

Using Satellite-Derived Surface Temperature Near Oil Refineries to Predict Heating Oil Prices

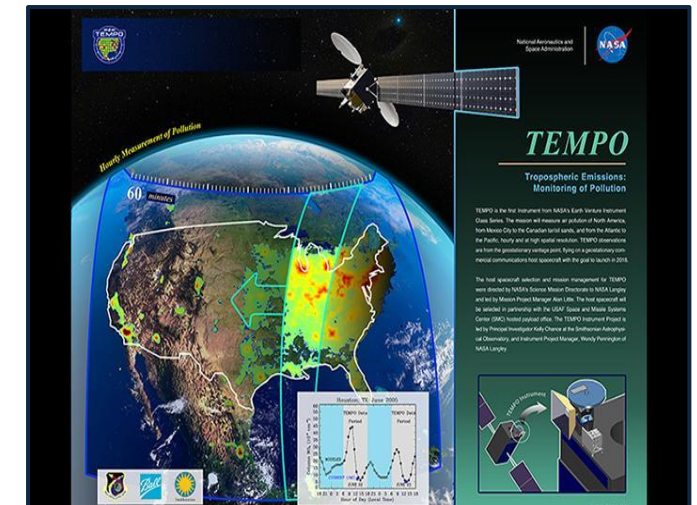
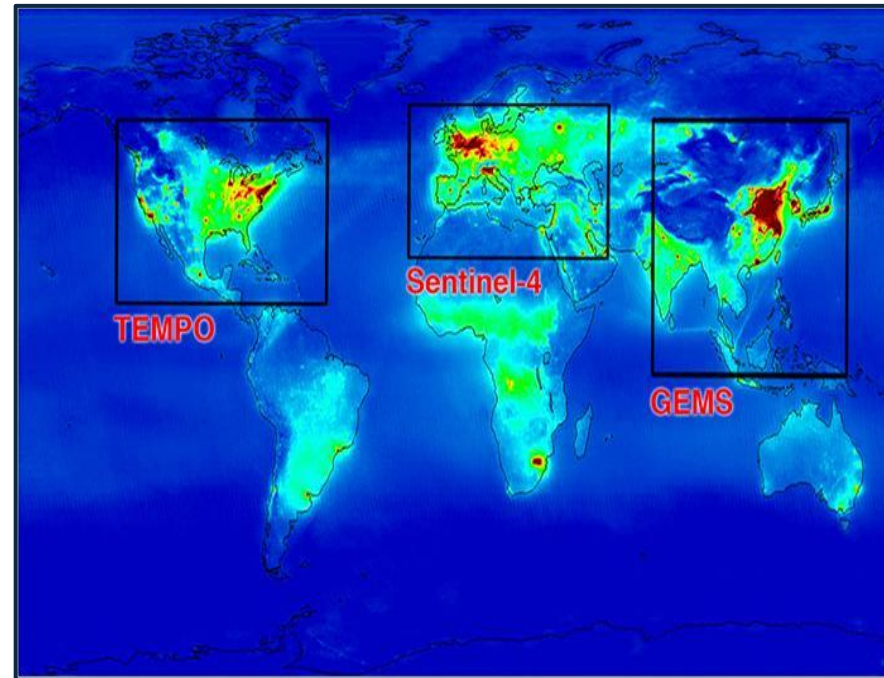
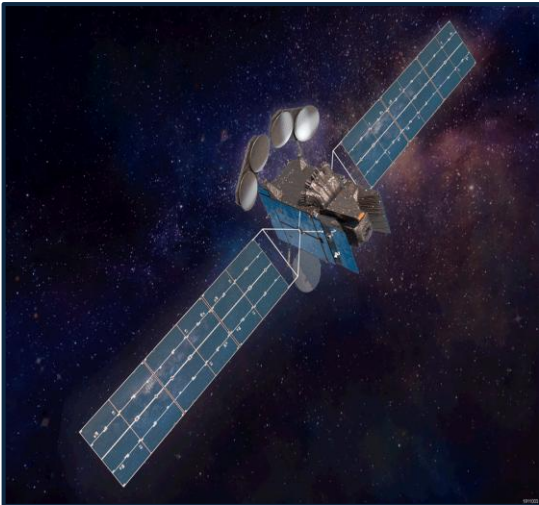
Matt Lertsmitivanta

- Thesis & Objectives
- Influence Behind the Project
- Model Selection
- Conclusion
- Next Steps

- **Goal:** Build a data-driven trading strategy using land surface temperature (LST) from satellite imagery to anticipate heating oil price movements.
- **Data:**
 - NYMEX: HO=F (heating oil futures)
 - MODIS satellite LST features (rolling, lagged, seasonal anomalies)
- **Methodology:**
 - PurgedKFold cross-validation for time-aware validation
 - Machine learning (ElasticNet & LightGBM) to predict returns
 - Rule-based strategy with z-score anomaly thresholds
- **Optimization:**
 - Strategy hyperparameters (lag, thresholds, volatility)
 - Train/test window selection via walk-forward backtest
- Leveraged parallel processing and geospatial feature extraction for efficient data fusion at scale.

Influence Behind the Project

- **Prev. Research Project @ MSFC TEMPO Mission**
 - Utilized satellite data to predict NO₂ on the surface of North America



- **Satellite**

MODIS Data: MODIS/Terra
Land Surface
Temperature/Emissivity
8-Day L3 Global 1km SIN
Grid V061



- **Total Predictive Features: 31**

- **Yahoo Finance: HO=F Price Data**

- **Time Horizon: 10 years**

- Start: 2015-1-1
- End: 2025-1-1

NY Mercantile - Delayed Quote • USD

Heating Oil Jun 25 (HO=F) ☆ Follow + Add holdings

2.1340 +0.0063 +(0.30%)

As of 4:59:55 PM EDT. Market Open.

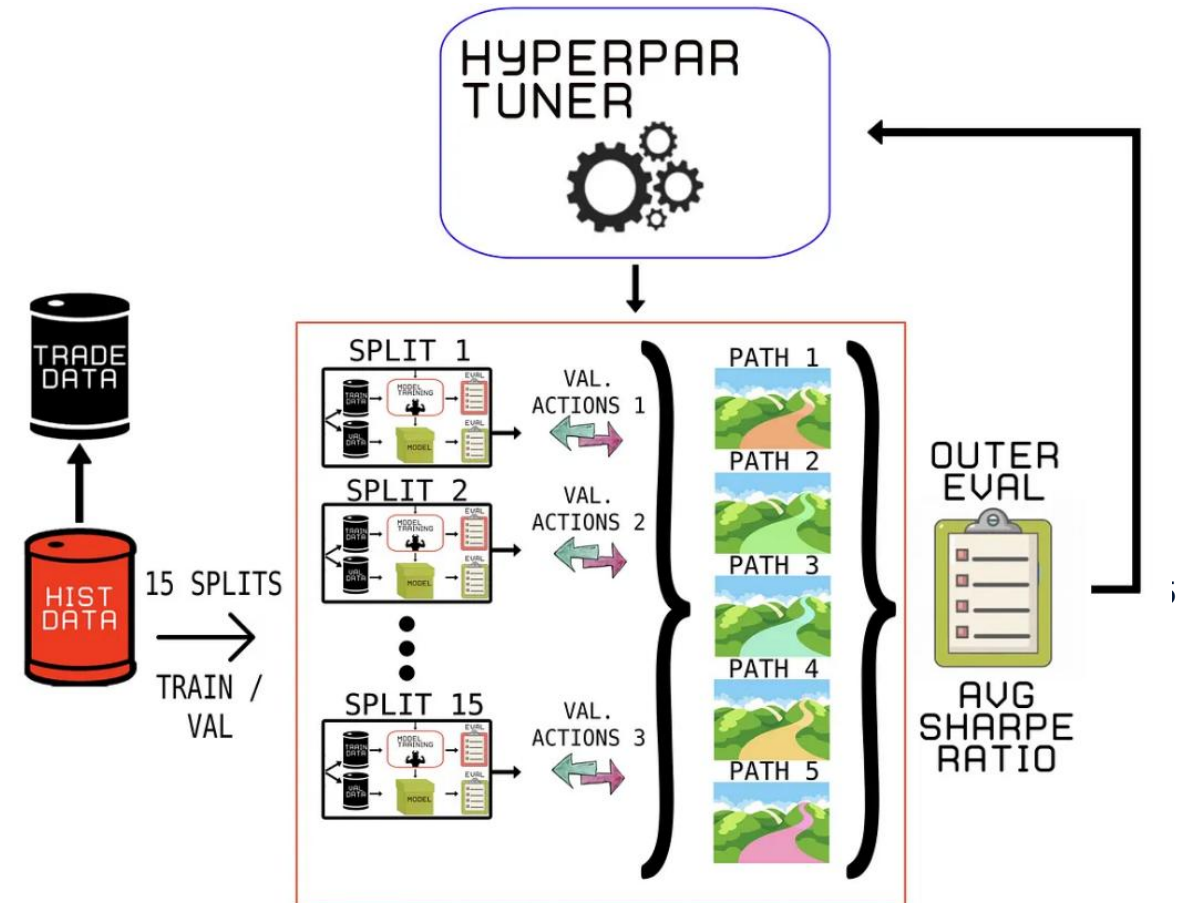
May 20, 2024 - May 20, 2025 Historical Prices Daily

