ISL Project Report

Grammatical Facial Expressions Recognition Using Machine Learning

Team Members

Sumitra Dey
(16234351)
&
Shubhabrata Mukherjee
(16201097)

Table of Contents

Page #

1. The problem statement:

The dataset used in our project was developed by Freitas *et al.*to recognize facial expressions used in Brazilian Sign Language “Libras” and to generate a machine learning based grammatical structure for an unambiguous interpretation [1]-[2]. Using Microsoft Kinect sensor, Freitas *et al.* recorded 9 categorical videos for each of two users “a” and “b” (total 18 videos) and divided the data in timeframes. The data was tabulated as follows: (a) an image of each frame, identified by a timestamp; (b) a text file containing one hundred coordinates (x, y, z) of points from eyes, nose, eyebrows etc., each line in the data corresponds to points extracted from one frame.

Our goal in this project is to identify whether a person is expressing any sign or is silent. Thus it is a classification problem based on the facial expression. We would also like to identify the nature of dialog from the expression if the time and complexity permits.

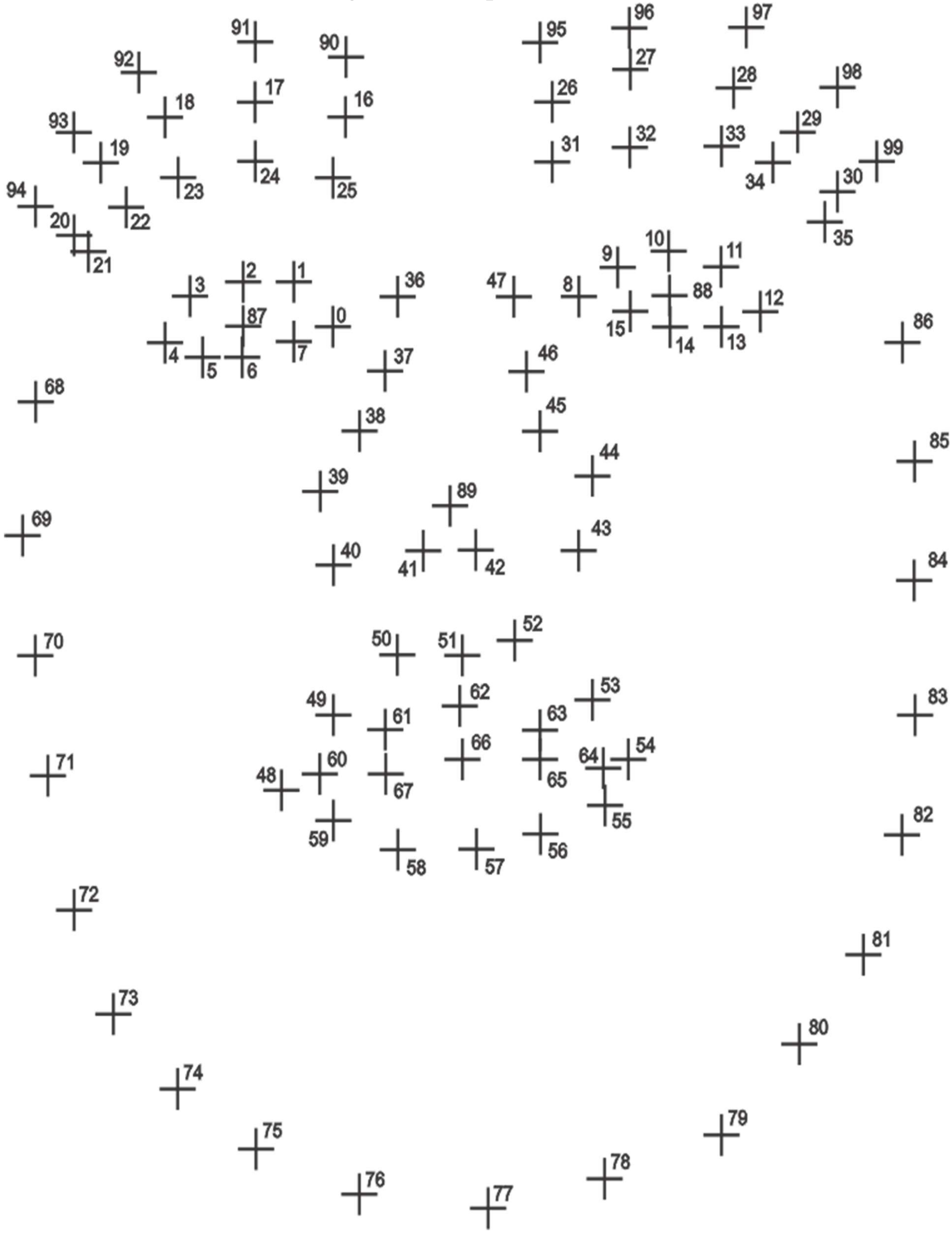
2. Literature survey:

We primarily used “Grammatical Facial Expressions Recognition with Machine Learning” by FREITAS, F. A.; Peres, S. M.; Lima, C. A. M.; BARBOSA, F. V. as our primary source of information. This paper followed mainly neural network multilayer perception [2] for grammatical facial expression recognition.

