1. Main Objective of Analysis

We analyze the “Dow Jones Index Data Set” from UCI Machine Learning Repository and perform clustering methods to determine if any structural patterns exist in the data. Perhaps any structure found could aid in identifying stock trends which could be useful information for potential investment strategies.

We perform PCA to determine if we may simplify structure of the data and see which features contribute the most to the variation in the data.

2. Description of Data

The data can be found from the UCI Machine Learning Repository (link to data).

Data consists of weekly stock data for the Dow Jones Index as reported by major stock exchanges from January 2011 to June 2011. Each row represents a week’s worth of data for a particular stock. There are 30 unique stocks with 25 weeks of data for each stock. The first few rows of the data can be seen in Figure 1.

Figure 1.

	quarter	stock	date	open	high	low	close	volume	percent_change_price	percent_change_volume_over_last_wk	previous_weeks_volume
0	1	AA	1/7/2011	$15.82	$16.72	$15.78	$16.42	239655616	3.79267	NaN	NaN
1	1	AA	1/14/2011	$16.71	$16.71	$15.64	$15.97	242963398	-4.42849	1.380223	239655616.0
2	1	AA	1/21/2011	$16.19	$16.38	$15.60	$15.79	138428495	-2.47066	-43.024959	242963398.0
3	1	AA	1/28/2011	$15.87	$16.63	$15.82	$16.13	151379173	1.63831	9.355500	138428495.0
4	1	AA	2/4/2011	$16.18	$17.39	$16.18	$17.14	154387761	5.93325	1.987452	151379173.0

There are 750 rows and 16 features. Features are explained as follows:

- ***quarter:*** the yearly quarter (1 = Jan-Mar; 2 = Apr-Jun)
- ***stock:*** stock symbol
- ***date:*** the last business day of the work week (this is typically a Friday)
- ***open:*** the price of stock at the beginning of the week
- ***high:*** the highest price of the stock during the week
- ***low:*** the lowest price of the stock during the week
- ***close:*** the price of the stock at the end of the week
- ***volume:*** the number of shares of stock that traded hands in the week
- ***percent_change_price:*** the percentage change in price throughout the week
- ***percent_change_volume_over_last_wk:*** the percentage change in the number of shares of stock that traded hands for this week compared to the previous week
- ***previous_weeks_volume:*** the number of shares of stock that traded hands in the previous week
- ***next_weeks_open:*** the opening price of the stock in the following week
- ***next_weeks_close:*** the closing price of the stock in the following week
- ***percent_change_next_weeks_price:*** the percentage change in price of the stock in the following week
- ***days_to_next_dividend:*** the number of days until the next dividend

- ***percent_return_next_dividend:*** the percentage of return on the next dividend

3. Data Exploration, Data Cleaning, Feature Engineering

We observe that some columns have 'object' data types due to the string, '$,' in the values. There also appears to be 30 missing observations for the 2 features, 'percent_change_volume_over_last_wk' and 'previous_weeks_volume.' These can be seen in Figures 2 and 3, respectively.

Figure 2.

```
raw_data.dtypes
```

```
quarter                                int64
stock                                 object
date                                  object
open                                  object
high                                  object
low                                   object
close                                 object
volume                                 int64
percent_change_price                 float64
percent_change_volume_over_last_wk   float64
previous_weeks_volume                float64
next_weeks_open                       object
next_weeks_close                      object
percent_change_next_weeks_price      float64
days_to_next_dividend                  int64
percent_return_next_dividend         float64
dtype: object
```

Figure 3.

```
raw_data.isnull().sum()
```

```
quarter                                0
stock                                  0
date                                   0
open                                   0
high                                   0
low                                    0
close                                  0
volume                                 0
percent_change_price                   0
percent_change_volume_over_last_wk    30
previous_weeks_volume                 30
next_weeks_open                        0
next_weeks_close                       0
percent_change_next_weeks_price        0
days_to_next_dividend                  0
percent_return_next_dividend           0
dtype: int64
```

We remove the string and convert appropriate columns to floats. We also remove the 30 missing observations from the data. We one hot encode the 'stock' and 'date' variables, resulting in a 720 x 68 dataframe.

We look at correlations between continuous variables shown in Figure 4 and note that all continuous features are at least mildly to very strongly correlated with each other.

Figure 4.

