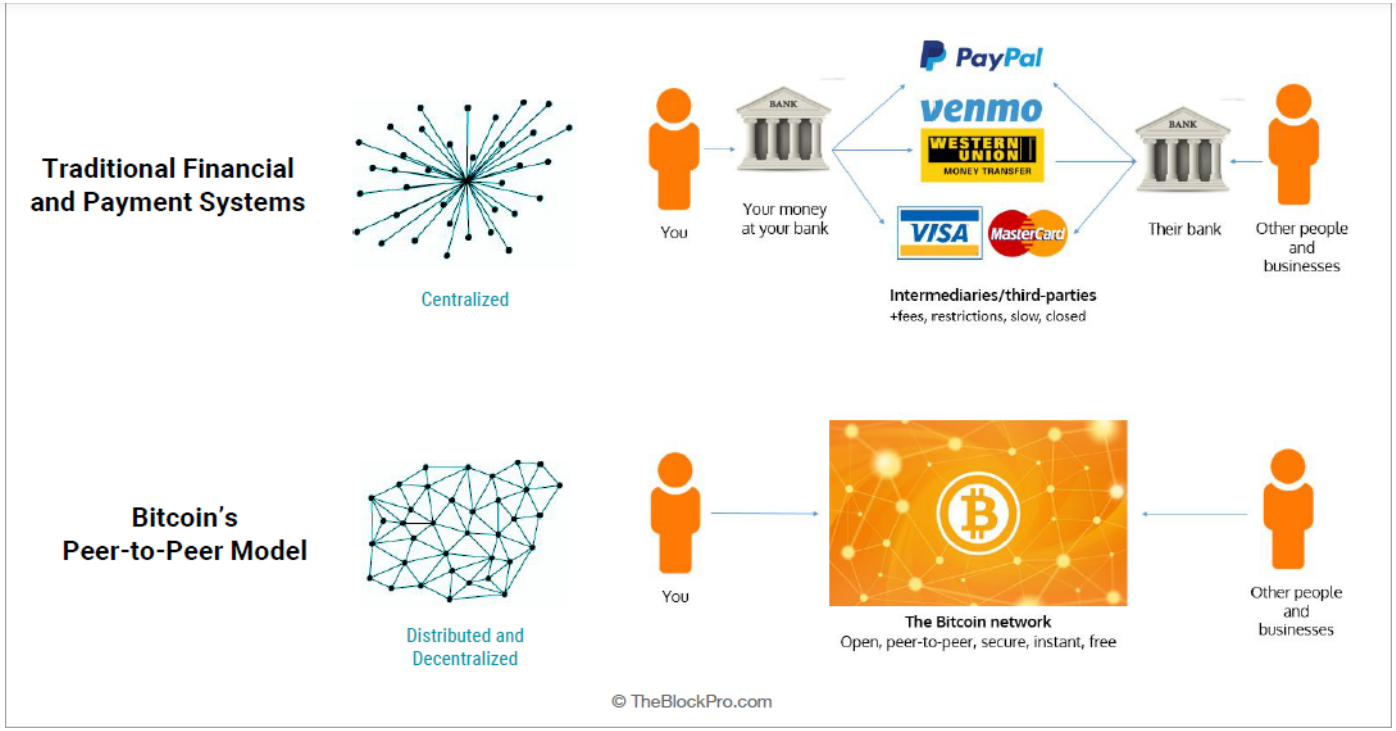
Rohan Sawant Follow

Sep 22, 2021 · 13 min read · Listen

Bitcoin price prediction using Machine Learning

Ref-https://currency.com/bitcoin-price-analysis-march-23-to-29

1. What is bitcoin?



Ref-https://theblockpro.com/what-is-bitcoin/

Bitcoin is a decentralized digital currency without any central administrator and can be sent directly from user to user on the bitcoin network. Network nodes verify transactions through cryptography and are recorded in a public distributed ledger called a blockchain which also track ownership, prevent tampering of transaction records, prevent double spending.

increases volatility. E.g., In Jan 2017 prices were around 1000 USD which later increased to 16000 USD by December 2017. Then it again saw some gain and fall, but in recent times the prices skyrocketed to 63000 USD.

2. Problem Statement.

One of the main problems with bitcoin is price volatility, which indicates the need for studying the underlying price model.

This case study is based on Time-series forecasting of Bitcoin prices. The paper considered bitcoin data from April 2013 to December 2019. It proposed this as both regression and classification problem. For regression the paper was predicting prices & for classification task where an attempt was made to predict increase/decrease in price. It was performed to predict next day, 30th day & 90th day price.

In this case study, we attempt to predict next day prices based on features of previous day using machine learning & deep learning regression algorithms.


As the target variable is highly volatile, we want to penalize the model when the error is higher. Root Mean Squared Error (RMSE) squares the errors before averaging, and thus RMSE gives a relatively high weight to large errors. We have also considered Mean Absolute Error (MAE) because of its easy interpretability.

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i|$$

$$n = \text{number of datapoints}$$
$$y_i = \text{Actual value}$$
$$\hat{y}_i = \text{Predicted value}$$

Metrics

4. Data Collection & Imputation

4.1 Web Scraping using bitinfocharts.com

Bitinfocharts provides following 17 raw features. The features were retrieved using a custom python script.

i) Number of transactions in blockchain per day

Bitcoin transactions are messages digitally signed using cryptography and sent to the entire Bitcoin Network for verification. The number of daily transactions highlights the value of the Bitcoin network to securely transfer

Blocks are files which permanently records most recent transactional information related to the Bitcoin network. It cannot be altered or removed after writing. Each time a block is 'completed', the next block comes in the blockchain. The maximum block size is currently set at 1 megabyte.

iii) Number of sent by addresses

These are distinct addresses from which payments are made every day.

iv) Number of active addresses

These are number of unique addresses taking part in a transaction by either sending or receiving Bitcoins.

v) Average mining difficulty (Hash/day)

It reflects how difficult the proof of work calculation with respect to the difficulty value set at the beginning (1). A higher difficulty means that it will take more computing power to mine the same number of blocks, making the


being created within a 2-week period, and will be reduced if less blocks are created.

$$\text{difficulty}_{new} = \frac{\text{difficulty}_{old} \times 2016\ blocks \times 10\ minutes}{\text{time took in minutes to mine the last 2016 blocks}}$$

Mining Difficulty Formula

vi) Average hash rate (hash/s)

Hash rate is a measure of the mining computational power per second used. It is measured as how many calculations per second can be performed. Machines with a high hash power are highly efficient and can process a lot of data in a single second.

Vii) Mining Profitability (USD)

It is mining profitability for 1 Hash/s


It is the total value of Bitcoins sent daily.

ix) Average & Median transaction fee (USD)

Each transaction can have a transaction fee determined by the sender and paid to miners who verify the transaction. Transactions with higher fees reward the Bitcoin miners to process them sooner than transactions with lower fees.

x) Average block time (minutes)

Block time is the time required to create the next block in a chain. It is a time taken by a blockchain miner to find a solution to the hash. Usually, it is around 10 minutes, but can fluctuate depending on the hash rate of the network.

xi) Average & Median Transaction Value (USD)

The average & median value of the transactions in Bitcoin

xii) Tweets & Google Trends to “Bitcoin” per day

xiii) Average Fee Percentage in Total Block Reward

Bitcoin block rewards are new bitcoins awarded to miners for being the first to solve a complex math problem and creating a new block of verified bitcoin transactions. Rewards were started at 50 BTC and it halves every 210,000 blocks. The current reward lies at 6.25. This feature is the ratio of the fee sent in a transaction to the reward for verifying that transaction by the other users.

xiv) Top 100 Richest Addresses to Total coins

This is the ratio between top 100 rich addresses to total coins.

4.2 Using Quandl

Following Two Features were collected using Quandl API

i) Miner Revenue (USD)

Total value of coin base block rewards and transaction fees paid to miners.


It is a total number of mined bitcoins that are currently circulating on the network. The total supply of BTC is limited to 21 million.

