Machine Learning Engineer Nanodegree

Capstone Project Proposal- S&P 500 Index price prediction

Akshat Bhardwaj

5th Dec 2018

Proposal

Domain Background

Domain background for this Capstone project is investment banking. The Standard & Poor's 500, often abbreviated as the S&P 500, or just the S&P,is an American stock market index based on the market capitalizations of 500 large companies having common stock listed on the NYSE or NASDAQ. The S&P 500 index components and their weightings are determined by S&P Dow Jones Indices. It differs from other U.S. stock market indices, such as the Dow Jones Industrial Average or the NASDAQ Composite index, because of its diverse constituency and weighting methodology.

This project and topic is relevant for predicting stock and index prices using AI and machine learning algorithms because this is a field where lots of data and experience are funneled and fed to different algorithm based prediction and trading. This project is another try to verify future pricing based on historical data and prediction analysis.

Personal motive for this project is to predict S&P 500 index price and to compare how various machine learning perform in predicting price for time series data based on historical variables like sales, price, PE ratio etc with supervised learning models. Motivation is also to implement supervised learning and Neural Networks on real time dataset to check validity against market value.

I could not find any study directly applicable to the features and dataset I have selected. But below given studies are similar to what I am trying to achieve.

Reference- http://www.diva-portal.org/smash/get/diva2:1213449/FULLTEXT01.pdf https://www.kaggle.com/amarpreetsingh/stock-prediction-lstm-using-keras/notebook

Problem Statement

Problem statement is to predict S&P 500 index price for next month and find short term trend in price movement. This is a **regression problem** based on historical data provided in time-series.

Some surveys show 30% of all stock trading done on NYSE is driven by machine learning and algorithmic models. Problem statement for this project is to verify how accurate price predictions can machine learning make based on historical monthly, quarterly and yearly data available about S&P500 index vs what has been the closing price of index. See the dataset features available in next section.

If machine learning stock trading are to be believed, then which algorithm I have selected below suits best in this given context and dataset.

Datasets and Inputs

Dataset-

I have selected a dataset from Quandl.com where MULTPL provides S&P500 index related attributes for free. This data is split into monthly, quarterly and yearly basis and available through API upto latest date. Dataset is a time-series since the inception of S&P500 index in 1871.

NaN or missing values-

Challenge is to handle NaN values for dates where yield and ratio are not available. Data is available in time-series since inception of S&P500 index along with other attributes about the index which are given below in the table.

Data API connection-

I would be using my personal key from my free account on Quandl data website where MULTPL provides API the following attributes for S&P500 index in time-series. All of the data combined together affects the price of S&P 500 index together with underlying constituent equity data. Underlying equity data is out-of-scope for this project but would be interesting to include for further studies.

Output data - “S&P 500 Real Price by Month” (Target output) prediction.

Input data- 35 features available in the dataset which shows sales, P/E, growth, dividend ratios etc per month, quarter and Annually.

Index name- S&P 500 Index

Asset class- Equities

Characteristics of the dataset-

1. There are 36 data points.
2. Time-series data is split per month, per quarter and per year.
3. Earning yield, price to sales ratios and yeilds are available in time-series.
4. Data available is linear and scaled version for many columns.
5. Columns like S&P 500 Sales by Year, S&P 500 Real Sales by Year have values much higher than other scaled down ratios which should be scaled down all together to avoid any biases.

