**Contest Question (1)**: Optimize the accuracy of seizure detection (assumed species-specific).

**My own Question of Interest:** How well can a dog-trained model detect seizures in humans?

**Data Handling:** All data handling was performed in R, and was, ultimately, a (fruitful and educational!) several week endeavor for me. I began by devising a function to 1) read in all clips one-by-one from a single directory; 2) convert them from MatLab to data.table format; 3) and use string-splitting of the clip’s filename to create columns for type (dog vs. human), numeric channel, id, seizure state, segment number, and, in humans only, EKG channel location. I tested it on a few files, and estimated, with parallel processing, importing all clips would run 4.5-7 minutes in total. I left this process running overnight on my 4 core, 16GB MacBook overnight, to find it had run for six hours before eating all of my remaining hard drive space. I spent a few days attempting to access Amazon Web Services, but could not make it work in time to deploy for this assignment. I tried again (locally) on a smaller subset of the data (just dogs), to find that running the function made my computer completely useless (oddly, Activity Monitor indicated 60 GB of RAM was in use). With only 16 GB of hardwired RAM to spare, I decided to reduce my data sample to a random subset of all Ictal and Interictal clips that would be no larger than 10^4 MB. I experimented with sample sizes, ultimately landing on a 2% random subset of all clips that, when read into ram, was approximately 11 GB and when run through the function took approximately 250 seconds to process, 100 seconds to write to RDS for later fast access, and, during later access, 35 seconds to read into ram from RDS.

**Modeling and Prediction:**

I experimented with my own feature selection using Generalized Linear Models and some information gleaned from the seizure literature, but this was computationally slow and fairly frustrating. So, I turned to SuperLearner (SL), an ensemble, targeted learning package. After a few hours of experimentation with the program’s syntax, I was able to test the utility of this package on a small, randomly-selected subset of data: on 10% of my Dog data, I employed internal and external 10-fold cross-validated targeted, supervised prediction with the SuperLearner Random Forests and Neural Network learning libraries with a Nelder-Mead AUC loss function. The following variables were included: ‘Data’ (numeric), ‘channel’ (numeric), Dog ID (numeric), segment number (numeric), and seizure state. (Note: conversion of all factors to numerics struck me as counter-productive, but is required by SuperLearner). With the parallel processing function enabled, this first run took a total of 3193.67 seconds (~ 53 minutes). A summary of the SL object indicated complete exclusion of the Neural Network coefficeints (i.e. weight = 0) and reliance only on the Random Forest library. It produced an AUC of 0.9962 (Figure 1). While this method produced an almost unbelievably high AUC, it was slow and, because I ran a “CV.SuperLearner” instead of a “SuperLearner” function, I could not use the SL object to predict seizure outcomes on the rest of my data. So I reran the same SuperLearner code, but with internal cross-validation only, to produce an SL object that could be used for prediction. Nervous about timing, I stuck with the 10% subsample as my training set. ‘Elapsed time’ was much shorter without the cross-validation, at 661.7 seconds. I then applied the resulting SuperLearner object to the remaining 90% of the Dog data, not expecting much as the algorithm had only been trained on 10% of the sample. Very surprisingly, the AUC came in at 0.9967, with a plot not much different from Figure 1. While not intended to be the final dog model (as it was trained with only 10% of the data), I was very happy with the AUC and decided to use my remaining time to explore more interesting aspects of the dataset.

Satisfied with my SL prediction on Dogs, I pivoted to testing its predictive powers on human patients. To do this, I had to reshape the human data to fit the parameters used in dog testing: frequency of 500 mhz only with no consideration for placement. This dropped the human data to 136,000 data points, 25% of which were in seizure state (Table 1). I also had to rerun the Dog SuperLearner model in a way that grouped all dogs together (i.e. drop the variable for Dog ID). This took 712 seconds. For comparison, the AUC of this SL object on the other 90% of the dog data produced an AUC of 0.9967, the same as when the identity of each dog was taken into consideration. The prediction ability of this new dog model on human seizure outcomes was worse than flipping a coin (random), with an AUC of 0.488, suggesting that a model trained on and validated to work quite well in dogs may not adequately predict seizures in humans. It is important to note, however, that only one patient met the requirements for inclusion in the Dog model, likely because the original study was not designed to answer my specific question. Future examinations of this question should involve multiple patients.

I then turned to the human data for prediction (Table 1). I opted to use this step as an opportunity to explore (and learn how to use!) h2o’s deep-learning package in predicting seizure state via multi-layer, feedforward neural networks, with Tanh dropout regularization, 2 hidden layers of 200 nodes each, and 10 total epochs. Human patient data were handled the same way dog data were handled, with an additional variable created to indicate EKG node placement. The data were evenly split into a test set and a validation set. After 2 hours running on my MacBook, only 10% of my deep learning process had completed. So, I decided to try my advisor’s new MacPro server. It’s ‘new’ to us, but is actually 7 years old. Still, it boasts 16 cores and 16 GB of ram... and more importantly, using it for this process wouldn’t tie up my laptop for hours on end. After a few hours of tinkering, I was able to successfully install h2o (and its dependencies) via ssh, setup an h2o instance, and begin my modelling (elapsed time: ~ 3 hours). The number of epochs was dropped from 10 to 4 for timing considerations. ROC AUC calculated on a random sample of 10,000 of the predicted data points came out to **0.9229** (Figure 2)-- I attempted to calculate ROC AUC on the full data set, but after 6 hours of processing, halted the operation.

I chose to compare the winning submission of [Michael Hills](https://github.com/MichaelHills/seizure-detection/blob/master/seizure-detection.pdf) to my human-prediction set. Instead of deep learning, he actively selected model features using Fast Fourier Transform (via log10) to each 1 second clip in the range of 1-47Hz and across all EEG channels. The range 1-47Hz was chosen through trial-and-error based dimension reduction, and FFT was chosen simply on a hunch, but later verified by a high ROC AUC. FFT values were then normalized by frequency. “Real” eigenvalues were then calculated on the correlation coefficients of these normalized FFT values. Then, after experimentation with logistic regression, decision trees, and SVM, he settled on Random Forest with 3000 trees for classification and, like me, chose cross validation with an ultimate split of 50%. His best AUC was substantially greater than mine at 0.9675.

|  |  |  |  |
| --- | --- | --- | --- |
| Table 1. Description of Data Subsets Employed | | | |
| **Data Subset** | **n (rows)** | **# Individuals Represented** | **% Data Representing Seizure State** |
| Dog Data | 1,280,000 | 4 Dogs | 10.50% |
| Patient Data | 57,071,000 | 8 Humans | 10.67% |
| Patient Data Reshaped for Dog Model | 136,000 | 1 Human | 25.00% |



**Figure 1.** AUC and summary information for the first CV.SuperLearner run on Dog data.

 

**Figure 2.** Left, AUC from using the Dog SuperLearner model to predict human seizure states (0.4880). Right, AUC from applying the h2o Deep Learning model to the validation set (AUC = 0.9229).