Mini-Project #2 Slidedoc

Brain-Computer Interface Movement Decoding

Zhengqi Wang
MEng in ECE @ Duke University

ECE 580
May 9, 2024

Presentation Overview

Project Goals Description

Project Goals

The primary goal of this project is to apply Support Vector Machines (SVMs) for the classification of electroencephalogram (EEG) data in a Brain-Computer Interface (BCI) context. We aim to distinguish between two types of movements (imagined and overt) based on EEG signals, leveraging the power of SVMs to handle high-dimensional data effectively.

$$\textit{EGG Signal} \Rightarrow \textit{Classifier} \Rightarrow \textit{Direction Prediction}$$

Significance of the Problem

Decoding movement intention from EEG data is a critical step in developing effective BCI systems. The ability to accurately classify imagined versus overt movements can lead to significant advancements in assistive technologies, enabling users to interact with their environment in intuitive and meaningful ways.

Applications and Data Relevance

Beyond BCIs, the techniques applied here can extend to various domains, such as medical diagnosis and neurorehabilitation. The project data, encompassing both imagined and overt movement EEG signals, provides a comprehensive basis for exploring the nuances of movement-related brain activity and its accurate classification.

Understanding the Project Data

Data Overview

Our project utilizes electroencephalograph (EEG) data, comprised of signals captured from an able-bodied subject performing two distinct tasks: actual arm movement, known as ”Overt” movement data, and the mere imagination of such movement, termed ”Imagined” movement data. This dualistic dataset poses a unique binary classification challenge where we discern between motor execution and motor imagery.

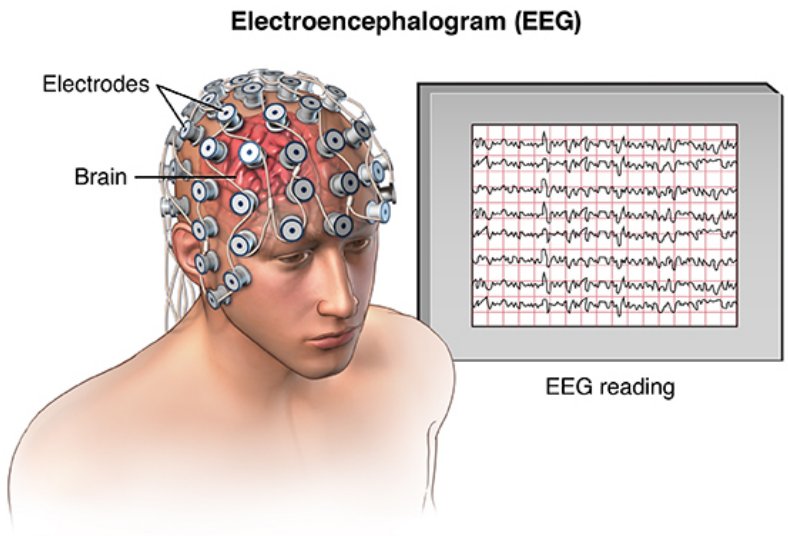


Figure: Electroencephalogram (EEG) signal acquisition using electrodes placed on the scalp.

Classification Context

In this binary hypothesis scenario, our task is to classify each EEG signal into one of two categories H_0 or H_1 representing left or right arm movement respectively. The exact correlation between these categories and actual arm movements remains to be inferred from our analysis.

Mathematical Formulation of SVMs

What is an SVM?

A Support Vector Machine (SVM) is a supervised machine learning model used for classification tasks. It aims to find the optimal hyperplane that separates different classes in the feature space. The optimal hyperplane is the one that maximizes the margin, which is the distance between the hyperplane and the nearest data point from either class.

The Optimal Hyperplane

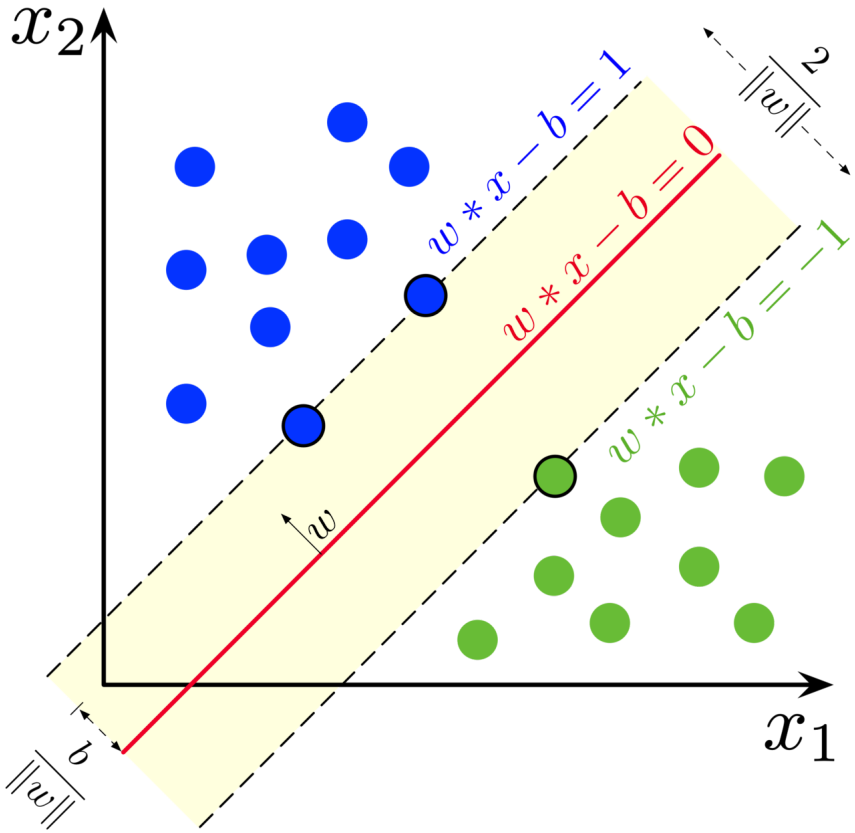
Mathematically, if we have a dataset with features $\mathbf{x}$ and labels $y \in \{-1, 1\}$, the hyperplane can be described by the equation $\mathbf{w} \cdot \mathbf{x} + b = 0$, where $\mathbf{w}$ is the weight vector and b is the bias. The goal of SVM is to find $\mathbf{w}$ and b such that the margin is maximized.

