Technical Summary of Facial Attractiveness Project

PCA:

I ran PCA on these features to see if we really need all these features. The expectation was that we would get a small set of components that upon rotation I could compare with a component for a particular type of facial fetaure. However, it turned out that almost 99% of the variation could be explained by just 1 component. Doing some more reading and spending time thinking about this result made us realize that all these points could be embedded on to a single manifold that upon rotation would collapse to a single linear representation. At this point, I decided to invalidate the use of PCA for our analysis since cutting down to just one feature to explain our variance in the data (which obviously isn't high at all) was not our intention to begin with.

```
combined = np.column_stack((np.asarray(features_x) , np.asarray(features_y)))
```

```
combined.shape
```

```
(500, 136)
```

```
combined
```

```
array([[  85,   89,   98, ...,  606,  611,  607],
       [  88,   86,   89, ...,  495,  496,  494],
       [ 118,  121,  133, ...,  634,  637,  634],
       ...,
       [ 312,  318,  343, ..., 1534, 1537, 1530],
       [ 192,  201,  213, ..., 1097, 1102, 1097],
       [ 181,  187,  208, ...,  930,  934,  931]], dtype=int64)
```

```
from sklearn.decomposition import PCA
```

```
pca = PCA(n_components=2)
```

```
pca.fit(combined)
```

```
PCA(copy=True, iterated_power='auto', n_components=2, random_state=None,
  svd_solver='auto', tol=0.0, whiten=False)
```

```
print(pca.explained_variance_ratio_)
```

```
[ 0.99431833  0.00372245]
```

Our plan was to try to run dimensionality reduction and test if it makes sense to use these reduction techniques before we process the data into a regression model. Since it might not make explanatory sense to reduce the entire data set to just one variable, I envision that we will probably keep the data and feed it into a regression model.

Then, I have done a preliminary linear regression, ridge and lasso regression model as well however, Ryan will go more into details for the ridge and lasso models so I'll not include that here. A preliminary linear regression helps us explain ~50% of the variability in the data as seen below. This is consistent with the results in the publication related to this dataset.

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(combined, Y, test_size=0.33, random_state=42)
```

```
X_train.shape
```

```
(335, 136)
```

```
from sklearn import datasets, linear_model
```

```
from sklearn.metrics import mean_squared_error, r2_score
```

```
regr = linear_model.LinearRegression()
```

```
regr.fit(X_train,y_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

```
y_pred = regr.predict(X_train)
```

```
# The coefficients
print('Coefficients: \n', regr.coef_)
# The mean squared error
print("Mean squared error: %.2f"
      % mean_squared_error(y_train, y_pred))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % r2_score(y_train, y_pred))

# Plot outputs
plt.scatter(X_test, y_test,  color='black')
plt.plot(X_test, y_pred, color='blue', linewidth=3)

plt.xticks(())
plt.yticks(())

plt.show()
```

```
Coefficients:
 [[-0.03284617  0.02901486 -0.00622812  0.05476053 -0.0657278   0.12513919
  -0.1647892   0.08400213 -0.00056747 -0.02802688 -0.0202565   0.02102697
  -0.01424751  0.03276084 -0.05077906  0.06901713 -0.04642176  0.02842491
  -0.06682542  0.07909213  0.00645239 -0.01457146 -0.01454085 -0.01203359
  -0.02614972  0.14420736 -0.04958517 -0.15893938  0.32534808 -0.23791386
   0.15072231  0.11357349 -0.1258242  -0.08112233 -0.09466987  0.0832176
   0.01377636  0.06988641 -0.11081145  0.01407732  0.00242719 -0.0118359
   0.01127598 -0.19906899  0.08834484 -0.06407751  0.01930778  0.11877733
   0.06141545 -0.00673611 -0.11933928 -0.17228017  0.00354433  0.114841
   0.00313741 -0.13280379  0.11653157 -0.00159977  0.10630297  0.03267212
  -0.06819599 -0.12826035  0.21393832  0.13062429  0.00376716 -0.23784209
   0.07326854  0.02226835  0.03963797 -0.06537726 -0.04037247 -0.00221746
   0.13002809 -0.05866689 -0.05287474  0.08921138 -0.02677864 -0.02598202
  -0.03856166  0.12217324 -0.08442543 -0.02639694  0.05381557 -0.01355733
   0.00415977  0.0127093  -0.01102282 -0.02290586  0.00059334  0.02587144
  -0.01428466 -0.02770865  0.057688    0.0505818  -0.07432076  0.00437421
   0.13848209 -0.224472    0.08953067  0.02380393  0.13310847  0.0317529
  -0.07766298 -0.09751175  0.00786785  0.14028074 -0.04785564  0.02610719
   0.01499493 -0.12629069 -0.0555311  -0.19292885  0.07931195 -0.0506167
   0.02838894  0.15966056 -0.04934273  0.05499867 -0.0956298  -0.02994016
   0.1265898  -0.04924849  0.04114478  0.03002002 -0.12041566 -0.05142445
   0.22281485 -0.06103927 -0.04632633  0.08721767 -0.10411409  0.01499757
   0.03416845  0.07447229 -0.07914449 -0.00583482]]
Mean squared error: 0.22
Variance score: 0.49
```

The other team members were researching the above mentioned so decided to test K- Nearest Neighbor Regression. The main idea is that the target is predicted by local interpolation of the targets associated of the nearest neighbors in the training set.

```
from sklearn.neighbors import KNeighborsRegressor
```

```
neigh = KNeighborsRegressor(n_neighbors=2)
```

```
neigh.fit(X_train, y_train)
```

```
KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkowski',
          metric_params=None, n_jobs=1, n_neighbors=2, p=2,
          weights='uniform')
```

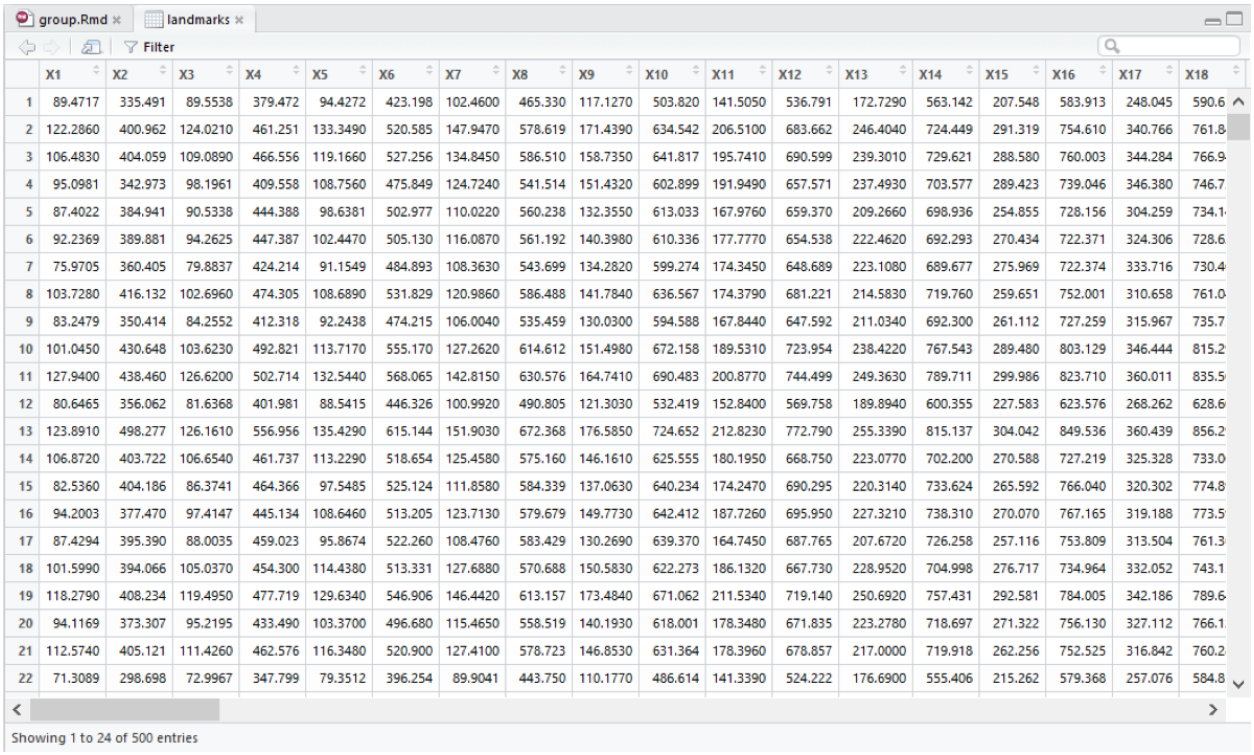
```
pred = (neigh.predict(X_train))
```

upon running a preliminary run in Python, we can see that we can make predictions using the KNN Regressor. The MSE is slightly reduced and the Variance is up by about 5 basis points. We will still go with our results for ridge and lasso regression for better explanation and accuracy.

```
Coefficients:
 [[-0.03284617  0.02901486 -0.00622812  0.05476053 -0.0657278   0.12513919
  -0.1647892   0.08400213 -0.00056747 -0.02802688 -0.0202565   0.02102697
  -0.01424751  0.03276084 -0.05077906  0.06901713 -0.04642176  0.02842491
  -0.06682542  0.07909213  0.00645239 -0.01457146 -0.01454085 -0.01203359
  -0.02614972  0.14420736 -0.04958517 -0.15893938  0.32534808 -0.23791386
   0.15072231  0.11357349 -0.1258242  -0.08112233 -0.09466987  0.0832176
   0.01377636  0.06988641 -0.11081145  0.01407732  0.00242719 -0.0118359
   0.01127598 -0.19906899  0.08834484 -0.06407751  0.01930778  0.11877733
   0.06141545 -0.00673611 -0.11933928 -0.17228017  0.00354433  0.114841
   0.00313741 -0.13280379  0.11653157 -0.00159977  0.10630297  0.03267212
  -0.06819599 -0.12826035  0.21393832  0.13062429  0.00376716 -0.23784209
   0.07326854  0.02226835  0.03963797 -0.06537726 -0.04037247 -0.00221746
   0.13002809 -0.05866689 -0.05287474  0.08921138 -0.02677864 -0.02598202
  -0.03856166  0.12217324 -0.08442543 -0.02639694  0.05381557 -0.01355733
   0.00415977  0.0127093  -0.01102282 -0.02290586  0.00059334  0.02587144
  -0.01428466 -0.02770865  0.057688    0.0505818  -0.07432076  0.00437421
   0.13848209 -0.224472    0.08953067  0.02380393  0.13310847  0.0317529
  -0.07766298 -0.09751175  0.00786785  0.14028074 -0.04785564  0.02610719
   0.01499493 -0.12629069 -0.0555311  -0.19292885  0.07931195 -0.0506167
   0.02838894  0.15966056 -0.04934273  0.05499867 -0.0956298  -0.02994016
   0.1265898  -0.04924849  0.04114478  0.03002002 -0.12041566 -0.05142445
   0.22281485 -0.06103927 -0.04632633  0.08721767 -0.10411409  0.01499757
   0.03416845  0.07447229 -0.07914449 -0.00583482]]
Mean squared error: 0.20
Variance score: 0.53
```

CFA:

Before performing model selection and multiple regression analysis, Principal Component Analysis, Common Factor Analysis and Cluster Analysis were performed on the facial landmarks data. The original data that was used had no definitions of columns. This led to attempting exploratory analyses without understanding the dataset, hence the x1, x2, x3 column names in picture 1. The column names were assigned manually since it represented nothing specific that was given with the data. It was guessed that the physical data were coordinates of some kind of facial structures, since all data were numerical.



