# Initial Setting Time Prediction for Calcium Aluminatum Cement

## 1. Introduction

In this project, we aim to improve the representation of cement manufacturing data, specifically for predicting the initial setting time of calcium aluminatum cement—a key property that affects the usability and structural performance of the final product. Accurate prediction of this setting time is crucial in industrial applications, as it determines how quickly the material can be handled after mixing.

To address this, we explore whether reducing feature redundancy and dimensionality through Principal Component Analysis (PCA) and Autoencoders (AE) leads to better model generalization and efficiency. The data comprises two versions: a raw dataset with ~60 original features, and an enhanced version with ~500 engineered features, including transformations and ratios that introduce multicollinearity and noise.

Our objective is to assess whether a more compact and structured representation of the input space improves predictive performance. We hypothesize that reducing irrelevant or redundant information will enhance the learning dynamics of a Multi-Layer Perceptron (MLP) tasked with predicting the setting time. We adopt a two-fold modeling strategy: starting from a simple, baseline MLP—similar to what could have been implemented in Lab 1—and progressively building a more complex architecture to observe potential improvements.

Through systematic comparison, we aim to quantify the trade-offs between linear (PCA) and nonlinear (AE) compression techniques, and evaluate whether the added complexity of AE is justified. Our work not only benchmarks feature compression techniques in this industrial setting, but also provides insight into effective model design and feature engineering for high-dimensional datasets common in scientific and manufacturing domains.

## 2. State of the art

Machine learning has become a common tool for predicting cement setting time, with models like Random Forests, Gradient Boosting, and Support Vector Regression widely used due to their strong performance on structured industrial data [1,2]. More recent approaches have combined multiple models in hybrid ensembles, showing improvements in generalization but often lacking domain-specific adaptation [3].

A recurring limitation is the naive handling of compositional data (e.g., chemical oxides or mineral phases), which violate assumptions of standard ML models. Compositional Data Analysis (CoDA), particularly ILR transformations, offers a principled way to embed such features in Euclidean space, yet remains largely absent from existing applications [4,5].

Furthermore, class imbalance across cement types is often addressed using generic oversampling techniques like SMOTE, which can produce physically unrealistic samples when applied without domain constraints [6]. Few studies incorporate augmentation strategies that preserve material and process coherence.

Overall, while prior work has established the value of machine learning in cement modeling, gaps remain in handling compositional structure, ensuring physical plausibility, and managing heterogeneity across types. Our methodology addresses these issues with domain-aware feature engineering, data balancing, and deep learning.

## 3. Methodology

### 3.1 Data Analysis

#### 3.1.1 Data Context and Privacy Constraints

The dataset used in this project originates from Cementos Molins, where one of the team members is currently completing an internship. Due to the proprietary nature of the data, certain restrictions apply: feature names have been anonymized and units are intentionally omitted from plots to protect industrial confidentiality. Importantly, scaling and anonymization were applied near the end of the preprocessing pipeline, after all feature engineering and data transformations had been performed. This allowed us to preserve original feature relationships and perform valid engineering steps, such as ratio calculations, aggregations, and transformations, on the raw data. As a result, while visualizations may appear stripped of labels or units, all preprocessing choices—such as outlier removal, redundancy checks, and encoding strategies—were based on the actual unscaled feature distributions. This balance ensured the technical rigor of our analysis while respecting data privacy constraints.

#### 3.1.2 Feature Inventory and Initial Filtering

The original dataset consisted of 76 features drawn from real production data provided by Cementos Molins. These included chemical compositions, phase measurements, physical properties (like Blaine fineness), and operational metadata. However, due to their operational origin, features varied in reliability, completeness, and modeling relevance.

Our goal in this stage was to perform a rigorous filtering process to retain only informative, clean, and non-leaking features. Particular attention was given to key domains such as **chemical composition (Chem_1 to Chem_11)**, **mineral phases (Phase_1 to Phase_13)**, and **fineness metrics (e.g., Blaine, d50)**—all of which are known to influence cement behavior.

Approximately 25 features were removed due to one of the following reasons:

- **High missingness:** features like Customer, Fe(ppm), or early setting times (e.g., init.6) were dropped due to excessive NaNs.
- **Data leakage:** variables such as final setting time (end) or strength after 6 hours were excluded as they would not be available at prediction time.
- **High noise or redundancy:** some engineered ratios or summations (e.g., Chem_Sum) showed limited utility or were derivable later.
- **Transformation into engineered features:** L.o.i. and L.o.i. calc. were merged into a single, cleaned "Ignition Loss" variable to represent volatile content more reliably.

After this filtering, the dataset was reduced from **76 to 48 features**, and the number of usable samples narrowed from **8321 to 6115** by retaining only the most relevant cement types. This cleaned version forms the foundation for all subsequent modeling efforts.

