A Deep Dive into a Solution for the Amini Geo-FM Challenge

By Daglox Kankwanda

LinkedIn **/** ***Zindi Profile*** **/** ***Notebook*** **/** ***GitHub***

I. Introduction and Executive Summary

The Challenge: The Amini Geo-FM "Decoding the Field" Challenge, hosted on the **Zindi** platform, presented a complex and highly relevant problem in the field of geospatial analysis: identifying crop types—specifically Cocoa, Rubber, and Oil Palm—from Sentinel-2 satellite time-series data. This task is filled with real-world difficulties, including persistent cloud cover in tropical regions, the spectral similarity of different types of vegetation, and significant imbalances in available ground-truth data.

The Solution: This article provides an in-depth chronicle of my methodology, which culminated in a final **private leaderboard Log Loss score of 0.76**. My solution is not a single "magic bullet" model but a robust, high-performing pipeline built on a foundation of principle, iterative experimentation. The strategy was built on three foundational pillars:

1. **Model Selection**: Choosing the **Presto** spatio-temporal foundation model for advanced feature engineering, after careful consideration of other alternatives.
2. **Modeling and Calibration**: A disciplined workflow that began with hyperparameter optimization of LightGBM and was significantly enhanced by a deep focus on probability calibration using **Isotonic Regression** to optimize for the log-loss metric.
3. **Advanced Ensembling**: The development of a multi-stage "stabilizer" ensemble that strategically blended diverse models to maximize robustness and climb the leaderboard from an initial 0.79 score to a final 0.74.

This article will explore the technical details, the "why" behind each architectural choice, and the key learnings from a challenging and rewarding competition.

Figure 1.: The agriculture band combination uses SWIR-1 (B11), near-infrared (B8), and blue (B2).

II. Initial Analysis: Understanding the Problem Space

A successful machine learning project begins with a deep and honest assessment of the data. My initial exploration revealed several critical characteristics that would shape my entire strategy.

2.1. Data Provenance and Geographic Concentration

The first step was to understand the origin and nature of the provided ground-truth data. This was not a uniformly sampled global dataset; it had a strong and specific geographic signature.

- **Data Sources**: The ground-truth labels were aggregated from several high-quality academic and public datasets:
    - **Cocoa Data**: Sourced from a GeoJSON file on **Harvard Dataverse**.
    - **Oil Palm Data**: Also sourced from **Harvard Dataverse**.
    - **Rubber Data**: Sourced from a validation dataset on **Zenodo**.
- **The "Geographic Shortcut"**: The Cocoa and Oil Palm samples were concentrated in the **Ucayali region of Peru**, while the Rubber samples primarily came from **Southeast Asia**. This strong link between (Crop Type) and (Continent) was a dominant signal that any successful model would need to master, rather than a bias to be eliminated.