Classification Decision

Once the model is trained, a new data point $\mathbf{x}$ is classified based on the sign of $\mathbf{w} \cdot \mathbf{x} + b$. If the result is positive, the data point is predicted to belong to one class; if negative, to the other.

Margin Maximization

The margin is defined as the distance between the hyperplane and the closest points from either class, known as support vectors. Maximizing the margin is equivalent to minimizing $\|\mathbf{w}\|$, subject to $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1$ for each data point $(\mathbf{x}_i, y_i)$ in the training set.



Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. Image adapted from [1].

Understanding Our SVM Classifier

Our SVM Implementation

Our SVM classifier is implemented as a convex optimization problem aiming to minimize the objective function:

$$\min_{w,c,\xi} \sum \xi_i + \alpha \times W^T W$$

subject to the constraints:

$$y_i \times (W^T X_i + C) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

for $i = 1, 2, \ldots, N$, where W is the weight vector, C is the bias, and ξ_i are the slack variables allowing soft-margin classification.

Regularization and Kernel Choice

Regularization, controlled by parameter α, helps prevent overfitting by penalizing large weights. We use a linear kernel as our baseline and then extend our analysis by experimenting with other kernels to observe their effects on classification performance.

Applying SVM to Classify EEG Data

Challenges of High-Dimensional Data

Our EEG dataset presents a high-dimensional space with 204 features for each of the 240 data points, which poses a potential risk of overfitting due to the ”curse of dimensionality”. This phenomenon can lead to deceptive separability in high-dimensional spaces, where even random noise can appear to have structure, compromising the generalizability of a model.

Why SVM for High-Dimensional Data

Despite the risk of overfitting associated with high-dimensional data, SVMs are particularly well-suited for such scenarios. Their efficiency in memory use, ability to handle non-linear boundaries through kernel trick, and intrinsic capacity to manage high-dimensional data without necessarily incurring overfitting [4]. All these benefits make SVMs an ideal choice for our EEG classification task.

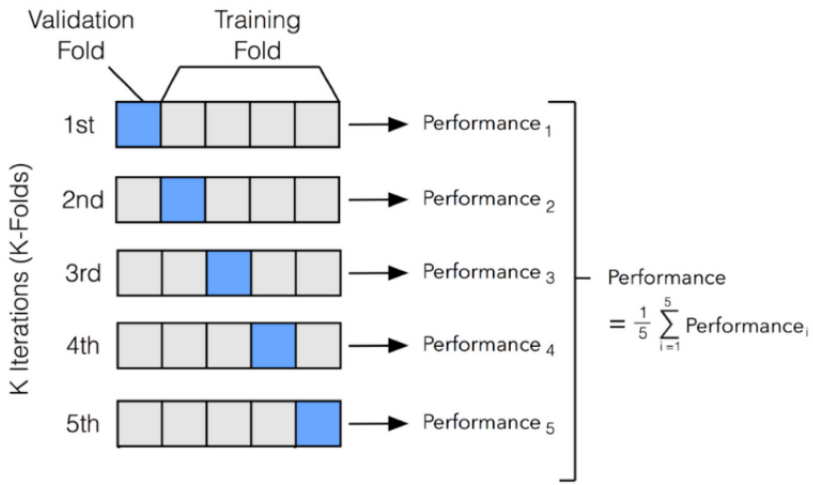
Our Approach with SVM

In our application, we leverage the SVM's robustness in high-dimensional spaces to mitigate overfitting. The SVM classifier constructs a hyperplane in this 204-dimensional space that best separates the classes. The support vectors—data points nearest to the hyperplane—help determine the position and orientation of the hyperplane, ensuring maximum margin and therefore, enhancing the classifier's generalizability to new data.

Evaluating SVM Performance with Cross-Validation

K-Fold Cross-Validation

For our EEG dataset, we apply K-fold cross-validation, a powerful variant where the data is partitioned into k subsets or folds. Each fold serves as a test set while the remaining $k - 1$ folds form the training set. This process is repeated k times, with each fold used exactly once as the test set, allowing every data point to contribute to both training and validation. The k results are then averaged to produce a single estimation, providing insight into the model's expected performance on unseen data.



Cross-validation logic description, cited from [5]

