The Analytics Edge, Final Report: Building an Article Recommender for Bleacher Report Basketball

for Professor Dimitris Bertsimas and Emma Gibson

by Stephen Albro & Cyrille Combettes
salbro@mit.edu, cyrille@mit.edu

May 13, 2018

1 Motivation and Project Abstract

As avid basketball fans, we have enjoyed reading a variety of NBA articles to keep up-to-date on the league during our busy year in the MBAn program. Bleacher Report, a primary digital destination for sports article readers, contains well written articles covering the league, and competes as advertisement real-estate with the likes of ESPN.com, FoxSports.com, CBSSports.com, SBNation.com, and others. In this paper, we develop a prototype for an article recommendation system using state of the art natural language processing techniques for both cleaning and analyzing bleacher report articles.

In an age of cut-throat digital competition, it is vital for websites to be able to retain visitors. The hope for a Bleacher Report article reader is two-fold. First, that the visitor jumps from article to article *on the website*, rather than returning to the search engine results page. Second, that the visitor falls in love with the content and coverage of Bleacher Report's articles, and develops a loyalty to the website.

We feel that having a micro targeted article recommendation system would provide Bleacher Report with an analytics-based edge over its competitors, since well targeted articles could increase the chances of prolonged visits, and increase website loyalty by making users feel known. A reasonable document-recommendation strategy embed documents as vectors and then use K Nearest Neighbors to group documents together. Our project focuses on exploring the right article embedding strategy. At first, we ran an experiment to find the most promising from among three primary methods: word2vec, Non-negative Matrix Factorization (NMF), and Latent Dirichlet Allocation (LDA). We evaluated each approach using a hand-crafted test set. A preliminary analysis revealed LDA to be the most promising strategy, and so we focused the rest of our efforts on developing a quality LDA topic model, which we then used to produce visualizations of the topic landscape of Bleacher Report's NBA articles.

2 Data Collection and Cleaning

2.1 Data Collection

Bleacher Report does not provide easy access to their articles in a public database. However, News API is a company that provides API access to major news articles from a variety of sources, including Bleacher Report. Using NewsAPI, we queried Bleacher Report NBA articles from October 17, the start of the 2017-2018 NBA season. Our entire query consisted of joining words like basketball and NBA with all 30 teams and the top 25 active players. In the end, we received 3314 URLs (with their corresponding authors, titles, and other metadata) in return. We then wrote a script to request the webpage for each URL and scrape its content, picking the paragraphs out of the HTML.

2.2 Data Cleaning

In the real world, data is never clean, and this is magnified in the case of textual data. There were a number of preprocessing steps we had to do before our articles were ready for analysis. In our cleaning efforts, we made decisions *based on the purpose of our task* - to cluster documents for better recommendation power. In particular, a bag-of-words approach to cleaning was preferred over one that preserved punctuation and grammar.

2.2.1 Punctuation Removal, Initial Stopwords, and Lowercasing

First, we tokenized each article. In natural language processing, tokenizing is a way of chopping up a document into pieces, the most obvious way being to tokenize by *word*, which is what we initially did. Next we discarded any non-alphabetical tokens (e.g. 8pm, 704, !, .), and converted every token to lower case. After converting everything to lower case, we did our first round of stop word removal (we remove more later). In NLP a stop word is any word that you remove because its insignificant for your application. In this first round, we removed basic articles (the, a) and about 170 other words deemed insignificant by Python's Natural Language Toolkit, mostly personal pronouns and simple verbs.

2.2.2 Specialized NBA Tokenization

Next, since we were analyzing NBA articles, we had to take into account the fact that each player (and each team for that matter) can be referred to in a variety of ways. For example, the NBA superstar LeBron James can be called LeBron James, LeBron, Bron, or James, and the Boston Celtics might be formally referred to as such in the title, but in the article's body they might just be Boston. For this reason, we compiled lists of team names, players, and coaches. For each article, we replaced all forms of each entity with its underscored full version, so that for example LeBron James would always be lebron_james, no matter if he occurred as Bron, James, or Lebron in a given sentence. At times it was necessary to infer from ambiguous usage. We concluded that, for

example, the word *Boston* should be replaced by boston_celtics only if the entire phrase Boston Celtics appeared somewhere in the article. This specialized, application-specific tokenization step allowed us to capture much more information about each superstar/team/coach than would for example, treating Bron as a separate player as Lebron.

2.2.3 Stemming Each Word

Next, we needed to address the fact that sometimes the same words take different forms. A reasonable assumption that we made the words playing, played, play, and plays all capture the same information as far as topic modeling goes. Thus we decided to *stem* each word, which in NLP means to convert every word into its root form. Python's Natural Language Toolkit provides a stemmer.

2.2.4 Trimming Down the Vocabulary

A model is only as good as its input, and so as a last step, we realized that we needed to trim down the vocabulary of our articles. In NLP, a *vocabulary* is simply the complete set of words that the model knows about. We already removed basic stop words, but figured that many more could be removed. For clustering documents, its important to identify information-containing words (player and team names, high-impact verbs, league terminology), and less important to identify mundane nouns and verbs (table, cup, gave, took) even if their not considered stop words.

To identify our functional vocabulary, we used a *document frequency* approach. The document frequency of a given word is simply the fraction of documents (in this case articles) that the word appears in. The word *the*, for example, has a document frequency of 1.0. If a word occurs too *infrequently*, it is probably a unique name or an article-specific entity, and thus unhelpful in forming clusters. On the other hand, if a word occurs too *frequently*, it is probably a common word and thus contains very little information related to topics.

Thus, we sought lower and upper document frequency cutoffs for including a word in the vocabulary. As a sanity check, we knew our upper cutoff had to be at least high than the document frequency of LeBron James (0.22, he the most famous superstar), since every player should be in our vocabulary. After seeing irrelevant words present with an upper cutoff of even 0.5, we knew that the upper cutoff had to be lower than that, and decided upon 0.23 after experimentation (at this point the eye test was possible). Our lower cutoff was 0.01, which means that if a word appeared in less than 34 out of 3314 total articles, it was discarded. In total, the new vocabulary consisted of about 3500 tokens with document frequencies within the range (0.01, 0.23), including the major players and teams, and, we think, the important, topic-filled sports and NBA terminology. To give a sense, some of the first few (stemmed) tokens printed from our vocab set include *fewest, monday, march, eric_bledso, toronto, decid, blowout, jae_crowd,* and *aggress.*

3 Initial Experiment to Find Most Promising Model

In order to cluster articles for recommendation, it is necessary to embed them in some vector space. There are many ways to approach this problem. For this, project we wanted to dive deep into one model rather than shallow into several. Therefore, we ran an initial experiment on three approaches to find the most promising. LDA turned out to be the most promising and the most exciting, so we leave that to be discussed in the following section. Other than LDA, the two article-embedding approaches we tried involved a word2vec model, and Non-Negative Matrix Factorizatoin (NMF). In this section we briefly discuss them, along with our experiment.

3.1 Word2vec

Word2vec is a modeling approach that aims to embed words as vectors by considering their context in a corpus. In a successful model, words that share common contexts correspond to vectors in close proximity in the space. In our case, we with to embed *articles*, not words, so the approach we tested was to *average the word embeddings in the title of the article.*

3.1.1 Word2vec Pre-processing

Word2vec training requires a set of sentences, not documents, so we began a different data-processing pipeline just for this model, starting from the scraped documents. In addition, we left in all of the words, as opposed to removing them as in the other models. The reason for this is that word2vec infers dependency information between words, and removing stop words would remove that linkage information.

3.1.2 Word2vec Training

The number of word2vec dimensions is a hyperparameter that depends on the application. Most literature recommends a value of 100, and this produced very reasonable results in our case.