group.Rmd | landmarks

Filter

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X14	X15	X16	X17	X18
1	89.4717	335.491	89.5538	379.472	94.4272	423.198	102.4600	465.330	117.1270	503.820	141.5050	536.791	172.7290	563.142	207.548	583.913	248.045	590.6
2	122.2860	400.962	124.0210	461.251	133.3490	520.585	147.9470	578.619	171.4390	634.542	206.5100	683.662	246.4040	724.449	291.319	754.610	340.766	761.8
3	106.4830	404.059	109.0890	466.556	119.1660	527.256	134.8450	586.510	158.7350	641.817	195.7410	690.599	239.3010	729.621	288.580	760.003	344.284	766.9
4	95.0981	342.973	98.1961	409.558	108.7560	475.849	124.7240	541.514	151.4320	602.899	191.9490	657.571	237.4930	703.577	289.423	739.046	346.380	746.7
5	87.4022	384.941	90.5338	444.388	98.6381	502.977	110.0220	560.238	132.3550	613.033	167.9760	659.370	209.2660	698.936	254.855	728.156	304.259	734.1
6	92.2369	389.881	94.2625	447.387	102.4470	505.130	116.0870	561.192	140.3980	610.336	177.7770	654.538	222.4620	692.293	270.434	722.371	324.306	728.6
7	75.9705	360.405	79.8837	424.214	91.1549	484.893	108.3630	543.699	134.2820	599.274	174.3450	648.689	223.1080	689.677	275.969	722.374	333.716	730.4
8	103.7280	416.132	102.6960	474.305	108.6890	531.829	120.9860	586.488	141.7840	636.567	174.3790	681.221	214.5830	719.760	259.651	752.001	310.658	761.0
9	83.2479	350.414	84.2552	412.318	92.2438	474.215	106.0040	535.459	130.0300	594.588	167.8440	647.592	211.0340	692.300	261.112	727.259	315.967	735.7
10	101.0450	430.648	103.6230	492.821	113.7170	555.170	127.2620	614.612	151.4980	672.158	189.5310	723.954	238.4220	767.543	289.480	803.129	346.444	815.2
11	127.9400	438.460	126.6200	502.714	132.5440	568.065	142.8150	630.576	164.7410	690.483	200.8770	744.499	249.3630	789.711	299.986	823.710	360.011	835.5
12	80.6465	356.062	81.6368	401.981	88.5415	446.326	100.9920	490.805	121.3030	532.419	152.8400	569.758	189.8940	600.355	227.583	623.576	268.262	628.6
13	123.8910	498.277	126.1610	556.956	135.4290	615.144	151.9030	672.368	176.5850	724.652	212.8230	772.790	255.3390	815.137	304.042	849.536	360.439	856.2
14	106.8720	403.722	106.6540	461.737	113.2290	518.654	125.4580	575.160	146.1610	625.555	180.1950	668.750	223.0770	702.200	270.588	727.219	325.328	733.0
15	82.5360	404.186	86.3741	464.366	97.5485	525.124	111.8580	584.339	137.0630	640.234	174.2470	690.295	220.3140	733.624	265.592	766.040	320.302	774.8
16	94.2003	377.470	97.4147	445.134	108.6460	513.205	123.7130	579.679	149.7730	642.412	187.7260	695.950	227.3210	738.310	270.070	767.165	319.188	773.5
17	87.4294	395.390	88.0035	459.023	95.8674	522.260	108.4760	583.429	130.2690	639.370	164.7450	687.765	207.6720	726.258	257.116	753.809	313.504	761.3
18	101.5990	394.066	105.0370	454.300	114.4380	513.331	127.6880	570.688	150.5830	622.273	186.1320	667.730	228.9520	704.998	276.717	734.964	332.052	743.1
19	118.2790	408.234	119.4950	477.719	129.6340	546.906	146.4420	613.157	173.4840	671.062	211.5340	719.140	250.6920	757.431	292.581	784.005	342.186	789.6
20	94.1169	373.307	95.2195	433.490	103.3700	496.680	115.4650	558.519	140.1930	618.001	178.3480	671.835	223.2780	718.697	271.322	756.130	327.112	766.1
21	112.5740	405.121	111.4260	462.576	116.3480	520.900	127.4100	578.723	146.8530	631.364	178.3960	678.857	217.0000	719.918	262.256	752.525	316.842	760.2
22	71.3089	298.698	72.9967	347.799	79.3512	396.254	89.9041	443.750	110.1770	486.614	141.3390	524.222	176.6900	555.406	215.262	579.368	257.076	584.8

Showing 1 to 24 of 500 entries

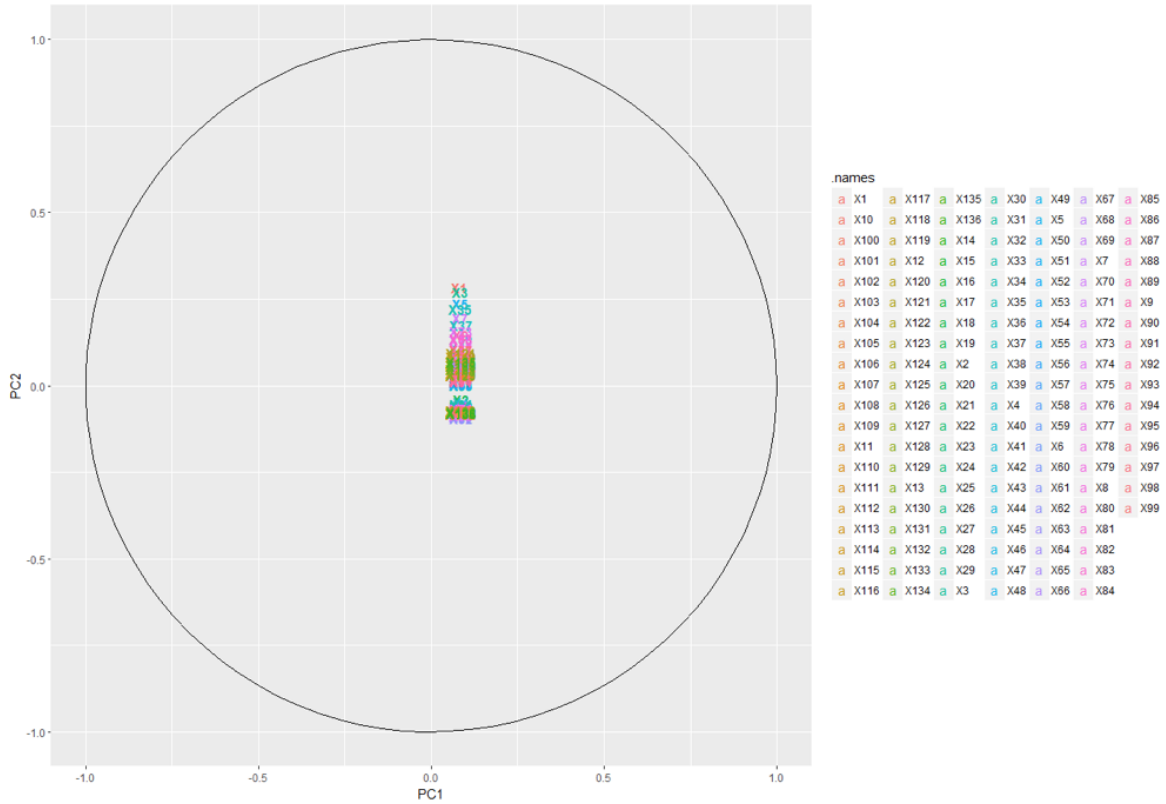
Picture 1: A view of the original dataset

The data ranged anywhere from a low twenties up to more than five hundred. Due to the diverse range of the numerical variables, it was necessary to apply scaling in the Principal Component Analysis. All Principal Component Analyses and rotations used on the facial landmarks data were performed with correlations and not covariances. The output from the Principal Component Analysis demonstrated that it did not have anything significant. The first principal component explained 98.9% of the cumulative proportion in the dataset in output 1. It was logical to conclude that the Principal Component Analysis suggested using only one principal component. In the Principal Component Analysis plot and the bi plot, picture 2 and 3, it was obvious that the entire data showed an extremely linear relationship with each other. The screeplot also recommended to use only one principal component, since the first value was extremely high, followed by low values after in picture 4.

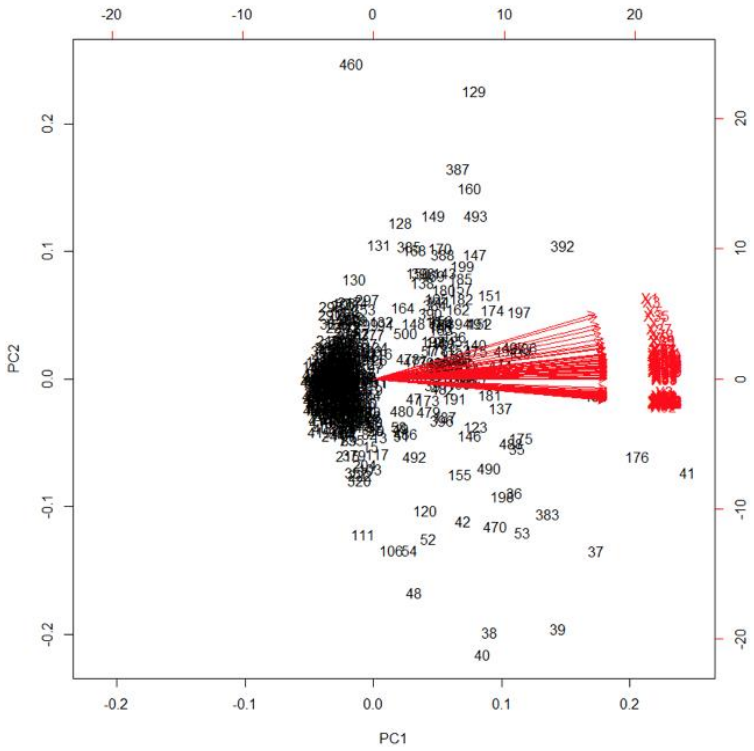
Console R Markdown

```
> summary(p)
Importance of components%s:
                          PC1    PC2     PC3     PC4     PC5     PC6     PC7     PC8     PC9    PC10    PC11    PC12    PC13    PC14
Standard deviation     11.598 0.9899 0.50659 0.27003 0.2339 0.17796 0.15211 0.1184 0.11307 0.09473 0.08074 0.07455 0.06870 0.06744
Proportion of Variance  0.989 0.0072 0.00189 0.00054 0.0004 0.00023 0.00017 0.0001 0.00009 0.00007 0.00005 0.00004 0.00003 0.00003
Cumulative Proportion   0.989 0.9962 0.99812 0.99866 0.9991 0.99929 0.99946 0.9996 0.99966 0.99973 0.99977 0.99982 0.99985 0.99988
                          PC15    PC16    PC17    PC18    PC19    PC20    PC21    PC22    PC23    PC24    PC25    PC26    PC27
Standard deviation     0.05020 0.04728 0.04377 0.04227 0.03735 0.03276 0.02952 0.02868 0.02739 0.02557 0.02308 0.01941 0.01648
Proportion of Variance 0.00002 0.00002 0.00001 0.00001 0.00001 0.00001 0.00001 0.00001 0.00001 0.00000 0.00000 0.00000 0.00000
Cumulative Proportion  0.99990 0.99992 0.99993 0.99995 0.99996 0.99996 0.99997 0.99998 0.99998 0.99999 0.99999 0.99999 1.00000
                          PC28    PC29     PC30     PC31     PC32    PC33     PC34     PC35     PC36     PC37    PC38     PC39
Standard deviation     0.01331 0.01185 0.009114 0.007273 0.005367 0.00523 0.004677 0.004349 0.003468 0.003173 0.00297 0.002744
Proportion of Variance 0.00000 0.00000 0.000000 0.000000 0.000000 0.00000 0.000000 0.000000 0.000000 0.000000 0.00000 0.000000
Cumulative Proportion  1.00000 1.00000 1.000000 1.000000 1.000000 1.00000 1.000000 1.000000 1.000000 1.000000 1.00000 1.000000
                          PC40     PC41  PC42     PC43     PC44     PC45     PC46     PC47     PC48     PC49     PC50      PC51
Standard deviation     0.002577 0.002156 0.002 0.001817 0.00173 0.001536 0.001357 0.001282 0.001253 0.001217 0.001116 0.0009568
Proportion of Variance 0.000000 0.000000 0.000 0.000000 0.00000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.0000000
Cumulative Proportion  1.000000 1.000000 1.000 1.000000 1.00000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.0000000
                          PC52      PC53       PC54       PC55       PC56       PC57       PC58      PC59       PC60       PC61       PC62
Standard deviation     0.0009065 0.000819 0.0007803 0.0006635 0.0006413 0.0006146 0.0005816 0.000543 0.0005067 0.000495 0.0004901
Proportion of Variance 0.0000000 0.000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.000000 0.0000000 0.000000 0.0000000
Cumulative Proportion  1.0000000 1.000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.000000 1.0000000 1.000000 1.0000000
                          PC63       PC64       PC65       PC66       PC67       PC68       PC69       PC70       PC71       PC72       PC73
Standard deviation     0.0004749 0.0004526 0.0004165 0.0003898 0.0003505 0.0003391 0.0003314 0.0003019 0.0002746 0.00027 0.0002551
Proportion of Variance 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.00000 0.0000000
Cumulative Proportion  1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.00000 1.0000000
                          PC74       PC75       PC76       PC77       PC78       PC79       PC80       PC81       PC82       PC83
Standard deviation     0.0002534 0.0002265 0.0002114 0.0002073 0.0002007 0.0001851 0.0001676 0.0001561 0.0001535 0.0001423
Proportion of Variance 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
Cumulative Proportion  1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
                          PC84       PC85       PC86       PC87       PC88       PC89       PC90       PC91       PC92       PC93
Standard deviation     0.0001395 0.0001317 0.0001259 0.0001103 0.0001033 9.004e-05 8.875e-05 7.901e-05 7.419e-05 7.115e-05
Proportion of Variance 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00
Cumulative Proportion  1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.000e+00 1.000e+00 1.000e+00 1.000e+00 1.000e+00
                          PC94       PC95       PC96       PC97       PC98       PC99      PC100      PC101      PC102      PC103
Standard deviation     6.739e-05 6.265e-05 5.794e-05 5.029e-05 4.746e-05 4.184e-05 3.783e-05 3.384e-05 3.035e-05 2.782e-05
Proportion of Variance 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00
Cumulative Proportion  1.000e+00 1.000e+00 1.000e+00 1.000e+00 1.000e+00 1.000e+00 1.000e+00 1.000e+00 1.000e+00 1.000e+00
```

