Article Recommendations for News Feed

Minghan Shen (sminghan)
CS224N Final Project

Abstract

This project considers the task of analyzing and clustering news articles so as to provide article recommendations to a user based on previously viewed articles. The project code tries to use different clustering methods for topic discovery or finding exemplar articles within the set of all articles. The results can then be used to make predictions. Given a set of articles read by a user, recommend similar articles the user may be interested to read. The intricacies of this task are each analyzed and discussed in this paper.

1 Introduction

Pulse News is an application for handheld devices that provides a well-organized compilation of RSS feeds, personalized for each user. It is fast becoming a popular way to read articles from various news sites, and also allows users save or share interesting ones with family and friends.

On their main site for regular browsers, pulse.me, they have a simple set of 16 categories for users to choose and start reading news, from a beautiful but otherwise disorganized mosaic. The main method for grouping articles is the RSS feed source. This is the case for both mobile and non-mobile platforms. Most users are then required to manually sieve through all the news in their combined feed to look for the topics and news that they are actually interested in. For example, one may be interested in automobiles but only Harley-Davidsons, and will be stuck searching through hundreds of articles on the latest supercar trends without finding a second article on the motorcycles. There is hence some motivation to find a set of articles the users may be interested in reading, based on what they have read before.

Due to time constraints and lack of communication, I was unable to get a dataset from Pulse News. I tried to manually create my own dataset, but this was time consuming and while the toy dataset of 100 articles provided some insight, it was too small for any proper experiments. This paper instead uses a BBC dataset provided online by the UC Dublin Machine Learning Group[1] to simulate having a set of data from Pulse News. The goal is to identify relations between articles, or automatically cluster articles so that suggestions of future readings can be made from the clusters of articles read by the user. Unfortunately this also meant that user data was not available for testing.

2 Preprocessing: Feature Extraction

Using Python for rapid prototyping and available Python libraries SciKitLearn (sklearn) and Natural Language Toolkit (NLTK), feature extraction on the data is done by selecting the key words from articles that may be used to identify each article's topic[2].

Tokenize and Clean – First we tokenize the text and remove punctuation using split() and strip() python commands.

POS tagging and stopword removal – Part of Speech tagging (NLTK.pos_tag) is used to identify nouns and named entities which are more likely to be the subject or topic of an article. Other parts of speech and stopwords are discarded.

Stemming – Permutations of the same words are combined by running the words through NLTK's Porter Stemmer. An alternative would be to hypernyms, but that while that would allow more connections it would also necessarily increase the size of the feature vector.

Convert to vector – Easily done by calling sklearn's Vectorizer, which creates a sparse matrix of "[example,feature] = frequency" entries.

Low frequency removal – Words that occur <3 times are removed to further shrink the feature vector

Term frequency and inverse document frequency (TF-IDF) – it is a statistic that increase with document frequency and decreases with corpus frequency. This is used to balance words that occur a lot and have high frequency in the data, and stop them from skewing results.

3 Experiments

Evaluation Metrics

These are a few external metrics used in the field of clustering research [3], including:

Homogeneity: The extent to which each cluster only contains one category

Completeness: The extent to which all items of one category are in the same cluster

V-Measure: Basically a mean of Homogeneity and Completeness. A random assignment to clusters will get a V-measure close to zero.

They rely on external data such as human labeling of the dataset, to calculate the similarity score. Given that our goal is to find recommendations, we want to find articles that are similar within each cluster and want a high Homogeneity. Completeness is less important because having two clusters for one category may just be a more fine grain of sub-categories, and is more of a limitation imposed by the human labeling.

As for intrinsic metrics, I discovered that they generally try to measure how well separated the clusters are. That is not relevant to our task since we are expecting overlapping sets of articles with even outliers that are not expected to fall under any topic cluster. More often than not the objective function of a clustering algorithm seeks to score well on one of the intrinsic metrics as a termination condition, and give skewed results if tested on a similar metric. To handle these we do not consider intrinsic metrics in this paper.

Models Testing

It quickly became obvious that simple k-means clustering and nearest neighbors did not work well on the small dataset, because there would be many articles which are not related to any of the other articles, basically singletons or outliers[4]. Since clustering algorithms typically try to classify all articles, the outliers will make things difficult, and can be likened to noise in the data. This noise is particularly noticeable in the small dataset I was working with.

