

CSC 405/605 Fall 2019

Budget Text Analysis
Project Review Document
Version 3.0

Prepared By;
Naseeb Thapaliya
Akash Meghani
Unnati Khivsera
Sultan Al Bogami
Miguel Gaspar Utrera

Tasks

Sultan Al Bogami

1. Reconstructed the data hierarchy for the project:
 - a. Collected the budget documents from 2019-2008.
 - b. Compressed them to reduce the size of the files.
 - c. Converted the PDF files into CSV files.
 - d. Standardized the naming scheme of the data files.
 - e. Organized the data into folders.
 - f. Uploaded the data to the project repository.
2. Preprocessed the data for FY2020:
 - a. Convert the csv files into data frames.
 - b. Label the data frames adding “organization” column.
 - c. Combine the data frames and insert a “year” column.
 - d. Drop unnecessary columns in the combined data frame.
 - e. Perform text normalization.
 - f. Produces a single csv file for the year in question.
3. Started the exploratory data analysis.

Miguel Gaspar Utrera

1. Did some analysis on the budget for Durham city for 2019 and 2020
2. My goal was to see how similar the two budget documents were
3. Using NLTK, genism and TFID I was able to use the most frequent words from each document and see their similarity

4. They are 57% similar, I can conclude the budget documents are reused and that is why they are very similar

Unnati Premchand Khivasara

Tasks: My main tasks for this project were to do sentiment analysis on the budget documents, compare the changes and spread of sentiments over budget documents for multiple years and for different cities and counties.

1. Data Extraction and Statistical Evaluation - The csv files created using a tool, were loaded in dataframes. This data was evaluated to get the hold of data for the budget text of different years and cities and associated sentiments.(3 hours)

2. Distribution Modeling - (4 hours)

https://github.com/UNCG-CSE/Budget_Text_Analysis/blob/master/src/EmotionAnalysis/Distribution.ipynb

- a. The sentiments of Guilford County(2008) were plotted using Histogram and KDE showed that the distribution is of type Poisson. Also KStest proved that Poisson Distribution fits well(pValue=0.99).

- b. The sent-count(count of occurrence of each words) values were plotted to see the distribution of Charlotte City Budget Document. The distribution proves to be of type Poisson using KDE and KStest(pvalue=1.0)

3. Hypothesis Testing - The budget documents should have variance in sentiments over the years, or in difference cities/counties based on the varying economies. The Hypothesis results shows that consecutive years will have more similarity in sentiments and also does not differ to a great extent over a long period. (4 hours)

https://github.com/UNCG-CSE/Budget_Text_Analysis/blob/master/src/EmotionAnalysis/HypothesisTesting.ipynb

- a. H_0 : The sentiments for Charlotte Document 2008 and 2020 are same
 H_1 : The sentiments for Charlotte Document 2008 and 2020 are not same

Two sample test was performed to prove this Hypothesis with threshold 0.05.

Result : pvalue = 0.28 so I accept a null Hypothesis.

b. H0 : The sentiments for Raleigh Document 2014 and 2015 are same

H1 : The sentiments for Raleigh Document 2014 and 2015 are not same

Two sample test was performed to prove this Hypothesis with threshold 0.05.

Result : pvalue = 0.98 so I accept a null Hypothesis.

4. Observing the order of Sentiments - For all counties/cities budget documents the frequencies of all sentiments and emotions were obtained. And it was observed that for each document Positive sentiment value was highest and Disgust occurred with least number of frequencies. Also the plot for comparison shows that when seen in descending order every city shows much similarity. (3 hours)

https://github.com/UNCG-CSE/Budget_Text_Analysis/blob/master/src/EmotionAnalysis/CountingWordsFrequency.ipynb

5. Sections Sentiment Coverage - If the distribution of sentiments is observed over pages of a budget document, it clearly shows that for specific sections the concentration of positive and negative sentiment is much higher. Although positive and negative sentiments are seen all over document text like Funds and Services. While the emotions are not seen in few sections at all. (2 hours)

https://github.com/UNCG-CSE/Budget_Text_Analysis/blob/master/src/EmotionAnalysis/Statistics.ipynb

4. Machine Learning -Question Asked: How to create a model that can classify and predict sentiments for future documents?

Initially steps of data-clean, preprocess, assigning affinity scores, features creation and vector tokenization (using TF/IDF) is done. The data is split into train and test data, and supervised algorithms as LogisticRegression, RandomForestClassifier and LinearSVC were used to create a model.

