Forecasting Tic Episodes

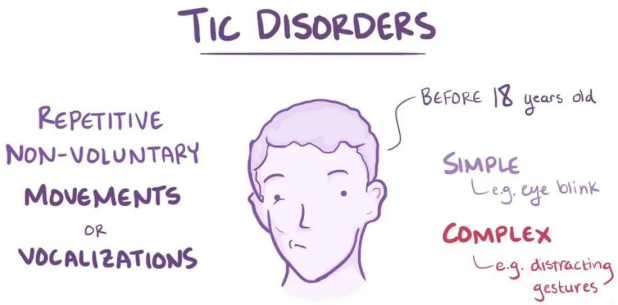
Machine Learning Approaches for Proactive Intervention

Outline

1. **Introduction**
    a. Motivation
    b. Problem Statement
    c. Related Work
2. **Data**
    a. Sources
    b. Processing
    c. Exploratory Analysis
3. **Methods**
4. **Results**
5. **Conclusion & Future Work**

INTRODUCTION

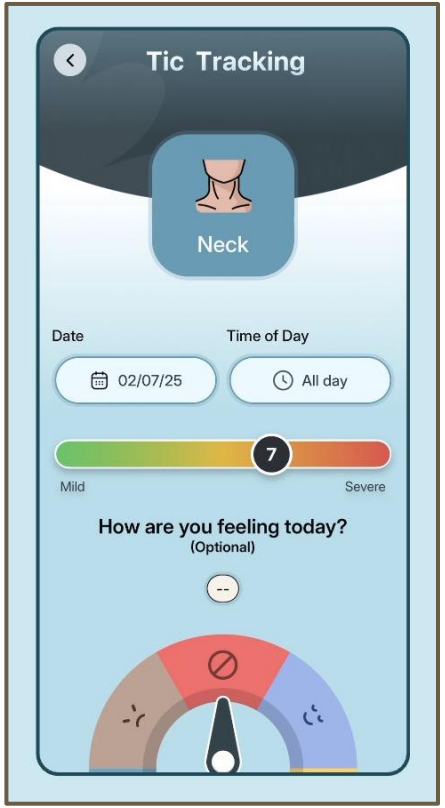
Motivation



- Tic disorders affect **millions of individuals worldwide**
- Predictive modeling of tic episode patterns remains **underexplored**
- Limited **user-level** data documenting occurrences of tic episodes

Data Collection

With access to **proprietary data** relating to user-level tic episodes, we are in a unique position to contribute to **predictive modeling** for episode occurrence, facilitating proactive care



TicVision Mobile App

Research Question

Given the nature of the data we have access to (covered later), we formulate our problem statement as follows:

“When a tic episode occurs, can we apply machine learning on features of the episode to predict factors relating to the subsequent occurrences of high-intensity tics?”

Research Question

"When a tic episode occurs, can we apply machine learning on features of the episode to predict factors relating to the subsequent occurrences of high-intensity tics?"

We further divide this research question into 3 sub-questions:

RQ1: Can we predict the intensity of the next tic episode?

RQ2: Can we predict whether the next episode will be a high-intensity one?

RQ3: Can we use our models to understand the importance and relationships of tic episode features to future episode intensity?

Prior Work

Studies Involving Tics

- **Bernabei et al. (2010)**: Wearable accelerometer tic detection
  - Detects motor tics using a 3-axis motion sensor
  - Shows feasibility of automated measurement but no forecasting
- **Cernera et al. (2022)**: Human Tic Detector (EMG + accelerometer)
  - Uses EMG + motion sensors to classify tics vs. voluntary movement
  - High accuracy in real-world conditions; again focused on detection, not prediction

Methodologically Similar

- **Chikersal et al. (2021):** Multi-level ML with sparse self-reported data
  - Uses hierarchical + temporal models to predict depression from self-reported, irregular smartphone data
  - Involves sparse, user-specific, and heterogeneous data
- **Alaa & van der Schaar (2018):** Automated Clinical Prognostic Modeling
  - Uses ensemble models for clinical prognostic modeling with sparse, heterogeneous, and missing data
  - Combines static (demographic) and dynamic (event/timeline) features, matching our approach of user-level + temporal features

DATA

Sources

- **Self-reported tic episode data** collected through a mobile health application over a six-month period from April 26 to October 25, 2025
- 89 unique users
- 1,533 filtered tic episodes
- Non-engineered features include: age, sex, tic type, intensity, time of day, mood, trigger