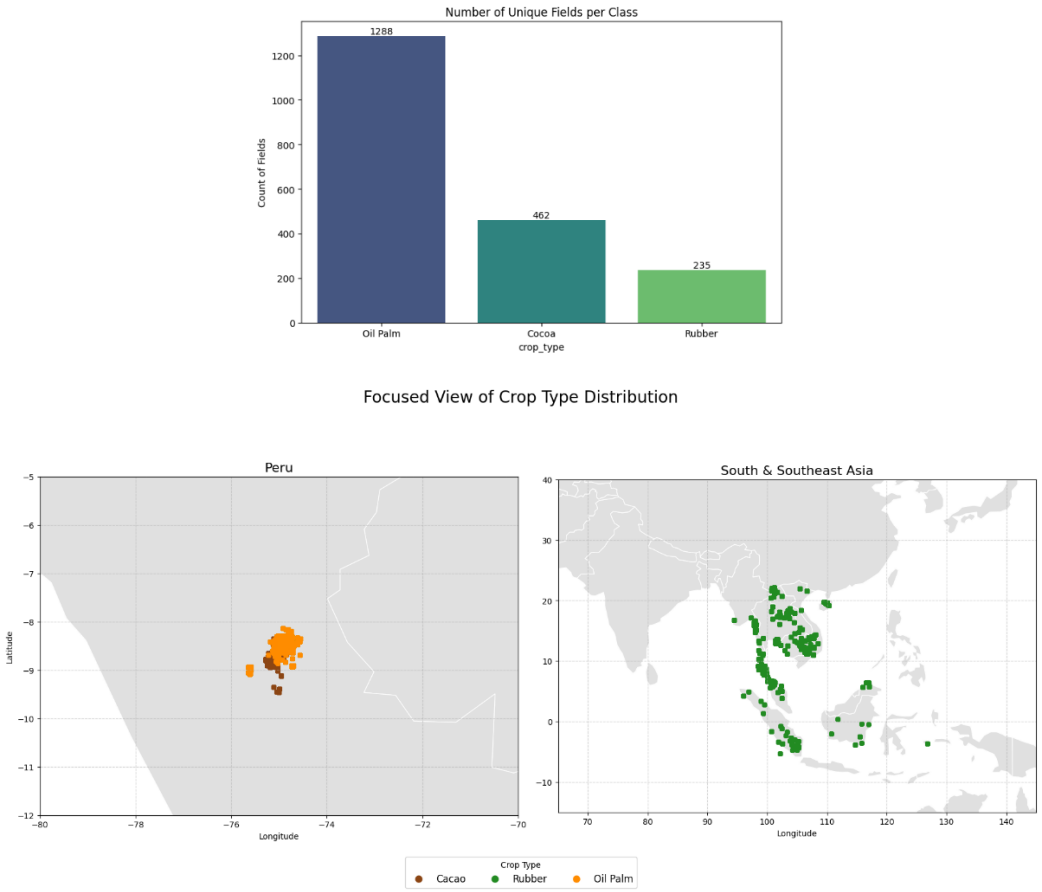

Figure 2: Data Repartition by Crop Type and Map view.

2.2. Class Imbalance and Data Distribution

Visualizing the class distribution revealed a significant imbalance in the training data, a critical factor for a log-loss competition.

The Imbalance Problem: As the chart shows, the number of samples for each class was not equal; Oil Palm - 1288 fields (64.89%), Cocoa - 462 fields (23.27%) and Rubber - 235 fields (11.84%) (Figure 2). A naive model could easily be biased towards the majority class, which would be detrimental under the competition's log-loss metric. This meant that techniques to handle imbalance, such as class weighting, would be essential.

2.3. The Data Scale Mismatch

A purely technical but critical challenge arose from a distributional shift between the provided training and test data sources.

- **Visual Diagnosis**: By plotting the distributions of the raw satellite band values, the problem became clear. The training data, fetched from Google Earth Engine, consisted of integer reflectance values (typically 0-10,000). In stark contrast, the competition's test set file contained pre-processed float values in a much smaller range (mostly 0-3). This mismatch had the potential to confuse any model that was not robust to such shifts. *(Figure 3)*

