Cryptocurrency Volatility Prediction Project
Final Report

1 Project Overview

This project focuses on predicting 7-day volatility for various cryptocurrencies using machine learning techniques. The workflow includes exploratory data analysis(EDA),feature engineering, model training, and deployment of a Streamlit based application(`app.py`) for real-time volatility predictions. The dataset, sourced from `dataset.csv`, contains historical price and volume data for multiple cryptocurrencies, which was preprocessed and enriched with technical indicators to enhance prediction accuracy.

The primary goal was to develop a robust model to forecast cryptocurrency volatility, visualized through an interactive interface that Includes candlestick charts and simulated price paths. The project leverages a LightGBM model, identified as the best performer, to provide reliable predictions.

2 Data Collection and Exploratory DataAnalysis(EDA)

2.1 Data Source

- **Dataset**: `dataset.csv`
- **Structure**: Contains 72,946 rows and 9 columns: `open`, `high`, `low`, `close`, `volume`, `marketCap`, `timestamp`, `crypto_name`, and `date`.
- **Key Observations**:
  - No missing values or duplicates were found, ensuring data integrity.
  - The dataset covers multiple cryptocurrencies, with `crypto_name` identifying each coin.
  - Price columns (`open`, `high`, `low`, `close`) and `volume` exhibited significant variation across cryptocurrencies, indicating diverse market behaviors.
  - `timestamp` and `date` columns were converted to datetime format for temporal analysis.

2.2 EDA Insights (`EDA.ipynb`)

- **Price Distribution**: Histograms with KDE curves for `open`, `high`, `low`, and `close` prices were generated for each cryptocurrency. These visualizations revealed:
  - Price clustering and skewness, highlighting common price ranges and outliers.
  - Overlapping price distributions for some cryptocurrencies, suggesting similar price behaviors.

- **Volume and Market Cap**: Bar plots showed significant differences in trading activity and market dominance across cryptocurrencies.
- **Recommendations**: No columns were dropped during EDA, but feature engineering (e.g., outlier removal, scaling) was suggested for model preparation.

3 FeatureEngineering(`feature_engineering.ipynb`)

Feature engineering was critical to enhance the predictive power of the model. The following features were derived or added:

- **Datetime Features**:
  - Extracted `year`, `month`, `day`, and `day_of_week` from the `date` column to capture temporal patterns relevant to volatility.
- **Technical Indicators**:
  - **Log Return**: Calculated as $\log(\frac{close}{open})$ to measure daily price changes.
  - **Volatility (14-day, 30-day)**: Computed as the standard deviation of log returns over 14 and 30-day windows to capture short- and mediumterm price fluctuations.
  - **Average True Range (ATR_14)**: Measured as the mean of true ranges over14days, where true range is $\max(\text{high}-\text{low}, |\text{high}-\text{close}_{t-1}|, |\text{low}-\text{close}_{t-1}|)$.
  - **BollingerWidth**: Calculatedas $\frac{\text{upper band}}{\text{rolling mean}}-\text{lower band}$, usinga14-dayrolling mean and standard deviation to gauge price volatility.
  - **Liquidity Ratio**: Defined as $\frac{volume}{marketCap}$to assess trading activity relative to market size.
  - **True Range (TR)**: Computed as $\max(\text{high} - \text{low}, |\text{high} - \text{close}|, |\text{low} - \text{close}|)$.
- **One-Hot Encoding**: Applied to `crypto_name` to create binary columns for each cryptocurrency, enabling the model to differentiate between coins.

The final dataset(`processed_crypto_with_target.csv`) contains 74columns, including the target variable `Volatility_7_target`, which represents the 7day volatility to be predicted.

4 Model Training (`MODEL_TRAINING.IPYNB`)

A variety of regression models were evaluated using a time-series split (80% training, 20% testing) to ensure chronological consistency. The evaluation metrics were Root Mean Squared Error (RMSE) and R^2 score.

