

Abstract

Feature selection is of great important for applications where dimensionality reduction, analysis, and pattern discovery are to be deployed. This need is perhaps more for systems with limited computing resources like IoT networks. In this paper, we considered time series datasets and propose a unsupervised learning technique to identify the top-k discriminative features. The technique used Principal Component Analysis (PCA) statistical foundation to deduce the relative importance of the principal components of the dataset with its coefficients along the principal directions, consequently uncovering the ranks of the features. We use multiple benchmark datasets for various experiments evaluate the performance of the proposed method in terms of its ability for feature selection and and its capacity to minimize the original by evaluating the data reconstruction error. Our proposed method compared with other existing methods, results verify increased efficiency and accuracy.

Background

The explosion of big data based upon technological advances presents its own challenges that are not sufficiently solved by the existing data reduction, analysis and feature selection methodologies. This is also the challenge of limited computing resources for some edge computing systems like the IoT networks and many other edge devices. The additional presence of noise in high dimensional datasets makes it more difficult to uncover significant patterns in data, and this affects the quality of the systems.

This makes feature selection and feature extraction important preprocessing steps for improved accuracy and efficiency in uncovering patterns and trends in a dataset and for reduction of data size useful for computing systems with limited computing resources. This leads to improved efficiency of the overall system.

Efficient Data Reduction Technique by selecting Top-K discriminative Features using Principal Component Analysis for Efficient Light-Weight AI Models

Principal Investigator: Dr. Cajetan Akujuobi

Prairie View A&M University, Prairie View, TX 77446



Fig.1 System diagram.

Results

				1		
Dataset	SVD	Test	Inference	N_samp	N_features	E
	Reconstr	Accuracy	_time	les		С
	uction		(sec)			
	Error					
Arrhythmia	0.715595	0.604396	0.012478	452	279	1
Ionosphere	0.364809	0.971831	0.013278	351	34	1
madelon	0.978347	0.785000	0.020385	2000	502	5
maaoron		017 00 000	0.020000	2000	502	
						<u> </u>
Gissette	0.828217	0.971972	0.035977	5999	5000	8
IoT	0.068727	0.974825	0.671137	80037	115	2
Intrusion						

