**Problem Statement**

There are two parts:

1. Constructing a PCA package with can make an interactive PCA plot with the gene intensity data from different samples or individuals. The package should be able to handle errors properly and should be tested on openCPU.
2. Analyse the given dataset in the light of PCA

**Solutions** (provided in R separately for part A and part B)

**Part A**

Summary: There exist many R packages for principal component analysis (PCA) analysis, however, this package can construct the interactive PCA plot and also it is customized for the specific type of data format. It is available on GitHub and can be installed from there on any system with R version >3.0.0 (most compatible with R version 3.3.2). It saves the plot as a static ‘png’ file format in the working directory and also generates an interactive PCA plot in the ‘html’ file format which directly opens in the web browser. For static plot “factoextra” library was used and for interactive plot “plotly” library was used. It has two functions, one for the PCA analysis and plotting with data scaling and other for the PCA analysis and plotting without data scaling. These options allow for revealing the effect of data scaling on PCA analysis. This option is crucial because PCA analysis is dependent on the variance maximizing phenomenon and variables with very high variance can make the analysis biased and unreliable by dominating the association between the variables in principal components (PCs). Thus, comparison of the results with and without scaling is vital to truly investigate the patterns in the data. The package also can handle many types of errors such as the absence of intensity or metadata file, files not in comma separated (.csv) format, absence of the “symbol” column in the intensity file, and different time units for the samples. The function can also handle the missing values as they may give an error in the PCA analysis. PCA analysis based on singular value decomposition on mean-centered data was performed using the ‘prcomp’ function from “Stats” package. For testing, the package was separately installed and run on a different system using zipped file and GitHub repository. The package was also tested on the openCPU where it worked fine; we could input the data to the package functions using html “GET” method and could retrieve the results using html “POST” method.

Methods:

The two R functions which can perform the PCA analysis along with the interactive plots are constructed. One performs the analysis with data scaling and other performs the data without scaling. In scaling or standardization, each data point is subtracted with mean and divided by the standard deviation that makes the data mean-centered with unit-variance. The r code for the functions is provided in the separate file “functions\_for\_package.R”.

For constructing a new package so that the functions are easily reusable by others, the “devtools” and “roxygen2” libraries were utilized. Once the libraries are installed and loaded in R session following commands were used to create a package:

**require(devtools)**

**require(roxygen2)**

**create("PCA")**

(This will create a new package “PCA” and will make a PCA named folder in the present working directory)

**setwd("./PCA")**

(Navigate to PCA directory. It will have R folder and a few other files. Now put the separate R script for each function into the R folder with proper initializations as required by the functions)

**document()**

(This will create the documentation page for each function based on the information provided in the function’s R script file. Now the package directory will have man directory and also the DESCRIPTION and NAMESPACE files)

**setwd("..")**

**install("PCA")**

(Check if the package is constructed without error and able to install on R. Use R version >3.0.0 and works best on R 3.3.2)

**require(PCA)**

(Load the constructed new R package)

**search()**

(Check if it is loaded properly)

**?plotpca\_withscaling**

**plotpca\_withscaling(genedata="Assignment-gene\_data.csv", metadata="Assignment-Meta\_data sheet.csv")**

**?plotpca\_withoutscaling**

**plotpca\_withoutscaling(genedata="Assignment-gene\_data.csv", metadata="Assignment-Meta\_data sheet.csv")**

(Check for the documentation of the functions from the new R package and also test run the function)

Now this package can be distributed as zipped file or via GitHub. I kept the project as GitHub repository on my GitHub account and also provided the zipped file. The commands to install and run this package on any system with R are mentioned below:

For zipped file

(Download and unzip the package folder and start a new R session. Set the working directory as the folder which contains the unzip package folder "PCA" and the two input files: intensity file and metadata file. Run following commands)

Note: package need ggplot2 version >3.0.0

**setwd("C:/Users/user/Desktop/check/PCA/")**

**if (!require("devtools")) install.packages("devtools")**

**require("devtools")**

**if (!require("roxygen2")) install.packages("roxygen2")**

**require("roxygen2")**

**install("PCA")**

**require(PCA)**

**search()**

**plotpca\_withscaling(genedata="Assignment-gene\_data.csv", metadata="Assignment-Meta\_data sheet.csv")**

**plotpca\_withoutscaling(genedata="Assignment-gene\_data.csv", metadata="Assignment-Meta\_data sheet.csv")**

From GitHub

(Start a new R session. Set the working directory as the folder which contains the two input files: intensity file and metadata file. Run following commands)

