CPSC 453 Final Project

Mark Torres (mpt24)

December 18, 2019

Pt I: Problem Statement

LDA is a common way to group text data into groups, via analyzing words and assigning topics to these texts. LDA doesn't require a distance metric, but rather uses probability distributions and a bag-of-words frequency-based to create its groupings. In contrast, clustering techniques (e.g., k-means, spectral clustering, Louvain) utilize and minimize some sort of distance metric in order to create groupings. I am interested in how typical clustering algorithms perform in contrast to LDA when performed on a corpus of Twitter data. I plan on using LDA to classify tweets into certain topics and examining what topics arise as a result. To compare with LDA, I plan to assign vector projections and distances to the corpus of the text data (using tf-idf and vectorization), which would allow me to perform clustering on the text based on distance metrics (e.g., Euclidean distance).

Here, I will compare the topics that arise from LDA with the topics that arise from the k-means++ algorithm. I am interested in seeing how the topics that arise from a frequency-based model (i.e., "bag-of-words") such as LDA differ from the topics that arise from a distance-based model such as k-means++.

Pt II: Data Collection

The data consisted of 628,696 tweets that were posted from September 1, 2019 up to October 21, 2019. The goal was to analyze general opinions relating to climate change, so the tweets were scraped according to the content of their hashtags. I scraped the tweets to align with recent climate change events (e.g., the September 2019 climate strikes, aka Global Week for Future) in order to get a large corpus of tweets in a concentrated time period. In particular, I searched for the following hashtags during my scraping:

Pro climate-change: #climatechange, #climatecrisis, #ActOnClimate, #ClimateEmergency, #ClimateStrike, #change, #environment, #nature

Anti climate-change: #climatehoax, #climatechangehoax, #climatescam, #climatefraud, #climategate, #trump, #fakenews, #conspiracy

There was an approximately 12:1 ratio between the pro-climate-change and anti-climate-change tweets. After collecting these tweets, I performed the necessary regular expressions in order to remove links, punctuation, retweets, and other irrelevant features of the text. In the end, I accumulated a corpus of 628,696 tweets with which to train my models. I would like to see if the models will recover some of the keywords that were used in the scraping process.

Pt III: Latent Dirichlet Allocation (LDA)

What is LDA? What motivates it?

Latent Dirichlet Allocation (LDA) is a generative model that infers the topics that are shared by a set of documents. It takes a corpus of documents and assumes that each document is made up of a set of topics, and that each word in a document refers to one of the topics of that document. In the LDA, each document is assumed to cover only a small set of topics at most, and each topic has a list of words that are associated with that topic. In essence, LDA assumes that documents are random mixtures of latent topics, and each topic is represented by a distribution of key words.

$$P(Z_{(m,n)} = v \mid \boldsymbol{Z}_{-(m,n)}, \boldsymbol{W}; \alpha, \beta)$$

$$\propto P(Z_{(m,n)} = v, \boldsymbol{Z}_{-(m,n)}, \boldsymbol{W}; \alpha, \beta)$$

$$= \left(\frac{\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)}{\prod_{i=1}^{K}\Gamma(\alpha_i)}\right)^{M} \prod_{j\neq m}\frac{\prod_{i=1}^{K}\Gamma\left(n^{i}_{j,(\cdot)}+\alpha_i\right)}{\Gamma\left(\sum_{i=1}^{K}n^{i}_{j,(\cdot)}+\alpha_i\right)} \left(\frac{\Gamma\left(\sum_{r=1}^{V}\beta_r\right)}{\prod_{r-1}^{V}\Gamma(\beta_r)}\right)^{K} \prod_{i=1}^{K}\prod_{r\neq v}\Gamma\left(n^{i}_{(\cdot),r}+\beta_r\right) \frac{\prod_{i=1}^{K}\Gamma\left(n^{i}_{m,(\cdot)}+\alpha_i\right)}{\Gamma\left(\sum_{i=1}^{K}n^{i}_{m,(\cdot)}+\alpha_i\right)} \prod_{i=1}^{K}\frac{\Gamma\left(n^{i}_{(\cdot),v}+\beta_v\right)}{\Gamma\left(\sum_{r=1}^{V}n^{i}_{(\cdot),r}+\beta_r\right)}$$

Figure 1. Derivation of the Latent Dirichlet Allocation (LDA) equation. Alternative derivations are possible and are appropriate for larger samples and number of topics, but for our purposes the base LDA implementation is sufficient.