Table 1: Result Table

Student: Faith Nwokoma

nput: $A \in H^{Corr}$, θ (cumulative variance explained) Output: k, the number of principal components to retain. begin 1: Uncover fraction of total explained variance 2: $f(k) \leftarrow \sum_{i=1}^{k} \lambda_i \sum_{i=1}^{r} \lambda_i$ for all $z \in \{1,, r\}$ 3: Choose the smallest k so that $f(k) \ge \theta$ and retain that number of k eigenvectors to keep explained variance θ in the new embedding. 4: return k end The (rank-k) weighted score of the i-th column of A is then computed as $wS_i^{(k)} = \Sigma_{j=1}^k w_j t_{i,j} $. Algorithm 2 - Weighted Scores (WS) Input: $A \in H^{r \times m_i}$, θ (cumulative variance explained) Output: $S_i \in R^{r \times m_i}$, θ (cumulative variance explained) Output: $S_i \in R^{r \times m_i}$, θ (cumulative variance explained) Output: $S_i \in R^{r \times m_i}$, θ (cumulative variance explained) Output: $S_i \in R^{r \times m_i}$, θ (cumulative variance explained) Output: $S_i \in R^{r \times m_i}$, θ (cumulative variance explained) Output: $S_i \in R^{r \times m_i}$, θ (cumulative variance explained) Output: $S_i \in R^{r \times m_i}$, θ (cumulative variance explained) Output: $S_i \in R^{r \times m_i}$, θ (cumulative variance explained) Output: $S_i \in R^{r \times m_i}$, θ (cumulative variance explained) Output: $S_i \in R^{r \times m_i}$, θ (cumulative variance explained) Output: $S_i \in R^{r \times m_i}$, θ (cumulative variance explained) 2: Compute the singular Value Decomposition $[U, S, V^{r}] \leftarrow SVD(A)$ 3: Identify the number k of principal components to retain $k \leftarrow Algorithm 1(A, \theta)$ 4: $M \leftarrow V_k$ 5: Build the weighted matrix $[wV_i]$ For $j \leftarrow 1$ os $w_{ij} \leftarrow W_{ij} = W_{ij} = W_{ij} = W_{ij}$, for all $i = (1, 2,, m)$. 7: Sortice variables, locording to their weights: $wS_i^{(k)} \ge w \otimes S_i^{(k)} \ge w \otimes S_m^{(k)} \ge w \otimes S_m^{(k)}$. end M_comp ts m Error 6: Number OF Samples Trained On		Algorithm 1 - Uncover the number k of PCs to retain
 i. Uncover fraction of total explained variance f(k) ← Σ^(k)_{k=1}λ_k, (Σ^k_{k=1}λ_k, for all z = {1,, r} i. Choose the smallest k so that f(k) ≥ θ and retain that number of k eigenvectors to keep explained variance θ in the new embedding. i. return k end The (rank-k) weighted score of the i-th column of A is then computed as wS^(k)_i = Σ^k_{j=1}w_jt_{i,j} . Agorithm 2 - Weighted Scores (WS) Input: A ∈ R^(N,m), θ (cumulative variance explained) Output: S, ∈ R^{n+N-m}, θ (cumulative variance explained) Compute the Singular Value Decomposition [U, S, V⁻¹] ← S¹D(A) Compute the propriorin of variance explained component For j ← 1 to t w_i + (x)¹(Σ^j_{i=1}λ_i (wV_k), j ← w_k + [M¹_{k+j}], end for Build the weighted matrix [wV_k] For j ← 1 to k w_k + (x) (Z^j_{k=1}λ_k) w_k + (x) = (X^j_{k=1}A_k), f(k) = 1 = (1, 2,, m). Sort the variables according to their weights: w_k + (x) = (X^j_{k=1}A_k) w_k + (x) = (X^j_{k=1}A_k)		Input: $A \in \mathbb{R}^{n \times m}$, θ (cumulative variance explained) Output: k, the number of principal components to retain. <i>begin</i>
 10) ← D_{k=1}A_k/D_{k=1}A_k for all z = {1,, 1} 2: Choose the smallest k so that f(k) ≥ d and retain that number of k eigenvectors to keep explained variance θ in the new embedding. 4: return k end The (rank-k) weighted score of the i-th column of A is then computed as wS_i^(k) = Σ_{j=1}^kw_jt_{i,j} . Algorithm 2: Weighted Scores (WS) Input: A ∈ R^{×Xm}, θ (cumulative variance explained) Output: S, ∈ R^{×Km}, θ (cumulative variance explained) Compute the singular Value Decomposition [U, S, V^T] + S^VD(A) Compute the singular Value Decomposition [U, S, V^T] + S^VD(A) Compute the interportion of variance carried by each component For j ← 1 to r A_j (+ (a²_j)(n − 1)), where s_j ∈ S end for Build the weighted matrix [wVi] For j ← 1 to k w_j (+ (a_j)(m + 1)), where s_j ∈ S end for Compute the ingited score for each variable w_j^{S(K)} ≥ ⊥ w_j^{S(}		1: Uncover fraction of total explained variance
