MP 2 - Brain-Computer Interface Movement Decoding

Jake Vigeant

April 29, 2023

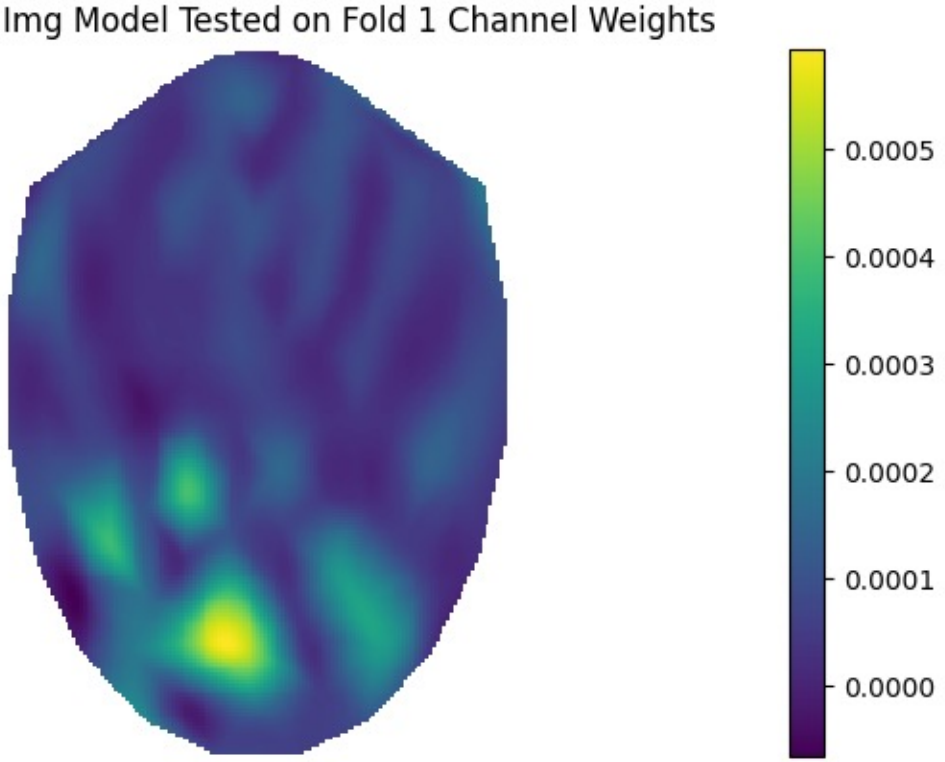
Results

Two-level cross-validation (single level)

Imagined Data – Fold 1

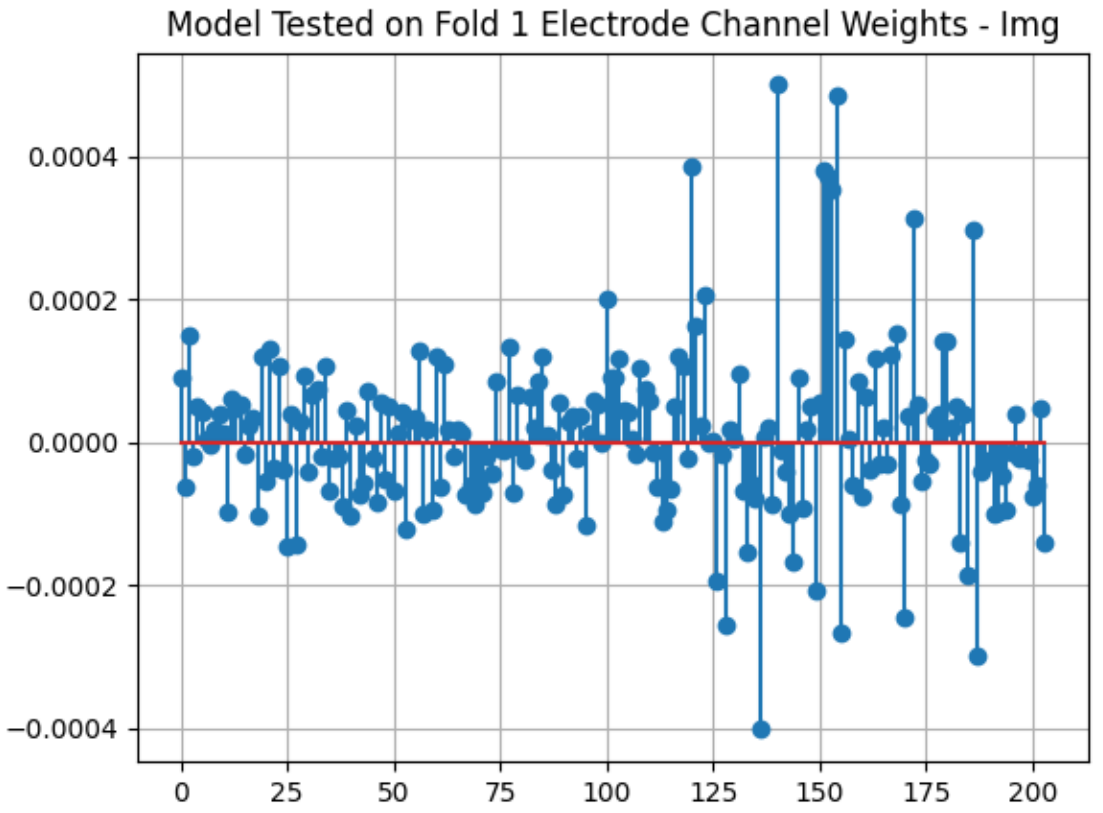
Electrode Weights on Brain Surface

This plot illustrates the model weights obtained from the model trained on folds 2-6 and tested on fold 1 of the first level of cross validation on the Imagined dataset. The visualization is layed out such that the forehead is the thinnest secion at the top, the nape of the neck is in the back, and the left and right regions of the brain are represented by the left and right halves of the image. Prior to making fold assignments, the data was shuffled to ensure representative folds. After determining the value of the regularization parameter in the second-level of cross validation, an SVM (Support Vector Machine) model with no kernel was fit to the data contained in folds 2-6, with fold 1's data points withheld for testing. The LinearSVC implementation from SciKitLearn was used. This generated an SVM model with a weight for each electrode channel feature. As such, the absolute value of the 204-lengthed weight vector could be visualized on the brain using the provided showChanWeights plotting function. The provided colorbar indicates that darker blue indicates a lower weight magnitude, while more yellow indicates a higher weight magnitude. There are several regions of the mapping that exhibit the darkest blue coloration, indicating that the electrode placed at that location holds a weight of 0 in the SVM model. There are few locations with dark yellow/green coloration, indicating that their magnitude disproportionally affects the weights vector. Thus, very few of the 204 electrode readings influence the SVM decision function in this model. The electrodes that are prominently lit are located near the nape of the neck in the visualization. This connects with the neuroscientific context of the data, as the cerebellum (proposed to be responsible for movement) is in this region.



