Crypto Forecasting App - Project Outline

Connor Capitolo
David Assaraf
Tale Lokvenec

Harvard John A. Paulson School of Engineering and Applied Sciences
IACS Institute for Applied Computational Science

Outline

- Project Scope
- Project Workflow
- Process/Data Flow
- Backend Infrastructure
- App Design
- Data
- Models

Problem Definition

The current state of the crypto market is extremely volatile. Due to lack of experience/involvement from traditional actors, there is a lack of systematic investment strategies in the crypto environment; therefore, there is an opportunity to extract value from an accurate prediction of the price dynamics of pairs.

The scope of this project is to create a Proof of Concept to see if there is opportunity when using Deep Learning in crypto markets.

Objectives

1. Bridging the lack of structure dealing with crypto exchanges in building a scalable and modular database architecture that will gather various features from different exchanges (starting with Binance) for ‘pairs’ (a ‘pair’ refers for instance to the dynamics of the market for BTC vs USDT)

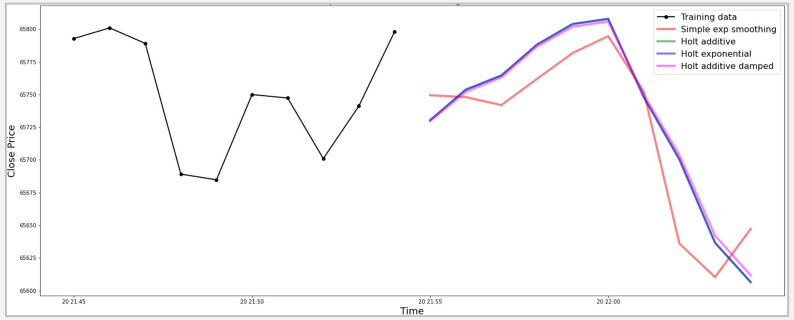
2. Building a predictive ML/DL model using real-time predictions that will enable us to gain insights as to how the market is evolving over time in order to inform trading decision making

Final Product (Tentative)

Home

Crypto Forecasting

Example Graph (Close Price on Y-axis, Time on X-axis)



BTCUSDT

Select Crypto Exchange
(drop-down)

Close Price Predictions

$X.xx	$X.xx	$X.xx	$X.xx
1 min.	5 min.	15 min.	30 min.
$X.xx	$X.xx	$X.xx	$X.xx
1 hour	1 day	1 week	1 month

Project Scope

Proof Of Concept (POC)

- Setup database infrastructure, gathering both the historical data and the real-time data from Binance exchange
- Perform data exploration and data processing
- Experiment on some baseline models: last price, exponential smoothing
- Develop training pipeline for one specific pair (BTC-USDT)
- Benchmarking our model against baseline models

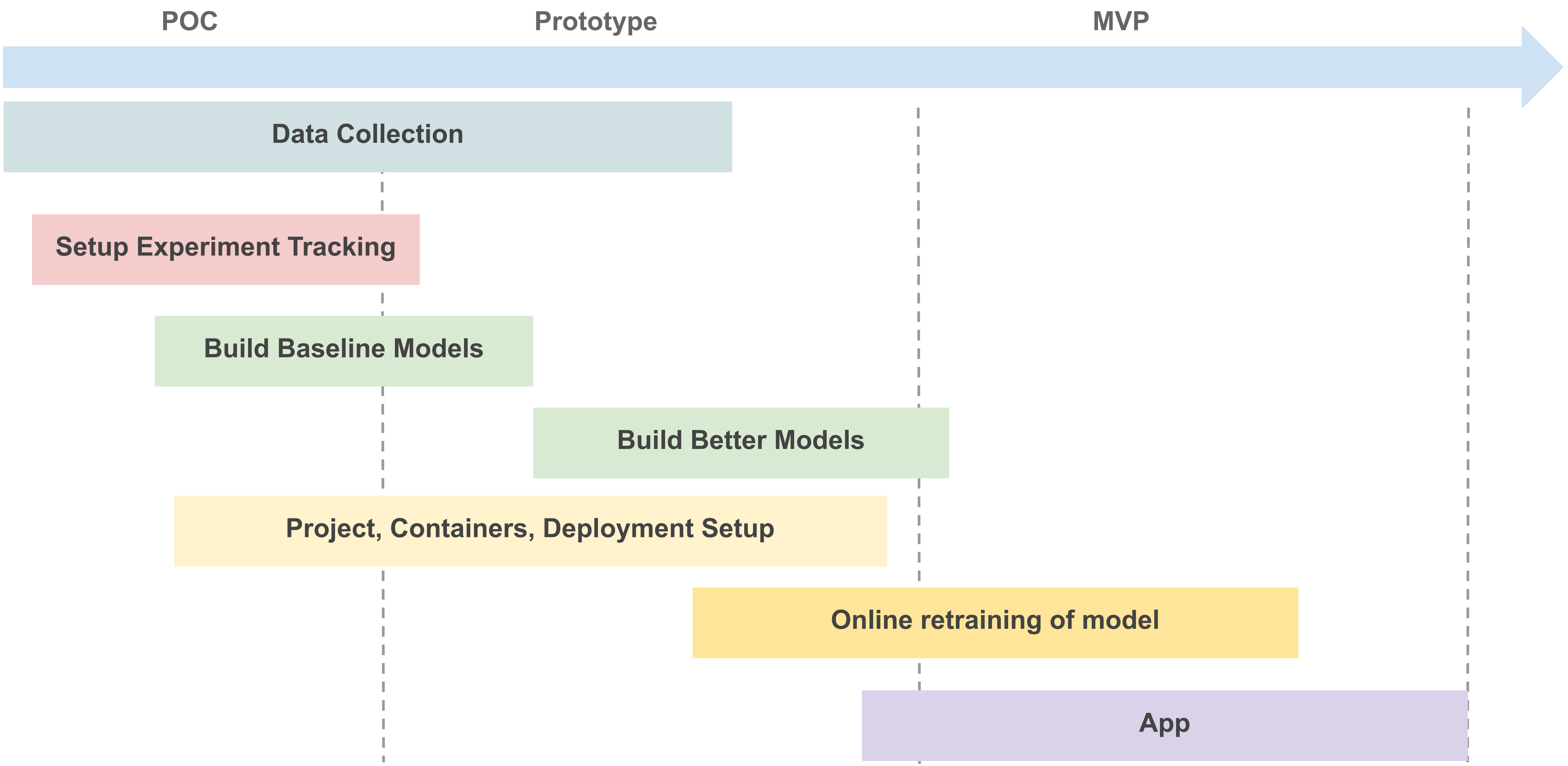
Prototype

- Develop a more advanced model, significantly improving over the different baselines
- Create a mockup of screens to see how the app will look
- Be able to launch the query of new pairs from the frontend side
- Perform regular data tests in order to verify the quality of the data
- Deploy models utilizing FastAPI to serve model predictions

Minimum Viable Product (MVP)

- Setup the front-end in order for users to interact with the API
- Provide real-time predictions using the deployed model
- Provide recommendations for user's investment
- Perform regular re-training of the deployed models

Project Workflow



Process (People)

- Collect initial data from Binance API
- Keep querying real-time data
- EDA on initial data
- Time Series modeling (single/multiple prediction approach)
- User selects a certain pair to obtain historical data and predictions
- View prediction results

