Classification of Alcoholic and Control Individuals using Logistic Regression on EEG data

End Semestral Project
B.Stat 3rd Semester

Tuhin Saha (BS2127)
Avijit Saha (BS2109)

Instructor:
Dr. Shyamal Krishna De
Applied Statistics Division
Indian Statistical Institute, Kolkata

Contents

1 Introduction

1.1 Objective

The electroencephalogram (EEG) signal is an electrical representation of the brain's working that reflects various physiological and pathological activities such as alcoholism. Alcohol can affect whole parts of the body but particularly affects the brain, heart, liver, and immune system; its effects on the brain can cause brain disorders. Nowadays, the automatic identification of alcoholic subjects based on EEG signals has become one of the challenging problems in biomedical research.

The goal of this project is to determine whether a person is an alcoholic or not, given the EEG data of that person. We would like to clear one possible misunderstanding: we are not taking an intoxicated person to the test, instead, we are testing whether an individual in a sober state is a habituated alcoholic or not.

1.2 About Dataset

This data arises from a large study to examine EEG correlates of genetic predisposition to alcoholism. It contains measurements from 64 electrodes placed on the subject's scalps sampled at 256 Hz (3.9-msec epoch) for 1 second.

There were two groups of subjects: ”alcoholic” and ”control”. Each subject was exposed to either a single stimulus (S1) or to two stimuli (S1 and S2) which were pictures of objects chosen from the 1980 Snodgrass and Vanderwart picture set. When two stimuli were shown, they were presented in either a matched condition where S1 was identical to S2 or in a non-matched condition where S1 differed from S2.

Shown here are example plots of control and alcoholic subject. The plots indicate voltage, time, and channel and are averaged over 10 trials for the single stimulus condition.

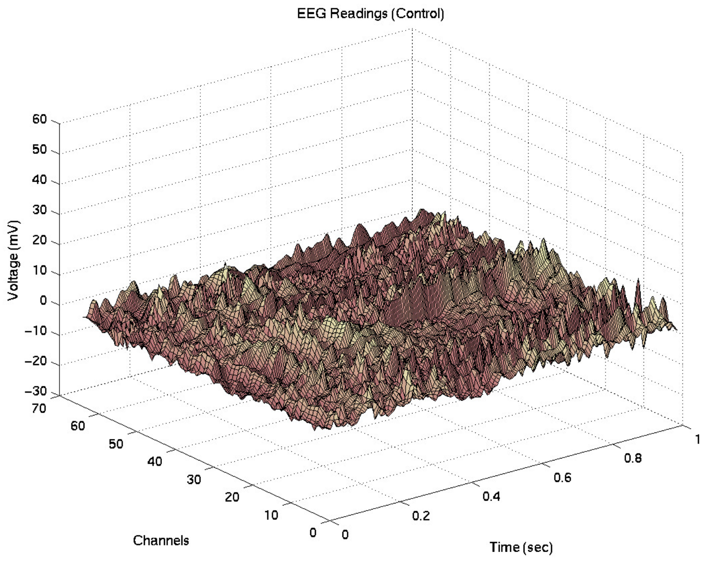

Figure 1: EEG Readings - Control

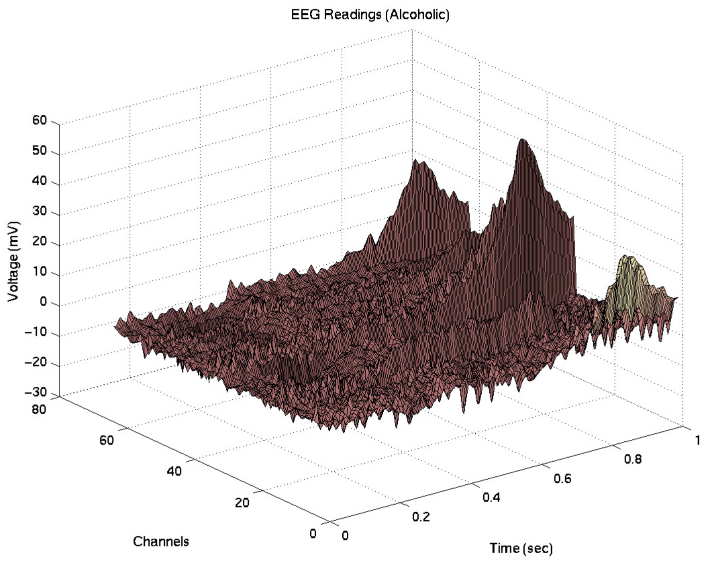

Figure 2: EEG Readings - Alcoholic

There were 122 subjects and each subject completed 120 trials where different stimuli were shown. The electrode positions were located at standard sites (Standard Electrode Position Nomenclature, American Electroencephalographic Association 1990). Zhang et al. (1995) describe in detail the data collection process.

The source of the data in this project is `https://www.kaggle.com/datasets/nnair25/Alcoholics?resource=download`. The original data has been sourced from Neurodynamics Laboratory at the State University of New York Health Center in Brooklyn. This can be found at `https://archive.ics.uci.edu/ml/datasets/eeg+database`.

The data used in this project has been pre-cleaned (A few observations were erroneous and were deleted from the dataset).

