MODELLING AND FORECASTING SHORT-TERM BITCOIN PRICES USING MACHINE LEARNING METHODS

Final Report

Information Technology Capstone Project

COMP5703

Group Members

1. Nathan Simonson (470519459)
2. Jing Shan (460247290)
3. Chenghao Qian (460402280)
4. Libai Sun (460140603)
5. Junqi Wang (450463619)

Abstract

This project's goal is interested in understanding bitcoin price fluctuations over time, and build a series of prediction models to forecast future price movements. Our model development is then supplemented by building a series of interactive visualisation tools highlighting bitcoin behaviour as well as our model performances. We use APIs to collect AUD bitcoin data from Cryptocompare which provides summary data on open, high, low and close AUD prices, and then extract relevant features to understand the time-series movement of bitcoin prices. This data is summarised on a daily basis. In our model our response variable chosen is the bitcoin close price. Through this we build four models LSTM, random forest, extreme gradient boosting and ARIMA, and compare their overall performances with one another. This allows us to apply traditional, current and advanced machine learning techniques in predicting future bitcoin prices. The purpose is to identify which model most accurately represents future bitcoin price movements by evaluating their respective prediction results against the actual bitcoin prices. From our research, we conduct the analysis aware time-series forecasting using the linearity model ARIMA will likely underperform given the high volatility of bitcoin prices. Following the successful implementation of our models, our final component is to build a series of interactive dashboards to assist users in understanding bitcoin price movements and how our prediction models perform compared to actual market.

Table of Contents

1. Introduction

Bitcoin is a cryptocurrency, which utilises decentralisation of currencies without a central bank of administrator. This enables peer-to-peer transactions without the need for any intermediaries (Y. Liu, 2017). Bitcoins are obtained either as a reward for verifying the transactions, or trading over a cryptocurrency exchange. As of 2018, there are approximately 1,500 crypto-assets in circulation with Bitcoin being the most widely traded across all crypto-markets. There are currently 16.8 million bitcoins traded with a cap limit of 21 million (C. Reynard, 2018). Moreover, in 2017 the bitcoin price soared as high as $25,000 AUD in December 2017 before dropping to below $10,000 AUD in April 2018 demonstrating its high volatility (CoinGecko, 2018).

Given this high volatility, predicting bitcoin price is not similar to developing a standard time-series prediction model, unlike forecasting sales where seasonality is present (C. Chatfield, 1988). This presents an opportunity to develop machine learning for predicting future bitcoin price, and evaluate which method produces the most accurate results.

For our experiment we obtain AUD bitcoin prices using Cryptocompare which provides daily open, high, low and close prices. We then build four machine learning models long short-term memory (LSTM), random forest, extreme gradient boosting and auto-regressive integrated moving average (ARIMA). We then use our models to predict a 7-day sequence of future results and compare which method produces the best output. Furthermore, we will develop interactive dashboards relating to the price movement of bitcoin and demonstrate how our model's prediction outputs compare to actual future bitcoin prices.

2. Related Literature

Many studies have been conducted for developing machine learning models to predict bitcoin prices, from linear regression models, support vector regression, time-series models, as well as deep learning models. Each model is tested in efforts to determine which demonstrates the best performance. Several studies have also tested the effectiveness of decision tree methods in predicting time-series forecasts.

S. Mehrmolaei and M. Keyvanpour (S. Mehrmolaei, 2016) conducted experiments across two time-series datasets, New York city births and US immigration time-series, and evaluated performance between standard ARIMA with best autoregressive, differencing and moving average configuration, against improved model which applies mean of estimation error in forecasting process. Results demonstrated standard ARIMA predicting reliable results from low root mean square error (RMSE) and mean average error (MAE), whilst improvements to ARIMA model yielded even lower RMSE and MAE results. Experiment demonstrates applicability of ARIMA for time-series data modelling and forecasts.

T. Guo and N. Antulov-Fantulin (T. Guo, 2018) conducted an alternative experiment approach whereby multiple regression models were developed and tested against one-another, with models ranging from standard and transformations of ARIMA time-series models, to ensemble decision based models such as random forest and gradient boosting. Data was collected from OKCoin exchange and model was developed from hourly bitcoin trades. Overall results showed ensemble trees produced least error levels, with random forest and extreme gradient boosting exhibiting RMSE results of 0.082 and 0.076 respectively, compared against ARIMA which had RMSE result of 0.109.

L. Kirichenko et. al. (2017) developed experiments on testing the effectiveness of several machine learning algorithms for predicting time series of e-commerce tasks. Experiments tested the effectiveness of both decision trees and LSTM in predicting daily conversion rates for an online store. Data provided included number of clicks, number of sales and corresponding conversion rates. Information about which language the customer used as well as country of origin was also captured. Results demonstrated that decision trees were the fastest to train and most interpretable, however did not predict well given the data had many characteristics. Alternatively LSTM demonstrated great predictive robustness, albeit was cumbersome to train and interpret. Comparison of MSE between both models were 53 for decision trees and 22 for LSTM.

ARIMA however relies on linear assumptions in the data, whereas bitcoin is highly non-linear in nature. S. McNally (S. McNally, 2018), whilst main focus was on RNNs and LSTMs, conducted an experiment comparing predictive performance of ARIMA models and found these produced the least accurate results. This highlights the need to use more sophisticated techniques for predicting bitcoin prices. Experiment designs the problem as a classification task on predicting an increase, decrease or no movement in sequential bitcoin prices. Overall results show LSTM provides the best performance, although accuracy is 52.78%. Further testing of LSTM on financial stock markets was conducted by S. Liu et al. (S. Liu, 2018) which shows how configuring the right hidden layer structure can return much more accurate results. Experiments showed improved accuracy when changing hidden layer of LSTM model from one layer to three layers.

Experiments conducted by Geourgoula et al. (I. Geourgoula, 2015) investigated the important attributes predicting bitcoin price behaviour whilst also implementing sentiment analysis using support vector machines (SVM). Findings showed the volume of Wikipedia views, as well as bitcoin hash rate resulted in a positive correlation to bitcoin price. Matta et al. (M. Matta, 2015) also researched relationships between bitcoin price and Google Trends. Results however only showed a weak correlation.

In conclusion, previous experiments have demonstrated that although ARIMA models are useful for time-series modelling, its application in bitcoin price movement is limited and there is a greater likelihood of building a more robust model using alternative techniques. Decision tree based algorithms appears to demonstrate some robustness to handling time series forecasts, in particular from T. Guo et. al. experiments this was shown to produce more accurate results compared to ARIMA. Previous research however strongly suggests LSTM is favourable in this instance as its memory-forget mechanism of bitcoin price movements is better handled by more complex models.

3. Research/Project Problems

3.1 Research/Project Aims & Objectives

This project aims to predict the future price of Bitcoin is Australian market using historical daily price. Four models will be used for prediction which includes ARIMA, LSTM, random forest and extreme gradient boosting. The project will provide a comparison of these models of the accuracy and give analysis.

Also, a dashboard will provide the visualisation of historical price and the final result. It will make the prediction result understandable.

3.2 Research/Project Questions

What dataset will be used? How many data will be chosen as training data?
- The amount of data is depend on the length of training data period and the frequency of price such as every minute, daily or monthly.

What prediction model will be used? What level of accuracy will be?
- Different models have different advantages and many factors will influence the accuracy of the result.

3.3 Research/Project Scope

The project has two parts, visualisation and prediction. In terms of the first part, first task is drawing those visualisation graphs and the second step is making them interact with users.

In the prediction part, the models will predict one combination of training set and prediction period. After that, more combination will be tested and choose the best one from each model to do the final comparison.

4. Methodologies

4.1 Methods

For the bitcoin exchange market, we found it quite similar to the stock exchange market. The useful statistic data related to time series data. There are four models we chose to utilize. Those models are ARIMA, random forest, extreme gradient boosting and LSTM.

LSTM

LSTM (Long Short-Term Memory) is a long-term and short-term memory network. It is a time recurrent neural network suitable for processing and predicting important events with relatively long intervals and delays in time series (Sundermeyer,Martin Schlüter, Ralf Ney, Hermann, 2012). The difference between LSTM and RNN is that it adds a “processor” to the algorithm to judge whether the information is useful or not. The structure of this processor is called the cell.