Currency in USD

Date	Open	High	Low	Close	Adj Close	Volume
May 20, 2025	2.1273	2.1423	2.1089	2.1340	2.1340	30,444
May 19, 2025	2.1444	2.1483	2.1180	2.1277	2.1277	70,187
May 16, 2025	2.1672	2.1820	2.1350	2.1406	2.1406	70,187
May 15, 2025	2.1892	2.1929	2.1336	2.1660	2.1660	79,886
May 14, 2025	2.1800	2.2174	2.1594	2.2061	2.2061	85,492
May 13, 2025	2.1112	2.1828	2.1026	2.1713	2.1713	74,464
May 12, 2025	2.0770	2.1411	2.0696	2.1111	2.1111	70,987
May 9, 2025	2.0539	2.0806	2.0454	2.0664	2.0664	83,828
May 8, 2025	1.9701	2.0578	1.9700	2.0400	2.0400	83,452
May 7, 2025	2.0016	2.0330	1.9712	1.9766	1.9766	67,450
May 6, 2025	1.9788	2.0299	1.9757	2.0088	2.0088	60,075
May 5, 2025	1.9600	1.9901	1.9338	1.9745	1.9745	55,274

- **Model Pipeline**

- Preprocess Data
- Train Model, inside this, run PurgedKFolds for each split
- Identify best fold
- Tune hyperparameters for this fold
- Output is **optimized**



- Data Preprocessing & Inference

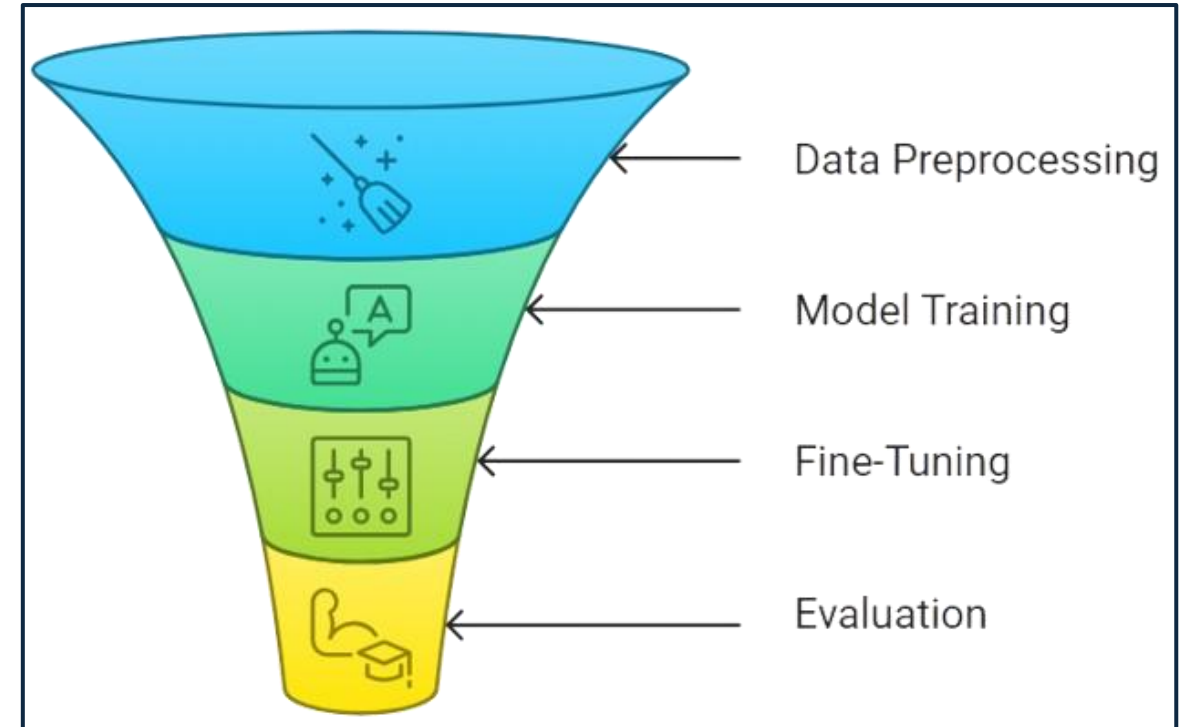
- PurgedKFold Validation

- Elastic Net Regularization

- Hyperparameter Tuning:
 - Alpha and L1 Ratio

- LightGBM (Gradient Boosting)

- Hyperparameter Tuning:
 - `n_estimators`, `learning_rate`, `num_leaves`, `min_child_weight`, `min_child_samples`, `reg_lambda`, `reg_alpha`, `linear_tree`, `subsample`, `subsample_freq`, `colsample_bytree`, `colsample_bynode`, `linear_lambda`, `min_data_per_group`, `max_cat_threshold`, `cat_l2`, `cat_smooth`



