A Comparison of Regression Techniques Using Bitcoin Price Data

Vai Suliafu, Ambuj Arora, Yicong Xiao

Context & Motivation

Bitcoin is a non-fiat, sovereign digital currency traded bilaterally via online peer-to-peer exchanges. Historically, crypto-currency exchanges have not required market participants to possess professional trading licenses and certifications such as those required by FINRA and the SEC in more mature financial markets. As a result, bitcoin markets lack the presence of institutional investors and their tremendous resources, giving average market participants greater opportunities for the discovery of alpha using advanced algorithmic techniques such as those from Machine Learning and Data Mining.

Thus, it was our goal to determine just how well commonly used regression techniques performed in the context of bitcoin price prediction.

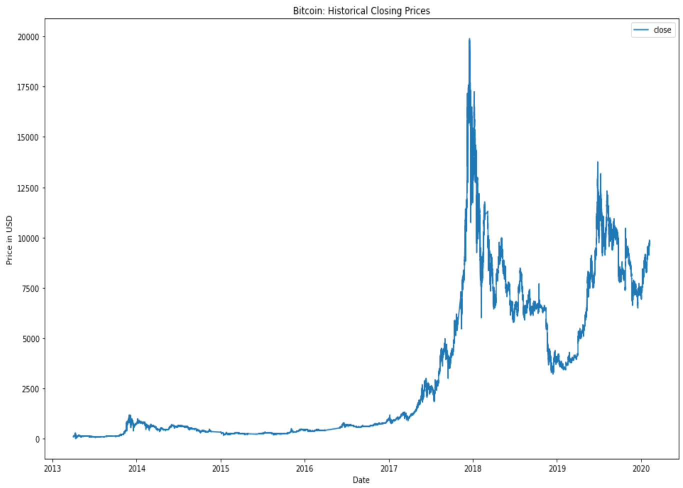
Data Collection

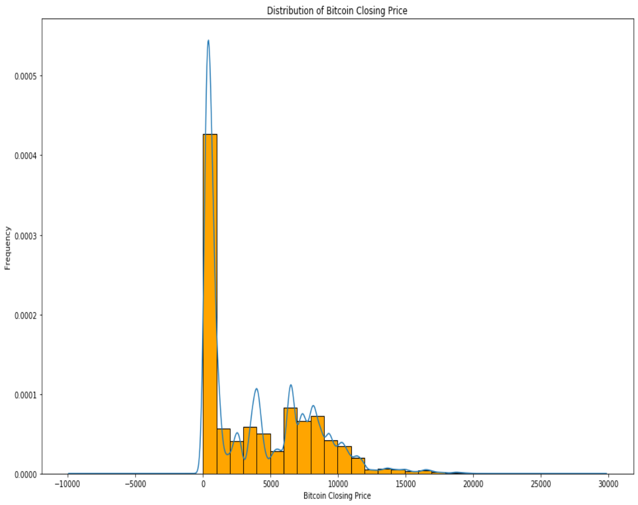
For our analysis, we utilized minute level bitcoin pricing data, originally scraped from the crypto-currency exchange BitFinex and presently hosted and maintained on Kaggle for exploratory analysis and discovery.

This data consisted of 2,638,113 observations, spanning from 4/01/2013 at 06:56:00 to 2/07/2020 at 18:22:00.

Surface Level Analysis of Price Charts

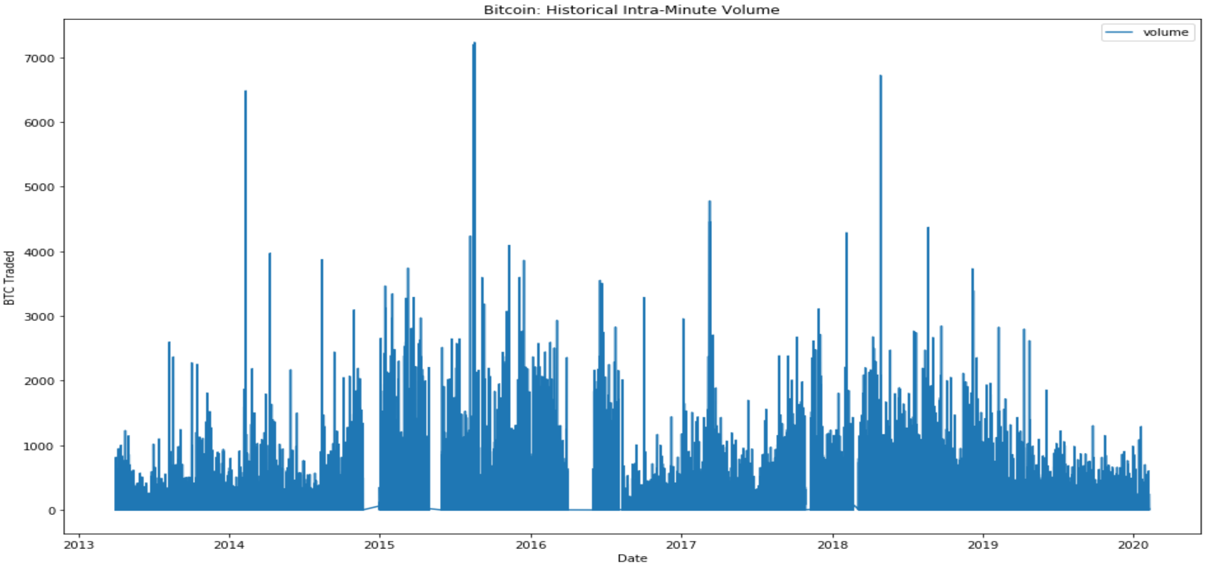
- The all-time low occurred on 4/11/2013 at 18:12:00 with a price of $1.06
- The all-time high occurred on 12/17/2017 at 12:12:00 with a price of $19,891.00
- Buying at the all-time low and selling at the all-time high represents a gain of 1,876,509.43%
- The distribution of prices is heavily right-skewed, with the large plurality being < $1.00





Surface Level Analysis of Volume Charts

- Average intra minute trading volume is 19.81 BTC
- The all-time high occurred on 8/18/2015 at 20:10:00 with 7228.08 BTC traded
- There are several periods where trading volume reaches near zero, most likely resulting from maintenance periods of the BitFinex exchange
- Volume has been down-trending since early 2018 all-time high in price



Data Preprocessing

- Timestamp Conversion: The UNIX timestamp was converted to DateTime format using Pandas' DateTime function
- Feature Creation: 68 additional technical analysis features such as EMA, MACD, RSI, PSAR, etc. were derived from the original OHLC data using the TA Python package
- Null Value Treatment: The additionally derived features required at most 43 initializing observations in order to calculate their values. Thus, we have dropped these initial 43 observations since many of the features could not yet be calculated and instead were null. Moreover, three of the additional features created by the TA package were composed entirely of null values. Thus, the features 'trend_psar_down', 'trend_psar_up' and 'trend_cci' were deleted entirely from the data
- Target Feature Creation: We created the target feature, 'nextClosingPrice' by shifting the closing price of the following observation down 1 level. Thus the target variable 'nextClosingPrice' at some time *t* is an identical value to the feature 'close' at some time *t+1*.
- Final dimensions of the data after preprocessing: **721038 x 63**

Key Idea

In order to analyze the performance of various regression models, we needed to devise a scheme for training each regression model. While methods such as cross validation and shuffled splitting are commonly used in this regard, such methods are not applicable to time series data due to the inherent assumption that observations in the data are independent of one another, which is false in the case of financial time series. Moreover, our initial exploration of the data suggested that due to the enormous Bitcoin price range and levels of volatility, models trained on data from the past would not prove effective in predicting future prices.

Thus, we sought to compare both the effectiveness and the cost of offline regression models trained on some past fixed portion of data versus online learning models, trained solely on some fixed number of previous observations *n*.

Offline Training

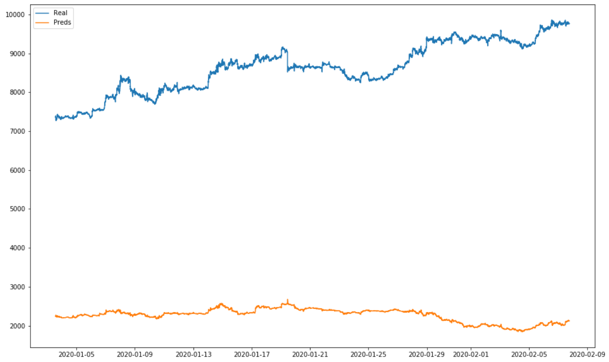
Training Procedure:

For each of the regression models, the following training procedure was followed:

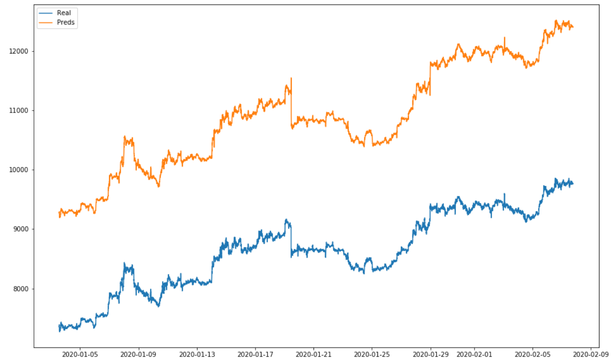
1. Separate the target feature ‘nextClosingPrice’ from the data
2. Let the first 2,588,440 observations be the training data
3. Let the first 2,588,440 values of ‘nextClosingPrice’ be the training target values
4. For the remaining 50,000 observations re-indexed from 0 to 49,999, drop the rows corresponding to odd indices (the subset should now be of length 25,000 instead of 50,000 - we do this to cover more ground with our tests while keeping training costs reasonable)
5. Repeat step 4 with the ‘nextClosingPrice’ series
6. Instantiate and ‘out-of-the-box’ regression model object and fit it using the data prepared in steps 2 and 3
7. Use the now trained model to generate predictions using the 25,000 rows of data held out from training
8. Compare the predictions to the observed target values from step 5 and calculate the average root mean square error

Offline Results:

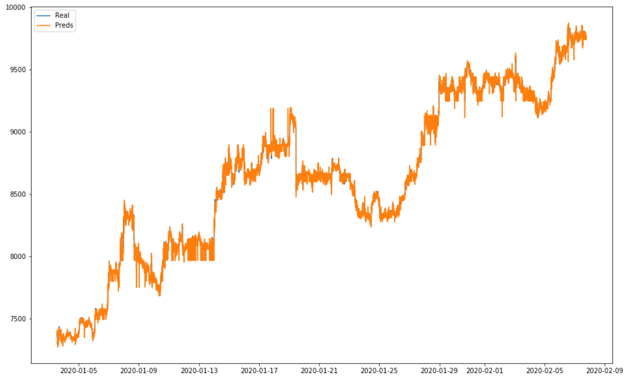
Lasso Regression



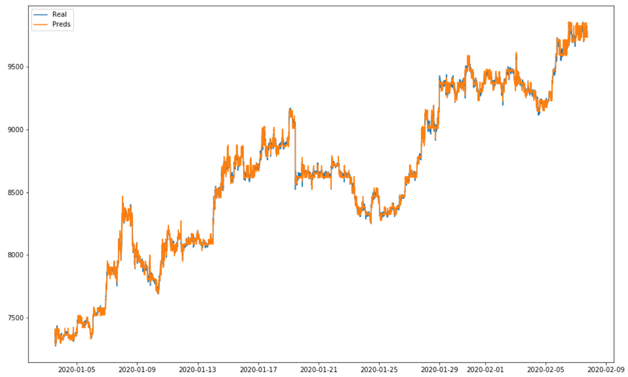
Ridge Regression



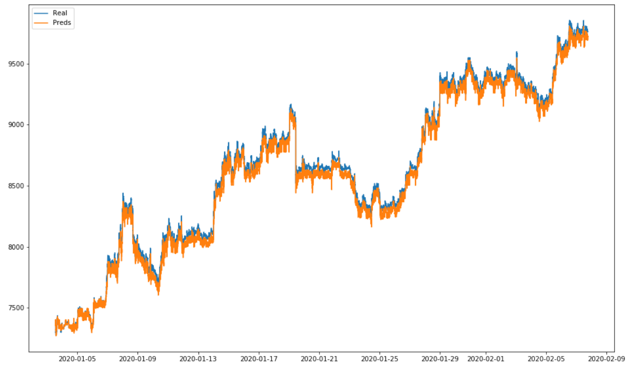
Decision Tree Regression



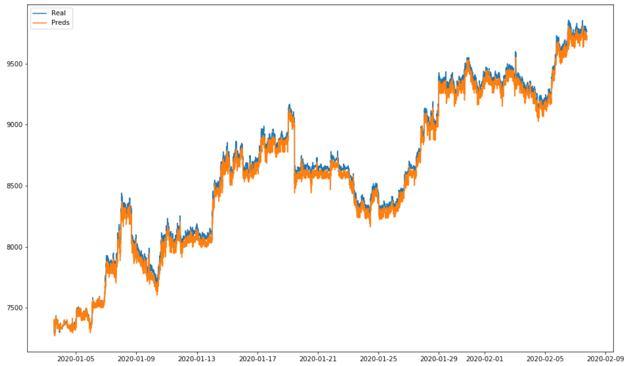
LightGBM Regression



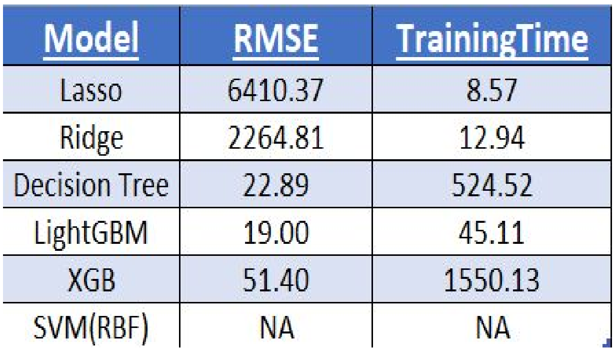
XGB Regression



SVM (RBF) Regression

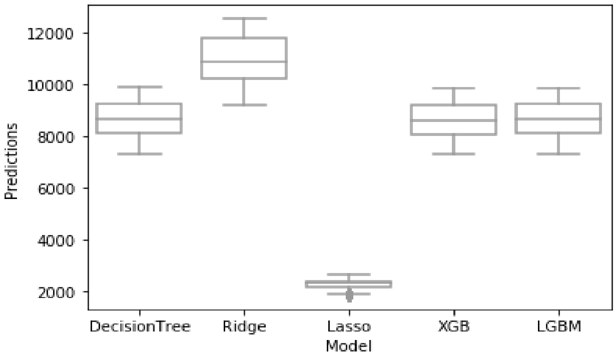


RMSE Summary



Model	RMSE	TrainingTime
Lasso	6410.37	8.57
Ridge	2264.81	12.94
Decision Tree	22.89	524.52
LightGBM	19.00	45.11
XGB	51.40	1550.13
SVM(RBF)	NA	NA

