UDACITY CAPSTONE PROJECT

Hugo Perrier, March 2017

1. Introduction

Stock Market

The ownership of a company can belong to a single person but most of the time the ownership is divided between several shareholders. The values of these company shares depend of the total market valuation of the company, which may change over time depending on a variety of factors. Stocks or company shares can be bought and sold; there is therefore a market for company shares called the Stock Market. Stock trades are performed in stock exchanges such as:
- NASDAQ
- London Stock Exchange Group
- Tokyo Stock Exchange Group

Depending on the country they are based in, these stock exchanges have different opening days and times. Stock trades can only be executed during the opening hours of a stock exchange.
The most valuable companies in 2015 according to the FT500 ranking (link) are:
- Apple
- Exxon Mobil
- Berkshire Hathaway
- Google
- Microsoft

A good capacity to understand and predict movements in the stock market is crucial for investors to make profitable investments. To help investors choose which investment will be most profitable, they can use large amount of data on the history of the stock prices of companies. To process all these data, predictive models are created using machine learning. Machine learning models are "trained" to predict future stock prices based on a set of available features. This report explores how machine learning can be used to predict stock prices.

Objective of the project

The objective of this project is to create a predictive tool that uses machine learning to predict future value of the NASDAQ 100 index (NDX) using historical stock prices of different companies. The NASDAQ 100 index is a stock market index related to the capitalization value of the 100 largest non-financial companies.

Section 2 of this report describes the data used to create a machine learning model for stock prices prediction, section 3 describes the numerical models involved in the creation of a prediction tool and section 4 shows results of the predictive model built.

2. Data description

Historical stock prices data

To create a machine learning model to make predictions, it is necessary to first "train" the model using past data. In the context of stock market pricing, the model is trained using historical data of the stock prices. For example we can use data from the past period 2003 to 2005 to train a model and then use that model to make predictions about the future. The stock price data consist of the following information:

Open High Low Close Volume ExDividend SplitRatio

and adjusted values:

Adj.Open Adj.Close Adj.Low Adj.High Adj.Volume

For a given day, the "Open" and "Close" values are the values of a stock at the opening and closing of the stock exchange. The "High" and "Low" value are the maximum and minimum values that the stock has reached during that day. The "Volume" is the total amount of stock that was sold on that day. A company can decide to give dividend to shareholders ("Ex-Dividend" feature) or modify the number of shares that compose the company (adjusting the share value to keep the total capitalization value constant), this is described by the feature "Split Ratio". After this actions are taken, the values of Open, Close, High and Low are adjusted accordingly (Adj features).

Data Acquisition

Quandl API

The python Quandl API allows users to query historical stock prices from databases. With the free version only one data point per day can be accessed and there is a maximum amount of queries that can be performed in every 24h period. This is used to get the stock prices of the companies in the NASDAQ 100.

Pandas stock price data reader

Historical stock prices from yahoo finance can be queried directly using a pandas module. This is used to get the values of the NASDAQ 100 index.

List of NASDAQ 100 companies

To obtain the current list of companies included in the NASDAQ 100 index, web scraping is performed on the NASDAQ website.

Data Format

The data obtained from Quandl API queries are in a pandas DataFrame format. The size of a dataset corresponding one year of historical stock prices for a single company is about 26kb in size.

Data Preprocessing

Missing data

Historical stock prices for a company on a given day may not be available for the following reasons:

- The company didn't exist at that time (Google was created in 1998)
- The company existed but was not traded in the stock market (Facebook entered the stock market in 2012)
- The stock market where that company's stock is traded was closed on that day
- The historical stock prices of that company are freely accessible in Quandl API

However, to train a machine learning model, the datasets can not contain missing values. Two options are available to ensure the dataset does not contain missing values:

- Model the missing values
- Remove the datapoints containing missing values

Both options can impact the predictions therefore the way missing values are handled needs to be explained.
In this project, all the companies whose data were not accessible through Quandl were not included in the model. If on a given day some companies have stock price data but others don't, the missing data are replaced with the data from the previous working day (This is called a forward fill method).

