Course: Data Exploration
Student Name:

- **Piotr Imiński 247088**
- **Jan Gorzela 247086**

Date: 17.02.2024

---

1. Introduction

In this report, we present the results of our machine learning task performed using Python. The objective of this exercise was to apply machine learning techniques to a given dataset, explore different algorithms, and evaluate their performance.

When selecting a topic for our project, we aimed to focus on a subject that is both relevant and impactful in today's financial landscape. After careful consideration, we chose to predict the prices of cryptocurrencies, we focused mainly on bitcoin, due to the growing interest in digital assets and the complexities involved in forecasting their value. Cryptocurrency markets are highly volatile, influenced by numerous factors such as market sentiment, regulations, and technological advancements. By exploring different prediction models, from machine learning algorithms to traditional financial analysis, we seek to provide insights into the methodologies used to anticipate price movements. This topic is particularly significant for investors, researchers, and financial analysts looking to navigate the dynamic world of cryptocurrencies with data-driven strategies.

2. Dataset overview

Having our topic chosen we proceed with finding appropriate database. During the process we looked into Kaggle datasets [1], [2] as well as api's [3] and Coingecko website[4]. Finally we decided to go with the dataset from Coingecko website. We choose that option because it consists of up to date inputs as well as it was easy to obtain and use.

- **Name of the dataset:** btc-usd-max
- **Source of the dataset:** Coingecko
- **Description of data:** Dataset consists of four features:
    - Date
    - Price
    - Market capitalization
    - Total volume

  Therefore it is perfect to use as timeseries in order to make a prediction for the future price.
- **Data preprocessing:** In order to obtain satisfactionary results we make a several runs of our models on differently preprocessed data. Firstly we changed the Date column into datetime format and then we used it as our index. After that we created a column with logarithmic scale as well as the normalized form of the Price column with the use of min max scaler. In addition we also expressed the data in the monthly basis to better predict the long term dependencies.

3. Tool used

- **Software used:** During this project we used Python with the following libraries:

- NumPy: A fundamental Python library for numerical computing, providing support for large, multi-dimensional arrays and mathematical functions.
- Pandas: Data analysis and manipulation library that provides data structures like DataFrames and Series, making it easy to handle, clean, and analyze structured data.
- Matplotlib: Visualization library that allows users to create plots, helping to visually analyse data and present it effectively.
- Scikit-learn: Machine learning library that provides tools for classification, regression, clustering, dimensionality reduction, and model evaluation, making it essential for building predictive models.
- TensorFlow: A deep learning and machine learning framework developed by Google, offering tools for building and training neural networks.
- Statsmodels: A statistical modelling library that includes SARIMAX (Seasonal AutoRegressive Integrated Moving Average with eXogenous variables), a powerful time series forecasting method used to analyze and predict trends in financial and economic data.

Used versions of aforementioned libraries are presented in Table 1.

Table 1. Versions of used libraries.

Library	version
Python	3.12.7
NumPy	2.0.2
Pandas	2.2.3
Matplotlib	3.10.0
Scikit-learn	1.6.1
TensorFlow	2.18.0
Statsmodels	0.14.4

- **Reason for choosing the tool:** We have chosen Python because it is a more suitable choice over Orange for cryptocurrency price prediction due to its flexibility, scalability, and the vast ecosystem of specialized libraries. While Orange provides a user-friendly, drag-and-drop interface for data analysis and machine learning, it lacks the depth and customization required for complex financial forecasting. Python, on the other hand, offers powerful libraries like NumPy, Pandas, Scikit-learn, TensorFlow, and Statsmodels (SARIMAX), which allow for advanced data preprocessing, feature engineering, deep learning models, and statistical time series forecasting. Additionally, Python supports automated pipelines, integration with real-time data sources, and custom model optimization – essential for adapting to the highly volatile nature of cryptocurrency markets. So it would be easier to develop the project further in the future.

