Multivariate Gait Data

Chuyi Hu
Brown University
Github Repository

December 15, 2024

1 Introduction

1.1 Motivation

The goal of this project is to predict whether a subject is wearing a brace (knee or ankle) based on their gait data, which includes joint angles collected during walking cycles. This research is motivated by its potential applications in clinical settings, such as tracking recovery and checking how well treatments are working. Additionally, in public spaces like airports or malls, detecting if someone is wearing a brace could allow staff to offer assistance or services to those who may need extra help. Automating this process makes gait analysis faster and more useful in both clinical and public settings.

1.2 Dataset

The dataset contains multivariate gait data [1], including joint angle measurements collected from 10 healthy subjects walking under three conditions: unbraced, knee-braced, and ankle-braced. For each subject, the data consists of 10 gait cycles across two legs (left and right), three joints (ankle, knee, and hip), and 101 time points per cycle. To prepare the data for analysis, I first collapsed rows with the same subject, leg, joint, and replication into one row, transforming the 101 time points into individual columns, with each column representing the joint angle. I then created a new binary target variable, where condition = 1 (unbraced) is labeled as brace = 0 (not wearing a brace), and conditions = 2 or 3 (knee- or ankle-braced) are labeled as brace = 1 (wearing a brace).

The previous work [2] of this dataset analyzes cyclic biomechanical data using statistical methods like SSANOVA to compare differences between groups or conditions. Unlike my project, it focuses on understanding group differences rather than building a predictive model to classify brace-wearing conditions. Therefore, I cannot compare my model's predictive power with previous work.

2 EDA

In the EDA, I analyzed the balance of the target variable (brace) and studied joint angle patterns across the gait cycle for braced and unbraced conditions. I created visualizations like average joint angles over time, variability in gait cycles, and the distributions of categorical features such as leg, joint, and subject. This helped me understand the data and prepare for modeling.

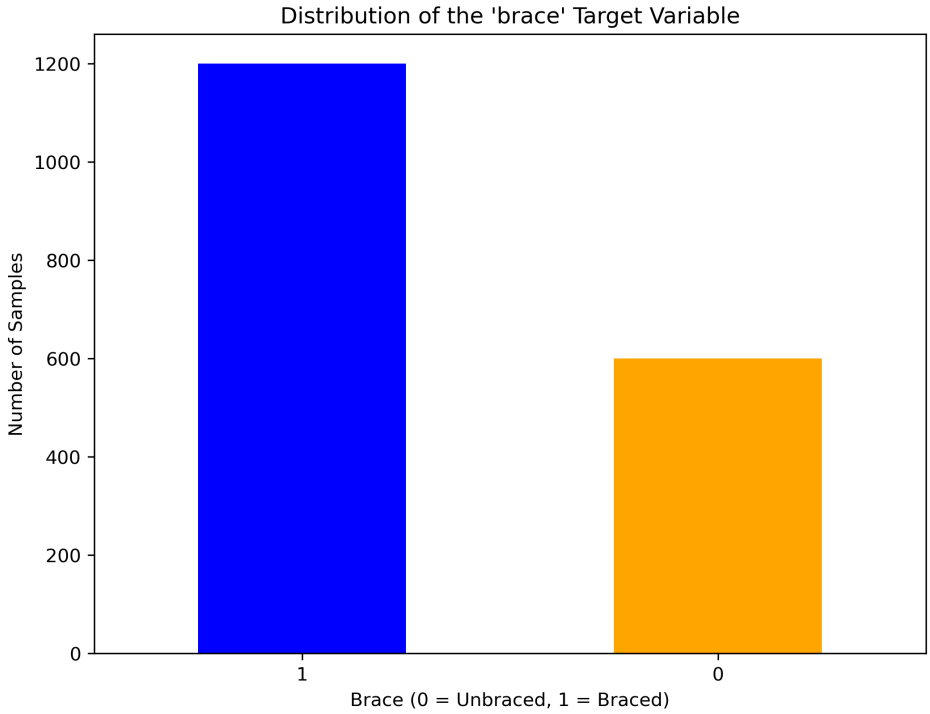
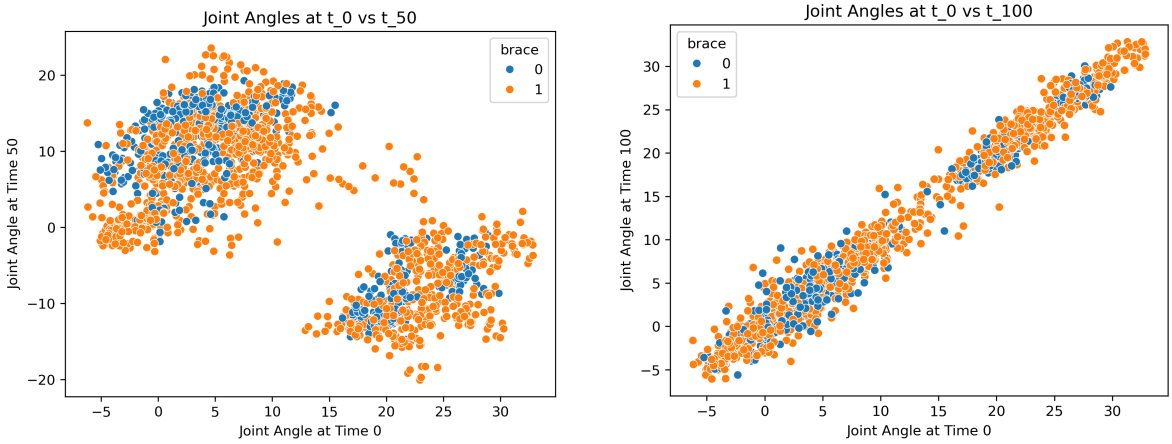


