TÉCNICO LISBOA

Machine Learning

MEEC

Project Report

Authors:

David Marafuz Gaspar (106541) david.marafuz.gaspar@tecnico.ulisboa.pt
Pedro Gaspar Mónico (106626) pedro.monico@tecnico.ulisboa.pt

Group 62

2025/2026 – 1st Semester, P1

1 Part 1 - Regression with Non-linear Models

1.1 Introduction

This part of the project focuses on applying regression methods to a non-linear mapping problem using synthetic data. The goal is to build a model that predicts an expensive sensor's measurement using readings from six cheaper sensors. The relationship between the sensors and the target is non-linear, requiring non-linear regression models to achieve high predictive accuracy.

1.2 Data Analysis and Preprocessing

Initial exploratory analysis included plotting feature histograms and feature-target scatter plots which revealed diverse feature distributions and complex, non-linear relationships with the target. The correlation matrix (Figure 1) quantitatively confirmed that Sensor 2 and Sensor 6 exhibited minimal correlation with the target, suggesting limited predictive value and potential for removal.

Outlier detection was also performed using Isolation Forest and Z-score methods, which identified 6 outliers. Subsequent analysis showed that removing these outliers had a negligible impact on model performance and all samples were retained.

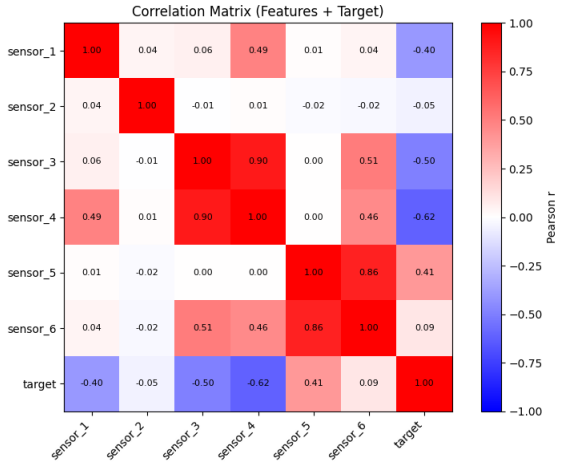

Figure 1: Sensors and Target Correlations

1.3 Methodology

To ensure fair and consistent comparison between models, the provided 700 samples were split into a training set of 500 samples and a validation set of 200 samples, maintaining a similar ratio to the actual test scenario (300/700). A fixed random state ($random\_state = 42$) was used across all experiments to ensure reproducibility.

The following models were implemented and evaluated: Polynomial Regression, KMeans with RBF transformation and Ridge Regression, Random Forest Regressor, Decision Tree Regressor, Kernel Ridge Regression.

All models (except Plynomial Regression) were evaluated using 5-fold cross-validation on the training set, with the R^2 score as the optimization metric. The best hyperparameters from cross-validation were used to train final models on the full 500-sample training set, which were then evaluated on the 200-sample validation set.

Based on initial findings regarding some sensors low correlation with the target, additional experiments were conducted by systematically removing individual sensors and sensor pairs, followed by hyperparameter re-tuning to assess the impact on model performance.

1.4 Experiments and Results

1.4.1 Model Comparison

The performance of various models using all six sensors after hyperparameter optimization via grid search with 5-fold cross-validation is summarized in Table 3.

Table 1: Comparison of regression models using all six sensors

Model	**CV R^2**	**Test R^2**
Polynomial	0.9867	0.8375
KMeans+RBF+Ridge	0.9829	0.9824
Random Forest	0.9598	0.9461
Decision Tree	0.9069	0.8622
KRR (all sensors)	0.9900	0.9853

Kernel Ridge Regression demonstrated superior performance among all models tested with the full feature set, achieving the highest cross-validation R^2 of 0.990 and test R^2 of 0.985. Given its outstanding performance, KRR was selected as the base model for subsequent feature analysis and optimization steps.

1.4.2 Feature Importance Analysis

To confirm the preliminary observation of Sensor 2's low correlation with the target variable, permutation feature importance was calculated for the KRR model using 10 repetitions. As shown in Figure 2, Sensor 2 (index 1) received a negative importance score, confirming that it was not contributing positively to the model's predictive capability and might be introducing noise. This quantitative evidence reinforced our decision to explore feature removal strategies.