$ Solution of a conjection is to keep explained variance of a the new embedding. 4: return k end The (rank-k) weighted score of the i-th column of A is then computed as wS_i^{(k)} = \Sigma_{j=1}^k w_j t_{i,j} .Algorithm 2 - Veighted Scores (WS)Thut: S_i \in R^{n \times k} and has the top k most representative ranked features of A.Begin1: (Compute the Singular Value Decomposition [(S, V+] - SVD(A)]2: Compute the proportion of variance carried by each component E_i = 1 to TA_j \in (\hat{c}_j^2/(\alpha - 1)), where s_j \in S3: Identify the number k of principal components to retain k \in A ligorithm 1(A, \theta)3: Identify the number k of principal components to retain k \in A ligorithm 1(A, \theta)3: Build the weighted matrix [wV_k]For j \in U to Tw_j \in A_j/(\Sigma_{j=1}^k \lambda_j)end for6: Compute the eighted score for each variable wS_1^{(m)} \geq \dots \geq wS_m^{(m)}endN_ComptsmError & Number OF Samples Trained On$		 I(k) ← Σ_{z=1} λ_z/Σ_{z=1} λ_z for all z = {1,, r} Choose the smallest k so that f(k) ≥ θ and retain that number of k eigenvectors to keep explained variance θ in
4: return k end The (rank-k) weighted score of the i-th column of A is then computed as wS _i ^(k) = Σ ^k _{j=1} w _j t _{i,j} . Agorithm 2 - K ^k × ^{kn} , θ (cumulative variance explained) Output: S _i ∈ R ^{k × m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k × m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k × m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k × m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k ∨ m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k ∨ m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k ∨ m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k ∨ m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k ∨ m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k ∨ m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k ∨ m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k ∨ m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k ∨ m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k ∨ m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k ∨ m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k ∨ m} , θ (cumulative variance explained) Output: S _i ∈ R ^{k ∨ m} , θ (cumulative variance explained) Output: S _i ∈ S ^k		the new embedding.
The (rank-k) weighted score of the i-th column of A is then computed as $wS_i^{(k)} = \Sigma_{j=1}^k w_j t_{i,j} $. Algorithm 2 · Weighted Scores (WS) Input: $A \in \mathbb{R}^{n \times m}$, θ (cumulative variance explained) Output: $S_r \in \mathbb{R}^{n \times k}$ and has the top k most representative ranked features of A. <i>Begin</i> 1. Compute the Singular Value Decomposition $ [C, S, V^{\dagger} \in SVD(A) $ 2. Compute the proportion of variance carried by each component For $j \leftarrow 1$ to r $\lambda_j \leftarrow (S_j^2/(n-1))$, where $s_j \in S$ and for 3. Identify the number k of principal components to retain $k \leftarrow Algorithm1(A, \theta)$ 4. $M \leftarrow V_k$ 5. Build the weighted matrix $[wV_k]$ For $j \leftarrow 1$ to k $w_j \leftarrow \lambda_j/\Sigma_{j=1}^{r-1}\lambda_j$ $[wV_k]_{n,j} \leftarrow w_j * [M]_{n,j}$ end for 6. Compute the eighted score for each variable $wS_j^{(h)} \ge \dots wS_i^{(h)} \ge \dots \ge wS_m^{(h)}$ 7. Sort the variables according to their weights: $wS_j^{(h)} \ge \dots wS_i^{(h)} \ge \dots \ge wS_m^{(h)}$ end N_Compt ts In Error & Number OF Samples Trained On		4: return k
The (rank-k) weighted score of the i-th column of A is then computed as $wS_i^{(k)} = \Sigma_{j=1}^k w_j t_{i,j} $. Algorithm 2 · Weighted Scores (WS) Input: $A \in \mathbb{R}^{n \times m}$, θ (cumulative variance explained) Output: $S_r \in \mathbb{R}^{n \times k}$ and has the top k most representative ranked features of A. <i>begin</i> 1: Compute the Singular Value Decomposition $ U, S, V^T \leftarrow SVD(A)$ 2: Compute the proportion of variance carried by each component For $j \leftarrow 1$ to r $\lambda_j \leftarrow (s_j^2/(n-1))$, where $s_j \in S$ end for $k \leftarrow Algorithm 1(A, \theta)$ 4: $M \leftarrow V_k$ 5: Build the weighted matrix $[wV_k]$ For $j \leftarrow 1$ to k $w_j \leftarrow \lambda_j/\Sigma_{j=1}^r \lambda_j$ $[wV_k]_{n,j} \leftarrow w_j * [M]_{n,j}$ end for 6: Compute the eighted score for each variable $wS_k^{(k)} = \Sigma_{j=1}^k w_j t_{i,j} $, for all $i = \{1, 2,, m\}$. 7: Sort the variables according to their weights: $wS_1^{(k)} \geq \geq wS_m^{(k)}$ end N_Comp ts h Error & Number OF Samples Trained On		cnu
