Using unsupervised learning algorithms to Screen Stocks by fundamental financial data

Dan MacCarthy

October 2021

A statue of a bull

Description automatically generated

Project Report

Discovering patterns in financial data using Clustering Analysis

GitHub URL

https://github.com/DanMacCarthy/UCD\_PA\_Dan\_MacCarthy\_SCDAE.git

Abstract

Fundamental analysis (FA) is a method of [measuring a stock's intrinsic value](https://www.investopedia.com/terms/v/valuation.asp) by examining related economic and financial factors.

This method of stock analysis is considered to be in contrast to [technical analysis,](https://www.investopedia.com/terms/t/technicalanalysis.asp) which forecasts the direction of prices through an analysis of historical market data such as price and volume. [Source Investopedia website]

Our objective is to compile fundamental stock data that will describe various growth rates, measures of management effectiveness, and profitability using annual financial data over a 10-year history.

Study Design and Methodology

Use unsupervised learning methods (clustering) to attempt to identify meaningful groups of data, without training the models on labelled data. We are allowing the models to find patterns within the unlabelled data, which may be of value in identifying potential opportunities for investment.

The data used to develop the model are the annual financial returns from the Balance Sheet, Income Statement and Cashflow statement for all the companies listed on the Standard and Poor (SNP500) index.

The study was purposefully designed to be agnostic to the following parameters:

**Stock Price** – no price information was given to the model

Measures such as market capitalisation, enterprise value, or price to earnings ratio were

excluded from the input features

**Company Size** – no data concerning the size of the company was disclosed to the model.

Other than all samples used in the study dataset were companies listed on the SNP500

**Company Sector** – sectors describe the company’s domain, examples are consumer staples, basic materials, pharmaceuticals, or technology. It was left to the model to find similar patterns of growth, profitability, debt and returns on investment, without regard to sector.

Dataset

There are many sources for fundamental stock data on the internet. But it is difficult to find 10-year fundamental data. For example, Yahoo finance fundamental data is for only 5 years.

Fundamental stock data was sourced from Nasdaq Data Link (<https://data.nasdaq.com/databases/SF1/documentation>)

This is reference grade fundamental stock data, providing up to 10 years of history with over 100 financial indicators and ratios for all companies listed on US stock exchanges.

Graphical user interface, text, application, chat or text message

Description automatically generated

The Source Data was downloaded using the Nasdaq Data Link web API

The Source Data is a 2-dimensional table with 10 years of fundamental stock information for more than 5,000 companies active on US stock exchanges.

The stock data is structured on an annual and quarterly basis.

As we are looking for stocks that have performed consistently well over the 10-year time horizon, we will use the annual data and ignore the quarterly information. Using annual data has the added advantage of smoothening out the inherent Qtr. to Qtr. variation which can have high variability.

Rows represent stocks for all companies listed on US exchanges, each stock can have up to 11 rows, each row representing a calendar year.

The source data has 110 columns or features describing the company’s fundamental financial information. This information has been extracted from the company’s financial statements (Income Statement, Cash Flow Statement and Balance Sheet)

The source data CSV was processed off-line:

Annual financial data was subset from the quarterly data (Filter Rows by Year)

Columns (Features) that had most relevance in measuring the financial health of the company were selected from the 110 columns of financial indicators.

This reduced the source data CSV to a more manageable size in preparation for loading.

Implementation Process

Headings in this section are numbered to correspond to section numbering in the Python script ‘Project\_Cluster’

**2.0 Feature Selection**

Each of the 110 feature columns were reviewed and a decision made on whether to include it in our analysis or discard it.

The objective was to select features that reflect what a fundamental analyst would use to determine the strength and value of a business.

The data features were selected from about 110 financial indicators/metrics from the source data.

The first iteration ‘Feature Selection’ was done off-line before loading the dataset.

**Selection criteria:**

Features based on stock price were excluded, we want the stock screener to be agnostic to price information, to focus on the intrinsic financial health of the companies, not the value that the market has put on past and future growth.

Thus, features that carry price information such as price to earnings ratio and market capitalisation were excluded from this analysis.

Features those fundamental analysts would use to evaluate the intrinsic value of the company stock were selected, such as measures of growth, gross and net profit margins, and debt relative to equity were included.

