Analysis of Different Machine Learning and Econometric Models to Predict Crypto Currencies High-Frequency Behaviour

Oğuzhan PINAR

Jan 12, 2023

Brief Summary

PURPOSE: Developing models to predict whether the price of a coin will increase in the next candlestick (kline) or not (binary label detection)

DEVELOPED MODULES: Data Collection, Feature extraction, and Modelling.

MODELS: Machine learning: XgBoost Classifier, RandomForest Classifier

Econometric models: Logistic Regression

BEST PERFORMANCE: XgBoost Classifier with 77% accuracy score.

Developed modules: 1. Data Collection

- Used klines data for 22 different coins
- **Data source:** Binance API (allows you to get klines data in any frequency)
- **Target variable:** close - open > 0
- 1 minute data for 52 days.

```
klines_data_dict["BTCUSDT"]
```

timestamp	timestamp	open	high	low	close	volume
2022-11-01 00:00:00	1667250000000	20401.49000000	20404.67000000	20395.13000000	20403.51000000	199.00681000
2022-11-01 00:01:00	1667250060000	20404.62000000	20410.92000000	20399.73000000	20407.85000000	132.22353000
2022-11-01 00:02:00	1667250120000	20407.85000000	20410.90000000	20400.00000000	20408.44000000	62.67916000
2022-11-01 00:03:00	1667250180000	20408.44000000	20411.44000000	20406.18000000	20408.71000000	38.52220000
2022-11-01 00:04:00	1667250240000	20408.71000000	20414.99000000	20406.50000000	20413.75000000	47.37171000
...	...	...	...	...	...	...
2022-12-22 23:56:00	1671742560000	16778.61000000	16786.87000000	16772.03000000	16785.43000000	160.45849000
2022-12-22 23:57:00	1671742620000	16785.43000000	16788.99000000	16781.80000000	16784.69000000	95.95904000
2022-12-22 23:58:00	1671742680000	16784.69000000	16789.42000000	16781.31000000	16782.91000000	151.90584000
2022-12-22 23:59:00	1671742740000	16782.91000000	16786.48000000	16779.16000000	16781.42000000	151.81699000
2022-12-23 00:00:00	1671742800000	16780.24000000	16795.41000000	16780.24000000	16792.50000000	204.20915000

74881 rows × 6 columns

Developed modules: 1. Data Collection

Biggest challenge: 500 rows per request

Solution: Divide and conquer:

- Divide the total duration to 500 rows sub-durations.
- Make a request for each sub-duration.
- Merge all results.

```
1 my_data_constructor.create_start_end_dates("1m")
```

	temp_starts	temp_ends
0	2022-11-01 00:00:00	2022-11-01 08:19:00
1	2022-11-01 08:20:00	2022-11-01 16:39:00
2	2022-11-01 16:40:00	2022-11-02 00:59:00
3	2022-11-02 01:00:00	2022-11-02 09:19:00
4	2022-11-02 09:20:00	2022-11-02 17:39:00
...	...	...
145	2022-12-21 08:20:00	2022-12-21 16:39:00
146	2022-12-21 16:40:00	2022-12-22 00:59:00
147	2022-12-22 01:00:00	2022-12-22 09:19:00
148	2022-12-22 09:20:00	2022-12-22 17:39:00
149	2022-12-22 17:40:00	2022-12-23 00:00:00

150 rows × 2 columns

Developed modules: 1. Data Collection

2nd challenge: Time complexity

Solution: Parallel Processing

- Allows you to initialize multiple threads at the same time.
- 5 times faster

```python
begin_time = dt.datetime.now()

start = dt.datetime(2022,11,1)
end = dt.datetime(2022,12,23)

my_data_constructor = data_constructor()
coin_list = ["BTC", "ETH", "BNB", "DOGE", "ADA", "MATIC", "DOT", "TRX", "LTC", "SOL", "UNI",
             "AVAX", "LINK", "XMR", "ATOM", "ETC", "XLM", "ALGO", "VET", "NEAR", "HBAR"]

klines_data_dict = {}
threads_dic = {}
pool = ThreadPool(processes=5)

#Start pooling:
for coin in coin_list:
    symbol = coin + "USDT"
    async_result = pool.apply_async(my_data_constructor.get_klines_data, args = (symbol, "1m", start, end))
    threads_dic[symbol] = async_result


#Get results:
for coin in coin_list:
    symbol = coin + "USDT"
    returned_df = threads_dic[symbol].get()
    klines_data_dict[symbol] = returned_df


time_cost_pooling = dt.datetime.now() - begin_time
```

Developed modules: 2. Feature Extraction

I used following features in the models:

1. Price_level(open) - profit (last 5 lags) - volume (last 5 lags) - range (last 5 lags)
2. The labels and profit of 5 min - 15 min - 1 hour - 12 hours - 1 day candlesticks
3. Other currencies (market) weighted labels (weight with volume)
4. Weekend - weekday dummies
5. Day time dummies (divide the day into 6 parts where each part is 4 hours)
6. Trend

The most important part: Prevent data leakage from future!