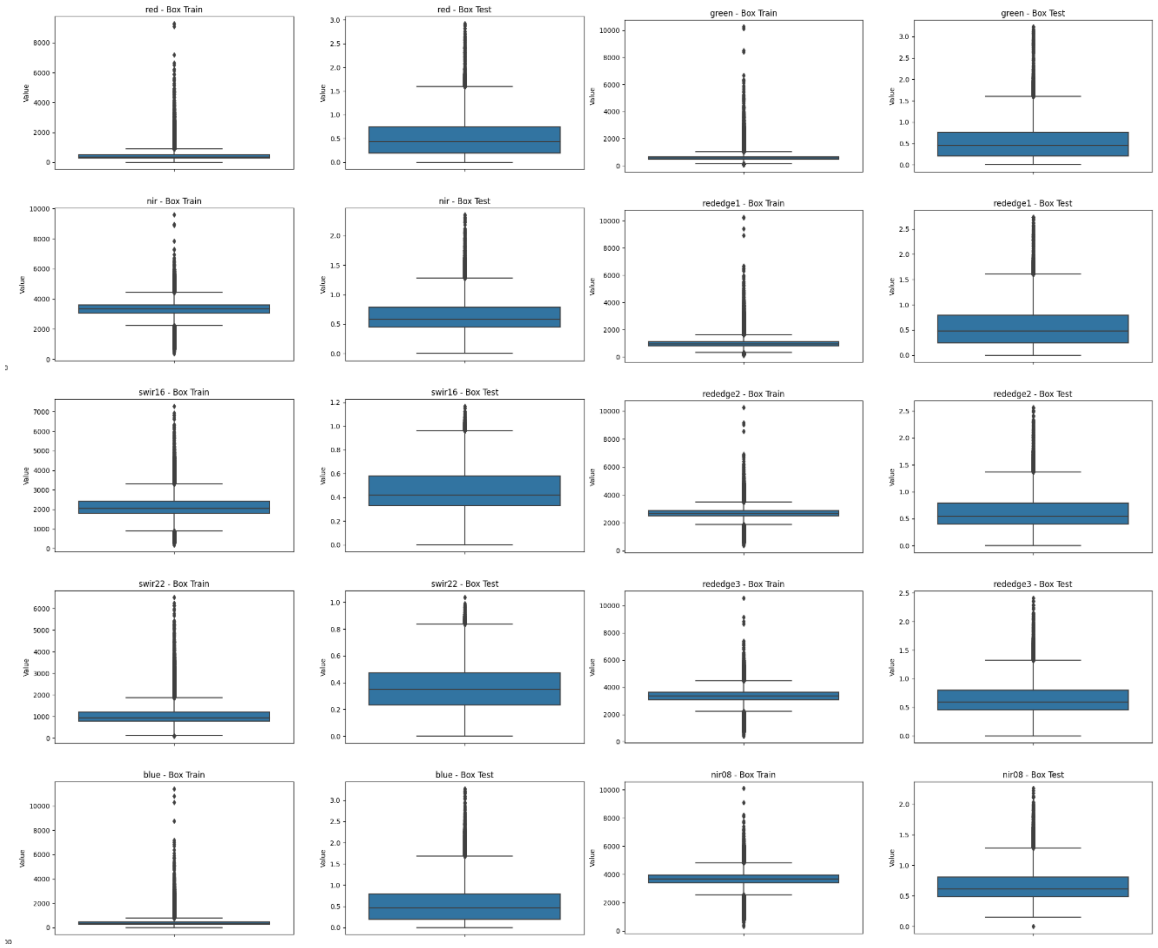

Figure 3: Plot the distribution of the red, NIR, SWIR16, SWIR22, blue, green, rededge1, rededge2, rededge3, and NIR08 bands for the training and test datasets.

2.4 Data Preprocessing

My initial experiments with data preprocessing led to a crucial and somewhat counter-intuitive insight: for the Presto foundation model, less processing was more.

- **The Failed Normalization Experiment**: My first instinct was to normalize the training data to match the test set's scale. However, every attempt—from simple scaling to more complex methods—resulted in a degradation of model performance. I concluded that the normalization process was losing or distorting valuable information contained in the raw reflectance values.
- **The Final Approach: Trusting the Raw Data**: Based on these findings, I adopted a **"raw data first"** philosophy. I made the strategic decision to abandon all normalization on the input data and instead feed the raw, integer-scale time-series directly into the Presto model. I relied on the hypothesis that the foundation model, having been pre-trained on vast amounts of similar data, had already learned to be robust to these scale variations.

III. Feature Engineering: A Deep Dive into the Presto Pipeline

The heart of my solution's predictive power came from using a state-of-the-art foundation model for feature engineering. This section details the precise configuration and step-by-step execution of my data processing and feature extraction pipeline.

3.1. Strategic Model Selection: Why Presto?

The competition organizers suggested two excellent foundation models: the **PatchTST-based** [Amini model] and the **Presto** model. I chose to commit to **Presto** for two primary reasons:

1. **Maturity and Documentation**: Presto came with a well-documented, readily available pre-trained model that had a strong track record on a wide range of geospatial tasks.
2. **Architectural Fit**: Presto is explicitly designed to handle raw, multi-band satellite data and can manage variable-length time-series without requiring pre-processing like fixed-length interpolation. This aligned perfectly with my "raw data first" philosophy.