Electrode Weight Stem Plot

This plot illustrates the electrode channel weights obtained from the SVM model that was trained on folds 2-6 and tested on fold 1 in the first layer of cross validation in the Imagined dataset. The LinearSVC implementation from SciKitLearn was used. Prior to making fold assignments, the data was shuffled to ensure representative folds. This model had its regularization parameter value selected by the second level of the cross validation, conducted on the training folds 2-6. The model was then fit to the training folds, yielding the 204-element vector that assigns a weight to each electrode channel feature as shown. The x-axis of the plot gives the electrode channel index. The y-axis of the plot gives the corresponding weight value. Most of the stems are close to the origin, indicating that these electrode features have little to no weight on the model. Few of the features have much higher magnitudes than the rest, indicating that they have a disproportionally large effect on the SVM model.



Largest Magnitude Electrode Weights

The table provides the 6 largest magnitude electrode channel weights from the no kernel SVM model trained on folds 2-6 and tested on fold 1 from the first level of the 2-level cross validation for the Imagined dataset. The LinearSVC implementation from SciKitLearn was used. This model had its regularization parameter value selected by the second level of the cross validation, conducted on the training folds 2-6. Prior to making fold assignments, the data was shuffled to ensure representative folds. This was obtained by sorting the 204-element weight vector by the absolute value of its entries, then constructing a table in matplotlib with the 6 highest. The electrodes with the highest values appear to be related spatial, as the variation in index among the top 6 electrodes is small compared to the total number of electrode readings comprising the weight vector.

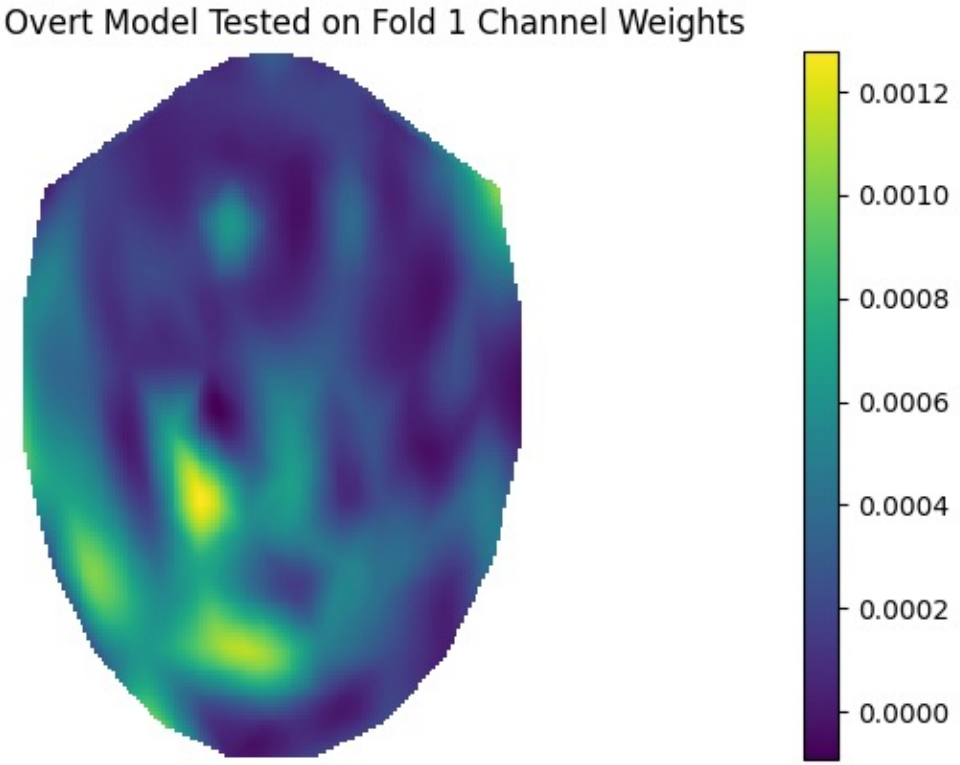
Model Tested on Fold 1 Top 6 Electrode Weights - Img

140	0.0004997687281077232
154	0.00048377220338120786
136	-0.000402002152331217
120	0.0003841217369485494
151	0.00037946009821204004
152	0.0003709673571650968 6

