Researcher: Dustin McAfee

Presentation Title: High Performance Clustering of Electroencephalogram (EEG) Data for Prediction of Seizure Events

Institution: University of Tennessee, Knoxville

Department: Department of Electrical Engineering and Computer Science

1 Abstract

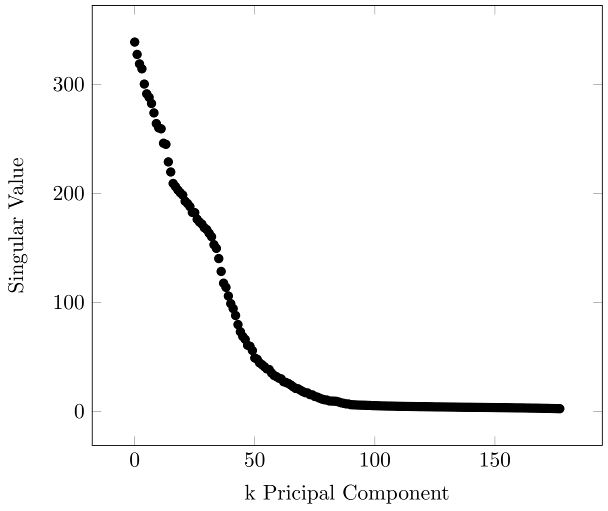

Figure 1: Scree Graph

If s_i is the ith principal component, then the amount of overall variance explained by s_i, denoted r_i^2, is computed using the below formula, Formula 1.

$$r_i^2 = \frac{s_i^2}{\sum_j s_j^2} \tag{1}$$

These values are computed in "data.py", saved in the file, "Singular_Values_Percent_Variance.txt", and plotted in a scatter plot as Figure 2, below.

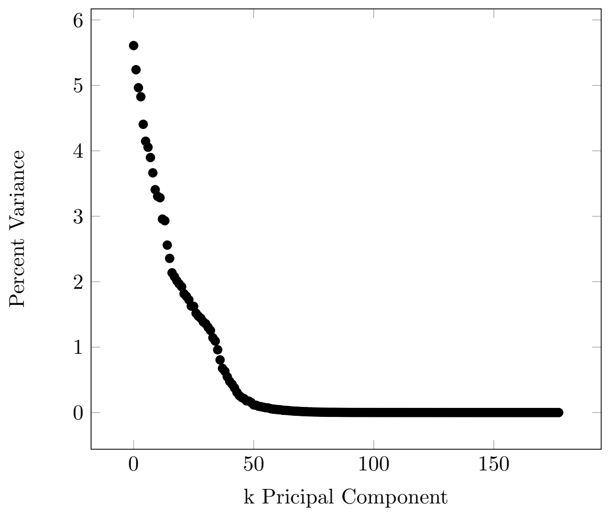

Figure 2: Percent Variance vs Principal Components

The first 40 principal components cover about 95.88% of the variance; looking at Figures 1 and 2 above, this looks to be a decent elbow point. The first two principal components are plotted against each other, and the projection of the Testing Dataset is color coded by category and shown below, in Figure 3. Blue marks seizure moments.

