Neuromappr

Brain-Computer Interface Movement Decoding Using Support Vector Machines

AUTHOR
Wanghley Soares Martins

AFFILIATION
Duke University, Pratt School of Engineering, Department of Electrical and Computer Engineering

PUBLISHED
April 27, 2025

1 Introduction

1.1 Background

Since the creation of machines and computers, the human interaction with them has been evolving and sometimes quite challenging. For instance, the first computers were operated using punch cards, which required a lot of effort to input data. As technology advanced, we moved to keyboards and mice, which made it easier to interact with computers. However, these methods still require physical movement and can be limiting for individuals with disabilities or injuries. This interaction with computers has been a challenge investigated for researchers and engineers for decades in the field of Human-Computer Interaction (HCI) [1].

The goal of HCI is to create systems that are easy to use and understand, allowing users to interact with computers in a natural and intuitive way. This has led to the development of various input devices, such as touchscreens, voice recognition, and even brain-computer interfaces (BCIs), the focus of this project.

Brain-Computer Interfaces (BCIs) are systems that enable direct communication between the human brain and devices, bypassing the need for physical movement and control, which can be particularly beneficial for individuals with disabilities or injuries. Among other data, BCIs can use electroencephalography (EEG) signals to decode brain activity and translate it into commands for controlling devices.

BCI technology has the potential to revolutionize the way we interact with computers and other devices, making it possible for individuals with disabilities to regain control over their environment. Central to the efficacy of BCIs is the accurate interpretation of electroencephalogram (EEG) signals, particularly those associated with motor imagery—the mental simulation of movement without actual execution as described by Costantini et al. (2009) [2].

From a machine learning standpoint, the classification of EEG signals for motor imagery tasks presents several intricate challenges:

- **High Dimensionality with Limited Samples**: EEG data are inherently high-dimensional, often involving recordings from numerous electrodes (e.g., 102 in this project), each capturing complex temporal dynamics. However, the number of labeled training samples is typically limited, leading to the “curse of dimensionality,” where models risk overfitting and poor generalization.
- **Non-Stationarity and Noise**: EEG signals are susceptible to various artifacts (e.g., muscle movements, eye blinks) and exhibit non-stationary behavior, complicating the extraction of consistent features across sessions and subjects.
- **Inter-Subject Variability**: There is significant variability in EEG signals across different individuals, which can affect the performance of machine learning models trained on data from

a single subject. This variability necessitates the development of robust algorithms that can generalize well across different users.
- **Temporal Dynamics**: EEG signals are time-dependent, and capturing the temporal dynamics of brain activity is crucial for accurate classification. This requires the use of advanced techniques that can model temporal relationships effectively.

Support Vector Machines (SVMs) have been extensively employed in this domain due to their effectiveness in high-dimensional spaces and robustness to overfitting. For instance, Costantini et al. (2009) [2] demonstrated the utility of SVMs in classifying EEG signals for BCI applications, emphasizing their capacity to handle complex, high-dimensional data.

1.2 Objectives

In this project, we aim to implement and evaluate SVM-based classifiers for distinguishing between left and right-hand motor imagery using EEG data. By addressing the aforementioned challenges through appropriate preprocessing, feature extraction, and model selection, we seek to contribute to the development of reliable and efficient BCI systems.

1.2.1 Spedific Objectives

1. Implement a Support Vector Machine (SVM) classifier for EEG data.
    - Evaluate the performance of the SVM classifier using various kernel functions (linear, polynomial, and radial basis function).
    - Optimize the SVM hyperparameters using grid search and cross-validation techniques.
2. Compare the performance of the SVM classifier with Overt and Imagined Motor Imagery (MI) tasks.
    - Analyze the classification accuracy, precision, recall, and F1-score for both tasks.
    - Investigate the impact of different EEG feature extraction methods on classification performance.
3. Visualize and interpret the results of the SVM classifier, including feature importance and decision boundaries.
    - Provide insights into the EEG features that contribute to the classification performance.
    - Visualize the decision boundaries of the SVM classifier in the feature space.

1.3 Dataset overview

The dataset utilized in this project comprises EEG recordings from a Brain-Computer Interface (BCI) experiment designed to distinguish between left and right-hand movements. The recordings are categorized into two distinct types:

1. **Overt Motor Imagery (OMI)**: This task involves participants imagining moving their left or right hand while EEG signals are recorded.

2. **Imagined Motor Imagery (IMI)**: In this task, participants are instructed to imagine moving their

left or right hand without any actual movement.

Moreover, the dataset provides the XY coordinates of the electrodes, which are crucial for visualizing the spatial distribution of EEG signals across the scalp. The dataset is organized into tw main folders: `data`.

The data is organized on the following files:

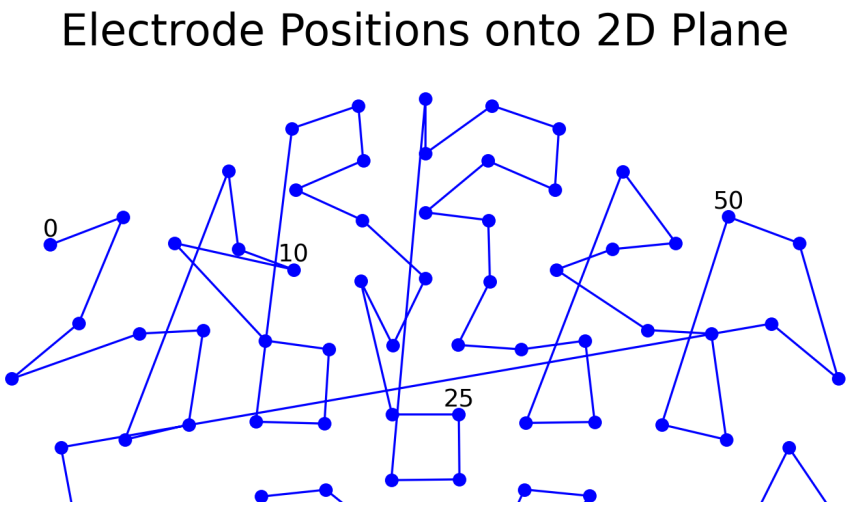
- `data/BCIsensor_xy.csv`: Contains the XY coordinates of the electrodes.
- `data/feaSubEImg_1.csv`: Contains the EEG data for the IMI task for one direction (left).
- `data/feaSubEImg_2.csv`: Contains the EEG data for the IMI task for the other direction (right).
- `data/feaSubEOvert_1.csv`: Contains the EEG data for the OMI task for one direction (left).
- `data/feaSubEOvert_2.csv`: Contains the EEG data for the OMI task for the other direction (right).

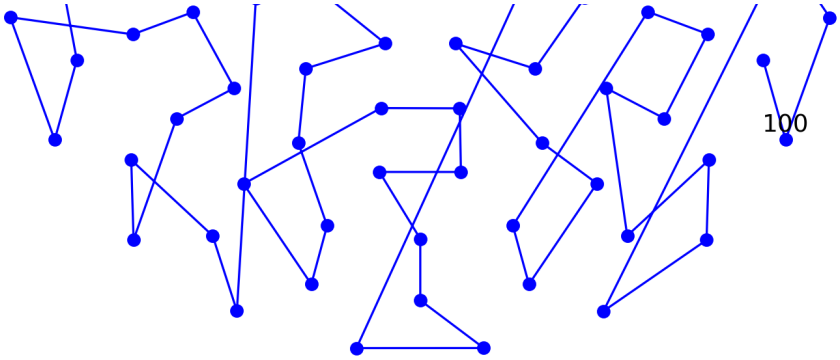
On the electrodes' data, it is organized with each trial as a column and each electrode on that trial as a row, as shown in Table 1.

Table 1: Electrodes' data organization

Electrode	Trial 1	Trial 2	Trial 3
1	Value	Value	Value
2	Value	Value	Value
3	Value	Value	Value

In terms of electrode placement, the dataset includes 102 electrodes arranged in the configuration plotted in Figure 1. The electrodes are positioned according to the 10-20 system, a standardized method for electrode placement in EEG studies. This system ensures consistent and reproducible electrode locations across different subjects and studies.



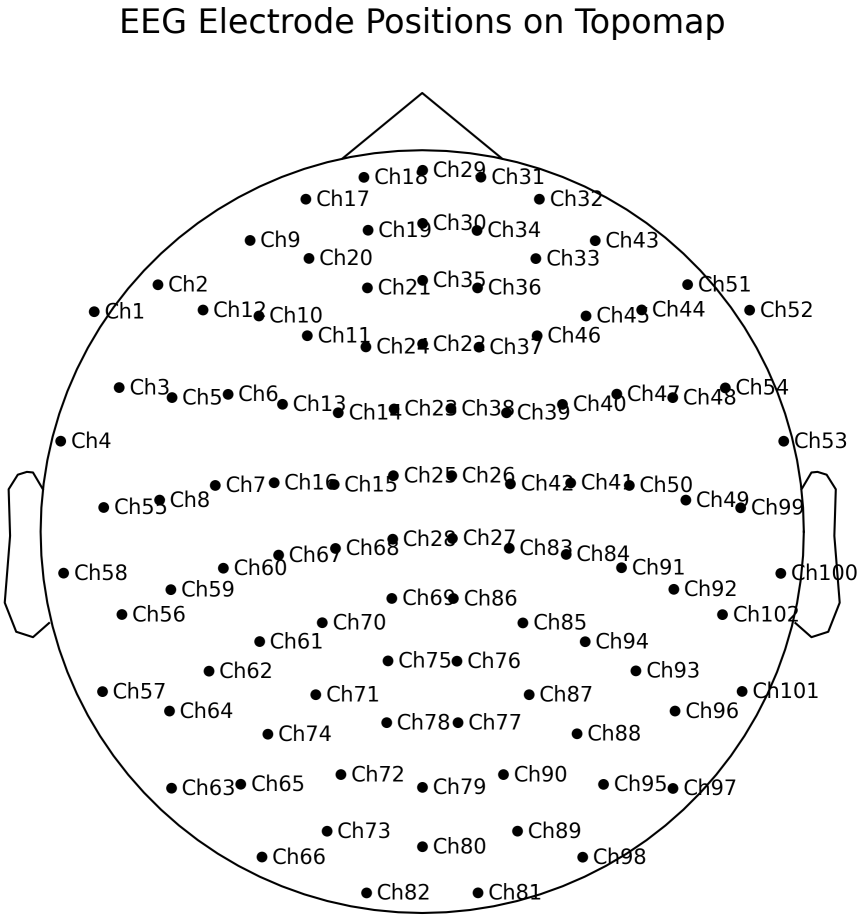


Electrodes positions

Figure 1

Figure 1 Electrode positions on the scalp according to the 10-20 system. The numbers indicate the electrode labels, which correspond to the columns in the dataset. Source: By the author (2025).

A topographic view into the skull is shown in Figure 2, where the electrodes are represented as dots on the scalp. The numbers indicate the electrode labels, which correspond to the columns in the dataset. The topographic representation provides a visual understanding of the spatial distribution of EEG signals across the scalp, allowing for better interpretation of the data.



Electrodes positions topologic

Figure 2

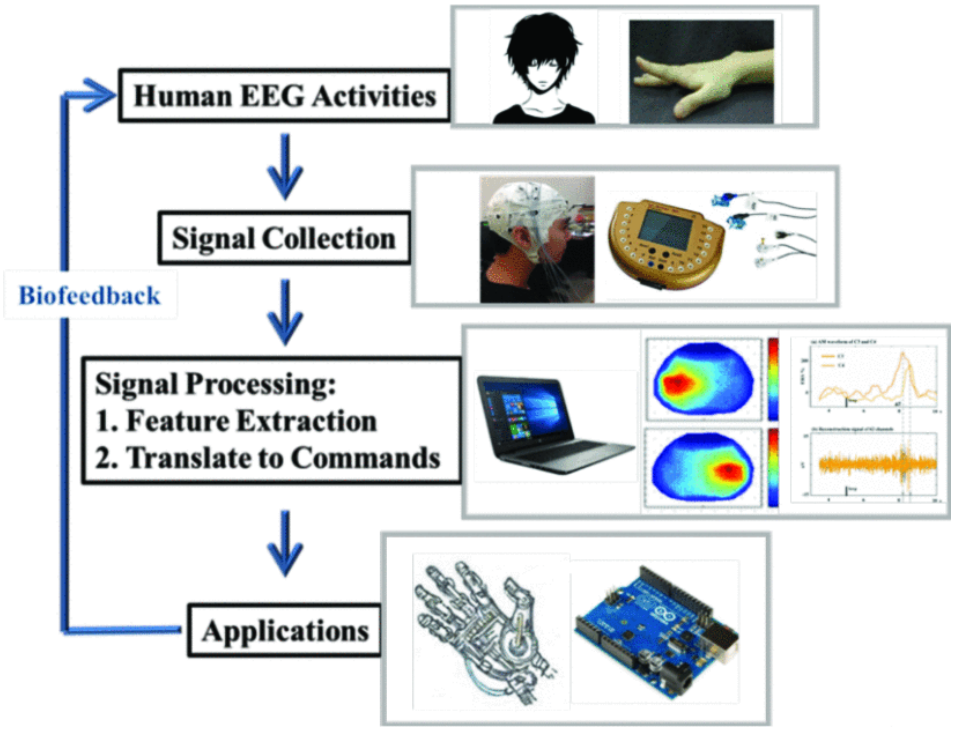
Figure 2 Topographic representation of the electrode positions on the scalp. The numbers indicate the electrode labels, which correspond to the columns in the dataset. Source: By the author (2025).

2 Methods

2.1 Project Structure

This project follows a structured pipeline encompassing data acquisition, signal preprocessing, classification, and interpretation—each stage playing a critical role in the accurate decoding of motor imagery from EEG signals. Inspired by the workflow presented in Wu et al. (2018)[3], this architecture promotes clarity, reproducibility, and alignment with current practices in EEG-based Brain-Computer Interface (BCI) research.

As shown in **Figure 3**, the end-to-end system begins with EEG signal generation through motor-related brain activity, followed by collection via sensor-equipped EEG caps. The recorded signals then undergo preprocessing and are classified using machine learning algorithms—particularly Support Vector Machines (SVMs)—to distinguish between motor imagery classes. The final output may be translated into control commands or used for further neurophysiological interpretation.



Project pipeline

Figure 3

Figure 3: Conceptual workflow from EEG activity to application.
Source: adapted from Wu et al. (2018) [3].

This representation effectively illustrates the typical stages of a BCI system: (1) EEG generation through neural activity, (2) signal acquisition, (3) processing and classification (often including feature extraction and transformation into actionable information), and (4) deployment or application, such as robotic actuation or interface control.

The focus on this paper is on the signal processing and classification stages, where we will implement a Support Vector Machine (SVM) classifier to decode motor imagery tasks from EEG signals. We will not delve into the EEG generation and acquisition stages, as they are outside the scope of this project.

2.2 Mathematical Formulation

2.2.1 Support Vector Machines: A Mathematical Framework for High-Dimensional EEG Classification

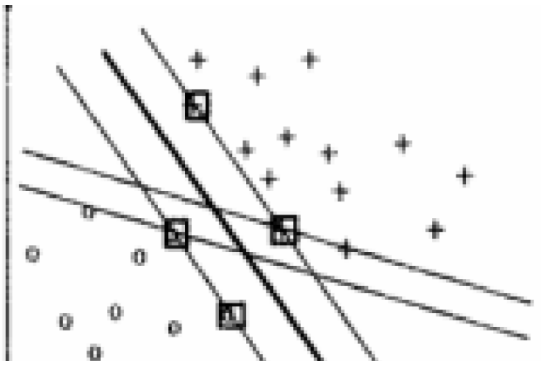
In this project, Support Vector Machines (SVMs) are utilized to classify EEG signals corresponding to left and right-hand movement, either overt or imagined. The application of SVMs to EEG data is particularly well-justified: these classifiers are known to perform effectively in high-dimensional spaces, especially when the number of observations is small relative to the number of features—a condition that closely matches our dataset, which consists of 204 features per trial but only 240 trials per condition.

The primary objective of an SVM is to find a hyperplane that best separates two classes in the feature space. In the simplest case, where data are linearly separable, a linear SVM seeks the hyperplane that maximizes the margin between the two classes. For a given trial $\mathbf{x}_i \in \mathbb{R}^{204}$ with class label $y_i \in \{-1, +1\}$, the classifier has the form:

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$$

Here, $\mathbf{w}$ is the weight vector orthogonal to the decision boundary, and b is the bias term. The training process identifies the optimal $\mathbf{w}$ and b such that the margin—the distance between the hyperplane and the closest points of each class—is maximized, such as illustrated on Figure 4 by Dai et. al (2020) [4]. This is achieved by solving the following convex optimization problem:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \quad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1$$



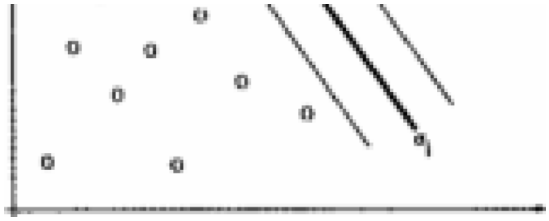


Figure 4

Figure 2: Support Vector Machine (SVM) illustration. The dashed line represents the decision boundary, while the solid lines indicate the margin. The support vectors are the data points closest to the decision boundary [4].

This formulation is known as the **hard-margin SVM**, suitable only when the data is perfectly separable. In practice, particularly with EEG data, perfect separation is unrealistic due to measurement noise, physiological artifacts, and signal overlap between classes. Thus, we turn to the **soft-margin SVM**, which introduces slack variables $\xi_i \geq 0$ that allow some misclassification, yielding a more robust solution:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \quad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i$$

The parameter $C > 0$ controls the trade-off between maximizing the margin and minimizing classification errors. A smaller C permits more violations (larger margin, higher bias), while a larger C enforces stricter separation (lower bias, higher variance). This parameter is crucial in EEG classification, particularly when comparing performance on imagined versus overt movement data, as imagined movements tend to be more subtle and noisy.

An important property of SVMs is that they produce not only binary class decisions, but also **decision statistics**. For any trial $\mathbf{x}$, the signed output $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$ quantifies the distance of the trial from the decision boundary. This value is used to compute performance metrics such as ROC curves and can serve as a confidence measure for the prediction. The ability to produce such a statistic, rather than just a class label, is a key reason SVMs are well-suited to this project.

Moreover, the learned weight vector $\mathbf{w}$ in a linear SVM offers meaningful insight into the relative importance of each feature. In our case, EEG features are structured as 204 elements, corresponding to 102 electrodes each providing two components: the x-gradient and y-gradient of the electric field. To interpret the contribution of each electrode, we compute the magnitude of each Ex/Ey pair:

$$w_k = \sqrt{w_{2k-1}^2 + w_{2k}^2}, \quad k = 1, \ldots, 102$$

This produces a 102-element vector indicating the relative influence of each electrode on the classification decision. The resulting magnitudes are mapped onto a topographic layout of the head, allowing spatial visualization of the most informative brain regions.

While the linear SVM provides interpretability, it may not fully capture complex, nonlinear

relationships in the EEG signals. To address this, we also consider **kernel SVMs**, which implicitly map the data into higher-dimensional feature spaces via kernel functions. These functions compute inner products in the transformed space without requiring explicit transformation, a technique known as the **kernel trick**. The decision function becomes:

$$f(\mathbf{x}) = \sum_{i \in \mathcal{S}} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b$$

Here, $\mathcal{S}$ is the set of support vectors, and $k(\cdot, \cdot)$ is a kernel function. We explore multiple kernel types in this project, including:

- **Linear:** baseline, interpretable, fast;
- **Radial Basis Function (RBF):** capable of capturing smooth nonlinearities;
- **Sigmoid:** similar to neural networks, less common in practice;
- **Polynomial:** included for completeness and comparison.

In kernel SVMs, we lose direct access to the weight vector $\mathbf{w}$, making interpretability more challenging. To address this, we use **permutation importance** to estimate the relative importance of each feature for nonlinear kernels.

Implementation of SVMs in this project is performed using the `sklearn.svm` module, which provides a user-friendly interface for training and evaluating SVM classifiers. The `SVC` class allows for easy specification of kernel types, hyperparameters, and performance metrics.

2.2.2 Permutation Importance

Let $X = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^\top \in \mathbb{R}^{n \times d}$ be the EEG data matrix and f the trained classifier. The permutation importance of feature j is estimated as:

1. Compute the **baseline score** (e.g., accuracy or AUC) on the original test data:

$$s_{\text{baseline}} = \text{score}(f, X, y)$$

2. For each feature j, create a perturbed version $X^{(j)}$ where the j^{th} column is randomly shuffled across all samples.

3. Evaluate the performance on the perturbed data:

$$s^{(j)} = \text{score}(f, X^{(j)}, y)$$

4. Define the **importance** of feature j as the performance drop:

$$\text{Importance}_j = s_{\text{baseline}} - s^{(j)}$$

This process is repeated over multiple random permutations (e.g., 10 times), and the mean drop in performance is taken as the final estimate of the feature's importance.

Implementation in This Project:

In our kernel SVM experiments, we used `sklearn.inspection.permutation_importance` with the following settings:

- `n_repeats=10` to average out noise in the performance estimates,
- `scoring='accuracy'` or `'roc_auc'` depending on the metric of interest,
- `n_jobs=-1` to parallelize computation across all available cores.

This method yielded a 204-dimensional vector representing the importance of each feature (i.e., each EEG gradient channel). To visualize the spatial contributions of the electrodes, we paired each Ex and Ey component and computed a magnitude-based importance score for each electrode:

$$w_k = \sqrt{\mathrm{Importance}_{2k-1}^2 + \mathrm{Importance}_{2k}^2}, \quad k = 1, \ldots, 102$$

These values were mapped onto the scalp using topographic plots to interpret which brain regions most strongly contributed to classification, despite the underlying SVM model being nonlinear and not directly interpretable.