Overt Data – Fold 1

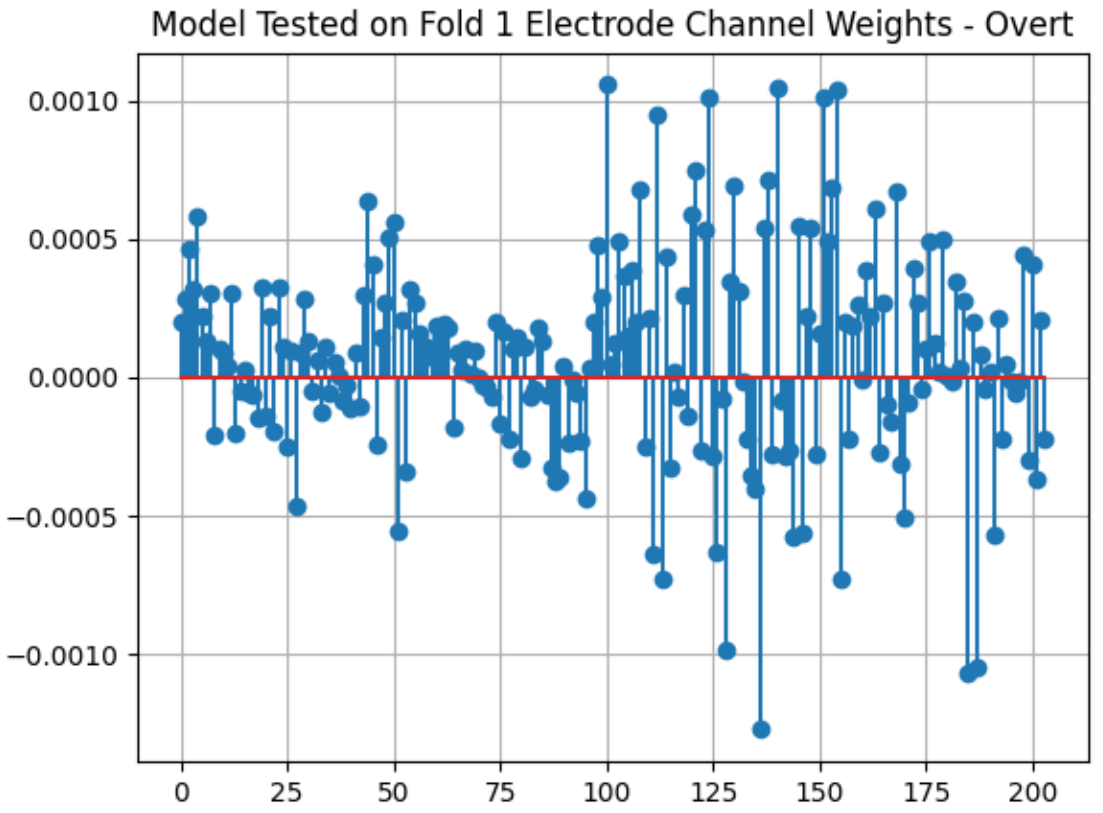
Electrode Weights on Brain Surface

This plot illustrates the model weights obtained from the model trained on folds 2-6 and tested on fold 1 of the first level of cross validation on the Overt dataset. The visualization is layed out such that the forehead is the thinnest secion at the top, the nape of the neck is in the back, and the left and right regions of the brain are represented by the left and right halves of the image. Prior to making fold assignments, the data was shuffled to ensure representative folds. After determining the value of the regularization parameter in the second-level of cross validation, an SVM (Support Vector Machine) model with no kernel was fit to the data contained in folds 2-6, with fold 1's data points withheld for testing. The LinearSVC implementation from SciKitLearn was used. This generated an SVM model with a weight for each electrode channel feature. As such, the absolute value of the 204-lengthed weight vector could be visualized on the brain using the provided showChanWeights plotting function. The provided colorbar indicates that darker blue indicates a lower weight magnitude, while more yellow indicates a higher weight magnitude. There are several regions of the mapping that exhibit the darkest blue coloration, indicating that the electrode placed at that location holds a weight of 0 in the SVM model. There are few locations with dark yellow/green coloration, indicating that their magnitude disproportionally affects the weights vector. Thus, very few of the 204 electrode readings influence the SVM decision function in this model. The electrodes that are prominently lit are located near the nape of the neck in the visualization. This connects with the neuroscientific context of the data, as the cerebellum (proposed to be responsible for movement) is in this region.



Electrode Weight Stem Plot

This plot illustrates the electrode channel weights obtained from the SVM model that was trained on folds 2-6 and tested on fold 1 in the first layer of cross validation in the Overt dataset. The LinearSVC implementation from SciKitLearn was used. Prior to making fold assignments, the data was shuffled to ensure representative folds. This model had its regularization parameter value selected by the second level of the cross validation, conducted on the training folds 2-6. The model was then fit to the training folds, yielding the 204-element vector that assigns a weight to each electrode channel feature as shown. The x-axis of the plot gives the electrode channel index. The y-axis of the plot gives the corresponding weight value. Most of the stems are close to the origin, indicating that these electrode features have little to no weight on the model. Few of the features have much higher magnitudes than the rest, indicating that they have a disproportionally large effect on the SVM model.



Largest Magnitude Electrode Weights

The table provides the 6 largest magnitude electrode channel weights from the no kernel SVM model trained on folds 2-6 and tested on fold 1 from the first level of the 2-level cross validation for the Overt dataset. The LinearSVC implementation from SciKitLearn was used. This model had its regularization parameter value selected by the second level of the cross validation, conducted on the training folds 2-6. Prior to making fold assignments, the data was shuffled to ensure representative folds. This was obtained by sorting the 204-element weight vector by the absolute value of its entries, then constructing a table in matplotlib with the 6 highest. The electrodes with the highest values appear to be related spatial, as the variation in index among the top 6 electrodes is small compared to the total number of electrode readings comprising the weight vector.

Model Tested on Fold 1 Top 6 Electrode Weights - Overt

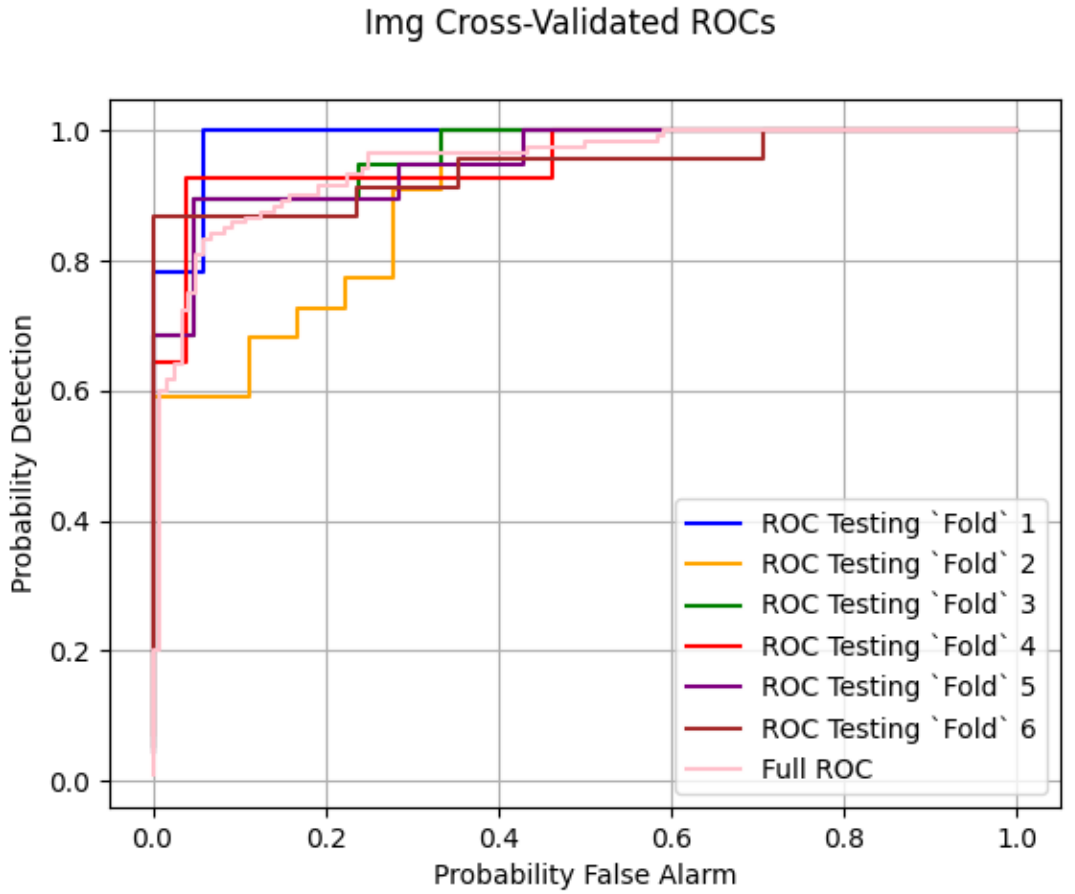
136	-0.0012732961180206316
185	-0.0010720561971510304
100	0.0010592298913671234
187	-0.0010487340243573028
140	0.0010485654275378298
154	0.0010381271814033288

