A Comparative Analysis of Machine Learning and Naive Models as Effective Benchmarks for Cryptocurrency Price Prediction

Advisor: Prof. Manuel Roveri
Co-Advisor: Ing. Alessandro Falcetta

Thesis by:
Samuel Kala
995367

Table of contents

Introduction and problem statement

Introduction and problem statement

- What is the **objective** of this thesis? **Assess** the **predictability** of **cryptocurrency price**
- Logical steps that conducted to this decision:
  - Utilize **time series forecasting** in a **financial setting**
  - Notoriously difficult to predict the market as suggested by the **Efficient Market Hypothesis** and **Random Walk** theories
  - Usually newer markets, such as that of cryptocurrencies, exhibit **inefficiencies** that in theory could be exploited to predict them. Furthermore, it is also believed that cryptocurrency markets are not driven by **fundamental factors**
- Given these considerations the crypto market seemed the most suitable financial niche where to apply **time series forecasting techniques**
- Main challenges: **non linear**, **non stationary** and **noisy data**

Introduction and problem statement

- How is the predictability of cryptocurrency price assessed?
  - Employment of Advanced Machine Learning models
  - Results of **ML models** compared to those obtained by simpler benchmark models, known as **Naive models**, usually used to model random walk series. (They predict that the value of the next time period is equal to the current value)
  - Metrics for the comparison: **MAE, RMSE, MAPE**
  - Analysis performed for 3 different forecast horizons: **1 day, 7 days, 30 days**
  - Cryptocurrencies employed: **Bitcoin**, **Ethereum**, **Binance Coin**, **Solana**, **Ripple**

…

Introduction and problem statement

…

- Models employed, divided according to the classification of the Python library Darts, used for this analysis:

- Baseline models (benchmark)
  - **Naive Drift** model
  - **Naive** model
  - **Naive Seasonal** model
- Regression Models
  - Random Forest (**RF**)
  - **LightGBM** model
  - **XGBoost** model
  - **CatBoost** model
- Deep Learning Models
  - Long short-term memory (**LSTM**)
  - Recurrent Neural network (**RNN**)
  - Gated Recurrent Unit (**GRU**) model
  - **Transformer** model
  - Temporal Fusion Transformer (**TFT**)
  - Neural Basis Expansion Analysis (**N-BEATS**)
  - Temporal Convolutional Network (**TCN**)

Introduction and problem statement

- Main differences with the analyzed literature:
  - Comparison between **ML models** and **Baseline models**, to verify if ML models are effective. If the performance of a certain ML model is worse that the performance of a Naive Model, that model is not worth to be considered
  - **"Computer science"** perspective. MAE, RMSE, MAPE performance metrics, instead of financial ones (ROI, Sharpe Ratio…)
  - **Greater number** of **Predictive Models** and **Cryptocurrencies** considered

Methodology

Methodology

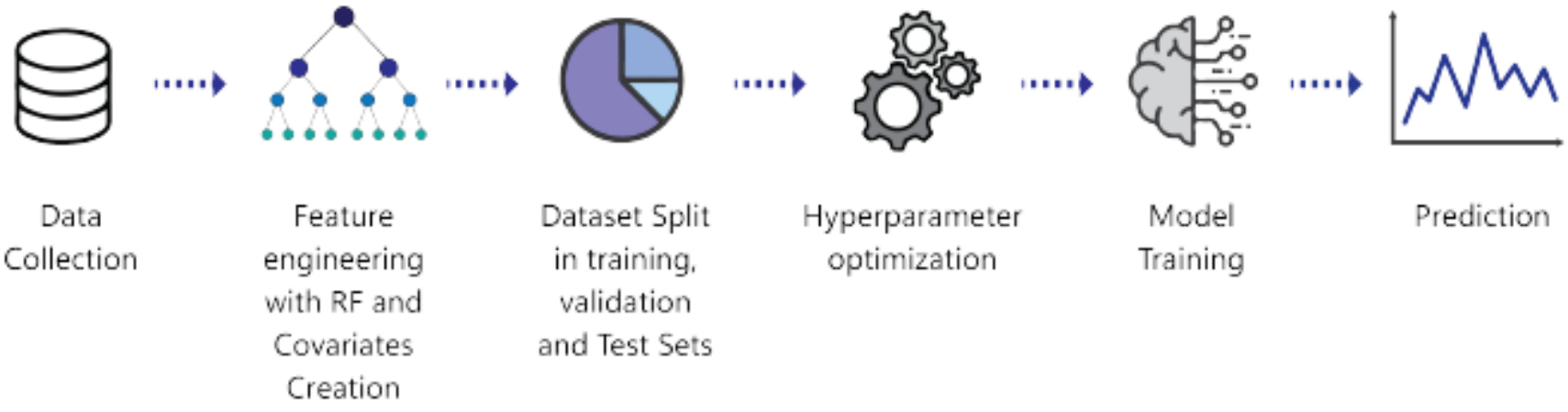


Figure 4: Pipeline from data collection to prediction

Methodology

- Data collection and preparation
  - metrics **Open, High**, **Low**, **Close** and **Volume** are retrieved using the Python library yfinance
  - To enhance the predictive accuracy of the ML models and ensure their robustness against **overfitting** and **non-stationary** data, additional features, known as **technical indicators**, are considered (easy to compute, used in many studies as features for financial ML forecasting even if their effectiveness is debated). Calculated using the Python Technical Analysis Library, which given OHLCV in input produces 90 technical indicators.

