A Machine Learning Pipeline for Avalanche Forecasting in Data-Sparse Regions

Ron Simenhois
Colorado Avalanche Information Center

1. Abstract

This report outlines our development and evaluation of a machine learning pipeline for forecasting avalanche hazard periods, with a particular focus on addressing the challenges posed by incomplete and sparse observational data. Traditional avalanche forecasting relies heavily on expert analysis of localized data, a resource that is often unavailable in many remote, avalanche-prone regions. To overcome this limitation, our project leverages simulated snowpack data and a sophisticated Positive-Unlabeled (PU) learning framework to develop a robust and generalizable forecasting model.

We developed a two-stage pipeline that first predicts the likelihood of an avalanche *event* occurring and then uses this prediction as a key feature to forecast a multi-level avalanche *hazard rating*. The core of the system is a custom "Spy+Bootstrap" algorithm using a RandomForest classifier, which successfully identifies hazardous periods from a noisy dataset where most safe days are unlabeled.

The final model demonstrates strong predictive power. We achieved an F1 score of **0.83** on the full dataset, which is approximately **94%** of the performance of a theoretical "Oracle" model trained on perfect data. Our analysis using SHAP (SHapley Additive exPlanations) reveals that the model's predictions are driven by physically intuitive factors, particularly the *change* in snowpack and weather conditions over 3- to 9-day windows. This project establishes a successful methodology for data-driven avalanche forecasting models that can be trained and deployed across diverse geographic areas, enhancing the capabilities of human forecasters.

2. Introduction

Avalanche forecasting is a critical public safety service, yet it remains a complex and data-intensive task. Forecasters traditionally synthesize a wide array of information, including manual snow pit data, weather station readings, and field observations of avalanche activity. However, they often restrict the collection of this data to accessible, high-use areas, leaving vast backcountry regions with little to no observational coverage. This data scarcity poses a significant barrier to producing reliable and widespread avalanche forecasts.

This project confronts this challenge by developing a machine learning pipeline that does not depend on direct, localized observations. Instead, it primarily utilizes outputs from the SNOWPACK simulation model, which provides detailed, physically-based estimates of snow stratigraphy and stability across a landscape.

A primary obstacle we faced in training a machine learning model for this task is the nature of the available labels. While a reported avalanche serves as a definitive "positive" label (an avalanche occurred), the *absence* of a report does not guarantee safety. An avalanche may have occurred but gone unobserved, or the conditions may have been hazardous without a triggering event. This creates a classic **Positive-Unlabeled (PU) learning problem**, where the model must learn to identify positive instances from a large pool of unlabeled data that contains a mix of true negatives and hidden positives.

In this report, we outline our methodology for tackling the PU learning problem, the architecture of the resulting two-stage forecasting pipeline, the model's performance, and key insights into the physical drivers of avalanche hazard as learned by the model.

3. Methodology

We centered the project's methodology on a two-stage modeling pipeline built within a Positive-Unlabeled (PU) learning framework.

3.1. The Positive-Unlabeled (PU) Learning Framework

We framed the forecasting task as a PU learning problem. After evaluating several techniques, including One-Class SVM, Bagging PU, and weighted methods, we determined that a two-step "Spy+Bootstrap" approach provided the most robust and accurate results.

The Spy+Bootstrap Algorithm:
We designed this algorithm to "clean" the unlabeled data to create a more reliable training set for the final classifier.

1. **Spy Phase (Reliable Negative Identification):** We hide a small fraction (15%) of the known positive samples (avalanche days) as "spies" within the large unlabeled dataset. We then train a preliminary classifier to distinguish these spies from the rest of the unlabeled data. By analyzing the model's output scores, we can establish a conservative threshold for classification. We identify any unlabeled sample scoring significantly lower than the spies as a **Reliable Negative (RN)**.
2. **Training Phase:** We construct a new, high-quality training dataset using only the original Labeled Positives (P) and the newly identified Reliable Negatives (RN). We then train the final classifier on this clean P vs. RN dataset, free from the noise of hidden positives that were present in the original unlabeled set.

3.2. Data and Feature Engineering

We trained and evaluated the model using data from the 2023-2024 and 2024-2025 winter seasons. To ensure high-quality training labels, we focused on seven backcountry polygons in Colorado that we selected for their high frequency of observed avalanche activity (top 90th percentile).

