FINAL PROJECT
Alvin Thomson

Introduction

The purpose of this project is to identify the key differences in the brain signals when presented with misleading and consistent stimuli. 25 participants were presented with stimuli that were consistent, misleading and indifferent while strapped to an EEG machine. The stimulus was presented for eight trials, and the peak values for neural components: p2, fn400, p3, and n2 were aggregated and stored. Along with the area under the curve for respective time series (150-275 ms, 150-350 ms, 250-400 ms, 300-500 ms). The following paper analyzes this data and draws conclusions based on the data.

Data Description

Based on this study there were nine datasets investigated. Four of these datasets represented four different neural ERP components of interest (FN400, P2, P3, N2). Each ERP component can be represented as a different portion of an EEG scan. These peaks can often be associated with different brain responses to stimuli. DIfferent ERP components manifest in different time intervals (P2: 150–275 ms, FN400: 300-500ms, P3: 250-400ms, N2: 150-350m). In addition to this, the five other datasets measured the area under the curve between each time interval. Each of these data points were measured by 32 electrode sites placed across the participants' scalp. The four peak datasets contained the peak voltage and latency (time from start) of the 32 electrodes at peak brain activity.
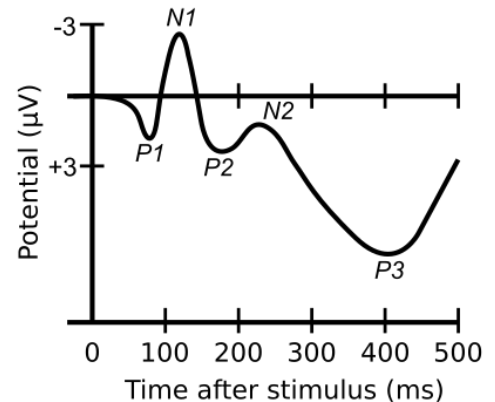
The grain on the dataset is represented by the peak values and/or AUC (area under curve) values for a participant in the study. Each dataset contains 24 entries for each participant who was studied in this experiment.

Data Preprocessing

Among the nine datasets csv files, two datasets (ALLTRIALS150275.txt and Peaks_P3) were formatted radically different from the others.

ALLTRIALS150275
Unlike the other csv datasets, this dataset used tabs as a delimiter. Unfortunately, the size of the delimiters varied (space between first column and second was 6 spaces, the space between second and third was 3 spaces, spaces between data points was 12). To make matters worse, some column names also had a single space between. The data also lacked new line symbols. This provided a unique challenge. To clean the data, I first manually removed any unnecessary

spaces between the few column names that had a space. Next, I utilized a regex[1] string to first identify spaces of variable length within the dataset and replace them with a comma. Next, I utilized a regex to add a new line symbol every 94th column (this step required some experimentation, 94 was the number of features for each of the alltrials datasets). In hindsight, I would of deleted the headers and replace them with the column names of another AllTrails dataset since they shared column headers.

PEAKS_P3

The peaks file was another unique challenge. There were several headers that did not have commas. This resulted in column headers that merged together and several data values without any column. This was slightly difficult to solve programmatically so I cleaned it manually (though you could do it with another regex). I solved this issue by looking for all the entries that ended with -V, and -L and manually added a comma to each entry that was missing a comma. Again in hindsight, it would have been much easier to replace the headers of this dataset with a clean one instead.

Data Exploration

This dataset was unlike anything I had ever explored. For each participant there were over 90 columns. Each cell contained either a latency or voltage value that corresponded to an electrode and stimulus type. The compactness of the data made it extremely difficult to comprehend trends or correlations. As a result I needed to separate and reformat every dataset into something more machine (and human) readable. A further explanation of the refactoring steps I performed will be provided below. The measures of spread for the final dataset is shown below:

|	Electrode	Target	L	V	fn400	n2	p2	p3	AUC
count	9000.000000	9000.000000	9000.000000	9000.000000	9000.000000	9000.000000	9000.000000	9000.000000	9000.000000
mean	14.500000	1.000000	0.500784	0.489205	0.250000	0.250000	0.250000	0.250000	0.502283
std	8.655922	0.816542	0.355223	0.263417	0.433037	0.433037	0.433037	0.433037	0.258893
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	7.000000	0.000000	0.159338	0.295272	0.000000	0.000000	0.000000	0.000000	0.316809
50%	14.500000	1.000000	0.506667	0.487045	0.000000	0.000000	0.000000	0.000000	0.505285
75%	22.000000	2.000000	0.840662	0.680742	0.250000	0.250000	0.250000	0.250000	0.687512
max	29.000000	2.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