4.1 Model Performance

Table 1: Model Evaluation Results

Model	RMSE	R^2
LGBMRegressor	27.0389	0.5103
CatBoostRegressor	27.3257	0.4999
GradientBoostingRegressor	27.7141	0.4855
RandomForestRegressor	27.7280	0.4850
XGBRegressor	28.6328	0.4509
Ridge	32.3472	0.2992
LinearRegression	32.3658	0.2984
Lasso	32.6487	0.2860
ElasticNet	33.2521	0.2594
AdaBoostRegressor	37.3831	0.0640
DecisionTreeRegressor	39.4679	-0.0434
SVR	39.5679	-0.0486
KNeighborsRegressor	39.6628	-0.0537

4.2 Hyperparameter Tuning

The LightGBM model was selected for hyperparameter tuning due to its superior performance (RMSE: 27.0389, R^2: 0.5103). A grid search with time-series crossvalidation (5 splits) was performed over the following parameters:

- `num_leaves`: [31, 50, 70]
- `learning_rate`: [0.01, 0.05, 0.1]
- `n_estimators`: [100, 200, 500]
- `max_depth`: [-1, 10, 20]
- `min_child_samples`: [20, 50, 100] **Best Parameters**:
- `learning_rate`: 0.05
- `max_depth`: 20
- `min_child_samples`: 50
- `n_estimators`: 200
- `num_leaves`: 31

Best RMSE (Cross-Validation): 31.0266

ThetunedLightGBMmodelwastrainedonthefulldatasetandsavedas`best_model.pkl` for deployment.

5 Application Development (`app.py`)

A Streamlit application was developed to provide an interactive interface for volatility predictions. Key features include:

- **User Inputs**:
  - Cryptocurrency selection from a dropdown menu.
  - Date input for prediction context.
  - Numerical inputs for `open`, `high`, `low`, `close`, `volume`, `marketCap`, `ATR_14`, and `Bollinger_Width`, with predefined ranges to ensure valid inputs.
- **DerivedFeatures**: Calculatedinreal-time, including`Log_Return`, `Liquidity_Ratio`, `TR`, `Volatility_14`, `Volatility_30`, `ATR_14`, and `Bollinger_Width`.
- **Prediction**: The LightGBM model predicts 7-day volatility, displayed with four decimal precision.
- **Visualizations**:
  - **Candlestick Chart**: Displays OHLC data for the selected date with an annotation for predicted volatility.
  - **Price Path Simulation**: Generates 100 simulated 7-day price paths using the predicted volatility, with a mean path and 95% confidence interval.

5.1 Error Handling

The application includes robust error handling for:

- Missing model or data files.
- Invalid input ranges, with warnings for out-of-range values.
- Feature mismatches between model expectations and user inputs.

6 Challenges and Resolutions

- **Feature Mismatch**: The model expects 73 features, but user inputs may lack some (e.g., historical volatility). Resolved by filling missing numerical features with dataset means and ensuring all expected columns are present.
- **Scaling**: Numerical features were scaled using `StandardScaler` to match the training data, ensuring consistent model performance.
- **Time-Series Nature**: A chronological split was used for model evaluation to prevent data leakage, and derived features (e.g., volatility) were computed with time-window constraints.

7 Conclusions and Recommendations

7.1 Key Findings

- The LightGBM model outperformed other models, achieving an RMSE of 27.0389 and an R^2 of 0.5103, indicating moderate predictive power for 7day volatility.
- Feature engineering significantly improved model performance by incorporating technical indicators like volatility, ATR, and Bollinger Width.
- The Streamlit application provides a user-friendly interface for real-time volatility predictions, enhanced by visualizations that aid interpretation.

7.2 Limitations

- Themodel'sR²of0.5103suggestsroomforimprovementincapturingvolatility dynamics.
- Limited historical data for some cryptocurrencies may affect feature calculations (e.g., volatility over short windows).
- Theapplicationreliesonuser-providedinputsfortechnicalindicators, which may introduce inaccuracies if not aligned with historical data.

7.3 Recommendations

- **Model Enhancement**: Explore deep learning models (e.g., LSTM) to capture temporal dependencies in volatility patterns.
- **Feature Expansion**: Incorporate externaldata(e.g., market sentiment, news events) to improve prediction accuracy.
- **Real-Time Data**: Integrate an API for live cryptocurrency data to automate feature calculations and reduce reliance on manual inputs.
- **User Interface**: Enhance the Streamlit app with historical data visualizations and model confidence metrics.

8 References

- Dataset: `dataset.csv`, `processed_crypto_with_target.csv`
- Notebooks: `EDA.ipynb`, `feature_engineering.ipynb`, `MODEL_TRAINING.IPYNB`
- Application: `app.py`
- Libraries: pandas, numpy, scikit-learn, LightGBM, Streamlit, Plotly