Input Features:
The model relies on a rich set of features we derived from SNOWPACK simulations and numerical weather prediction models. These features describe the state of the snowpack and recent weather, which we categorized as:

- **Weak Layer Properties:** Stress, grain size, viscosity, density, etc.
- **Slab Properties:** Density, hardness, thickness, and load of the snow layers overlying the weak layer.
- **Upper Snowpack Properties:** Characteristics of the top 15 cm of snow.
- **Weather Data:** Wind speed, wind direction, temperature, and new snow amounts.

A critical step in our feature engineering was the creation of **temporal statistics**. Instead of relying on single-day measurements, we calculated the mean and standard deviation of most features over 3- and 9-day rolling windows.

3.3. Two-Stage Modeling Pipeline

Our forecasting pipeline consists of two distinct models that run in sequence:

1. **Avalanche Event Model (PU Classifier):** This is the core PU learning model we described above. It takes the snowpack and weather features as input and outputs a probability score (from 0 to 1) indicating the likelihood of an avalanche *event* occurring. We train this model using the Spy+Bootstrap method with a RandomForest classifier.
2. **Avalanche Hazard Model (Multi-Class Classifier):** The output from the event model—the raw and adjusted probability scores—becomes a crucial input feature for this second model. The hazard model is a multi-class classifier (LGBMClassifier) that predicts the final avalanche hazard rating (e.g., Low, Moderate, Considerable, High). This stacked approach enables the second model to capitalize on the nuanced output of the first, resulting in a more accurate final forecast.

We orchestrate the entire process, from data generation to final prediction, with a master Python script (run_pipeline.py), ensuring a reproducible and automated workflow.

4. Results and Performance

We evaluated the model's performance against a theoretical benchmark known as the **"Oracle" model**. The Oracle is a RandomForest classifier trained on perfect, fully labeled data—a scenario not achievable in the real world. We designed our PU model to approach Oracle's performance closely.

To simulate realistic data scarcity, we trained our model with a P_Fraction of 0.5, meaning we labeled only 50% of the true hazardous periods as positive during training.

Final Model Performance (P_Fraction = 0.5):

Model	F1 Score (Tuned)	Precision	Recall	AUC
Spy+Bootstrap (RF)	**0.741**	**0.689**	**0.800**	**0.907**
Spy+Bootstrap (XGBoost)	0.606	0.636	0.579	0.828
Oracle (Benchmark)	*0.797*	*0.717*	*0.897*	*0.957*

The selected RandomForest-based model achieved an F1 score that is **~94%** of the theoretical maximum performance set by the Oracle. The final report for the avalanche event model, which we trained on the full dataset, shows a final **F1 score of 0.833** and an **AUC of 0.954**. This result demonstrates the effectiveness of the Spy+Bootstrap methodology in recovering a strong predictive signal from incomplete data.

The multi-class hazard model also performed well, achieving a weighted F1 score of **0.927** across all hazard levels on the full dataset. However, a deeper analysis of the model's performance reveals important nuances. While generally accurate, the confusion matrix (Figure 1) shows a tendency for the model to misclassify the highest hazard level. When the true hazard is Level 4, the model correctly identifies it only 81% of the time, with a combined **19% "leakage"** where it incorrectly predicts a lower Hazard Level 3 (16%) or Level 2 (3%). This indicates a conservative bias against predicting the most severe outcomes.

This behavior is also reflected in the model's calibration, as shown in the reliability diagrams (Figure 2). For Hazard Levels 2 and 3, the model tends to be **over-confident**; its predicted probabilities are slightly higher than the actual frequency of those events. Conversely, for Hazard Level 4, the model is **under-confident**, particularly in the middle probability ranges. This suggests that the model's overprediction of mid-range hazards (Levels 2 and 3) may be capturing some events that were, in reality, more severe (Level 4), leading to the observed leakage and underprediction for the highest threat level.

