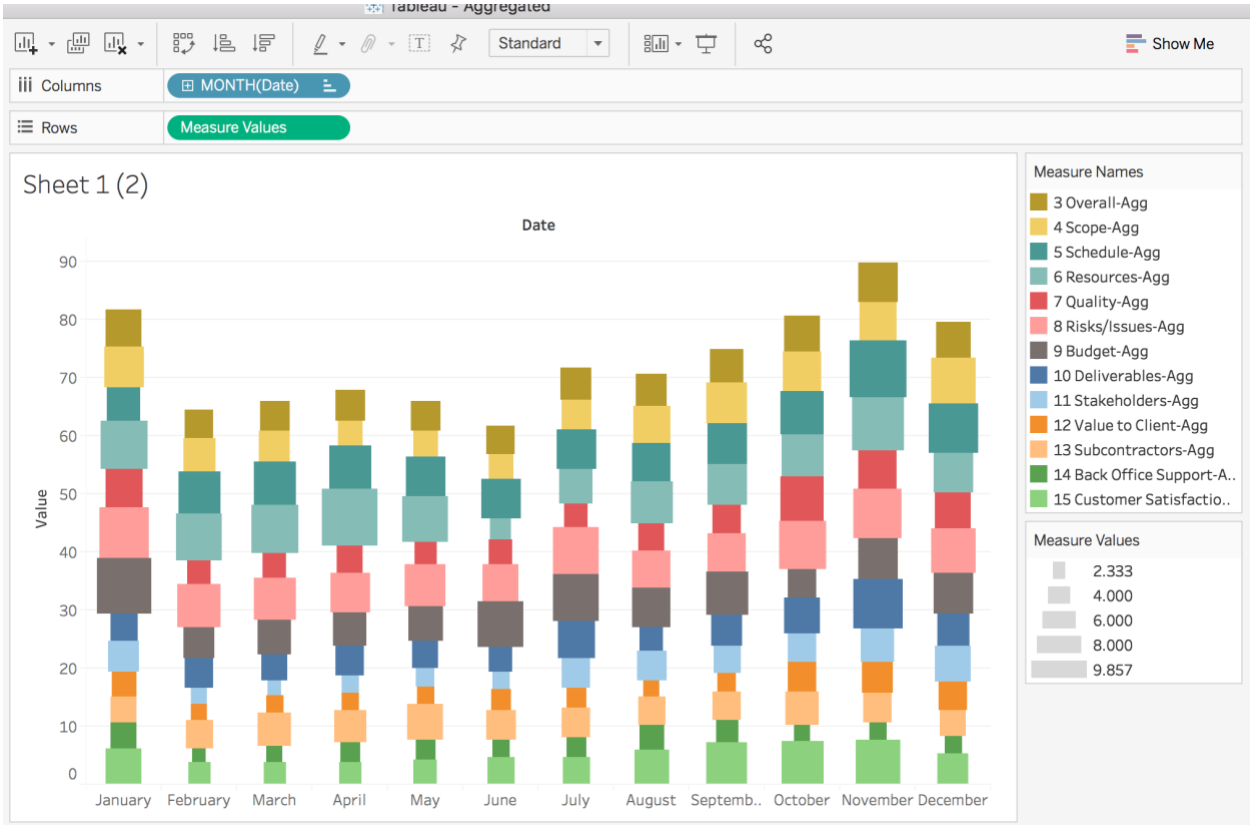
We examined Attain risk data to understand if it is possible to predict recurring risks from the risk factors provided. The risk data we got has the probability, impact, and criticality for 8 different projects over time and each project has the same risk factors such as resources and quality. Here, the criticality equals probability times the impact. We want to see if there is a pattern in the risk data and those risks can be predicted by a model. Therefore, the company can know on which month the risk for the project will increase and be well prepared for those risks.