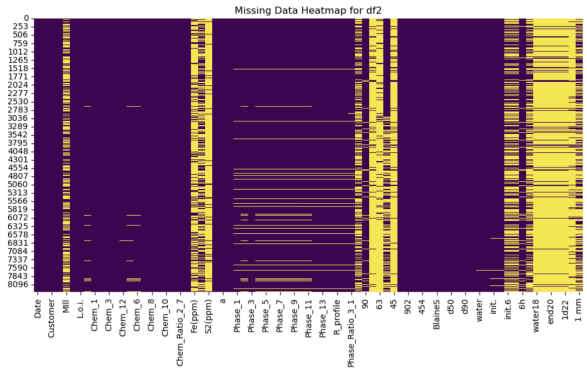


*Figure 1: Missing Data HeatMap before Initial Filtering*

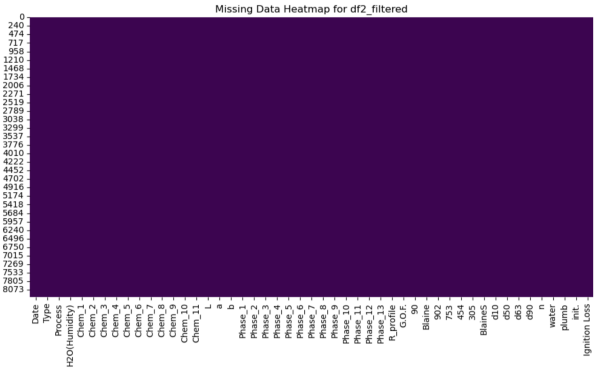


*Figure 2: Missing Data HeatMap after Initial Filtering*

### 3.1.3 Dataset Preparation and Experimental Setup

To systematically assess the impact of feature engineering on model performance, we structured our workflow around two parallel dataset versions: one preserving the cleaned raw features, and another serving as a foundation for enhancement through domain-informed transformations and derived variables.

**Two Dataset Versions**

- **Raw Dataset**: This version consists of the 48 features retained after a thorough cleaning of the original dataset. No synthetic or engineered features are added. It represents a minimal-preprocessing scenario and serves as the baseline reference. One important limitation of this version is the presence of class imbalance across the different cement types, which is not corrected here in order to preserve the data in its natural form.
- **Enhanced Dataset**: A deep copy of the raw dataset is created to support advanced feature engineering and resampling strategies. In this version, we apply transformations such as ratios, statistical aggregates, and domain-informed combinations of features. Additionally, we address class imbalance—specifically across cement types—through either data augmentation or sample reduction, with the goal of obtaining a more uniform representation that supports better generalization during training.

**Data Splitting Strategy**

To ensure consistent training and evaluation across both dataset versions, we applied the following splitting procedure:

1. **Train-Test Split (80-20%)**: The full dataset is initially divided into a training set (80%) and a test set (20%). The test set remains completely untouched throughout model development and is used only for final performance evaluation.
2. **Train-Validation Split (85-15% of training data)**: The training set is further partitioned into training (85%) and validation (15%) subsets.

All splits are **stratified according to cement type**, in order to preserve class distribution across subsets and ensure representative sampling. Stratification is **not** applied based on the cement process, as this variable is not considered a reliable stratification target in this context.

Each split is also splitted by cement type for all subsequent modeling efforts.

#### 3.1.4 Transformations for Feature Enhancement

Once the raw dataset was cleaned and split into training, validation, and test sets, we began preparing the enhanced version of the data by applying domain-aware transformations that aim to improve model compatibility and performance.

**Transforming Compositional Data to Euclidean Space**

A key characteristic of the original dataset is the presence of compositional features—variables that represent proportions and always sum to 100%. This applies particularly to two groups: chemical composition (Chem_1 to Chem_11) and phase composition (Phase_1 to Phase_13). These types of features lie in a constrained geometric space called a *simplex*, which violates the assumptions of most standard machine learning models (including MLPs and Autoencoders), as they operate in standard Euclidean space.

To address this, we applied a transformation technique that maps these compositional features into an unconstrained Euclidean space while preserving their relative relationships. This ensures that the model can interpret the information in these features effectively without being biased by the compositional constraints. Specifically, we used the Isometric Log-Ratio (ILR) transformation, a method commonly applied in compositional data analysis. The ILR transform operates by projecting the components of a composition—once normalized to sum to one—into a lower-dimensional Euclidean space using orthonormal log-ratio coordinates. This enables the model to access and learn from the structure of the original proportions without being misled by their inherent sum constraint.

#### 3.1.5 Outlier Identification and Filtering Strategy

Following the transformation of compositional features, the next step in our enhanced dataset preparation focused on identifying potential outliers. While outlier removal is a common preprocessing step, our analysis revealed that the nature of the data required a more cautious and tailored approach.

Initial statistical assessments showed that the majority of our feature distributions were highly non-normal. Many exhibited skewed shapes, clustered peaks, or multimodal behavior. This made standard univariate outlier detection methods, such as interquartile range or percentile filtering, inappropriate. In particular, such methods risk removing legitimate values that fall in the tails of valid distributions or retaining suspicious values in the valleys between distribution clusters—especially in mixtures of distributions. Likewise, multivariate techniques that assume normality, such as Mahalanobis distance, were also unsuitable due to the lack of multivariate normality in the data.

To address this, we implemented a custom, non-parametric approach that performed robust density-based univariate detection and a separate multivariate anomaly detection using Isolation Forest. Instead of immediately filtering samples, we generated masks identifying potentially anomalous data points under both univariate and multivariate criteria.

This two-stage flagging system allowed us to evaluate the impact of different outlier handling strategies. Specifically, we defined two logical removal criteria:

- **OR Logic**: removing any sample flagged as an outlier by either the univariate or multivariate method
- **AND Logic**: removing only those samples flagged by both methods simultaneously

In practice, these strategies were compared across cement types. We analyzed how each affected distributional properties like skewness, kurtosis, and normality (via statistical tests such as Shapiro-Wilk and Mahalanobis-based KS testing). This allowed us to quantify whether removing flagged data led to statistically healthier training sets via scoring.