**setwd("C:/Users/user/Desktop/check/")**

**if (!require("devtools")) install.packages("devtools")**

**require(devtools)**

**install\_github("shubhamj1510112/PCA")**

**require(PCA)**

**search()**

**plotpca\_withscaling(genedata="Assignment-gene\_data.csv", metadata="Assignment-Meta\_data sheet.csv")**

**plotpca\_withoutscaling(genedata="Assignment-gene\_data.csv", metadata="Assignment-Meta\_data sheet.csv")**

The package was also tested on the openCPU system so that its deployment as an app with openCPU can be done later. Commands for the openCPU testing are the following:

**if (!require("opencpu ")) install.packages("opencpu ")**

**require(opencpu)**

**ocpu\_start\_server()**

(Once the server is started, open the server page paste the “<http://localhost:5656/ocpu>” link in the browser. From here we can test the package locally. Use GET method and navigate to “../library/PCA” in the endpoint and press enter it will show the details of the package. Use GET method and navigate to “../library/PCA/R/plotpca\_withscaling” in the endpoint and press enter it will show the details of the function. Now use POST method and navigate to “../library/PCA/R/plotpca\_withscaling” in the endpoint and input the two parameters ‘genedata’ and ‘metadata’ with their input files and press enter. The function will run for some time and links to different results will appear. Using these links, we can access the results. Same can be done to test the other function “plotpca\_withoutscaling”)

**Part B**

Background:

PCA is a dimensionality reduction and multivariate exploratory data analysis technique that can be used to summarize and visualize the multidimensional data. It drives a set of new variables which are linear combinations of the existing variables such that they are orthogonal and independent to each other. These variables are sorted based on the amount of variance they explain of the total variance in the original data, i.e. PC-1 explains the highest variance, PC-2 next highest and so on. Total number of PCs is always min(n, p) where n is a number of observations and p is a number of variables. There are two methods to achieve the PCA analysis results, one is singular value decomposition on the standardized data, and other is eigenvalue decomposition of centered co-variance data. Overall, PCA is a technique of data transformation which transforms the possibly correlated variables into less number of orthogonal and independent variables called principal components.

Results and Discussion:

The r code for this part is provided separately in the file “pca\_analysis\_on\_actualdata.R”.

