Price Prediction App

Presented by: Rob Wygant

Overview

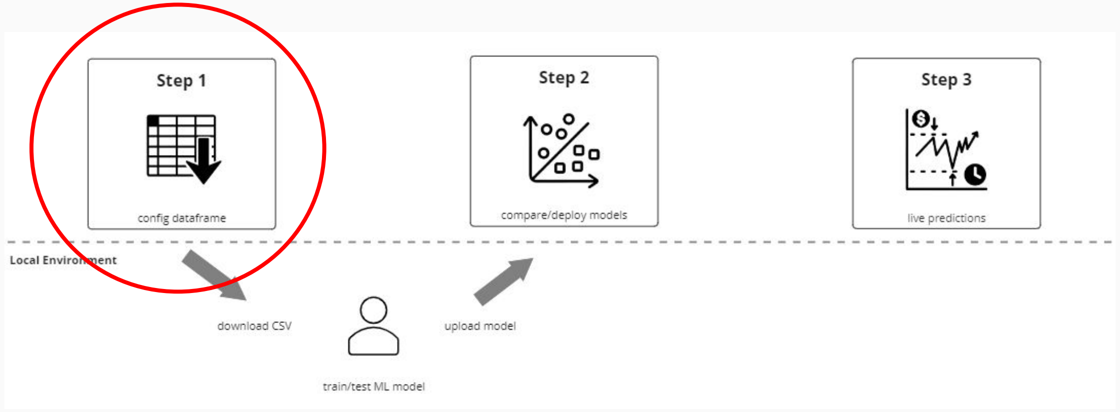
- About
- Workflow/User Interface
- User Interface
- System Design
- Cloud Infrastructure
- Future Work

About Price Prediction App

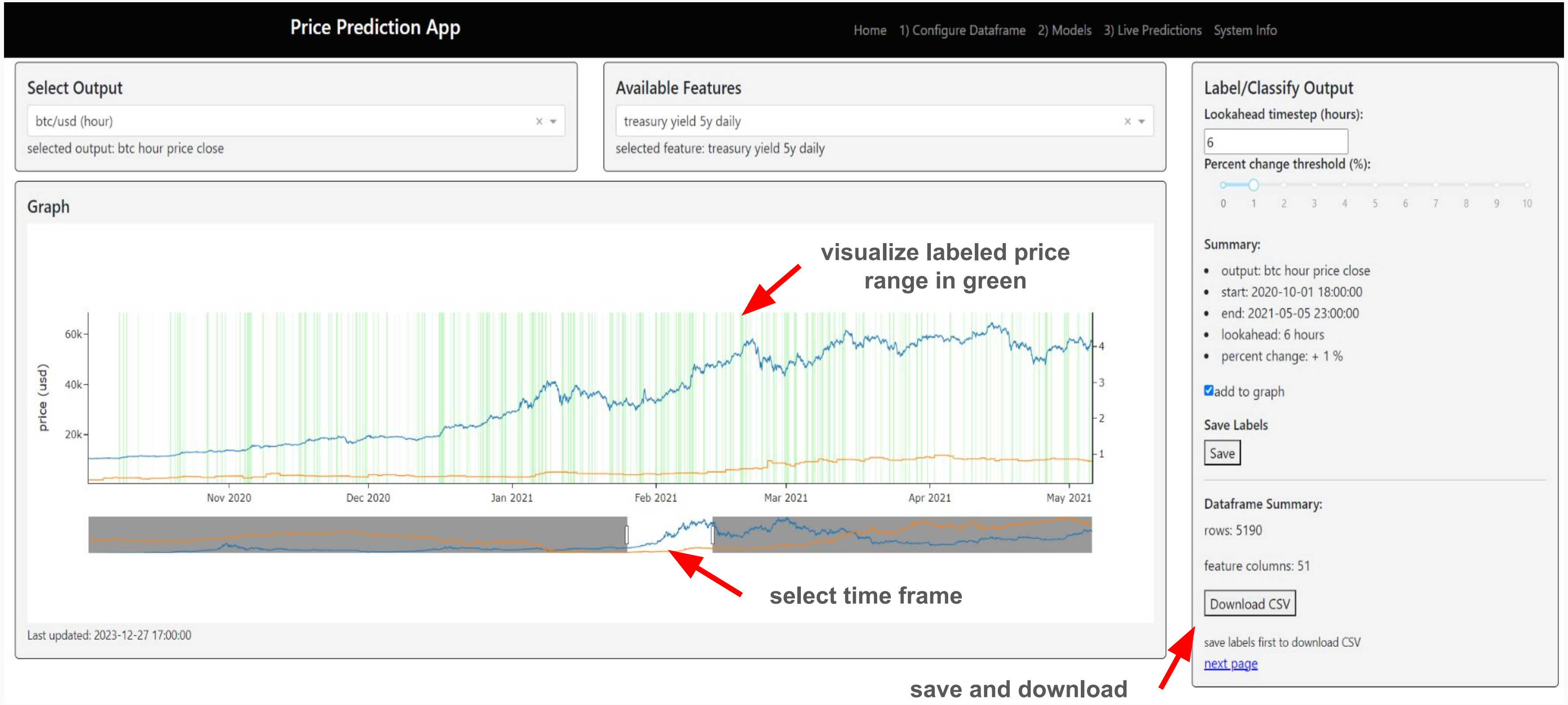
- **What?** Provides basic infrastructure to label/train, compare, and deploy machine learning classifiers on future price movement of financial assets.
- **Why?** Streamline the infrastructure setup required to compare and deploy live models so focus can be put into model development/iteration
- **Who?** Data Scientist/Analytics, Machine Learning Engineers, Quantitative Finance