The closest vectors to lebron_james are his (all-star) teammates, his hometown, his team name, and his trash-talking nemesis and other all-star rivals.

```
('isaiah_thomas', 0.6015753149986267),
('kevin_love', 0.5609138011932373),
('kyrie_irving', 0.5215520262718201),
('cavaliers', 0.5125935077667236),
('cleveland', 0.5111081004142761),
('russell_westbrook', 0.5105723142623901),
('kanter', 0.508220374584198),
('stephen_curry', 0.5009845495223999)]
```

The model correctly separates teams from the Eastern Conference from teams in the Western Conference, which you would expect from a Bleacher Report corpus with many articles about each conference separately. The model also places team mates in close proximity and can differentiate

between teammates and non teammates.

```
In [13]: model.wv.doesnt_match('houston_rockets utah_jazz golden_state_warriors orlando_magic'.split())
Out[13]: 'orlando_magic'

In [15]: model.wv.doesnt_match('james_harden chris_paul cappella stephen_curry'.split())
Out[15]: 'stephen_curry'
```

3.2 Non-Negative Matrix Factorization

NMF is a topic modeling approach blah blah blah (CYRILLE INSERT INFO HERE)

3.3 Experimental Results: LDA Wins

(WE NEED TO FILL IN THIS SECTION WITH OUR EXPERIMENT DETAILS)

4 Improving our Best Model: Topic Modeling with Latent Dirichlet Allocation

We spent the rest of our project diving more deeply into finding a quality LDA model, and using that to explore the landscape of topics for Bleacher Report NBA articles.

4.0.1 Background on LDA

In the following section, define
$D = \{d_1, d_2, ...d_N\}$, the set of N documents (articles) obtained from Bleacher Report
$V = \{w_1, w_2, ...w_M\}$, the set of M words in the final vocabulary
$T = \{t_1, t_2, ..., t_K\}$, the set of K topics in the documents

The first way we approached the question of document similarity was topic modeling. In topic modeling, we attempt to infer which *topics* are present among the documents, and then after that, to identify which topics belong to each particular document, and in what proportion. Here a *topic* is simply a probability distribution over the words in the vocabulary. For each topic, each word has a probability of being chosen. The task is difficult, because we are trying *both* to infer the topics themselves (all we provide in advance is K, the number of topics), *as well as* the topic-composition of each document.

We decided upon a specific Topic Model known as Latent Dirichlet Allocation (LDA). LDA assumes that documents are created according to the following generate process:

1. Randomly generate a distribution over topics[1].
2. To come up with each word in the document:
   a. Randomly choose a topic.

[1]This requires a distribution over a distribution. The topic distribution is assumed to have a Dirichlet prior, giving the model its name

b. Randomly choose a word from that topic.

The main task of LDA is then to infer two parameters:
β_t: the distribution over words for topic t: $\mathcal{P}(w|t) \quad \forall w \in V$
θ_{dt}: the proportion of document d that came from topic t: $\mathcal{P}(t|d) \quad \forall t \in T$

LDA infers these parameters according to which values make the data the most likely.

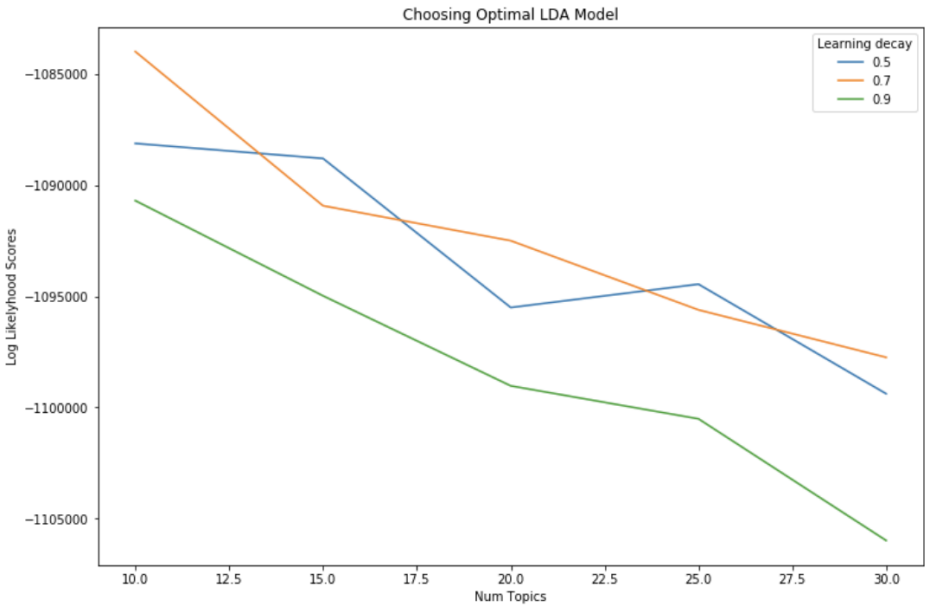
These estimates allow us to express each document as a vector, which is a distribution over the topics it contains. The vectorized documents can then be clustered based on their topical composition. The idea is that if I love the topic of statistics and I also love the topic of Lebron James, I will be recommended articles about Lebron James' statistics.

This is a bag-of-words model because the generative story says nothing about grammar. For topic modeling, we do not care about grammar anyway.

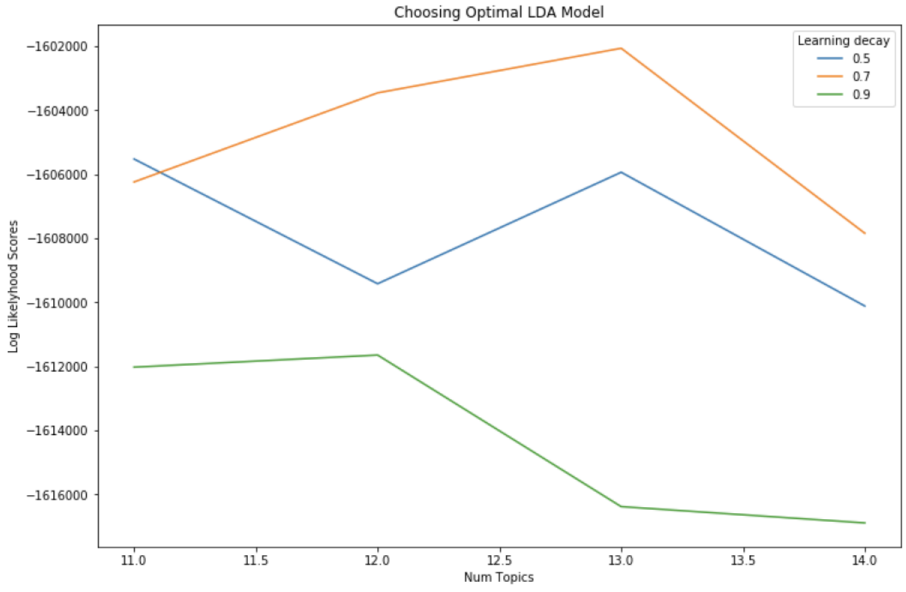
4.0.2 Training LDA with Grid Search

We trained LDA models on the cleaned Bleacher Report articles using Python's sklearn libraries. When training the models, two hyper-parameters needed to be fine-tuned. The first was K, the number of topics to ask for. The second was γ, the learning rate of the solver. (TALK MORE ABOUT THIS?). We decided to use a grid search cross validation approach. Since topic modeling is an unsupervised problem, there was no need for a separate validation set. Instead we used the log-likelihood of the trained model as our measure of model quality.

We searched $k \in \{10, 15, 20, 25, 30\}$ and $\gamma \in \{.5, .7, .9\}$.
Here is a chart of the GridSearch results, the log likelihoods for each parameter duo.