Optimizing Regularization Parameter λ

Regularization and Hyperparameter λ

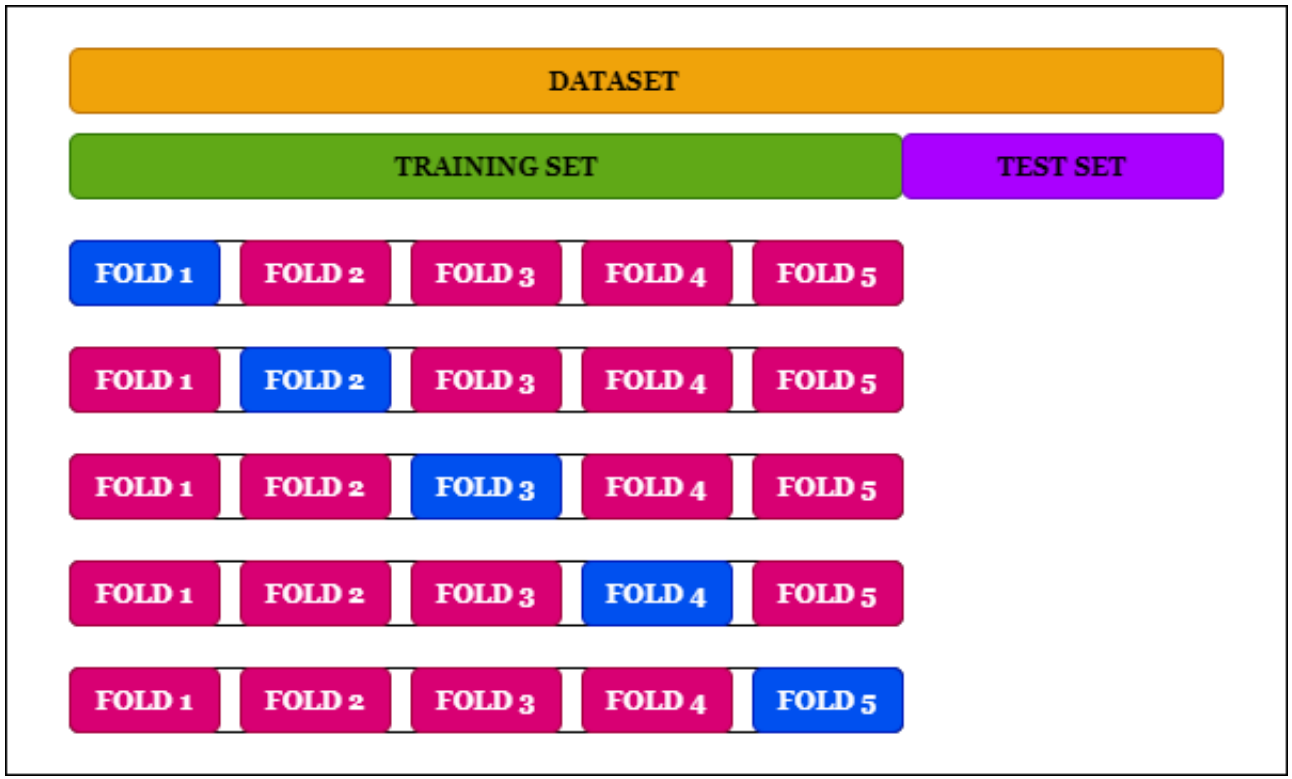
Regularization helps to prevent overfitting by penalizing large weights in the SVM model. The hyperparameter λ controls the strength of this penalty. We select the value of λ that results in the highest average cross-validation accuracy, balancing the model's complexity and its performance on unseen data.

Require: dataset, lambdaList
  lambdaAccuracies $\leftarrow$ []
  for all λ in lambdaList **do**
    crossValAccuracyList $\leftarrow$ []
    for trainingSet, testSet in split(dataset) **do**
      crossValAccuracy $\leftarrow$ trainAndTestSVM(trainingSet, testSet, λ)
      crossValAccuracyList.append(crossValAccuracy)
    end for
    lambdaAccuracies.append(mean(crossValAccuracyList))
  end for
  bestLambda $\leftarrow$ lambdaList[argmax(lambdaAccuracies)]

Two-Level Cross Validation

Purpose of Two-Level Cross Validation

Two-level cross-validation is an approach where an inner loop is used to determine the optimal hyperparameters (e.g., λ) for each fold of an outer cross-validation loop. The outer loop then assesses the performance of the model using the optimal parameters found by the inner loop. This approach provides a robust evaluation of the model's predictive power.



Two level cross-validation logic description

Selection of λ

We select the value of λ that results in the highest average cross-validation accuracy, balancing the model's complexity and its performance on unseen data.

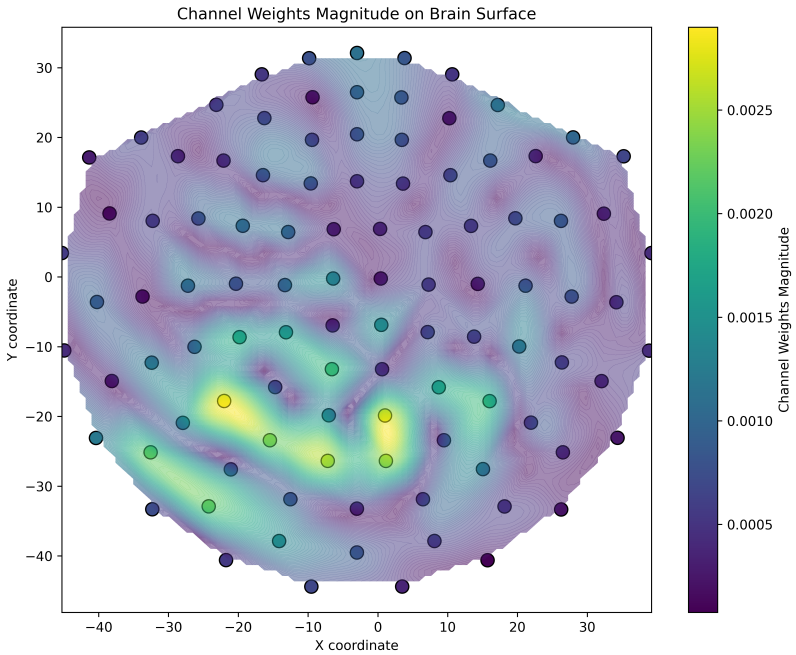
Pseudocode for Two-Level Cross Validation

```
Require: dataset, lambdaList
  crossVal1AccuracyList ← []
  for trainingSet1, testSet1 in split(dataset) do
    lambdaAccuracies ← []
    for λ in lambdaList do
      crossVal2AccuracyList ← []
      for trainingSet2, testSet2 in split(trainingSet1) do
        crossVal2Accuracy ← trainAndTestSVM(trainingSet2, testSet2, λ)
        crossVal2AccuracyList.append(crossVal2Accuracy)
      end for
      lambdaAccuracies.append(mean(crossVal2AccuracyList))
    end for
    bestLambda ← lambdaList[argmax(lambdaAccuracies)]
    crossVal1Accuracy ← trainAndTestSVM(trainingSet1, testSet1, bestLambda)
    crossVal1AccuracyList.append(crossVal1Accuracy)
  end for
  accuracy ← mean(crossVal1AccuracyList)
```

