# Bitcoin / Oil Price Correlation and Prediction
# An End to End Data Pipeline

Victor Constantinescu, Cristian Cordoș, Matei Neaga, Matei Popescu
National University of Science and Technology POLITEHNICA Bucharest, Romania
vconstantinescu2710@stud.acs.upb.ro, ioan.cordos@stud.acs.upb.ro
matei.neaga@stud.mec.upb.ro, matei_calin.popescu@stud.acs.upb.ro

February 28, 2025

## Abstract

The purpose of this project is to create a data processing pipeline using a Big Data technology stack. The pipeline will involve data acquisition and the execution of a processing flow: acquiring the data from public sources, processing it, storing it, and visualizing it. For this project we have built an application that computes and displays the correlation and forecast of future correlations between Bitcoin and oil prices.

## 1 Introduction

There are two main objectives aimed at by this project:

- To explore the relationship between oil and bitcoin prices and create a prediction model based on this correlation
- To explore the rapid development and deployment of microservice based web application

Previous work in exploring the relationship between oil and bitcoin prices ([6][7]) draws conclusions as the following:

- Higher oil prices tend to raise the cost of producing Bitcoins
- Higher volatility of Bitcoin returns is associated with higher oil prices
- Oil price improves the in-sample and out-of-sample forecasts of Bitcoin prices
- Higher utility gains can be derived in Bitcoins when oil price is observed
- There is a positive correlation between oil and cryptocurrencies during normal and bullish conditions
- Rising fluctuations in oil demand shocks brings significant movement in cryptocurrencies
- Oil demand shocks and cryptocurrencies returns move in same directions

The appearance of microservice-based web applications has transformed the landscape of modern software engineering. Unlike traditional monolithic architectures, microservices decompose applications into smaller, independently deployable services, each addressing a specific business capability. This architectural shift enables enhanced scalability, flexibility, and resilience, meeting the growing demands of distributed systems and cloud-native environments.

Also this architectural shift enables application developers to massively reuse standard functional blocks like databases, authentication systems, data processing pipelines, etc

## 2 Individual Opinions

### 2.1 LTSM Network Design - Victor Constantinescu

Developing a Python script that leverages TensorFlow's LSTM networks to generate predictions while interacting with a PostgreSQL database showcases the integration of advanced machine learning techniques with robust data management practices. This implementation effectively combines the predictive power of deep learning with the reliability of SQL-based data handling, providing a comprehensive solution for forecasting and storing results.

The script employs TensorFlow's LSTM models for time-series predictions of Bitcoin and oil prices. It starts by retrieving historical data from the PostgreSQL database using a SQL query. After generating predictions, the results are inserted into a dedicated table in the database through another SQL query. This creates a seamless workflow for data retrieval, model inference, and result storage.

I developed three scripts: the first for data preprocessing, which involved removing null values and applying a min-max scaler to the data; the second for training the LSTM model with the following hyperparameters: a sequence length of 5, a dropout rate of 0.2 for both LSTM layers, the ReLU activation function, and the Adam optimizer with a learning rate of 0.001. The model was trained over 100 epochs, with the data split into 80% for training and 20% for testing. The final script was for inference, where the last five values from the database for both Bitcoin and oil prices were used to predict the next day's prices.

### 2.2 Microservice architecture - Cristian Cordoș

Developing and deploying an application on an Azure Virtual Machine (VM) with Dockerized microservices, such as Grafana and PostgreSQL, alongside a Python backend running directly on the host machine, demonstrates a balanced and efficient approach from a DevOps perspective. The combination of Azure's infrastructure and Docker's containerization capabilities, complemented by the simplicity of hosting the Python backend natively, creates a flexible and streamlined deployment process.

Azure VMs provide a robust and customizable environment for deploying applications. With pre-configured Linux-based images, setting up the VM to support both Docker and a native Python runtime is straightforward. Azure's support for various programming environments and its scalability options make it a reliable choice for hosting diverse workloads.

Docker plays a pivotal role in isolating Grafana and PostgreSQL. Containerizing these services ensures they operate independently, avoiding dependency conflicts with the Python backend running on the host. This modular architecture makes it easy to deploy, maintain, and update the containerized components.

