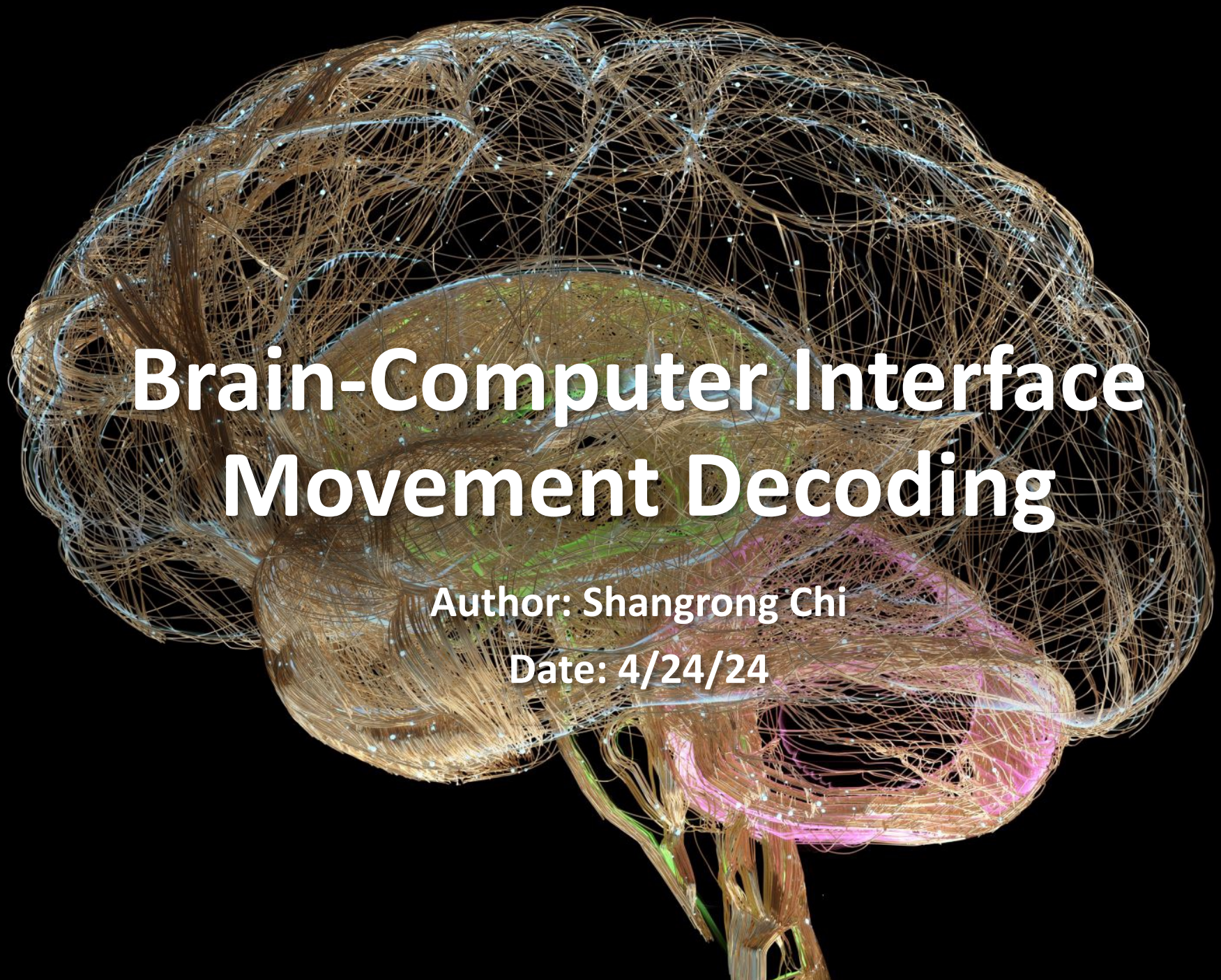


Brain-Computer Interface Movement Decoding

Author: Shangrong Chi

Date: 4/24/24

Introduction to Brain-Computer Interface Movement Decoding

- **Objective**: The primary goal of this project is to develop and evaluate a Support Vector Machine (SVM) classifier that provides a decision statistic for decoding movement intention from EEG data, specifically focusing on differentiating between actual and imagined movements of left or right arms.
- **Importance**: Brain-Computer Interfaces (BCIs) are crucial as they bridge the gap between the human brain and external devices, providing life-enhancing technologies for individuals with disabilities by translating neural activity into actionable commands.
- **Scope**: This project aims to address the challenge of accurately decoding and classifying EEG signals into distinct movement intentions, thereby enhancing the efficacy and responsiveness of BCIs in real-world applications.

Objectives and Significance of Movement Decoding

- **Goals**: This project seeks to:
  - Implement an SVM classifier capable of providing decision statistics.
  - Compare the classifier's efficacy across different SVM kernels and parameters using a robust two-level cross-validation approach.
  - Enhance understanding and application of BCIs in interpreting EEG data for movement intention, which could improve device responsiveness and user experience.
- **Significance**: Decoding movement intention from EEG data is a pivotal problem in neuroscience and rehabilitation engineering. Accurate classification of such data can significantly impact the design of assistive technologies, allowing for better integration and user control.
- **Relevance**: The findings from this project could be applied in various domains, including medical rehabilitation, gaming, and virtual reality, where nuanced understanding and control of movement are valuable. Moreover, insights gained could inform future research in neuroprosthetics and help refine algorithms that interpret neural signals.

Overview of the Classification Challenge

- **Hypothesis Definition**: This project defines a binary classification problem where the two hypotheses are:
  - **H0 (Imagined Movement)**: The hypothesis that the EEG signal corresponds to an imagined movement, without physical execution.
  - **H1 (Overt Movement)**: The hypothesis that the EEG signal corresponds to an actual physical movement of the left or right arm.
- **Data Types**: The EEG data used in this project fall into two categories:
  - **Overt Movement Data**: Signals recorded while the subject performs physical movements.
  - **Imagined Movement Data**: Signals recorded while the subject imagines movement without physical execution. These signals are inherently subtler and more complex to decode due to the lack of physical markers in the EEG data.
- **Challenges of High-Dimensional Data**:
  - **Dimensionality**: Each data sample consists of 120 features, which introduces significant challenges due to the "curse of dimensionality". This phenomenon leads to overfitting where the model might capture noise as well as the signal.
  - **Sample Size**: With only 204 samples per movement type, the small sample size relative to the number of features further complicates the classification task. This scenario is particularly problematic for machine learning models like SVM, which perform best when there is a large amount of data to support the feature space.