Rather than applying a one-size-fits-all rule, we selected the most appropriate filtering strategy for each cement type based on this analysis. For example, cement types like ISTRA 50 benefited from the stricter AND filter, while others like CEMFAST showed no statistical gain from any filtering and were kept as-is.

```
📦 Saved AND for ISTRA 50
📦 Saved OR for ISTRA 45
📦 Saved OR for ISTRA 50 5.0
📦 Saved OR for ISTRA 40
📦 Saved OR for Lumnite SG
📦 Saved RAW for CEMFAST
📦 Saved OR for Lumnite SG5
```

| | Cement Type | Best | Score AND | Score OR | Score RAW |
|---|---|---|---|---|---|
| 0 | ISTRA 50 | AND | 0.500000 | 0.466667 | 0.0 |
| 1 | ISTRA 45 | OR | 0.200000 | 0.666667 | 0.0 |
| 2 | ISTRA 50 5.0 | OR | 0.433333 | 0.666667 | 0.0 |
| 3 | ISTRA 40 | OR | -0.100000 | 0.533333 | 0.0 |
| 4 | Lumnite SG | OR | 0.333333 | 0.666667 | 0.0 |
| 5 | CEMFAST | RAW | -0.066667 | -0.200000 | 0.0 |
| 6 | Lumnite SG5 | OR | 0.333333 | 0.400000 | 0.0 |

*Figure 3: Cement Type Final Selection*

This cautious approach to outlier handling ensured that we avoided overcleaning the data or introducing bias, preserving data integrity while improving robustness for downstream modeling.

### 3.1.6 Data Augmentation and Data Reduction

After addressing class imbalance and outlier filtering, we performed data augmentation and reduction to ensure robust and balanced modeling datasets for each cement type. Our primary goals were twofold: (1) augment underrepresented cement types up to a target of 1000 high-quality samples, and (2) reduce overrepresented cement types to this same sample size, while preserving their original distribution characteristics.

**Data Augmentation Strategy**

For cement types with fewer than 1000 training samples, we used a custom **nearest neighbors interpolation** method with **multivariate plausibility constraints** to generate synthetic samples. The process involved:

- **Interpolation via KNN**: For each synthetic sample, we randomly selected a real sample and identified its *k* nearest neighbors in feature space. We then interpolated a new point using a Dirichlet-weighted average of these neighbors. The interpolation was biased toward the base sample to preserve local structure.
- **Inverse ILR Transformation**: We reverted interpolated ILR coordinates for chemical and phase compositions to their original compositional form using a custom ilr_inv function, ensuring valid compositional totals.
- **Multivariate Plausibility Filtering**: Each synthetic sample was validated using **robust Mahalanobis distance** (via MinCovDet) across multiple feature subspaces.
- Only samples that fell within the 99.5% confidence ellipsoid for **all** subspaces were retained.
- **Process Assignment**: A process label was assigned to each synthetic sample using minimum Mahalanobis distance to real samples in each process cluster, with a probabilistic fallback if ambiguity existed.
- **Final Check**: Only samples that met all plausibility constraints and preserved overall feature range bounds were included. The Augmented flag was set to 1 for synthetic rows.

UMAP visualizations confirmed that synthetic samples blended well with the original distribution in feature space. Additionally, the process label distributions remained stable post-augmentation (with minimal drift), as confirmed by comparative histograms and summary tables.

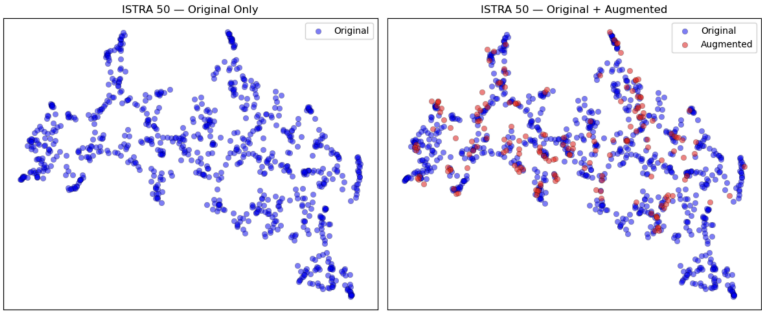


*Figure 4: ISTRA 50 UMAP comparison. ( Original: 823 | Augmented: 177 )*

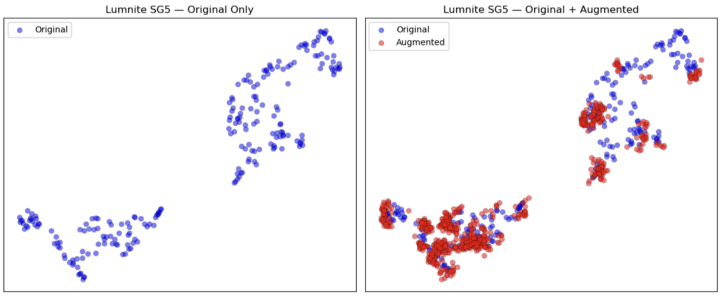


*Figure 5: LUMNITE SG UMAP comparison. ( Original: 223 | Augmented: 777 )*

As can be seen, once the increased number of samples begins to approach and exceed 50%, the increased distribution becomes increasingly "dangerous" with respect to the original, resulting in the possibility of being less reliable.