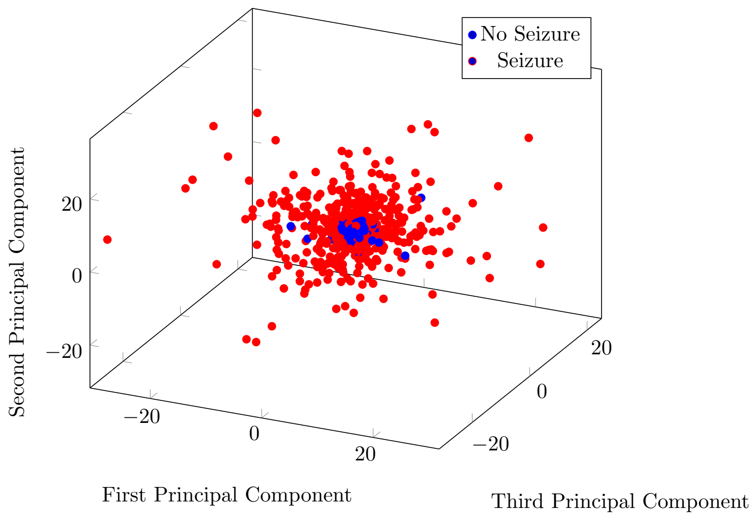

Figure 3: Testing Dataset Projected onto first 3 Principal Components

The K-Means clustering algorithm is implemented in "kmeans.py," and takes one argument: k number of clusters, (see "README.txt"). The initial values (seeds) for the centroids are chosen based on the K-Means++ algorithm [1]. The first seed is chosen at random, then for each data point, x, the minimum difference between each x and chosen centroids are used as weights for picking the next centroid. Each new centroid is chosen at random from the dataset, using the weighted probability distribution where the weights are proportional to the distances to the nearest centroid (See function "centroid" in "kmeans.py"). The purpose here is an attempt to initially start with clusters that are relatively spread out. The program converges when the sum of the differences of each centroid is less than 0.005.

The minimum intercluster distance and maximum intracluster distance are computed using the distance matrix. This algorithm is run for k clusters, where $k = 2, 3$, on the standardized preprocessed numerical dataset projected onto the first 40 Principal Components. The maximum intercluster distance is computed using the maximum distance between any two points of different clusters, and similarly, the minumum intracluster distance is computed using the minimum of any two points of the same cluster. These two metrics, the Dunn index, and the number of iterations until convergence for each of the k clusterings are reported in Figure 4, below.

	k=2	k=3
Min Inter	9.94	0.74
Max Intra	92.77	91.71
Dunn	0.0963	0.0080
# Iterations	20	27

Figure 4: Dunn index for different K values on Projected Training Dataset

	k=2	k=3
Min Inter	1.47	0.39
Max Intra	89.69	86.68
Dunn	0.0164	0.0045

Figure 5: Dunn index for different K values on Projected Testing Dataset

		Prediction outcome		
		n	**p**	**Total**
Actual value	**n′**	7064	3	7067
	p′	1527	206	1733
	Total	8591	209	

Figure 6: Confusion Matrix for K = 2 on Projected Training Dataset (82.61% Accuracy, 11.86% Sensitivity)

		Prediction outcome		
		n	**p**	**Total**
Actual value	**n′**	2132	2	2134
	p′	500	65	565
	Total	2632	67	

Figure 7: Confusion Matrix for K = 2 on Projected Testing Dataset (81.44% Accuracy, 11.66% Sensitivity)

		Prediction outcome		
		n	**p**	**Total**
Actual value	**n′**	2423	4644	7067
	p′	1481	252	1733
	Total	3904	4809	

Figure 8: Confusion Matrix for K = 3 on Projected Training Dataset (30.40% Accuracy, 14.54% Sensitivity)

		Prediction outcome		
		n	**p**	**Total**
Actual value	**n′**	701	1433	2134
	p′	489	76	565
	Total	1190	1509	

Figure 9: Confusion Matrix for K = 3 on Projected Testing Dataset (28.79% Accuracy, 13.45% Sensitivity)

The K-Means clustering of the preprocessed training dataset for 2 (3) clusters is saved in the file "Data_Kmeans_2.txt" ("Data_Kmeans_3.txt"), and the clusters are color coded and plotted against the testing dataset projected onto the first 3 principal components (see files "Testing-Data_Kmeans_2.txt" and "TestingData_Kmeans_3.txt") in Figure 10 (Figure 11), below.

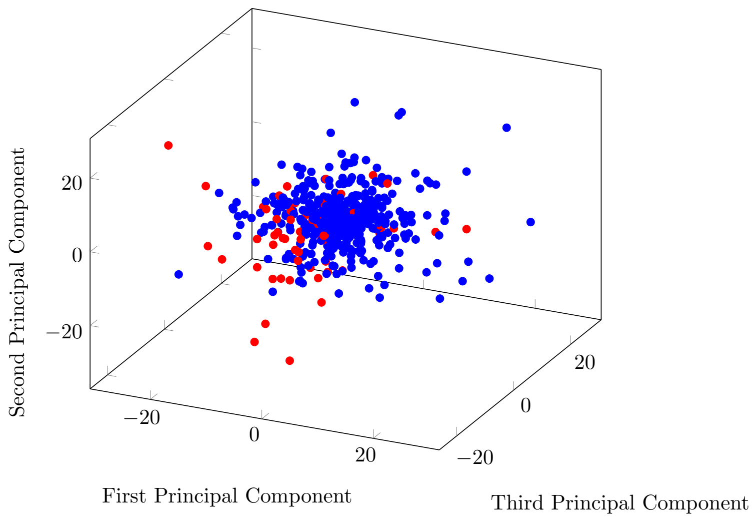

Figure 10: K-Means Clustering (K=2) of Standardized Testing Dataset vs First 3 Principal Components