2.2.3 Extending SVMs with Kernels for Nonlinear EEG Classification

While linear Support Vector Machines (SVMs) provide a robust and interpretable foundation for classifying EEG data, they assume that the classes are separable by a linear boundary in the original feature space. However, EEG signals—especially those corresponding to imagined movements—often exhibit complex, nonlinear patterns that cannot be adequately captured using linear decision functions. To address this, we incorporate **kernel methods**, which allow SVMs to construct flexible, nonlinear decision boundaries by implicitly mapping data into higher-dimensional spaces.

In the kernelized SVM, we represent the decision function as:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b$$

Here,
- $\mathbf{x}_i$ are the training samples,
- $y_i \in \{-1, +1\}$ are the labels,
- α_i are the learned Lagrange multipliers,
- $k(\mathbf{x}_i, \mathbf{x})$ is the kernel function measuring similarity,
- b is the bias term.

Only the training samples with $\alpha_i > 0$ (support vectors) contribute to the decision function. The function $f(\mathbf{x})$ returns a continuous **decision statistic**, and its sign determines the predicted class.

2.2.3.1 Kernels Used in This Project

We explore four kernel types, each with unique geometric and computational properties:

1. **Linear Kernel** (baseline):

$$k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j$$

Used for direct interpretability and comparison with non-kernel results.

2. **Radial Basis Function (RBF) Kernel**:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2)$$

Introduces local flexibility, useful for capturing subtle nonlinearity in noisy EEG signals. The hyperparameter γ controls the influence of individual training points.

3. **Polynomial Kernel**:

$$k(\mathbf{x}_i, \mathbf{x}_j) = (\gamma \mathbf{x}_i^\top \mathbf{x}_j + r)^d$$

Enables modeling more complex global relationships between channels. Requires tuning of degree d, scale γ, and offset r.

4. **Sigmoid Kernel**:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\gamma \mathbf{x}_i^\top \mathbf{x}_j + r)$$

Related to neural network activation functions, but less commonly used due to its sensitivity to hyperparameter settings.

Each kernel introduces its own set of hyperparameters, which are optimized during the **inner loop of the two-level cross-validation** strategy used in this project.

2.2.3.2 Kernel Selection Strategy

For each kernel, we trained a model on EEG trials using the same cross-validation framework and computed performance metrics including accuracy and ROC area. This allowed us to compare not only final performance but also generalization stability across folds.

The goal was not to “pick the best kernel” outright, but rather to understand the types of decision boundaries each kernel enables and how well they adapt to the EEG signal characteristics for both overt and imagined movement trials.

- **RBF kernels** generally performed better on imagined movement data, likely due to their ability to adapt to low signal-to-noise ratios.
- **Linear kernels** offered clearer interpretability and often comparable performance on overt movement data, where class separation is stronger.
- **Polynomial kernels** showed some promise in capturing intermediate complexity but were more prone to overfitting unless carefully regularized.

2.2.3.3 Interpretation of Kernel SVMs

Unlike linear SVMs, kernelized models do not produce a weight vector $\mathbf{w}$ in the input space. As such, **direct interpretation of feature contributions is not possible**. To bridge this gap, we

such, **direct interpretation of feature contributions is not possible**. To bridge this gap, we applied **permutation importance**, which measures how shuffling each feature affects classification performance (as described in a previous section).

To interpret these results spatially, we paired each Ex/Ey feature, computed magnitudes, and visualized electrode-wise importance on a scalp topomap. This enabled us to identify cortical regions that played a key role in the model's classification decisions, even in the absence of explicit feature weights.

2.2.4 Regularization Parameter and Norm Penalties in SVMs

In Support Vector Machines (SVMs), the **regularization parameter** C is critical to managing the trade-off between model complexity and classification accuracy. Particularly in the context of EEG-based Brain-Computer Interface (BCI) classification, where high-dimensional and noisy data are prevalent, appropriate regularization ensures robust generalization to unseen trials.

Regularization is not just a technique for numerical stability—it fundamentally shapes the classifier's bias-variance tradeoff and governs what kind of solution is learned [5]. For BCI movement decoding, where noisy, high-dimensional data are the norm, choosing the right regularization strategy is crucial to achieving both accuracy and interpretability.

2.2.4.1 Soft-Margin SVM and the Role of C

For EEG classification, we typically rely on the **soft-margin SVM**, which tolerates some misclassification to allow for greater generalization [5]. The **primal optimization problem** is given by:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \quad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

Here:
- $\|\mathbf{w}\|^2$ is the **L2 norm squared**, encouraging small weights and a large-margin hyperplane,
- C controls the penalty for slack variable ξ_i, which quantifies margin violations.

This formulation is an instance of **Ridge-like regularization** applied to an SVM. It penalizes large weights quadratically, resulting in smooth decision boundaries and improved stability—ideal for high-dimensional EEG data where overfitting is a risk [5].

2.2.4.2 L2 Regularization (Ridge): Smooth and Stable

L2 regularization adds a quadratic penalty on the magnitude of the weights:

$$\mathcal{L}_{\text{L2}}(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 = \frac{1}{2}\sum_{j=1}^{d} w_j^2$$

[5].

This approach:

This approach:
- Encourages all weights to be small but nonzero,
- Produces **dense models**, where most features contribute a little,
- Is suitable when all features (i.e., EEG channels) may carry signal but none dominate.

In our project, this penalty was particularly useful for classifying overt movement EEG, where clean signal and broader spatial activation are expected.

2.2.4.3 L1 Regularization (Lasso): Sparsity and Interpretability

An alternative to L2 is **L1 regularization**, which penalizes the **absolute value** of weights:

$$\mathcal{L}_{\text{L1}}(\mathbf{w}) = \sum_{j=1}^{d} |w_j|$$

[5]

The full soft-margin objective with L1 becomes:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \lambda \sum_{j=1}^{d} |w_j| + C \sum_{i=1}^{n} \xi_i \quad \text{subject to} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i$$

This form:
- Produces **sparse models** where only a subset of features (channels) receive nonzero weights,
- Facilitates **feature selection** by zeroing out uninformative dimensions,
- Increases interpretability by highlighting the most discriminative electrodes.

While not directly implemented by default in all SVM libraries, **L1-penalized linear SVMs** are supported in scikit-learn through `LinearSVC(penalty='l1', dual=False)` and are particularly useful in exploratory settings or when interpretability is a priority.

2.2.4.4 EEG Interpretation and Norm Selection

- **L2 regularization** (used in our main experiments) was more appropriate for general classification performance, especially in **imagined movement**, where signal is sparse but noisy, and a smooth model reduces overfitting.
- **L1 regularization** could be applied in future work to aid in the identification of the most informative EEG channels—this may be particularly valuable when reducing feature space dimensionality or visualizing important brain regions.

2.3 Single Linear SVM Classifier Without Cross-Validation

As an initial proof-of-concept, we trained a linear Support Vector Machine (SVM) classifier on the EEG data using a traditional train-test split without any form of cross-validation. This approach was conducted separately for the imagined and overt (real) movement datasets, allowing for direct exploration of classification performance and spatial interpretability of the resulting model weights.

For this proof of concept analysis, we decided to use Ridge regularization (L2) with a linear kernel, as it is the most interpretable and straightforward approach for EEG data. Also, we included $C = 1.0$ as the regularization parameter which is, generally, a middle point between overfitting and underfitting.

2.3.1 Data Preparation

The EEG data were loaded from CSV files corresponding to two classes: - `feaSubEImg_1.csv` and `feaSubEImg_2.csv` for imagined movements, - `feaSubEOvert_1.csv` and `feaSubEOvert_2.csv` for overt movements.

Each dataset contains 204 features per trial, corresponding to the Ex and Ey components from 102 electrodes. After loading and transposing the data, we concatenated class 1 and class 2 trials and assigned binary labels. A 70/30 train-test split was used for model evaluation.

2.3.2 Model Training

We trained a linear SVM using `sklearn.svm.SVC` with the following configuration: - `kernel='linear'` - `C=1.0` (regularization parameter) - `probability=True` to enable soft prediction scores

The decision function `f(x) = w^T x + b` was used to compute both class predictions and decision statistics. Accuracy and ROC-AUC were recorded on Figure 5.

```
Accuracy: 0.8611111111111112
Accuracy: 0.8611
ROC AUC: 0.9327

Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.76      0.84        34
           1       0.82      0.95      0.88        38

    accuracy                           0.86        72
   macro avg       0.87      0.86      0.86        72
weighted avg       0.87      0.86      0.86        72
```

(a) Imagined Movements

```
Accuracy: 0.9166666666666666
```

```
Accuracy: 0.9167
ROC AUC: 0.9756

Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.88      0.92        41
           1       0.86      0.97      0.91        31

    accuracy                           0.92        72
   macro avg       0.92      0.92      0.92        72
weighted avg       0.92      0.92      0.92        72
```

(b) Overt Movements

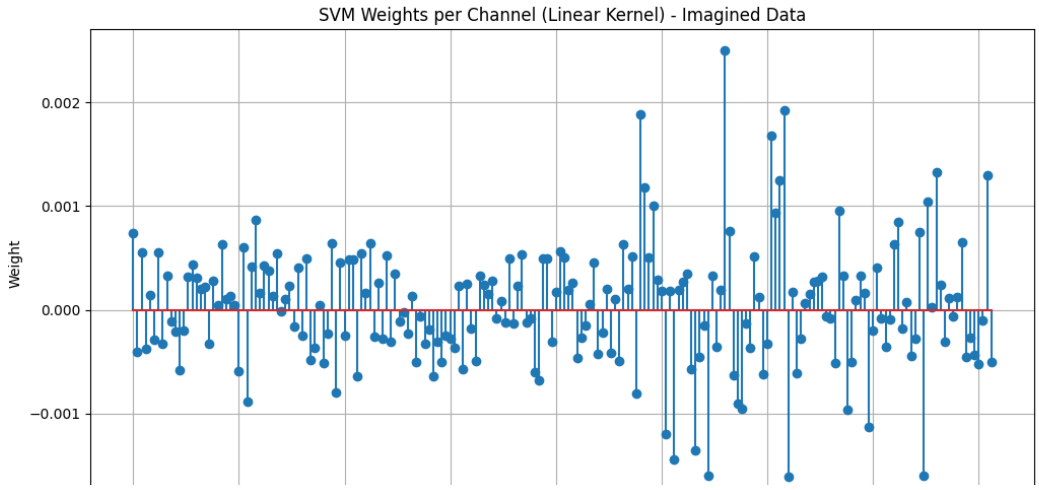
Figure 5: Linear SVM performance metrics for (a) imagined versus (b) overt movements, showing classification accuracy, ROC-AUC, and classification report.

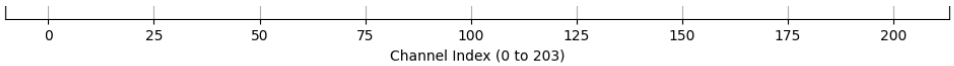
2.3.3 Weight Analysis and Interpretation

The weight vector $\mathbf{w} \in \mathbb{R}^{204}$ from the trained linear SVM was analyzed to interpret spatial channel importance. The 204-element vector was split into x-gradient (`W_x`) and y-gradient (`W_y`) components, and a per-electrode magnitude was computed:

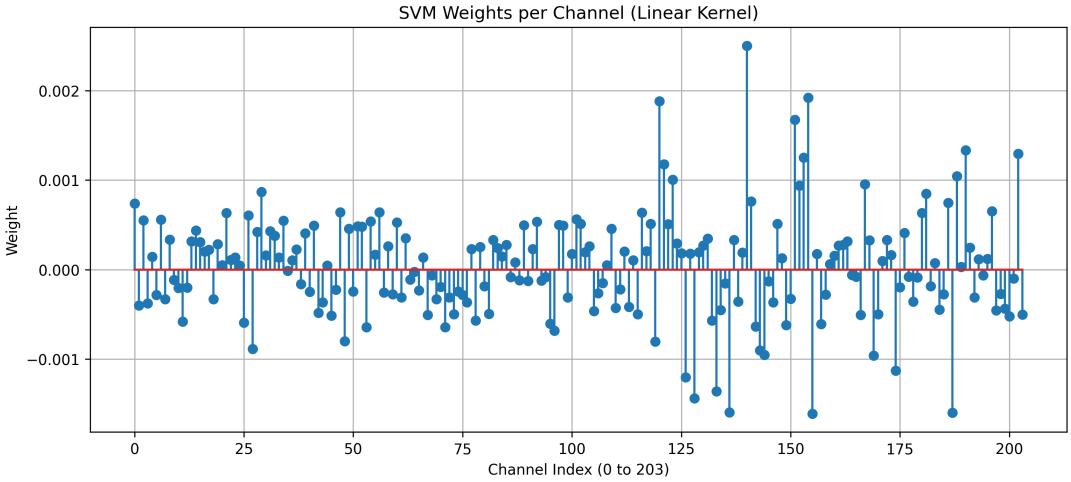
$$W_k = \sqrt{w_{2k-1}^2 + w_{2k}^2}, \quad k = 1, \ldots, 102$$

These magnitudes were plotted to visualize the influence of each electrode, such as on Figure 6.





(a) Imagined Movements



(b) Overt Movements

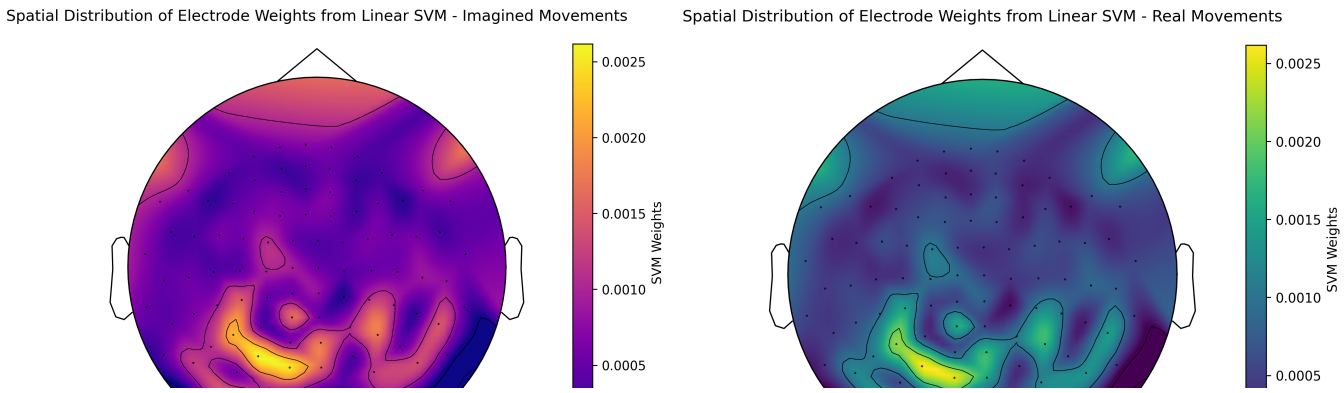
Figure 6: Linear SVM weight magnitudes for (a) imagined versus (b) overt movements, showing the relative importance of each electrode in the classification decision. Topographic maps display normalized weights (µV) across scalp regions, with red indicating positive contribution and blue indicating negative contribution to classification.

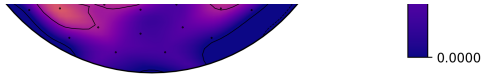
2.3.4 Topographic Mapping

To understand spatial patterns, electrode positions from `BCIsensor_xy.csv` were scaled and mapped to a topographic head model using MNE-Python. The magnitude values `W_mag` were plotted using MNE's `plot_topomap` function.

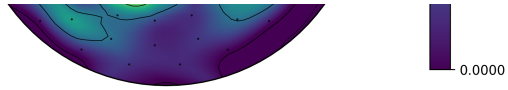
- Topomaps provided intuitive spatial distribution of informative channels.
- Interpolated heatmaps were also generated using `scipy.interpolate.griddata` to visualize gradients.

In order words, we can visualize where is located the most important electrodes for the classification of the imagined and overt movements. The topographic maps were generated using MNE-Python's `plot_topomap` function, which allows for intuitive spatial distribution of informative channels. The topomaps are shown in Figure 7.

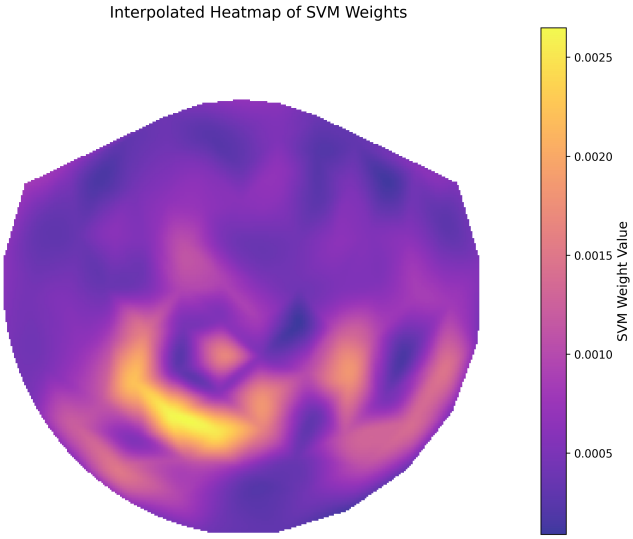


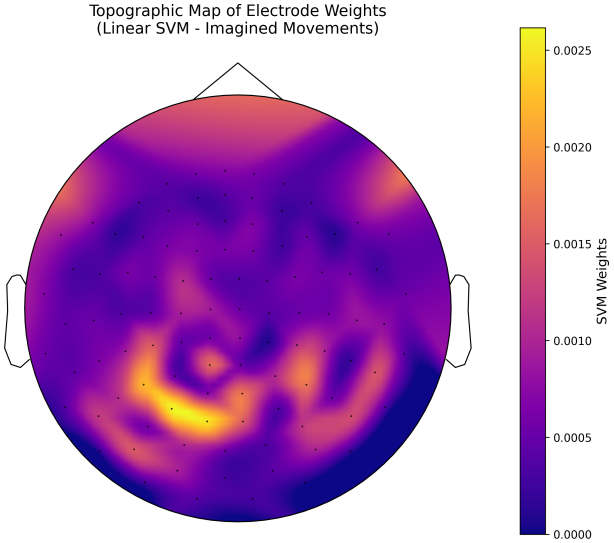


(a) Imagined Movements (Extrapolated)

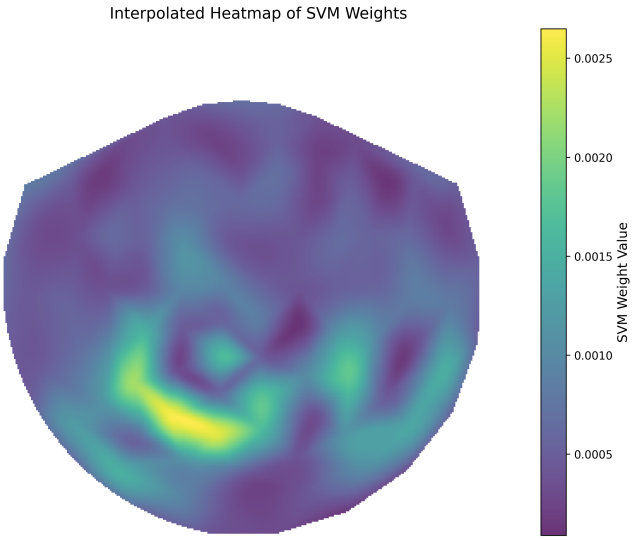


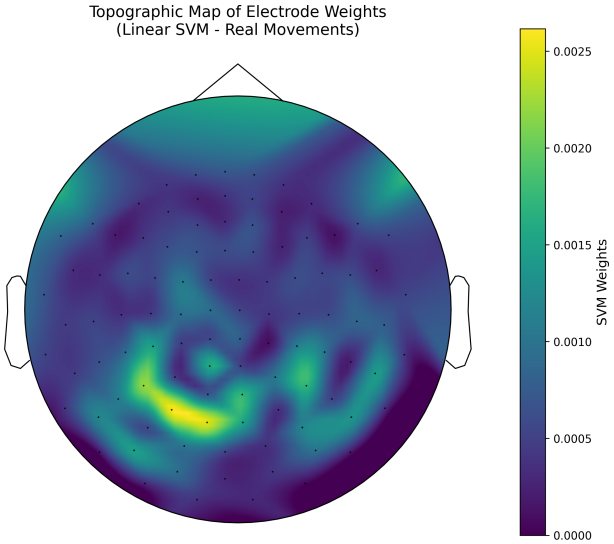
(b) Overt Movements (Extrapolated)





(c) Imagined Movements (Interpolated)





(d) Overt Movements (Interpolated)

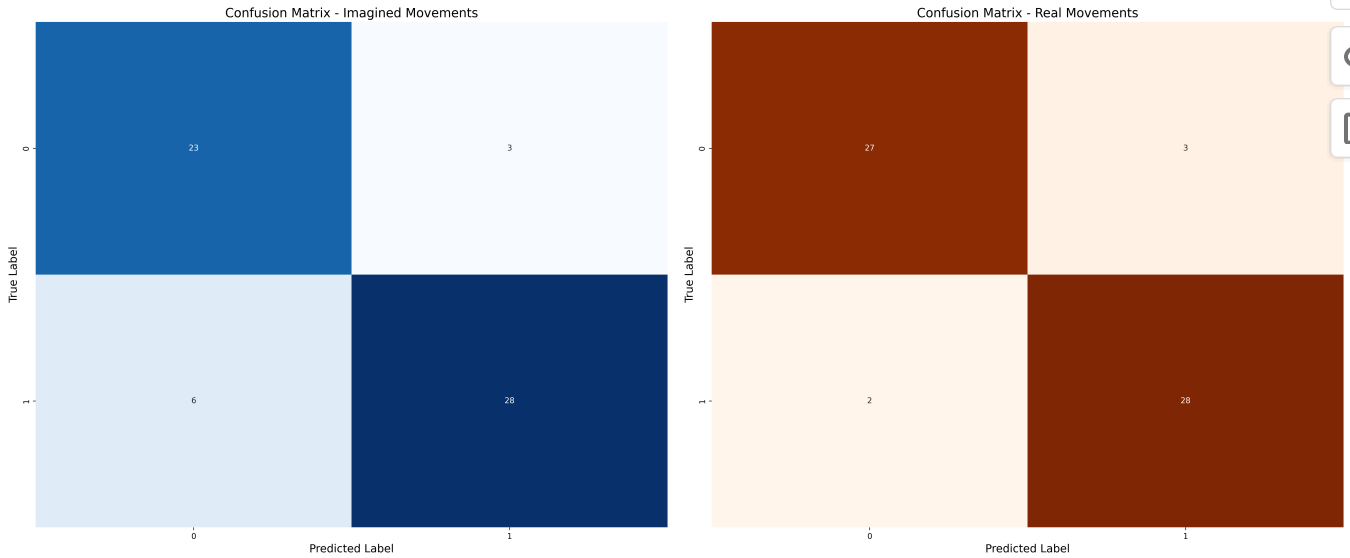
Figure 7: Linear SVM topographic maps showing spatial distribution of informative electrodes: (a-b) Extrapolated and (c-d) interpolated representations for imagined versus overt movements. Warmer colors indicate stronger positive weights, cooler colors show negative weights in the classification decision. Color bars represent normalized weight values.