4. Machine Learning task

In our cryptocurrency price prediction project, we employed a regression-based supervised learning approach to forecast future prices based on historical data. Regression was chosen because cryptocurrency prices are continuous numerical values

influenced by various factors. We decided to use two different methods: SARIMAX (Seasonal AutoRegressive Integrated Moving Average with Exogenous variables) for time series forecasting [5] and a GRU (Gated Recurrent Unit) neural network for deep learning-based sequence modelling [6]. The target variable in our models was the future price of a selected cryptocurrency. In order to choose the appropriate features we plotted the correlation matrix (Figure 1). After analysing the correlation matrix, we come to conclusion that all the features are highly correlated therefore we decided to use only price.

	price	market_cap	total_volume
price	1.000000	0.999549	0.700593
market_cap	0.999549	1.000000	0.694193
total_volume	0.700593	0.694193	1.000000

Figure 1. Corelation matrix.

In the SARIMAX model, we expressed prices on a monthly basis to smooth short-term fluctuations and better capture long-term trends. Additionally, we experimented with different starting dates to identify the optimal period for model training, improving forecasting accuracy. To fine-tune the seasonal parameters, we plotted the data and conducted research on cryptocurrency trends, identifying cyclic behaviours that could enhance model performance.
For the GRU-based approach, we transformed the data into sequential inputs, where each input contained a sequence of past 360 price values, and the model was trained to predict the next price. The dataset was first normalized using MinMaxScaler to scale values between 0 and 1, ensuring stability in neural network training.
To obtain the best results, we applied Grid Search to optimize hyperparameters for SARIMAX model, we tuned parameters such as order (p, d, q) and seasonal components, ensuring the model effectively captured cryptocurrency price patterns. For the GRU model, we optimized factors like the number of units, learning rate, batch size, and dropout rate to prevent overfitting. Additionally, we implemented Early Stopping in the GRU model, which monitored validation loss during training and stopped the process when no further improvement was observed, preventing unnecessary computation and mitigating overfitting. The dataset was split into 80% training and 20% testing sets, ensuring a fair evaluation of model performance.

5. Algorithms used

In our project, we selected two machine learning algorithms: SARIMAX (Seasonal AutoRegressive Integrated Moving Average with Exogenous variables) and a GRU (Gated Recurrent Unit) neural network. Each algorithm was chosen to capture different aspects of price movements, combining statistical time series modelling with deep learning for more robust forecasting.

Algorithm 1: SARIMAX

- Description:
  SARIMAX is an extension of the ARIMA (AutoRegressive Integrated Moving Average) model, which is widely used for time series forecasting. It consists of the following key components:
  - Autoregressive component (p): Uses past values of the time series to predict future values. It assumes that past price movements influence future ones.
  - Differencing component (d): Makes the time series stationary by subtracting previous values from the current ones to remove trends.
  - Moving Average component (q): Accounts for past forecasting errors to improve predictions.
  - Seasonality (S): Captures repeating patterns in the data, such as monthly or yearly trends.
  - Seasonal Components (P, D, Q, m) – Captures repeating patterns in the data, where:
    - P: Captures autoregressive patterns in the seasonal component.
    - D: Removes seasonal trends to make the series stationary.
    - Q: Captures seasonal moving average components.
    - m: Defines the length of the seasonal cycle
  - Exogenous Variables (X): Allows the inclusion of external factors that may influence the target variable, though we did not use additional external variables in this case.

  SARIMAX works by identifying the best combination of these parameters (p, d, q, seasonal components) to fit the data. The model learns from historical prices and generates forecasts based on identified trends and seasonal fluctuations.

- **Reason for Choosing:**
  We choose the SARIMAX algorithm because after plotting and analysing our data we discovered that it exhibit seasonal patterns and long-term trends. By expressing prices on a monthly basis and analysing different starting dates, we aimed to identify cyclical behaviours that influence price movements. The model's ability to capture linear dependencies in time series data makes it a strong choice for understanding long-term market trends.

Algorithm 2: GRU (Gated Recurrent Unit) Neural Network

- **Description:**
  GRU is a type of Recurrent Neural Network (RNN) specifically designed to handle sequential data, making it ideal for time series forecasting. Unlike traditional feedforward networks, GRUs maintain a memory of past values to influence future predictions. They achieve this through two key mechanisms:
  - Update Gate: Controls how much past information is retained and how much new information is incorporated.
  - Reset Gate: Determines how much of the past information is forgotten, allowing the model to adapt to new trends.

