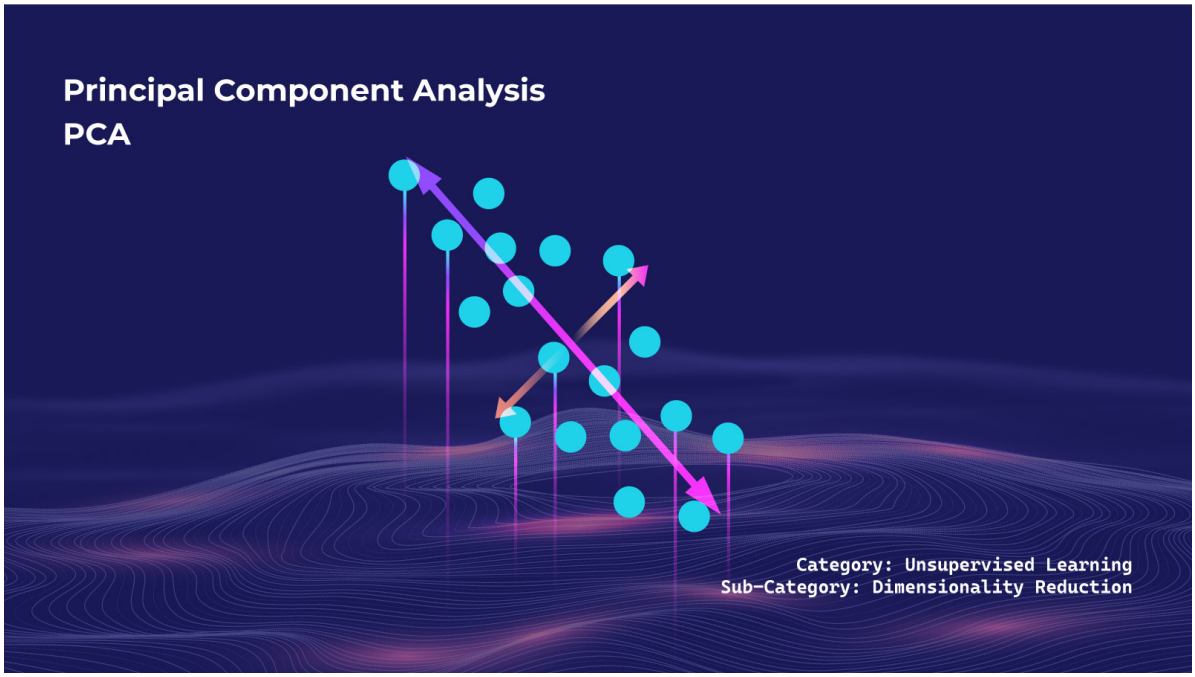
Principal Component Analysis

Hector R. Gavilanes, Chief Information Officer
Gail Han, Chief Operating Officer
Michael T. Mezzano, Chief Technology Officer

2023-11-20

Table of contents

Home



In recent times, the utilization of large datasets has become widespread across numerous fields. To confront the challenges posed by these complex datasets with multiple variables, unsupervised learning algorithms, particularly Principal Component Analysis (PCA), assume a crucial role in various tasks such as dimensionality reduction, feature extraction, and data visualization. The roots of PCA can be traced back to Karl Pearson's conceptualization in 1901, with subsequent formalization by Harold Hotelling in 1933. PCA's primary objective is to reduce data dimensionality while preserving its inherent variability. It achieves this by generating principal components, which serve as representatives for variables, thereby simplifying data representation and enhancing exploratory data analysis. The underlying mathematical framework of PCA revolves around identifying eigenvectors and eigenvalues of the covariance matrix. With the development of software packages, PCA has become easily accessible for data analysis. However, PCA is most effective when data patterns result in statistical variance that can be captured by the principal components, making it a potent tool for data exploration but less suitable for non-linear or non-orthogonal data. Careful consideration must be given to proper data scaling. PCA finds application in diverse fields such as archaeology, neuroscience, and

the arts, enabling practitioners to explore datasets, preprocess data, and streamline analysis. Moreover, it has found utility in computer vision for tasks like facial recognition, showcasing its prowess in reducing dimensionality while preserving crucial information. Overall, PCA is a versatile and widely used technique in statistics and data analysis, providing a valuable means to extract insights from complex datasets and simplify decision-making processes across diverse domains.

1 Introduction

In recent years, the exponential growth of large datasets with numerous variables have become increasingly prevalent across various fields [1], including but not limited to data analysis, image processing, genetics, finance, and signal processing [2]. Analyzing, processing, and visualizing these multivariate datasets pose significant challenges. Therefore, the significance of unsupervised learning algorithms comes into play, particularly in tasks like dimensionality reduction, feature extraction, and visualization of complex data sets. Unsupervised learning is a collection of algorithms to classify raw data [3]. Clustering, and density estimation methods, often serve as a crucial preliminary step in dimensionality reduction where the objective is to find patterns and correlations aiding in organizing the original data. The underlying structures and relationships within the data can inform the selection and application of dimensionality reduction techniques.

Moreover, dimensionality reduction (DR) techniques aim to mitigate the challenge of extracting valuable insights from complex data by reducing the number of dimensions, decreasing computational complexity, eliminating irrelevant and redundant data, improving algorithm accuracy, and facilitating efficient data visualization [4]. Unsupervised learning and dimensionality reduction are indispensable tools in data analysis, and machine learning. While unsupervised learning aims to uncover patterns and correlations between raw data, dimensionality reduction simplifies data representation. One widely used DR technique is Principal Component Analysis (PCA). The objectives of the present research project are the comprehensive exploration of PCA, encompassing an introduction to its core concepts, a discussion of its purposes and functions as well as a demonstration of its application through step-by-step examples.

Principal Component Analysis is best described as a dimension reduction technique for statistical data. PCA has its roots in the work of Karl Pearson in his 1901 paper "On Lines and Planes of Closest Fit to Systems of Points in Space" [5] with the later christening and formal development of the technique by Harold Hotelling in 1933 [6]. Pearson initially conceptualized PCA as a geometric interpretation within statistics; subsequently, PCA gained recognition as a more suitable method than analysis of variance for modeling response data [7]. The aim of PCA is to reduce the dimensionality of a dataset without loss of information about the variability of the data. By creating principal components that act as analogues for variables, the statistical information of the dataset can be preserved and compressed into a more easily representable form. A common example of the application of PCA involves reducing an

n-dimensional data set into 2 principal components which can be plotted on a graph representing the relationships among the original variables; for this reason, PCA is often described as an exploratory data analysis tool.

1.1 Theoretical and mathematical foundations

The fundamental concept behind PCA is to explain the variability in a set of correlated variables with a smaller set of uncorrelated variables, thus mitigating issues such as multicollinearity. The geometric properties of PCs facilitate an intuitive interpretation of key features within complex multivariate datasets [8]; the first principal component represents the direction with the greatest variation in the original data, while subsequent components are uncorrelated with the previous ones. Each component can be interpreted as the direction that maximizes the variance of the original data when projecting new observations onto the components [1]. PCA aims to capture a significant proportion of the original variables' variation in the first few components, offering a practical lower-dimensional summary. While other methods may involve weighted averages across related variables to reduce dimensions, PCA often achieves similar results with minimal loss of variance information [9]. The mathematical foundations of PCA center around data of p variables in n observations, represented by an $n \times p$ matrix X with the goal of finding a linear combination of the columns of X which maximize variance. By maximizing the variance in these linear combinations, we can capture the largest amount of statistical information possible among the dimensions of the dataset. As described by Jolliffe and Cadima [2], this boils down to finding the eigenvectors (a) and the largest eigenvalues (λ) of the covariance matrix S, where Xa_k are the linear combinations called the principal components.

With the development of software packages in computer languages such as R and Python, the computational burden of PCA for large datasets can be easily handled by software; PCA has become easy to use as part of any data processing pipeline. Abdi and Williams describe PCA as "probably the most popular multivariate statistical technique … used by almost all scientific disciplines." [10] Abdi and Williams also emphasize that the goals of PCA should be extracting only the most important information from a dataset to both compress and simplify the dataset and provide a way to analyze the structure of the observations and variables more easily. Importantly, the authors also offer a geometric description of the principal components as orthogonal factors to the original axis of the dataset; another strength of PCA is the fact that the technique can be explained and expressed through several mathematical avenues. Finally, Abdi and Williams offer methods to evaluate the quality of the "PCA model" in reconstructing the original data matrix using the derived principal components. For example, calculating the residual sum of squares, or RESS, after rebuilding X provides a way to identify model accuracy by seeking a minimal RESS value from X and X.

Lever and Altman [11] offer similar praises of PCA while echoing the cautions of prior authors as a powerful data exploration tool with clear limitations. PCA is best when interesting

patterns in data produce statistical variance which can be capture by the principal components, but the technique is far less effective when patterns in data are non-linear or non-orthogonal or when the maximization of variance fails to produce interesting clusters in the principal component space. Scaling is often necessary for PCA to ensure compatibility across variables with different scales and ranges. The original data is typically standardized to have a mean of 0 and a variance of 1 [8]. PCA can be used improperly to produce results that obfuscate the actual statistical content of the dataset; scaling may influence the analysis with prior knowledge of the data, so the decision to scale the data and the scaling methodology should be considered carefully.

1.2 Applications and extensions

1.2.1 In Archeology, Neuroscience, and the Arts

PCA has been utilized by many authors across numerous fields as an essential part of a data analysis pipeline. As an example of the application of PCA, Jolliffe and Cadima [2] present a dataset of 88 observations with 9 variables of measurement for fossilized mammal teeth including length in two dimensions, width in two dimensions, and more. With the R statistical language, finding and displaying the principal components of the dataset becomes trivial work where patterns can be identified graphically in a two-dimensional (PC1 x PC2) plot, versus the original scatterplot which would be displayed in nine-dimensional space. The development of statistical software has made it straightforward for practitioners to use PCA for data exploration and dimensionality reduction. Authors such as Maindonald and Braun [12] have produced publications which present techniques and step-by-step methodologies of using R to calculate principal components and graphically display the results. Their presentation includes code snippets which can be run in R software environments along with worked examples and a discussion of results, facilitating the use of the technique for practitioners of all experience levels.

Felipe Gewers and their research team directly expound on Abdi and Williams' description of PCA as "the most popular multivariate statistical technique [...]" in an analysis of publications which use PCA: Across twenty-three disciplines ranging from Neuroscience to the Arts, PCA is used to explore a dataset before analysis, used as part of a spectrum of statistical analysis tools prior to modeling and analysis, or used to preprocess and simplify data for direct analysis and modeling [13]. The authors also discuss the broad effectiveness of PCA in representing more than 50% of the variance in most datasets using only the first three principal components. They also highlight how differences in captured variance appear between standardization and non-standardization, and in different fields of study; PCA can be effective in an incredibly diverse array of data when applied appropriately. As an exploratory technique with a long tenure in the field of statistics and data analysis, PCA is tried and true in simplifying datasets and allowing practitioners to identify and explore the largest sources of statistical information present in their data.

Principal Component Analysis is also implemented in Scikit Learn with randomized Singular Value Decomposition (SVD) for dimensionality reduction in data analysis, particularly for face recognition and similar high-dimensional data applications. For instance, with an image of 4096 dimensions (64x64 pixel gray scale images) PCA can transform the data into a lower 200 dimension format that still captures the essential information [14]. With randomized SVD it becomes computationally more efficient to approximate the singular vectors, which are then used to perform the transformation. This approach significantly reduces computation time and memory usage. Additionally, PCA decomposes a multivariate dataset into orthogonal components that explain the maximum amount of variance. It can center and optionally scale the input data before applying SVD. Scaling can be useful for downstream models, such as Support Vector Machines with the RBF kernel and K-Means clustering, which assume certain properties of the data distribution.

1.2.2 Computer Vision and Pattern Recognition

The Eigenfaces concept has emerged as a groundbreaking approach to facial detection and recognition. The Eigenfaces algorithm employs PCA to extract essential facial features, reducing the dimensionality of face images while retaining crucial information. The Eigenfaces technique represents facial images in high-dimensional space while projecting the images onto a lower-dimensional space. Turk and Pentland demonstrated the effectiveness of Eigenfaces in recognizing faces under various conditions, including variations in lighting, facial expressions, and pose [15]. The potential implications of this research extend beyond face recognition, and have broader implications for image analysis, computer vision, and biometric systems. The paper Eigenfaces by Zhang and Turk provided an insightful exploration of the Eigenfaces method [16], considered as the first working technique in facial recognition. Eigenfaces leverage the power of PCA to represent facial images compactly and efficiently, making it possible to recognize faces in various contexts.

The idea of principal components to represent human faces was developed by Sirovich and Kirby in 1987 and used by Turk and Pentland in 1991 [15] for face detection and recognition. The authors describe the mathematical principles behind PCA, and its adaptation for facial recognition, capturing the most prominent facial features while reducing the dimensionality of the data. Specifically, the eigenfaces are the principal components of a distribution of faces, or the eigenvectors of the covariance matrix of the set of face images, where an image with N pixels is considered a point (or vector) in N-dimensional space. Emphasis is placed on the versatility of Eigenfaces in handling variations in lighting, pose, and facial expressions, making it a robust tool for real-world applications.

This application of PCA represents one of many for a longstanding technique in the rapidly growing field of statistics and data analysis. The unsupervised learning algorithm plays a vital role in dimensionality reduction, feature extraction, and visualization of complex data sets. Understanding PCA is crucial for researchers, and students seeking to extract meaningful insights and patterns from high-dimensional data to simplify the decision-making process.

2 Methods

The aim of Principal component analysis (PCA) is to reduce the dimensionality of multivariate data while preserving the variability present in the data. The principal components derived from the dataset are orthogonal variables represented by linear combinations of the original variables which maximize variance. The first principal component (PC) captures the most variance, followed by the second orthogonal principal component, and so on. There can be as many PCs as there are variables in the original data, but the technique is typically used to simplify high-dimension data for improved interpretability. Principal components can be calculated using eigenvalue decomposition or the singular value decomposition (SVD) of the data matrix, so data must be preproccesed and several assumptions met for PCA to yield meaningful results.

2.1 Assumptions

For PCA to be effective, the data should be continuous (although adaptations of PCA exist for other numeric data structures) and normally distributed, although the the distribution of the data does not truly matter when utilizing PCA as an exploratory methodology. More importantly, the data should be linearly related or the linear combinations of the principal components cannot meaningfully capture the variance of the data. Ideally, the variables should be similar in scale, and free from extreme outliers, or missing values, although this can be addressed in preprocessing, and implementations of PCA such as robust PCA have been developed to address these challenges. [2]

2.2 Preprocessing

Preprocessing data for PCA is straightforward. Missing data should be handled using a method appropriate for the dataset, such as imputation based on the mean or median of the variable observations. After this the variables should be centered and scaled, to a mean of 0 and a standard deviation of 1, although statistical software libraries for SVD and PCA may include this as an option within the function. [17]

2.3 Eigenvectors

PCA uses eigenvectors and their corresponding eigenvalues to calculate the principal components; a brief overview is given here. Eigen is a German word meaning *inherent* or *characteristic*, and an eigenvector can be described geometrically as a nonzero vector a of a linear transformation matrix M which does not change direction when the transformation is applied; the only change that occurs is a scaling by factor λ, the eigenvalue of the eigenvector a. Such a characteristic vector is useful in PCA, where the goal is to maximize variance while reducing dimensionality, and in this context the eigenvectors and eigenvalues can be thought of as the inherent components of the dataset which contain the most important information. Eigenvalues can be calculated from the characteristic polynomial of the matrix, by taking the determinant of $M - \lambda I$, where I is the identity matrix. Setting this expression equal to zero allows the calculation of the eigenvalues as the roots of the characteristic polynomial; the resulting equation is called the characteristic equation:

$$det(M - \lambda I) = 0 \tag{2.1}$$

An eigenvalue λ_k can be used to solve for some eigenvector a_k with the equation $(M-\lambda I)a = 0$. With PCA, we can use the eigenvectors of the covariance matrix to compute the PCs. [18]

2.4 Principal Component Analysis

In this approach to PCA, SVD is used to extract the most information (variance) from the data matrix while reducing the dimensionality of the data. The first principal component will have the largest possible variance (also called inertia), whose value is defined as a factor score. Factor scores represent a geometric projection of the observations onto the PCs. The second PC, orthogonal to the first, has the second largest variance, and the third PC would continue this pattern. The calculation of PCs via SVD can be understood with the use of matrix operations on a dataset. [19]

2.4.1 Centering and Scaling

Let our dataset be represented by the $N \times P$ matrix X comprised of N observations of P variables in the data set, where any element x_{np} represents the nth observation of variable p in the dataset. The matrix X has rank A where $A \leq min\{N, P\}$. The data in X is centered and scaled, such that the mean of each column X_p is 0 and every x_{np} has been standardized with scaled unit variance. We can represent this with the formula:

$$z_{np} = \frac{x_{np} - \bar{x}_p}{\sigma_p} \tag{2.2}$$

2.4.2 Eigendecomposition of the Covariance Matrix

The aim of PCA is to find some linear combination of the columns of X which maximizes the variance. If we define a as a vector of constants $a_1, a_2, a_3, ..., a_p$, then Xa represents the linear combination of interest. The variance of Xa is represented by $var(Xa) = a^T Sa$, with the covariance matrix S, and T representing the transpose operator. Finding the Xa with maximum variance equates to finding the vector a which maximizes the quadratic $a^T Sa$, where $a^T a = 1$. We can write this as $a^T Sa - \lambda(a^T a - 1)$, with the Lagrange multiplier λ. [20] Equating this expression to the null vector 0 allows us to differentiate with respect to a:

$$Sa - \lambda a = 0 \Rightarrow Sa = \lambda a \tag{2.3}$$

Therefore, a is a unit-norm eigenvector with eigenvalue λ of the covariance matrix S. The largest eigenvalue of S is λ_1 with the eigenvector a_1, which we can define for any eigenvector a:

$$var(Xa) = a^T Sa = \lambda a^T a = \lambda \tag{2.4}$$

Any $p \times p$ real symmetric matrix has exactly p real eigenvalues λ_k for $k = 1, ..., p$. The corresponding eigenvectors of these eigenvalues can be defined to form an orthonormal set of vectors such that $a_k^T a_{k^T} = 1$ if $k = k^T$ and zero otherwise. If we consider that S is such a matrix and impose the restriction of orthogonality to the different coefficient vectors of S, the full set of eigenvectors of S represent the solutions to finding linear combinations Xa_k which maximize variance while minimizing correlation with prior linear combinations. Xa_k then represent the linear combinations which are the principal components of the dataset with eigenvectors a_k and eigenvalues λ_k. The elements of Xa_k are the factor scores of the PCs, while the elements of the eigenvectors a_k represent the loadings of the PCs. [2]

2.4.3 Singular Value Decomposition

Next we define the singular value decomposition of X. Let L be the $N \times A$ matrix of left singular vectors of the matrix; that is, the columns of L are made up of the eigenvectors of XX^T. Let R be the $P \times A$ matrix of right singular vectors; the columns of R are made up of the eigenvectors of $X^T X$. Finally, let D be the diagonal matrix of singular values, meaning the singular values in D are the square roots of the eigenvalues of XX^T and $X^T X$, and D^2 is defined as the diagonal matrix of the non-zero eigenvalues. We can define the singular value decomposition of matrix X as:

$$X = LDR^T \tag{2.5}$$

In this context, the eigenvalues represent the variances of the principal components and summarily contain the important information for the dataset, and we can obtain the PCs of X from the SVD. [10] With the identity matrix I, the $I \times R$ matrix of factor scores can be expressed as:

$$F = LD \tag{2.6}$$

These factor scores are calculated from the coefficients of the linear combinations in matrix R, which can be defined as a projection matrix of the original observations onto the PCs, i.e. the product of X and R:

$$F = LD = LDR^T R = XR \tag{2.7}$$

The matrix R is also referred to as a loading matrix, and X is often described as the product of the factor score matrix and the loading matrix:

$$X = FR^T \tag{2.8}$$

with the decomposition of $F^T F = D^2$ and $R^T R = I$.

