G-Research Crypto Forecasting

Kaggle

Leung, Ko Tsun
Cheng, Tsz Yui
Yang, Po-Yen

About

Overview

G-Research Crypto Forecasting

Agenda

INTRODUCTION

Introduction

As cryptocurrency market has been skyrocketing recently, and the price have been wildly volatile, G-Research is trying to predict some of the cryptocurrency price movements in advance and forecast short-term log returns in 14 popular cryptocurrencies.

In this project, we utilized **light Gradient Boosting Machine** (light GBM), **LSTM** (Long Short-term Memory), **Multivariate Times Series Transformer**, and **Wavenet** as our prediction models.

Problem Statement

Observation

- It is estimated that over $40 billion worth of cryptocurrencies are traded everyday

Amount of features

- Motivated by this, we want to study whether we can predict the short-term movements of the 14 most popular cryptocurrencies
- We have lots of past cryptocurrency market prices provided by G-Research

Data Overview

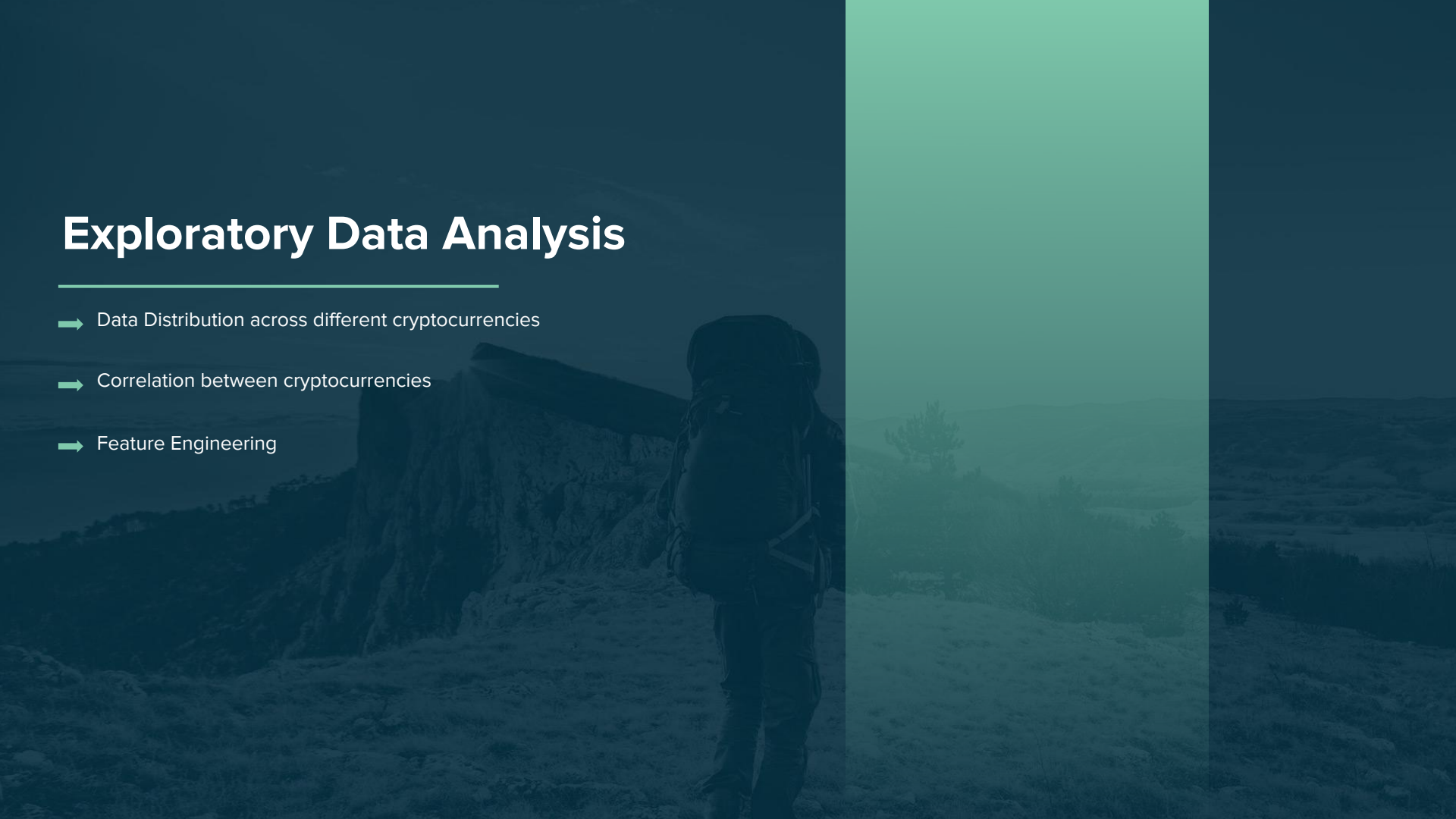
The dataset contains minute candlestick data from 1 Jan 2018 to now

Features	Explanation
Timestamp	All timestamps are returned as second Unix timestamps (the number of seconds elapsed since 1970-01-01 00:00:00.000 UTC). Timestamps in this dataset are multiples of 60, indicating minute-by-minute data.
Asset_ID	The asset ID is unique and corresponds to one of the cryptocurrencies (e.g. Asset_ID = 1 for Bitcoin). The mapping from Asset_ID to crypto asset is contained in "asset_details.csv".
Count	Total number of trades in the given time interval (minute)
Open	Opening price of the time interval (in USD)
High	Highest price reached during the time interval (in USD)
Low	Lowest price reached during the time interval (in USD)
Close	Closing price of the time interval (in USD)
Volume	Quantity of asset bought and sold (in USD)
VWAP	Volume Weighted Average Price
Target	Residual log-return for the asset over a 15-minute timeframe

Data & Data Preprocessing

Missing value handling

- Fill NaN with 0
- Use .reindex() with forward filling to fill the gaps in time series data



Exploratory Data Analysis

- Data Distribution across different cryptocurrencies
- Correlation between cryptocurrencies
- Feature Engineering