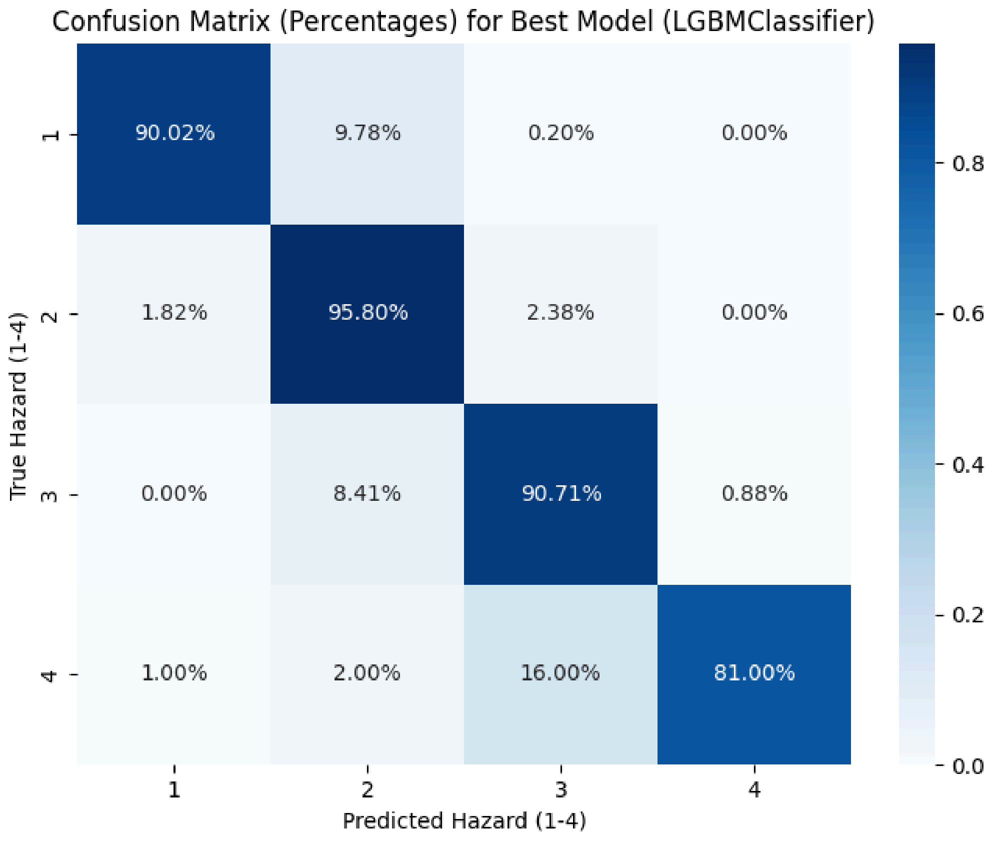

Figure 1: Confusion matrix for the final hazard model (LGBMClassifier). Values show the percentage of predictions for each true hazard level (rows) versus the predicted hazard level (columns). The strong diagonal indicates high classification accuracy across all levels.