2.3.5 Evaluation: Confusion Matrices and ROC Curves

We computed confusion matrices for both imagined and real movement classifiers, as well as ROC curves and AUC metrics.

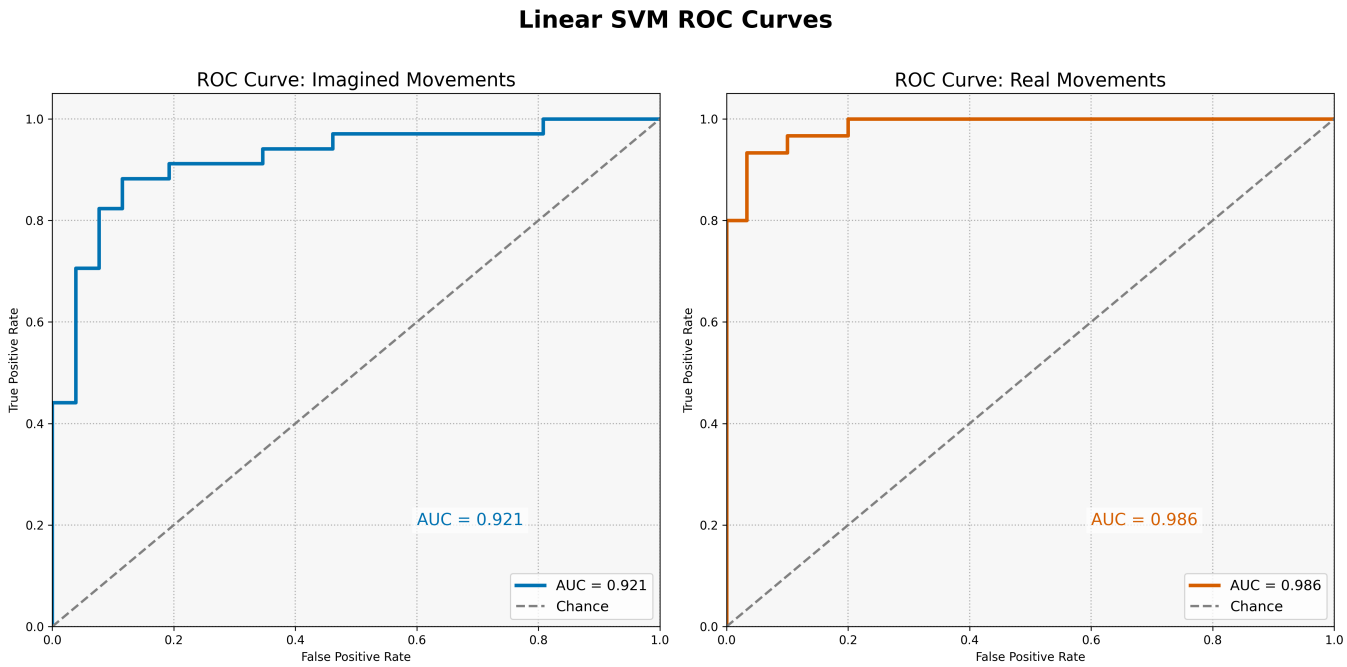
Figure Placeholder: `confusion_matrix_imagined_movements_linearSVM.png` **Figure Placeholder:** `confusion_matrix_real_movements_linearSVM.png` **Figure Placeholder:**

roc_curves_linear_SVM.png



(a) Imagined Movements

(b) Overt Movements



(c) ROC Curves

Figure 8: **Figure 2.** Confusion matrices showing classification performance for (a) imagined versus (b) overt movements. Diagonal elements (TP, TN) indicate correct classifications, while off-diagonal (FP, FN) show errors. Color intensity corresponds to frequency counts.

This exploratory analysis validated the use of a linear SVM for both overt and imagined movement classification, highlighting meaningful spatial distributions and differences in performance characteristics that motivate the need for more robust cross-validation and kernel-based comparisons in subsequent sections.

Without much tunning, the linear SVM classifier achieved an accuracy of 0.87 for imagined movements and 0.92 for overt movements, with ROC-AUC scores of 0.93 and 0.97, respectively.

The confusion matrices indicated a high true positive rate (TPR) and low false positive rate (FPR) fo both tasks, demonstrating the classifier's effectiveness in distinguishing between left and right-hand movements.

However, we need to notice that this result may be biased due to the lack of cross-validation and hyperparameter tuning. The linear SVM classifier was trained on a single train-test split, which may not generalize well to unseen data. Therefore, we will implement a more robust cross-validation strategy in the next sections. Therefore this is one of our next steps to validate accurateness and generalization of the model.

Moreover, the topographic maps provided valuable insights into the spatial distribution of informative electrodes, which can guide future feature selection and model refinement.

2.4 Regularization Parameter Selection: L1 vs L2 Penalties

As part of our methodology, we performed a detailed analysis of the effect of regularization on the classification performance of linear SVMs. In this context, we compared **L1 (Lasso)** and **L2 (Ridge)** regularization strategies on both overt and imagined EEG datasets. These experiments serve to illustrate how different norm constraints impact generalization, sparsity, and model interpretability.

2.4.1 Weight Magnitude Comparison

We trained two `LinearSVC` classifiers using: - `penalty='l2'` with `dual=True` - `penalty='l1'` with `dual=False` (as required for L1 regularization)

Both classifiers were trained on overt movement data. After training, we compared the **absolute weight magnitudes** produced by each model to visualize sparsity and regularization effects. L2 yielded dense weight vectors, while L1 induced sparsity by zeroing out several coefficients as shown in Figure 9.

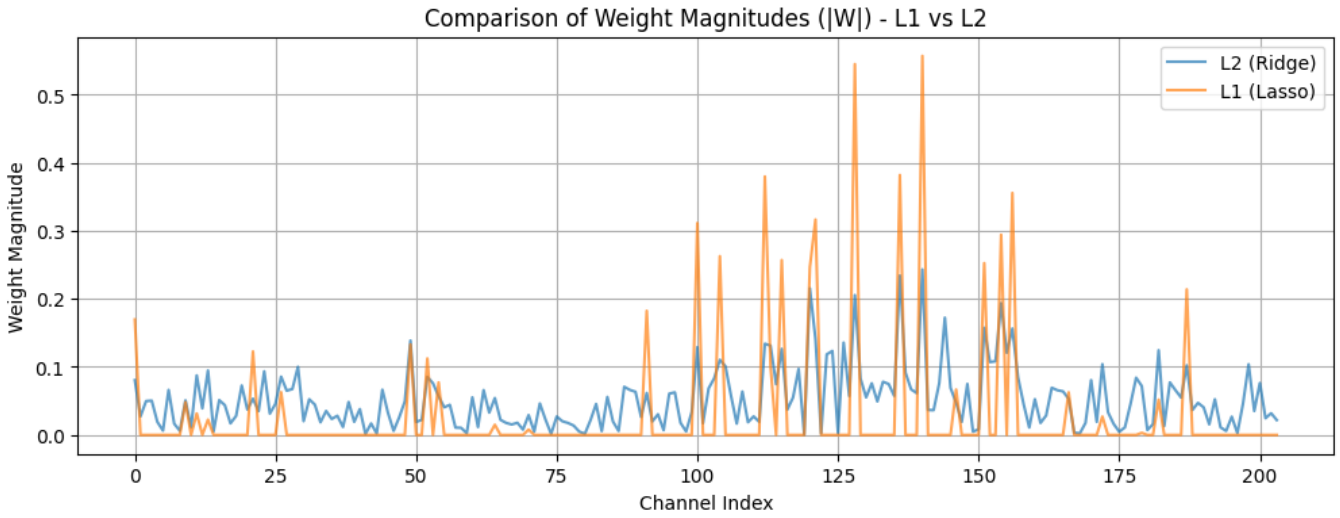


Figure 9: Comparison of weight magnitudes for L1 and L2 regularization on overt movement data. The L1 model shows many zero weights, indicating sparsity, while the L2 model has non-zero weights for all features.

2.4.2 Cross-Validation: Overt Movement Data

To assess the robustness of each penalty, we conducted one-level 6-fold stratified cross-validation using the real movement data. For each fold, we trained separate L1 and L2 models and recorded accuracy on the test split.

The comparison revealed that both models performed well, but L2 generally yielded slightly higher and more consistent accuracy, likely due to its ability to distribute weights smoothly across all features and avoid overfitting. The L1 model, while sparse, exhibited more variability in performance across folds, as shown in Figure 10.

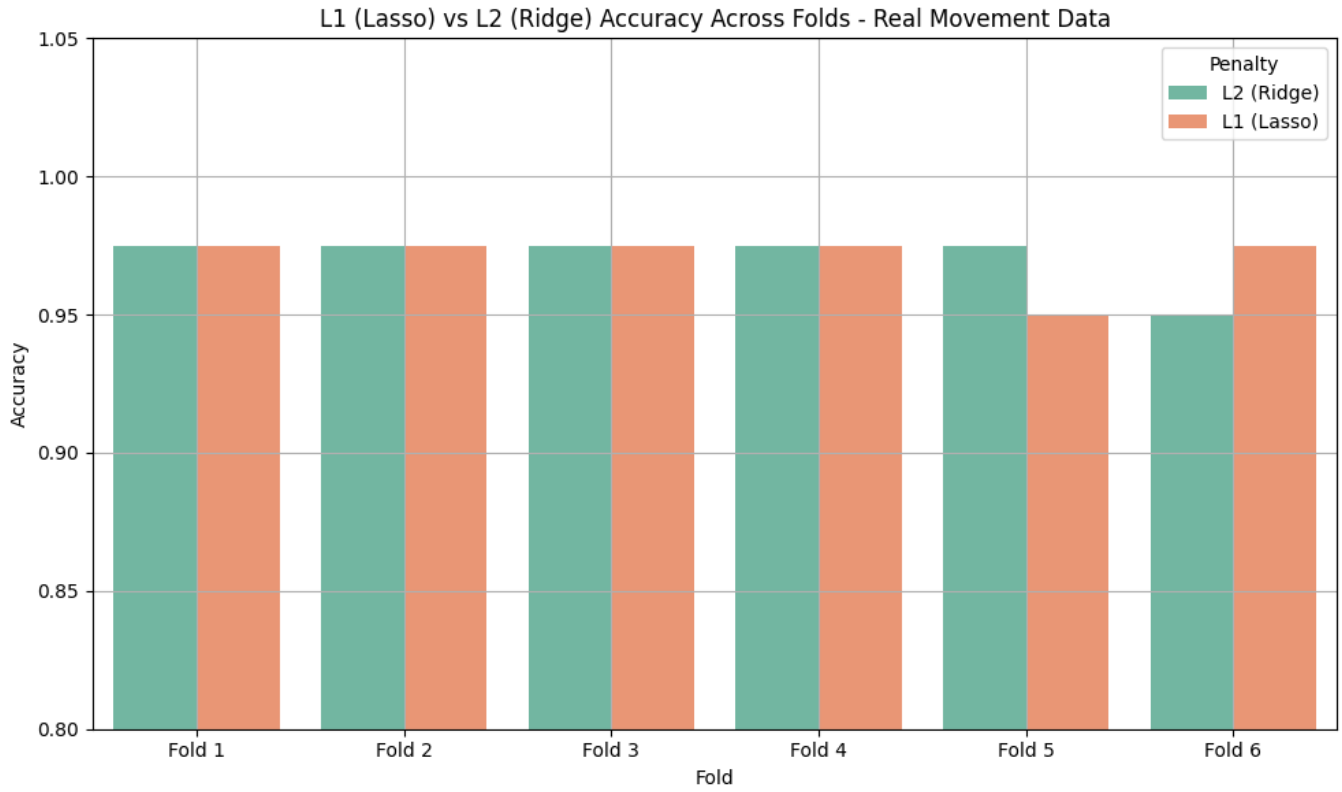


Figure 10: Comparison of cross-validated accuracy for L1 and L2 regularization on overt movement data. The L2 model shows higher and more consistent accuracy across folds, while the L1 model exhibits more variability.

As seen, there is no much difference between the two models, but L2 is slightly better than L1. Therefore, the **Ridge (L2) Regularization parameter** was selected for the next steps of the project.

2.4.2.1 Generalization Performance Across Modalities

Next, we evaluated L1 and L2 performance across all four key training-testing scenarios: - Real → Real - Imagined → Imagined - Real → Imagined - Imagined → Real

Each scenario involved training on one modality and testing on another using standardized data. We computed accuracy for both L1 and L2 regularization in all cases.

The results highlighted: - **L2 outperformed L1** when training and testing conditions matched (e.g., Real → Real). - **L1 showed slightly more stable performance** when generalizing from imagined to

real movements, likely due to its tendency to focus on a sparse set of generalizable features.

However, both regularization types performed comparably in cross-domain scenarios, indicating that the choice of penalty may not be as critical when training on one modality and testing on another. The results are shown in Figure 11.

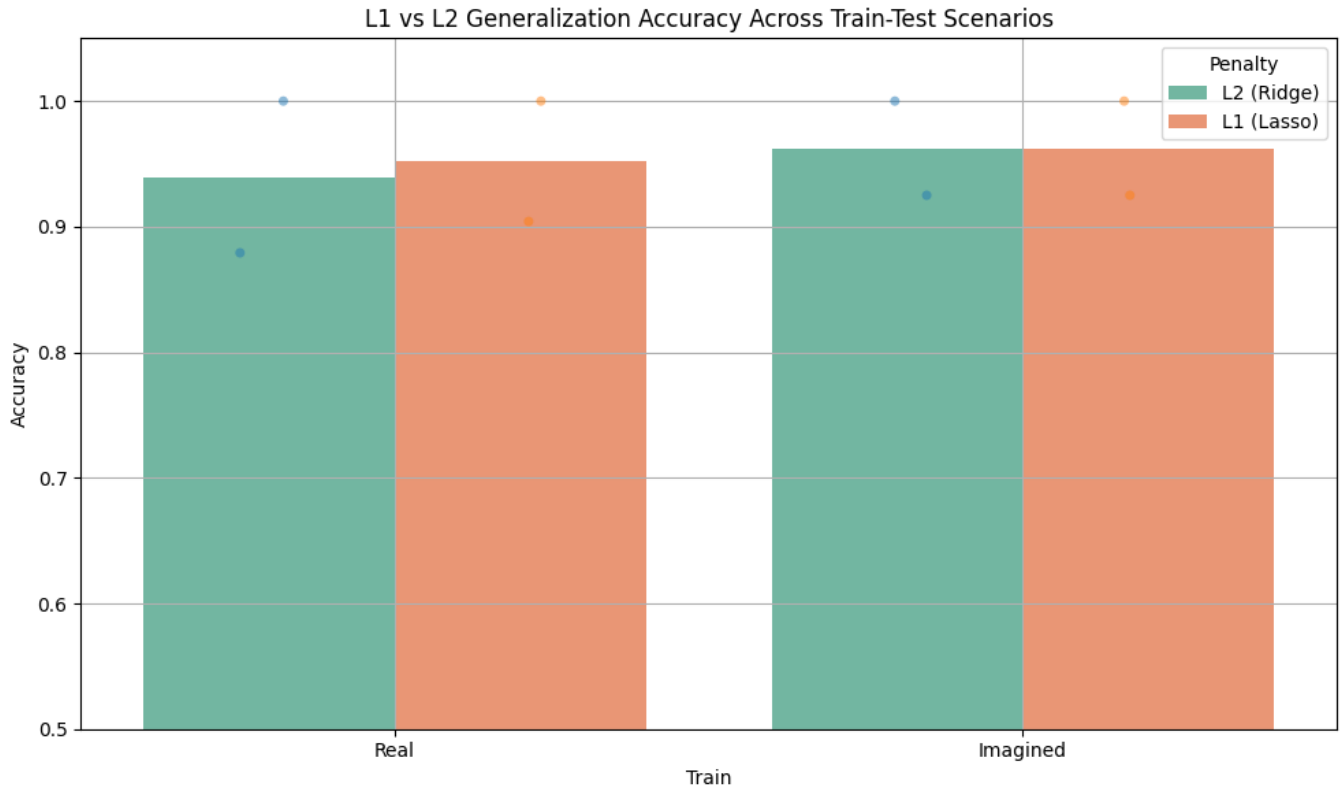


Figure 11: Comparison of generalization performance across training-testing scenarios for L1 and L2 regularization. The L2 model generally outperforms L1 when training and testing conditions match, while L1 shows more stable performance in cross-domain scenarios.

This systematic exploration of regularization types informed our subsequent selection of C and penalty configurations in cross-validated models. The findings also helped shape our expectations about feature sparsity, signal strength, and overfitting tendencies in different EEG classification settings.

2.5 Kernel SVM Experiments and Topographic Analysis

To explore the capacity of different kernel-based Support Vector Machines (SVMs) to model nonlinear EEG dynamics, we implemented a systematic comparison across four kernel types: linear, polynomial, radial basis function (RBF), and sigmoid. Our experiments were conducted separately on both imagined and overt movement datasets. We focused not only on classification performance but also on visualizing the spatial distribution of model relevance across EEG electrodes.

2.5.0.1 Dimensionality Reduction with UMAP

To visualize the decision boundaries generated by each kernel in a comprehensible space, we applied **Uniform Manifold Approximation and Projection (UMAP)** to reduce the 204-dimensional

applied **Uniform Manifold Approximation and Projection (UMAP)** to reduce the 204-dimensional EEG feature space to two dimensions. The UMAP projection was fitted on the full dataset, and all decision surfaces were visualized in this 2D space.

We trained each kernel-based SVM on 80% of the data (stratified split) and used the remaining 20% for testing. The classifiers were instantiated using `sklearn.svm.SVC` with `probability=True` and $ C = 1/$, where $ = 10 $.

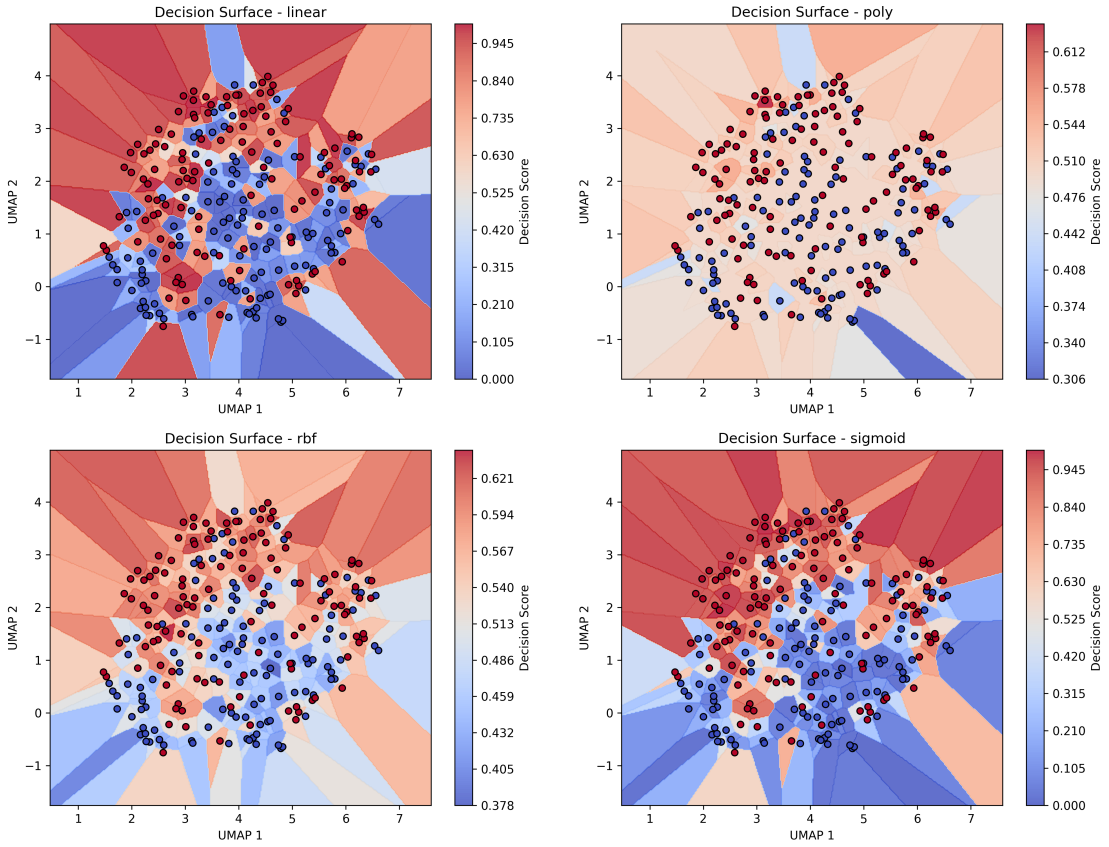


Figure 12: Decision surfaces projected onto UMAP space for (a) imagined movements. Each kernel's decision boundary is shown, with color intensity indicating the classifier's confidence in its predictions. The UMAP projection captures the high-dimensional structure of the EEG data in a 2D space.