Three cells are placed in a cell, called an input gate, a forgetting gate and an output gate. A message enters the LSTM network and can be judged according to the rules (R. Rojas, 1996). Only information that complies with the algorithm's certification will be retained, and information that does not match will be forgotten through the forgotten gate (F.A. Gers, J. Schmidhuber, F. Cummins, 2000). It is nothing more than the working principle of one in and two out, but it can solve the long-standing problems in the neural network under repeated operations. It has been proved that LSTM is an effective technique for solving the problem of long-order dependency, and the universality of this technology is very high, resulting in a lot of possibilities (Donahue,Darrell,T, 2015). Researchers have proposed their own variable versions based on LSTM, which allows LSTM to handle ever-changing vertical problems.

ARIMA

Autoregressive integrated moving average (ARIMA) is a model which treats a data sequence formed by the predicted object over time as a random sequence, and uses a mathematical model to approximate the sequence (C. Chatfield, 1996). ARIMA, also denoted as ARIMA(d,p,q), has three variables d, p and q which alter the state of the model. Each variable represents the following:

- p - Represents the lags (lags) of the time series data used in the prediction model, also called auto-regressive (AR) (G.E.P. Box, G. Jenkins, 1970).
- d - Represents the time series data being of several-order differential, which is stable. This is called the integrated item (I) (G.E.P. Box, G. Jenkins, 1970).
- q - Represents the hysteresis (lags) of the prediction error used in the prediction model, also known as the moving average (MA) (G.E.P. Box, G. Jenkins, 1970).

Once this model is identified, it can predict future values from past and present values of the time series (C. De Groot, D. Wurtz, 1999). Modern statistical methods and econometric models have been able to help companies predict the future to some extent.

Random Forest

Decision trees are a method which can be used for both classification and regression tasks (Bishop, 2012), whereby a tree of nodes is grown with each node representing an attribute from the independent variable set. These nodes divide the predictor space into J distinct and mutually exclusive region $R_1,...,R_J$, whereby each of the regions are defined through binary splits and completed recursively. For each of R_J, in regression models the prediction is the mean of the independent variables in each respective region (Givens, 2012).

For regression models decision trees seek to minimise the function:

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$

Random forest extends on decision trees by building an ensemble (or forest) of decision trees, however each decision tree in the forest only selects a subset of the total independent attributes. For regression, the prediction is then an average of all the mean predictions from each decision tree in the forest, and the average of all the votes per class for classification. This process is known as bagging, or bootstrap aggregating, and can produce models to have very good predictive accuracy (Given, 2012).

Extreme Gradient Boosting

Gradient boosting is a greedy algorithm, whereby the algorithm produces a prediction model in the form of an ensemble of weak learners and combines them into a strong learner. This can be used for solving classification and regression problems. It works by sequentially applying the weak learners to weighted versions of the data. Here more weight is given to those data which were misclassified in previous iterations (Murphy, 2012).

The algorithm works by utilising the following method:

- Define some loss function function to minimise. Example for a regression problems is $\frac{1}{n}\Sigma_{i=1}(\hat{y}_i - y_i)^2$, where $\hat{y} = F(x)$ is the prediction model and i is an index over some training set.
- As gradient boosting assumes K iterations, then for $1 <= k <= K - 1$ we assume that each model $F_k(x)$ can be improved by constructing a new model $F_{m+1}(x) = F_m(x) + h(x)$ whereby $h(x)$ is an estimator to the model.
- Estimator is typically interpreted in the following:
  $F_{m+1}(x) = F_m(x) + h(x) = y$
  $=> h(x) = y - F_m(x)$
  The interpretation here is that as the sequential iterations are executed where the form $y - F_m(x)$ reflect the residuals, but can also be viewed as the negative gradients from the function $\frac{1}{2}(y - F(x))^2$ (Friedman, 1999).

Hence the gradient boosting algorithm is also a gradient descent algorithm defined by some loss function (Friedman, 1999). These loss functions can be one of squared error, absolute error, exponential loss or logloss (Murphy, 2012).

4.2 Data visualisation

4.2.1 Visualisation Aims

In this project, it is aimed to visualise the historical bitcoin data and prediction results, and then create a blog dashboard to present and explain bitcoin prediction modelling. This project is to visualize the following objectives:

1. To visualise bitcoin related data, such as price, volume, hash rate, cost per transaction and difficulty.

2. To visualise the modelling results, such as predicted results, comparison chart etc.
3. To create a blog dashboard to generate all findings and analysis of this project.
4. To achieve interaction effects in each chart and dashboard.

4.2.2 Data Analysis

Bitcoin price is time series data which shows the characteristics of time serious. Time series is a series of data points indexed using time order (Brillinger, 1981). Each data point is close together in time and they are dependent of their adjacent values. Bitcoin price is time dependent data and volatility varies over time. Furthermore, trend should be considered when doing time series analysis. As a result, visualisation method should attach importance to showing the trend and time-dependent of this dataset. According to Tominski and Aigner (2015), there are two kinds of chart to visualise time series, including overlapping charts and separated charts.

4.2.3 Visualisation Tool Comparison and Selection

There are a great number of visualisation tools, such as Echarts, Tableau, D3, Chart.js, Plotly, etc. We made a comparison between these tools and found Plotly is more simple and suitable for our project. The comparison table is presented as below in Figure 1.

Criteria	Plotly	Tableau	Matlabplot	D3.js
Implementation Speed	Good	Good	Good	Good
Scalability	Good	Very Good	Excellent	Excellent
Pricing	99 dollars per year for students	Free one year for students; 35 dollars per month.	Free	Free
Import file	csv, excel, SQL, URL	txt, csv, tab, tsv, SQL	all data	csv, excel, SQL, URL, json, text,vtml, Tsv
Interactive Visualization	Good	Good	Good	Very Good
Dashboard Support	Good	Good	-	-
Development Environment	Plotly.py, Plotly.R and Plotly.js	R, python	python	Javascript
Presentation	Good	Good		
operability	Very Good	Good	Just so so	Just so so
Export file	html, Matlab.py, image, json, R, Plotly.js, Node.js	image	image	d3.js

Generally speaking, Matlablib and D3.js are powerful to draw any charts with code, but it is not friendly to users who are poor in programming and cannot make a dashboard or presentation directly. Tableau outperforms other tools in some functions, especially mobile support and responsive dashboard, handle large amounts of data and easy to implementation. However, Tableau cannot custom existing formatting in Tableau, but Plotly can edit graph using codes. Plotly offers a lot of competitive advantages. First of all, various types of chart templates are provided in Plotly, such as scatter, bar, line, area, heatmap, table, contour, pie, 3D chart, maps, finance charts, distributions charts, etc. For our project, finance charts are needed in visualisation of bitcoin price. Secondly, the Plotly's function of exporting source code based on Plotly.py, Plotly.R and Plotly.js and exporting charts as URL contributes a flexible and powerful visualisation, presentation and application. Furthermore, team collaboration, sharing link and embedding plot are user-friendly to developers and simple users.

Therefore, Plotly is selected to do main part of visualisation of our project. Matplotlib library will be used in LSTM modelling for visualisation in python, while highchart and ggplot2 will be used in R.

4.2.5 Visualisation Methods

1) Candlestick chart

   Bitcoin (□) is one kind of cryptocurrencies, a form of electronic cash, which is widely traded in the world. Bitcoin price is quite similar to financial data, such as foreign currency and stock price. As a result, visualisation of bitcoin price can borrow the experiences of financial data’s visualisation. We created an interactive candlestick chart to show bitcoin price with high, low, open, close and volume. Besides, we followed the bitcoin’s rule and drew the increasing line in green and decreasing line in red, which is opposite to stock price. In the meanwhile, trade volume is also draw into same legend, which can show the relationship between price and volume. There are many buttons in chart to achieve interaction, such as 1m, 3m, 6m, 1y and Max. Furthermore, when your mouse pointer hovers over the line, the exact data could appear.

   The candlestick chart of bitcoin price is shown as below.

   Figure 1 Bitcoin price

2) Line charts

   Line chart is mostly used to visualise time series data. In this project we visualise the bitcoin hash rate, difficulty, cost per transaction and predicted results using interactive line charts.

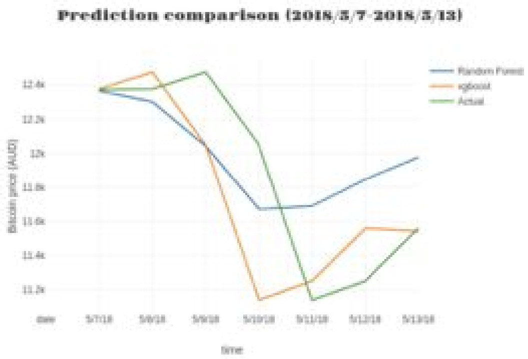


Figure 2 prediction comparison

3) Dashboard

   Dashboard will be created to explain the results using visualization and further discussion. Dash provides dashboard template which is easy to operate. We create multiple buttons including Homepage, Historical data, ARIMA, Random Forest, xgboost, LSTM, Comparison and Team.

