Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

RandPro- A practical implementation of random projection-based feature extraction for high dimensional multivariate data analysis in R

R. Siddharth [*], G. Aghila

National Institute of Technology Puducherry, Karaikal, India

A R T I C L E   I N F O

Article history:
Received 15 July 2020
Received in revised form 5 November 2020
Accepted 13 November 2020

Keywords:
Dimension reduction
Johnson–Lindenstrauss lemma
RandPro
R Programming
Software description

A B S T R A C T

The performance of the high dimensional multivariate data analysis is seriously affected by the curse of dimensionality. Feature extraction acts as an important pre-processing step in data analysis process to avoid the curse of dimensionality. Random projection method is the most underrated feature extraction technique that performs extremely well in the case of high dimensional data analysis. This technique is known for its characteristics like data independent projection, simpler computation and distance preserving property. The Johnson–Lindenstrauss lemma is the idea behind random projection method. It states that the small set of points in the high dimensional space can be embedded into smaller subspace and also approximately preserves the distance with higher probability. This article describes a practical implementation of random projection method in the popular statistical programming language R and it is compared with the other similar implementations. The software package for random projection method has been uploaded in Comprehensive R Archive Network(CRAN) repository as *RandPro* and the code has been distributed in github. The RandPro package is tested with different types of data including text, image and sensor data. The result shows that the RandPro package preserves the pairwise distance between the data points in the corresponding low dimensional space for further processing.

Code metadata

Current code version	Version 0.2.2
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-20-00006
Codeocean compute capsule	https://codeocean.com/capsule/5704360
Legal Code License	GPL-2 – GPL-3
Code versioning system used	git
Software code languages, tools, and services used	R
Compilation requirements, operating environments & dependencies	R Packages: caret, stats, e1071
If available Link to developer documentation/manual	https://cran.r-project.org/web/packages/RandPro/RandPro.pdf
Support email for questions	r.siddharthcse@gmail.com

1. Motivation and significance

Most of the data collected in the real world (transcriptional data, mass spectrometry, DNA micro array, large hyper-spectral images, time series data, etc.) are of multivariate data. The state-of-the-art applications use multivariate data analysis to extract the information from high dimensional input data. Generally, high dimensional data negatively impacts the performance of conventional multivariate statistical methods. For example, consider a drug industry that uses high dimensional mass spectrometry imaging (MSI) data to classify the diseases. The observations from images have dimensions ranging from thousands to millions. The important task for the data scientist is to detect the diseases from high dimensional data using statistical techniques. The number of dimensions in the input data plays a vital role while analyzing the performance. The higher number of dimensions negatively impacts the running time of the underlying

[*] Corresponding author.
E-mail address: r.siddharthcse@gmail.com (R. Siddharth).

classification algorithm. Due to the noisy and uninformative feature set it may affect the accuracy of classifier. These problems are generally called as "curse of dimensionality" in statistics and machine learning [1]. Structural simplification or dimension reduction is one of the indispensable prerequisite of multivariate data analysis for reducing the number of dimension in the high dimensional space [2]. Feature extraction is a widespread structural simplification method where the original high dimensional input data is approximated with fewer dimensions, still maintaining the same structure as the original data. It acts as a vital pre-processing step in several applications like MSI analysis, text analysis, bio-informatics, etc to project the high dimensional data in a low dimensional space [3].

Random Projection (RP) method reduces the complexity of multivariate data by projecting the high dimensional input data into low dimensional subspace by approximately preserving the distance between data points with higher probability. RP method achieves the distance preserving property based on Johnson–Lindenstrauss (JL) lemma. The lemma states that when 'P' points in high dimensional data are randomly projected into lower $O(\frac{logP}{\epsilon^2})$ dimension then the pairwise distances between the points are distorted only in the range of $(1 \pm \epsilon)$. RP method is one of the less explored feature extraction technique when compared with the most prominent method like Principal component analysis (PCA) [4]. PCA is an extremely useful method in data analysis where the application goes far beyond the dimension reduction and it has an excellent maximum variance interpretation. In comparison, both RP method and PCA projects the high dimensional data into low dimensional space by multiplying the data with projection matrix. The key difference is the direction of the projection in PCA is dependent on the input data whereas it is completely random in RP method. It implies that the projection matrix of PCA is generated based on the input feature set. In contrast, the projection matrix of RP method is filled with independent and identically distributed (IID) random values just by knowing only the number of dimension. Even though the projection is random, RP method nearly preserves all the pairwise distance between any two samples in the projected low dimensional subspace with the controlled amount of error. Albeit having many applications, RP outperforms PCA in the following use cases:

Case 1- Speed up in high dimensional data analysis: The process of selecting principal components in PCA is an expensive operation. The steps to perform PCA are (i) Calculate the covariance matrix from the original dataset (ii) Deduce the Eigenvalues and Eigenvectors from the covariance matrix (iii) Create the projection matrix by selecting the first 'n' principal components (iv) project the high dimensional input data by multiplying with the projection matrix [5]. While projecting the high dimensional data in low dimensional space, RP method requires less computational resources when compared to the PCA due to its randomness and data independent property. The process of RP method is fairly simple when compared to PCA. It includes (i) The number of dimensions required for projection is calculated based on Johnson–Lindenstrauss lemma (ii) Create the projection matrix filled with IID random values (iv) Project the high dimensional input data by multiplying with the projection matrix. Due to this simpler computation, RP method is best suited for applications that require speed up in high dimensional data analysis.

Case 2- Analyzing the data streams: Internet of things and sensor applications does not provide all the data at once. The data has to be updated over certain period of time. For every time interval, PCA has to find the corresponding principal components in order to project the data [6]. But in case of RP method, the same projection matrix can be reused for entire data due to its data independent property. This makes the RP method as a viable option for analyzing the data streams [7].