1.3 Attribute Information of the Data

Each trial is stored in a separate CSV file. A sample of how one CSV file looks is given below:

	A	B	C	D	E	F	G	H	I	J
1		trial number	sensor position	sample num	sensor value	subject identifier	matching condition	channel	name	time
2	5	23	FP1	0	-0.916	a	S2 match	0	co2a0000369	0
3	6	23	FP1	1	-1.404	a	S2 match	0	co2a0000369	0.00390625
4	7	23	FP1	2	-0.916	a	S2 match	0	co2a0000369	0.0078125
5	8	23	FP1	3	0.061	a	S2 match	0	co2a0000369	0.01171875
6	9	23	FP1	4	2.014	a	S2 match	0	co2a0000369	0.015625
7	10	23	FP1	5	3.967	a	S2 match	0	co2a0000369	0.01953125
8	11	23	FP1	6	5.432	a	S2 match	0	co2a0000369	0.0234375
9	12	23	FP1	7	5.92	a	S2 match	0	co2a0000369	0.02734375
10	13	23	FP1	8	6.409	a	S2 match	0	co2a0000369	0.03125
11	14	23	FP1	9	6.897	a	S2 match	0	co2a0000369	0.03515625
12	15	23	FP1	10	8.362	a	S2 match	0	co2a0000369	0.0390625
13	16	23	FP1	11	9.338	a	S2 match	0	co2a0000369	0.04296875
14	17	23	FP1	12	9.827	a	S2 match	0	co2a0000369	0.046875

Figure 3: Screenshot of Data108.csv

The columns of the data are:

- trial number
- sensor position - The position of the scalp where the sensor is placed. There are 64 such different positions
- sample number - This is the frequency value and it ranges from 0 to 255, both inclusive
- sensor value - The value of the measurement received in the sensor measured in micro-volts
- matching condition - The condition under which the measurements were taken. The values are as below:
  - S1 obj - The first object is shown
  - S2 match - The second object is shown in a matching condition
  - S2 nomatch - The second object is shown in a non-matching condition
- channel number - A unique number corresponding to every unique sensor position. It ranges from 0 to 63, both inclusive
- name - A serial code assigned to each subject
- time - Inverse of the frequency(sample num) measured in seconds
- subject identifier - The class to which the subject belongs to. The values are Alcoholic(a) or Control(c), essentially the output variable

2 Modifying the dataset

2.1 The Reason for the Modification

In the raw dataset, we have 948 CSV files. Each file corresponds to a particular trial of a particular person. In each sensor position, the sensor value for all the frequencies in the range was obtained. So, each CSV file has 64X256, i.e. 16384 rows. Essentially, all this information throughout these 16384 rows is only for 1 trial, i.e, 1 observation. We would need this information to be somehow compressed into 1 row so that we can apply statistical methods to this new table. Otherwise, it would be difficult to apply statistical methods onto 948 different files and somehow make them cohesive.

2.2 The Modification

Instead of taking the values of channel and frequency, we incorporate them into columns and take the sensor value in each instance as the value for that column. For example, if some row has channel 43, frequency 68, and sensor value 6, then the corresponding column name would be C43_F68 and the value for that column would be 6. So, there are 64X256 = 16384 columns and 948 rows in our new dataset. Thus, our new dataset will be a 948x16384 matrix. Just a small change, we would need one more column to indicate whether the trial was for S1 obj, S2 match, or S2 nomatch.

2.3 The Issue with this Method

If we want to apply logistic regression to this, we would need the covariates to be linearly independent, otherwise, the model will not be unique. Indeed, in our case, the covariates are not independent. This can be verified from the correlation matrix. In fact, it is quite reasonable to argue that for the same sensor position, if the frequency changes just a bit, the two values should not be much far apart, i.e. there is some relation between them. The way to overcome this has been discussed later. The correlation matrix for this is given below.

	C0_F0	C0_F1	C0_F2	C0_F3	C0_F4	C0_F5	C0_F6	C0_F7	C0_F8	C0_F9	C0_F10
C0_F0	1	0.531076	-0.34768	-0.62255	0.094726	0.816992	0.536757	-0.3425	-0.75554	-0.096	0.721203
C0_F1	0.531076	1	0.521774	-0.48535	-0.62204	0.195792	0.856822	0.514125	-0.45297	-0.75484	-0.03735
C0_F2	-0.34768	0.521774	1	0.384183	-0.49061	-0.53867	0.275368	0.837989	0.408023	-0.51622	-0.7212
C0_F3	-0.62255	-0.48535	0.384183	1	0.49471	-0.426	-0.59496	0.123598	0.813384	0.472683	-0.42526
C0_F4	0.094726	-0.62204	-0.49061	0.49471	1	0.454218	-0.48255	-0.62106	0.184183	0.840701	0.477641
C0_F5	0.816992	0.195792	-0.53867	-0.426	0.454218	1	0.454638	-0.42404	-0.59354	0.193379	0.839125
C0_F6	0.536757	0.856822	0.275368	-0.59496	-0.48255	0.454638	1	0.511701	-0.44939	-0.6149	0.165798
C0_F7	-0.3425	0.514125	0.837989	0.123598	-0.62106	-0.42404	0.511701	1	0.428996	-0.50883	-0.59712
C0_F8	-0.75554	-0.45297	0.408023	0.813384	0.184183	-0.59354	-0.44939	0.428996	1	0.442709	-0.47013
C0_F9	-0.096	-0.75484	-0.51622	0.472683	0.840701	0.193379	-0.6149	-0.50883	0.442709	1	0.472449
C0_F10	0.721203	-0.03735	-0.7212	-0.42526	0.477641	0.839125	0.165798	-0.59712	-0.47013	0.472449	1

Figure 4: The Initial Correlation Matrix

Also, the number of covariates is just ridiculously high. So, the model would be highly expensive in terms of space and time complexity. Interestingly, the way we handle the previous problem simultaneously solves this problem too!

2.4 An Effective Way Out

2.4.1 A Bit of Theoretical Stuff

2.4.1.1 Lemma 1

Let $X_1, X_2, ...X_n$ be n random variables and $X = (X_1, X_2, ..., X_n)^T$. Let v be any vector. So, $v^T X$ is a linear combination of the given random variables. Then,

$$var(v^T X) = v^T \Sigma v$$

where Σ is the covariance matrix of $X_1, X_2, ...X_n$.