	Feature_1	Feature_2	Correlation	Abs_Correlation
22	low	volume	-0.524210	0.524210
30	close	volume	-0.523935	0.523935
41	volume	next_weeks_open	-0.523710	0.523710
42	volume	next_weeks_close	-0.523208	0.523208
3	open	volume	-0.522450	0.522450
...	...	...	...	...
17	high	next_weeks_open	0.999476	0.999476
21	low	close	0.999549	0.999549
12	high	close	0.999555	0.999555
0	open	high	0.999613	0.999613
34	close	next_weeks_open	0.999917	0.999917

We also note that many of the continuous variables are skewed. Figure 5 shows the skew values of these features, and Figure 6 shows the skew values after taking a cube root transformation. The transformation has reduced the skew of these columns, however, 'previous_weeks_volume' and 'volume' still have somewhat high skew. Rather than trying even more complicated transformations, we continue onward with the cube root transformation.

Figure 5.

```
previous_weeks_volume                3.256532
volume                               2.868761
percent_change_volume_over_last_wk   2.544001
low                                  1.281314
next_weeks_close                     1.280180
close                                1.275546
next_weeks_open                      1.274181
open                                 1.271666
high                                 1.271656
percent_return_next_dividend         0.398394
percent_change_next_weeks_price     -0.137200
percent_change_price                -0.414007
dtype: float64
```

Figure 6.

```
previous_weeks_volume                1.248195
volume                               1.208808
percent_return_next_dividend         0.398394
high                                 0.288950
open                                 0.285054
next_weeks_open                      0.283376
next_weeks_close                     0.282736
close                                0.282267
low                                  0.280726
percent_change_volume_over_last_wk   0.091051
percent_change_next_weeks_price     -0.137200
percent_change_price                -0.414007
dtype: float64
```

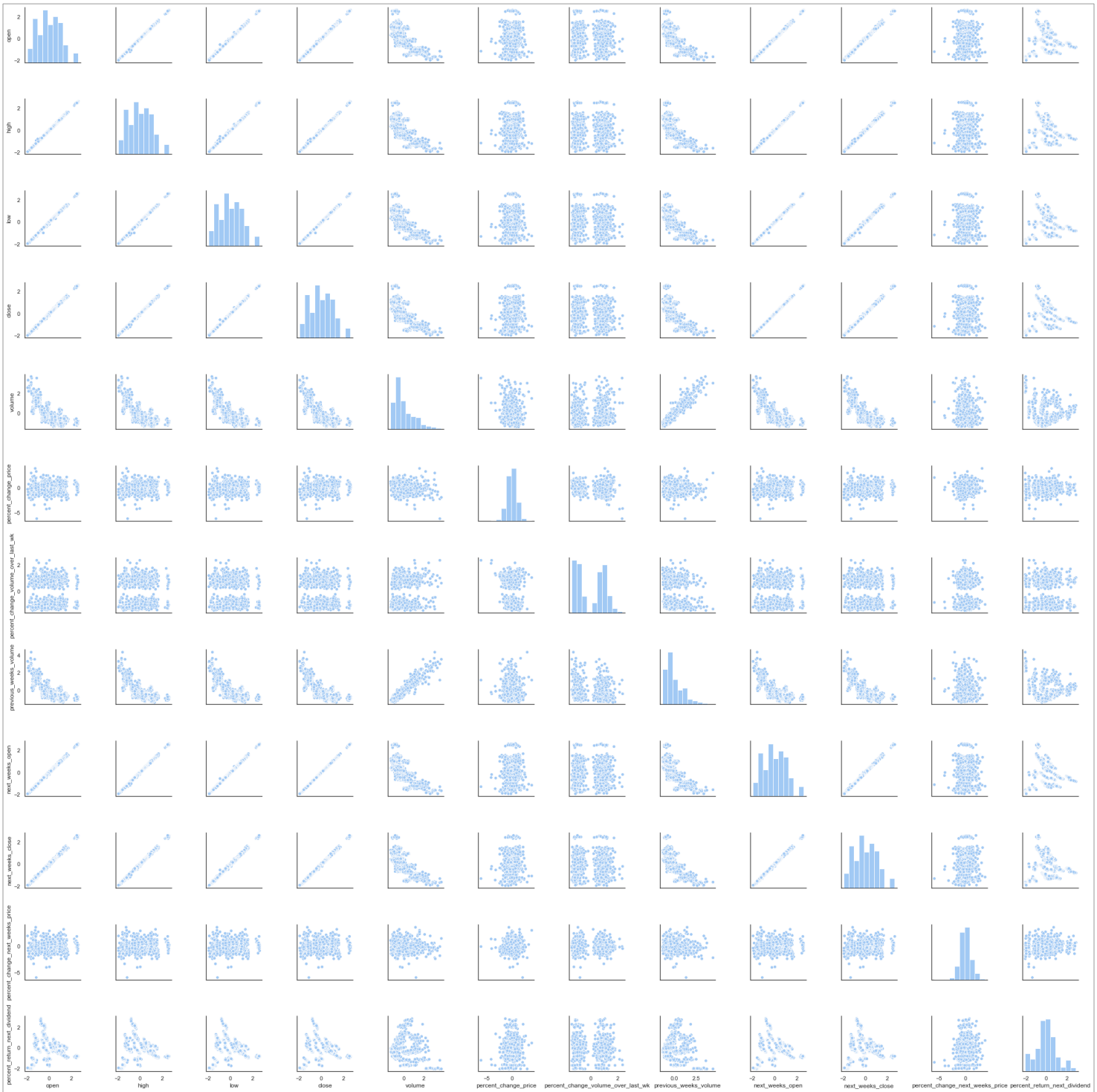
We then perform standard scaling on float columns. Figure 7 shows summary statistics after using standard scaling.