High-Dimensional Data and SVM Challenges

- For Support Vector Machines, high-dimensional data presents unique challenges that are not limited to just overfitting:

1. **Sparse Data Representation**: In high-dimensional spaces, data points are often sparse, and the distance between points tends to be similar (a phenomenon known as the "curse of dimensionality"). This sparsity can make it difficult for SVM to effectively define a margin and find a good hyperplane, especially if the data are not linearly separable in the high-dimensional space.
2. **Feature Relevance**: Not all features in high-dimensional data are relevant for making predictions. Some features might be noise, which can disrupt the SVM's ability to establish a robust decision boundary.
3. **Computational Complexity**: Although SVMs are relatively efficient with medium-sized datasets, their training time can become prohibitively long as the number of features grows. This is particularly true when using kernel-based SVMs, where the complexity can scale between quadratic and cubic with respect to the number of samples.

Goals and Importance of BCI Movement Decoding

- **Goals**:
  - Develop an SVM classifier that not only classifies EEG signals into 'overt' and 'imagined' movement categories but also outputs a decision statistic for each classification to quantify the certainty of predictions.
  - Optimize classifier performance using two-level cross-validation to ensure the model is robust and performs consistently across different data splits.
  - Enhance the interpretability of the classifier by providing decision statistics, helping end-users understand the confidence level of predictions which is crucial for real-world applications.
- **Significance**:
  - This project tackles a critical challenge in brain-computer interface technology — improving the accuracy and usability of systems that interpret EEG signals for controlling prosthetic devices and other assistive technology.
  - By providing a decision statistic alongside classifications, the project aims to offer more nuanced feedback that could improve user interaction with BCI devices

SVM Classifier and Decision Statistics

- **SVM Classifier**:
  - **Definition**: It's a set of supervised learning methods in machine learning used for classification, regression and outlier detection.
  - **Rationale for SVM Usage**: SVM is chosen for its effectiveness in handling high-dimensional data like EEG signals, where it can efficiently separate data points into binary classes (imagined vs. overt movements) using hyperplanes. This capability makes it ideal for robust, clear classification needed in BCIs.
- **Mathematical Background**:
  - **Basic Principles**: Support Vector Machines work by finding a hyperplane that best divides a dataset into classes. The formulation can be given by:

$$y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i,\ \xi_i \geq 0 \quad \forall i$$

  - subject to: $\min_{\mathbf{w},b,\xi} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\xi_i$
  - where **w** is the normal vector to the hyperplane, *b* is the bias term, C is the penalty parameter of the error term, and $\phi(\mathbf{x}i)$ potentially maps **x***i* into a higher-dimensional space.

SVM Classifier and Decision Statistics

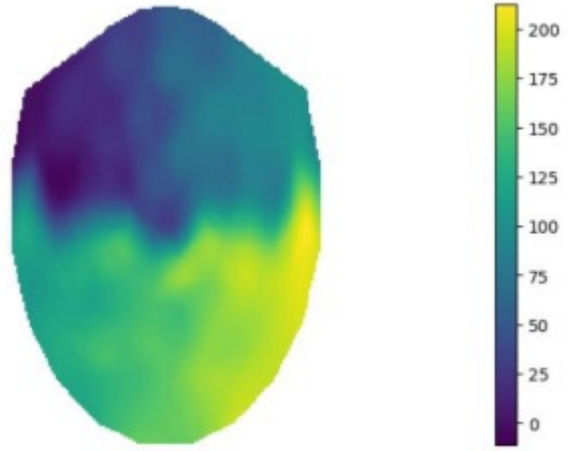
- **Decision Statistic vs. Binary Decision**:
  - **Importance of Decision Statistics**: Unlike a simple class label output, a decision statistic (the distance of a sample from the hyperplane) provides additional quantifiable information on how "confident" the SVM is about its classification. This is crucial for applications requiring more nuanced decision-making:

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

  - The sign of $f(x)$ determines the class, while its magnitude gives a sense of certainty, useful for threshold-based decisions or when prioritizing intervention strategies.

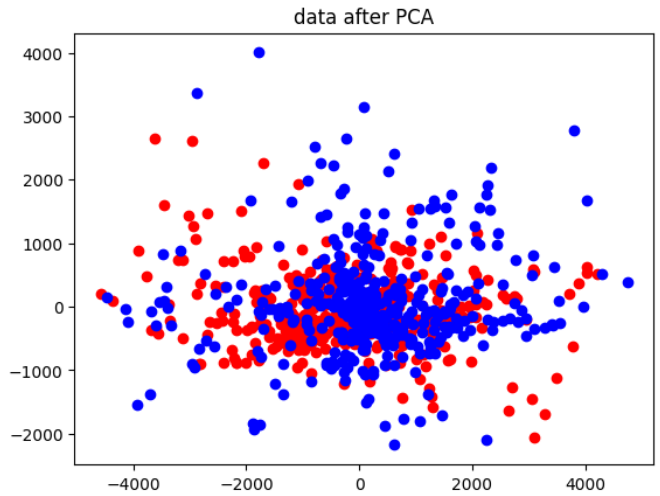
Exploring EEG Data: Visualization and PCA

- **Data Visualization Techniques**:
  - **Heat Maps**: Used to show the concentration of EEG activity across different regions of the brain, helping to identify areas of interest.
  - **3D Brain Maps**: Projects EEG data onto a three-dimensional model of the brain, visually highlighting active regions during overt and imagined movements.



Heat map: the concentration of EEG activity

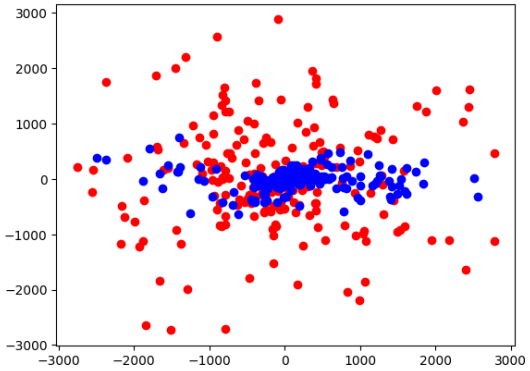
- **Initial Attempts at Dimensionality Reduction**:
  - **Principal Component Analysis (PCA)**: Initially used to reduce data dimensionality by capturing the principal axes of variance. Despite its utility in compressing data, PCA's unsupervised nature meant it did not optimize for class separability, leading to suboptimal classification performance.
  - **Visualization with PCA**: Presented initial findings using PCA-transformed data, offering insights into the general structure and spread of the EEG data without utilizing class labels. However, after PCA dimensionality reduction, we found that the classification of data lacked clear rules



