Classification of Alcoholism

Group 124 (CS)
Nathan Bui, Ryan Bui

Introduction and Motivation

- Our project attempts to classify alcoholics through machine learning, utilizing previously recorded EEG data.
- Our motivation is derived from the fact that alcoholism is a major problem within our society. Therefore, we wanted to determine how severely alcoholism affects one's mental states, and from this be able to identify those who are suffering from this disorder.

Related Work

- In regards to class content, our project relates to topics discussed within the BCI Review. To be specific, our project is classification, so it was necessary for us to use feature selection methods to "maximize our performance". Specifically, we analyzed the importance of each sensor, or feature, in regards to our results.
- Outside of class, there is BCI-based study that is very similar to our own research. In this study, they recorded EEG signals and analyzed the differences between drunk and non-drunk subjects.

Data Explanation

- Data
    - From a large study to examine EEG correlation from genetic predisposition to alcoholism
    - Data was recorded with 64 electrodes on subjects scalp at standard sites for a period of 1 second
    - Two different groups: alcoholic and control
    - Subjects were given three different kinds of stimuli where they were shown either one picture, two identical pictures, or two different pictures
    - During this study, there were 122 subjects and each subject complete 120 trials
    - Because the data set was very large, we focused on the data where the subject was shown only one picture and limited the number of files used

- Citation:
    - UCI Machine Learning Repository
    - Link: https://archive.ics.uci.edu/ml/datasets/eeg+database
    - Owner: Henri Begleiter, Neurodynamics Laboratory, State University of New York

Methods

- Implementation
  - Used Anaconda as our main library and package manager
  - Jupyter Notebook was our coding platform
  - The main packages we used to develop our model was sklearn and pandas
  - For data analysis, we used numpy, seaborn, and matplotlib
- Github :

Cleaned up dataset

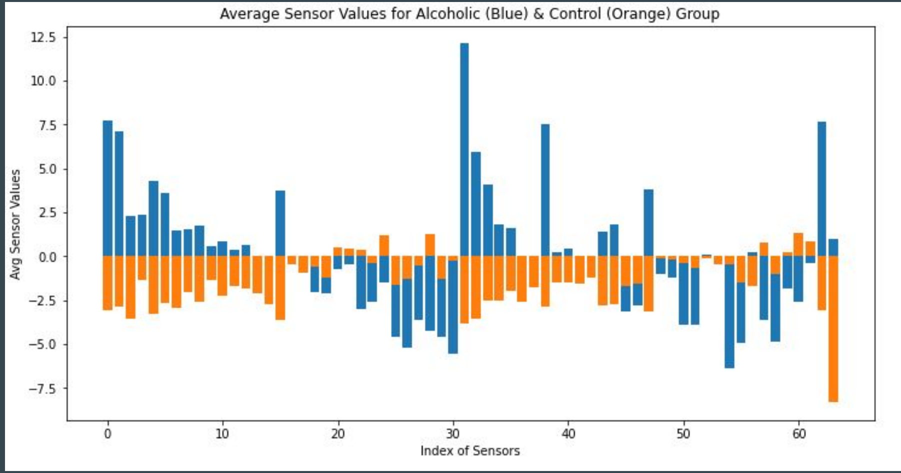
Removed specific features of the original data and only used sensor position, subject identifier, and sensor value to create our models.

	Unnamed: 0	trial number	sensor position	sample num	sensor value	subject identifier	matching condition	channel	name	time
0	5	30	FP1	0	-3.550	a	S1 obj	0	co2a0000364	0.000000
1	6	30	FP1	1	-5.015	a	S1 obj	0	co2a0000364	0.003906
2	7	30	FP1	2	-5.503	a	S1 obj	0	co2a0000364	0.007812

	FP1	FP2	F7	F8	AF1	AF2	FZ	F4	F3	FC6	...	PO7	PO8	FCZ	POZ	OZ	P2	P1	CPZ	nd
0	-5.91	3.194	-5.534	12.278	-6.907	-6.917	-7.314	-2.553	-2.004	2.594	...	11.485	6.846	-3.489	4.364	2.024	3.845	3.062	2.675	-5.025
1	-9.705	-11.444	-4.201	-10.447	-4.415	-5.29	-0.193	-1.373	-1.261	-10.315	...	5.442	-3.377	1.79	-2.116	-4.15	-3.886	-2.207	-1.17	-9.552
2	-10.468	-13.204	-10.803	-12.553	-7.141	-7.334	-4.496	-7.233	-3.499	-7.568	...	4.751	-7.192	-2.085	-2.797	-5.157	-0.468	-0.285	1.394	-9.623
3	1.465	0.488	0.427	-6.989	-1.048	-4.395	-3.184	-7.65	-3.621	-7.599	...	-9.247	-12.258	-1.414	-2.075	-4.598	-1.648	0.671	1.597	1.577
4	6.632	9.44	8.575	-9.644	1.241	-1.18	-0.753	-0.712	2.411	-1.302	...	4.924	0.905	-0.956	1.811	0.783	1.719	2.543	1.475	6.592
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

Exploratory data analysis

- After cleaning up the dataset, we dived into the data to develop a better understanding, as well as seeing if there was any interesting correlations.
- The main graph (shown below) we generated compared the average sensor values of each sensor, between the control group and the alcoholic group.