Measuring Profitability

Both gross profit margin and net profit margin features will be included.

Growth Rate Features

Key Features that capture growth rates for sales, earnings, free cash flow from operations and book value are the foundation for fundamental analysis. Growth rates will be expressed on a per share basis. These features will be included in the analysis.

Dividend Yield

Mature, high cap companies with relatively low growth usually pay a dividend.

Younger, high growth companies usually do not pay a dividend.

Thus, we will use dividend yield as a marker to inform our models on each companies’ dividend policy, as this will be a useful feature to discriminate between ‘growth’ and ‘value’ companies

Measuring Debt

Debt to Equity Ratio will be used to describe the company’s debt relative to the value of the company measured by ‘equity

**2.0 Feature Selection (Continued)**

Metrics of management performance

Return on Equity (ROE)

ROE is considered a gauge of a corporation's profitability and how efficient it is in generating profits.

investors can consider an ROE near the long-term average of the S&P 500 (14%) as an acceptable ratio and anything less than 10% as poor.

Return on Invested Capital (ROIC)

Return on invested capital (ROIC) is the amount of money a company makes that is above the average cost it pays for its debt and equity capital.

**3.0 Importing Data**

Two SHARADAR datasets were downloaded from Nasdaq Data Link

1. The complete fundamental financial data for all companies listed on US stock exchanges (‘SHARADAR\_all\_stocks\_selected\_features.csv’)
2. The current tickers and names of stocks listed on the Standard and Poor 500 index, which is used a commonly used benchmark for the market. (‘SHARADAR\_SP500.csv’)

Both datasets were downloaded as CSV files from Nasdaq Data Link.

Both datasets were loaded into PyCharm IDE as pandas DataFrames using the load\_csv method

**3.1 Extract the stock data of interest for the study**

The SNP500 stocks were extracted from the source data DataFrame using the merge method as a filter.

**4.0 Data Cleaning**

The datatypes for each feature were checked to verify that the datatype is appropriate for the analysis.

The ‘dividend yield’ feature required ‘NaN’ values to be replaced with ‘0’, using the fillna method.

A summary of missing values 'NaN' by column was generated and rows that have 'NaN' for up to 5 features were dropped.

Features that had significant number of ‘NaN’ values were dropped where equivalent alternative features were available.

The data was checked for ‘out-of-range’ values using the describe( ) method

Where the min / max data values did not make any sense, these rows were discarded using a filtering condition.

During this phase the feature selection was completed with several features discarded and only those features required for the study retained.

**5.0 Feature Engineering**

The fundamental stock data was not yet in a structure that would suit clustering analysis.

It was necessary to carry out significant feature engineering to transform the features into a suitable structure.

Compress time-series data into a summary metric.

I needed to transform this time series structure:

Table, Excel

Description automatically generated

Into this:

Table

Description automatically generated with low confidence

For many features it was best to use the most recent annual values:

Net profit margin, gross profit margin, return on equity, return on capital invested, equity to debt ratio, and dividend yield features were engineered by simply extracting the most recent annual filing for the relevant metric.

For the 4 measures of growth

earnings per share,

book value per share,

cashflow from operations per share

revenue per share

I wanted to use compound annual growth rate as a metric, but CAGR can’t take negative values.

**5.0 Feature Engineering (cont’d)**

Instead, I calculated the year-to-year change in the above features and took the mean over a 10-year period.

I found the following technique on Datacamp’s Certificate in Data Analytics for Finance, to calculate year-on-year growth

To compute an average growth rate over 10 years

Get the start value for the first year

Divide each year’s value by the start value (normalise time-series using start value)

Take the mean of the series of year-on-year growth

This is the average growth rate for the feature

Implementation in Python

bvps\_growth = []  
**for** ticker **in** snp\_tickers.index:  
 series = snp500\_raw\_features.loc[ticker,**'bvps'**] *# series object to store bvps values for each loop on ticker* start\_value = series[0] *# identify the starting value of bvps* long\_term\_growth = series[0:-1].div(start\_value).mean() *#subset all the values, divide by start\_date, take mean* bvps\_growth.append(long\_term\_growth) *# append to list on each iteration*

*# Create dataframe to store the values for each company*df7 = pd.DataFrame(bvps\_growth, columns=[**'bvps\_growth'**], index = snp\_tickers.index)

Thus, the transformed features for eps, bvps, cashflow and revenue per share were a measure of long-term average growth trend over the last 10 years. A crude measure, but it rewards consistent high growth rates over a long-time horizon.