Red for label 1, blue for label 2

Optimizing Feature Space: LDA and Other Techniques

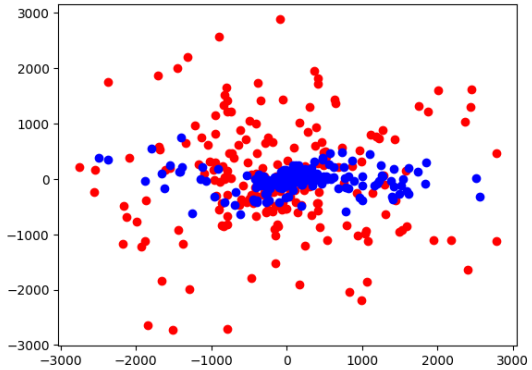
- **Exploring Non-linear Techniques**:
  - **t-Distributed Stochastic Neighbor Embedding (t-SNE)**: This method was explored for its capability to visualize high-dimensional data in low-dimensional spaces. Useful for exploratory analysis, it illustrated how data clusters by movement type, though it was not stable enough for consistent classification tasks.
  - **Autoencoders**: Investigated for their potential to encode EEG data into a compact representation, capturing non-linear relationships better than PCA. However, the lack of supervision in feature relevance to movement classification limited their utility.
- **Optimal Feature Reduction with LDA**:
  - **Linear Discriminant Analysis (LDA)**: Chosen for its supervised approach, LDA effectively reduces dimensions by maximizing class separability, crucial for the SVM classifier used in this project.
  - **Purpose**: Linear Discriminant Analysis (LDA) is a statistical method used to find a linear combination of features that characterizes or separates two or more classes of objects or events. The resulting combination may be used as a linear classifier, or more commonly, for dimensionality reduction before later classification.



Data after CLF and PCA;
Red for label 1, blue for label 2

Optimizing Feature Space: LDA and Other Techniques

- **Mathematical Formulation**:
  - **Between-class and within-class scatter**: The fundamental idea behind LDA is to project the features in higher-dimensional space onto a lower-dimensional space in a way that maximizes the separation between multiple classes.



Data after CLF and PCA;
Red for label 1, blue for label 2

- **Scatter Matrices**:
  - **Within-class scatter matrix** S_W is a summation of scatter matrices for each class:

$$S_W = \sum_{k=1}^{K} \sum_{y_i=k} (\mathbf{x}_i - \mathbf{m}_k)(\mathbf{x}_i - \mathbf{m}_k)^T$$

  - **Between-class scatter matrix** S_B is computed based on the distance between the class means and the overall mean:

$$S_B = \sum_{k=1}^{K} n_k (\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^T$$

- **Maximization Criterion**:
  - The objective is to find the vector $\mathbf{w}$ that maximizes the ratio of the determinant of the between-class scatter to the within-class scatter:

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \frac{\mathbf{w}^T S_B \mathbf{w}}{\mathbf{w}^T S_W \mathbf{w}}$$

  - This is typically solved as a generalized eigenvalue problem:

$$S_B \mathbf{w} = \lambda S_W \mathbf{w}$$

Simulation Parameters and Validation Methods

- **Cross-Validation**:
  - **Two-Level Cross-Validation Explained**:
    - **Purpose**: Two-level cross-validation is used to robustly estimate the model's performance and avoid overfitting, especially important in high-dimensional spaces like EEG data analysis.
    - **First Level (Inner Loop)**: This level of cross-validation is used primarily for model selection, particularly for tuning hyperparameters such as the regularization parameter CC in SVM. The dataset is split into multiple folds, where each fold serves as the validation set once, and the rest as the training set.
    - **Second Level (Outer Loop)**: After selecting the best parameters from the inner loop, the outer loop is used to validate the model's performance. This process is repeated with each fold serving as the test set once, and all remaining data used for training (which again includes an inner cross-validation for parameter tuning).
    - **Implementation**: For example, if 5-fold cross-validation is used in both levels, the total dataset is initially split into 5 outer folds. For each outer fold, the remaining data are split again into 5 new folds for the inner cross-validation process.

Simulation Parameters and Validation Methods

- **Kernels Tested**:
  - **Baseline Kernel**:
    - **Linear Kernel**: As the simplest form, the linear kernel serves as the baseline for comparison. It is useful for understanding the basic separability of the data in its original form.
  - **Additional Kernels Explored**:
    - **Polynomial Kernel**: Allows for the classification of data that is polynomially separable by transforming the input features into polynomial features of higher degree.
    - **Radial Basis Function (RBF) Kernel**: Excellent for handling non-linear data separations; it maps features into a higher-dimensional space where the classes are more likely to be linearly separable.
    - **Sigmoid Kernel**: Mimics the neural activation function, useful for modeling neural interactions such as those in EEG data.
- **Evaluation Metrics**:
  - **ROC Curves (Receiver Operating Characteristic)**: Used to visualize the performance of the classifier at various threshold settings, providing insights into the sensitivity and specificity of the model.
  - **Accuracy Scores**: Quantifies the overall percentage of correctly classified instances, serving as a straightforward metric to assess model performance.

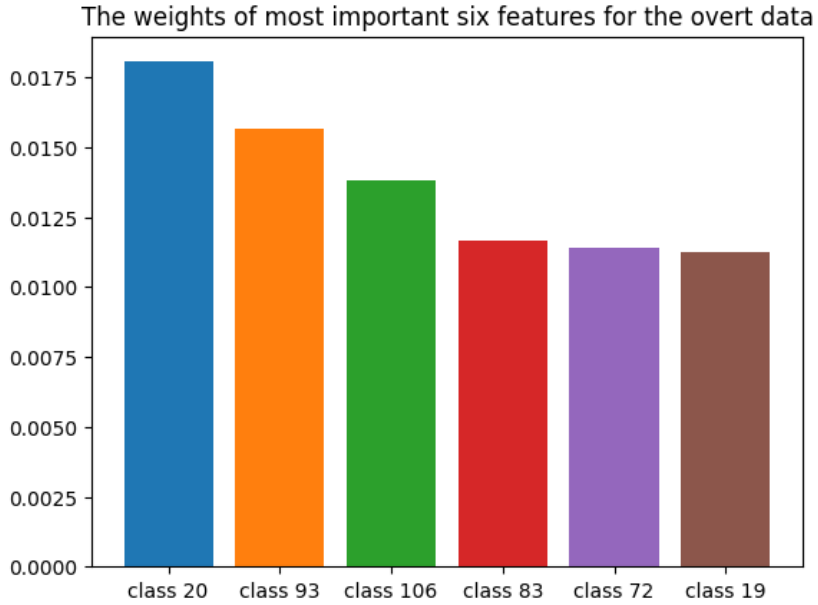
Training and Testing Configurations and Parameter Optimization