The Python backend running directly on the Azure VM allows for lightweight execution without the overhead of containerization. This approach benefits from direct access to system resources and integrates seamlessly with the underlying crontab for periodic execution. Managing the Python environment on the host system is simplified through tools like virtual environments and package managers, ensuring dependencies are well-organized and conflicts are minimized.

From a DevOps perspective, this hybrid architecture leverages the strengths of both containerized and native deployment models. Docker ensures portability and consistency for Grafana and PostgreSQL, while the direct execution of the Python backend on the host simplifies the overall setup and reduces potential runtime complexities associated with container orchestration. This approach is particularly useful when the backend's requirements are minimal or straightforward.

### 2.3 Data visualization - Matei Neaga

From the perspective of creating the final charts, the data was extracted following the ingestion into PostgresDB of the data describing the price fluctuations of Bitcoin and crude oil over the past 15

years. The charts were created using the Grafana framework, a container that was deployed on local infrastructure and connected to the Postgres service. The link between these was established through the connection of the two containers to a local network.

The chart representing the price of Bitcoin highlights a general trend of significant growth starting in 2020, peaking in 2021. After this peak, the price dropped sharply, marking a strong correction, followed by a period of stabilization and moderate fluctuations around the $20,000 mark until mid-2023. Subsequently, Bitcoin experienced steady growth, suggesting a potential recovery or a renewed wave of market interest. As the latest notable news, the cryptocurrency reached a new historic high at the end of 2024, approximately $100,000.

On the other hand, the chart on the right, which illustrates the price of crude oil, shows a somewhat linear trend. In 2020, there was a unique moment when the price dropped dramatically, likely due to the pandemic period. After this period, the price of oil gradually recovered, reaching peaks in 2022, followed by stabilization and slight decreases starting in 2023.

Both charts reflect the dynamics of financial and energy markets, highlighting the impact of global and economic events on prices and emphasizing the utility of Grafana for trend analysis based on data from an SQL database such as Postgres.

### 2.4 Data ingestion - Matei Popescu

In order to correlate bitcoin and oil and forecast their future prices, enough data first had to be extracted, with a way to fetch and update the prices daily.

The initial approach involved exploring the Nasdaq Data Link API as the primary source for both bitcoin and oil prices. The bitcoin data had multiple available metrics, the one chosen being the most relevant (market price). Using the Nasdaq API and other python libraries, it was relatively straightforward to access and download the data.

Getting the oil data proved to be more difficult, as the available data on Nasdaq Data Link was insufficient. The first attempt was with the "JODI" database, which had very diverse data about different types of oil or natural gas, different ways of production, different flows etc., however it didn't work due to the frequency of the data - once per month, which was insufficient, as opposed to the daily updates for bitcoin. The other database option on Nasdaq Data Link was "OPEC", specified as having daily crude oil price updates, however, that data stopped being updated more than a year ago, thus we had to find an alternative.

The database we went with for oil data, outside of what Nasdaq Data Link had to offer, was WTI (Western Texas Intermediate), which had the necessary daily oil prices.

After combining different APIs and scripts, the final script was able to extract both sets of data from the year 2010 up to the present day, in CSV format for both sets (Date/Price). Using PostgreSQL, the data was ingested into the virtual machine's database to be used for forecasting.

The last step of the data ingestion process was to create a script that would update the database each day with the new prices. This was a relatively simple process, as this only required modifying the already described scripts to only fetch data from one day ago (as opposed to data from 2010 up to the present day), and to only add the new prices to their respective tables instead of creating new tables in the database. After testing that it runs without issues, this script was set to run every 24 hours to fetch the new data every day.

In summary, the data ingestion side of the project was relatively simple, straightforward, but an indispensible part of any end-to-end data pipeline, as any such project first starts with the right data.