I decided to work instead on a dataset provided by the Dublin Machine Learning Group, a dataset of 2235 articles from the BBC ranging over 5 categories. The data is already preprocessed by stemming through PorterStemmer, vectorized with stop words and low frequency words removed, then put into a Market Matrix format. I was able to read the data into my Python script and run some simple tests on the dataset. After running the preprocess step there were 9635 features in the feature vector.

The dataset provides news stories from 5 broad categories, with labels provided in a separate file. These categories are used to calculate the Homogeneity of our results. If two articles are in the same cluster, we expect them to come from the same category. Tests were run on a 3.0 GHz dual-core with 12 GB ram.

	Homogeneity	V-Measure	Clusters	Fitting Time (s)
K-Means	0.834	0.837	5	15
Hierarchical Cluster	0.964	0.412	957	90
Affinity Propagation	0.911	0.388	370	400

Results from running on (2225, 9635) (article,feature) matrix, BBC dataset

K-Means

This is the most well-known clustering algorithm, used in various fields of machine learning to find patterns in data. Given that the data of n examples probably has k clusters, K-means guesses k initial cluster positions (or randomizes) and then tries to maximize a basic square-error function of the distance of each example from its assigned cluster center. The algorithm is very fast, with a time complexity of O(n k T), T is the number of iterations needed to reach convergence (or whatever maximum iterations is set to). Running the algorithm (n_clusters=5, n_init=10, tol=1e-6) on the dataset yielded a 0.834 Homogeneity. The weakness of this algorithm for our particular task is the fact that you need to set the expected number of clusters before running the algorithm. This is impossible when we are trying for knowledge discovery, since we cannot know beforehand how many clusters would fit the unseen unlabelled data best. In the next two models we explore more unsupervised clustering algorithms that do not have a fixed number of clusters.

Hierarchical Clustering

This model constructs a dendrogram of clusters with bottom up construction [5]. Each article starts as a singleton cluster and with each iteration a pair of nearest clusters are combined into a larger tree, and together is considered a new cluster for the next iteration. This model encompasses the idea that each article may be in some way related to another article without being in the same category.

This sounded good and so I tried to implement this to cluster the articles. I soon found that while it was intuitive and the tree provides some good data, the main problem with this implementation is the huge tree cannot easily be flattened into a set of usable clusters. This degenerates back into calculating a distance matrix and in my opinion defeats the purpose of generating the tree. I also found parameter tuning to be very difficult as the algorithm tended to choose singleton clusters to maximize its objective function. I also found that when the clusters are too numerous, it is very easy to score a high homogeneity score. So I decided to also include the V-Measure to have a gauge of how complete the clustering was. The implementation of hierarchical clustering is cubic, but quadratic implementations in C exist[6].

Affinity Propagation

Finally I tried Affinity Propagation [7]. The algorithm tries to identify exemplars or main topic articles and forms clusters using these as the centers. The iteration goes by each data point exchanging messages until the objective function is maximized. The objective function for this algorithm is net similarity in the cluster, which is close so what we desire. The advantage of this algorithm is that it is less susceptible to local minima, since it tests (with the messages) almost all permutations before converging. The drawback for this is that it takes much more time than the other two algorithms. The original researchers claim exact affinity propagation was able to cluster 23,000 data points based on all pairwise similarities (23000x23000) in a few hours on a 16GB 2.4GHz machine, however this is still much slower than a k-means algorithm with numerous random restarts. The time complexity is quadratic to number of points but the constant term (number of iterations) can be significant when the number of points is small. In our test case the algorithm took 700 iterations to converge.

Error Analysis and Model Selection

All three models give a good level of Homogeneity within the discovered clusters. The K-means algorithm was very susceptible to local minima in our relatively small dataset, and increasing the number of random restarts did not seem to alleviate the problem of high variance in results. Hierarchical Clustering was a promising algorithm, but due to lack of understanding and time I was unable to produce a suitable parameter tuning code for the model. The homogeneity was high but because cluster sizes were small the results were not useful for predicting and suggesting future reads, since it is likely that the query contains all the elements of the cluster it matches. To lower the number of clusters the tolerance value for the flattening of clusters has to be tweaked, but in my experiments the decrease in clusters resulted in an exponential decrease in scores as well. In practice, it appears that the final clustering is done by manually inspecting the dendrogram (which is good for human visualization).