Require: words $w \in$ corpus $\mathcal{D} = (d_1, d_2, \ldots, d_m)$

procedure LDA-GIBBS(w, α, β, T)
    randomly initialize z and increment counters
    loop for each iteration
        loop for each word w in corpus $\mathcal{D}$
            Begin
                word $\leftarrow w[i]$
                tp $\leftarrow z[i]$
                $n_{d,tp} -= 1; n_{word,tp} -= 1; n_{tp} -= 1$
                loop for each topic j $\in \{0, \ldots, K-1\}$
                    compute $P(z_i = j | z_{-i}, w)$
                $tp \leftarrow sample \quad from \quad p(z|.)$
                $z[i] \leftarrow tp$
                $n_{d,tp} += 1; n_{word,tp} += 1; n_{tp} += 1$
            End
    Compute $\phi^{(z)}$
    Compute θ_d
    return $z, \phi^{(z)}, \theta_{\mathcal{D}}$ ▷ Output
end procedure

Figure 2. The LDA algorithm, with Gibbs sampling.

What did I expect from the LDA model?

LDA is a bag-of-words model, so it relies on the frequencies of the words (this is in contrast to bigram or n-gram models that rely on groups of words or the context of the words). Because of this, I expected the topics to strongly represent terms that appeared in higher frequencies (regardless of the importance of the words, so perhaps commonly used words such as "act" would be given more weight than the keywords that I used to scrape the tweets, such as "change"). I was worried that LDA would not provide robust results because of how short tweets tend to be, but past research shows that LDA is still robust on a corpus of Twitter data (Weng et al., 2010).

Pt IV: K-Means++

Preparation for clustering approaches (turning tweets into vectors)

A common thread shared between the clustering approaches that I attempted was the need for a distance metric. The various clustering approaches all aim to optimize some form of distance metric, so in preparation, I needed to convert the tweets into numeric vectors.

To do so, I implemented a method known as term frequency – inverse document frequency (tf-idf), which allows us to determine the importance of a particular word in a document (in this case, a given tweet). It works by analyzing how often a given word appears in a document, but is offset by the number of documents that the word appears in. For example, a word such as "the" would receive a low score because it can appear often in one given document (since it is a common stopword) and likely appears in the majority of documents. In contrast, a keyword such as "election" may receive a higher score because it might appear frequently in one given document but not be common in general.

We can use this methodology to create "scores" for each of the words in our corpus. For example, a tweet such as:

> "I believe in climate change"

Could then be represented with the following vector:

> [1, 5, 1, 5, 7]

This process is known as vectorization. In this way, we can represent each of the tweets in our corpus using a multidimensional vector, from which we can create distance metrics in order to use approaches such as K-Means++ Clustering.

K-Means++ Clustering

K-Means clustering aims to partition *n* observations into *k* clusters, where each observation is assigned to the nearest cluster (by distance). The k-means clustering algorithm minimizes within-cluster variances and mean squared errors, but is susceptible to the presence of outliers (alternative implementations, such as k-mediods clustering, help circumvent these problems). K-means clustering can quickly converge to local minima solutions, based on the random initialization. In this report, we use the k-means++ algorithm, which is an implementation of the regular k-means algorithm that also improves the initialization of the starting points used in the algorithm (Arthur and Vassilvitskii, 2006).