5. Resources

Various softwares tools listed as below are applied to support the visualization and prediction of this project.

5.1 Hardware & Software

S No	Type	Software & Version	Comment
1	Operating System	Windows, Mac OS	These two operating systems support this task's coding and visualization.

2	Programming Language	Python 3.6 R	Python is widely used programming language for the task of Data Science. Furthermore, R might be used to visualize.
3	Coding Platform	Jupyter Notebook, PyCharm 2018.1 RStudio	These three softwares support us to use python and R language to code.
4	Management & Communication Tools	Slack, MS Project	Slack is a professional communication tool to provide a online discussion among teammates. MS project helps us to establish the project plan.

5	Reporting Tools	Office 365	Office 365 provides powerful office softwares, which supports our writing, email, simple graph drawing and so on.
6	Visualization Tools	Python(Matplotlib) Plotly	These softwares provide interactive visualisation for analysis and predicted results through dashboard.

5.2 Materials

Except above softwares, we also use some other resources to support our project.

No.	Name	Application

1	Machine learning frameworw：Tensorflow （https://www.tensorflow.org/）	TensorFlow is an open-source software library for dataflow programming, which supports our prediction modelling. We learn how to build prediction model with the help of TensorFlow.
2	Machine learning： scikit learn （https://scikit-learn.org/stable/）	Scikit learn is an open source software to help preprocess data, build machine learning models and optimize models in our project.

3	Software developing platform: Github	Github brings us together to share and code. Furthermore, we also learn some good examples sharing on Github.
4	Online community to learn and share programming knowledges: Stack Overflow	Stack Overflow is a website which can help us get some answers when we have confusion on programming or other IT related questions.

5.3 Roles & Responsibilities

This is a group cooperation project, which needs all team members‘ contribution.

All of the team members are responsible for background research and proposal and report writing. Furthermore, everybody has his/her own duty and should be mainly responsible for different tasks in this project.

Name	Role	Responsibilities

NATHAN SIMONSON	Leader	Lead all works for achieving this project's goal; Provide direction, instructions and guidance to a group of individuals; Communicate with team members and monitor team members. Build random forest and xgboost machine learning models and optimize models. Contribute to addressing issues of other members.

JING SHAN	Data Visualiser	Mainly responsible for all visualization tasks; Visualise the data for data analysis and the comparison data between prediction results and actual results; Build an interactive dashboard to show the final prediction works.

CHENGHAO QIAN	Troubleshooter	Mainly responsible for solving problems and difficulties during this project; Build the ARIMA model and optimize the model.
LIBAI SUN	Data Scientist/Dev	Mainly responsible for prediction modelling; Collect and clean the data; Build the LSTM model and optimize the model;

JUNQI WANG	Meeting Minutes Taker	Arrange weekly meeting and inform everyone; Taking notes during the meeting about the decisions and discussion of this project; Remind the task progress.

6. Milestones / Schedule

Milestone	Tasks	Reporting	Date
Week-1	Background research	None	30/07/2018-01/08/2018
Week-2	Dataset collection	The dataset that contains a large amount of data.	01/08/2018 - 09/08/2018
Week-3	Data preprocessing	Transfer the data into a useful format.	08/08/2019 - 15/08/2018
Week-4	Visualisation	Visualization drafts.	15/08/2018 - 22/08/2018
Week-5	Write proposal report	Proposal Report	22/08/2018 - 31/08/2018
Week-6	Dashboard design and model selection	Background research and selected models	31/08/2018 – 05/09/2018

Week-7	Build prediction model	Draft prediction model.	05/09/2018 – 12/09/2018
Week-8	Build prediction model	Draft prediction model.	12/09/2018 – 19/09/2018
Week-9	Write progress report	Progress Report	19/09/2018 – 26/09/2018
Week-10	Optimization of models	Optimized models	26/09/2018 – 10/10/2018
Week-11	Further improvement of models and output results for comparison	Optimized models and comparison of models	10/10/2018 – 17/10/2018
Week-12	Make presentation slides	Presentation draft	17/10/2018 – 24/10/2018
Week-13	Prepare presentation	Presentation slides	24/10/2018 – 31/10/2018
Week-14	Evaluate and discuss models and write final report	Final report	1/11/2018 – 7/11/2018
Week-15	Write final report and edit report	Final report	7/11/2018 – 14/11/2018

7. Results

Feature Engineering

Original data collected from Cryptocompare provides daily summary data of open, low, high and close bitcoin prices. The volumes from and to are also provided. A sample of 10 observations provided as such.

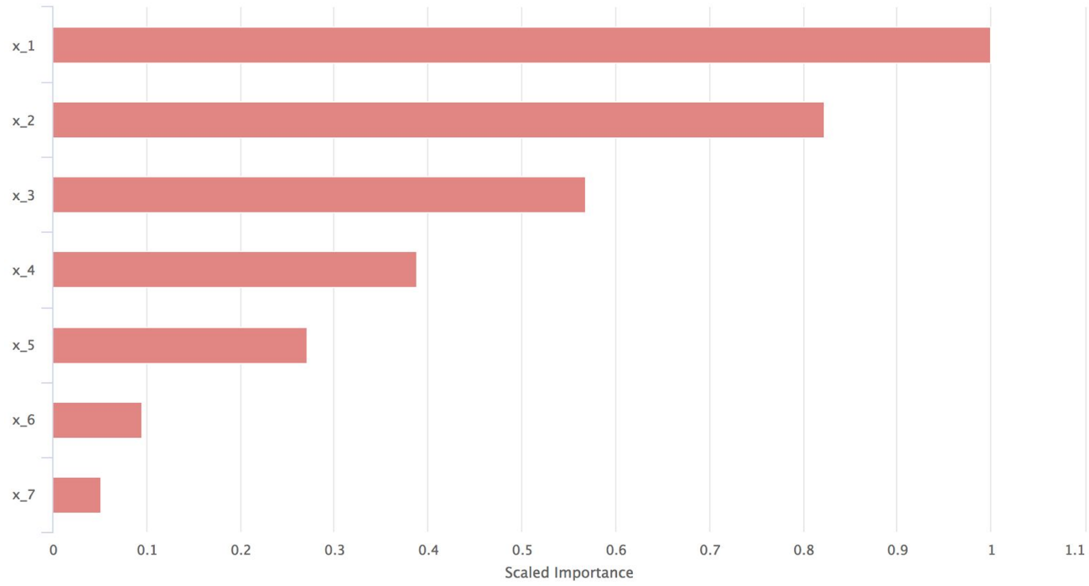
	date	datetime	time	open	high	low	close	volumefrom	volumeto
1	2013-03-30	2013-03-30T00:00:00Z	1364601600	101.05	102.95	96.64	100.25	16.75	1678.68
2	2013-03-31	2013-03-31T00:00:00Z	1364688000	100.25	118.3	69.67	99.99	8.8	837.63
3	2013-04-01	2013-04-01T00:00:00Z	1364774400	120.54	132.31	83.82	110.86	27.43	2820.23
4	2013-04-02	2013-04-02T00:00:00Z	1364860800	110.86	120.54	91.23	119.53	115.79	13194.41
5	2013-04-03	2013-04-03T00:00:00Z	1364947200	103	157.25	95.86	157.08	45	5675.19
6	2013-04-04	2013-04-04T00:00:00Z	1365033600	121.4	157.25	117.22	135.52	67.2	8529.93
7	2013-04-05	2013-04-05T00:00:00Z	1365120000	145.73	156.95	135.52	156.95	25.03	3631.02
8	2013-04-06	2013-04-06T00:00:00Z	1365206400	157.46	170.98	149.26	163.8	17.68	2866.99
9	2013-04-07	2013-04-07T00:00:00Z	1365292800	163.8	176.25	139.23	139.23	14.53	2314.12
10	2013-04-08	2013-04-08T00:00:00Z	1365379200	154.56	201.85	120.08	188.89	35.49	5642.4

Our objective is to predict daily future bitcoin price data, and so we only leverage data from the dates as well as the close price. From this we conduct feature engineering such that we capture the previous 7 days of bitcoin prices against the current bitcoin price as at specific daily points. A screenshot of the feature engineering is screenshot below:

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	y
1	11188.16	11658.49	11439	11597.1	11696.14	11290.05	10646.88	11039.19
2	11039.19	11188.16	11658.49	11439	11597.1	11696.14	11290.05	10544.26
3	10544.26	11039.19	11188.16	11658.49	11439	11597.1	11696.14	10148.61
4	10148.61	10544.26	11039.19	11188.16	11658.49	11439	11597.1	10319.21
5	10319.21	10148.61	10544.26	11039.19	11188.16	11658.49	11439	9204.67
6	9204.67	10319.21	10148.61	10544.26	11039.19	11188.16	11658.49	8934.94
7	8934.94	9204.67	10319.21	10148.61	10544.26	11039.19	11188.16	9062
8	9062	8934.94	9204.67	10319.21	10148.61	10544.26	11039.19	8835.98
9	8835.98	9062	8934.94	9204.67	10319.21	10148.61	10544.26	9090.31
10	9090.31	8835.98	9062	8934.94	9204.67	10319.21	10148.61	9581.43

where y reflects the current price and x_i refers to previous daily prices from i days ago.