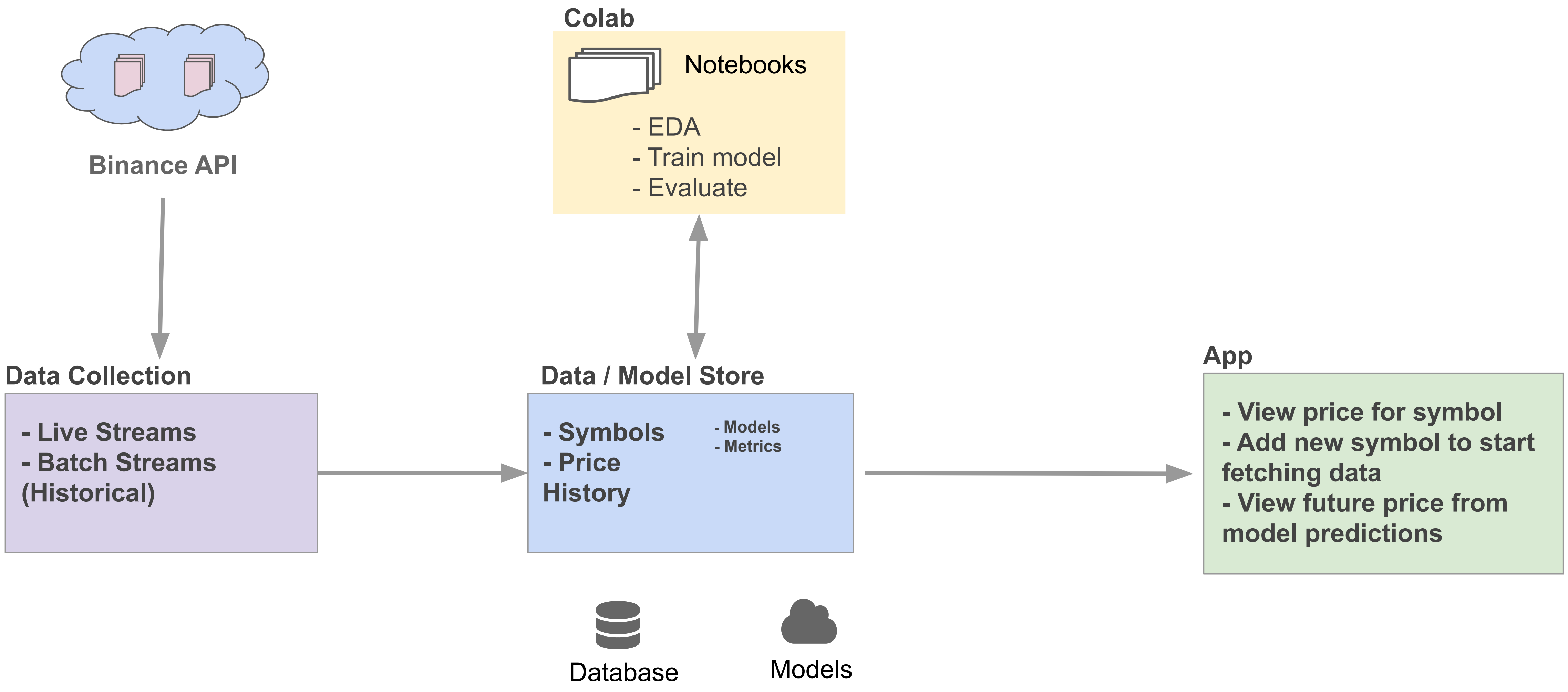
Execution (Code)

- Preprocess the initial data for both time-series and tabular modeling
- Use the best model to make prediction
- Return results to user as a minute-by-minute predictions over 24 hour period
- Track best model

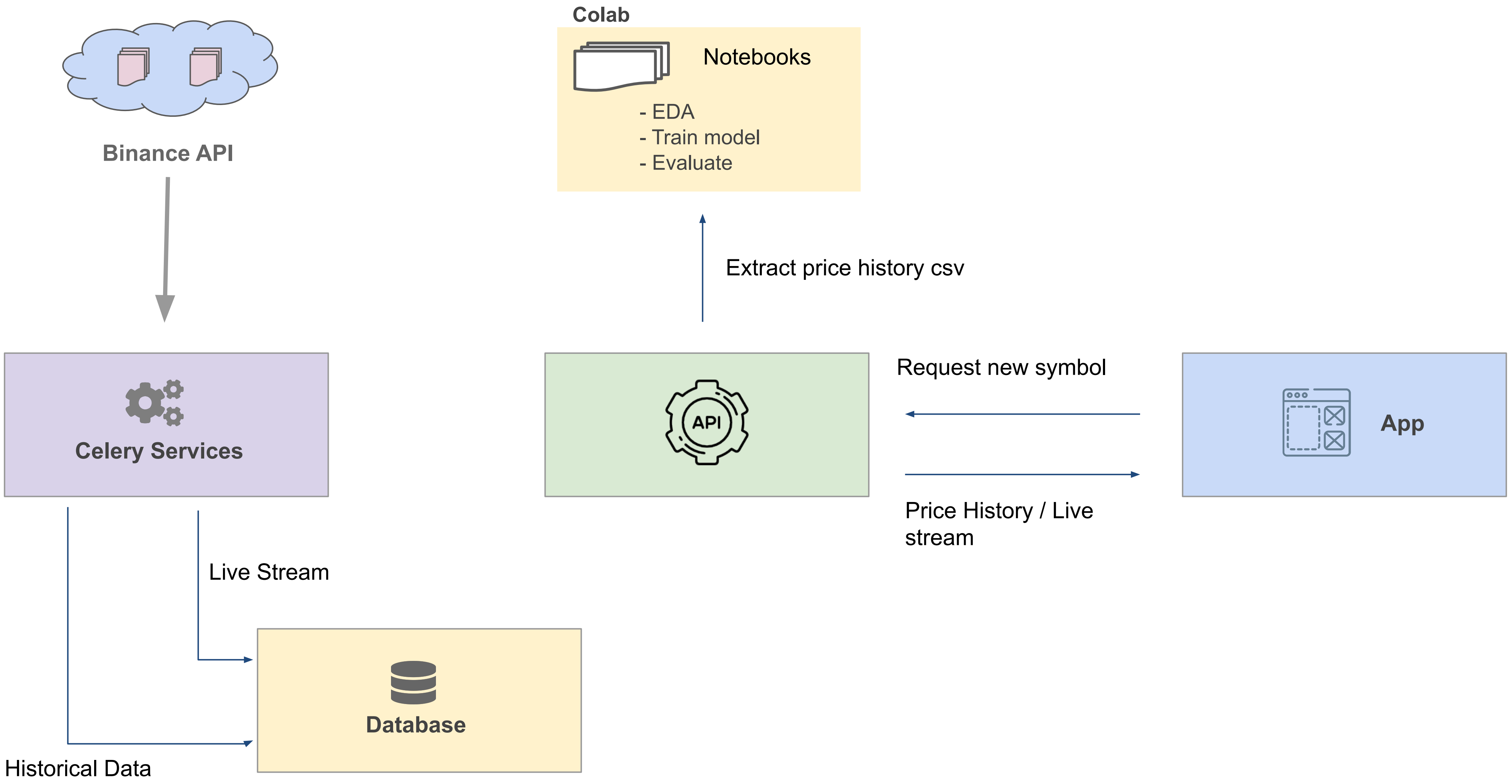
State (Source, Data, Models)

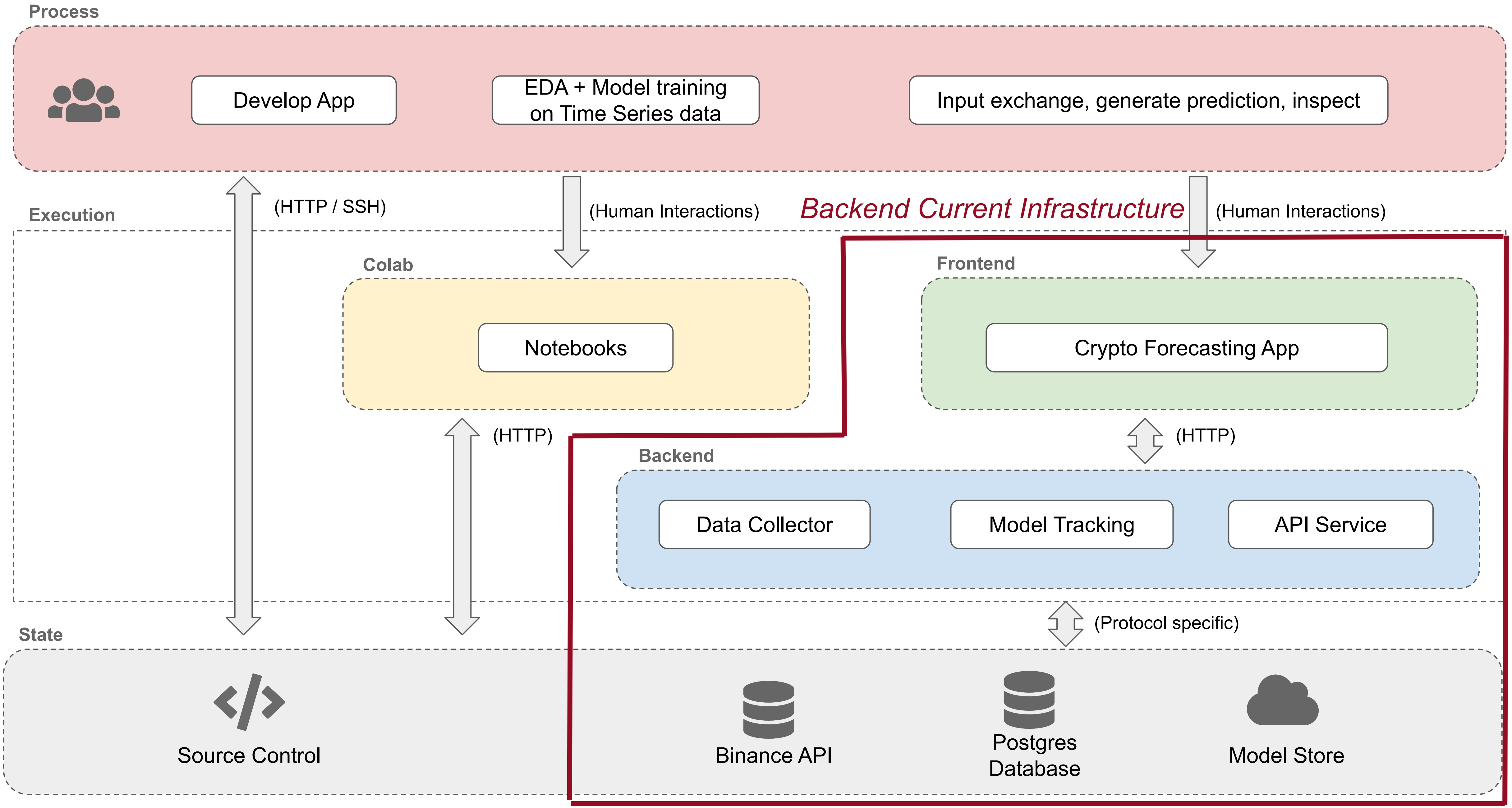
- Save data to a common store and keep updating
- Create tables and save in Postgres database
- Save model weights
- Information on pre processing