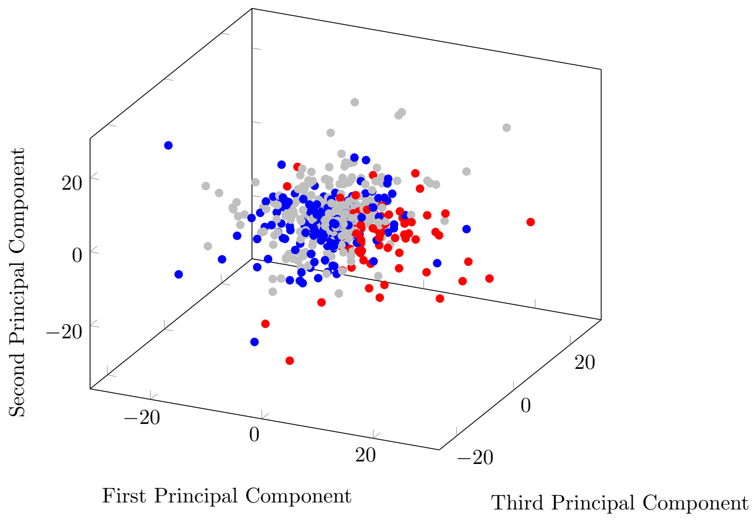

Figure 11: K-Means Clustering (K=3) of Standardized Testing Dataset vs First 3 Principal Components

The Expectation Maximization algorithm assumes Gaussian models for each cluster. The program (”exmax.py”) borrows a few functions from the K-Means program (”kmeans.py”), such as the initialization of the centroids (the K-Means++ method [1]), assigning points to labels, and calculating the sum of changes in the centroids. Similarly to the K-Means program, the Expectation Maximization program converges when the sum of changes in the centroids are less than 0.005

This program is run on different sized clusters. Contrast to the K-Means algorithm, the Dunn index is not a good index for validation of clusters of Gaussian distributions, so a different index is used for validation of the clusters in this case. Instead of using the ratio of minimum intercluster distance to maximum intracluster distance (which is good for spherical distributions), the index chosen is the Silhouette index. To compute this index, the average intracluster distance, $a(i)$ is calculated per data point, i. The smallest average intercluster distance, $b(i)$, per data point, i, is also computed. Equation 2, below, computes the Silhouette index per data point (see function "avg_sil_index" in "exmax.py").

$$\frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \tag{2}$$

This value is averaged over the entire dataset to produce the Silhouette index for the dataset, which better measures how appropriately the data has been clustered for the Gaussian clusters.

To this end, $k = 2, 3$ clusters are run and their respective Silhouette indices and number of iterations are reported in Figure **??**, below.

k	k=2	k=3
Silhouette	0.3789	0.0580
# Iterations	13	14

Figure 12: Silhouette indices for Training Dataset k values

		Prediction outcome		
		n	**p**	**Total**
Actual value	**n′**	6829	238	7067
	p′	72	1661	1733
	Total	6901	1899	

Figure 13: Confusion Matrix for K = 2 on Projected Training Dataset (96.48% Accuracy, 95.85% Sensitivity)

		Prediction outcome		
		n	**p**	**Total**
Actual value	**n′**	1895	239	2134
	p′	107	458	565
	Total	2002	697	

Figure 14: Confusion Matrix for K = 2 on Projected Testing Dataset (87.22% Accuracy, 81.10% Sensitivity)

		Prediction outcome		
		n	**p**	**Total**
Actual value	**n′**	4191	2876	7067
	p′	0	1733	1733
	Total	4191	4609	

Figure 15: Confusion Matrix for K = 3 on Projected Training Dataset (67.32% Accuracy, 100% Sensitivity)

	Prediction outcome		
Actual value	**n**	**p**	**Total**
n′	1266	868	2134
p′	44	521	565
Total	1310	1389	

Figure 16: Confusion Matrix for K = 3 on Projected Testing Dataset (66.25% Accuracy, 92.23% Sensitivity)

Figures 17 and 18, below, show clusterings using the Gaussian Mixture Expectation Maximization model run on the dataset projected onto the first 40 principal components with k=2 and k=3 (saved in files "Data_EM_2.txt" and "Data_EM_3.txt"), respectively.