Using random forest, we conduct feature importance to evaluate which features have the most influence on determining the current bitcoin price. Our results were as follows:



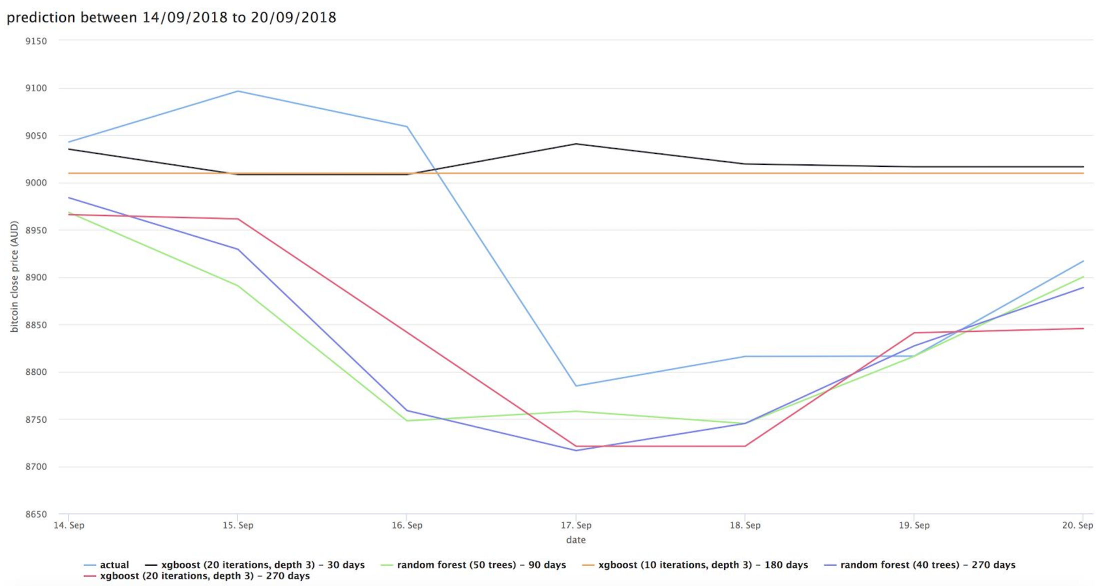
Our results show that of the features used, data from the prior day is the most important attribute towards predicting the current bitcoin price.

Random Forest and Extreme Gradient Boosting

For each model developed, our performance results are evaluated against one another by comparing RMSE, model training time and overall visualisation comparisons. This last element is important to understand where the model is making errors.

Train Data Size	Model (parameters)	RMSE	Training Time
30 days	random forest (30 trees)	708.1355	0.016s
30 days	random forest (40 trees)	638.9019	0.018s
30 days	random forest (50 trees)	709.0293	0.018s
30 days	xgboost (5 iterations, depth 3)	1068.4245	0.045s
30 days	xgboost (10 iterations, depth 3)	632.2774	0.109s
30 days	xgboost (20 iterations, depth 3)	554.1721	0.140s
90 days	random forest (30 trees)	753.2668	0.039s
90 days	random forest (40 trees)	780.2118	0.041s
90 days	random forest (50	554.6966	0.046s

	trees)		
90 days	xgboost (5 iterations, depth 3)	995.1026	0.067s
90 days	xgboost (10 iterations, depth 3)	896.9155	0.100s
90 days	xgboost (20 iterations, depth 3)	783.3572	0.155s
180 days	random forest (30 trees)	467.0923	0.075s
180 days	random forest (40 trees)	485.9752	0.077s
180 days	random forest (50 trees)	508.8293	0.087s
180 days	xgboost (5 iterations, depth 3)	902.2502	0.056s
180 days	xgboost (10 iterations, depth 3)	462.5072	0.097s
180 days	xgboost (20 iterations, depth 3)	635.8967	0.194s
270 days	random forest (30 trees)	591.0819	0.120s
270 days	random forest (40 trees)	501.7514	0.113s
270 days	random forest (50 trees)	630.1807	0.140s
270 days	xgboost (5 iterations, depth 3)	824.8360	0.071s
270 days	xgboost (10 iterations, depth 3)	454.6041	0.102s
270 days	xgboost (20 iterations, depth 3)	412.4730	0.206s



Our results demonstrate that ensemble tree models with smaller training sizes don't perform as well as those with larger training sizes. This was demonstrated by the decrease in RMSE results. We also observe that each model was trained in a reasonably short time period, regardless of training size or model complexity. Overall the most effective method was XGBoost with parameters 20 iterations and depth 3, using 270 days of training data, and from visual chart demonstrates to predict same curve behaviour as actual bitcoin close price.

From our visualisation we notice the following observations:

- Smaller training times tend to produce results which don't reflect any trend, as was evident from xgboost (20 iterations, depth 3) using 30 days training data.
- More complex models with larger training periods tend to predict the future trends very well. This was demonstrated by both random forest (40 trees) and xgboost (20 iterations, depth 3) using 270 days of training days.

LSTM

Four types of LSTM models are built to predict Bitcoin price over 7 days based on estimated date, 18/Jun/2018, 18/Jul/2018, 18/Aug/2018, 24/Aug/2018 and 14/Sep/2018.

The first LSTM model is to use 90 days, 120 days and 270 days separately as train data to predict 7 days Bitcoin price since the dates mentioned before. The model parameters are 20 neurons, 50 epoch, batch_size equals to 4, Mean Absolute Error as loss function, adma as optimizer and normalized data as train data.

For this model which predicts single points data based on known data, the RMSE between true test data and predicted test data shows as figure 1 below.

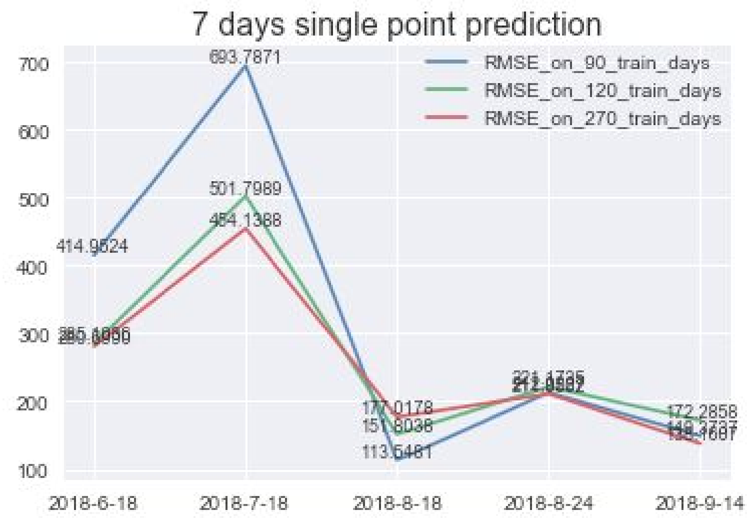


figure 1. RMSE between true data and predicted data

From the figure 1, one conclusion can be proved that the more days get trained, the RMSE will be smaller, the model will be more accurate. On the other hand, when the data is rather stationary, the RMSE does not have too much difference over different training days.

The second LSTM model is to use two input dimensions and add a new axis as predicted price to predict next predicted price t+1, feeding the current prediction back into the window to make prediction at the next time step, which technique also known as moving-forward window.

So long as we convert input variables to 3D vector form, LSTM could be built. The model parameters are the same as the model above, the results of sequence price prediction is shown as below. The figure on the left hand side is the change percentage of true data and predicted data, on the other hand, the figure on the right is the real price of true data and predicted data.

When the window size equals to 90 and prediction days equal to 7, the predicted percentage volatiles more dramatically but the difference between real price and predicted price is quite larger.