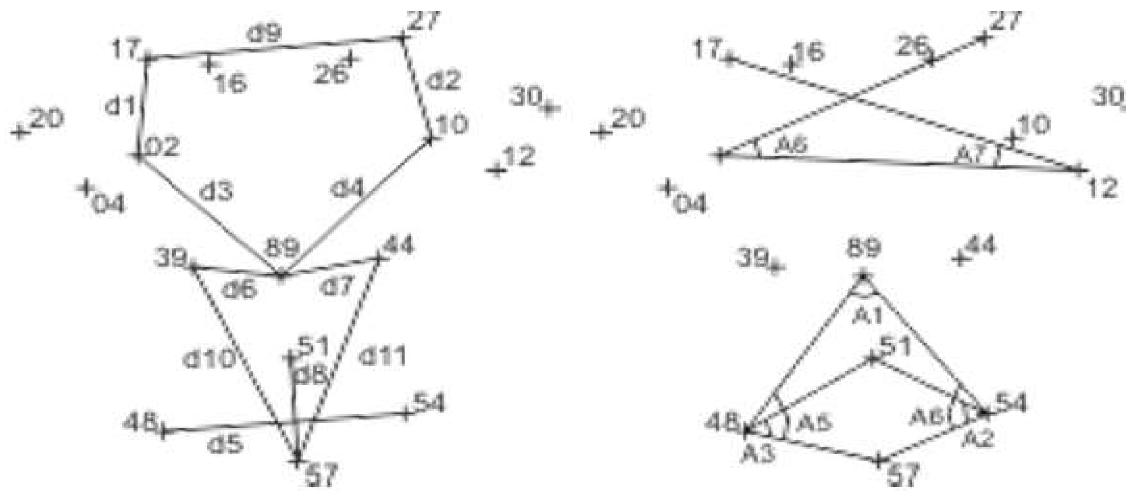
3. Description of the dataset

The original dataset is a multivariate sequential with real attributes. There are 27956 instances and 100 points (3 coordinates per attribute) or attributes. The dataset is organized in 36 files: 18 data point files and 18 target files, one pair for each video which compose the dataset. The name of the file refers to each video: the letter corresponding to the user (A and B), name of grammatical facial expression and a specification (target or data points). Total 100 points on face located using coordinates x and y (given in pixels) and z (in millimeters). The 100 points are marked serially as shown in Fig. 1. below.

Fig 1: Point representation of face



In [2], the authors have implemented supervised neural networks with Multi-layer perception (MLP) with the Matlab R Neural Toolbox. They reduced the 300 (x,y,z) coordinates first to 100 2D points with 200 (x,y) coordinates. Then further reduced the variable space with neural network to a subset of 17 significant points. With these 17 significant points they generated 11 distances and 7 angles and used these 18 transformed variable in all of their experimentations.



(a) (b)

Fig. 2. Reduced and transformed set of variables.

4. Prepossessing Steps in current study:

A. Unsupervised learning:

The used data set has a total 300 predictors. Each file of any user (A or B) have around 12000 observations or samples. So, in our case we have p = 300 and n = 12000. So, to examine a relatively significant set of predictors we used **“Principal component analysis”** as an unsupervised learning technique for preprocessing the data.

B. Dimension reduction:

_For dimension reduction we mainly used variable transformation. We transformed the 17 data points and derived total 11 distances and 7 angles from [2].

5. Type of approaches followed:

For this project we have followed an optimum combination of supervised and unsupervised learning. Below is a short description of each method used and their applicability in our scope.

Supervised learning:

Once we can extract the features which is significant to determine the class of data, we follow several supervised learning approaches as below:

A. Logistic regression:

As our goal is to perform some qualitative prediction, we selected "Logistic regression" as the primary method to divide data in 2 main classes called "Talking" and "Silent".

B. Validation methods:

We primarily used validation set approach to determine the amount of classification error. For a more unbiased validation we also used K fold cross validation approach.

C. Experimental approaches:

We also explored the data set with a few experimental methods as discussed below:

(i) Time series approach:

As the type of data is temporal in nature, hence we have explored several time series techniques to analyze time domain variation of different significant points in face. In our final opinion, we found time series analysis is not an appropriate method for our modeling as, the data is not continuous rather mostly stochastic in nature.

(ii) Support Vector machine:

We also explored the possibility to use support vector machine as we have 2 distinct classes (Silent and Talking). But as per our observation, we are working with a very large data set and SVM is not an ideal method for our problem, because, SVM is not suitable for Classification of large data sets as in order to find a separation hyperplane costs an intensive computational Complexity.

6. The model: detailed description

As mentioned in (4 – A) we mainly followed PCA for data preprocessing. The scree plot shows (Fig -2) the first principal component is responsible for around 90% of variation in data. Hence, we are able to use PCA 1 as a primary tool to validate the 17 significant points as described in paper [2].