**Data Reduction Strategy**

For cement types exceeding 1000 samples, we applied **balanced trimming**, leveraging clustering and statistical distance to preserve distributional integrity:

1. **Clustering and Joint Distribution Modeling**:
    - Applied **KMeans clustering** on UMAP-reduced feature space (after selecting robust numeric variables).
    - Modeled the **joint distribution of cluster vs. process** as a categorical 2D frequency table.
    - Derived the target sample count for each (cluster, process) combination based on the observed proportion.
2. **Random Stratified Trimming**:
    - For each (cluster, process) group, we sampled with or without replacement to match the target count.
    - Ensured balanced and proportional representation across both cluster and process dimensions.
3. **Mahalanobis-Based Trimming** (Alternative):
    - Instead of random sampling, selected the n most central samples per (cluster, process) group using Mahalanobis distance to the local MCD-estimated cluster center.
    - This method prioritized the most “typical” points from each group, potentially reducing outlier influence further.
4. **Final Comparison**:
    - For **ISTRA 40**, we evaluated both trimming strategies against the original dataset using:
        - **Univariate Wasserstein distances** (distributional shift per feature)
        - **Multivariate Mahalanobis distances** (global distributional shift)
    - Both approaches yielded acceptable results, with the **random-stratified** version achieving a slightly better univariate fit and lower multivariate deviation overall.

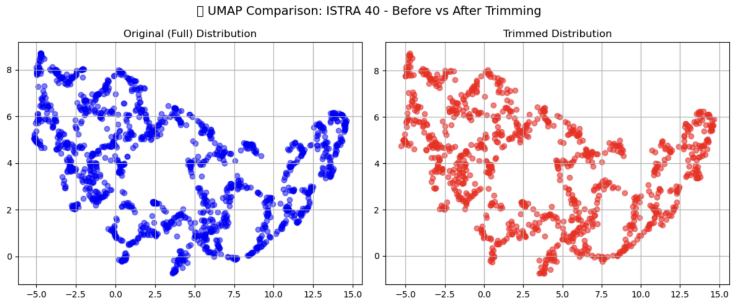


*Figure 6: Original vs. Random Stratified Distributions ( Original: 1451 | Reduced: 1000 )*

5. **Final Size Guarantee**:
    - If trimming overshot the 1000-target due to resampling, we further reduced the dataset by removing samples that were most similar (i.e., minimum pairwise distance) to each other until exactly 1000 samples were retained.

**3.1.7 One-Hot Encoding, Feature Engineering, and Scaling**

**One-Hot Encoding:**
We first applied one-hot encoding to the categorical features *Type* and *Process*, which allows the models to differentiate between different cement categories and manufacturing lines without assuming any ordinal relationship.

**Feature Engineering:**
For the enhanced dataset, we introduced a wide range of domain-informed features. These include chemical-to-chemical, phase-to-phase, and cross chemical–phase ratios, as well as aggregate totals and derived metrics (e.g., whiteness, $H_2O$-to-Ignition Loss ratio). These additions aimed to enrich the feature space with interpretable interactions relevant to cement composition and performance.

**Scaling:**
All continuous numerical features (excluding the one-hot encoded variables) were scaled using a **Quantile Transformer** to handle skewed distributions. The target variable (*initial setting time*) was scaled separately using **MinMax scaling** to normalize its range.

**Final Overview:**
After processing, the **raw training dataset** consisted of **4156 samples and 58 features**, while the **enhanced dataset** grew to **7000 samples and 474 features**, reflecting the effect of both feature augmentation and synthetic sample generation.

### 3.2 Model and Optimization

#### 3.2.1 Autoencoder

To reduce dimensionality and noise while attempting to preserve non-linear relationships that PCA may overlook, we implemented an Autoencoder compression strategy. The main objective was to compress the high-dimensional dataset we had into more informative and lower-dimensional features by also preserving the key patterns that contribute to the prediction of the initial setting time. Furthermore, we wanted to know how this tool helped or not both of the MLPs.

In addition to that we performed Principal Component Analysis (PCA) to both datasets, this was to compare the results and really value if the work the autoencoder is doing can be done with PCA which has less computational cost but only reduces dimensions with linearly dependent variables.

Our Autoencoder is a fully connected, symmetric architecture applied directly to the entire dataset except for the categorical one-hot encoded features, which are excluded. We chose an encoding approach where the entire numerical feature space was compressed into a latent representation. This allowed us to evaluate whether the autoencoder could capture nonlinear patterns from a high dimensional feature space.

Architecture Overview

- Encoder: Linear → ReLU → Linear → Bottleneck
- Decoder: Bottleneck → Linear → ReLU → Linear → Output

The latent dimension is chosen based on input dimensionality e.g., 32 for less than 64 inputs, 64 for less than 128, up to “input dimension // 2 for very large inputs. In addition, a single layer is added if dimensionality is large.

The training and optimization strategy is the Mean Squared Error (MSE) as the loss function, the Adam Optimizer with lr = 0.001, the Scheduler is ReduceLROnPlateau with factor 0.75, and an early stopping of 50 epochs of patience.

The training strategy combines learning rate adaptation and early stopping to avoid overfitting and excessive computation. The learning rate scheduler reduces the step size when loss plateaus, and early stopping halts training once no further progress is observed.

