Project Report

EEG State Discrimination Using Machine Learning

Bar Lehmann
Machine Learning Student
bar.lehmann@gmail.com

BIOF 509 ,
Dr. James Anibal

October 30, 2020

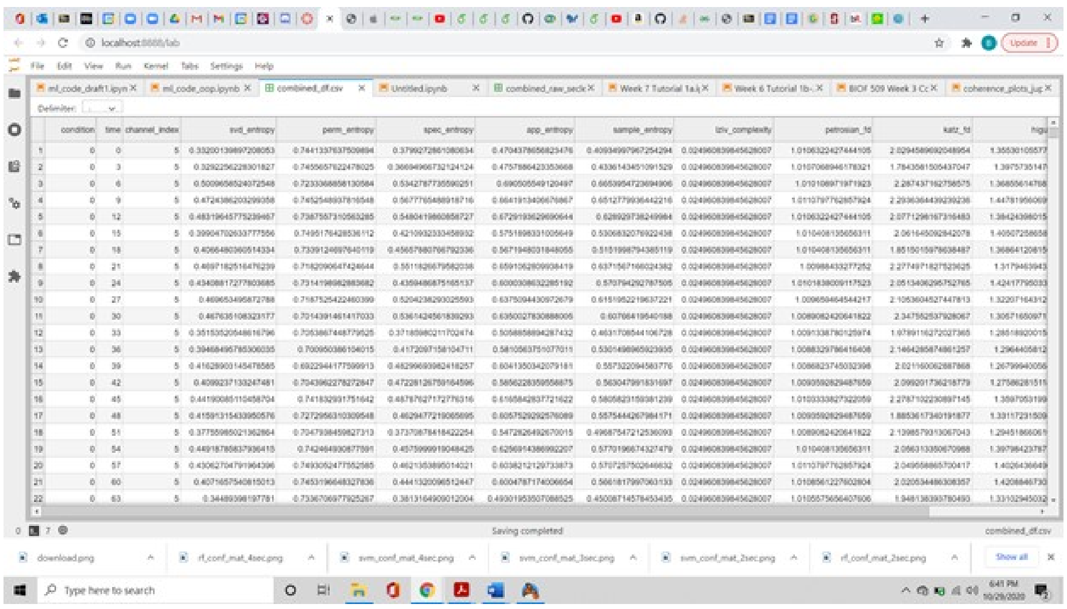
Evaluation Aims

This evaluation's overall aim is to evaluate the EEG State Discrimination tool's performance in relation to its results on the dataset used in this evaluation. This particular dataset evaluation serves as a "control" to future projects in that this dataset is simulating a random (or at least, close to random) allocation of condition to the samples of the dataset. In other words this evaluation is using a dataset containing only one state that has been partitioned into the first and second half labeled as state 1 and state 2 respectively. The goal is to assess whether two states that are intended to not be different from one another, are actually found to not be different through the test of machine learning algorithms. While a lack of predictability is expected since the condition of recording is no different in the first half than the second half, it is important to do such an evaluation to ensure that later evaluations do not run into confounds which might make the second half different from the first.

This tool will extract a set of EEG metrics (i.e. the columns/features) from the raw data, and subsequently the tool allows one to run machine learning over these features. This is a preliminary evaluation, and later evaluations will be needed for a more comprehensive assay of this tool. This evaluation thus aims to begin to establish a baseline of predictive ability between a first and second half of a neutral eyes-closed EEG recording. It is important to assess this since differences in the states (the first vs. the second half of the recording) can still exist as result of confounding variables such as artifacts of the EEG, or other possible relationships related to the EEG signal that create unintended differences between the first and second half of the recording. This evaluation is a test of both the tool itself and the dataset, both of which are shedding light on each other. In shedding light on the analysis of this particular dataset, the functioning of the tool itself and areas that it can be improved upon in further revisions of the tool are able to be discovered too. Using this approach of analyzing a "control" dataset was chosen partly due to emergent difficulties obtaining the datasets originally discussed in the project proposal as well as due to the availability of datasets that would fit to the framework for this tool- i.e. a tool that splits data into two halves down a middle point. Though this restriction is certainly not arbitrary, it is intentionally chosen and relates to the particular datasets it is meant to process in the future.

