SUPPLEMENTARY MATERIAL:

ENVIRONMENTAL PREDICTION OF PATIENT DISCOMFORT

1.GLOSSARY OF TERMS:

- **Binary Classification**: A supervised learning task involving two outcome categories. In this project, predicting whether a patient experiences discomfort (1) or not (0).
- **Logistic Regression (LR)**: A linear model used for classification that predicts the probability of a binary outcome based on input features.
- **Random Forest (RF)**: A non-linear ensemble model that builds multiple decision trees and uses majority voting for prediction. It is robust to noise and handles feature interactions well.
- **Feature Selection**: Identifying the most relevant input variables to improve model performance, reduce complexity, and eliminate noise from irrelevant features.
- **Class Imbalance**: A condition where one class dominates the dataset (e.g., majority of samples showing "no discomfort"), potentially biasing the model.
- **Train-Test Split**: The division of the dataset into a training portion (70%) and a testing portion (30%) to ensure unbiased evaluation.
- **Cross-Validation (CV)**: A technique (10-fold used here) to partition the training data into subsets, ensuring models are validated on different data splits to assess generalizability.
- **Lambda (λ)**: The regularization parameter used in logistic regression to penalize large coefficients and prevent overfitting. The best λ was selected via grid search.
- **Regularization**: A method to reduce overfitting by adding a penalty (controlled by λ) to the loss function, discouraging complex models.
- **Confusion matrix** – 2×2 table listing true positives, false positives, true negatives, and false negatives.
- **Precision**: The proportion of predicted positives that are actually correct (TP / (TP + FP)). High precision indicates few false positives.
- **Recall (Sensitivity)**: The proportion of actual positives correctly identified (TP / (TP + FN)). High recall means fewer false negatives.
- **F1-Score**: The harmonic mean of precision and recall, giving a balanced measure of model performance on imbalanced data.
- **AUC (Area Under Curve)**: A scalar summary of the ROC curve, representing the model's ability to distinguish between the two classes. Higher values indicate better performance.
- **ROC Curve**: A graphical plot illustrating the trade-off between the true positive rate and false positive rate at different thresholds.
- **Training Error**: The fraction of misclassifications on the training data, used to assess underfitting or overfitting.
- **Test Error**: The error calculated on unseen test data, indicating generalization performance.
- **Training Time / Prediction Time**: Metrics capturing how long a model takes to train and make predictions. Important for operational deployment.

2. Intermediate Results and Visual Analysis

2.1 Class distribution variables:

-Comfort (class 0): 103 sessions (10.3%)

-Discomfort (class 1): 897 sessions (89.7%)

2.2 Feature variability:

Feature	Mean	StdDev
Temperature	25.095	2.9241
humidity	54.668	8.7868
noise_level	54.087	14.578
aqi	101.49	28.015
wind_speed	1.0841	0.50669
lighting	547.46	269.53
oxygen_concentration	20.264	0.43435
co2_concentration	670.16	191.48
air_pressure	1000.2	11.562
air_flow_speed	1.1163	0.51146
air_pollutants	49.389	29.097

Table 1. Summary statistics of environmental features

2.3 Descriptive Statistics & Implementation Impact:

Initial data inspection revealed that sensor scales differed by orders of magnitude (e.g., Air Pressure ~1000 vs. Wind Speed ~1. This necessitated Z-Score Standardization. Without this, the Logistic Regression solver failed to converge during early trials.

2.4 Class Imbalance Handling Choice:

Frequency-based class weighting was selected over sampling methods to preserve the original data distribution and avoid overfitting. It integrated seamlessly with both Logistic Regression and Random Forest in MATLAB and significantly improved recall and F1 without generating synthetic data.