Workflow

- Step 1: Configure dataframe
  - Select target financial asset price to be classified (currently serving only BTC/USD)
  - Explore over 50 macro economic indicators updated hourly
  - Select time frame of interest
  - Label historic price movement based on a lookahead/percent change threshold
  - Download CSV for local model training



User Interface (Step 1)



Price Prediction App

Home 1) Configure Dataframe 2) Models 3) Live Predictions System Info

Select Output

btc/usd (hour)

selected output: btc hour price close

Available Features

treasury yield 5y daily

selected feature: treasury yield 5y daily

Graph

visualize labeled price range in green

select time frame

Last updated: 2023-12-27 17:00:00

Label/Classify Output

Lookahead timestep (hours):

6

Percent change threshold (%):

Summary:

- output: btc hour price close
- start: 2020-10-01 18:00:00
- end: 2021-05-05 23:00:00
- lookahead: 6 hours
- percent change: + 1 %

☑add to graph

Save Labels

Save

Dataframe Summary:

rows: 5190

feature columns: 51

Download CSV

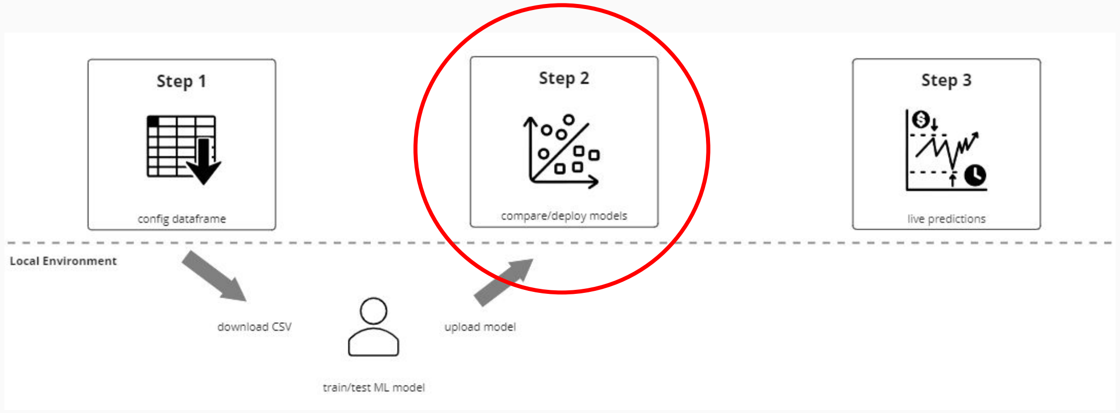
save labels first to download CSV

next page

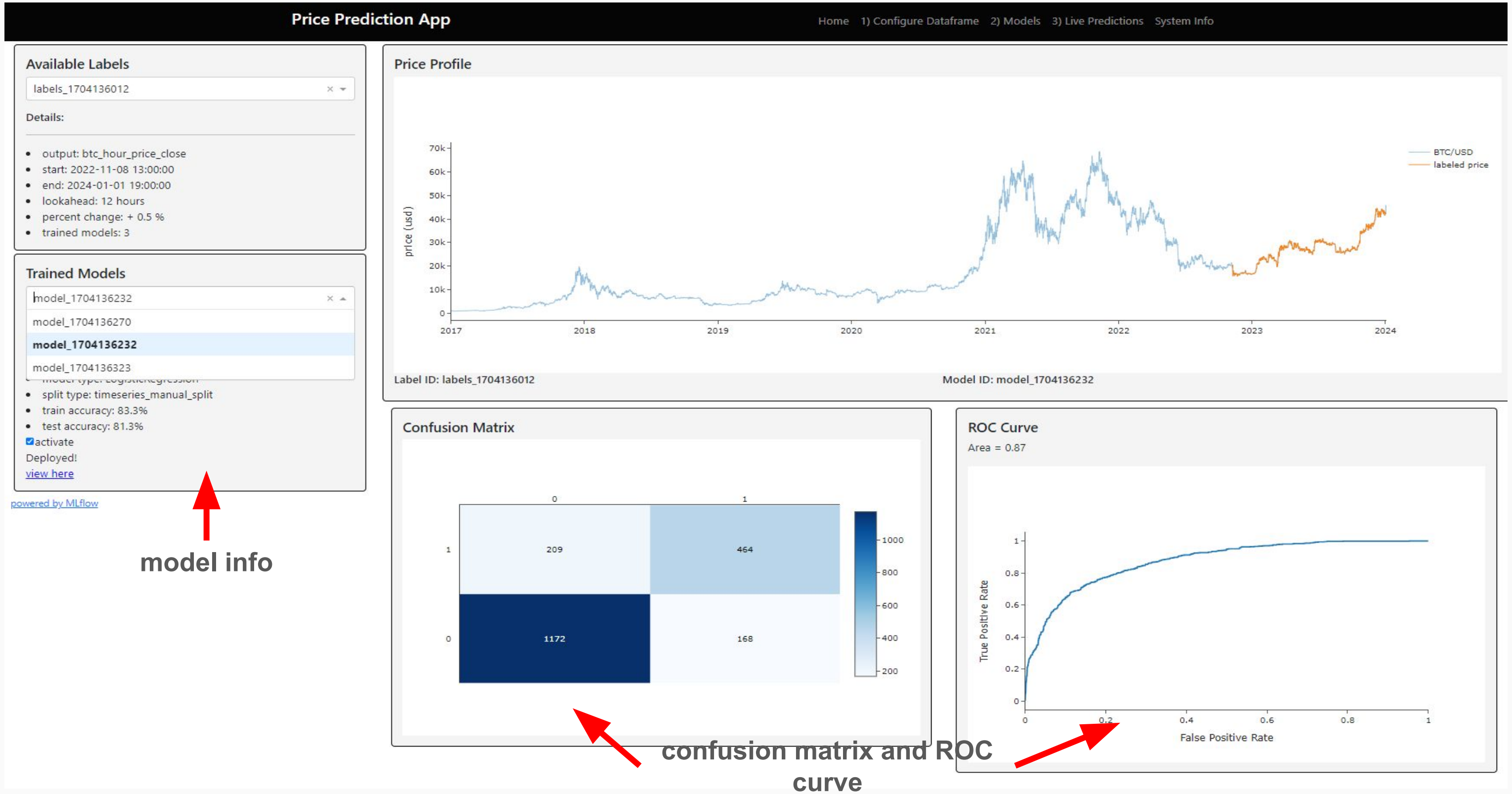
save and download

Workflow

- Step 2: Upload and compare trained models
  - Train and develop model pipeline locally
  - Upload to Price Prediction App database and MLflow instance
  - Compare models grouped by labels
  - Activate specific models for live deployment



User Interface (Step 2)



Details:

- output: btc_hour_price_close
- start: 2022-11-08 13:00:00
- end: 2024-01-01 19:00:00
- lookahead: 12 hours
- percent change: + 0.5 %
- trained models: 3

- model type: LogisticRegression
- split type: timeseries_manual_split
- train accuracy: 83.3%
- test accuracy: 81.3%

☑ activate

Deployed!