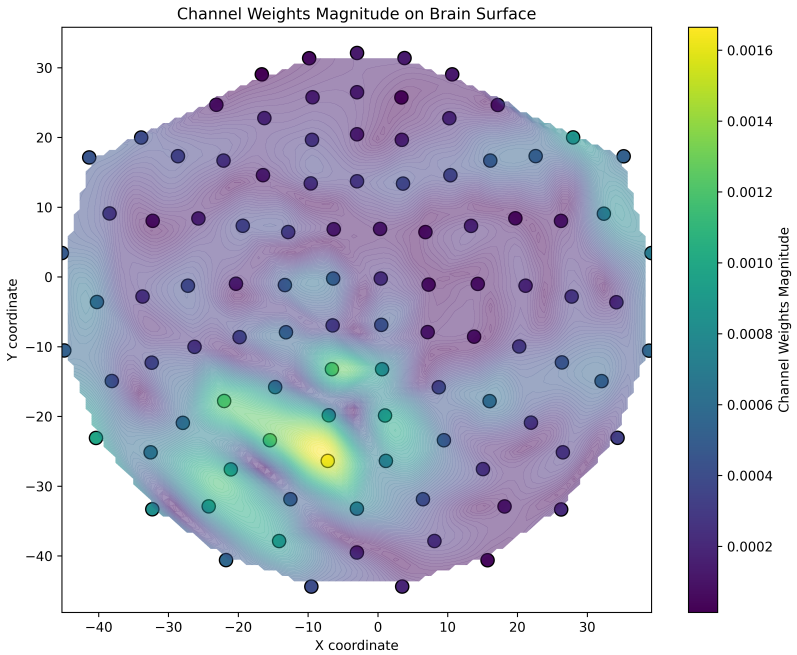
Comparing Channel Weights for Imagined vs. Overt Movement

Understanding Channel Weights

- The SVM classifier assigns weights to each channel based on its importance in the decision-making process. These weights, when visualized on the brain's surface, highlight areas of significant activity. Channels with higher weights are deemed more influential for predicting the movement type, providing insight into different brain regions' roles during actual and imagined movements.
- A channel with a high amplitude weight indicates that it is very effective in forecasting the preferred direction of the subject. By analyzing data obtained from both actual and mental movement, we may determine if distinct regions of the brain are stimulated during real motion or mere mental simulation of motion. The images for imagined and overt movement weights are attached as below slide.



Imagined Movement Weights



Overt Movement Weights

Imagined Dataset: Channel Weights Visualization

Interpretation of Channel Weights

After training our SVM model on the overt movement data, the channel weights can be visualized to understand their influence. Channels with higher weights are indicative of stronger influence in the decision-making process of the classifier. The mean AUC of each fold is **0.965**.

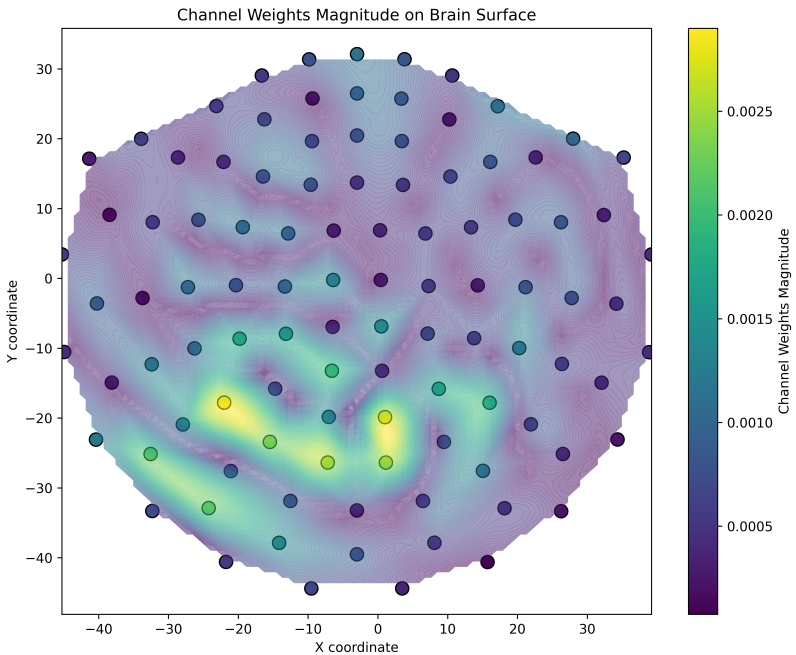


Figure: Channel weights mapped on the brain surface.

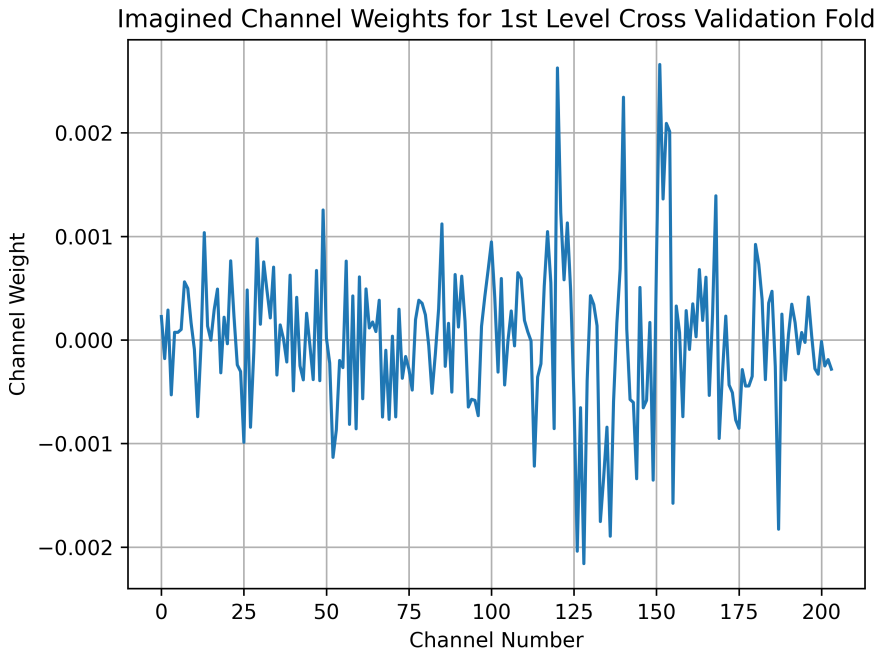


Figure: Channel weights for the first level cross-validation fold.