Algorithm 2 - Weighted Scores (WS) Imput: $A \in R^{n \times m}$, θ (cumulative variance explained) Output: $S_i \in R^{n \times M_i}$, θ (cumulative variance explained) Output: $S_i \in R^{n \times M_i}$, θ (cumulative variance explained) Output: $S_i \in R^{n \times M_i}$, θ (cumulative variance explained) Output: $S_i \in R^{n \times M_i}$, θ (cumulative variance explained) Output: $S_i \in R^{n \times M_i}$, θ (cumulative variance explained) Output: $S_i \in R^{n \times M_i}$, θ (cumulative variance explained) Output: $S_i \in R^{n \times M_i}$, θ (cumulative variance explained) $S_i \in R^{n \times M_i}$, θ (cumulative variance explained) $S_i \in R^{n \times M_i}$, θ (cumulative variance explained) $i \in C_i$ ($S_i^j/(n - 1)$), where $s_i \in S$ $i \in N_i^j \in I$ is a fixed of or explained in the invest of principal components to retain $k \leftarrow Algorithm (A, \theta)$ $i \in M \leftarrow V_i$ S_i Build the weighted matrix $[wV_i]$ $i \in V_i - \sqrt{\lambda_j} (\Sigma_{j=1}^j \lambda_j)$ $i \in V_i = [\Sigma_{j=1}^k w_j(i_j), i]$ end for Compute the eighted score for each variable $wS_i^{(k)} = [\Sigma_{j=1}^k w_j(i_j), i]$ $wS_i^{(k)} = [\Sigma_{j=1}^k w_j(i_j), i]$ end N_compt ts ws ws ws		The (rank-k) weighted score of the i-th column of A is then computed as $wS_i^{(k)} = \Sigma_{i=1}^k w_i t_{i,i} $.
Algorithm 2 - Weighted Scores (WS) Input: $A \in R^{m \times m}$, θ (cumulative variance explained) Output: $S_r \in R^{m \times k}$ and has the top k most representative ranked features of A. <i>legin</i> 1: Compute the Singular Value Decomposition $[U, S, V^T] \leftarrow SVD(A)$ 2: Compute the proportion of variance carried by each component For $j \leftarrow 1$ to r $\gamma_j \leftarrow (s_j^2/(n-1))$, where $s_j \in S$ end for 3: Identify the number k of principal components to retain $k \leftarrow Algorithm1(A, \theta)$ 4: $M \leftarrow V_k$ 5: Build the weighted matrix $[wV_k]$ For $j \leftarrow 1$ to k $w_j \leftarrow \lambda_j Z_j^* = \lambda_j$ $[wV_k]_{*,j} \leftarrow w_j * [M]_{*,j}$ end for 6: Compute the eighted score for each variable $wS_1^{(k)} = [\Sigma_{j=1}^{k-1}w_j n_j(L_j)$ for all $i = \{1, 2,, m\}$. 7: Sort the variables according to their weights: $wS_1^{(k)} \ge \otimes S_1^{(k)} \ge \ge wS_m^{(k)}$ end multi-score of a score of cach variable $wS_1^{(k)} \ge \otimes S_1^{(k)} \ge \ge wS_m^{(k)}$ end for 9: Compute the eighted score for each variable $w_j \in \lambda_j Z_j^* = \lambda_j$ $wS_1^{(k)} \ge \otimes W_m^{(k)} \ge \ge wS_m^{(k)}$ end multi-score of the score of the s		1 1 <u>j</u> =1 <u>j</u> 4,j1
Input: $A \in \mathbb{R}^{n \times m}$, θ (cumulative variance explained) Output: $S_i \in \mathbb{R}^{n \times m}$, and has the top k most representative ranked features of A. <i>begin</i> 1. Compute the Singular Value Decomposition $[U, S, V^T] \leftarrow SVD(A)$ 2. Compute the proportion of variance carried by each component For $j \leftarrow 1$ to r $\lambda_j \leftarrow (s_j^2/(n-1))$, where $s_j \in S$ end for 3. Identify the number k of principal components to retain $\mathbf{k} \leftarrow A \operatorname{Igorithm}(A, \theta)$ 4. $M \leftarrow V_k$ 5. Build the weighted matrix $[wV_k]$ For $j \leftarrow 1$ to k $w_j \leftarrow \lambda_j/2\Sigma_{j-1}^r\lambda_j$ $[wV_k]_{-k_j} \leftarrow w_j * [M]_{+,j}$ end for 6. Compute the eighted score for each variable $wS_k^{(h)} = \Sigma_{j-1}^n w_j + t_{j,j} $ for all $i = \{1, 2,, m\}$. 7. Sort the variables according to their weights: $wS_1^{(h)} \ge \otimes S_m^{(h)} \ge \ge wS_m^{(h)}$ <i>end</i> N_Comp is Intervor & Number OF Samples Trained On		Algorithm 2 - Weighted Scores (WS)
N_Comp 1: Compute the Singular Value Decomposition $[U, S, V^T] \leftarrow SVD(A)$ 2: Compute the proportion of variance carried by each component For $j \leftarrow 1$ to r $\lambda_j \leftarrow (s_j^2/(n-1))$, where $s_j \in S$ end for 3: Identify the number k of principal components to retain $k \leftarrow Algorithm1(A, \theta)$ 4: $M \leftarrow V_k$ 5: Build the weighted matrix $[wV_k]$ For $j \leftarrow 1$ to k $w_j \leftarrow \lambda_j (\Sigma_j^r = \lambda_j)$ $[wV_k]_{*,j} \leftarrow w_j * [M]_{*,j}$ end for 6: Compute the eighted score for each variable $wS_i^{(k)} = [\Sigma_{j=k}^k w_j t_{i,j}]$, for all $i = \{1, 2,, m\}$. 7: Sort the variables according to their weights: $wS_i^{(k)} \ge wS_i^{(k)} \ge \ge wS_m^{(k)}$ end N_Comp 3: DEFOR & Number OF Samples Trained On		Input: $A \in \mathbb{R}^{n \times m}$, θ (cumulative variance explained) Output: $S_r \in \mathbb{R}^{n \times k}$ and has the top k most representative ranked features of A