Label ID: labels_1704136012

Model ID: model_1704136232

	0	1
1	209	464
0	1172	168

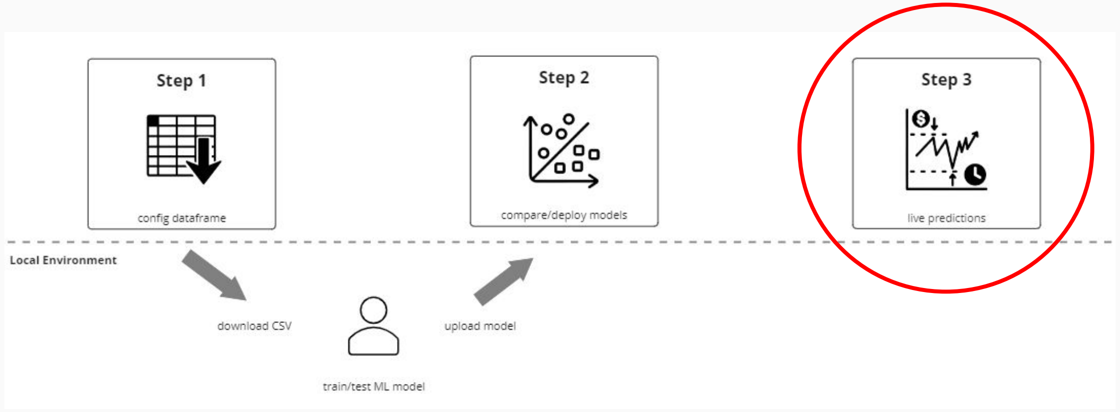
Area = 0.87

model info

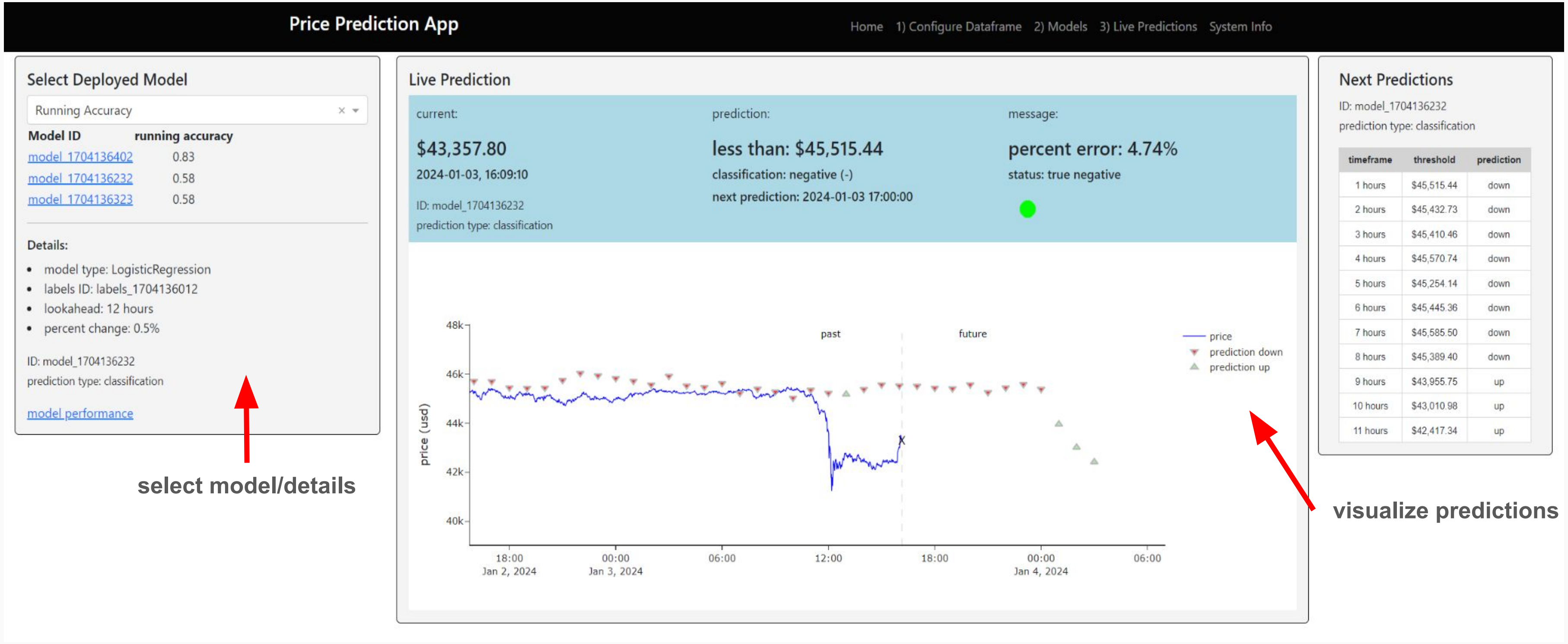
confusion matrix and ROC curve

Workflow

- Step 3: Live predictions
    - Visualize live price and available models
    - Select from deployed models and view predicted hourly price movement
    - Observe model performance based on the live prediction results



User Interface (Step 3)