- **Data Preprocessing & Inference**

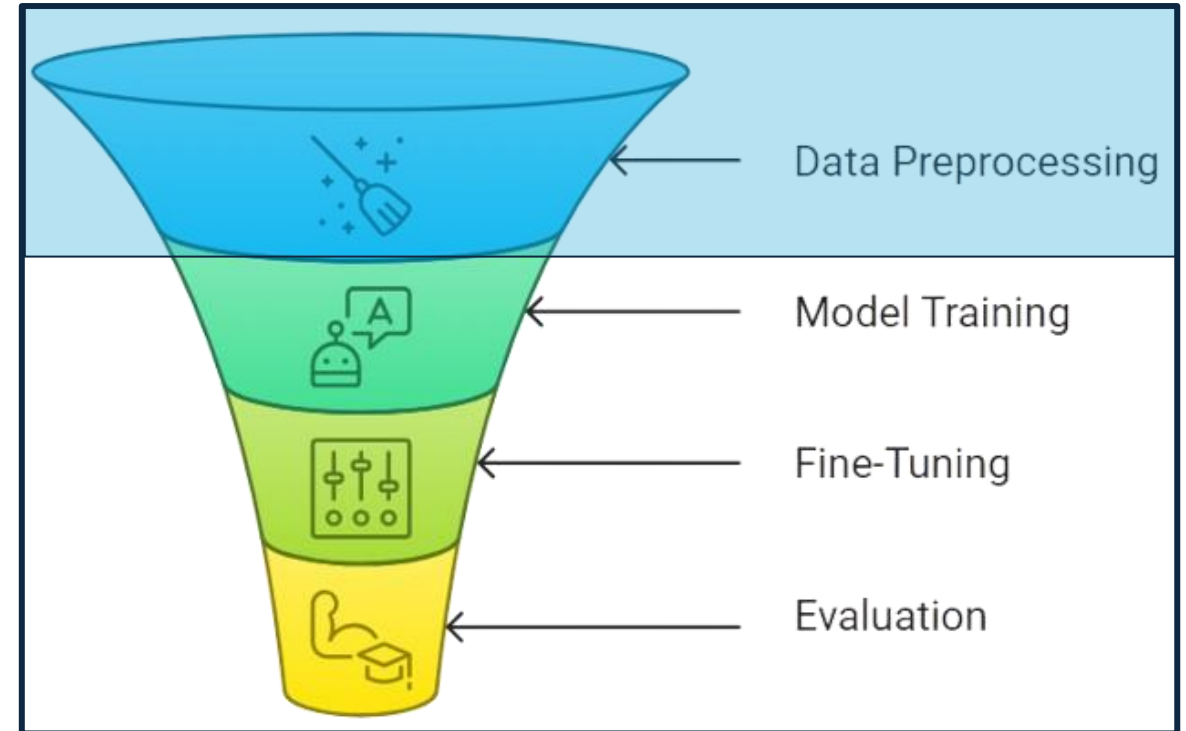
- PurgedKFold Validation

- Elastic Net Regularization

- Hyperparameter Tuning:
 - Alpha and L1 Ratio

- LightGBM (Gradient Boosting)

- Hyperparameter Tuning:
 - `n_estimators`, `learning_rate`, `num_leaves`, `min_child_weight`, `min_child_samples`, `reg_lambda`, `reg_alpha`, `linear_tree`, `subsample`, `subsample_freq`, `colsample_bytree`, `colsample_bynode`, `linear_lambda`, `min_data_per_group`, `max_cat_threshold`, `cat_l2`, `cat_smooth`



- **MODIS:** Finding Missing Lat/Lon
 - Parsed through files, identified each tile's horizontal index (h) and vertical index (v)
 - Each h, v is a 1200 x 1200pixel grid, 1 pixel = 1 km
 - Multiply x and y by the index, then divide by Earth's radius
 - Obtain lat/lon and append to df

```
def extract_tile_indices(filename):
    """
    Extracts MODIS tile indices h, v from a granule_id or filename.
    """
    match = re.search(r'\.h(\d{2})v(\d{2})\. ', filename)
    if match:
        return int(match.group(1)), int(match.group(2))
    return None, None

def modis_tile_to_latlon(h, v, rows=1200, cols=1200, pixel_size=1000):
    """
    Generates latitude and longitude arrays for a given MODIS tile
    using the Sinusoidal projection.
    """
    EARTH_RADIUS = 6371007.181 # meters
    origin_x = -20015109.354 # Sinusoidal grid origin (upper-left)
    origin_y = 10007554.66

    tile_dim_m = rows * pixel_size
    x0 = origin_x + h * tile_dim_m
    y0 = origin_y - v * tile_dim_m

    x = x0 + (np.arange(cols) + 0.5) * pixel_size
    y = y0 - (np.arange(rows) + 0.5) * pixel_size

    xv, yv = np.meshgrid(x, y)
    lon = np.degrees(xv / EARTH_RADIUS)
    lat = np.degrees(np.arcsin(yv / EARTH_RADIUS))

    return lat, lon
```

- **HO=F Dataset**
 - Filtered the dates to match MODIS dataset
 - Extracted only 'date' and 'close' price for each day
- **Merging Issue**
 - MODIS -> Every 8 Days
 - HO=F -> Every day
 - **Fix:**
 - Assign/Group 'close' point date to nearest MODIS date
- **Finally, merged data into 'merged_df'**

ho_df

	date	Open	High	Low	Close	Adj Close	Volume
0	31-Dec-24	2.3105	2.3315	2.2892	2.3206	2.3206	54430
1	30-Dec-24	2.2405	2.3173	2.2397	2.2995	2.2995	19299
2	27-Dec-24	2.2065	2.2564	2.1990	2.2448	2.2448	17139
3	26-Dec-24	2.2230	2.2494	2.1969	2.2053	2.2053	17639
4	24-Dec-24	2.2387	2.2498	2.2169	2.2215	2.2215	15731
...
2510	8-Jan-15	1.6970	1.7195	1.6766	1.7110	1.7110	67237
2511	7-Jan-15	1.7020	1.7256	1.6715	1.6999	1.6999	75085
2512	6-Jan-15	1.7500	1.7629	1.7025	1.7262	1.7262	73014
2513	5-Jan-15	1.7955	1.7996	1.7386	1.7492	1.7492	66059
2514	2-Jan-15	1.8533	1.8774	1.7871	1.7957	1.7957	46068

modis_df

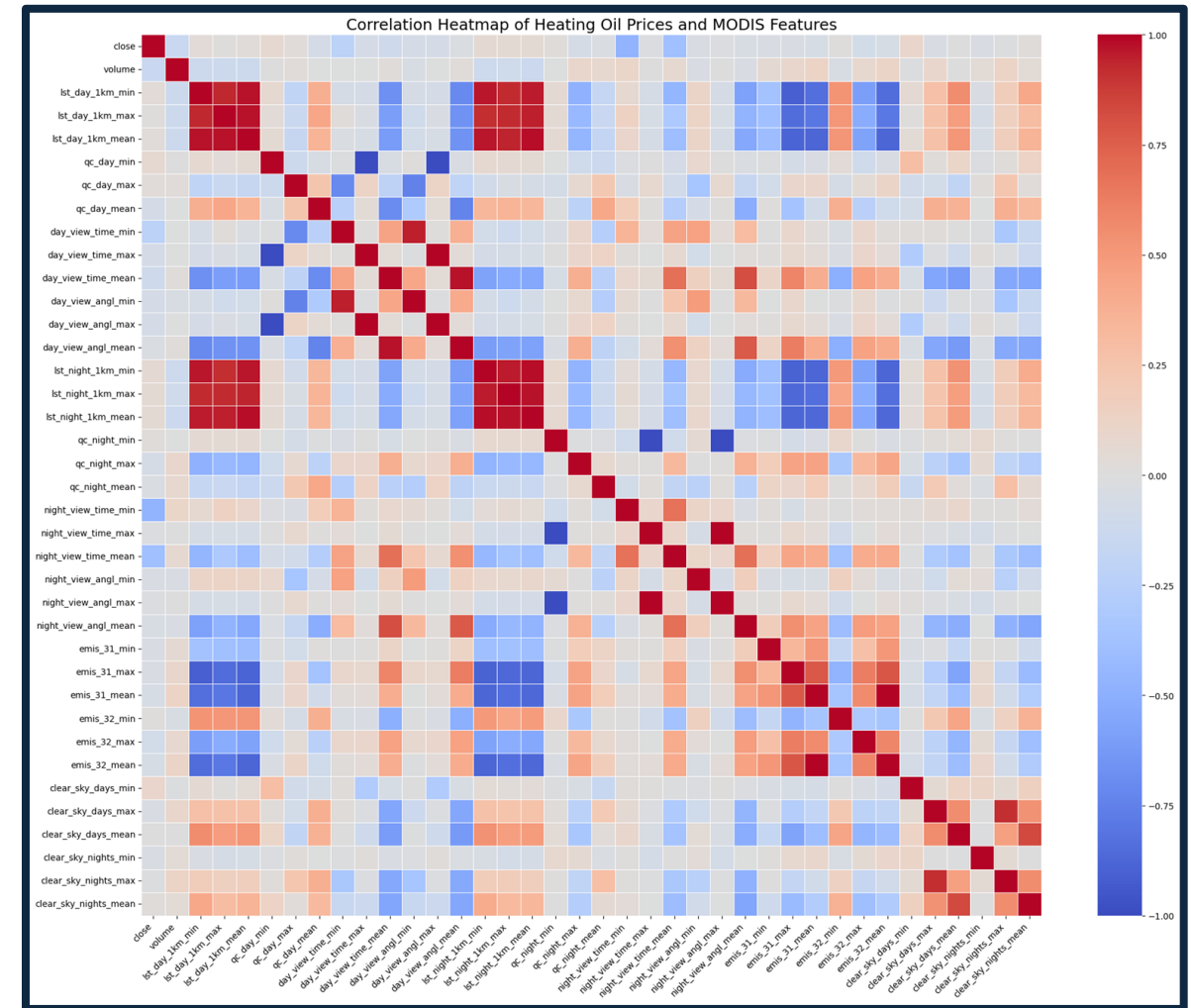
	date	granule_id
77	2024-12-26	MOD11A2.A2024361.h12v05.061.2025004043511.hdf
82	2024-12-18	MOD11A2.A2024353.h11v05.061.2024366031841.hdf
87	2024-12-10	MOD11A2.A2024345.h11v04.061.2024359030232.hdf
92	2024-12-02	MOD11A2.A2024337.h11v05.061.2024347151051.hdf
97	2024-11-24	MOD11A2.A2024329.h12v04.061.2024338044719.hdf