After discovering that the best hyper-parameters were found to be $k = 10$, $\gamma = 0.7$, we decided to narrow down the grid search further, since we cared a lot about finding the highest-likelihood number of topics. We searched $k \in \{10, 11, 12, 13, 14\}$ and still $\gamma \in \{.5, .7, .9\}$.



After this last grid search, we moved forward with the optimal $k = 13$ topics (and $\gamma = 0.7$).

4.0.3 LDA Results and Visualization

We were very pleased with the results, which we visualized using LDAvis, a method for visualizing LDA topics developed by Carson Siever and Kenneth Shirley [2].

The visualization has two primary components.

On the left, the blue circles represent the topics. The size of each circle represents the proportion of its topic occurance across all documents. The positions of the circles is meant to be a 2D projection of their locations with respect to each other in the topic space. Specifically, a matrix of inter-topic distances is generated, [3], where each row in the matrix contains one topic's distances to every other topic. The K-dimensional distance rows are then shown in 2 dimensions using Principal Component Analysis [4] This visualization technique is known as *multidimensional scaling*, and typically performed on distance matrices as in this case.

On the right, the red bars indicate the estimated number of times a given term came from a given topic. When you hover over a topic, the 30 most relevant terms from that topic are shown.[5] Our visualization is interactive, and we recommend that you run it in our notebook, RUNME.ipynb. However we we show a few topics here.

Second, it shows the top-30 most salient terms for each cluster, and for the documents overall. When you hover over a topic, the top 30 most salient terms for that topic are shown in decreasing order.

As avid NBA fans, we were very pleased to discover that our topics made great sense. The largest topic, 1, is all about player statistics (steals, blocks, defensive and offensive rates, ranks, percentages, etc.). It makes sense that this is the most prevalent topic in a corpus full of sports articles.

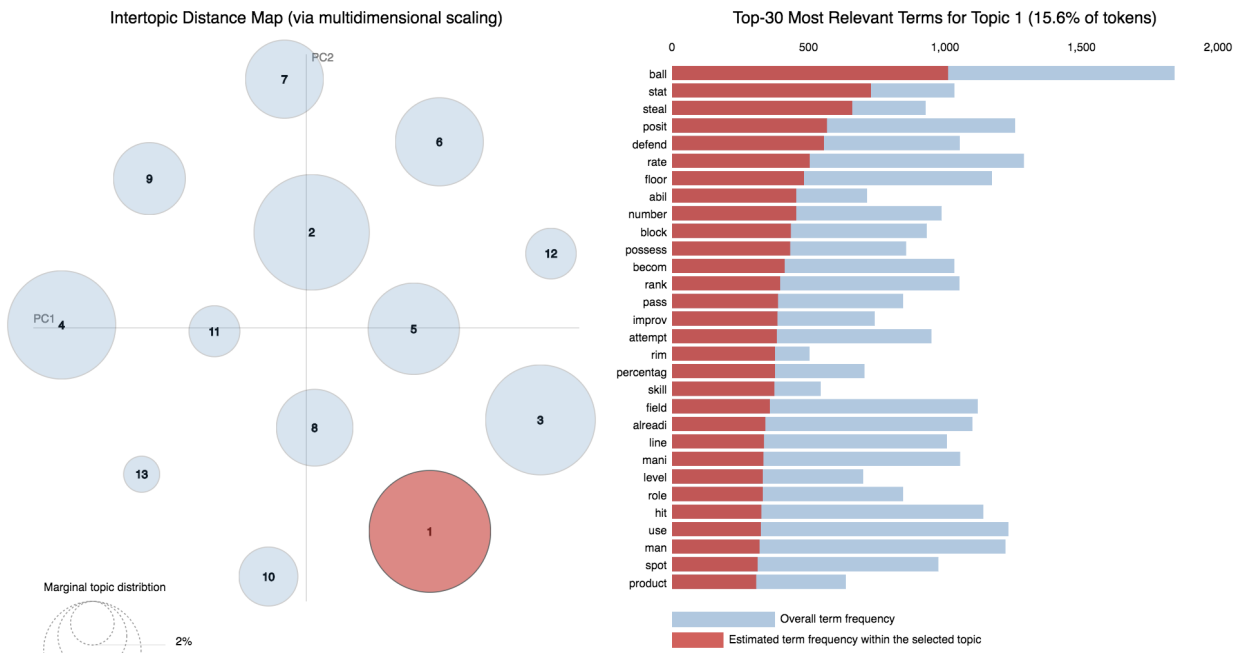
---

[2] Sievert, Carson and Shirley, Kenneth. "LDAvis: A method for visualizing and interpreting topics". *Proceedings of the Workshop of Interactive Language Learning, Visualization, and Interfaces*, pages 63-70, Baltimore, Maryland, USA, June 27, 2014. Their Github can be found here

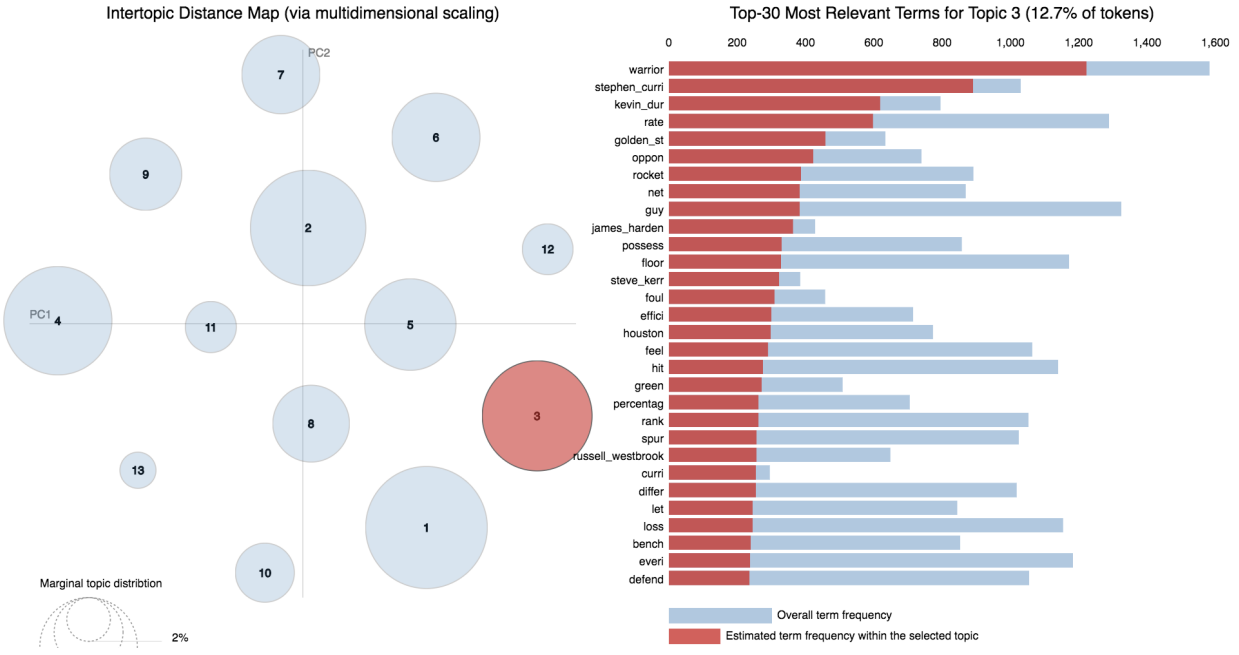
[3] Remember that a topic is a probably distribution over all words in the vocabulary. The distance between two topics is found by computing the Jensen-Shannon diveragence between their distribution.