Finally, the Affinity Propagation uses an objective function that suits our needs; an affinity matrix between articles allows us to easily find the clusters of similar articles while not really penalizing lack of completeness. The results are promising as well, producing 370 clusters which give an average of 6 articles per cluster. Thus, using the result from fitting the Affinity Propagation algorithm, we can provide recommendations based on a short reading history of the user.

Test Application

To try out the algorithm, we want to try suggesting some articles to a user. Since our BBC dataset does not provide the original articles, for copyright and space saving reasons, we turn back to the toy dataset I created at the start of the project. My hypothetical user Neil, has an interest in Astronomy and NASA, and wants to read things on Pulse News related to the Cosmos. Let's say he reads the following 3 articles:

1) Comets Lay Siege Around Nearby Star Systems
2) Space Station Will Turn to Face the Sun
3) No Organics Yet For Mars Rover Curiosity, NASA Warns

By looking at the affinity matrix calculated with negative Euclidean distances between articles and used to create the clusters, we are able to in turn calculate the L2-norm distance between the set of read articles with the remaining articles. The results are from matching each article first as a singleton, then as a trio. Suggested matches are the entries with top affinity scores.

Command Line Output:

```
individual matches
input:  Comets Lay Siege Around Nearby Star Systems
matches:
-  Space Station Will Turn to Face the Sun
-  Wired Science Space Photo of the Day: Eskimo Nebula

input:  Space Station Will Turn to Face the Sun
matches:
-  Comets Lay Siege Around Nearby Star Systems
-  To Fight Winter Blahs, Sweden Offers Light Therapy At The Bus Stop

input:  No Organics Yet For Mars Rover Curiosity, NASA Warns
matches:
-  Frozen Water and Organic Material Discovered on Mercury
-  How to Beat Shady Data Dealers: Selling Our Own Info

matched together
combined match:
-  Wired Science Space Photo of the Day: Eskimo Nebula
-  Frozen Water and Organic Material Discovered on Mercury
-  Scientists Find Clearest Evidence Yet Of Monumental Polar Ice Melt
-  The Classics: 'Star Wars: Shadows of the Empire'
```

4 Conclusion

The results are very promising for our toy dataset. It is clear that with more reader data, the accuracy of the match increases, with the trio combined matching a set of all 4 articles related somewhat to the Cosmos. I can imagine Neil would read all four articles if they were suggested to him.

There are limitations to the methods used, such as space complexity of keeping the entire affinity matrix in memory. It would also be infeasible to compare each article to the entire dataset. However a simple modification of the algorithm would be to only compare articles which are in nearby clusters (which has already been calculated but not used in the application).

Other research has also been done to control the number of clusters produced by these unsupervised clustering algorithms that do not have a fixed prior expected number of clusters [8]. Another point of consideration is the idea of temporal topics and events, where certain named events are only relevant for a fixed period of time. A simple adjustment would be to have a time window when considering a user's reading history, that way the features matched would automatically be relevant to the time frame. In addition, Pulse News in particular does not feature older news because it is primarily an RSS feed aggregator.

References

[1] Greene, D. and Cunningham, P. (2006), "Practical solutions to the problem of diagonal dominance in kernel document clustering", Proc. 23rd International Conference on Machine learning (ICML 2006)

[2] Bouras, Christos, and Vassilis Tsogkas. "Assigning Web News to Clusters."*Internet and Web Applications and Services (ICIW), 2010 Fifth International Conference on*. IEEE, 2010.

[3] Amigó, Enrique, et al. "A comparison of extrinsic clustering evaluation metrics based on formal constraints." *Information retrieval* 12.4 (2009): 461-486.

[4] Samir Khuller. "Clustering with outliers and generalizations", University of Maryland, Talk. Information retrieved: http://www.cs.umd.edu/~samir/talks/ITA.pdf

[5] Chung, Seokkyung, and Dennis McLeod. "Dynamic pattern mining: an incremental data clustering approach." *Journal on Data Semantics II* (2005): 85-112.

[6] Daniel Müllner, "*fastcluster:* Fast hierarchical clustering routines for R and Python", Information retrieved: http://math.stanford.edu/~muellner/fastcluster.html

[7] Frey, Brendan J., and Delbert Dueck. "Clustering by passing messages between data points." *science* 315.5814 (2007): 972-976.

[8] Salvador, Stan, and Philip Chan. "Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms." *Tools with Artificial Intelligence, 2004. ICTAI 2004. 16th IEEE International Conference on*. IEEE, 2004.