- Correlation Matrix (31 Features)

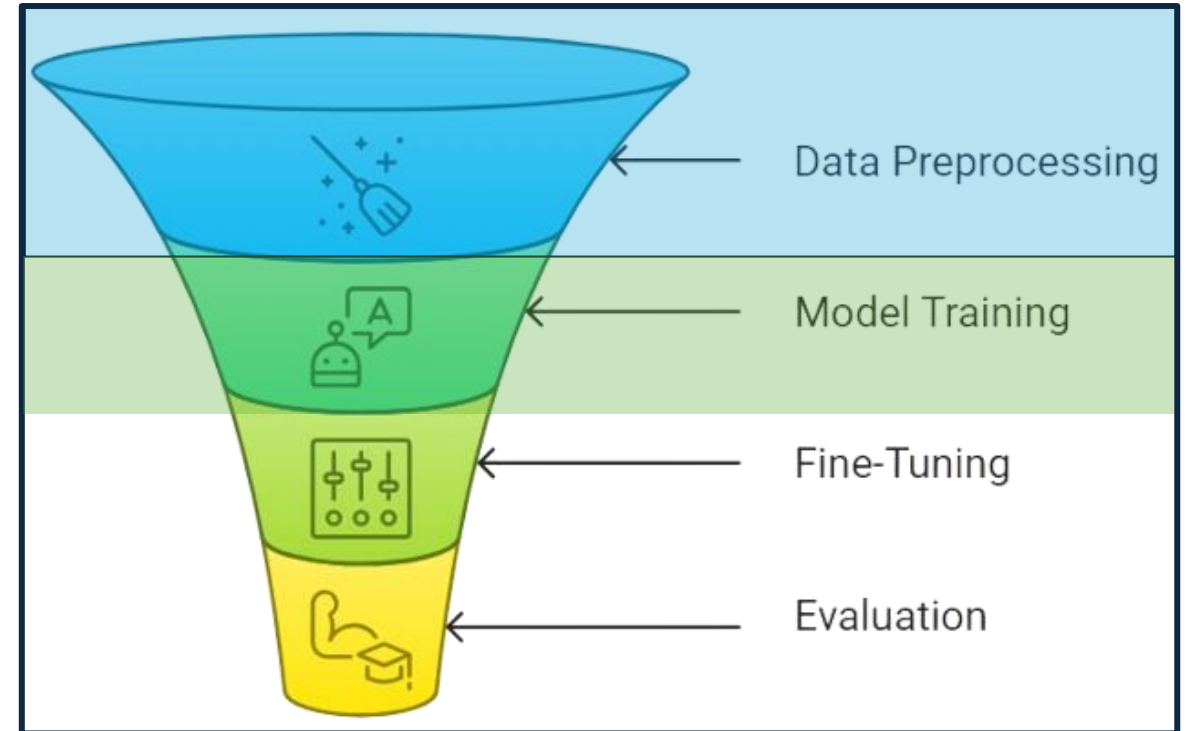
- **White:** 0.0
- **Red:** 1.0
- **Blue:** -1.0

- Extracted 9 Features (incl. y)

- 'date',
- 'close',
- 'volume',
- 'day_view_time_min',
- 'night_view_time_min',
- 'clear_sky_days_min_lag7',
- 'emis_32_max_lag7',
- 'lst_day_1km_mean_lag7',
- 'lst_day_1km_mean_roll7'

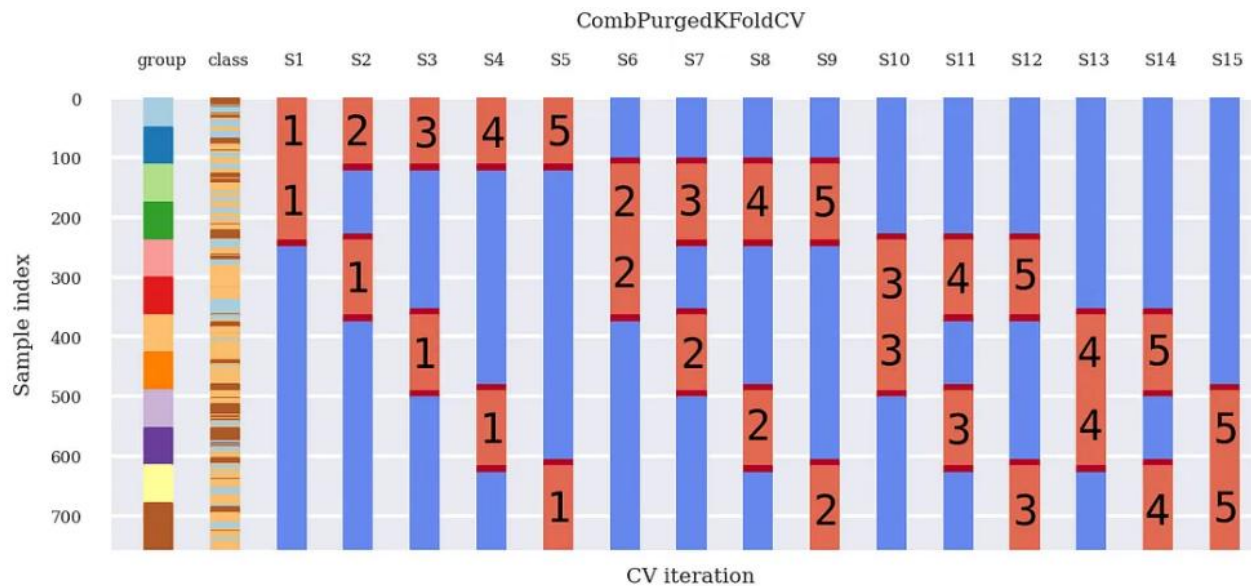


- **Correlation Matrix** 
- **PurgedKFold Validation**
- Elastic Net Regularization
 - Hyperparameter Tuning:
 - Alpha and L1 Ratio
- LightGBM (Gradient Boosting)
 - Hyperparameter Tuning:
 - `n_estimators`, `learning_rate`, `num_leaves`, `min_child_weight`, `min_child_samples`, `reg_lambda`, `reg_alpha`, `linear_tree`, `subsample`, `subsample_freq`, `colsample_bytree`, `colsample_bynode`, `linear_lambda`, `min_data_per_group`, `max_cat_threshold`, `cat_l2`, `cat_smooth`