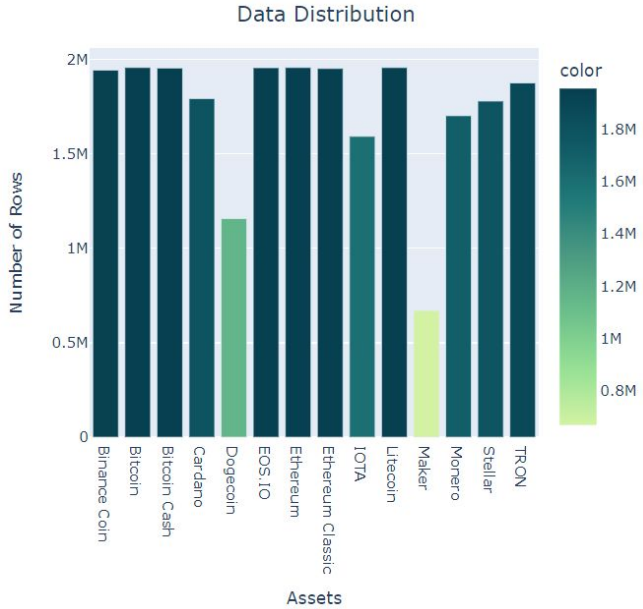
Exploratory Data Analysis

- An open-ended process where we calculate statistics and make figures to **find trends, anomalies, patterns, or relationships** within the data

Objectives:

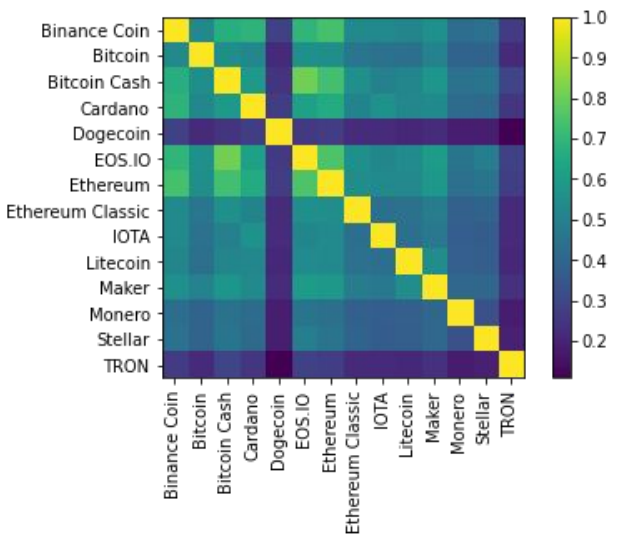
- To learn what the data can tell us
- Make decision on our model choices by helping us determine which **features** to use

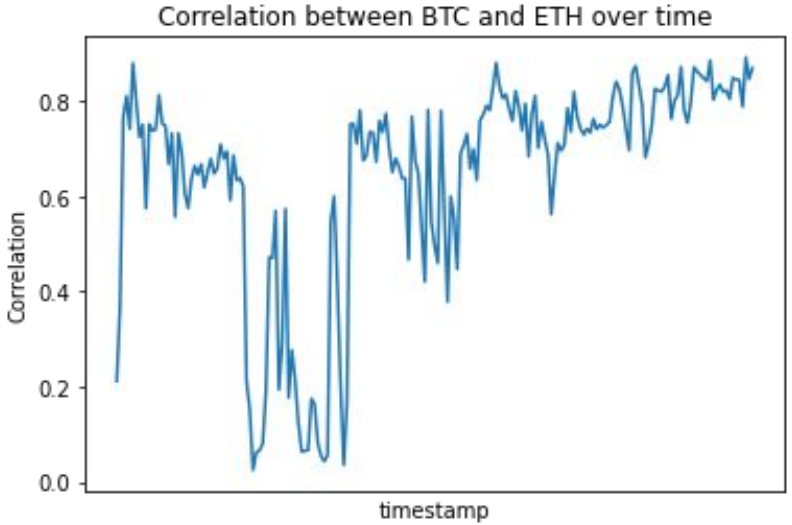
EDA - Data Distribution across different cryptocurrencies



Characterisics

- Most assets account for around 2 million data rows
- Fewer data for Dogecoin and Maker
- Reason: They have not gained popularity and hence trading volume until early 2021



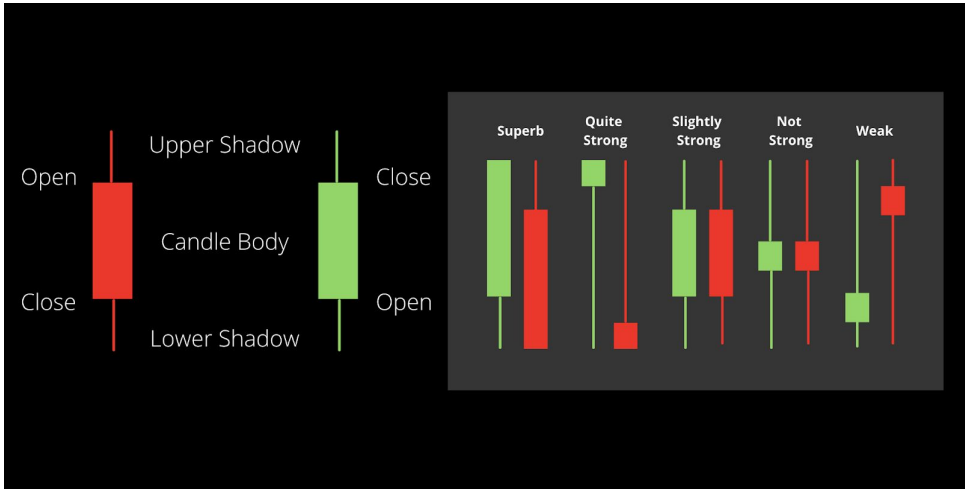


EDA

Correlation between cryptocurrencies

- Between BTC and ETH, a high but variable correlation is observed
- High volatility in correlation due to continuous change in mean, volatility, volume, etc.
- From the correlation matrix, some assets (e.g. bitcoin) has a much higher pairwise correlation than the others

Feature Engineering



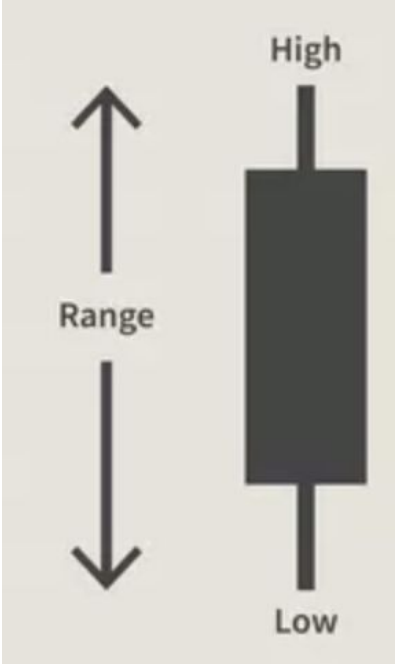
Upper/ Lower Shadows

- Represent the price action as the open/close prices differ from the high/low prices
- Length and position of the shadow can help us gauge market sentiment