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 = \arg\min_{\mathbf{S}} \sum_{i=1}^{k} |S_i| \operatorname{Var} S_i$$

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \frac{1}{2|S_i|} \sum_{\mathbf{x},\mathbf{y} \in S_i} \|\mathbf{x} - \mathbf{y}\|^2$$

Figure 3. Objective function of the k-means algorithm. The goal is to minimize the within-cluster sum of squares (*top*), which is equivalent to minimizing the pairwise squared deviations of points in the same cluster (*bottom*).

K-Means ++ Initialization:

1. Choose one center uniformly and at random from the data points.
2. For each observation, calculate the distance between that observation and the center.
3. Choose one new data point to be the new center, using a probability distribution weighted by the distance between that observation and the center.
4. Repeat steps 2 and 3 until *k* centers have been chosen.
5. Implement K-Means algorithm.

Implementation of K-Means:

1. For each observation, calculated the distance between that observation and each of the cluster means.
2. Assign each observation to the cluster that minimizes the distance.
3. For each cluster, assign a new mean that is the centroid of that cluster.
4. Repeat steps 2 and 3 until convergence is reached (one example convergence criterion is that the cluster assignments no longer change).

Figure 4. Breakdown of the K-Means++ algorithm implementation.

Pt V: Comparison of LDA vs. K-Means++

Outcomes of Latent Dirichlet Allocation

Results

For my implementation of the LDA algorithm, I used 10 topics. I also initialized the model with the following parameters:

- Number_of_topics = 10
- Update_every = 500 (so, the parameters of the model would be updated after every 500 tweets)
- Chunk_size = 500
- Passes = 2 (so, two epochs)

The following were the most prevalent keywords associated with each topic:

Topic 1	**Topic 2**	**Topic 3**	**Topic 4**	**Topic 5**
Climatechange	Climatechange	Climatechange	Climatechange	Climatechange
Climatecrisis	Climatehoax	Climatecrisis	Climatecrisis	Climatestrike
Climatehoax	Climatecrisis	Climatestrike	Climatestrike	Climatehoax
Climatestrike	Climatefraud	Climatehoax	Climatehoax	Gretathunberg
Actonclimate	Tweet	Actonclimate	Actonclimate	Globalwarming
Topic 6	**Topic 7**	**Topic 8**	**Topic 9**	**Topic 10**
Climatechange	climate	climatecrisis	Climatechange	Climatestrike
Climatecrisis	Climatestrike	Gretathunberg	Climatehysteria	Change
Dare	Love	Climatescam	Climatestrike	Greta
Globalwarming	Actonclimate	Climatecult	Globalwarming	Climatehoax
Climatehysteria	Peacefulprotest	Strike	Facts	Daysofeve

Figure 5. Each of the 10 topics discovered by the LDA algorithm. Note that 78.5% of the tweets were classified under Topic 1 (each of the other topics contained anywhere between 2.2% and 2.8% of the tweets). Therefore, the results for Topic 1 should be weighed the most heavily.