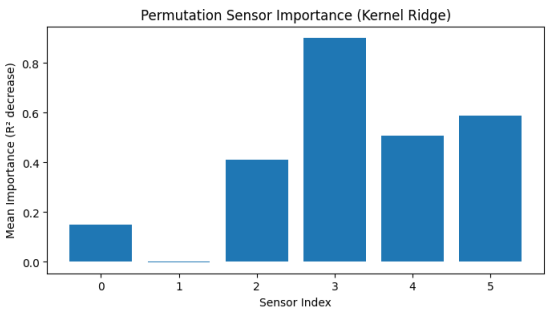

Figure 2: Permutation feature importance

1.4.3 Feature Removal Experiments

Based on the feature importance analysis and the initial correlation results, systematic feature removal experiments were conducted. We iteratively removed individual sensors and sensor pairs, followed by grid search to re-optimize hyperparameters for each feature subset.

The results, summarized in Table 2, show that removing Sensor 2 alone provided the most significant improvement, achieving the highest test R^2 score of 0.992. While removing both Sensors 1 and 2 also improved performance, the single removal of Sensor 2 yielded the optimal balance of model complexity and predictive accuracy.

Based on these results, the final model selected was Kernel Ridge Regression with Sensor 2 removed, as it achieved the highest test performance while maintaining excellent cross-validation scores.

Table 2: Impact of feature removal on KRR performance

Model Configuration	CV R^2	Test R^2
KRR (all sensors)	0.9900	0.9853
KRR (remove Sensor 2)	0.9933	0.9916
KRR (remove Sensors 1 & 2)	0.9901	0.9914

1.5 Conclusion

The final selected model for submission is a Kernel Ridge Regression with Sensor 2 removed, using hyperparameters gamma=0.14 and alpha=0.002, which emerged as the most effective model for this non-linear regression task. When evaluated on the official test set of 300 samples, this model achieved an R^2 score of 0.99197, which closely matches the validation performance of 0.9916 obtained on our held-out test set of 200 samples.

2 Part 2 - Problem 1 - Exercise Recognition

This part of the project focuses on the classification of physical therapy exercises performed by stroke survivors. Using 2D skeletal data extracted from video recordings through MediaPipe, the objective is to develop a robust machine learning model capable of distinguishing between three functional exercises: E1 (Brushing Hair), E2 (Brushing Teeth), and E5 (Hip Flexion).

2.1 Data Analysis and Preprocessing

We started by visualizing the skeleton coordinate positions to obtain an intuitive understanding of the movement patterns in our data. By plotting the mean positions of keypoints with body connections and the deviations, we could observe the characteristic poses for each exercise. This visual analysis helped identify which body parts and joints were most active during different exercises.

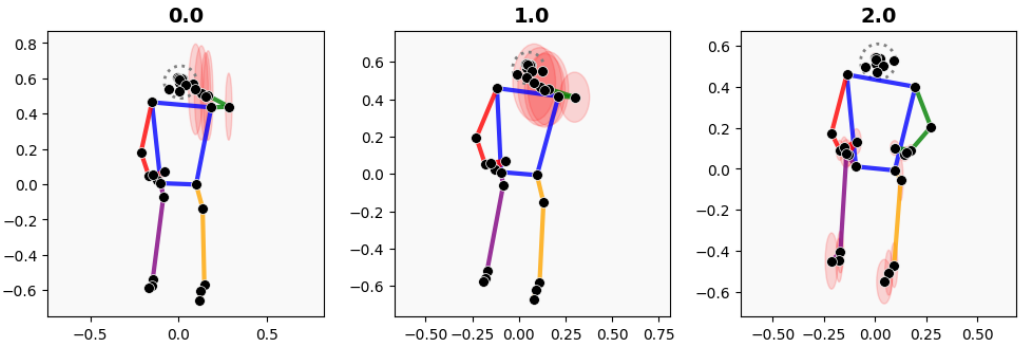

Figure 3: Visualization of the motion patterns for the three exercises performed by Patient 2

2.2 Methodology

To ensure a fair comparison between all our machine learning models, we used a consistent testing approach. For the initial evaluation phase, we set aside the data from three specific patients and used it exclusively for testing. Every model was trained without seeing this group and was then evaluated on it. This method guaranteed that any differences in performance were due to the models themselves and not to changes in the test data.

2.3 Baseline Model Evaluation

Four supervised models were evaluated for the exercise classification task: MLP, SVM, Random Forest, and XGBoost.

The Multi-Layer Perceptron (MLP) provided a strong non-linear baseline. After tuning layer sizes, activation functions, and regularization parameters, the best configuration reached an F1-score of 0.8052 using the Leave-One-Group-Out validation. The confusion matrix showed that most errors came from similar upper-body exercises (E1 and E2), highlighting a key challenge and pointing toward the need for more sophisticated feature selection to improve class separation.