2.4.1.2 Lemma 2

For any quadratic form x^{T}Ax, and

$$\max_{\|x\|=1} x^T Ax = \max_{\mathbf{y}\neq 0} \frac{\mathbf{y}^{\mathrm{T}}\mathbf{A}\mathbf{y}}{\mathbf{y}^{\mathrm{T}}\mathbf{y}} = \lambda$$

where λ is the largest eigenvalue of A. Moreover, $\frac{y^T Ay}{y^T y} = \lambda \Leftrightarrow y$ is an eigenvector of A corresponding to λ. And, the maximization actually works in a layered way.

2.4.2 Principal Component Analysis

Principal Component Analysis is a standard method for the recognition of statistical design in order to reduce dimensionality and is used for feature extraction. It is used to preserve important information regarding the pattern and is used to remove redundant information.

We are going to apply principal component analysis on the 948×16384 sensor value data matrix to get 16384 dimensional mean of row vectors, 947 orthogonal unit vectors of dimension 16384 spanning row space of the data matrix, components of every row vector of data matrix along the row space spanning unit vectors as 'scores' and non zero eigen-values of the covariance matrix of the sensor value data from which we can know the amount of variability along the direction of corresponding eigen-vectors.

We have to reduce the dimensions of the data without losing much variability. From the eigenvalue plot, it is pretty clear that eigen-values corresponding to eigen-vectors are close to 0 after 150 dimensions, and also more than 90% variability is explained by only 133 vectors. So, we are going to truncate the 'scores' matrix to get a 948 × 133 dimensional matrix which is included in 'new_df'.

3 Classification through Logistic Regression

3.1 A Small Change to the Dataset Again

The column for the matching condition is a categorical variable and it has 3 possible values. We handle this by dummy variables. So, we make 3 new columns instead of the matching condition column - S1 obj, S2 match, and S2 nomatch. Now if a trial was for S1 obj, then the

S1 obj value would be 1 and its value for the other two columns would be 0. Similarly, we do the same for S2 match and S2 nomatch.

One more thing, it is obvious that the sum of these values for these 3 columns in a row is 1. So, it makes sense to only add 2 of them, say we exclude S2 nomatch. So, for eg, if both of them had 0 as value, then the trial was for S2 nomatch and if one of them had a 1, the S2 nomatch value was 0. So, finally, the modification is that we remove the ”matching condition” column, we add two columns ”S1 obj” and ”S2 match” and their values are as explained above.

3.2 Finally, Logistic Regression

In the dataset, we had two folders of data ”SMNI_CMI_TRAIN” and ”SMNI_CMI_TEST”. We merge these two sets because they come from the same dataset initially and were divided into two equal parts. Now we split the merged dataset in a 0.8 ratio for Train vs Test. We will apply logistic regression on the Train part and test the model on the Test part.

Firstly we fit a logistic regression model using all 133 principal components and our dummy variables for ”S1 obj” and S2 match”. Here is the output.
Now let’s see the output(All the related code has been given later).

```
Call:
glm(formula = fmla, family = "binomial", data = train_reg)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.4874  -0.0911   0.0000   0.1142   2.5821

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -9.562e-01  4.726e-01  -2.023 0.043035 *
S1.obj      -4.699e-01  7.149e-01  -0.657 0.510934
S2.match     3.584e-01  6.418e-01   0.558 0.576507
X1          -1.263e-03  4.540e-04  -2.781 0.005414 **
X2           3.826e-05  5.024e-04   0.076 0.939289
X3           7.210e-03  1.131e-03   6.373 1.86e-10 ***
X4          -6.834e-03  1.334e-03  -5.124 2.99e-07 ***
X5           1.266e-03  1.136e-03   1.114 0.265335
X6          -2.079e-03  1.075e-03  -1.934 0.053160 .
X7          -8.295e-04  9.281e-04  -0.894 0.371450
X8           3.545e-03  1.260e-03   2.814 0.004888 **
X9           4.735e-03  1.247e-03   3.797 0.000146 ***
X10         -7.044e-03  1.616e-03  -4.358 1.31e-05 ***
X11          1.158e-02  1.937e-03   5.980 2.23e-09 ***
X12         -1.175e-02  2.002e-03  -5.872 4.32e-09 ***
X13          7.265e-03  1.612e-03   4.508 6.54e-06 ***
X14          2.325e-02  3.104e-03   7.490 6.88e-14 ***
```