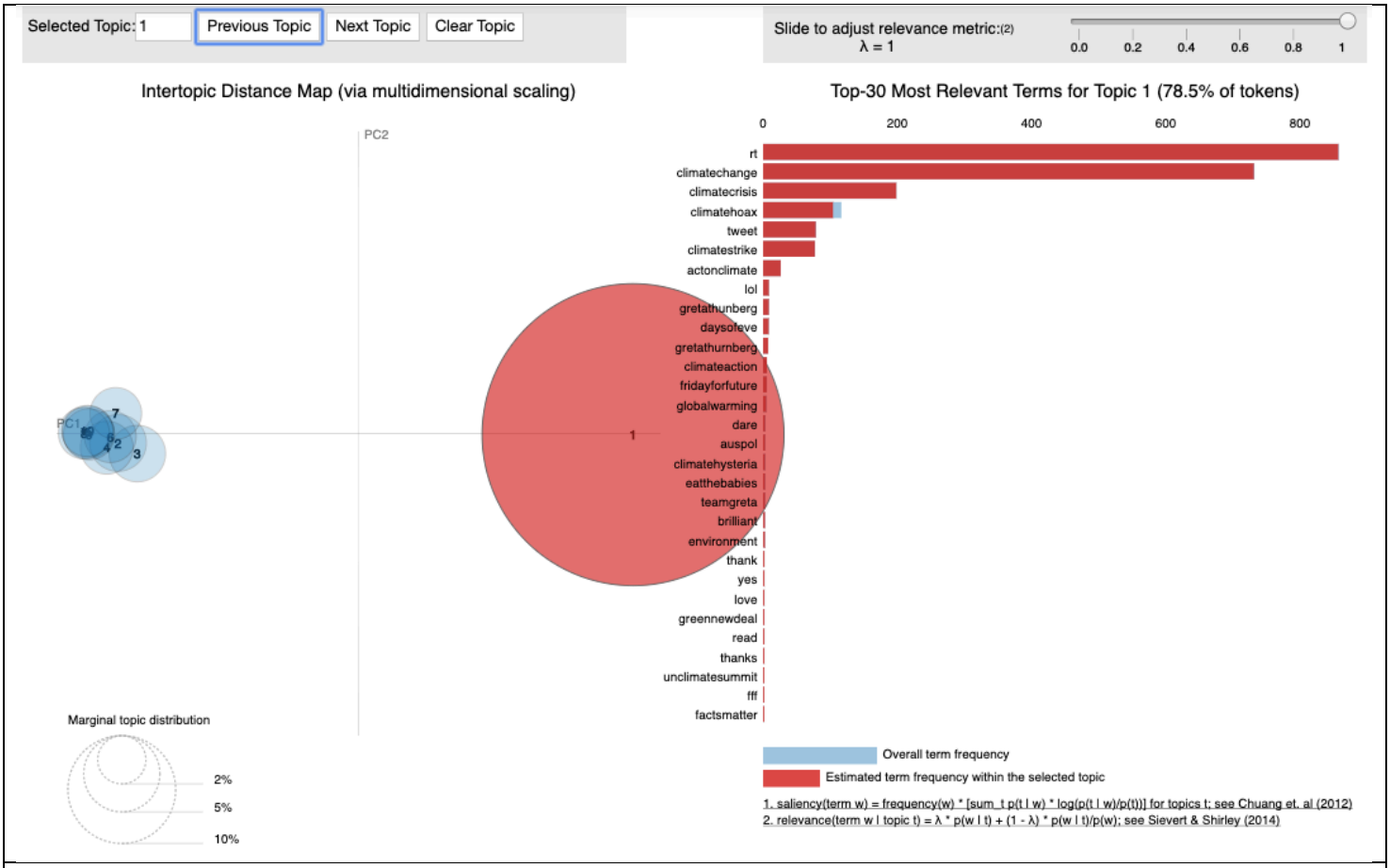

Figure 6. Example output of the LDA algorithm. The intertopic distance map (*left*) is a visualization of the topics as circles in a 2-D plane, whose centers are calculated by computing the Jensen-Shannon divergence between the topics and then using multidimensional scaling to project the inter-topic distances onto two dimensions. The area of each circle indicates the prevalence of each topic in the corpus. The list of salient terms (*right*) indicate the terms that are most relevant to the selected topic (here, I looked at topic 1, which was relevant for 78.5% of the tweets in the corpus). The length of the two overlaid bars indicate the overall frequency of that term in the corpus itself (the blue bar) and the frequency of that term in the selected topic (the red bar).

