Advanced Data Analysis

Assignment - 2

Mrunali Vikas Patil
2024-02-04

PROBLEM 1

A) Why do we use regularized regressions? Give examples of when you would use ridge versus lasso regressions?

Regularized regressions are used to prevent overfitting, enhance the prediction accuracy, and handle multicollinearity in the data. They do this by introducing a penalty term to the loss function used to train the regression model. This penalty term shrinks the coefficients of less important features towards zero, effectively performing feature selection or reducing the impact of less important features.

- Ridge Regression: This technique is used when we have a dataset with multicollinearity among the features, where several features are highly correlated. In ridge regression, the penalty term is the square of the magnitude of the coefficients. This type of regularization shrinks the coefficients evenly but does not reduce them to zero, which means it does not perform feature selection but reduces overfitting. It's particularly useful when we have more predictors than observations.
- Lasso Regression: Lasso is used when we aim to enhance the interpretability of the model by reducing the number of features. The penalty term in lasso regression is the absolute value of the magnitude of the coefficients. This can reduce the coefficients of less important features to zero, effectively removing them from the model. This is useful in scenarios where feature selection is crucial, or we have a high-dimensional dataset with many features not all of which are important for predicting the target variable.

B) How would we treat overfitting in a model with too many variables compared to the sample size?

To treat overfitting in a model with too many variables, especially when the number of variables is comparable to or exceeds the sample size, we can:

- Use Regularization Techniques: As mentioned earlier, ridge and lasso regressions are effective in dealing with overfitting by introducing a penalty term that constrains the size of the coefficients.

- Feature Selection: Manually or using algorithms to select a subset of the most important features, thereby reducing the dimensionality of the problem.
- Cross-Validation: Use cross-validation techniques to assess the model's performance on unseen data and ensure that the model is not just memorizing the training data.
- Pruning: In the context of decision trees, pruning back the branches of the tree after it has been fully grown can reduce overfitting.
- Reducing Model Complexity: Simplify the model by choosing a simpler model with fewer parameters.

C) How do we check the assumptions of linear regression in R? Give an example for how each assumption may be violated.

In R, various diagnostic plots and tests can be used to check the assumptions of linear regression, which are:

- Linearity: The relationship between the predictors and the target variable is linear. This can be checked using scatter plots of the residuals versus fitted values or the predictors versus the response. Violation example: A quadratic or exponential relationship between the predictors and the response.
- Homoscedasticity: The variance of the error terms is constant across all levels of the independent variables. This can be examined using a residual versus fitted values plot. Violation example: The residuals fan out or form a funnel shape as the fitted values increase.
- Independence of Errors: The residuals are independent of each other. This can be assessed using a Durbin-Watson test in R, which checks for autocorrelation in the residuals. Violation example: In time series data, where the residuals could be correlated with each other.
- Normality of Residuals: The residuals of the model are normally distributed. This can be checked using a Q-Q plot (quantile-quantile plot) or a Shapiro-Wilk test. Violation example: The Q-Q plot deviates significantly from a straight line.
- No Multicollinearity: The independent variables are not too highly correlated. This can be checked using Variance Inflation Factors (VIF). Violation example: Two or more variables are linear combinations of each other, leading to inflated standard errors.

PROBLEM 3

The study titled "Development and psychometric testing of hybrid education competence instrument" focused on creating and validating a tool to assess educators' competencies in hybrid teaching environments in health and social care fields. It conducted exploratory factor analysis to discern five key competencies from survey responses, which were subsequently tested for internal consistency. This analysis resulted in a reliable instrument that can aid in both self-assessment for educators and broader educational design, addressing a crucial need in contemporary hybrid teaching landscapes.