# 3 Implementation

## 3.1 System Architecture

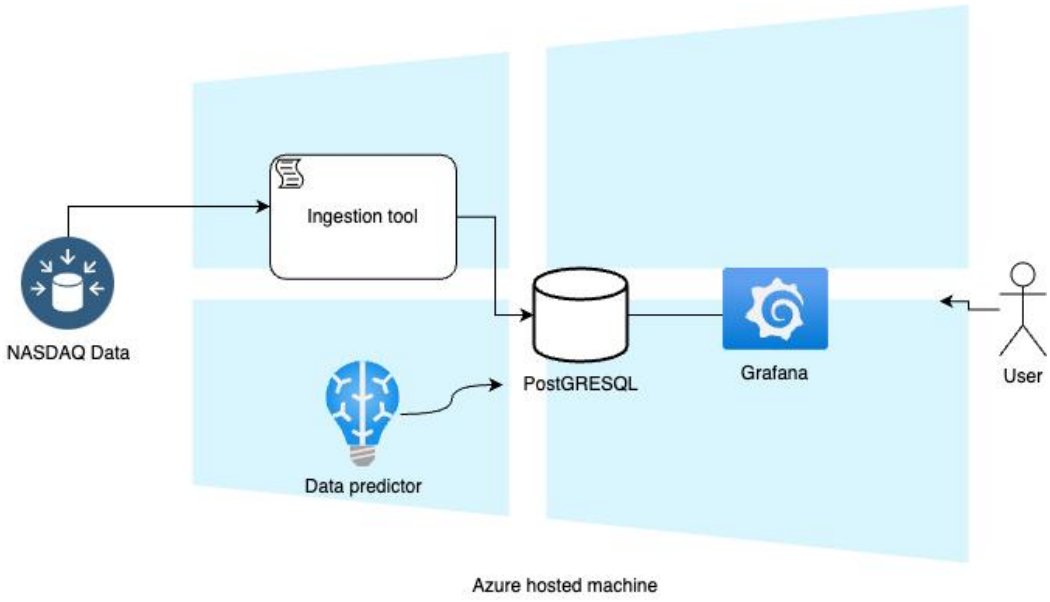


Figure 1: System architecture.

The application is hosted in Microsoft Azure Cloud, on a Standard B4ms machine having 4vCPUs and 16GiB of memory. The chosen operating system is Linux Debian 12. As depicted in Figure 1, there are 4 main components in the system, which run either as docker container or as cron-controller Python scripts:

- **Ingestion tool** is a python script running periodically pulls data from the external sources, does some minimal processing to it and save into the database
- **Data predictor** is also a pyhton script that runs periodically and which runs through it's neural network the newly ingested data. Then it updates the predictions in the database and it retrains the model with the new data.
- **PostGRESql** is hosted in a docker container. The database schema is explained in detail in a further chapter.
- **Grafana** is a graphical data visualization tool which also serves as the application UI. Various dashboards are shown, like oil and Bitcoin prices, correlations between them and future predictions.

Figure 2 shows in detail the workflow of the system.

