Unsupervised Learning Techniques to Cluster Stock News Headlines

Introduction

Stocks frequently see quick price movements after following big news. However, looking at many news headlines, many editorial and opinion type articles are mixed in with news on actual events. For example, an article titled “Five Stocks to Include in Your Retirement Portfolio” is unlikely to cause a price movement whereas articles reporting significant events about a company such as a new product offering, a change in an important leadership role, or recent earnings reportings are far more likely to trigger significant price movements. In this project, I will explore various unsupervised learning techniques to cluster stock news headlines in an attempt to separate editorial and opinion articles from event-driven articles.

The main objective of this analysis is to be able to separate stock news headlines that are more editorial in nature and don’t describe an event that may affect a company’s stock price.

Data Background and Preprocessing

The data were gathered using the EOD historical data API. All news articles in which a NASDAQ stock was tagged were gathered for a couple months in 2023. The data includes the headline, content, and sentiment analysis values for the article. In total, 2601 articles were collected.

The title only was used first because there should be enough information in the title to decide if the article is editorial in nature. First, it was observed that duplicate news articles were gathered so they were removed, bringing the total count of news articles down to 1695.

Next, non-word characters and numbers were removed and the data were converted to all lower case as is common in natural language processing. Finally, the data were further processed using some features from the Natural Language Toolkit (nltk). Stemming was applied, which converts words back to their root form. For example, “stopping”, “stopped”, and “stopper” can all be converted back to their root form “stop”. Stopwords were removed using nltk as well. Stopwords are words that carry little or no meaningful information in a sentence such as “the”, “of”, “and”, etc.

Finally, the transformed titles were turned into a sparse matrix using the Term Frequency-Inverse Document Frequency (TF-IDF) feature of scikit-learn.

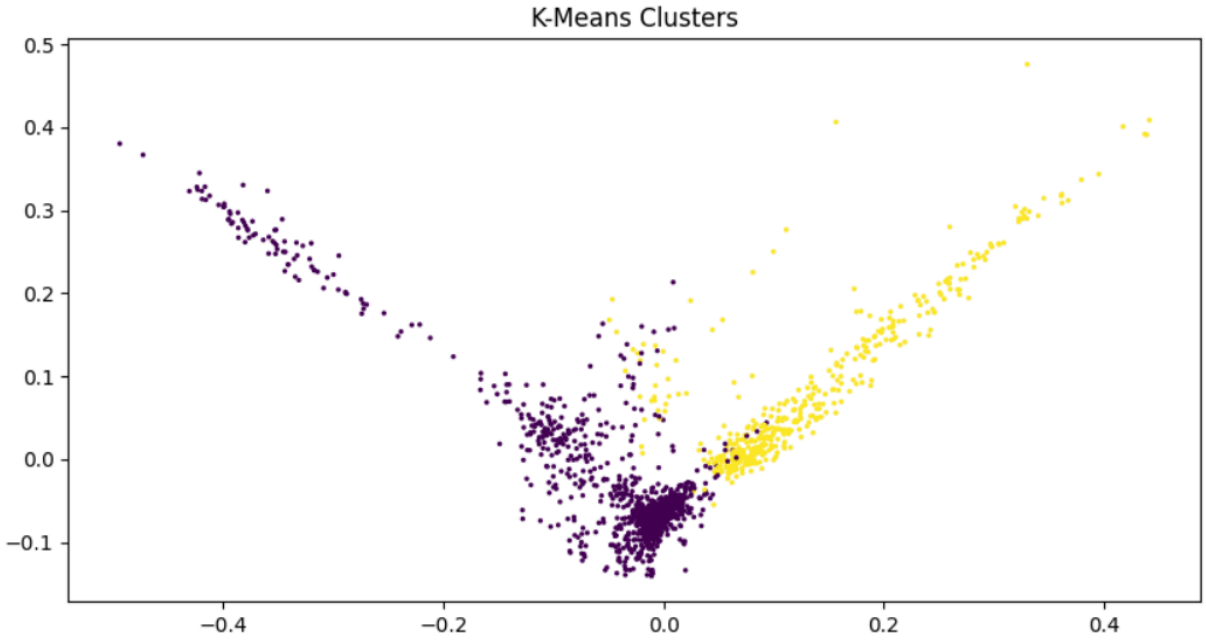
Unsupervised Learning Methods

Kmeans

Kmeans clustering is an unsupervised machine learning algorithm that partitions data into a specified number of clusters (k) by iteratively assigning data points to one of k clusters based on the nearest centroid point, and then updating the centroid as the mean of the assigned points until convergence.