Distribution of Predicted Values



Online Training

Training Procedure:

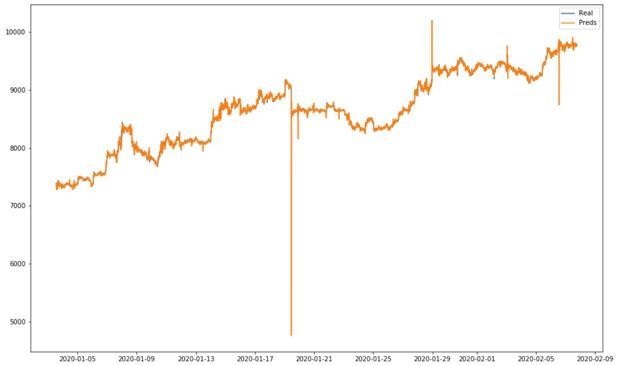
For each of the regression models, the following training procedure was followed:

1. Separate the target feature 'nextClosingPrice' from the data
2. Initialize a variable *k* such that it equals the index of the row 50,000th from the end (row 2,588,440)
3. For each *k* from 2,588,440 until the end of the data (incrementing by +2), do:
    a. For each value *w* in the set *W* = {60, 90, 120, 150}, do the following:
        i. Let the subset of observations from indices [*k*-*w*, *k*) be the set of training data
        ii. Let the subset of values in 'nextClosingPrice' from indices [*k*-*w*, *k*) be the training target values
        iii. Let the single observation at index *k* be the test training data
        iv. Let the single value at index *k* of 'nextClosingPrice' be the test target value
        v. Instantiate an 'out-of-the-box' regression model object and fit it using the data prepared in steps i and iii
        vi. Use the now trained model to generate a prediction for the target value at index *k* of 'nextClosingPrice', using the prepared test observation

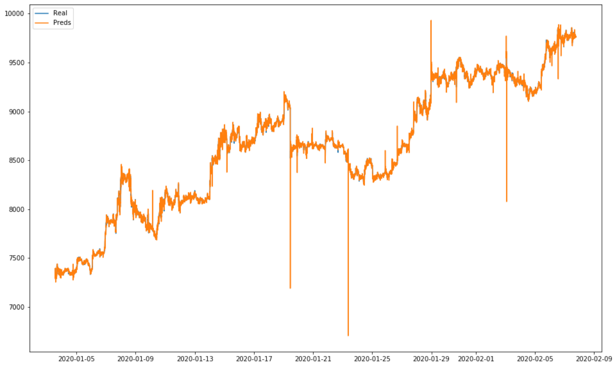
4. Compare all the predictions generated to the actual observed values from the same indices and calculate the average root mean square error

<u>Online Results (top performer only):</u>

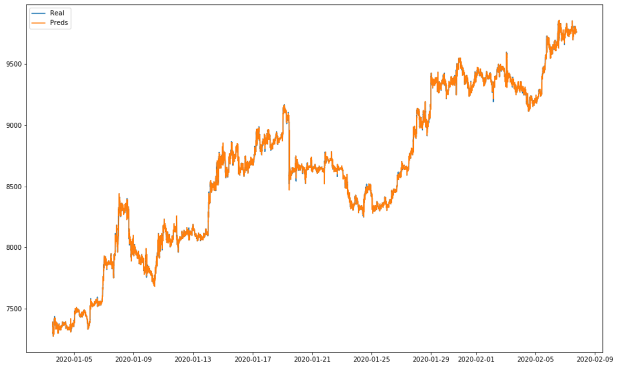
Lasso Regression (*w* = 60)



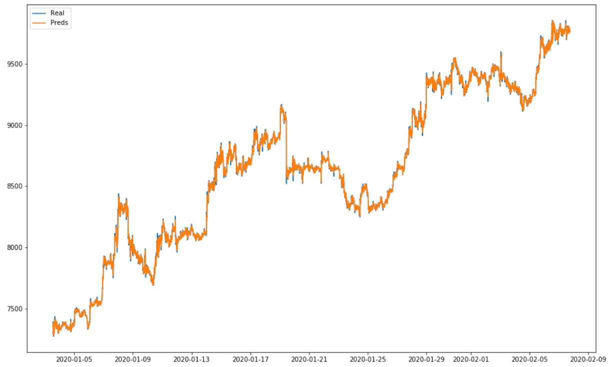
Ridge Regression (*w* = 150)



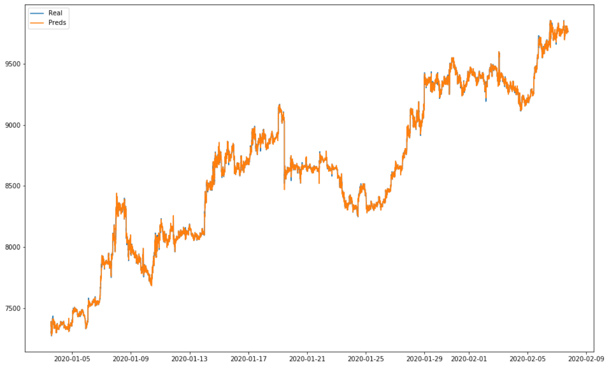
Decision Tree Regression (*w* = 60)



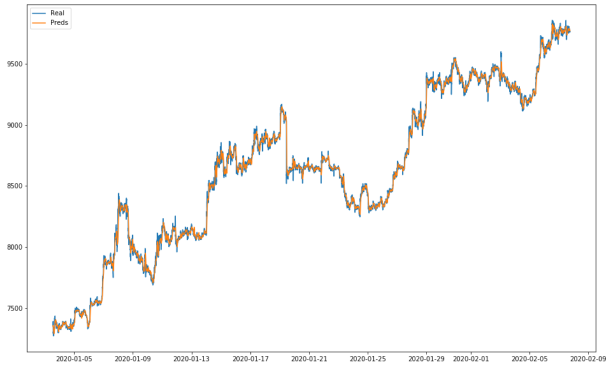
LightGBM Regression (*w* = 150)



XGB Regression (*w* = 60)

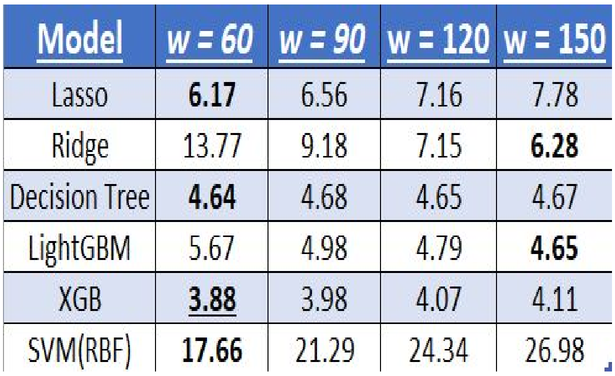


SVM (RBF) Regression (*w* = 60)



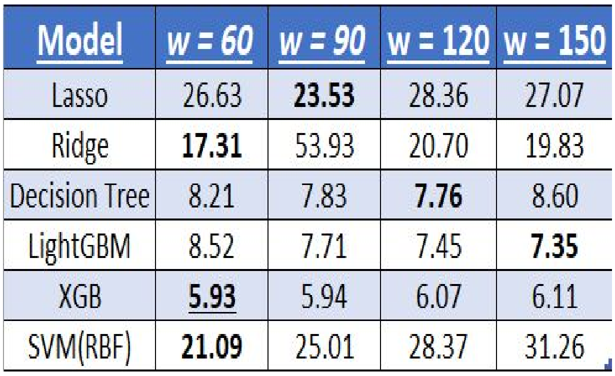
The results for each model with a different parameter w are shown below.

RMSE Mean Summary



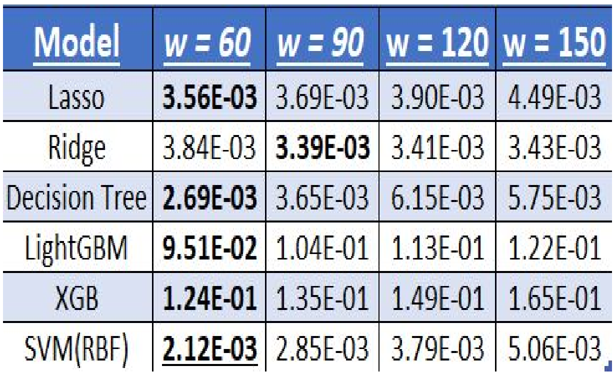
Model	w = 60	w = 90	w = 120	w = 150
Lasso	**6.17**	6.56	7.16	7.78
Ridge	13.77	9.18	7.15	**6.28**
Decision Tree	**4.64**	4.68	4.65	4.67
LightGBM	5.67	4.98	4.79	**4.65**
XGB	**3.88**	3.98	4.07	4.11
SVM(RBF)	**17.66**	21.29	24.34	26.98

RMSE SD Summary



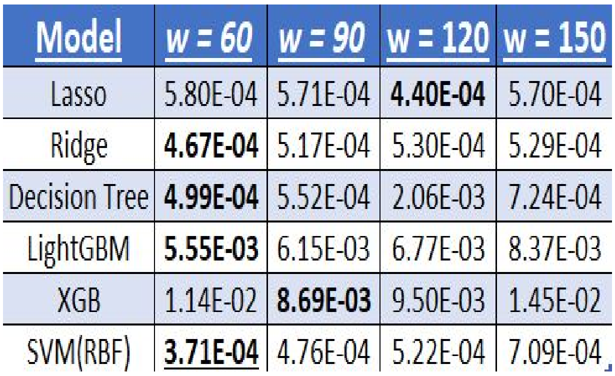
Model	w = 60	w = 90	w = 120	w = 150
Lasso	26.63	**23.53**	28.36	27.07
Ridge	**17.31**	53.93	20.70	19.83
Decision Tree	8.21	7.83	**7.76**	8.60
LightGBM	8.52	7.71	7.45	**7.35**
XGB	**5.93**	5.94	6.07	6.11
SVM(RBF)	**21.09**	25.01	28.37	31.26

TrainingTime Mean Summary



Model	w = 60	w = 90	w = 120	w = 150
Lasso	**3.56E-03**	3.69E-03	3.90E-03	4.49E-03
Ridge	3.84E-03	**3.39E-03**	3.41E-03	3.43E-03
Decision Tree	**2.69E-03**	3.65E-03	6.15E-03	5.75E-03
LightGBM	**9.51E-02**	1.04E-01	1.13E-01	1.22E-01
XGB	**1.24E-01**	1.35E-01	1.49E-01	1.65E-01
SVM(RBF)	**2.12E-03**	2.85E-03	3.79E-03	5.06E-03

TainingTime SD Summary

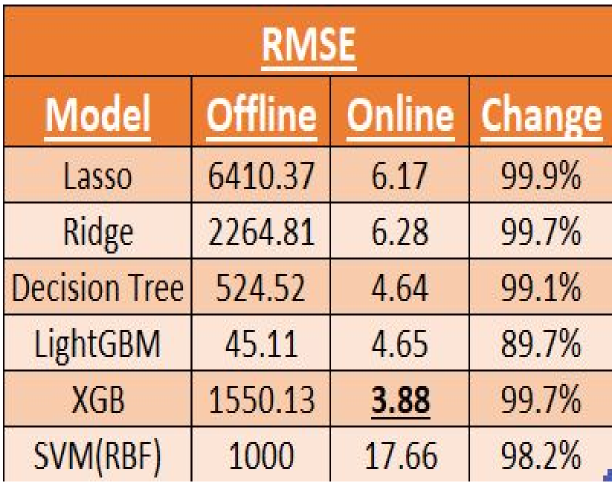


Model	w = 60	w = 90	w = 120	w = 150
Lasso	5.80E-04	5.71E-04	**4.40E-04**	5.70E-04
Ridge	**4.67E-04**	5.17E-04	5.30E-04	5.29E-04
Decision Tree	**4.99E-04**	5.52E-04	2.06E-03	7.24E-04
LightGBM	**5.55E-03**	6.15E-03	6.77E-03	8.37E-03
XGB	1.14E-02	**8.69E-03**	9.50E-03	1.45E-02
SVM(RBF)	**3.71E-04**	4.76E-04	5.22E-04	7.09E-04

Comparing Offline and Online

As shown to the right, the above experiments clearly indicate that for all of the tested models, the online learners dramatically outperform their offline counterparts.

Moreover, we can see that the XGBRegressor was the most powerful model in accurately predicting the future closing price of bitcoin, at the cost of being the slowest. We can also see that the support vector machine with radial basis function kernel performed the worst in generating accurate predictions and yet the best in terms of



RMSE			
Model	Offline	Online	Change
Lasso	6410.37	6.17	99.9%
Ridge	2264.81	6.28	99.7%
Decision Tree	524.52	4.64	99.1%
LightGBM	45.11	4.65	89.7%
XGB	1550.13	**3.88**	99.7%
SVM(RBF)	1000	17.66	98.2%

speed. We can also see that on average, non-linear, tree-based learners such as Decision Tree, LGBM and XGB outperformed the linear models, suggesting non linear separability within the data.

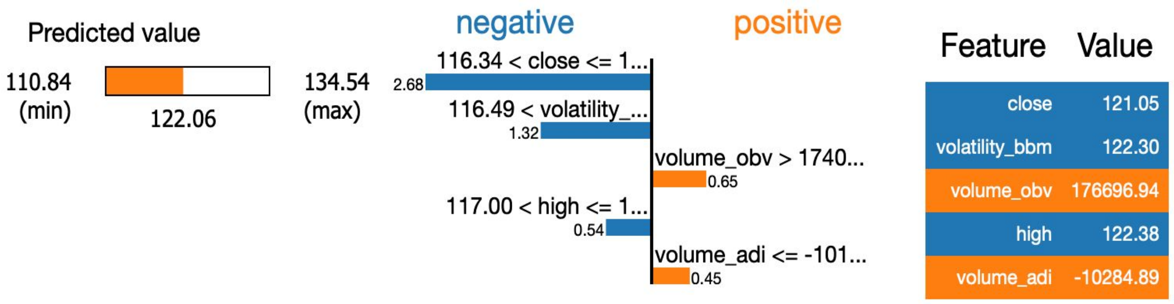
Analysing Feature Importances

In order to better understand the reasoning for such contrasting performance across regressors, we first wanted to understand which specific features each model considered most important. To this end, we utilized LIME.