Figure 1: Distribution of the Target Variable 'brace'.

This plot shows the distribution of samples between "braced" (1) and "unbraced" (0) categories, with the "braced" group having slightly more samples than the "unbraced" group. However, the difference is not significant enough to consider the dataset imbalanced, so no special handling for class imbalance is required during model training.



(a) t_0 vs t_50

(b) t_0 vs t_100

Figure 2: Joint Angles at Specific Time Points by Bracing Condition.

These two plots show the distribution of joint angles between different time points. The first plot (at time 0 and time 50) reveals two distinct clusters that suggest variability in movement patterns within the gait cycles. In contrast, the second plot (at time 0 and time 100) displays a strong linear relationship, indicating consistent progression in movement throughout the gait cycle for both braced and unbraced conditions.

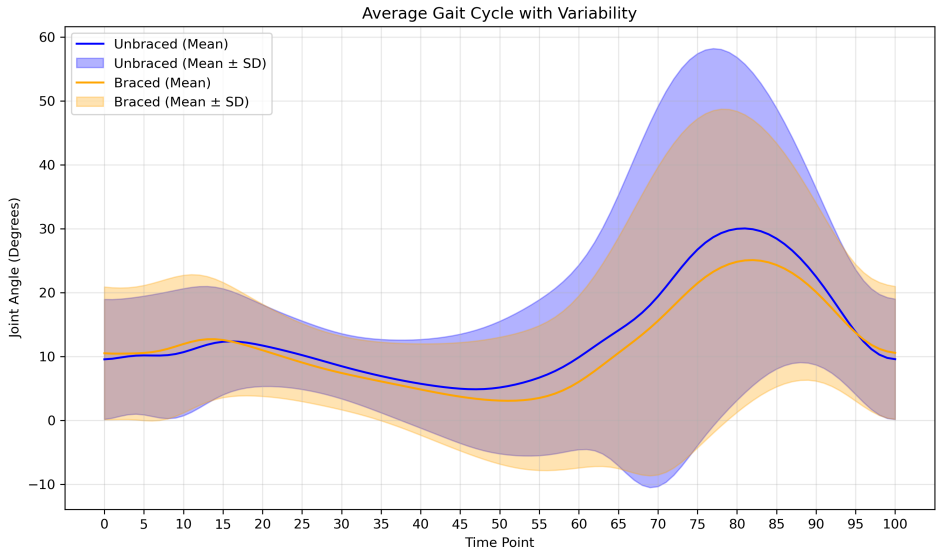


Figure 3: Average Gait Cycle with Variability (mean ± sd) for Braced and Unbraced Conditions.

