🔬 AI Cement Plant Optimization Platform: Model Documentation

Model 1.0: Clinker Free Lime Quality Predictor (KPI: `clinker_free_lime_pct`)

This model is engineered to predict the percentage of free lime in clinker, a critical quality determinant for the final product, enabling **early, prescriptive control of raw material blending**.

Detail	Specification
Target KPI	`clinker_free_lime_pct` (Percentage)
Process Stage	Pyroprocessing (Kiln)
Optimization Goal	Key quality control metric; stability and adherence to target limits.
Algorithm	Vertex AI AutoML Tabular Regression
Training Budget	1 hour on Vertex AI (Hackathon Minimal Cost)

I. Model Rationale and Data Context

Logic and Dependencies

The model captures the inherent **process delay** (residence time) between raw material input and finished clinker output. It is trained to predict the `clinker_free_lime_pct` using material and kiln conditions from approximately **5 hours earlier**. This constitutes a crucial cross-process optimization link between Blending and Pyroprocessing.

Training Data Specifications

Parameter	Value
Source	Synthetic data generated using logic in `data_generator.py`

Parameter	Value
Size (Rows)	50,000 rows
Feature Transformation	None (Raw feature values used)

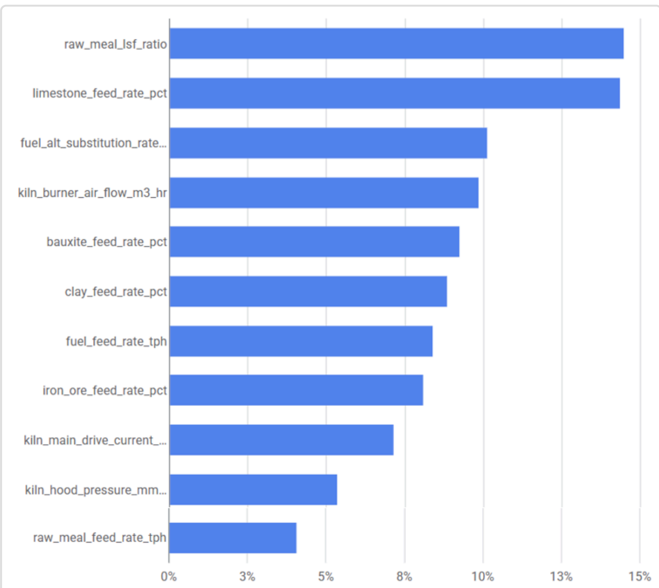
II. Input Features

The model uses 11 features, with a 5-hour lag applied to key material variables to account for process delay.

Feature Category	Features Used	Lag Time Applied
Blending / Raw Meal	`raw_meal_lsf_ratio`, `limestone_feed_rate_pct`, `clay_feed_rate_pct`, `iron_ore_feed_rate_pct`, `bauxite_feed_rate_pct`, `raw_meal_feed_rate_tph`	5 Hours
Kiln Operations	`fuel_feed_rate_tph`, `fuel_alt_substitution_rate_pct`, `kiln_hood_pressure_mmH2O`, `kiln_burner_air_flow_m3_hr`, `kiln_main_drive_current_amp`	Mixed / Non-lagged

Feature Importance (Model 1)

Recommendation: This visual provides the most immediate insight into the model's structure.



Feature Importance Graph for Model 1 (Clinker Free Lime)

Model 2.0: Kiln Specific Thermal Energy Predictor (KPI: `kiln_specific_thermal_energy_Kcal/kg_clinker`)

This model is designed to predict the energy efficiency of the kiln, which is the primary metric for **cost optimization and sustainability**.

Detail	Specification
Target KPI	`kiln_specific_thermal_energy_Kcal/kg_clinker`
Process Stage	Pyroprocessing (Kiln)
Optimization Goal	Predict energy consumption in real-time to enable cost minimization.
Algorithm	Vertex AI AutoML Tabular Regression
Training Budget	1 hour on Vertex AI (Hackathon Minimal Cost)

I. Model Rationale and Data Context

Logic and Dependencies

This model predicts the kiln's energy consumption based on the **real-time** fuel inputs and current kiln conditions. Since changes in fuel rate and kiln dynamics have a **near-instantaneous effect** on energy usage, this model primarily uses **non-lagged (real-time) variables**.

It provides immediate feedback on the energy impact of operational adjustments, crucial for energy optimization strategies.

Training Data Specifications

Parameter	Value
Source	Synthetic data generated using logic in `data_generator.py`
Size (Rows)	50,000 rows
Feature Transformation	None (Raw feature values used)

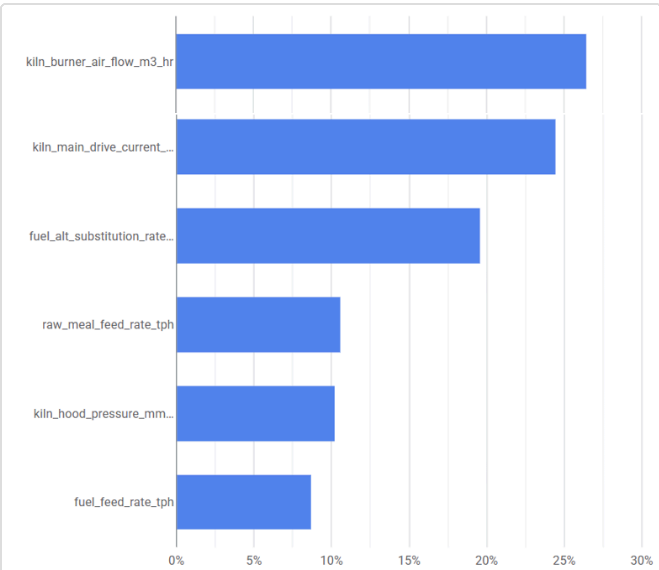
II. Input Features

The model uses 6 features directly linked to the kiln's instantaneous thermal state.

Feature Category	Features Used	Lag Applied
Kiln Operations	`raw_meal_feed_rate_tph`, `kiln_hood_pressure_mmH2O`, `kiln_main_drive_current_amp`	Real-Time (Non-lagged)
Fuel Inputs	`fuel_feed_rate_tph`, `fuel_alt_substitution_rate_pct`, `kiln_burner_air_flow_m3/hr`	Real-Time (Non-lagged)

Feature Importance (Model 2)

Recommendation: Include the feature importance graph to show which real-time variables are most predictive of thermal energy usage.



Feature Importance Graph for Model 2 (Thermal Energy)

III. Production Scaling and Next Steps

> These models were developed and trained using a minimal **1-hour budget** on Vertex AI AutoML to control costs during the hackathon phase. While this demonstrates the

> platform's architectural capability, the resulting predictive accuracy may not be ideal for a high-stakes production environment.

For deployment in a production-level solution, we recommend the following strategic enhancements to achieve best-in-class performance:

- **Increase Training Budget:** Significantly increase the AutoML training budget (e.g., to **6-8 hours** per model). This allows the algorithm to explore more complex deep learning architectures and feature interactions, maximizing predictive accuracy.
- **Explore Feature Engineering:** Conduct in-depth feature engineering and transformation techniques (e.g., standardization, polynomial features, or specialized time-series techniques) to improve the signal-to-noise ratio in the input data.
- **Validate on Real Data:** Once integrated, the models must be validated and periodically retrained using live, production-level plant data to ensure long-term stability and performance.