Task 1:

Combining the data:

One approach to combining the financial data of 10 companies while preserving the granularity of each record would be to append an ordinal attribute to each record in the data set. This attribute *company* will have a value $\in \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$. The value of the *company* attribute x will categorize the record into company x. An ordinal attribute was chosen because the 10 companies are given an abstract ordering in this approach. Therefore, we can combine the data into one set while retaining the granularity of each company and its financial records.

Possible Findings:

The companies I selected maintain strong positions in the video game industry. For each company excluding Microsoft, their primary focus is the video game market. Nintendo's quarterly earnings showed a significant increase in revenue during the first months of the pandemic lockdown in the US. Reports claimed the video game market increased in size because of students and parents being forced into remote work and remote learning modalities. For example, business insider reported in August 2020 Nintendo had surged by 428% since the pandemic. I would like to explore the increase in market capitalization coinciding with pandemic lockdowns in the US from March 2020 to March 2021. I expect to find a comparable surge in market capitalization across all 10 companies in the given time frame. The US was selected for this exploration because all 10 companies have a sizable foothold in the American video game market. The timeline of the lockdowns are abstractions since there is no specific date when all American states rolled out and lifted the lockdowns.

I would also like to discover which companies saw the greatest percentage increase in market capitalization during the US lockdowns compared to pre-pandemic market capitalization. This can provide insight into which companies in the American video game market were best positioned to benefit from the pandemic. Exploring this question would involve calculating pre-pandemic market capitalization averages and comparing it to the peak market capitalization from March 2020 – 2021.

Classification Target:

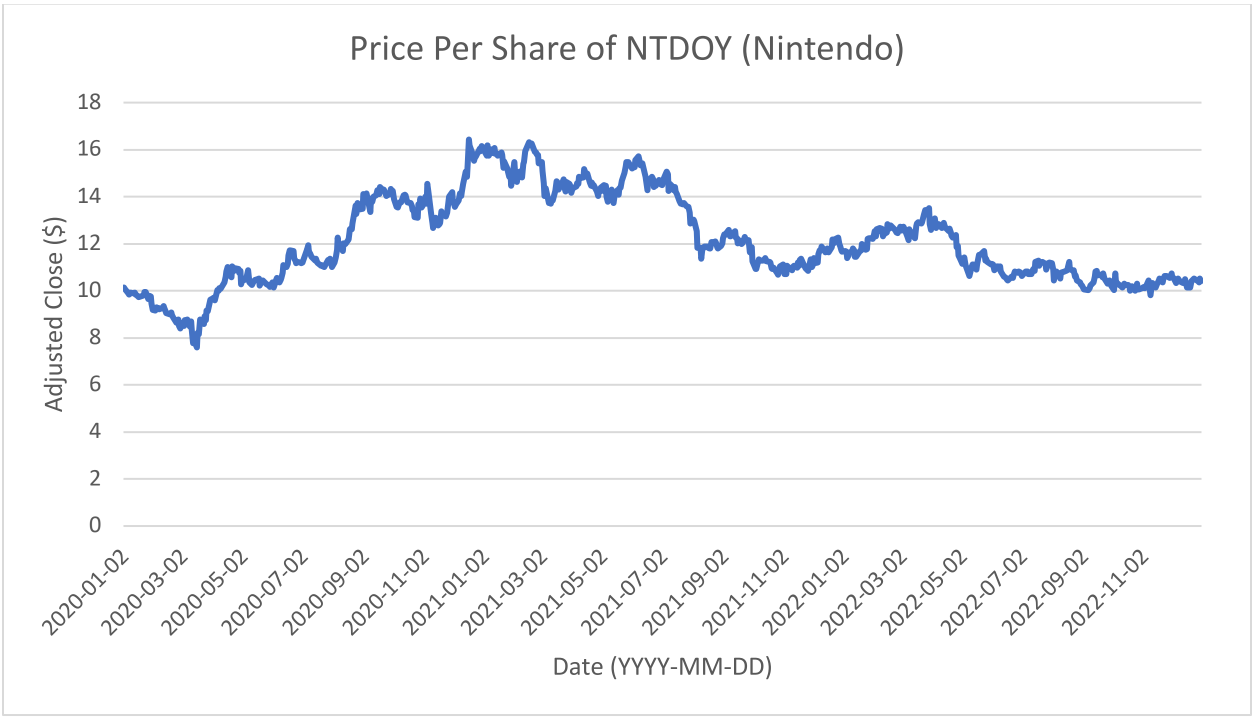
Given a set of records, each containing information on the volume and price per share of a stock on a trade day from January 2020 – 2023, determine whether the price per share (*Adj Close*) will increase or decrease in the next trade day. The implementation of the classifier will be discussed in Task 6. We are predicting the direction of the stock; therefore the class attribute will be *Direction*.