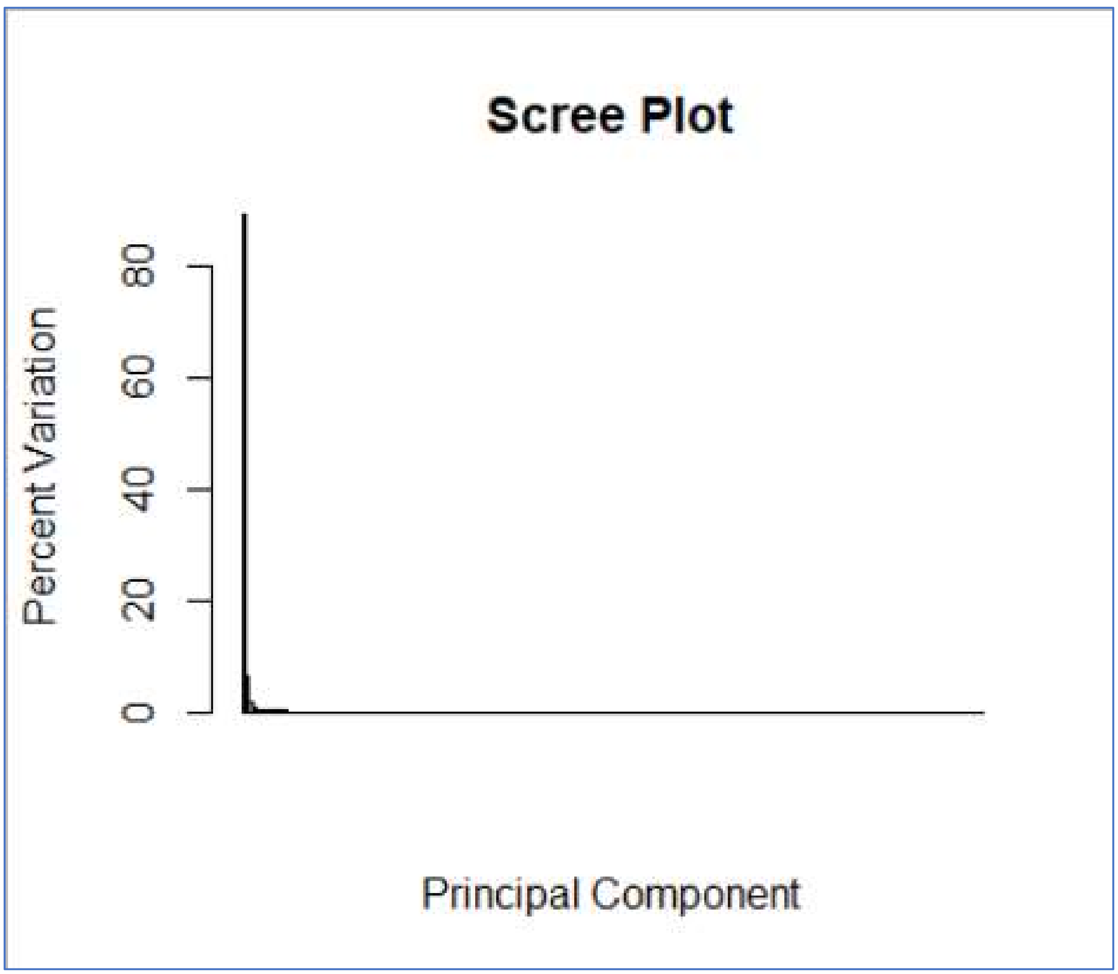


Fig 3: Scree plot on original data set

In our case PCA served 2 purpose, it used as a tool for preprocessing, as well as PCA was used as means of subset selection to perform main classification operation on fewer number of predictors. Below is a comparison of the loading for top 10 most significant variable as per PCA. This tells us the amount of weightage the variables have in the overall variance of the data.

```
> ## show the names of the top 10 predictors with scores (and +/- sign)
> pca$rotation[top_20_coords,1]
        X53         Y52         X52         X51         X48         Y49         X50         Y50 
-0.07294097 -0.07293365 -0.07291740 -0.07291547 -0.07291314 -0.07291085 -0.07290769 -0.07290709 
        Y47         Y51         X49         Y48         X47         X46         Y54         X55 
-0.07290616 -0.07290540 -0.07290031 -0.07289796 -0.07288727 -0.07287937 -0.07287854 -0.07287625 
        Y55         Y46         X54         Y53 
-0.07287256 -0.07286928 -0.07286432 -0.07285738 
```

Fig 4: Loading value for predictors in original data set

In next step, we perform dimension reduction method. From the 17 most significant variable we obtained, we construct a distance and angle matrix. The distances and angles used for this matrix construction are described as below:

Distance	Points used
d1	2,17
d2	10,27
d3	2,89
d4	10,89
d5	48,54
d6	39,89
d7	44,89
d8	51,57
d9	17,27
d10	39,57
d11	44,57

Angle	Points used
a1	48,89,54
a2	51,54,57
a3	51,48,57
a4	89,54,57
a5	89,48,57
a6	27,2,10
a7	17,10,2

Table 1: Distances and angles

We constructed and used two separate function **myangle** () to calculate angle between 3 points and **mydistance** () to calculate distance between 2 points. And constructed the matrix named **new.mat**. This matrix used as an input to create our classifications model.

Finally, we will now classify our dimensionally reduced data set into two major class as (i) Silent and (ii) Talking. We used logistic regression for this primary classification with 18 predictors.

We first perform validation set approach. Here we used around 50% of data sample as training set and rest for test set. In next step for more accurate estimation we followed K fold cross validation approach. We selected K = 10.

Next, we again used PCA in this step to validate and improve the results of classification. First principal component gives us 99% weightage. So, using PCA 1 we selected only first 10 parameters with highest loading score. We used only these 10 predictors instead of all 18 we used last time.

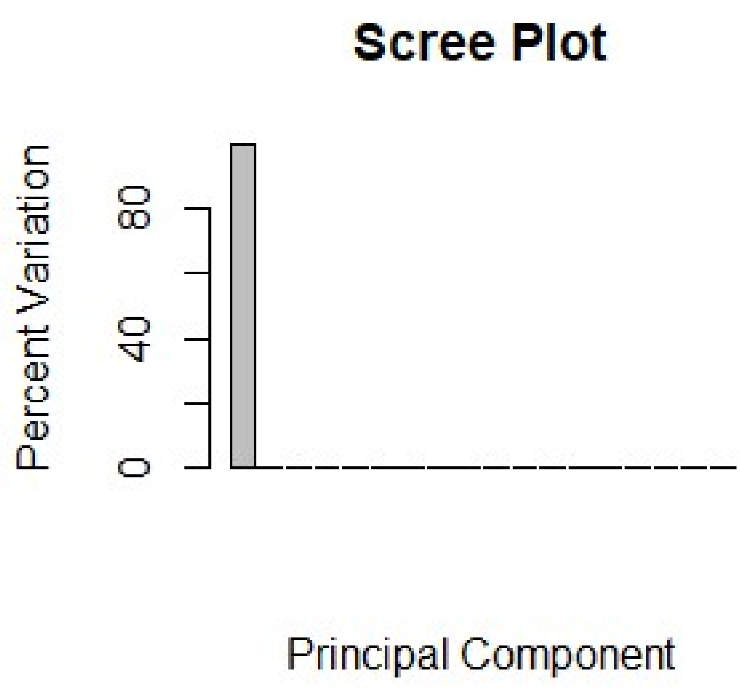


Fig 5: Scree plot on reduced data set with 11 angles and 7 distances

We could achieve lower classification error in only 10 variables using PCA rather than using all 18 parameters used.