The loadings represent the weights of the original variables in the computation of the PCs; in other words, the correlation from -1 to 1 of each variable with the factor score.

In a geometric interpretation of PCA, the factor scores measure length on the Cartesian plane. This length represents the projection of the original observations onto the PCs from the origin at $(0, 0)$. This is especially useful as a visualization of higher dimension data in two dimensions by utilizing the first two PCs which capture the most variance in the original data. [11]

2.5 Interpretation of the Principal Components

There are several ways to interpret the PCs derived from the analysis. Since the eigenvalues represent the variance of the PCs, the proportion of the eigenvalues explain the proportion of variation in the dataset. Using a scree plot, these eigenvalues are plotted to show how much variation each PC explains. Another commonly used tool is a biplot, a combination of the plots of the factor scores (points) and the loadings (vectors) for two PCs (typically PC1 and PC2). The biplot is meant to visually capture the relationship between the original variables and the principal components. Clusters of points represent highly correlated variables, and vector lengths represent the variability captured in that direction on the principal component axis. While many methods and tools exist to interpret the results of PCA, the usefulness of each depends on the needs of the analysis. [18]

3 Examples

Moving beyond the theoretical foundations of principal components, how is PCA applied to data? We offer two examples; the first a demonstration of the manual calculation of principal components, and the second implementing PCA on a large dataset using R.

3.1 Manual calculation of principal components

In this illustration, we have access to the two grades of four students in a statistics subject. We aim to employ principal component analysis as a means to reduce the dimensionality from two variables to a singular variable. This transformation will effectively represent students' performance in the subject with a more compact and interpretable measure. This example is adapted from the resource *How to compute principal components* [21].

Scores	**Basic Stats**	**Advanced Stats**
Student 1	**4**	**11**
Student 2	**8**	**4**
Student 3	**13**	**5**
Student 4	**7**	**14**
Mean	$\bar{x} = \mathbf{8}$	$\bar{y} = \mathbf{8.5}$

3.1.1 Calculate the covariance matrix M

$$\begin{bmatrix} \text{cov}(x,x) & \text{cov}(x,y) \\ \text{cov}(y,x) & \text{cov}(y,y) \end{bmatrix}$$

$\Rightarrow \text{cov}(x,x) = \text{var}(x) = \mathbf{E}(x^2)$ - $\mathbf{E}\ (x)^2 =$

$$\frac{(16+0+25+1)}{3} = 14$$

$\Rightarrow \text{cov}(y,y) = \text{var}(y) = \mathbf{E}(y^2)$ - $\mathbf{E}\ (y)^2 =$

$$\frac{(6.25+20.25+12.25+30.25)}{3} = 23$$

$\Rightarrow \text{cov}(x, y) = \text{cov}(y, x) = \mathbf{E}(xy)$ - $\mathbf{E}\ (x)\mathbf{E}\ (y) =$

$$\frac{(-10 + 0 - 17.5 - 5.5)}{3} = -11$$

$\Rightarrow$ Covariance Matrix M

$$\begin{bmatrix} 14 & -11 \\ -11 & 23 \end{bmatrix}$$

3.1.2 Compute the singular value decomposition (SVD)

We can obtain the principal components and loadings from SVD of the covariance matrix M since covariance matrix M is a square matrix:

$$\begin{bmatrix} 14 & -11 \\ -11 & 23 \end{bmatrix} \times \textit{Any vector} = \lambda \times \textit{Any vector}\ ,\ (\textit{vector} \neq 0)$$

$$det(M - \lambda I) = 0$$

3.1.2.1 Obtain eigenvalues of the covariance matrix $\rightarrow \lambda_1$ & λ_2

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\lambda I = \lambda \times \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}$$

$$\begin{bmatrix} 14 & -11 \\ -11 & 23 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} = \begin{bmatrix} 14 - \lambda & -11 \\ -11 & 23 - \lambda \end{bmatrix}$$

$\Rightarrow$

$$det \begin{bmatrix} 14 - \lambda & -11 \\ -11 & 23 - \lambda \end{bmatrix} = 0$$

$\Rightarrow$

$$(14 - \lambda)(23 - \lambda) - (-11)(-11) = 0$$

$$\lambda^2 + 37\lambda - 201 = 0$$

$\Rightarrow \lambda_1 = 30.3849$, $\lambda_2 = 6.6152$ (eigenvalues for Covariance Matrix M)

3.1.2.2 Obtain eigenvector of λ_1

$$(M - \lambda_1 I) \times U_1 = \mathbf{0}$$

$\Rightarrow$

$$\begin{bmatrix} 14-\lambda & -11 \\ -11 & 23-\lambda \end{bmatrix} \times \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \mathbf{0}$$

$$(14-\lambda)u_1 - 11u_2 = 0$$

$\Rightarrow$

$$-16.3849u_1 - 11u_2 = 0$$

$\Rightarrow$

$$-16.3849u_1 = 11u_2$$

$\Rightarrow$

$$u_1 = \frac{11}{-16.3849}u_2$$

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = u_2 \begin{bmatrix} \frac{11}{-16.3849} \\ 1 \end{bmatrix}$$

$\Rightarrow$

$$u_2 \begin{bmatrix} -11 \\ 16.3849 \end{bmatrix}$$

$$\begin{bmatrix} 16.3849 & 11 \end{bmatrix} \times \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} 11 \\ -16.3849 \end{bmatrix}$$

Normalized Eigenvector

$\Rightarrow \lambda_1$: e_1

$$\frac{1}{\sqrt{11^2 + 16.3849^2}} \begin{bmatrix} 11 \\ -16.3849 \end{bmatrix} = \begin{bmatrix} 0.5574 \\ -0.8303 \end{bmatrix}$$

$\Rightarrow \lambda_2$ e_2 (Right singular vector) =

$$\begin{bmatrix} 0.8303 \\ 0.5574 \end{bmatrix}$$

Table 3.3: Dataframe structure

Student	BasicStats	AdvancedStats
1	4	11
2	8	4
3	13	5
4	7	14

Table 3.4: Importance of components

	PC1	PC2
Standard deviation	1.270042	0.6220884
Proportion of Variance	0.806500	0.1935000
Cumulative Proportion	0.806500	1.0000000

3.1.3 Derive The new dataset

First Principal Component (PC1)

$$P_{11} = e_1^T \times \begin{bmatrix} 4 - mean(x) \\ 11 - mean(y) \end{bmatrix} = \begin{bmatrix} 0.5574 & -0.8303 \end{bmatrix} \begin{bmatrix} 4 - 8 \\ 11 - 8.5 \end{bmatrix} = -4.3052$$

$$P_{12} = e_1^T \times \begin{bmatrix} 8 - mean(x) \\ 4 - mean(y) \end{bmatrix} = \begin{bmatrix} 0.5574 & -0.8303 \end{bmatrix} \begin{bmatrix} 8 - 8 \\ 4 - 8.5 \end{bmatrix} = 3.7361$$

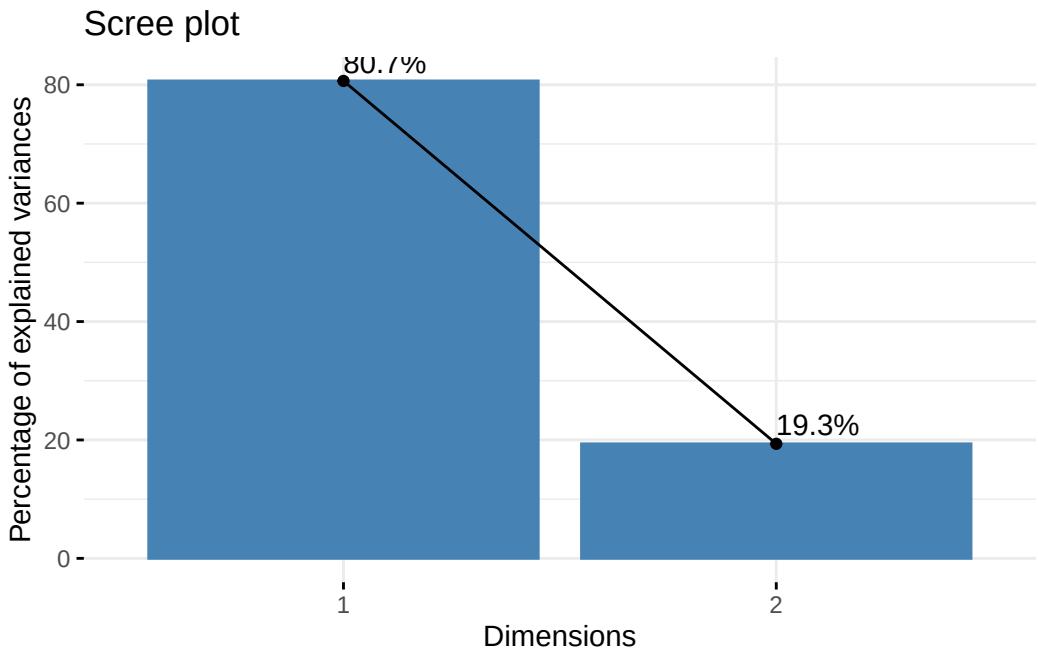
$$P_{13} = e_1^T \times \begin{bmatrix} 13 - mean(x) \\ 5 - mean(y) \end{bmatrix} = \begin{bmatrix} 0.5574 & -0.8303 \end{bmatrix} \begin{bmatrix} 13 - 8 \\ 5 - 8.5 \end{bmatrix} = 5.6928$$

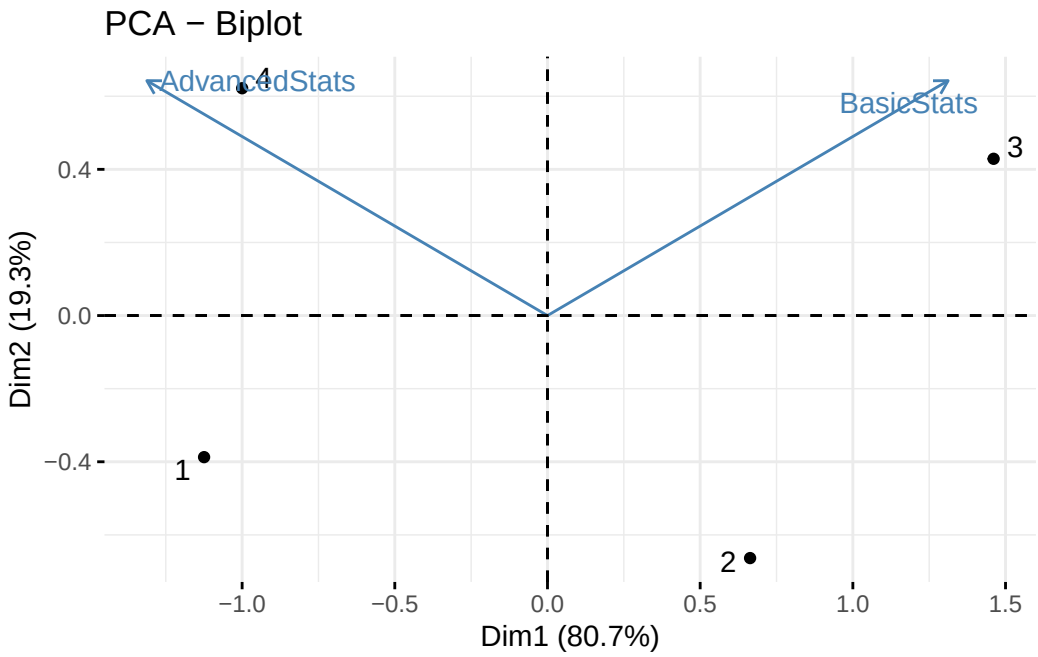
$$P_{14} = e_1^T \times \begin{bmatrix} 7 - mean(x) \\ 14 - mean(y) \end{bmatrix} = \begin{bmatrix} 0.5574 & -0.8303 \end{bmatrix} \begin{bmatrix} 7 - 8 \\ 14 - 8.5 \end{bmatrix} = -5.1238$$

The new dataset (Left singular vector)

	Student 1	**Student 2**	**Student 3**	**Student 4**
PC1	**-4.3052**	**3.7361**	**5.6928**	**-5.1238**

3.1.4 Verifying and visualizing the results





3.1.5 Results

The first principal component of the data captures 80.7% of the variation (or information) while reducing the dimensionality of the dataset from 2 variables to 1. Small datasets such as this make the hand-calculation of principal components feasible and easy to follow, but the strengths of PCA are especially evident when software is used to enable principal component analysis for large datasets. In the next example, we demonstrate how PCA can be used with a large dataset using the R programming language.

3.2 PCA on a large dataset using R

For this application of PCA, the Abalone dataset from the UCI Machine Learning Repository is used [22]. This dataset contain 4177 observations of 9 variables which record characteristics of each abalone including sex, length, diameter, height, weights, and the number of rings. The variables, apart from sex, are continuous and correlated making the dataset an ideal candidate for demonstrating dimensionality reduction via PCA.

3.2.1 Libraries

First, the appropriate and necessary libraries are loaded in R. These provide the functions which serve as the backbone of the analysis, handling the computational aspects of PCA as well as visualizing the results.

```
# Load necessary libraries
library(tidyverse)  # for handling missing values
library(corrplot) # for plotting the correlation matrix
library(factoextra) # PCA plots
library(summarytools) #produces summary stats
```

3.2.2 Data Preparation

The dataset contains 9 variables with 1 categorical variable and 8 numeric variables. The dataset contains no missing values. For this example in applying principal component analysis, we exclude the categorical variable 'Sex' and focus the PCA on the numerical dimensions of the Abalone. For analyses involving a mix of numeric and non-numeric variables other factor analysis techniques can be used, such as factor analysis of mixed data [23].

```
# Load dataset
abalone <- read.csv('./abalone/abalone.csv')
```

	Diameter	Height	Length	Rings	Shell_weight	Shucked_weigh
Mean	0.4078813	0.1395164	0.5239921	9.9336845	0.2388309	0.3593675
Std.Dev	0.0992399	0.0418271	0.1200929	3.2241690	0.1392027	0.2219629
Min	0.0550000	0.0000000	0.0750000	1.0000000	0.0015000	0.0010000
Q1	0.3500000	0.1150000	0.4500000	8.0000000	0.1300000	0.1860000
Median	0.4250000	0.1400000	0.5450000	9.0000000	0.2340000	0.3360000
Q3	0.4800000	0.1650000	0.6150000	11.0000000	0.3290000	0.5020000
Max	0.6500000	1.1300000	0.8150000	29.0000000	1.0050000	1.4880000
MAD	0.0963690	0.0370650	0.1186080	2.9652000	0.1475187	0.2349921
IQR	0.1300000	0.0500000	0.1650000	3.0000000	0.1990000	0.3160000
CV	0.2433058	0.2998003	0.2291884	0.3245693	0.5828504	0.6176489
Skewness	-0.6087607	3.1265706	-0.6394138	1.1133019	0.6204809	0.7185815
SE.Skewness	0.0378868	0.0378868	0.0378868	0.0378868	0.0378868	0.0378868
Kurtosis	-0.0482711	75.8953091	0.0616411	2.3239123	0.5281636	0.5912553
N.Valid	4177.0000000	4177.0000000	4177.0000000	4177.0000000	4177.0000000	4177.0000000
Pct.Valid	100.0000000	100.0000000	100.0000000	100.0000000	100.0000000	100.0000000

```
data_desc = descr(abalone, plain.ascii = FALSE, headings = FALSE) # descriptive statistics

data_desc %>%
  kbl(align= 'l') %>%
  kable_paper("hover")
```

The summary statistics show the differences in measurement between variables, with some variables such as diameter and viscera weight having small ranges and others, namely rings, having relatively large ranges. For this reason, scaling of the variables is a crucial step in PCA to ensure results accurately capture the variance in the data.