Methodology

- Feature engineering and selection (RF regressor utilized, as it is one of the most popular methods for stock market application)

  Useful for:

  - **Reducing** irrelevant variables, **Computational cost**, **overfitting** problem
  - Finding a suitable balance between a **small** number of features (possibility that information is not enough to make accurate predictions) and a **high** number of features (**curse of dimensionality**)

  Main steps:

  - **Dataset splitting**: 80% training, 20% testing
  - Model **initialization** and **training**
  - **Feature importance assessment**: features ranked according to their importance
  - **Thresholding**: features with importance >= 0.01 are kept

Methodology

- Datasets construction:

  data is divided into Training, Validation and Test sets in the following way:

  - Test set : **last 365** days of the dataset
  - Validation set: **28** days before the Test set
  - Training set: **remaining** points

  To improve the forecast Darts offers:

  - **Past covariates**: features known only into the past. They are set equal to the filtered technical indicators
  - **Future covariates**: features known in advance, into the future. They consists of temporal attributes (month, year, day, day of the week…)
  - **TARGET SERIES** (series to predict): series of **Close** prices

Methodology

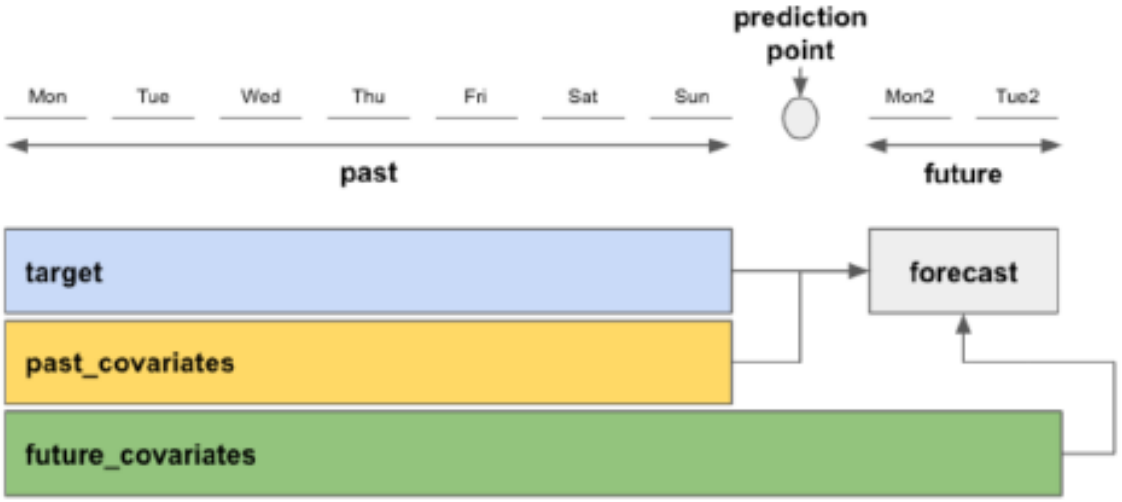


Figure 5: How forecasting models work with target and covariates for a prediction with forecast horizon n=2. Image from `https://unit8co.github.io/darts/userguide/covariates.html`

Methodology

- Hyperparameter optimization:
  - Performed only on **DL models** using the Python library Optuna

  The process involves:
  - Establishing an objective function, designed to minimize a certain metric (**MAE** in this case)
  - Within this function assign for each parameter a **search space**
  - The optimization is set to run for a maximum of **7200 seconds** or **100 trials**, whichever is reached first, and a callback function is used to implement early stopping if the validation loss does not improve by at least **0.001** for **5 consecutive epochs**
  - **High computational** cost, so hyperparameters found for Bitcoin are also employed for the other cryptocurrencies