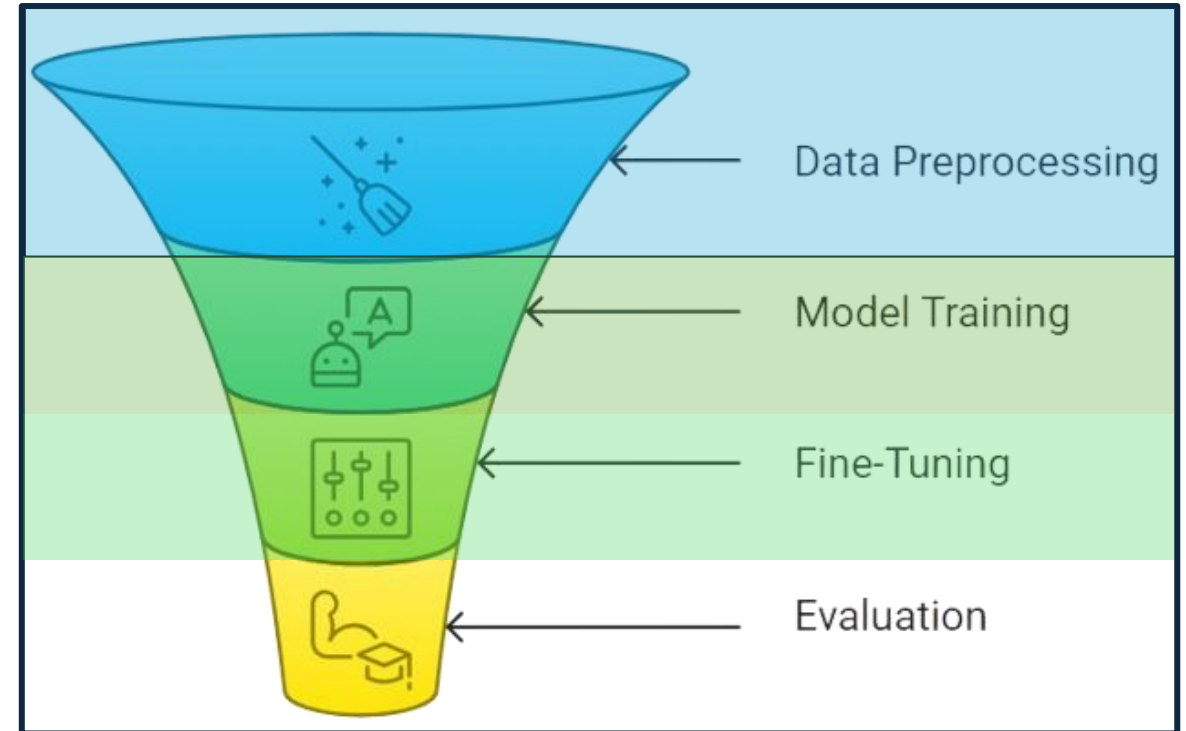


PurgedKFold Validation

- **Cross Val. Strat. (3-fold, 30-day gap)**
 - Prevents data leakage by leaving a buffer between training and test sets
 - Mimics real-world trading where future data is unavailable
- **Time-aware Splitting**
 - Ensures train and test sets are strictly ordered in time
 - Improves generalization for time-series prediction
- **Random Feature Dropout**
 - Randomly removes some input features during training
 - Prevents overfitting and forces the model to use diverse signals
- **Gaussian Noise Injection**
 - Adds small random fluctuations to features
 - Simulates real-world signal noise and improves model robustness



- **Correlation Matrix** ✓
- **PurgedKFold Validation** ✓
- **Elastic Net Regularization**
 - Hyperparameter Tuning:
 - Alpha and L1 Ratio
- **LightGBM (Gradient Boosting)**
 - Hyperparameter Tuning:
 - `n_estimators`, `learning_rate`, `num_leaves`, `min_child_weight`, `min_child_samples`, `reg_lambda`, `reg_alpha`, `linear_tree`, `subsample`, `subsample_freq`, `colsample_bytree`, `colsample_bynode`, `linear_lambda`, `min_data_per_group`, `max_cat_threshold`, `cat_l2`, `cat_smooth`



Why Elastic Net:

- Combines L1 (Lasso) and L2 (Ridge) penalties to handle multicollinearity and prevent overfitting.

Purpose in This Project:

- Regularizes noisy satellite-derived features (e.g., rolling temps, lagged values).

Cross-Validation Method:

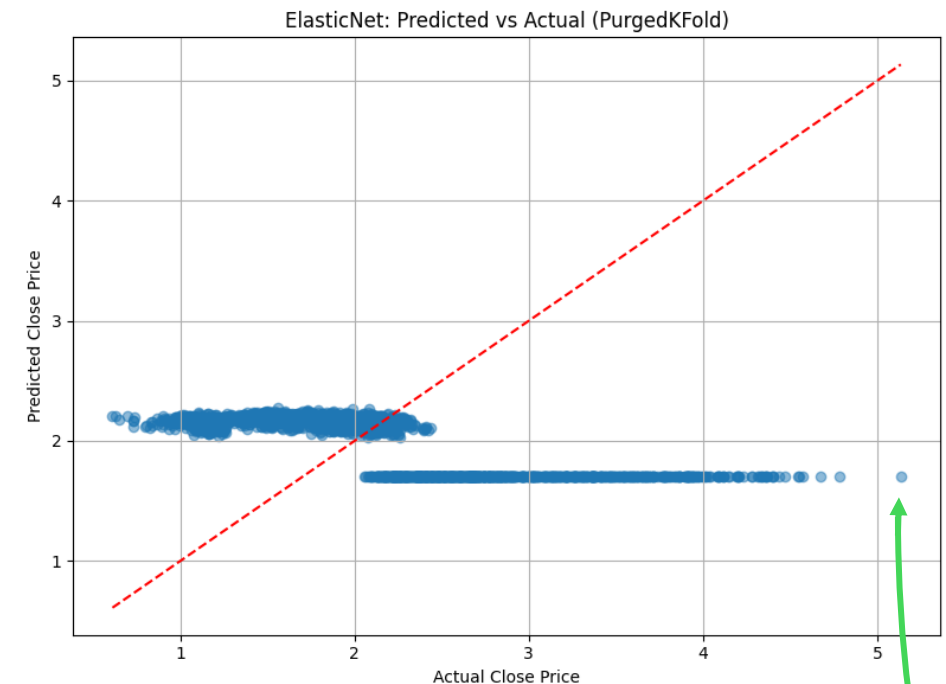
- Used **PurgedKfold** to avoid data leakage across time in model evaluation.

Model Behavior:

- Predictions generally stable but biased toward mean; struggles with extreme values.

Diagnostics Included:

- Predicted vs Actual plots for each fold



- **CV1 RMSE:** 0.6249
- **CV2 RMSE:** 0.5564
- **CV3 RMSE:** 1.323

Elastic Net adds robustness but may underreact to rare but impactful anomalies.