Feature Engineering

Range = High - Low

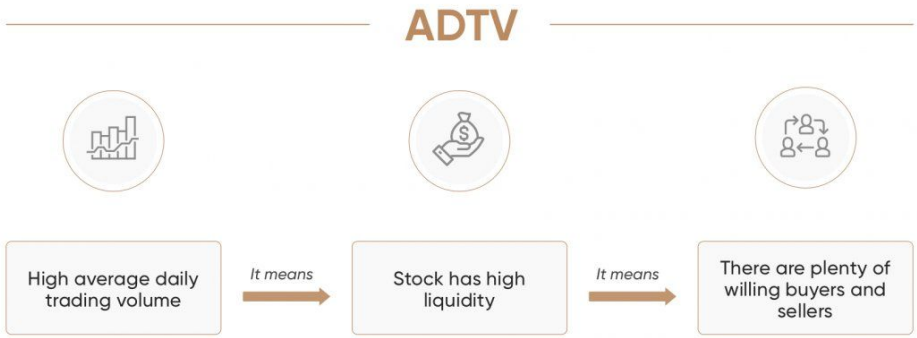
- Relative difference between high and low
- Defines the historical volatility
- It is a good indicator of risk



Feature Engineering

Average Trading Volume = Volume / Number of trade

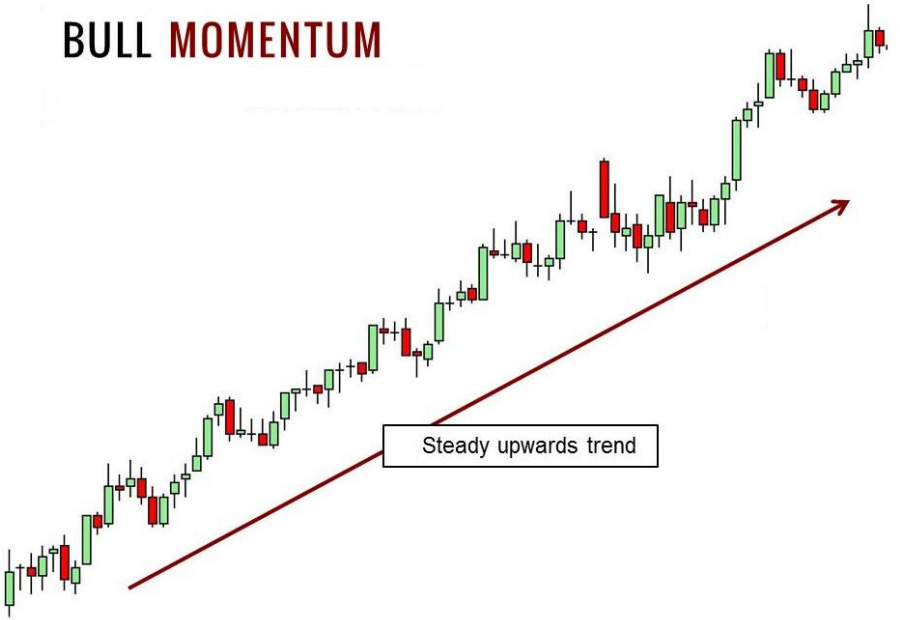
- Spread is the difference between period highs and lows.
- It measures the market sentiment toward security.
- Generally, if the mean trade is extremely high, meaning a major institutional investor has traded a large volume of cryptocurrencies on the market.
- It is used as an indicator of this behavior.



Feature Engineering

Log Price Change = Log(Close / Open)

- It can be used as the indicator of the price momentum of the asset.



Feature Engineering

|Close - VWAP|

- We created it and added to our features as we think it is meaningful
- Volume weighted average price (VWAP) is a popular trading benchmark used by traders that gives the average price an asset has traded in a given time interval, based on both volume and price.
- For institutional investors, they aim at trading at VWAP as close as possible since it can reduce their cost to absorb multiple layers in the market order book.
- When the price deviates much from the VWAP, investors tend to buy or sell more to push the price back to VWAP.

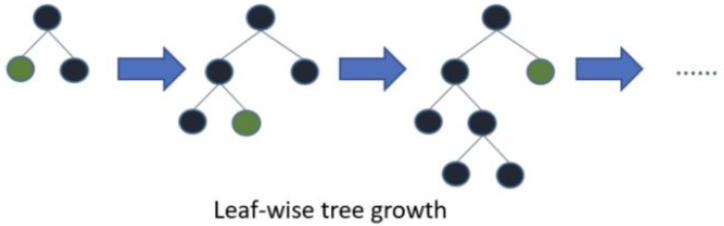
Models

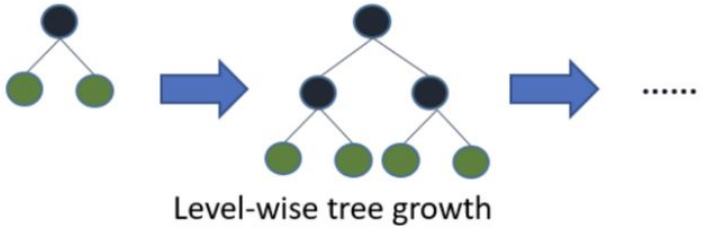
- Light Gradient Boosting Machine (LightGBM)
- Long Short-term Memory (LSTM)
- Multivariate Time Series Transformer
- Wavenet
- Model Techniques

1. LIGHT GRADIENT BOOSTING MACHINE **(Light GBM)**

Revolutionary gradient boosting decision tree (GBDT) method

- Developed by Microsoft in 2017, the most popular framework in Kaggle
- Fast, distributed, high-performance gradient boosting framework based on decision tree algorithm
- GBDT: ensemble model to learn decision by fitting residual errors in each iteration
- Source code (in C++): https://github.com/microsoft/LightGBM
- An improved model compared to XGBoost in terms of hardware and algorithms
- LGBM is Leaf-Wise Tree Growth, compared to Level-Wise Tree Growth in XGBoost
- User can minimize overfitting by limiting tree depth or number of leaves





1. LIGHT GRADIENT BOOSTING MACHINE **(Light GBM)**

Gradient Based One Side Sampling (GOSS) & Exclusive Feature Bundling(EFB)

- We want to select the Best Split with instances of larger gradients as they are undertrained
- XGBoost uses a histogram-based algorithm with O(#bins * #feature) times, but still needs O(#data * #feature) to construct histogram
- GOSS is a novel sampling method to retain large gradient instances, while keeping the data distribution by random sampling on small gradient instances
- EFB is used to bundle mutually exclusive features to reduce the complexity to O(#data * #bundle) where #bundle << #feature
- Takeaway: LightGBM adopts an effective way to split the decision tree based on gradient