Kmeans was applied to the sparse matrix specifying two clusters. This resulted in one cluster of size 1239 and the other of 456. A few of the headlines assigned to each cluster were investigated. The smaller cluster seemed to contain editorial headlines such as “Buying These 3 Stocks Could Be the Smartest In…” and “Got $1,000? 3 Top Growth Stocks to Buy That Co…”. The larger cluster seemed to contain news about actual events such as “American Airlines’ Pilots Ratify Contract With…” and “SoftBank rises as chip unit Arm files for Nasd…”.

We can visualize the results by plotting a PCA and coloring each cluster a different color:



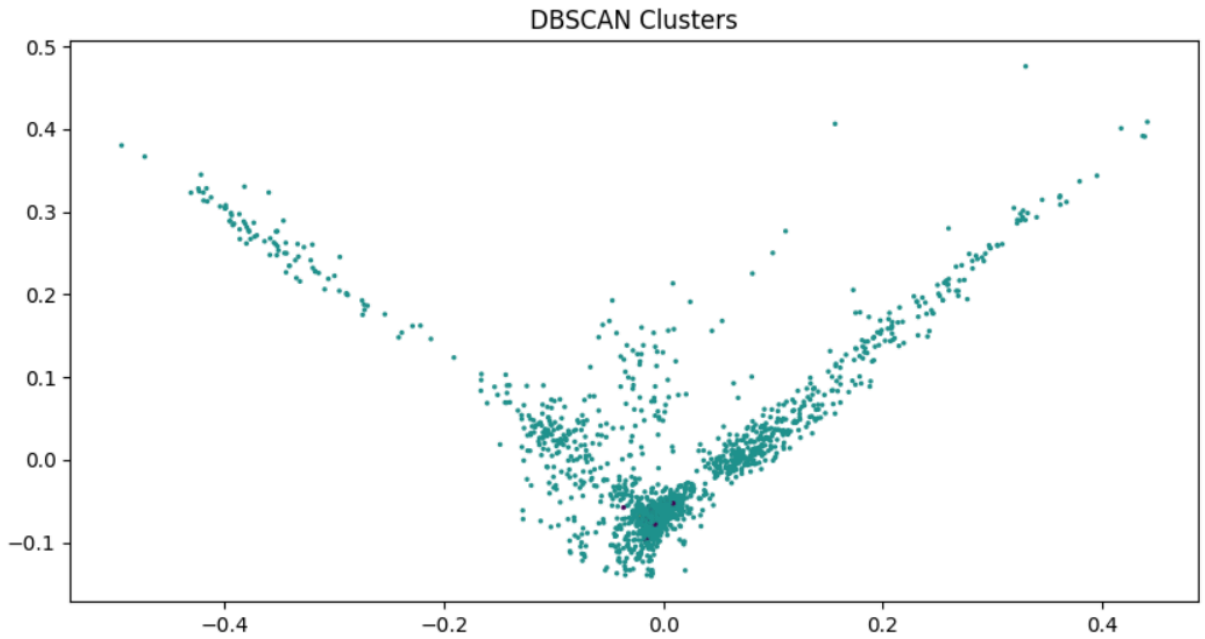
The PCA looks really good in that you can see the separation between the two clusters.

These results were promising but need to be more thoroughly inspected. First we will try other unsupervised methods.