3.2.3 Feature Scaling

Standardization ensures all variables, also called features, are on the same scale, and the scale function allows us to center the data to a mean of 0 and variance of 1. This ensures no single feature has an outsized effect during the principal component analysis.

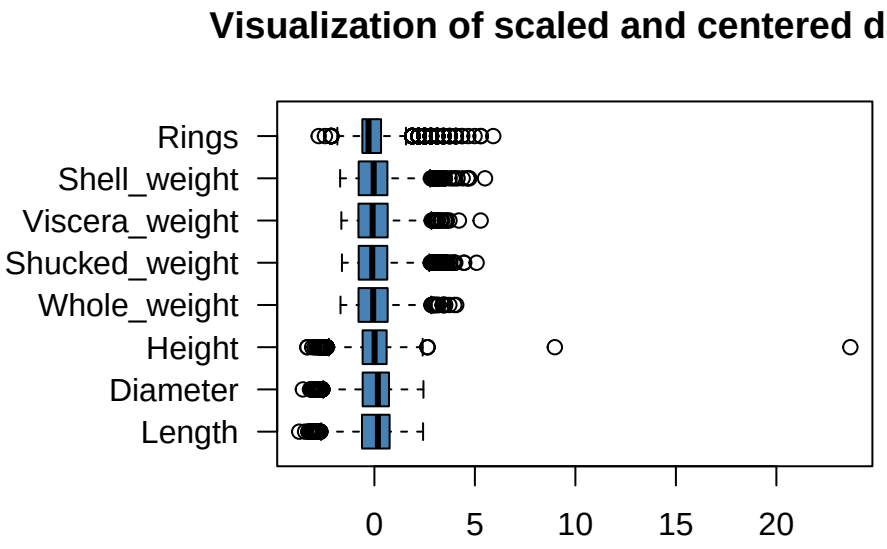
```
# Select only the numeric variables
abalone = select_if(abalone, is.numeric)

# Standardization of numerical features
abalone_sc <- scale(abalone, center = TRUE, scale = TRUE)
```

	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_w
	Min. :-3.7387	Min. :-3.5558	Min. :-3.33555	Min. :-1.68589	Min. :-1.6145	Min. :-1.64
	1st Qu.:-0.6161	1st Qu.:-0.5832	1st Qu.:-0.58614	1st Qu.:-0.78966	1st Qu.:-0.7811	1st Qu.:-0.7
	Median : 0.1749	Median : 0.1725	Median : 0.01156	Median :-0.05963	Median :-0.1053	Median :-0
	Mean : 0.0000	Mean : 0.0000	Mean : 0.00000	Mean : 0.00000	Mean : 0.0000	Mean : 0.0
	3rd Qu.: 0.7578	3rd Qu.: 0.7267	3rd Qu.: 0.60926	3rd Qu.: 0.66123	3rd Qu.: 0.6426	3rd Qu.: 0.
	Max. : 2.4232	Max. : 2.4397	Max. :23.68045	Max. : 4.07178	Max. : 5.0848	Max. : 5.2

```
sc_sum = summary(abalone_sc)

kbl(sc_sum, align= 'l') %>%
  kable_paper("hover")
```

Viewing the data after scaling and centering, values greater than 3 or less than -3 represent outliers more than 3 standard deviations from the mean. Based on the ranges of the variables, we should view a boxplot of the data to further investigate.

```
# Plot a boxplot to visualize potential outliers
par(mar=c(4, 8, 4, 4))
boxplot(abalone_sc, col = "steelblue", main = "Visualization of scaled and centered data",
```



	Outliers
Length	15
Diameter	13
Height	5
Whole_weight	19
Shucked_weight	37
Viscera_weight	22
Shell_weight	27
Rings	62

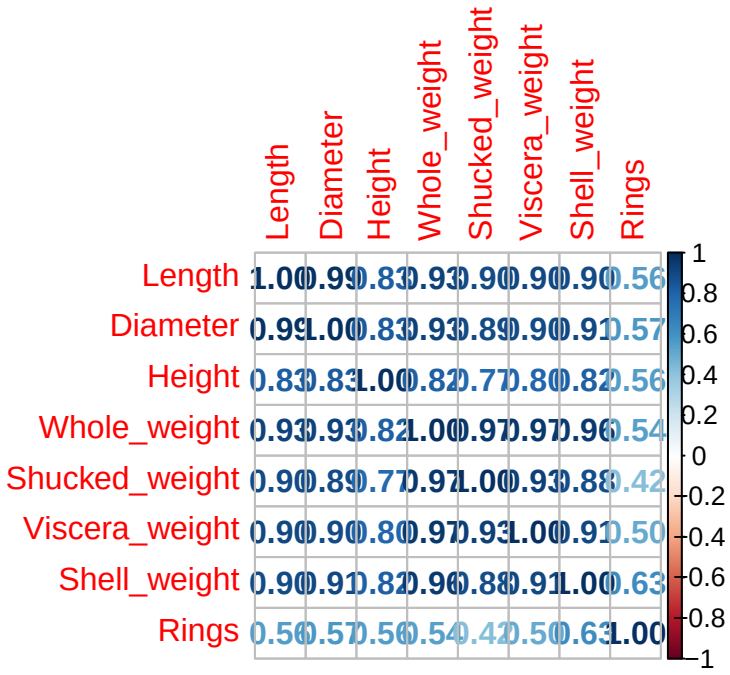
Are there enough outliers to be a cause for concern? We can see how many lie outside of the third standard deviation of the data for each variable.

```
outs = colSums(abalone_sc > 3 | abalone_sc < -3)

kbl(outs, col.names = ('Outliers'), align = 'l') %>%
  kable_paper("hover")
```

Of the 4177 observations, at most 62 in a single variable (Rings) are outliers. The tolerance for outliers will differ depending on the investigation, but for our illustrative purposes this number is well within tolerance for principal component analysis.

Lastly, we can investigate the correlation among the variables. PCA is best used with linearly correlated data. If the data is not correlated, the results of PCA will be less meaningful.

```
# Calculate correlations and round to 2 digits
abalone_corr <- cor(abalone_sc)
corrplot(abalone_corr, method="number")
```



Our scaled and centered data has strong linear correlations and contains a relatively small number of outliers. We can now calculate the principal components of the dataset.

3.2.4 PCA via Singular Value Decomposition

The prcomp() function [17] performs principal component analysis on a dataset using the singular value decomposition method, which utilizes the covariance matrix of the data.

```
# Apply PCA using prcomp()
abalone_pca <- prcomp(abalone_sc)

sum_pca = as.data.frame(summary(abalone_pca)$importance)

kbl(sum_pca, align= 'l', caption = "Importance of components") %>%
  kable_paper("hover")
```

```
# Principal Component scores vector
pc_scores <- abalone_pca$x

# Std Deviation of Components
component_sdev <- abalone_pca$sdev
```

Table 3.5: Importance of components

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	P
Standard deviation	2.590838	0.8340342	0.508373	0.4074185	0.2914613	0.251938	0.1126684	0
Proportion of Variance	0.839050	0.0869500	0.032310	0.0207500	0.0106200	0.007930	0.0015900	0
Cumulative Proportion	0.839050	0.9260100	0.958310	0.9790600	0.9896800	0.997610	0.9992000	1

```
# Eigenvector or Loadings
eigenvector <- abalone_pca$rotation

# Mean of variables
component_mean <- abalone_pca$center

# Scaling factor of Variables
component_scale <- abalone_pca$scale

# Proportion of variance explained by each PC
variance_explained <- component_sdev^2 / sum(component_sdev^2)

# Cumulative proportion of variance explained
cumulative_variance_explained <- cumsum(variance_explained)

# Retain components that explain a percentage of the variance
num_components <- which(cumulative_variance_explained >= 0.92)[1]

# Select the desired number of principal components
selected_pcs <- pc_scores[, 1:num_components]
```

The first 2 principal components alone explain 92% of the variance in the data.

3.2.4.1 Loading of First Two Components

The loading are the weights assigned to each variable for that particular principal component.

```
# Access the loadings for the first two principal components
loadings_first_two_components <- eigenvector[, 1:2]

# Print the loadings for the first two principal components

kbl(loadings_first_two_components, align= 'l', caption = "Loadings for the first two princ
```

Table 3.6: Loadings for the first two principal components

	PC1	PC2
Length	0.3721385	0.0682827
Diameter	0.3730941	0.0400480
Height	0.3400268	-0.0704631
Whole_weight	0.3783075	0.1373462
Shucked_weight	0.3624545	0.2988399
Viscera_weight	0.3685578	0.1729785
Shell_weight	0.3707578	-0.0454004
Rings	0.2427128	-0.9212039

```
kable_paper("hover")
```

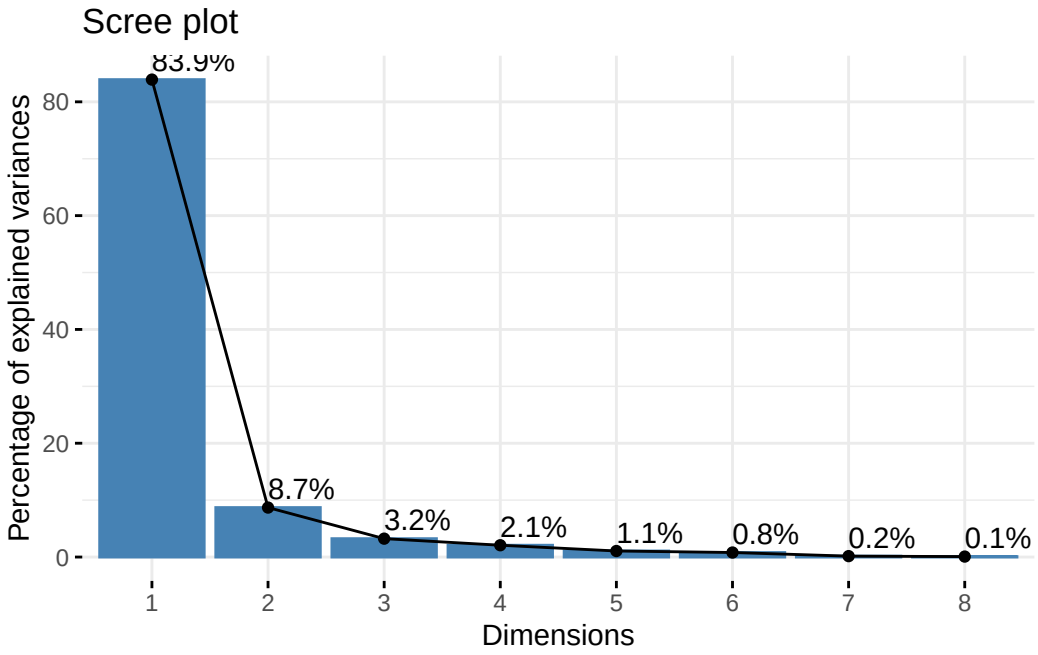
3.2.4.2 PCA - Elements

The values in **abalone_pca$x** are the coordinates of each observation in the new principal component space. These coordinates are the scores for each observation along each principal component. The eigenvectors of the covariance or correlation matrix of the data represent the directions of maximum variance in the dataset.

3.2.5 Visualization

3.2.5.1 Scree Plot - Cumulative Variance Explained

```
fviz_eig(abalone_pca, addlabels = TRUE)
```

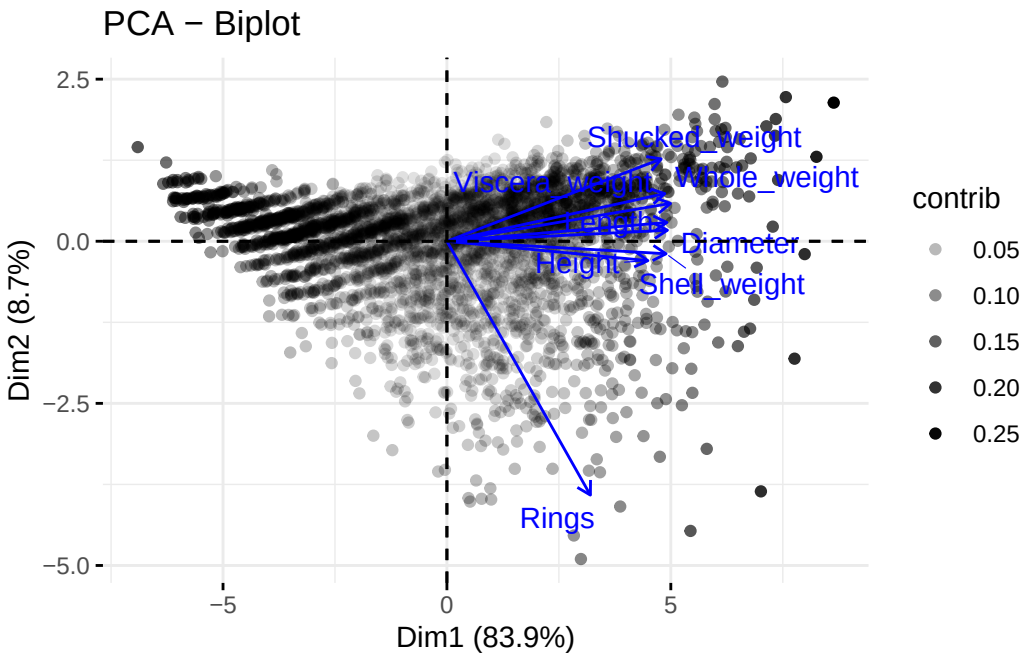


The scree plot visualizes the variance captured by each PC. PC1 explains 83.9% of the variance, and PC2 explains 8.7% variance.

3.2.5.2 Biplot

The correlation between a variable and a principal component is used as the coordinates of the variable on the PC, shown as dimensions on the biplot. Dim1 corresponds to PC1, and Dim2 to PC2. The representation of variables differs from the plot of the observations: The observations are represented by their projections, but the variables are represented by their correlations [10].

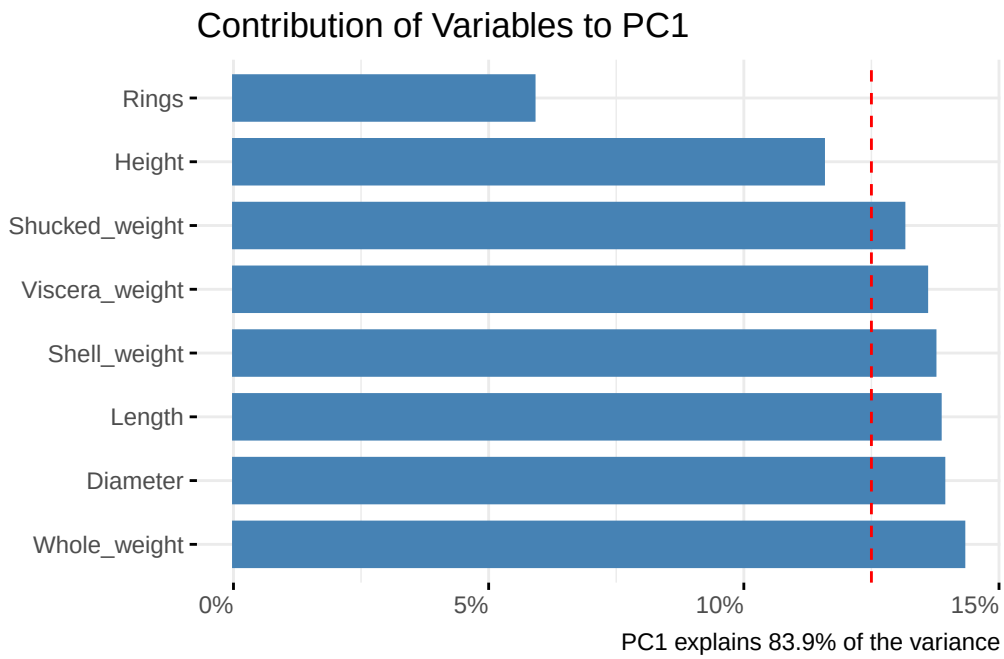
```
fviz_pca_biplot(abalone_pca, label = "var", alpha.ind = "contrib", col.var = "blue", repel
```



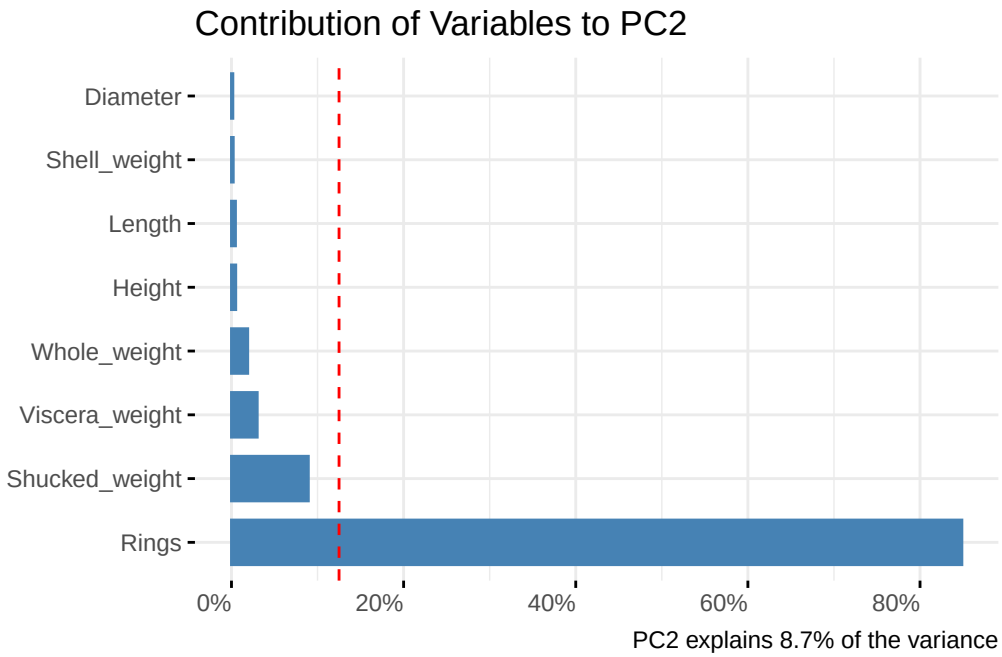
3.2.5.3 Variable Contribution

Top variable contribution for the first two principal components.

```
# Contributions of variables to PC1
pc2_contribution <- fviz_contrib(abalone_pca, choice = "var", axes = 1, top = 20)

# Modify the theme to rotate X-axis labels to 90 degrees
pc2_contribution +
  theme(
    axis.text.x = element_text(angle = 0),
    plot.title = element_text(hjust = 0)  # horizontal justification
  ) +
  coord_flip() +
  labs(title = "Contribution of Variables to PC1",
       y = "Percentage Contribution",
       x = "",
       caption = "PC1 explains 83.9% of the variance") +
  scale_y_continuous(labels = scales::percent_format(scale = 1,
                                                     accuracy = 1))
```




```
# Contributions of variables to PC2
pc2_contribution <- fviz_contrib(abalone_pca, choice = "var", axes = 2, top = 12)

# Modify the theme to rotate X-axis labels to 90 degrees
pc2_contribution +
  theme(
    axis.text.x = element_text(angle = 0),
    plot.title = element_text(hjust = 0)  # horizontal justification
  ) +
  coord_flip() +
  labs(title = "Contribution of Variables to PC2",
       y = "Percentage Contribution",
       x = "",
       caption = "PC2 explains 8.7% of the variance") +
  scale_y_continuous(labels = scales::percent_format(scale = 1,
                                                     accuracy = 1))
```



3.2.6 Results

The first principal component captures 83.9% of the variance in the data. This linear combination has relatively equal loadings for whole weight, diameter, length, shell weight, viscera weight, and shucked weight, with height and rings having lower loadings. The second principal component is mostly influenced by the variable rings which makes up over 80% of the contribution to PC2. The biplot is an effective visualization of how each variable contributes to PC1, or dimension 1 on the graph, and PC2, or dimension 2 on the graph. The length and direction of each vector represent the contribution of each variable to the principal components; whole weight and rings are the longest, representing the largest contributions to PC1 and PC2 respectively.