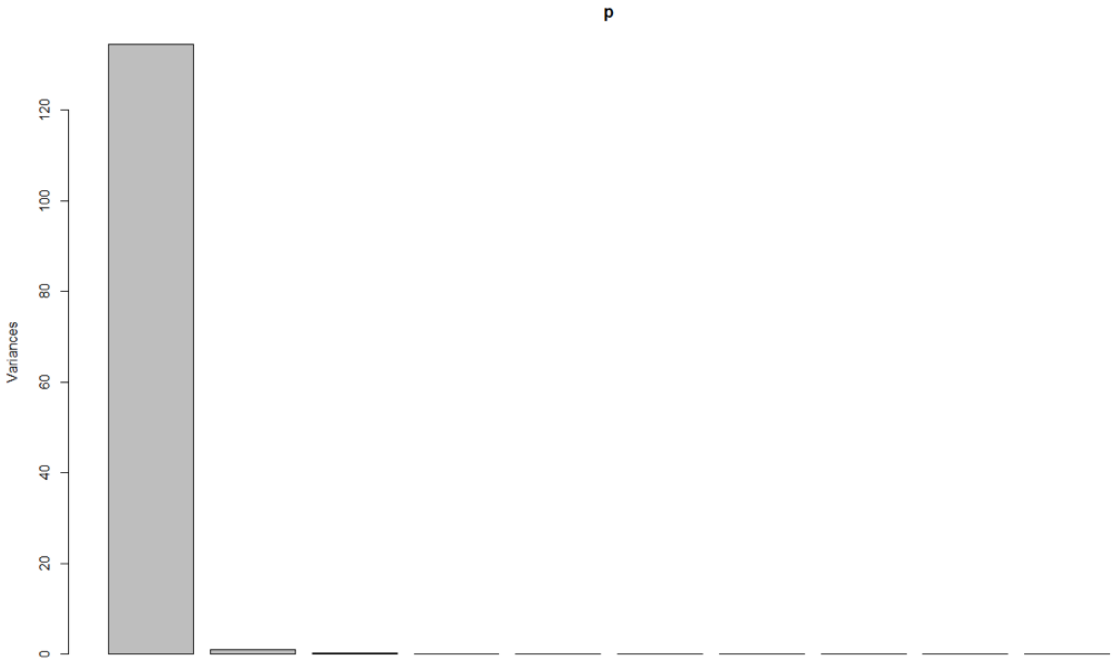
Output 1: Output from the Principal Component Analysis (original data)



Picture 2: Principal Component Analysis plot (original data)



Picture 3: Bi plot (original data)



Picture 4: Scree plot (original data)

After checking the insignificant results from the scaled Principal Component Analysis, rotation with “Varimax” was performed. In output 2, the rotation loadings were very close to 1. All loading values were positive as well. The first principal component explained 98.9 percent of the proportion variance in the data. This supported that only one principal component was recommended.

```
Loadings:
  [1] 0.947 0.995 0.953 0.997 0.963 0.997 0.972 0.997 0.982 0.997 0.990 0.997 0.994 0.997 0.996 0.996 0.997 0.996 0.997 0.997 0.997
 [22] 0.997 0.997 0.997 0.997 0.997 0.997 0.997 0.998 0.997 0.998 0.997 0.998 0.995 0.970 0.994 0.981 0.992 0.989 0.990 0.992 0.991
 [43] 0.994 0.992 0.997 0.992 0.998 0.990 0.998 0.990 0.998 0.991 0.998 0.993 0.997 0.996 0.997 0.996 0.997 0.995 0.997 0.995 0.996
 [64] 0.996 0.997 0.996 0.997 0.996 0.998 0.996 0.998 0.996 0.986 0.996 0.990 0.996 0.994 0.996 0.996 0.996 0.994 0.996 0.990 0.996
 [85] 0.998 0.996 0.998 0.996 0.998 0.996 0.998 0.996 0.998 0.996 0.998 0.996 0.993 0.997 0.996 0.997 0.997 0.996 0.998 0.996 0.998
[106] 0.996 0.998 0.997 0.998 0.997 0.998 0.997 0.998 0.997 0.997 0.997 0.997 0.995 0.997 0.994 0.997 0.997 0.997 0.998 0.996
[127] 0.998 0.997 0.998 0.997 0.998 0.997 0.997 0.997 0.997 0.997

                 PC1
SS loadings    134.508
Proportion Var   0.989
```

Output 2: Loadings output from the scaled rotation w/ “Varimax” (original data)

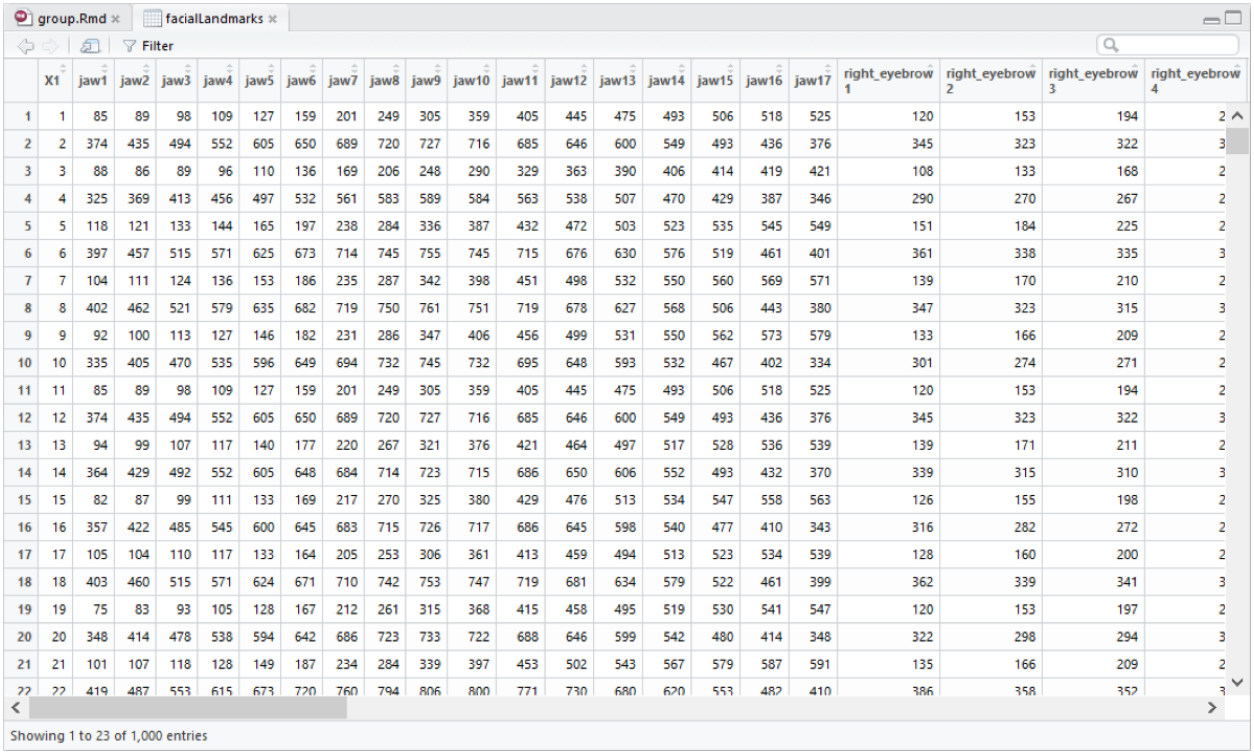
Common Factor Analysis was utilized, following the unsatisfying Principal Component Analysis and rotation. In output 3, nfactor of 1 and 2 were tried, but R gave an error stating “unable to optimize from this starting value.” Singularity issues might have caused this error, since the relationship between the variables were extremely linear. The lowest nfactor was selected, which was 3. The output of the Common Factor Analysis illustrated that 2 factors were ideal for the facial data.

```
X15   0.691   0.713
X17   0.700   0.703
X35   0.576   0.813
X37   0.613   0.785
X39   0.642   0.760
X41   0.661   0.741
X43   0.675   0.726
X55   0.698   0.706
X57   0.697   0.707
X59   0.696   0.707
X61   0.695   0.708
X63   0.683   0.722
X65   0.690   0.714
X67   0.696   0.707
X73   0.628   0.774
X75   0.647   0.758
X77   0.667   0.739
X79   0.678   0.727
X81   0.665   0.741
X83   0.644   0.760
X97   0.668   0.736
X99   0.682   0.722
X101  0.692   0.712
X103  0.698   0.706
X115  0.698   0.705
X117  0.691   0.713
X119  0.681   0.723
X121  0.674   0.731
X123  0.693   0.712
X125  0.698   0.705
X133  0.698   0.705
X135  0.692   0.712

                Factor1 Factor2 Factor3
SS loadings      74.258  60.138   1.325
Proportion Var    0.546   0.442   0.010
Cumulative Var    0.546   0.988   0.998
```