Outcomes of K-Means++ Clustering

Results

For my initial implementation of the K-Means++ algorithm, I used k = 10. The following were the keywords associated with each topic:

Topic 1	**Topic 2**	**Topic 3**	**Topic 4**	**Topic 5**
Climatechange	Climateemergency	Catania	Montreal	Change
Climatestrike	Climatechange	Fridaysforfuture	Street	Spoken
Today	Climatestrike	Climateactionweek	Time	Continue
People	Action	Climateactionnow	Thousand	Leader
Climateemergency	Fridayforfuture	500	Massive	Listen
Fridayforfuture	Globalclimatestrike	climatestrike		
Topic 6	**Topic 7**	**Topic 8**	**Topic 9**	**Topic 10**
Climate	Future	Climatecrisis	Fridaysforfuture	Klimatstrejk
Change	Strike	Action	Schoolstrike4climate	FridayforFuture
Strike	Friday	Million	Vancouver	Coming
Million	School	Scientist	Chile	Change
Global	Climatestrike	Respected	Italy	500
Action	klimatsrtrejk	World	Million	climatestrike

Figure 7. Each of the 10 topics discovered by the K-Means++ algorithm, as well as the keywords associated with each topic. The K-Means++ algorithm recovered many of the keywords that were used in the initial query (e.g., climatechange, climatestrike) as well as uncovered common co-occurring terms not used in the initial scrape but were nonetheless relevant at the time (e.g., FridayforFuture)

Did it recover the intended keywords?

The k-means++ algorithm seems to have recovered many of the keywords used in the initial tweet-scraping process. However, it only seemed to have recovered the pro-climate-change keywords. None of the topics contain keywords that were used to scrape anti-climate-change tweets. I find this to be somewhat surprising but also probable because of the imbalance between pro-climate-change and anti-climate-change tweets (in this corpus, I was able to scrape 12 pro-climate-change tweets for every 1 anti-climate-change tweet). It is possible that a few of the anti-climate-change tweets were classified under the same topic as some pro-climate-change tweets (e.g., anti-climate-change tweets likely had the word “climate” in it, so they were perhaps classified in these clusters). It doesn’t seem like I had sufficient anti-climate-change tweets for the algorithm to recover the anti-climate-change tweets as their own cluster.

Trends and interesting findings

In addition to recovering the intended keywords, the K-Means++ algorithm also provided other interesting results. For one, it recovered other interesting keywords (and likely hashtags) that seemed to co-occur with the tweets that I had scraped (e.g., FridaysForFuture, SchoolStrike4Climate). Moreover, the keywords defining the topics reinforce the fact that the tweets were scraped during Fall 2019 (e.g., FridaysForFuture, Change, ClimateEmergency, millions), when there was a great deal of news coverage and advocacy regarding climate change (e.g., events such as the Fridays for Future climate strike and millions hitting the streets globally to protest climate change). Since I scraped tweets from online, the k-means++ algorithm also was able to pick up non-English words (e.g., klimatsrtrejk, which is Swedish for "climatestrike"- picking up on this term makes sense because of the work of Greta Thunberg, a 16-year old Swedish student and climate change advocate).

Modifying number of clusters

I also decided to experiment with modifying the number of clusters. I postulated that using fewer clusters could lead to consolidation of the key topics (since many of the original topics greatly overlapped in content and keywords). Moreover, I also postulated that using more clusters could perhaps recover topics that could uncover the anti-climate-change keywords and tweets (the initial implementation didn't have any topics specific to anti-climate-change).

I experimented with several values of *k* to see the effect of using fewer clusters, but saw that using k = 5 was the optimal number for recovering a wide enough range of topics and keywords without substantially removing other keywords (e.g., using k = 3 removed keywords relating to events, such as "Montreal" and "Auckland", both sites of large-scale climate-related marches. In contrast, using k = 8 led to a few redundant topics that could have used consolidation). Using k = 5, I found the following topics and associated keywords:

Topic 1	**Topic 2**	**Topic 3**
Climatestrike	Fridaysforfuture	Climateemergency
Climatechange	Climatestrike	Climatechange
Climatecrisis	klimatsrtrejk	Climatestrike
Montreal	change	Climateaction
People	coming	Today
Street	Italy	Globalclimatestrike
Topic 4	**Topic 5**	
Wellington	City	
Auckland	Iowa	
Fridaysforfuture	Fridaysforfuture	
Climatestrike	Climatestrike	
Growing	School	

Demand Thousand	Strike Happy	

Figure 8. Each of the 5 topics uncovered by the K-Means++ algorithm. The topics and keywords were similar to the implementation with 10 topics, with the exception of several new keywords (e.g., Wellington).

