

Stock Price Prediction Project

Egar Garcia

4/13/2019

Contents

1	Introduction	1
1.1	Overview	1
1.2	Objective	2
1.3	Dataset	2
1.4	Key steps	2
2	Analysis	3
2.1	Data exploration	3
2.2	Dataset retrieval	4
2.3	Data visualization	6

1 Introduction

1.1 Overview

A stock market, also called equity market or share market, is a network of economic transactions, where the stocks of public companies can be bought and sold. The equity market offers companies the ability to access capital in exchange of a portion in the ownership of the company for interested outside parties.

In the stock market, other financial securities like exchange traded funds (ETF), corporate bonds and derivatives based on stocks, commodities, currencies, bonds, etc. can be traded. However, for purpose of this project only the exchange of stocks will be considered.

It is common to use stock market and stock exchange interchangeably, but the stock market is a general superset of the stock exchange. If someone is trading in the stock market, it means that it buys and sells stock/shares/equity on one (or more) of the stock exchange(s) that are part of the overall stock market.

The stock market offers an opportunity to investors to increase their income without the high risk of entering into their own businesses with high overheads and startup costs. On the other hand, selling stocks helps companies themselves to expand exponentially, when a company's shares are purchased it is generally associated with the increased in the company's worth. Therefore, trading on the stock market can be a win-win for both investor and owner.

The stock exchange that are leading the U.S. include the New York Stock Exchange (NYSE), Nasdaq, BATS and Chicago Board Options Exchange (CBOE). The Dow Jones Industrial Average (DJIA) is a price-weighted average of 30 significant stocks traded on the NYSE and the Nasdaq, it is the most closely watched market indicator in the world and it is generally perceived to mirror the state of American economy.

Projecting how the stock market will perform is a very difficult thing to do, there are so many factors involved in the prediction, some of them emotional or irrational, which combined with the prices volatility make difficult to predict with a high degree of accuracy. Abundant information is available in the form of historical stock prices, which make this problem suitable for the use of machine learning algorithms.

Investment firms, hedge funds and individuals have been using financial models to better understand the market behavior and attempt to make projections in order to make profitable investments and trades.

1.2 Objective

The purpose of this project is to build stock price predictors. More specifically, the problem is to predict the closing prices of the trading days existing in a queried date range for a given company's stock. For the scope of this project only the companies included in the Dow Jones Industrial Average are considered.

To address the problem, different machine learning methods are used, they take historical stock data for a particular company over a certain date range (in the past) as training input, and output projected estimates for a given queried date range (in the future).

1.3 Dataset

There are multiple options to obtain the historical stock records needed to feed the machine learning algorithms, a popular source (and the one used in this project) is an API provided by IEX Group Inc. (<https://iextrading.com>), which provides access to stock historical records to developers and engineers for free.

The API provided by IEX (which documentation can be found at <https://iextrading.com/developer/docs/#chart>) allows to retrieve historical stock price information for a maximum of 5 years back to the current date. The API can provide historical records for several companies, but for this project only the records for the ones in the Dow Jones are considered.

Each one of the historical records corresponds to the information of a trading day for a specific company. An important aspect to notice is that the market is closed on weekends and some defined holidays (for which there are no records). The historic stock price records contain the following columns:

- **date**: Trading day
- **open**: Opening price
- **high**: Highest price
- **low**: Lower price
- **close**: Closing price
- **volume**: Number of shares traded
- **unadjustedVolume**: Number of shares traded also considering companies adjustments
- **change**: Change of closing price relative to the day before
- **changePercent**: Change in percent value
- **vwap**: Volume Weighted Average Price
- **label**: Formatted version of the date
- **changeOverTime**: Metric considered to measure the changes bases on weighted dates

1.4 Key steps

1. Building the mechanisms to retrieve the dataset into the local machine's storage, and to keep it updated.
2. Performing an analysis and exploration of the dataset, aiming to select only the features that are relevant to perform predictions.
3. Building models that represent the machine learning methods used in this project to address the problem.
4. Evaluation of the different models with common training and test sets, and doing the comparison based on the RMSE against the ground truth.
5. Analyzing the predicting performance of each one of the models, and discussing the results.