Output 3: Output from Common Factor Analysis (original data)

Since the original data did not have any given columns, further analysis was impossible. One of the team member, Ryan, processed the original data in python using a code from the developers' resource page in order to gather column names and details of the data. After the data processing, the newly processed data finally had specified column names in picture 5.



group.Rmd × facialLandmarks ×

Filter

	X1	jaw1	jaw2	jaw3	jaw4	jaw5	jaw6	jaw7	jaw8	jaw9	jaw10	jaw11	jaw12	jaw13	jaw14	jaw15	jaw16	jaw17	right_eyebrow 1	right_eyebrow 2	right_eyebrow 3	right_eyebrow 4
1	1	85	89	98	109	127	159	201	249	305	359	405	445	475	493	506	518	525	120	153	194	2
2	2	374	435	494	552	605	650	689	720	727	716	685	646	600	549	493	436	376	345	323	322	3
3	3	88	86	89	96	110	136	169	206	248	290	329	363	390	406	414	419	421	108	133	168	2
4	4	325	369	413	456	497	532	561	583	589	584	563	538	507	470	429	387	346	290	270	267	2
5	5	118	121	133	144	165	197	238	284	336	387	432	472	503	523	535	545	549	151	184	225	2
6	6	397	457	515	571	625	673	714	745	755	745	715	676	630	576	519	461	401	361	338	335	3
7	7	104	111	124	136	153	186	235	287	342	398	451	498	532	550	560	569	571	139	170	210	2
8	8	402	462	521	579	635	682	719	750	761	751	719	678	627	568	506	443	380	347	323	315	3
9	9	92	100	113	127	146	182	231	286	347	406	456	499	531	550	562	573	579	133	166	209	2
10	10	335	405	470	535	596	649	694	732	745	732	695	648	593	532	467	402	334	301	274	271	2
11	11	85	89	98	109	127	159	201	249	305	359	405	445	475	493	506	518	525	120	153	194	2
12	12	374	435	494	552	605	650	689	720	727	716	685	646	600	549	493	436	376	345	323	322	3
13	13	94	99	107	117	140	177	220	267	321	376	421	464	497	517	528	536	539	139	171	211	2
14	14	364	429	492	552	605	648	684	714	723	715	686	650	606	552	493	432	370	339	315	310	3
15	15	82	87	99	111	133	169	217	270	325	380	429	476	513	534	547	558	563	126	155	198	2
16	16	357	422	485	545	600	645	683	715	726	717	686	645	598	540	477	410	343	316	282	272	2
17	17	105	104	110	117	133	164	205	253	306	361	413	459	494	513	523	534	539	128	160	200	2
18	18	403	460	515	571	624	671	710	742	753	747	719	681	634	579	522	461	399	362	339	341	3
19	19	75	83	93	105	128	167	212	261	315	368	415	458	495	519	530	541	547	120	153	197	2
20	20	348	414	478	538	594	642	686	723	733	722	688	646	599	542	480	414	348	322	298	294	3
21	21	101	107	118	128	149	187	234	284	339	397	453	502	543	567	579	587	591	135	166	209	2
22	22	419	487	553	615	673	720	760	794	806	800	771	730	680	620	553	482	410	386	358	352	3

Showing 1 to 23 of 1,000 entries

Picture 5: A view of the new processed dataset

Unfortunately, the odd rows had the coordinates for x-values, and the even rows had the coordinates for y-values. Therefore manual division into two datasets(x and y) in R was necessary. Although the physical numerical values changed during the python process, the two datasets are essentially the same data. Due to the ranges of the data values, scaling was used for the Principal Component Analysis. In output 4, it portrayed that the first principal component explained 99.2 percent of the cumulative proportion of the data for the x's. In output 5, it portrayed that the first principal component explained 99.7 percent of the cumulative proportion of the data for the y's. Both the x and y Principal Component Analysis outputs recommended only 1 principal components. Similar to the original dataset, the Principal Component Analysis did not show any great results.

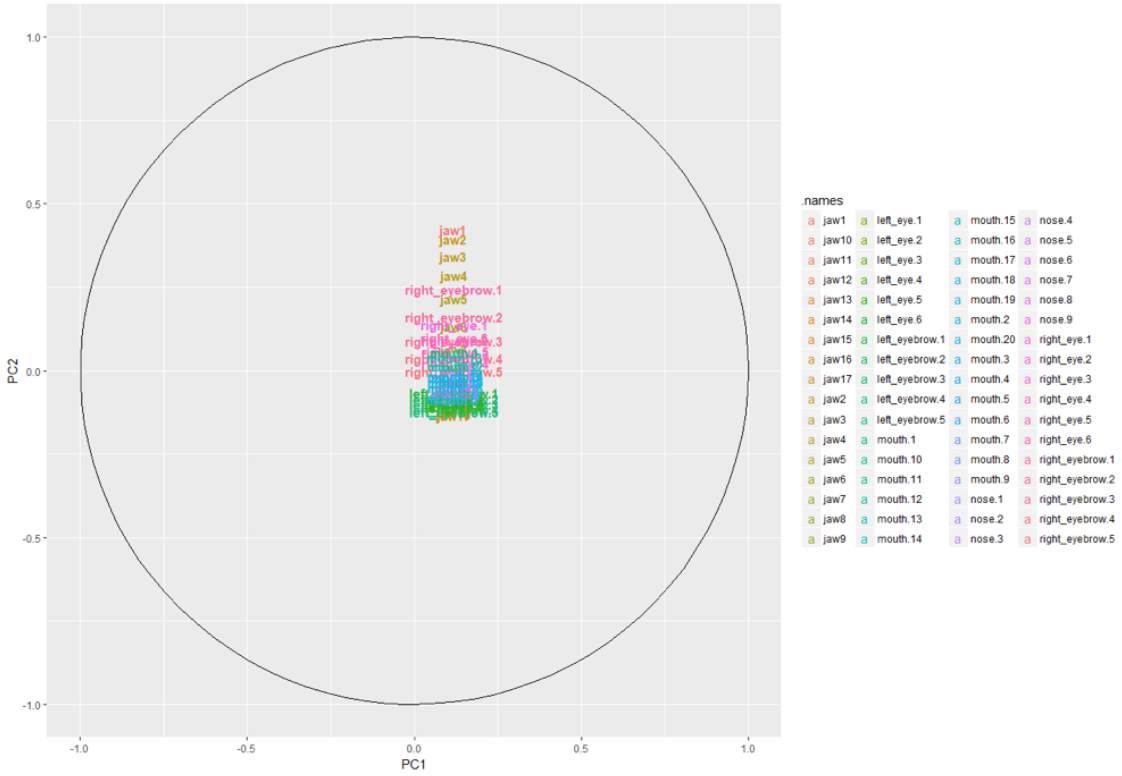
```
Importance of components%s:
                          PC1     PC2     PC3     PC4     PC5     PC6     PC7     PC8     PC9    PC10    PC11    PC12    PC13
Standard deviation     8.2151 0.60722 0.27500 0.15905 0.12210 0.09907 0.07232 0.05923 0.04597 0.04117 0.03315 0.02527 0.02440
Proportion of Variance 0.9925 0.00542 0.00111 0.00037 0.00022 0.00014 0.00008 0.00005 0.00003 0.00002 0.00002 0.00001 0.00001
Cumulative Proportion  0.9925 0.99788 0.99900 0.99937 0.99959 0.99973 0.99981 0.99986 0.99989 0.99992 0.99993 0.99994 0.99995
                          PC14    PC15    PC16    PC17    PC18    PC19    PC20    PC21    PC22    PC23    PC24    PC25    PC26
Standard deviation     0.02255 0.02109 0.01978 0.01796 0.0155 0.01432 0.01335 0.01148 0.01131 0.009883 0.009047 0.008272 0.007834
Proportion of Variance 0.00001 0.00001 0.00001 0.00000 0.0000 0.00000 0.00000 0.00000 0.00000 0.000000 0.000000 0.000000 0.000000
Cumulative Proportion  0.99996 0.99996 0.99997 0.99998 1.0000 0.99998 0.99998 0.99999 0.99999 0.999990 0.999990 0.999990 0.999990
                          PC27     PC28     PC29     PC30     PC31     PC32     PC33     PC34     PC35    PC36     PC37     PC38
Standard deviation     0.007224 0.007066 0.006155 0.005783 0.005587 0.005227 0.004991 0.004795 0.004543 0.00442 0.004025 0.003768
Proportion of Variance 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.00000 0.000000 0.000000
Cumulative Proportion  0.999990 0.999990 0.999990 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.00000 1.000000 1.000000
                          PC39     PC40     PC41     PC42     PC43     PC44     PC45     PC46     PC47     PC48     PC49     PC50
Standard deviation     0.003435 0.003323 0.003171 0.002925 0.002842 0.002641 0.002588 0.002345 0.002263 0.002177 0.002114 0.002036
Proportion of Variance 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
Cumulative Proportion  1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000
                          PC51     PC52     PC53    PC54     PC55    PC56    PC57     PC58     PC59     PC60     PC61     PC62
Standard deviation     0.001967 0.001941 0.001883 0.00178 0.001688 0.00161 0.00157 0.001559 0.001529 0.001479 0.001443 0.001389
Proportion of Variance 0.000000 0.000000 0.000000 0.00000 0.000000 0.00000 0.00000 0.000000 0.000000 0.000000 0.000000 0.000000
Cumulative Proportion  1.000000 1.000000 1.000000 1.00000 1.000000 1.00000 1.00000 1.000000 1.000000 1.000000 1.000000 1.000000
                          PC63     PC64     PC65     PC66     PC67     PC68
Standard deviation     0.001368 0.001311 0.001291 0.001276 0.001248 0.001165
Proportion of Variance 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
Cumulative Proportion  1.000000 1.000000 1.000000 1.000000 1.000000 1.000000
```

Output 4: Output from the Principal Component Analysis for x (processed data)