Process Flow



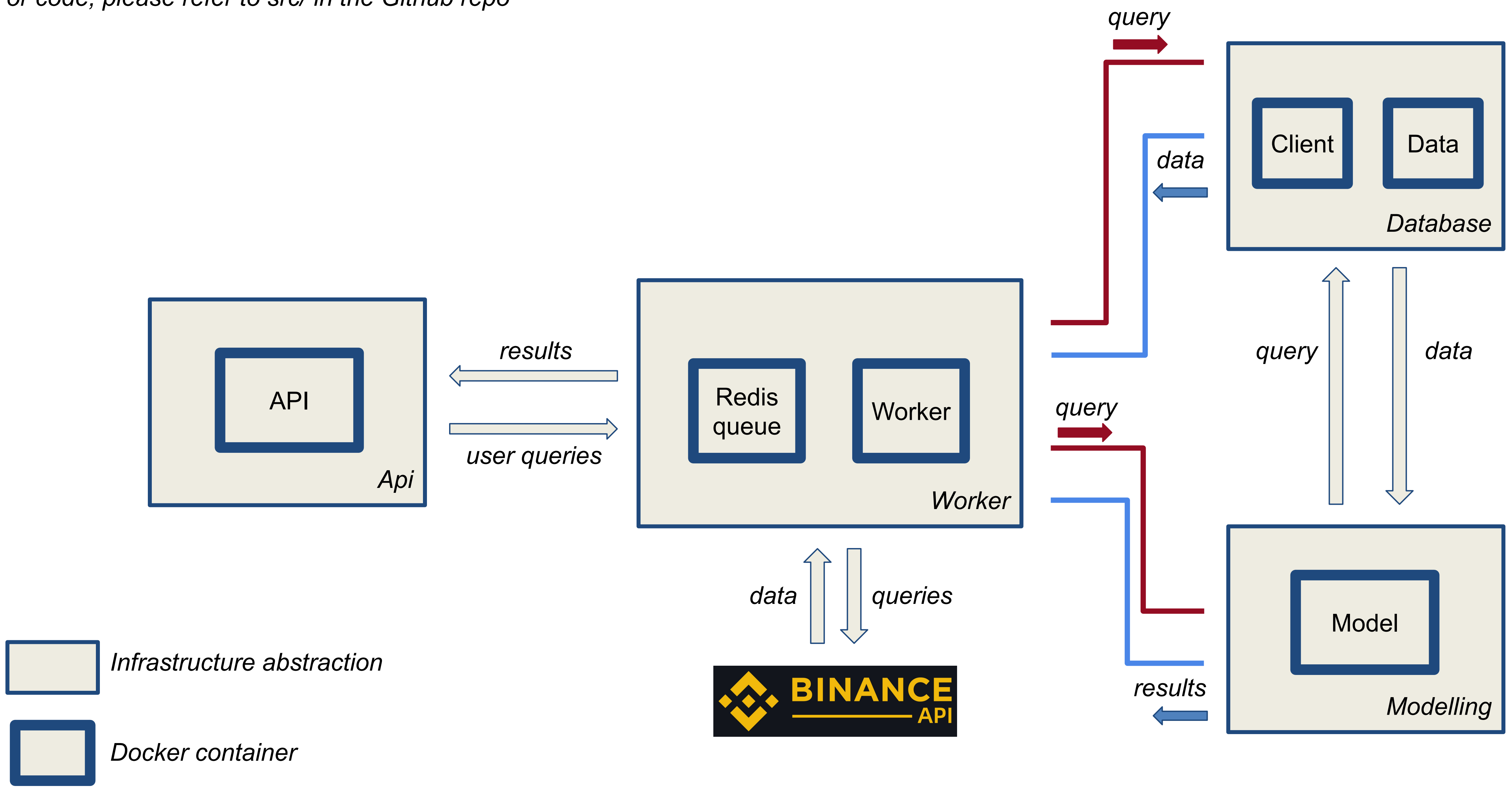
Data Flow


Colab
Notebooks
- EDA
- Train model
- Evaluate
Binance API
Extract price history csv
Request new symbol
Celery Services
API
App
Price History / Live stream
Live Stream
Database
Historical Data

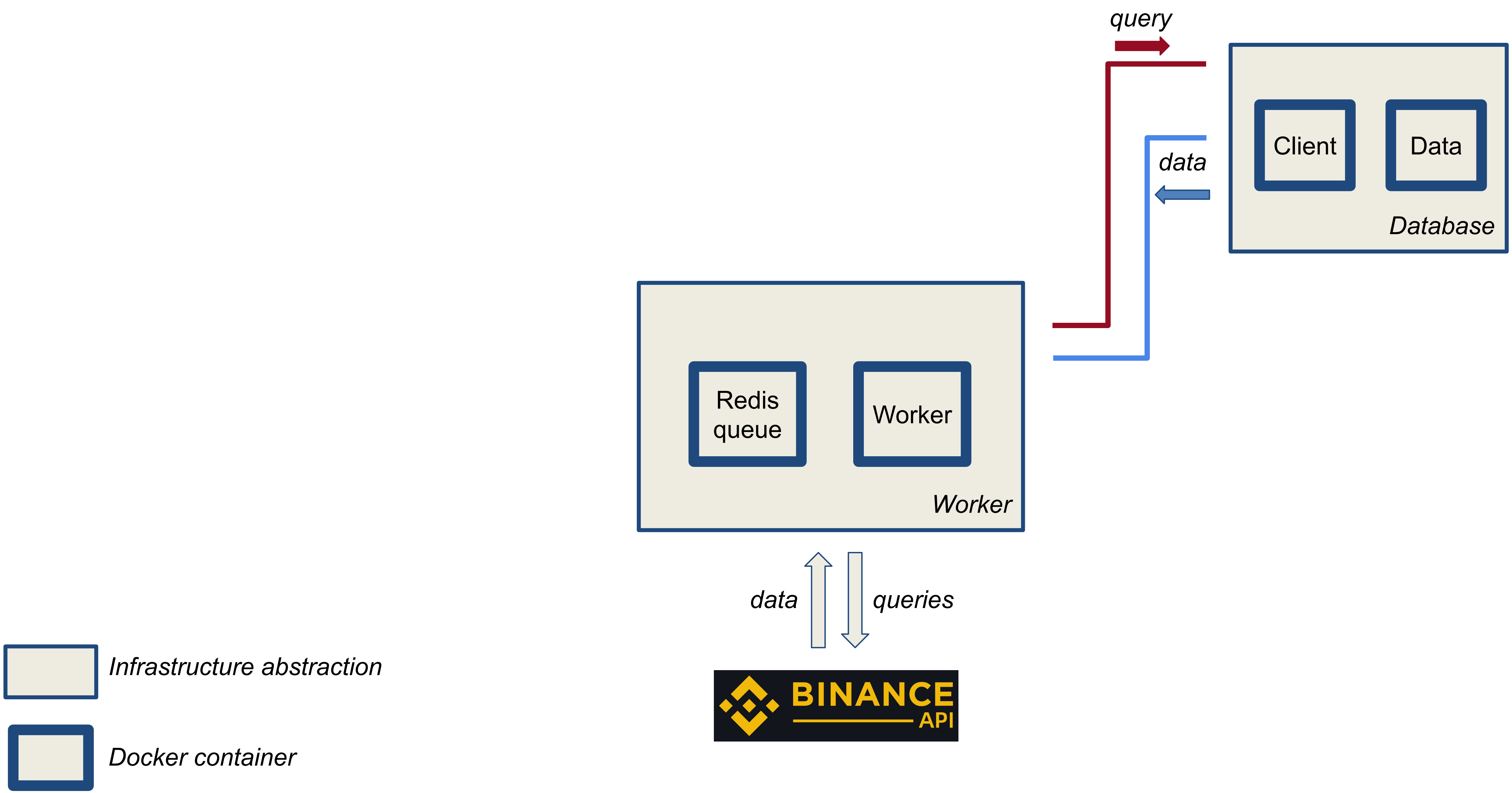

Process
Develop App
EDA + Model training on Time Series data
Input exchange, generate prediction, inspect
Execution
(HTTP / SSH)
(Human Interactions)
Backend Current Infrastructure
(Human Interactions)
Colab
Notebooks
Frontend
Crypto Forecasting App
(HTTP)
(HTTP)
Backend
Data Collector
Model Tracking
API Service
(Protocol specific)
State
Source Control
Binance API
Postgres Database
Model Store