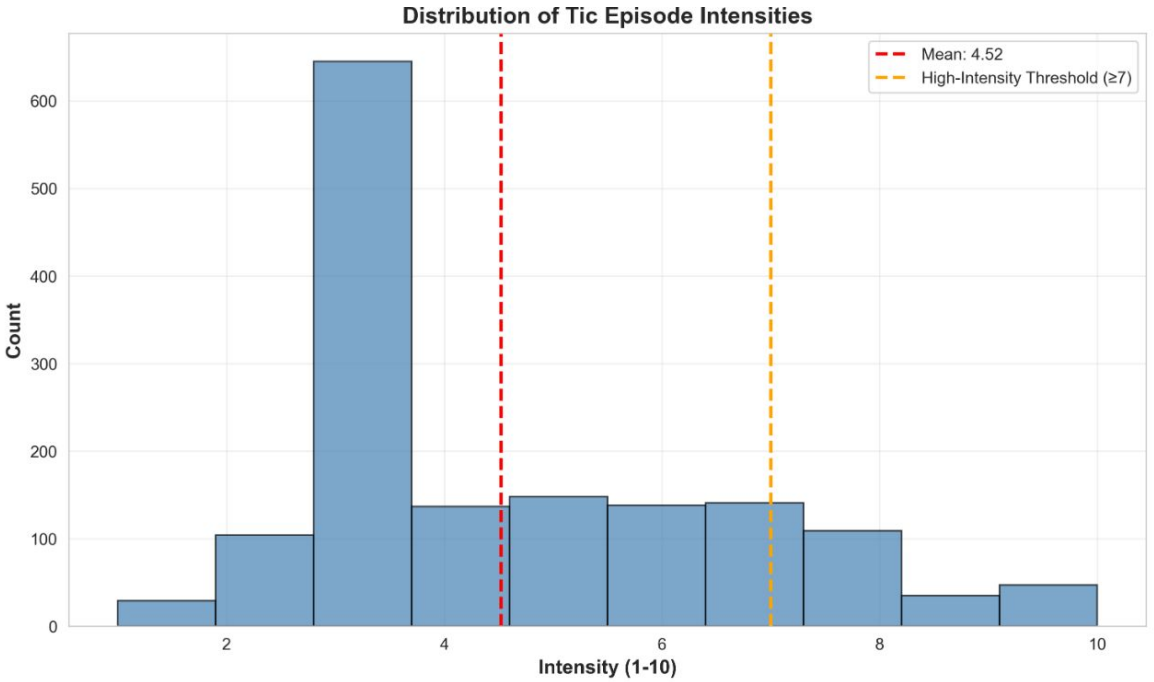
Processing

- **Filter** for samples with intensity information (only 0.2% of data excluded)
- **Filter** for duplicate entries (preserve first occurrence)
- Missing data in optional fields preserved by encoding missingness (>40% missingness for mood, trigger)
- Categorical encoding for mood, trigger, tic type
- For RQ1, we define a **high-intensity tic** episode as having intensity >=7 (on a scale of 1-10)

Exploration & Analysis

Tic Intensity distribution

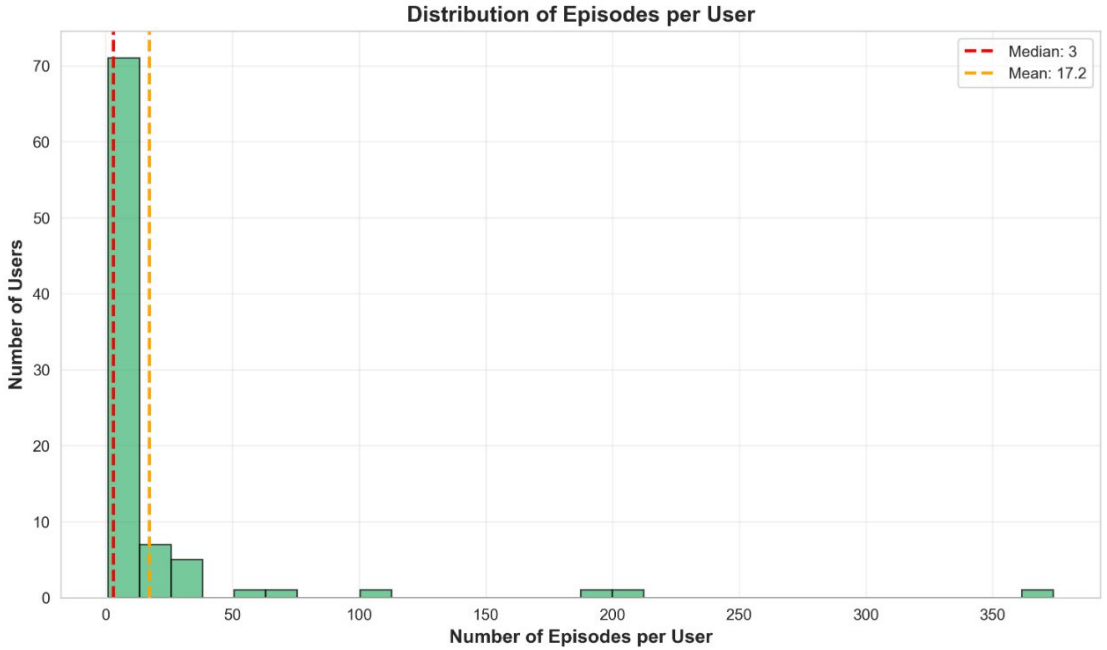
- Significant peak for intensity level 3
- Roughly bell-shaped otherwise, with slight increment at intensity level 10
- 21.7% high-intensity episodes



