Trading Project - Milestone 2

This project implements a quantitative trading strategy focusing on order execution optimization using machine learning and market microstructure analysis.

Project Structure

Core Components

- `main.py`: Main execution script that orchestrates the entire pipeline
- `data_loader.py`: Handles data loading and preprocessing
- `data_preprocessor.py`: Processes raw market data into structured format
- `factors.py`: Implements various market microstructure factors
- `targets.py`: Defines target variables for prediction
- `scoring_model.py`: Machine learning model for generating trading signals
- `strategy.py`: Core trading strategy implementation
- `model_evaluation.py`: Evaluates model performance
- `factor_analysis.py`: Analyzes factor performance and relationships

Data Flow

1. **Data Processing**
   - Raw market data is processed through `data_preprocessor.py`
   - Processed data is stored in `processed_data/` directory
   - Data includes order book, trades, and market microstructure features
2. **Feature Engineering**
   - `factors.py` calculates various market microstructure factors
   - Factors include spread, volatility, order flow imbalance, etc.
3. **Model Training**
   - `scoring_model.py` trains LightGBM models for each target
   - Models are saved in `models/` directory
   - Training uses top 50 features for each target
4. **Strategy Execution**
   - `strategy.py` implements the core trading logic
   - Uses dynamic thresholds based on market conditions
   - Records execution quality metrics

Key Outputs

1. **Model Outputs**
   - Model files: `models/{symbol}_{target}_model.joblib`
   - Score files: `results/{symbol}_{target}_scores.csv`
2. **Strategy Results**
   - Execution records in `strategy_results/` directory
   - Performance metrics and analysis reports
3. **Analysis Outputs**
   - Factor analysis results in `analysis/` directory

- Score distribution plots in `figure/` directory

Dependencies

Key dependencies are listed in `requirements.txt`: - pandas - numpy - lightgbm - matplotlib - scikit-learn

Workflow

1. **Data Preparation**

   ```
   python data_preprocessor.py
   ```

2. **Feature Generation**

   ```
   python main.py
   ```

3. **Strategy Testing**

   ```
   python test_strategy.py
   ```

4. **Model Evaluation**

   ```
   python model_evaluation.py
   ```

Key Features

- **Dynamic Thresholds**: Strategy adjusts execution thresholds based on market conditions
- **Market State Analysis**: Considers multiple market states for execution decisions
- **Performance Tracking**: Comprehensive transaction cost analysis (TCA)
- **Machine Learning Integration**: Uses LightGBM for signal generation
- **Multi-Symbol Support**: Currently supports AMZN, GOOG, MSFT, and INTC

Output Analysis

The project generates several types of analysis: 1. Score distributions for different targets 2. Execution quality metrics 3. Factor performance analysis 4. Strategy performance reports

Notes

- All processed data is stored in feather format for efficient I/O
- Logs are maintained in the `logs/` directory
- Results are organized by symbol and target type