LIME (Local Interpretable Model-Agnostic Explanations):

- LIME is a tool used for explaining the local behavior of a model ‘around’ the index of the instance being predicted
- LIME works by approximating the true underlying model locally with a more interpretable one
- LIME’s output consists of a ranked list of features and their corresponding importances, providing easy interpretability and understanding of models developed in training
- LIME is able to explain any model without needing to ‘peak’ in, making it a model-agnostic tool for assessment

Example:



Feature	Value
close	121.05
volatility_bbm	122.30
volume_obv	176696.94
high	122.38
volume_adi	-10284.89

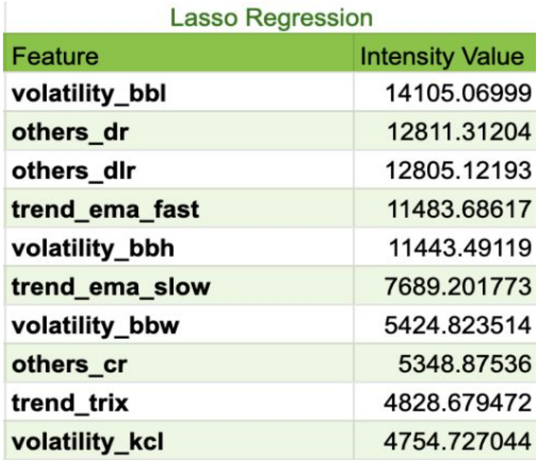
Shown above is one example of LIME being used to assess the local importance of features for a single prediction. A user can easily interpret the visual information to better understand a possible range of predicted values, the features ranked in order of significance as well as the actual direction of magnitude of impact for each feature.

For instance, the features close, volatility_bbm, volume_obv, high and volume_adi are the top 5 important features; with -2.68, -1.32, +0.65, -0.54 and +0.45 as their intensity values. A positive sign means that the feature had a positive impact on the prediction and a negative means it had a negative impact on the prediction.

Using LIME:

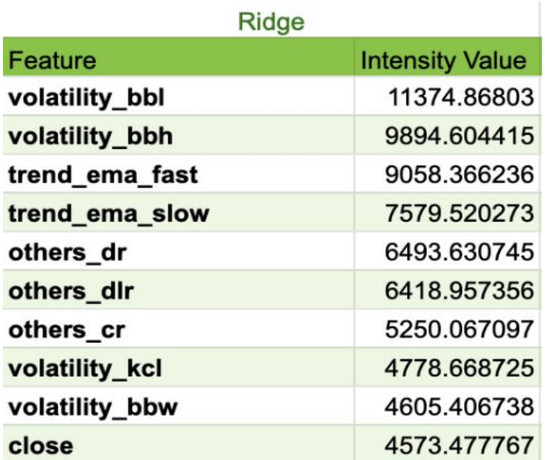
Since our experiments proved that online models with restricted training input were superior in every scenario, only the online models were analyzed using LIME. Moreover, since LIME can only explain the feature importances for a single predicted observation, a fresh, fully independent run of LIME for each and every observation used in testing was required. Then for each regressor model, we could iterate through the LIME output for each of the 25,000 test observations in order to calculate the average importance of all features across all 25,000 predictions for that particular model.

The results of LIME and the subsequent average of feature importances across all models are shown below.



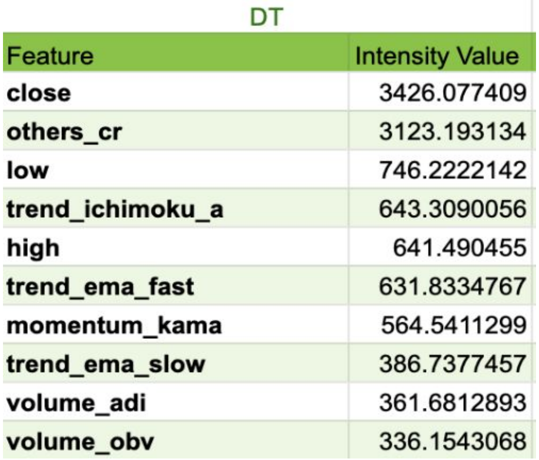
Lasso Regression

Feature	Intensity Value
volatility_bbl	14105.06999
others_dr	12811.31204
others_dlr	12805.12193
trend_ema_fast	11483.68617
volatility_bbh	11443.49119
trend_ema_slow	7689.201773
volatility_bbw	5424.823514
others_cr	5348.87536
trend_trix	4828.679472
volatility_kcl	4754.727044



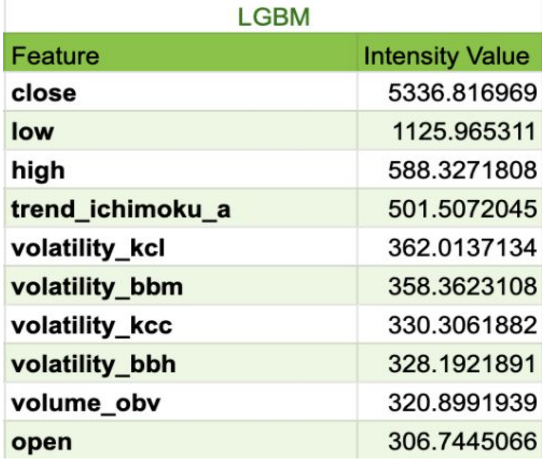
Ridge

Feature	Intensity Value
volatility_bbl	11374.86803
volatility_bbh	9894.604415
trend_ema_fast	9058.366236
trend_ema_slow	7579.520273
others_dr	6493.630745
others_dlr	6418.957356
others_cr	5250.067097
volatility_kcl	4778.668725
volatility_bbw	4605.406738
close	4573.477767



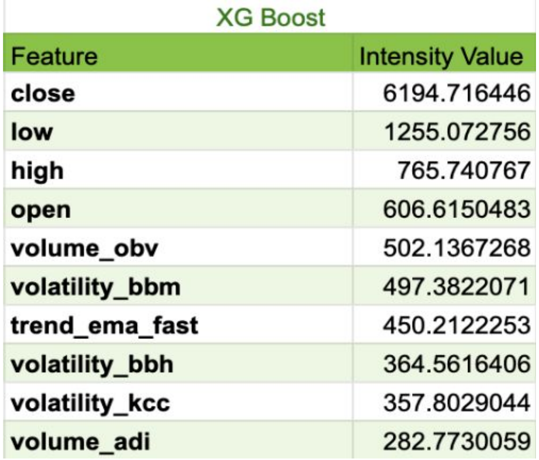
DT

Feature	Intensity Value
close	3426.077409
others_cr	3123.193134
low	746.2222142
trend_ichimoku_a	643.3090056
high	641.490455
trend_ema_fast	631.8334767
momentum_kama	564.5411299
trend_ema_slow	386.7377457
volume_adi	361.6812893
volume_obv	336.1543068



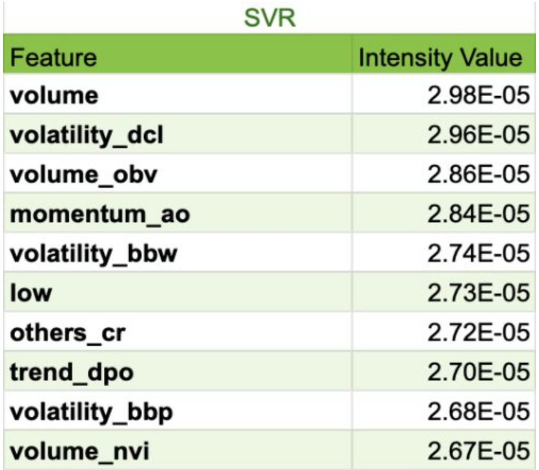
LGBM

Feature	Intensity Value
close	5336.816969
low	1125.965311
high	588.3271808
trend_ichimoku_a	501.5072045
volatility_kcl	362.0137134
volatility_bbm	358.3623108
volatility_kcc	330.3061882
volatility_bbh	328.1921891
volume_obv	320.8991939
open	306.7445066



XG Boost

Feature	Intensity Value
close	6194.716446
low	1255.072756
high	765.740767
open	606.6150483
volume_obv	502.1367268
volatility_bbm	497.3822071
trend_ema_fast	450.2122253
volatility_bbh	364.5616406
volatility_kcc	357.8029044
volume_adi	282.7730059



SVR

Feature	Intensity Value
volume	2.98E-05
volatility_dcl	2.96E-05
volume_obv	2.86E-05
momentum_ao	2.84E-05
volatility_bbw	2.74E-05
low	2.73E-05
others_cr	2.72E-05
trend_dpo	2.70E-05
volatility_bbp	2.68E-05
volume_nvi	2.67E-05

LIME Analysis Takeaways:

- The most significant explainers of Bitcoin Price are the features 'close', which represent the previous closing price and 'volatility_bbl', which represents the lower level of the Bollinger Bands technical indicator

- The linear regression models (namely the Lasso and Ridge Regressors) which considered 'volatility_bbl' overfit our data with out of the box parameters

    - Evidence of this can be seen on page 5 when viewing the predictions of each model. It's clear that both Lasso and Ridge regression had a handful of outlier predictions, which were almost infinitely far from the true observed value

    - Additionally, the table of standard deviations for RMSE support that these outlier predictions severely skewed the average performance metrics

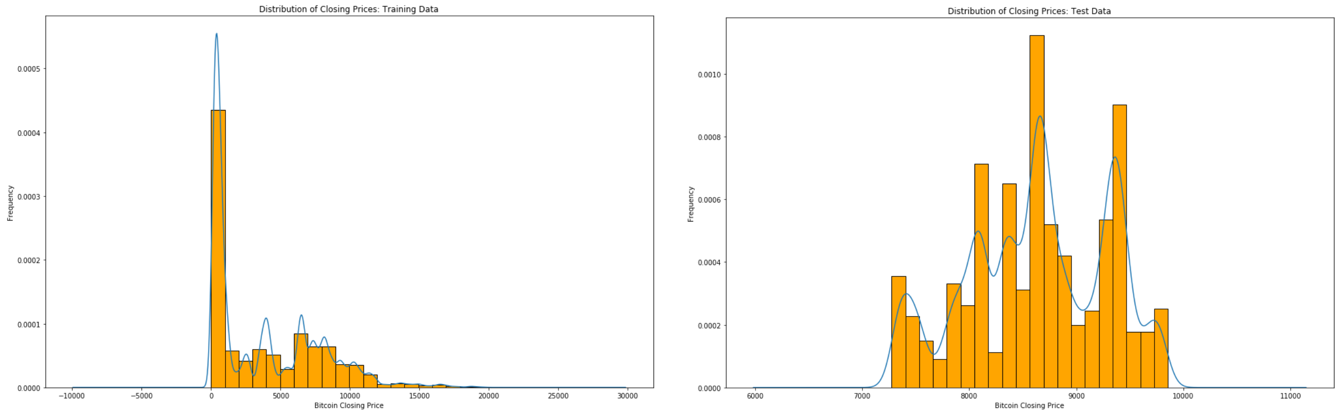
- Models built from decision trees (namely the Decision Tree, LGBM, and XGB) considered previous closing prices as the most important factors in predicting the price of Bitcoin

    - Moreover, these same models gave more weighting to the original features of the data (OHLC and volume), while the other models mainly considered the derived features

- The total number of unique features used across all models does not appear to be very close to the total number of features used for training. This suggests that the data could possibly benefit from methods for dimensionality reduction such as Singular Value Decomposition or Principal Component Analysis

In addition to analyzing the feature importances for each of the tested models, we also wanted to analyze the distributions of data used for training versus the distributions of data used for generating testing predictions.

Data Distributions:



Shown above is the kernel density estimate for the portion of data used in training offline models (left) compared to the kernel density estimate of the final 50,000 observations used for testing.

This comparison of distributions gives even more evidence for what we suspected in our surface level analysis of the pricing and volume data, which was that the volatility and range of prices was too large for a statically trained offline model.

Moreover, a Cramer Von-Mises parametric test failed to reject the null hypothesis that the data used for testing was distributed normally with parameters mu and sigma approximated by their maximum likelihood estimates. However, for the same test on the portion of data used for training, we could successfully reject the null at any reasonable level alpha.

When combined, the visual assessment and statistical analysis give significant evidence for the underlying cause of such drastic performance differences between offline and online models: the random variable which represents the true bitcoin price has a non-fixed, evolving distribution. Thus, models trained on past prices (and thus poorly aggregated previous distributions) cannot adjust to future scenarios where the underlying distribution has changed completely.

Finally, we reason that the underlying changes in distribution are driven by corresponding changes in the population of bitcoin market participants. Since the most common barriers to entry (professional certifications, costly advising and brokerage firms) are absent in bitcoin markets, market participants can more easily transition in and out of active market participation. Thus, only the most recently available data is indicative of the present true population & distribution. Though we generally observed the best results by using smaller window sizes *w*, it is recommended that this parameter be tuned on some fixed schedule if ever placed in production.

Summary

- We collected OHLC data for bitcoin at the 1-minute level spanning from April of 2013 to February 2020, for over 2.6 million rows of data

- We utilized the python package TA (Technical Analysis) to create ~72 additional features

- We trained offline models on the first 98% of observations and generated predictions for every other minute of the final 50,000 minutes (thus 25,000 predictions total)

- We trained online models such that for each prediction, a new model was trained only on some fixed size *w* of previous observations (*W* = {60, 90, 120, 150})

- A comparison of offline models and online models showed that online models significantly outperformed their offline counterparts

- Non-linear tree based models out performed linear models, suggesting the data is not linearly separable

- The ‘out-of-the-box’ regularization parameters for the Lasso and Ridge models are not rigorous enough to defend against over-fitting in the case of highly volatile data like bitcoin prices

- Additional data does not necessarily improve results - especially when the additional data is from a different population & distribution

- XGB was the best regressor in terms of both error and reliability, as it was the non-linear model with the strongest regularization efforts

- XGB was the slowest and most computationally intensive model. However, it is still *fast enough* such that in a live production setting, a new XGB model could successfully be trained and generate a prediction before the close of the next minute price

- The feature intensity values given by LIME for Support Vector Regressor are in the order of e-05, which means that SVR is giving almost no importance to any feature and hence is not able to learn efficiently

- LIME analysis gave evidence that dimensionality reduction techniques such as Singular Value Decomposition and Partial Component Analysis could preserve the performance of the models while reducing storage and computation costs

Future Plans

Though we have successfully compared various popular regression models to one another in the context of their ability to predict the future price of bitcoin, we have not benchmarked the models against some absolute standard of performance in order to determine if the predictions are accurate on an absolute basis. For instance, while we know that an ‘out-of-the-box’ XGB model trained on the previous 60 minutes of available data can produce predictions with a RMSE of 3.88, we do not know whether 3.88 is an *accurate enough* prediction to be used in a live trading setting.

With more time, we would also like to analyze a wider variety of models. Our results suggested that ensemble methods such as LBGM and XGB gave the most powerful mix of performance and cost, indicating to us that we could achieve better results still by applying more sophisticated ensemble techniques such as boosting, bagging and stacking. In fact, our LIME analysis seemed to lay the groundwork for the systematic development of non-correlated ensemble models, which could in theory aggregate the strengths and weaknesses of multiple models into one final model.