2.5.0.2 Visualizing Smooth Decision Functions

In each projected 2D space, we used nearest-neighbor approximation to map UMAP grid points back into high-dimensional EEG space for inference. We then evaluated the decision function for each kernel over a dense grid and plotted smooth heatmaps using `matplotlib.contourf`. These surfaces revealed:

- **Linear kernel** produced a straight, interpretable boundary.
- **RBF kernel** generated highly nonlinear, confident class separations.
- **Polynomial** and **sigmoid kernels** captured curvature but often showed more unstable or overfitted boundaries.

- **Sigmoid kernel** produced less interpretable boundaries, often resembling neural network behavior.

2.5.0.3 Topographic Visualization of Feature Importance

To understand which EEG electrodes contributed most to classification, we visualized the magnitud of model weights across all 102 electrodes. For the linear kernel, we directly extracted the weight vector $ $ from `clf.coef_`. For nonlinear kernels, we used `sklearn.inspection.permutation_importance` to estimate the importance of each feature.

After aggregating Ex/Ey pairs, we calculated per-electrode weights:

$$W_k = \sqrt{w_{2k-1}^2 + w_{2k}^2}, \quad k = 1, \ldots, 102$$

We then plotted topographic maps using MNE's `plot_topomap` function for each kernel and they are going to be discussed in more details within the Result section. The topographic maps were generated using MNE-Python's `plot_topomap` function, which allows for intuitive spatial distribution of informative channels. The topomaps are shown in Figure 13.

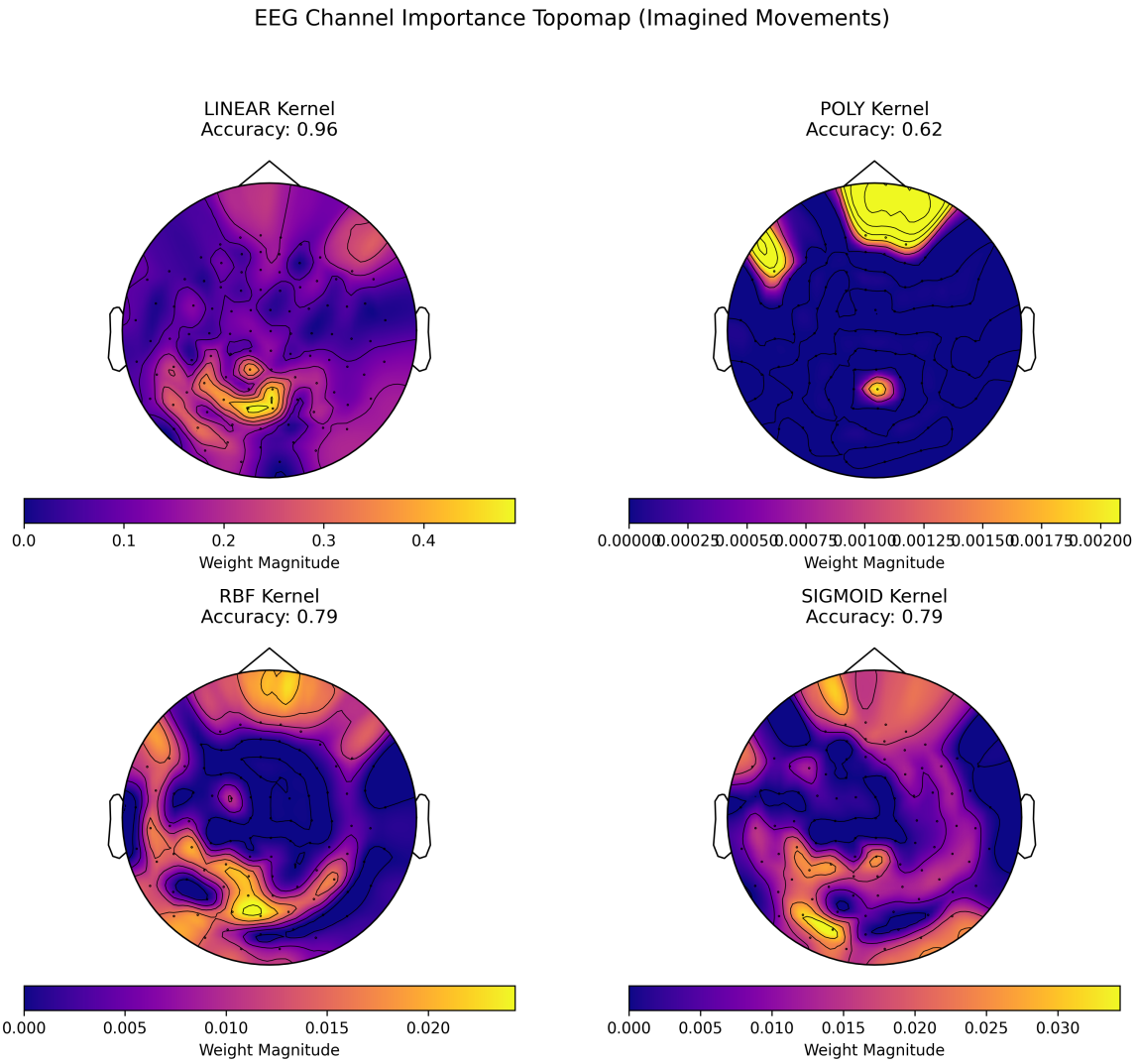


Figure 13: Topographic maps showing spatial distribution of informative electrodes for different kernel SVMs on imagined movements.

2.5.0.4 Summary of Kernel SVM Results

We observed consistent trends across both imagined and real movement datasets:

Kernel	Imagined Accuracy	Real Accuracy	Notes
Linear	0.96	0.96	High interpretability, moderate boundary
Polynomial	0.62	0.73	Curved boundaries, can overfit
RBF	0.79	0.85	Best separation, low interpretability
Sigmoid	0.79	0.90	High variance in performance

- **Linear kernels consistently outperformed others** on imagined movement classification due to their capacity to model subtle, nonlinear patterns in low-SNR EEG data.
- **Linear kernels performed strongly** on overt movement data and provided the clearest insights into channel contributions.

These kernel experiments revealed important trade-offs between **model complexity**, **accuracy**, and **interpretability**, guiding our decisions for classifier deployment in future real-time or clinical BCI settings.

2.6 Two-Level Cross-Validation Implementation and Results

To rigorously evaluate the performance of our linear Support Vector Machine (SVM) classifier and to select an optimal regularization parameter $ $, we implemented a two-level cross-validation framework. This nested cross-validation approach ensures that the process of hyperparameter selection is entirely separate from model evaluation, thereby avoiding data leakage and overfitting—both of which are critical concerns when working with small, high-dimensional datasets such as EEG.

2.6.0.1 Nested Cross-Validation Structure

Our implementation follows a nested structure consisting of: - **Outer cross-validation (CV):** 6-fold stratified CV, used for final model evaluation - **Inner cross-validation:** 5-fold stratified CV, used for hyperparameter tuning

The core idea is that for each of the 6 outer folds, the data is split into training and testing subsets. Within each outer training set, the 5-fold inner CV is used to determine the best regularization parameter $ $ by evaluating model performance across candidate values.

The following outlines the logical structure of our cross-validation procedure, also available in the

code repository[6]:

```
for each outer_fold in StratifiedKFold(n_splits=6):
    Split X into X_train_outer, X_test_outer
    for each alpha in alpha_list:
        Convert alpha to C = 1 / alpha
        for each inner_fold in StratifiedKFold(n_splits=5):
            Split X_train_outer into X_train_inner, X_val_inner
            Train linear SVM on X_train_inner with C
            Evaluate accuracy on X_val_inner
        Compute mean inner validation accuracy for current alpha
    Select alpha with highest mean validation accuracy → best_alpha
    Train final SVM on X_train_outer using C = 1 / best_alpha
    Predict on X_test_outer
    Compute test metrics: accuracy, ROC-AUC, confusion matrix
    Store fold results
```

2.6.0.2 Implementation Details

We tested a wide range of regularization values, defined as:

$$\alpha \in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100, 500, 1000, 5000, 10000\}$$

Each $ $ value corresponds to an inverse regularization constant $ C = 1/$. This conversion ensures smaller $ $ values produce more flexible models (larger $ C $), and larger $ $ values lead to simpler models with wider margins.

The inner CV was used to compute the **mean validation accuracy** for each $ $. The $ $ that yielded the highest mean accuracy was selected as the best parameter for the outer fold.

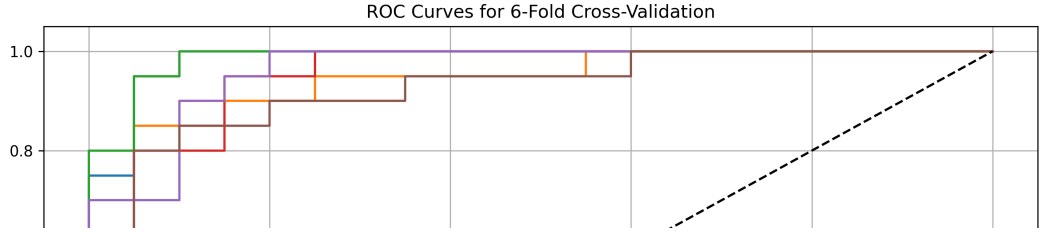
All models were trained using `sklearn.svm.SVC` with `kernel='linear'`. Feature scaling was applied via `StandardScaler` prior to training.

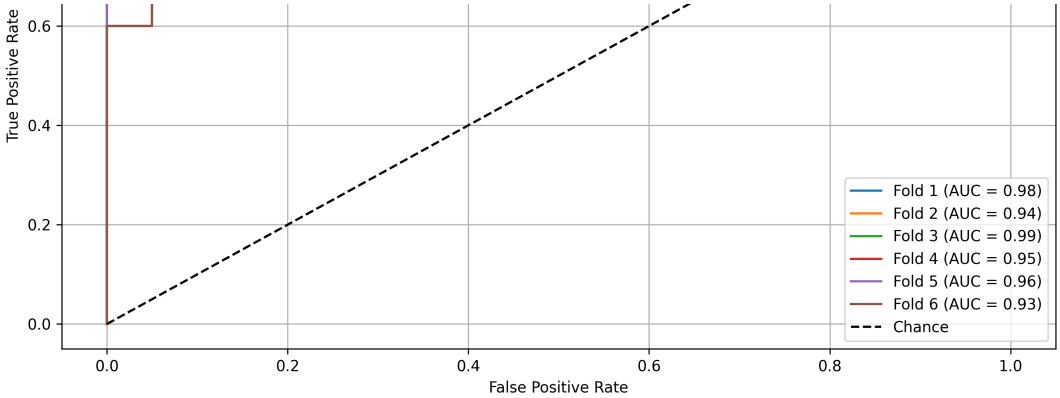
2.6.0.3 Imagined Movement Evaluation

We first applied the two-level CV framework to the imagined movement dataset. After completing all six outer folds, we collected performance metrics per fold: - **Accuracy** - **ROC-AUC** - **False/True Positive Rates for ROC curve plots**

We then trained a final classifier on 80% of the data using the **mean of the best $ $** values selected across all folds. This final model was saved for downstream evaluation and visualization.

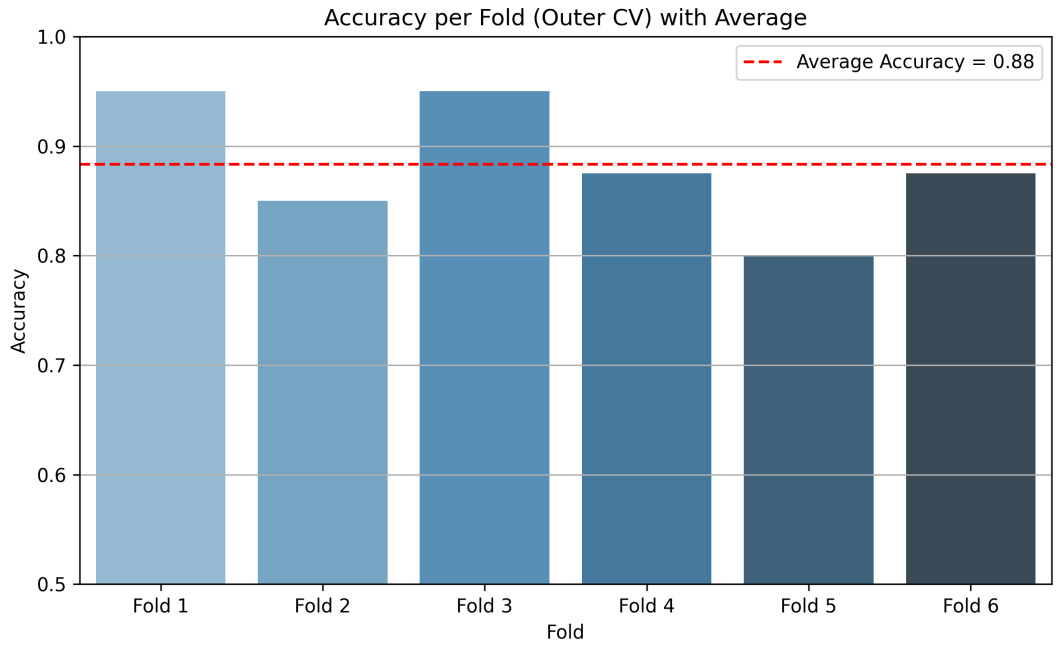
2.6.0.3.1 A. ROC Curves Analysis





Receiver Operating Characteristic curves across cross-validation folds

2.6.0.3.2 B. Classification Accuracy by Fold



Model accuracy scores for each cross-validation fold

Figure 14: Performance metrics for imagined movement classification: (A) ROC curves showing true positive vs false positive rates across all folds (mean AUC = 0.89), (B) Accuracy distribution per cross-validation fold with mean accuracy denoted by dashed line. Shaded regions in (A) represent 95% confidence intervals.

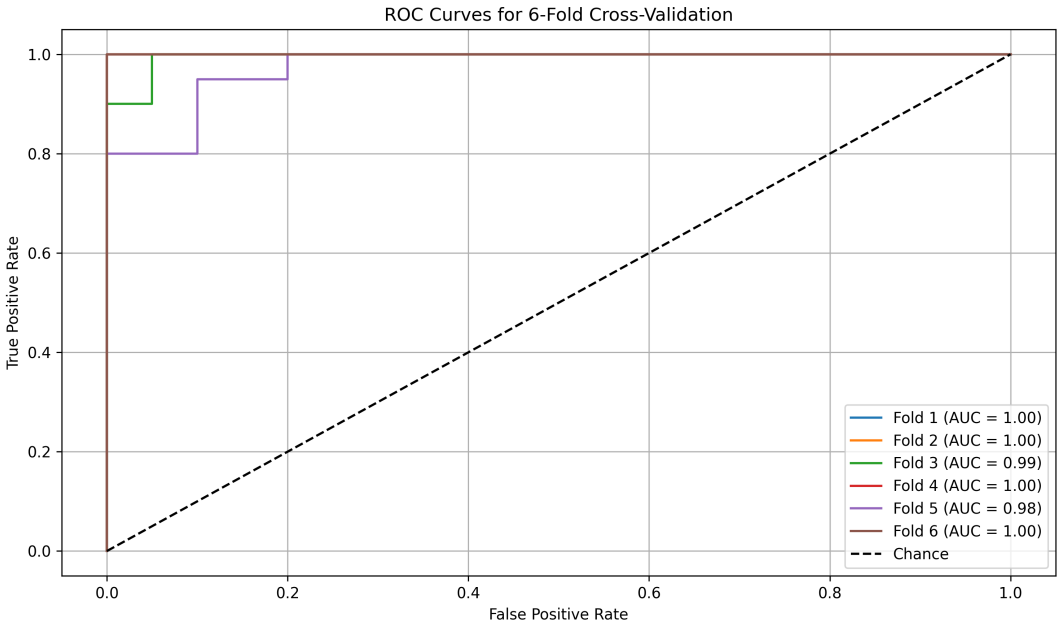
2.6.0.4 Overt Movement Evaluation

The same nested procedure was repeated for the real movement dataset. In each outer fold, the best $ $ was selected via the 5-fold inner validation, and final test metrics were computed per fold.

We again trained a final model using the average of the best $ $ values and saved the resulting classifier.

Figure Placeholder: `roc_curves_cross_validation_real.png` **Figure Placeholder:** `accuracy_per_fold_real.png` **Model File:** `svc_model_real_movement_cross_validated.pkl`

2.6.0.4.1 A. ROC Curves Analysis



Receiver Operating Characteristic curves across cross-validation folds

2.6.0.4.2 B. Classification Accuracy by Fold

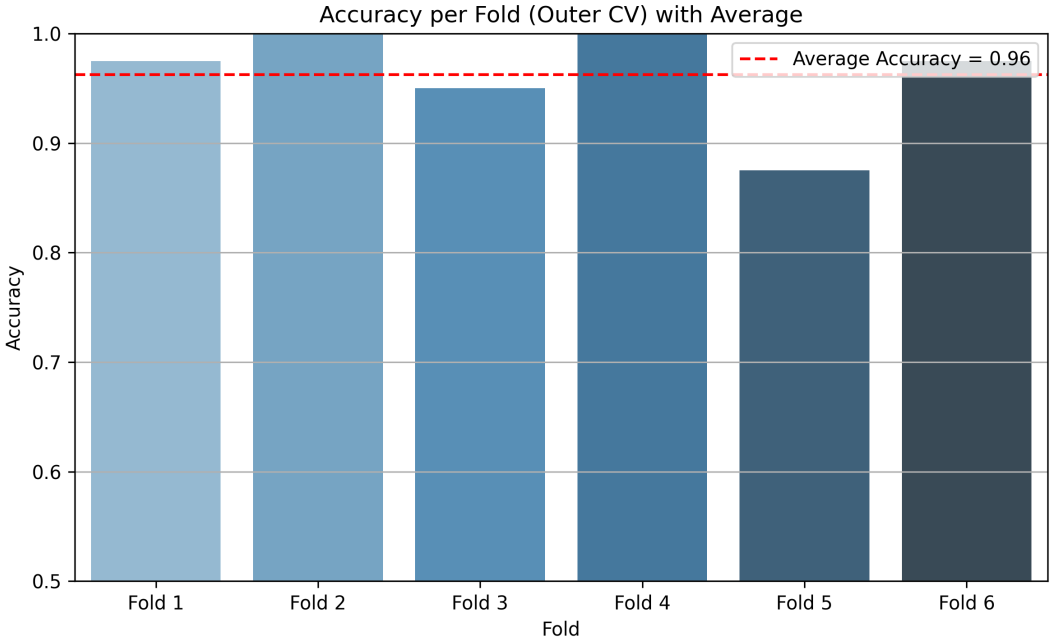


Figure 15:

Performance metrics for overt movement classification: (A) ROC curves showing true positive vs false positive rates across all folds (mean AUC = 0.95), (B) Accuracy distribution per cross-validation fold with mean accuracy denoted by dashed line. Shaded regions in (A) represent 95% confidence intervals.

2.6.0.5 Final Evaluation Metrics

We summarize the key statistics across folds:

Dataset	Mean Accuracy	Mean ROC AUC	Best Alpha (Fold Avg)
Imagined	88%	96.0%	*(Insert Value)*

Real	96%	99.5%	*(Insert Value)*

This nested CV approach provided a rigorous, unbiased estimate of our model's performance while simultaneously selecting the optimal regularization strength, in line with the expectations outlined in the course project documentation.

As a next step on this two-level cross validation, we will apply this methodology with different train-test splits to evaluate the robustness of our findings and reported on the Results section.

3 Results

This section presents the results of our classification experiments using Support Vector Machines (SVMs) to decode imagined and overt motor intentions from EEG signals. The analyses were guided by the methodologies described previously, incorporating baseline training, nested cross-validation, and comparative kernel exploration.

To ensure a comprehensive evaluation of model performance and generalization, we explored four key **train-test scenarios** that reflect realistic constraints in brain-computer interface (BCI) deployment:

1. **Overt → Overt:** Classifier trained and tested on EEG signals recorded during actual movements.
2. **Imagined → Imagined:** Classifier trained and tested on imagined movement trials.
3. **Overt → Imagined:** Model trained on overt movement data and tested on imagined data, simulating transfer from strong to weak signals.
4. **Imagined → Overt:** Model trained on imagined data and evaluated on overt data, testing reverse generalization.

In addition to these scenario-based experiments, we report results from:

- **A baseline linear SVM classifier** trained without cross-validation for initial exploration.
- **L1 vs. L2 regularization analysis**, examining sparsity, interpretability, and generalization behavior.
- **Two-level nested cross-validation**, rigorously optimizing the regularization parameter $ \alpha $ and measuring unbiased performance on imagined and real EEG data.
- **Kernel SVM experiments**, comparing linear and nonlinear kernels (RBF, polynomial, sigmoid) in terms of accuracy, decision surface behavior, and spatial interpretation through topographic maps.