Data

This performance evaluation of the EEG State Discrimination tool uses raw amplitude time series data from a 19 channel EEG dataset of a closed-eyes recording of 6 minutes, sampled at 256 Hz of neutral/baseline state. The raw data is in European Data Format (.EDF). The dataset used is created from this raw data using the tools 'preprocess raw' function that performs an initial feature extraction on the data. This function can be set to include all channels, but by default includes only 7 channels of interest (by default it is set to those channels least liable to artifact / noise that may contaminate the data) which is a highly relevant and also easier dataset to analyze. The features it uses include 10 different complexity measures (i.e. Higuchi Fractal Dimension, Sample Entropy, Spectral Entropy, etc.), as well as the channel index number. These features are used since they are newer and less researched features that have still shown themselves to be correlated with a range of neurological disorders, mental health conditions, and states of consciousness such as anesthesia, sleep, or being awake. Furthermore, while there are many software tools used to analyze traditional FFT wavebands of the EEG, there are not many tools available for citizen scientists to do more research on them. Each sample is using 768 datapoints (3 seconds of data) from the raw file as input to the entropy functions. The first half of each selected channels' recording is dubbed state = '0', and the second as state = '1' despite that there has been no intended state change (nor any other known change that might impact the data) throughout the recording.



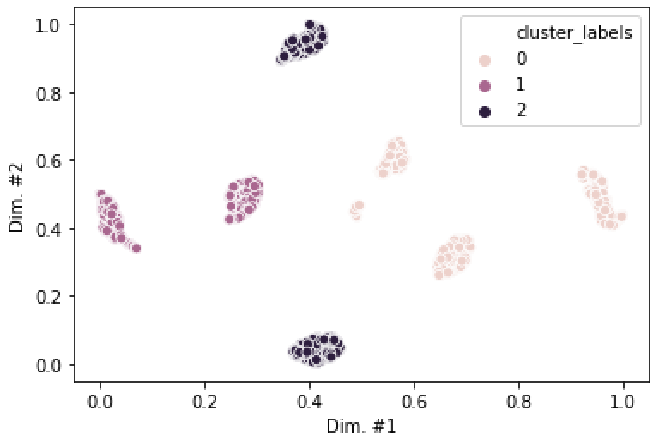
	condition	time	channel_index	svd_entropy	perm_entropy	spec_entropy	app_entropy	sample_entropy	lziv_complexity	petrosian_fd	katz_fd	higu
1	0	0	5	0.33200139897208053	0.7441337637509694	0.3799272861080634	0.4704378656825476	0.40934997967254294	0.024960839845628007	1.0106322427444105	2.0294589692048954	1.35530105577
2	0	3	5	0.3292256228301827	0.7455657622478025	0.36694966732124124	0.4757886423353668	0.4336143451091529	0.024960839845628007	1.0107068946178321	1.7843581505437047	1.3975735147
3	0	6	5	0.5009658524072548	0.7233368858130584	0.5342787735590251	0.690505549120497	0.6653954723694906	0.024960839845628007	1.010108971971923	2.287437162758575	1.36855614768
4	0	9	5	0.4724386203299358	0.7452548937816548	0.5677765488918716	0.6641913406676867	0.6512779936442216	0.024960839845628007	1.0110797762857924	2.2936364439239236	1.44781956069
5	0	12	5	0.48319645775239467	0.7387557310563285	0.5480419860858727	0.6729193629690644	0.628929738249984	0.024960839845628007	1.0106322427444105	2.0771298167316483	1.38424398015
6	0	15	5	0.39904702633777556	0.7495176428536112	0.4210932333456932	0.5751898331005649	0.5306832076922438	0.024960839845628007	1.010408135656311	2.061645092942078	1.40507258658
7	0	18	5	0.4066480360514334	0.7339124697640119	0.4565798076792336	0.5671948031848055	0.5151998794385119	0.024960839845628007	1.010408135656311	1.8515015978638487	1.36864120815
8	0	21	5	0.4097182516476239	0.7182090647424644	0.5511826679582038	0.6591062809938419	0.6371567166024362	0.024960839845628007	1.00988433277252	2.2774971827523625	1.3179463943
9	0	24	5	0.43408817277803685	0.7314198982883682	0.4359486875165137	0.6000308632285192	0.570794292787505	0.024960839845628007	1.0101838009117523	2.0513406295752765	1.42417795033