#### 3.2.2 MLP

As mentioned we implemented two Multi-Layer Perceptrons (MLPs) and the goal was to determine whether an improved and more complex representation helps MLPs learn more efficiently and generalize better.

The simpler MLP uses a shallow architecture and no regularization techniques whilst the more advanced MLP deepens the architecture and includes dropout, weight decay, adaptive learning rate scheduling, and early stopping. Both models are trained under the same data conditions and evaluated using the same metrics, ensuring fair comparisons. This strategy not only quantifies the benefit of architectural complexity but also reveals how sensitive each model is to feature engineering and dimensionality reduction.

**Baseline MLP**

The baseline MLP mimics a conventional feedforward neural network that could have been implemented early in the course. It consists of a three-layer architecture with fixed units and ReLU activation:

Architecture Overview: Input → Linear (64) → ReLU → Linear (32) → ReLU → Linear (1) → Output

This model was used across all datasets (raw, PCA, AE) to measure how much performance gain could be attributed solely to better representations, independent of architectural complexity. Training was performed using Mean Squared Error (MSE) as the loss function, with the Adam optimizer and a learning rate of 0.001. A fixed number of 500 epochs was used, with performance metrics logged every 50 steps. No learning rate scheduler or early stopping was applied, as the goal was to maintain a consistent training process across datasets.

**Complex MLP**

The complex MLP was designed to scale with the size of the input. The number of layers and neurons were dynamically selected according to the number of features, based on predefined thresholds. The design includes deeper architectures and regularization mechanisms to enhance generalization, especially on the high-dimensional enhanced dataset.

Architecture Overview: Input → [Linear → GELU → Dropout] × N → Linear → Output

Where N is the number of hidden layers determined by input dimensionality. For example, for 500+ features, the architecture becomes (1024, 512, 128, 64), while for lower-dimensional inputs, shallower networks are used. All layers use the GELU activation function, which has shown improved performance in deep architectures due to its smooth nonlinearity. A dropout layer ($p = 0.1$) is added after each activation to reduce overfitting.

The training strategy for this model is more sophisticated. We again use MSE as the loss, but include weight decay (L2 regularization) with a factor of $1 \times 10^{-4}$. The optimizer is Adam, and a ReduceLROnPlateau scheduler reduces the learning rate by a factor of 0.8 after 5 epochs without improvement in validation error. Additionally, early stopping with a patience of 100 epochs is used to terminate training once no further gain is observed.

**4. Experiments**

We designed a structure of 16 experiments in order to evaluate the impact of dimensionality reduction on MLP performance. These experiments follow a 2 x 2 x 2 x 2 setup:

- 2 Datasets: Raw and Enhanced
- With/without PCA
- With/without Autoencoder
- 2 MLP models: Simple and Improved

For PCA we grouped correlated features with threshold = 0.9, and replaced each group with principal components capturing 95% of variance. This block-wise PCA reduces redundancy while also maintaining interpretability.

Autoencoder compression was applied globally across all numerical features, excluding one-hot encoding variables. This global approach allowed us to evaluate the model's ability to learn meaningful compressed representations across the full feature space, particularly in the high dimensional enhanced dataset. After training, no filtering step was applied, all Autoencoder-transformed features were retained regardless of reconstruction performance.

The experimental design allows us to compare multiple factors. First, we assess the difference between the raw and enhanced datasets to understand whether richer, engineered features provide better learning signals. Second, we evaluate linear PCA against nonlinear autoencoder compression to determine whether nonlinear transformations are more effective at reducing redundancy and capturing structure. Finally, we compare a baseline MLP against an improved, deeper version to see whether more expressive architectures can benefit from the compressed representations. Altogether, these comparisons aim to measure the impact of preprocessing techniques on model performance, especially in terms of reconstruction quality and downstream prediction.

**5. Results**

**Complex MLP Summary table**

For all datasets 1000 epochs were used in training.

| Dataset | Train_R2 | Val_R2 | Test_R2 | Train_RRMSE | Val_RRMSE | Test_RRMSE | Train_MSE | Val_MSE | Test_MSE |
|---|---|---|---|---|---|---|---|---|---|
| enhanced_pca_ae | 0.957663 | 0.925775 | 0.888618 | 0.103495 | 0.134214 | 0.167043 | 0.001077 | 0.001785 | 0.002740 |
| raw_ae | 0.925859 | 0.883485 | 0.886811 | 0.133420 | 0.165441 | 0.168392 | 0.001701 | 0.002560 | 0.002784 |
| enhanced_ae | 0.955294 | 0.926270 | 0.886393 | 0.106351 | 0.133766 | 0.168702 | 0.001137 | 0.001773 | 0.002794 |
| enhanced_pca | 0.955939 | 0.922561 | 0.885145 | 0.105581 | 0.137089 | 0.169627 | 0.001120 | 0.001863 | 0.002825 |
| raw | 0.923042 | 0.884168 | 0.883671 | 0.135931 | 0.164954 | 0.170712 | 0.001766 | 0.002545 | 0.002861 |
| raw_pca_ae | 0.920368 | 0.884347 | 0.881454 | 0.138273 | 0.164827 | 0.172331 | 0.001827 | 0.002541 | 0.002916 |
| enhanced | 0.935084 | 0.922675 | 0.880923 | 0.128155 | 0.136988 | 0.172716 | 0.001651 | 0.001860 | 0.002929 |
| raw_pca | 0.921247 | 0.881346 | 0.880331 | 0.137508 | 0.166952 | 0.173145 | 0.001807 | 0.002607 | 0.002943 |

