Objectives:

- Segment publicly traded stocks into a set of like-groups
- Outline workflow, techniques used, decisions made, reasoning, and visualizations

Workflow Overview:

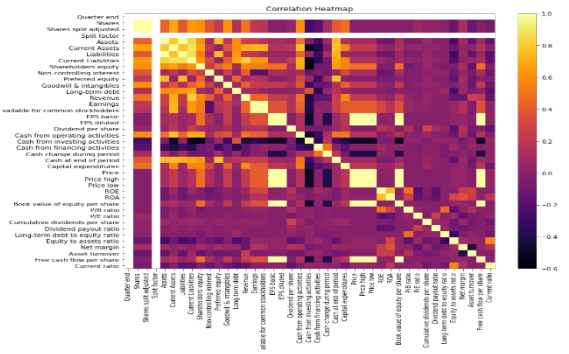
1. Data Setup – Packages, Helper Functions
2. Data Exploration: Descriptive Analysis, Null Values, Correlation, Visualizations
3. Data Preparation – Feature Engineering, Data Cleaning, and Data Scaling
4. Data Modeling – KMeans, PCA, TSNE, UMAP, and DBSCAN

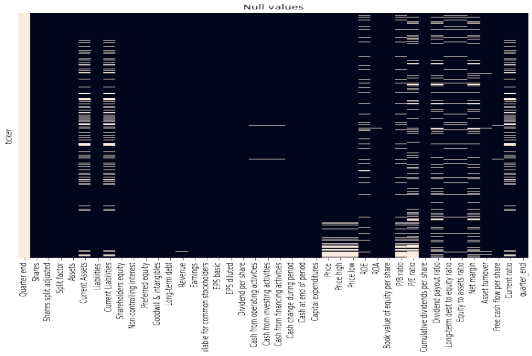
Data Setup:

For the set up I installed python packages such as (numpy, pandas, seaborn, matplotlib.pyplot, PCA, StandardScaler, TSNE, KMeans, DBSCAN, and UMAP). Additionally created a helper function fillndrop() that takes parameter of a list of columns to manipulate, method of manipulation, and the data frame.

Data Exploration:

First step was to load stock-fundamentals.csv as a pandas data frame. To create some humor for myself and assigned the variable as 'stonks' and for efficiency set the index to 'ticker'. Next called the head(), info(), and description() functions on 'stonks'. I noticed there was significant variability between the data point, so the dataset would need to be scaled. With the help of isnull() I also noticed there were also quite a few null values which needed to be address for effectiveness of the machine learning methods. Lastly, I created a few quick visualizations with heatmap and histograms.





Data Preparation:

First step was some feature engineering. I simplified 'quarter_end' from YYYY-MM-DD to just MM (month). The old format didn't provide much value. I found an article that listed important metrics for stocks. I noticed the data frame had price and sales ('Revenue') but not the price to sales ratio ('P/S ratio').

Next step was to clean the data with my created helper function fillndrop(). I dropped 'Quarter end' because all values were Null; 'Split factor' because all values were '1'; and 'quarter_end' because feature engineered made variable obsolete. I then filled the remaining NA with the mean, and although it reduces the variance it is a good alternative to removing observations or variables. Lastly, I dropped

the row with ticker 'BRK' that stands for Berkshire Hathaway Inc. Class A (I will explain why when discussing PCA modeling).

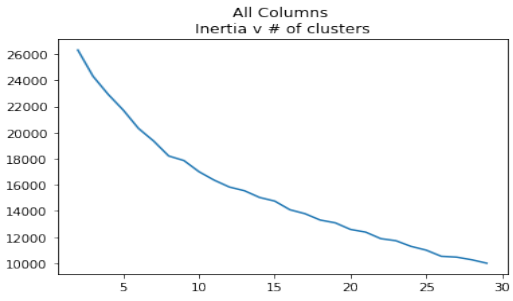
Last step was scaling the data. I decided to choose Standard Scalar because of my familiarity with the underlying process.

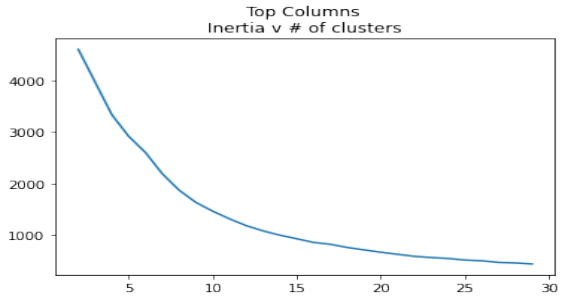
Data Modeling

For modeling and clustering I used 5 methods. First KMeans, then PCA, then TSNE, then UMAP, and finally a combination of DBSCAN and KMeans.

KMeans

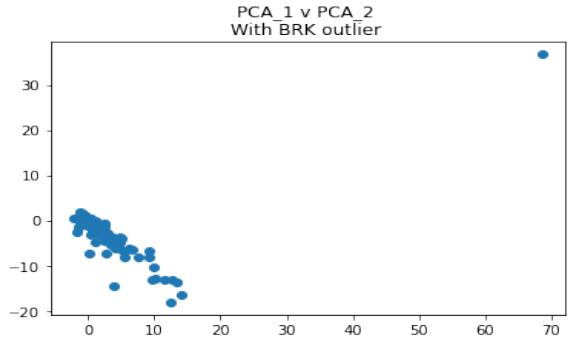
For KMeans I first tried using all the scaled numeric columns to create an inertia v cluster graph to use the elbow method to see how many clusters would be used. With all the variables it was hard to see a distinct elbow. I did however try using less variable. I referenced the articles that discussed top metric for stock and used those columns which were 'ROE', 'P/E ratio', 'P/B ratio', 'P/S ratio', 'Free cash flow per share', 'Dividend payout ratio', and 'Long-term debt to equity ratio'. The inertia graph with reduced dimension produced a more distinct elbow, which told me some dimensional reduction would be needed for effective clustering. Additional since I used more than two columns, I was not able to give a graphical representation of the clusters in a scatter plot.