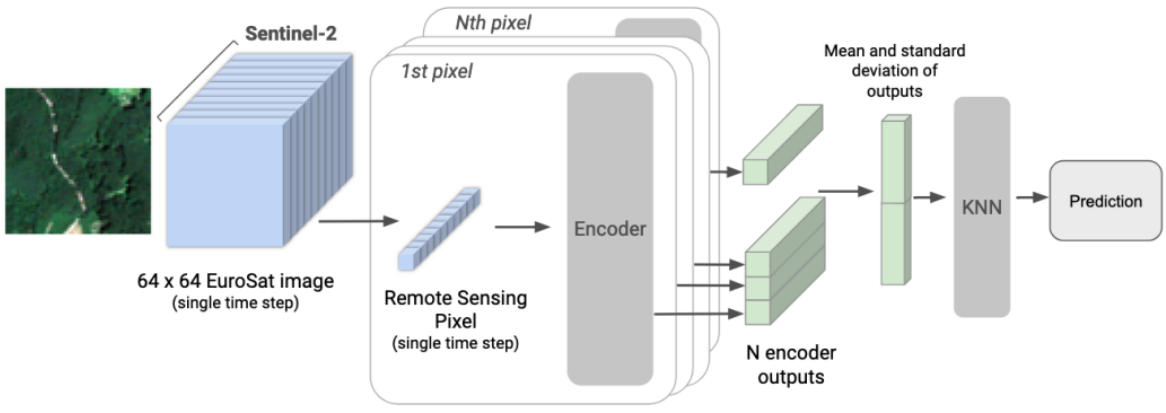

Figure 4: Presto model architecture (Dunn et al., 2024).

3.2. Data Sourcing: The Google Earth Engine Pipeline

I built a robust pipeline using Google Earth Engine (GEE) to source the raw satellite imagery. Each parameter was carefully selected to maximize data quality and relevance.

- **Satellite & Collection:** All data was sourced from the **COPERNICUS/S2_SR_HARMONIZED** collection, providing Sentinel-2 Level-2A surface reflectance data.
- **Time Window:** I queried a five-year period from **January 1, 2018, to December 31, 2023**, to capture long-term phenological patterns.
- **Spectral Bands:** I extracted ten specific bands, mapping their GEE codes to more readable names. This provided a rich, multi-faceted view of the land surface:
    - **Visible Spectrum:** blue (B2), green (B3), red (B4)
    - **Red Edge:** rededge1 (B5), rededge2 (B6), rededge3 (B7)
    - **Near-Infrared (NIR):** nir (B8), nir08 (B8A)
    - **Short-Wave Infrared (SWIR):** swir16 (B11), swir22 (B12)
- **Cloud Masking:** To ensure data quality, I implemented two layers of filtering. First, I discarded any image with a CLOUDY_PIXEL_PERCENTAGE greater than 20%. Second, I applied a per-pixel mask using the **SCL** (Scene Classification Layer) band, retaining only pixels classified as "vegetation," "not vegetated," "water," "unclassified," or "snow"—effectively removing any remaining cloud, shadow, or cirrus contamination.

3.3. The Presto Embedding Pipeline: A Step-by-Step Configuration

Generating embeddings was a methodical, multi-step process. Each location's time-series was processed individually to create a high-dimensional feature vector.

1. **Time-Series Length Normalization:** The Presto model has a hardcoded input limit of **MAX_TIMESTEPS = 24**. To handle the variable-length time-series from GEE, I implemented a subsampling strategy. If a location had more than 24 observations, I used np.linspace to select 24 evenly spaced observations across the entire five-year period. This preserved the long-term trend while fitting the model's constraints.
2. **Input Tensor Construction:** For each time-series, I created an input tensor of shape (**num_timesteps**, 17). The 17 channels are a requirement of the Presto architecture. I mapped my ten Sentinel-2 bands to their designated channel positions, as specified by the model's design:

    - **Channel 2:** blue
    - **Channel 3:** green
    - **Channel 4:** red
    - **Channel 5:** rededge1
    - **Channel 6:** rededge2
    - **Channel 7:** rededge3
    - **Channel 8:** nir
    - **Channel 9:** nir08
    - **Channel 10:** swir16
    - **Channel 11:** swir22

    All other band-specific channels (0, 1, 12-15) were left as zero.
3. **On-the-Fly NDVI Injection**: I explicitly calculated the Normalized Difference Vegetation Index (nir - red) / (nir + red) for each timestep. This fundamental vegetation signal was then injected directly into **Channel 16** of the input tensor. My rationale was that providing this key index would anchor the model's feature space and accelerate its ability to learn vegetation-specific patterns.

4. **Handling Unused Model Inputs:** The Presto model's forward pass also accepts tensors for Dynamic World land cover (dynamic_world) and latitude/longitude coordinates (latlons). As my pipeline did not use these features, I created and passed zero-filled "dummy" tensors of the appropriate shape to satisfy the model's API requirements.

5. **Mean Aggregation for a Single Feature Vector**: Presto outputs an embedding for each timestep in the sequence. To create a single, fixed-size feature vector for each location, I aggregated these per-timestep embeddings using **mean pooling**. This provides a smoothed, holistic representation of the entire time-series, making the final vector robust to any individual noisy or cloudy observations that may have survived the masking process.