**Baseline MLP Summary table**

For all datasets 500 epochs were used in training.

| Dataset | Train_R2 | Val_R2 | Test_R2 | Train_RRMSE | Val_RRMSE | Test_RRMSE | Train_MSE | Val_MSE | Test_MSE |
|---|---|---|---|---|---|---|---|---|---|
| enhanced_ae | 0.999150 | 0.921344 | 0.862137 | 0.014669 | 0.138162 | 0.185842 | 0.000022 | 0.001892 | 0.003391 |
| enhanced_pca | 0.996777 | 0.909659 | 0.846218 | 0.028556 | 0.148069 | 0.196279 | 0.000082 | 0.002173 | 0.003782 |
| enhanced_pca_ae | 0.997298 | 0.910391 | 0.838236 | 0.026148 | 0.147468 | 0.201308 | 0.000069 | 0.002155 | 0.003979 |
| enhanced | 0.995360 | 0.906532 | 0.837813 | 0.034261 | 0.150610 | 0.201571 | 0.000118 | 0.002248 | 0.003989 |
| raw_ae | 0.988976 | 0.830673 | 0.819581 | 0.051447 | 0.199441 | 0.212599 | 0.000253 | 0.003720 | 0.004438 |
| raw_pca_ae | 0.989445 | 0.844723 | 0.817606 | 0.050342 | 0.190987 | 0.213759 | 0.000242 | 0.003412 | 0.004486 |
| raw | 0.994644 | 0.816037 | 0.811325 | 0.035859 | 0.207881 | 0.217409 | 0.000123 | 0.004042 | 0.004641 |
| raw_pca | 0.989494 | 0.799793 | 0.790986 | 0.050224 | 0.216865 | 0.228827 | 0.000241 | 0.004399 | 0.005141 |

The tables show that the baseline MLP tends to overfit—with near-perfect $R^2$ on the training set and substantially lower values on validation and test sets—while the complex MLP maintains a more balanced performance, particularly on enhanced feature sets. Across both architectures, the dataset that consistently yielded the best results was the enhanced_ae representation for the baseline MLP and the enhanced_pca_ae representation for the complex MLP. For the baseline model, enhanced_ae achieved the highest test $R^2$ of 0.8621 and the lowest test RRMSE of 0.1858, slightly outperforming all other variants. In contrast, the complex MLP reached its peak performance with enhanced_pca_ae, obtaining a test $R^2$ of 0.8886 and a test RRMSE of 0.1670. These findings suggest that while a moderately compressed nonlinear representation is optimal for simpler models, combining both PCA and Autoencoder compression is most effective when using a deeper MLP. The baseline MLP also yields higher test errors, especially on the raw representations, with test RRMSE values often exceeding 0.20. Other strong performers included raw_ae and enhanced ae, as well indicating high test R2 values and test RRMSE. The raw model, trained without any compression or enrichment, was the weakest of all, confirming the necessity of feature engineering in this context.

**Detailed Metrics by Cement Type for Enhanced_PCA_AE and complex MLP:**

This table shows the metrics for each of the seven cement types for the model which performed better, which as seen in the previous tables, was the complex MLP with the enhanced_pca_ae dataset.

**Mean $R^2$:** 0.5443
**Mean RRMSE:** 0.1535

| Dataset | Type | Samples | R2 | RRMSE | MSE |
|---|---|---|---|---|---|
| enhanced_pca_ae | Type_Lumnite SG5 | 62 | 0.646241 | 0.146109 | 0.001393 |
| enhanced_pca_ae | Type_ISTRA 50 | 206 | 0.643092 | 0.157152 | 0.003758 |
| enhanced_pca_ae | Type_ISTRA 40 | 381 | 0.595674 | 0.132589 | 0.000869 |
| enhanced_pca_ae | Type_Lumnite SG | 168 | 0.551055 | 0.130595 | 0.001191 |
| enhanced_pca_ae | Type_CEMFAST | 85 | 0.517165 | 0.228935 | 0.001639 |
| enhanced_pca_ae | Type_ISTRA 45 | 177 | 0.475486 | 0.115049 | 0.001017 |
| enhanced_pca_ae | Type_ISTRA 50 5.0 | 145 | 0.381716 | 0.163951 | 0.011325 |

Types Lumnite SG5 and ISTRA 50 achieve the highest $R^2$ values, both above 0.64, with RRMSE values within the acceptable 0.14–0.16 range, indicating relatively strong predictive power and consistent error rates. In contrast, performance drops significantly for Type ISTRA 50 5.0, which has the lowest $R^2$ (0.38) and the highest MSE, suggesting that maybe this formulation is more variable. Notably, despite having fewer samples, Type CEMFAST shows decent generalization ($R^2 = 0.51$), while ISTRA 45 achieves the lowest RRMSE (0.115) even with a relatively modest $R^2$.