The RP method is applied in numerous applications like dimension reduction, dimension expansion, matrix completion and privacy preservation between the features in distributed estimation [8]. Ref. [9] discusses the theoretical background and its relevant hypothesis tests based on RP. Albeit the requirement of RP method is vast in many applications, the term underrated is used because of the lack of practical implementations. In comprehensive R archive network (CRAN) repository, the implementation of other feature extraction techniques like principal component analysis (PCA), linear discriminant analysis (LDA) [10], singular value decomposition (SVD) [11] are available. But the straightforward implementation of RP method based feature extraction is not available in one of the widely used statistical programming language R. The other implementations in scikit-learn library for Python [12] and WEKA [13] also lacks some of the important features of RP method. The main objective of the proposed R package is to provide the basic functionalities of RP method such as: (i) finding the minimum number of dimensions required to project the high dimensional data using JL lemma (ii) generating the projection matrix using the available standard distributions. The implementation of RP method based feature extraction and classification has been uploaded into the CRAN repository in the name of *RandPro* [14]. This package assists the R users to perform RP method based feature extraction with ease and simple functions. The key contribution of the proposed work are

- Identify and implement the core functionalities of RP method to reduce the learning burden of novice R users.
- The built-in classifier helps to perform both feature extraction and classification in a single function.
- Provide options for R user to create random matrix with four widely used distribution.

The article is organized as follows. Section 2 provides a brief review of RP method and its properties and also discusses the proof of JL lemma. Section 3 provides a detailed description of functions that are available in the RandPro package followed by a comparison chart. This chart highlights the uniqueness of RandPro package over other similar implementations. Section 4 discusses about the evaluation of RandPro package over different dataset and Section 5 concludes the article with future scope.

2. Statistical background

RP method is one of the projection based feature extraction techniques used for structural simplification by projecting the data from high dimensional into low dimensional space without distorting too much of information. As the name suggests, this method chooses a random low dimensional subspace for projecting the high dimensional data [15]. Consider an input data matrix D of N rows and P columns D_{NxP}. The original data matrix is projected into a k-dimensional subspace using a random matrix of P rows and k columns R_{Pxk}. The projection of data is represented as $P_{Nxk} = D_{NxP}.R_{Pxk}$. In RP method, the complexity of the operation is $O(kNP)$ when the matrix is dense. If the matrix is sparse and filled with c non-zero entries then the complexity is $O(ckP)$. Johnson–Lindenstrauss (JL) transform acts as the basic guideline for the RP method. In random matrix R_{Pxk}, the value of k has been identified using JL lemma.

2.1. Johnson–Lindenstrauss lemma

The JL lemma states that the small set of data points in the original high dimensional space can be embedded into an arbitrary low dimensional space in such a way that the distances between the points are nearly preserved [16]. The literature shows that the lemma has been successfully applied in dimension reduction, compressed sensing, manifold learning and graph embedding [17]. The lemma states that

Lemma. *Given an error tolerant value $0 < \epsilon < 1$ and a number of data points N, let k be a positive integer such that*

$$k \geq \frac{24}{3\epsilon^2 - 2\epsilon^3} \log P \tag{1}$$

then for any set A of N points $\in \Re^P$ there exists a linear map $f : \Re^P \to \Re^k$ such that for P-dimensional data matrix with N points for all $x_i, x_j \in A$

$$(1-\epsilon)\|x_i - x_j\|^2 \leq \|f(x_i) - f(x_j)\|^2 \leq (1+\epsilon)\|x_i - x_j\|^2 \tag{2}$$

Notations.

- P: Number of features in the high dimensional input data A
- N: Number of samples or data points in the high dimensional input data A
- ϵ: Error tolerant value where $0 < \epsilon < 1$
- k: Number of dimensions required to projected the data where $k < P$
- f: Projection function
- $\|x_i - x_j\|^2$: Pairwise ℓ_2 distance between the ith and jth column of A

After projection, the distance between any two points is distorted within the factor of $(1 \pm \epsilon)$ [18]. The way to create the projection function is fairly simple and independent of the original data matrix A.

$$f = \frac{1}{\sqrt{k}} R \tag{3}$$

where R is the random projection matrix: $R_{i,j} \overset{IID}{\sim} \mathcal{N}(0, 1)$.

2.2. Uniqueness of RP method

The unique properties of the RP method makes this to stand apart from the rest of other feature extraction algorithms to work for high dimensional multivariate data analysis. The four important properties are:

2.2.1. Data independent property

The Eq. (1) implies that the value of k is based only on the number of features in the original data P and error tolerant value ϵ. As k is independent of the original data A and its number of samples. The projection function f can use the same projection matrix R and k for any data A as long as they have the same P and ϵ. The target embedding space can be chosen without looking at the individual samples from the original set A [19]. The simplicity and universality characteristics of this technique makes this as one of the powerful tool for multivariate high dimensional data analysis. The data independent property of the RP method helps the data stream analysis (IoT sensor data analysis) to reuse the same projection matrix for all the dataset collected from different time interval $t_1, t_2 \ldots t_x$.

2.2.2. Distance preserving property

RP method reduces the dimension of the high dimensional data while approximately preserves the Euclidean distance between any two data points in the corresponding lower dimension with larger probability. The distance preserving property of JL lemma is examined by relating distance and affinity to provide a theoretical guarantee [20].

2.2.3. Dimension expansion

From Eq. (1), it is identified that the k is a function of P and ϵ. For example, when P=10 and $\epsilon = 0.2$ then the value of k is 531. This implies that the value of P must be large enough to achieve dimension reduction for such ϵ value. But the value of P can also be smaller than k and such low distortion transform is actually used in neural networks for dimension expansion in-order to ensure the sparsity of data in the output space [21].

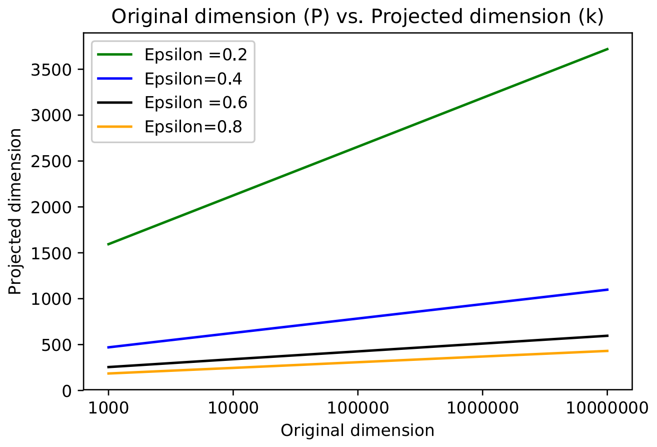


Fig. 1. The impact of different ϵ value over the original dimension 'P'. The projected dimension 'k' is calculated based on P and k using Eq. (1).