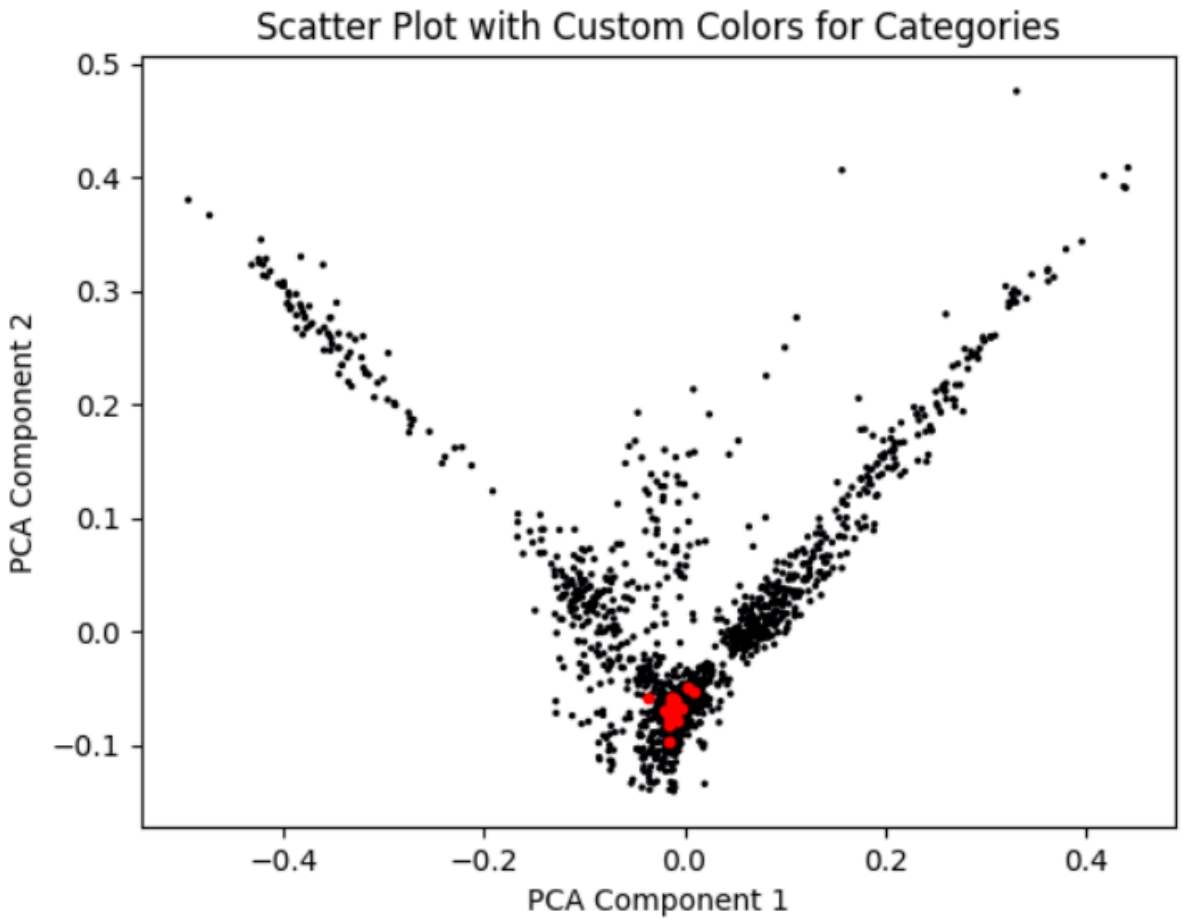
DBSCAN

DBSCAN is actually a clustering algorithm rather than a partitioning algorithm like Kmeans and Hierarchical Cluster in that not all data points necessarily need to be assigned to cluster. Rather than number of clusters, the user specifies a maximum distance (epsilon) points can be apart in order to be considered part of a cluster and a minimum number of points (min_samples).

For this problem, we know we want two clusters and having read many of the news titles and I have a rough idea that most can be classified as editorial or news. Furthermore, there are probably more editorial articles than real news articles. So we can use the number of clusters and number of noise points as a metric to help us reach a satisfactory epsilon and min_samples. The two input parameters, epsilon and min_samples, can be difficult to optimize and good values are not intuitive to estimate. So I wrote a loop to cycle through lists of epsilons and min_samples and make a DBSCAN model for each combination of epsilon and min_samples in the lists. The loop was ran three time, adjust and fine tuning the two variables. It was observed small changes in the two variables can produce big results with our metrics. It was found that an epsilon of 1.3 and min_samples of 7 produces two clusters with only 3 noise points. However, the sizes of each cluster were disappointing: one cluster had 1657 points and the other only 35.