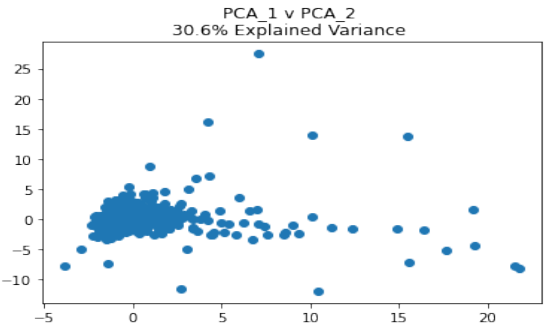




PCA

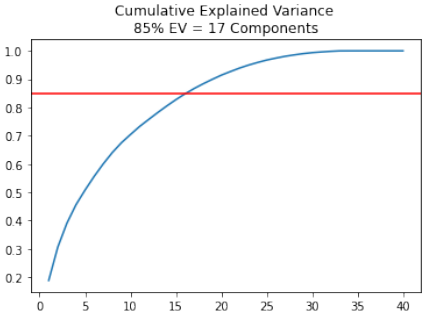
Next, I explored PCA, but decided to go back to using all the columns and letting the model do the work for dimensionality reduction. After fit_transform() to the data, I plotted PCA_1 and PCA_2 which had 38% of the explained variance. However, there was a significant outlier which is weird because I scaled my data. I found the stock BRK had a share price of 365,000. I took out the observation because Standard Scalar is quite sensitive to outliers, and this was a huge outlier. When I reran PCA_1 and PCA_2 the explained variance was 30.6% but the graph looked better. With or without BRK an explained variance of less than 40% is not sufficient given the fact there are more advanced methods in our tool kit for dimensionality reduction.



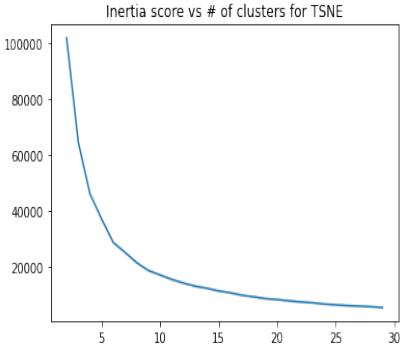


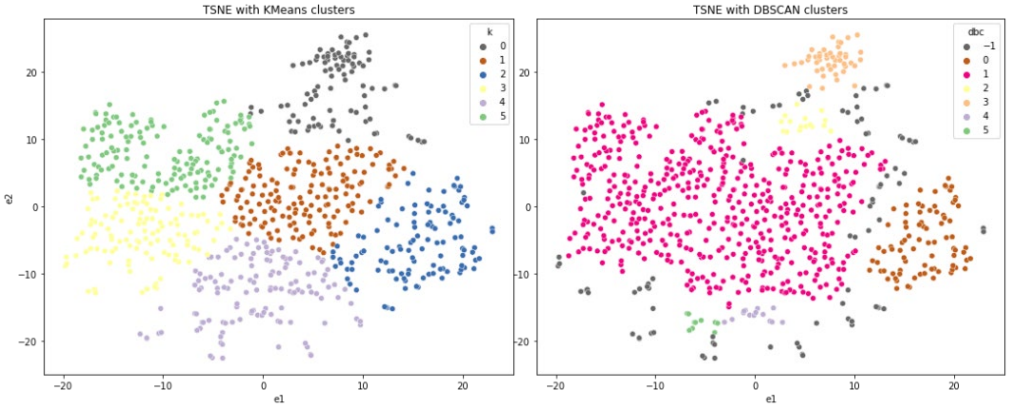
TSNE

I decide to use TSNE to create two embeddings, but first reduce the dimensionality with PCA. I create a cumulative explained variance graph to decide on how many components to pass through. I decided on a target explained variance of 85% because it used 17 components which is less than half the original dimension. Additionally, I created a heat map to glance under the hood what the weights were for the different components.



After feeding the 17 components into TSNE, I graphed a scatter plot of the two embedding and wanted to fit_predict() clusters on top. My goal was to use two different clustering methods (KMeans and DBSCAN) and get the similar results. Unfortunately, it was to no avail.





UMAP

Lastly, I decided to use UMAP because it claims it does a better job at maintaining the relationship during dimensionality reduction. Unlike TSNE, UMAP does not need to be manually feed PCA components. Same as before, I graphed the two dimensional outputs and tried to fit_predict() clusters on top. Fortunately, I was able to create similar clustering using KMeans and DBSCAN. The 5-clusters generated by DBSCAN matches nicely with the 5-cluster elbow point of inertia score for KMeans. Ultimately, UMAP was the best tool for dimension reduction. Additionally, there are 5 clusters that publicly traded stocks could be group. This clustering insight is supported by two different clustering methods (KMeans and DBSCAN) that produced very similar result.