X -> feature vectors formed out of sentences of a Guilford County funds section(2008)

Y -> Classified sentiments

https://github.com/UNCG-CSE/Budget_Text_Analysis/blob/master/src/EmotionAnalysis/MachineLearning3.ipynb

Observation : RandomForestClassifier and LinearSVC classification models proves to predict and classify the sentiments of test data than model using LogisticRegression. These both models predict the results with same accuracy as well as rmse values. While if the proportion of train vs test data is changed the LinearSVC does not show the same accuracy as that of RandomForestClassifier, whose accuracy remain highest and Root mean square error value is lowest.(8 hours)

Akash Meghani

Emotional and sentiment analysis

1) Data exploration and statistical evaluation of data:

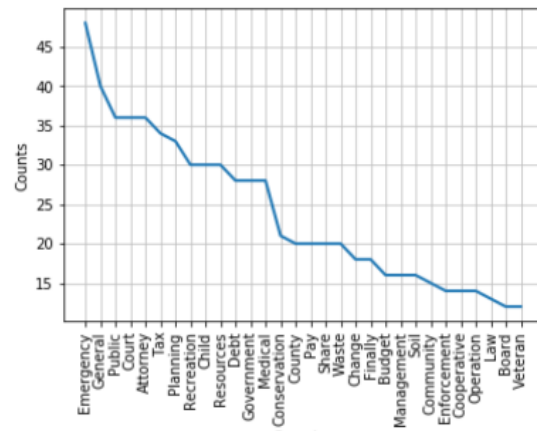
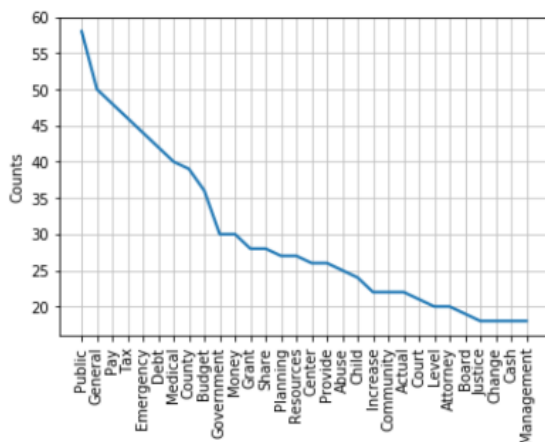
Below picture shows most influential words in Guilford county budget document in 2008 and 2020. For example, public was the most influential word in 2020 but in 2008 during recession emergency word was used the most.

Most Influential Words in Guilford County (2020 and 2008)

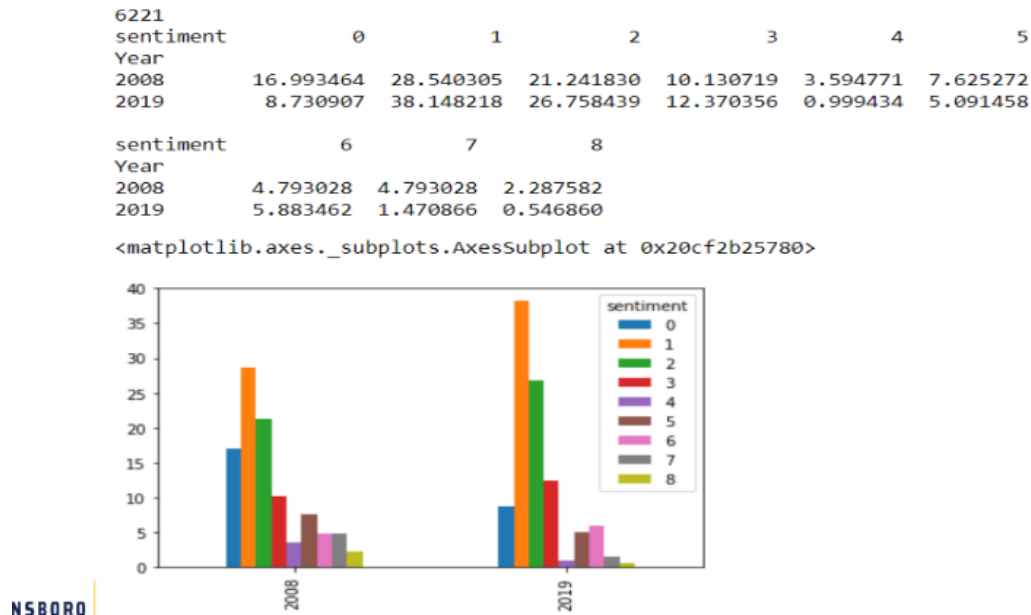
```
[('Public', 58),
 ('General', 50),
 ('Pay', 48),
 ('Tax', 46),
 ('Emergency', 44),
 ('Debt', 42),
 ('Medical', 40),
 ('County', 39),
 ('Budget', 36),
 ('Government', 30)]
```

```
[('Emergency', 48),
 ('General', 40),
 ('Public', 36),
 ('Court', 36),
 ('Attorney', 36),
 ('Tax', 34),
 ('Planning', 33),
 ('Recreation', 30),
 ('Child', 30),
 ('Resources', 30)]
```