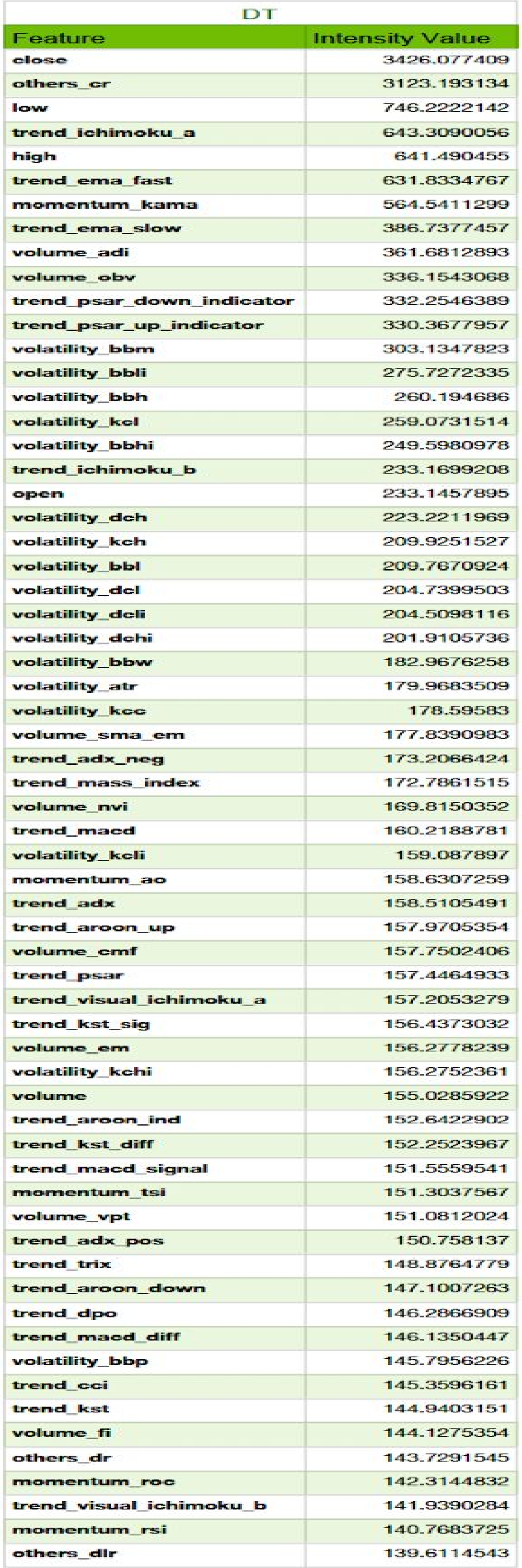
Finally, we would like to apply a similar analysis to other existing crypto-currencies (alt-coins). Like bitcoin, each alt-coin has its own highly unique use case. Thus, it is reasonable to hypothesize that the population of market participants for each of these other alt-coins is also highly unique. Moreover, alt-coint market caps are many orders of magnitude smaller than the Bitcoin market, thereby attracting less and less ‘sophisticated’ institutional investors. Thus, it’s reasonable to suspect that the specific characteristics, which underlie the bitcoin market, allow for powerful machine learning algorithms to be effective should follow the same principles in alt-coin markets. However, a proper analysis is required to fully flesh this out.

Appendix

*****STATEMENT OF CREDIT******

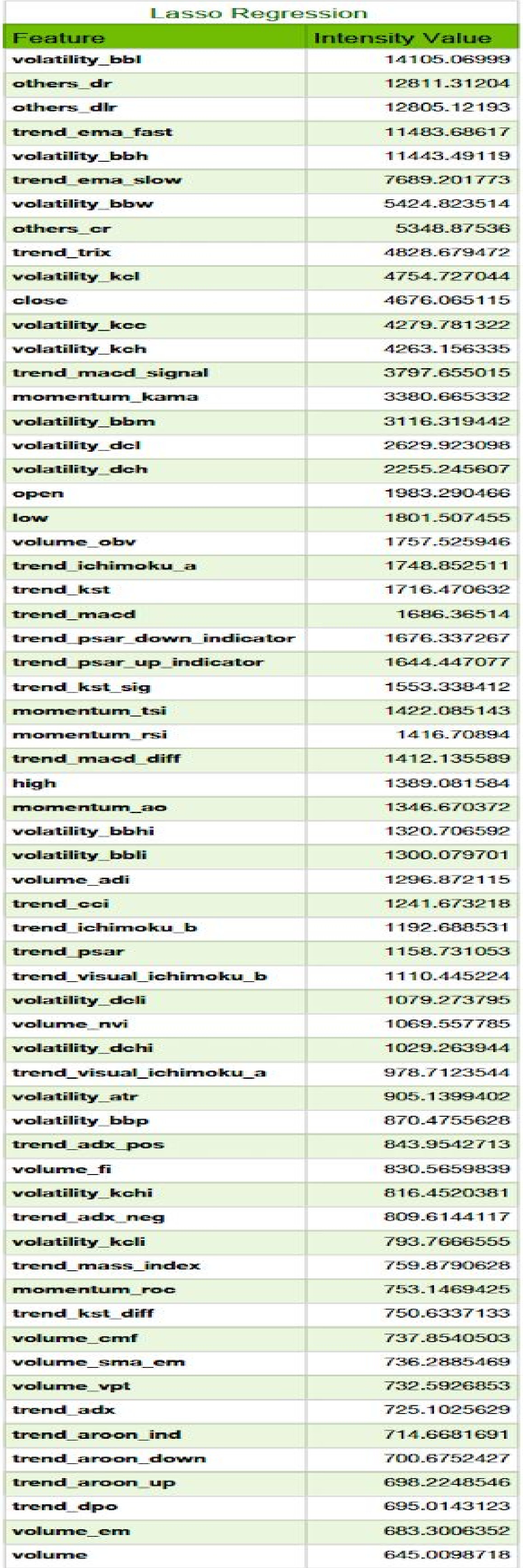
Each one of us worked on every part of the project. It was a fully collaborative process with valuable contributions from all members.

- **Feature intensities for each model in order of decreasing feature importance**



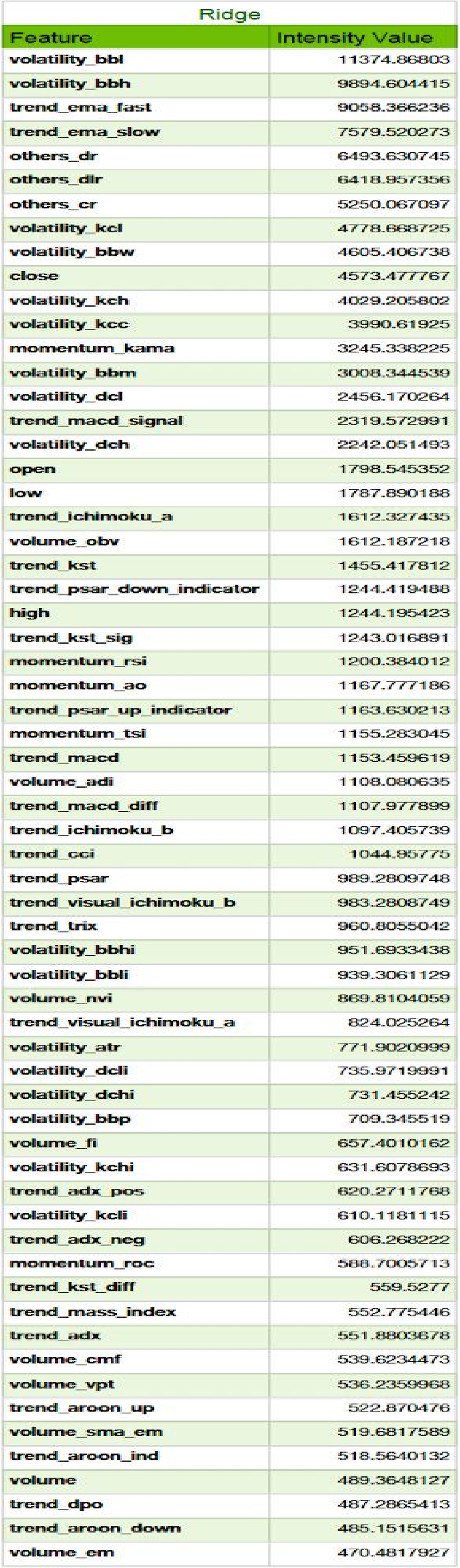
DT

Feature	Intensity Value
close	3426.077409
others_cr	3123.193134
low	746.2222142
trend_ichimoku_a	643.3090056
high	641.490455
trend_ema_fast	631.8334767
momentum_kama	564.5411299
trend_ema_slow	386.7377457
volume_adi	361.6812893
volume_obv	336.1543068
trend_psar_down_indicator	332.2546389
trend_psar_up_indicator	330.3677957
volatility_bbm	303.1347823
volatility_bbli	275.7272335
volatility_bbh	260.194686
volatility_kcl	259.0731514
volatility_bbhi	249.5980978
trend_ichimoku_b	233.1699208
open	233.1457895
volatility_dch	223.2211969
volatility_kch	209.9251527
volatility_bbl	209.7670924
volatility_dcl	204.7399503
volatility_dcli	204.5098116
volatility_dchi	201.9105736
volatility_bbw	182.9676258
volatility_atr	179.9683509
volatility_kcc	178.59583
volume_sma_em	177.8390983
trend_adx_neg	173.2066424
trend_mass_index	172.7861515
volume_nvi	169.8150352
trend_macd	160.2188781
volatility_kcli	159.087897
momentum_ao	158.6307259
trend_adx	158.5105491
trend_aroon_up	157.9705354
volume_cmf	157.7502406
trend_psar	157.4464933
trend_visual_ichimoku_a	157.2053279
trend_kst_sig	156.4373032
volume_em	156.2778239
volatility_kchi	156.2752361
volume	155.0285922
trend_aroon_ind	152.6422902
trend_kst_diff	152.2523967
trend_macd_signal	151.5559541
momentum_tsi	151.3037567
volume_vpt	151.0812024
trend_adx_pos	150.758137
trend_trix	148.8764779
trend_aroon_down	147.1007263
trend_dpo	146.2866909
trend_macd_diff	146.1350447
volatility_bbp	145.7956226
trend_cci	145.3596161
trend_kst	144.9403151
volume_fi	144.1275354
others_dr	143.7291545
momentum_roc	142.3144832
trend_visual_ichimoku_b	141.9390284
momentum_rsi	140.7683725
others_dlr	139.6114543



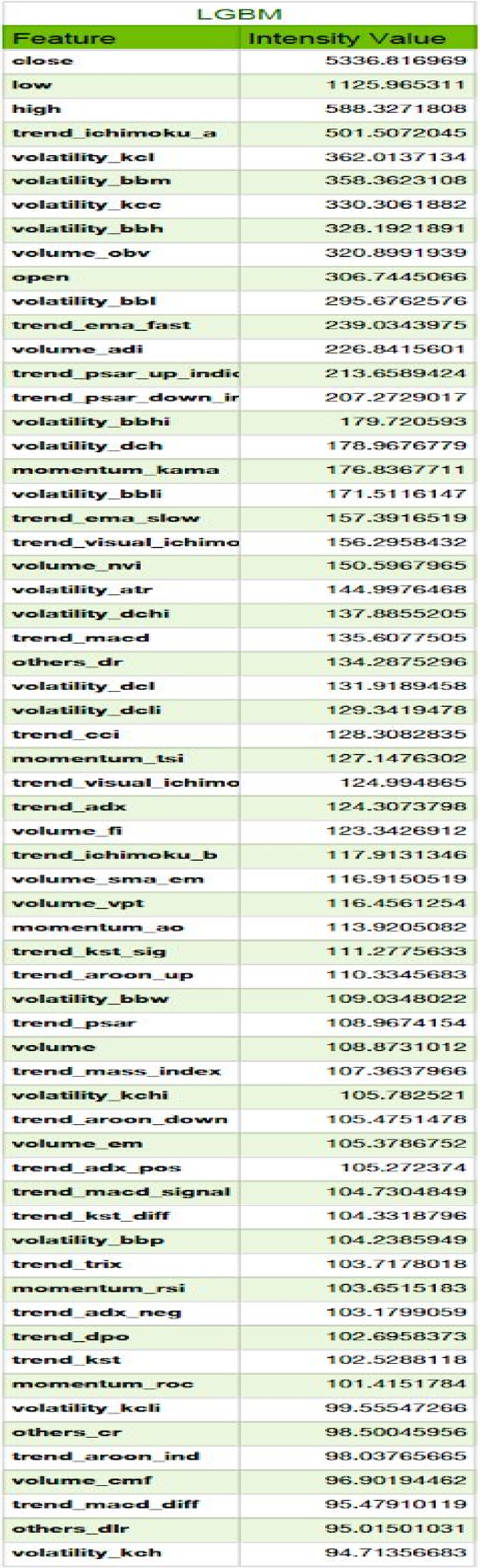
Lasso Regression

Feature	Intensity Value
volatility_bbl	14105.06999
others_dr	12811.31204
others_dlr	12805.12193
trend_ema_fast	11483.68617
volatility_bbh	11443.49119
trend_ema_slow	7689.201773
volatility_bbw	5424.823514
others_cr	5348.87536
trend_trix	4828.679472
volatility_kcl	4754.727044
close	4676.065115
volatility_kcc	4279.781322
volatility_kch	4263.156335
trend_macd_signal	3797.655015
momentum_kama	3380.665332
volatility_bbm	3116.319442
volatility_dcl	2629.923098
volatility_dch	2255.245607
open	1983.290466
low	1801.507455
volume_obv	1757.525946
trend_ichimoku_a	1748.852511
trend_kst	1716.470632
trend_macd	1686.36514
trend_psar_down_indicator	1676.337267
trend_psar_up_indicator	1644.447077
trend_kst_sig	1553.338412
momentum_tsi	1422.085143
momentum_rsi	1416.70894
trend_macd_diff	1412.135589
high	1389.081584
momentum_ao	1346.670372
volatility_bbhi	1320.706592
volatility_bbli	1300.079701
volume_adi	1296.872115
trend_cci	1241.673218
trend_ichimoku_b	1192.688531
trend_psar	1158.731053
trend_visual_ichimoku_b	1110.445224
volatility_dcli	1079.273795
volume_nvi	1069.557785
volatility_dchi	1029.263944
trend_visual_ichimoku_a	978.7123544
volatility_atr	905.1399402
volatility_bbp	870.4755628
trend_adx_pos	843.9542713
volume_fi	830.5659839
volatility_kchi	816.4520381
trend_adx_neg	809.6144117
volatility_kcli	793.7666555
trend_mass_index	759.8790628
momentum_roc	753.1469425
trend_kst_diff	750.6337133
volume_cmf	737.8540503
volume_sma_em	736.2885469
volume_vpt	732.5926853
trend_adx	725.1025629
trend_aroon_ind	714.6681691
trend_aroon_down	700.6752427
trend_aroon_up	698.2248546
trend_dpo	695.0143123
volume_em	683.3006352
volume	645.0098718