[4] Though our topics make wonderful sense, we cannot detect any semantic meaning in the first two principal components themselves.

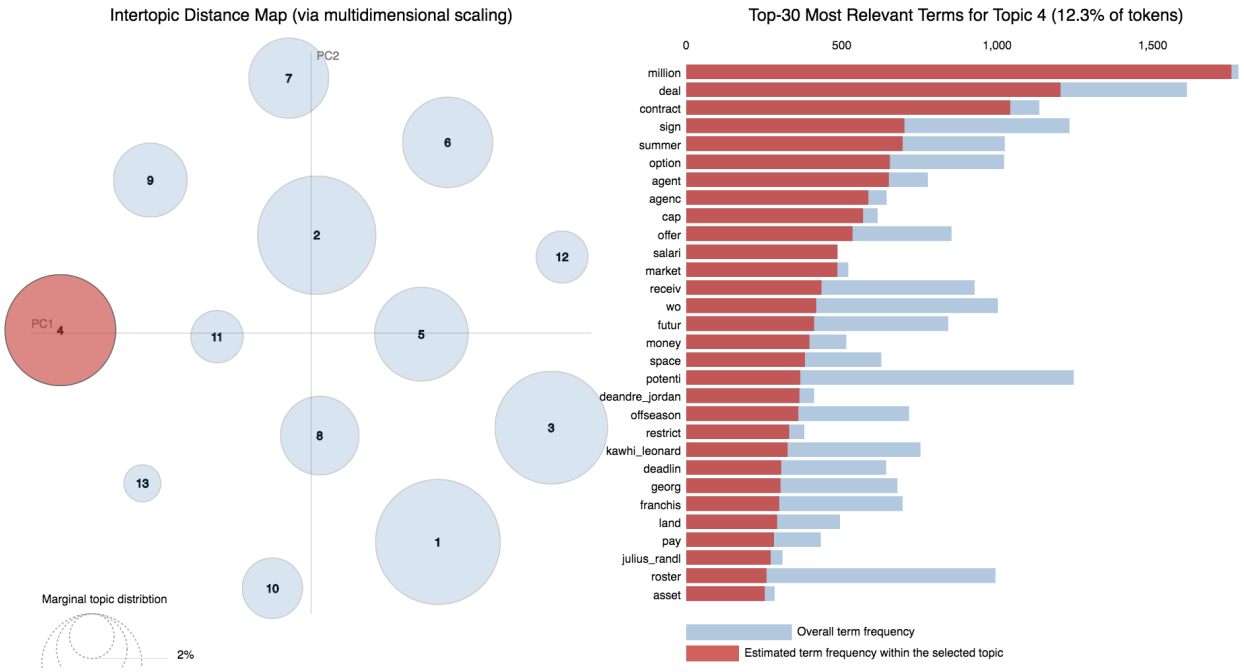
[5] relevance$(w|t) = p(w|t) + p(w|t) * p(w)$



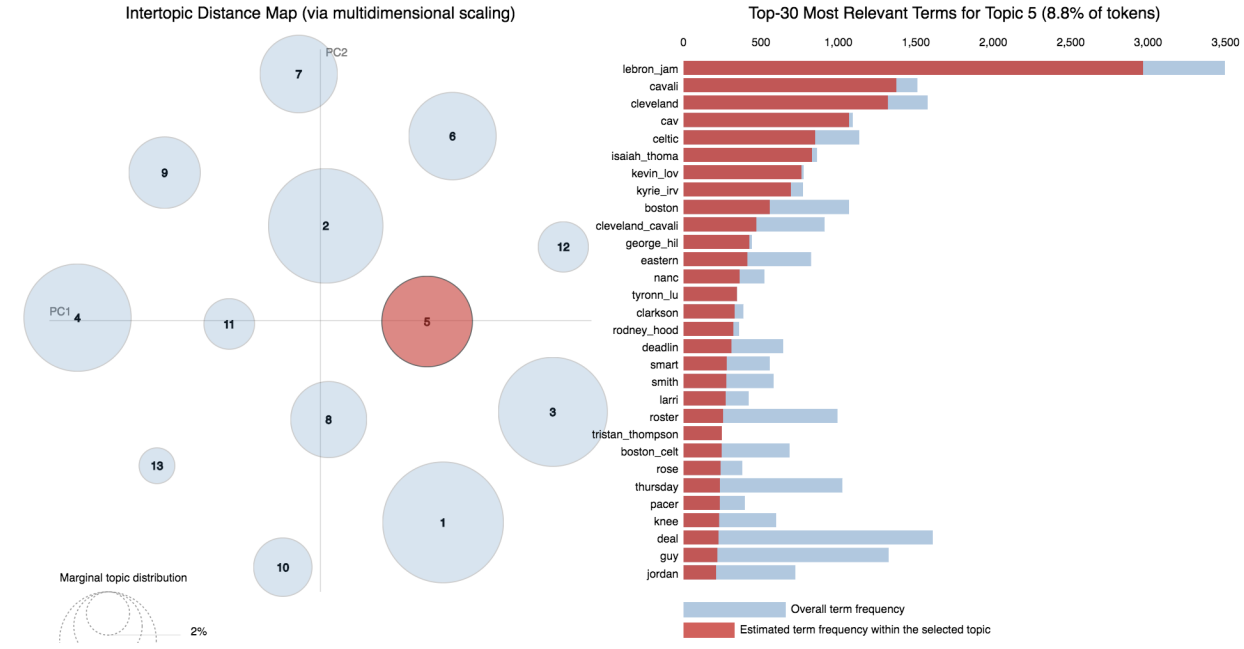
To its right, topic 3, seems to be about the Golden State Warriors, the best team in the NBA (mostly because of superstars Stephen Curry and Kevin Durant), and their competition in the Western Conference (including James Harden's Rockets and Russel Westbrook's Thunder).



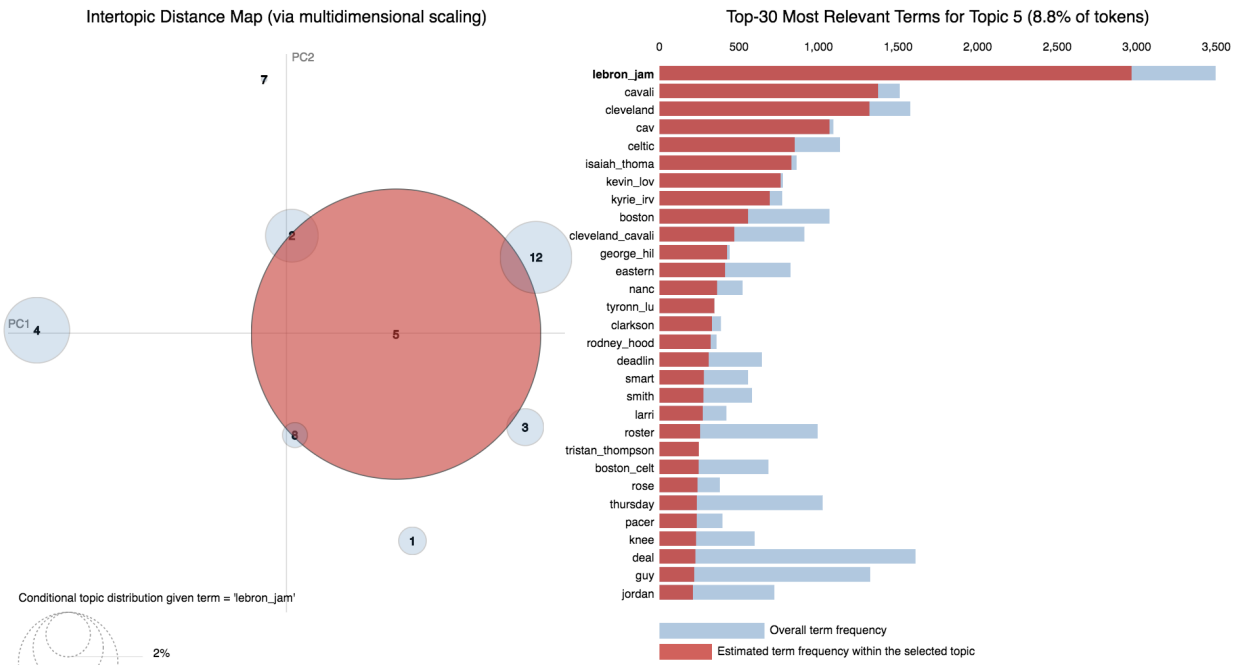
Topic 4 is all about the economics of the NBA, including trade deals, free agency signing decisions, and salary information.