  Unlike standard RNNs, GRUs solve the vanishing gradient problem, which allows them to effectively capture long-term dependencies without losing information

over time. This makes GRUs particularly suitable for modelling complex nonlinear relationships in financial time series data.

- **Reason for Choosing:** We selected GRU because deep learning models like RNNs and GRUs excel at capturing nonlinear dependencies in time series data, which is crucial given the highly volatile nature of cryptocurrency markets. Unlike traditional time series models that assume linear relationships, GRUs can adapt to sudden market shifts and capture hidden patterns that may not be evident in statistical models.

6. Model training and evaluation

Model 1: SARIMAX

Training Data

The dataset was split into training and testing sets using an 80/20 ratio. The first 80% of the data was used for training the model, while the remaining 20% was reserved for evaluation. This ensures that the model is trained on a majority of the data while maintaining a portion for testing its predictive capabilities.

Model Training

To train the SARIMAX model, we followed these steps:

1. **Data Preprocessing:** Missing values were handled, and time series stationarity was checked using the Augmented Dickey-Fuller (ADF) test.
2. **Feature Selection:** Correlated variables were identified and deleted.
3. **Model Configuration:** The SARIMAX model was initialized with selected hyperparameters (p, d, q) for the autoregressive, differencing, and moving average components, as well as (P, D, Q, m) for seasonal components.
4. **Parameter Tuning:** Hyperparameters were optimized using grid search to find the best-fitting model. Therefore, in the next steps we used the following parameters:
    - (p, d, q) = (1, 1, 1)
    - (P, D, Q, m) = (1, 1, 1, 48)

Evaluation Metrics

To assess the model's performance, we used the Mean Absolute Percentage Error (MAPE), which quantifies the average percentage error between predicted and actual values. A lower MAPE value indicates better forecasting accuracy. However in the case of cryptocurrency markets, where prices are highly volatile and influenced by various external factors, shape analysis of predictions can serve as a valuable indicator. Even if exact price levels deviate, capturing the overall trend and key movements—such as uptrends, downtrends, and reversal points—can provide meaningful insights for traders and investors. Recognizing these patterns can aid in decision-making, as historical trends often repeat themselves in crypto markets. Thus, despite higher error margins, a well-aligned prediction shape can still offer useful strategic guidance in financial planning.

Results:

- **Prediction Results for BTC Price (2024-2025) (Figure 2)**

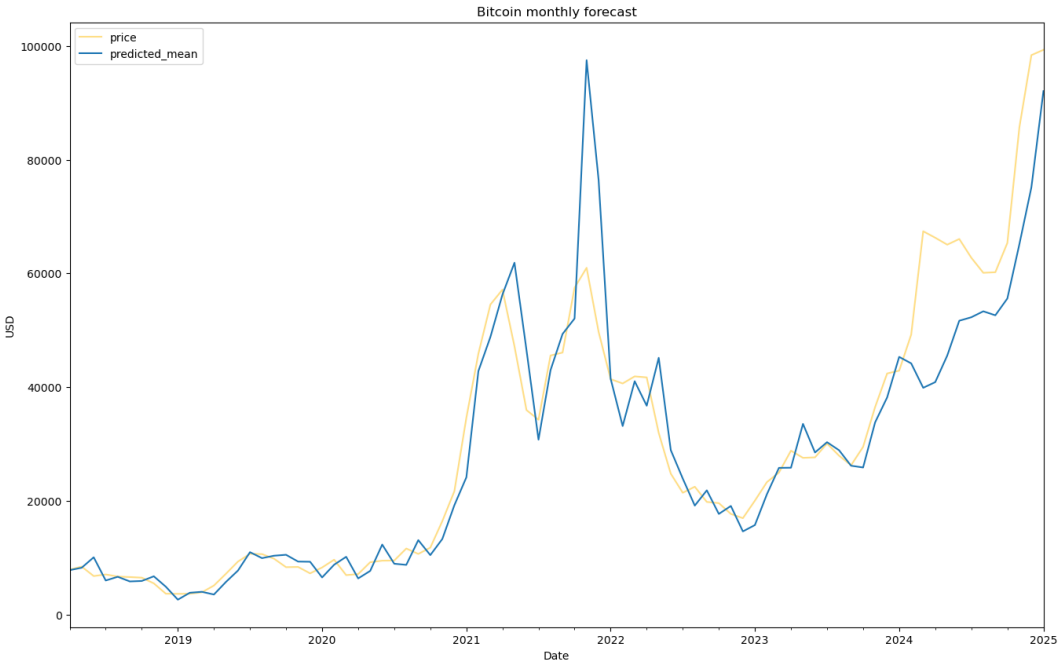

Figure 2. Prediction Results for BTC Price (2024-2025).

The plot illustrates the predicted Bitcoin price from 2024 to 2025, compared to the actual values. The forecasted trend closely follows the real price movements, capturing the overall shape and fluctuations effectively. Despite a Mean Absolute Percentage Error (MAPE) of 19.78%, the model demonstrates a strong predictive ability, suggesting that SARIMAX effectively captures the underlying patterns in the BTC price data. Some deviations exist, but the general trend alignment indicates a reasonably accurate forecast.

- **Extended Prediction Results for BTC Price (2023-2025) (Figure 3)**

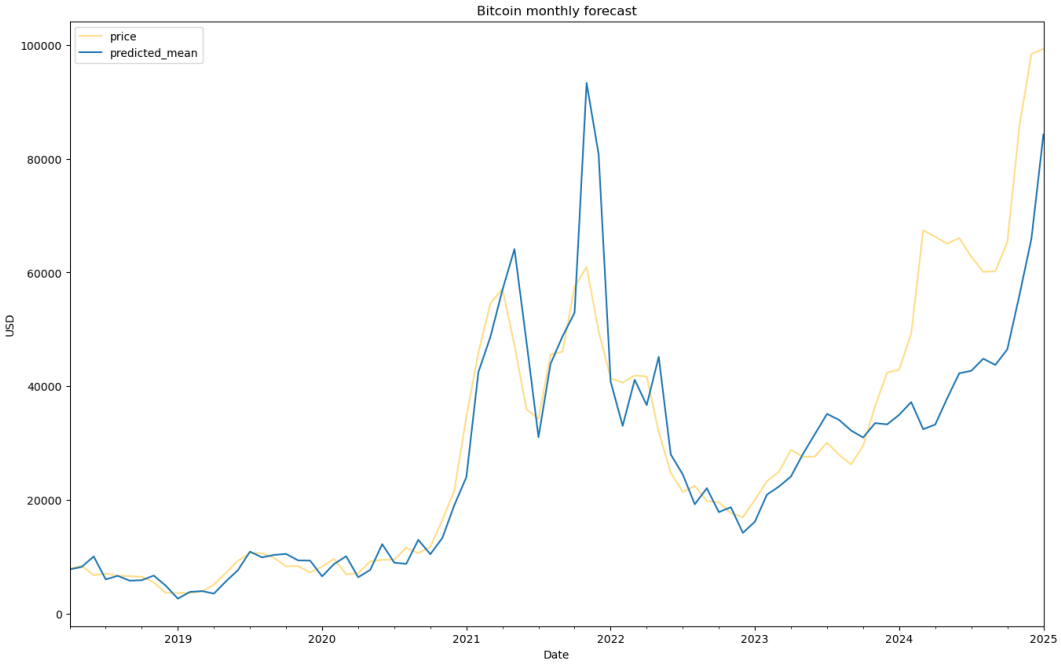

Figure 3. Extended Prediction Results for BTC Price (2023-2025).