4.2 Using investpy

The OHLC features & target variable is mined using investpy API, which retrieves data from investing.com. Values of the previous day are used to predict next day closing price.

i) Opening Price

ii) Highest Price

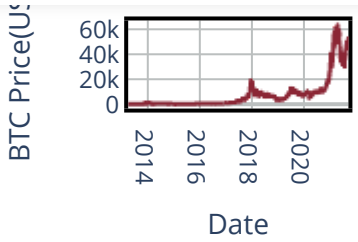
iii) Lowest Price

iv) Closing Price

Final dataset was considered from 1 April 2013 to 18 September 2021 having 3093 rows, 23 features, date & target variable. We imputed missing values using a simple moving average. Window size for SMA was selected based on

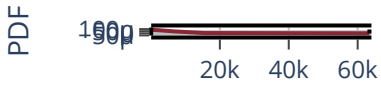
5. Exploratory Data Analysis.

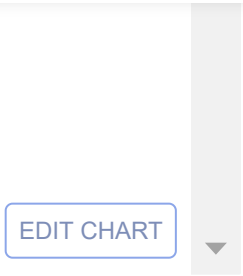
5.1 Univariate analysis of Target variable



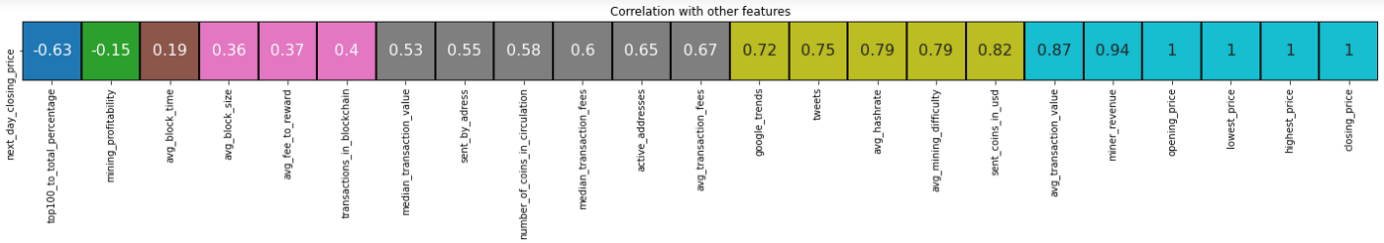
EDIT CHART

KDE Plot-Bitcoin Actual Closing

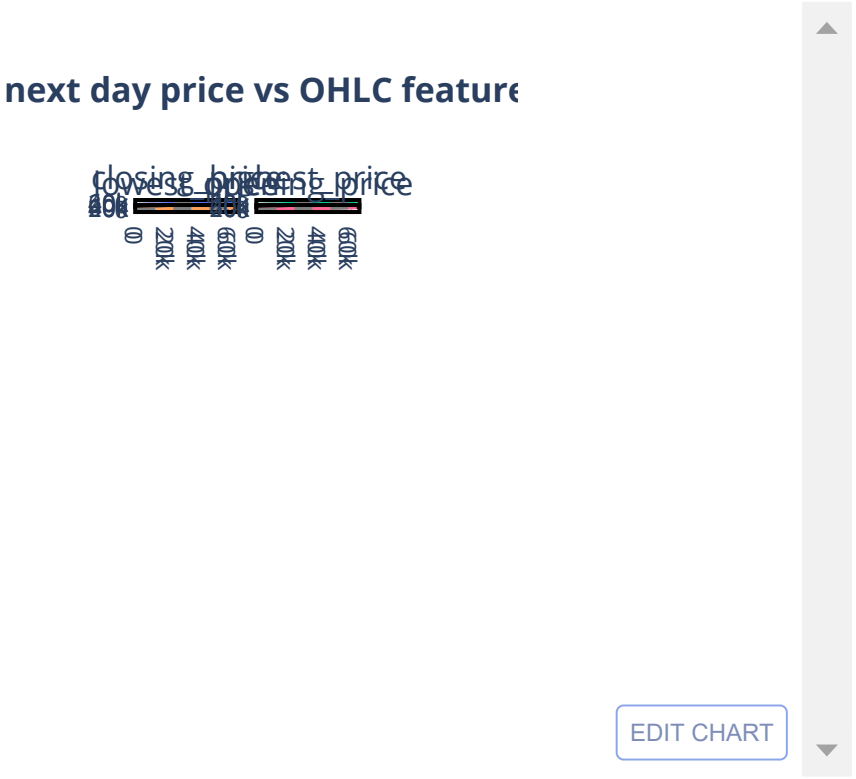




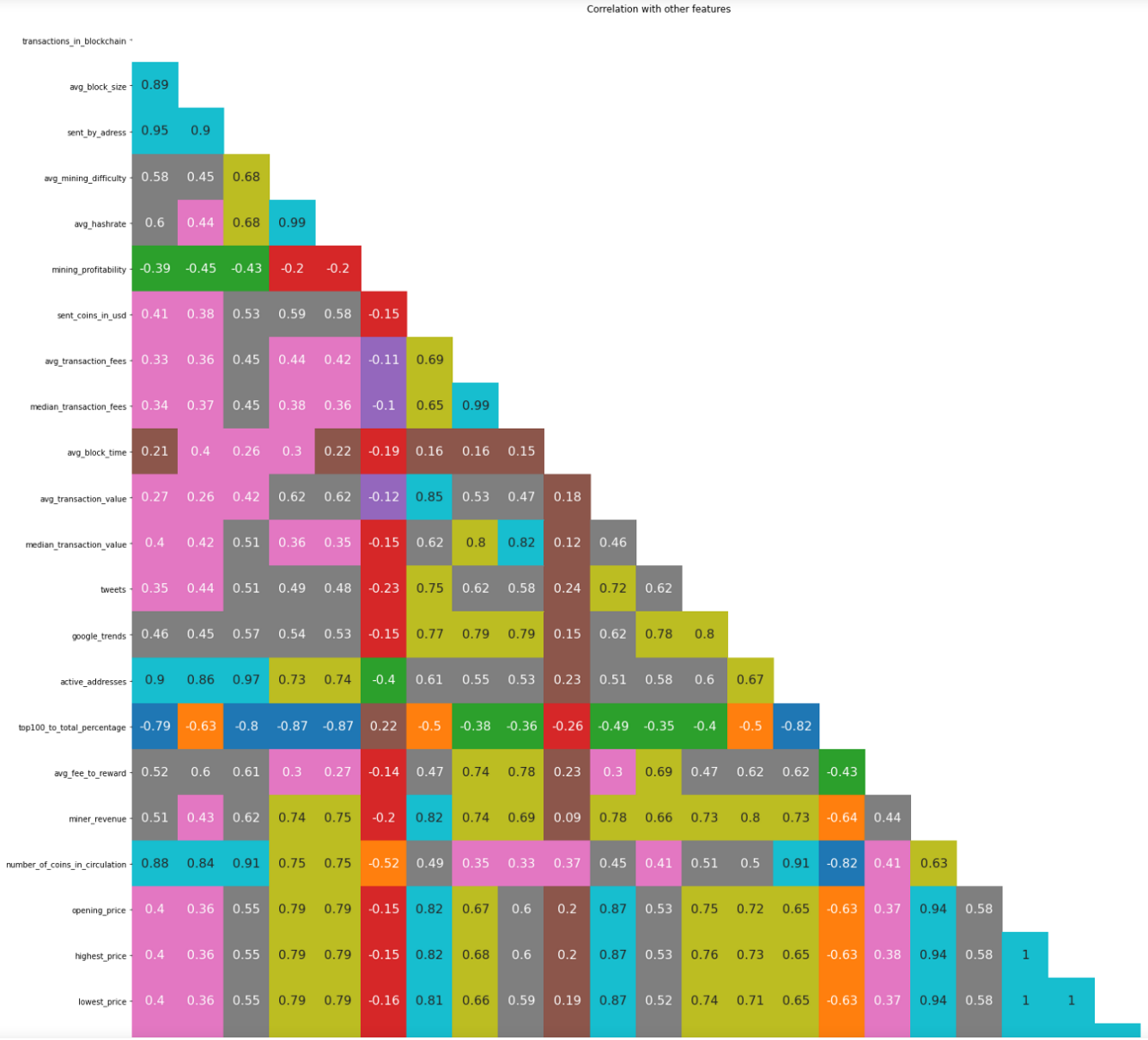
We can observe price volatility from skewed KDE distribution & line chart. There are two big spikes in distribution. From April 2013 to April 2017, prices were stable between 100 USD to 1000 USD. From May 2017, the trend started increasing and reached 19345 USD by Dec 2017. From Then to Sep 2020, prices were stable, but having some increasing/decreasing trends but not huge spikes. In Oct 2020. Prices started to increase heavily and reached 63500 USD by April 2021. It continued to have some increase/decrease, but fell down sharply to 30000 USD by July 2021. After July to September 2021, the prices are increasing and currently are in range, from 40000 USD to 50000 USD.