Methodology

Algorithm 1 Hyperparameter Optimization Using Optuna

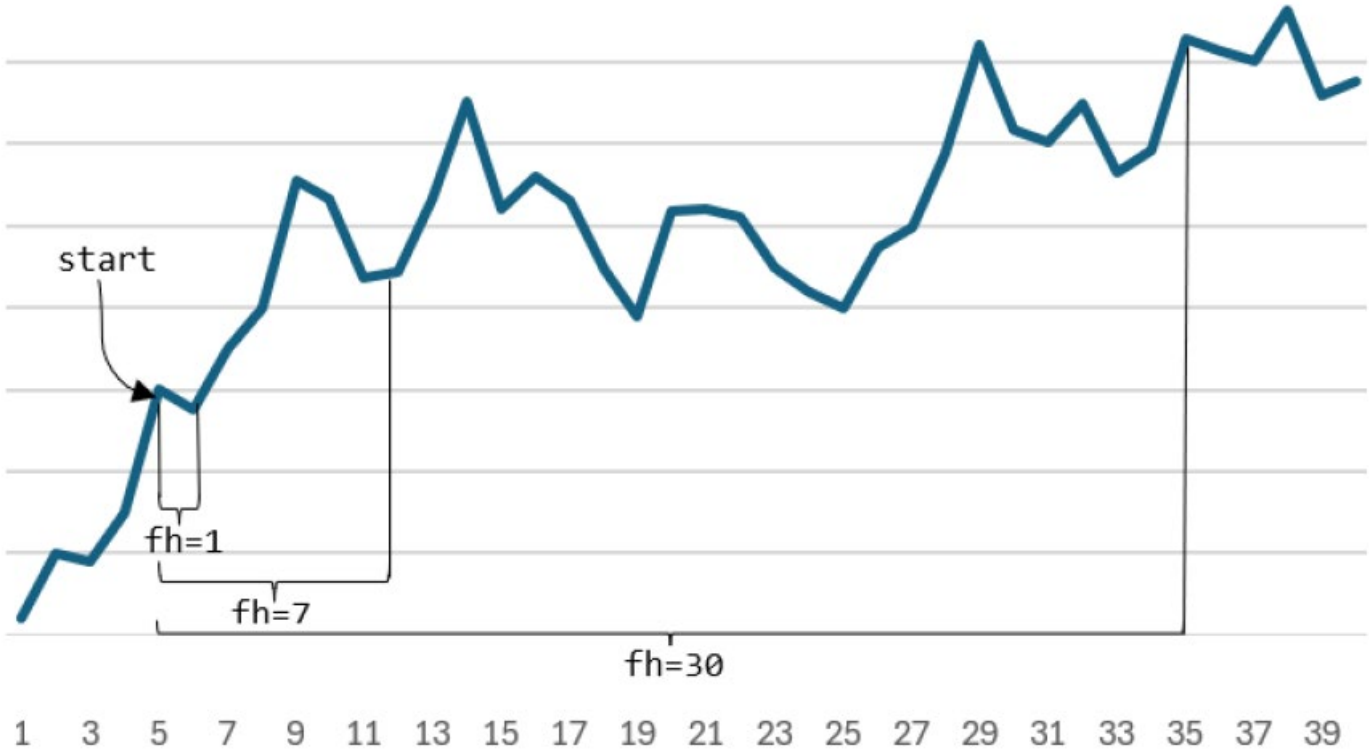
```
Define the objective function that returns the MAE
Configure the search space for hyperparameters
Create an Optuna study to minimize the objective function
Set maximum optimization time to 7200 seconds or 100 trials
for each trial in the study until max time or trials is reached do
    Optimize the trial
    Evaluate the trial's performance
    if the trial's performance is the best so far then
        Update the best hyperparameters
    end if
    if validation loss does not improve by at least 0.001 for 5 consecutive epochs then
        Trigger early stopping
        Break the loop
    end if
end for
Report the best trial's hyperparameters and performance
```

Experiments and results

Experiments and results

- Backtesting:

  Performed on the test sets of each cryptocurrencies (last 365 data points) for three forecast horizons: 1, 7, 30 days.



Experiments and results

Coin	BTC								
Forecast Horizon (days)	1			7			30		
Error metric	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
Naive Drift	476.25	**725.23**	**1.55**	**1328.98**	**1891.77**	**4.35**	3096.52	3882.04	9.72
Naive	**475.84**	725.66	1.55	1330.97	1901.69	4.35	3149.36	3991.02	9.83
Naive Seasonal	1331.75	1897.01	4.37	1330.97	1901.69	4.35	3585.09	4454.36	11.14
Random Forest	3280.62	3834.22	11.3	3662.35	4256.59	12.61	3859.26	4384.73	13.25
LightGBM	1712.21	2009.12	5.74	5354.74	5916.02	18.26	6670.08	7199.52	22.63
XGBoost	2854.8	3367.24	9.59	4639.52	5348.58	16.12	4712.71	5386.84	16.02
CatBoost	1909.29	2392.27	6.24	2571.12	3142.43	8.55	2781.89	3361.19	9.08
LSTM	1531.2	2064.73	5.09	1893.41	2484.03	6.35	1939.79	2544.59	6.37
RNN	1394.99	1673.56	4.58	1520.87	2149.27	4.93	**1556.06**	2214.82	**4.94**
GRU	1446.41	1624.32	4.94	1514.78	1931.07	5.25	1586.9	**1998.27**	5.42
Transformer	1291.32	1624.79	4.44	1918.76	2530.92	6.4	2109.36	2738.29	6.89
TFT	2440.7	2980.56	8.25	5379.69	6400.79	17.99	6931.85	7990.28	22.34
NBEATS	2294.18	2742.17	7.78	2953.69	3537.82	9.94	3129.18	3729.91	10.4
TCN	2556	3351.8	8.04	2575.08	3379.36	8.07	2730.76	3549.02	8.45