**$R^2$ Curve and Prediction Distribution for Baseline MLP on Enhanced_AE Dataset**

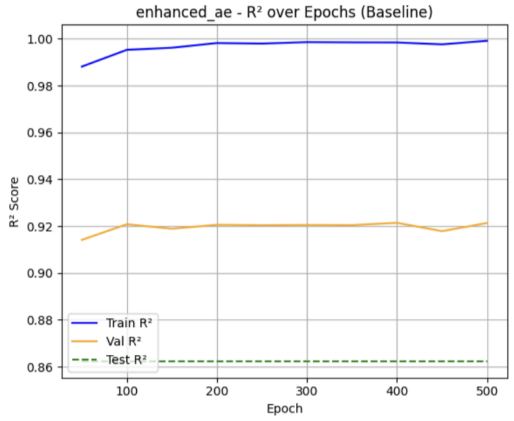


*Figure 7: $R^2$ over Epochs (Baseline MLP on enhanced_ae)*

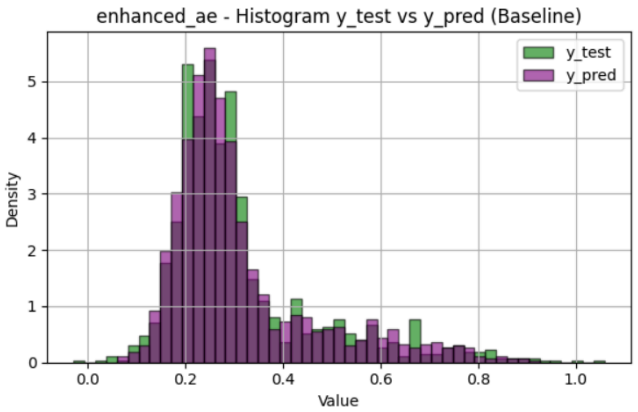


*Figure 8: Predicted vs Real Histogram (Baseline MLP on enhanced_ae)*

The baseline MLP trained on the enhanced_ae dataset exhibits stable training behavior, with the R2 score on the training set rapidly increasing and plateauing near 1.0. Validation R2 remains consistently around 0.92, showing as well that the model has strong generalization. However there is a little downgrade on the Test set, which may suggest a little overfitting. The histogram of predicted vs true values indicates strong alignment in the main peak of the distribution, but some underprediction in the tails, particularly for higher setting time values. This could be because these regions are underrepresented in the training data. Overall, the model fits the dominant patterns well but may struggle slightly in sparse areas.

**$R^2$ Curve and Prediction Distribution for Complex MLP on Enhanced_PCA_AE Dataset**

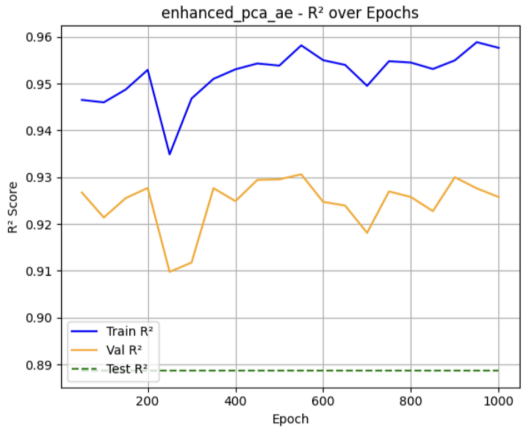


*Figure 9: $R^2$ over Epochs (Complex MLP on enhanced_pca_ae)*

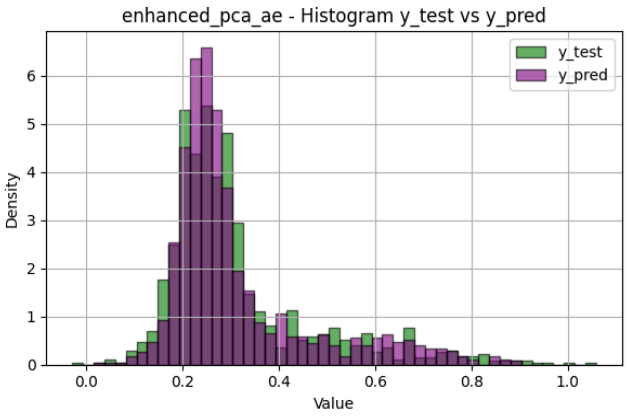


*Figure 10: Predicted vs Real Histogram (Complex MLP on enhanced_pca_ae)*

The complex MLP trained on the enhanced dataset and having applied pca and the Autoencoder to it, demonstrates high training performance, with R2 scores consistently above 0.92, indicating a moderate variability in generalization but still being a strong overall fit. The relatively stable R2 test near 0.89 confirms a good generalization to unseen data. The histogram reveals an excellent alignment between predicted and actual values in the core distribution range, between 0.2 and 0.3, while slight underprediction in higher target ranges. The model captures the main structure of the data well and improves upon the baseline performance, particularly in terms of reduced test error and better tail convergence as well.

**6. Discussion and Future work**

The results of this study confirm that domain-informed feature engineering, when combined with dimensionality reduction techniques such as PCA and Autoencoders, leads to measurable improvements in predictive performance for cement setting time. These improvements validate our hypothesis that reducing feature redundancy and dimensionality enhances learning in neural models—especially in complex, noisy industrial datasets. Importantly, PCA and Autoencoders demonstrated complementary strengths: PCA provided robust denoising and compact representations, while Autoencoders, especially when applied post-PCA, captured richer, non-linear structures that deeper networks could exploit.

Despite consistent performance gains, improvements were often moderate in magnitude. A central factor behind this is the anonymized and scaled nature of the target variable, which may have limited the expressiveness of model predictions. In internal experiments where the original scale of the target was preserved, the same architecture achieved significantly better performance ($R^2 > 0.94$, RRMSE ~ 0.1) on the

enhanced_pca_ae dataset. In contrast, results on the raw dataset remained unchanged, underscoring the importance of structured preprocessing rather than model tuning alone.