Exploration & Analysis

User Report distribution

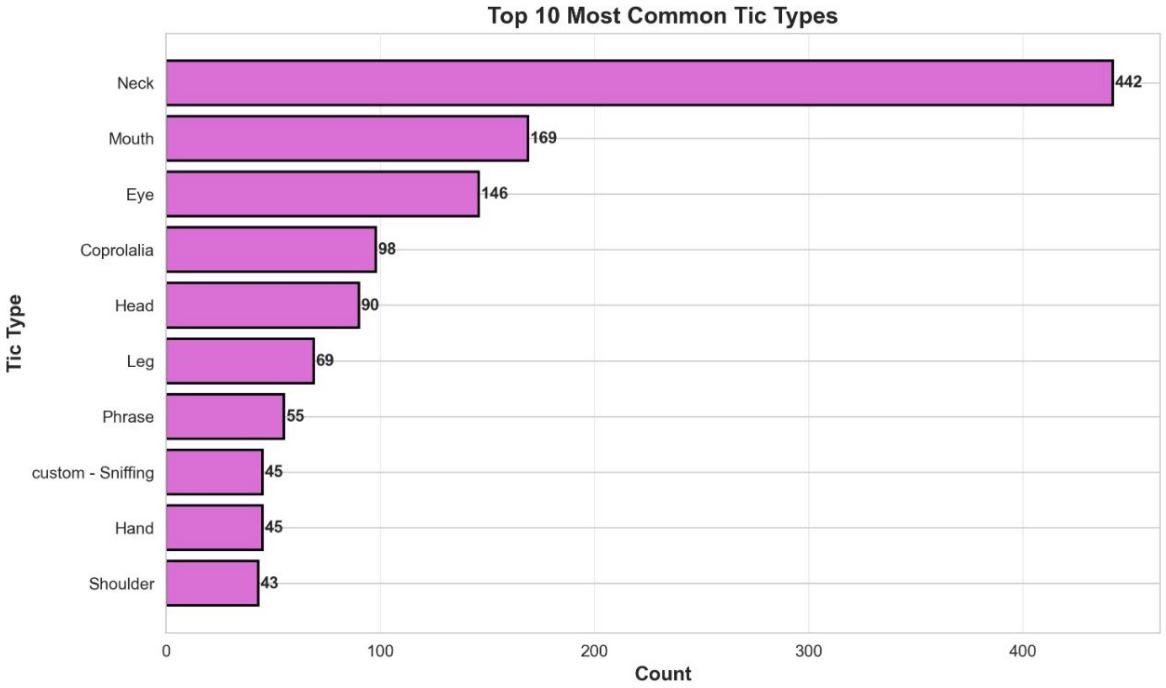
- Strong skew towards few reports
- Use user reporting information to design representative evals



Exploration & Analysis

Tic Type

- 82 types
- Some much more common; others may not provide strong signals



Feature Engineering

Besides user-reported features, we **engineered and combined** features that we believe provide useful information. These include:

- **Volatility features:** capture trends and variance in recent user-level tic episodes
- **User-level features:** capture user-specific intensity mean, variance, and high-intensity episode rate
- **Time-window statistics:** capture occurrences of tic episodes in a fixed window prior to the current episode

Feature Engineering

Category	Count	Example Features	Rationale
Temporal	6	hour, day_of_week, is_weekend	Circadian and weekly cycles
Sequence	9	prev_intensity_1/2/3, time_since_prev_hours	Recent episode history
Time-Window	10	window_7d_mean_intensity, window_7d_std	Weekly aggregation statistics
User-Level	5	user_mean_intensity, user_std_intensity	Individual baselines
Categorical	4	type_encoded, mood_encoded, trigger_encoded	Episode characteristics
Engineered	4	intensity_trend, volatility_7d	Computed volatility metrics
Interaction	6	mood_x_timeOfDay, trigger_x_type, weekend_x_hour	Non-linear feature combinations
Total	**44**	-	-