S&P 500 Dividend Yield by Month	MULTPL/SP500_DIV_YIELD_MONTH
S&P 500 PE Ratio by Month	MULTPL/SP500_PE_RATIO_MONTH
Shiller PE Ratio by Month	MULTPL/SHILLER_PE_RATIO_MONTH
S&P 500 Earnings Yield by Month	MULTPL/SP500_EARNINGS_YIELD_MONTH
S&P 500 Inflation Adjusted by Month	MULTPL/SP500_INFLADJ_MONTH
S&P 500 Price to Sales Ratio by Quarter	MULTPL/SP500_PSR_QUARTER
S&P 500 Dividend by Month	MULTPL/SP500_DIV_MONTH
S&P 500 Dividend by Year	MULTPL/SP500_DIV_YEAR
S&P 500 Dividend Growth by Year	MULTPL/SP500_DIV_GROWTH_YEAR
S&P 500 Dividend Growth by Quarter	MULTPL/SP500_DIV_GROWTH_QUARTER
S&P 500 Price to Book Value by Quarter	MULTPL/SP500_PBV_RATIO_QUARTER
Shiller PE Ratio by Year	MULTPL/SHILLER_PE_RATIO_YEAR
S&P 500 PE Ratio by Year	MULTPL/SP500_PE_RATIO_YEAR
S&P 500 Dividend Yield by Year	MULTPL/SP500_DIV_YIELD_YEAR
S&P 500 Price to Sales Ratio by Year	MULTPL/SP500_PSR_YEAR
S&P 500 Earnings Yield by Year	MULTPL/SP500_EARNINGS_YIELD_YEAR
S&P 500 Price to Book Value by Year	MULTPL/SP500_PBV_RATIO_YEAR

S&P 500 Inflation Adjusted by Year	MULTPL/SP500_INFLADJ_YEAR
S&P 500 Real Price by Month (Target output)	MULTPL/SP500_REAL_PRICE_MONT
S&P 500 Sales by Year	MULTPL/SP500_SALES_YEAR
S&P 500 Sales Growth Rate by Year	MULTPL/SP500_SALES_GROWTH_YEAR
S&P 500 Sales by Quarter	MULTPL/SP500_SALES_QUARTER
S&P 500 Real Sales Growth by Quarter	MULTPL/SP500_REAL_SALES_GROWTH_QUARTER
S&P 500 Sales Growth Rate by Quarter	MULTPL/SP500_SALES_GROWTH_QUARTER
S&P 500 Real Sales Growth by Year	MULTPL/SP500_REAL_SALES_GROWTH_YEAR
S&P 500 Real Earnings Growth by Year	MULTPL/SP500_REAL_EARNINGS_GROWTH_YEAR
S&P 500 Real Sales by Year	MULTPL/SP500_REAL_SALES_YEAR
S&P 500 Real Earnings Growth by Quarter	MULTPL/SP500_REAL_EARNINGS_GROWTH_QUARTER
S&P 500 Earnings Growth Rate by Quarter	MULTPL/SP500_EARNINGS_GROWTH_QUARTER
S&P 500 Real Sales by Quarter	MULTPL/SP500_REAL_SALES_QUARTER
S&P 500 Earnings by Month	MULTPL/SP500_EARNINGS_MONTH
S&P 500 Book Value Per Share by Year	MULTPL/SP500_BVPS_YEAR
S&P 500 Earnings by Year	MULTPL/SP500_EARNINGS_YEAR
S&P 500 Earnings Growth Rate by Year	MULTPL/SP500_EARNINGS_GROWTH_YEAR
S&P 500 Book Value Per Share by Quarter	MULTPL/SP500_BVPS_QUARTER
S&P 500 Real Price by Year	MULTPL/SP500_REAL_PRICE_YEAR
Date	Timeseries since 1871

Reference and definition of data variables-

Description of all data columns can be found on found on the following site- https://www.quandl.com/data/MULTPL-S-P-500-Ratios

Growth Rates

From <https://www.investopedia.com/terms/g/growthrates.asp>

Price/Earnings To Growth - PEG Ratio

From <https://www.investopedia.com/terms/p/pegratio.asp>

S&P 500 Dividend Yield by Month

DESCRIPTION - S&P 500 dividend yield (12 month dividend per share)/price. Yields following September 2018 (including the current yield) are estimated based on 12 month dividends through September 2018, as reported by S&P. Sources: Standard & Poor's for current S&P 500 Dividend Yield. Robert Shiller and his book Irrational Exuberance for historic S&P 500 Dividend Yields.