Price Prediction App

Home 1) Configure Dataframe 2) Models 3) Live Predictions System Info

Select Deployed Model

Running Accuracy

Model ID	running accuracy
model_1704136402	0.83
model_1704136232	0.58
model_1704136323	0.58

Details:

- model type: LogisticRegression
- labels ID: labels_1704136012
- lookahead: 12 hours
- percent change: 0.5%

ID: model_1704136232
prediction type: classification

model performance

Live Prediction

current:
$43,357.80
2024-01-03, 16:09:10
ID: model_1704136232
prediction type: classification

prediction:
less than: $45,515.44
classification: negative (-)
next prediction: 2024-01-03 17:00:00

message:
percent error: 4.74%
status: true negative
●

Next Predictions

ID: model_1704136232
prediction type: classification

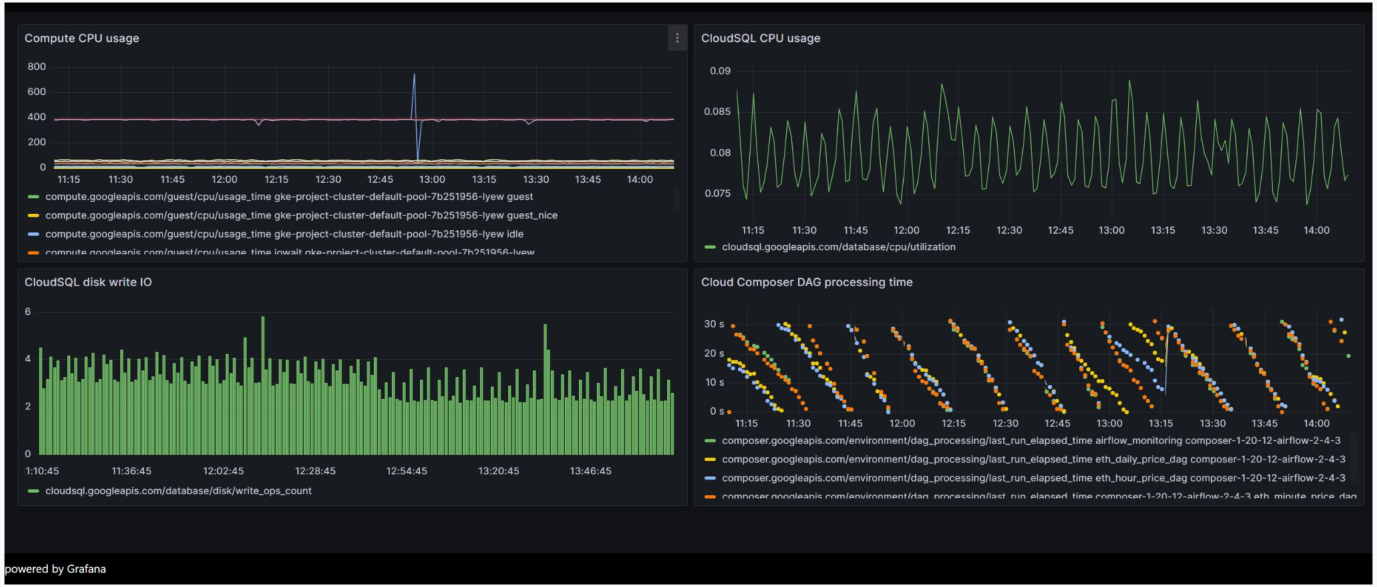
timeframe	threshold	prediction
1 hours	$45,515.44	down
2 hours	$45,432.73	down
3 hours	$45,410.46	down
4 hours	$45,570.74	down
5 hours	$45,254.14	down
6 hours	$45,445.36	down
7 hours	$45,585.50	down
8 hours	$45,389.40	down
9 hours	$43,955.75	up
10 hours	$43,010.98	up
11 hours	$42,417.34	up