Developed modules: 3. Modelling

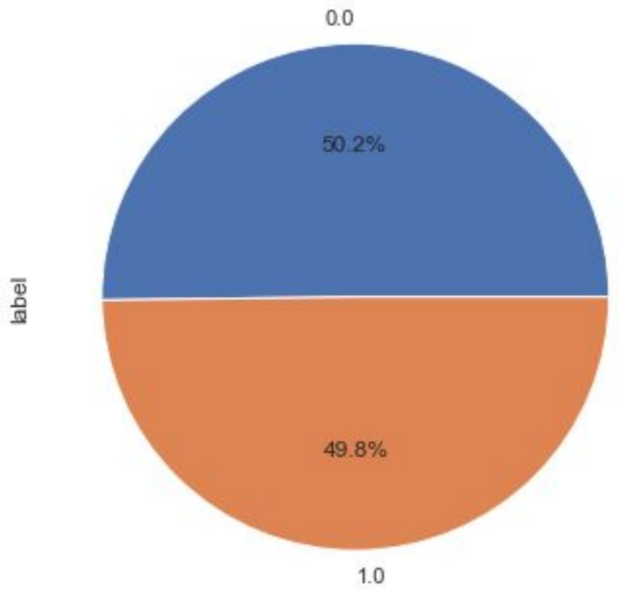
Modelling steps are:

1. Checking the data
2. Train - test split
3. Grid search with cross validation on train data (hyperparameter tuning)
4. Model fitting
5. Performance analysis on test data

I did parameter tuning only for BTCUSD, and used best parameters to create model for all 21 coins.