- Application of Factor Analysis: The authors applied Exploratory Factor Analysis (EFA) to identify underlying factors from 46 items that measure hybrid education competencies. EFA is used to detect the structure among the variables in the questionnaire, allowing the authors to understand which items group together.
- Type of Factor Rotation: They used Promax rotation, an oblique rotation method that allows the factors to be correlated. This is appropriate for their study as they expected the factors of hybrid teaching competencies to be interrelated.
- Number of Factors: They concentrated on five factors, determined through EFA. They arrived at this number by considering eigenvalues greater than 1, the scree plot, and the interpretability of the factors. These five factors explained 70.83% of the total variance in their data.
- Breakdown and Significance of Factor Names: Each factor represents a cluster of items that measure a specific domain of hybrid teaching competence. For example, one factor might represent 'Planning and Resourcing' while another might capture 'Technological Competence'. The names reflect the conceptual content of the items within each factor.
- Evaluating Stability of Components (Factorability): The authors evaluated factorability by verifying sampling adequacy using the Kaiser-Meyer-Olkin measure and ensuring the suitability of the correlation matrix for factor analysis using Bartlett's test of sphericity.
- Use of Factors in Later Analysis: Yes, the factors like planning lessons, using technology, interacting with students, teaching online skills, and being fair were important for teachers in hybrid classes. They used these factors to understand how good they were at hybrid teaching. They found out that they needed to look at each factor more closely to really understand how well they were doing. Overall, the study showed that it's important for teachers to be good at all these things to teach hybrid classes well.
- Overall Conclusions from Factor Analysis: Factor Analysis allowed the authors to identify and validate the key dimensions of hybrid education competence, ensuring that the instrument they developed is both reliable and meaningful for assessing educators' competencies in hybrid teaching environments.

PROBLEM 4

Load necessary libraries.

```
library(readr)
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(ggplot2)
library(factoextra)

## Welcome! Want to learn more? See two factoextra-related books at
https://goo.gl/ve3WBa

library(psych)

##
## Attaching package: 'psych'

## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha

library(GGally)

## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2

library(corrplot)

## corrplot 0.92 loaded

library(FactoMineR)
```

Read the dataset

```
data <- read.csv("/Users/mrunalipatil/Downloads/BIG5.csv")
```