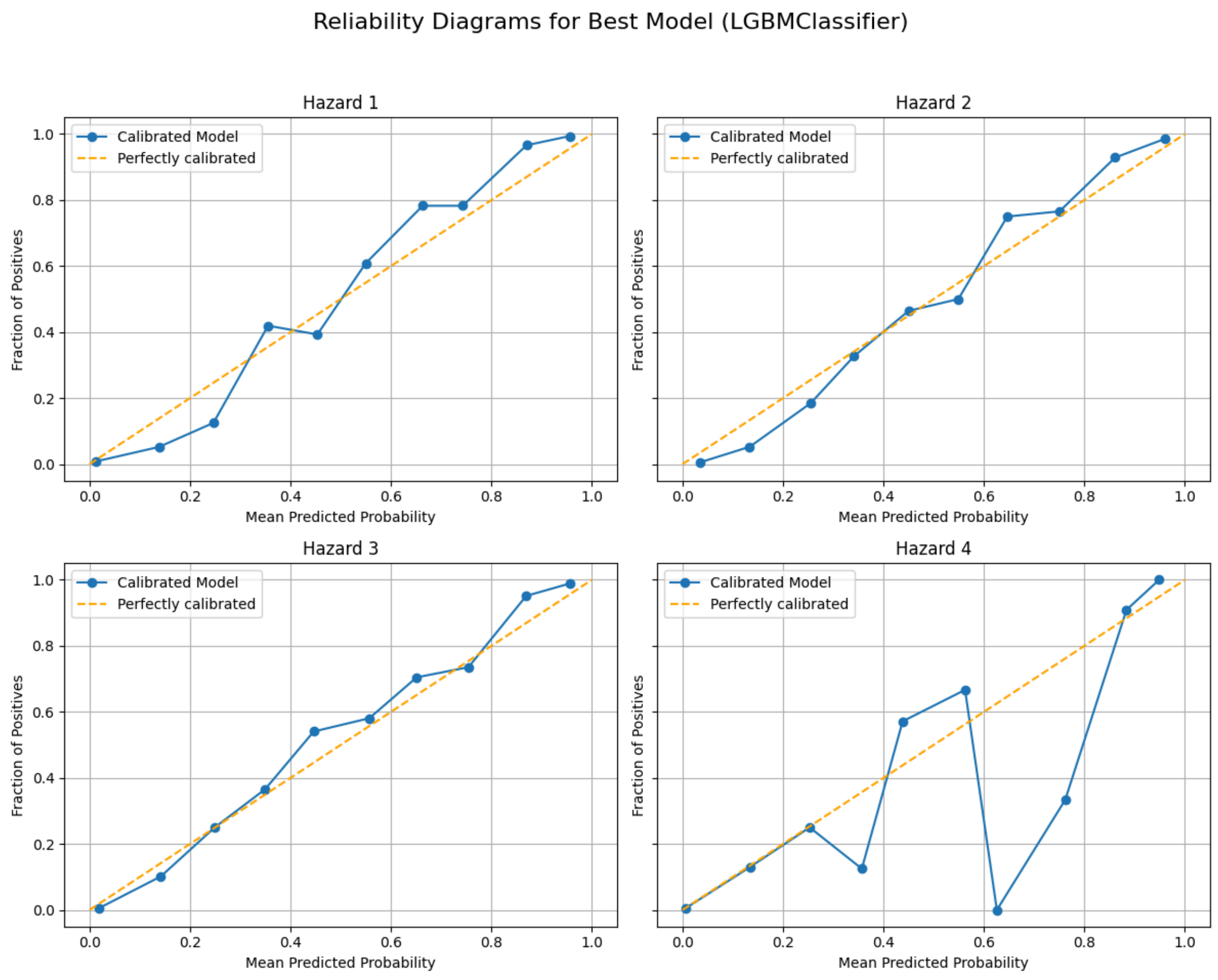

Figure 2: Reliability diagrams for the final hazard model. ***How to read this plot:*** *Each chart compares the model's predicted probability (x-axis) to the actual observed frequency of that hazard level (y-axis). The dashed orange line represents a perfectly calibrated model where a 40% prediction corresponds to a 40% real-world frequency. The blue line shows our model's performance. A blue line below the diagonal indicates overconfidence (the model predicts a higher probability than what actually occurs), while a line above it indicates underconfidence. Overall, except for level 4, where the predicted probability ranges from 60% to 70%, the plots suggest that the model's predicted probabilities are well-calibrated, closely tracking the true frequency of events.*

5. Model Output and Application

The ultimate goal of this pipeline is to produce an actionable daily first-guess forecast that can aid human experts. Figure 3 illustrates the model's output for January 14, 2024, and highlights its core value proposition.