Correlation of target variable with other features



It shows very good correlation with OHLC features and scatter plot also gives a linear relationship. These features will be very useful for prediction. Other well correlated features are Miner revenue, average transaction value. They do

1 1 1 Analysis Export

	feature1	feature2	corre
1	opening_price	highest_price	1
2	closing_price	highest_price	1
3	closing_price	lowest_price	1
4	opening_price	lowest_price	1
5	lowest_price	highest_price	1
6	opening_price	closing_price	1
7	avg_hashrate	avg_mining_difficulty	0.99
8	median_transaction_fees	avg_transaction_fees	0.99

Feature to Feature Correlation created with datapane

We can observe that there is very good correlation between OHLC features, as distribution will be closely similar to each other.

Average fee and median fee follow a very similar distribution, which implies that on a single day, neither the fees are paid heavily nor they are too less.

There is a good correlation between sending address & active addresses, as sent address is a subset of active addresses.

Miner revenue generates new bitcoins as a reward, thus increasing supply of bitcoins, which ultimately impacts the price.

6. Feature Engineering- Technical Indicators.

Technical traders or investors use technical indicators to predict future price trends and make trading decisions. Technical indicators are calculated from historical data, which provides insights about data pattern. There are different indicators which show trend, momentum, volatility, volume. e.g., Moving averages, Rate of change, Bollinger bands, Moving Average Convergence Divergence, standard deviation, etc.

We are considering following technical indicators of each feature for a period


6.1 Simple Moving Average

Loading graph

SMA calculates the average of a selected range of feature values for a number

Loading graph

WMA assigns a greater weighting to the most recent data points, and less weighting to data points in the distant past. The sum of the weighting should add up to 1 (or 100 percent)

$$\text{WMA} = \frac{\text{Price}_1 \times n + \text{Price}_2 \times (n-1) + \cdots + \text{Price}_n}{\frac{n(n+1)}{2}}$$

6.3 Exponential Moving Average

The EMA also places a greater weight on the most recent data points like WMA but the rate of decrease between one price and its preceding price is not consistent. The difference in the decrease is exponential.

$$EMA_{Today} = (Value_{Today} \times (\frac{Smoothing}{1 + Days})) + EMA_{Yesterday} \times (1 - (\frac{Smoothing}{1 + Days}))$$

6.4 Double Exponential Moving Average

Edit chart
Loading graph

DEMA responds more quickly to near-term price changes than a normal exponential moving average (EMA). It helps to filter out noise.

$$DEMA = 2 \times EMA - EMA(EMA)$$

6.5 Triple Exponential Moving Average

Edit chart
Loading graph

It uses multiple EMA calculations and subtracts the lag to create a trend following indicator that reacts quickly to price changes.

$$TEMA = 3 \times EMA - 3 \times EMA(EMA) + EMA(EMA(EMA))$$

6.6 Standard Deviation

Edit chart
Loading graph

Standard deviation is the statistical measure of market volatility, measuring how widely feature values are dispersed from the average feature values. If feature values trade in a narrow trading range, the standard deviation will return a low value that indicates low volatility. Conversely, if feature values swing wildly up and down, then the standard deviation returns a higher value that indicates high volatility.

6.7 Variance

Variance is another statistical measure of market volatility, measuring how widely feature values are dispersed from the average feature values. It is interpreted similarly to standard deviation, variance is square of standard deviation.

6.8 Relative Strength Index

An asset is considered oversold or undervalued when the RSI drops below 30. On the other hand, it's deemed to be overbought if the RSI goes above 70.

$$\text{RSI} = 100 - \frac{100}{1 + \text{RS}}$$

$$\text{RS} = \frac{n_{up}}{n_{down}}$$

$$n_{up} = \text{average gain of n} - \text{day}$$
$$n_{down} = \text{average loss of n} - \text{day}$$

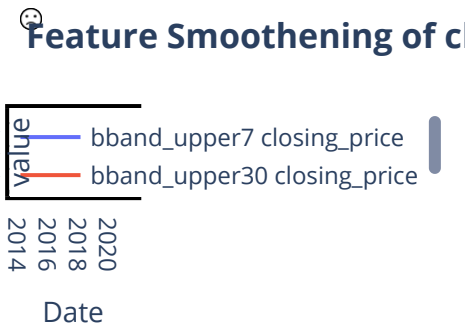
6.9 Rate of Change

Edit chart
Loading graph

Measures the percentage change in price between the current feature value and the feature a certain number of periods ago. A rising ROC above zero typically confirms an uptrend, while a falling ROC below zero indicates a downtrend.

6.10 Bollinger Bands

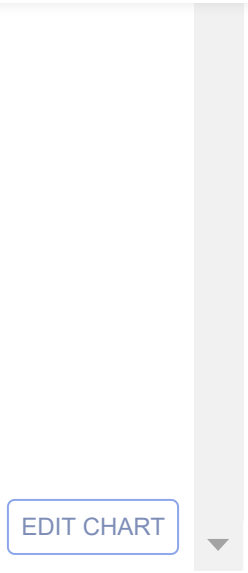
EDIT CHART



Bollinger Bands are envelopes (Upper and Lower range levels) plotted at a standard deviation level above and below a simple moving average of the price. Because the distance of the bands is based on standard deviation, they adjust to volatility swings in the underlying price. Bollinger bands help determine whether values are high or low on a relative basis. They are used in pairs, both upper and lower bands, and in conjunction with a moving average.

$$\text{Upper band} = \text{SMA}_{\text{n\_day}} + 2(\text{SD}_{\text{n\_day}})$$
$$\text{Lower band} = \text{SMA}_{\text{n\_day}} - 2(\text{SD}_{\text{n\_day}})$$

6.11 Moving Average Convergence Divergence



The MACD represents a trend following indicator that highlights whether the short-term price momentum is moving in the same direction as the long-term price momentum, and in cases where it's not, then it's used to determine if a trend change is near. The MACD consists of four components.

MACD line- shows the variation between the slow-moving average and the fast-moving average

Signal line-for signaling fluctuations in price momentum

After feature engineering, total number of features increased to 851.