We also critically evaluated different target scaling strategies. QuantileTransformer introduced instability in predictions due to distribution distortion; RobustScaler consistently underperformed; and StandardScaler was excluded given the clear non-normality of the target distribution. Among the tested options, MinMax scaling emerged as the most stable and interpretable, largely unaffected by the presence of extreme values. However, these results further reinforce that most performance gains stemmed from careful data preparation and representation learning—not from scaling choices.

Several methodological limitations surfaced. Applying Autoencoders to the entire feature space may have introduced noise from irrelevant variables, diluting the benefit of non-linear encoding. Future work should investigate training Autoencoders on semantically coherent feature subsets (e.g., chemistry, phases, granulometry), potentially improving representation quality and interpretability. Similarly, our augmentation techniques helped balance the dataset but may have introduced implausible data points due to unstructured sampling in feature space. Conditional generative models, such as CGANs or VAE-based samplers, offer a promising direction to generate more realistic and domain-consistent samples.

From a modeling standpoint, deeper MLPs clearly benefited from the enhanced data representations, but also incurred higher parameter costs. Further exploration of parameter-efficient architectures—such as residual connections, attention-based compression, or bottleneck layers—could provide a better trade-off between complexity and generalization. Additionally, transfer learning across cement types or semi-supervised training using unlabeled data could help address sample scarcity for minority classes.

Finally, applying Autoencoders in an unsupervised setting for anomaly detection could support failure prediction or quality control tasks, shifting the focus from average behavior to early warning systems. Collaborative work with industrial partners to maintain interpretable scaling for target variables would also enable clearer benchmarking and facilitate real-world deployment.

In summary, this work demonstrates that deliberate, context-aware preprocessing combined with scalable deep learning methods can yield reliable and generalizable models, setting a strong foundation for further innovation in industrial ML pipelines.

**7. Conclusions**

This project successfully achieved its primary objective: accurately predicting the initial setting time of calcium aluminatum cement by enhancing data representations through domain-specific preprocessing and dimensionality reduction techniques. The combination of chemical knowledge, compositional data handling, and structured modeling pipelines allowed us to reach high predictive performance in a challenging industrial context.

A key finding was that both PCA and Autoencoders significantly improved input representations, though their contributions differed depending on model complexity. PCA was highly effective for simpler models, offering compact representations with minimal information loss. In contrast, Autoencoders demonstrated greater value in deeper architectures, where their non-linear encoding captured more complex relationships. Notably, the Autoencoder achieved its best results when applied after PCA, suggesting that denoising and reducing redundancy first allowed it to learn richer embeddings with fewer unnecessary weights. Together, they proved complementary—bringing out the best of both approaches.

The best-performing model, a deep MLP trained on the enhanced_pca_ae dataset, reached an $R^2$ of 0.8886 and RRMSE of 0.1670, indicating high predictive accuracy. This performance reflects not only architectural advances, but also the extensive efforts made to improve the dataset itself—through careful cleaning, ILR transformations, data balancing, and feature construction. Compared to the raw dataset, the enhanced version consistently delivered stronger results, underscoring the impact of thoughtful data preparation.

However, prediction performance varied across cement types, particularly for underrepresented or highly variable formulations. This points to a limitation in generalization and highlights the need for future work on type-specific tuning or adaptive learning strategies that can better handle inter-class heterogeneity.

The results also confirmed our hypothesis: reducing redundant or irrelevant features in noisy, high-dimensional data enables better learning. Yet, we also noted that the effect of different scalers on prediction quality was nuanced. While MinMax scaling performed robustly, alternative techniques such as the QuantileTransformer and RobustScaler significantly worsened model performance. The NormalScaler was avoided due to the non-Gaussian nature of the data. This highlights that preprocessing choices—even seemingly minor ones—can meaningfully influence final results.

Looking ahead, several improvements are worth exploring. Autoencoders could be redesigned to operate on semantically grouped features (e.g., chemistry, phases, granulometry) to better capture domain structure. Conditional generative models might offer more realistic data augmentation. And preserving the original target scale would help reflect real-world performance more transparently.

Overall, this project demonstrates how structured dimensionality reduction and deep learning, when paired with targeted domain preprocessing, can effectively tackle complex regression tasks in scientific manufacturing settings.

**References**


1. Kılıç, H., & Gürkan, H. (2023). Prediction of setting time in cement using SVR and ANN models. *Construction and Building Materials*, 345, 128309.

2. Liu, Y., et al. (2022). Predicting cement performance using gradient boosting regression and physical process indicators. *Journal of Materials in Civil Engineering*, 34(1), 04021345.

3. Zhao, X., & Zhang, Y. (2024). Hybrid M5–ELM–RF ensemble models for setting time prediction of blended cements. *Expert Systems with Applications*, 224, 119929.

4. Egozcue, J. J., et al. (2003). Isometric logratio transformations for compositional data. *Mathematical Geology*, 35(3), 279–300.

5. Templ, M. (2020). *Applied Compositional Data Analysis: With Worked Examples in R*. Springer.

6. Fernández, A., et al. (2018). SMOTE for regression: Advances and current trends. *Information Sciences*, 456, 278–294.