When extending the prediction period to 2023-2025, the model continues to capture the general trend of BTC price movements. However, the accuracy slightly decreases, as indicated by the increase in Mean Absolute Percentage Error (MAPE) to 23.48%. While the forecast still follows the overall shape of the actual values, larger deviations occur, suggesting that the model struggles more with longer-term predictions. This could be due to increased volatility or limitations in capturing long-term dependencies. Nonetheless, the results remain reasonably aligned with the observed data, demonstrating the model's ability to provide meaningful forecasts over an extended period.

- **Predicting the Bull Market from 2020 (Figure 4)**

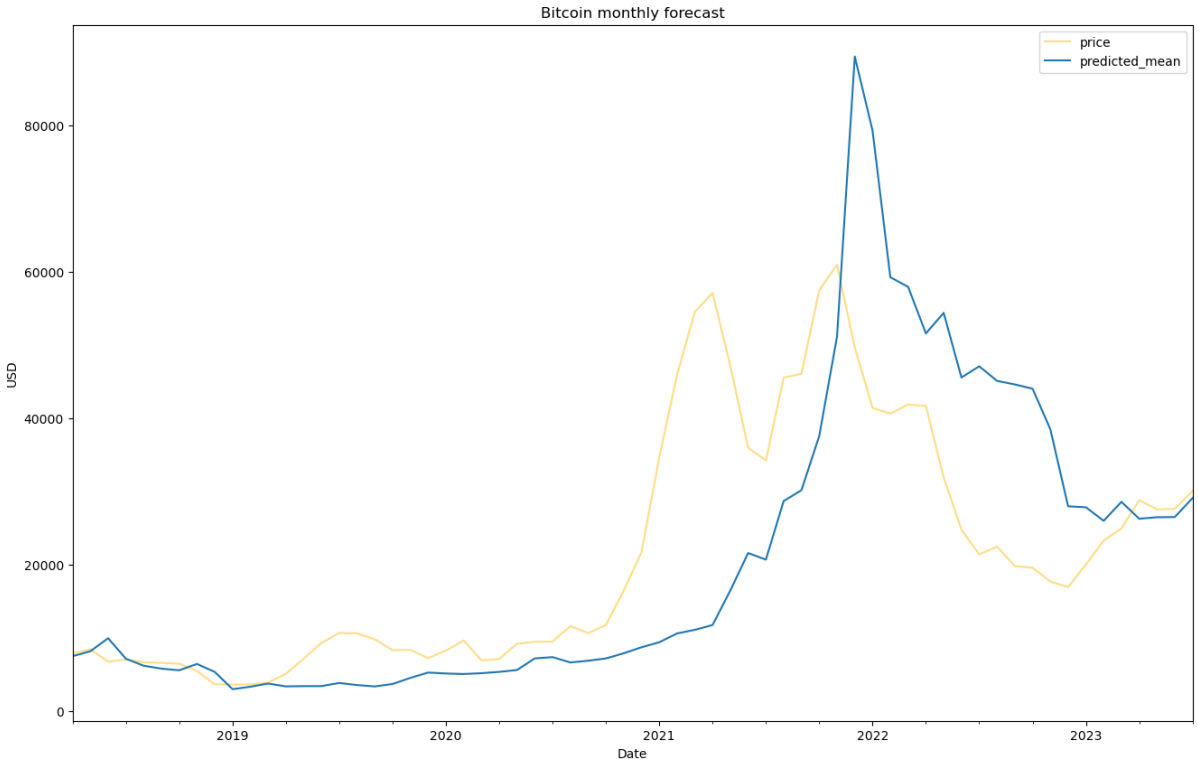

Figure 4. Predicting the Bull Market from 2020.

In an attempt to forecast the BTC bull market starting from 2020, the model produced a significantly higher Mean Absolute Percentage Error (MAPE) of 45.64%. While this indicates a considerable deviation in exact price levels, the overall shape of the prediction remains well-aligned with the actual market trend. Despite the large error margin, the model successfully captured the general direction of the price movement, which, in hindsight, could have served as a valuable financial indicator. This suggests that even with lower precision, such forecasts might provide useful insights for long-term investment strategies.