The Support Vector Machine (SVM) achieved the highest performance, with an F1-score of 0.8210. While different kernel functions (linear, RBF, and polynomial) improved class separability, the model still showed confusion between the similar exercises E1 and E2.

Tree-based ensembles were also tested, Random Forest achieved 0.7607 F1-score, outperforming XGBoost's 0.7387.

Table 3: Comparison of F1-scores for exercise classification models

Model	**F1-score**
SVM	0.8210
MLP	0.8052
Random Forest	0.7607
XGBoost	0.7387

The performance evaluation clearly indicated MLP and SVM as the top-performing models, with F1-scores of 0.8052 and 0.8210 respectively. This close competitive performance justified their selection for focused feature engineering in subsequent development stages.

2.4 Feature Selection

To optimize the feature set, we implemented a systematic permutation importance analysis combined with progressive feature elimination.

2.4.1 Permutation Importance Analysis

Permutation importance was computed using F1-macro scoring. This technique measures the importance of each feature by randomly shuffling its values and observing the corresponding decrease in model performance. Features that, when shuffled, caused significant performance

drops were considered important, while those showing negligible or even positive impact when shuffled were identified as candidates for removal.

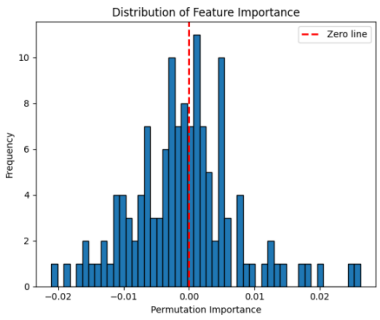

Figure 4: Permutation Importance MLP

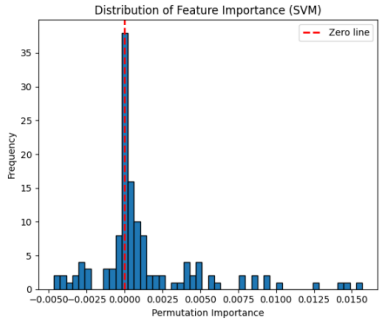

Figure 5: Permutation Importance SVM

For the MLP, permutation importance identified 73 features with negative importance values. For the SVM with an RBF kernel, 31 features were found to have adverse effects on model performance, while 17 were neutral.

2.4.2 Progressive Feature Elimination

Using the importance rankings, we implemented an iterative elimination process. The process initialized with all 132 features. During each iteration, permutation importance was recalculated on the current feature subset to identify the least informative feature. This feature was then permanently removed. This methodology enabled identification of the minimal feature subset that achieved maximum F1-score, effectively eliminating redundant features while maintaining optimal discriminative capability.

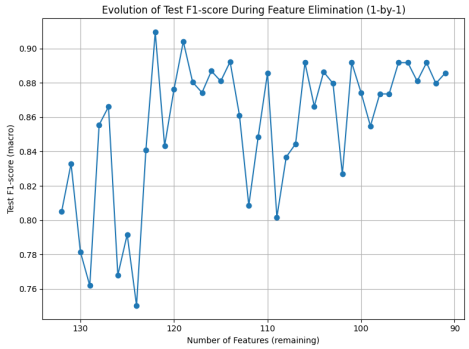

Figure 6: Feature Elimination for MLP

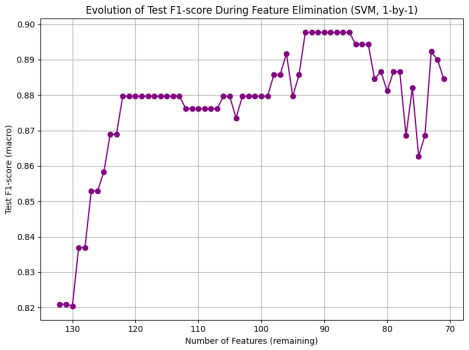

Figure 7: Feature Elimination for SVM

The progressive feature elimination for the MLP demonstrated that performance improved systematically as non-informative features were removed. The optimal configuration used 122 features, achieving a test F1-score of 0.9095, as further feature removal resulted in performance degradation. This represents a substantial improvement over the baseline performance of 0.8052. The SVM achieved its peak performance with a subset of 93 features, achieving an F1-score of 0.8977 compared to the baseline of 0.8210.