This plot shows the mean joint angles across the gait cycle for subjects in braced and unbraced conditions, with shaded regions representing 1 standard deviation around the mean. The average joint angle for unbraced subjects (blue line) is consistently higher compared to braced subjects (orange line). The shaded regions indicate greater variability in the unbraced condition, particularly during the middle and later phases of the gait cycle. This suggests that wearing a brace reduces both the range of motion and the variability in gait patterns, likely providing additional stability.

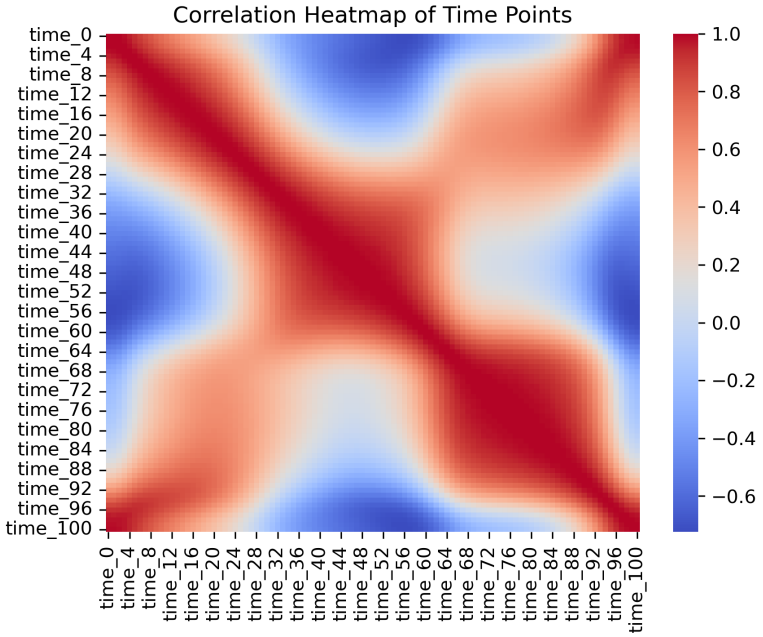


Figure 4: Heatmap of correlations between time points (joint angles).

This heatmap shows the correlation between joint angles recorded at different time points across the gait cycle. Strong positive correlations (red regions) are observed between adjacent time points, indicating that joint angles at nearby time steps are highly similar. In contrast, negative correlations (blue regions) appear between early and late phases of the gait cycle, reflecting opposing trends in joint angle movements. This pattern also highlights the cyclical nature of gait, where joint angles return to their starting positions at the end of each cycle.

3 Methods

3.1 Splitting

The dataset was split into 60% training, 20% validation, and 20% test sets using GroupShuffleSplit. This method ensures that subjects (groups) were not shared between splits, preventing data leakage and maintaining the non-IID properties of the data, which is crucial for subject-specific gait patterns.

3.2 Preprocessing

I used OneHotEncoder to transform all the categorical variables (leg, joint, subject, and replication) into binary features, converting them into a format suitable for machine learning models. Additionally, I applied StandardScaler to all the time variables (from time_0 to time_100) to ensure each time step had a mean of 0 and a standard deviation of 1. This preprocessing step standardized the numerical features and allowed the model to handle both categorical and continuous variables effectively.

3.3 ML Pipeline

I used 5-fold cross-validation (cv=5) within the GridSearchCV pipeline to evaluate model performance. The data was split into 5 folds, where the model was trained on 4 folds and validated on the remaining fold, cycling through all folds. This approach ensures robust evaluation by averaging results across multiple splits. To assess model performance, I primarily used accuracy to evaluate model performance, as the dataset is balanced and accuracy provides a clear measure of overall correctness. I also used the F1-score as a supporting metric to account for precision and recall, ensuring a balanced evaluation of the model's classification performance.

3.4 ML Algorithms and Parameters

I explored four supervised machine learning algorithms for this project: Logistic Regression, Random Forest Classifier, Support Vector Classifier (SVC), and XGBoost. Each algorithm was chosen for its unique strengths, with Logistic Regression serving as a baseline linear model, while Random Forest, SVC, and XGBoost provided non-linear capabilities to better capture complex patterns in the data. The table below shows the ML algorithms and the hyperparameters I tuned for each algorithms, these hyperparameter choices were systematically tested using GridSearchCV to identify the best-performing configuration for each model.