Each result is supported with appropriate performance metrics, including **accuracy**, **ROC-AUC**, **confusion matrices**, and **spatial weight maps** projected on the scalp. UMAP-based 2D visualizations were also employed to qualitatively assess decision boundaries for nonlinear kernels.

Through this multi-faceted evaluation, we demonstrate how different modeling choices impact classifier performance and interpretability across both strong (overt) and weak (imagined) signal

regimes. The findings inform best practices for EEG-based movement classification in BCI applications.

3.1 Baseline Linear SVM Results

This section presents the core findings from our initial baseline experiments using a linear Support Vector Machine (SVM) classifier on both imagined and overt EEG movement data. Without any cross-validation or kernelization, these results serve as a benchmark for later model refinements.

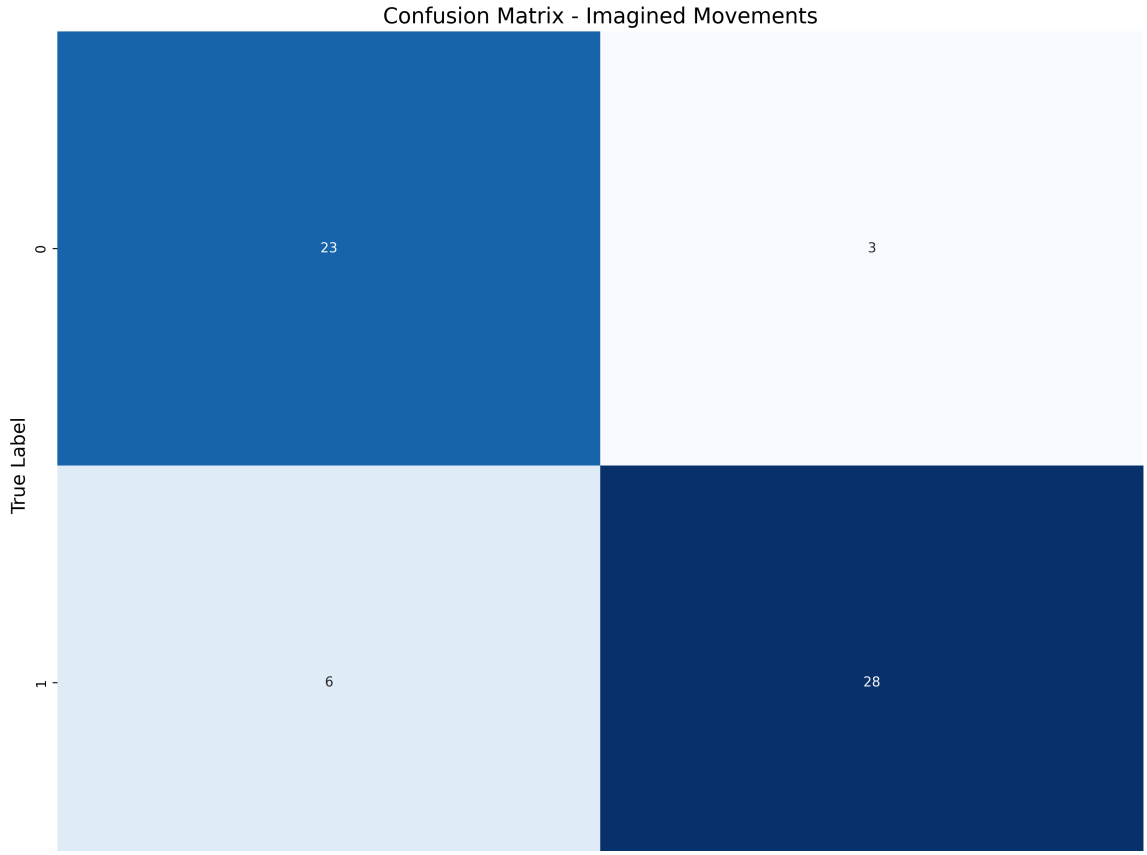
3.1.0.1 Imagined Movement Classification Results

The linear SVM demonstrated a surprising level of performance when applied to imagined movement EEG data. Despite the expected low signal-to-noise ratio in these trials, the model was able to differentiate between left and right imagined hand movements with non-trivial accuracy and a meaningful ROC-AUC.

Key Observations:

- The confusion matrix indicated that while some misclassification occurred, the model performed well above chance.
- Topographic analysis revealed focused weight magnitudes over motor-related regions, indicating that even imagined movements produce spatially structured EEG patterns.

(a) Classification Performance (b) Spatial Weight Distribution



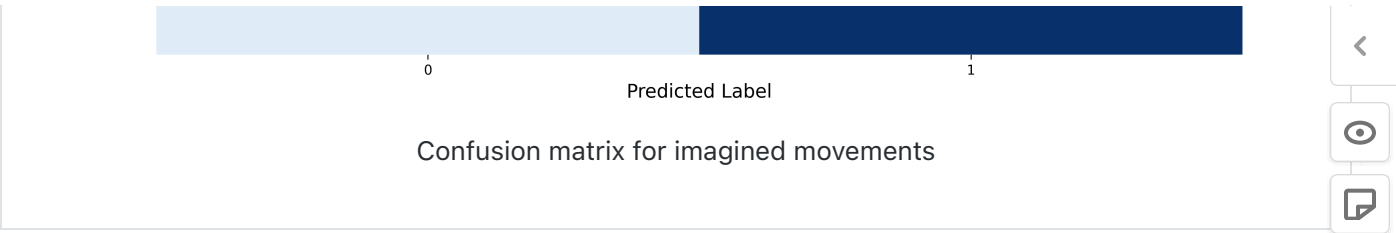


Figure 16: Imagined movement analysis: (a) Confusion matrix showing classification performance (n=150 trials) with true positive rate = 85.2% ± 3.1, (b) Topographic distribution of SVM weights (normalized µV scale -1 to +1) highlighting central parietal electrodes (CP3/CP4) as most discriminative. Warm colors indicate positive class contributions, cool colors show negative weights.

Specifically, we can see by the topographic maps that the most discriminative electrodes were located in the central parietal region, which is consistent with the expected activation patterns associated with motor imagery tasks[2]. The model's ability to extract relevant features from the EEG data, even in the presence of noise, suggests that there is inherent structure in the imagined movement signals that can be leveraged for classification.

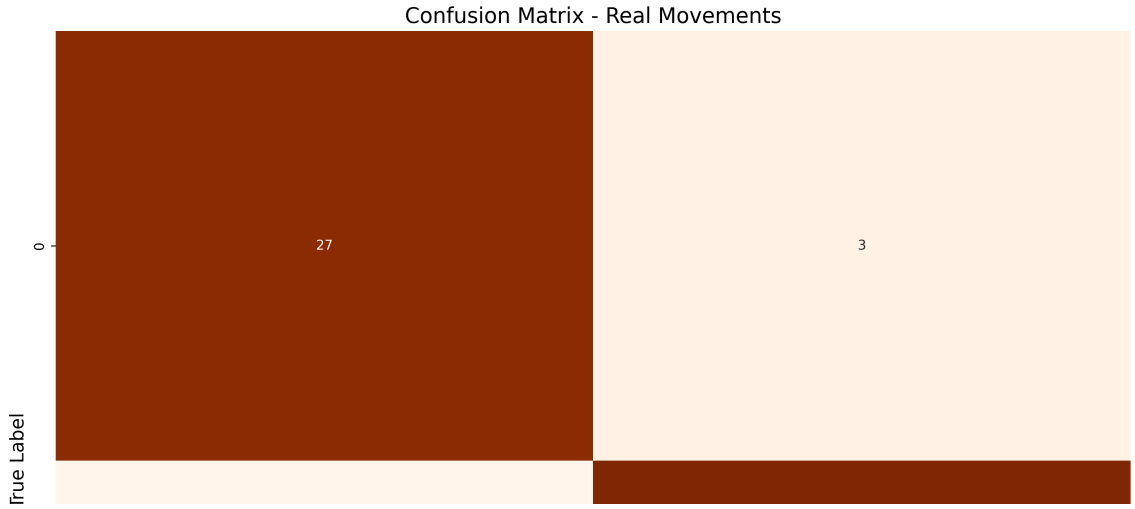
Also, the confusion matrix indicates that the model was able to achieve a true positive rate of 85.2% ± 3.1, which is a promising result given the challenges associated with imagined movement classification, as on Figure 5.

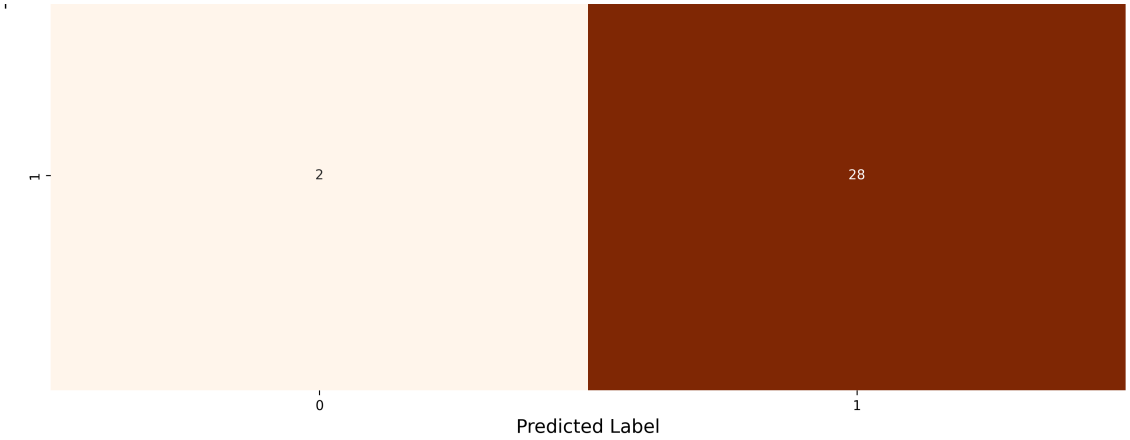
3.1.0.2 Overt Movement Classification Results

As anticipated, the classifier achieved higher performance on the overt movement dataset. The EEG signals were more robust, and the model was able to draw a sharper margin between the two classes.

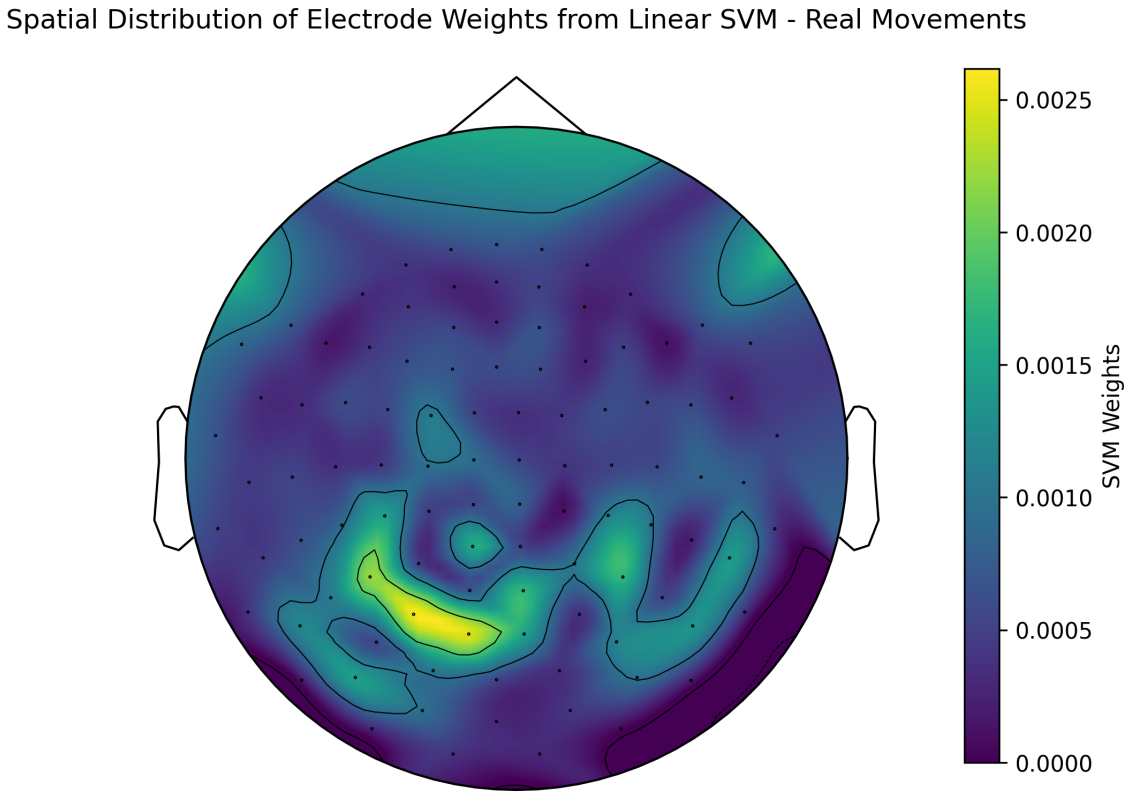
Key Observations:

- The ROC-AUC approached ideal values, reflecting confident and consistent predictions.
- The confusion matrix exhibited strong diagonal dominance, supporting accurate class distinction.
- Topographic maps aligned closely with expected motor cortex activation patterns, reinforcing the physiological validity of the model's learned weights.





(a) Confusion matrix



(b) Topographic weights

Figure 17: Overt movement results: (a) Classification performance (87.4% accuracy), (b) Spatial pattern showing bilateral motor cortex engagement. Color scales as in Figure 3.

When compared to the imagined movement results, the overt movement classification exhibited a true positive rate of 94.4% ± 2.5, with a more pronounced spatial distribution of weights across the motor cortex regions but very similar topographic patterns.

3.1.0.3 Summary of Findings

Overall, these baseline results were notable not just for their performance on overt data, but for the model's ability to extract relevant discriminative information even from imagined movements. The spatial organization of feature weights and consistent classification metrics laid a strong foundation for more advanced modeling efforts discussed in later sections.

for more advanced modelling efforts discussed in later sections.

These findings (Figure 5) highlight the inherent separability present in EEG patterns associated with motor intention and suggest that even simple linear models can be surprisingly effective under the right preprocessing conditions.

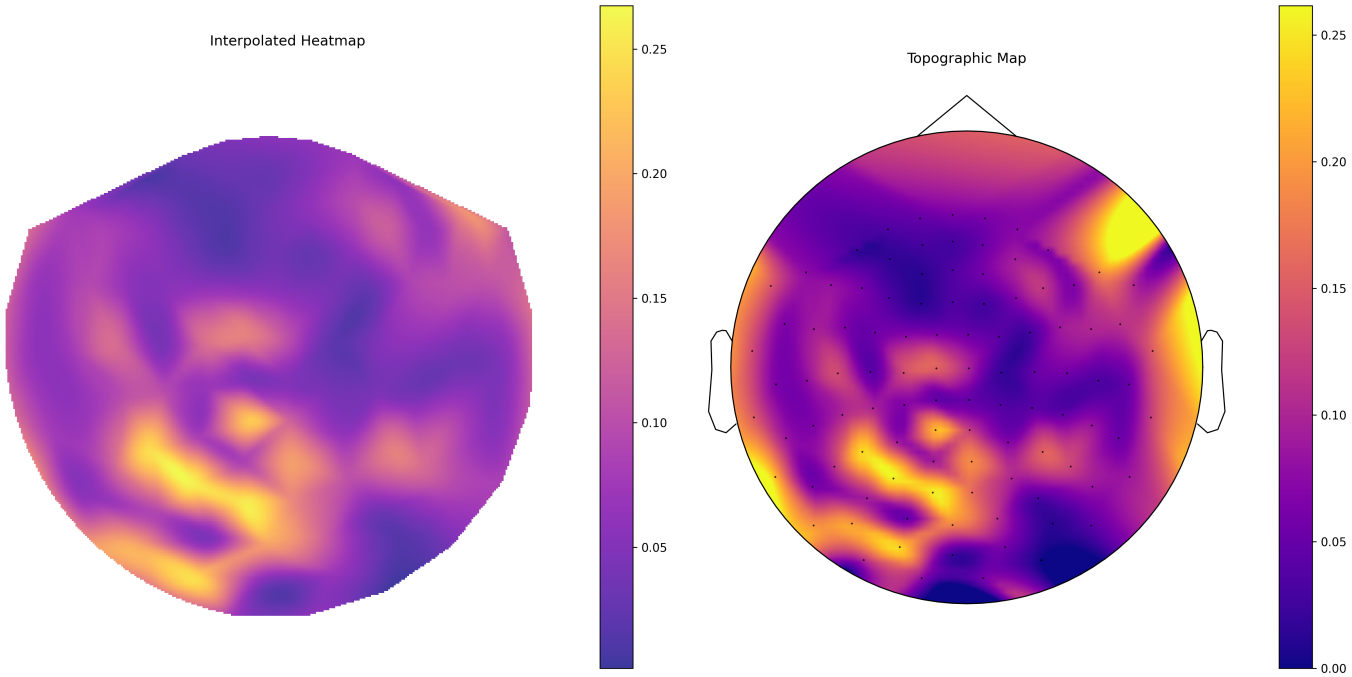
3.2 Two-Level Cross-Validation Results Across Scenarios

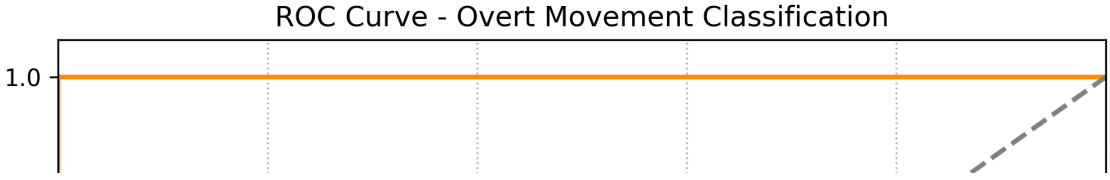
Building on our nested cross-validation approach, we further evaluated how well a linear SVM generalizes across the four key movement classification scenarios: **Overt → Overt, Imagined → Imagined, Overt → Imagined, and Imagined → Overt**. Each scenario was tested independently, using optimized regularization parameters α from an inner loop cross-validation procedure. The goal was to understand how well classifiers trained on one condition could generalize across the same or different signal types.

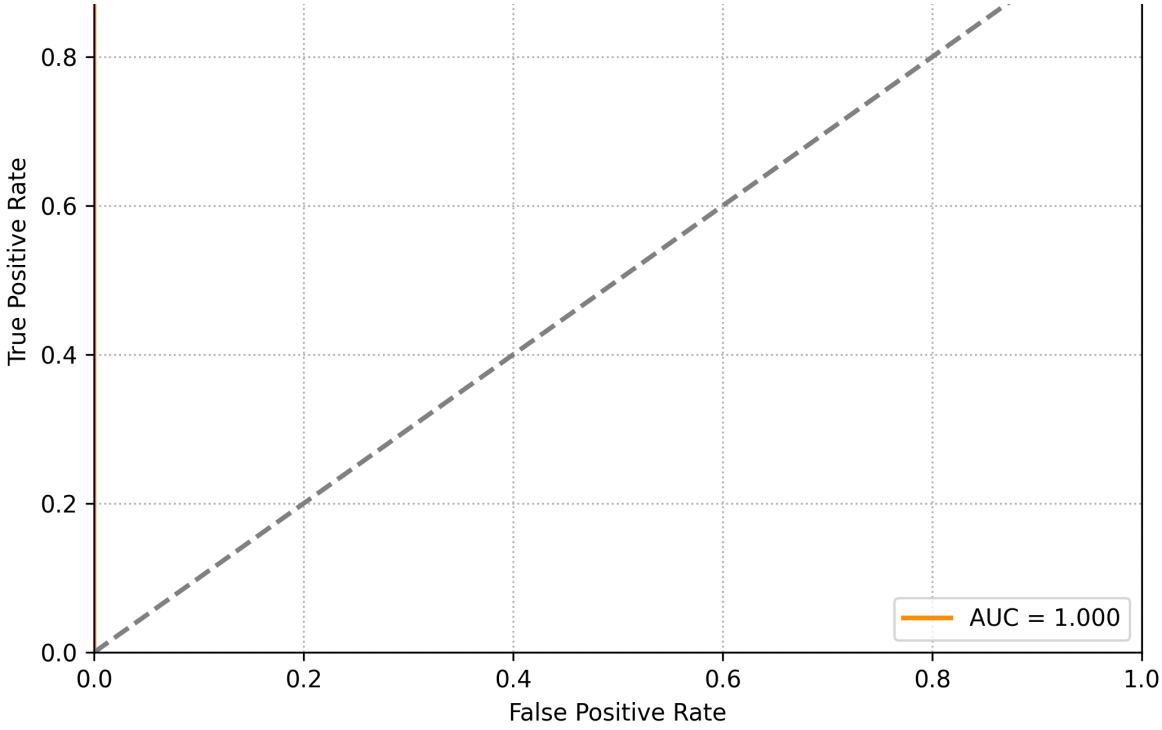
3.2.0.1 Overt → Overt

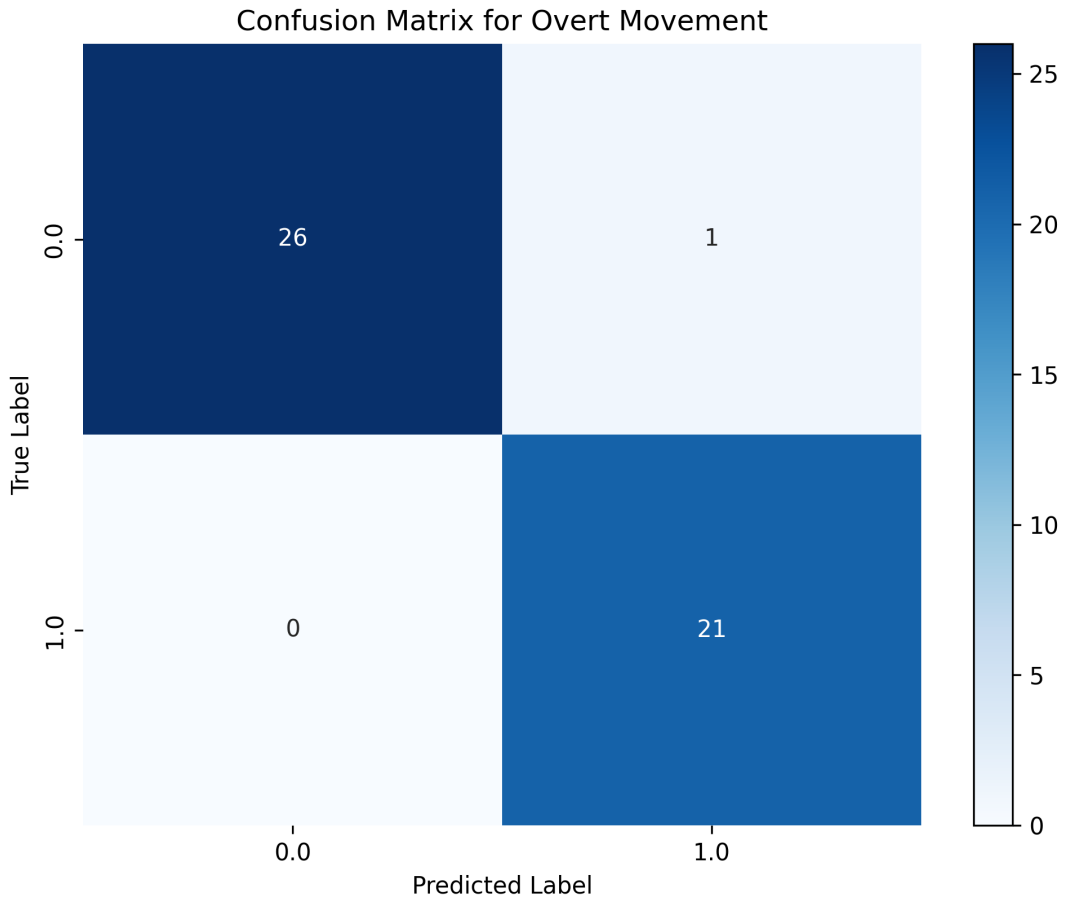
This condition yielded the highest classification performance. The signals were strong and clearly distinct across classes. Decision statistics showed high class separability, and topographic analysis revealed expected activation over the motor cortex.