Two-level cross-validation (all levels)

Imagined Data

Fold-Level ROC Curves

This plot shows the ROC performance curves generated for each of the 6 linear SVM models yielded by the 1st layer of the shuffled 6-fold cross validation in the Imagined dataset. Each fold was successively held out for training, and an SVM model was fit to the remaining 5 using the regularization parameter value that was determined by the 5-fold 2nd cross validation layer. The model was then tested on the held-out fold from the first layer by calculating the decision statistic for each point. The decision statistic was calculated using a provided scikit learn function whose internal implementation dots the model's 204-element weight vector with the 204 electrode features for each data point and adds on an intercept term. Then, an ROC curve was generated by considering each resulting decision statistic as a potential threshold. The full ROC curve shown in pink represents the holistic performance across all the folds. This was generated by aggregating the decision statistics generated for each point when it was in the testing fold. The individual fold ROCs appear to vary around the full ROC curve. Some perform better, while some bow in further. It seems that the full ROC curve is the average of the individual fold curves at each point.



Fold-Level Classifier Accuracies

This table lists the performance for each of the 6 linear SVM models yielded by the 1st layer of the shuffled 6-fold cross validation on the Imagined dataset. Each fold was successively held out for training, and an SVM model was fit to the remaining 5 using the regularization parameter value that was determined by the 5-fold 2nd cross validation layer. The LinearSVC implementation from SciKitLearn was used for the SVM. The full ROC accuracy was obtained by aggregating the decision statistic and resulting classification for each data point when it was included in the testing fold once. The performance metric displayed is the probability of a correct decision, calculated as the number of correct classifications over the total number of testing points. The accuracies were calculated with a decision statistic threshold of 0. The accuracy for the full ROC is the average of the values for each fold. The variance between each fold and the overall mean is low enough such that each strand represents the complete accuracy reasonably well.

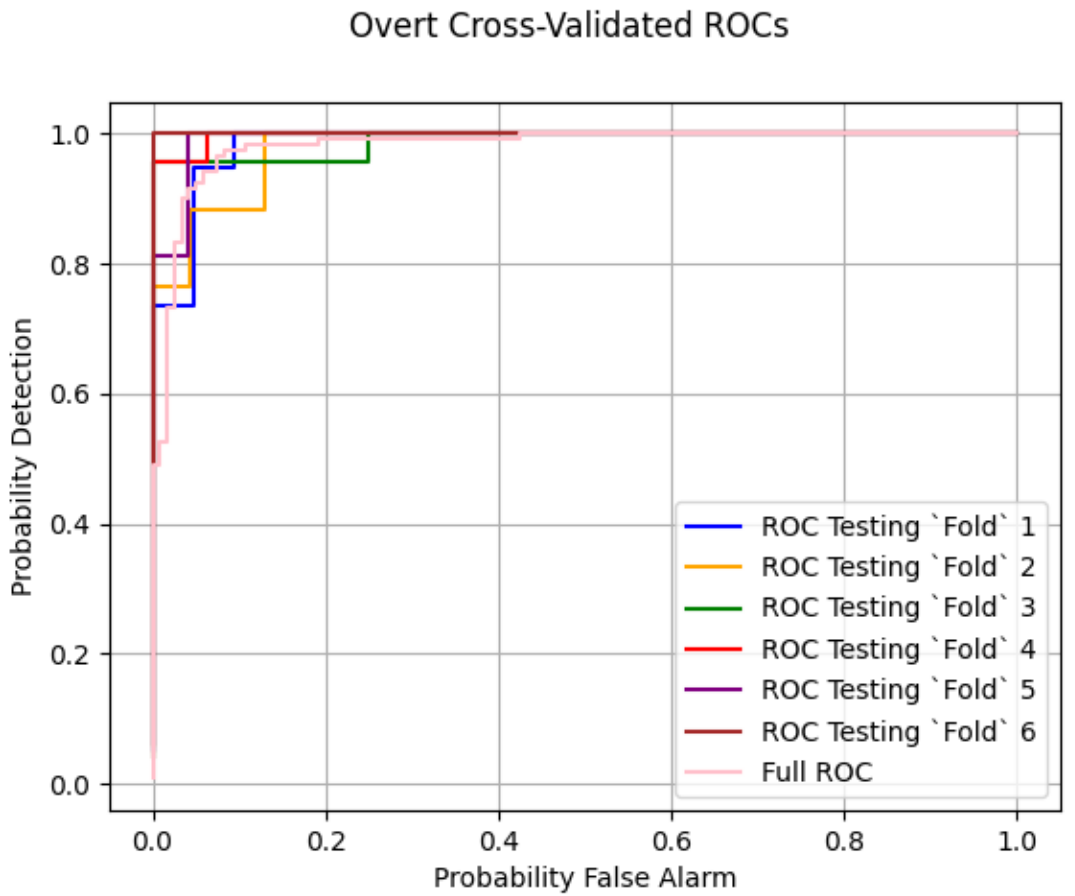
Fold Accuracies P(cd) - Img

ROC Testing Fold 1	87.5%
ROC Testing Fold 2	77.5%
ROC Testing Fold 3	92.5%
ROC Testing Fold 4	90.0%
ROC Testing Fold 5	92.5%
ROC Testing Fold 6	92.5%
Full ROC	88.75%

Overt Data

Fold-Level ROC Curves

This plot shows the ROC performance curves generated for each of the 6 linear SVM models yielded by the 1st layer of the shuffled 6-fold cross validation in the Overt dataset. Each fold was successively held out for training, and an SVM model was fit to the remaining 5 using the regularization parameter value that was determined by the 5-fold 2nd cross validation layer. The model was then tested on the held-out fold from the first layer by calculating the decision statistic for each point. The decision statistic was calculated using a provided scikit learn function whose internal implementation dots the model's 204-element weight vector with the 204 electrode features for each data point and adds on an intercept term. Then, an ROC curve was generated by considering each resulting decision statistic as a potential threshold. The full ROC curve shown in pink represents the holistic performance across all the folds. This was generated by aggregating the decision statistics generated for each point when it was in the testing fold. The individual fold ROCs appear to vary around the full ROC curve. Some perform better, while some bow in further. It seems that the full ROC curve is the average of the individual fold curves at each point.