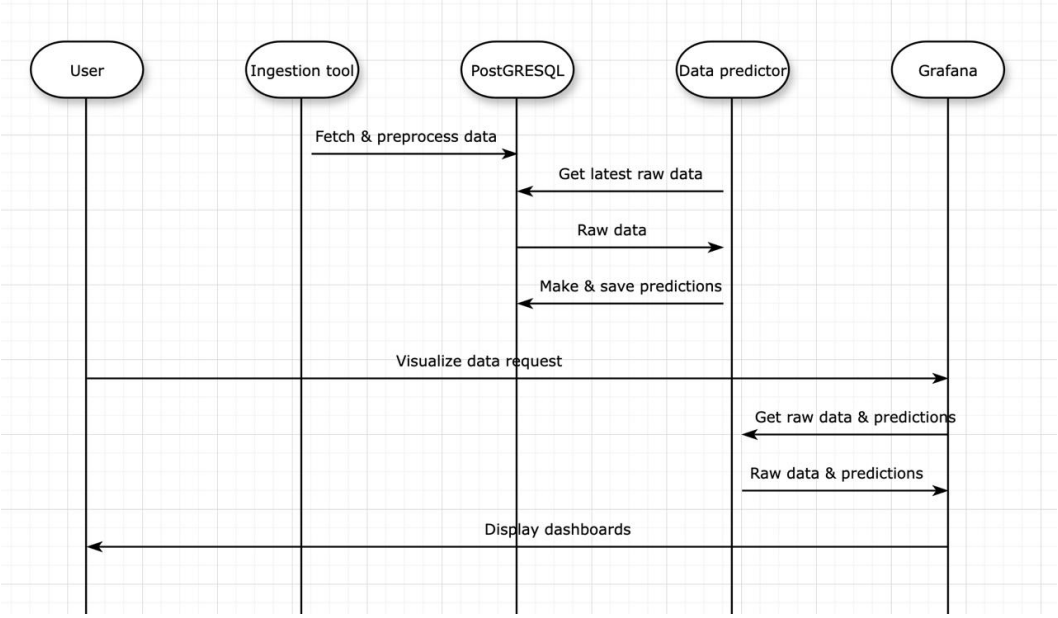


Figure 2: Workflow

### 3.2 Neural Network Design

LSTM (Long Short-Term Memory) networks are a type of recurrent neural network (RNN) designed to process and predict sequences of data. Unlike traditional RNNs, LSTMs are equipped with memory cells that can capture long-term dependencies, making them particularly effective for time-series forecasting. Their structure includes forget, input, and output gates, which enable them to selectively remember or discard information from past time steps. This capability allows LSTMs to handle complex temporal relationships and patterns in sequential data, such as financial time series or stock prices.

In Figure 3, I plotted the training loss versus validation loss for the Bitcoin and oil price models, as well as for a third model used to predict the correlation between these two variables. The most stable model was the one predicting oil prices, while the model predicting Bitcoin prices exhibited the highest level of volatility.