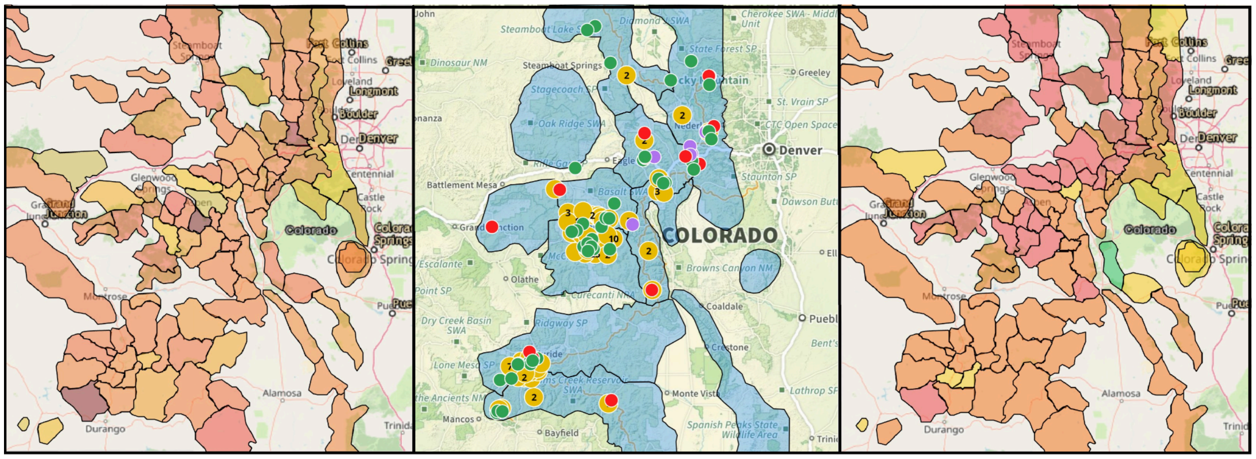

Figure 3: A comparison of model outputs and observed avalanches. ***Center:*** *Observed avalanche activity from Jan 13-15, 2024. Note the high concentration of reports in high-traffic areas.* ***Left:*** *The probability of an avalanche event around Jan 14, 2024, as predicted by our first-stage model.* ***Right:*** *The final multi-class avalanche hazard rating from our second-stage model for the same day.*

The central panel of Figure 3 shows observed avalanche activity, which is heavily concentrated in areas with high observer density (e.g., popular backcountry skiing zones). Between these areas, there are vast regions with few or no avalanche reports. This does not mean no avalanches occurred there; it may simply reflect a lack of observation.

This is where the model's utility becomes clear. The left and right panels show the model's predicted event probability and final hazard rating for the entire region. By using simulated snowpack and weather data, the model can "fill in the gaps," providing a spatially continuous assessment of conditions. It identifies high hazards in areas with many observations, but also flags other, unobserved regions as dangerous, providing a more comprehensive picture of the landscape-level hazard. While these models are not a silver bullet, we are encouraged that their outputs can directly inform forecasters' decisions, especially by drawing attention to data-sparse areas that might otherwise be overlooked.

To further enhance the utility of the forecast, we added a **confidence** parameter to the model's output. This value represents the model's final, calibrated probability for its chosen hazard rating. For a forecaster, this provides a crucial layer of insight into the model's certainty. A high confidence score (e.g., >80%) indicates that the model's prediction is stable and based on strong signals in the input data. Conversely, a low confidence score can act as a flag, signaling that the conditions are ambiguous or unusual, and that the model's output for that specific region should be treated with more caution, warranting closer human inspection.

6. Model Insights from SHAP Analysis

To ensure the model's predictions were based on sound physical principles, we used SHAP (SHapley Additive exPlanations) to interpret its decision-making process. The analysis revealed several key insights that align with expert knowledge in avalanche science.

Insight 1: The Critical Role of Temporal Dynamics
The most influential features were overwhelmingly statistical measures we calculated over 3- to 9-day windows. The model learned that change over time is more predictive than any single-day snapshot.

- **Volatility as a Predictor:** High standard deviations in weak layer stress (weak_layer_stress_std_5d) and crystal shape (weak_layer_rc_flat_std_3d) were strong indicators of increasing hazard. This shows the model identifies periods of rapid change and instability.
- **Persistence as a Predictor:** The mean wind speed over multiple days (wind_speed_mean_5d, wind_speed_mean_9d) was a top feature, indicating that sustained periods of wind loading are more significant than short-lived gusts.

Insight 2: Forecasting is a "Sum of Parts" Problem
No single feature dominated the model's predictions. Instead, the model spread importance across dozens of indicators from different domains (weather, slab, weak layer), as seen in the SHAP plots for both the event model (Figure 4) and the final hazard model (Figure 5). This mirrors the process of a human forecaster, who synthesizes many weak signals into a holistic assessment. The model's strength lies in its ability to integrate these diverse and interacting factors. Notably, the adjusted_score from the event model is a highly influential feature in the final hazard model, validating our two-stage approach.
Insight 3: Complexity is Necessary
Our attempts to simplify the model by removing features with low SHAP values or high correlation resulted in a decrease in predictive accuracy. This suggests that even seemingly minor features provide important contextual information that is crucial for making accurate predictions in complex systems, such as snowpacks.