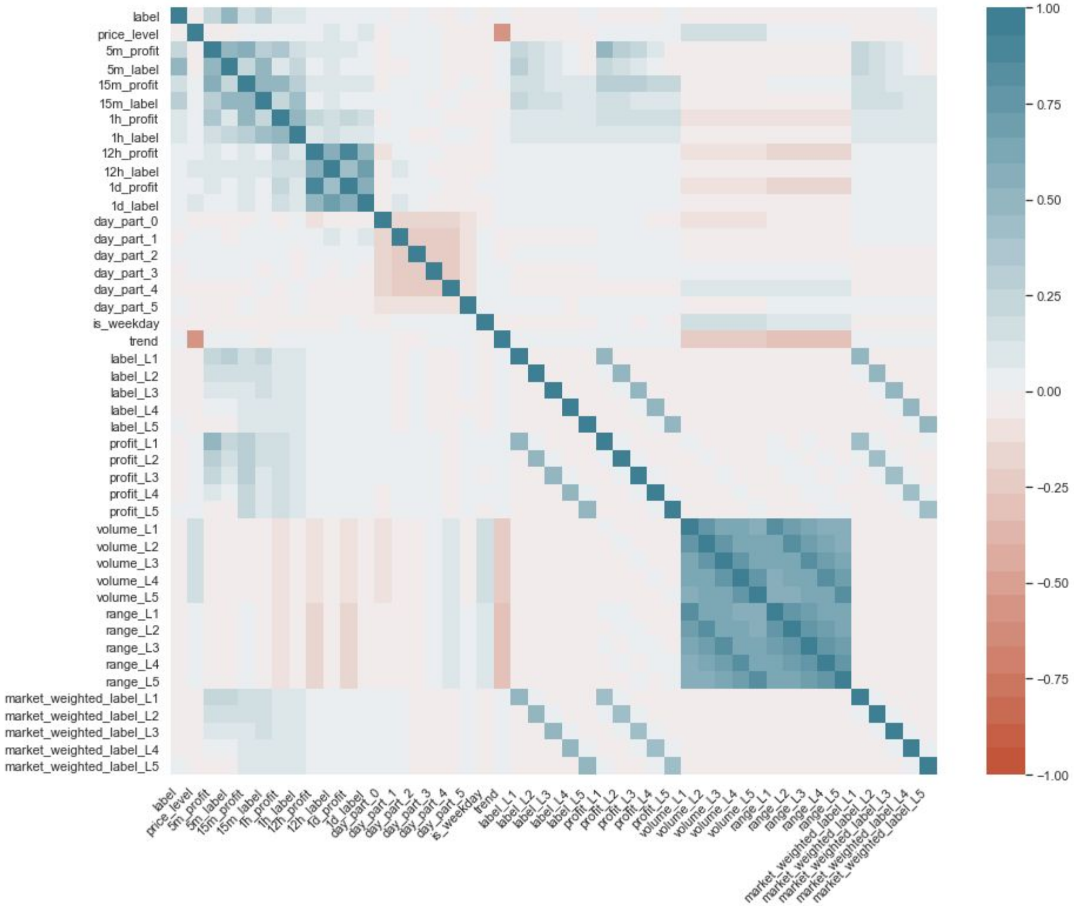
Developed modules: 3. Modelling

Descriptive Data Analysis:



Developed modules: 3. Modelling

Descriptive Data Analysis:



Developed modules: 3. Modelling

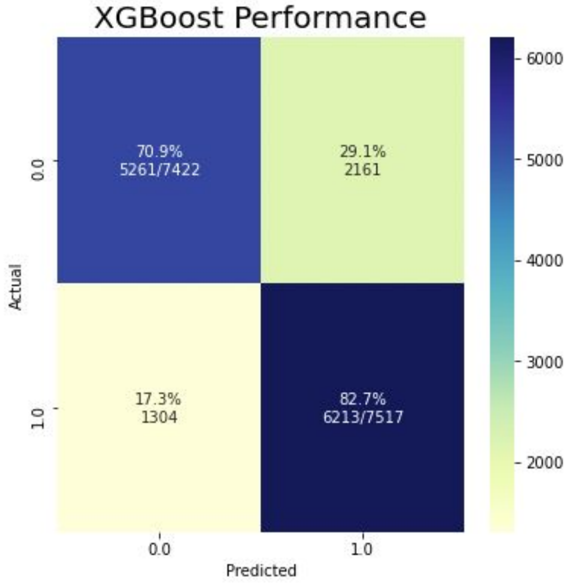
Modelling steps:

1. Train test split: 80% of data is train set - last 20% is test set
2. Grid search with cross validation on train data (hyperparameter tuning)
3. Model fitting with best parameters
4. Performance analysis

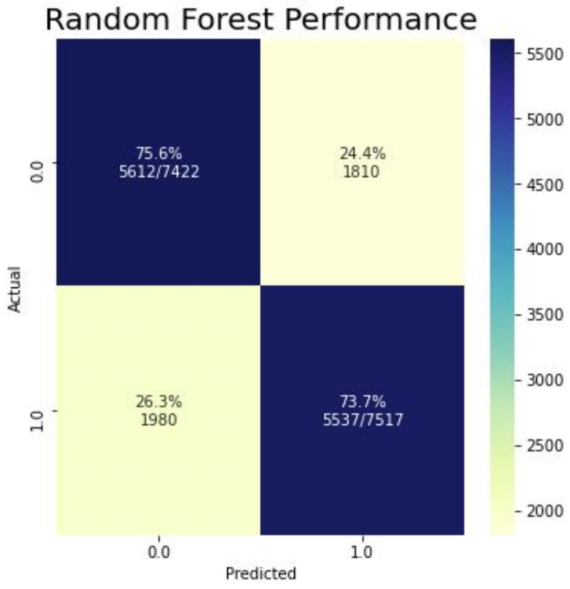
Developed modules: 3. Modelling

Model performances for BTCUSDT:

Accuracy: 0.7680567641743089



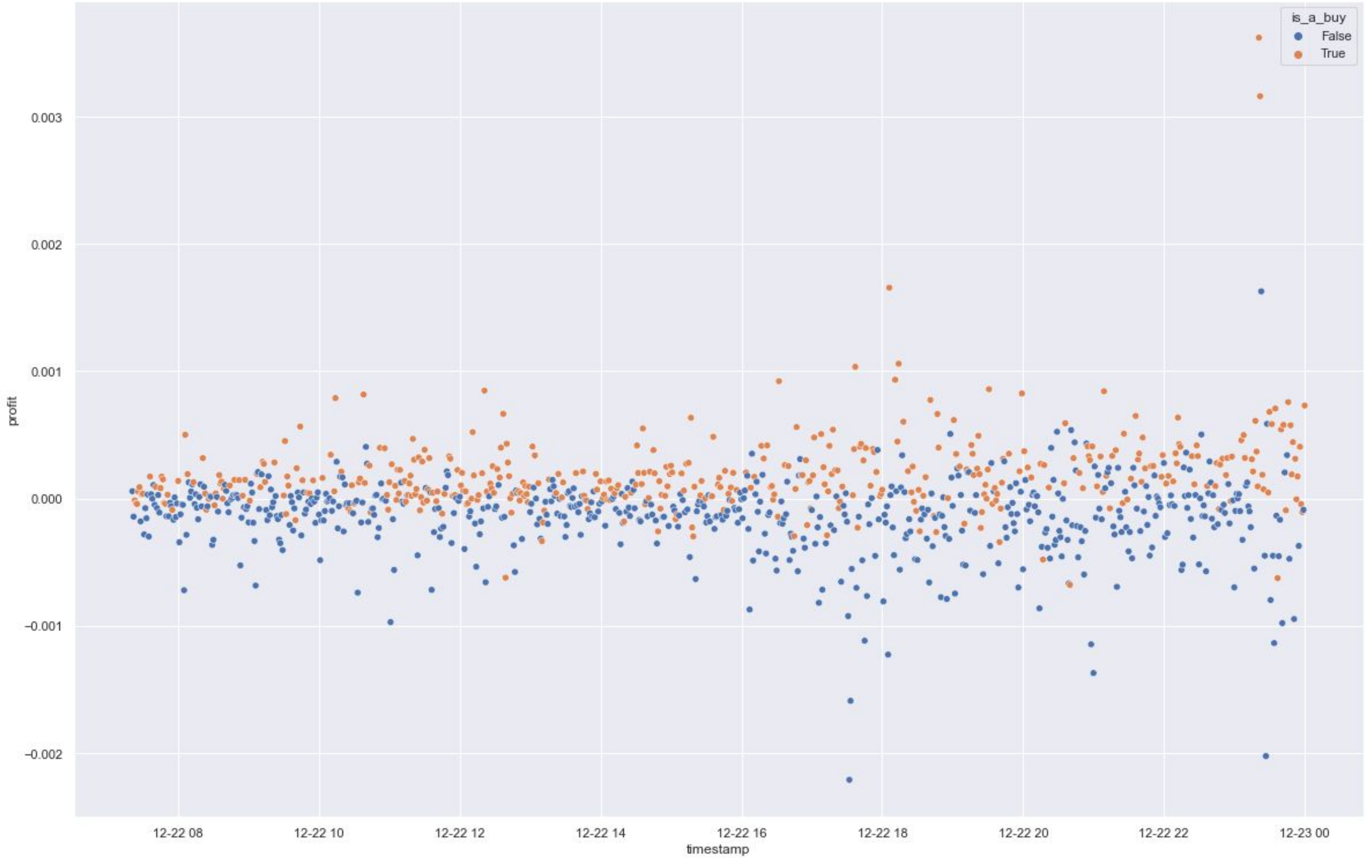
Accuracy: 0.7463016266149006



Accuracy: 0.5035142914519044

Developed modules: 3. Modelling

Model predictions for BTCUSDT:



Developed modules: 3. Modelling

Model performances (accuracy) in test sets for each coin is like following:

```
{'BTCUSDT': 0.77,     'ETHUSDT': 0.77,     'BNBUSDT': 0.74,
 'DOGEUSDT': 0.78,    'ADAUSDT': 0.76,     'MATICUSDT': 0.76,
 'DOTUSDT': 0.64,     'TRXUSDT': 0.77,     'LTCUSDT': 0.77,
 'SOLUSDT': 0.72,     'UNIUSDT': 0.69,     'AVAXUSDT': 0.77,
 'LINKUSDT': 0.76,    'XMRUSDT': 0.81,     'ATOMUSDT': 0.76,
 'ETCUSDT': 0.78,     'XLMUSDT': 0.83,     'ALGOUSDT': 0.78,
 'VETUSDT': 0.81,     'NEARUSDT': 0.77,    'HBARUSDT': 0.88}
```

Results & Further Improvements:

Results:

- XGBoost is the best performing model
- RandomForest predictions are the safest
- Logistic Regression performs very poorly

Further Improvements:

- Wider hyperparameter tuning can be applied.
- Backtesting and nowcasting analysis should be done to see the actual profitability of the model.
- Profit rates could be used as performance metric instead of accuracy while tuning the model
- Some other features can be included (like macro data (inflation unemployment etc.), data from stock markets...)

Thank you for listening

Oğuzhan PINAR