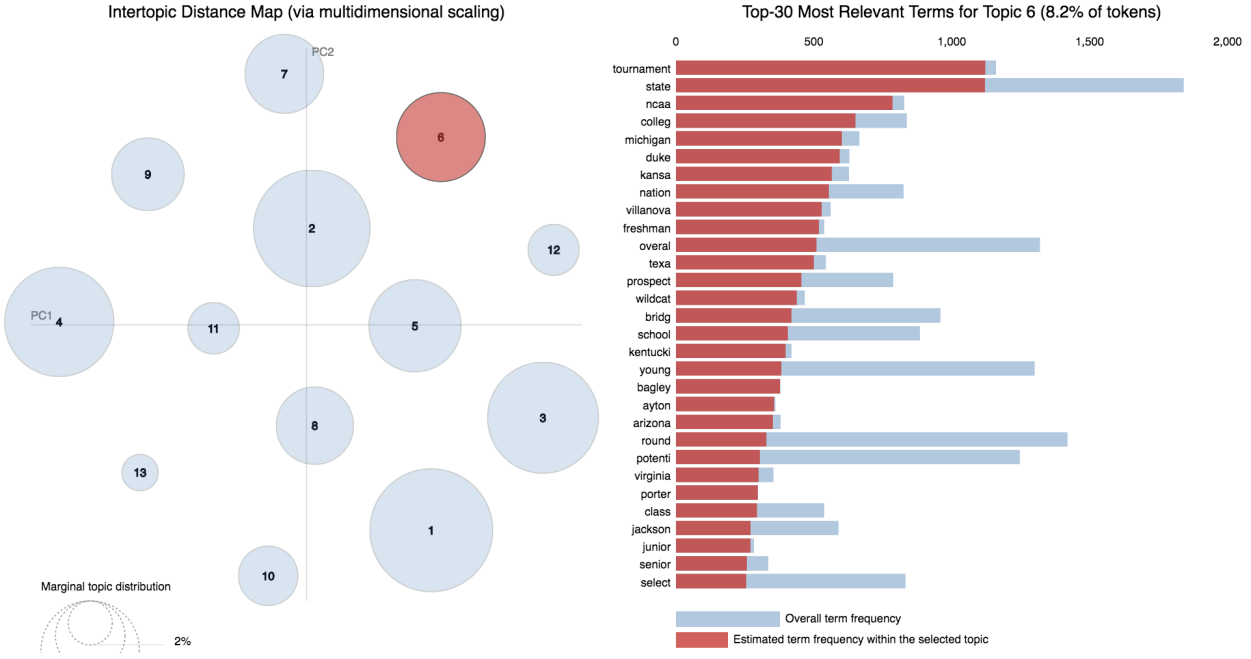
Topic 5 is our personal favorite, as it is all about Lebron James. The most salient words in topic 5 are Lebron, his team mates, his team name, his coaches, and his major rivals.



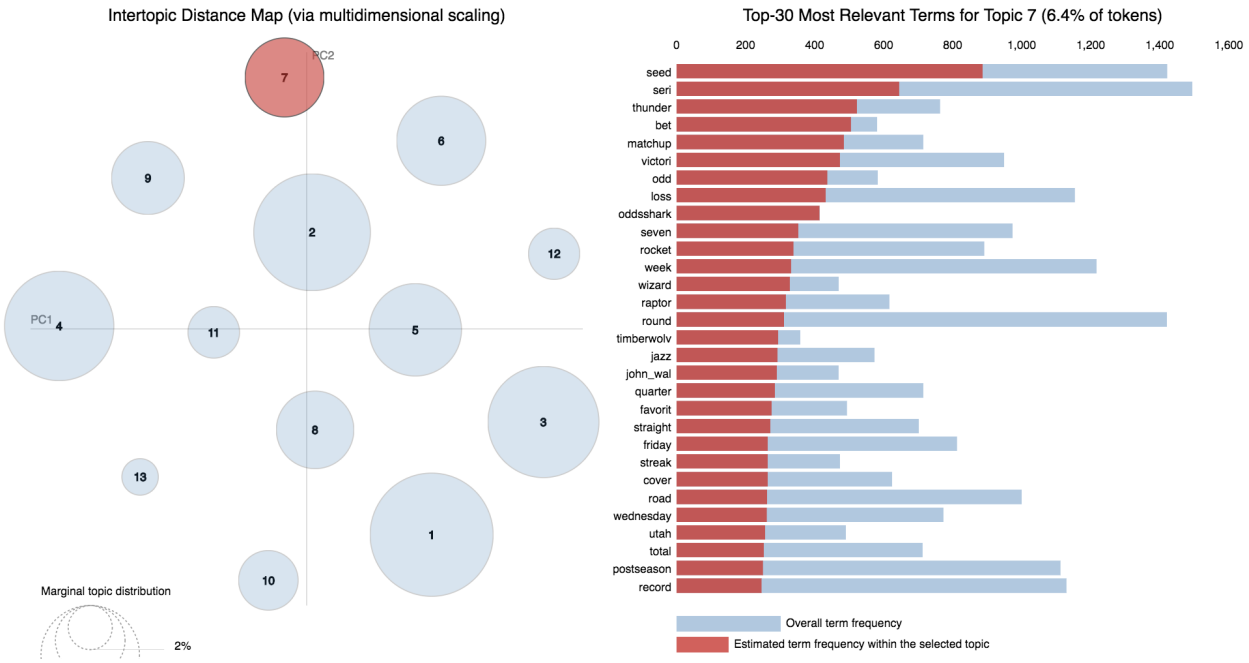
In fact, if you hover over the term *lebron_james*, you can see his primary topics. Topic 5 dominates as his primary topic.