7. Results and visualizations

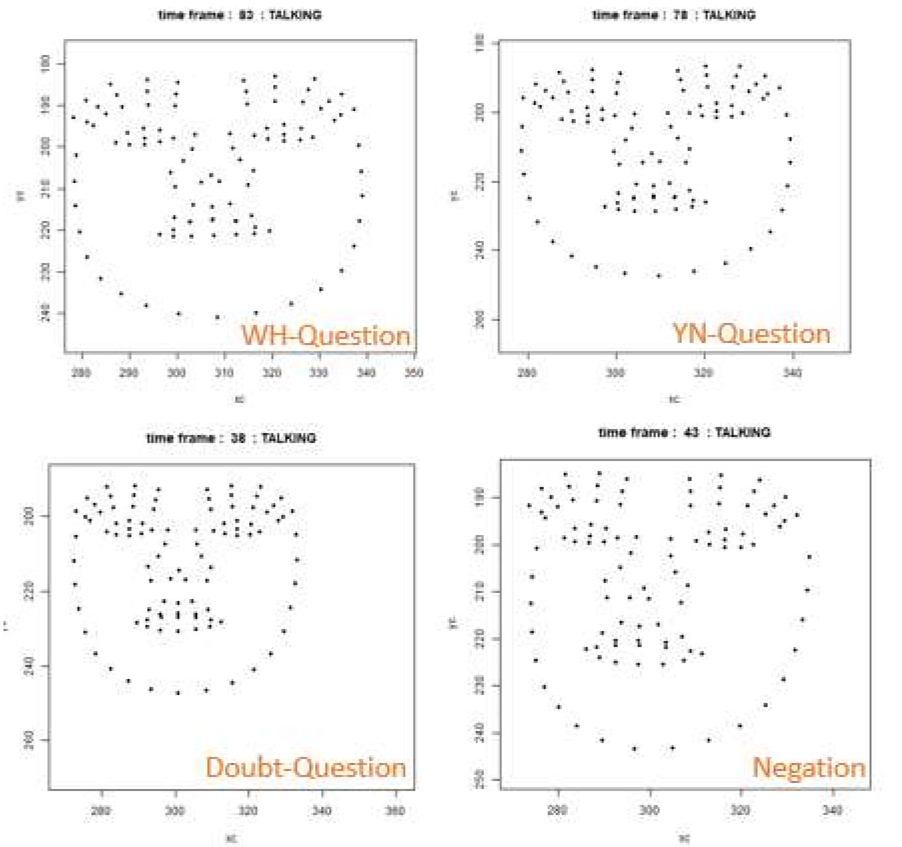


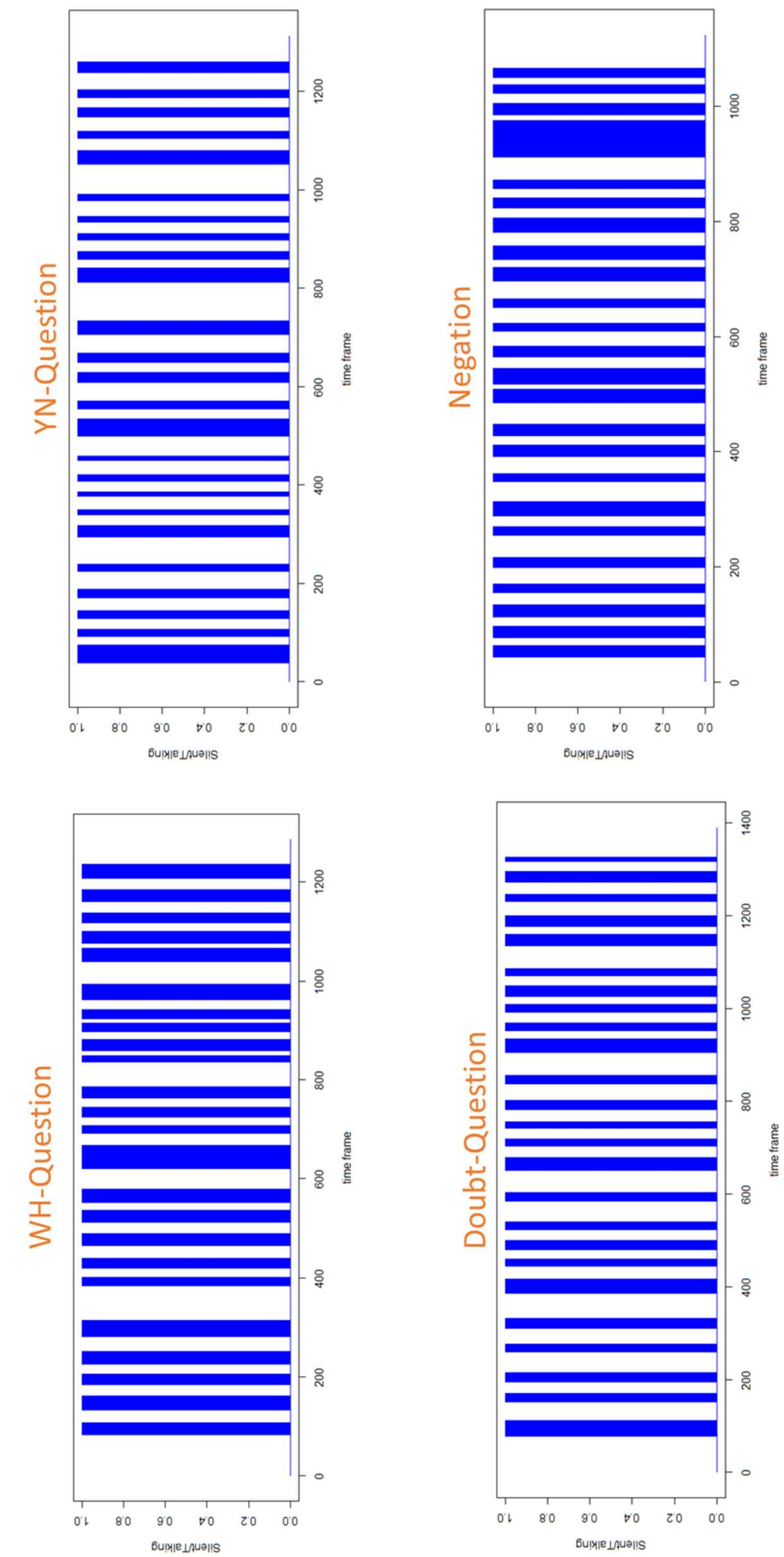
Fig 6a: Sample faces from different conversation

Semantic Functions	Eyebrows	Eyes	Mouth	Head
WH-question	↑			↑
Yes/no question	↑			↓
Doubt question	↓	∗	∗	⊖
Topic		⋄		↓
Negation	↓		∩	↔
Assertion				↕
Conditional clause		As in yes/no question		
Focus		As in topic		
Relative clause	↑			

↑ – upward head; ↓ – downward head
↕ – up and downward head; ↔ – left and rightward head
∗ – compressed mouth; ⋄ – open mouth; ∩ – downward mouth
⊕ – aproximation; ⊖ – detachment