2.2.4. Controlled amount of error

The ϵ is an error tolerant parameter which also plays a vital role in RP method to control the distortion of distance between the data in high dimensional space and its projected low dimensional space. The value of ϵ along with P is used to calculate the corresponding low dimensional size k. The ϵ value can be adjusted by the user where the higher ϵ value indicates that the user is ready to accept more distortion of distance and it requires less number of dimension to project the data. The impact of different ϵ value over various input dimension is plotted in Fig. 1. The X-axis shows the input dimension (P) and the value of projected dimension (k) in Y-axis is calculated using Eq. (1) with four different ϵ values: 0.2, 0.4, 0.6 and 0.8. The figure clearly shows the impact of ϵ value and it is inversely proportional to the value of k. Hence the user can control the error by adjusting the ϵ value based on their preferences.

Apart from these four unique properties, RP method is also known for its simpler computation. The linear time complexity of RP method outperforms most of the other feature extraction algorithms like PCA, SVD and LDA that have quadratic time complexity. The next section discusses about the proposed RandPro package that contains the implementation of RP method based feature extraction and classification in R language.

3. Software description

R is one of the top 5 statistical programming language that holds a large user base in the data science community. The popularity of sparkR shows the importance and necessity of R in the industrial research for solving big data problems [22]. The RandPro package evolved with the motivation to carry out dimension reduction using RP method in R. The mainstream machine learning tool kit "WEKA" and Python's scikit-learn have their own implementations of the RP method. The RandPro package (version 0.2.2) in the CRAN repository provides the core functionalities of RP method with built-in classifier.

3.1. Functions in RandPro

The key functions of the RandPro package are:

1. dimension() - determines the number of dimension required to project the data in low dimensional space using Johnson–Lindenstrauss lemma.
2. form_matrix() - creates a projection matrix filled with four suitable distributions that are primarily used in the research community.

3. classify() - performs the classification with RP method based feature extraction.

3.1.1. Dimension()

The *dimension()* function is used to find the minimum number of dimension required to project the data from high dimensional space to low dimensional space. The function uses Eq. (1) to find the k value. The function takes two arguments (i) the number of data points in high dimensional space as a mandatory input argument and (ii) the optional epsilon (ϵ) value. RP creates the projection matrix even before the arrival of input data by just knowing only the number of dimension. The default value of ϵ is 0.1 and it must be in the range of 0 to 1. ϵ is the measurement for approximate distortion of distance between the high dimensional space and its corresponding low dimensional space. The ϵ value 0.05 means the deviation of distance between the high dimensional and low dimensional data should not exceed 5%. The larger ϵ value requires less number of dimension to project the data because of the acceptance of deviation between the distances. Hence the value of ϵ is inversely proportional to the computational requirement. The basic description of the dimension() function is explained in Listing 1.

```
#Install RandPro package
devtools::load_all()

#load library
library(RandPro)

#Calculate minimum dimension using eps =0.5 for 1000000 sample
k <- dimension(1000000,0.5)


#Calculating minimum dimension using different epsilon value for
       1000000 sample
eps <- c(0.5,0.1)
k<- dimension(1000000,eps)
```

Listing 1: Examples of dimension() function in RandPro package

3.1.2. Form_matrix()

RP method does not project the data into a completely arbitrary random low dimension subspace rather it have an underlying structure based on the distribution used to fill the elements of the random projection matrix R. This function takes five input arguments: number of rows, columns and JLT (JL transform) are mandatory inputs. JLT is a Boolean variable where TRUE indicates the function to use dimension() for finding the size of low dimensional projection. The ϵ value is optional and only activated when JLT is set to TRUE. The default value of ϵ is 0.1 and default projection distribution is Gaussian. The form_matrix() function creates the projection matrix filled with IID values using any one of the listed four widely used projection distributions:

1. *gaussian* - The random projection matrix R has been filled with the Gaussian distribution $\mathcal{N}(0, 1/min)$, where 'min' is the minimum number of dimensions and $\mathcal{N}$ is the density function of the distribution. In probability theory, Gaussian distribution is a continuous probability distribution used to represent real-valued random variables whose distributions are not known. The rows of the projection matrix are orthogonal to each other. The probability density function of Gaussian distribution is given in Eq. (4). Where μ is mean, σ is standard deviation and σ^2 is variance.

$$f(x \mid \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{4}$$

2. *probability* - The projection matrix R has been filled with the entries [+1, −1] of equal probability based on Eq. (5)

$$R_{ij} = \begin{cases} +1 & \text{p} = 1/2 \\ -1 & \text{p} = 1/2 \end{cases} \tag{5}$$

3. *achlioptas* - Ref. [23] proposed sparse random distributions where the matrix has been formed using Eq. (6). The $\frac{2}{3}$rd of the matrix is filled with zero makes it more sparse and reduce the computation even more.

$$R_{ij} = +\sqrt{3} \begin{cases} +1 & \text{p} = 1/6 \\ 0 & \text{p} = 2/3 \\ -1 & \text{p} = 1/6 \end{cases} \tag{6}$$

4. *li* - Ref. [24] proposed a very sparse RP method, where the projection matrix R is generated using the entries -1,0,+1 with the probability $\{\frac{1}{2\sqrt{D}}, 1-\frac{1}{\sqrt{D}}, \frac{1}{2\sqrt{D}}\}$. This method achieves $\sqrt{D}$−fold speedup with marginal loss of accuracy, where D is dimension of the data. This method further increases the computational speed by setting s = $\sqrt{D}$. Here dimension D is directly proportional to the speed up.

$$R_{ij} = +\sqrt{s} \begin{cases} +1 & \text{p} = 1/2\text{s} \\ 0 & \text{p} = 1 \text{ - } 1/2\text{s} \\ -1 & \text{p} = 1/2\text{s} \end{cases} \tag{7}$$

Listing 2 provides various examples for initializing different parameters in the form_matrix() function.