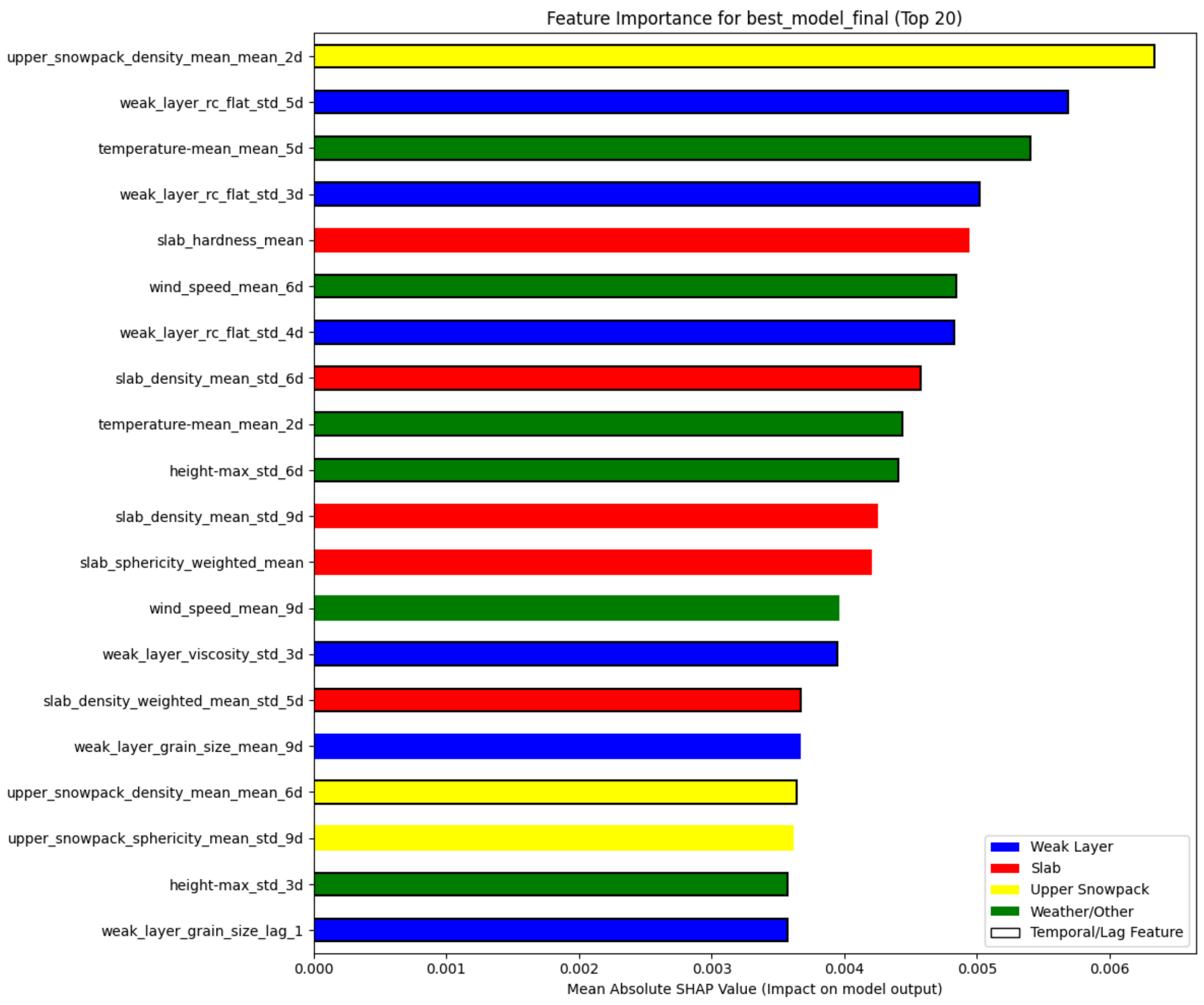

Figure 4: SHAP feature importance for the first-stage ***Avalanche Event Model****. The plot highlights the model's reliance on temporal statistics across weather, weak layer, slab, and upper snowpack domains to predict the probability of an avalanche occurring.*