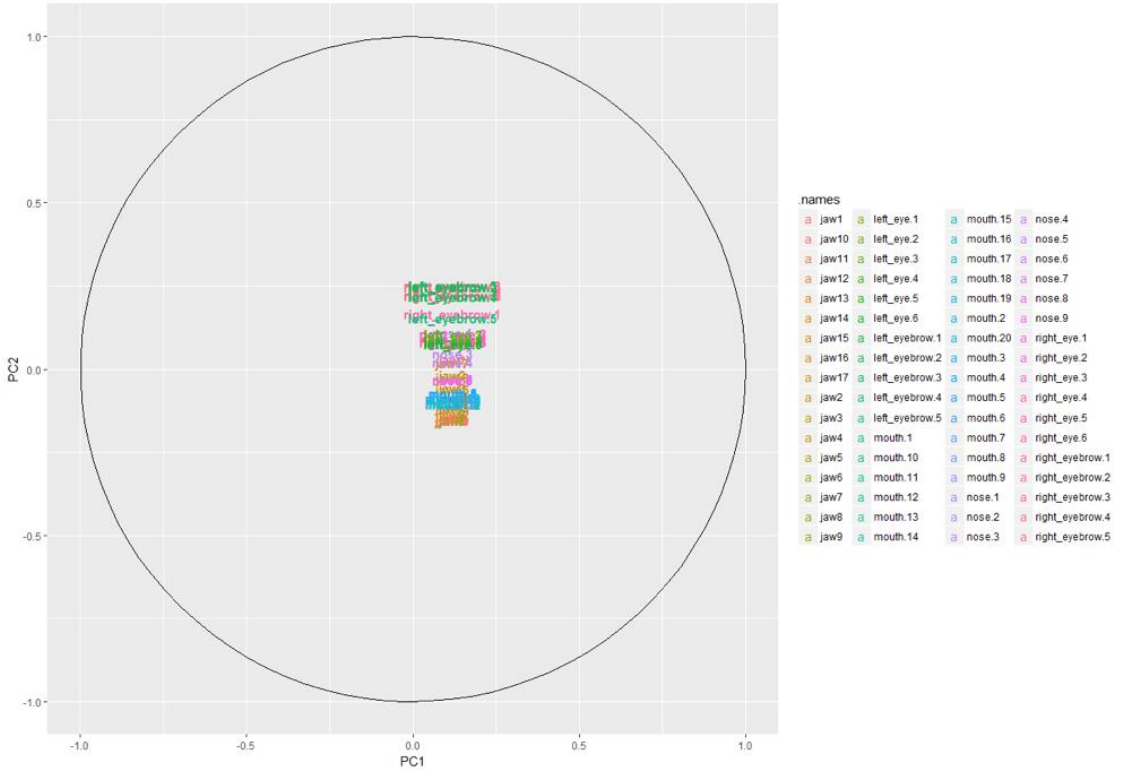
```
Importance of components%s:
                          PC1     PC2     PC3     PC4     PC5     PC6     PC7     PC8     PC9    PC10    PC11    PC12    PC13
Standard deviation     8.2365 0.29800 0.17956 0.14775 0.06541 0.05570 0.04975 0.04427 0.03787 0.03245 0.02524 0.02169 0.01934
Proportion of Variance 0.9976 0.00131 0.00047 0.00032 0.00006 0.00005 0.00004 0.00003 0.00002 0.00002 0.00001 0.00001 0.00001
Cumulative Proportion  0.9976 0.99894 0.99942 0.99974 0.99980 0.99985 0.99988 0.99991 0.99993 0.99995 0.99996 0.99997 0.99997
                          PC14    PC15   PC16    PC17    PC18    PC19   PC20    PC21    PC22    PC23    PC24    PC25    PC26
Standard deviation     0.01814 0.01684 0.0139 0.01308 0.01186 0.01043 0.0104 0.009601 0.007767 0.007318 0.007137 0.006707 0.006262
Proportion of Variance 0.00000 0.00000 0.0000 0.00000 0.00000 0.00000 0.0000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
Cumulative Proportion  0.99998 0.99998 1.0000 0.99999 0.99999 0.99999 1.0000 0.999990 0.999990 0.999990 1.000000 1.000000 1.000000
                          PC27     PC28     PC29     PC30     PC31     PC32     PC33     PC34     PC35     PC36     PC37    PC38
Standard deviation     0.005524 0.004789 0.004066 0.003715 0.003568 0.003381 0.003351 0.003149 0.00297 0.002832 0.002563 0.00243
Proportion of Variance 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.00000 0.000000 0.000000 0.00000
Cumulative Proportion  1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.00000 1.000000 1.000000 1.00000
                          PC39     PC40     PC41     PC42     PC43     PC44     PC45     PC46     PC47     PC48     PC49     PC50
Standard deviation     0.002344 0.002268 0.00217 0.002065 0.001899 0.001862 0.001781 0.001633 0.001604 0.00155 0.001528 0.001483
Proportion of Variance 0.000000 0.000000 0.00000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.00000 0.000000 0.000000
Cumulative Proportion  1.000000 1.000000 1.00000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.00000 1.000000 1.000000
                          PC51     PC52     PC53     PC54     PC55     PC56     PC57     PC58     PC59     PC60      PC61      PC62
Standard deviation     0.00138 0.001363 0.001317 0.001246 0.001208 0.001183 0.001127 0.001111 0.001084 0.001045 0.0009981 0.0009681
Proportion of Variance 0.00000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.0000000 0.0000000
Cumulative Proportion  1.00000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.0000000 1.0000000
                          PC63       PC64       PC65       PC66       PC67       PC68
Standard deviation     0.0009478 0.0008826 0.0008522 0.0008373 0.000781 0.0007489
Proportion of Variance 0.0000000 0.0000000 0.0000000 0.0000000 0.000000 0.0000000
Cumulative Proportion  1.0000000 1.0000000 1.0000000 1.0000000 1.000000 1.0000000
```

Output 5: Output from the Principal Component Analysis for y (processed data)

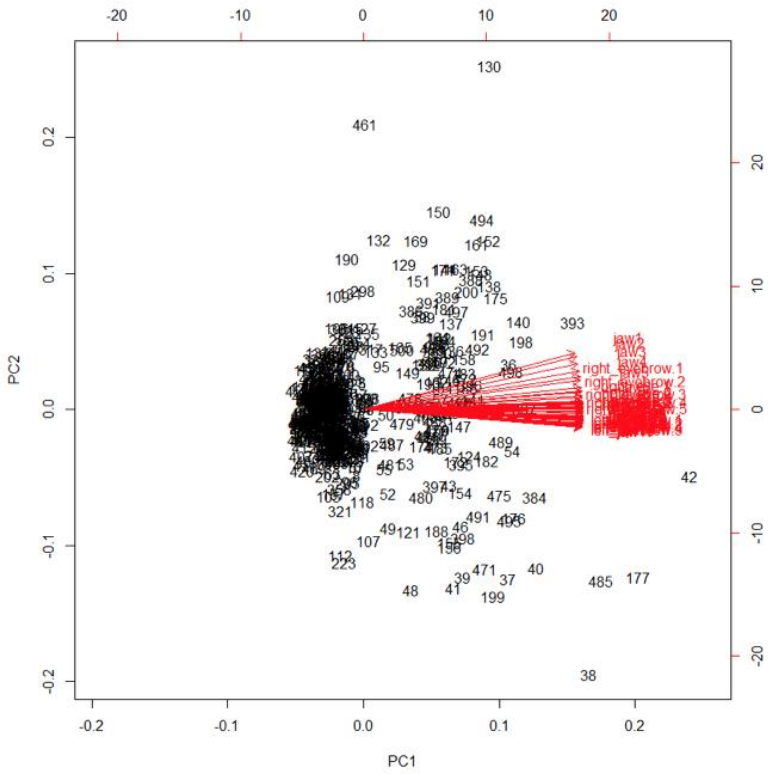
In the Principal Component Analysis plot and the bi plot for both x and y, picture 6, 7, 8 and 9, it was clear that the variables in the facial data had very strongly linear relationships. The screeplots for both x and y also suggested to use only one principal component. The first value had the highest value, followed by low values for both x and y in picture 10 and 11. This was expected in a way, since the original data had similar analyses outputs.



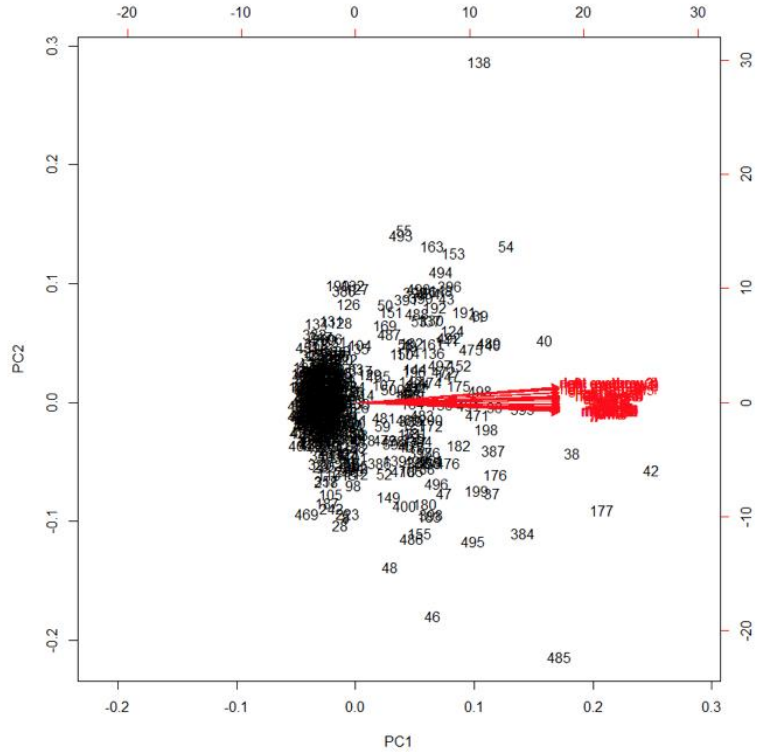
Picture 6: Principal Component Analysis plot for x (processed data)



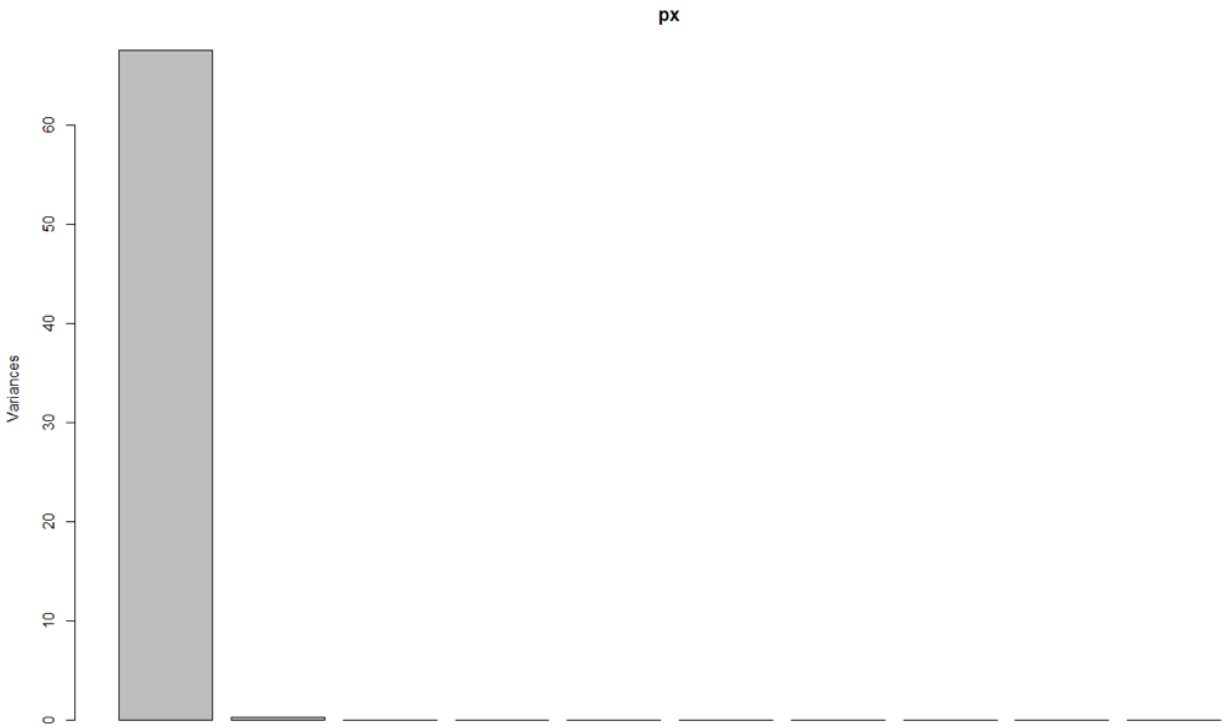
Picture 7: Principal Component Analysis plot for y (processed data)