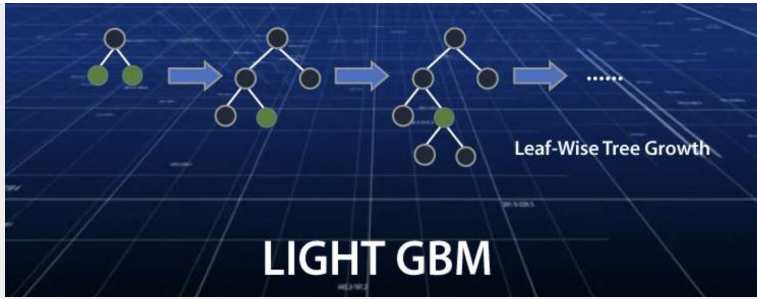
Algorithm 3: Greedy Bundling

Input: F: features, K: max conflict count
Construct graph G
searchOrder $\leftarrow$ G.sortByDegree()
bundles $\leftarrow$ {}, bundlesConflict $\leftarrow$ {}
for i ***in*** *searchOrder* **do**
  needNew $\leftarrow$ True
  for $j = 1$ ***to*** *len(bundles)* **do**
    cnt $\leftarrow$ ConflictCnt(bundles[j],F[i])
    if *cnt + bundlesConflict[i]* $\leq K$ **then**
      bundles[j].add(F[i]), needNew $\leftarrow$ False
      break
  if *needNew* **then**
    Add $F[i]$ as a new bundle to *bundles*
Output: *bundles*

1. LIGHT GRADIENT BOOSTING MACHINE **(Light GBM)**

Advantages of LightGBM

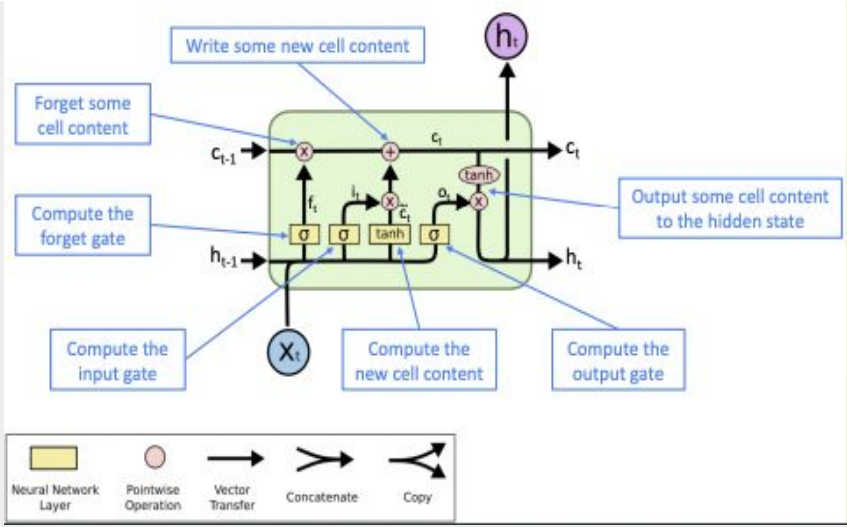
- Faster training speed and higher efficiency
- Lower memory usage
- Better accuracy than any other boosting algorithm
- Parallel learning supported
- Support categorical variables
- Compatible with large datasets



2. LONG SHORT-TERM MEMORY **(LSTM)**

Advanced Recurrent Neural Network (RNN) method

- Proposed by Hochreiter and Schmidhuber in 1997
- Special kind of RNN capable of handling long-term dependencies and the vanishing gradient problem faced by RNN
- RNN: remember the previous information and use it for processing current input
- Consists of forget gate, input gate, and output gate



2. LONG SHORT-TERM MEMORY **(LSTM)**

Forget Gate

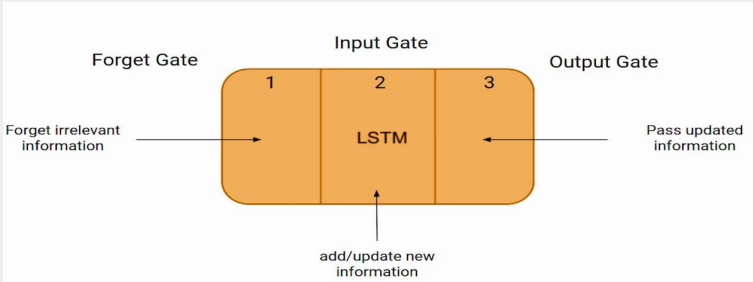
- Decide whether we should keep the information from the previous timestamp or forget it
- The equation for the forget gate is shown on the right
- Lastly a sigmoid function is applied to the calculated value
- The f_t is then multiplied with the cell state of the previous timestamp
- If f_t is 0, then the network will forget everything, otherwise, the network remember everything

Forget Gate:

- $f_t = \sigma(x_t * U_f + H_{t-1} * W_f)$

$C_{t-1} * f_t = 0$ ...if $f_t = 0$ (forget everything)

$C_{t-1} * f_t = C_{t-1}$ ...if $f_t = 1$ (forget nothing)



2. LONG SHORT-TERM MEMORY **(LSTM)**

Input Gate & New Information

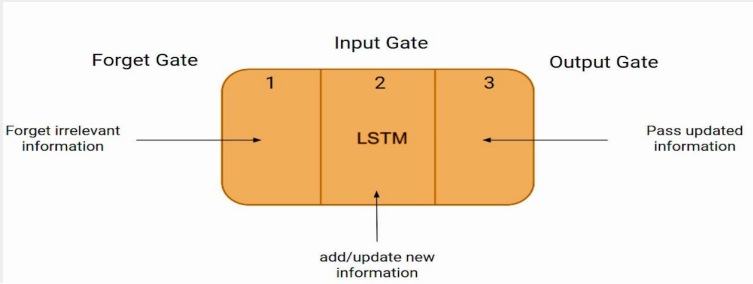
- Used to quantify the importance of the new information carried by the input
- The equation for the input gate is shown on the right
- Lastly a sigmoid function is applied to the calculated value
- The new information needed to be passed to the cell state is a function of the hidden state at the previous timestamp t-1 and input x at timestamp t
- If N_t is negative, then the information is subtracted from the cell state, otherwise, the information is added to the cell state
- N_t will not be added to the cell state directly, the equation for update is shown on the right

Input Gate:

- $i_t = \sigma(x_t * U_i + H_{t-1} * W_i)$

- $N_t = \tanh(x_t * U_c + H_{t-1} * W_c)$ (new information)

$C_t = f_t * C_{t-1} + i_t * N_t$ (updating cell state)