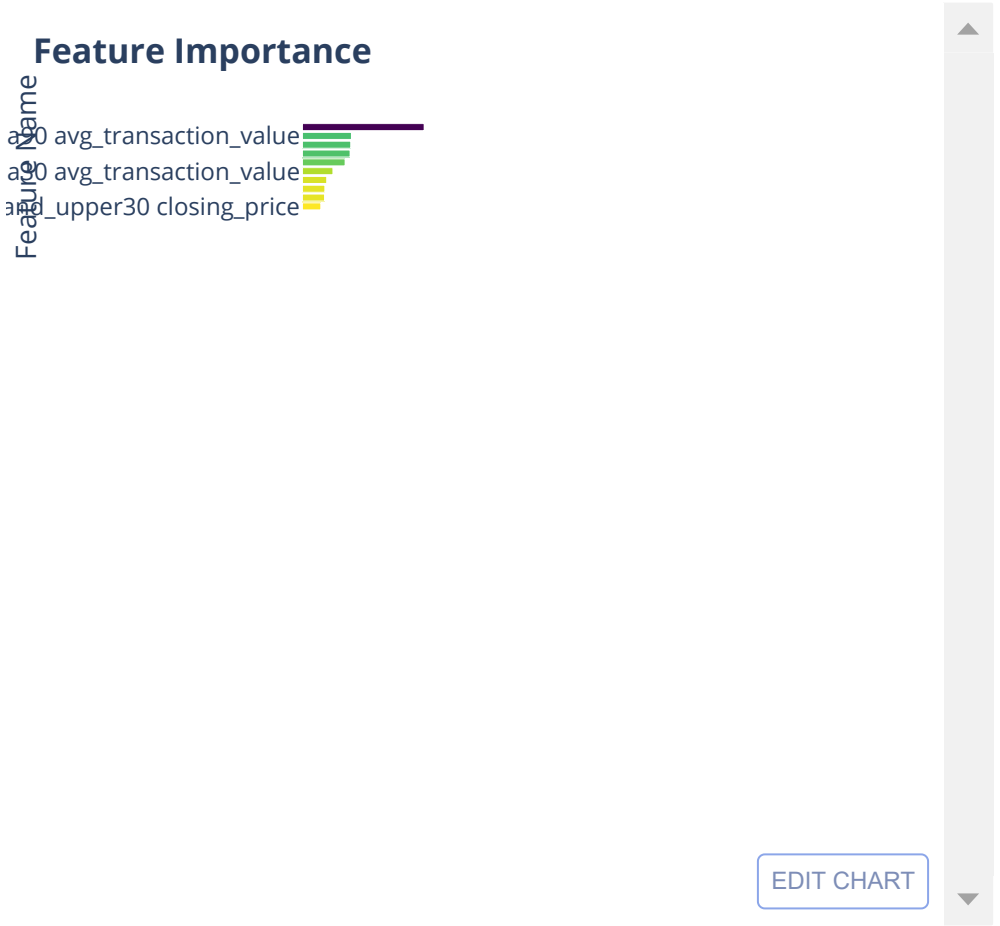
7. Feature Scaling

For scaling the features, robust scalar followed by min-max scalar was used. Robust scalar is similar to min max scalar, but it uses the interquartile range, so that it is robust to outliers.

$$\text{robust scalar}(X) = \frac{x_i - \text{Median}(X)}{\text{IQR}(X)}$$

$$\text{MinMax scalar}(X) = \frac{x_i - \text{Min}(X)}{\text{Max}(X) - \text{Min}(X)}$$

important features.



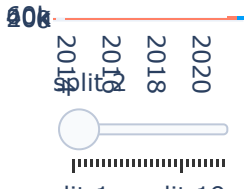
split cross validation, but results were mostly overfitting.

Thus, we decided to create multiple splits and train model for each split separately. Each train Split will consist of 500 data points & next 100 data points will be used for testing. E.g., If there are 10 splits, 10 models will be trained and each model will predict 100 data points. To summarize, prediction of the next 100 days will be based on data considered of last 500 days. The final metric will be mean over metric reported by each split.

```
train_window = 500
test_window = 100
train_splits = []
test_splits = []
for i in tqdm(range(train_window, len(final_df),test_window)):
    train_split = final_df[i-train_window:i]
    test_split = final_df[i:i+test_window]
    train_splits.append(train_split)
    test_splits.append(test_split)
```

train_test_split hosted with ❤ by GitHub view raw

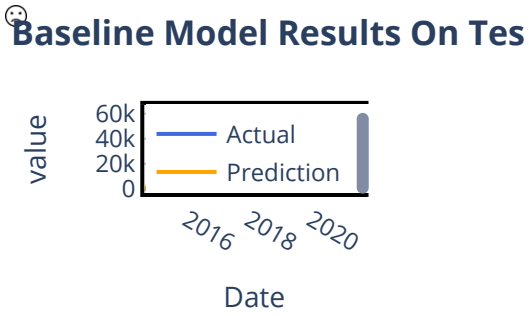

Multiple Train Test Split

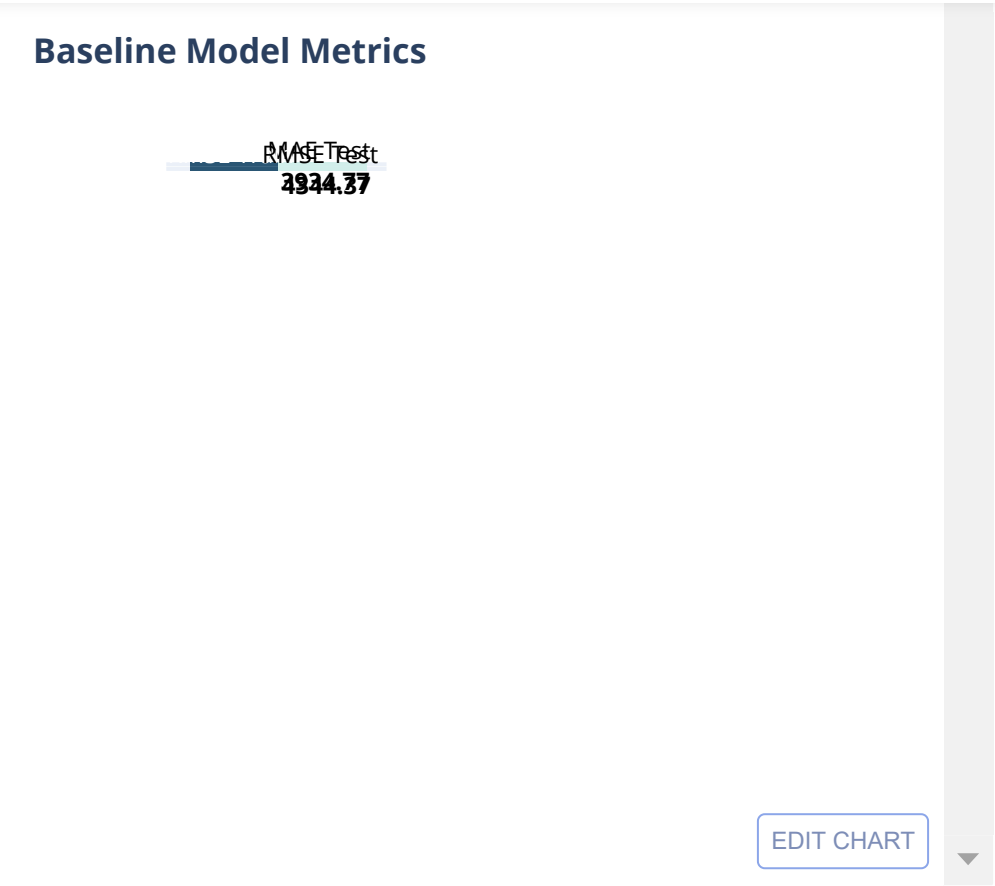


10. Model Training

10.1 Baseline Model

As a baseline, for each split, predictions were made using 90th percentile.





10.2 Support Vector Regression

Unlike Linear Regression models that try to minimize the error between the real and predicted value, the SVR tries to fit the best line within a threshold

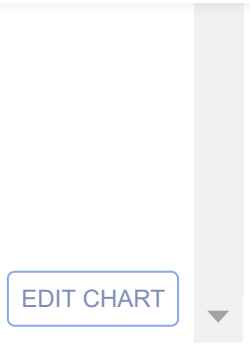
Here we have used SVR with Regularization parameter C =10000 with RBF Kernel with kernel coefficient gamma set to auto = 1 / n_features

```
for i in tqdm(range(len(train_splits))):
    Xtrain_split = train_splits[i].drop(['next_day_closing_price','Date'],axis=1).values
    Xtest_split = test_splits[i].drop(['next_day_closing_price','Date'],axis=1).values

    ytrain_split = train_splits[i]['next_day_closing_price'].reset_index(drop=True).values
    ytest_split = test_splits[i]['next_day_closing_price'].reset_index(drop=True).values

    svr = SVR(C=10000,gamma='auto',kernel='rbf')
    svr.fit(Xtrain_split, ytrain_split)

    ytrain_pred = svr.predict(Xtrain_split)
    ytest_pred = svr.predict(Xtest_split)

    MAE_train,RMSE_train = calculate_metrics(ytrain_split,ytrain_pred)
    MAE_test,RMSE_test = calculate_metrics(ytest_split,ytest_pred)

    print(f'MAE Train - {MAE_train} RMSE Train - {RMSE_train})
    print(f'MAE Test - {MAE_test} RMSE Test - {RMSE_test})
    print('*'*100)
```

svr_model hosted with ❤ by GitHub view raw

EDIT CHART

SVR Model Metrics



SVR is able to predict good results and capture the trend, even when there is a sudden spike in prices.