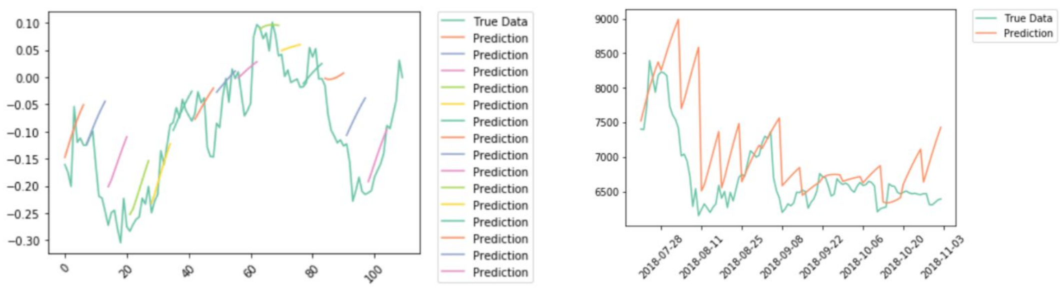


figure 2. Percentage and price change when window size = 90 days

The RMSE from 06-18-18 to 06-24-18 is around 726, which is almost twice of the RMSE, 414, in the same time period in the first model.

When the window size equals to 120 days and prediction days equal to 7 days, the change over time shows as figures below.

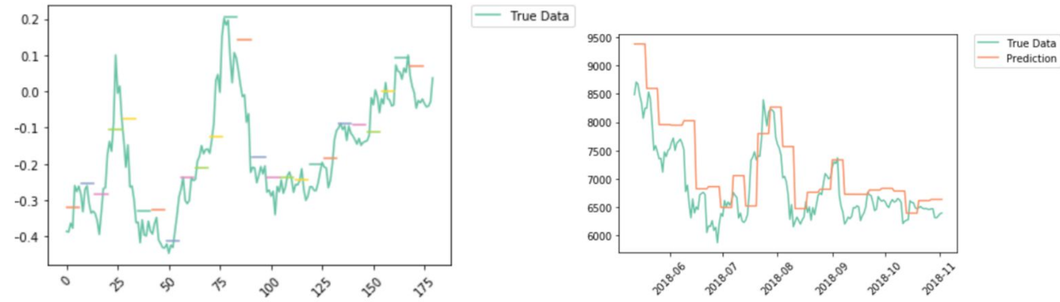


figure 3. Percentage and price change when window size = 120 days

When the window size equals to 270 days and prediction days equal to 7 days, the change over time shows as figures below.

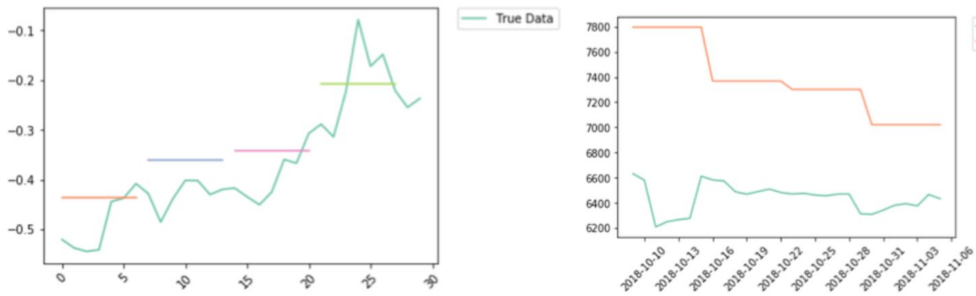


figure 4. Percentage and price change when window size = 270 days

For this model which predicts sequence data based on known and predicted data, the RMSE between true test data and predicted test data shows as figure 5 below. As for the moving-forward window, the comparison is shown as below, the less windows are created to predict, the smaller RMSE and the more accurate the model will be.

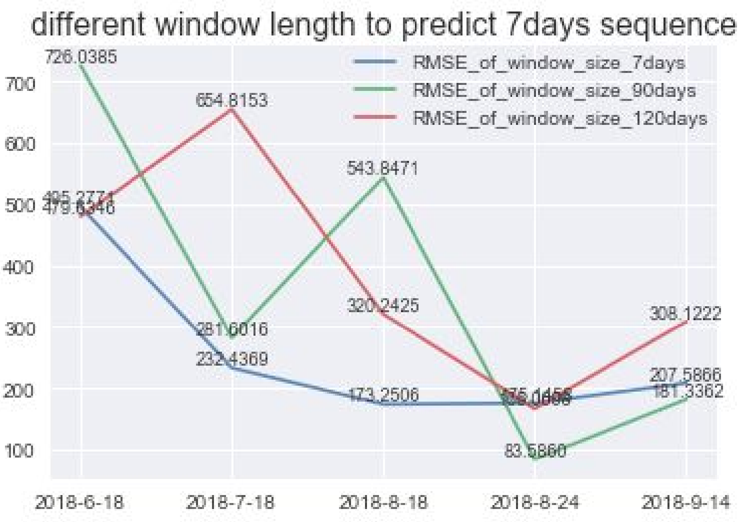


figure 5. six dimensions input, RMSE between true data and predicted sequence data

The RMSE in second model is higher than the first model and the reason is understandable, the longer the predictions are, the more volatile will be between data in different dates.

The third model is to add more features and do the training data and test data, which adds a great benefit in time series forecasting, where traditional machine learning methods can be difficult to adapt to multivariate or multiple input forecasting problems.

The testing data is 15%, the window_size (sequence length is 14 days), the prediction sequence is 14 days, the input of LSTM model is two dimensions, including average price and average volume after eliminating multicollinearity.

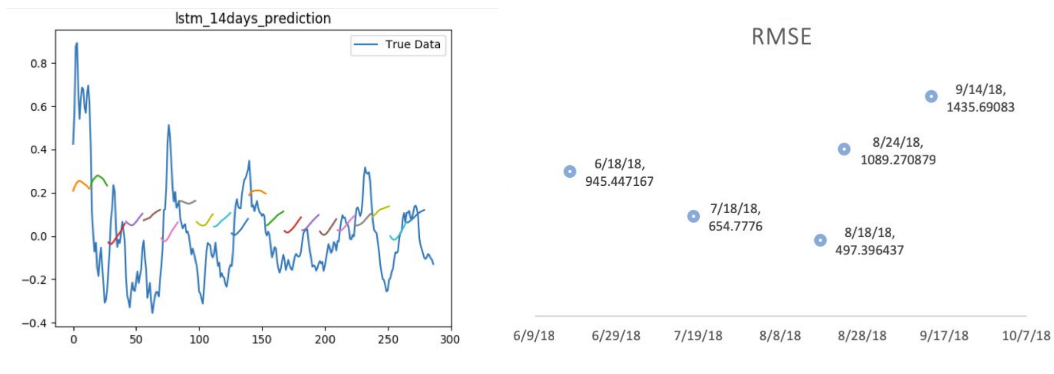


figure 6. RMSE over 14 days, input = average price and average volume

The parameters of the forth model is testing data with 15%, the window_size (sequence length is 14 days), the prediction sequence is 14 days, the input of LSTM model is six dimensions, including close price, open price, high price, low price, volumefrom and volumeto.

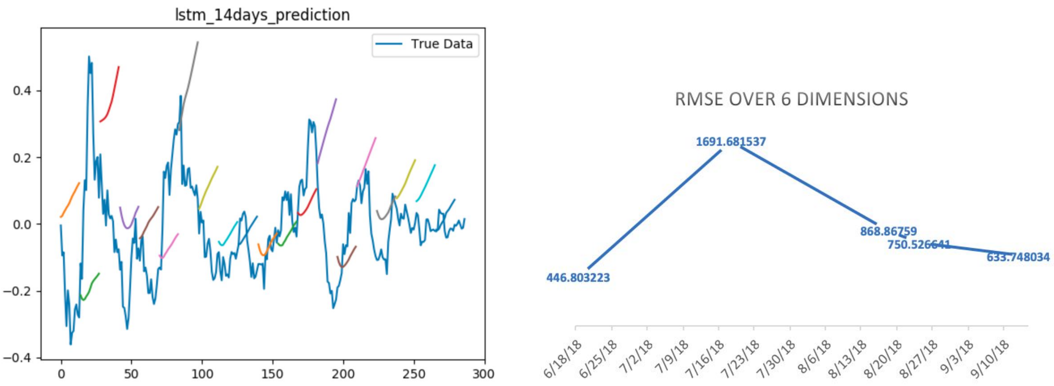


figure 7. RMSE over 14 days, input is 6 dimensions

According to the outcome shows in figure 8, the model with more features is not superior to the model with less features. Even though the forth model has more features, they are expressing the similar information as the third model only with two features. In the third model, the features are price and volume values after eliminating multicollinearity, and the features are four kinds of price and two kinds of volume in the forth model. The performance of these two models does not have a big difference, which proves that deep learning model is robust to the data with multicollinearity property.