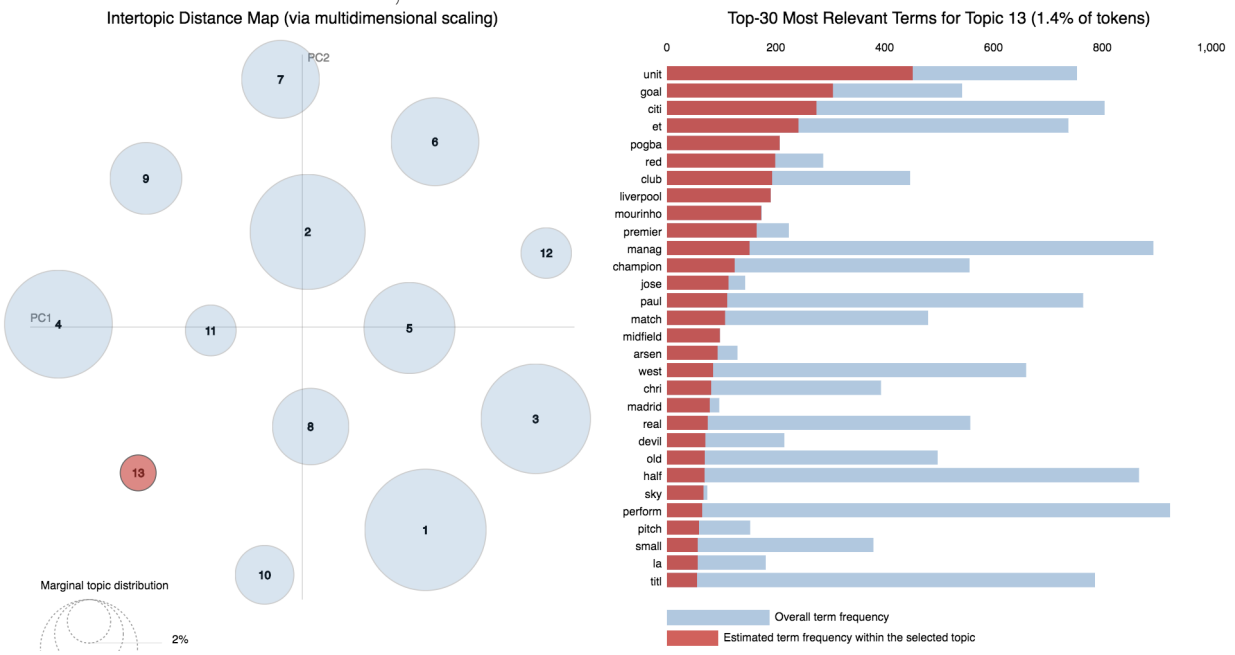
Topic 6 collected the terms pertaining to the NCAA March Madness college basketball tournament, which pushes aside the NBA in popularity for a brief month as the college kids take center stage. Michigan played Villanova for the championship this year.



Topic 7 seems to be about the playoffs, seeding, and exciting first-round matchups.



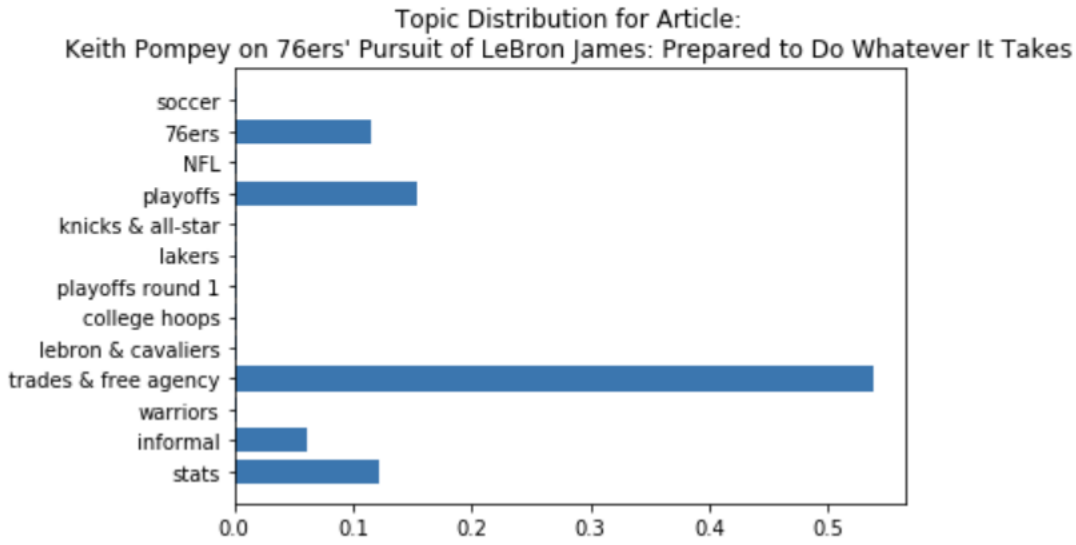
Topic 13 is actually about soccer! The NewsAPI must have returned a handful of soccer-related articles with the basketball ones, and our LDA model discovered that.



We have left the other topics in the appendix for brevity.

5 Producing a Recommendation

Let us see our recommender system in action. We took a new article from May 13, 2018, entitled *Keith Pompey on 76ers' Pursuit of Lebron James: Prepared to Do Whatever It Takes*, and expressed it as a distribution over the 13 topics using our trained LDA model. Here is the distribution over topics. We've labeled each topic according to what we think it primarily represents (as specified above and in the appendix).



Not suprsingly, this article is mostly about free agency and trades. Also, notice that the topic of the 76ers is prominent.

Then, we found the nearest article to it using L2 norm, from all articles in the Bleacher Report corpus expressed in terms of their topic mixtures. Here's are the top 5 recommendations:

```
Dennis Lindsey on Suns-Jazz Fight: 'You Expect Players to Be Protected' by NBA
'I Feel That I Was Chosen to Do It': Chris Bosh Gets Serious About NBA Comeback
NBA Free Agents 2018: Top Rumors, Speculation and Predictions
Predicting Fallout from LeBron James' 2018 Free-Agency Decision
6 Bold Predictions for 2018 NBA Free Agency
```

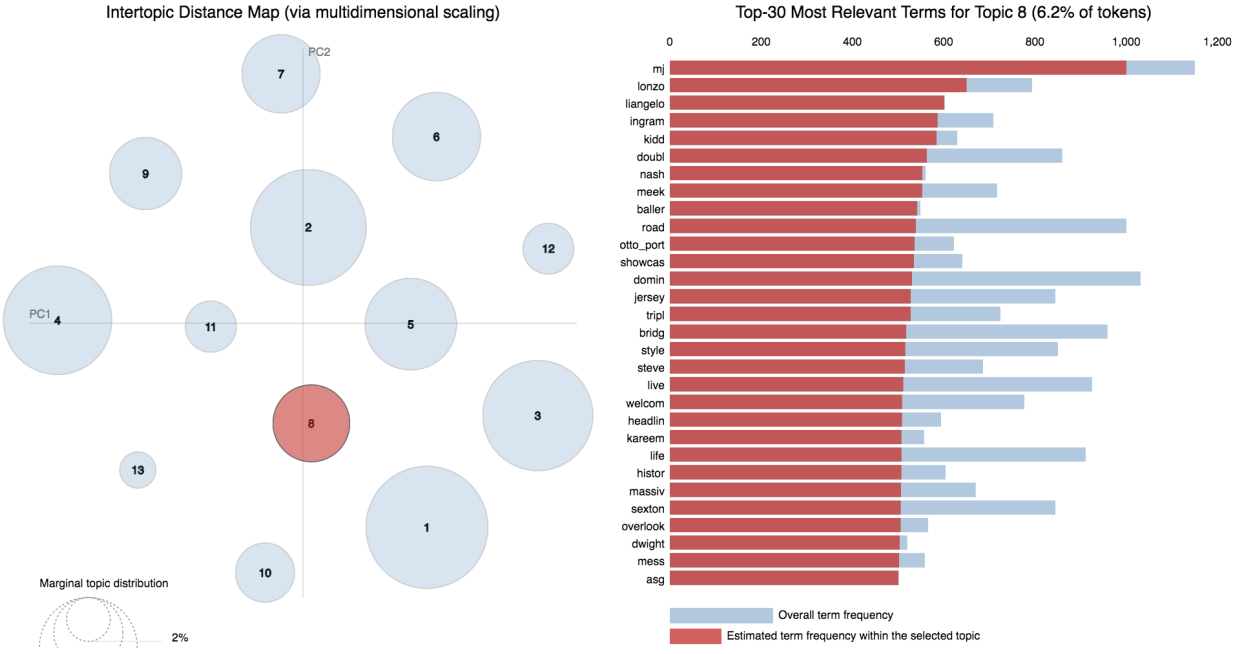
While some articles feel like they are not relevant (notably the first two), notice that the other three recommendations are on-point in discussing Lebron James' upcoming free agency decisions.