All the original features were re-engineered to compress the time series data into single metrics.

No absolute values remain everything is a ratio or a growth rate, so patterns of financial strength and growth can be found independent of the company sector, stock size or stock price information.

The engineered features are all ‘larger is better’ metrics, higher values are stronger and better than lower values

The original features were stored in SNP500\_raw\_features.csv

The engineered features were stored in SNP500\_new\_features.csv

**6.0 Exploratory Data Analysis**

Each feature was plotted on a histogram / kde plot to estimate what the distribution pattern looked like. Most features looked non-gaussian, which was useful in choosing a scaler algorithm later in the analysis.

The linear correlation between the features was plotted on a triangular heatmap

Chart, histogram

Description automatically generated

I expected some features to be more strongly correlated

I found that one feature ‘equity to debt’ was perfectly correlated to ‘total assets to total liabilities’, so I deleted one of these features as they were essentially the same.

The feature box-plots were very compressed due to some extreme outliers.

Chart, box and whisker chart

Description automatically generated

I took note of the tickers for the positive outliers as I wanted to see what was driving these anomality’s and they could be interesting. I then dropped these extreme values from the dataset as they would bias the clustering.

**6.0 Exploratory Data Analysis (Continued)**

Here is the feature box plot with the extreme outliers removed

Chart, box and whisker chart

Description automatically generated

On the box plot, the interquartile range (75 – 25 percentiles) the scale is very compressed by outliers, with most points sitting close to the zero line.

This does not look like the usual box plot, but it is good news as our objective is to identify standout stocks with superior performance.

**7.0 Data Preparation for Cluster Analysis**

The data required significant pre-processing for cluster analysis

7.1 The features data for the SNP500 companies were transformed into a NumPy array

7.2 The data was scaled

My first iteration used the whiten normalising method to standardise the data.

This was not effective for the downstream PCA dimensional reduction step as the features could not be transformed into a small set of principal components.

The root cause of the problem was the distributions of the features were non-normal.

PCA components un-scaled data PCA components whiten scaled data

Chart, bar chart, histogram

Description automatically generated

Chart, bar chart, histogram

Description automatically generated

Scaling the data to a range using MinMaxScaler had better results

PCA components using MinMaxScaler

Chart, bar chart

Description automatically generated

The data was scaled using MinMaxScaler

**7.0 Data Preparation for Cluster Analysis (Continued)**

7.3 Estimate the intrinsic dimension of the feature data

The first step was to estimate the intrinsic dimension

The principal components were plotted against variance and an intrinsic dimension of 3 was estimated from the plot

Chart, bar chart

Description automatically generated

**8.0 Dimension reduction of features using PCA method**

A PCA model was created with n\_components = 3

The model was fit to the scaled data

The transform method was called to reduce the dimensions of the scaled features from 10 to 3

The scaled and transformed data was now ready for clustering analysis

**9.0 Partition based clustering (using K-means algorithm)**

The K-means algorithm relies on the idea that there are a specific number of data groups, called clusters. Each data group is scattered around a central point with which they share some key characteristics

The K-Means algorithm requires an estimate of the number of clusters (K)

**9.1 Determine K (No. of clusters) using elbow plot (num\_clusters vs distorsion)**

Chart

Description automatically generated

From inspecting the elbow plot, the estimate of optimal K is between 20 to 30 clusters.

I will use K=20 as the smaller number of clusters is more manageable for analysis.

**9.2 Run K-Means**

The SciPy.Cluster.vq package was used to apply the K-means algorithm

Cluster centres and labels were generated

The cluster labels were loaded into a dataframe and merged with the new\_features dataframe called ‘cluster\_labels’ for downstream analysis

The groupby method was used to group feature means by cluster labels to determine the characteristics of each cluster.