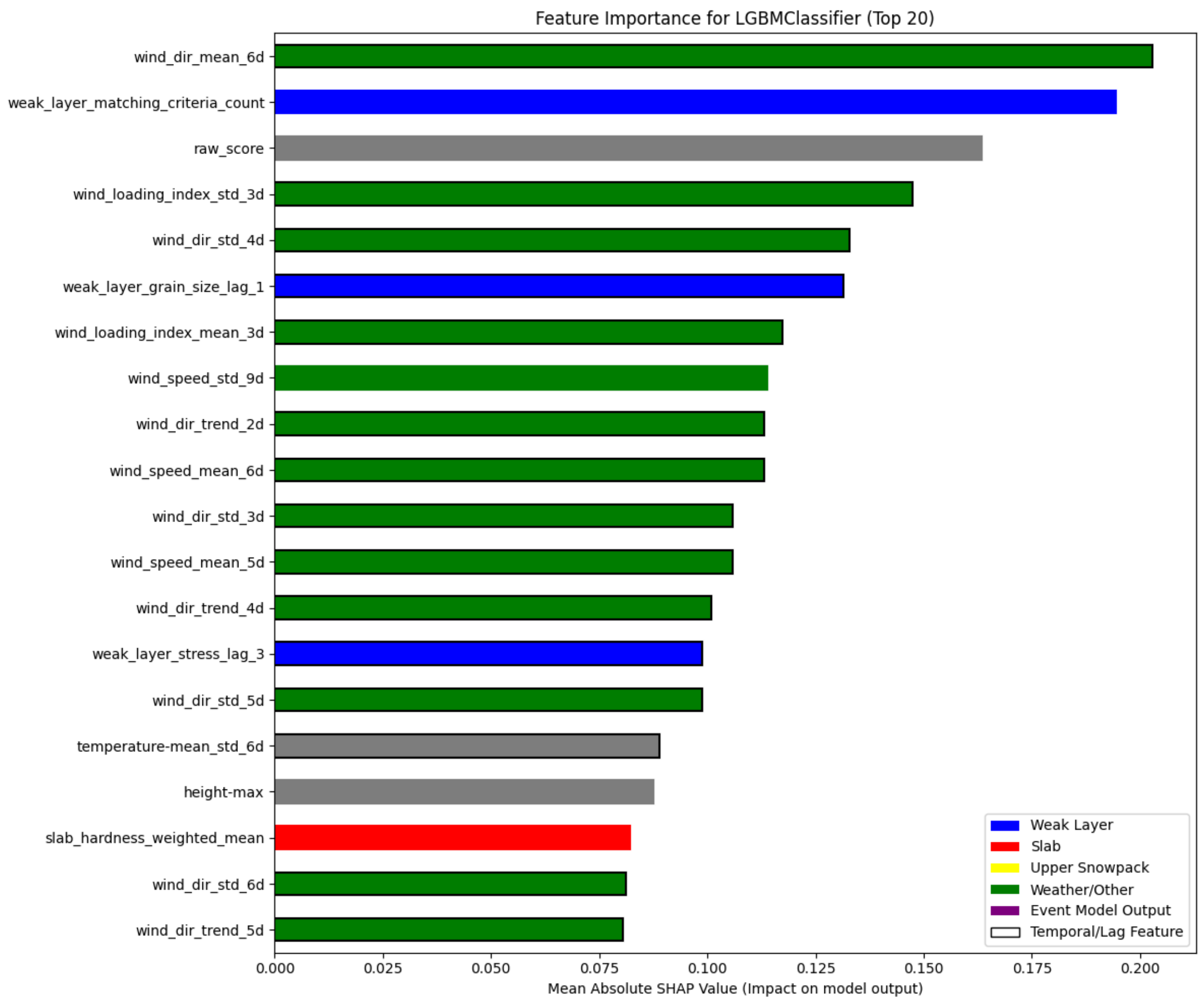

*Figure 5: SHAP feature importance for the second-stage **Avalanche Hazard Model**. The adjusted_score from the first model is a top predictor, alongside various measures of wind and weak layer properties, demonstrating the success of the stacked pipeline.*

7. Pipeline Architecture and Deployment

We structured the project as an end-to-end Python pipeline designed for operational forecasting.