It will be seen that the channel weights are comparatively feeble in comparison to those for the Support Vector Machine (SVM) trained on the hypothetical dataset.

Channel Number	Channel Weight
122	0.009886
151	0.009889
186	0.005613
140	0.006120
163	0.006544
187	0.006645

Table: Imagined Dataset: List of the 6 dominant channels and their signed weights

Summary around Imagined Dataset

According to the imagined dataset, the channel weights increase as the numbering of the channels increases. Such a performance represents a left-handed shift of the tried graph. On the Brain surface table, the lower left hand is lighter in color and has a greater weight, building up two bright spots. Again, this indicates that subjects were trying to move their left hand.

Interpretation

The ROC curve analysis and accuracies across folds for the imagined dataset indicate a high-performing model with mean accuracy of **0.965**, with an emphasis on lower regularization, signified by the repeated selection of a smaller lambda value. This consistency allows for a more complex model that can capture the nuances of imagined movements.

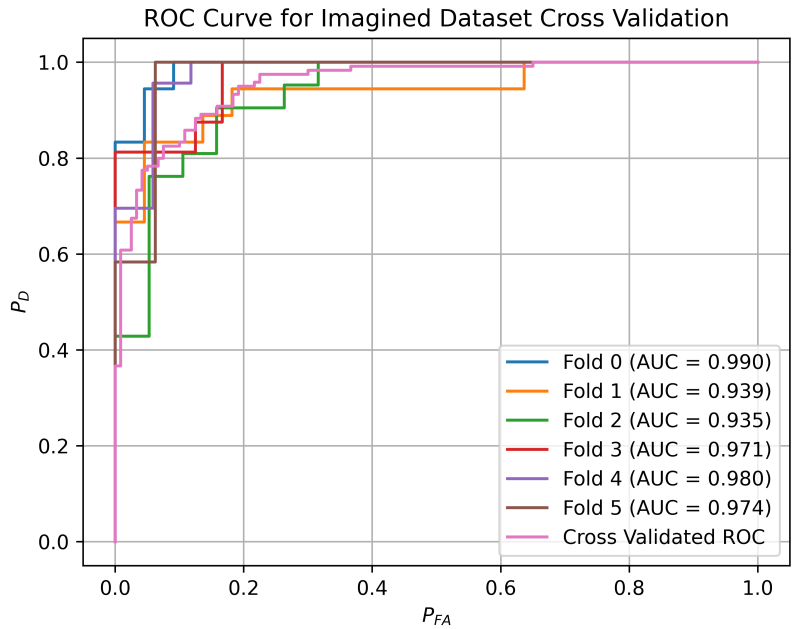


Figure: ROC Curves for each fold and cross-validated ROC for Imagined Dataset.

Table: Accuracies and Best Lambda Values for each Fold.

Fold	Accuracy	Best Lambda Value
0	0.990	1×10^{-6}
1	0.939	1×10^{-4}
2	0.935	1×10^{-6}
3	0.971	1×10^{-6}
4	0.980	1×10^{-6}
5	0.974	1×10^{-6}

Overt Dataset: Channel Weights Visualization

Interpretation of Channel Weights

After training our SVM model on the overt movement data, the channel weights can be visualized to understand their influence. Channels with higher weights are indicative of stronger influence in the decision-making process of the classifier. The mean AUC of each fold is **0.995**.

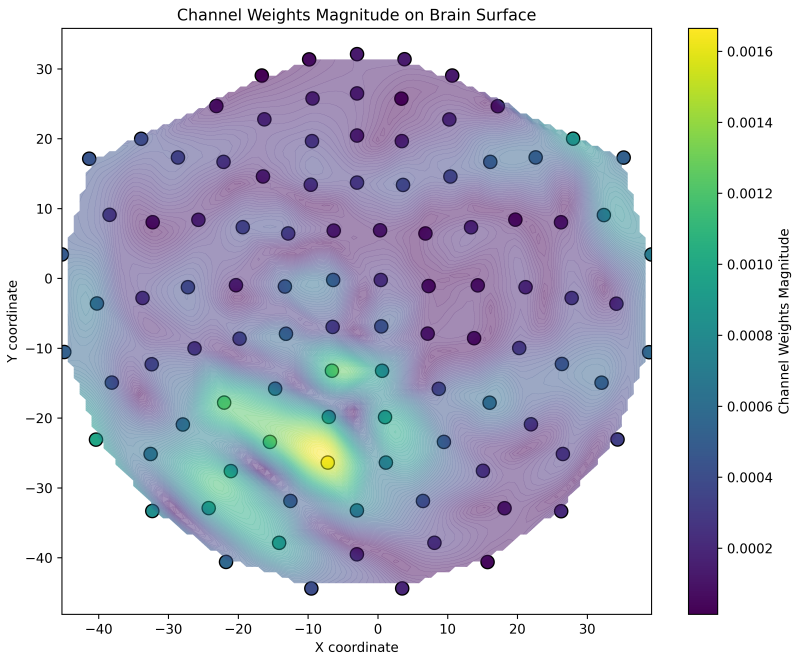


Figure: Channel weights mapped on the brain surface.