PCA is primarily an exploratory tool, which allows us to visualize high-dimensional data in lower dimensions as shown above in the biplot and accompanying scree plot. These PCs can be used to explore data in other ways, such as looking for trends and patterns in the data or identifying clusters and outliers. In the formal analysis in the following chapter, the applications of PCA are further explored through the development of a regression model on the principal components of a dataset.

4 Dataset

This is where we can put everything about our dataset description and visualization.

The principal data used in this analysis was obtained from the "Consumer Assessment of Healthcare Providers and Systems (CAHPS) In-Center Hemodialysis Survey", which is administered to in-center hemodialysis (ICH) facilities by approved survey vendors under the Centers for Medicare & Medicaid Services (CMS). The dataset comprises 39 variables consisting of state-level averages of common dialysis quality measures. The version in this analysis was released on July 19, 2023 through the data.cms.gov website. [24]

The structure of the dataset can be categorized into three main components:

1. Index Variable: The primary index variable is "State," encompassing all 50 states and 6 U.S. territories, namely American Samoa, the District of Columbia, Guam, the Northern Mariana Islands, Puerto Rico, and the U.S. Virgin Islands.
2. Response Variables: The dataset incorporates 24 response variables aligned with ratings of patient care quality in dialysis facilities. These variables pertain to various aspects of dialysis procedures, such as transfusions, fistula usage, infections, hospitalizations, incident patient waitlisting, and readmissions.
3. Classification of Dialysis Patients: Fourteen variables within the dataset classify dialysis patients based on parameters including dialysis adequacy (Kt/V), type of dialysis (hemodialysis vs. peritoneal dialysis), normalized protein catabolic rate (nPCR), hypercalcemia level (Serum Calcium, Mg/dL), serum phosphorus level (Mg/dL), and average hemoglobin (Hgb) level.

The selection of this dataset for our analysis is driven by its inherent characteristic of multicollinearity among variables, indicating that certain variables are less significant in explaining the variability of the response variables. Additionally, all variables in the dataset are numeric, except for the index variable that is excluded from the subsequent Principal Component Analysis (PCA). Our objective is to utilize this dataset to illustrate the efficacy of PCA in dimension reduction and the efficient visualization of data.

4.1 Renaming Variables

In our data preparation process, we have efficiently removed white spaces, and edited variable names, enhancing the readability and interpretability of the dataset. This meticulous

	better_fistula	better_hospital_readmission	better_hospitalization	better_infection	be
Mean	4.928571	2.446429	1.250000	40.000000	3.4
Std.Dev	9.444300	3.201816	2.225881	54.678232	5.1
Min	0.000000	0.000000	0.000000	0.000000	0.0
Q1	0.000000	0.000000	0.000000	5.500000	0.0
Median	2.000000	1.500000	1.000000	22.000000	1.0
Q3	5.500000	3.500000	2.000000	52.000000	4.5
Max	50.000000	17.000000	15.000000	290.000000	24
MAD	2.965200	2.223900	1.482600	26.686800	1.4
IQR	5.250000	3.250000	2.000000	45.250000	4.2
CV	1.916235	1.308771	1.780705	1.366956	1.4
Skewness	3.363870	2.328926	4.353682	2.635591	2.2
SE.Skewness	0.319000	0.319000	0.319000	0.319000	0.3
Kurtosis	11.890419	6.698317	23.383843	7.991801	5.3
N.Valid	56.000000	56.000000	56.000000	56.000000	56
Pct.Valid	100.000000	100.000000	100.000000	100.000000	10

effort adds to the overall clarity, making it quicker, and more meaningful for further examination. For instance, "hypercalcemia_calcium > 10.2Mg", was used in replacement of Percentage.Of.Adult..Patients.With.Hypercalcemia..Serum.Calcium.Greater.Than.10.2.Mg.dL.

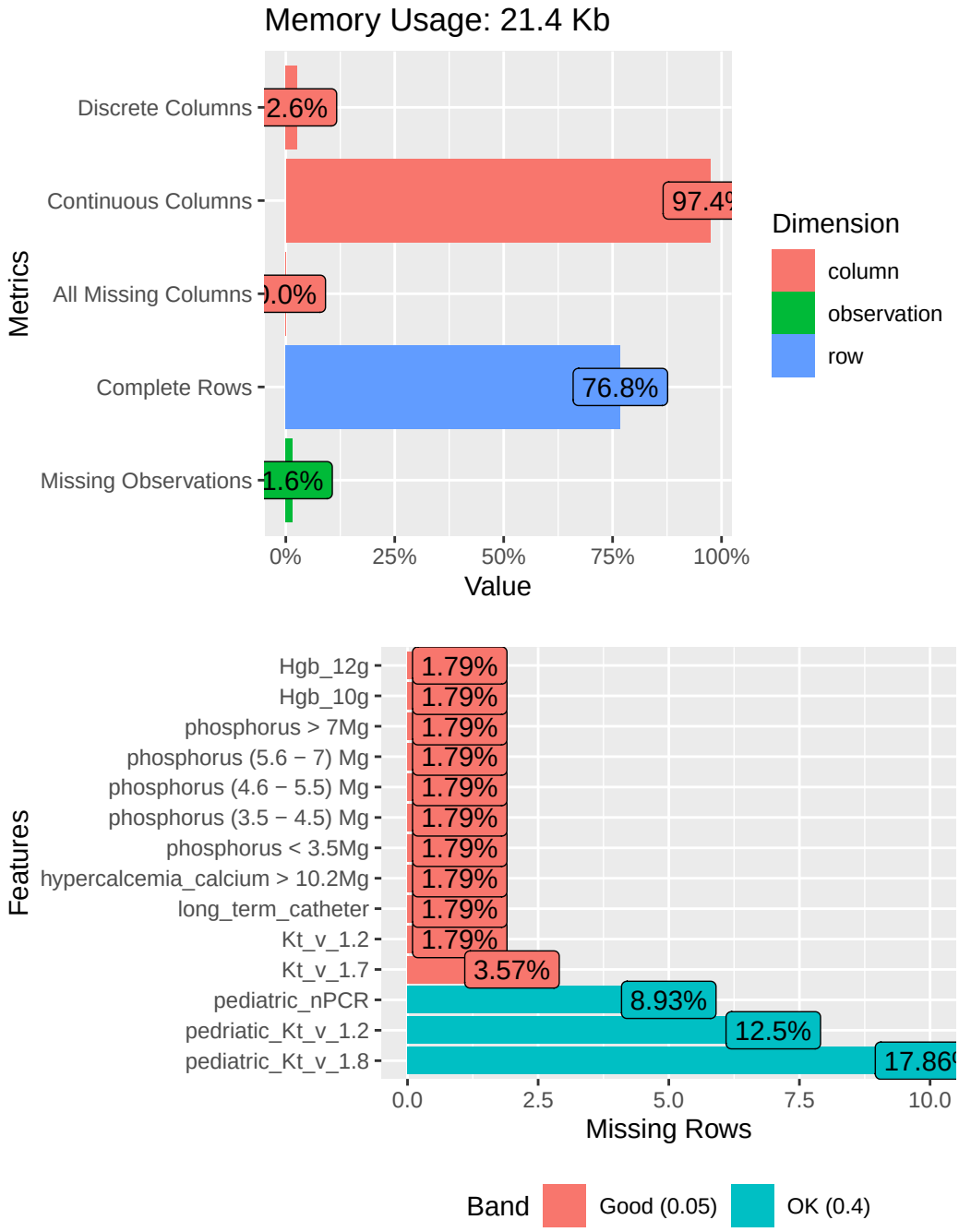
4.2 Statistical Summary

The dataset contains 56 observations of 39 variables with 1 discrete variable (States/Territories) and 38 continuous variables. 13 observations have at least 1 missing record, with 34 missing observations in total. Most missing data in the dataset occurs in variables relating to pediatric patient data.

The histograms of the data reveal that majority of the variables are skewed right, with the QQ plots supporting that very few of the variables are normally distributed. Finally, many of the variables are highly correlated as expected based on the design of the dataset.

4.3 Missing Values Detection

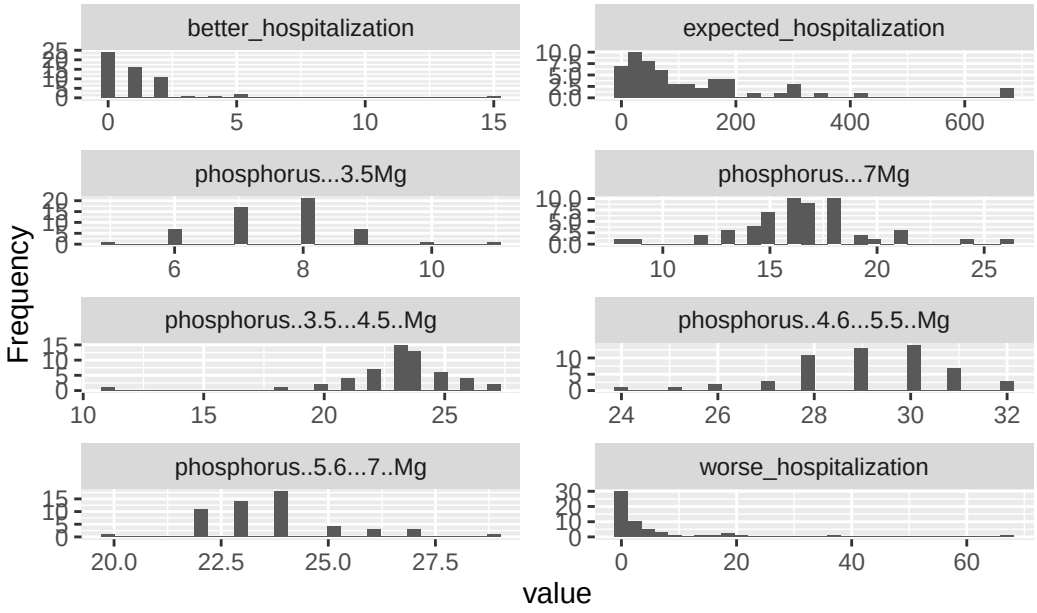
- The 34 missing observations represent 1.6% of the dataset.
- 14 variables have missing observations.
- A table was generated to count missing values for all variables.


Memory Usage: 21.4 Kb
Discrete Columns
Continuous Columns
All Missing Columns
Complete Rows
Missing Observations
Metrics
2.6%
97.4%
0.0%
76.8%
1.6%
0%
25%
50%
75%
100%
Value
Dimension
column
observation
row
Hgb_12g
Hgb_10g
phosphorus > 7Mg
phosphorus (5.6 − 7) Mg
phosphorus (4.6 − 5.5) Mg
phosphorus (3.5 − 4.5) Mg
phosphorus < 3.5Mg
hypercalcemia_calcium > 10.2Mg
long_term_catheter
Kt_v_1.2
Kt_v_1.7
pediatric_nPCR
pedriatic_Kt_v_1.2
pediatric_Kt_v_1.8
Features
1.79%
3.57%
8.93%
12.5%
17.86%
0.0
2.5
5.0
7.5
10.0
Missing Rows
Band
Good (0.05)
OK (0.4)

	Missing_Values
State	0
better_transfusion	0
expected_transfusion	0
worse_transfusion	0
better_infection	0
expected_infection	0
worse_infection	0
Kt_v_1.2	1
Kt_v_1.7	2
pedriatic_Kt_v_1.2	7
pediatric_Kt_v_1.8	10
pediatric_nPCR	5
better_fistula	0
expected_fistula	0
worse_fistula	0
long_term_catheter	1
hypercalcemia_calcium > 10.2Mg	1
phosphorus < 3.5Mg	1
phosphorus (3.5 - 4.5) Mg	1
phosphorus (4.6 - 5.5) Mg	1
phosphorus (5.6 - 7) Mg	1
phosphorus > 7Mg	1
better_hospitalization	0
expected_hospitalization	0
worse_hospitalization	0
better_hospital_readmission	0
expected_hospital_readmission	0
worse_hospital_readmission	0
better_survival	0
expected_survival	0
worse_survival	0
incident_transplant_waitlist_better	0
incident_transplant_waitlist_expected	0
incident_transplant_waitlist_worse	0
prevalent_transplant_waitlist_better	0
prevalent_transplant_waitlist_expected	0
prevalent_transplant_waitlist_worse	0
Hgb_10g	1
Hgb_12g	1

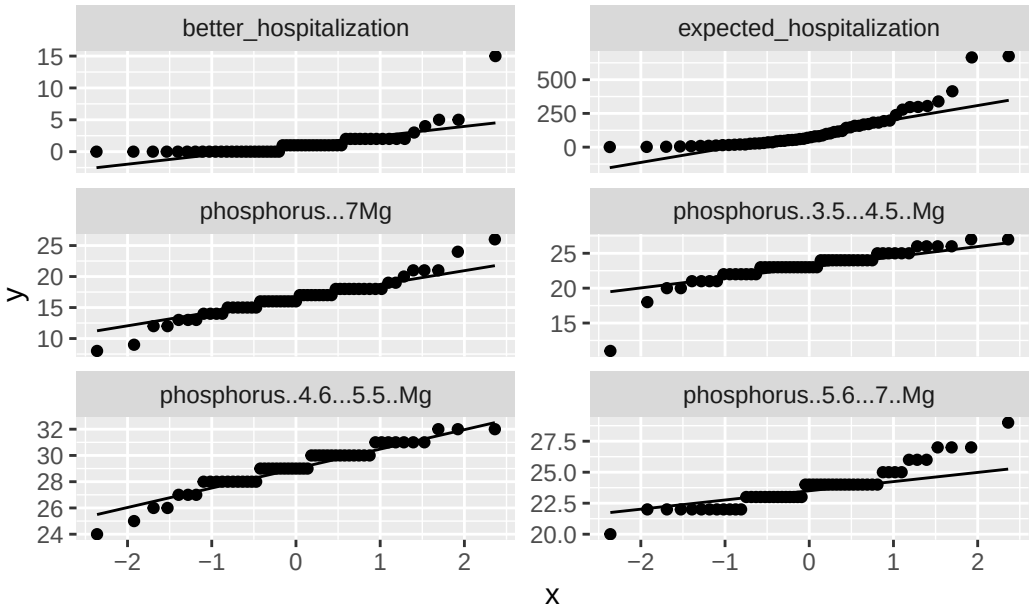
4.4 Data Distribution

- Histograms were used to display a sample (8 variables) of the distribution in respect to the predictor variable.
- Normality is not assumed. The majority of the observations in each variable do not meet the normality assumption.