Algorithm	**Tuned Hyperparameters**
Logistic Regression	penalty: ['l1', 'l2'], C: [0.01, 0.1, 1, 10, 100]
Random Forest	max_depth: [1, 3, 10, 30, 100], max_features: [0.25, 0.5, 0.75, 1.0]
SVC	'C': [0.1, 1, 10, 100, 1000], 'gamma': [0.001, 0.01, 0.1, 1, 10, 100]
XGBoost	'max_depth': [1, 3, 10, 30, 100], 'reg_alpha': [0e0, 1e-2, 1e-1, 1e0, 1e1, 1e2], 'reg_lambda': [0e0, 1e-2, 1e-1, 1e0, 1e1, 1e2]

Table 1: ML Algorithms and Tuned Hyperparameters

3.5 Considerations

Several considerations were taken into account while building the machine learning pipeline to ensure reliability and fairness.

- GroupShuffleSplit was used to split the data into training, validation, and test sets, ensuring that subjects (groups) were not shared across splits to prevent data leakage.
- For preprocessing, StandardScaler was applied to the time-related features to ensure all time points were on the same scale, while categorical variables were transformed using OneHotEncoder for proper model input.
- During model evaluation, 5-fold cross-validation was implemented to minimize bias from a single train-test split and to provide robust performance metrics.
- For non-deterministic models like Random Forest and XGBoost, I accounted for randomness by running the models multiple times with different random seeds to observe performance consistency.
- Finally, I used accuracy as the primary metric due to the balanced dataset and included the F1-score to ensure precision and recall were equally considered.

These steps collectively ensured a robust, fair, and interpretable evaluation of all machine learning models.

4 Results

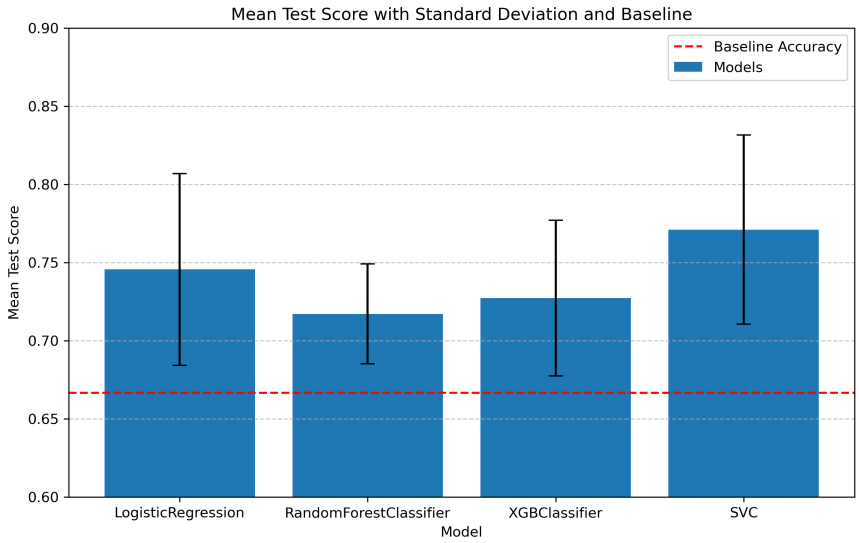
4.1 ML Model Performance

I evaluated each model using five random states (0, 2, 4, 8, 42) to ensure robust performance. For each random state, I tuned hyperparameters using GridSearchCV and selected the best parameter set for each model and evaluated them on the test set, resulting in 20 models (4 models $\times$ 5 random states). Finally, I calculated the mean and standard deviation of the evaluation metrics (accuracy and F1-score) to summarize the results, ensuring reliable and unbiased performance assessment across different splits. This table shows that the SVC model achieves the highest accuracy and F1 score.