1. Descriptive analysis

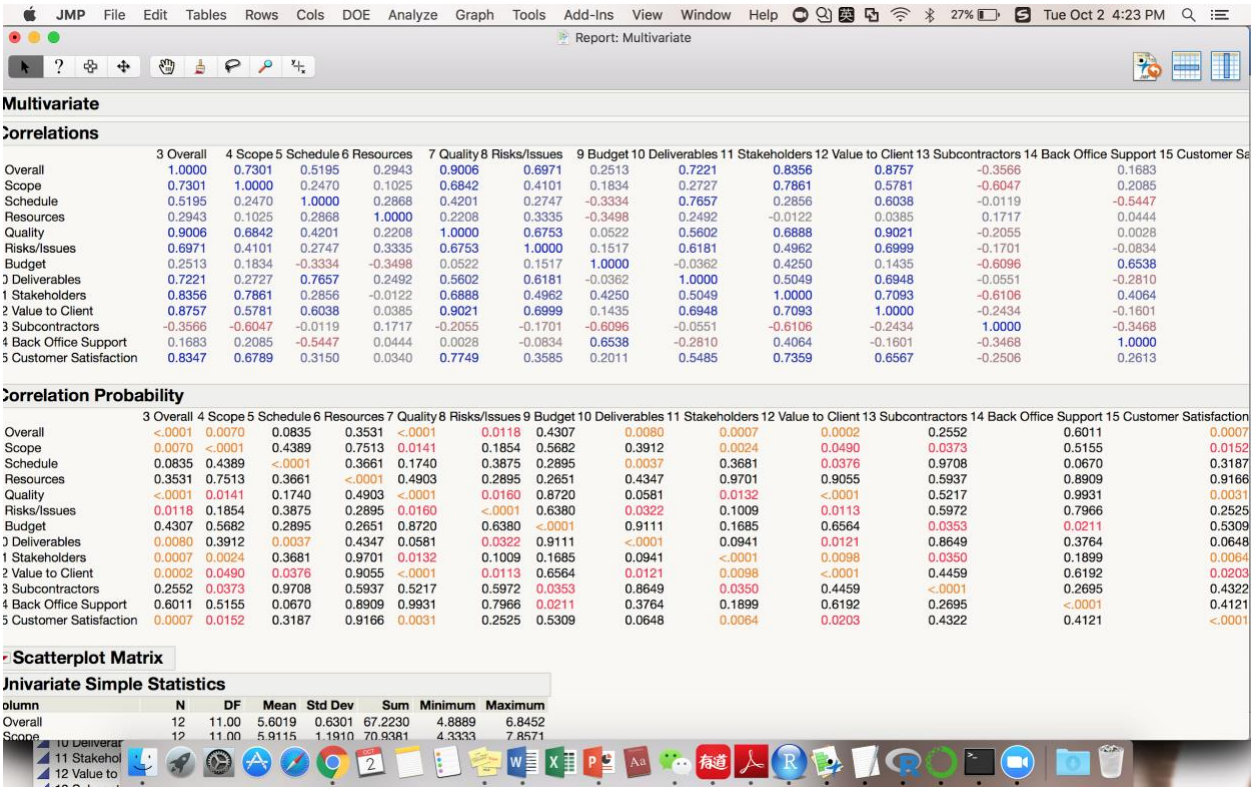
For descriptive analysis, we utilize aggregated values of each factor for plotting trending lines across all months. Specifically, we preprocess average of each factor for every same month across all projects. Then we could see in general the trending fluctuation of each factor over time. According to the result, we found, in general, project risk getting worse in winter focusing in November, December, and January. On the contrary, May, June, and February are generally likely to have a smooth project going.



Also, we would like to show the percentage of each factor accounting for the total every month. So we also visualized aggregated bar plot over time. According to the plot shows, the highest level of criticality is 9.857 and the lowest level of criticality is 2.333. Besides, Budget and Resource are generally the most dangerous factors which we should pay more attention to.



