#### **Predicting-BTC-USD-Prices-Using-MLOps-on-AWS**

#### **Introduction**

This report outlines the development, deployment, and evaluation of a predictive model for BTC-USD prices using Machine Learning Operations (MLOps) techniques. The project leverages Flask for local deployment, MLflow for experiment tracking, and explores both GRU and LSTM models for predicting cryptocurrency prices.

#### **Project Overview**

The objective is to build a robust predictive model that can forecast hourly BTC-USD prices based on historical data sourced from the CryptoCompare API. The workflow encompasses data fetching, preprocessing, model training, evaluation, and deployment. The system is designed to operate locally and on cloud platforms like AWS SageMaker for scalability.

#### **Methodology**

**1. Data Collection and Preprocessing**

* **Data Source:** Historical BTC-USD price data was obtained from the CryptoCompare API using the requests library.
* **Data Preprocessing:**
  + Price data was normalized using MinMaxScaler to scale values between 0 and 1.
  + Sequential data was prepared using sliding windows to facilitate input into the models.

**2. Model Selection and Training**

* **Model Architectures:** Two recurrent neural network (RNN) architectures were evaluated:
  + **GRU Model:** A Sequential model with a GRU layer, dropout regularization, and dense output layer.
  + **LSTM Model:** An alternative LSTM model was explored for comparison with the GRU model.
* **Model Training:**
  + Models were trained using hyperparameters like window size, GRU units, dropout rate, epochs, and batch size.
  + Training was conducted locally, with metrics logged using MLflow for tracking and analysis.

**3. Evaluation and Metrics**

* **Evaluation Metrics:**
  + Mean Squared Error (MSE)
  + Mean Absolute Error (MAE)
* **Performance Assessment:** Both GRU and LSTM models were evaluated based on their ability to predict BTC-USD prices accurately. Metrics were calculated to assess model performance against actual data.

**4. Deployment**

* **Local Deployment:** Flask was utilized to deploy the trained model locally, exposing endpoints for real-time predictions.
* **MLflow Integration:** MLflow was integrated into the workflow to log experiment parameters, metrics, and model versions systematically.
* **SageMaker Deployment:** The model was also deployed on AWS SageMaker for cloud-based predictions, utilizing its scalable infrastructure.

**5. Challenges Faced**

* **System Stability:** Issues with Ubuntu's stability resulted in intermittent disconnections during development and deployment phases.
* **Model Performance:** Performance variations were observed between GRU and LSTM models based on hyperparameter tuning and data preprocessing techniques.
* **Connection issues:** connecting our project online

#### **Findings and Results**

* **Model Performance:** Both GRU and LSTM models achieved reasonable accuracy in predicting BTC-USD prices. The GRU model demonstrated slightly better performance in certain configurations, highlighting the importance of architecture selection and hyperparameter tuning.

Current paramenters: window\_size, lstm\_units, dropout\_rate, epochs, batch\_size

#### **Conclusion**

The project successfully implemented a predictive model for BTC-USD prices using advanced RNN architectures and MLOps practices. Local deployment via Flask and integration with MLflow enabled efficient experimentation and evaluation. Challenges related to system stability and model performance will need to be resolved before we deploy.

#### **Future Directions**

* **Enhanced Data Features:** Incorporate additional features such as sentiment analysis or macroeconomic indicators for improved prediction accuracy.
* **Automated Pipelines:** Develop automated pipelines for seamless data fetching, preprocessing, training, and deployment.
* **Advanced Architectures:** Explore transformer-based architectures like Transformers or hybrid models for enhanced time-series forecasting capabilities.

#### **Recommendations**

* **Strengthen System Stability:** Address system stability issues on Ubuntu or consider alternative operating systems to ensure uninterrupted development and deployment workflows.
* **Optimize Hyperparameters:** Continue to fine-tune model hyperparameters and experiment with different architectures to improve prediction accuracy and reliability.