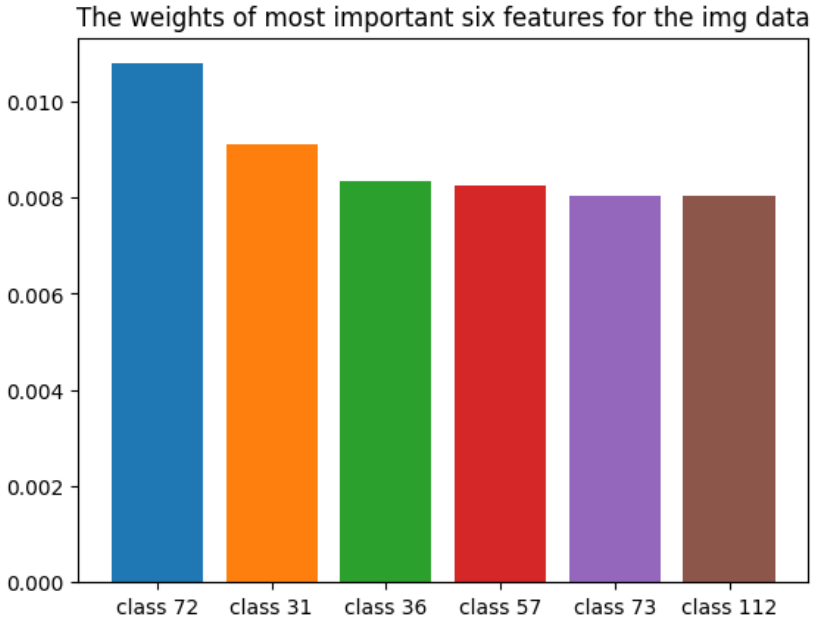
- **Training and Testing Scenarios**:
  - **Direct Use of Raw Data**: Initially, using the raw, high-dimensional data without preprocessing for SVM training proved impractical due to excessive computational times (incomplete outer cross-validation overnight) and poor classification performance, nearly equivalent to random guessing.
  - **Use of PCA for Dimensionality Reduction**: Reducing dimensions to around twenty using PCA decreased training time but was still slow, particularly when testing higher values of regularization parameter $\alpha\alpha$. The accuracy improvement was minimal; with linear kernels achieving about 52% accuracy and non-linear kernels (like RBF and Polynomial) slightly better at around 60%.
  - **Adoption of Non-Linear Dimensionality Reduction**: Techniques such as t-SNE and autoencoders were explored but did not significantly outperform PCA in terms of enhancing SVM classification accuracy.
- **Optimal Preprocessing with LDA**:
  - **Linear Discriminant Analysis (LDA)**: LDA provided substantial improvements, optimizing the feature space effectively by reducing data to one dimension, given the binary nature of the classification (imagined vs overt movements). This constrained the choice to linear kernels; however, this approach yielded an accuracy of over 80% on test sets, significantly outperforming other methods.
  - **Impact of Reduced Dimensionality**: By simplifying the data structure to the most discriminative direction, LDA focused the SVM's learning on the most relevant features, drastically improving both training efficiency and predictive accuracy.

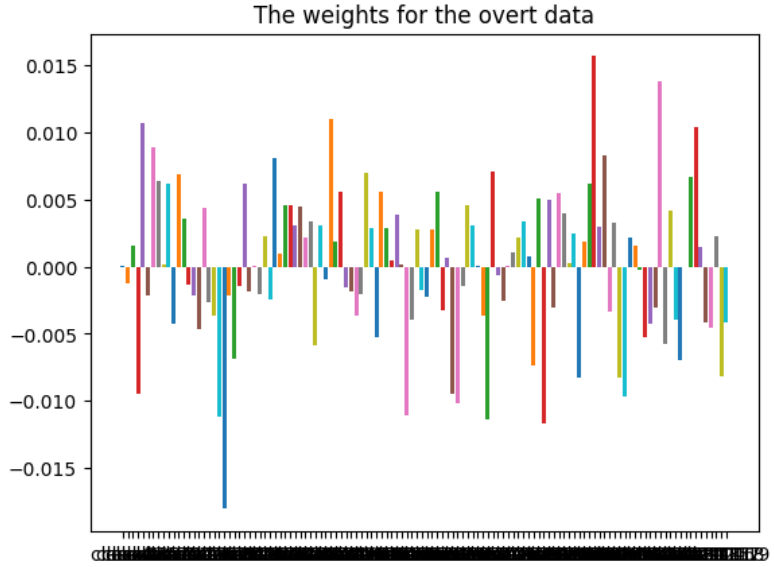
Training and Testing Configurations and Parameter Optimization

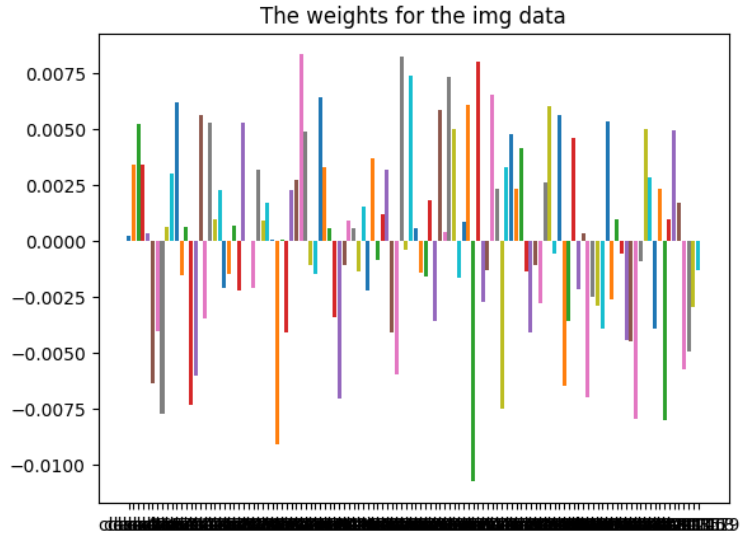
- **Parameter Selection**:
  - **Regularization Parameter (α)**: The choice of α was critical, especially when dealing with different kernels and preprocessing methods. Higher α values often led to longer training times without commensurate gains in model performance. I tested different `for alpha in [10000,100,1,0.01]:`
  - alpha, the best choice is very frequently 10000.
  - **Kernel Choice**: The linear kernel was chosen as the datasets after LDA only have 1 dimension as there are only to classes.
- **Kernel Testing**:
  - **Linear Kernel**: Best performance post-LDA, providing a good balance between training efficiency and accuracy.
  - **Other Kernels**: Explored but discarded due to longer training times and marginal accuracy improvements over the linear kernel when used with PCA or raw data.