Fig 6b: Grammatical Sign Rules

Fig 6c: Temporal nature of data



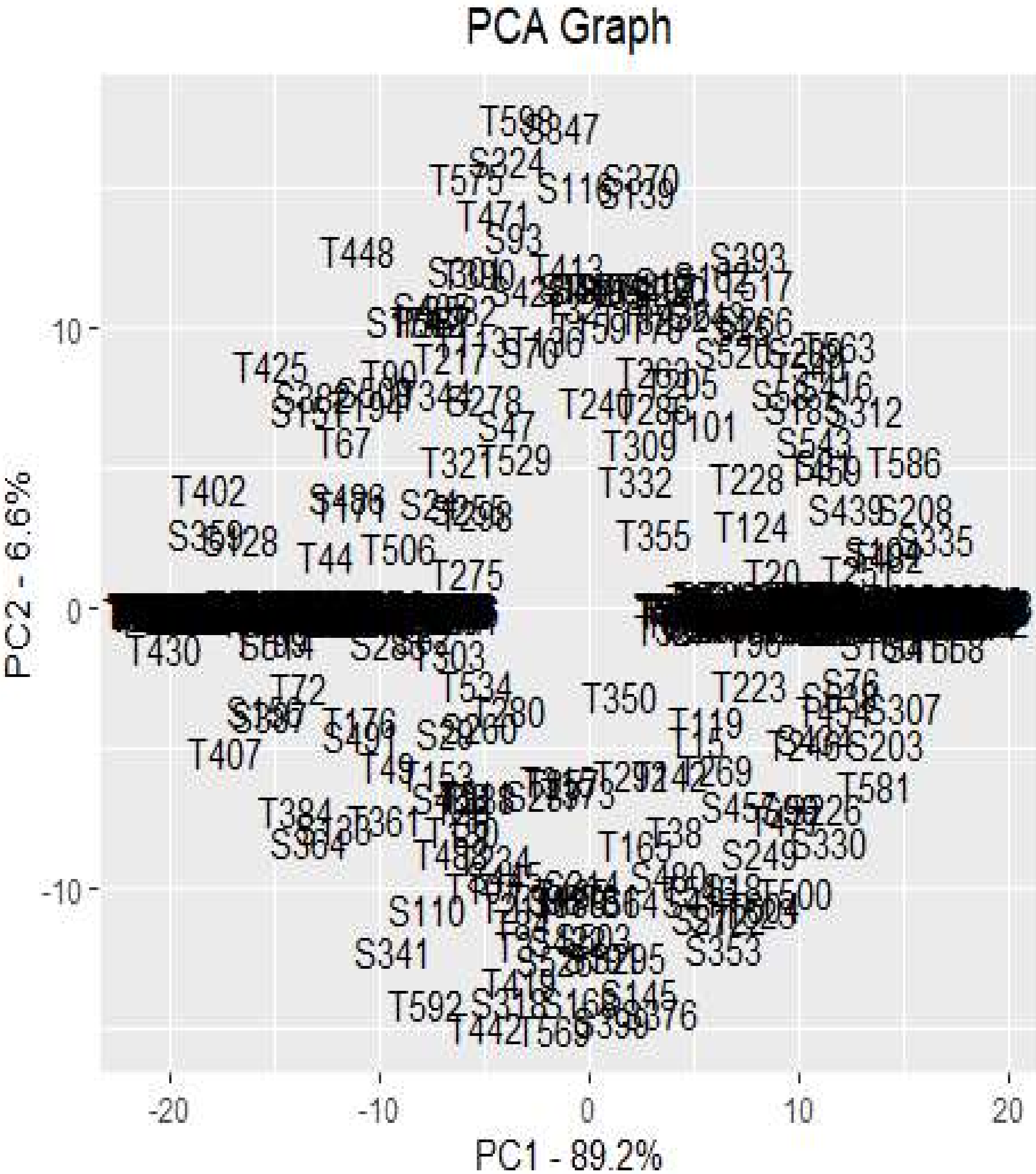


Fig 7: PCA on Raw Data Set (200 variables)

```
> summary(myglm1)

Call:
glm(formula = V1 ~ ., family = binomial, data = m.train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.2109  -0.1128  -0.0064   0.0215   3.2968

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  537.049    198.519   2.705  0.00682 **
d1            78.032    168.321   0.464  0.64294
d2           -24.970    172.052  -0.145  0.88461
d3           119.288     44.714   2.668  0.00763 **
d4           -32.491     44.142  -0.736  0.46170
d5            46.310     14.277   3.244  0.00118 **
d6          -351.126    162.372  -2.162  0.03058 *
d7           203.415    153.328   1.327  0.18462
d8            78.503     19.091   4.112 3.92e-05 ***
d9           -24.829      5.744  -4.322 1.54e-05 ***
d10          -97.180     59.237  -1.641  0.10089
d11           -3.680     62.212  -0.059  0.95284
a1            -2.367      1.909  -1.240  0.21491
a2            -7.293      7.066  -1.032  0.30198
a3            10.609      6.980   1.520  0.12855
a4            19.300      9.409   2.051  0.04025 *
a5           -20.263      7.824  -2.590  0.00960 **
a6            46.809     36.525   1.282  0.20000
a7           -69.557     36.884  -1.886  0.05932 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1759.62  on 1284  degrees of freedom
Residual deviance:  226.39  on 1266  degrees of freedom
AIC: 264.39

Number of Fisher Scoring iterations: 10
```

Fig 8: Logistic regression on 11 distance and 7 angles

```
> glm.prob = predict(myglm1,m.test, type = "response")
> glm.pred = rep(0,nrow(m.test))
> glm.pred[glm.prob > .5] = 1
> testv1 = m.test$V1
> table(glm.pred,testv1)
        testv1
glm.pred   0   1
       0 262  61
       1  52 268
> mean(glm.pred==testv1)
[1] 0.8242613
```