3.1.3. Classify()

The classify() function allows the R users to combine the task of RP method based dimension reduction as well as classification within a single function. The built-in classifier eases the task by removing the additional step of pre-processing before classification in a same function. This function takes four mandatory parameters such as training data, test data, class labels of training data and class labels of test data. The optional parameters are ϵ, distributions to generate random projection matrix and classification algorithm. This function supports classification algorithms like k-nearest neighbor(k-NN), support vector machine(SVM) and naive bayes classifier. This classify() utilizes both dimension() and form_matrix() to get the number of dimension for projecting the training data and test data and generating the projection matrix respectively. The training data and test data has been projected into the low dimensional space by multiplying with the projection matrix. At last the reduced matrix was given to the classifier. The confusion matrix is the output of the classify() where we can calculate the performance of the classifier including accuracy, sensitivity, specificity, etc. The Listing 3 illustrates how the classify() works with the human activity recognition (HAR) data and the executable code along with the data is available in the codeocean capsule mentioned in Code Metadata.

The more details and examples about each function is available in the reference manual of the RandPro package that can be accessed from the CRAN repository.[1]

3.1.4. Comparison of implementation of RP method

From the perspective of statistician, the practical implementation of RP method is available in popular languages like Python, R and WEKA tool. The Python scikit machine learning package comes with basic implementation of RP method with less add-on features. In WEKA machine learning tool, the basic functionalities like finding the number of points for low dimensional projection (k) using JL lemma is not available. Here users have to assume the 'k' value which is not helpful for all the users. The another R package RPEnsemble [25] also uses RP method for ensemble classification. The core functionalities of RP method has been taken in-depth consideration while implementing RandPro package which is not the case for RPEnsemble package. The focus of RPEnsemble package is ensemble-based classification which is

[1] https://cran.r-project.org/web/packages/RandPro/index.html

```
# Load Library
library(RandPro)

# Matrix filled with Default Gaussian distribution without applying JL transform where the size of the matrix is same as original.
mat <- form_matrix(600,1000,FALSE)

# Matrix filled with Default Gaussian distribution and the JLT is set TRUE with default $\epsilon$ value 0.1. Here the size of the output
      matrix is reduced using JL transform with 10% error tolerant.
mat <- form_matrix(300,100000,TRUE)

# Matrix filled with probability distribution and the JLT is set TRUE with $\epsilon$ value 0.5. Here the size of the output matrix is
      reduced using JL transform with 50% error tolerant.
mat <- form_matrix(250,1000000,TRUE,0.5,"probability")

# Matrix filled with li distribution and other paramters are same like above example.
mat <- form_matrix(250,1000000,TRUE,0.5,"li")

# Matrix filled with achlioptas distribution without applying JL transform.
mat <- form_matrix(250,1000000,FALSE,"achlioptas")
```

Listing 2: Examples of form_matrix() function with different parameters

```
#Load Library
library(RandPro)


#Read HAR training data and convert into matrix
train_data <- scan('/data/X_train.txt')
train_data <- matrix(train_data,ncol = 561,byrow = TRUE)
dim(train_data)


#Read HAR test data and convert into matrix
test_data <- scan('/data/X_test.txt')
test_data <- matrix(test_data,ncol = 561,byrow = TRUE)
dim(test_data)

#read HAR training label and convert into matrix
train_label <- scan('/data/y_train.txt')
train_label <- matrix(train_label,ncol = 1,byrow = TRUE)
dim(train_label)

#read HAR test label and convert into matrix
test_label <- scan('/data/y_test.txt')
test_label <- matrix(test_label,ncol = 1,byrow = TRUE)
dim(test_label)


#Factor train label and test label to categorize the resultant actions
train_label <- factor(train_label)
test_label <- factor(test_label)


# Classify HAR data with random projection where epsilon value is 0.75, projection matrix is generated using Gaussian distribution and
      classified using k-Nearest neighbor classifier.

result <- classify(train_data = train_data,test_data = test_data,train_label = train_label,test_label = test_label,eps = 0.75,projection = '
      gaussian',classifier = 'knn')

#Print Confusion matrix
result
```

Listing 3: Classification of HAR data with k-NN classifier using the classify() function in RandPro package

different from the focus of RandPro package. The comparison of RandPro package and other similar implementations are listed in Table 1 that highlights the uniqueness of RandPro package. The important features of the RP method has been compared over different implementations. The RandPro package is the right mix of strong core functions and usable add-on functions.

4. Illustrative examples

This section clearly illustrates the real world implementation of RP method based feature extraction and classification using RandPro R package. The experiments have been carried out in the system with 8th generation Intel i7 processor with 16GB of RAM and 256GB of SSD storage. The latest version of R (V 3.5) environment for windows has been used for evaluation. This section further discusses the different types of standard datasets used for investigating the performance of RandPro package and its corresponding results.

4.1. Datasets

The RandPro package aims to perform the structural preservation for almost all kind of widely used data including text data, image data and sensor data. Table 2 lists the well-known and standard dataset that are used for testing the performance of the RandPro package. HAR is a sensor data collected from smartphone to recognize the human activity. 30 volunteers performed six activities (Walking, Walking_Upstairs, Walking_Downstairs, Sitting, Standing and Laying) wearing a smartphone on their waist. 70% of the data was selected for training the classifier and 30% to test the classifier [26]. Fashion MNIST is an image dataset that consist of 70 000 gray scale images where each image is 28 * 28 for a total of 784 pixel information. The pixel value is an integer between 0 to 255. The training set consist of 60 000 images and test set consist of 10 000 images. Each image is assigned to any one of the following class labels: T-shirt/Top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag and Ankle boot [27]. REUTERS-5 (referred as Reuters) is one of the widely accepted text categorization dataset that was originally developed by Carnegie group and Reuters. The dataset consist of 5146 documents for training and 2062 documents for testing. Each document is pre-processed by changing the text to lower case and removing the numbers, stop words, punctuation and white spaces from the texts. There are five class labels: acq, crude, earn, money-fx and trade.