SVM Classifier Weights



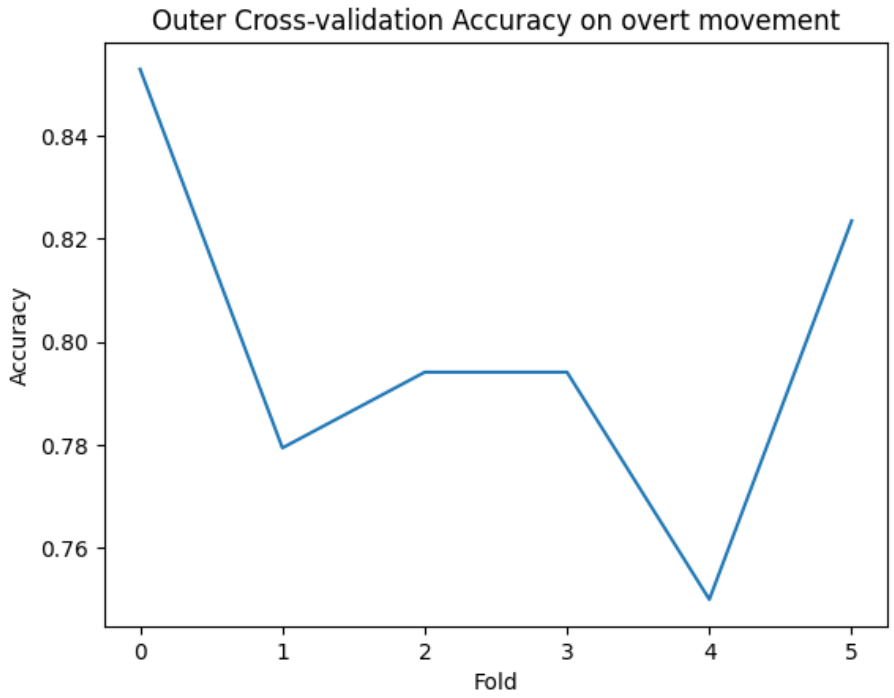


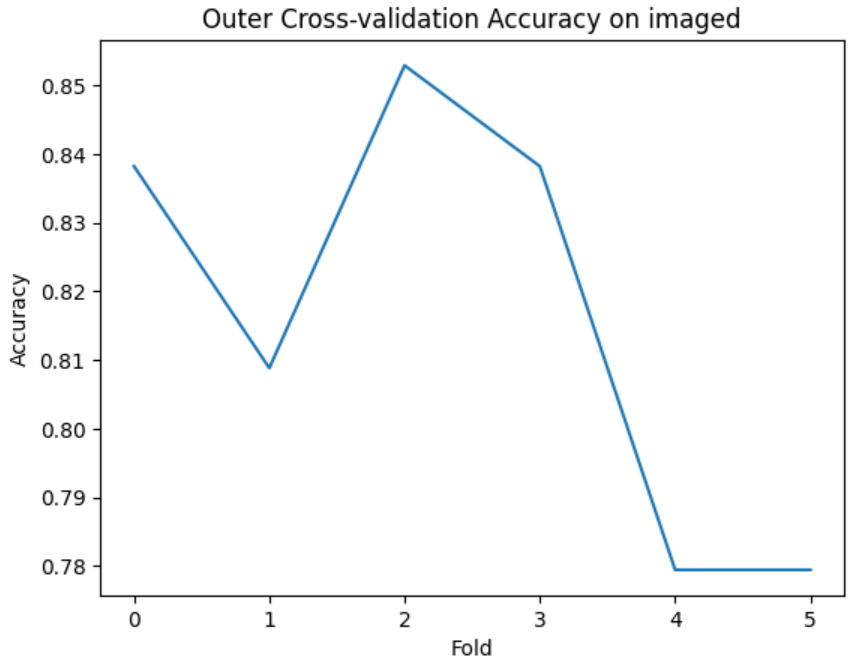




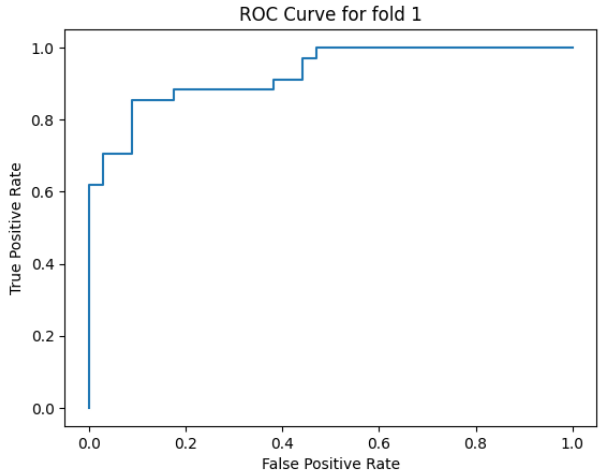
Evaluation of SVM Classifier Performance

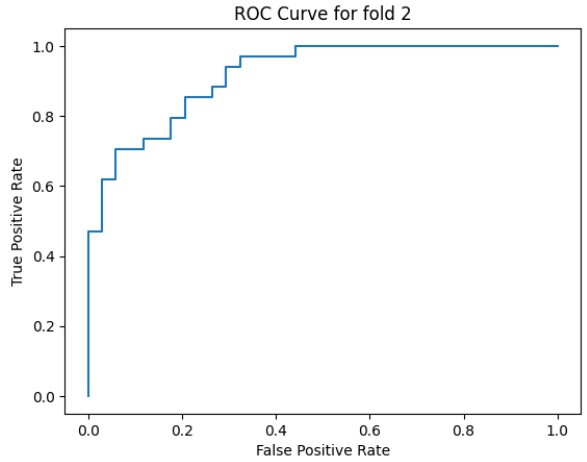
- **Results Overview**:
  - **Accuracy Results**: The SVM classifier, when tuned with an optimal alpha (regularization parameter) of 10000, achieved an average accuracy of 80.62%. This suggests robust performance across the different splits of data.

