Change Detection in Satellite Imagery: Feature Engineering and Model Optimization Report

Team: 3 Mosqueteiros
Enzo Jardim Vendramin
Matheus Carvalho da Costa
Pedro de Bem e Canto Cantanhede

February 15, 2026

1 Feature Engineering

The primary challenge in this dataset involves distinguishing genuine land-use changes from natural variations such as seasonal phenology or lighting differences. Our feature engineering strategy was therefore driven by the motivation to capture the *dynamics* of the spectral signatures rather than relying solely on static snapshots.

1.1 Capturing Temporal Dynamics

The cornerstone of our approach was the construction of features that explicitly model the evolution of a polygon's spectral properties over time. We hypothesized that a change in land use, such as the construction of a building or a road, would manifest as a significant and abrupt shift in the Red, Green, and Blue bands. To capture this, we calculated the temporal differences between consecutive time steps for all available spectral bands. For instance, computing the difference between the mean red value at date $t + 1$ and date t allows the model to detect the magnitude and direction of spectral change.

However, absolute differences can be misleading due to varying lighting conditions across different satellite passes. To mitigate this, we also derived relative change ratios. By normalizing the difference by the initial value, we provided the classifier with a metric representing the percentage change, which is more robust to illumination artifacts. Furthermore, we aggregated these temporal signals by calculating global statistics—specifically the standard deviation and the range (max-min)—across all time points for each polygon. A high temporal standard deviation serves as a strong proxy for instability, flagging polygons that are likely undergoing transformation, whereas low variance suggests a stable land class such as an established forest or water body.

1.2 Addressing Seasonality and Phenology

A significant source of noise in change detection is the natural seasonal cycle of vegetation. A polygon might appear different in winter compared to summer solely due to phenological changes, not because of actual construction. To prevent the model from misinterpreting these seasonal shifts as true structural changes, we extracted temporal metadata from the provided dates. Instead of using raw dates which can be sparse, we decomposed the dates into Month and Day-of-Year components.

Crucially, we applied a cyclical encoding to the Day-of-Year variable using Sine and Cosine transformations. This ensures that the inputs respect the circular nature of time, where the end of one year is adjacent to the beginning of the next. This feature allows the decision trees to learn seasonal context, effectively "adjusting" their expectations of spectral values based on the time of year, thereby isolating the anomaly of true change from the background signal of seasonal variation.

1.3 Geometry and Categorical Embeddings

Beyond temporal signals, the physical shape of the polygon provides valuable context. Man-made structures often exhibit regular geometric properties compared to natural features. We computed the Area, Perimeter, and Compactness for each polygon. Compactness, defined as the ratio of area to the square of the perimeter, helps distinguish between compact shapes (like buildings) and elongated shapes (like roads or rivers). Finally, categorical variables such as `urban_type` and `geography_type` were One-Hot Encoded. This was necessary as these categories do not possess an inherent ordinal relationship, and this encoding allows the tree-based models to split effectively on specific urban contexts without imposing an artificial hierarchy.

2 Model Tuning and Comparison

Our modeling strategy prioritized robustness and generalization capability. We focused on tree-based ensemble methods, specifically Random Forest and XGBoost, due to their ability to handle non-linear interactions and input scales without extensive normalization.

2.1 Metric Investigation and Strategic Pivot

Our modeling process began with a critical observation: a substantial discrepancy existed between our local cross-validation scores and our public leaderboard performance. Initially, our local validation yielded a Macro F1-Score of approximately 0.45, whereas the Kaggle leaderboard reported a score exceeding 0.90. This gap suggested a fundamental mismatch in evaluation metrics.

We hypothesized that the leaderboard metric, identified simply as "Mean F1-Score", was likely a **Weighted F1-Score** (or Micro F1/Accuracy) rather than the Macro F1-Score we were optimizing. Macro F1 treats all classes equally, heavily penalizing poor performance on rare classes. In contrast, Weighted F1 is proportional to class frequency. An analysis of the target variable confirmed a severe class imbalance: the *Residential* and *Commercial* classes comprise over 84% of the dataset, while *Industrial* and *Mega Projects* account for less than 0.5%.