2. LONG SHORT-TERM MEMORY **(LSTM)**

Output Gate

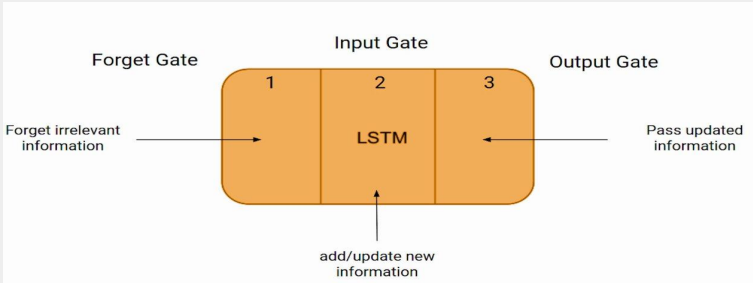
- Predict the output
- The equation for the output gate is shown on the right
- Lastly a sigmoid function is applied to the calculated value
- The hidden state is a function of Long term memory (C_t) and the current output
- Apply the Softmax function to take the output of the current timestamp

Output Gate:

- $o_t = \sigma(x_t * U_o + H_{t-1} * W_o)$

$$H_t = o_t * \tanh(C_t)$$

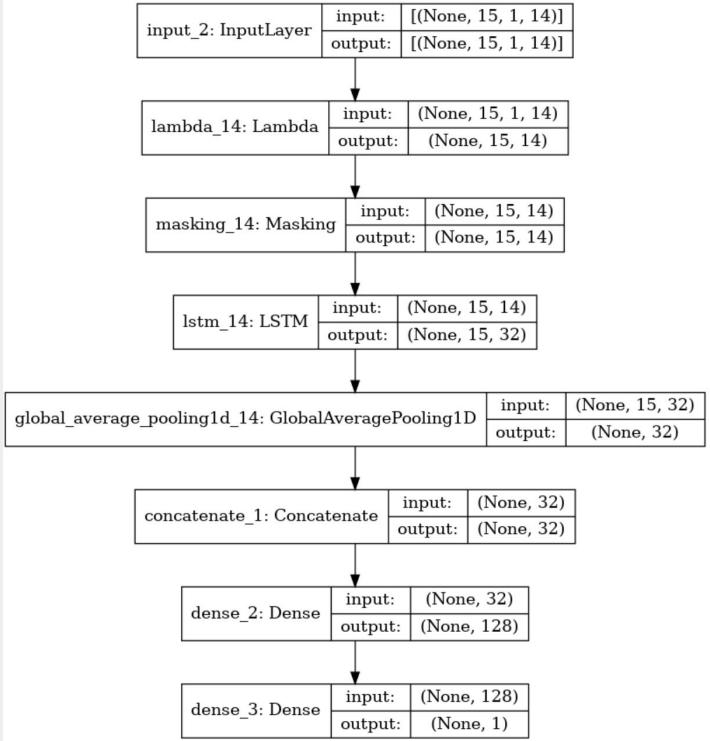
$$\text{Output} = \text{Softmax}(H_t)$$



2. LONG SHORT-TERM MEMORY **(LSTM)**

Advantages of LSTM

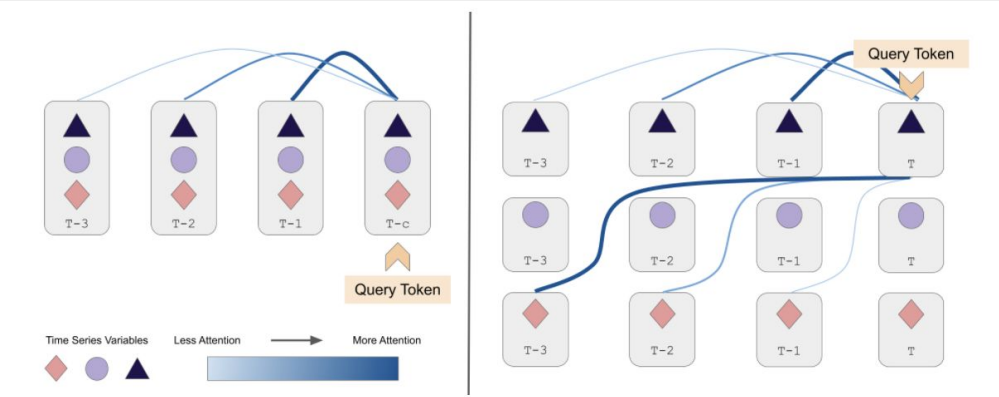
- Rectifies the short-memory problems from RNN
- Solve exploding and vanishing gradient problems
- Well-suited in time-series-based data



3. Multivariate Time Series Transformer

State-of-art model for Natural Language Processing

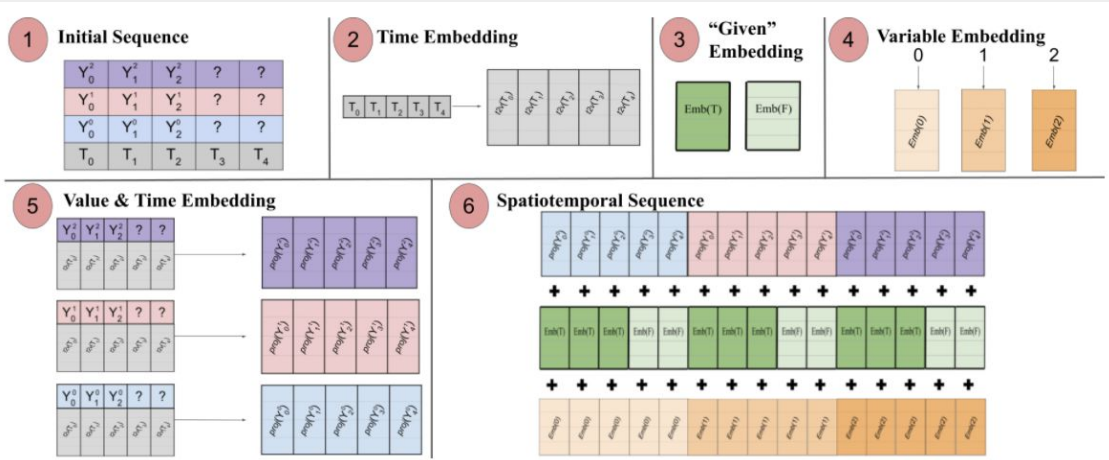
- Deep learning model with a sequence of attention-based transformer blocks
- Proposed in October 2020 by researchers from IBM Research & Brown University
- We extended the work to financial time series prediction
- Cryptocurrency prediction task is a multivariate time series problem: temporal & spatial relationship



