Isaac Whittaker
Jacob Braswell
Tyki Wada

Background

The EEG (Electroencephalogram) monitors brain wave activity to sense abnormalities such as brain disease, lesions, and other electrical activity occurring in the brain (ie. trauma, body movement, drug intoxication, brain damage). This test is administered by attaching electrodes with wires onto the scalp, detecting tiny electrical charges emitted to brain cells. Data is then collected and visualized on a screen for analysis.

Data and Task

This analysis of the EEG Eye State file consists of data that describes attributes of the EEG test corresponding to whether the eye is detected as opened or closed. From the initial exploratory data analysis, EEG attributes were described to all have similar means and quartiles with varying degrees of variance (34 ~ 5800). Eye detection followed a binomial distribution with 44% detection.

Several classification models were evaluated to determine optimum results: Neural Networks, Logistic Regression, Random Forest, and K-Nearest Neighbors. All models were split into train and test sets of 70-30 using a seed of 40 as to ensure consistency in comparison. These models were chosen out of experiment and out of a desire to learn in Python. The initial assumption surmised the neural networks to have the best classification performance as it has a large number of free parameters (allowing for deeper complexity) and its ability to learn these parameters. However, after careful consideration of the data, interpretation, accuracy, sensitivity/specificity, out of bag error, and area under the curve metrics, it was decided that XGBoost would be the best classifier.

Model Comparison

Neural Network:

The neural network was created using python's sci-kit learn. The data was first processed into 90-10 train and test, then standardized by a scalar. Parameters were learned after running 3 layers and 12 hidden nodes.

Our results indicate an accuracy of 51.05%. This suggests fewer than 49% accuracy when predicting values out of our trained data.

From the confusion matrix we observe specificity to be 83.32%, indicating the ability to predict eye state when it is not open. Sensitivity, on the other hand, was found to be 11.42% suggesting the model's capacity to predict whether the eye was open when it truly was. As the aim is to maximize eye detection when true, a sensitivity score of 11% is not be the optimum model for

this task. Overall, this model detected when the eye was closed better than when the eye was open.

Another metric that is used to measure the performance of this model is the area under the curve: 47.38%. This is slightly worse than a simple coin flip and the neural network model does not perform as well as other models for this particular analysis. This may be due to the simple composition of the network with 3 layers and 12 hidden nodes. For this model to determine utility, a more complex model (ie. RNN) may need to be considered.

Logistic Regression:

This model was evaluated using statsmodels and sci-kit learn in python. The data was first split 70-30 into training and test data, then run on both for simple evaluation.

The overall results of this model display an accuracy of 62.86%, sensitivity of 47.08%, and specificity of 75.73%. This model outperformed the neural network accuracy by over 20%. And specified the model's capacity to predict when the eye is not open when it truly was not, by less than 36%. However, the model's ability (sensitivity) to detect whether the eye was open when true, well exceeded that of the neural network.

Though better, these results still suggest the model not to be practical for this specific test. Although logistic regression serves as a great model to predict binary predictors (0 and 1), the accuracy of this test and area under the curve indicate moderate predictive capability. Some aspects that may improve the model performance may

The model may be further improved in two ways. 1) Setting cutoff values for logistic regression to categorize eye movement may better capture precise results. However, with a mean value of eyeDetection around .5, varying the cutoff value may not change the model performance in this specific case. 2) Using reported p-values after fitting the model to determine statistically significant effects may also yield better results. From our model, the least 'important' indicators (beta values) seem to be O2, O1, and F8 as they seem to be insignificant on the effect of eye detection.

Random Forest:

Parameters for the Random Forest were run using a five-fold grid-search cross validation method in python's sci-kit learn. Best values of n_estimators and max_depth were, respectively, 100 and 50.

The Random Forest displayed the following metrics after a grid-search cross validation was run: Accuracy, 93.66%; Sensitivity, 94.07%; Specificity, 95.05%. Outperforming the neural network

and logistic regressions, this model had an acutely high accuracy meeting, most likely due to the high sensitivity and specificity values.