10.3 Extreme Gradient Boosting Regression

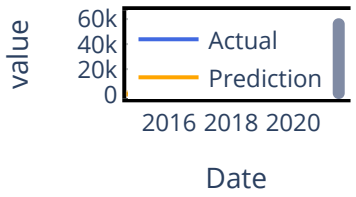
It is a boosting ensemble model which is constructed from decision tree models. Trees are added sequentially to the ensemble and fit to correct the prediction errors made by prior models.

Here we have used 500 trees which tries to minimize squared errors.

```
1   for i in tqdm(range(len(train_splits))):
```


```
7       xgbm_reg = xgb.XGBRegressor(n_estimators=500,max_depth=3,objective='reg:squarederror',
8                                   learning_rate =0.01,n_jobs=-1)
9       xgbm_reg.fit(Xtrain_split, ytrain_split)
10
11      ytrain_pred = xgbm_reg.predict(Xtrain_split)
12      ytest_pred = xgbm_reg.predict(Xtest_split)
13
14      MAE_train,RMSE_train = calculate_metrics(ytrain_split,ytrain_pred)
15      MAE_test,RMSE_test = calculate_metrics(ytest_split,ytest_pred)
16
17      print(f'MAE Train - {MAE_train} RMSE Train - {RMSE_train})
18      print(f'MAE Test - {MAE_test} RMSE Test - {RMSE_test})
19      print('*'*100)
```

xgb_model hosted with ❤ by GitHub view raw



EDIT CHART

XGBoost Model Metrics

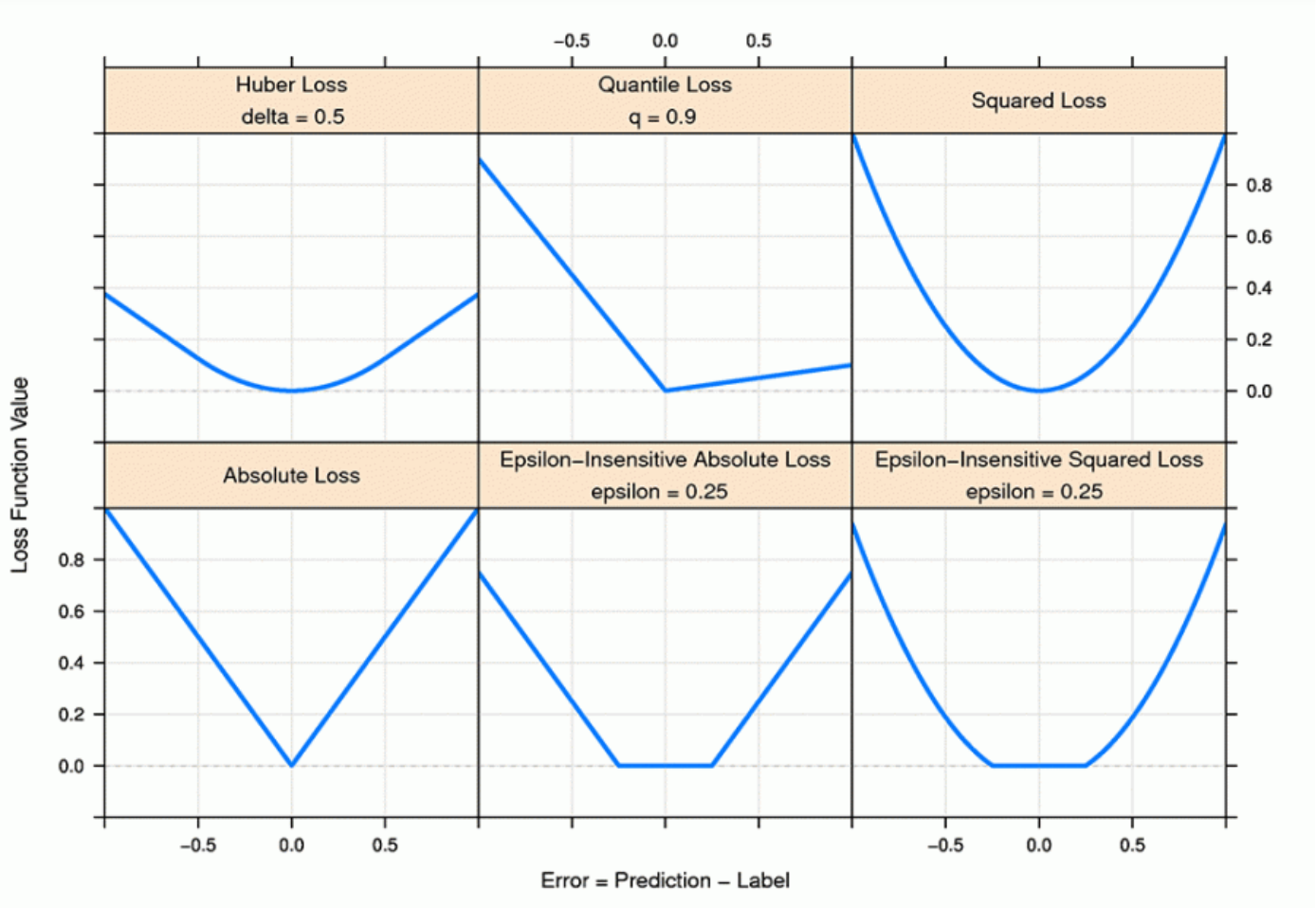
XGBoost is highly overfitting. We can observe that when there is a sudden spike in prices, It predicts unsatisfactory results.

10.4 Linear Regression using Stochastic gradient descent

Linear regression attempts to build a linear relationship between the input variables and target variable by fitting a linear equation to observed data by minimizing the least squares. It needs some regularization, as weights can grow exponentially to minimize errors.

We can use gradient descent to iteratively minimize the error of the model. It works by starting with random values for each coefficient. A learning rate is used as a scale factor, and the coefficients are updated in the direction towards minimizing the error. The process is repeated until a minimum sum squared error is achieved, or no further improvement is possible.