2.5 Generalization & Overfitting Check:

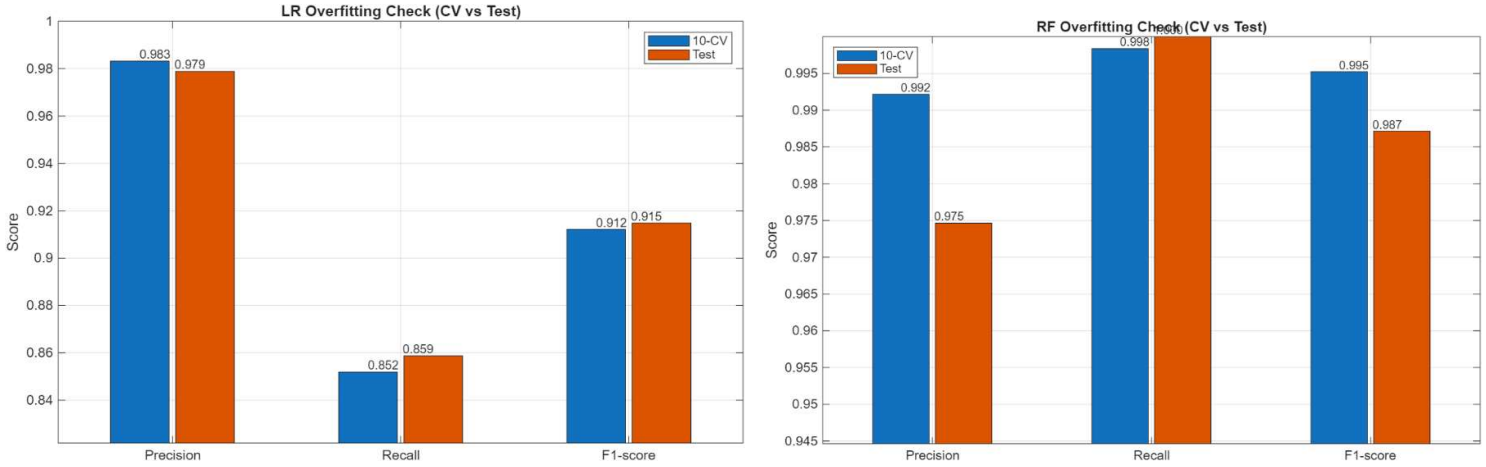

Figure 1: Overfitting analysis for LR (left) and RF (right). The minimal gap between CV and Test F1-scores confirms that both models generalize effectively to unseen data.

2.6 Discrimination Analysis:

Before comparing models, we evaluated individual ROC curves. Both models showed high "lift" over the random-chance baseline.