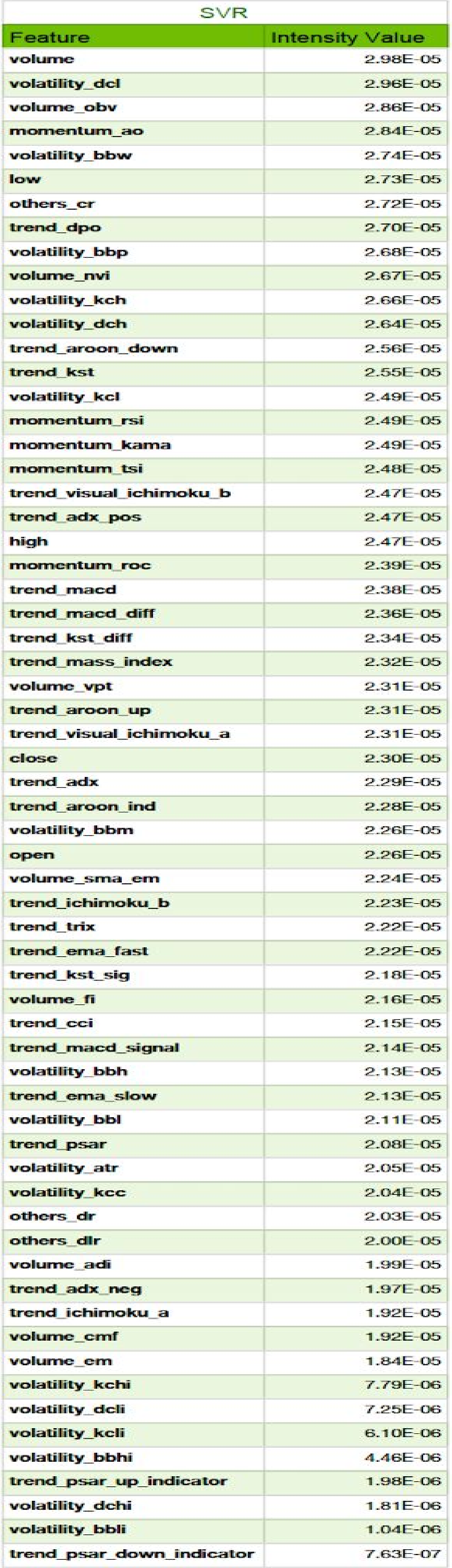
Ridge

Feature	Intensity Value
volatility_bbl	11374.86803
volatility_bbh	9894.604415
trend_ema_fast	9058.366236
trend_ema_slow	7579.520273
others_dr	6493.630745
others_dlr	6418.957356
others_cr	5250.067097
volatility_kcl	4778.668725
volatility_bbw	4605.406738
close	4573.477767
volatility_kch	4029.205802
volatility_kcc	3990.61925
momentum_kama	3245.338225
volatility_bbm	3008.344539
volatility_dcl	2456.170264
trend_macd_signal	2319.572991
volatility_dch	2242.051493
open	1798.545352
low	1787.890188
trend_ichimoku_a	1612.327435
volume_obv	1612.187218
trend_kst	1455.417812
trend_psar_down_indicator	1244.419488
high	1244.195423
trend_kst_sig	1243.016891
momentum_rsi	1200.384012
momentum_ao	1167.777186
trend_psar_up_indicator	1163.630213
momentum_tsi	1155.283045
trend_macd	1153.459619
volume_adi	1108.080635
trend_macd_diff	1107.977899
trend_ichimoku_b	1097.405739
trend_cci	1044.95775
trend_psar	989.2809748
trend_visual_ichimoku_b	983.2808749
trend_trix	960.8055042
volatility_bbhi	951.6933438
volatility_bbli	939.3061129
volume_nvi	869.8104059
trend_visual_ichimoku_a	824.025264
volatility_atr	771.9020999
volatility_dcli	735.9719991
volatility_dchi	731.455242
volatility_bbp	709.345519
volume_fi	657.4010162
volatility_kchi	631.6078693
trend_adx_pos	620.2711768
volatility_kcli	610.1181115
trend_adx_neg	606.268222
momentum_roc	588.7005713
trend_kst_diff	559.5277
trend_mass_index	552.775446
trend_adx	551.8803678
volume_cmf	539.6234473
volume_vpt	536.2359968
trend_aroon_up	522.870476
volume_sma_em	519.6817589
trend_aroon_ind	518.5640132
volume	489.3648127
trend_dpo	487.2865413
trend_aroon_down	485.1515631
volume_em	470.4817927



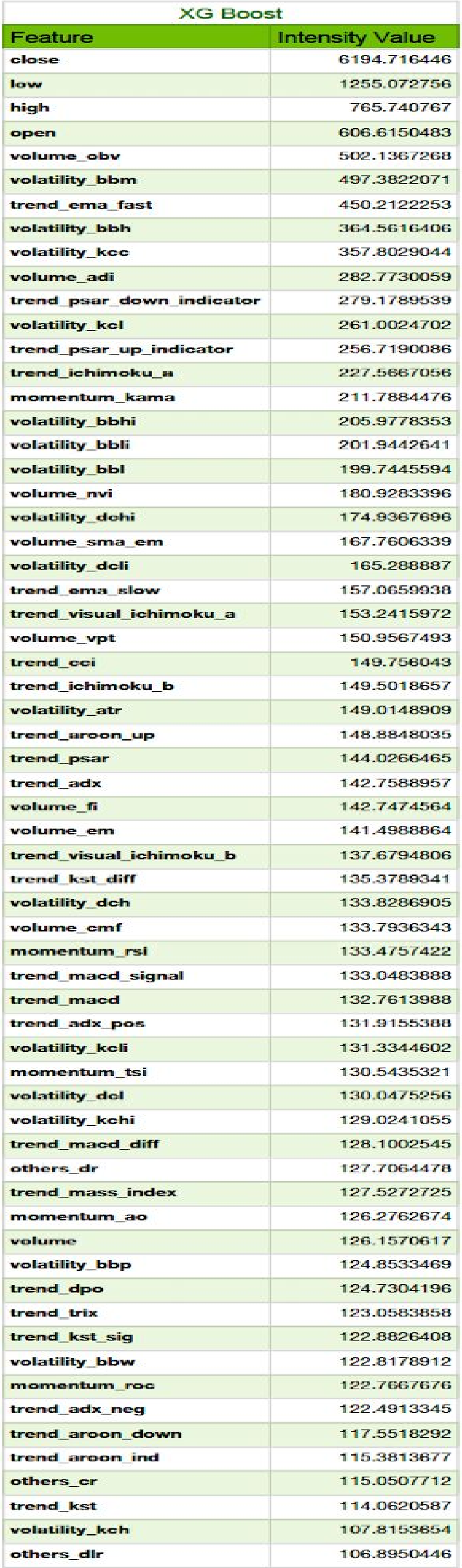
LGBM

Feature	Intensity Value
close	5336.816969
low	1125.965311
high	588.3271808
trend_ichimoku_a	501.5072045
volatility_kcl	362.0137134
volatility_bbm	358.3623108
volatility_kcc	330.3061882
volatility_bbh	328.1921891
volume_obv	320.8991939
open	306.7445066
volatility_bbl	295.6762576
trend_ema_fast	239.0343975
volume_adi	226.8415601
trend_psar_up_indic	213.6589424
trend_psar_down_ir	207.2729017
volatility_bbhi	179.720593
volatility_dch	178.9676779
momentum_kama	176.8367711
volatility_bbli	171.5116147
trend_ema_slow	157.3916519
trend_visual_ichimo	156.2958432
volume_nvi	150.5967965
volatility_atr	144.9976468
volatility_dchi	137.8855205
trend_macd	135.6077505
others_dr	134.2875296
volatility_dcl	131.9189458
volatility_dcli	129.3419478
trend_cci	128.3082835
momentum_tsi	127.1476302
trend_visual_ichimo	124.994865
trend_adx	124.3073798
volume_fi	123.3426912
trend_ichimoku_b	117.9131346
volume_sma_em	116.9150519
volume_vpt	116.4561254
momentum_ao	113.9205082
trend_kst_sig	111.2775633
trend_aroon_up	110.3345683
volatility_bbw	109.0348022
trend_psar	108.9674154
volume	108.8731012
trend_mass_index	107.3637966
volatility_kchi	105.782521
trend_aroon_down	105.4751478
volume_em	105.3786752
trend_adx_pos	105.272374
trend_macd_signal	104.7304849
trend_kst_diff	104.3318796
volatility_bbp	104.2385949
trend_trix	103.7178018
momentum_rsi	103.6515183
trend_adx_neg	103.1799059
trend_dpo	102.6958373
trend_kst	102.5288118
momentum_roc	101.4151784
volatility_kcli	99.55547266
others_cr	98.50045956
trend_aroon_ind	98.03765665
volume_cmf	96.90194462
trend_macd_diff	95.47910119
others_dlr	95.01501031
volatility_kch	94.71356683



SVR

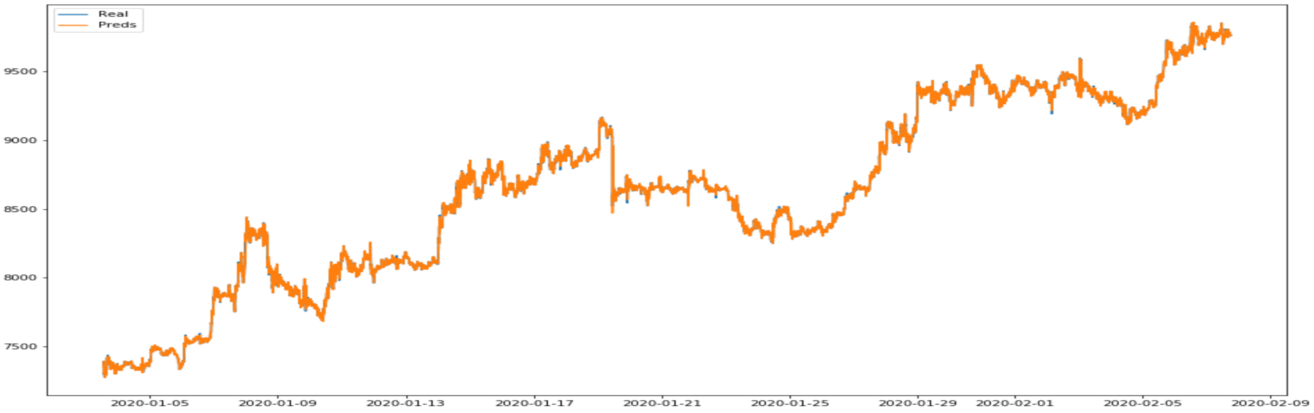
Feature	Intensity Value
volume	2.98E-05
volatility_dcl	2.96E-05
volume_obv	2.86E-05
momentum_ao	2.84E-05
volatility_bbw	2.74E-05
low	2.73E-05
others_cr	2.72E-05
trend_dpo	2.70E-05
volatility_bbp	2.68E-05
volume_nvi	2.67E-05
volatility_kch	2.66E-05
volatility_dch	2.64E-05
trend_aroon_down	2.56E-05
trend_kst	2.55E-05
volatility_kcl	2.49E-05
momentum_rsi	2.49E-05
momentum_kama	2.49E-05
momentum_tsi	2.48E-05
trend_visual_ichimoku_b	2.47E-05
trend_adx_pos	2.47E-05
high	2.47E-05
momentum_roc	2.39E-05
trend_macd	2.38E-05
trend_macd_diff	2.36E-05
trend_kst_diff	2.34E-05
trend_mass_index	2.32E-05
volume_vpt	2.31E-05
trend_aroon_up	2.31E-05
trend_visual_ichimoku_a	2.31E-05
close	2.30E-05
trend_adx	2.29E-05
trend_aroon_ind	2.28E-05
volatility_bbm	2.26E-05
open	2.26E-05
volume_sma_em	2.24E-05
trend_ichimoku_b	2.23E-05
trend_trix	2.22E-05
trend_ema_fast	2.22E-05
trend_kst_sig	2.18E-05
volume_fi	2.16E-05
trend_cci	2.15E-05
trend_macd_signal	2.14E-05
volatility_bbh	2.13E-05
trend_ema_slow	2.13E-05
volatility_bbl	2.11E-05
trend_psar	2.08E-05
volatility_atr	2.05E-05
volatility_kcc	2.04E-05
others_dr	2.03E-05
others_dlr	2.00E-05
volume_adi	1.99E-05
trend_adx_neg	1.97E-05
trend_ichimoku_a	1.92E-05
volume_cmf	1.92E-05
volume_em	1.84E-05
volatility_kchi	7.79E-06
volatility_dcli	7.25E-06
volatility_kcli	6.10E-06
volatility_bbhi	4.46E-06
trend_psar_up_indicator	1.98E-06
volatility_dchi	1.81E-06
volatility_bbli	1.04E-06
trend_psar_down_indicator	7.63E-07