We are using squared epsilon insensitive as loss function which squares beyond ε-region where epsilon was set to 0.01. Elastic net was used for regularization. Initial learning rate was set to 0.01, and it was decreased

Comparison of various loss functions(Ref- https://aws.amazon.com/blogs/machine-learning/train-faster-more-flexible-models-with-amazon-sagemaker-linear-learner/)

```
for i in tqdm(range(len(train_splits))):
```

```
7       sgd_reg = SGDRegressor(loss='sq                    tive',alpha=0.0001,penalty='elasticnet',sh
8                              tol=0.000                   psilon=0.01,learning_rate='adaptive',max_it
9       sgd_reg.fit(Xtrain_split, ytrain_split)
10
11      pickle.dump(sgd_reg, open(f'/content/drive/MyDrive/Self Case studies/CS01 Bitcoin Price Foreca
12
13      ytrain_pred = sgd_reg.predict(Xtrain_split)
14      ytest_pred = sgd_reg.predict(Xtest_split)
15
16      MAE_train,RMSE_train = calculate_metrics(ytrain_split,ytrain_pred)
17      MAE_test,RMSE_test = calculate_metrics(ytest_split,ytest_pred)
18
19      print(f'MAE Train - {MAE_train} RMSE Train - {RMSE_train})
20      print(f'MAE Test - {MAE_test} RMSE Test - {RMSE_test})
21      print('*'*100)
```

linear regression hosted with ❤ by GitHub view raw

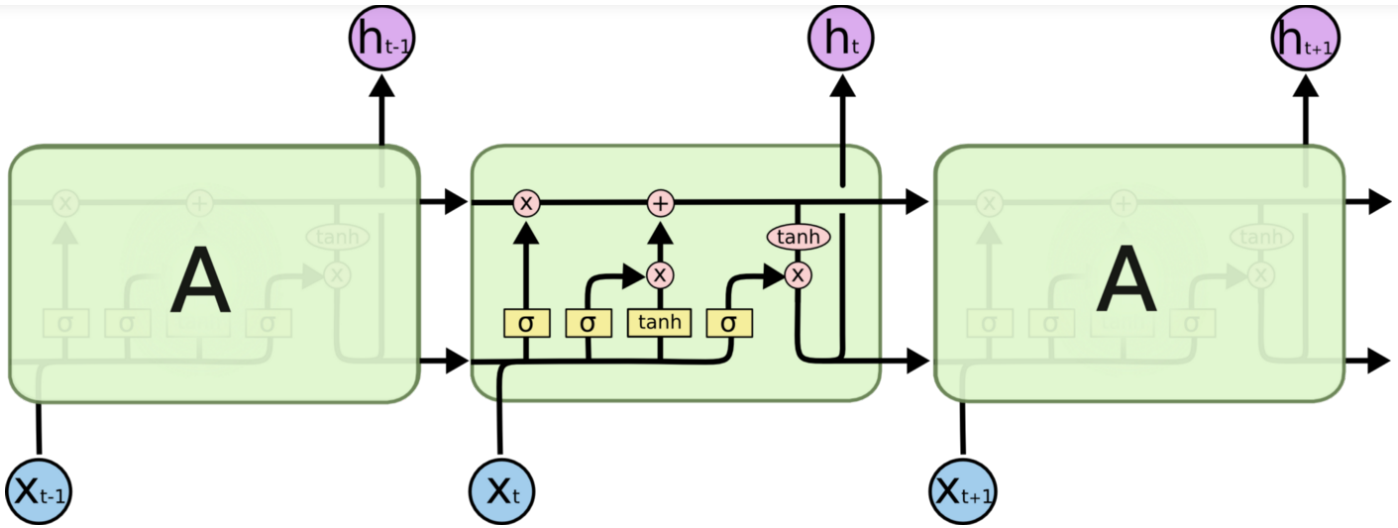
EDIT CHART

Linear Regression Model Metri

This model performs significantly better than SVR and XGBoost. As observed earlier in correlation, there was a very good linear relationship between some features and target variable. This ultimately has made linear models like SVR & Linear regression fit a good hyperplane on data.

10.5 Long Short-Term Memory (LSTM)

Long Short-Term Memory networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems. It has the ability to remove or add information to the cell state, carefully regulated by structures called gates. Gates are a way to optionally let information through. They are composed out of a sigmoid neural net layer and a point wise multiplication operation.

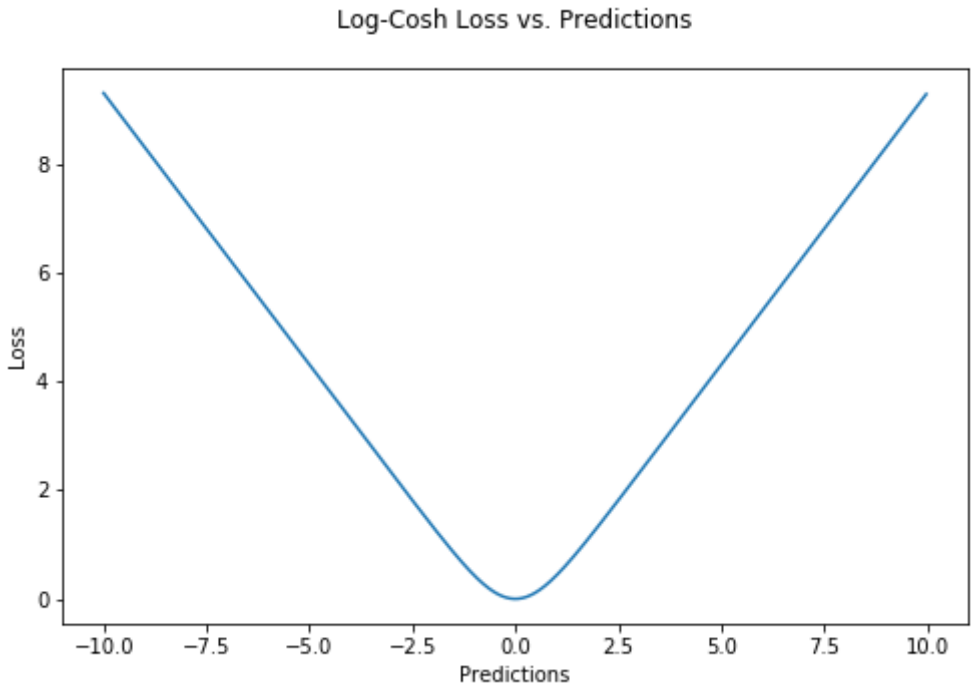



LSTM (Ref-https://colah.github.io/posts/2015-08-Understanding-LSTMs/)

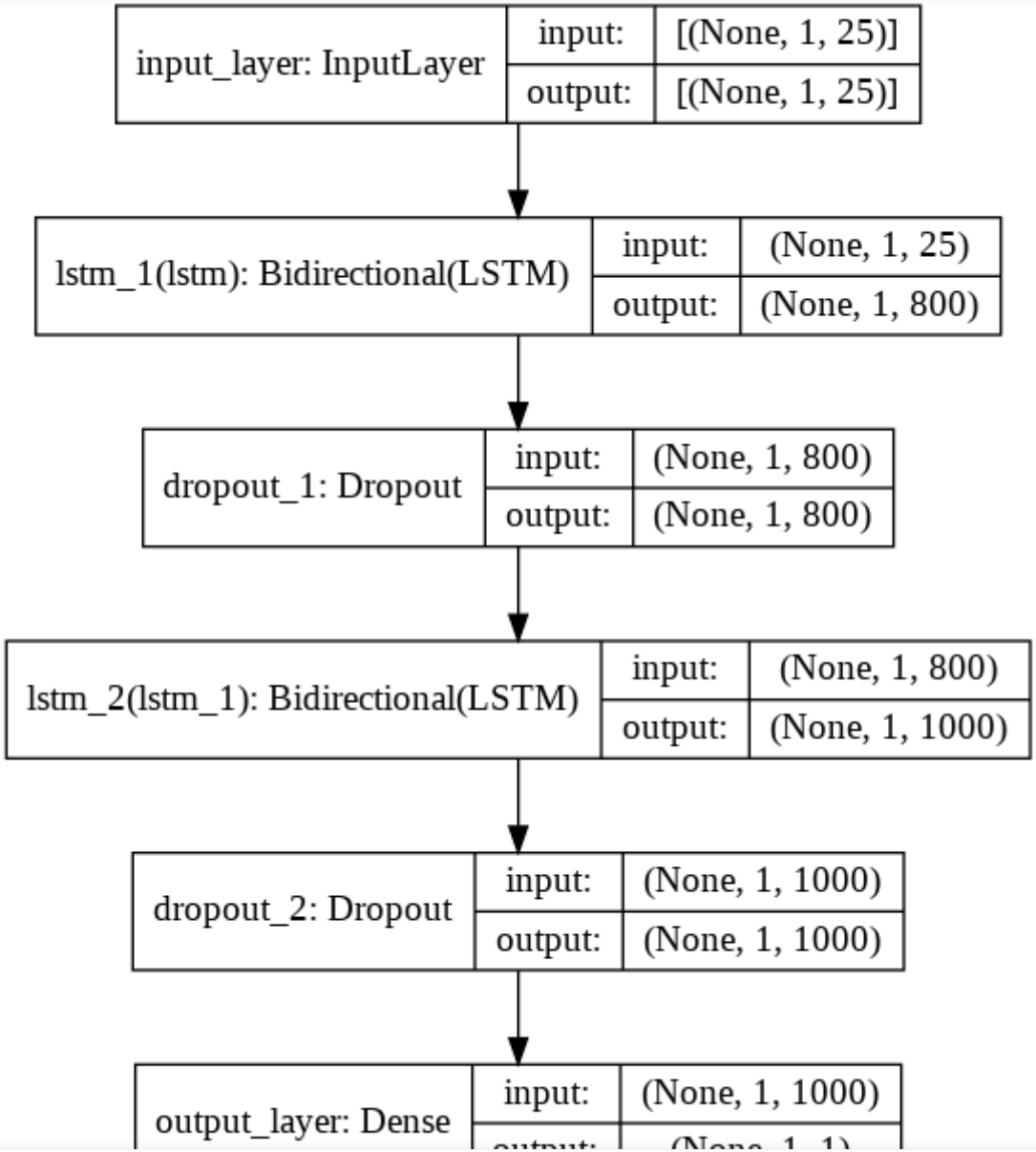
We are feeding input to a bidirectional layer of 400 cells, followed by a dropout of 25%. This output is fed to a bidirectional layer of 500 cells, which is followed by dropout of 30%. RELU activation function was used to avoid vanishing or exploding gradient problem. Adam optimizer was used for updating the weights.