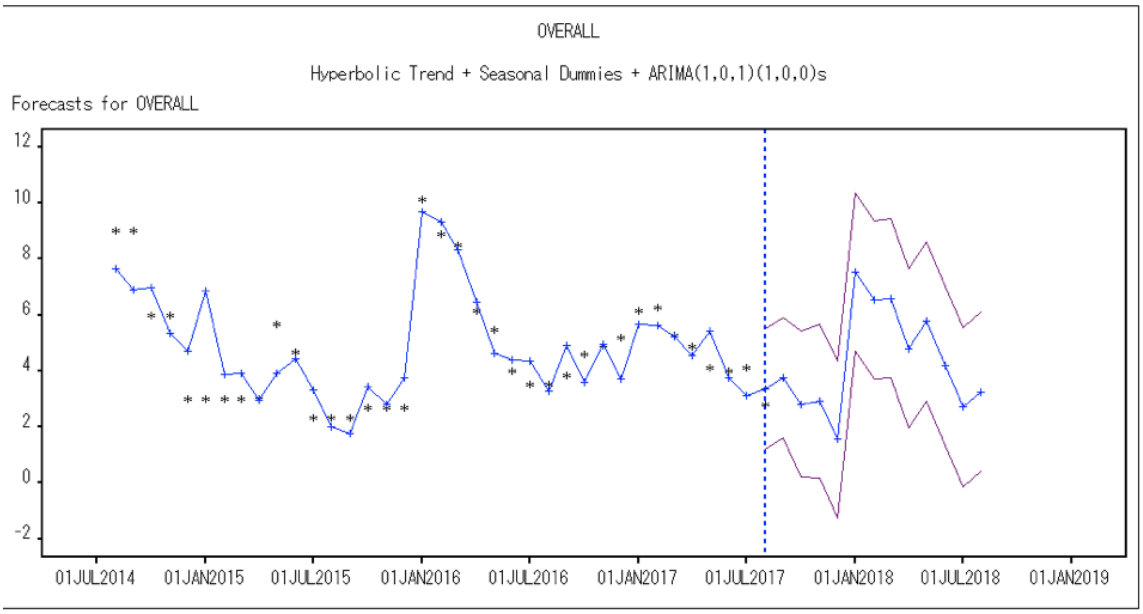
In addition, we are also curious about the multicollinearity of these measure factors. So we also conduct correlation analysis by a binary table as well as the statistical probability to see how strong their positive or negative relationships.

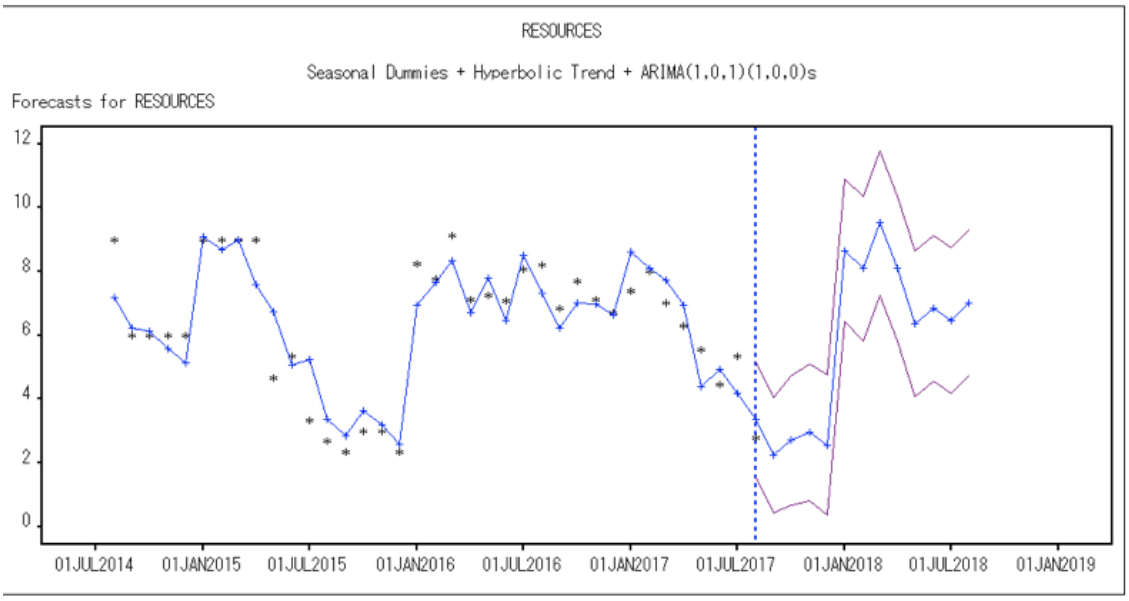


2. Time Series Predictive Analysis

We have the criticalities for 8 different projects over time and each project has the same 13 risk factors such as resources and quality. We wanted to see if there was a pattern in the risk data and those risks could be predicted by a model. But the time series models for the overall risk and risks for different projects we got had a low R square. Therefore, we made some improvements for the models such as adding seasonal dummies, different kind of trends and ARMA model for

seasonals. Here are the plots for two time series models as examples.