Distribution of most influential Words in Guilford County (2020 and 2008)



Distribution of Emotions in General fund summary section (2020 and 2008) for Charolette



Time taken to perform the task: Around 4 to 5 hours

Link to the Ipython notebook : https://github.com/UNCG-CSE/Budget_Text_Analysis/blob/master/src/EmotionAnalysis/Budget_Analysis_Emotions1.ipynb

Results explained:

The graph above shows that difference of emotions between 2008 and 2019. This experiment was to show that there was a lot of difference in emotions in 2008 (During Recession) compared to 2019. The graph shows that positivity has increased in 2019 and negativity has decreased.

2) Hypothesis Testing:

H0 -> The sentiments remain same for service part from 2008 and 2020 in Charlotte city document.

H1 -> Sentiment changes for service part from 2008 to 2020 in charlotte city document
To prove this Hypothesis two sample is performed and p-value threshold is $p = 0.05$

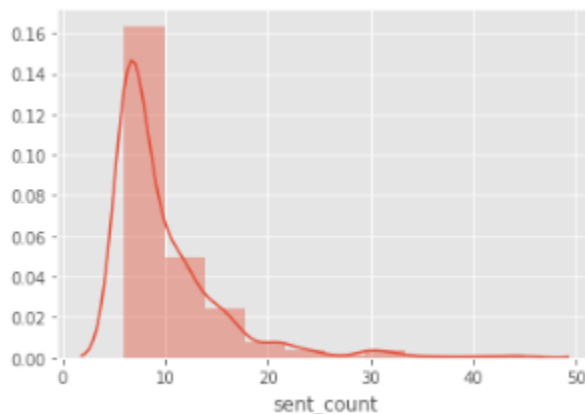
Time taken to perform the task: Around 2 hours

Result:

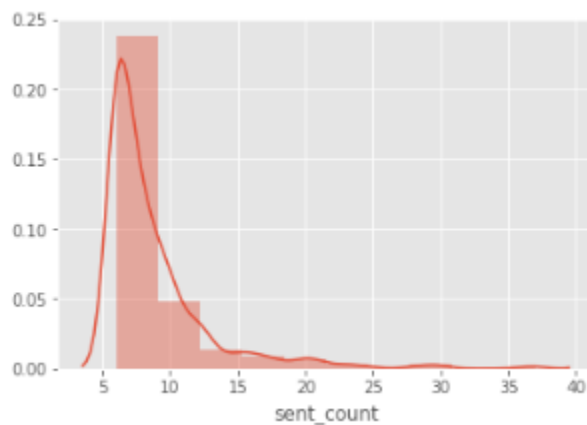
After applying the test, we found out that P-Value is greater than threshold (0.56) therefore we were failed to reject null hypothesis.

3) Probability Distribution:

I have concatenated Guilford county, Durham county, Durham city, charlotte city, Raleigh city:
Took negative sentiment counts (at least more than 5 times)



Took positive sentiment counts (at least more than 5 times):



Time taken to perform the task: Around 2 hours

Link to the Ipython notebook : https://github.com/UNCG-CSE/Budget_Text_Analysis/blob/master/src/EmotionAnalysis/Combined%20Distribution.ipynb

Result:

I applied K-test and found out that the p-value is 1.0 for poisson distribution, it means Poisson distribution is fitting well in this.

4) Machine Learning:

Reforming data and data exploration:

Data given to us was in single word therefore it was not making sense for us to apply machine learning on top of it. I have parsed the data again and reformed it in a data frame. Here are the steps used to reform the data:

- 1) Parsed the pdf file with the help of Tika parser.
- 2) Converted it to string
- 3) Broke the data into sentences
- 4) Data cleaning (Removing unwanted characters, Tokenization, Removing stop words, Lemmatizing/Stemming.)
- 5) Dropped the rows which are empty
- 6) Used Affin library from python to assign Affin values
- 7) Assigned the sentiments accordingly

New data frame looks like:

	text	afinn_score	emotion
0	General revenues projected rebound from econom...	0.0	1
1	City continues face limitations balancing prio...	-1.0	0
2	However City employees continue work hard prev...	-2.0	0
3	Examples prior year reductions listed below	0.0	1
4	complete listing unfunded budget requests prov...	0.0	1

Question: My data is charlotte city funds for 2008 where according to sentiment analysis negativity is higher. We are classifying how accurately it is classifying sentiments in the document.