2 Analysis

2.1 Data exploration

The API provided by IEX retrieves the data in JSON format, which at the end contains records, where each record corresponds to the information of a trading day for a specific company, a company is identified by a ticker symbol (also known as stock symbol).

For example to retrieve the historical stock prices for the ticker symbol “MSFT” (Microsoft Corporation) of the last 5 trading days, the call to the API would be <https://api.iextrading.com/1.0/stock/msft/chart/5d>. The following code is an example of how to retrieve those historical records and how they look like:

```
as.data.frame(  
  fromJSON(  
    content(GET('https://api.iextrading.com/1.0/stock/msft/chart/5d'), 'text'),  
    flatten = TRUE))
```

	date	open	high	low	close	volume	unadjustedVolume	change
## 1	2019-04-08	119.81	120.02	118.64	119.93	15116186	15116186	0.04
## 2	2019-04-09	118.63	119.54	118.58	119.28	17611981	17611981	-0.65
## 3	2019-04-10	119.76	120.35	119.54	120.19	16477169	16477169	0.91
## 4	2019-04-11	120.54	120.85	119.92	120.33	14209121	14209121	0.14
## 5	2019-04-12	120.64	120.98	120.37	120.95	19745143	19745143	0.62

	changePercent	vwap	label	changeOverTime
## 1	0.033	119.6321	Apr 8	0.000000000
## 2	-0.542	119.1657	Apr 9	-0.005419828
## 3	0.763	120.0173	Apr 10	0.002167931
## 4	0.116	120.2224	Apr 11	0.003335279
## 5	0.515	120.6888	Apr 12	0.008504961

As in can be seen, each historic stock price record contains the columns: **date**, **open**, **high**, **low**, **close**, **volume**, **unadjustedVolume**, **change**, **changePercent**, **vwap**, **label** and **changeOverTime**. The purpose of this project is to predict the future values of the closing price, which corresponds to the column **close**.

To make predictions is necessary to chose a subset of columns which values can be known a priori and used for the prediction, unfortunately all the column values except **date** and **label** (which finally is a variant of **date**) are unknown before the occurrence of the respective trading day. That means that the only data that can be used to predict the future closing prices are the past closing prices and the company itself, and then the only relevant columns are **date** and **close**.

Per year there are from 251 to 253 trading days, since the market is closed on weekends and some holidays:

- 2014 had 252 trading days
- 2015 had 252 trading days
- 2016 had 252 trading days
- 2017 had 251 trading days
- 2018 had 251 trading days
- 2019 is expected to have 252 trading days
- 2020 is expected to have 253 trading days

That means that each company in the Dow Jones would have from 251 to 253 historical records per year, except for “DOW” (Dow Inc.) since its records started to appear on 2019-04-02.

For example, in a 5 years period from 2014-04-14 to 2019-04-12, there are 1259 historical records per company, with the exception of “DOW” which has 18. If the historical records of the actual Dow Jones Industrial Average (which ticker symbol is “DIA”) are also included in the dataset they would add another 1259 records. Then for this 5 year period there are 37788 historical stock price records in total ($1259 \times 29 + 18 + 1259 = 37788$).

2.2 Dataset retrieval

The strategy to retrieve the dataset is to get the historical stock price records for each one of the 30 stocks in the Dow Jones (the list of stocks is retrieved from the file `dow_jones_stocks.csv`), plus the ones for the Dow Jones Industrial Average, and then combining them in a single data-frame.

For the historical stock price records only the columns `date` and `close` are picked, and one new column `symbol` is added to indicate the ticker symbol of the company which the record belongs to.