3. Multivariate Time Series Transformer

Spacetimeformer

- Transform a multivariate time series signal into spatiotemporal sequence
- Value of each variable is projected to a high-dimensional space with a feed-forward layer
- Add information about timestep and variable corresponding to each token
- Time and variable embeddings are initialized randomly and trained with the rest of the model
- Set the values to be predicted at future timesteps to be 0, and tell the model which ones are missing with a binary “given” embedding
- Spacetimeformer can construct a spatiotemporal graph across time and variable space



3. Multivariate Time Series Transformer

Model Architecture

- A transformer-based encoder-decoder architecture processes the sequence and predicts the value of each variable at future timesteps as seperate tokens
- Re-stack the predictions and train to minimize prediction-error metric(MSE)
- Create a range of forecasts by outputting mean and std of a normal distribution

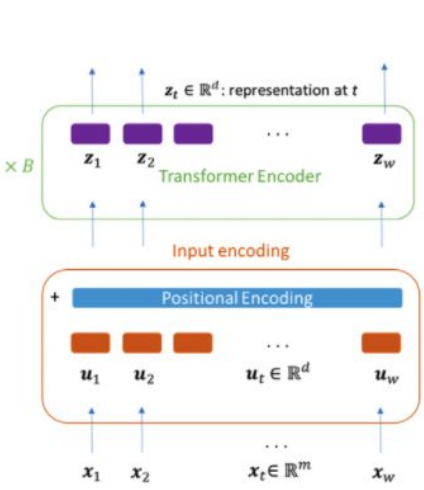
Model Details

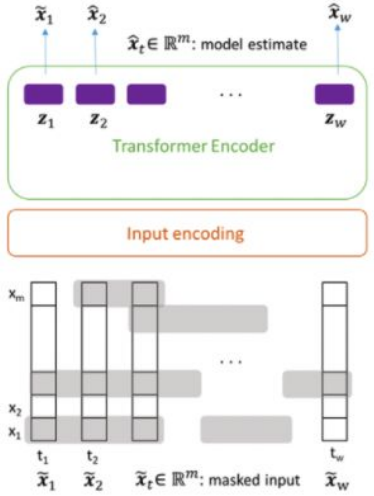
- Trained with Keras for 1000 epochs, Loss function = Masked MSE loss
- Masking probability of 15% is applied to sampled sequences features independently at random in geometric distribution

```
def get_transformer(x):
    n_heads = 8
    d_model = 128
    dropout = 0.1
    freeze = False
    num_layers = 3
    activation = 'gelu'
    dim_feedforward = 256

    x = Dense(d_model)(x) * math.sqrt(d_model)
    x = TransformerEncoder(
        d_model,
        n_heads,
        dim_feedforward,
        dropout * (1.0 - freeze),
        activation = activation)(x)

    return x
```

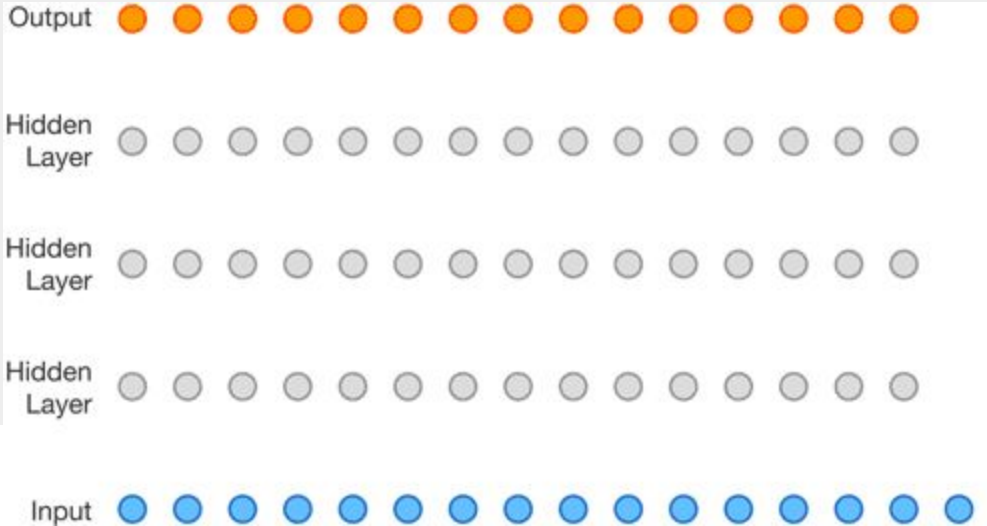




4. Wavenet

Deep generative model for raw audio

- Proposed by DeepMind from Google, and it is the best existing Text-to-Speech system
- It can mimic any human voice and reduce the gap with human performance by over 50%
- We extended it to the cryptocurrency prediction, as it showed extraordinary result with other time series prediction problems



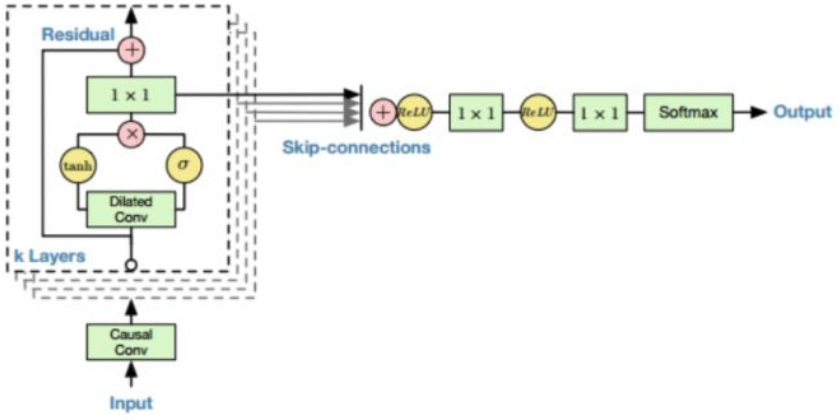
4. Wavenet

```
def get_wavenet(x):
    def wave_block(x, filters, kernel_size, n):
        dilation_rates = [2 ** i for i in range(n)]
        x = Conv1D(filters = filters, kernel_size = 1, padding = 'same')(x)
        res_x = x
        for dilation_rate in dilation_rates:
            tanh_out = Conv1D(filters = filters, kernel_size = kernel_size,
                              padding = 'same', activation = 'tanh', dilation_rate = dilation_rate)(x)
            sigm_out = Conv1D(filters = filters, kernel_size = kernel_size,
                              padding = 'same', activation = 'sigmoid', dilation_rate = dilation_rate)(x)
            x = Multiply()([tanh_out, sigm_out])
            x = Conv1D(filters = filters, kernel_size = 1, padding = 'same')(x)
            res_x = Add()([res_x, x])
        return res_x
    x = wave_block(x, 16, 3, 8)
    x = wave_block(x, 32, 3, 5)
    return x
```