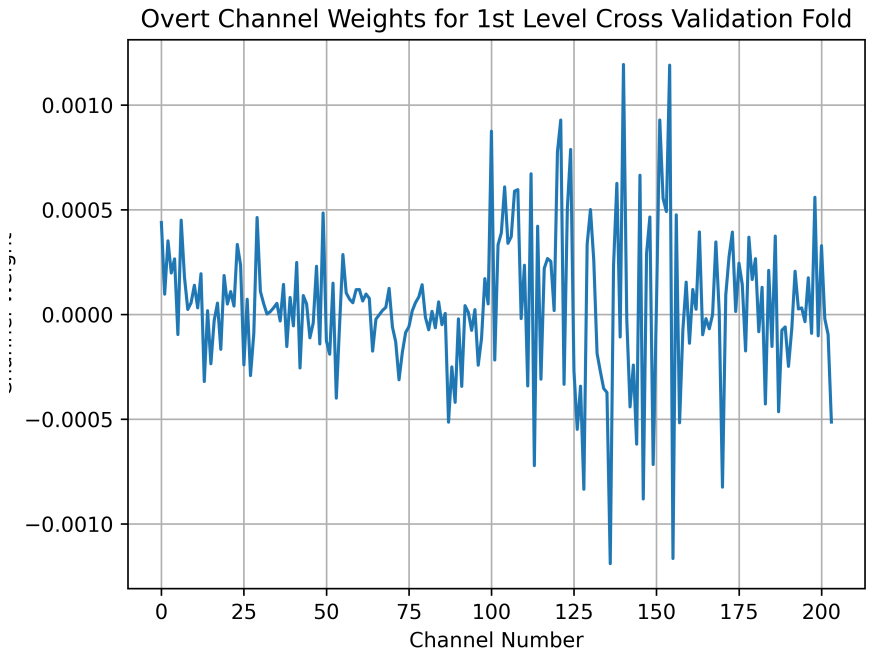


Figure: Channel weights for the first level cross-validation fold.

It will be seen that the channel weights are comparatively feeble in comparison to those for the Support Vector Machine (SVM) trained on the hypothetical dataset.

Channel Number	Channel Weight
202	0.004384
169	0.004876
104	0.005613
157	0.006120
156	0.006544
138	0.006645

Table: Overt Dataset: List of the 6 dominant channels and their signed weights

Summary around Overt Dataset

According to the overt dataset, the channel weights increase as the numbering of the channels increases. Such a performance represents a left-handed shift of the tried graph. On the Brain surface table, the lower left hand is lighter in color and has a greater weight, building up one long-oval shaped bright spots. Again, this indicates that subjects were trying to move their left hand.

Interpretation

The table shows that the model achieves high accuracy across different folds. It is noteworthy that the same small lambda value is selected multiple times, suggesting that the optimal model complexity for this dataset is low.

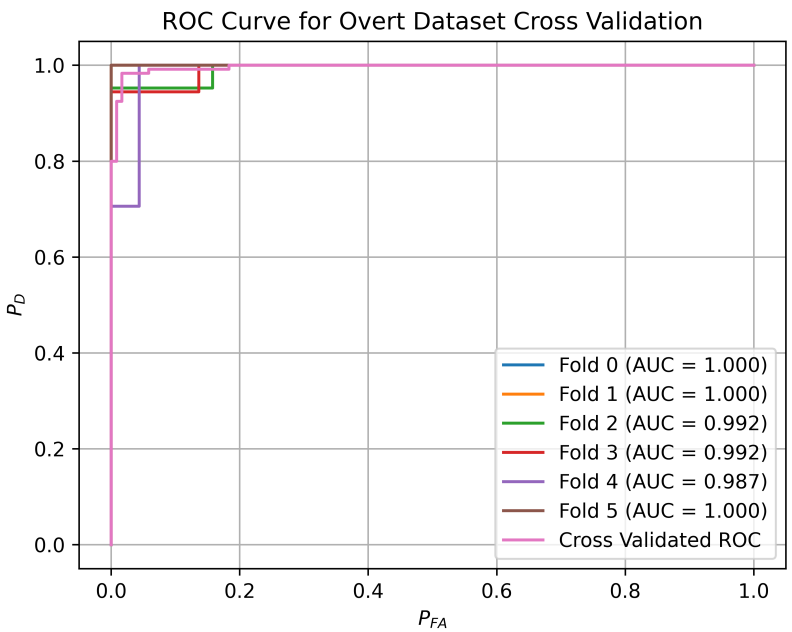


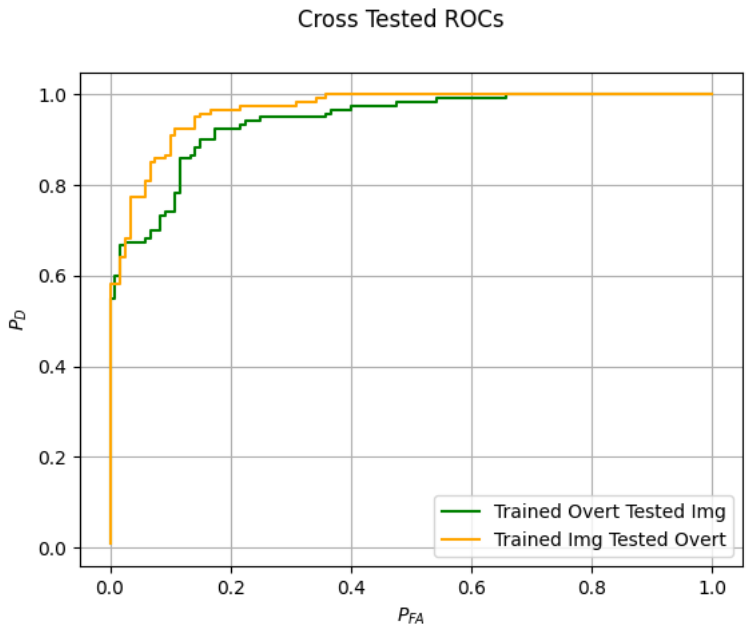
Figure: ROC Curves for each fold and cross-validated ROC for Overt Dataset.