4.2. Evaluation metrics

The literature already proved that the feature extraction technique obviously reduces the time and space complexity of the

Table 1
Comparison of different implementation of RP method.

S.No.	Features	RandPro (R)	RPEnsemble (R)	sklearn.random_projection (Python)	Random Projection (WEKA)
1.	Function for JL lemma to find 'k' value.	Present	Not present	Present	Not present
2.	Function to generate the projection matrix based on "Gaussian Distribution"	Present	Present	Present	Present
3.	Function to generate the projection matrix based on "Probability Distribution"	Present	Not present	Not present	Not present
4.	Function to generate the projection matrix based on "Achlioptas Distribution"	Present	Not present	Present	Present
5.	Function to generate the projection matrix based on "Li Distribution"	Present	Not present	Present	Present
6.	Inbuilt classifier	Present	Present	Not present	Not present

Table 2
Dataset.

Dataset	Dimension	Type
HAR	10 299 * 561	Sensor data
MNIST Fashion	70 000 * 784	Image data
Reuters	7208 * 22 034	Text data

original high dimensional data [28]. So the comparison of time and space complexity between original data and projected data is not the optimal choice to prove the validity of RandPro package. Hence the performance evaluation is based on how well the properties of RP method is maintained in the RandPro package. The two important parameters are:

- The amount of original information is preserved in the projected low dimensional space.
- The performance of classifier in terms of running time, accuracy and F1-score with RandPro package.

The above two parameters are evaluated using the given standard metrics

1. Distance Distortion: Comparing the distance between data points in the original high dimensional space and its corresponding projected low dimensional space helps to identify how much original information is preserved in the projected space. This pairwise distance distortion is measured using Eq. (8)

$$Distance\_distortion = \sum_{x_i,x_j \in A} \frac{\|f(x_i) - f(x_j)\|^2 - \|x_i - x_j\|^2}{\|x_i - x_j\|^2} \tag{8}$$

2. Accuracy:

$$Accuracy = \frac{NCP}{TNP} \tag{9}$$

where NCP is number of correct prediction and TNP is total number of prediction.

3. F-score: To find the harmonic average of precision and recall for the classifier.

$$\begin{aligned} F-score &= 2 * \frac{precision * recall}{precision + recall}, \\ precision &= \frac{TP}{TP+FP}, \\ recall &= \frac{TP}{TP+FN} \end{aligned} \tag{10}$$

where TP - true positive, FP - false positive and FN - false negative

4.3. Results

This section shows the practical significance of RandPro package and the results are divided into two subsections based on the evaluation parameters listed above.

4.3.1. Distance distortion distribution

Fig. 2 represents the pairwise distance distortion between the original high dimensional data and the projected low dimensional data of HAR (Figs. 2(a), 2(b)), MNIST (Figs. 2(c), 2(d)) and Reuters data (Figs. 2(e), 2(f)). The number of dimensions in the projected space has been calculated with $\epsilon = 0.5$ and random matrix has been generated using Gaussian distribution. The X-axis in the figure represents the distortion between the data and Y-axis represents the number of bins in the histogram of particular distortion. This result shows that the distortion between data is not more than 0.25 and it implies that the distance preserving property of RP method has been successfully carried out using RandPro package for all the three datasets.

4.3.2. Performance of classifier

This experiment utilizes the *classify()* function in the RandPro package to perform RP method based classification. Based on the output confusion matrix, the performance of the classifier is evaluated in terms of two important metrics accuracy and F-score. The k-nearest neighbor(k-NN) classifier is used to create the model and results from the classifier are fine tuned with repeated 5-fold cross validation over 20 iterations. The value of 'k' in the k-NN algorithm is selected based on the repeated cross validation with different 'k' values. Initially all the three datasets are classified using k-NN(k = 7 for HAR and MNIST, k = 9 for Reuters) with original dimensions. The accuracy of HAR, MNIST and Reuters data are 90.26, 82.68 and 89.93 respectively. The F-score of HAR, MNIST and Reuters data are 89.88, 82.15 and 89.33 respectively. Later the original data is projected into the low dimensional space and classified using the same k-NN algorithm with different ϵ value and distribution. Fig. 3 shows the accuracy and F-score of HAR, MNIST and Reuters data with ϵ values of 0.5,0.7 and 0.9. The x-axis in resultant graph denotes distributions used to generate the random projection matrix R. The y-axis of Figs. 3(a), 3(c) and 3(e) denotes the accuracy of classifier measured in percentage and the y-axis of Figs. 3(b), 3(d) and 3(f) denotes the F-score of classifier measured in percentage. The results show that the accuracy and F-score of the k-NN classifier is almost equivalent to the original high dimensional input data with different distributions and ϵ value. In terms of running time, the k-NN classification of HAR data without any dimension reduction took approximately 9594 s whereas PCA-based feature extraction took approximately 2455 s and the proposed classify() function with RP method based feature extraction took only 398 s approximately. proc.time() function has been used to calculate the process time in seconds.