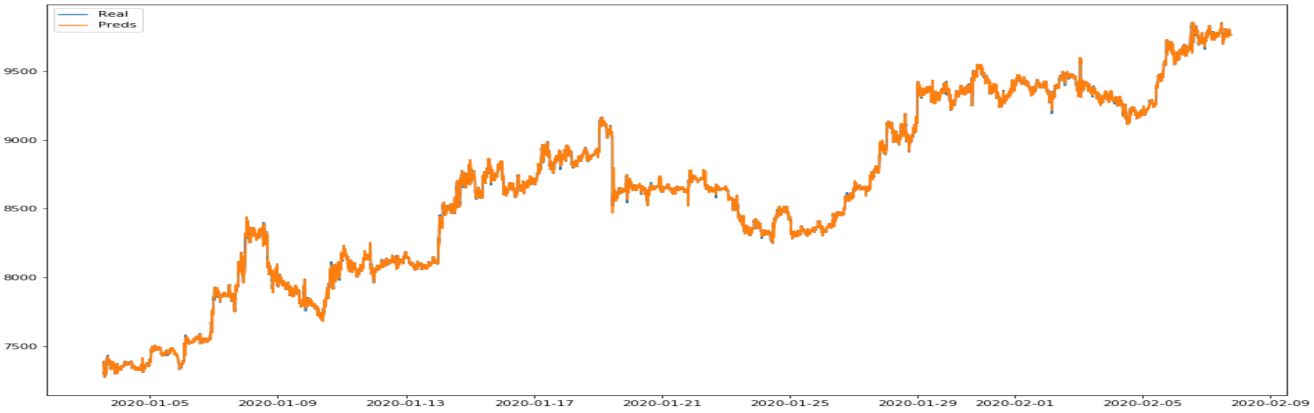


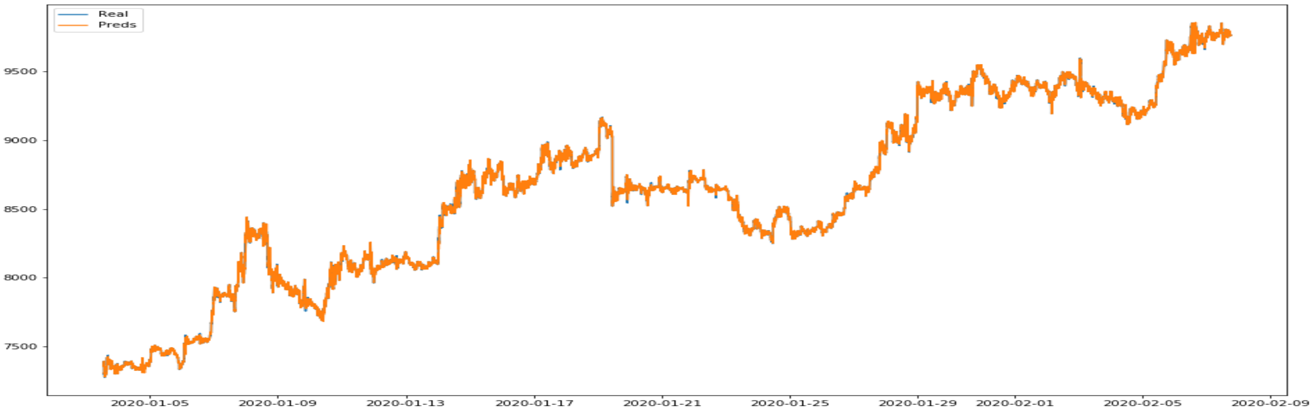
XG Boost

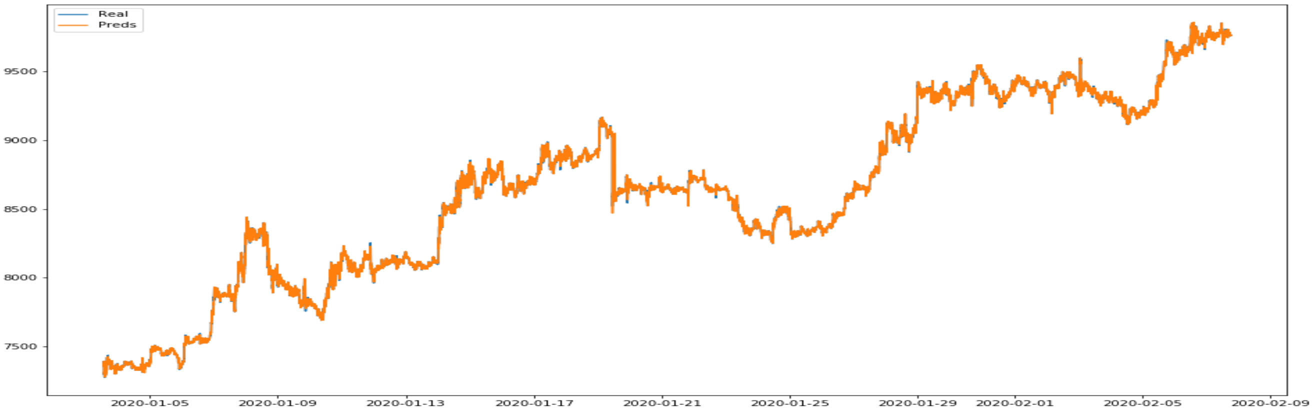
Feature	Intensity Value
close	6194.716446
low	1255.072756
high	765.740767
open	606.6150483
volume_obv	502.1367268
volatility_bbm	497.3822071
trend_ema_fast	450.2122253
volatility_bbh	364.5616406
volatility_kcc	357.8029044
volume_adi	282.7730059
trend_psar_down_indicator	279.1789539
volatility_kcl	261.0024702
trend_psar_up_indicator	256.7190086
trend_ichimoku_a	227.5667056
momentum_kama	211.7884476
volatility_bbhi	205.9778353
volatility_bbli	201.9442641
volatility_bbl	199.7445594
volume_nvi	180.9283396
volatility_dchi	174.9367696
volume_sma_em	167.7606339
volatility_dcli	165.288887
trend_ema_slow	157.0659938
trend_visual_ichimoku_a	153.2415972
volume_vpt	150.9567493
trend_cci	149.756043
trend_ichimoku_b	149.5018657
volatility_atr	149.0148909
trend_aroon_up	148.8848035
trend_psar	144.0266465
trend_adx	142.7588957
volume_fi	142.7474564
volume_em	141.4988864
trend_visual_ichimoku_b	137.6794806
trend_kst_diff	135.3789341
volatility_dch	133.8286905
volume_cmf	133.7936343
momentum_rsi	133.4757422
trend_macd_signal	133.0483888
trend_macd	132.7613988
trend_adx_pos	131.9155388
volatility_kcli	131.3344602
momentum_tsi	130.5435321
volatility_dcl	130.0475256
volatility_kchi	129.0241055
trend_macd_diff	128.1002545
others_dr	127.7064478
trend_mass_index	127.5272725
momentum_ao	126.2762674
volume	126.1570617
volatility_bbp	124.8533469
trend_dpo	124.7304196
trend_trix	123.0583858
trend_kst_sig	122.8826408
volatility_bbw	122.8178912
momentum_roc	122.7667676
trend_adx_neg	122.4913345
trend_aroon_down	117.5518292
trend_aroon_ind	115.3813677
others_cr	115.0507712
trend_kst	114.0620587
volatility_kch	107.8153654
others_dlr	106.8950446