Data Visualization

Once the data was refactored, it was much easier to visualize. Below is a visualization of fn400. (Red = Consistent, Blue = Control, Green = Misleading). From the beginning, it was abundantly clear that there existed no simple relationship between the stimuli and Latency/Voltage).

---

[1] Every Regex Expression along with other components used in this assignment was generated using Generative AI
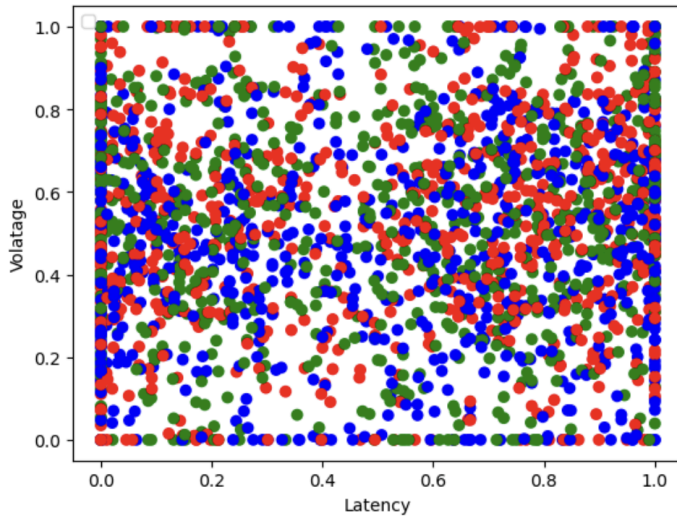
In order to simplify the visualization I also limited the stimulus to one electrode:
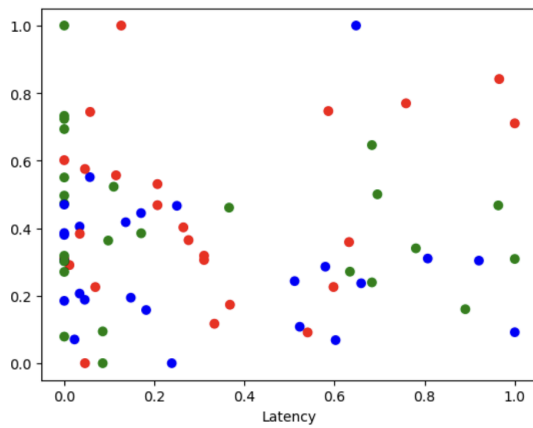
And also limited it to one subject:
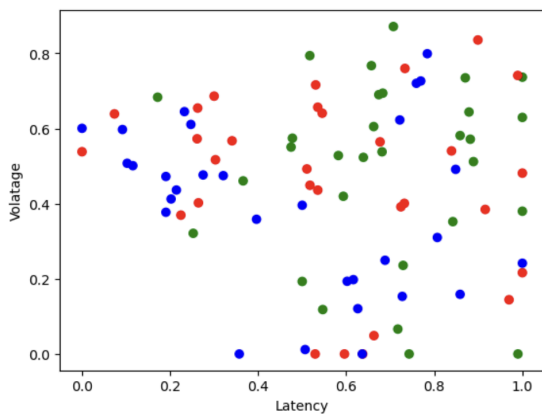
Finally I tried to perform the same calculation across all neural components filtered by electrode and/or participant but this also proved to be largely incomprehensible. Below is a picture of a scatter plot across all participant and neural components but limited to one electrode.