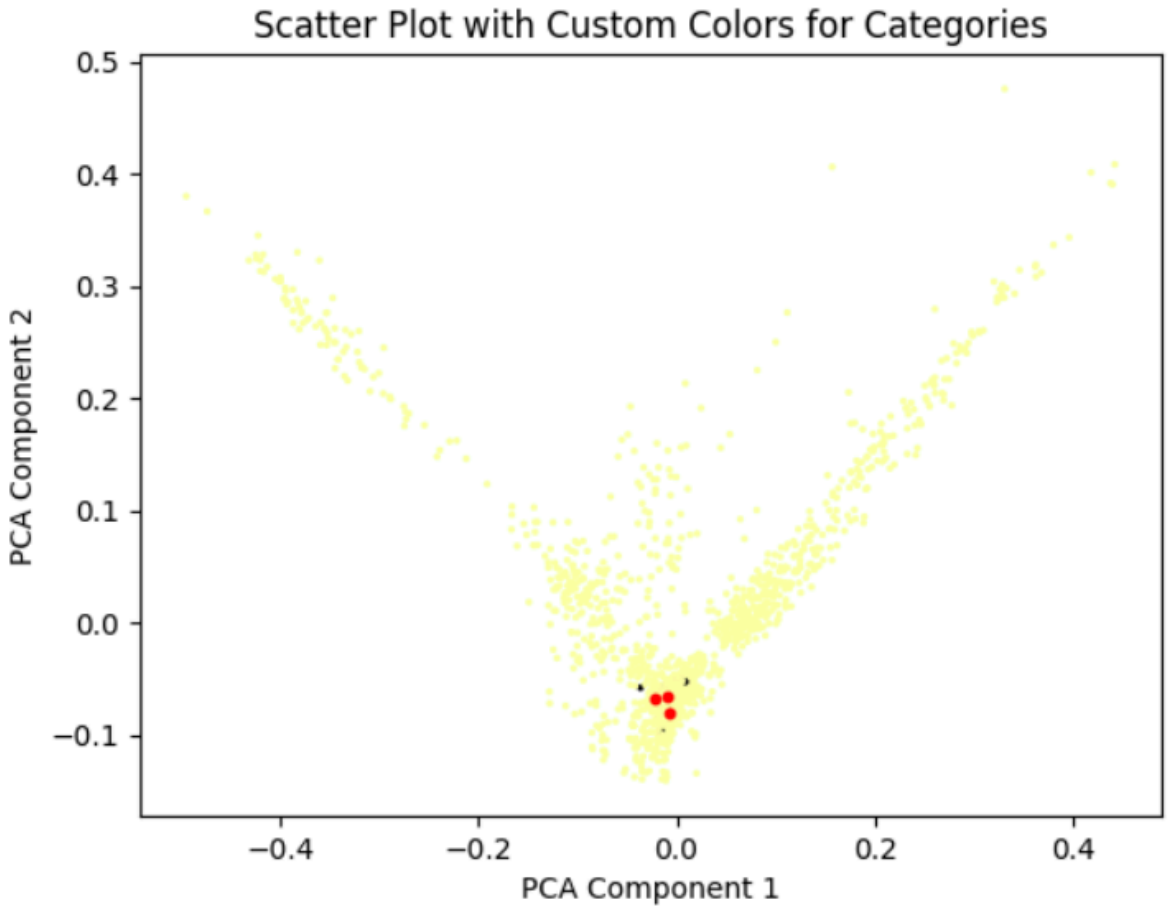


The location of one of the clusters is difficult to see so I remade the PCA, plotting the large cluster first and then plotting the smaller cluster on top with larger points and a more contrasting color for better visibility.



In the PCA we can see that the data points that were labeled as a separate cluster (in red) are in the middle area where the headlines are probably more ambiguous. This further supports that DBSCAN is the wrong tool for this problem.

Out of curiosity, I investigated where the noise points are on the PCA. In the following plot, the noise points are colored in large red circles:



It's very interesting that the smaller cluster and noise points exist in an area of high density on the PCA. This may indicate that the PCA does not capture a lot of the variability in the actual data.