Backend Current Infrastructure

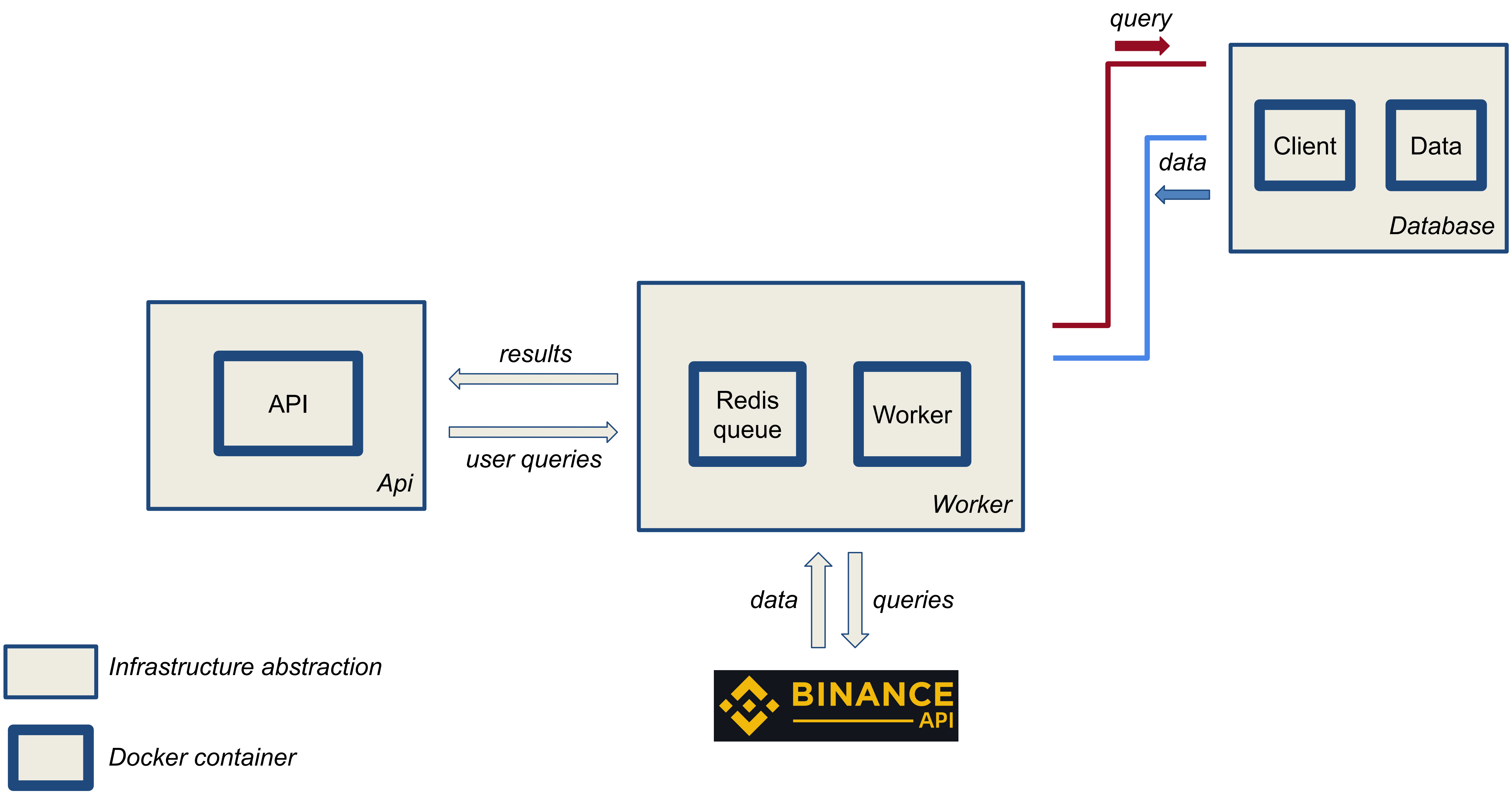
For code, please refer to src/ in the Github repo



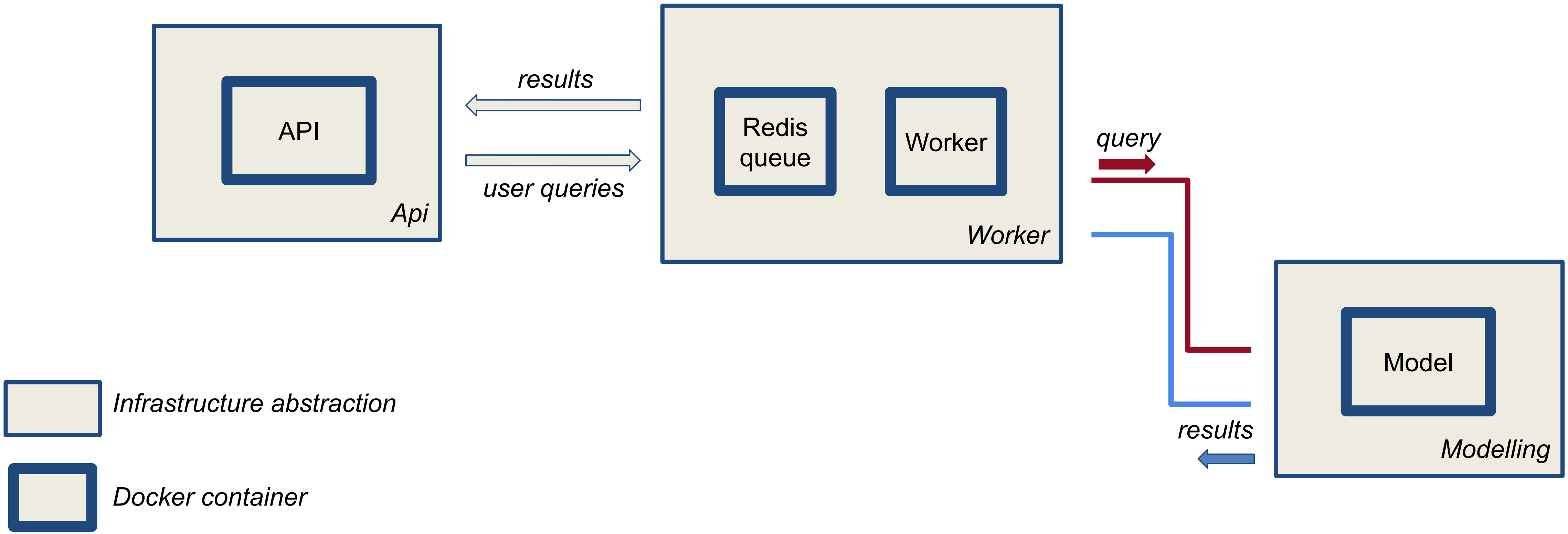
Backend Current Infrastructure: Update data with online streams