Table: Accuracies and Best Lambda Values for each Fold.

Fold	Accuracy	Best Lambda Value
1	1.0	1×10^{-8}
2	1.0	1×10^{-8}
3	0.975	1×10^{-6}
4	0.95	1×10^{-5}
5	0.975	1×10^{-6}
6	0.95	1×10^{-5}

Analysis

For Same-Train cases, we evaluate the model using cross-validation and report both the ROC and accuracy metrics. In Cross-Train scenarios, after selecting the regularization parameter through cross-validation, the model is trained on the entire training set and then tested on a separate test set. The outcomes are quantified through the ROC and accuracy. The accuracy of ”Trained Overt Tested Imaged” is better than ”Trained Imaged Tested Overt”.



Training and Testing on Different Datasets

Training/Testing Scenarios

In order to examine the distinctions between the two datasets, I designed an experiment in which one dataset was used for training and the other for assessing. A random division was performed on both datasets to create a training set and a testing set. After selecting the optimal lambda value using a single level of cross-validation to train an SVM, I evaluated the model on one of these datasets.

Test	Train	Accuracy
Overt	**Overt**	0.9791
Overt	**Imagined**	0.8958
Imagined	**Imagined**	0.8333
Imagined	**Overt**	0.9583

The outcomes of the four potential permutations are displayed in the subsequent table. It can be observed that the SVM performs inadequately when tested on imagined data, but it performs well when tested on overt data, despite having been trained on the imagined dataset.

Construction of Feature Filtering Method

I attempted to use feature selection to lower the dataset's dimensionality. After doing cross validation, I identified the most effective channels and tested out different combinations of their numbers to see the impact on performance. Reducing the number of channels makes the model work with less data, but it also makes high dimensionality problems less of a problem.

Number of Channels	Accuracy for Overt	Accuracy for Imagined
204	0.975	0.865
180	0.979	0.874
150	0.958	0.894
120	0.975	0.894
90	0.982	0.919
60	0.958	0.910
30	0.949	0.883

Table: Feature filtering outcome.

From the table before, we can discovered that the model's performance was much enhanced by lowering the dimensionality. Results for the overt and imagined datasets were much better when 90 of the strongest channels were used. The overt dataset showed a 1% improvement, whereas the imagined dataset showed a 5.5% improvement. As a result of the noticeable performance boost, it is clear that the majority of the EEG channels do not contain very relevant information; hence, eliminating them improves dimensionality more than it hurts. It is evident that after 30 channels remain, an excessive amount of data is eliminated, leading to a decline in performance.

Possible Improvements 1

Increase the value range of regularization parameters

In the original requirements of the job, the regularization parameters have four values, namely 0.01, 1, 100, and 1000. In the actual experiment, I expanded the value range of the regularization parameters to 1e-10 to 1e+10, taking 20 values are obtained. This refines the value of the regularization parameter, which is helpful to improve the effect of SVM and significantly enhances accuracy, increasing by 1-2%.

Dataset	α **range from** $\{$**0.01, 1, 100, 1000**$\}$	α **range from [1e-10, 1e+10]**
Overt	0.954	0.975
Imagined	0.863	0.875

Table: Accuracy performance related to α range selection.

Possible Improvements 2

Experiment with different kernels

Four kernels—linear, polynomial, radial basis function/gaussian, and sigmoid—were tested in this experiment. For the whole 2-level cross validation, we utilised each kernel 10 times before averaging them. In my experience, compared to a linear kernel, the performance drops significantly when using any non-linear kernel. Because there are already a lot of features in the data, adding more complexity by translating it to a higher-dimensional feature space can make the model unusable for generalisation. This can introduce a non-linearity in the model, which can be useful if the data isn't linearly separable.

Kernel	**Accuracy**	
	Overt	**Imagined**
Linear	0.957	0.880
RBF	0.932	0.858
Polynomial	0.891	0.810
Sigmoid	0.919	0.804

Table: Experiment with different kernel.

Performance Compare between Overt and Imagined Dataset

We can noticed that the SVM weights for the imagined dataset are much bigger, which is probably because the imagined EEG signals are weaker. In contrast to the imagined dataset SVM, which used 45 support vectors on average, the overt dataset SVM used 34. As said before, this is probably due to the fact that the imagined dataset is more intricate and less readily separated than the overt dataset. More complicated imagined datasets tend to have more data points along the border, which means more support vectors as more of those points will be located inside the margin. The data distribution is the only element to be considered, since both SVMs had the least value for lambda.

Dataset	Overt	Imagined
Accuracy	0.975	0.875
Support Vectors Numbers	34	45

Table: Outcome Result

Compared to the imagined dataset, the overt dataset significantly improves the SVM's performance. Compared to the imagined dataset, the overt dataset yields around 10% better SVM accuracy. This makes perfect sense on an intuitive level, as really moving a hand would need more mental control than just seeing it moving. Consistent with other studies, this one also finds that physical than imagined processes have a better time classifying motion. The idea that real-life motion performance is more reliably represented by brain activity than only seeing it in one's head without any input from the senses lends credence to this. So, in contrast to the imagined dataset, the overt dataset will make EEG signal separation easier.

Analyzing Influences on SVM Performance

Dimensionality of the Dataset

The number of EEG channels used alters the dimensionality and, consequently, the accuracy of the classification. Our findings reveal that a reduction in dimensionality, by focusing on the most informative channels, improves accuracy, especially when using around 60 channels. This suggests that higher complexity in high-dimensional spaces may hinder the model's ability to generalize.

Dataset Characteristics: Overt vs. Imagined Movement

The type of movement data (overt vs. imagined) significantly affects model performance. In our experiments, models trained and tested on overt data achieved approximately 10% higher accuracy compared to those using imagined data, indicating a more challenging classification task for imagined movements.