I also used a few clusters to investigate the effects of using more than 10 clusters in order to see if it were possible to uncover topics related to anti-climate-change (the initial implementation could not recover a topic related to anti-climate-change). I settled with using k = 15 because it again seemed to provide the best combination of breadth and specificity as above. Using k = 15, I found the following topics and associated keywords:

Topic 1	**Topic 2**	**Topic 3**	**Topic 4**	**Topic 5**
Climateemergency Climatechange Climatecrisis Climateaction Fridaysforfuture	Change Coming Climatestrike Spoken Continue	Climatechange Climateaction Preach Gretathunberg Real	Quebec Parliament Wait Incredible Planet	Fridaysforfuture Climatestrike Italy People Vancouver
Topic 6	**Topic 7**	**Topic 8**	**Topic 9**	**Topic 10**
Climatechange Million Climatechange Climatestrike Vancouver	Climatestrike Climatecrisis People Fridaysforfuture Million	Newfoundland Fridaysforfuture Climatestrike Friday Continue	Thousand Street Milan Massive Demand	Climatechangehoax Breakthrough Climatechange Climatescam Proposed
Topic 11	**Topic 12**	**Topic 13**	**Topic 14**	**Topic 15**
City Iowa Fridayforfuture Schoolstrike4climate Climatestrike	Montreal Official Street Today Ending	Santiago Chile Climatestrike Valparaiso Video	Protesting Government Netherlands Barcelona Future	Climate Climatestrike Globalclimatestrike Action Million

Figure 9. Each of the 15 topics uncovered by the K-Means++ algorithm. The algorithm was able to successfully recover 1 topic that referred to anti-climate-change tweets (*see Topic 10*).

By using k = 15, the algorithm was able to successfully recover 1 topic that encompassed the anti-climate-change tweets (see Topic 10). Notably, using k = 15 was the smallest k that I used that was able to recover 1 topic where the most heavily weighted keyword in that topic was

one of the anti-climate-change keywords that I had initially used. Given that the ratio of pro-climate-change tweets to anti-climate-change tweets was around 12:1, it makes sense that for the anti-climate-change tweets to be represented with their own topic required at least 13 clusters (in this case, 15 was the optimal amount). Because of this, it seems like overcoming the imbalance of tweets for each side requires using number of clusters *k* matching the ratio of one side to another.

Interestingly, increasing the number of clusters also revealed more nuanced topics with a different set of keywords (e.g., Topic 3 includes "Greta Thunberg" as a keyword, and she was a key advocate in the climate change movements), albeit at the cost of sometimes increasing overlap across topics.

Comparison of LDA and K-Means++

Similarities between the outcomes

Both LDA and K-Means++ were able to recover many of the keywords that were used to scrape the tweets, so they successfully discovered the intended general theme uniting all the tweets (that is, climate change). Moreover, several of the keywords (e.g., "climatechange") were treated as similarly important in both the LDA and the k-means++ implementations.

Differences between the outcomes

The LDA and K-Means++ algorithms, interestingly, provided distinctly different topics.