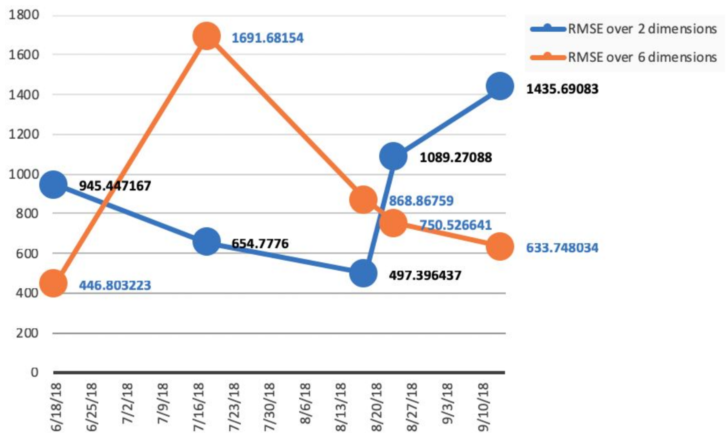


figure 8. Forteen days, RMSE comparison over different feature dimensions

The other model with the same parameters but the window_size (sequence length is 7 days) and the prediction sequence is 7 days, the real and predicted percentage changes shows below.s

The real and predicted percentage changes of six dimensions show below.

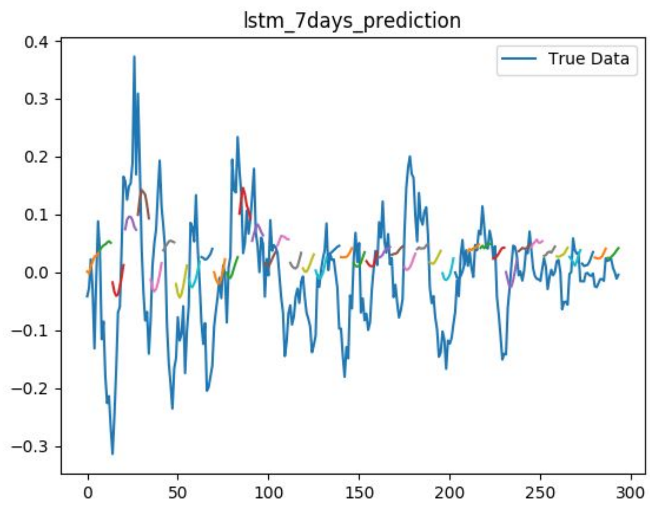


figure 10. RMSE over 7 days, input is 6 dimensions

According to the outcome shows in figure 11, the model with more features is obviously superior than the model with less features, which proves that deep learning model is robust to the data with multicollinearity property and do not have to eliminate multicollinearity before building the model.

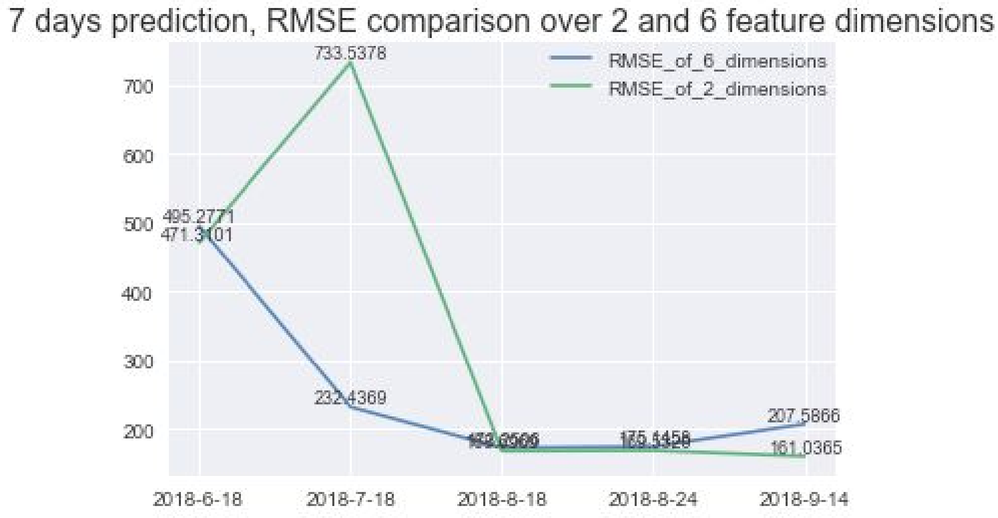


figure 11. seven days, RMSE comparison over different feature dimensions

The last model is to build a more complicated model, including more layers and more neurons. The model will have three layers, including one input layer, two hidden layers and one output layer. For the parameters of

The structure is (2,50,100,1) and input data is two dimensions including non-multicollinearity price and volume.

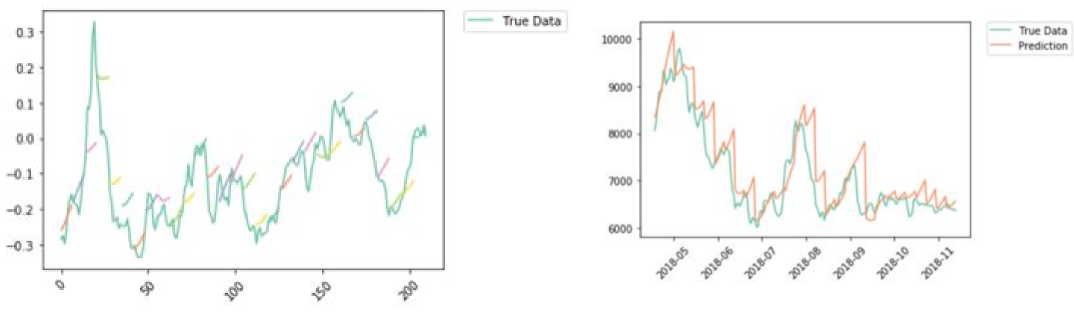


figure 12. (2,50,100,1) and input data is two dimensions

The other structure is to use six dimensions input, including close, open, high and low price to train into stateful LSTM layers as (6,50,100,1), the prediction figures are shown as below.

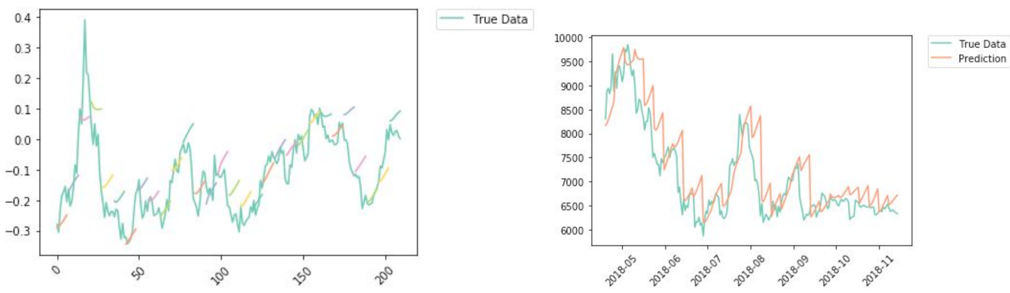


figure 13. (6,50,100,1) and input data is two dimensions

The comparable structure is to input 6 dimensions and same stateful LSTM model but the layers as (1,50,100,1), the prediction figures are shown as below.

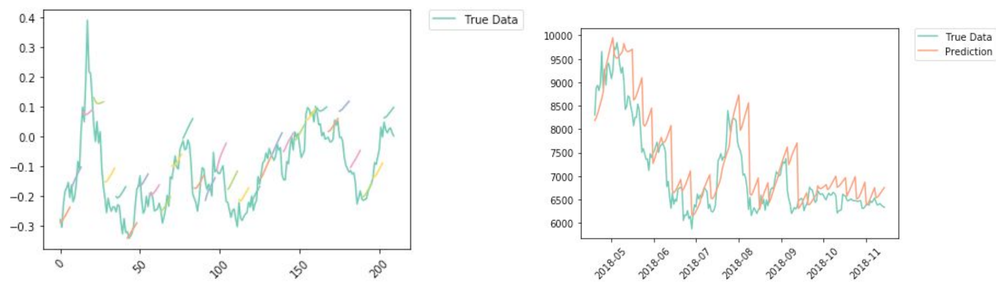


figure 14. (1,50,100,1) and input data is six dimensions

As for the comparison of these different parameters combination, the outcome shows as below, there is not much difference towards accuracy performance between different neurons set in layer one.