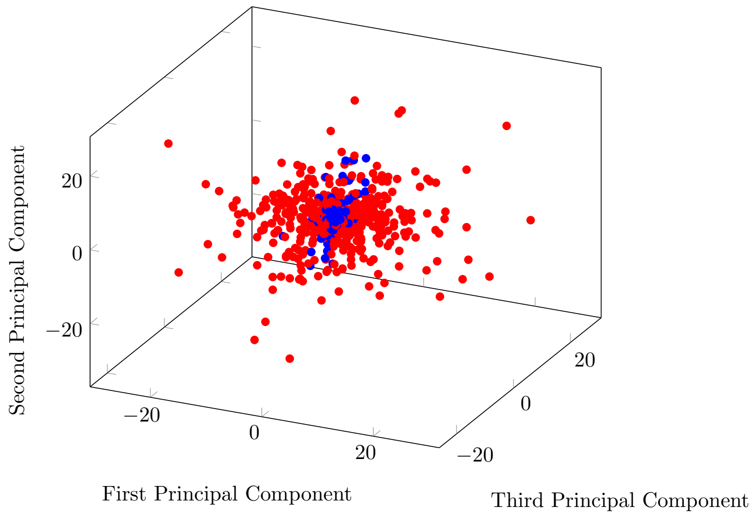

Figure 17: EM Clustering (K=2) of Standardized Testing Dataset Projected onto First 3 Principal Components

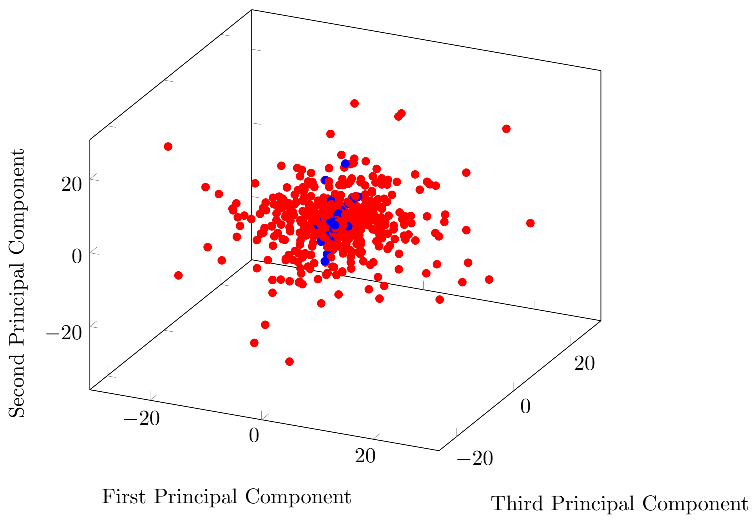

Figure 18: EM Clustering (K=3) of Standardized Testing Dataset Projected onto First 3 Principal Components

References

[1] Arthur, David & Vassilvitskii, Sergei. (2007). K-Means++: The Advantages of Careful Seeding. Proc. of the Annu. ACM-SIAM Symp. on Discrete Algorithms. 8. 1027-1035. 10.1145/1283383.1283494.

Appendices

A Three Dimensional Plots

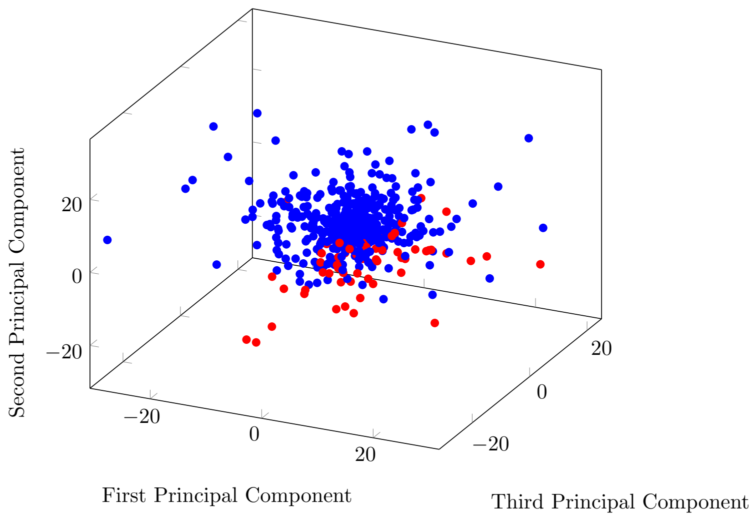

Figure 19: K-Means Clustering (K=2) of Standardized Data vs First 3 Principal Components