Below is a list of data that couldn't be accessed using Quandl:

- JD, NCLH : Non American companies data are not available in the Quandl WIKI free database
- KHC : Kraft Heinz Company didn't exist in 2013, Kraft and Heinz merged in 2015
- PYPL : Paypal was a wholly owned subsidiary of eBay until 2015
- WBA : Walgreens Boots didn't exist in 2013

Feature naming

When the data for different companies are queried from Quandl, they all have the same feature names; it is thus necessary to run a renaming operation when joining the datasets of the different companies. The company "ticker" symbol is simply

added to the feature name: "Open" feature for "Apple" company (Ticker "AAPL") becomes "AAPL_Open".

Split of the data into train, cross validation and test datasets

To create a machine learning model we create a set of data called training set for which the true value of NDX are known. This set is used to find the model (with fixed hyperparameters) coefficients that minimize the error between predictions and true values of NDX.
Then a cross validation set similar to the training set is used to find the hyperparameters that make the predictions as general as possible. In other words, the model shouldn't just be able to predict NDX values from the dataset used for training but it should predict well NDX values for any other dataset.
Finally a test set is created to evaluate the performance of the model on data that were not used in the training process. The test dataset has to contain data posterior to the data in the training and cross validation sets as it is not possible to train a model using data from the future.

Creation of the final features

The objective of a predictive model is to predict future values of the NASDAQ 100 index, therefore company stock prices of day N should be used to predict NDX value of day N+1 or N+x. It is thus necessary to shift in time the NDX column compared to the feature columns.

Finally, we need to choose which data are used as features to predict NDX, we can't use all of the historical data of every company prior to day N+1 to predict NDX on day N+1. It is therefore necessary to decide how many historical data to use for prediction, which features we want to use and adapt the dataset accordingly. A mock-up dataset that could be used in a machine learning model using just the "Open" data from the previous 2 days of companies "X" and "Y" is shown below (the date is written down to help with explanation but it is not used as a feature):

Date	X Open Day N-2	X Open Day N-1	Y Open Day N-2	Y Open Day N-1	NDX day N
01/01/98	100	105	10	11	99
02/01/98	105	110	11	12	87

3. Numerical models

Software requirements

The software requirements to build the predictive models are:

- python: Main programing language
- numpy: Package for scientific computing in python
- pandas: Package for data structures with python
- matplotlib: Plotting with python
- sklearn: Package for machine learning in python
- urllib: Package for data fetching across the web
- re: Package for regular expression operations
- Quandl: API to access historical stock prices from Quandl databases using Python

Regression models

A regression model relates a set of features (historical stock prices of companies) to a prediction (NASDAQ 100 index "NDX"). The regression model specifies the type of relation between the inputs and outputs of the model (ex. linear relation) but the exact parameters of the model have to be calculated using known data.

In Machine Learning, an optimization algorithm is used to find the model parameters that minimize the difference between the regression model output and the true value of the output on a "training" data set. Then the performances of different regression models (or similar models with different "hyperparameters") are compared on a different data set called "cross-validation dataset". This makes sure that the accuracy of model predictions are not limited to the training dataset but generalizable to other datasets (a model that does not generalize well to other datasets is said to "overfit").

The models used in the present work are:

- Linear Regression: parameters normalize (True/False)
- Support Vector Regression: parameters kernel (linear/poly/rbf)

Data clustering

Each of the companies in the NASDAQ 100 has an influence on the value of the NASDAQ 100 index value but using the historical data from all these companies to create a machine learning model would have a high computational cost (270 working day per year per company = 270 datapoints per year, 100 companies * 12 stock data per company per working day * Nday (number of days of historical data used) = 1200 Nday features, input matrix has a size of 270*1200Nday). It would then take a long time to train the models on a laptop.

To reduce the amount of data to work with companies that have similar behaviors can be grouped together. To do so, an unsupervised learning "clustering technique"

is used: KMeans clustering. The Kmean clustering method takes as input the historical stock prices of all companies and a user defined number of desired clusters and outputs a list of companies in each cluster.
In practice, we calculate the daily variation for each datapoint:
"Variation" = "Close" - "Open" and use the "Variation" variable as input data for the clustering.