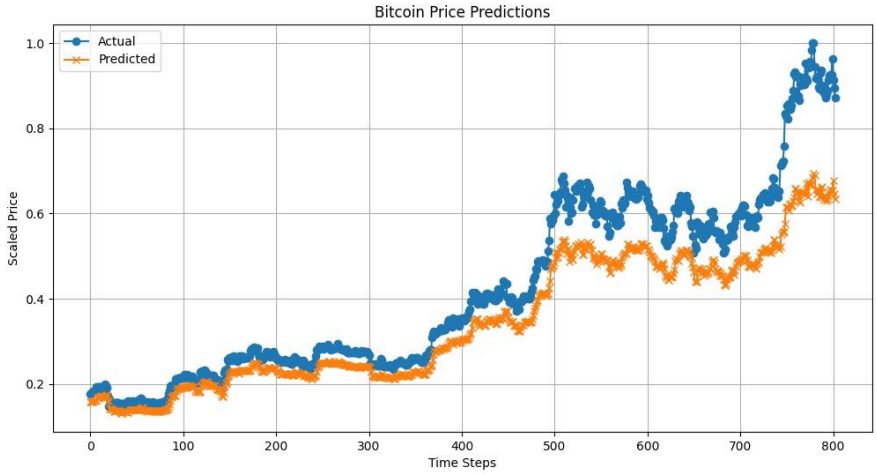


Figure 3: Bitcoin training vs validation loss

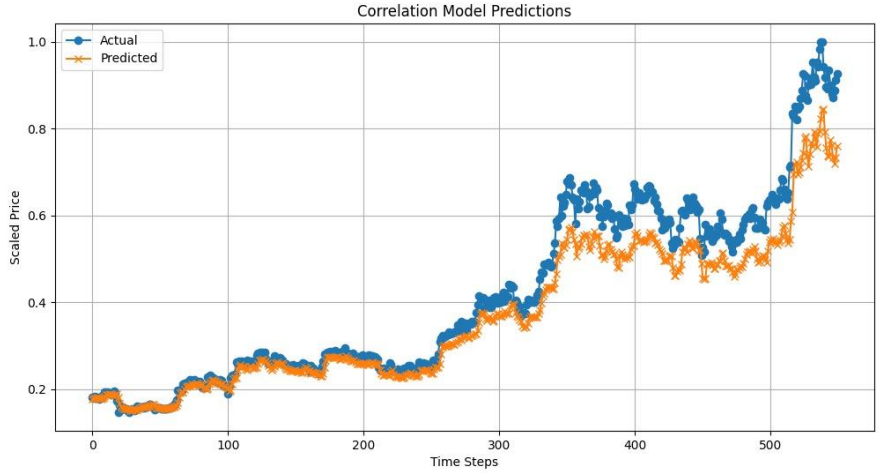


Figure 4: Oil price training vs validation loss

In Figure 4, the predicted oil prices are compared to the actual scaled oil prices. The model performs well, capturing the general trends in the actual data. However, there are slight discrepancies where the model struggles to replicate sharp spikes or dips. These variations suggest the potential benefit of incorporating additional features or extending the sequence length to better capture temporal dependencies. Overall, the model demonstrates a strong correlation between predicted and actual values, making it suitable for practical forecasting tasks.

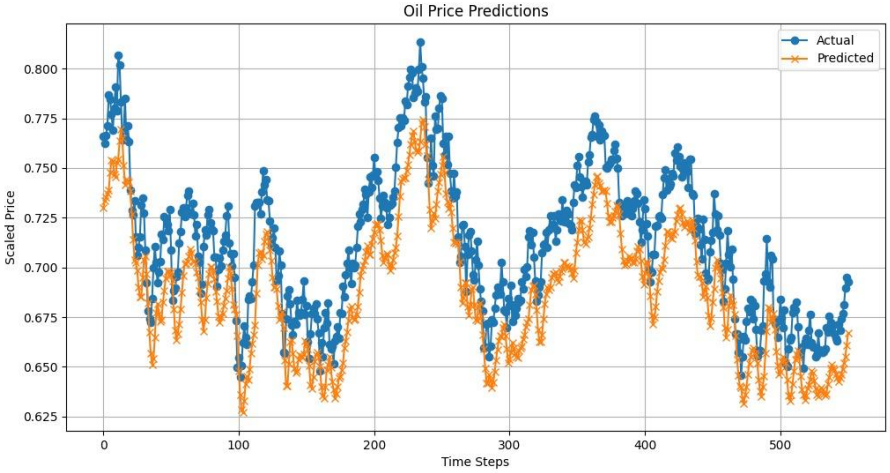


Figure 5: Predicted vs actual oil prices

Figure 5 illustrates predictions from the correlation model, which uses both Bitcoin and oil prices to forecast Bitcoin prices. The model effectively captures overall trends and aligns well with actual values during stable periods. However, it exhibits a noticeable lag during rapid changes. This suggests that while the model effectively highlights interactions between Bitcoin and oil prices, improvements such as additional training data or regularization could enhance its responsiveness.

In the final figure, Figure 6, the Bitcoin price predictions show general alignment with actual values but with greater deviations compared to oil price predictions. This increased volatility may result from the inherent unpredictability of Bitcoin prices or the limited size of the training dataset.

While the model captures broader trends, it occasionally struggles with sharp fluctuations. Future enhancements could involve incorporating more diverse features or fine-tuning hyperparameters to reduce prediction errors.

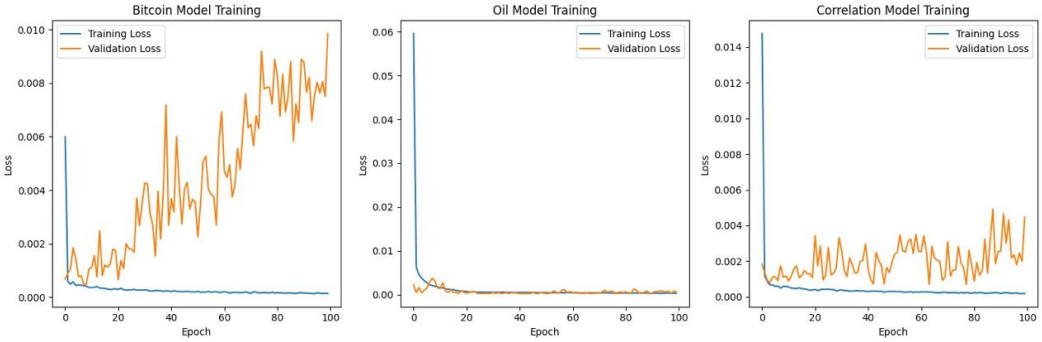


Figure 6: Model training loss

### 3.3 Data Sources and Storage

The following free, public NASDAQ databases were used as data sources:

- http://www.eia.gov/dnav/pet/hist_xls/RWTCd.xls
- https://data.nasdaq.com/databases/BCHAIN

The following database schema was employed to store the raw and processed data:

**Table:** `btc_daily`

| Column | Type | Collation | Nullable | Default |
|---|---|---|---|---|
| id | integer | | not null | nextval('btc_daily_id_seq'::regclass) |
| date | date | | not null | |
| value | double precision | | not null | |

**Table:** `oil_daily`

| Column | Type | Collation | Nullable | Default |
|---|---|---|---|---|
| id | integer | | not null | nextval('oil_daily_id_seq'::regclass) |
| date | date | | not null | |
| value | double precision | | not null | |

**Table:** `predictions`

| Column | Type | Collation | Nullable | Default |
|---|---|---|---|---|
| id | integer | | not null | nextval('predictions_id_seq'::regclass) |
| date | date | | not null | |
| prediction_bitcoin | numeric(20,8) | | not null | |
| prediction_oil | numeric(10,2) | | not null | |
| created_at | timestamp without time zone | | | CURRENT_TIMESTAMP |

## 4 Results

The project focused on a series of automated tasks, referred to as pipelines. It consisted of a script running daily on a virtual machine, aimed at collecting, processing, and storing data using public sources such as NASDAQ, as well as PostgreSQL for data management. From an infrastructure perspective, the setup was implemented using Microsoft Azure as the cloud provider, which proved ideal for containerization with Docker, allowing for scalability and flexibility of the application.

From the architectural standpoint of the LSTM model used for predicting Bitcoin and oil prices, it demonstrated a good ability to capture general trends. The model employs TensorFlow's LSTM models for time-series predictions of Bitcoin and oil prices. It starts by retrieving historical data from the PostgreSQL database using a SQL query. After generating predictions, the results are inserted into a dedicated table in the database through another SQL query.