Show Structure of Dataset

```
str(data)
```

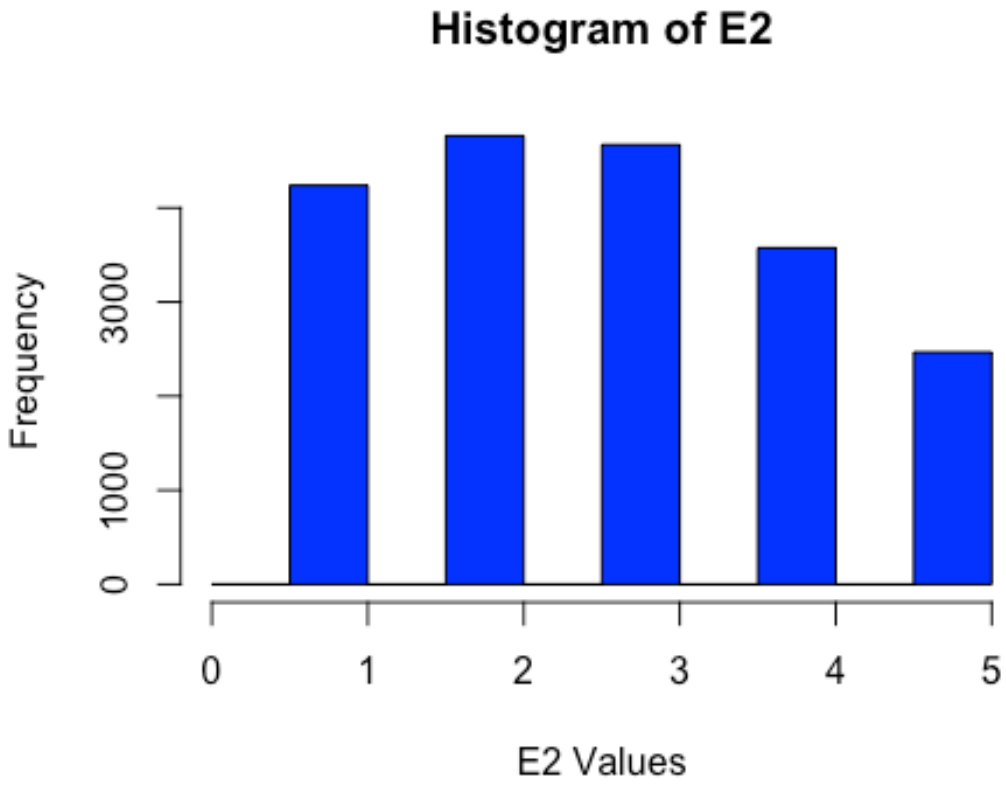
```
## 'data.frame':    19719 obs. of  50 variables:
##  $ E1 : int  4 2 5 2 3 1 5 4 3 1 ...
##  $ E2 : int  2 2 1 5 1 5 1 3 1 4 ...
##  $ E3 : int  5 3 1 2 3 2 5 5 5 2 ...
##  $ E4 : int  2 3 4 4 3 4 1 3 1 5 ...
##  $ E5 : int  5 3 5 3 3 1 5 5 5 2 ...
##  $ E6 : int  1 3 1 4 1 3 1 1 1 4 ...
##  $ E7 : int  4 1 1 3 3 2 5 4 5 1 ...
##  $ E8 : int  3 5 5 4 1 4 4 3 2 4 ...
##  $ E9 : int  5 1 5 4 3 1 4 4 5 1 ...
##  $ E10: int  1 5 1 5 5 5 1 3 3 5 ...
##  $ N1 : int  1 2 5 5 3 1 2 1 2 5 ...
##  $ N2 : int  5 3 1 4 3 5 4 4 4 2 ...
##  $ N3 : int  2 4 5 4 3 4 2 4 5 5 ...
##  $ N4 : int  5 2 5 2 4 5 4 4 3 2 ...
##  $ N5 : int  1 3 5 4 3 1 2 1 3 3 ...
##  $ N6 : int  1 4 5 5 3 4 2 1 5 4 ...
##  $ N7 : int  1 3 5 5 3 4 3 1 5 3 ...
##  $ N8 : int  1 2 5 5 3 1 2 1 4 2 ...
##  $ N9 : int  1 2 5 4 3 5 2 1 3 3 ...
##  $ N10: int  1 4 5 5 4 2 2 1 3 4 ...
##  $ A1 : int  1 1 5 2 5 2 5 2 1 2 ...
##  $ A2 : int  5 3 1 5 5 2 5 5 5 3 ...
##  $ A3 : int  1 3 5 4 3 3 1 1 1 1 ...
##  $ A4 : int  5 4 5 4 5 4 5 4 5 4 ...
##  $ A5 : int  2 4 1 3 1 3 1 3 1 2 ...
##  $ A6 : int  3 4 5 5 5 4 5 3 5 4 ...
##  $ A7 : int  1 2 1 3 1 3 1 1 1 3 ...
##  $ A8 : int  5 3 5 4 5 5 5 3 5 3 ...
##  $ A9 : int  4 4 5 4 5 5 4 4 5 3 ...
##  $ A10: int  5 3 5 3 5 3 5 5 4 2 ...
##  $ C1 : int  4 4 4 3 3 2 2 4 4 5 ...
##  $ C2 : int  1 1 1 3 1 5 4 2 3 2 ...
##  $ C3 : int  5 3 5 4 5 4 3 5 5 4 ...
##  $ C4 : int  1 2 1 5 3 3 3 1 2 2 ...
##  $ C5 : int  5 3 5 1 3 3 3 4 5 3 ...
##  $ C6 : int  1 1 1 4 1 4 3 1 2 2 ...
##  $ C7 : int  4 5 5 5 1 5 3 4 5 4 ...
##  $ C8 : int  1 1 1 4 3 3 3 1 2 2 ...
##  $ C9 : int  4 4 5 2 3 5 3 3 4 4 ...
##  $ C10: int  5 4 5 3 3 3 3 5 3 4 ...
##  $ O1 : int  4 3 4 4 3 4 3 3 3 4 ...
##  $ O2 : int  1 3 5 3 1 2 1 1 3 2 ...
##  $ O3 : int  3 3 5 5 1 1 5 5 5 5 ...
##  $ O4 : int  1 3 1 2 1 3 1 1 3 2 ...
##  $ O5 : int  5 2 5 4 3 3 4 4 5 4 ...
##  $ O6 : int  1 3 1 2 1 5 1 1 1 1 ...
##  $ O7 : int  4 3 5 5 3 5 4 5 5 4 ...
##  $ O8 : int  2 1 5 2 1 4 3 3 3 3 ...
```

```
##  $ O9 : int  5 3 5 5 5 5 3 2 4 4 ...
##  $ O10: int  5 2 5 5 3 3 4 5 5 4 ...
```

```
summary(data$E1)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   2.000   3.000   2.629   4.000   5.000
```

```
hist(data$E2, main = "Histogram of E2", xlab = "E2 Values", col = "blue")
```