Training and Test Data

- 80/20 Split

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

- Data was shuffled before the split and the data was separated based on different time points

Classifiers

1. Random Forest Classifier
    a. Performs both regression and classification task
    b. Good predictions and handles large datasets
    c. Check Feature Importance
2. SVC Classifier
    a. Memory efficient
    b. Effective in high dimensional spaces
3. LDA Transform into Random Forest Classifier
    a. Dimension reduction while keeping the original level of information

Random Forest Classification

```python
# Create model and check accuracy and precision of predictions for Random Forest Classification
rf_model = RandomForestClassifier(n_estimators = 1000)
rf_model.fit(X_train, y_train)
target_pred = rf_model.predict(X_test)
print("Random Forest Classifier Accuracy:", metrics.accuracy_score(y_test, target_pred))
print("Random Forest Classifier Precision:", metrics.precision_score(y_test, target_pred, pos_label = 'a'))
```

SVC Classification

```python
# Create model and check accuracy and precision of predictions for SVC Classification
svm_model = svm.SVC(kernel = 'linear')
svm_model.fit(X_train, y_train)
y_pred = svm_model.predict(X_test)
print("SVC Accuracy:", metrics.accuracy_score(y_test, y_pred))
print("SVC Precision:", metrics.precision_score(y_test, y_pred, pos_label="a"))
```

LDA Transform to Random Forest Classification

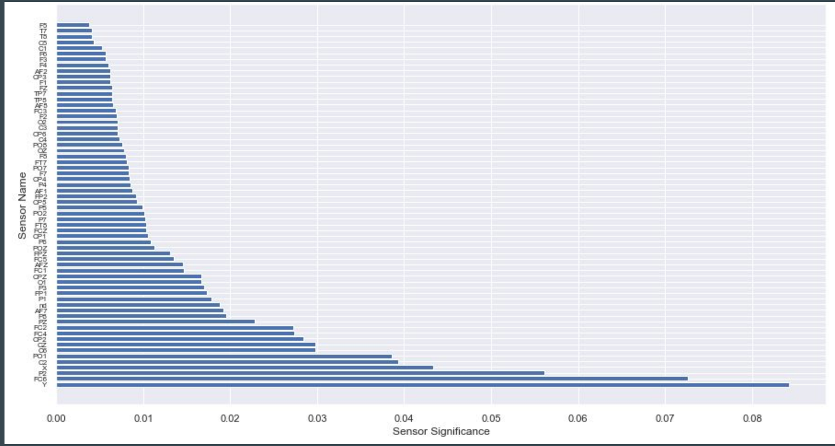
```python
# Create model and check accuracy and precision of predictions for LDA Transform to Random Forest Classification
LDA_model = LinearDiscriminantAnalysis()
X_LDA_train = LDA_model.fit_transform(X_train, y_train)
X_LDA_test = LDA_model.transform(X_test)
rf_LDA_model = RandomForestClassifier(n_estimators = 1000)
rf_LDA_model.fit(X_LDA_train, y_train)
y_pred = rf_LDA_model.predict(X_LDA_test)
print("LDA Accuracy:", metrics.accuracy_score(y_test, y_pred))
print("LDA Precision:", metrics.precision_score(y_test, y_pred, pos_label="a"))
```

Results: Sensor as Features

Classifier	Accuracy	Precision
Random Forest	0.9853515	0.9919517
SVC	0.9082031	0.9133064
LDA Transform	0.8457031	0.8251879

Top Ten Significant Sensors

Y	0.084287
FC6	0.072568
P2	0.056099
X	0.043360
C2	0.039297
PO1	0.038546
C6	0.029791
CZ	0.029744
CP2	0.028437
FC4	0.027384



Results: Top Ten Sensor as Features

Classifier	Accuracy	Precision
Random Forest	0.9677734	0.9700598
SVC	0.8291015	0.8097928
LDA Transform	0.7050781	0.6884328

Reorganized Data

Features based on 256 different time point in the span of 1 second

	T0	T1	T2	T3	T4	T5	T6	T7	T8	T9	...	T248	T249	T250	T251	T252	T253	T254	T255	Sens
0	-2.218	-1.241	-1.241	-1.729	-2.706	-2.218	-1.729	0.224	2.177	3.642	...	-11.007	-8.565	-6.612	-5.636	-4.659	-3.194	-2.706	-2.706	
1	0.814	-1.628	-3.092	-3.092	-1.139	0.814	2.767	4.232	5.697	6.673	...	26.693	24.74	22.786	21.322	20.833	20.345	21.322	21.81	
2	-6.337	-7.802	-6.826	-3.896	-0.478	1.475	2.452	2.94	3.428	4.893	...	6.358	5.87	6.846	9.288	11.729	13.682	15.147	15.147	
3	-3.479	-1.526	0.916	2.869	3.357	2.869	2.38	1.892	1.892	1.892	...	5.798	5.31	3.357	0.916	-0.549	0.427	2.869	5.31	
4	2.574	-0.356	-0.356	5.015	5.015	8.433	-1.821	10.386	1.597	3.062	...	-2.309	3.062	2.085	0.621	5.992	2.085	1.597	5.015	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	

Results: Time Points as Features

Classifier	Accuracy	Precision
Random Forest	0.90625	0.9083969
SVC	0.8007812	0.8076923
LDA Transform	0.7890625	0.7984496

Results: Final

- Sensor as Features produced the best models
- Random Forest Classifier showed the best accuracy and precision across all the models
- Correlation between alcoholic and EEG data
- What went wrong
  - Accuracy and Precision decreased as we attempted to manipulate the data
  - Trouble organizing the data to ensure that all factors were held accountable

Discussion

- What we learned
  - How to analyze, filter, and clean EEG datasets
  - Based on the accuracy and precision values we obtained, it can be assumed that there is a difference between alcoholic and non-alcoholic individuals when focused on the electrical activity of one's scalp
- Some improvements
  - For our classifier, we only used the data for one stimulus within the dataset. For the future, it may improve the performance of the model if we included more data
  - We could have used more, different classification models to compare/improve performance
    - Further understanding the ways we can use algorithms to understand the data
  - The time period being 1 seconds seems too short and should possibly be extended to account for more variability