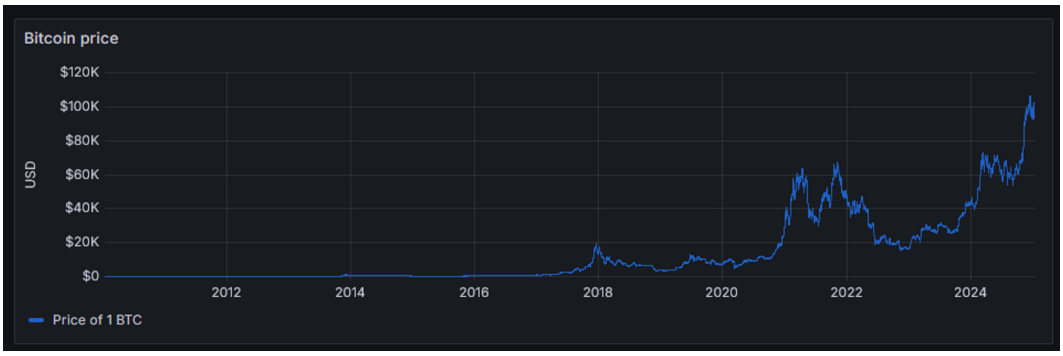


Figure 7: Bitcoin price over the last 15 years

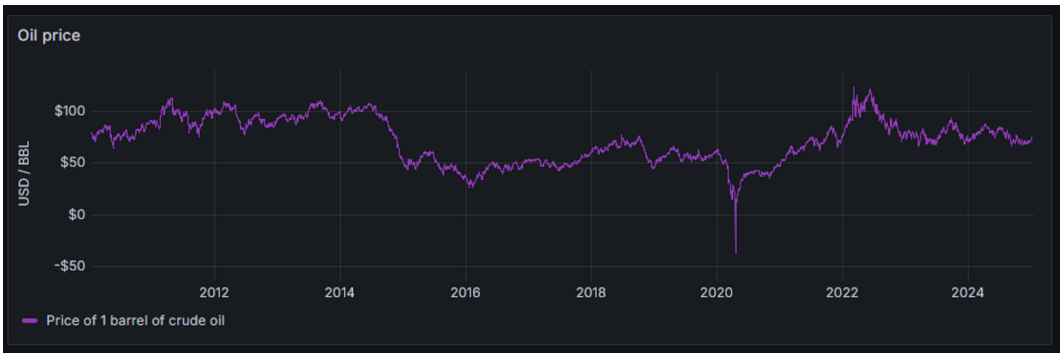


Figure 8: Crude Oil price over the last 15 years

Grafana was used to provide a powerful UI capable of presenting historical trends and predictions, facilitating the interpretation of interactions between financial and energy markets. A significant positive relationship was observed between Bitcoin and oil prices, especially under normal and optimistic market conditions. The initial graphs specifically depict the evolution of Bitcoin (Fig. 7) and crude oil (Fig. 8) prices over the past 15 years.

Subsequently, the graph in Figure 9 shows the LSTM model's prediction for Bitcoin prices over the next five days, while the final graph in Figure 10 similarly presents the prediction for oil prices.

The project showcases the benefits of microservice architecture, such as scalability, modularity, and ease of maintenance. The separation of components into independent services (e.g., visualization, processing, and storage) contributed to a pleasant and manageable learning process.