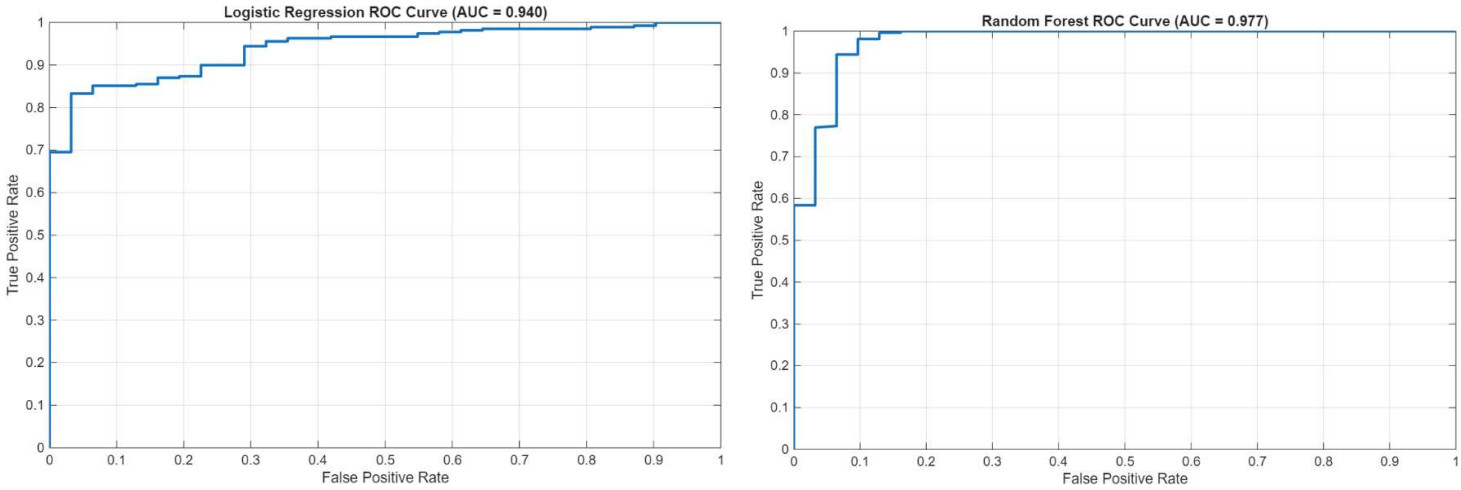

Figure 2: Individual discriminative power of the linear vs. non-linear approach

3. Implementation Details and Parametrization

3.1 Hyperparameter Tuning (Grid Search):

We performed extensive tuning to find the optimal balance between bias and variance.

-Logistic Regression:

- Ridge λ values tested:
$$\lambda \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}.$$
- Best CV F1-score (~0.912) at λ = 1e-5.
- Larger λ (≥ 0.1) produced lower recall and F1, indicating underfitting; these configurations were discarded.

-Random Forest:

- Grid over:
  - Number of trees: [10, 20, 50, 100, 150]
  - Minimum leaf size: [1, 3, 5, 10]
- Best CV F1-scores (~0.995–0.998) obtained for 20–50 trees and leaf size 10.
- Forests with 100–150 trees and very small leaves increased training time without improving F1 or AUC; they were not used in the final model.

3.2 Implementation Issues and Resolutions:

- At first, we chose 80/20 stratified split. We worked on with it, but the result was not satisfactory since the test set is quite less for 1000 data. So, we used a 70/30 stratified split.
- We prioritized 10-Fold CV over Out-of-Bag (OOB) error. While OOB is faster, CV provides a more principled comparison for Logistic Regression, ensuring both models were evaluated on the same data partitions.

- Initial experiments focused on raw accuracy, which was misleadingly high even for poorly calibrated models. After observing poor recall for discomfort in some configurations, the evaluation was refocused on Recall and F1 as primary metrics.
- Training and prediction for LR were extremely fast, so a fixed random seed (rng) and consistent partitioning logic were used to ensure reproducible train/test splits and CV folds.
- We tested forests with 150+ trees. These increased training time significantly (~2.5s) with no measurable gain in AUC, leading us to favor the leaner 20-tree model for computational efficiency.

3.3 Baseline & Balanced Model Comparison (10-Fold CV):

Model	CV AUC	CV Error	CV F1
LR Baseline	0.5126	0.0000	0.9457
LR Weighted	0.9437	0.1528	0.9089
RF Baseline	0.9989	0.0000	0.9923
RF Weighted	0.9997	0.0000	0.9983

Table 2: Baseline & Balanced Model Comparison (10-Fold CV)

-The Problem: The LR Baseline shows a 0% error rate but an AUC of only 0.51. This is a major "red flag"—it means the model had zero ability to actually distinguish between classes and was just exploiting the data imbalance to look accurate.

-The Solution: Once we applied frequency-based weighting, the LR model's AUC jumped from 0.51 to 0.94. Although the error rate increased slightly, the model finally started "learning" the actual patterns of the minority class. This taught us that in healthcare datasets, metrics like Accuracy and F1 can be dangerously misleading if they aren't cross-checked against AUC and Recall.

3.4 Key lessons learned:

- Recall is the clinical priority. In healthcare monitoring, a False Negative (missing discomfort) is a much higher risk than a False Positive. In our project, the RF model's 1.00 Recall makes it the superior choice for patient safety.

- Accuracy is misleading**.** From this project we came to know that in imbalanced medical data, F1-score and AUC are the only reliable measures of success.

- Cross validation is crucial.

- Although we used class weights for handling class imbalance, which is effective way to reduce bias towards majority class without changing the clinical dataset (which was our motive), this strategy is not supported for all applications. In this health-related project, our priority was patient safety, so preserving the true outcome distribution was important. In general, however, the most suitable class imbalance-handling method depends on the specific objective, data characteristics and operational constraints of each application.

- Structured pipeline helps.