Figure 7.

	count	mean	std	min	25%	50%	75%	max
open	720.0	-0.0	1.0	-1.94	-0.71	-0.06	0.74	2.63
high	720.0	0.0	1.0	-1.93	-0.70	-0.06	0.75	2.60
low	720.0	-0.0	1.0	-1.93	-0.74	-0.06	0.73	2.60
close	720.0	0.0	1.0	-1.94	-0.68	-0.06	0.74	2.59
volume	720.0	0.0	1.0	-1.36	-0.73	-0.34	0.50	3.72
percent_change_price	720.0	-0.0	1.0	-6.17	-0.53	-0.01	0.64	3.94
percent_change_volume_over_last_wk	720.0	-0.0	1.0	-1.42	-0.99	0.23	0.92	2.35
previous_weeks_volume	720.0	-0.0	1.0	-1.35	-0.73	-0.34	0.51	4.39
next_weeks_open	720.0	0.0	1.0	-1.94	-0.69	-0.06	0.74	2.62
next_weeks_close	720.0	0.0	1.0	-1.94	-0.68	-0.07	0.74	2.63
percent_change_next_weeks_price	720.0	-0.0	1.0	-5.87	-0.54	-0.06	0.60	3.64
percent_return_next_dividend	720.0	0.0	1.0	-2.05	-0.52	-0.04	0.53	2.86

Figure 8 shows pair plots and histograms of continuous variables. We can see the high correlations between features noted earlier. It also appears that, at least in two dimensions, groups form when 'percent_change_volume_over_last_wk' is plotted against another feature. There appears to be two groups, one with positive and one with negative percent changes in volume of stocks traded.

Figure 8.


open
high
low
close
volume
percent_change_price
percent_change_volume_over_last_wk
previous_weeks_volume
next_weeks_open
next_weeks_close
percent_change_next_weeks_price
percent_return_next_dividend

4. Modeling

We try K-means, hierarchical agglomerative clustering, DBSCAN, and mean shift. Because of the high correlation among features, we also perform PCA.

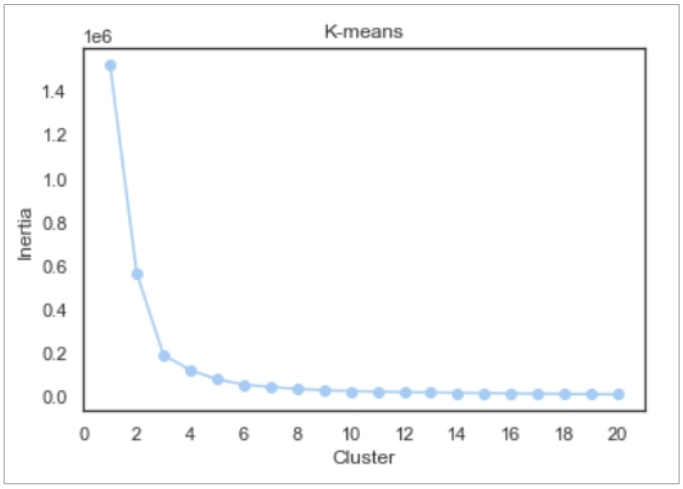
4.1 K-means

We fit different K-means models with clusters ranging from 1 to 20 and examine the resulting inertia values. Figure 9 shows the resulting inertia values for each number of clusters, and Figure 10 shows a plot of the inertia values against the number of clusters. There appears to be an elbow at around 3 clusters, and inertia reduces at a slower pace as the number of clusters increases.