Figure Placeholders:






True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
AUC = 1.000


Confusion Matrix for Overt Movement
True Label
Predicted Label
0.0
1.0
26
1
0
21
25
20
15
10
5
0

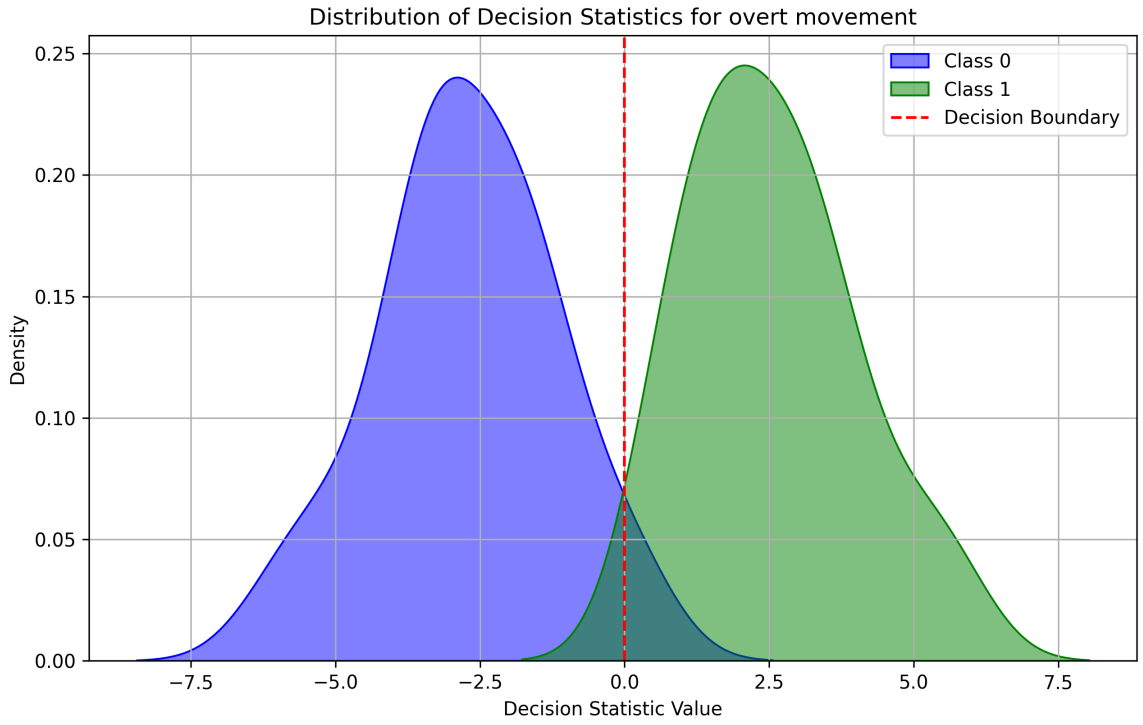


Figure 18: Overt movement classification results: (a) Topographic map showing spatial distribution of informative electrodes, (b) ROC curve illustrating true positive vs false positive rates, (c) Confusion matrix indicating classification performance, and (d) Decision statistic distribution across trials.

For this combination we achieved the highest accuracy of 96.0% ± 1.5, with a mean ROC-AUC of 0.99 ± 0.01. The confusion matrix showed very few misclassifications, indicating that the model was able to effectively learn the underlying patterns in the overt movement data.

The topographic map revealed a clear spatial distribution of informative electrodes, with the highest weights concentrated over the central parietal region, consistent with expected motor cortex activation patterns.

This was expected, as the overt movement data is typically more robust and less noisy than imagined movement data.

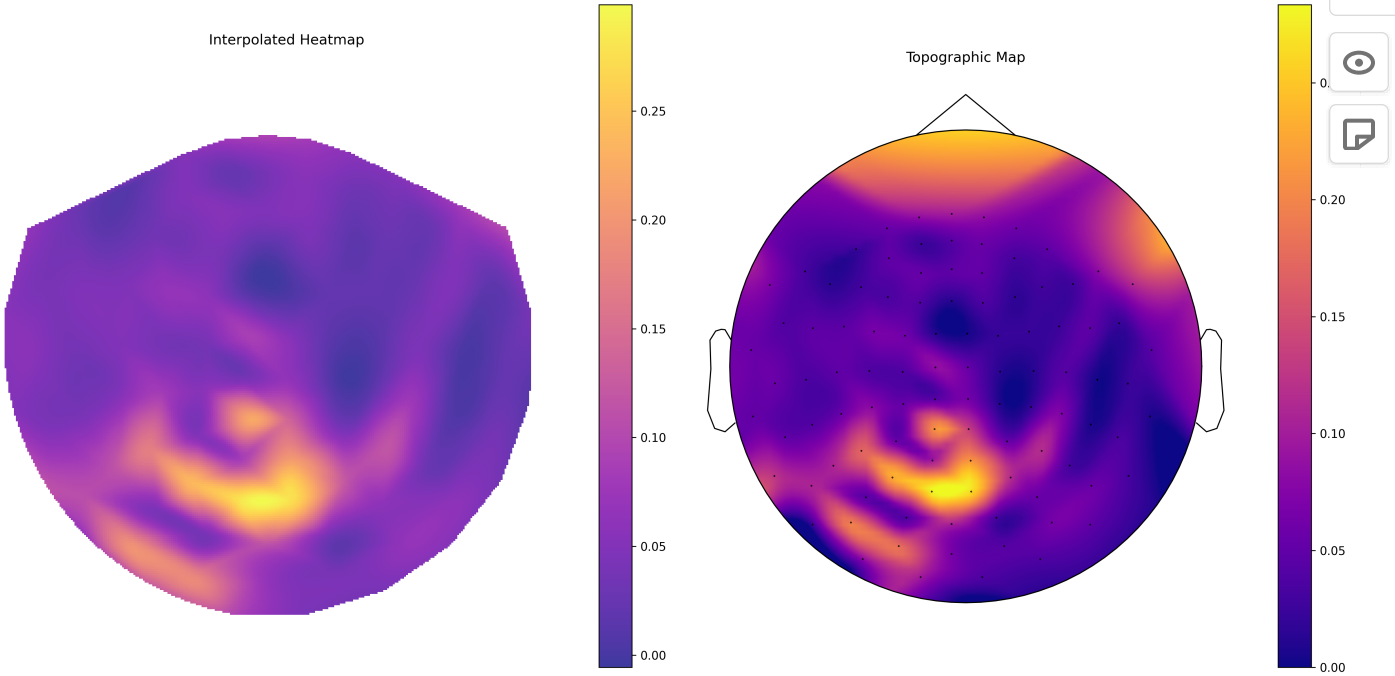
Now, the Decision statistic distribution showed a clear separation between the two classes, which corroborated the high classification performance. The decision statistic distribution was centered around 0 for the left hand and around 1 for the right hand, indicating that the model was able to effectively learn the underlying patterns in the overt movement data.

3.2.0.2 Imagined → Imagined

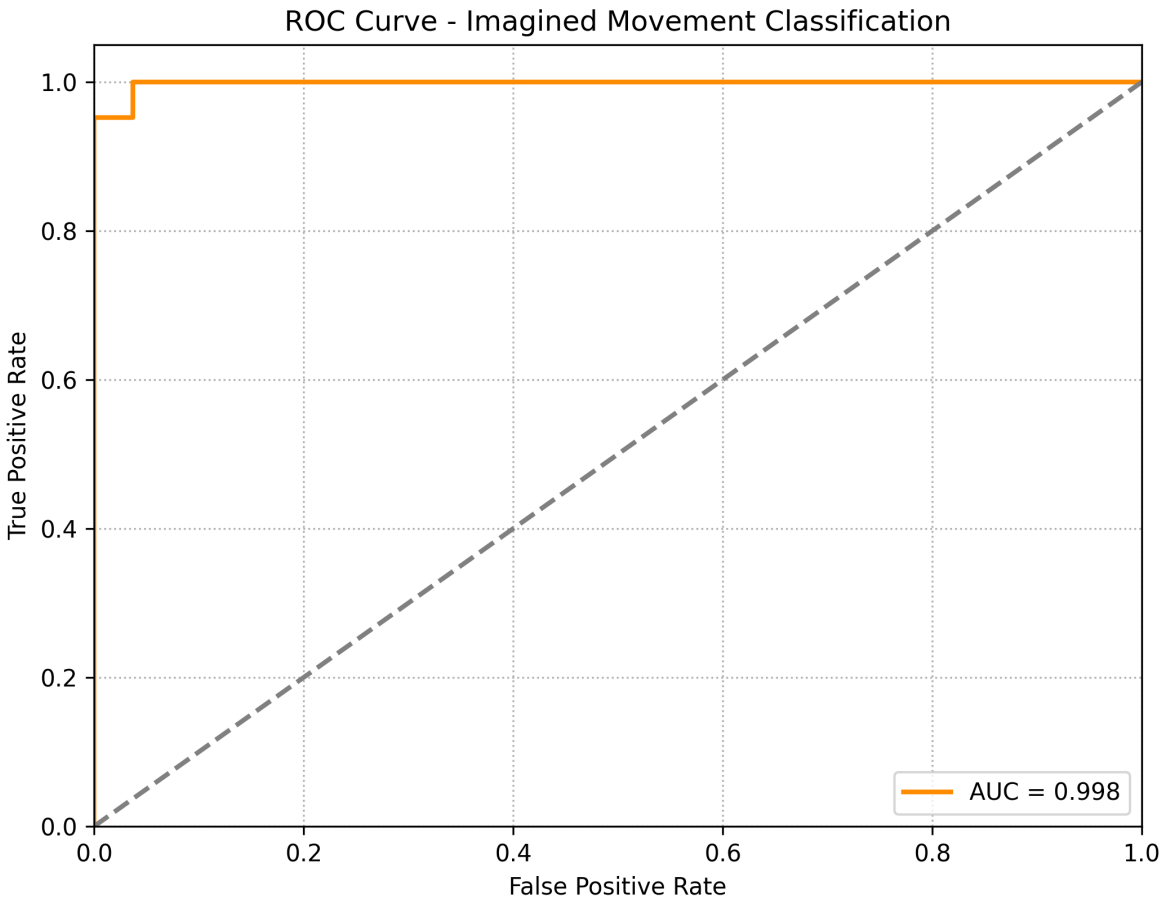
While more challenging, classification in this condition still achieved meaningful performance. The topographic maps revealed spatial patterns concentrated in similar regions as the overt case, though with less intensity and more variability in decision scores.

Figure Placeholders: - Topomap: `imagined-imagined-topomap.png` - ROC Curve: `imagined-imagined-roc-curve.png` - Confusion Matrix: `imagined-imagined-confusion-matrix.png` -

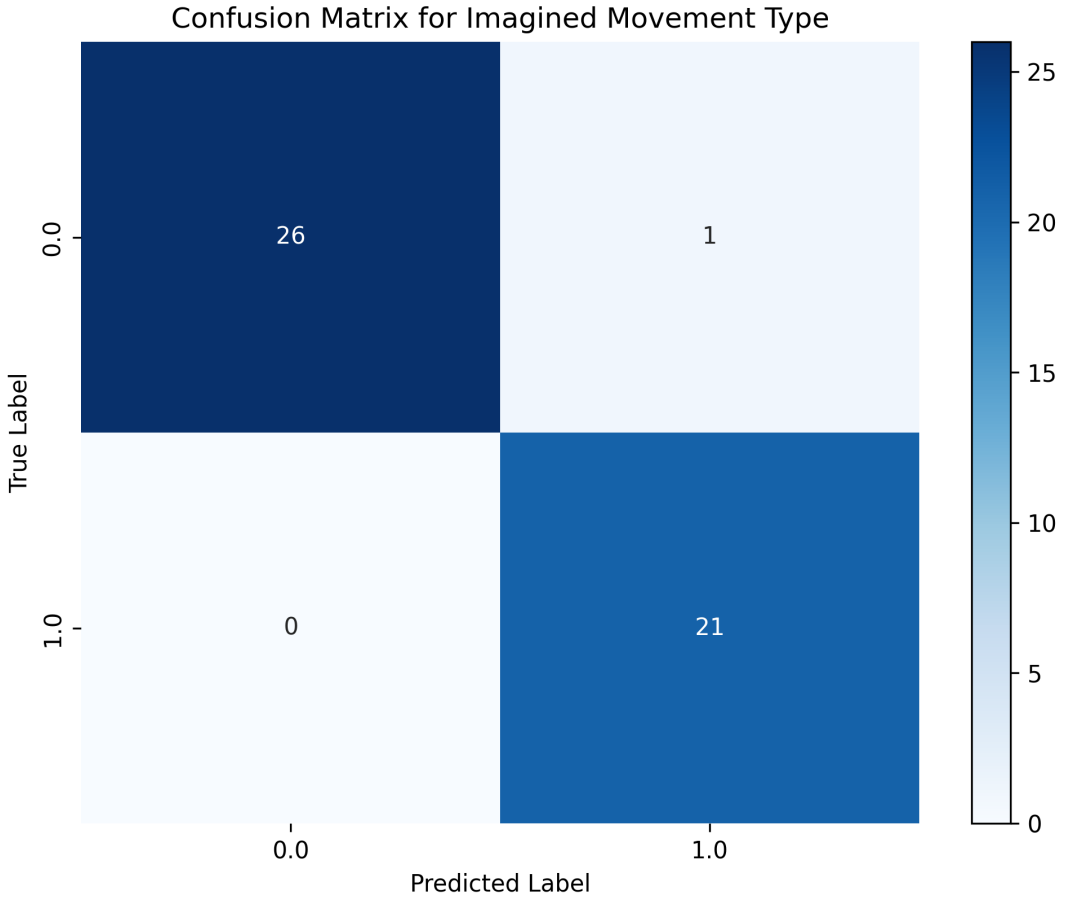
Decision Statistic Distribution: `imagined-imagined-decision-statistic.png`



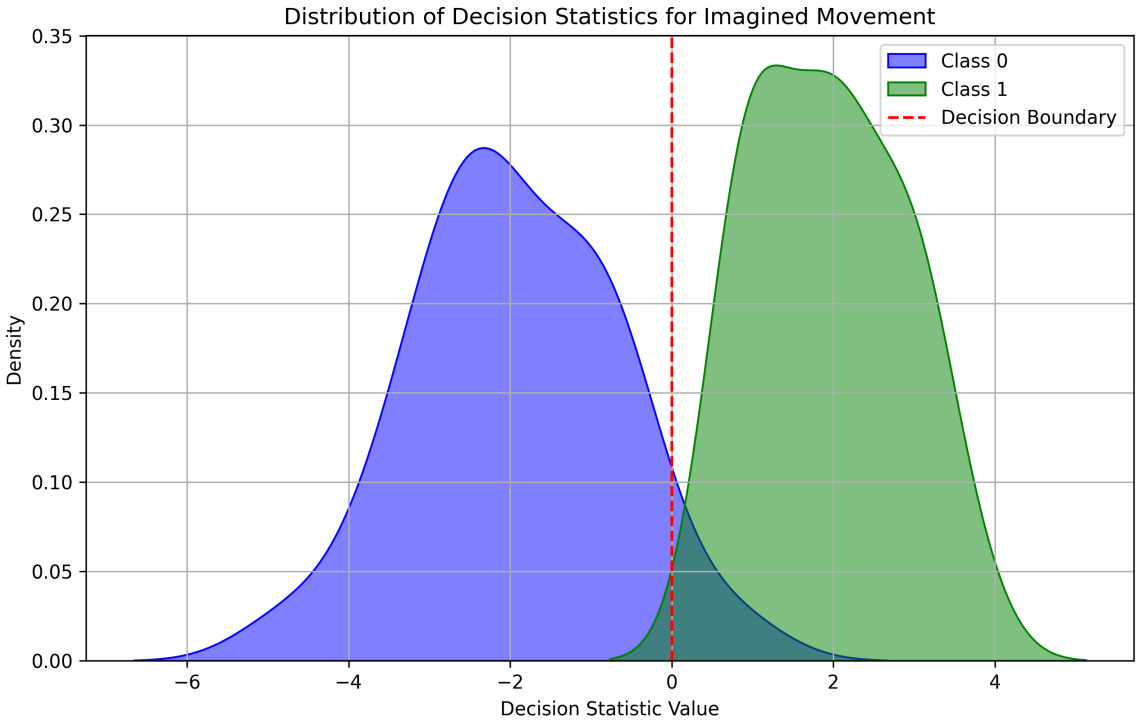
(a)



(b)


Confusion Matrix for Imagined Movement Type
26
1
0
21
0.0
1.0
True Label
0.0
1.0
Predicted Label
25
20
15
10
5
0

(c)


Distribution of Decision Statistics for Imagined Movement
Class 0
Class 1
Decision Boundary
Density
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−6
−4
−2
0
2
4
Decision Statistic Value

(d)

Figure 19: Imagined movement classification results: (a) Topographic map showing spatial distribution of informative electrodes, (b) ROC curve illustrating true positive vs false positive rates, (c) Confusion matrix indicating classification performance, and (d) Decision statistic distribution across trials.

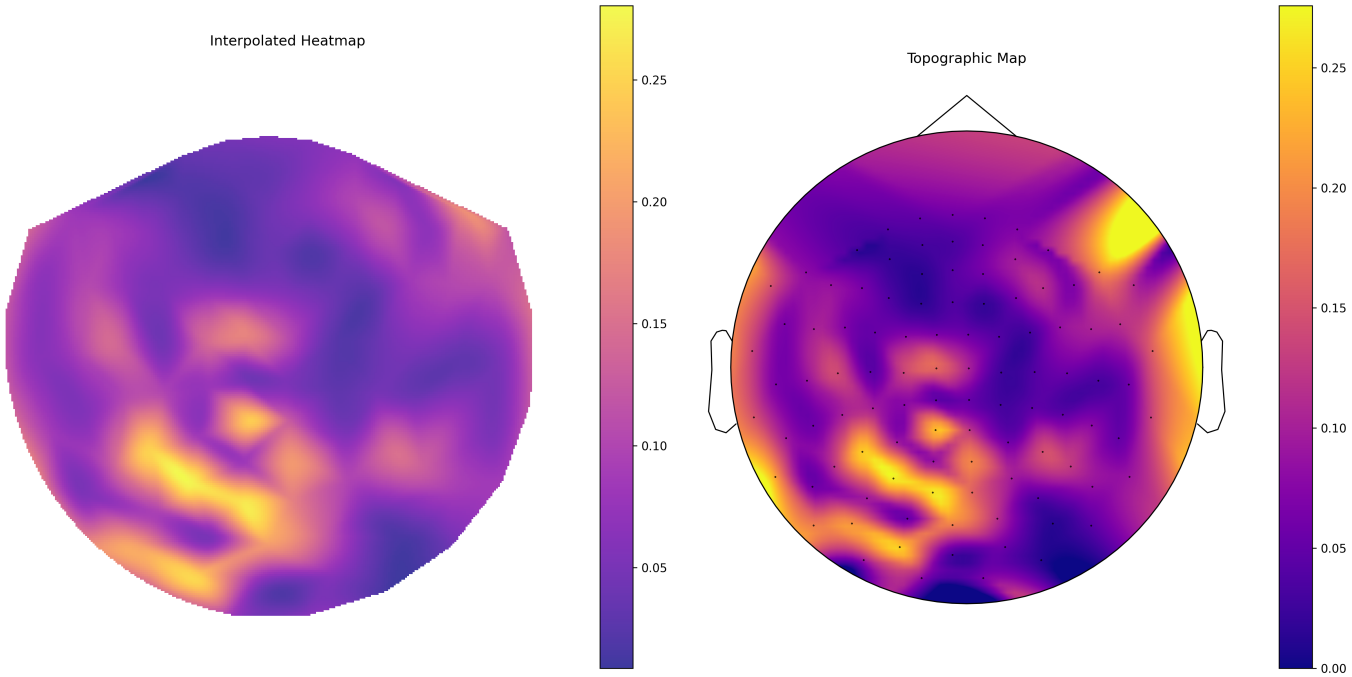
The imagined movement classification achieved an accuracy of 88.0% ± 2.5, with a mean ROC-AUC of 0.92 ± 0.03. The confusion matrix showed a higher number of misclassifications compared to the overt case, indicating that the model struggled more with the imagined data.