Fold-Level Classifier Accuracies

This table lists the performance for each of the 6 linear SVM models yielded by the 1st layer of the shuffled 6-fold cross validation on the Overt dataset. Each fold was successively held out for training, and an SVM model was fit to the remaining 5 using the regularization parameter value that was determined by the 5-fold 2nd cross validation layer. The LinearSVC implementation from SciKitLearn was used for the SVM. The full ROC accuracy was obtained by aggregating the decision statistic and resulting classification for each data point when it was included in the testing fold once. The performance metric displayed is the probability of a correct decision, calculated as the number of correct classifications over the total number of testing points. The accuracies were calculated with a decision statistic threshold of 0. The accuracy for the full ROC is the average of the values for each fold. The variance between each fold and the overall mean is low enough such that each strand represents the complete accuracy reasonably well.

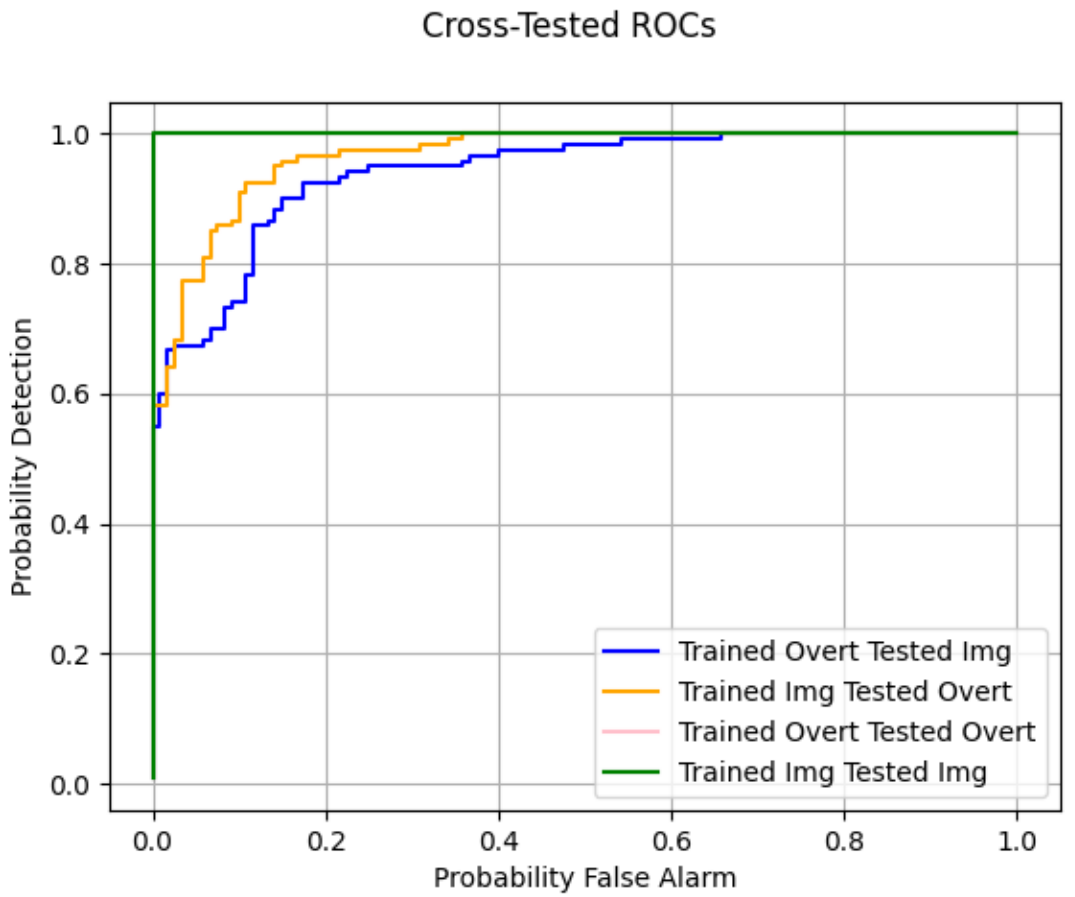
Fold Accuracies P(cd) - Overt

ROC Testing Fold 1	92.5%
ROC Testing Fold 2	90.0%
ROC Testing Fold 3	97.5%
ROC Testing Fold 4	90.0%
ROC Testing Fold 5	97.5%
ROC Testing Fold 6	92.5%
Full ROC	93.33%