Core Scripts:

- snowpack_reader.py: A hardware-adaptive module for efficiently parsing and processing SNOWPACK .pro files, with support for GPU acceleration.
- train_avalanche_event_model.py: Implements the PU learning, feature selection, and training for the first-stage event model.
- train_avalanche_hazard_model.py: Trains the second-stage multi-class hazard model.
- run_pipeline.py: An orchestration script that executes the entire data processing and

model training sequence in the correct order.
- run_prediction.py: A script to generate daily predictions for a specified date, which can optionally generate a map visualization of the results.

This modular architecture allows for easy maintenance, retraining, and deployment in an operational environment.

8. Future Work

While this work provides a strong foundation, we have identified several avenues for future enhancement that could further improve the model's performance and utility:

8.1. Addressing Model Under-Confidence at Level 4

As noted in the performance analysis, the model shows a tendency to be under-confident when predicting the highest hazard level. This is a common issue stemming from the rarity of Hazard 4 events in the training data. We plan to address this through two primary strategies:

- **Post-Hoc Calibration:** We will apply calibration techniques to the model's raw probability outputs. Methods like **Isotonic Regression** or **Platt Scaling** can create a mapping function that corrects the systematic under-prediction, ensuring that a predicted 70% probability more closely reflects a true 70% frequency of occurrence.
- **Training Process Adjustments:** We will experiment with techniques to address the class imbalance directly during training. By assigning a higher **class weight** to Hazard 4 events, we can force the model to penalize misclassifications of this critical category more heavily. Additionally, data resampling techniques such as **SMOTE (Synthetic Minority Over-sampling Technique)** can be used to generate more training examples of high-hazard days.

8.2. Improving Data Pipeline Scalability

The current data pipeline processes SNOWPACK files in a serial manner. To handle larger geographic areas or higher-resolution simulations, we will re-engineer this component using a library like `xsnow` to parallelize the reading and processing of SNOWPACK information. This will create a more sophisticated and scalable data input pipeline that can support more complex models.

8.3. Improving Input Data with Averaged Profiles

The current model uses data from the single closest simulation point. In future versions, we will test using an "average" snowpack profile for each polygon, which may create a more representative and less noisy input signal.

8.4. Systematic Feature Reduction

Although our initial attempts at simplification reduced accuracy, a more careful, iterative process of culling low-impact and highly correlated features could lead to a faster, more interpretable model without sacrificing performance.

9. Conclusion

This project explored the development of a machine learning pipeline for avalanche forecasting, with a focus on addressing the challenges posed by sparse data. By framing the problem within a Positive-Unlabeled learning context and implementing a custom Spy+Bootstrap algorithm, we developed a model that shows promise in navigating the challenges of incomplete data. The model achieved performance approaching that of a theoretically perfect, fully informed model. The insights derived from SHAP analysis suggest that the model's decision-making is consistent with established snow science principles, which is an encouraging result. This work contributes to the ongoing effort of applying machine learning to enhance avalanche safety and provides a potential framework for data-driven forecasting support in the backcountry.

10. Code Availability

The complete source code for the data processing, model training, and analysis pipeline described in this report is publicly available on GitHub at: https://github.com/ronimos/avalanche_hazard_prediction_model.

11. References

Core Methodologies & Models

- **Positive-Unlabeled (PU) Learning:**
    - Elkan, C., & Noto, K. (2008). Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 213-220).
    - Liu, B., Dai, Y., Li, X., Lee, W. S., & Yu, P. S. (2003). Building text classifiers using positive and unlabeled examples. In *Third IEEE International Conference on Data Mining* (pp. 179-186).
- **Spy Method:**
    - Liu, B., Lee, W. S., Yu, P. S., & Li, X. (2002). Partially supervised classification of text documents. In *ICML'02: Proceedings of the Nineteenth International Conference on Machine Learning* (pp. 387-394).
- **Random Forest:**

- Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1), 5–32.

- **LightGBM (LGBMClassifier):**
  - Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... & Liu, T. Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems 30* (pp. 3146-3154).

Model Interpretation & Evaluation

- **SHAP (SHapley Additive exPlanations):**
  - Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30* (pp. 4765-4774).
- **Evaluation Metrics (Precision, Recall, F1-Score, AUC):**
  - Powers, D. M. (2011). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness & correlation. *Journal of Machine Learning Technologies*, 2(1), 37-63.
  - Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8), 861-874.