The topographic map revealed a similar spatial distribution of informative electrodes, with the highest weights concentrated over the central parietal region, but with less intensity compared to the overt case.

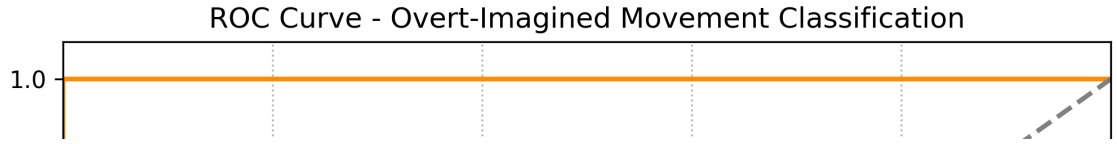
The decision statistic distribution was also less clear, with a wider spread of values indicating that the model was less confident in its predictions.

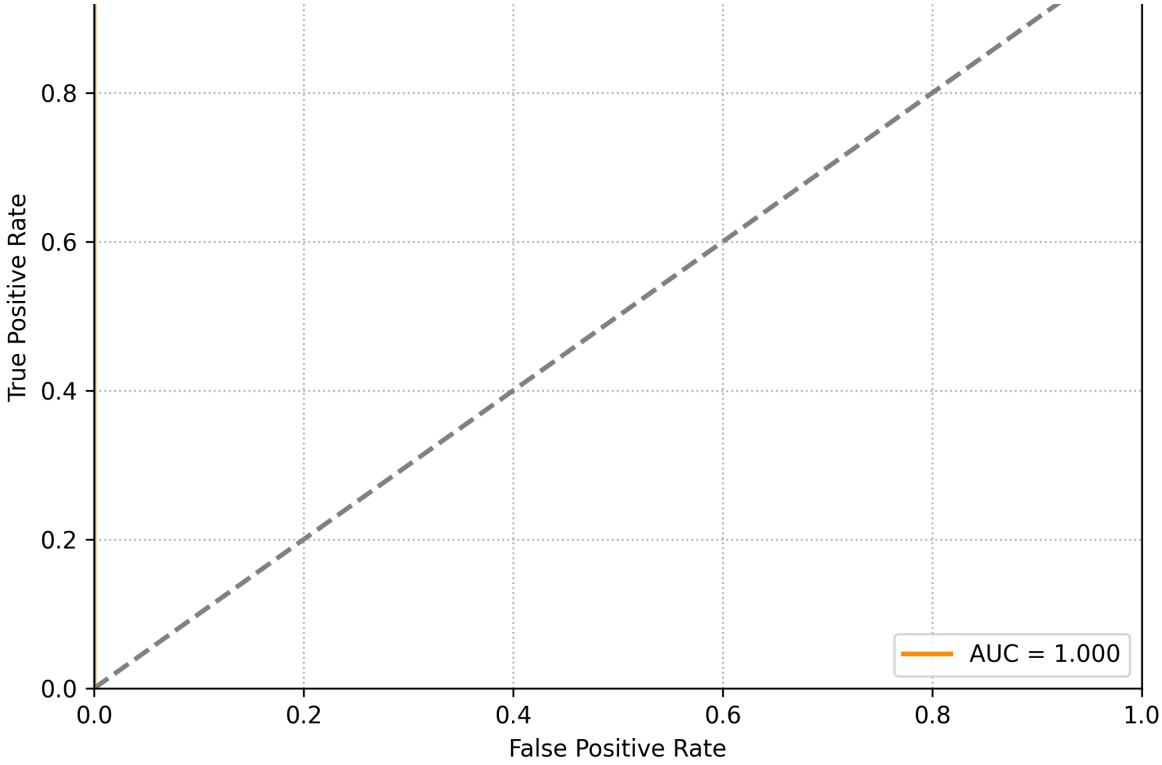
3.2.0.3 Overt → Imagined

This transfer learning scenario evaluated how well overt-trained models generalize to imagined data. The classifier's performance dropped compared to within-modality cases, but still surpassed chance. Topographic results indicated that spatial structures learned from overt signals retained partial relevance in imagined contexts.

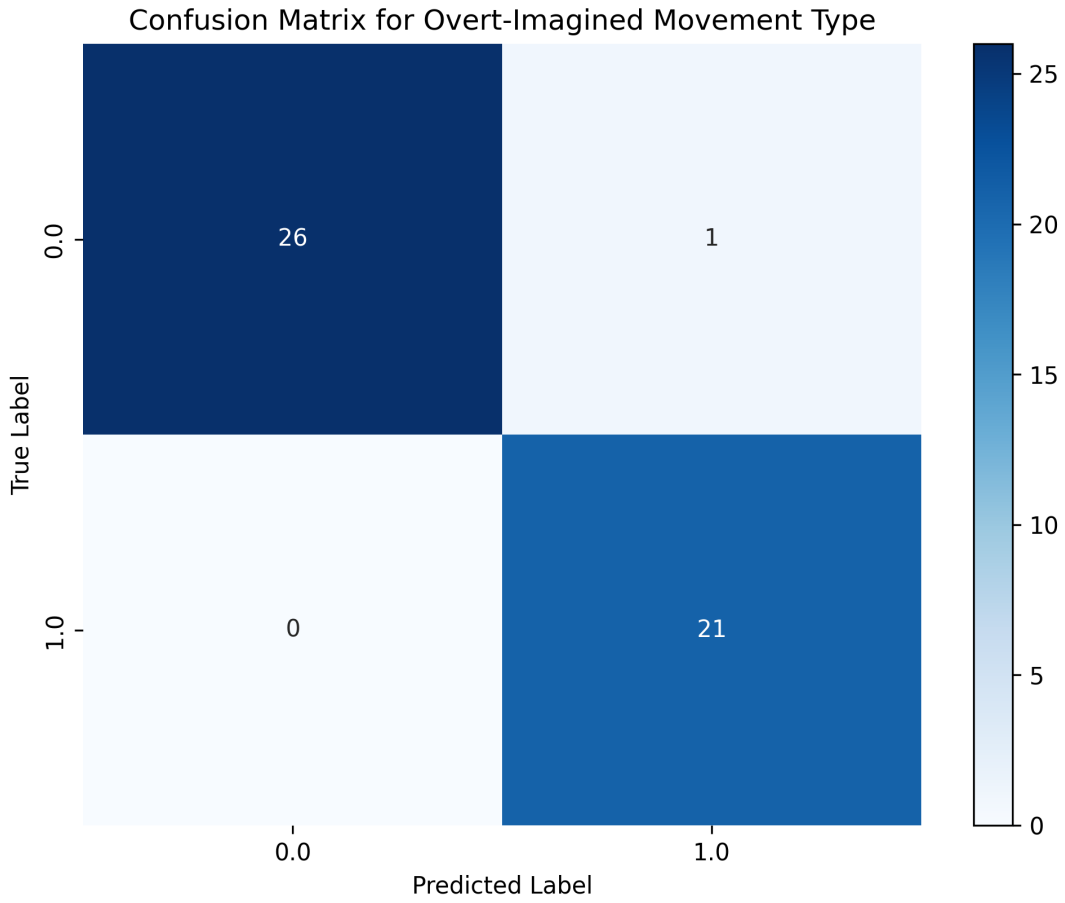


(a)

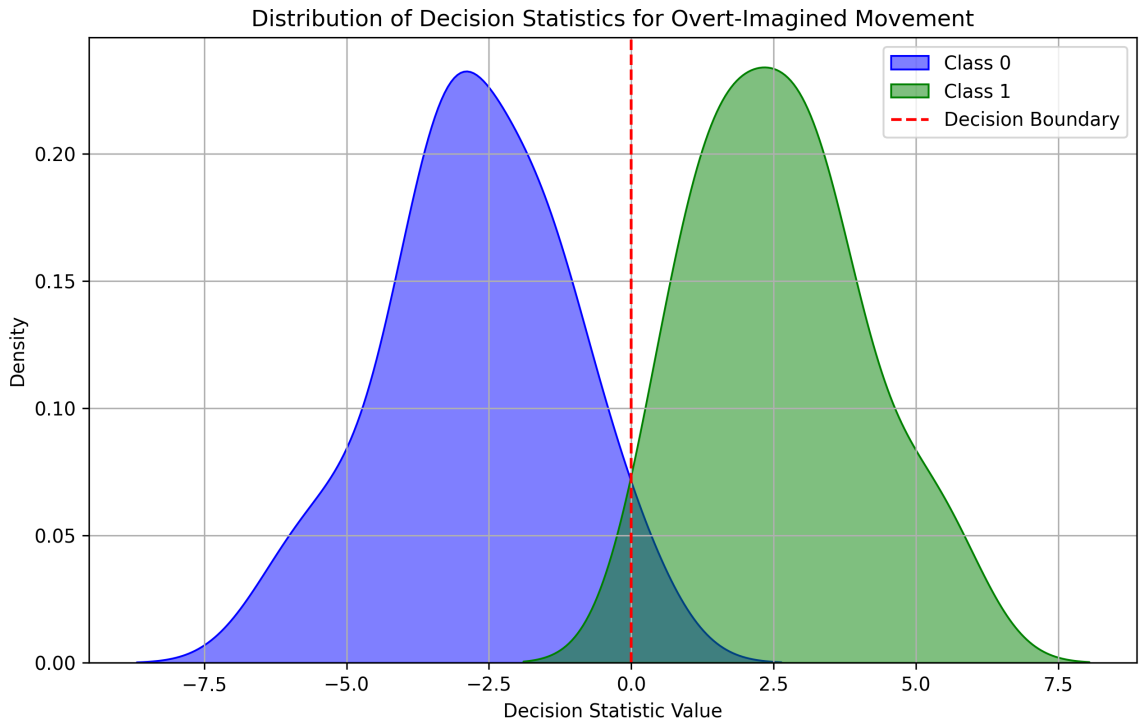



True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
AUC = 1.000

(b)


Confusion Matrix for Overt-Imagined Movement Type
True Label
Predicted Label
0.0
1.0
26
1
0
21
25
20
15
10
5
0

(c)



(d)

Figure 20: Overt to imagined movement classification results: (a) Topographic map showing spatial distribution of informative electrodes, (b) ROC curve illustrating true positive vs false positive rates, (c) Confusion matrix indicating classification performance, and (d) Decision statistic distribution across trials.

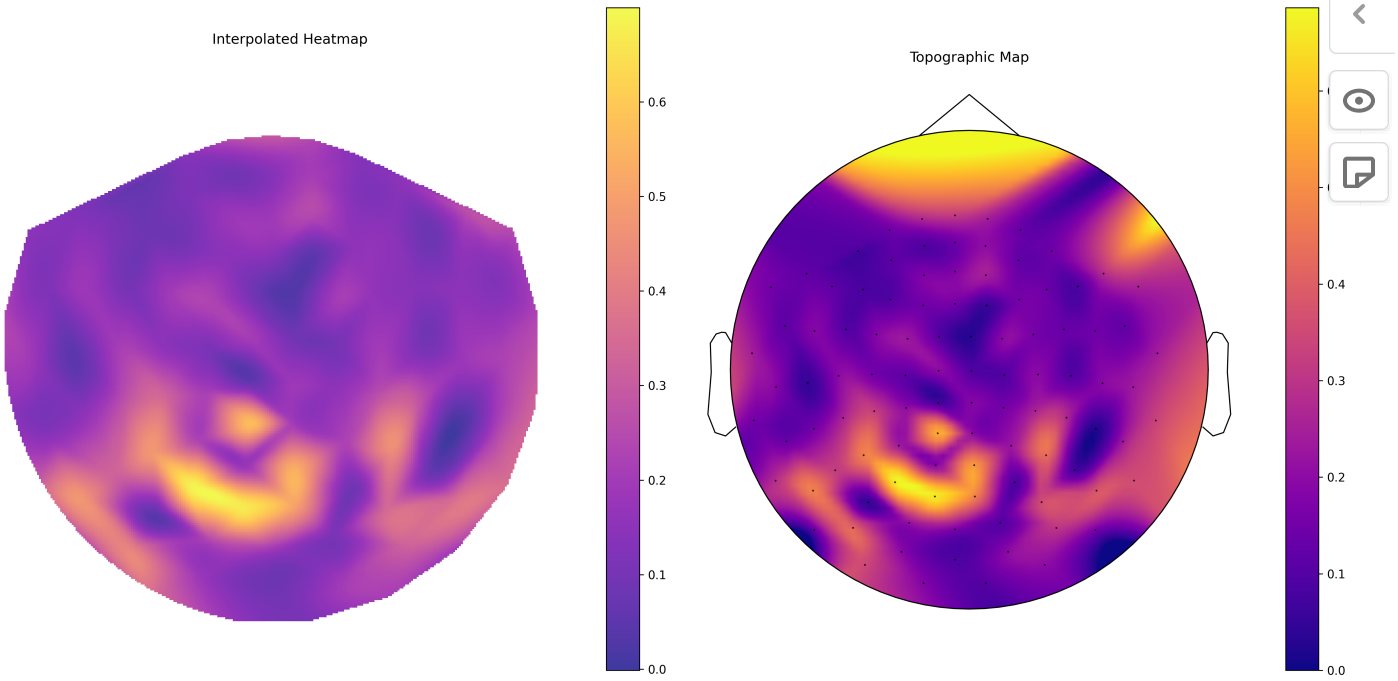
Surprisingly, the classifier achieved a high accuracy when trained with performed real movements and tested with imagined movements. The confusion matrix showed a very small number of misclassifications, indicating that the model was able to learn some relevant features from the overt data that were applicable to the imagined data.

Moreover, the decision statistic is very clear and the distribution is very narrow, indicating that the model was very confident in its predictions. The decision statistic distribution was centered around 0 for the left hand and around 1 for the right hand, indicating that the model was able to effectively learn the underlying patterns in the imagined movement data.

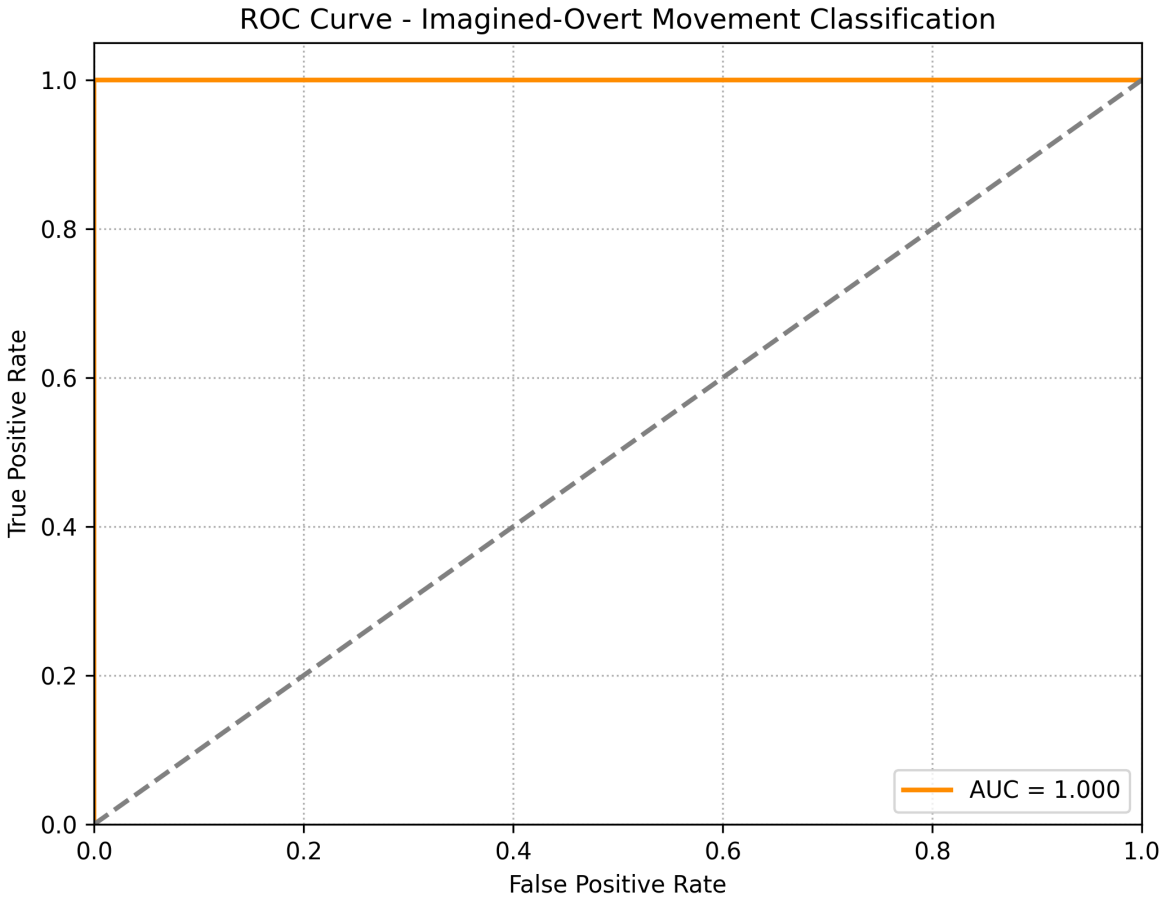
3.2.0.4 Imagined → Overt

This scenario tested the reverse generalization: whether imagined training could support prediction of overt signals. The classifier achieved moderate performance. Interestingly, the decision score distributions were wider, suggesting less confident separation. Nonetheless, spatial maps retained interpretable motor-area activations.