LightGBM (Light Gradient Boosting Machine)

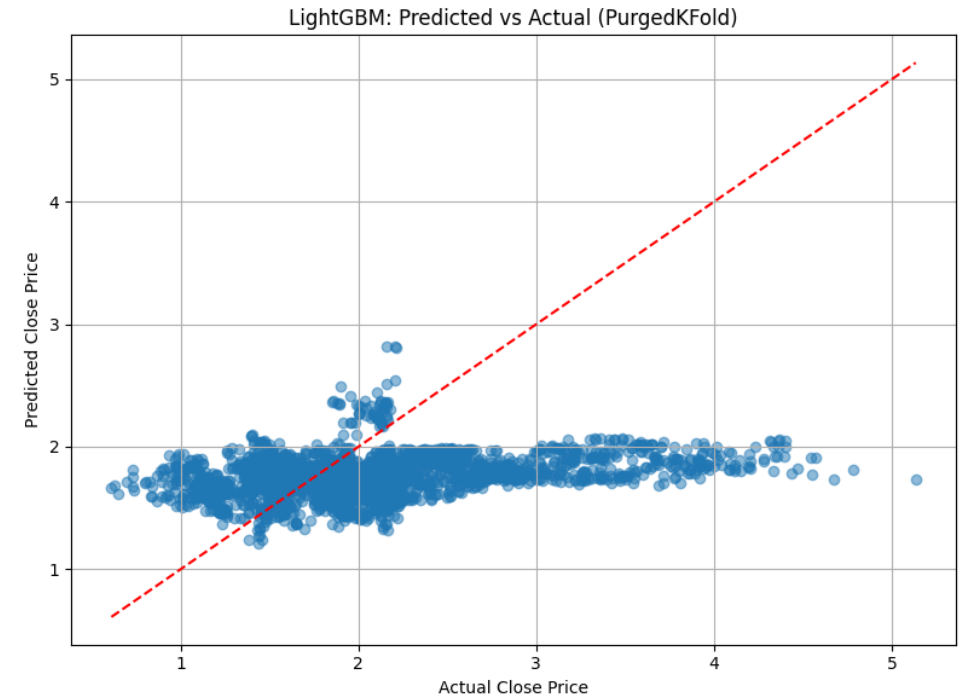
- A powerful, high-performance machine learning algorithm for regression and classification
- Randomized hyperparameter search

Gradient Boosting Framework

- Builds decision trees **sequentially**, each one improving on the previous
- Focuses learning on the **hard-to-predict** samples (boosting)

Why Use LightGBM in This Project?

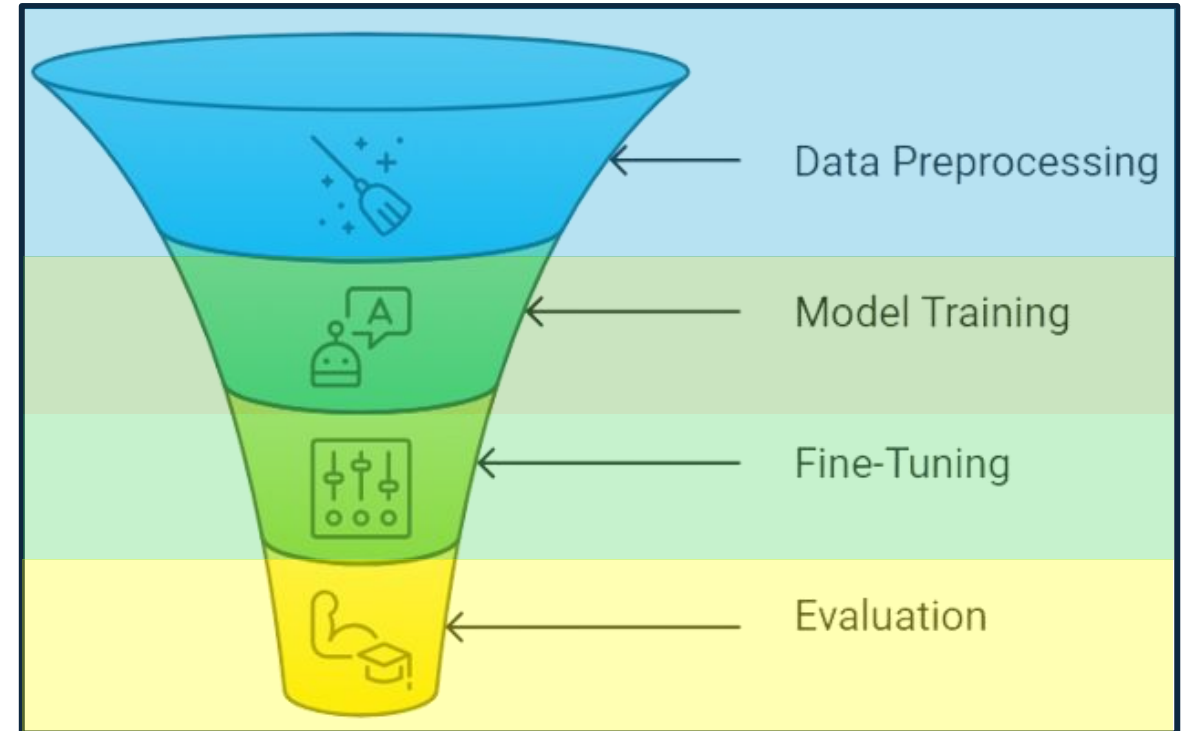
- Handles large datasets and **high-dimensional features** efficiently
- Automatically captures **nonlinear relationships** between variables
- Compatible with **PurgedKFold** and **custom noise injection**, making it ideal for time-series modeling



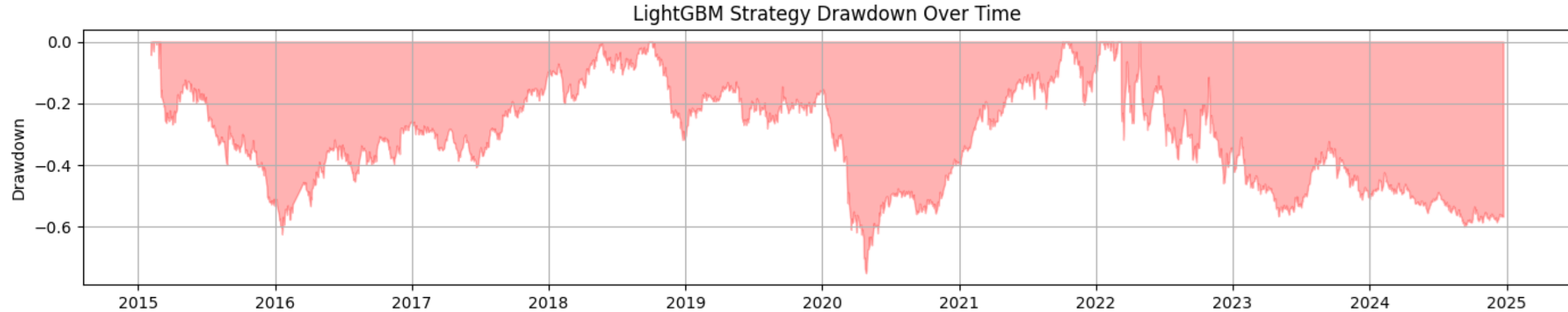
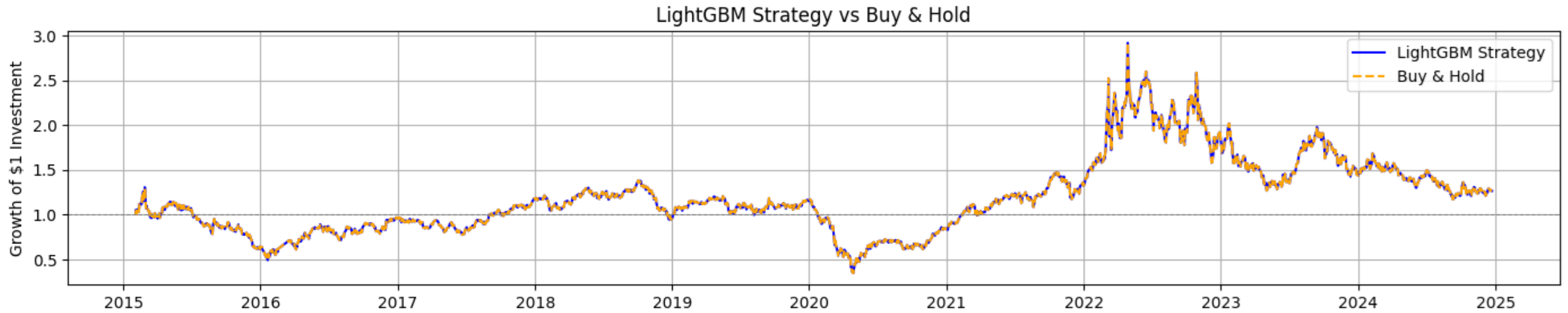
Interpretation