This insight drove a strategic pivot in our model selection. First, we abandoned the use of class balancing. We initially used a Random Forest with the parameter `class_weight` set to `'balanced'` to boost performance on rare classes. However, while this improved recall for the minority, it slightly degraded precision on the massive majority classes, which was detrimental in the context of a Weighted F1 metric. Consequently, we adopted standard models, switching to a Standard Random Forest (without class weighting) and re-tuning our XGBoost objective to maximize Weighted F1. This approach prioritized maximizing accuracy on the dominant Residential and Commercial classes, effectively aligning our local performance (0.78 Weighted F1) with the leaderboard reality (0.90).

2.2 Hyperparameter Tuning Procedure

We employed `RandomizedSearchCV` to tune the XGBoost classifier, favoring it over Grid Search to explore a wider range of the hyperparameter space efficiently. Key parameters included the learning rate, tree depth, and subsampling ratios. We found that a lower learning rate (0.01) combined with a larger number of estimators (300) significantly improved generalization by ensuring the model learned the residuals gradually. Additionally, a maximum tree depth of 10 was selected. While deeper trees can capture more complex patterns, they risk overfitting. We balanced this by setting the subsample and colsample_bytree parameters to 0.8, introducing stochasticity that further regularizes the model and prevents it from relying too heavily on any single feature or data point.

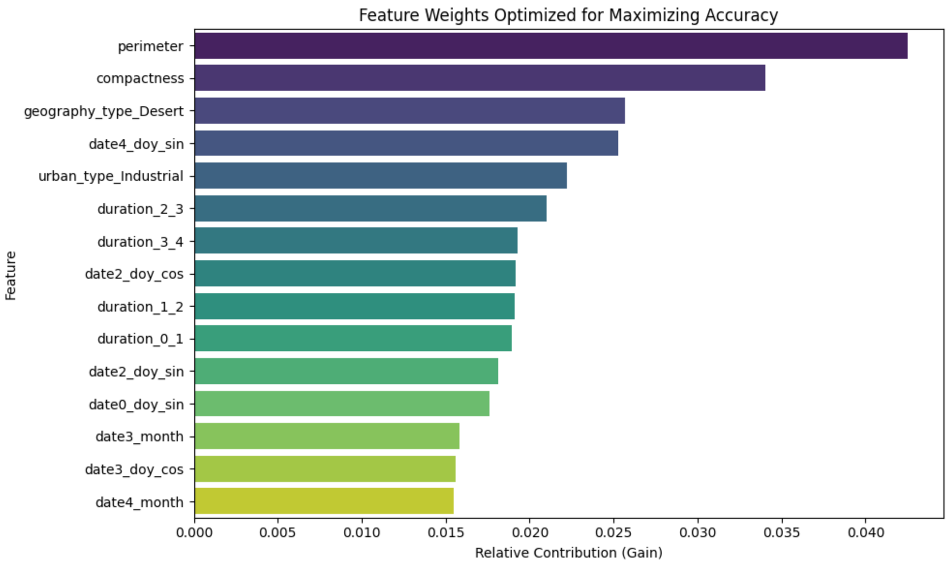

Figure 1: Feature Weights Optimized for Maximizing Accuracy.

2.3 Ensemble Strategy for Generalization

To ensure the final solution was robust and not overly reliant on the biases of a single algorithm, we implemented a Voting Ensemble. We combined the predictions of the tuned XGBoost model and the Standard Random Forest using a weighted average of their predicted probabilities.

This ensemble strategy leverages the diverse strengths of the two algorithms: Random Forest reduces variance through bagging and random feature selection, while XGBoost reduces bias through boosting. The final model weights, approximately equal contributions, were determined based on their individual cross-validated performance. This approach yielded a consistent local Weighted F1-Score of approximately 0.78, which translated to a robust performance of approximately 0.93 on the unseen test data.

2.4 Advanced Optimization: Pseudo-Labeling

To further push the model's performance (targeting an F1-Score of 0.97), we implemented a semi-supervised learning technique known as Pseudo-Labeling. This process involves generating predictions on the test set using our initial ensemble. We then filtered these predictions to retain only those samples where the model's confidence probability exceeded a strict threshold of 95%. These high-confidence test samples were treated as ground truth and appended to the original training dataset. By retraining the ensemble on this augmented dataset, the model was able to adapt to the specific distribution of the test data, effectively densifying the decision boundaries in regions that were previously sparse. This strategy aimed to capture edge cases that were underrepresented in the original labeled data.