The loss function used is the logarithm of the hyperbolic cosine of the prediction error. `log(cosh(x))` is approximately equal to `(x ** 2)/2` for small `x`

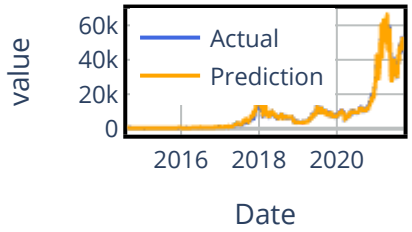
$$L(y, y^p) = \sum_{i=1}^{n} \log(\cosh(y_i^p - y_i))$$



Logcosh Loss Function(Ref-https://heartbeat.fritz.ai/5-regression-loss-functions-all-machine-learners-should-know-4fb140e9d4b0)


input_layer: InputLayer
input: [(None, 1, 25)]
output: [(None, 1, 25)]
lstm_1(lstm): Bidirectional(LSTM)
input: (None, 1, 25)
output: (None, 1, 800)
dropout_1: Dropout
input: (None, 1, 800)
output: (None, 1, 800)
lstm_2(lstm_1): Bidirectional(LSTM)
input: (None, 1, 800)
output: (None, 1, 1000)
dropout_2: Dropout
input: (None, 1, 1000)
output: (None, 1, 1000)
output_layer: Dense
input: (None, 1, 1000)