10	0	27	5	0.469653495872788	0.7187525422460399	0.5204238293025593	0.6375094430972679	0.6151952219637221	0.024960839845628007	1.009659464544217	2.1053604527447813	1.32207164312
11	0	30	5	0.467635108323177	0.7014391461417033	0.5361424561838293	0.6350027830888006	0.60766419540188	0.024960839845628007	1.0089082420641822	2.347562537928067	1.30671650971
12	0	33	5	0.35153520548616796	0.7053867448779525	0.371865980211702474	0.5058858894287432	0.4631708544106728	0.024960839845628007	1.0091338780125974	1.9799116272027365	1.28518920015
13	0	36	5	0.39468495785306035	0.700950386104015	0.4172097158104711	0.5810563751077011	0.5301498965923935	0.024960839845628007	1.0088329786416408	2.1464285874861257	1.2964405812
14	0	39	5	0.41628900145478585	0.6922944177599913	0.48299693982418257	0.6041350342079181	0.557322094560776	0.024960839845628007	1.0086823745032398	2.021160062887868	1.26799940056
15	0	42	5	0.4099237133247481	0.7043962278272847	0.47228126759164596	0.5856228359556875	0.563047991831697	0.024960839845628007	1.0093592829487659	2.099201736218779	1.27586281515
16	0	45	5	0.44190085110458704	0.741832931751642	0.46787627172776316	0.6165842837721622	0.5805823159381239	0.024960839845628007	1.010333827322059	2.2787102230897145	1.3597053199
17	0	48	5	0.41591315433950576	0.7272956310309548	0.4629477219065695	0.6057529292576089	0.5575444267984171	0.024960839845628007	1.0093592829487659	1.8853617340191877	1.33117231509
18	0	51	5	0.37755985021362864	0.7047938459827313	0.37370878418422254	0.5472826492670015	0.49687547212536093	0.024960839845628007	1.0089082420641822	2.1398579313067043	1.29451866061
19	0	54	5	0.44918785837936415	0.742464930877591	0.4575999919048425	0.6256914386992207	0.5770196674327479	0.024960839845628007	1.010408135656311	2.056313350670988	1.39796423787
20	0	57	5	0.43062704791964396	0.7493052477552585	0.4621353895014021	0.6038212129733873	0.5707257502646632	0.024960839845628007	1.0110797762857924	2.049558865700417	1.40264366649
21	0	60	5	0.4071657540815013	0.7453196648327836	0.4441320098512447	0.6004787174006654	0.5661817997060133	0.024960839845628007	1.0108561227602804	2.020534486308357	1.4208846730
22	0	63	5	0.34489398197781	0.7336706977925267	0.3813164909012004	0.49301953507088525	0.45008714578453435	0.024960839845628007	1.0105575656407606	1.948138393780493	1.33102945032

Feature Extraction

This tools 'first' class (that which is typically used first) serves to take in a raw EEG dataset and to extract a set of 10 entropy-related features. This tools 'preprocess raw' function appears to work well as can be judged by examining the pandas dataframe or the output as a CSV. There is a clear correspondence there to the specified channels, times, and other variables. For example, if 3 seconds is the specified desired sample to be created, and the sampling frequency is 256 Hz, then the total samples in the created spreadsheet will be 120 samples extracted from sets of 768 datapoints each. There will be 60 samples in condition 0 and another 60 samples in condition 1. This corresponds to there being 6 minutes of recording or 360 seconds. The CSV output has been examined for possible discrepancies with expected values and is consistent with expected values apart from Lempel-Ziv entropy which is a novel type of entropy that this specific evaluation is ignoring, despite that it is included within the features extracted.