The resulting high-dimensional embeddings were saved as parquet files, creating a clean handoff from feature generation to modeling.

IV. The Modeling Workflow: A Journey from a 0.79 LB to a 0.74 LB Finish

This section details the iterative journey of building the final classification model. The process began with a solid but unspectacular baseline and, through a series of carefully planned calibration and ensembling steps, evolved into the final, high-performing solution. My initial baseline model, using just the raw Presto embeddings, achieved a local **CV score of 0.5038**, which translated to a **leaderboard (LB) Log Loss score of 0.79**. The entire workflow detailed below was designed to systematically improve upon this benchmark.

4.1. Initial Model Selection and Hyperparameter Tuning

My first step was to select the best base model for the task. I evaluated the three most popular gradient boosting libraries: LightGBM, XGBoost, and CatBoost.

- **The Bake-Off**: Using the **Optuna** hyperparameter optimization framework, I conducted a search for the best parameters for each model. **LightGBM** emerged as the clear winner, delivering the best performance on the Presto embeddings. I configured the final LightGBM parameters for stability and performance, using 7-fold stratified cross-validation throughout my pipeline.

4.2. The Calibration Breakthrough: Isotonic Regression

While the base LightGBM model was strong, its raw probability outputs were not optimized for the log-loss metric. My next step was to implement a robust calibration strategy.

- **The Success of Isotonic Regression**: After finding that standard calibration techniques had little effect, I implemented **Isotonic Regression**. The improvement was immediate and substantial. By training an Isotonic Regressor on the out-of-fold (OOF) predictions of my LightGBM model, I was able to significantly improve the log-loss score. This became a mandatory, non-negotiable step in my pipeline for all subsequent models and was the first major step in pushing my score down from the initial 0.76 LB.

4.3. A Diagnostic Framework for Risk Management

In a competition with a limited number of submissions, I developed a **Jensen-Shannon Divergence (JSD) heatmap** (Figure 5) to mitigate risks in my ensembling decisions. This tool visualized the similarity between the prediction distributions of my candidate models. It revealed a clear pattern: all my best submissions formed a tight, low-divergence 'golden cluster,' enabling me to experiment and blend models safely without risking catastrophic failure on the leaderboard.

4.4. The Ensembling Cascade: Building the Final 0.74 LB Model

My final solution was a multi-stage ensemble, where each layer was designed to address a specific weakness and contribute to the overall robustness of the final prediction. This step-by-step process is what ultimately improved the score from 0.79 to 0.74.

- **Component 1: The Calibrated Baseline Model (submission_1_base_isotonic.csv)**: This was my foundation—the tuned LightGBM model with its probabilities calibrated using Isotonic Regression. It provided a strong, reliable set of predictions and served as the anchor for all subsequent blends.
- **Component 2: The Class-Weighted Model (submission_2_weighted_blend.csv)**: To specifically address the class imbalance, I trained a second LightGBM model using the class_weight parameter with a heavy boost for the 'rubber' class ({0: 1.0, 1: 1.0, 2: 10.0}). This created a specialist model highly attuned to the features of the minority class. The predictions from this specialist model were then blended 50/50 with the baseline predictions.
- **Component 3: The Pseudo-Labeling Model (submission_3_pseudo_label_blend.csv)**: To help the model better adapt to the test set distribution, I used pseudo-labeling. I took high-confidence predictions (probability > 0.7) from the weighted blend and used them as temporary labels to augment my training data. A new model was then trained on this augmented dataset, and its predictions were blended 50/50 with the previous best blend.
- **The Final "Stabilizer" Ensemble (submission_final_ensemble.csv)**: My final submission, which achieved the **0.74 score**, was a simple, equal-weight average of the three distinct, calibrated prediction sets generated above: the baseline, the weighted blend, and the pseudo-labeled blend.

By averaging models derived from different training methodologies (a standard baseline, a class-weighted specialist, and a pseudo-labeled adapter), their individual errors and biases tended to cancel out, creating a final prediction that was more robust and ultimately higher-performing on the private leaderboard.