Figure 9.

	clusters	inertia	model
0	1	1.52381e+06	KMeans(algorithm='auto', copy_x=True, init='k-...
1	2	567226	KMeans(algorithm='auto', copy_x=True, init='k-...
2	3	193474	KMeans(algorithm='auto', copy_x=True, init='k-...
3	4	123970	KMeans(algorithm='auto', copy_x=True, init='k-...
4	5	81707	KMeans(algorithm='auto', copy_x=True, init='k-...
5	6	57644.9	KMeans(algorithm='auto', copy_x=True, init='k-...
6	7	46313.8	KMeans(algorithm='auto', copy_x=True, init='k-...
7	8	38279.7	KMeans(algorithm='auto', copy_x=True, init='k-...
8	9	31641.9	KMeans(algorithm='auto', copy_x=True, init='k-...
9	10	27566.8	KMeans(algorithm='auto', copy_x=True, init='k-...
10	11	25374.4	KMeans(algorithm='auto', copy_x=True, init='k-...
11	12	22738.3	KMeans(algorithm='auto', copy_x=True, init='k-...
12	13	21152.1	KMeans(algorithm='auto', copy_x=True, init='k-...
13	14	19790.7	KMeans(algorithm='auto', copy_x=True, init='k-...
14	15	18624.3	KMeans(algorithm='auto', copy_x=True, init='k-...
15	16	17297.8	KMeans(algorithm='auto', copy_x=True, init='k-...
16	17	15926.6	KMeans(algorithm='auto', copy_x=True, init='k-...
17	18	14916.8	KMeans(algorithm='auto', copy_x=True, init='k-...
18	19	13823.3	KMeans(algorithm='auto', copy_x=True, init='k-...
19	20	12973.1	KMeans(algorithm='auto', copy_x=True, init='k-...