N_Comp I. Compute the proportion of variance carried by each component For j ← 1 to r $\lambda_j \leftarrow (\hat{s}_j^*/(n-1))$, where $s_j \in S$ end for 3. Identify the number k of principal components to retain $k \leftarrow Algorithm1(A, \theta)$ 4. $M \leftarrow V_k$ 5. Build the weighted matrix $[wV_k]$ For j ← 1 to k $w_j \leftarrow \lambda_j / \Sigma_{j=1}^r \lambda_j$ $[wV_k]_{*,j} \leftarrow w_j * [M]_{*,j}$ end for 6. Compute the eighted score for each variable $wS_1^{(k)} = [\Sigma_{j=1}^k w_j t_{i,j}]$, for all i = {1, 2,, m}. 7. Sort the variables according to their weights: $wS_1^{(k)} \ge wS_i^{(k)} \ge \ge wS_m^{(k)}$ end M_Comp ts n Error & Number OF Samples Trained On		begin
N_Comp b For <i>j</i> ← 1 to <i>r</i> $\lambda_j \leftarrow (s_j^2/(n-1))$, where $s_j \in S$ end for 3: Identify the number k of principal components to retain $k \leftarrow Algorithm1(A, \theta)$ 4: $M \leftarrow V_k$ 5: Build the weighted matrix $[wV_k]$ For $j \leftarrow 1$ to k $w_j \leftarrow \lambda_j / \Sigma_{j=1}^r \lambda_j$ $[wV_k]_{k+j} \leftarrow w_j * [M]_{*,j}$ end for 6: Compute the eighted score for each variable $wS_1^{(k)} = [\Sigma_{j=1}^k w_j t_j (\lambda_j)]$, for all $i = \{1, 2,, m\}$. 7: Sort the variables according to their weights: $wS_1^{(k)} \geq wS_1^{(k)} \geq \geq wS_m^{(k)}$ <i>end</i> N_Comp ts n Error & Number OF Samples Trained On		[U, S, V^T] $\leftarrow SVD(A)$
n Error & Number OF Samples Trained On		2: Compute the proportion of variance carried by each component For $j \leftarrow 1$ to r $\lambda_i \leftarrow (s_i^2/(n-1))$, where $s_i \in S$
$\mathbf{k} \leftarrow \operatorname{Algorithm}(A, \theta)$ 4: $M \leftarrow V_k$ 5: Build the weighted matrix $[wV_k]$ For $\mathbf{j} \leftarrow 1$ to \mathbf{k} $w_j \leftarrow \lambda_j / \Sigma_{j=1}^r \lambda_j$ $[wV_k]_{*,j} \leftarrow w_j * [M]_{*,j}$ end for 6: Compute the eighted score for each variable $wS_i^{(k)} = \Sigma_{j=1}^k w_j t_{i,j} $, for all $\mathbf{i} = \{1, 2,, m\}$. 7: Sort the variables according to their weights: $wS_1^{(k)} \ge \otimes W_m^{(k)} \ge \ge wS_m^{(k)}$ end N_Comp ts n Error & Number OF Samples Trained On		end for3: Identify the number k of principal components to retain
S: Build the weighted matrix $[wV_k]$ For $j \leftarrow 1$ to k $w_j \leftarrow \lambda_j / \sum_{j=1}^r \lambda_j$ $[wV_k]_{*,j} \leftarrow w_j * [M]_{*,j}$ end for 6: Compute the eighted score for each variable $wS_i^{(k)} = \Sigma_{j=1}^k w_j t_{i,j} $, for all $i = \{1, 2,, m\}$. 7: Sort the variables according to their weights: $wS_1^{(k)} \ge wS_i^{(k)} \ge \ge wS_m^{(k)}$ end N_Comp ts n Error & Number OF Samples Trained On		$k \leftarrow Algorithm1(A, \theta)$ 4: $M \leftarrow V_{i}$
n Error & Number OF Samples Trained On		5: Build the weighted matrix $[wV_k]$ For $i \leftarrow 1$ to k
$ W_k _{i+j} \leftarrow W_j * W_j _{i+j}$ end for 6: Compute the eighted score for each variable $WS_i^{(k)} = \Sigma_{j=1}^k w_j t_{i,j} $, for all $i = \{1, 2,, m\}$. 7: Sort the variables according to their weights: $wS_i^{(k)} \ge wS_i^{(k)} \ge \ge wS_m^{(k)}$ end N_Comp ts		$ \begin{array}{c} \text{ror} \mathbf{j} \leftarrow \mathbf{i} \text{ to } \mathbf{k} \\ w_j \leftarrow \lambda_j / \Sigma_{j=1}^r \lambda_j \\ [wW_j] (w, w) \neq [M] \end{array} $
 b. Complete the equators core for each variable wS_i^(k) = Σ_j^k=uy_jt_{i,j} , for all i = {1, 2,, m}. 7: Sort the variables according to their weights: wS₁^(k) ≥ ≥ wS_m^(k) end 		$[wv_k]_{*,j} \leftarrow w_j * [w]_{*,j}$ end for
7: Sort the variables according to their weights: $wS_1^{(k)} \ge wS_i^{(k)} \ge \ge wS_m^{(k)}$ end N_Comp its		6: Compute the eighted score for each variable $wS_i^{(k)} = \Sigma_{j=1}^k w_j t_{i,j} $, for all $i = \{1, 2,, m\}$.
n Error & Number OF Samples Trained On		7: Sort the variables according to their weights: $wS_1^{(k)} \ge wS_i^{(k)} \ge \ge wS_m^{(k)}$
N_Comp ts		end
N_Comp tts		
n Error & Number OF Samples Trained On	N_Comp	
n Error & Number OF Samples Trained On	ts	
n Error & Number OF Samples Trained On		
n Error & Number OF Samples Trained On		
n Error & Number OF Samples Trained On		
n Error & Number OF Samples Trained On		
n Error & Number OF Samples Trained On		
on Error & Number OF Samples Trained On		
on Error & Number OF Samples Trained On		
on Error & Number OF Samples Trained On		
n Error & Number OF Samples Trained On		
	n Error & N	umber OF Samples Trained On