Different components

- Dilated causal convolutions
    - Causal convolution is used to constrain so that the prediction cannot depend on any of the future timestamps
    - Dilated convolution is convolution where the filter is applied over an area larger than its length
    - by skipping input values with certain steps
    - enabling networks to have larger receptive fields
- Gated activation units are used, as it can control which information to pass through, better than ReLU
- Both residual and parametrized skip connections are used to enable training of much deeper models

$$\mathbf{y} = \tanh(W_f * \mathbf{x}) \odot \sigma(W_g * \mathbf{x})$$



5. Model Techniques

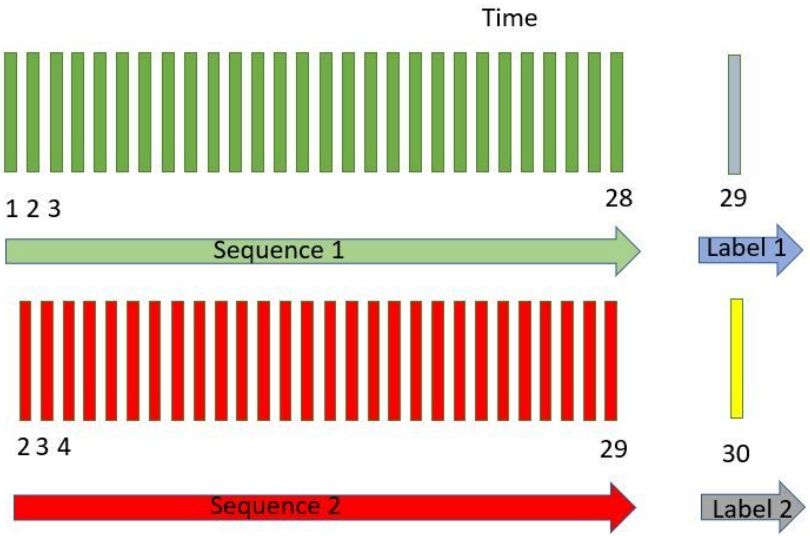
Time Series Model Techniques

- Time Series Sliding Window
- Time Series Cross Validation

5. Model Techniques

Time Series Sliding Window

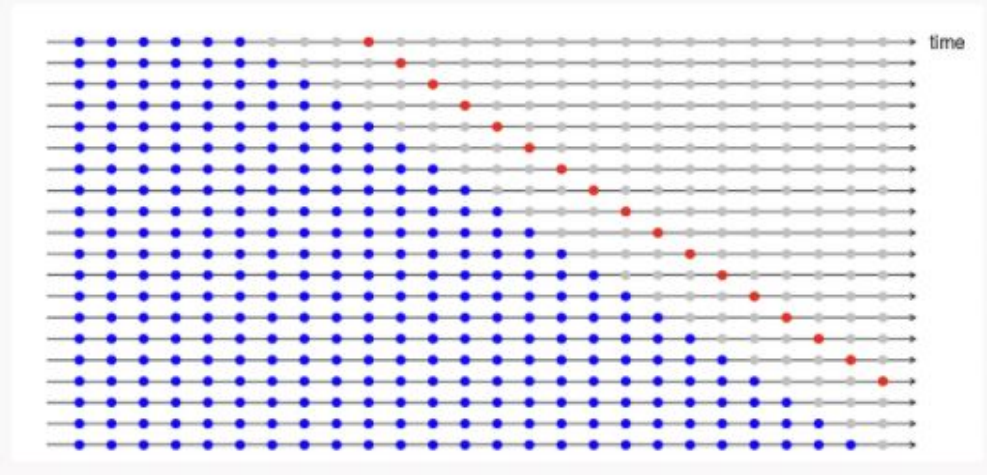
- Use a sliding window to build sequences and labels
- Lag method in statistics and time series analysis
- A sliding windows size of 15 is used for each variable
- Turn the dataset into sequences of 15 minutes and their labels



5. Model Techniques

Time Series Cross Validation

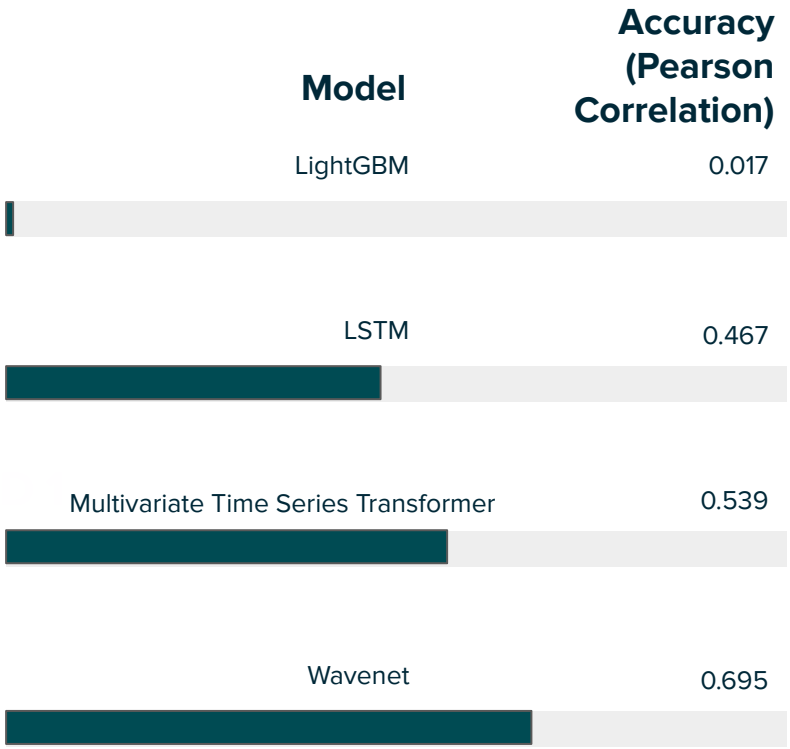
- The original version chooses random samples and assigns them to either the test set or the train set.
- However, when it comes to time series, we cannot apply the original cross-validation as it makes no sense to use the values from the future to predict values in the past.
- Hence, we have to apply time series split cross-validation, where the data are chosen in order and time series split divides the training set into two folds at each iteration on the condition that the validation set is always in front of the training set.
- In our analysis, we used the 10-folds cross-validation technique to train our models.