Fig 9: Logistic Regression results using validation set approach

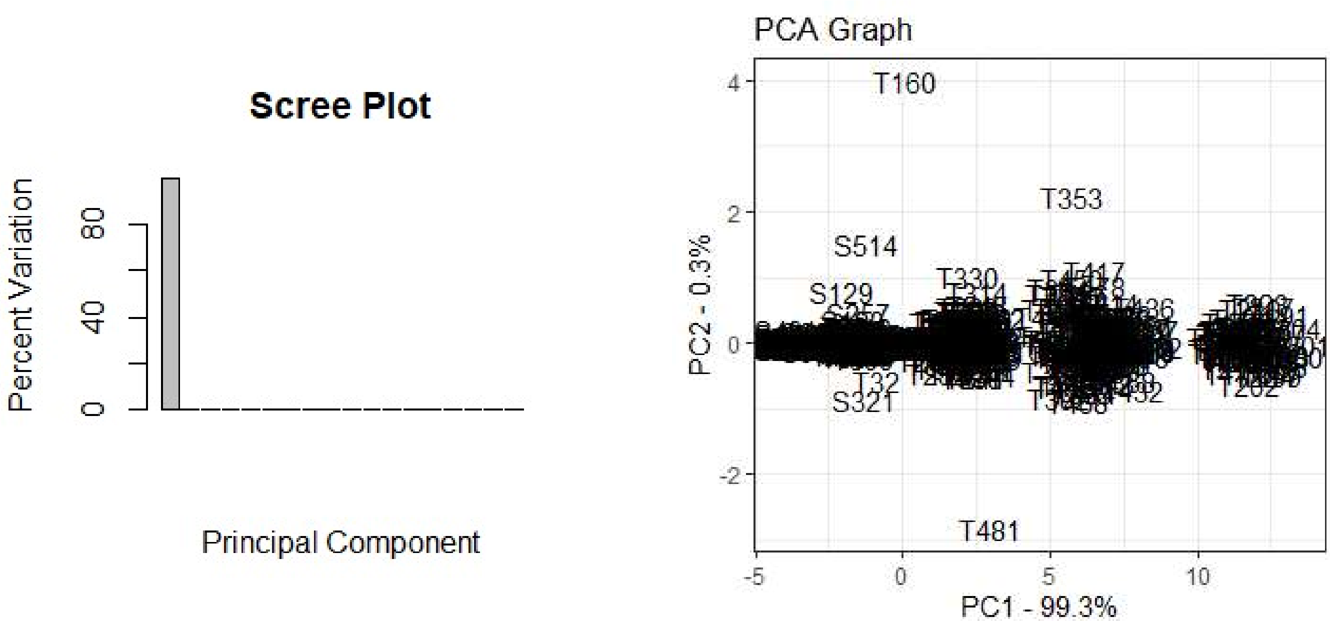


Fig 10: PCA on Transformed and Reduced Set of Variables

```
> ## show the names of the top 10 predictors with scores (and +/- sign)
> pca$rotation[top_10_da,1]
      d10        d7        a1        d8        d9       d11
0.2360500 0.2360377 0.2359802 0.2359791 0.2359768 0.2359725
       a2        d6        d4        d5
0.2358671 0.2358536 0.2358249 0.2357947
```

Fig 11: Loading value for predictors in reduced Set of Variables

```
> summary(myglm2)

Call:
glm(formula = V1 ~ d10 + d7 + a1 + d8 + d9 + d11 + a2 + d6 +
    d4 + d5, family = binomial, data = m.train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.1939  -0.1352  -0.0129   0.0342   3.3254

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  310.207     46.716   6.640 3.13e-11 ***
d10            6.823      2.500   2.730  0.00634 **
d7          -174.715     32.506  -5.375 7.67e-08 ***
a1            -5.812      1.087  -5.348 8.91e-08 ***
d8            64.971     12.123   5.359 8.36e-08 ***
d9            -8.705      1.622  -5.368 7.95e-08 ***
d11         -114.642     22.311  -5.138 2.77e-07 ***
a2             4.416      1.893   2.333  0.01964 *
d6            87.851     16.460   5.337 9.43e-08 ***
d4            79.161     15.232   5.197 2.03e-07 ***
d5            42.354      8.410   5.036 4.75e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1759.6  on 1284  degrees of freedom
Residual deviance:  253.5  on 1274  degrees of freedom
AIC: 275.5

> glm.prob = predict(myglm2,m.test, type = "response")
> glm.pred = rep(0,nrow(m.test))
> glm.pred[glm.prob > .5] = 1
> testv1 = m.test$V1
> table(glm.pred,testv1)
        testv1
glm.pred   0   1
       0 253  46
       1  61 283
> mean(glm.pred==testv1)
[1] 0.8335925
```

Fig 12: Logistic regression on 10 variables.

```
> #fix(new.mat)
> m=new.mat
>
> # Define train control for k fold cross validation
> train_control = trainControl(method="cv", number=10)
> # Fit Naive Bayes Model
> m[,1]=as.factor(m$V1)
> #fix(m)
> ######### CV with 18 variables ################
> model = train(V1 ~., data=m, trControl=train_control, method$
> # Summarise Results
> print(model)
Generalized Linear Model

1286 samples
  18 predictor
   2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 1158, 1157, 1157, 1158, 1157, 1157, .$
Resampling results:

  Accuracy   Kappa
  0.9097808  0.8188281
```

Fig 13: Cross-validation on 18 variables

```
> #fix(new.mat)
> m=new.mat
>
> # Define train control for k fold cross validation
> train_control = trainControl(method="cv", number=10)
> # Fit Naive Bayes Model
> m[,1]=as.factor(m$V1)
> #fix(m)
> ######### CV with 10 variables chosen by PCA ################
> model = train(V1 ~ d10+d7+a1+d8+d9+d11+a2+d6+d4+d5, data=m, $
> # Summarise Results
> print(model)
Generalized Linear Model

1286 samples
  10 predictor
   2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 1157, 1157, 1158, 1157, 1158, 1157, .$
Resampling results:

  Accuracy   Kappa
  0.9113312  0.8219977
```