Show Column Names

```
names(data)
```

```
##  [1] "E1"  "E2"  "E3"  "E4"  "E5"  "E6"  "E7"  "E8"  "E9"  "E10" "N1"
"N2"
## [13] "N3"  "N4"  "N5"  "N6"  "N7"  "N8"  "N9"  "N10" "A1"  "A2"  "A3"
"A4"
## [25] "A5"  "A6"  "A7"  "A8"  "A9"  "A10" "C1"  "C2"  "C3"  "C4"  "C5"
"C6"
## [37] "C7"  "C8"  "C9"  "C10" "O1"  "O2"  "O3"  "O4"  "O5"  "O6"  "O7"
"O8"
## [49] "O9"  "O10"
```

Check for Missing Values across all variables

```
sum(is.na(data))
```

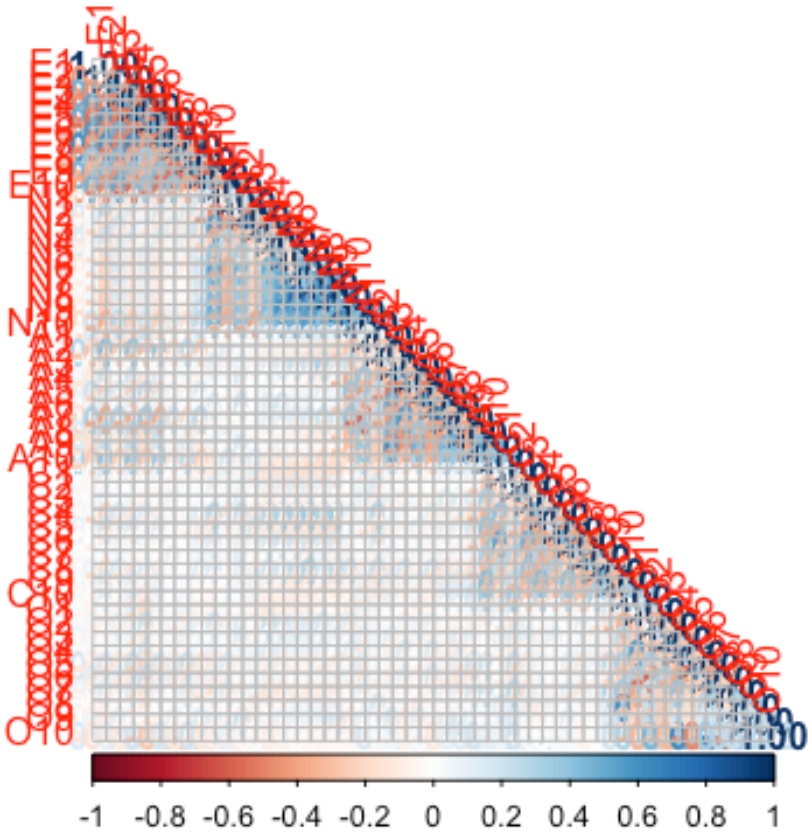
```
## [1] 0
```

Calculating the correlation matrix using Spearman's method

```
M <- cor(data, method = "spearman")
```

Plotting the correlation matrix

```
corrplot(M, method = "number", type = "lower")
```



Saving the correlation plot as a PNG file

```
png("correlation_plot.png")
corrplot(M, method = "number", type = "lower")
dev.off()
```

```
## quartz_off_screen
##                 2
```

Running a correlation test

```
MCorrTest <- corr.test(data, method = "spearman", adjust = "none")
```

Extracting the p-values from the correlation test

```
M <- MCorrTest$p
```

Determining which correlations are significant at the 0.01 level

```
MTest <- ifelse(M < .01, TRUE, FALSE)
```

Counting the number of significant correlations for each variable

```
significant_correlations <- colSums(MTest) - 1 # Subtracting 1 for the
diagonal elements
```

Printing the number of significant correlations for each variable

```
print(significant_correlations)
```

```
##  E1  E2  E3  E4  E5  E6  E7  E8  E9 E10  N1  N2  N3  N4  N5  N6  N7  N8
N9 N10
##  45  47  48  46  48  47  46  40  44  44  45  45  47  42  47  46  47  49
48  46
##  A1  A2  A3  A4  A5  A6  A7  A8  A9 A10  C1  C2  C3  C4  C5  C6  C7  C8
C9 C10
##  44  46  44  44  44  39  47  46  44  49  44  37  46  48  47  39  41  49
45  47
##  O1  O2  O3  O4  O5  O6  O7  O8  O9 O10
##  45  46  45  44  47  45  46  41  48  46
```