Feature Generation and Transformation

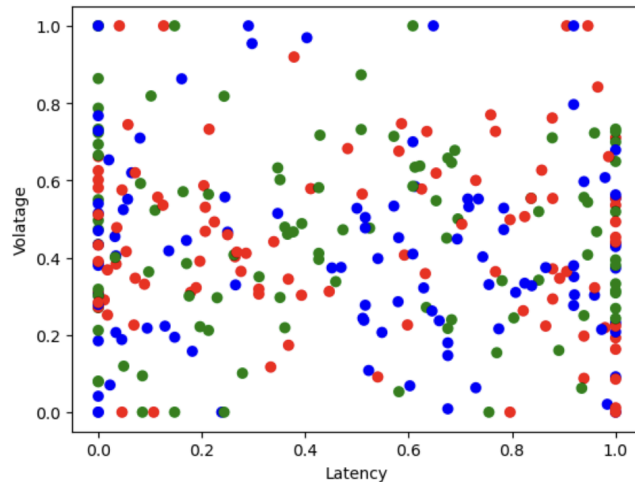The peaks and alltrials datasets went through two separate workflows separately before they ultimately merged together. We will discuss each one in detail.

Peaks
Data Pipeline:
1) *Scale the data* - That way the voltage values would not be considered more potent than the latency (even though V values were much larger). The file value was removed and reapplied after this step

Once the data was scaled we needed to separate column names into a more granular format. That way they could be separate features in the dataset and utilized by an ML model

2) *Melt the Data* - I melted the data into a three column dataset. The file value was utilized as an index, and every other column was reduced to a row and corresponding value.

	File	Electrode_Condition	Value
0	Training_000018	Fz-PeakDetection_Consistent-L	0.942857
1	Training_000019_002	Fz-PeakDetection_Consistent-L	0.914286
2	Training_000020_003	Fz-PeakDetection_Consistent-L	0.000000
3	Training_000021_004	Fz-PeakDetection_Consistent-L	0.914286
4	Training_000022_005	Fz-PeakDetection_Consistent-L	1.000000

Since the Electrode_Condition had so much information packed into a single column (Electrode, Target, Latency/Voltage), I needed to break it down into three separate features.

3) *Use Regex to Parse through the Column Name.* Specifically I looked for everything before the first dash to separate the electrode name, and everything between Underscore for the target value, and finally everything after a dash that resembled L or V. (Some peaks file headers varied in syntax so the regex was modified)

This separated the data into three more columns.

	File	Electrode_Condition	Value	Electrode	Type	Target
0	Training_000018	Fz-PeakDetection_Consistent-L	0.942857	Fz	L	Consistent
1	Training_000019_002	Fz-PeakDetection_Consistent-L	0.914286	Fz	L	Consistent
2	Training_000020_003	Fz-PeakDetection_Consistent-L	0.000000	Fz	L	Consistent
3	Training_000021_004	Fz-PeakDetection_Consistent-L	0.914286	Fz	L	Consistent
4	Training_000022_005	Fz-PeakDetection_Consistent-L	1.000000	Fz	L	Consistent

However I was unhappy with confusion required in having a Type column in which values oscillated from Latency and Voltage, which could easily confuse the model since values no longer represent one scale or type of value. This required me to generate a pivot table

4) *Pivot the data.* The data was pivoted with a three column index (File, Electrode, and Target) causing the type column to become features (latency and voltage). Pivot tables by nature aggregate values, in order to prevent this, I needed to utilize three columns in my index, that way every value was unique and there was nothing to be averaged.

Type	File	Electrode	Target	L	V
0	Training_000018	C3	Consistent	0.625000	0.577642
1	Training_000018	C3	Control	0.290323	1.000000
2	Training_000018	C3	Misleading	0.508197	0.873079
3	Training_000018	C4	Consistent	0.312500	0.383943
4	Training_000018	C4	Control	0.000000	0.118873
...	...	...	...	...	...

A similar pipeline was conducted for each of the peaks datasets. Next I wanted to merge these datasets, but wanted to keep their respective neural components. So I added a one-hot encoded neural component to display which component was being addressed.