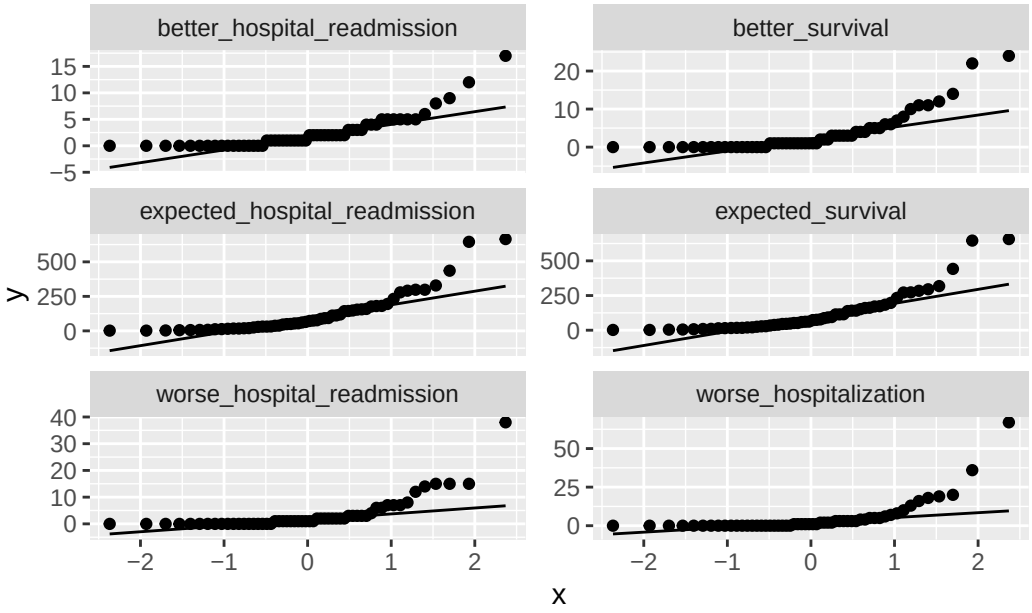


4.5 Normal QQ Plot of Residuals

- It is apparent that the variables have heavy left and right tails.
- The presence of outliers is consistent though the entire dataset.


better_hospitalization
expected_hospitalization
phosphorus...7Mg
phosphorus..3.5...4.5..Mg
phosphorus..4.6...5.5..Mg
phosphorus..5.6...7..Mg
y
x

Page 1


better_hospital_readmission
better_survival
expected_hospital_readmission
expected_survival
worse_hospital_readmission
worse_hospitalization
y
x

Page 2

4.6 Imputation of Missing Values

- After substituting missing values with the mean, we are now able to proceed with standardizing the dataset.
- The standardization process will be implemented in the Analysis chapter.
- A table was created to confirm the absence of any missing observations in the data frame.
- Due to its categorical data type, the "State" variable was omitted from the analysis.

In conclusion, the dataset has undergone essential pre-processing steps, rendering it well-prepared for outliers detection and normalization, which are crucial for robust and accurate model development. Missing observations have been imputed using the mean, ensuring that all variables within the dataset are now numerical. These preliminary steps have not only enhanced the dataset completeness, but also set the stage for further advanced data analysis.

	Missing_Values	Type
better_transfusion	0	double
expected_transfusion	0	double
worse_transfusion	0	double
better_infection	0	double
expected_infection	0	double
worse_infection	0	double
Kt_v_1.2	0	double
Kt_v_1.7	0	double
pedriatic_Kt_v_1.2	0	double
pediatric_Kt_v_1.8	0	double
pediatric_nPCR	0	double
better_fistula	0	double
expected_fistula	0	double
worse_fistula	0	double
long_term_catheter	0	double
hypercalcemia_calcium > 10.2Mg	0	double
phosphorus < 3.5Mg	0	double
phosphorus (3.5 - 4.5) Mg	0	double
phosphorus (4.6 - 5.5) Mg	0	double
phosphorus (5.6 - 7) Mg	0	double
phosphorus > 7Mg	0	double
better_hospitalization	0	double
expected_hospitalization	0	double
worse_hospitalization	0	double
better_hospital_readmission	0	double
expected_hospital_readmission	0	double
worse_hospital_readmission	0	double
better_survival	0	double
expected_survival	0	double
worse_survival	0	double
incident_transplant_waitlist_better	0	double
incident_transplant_waitlist_expected	0	double
incident_transplant_waitlist_worse	0	double
prevalent_transplant_waitlist_better	0	double
prevalent_transplant_waitlist_expected	0	double
prevalent_transplant_waitlist_worse	0	double
Hgb_10g	0	double
Hgb_12g	0	double

5 Analysis

5.1 Data Preparation

This analysis uses the Dialysis dataset introduced previously in the Dataset chapter. Within this context, we restructured the variables, enhancing their meaningfulness and facilitating inferential analysis. Additionally, the mean imputation approach addressed missing values.

5.2 Assumptions

PCA relies on certain assumptions, the fulfillment of which is essential for the validity of this technique.

- **Linearity**: PCA assumes that the dataset is a linear combination of the variables. The variables exhibit relationships among themselves [25].
- **Importance of mean and covariance**: PCA assumes that the directions of maximum variance will contain good features for discrimination.
- **Correlation between features**: PCA assumes a correlation between features.
- **Normalization of data**: Normalization of data is necessary to apply PCA. Unscaled data can cause relative comparison problems of the dataset [25]. In addition, the data should contain no significant outliers.
- **Orthogonality**: The principal components are orthogonal to each other.
- **Sampling adequacy**: PCA assumes that there is sampling adequacy.

5.3 Feature Scaling

5.3.1 Standardization

It is important to Mean-Center the data prior to PCA model building to ensure the first Principal Component is in the direction of maximum variance.

- Standardization produces Mean, $\mu = 0$, and Standard Deviation, $\sigma = 1$.
- We can rewrite this as:

$$Z = \frac{x - \mu}{\sigma}$$

$$Z \sim N(0, 1)$$

```
# Find the index position of the target feature
target_name <- "expected_survival"
target_index <- grep(target_name,
                     colnames(train_data))
```

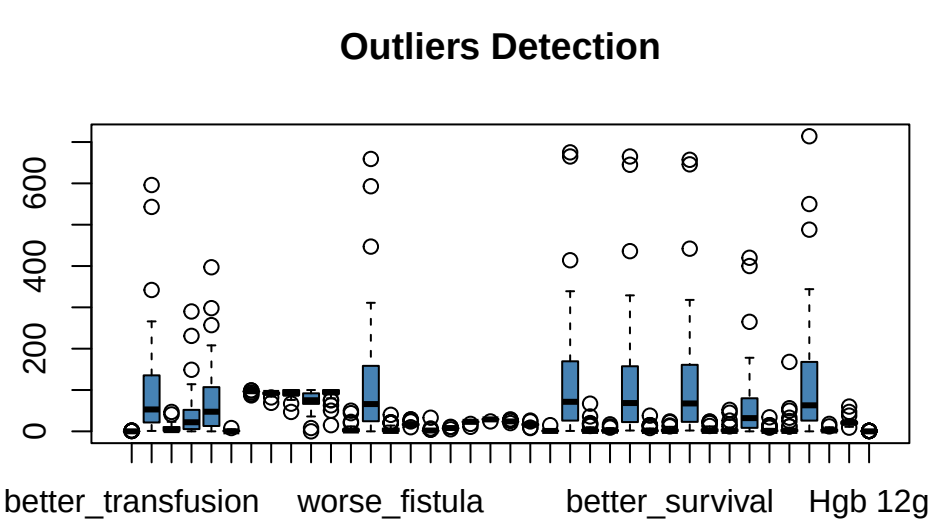
```
# Standardization Numerical Features
train_data_sc <- scale(train_data[, -target_index])
```

5.4 PCA Requirements

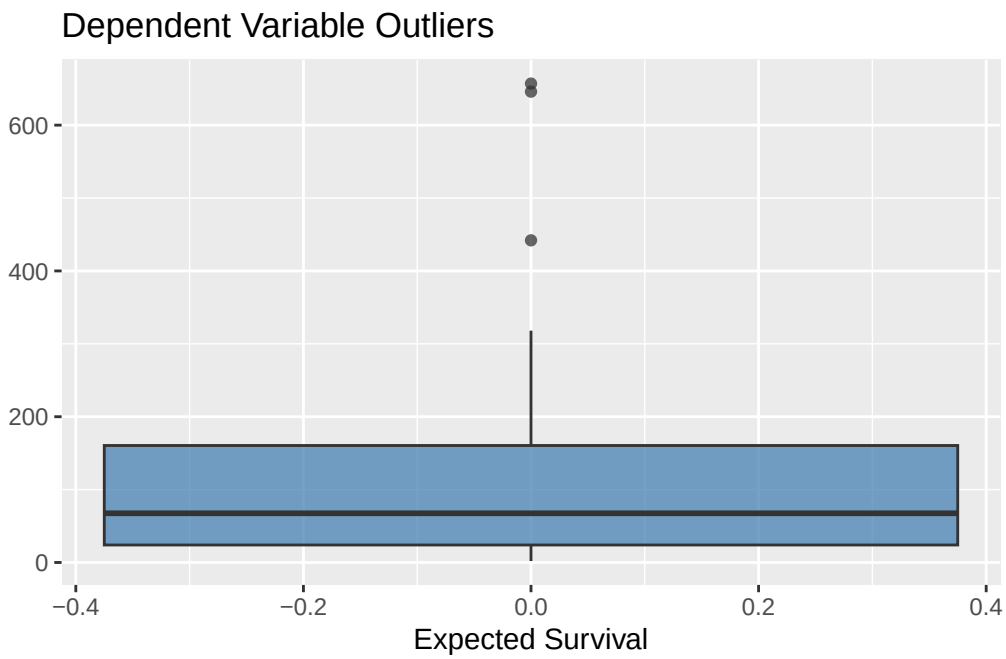
5.4.1 Outliers Detection

- There are some outliers in the data frame including the dependent variable.
- However, there are three outliers with no high leverage.
- Outliers are important because these can have a disproportionate influence on results.

```
# Plot a boxplot to visualize potential outliers
boxplot(train_data, main = "Outliers Detection",
        col = "steelblue")
```

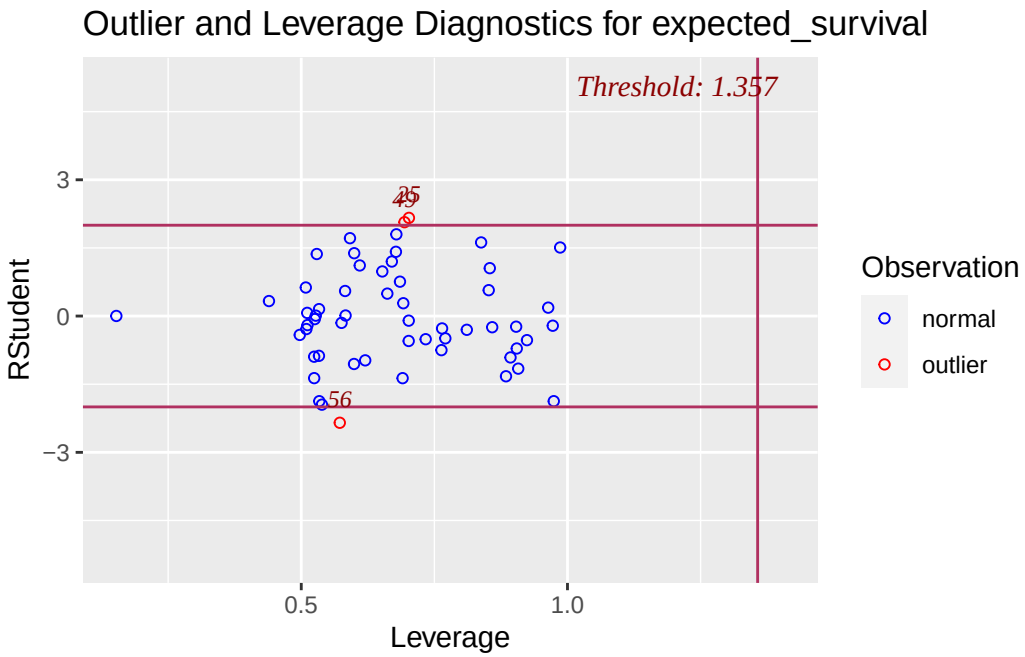



```
# Dependent Variable outliers
train_data %>%
  ggplot(aes(y=expected_survival)) +
  geom_boxplot(fill="steelblue", alpha=0.75) +
  xlab("Expected Survival")+
  ylab("")+
  ggtitle("Dependent Variable Outliers")
```



5.4.2 Leverage

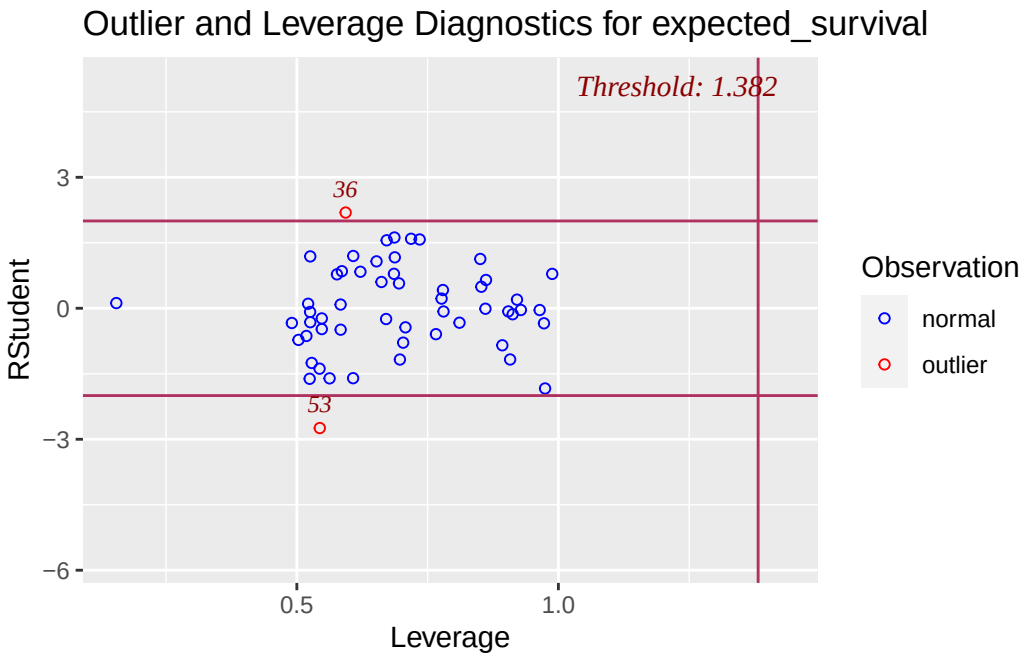
```r
set.seed(my_seed)

# Fit regression model
ordinary_model <- lm(expected_survival ~ ., data = train_data)

# Print the model summary
summary(ordinary_model)

# Residual Diagnostics
ols_plot_resid_lev(ordinary_model)
```



5.4.3 Removing Outliers

After removing one outlier from the dataset, we discovered that the final results produced minimal variability. Therefore, we decided to present a model that captures all the data points' information; thus, no observations will be removed from the final model.

```
# No Outliers subset
no_outliers_df <- slice(train_data, -c(56))

set.seed(my_seed)

# Fit regression model
no_outliers_model <- lm(expected_survival ~ ., data = no_outliers_df)

# Print the model summary
summary(no_outliers_model)

# Residual Diagnostics
ols_plot_resid_lev(no_outliers_model)
```



5.4.4 Correlations

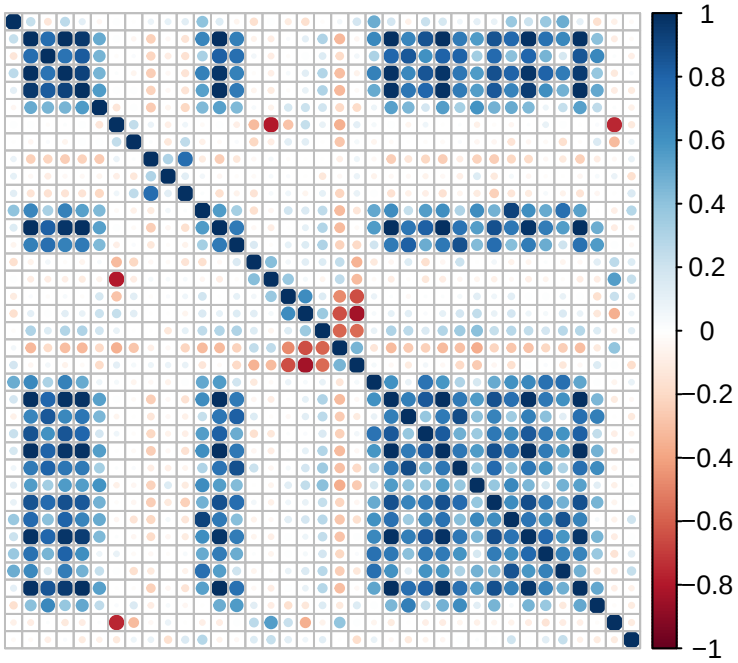
- Multicollinearity is present in the data set.
- 28 Correlated features were identified using a threshold = 0.30.

```
# Calculate correlations and round to 2 digits
corr_matrix <- cor(train_data_sc)
corr_matrix <- round(corr_matrix, digits = 2)

# Print names of highly correlated features; threshold > 0.30
high <- findCorrelation(corr_matrix, cutoff = 0.30, names = TRUE)

# Create a data frame with an index column
high_corr_df <- data.frame(
  Count = 1:length(high),
  Feature = high
)

# table format
kbl(high_corr_df, caption = "Highly Correlated Features") %>%
  kable_paper("hover")
```

Table 5.1: Highly Correlated Features

Count	Feature
1	expected_transfusion
2	better_infection
3	expected_hospitalization
4	expected_hospital_readmission
5	expected_fistula
6	prevalent_transplant_waitlist_expected
7	incident_transplant_waitlist_expected
8	expected_infection
9	worse_survival
10	incident_transplant_waitlist_better
11	better_hospital_readmission
12	worse_transfusion
13	worse_hospital_readmission
14	worse_fistula
15	incident_transplant_waitlist_worse
16	better_fistula
17	prevalent_transplant_waitlist_better
18	better_survival
19	worse_infection
20	phosphorus (5.6 - 7) Mg
21	prevalent_transplant_waitlist_worse
22	phosphorus (3.5 - 4.5) Mg
23	phosphorus > 7Mg
24	better_transfusion
25	Hgb 10g
26	hypercalcemia_calcium > 10.2Mg
27	pediatric_nPCR
28	long_term_catheter

```r
corrplot::corrplot(corr_matrix, type = "full", tl.pos = "n")
```