RESULTS

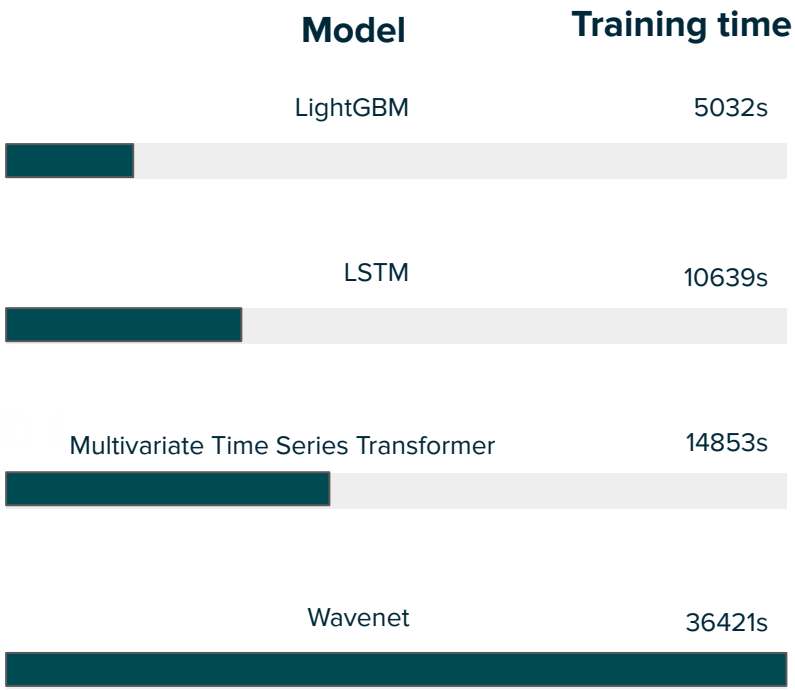
Results

The following models and architectures are used as trials. **Wavenet** is chosen as our model as it achieves the highest accuracy

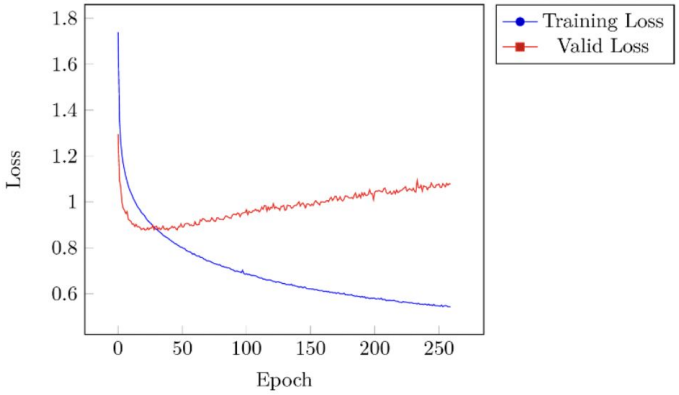


Results

Although **Wavenet** has the highest accuracy, it has the largest computation time during training stage, as its model complexity is high
LSTM is fast to implement, train, and test, so it can serve as a baseline to this problem

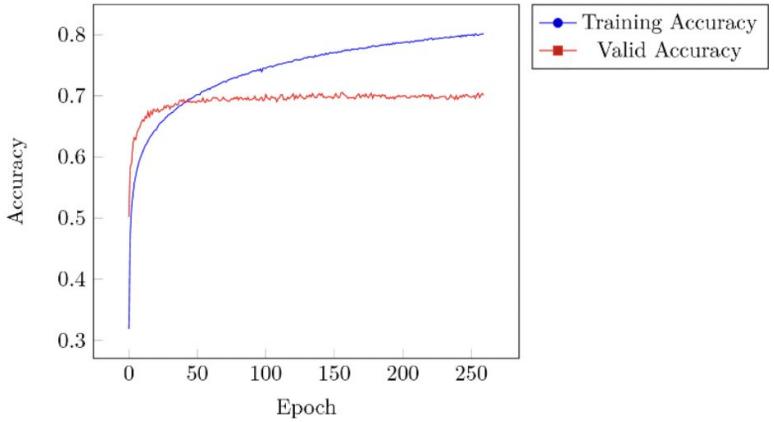


Results



Mean squared error for each epoch for Wavenet model

Wavenet model can nicely capture the multivariate time-series relationship
Early stopping should be used to avoid overfitting



Validation accuracy converges to 0.7 after 50 epochs

BACKTESTING

Backtesting

Start	0.0
End	729.0
Duration	729.0
Exposure Time [%]	69.45
Equity Final [$]	353.5889
Equity Peak [$]	100000.0
Return [%]	60.646411
Buy & Hold Return [%]	-24.660214
Return (Ann.) [%]	0.0
Volatility (Ann.) [%]	NaN
Sharpe Ratio	NaN
Sortino Ratio	NaN
Calmar Ratio	0.0
Max. Drawdown [%]	-10.646232
Avg. Drawdown [%]	-10.646232
Max. Drawdown Duration	159.0
Avg. Drawdown Duration	159.0

- We have performed a backtesting analysis to further validate our conclusion.
- We first start at $100,000 cash.
- A simple strategy is used: when a cryptocurrency is consecutively predicted to have a positive correlation in the next 15 minutes for 3 minutes, buy $5,000 dollars equivalent amount of this asset. Otherwise, sell the asset with the same amount.
- We tested this strategy on the portfolio of multiple assets for two years, with the weighting specified in the asset_details.csv provided by the competition organizer.
- It can gain 60% profit in 2 years from the start of 2018 to the end of 2019. It proved that our strategy successfully passed the backtesting analysis and it outperforms the price movement of the portfolio of cryptocurrencies.

CONCLUSION

Conclusion

Model Comparison

1) Among the four models, Wavenet performed the best but needs the longest computational time
2) Wavenet is suitable for predicting cryptocurrency markets as it is similar to the stock market

Conclusion

Improvements

1) Try training different deep learning time series models, for example ensembling methods, data augmentation, etc
2) Extend or Shorten the prediction period by using Hierarchical Time-series (HTS)

Reference

A Transformer-based Framework for Multivariate Time Series Representation Learning https://arxiv.org/abs/2010.02803
Self-supervised Transformer for Multivariate Clinical Time-Series with Missing Values
https://arxiv.org/abs/2107.14293
Wavenet variations for financial time series prediction: the simple, the directional-Relu, and the probabilistic approach
https://medium.com/analytics-vidhya/wavenet-variations-for-financial-time-series-prediction-the-simple-the-directional-relu-and-the-4860d8a97af1
Introduction to Long Short Term Memory (LSTM)
https://www.analyticsvidhya.com/blog/2021/03/introduction-to-long-short-term-memory-lstm/


Thank You