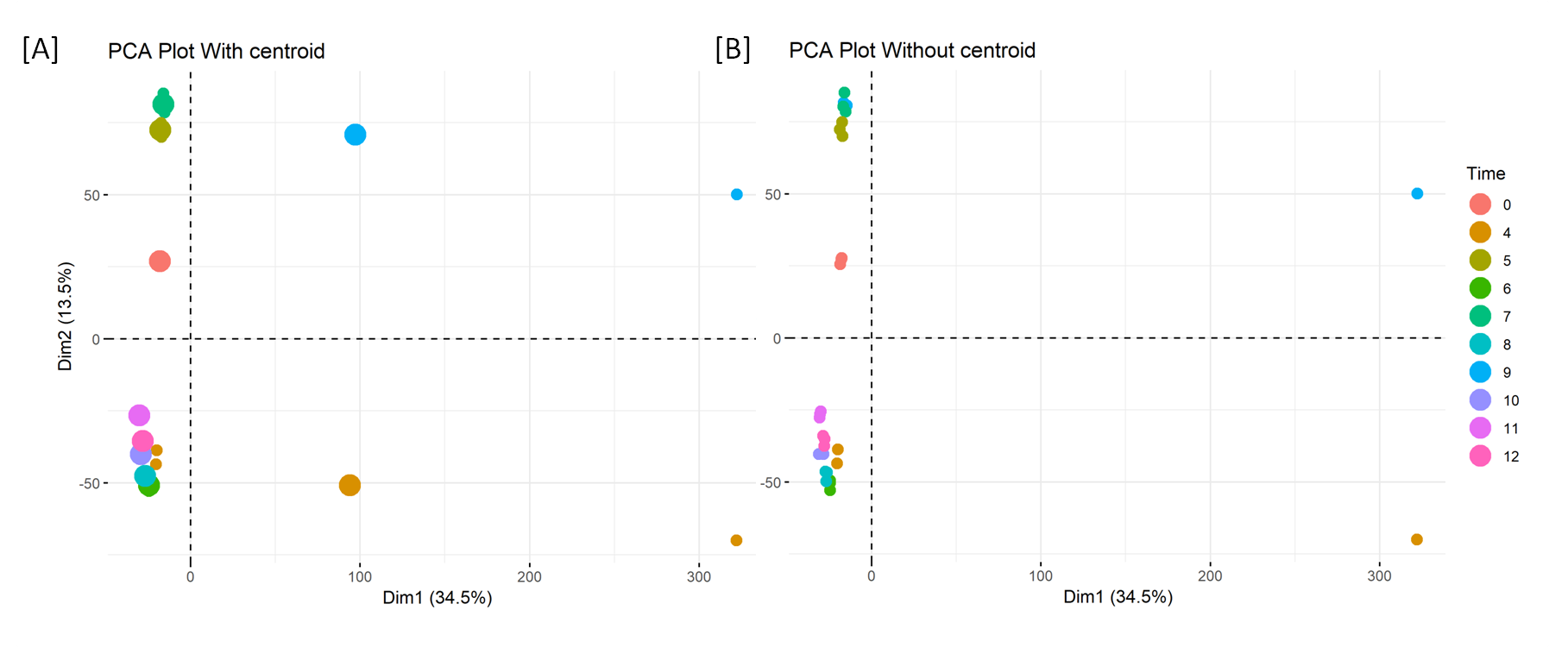
*Data pre-processing*

Data is distributed across two files, intensity file with gene intensity values for each sample, and metadata file with time data for each sample. From the data it is clear that samples/individuals are the observations and gene intensities are the variables. Data was read from the files into data frame and pre-processed. In the intensity data one gene had the missing intensity value for the sample S21. There are two options to deals with it, impute the intensity value or remove the particular observation. Missing values could occur due to many reasons and method to deal with them depends on the cause and quantity of missing values. As the biological details of the samples and data collection method is not provided and also the number of observations with missing values were <1% of total data, I decided to remove the missing value observation and take only the complete cases for further analysis. Outlier treatment is also an essential pre-processing step, however, in PCA centering and scaling of raw data will involved so the effect of outliers will be reduced and also statistical modelling is not performed further so outlier treatment is not required at this stage. The intensity data was transposed so that sample become observations and gene intensities become variables. Further, the intensity and metadata was merged to a single data frame with sample ID as first column, gene intensities as predictor variables, time and unit variables as class variables. All the gene intensity predictor variables were converted to numeric variables.

PCA is a variance maximizing exercise; variables with high variance will dominate the large variance PCs and will make our analysis biased for those variables. So if our variables have equal variances than scaling is not required else scaling or standardization is indispensable. In the gene intensity data the variances of different genes were not equal, thus, I performed PCA analysis on the scaled data. Genes with very less variance in the intensity values across samples will not contribute to PCs and will not be useful for extracting patterns from the data, they are also called as the near zero variance or zero predictor value variables. Therefore, such genes were identified using the “nearZeroVar” function of the “caret” package and were not used for the further analysis. Many genes were repeated in the intensity files and also many may have correlated intensities. Such cases need to be removed because they carry redundant information. Therefore, a correlation analysis to remove any kind of multicollinearity effect was performed. Due to computational intensiveness, correlation analysis was performed on the cluster and not included in this package. The data after this preprocessing step was used for PCA analysis.

*PCA analysis*

I used the covariance matrix based eigenvalue decomposition method to perform the PCA analysis. Many packages are available in R for PCA analysis which gives comparable results; however, the data was of moderate size, so the memory and time constraint was not a problem. Therefore, I utilized the gold standard “prcomp” function from the “Stats” package for PCA analysis. The data points or samples were labelled based on their time point value. To understand the central tendency of data distribution the centroid method was used.