select model/details

visualize predictions

Workflow

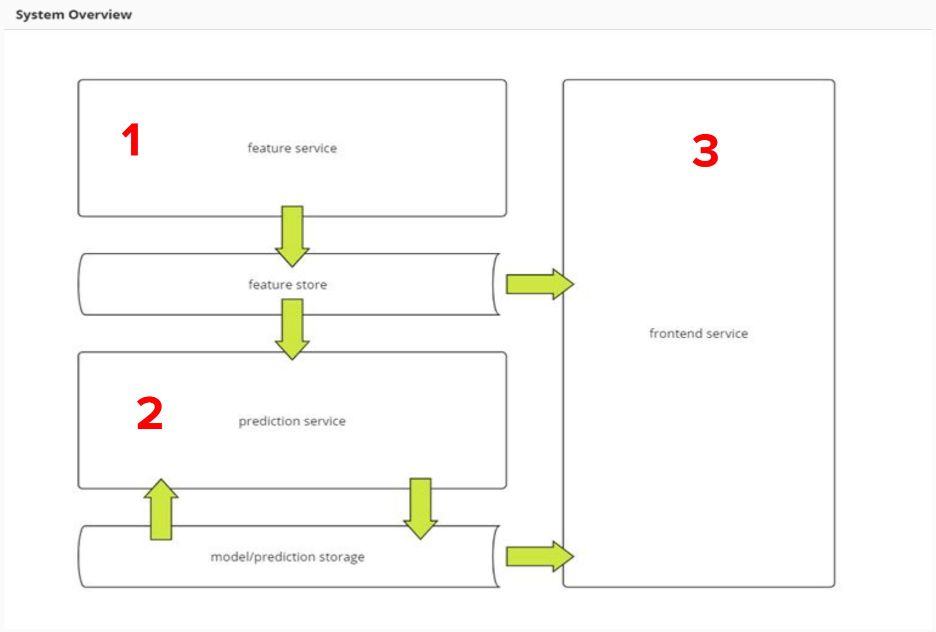
- System Info
  - Observibility for cloud infrastructure
  - Powered by Grafana and Google Cloud APIs



Compute CPU usage
800
600
400
200
0
11:15 11:30 11:45 12:00 12:15 12:30 12:45 13:00 13:15 13:30 13:45 14:00
compute.googleapis.com/guest/cpu/usage_time gke-project-cluster-default-pool-7b251956-lyew guest
compute.googleapis.com/guest/cpu/usage_time gke-project-cluster-default-pool-7b251956-lyew guest_nice
compute.googleapis.com/guest/cpu/usage_time gke-project-cluster-default-pool-7b251956-lyew idle
CloudSQL CPU usage
0.09
0.085
0.08
0.075
11:15 11:30 11:45 12:00 12:15 12:30 12:45 13:00 13:15 13:30 13:45 14:00
cloudsql.googleapis.com/database/cpu/utilization
CloudSQL disk write IO
6
4
2
0
1:10:45 11:36:45 12:02:45 12:28:45 12:54:45 13:20:45 13:46:45
cloudsql.googleapis.com/database/disk/write_ops_count
Cloud Composer DAG processing time
30 s
20 s
10 s
0 s
11:15 11:30 11:45 12:00 12:15 12:30 12:45 13:00 13:15 13:30 13:45 14:00
composer.googleapis.com/environment/dag_processing/last_run_elapsed_time airflow_monitoring composer-1-20-12-airflow-2-4-3
composer.googleapis.com/environment/dag_processing/last_run_elapsed_time eth_daily_price_dag composer-1-20-12-airflow-2-4-3
composer.googleapis.com/environment/dag_processing/last_run_elapsed_time eth_hour_price_dag composer-1-20-12-airflow-2-4-3
powered by Grafana