The following code is the implementation of the mechanism to retrieve the dataset:

```
#' Loading the set of ticker symbols of the companies contemplated in
#' the Dow Jones Industrial Average.
dow_jones_stocks <- read.csv('dow_jones_stocks.csv', stringsAsFactors = FALSE)

#' This is the symbol used by the actual average, i.e. the Dow Jones Industrial Average.
djia_symbol <- 'DIA'

#' This is the template to create the URL to extract historical stock prices
#' from the IEX API.
iex_api_url_template <- 'https://api.iextrading.com/1.0/stock/%s/chart/%s'

#' Retrieves the historic prices for a particular stock from the data source.
#'
#' @param ticker_symbol The ticker symbol or symbols to filter the data.
#' @param hist_period The period to retrieve historical records,
#' p.e '5y' for 5 years, '1y' for 1 year, '1m' for 1 month, etc.
#' @return The dataframe containing the historic prices.
get_historical_records <- function(ticker_symbol, hist_period = '5y') {
  # Getting the historic records from IEX into a dataframe
  call_url <- sprintf(iex_api_url_template, ticker_symbol, hist_period)
  resp <- GET(call_url)
  historical_prices <- as.data.frame(fromJSON(content(resp, 'text'), flatten = TRUE))

  # Adding the ticker symbol as a column
  historical_prices['symbol'] = ticker_symbol
  # Converting the date column to a Date type
  historical_prices$date <- as.Date(historical_prices$date, format='%Y-%m-%d')

  # Only choosing the columns: symbol, date, close
  historical_prices %>% select(symbol, date, close)
}

#' Gets a dataframe containing historic prices for stocks in
#' the Dow Jones Industrial Average.
#'
#' @param hist_period The period to retrieve historical records,
#' p.e '5y' for 5 years, '1y' for 1 year, '1m' for 1 month, etc.
#' @return The dataframe containing the historic prices.
get_dow_jones_dataframe <- function(hist_period = '5y') {
  # Getting the historic records of the Dow Jones Industrial Average
  historical_prices <- get_historical_records(djia_symbol, hist_period = hist_period)

  # Retrieves the historic records for each one of the ticker symbols in the
  # Dow Jones Industrial Average
```

```

for (tickers_symbol in dow_jones_stocks$symbol) {
  historical_prices <- rbind(historical_prices,
                             get_historical_records(tickers_symbol,
                                                      hist_period = hist_period))
}

historical_prices
}

```

The IEX's API only allows to get historical records for a maximum of the previous 5 years, however the management of the dataset does not have to be limited by this constraint, since an old dataset can be updated just by adding the missing records. In this way a mechanism can be implemented to keep data beyond 5 years old and keeping a dataset updated.

The following is the implementation of the function in charge of updating a dataset by adding the missing recent records:

```

## Updates a dataframe containing historic prices for stocks in
## the Dow Jones Industrial Average,
## by retrieving the most recent records from the information source.
##
## @param historical_prices The dataframe containing stock price historical records.
## @return The dataframe containing the historic prices.
update_dow_jones_dataframe <- function(historical_prices) {
  # Getting the amount of days that need to be updated
  last_recorded_day = max(historical_prices$date)
  today = Sys.Date()
  days_to_update = difftime(today, last_recorded_day, units="days")

  # Deciding the historic period to request the source according
# to the days that need to be updated
  hist_period = '5y'
  if (days_to_update < 1) {
    return(historical_prices)
  } else if (days_to_update < 28) {
    hist_period = '1m'
  } else if (days_to_update < 365) {
    hist_period = '1y'
  }

  # Getting the data frame containing the missing records
  last_historical_prices <- get_dow_jones_dataframe(hist_period = hist_period)

  # Adding the missing records and removing duplicates
  historical_prices <- rbind(historical_prices, last_historical_prices)
  historical_prices[!duplicated(historical_prices[c('symbol', 'date')]),]
}

```

The following is a variable that specifies the file's name where the dataset is being stored, and which can be periodically updated:

```

## The file's name where the dataset is stored.
dow_jones_dataframe_filename <- 'dow_jones_dataframe.Rda'

```

2.3 Data visualization