- Better than Elastic Net Regularization
- Slight underestimation in volatile/high-price regimes
- Can be fine-tuned further or blended with ElasticNet

- **Correlation Matrix** ✓
- **PurgedKfold Validation** ✓
- **Elastic Net Regularization** ✓
 - Hyperparameter Tuning:
 - Alpha and L1 Ratio
- **LightGBM (Gradient Boosting)** ✓
 - Hyperparameter Tuning:
 - `n_estimators`, `learning_rate`, `num_leaves`, `min_child_weight`, `min_child_samples`, `reg_lambda`, `reg_alpha`, `linear_tree`, `subsample`, `subsample_freq`, `colsample_bytree`, `colsample_bynode`, `linear_lambda`, `min_data_per_group`, `max_cat_threshold`, `cat_l2`, `cat_smooth`



Elastic Net Regularization & LightGBM Evaluation



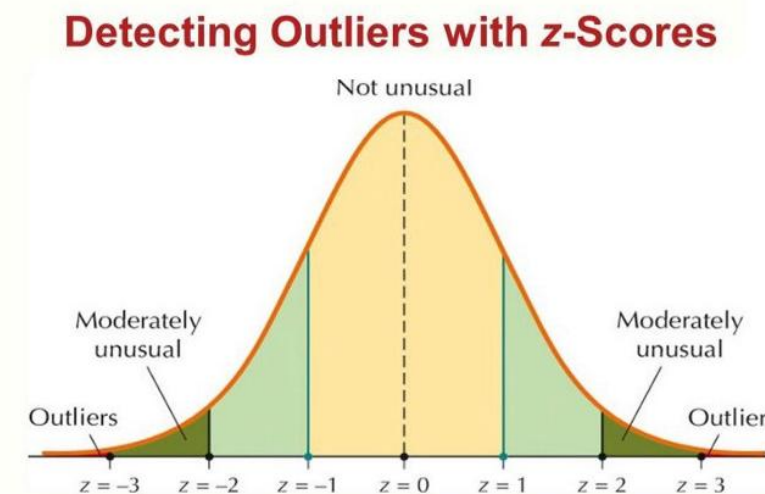
Sharpe Ratio: 0.27

Total Return: 30.84%

Annual Return: 2.78%

Z-Score Anomaly Testing

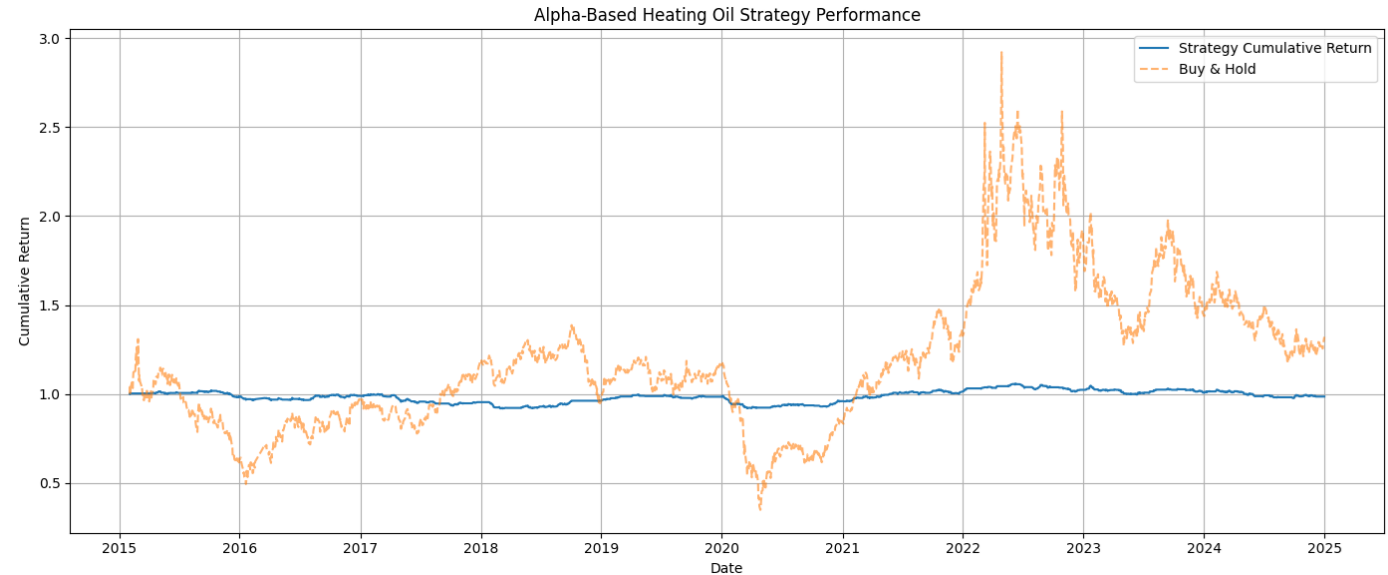
- **Purpose:** Detect days when satellite temperature data deviates significantly from seasonal norms.
- **Method:** Calculated seasonal average temperature per day of year
 - **Computed anomalies:** deviation from seasonal average
 - **Applied Z-score:** standardized the anomaly over a rolling window
- **Signal Logic:**
 - If $Z > \text{threshold}$ → signal overheat → short position
 - If $Z < \text{threshold}$ → signal cold anomaly → long position
- **Tuned Hyperparameters:**
 - Lag days, Z thresholds, rolling window size, and region weights