https://www.quandl.com/data/MULTPL/SP500_DIV_YIELD_MONTH-S-P-500-Dividend-Yield-by-Month

S&P 500 Real Sales by Quarter

DESCRIPTION- Trailing twelve month S&P 500 Sales Per Share (S&P 500 Revenue Per Share) inflation-adjusted September, 2018 constant Source: Standard & Poor's

https://www.quandl.com/data/MULTPL/SP500_REAL_SALES_QUARTER-S-P-500-Real-Sales-by-Quarter

Solution Statement

I would collect the data from APIs and join different columns on time-series. There are 36 data points in time-series. There would be many NaN values after joining data point on for which following methods could be used-

1. Scikit-kit learn Imputation - LOCF’ed (last observation carried forward)
2. Linear Interpolation.

Monthly data is balanced but quarterly and annual data can be considered as unbalanced but using Imputation or Interpolation methods, I can balance the data across time-series.

Once data is prepared, I would run neural network to predict monthly price of the index. There would visuals to show times series and distribution of dataset.

I would use linear regression model, random forest prediction and multi-layer deep learning model for predicting the price. I would compare the results from three approaches using F-score, precision score and price accuracy.

Output data - “S&P 500 Real Price by Month” (Target output) prediction.

Input data- 35 features available in the dataset which shows sales, P/E, growth, dividend ratios etc per month, quarter and Annually.

Benchmark Model

I would benchmark linear regression model and the live market price of S&P 500 index price against the predicted price.I don’t have any example of benchmark model used for this dataset hence i would benchmark results from linear regression for myself and compare with LSTM neural network results.

Evaluation Metrics

I would calculate and compare the following metrics for benchmarking and results-

Evaluation metrics-

1. Mean Square Error (mse)
2. LSTM Loss function - Loss function measures the distance between the LSTM model’s output and desired output during training for expedite learning.
3. LSTM Validation Loss - loss on validation dataset.

Project Design

Data capture and data preparation-

I would download the data using API from Quandl into dataframe. I would join all datasets on date and create a master dataset. Scikit-learn imputation or Linear interpolation can be used to fill in missing values for quarterly and yearly columns. I would use Scikit learn imputer or Linear interpolation to fill up missing values in the time-series dataframe. Dataset would be split chronologically 80% for training and 20% for testing.

Data Analysis-

I would apply StandardScaler() from scikit-learn on the dataset for standardization. Then I would analyze the dataset visually to find correlation between variables using a heatmap.

Data scaling-

I would use StandardScaler() function to scale the dataset in pandas dataframe.

Training and testing dataset for Modelling-

I would split 80%-20% of entire data into training and test datasets in chronological order to avoid look-ahead bias during training and testing. Then i would fit and predict using linear regression model and a neural network. I would also compare the results from linear regression and neural network. I could use Long Short-Term Memory (LSTM) Recurrent Neural Network for price prediction. LSTM- Long Short Term Memory (LSTM) network is a variation of Recurrent Neural Network (RNN). It was invented to solve the vanishing gradient problem created by vanilla RNN. It is claimed that LSTMs are capable of remembering inputs with longer time steps.

Model Evaluation-

Compare Means-Square Error between linear regression and LSTM model. I would also compare predicted price vs actual price of S&P 500 index.

Reference-

Long Short-Term Memory (LSTM)
https://en.wikipedia.org/wiki/Long_short-term_memory

Time-series prediction-

https://en.wikipedia.org/wiki/Time_series

LSTM time-series price prediction research paper -
http://www.diva-portal.org/smash/get/diva2:1213449/FULLTEXT01.pdf

Multivariate Time Series Forecasting with LSTMs in Keras

https://machinelearningmastery.com/multivariate-time-series-forecasting-lstms-keras/

How to Make Predictions with Long Short-Term Memory Models in Keras

https://machinelearningmastery.com/make-predictions-long-short-term-memory-models-keras/