Fig 14: Cross-validation on 10 variables

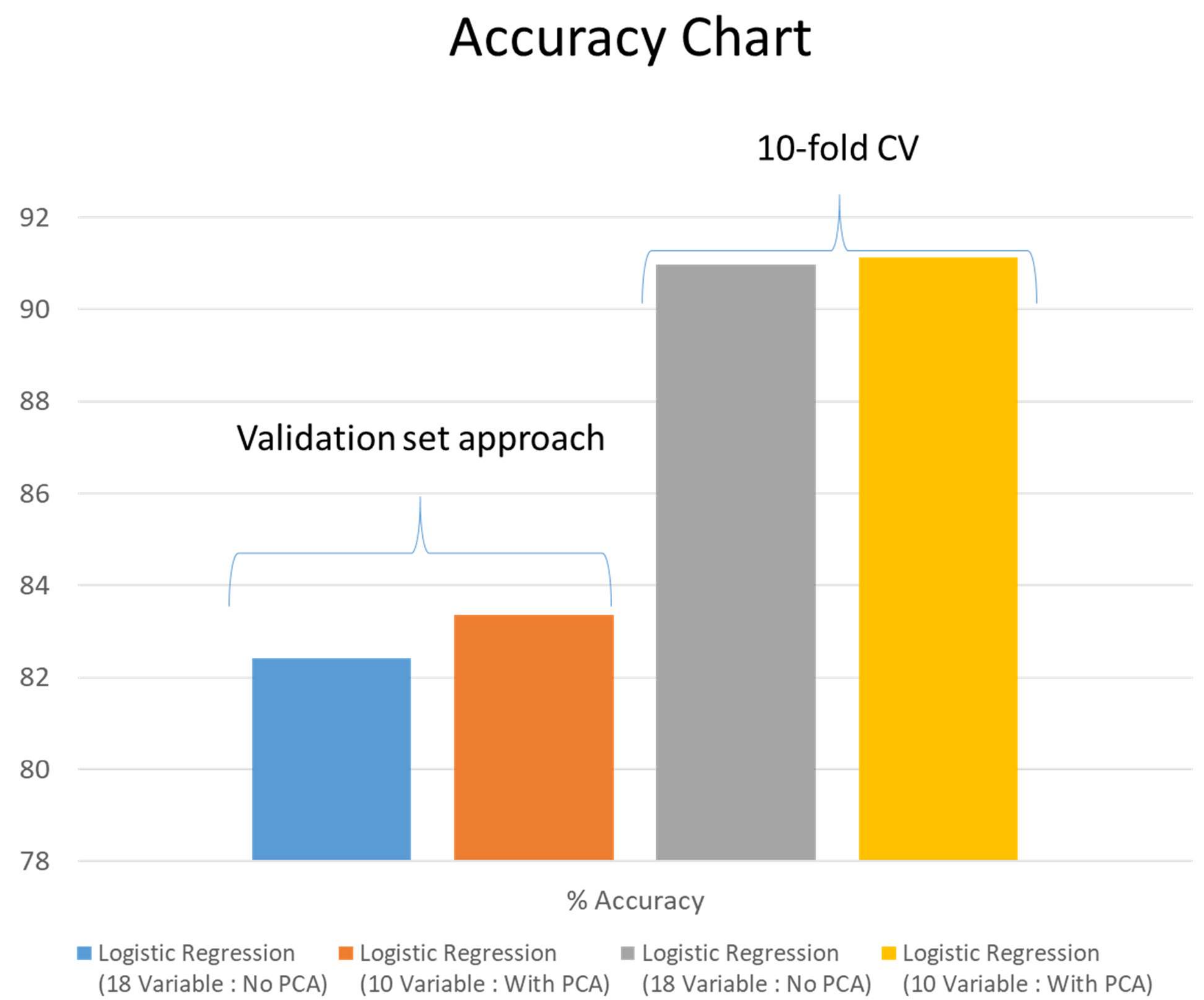
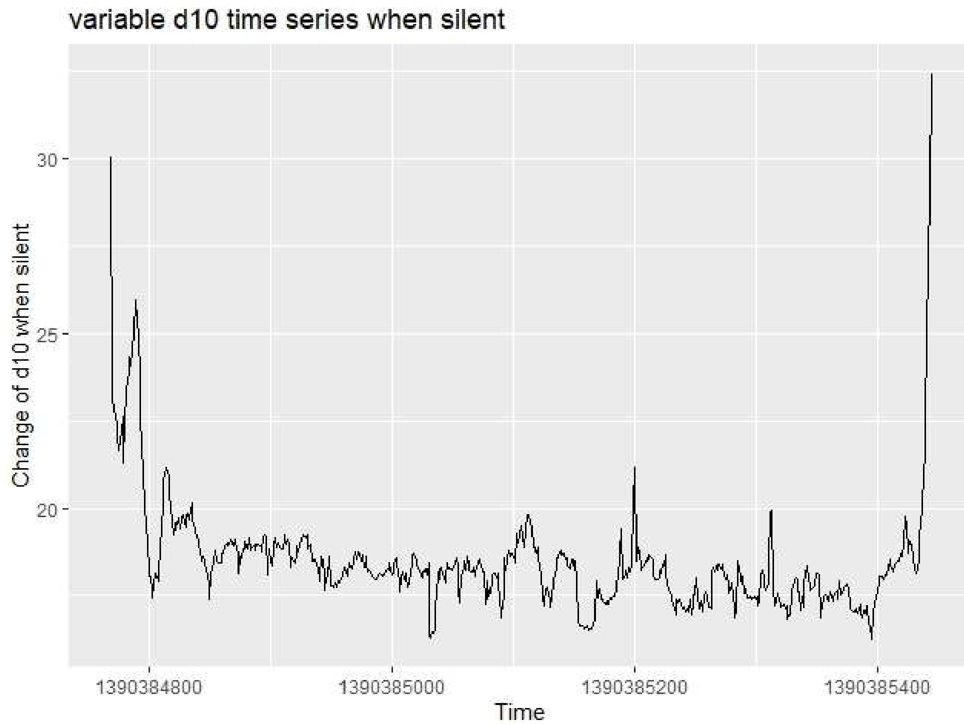
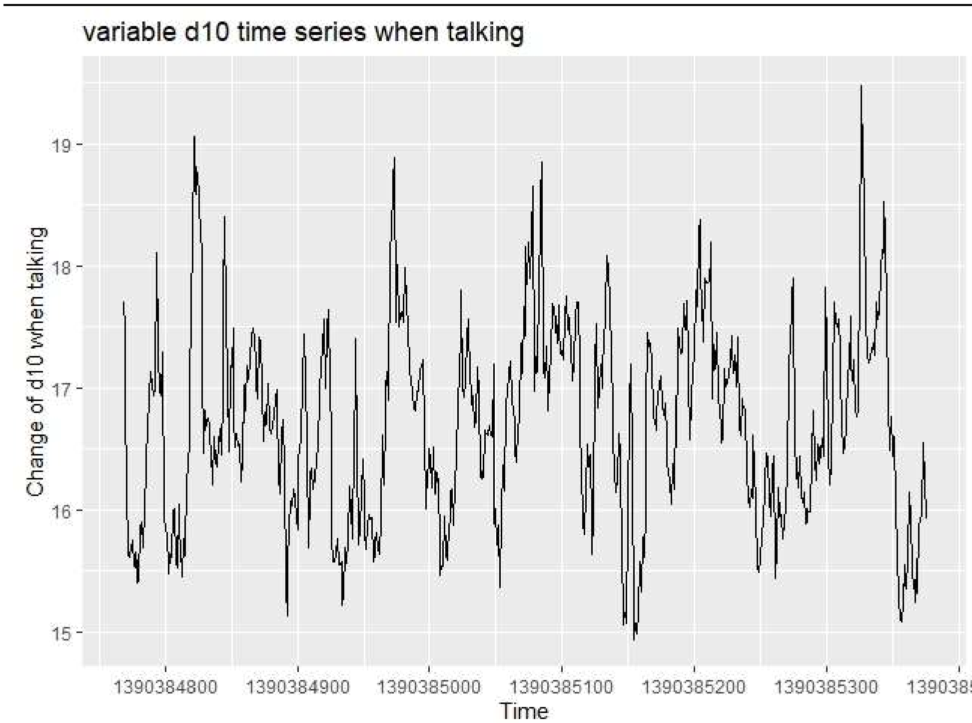
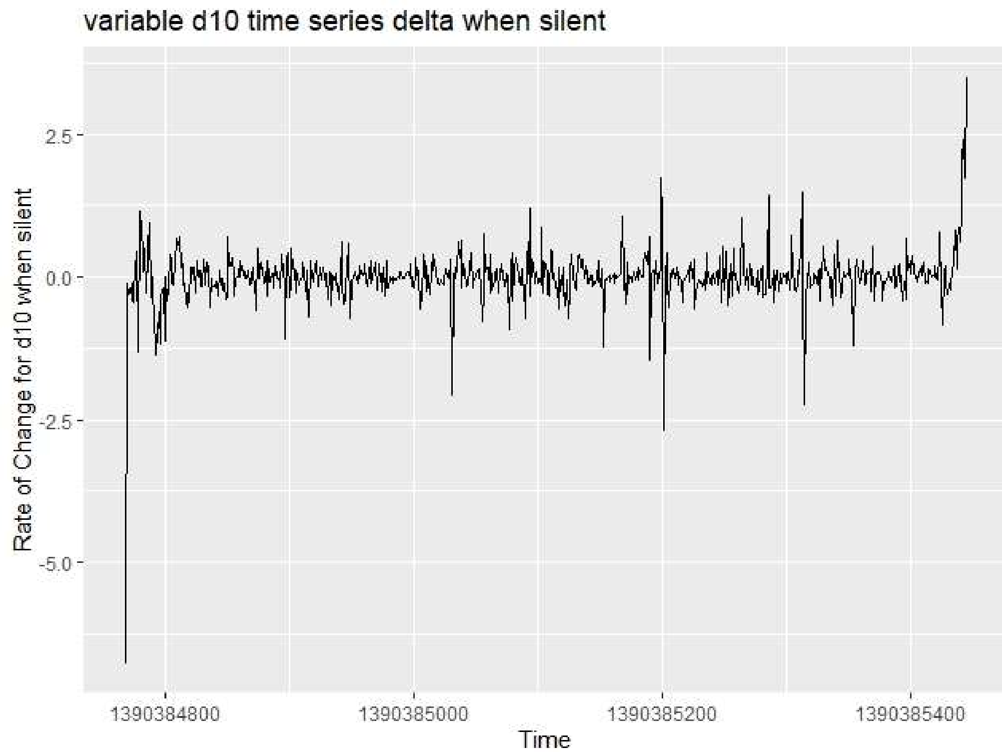