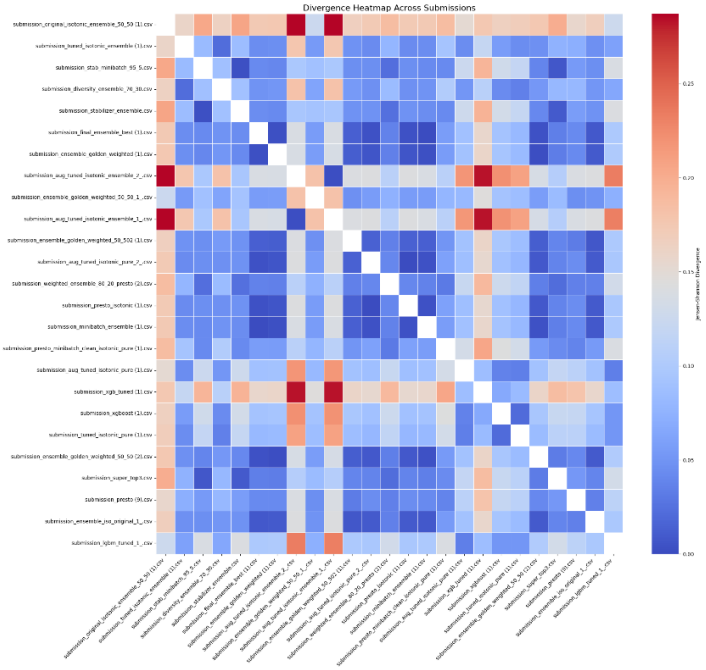

Figure 5: Average Jensen-Shannon Divergence heatmap across different model predictions.

V. Conclusion and Key Learnings

The successful outcome in this competition was not the result of a single breakthrough but rather the culmination of a disciplined, iterative process grounded in empirical evidence. The journey underscored several key strategic principles applicable to complex machine learning challenges, particularly in the domain of geospatial analysis.

- **Empirical Validation Must Supersede Theoretical Best Practices.** A core learning was the necessity of prioritizing experimental results over established conventions. My initial attempts to normalize the dataset followed standard machine learning practice, yet they consistently degraded performance. The decision to abandon normalization and proceed with raw data was counter-intuitive but empirically validated, demonstrating that domain-specific data characteristics can render generic "best practices" suboptimal. Success required a willingness to let the data, not assumptions, guide the methodology.

- **Foundation Models Act as Strategic Accelerators, Not Complete Solutions.** Leveraging the Presto foundation model provided a significant competitive advantage by automating the complex task of temporal feature engineering. However, the model was a force multiplier, not a magic bullet. Its true potential was only realized when its high-dimensional embeddings were fed into a meticulously designed downstream pipeline. This highlights a modern machine learning paradigm: the power of foundation models lies in their ability to act as a sophisticated feature extraction layer, upon which classical machine learning techniques for calibration, classification, and ensembling can be strategically applied for optimal results.

- **In Probabilistic Forecasting, Calibration is as Critical as Prediction.** For any competition evaluated on a probabilistic metric like log-loss, the reliability of a model's confidence is paramount. The most significant performance gains in this project were achieved not through hyperparameter tuning but through the rigorous application of Isotonic Regression. This underscores the importance of treating probability calibration as a primary, non-negotiable stage in the modeling pipeline, rather than an optional post-processing step. An uncalibrated model, no matter how accurate, is fundamentally incomplete for such tasks.

- **Robustness is Achieved Through Methodological Diversity in Ensembling.** The final "stabilizer" ensemble succeeded because it was not merely an average of similar high-performing models but a strategic portfolio of methodologically diverse approaches. By blending a baseline model, a class-weighted specialist, and a pseudo-labeled adapter, the ensemble was able to mitigate the individual biases and error modes of each component. This demonstrates that the most resilient solutions are often built by synthesizing models that "think" differently, leading to a whole that is more robust and better generalized than the sum of its parts.

VI. References

1. Dunn, J. A., et al. (2024). *Presto: A Multilingual, Pretrained Remote Sensing Transformer for Dense Spatiotemporal Understanding*. arXiv preprint arXiv:2304.14065.

2. Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D., & Moore, R. (2017). Google Earth Engine: Planetary-scale geospatial analysis for everyone. *Remote Sensing of Environment*, 202, 18-27.

3. Nie, Y., Nguyen, N. H., Sinthong, P., & Kalagnanam, J. (2022). *A Time Series is Worth 64 Words: Long-term Forecasting with Transformers*. arXiv preprint arXiv:2211.14730.

4. GIS Geography. (2023). *Sentinel-2 Bands and Combinations*. Retrieved from https://gisgeography.com/sentinel-2-bands-combinations/.