Table 4: Results for Bitcoin

Experiments and results

Coin	ETH								
Forecast Horizon (days)	1			7			30		
Error metric	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
Naive Drift	32.91	**47.99**	1.75	82.69	110.38	4.39	151.81	190.98	7.94
Naive	**32.88**	48	**1.74**	82.61	110.93	4.38	153.86	196.73	7.98
Naive Seasonal	82.14	110.28	4.37	82.61	110.93	4.38	169.81	217.13	8.79
Random Forest	45.1	61.11	2.43	**54.9**	**73.33**	**2.95**	**54.89**	**73.62**	**2.93**
LightGBM	54.45	68.17	2.94	141.61	172.95	7.64	337.15	468.86	17.39
XGBoost	65.7	92.23	3.56	175.87	218.38	9.37	377.6	514.31	18.85
CatBoost	100.28	128.01	5.34	236.24	298.6	12.3	244.11	324.78	12.61
LSTM	123.98	149.49	6.63	109.26	135.28	6	131.28	161.57	7.2
RNN	56.65	74.19	3.04	91.37	122.91	4.88	116.34	140.99	6.31
GRU	69.73	88.52	3.75	96.17	129.05	5.12	99.53	129.68	5.3
Transformer	119.24	141.9	6.31	118.04	152.59	6.34	151.56	191.16	8
TFT	142.89	182.9	7.6	272.15	349.5	14.1	267.81	348.87	13.73
NBEATS	160.02	199.41	8.85	231	297.61	12.84	287.03	363.68	15.71
TCN	147.35	195.3	8.09	147.14	194.72	8.06	217.2	271.93	11.99

Table 5: Results for Ethereum

Experiments and results

Coin	BNB								
Forecast Horizon (days)	1			7			30		
Error metric	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
Naive Drift	4.33	6.51	1.58	11.62	16.59	4.28	25.07	34.38	9.53
Naive	**4.33**	**6.5**	**1.58**	**11.47**	**16.54**	**4.22**	24.56	33.79	9.24
Naive Seasonal	11.41	16.46	4.2	11.47	16.54	4.22	26.31	36.35	9.94
Random Forest	11.09	13.77	4.49	14.06	17.4	5.72	**14.6**	**18.12**	6.02
LightGBM	17.71	22.75	7.39	30.76	36.13	12.31	31.7	36.91	12.81
XGBoost	14.79	19.19	6.2	31.27	38.26	12.54	36.78	45.47	14.64
CatBoost	12.21	14.8	4.81	28.32	33.45	11.39	30	35.26	12.18
LSTM	20.58	25.42	7.66	22.47	30.34	8.25	23.09	31.57	8.62
RNN	9.61	12.11	3.59	14.45	19.85	5.34	14.81	20.47	**5.53**
GRU	9.53	11.82	3.59	21.94	26.09	8.47	23.35	27.38	9.1
Transformer	14.49	18.09	5.34	22.06	28.08	8.24	22.56	28.43	8.54
TFT	42.39	50.29	16.63	51.26	62.4	20.07	63.52	77.55	25.76
NBEATS	16.31	19.57	6.1	36.38	45.33	13.52	37.52	47.24	14.1
TCN	23.52	30.99	8.52	23.6	31.19	8.56	25.21	33.61	9.23

Table 6: Results for Binance coin

Experiments and results

Coin	SOL								
Forecast Horizon (days)	1			7			30		
Error metric	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
Naive Drift	1.36	2.37	3.45	3.57	6.39	9.65	9.41	14.94	22.09
Naive	**1.36**	**2.37**	**3.44**	3.57	6.43	9.61	9.52	15.37	21.68
Naive Seasonal	3.54	6.38	9.56	3.57	6.43	9.61	10.58	16.81	23.86
Random Forest	1.95	3.46	5	**2.45**	**4.57**	**5.93**	**2.69**	**5.23**	**6.15**
LightGBM	3.39	7.99	6.6	11.27	21.58	30.69	24.93	31.45	86.52
XGBoost	3.24	5.94	7.68	11.19	18.4	31.15	17.19	21.95	55.96
CatBoost	10.66	11.75	43.99	14.08	15.12	58.47	17.2	18.46	69.2
LSTM	4.92	7.91	13.02	5.88	10.21	15.31	6.19	10.46	16.04
RNN	6.87	7.6	27.1	6.13	8.69	19.4	7.73	10	25.89
GRU	2.95	4.19	9.39	5.53	8.13	16.9	6.4	8.92	20.17
Transformer	6.83	10.48	16.65	17.25	20.12	55.59	21.11	25.21	64.27
TFT	8.38	9.59	32.43	13.76	16.91	48.31	14.12	17.68	48.28
NBEATS	11.7	13.8	43.78	13.68	17.53	44.45	16.48	20.67	52.23
TCN	6.06	8.76	20.28	6.22	8.8	21.08	7.57	10.32	25.1