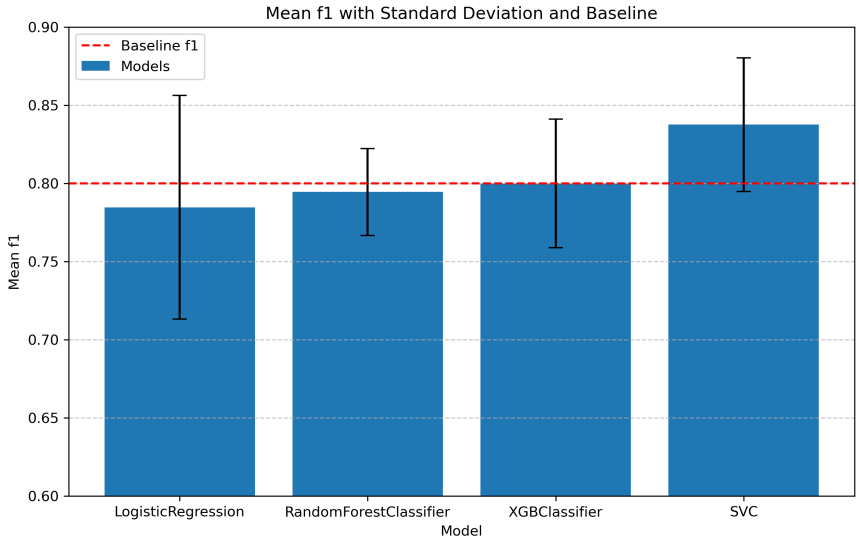
Model Name	**Test Accuracy Scores**	**Test F1 Scores**	**Mean & Std (Accuracy)**	**Mean & Std (F1)**
LogisticRegression	[0.7222, 0.8389, 0.6528, 0.7750, 0.7389]	[0.7630, 0.8858, 0.6684, 0.8228, 0.7834]	Mean: 0.7456, Std: 0.0613	Mean: 0.7847, Std: 0.0716
RandomForestClassifier	[0.6861, 0.7167, 0.7194, 0.7750, 0.6889]	[0.7621, 0.8083, 0.7960, 0.8383, 0.7676]	Mean: 0.7172, Std: 0.0320	Mean: 0.7945, Std: 0.0279
XGBClassifier	[0.6667, 0.7722, 0.6806, 0.7944, 0.7222]	[0.7436, 0.8423, 0.7658, 0.8484, 0.8000]	Mean: 0.7272, Std: 0.0498	Mean: 0.8000, Std: 0.0412
SVC	[0.7028, 0.7472, 0.7417, 0.8806, 0.7833]	[0.7838, 0.8372, 0.8114, 0.9121, 0.8434]	Mean: 0.7711, Std: 0.0604	Mean: 0.8376, Std: 0.0428