Computational resources

The computer used for these calculations is a Macbook Pro 13-inch, Late 2011) with 8 GB of 1333 MHz DDR3 RAM and a 2.4 GHz Intel Core i5 processor. The linear regression models (linear regression + SVR(kernel="linear")) could be trained in a fraction of a second on a laptop with a 2 years long dataset for the training set, 2 features per companies and a few companies. On the other hand the training times for the non-linear models quickly become prohibitively long as the number of features and datapoints increases.

Prediction score

The R^2 score is used to score the result of the predictions. A R^2 score of 1.0 means the predictions correspond exactly to the true output, a R^2 of 0.0 is returned when the model predicts a constant output independently from the features and the R^2 score can be negative if the predictions are very different from the true output.

4. Results and analysis

Company clusters

We apply the clustering method described in the previous section for different number of clusters. The data used for the clustering corresponds to the stock prices data of the NASDAQ 100 companies for the year 2013. Results for nCluster = 10 are shown below:

Composition of clusters:

1. Analog Devices, Inc., Applied Materials, Inc., Broadcom Limited, Intel Corporation, KLA-Tencor Corporation, Lam Research Corporation, Microchip Technology Incorporated, Maxim Integrated Products, Inc., NVIDIA Corporation, QUALCOMM Incorporated, Skyworks Solutions, Inc., Texas Instruments Incorporated, Xilinx, Inc. |
2. Dollar Tree, Inc., Fastenal Company, Hasbro, Inc., Marriott International, Mattel, Inc., Mondelez International, Inc., Ross Stores, Inc., Tractor Supply Company, Ulta Beauty, Inc. |
3. American Airlines Group, Inc., Adobe Systems Incorporated, Akamai Technologies, Inc., Amazon.com, Inc., Baidu, Inc., Facebook, Inc., Liberty Interactive Corporation, Mylan N.V., Netflix, Inc., The Priceline Group Inc., TripAdvisor, Inc., Tesla, Inc., Yahoo! Inc. |
4. Alexion Pharmaceuticals, Inc., Amgen Inc., Biogen Inc., BioMarin Pharmaceutical Inc., Celgene Corporation, Gilead Sciences, Inc., Incyte Corporation, Regeneron Pharmaceuticals, Inc. |
5. Automatic Data Processing, Inc., Cognizant Technology Solutions Corporation, Monster Beverage Corporation, Paychex, Inc. |
6. Charter Communications, Inc., Comcast Corporation, Costco Wholesale Corporation, DISH Network Corporation, Liberty Global plc |
7. Activision Blizzard, Inc, Electronic Arts Inc., Intuitive Surgical, Inc. |
8. Apple Inc., Autodesk, Inc., Cerner Corporation, Cisco Systems, Inc., CSX Corporation, Cintas Corporation, Discovery Communications, Inc., Discovery Communications, Inc., Expedia, Inc., Fiserv, Inc., Twenty-First Century Fox, Inc., Twenty-First Century Fox, Inc., Alphabet Inc., Illumina, Inc., Intuit Inc., J.B. Hunt Transport Services, Inc., Micron Technology, Inc., O'Reilly Automotive, Inc., PACCAR Inc., Starbucks Corporation, Seagate Technology PLC, Symantec Corporation, Viacom Inc., Vodafone Group Plc, Verisk Analytics, Inc., Western Digital Corporation |
9. CA Inc., Check Point Software Technologies Ltd., Citrix Systems, Inc., eBay Inc., Express Scripts Holding Company, Microsoft Corporation, SBA Communications Corporation, Sirius XM Holdings Inc., T-Mobile US, Inc. |
10. Hologic, Inc., Henry Schein, Inc., Vertex Pharmaceuticals Incorporated, DENTSPLY SIRONA Inc. |

In this exemple, we can see that cluster 3 contains mostly tech companies, cluster 4 contains pharma companies and cluster 7 contains video game companies. Some

clusters are more difficult to describe such as cluster 2 that contains hotel, food, beauty and variety store companies.