Methods

Models

We started with linear & logistic regression. However, since we are dealing with a **high-rate of missingness**, the best approaches are robust to missing values and can learn non-linear fits around high-dimensional categorical features. We use the following ensemble baselines:

- Random forests
- XGBoost (sequential boosting)
- LightGBM (memory-optimized sequential boosting)

Metrics

RQ1: To evaluate model performance on next-intensity prediction, we use MAE (mean absolute error) as our primary metric

RQ2: To evaluate model performance on high-intensity episode classification for the next episode, we use precision, recall, and the F-1 score as our primary metrics

Evaluation Strategy

We test two evaluation strategies, representing 2 different prediction utilities:

1. **User-grouped Validation:** generalization to new patients never seen during training, answering: "Can the model predict for patients it has never encountered?"
2. **Temporal-grouped Validation:** generalization to future time periods for known patients, answering: "Can the model predict future episodes for patients it has already learned from?"

Evaluation Strategy

1. **User-grouped Validation:** train-test split with 80% of users in train, 20% in test
2. **Temporal-grouped Validation:** train-test split with first ~70% of timestamps in the train set, and the rest in the test set. Users appear in *both the train and test set.*

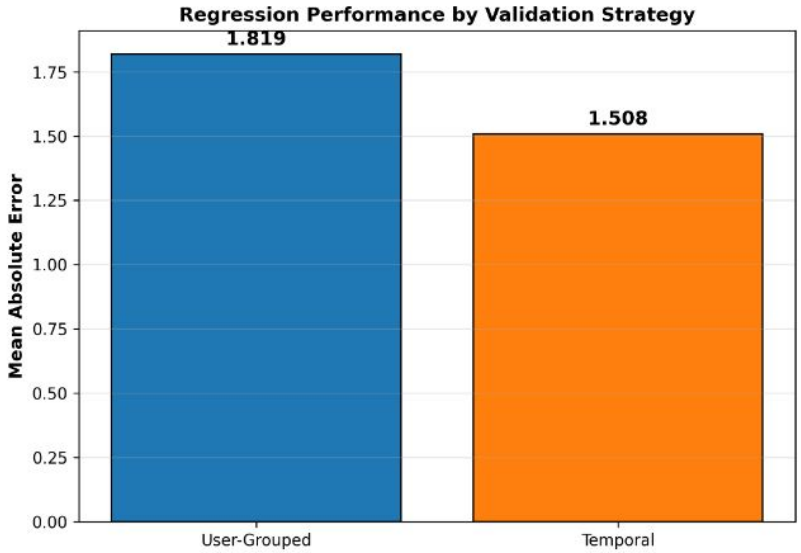
Hyperparameter Search Strategy

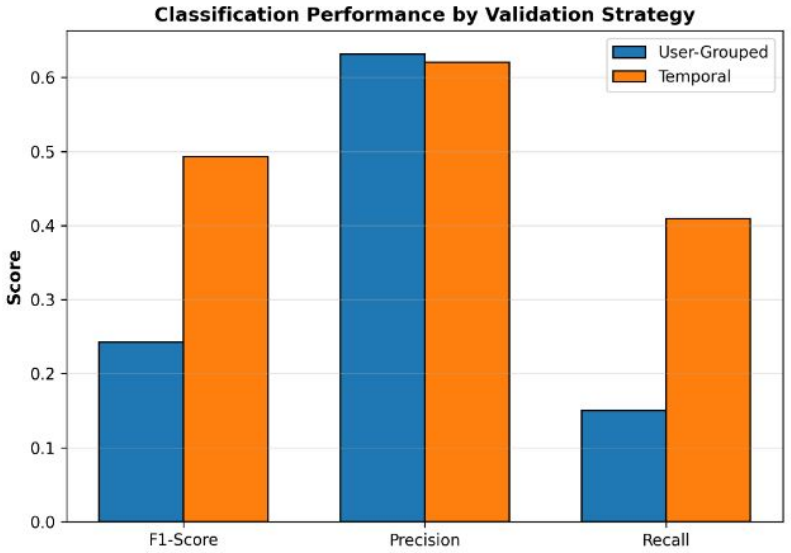
- **Hyperparameters** included num_estimators, min_sample_split, min_sample_leaf, max_depth, and L1/L2 penalties for boosting
- We employed **RandomizedSearchCV from scikit-learn**, which samples hyperparameter configurations from specified distributions rather than exhaustively evaluating all combinations as in grid search
- Hyperparameters evaluated on **3-fold cross-validation** with user-grouped data (only one possible split for temporal data)