Table 7: Results for Solana

Experiments and results

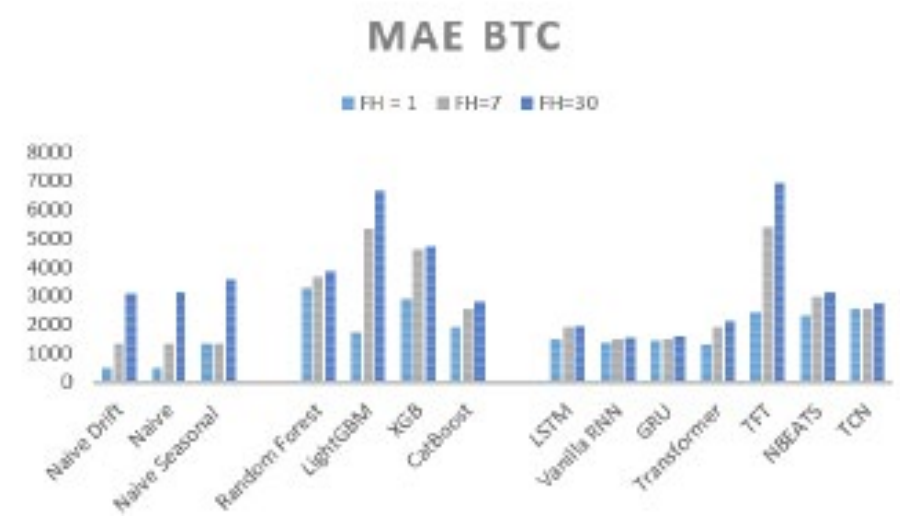
Coin	XRP								
Forecast Horizon (days)	1			7			30		
Error metric	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
Naive Drift	0.0118	0.0244	2.16	0.0332	0.0559	5.92	0.0803	0.1089	14.34
Naive	**0.0118**	**0.0243**	**2.16**	0.033	0.0558	5.89	0.0797	0.1087	14.19
Naive Seasonal	0.0328	0.0555	5.87	0.033	0.0558	5.89	0.086	0.1121	15.43
Random Forest	0.0187	0.0303	3.43	**0.0205**	**0.031**	**3.78**	**0.0212**	**0.0319**	**3.87**
LightGBM	0.0218	0.0328	3.97	0.0751	0.1055	13.29	0.1199	0.1499	21.26
XGBoost	0.0203	0.0323	3.68	0.0731	0.0985	12.83	0.1048	0.1292	18.54
CatBoost	0.0217	0.0329	3.98	0.0556	0.0799	10.05	0.0637	0.0895	11.38
LSTM	0.0459	0.0611	8.65	0.0582	0.0782	10.97	0.0634	0.0855	11.68
RNN	0.0202	0.0316	3.73	0.038	0.058	6.87	0.0424	0.0615	7.63
GRU	0.02	0.0334	3.7	0.0545	0.0738	9.87	0.0671	0.0853	12.1
Transformer	0.0302	0.0413	5.68	0.0527	0.0751	9.49	0.0677	0.0912	11.98
TFT	0.0379	0.0539	7.27	0.0966	0.1264	17.63	0.0941	0.1267	16.75
NBEATS	0.045	0.0603	8.73	0.1219	0.1511	24.54	0.1446	0.1755	28.05
TCN	0.0541	0.0788	10.32	0.0542	0.0779	10.3	0.0669	0.0908	12.17

