**Code Summary**

**Overview**

This code implements an **AI-powered time series forecasting framework** specifically designed for **bi-weekly financial data prediction**. The system combines traditional machine learning models with Large Language Models (LLMs) to automatically generate feature engineering strategies.

**Key Components**

**1. FeatureEngineeringAgentLLM Class**

* **Purpose**: Uses an LLM to automatically generate feature engineering strategies
* **Key Methods**:
  + \_build\_prompt(): Creates prompts for the LLM describing the data structure
  + \_call\_llm(): Calls the LLM to get feature engineering recommendations
  + \_create\_lag\_features(): Creates lagged versions of features (e.g., previous 1, 2, 4 periods)
  + \_create\_rolling\_features(): Creates rolling statistics (mean, std) over time windows
  + \_create\_time\_features(): Extracts time-based features (month, quarter, day of week)
  + process\_features(): Main method that orchestrates feature creation

**2. ValidationStrategy Class**

* **Purpose**: Handles proper time series validation (prevents data leakage)
* **Key Methods**:
  + time\_based\_split(): Splits data chronologically into train/validation/test sets
  + expanding\_window\_validation(): Walk-forward validation for robust model evaluation

**3. ForecastingAgentLLM Class**

* **Purpose**: Main forecasting engine with comprehensive validation
* **Key Features**:
  + Supports **Random Forest** and **Linear Regression** models
  + Handles **panel data** (multiple entities × time periods)
  + Focuses on **bi-weekly forecasting** by filtering to actual target changes
* **Key Methods**:
  + \_prepare\_biweekly\_data(): Extracts only bi-weekly data points from daily data
  + validate\_with\_holdout(): Standard train/test validation
  + cross\_validate(): Expanding window cross-validation
  + \_calculate\_metrics(): Computes MAE, RMSE, MAPE, R²

**4. Framework Classes**

* **EnhancedOrchestratorAgent**: Coordinates the entire workflow
* **EnhancedAgenticForecastingFramework**: Main entry point with two modes:
  + run(): Basic forecasting (backward compatibility)
  + run\_with\_validation(): Enhanced workflow with proper validation

**Data Structure**

The framework expects **panel data** with:

* **MultiIndex**: (PERMNO, date) - entity identifier and date
* **Target**: currentShortPositionQuantity (bi-weekly financial metric)
* **Features**: Various financial indicators, market data, analyst recommendations

**Workflow Process**

1. **Data Analysis**: Analyzes input data structure and identifies column types
2. **LLM Feature Strategy**: Uses TinyLlama model to generate feature engineering plan
3. **Feature Engineering**: Implements the LLM's recommendations (lags, rolling stats, time features)
4. **Model Training**: Trains multiple models and selects best performer
5. **Validation**: Uses time-aware validation to prevent lookahead bias
6. **Forecasting**: Generates predictions with confidence intervals

**Usage Example**

python

framework = EnhancedAgenticForecastingFramework()

results = framework.run\_with\_validation(

panel\_df[columns],

target\_column='currentShortPositionQuantity'

)