Classification performance

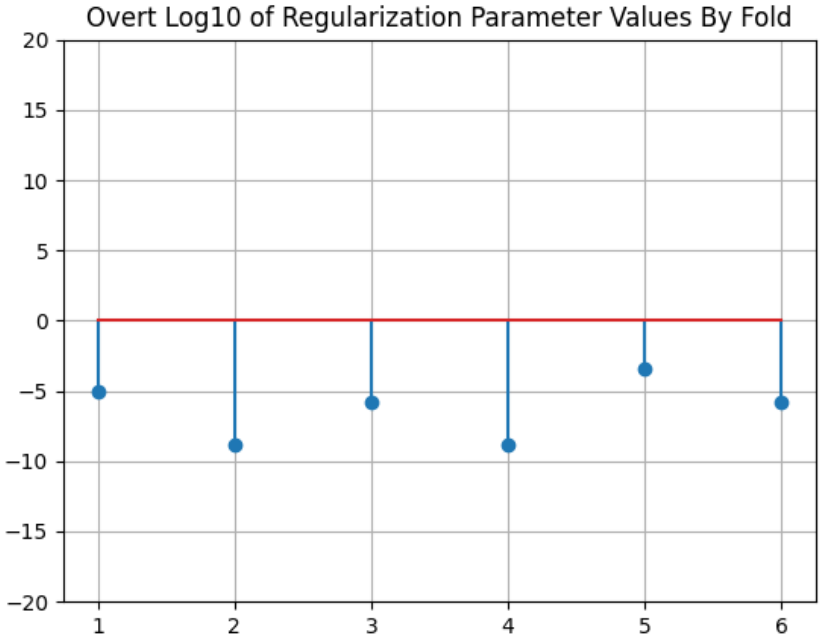
Cross-Data ROC Curves

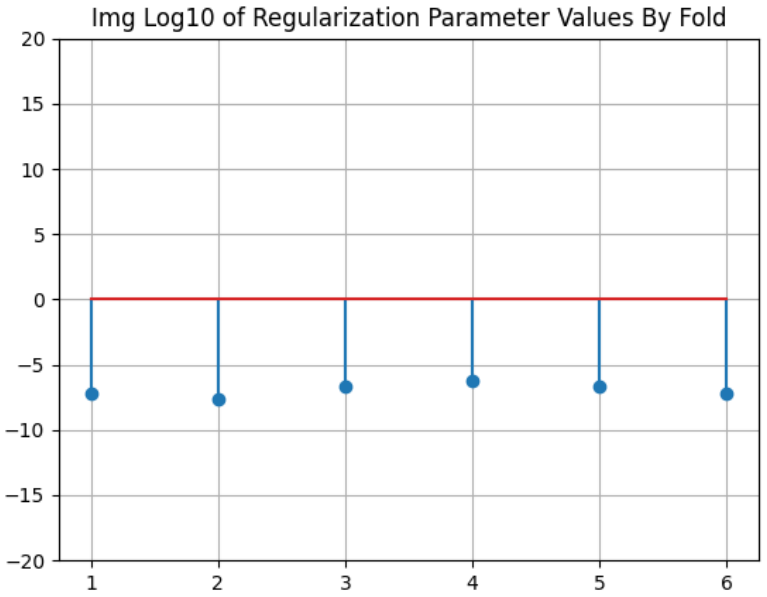
This ROC curve was generated by training a linear SVM model, implemented with SciKit Learn's LinearSVC object on each of the two datasets. Then, each model was tested on the opposite dataset. The ROC curves were constructed by using each testing decision statistic as a threshold. This generated the 2 cross-data ROC curves shown in the plot and labeled in the legend. The ROC curves from incestuous training/testing on each dataset are also shown to illustrate ideal performance for each model. The ideal case for both the Overt and Imagined datasets is the perfect ROC curve, indicating that the datasets are perfectly linearly separable and that a separating hyperplane that misclassifies 0 points is obtainable. The Imagined trained model performed better than the Overt trained model in cross-testing, as the blue curve bows in further in the plot than the orange one. This performance difference makes sense given the data context. There is likely more noise and variance in the Imagined dataset than the Overt one due to the nature of brain scans (imagining a movement is likely to generate a weaker signal than actually performing it). The fold-level accuracies support this, as the Imagined models performed worse on average than the Overt ones. So, the model trained and accustomed to this variance would perform better on the new Overt data that is more simple to predict. Conversely, the easier Overt data will generate a model unable to handle this variance with a margin that is potentially too large. At roughly the 65% probability of detection mark, the trained on Overt performs slightly better than the trained on Imagined. This indicates that the models have slightly different slopes/boundary shapes substantiating the idea that the slope of the Overt model may be too flat to accommodate the variance exhibited in the Imagined dataset.



Regularization Parameters by Fold

These stem plots depict the alpha values for the Linear SVM models fit in each step of the first-level of cross validation. The alpha value is the regularization parameter of the model. This parameter determines the weighing of the classification loss for the model against the coefficient magnitude. In the case of an SVM, larger coefficients mean that the margin will be smaller, so the square of the weight vector is regularized. The X-axis indicates the testing fold number, while the Y-axis is logarithmically scaled to indicate the power of 10 at which the regularization parameter fell. For each dataset, LinearSVC models from SciKit learn were fit using two-level cross validation, one level to set aside a testing fold for the entire model, and a second level to select an alpha value for that iteration. A wide range of alpha values was used, with 10^-15 all the way through 10^15 (3 values per decade) being tested at the second cross validation level. The performance metric used was probability of correct decision for this. There is more variance in the alpha values for the Overt dataset models than there is for the Imagined ones. The optimal Imagined parameter appears to vary closely around 10^-7 based upon the specific random fold used. For the Overt dataset, the values span from 10^-3 to almost 10^-10. The average value for both datasets appears about the same, but the Overt variance is much higher. This variance is likely due to variations in the specific datapoints used for each fold instead of underlying variance in the data, as the cross-fold performance for each fold did not vary appreciably. So, this variance is likely the product of a lower underlying classification loss for this dataset. The overall loss depends more on the regularization term as the Overt classification loss is lower than that of the Imagined dataset. So, the regularization parameter will change more to distinguish the models between folds without the loss term dominating.





Explanation/Interpretation Prompts

Project Approach

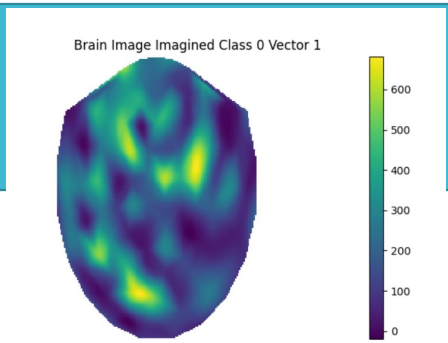
Efficacy of SVM's With High-Dimensional Data and Few Training Observations

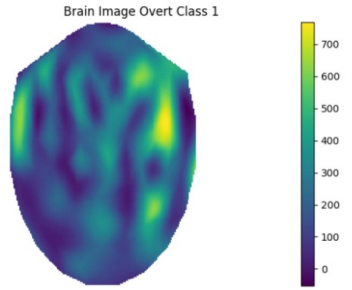
It is problematic for the other classifiers discussed in this course when the amount of data provided is unable to adequately cover a problem with many features. Support Vector Machines (SVM), however, are able to overcome this as they produce a model that is sparse in many of the features. In training an SVM, the loss function includes a term that weighs the size of the margin for the classifier's boundary. The margin is the distance from the closest datapoint to the decision boundary. The margin is affected only by points close to the boundary, since any points further than those near the boundary would inherently have a larger margin and not be the limiting factor for the classifier's overall margin. As such, the decision boundary is shaped heavily by the few observations clustered near it that determine the margin. This feature imposes sparsity on the weight vector that represents the decision boundary, as any irrelevant features will be excluded if they do not affect the datapoints that comprise the separating hyperplane. As the hyperplane is the product of few points, data that contain outliers may adversely affect performance moreso than other more holistic models. In general, SVMs can fit any arbitrary boundary if a kernel is used, although this was not the case in this project. So, they can work to separate data in a high dimensional space where the correct decision boundary is not visualizable.

Two-Level Cross-Validation's Helpfulness

The two-level cross validation performed in this project provides benefit as it increases the robustness of the model training regimen. The two-level cross validation implemented in this project split the full dataset into 6 folds after shuffling. Shuffling was performed to randomly reorder the data and ensure that the classes were properly distributed among the folds. This was necessary as the data files were organized such that each file contained data points corresponding to only one class. Then, one of the 6 folds was set aside per iteration to serve as a testing dataset. The remaining 5 were then fed into the second level of the cross validation. Here, the 5 training folds from level 1 were split into 5 new folds after another round of shuffling. One of these folds was set aside per iteration for testing, and the remaining 4 were used to fit a linear SVM with a list of alpha values. Performance on the testing fold determined which value of alpha was optimal. This alpha value was then used to fit a linear SVM model to the 5 training folds from the first level of the cross validation. The performance was computed for each of the 6 models along with an ROC curve, and the final model selected was the one that provided the best performance of the 6. Single level cross validation would require the regularization hyperparameter to be fixed between folds. This would have to be selected by trial and error of several values prior to the cross validation in a process similar to the second level of the two-level cross validation. But, this would constrain the same value of alpha for each of the models generated. So, allowing each model to select its own alpha based upon the data which it is trained upon is a natural approach that would appear to increase the performance of each model. This approach provides the largest benefit in that it helps mitigate overfitting to the training data. If there were no unseen data to test on, the model would likely be too biased towards the data which it was trained on, producing high variance in results for new data. The additional benefit to including a second layer of cross validation is that it also diminishes the likelihood that the model will overfit the testing data. In single-level approaches, simply the highest performance on the testing set is chosen. The second level allows performance to optimize for the hyperparameter as well, a feature of the model that will generalize beyond the specific datapoints on which it happened to be tested.

