CalmSense

Continuously detect user stress using **multiple sensors** (e.g., humidity, temperature, steps, HRV, GSR) and **IoT** infrastructure. By merging **environmental** and **physiological** signals, CalmSense aims to **pinpoint** stress states and **alert** or **assist** users in real time.

Multimodal, Multi-Sensor: Stress often arises from **environmental** (high temperature, humidity), **physiological** (heart rate, GSR changes), and **behavioral** (activity level) factors. CalmSense **fuses** these streams for better predictive power than any single sensor alone.

Use Case: Healthcare settings (monitoring patients' stress), corporate wellness programs (employee stress tracking), or personal wearable apps (daily stress insights).

IoT Architecture & Data Flow

1. **Sensors**: Deployed in **wearable** devices (e.g., wristbands with HRV/GSR), **ambient** sensors (temperature/humidity modules), plus **activity trackers** (step count).
2. **Data Transmission**: Typically, over **Bluetooth Low Energy (BLE)** or **Wi-Fi** to a **gateway** or **cloud** service. Data is **timestamped** and **aggregated** in a single pipeline.
3. **Data Pipeline** (On Edge or Cloud): Receives raw sensor data, runs the ML pipeline described below, and **outputs** an updated stress probability or class label.
4. **User Feedback**: The system can push **alerts** (notifications) when stress crosses a threshold, or store data for **further analysis** in a web-based dashboard.

Core Tools & Libraries

Python & Streamlit

1. **Streamlit** for an interactive UI (tabs: data prep, model training, interpretation).
2. Python ecosystem for robust data manipulation.

3. **Scikit-learn**: Outlier detection, random forests, partial dependence plots, permutation importance.
4. **XGBoost**: Gradient boosting for tabular classification.
5. **Imbalanced-Learn (SMOTE)**: Corrects class imbalance by oversampling minority classes.
6. **SHAP**: Local explanation of predictions—**vital** for understanding stress triggers.
7. **Plotly**: Rich, interactive visualizations (e.g., confusion matrices).

Data Handling & Pipeline Details

Data Loading & Renaming

1. Reads a CSV (e.g., Stress-Lysis.csv) that has columns like Humidity → humidity, Temperature → temperature, Step_count → steps, Stress_Level → stress_label.
2. Missing or NaN rows can be dropped or imputed. This ensures uniform numeric arrays for ML.

Noise Injection (Optional)

A small **Gaussian noise** is added to humidity or temperature if the dataset appears too clean, simulating real sensor drift. This avoids overly optimistic results.

Outlier Removal (Hybrid Approach)

- **IsolationForest**: Identifies unusual patterns in multi-dimensional numeric data (contamination ~2%).
- **IQR-based**: For each numeric column, flags points beyond Q1 - 1.5*IQR or Q3 + 1.5*IQR.
- Any row flagged by **either** method is removed. This robust approach eliminates both global outliers (IQR) and suspicious high-dimensional anomalies (IsolationForest).

Missing Data Imputation

A **SimpleImputer** (median or mean) fills any leftover gaps. This maintains dataset consistency without discarding too many samples.

SMOTE for Imbalance

If "High Stress" samples are fewer than "Low/Medium Stress," **SMOTE** oversamples the minority class in the **training set** to improve recall for rare classes.

Feature Engineering

- An interaction feature, e.g. hum_temp_interact = humidity * temperature, captures synergy between environment variables.
- Additional signals (like heart_rate_variability / steps) can be integrated similarly, broadening the ML model's scope.

Machine Learning Workflow

Train/Test Split & Scaling

- Typically, an 80/20 or user-chosen ratio.
- **StandardScaler** standardizes numeric columns (mean ~ 0, variance ~1). This is especially beneficial for SVM's RBF kernel.

Models

- **RandomForest** (robust to outliers, interpretable feature importances).
- **XGBoost** (gradient boosting, can leverage GPUs, often excels in tabular tasks).
- **SVM** (powerful for smaller, well-scaled data, can capture complex decision boundaries).

Evaluation

- Predictions on the **hold-out** test set produce **accuracy**, **precision**, **recall**, **F1**, plus a **confusion matrix**.
- Each model's performance is displayed in a **Streamlit** tab. The user picks the best or keeps them all for further comparison.

Interpretability & Explanation

Permutation Importance

For each feature (like temperature or steps), it randomly permutes its values and measures the **drop** in model accuracy, yielding a **ranking** that shows which sensors matter most.

Partial Dependence Plots (Multi-Class)

- For a chosen feature, partial dependence depicts how predicted stress changes when that feature alone varies (others averaged).
- If multi-class (e.g. Low/Medium/High), the pipeline explicitly **selects the target class** to avoid the "target must be specified" error. This highlights, say, how rising temperature specifically impacts the "High Stress" prediction probability.

SHAP (SHapley Additive exPlanations)

- Explains each prediction by distributing the classification "credit" among features. For multi-class, the pipeline either picks a class (like High Stress) or averages across classes.
- Beeswarm or bar plots reveal how each sensor reading (humidity, temperature, steps) influences the prediction for a **specific** instance, enabling **fine-grained** diagnosis of stress triggers.

Real-Time Simulation

- After selecting the best model, a **simulation** step processes the last ~10 rows with a small **time delay** (e.g., 0.3 seconds).
- The pipeline **mimics** a **live IoT feed**, updating a table or chart with the newly predicted stress label.
- This demonstrates how CalmSense **would** operate in continuous sensor environments—**moment-by-moment** stress detection.

Integration & Use Cases

IoT Deployment

- CalmSense can run **on a local edge device** (e.g., Raspberry Pi with sensor data input) or in the **cloud**, receiving sensor streams over MQTT or HTTP.
- The final classification results can be displayed in a **web dashboard** or a **mobile app**.

Healthcare / Workplace

- In clinics, staff might watch real-time stress curves for patients or staff. Alarms could trigger if stress remains high for too long.
- In offices, employees could view personal daily stress patterns and adopt timely stress management strategies.

Scalability

Additional sensors (like GSR, EEG) are integrated with minimal changes, simply by adding new columns and features in the data pipeline. The ML steps remain the same.

Conclusions & Future Work

CalmSense merges **multi-sensor** data (environmental + physiological + activity) into a **unified ML pipeline** that effectively **detects** and **explains** stress in near-real-time. The pipeline is **robust** (handling outliers, missing values, and class imbalance) and **transparent** (via permutation importance, partial dependence, SHAP).

Future expansions could include

1. **Hyperparameter Tuning** (Bayesian or grid) for each model.
2. **Ensemble Stacking** (e.g., combining RF + XGB + SVM).
3. **Edge Optimization** (model pruning, quantization) for on-device inference.
4. **Continuous / Online Learning**, adjusting the model as new sensor data accumulates.