Task 2:

The dimensions I have selected from the data are Adjusted Close and Volume.

Dimension 1:

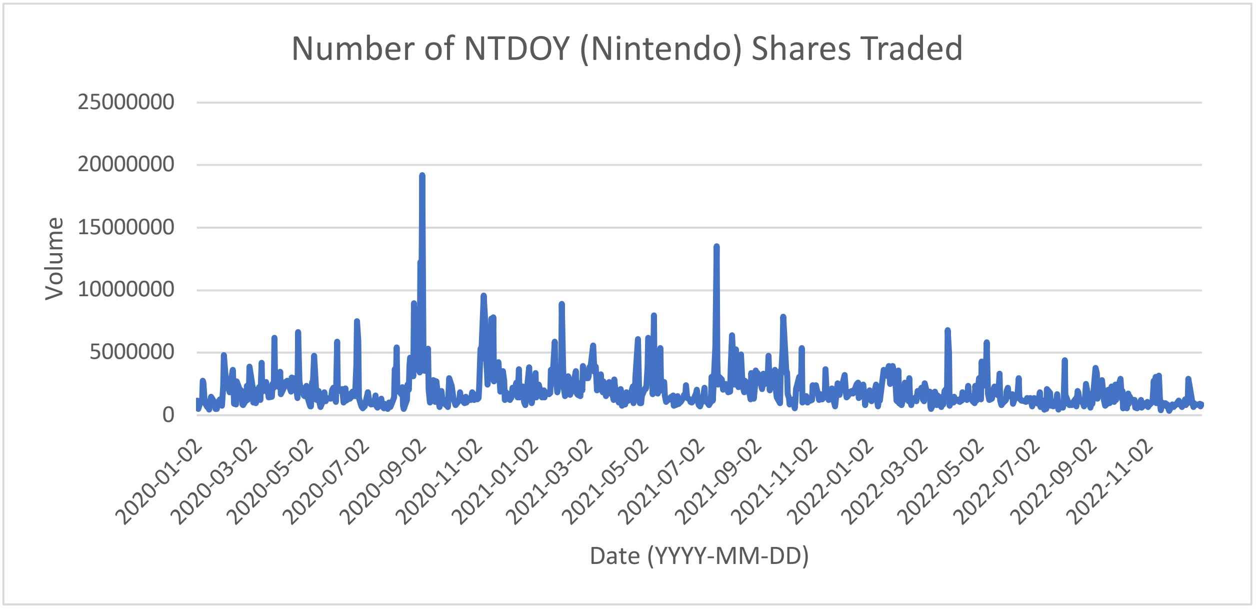
Adjusted close refers to the closing price of one share of stock adjusted according to financial decisions of the company. Closing price refers to the price of one share at the end of a trading day. At the end of a trading day, the company may perform a stock split, dividend, or other action which affects the price per share. Therefore, Adjusted Close is an accurate measurement of price per share at the end of a trading day.



Insight we may get from the price per share over a time interval is the fluctuation in value of the company over that time interval. As the valuation of a company increases, the price per share increases. Surges in the price per share such as in January 2021 represent an increase in the valuation of the company (Nintendo). Increased valuation may be the result of positive performance, news, or favorable industry conditions for the company. The 428% surge in Nintendo's revenue from pre-pandemic levels, and the promising Q4 quarterly earnings report are probable causes for the surge in price per share in January 2021. Therefore, insight we can gain from price per share is an increase or decrease in the valuation of a company.