It should be noted that the clustering is not very stable, the following parameters significantly change the content of clusters:

- Time period of the input data
- Number of desired clusters
- Number of initialization (Kmeans clustering methods randomly initialize the center (in parameter space) of the clusters, the initialization might affect the final clustering so several runs with different random initialization are performed)

It is still a good way to reduce the amount of data we use to build the predictive model.

Next day NDX prediction: Basics

This section describes the general procedure followed to build a predictive model:

1. Clustering (Optional)
   - All Nasdaq 100 Company 2013 stock price data are acquired
   - Company that can't be accessed with Quandl are removed from the company list
   - Missing values in the dataset are filled (forward fill)
   - The "Variation" = "Close" - "Open" variable is calculated
   - The desired number of company clusters is created using the "Variation" feature as input data.
   - The first company (in alphabetical order) of each cluster is chosen to represent the whole cluster.
   - A list of selected companies is saved to be used in the predictive model creation.
2. Predictive model
   - Stock price data of the companies chosen in the clustering process are acquired. The 2014-2015 period is used for the training/cross validation datasets and the 2016 period is used for the test set.
   - Missing values are filled (forward fill)
   - The "Variation" = "Close" - "Open" variable is calculated. (It can be used as a potential engineered feature)
   - NASDAQ 100 index data are acquired from yahoo finance (using pandas finance data reader).
   - The features to be used to build the predictive model are chosen:
   - Type of stock price data (ex. ["Open", "Close"] or ["Open", "Variation", "Close"])
   - Number of days used for prediction (ex. data from the last 3 days, data from the last 7 days, ...)
   - The final feature matrix is created (drop undesired features, create previous days features, give the feature matrix the right shape, etc)

- Choose a machine learning regression model
- Train the model using the training data
- Predict NDX and score the model on the train, cv and test sets

Next day NDX prediction: Features

In this section, the influence of the choice of features on predictions is tested. A linear regression model is fitted to the data of the companies Amazon and Netflix from the last 3 days.

Features	**Score**
Open, Close	0.191118
Var, Close	0.188922
AdjOpen, AdjClose	0.266558
AdjOpen, AdjClose, Ex-Dividend	0.266558
AdjOpen, AdjClose, AdjLow	0.515742
AdjOpen, AdjClose, AdjHigh	0.241427
AdjOpen, AdjClose, Split	0.260417

The results presented in the table show that the best prediction score is obtained when adjusted variables are used. In particular, using just the "AdjOpen" and "AdjClose" features give the best result, adding more features do not increase the prediction score. For the rest of the report the features "AdjOpen" and "AdjClose" will be used.

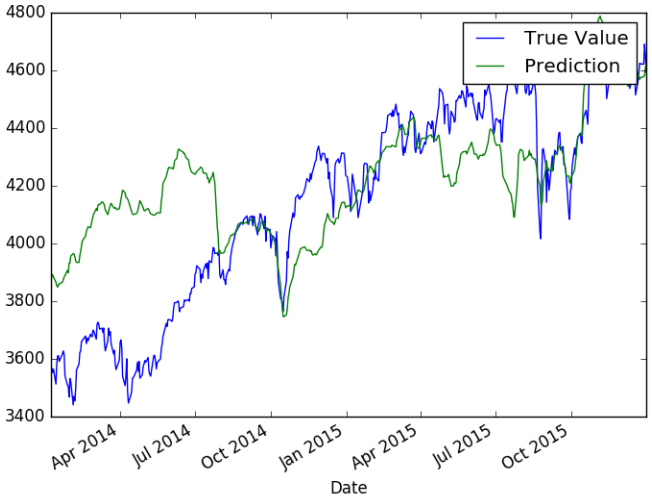
Next day NDX prediction: Single company data

In this section a linear regression model is fitted using "AdjOpen" and "AdjClose" data from the last 3 days for a single companies. The table below shows the best prediction results on the test dataset.