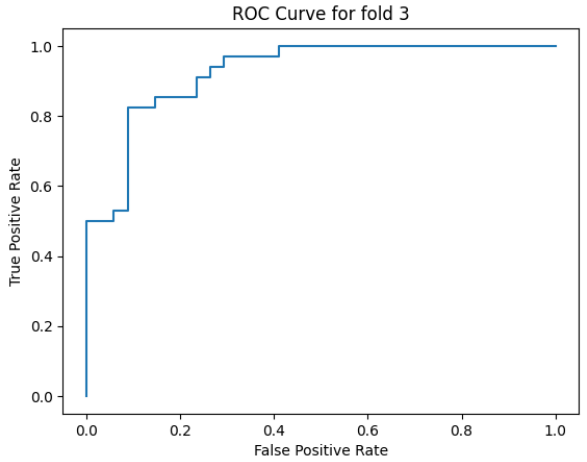


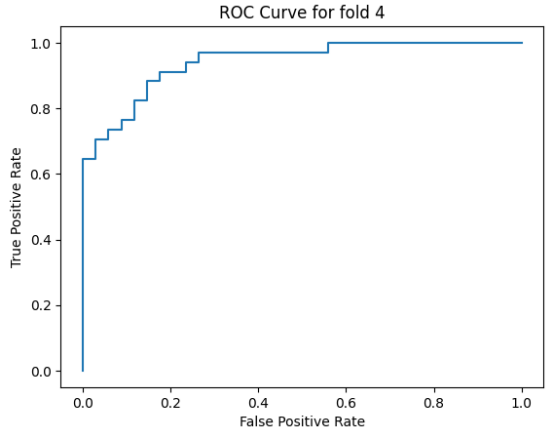


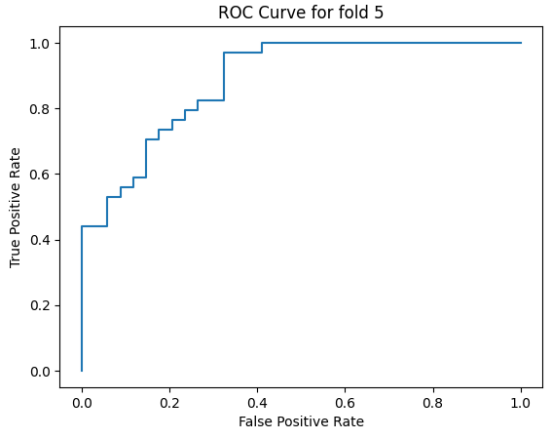
Evaluation of SVM Classifier Performance

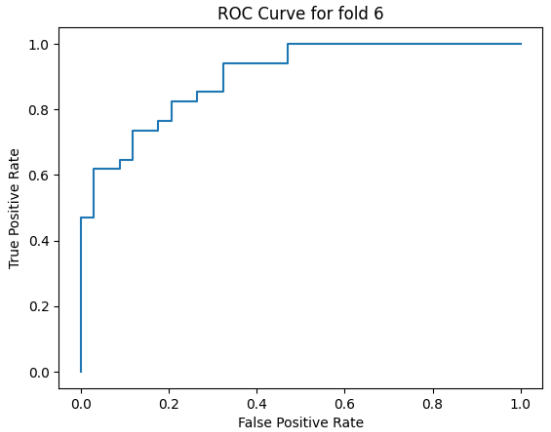
- **Results Overview**:
  - ROC curve for each fold(overt movement for example)
- 







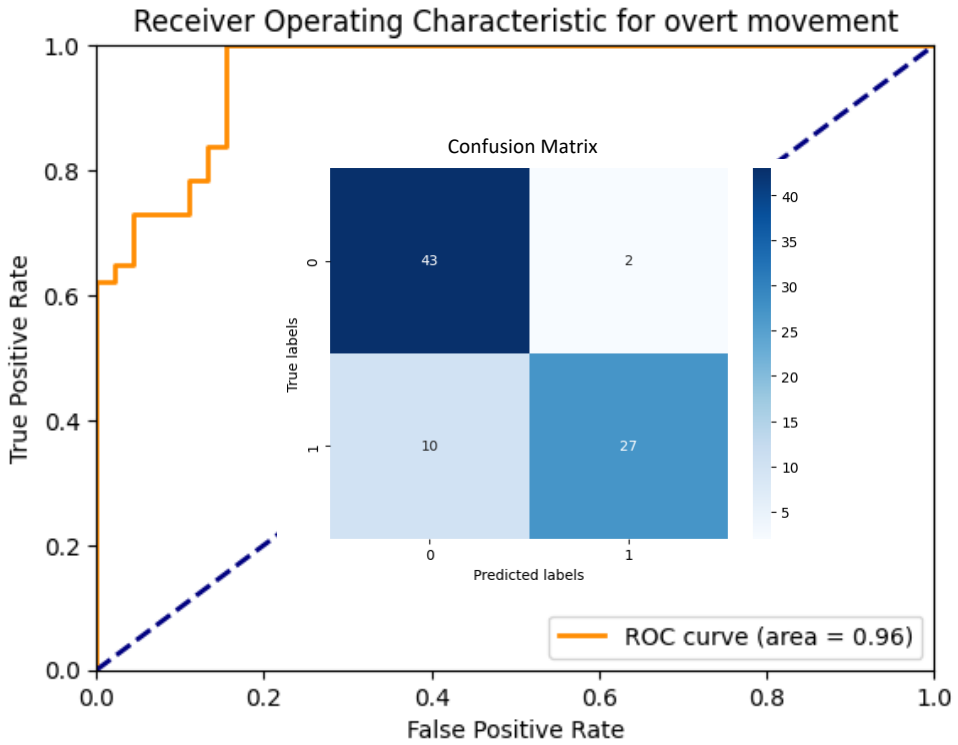


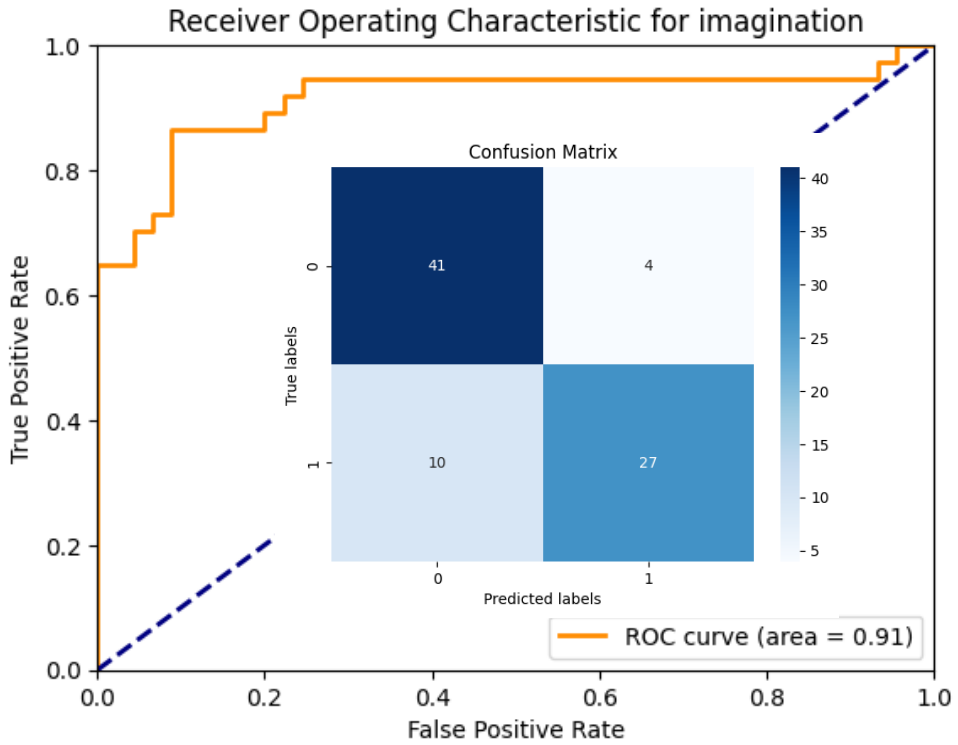


The model has a good and stable performance in each fold, which means that our model has learned some inherent features in the data rather than noise, and the parameter are reasonable.