5.5 Full Model Regression

- The Adjusted $R^2 = 99.99\%$ is an indication of over-fitting, or high variance.
- This dataset is suitable to apply PCA to find the adequate variance balance.

```r
set.seed(my_seed)

# Fit a multiple linear regression model
full_model <- lm(expected_survival ~ ., data = train_data)

# Print a summary of the regression model
tab_model(full_model, title = "Full Model Regression",
          string.p="P-value", string.stat = "T-score",
          string.se = "Std. Error",
          string.resp = "Response",
          string.ci = "Conf Int.",
          show.se=T, show.stat = T,
           CSS = list(
```

```
    css.depvarhead = 'font-weight: bold; text-align: left;',
    css.summary = 'color: #10759B; font-weight: bold;'
  ))
```

Table 5.2: Full Model Regression

			expected_survival		
Predictors	Estimates	Std. Error	Conf Int.	T-score	P-value
(Intercept)	93.71	41.88	5.73 – 181.69	2.24	**0.038**
better transfusion	1.04	0.75	-0.53 – 2.62	1.39	0.181
expected transfusion	0.00	0.03	-0.07 – 0.07	0.03	0.974
worse transfusion	0.13	0.08	-0.03 – 0.29	1.76	0.096
better infection	-0.27	0.09	-0.46 – -0.07	-2.92	**0.009**
expected infection	-0.04	0.06	-0.17 – 0.09	-0.57	0.574
worse infection	-0.11	0.27	-0.68 – 0.46	-0.41	0.687
Kt v 1 2	0.25	0.25	-0.28 – 0.78	0.99	0.335
Kt v 1 7	-0.00	0.05	-0.11 – 0.11	-0.00	0.996
pedriatic Kt v 1 2	-0.04	0.03	-0.11 – 0.03	-1.21	0.241
pediatric Kt v 1 8	-0.02	0.01	-0.04 – 0.00	-1.89	0.076
pediatric nPCR	0.04	0.03	-0.01 – 0.10	1.75	0.097
better fistula	-0.18	0.11	-0.41 – 0.04	-1.73	0.101
expected fistula	0.01	0.08	-0.15 – 0.16	0.08	0.935
worse fistula	-0.03	0.11	-0.25 – 0.19	-0.28	0.780
long term catheter	0.31	0.07	0.16 – 0.47	4.19	**0.001**
hypercalcemia	-0.19	0.09	-0.37 – -0.00	-2.13	**0.048**
calcium > 10 2Mg					
phosphorus < 3 5Mg	0.35	0.48	-0.66 – 1.36	0.73	0.476
phosphorus (3 5 - 4 5) Mg	-1.73	0.46	-2.71 – -0.76	-3.73	**0.002**

phosphorus (4 6 - 5 5) Mg	-1.36	0.46	-2.33 – -0.40	-2.97	**0.008**
phosphorus (5 6 - 7) Mg	-0.80	0.44	-1.72 – 0.12	-1.82	0.085
phosphorus > 7Mg	-1.44	0.46	-2.41 – -0.48	-3.15	**0.005**
better hospitalization	0.08	0.32	-0.59 – 0.76	0.26	0.797
expected hospitalization	0.70	0.15	0.38 – 1.03	4.55	**<0.001**
worse hospitalization	0.52	0.22	0.05 – 0.98	2.34	**0.031**
better hospital readmission	0.38	0.24	-0.13 – 0.88	1.55	0.139
expected hospital readmission	0.31	0.13	0.03 – 0.60	2.34	**0.031**
worse hospital readmission	0.09	0.24	-0.41 – 0.59	0.39	0.704
better survival	-0.57	0.15	-0.88 – -0.27	-3.94	**0.001**
worse survival	-0.91	0.14	-1.20 – -0.62	-6.64	**<0.001**
incident transplant waitlist	0.01	0.13	-0.27 – 0.29	0.09	0.926
better incident transplant waitlist	0.03	0.04	-0.05 – 0.12	0.82	0.421
expected incident transplant waitlist	0.31	0.12	0.06 – 0.56	2.61	**0.018**
worse prevalent transplant waitlist	0.15	0.11	-0.08 – 0.38	1.36	0.191
better					

prevalent transplant waitlist	0.04	0.10	-0.17 – 0.25	0.41	0.690
expected prevalent transplant waitlist worse	0.01	0.19	-0.39 – 0.40	0.03	0.975
Hgb 10g	-0.10	0.05	-0.20 – -0.00	-2.14	**0.046**
Hgb 12g	-0.36	0.66	-1.75 – 1.04	-0.54	0.597
Observations	56				
R^2 / R^2 adjusted	1.000 / 1.000				

5.6 PCA Implementation

5.6.1 SVD - Singular Value Decomposition

We will focus on Singular Value Decomposition which is a classic approach for PCA analysis.

Singular Value Decomposition is a factorization technique used in linear algebra to decompose a matrix into three matrices.

1. L: A matrix whose columns are the left singular vectors of the original matrix.
2. D: A diagonal matrix whose entries are the singular values of the original matrix.
3. R: A matrix whose columns are the right singular vectors of the original matrix.

The SVD is closely related to the eigenvalue decomposition (EVD), which is another factorization technique used in linear algebra. While the EVD can only be applied to square matrices, the SVD can be applied to any matrix, including rectangular matrices. The SVD is also more numerically stable than the EVD, making it a preferred method for many applications.

- Note: The Spectral Decomposition approach is used with the princomp() function.

```
# Apply PCA using prcomp()
data_pca <- prcomp(train_data_sc, center = TRUE, scale. = TRUE)
pca_summ <- summary(data_pca)$importance

# Transpose the matrix
transposed_pca_summ <- t(pca_summ)
```

```r
# table format
kbl(transposed_pca_summ, caption = "Pricipal Components Analysis",
    digits = 4) %>%
  kable_paper("hover")
```

5.6.2 PCA - Elements

- The values in '**data_pca$x`** are the coordinates of each observation in the new principal component space. These coordinates are the scores for each observation along each principal component.
- The eigenvectors of the covariance, or correlation matrix of the data represent the directions of maximum variance, or information in the dataset.

```r
# Principal Component scores vector
pc_scores <- data_pca$x

# Std Deviation of Components
component_sdev <- data_pca$sdev

# Eigenvector, or Loadings
pc_loadings <- data_pca$rotation

# Mean of variables
component_mean <- data_pca$center

# Scaling factor of Variables
component_scale <- data_pca$scale
```

5.6.3 Loadings of First Two Components

- The loading 'data_pca$rotation' are the weights assigned to each original variable for that particular principal component.

```r
# Access the loadings for the first two principal components
loadings_first_two_components <- pc_loadings[, 1:2]

# Print the loadings for the first two principal components
# print(loadings_first_two_components)
```

Table 5.3: Pricipal Components Analysis

	Standard deviation	Proportion of Variance	Cumulative Proportion
PC1	3.8854	0.4080	0.4080
PC2	1.8717	0.0947	0.5027
PC3	1.8377	0.0913	0.5940
PC4	1.7487	0.0826	0.6766
PC5	1.4227	0.0547	0.7313
PC6	1.2508	0.0423	0.7736
PC7	1.1280	0.0344	0.8080
PC8	1.0933	0.0323	0.8403
PC9	0.9449	0.0241	0.8644
PC10	0.9081	0.0223	0.8867
PC11	0.8185	0.0181	0.9048
PC12	0.7790	0.0164	0.9212
PC13	0.6815	0.0126	0.9338
PC14	0.6366	0.0110	0.9447
PC15	0.5710	0.0088	0.9535
PC16	0.5055	0.0069	0.9604
PC17	0.5018	0.0068	0.9673
PC18	0.4884	0.0064	0.9737
PC19	0.4625	0.0058	0.9795
PC20	0.4169	0.0047	0.9842
PC21	0.3534	0.0034	0.9876
PC22	0.3247	0.0029	0.9904
PC23	0.2884	0.0022	0.9926
PC24	0.2604	0.0018	0.9945
PC25	0.2259	0.0014	0.9959
PC26	0.2149	0.0013	0.9971
PC27	0.1982	0.0011	0.9982
PC28	0.1670	0.0008	0.9989
PC29	0.1268	0.0004	0.9994
PC30	0.1091	0.0003	0.9997
PC31	0.0838	0.0002	0.9999
PC32	0.0509	0.0001	0.9999
PC33	0.0317	0.0000	1.0000
PC34	0.0287	0.0000	1.0000
PC35	0.0143	0.0000	1.0000
PC36	0.0081	0.0000	1.0000
PC37	0.0049	0.0000	1.0000

```r
# table format
kbl(loadings_first_two_components,
    caption = "Loadings of First Two Principal Components",
    digits = 4) %>%
  kable_paper("hover")
```

5.6.4 PCA - Cumulative Variance

```r
# Proportion of variance explained by each PC
variance_explained <- component_sdev^2 / sum(component_sdev^2)

# Cumulative proportion of variance explained
cumulative_variance_explained <- cumsum(variance_explained)

# Create a data frame with an index column
cumulative_variance_explained_df <- data.frame(
  Pincipal_Component = 1:length(cumulative_variance_explained),
  Cumulative_Variance_Explained = cumulative_variance_explained
)

# Create a kable table with an index column
kbl(cumulative_variance_explained_df, align = "cl",
    caption = "PCA: Cumulative Variance Explained",
    digits = 4) %>%
  kable_paper("hover")
```

5.6.5 PCA - Number of Principal Components

- We can conclude that 9 Principal Components explain 86% of the variance.

```r
# Retain components that explain a percentage of the variance
num_components <- which(cumulative_variance_explained >= 0.86)[1]

# Select the desired number of principal components
selected_pcs <- pc_scores[, 1:num_components]

# table format
kbl(selected_pcs, caption = "Components Explaining 86% Variance",
    digits = 4) %>%
```

Table 5.4: Loadings of First Two Principal Components

	PC1	PC2
better_transfusion	0.0506	0.0151
expected_transfusion	0.2536	-0.0214
worse_transfusion	0.2058	-0.1647
better_infection	0.2526	-0.0073
expected_infection	0.2455	-0.0652
worse_infection	0.1381	0.0637
Kt_v_1.2	0.0031	0.0081
Kt_v_1.7	-0.0078	0.0444
pedriatic_Kt_v_1.2	-0.0636	0.0527
pediatric_Kt_v_1.8	-0.0139	0.0028
pediatric_nPCR	-0.0394	0.0768
better_fistula	0.1701	0.1849
expected_fistula	0.2521	-0.0558
worse_fistula	0.1932	-0.0481
long_term_catheter	-0.0054	0.0409
hypercalcemia_calcium > 10.2Mg	-0.0276	0.1049
phosphorus < 3.5Mg	0.0161	0.3759
phosphorus (3.5 - 4.5) Mg	0.0420	0.4258
phosphorus (4.6 - 5.5) Mg	0.0888	0.2665
phosphorus (5.6 - 7) Mg	-0.0954	-0.3297
phosphorus > 7Mg	-0.0293	-0.4400
better_hospitalization	0.1555	0.0398
expected_hospitalization	0.2535	-0.0353
worse_hospitalization	0.1863	-0.1048
better_hospital_readmission	0.2100	-0.0462
expected_hospital_readmission	0.2537	-0.0414
worse_hospital_readmission	0.2010	-0.0432
better_survival	0.1523	0.1688
worse_survival	0.2288	-0.0662
incident_transplant_waitlist_better	0.2016	0.1513
incident_transplant_waitlist_expected	0.2499	-0.0578
incident_transplant_waitlist_worse	0.1913	-0.0199
prevalent_transplant_waitlist_better	0.1618	0.1763
prevalent_transplant_waitlist_expected	0.2501	-0.0703
prevalent_transplant_waitlist_worse	0.1202	-0.2285
Hgb 10g	-0.0299	-0.0804
Hgb 12g	-0.0006	0.1765

Table 5.5: PCA: Cumulative Variance Explained

Pincipal_Component	Cumulative_Variance_Explained
1	0.4080
2	0.5027
3	0.5940
4	0.6766
5	0.7313
6	0.7736
7	0.8080
8	0.8403
9	0.8644
10	0.8867
11	0.9048
12	0.9212
13	0.9338
14	0.9447
15	0.9535
16	0.9604
17	0.9673
18	0.9737
19	0.9795
20	0.9842
21	0.9876
22	0.9904
23	0.9927
24	0.9945
25	0.9959
26	0.9971
27	0.9982
28	0.9989
29	0.9994
30	0.9997
31	0.9999
32	0.9999
33	1.0000
34	1.0000
35	1.0000
36	1.0000
37	1.0000

```
    kable_paper("hover", full_width = F)
```

\begin{table}

\caption{Components Explaining 86% Variance}

PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9
-3.1378	0.7718	-0.7364	-0.9222	-0.3593	0.3854	1.7645	0.7137	-1.2776
0.9109	-1.3736	-0.8907	-0.0460	0.0228	0.3267	1.9095	0.3203	0.3158
-1.3650	-1.7656	-0.1449	0.2592	-1.2857	-0.3762	-0.7856	-0.9204	-1.2626
-3.0594	0.3348	-0.0141	-0.1395	0.0920	-0.2825	-0.1286	0.1373	0.0780
-0.1991	-0.2073	-0.7149	-1.3756	-1.1993	0.0319	0.7499	0.7160	-0.1954
15.2032	2.6187	-7.7363	4.0519	-0.8244	-2.8218	-0.3937	0.7540	0.1236
-1.1596	1.1216	-0.7112	-0.6842	0.2286	-0.3821	-0.5148	-1.3731	0.5560
-1.7567	2.4464	0.2336	-0.7871	-0.1098	-1.0603	-0.1434	-1.6411	1.1462
-2.5605	1.0785	-0.0207	-0.0018	-0.2474	1.1930	-0.1234	1.6484	1.1223
-2.2725	0.7670	-0.0552	-1.0045	2.4430	-1.8092	1.3011	0.0402	0.7853
10.2101	-3.0951	5.5631	-2.7079	-3.2328	-1.4788	-1.0195	0.3946	0.0182
5.1286	-1.7705	1.7849	-0.1628	0.5176	-0.3672	1.2737	0.3839	-0.1914
-3.4958	-4.4206	-3.1425	0.0404	-0.7928	0.4217	0.1087	-1.1395	-1.5791
-1.7134	0.8230	-1.9663	-1.0871	0.7505	0.4357	1.1010	0.8853	-1.1906
-1.9511	0.9075	-0.3459	-1.4707	-0.7735	-0.7810	-1.0116	-1.4774	0.2215
-2.6296	0.9149	-0.5263	-1.3484	-0.9348	0.5181	0.3570	0.8863	-0.4488
4.7463	0.3146	2.3709	-0.2528	0.8329	1.4290	-0.8492	0.0128	-0.0479
0.3868	-0.9290	0.6870	-0.6117	1.5374	-0.1113	0.2265	0.7691	-0.8385
-1.6767	0.2850	-0.9842	-1.8133	-0.8136	0.7518	0.8450	-0.2605	0.7893
-0.7345	-1.3222	0.3700	-0.8448	2.0553	-1.6018	0.7571	-1.1866	1.6751
0.9393	-0.6999	-0.1399	-0.5978	-0.0203	-0.0046	1.3170	0.4944	0.4746
-0.8326	1.5874	-0.7312	0.2910	0.1199	1.1100	-1.6455	1.1312	-0.0448
0.8725	1.0451	0.2775	0.2175	-0.6226	0.5429	-0.2118	0.9699	1.6165
-2.7730	0.5328	-0.3857	-0.9341	0.8943	-2.7237	-0.4738	-2.6266	1.1451
2.0146	0.0014	0.6403	-0.6929	-0.1677	0.4555	-0.3840	0.2754	1.0045
-1.1376	0.6676	0.0612	0.2444	-0.4465	0.9265	-0.6966	0.9414	0.1757
0.1256	-0.9951	1.1503	-0.4897	0.7108	1.3854	0.0279	-0.3615	-0.4521
-3.6486	-5.1247	0.0491	6.3322	-1.4438	1.4726	-0.7719	-0.9060	2.9209
-0.3307	0.2793	-0.0985	-0.3388	-0.5950	-0.5761	2.0137	0.0524	0.9485
-3.1386	-2.4847	-2.0044	1.0661	-0.5877	-0.0292	-0.7689	-0.0432	-0.9302
1.8304	-0.8659	-0.6292	-0.2888	-0.3585	1.1522	1.4099	-0.2387	0.1117
-2.6300	1.8820	1.4790	1.5197	0.1484	0.9943	1.0138	1.7547	-0.2904
-1.9654	1.4437	0.4239	-1.4217	0.3452	0.1489	0.2267	0.8292	0.0537
-2.9434	-0.4453	-0.1869	0.0501	-0.3471	-0.3884	-1.8441	0.4520	0.3853
1.9335	4.2274	-1.5429	-0.8354	-0.5412	2.3739	-0.2919	0.2478	0.0526
-1.4611	0.5551	-1.4467	-0.2732	0.4168	-0.3335	0.0306	1.0760	-0.1292
-1.6820	-0.9253	-0.3177	-0.2737	-0.7852	0.2036	-0.4687	0.5302	-0.4388
6.7595	3.6388	1.0562	2.3514	2.4970	3.4372	-1.4276	-3.6075	-1.6018
3.4507	-1.1824	2.5699	-1.1129	-0.0546	1.4361	-0.6591	-1.0345	-0.1605
-1.0984	-1.5241	-0.5320	-0.4777	-0.9995	-0.8030	-0.1950	-1.6084	-0.6964
-1.9193	-0.0165	-1.0806	-0.5872	-0.6324	-0.2014	-1.3158	-1.3027	0.1718
3.4962	1.5199	1.3109	-0.3007	2.5985	1.0499	-0.2313	-0.0969	0.8464
-0.2522	3.0718	2.1407	-1.9255	-2.2719	-3.4093	-2.1544	0.2271	-1.2587
-3.0135	-0.6858	-1.1712	-0.8551	-0.4428	0.8391	-1.0039	0.4085	-0.1121
0.3392	-0.5133	0.0391	-0.1766	-0.5924	1.0593	1.1663	0.5207	-0.3248
-2.1136	2.6100	0.5330	-0.8478	0.0752	0.1416	1.0927	-0.3588	1.3478
1.3707	-2.9104	1.0848	1.3163	6.1381	-2.1436	0.9717	0.1946	-1.2873
13.0708	-2.7257	0.4257	-0.4752	-0.7834	0.5789	0.9620	0.5196	-0.2282
-2.6465	-0.3807	-0.6944	1.0365	-0.9202	0.4536	0.2146	-0.6048	-0.7532
1.7505	-0.1798	-0.2689	-0.3411	0.6075	-0.7767	-0.0058	0.2781	1.5233