X15	7.182e-03	1.947e-03	3.689	0.000225	***
X16	6.270e-03	1.952e-03	3.213	0.001315	**
X17	1.018e-02	2.689e-03	3.786	0.000153	***
X18	-2.221e-02	3.465e-03	-6.411	1.45e-10	***
X19	-5.760e-03	2.443e-03	-2.357	0.018402	*
X20	-2.753e-03	2.257e-03	-1.220	0.222462	
X21	8.633e-03	2.516e-03	3.432	0.000600	***
X22	-4.731e-03	2.135e-03	-2.216	0.026701	*
X23	3.564e-03	2.085e-03	1.709	0.087472	.
X24	-3.892e-03	2.472e-03	-1.574	0.115399	
X25	7.277e-03	2.420e-03	3.007	0.002637	**
X26	-4.040e-03	2.419e-03	-1.670	0.094857	.
X27	-6.615e-03	3.013e-03	-2.196	0.028123	*
X28	-4.407e-03	2.401e-03	-1.835	0.066475	.
X29	-1.862e-02	3.406e-03	-5.465	4.63e-08	***
X30	5.550e-03	2.763e-03	2.008	0.044601	*
X31	6.366e-03	2.603e-03	2.445	0.014473	*
X32	-3.874e-03	2.603e-03	-1.488	0.136733	
X33	-3.692e-03	2.754e-03	-1.341	0.180034	
X34	-2.722e-03	2.531e-03	-1.076	0.282054	
X35	-3.531e-03	2.973e-03	-1.188	0.234948	
X36	-3.067e-03	2.885e-03	-1.063	0.287767	
X37	6.229e-03	3.046e-03	2.045	0.040871	*
X38	7.017e-04	2.903e-03	0.242	0.808980	
X39	-1.790e-03	2.853e-03	-0.627	0.530457	
X40	8.995e-03	3.358e-03	2.679	0.007393	**
X41	9.059e-03	3.356e-03	2.699	0.006951	**
X42	1.038e-03	3.078e-03	0.337	0.735941	
X43	-6.050e-03	2.987e-03	-2.025	0.042850	*
X44	-4.407e-03	3.528e-03	-1.249	0.211639	
X45	-7.183e-03	3.229e-03	-2.225	0.026098	*
X46	6.999e-03	3.123e-03	2.241	0.025030	*
X47	-1.934e-03	3.514e-03	-0.550	0.582137	
X48	-1.086e-02	3.963e-03	-2.741	0.006132	**
X49	1.209e-03	3.349e-03	0.361	0.718095	
X50	1.079e-02	3.722e-03	2.898	0.003751	**
X51	-4.292e-03	3.488e-03	-1.231	0.218483	
X52	-2.561e-02	4.854e-03	-5.277	1.32e-07	***
X53	-1.149e-02	4.307e-03	-2.668	0.007629	**
X54	2.092e-02	4.187e-03	4.996	5.85e-07	***
X55	-3.193e-03	4.061e-03	-0.786	0.431727	
X56	-4.413e-04	3.731e-03	-0.118	0.905868	
X57	-2.783e-02	4.849e-03	-5.739	9.51e-09	***
X58	-1.175e-03	3.836e-03	-0.306	0.759406	
X59	-5.569e-03	4.018e-03	-1.386	0.165696	
X60	1.724e-02	4.444e-03	3.879	0.000105	***

X61	1.202e-02	4.052e-03	2.967	0.003007	**
X62	-7.776e-03	3.994e-03	-1.947	0.051524	.
X63	1.119e-02	4.469e-03	2.505	0.012252	*
X64	1.164e-03	4.479e-03	0.260	0.795045	
X65	7.716e-03	4.576e-03	1.686	0.091726	.
X66	-1.280e-03	4.599e-03	-0.278	0.780701	
X67	-3.123e-02	5.973e-03	-5.229	1.71e-07	***
X68	4.982e-03	4.592e-03	1.085	0.277952	
X69	3.349e-03	4.251e-03	0.788	0.430808	
X70	8.529e-03	5.099e-03	1.673	0.094397	.
X71	1.021e-02	4.655e-03	2.194	0.028245	*
X72	-1.551e-03	5.468e-03	-0.284	0.776673	
X73	-1.399e-02	5.373e-03	-2.604	0.009202	**
X74	-2.914e-02	6.240e-03	-4.670	3.01e-06	***
X75	2.090e-02	5.259e-03	3.974	7.06e-05	***
X76	1.376e-03	4.811e-03	0.286	0.774930	
X77	7.502e-03	5.210e-03	1.440	0.149879	
X78	2.120e-03	5.183e-03	0.409	0.682494	
X79	-1.582e-02	5.720e-03	-2.765	0.005694	**
X80	-2.412e-02	5.683e-03	-4.245	2.19e-05	***
X81	5.691e-03	4.704e-03	1.210	0.226360	
X82	-1.558e-03	5.541e-03	-0.281	0.778626	
X83	7.326e-03	4.857e-03	1.508	0.131495	
X84	1.032e-02	5.360e-03	1.925	0.054167	.
X85	8.853e-04	6.169e-03	0.144	0.885895	
X86	9.383e-03	5.512e-03	1.702	0.088679	.
X87	9.470e-03	5.517e-03	1.716	0.086088	.
X88	-8.917e-03	5.279e-03	-1.689	0.091200	.
X89	1.196e-02	5.669e-03	2.110	0.034875	*
X90	-1.014e-02	5.506e-03	-1.841	0.065559	.
X91	2.003e-02	6.158e-03	3.253	0.001141	**
X92	3.671e-03	5.365e-03	0.684	0.493802	
X93	1.249e-02	5.753e-03	2.170	0.029984	*
X94	1.286e-02	5.023e-03	2.560	0.010475	*
X95	2.412e-03	6.103e-03	0.395	0.692686	
X96	-2.423e-02	5.874e-03	-4.125	3.71e-05	***
X97	5.742e-03	5.825e-03	0.986	0.324232	
X98	3.042e-02	7.041e-03	4.320	1.56e-05	***
X99	-6.412e-03	5.298e-03	-1.210	0.226129	
X100	-9.961e-03	6.164e-03	-1.616	0.106121	
X101	2.635e-03	6.302e-03	0.418	0.675854	
X102	2.161e-02	6.065e-03	3.563	0.000367	***
X103	3.258e-02	7.136e-03	4.566	4.97e-06	***
X104	-1.294e-02	5.862e-03	-2.208	0.027229	*
X105	8.796e-03	6.016e-03	1.462	0.143691	
X106	6.470e-03	6.198e-03	1.044	0.296532	

```
X107         2.330e-03  6.019e-03   0.387 0.698717    
X108         7.750e-03  6.526e-03   1.188 0.235002    
X109         6.606e-03  6.476e-03   1.020 0.307718    
X110        -3.515e-03  7.006e-03  -0.502 0.615865    
X111        -2.111e-03  6.681e-03  -0.316 0.752042    
X112         1.012e-02  6.336e-03   1.598 0.110135    
X113        -3.543e-03  6.142e-03  -0.577 0.564010    
X114        -1.277e-02  6.662e-03  -1.918 0.055160 .  
X115        -5.254e-03  6.474e-03  -0.812 0.417043    
X116         2.765e-02  6.806e-03   4.063 4.84e-05 ***
X117        -3.731e-02  7.905e-03  -4.720 2.35e-06 ***
X118        -5.699e-05  6.364e-03  -0.009 0.992854    
X119         1.373e-03  6.698e-03   0.205 0.837609    
X120         4.506e-02  7.975e-03   5.650 1.60e-08 ***
X121         1.386e-03  6.626e-03   0.209 0.834282    
X122        -3.179e-03  6.615e-03  -0.481 0.630838    
X123        -1.139e-02  7.295e-03  -1.562 0.118291    
X124        -1.340e-02  6.887e-03  -1.946 0.051682 .  
X125         2.595e-02  7.045e-03   3.683 0.000230 ***
X126        -1.571e-02  6.486e-03  -2.422 0.015429 *  
X127         1.055e-02  6.625e-03   1.592 0.111390    
X128        -1.253e-02  7.068e-03  -1.772 0.076353 .  
X129        -1.013e-03  7.829e-03  -0.129 0.897100    
X130         6.723e-03  7.559e-03   0.889 0.373773    
X131        -2.279e-02  6.876e-03  -3.314 0.000919 ***
X132         1.165e-02  7.193e-03   1.619 0.105458    
X133        -3.283e-02  7.697e-03  -4.265 2.00e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1048.0  on 755  degrees of freedom
Residual deviance:  287.2  on 620  degrees of freedom
AIC: 559.2

Number of Fisher Scoring iterations: 9
```