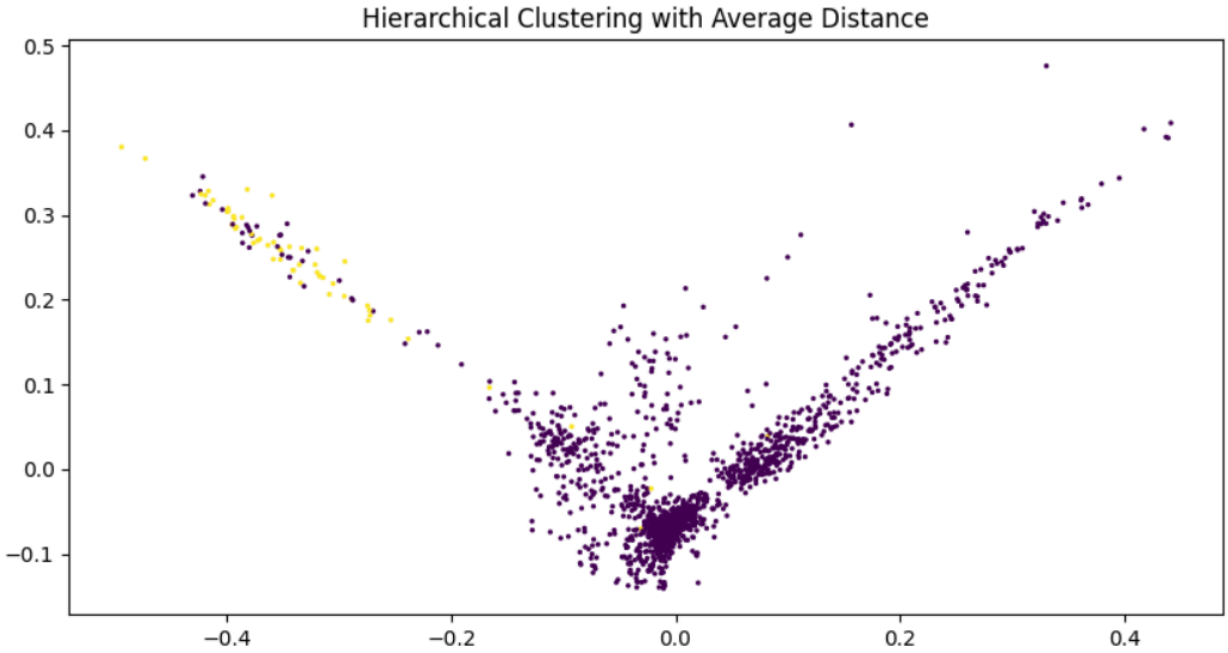
Agglomerative Hierarchical Cluster

Agglomerative hierarchical clustering is a "bottom up approach" in that it beings with each data point being separate and then groups them one by one to form clusters until they reach a specified number of clusters. In this way, you get a hierarchy of points that are closer together by whichever distance metric you choose to use. Another parameter that requires specification is the linkage, which governs how distance is calculate from an already formed cluster