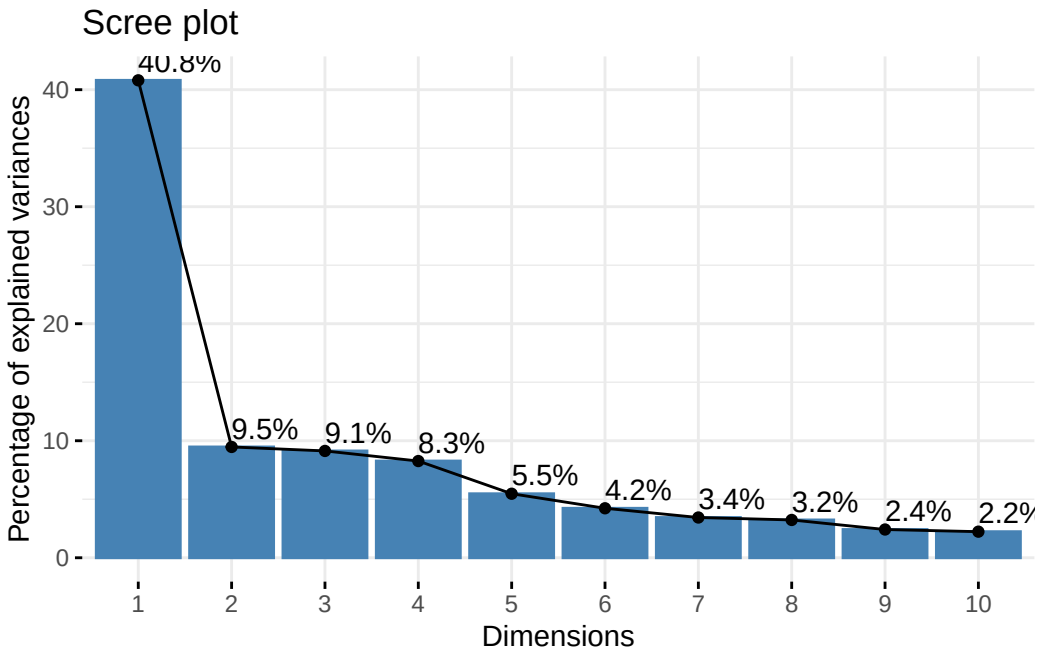
\end{table}

5.7 Visualization

5.7.1 Scree Plot - Cumulative Variance Explained

- PC1 explains 40.8% variance.
- PC2 explains 9.5% variance.

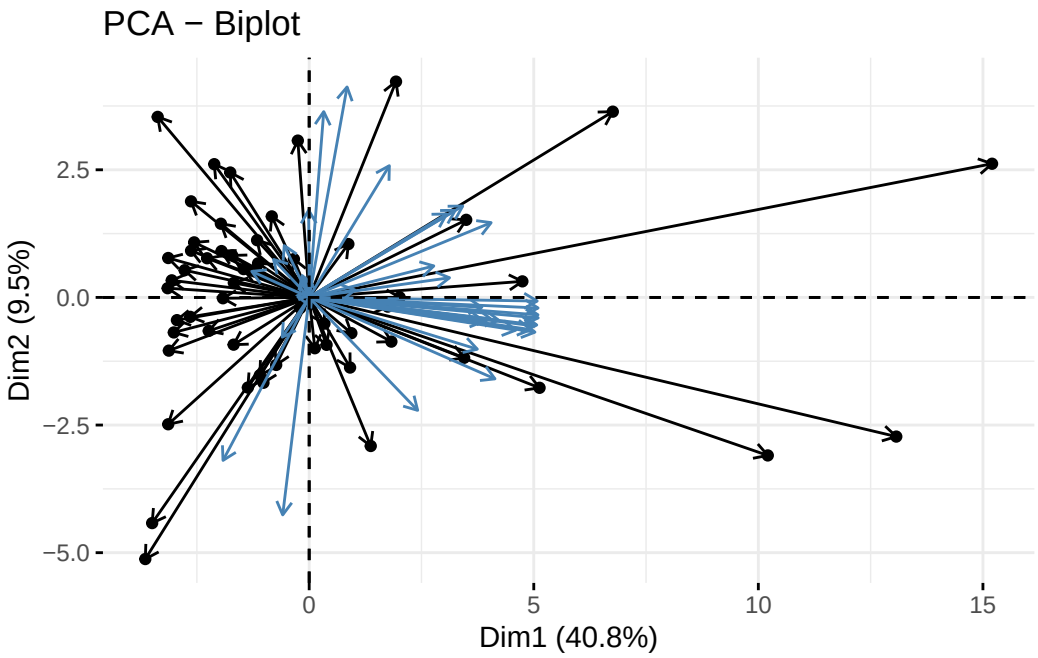
```
fviz_eig(data_pca, addlabels = TRUE)
```



5.7.2 Biplot

- The correlation between a variable and a principal component (PC) is used as the coordinates of the variable on the PC. The representation of variables differs from the plot of the observations: The observations are represented by their projections, but the variables are represented by their correlations [10].
- PC1 is represented in black which displays the longest distance of its projection.
- PC2 is represented in blue which displays a shorter distance as expected.

```
fviz_pca_biplot(data_pca,
                geom = c("point", "arrow"),
                geom.var = "arrow")
```



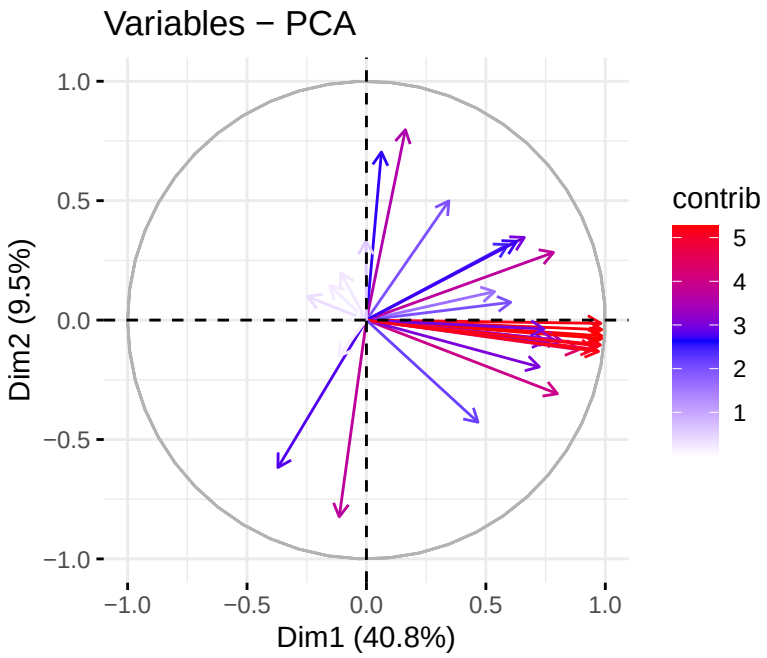
5.7.3 Correlation Circle

The plot below is also known as variable correlation plots. It shows the relationships between all variables. It can be interpreted as follow:

- Positively correlated variables are grouped together.
- Negatively correlated variables are positioned on opposite sides of the plot origin (opposed quadrants).
- The distance between variables and the origin measures the quality of the variables on the factor map. Variables that are away from the origin are well represented on the factor map.

```
# Control variable colors using their contributions
fviz_pca_var(data_pca, col.var = "contrib",
   gradient.cols = c("white", "blue", "red"),
   geom.var = "arrow",
```

```
    ggtheme = theme_minimal())
```

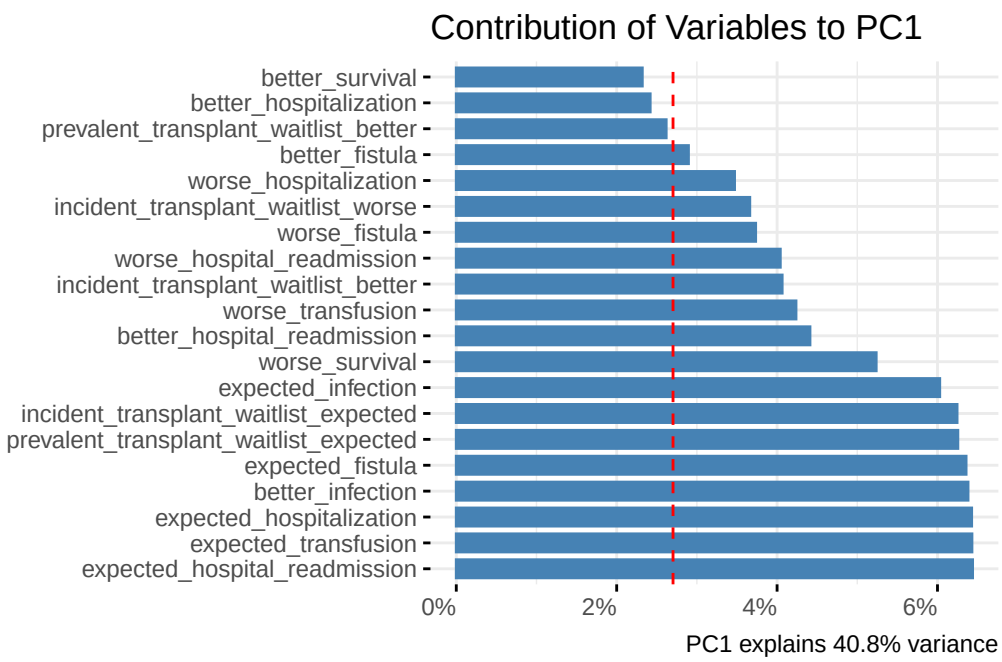


5.7.4 Variable Contribution

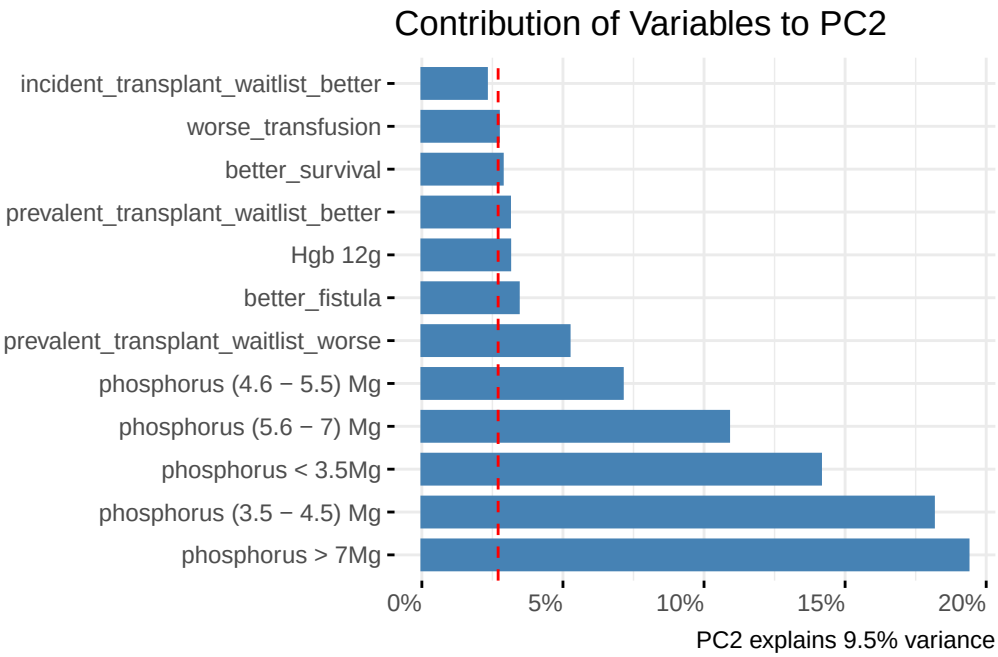
Top variable contribution for the first two principal components.

```
# Contributions of variables to PC1
pc2_contribution <- fviz_contrib(data_pca, choice = "var", axes = 1, top = 20)

# Modify the theme to rotate X-axis labels to 90 degrees
pc2_contribution +
  theme(
    axis.text.x = element_text(angle = 0),
    plot.title = element_text(hjust = 0)  # horizontal justification
  ) +
  coord_flip() +
  labs(title = "Contribution of Variables to PC1",
       y = "Percentage Contribution",
       x = "",
       caption = "PC1 explains 40.8% variance") +
  scale_y_continuous(labels = scales::percent_format(scale = 1,
```

```
                                                    accuracy = 1))
```




```
# Contributions of variables to PC2
pc2_contribution <- fviz_contrib(data_pca, choice = "var", axes = 2, top = 12)

# Modify the theme to rotate X-axis labels to 90 degrees
pc2_contribution +
  theme(
    axis.text.x = element_text(angle = 0),
    plot.title = element_text(hjust = 0)  # horizontal justification
  ) +
  coord_flip() +
  labs(title = "Contribution of Variables to PC2",
       y = "Percentage Contribution",
       x = "",
       caption = "PC2 explains 9.5% variance") +
  scale_y_continuous(labels = scales::percent_format(scale = 1,
                                                     accuracy = 1))
```



5.8 Model Building

5.8.1 Data Splitting into Training & Test set

```
# reproducible random sampling
set.seed(my_seed)

# Create Target y-variable for the training set
y <- train_data$expected_survival
# Split the data into training and test sets
split <- sample.split(y, SplitRatio = 0.7)
training_set <- subset(train_data, split == TRUE)
test_set <- subset(train_data, split == FALSE)
```

5.8.2 Feature Scaling: Standardization

- Standardization ensures all features are on the same scale, and this method is less sensitive to outliers.

```r
# Feature Scaling: Standardization
# Perform centering and scaling on the training and test sets
sc <- preProcess(training_set[, -target_index],
                 method = c("center", "scale"))
training_set[, -target_index] <- predict(
  sc, training_set[, -target_index])
test_set[, -target_index] <- predict(sc, test_set[, -target_index])
```

5.8.3 Applying PCA to Training & Test sets

```r
# Perform Principal Component Analysis (PCA) preprocessing on the training data
pca <- preProcess(training_set[, -target_index],
                  method = 'pca', pcaComp = 8)

# Apply PCA transformation to original training set
training_set <- predict(pca, training_set)

# Reorder columns, moving the dependent feature index to the end
training_set <- training_set[c(2:9, 1)]

# Apply PCA transformation to original test set
test_set <- predict(pca, test_set)

# Reorder columns, moving the dependent feature index to the end
test_set <- test_set[c(2:9, 1)]
```

5.9 PCA Full Model: 8 Principal Components

```r
# reproducible random sampling
set.seed(my_seed)

# Fit a multiple linear regression model
pca_full_model <- lm(expected_survival ~ ., data = training_set)

# Print a summary of the regression model
tab_model(pca_full_model, title = "8 Principal Components Regression",
          string.p="P-value", string.stat = "T-score",
          string.se = "Std. Error",
```

```
        string.resp = "Response",
        string.ci = "Conf Int.",
        show.se=T, show.stat = T,
        CSS = list(
           css.depvarhead = 'font-weight: bold; text-align: left;',
           css.summary = 'color: #10759B; font-weight: bold;'
         ))
```

Table 5.6: 8 Principal Components Regression

	expected_survival				
Predictors	Estimates	Std. Error	Conf Int.	T-score	P-value
(Intercept)	97.05	2.84	91.25 – 102.85	34.19	**<0.001**
PC1	29.40	0.71	27.95 – 30.86	41.22	**<0.001**
PC2	4.59	1.33	1.87 – 7.31	3.44	**0.002**
PC3	-1.11	1.64	-4.46 – 2.24	-0.68	0.504
PC4	4.99	1.75	1.42 – 8.56	2.85	**0.008**
PC5	-3.27	2.25	-7.86 – 1.31	-1.46	0.156
PC6	6.98	2.44	2.01 – 11.95	2.87	**0.008**
PC7	-6.60	2.56	-11.83 – -1.37	-2.58	**0.015**
PC8	3.80	2.82	-1.96 – 9.56	1.35	0.188
Observations	39				
R^2 / R^2 adjusted	0.983 / 0.979				