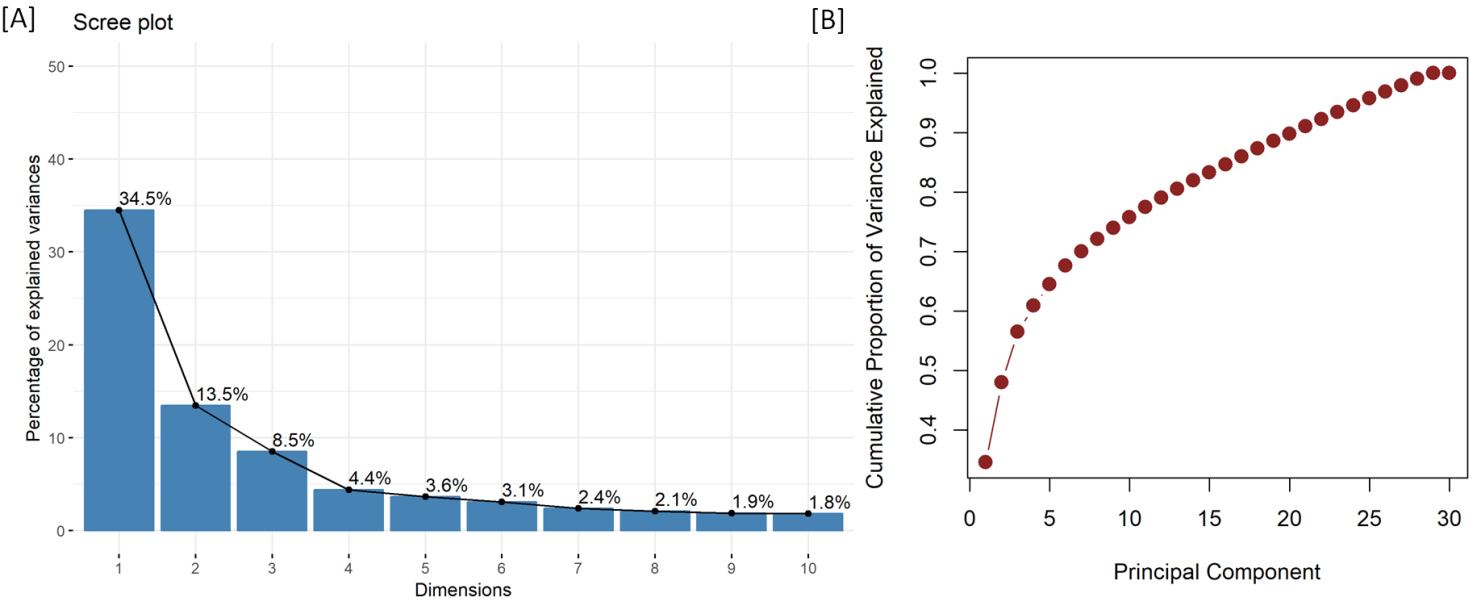


**Figure 1: PCA plot with scaled data A) with centroid B) without centroid**

Each data point is a sample (total 30 points). The data points are colour coded based on the time point value of the particular sample.

From the above plots, it is apparent that samples of time point 0 clusters separately and in the middle of other time points. In general, samples of same time point clusters together. However, the three samples of time point 4 and 9 do not cluster together, maybe the samples were very different from each other, or it indicates some error in data collection or processing. So, if we ignore these time points, we can clearly see that early time points 5 and 7 clusters together and late time points 8,10,11,12 clusters together. Interestingly, the time point 6 clusters with the late time points than with early time points.

Moreover, PC1 and PC2 together explain only 48% (34.5+13.5) of the total variance in the data. Since they explain less than 50% of the variance, thus, we can say that the gene intensities were not very correlated with each other and investigation of PCs is crucial in understanding the complete patterns in the data. Two understand how many PCs are needed to explain sufficient variance of data I made a normal scree plot and a cumulative scree plot.