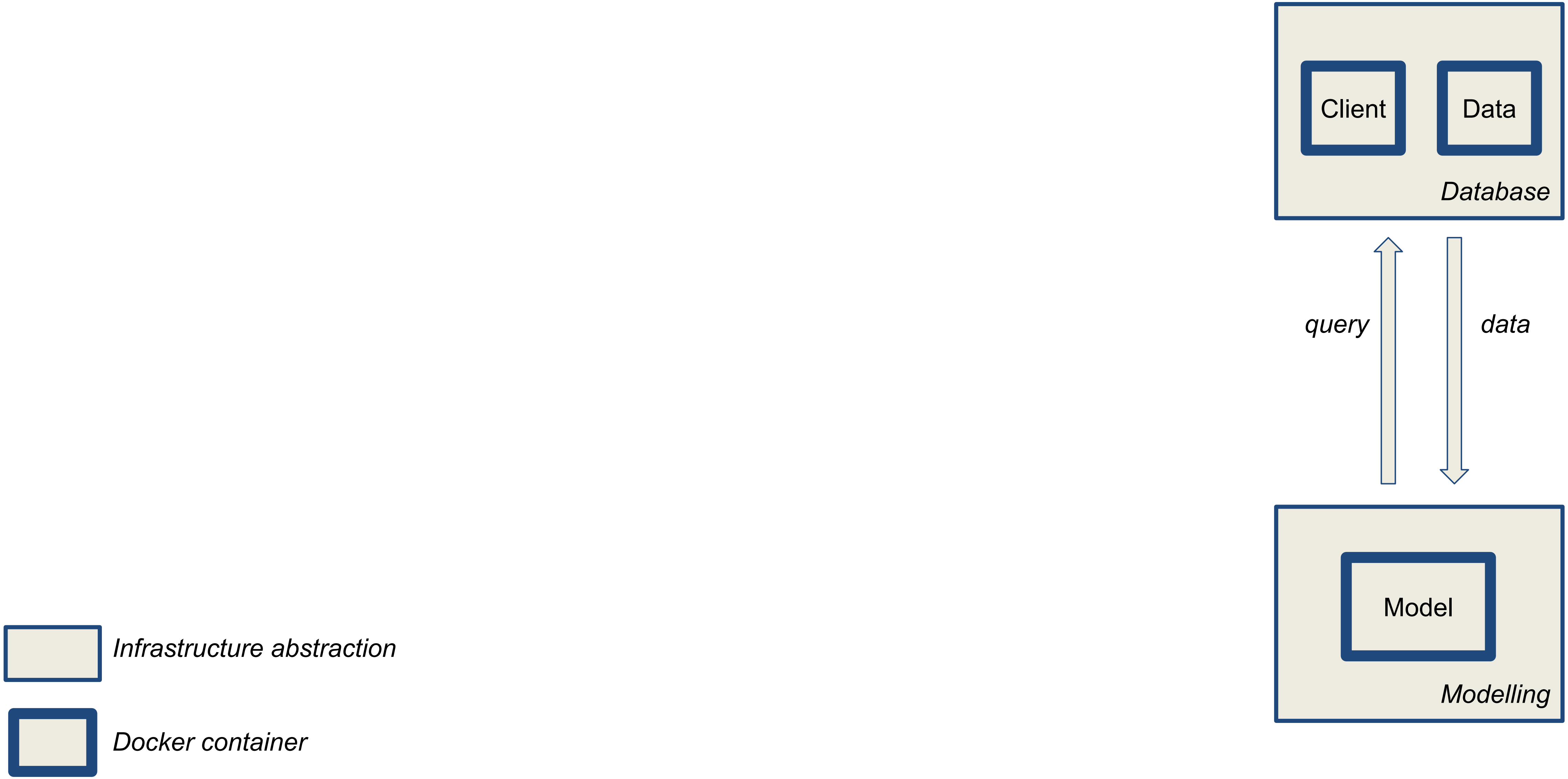
Backend Current Infrastructure: Update data with new pairs



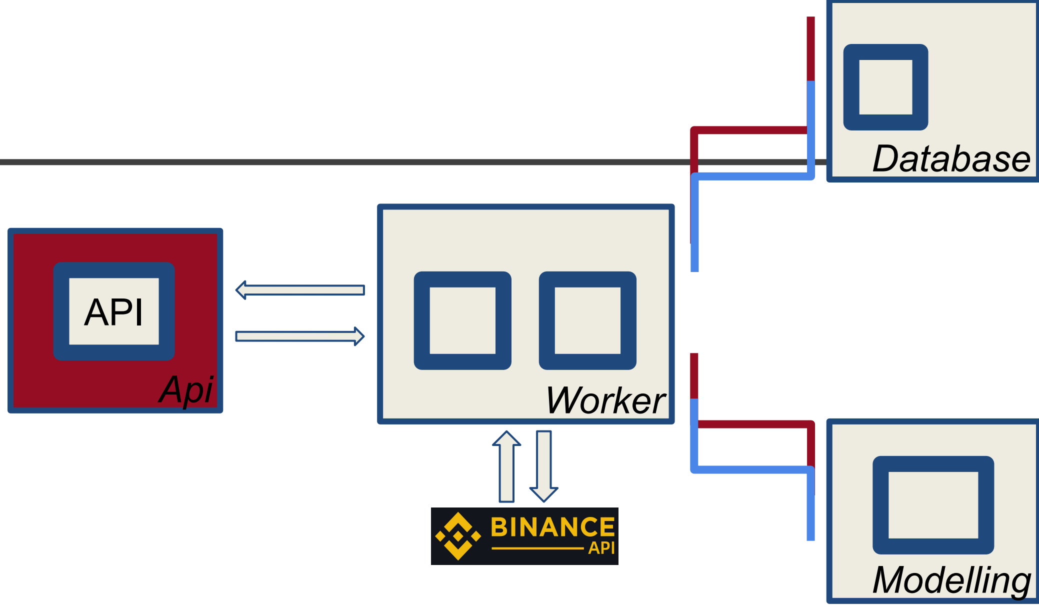
Backend Current Infrastructure: Get Model predictions



Backend Current Infrastructure: Model retraining

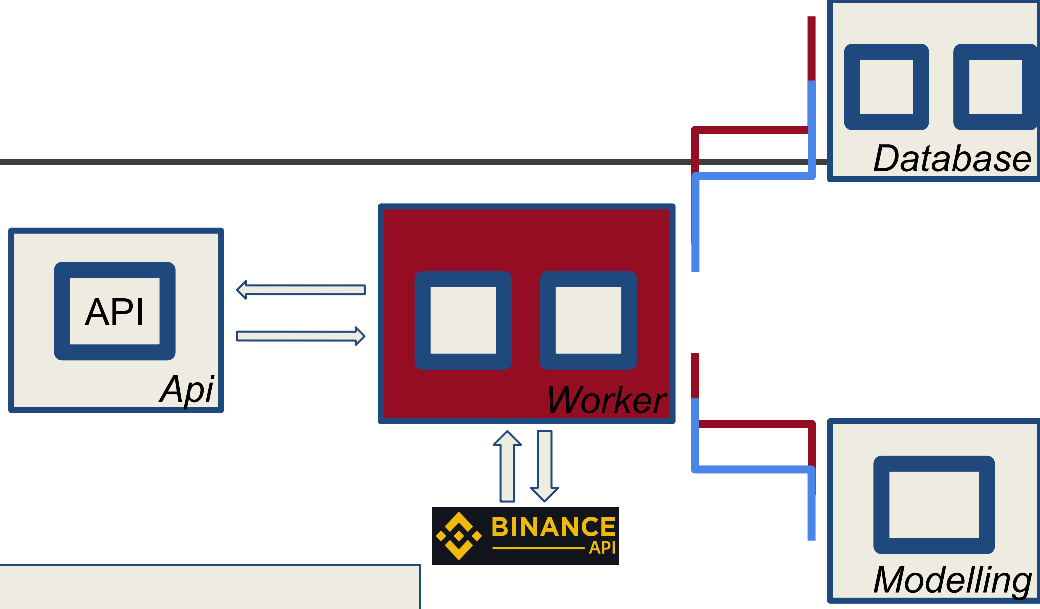


Current Infrastructure: API



- Use FastAPI implementation of REST API
  - Implementation of the different tasks in routers
- Use Uvicorn in order to host the server
- Use Celery as a Task Manager
  - Takes into account the different sequences of events that need to be run

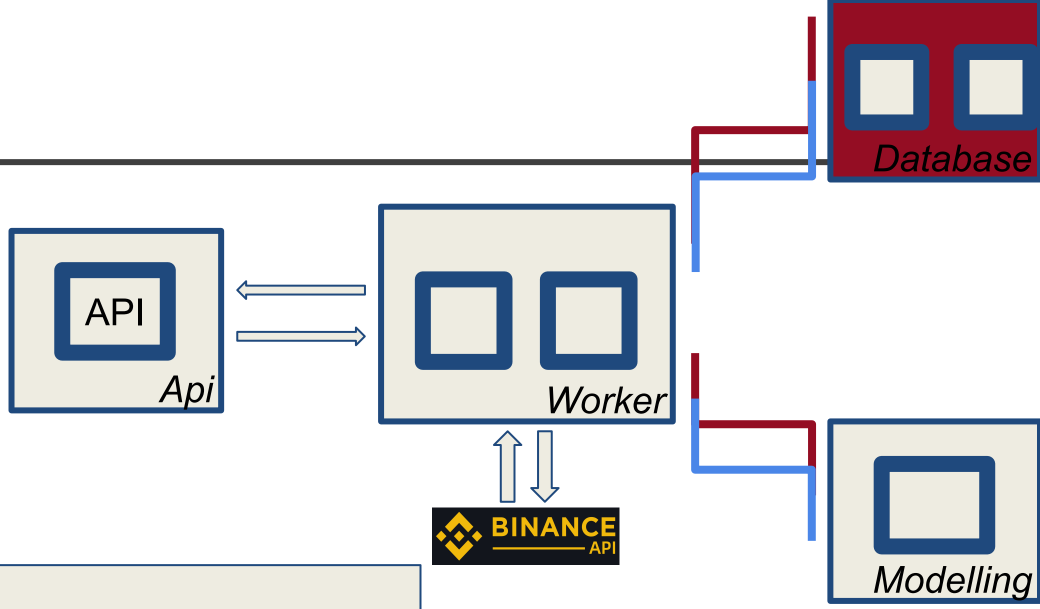
Current Infrastructure: Worker



Redis queue: non persistent queue, gathering the tasks that the API has been instructed to do.