X = Vectorized text (vectors)

Y = emotions

Used This vectorizer which breaks text into single words and bi-grams and then calculates the TF-IDF representation.

```
RMSE : 0.28867513459481287
Accuracy : 91.67%
```

Time taken to perform the task: Around 10 hours

Link to the Ipython notebook https://github.com/UNCG-CSE/Budget_Text_Analysis/blob/master/src/EmotionAnalysis/Combined_Machine_%20Learning.ipynb

Results explained:

Here 91% times classifier was able to classify the emotions correctly and root mean square error is 0.28.

NASEEB THAPALIYA

TOTAL HOURS ALLOCATED: 20 HRS

(Please, note that both the project 2 and project 3 tasks are included in this project report.)

Tasks: The main tasks for the project were performing topic modeling with visualization for over the years, compute statistical analysis, implement machine learning and study the latent semantic structure of budget texts over the years. The tasks and their explanation in a sequential manner is shown below:

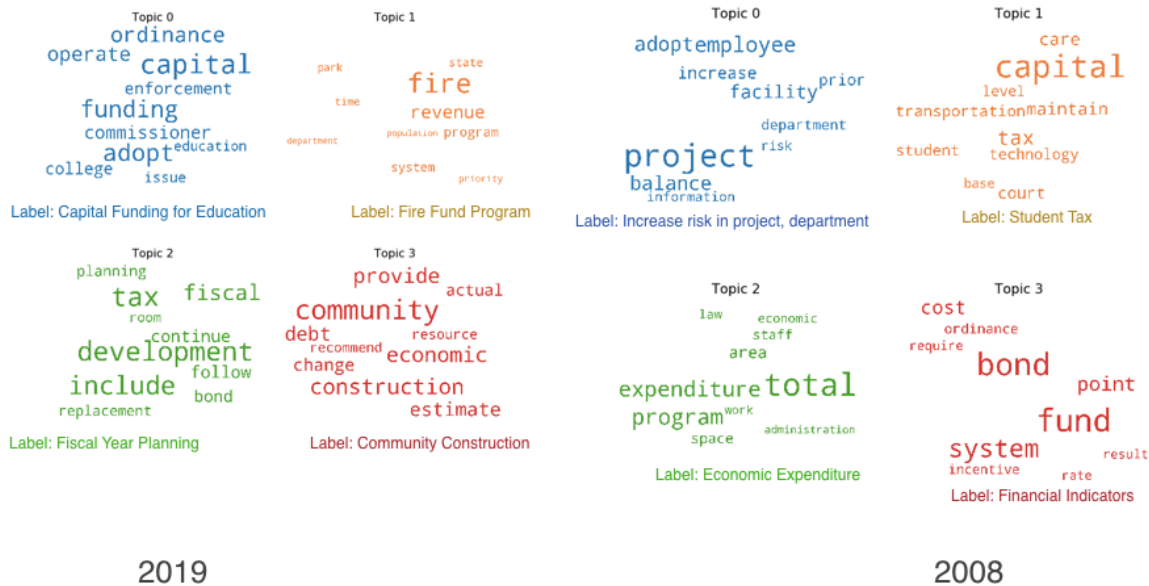
1) Perform topic modeling to get the topic distributions over the years(2008->2020)(5 hours):

Here, Latent Dirichlet Allocation(LDA) which is a very popular algorithm for topic

modeling with excellent implementations in the Python's Gensim package is used for topic modeling. The Goal was to find the topic distributions over the years, and get the number of key topics which was prevalent over the years, and compare them. Below screenshot shows the the top 10 topics we got for the budget text for year 2008 using LDA model:

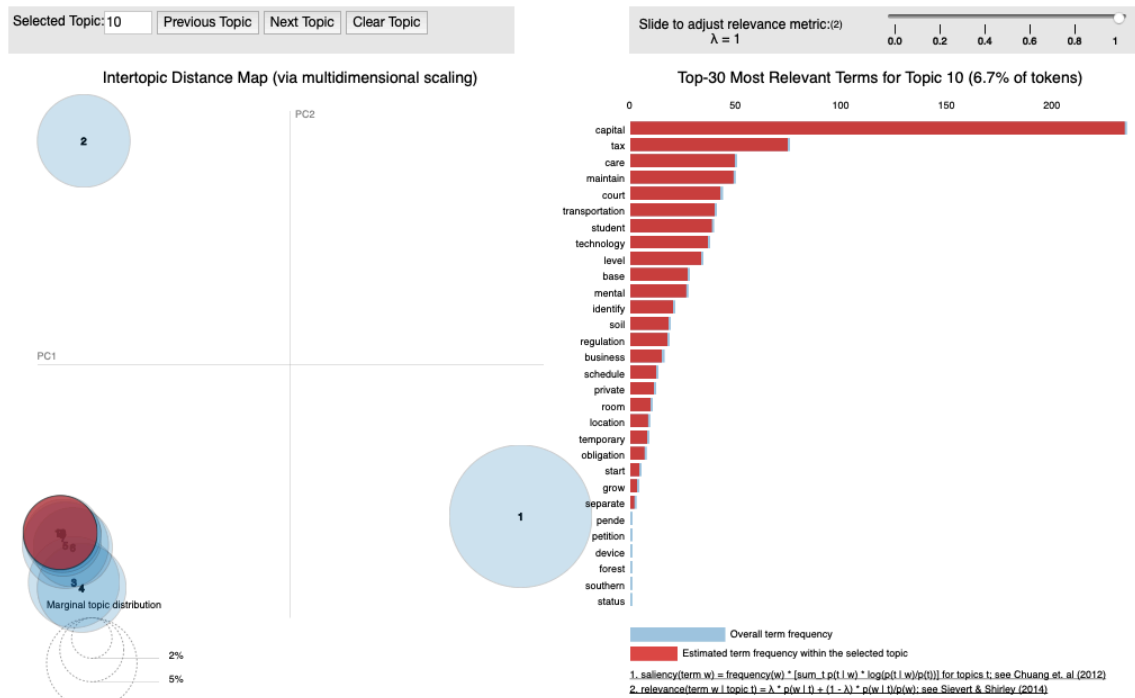
```
[0,
'0.315*total" + 0.056*commissioner" + 0.052*park" + 0.051*property" + '
'0.044*security" + 0.044*resource" + 0.035*policy" + 0.032*economic" + '
'0.027*performance" + 0.026*amend"'),
1,
'0.196*program" + 0.153*provide" + 0.106*major" + 0.064*grant" + '
'0.063*exist" + 0.053*operation" + 0.039*information" + 0.037*change" + '
'0.035*work" + 0.034*care"'),
2,
'0.115*fund" + 0.110*summary" + 0.108*fire" + 0.078*area" + '
'0.062*current" + 0.060*solid" + 0.048*state" + 0.041*level" + '
'0.040*percent" + 0.039*estimate"'),
3,
'0.187*fiscal" + 0.086*debt" + 0.074*unit" + 0.068*water" + '
'0.060*infrastructure" + 0.050*issue" + 0.044*goal" + 0.042*remain" + '
'0.042*government" + 0.041*base"'),
4,
'0.206*adopt" + 0.107*replacement" + 0.090*support" + 0.083*increase" + '
'0.075*number" + 0.044*charge" + 0.041*planning" + 0.038*additional" + '
'0.038*require" + 0.038*site"'),
5,
'0.219*capital" + 0.134*expenditure" + 0.100*management" + '
'0.072*equipment" + 0.050*balance" + 0.044*vehicle" + 0.040*begin" + '
'0.036*improve" + 0.030*identify" + 0.026*law"'),
6,
'0.242*include" + 0.164*community" + 0.095*school" + 0.075*impact" + '
'0.037*rate" + 0.033*maintain" + 0.027*recommend" + 0.027*associate" + '
'0.026*pay" + 0.024*resident"'),
7,
'0.259*year" + 0.150*funding" + 0.117*public" + 0.080*development" + '
'0.077*actual" + 0.044*plan" + 0.029*annual" + 0.024*life" + '
'0.021*address" + 0.019*help"'),
8,
'0.047*service" + 0.019*system" + 0.013*building" + 0.012*improvement" + '
'0.012*operate" + 0.011*transfer" + 0.010*cost" + 0.010*source" + '
'0.010*complete" + 0.009*future"'),
9,
'0.260*budget" + 0.207*project" + 0.180*facility" + 0.133*revenue" + '
'0.052*tax" + 0.024*appropriate" + 0.024*control" + 0.015*specific" + '
'0.014*population" + 0.011*food"')]
```

Then, the topic distribution for other years were computed and compared with the topics distribution. The comparison between topics from 2008 and 2019 are shown below:



As, we can see the top topics were 2008 and 2019 were quite different. The topics for 20 talked more about these topics: Capital funding for Education, Fire fund programs, Fiscal year planning and, and, community construction. Fire problem was one of the major problems in 2019, so the budget was allocated for it. And, the budget texts has significant mention for capital funding for education, and community construction. Also, it's no surprise that talk about tax and fiscal year was going on in a budget document. Similarly, in 2008, the topics that were more popular were about economic expenditure and financial indicators, may be due to recession. Overall, the budget documents for 2019 and 2008 was found to be quite dissimilar.