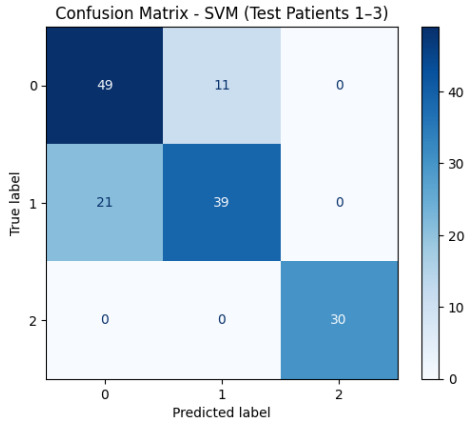

Figure 8: Confusion Matrix for the SVM Model before feature selection

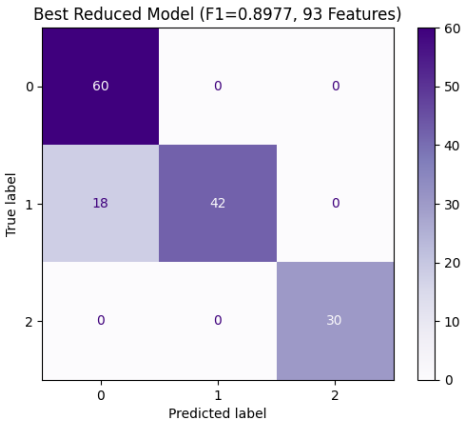

Figure 9: Confusion Matrix for the SVM Model after feature selection

Comparison of confusion matrices (for *SVM* model) before and after feature selection demonstrates a significant reduction in misclassification between exercises E1 and E2, confirming that feature elimination enhanced the model's ability to distinguish subtle movement patterns.

Table 4: Comparison of F1-score with feature selection

Model	Previous	Feature Selection
MLP	0.8052	0.9095
SVM	0.8210	0.8977

2.5 Final Model Validation and Selection

The comprehensive validation phase was conducted to assess model robustness across different patient combinations. All possible combinations of three patients were used as test sets, resulting in 364 distinct validation scenarios. Based on the overall evaluation, the SVM model with RBF kernel and feature removal was selected as the best model. The optimal hyperparameters identified through grid search were: C=1.0, kernel=RBF, and gamma='scale'. The final model, utilizing 93 optimized features, achieved an F1-score of 0.8977 on the original test set and maintained strong consistency across diverse patient groups.

Table 5: Comparison of F1-score across all diferent patient combinations

Model	F1-score
MLP	0.8480 ± 0.0795
SVM	0.8567 ± 0.0872

2.6 Conclusions

When evaluating the model on the new four patients, the overall F1-score decreased to 0.7604, which was lower than the results obtained in previous experiments. Upon further analysis, it became evident that the initial feature removal process, optimized using only used patients 1, 2, and 3 as the test set, introduced a degree of overfitting to those specific subjects. As a result, the selected features did not generalize as effectively to new patients, highlighting the importance of including a broader and more diverse set of subjects during feature optimization to ensure true model generalization.

3 Part 2 - Problem 2 - Impaired Side Recognition

The last part of this project focuses on the classification of stroke patients according to their affected body side. The objective is to develop a machine learning model capable of determining whether a patient is affected on the left (label 0) or right (label 1) side.

3.1 Data Analysis and Preprocessing

Initial analysis involved visualizing complete skeletal sequences through animation to understand exercise dynamics and range of motion. We examined Patient 1's exercises to establish baseline movement patterns. Subsequent analysis focused on average skeletal positions and coordinate variances (Figure 10), revealing asymmetries that informed our feature engineering strategy.

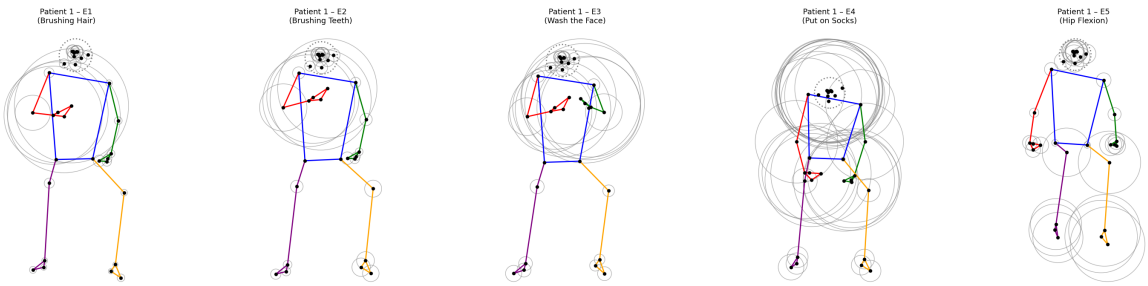

Figure 10: Visualization of the three different exercises of Patient 2

3.2 Baseline Development and Feature Engineering

We established a baseline using 132 static features comprising mean positions and coordinate variances. An SVM with RBF kernel was selected for its computational efficiency in initial experiments. To address the patient-level classification challenge, we implemented two aggregation strategies: majority voting and probability averaging across exercise sequences. A comprehensive leave-group-out cross-validation framework (364 splits with enforced class balance) yielded the following performance across different feature sets.