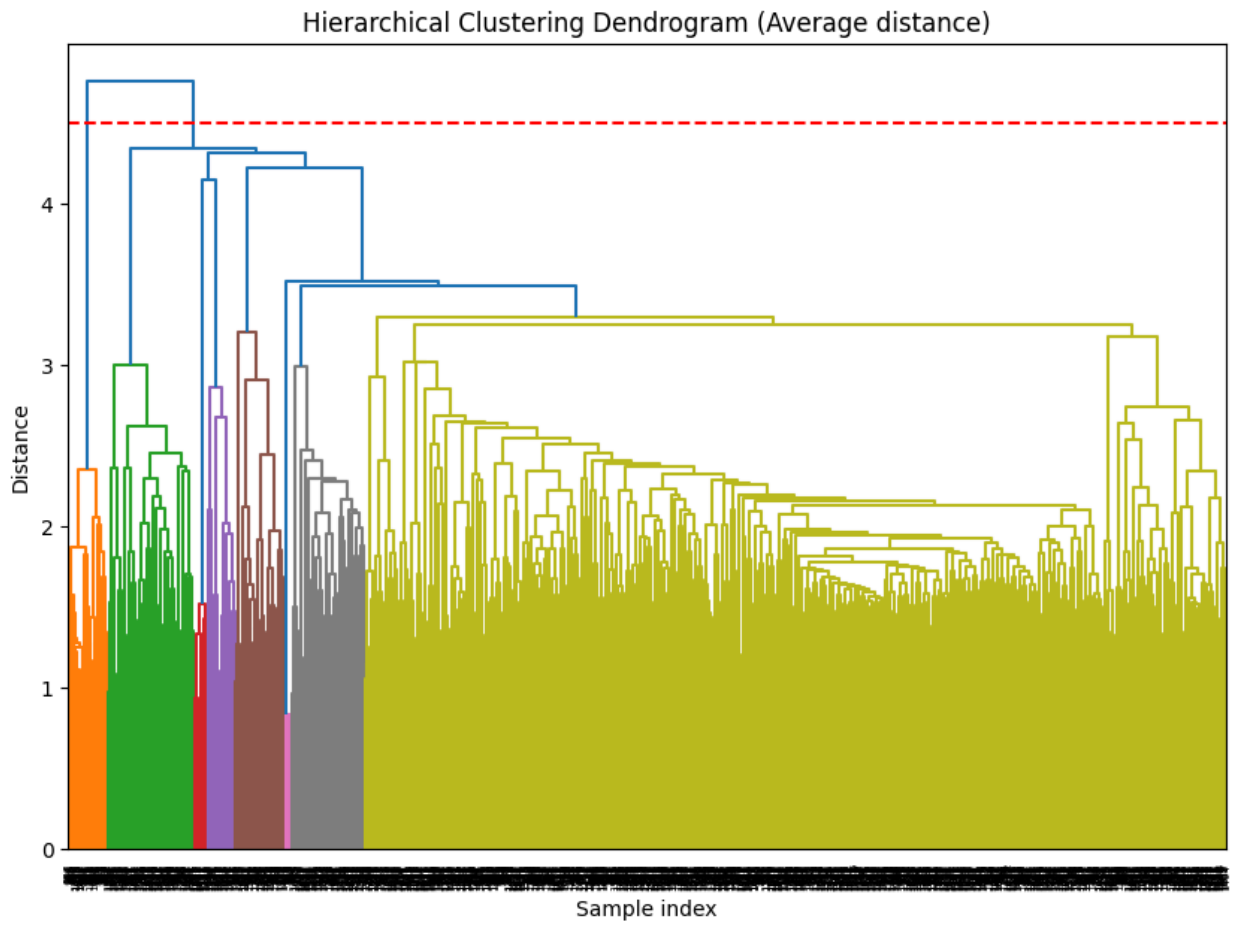
For this project, I explored two types of distances/linkage combinations:

1. Euclidean with ward linkage
2. Cosine distance with average distance

Using Euclidean distance with average distance and ward linkage gave some, but not great, separation of the two expected clusters. There were 1636 data points in one cluster and 59 in the other.