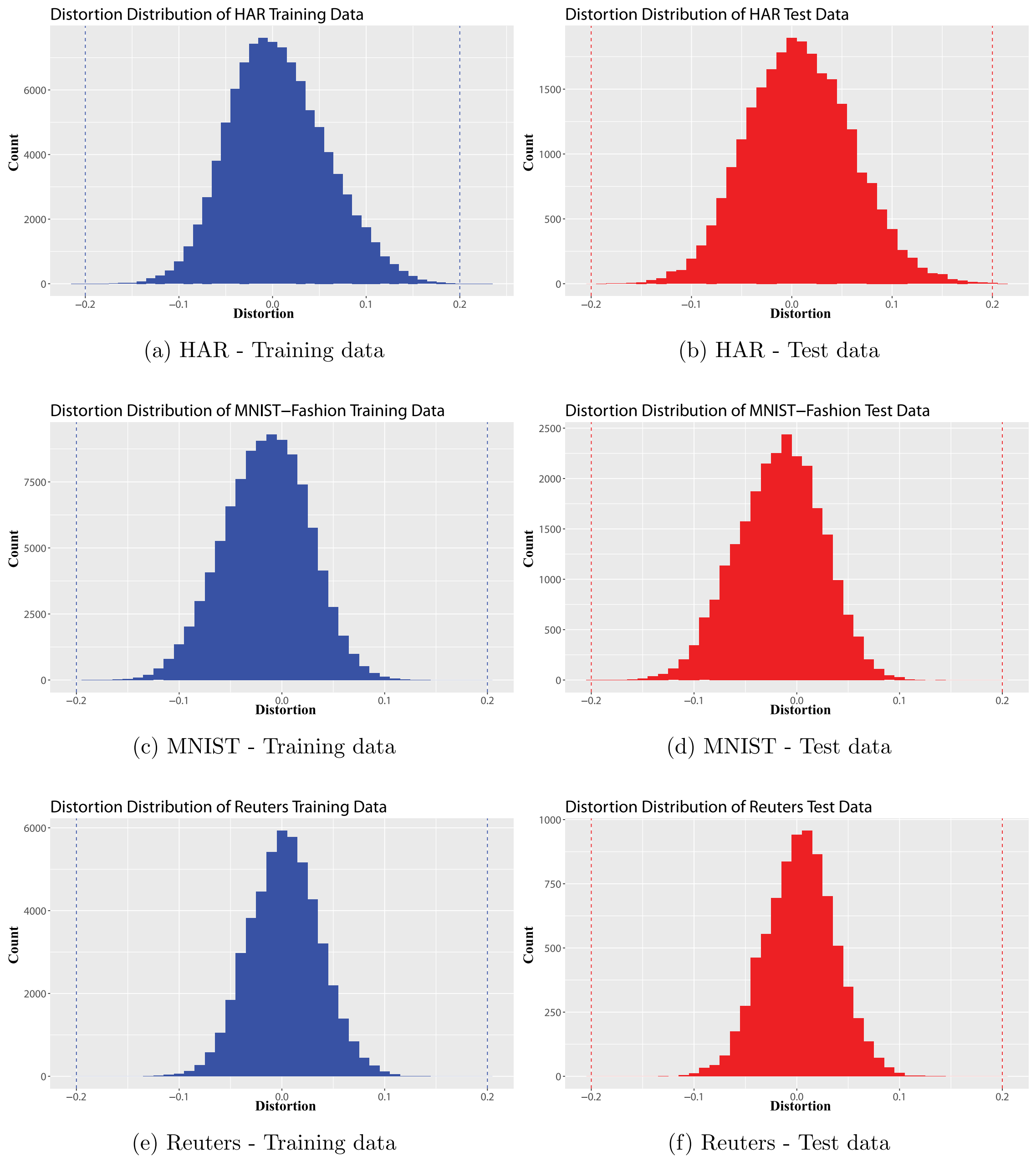
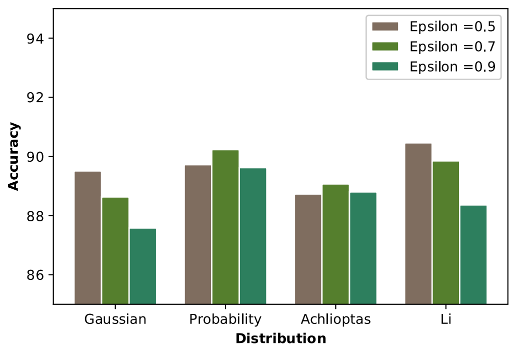


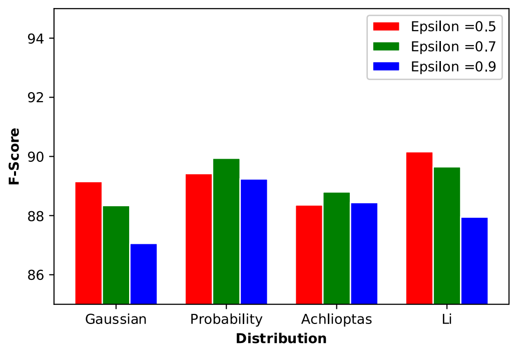
Fig. 2. Histogram representation of distance distortion distribution over the samples in HAR, MNIST and Reuters data with $\epsilon = 0.5$ and the difference in distance is calculated using Eq. (8).

Other than PCA, the execution time of the proposed RandPro package is compared with other similar implementation in Python and WEKA. For fair comparison, the basic functionality of the random projection is divided into three functions and the execution time of each function is compared. Table 3 lists the three function where function-1 is to identify the size of low dimensional subspace using JL lemma. Function-2 is to project the original high dimensional data 'P' into low dimensional 'k' using RP method and the impact of different dataset length 'P' is analyzed. Function-3 performs k-NN classification over the projected HAR training and test data. In this experiment, the ϵ value is 0.75 and projection matrix R is filled with Gaussian distribution. Since the Function-1 & 3 are not available in WEKA and Function-3 is not available in Python, comparing the execution time of each function is the best way to show the results. The primary computation of random projection is matrix multiplication. This experiment utilizes the conventional way of multiplying two matrices using nested for loop to avoid biased comparison. From the execution time presented in Table 3, it is identified that the WEKA is faster than RandPro package and Python's scikit-learn. The speedup in WEKA is mainly due to the underlying Java language. Java is a compiled language and generally faster than R and Python in terms of loop performances. On other hand, Python and R are interpreted language where the loop performance is comparatively slower. It is also observed that the RandPro R package have superior loop performances over Python's scikit-learn. In addition to the results, the other libraries and operators for matrix multiplication are studied. The %*% operator in R for matrix multiplication is faster than WEKA and numpy implementation in python (numpy.matmul()) is faster than both %*% operator and WEKA.