In addition to this, the topic modeling dashboard was created using pLYDavis Gensim module, as well for better view of topic distributions and analysis:



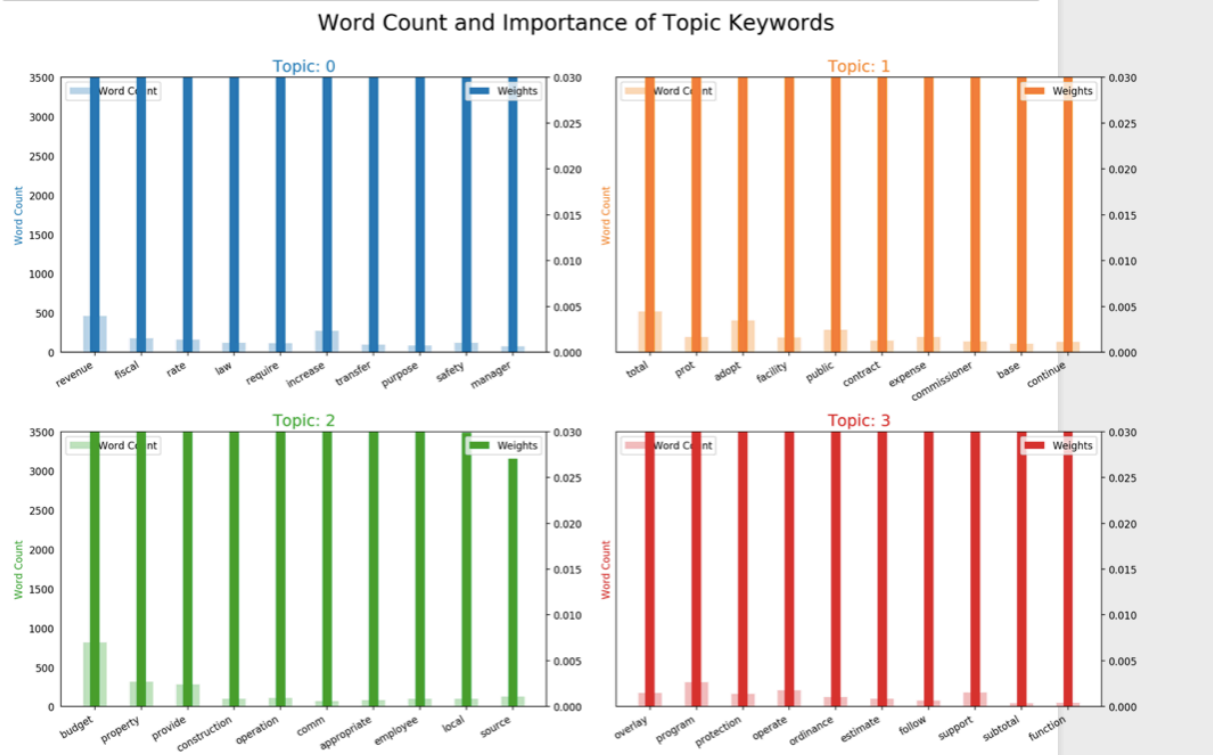
The Link to the Ipython notebook for above tasks in GitHub:

https://github.com/UNCG-CSE/Budget_Text_Analysis/blob/master/src/TopicModelling/Combined_data_Set_Topic_Modeling.ipynb

https://github.com/UNCG-CSE/Budget_Text_Analysis/blob/master/src/TopicModelling/Guilford_2008_topic_modeling.ipynb

2) Compute Statistical analysis and coherence scores(2 hours):

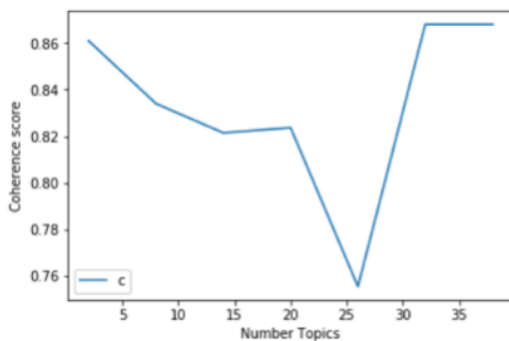
Here, the word count for the words in the budget documents were computed. And, also the number of topics in relation to the word count were computed. Here, this gives the importance of topic keywords in the entire document in terms of frequency. We can see that in the figure below:



The above figure implicates that the document is very huge with very more than 1 million words. Hence, the topic keywords importance compared to every words in the document is pretty low. For instance, one of the key words 'budget' is only mentioned about 1000 times in the document. This shows the frequency of the keywords over the topic distribution.