From the summary of the model, it is pretty clear that most of the covariate principal components are insignificant in this model. And, we also note that though the number of covariates has been reduced down to 135, this number is still quite large.

Next, we observe that from the knowledge about the test method, it is clear that the matching condition should have had a significant influence on the model but the result shows there is no significant contribution from ”S1 obj” and ”S2 match”. This may happen because the matching-condition covariates are significantly correlated to other covariates which are in turn highly significant and so the influence of the matching condition is not much as was expected earlier. Essentially, the effect of the matching condition is contributed by other significant covariates. Here is the correlation matrix given below:

	S1.obj	S2.match	S2.momat	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
S1.obj	1	-0.50835	-0.49639	-0.31489	0.155626	0.168733	0.201828	-0.12163	0.017307	0.005456	0.092885	0.21233	-0.13297	-0.10939	-0.00985	0.007892	-0.07375	-0.02904	0.036256
S2.match	-0.50835	1	-0.49522	0.199077	-0.12108	-0.12549	-0.2092	0.200388	-0.06052	-0.01398	-0.06053	-0.06879	0.183979	0.139866	0.000255	0.00152	0.051303	0.101146	-0.04866
S2.momat	-0.49639	-0.49522	1	0.116989	-0.03495	-0.04372	0.007278	-0.0793	0.043552	0.008592	-0.03269	-0.14486	-0.05132	-0.03064	0.009679	-0.00949	0.02268	-0.07266	0.01248
1	-0.31489	0.199077	0.116989	1	-7.37E-17	-9.85E-16	-1.83E-16	-3.97E-16	-6.55E-17	-1.51E-17	5.48E-16	-1.85E-16	1.10E-15	2.93E-16	7.97E-16	-3.28E-16	-1.51E-15	1.02E-15	6.12E-16
2	0.155626	-0.12108	-0.03495	-7.37E-17	1	-9.19E-16	1.23E-15	9.98E-16	-4.46E-18	-3.07E-17	-6.82E-16	1.10E-15	-1.01E-15	-1.26E-17	-8.79E-17	2.93E-17	3.32E-16	-6.39E-16	1.88E-17
3	0.168733	-0.12549	-0.04372	-9.85E-16	-9.19E-16	1	6.10E-16	1.99E-16	1.04E-17	1.20E-17	-1.01E-16	1.46E-16	5.27E-16	-5.49E-17	-6.05E-17	-8.71E-18	-2.90E-16	6.56E-16	2.18E-16
4	0.201828	-0.2092	0.007278	-1.83E-16	1.23E-15	6.10E-16	1	-3.08E-16	1.26E-16	6.99E-17	6.02E-16	-9.07E-16	7.12E-17	4.86E-18	-6.14E-17	-9.05E-17	5.19E-16	-2.65E-16	-2.99E-16
5	-0.12163	0.200388	-0.0793	-3.97E-16	9.98E-16	1.99E-16	-3.08E-16	1	1.06E-17	1.08E-16	7.86E-16	4.18E-18	-2.27E-16	-3.36E-16	4.65E-17	-1.24E-16	-2.06E-17	-2.60E-17	-1.20E-17
6	0.017307	-0.06052	0.043552	-6.55E-17	-4.46E-18	1.04E-17	1.26E-16	1.06E-17	1	1.21E-15	-6.27E-17	-4.95E-16	-8.99E-16	-5.55E-16	8.80E-16	-6.58E-16	-3.02E-16	-1.28E-15	-5.64E-16
7	0.005456	-0.01398	0.008592	-1.51E-17	-3.07E-17	1.20E-17	6.99E-17	1.08E-16	1.21E-15	1	9.18E-16	3.37E-16	-2.39E-16	-8.94E-16	2.13E-16	2.45E-16	7.82E-16	4.00E-16	8.46E-17
8	0.092885	-0.06053	-0.03269	5.48E-16	-6.82E-16	-1.01E-16	6.02E-16	7.86E-16	-6.27E-17	9.18E-16	1	3.65E-16	1.06E-15	6.37E-17	-1.99E-16	1.70E-16	1.55E-15	-8.23E-16	-2.34E-16
9	0.21233	-0.06879	-0.14486	-1.85E-16	1.10E-15	1.46E-16	-9.07E-16	4.18E-18	-4.95E-16	3.37E-16	3.65E-16	1	1.16E-15	-6.30E-16	4.12E-16	-1.06E-16	-8.29E-16	3.92E-16	1.45E-16
10	-0.13297	0.183979	-0.05132	1.10E-15	-1.01E-15	5.27E-16	7.12E-17	-2.27E-16	-8.99E-16	-2.39E-16	1.06E-15	1.16E-15	1	-2.11E-16	-2.36E-16	3.10E-16	5.19E-16	-1.27E-16	5.75E-17
11	-0.10939	0.139866	-0.03064	2.93E-16	-1.26E-17	-5.49E-17	4.86E-18	-3.36E-16	-5.55E-16	-8.94E-16	6.37E-17	-6.30E-16	-2.11E-16	1	3.07E-15	-4.68E-16	2.02E-16	4.44E-16	4.19E-16
12	-0.00985	0.000255	0.009679	7.97E-16	-8.79E-17	-6.05E-17	-6.14E-17	4.65E-17	8.80E-16	2.13E-16	-1.99E-16	4.12E-16	-2.36E-16	3.07E-15	1	6.48E-16	-2.42E-16	3.70E-16	-8.87E-17
13	0.007892	0.00152	-0.00949	-3.28E-16	2.93E-17	-8.71E-18	-9.05E-17	-1.24E-16	-6.58E-16	2.45E-16	1.70E-16	-1.06E-16	3.10E-16	-4.68E-16	6.48E-16	1	-6.60E-17	-3.63E-16	3.76E-16
14	-0.07375	0.051303	0.02268	-1.51E-15	3.32E-16	-2.90E-16	5.19E-16	-2.06E-17	-3.02E-16	7.82E-16	1.55E-15	-8.29E-16	5.19E-16	2.02E-16	-2.42E-16	-6.60E-17	1	3.71E-16	9.01E-17
15	-0.02904	0.101146	-0.07266	1.02E-15	-6.39E-16	6.56E-16	-2.65E-16	-2.60E-17	-1.28E-15	4.00E-16	-8.23E-16	3.92E-16	-1.27E-16	4.44E-16	3.70E-16	-3.63E-16	3.71E-16	1	3.99E-16
16	0.036256	-0.04866	0.01248	6.12E-16	1.88E-17	2.18E-16	-2.99E-16	-1.20E-17	-5.64E-16	8.46E-17	-2.34E-16	1.45E-16	5.75E-17	4.19E-16	-8.87E-17	3.76E-16	9.01E-17	3.99E-16	1
17	-0.12241	0.102703	0.01996	-5.80E-16	5.36E-16	-7.33E-17	-5.22E-16	1.25E-17	-1.24E-15	3.09E-16	2.18E-17	9.37E-17	3.83E-16	5.04E-17	-4.71E-16	1.11E-15	-4.63E-16	-3.36E-16	9.56E-16