One problem with the corpus that I had used was that there was a stark imbalance in the prevalence of pro-climate-change and anti-climate-change tweets. In the K-means++ implementation, this imbalance was magnified and a topic that uniquely encompassed the anti-climate-change tweets was only possible when *k* was increased to 15. Since the clusters were likely more proportional in size, it makes sense that a uniquely anti-climate-change topic would only occur when the number of clusters approximates the ratio of pro-climate-change and anti-climate-change clusters. However, such an imbalance wasn't as large of a problem with the LDA implementation, since the topics had a higher frequency of anti-climate-change keywords compared to the k-means++ implementation. This stark difference could have been motivated by the fact that the k-means++ implementation could have converged to clusters of similar sizes, while the LDA had a clear imbalance in how many tweets were classified under each topic (Topic 1 contained 78.5% of the tweets, while each of the rest of the topics contained anywhere from 2.2% to 2.8% of the tweets). In the original corpus, the pro-climate-change tweets outnumbered the anti-climate-change tweets 12:1. However, it is possible that in the LDA implementation, a large proportion of the pro-climate-change tweets were classified under Topic 1, so that the ratio of pro-climate-change and anti-climate-change tweets in the other topics was more balanced. If this is the case, then it makes sense why anti-climate-change keywords would be more readily represented in the other topics – the frequency of pro-climate-change and anti-climate-change tweets is more balanced.

Another key difference between the two algorithms is that the topics uncovered by LDA had both less uniform keywords (in terms of pro vs. anti-climate change) and a smaller range of keywords. The topics in LDA were more likely to include both pro-climate-change and anti-climate-change words (e.g., having both "climatechange" and "climatescam" as keywords), whereas the keywords uncovered through the K-Means++ method tended to be more homogeneous in whether they were pro-climate-change or anti-climate-change. Moreover, the topics uncovered in LDA had a similar set of keywords (e.g., "climatechange", "climatestrike"), and although there were similarities in the keywords used in the K-Means++ algorithm, there was a larger set of unique keywords in the K-Means++ algorithm than in LDA. These key differences could have appeared because LDA is a frequency-based model while K-Means++ is a distance-based model (that uses tf-idf, a weighted frequency metric). It is possible that LDA examined the tweets and only cared about whether the terms in two tweets had similar frequencies, regardless of what the terms actually are. In contrast, the K-Means++ implementation required a document matrix which reveals how often terms co-occur with each other, so the K-Means++ algorithm would weigh more heavily terms that actually co-occur. Therefore, it makes sense that the K-Means++ algorithm reveals more homogeneous topics – the distances are based off a matrix that examines how often words appear together, so the algorithm converges onto clusters where the words tend to co-occur (and they are more likely to co-occur if they reveal similar sentiments about the topic). Moreover, since the LDA weighs keywords just by frequency alone, it makes sense that the set of unique keywords is smaller in LDA than it is in K-Means++, since LDA looks purely at frequencies and, for any given subsample of the tweets, the same words will likely be most prevalent. In contrast, K-Means++ places a greater importance not only on prevalent words, but which words appear in conjunction with these words even if they themselves are not common (and these infrequent yet important words are in fact given a larger weighing because of the tf-idf implementation). Therefore, the topics in the K-Means++ algorithm are more likely to have a diverse set of keywords.

Personally, I believe that the topics and keywords from the K-Means++ algorithm are both more useful and more interpretable than the LDA implementation, largely because the topics and keywords are more unique, homogeneous, and differentiated (this is likely driven by the fact that K-Means emphasizes both within-cluster similarity and the existence of distinct, unique clusters. In contrast, LDA emphasizes only frequencies, irrespective of the uniqueness of the topics).

Potential limitations

tf-idf implementation and vectorization

My implementation of the k-means++ algorithm was dependent on the vectorization and tf-idf process that transformed each tweet into a vector. It is possible that this process weighted words differently than I had intended. For example, because tf-idf aims to emphasize "important" words that do not necessarily appear in all documents, it is possible that a critical word such as "climate" was penalized (since, in a corpus of tweets regarding climate, it is likely

that the word “climate” appeared in most tweets and as such may have been penalized). This could also have been a problem with the keywords that I used to scrape the tweets – they were too common and were penalized as a result.