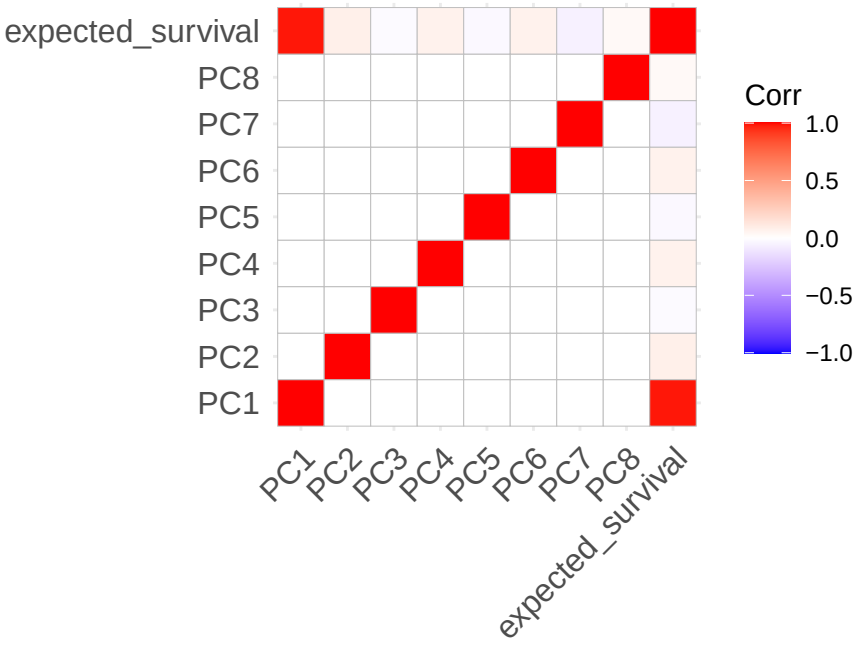
```
# Calculate PRESS
# cat("PRESS: ", PRESS(pca_full_model), "\n")
PRESS_8pc <- PRESS(pca_full_model)
# Calculate predicted R^2
# cat("Predicted R^2: ", pred_r_squared(pca_full_model), "\n")
R2_8pcs <- pred_r_squared(pca_full_model)

predict_8pc <- cbind(PRESS_8pc, R2_8pcs)
# Print PRESS, predicted R^2
# table format
kbl(predict_8pc, caption = "8 Principal Components Prediction Metrics",
    digits = 4) %>%
  kable_paper("hover", full_width = F)
```

Table 5.7: 8 Principal Components Prediction Metrics

PRESS_8pc	R2_8pcs
31662.39	0.943

5.9.1 Visualization of Uncorrelated PCA Matrix

```
# Visual of Principal Components un-correlation
corr_matrix <- cor(training_set)
ggcorrplot(corr_matrix)
```



5.10 PCA: 2 Principal Components

```
# Create a subset with 2 principal components
significant_pcs = c(1,2,9)
train_pca <- training_set[, significant_pcs]
test_pca <- test_set[, significant_pcs]
```

```r
# reproducible random sampling
set.seed(my_seed)

# Fit a multiple linear regression model
reg_model <- lm(expected_survival ~ .,
                data = train_pca)

# Print a summary of the regression model
tab_model(reg_model, title = "2 Principal Components Regression",
          string.p="P-value", string.stat = "T-score",
          string.se = "Std. Error",
          string.resp = "Response",
          string.ci = "Conf Int.",
          show.se=T, show.stat = T,
          CSS = list(
             css.depvarhead = 'font-weight: bold; text-align: left;',
             css.summary = 'color: #10759B; font-weight: bold;'
           ))
```

Table 5.8: 2 Principal Components Regression

	expected_survival				
Predictors	Estimates	Std. Error	Conf Int.	T-score	P-value
(Intercept)	97.05	3.58	89.78 – 104.32	27.07	**<0.001**
PC1	29.40	0.90	27.58 – 31.23	32.64	**<0.001**
PC2	4.59	1.68	1.18 – 8.00	2.73	**0.010**
Observations	39				
R^2 / R^2 adjusted	0.968 / 0.966				

```r
# Calculate PRESS
# cat("PRESS: ", PRESS(reg_model), "\n")
PRESS_2pc <- PRESS(reg_model)

# Calculate predicted R^2
# cat("Predicted R^2: ", pred_r_squared(reg_model), "\n")
R2_2pc <- pred_r_squared(reg_model)

# Print 2PC prediction results
predict_2pc <- cbind(PRESS_2pc, R2_2pc)
```

Table 5.9: 2 Principal Components Prediction Metrics

PRESS_2pc	R2_2pc
25699.6	0.9538

```
# table format
kbl(predict_2pc, caption = "2 Principal Components Prediction Metrics",
    digits = 4) %>%
  kable_paper("hover", full_width = F)
```

5.10.1 Principal Components Regression

- PCA is used to calculate principal components that can then be used in principal components regression. This type of regression is often used when multicollinearity exists between predictors in a data set.

```
# reproducible random sampling
set.seed(my_seed)

y = train_pca$expected_survival

# fit PCR
pcr_model <- pcr(y ~ PC1+PC2, data=train_pca, validation="CV")
print(pcr_model)
```

```
Principal component regression, fitted with the singular value decomposition algorithm.
Cross-validated using 10 random segments.
Call:
pcr(formula = y ~ PC1 + PC2, data = train_pca, validation = "CV")
```

```
# table format
kbl(pcr_model$residuals, caption = "PCA Residuals",
    digits = 4) %>%
  kable_paper("hover", full_width = F)
```

Table 5.10: PCA Residuals

	y.1 comps	y.2 comps
1	0.1506	-0.7094
2	14.8415	8.7438
3	-9.9048	-15.6499
4	-11.9950	-10.5934
5	14.5581	7.4172
6	10.5628	-4.1105
7	-1.1997	-1.8621
8	-13.8770	-8.7908
10	-10.6707	-7.5478
13	15.9325	-6.8066
14	-28.4995	-32.2191
15	13.7912	10.2807
19	2.2654	-4.9708
20	20.7004	18.6124
22	-22.7913	-18.7054
24	0.3036	-1.8158
27	15.1056	16.6785
29	-18.5425	-18.3174
30	7.9759	-3.5721
31	61.8084	55.7883
32	-20.9204	-5.0790
33	-12.8183	-9.2901
34	1.6594	-0.6758
35	-17.3032	-12.3010
36	-17.2435	-19.9354
37	-4.2019	-9.4719
38	-90.8171	-71.1562
39	57.1428	59.6527
41	10.0589	4.0816
42	18.5721	29.7334
44	3.2955	-4.0228
45	8.2807	7.3105
46	-17.5634	-10.1654
49	14.4348	13.0494
50	5.5387	3.5868
51	-9.4323	29.4206
54	10.5787	15.4795
55	-1.1796	1.1198
56	1.4025	-3.1866

5.10.2 PCA: Cross-Validation Model

```
# reproducible random sampling
set.seed(my_seed)

# Cross-validation with n folds
k_10 <- trainControl(method = "cv", number = 10)

# training the model
model_cv <- train(expected_survival ~ .,
                  data = train_pca,
                  method = "lm",
                  trControl = k_10)

# Print Model Performance
print(model_cv)
```

```
Linear Regression

39 samples
 2 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 35, 35, 35, 35, 35, 35, ...
Resampling results:

  RMSE     Rsquared   MAE
  21.7825  0.9761582  16.14872

Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
# Metrics
cv_results = model_cv$results
kbl(cv_results, caption = "PCA: Cross-Validation Metrics",
    digits = 4) %>%
  kable_paper("hover", full_width = F)
```

Table 5.11: PCA: Cross-Validation Metrics

intercept	RMSE	Rsquared	MAE	RMSESD	RsquaredSD	MAESD
TRUE	21.7825	0.9762	16.1487	14.0372	0.0377	8.5694

5.11 Predictions

```
# Find the index position of the target feature
pred_target_index <- grep(target_name,
                      colnames(test_pca))
#cat("Target Feature Index =", pred_target_index)

# Create Predicted Target Feature (y-test)
y_test <- test_pca[pred_target_index]
```

```
# Predictions using the Cross-Validation model
y_pred = predict(model_cv, newdata = test_pca[, -pred_target_index])
```

```
# Prediction Results from y_predictions
y_pred <- round(y_pred, digits = 0)
```

```
# Transform y_test from data frame to numeric
y_test <- as.numeric(unlist(y_test))

prediction_comparison <- cbind(y_pred, y_test)
# table format
kbl(prediction_comparison) %>%
  kable_paper("hover", full_width = F)
```

5.11.1 Prediction Metrics

```
# Calculate Mean Absolute Error (MAE)
MAE_value <- mae(y_pred, y_test)
#cat("MAE =", mae_value)

# Calculate MSE
MSE_predict <- mean((y_pred - y_test)^2)
#cat("\nMSE =", mse_predict)
```

	y_pred	y_test
9	35	16
11	520	442
12	310	318
16	26	25
17	298	284
18	125	152
21	159	160
23	149	141
25	191	197
26	81	95
28	2	2
40	86	78
43	113	38
47	182	171
48	585	657
52	33	7
53	88	92

MAE_value	MSE_predict	RMSE_predict	predicted_R2
21.8824	1142.235	33.797	0.96

```
# Calculate RMSE
RMSE_predict <- sqrt(mean((y_pred - y_test)^2))
#cat("\nRMSE =", rmse_predict)

# Calculate R-squared (R^2)
predicted_R2 <- 1 - sum((y_test - y_pred)^2) /
  sum((y_test - mean(y_test))^2)
# cat("\nPredicted R^2 =", predicted_r2)

prediction_metrics_df <- cbind(MAE_value, MSE_predict,
                               RMSE_predict, predicted_R2)
# table format
kbl(prediction_metrics_df, digits = 4) %>%
  kable_paper("hover", full_width = F)
```

5.12 Training Conclusion

In conclusion, this project has demonstrated the effectiveness of Principal Component Analysis (PCA) in dimension reduction with the following key points:

- PCA was able to reduce from 37 features down to just 2 principal components.
- The best score of R^2 = 97.61% was from the Linear Regression with Cross-validation model.
- The predicted R^2 = 96%
- The average deviation between the predicted values, and observed values for 'Expected Survival' is RMSE = 33.8.
- The model has not been exposed to unseen data with a large amount of observations to asses its robustness, and reliability.

6 Results

The Dialysis dataset consists of 55 observations of 39 variables with 1 discrete variable (States/Territories) and 38 continuous variables. For the principal component analysis, 37 of the 38 continuous variables were used; the variable expected_survival was excluded from PCA in order to use it as a target for model building in the latter part of the analysis.

Principal component analysis was performed using a singular value decomposition approach. Among the resulting principal components, the first PC captures 40.80% of the variance in the data, and the first two principal components capture 50.27% of the variance. The first four PCs capture 67.66% of the variance, or just over two-thirds; after the fourth PC, the variance captured by each successive PC begins to diminish relative to PCs one through four. The first ten PCs capture 88.67% of the variance, and in terms of dimensionality reduction over 90% of the information in the dataset can be explained by only 11 PCs when compared to the original 38 continuous variables.

The variables which contribute the most to PC1 are expected_hospital_readmission, expected_transfusion, and expected_hospitalization, although the percent of total contribution to PC1 by any one variable is not outsized relative to the remaining variables. PC2, which is orthogonal to PC1, has relatively large contributions from the five variables measuring levels of phosphorus; patterns or trends in the data such as these can be further explored using other methodologies after being highlighted in PCA [26].

From here, the analysis extends PCA via model building using linear regression with a technique known as principal component regression, with expected_survival used as the response variable. The data is first split into training and testing sets; then the data is centered and scaled, after which PCA is applied to each set. The first model is created for illustrative purposes, using 8 principle components from the training set of 39 observations. The estimates and significance of each PC regressor demonstrates the differences between variance captured from the data and usefulness in a linear model; for example, PC4 is a significant regressor despite capturing less variance than PC3 in the training data.

The analysis concludes by building and comparing two linear regression models using the first two prinicipal components. The first model is a straightforward linear model using the lm() function from the stats package in R [27]. The second model uses 10-fold cross-validation for a linear model using the train() and trainControl() functions from the caret package in R [28]. Both models produce an R^2 above 96% and a predicted R^2 above 95% with a 1% advantage on the cross-validation model. Although the interpretations of the regressors for these PCA models is different than those of a linear regression on the original

variables, the lower dimensionality of the data may be desirable as a more simple model which still captures a large portion of the variance in the original data.

7 Discussion

Principal Component Analysis (PCA) is a foundational multivariate analysis technique that has been widely employed to extract essential information from intricate multivariate datasets and effectively reduce dimensionality. Due to its simplicity and versatility, PCA has become one of the widely adopted tools for understanding and exploring data features in a multivariate dataset. This approach leverages singular value decomposition to restructure datasets and facilitates subsequent statistical analysis. By doing so, PCA simplifies complexity, eliminating superfluous details and redundant information arising from the original dataset. The outcome is a set of principal components that capture most of the variance explaining the original variables. To accomplish this, PCA first converts the dataset into a covariance matrix and then employs singular value decomposition to identify eigenvalues and eigenvectors, representing the loadings of the newly generated principal components. These components can typically account for over 70-80% of the original variables' variances.

PCA's importance in data analysis is underscored by its adaptability to various scenarios and data types, including binary, ordinal, compositional, and discrete data. Moreover, the PCA algorithm has proven effective in reducing the dimensions of vast datasets with high accuracy, significantly improving classification tasks. It plays a crucial role in exploratory data analysis and preliminary data processing, acting as a feature extraction and dimensionality reduction tool. One of its main benefits is its ability to mitigate multicollinearity issues, which can otherwise lead to biased results in statistical analyses.

Despite its numerous advantages, PCA does come with certain limitations. For instance, it is sensitive to the presence of outliers, which can distort the results and compromise its effectiveness. Furthermore, the new features or components generated through PCA are not readily interpretable, making it challenging to explain their meaning in a straightforward manner. Nonetheless, PCA remains a pivotal technique in data analysis, offering a powerful means to navigate complex datasets and uncover their underlying structures.

8 References

[1] M. Ringnér, "What is principal component analysis?" *Nature biotechnology*, vol. 26, no. 3, pp. 303–304, 2008.

[2] I. T. Jolliffe and J. Cadima, "Principal component analysis: A review and recent developments," *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, p. 20150202, 2016.

[3] D. Esposito and F. Esposito, *Introducing machine learning*. Microsoft Press, 2020.

[4] B. M. S. Hasan and A. M. Abdulazeez, "A review of principal component analysis algorithm for dimensionality reduction," *Journal of Soft Computing and Data Mining*, vol. 2, no. 1, pp. 20–30, 2021.

[5] K. Pearson, "LIII. On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, vol. 2, no. 11, pp. 559–572, 1901.

[6] H. Hotelling, "Analysis of a complex of statistical variables into principal components." *Journal of educational psychology*, vol. 24, no. 6, p. 417, 1933.

[7] R. A. Fisher and W. A. Mackenzie, "Studies in crop variation. II. The manurial response of different potato varieties," *The Journal of Agricultural Science*, vol. 13, no. 3, pp. 311–320, 1923.

[8] M. Greenacre, P. J. Groenen, T. Hastie, A. I. d'Enza, A. Markos, and E. Tuzhilina, "Principal component analysis," *Nature Reviews Methods Primers*, vol. 2, no. 1, p. 100, 2022.

[9] B. Everitt and T. Hothorn, *An introduction to applied multivariate analysis with r*. Springer Science & Business Media, 2011.

[10] H. Abdi and L. J. Williams, "Principal component analysis," *WIREs Computational Statistics*, vol. 2, no. 4, pp. 433–459, 2010, doi: https://doi.org/10.1002/wics.101. Available: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wics.101

[11] J. Lever, M. Krzywinski, and N. Altman, "Points of significance: Principal component analysis," *Nature methods*, vol. 14, no. 7, pp. 641–643, 2017.

[12] J. Maindonald and J. Braun, *Data analysis and graphics using r: An example-based approach*, vol. 10. Cambridge University Press, 2006.

[13] F. L. Gewers *et al.*, "Principal component analysis: A natural approach to data exploration," *ACM Computing Surveys (CSUR)*, vol. 54, no. 4, pp. 1–34, 2021.

[14] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[15] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[16] S. Zhang and M. Turk, "Eigenfaces," *Scholarpedia*, vol. 3, no. 9, p. 4244, 2008.

[17] R Core Team, "Prcomp, a function of r: A language and environment for statistical computing." R Foundation for Statistical Computing, Vienna, Austria, 2023. Available: https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/prcomp. [Accessed: Oct. 16, 2023]

[18] S. R. Bennett, "Linear algebra for data science." 2021. Available: https://shainarace.github.io/LinearAlgebra/index.html. [Accessed: Oct. 16, 2023]

[19] J. Hopcroft and R. Kannan, *Foundations of data science*. 2014.

[20] D. G. Luenberger, *Optimization by vector space methods*. John Wiley & Sons, 1997.

[21] E. K. CS, "PCA problem / how to compute principal components / KTU machine learning." YouTube, 2020. Available: https://youtu.be/MLaJbA82nzk. [Accessed: Nov. 01, 2023]

[22] S. Nash Warwick and W. Ford, "Abalone." UCI Machine Learning Repository, 1995.

[23] J. Pagès, *Multiple factor analysis by example using r*. CRC Press, 2014.

[24] "Quarterly dialysis facility care compare (QDFCC) report: July 2023." Centers for Medicare & Medicaid Services (CMS). Available: https://data.cms.gov/provider-data/dataset/2fpu-cgbb. [Accessed: Oct. 11, 2023]

[25] F. Chumney, "PCA, EFA, CFA," pp. 2–3, 6, Sep., 2012, Available: https://www.westga.edu/academics/research/vrc/assets/docs/PCA-EFA-CFA_EssayChumney_09282012.pdf

[26] R. Bro and A. K. Smilde, "Principal component analysis," *Analytical methods*, vol. 6, no. 9, pp. 2812–2831, 2014.

[27] R Core Team, "Lm: Fitting linear models." R Foundation for Statistical Computing, Vienna, Austria, 2023. Available: https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/lm. [Accessed: Nov. 08, 2023]

[28] Kuhn and Max, "Building predictive models in r using the caret package," *Journal of Statistical Software*, vol. 28, no. 5, pp. 1–26, 2008, doi: 10.18637/jss.v028.i05. Available: https://www.jstatsoft.org/index.php/jss/article/view/v028i05

Presentation