Clustering

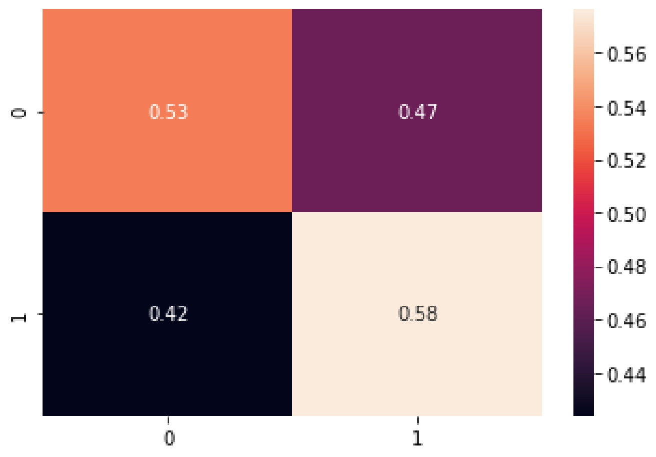
To get an initial visual to help us assay clusters in the data, we use agglomerative clustering. This method does away with the labels of our initial dataframe (channel indices, complexity values, etc.), and replaces them with two dimensions that help one to get an intuition for clusters that occur in the data. In order to dimensionally reduce the dataset from 11 labels to only 2, this tool uses the UMAP package. We perform scaling automatically before performing the UMAP dimensionality reduction in order to ensure that the different scaling of our initial columns do not skew the results. The names of the dimensions are only 'Dim. 1' and 'Dim. 2' due to the lack of specific information either dimension contains.

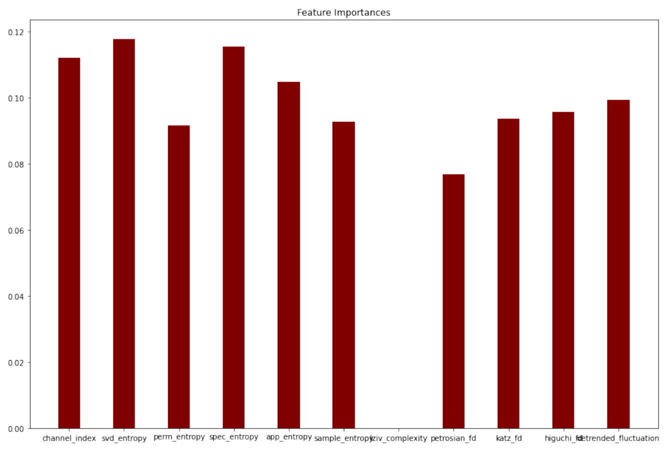


The clustering analysis shows 8 separate groups suggesting that there are about 8 separate and distinct groupings in our data. This is likely to be related to the fact that a few of the complexity features may be highly correlated with each other while other complexity features are likely to not be correlated with each other. After seeing the groupings, we continue our analysis with classification algorithms of Support Vector Machine and then Random Forest to see how accurately we can classify the putative "state 1" vs. "state 2" of the data.

Random Forest (RF)

The random forest method uses the random forest algorithm. It also splits the dataset into 5 parts for cross validation. This step ensures that we are not picking a biased subsample of our data (such as one containing a motion artifact) and training based on this. The splitting is done using the "Stratfied KFold" function of Scikit learn. This function splits the data at random intervals to further help bolster unbiased sampling. Each of these samples are subsequently classified into training and testing subsets. Iterations through the pairs of training and testing datasets will generate a feature importance score as well as an accuracy score which will each respectively be taken together to give a cross validated feature importance to each feature, and an overarching accuracy to the classification. In this method, a confusion matrix is automatically generated to easily portray the results and the accuracy score is provided as well.