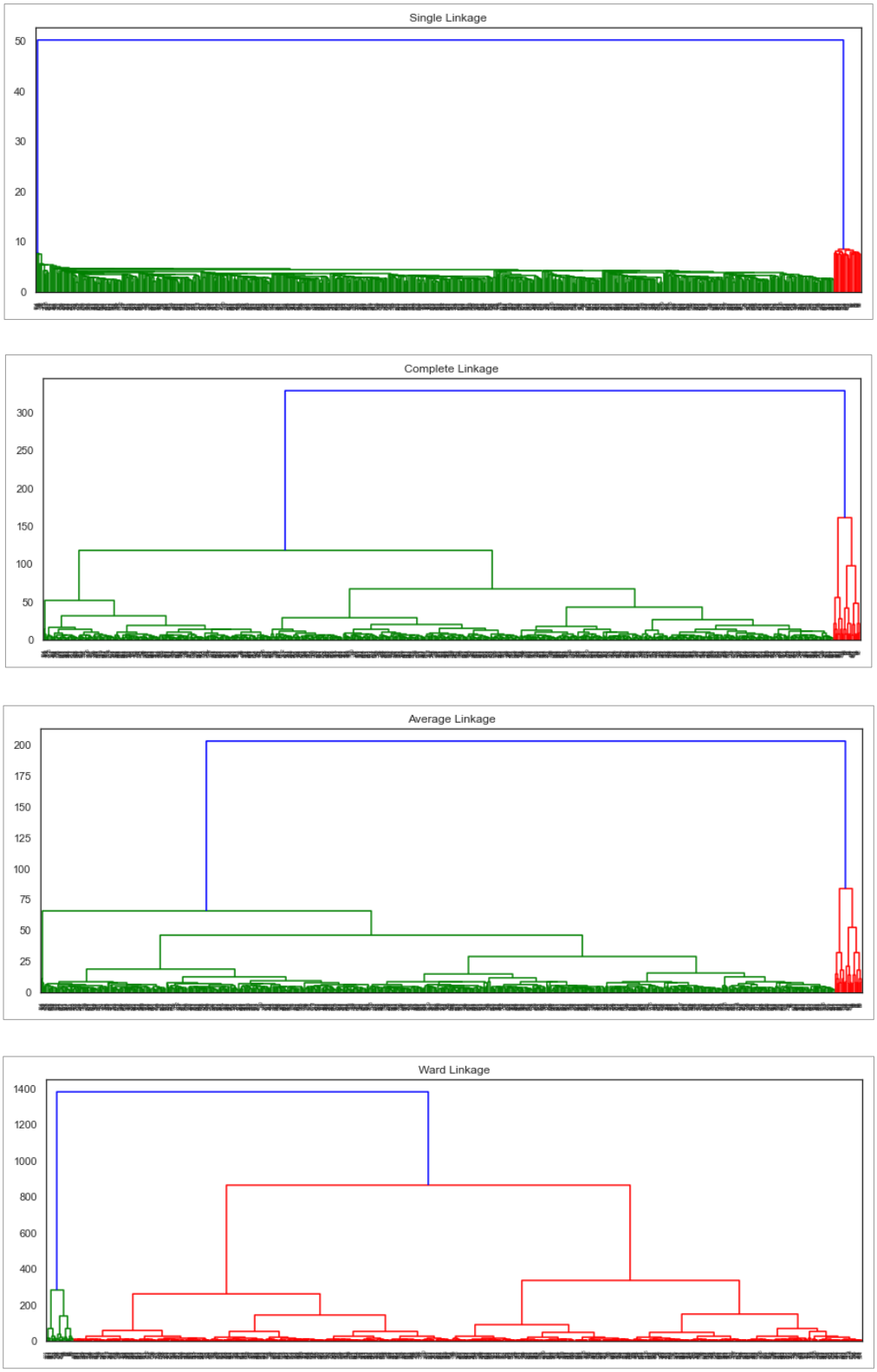
Figure 10.



4.2 Hierarchical Agglomerative Clustering

We perform hierarchical clustering using “single,” “complete,” “average,” and “ward” linkage and examine resulting dendrograms. These can be seen in Figure 11.

Figure 11.



It would seem that based on the dendrograms, a logical cut off point for all hierarchical models would be 2 groups.

4.3 DBSCAN

For DBSCAN we try epsilon values ranging from 1 to 3 in increments of 0.25 and density thresholds of 3, 4, 5, 6, and 7. Figure 12 shows the resulting number of clusters for each combination of epsilon and density thresholds. It appears that as the radius of the local neighborhood increases, more clusters are able to be found, and the lower the n_clu, the more clusters are able to be found for the increasing epsilons. This would perhaps suggest that beyond the 2.00 epsilon, the data is randomly dispersed, but within the 2.00 epsilon, there appears to be one grouping. Figure 13 shows the average density threshold and average number of clusters for each epsilon. Again, it is observed that as epsilon increases past 2.00, more clusters begin to form.

Figure 12.

	epsilon	n_clu	clusters
0	1.00	3.0	1.0
1	1.00	4.0	1.0
2	1.00	5.0	1.0
3	1.00	6.0	1.0
4	1.00	7.0	1.0
5	1.25	3.0	1.0
6	1.25	4.0	1.0
7	1.25	5.0	1.0
8	1.25	6.0	1.0
9	1.25	7.0	1.0
10	1.50	3.0	1.0
11	1.50	4.0	1.0
12	1.50	5.0	1.0
13	1.50	6.0	1.0
14	1.50	7.0	1.0
15	1.75	3.0	1.0
16	1.75	4.0	1.0
17	1.75	5.0	1.0
18	1.75	6.0	1.0
19	1.75	7.0	1.0
20	2.00	3.0	8.0
21	2.00	4.0	1.0
22	2.00	5.0	1.0
23	2.00	6.0	1.0
24	2.00	7.0	1.0
25	2.25	3.0	25.0
26	2.25	4.0	3.0
27	2.25	5.0	1.0
28	2.25	6.0	1.0
29	2.25	7.0	1.0
30	2.50	3.0	52.0
31	2.50	4.0	15.0
32	2.50	5.0	8.0
33	2.50	6.0	2.0
34	2.50	7.0	1.0
35	2.75	3.0	58.0
36	2.75	4.0	30.0
37	2.75	5.0	19.0
38	2.75	6.0	14.0
39	2.75	7.0	9.0
40	3.00	3.0	42.0
41	3.00	4.0	30.0
42	3.0	5.0	20.0
43	3.0	6.0	17.0
44	3.0	7.0	14.0