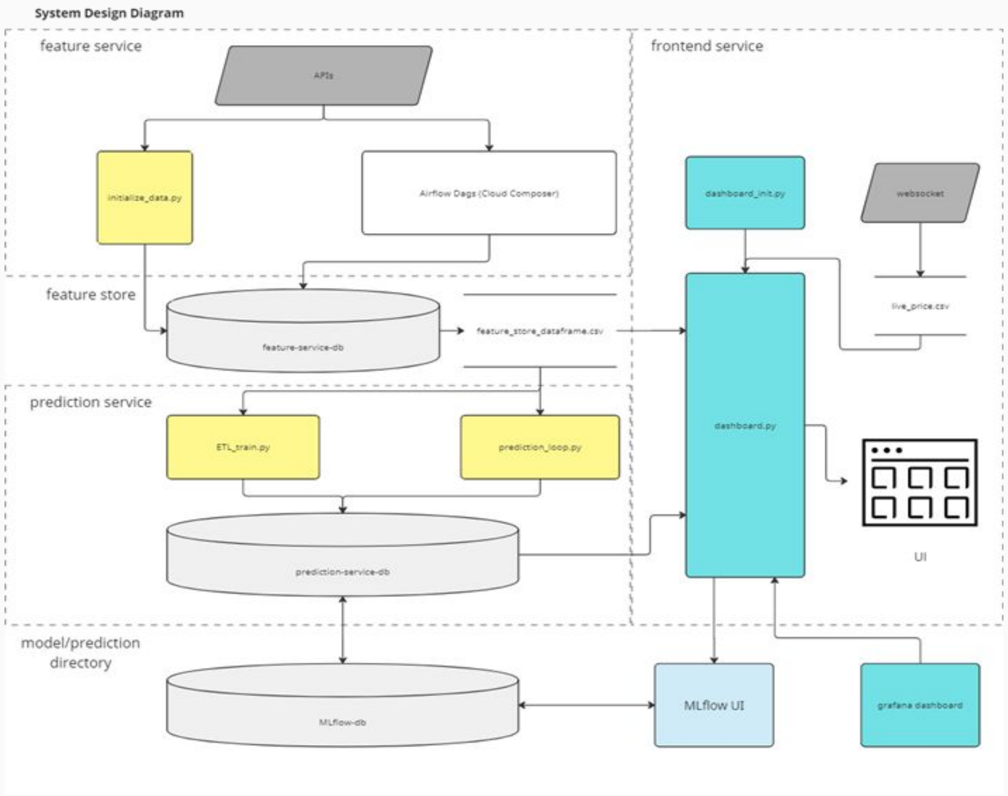
System Design

- Containerized, independent services architecture
- Two databases serving the feature store and prediction service



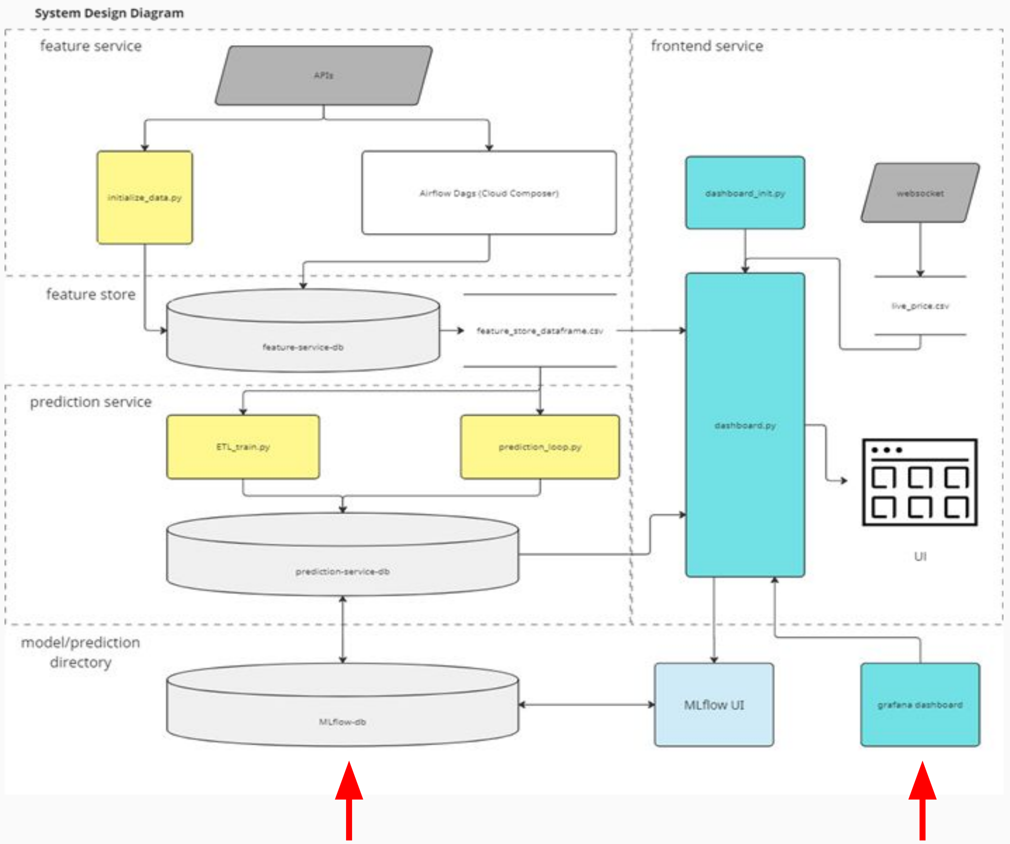
System Design

- Feature Service
  - Routes data from various external price data APIs to feature service database
  - Apache Airflow/scheduling loop function
- Prediction Service
  - Entry point for saving labels and trained model
  - Scheduling loop to pull active model objects and make predictions (hourly)
- Frontend Service
  - User Interface
  - Plotly Dash