Fig 15: Cross-validation on 10 variables

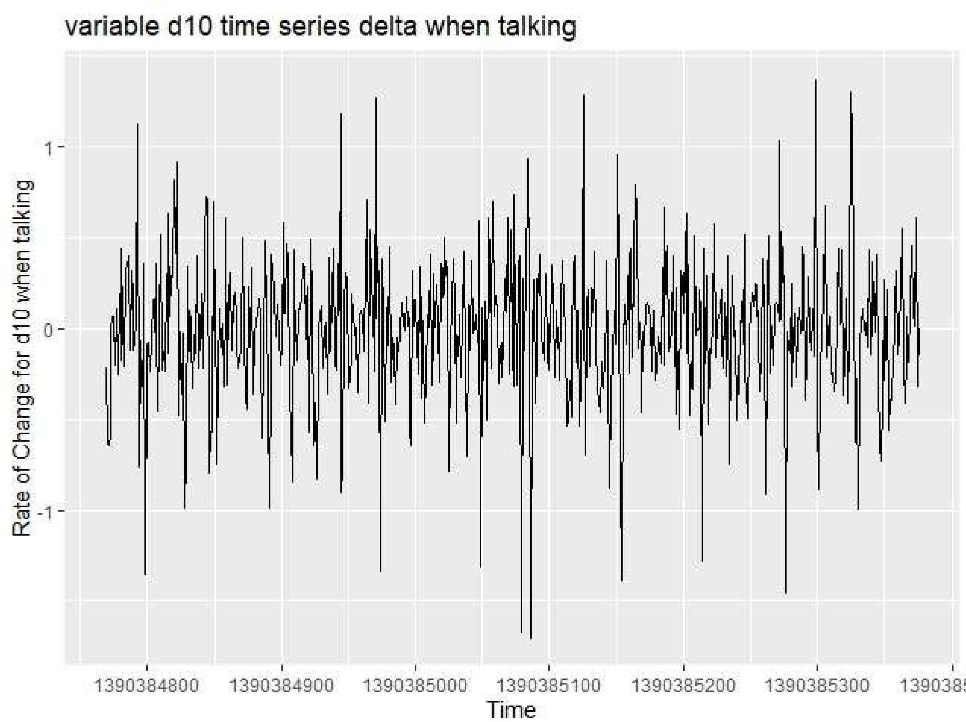
Fig. 16: Time series analysis of d10







variable d10 time series delta when silent
Rate of Change for d10 when silent
2.5
0.0
-2.5
-5.0
1390384800
1390385000
1390385200
1390385400
Time



variable d10 time series delta when talking
Rate of Change for d10 when talking
1
0
-1
1390384800
1390384900
1390385000
1390385100
1390385200
1390385300
1390385
Time

8. Conclusions:

The automated analysis of facial expressions has been widely used in different research areas, such as biometrics or emotional analysis. Most of the advanced works has been carried out in this problem using neural network. But we have only followed a semi-supervised learning approach in this project. We are able to achieve 91% accuracy by just using 10 significant predictors among total 300 predictor in parent data set. PCA has been used as an efficient tool throughout our project for identifying the significant predictors. Hence, we can conclude that, with an optimal combination of Supervised and Unsupervised learning techniques can give an accuracy level near to the neural network. This project should be an excellent starting point for applying classification techniques for solving real life problems.

9. Credits:

[1] http://archive.ics.uci.edu/ml/datasets/Grammatical+Facial+Expressions
[2] FREITAS, F. A.; Peres, S. M.; Lima, C. A. M.; BARBOSA, F. V. "Grammatical Facial Expressions Recognition with Machine Learning", Proceedings of the Twenty-Seventh International Florida Artificial Intelligence Research Society Conference.

[1]

[1] Prepared by Shubhabrata Mukherjee and Sumitra Dey