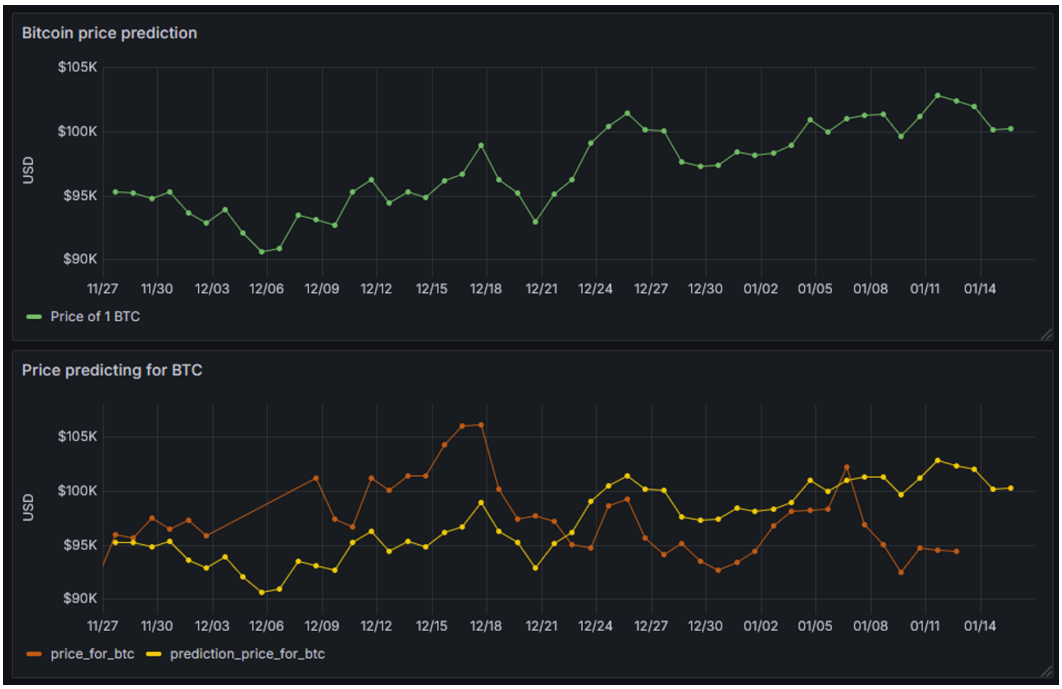


Figure 9: Bitcoin price prediction for the next 5 days

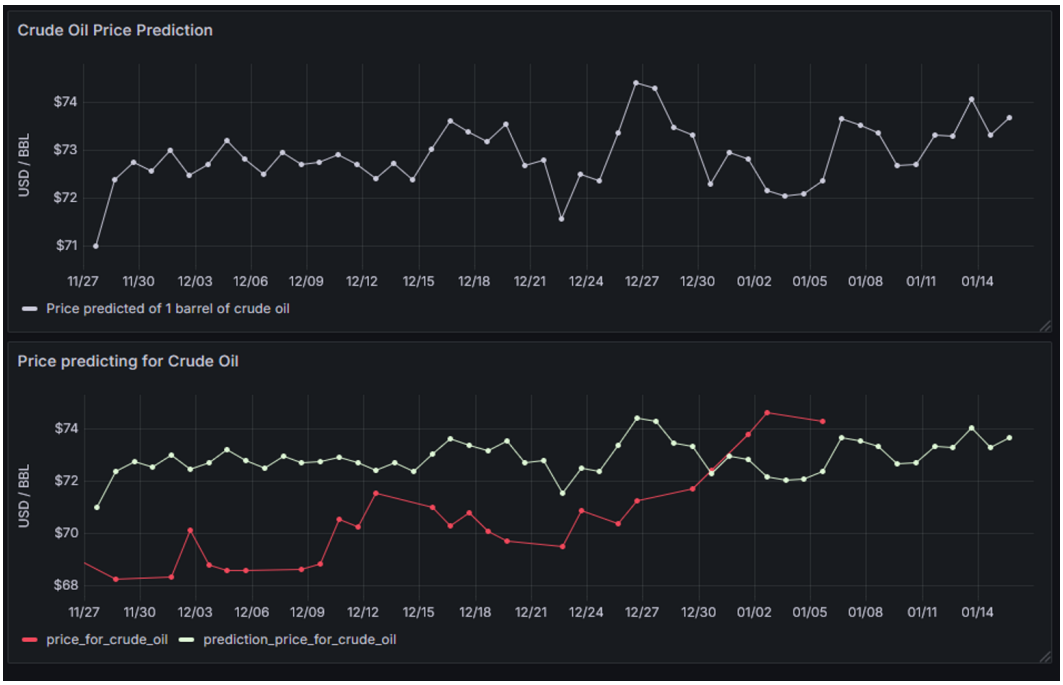


Figure 10: Crude Oil price prediction for the next 5 days

For future improvements, several suggestions could be considered: introducing additional features to better reflect temporal dependencies, implementing the entire pipeline through a specialized framework such as Jenkins or GitHub Actions, or revising the model architecture for more accurate predictions.

## Bibliography

[1] Gabriel Petrea, Roxana-Adriana Puiu, Bogdan-Costel Mocanu, and Omer Mohammed Khodayer Al-Dulaimi (2024) *Determining the Degree of Conviction of Students in University Selection Using the Random Forest Algorithm: An Approach for Adaptive and Personalized Decision Support System in Education*
[2] Juan Carlos Farah, Joana Soares Machado, Pedro Torres da Cunha, Sandy Ingram, and Denis Gillet (2021) *An End-to-End Data Pipeline for Managing Learning Analytics*
[3] Eli Schwartz , Raja Giryes , and Alex M. Bronstein [2018) *DeepISP: Toward Learning an End-to-End Image Processing Pipeline*
[4] Kaichong Zhang (2024) *Incorporating Deep Learning Model Development With an End-to-End Data Pipeline*
[5] Aiswarya Raj, Jan Bosch, Helena Holmstrom Olsson, Tian J. Wang (2020) *Modelling Data Pipelines*
[6] Afees A. Salisu, Umar B. Ndako, Xuan Vinh Vo (2023) *Oil price and the Bitcoin market*
[7] Muhammad Abubakr Naeem, Sitara Karim, Afsheen Abrar, Larisa Yarovaya, Adil Ahmad Shah (2023) *Non-linear relationship between oil and cryptocurrencies: Evidence from returns and shocks*.