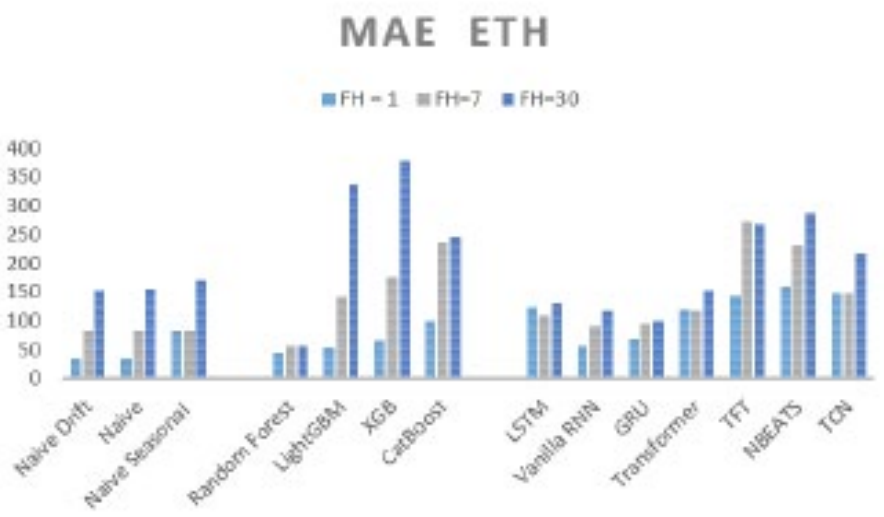
Table 8: Results for Ripple

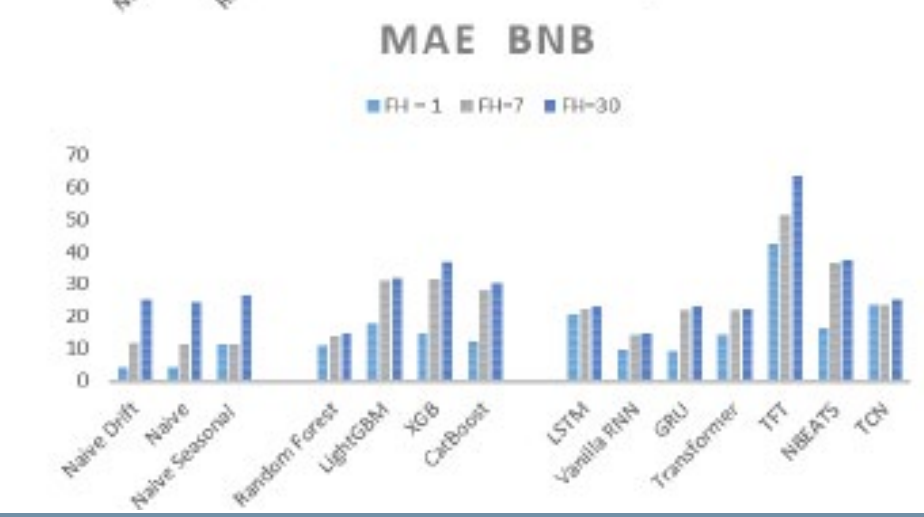
Experiments and results

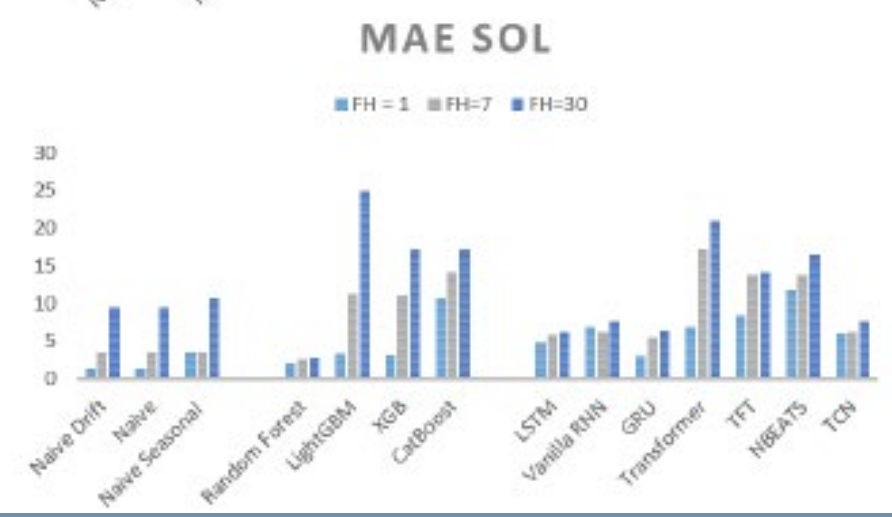
- Comments on the results:
  - **Naive models** perform **better** than **ML models** for **shorter forecast horizons**. For the **1-day** forecast horizon, they always outperform the other models, suggesting a **Random Walk** for this interval
  - For the **7-day** forecast horizon, **Naive** models are **better** than the ML models only for **Bitcoin** and **Binance Coin**.
  - For the **30-day** forecast horizon, **ML models** perform **better**. **Random Forest** performs the best for all the considered coins except Bitcoin. In this case **RNN** and its variant **GRU** achieve a better performance. (likely because greater amount of data).
  - in general, **simpler models perform better** both for Regression (RF the best) and the Deep Learning model categories (RNN, LSTM, GRU the best). Reasons may be **noisy, insufficient data, overfitting** and **hyperparameter tuning.**