**9.3 Visualising the cluster pattern**

A two-dimensional t-SNE plot was generated colour coded to the K-means clusters to visualise the distribution of clusters

t-SNE plot of K-means clustering

Chart, scatter chart

Description automatically generated

**10.0 Agglomerative based clustering (using Hierarchical clustering)**

While K-Means algorithm is concerned with centroids, hierarchical clustering tries to link each data point by a distance measure, to its nearest neighbour, creating a cluster.

There are different linkage methods

Ward – Tends to look for spherical clusters, very cohesive inside and extremely differentiated from other groups. Finds clusters of similar size. It only works with the ‘Euclidean’ distance measure.

Complete – clusters created using this method tend to be comprised of highly similar observations, making resulting groups quiet compact.

There are 3 distance measures - Euclidean, Manhattan and Cosine

The combinations of linkage methods and distance measures generate different cluster structures.

After some trial-and-error iterations, the Ward linkage method with Euclidean distance measure was selected.

The linkage, fcluster, dendrogram packages from the scipy.cluster.hierarchy library were used.

The scaled data was fitted to a Linkage model using ward linkage method and Euclidean distance.

A dendrogram plot was generated using stock tickers as the data labels

Chart, histogram

Description automatically generated

The plot is too busy to pick out individual ticker labels.

But this plot was used to pick the optimum distance of 1.25 to select the number of clusters for further analysis.

The cluster – ticker pairs were saved to 'hierarchical\_clusters.csv' for analysis

Results

Evaluation of clustering is challanging without a pre-determining, pre-defined class.

With a pre-determining class, we can use cross tabulation.

If we don’t have labels, we need a way of measuring the quality of a clustering solution that only uses the clusters and the samples themselves.

A good clustering solution has tight clusters, meaning that the samples in each cluster are bunched together not spread out.

For K-Means, measures such as inertia / distortion measure clustering quality

Distortion measures how far samples are from their centroids.

Lower values of distortion are better

The elbow plot below demonstrates the quality of clustering

Chart

Description automatically generated

We used K=20, and the elbow plot shows that this is a good trade off between the quality of the clusters and the number of clusters.

Results from K-Means Clustering

The groupby method was used to group feature means by cluster labels to determine the characteristics of each cluster.

For each cluster group the average of each feature was calculated

Low average values were given a warning flag ‘red-flag’

High average values were given a green flag

Clusters 2, 4 and 9 have strong financial metrics

Clusters 0, 1, 10, and 13 have weak financial metrics

Looking for patterns in features averaged across the cluster groups



Results from Hierarchical Clustering

Dendrogram of SNP500 companies clustered by financial data

Chart, histogram

Description automatically generated

A CSV file 'hierarchical\_clusters.csv' with cluster labels and corresponding tickers was exported.

Due to time constraints, it was not possible to carry out further analysis.

An obvious disadvantage of this study is the density of the results generated

The results are too busy, and the data overwhelms the ability to analyse it.

Better to present smaller datasets to this process, such as companies grouped by sector, to determine ‘best-of-breed’, where one or two companies outperform their sector peers.

Less dense results will enable better focus

Insights

Insight 1

As a proof of concept exercise this project was useful.

Cluster analysis methods can be applied to stock fundamental data to find common patterns and identify superior performance.

Further refinement of the method would be required, but it has potential.

Insight 2

The SNP500 companies are a homogeneous group

The dendrogram shows that most clustering occurs early at small distances.

The clusters formed are large, grouping together many companies with similar financial patterns.

It is the singletons, those companies that resist clustering until late in the hierarchical clustering process that are of most interest, as they have strong fundamentals.

Insight 3

Just because the models can process masses of data does not mean that it is wise to apply huge datasets to them.

The results are too busy, and the data overwhelms the ability to analyse it.

Better to present smaller datasets to this process, such as companies grouped by sector, to determine ‘best-of-breed’, where one or two companies outperform their sector peers.

Less dense results will enable better focus.

To attempt to analyse the total SNP500 constituents in one go was overreach.

Insight 4

Hierarchical clustering is more explainable, more visual, and more intuitive than K-Means clustering.

References

Practical Statistics for Data Scientists – Peter Bruce, Andrew Bruce & Peter Gedeck

O’Reilly 2020

DataCamp - Specialist Certificate in Data Analytics Essentials

DataCamp - Certificate in Data Analytics for Finance

Investopedia website https://www.investopedia.com