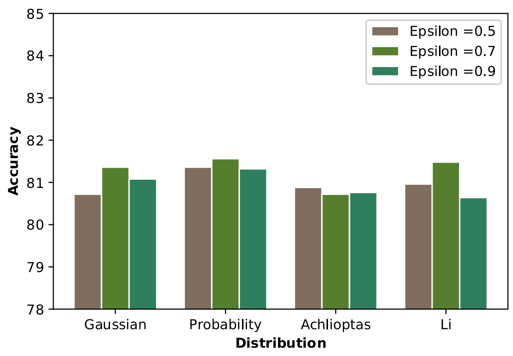
The results show that RandPro package maintains the original information in the projected space with the near equivalent accuracy and F-score. The objective of RandPro package is achieved



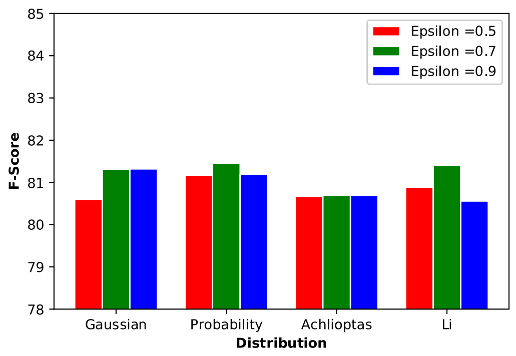
(a) HAR - Accuracy



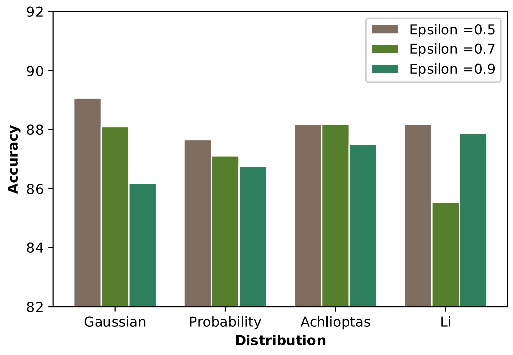
(b) HAR - F-Score



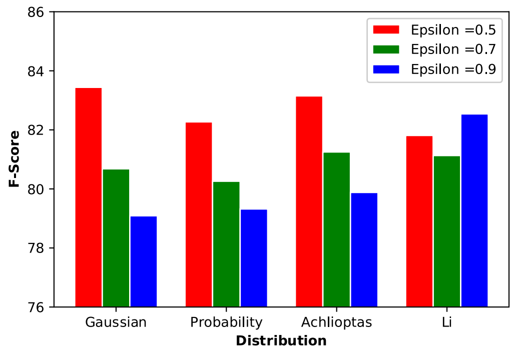
(c) MNIST - Accuracy



(d) MNIST - F-Score



(e) Reuters - Accuracy



(f) Reuters - F-Score

Fig. 3. Accuracy and F-Score of k-NN classifier over HAR, MNIST and Reuters data with four distributions available in the RandPro package with three different ϵ value.

and it reduces the burden of statisticians who want to perform RP method based feature extraction in R for high dimensional multivariate data analysis.

5. Impact

The RandPro package helps the R users to implement the core functionalities of RP method based feature extraction. As highlighted in Table 1, RandPro offers better functionality over other similar implementation. In this data driven world, there are various application that requires RP method based feature extraction and the next subsection discusses about the utilization of RandPro package by independent experts for their application.

5.1. Independent validation

Dr. Michael W. Dorrity, Senior Fellow, Department of Genome Sciences, University of Washington utilized the RandPro package for dimensionality reduction of transcriptional data from Saccharomyces cerevisiae, with a goal of finding interactions between genes. The package is tested in Mac OS Mojave environment. This work is published in Nature Communications journal and the author mentioned about the RandPro package in Page Number:5, Section: Methods, Subsection: Dimensionality reduction and clustering, Line Number: 18 [29]. In their feedback of the RandPro package, they appreciated the software ran very quickly and the documentation was sufficient to execute their application without any previous experience with the package.

RandPro package is also utilized for nearest neighbor classification of high dimensional low sample size data [30]. Apart from that, Dr. Matthew Crump, Associate Professor, Department of Psychology, Brooklyn College of CUNY also mentioned about the RandPro package and recommended it over PCA for classification purpose in twitter on 21st March 2019.

Table 3
Execution time comparison of functions in RandPro R package with other similar implementations (in seconds).

	Function-1: Find 'k' value using JL lemma	Function-2: Project 'P' with $\epsilon = 0.75$					Function-3: Classify HAR data using k-NN
		P = 200	P = 500	P = 1000	P = 1500	P = 2000	
Python	0.0009	159.77	474.18	1043.34	1693.78	2358.09	N/A
WEKA	N/A	0.31	1.16	2.72	6.66	9.57	N/A
RandPro- R	0.007	31.30	88.78	208.81	355.44	487.56	397.64

N/A — Function not available, HAR — Human activity recognition data.