Dimension 2:

Volume is the number of shares bought and sold in a trade day. Volume indicates the demand and liquidity of a stock. High volume signifies a high number of shares purchased and thus a high demand. Additionally, high volume indicates a high number of investors purchasing the stock resulting in a high liquidity.



Insight we may derive from the number of shares traded per day over an interval of time is the fluctuation in demand for the shares over that interval of time. An increase in the demand of a stock indicates that investors maintain a promising forecast for the company, or believe the stock is currently undervalued based on the company's recent financial performance and projections. In the case above, demand surges in September 2020 and August 2021. These surges coincide with positive news of the company's performance. Particularly September 2020 (which is the larger surge), occurs after the Business Insider report discussed earlier. Therefore, insight we can gain from the number of shares traded per day is an increase or decrease in the demand of the share.

Task 3:

The dimensions I have chosen from the data are Adjusted Close and Date.

Preprocessing Technique 1:

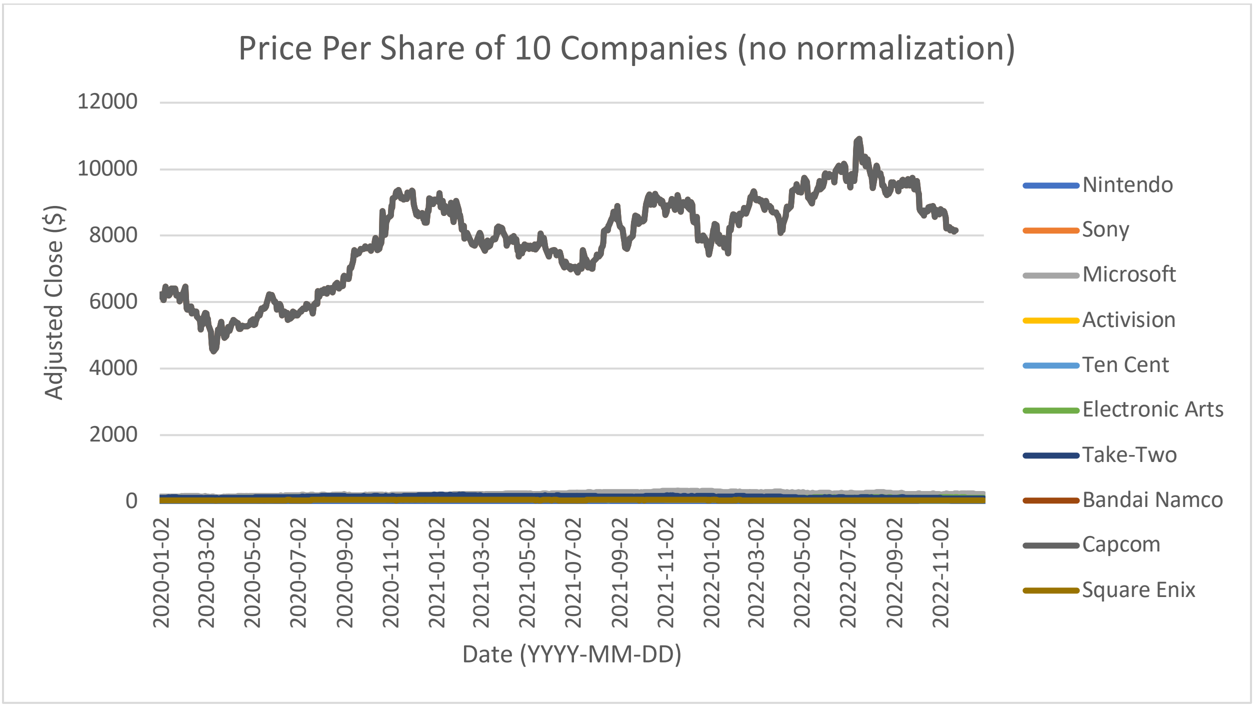
Adjusted Close is the price per share for 10 companies from January 2020 – 2023. Adjusted Close requires the normalization pre-processing technique because the price per share across the selected companies varies greatly. For instance, the ATVI (Activision) price per share ranges from 51 to 102 dollars from January 2020 – 2023. Comparatively, the MSFT (Microsoft) price per share ranges from 208 to 338 dollars in the same time interval. Normalizing the price per share across all companies places the measurements on the same scale for analysis. Normalized adjusted close is calculated by the following formula:

$$x_{norm} = \frac{(x - x_{avg})}{x_{sd}}$$

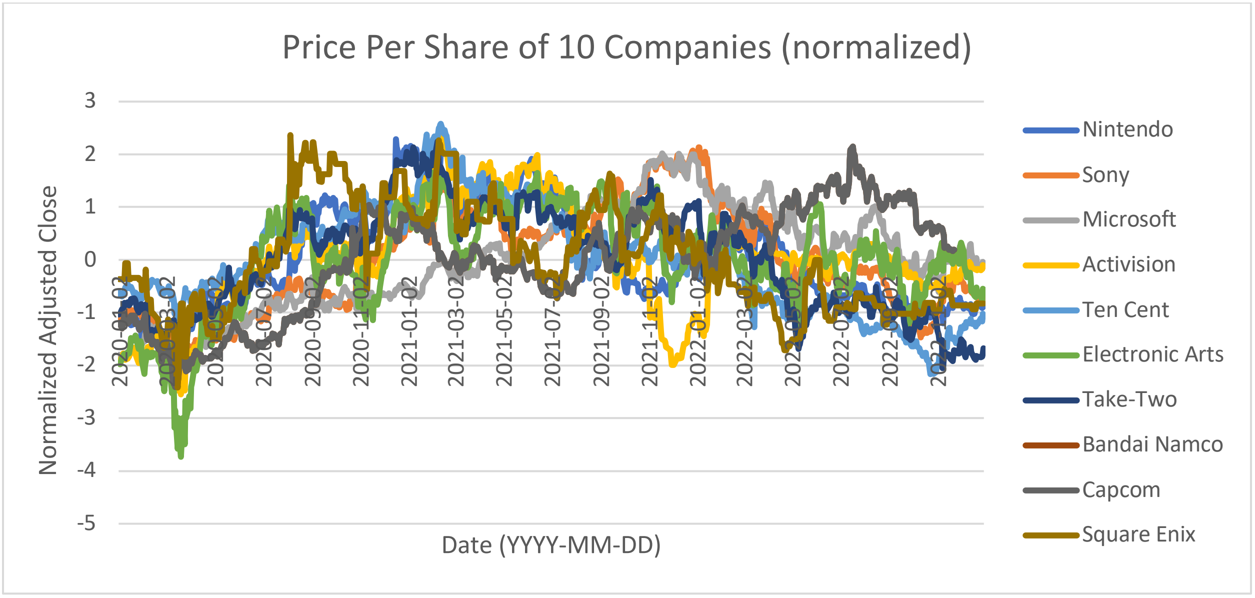
Whereby x is the adjusted close of the record, x_{norm} is the normalized adjusted close, x_{avg} is the average adjusted close across all records for the company, and x_{sd} is the standard deviation of adjusted close. Normalization was achieved in Excel through the formula, where B is the *Adj Close*, and B_{norm} is the normalized *Adj Close*

$$B_{norm} = (\$B2 - \text{AVERAGE}(\$B\$2{:}\,\$B\$757))/\text{STDEV}(\$B\$2{:}\,\$B\$757)$$

Before Normalization:



After Normalization:

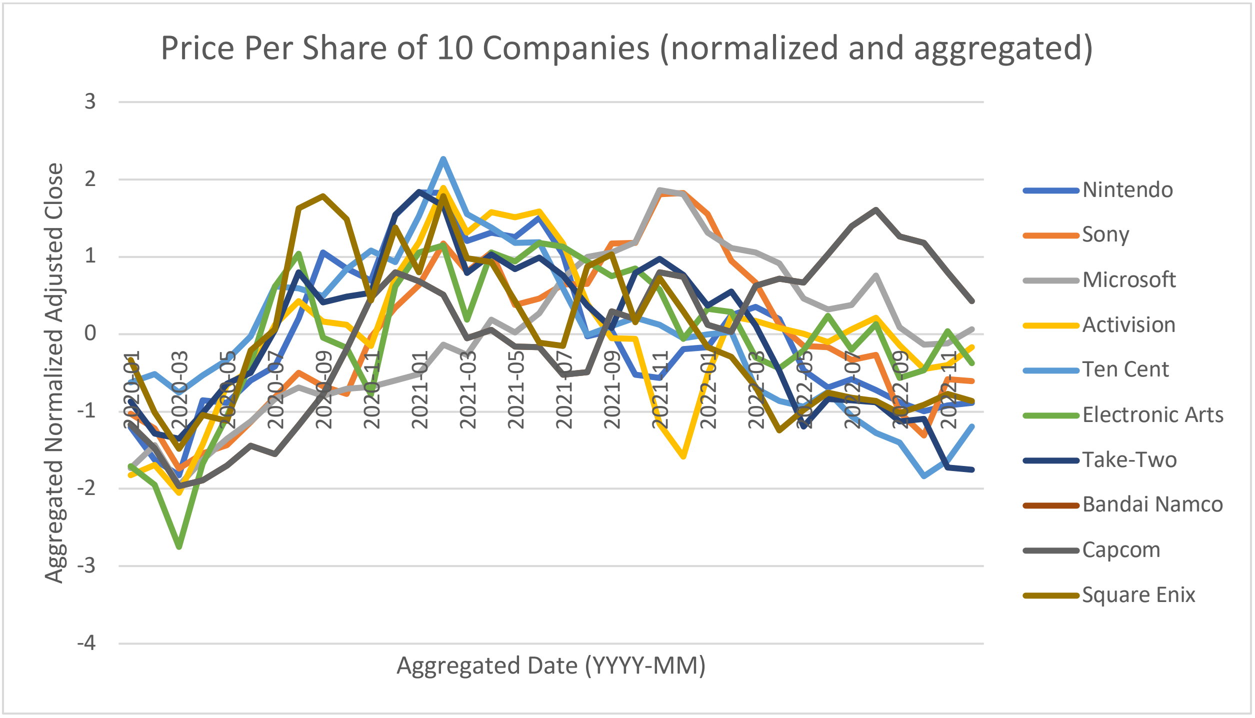


From the normalization pre-processing technique, an insight we can derive is how the companies within the video game industry *move* with respect to each other. Aligning the valuation of these companies to one scale allows us to identify patterns amongst the companies. When companies increase or decrease in valuation at similar points in time, the cause is a shift in the video game industry rather than each individual company acting in unison. For instance, in February 2021 the price per share for Square Enix, Electronic Arts, Ten Cent, Take-Two, Sony, Nintendo, and Activision peaked (or soared significantly in the case of Square Enix) to a three-year high from January 2020 – 2023. This pattern is not the result of individual actions from the companies, but a growth within the industry that manifests itself in an increased valuation of the most competitive companies. Given the context of the global pandemic, this growth is likely the result of an influx of new and existing customers purchasing the products or services from the companies. Therefore, normalizing adjusted close measurements across the 10 companies can provide insight into trends within the video game industry.

Preprocessing Technique 2:

Date requires the aggregation pre-processing technique. Each company contains records across 36 months. Each month contains several records for the price per share. Records for individual days in the month can be aggregated to one record for the entire month. Averaging the normalized Adjusted Close in a month provides a close estimate of the normalized Adjusted Close over that entire month. As a result, each company contains 36 records for the price per share instead of over 700. The reduction in data is significant, yet enough detail is retained to provide the same insights. The graph below displays the aggregation and normalization pre-processing techniques applied to the *Adj Close* and *Date* dimensions. Compared to the previous graph above which is normalized but not aggregated, the important details are retained. The peak in price per share for Square Enix, Electronic Arts, Ten Cent, Take-Two, Sony, Nintendo, and Activision are present in the same timeframe of February 2021. Lowest price per share for Activision (yellow) and Electronic Arts (green) also occur in the same timeframe of March 2020. Therefore, the general trends in price per share across 10 companies is retained after normalization and aggregation. Aggregation in Excel was achieved through Pivot Tables, which allowed us to group records by YYYY-MM and average the values of all grouped records.