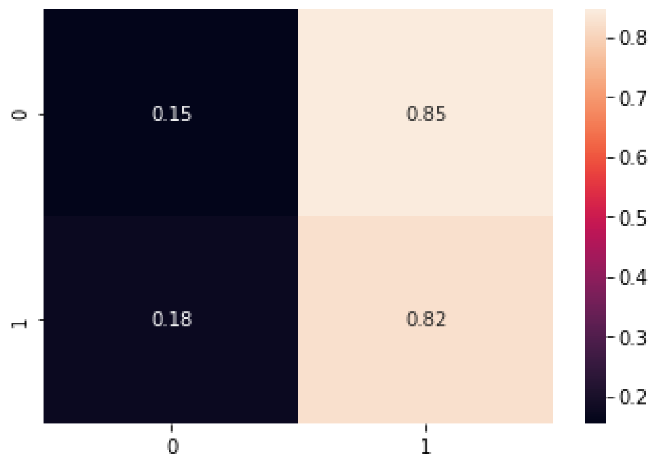


The accuracy score is consistently right around .55 which is clearly very similar to the results shown by the confusion table of .52 true negatives and .59 true positives. These results suggest that the predictive power of judging the first half from the negative is very close to random, though perhaps there does exist a small difference inherent in the halves themselves too. Running this random forest function repeatedly give very similar results. This suggests that, at least in this particular EEG recording, the hypothesis that the difference between the first and second half of the recording is indeed not representative of any highly significant established difference as captured by a range of 10 complexity measures. The importance of the slight difference is a subject for further evaluation. These results are consistent when the criterion used in the is ‘gini’ and the number of estimators used is 25 since these were recommended by doing a grid search for optimal functioning of Random Forest arguments for this dataset.

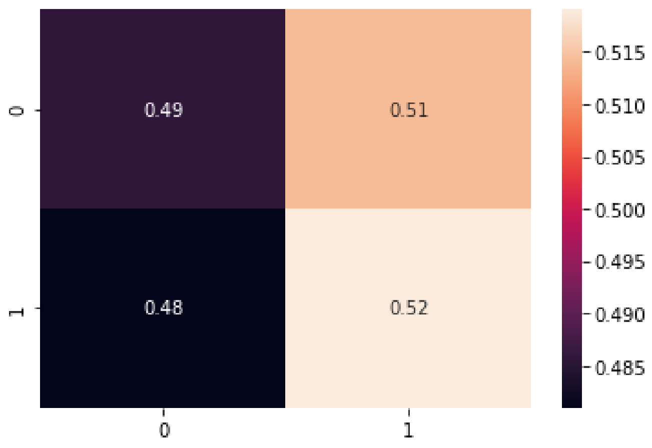
Support Vector Machine (SVM)

The SVM function is done very similarly to the Random Forest function in that this algorithm first splits the sample dataset into 5 parts as cross validation (each set is treated as test dataset. The splitting is similarly done using the “Stratfied KFold” function of Scikit learn splitting the data at random intervals to further help bolster unbiased sampling. Each of these samples are classified into training and testing subsets. Iterations through the pairs of training and testing datasets will generate an accuracy scores which will be taken together and averaged to give a cross validated accuracy score to the classification. A confusion matrix is automatically generated to easily portray the results and the accuracy score is provided as well.

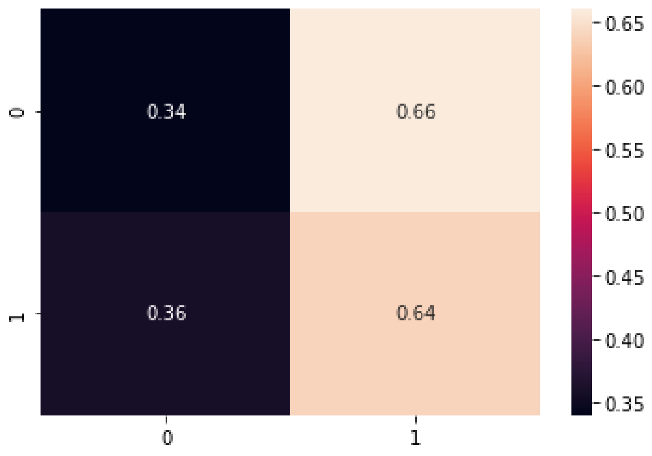
Interestingly, while the Random Forest classifier had all boxes of the confusion matrix at about .50 - no better than chance, the Support Vector Machine algorithm provides a very different result. At first it seems as though the SVM classifier supports the Random Forest results since the average accuracy score is .49. However the confusion matrix consistently shows that correct negatives (“state” = ’0’) are at .15, while correct positives negatives (“state” = ’1’) are at .85. This huge discrepancy suggests several possible explanations. It is possible that the classifier biased to some artifact (noise) that is present only in the latter half of the recording (corresponding to only “state”=’1’). If this noise is being used as a key component of the prediction, it may explain why there is such a contrast between the accuracy for predicting state ‘0’ vs. state ‘1’. It may also be the case that in state=’1’ the client has already settled more comfortably in their seat closing their eyes for a while, which might lead to increased alpha waves, alpha amplitudes, or perhaps other differences that make it more discernible, yet this quality might also be a red herring in the other half of the recording. It is certainly not yet clear why this discrepancy is so vivid with the SVM but not at all visible in the RF classifier. A grid search further confirms that even with the optimal hyper-parameters including gamma = ‘.1’ and C=100, there is no change in these results.