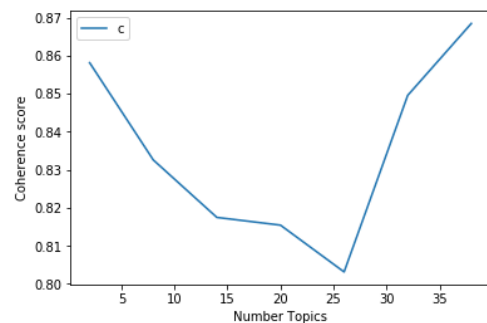
Also, I computed the topic coherence scores for 2008 and 2019. The Coherence score is simply a measure used to evaluate topic models: methods that automatically generate topics from a collection of documents, using latent variable models.

A good model will generate coherent topics, i.e., topics with high topic coherence scores. Good topics are topics that can be described by a short label, therefore this is what the topic coherence measure should capture. The figures below show the coherent scores:



Coherence Score: 0.8256146597574272

2019



Coherence Score: 0.8247949042506306

2008

The coherence score for 2019 was found to be 82.56% and the coherence score for 2008 was found to be 82.79%, from which we can implicate that as they are average coherence scores, we can accept that the topics are somewhat coherence, and we can also decide on the number of topics, to get the best distribution of the topics.

The Link to the Ipython notebook for above tasks in GitHub:

https://github.com/UNCG-CSE/Budget_Text_Analysis/blob/master/src/TopicModelling/Combined_data_Set_Topic_Modeling.ipynb

https://github.com/UNCG-CSE/Budget_Text_Analysis/blob/master/src/TopicModelling/Guilford_2008_topic_modeling.ipynb

3) Supervised Machine Learning(10 hours):

Machine learning should be performed to learn from the data, and the following question was formulated to perform the supervised machine learning and learn whether this will hold true or not:

Question:

“ Does a topic model for one year can identify the latent semantic structure that persists over time in this budget text domain ?”

The methodology adopted to solve the above question is shown by the diagram below:

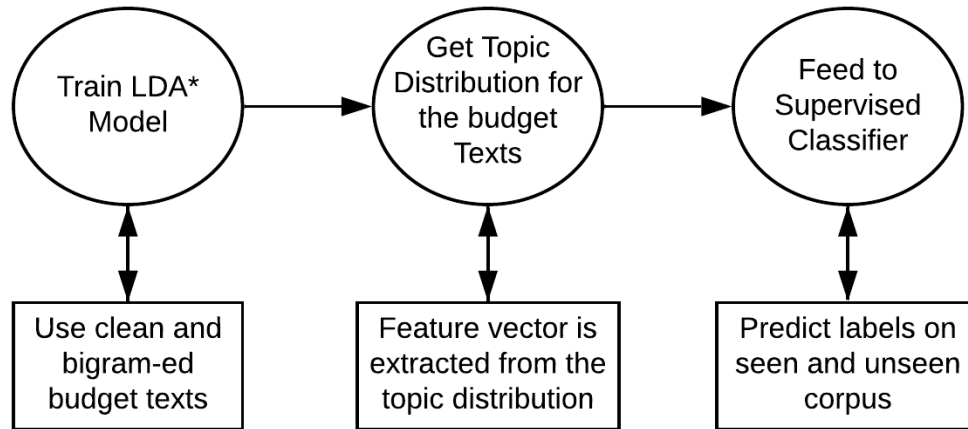


Fig: Project Methodology

Here, the goal was to use the LDA Topic Models as a classification model input. The processes involved to achieve the method are as follows:

- Train the LDA Model on the budget texts from 2019.
- Grab Topic distributions for every budget texts using the LDA Model
- Use Topic Distributions directly as feature vectors in supervised classification models (Logistic Regression, SVM, etc) and get F1-score.
- Use the same 2019 LDA model to get topic distributions from 2018 and 2020 (the LDA model did not see this data!)
- Run supervised classification models again on the 2018 and 2020 vectors and see if this generalizes.

At first, the topic distributions are converted into feature vectors which will be used for training the model. i.e. here, for the supervised learning, the input(x) is the topic distributions obtained through the LDA model. The process of extraction of feature vectors is shown below:

```
In [108]: train_vecs = []
          for i in range(len(GC_df)):
              top_topics = lda_model.get_document_topics(corpus[i], minimum_probability=0.0)
              topic_vec = [top_topics[i][1] for i in range(10)]
              topic_vec.extend([GC_df.iloc[i].sent_count]) # counts of reviews for restaurant
              topic_vec.extend([len(GC_df.iloc[i].word)]) # length review
              train_vecs.append(topic_vec)

In [109]: train_vecs[2]
Out[109]: [0.04846649,
            0.042821117,
            0.03781131,
            0.0386842,
            0.055064,
            0.050130684,
            0.043984495,
            0.087888956,
            0.54818475,
            0.046964042,
            36,
            4]
```

Here, the `train_vecs` are the array of feature vectors which will be used as 'x' for supervised machine learning.

The models used for supervised machine learning are: Logistic regression and SVM which are both the extension of linear regression models.