Company	**Score**
Maxim Integrated Products, Inc.	0.823026
Microchip Technology Incorporated	0.796815
Cisco Systems, Inc.	0.706128
Lam Research Corporation	0.656457
T-Mobile US, Inc.	0.607240

The company that yields the best prediction data is Maxim Integrated Products: "Maxim Integrated is an American, publicly traded company that designs, manufactures, and sells analog and mixed-signal integrated circuits." [(Wikipedia)](https://en.wikipedia.org/wiki/Maxim_Integrated)

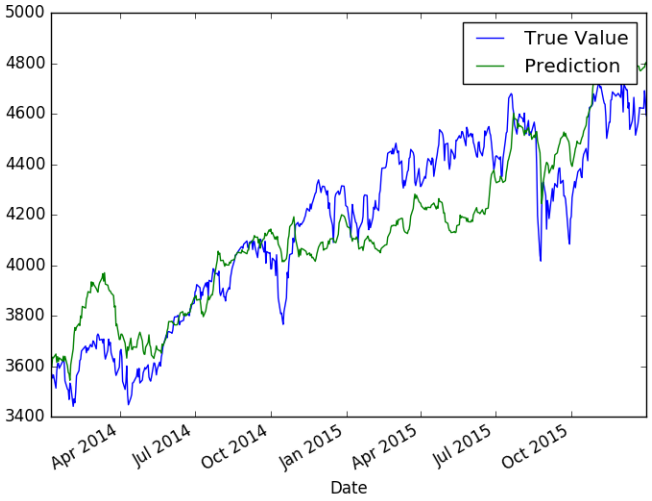
Results (using Maxim Integrated Products data as features) on the training set



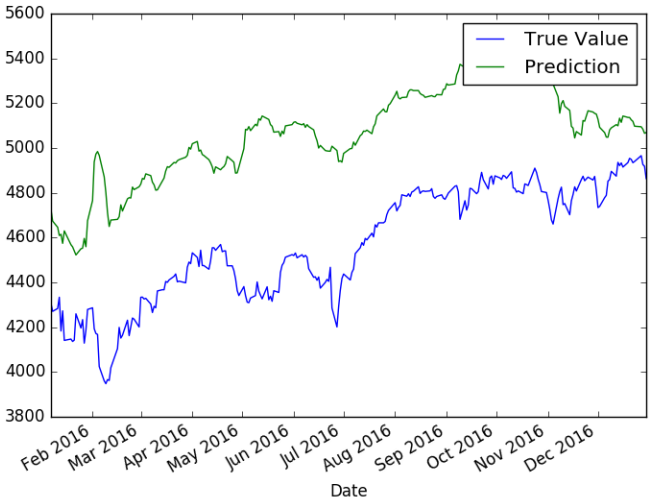
Results (using Maxim Integrated Products data as features) on the test set

It appears that the predictions on the training set are not so good but predictions on the test set are much better. For comparison, the results for the company Facebook that gives much lower results on the test set (Negative score) are shown below:

Results (using Facebook data as features) on the training set



Results (using Facebook data as features) on the test set



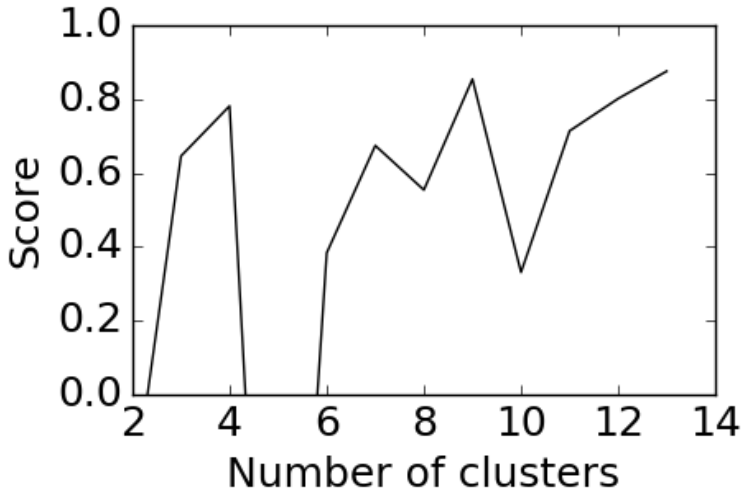
We observe that the predictions on the training set are better but the test set prediction are quite different from the true values.