Figure 5: The Correlation Matrix

We can see the correlation matrix also supports the reasoning as "S1 obj" has a correlation coefficient of more than 0.2 with several other significant covariates(eg. X4) and similar results for "S2 match" also.

Hence we have chosen the 23 most significant covariates in the previous model namely X3, X4, X10, X11, X12, X13, X14, X18, X29, X52, X54, X57, X67, X74, X75, X80, X96, X98, X103, X116, X117, X120, X133 and have fit a logistic regression model using these 23 covariates only. Here is the summary of the updated model:

```
Call:
glm(formula = fmla_update, family = "binomial", data = train_reg)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6000  -0.6404   0.0569   0.6361   3.4420

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.2241429  0.1030265  -2.176 0.029586 *
X3           0.0022803  0.0003997   5.705 1.16e-08 ***
X4          -0.0024270  0.0004715  -5.148 2.64e-07 ***
X10         -0.0017227  0.0006478  -2.659 0.007835 **
X11          0.0052314  0.0007718   6.779 1.21e-11 ***
X12         -0.0041955  0.0007886  -5.320 1.04e-07 ***
X13          0.0023867  0.0007229   3.302 0.000961 ***
X14          0.0086102  0.0009652   8.920  < 2e-16 ***
X18         -0.0068978  0.0010057  -6.859 6.95e-12 ***
X29         -0.0058896  0.0012767  -4.613 3.97e-06 ***
X52         -0.0087851  0.0017259  -5.090 3.58e-07 ***
X54          0.0085811  0.0017576   4.882 1.05e-06 ***
X57         -0.0081921  0.0017985  -4.555 5.24e-06 ***
```

```
X67          -0.0087531  0.0021070  -4.154 3.26e-05 ***
X74          -0.0083934  0.0023231  -3.613 0.000303 ***
X75           0.0102005  0.0023191   4.399 1.09e-05 ***
X80          -0.0082447  0.0024512  -3.364 0.000770 ***
X96          -0.0069614  0.0027620  -2.520 0.011723 *
X98           0.0058844  0.0028549   2.061 0.039289 *
X103          0.0122848  0.0029778   4.125 3.70e-05 ***
X116          0.0078587  0.0031058   2.530 0.011394 *
X117         -0.0123944  0.0031812  -3.896 9.78e-05 ***
X120          0.0159639  0.0033222   4.805 1.55e-06 ***
X133         -0.0113184  0.0035451  -3.193 0.001410 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1048.02  on 755  degrees of freedom
Residual deviance:  632.14  on 732  degrees of freedom
AIC: 680.14

Number of Fisher Scoring iterations: 6
```

From null deviance and residual deviance, we observe that our model has a χ^2 value of 415.82 with 23 degrees of freedom and has a p-value less than 0.000001, which shows a high significance of our updated model.