Experiments and results

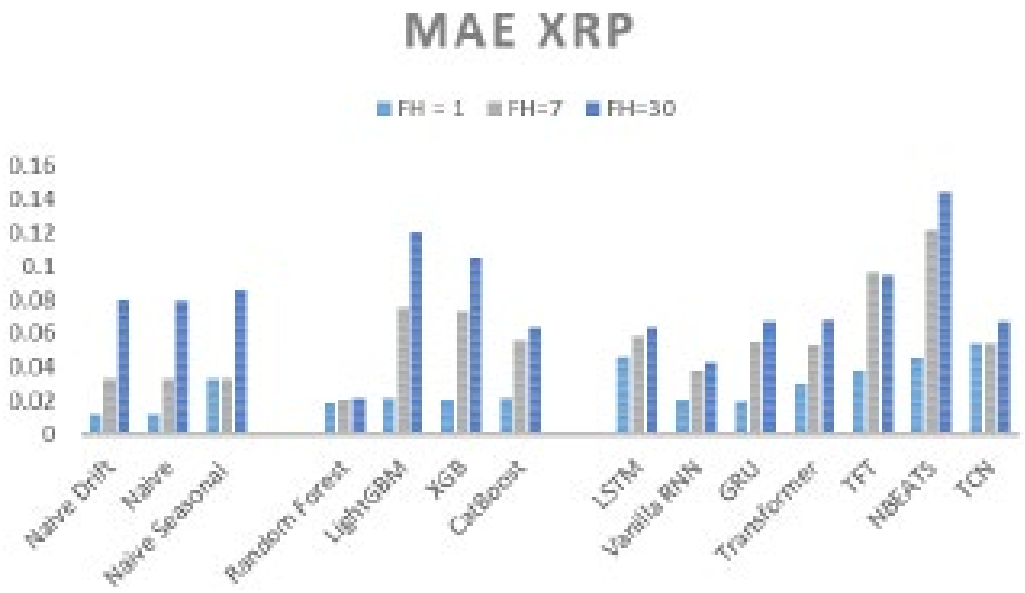








Experiments and results



Comparison with the existing literature

Comparison with the existing literature

- Comparison with 2 papers (same time intervals, dataset division, error metrics, coins, different models, features)
  - Time-series forecasting of Bitcoin prices using high-dimensional features: a machine learning approach" by Mudassir et al. (2020) (comparison_1 folder in github)
  - "Forecasting of Cryptocurrency Prices Using Machine Learning" by Derbentsev et al. (2020) (comparison_2 folder in github)
- For the first comparison results are too different (problems in the comparison paper code)
- For the second paper results are comparable.
  - Coins: **BTC, ETH, XRP**
  - Error metrics: **RMSE, MAPE**
  - Predictive models: **BART, MLP, RF**(only one in common) for three different time lags: **7,14, 21**
  - **1482** data **points** from **August 1, 2015**, **80% training**, **20% validation**, last **90 data points test set**
  - **one-step forecasting technique**, without adjusting the models' parameters. (as in the main experiment when forecast horizon = 1)

Comparison with the existing literature

- Results of the comparison:
  - For the comparison study **BART** and **MLP** perform the best, but these results are **worse or as good** as the results obtained by the **Naive Models** employed in this thesis
  - The predicted series in the comparative paper closely resemble the target series but **shifted** forward by **one day**, similar to predictions made by Naive models
  - These considerations suggest that **BART** and **MLP** may **not** have **adequately learned** from the past data, behaving like Naive Models. As in the main experiment for forecast horizon = 1, data seem to follow a **random walk pattern**

Comparison with the existing literature

Coin	BTC		ETH		XRP	
Error	RMSE	MAPE	RMSE	MAPE	RMSE	MAPE
Naive Drift	**512.74**	**3.58**	11.77	**3.38**	0.0151	3.13
Naive	512.91	3.59	**11.76**	3.39	**0.0151**	**3.12**
Naive Seasonal	1208.6	9.58	28.69	8.73	0.038	8.19
Random Forest	628.87	4.77	15.47	5	0.031	6.99
LightGBM	695.29	4.84	34.28	11.12	0.0468	10.79
XGBoost	857.84	6.47	19.47	6.69	0.0677	15.09
CatBoost	887.96	6.32	42.66	15.95	0.0745	19.58
LSTM	1358.27	9.87	83.91	32.81	0.071	17.85
RNN	1075.84	9.82	14.93	4.67	0.023	4.93
GRU	781.7	5.98	44.56	16.56	0.0352	7.93
Transformer	1235.22	9.5	116.06	47.87	0.0979	27.02
TFT	1636.61	12.12	81.16	29.76	0.0938	22.99
NBEATS	1091.85	9.65	103.45	41.67	0.0944	25.1
TCN	2197.98	17.97	90.01	33	0.0517	12.95

Table 12: Results obtained replicating Derbentsev et al.

Table 1 Out-of-sample accuracy performance results for different lags

	BTC		ETH		XRP	
	MAPE, %	RMSE	MAPE, %	RMSE	MAPE, %	RMSE
Lag $p = 7$						
BART-7	3.71	535.2	**3.39**	**11.74**	**3.07**	**0.0154**
RF-7	7.11	971.9	7.44	21.8	3.94	0.0196
MLP-7	**3.69**	**529.8**	3.53	12.17	**3.07**	**0.0153**
Lag $p = 14$						
BART-14	**3.83**	**541.9**	**3.37**	**11.86**	3.42	0.0167
RF-14	5.60	756.9	6.48	19.82	4.08	0.0203
MLP-14	3.95	559.1	3.51	12.16	**3.41**	**0.0162**
Lag $p = 21$						
BART-21	**3.94**	**558.5**	**3.69**	**12.55**	3.83	0.0183
RF-21	5.54	739.3	4.52	14.55	3.92	0.0212
MLP-21	4.28	610.8	3.84	13.17	**2.98**	**0.0151**