After Aggregation:



Reducing the scale of data through aggregation allows us to derive similar insights from a smaller set of data. Therefore, the insights we expect from aggregation are identical to the insights we could derive before aggregation. Insights before aggregation were trends within the video game industry reflected through the share prices of 10 predominant companies. As discussed above, the trends prior to pre-processing are retained after normalization and aggregation. This result was expected, given each company had 36 data points. Across 10 companies, 360 data points were enough to map the general trends in the industry across 3 years.

<u>Task 4:</u>

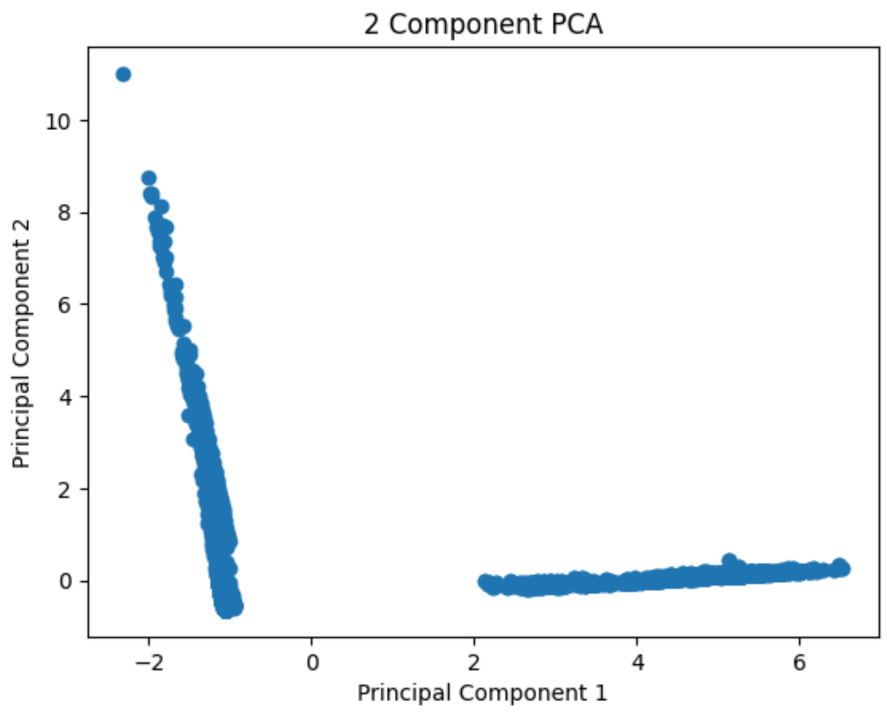
Implementation & Selecting PCs:

Principal Component Analysis was performed in a Google Collaboratory Python Notebook. The Task 0 data set was uploaded to the Collaboratory Notebook for PCA. The data set was read into the program through Python's pandas library. The features *Open*, *High*, *Low*, *Close*, *Adj Close*, and *Volume* were standardized using Python's sklearn library. Specifically, the StandardScaler class of the library which subtracts the mean from the feature and divides by the standard deviation (similar to the *Adj Close* in task 3). With the features standardized to one scale, data reduction is applied to the features through the PCA class from the sklearn library. An instance of the PCA class is initialized. The method $fit\_transform()$ is invoked, and the standardized features are passed in as parameters. The method returns a multi-dimensional array of n principal components given n features.