Also, we calculated McFadden's R-squared and it comes out to be 0.397 which is quite good in this aspect.

4 Accuracy Tests

4.1 Confusion Matrix

The accuracy of the initial model is shown below:

```
   predict_reg
     0  1
  0 85 12
  1 17 78
[1] "Accuracy = 0.848958333333333"
```

The confusion matrix suggests for the null hypothesis "The subject is not alcoholic", the Type-I error probability is less than 12.5%. Though the model has a high accuracy of 84.9% the model is not very impressive because of an excessive number of insignificant covariates as we saw before.

The accuracy of the updated model is shown below:

```
   predict_reg_update
     0  1
  0 83 14
  1 22 73
[1] "Accuracy = 0.8125"
```

From the Confusion matrix of the updated model, we can see the Type-I error has slightly increased to 14.5%. But we have achieved more than 81% accuracy using only 23 covariates. So through this update of model, we have avoided using more than 100 extra covariates only losing 3% accuracy.

5 Conclusion

So, in this project, we demonstrated the use of logistic regression to classify between alcoholic and non-alcoholic people. A natural question that arises is how that may be of practical use. Sometimes, it is very important to know whether a person is an alcoholic or not. Let's for instance, take the case of steroid usage in medicine. In some medical procedures, doctors administer steroids to patients and their dosage varies depending on various factors and one of these factors is whether the patient is alcoholic or not. In fact, administering potent steroids to alcoholic individuals may even cause death in some cases. So, the determination of whether the patient is alcoholic or not becomes significant. The patient may not tell the truth due to societal pressure or some other reason and the resulting consequence can be fatal. So, here is one possible application of our project. In fact, the same techniques may be applied to derive a similar model for the effects of smoking and other drugs.

6 All the Related Code

6.1 Initial Dataset Modification

The codes only in 6.1 are in Python and the rest have been done in R

```python
# In this code, we take all the CSV files, modify them and
↪ then produce one single modified CSV file

import pandas as pd

# Create an empty dataframe with the relevant columns
fdf = pd.DataFrame()
fdf['S1 obj'] = ''
fdf['S2 match'] = ''
for c in range(64):
    for f in range(256):
        col_name = 'C' + str(c) + '_F' + str(f)
        fdf[col_name] = ''

# Fill up the columns
```

```
for i in range(1, 481): # For SMNI_CMI_TRAIN, it would be 469 instead of 481
    # For SMNI_CMI_TRAIN do the following:
    # df = pd.read_csv("./SMNI_CMI_TRAIN/Data"+str(i)+".csv")
    df = pd.read_csv("./SMNI_CMI_TEST/Data"+str(i)+".csv")

    ndf = pd.DataFrame()
    ndf['S1 obj'] = ''
    ndf['S2 match'] = ''
    for c in range(64):
        for f in range(256):
            col_name = 'C' + str(c) + '_F' + str(f)
            ndf[col_name] = ''

    lst = []
    if df.loc[0].at["matching condition"] == "S1 obj":
        lst.append(1)
        lst.append(0)
    elif df.loc[0].at["matching condition"] == "S2 match":
        lst.append(0)
        lst.append(1)
    else:
        lst.append(0)
        lst.append(0)

    for j in range(16384):
        lst.append(df.loc[j].at["sensor value"])
    ndf.loc[i-1] = lst
    fdf = pd.concat([fdf, ndf], ignore_index = True)
    fdf.reset_index()

# We transfer this dataframe to a csv file
fdf.to_csv("./SMNI_CMI_TEST/newDatafinal_Test.csv")
# For SMNI_CMI_TRAIN: fdf.to_csv("./SMNI_CMI_TRAIN/newDatafinal_Train.csv")
```

6.2 Principal Component Analysis Code

```
# Import the two datasets and merge them
df1 <- read.csv(file = "./SMNI_CMI_TEST/newDatafinal_Test.csv")
df2 <- read.csv(file = "./SMNI_CMI_TRAIN/newDatafinal_Train.csv")
df <- rbind(df1,df2)

# Correlation Matrix of the covariates
```

```
cor(df)

# Principal Component Analysis Code
meanx = apply(df[,-1:-4],2,mean)
y = scale(df[,-1:-4],scale=F) #Rows of y are the cases
A = y %*% t(y)
eig = eigen(A)
P = t(y) %*% eig$vec[,-948]
Q = apply(P,2,function(x) x/sqrt(sum(x*x)))
scores = y %*% Q

# Plot the eigenvalues and decide which components to take
plot(eig$values) # The plot is shown after this code
which(cumsum(eig$values / sum(eig$values)) >= 0.9)[1] # This value comes out to be 133.

# We add the S1 obj and S2 match columns to the dataset and turn it to a new csv
new_df=cbind(df[,2:3],scores[,1:133])
write.csv(new_df,"PCA_total.csv")
```