Domain knowledge and Electrode Data Mapping





The context for our data lies in the neuroscience behind brain scans and muscular movement signals. The cerebellum is the brain region of interest for this project, as it is responsible and active during complex movement (National Center for Biotechnology Information). The cerebellum is located in the back of the brain near the brain stem. The image of a top-down cross section, similar to that of the presented brain visualizations in this project, indicates this to be the case. Additionally, for class 1 (right movement), the left side of the brain is active, and the converse is true for class 0. This is consistent with neuroscience that claims the left hemisphere of the brain operates the right side of the body and so forth. So, it would appear near the back of the brain image visualizations presented in this project. Thus, the movement plots in this project should include activity in the back of the brain near the cerebellum. Visualizations of data from both datasets illustrate this as shown in the plots presented on the right. Each plot has strong levels of activity at the bottom of the image, near where the cerebellum would be. Further, understanding the difference between imagining a movement and actually performing it helps clarify the data used in this project. If the data were given to me blindly, it may be hard to generate performance expectations on the relative datasets. It may also be hard to interpret the cross-data tested models and understand the performance discrepancies between the two. But, knowledge that the data sets represent overt versus imagined movement helps inform the assumptions that the second dataset should be noiser. Understanding this clarifies the classifier performance throughout, as this variance helps inform the results and margin achieved by all the classifiers.

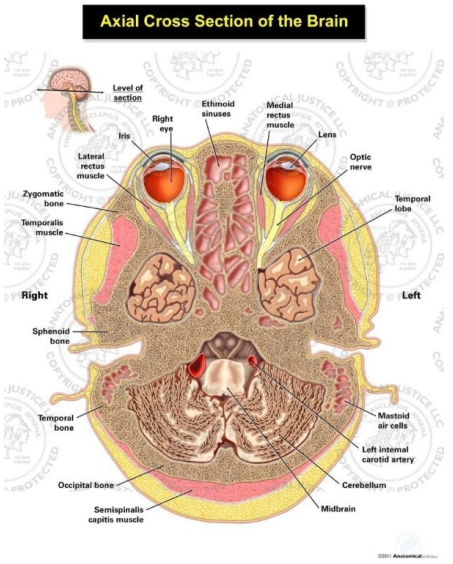


Image from (Anatomical Justice)

Project Results

Differing ROC Curves

The ROCs obtained from cross-testing the Overt and Imaginary models differ in performance meaningfully. As can be seen in the Cross-Data ROC curves section, the model trained on the Overt dataset performed worse on the Imaginary dataset than the model trained on the Imaginary dataset performed on the Overt dataset. Both datasets themselves are fully separable by a linear boundary in space, as the incestuously trained/tested ROC curves for each are perfect. So, this discrepancy is likely due to the variable nature of these respective datasets. This would come into play in testing on newly seen data. A brain scanned in the process of making an overt movement likely provides stronger indication of the movement direction than a brain scanned while imagining the same movement. Each of the brain scans presented in this project showcase the strongest activity in the back of the brain, near the cerebellum. The cerebellum activates more strongly when it is time to send the signals needed to actuate the movement, so the overt datapoints should be less noisy and variant than those of the imagined dataset. This is supported by the fact that the fold-level accuracies are lower for the Imagined dataset than they are for the Overt. From this, a model trained on the Overt dataset may over-regularize as the classification task is easier. If the model overregularizes, the slope of the separating hyperplane will be too low to accommodate high variance in any new testing data. Since the model was trained on solely overt data and then tested on the imaginary data, we would expect the model to perform poorly on the new degree of variance. The converse is true for the model trained on the Imagined dataset. That model would not be able to regularize as much, so the hyperplane will be steeper. This steepness would overfit the Imagined dataset more, but, as this is the more difficult of the two datasets, testing it on easier data later will allow for better overall performance. There are some blips in the curves where they intersect or the overt-trained model outperforms. This further indicates that the shape of the hyperplanes differ between models. These pointwise disparities can arise from examining decision statistic thresholds that far extend the margin for the Imagined model.

Differing Cross-Fold Performance Consistency

The performance on the Overt and Imagined datasets differs appreciably across the folds and holistically. The full model cross-validated accuracy for the Overt data (representative of the average of each fold's performance) was higher than that of the Imagined data by about 5%. The primary confounding variable between these two datasets is the context under which they were generated. The brain scan of an individual imagining a movement likely is nosier than that of an individual performing the same movement. So, one would expect that the variance near a separating hyperplane would be higher for the Imagined dataset than the Overt dataset due to this noise. Increased boundary variance will diminish the margin of the SVM model that results from the training regimen. Higher margins leave the model with more space to accurately predict new datapoints as they are seen by the model. The accuracies were generated by examining the probability of correct decision for the model on the unseen testing fold. So, the cleaner, higher margin, models generated in training on the Overt data folds should generalize better to the new data than that of the Imagined dataset. The ROC curves generated from training and testing on the whole of both the Overt and Imagined datasets are perfect. This indicates that there was no underlying difference in geometry that prevented one dataset from being classified as well as the other. Thus, variance in model accuracy must be due to changes in the orientation of the boundary.

The variance in performance across folds is higher for the Imagined data than it is for the Overt as well. This means that the specific data points selected for each training fold influenced the model more substantially than that of the more stable Overt dataset. For this to be true, the variance among the data must be higher. SVMs are very sensitive to outliers and variance as the final model depends fully on the points situated near the boundary. So, in a more variable dataset it is to be expected that the model will vary more based on the specific points selected for the training folds. More variable models generate more variable performance as they will have smaller margins, diminishing the probability of correct detection for some testing folds, while increasing it for others. This further supports the previous conclusion drawn.