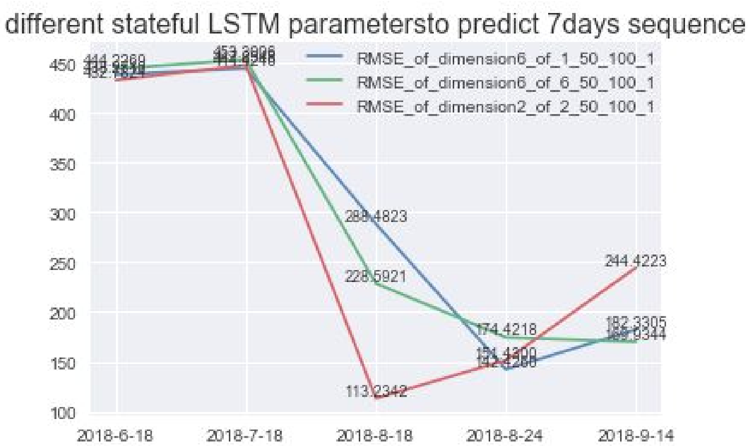


figure 15. different stateful LSTM parameters and 7-days prediction

The other comparison towards single layer LSTM and stateful LSTM is listed below. More layers and more neurons will be able to learn more features of data, then the model could be more accurate or could be over-fitting. From the outcome shows below, one layer LSTM performs better than three layers, the reason might be that the price of Bitcoin is volatile therefore the more neurons and layers will cause overfitting and less accurate on test dataset prediction.

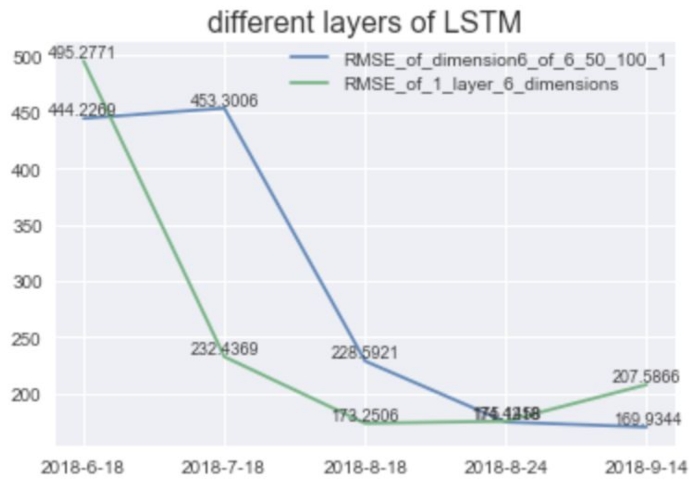


figure 16. different LSTM layers comparison

8. Discussion

In the project we use daily Cryptocompare data and build a series of predictive regression machine learning models : ARIMA, random forest, extreme gradient boosting and LSTM. We setup up control groups range from 30 days to 270 days to conduct experiments and evaluated which model most accurately predicts future bitcoin price.The results shown below:

Train size	Model (parameters)	RMSE	Training time
30 days	random forest (40 trees)	638.9019	0.016s
90 days	xgboost (10 iterations, depth 3)	896.9155	0.100s
180 days	random forest (50 trees)	508.8293	0.087s
270 days	xgboost (20 iterations, depth 3)	412.4730	0.206s
30 days	ARIMA (1, 1, 1)	189.7593	0.171s
90 days	ARIMA (1, 1, 2)	113.9405	0.510s
180 days	ARIMA (1, 1, 1)	94.5382	0.236s
270 days	ARIMA (1, 1, 1)	91.1831	0.448s
360 days	ARIMA (1, 1, 1)	49.0945	0.338s

90 days	Single layer LSTM (window size = 90 days, six dimensions), sequence prediction	83.58	12.3s
90 days	Single layer LSTM,window size = 7days, Single time point prediction	113.64	0.33s
1700 days	Stateful LSTM (2,50,100,1), window size = 7 days	113	3.2s

As can be seen from the table,we chose 2 measurements, RMSE for model accuracy and training time to measure the complexity required to train.It's obvious that with training size going up,models' RMSE is decreasing.Especially,ARIMA achieves best result with RMSE 49.0945 when feeding 360 days data.Since the training size is not too large,the numeric value is not very important.
We conclude that the ensemble trees method does not accurately represent future bitcoin prices.ARIMA methods demonstrated best results, with training size using 360 days being the most effective.

Ensemble Trees

Overall ensemble tree methods did not accurately represent future bitcoin prices.
Visualisations demonstrate models don't appear suitable for time-series predictions.

ARIMA

Overall ARIMA methods demonstrated best results, with training size using 360 days being the most effective.
Reasons may be ARIMA handles non-stationarity most effectively.

LSTM

Overall LSTM performed the most good in both predictions as well as runtime.

We chose 2 measurements, RMSE for model accuracy and training time to measure the complexity required to train.

Overall results demonstrated that ARIMA model most accurately predicts future bitcoin prices.
Future work will further investigate more parametric configurations of LSTM, as many research papers highlight this model best predicts future bitcoin prices.

For our models we chose 4 techniques:
8.1 ARIMA
8.2 Random forest
8.3 Extreme gradient boosting
8.4 LSTM

9. Limitations and Future Works

For the decision tree models, we observed that using longer training periods resulted in more robust models to accurately predict future bitcoin prices and movements. However we note that a limitation of using decision tree based models, such as random forest and extreme gradient boosting, is likely to occur on new test data outside the domain of the original independent training set. This issue is due to the nature of how decision trees create predictions based on regions of the domain space.

Other additional future works for random forest and extreme gradient boosting would be building a probabilistic confidence interval on top of the deterministic predictions. This is of particular importance given the prediction model is for regression purposes, but also may be necessary given the high volatility demonstrated by bitcoin price movements. As the models are decision tree based, one method to build a confidence interval may be to use bootstrapping and build an ensemble of random forest and extreme gradient boosting models. This will provide a distribution of regression prediction outputs which can then be used to build a confidence interval.

From the results of model LSTM, more dimensions will help for increasing the model accuracy. Therefore, more feature engineering is ought to be in the future, for example, web crawling data from Twitter and Reddit, which contain sentiment index about Bitcoin price everyday, could be two more dimensions and provide extra information when the LSTM model predicts the future Bitcoin price.

The other future works could be done towards current LSTM could be data preprocessing as we have known that stationary data will help the model to increase the accuracy and decrease the RMSE. A proven good method to achieve stationary data is difference, however, when difference is applied to the LSTM model, the loss will turn to be nan very quick so the model cannot converge and get results, even I have tried different optimizer, batch size, dropout rate and sometimes learning rate, even different feature combinations and different LSTM structure, but all to no avail, the loss all goes to nan so the model cannot be trained and applied. In the future, I would do more research to solve this issue and apply difference preprocess to data before normalization, extract window and training.

Meeting Minutes

No. 1	Date: 08/08/2018
Members: 1. Nathan SIMONSON (470519459) 2. Junqi WANG (450463619) 3. Jing SHAN (460247290) 4. Chenghao QIAN (4604042280) 5. Libai SUN (460140603)	
Meeting Contents: Introduce ourselves to each other. Discuss about the topics and figure out the scope of the project.	
Tasks: Do background research of bitcoin. Find API to search for Bitcoin historical data.	

No. 2	Date: 15/08/2018
Members: 6. Nathan SIMONSON (470519459)	

7. Junqi WANG (450463619)
8. Jing SHAN (460247290)
9. Chenghao QIAN (4604042280)
10. Libai SUN (460140603)

Meeting Contents:
Compare the difference of dataset and try to choose one.

Tasks:
Nathan: Develop pipeline and prepare writing literature review.
Junqi: Define the scope and design a schedule for project.
Jing, Libai and Chenghao: Do research about deep learning and LSTM and find examples.

No. 3	Date: 22/08/2018

Members:
10. Nathan SIMONSON (470519459)
11. Junqi WANG (450463619)
12. Jing SHAN (460247290)
13. Chenghao QIAN (4604042280)
14. Libai SUN (460140603)

Meeting Contents:
Discuss the proposal. Assign tasks to every member.

Tasks:
Nathan: Write abstract, introduction and literature review. Check gramma.
Jing: Write resource and check the format of reference list.
Chenghao: Write method part in the report.
Junqi: Write project problems, expected outcomes and schedule. Unify the format.