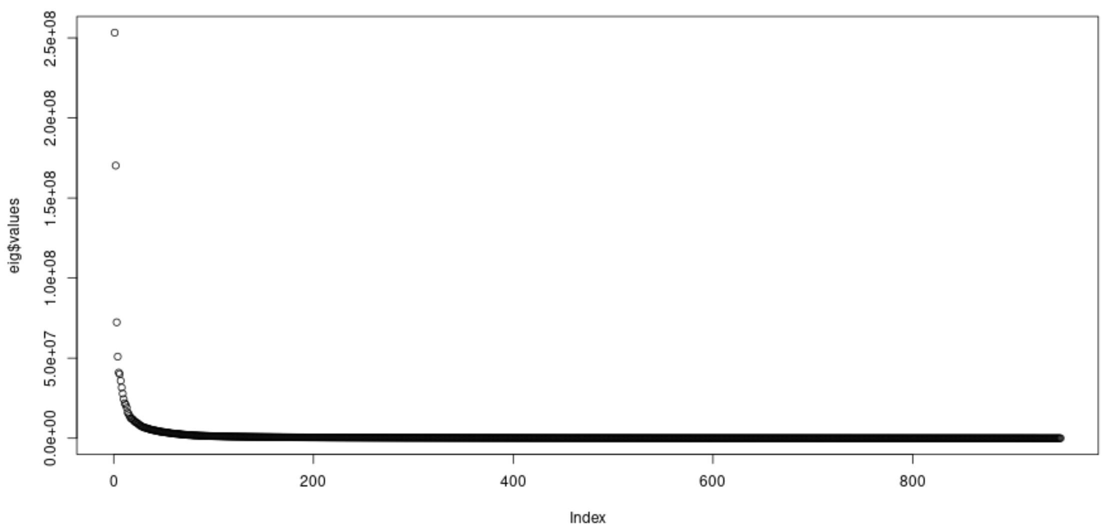

Figure 6: Plot of the EigenValues

6.3 Adding the output column to PCA_total.csv

```
rm(list = ls())
set.seed(150)

# We add the output column from SMNI_CMI_TEST
```

```r
data_al_pca <- read.csv("./PCA_total.csv")
otpt1 = c()
filename = paste("./SMNI_CMI_TEST/Data", as.character(1), ".csv", sep ="")
dt = read.csv(filename, nrows = 1)
id = dt$subject.identifier[1]
if (id == "c") {
  otpt1 = c(otpt1, 0)
} else {
  otpt1 = c(otpt1, 1)
}

for (i in 2:480) {
  filename = paste("./SMNI_CMI_TEST/Data", as.character(i), ".csv", sep ="")
  dt = read.csv(filename, nrows = 1)
  id = dt$subject.identifier[1]
  if (id == "c") {
    otpt1 = c(otpt1, 0)
  } else {
    otpt1 = c(otpt1, 1)
  }
}

# We do the same for SMNI_CMI_TRAIN
otpt2 = c()
filename = paste("./SMNI_CMI_TRAIN/Data", as.character(1), ".csv", sep ="")
dt = read.csv(filename, nrows = 1)
id = dt$subject.identifier[1]
if (id == "c") {
  otpt2 = c(otpt2, 0)
} else {
  otpt2 = c(otpt2, 1)
}

for (i in 2:468) {
  filename = paste("C:/Users/TAPAS/Desktop/Stat 3 Project/alcohol_data/SMNI_CMI_TRAIN/Data", as.character(i), ".csv", sep ="")
  dt = read.csv(filename, nrows = 1)
  id = dt$subject.identifier[1]
  if (id == "c") {
    otpt2 = c(otpt2, 0)
  } else {
    otpt2 = c(otpt2, 1)
```

```
    }
}

# Now we combine otpt1 and otpt2
otpt = c(otpt1, otpt2)
data_al_pca$output = otpt
write.csv(data_al_pca,"PCA_total_with_response.csv")
```

6.4 Logistic Regression Code

```
set.seed(150)

# Importing library
library ("caTools")

my_data <- read.csv(file = './PCA_total_with_response.csv')

xnam <- paste("X", 1:133, sep="")
fmla <- as.formula(paste("output ~ S1.obj + S2.match + ",
  ↪  paste(xnam, collapse= "+")))

# Splitting dataset
split <- sample.split(my_data, SplitRatio = 0.8)

train_reg <- subset(my_data, split == "TRUE")
test_reg <- subset(my_data, split == "FALSE")

# Training initial model
logistic_model <- glm(fmla,
                      data = train_reg,
                      family = "binomial")

# Summary
summary(logistic_model)

# Predict test data based on model
predict_reg <- predict(logistic_model,
                       test_reg, type = "response")

# Changing probabilities
predict_reg <- ifelse(predict_reg >0.5, 1, 0)

# Evaluating model accuracy
# using confusion matrix
table(test_reg$output, predict_reg)

```

```
missing_classerr <- mean(predict_reg != test_reg$output)
print(paste('Accuracy =', 1 - missing_classerr))
```

6.5 Model Improvement

```
# Training updated model using less covariates
fmla_update <- "output ~ X3 + X4 + X10 + X11 + X12 + X13 + X14
↪  + X18 + X29 + X52 + X54 + X57 + X67 + X74 + X75 + X80 +
↪  X96 + X98 + X103 + X116 + X117 + X120 + X133" # only 23
↪  most relevant covariates included
logistic_model_update <- glm(fmla_update,
                             data = train_reg,
                             family = "binomial")

# Summary
summary(logistic_model_update)

# Predict test data based on model
predict_reg_update <- predict(logistic_model_update,
                              test_reg, type = "response")

# Changing probabilities
predict_reg_update <- ifelse(predict_reg_update >0.5, 1, 0)

# Evaluating model accuracy using confusion matrix
table(test_reg$output, predict_reg_update)

missing_classerr_update <- mean(predict_reg_update !=
↪  test_reg$output)
print(paste('Accuracy =', 1 - missing_classerr_update))
```

7 Acknowledgement

We would like to thank Prof. Shyamal Krishna De for giving us this oppurtunity to work on this project and helping us whenever we faced hurdles.

8 References

- Data Driven Science and Engineering by Steven L Brunton, J Nathan Kutz
- Linear Algebra by by A. Ramachandra Rao, P Bhimasankaram
- Statistical Inference by George Casella, Roger L. Berger
- R. Kaur, Er. Himanshi-"Face recognition using Principal Component Analysis"- IEEE Paper, July 2015