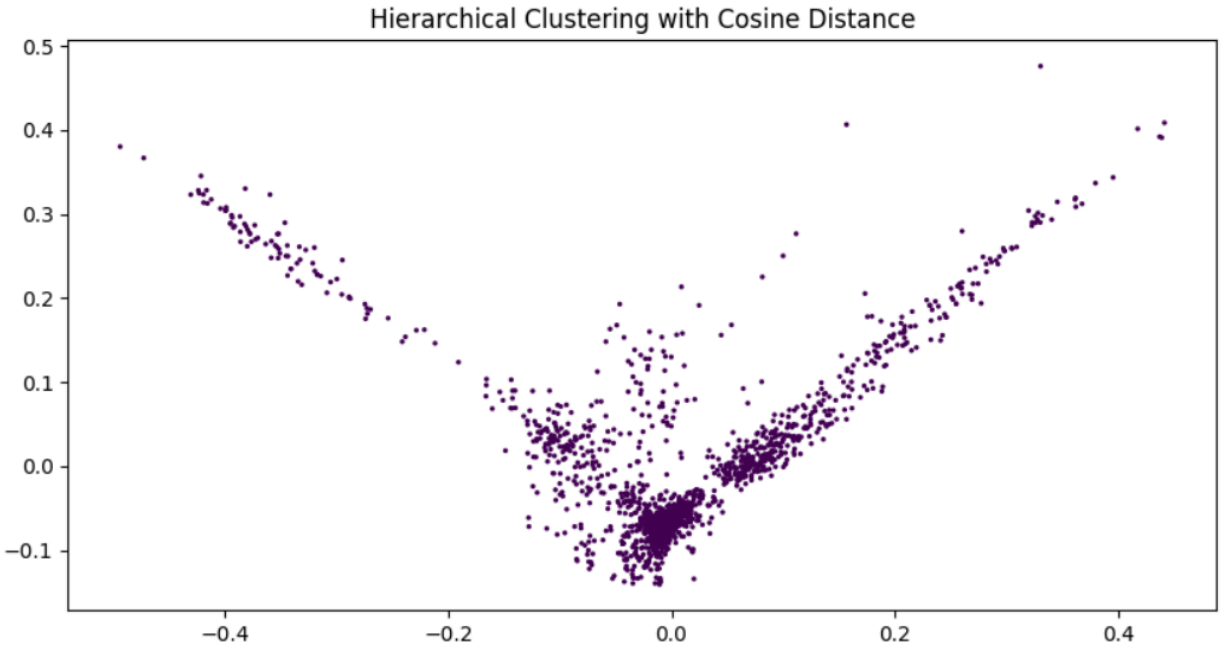
Looking at the PCA, it did pick out points for one cluster that are far from the center (top left) so it seemed to recognize the most obvious points. However, I am certain there are far more editorial headlines that should be clustered.



In the above dendrogram, the hierarchy of clusters is drawn by default until it merges into one cluster so I drew a red line where the hierarchy would be cut to get two clusters. This means that when there is only two clusters, the orange data is our separate cluster (yellow dots on the above PCA) and the green, red, purple, brown, pink, grey, and yellow clusters are all together before they merge with the orange.

The clustering could potentially be improved if we artificially merge a few of the other colors like the green, red, and purple and see if those colors contain editorial articles. But at this point, other avenues seem more promising such as Kmeans or other distance metrics for hierarchical clustering.

Cosine distance in hierarchical clustering performed poorly and only separated one headline into a cluster.



Key Findings

- Kmeans clustering performed best at separating editorial headlines from actual event-driven headlines. Clearly, this model suits the main objectives of this project best.
- DBSCAN clearly was not the right tool for the problem. It became clear that we are looking for a partitioning algorithm rather than pure clustering algorithm.
- Hierarchical clustering using average distance and ward linkage performed better than using cosine distance but still not as good as Kmeans. This was a little surprising as cosine distance is often used with text data.
- Upon manual inspection of the headlines clustered by Kmeans, one group was nearly all editorial articles which is good. The other group, however, contained a mix of event driven articles and editorial articles. So this model would have a high false negative rate for detecting editorial articles. This may be unideal when trying to identify buying opportunities for stocks with recent news.

Possible Flaws and Future Follow Up

A potential flaw in this study is that the classification is assumed to be binary. That is,it assumes all news articles are classified into either editorial or event-driven. But it's possible some are in a gray area in between. For example, there are headlines such as "Q3 Earnings of Company XYZ to be Reported: Here's What to Expect" which covers an upcoming event (an earnings report)

but also adds an opinion. Another flaw is that only headlines were used. We may have gotten better results if the entire content was considered. Finally, the TD-IDF algorithm is fairly simply in that only word counts are consider. An algorithm that better understands the context of words or the meanings of sentences may have achieved better results.

Unsupervised methods were used for this problem simply because the data was not labeled as being editorial or event-driven. Follow up studies could attempt to label the data by hiring someone to go through several hundred articles and decide which are editorial and which are event-driven. That would turn this into a supervised learning problem. A follow up project could also attempt to use a more sophisticated algorithm that understands the context of words in sentences such as a recurrent neural network.