Next day NDX prediction: Number of clusters

Using data of more than one company could improve the results, however including too many companies would increase the computational cost. Selecting the companies that give the best results on the single company test is probably not a good idea as this company might carry mostly the same information (Indeed the top 3 companies are all related to sales of electronic components). A better way to do it is to use a clustering algorithm to group similar companies and pick a company companies from each cluster to make predictions.

In this section the number of clusters is varied and we show the influence on the prediction results. Here the features "Open" and "Close" of the last 3 days are used for predictions.

Result on the test dataset

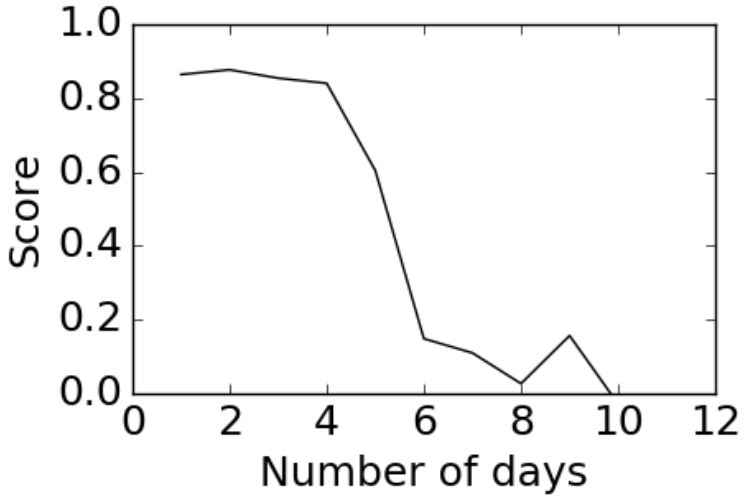


The graphs above show that the predictions on the test set are very variable, some have negative score (prediction very far away from expected result) and some yield better prediction score. As mentioned before, the clustering is very sensitive to the number of cluster chosen so that could explain the variability of the results.

Next day NDX prediction: Number of days of historical data

In this section, the influence of the number of days of historical data used for prediction is tested.

Result on the test dataset



It appears that using fewer days for prediction gives better results. The best prediction score is obtained using 2 days.

Next day NDX prediction: Machine Learning Model

In this section, the influence of the Machine Learning model is tested.

Non linear models

- The SVR(kernel="poly") model could not be used as the computation cost was too high.
- The SVR(kernel="rbf") strongly overfitted the training dataset, this is probably due to the small number of data (540 for a 2 years period) compared to the number of features. The predictions on the test dataset were then very bad.

Linear models

The linear model used (Linear Regression, Ridge Regression, SVR(kernel="linear")) showed very similar results. The ridge regression was just slightly better than other models.

To make better predictions, it would probably be better to have access to much more data than daily stock prices. Some expensive databases content stock prices data with frequencies lower than a second. With this amount of data it would be possible to prevent the non-linear models from overfitting and yield better results.

5. Conclusions

In this work a machine learning model was set up to predict future value of the NASDAQ 100 index using historical stock prices of NASDAQ companies. A pipeline to acquire data, preprocess data, create training, cross validation and test data sets, fit the machine learning model and make predictions was set up.

It was shown that using Adjusted data as features makes a big difference in the prediction score. Then predictions were made using data from just one company and it was shown that high prediction scores can be reached. Using the stock prices data from the company "Maxim Integrated Products" gave the best results.

A clustering strategy was tested to group similar companies and to use just a few companies to represent all NASDAQ 100 companies when doing predictions. The company clustering did not improve the prediction results compared to using just a single company to make predictions. Another parameter that was tested is the number of days of historical data used to make predictions. It was shown that using just the 2 or 3 previous days to make predictions gives the best prediction results.

Finally, different machine learning models were tested and the linear regression models gave the best results. It was argued that the limited number of datapoints (just daily stock prices) caused the non-linear models to overfit and that access to more data could improve the results.