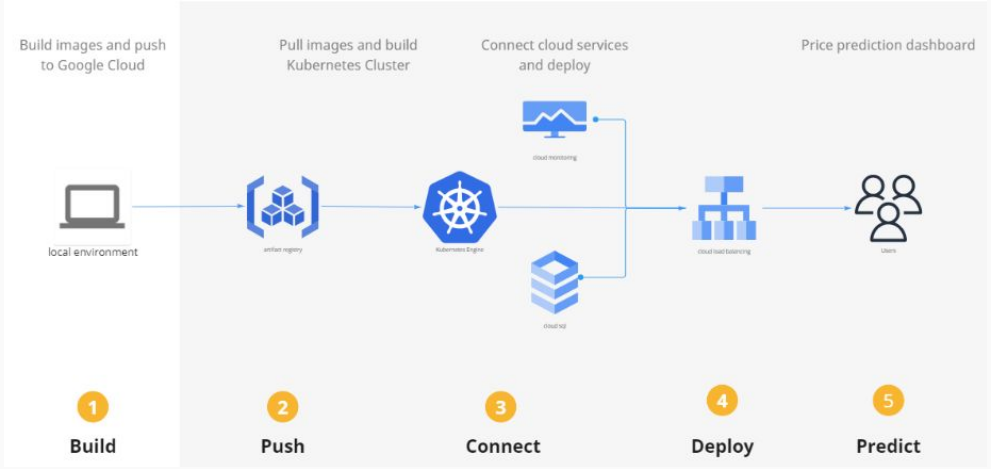
System Design

- MLFlow
  - Stores model directory info for trained models
  - Runs in parallel with prediction service database
  - Accessible via API
- Grafana
  - Embedded dashboard for cloud infrastructure observability
  - Connects with Google Cloud API



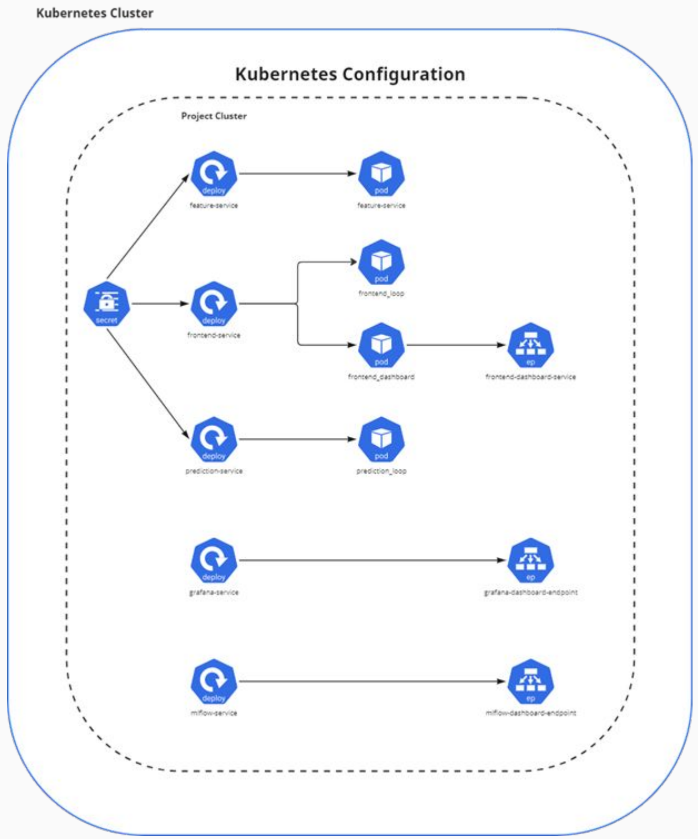
Cloud Infrastructure

- Deployed with Google Cloud Computing
  - CloudSQL (PostgreSQL instance)
  - Container Registry
  - Kubernetes Engine
  - Cloud Storage
- Deployment flow


Build images and push to Google Cloud
Pull images and build Kubernetes Cluster
Connect cloud services and deploy
Price prediction dashboard
local environment
artifact registry
Kubernetes Engine
cloud monitoring
cloud sql
cloud load balancing
Users
1
Build
2
Push
3
Connect
4
Deploy
5
Predict

Cloud Infrastructure

- Kubernetes Cluster
    - Database and API secrets
    - Single namespace for the three services
    - Separate MLflow namespace for Helm deployment



Future work

- Add multiple target asset pairs (other than just BTC/USD)
- Integrate Kubeflow for model pipeline deployment
- Implement unit testing
- Continue to refactor and simplify code