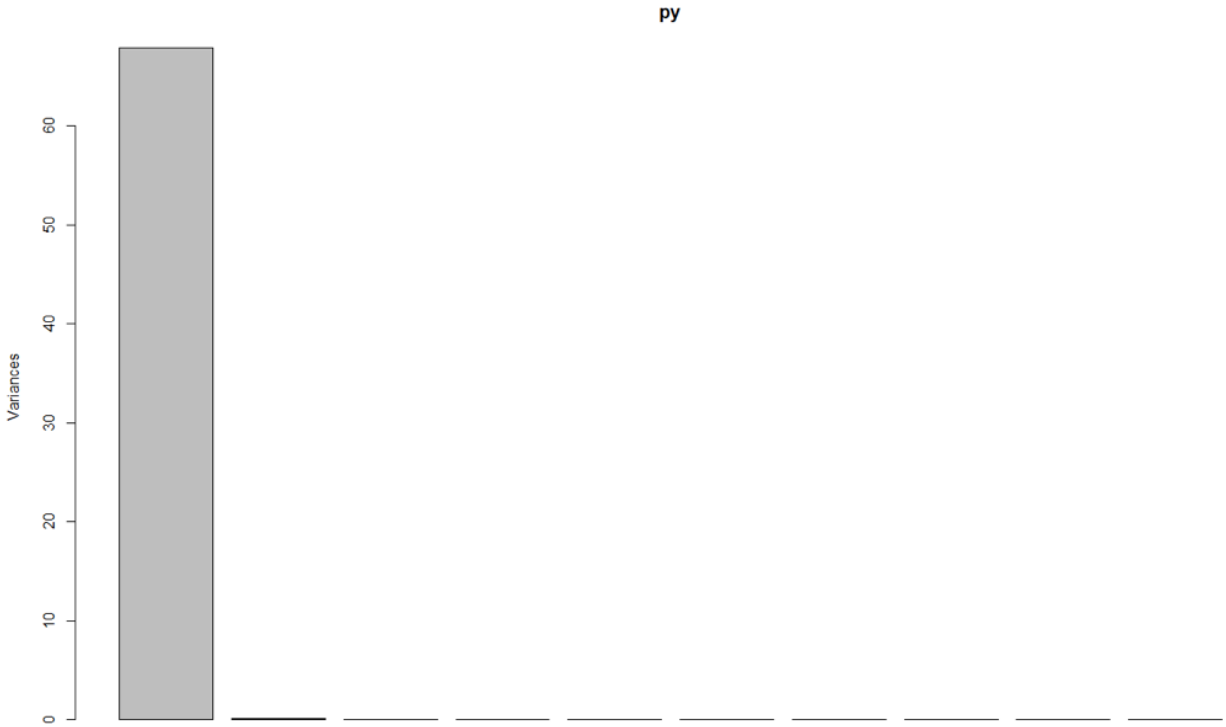
Picture 8: Bi plot for x (processed data)



Picture 9: Bi plot for y (processed data)



Picture 10: Scree plot for x (processed data)



Picture 11: Scree plot for y (processed data)

Rotation with “Varimax” was utilized in order to hopefully find a different perspective of the data. However, all the loading values were extremely close to 1 for both x and y. All the loadings were positive as well for both x and y in output 6 and 7. The first principal component explained 99.2 percent and 99.8 percent of the proportion variance for x and y respectively. The root mean square of the residuals were also almost 0 for both x and y. This suggests strong linear relationships between the variables.

```
Loadings:
 [1] 0.963 0.970 0.977 0.982 0.987 0.993 0.997 0.998 0.998 0.998 0.998 0.997 0.997 0.996 0.996 0.996 0.996 0.985 0.992 0.996 0.998
[22] 0.998 0.998 0.998 0.997 0.997 0.996 0.999 0.999 0.999 1.000 1.000 1.000 1.000 0.999 0.999 0.994 0.997 0.998 0.999 0.998 0.997
[43] 0.999 0.998 0.998 0.997 0.998 0.998 0.998 0.999 1.000 1.000 0.999 0.999 0.998 0.999 0.999 0.999 0.999 0.999 0.998 1.000 1.000
[64] 0.999 0.998 0.999 1.000 1.000

                 PC1
SS loadings    67.487
Proportion Var  0.992
> summary(p2x)

Factor analysis with Call: psych::principal(r = xfLand, nfactors = 1, rotate = "varimax",
    covar = FALSE, scores = TRUE)

Test of the hypothesis that 1 factor is sufficient.
The degrees of freedom for the model is 2210  and the objective function was  288.18
The number of observations was  500  with Chi Square =  136835.8  with prob <  0

The root mean square of the residuals (RMSA) is  0.01
```

Output 6: Loadings output from the scaled rotation w/ “Varimax” for x (processed data)

```
Loadings:
 [1] 0.997 0.998 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.998 0.997 0.998 0.997 0.997 0.997
[22] 0.997 0.998 0.997 0.997 0.997 0.998 0.999 0.999 0.999 0.999 1.000 1.000 0.999 1.000 1.000 0.999 0.999 0.999 1.000 0.999 0.999
[43] 1.000 0.999 0.999 0.999 0.999 1.000 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.999 0.999
[64] 0.999 0.999 0.999 0.999 0.999

                 PC1
SS loadings    67.839
Proportion Var  0.998
> summary(p2y)

Factor analysis with Call: psych::principal(r = yfLand, nfactors = 1, rotate = "varimax",
    covar = FALSE, scores = TRUE)

Test of the hypothesis that 1 factor is sufficient.
The degrees of freedom for the model is 2210  and the objective function was  300.51
The number of observations was  500  with Chi Square =  142694.4  with prob <  0

The root mean square of the residuals (RMSA) is  0
```

Output 7: Loadings output from the scaled rotation w/ “Varimax” for y (processed data)

Then, Common Factor Analysis was performed to demonstrate any latent factors in the data. In output 8 and 9, nfactor of 1 was tried, but R gave an error stating “unable to optimize from this starting value.” Therefore the next lowest value, 2, was selected as the nfactor. For both x and y, 2 factors were chosen, since the Common Factor Analysis stated that 2 factors explained close to 100% of the cumulative variances in the data. Most factor values were in the 0.6 to 0.8 range, which did not really show any significant results.

left_eye.4	0.799	0.600
left_eye.5	0.796	0.604
left_eye.6	0.791	0.611
mouth.1	0.734	0.676
mouth.2	0.752	0.658
mouth.3	0.764	0.645
mouth.4	0.771	0.636
mouth.5	0.777	0.629
mouth.6	0.784	0.620
mouth.7	0.791	0.610
mouth.8	0.785	0.619
mouth.9	0.777	0.628
mouth.10	0.771	0.636
mouth.11	0.764	0.645
mouth.12	0.751	0.658
mouth.13	0.741	0.669
mouth.14	0.764	0.645
mouth.15	0.771	0.637
mouth.16	0.777	0.629
mouth.17	0.789	0.613
mouth.18	0.777	0.629
mouth.19	0.771	0.636
mouth.20	0.764	0.644
jaw1	0.564	0.821
jaw2	0.581	0.812
jaw3	0.607	0.792
jaw4	0.633	0.769
jaw5	0.664	0.741
jaw6	0.701	0.707
right_eyebrow.1	0.650	0.755
right_eyebrow.2	0.688	0.722
right_eye.1	0.699	0.713

	Factor1	Factor2
SS loadings	38.944	28.900
Proportion Var	0.573	0.425
Cumulative Var	0.573	0.998

Output 8: Output from Common Factor Analysis for x (processed data)

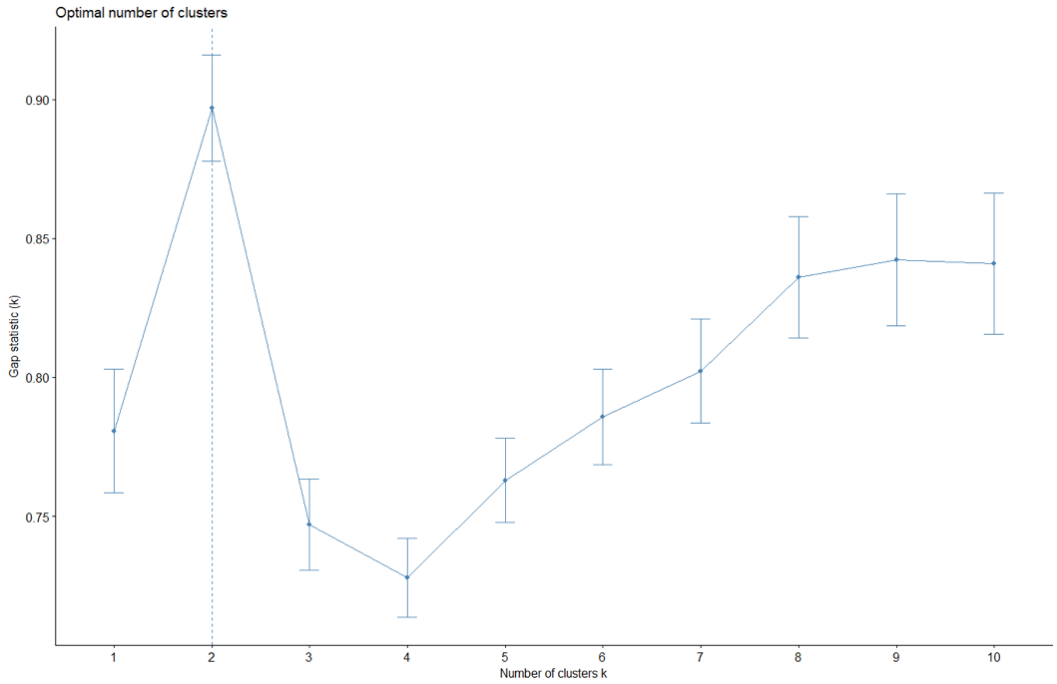
right_eye.1	0.745	0.666
right_eye.2	0.742	0.670
right_eye.3	0.742	0.670
right_eye.4	0.746	0.666
right_eye.5	0.747	0.665
right_eye.6	0.747	0.664
left_eye.1	0.747	0.665
left_eye.2	0.742	0.669
left_eye.3	0.742	0.669
left_eye.4	0.746	0.665
left_eye.5	0.748	0.663
left_eye.6	0.748	0.664
mouth.1	0.781	0.624
mouth.2	0.778	0.628
mouth.3	0.776	0.631
mouth.4	0.776	0.630
mouth.5	0.776	0.631
mouth.6	0.778	0.628
mouth.7	0.782	0.623
mouth.8	0.782	0.623
mouth.9	0.781	0.624
mouth.10	0.781	0.624
mouth.11	0.781	0.624
mouth.12	0.782	0.623
mouth.13	0.781	0.624
mouth.14	0.780	0.626
mouth.15	0.780	0.626
mouth.16	0.780	0.626
mouth.17	0.781	0.624
mouth.18	0.778	0.627
mouth.19	0.778	0.627
mouth.20	0.778	0.627

	Factor1	Factor2
SS loadings	39.494	28.425
Proportion Var	0.581	0.418
Cumulative Var	0.581	0.999

Output 9: Output from Common Factor Analysis for y (processed data)