Evaluation of SVM Classifier Performance

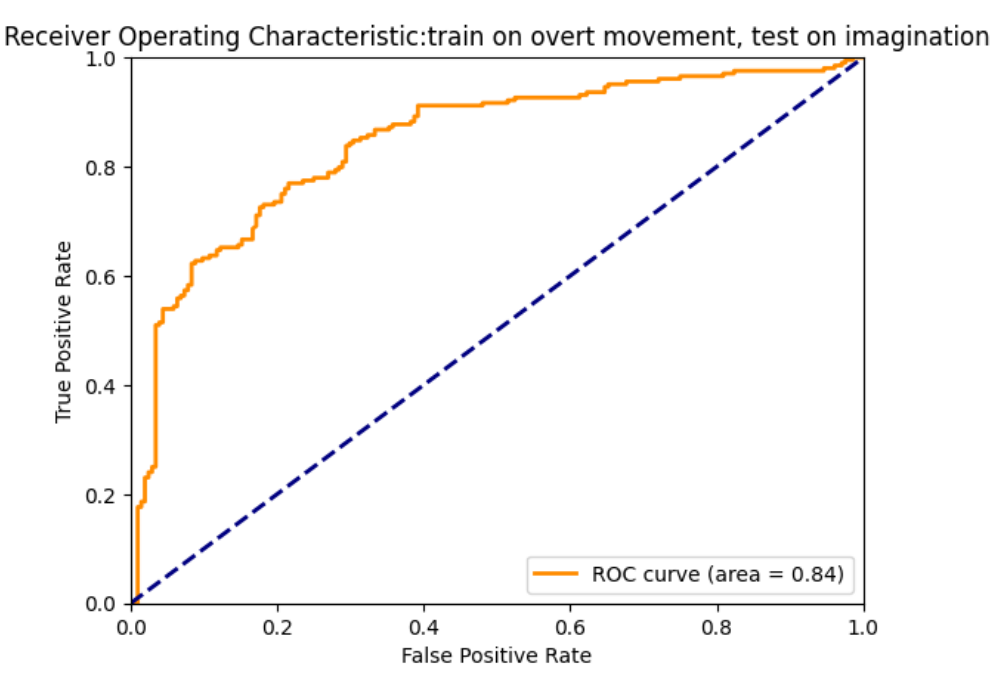
- **ROC Curve Analysis**: The following graphs showed the ROC curves derived from the final model testing. Tarea under the ROC curve (AUC) for the test sets provided a quantitative measure of the classifier's ability to discriminate between the two classes.
  - These two graphs show the ROC curve for 8:2 division of the training set and the test set, and the confusion matrix, in the same mode (i.e. both imaginary). Pretty good curve

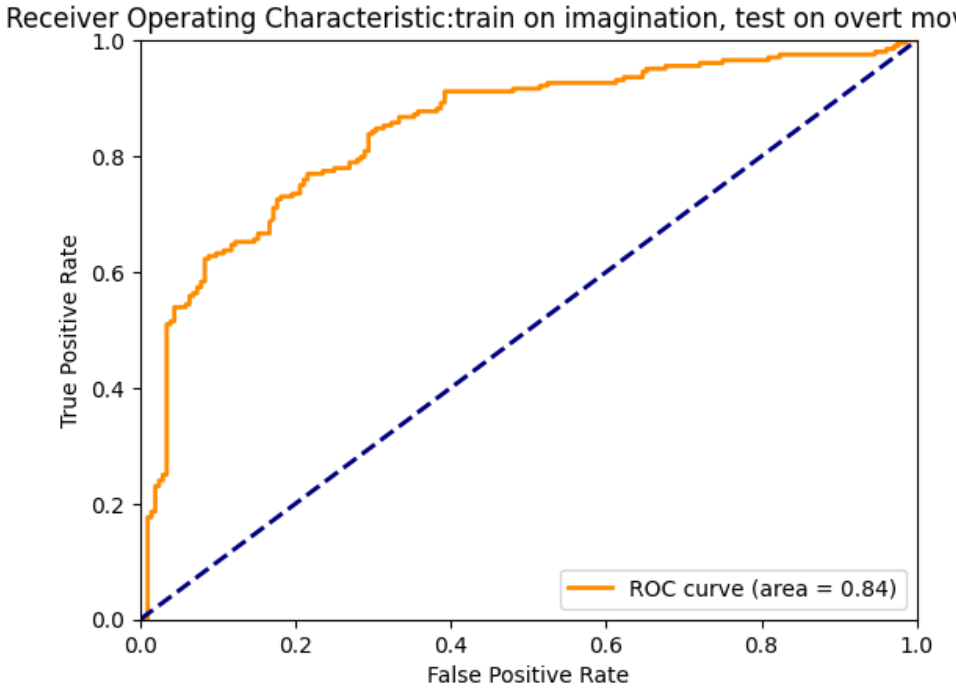




Evaluation of SVM Classifier Performance

The following image shows the ROC curve when the training set and the test set do not belong to the same mode. The area of the curve is obviously reduced, but it's still good





Interpretation of Results and Comparative Insights

- **SVM Performance**:
  - The SVM classifier achieved an average accuracy of 81.62% and high ROC AUC scores across various test sets, demonstrating its strong discriminatory power. These metrics reflect the classifier's robustness in handling complex EEG data characterized by high dimensionality and noise. The high scores indicate not only the effectiveness of the SVM in general but also the success of the regularization and kernel choices made during the model tuning phase. This suggests that the SVM could reliably distinguish between the neural signatures of imagined versus overt movements.
- **Impact of LDA**:
  - The application of Linear Discriminant Analysis (LDA) as a dimensionality reduction technique reduced the feature space to the most informative direction, which significantly enhanced the SVM's accuracy. By simplifying the input feature set to the single most discriminative linear combination of the original features, LDA allowed the SVM to focus on the variance that most effectively separates the two classes. This targeted reduction facilitated a clearer margin between classes, thus improving classification performance and potentially reducing the model's training and prediction time.