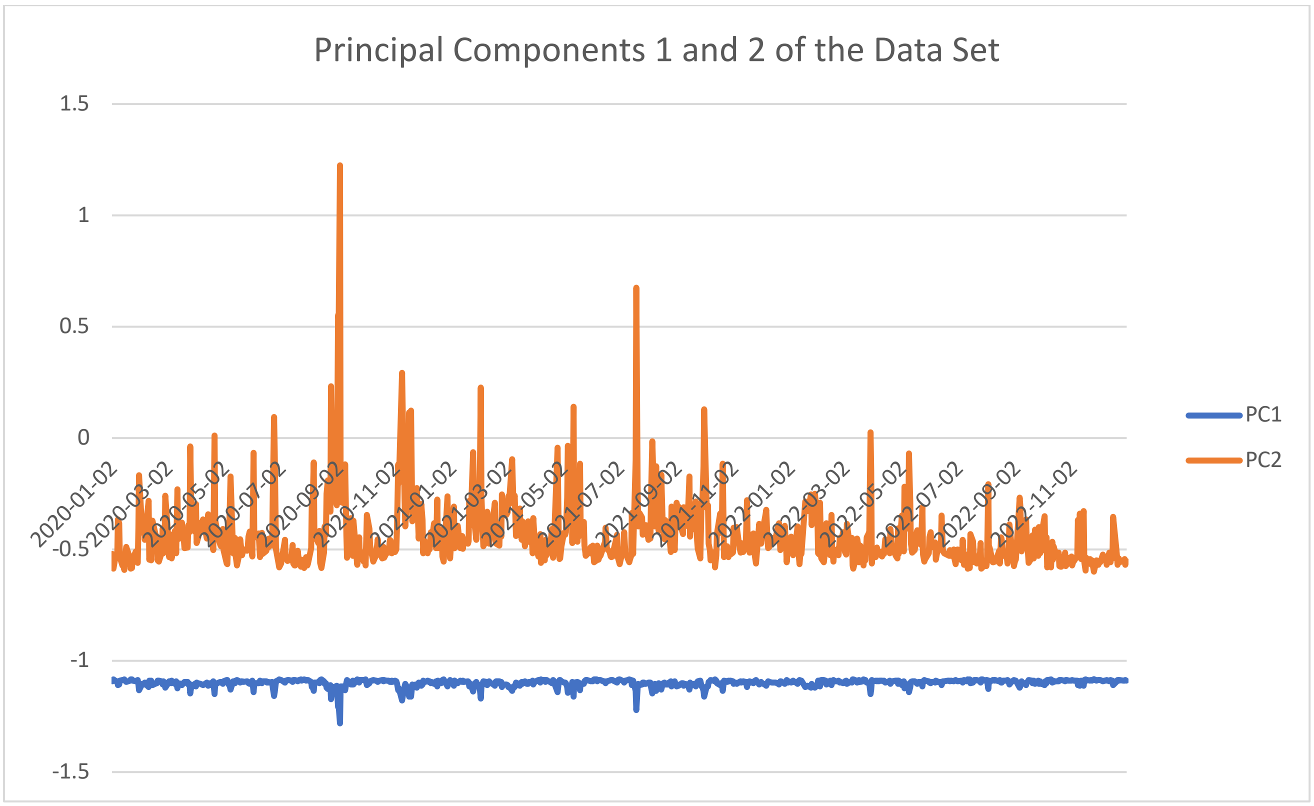
Now that we have computed the principal components, we need to select the best two. To understand what the best principal component is, we need to understand the intentions of PCA. Each principal component in the analysis captures a variance of features in the data set. The most useful principal component is the one that captures the highest variance of features. Given that we can select two principal components, we should find two components with the largest combined variance. The PCA class contains a list attribute which holds the variance for each principal component, called the $explained\_variance\_ratio$. Through list indexing, we access the variance of each principal component to determine which two PCs capture the highest variance. The first and second principal components capture 84.14% and 15.84% of the variance in the entire data set respectively. The variance captured by PC1 and PC2 is a combined 99.98% of the dataset. The remaining four principal components capture less than 0.2% of variance. Therefore, the best principal components are PC1 and PC2.

Now that we have selected our principal components, we need to create a new data frame and import the PC1 and PC2 columns. Once again, the pandas library provides support for data frames in Python. Through list indexing, we import the first two columns from the multi-dimensional array containing n principal components. We now have a table containing the two best principal components, and their values for every instance in the data set. To create a visualization of the principal components, we can use a 2-component scatter plot graph. This requires Python's matplotlib library which provides methods for creating a scatterplot from a list of data points. The methods are $scatter()$ which takes a list of data points as a parameter and initializes a scatter plot graph. Additional methods such as $xlabel()$, $ylabel()$, and $title()$ modify the appearance of the scatter plot. The scatter plot is shown below.