Model 2: GRU NN

To train and evaluate our model we firstly split the dataset into training (80%) and testing (20%) sets. This ensured that the model learned from historical data while being validated on unseen data to assess their generalization capability. Then the dataset was

normalized using MinMaxScaler, which scaled values between 0 and 1, helping improve training stability. Next we transformed the dataset into sequential inputs, where each input contained 360 previous time steps to predict the next price.
The model was trained using mean squared error (MSE) loss and optimized using the Adam optimizer. To prevent overfitting, we applied Early Stopping, which monitored validation loss and stopped training when no further improvement was observed.
In implementation we used Sequential model consisting of three GRU layers with activation functions tanh. The first layer has 128 units and the two following have 64 hidden units. What is more the return sequence parameter is set to true. After each of the aforementioned layers we added the dropout layer with the rate of 20% in order to prevent overfitting. Next we added the dense layer with 32 units and activation function relu to learn additional nonlinear patterns in the extracted features. Finally we added a dense layer with 1 unit as an output layer. The structure of the model is additionally presented in Figure 5.

Layer (type)	Output Shape	Param #
gru (GRU)	(None, 360, 128)	50,304
dropout (Dropout)	(None, 360, 128)	0
gru_1 (GRU)	(None, 360, 64)	37,248
dropout_1 (Dropout)	(None, 360, 64)	0
gru_2 (GRU)	(None, 64)	24,960
dropout_2 (Dropout)	(None, 64)	0
dense (Dense)	(None, 32)	2,080
dense_1 (Dense)	(None, 1)	33

Figure 5. Visualisation of the model structure.

After defining the model we proceeded to the training step using previously discussed datasets. Then we plot the history of the model training and validation loss in order to verify whether the model converges correctly (Figure 6) and presented the final training and validation loss in Table 2.