Now, at first the Supervised Classification is conducted by using the training data which is the topic distribution we got from LDA model 2019, such that:

- `X = [train_vecs];`
- `Y = [predicted_labels];`

Here, we got the following classification results, as shown below, and, in the bar graph:

```
Logistic Regression Val f1: 0.869 +- 0.003
Logistic Regression SGD Val f1: 0.855 +- 0.008
SVM Huber Val f1: 0.870 +- 0.003
```

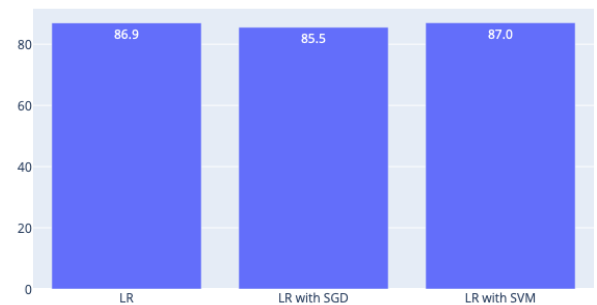


Fig: f1 scores on training dataset

The Link to the Ipython notebook for above tasks in GitHub:

https://github.com/UNCG-CSE/Budget_Text_Analysis/blob/master/src/machine_learning/train_lda_model_and_prepare_corpus_2019.ipynb

Also, the LDA model was saved for future use on the testing corpus from 2018, and 2020 as same LDA model is used for different corpus.

Then, the LDA model from 2019 was used to extract the topic distributions from the corpus of data from 2018 and 2020. The testing data was created for 2018 and 2020 respective using the obtained topic distributions following the same process as before. The following f1 scores were computed for the years:

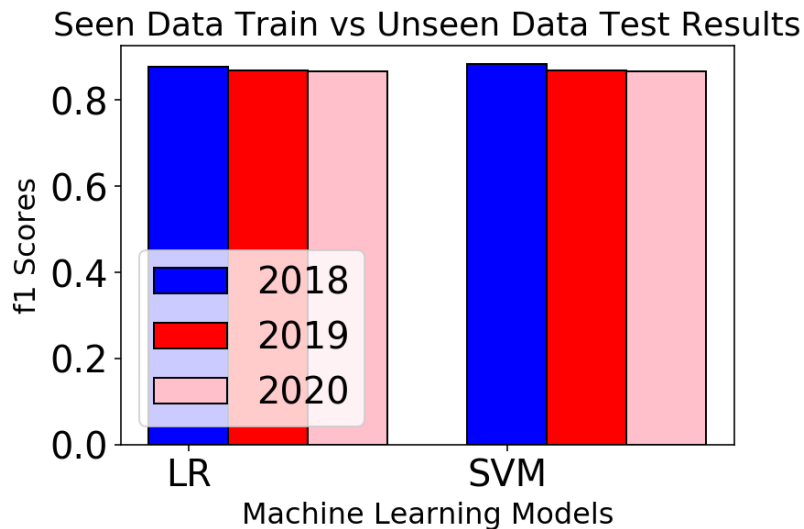
- For 2018:

0.8775611031997443
0.883026010151702

- For 2020:

0.8663699340718182
0.8665751454533569

Surprisingly, the f1scores for logistic regression and SVM for both the years 2018 and 2019, was found to be pretty similar to that of 2019. The comparisons is shown in bar diagram as follows as well:



Here, we can see from the bar-diagram that the f1scores for over the year were pretty similar. From this, we can implicate that the topic model for 2019 year can identify the latent semantic structure that persists over time in this budget text domain. From this approach, we can also conclude that same LDA model for a single year can be used over the years to get the topic distributions. This on one hand saves time and effort, as well as helps to enlighten that in budget documents the topic distributions over the years are pretty similar and unchanged.

The Link to the Ipython notebook for above tasks in GitHub:

https://github.com/UNCG-CSE/Budget_Text_Analysis/blob/master/src/machine_learning/test_lda_model_on_unseen_data_2018.ipynb

https://github.com/UNCG-CSE/Budget_Text_Analysis/blob/master/src/machine_learning/test_lda_model_on_unseen_data_2020.ipynb

4) Hypothesis Testing (2 hours):

- H_0 (null hypothesis) -> The ML models are similar and perform for all the year
- H_1 -> The ML models are truly different and perform differently.
- Condition for Hypothesis taken such that p-value threshold is $p = 0.05$

We got the following results from the hypothesis test:

```
chi-squared: 10.861150070126227  
p-value: 0.0009820269000594094
```

We got p-value nearly converging to zero, and, significantly less than the p value of 0.05. Hence, the null hypothesis was rejected, and we could conclude that the models were completely different.