Visualization:



Another visualization technique involves plotting the data frame containing PC1 and PC2 onto a similar graph from previous examples. The result is shown below.



This visualization emphasizes the differences between the first and second principal component. The first principal component captures a high level of variance (> 84%) by representing data points that are furthest from the mean of the feature. Standardizing and plotting the features *Open*, *High*, *Low*, *Adj Close*, and *Volume* will reveal outliers from the mean that are close to or intersect with the PC1 line (blue) in the graph above. The second principal component captures the general patterns in the data that were not captured by the first principal component. The principal components capture over 99% of the variance in the original data set. The surge in Nintendo stocks in September 2020 and the increase in share prices among several companies in February 2021 are captured by the two components.

Preprocessing:

The data pre-processing technique required for PCA is normalization (implementation discussed earlier). Aligning the features *Open*, *High*, *Low*, *Adj Close*, and *Volume* to the same scale allows for principal components to capture the variance, thereby reducing the amount of data.

Task 5:

Dimensions for Classification:

We are predicting the future movements of the price per share given a record from one of the 10 companies. To achieve this, the following dimensions are required: *Open*, *Adj Close*, *Company*, and *Volume*. The difference of *Adj Close* and *Open* indicates whether the price per share increased or decreased. The *Volume* of a stock correlates to its demand, which may also be an indicator of the price per share in upcoming trade days. The *Company* of a record determines the context in which the classifier evaluates the class attribute of the record. This is necessary because we want to avoid the classifier evaluating the price per shares of Nintendo with respect to Sony or Bandai Namco.

Meaning of Similarity:

The concept of similarity for this problem involves the following. Given a record, how similar is it to records in our training which saw an increase in price per share on the following trade day. How similar is the record to records in our training set that showed a decrease in price per share the following day.

Task 6:

Implementation:

Given the task 0 data set, drop the columns *Date*, *High*, *Low*, and *Close*. Append the class attribute *Direction* to the data set. Now the data set has features *Open*, *Adj Close*, *Company*, *Volume*, and class *Direction*. Classify every record before parsing into training and test sets. The *Direction* of a record refers to the future movement of the price per share. The value of *Direction* is 1 when the price per share increases in the following trade day. Conversely, the value is 0 when the price per share decreases in the following trade day. Since there are an odd number of records for each company, we will fill the last record for each company with *Direction* value of the previous record. Here, we are assuming that the current direction of the price per share will continue to increase or decrease. This is a reasonable assumption because the number of changes in *Direction* is insignificant compared to the number of records. In other words, the price per share does not change often.

Now that every record is classified, we can split the data set into features and the class attribute through list indexing. We will use stratified sampling to avoid overfitting while extracting the training and test sets. Through Python's sklearn library, the $train\_test\_split()$ method provides the functionality for splitting and sampling features and the class attribute. Specify in the parameter of the method $stratify = y$, which ensures the *Direction* values are proportional in the training and test set. In other words, the percentage of records in the training set with *Direction* of 1 are equal to the percentage in the test set. The $DecisionTreeClassifier$ class in the sklearn library provides the functionality to train and test our model. Create an instance of the $DecisionTreeClassifier$ class. On this instance, call the method $fit()$ and pass the training set features and class attribute as parameters. This will train a model on the training set. On this instance, call the method $predict()$ and pass the test set features and class attribute as parameters. This will test the model on the test set and return the list of predicted class values. To determine the accuracy of the model, the method $accuracy\_score()$ from the sklearn library provides the functionality. Invoke this method and pass in the class attributes from the test set, and the predicted class attributes returned from $predict()$. This method will compare the expected class with the predicted class and return a decimal value representing the success rate of our model. Multiply the decimal value by 100 to get the accuracy percentage (%) and print the value.

Preprocessing:

The data set does not require any preprocessing, because the features and class attributes are discrete and non-continuous number values.

Accuracy:

Given the features of the test set, our model predicts the *Direction* of the stock with ~60% accuracy. Stock markets are difficult to predict because there are many important factors that our model could not be trained to consider. Factors such as inflation, actions by the company, and industry-wide trends may have caused a low accuracy rate.