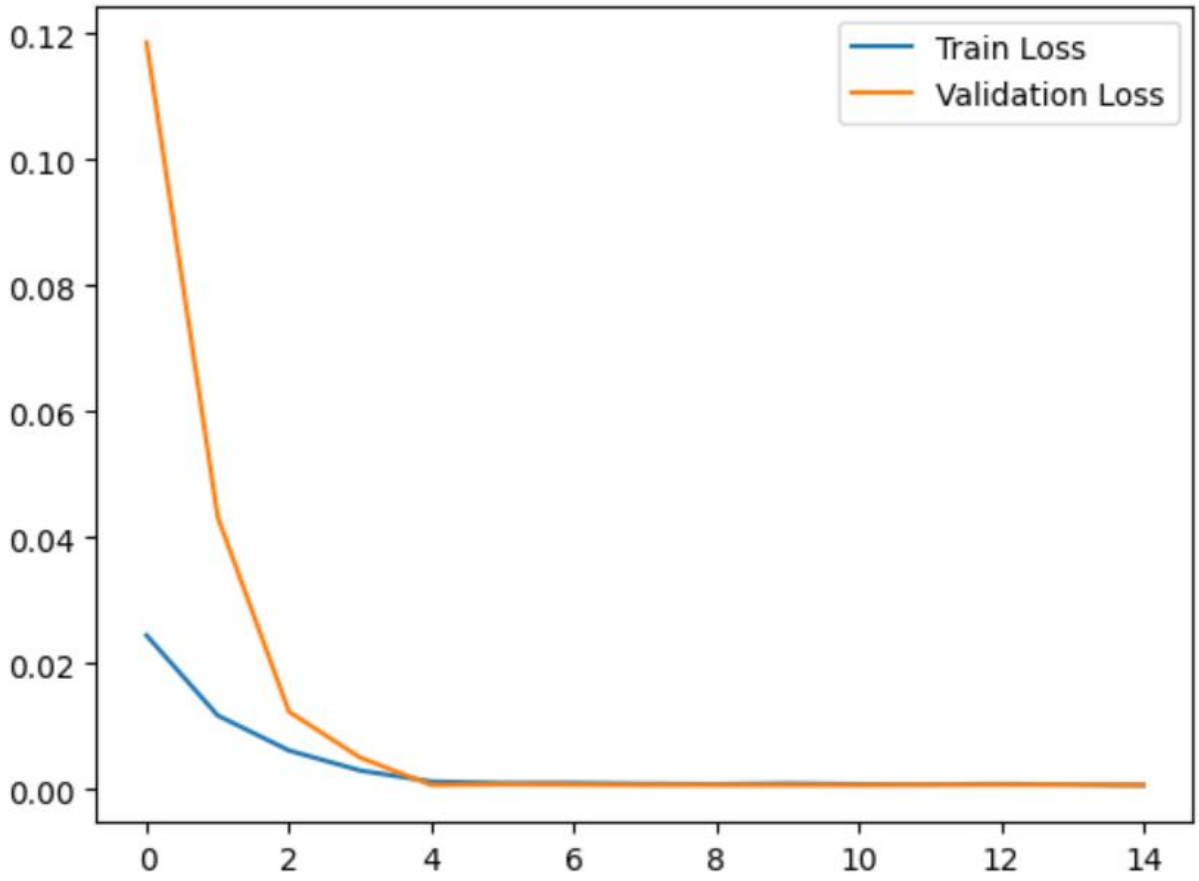

Figure 6. Model training loss.

Table 2. Training loss.

Learning Rate	Training Loss	Validation Loss
0.0001	0.000661	0.000977

In the following step, in order to assess model performance, we used the following metrics:

- Root Mean Squared Error (RMSE), which penalizes large errors more than MAE, providing insight into prediction accuracy.
- Mean Absolute Percentage Error (MAPE), which expresses error as a percentage, making it easier to interpret performance.
- Visualisation of predicted prices and visual comparison.

The outcome is presented in Figure 7 and 8.

```
Root Mean Square Error : 0.0291
Mean Absolute Percentage Error :  4.7767 %
```

Figure 7. GRU model performance.

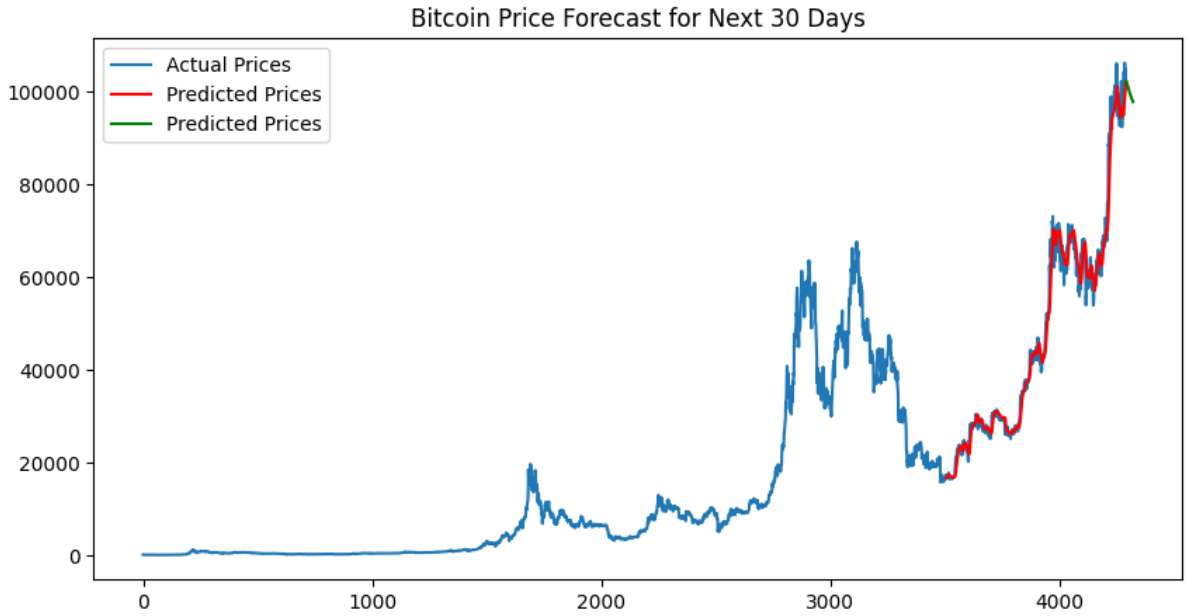

Figure 8. GRU price prediction.