Worker: interacting with Redis and Celery Task Manager in order to execute actions

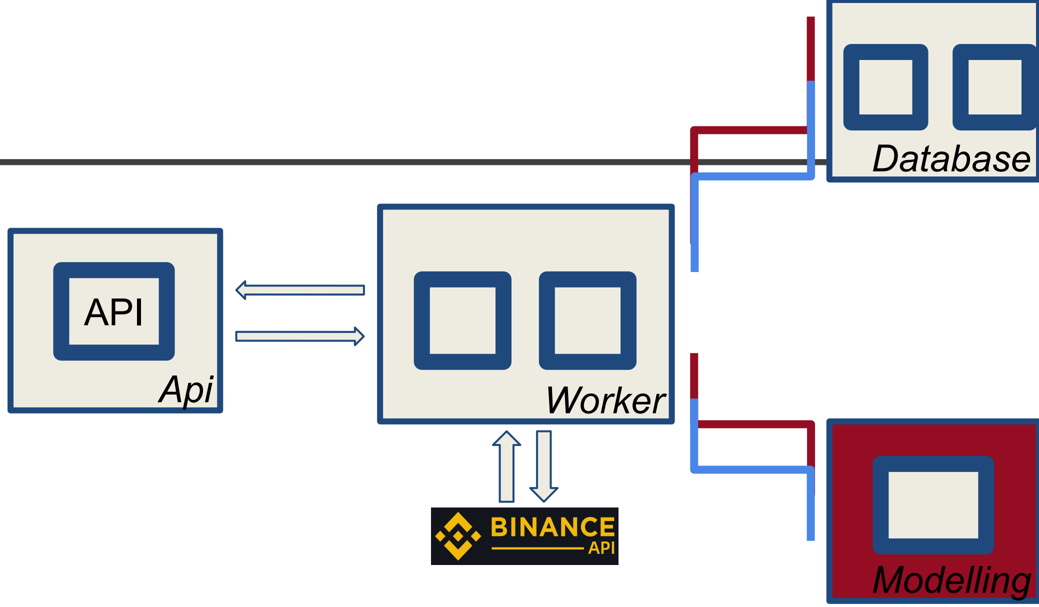
Current Infrastructure: Database



Client: migration scripts that interact with the database. Uses amacneil version of dbmate in order to perform data migration

Database: Postgresql database

Current Infrastructure: Model



Work in Progress

Dataset(s)

For code, please refer to src/api_for_data_download in the Github repo

- Dataset(s):
  - Historical data queried from Binance API (dtype: candlesticks)
    - earliest timestamp for the BTC-USDT pair: 2017-08-17 04:00:00
  - Real time data updating from Binance API (dtype: candlesticks) using websockets

- Dataset(s) size:
  - Number of datasets: 1,612 (one dataset per pair)
  - Size of dataset per exchange (~0.3Gb)
  - Total dataset(s) size (~500Gb)

- For the EDA and the initial modelling phases focus on the BTC-USDT pair
- The modelling will be extended to all 1,612 pairs for the final app

Dataset(s) Features From BTC-USDT Pair

Candle Feature	Description
Open Time	Candle Open Time
Open	Open Price in Quote (Secondary) Asset Units
High	High Price in Quote (Secondary) Asset Units
Low	Low Price in Quote (Secondary) Asset Units
Close	Close Price in Quote (Secondary) Asset Units
Volume	Total Trade Volume in Base (Primary) Asset Units
Close Time	Candle Close Time
Qupte Asset Volume	Total Trade Volume in Quote (Secondary Asset Units
Number of Trades	Total Number of Trades
Taker Buy Base Asset Volume	Taker (Matching Existing Order) Buy Base Asset Volume
Taker Buy Quote Asset Volume	Taker (Mathcing Existing Order) Buy Quote Asset Volume
Ignore	Safe to Ignore

Dataset(s) Quality

Data Types:

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2188604 entries, 0 to 2188603
Data columns (total 12 columns):
 #   Column                        Dtype
---  ------                        -----
 0   Open Time                     int64
 1   Open Price                    float64
 2   High price                    float64
 3   Low Price                     float64
 4   Close Price                   float64
 5   Volume Traded                 float64
 6   Close Time                    int64
 7   Quote asset Volume            float64
 8   Number of Trades              int64
 9   Taker buy base asset volume   float64
 10  Taker buy quote asset volume  float64
 11  NA                            float64
dtypes: float64(9), int64(3)
memory usage: 217.1 MB
```

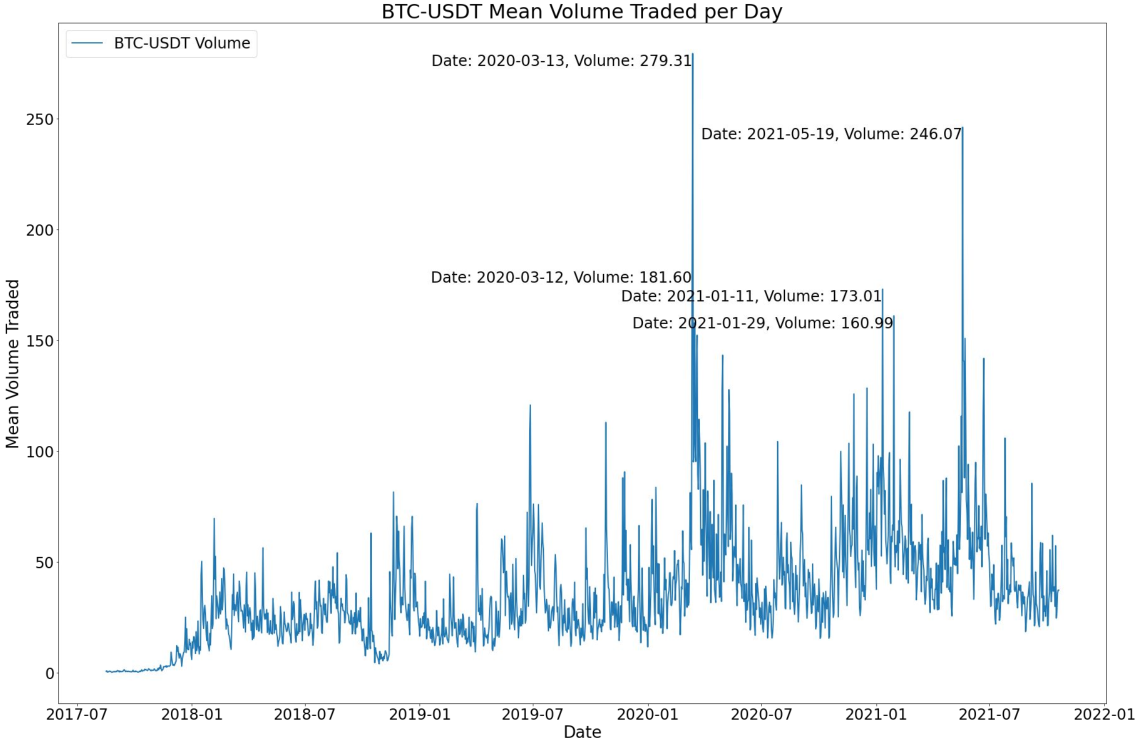
Missing Values:

```
Open Time                       0
Open Price                      0
High price                      0
Low Price                       0
Close Price                     0
Volume Traded                   0
Close Time                      0
Quote asset Volume              0
Number of Trades                0
Taker buy base asset volume     0
Taker buy quote asset volume    0
NA                              0
dtype: int64
```

EDA - Time Series Data

Mean Price per Day

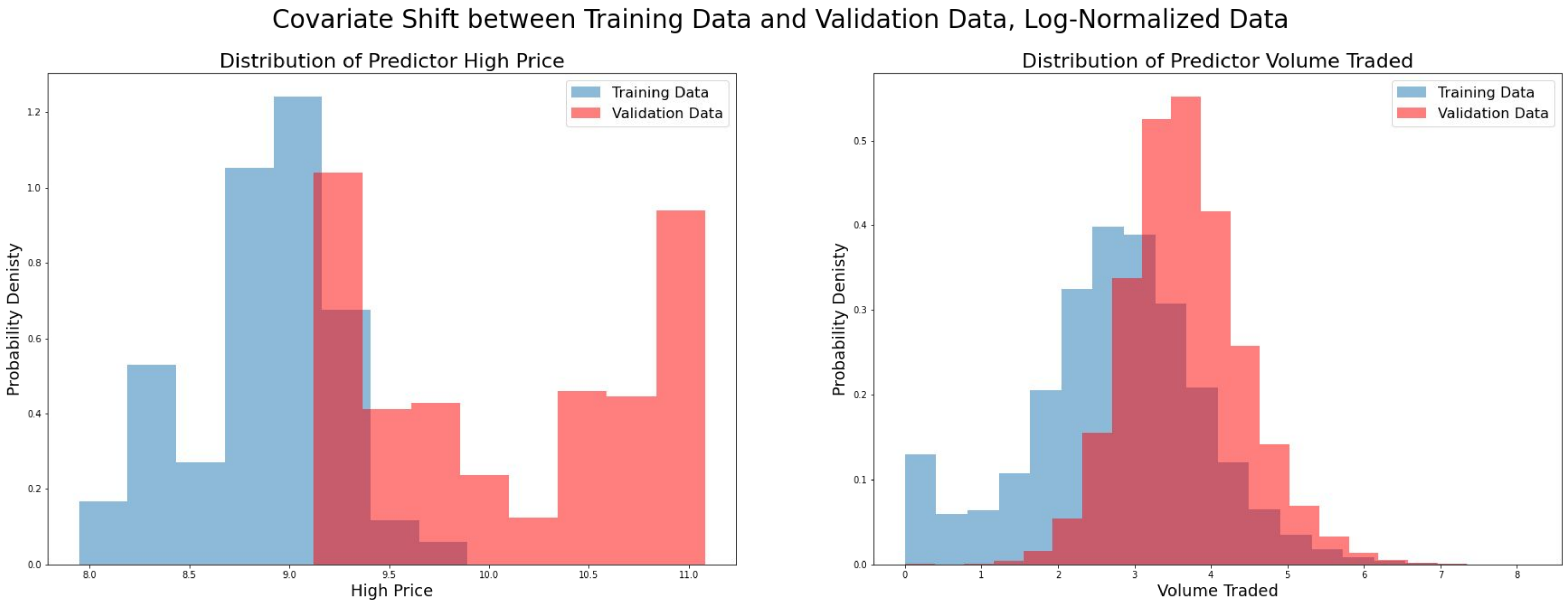
Mean Volume per Day



Key observations:

- The relative value of BTC-USDT increases rapidly due to COVID-19 pandemic
- The trading volume of BTC-USDT is fairly flat, with a few spikes due to COVID-19 pandemic

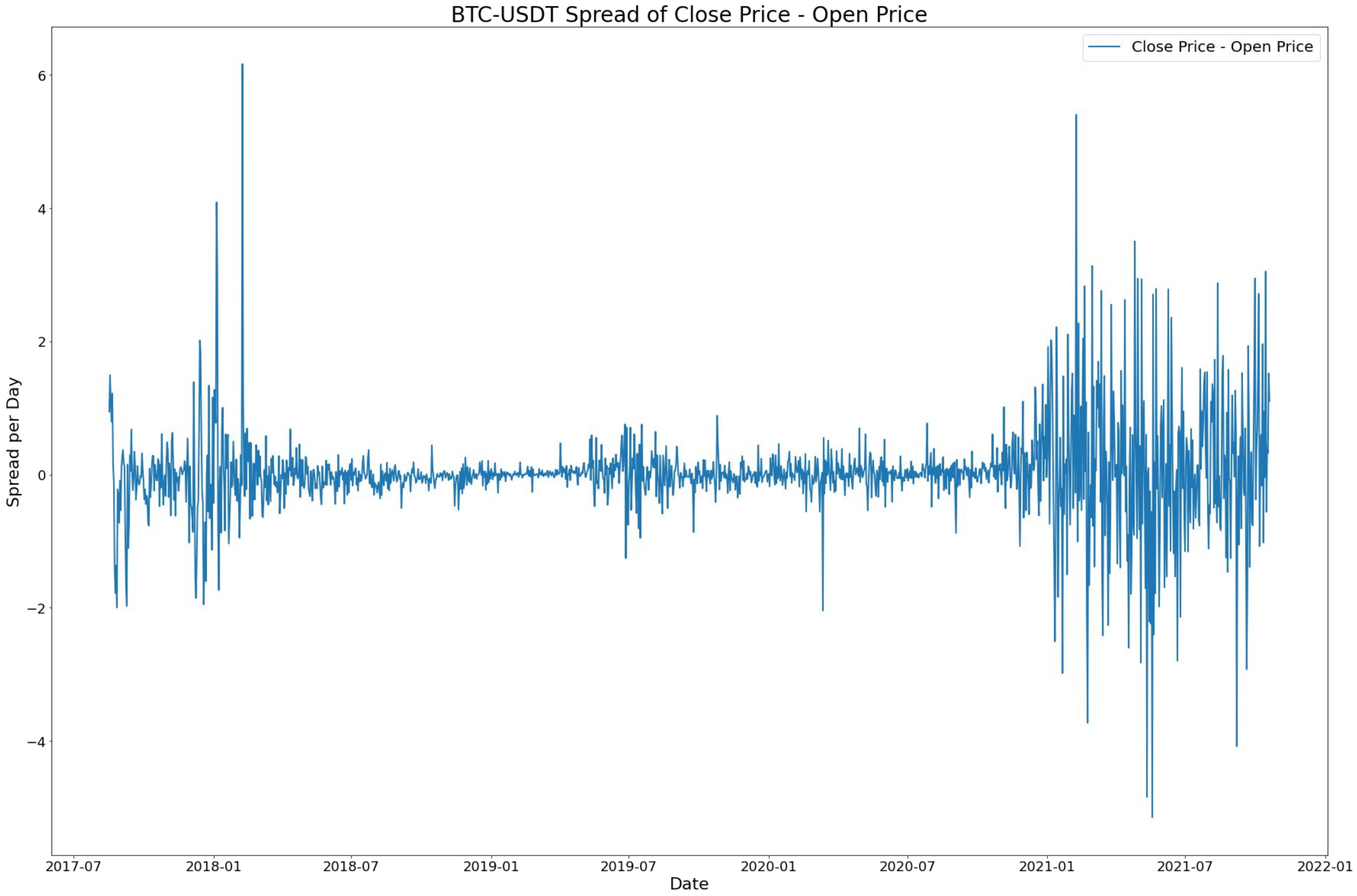
EDA - Out of Distribution Validation Data



Key observations:

- Observable covariate shift in variable `High Price`
- Behavior hinted in the previous slide; validation data post COVID-19

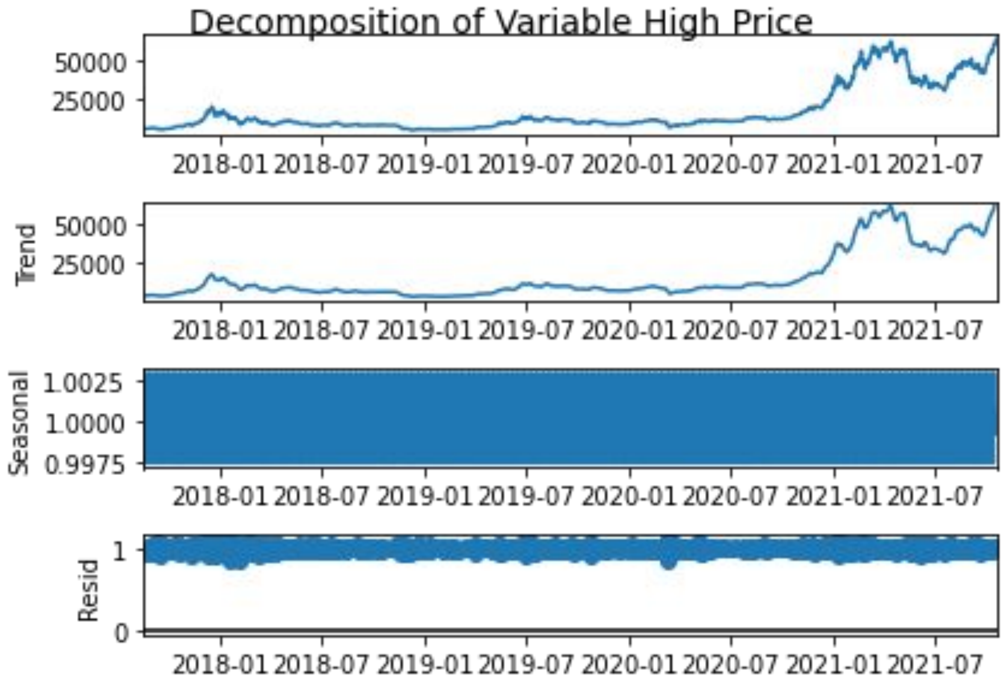
EDA - Response Variable(s)

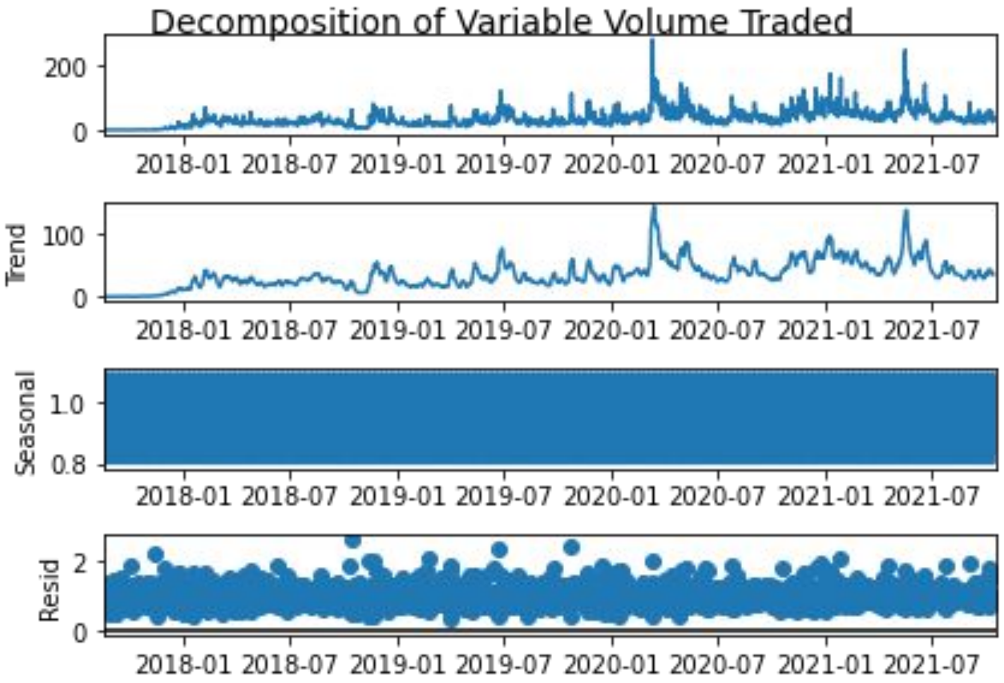


Key observations:

- The spread of `Open Price` - `Close Price` is fairly constant before COVID-19 pandemic
- Since start of COVID-19 pandemic, significantly larger volatility
- This information should be taken into account during modelling/choosing response variable

EDA - Variable Decomposition





Key observations:

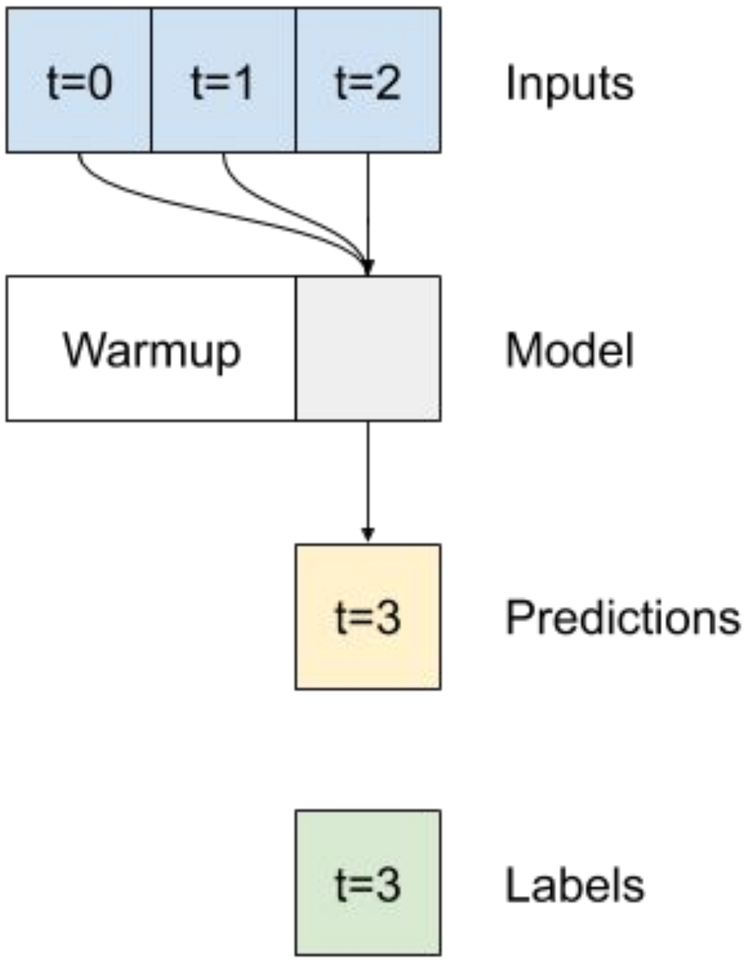