Table 2: Comparison of Model Performance Metrics Across Five Random States

4.2 Baseline



(a) Accuracy Scores



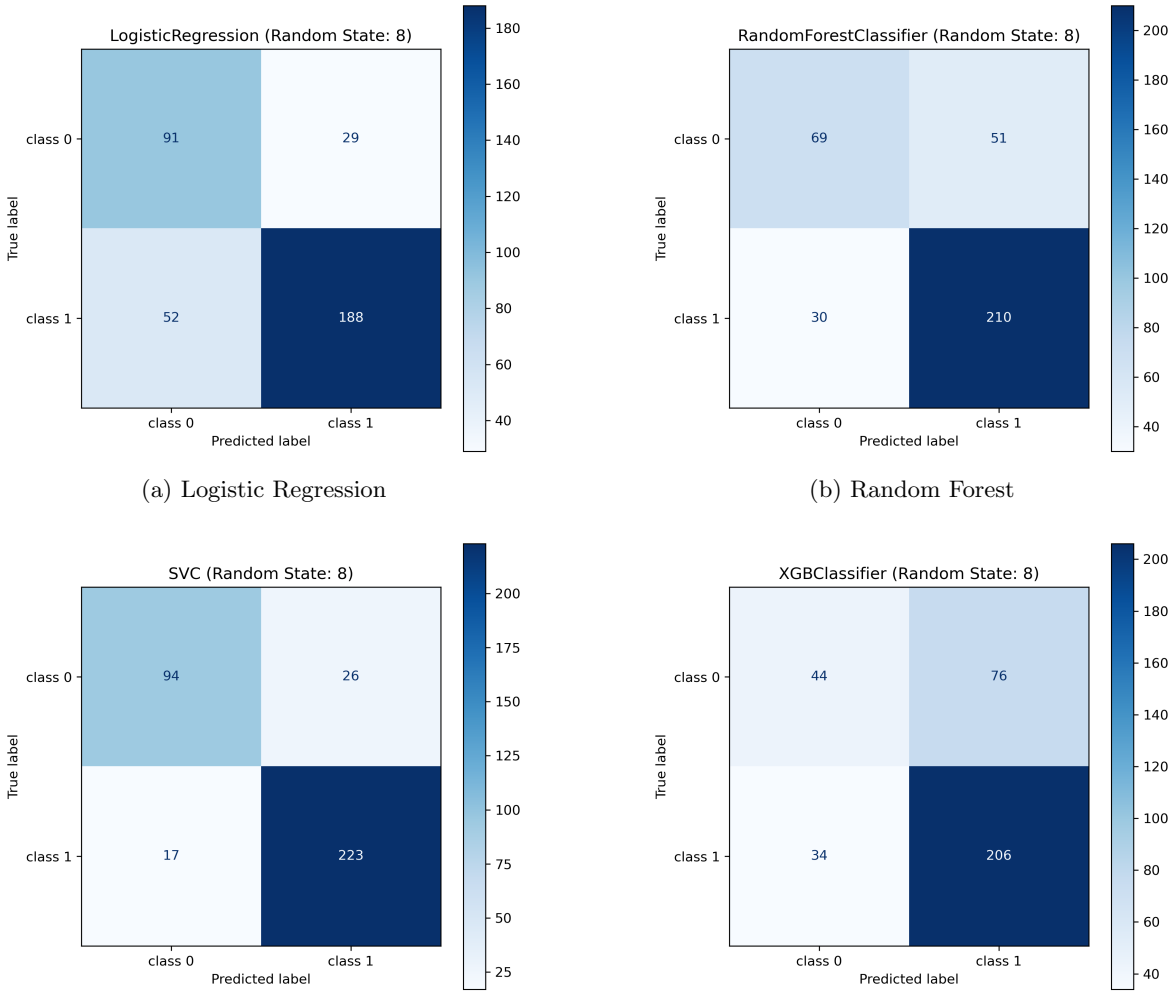
(b) F1 scores

Figure 5: Mean test scores for the four models with standard deviations compared to the baseline.

The baseline accuracy of this problem is 0.6667, and the baseline F1 score is 0.8000. These two plots compare the performance of four machine learning models using mean test accuracy and mean F1 scores with standard deviations across five random states. From the plots we can see, all models surpass the baseline accuracy, with SVC achieving the highest accuracy. In terms of the F1 score, SVC also performs the best, clearly exceeding the baseline and showing robust performance across different random states. The error bars indicate the variability, with SVC demonstrating consistent performance compared to the other models.

4.3 Confusion Matrix

I created confusion matrices for each model at random state = 8 to evaluate their classification performance by analyzing the number of true positive, true negative, false positive, and false negative predictions. The confusion matrix is a useful tool for visualizing the performance of classification models, particularly in binary classification tasks, as it allows me to assess how well the models identify both classes and pinpoint specific misclassifications.



(a) Logistic Regression

(b) Random Forest

(c) SVC

(d) XGBoost

Figure 6: Confusion Matrices for Different Models

Model	Recall	Precision
Logistic Regression	0.78	0.87
RandomForestClassifier	0.875	0.80
XGBClassifier	0.86	0.73
SVC	0.93	0.90

Table 3: Model Performance: Precision and Recall

SVC outperforms all models with a recall of 0.93 and precision of 0.90, making it the most balanced and reliable model for identifying braced and unbraced instances. RandomForestClassifier and XGBClassifier achieve competitive recall scores (0.875 and 0.86), but XGBClassifier has lower precision (0.73), indicating more false positives. Logistic Regression performs reasonably well but falls short of SVC's overall performance.

4.4 Feature Importance

4.4.1 Global Feature Importance

Permutation importance measures the decrease in model performance when a feature's values are randomly shuffled, providing an estimate of its importance. Features such as time_12, time_11, and time_53 are identified as the most influential, with significant drops in accuracy when perturbed. The red dashed line represents the baseline test score, allowing for comparison against perturbed scores.

