Capstone Project Proposal

Project Title: *Predicting Short-Term Cryptocurrency Returns Using Machine Learning (G-Research Crypto Forecasting)*

---

1. Domain Background

Cryptocurrency markets are volatile, nonlinear, and affected by strong cross-asset correlations. Machine learning—especially gradient-boosting models like LightGBM—has become a popular approach in quantitative finance for modeling noisy, high-frequency financial data.

This project uses the dataset from the **G-Research Crypto Forecasting Kaggle competition**, which provides minute-level OHLCV data and a future return “Target” for multiple crypto assets. The objective is to predict short-term price movements, a task that can meaningfully contribute to algorithmic trading strategies even when predictive correlations are small.

---

2. Problem Statement

The task is:

> ***To predict short-term cryptocurrency returns (the provided “Target”) using historical price data and engineered features.***

Challenges include:

- Highly noisy target values
- Missing timestamps and asset-specific trading periods
- The need to avoid data leakage
- Non-stationary market behavior

A successful predictive model must capture meaningful patterns while remaining robust to market noise.

---

3. Datasets and Inputs

The dataset originates from the **G-Research Crypto Forecasting** Kaggle competition. In this project, I use a **preprocessed version** (`merged_df`) that contains:

Column	Description

Column	Description
`timestamp`	Unix time (minute-level)
`Asset_ID`	Identifier for each asset
`Close`	Close price
`Target`	15-minute future return (provided by competition)

Data Preparation

1. Align timestamps across assets
2. Forward-fill missing Close data (limit = 60 minutes)
3. Sort by timestamp
4. Generate lag-based and market-relative features
5. Remove rows with missing values

Features Engineered

- Lag returns: `return_1m`, `return_5m`, `return_15m`, `return_30m`, `return_60m`
- Rolling trend indicators: `trend_15m`, `trend_60m`
- Cross-sectional deviations: `diff_ret_1m`, … `diff_ret_60m`
- Market price deviation: `diff_price` (optional)

Final dataset size after processing: **~1.2–1.5 million rows** (depending on filtering).

---

4. Proposed Solution

The proposed solution uses **LightGBM regression** to model the short-term price return (`Target`). Reasons for choosing LightGBM:

- Handles nonlinear tabular patterns well
- Efficient for large datasets
- Supports early stopping and fast training
- Popular in financial competitions

Model Pipeline

1. **Preprocess data** (timestamp alignment, forward-fill, feature generation)
2. **Use forward-chaining time-series cross-validation** (7 folds)
3. **Train LightGBM models** with parameters:

```
learning_rate = 0.05
num_leaves = 256
n_estimators = 10000
early_stopping_rounds = 50
```

4. **Evaluate models using correlation metric**
5. **Select the best fold model**
6. **Log everything to MLflow** (parameters, metrics, artifacts)

The entire process simulates real-world forecasting and avoids data leakage.

---

5. Benchmark Model

To determine whether the ML model adds value, we compare it to two simple baselines.

Benchmark 1 — Zero Prediction

Predict future return = 0.

Expected correlation: **≈ 0**

Benchmark 2 — Copy Last Return

Predict:

```
ŷ_t = return_1m(t)
```

Expected correlation: **≈ 0.01 – 0.015**

Any model achieving **> 0.02 correlation** is considered meaningful in this domain.

---

6. Evaluation Metrics

The primary metric is:

Pearson Correlation Coefficient

[corr = \frac{cov(y, \hat)}{\sigma_y \sigma_{\hat}}]

Reasons for using correlation:

- Return magnitude is less important than directional accuracy
- RMSE is not meaningful in noisy financial targets
- Correlation is the official competition metric
- Robust to scaling differences

Secondary metrics:

- Fold correlation values
- Average correlation across all CV folds

---

7. Project Design

Step 1 — Data Processing

- Align timestamps
- Forward-fill Close prices
- Construct lag-based and market-based features

Step 2 — Time-Series CV Split

Use 7-fold forward-chaining split:

```
Train: [start → t1], Validate: [t1 → t2]
Train: [start → t2], Validate: [t2 → t3]
...
```

Step 3 — Train LightGBM

Track:

- Parameters
- Per-fold metrics
- Best model checkpoint

Step 4 — Model Logging with MLflow

- Store best model under artifact_path="model"
- Track fold metrics
- Register model for reproducibility

Step 5 — Model Evaluation

- Compare with benchmark baselines
- Compute average correlation
- Review feature importances

Step 6 — Final Results and Report

The final deliverables will include:

- Python training script
- Preprocessing code
- MLflow experiment results
- Model artifact
- Capstone proposal PDF (this document)

Expected Outcome

A LightGBM model achieving:

```
Average correlation ≈ 0.02 – 0.05
```

Which outperforms the baseline and demonstrates real predictive power in short-term crypto returns.

Conclusion

This project applies machine learning techniques to a real-world financial forecasting problem. By combining:

- Feature engineering
- Time-series validation
- Gradient-boosted models
- MLflow experiment tracking

the project aims to build a reproducible, production-grade forecasting pipeline. Even a small improvement in predictive correlation can be valuable in algorithmic trading.