PCA Plot functions

```
pca_result <- prcomp(data, center = TRUE, scale. = TRUE)
```

Function to plot PCA loadings for the first two principal components

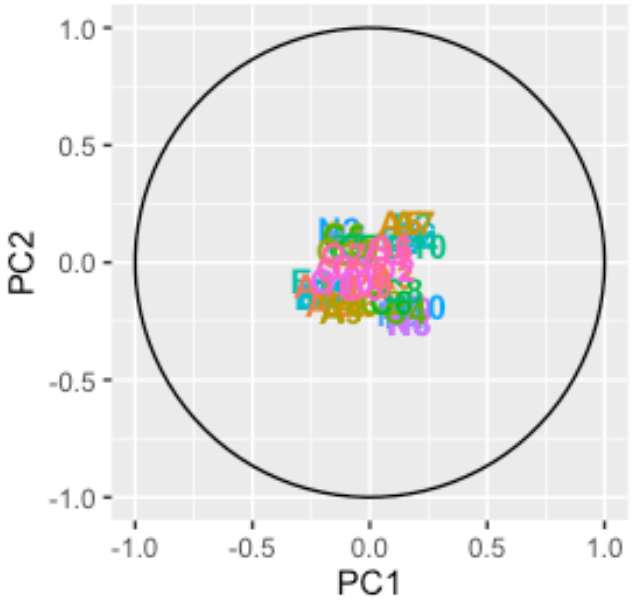
```
PCA_Plot <- function(pcaData) {
  theta <- seq(0, 2*pi, length.out = 100)
  circle <- data.frame(x = cos(theta), y = sin(theta))
  p <- ggplot(circle, aes(x, y)) + geom_path()

  loadings <- data.frame(pcaData$rotation, .names =
row.names(pcaData$rotation))
  p + geom_text(data = loadings, mapping = aes(x = PC1, y = PC2, label =
.names, colour = .names, fontface = "bold")) +
    coord_fixed(ratio = 1) + labs(x = "PC1", y = "PC2")
}
```

Function to plot PCA loadings for the third and fourth principal components

```
PCA_Plot_Secondary <- function(pcaData) {
  theta <- seq(0, 2*pi, length.out = 100)
  circle <- data.frame(x = cos(theta), y = sin(theta))
  p <- ggplot(circle, aes(x, y)) + geom_path()

  loadings <- data.frame(pcaData$rotation, .names =
row.names(pcaData$rotation))
  p + geom_text(data = loadings, mapping = aes(x = PC3, y = PC4, label =
.names, colour = .names, fontface = "bold")) +
    coord_fixed(ratio = 1) + labs(x = "PC3", y = "PC4")
}
```

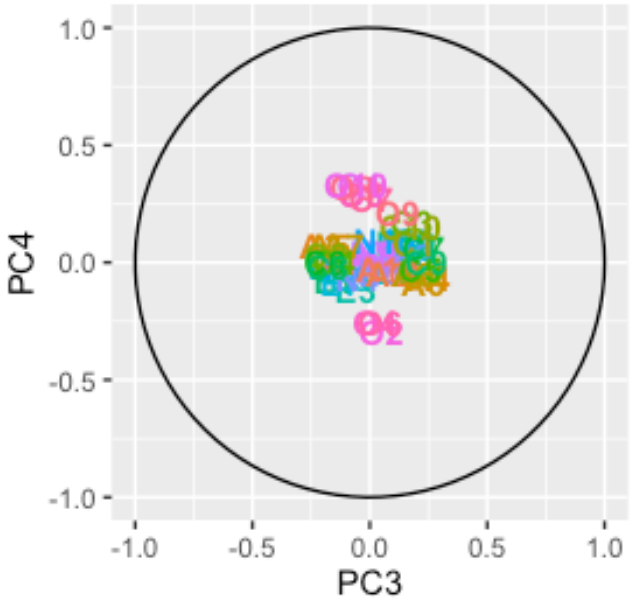
Call the plot functions with the PCA results

```
PCA_Plot(pca_result)
```