- Visible trend for variable `High Price`, as already hinted in previous EDA slides
- No clear trend for variable `Volume Traded`

Initial Modeling Decisions

- 70-20-10 training-validation-test split
- Standardize the data (based on training set)
- For Tensorflow modeling, remove *Close Time, Open Time, NA*
  - Will perform future feature engineering utilizing *Close Time, Open Time*
- Metrics: Mean Squared Error (MSE), Mean Absolute Error (MAE)
- Prediction on *Close Price*

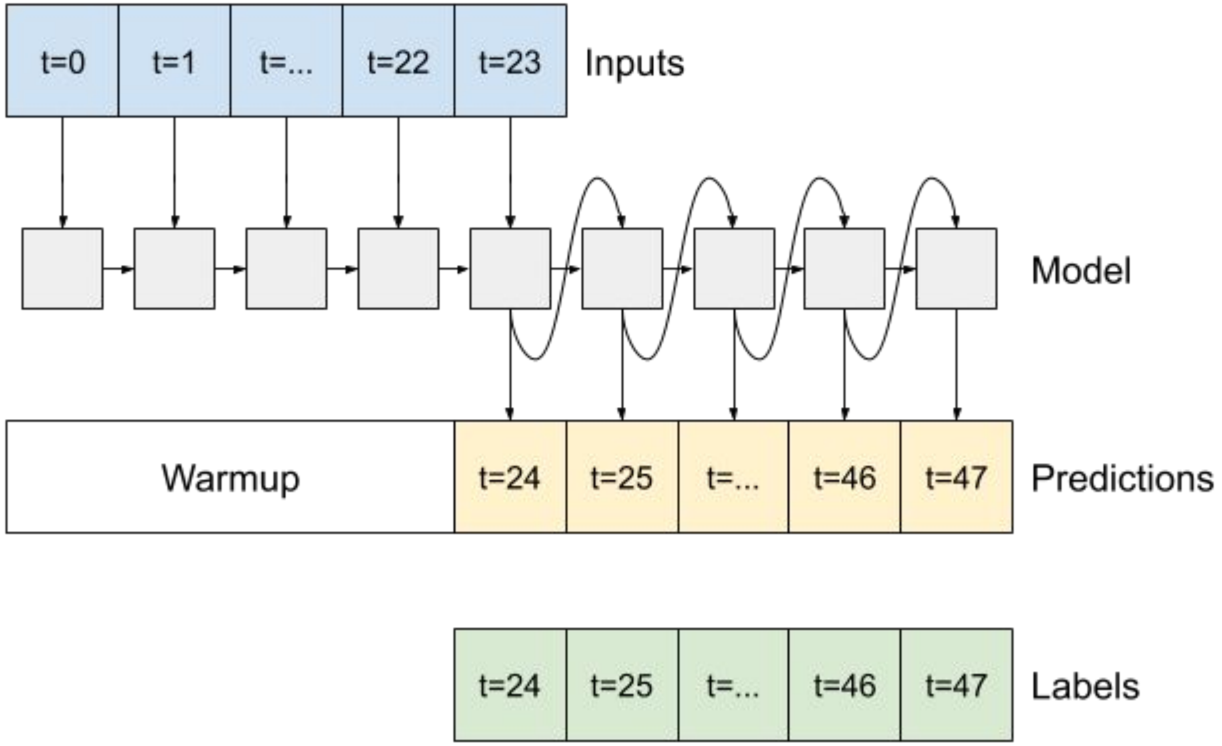
Modeling Approaches

Current: Multivariate, single-step



Future: Autoregressive

each model's output can be fed back into itself at each step and predictions can be made conditioned on the previous one



[Credit]

Models

- **Last Prediction on Close Price (Baseline)**
- **Exponential Smoothing**
- **Linear**
- **FFNN**
- **Multi-step FFNN**
- **CNN**
- **LSTM**

Models - Preliminary Results

Models	Validation MSE	Validation MAE	Test MSE	Test MAE
Baseline	0.00034	0.00993	0.00031	0.01155
Exponential Smoothing	0.00026	0.00832	0.000004	0.00294
Linear	0.00088	0.01878	0.00104	0.02689
FFNN	0.00482	0.05233	0.00631	0.07420
Multi-step FFNN	0.00720	0.06110	0.00960	0.08904
CNN	0.00130	0.02275	0.00147	0.03053
LSTM	66.57594	5.32205	85.41578	8.65671

Next Steps

- Backend completion
- Feature engineering
- Handling out-of-distribution data (covariate shift) from COVID-19
- Hyperparameter tuning to beat Baseline model
- Autoregressive approach
- Front End Building