**Figure 2: PCA scree plot A) Normal scree plot B) Cumulative scree plot**

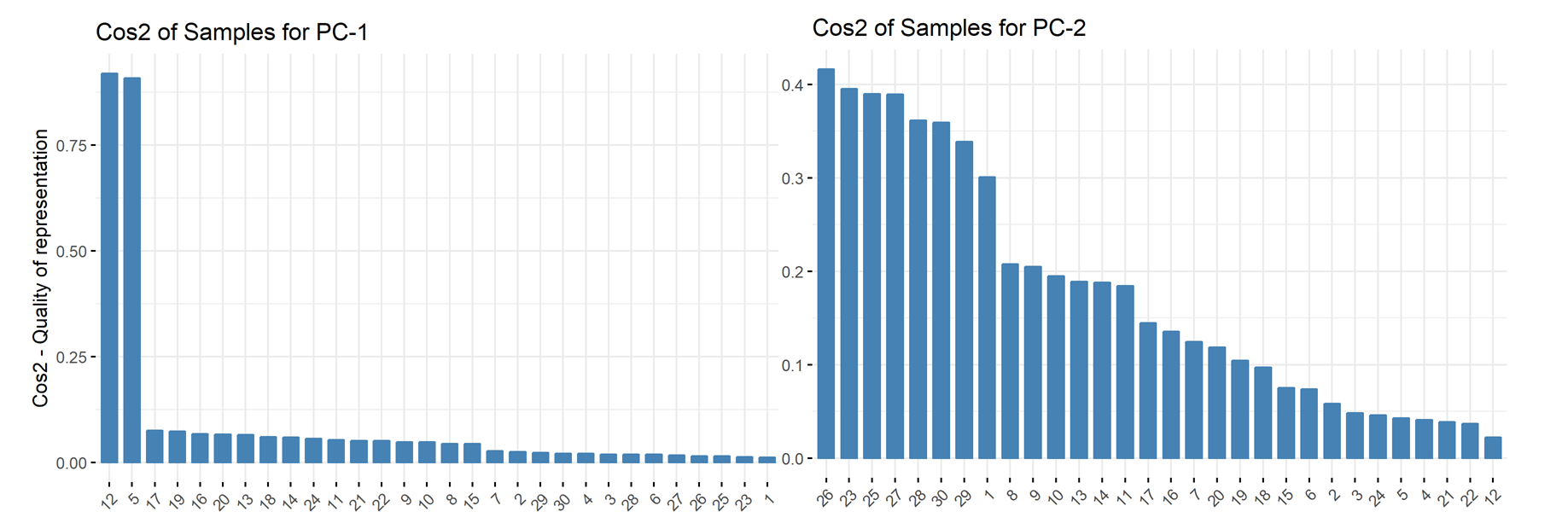
From the above plots, it is apparent that eight PCs are required to explain the ~72% variance in the data, and 21 PCs can explain up to 90% of the variance in the data. This suggests that for any kind of further statistical or machine learning modelling 21 PCs are sufficient; this significantly reduces the predictor variables. The genes with the highest contribution to specific PCs are identified using the “FactoMineR” package.

**Table 1: Top-ranked genes based on their contributions to PC1-4**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Rank | PC1 | PC2 | PC3 | PC4 |
| 1 | Kpna3 | Tbrg4 | Olfr1391 | Dcbld1 |
| 2 | Snora73a | Tram1l1 | Tmem208 | Tbx22 |
| 3 | Tasp1 | Wbp2nl | Kcnk16 | Pcdha5 |
| 4 | Galns | Rps19 | Plk5 | Slc26a3 |
| 5 | Uts2b | Park7 | Olfr952 | Btbd1 |

This reveals that PC1 which explains the highest variance in the data has the most dependence on the kpna3, Snora73a, Tasp1, Galns, and Uts2b genes. We can observe that for each PC the top contributor genes are different which accounts for their orthogonal and independent nature.

In PCA transformation, different samples will have different quality of representation based on the individual sample variance explained by different PCs. This representation quality on factor map is assessed by “cos2” parameter.



**Figure 3: Cos-2 plot of samples for PC-1 and PC-2**

From the Cos-2 plot of PC-1 and PC-2, it is apparent that PC-1 is biased towards sample 12 and 5, whereas PC-2 has better-represented samples 26, 23, 25, 27, 28, 29, and 30. However, none of the PC-1 and PC-2 had cos2 > 0.5 for all the samples which suggest the need of other PCs in explaining the complete variance present in the data. This corroborates with our previous finding that 21 PCs are required to explain up to 90% of the total variance in the data.

**Conclusion**

The package for generating an interactive PCA plot from the genes intensity data for samples/individual is constructed successfully. The detailed PCA analysis of the data reveals that early time points well separate from the late time points, although some time points. 21 PCs can explain up to 90% of the variance in the data which suggest the possible use of these less number of variables (in comparison to >20,000 genes) for further statistical or machine learning based modelling.