- ## **Decision Tree Models**

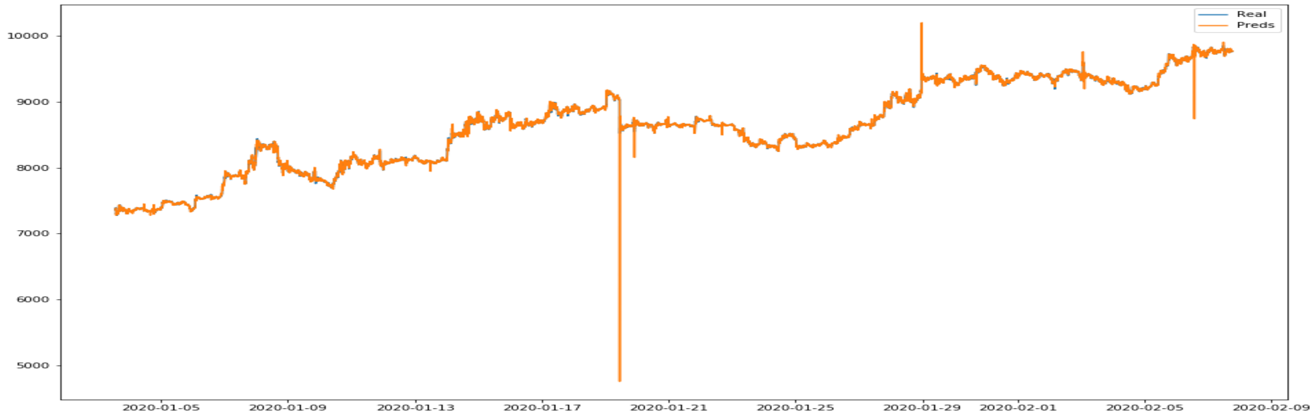


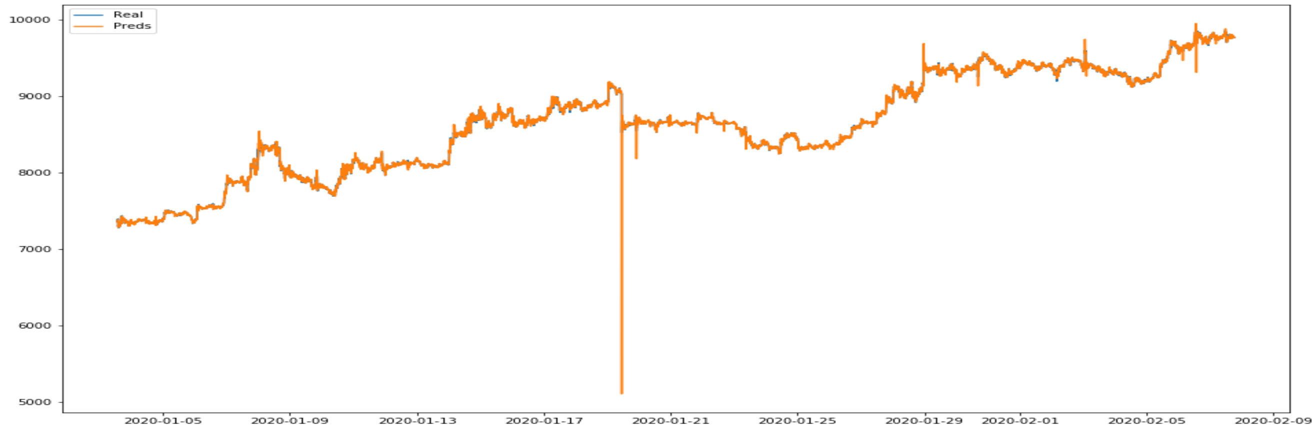


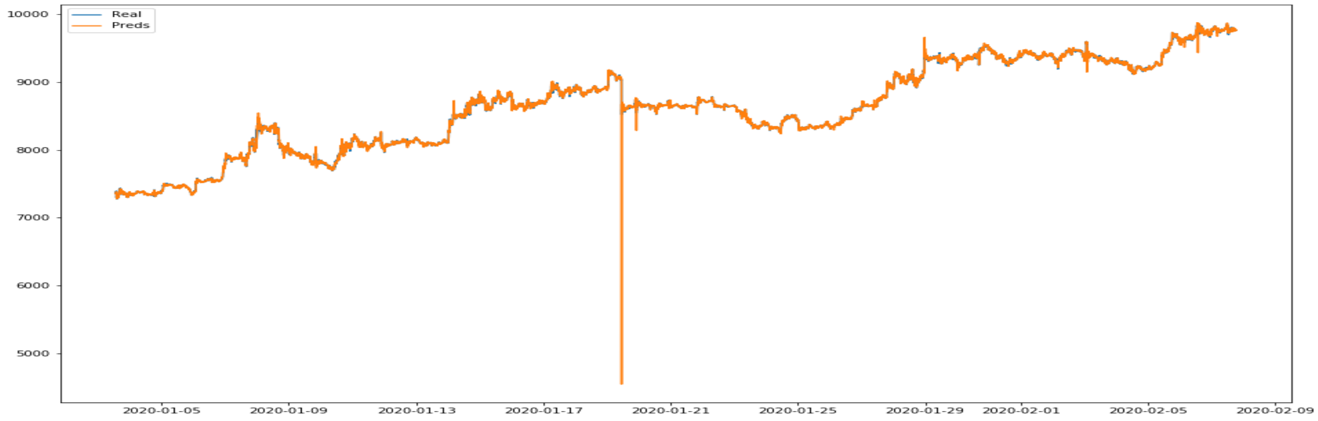


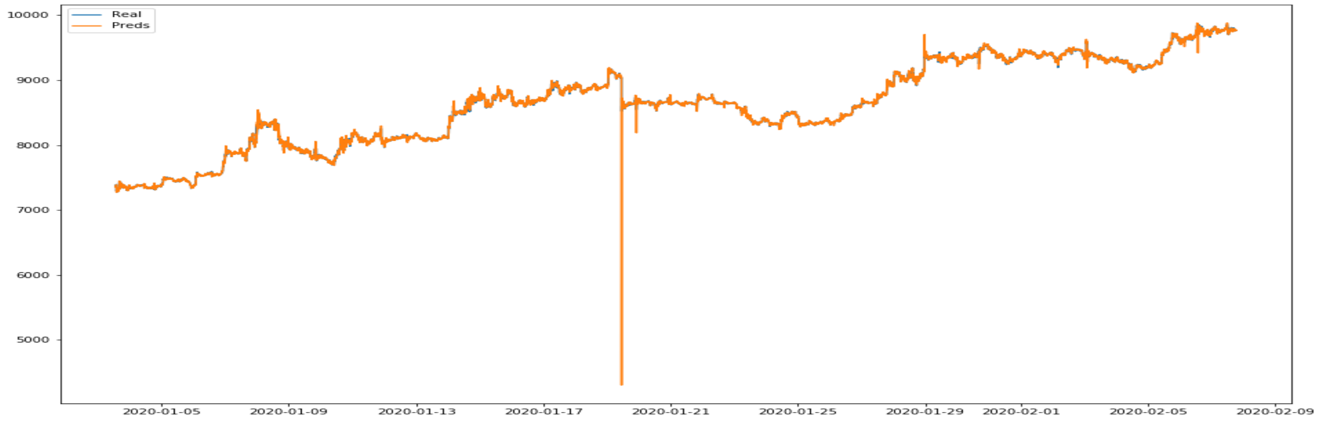


- **Lasso Regression Models**

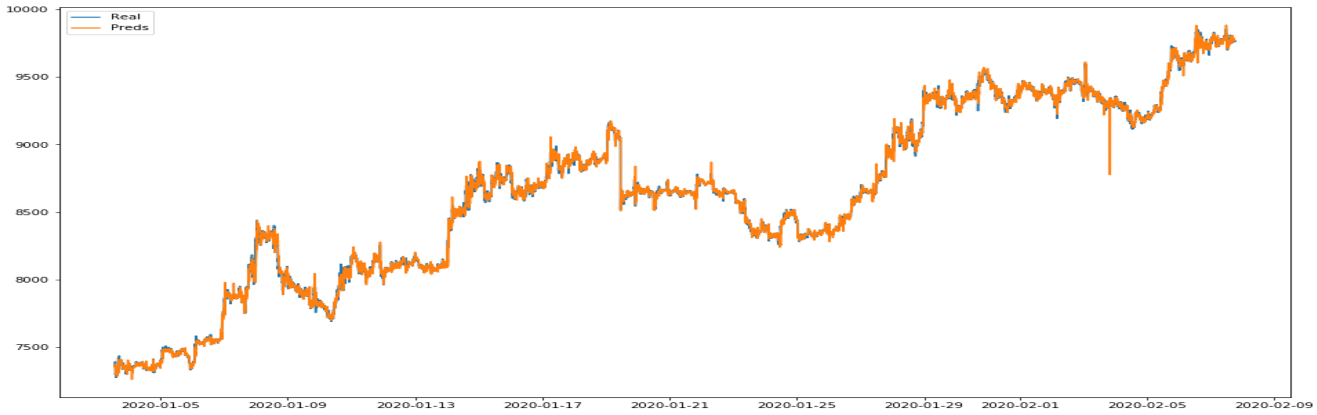


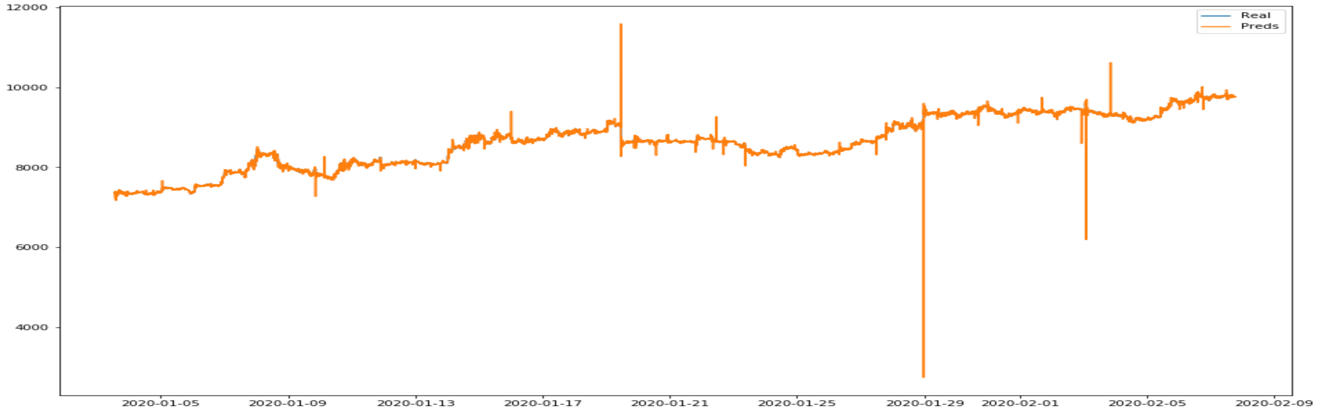


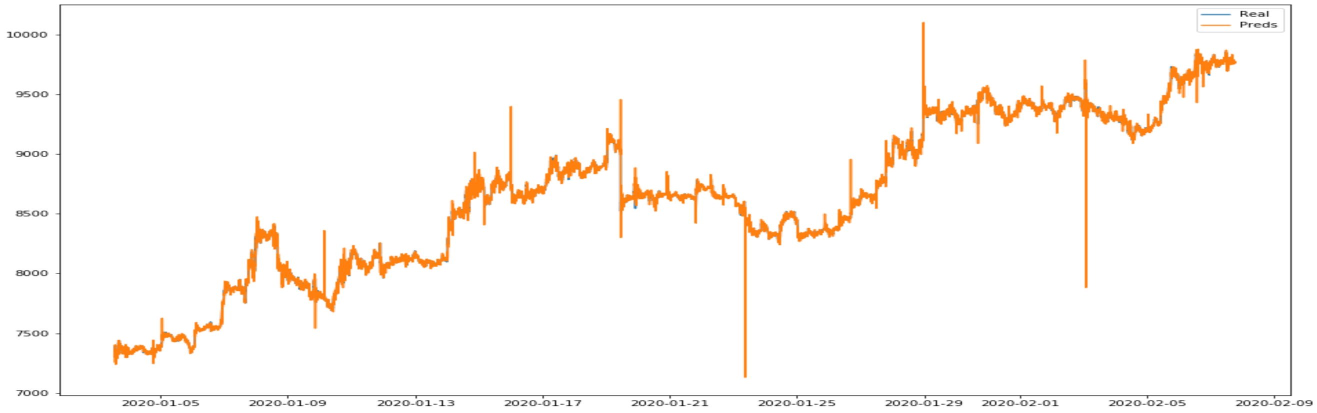


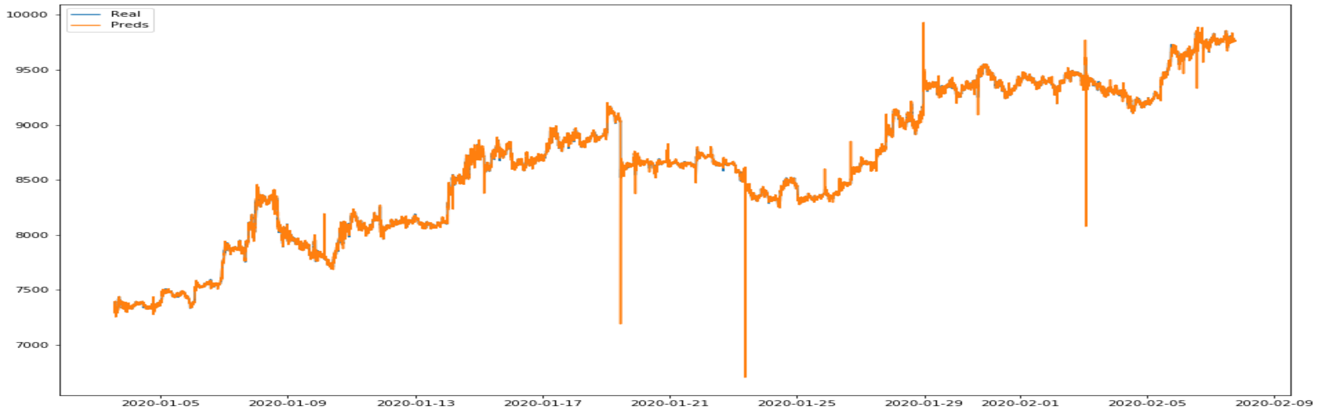


- **Ridge Regression Models**

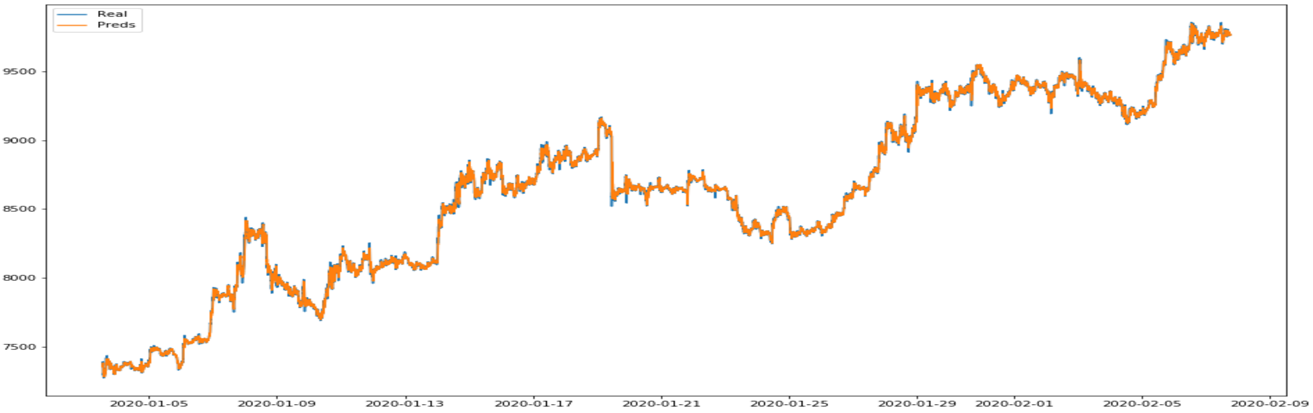


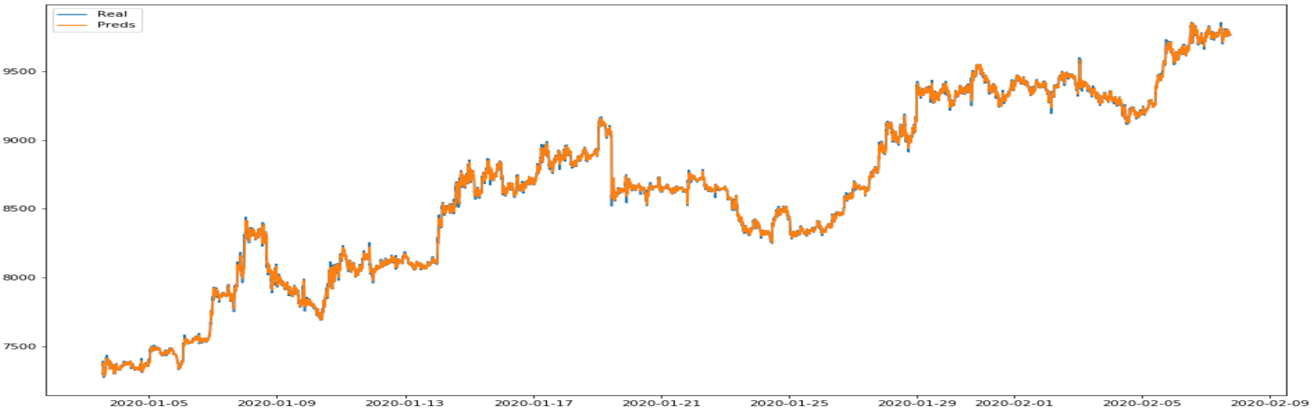


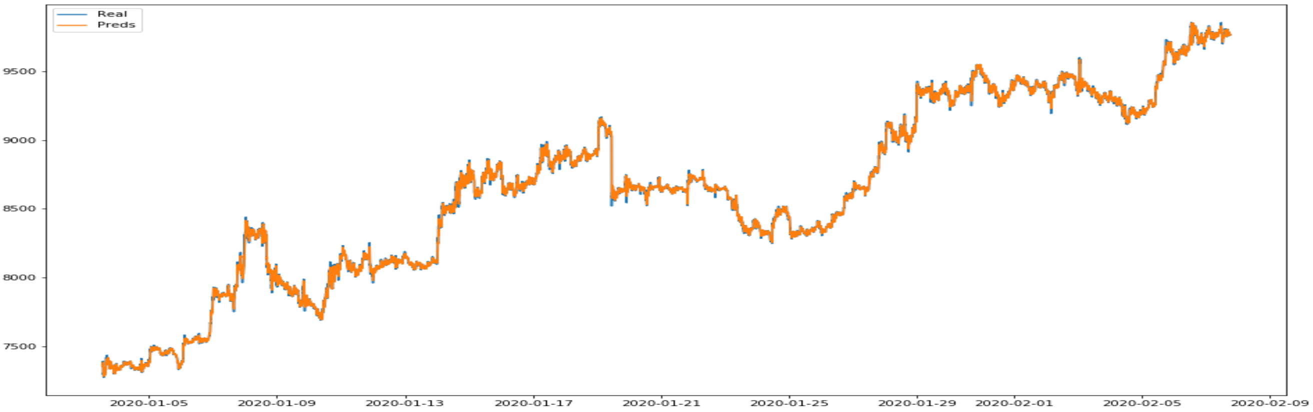


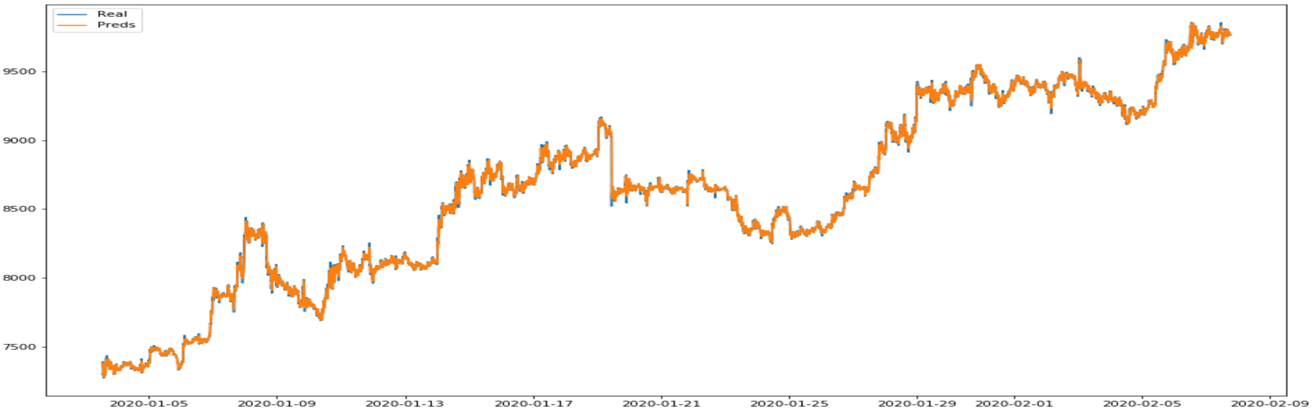


- **LGBM Regression Models**

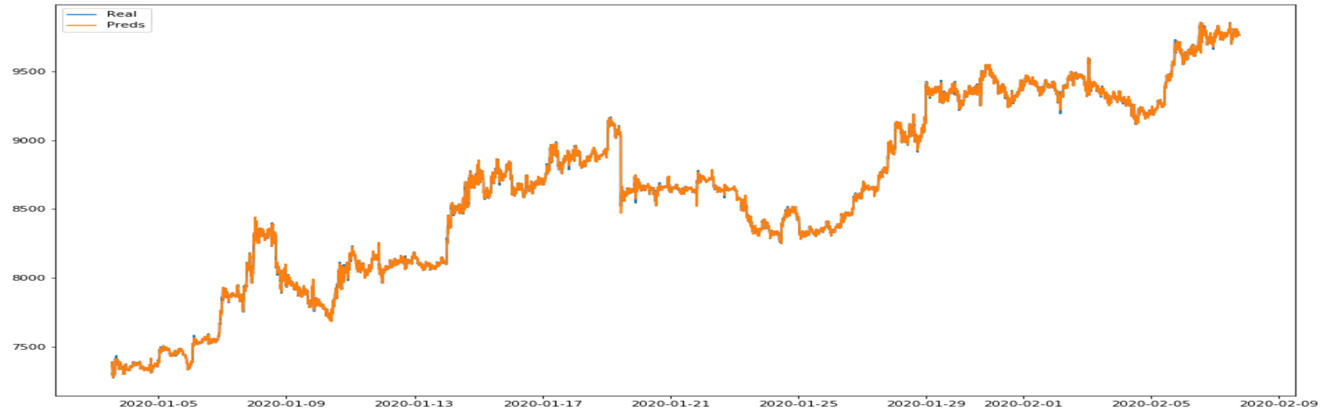


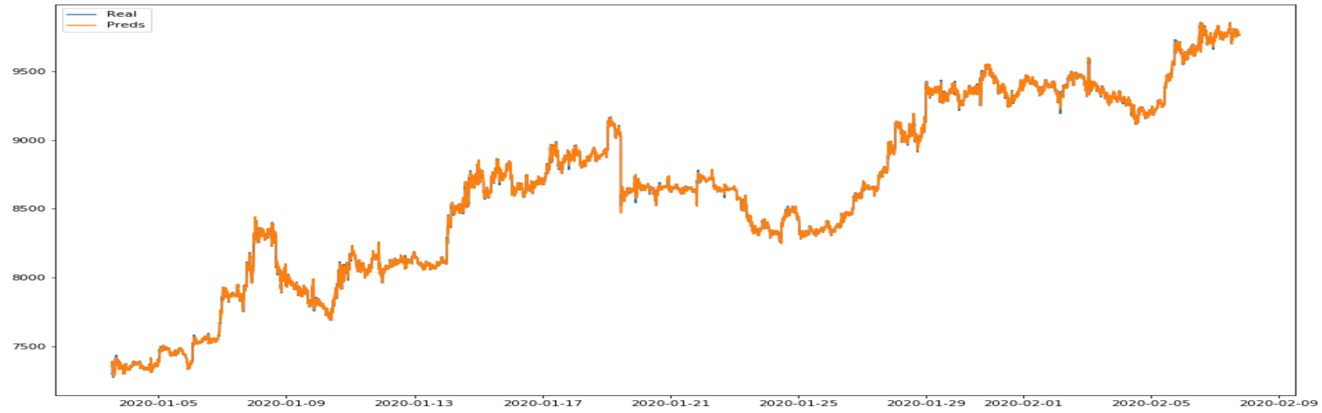


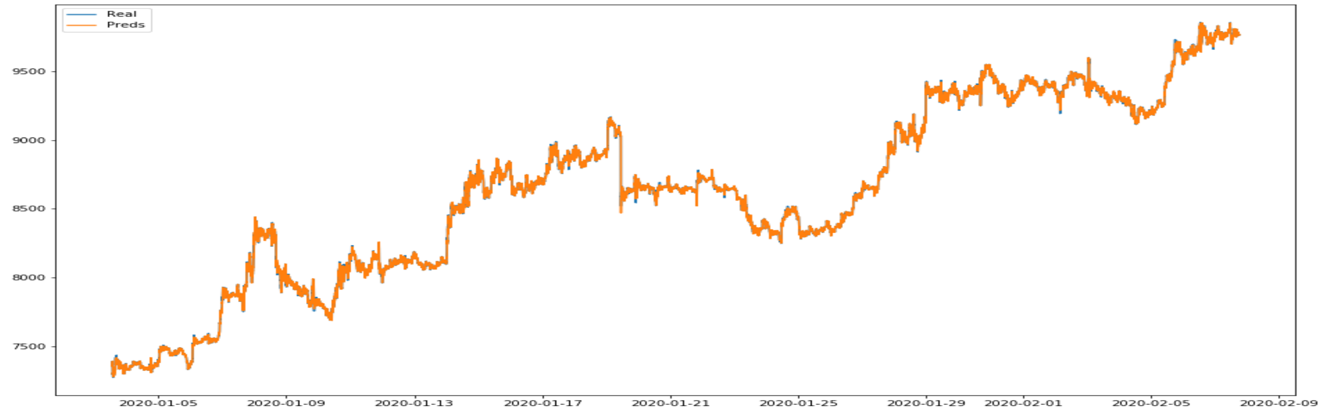


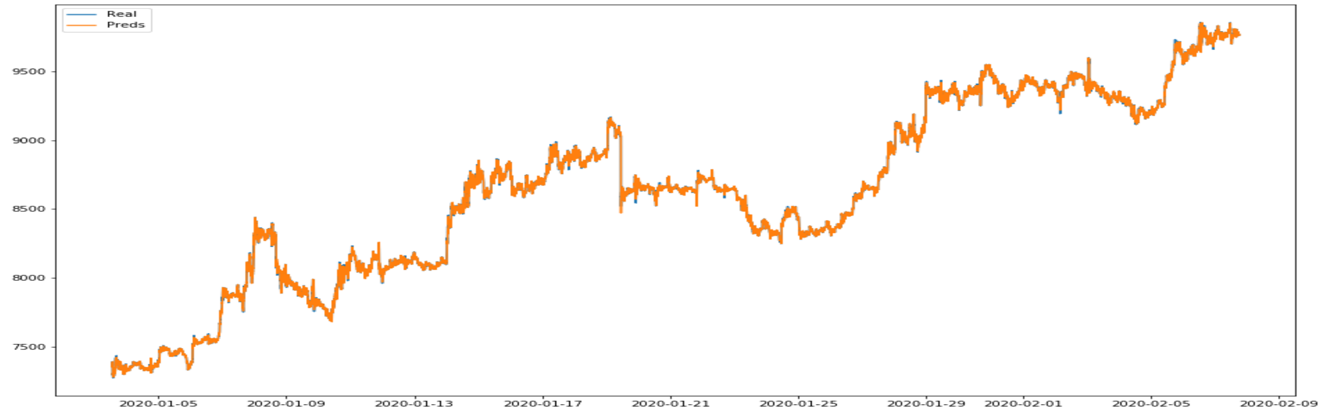


- **XGB Regression Models**

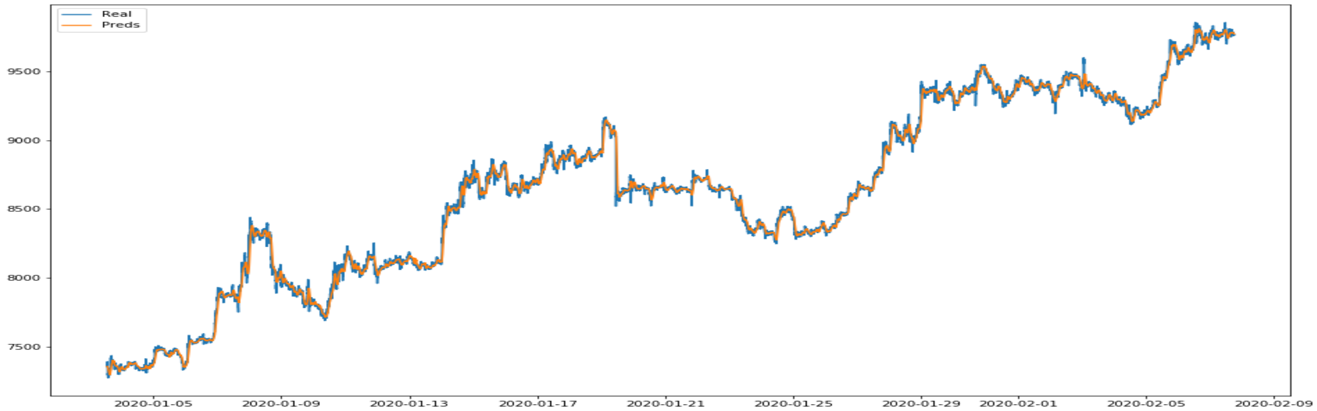


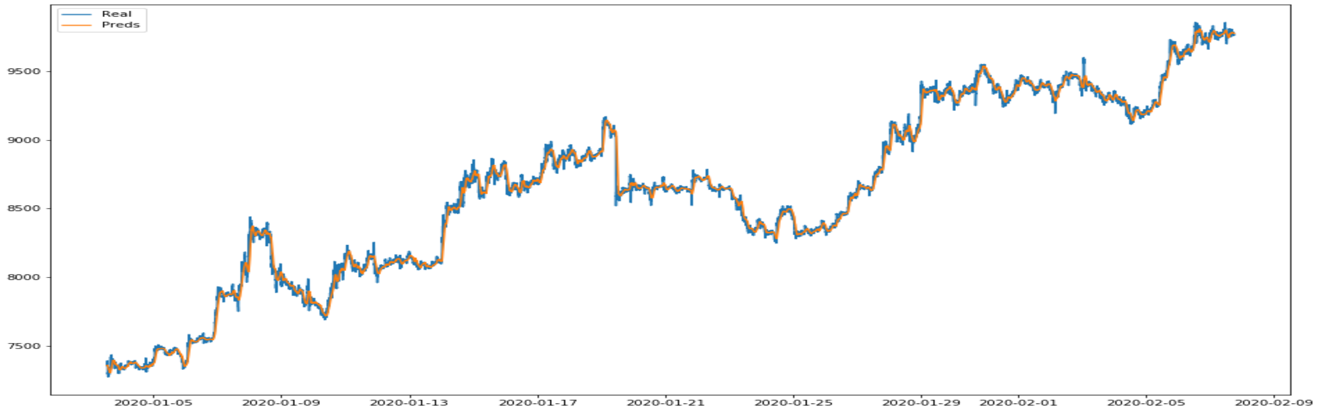


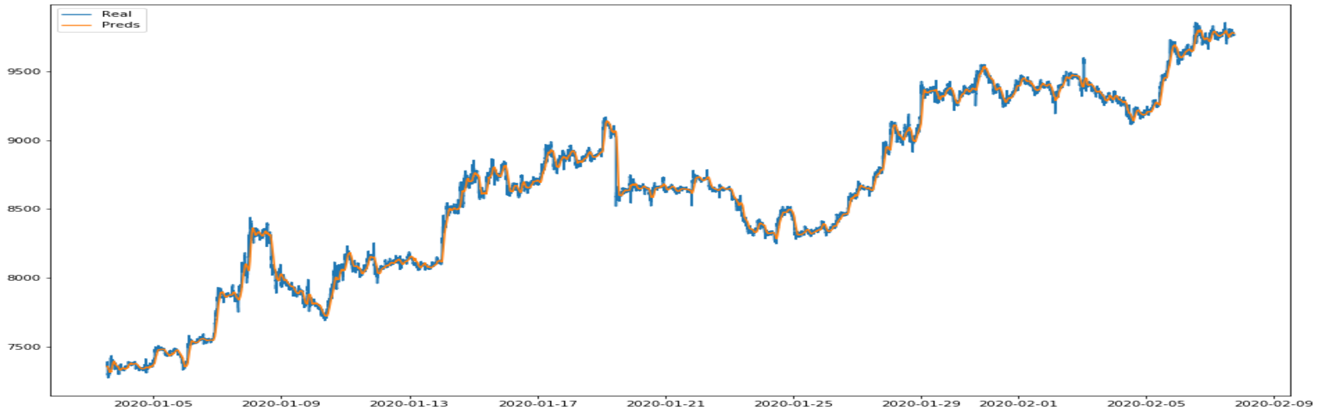