Results

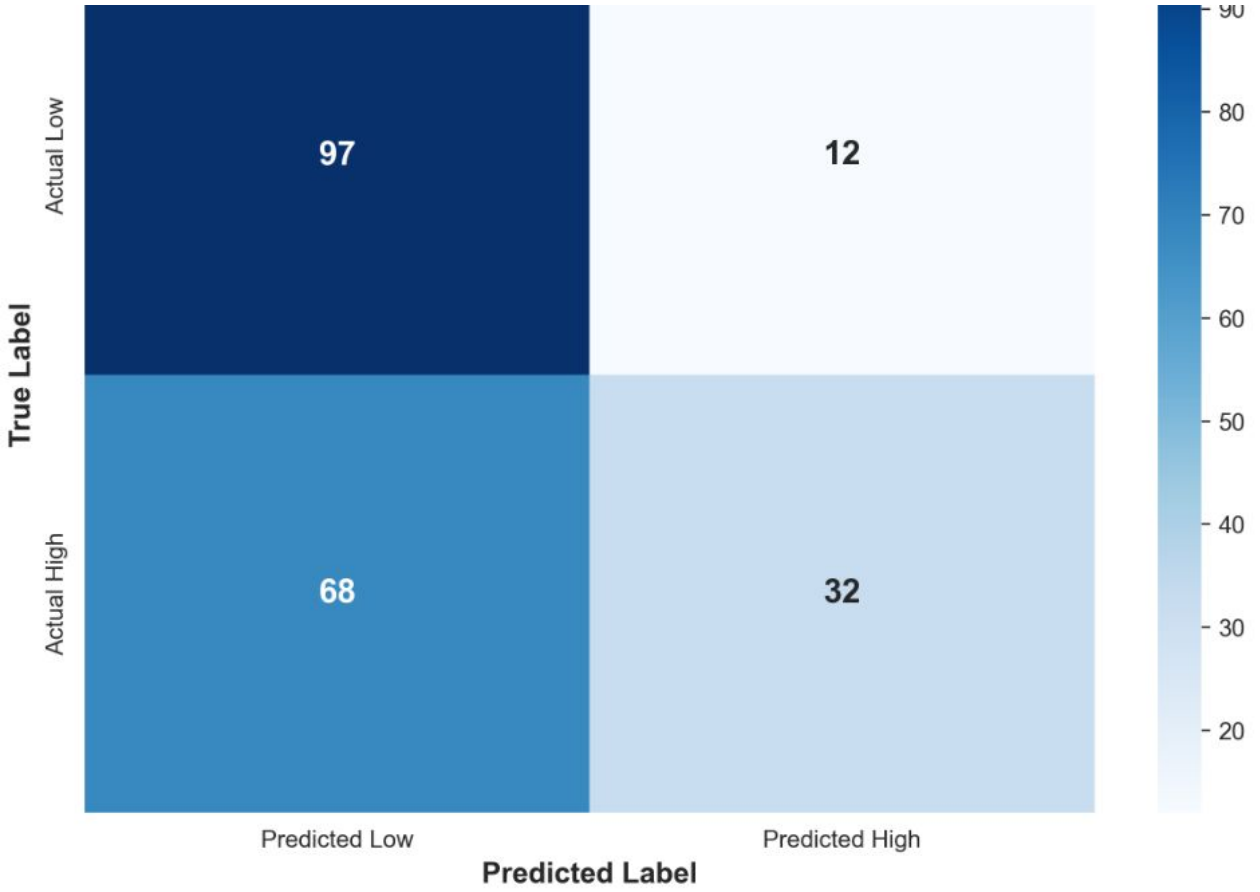
Results by Strategy

Best Model: Random Forest for regression, XGBoost for classification (0.5 threshold)





Confusion Matrix



Significance Testing

To test whether the results produced are significant, we compared regression results to simply predicting the global/user-level mean

- Random Forest achieves **26.8%** reduction in MAE over global baseline
- Random Forest achieves **24.3%** reduction in MAE over user-level baseline

Results by Strategy

Optimal Random Forest Configs (Regression):

- n_estimators = 100
- max_depth = 5
- min_samples_split = 2
- min_samples_leaf = 1
- max_features = 1.0

Optimal XGBoost Configs (Classification):

- n_estimators = 100
- max_depth = 10
- learning_rate = 0.1
- subsample = 1.0
- colsample_bytree = 0.8
- reg_alpha = 0.0
- reg_lambda = 0.1