Figure 13.

```
dbscan_results.groupby(['epsilon']).mean()
```

epsilon	n_clu	clusters
1.00	5.0	1.0
1.25	5.0	1.0
1.50	5.0	1.0
1.75	5.0	1.0
2.00	5.0	2.4
2.25	5.0	6.2
2.50	5.0	15.6
2.75	5.0	26.0
3.00	5.0	24.6

4.4 Mean Shift

We try mean shift with various bandwidth levels. Figure 14 shows the results. For small bandwidths, multiple clusters are found. For medium sized bandwidths, 2 to 4 clusters are found. For large bandwidths, only 1 cluster is found. It appears a bandwidth of 150 is enough to be able to cluster the entire data into 1 cluster.

Figure 14.

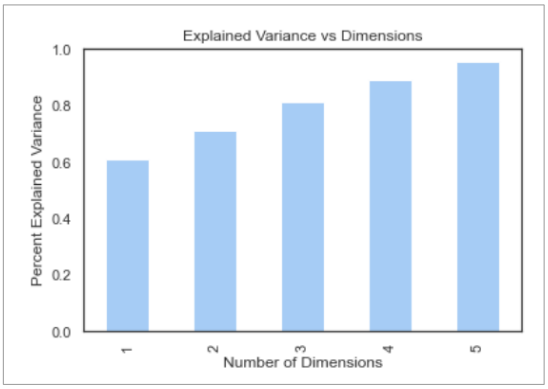
	bandwidth	clusters
0	5	44
1	10	17
2	20	9
3	35	4
4	50	3
5	75	2
6	100	2
7	150	1
8	200	1
9	300	1
10	400	1
11	500	1

4.5 PCA

We try PCA on continuous variables. When looking at the proportion of explained variance seen in Figure 15, it seems that at least 5 principal components are needed to explain most of the variance in the data. This perhaps suggests we may reduce the original 12 continuous features to 5 linear combinations of the original 12 features.

Figure 15.

# of components	model	var
1	PCA(copy=True, iterated_power='auto', n_compon...	0.607851
2	PCA(copy=True, iterated_power='auto', n_compon...	0.711313
3	PCA(copy=True, iterated_power='auto', n_compon...	0.811354
4	PCA(copy=True, iterated_power='auto', n_compon...	0.889659
5	PCA(copy=True, iterated_power='auto', n_compon...	0.954909