Choice of Kernel in SVM

Experimentation with different SVM kernels demonstrated that nonlinear kernels significantly decreased model accuracy. This result aligns with our observation that increasing dimensionality—here, through the use of kernels—adversely impacts model performance. Conversely, reducing dimensionality enhances accuracy.

Conclusion

Our analysis underscores the importance of dataset characteristics, dimensionality, and kernel choice in SVM classification tasks. An optimal balance of these factors is crucial for achieving high model accuracy and generalizability.

Limits With The Approach

Recognizing the Boundaries of Our Methodology

1. **Performance with Imagined Data**: Our method does not perform optimally on imagined movement data, suggesting that the SVM's linear decision boundary may not capture the complex patterns present in EEG signals during imagined activities. This limitation could stem from inherently indistinct signals associated with imagined movements or from biases introduced by the model's simplicity.
2. **Lack of Domain Knowledge Utilized**: The efficacy of our machine learning model is also limited by a lack of integration of domain-specific knowledge from neuroscience. Extensive research exists on the classification of EEG signals, including differentiating between physical and imagined movements, as well as left versus right movements. Incorporating these insights could refine our feature selection and preprocessing, leading to improved model performance.

Moving Forward: Addressing these limits involves exploring non-linear models, enhancing domain expertise in feature engineering, and considering advanced preprocessing techniques. Collaboration with domain experts and reviewing current literature could significantly augment our classification accuracy and general understanding of EEG signal interpretation.

Possible Further Improvements

Innovations in Classifier Improvement and Validation

1. **Utilizing Domain Knowledge**: Incorporating neuroscience insights into our model could significantly boost its interpretability and accuracy. If specific brain regions are known to be involved in left/right hand movements or in differentiating between physical and imagined activity, integrating this knowledge could allow the SVM to focus on the most relevant EEG channels, thus potentially improving classification performance.
2. **Feature Extraction Techniques**: We have explored advanced feature extraction methods to refine the input data for our SVM. Techniques such as calculating statistical moments (e.g., mean, standard deviation) within brain lobes or applying wavelet transformations aim to distill raw channel data into more informative features. This not only reduces the dimensionality of the problem but also encapsulates important signal characteristics that raw data might not capture explicitly, potentially aiding in distinguishing between different types of movement.

Innovations in Classifier Improvement and Validation

1. **Incorporating Time-Series Data**: Incorporating time-series analysis could revolutionize the effectiveness of BCIs. EEG signals are dynamic and their temporal patterns carry rich information that could improve classification accuracy. Advanced models that can handle time-series data might be able to make more nuanced predictions, drawing upon the temporal fluctuations in brain activity to differentiate between left and right hand movements.
2. **Exploring Models Post Feature Extraction**: Post dimensionality reduction, the landscape of applicable models expands. With a reduced feature set, we could explore a variety of classifiers beyond SVM, such as Naive Bayes or relevant vector machines, which may excel in lower-dimensional spaces. This diversification could particularly benefit the classification of imagined movements, potentially overcoming the limitations of SVM's linear decision boundary.
3. **Broader Implications**: These enhancements point towards a BCI that is not only more accurate in its classifications but also more adaptable to the complexities inherent in EEG data. Embracing these methodologies could lead to breakthroughs in both the interpretability and functionality of machine learning applications in neuroscience.

Collaboration and Mutual Success

Collaboration Summary

In this assignment, it is necessary to communicate with your classmates and help them.

- **Qiqi Chen** and I discussed ways to improve accuracy. Chen suggested that we can expand the range of regularization parameters to improve the accuracy by using more regularization parameters. I think his suggestion is very reasonable and adopted it. At the same time, I also shared with her my improvement of accuracy by using feature filtering to retain dimensions with more features. He thought this was a very interesting and worthwhile area to look into.
- I was in trouble with two-level cross validation in code. My two-level cross validation part took a long time to run, and it took 5-7 minutes to calculate 20 alpha values on six blocks. Then **Jingjing Feng** helped me. She realized that I was using the KFlod function package to partition the blocks outside the two-level cross validation loop, so she suggested me to partition the blocks inside the two-level cross loop. I tried to follow her suggestion, and the speed of the code has been improved significantly. It now takes only a few seconds to complete the calculation.

Python Libraries

The following Python libraries were crucial in the development of our project:

- **NumPy** (np): Provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays.
- **Pandas** (pd): Offers data structures and operations for manipulating numerical tables and time series.
- **Matplotlib** (plt): A plotting library for creating static, interactive, and animated visualizations in Python.
- **SciPy** (scipy.interpolate, scipy.io): Used for various scientific computing tasks such as signal processing, optimization, interpolation, and file I/O.
- **Scikit-learn**: Used for machine learning and statistical modeling including classification, regression, clustering, and dimensionality reduction. Specific modules used:
  - `train_test_split`, `KFold`, `GridSearchCV` for model selection and evaluation.
  - `StandardScaler` for preprocessing.
  - `SVC` for implementing Support Vector Machine classifier.
  - `roc_curve`, `auc`, `accuracy_score` for model evaluation metrics.

References: Citations

Suthaharan, Shan.
Support vector machine.
In *Machine learning models and algorithms for big data classification: thinking with examples for effective learning*,
pages 207–235. Springer, 2016.

Ruslan Klymentiev,
EEG Data Analysis: Alcoholic vs Control Groups,
January 5, 2019,
https://rklymentiev.com/post/eeg-preprocessing-erp/.

Fushiki, Tadayoshi.
Estimation of prediction error by using K-fold cross-validation.
Statistics and Computing, 21:137–146, Springer, 2011.

Maldonado, Sebastián and López, Julio.
Dealing with high-dimensional class-imbalanced datasets: Embedded feature selection for SVM classification.
Applied Soft Computing, 67:94–105, Elsevier, 2018.

Browne, Michael W.
Cross-validation methods.
Journal of Mathematical Psychology, 44(1):108–132, Elsevier, 2000.