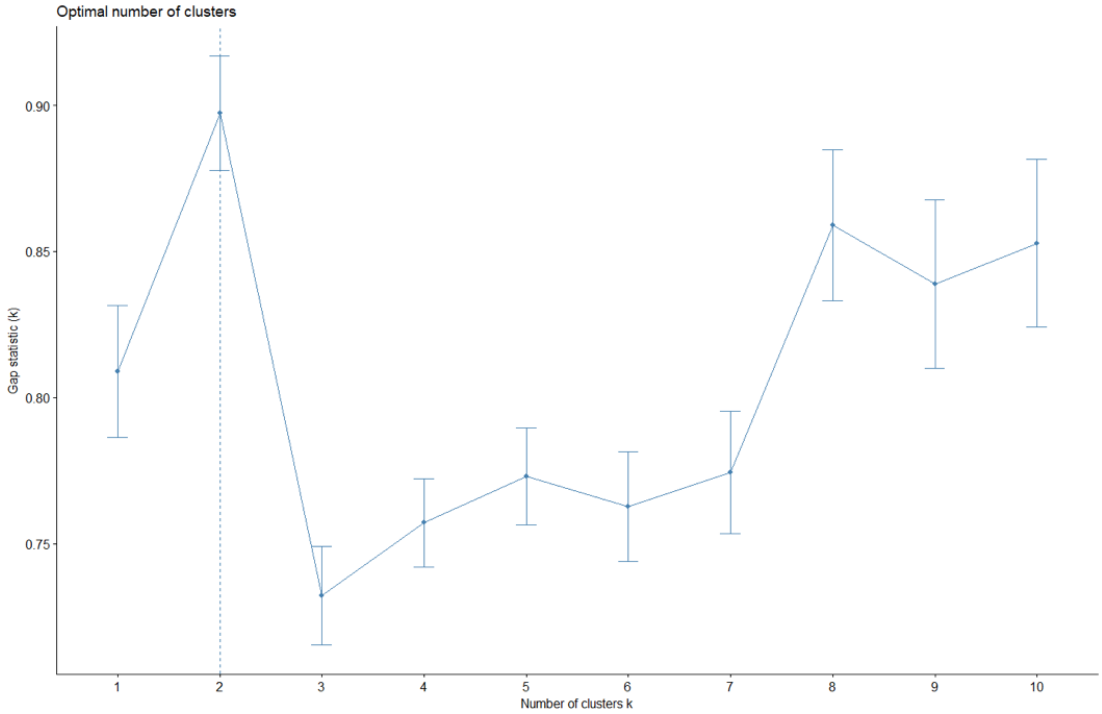
Clustering:

Since Principal Component Analysis and Common Factor Analysis did not show great results, Cluster Analysis was performed additionally. The picture 12 and 14 showed the optimal number of clusters for x and y. 2 clusters were chosen and the plot included how spread apart the data is in picture 13 and 15.



Picture 12: Plot of optimal number of clusters for x (processed data)

Picture 13: Cluster Analysis plot for x (processed data)



Picture 14: Plot of optimal number of clusters for y (processed data)

Picture 15: Cluster Analysis plot for y (processed data)

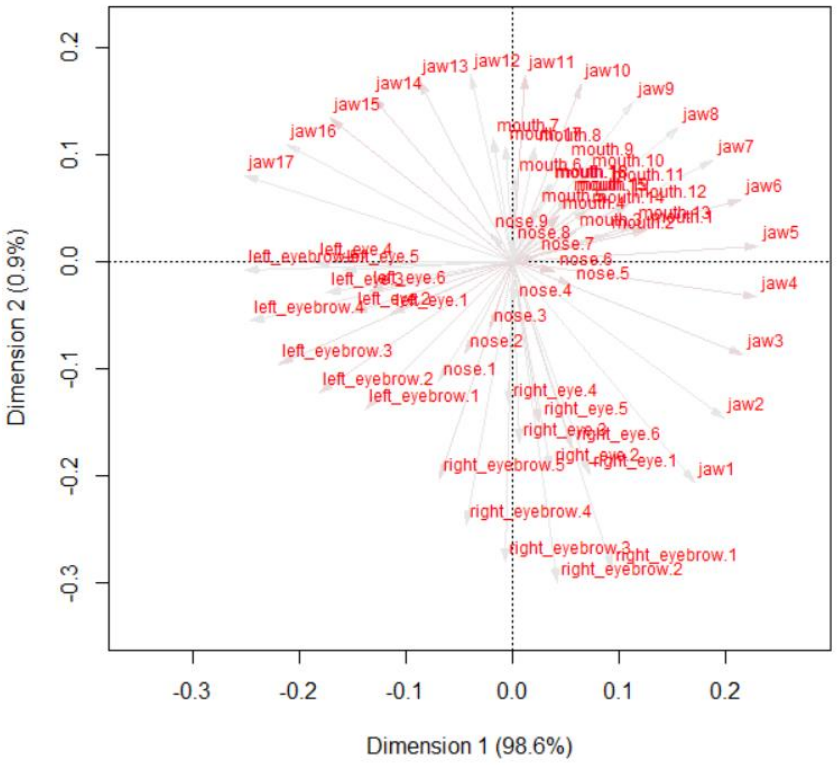
Due to the strong relationships/linearity issues in the data, Principal Component Analysis, rotation, Common Factor Analysis and Cluster Analysis did not bring any great conclusions in terms of model selection. It also does make sense that these analysis methods did not do much, because facial structure can change so much. It illustrated that y values had more linear relationships than x values, since there are more room to change vertically than horizontally when talking about facial landmarks. Additional exploratory analysis methods will be used to determine a suitable model.

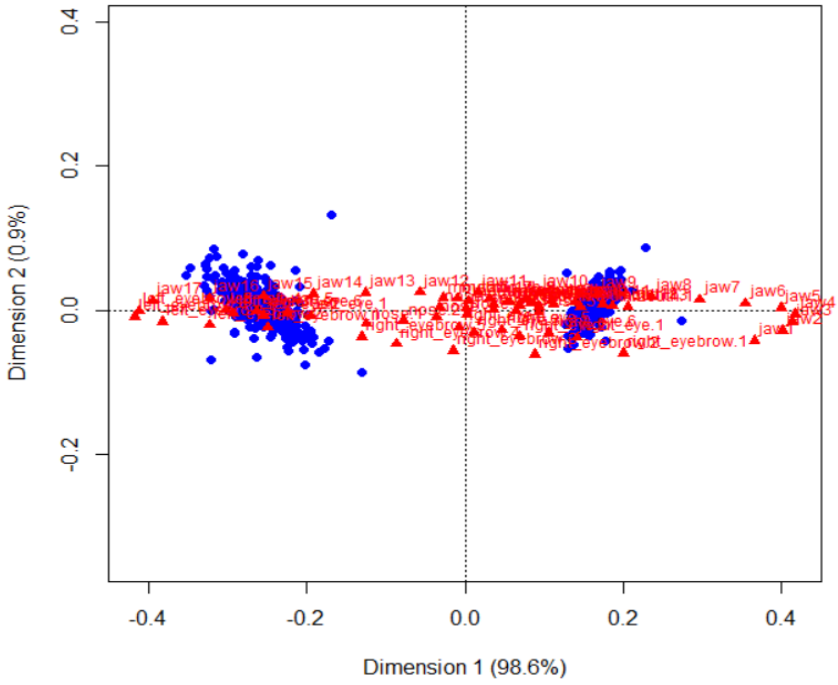
Correspondence Analysis:

I select correspondence analysis to explore the dataset. Firstly, I store all 68 variables and run CA on them. Using the plot, I demonstrate the facial landmarks on the graph. As a result, it shows which variables are best representative. In order to interpret the plot, you need to draw two arrows between the variables. The acute the angle, the more these 2 variables correspond. On the other hand, an obtuse angle between variables means that these 2 variables do not correspond a lot. For instance, as illustrated on the plot below, the angle between 'jaw16' and 'jaw17' is very acute. This means that these 2 variables correspond to each other very well. Also, to the contrary, 'jaw17' and 'righteyebrow' 5 depicts an obtuse angle, which means these 2 variables do not correspond well.

Another visualization of the CA is drawing a line of origin to create a scale. I decided not to use this plot as the visualization was not clearly illustrated due to many variables in our dataset. However, the process goes along like this. Firstly, draw a line of origin through a variable to create a scale. Then, with each variable draw a perpendicular line to the scale. The closer the perpendicular strikes the line, the more these two variables correspond and vice-versa. To illustrate to some extent, drawing a line of origin through jaw7 means we created a 'jaw7' scale. By visualizing the perpendicular lines amongst the variables, we can depict that amongst the many variables which will correspond well with 'jaw7' some to mention are 'jaw6','righteyebrow1' and 'jaw8'. On the other side, 'jaw17','jaw14' and 'jaw13' would be some of the many variables which would correspond less.

Generally, our dataset is very correlated as we are dealing with a dataset of the same type. Correspondence analysis is an appropriate technique to explore relationships amongst variable response categories and can play a role in analyzing facial landmark data. Though of its effectiveness, further analysis should be explored to get a better understanding of the dataset.


Dimension 2 (0.9%)
Dimension 1 (98.6%)
jaw1
jaw2
jaw3
jaw4
jaw5
jaw6
jaw7
jaw8
jaw9
jaw10
jaw11
jaw12
jaw13
jaw14
jaw15
jaw16
jaw17
nose.1
nose.2
nose.3
nose.4
nose.5
nose.6
nose.7
nose.8
nose.9
left_eyebrow.1
left_eyebrow.2
left_eyebrow.3
left_eyebrow.4
right_eyebrow.1
right_eyebrow.2
right_eyebrow.3
right_eyebrow.4
right_eyebrow.5
right_eye.1
right_eye.2
right_eye.4
right_eye.5
right_eye.6


Dimension 2 (0.9%)
Dimension 1 (98.6%)

Correspondence analysis was interesting and valuable to our understanding of the dataset. I used R to create these plots. I stored and ran CA on all 68 variables. After they were stored, I created two different types of plots. The figure below is a screenshot of my code in R which illustrates the process and acquiring of the plots.

```
setwd("c:/datasets")
library(ca)


#Loading the  dataset
store <- read.csv('facialLandmarks.csv', header=TRUE)
#looking at the dataset
head(store)



Men=store[2:68]
Men
c=ca(Men)
c



plot(c)
plot(c, mass=T, contrib="absolute",
     map="rowgreen", arrows=c(F, T))
```

Regression and Regularized Regression:

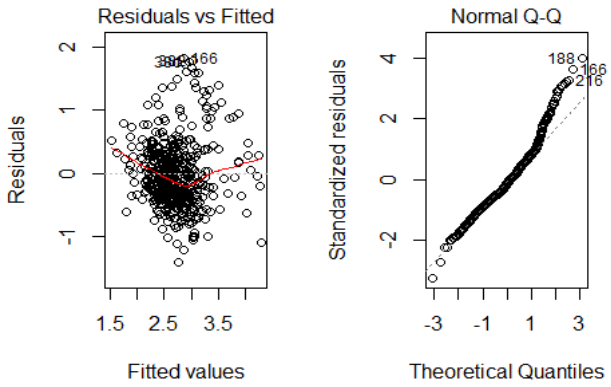
I analysed the face feature data by regression and classification. The goal is to predict exact attractiveness score for each person based on face features. I used multiple linear regression (MLR) as well as principal component regression (PCR).

Firstly, I used a multiple linear regression to explore the relations among the response attractiveness label. In multiple regression model, each part of faces such as left eye has two coordinates x and y. I flatted them into 1 row and gave appropriate names of the features with 'x' and 'y, so i have 136 predictors and 1 response. Then, I applied 10-folds cross validation to check the performance of model. For 10-folds cross validation, I randomized the raw data and divided the randomized data into equally 10 parts. For each part, I used the other 9 parts as the training data to train model and predict on the selected part. Overall, I used MSE as a measurement of performace. My result shows that the MSE is around 0.68, which appears to be pretty good.

```
mse = mean((dataVld$ratings - dataVld$pred)^2)
mse

[1] 0.6757903
```

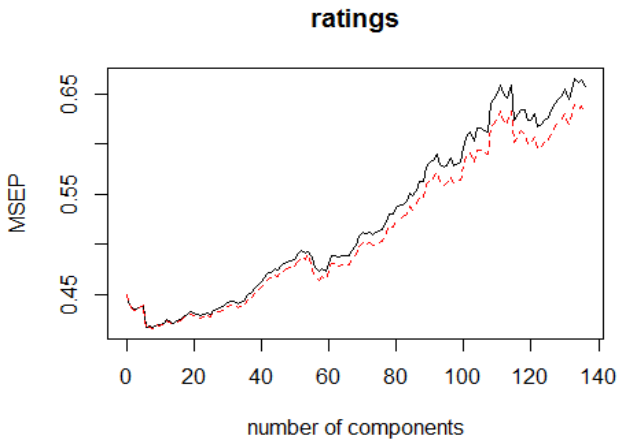
The result shows that MSE is 0.6757903.