	File	Electrode	Target	L	V	fn400	n2	p2	p3
0	Training_000018	C3	Consistent	0.264368	0.402273	1	0	0	0
1	Training_000018	C3	Control	0.806818	0.310069	1	0	0	0
2	Training_000018	C3	Misleading	0.365854	0.460715	1	0	0	0
3	Training_000018	C4	Consistent	0.517647	0.449013	1	0	0	0
4	Training_000018	C4	Control	0.769231	0.726599	1	0	0	0
...	...	...	...	...	...	...	...	...	...
2245	Training_000042_027	TP10	Control	0.060606	0.422181	1	0	0	0
2246	Training_000042_027	TP10	Misleading	0.080808	0.259728	1	0	0	0
2247	Training_000042_027	TP9	Consistent	0.881720	0.661044	1	0	0	0
2248	Training_000042_027	TP9	Control	0.825000	0.623602	1	0	0	0
2249	Training_000042_027	TP9	Misleading	0.971429	0.651374	1	0	0	0

All Trails

The alltrials dataset was simpler in the fact that each electrode had only value (area under the curve). Pipeline:

1) Dropped Fnp1 Columns from Analysis
2) Scale the Data. The file value was briefly excluded and the rest of the dataset was scaled.
3) Melt the Data. Once again one column contained too much information, so the data was melted into a two column format

	File	Electrode_Condition	AUC
0	Training_000018	Fz_BaselineCorrectionCONSISTENT	0.484940
1	Training_000019_002	Fz_BaselineCorrectionCONSISTENT	0.274322
2	Training_000020_003	Fz_BaselineCorrectionCONSISTENT	0.803439
3	Training_000021_004	Fz_BaselineCorrectionCONSISTENT	0.858007
4	Training_000022_005	Fz_BaselineCorrectionCONSISTENT	0.805356
...	...	...	...
2245	Training_000038_023	Fp2_BaselineCorrectionCONTROL	0.204478
2246	Training_000039_024	Fp2_BaselineCorrectionCONTROL	0.556565
2247	Training_000040_025	Fp2_BaselineCorrectionCONTROL	0.452178
2248	Training_000041_026	Fp2_BaselineCorrectionCONTROL	0.097821
2249	Training_000042_027	Fp2_BaselineCorrectionCONTROL	0.534941

4) Use Regex to separate Electrode_Condition into separate features.

	File	AUC	Electrode	Target
0	Training_000018	0.484940	Fz	Consistent
1	Training_000019_002	0.274322	Fz	Consistent
2	Training_000020_003	0.803439	Fz	Consistent
3	Training_000021_004	0.858007	Fz	Consistent
4	Training_000022_005	0.805356	Fz	Consistent
...	...	...	...	...
2245	Training_000038_023	0.204478	Fp2	Control
2246	Training_000039_024	0.556565	Fp2	Control
2247	Training_000040_025	0.452178	Fp2	Control
2248	Training_000041_026	0.097821	Fp2	Control
2249	Training_000042_027	0.534941	Fp2	Control

Since there was only one type of value, a pivot was not required.

Once all these steps were completed. I joined the alltrials dataset with peak files. This was possible because each neural component was associated with a specific time frame. (P2: 150–275 ms, FN400: 300-500ms, P3: 250-400ms, N2: 150-350m).

1) *Inner Join.* Since each dataset had the same patient, electrode, and target values I utilized an inner join (on these common features) so that the peaks dataset would have an extra column for the area under the curve (AUC)
2) Finally I concatenated all these datasets into one large dataset to be fed into the machine learning model.

	File	Electrode	Target	L	V	fn400	n2	p2	p3	AUC
0	Training_000018	C3	Consistent	0.264368	0.402273	1	0	0	0	0.290351
1	Training_000018	C3	Control	0.806818	0.310069	1	0	0	0	0.472220
2	Training_000018	C3	Misleading	0.365854	0.460715	1	0	0	0	0.370680
3	Training_000018	C4	Consistent	0.517647	0.449013	1	0	0	0	0.427806
4	Training_000018	C4	Control	0.769231	0.726599	1	0	0	0	0.659688
...	...	...	...	...	...	...	...	...	...	...
8995	Training_000042_027	TP10	Control	0.175676	0.187381	0	0	0	1	0.321755
8996	Training_000042_027	TP10	Misleading	0.932432	0.214467	0	0	0	1	0.366685
8997	Training_000042_027	TP9	Consistent	0.597222	0.294440	0	0	0	1	0.442891
8998	Training_000042_027	TP9	Control	1.000000	0.391564	0	0	0	1	0.359571
8999	Training_000042_027	TP9	Misleading	0.027027	0.216849	0	0	0	1	0.364039