To capture movement dynamics relevant to the classification problem, we engineered dynamic features by calculating velocity and acceleration magnitudes for left and right body sides.

Velocity was computed as the Euclidean distance between consecutive frame positions, while acceleration was derived from velocity differences. For each side, we extracted mean, standard deviation, and maximum values for both velocity and acceleration. Additionally, we computed asymmetry ratios between left and right sides for these metrics, resulting in 18 dynamic features that quantified movement imbalances.

Table 6: Performance comparison across feature engineering stages

Feature Set	Aggregation Method	Balanced Accuracy
Static (132 features)	Majority Voting	0.4203 ± 0.1992
Static (132 features)	Probability Averaging	0.4148 ± 0.1852
Static + Dynamic (150 features)	Majority Voting	0.5309 ± 0.2456
Static + Dynamic (150 features)	Probability Averaging	0.5227 ± 0.2742

3.3 Feature Importance and Selection

Building upon the baseline results, we implemented a systematic feature selection strategy to identify the most discriminative features for impaired side classification. Using linear SVM coefficients as feature importance measures, we iteratively added features in order of their importance scores, evaluating performance at each step through our balanced cross-validation framework (364 splits with 2 left-impaired and 1 right-impaired patient per test set).

The optimization process was applied to two feature sets: static features only (132 features) and combined static + dynamic features (150 features). The results demonstrate that strategic feature selection significantly outperforms using the full feature set:

Table 7: Performance comparison of optimized feature subsets

Feature Set	Aggregation Method	Features	Balanced Accuracy
Static Only	Majority Voting	34	0.8393 ± 0.2276
Static Only	Probability Averaging	34	0.7390 ± 0.2734
Static + Dynamic	Majority Voting	18	0.8901 ± 0.1924
Static + Dynamic	Probability Averaging	18	0.8929 ± 0.1992

Considering only static features, the optimal subset contained 34 features per exercise, while the combined feature set achieved peak performance with only 18 selected features. This optimized feature set achieved approximately 89% balanced accuracy averaged across 364 splits, representing a substantial improvement over baseline performance. The results clearly demonstrate that feature quality significantly outweighs feature quantity for this classification task.

Given these superior results, we selected the 18-feature set from the combined static and dynamic features for subsequent hyperparameter optimization through grid search.

3.4 Model Comparison and Hyperparameter Optimization

With the optimal 18-feature set established, we conducted comprehensive model comparison between Support Vector Machines and Random Forests. Both models underwent extensive grid search with unbalanced splits to replicate real-world test conditions. SVM optimization utilized 100 folds for robust evaluation, while Random Forest employed 50 folds due to computational constraints.

Table 8: Model performance comparison with optimized feature set

Model	Aggregation Method	Balanced Accuracy
SVM	Majority Voting	0.9142 ± 0.1601
SVM	Probability Averaging	0.8967 ± 0.1849
Random Forest	Majority Voting	0.7600 ± 0.2481
Random Forest	Probability Averaging	0.7667 ± 0.2500

SVM demonstrated superior performance across both aggregation methods, with the optimal model identified through grid search achieving the highest average balanced accuracy of 0.9142 using Majority Voting.

3.5 Final Model Selection

Based on the comprehensive evaluation, we selected the SVM model with polynomial kernel for final implementation. The optimal hyperparameters identified through grid search were: C=10.0, polynomial degree=3, and gamma=0.01. The model was trained on the complete dataset using the optimized 18-feature set. For prediction, we employed a hierarchical aggregation strategy: primary classification via Majority Voting across exercise sequences, with Probability Averaging serving as a tie-breaker for cases with equal votes for each side.

3.6 Conclusions

When evaluated on the final test set, the model achieved a balanced accuracy of 0.6667, which was significantly lower than our validation results and below the expected performance range. This discrepancy can be explained by the test data our model was evaluated, since it suffered augmentation through coordinate mirroring across the y-axis doubling the test size. To improve our model's performance, we could implement similar data augmentation during training by mirroring all coordinate data. This approach would help create a more robust model by doubling the size of our training dataset.