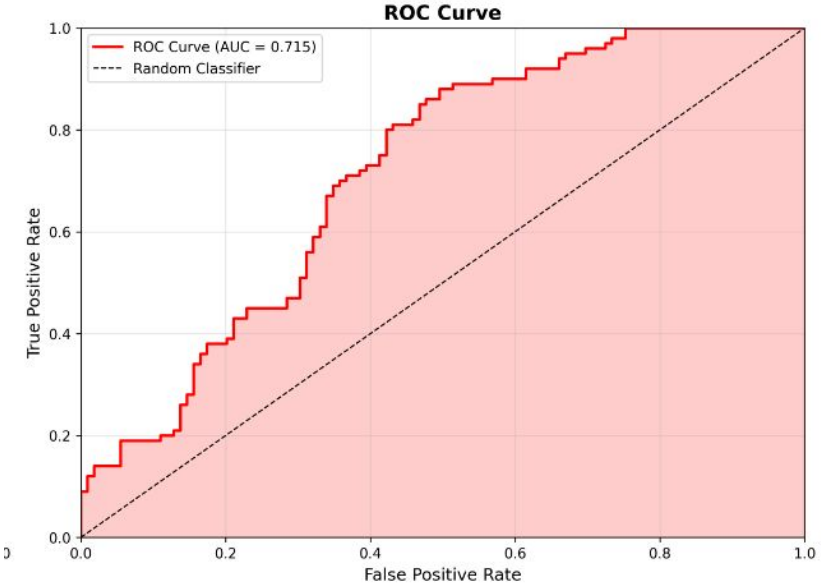
Results by Strategy

Implications:

- Better performance on **temporal split** for both classification and regression tasks
- **Prior user data** is important in predicting next-episode intensity
- Users have distinct **profiles**

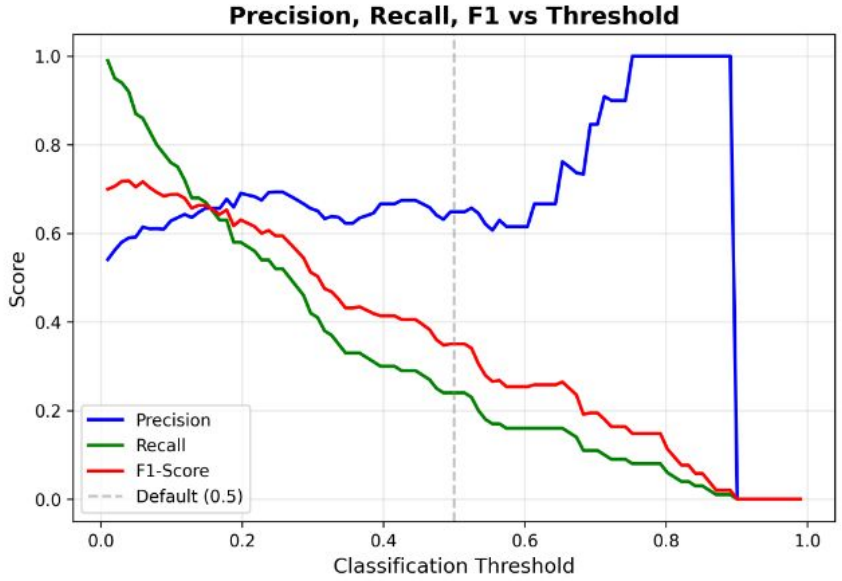
Threshold Optimization for Classification

- For prior results, we used a classification threshold of 0.5
- To evaluate if there was scope for performance improvement by adjusting this threshold, we plotted the ROC Curve
- Resulting AUC: **0.715**



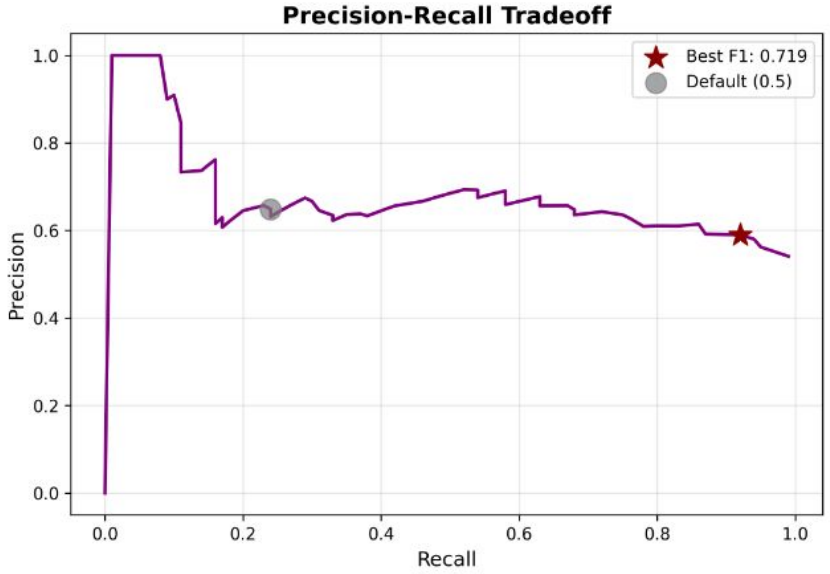
Threshold Optimization for Classification