Concluding Notes on Processing

It was already extremely clear that the high dimensionality of this dataset was going to make the process extremely difficult. I had some experimental methods that I did not attempt but would

have if given more time. One of these was to utilize PCA to reduce the number of dimensions in the model. Nevertheless, I chose against using this method in the end, because of the higher interdependence each feature has with the other.

Machine Learning Model

Once the dataset was completely merged and cleared, I prepared it for the machine learning model. Primarily I label encoded both the target value and the electrode value. I also removed File value to make the data more generalizable.

I considered dummy_encoding for the electrodes, but ultimately decided against it. This is because there are 32 electrodes. Adding 31 new column to the dataset could help stop the model from over emphasizing one model over another but it would exacerbate the curse of dimensionality that this dataset was facing.

Given more time for processing, the most ideal method is to dummy encode the electrodes and perform PCA to reduce its dimensionality.

Once the data was encoded, I split the data into 60% Training and 40% Test (I also tried a 70-30 split and got better results). I trained two machine learning models (both of which are ideal for complex relationships) a Random Forest Classifier and a Gradient Boosted Classifier. Unfortunately, I was left with undesirable results, both models provided an accuracy between 35 and 39%. With three classes (Consistent, Control, and Misleading) this is only slightly better than complete guessing (33%).

Conclusion

From this study, it is evident that brain EEG data is extremely complex. I was able to identify no simple relationships between any of the values, nor was I able to create a model that would accurately distinguish between normal and misleading brain stimuli. In hindsight, one of the largest issues I felt was the amount of dimensionality in this data. To mitigate this, I experimented removing features but got no higher accuracy. In the future I might employ PCA to reduce the dimensionality. I performed a correlation matrix to identify any relationships between variables

	L	V	fn400	n2	p2	p3	AUC	Electrode_Code	Target_Code
L	1.000000	0.010947	0.012824	0.030050	-0.062643	0.019770	-0.049390	-0.035021	-0.019045
V	0.010947	1.000000	0.052506	0.141263	-0.124661	-0.069108	0.761319	0.010632	-0.002051
fn400	0.012824	0.052506	1.000000	-0.333333	-0.333333	-0.333333	0.014618	-0.000000	-0.000000
n2	0.030050	0.141263	-0.333333	1.000000	-0.333333	-0.333333	0.009022	-0.000000	0.000000
p2	-0.062643	-0.124661	-0.333333	-0.333333	1.000000	-0.333333	-0.013779	0.000000	0.000000
p3	0.019770	-0.069108	-0.333333	-0.333333	-0.333333	1.000000	-0.009860	0.000000	0.000000
AUC	-0.049390	0.761319	0.014618	0.009022	-0.013779	-0.009860	1.000000	0.017167	-0.018776
Electrode_Code	-0.035021	0.010632	-0.000000	-0.000000	0.000000	0.000000	0.017167	1.000000	0.000000
Target_Code	-0.019045	-0.002051	-0.000000	0.000000	0.000000	0.000000	-0.018776	0.000000	1.000000

The only clear relationship existed between Voltage and AUC which of course was expected. All in all, I found this project extremely fun and intriguing, but was slightly disappointed with its result. Further analysis needs to be done to identify if any differences exist between brain activity when faced with misleading stimuli compared to normal stimuli.

Works Cited

"Pandas." *Pandas*, pandas.pydata.org/. Accessed 19 Dec. 2024.

"W3schools.Com." *W3Schools Online Web Tutorials*, www.w3schools.com/. Accessed 19 Dec. 2024.

"What is P2, FN400, P3 and N2 in ECG data?" prompt. ChatGPT, 13 Feb. version, OpenAI, 19 Dec. 2023, chat.openai.com.