Residuals plot shows there is 'double bow' pattern. The normal qq plot shows there are lots of points far away from the straight line at the tail, indicating model transformation is needed to improve the model.

```
pcr.MSE = mean((predict(fit, dataVld, ncomp = 5)- dataVld$ratings)^2)
pcr.MSE↵

[1] 0.4271767↵
```




```
pcr.MSE = mean((predict(fit, dataVld, ncomp = 5)- dataVld$ratings)^2)
pcr.MSE↵

[1] 0.4271767↵
```

Validation plot shows that best number of components for PCR is 5 as it has lowest MSE 0.4271767.

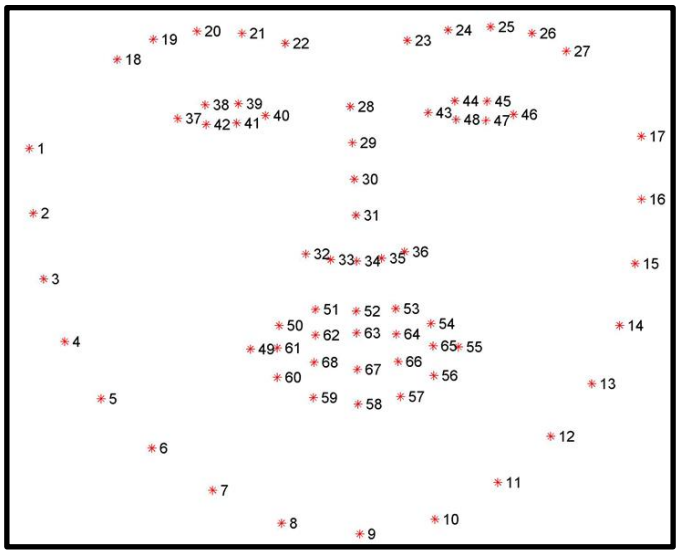
The goal of classification is to predict whether each person is attractive of not based on his/her face features. Here, I used linear discriminant analysis (LDA) and misclassification error rate as measurement to explore the face data.

Firstly, I divided the continuous ratings of faces into two groups. One group is 'Yes' and the other group is 'No' decided by whether the attractiveness ratings is above the mean of ratings. Then I applied 10-fold cross validation and computed the overall accuracy in classification. The score ranges from 1 to 5, with an average of 3. Above average score, I used label 'Yes', which means attractive. Below avarage score, I used label 'No', which means not attractive. My result shows that the overall accuracy is about 0.608. It appears that our LDA model is a little better than randomly selection of 'Yes' or 'No'. One reason might be collinearity in the features of faces. The other reason, I think, might be the cut-off of the attractiveness is not suitable. From the LDA result, we can see there are 108 mistakes when predicting people looking good as bad. Overall, LDA model is very efficient and useful. In order to improve our performance, we can try other techniques such as Lasso Regression.

There were some issues with the original data downloaded from the website. First, none of the data was labelled. There was no columns to go off of so we could not understand any of the results we would have gotten. Second, we did not have access to the python landmark tool to test on our own images. While this would not have been the end of the world it was something I wanted to do.

The process of getting the landmarks was involved and had many issues. I had very little experience with Python and the libraries needed only worked within Linux. The Python code will be listed in the Appendix. So after many hours of troubleshooting I was able to run our 500 image data set through the script using a for loop to code all at once. Below is an image of the facial landmarks that are numbered. Each one of those landmarks had an x and y coordinate and this is the data we would use for our modeling.

The way that the python script was grabbing facial features in of itself was a machine learning technique as I imported a pre-trained model to grab the features from our images. This model had been given hundreds of different faces with pre-mapped features and then was able to find these features on new images. I used for loops and array manipulation to efficiently process all of our images so once the script was running it was fairly easy to extract the data to a usable file.



68 landmarks making up 7 major portions of the face.

The regression process first started with splitting the data which now has the shape of 500x136. I used the floor function to split both the facial data (Dependent data) and the attractiveness ratings (Independent data). I started with the full model using the training data:

```
Residual standard error: 0.607 on 238 degrees of freedom
Multiple R-squared:  0.5013,    Adjusted R-squared:  0.2164
F-statistic: 1.759 on 136 and 238 DF,  p-value: 7.305e-05

> mape <- mean(abs((fullPredict - testy))/testy)
> mape
[1] 0.2341761
```

As you can see this model is fairly average and most noteable you can see the difference between the R^2 and adj. R^2 because we are working with 136 variables. After this I then proceeded to perform Backwards stepwise selection to automatically select the variables. This gave pretty good improvements but still this model is no where near perfect. Adj R^2 of ~.35, MAPE of .224, RMSE of 1.22 and MSE of .69. This model improves in all aspects over the full model but the error values are still very high.

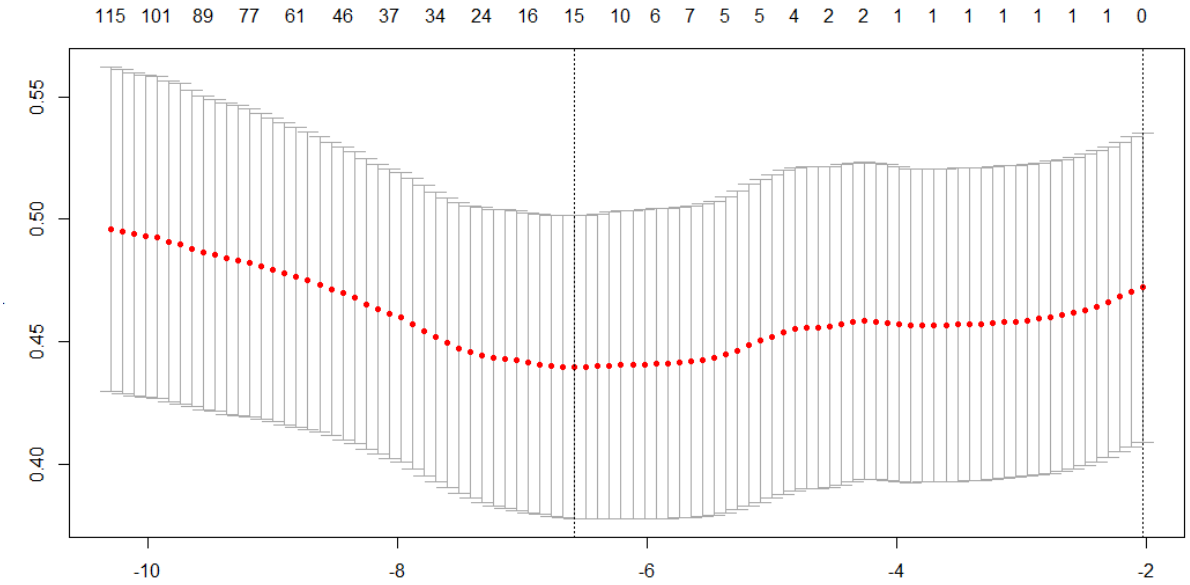
```
Residual standard error: 0.5529 on 308 degrees of freedom
Multiple R-squared:  0.4645,    Adjusted R-squared:  0.3498
F-statistic: 4.049 on 66 and 308 DF,  p-value: < 2.2e-16

> mape <- mean(abs((predictions - testy))/testy)
> mape
[1] 0.2239169
```

```
> rsePredict
[1] 1.224385
> mse
[1] 0.6955912
```

At this point I know that we have incredibly correlated data and that adding a penalty constraint on our regression could make a much more reliable model. I first performed ridge regression but halfway through I realized that it would be nice if we could automatically eliminate some of our 136 variables. So I decided to proceed with Lasso Regression. I did this using Cross validation and glmnet library to test for a wide range of lambda's to find the best one for our model. Below

(Graph 1) is the plot of the lambda's attempting to find the minimize the MSE (y axis). The final lambda we get is .001387 so that is the lambda we will use for our regression.



So after fitting our model we get 17 remaining variables. There seems to be a huge emphasis on jaw points in terms of attractiveness. Below is the full breakdown of the variables. The biggest representation amonst the variables is the Jaw group of x and y variables. 9 of our 17 variables were related to jaw structure. However if you look at the impact of each variable jaw7, mouth 5 y, jaw9 y and nose 7 y had the biggest individual impacts.

[,1]	[,2]
"(Intercept)"	"2.51768583564636"
"jaw1"	"-0.000846857956314382"
"jaw5"	"0.0083918385021834"
"jaw8"	"-0.000346279554129708"
"jaw12"	"-0.00148983913777072"
"jaw13"	"-2.26526868919727e-08"
"right_eyebrow 1"	"-0.00635888417389142"
"right_eyebrow 5"	"0.000178266138984 61"
"left_eyebrow 4"	"0.000833280498734204"
"jaw1 y"	"-0.00458380337084622"
"jaw7 y"	"4.5306074801391e-05"
"jaw8 y"	"0.000129577379228305"
"jaw9 y"	"3.03443105097005e-09"
"nose 4 y"	"0.000864785092852011"
"nose 7 y"	"2.4683011315565e-05"
"left_eye 5 y"	"0.0035838566350521"
"mouth 5 y"	"4.21562440709026e-05"

We can then run the same prediction analysis on this model as we did the other two. We got the output below. This is a much better model in terms of prediction on our testing data.

```
> rsePredict
[1] 0.6517649
> mse
[1] 0.3670251
> mapeLasso <- mean(abs((yPredict -
> mapeLasso
[1] 0.1710937
```

LASSO regression is a means of working with correlated data and minimizing the model to get the least amount of error. LASSO stands for least absolute shrinkage and selection. The name itself describes what it is doing. LASSO creates a lambda in which is a penalty parameter in order to reduce over fitting. The benefit of using LASSO over ridge in this example is that we wanted to eliminate some of our variables and LASSO sets weak impact features to 0 whereas Ridge keeps all variables and just makes coefficients smaller. Since our model had 136 variables and we were unable to do PCA

Lasso regression seemed like the perfect fit for our data. This showed as it gave us the best outcomes based on the 5 models we created.

When this model was used for prediction on the face of a team member and the face of a celebrity to test it on outside data it responded with unexpected results. The celebrity's face was at an angle and not completely centered. There are some issues when testing computer vision models like this however. The model is incredibly affected by the position, angle, and expression on the face. There are many ways this project could be improved. For example we could have created a much more massive data set that took the distances between all points on the face and use that instead of our data. This is much more complicated and harder to manage than our data but it would have ironed out the inconsistencies between faces in the original data set and the outside test images.

The practical uses of this analysis are pretty self explanatory as they could be used to rate attractiveness. However this is only for one ethnicity and attractiveness is incredibly subjective so building a perfect model would never be possible. One person who one group of raters thinks is attractive could be the opposite for another group. I do not believe the real world uses of this would extend far beyond personal use for recreation as it is interesting to see if what machine learning predicts your attractiveness as. I see no real applications even in professions such as modeling since there is much more than just base attractiveness in people.