Multi-Date Satellite Change Detection

Kaggle Team Name: Armand Coiffe

Team Members: Mathieu Morales, Lucas Grille, Armand Coiffe

1 Feature Engineering

Our feature engineering pipeline is structured around three main components: (1) temporal consistency, (2) spectral enrichment, (3) geometric extraction.

The dataset contains five temporal observations per geographic zone, including construction status, RGB image statistics and polygon geometry.

Since the competition metric is Macro-F1, our objective was to design informative features capable of improving class discrimination under strong class imbalance.

1.1 Exploratory Analysis and Class Imbalance

An initial analysis revealed a strong imbalance between classes as previously said. Some change types contain significantly more samples than others.

For this reason, during internal validation we monitored both:

- **Macro-F1**, to remain aligned with Kaggle evaluation,
- **Weighted F1**, to account for class support and overlook the overestimation of performance on dominant classes.

Weighted F1 ensures that minority classes are not ignored during model selection.

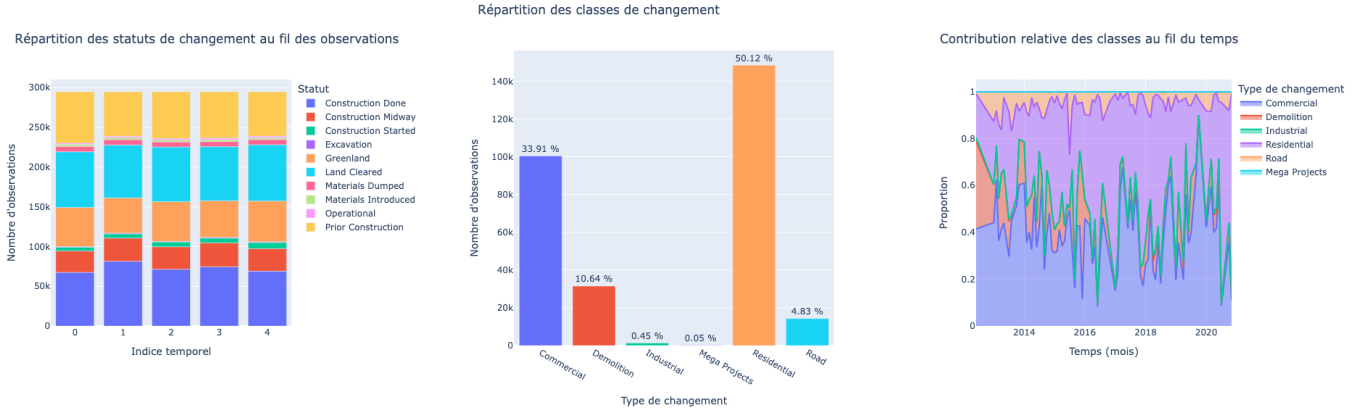

Figure 1: Exploratory analysis, class distribution and relative contribution of change types.

The first plot shows the evolution of construction statuses across dates, confirming that change is inherently temporal and justifying chronological reordering and time-gap features.

The second plot highlights strong class imbalance, with some categories largely dominating others. This motivated the use of Weighted F1 during validation.

The third plot illustrates different temporal behaviors across change types, supporting the creation of progression and duration features.

1.2 Chronological Reordering

The five date columns are not always chronologically ordered.

Since change detection relies on temporal progression, misordered timestamps would produce incoherent sequences.

For each sample, we:

- Sorted the five dates in ascending chronological order,
- Applied the same permutation to associated status and RGB features.

This guarantees temporal consistency before any feature derivation.

1.3 Missing Values Handling – Strategy 1

Several imputation strategies were implemented. The selected strategy is **S1_trainfit_keep**, see notebook. This strategy helped us to earn more than 2percent in kaggle macro score, it consists of:

- Imputing numerical variables using training-set medians,
- Imputing categorical status variables using training-set modes,
- Applying the same parameters to the test set,
- Replacing missing dates with a sentinel value ("1900-01-01"),
- Adding missingness indicators: `has_missing_date`, `nb_missing_dates`, `n_dates_valid`.

This approach prevents data leakage, ensures train/test consistency, and preserves maximum information.

1.4 Temporal Features

Dates were converted to datetime format. We computed inter-date differences: $\Delta_i = date_{i+1} - date_i$

We also engineered:

- **adv**: a progression indicator derived from status evolution,
- **duration**: an estimated construction duration based on detected progression.

These features explicitly model construction dynamics.

1.5 Advanced Spectral Features

- From RGB means and standard deviations, we generated enriched spectral indices: NDGR (Green–Red ratio), Green dominance, Normalized green, Excess Green (ExG), Contrast, saturation, luminosity, Channel variances and dispersion indicators.
- We also created temporal spectral aggregates such as: Green range, Green temporal trend, Consecutive variations, Volatility scores.

These descriptors capture vegetation removal and surface transitions.

1.6 Geometric Features

From polygon geometry, we extracted:

- Area and perimeter, Bounding-box dimensions, Length-to-width ratio, Perimeter-area ratio, Thinness indicator, Vertex count, Elongation via rotated rectangle, and more
- Binary indicators: `is_mega_area`, `is_very_elongated`, and more.