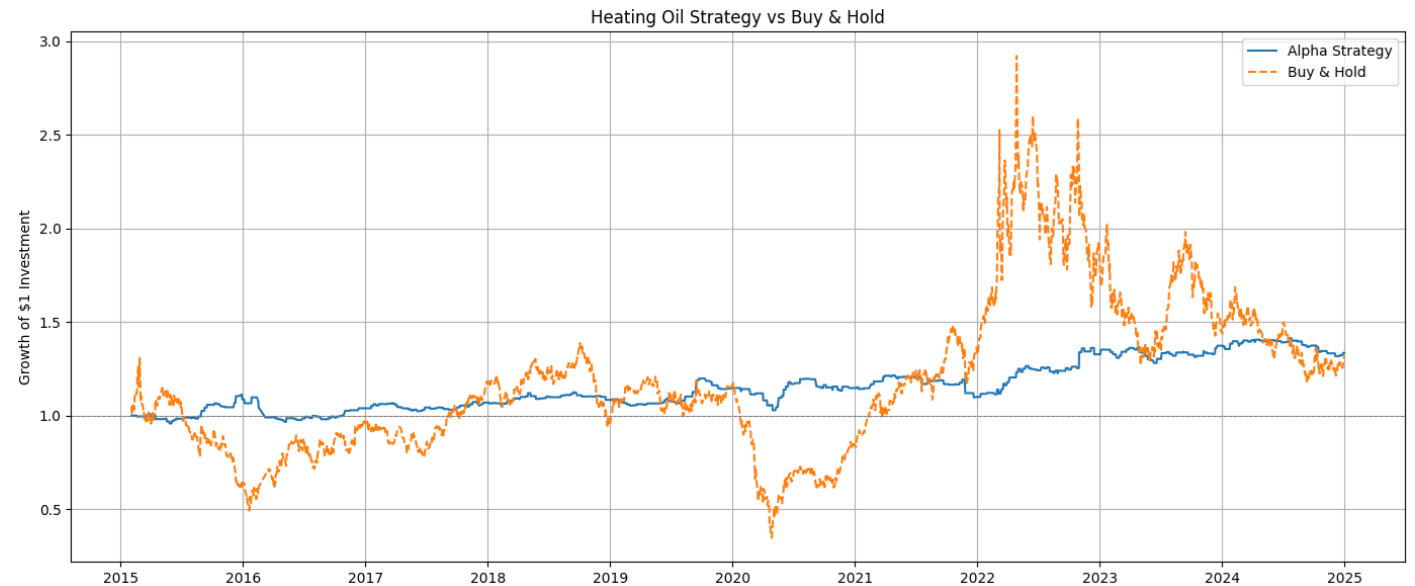
$$Z = \frac{\text{anomaly} - \text{rolling mean}}{\text{rolling std}}$$

Z-Score Anomaly Testing

- **Initial Strategy Found:**
 - Lag Days: 7
 - Z-Score Thresh: High = +1.5
Low = -1.5
 - Volatility Window: 10 days
 - Alpha Signal: Refinery = 0.7
Northeast = 0.3
- **Total Return: 33.50%**
- **Annual Return: 2.99%**
- **Sharpe Ratio: 0.43**



- **Tuned Hyperparameters!!**
- **Best Strategy Found:**
 - Lag Days: 5
 - Z-Score Thresh: High = 1.5
Low = -1.5
 - Volatility Window: 5 days
 - Alpha Signal: Refinery = 0.5
Northeast = 0.5
- **Total Return: 33.50%**
- **Annual Return: 2.99%**
- **Sharpe Ratio: 0.43**

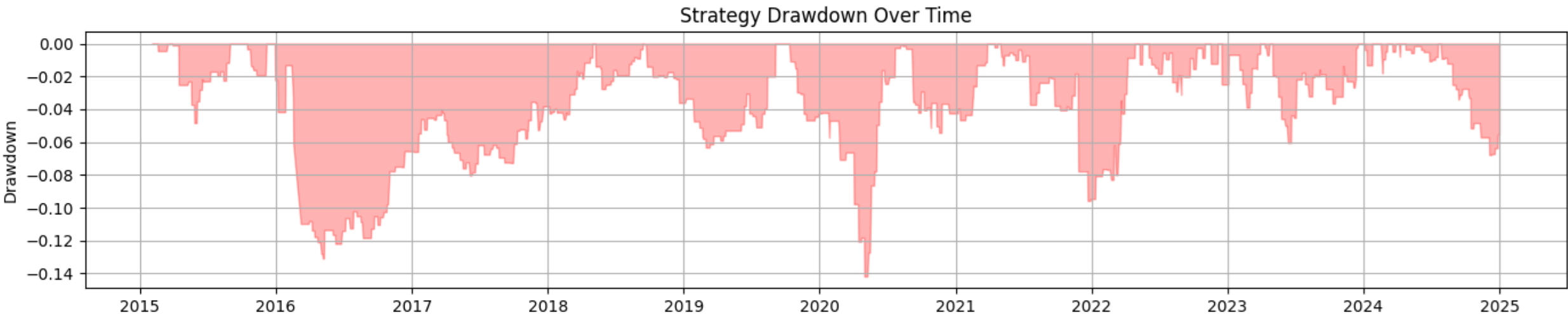
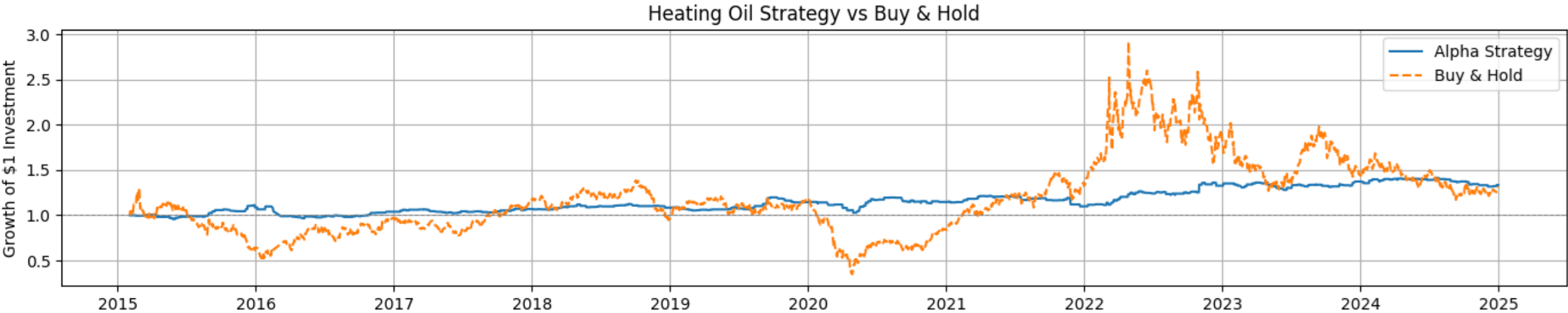


Z-Score Anomaly Testing Results

Consistent Growth: Alpha strategy steadily outperformed buy-and-hold over time.

Position Size: Adapts based on temperature anomalies and market volatility.

Lower Exposure to Spikes: Strategy avoids extreme market rallies and crashes.



Drawdowns indicate risk: They show how far the portfolio falls from its peak.

Moderate Drawdowns: Some sharp dips (up to -14%), but recovered steadily.

Alpha Strategy: Includes a stop-loss: halts trading when drawdown exceeds -10%.

Z-Test Walk-Forward

- **Best Train Window: 540**
- **Best Test Window: 60**
- **Total Return: 46.78%**
- **Annual Return: 5.17%**
- **Sharpe Ratio: 0.91**

LightGBM Walk-Forward

- **In-progress, not looking good**

- **Elastic Net Regularization (ENet)**
 - Sharpe Ratio: 0.27 | Total Return: 30.84% | Annual Return: 2.78%
 - Stable but conservative predictions; under reacts to rare extremes.
- **LightGBM (Gradient Boosting)**
 - Stronger than ENet; handles nonlinear relationships.
 - Sharpe Ratio: 0.27 | Total Return: 30.84% | Annual Return: 2.78%
 - Underperforms in walk-forward testing (in-progress).
- **Z-Score Anomaly Strategy**
 - Best Result: Sharpe Ratio: 0.91 | Total Return: 46.78% | Annual Return: 5.17%
 - Combines seasonal anomaly detection + volatility-aware weighting.
- **Drawdowns**
 - Strategy includes 10% stop-loss; protects against prolonged declines.
 - Maximum drawdown: ~14%; recovered over time.

- Refine LightGBM walk-forward retraining to improve robustness.
- Explore model blending (e.g., ENet + LightGBM + anomaly rule).
- Add alternative satellite features (e.g., snow cover, cloud masks).
- Automate retraining pipeline and real-time deployment prototype.
- Apply framework to other commodities (e.g., natural gas, electricity).



<https://github.com/spacelertser2004>

Thank You



Appendix