Results

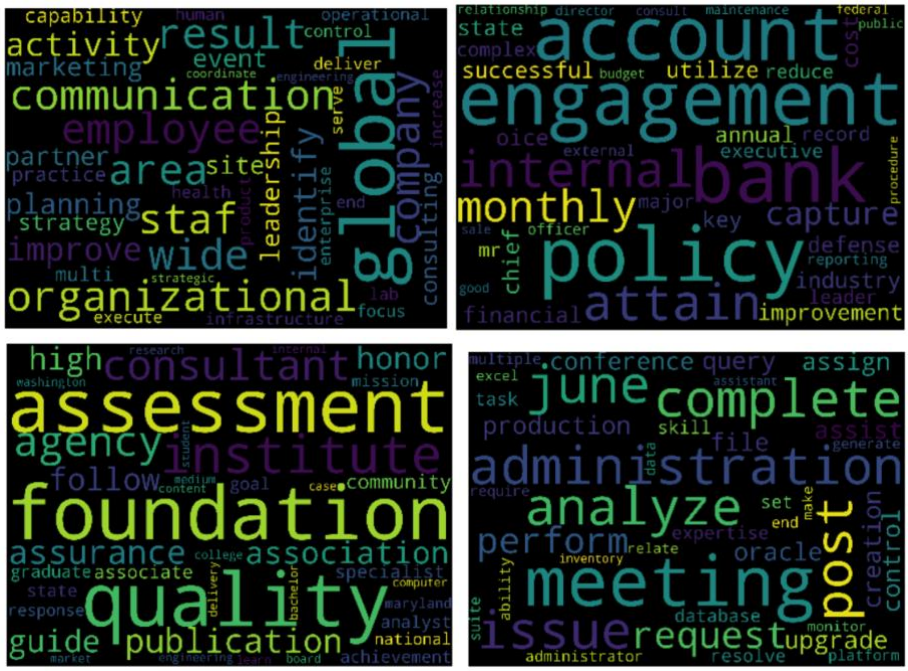
Resume Text Mining

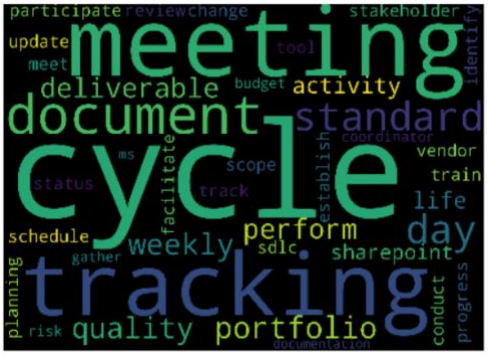
1. Resume Topic Modelling with LDA model.

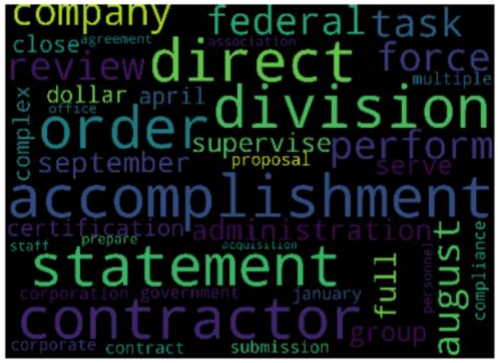
With the help of the optimal LDA model, 8 topics from the resume files were concluded and we could demonstrate the words in the 8 topics using wordcloud. The visualizat show the frequency of the words in each topic that we get from the optimal LDA model, which can give us a clear view of the frequency of the words. Here is the output of the wordcloud for the 8 topics respectively.

```
In [83]: topic1=optimal_model.show_topics(num_words=40,formatted=False)[0][1]
         topic1=[x[0] for x in topic1]
```