These features help distinguish linear structures from large compact constructions.

1.7 Encoding and PCA

Change status variables were one-hot encoded consistently on train and test sets. Multi-valued columns were processed using MultiLabelBinarizer that avoided more sparse columns (from one hot encoding).

Principal Component Analysis (PCA) was applied exclusively to colorimetric features after standardization, retaining 85% (tuned parameter) of explained variance. This reduces redundancy while preserving essential spectral information.

2 Model Tuning and Comparison

2.1 Evaluation Protocol

We used a stratified train/validation split to preserve class distribution. Due to strong imbalance, **Weighted F1-score** was used as the primary internal validation metric. Macro-F1 was also monitored to remain aligned with Kaggle evaluation. Confusion matrices were systematically analyzed for class-wise performance.

2.2 Compared Models

We evaluated several classifiers for this multi-class task.

Logistic Regression was used as a linear baseline. However, due to the complex interactions between temporal, spectral and geometric features, its performance was limited.

We then tested **Random Forest**, which is well suited for heterogeneous tabular data and can capture non-linear relationships effectively.

Finally, we evaluated **XGBoost**, a boosting model capable of modeling complex feature interactions and often performing strongly on structured data.

Model selection was based on validation Weighted F1-score and class-wise confusion matrix balance.

2.3 Final Model Selection

The final model was selected based on: Highest validation Macro-F1 performance, Stable Weighted F1 and balanced confusion matrices. For training, we first divide our train set into two to obtain a validation set, then we retrain the model each time on the entire train set to obtain a maximum amount of data. To try to get the best scores, we tried to weight the classes, but without success, so we abandoned the idea. We tested several loss functions, but the best one was mlogloss.

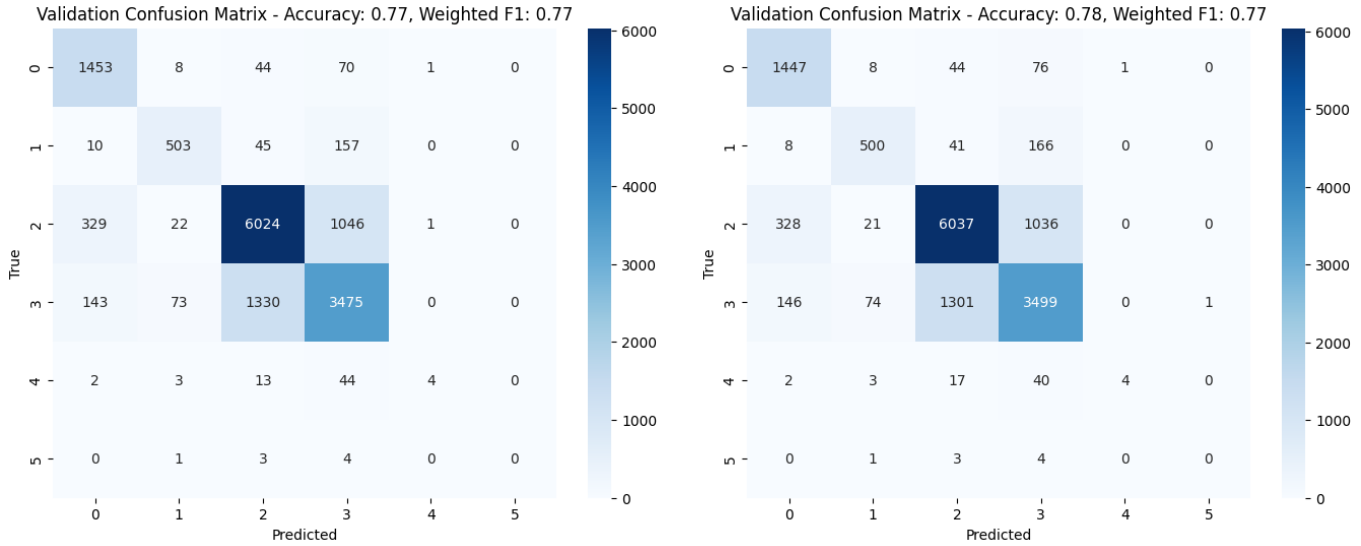

Figure 2: Validation and training confusion matrices of the selected model. On the left XGboost and on the right randomForest

The confusion matrices show consistent performance across the main classes, with strong diagonal dominance indicating good predictive accuracy.

Both models achieve similar Weighted F1 scores, confirming robustness despite class imbalance. XGBoost slightly improves minority-class predictions, while Random Forest shows stable overall behavior.

Misclassifications mainly occur between structurally similar classes, suggesting overlapping feature distributions rather than model instability.

2.4 Overfitting Control

Overfitting was mitigated through: hyperparameter tuning, we tried to vary the parameters of our models, for example by trying to find the optimal depth of the trees, the amount of tree,and also thanks to dimensionality reduction via PCA. We also tried different loss format with already weighted sample.

2.5 Conclusion

On the available data of the test set XGBoost provided the best score in kaggle but Random Forest provided simular result. We kept both models on the kaggle in case the other performs better on the total test set.