Even if the tf-idf and vectorization process assigned lower values for these words than intended, the k-means++ algorithm likely made up for it. For example, if I had intended the word “climate” to have a weighing of 10 but it instead had a weighing of 5, this would not have been detrimental to the k-means++ implementation since it would have that same weighing of 5 across all vectorized tweets. Therefore, so long as the words had the same weighing across the documents, the distance metric would overcome any shortcomings of the tf-idf process. The results seem to reinforce this, since words that I feared would not appear sufficiently often (e.g., “climate”, “climatechange”) did in fact appear in several of the topics that were discovered.

LDA

One key problem with LDA on Twitter data is the possibility that each document (in this case, a tweet), is too short in length. This could potentially be a problem because the LDA model assumes that each document consists of several potential topics, but it is difficult to reveal multiple topics in one document when one document (here, a tweet) could perhaps be only 5 words long. However, past research (Weng et al., 2010) reveals that, given a sufficiently large corpus, LDA still provides robust results, so in the present implementation I felt that LDA would be an appropriate model to use.

K-Means++

One problem with using a k-means++ algorithm is that although it has a fast convergence rate, it often converges to a local minimum. To circumvent this problem, I reran the algorithm multiple times and found that the topics and keywords were nearly identical, which shows the robustness of the k-means++ algorithm. I wasn’t surprised that the topics and keywords were nearly identical because even with the random initialization used by the k-means++ algorithm, the size of the corpus (600,000+ observations) seemed to suggest that similar centroids would be chosen during each iteration.

Another potential problem for the k-means++ algorithm is how it deals with tweets of different lengths. Although each tweet is represented by a vector whose length is equal to the length of the dictionary of the corpus, tweets that may seem semantically similar to a human might be represented as further away using the k-means++ algorithm (for example, a tweet with 5 words and a tweet with 10 words, even if they have the same message, might still be perceived as further away).

Possibility of using other distance-based algorithms

It would be interesting to see how other distance-based algorithms, such as spectral clustering and Louvain, would have performed on this dataset. I would suspect that similar keywords would be uncovered, because it seems that the K-Means++ algorithm was heavily influenced by the use of tf-idf and vectorization and how the tweets were vectorized as a result. It seems that since the input matrix already, in an intuitive sense, encodes the "similarity" of tweets (the document-term matrix provides the frequency that words co-occur), that spectral clustering and Louvain, which provide clusters based on similarity/adjacency metrics, should provide results that are similar to the K-Means++ algorithm. Although the results would be different since spectral clustering and Louvain are nonlinear methods, I would suspect that the difference would not be drastic mostly because of the fact that the input to all these models (the document-term matrix) is, in a sense, already computing the "similarity" of the inputs.

Pt VI: Conclusion

In this project, I sought to compare how a clustering algorithm that we reviewed in class (K-Means++) would perform against a typical NLP model (LDA) in a task to find the topics of a corpus. I found clear differences in how the distance-based K-Means++ model performed as compared to the frequency-based LDA model. Although the topics, to a human reader, were similar in both, I found that the K-Means++ algorithm proposed more nuanced and clearly differentiated topics than the LDA, which can be attributed to key differences in the inner workings of both algorithms (i.e., the K-Means++ algorithm, with the tf-idf implementation, heavily weighs words that co-occur, while the LDA model heavily weighs the raw frequency of the words). I discussed in more detail the differences in results that I found in the implementations of both models and other reasons why they may have differed.

I personally found that the K-Means++ algorithm was more useful in clustering the corpus, since it provided more differentiated and interpretable topics. In contrast, the topics in the LDA model were too similar to be of much use. All in all, this project was an interesting way to learn about the use cases of distance-based clustering models and how they compare to other unsupervised learning methods.

Works Cited

Arthur, D., & Vassilvitskii, S. (2006). K-means++: The advantages of careful seeding.

Blei, D.M., Ng, A.Y., & Jordan, M.I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*.

Weng, J., Lim, E.P., Jiang, J., & He, Q. (2010). Twitterrank: Finding topic-sensitive influential Twitterers. *Proceedings of the Third ACM International Conference on Web Search & Data Mining: February 3-6, 2010, New York*.