6 Discussion

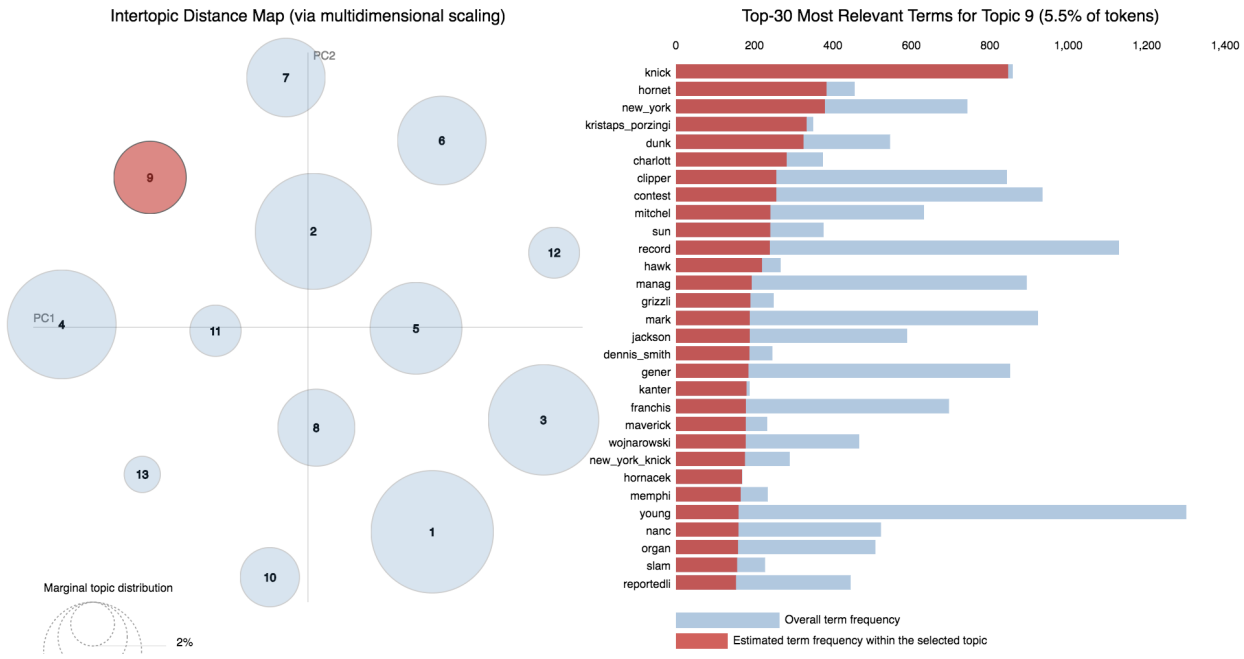
Most of the world's data is in the form of unstructured natural language, and using Natural Language Processings provides an analytics edge in many domains, not the least of which is online advertising. It is no surprise that NLP is one of the hottest fields in research right now. In this project we learned a lot about NLP, from data cleaning and data scraping to building topic models on a corpus of articles. We learned a lot about three common ways of embedding documents in a vector space, and then took a deep dive into training and visualizing one of them, Topic Modeling with Latent Dirichlet Allocation. The vector-space article embedding was motivated by a desire to cluster articles to produce quality recommendations. If hired by Bleacher Report, our first step would be to do a live test of our LDA-based recommender system.

7 Appendix

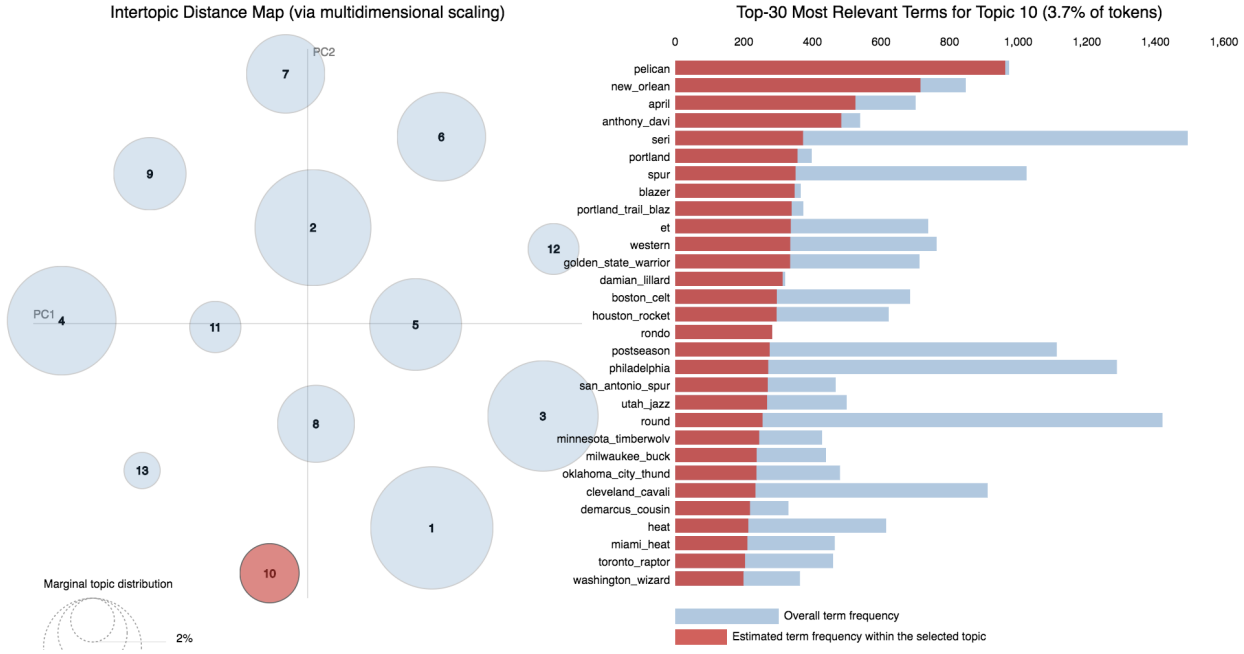
Topic 8 seems to capture the multiple stories around the Los Angeles Lakers this season, always a popular NBA team.



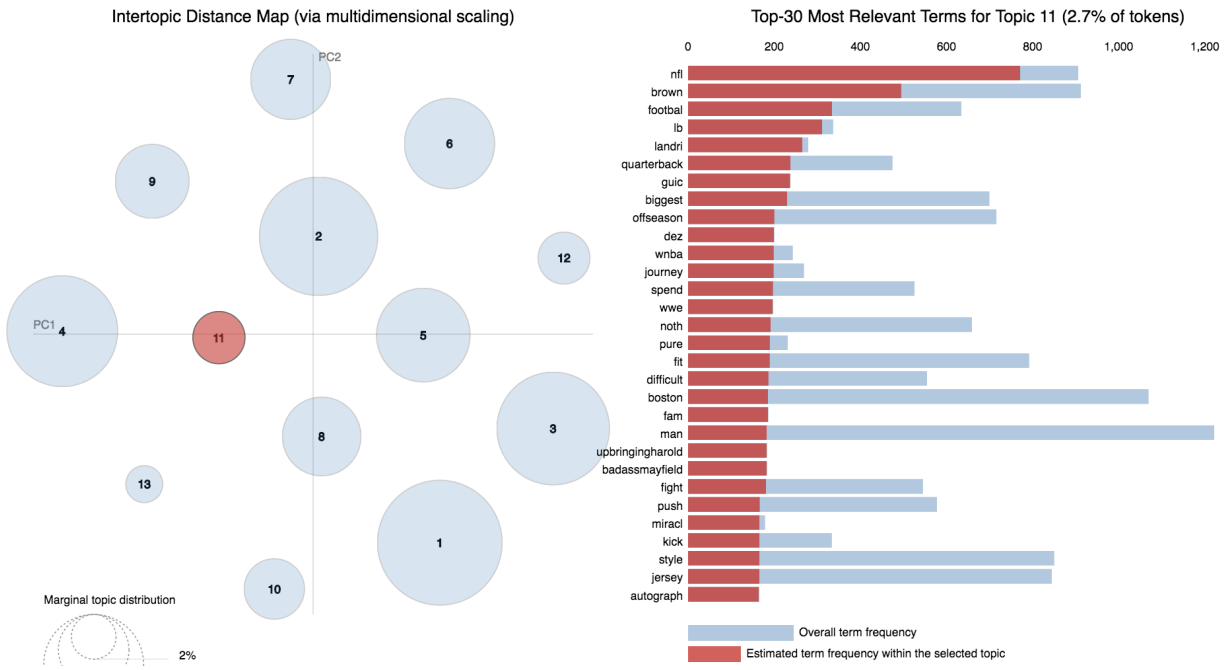
Topic 9 seems to merge two distinct topics, in our opinion - the all-star game and the New York Knicks.