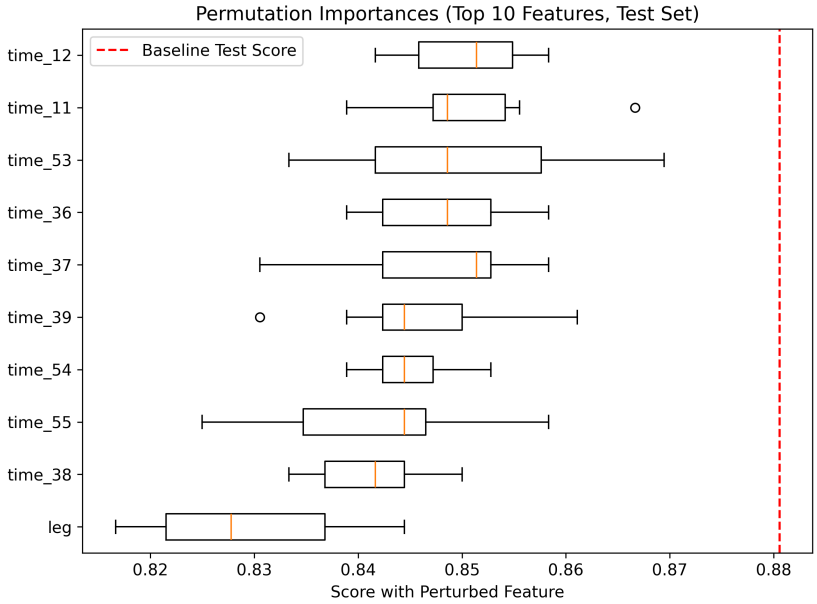


Figure 7: Top 10 features test score of the Permutation Importance with baseline.

SHAP values provide insight into how much each feature contributes to the model's predictions, with higher values indicating greater influence. Features such as std__time_12, std__time_13, and std__time_11 are the most impactful, aligning with the findings of permutation importance.

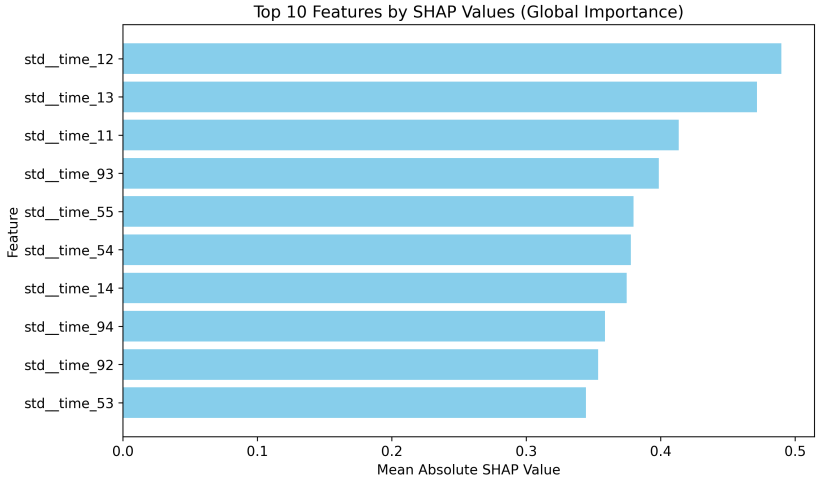


Figure 8: Top 10 Features by SHAP Values.

While both plots agree on the importance of time-based features, there are slight differences in feature rankings. Permutation importance focuses on the overall performance drop, whereas SHAP values highlight individual contributions to predictions. Together, these methods provide a comprehensive understanding of the feature importance in the model.

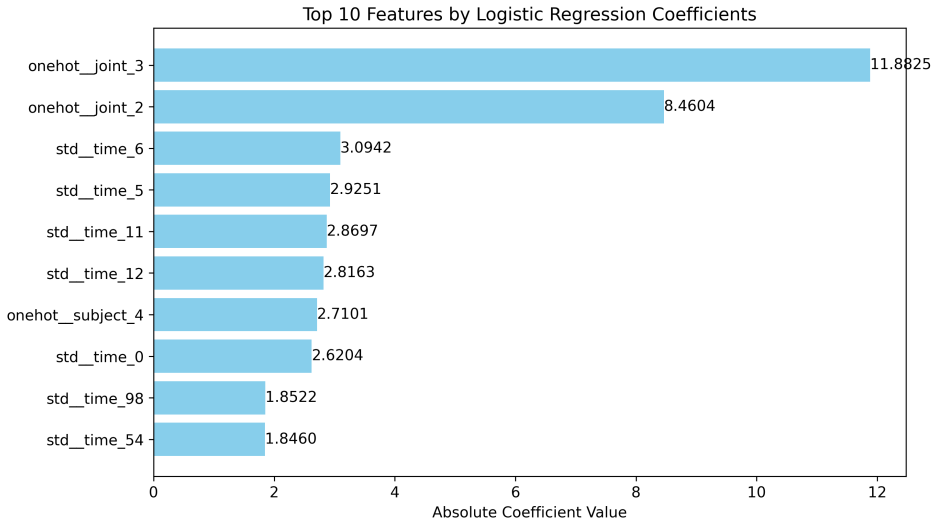
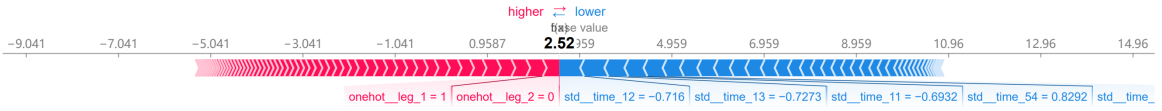