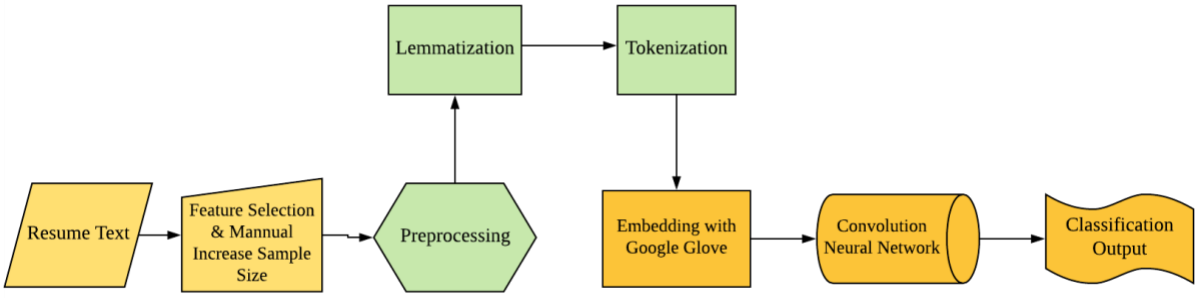
```
In [84]: # Generating the wordcloud with the values under the category dataframe
         from wordcloud import WordCloud, STOPWORDS
         firstcloud = WordCloud(
                                   stopwords=STOPWORDS,
                                   background_color='black',
                                   width=2500,
                                   height=1800
                                  ).generate(" ".join(topic1))
         plt.imshow(firstcloud)
         plt.axis('off')
         plt.show()
```