Interpretation of Results and Comparative Insights

- **Performance in Cross-Data Training/Testing**:
  - When SVM classifiers were trained on overt movement data and tested on imagined movement data, and vice versa, the performance metrics (accuracy and AUC) showed a noticeable decrease compared to the within-data scenarios. This suggests inherent differences in the neural signatures captured during actual and imagined movements. These discrepancies likely stem from the more pronounced and consistent neural activity associated with actual movements compared to the more nuanced and variable signals of imagined actions.
- **Neurological Signal Variability**:
  - The drop in performance in cross-data scenarios underscores the unique characteristics of neural patterns associated with each type of movement. Actual movements generate robust and easily detectable motor neuron activations that are directly tied to physical actions, whereas imagined movements might not consistently evoke strong neural responses, leading to less distinct data patterns.
  - This variability in signal quality and consistency between actual and imagined movements poses additional challenges for the classifier. It highlights the importance of tailored feature extraction and model training approaches for different types of EEG data.
- **Implications for BCI Applications**:
  - These findings are critical for the design of brain-computer interfaces (BCIs) that need to operate reliably under various conditions. Understanding the distinction in performance between imagined and overt movement data can guide the development of more adaptive algorithms, which could switch strategies depending on the detected movement type.
  - Additionally, these results emphasize the need for BCIs to be trained on diverse datasets that include both overt and imagined movements to ensure the system is well-calibrated and robust against the inherent variability in EEG signals.

Project Summary

- **High Classification Accuracy**: The project achieved a high degree of accuracy, with the best models exceeding 80% in distinguishing between imagined and overt movements. This success underscores the capability of SVM classifiers, particularly when combined with effective preprocessing like LDA, to handle complex, high-dimensional EEG data.
- **Superiority of LDA**: Linear Discriminant Analysis emerged as the most effective method for dimensionality reduction in this context, significantly outperforming PCA and other non-linear methods tested. LDA's ability to reduce features to the most discriminative direction enhanced the SVM's performance by focusing on the most informative aspects of the EEG signals.
- **Challenges with Cross-Data Training**: Training the classifier on one type of movement data (overt or imagined) and testing it on the other type highlighted the inherent differences in neural patterns associated with each movement type, leading to reduced performance in cross-data scenarios.
- **Importance of Tailored Features**: The findings emphasized the importance of selecting and engineering features that are specifically tailored to the nuances of the neural signals associated with different types of movements.

Future Directions

- **Algorithmic Adaptations for Variability**: Future research could explore the development of adaptive algorithms that can dynamically adjust their strategies based on the type of movement being detected (overt vs. imagined). This could involve real-time analysis and classification adjustments to accommodate the variability in signal strength and pattern.
- **Enhanced Data Collection**: Expanding the dataset to include a more diverse array of subjects, movement types, and conditions could help in further refining the classifiers. Increased data diversity would provide a more robust foundation for training and testing, likely improving the generalizability of the models.
- **Deep Learning Approaches**: Investigating deep learning models, particularly Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), might offer improved performance over SVM for EEG data classification by capturing more complex and subtle patterns within the data.
- **Integration with Real-Time BCI Systems**: Implementing and testing the optimized classifiers within real-time BCI systems would be a critical step towards practical application. This could include pilot studies or controlled trials to assess the usability and effectiveness of the BCIs in everyday and clinical settings.
- **Cross-Disciplinary Studies**: Collaborating with neuroscientists and cognitive psychologists to better understand the neural and psychological mechanisms underlying imagined and overt movements could provide deeper insights that would inform further refinement of classification algorithms.

Python Library References

Pandas

- **Functionality**: Data manipulation and analysis tool.
- **Usage**: Loading and processing data in DataFrame format.
- **Website**: Pandas

NumPy

- **Functionality**: Fundamental package for scientific computing.
- **Usage**: Employed for numerical operations on data arrays.

Scikit-learn (sklearn)

•**Functionality**: Tool for data mining and data analysis.
•**Usage**: Implements machine learning models and preprocessing methods.
•**Modules Used**: train_test_split, svm, LinearDiscriminantAnalysis, cross_val_score.

Matplotlib

•**Functionality**: Library for creating visualizations.
•**Usage**: Generating plots such as ROC curves and scatter plots.

Warnings

•**Functionality**: Module used to manage warnings.
•**Usage**: Suppressing warnings to enhance output clarity.

Other References & Collaborations

Other resources:

- **Google search, scikit-learn website, YouTube machine learning videos, ChatGPT.**

- **Website:**
- https://scikit-learn.org/stable/modules/svm.html
- https://www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/
- https://en.wikipedia.org/wiki/Support_vector_machine
- https://www.geeksforgeeks.org/support-vector-machine-algorithm/
- https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47
- https://www.mathworks.com/help/stats/support-vector-machines-for-binary-classification.html
- https://www.datacamp.com/tutorial/svm-classification-scikit-learn-python
- https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/
- https://en.wikipedia.org/wiki/Linear_discriminant_analysis#:~:text=Linear%20discriminant%20analysis%20(LDA)%2C,classes%20of%20objects%20or%20events.
- https://www.ibm.com/topics/linear-discriminant-analysis
- https://www.geeksforgeeks.org/ml-linear-discriminant-analysis/

I worked **independently** because

- My personality is introverted.
- More flexibly to design the project.

The sources I used showed in the left and below.

- https://neptune.ai/blog/cross-validation-in-machine-learning-how-to-do-it-right
- https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
- https://scikit-learn.org/stable/modules/cross_validation.html
- https://www.geeksforgeeks.org/cross-validation-machine-learning/
- https://www.geeksforgeeks.org/cross-validation-machine-learning/
- https://machinelearningmastery.com/k-fold-cross-validation/
- https://www.geeksforgeeks.org/auc-roc-curve/
- https://en.wikipedia.org/wiki/Kernel_method#:~:text=In%20machine%20learning%2C%20kernel%20machines,classifiers%20to%20solve%20nonlinear%20problems.
- https://dida.do/blog/what-is-kernel-in-machine-learning
- https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc?hl=zh-cn#:~:text=An%20ROC%20curve%20(receiver%20operating,False%20Positive%20Rate – [in Chinese but useful]
- https://www.cs.cmu.edu/~tom/10701_sp11/slides/Kernels_SVM_04_7_2011-ann.pdf
- https://www.cs.cmu.edu/~schneide/tut5/node42.html


Thanks for Reading!