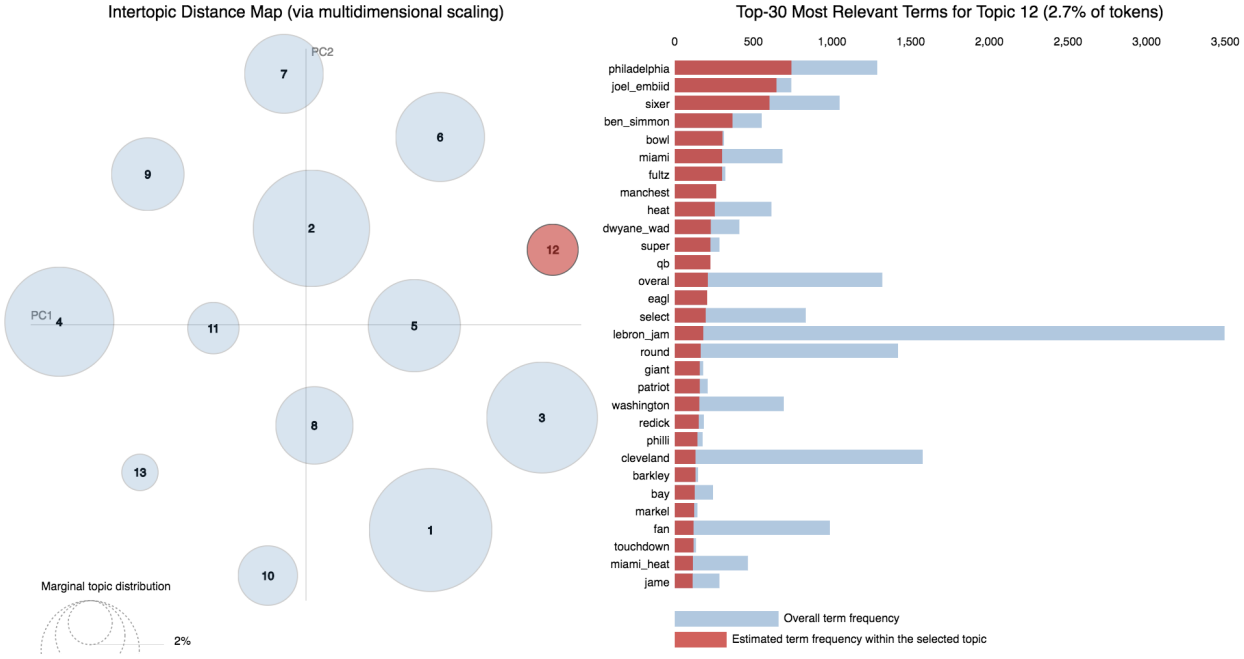
Topic 10 is about the playoffs, and includes the most notable postseason teams.



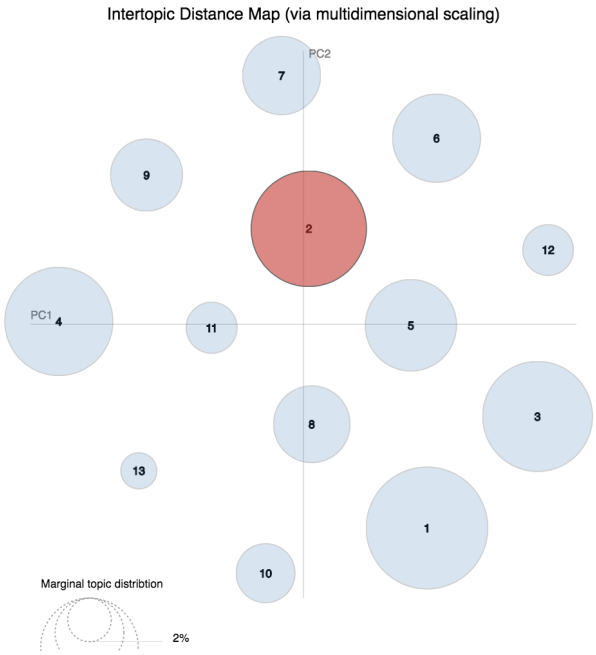
Topic 11 is actually about the National Football League! The NewsAPI must have returned a handful of football related articles with the basketball ones, and our LDA model discovered that.

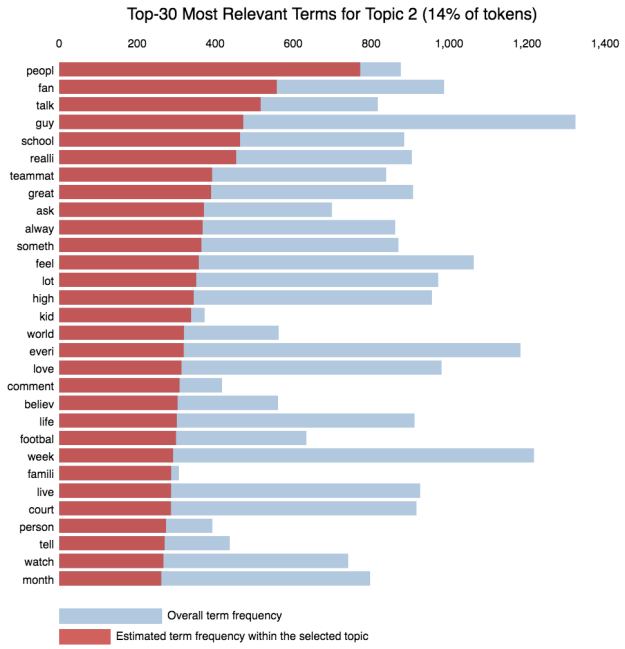


Topic 12 seems to be about the Philadelphia 76ers and their rivalry with the Miami Heat this season. The 76ers have been one of this seasons biggest sub-stories because they have two budding young superstars and are seeking to land Lebron James during free agency this summer.



Topic 2, located in the middle of the topic clusters, seems to capture informal language.


Intertopic Distance Map (via multidimensional scaling)
PC2
PC1
7
6
9
2
12
4
11
5
3
8
13
1
10
Marginal topic distribtion
2%


Top-30 Most Relevant Terms for Topic 2 (14% of tokens)
0
200
400
600
800
1,000
1,200
1,400
peopl
fan
talk
guy
school
realli
teammat
great
ask
alway
someth
feel
lot
high
kid
world
everi
love
comment
believ
life
footbal
week
famili
live
court
person
tell
watch
month
Overall term frequency
Estimated term frequency within the selected topic