Gisette

Madelon

IOT Intrusion

Conclusion

In this paper we propose an effective feature reduction technique that uses Principal Component Analysis and Singular Value Decomposition. It leverages the statistics of the principal components to identify the features that retain the maximum variability of the data, helping to reduce the reconstruction error. Our experiments conducted on various public datasets shows that while our method picks the topmost representative k features by the validated the accuracy, reconstruction error values shows that not much information is lost in the transformation the and process performance of the model on real IoT data shows that our algorithm performed well. The number of Principal components to retain should also be careful decided as we discovered at the number of components, the higher the reconstruction error and lower the accuracy. However, the margin of interest lies at the region where significant increase in the number of PCs, has little effect on the accuracy and reconstruction error. This makes the case for industry wide adoption of the process.

References

[1] R. Kavitha and E. Kannan, "An efficient framework for heart disease classification using feature extraction and feature selection technique in data mining," 2016 International Conference on Emerging Trends in Engineering, Technology and Science (ICETETS), Pudukkottai, India, 2016, pp. 1-5.

[2] M. F. I. Ibrahim and A. A. Al-Jumaily, "PCA indexing based feature learning and feature selection," 2016 8th Cairo International Biomedical Engineering Conference (CIBEC), Cairo, Egypt, 2016, pp. 68-71.

[3] Y. Li, K. Shi, F. Qiao and H. Luo, "A Feature Subset Selection Method Based on the Combination of PCA and Improved GA," 2020 2nd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI), Taiyuan, China, 2020, pp. 191-

[4] C. Yumeng and F. Yinglan, "Research on PCA Data Dimension Reduction Algorithm Based on Entropy Weight Method," 2020 2nd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI), Taiyuan, China, 2020, pp. 392-

[5] H. Shah and K. Verma, "Voltage stability monitoring by different ANN architectures using PCA based feature selection," 2016 IEEE 7th Power India International Conference (PIICON), Bikaner, India, 2016, pp. 1-6.

[6] A. P. Kale and S. Sonavane, "PF-FELM: A Robust PCA Feature Selection for Fuzzy Extreme Learning Machine," in IEEE Journal of Selected Topics in Signal Processing, vol. 12, no. 6, pp. 1303-1312, Dec. 2018.