Domain Knowledge and Electrode Channel Weights Mapping

Understanding the source and context of the project's data helps to inform understanding of the model's that result from training on it. Specifically, with neuroscientific belief regarding brain activation and motor response, it is to be expected that the most relevant neuronal regions in triggering and thinking about movement are in the cerebellum and the hemispheres opposite the direction of movement. So, it is to be expected that brain visualizations of the model weights would indicate prominence in the cerebellum and hemispheric regions that dictate the direction to move. The problem necessitates that few of the electrodes represent the classification result, as these surfaces represent a fraction of the total number of electrodes. This makes a sparse SVM well suited for this classification problem. Understanding this context allows for quick generalizations and analyses to be made regarding the correctness and feasibility of an SVM model, as a model that does not weigh these features highly would likely be unsuitable for the purposes of this project. The weight vector brain visualizations for both datasets represent these expectations, helping validate their usefulness. In both the over and imagined datasets, there is a clear yellow/green mass near the back of the brain. This indicates that the SVM models prioritize the electrodes placed at the back of the head, near the cerebellum, in making a classification decision. Additionally, the stem plot of weight values for both datasets show spatial patterns. Similarly indexed electrodes seem to share continuous values with their neighbors. Many of the regions are 0 or very close as well. This further indicates that the model prioritizes the weight of few electrodes, and further illuminates that these selected electrodes are spatially correlated. This is reassuring, as this is a rational approach to the problem given the underlying neuroscience. The visualizations also provide clues about the relative model performances. The yellow regions on the Imagined dataset are less yellow and have lower magnitude weights than those of the overt data weights visualization. This indicates that assumptions regarding the noisiness of the imagine dataset are correct, as the weights generated by the SVM model less strongly prioritize the relevant brain regions.

Differing Regularization Parameter

The regularization parameter varies in magnitude when comparison is made between the testing folds of the Overt dataset and the testing folds of the Imagined dataset. The regularization parameter determines the degree to which the size of the coefficients will be weighed against the classification loss of the model. Thus, a higher regularization parameter value will mean that the training will focus more on the regularization term in minimizing loss. This means that the resulting model will prioritize a higher margin if the regularization parameter is higher, as a smaller slope for the model's weight coefficients will represent a flatter boundary. Flatter boundaries will inherently have a higher margin. So, looking at the degree of the regularization parameter will allow the margin of the underlying model to be uncovered. The average regularization parameter values are roughly the same on average, but the Overt model has more variance. This means some folds take on much larger values than that of the Imagined dataset. The plot illustrating this is logarithmically scaled, so the difference of 4 grid units between the largest parameter values in the Overt dataset and the mean for the Imagined dataset represents a factor of 10^4 difference in magnitude. Thus, the difference in classifier margin between the two is significant. This is to be expected, as the data's context would imply that the Overt dataset be more separable than the Imagined one. The Overt dataset is less noisy, indicative of the higher fold-level performance throughout than the Imagined dataset. So, it makes sense that it is able to accommodate a higher degree of regularization than the Imagined one.

Performance Trends in Cross-Tested Models

The imagined trained model performed better on the overt testing dataset than the overt trained model performed on the imagined testing dataset. This is due to underlying differences in the training datasets. As previously discussed, the imagined data set has more variance and noise than the overt dataset. So, it would be easier to classify the overt dataset accurately than the imagined dataset. The model trained on the overt dataset was not exposed to high enough variance to adequately prepare for the imagined testing set. It generated too high a margin on the overt dataset to generalize to the more variable imagined data. The model trained on the Imagined data was of lower margin, and less accurate in cross-fold accuracy analyses. But, due to the data sequence this is better for this specific problem. Context of the data informs this analysis, as without knowing that the Imagined dataset is more noisy than the Overt one, it may be confusing that the higher margin classifier generalized worse. But, the Overt classifier was not trained on the worst-case data. So, it is unable to perform as well when it does encounter it. This leads to an inflated margin that does not represent the holistic classification task if we wish to perform optimally on both kinds of data. This yields the lesson that it is necessary to ensure the rigor of the data we train a model on. It is important to make the training data difficult and representative of the most challenging corner cases that the model may encounter in deployment. This will generate as large a margin as possible while still preventing over-bias towards specific cases in the training set. To generate a fully optimal model for this project, implementing a model that trains on both the Overt and Imagined datasets would encapsulate the full range of variance expected in deployment, while avoiding overfitting too high a margin (overt) or overfitting to the specific outlying datapoints (imagined). This indicates that in designing a classifier it is important to ensure that the training data contains an equal proportion of noise and signal as is expected in real-world deployment. This will ensure that the model provides an optimal margin for the exact task we wish to deploy it in, versus underfitting or overfitting on too easy or too noisy a training set.

Collaborators

I compared final results with Namh Lahade, Aarzu Gupta,Mike Khabib, and Monty Truitt. This included comparing the magnitude of alpha values across both datasets. I was initially dubious regarding the correctness of the results I generated, so this comparison allowed me to be confident that the model I built was correct. I ultimately was informed by Mike that I had not considered a large enough range of alpha values given the decision function for the scikit learn model I chose to use. The same is true for my final ROC curves, as I compared the complete ROC curve for each dataset with these people. In writing the code to generate the complete ROC curve for each dataset, I consulted Namh to be sure that the correct procedure was to combine the list of decision statistics for each fold. I discussed with Namh and Aarzu on how to generate a table in matplotlib in order to easily display the top 6 electrode weights for each model. I discussed the contextual/domain information prompts with Namh and Aarzu to determine what was being asked of me and to uncover that viewing the average data point from each class's brain scan would be helpful in piecing together a response.

Citations

Packages

Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
- Used Scikit-learn package to implement SVM Classifier (LinearSVC Object) and Kfold Cross Validation

"Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp.
- Used Matplotlib package to generate all visuals

McKinney, W., & others. (2011). pandas: a foundational Python library for data analysis and statistics.
-Used Pandas package to read, store, and manipulate electrode data as DataFrames

Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. Nature 585, 357–362 (2020).
- Used Numpy package to store information as manipulable arrays and access mathematical operators for entire arrays

Jones, E., Oliphant, T., Peterson, P., et al. (2020). SciPy: Open source scientific tools for Python.
- Used Scipy package to implement Gaussian KDE and support showChanWeights function

Other

National Center for Biotechnology Information. (2000). The Neural Basis of Motor Control. In Neuroscience (2nd ed., Purves D, Augustine GJ, Fitzpatrick D, et al., editors). Sunderland (MA): Sinauer Associates. Available from: https://www.ncbi.nlm.nih.gov/books/NBK11024/#:~:text=The%20primary%20function%20of%20the,neurons%2C%20to%20reduce%20the%20error.

Anatomical Justice. (n.d.). Cross Section of the Brain - Ocular Level. Retrieved May 1, 2023, from https://anatomicaljustice.com/product/cross-section-of-the-brain-ocular-level/.