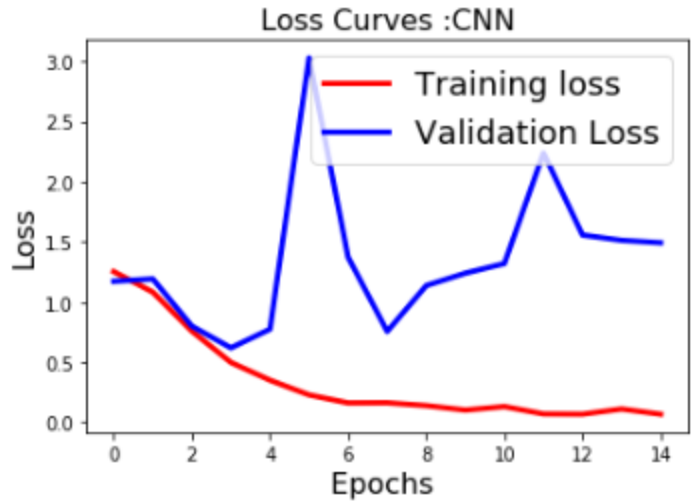


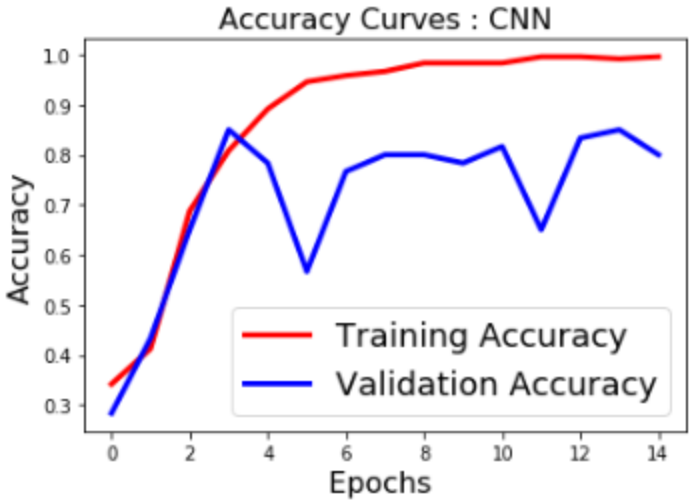


2. Resume Text Classification



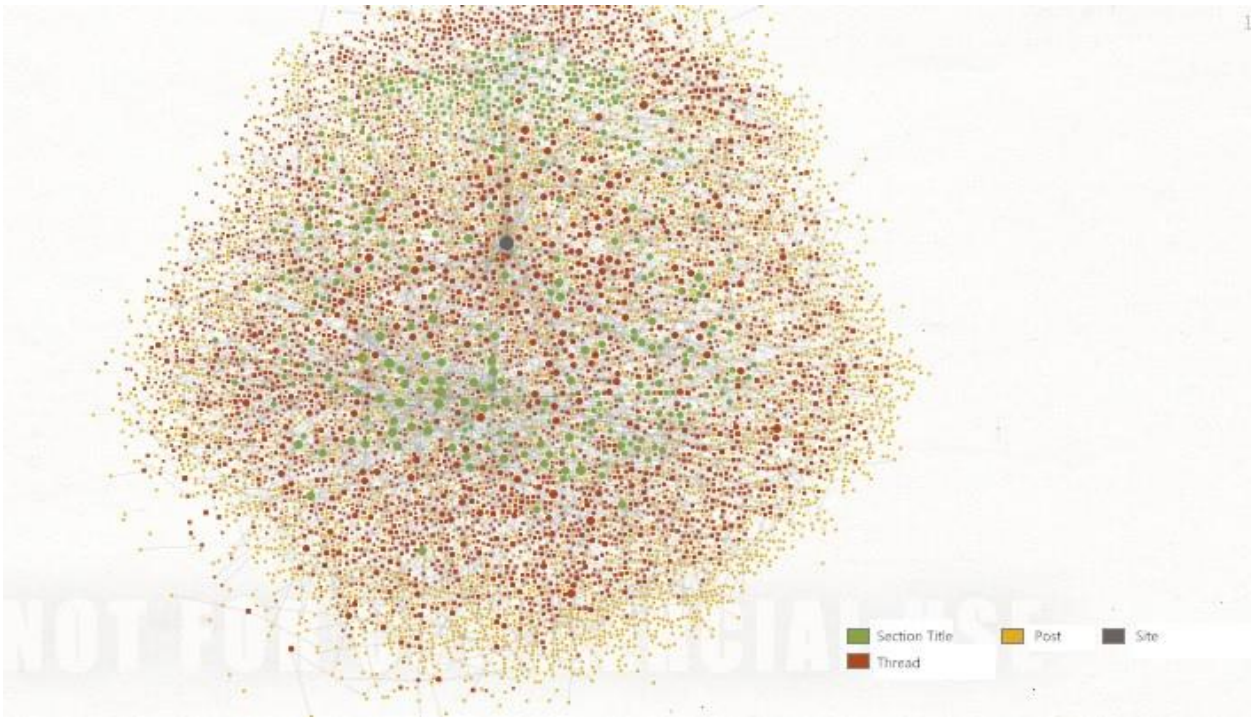
As for the resume classification, the convolutional model reached a accuracy of 0.85 after validation. The loss curves and accuracy curves are shown below.





HAMS (Human Analytics Managed Services)

Visualization



The gray nodes are sites like “reddit.com”, “4chan.org”, “freerepublic.com”, etc. In this case, the site is only “reddit.com”. Next level of green nodes are section titles (subreddit) under different sites for instance, “STRAYING OUTSIDE OF THE STATUS QUO IS FOR MELVINS”, etc. The red inferior nodes are different threads under section titles like “Centrist conspiracy theorists”, “the meaning of christmas”, etc. The outest yellow nodes are posts under different threads.

Risk Data

As we expected, the R square increased a lot, which is more than 0.7. Here are the results from SAS for the overall risk and the resources risk. For example, the resources risk factor, R Square is even more than 0.8. The R square increased and the error decreased significantly after we use the hyperbolic trend. With those descriptive analyses and time series models, we can predict the risks for each risk factor and overall factor, which can provide those engagement planners with insightful information when they are executing the projects.