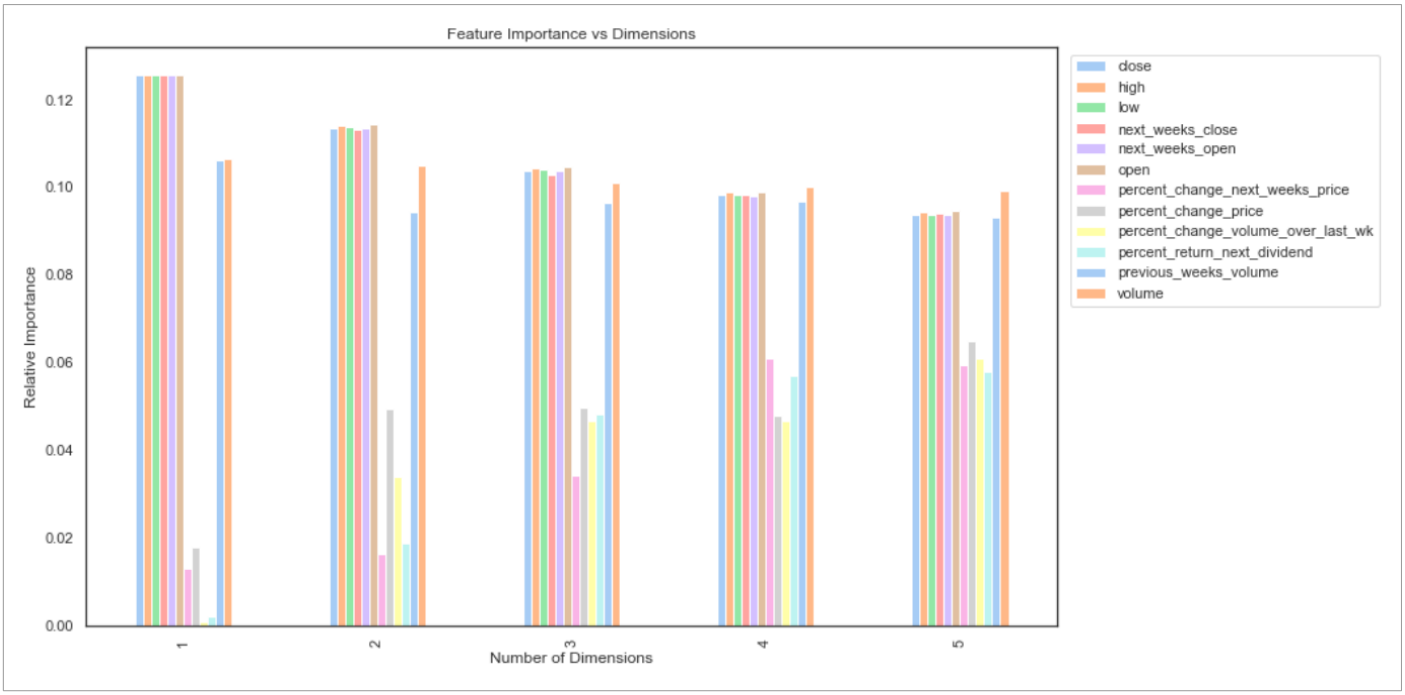


When examining the feature importances, ‘close,’ ‘high,’ ‘low,’ ‘next_weeks_close,’ ‘next_weeks_open,’ ‘open,’ previous_weeks_volume,’ and ‘volume,’ all seem to explain the most variance of each principal component. Each of these features explains about the same amount, 9-12% of the variance, for each component. We notice that the contributions of these variables’ feature importances decrease slightly as we add more principal components. ‘Percent_change_next_weeks_price,’ ‘percent_change_price,’ percent_change_volume_over_last_wk,’ ‘percent_return_next_dividend,’ appear to contribute little to each principal component, although as the number of principal components increases, their feature importances appear to also increase. These results can be collectively seen in Figures 16 and 17.

Figure 16.

# of components features	1	2	3	4	5
close	0.125629	0.113342	0.103816	0.098155	0.093777
high	0.125493	0.114161	0.104390	0.098770	0.094241
low	0.125572	0.113653	0.103964	0.098239	0.093894
next_weeks_close	0.125636	0.113161	0.102846	0.098444	0.094031
next_weeks_open	0.125622	0.113352	0.103821	0.098139	0.093790
open	0.125461	0.114524	0.104611	0.098828	0.094791
percent_change_next_weeks_price	0.012943	0.016263	0.034334	0.060927	0.059334
percent_change_price	0.017849	0.049371	0.049780	0.047874	0.064926
percent_change_volume_over_last_wk	0.000926	0.033907	0.046868	0.046812	0.060992
percent_return_next_dividend	0.002257	0.018740	0.048153	0.056925	0.058067
previous_weeks_volume	0.106135	0.094462	0.096451	0.096715	0.093087
volume	0.106477	0.105065	0.100965	0.100171	0.099068

Figure 17.



5. Recommendation for Final Model/Summary of Key Findings and Insights

Rather than suggesting 1 single model, I believe we should assess the aggregate of the models presented together. Each model appears to suggest that the data can be clustered into 2 to 3 groups. One group appears to be a rather homogeneous density of points. The other group seems to be one with much variation within. PCA also seems to suggest that the original 12 continuous features can be reduced to 5 features.

6. Suggestions for Next Steps

While it appears that this data can be clustered into 2 or so groups, it would be more useful for financial and economic purposes if we could find out which stocks and times each group is comprised of. We could possibly try to look at the dendrograms more closely to achieve this. It may be of interest to try kernel PCA in addition to trying to determine somehow if the variation we are seeing captured in the first 5 principal components of the linear PCA is only due to the proposed cluster which seems to have a lot of variation. Finally, we could maybe redo the analysis by re-scaling the float columns before transforming skewed columns.