Machine Learning Engineer Nanodegree

Capstone Proposal

David Pham

December 2nd, 2017

Proposal

Investment and Trading Capstone Project: predicting stock prices

Domain Background

According to the link in the proposal about the investment and trading capstone project

> Investment firms, hedge funds and even individuals have been using financial models to better understand market behavior and make profitable investments and trades. A wealth of information is available in the form of historical stock prices and company performance data, suitable for machine learning algorithms to process.

Predicting stock prices is difficult task but the usual goal is to diversify our risks and try to get the best return possible. Understanding the decision of the our algorithm is also paramount and it would be interesting if we could create algorithms that could discover or retrieve structures in the financial instruments.

This subject is interesting because it would allow to study algorithms to classify financial with minimal information and maybe discover clusters only with the price movements. We are interested in the dependency structure of the financial assets.

This algorithm should find clusters of assets that should move together usually in lock step. In the equity world, this is usually the case for stocks belonging to the same sector or same countries: the relative performance of the share price of Google and Apple tend to move together as they are both giant tech companies, whereas Goldman Sachs and JP Morgan also move together.

If this analysis seems trivial for the vanilla equities, this is much more interesting when financial derivatives are included among the assets. For example, the daily performance of the price of a vanilla future on Google should be similar to the performance of Google share price.

The ultimate goal, beyond the scope of this project, would be to test the algorithm on a mixed dataset with futures and equities and to observe if the algorithm could discover and create this underlying/overlaying and sectorial structure.

Finding the dependency structure beyond the price movement could help to detect new investment strategies, but also about data control quality.

Dependency discovery and correlation cluster is a common problem. See Liu et al. with their youtube video, Stackoverflow discussion, Statsoverflow discussion. Copula theory is also well-known method to find dependency structure with only price inputs. But it only works on low-dimensional output.

Problem Statement

In this project, we would focus to create an algorithm whose input are closing prices (adjusted or non-adjusted, to be determined) of US equities and outputs two things:

1. The group of our unsupervised algorithm
2. The ability to create a vector representation of the map of the financial assets.

The algorithm could be extendable to financial futures instruments as long as we can keep end of day prices.

In a technical jargon, the world of financial assets could be assigned to a id, and we want to apply deep learning techniques to classify the assets and maybe discover hidden structures. This is a unsupervised learning problem, but we might have some intuition in some group of data.

The goal is to assign group to every asset in our dataset and each group should capture the price movement of each of its constituents.

More precisely, measure of association between assets needs to be computed for created and these measures should only rely on the end of prices of the assets. The first natural steps will be to compute the relative difference of the prices (their daily percentage performance), find a smart standardizing procedure and then apply standard measure of association (typically Pearson's correlation).

This measures of association could be translated as a proxy for a distance between the assets. As we have distance, we could apply clustering algorithms to find the nearest neighbors and apply linkage methods.

To explain how to create a vector representation of the financial assets, one can map each ticker to a unique numerical id. Then we could create observations similar to a *corpus* by grouping the financial asset and the assets whose daily relative performance is the closest. A neural network could be then be trained to predict those neighbors given the financial asset, and conversely predict the financial asset given the neighbors. In this sense, this is similar to Word2Vec, whose goal is to find a vector representation of words (which are assigned unique id).

Datasets and Inputs

As suggested from Udacity, we would like to use the data provided from Quandl. They have an open data providing the required data with the Wiki EOD Stock Prices.

We intend to use the adjusted and non-adjusted closing price of all the equities. We intend to use the whole date range of the data set, but we give us the right the restrict to a more limited date range if we lack computation power or if the algorithms would only work on recent data.

Splitting dataset into training, test and validation set could be tricky. We intend to make in two dimension: one with the time and the other on the ticker dimension. As for the time split, training set could be all the data whose date is before 2010, test set is data between 2011 and 2015 and validation set is 2017. Regarding the split in ticker, we could just separate the ticker to have sets with 60, 20, 20 percents of tickers.

As for additional features to control our output would be to gather the industry of each ticker. According to this answer and pandas-finance, it is feasible to scrap yahoo financial to get this information.

Solution Statement

We might have the following sub problems to solves:

1. We apply some transformation to the data by computing the Z-scores of the day to day price difference. For this, we would need to compute a sample mean, and sample deviation and the exact procedure should be determined. This works as a standardizing steps.
2. Samples correlations will be computed (this could be computationally intensives with a lot of data).
3. An neural network will try to find clusters with the new transformed data.

The deep learning algorithm will be inspired from the Word2Vec from Google for word embedding. Please, read the problem statement for more details.

Benchmark Model

Obviously, as we try to create clusters, a base model could be random guessing.

Another more sophisticated baseline would be to use the correlation matrix and Hierarchial clustering, also see scikit, to create a graph of clusters with the maximum distance between groups as linkage criteria by using the absolute value of the element of the correlation matrix as a proxy for a metric distance. The algorithm would start with a graph with no edges and would gather groups only if their correlation is high.

Evaluation Metrics

We will use the silhouette score as evaluation metrics. The silhouette score measures how appropriately the data have been clustered.

As for the vector representation of the financial assets, we would use the accuracy of our classifiers. As the classifiers tries to predict the nearest neighbors when using the standardized relative daily performance. A good representation should be attained when our classifiers have good performance.

Project Design

On an abstract level, the project would be designed as follow. Data will be acquired and the necessary transformation will be performed. From this point, the benchmark models will be studied in order to get some intuition, then the models creating the vector representations will be implemented. This last step should be the main challenge of the project as many possibilities exists to find a good model. The analysis and diagnostic of results should then be conducted, hopefully with data visualizations.

On a more detailed view, we would follow the following steps. The first part is to download and understanding the data. A python module should probably be written to record all the transformations and make it reproducible. Then data is cached to avoid doing the same computation repeatedly . Then the benchmark model is implemented to set our expectations. Then, we will create the neural network (in keras and tensorflow). Tensorboard should be then used to inspect the embeddings and we should try to make exports in csv and maybe to polish the visualizations. Finally, we should try several hyper parameters to observe any difference and report them.