LSTM Results On Test Data



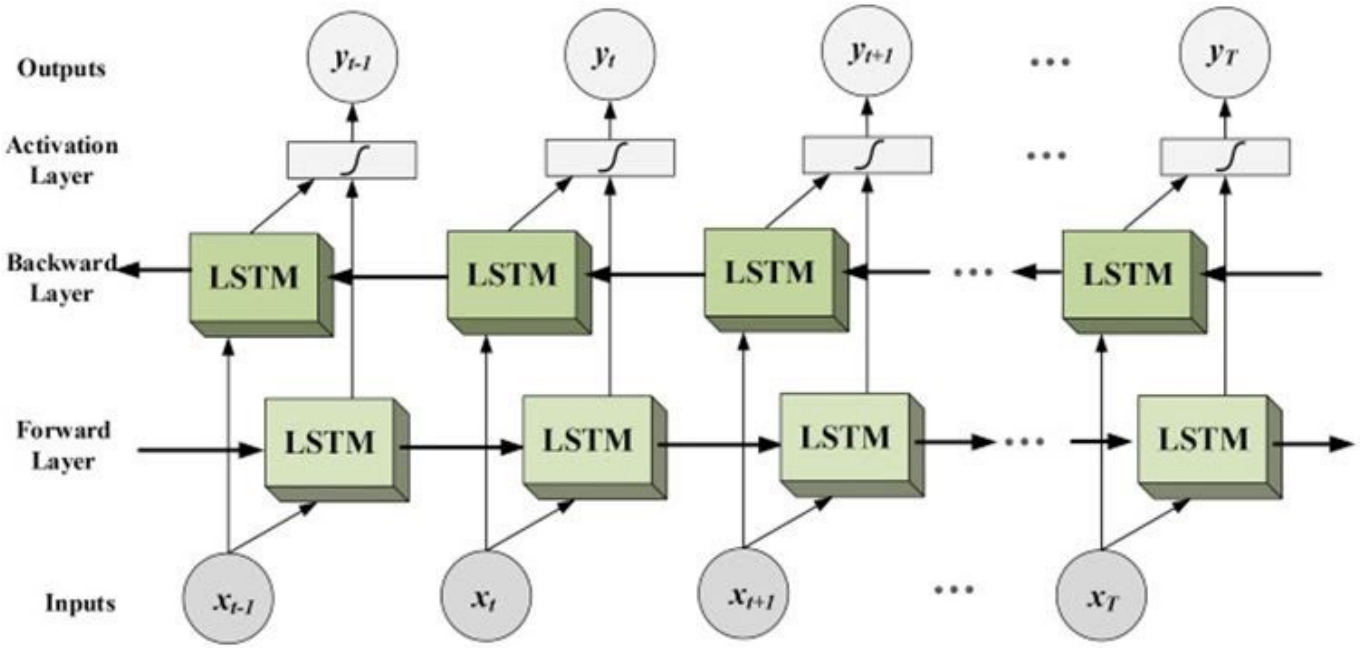
EDIT CHART

LSTM Model Metrics

EDIT CHART

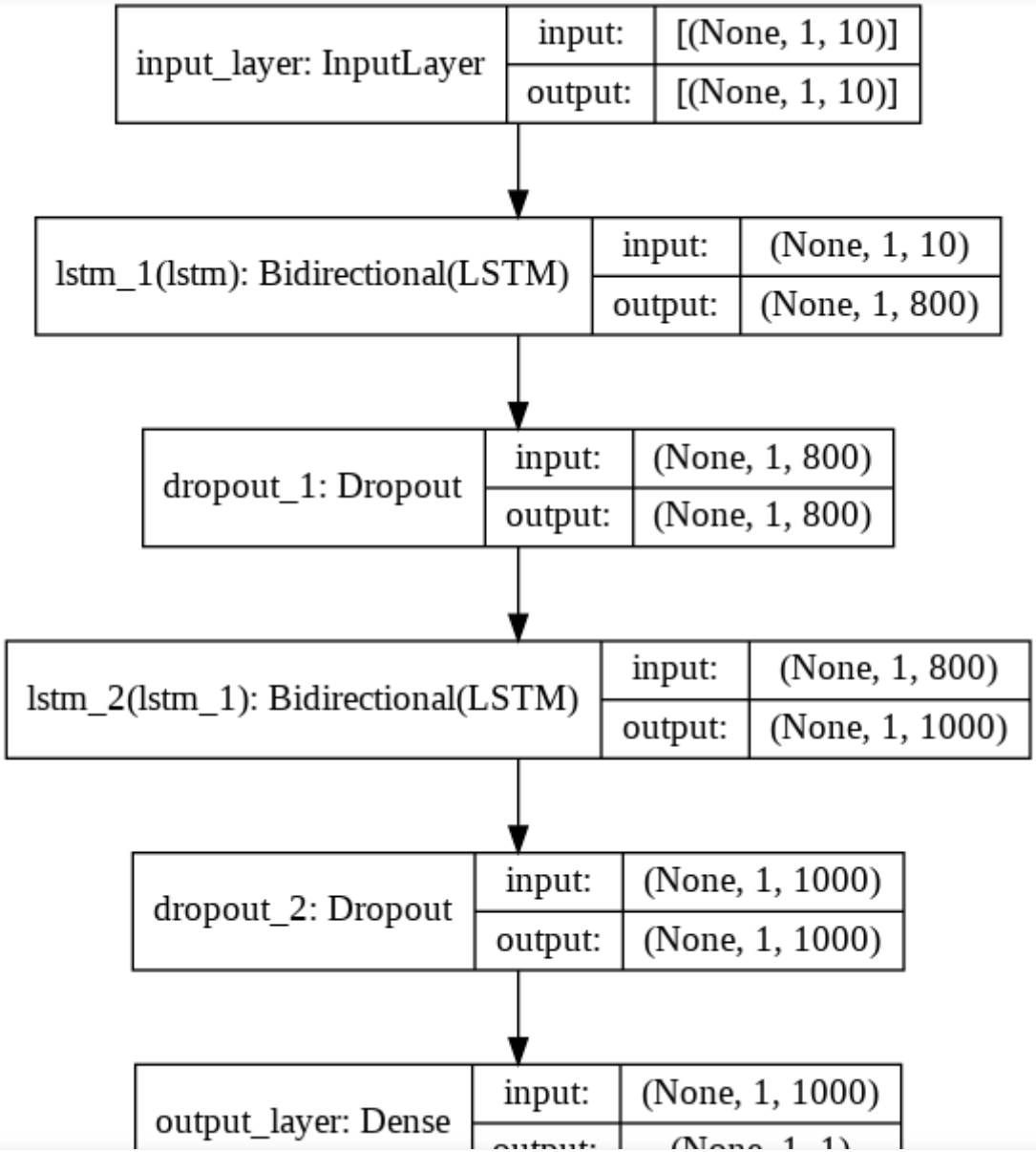
10.6 Bidirectional Long Short-Term Memory (LSTM)

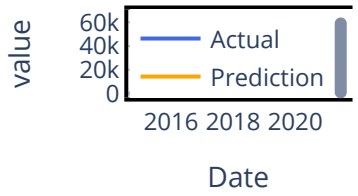
Bidirectional long-short term memory (bi-lstm) is the process of making neural network to have the sequence information in both directions backwards (future to past) or forward (past to future).



Bidirectional LSTM(Ref-https://analyticsindiamag.com/complete-guide-to-bidirectional-lstm-with-python-codes/)

The model architecture was kept same as LSTM with LSTM cells replaced as


input_layer: InputLayer
input: [(None, 1, 10)]
output: [(None, 1, 10)]
lstm_1(lstm): Bidirectional(LSTM)
input: (None, 1, 10)
output: (None, 1, 800)
dropout_1: Dropout
input: (None, 1, 800)
output: (None, 1, 800)
lstm_2(lstm_1): Bidirectional(LSTM)
input: (None, 1, 800)
output: (None, 1, 1000)
dropout_2: Dropout
input: (None, 1, 1000)
output: (None, 1, 1000)
output_layer: Dense
input: (None, 1, 1000)



EDIT CHART

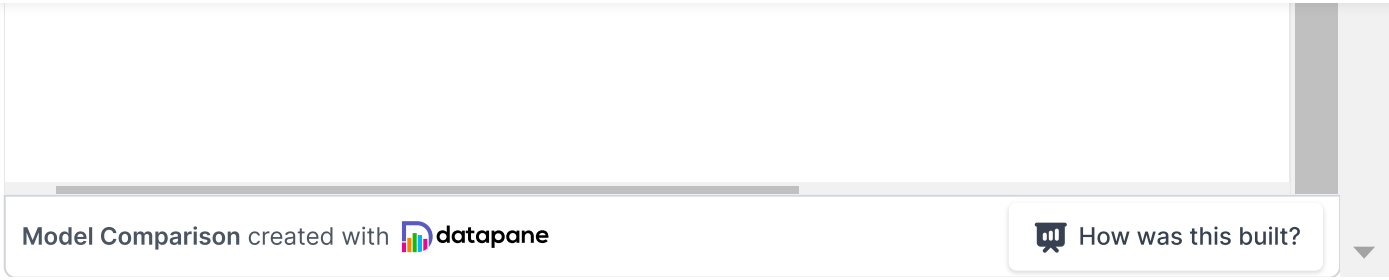
BI-LSTM Model Metrics

EDIT CHART

10.7 Comparison of all models

1 1 1 Explore data Run SQL Query Export

	Model	MAE Train	MAE Test	
1	Support Vector Regression	246.5	769	3
2	Linear Regression	244.3	571.7	3
3	XGBoost	107.1	1510	1
4	LSTM	169.2	468.9	2
5	BI-LSTM	159.6	435.1	2

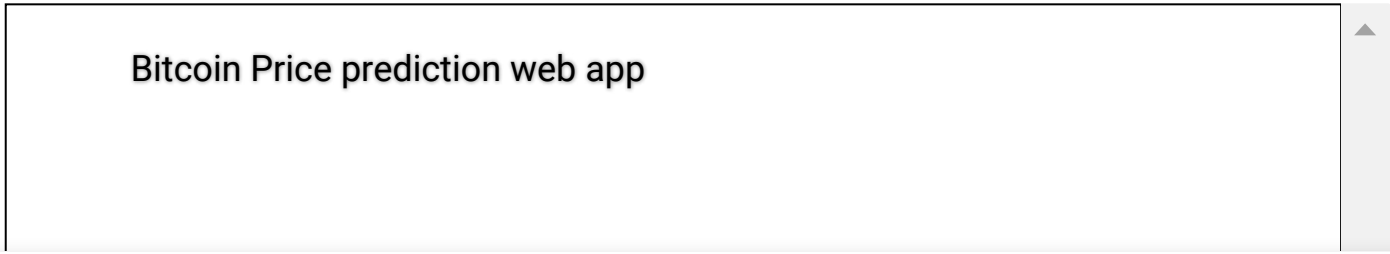


Linear regression & LSTM making good predictions. Due to test time complexity benefits, we will deploy linear regression model.

11. Model Deployment

Model was deployed using linear regression.

Bitcoin Price Prediction



Github Repository-

GitHub - rohansawant7978/bitcoin-price-forecasting

Contribute to rohansawant7978/bitcoin-price-forecasting development by creating an account on GitHub.

github.com

LinkedIn Profile-

Rohan Sawant - Data Scientist - Krisopia | LinkedIn

A data science professional with the skill to analyze and solve real world industry problems. Ability to understand and

12. References

1. https://www.appliedaicourse.com/
2. https://link.springer.com/content/pdf/10.1007/s00521-020-05129-6.pdf
3. https://www.stilt.com/blog/2021/07/how-does-cryptocurrency-gain-value/
4. https://theblockpro.com/what-is-bitcoin/
5. https://bitinfocharts.com/
6. https://www.investing.com/
7. https://data.nasdaq.com/data/BCHAIN/MKPRU-bitcoin-market-price-usd
8. https://top10stockbroker.com/cryptocurrency/bitcoin-technical-indicators/
9. https://www.mycryptopedia.com/best-8-bitcoin-indicators-for-cryptocurrency-trading/
10. https://towardsdatascience.com/unlocking-the-true-power-of-support-vector-regression-847fd123a4a0
11. https://machinelearningmastery.com/xgboost-for-regression/
12. https://machinelearningmastery.com/linear-regression-for-machine-

14. https://colah.github.io/posts/2015-08-Understanding-LSTMs/
15. https://analyticsindiamag.com/complete-guide-to-bidirectional-lstm-with-python-codes/
16. https://heartbeat.fritz.ai/5-regression-loss-functions-all-machine-learners-should-know-4fb140e9d4b0
17. https://www.investopedia.com/

13.Future Work

We can attempt to predict 7th & 15th & 30th day price by using similar approach. Similar to original reference, we can attempt to predict increase or decrease price.