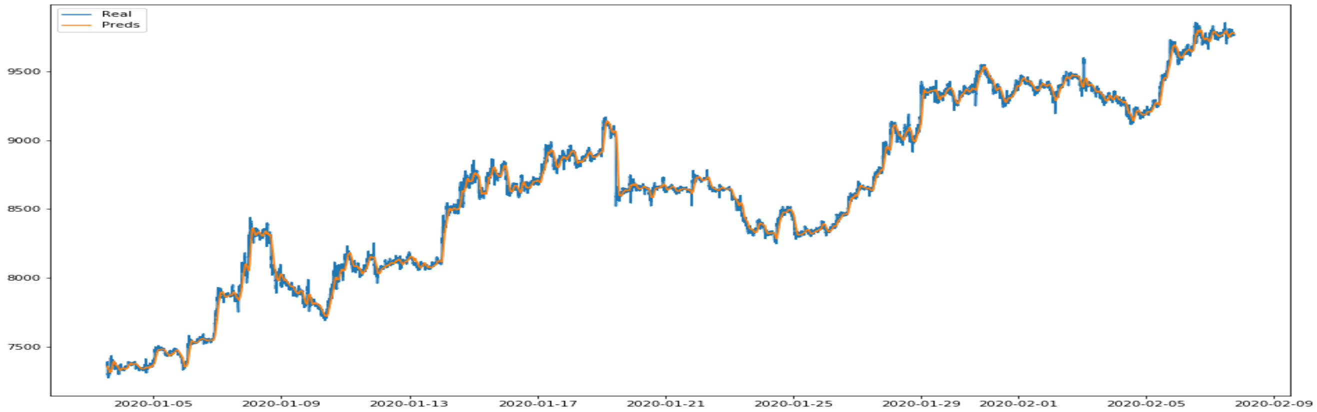


- **SVR Regression Models**



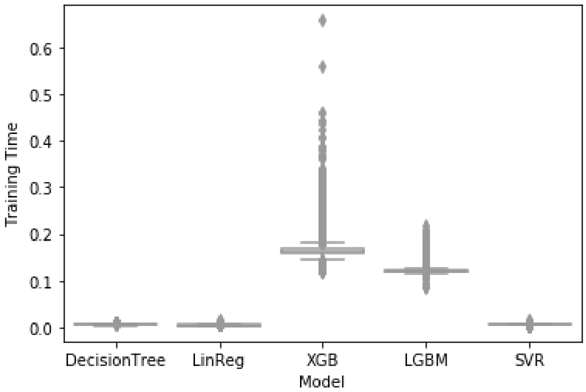


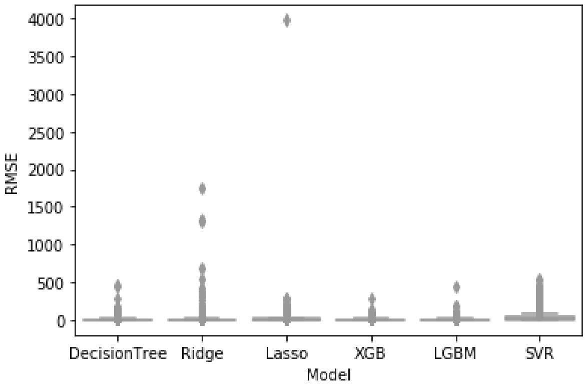


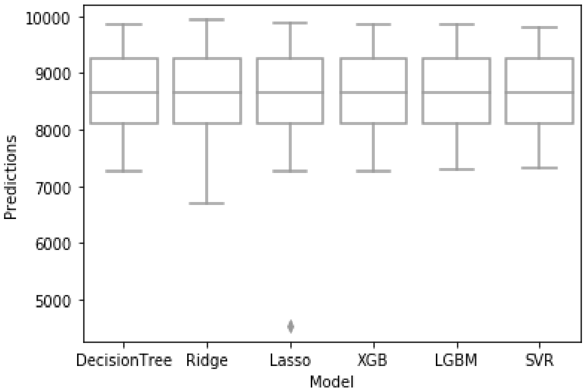


Comparative Plots

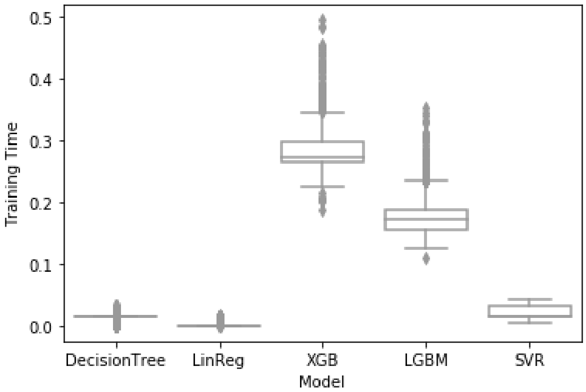
Online Learners:

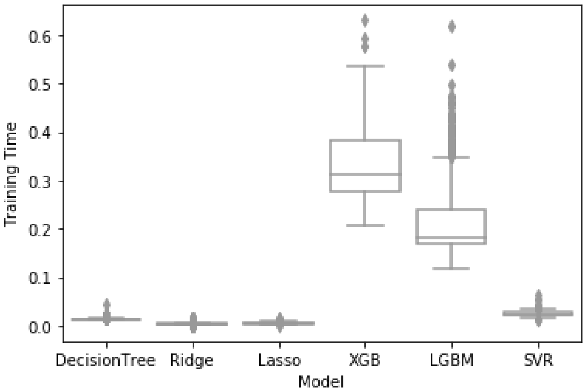






Offline Learners:





References

- Data Collection
  - https://www.kaggle.com/tencars/392-crypto-currency-pairs-at-minute-resolution

- Modeling
  - https://scikit-learn.org/stable/
  - https://xgboost.readthedocs.io

- LIME - Feature Importance
  - Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. “Why Should I Trust You?”: Explaining the Predictions of Any Classifier. https://arxiv.org/abs/1602.04938v3
  - https://homes.cs.washington.edu/~marcotcr/blog/lime/
  - https://towardsdatascience.com/understanding-model-predictions-with-lime-a582fdff3a3b

- Trading Indicators
  - https://www.tradingview.com/scripts/cryptocurrency/
  - https://pypi.org/project/ta/