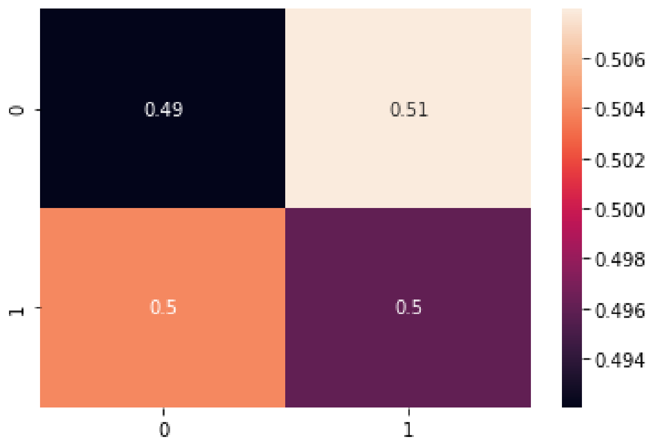
If the sample width (number of seconds in each new sample of the dataframe created from the raw data) is changed from 3 to 2 seconds. The confusion matrix for SVM classification looks like this:



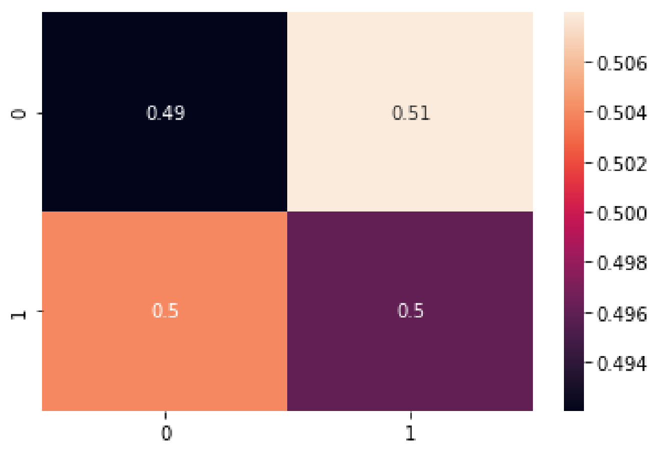
If it is changed to 4 seconds, the confusion matrix for SVM classification looks like this:



And at 5 seconds the confusion matrix for SVM classification looks like:



Meanwhile, the confusion matrix for random forest classification remains at .50 for each box between 2-10 seconds.



Again, it is unclear why it is that there is such an unusual discrepancy between the SVM vs. the RF confusion matrices around 3 seconds, or why this discrepancy is reduced at two seconds, or above three seconds. Other studies have taken complexity measures at 5 seconds sampled at 128 Hz, so it may be that less than 5 seconds may not be an optimal setting for the sample duration of complexity features. Alternatively, 3 seconds and the SVM algorithm might be especially suitable for certain predictions of the second half of a recording for some yet unknown reason. While it appears that the sample-duration settings (i.e. 3-5 seconds) are close to those used in other EEG complexity metrics studies, it will be important to do further research about different complexity metrics' optimal settings for amount of samples and seconds for improving this tool.

Scaling

The MinMax scaling function clearly works and has been tested. Since the values are close to being scaled and there were problems using the scaled data with the Support Vector Machine algorithm at this point, the use of scaling with SVM and Random Forest will need to wait for a later edition of this project. However, it is possible to print this dataframe and see that indeed the scaling occurs successfully.

Next Steps

Overall the tool appears to be functioning properly in its feature extraction and its ability to operate on the different functions present. The main discrepancy is between the Random Forest and the SVM classifiers, in which the confusion matrices are very different around 3 seconds, though less so below and above 3 seconds. It will be key to evaluate the reason for this difference in order to proceed with further evaluation. It seems most likely that the issue has most to do with an optimal sampling number for each complexity measure, but there is also a small possibility that this might also have to do with bugs in the code.