Pipeline Architecture Document

Cryptocurrency Liquidity Prediction Project

Contents

1 Overview **2**

2 Objective **2**

3 Pipeline Architecture Stages **2**

4 Data Flow Diagram **3**

5 Benefits of the Pipeline **3**

6 Conclusion **4**

1 Overview

This document provides an architectural walkthrough of the full machine learning pipeline for the cryptocurrency liquidity prediction system. It describes each stage, from raw data to deployed prediction, and visualizes how data flows through the process.

2 Objective

To architect a robust pipeline that transforms raw crypto market data into actionable liquidity predictions, while maintaining modularity, interpretability, and ease of deployment.

3 Pipeline Architecture Stages

The data pipeline follows these primary stages:

1. **Raw Data Ingestion**:
   - Input files: CoinGecko historical CSVs located in `data/raw/`
   - Collected over multiple dates and assets (e.g., BTC, ETH)
2. **Data Preprocessing** (`01_data_preprocessing.ipynb`):
   - Missing value imputation
   - Date formatting and parsing
   - Temporal feature extraction
   - Output: `cleaned_crypto_price.csv`
3. **Exploratory Data Analysis** (`02_eda.ipynb`):
   - Visual analysis of price trends, volatility, volume
   - Correlation and box plots
   - Seasonality and decomposition
4. **Feature Engineering** (`03_feature_engineering.ipynb`):
   - Create lag features, rolling means, percentage changes
   - Normalize or scale numerical fields
   - Output: `features.csv`
5. **Model Selection** (`04_model_selection.ipynb`):
   - Compare Linear, Ridge, Lasso, Random Forest, XGBoost, LightGBM
   - Choose best performer using RMSE, MAE, R^2
6. **Model Training** (`05_model_training.ipynb`):
   - Train XGBoost with optimized hyperparameters

- Save model as `final_xgboost_model.pkl`

7. **Deployment** (`deployment/app.py`):
   - Streamlit interface for real-time prediction
   - Loads trained model and takes user inputs

4 Data Flow Diagram

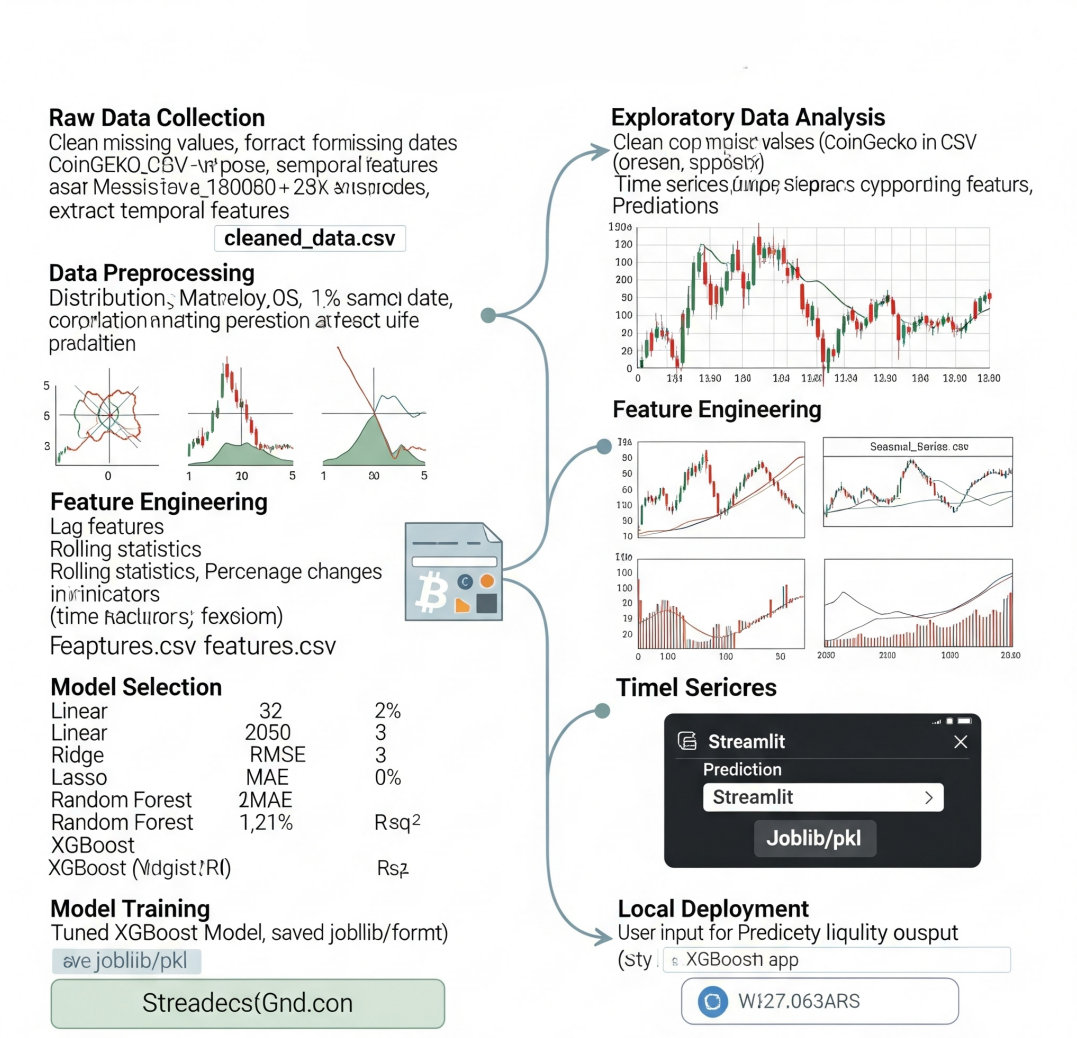

Figure: Data Flow from Raw to Prediction

5 Benefits of the Pipeline

- **Modularity:** Every stage is encapsulated in its own script/notebook
- **Maintainability:** Easy to swap models or adjust preprocessing
- **Reusability:** Can adapt to other time-series prediction problems
- **Scalability:** Can be extended to multi-asset forecasting or deployed to cloud pipelines

6 Conclusion

This pipeline architecture forms the backbone of the project, enabling a clean transition from raw data to robust, explainable, and scalable liquidity predictions. The architecture supports future enhancements including cloud-based prediction APIs or integration with live market feeds.