Statistics of Fit

OVERALL

Hyperbolic Trend + Seasonal Dummies + ARIMA(1,0,1)(1,0,0)s

Statistic of Fit	Value
Mean Square Error	1.18436
Root Mean Square Error	1.08829
Mean Absolute Percent Error	20.56570
Mean Absolute Error	0.81479
R-Square	0.731

Statistics of Fit

RESOURCES

Seasonal Dummies + Hyperbolic Trend + ARIMA(1,0,1)(1,0,0)s

Statistic of Fit	Value
Mean Square Error	0.71181
Root Mean Square Error	0.84369
Mean Absolute Percent Error	12.36031
Mean Absolute Error	0.66160
R-Square	0.839

Conclusion

The upfront work, in the business cycle, that GW is doing will allow Attain planners to better source jobs. Improvement to both the service sets planned, and to the visibility into specific staffing skills will allow Attain to more precisely respond to customer's desired work requests. Likewise, the risk analysis done by GW will give Attain consultants more visibility into the risks they need to watch for as they execute consulting engagements. Both of these capabilities make use of AttainAnswers, and its new analytic capabilities, a strong addition to the consultants' ability to deliver top quality work.

The development of an analytics services, Human Analytics Management Service (HAMS), can be further improved from two aspects. One way is to increasing the size of threatening statements in the train dataset, helping model better learn threatening statements posts. On the other hand, we could add more features in data preprocessing like POS tagging in natural language toolkit library which marks up a word in a text as corresponding to a particular part of text, based on both its definition and its context. By adding more features, model could better grasp the pattern in different texts.

(The development of an analytics capability from scratch, Human Analytics Management Service (HAMS), provided GW students an opportunity to design & build an analytics service from the ground up. Additionally, it allows Attain to prototype a new business model via Managed Services. This service is timely because it focuses on how customers can use big data analytics to find instances of hate speech that may trend toward violent acts.)

All of the projects share a common pedagogical thread, the data and opportunity to design & manipulate models iteratively, in order to understand how models settle down into the optimal combination of submodules. For example, there were many different ways that Attain risks were analyzed. Having the benefit of modeling risks in several different configurations allows GW students to understand the best model for understanding risks. Students analyzing Attain staff & designing HAMS found similar benefits. Creating a repurposable AI model to process Attain staff resumes allow Attain to reuse the analysis model as new staff comes aboard. Similarly students combining internet data services, intermediate repositories, and filters allows HAMS to be a modular set of code that, itself, can be dynamically altered for separate data sources, search terms and filters. In each case, GW students get good experience, via model theories or trial & error, experimenting with various model iterations to see how models change in various Configurations.

Appendix

A. Topic Modelling Results

Finding the dominant topic in each resume files

	Document_No	Dominant_Topic	Topic_Perc_Contrib	Keywords	Index	Text	Job_type
0	0	2.0	0.6088	product, enterprise, schedule, risk, quality, ...	1	Technical IT Agile Project Manager / Scrum Mas...	Project Management
1	1	2.0	0.3509	product, enterprise, schedule, risk, quality, ...	2	Certified Project Management Professional (PMP...	Project Management
2	2	5.0	0.2689	research, delivery, health, communication, dir...	3	Mr. Agee serves as a Partner and Board Member ...	Other
3	3	5.0	0.3509	research, delivery, health, communication, dir...	4	Ms. Morin-Kensicki brings over 20 years of pro...	Project Management
4	4	1.0	0.3749	network, server, computer, microsoft, window, ...	5	Information Technology Professional with six y...	IT & SD
5	5	2.0	0.3225	product, enterprise, schedule, risk, quality, ...	6	Mr. Thomas Pagurek\n\nTechnical Services Manag...	IT & SD
6	6	3.0	0.2573	contract, proposal, task, government, federal,...	7	I'm looking to forward my career, which focuse...	Project Management
7	7	1.0	0.2844	network, server, computer, microsoft, window, ...	8	Attain LLC\nChief Engineer & Infrastructure Op...	IT & SD
8	8	4.0	0.2950	analyst, assist, document, update, skill, coor...	9	Chesapeake Bay Program | Annapolis, MD\t(05/20...	Analyst&Specialist
9	9	4.0	0.3233	analyst, assist, document, update, skill, coor...	10	Mary Baldwin University Library Assistant, Gra...	Analyst&Specialist
10	10	4.0	0.3397	analyst, assist, document, update, skill, coor...	11	HIGHLY ACCOMPLISHED MANAGEMENT PROFESSIONAL WI...	Project Management
11	11	3.0	0.2782	contract, proposal, task, government, federal,...	12	•\t10+ years of IT consulting experience in pr...	Project Management
12	12	0.0	0.3790	database, test, tool, oracle, testing, web, pr...	13	Senior Quality Assurance Tester and Performanc...	IT & SD
13	13	1.0	0.4134	network, server, computer, microsoft, window, ...	14	B.S. Information Technology, East Carolina Uni...	IT & SD
14	14	4.0	0.2911	analyst, assist, document, update, skill, coor...	15	Published article "Providing Value through Imp...	Consulting
15	15	4.0	0.2157	analyst, assist, document, update, skill, coor...	16	Mr. Kallum Gagnier is a graduate from Oregon S...	Analyst&Specialist
16	16	1.0	0.4499	network, server, computer, microsoft, window, ...	17	Mr. Chris A. Christopher started his Informati...	Consulting
17	17	2.0	0.2527	product, enterprise, schedule, risk, quality, ...	18	Involved in the many aspects of delivering Sal...	Consulting
18	18	3.0	0.3313	contract, proposal, task, government, federal,...	19	PROFESSIONAL SUMMARY\nDeadline-driven proposal...	Project Management
19	19	5.0	0.2563	research, delivery, health, communication, dir...	20	Ms. Mary E Pouleson\nI am the UX Manager at th...	Project Management