Furthermore, an out of bag score was also measured as an alternative method to the grid-search cross validation and was found to be near 93%. This suggests 7% and greater of the predicted values in the “out of sample” data (⅓ of the unused data in creation of the trees), is accurate. Since our values of .oob_score_ fall beyond 50%, our ensemble outperforms a simple coin flip by 43%.

XGBoost:

Parameters for XGBoost were also run similarly to the random forest model using a five-fold grid-search cross validation in sci-kit learn with a 70-30 train-test split. The best n_estimators were discovered to be 250 and max_depth, 15.

This boosting method output the following metrics: Accuracy, 94.17%; Sensitivity, 94.25%; Specificity, 95.39%. It is also important to note the higher sensitivity in detecting eye state when the eye is truly open. These scores are similar, although seem to, slightly (~1%), outperform the random forest model. This performance increase may be due to the data having a low variance across other indicators and a potential bias, allowing the boosting process to aggregate the output from its incremental model-building process.

K-Nearest Neighbors:

The KNN model was built under sci-kit learn in python. Best parameters were evaluated using a grid-search cross validation after splitting the data into train and test sets. The best parameters utilized a uniform euclidean distance of 1 neighbor.

Overall, the performance of the neighbors outshined that of all other models: Accuracy, 96.78%; Sensitivity, 96.66%; Specificity, 96.69%. With one neighbor, we notice that there are two overall groups of indicators that influence the detection of eye state. However, merely identifying these two groups does not explain the most important indicators.

It is recognized that the data provided is not labeled, hence the usage of the K-nearest neighbors may not have been appropriate as compared to clustering methods such as K-means, or PAM. However, these models do not justify the absence of interpretability of our eye-state data, especially when considering these identifiers to be computer generated.

Best Model

In selecting the best model, three key metrics were considered. Accuracy, sensitivity, and interpretability. These metrics were chosen as the goal to measure the accuracy of prediction overall, accuracy of eye state when eye was truly opened, and for the sake of relating our data

to our model. The highest scores of accuracy and sensitivity, in descending order, were produced by the K-nearest neighbors, XGBoost, and Random Forest models. Model interpretability was mainly a factor of XGBoost and Random Forest.

Although the KNN model had the maximum ability to detect eye state when the eye was both open and closed, the ability to detect which indicators had the most impact was constrained. Furthermore, with average accuracy scores from the decision trees straying from the neighbors classifier by only 1.8 standard deviations, it seems safe to assume similarity in model performance.

Consequently, XGBoost was selected as the best classifier for this specific data task due to the optimum balance between accuracy (94.17%), sensitivity (94.25%), and interpretability (95.39%). The five ‘most important’ indicators were identified as: P7, O1, F7, F8, and AF4. These indicators are most important as they are the most closely related with the dependent variable and contribute the most variation. For further analysis, p-values may be compared with a benchmark score to parse features that prove no effect to the benchmark, then retrained.

Summary

The goal of this analysis was to predict whether the eye was open or closed, interpreting various indicators received by nodes around the brain. Various models were run on the data for optimal performance. For the analysis the data was split randomly into two sections, one being the training set --what the model was fit on-- and the test set--what the trained model was tested on to check accuracy and the other metrics we used. Accuracy, Specificity, Sensitivity and area under the curve were used as metrics to measure model performance. Cross validation and feature selection was also used to determine values significant in predicting and detecting eye state. The model chosen was selected based on optimal interpretability and relevant metrics, in particular Sensitivity and accuracy. From these criterium, XG Boost was selected as the best model for the problem.

Several other considerations may be useful in improving model results. One in particular may have been to utilize a feature selection method through a lasso regression method, pulling the most ‘important’ features. Another is to improve the complex neural network to better learn the data. Finally, better data collection or experimentation with the train and test split ration may improve the accuracy of our models.