Figure 11: Results of Derbentsev et al. (2020)

Conclusions

Conclusions

- **Naive models** outperform more complex models for **one-day** forecast horizons
- For **extended forecast** periods of seven and thirty days, **complex models** perform the best, with Random Forest frequently emerging as the superior model.
- Why? One possible answer:
  - **Naive models better in short-term forecasting** could be attributed to the **random walk** characteristic of cryptocurrency prices at these intervals.
  - For **extended forecast horizons**, the **efficacy of ML models** possibly indicates of **broader trends** or **cycles** that these models can exploit

In summary these results seem to suggest that **cryptocurrency prices do not strictly follow a random walk process**, because there is a certain degree of **predictability** for extended forecast horizons.

- Future research proposals: **more advanced algorithmic approaches** (e.g. innovative feature engineering, new validation methods, new models or ensemble techniques).

Appendix

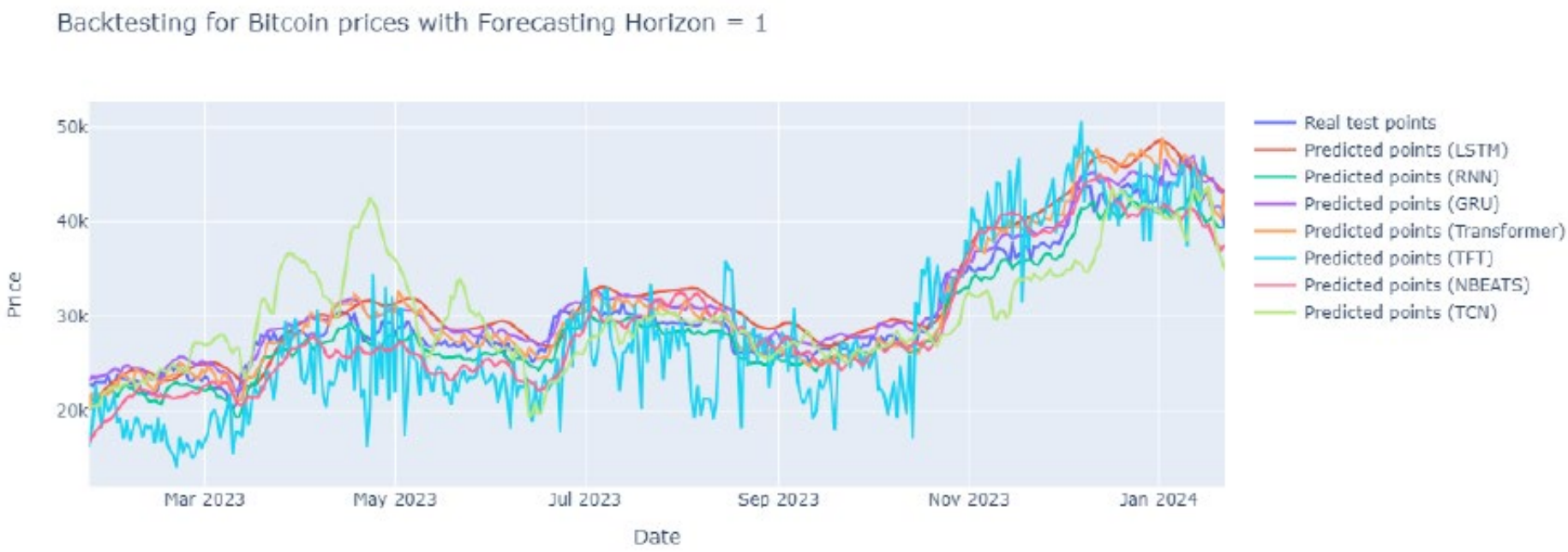
Appendix


Backtesting for Bitcoin prices with Forecasting Horizon = 1
Price
45k
40k
35k
30k
25k
20k
Mar 2023
May 2023
Jul 2023
Sep 2023
Nov 2023
Jan 2024
Date
Real test points
Predicted points (Naive)
Predicted points (Naive Drift)
Predicted points (Naive Seasonal)

Appendix


Backtesting for Bitcoin prices with Forecasting Horizon = 1
Price
45k
40k
35k
30k
25k
20k
Mar 2023
May 2023
Jul 2023
Sep 2023
Nov 2023
Jan 2024
Date
Real test points
Predicted points (Random Forest)
Predicted points (LightGBM)
Predicted points (XGBoost)
Predicted points (CatBoost)

Appendix


Backtesting for Bitcoin prices with Forecasting Horizon = 1
Price
50k
40k
30k
20k
Mar 2023
May 2023
Jul 2023
Sep 2023
Nov 2023
Jan 2024
Date
Real test points
Predicted points (LSTM)
Predicted points (RNN)
Predicted points (GRU)
Predicted points (Transformer)
Predicted points (TFT)
Predicted points (NBEATS)
Predicted points (TCN)

Thanks for the attention