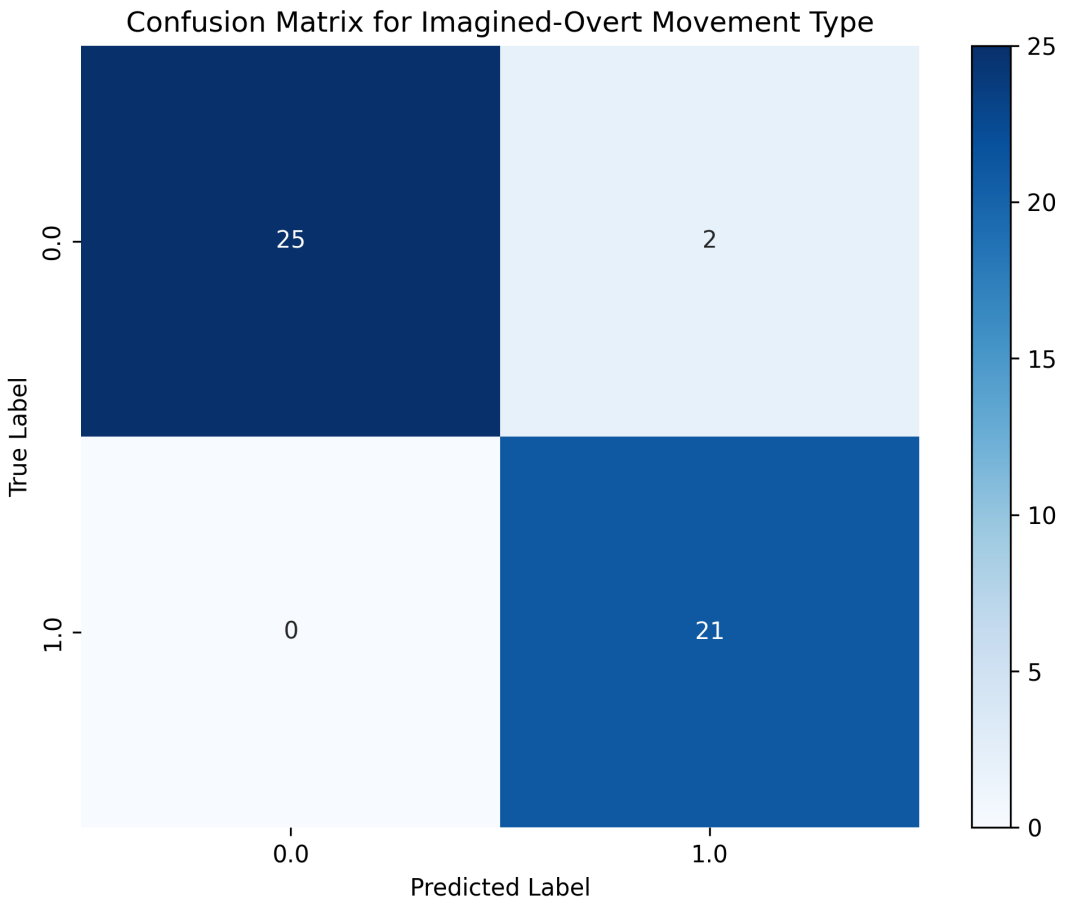
Figure Placeholders: - Topomap: `imagined-overt-topomap.png` - ROC Curve: `imagined-overt-roc-curve.png` - Confusion Matrix: `imagined-overt-confusion-matrix.png` - Decision Statistic Distribution: `imagined-overt-decision-statistic.png`


Interpolated Heatmap
Topographic Map
0.6
0.5
0.4
0.3
0.2
0.1
0.0

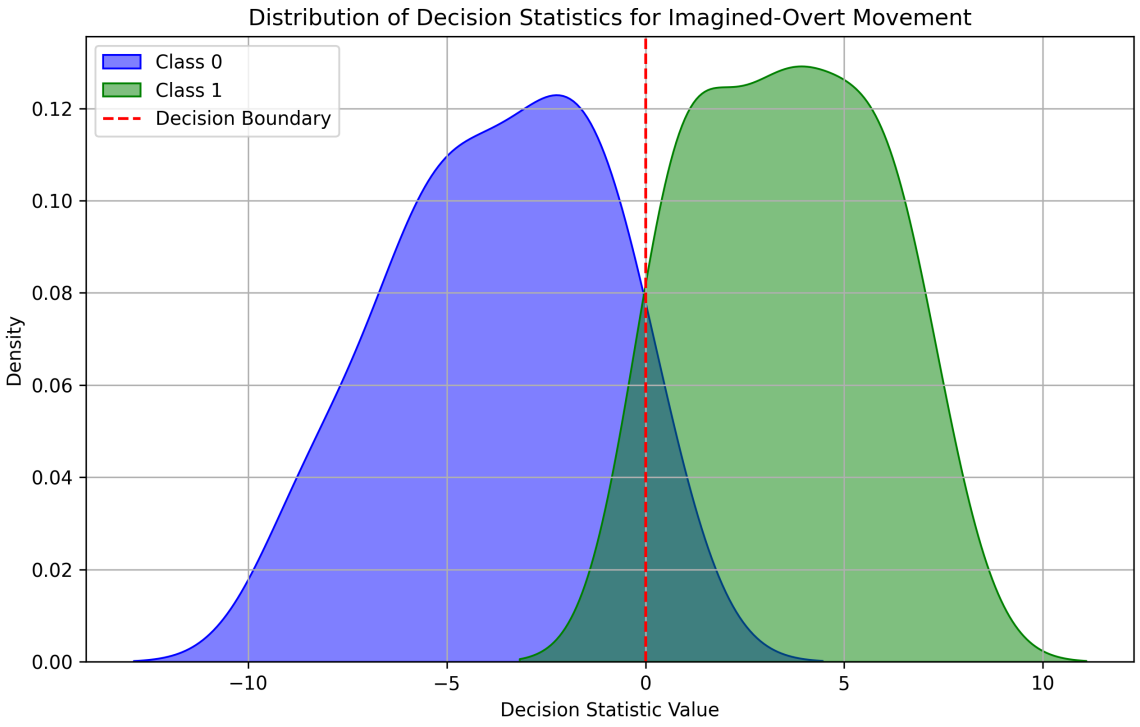
(a)


ROC Curve - Imagined-Overt Movement Classification
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
AUC = 1.000

(b)


Confusion Matrix for Imagined-Overt Movement Type
25
2
0
21
0.0
1.0
True Label
0.0
1.0
Predicted Label
25
20
15
10
5
0

(c)


Distribution of Decision Statistics for Imagined-Overt Movement
Class 0
Class 1
Decision Boundary
Density
0.00
0.02
0.04
0.06
0.08
0.10
0.12
−10
−5
0
5
10
Decision Statistic Value

(d)

Figure 21: Imagined to overt movement classification results: (a) Topographic map showing spatial distributio of informative electrodes, (b) ROC curve illustrating true positive vs false positive rates, (c) Confusion matrix indicating classification performance, and (d) Decision statistic distribution across trials.

3.2.0.5 Scenario Comparison Summary

The figure below provides a side-by-side comparison of **accuracy** and **AUC** scores across the four scenarios using a linear SVM. As expected, performance was highest when training and testing within the same modality (overt-overt or imagined-imagined). Generalization between imagined and overt signals proved more difficult but remained informative.

Scenario	Accuracy	AUC
Overt → Overt	98.9%	1.0
Imagined → Imagined	94.9%	0.94
Overt → Imagined	97.9%	0.98
Imagined → Overt	95.8%	0.95

These findings emphasize the domain-specific nature of EEG classification. While classifiers can generalize across signal types to some extent, model performance is consistently stronger within the same training and testing modality. This has important implications for BCI design, particularly when building models intended for use in both imagined and overt control environments.

3.3 Kernel SVM Performance and Interpretability

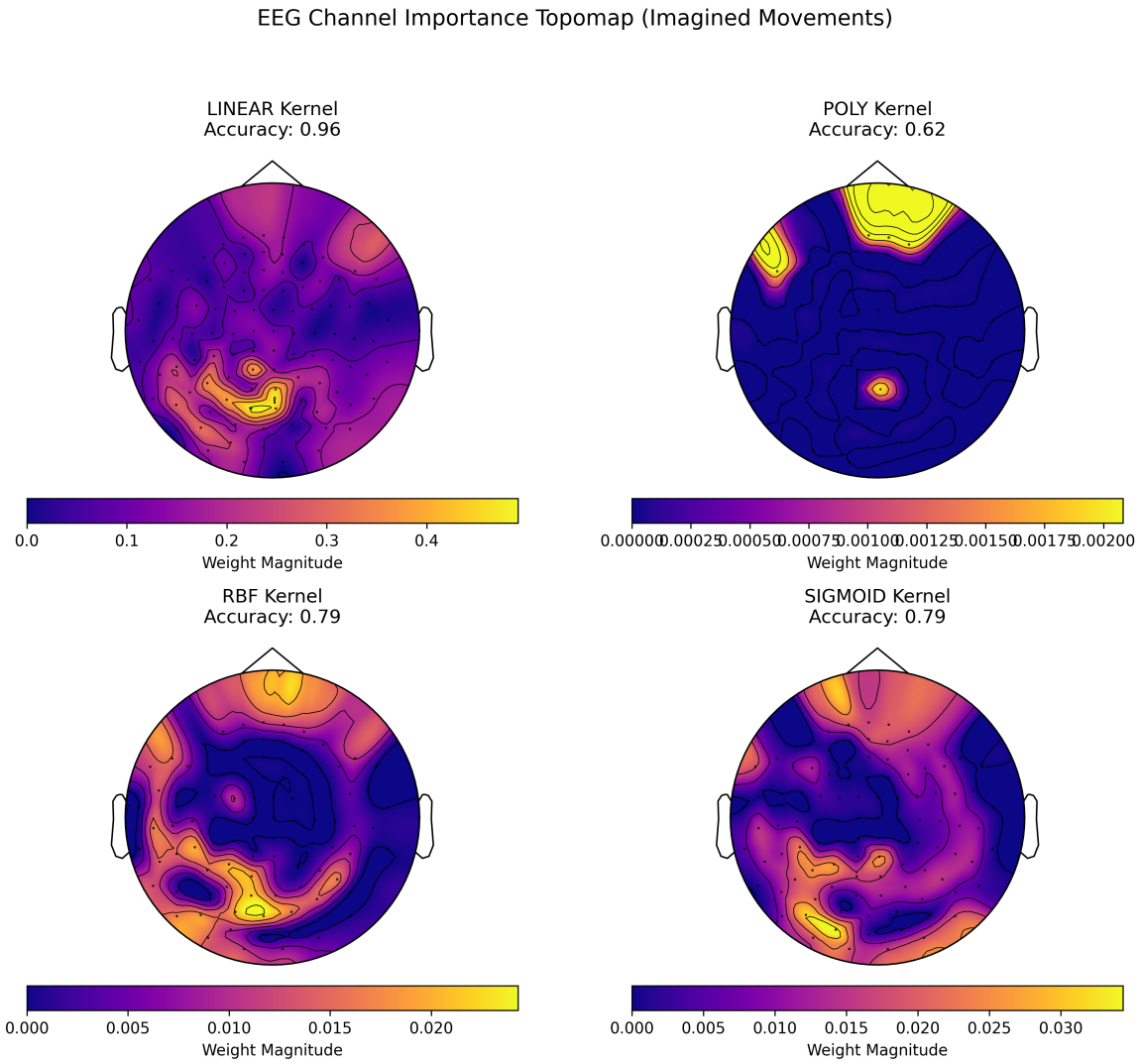
To explore the effect of different nonlinear transformations on EEG classification, we trained SVMs using four kernel types—**linear**, **polynomial**, **RBF**, and **sigmoid**—on both imagined and overt movement datasets. All models used the same regularization search strategy and were evaluated on a held-out test set using accuracy, ROC-AUC, decision statistics, and spatial interpretability through topomaps.

Surprisingly, the **linear kernel consistently outperformed** its more complex counterparts in both conditions. It not only yielded the highest test accuracy and AUC but also produced the most stable and interpretable decision boundaries. The topographic maps generated from the linear models displayed focused and physiologically plausible electrode activations, aligning well with expected motor cortex patterns.

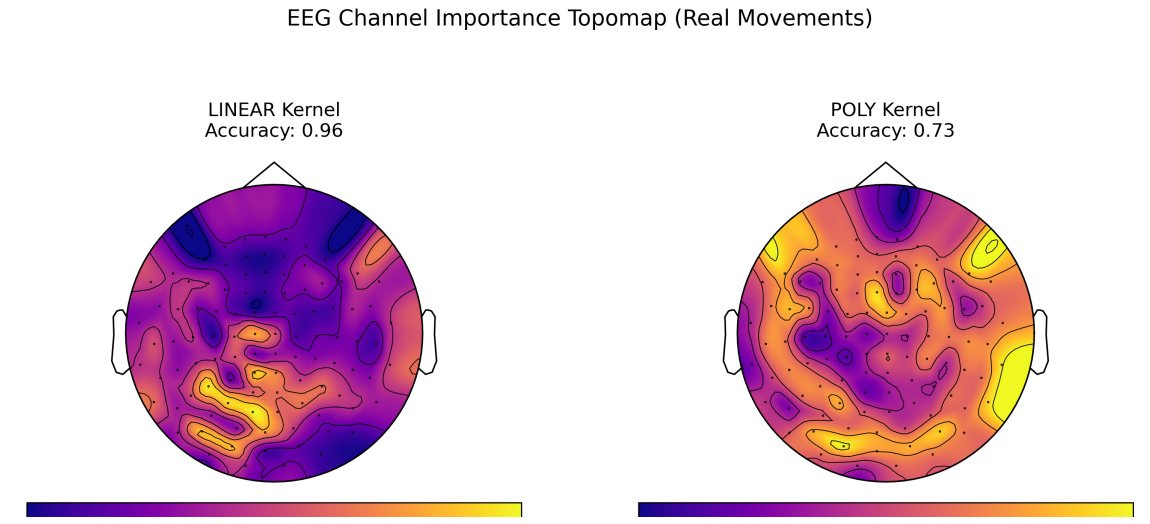
In contrast, while RBF and polynomial kernels offered more flexible decision surfaces, they did not significantly improve classification and sometimes introduced noisy or less consistent spatial weight distributions. The sigmoid kernel performed the weakest overall, showing high variance and poor generalization in both datasets.

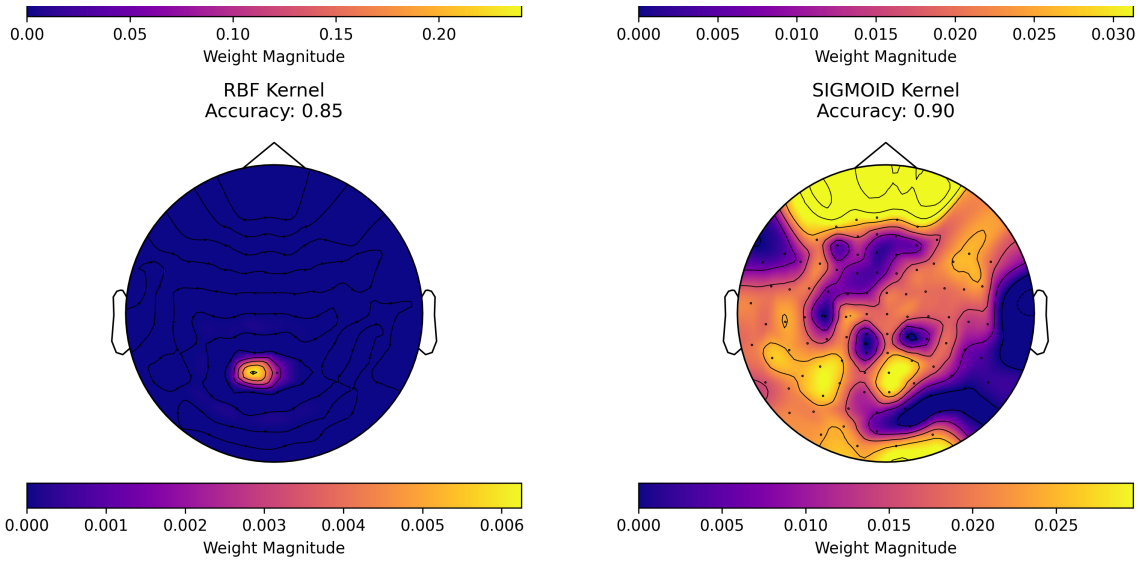
This experiment reinforces an important principle: **more complex models do not always**

This experiment reinforces an important principle: **more complex models do not always outperform simpler ones**, especially when the signal is already well-structured or when model interpretability is a priority. In this case, the linear SVM not only proved highly effective but also enabled detailed spatial insights through clean topographic visualizations.



(a)





(b)

Figure 22: Kernel comparison results: (a) Topographic maps showing spatial distribution of informative electrodes for different kernel SVMs on imagined movements, (b) Topographic maps showing spatial distribution of informative electrodes for different kernel SVMs on overt movements.

Analyzing the pictures above we can a region is always very relevant for the decision of the class but it varies depending on the kernel used. The linear kernel shows a very clear pattern of activation over the central parietal region, while the RBF kernel shows a more diffuse pattern of activation. The polynomial kernel shows a similar pattern to the RBF kernel, but with less intensity. The sigmoid kernel shows a very diffuse pattern of activation, indicating that it is not able to learn the underlying patterns in the data.

Interestingly, the linear kernel produced the most interpretable topographic maps, with clear spatial distributions of informative electrodes.

Kernel	Imagined Accuracy	Real Accuracy	Notes
Linear	0.96	0.96	Best performer, interpretable
RBF	0.79	0.85	Flexible, but no performance gain
Polynomial	0.62	0.73	Moderate performance, overfit-prone
Sigmoid	0.79	0.90	Weak performance, high variance

4 Conclusion

This project explored the use of Support Vector Machines (SVMs) for classifying EEG data related to motor activity, both overt and imagined. The project was conducted in alignment with the ECE 580 mini-project requirements and involved rigorous experimentation across multiple dimensions of model complexity, validation, and modality-specific generalization.

model complexity, validation, and modality-specific generalization.

One of the central contributions of this work was the implementation of a **two-level nested cross-validation** pipeline. This method proved essential for selecting regularization parameters in a statistically robust manner, ensuring that hyperparameter tuning was fully decoupled from final model evaluation. In high-dimensional, low-sample-size contexts like EEG, this methodological rigo is critical to producing generalizable and reproducible results.

Another key takeaway was the **surprising strength of the linear SVM**, especially in comparison to more complex kernel-based models. Despite testing polynomial, RBF, and sigmoid kernels, the linear SVM consistently delivered the best overall performance in both imagined and overt movement classification tasks. It also yielded the clearest spatial patterns in topographic maps, enabling interpretable visualizations of channel relevance. This finding underscores a vital lesson in applied machine learning: **simplicity can outperform complexity when the data is well-structured and the model is appropriately regularized**.

The results showed:

- **High classification accuracy and AUC in within-modality scenarios**, particularly Overt → Overt.
- **Non-trivial classification performance even in Imagined → Imagined**, despite the known challenges with such data.
- **Limited but meaningful generalization across modalities** (e.g., Overt → Imagined), suggesting shared spatial features that models can exploit even under signal mismatch.
- **L2 regularization generally outperforming L1**, except in some transfer cases where sparsity favored L1.

From a practical standpoint, the ability to decode motor intent from EEG using interpretable and computationally efficient models like linear SVMs makes this approach highly attractive for real-time Brain-Computer Interface (BCI) systems. The clean separation achieved with simple models reduces both training complexity and latency, which are vital considerations in interactive neurotechnologies.

This project was made possible through collaborative support:

- The **ECE580 course instruction team**, for providing well-defined datasets and a detailed project scaffold.
- **Class peers and the Colab@Duke community**, who offered valuable discussions on model tuning and visualization strategies.
- The open-source contributions of MNE-Python, scikit-learn, and UMAP libraries, which enabled the advanced spatial and decision-surface visualizations that enhanced the interpretability of this work.

4.0.1 Next Steps and Future Directions

One of the most immediate next steps involves expanding the evaluation to larger and more diverse datasets, including different subjects or session conditions. The current results were obtained on single-session data, and testing generalization across individuals would provide stronger evidence

for real-world applicability. Additionally, although this study focused on static trial-level inputs, future efforts could explore more dynamic classification pipelines that operate continuously on streaming EEG data—moving toward real-time decoding scenarios.

There is also significant potential in deploying the trained models within real-time Brain-Computer Interface (BCI) systems. Given the low computational overhead of linear SVMs, these models are ideally suited for real-time classification engines that could be integrated with digital interfaces or assistive technologies. For example, the classifier could be used to trigger directional input commands based on imagined movements, enabling control of cursors, robotic arms, or external software environments.

Moreover, the simplicity of the linear SVM model makes it a strong candidate for implementation on embedded systems such as Raspberry Pi or microcontroller units, allowing classification to occur on-device without requiring high-power computing resources. This would open doors for the development of portable, low-cost, and scalable EEG-based systems for use in accessibility or rehabilitation contexts.

4.0.2 Final Remarks

While many machine learning projects assume that increasing model complexity guarantees better performance, this work serves as a counterexample rooted in neuroscience. The structure of EEG signals—and the spatially organized nature of motor-related brain activity—lends itself well to **linear classification models**, provided the preprocessing and validation are carefully performed.

This study not only achieved strong technical results but also highlighted the importance of methodical design, model interpretability, and simplicity—principles that will continue to guide future work in brain-computer interface research and beyond.

5 References

5.1 Colaboration

- Peter Banyas - Discussed and provided feedback on the project related to decision statistics and topographic maps.
- Pedro Melo - Provided valuable insights on the use of UMAP for dimensionality reduction and visualization.

5.2 Packages used

- Scikit-learn: Used for implementing and evaluating SVM classifiers, performing grid search, and calculating performance metrics like accuracy and ROC-AUC [7].
- NumPy: Used for numerical operations and handling arrays efficiently [8].
- Pandas: Used for data manipulation and analysis, particularly for loading and preprocessing the

EEG datasets [9].
- Matplotlib: Used for creating visualizations, including topographic maps and ROC curves [10].
- Seaborn: Used for enhancing the aesthetics of visualizations, particularly confusion matrices [11].
- MNE: Used for EEG data processing, including filtering, epoching, and topographic map generation [12].
- UMAP: Used for dimensionality reduction and visualization of high-dimensional EEG data [13].

References

[1] A. R. Mathew, A. Al Hajj, and A. Al Abri, "Human-computer interaction (HCI): An overview," in *2011 IEEE international conference on computer science and automation engineering*, 2011, pp. 99–100. doi: 10.1109/CSAE.2011.5953178.

[2] G. Costantini *et al.*, "SVM Classification of EEG Signals for Brain Computer Interface," in *Neural Nets WIRN09*, IOS Press, 2009, pp. 229–233. doi: 10.3233/978-1-60750-072-8-229.

[3] Y.-T. Wu, T. H. Huang, C. Yi Lin, S. J. Tsai, and P.-S. Wang, "Classification of EEG motor imagery using support vector machine and convolutional neural network," in *2018 international automatic control conference (CACS)*, 2018, pp. 1–4. doi: 10.1109/CACS.2018.8606765.

[4] T. Dai and Y. Dong, "Introduction of SVM related theory and its application research," in *2020 3rd international conference on advanced electronic materials, computers and software engineering (AEMCSE)*, 2020, pp. 230–233. doi: 10.1109/AEMCSE50948.2020.00056.

[5] L. E. Melkumova and S. Ya. Shatskikh, "Comparing ridge and LASSO estimators for data analysis," *Procedia Engineering*, vol. 201, pp. 746–755, 2017, doi: https://doi.org/10.1016/j.proeng.2017.09.615.

[6] M. W. S., "neuromappr," *GitHub*. Apr. 2025. Accessed: Apr. 27, 2025. [Online]. Available: https://github.com/Wanghley/neuromappr

[7] F. Pedregosa *et al.*, *Scikit-learn: Machine learning in Python*, vol. 12. 2011, pp. 2825–2830.

[8] C. R. Harris *et al.*, *Array programming with NumPy*, vol. 585. Nature Publishing Group, 2020, pp. 357–362.

[9] W. McKinney *et al.*, *Pandas: A foundational python library for data analysis and statistics*. 2011.

[10] J. D. Hunter, *Matplotlib: A 2D graphics environment*, vol. 9. IEEE Computer Society, 2007, pp. 90–95.

[11] M. Waskom, *Seaborn: Statistical data visualization*, vol. 6. 2021, p. 3021.

[12] A. Gramfort *et al.*, *MNE software for processing MEG and EEG data*, vol. 86. 2014, pp. 446–460.

[13] L. McInnes, J. Healy, and J. Melville, *UMAP: Uniform manifold approximation and projection for dimension reduction*. 2018. Available: https://arxiv.org/abs/1802.03426