- AUC (0.715) being significantly better than F-1 (0.49) indicates scope for performance improvement via thresholding; the model is an effective **ranker**

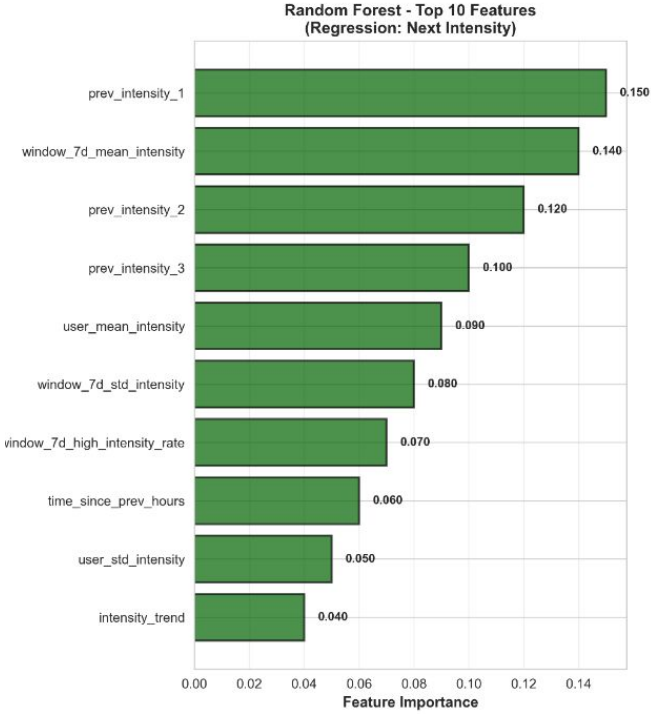


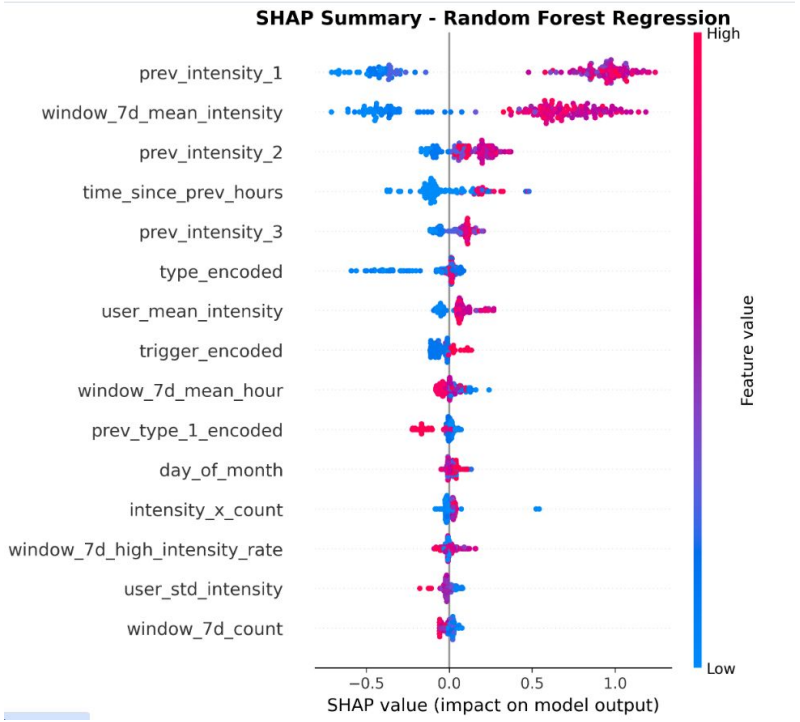
Threshold Optimization for Classification

- Very low threshold (0.04) yielded best performance
- Increased recall dramatically without affecting precision
- Suggests that even very weak signals from episode features indicate high intensity episodes