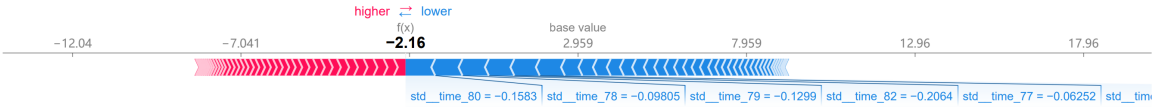
Figure 9: Top 10 Features by Logistic Regression Coefficients.

This bar plot illustrates the top 10 most important features identified by the Logistic Regression model based on their absolute coefficient values. The features onehot__joint_3 and onehot__joint_2 are the most significant predictors, with coefficients of 11.88 and 8.46, respectively. This result highlights the combined influence of both joint-specific and temporal features on the model's performance.

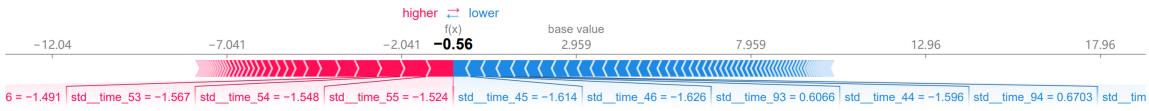
4.4.2 Local Feature Importance



(a) Data Point: 0



(b) Data Point: 100



(c) Data Point: 200

Figure 10: SHAP Force Plots for Individual Predictions at Indices 0, 100, and 200.

The three SHAP force plots illustrate the impact of individual features on the model's prediction for three data points. The red segments indicate features that push the prediction towards a higher value (positive influence), while the blue segments pull it towards a lower value (negative influence).

- Index 0: The most influential features pushing the prediction higher include onehot__leg_1 and onehot__leg_2, while std__time_12 and std__time_13 have strong negative effects. The final prediction is slightly above the base value at 2.52.

- Index 100: Here, the prediction of -2.16 is mainly influenced negatively by features like std__time_80, std__time_78, and std__time_79. The combined contributions of these features strongly lower the output from the base value.
- Index 200: This prediction is close to the base value at -0.56, with significant negative contributions from std__time_53, std__time_54, and std__time_55, while positive influences are observed from std__time_93 and std__time_94.

5 Outlook

To further improve the predictive performance and interpretability of the model, several enhancements can be explored. First, incorporating additional biomechanical features, such as force data or joint velocity, may provide richer insights into gait patterns. Second, more advanced feature engineering techniques, like time-series transformations or wavelet analysis, could capture subtle temporal dynamics. Additionally, exploring ensemble methods like stacking or voting classifiers may combine the strengths of different models for better accuracy. For interpretability, applying SHAP interaction values or advanced visualization tools can help identify feature dependencies and non-linear relationships. Finally, collecting a larger and more diverse dataset, including subjects with varying walking conditions, would ensure greater model generalizability and robustness in real-world applications.

References

[1] Nathaniel Helwig and Elizabeth Hsiao-Wecksler. Multivariate gait data. *UCI Machine Learning Repository*, 2016. [Dataset].

[2] Nathaniel E. Helwig, K Alex Shorter, Ping Ma, and Elizabeth T. Hsiao-Wecksler. Smoothing spline analysis of variance models: A new tool for the analysis of cyclic biomechanical data. *Journal of biomechanics*, 49 14:3216–3222, 2016.

Word Count: 1930