No. 4	Date: 29/08/2018

Members:
15. Nathan SIMONSON (470519459)
16. Junqi WANG (450463619)
17. Jing SHAN (460247290)
18. Chenghao QIAN (4604042280)
19. Libai SUN (460140603)

Meeting Contents: Discuss about the models and decide which model to use. Decide every member's task.

Tasks:
Nathan: Help Chenghao and Libai with their models.
Junqi and Jing: Visualisation of historical price.
Chenghao: Build ARIMA prediction model.
Libai: Build LSTM model.

No. 5	Date: 05/09/2018

Members:
20. Nathan SIMONSON (470519459)
21. Junqi WANG (450463619)
22. Jing SHAN (460247290)
23. Chenghao QIAN (4604042280)
24. Libai SUN (460140603)

Meeting Contents:
Discuss about the models and prediction period. Find other factors that may influence the result of prediction.

Tasks:
Nathan: Help Chenghao and Libai with their feature extraction.
Junqi and Jing: finish the visualization of historical price.
Chenghao: Finish the ARIMA prediction model.
Libai: Finish the LSTM model.

No. 6	Date: 12/09/2018

Members: 25. Nathan SIMONSON (470519459) 26. Junqi WANG (450463619) 27. Jing SHAN (460247290) 28. Chenghao QIAN (4604042280) 29. Libai SUN (460140603)
Meeting Contents: Review the draft result of prediction models and communicate the prediction data format.
Tasks: Nathan, Chenghao, Libai: Adjust the output of prediction format. Junqi, Jing: Working on making visualisation and dashboard.

No. 7	Date: 19/09/2018
Members: 30. Nathan SIMONSON (470519459) 31. Junqi WANG (450463619) 32. Jing SHAN (460247290) 33. Chenghao QIAN (4604042280) 34. Libai SUN (460140603)	
Meeting Contents: Discuss about the progress report and arrange tasks to every member.	
Tasks: Write progress report and finish the draft before next meeting.	

No. 8	Date: 03/10/2018
Members: 35. Nathan SIMONSON (470519459) 36. Junqi WANG (450463619) 37. Jing SHAN (460247290) 38. Chenghao QIAN (4604042280) 39. Libai SUN (460140603)	
Meeting Contents: Combination of all parts of progress report together and make adjustments.	
Tasks: Make improvement of models and dashboard.	

No. 9	Date: 10/10/2018
Members: 40. Nathan SIMONSON (470519459) 41. Junqi WANG (450463619) 42. Jing SHAN (460247290) 43. Chenghao QIAN (4604042280) 44. Libai SUN (460140603)	
Meeting Contents: Review the prediction models and LSTM still not have the ideal result. Compare ARIMA with other two models.	
Tasks: Libai: Finish the LSTM model. Nathan: Help Libai with the model building process. Chenghao: Update ARIMA model. Jing and Junqi: Finish the comparison of models.	

No. 10	Date: 17/10/2018
Members: 45. Nathan SIMONSON (470519459) 46. Junqi WANG (450463619) 47. Jing SHAN (460247290)	

48. Chenghao QIAN (4604042280) 49. Libai SUN (460140603)	
Meeting Contents: Compare the result and discuss the requirement of presentation.	
Tasks: The members need to make a draft ppt of the content before next meeting.	

No. 11	Date: 24/10/2018
Members: 50. Nathan SIMONSON (470519459) 51. Junqi WANG (450463619) 52. Jing SHAN (460247290) 53. Chenghao QIAN (4604042280) 54. Libai SUN (460140603)	
Meeting Contents: Discuss ppt arrangement and discuss about presentation.	
Tasks: All the members need to prepare presentation and ppt before next meeting.	

No. 12	Date: 30/10/2018
Members: 55. Nathan SIMONSON (470519459) 56. Junqi WANG (450463619) 57. Chenghao QIAN (4604042280) 58. Libai SUN (460140603)	
Meeting Contents: Presentation practice. Mock presentation.	
Tasks: Prepare final report.	

References

- Givens, G. H., & Hoeting, J. A. (2012). Computational statistics (2nd ed.). Chicester: Wiley.
- Bishop, C. M. (2006). Pattern recognition and machine learning. New York: Springer.
- Murphy K. (2012). Machine Learning - A Probabilistic Perspective. The MIT Press.
- Friedman J. (1999). Greedy Function Approximation: A Gradient Boosting Machine.
- Kirichenko L., Radivilova T. & Zinkevich I. (2017). Forecasting Weakly Correlated Time Series in Tasks of Electronic Commerce.
- Chatfield, C. and Yar, M., 1988. Holt-Winters forecasting: some practical issues. The Statistician, pp.129-140.
- CoinGecko. (2018). Bitcoin Price Chart (BTC/AUD) | CoinGecko. [online] Available at: https://www.coingecko.com/en/price_charts/bitcoin/aud [Accessed 30 Aug. 2018].
- Georgoula, I., Pournarakis, D., Bilanakos, C., Sotiropoulos, D. and Giaglis, G.M., 2015. Using time-series and sentiment analysis to detect the determinants of bitcoin prices.
- Gers, F.A., Schraudolph, N.N. and Schmidhuber, J., 2002. Learning precise timing with LSTM recurrent networks. Journal of machine learning research, 3(Aug), pp.115-143.
- Guo, T. and Antulov-Fantulin, N., 2018. An experimental study of Bitcoin fluctuation using machine learning methods. arXiv preprint arXiv:1802.04065.
- Lee, K.H. and Jo, G.S., 1999. Expert system for predicting stock market timing using a candlestick chart. Expert systems with applications, 16(4), pp.357-364.
- Liu, S., Liao, G. and Ding, Y., 2018, May. Stock transaction prediction modeling and analysis based on LSTM. In 2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA). IEEE.
- Liu, Y., Liu, X., Zhang, L., Tang, C. and Kang, H., 2017, November. An efficient strategy to eliminate malleability of bitcoin transaction. In Systems and Informatics (ICSAI), 2017 4th International Conferen
- Matta, M., Lunesu, I. and Marchesi, M., 2015, June. Bitcoin Spread Prediction Using Social and Web Search Media. In UMAP Workshops.
- McNally, S., Roche, J. and Caton, S., 2018, March. Predicting the price of Bitcoin using Machine Learning. In Parallel, Distributed and Network-based Processing (PDP), 2018 26th Euromicro International Conference on (pp. 339-343). IEEE.
- Mehrmolaei, S. and Keyvanpour, M.R., 2016, April. Time series forecasting using improved ARIMA. In Artificial Intelligence and Robotics (IRANOPEN), 2016 (pp. 92-97). IEEE.
- Reynard, C. (2018). What are the top 10 cryptocurrencies?. [online] The Telegraph. Available at: https://www.telegraph.co.uk/technology/digital-money/top-10-popular-cryptocurrencies-2018/ [Accessed 30 Aug. 2018].
- Robertson, G., Fernandez, R., Fisher, D., Lee, B. and Stasko, J., 2008. Effectiveness of animation in trend visualization. IEEE Transactions on Visualization and Computer Graphics, 14(6).
- Tolle, K.M., Microsoft Corp, 2012. User interface pane for an interactive chart. U.S. Patent 8,321,781.
- Yue, X., Shu, X., Zhu, X., Du, X., Yu, Z., Papadopoulos, D. and Liu, S., 2018. BitExTract: Interactive Visualization for Extracting Bitcoin Exchange Intelligence. IEEE transactions on visualization and computer graphics.
- C. Chatfield,Model uncertainty and forecast accuracy, 1996.Forecasting, 15 pp. 495-508
- G.E.P. Box, G. Jenkins, 1970. Time Series Analysis, Forecasting and Control, Holden-Day, San Francisco, CA
- Sundermeyer, Martin Schlüter, Ralf Ney, Hermann,2012. "LSTM neural networks for language modeling", In INTERSPEECH-2012, 194-197.
- F.A. Gers, J. Schmidhuber, F. Cummins, 2000. Learning to forget: continual prediction with LSTM Neural Computation, 12 (10) , pp. 2451-2471
- Donahue,Darrell.T., 2015. Long-term recurrent convolutional networks for visual recognition and description. In CVPR.
- C. De Groot, D. Wurtz, 1999.Analysis of univariate time series with connectionist nets: a case study of two classical examples Neurocomputing, 3 , pp. 177-192

- R. Rojas, 1996. Neural Networks: A Systematic Introduction, Springer-Verlag, Berlin
- Brillinger, D. R. (1981). Time series: data analysis and theory(Vol. 36). Siam.