Feature Importance: Regression

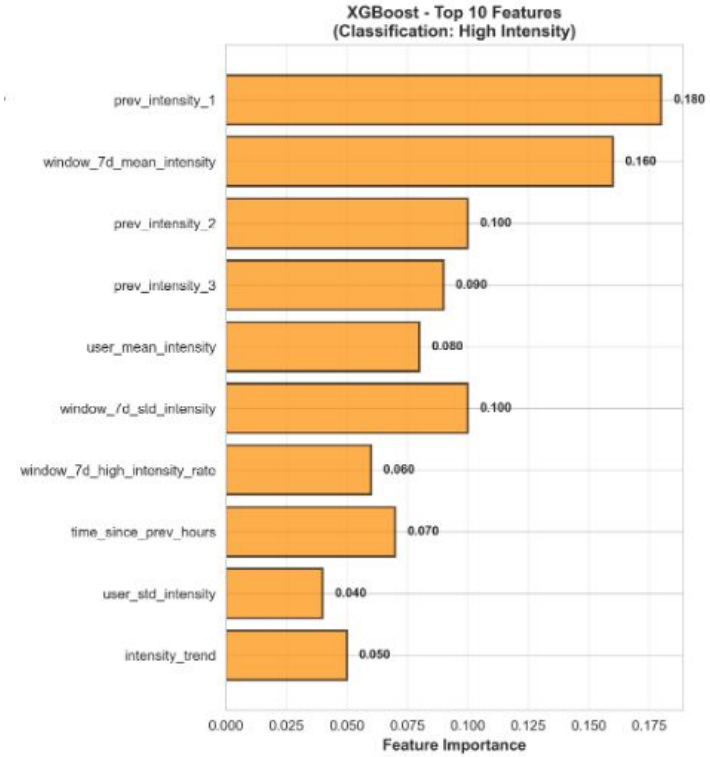


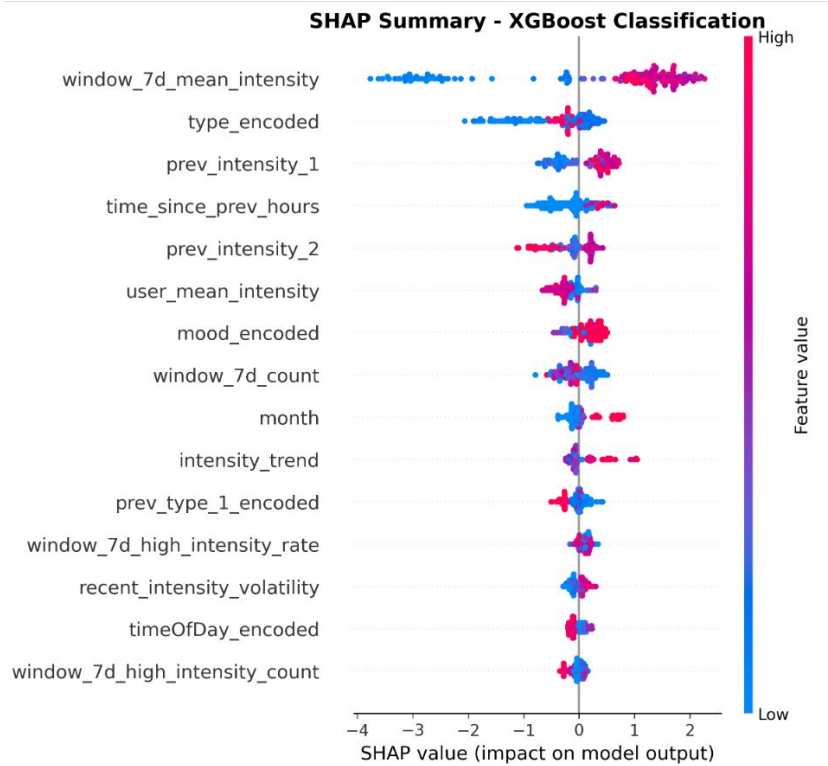


Feature Importance: Regression

- Agreement between feature importance and SHAP scores
- Previous intensities and rolling mean provide most information, generally push model predictions up when high, down when low
- Suggests **clustering** in episode intensity

Feature Importance: Classification





Feature Importance: Classification

- Similar to regression results: intensity clustering
- **Type** seems to be a stronger predictor in SHAP analysis
- User-specific mean, variance is important
- Both **user** and **temporal** clustering

Conclusions & Future Work

Key Conclusions

- Random Forest approaches are effective predictors of **next tic episode intensity** (1.51 MAE for temporal evaluation, 1.82 for user-based)
  - **Clinical utility:** preparation specific to intensity of next episode
- XGBoost models are effective for classifying **if the next tic episode will be high intensity** (0.72 F-1 for 0.04 threshold)
  - **Clinical utility:** proactive preparation for impending high-intensity episode
- High feature importance for previous intensities, previous week's episodes, and user-level features suggest **temporal** and **user-level clustering** in tic episode intensity

Future Work

- We briefly looked at next-high-intensity episode prediction, however very **high target variance** resulted in results with questionable utility
  - Target transformations, different modeling approaches could make this a feasible problem
- User-level **online learning**
  - User-specific data could be used to fine-tune generic models for more personalized, accurate predictions that leverage user-level clustering in intensities
- **Deployment** of models
  - Development of a warning/intensity prediction system that can be deployed in a patient-doctor interface to promote proactive care

Thank You!