The low MAPE and high alignment of the charts indicates that the model effectively captured historical trends and seasonal patterns, making it a strong choice for cryptocurrencies price forecasting.

- **Comparison of models:**

While SARIMAX provided better interpretability and captured long-term trends effectively, GRU demonstrated superior performance in modelling complex, nonlinear dependencies in cryptocurrency price movements. The GRU model's ability to learn from sequential patterns allowed it to adapt better to market volatility. However, SARIMAX remains useful for understanding seasonal trends and general price directions. To conclude we obtained overall better predicted results for future from SARIMAX model, however the GRU model is also promising and has great potential for further development.

7. Discussion

- Challenges Encountered

During the model development process, we faced several challenges. One of the primary issues was the high volatility of cryptocurrency prices, making it difficult to achieve consistently accurate predictions. Additionally, due to the high correlation among features, we decided to use only a single feature—the historical price itself—to avoid redundancy. Another challenge was selecting optimal hyperparameters for both the SARIMAX and GRU models. The SARIMAX model required careful tuning of seasonal parameters, and we experimented with different starting dates and time intervals to improve performance. For the GRU model, we had to fine-tune the number of layers, units per layer, dropout rates, and sequence length, which was computationally expensive.

- Interpretation of Results

The results indicate that both SARIMAX and GRU models have their strengths and limitations. The SARIMAX model performed well in capturing seasonal trends and long-term patterns, but it struggled with sudden price spikes and market anomalies. On the other hand, the GRU model was more adaptable to rapid price fluctuations and short-term dependencies, leading to lower error metrics compared to SARIMAX.

- Model Improvement

Several improvements could enhance the predictive accuracy of the models. First, further hyperparameter tuning using a more extensive grid search or Bayesian optimization could yield better configurations. For the SARIMAX model, experimenting with additional exogenous variables, such as trading volume, social media sentiment, or macroeconomic indicators, might improve performance. For the GRU model, increasing the depth of the network, adding attention mechanisms, or incorporating a hybrid approach could enhance robustness. Additionally, using more extensive historical data and refining the train-test split strategy might lead to better generalization and improved forecasts.

8. Conclusion

In this study, our aim was to predict prices of the bitcoin using two different machine learning approaches: SARIMAX and a GRU neural network. Our methodology involved preprocessing historical price data, selecting a single feature due to high correlation among variables, and applying two predictive models to analyse price trends. The SARIMAX model was fine-tuned by experimenting with different seasonal parameters, starting dates, and monthly price aggregation, while the GRU model utilized sequence-based input, and Early Stopping to enhance training performance.

The SARIMAX model yielded the best results, primarily because it allowed us to predict prices over a longer time horizon with relatively stable performance. While the GRU model performed well in capturing short-term fluctuations, its predictions became less reliable over extended periods due to the limitations of deep learning in time series forecasting without additional external factors. SARIMAX, on the other hand, effectively leveraged historical patterns and seasonal trends, making it the most robust and interpretable model for long-term cryptocurrency price prediction.

For future improvements, we could enhance the model by integrating additional external features, such as market sentiment analysis, or macroeconomic indicators, to improve prediction accuracy. Additionally, a hybrid approach, combining SARIMAX for long-term trends and GRU for short-term volatility, could be explored. Further optimization, including Bayesian hyperparameter tuning, or adding attention mechanisms to the GRU model may also yield promising results.

9. References

[1] “https://www.kaggle.com/datasets/kannapat/btc-usd-historical-price-2014-2024.”

[2] “https://www.kaggle.com/datasets/novandraanugrah/bitcoin-historical-datasets-2018-2024.”

[3] “https://www.coingecko.com/en/api.”

[4] “https://www.coingecko.com/pl/waluty/bitcoin/historical_data.”

[5] “https://www.kaggle.com/code/fangya/cryptocurrency-data-visualization-arima.”

[6] “https://www.kaggle.com/code/rifkyahmadsaputra/prediction-bitcoin-prices-with-gru-rnn.”