Finding the most representative sentences from each topic

	Topic_Num	Topic_Perc_Contrib	Keywords	Index	Text	Job_type
0	0.0	0.6861	database, test, tool, oracle, testing, web, pr...	60	Oracle Certified Professional with over 17 yea...	IT & SD
1	1.0	0.5974	network, server, computer, microsoft, window, ...	22	PROFESSIONAL EXPERIENCE\nATTAIN LLC, Senior Co...	Consulting
2	2.0	0.6088	product, enterprise, schedule, risk, quality, ...	1	Technical IT Agile Project Manager / Scrum Mas...	Project Management
3	3.0	0.7624	contract, proposal, task, government, federal,...	54	Mr. Spotz is an experienced National Contract ...	Project Management
4	4.0	0.3748	analyst, assist, document, update, skill, coor...	29	Accounting professional with approximately ten...	Other
5	5.0	0.6511	research, delivery, health, communication, dir...	27	Ms Deepa Bedi, has been part of the Salesforce...	Client Services

Bibliography

Bird, S. (2016). Natural language processing with python. Place of publication not identified: OReilly Media.

Britz, Denny. “Understanding Convolutional Neural Networks for NLP.” *WildML*, 10 Jan. 2016, Retrieved from www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/.

Chollet, Francois. “The Keras Blog.” *The Keras Blog ATOM*, 2016, Retrieved from blog.keras.io/using-pre-trained-word-embeddings-in-a-keras-model.html.

Conneau, Alexis, Schwenk, Holger, Barrault, Lecun, & Yann. (2017, January 27). Very Deep Convolutional Networks for Text Classification. Retrieved from https://arxiv.org/abs/1606.01781

Ignatow, G., & Mihalcea, R. (2017). Text mining: A guidebook for the social sciences. Los Angeles: SAGE

Kim, & Yoon. (2014, September 03). Convolutional Neural Networks for Sentence Classification. Retrieved from https://arxiv.org/abs/1408.5882

Prabhakaran, S. (2018, March). Topic Modeling with Gensim Retrieved from https://www.machinelearningplus.com/nlp/topic-modeling-gensim-python/

Susan, Li. (2018, May 31). Topic Modeling and Latent Dirichlet Allocation (LDA) in Python Retrieved from https://towardsdatascience.com/topic-modeling-and-latent-dirichlet-allocation-in-python-9bf156893c24

Thomas Davidson, Dana Warmsley, Michael Macy, & Ingmar Weber. (2017). "Automated Hate Speech Detection and the Problem of Offensive Language." ICWSM.

Zhang, Ye, Wallace, & Byron. (2016, April 06). A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification. Retrieved from https://arxiv.org/abs/1510.03820