6. Conclusion

The RandPro package demonstrated in this article provides the practical overview of the random projection method in R language. This package helps the R users to work with the core functionalities of RP method in a straight forward approach. The experiments and results show that the RP method is the good choice for dimension reduction in multivariate big data analysis due to the advantage of distance preservation in the projected space and negligible changes in the accuracy and F-score. Future studies are planned in a direction to provide in-built clustering algorithm and also plans to provide a template for all the three function where expert R users can design user specific functions.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Bellman RE. Dynamic programming. Princeton University Press; 1957.
[2] Adragni KP, Al-Najjar E, Martin S, Popuri SK, Raim AM. Group-wise sufficient dimension reduction with principal fitted components. Comput Stat 2016;31(3):923–41. http://dx.doi.org/10.1007/s00180-015-0611-9.
[3] Chiplunkar R, Huang B. Output relevant slow feature extraction using partial least squares. Chemometr Intell Lab Syst 2019;191:148–57. http://dx.doi.org/10.1016/j.chemolab.2019.07.003, URL http://www.sciencedirect.com/science/article/pii/S0169743919300693.
[4] Smallman L, Underwood W, Artemiou A. Simple Poisson PCA: an algorithm for (sparse) feature extraction with simultaneous dimension determination. Comput Stat 2019. http://dx.doi.org/10.1007/s00180-019-00903-0.
[5] Martínez AM, Kak AC. PCA versus LDA. IEEE Trans Pattern Anal Mach Intell 2001;23(2):228–33. http://dx.doi.org/10.1109/34.908974.
[6] Wold S, Esbensen K, Geladi P. Principal component analysis. Chemometr Intell Lab Syst 1987;2(1):37–52. http://dx.doi.org/10.1016/0169-7439(87)80084-9, URL http://www.sciencedirect.com/science/article/pii/0169743987800849.
[7] Xie H, Li J, Zhang Q, Wang Y. Comparison among dimensionality reduction techniques based on Random Projection for cancer classification. Comput Biol Chem 2016;65:165–72. http://dx.doi.org/10.1016/j.compbiolchem.2016.09.010, URL http://www.sciencedirect.com/science/article/pii/S1476927116304108.
[8] Heinze-Deml C, McWilliams B, Meinshausen N. Preserving privacy between features in distributed estimation. Stat 2018;7(1):e189. http://dx.doi.org/10.1002/sta4.189, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/sta4.189.
[9] Fraiman R, Moreno L, Vallejo S. Some hypothesis tests based on random projection. Comput Stat 2017;32(3):1165–89. http://dx.doi.org/10.1007/s00180-017-0732-4.
[10] Venables WN, Ripley BD. Modern applied statistics with S. Fourth. New York: Springer; 2002, URL http://www.stats.ox.ac.uk/pub/MASS4.
[11] Korobeynikov A, Larsen RM, Laboratory LBN. svd: Interfaces to various state-of-art SVD and eigensolvers. 2016, URL https://CRAN.R-project.org/package=svd. R package version 0.4.
[12] Buitinck L, Louppe G, Blondel M, Pedregosa F, Mueller A, Grisel O, et al. API design for machine learning software: experiences from the scikit-learn project. In: ECML PKDD workshop: languages for data mining and machine learning, 2013, pp. 108–122.
[13] Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH. The WEKA data mining software: an update. SIGKDD Explor 2009;11(1):10–8.
[14] Aghila G, Siddharth R. RandPro: Random projection with classification. 2020, URL https://CRAN.R-project.org/package=RandPro. R package version 0.2.2.
[15] Arriaga RI, Rutter D, Cakmak M, Vempala SS. Visual categorization with random projection. Neural Comput 2015;27(10):2132–47. http://dx.doi.org/10.1162/NECO_a_00769.
[16] Johnson W, Lindenstrauss J. Extensions of Lipschitz mappings into a Hilbert space. In: Conference in modern analysis and probability. Contemporary mathematics, vol. 26, American Mathematical Society; 1984, p. 189–206.
[17] Bingham E, Mannila H. Random projection in dimensionality reduction: Applications to image and text data. In: Proceedings of the seventh ACM SIGKDD international conference on knowledge discovery and data mining. New York, NY, USA: ACM; 2001, p. 245–50. http://dx.doi.org/10.1145/502512.502546.
[18] Dasgupta S. Experiments with random projection. In: Proceedings of the 16th conference on uncertainty in artificial intelligence. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.; 2000, p. 143–51, URL http://dl.acm.org/citation.cfm?id=647234.719759.
[19] Ravindran S, Aghila G. A data-independent reusable projection (DIRP) technique for dimension reduction in big data classification using k-nearest neighbor (k-NN). Natl Acad Sci Lett 2019. http://dx.doi.org/10.1007/s40009-018-0771-6, URL https://doi.org/10.1007/s40009-018-0771-6.
[20] Li G, Gu Y. Distance-preserving property of random projection for subspaces. In: 2017 IEEE international conference on acoustics, speech and signal processing. 2017, p. 3959–63. http://dx.doi.org/10.1109/ICASSP.2017.7952899.
[21] Li W, Mao J, Zhang Y, Cui S. Fast similarity search via optimal sparse lifting. In: Proceedings of the 32nd international conference on neural information processing systems. Red Hook, NY, USA: Curran Associates Inc.; 2018, p. 176–84.
[22] Zaharia M, Xin RS, Wendell P, Das T, Armbrust M, Dave A, et al. Apache spark: A unified engine for big data processing. Commun ACM 2016;59(11):56–65. http://dx.doi.org/10.1145/2934664.
[23] Achlioptas D. Database-friendly random projections: Johnson-lindenstrauss with binary coins. J Comput System Sci 2003;66(4):671–87. http://dx.doi.org/10.1016/S0022-0000(03)00025-4.
[24] Li P, Hastie TJ, Church KW. Very sparse random projections. In: Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining. New York, NY, USA: ACM; 2006, p. 287–96. http://dx.doi.org/10.1145/1150402.1150436, URL http://doi.acm.org/10.1145/1150402.1150436.
[25] Cannings TI, Samworth RJ. RPEnsemble: Random projection ensemble classification. 2016, URL https://CRAN.R-project.org/package=RPEnsemble. R package version 0.3.
[26] Anguita D, Ghio A, Oneto L, Parra X, Reyes-Ortiz JL. A public domain dataset for human activity recognition using smartphones. In: 21th European symposium on artificial neural networks, computational intelligence and machine learning. Bruges, Belgium: 2013.
[27] Xiao H, Rasul K, Vollgraf R. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. 2017, CoRR abs/1708.07747. URL http://arxiv.org/abs/1708.07747.
[28] Pal NR, Bezdek JC. Complexity reduction for "large image" processing. IEEE Trans Syst Man Cybern B 2002;32(5):598–611. http://dx.doi.org/10.1109/TSMCB.2002.1033179.
[29] Dorrity MW, Saunders LM, Queitsch C, Fields S, Trapnell C. Dimensionality reduction by UMAP to visualize physical and genetic interactions. Nature Commun 2020;11(1):1537. http://dx.doi.org/10.1038/s41467-020-15351-4.
[30] Roy S, Dutta S, Ghosh AK, Sarkar S. On the use of a new class of dissimilarity measures for nearest neighbor classification of high dimension, low sample size data. 2019, URL arXiv:arXiv:1902.03295.