a	A1	a	C7	a	N4
a	A10	a	C8	a	N5
a	A2	a	C9	a	N6
a	A3	a	E1	a	N7
a	A4	a	E10	a	N8
a	A5	a	E2	a	N9
a	A6	a	E3	a	O1
a	A7	a	E4	a	O10
a	A8	a	E5	a	O2
a	A9	a	E6	a	O3
a	C1	a	E7	a	O4
a	C10	a	E8	a	O5
a	C2	a	E9	a	O6
a	C3	a	N1	a	O7
a	C4	a	N10	a	O8
a	C5	a	N2	a	O9
a	C6	a	N3		

```
PCA_Plot_Secondary(pca_result)
```



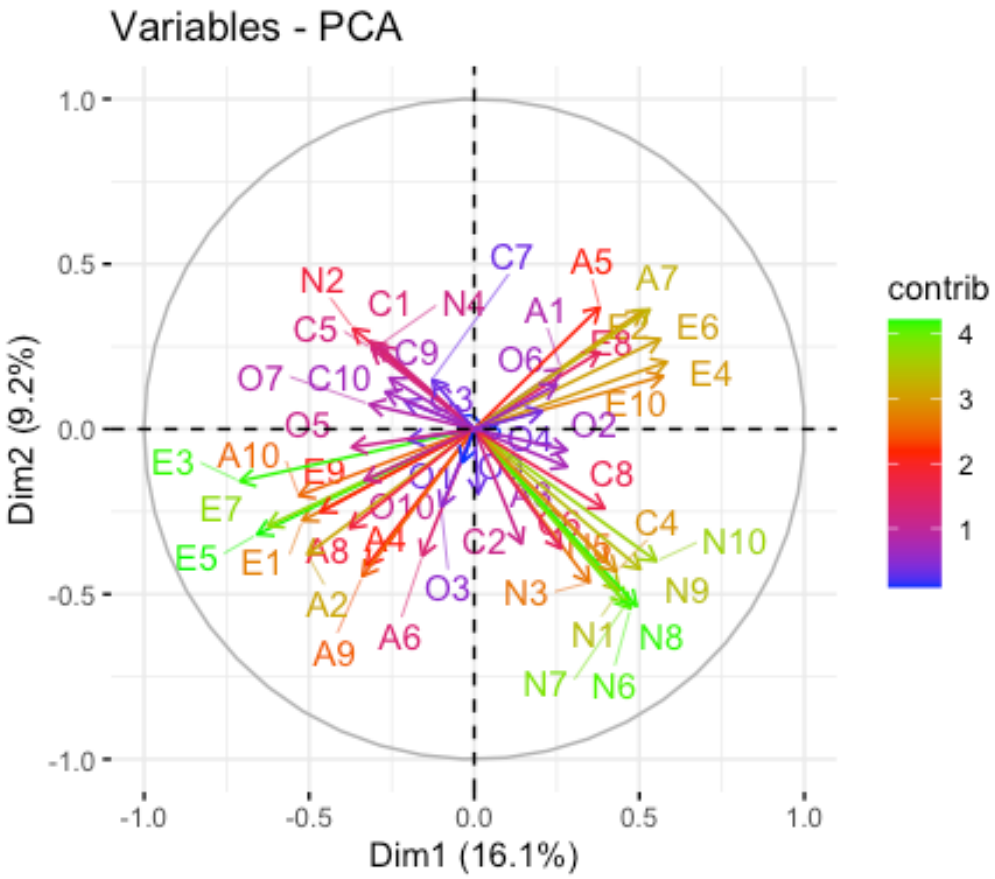
a	A10	a	C8	a	N5
a	A2	a	C9	a	N6
a	A3	a	E1	a	N7
a	A4	a	E10	a	N8
a	A5	a	E2	a	N9
a	A6	a	E3	a	O1
a	A7	a	E4	a	O10
a	A8	a	E5	a	O2
a	A9	a	E6	a	O3
a	C1	a	E7	a	O4
a	C10	a	E8	a	O5
a	C2	a	E9	a	O6
a	C3	a	N1	a	O7
a	C4	a	N10	a	O8
a	C5	a	N2	a	O9
a	C6	a	N3		

Create a biplot of variable contributions

```
biplot <- fviz_pca_var(pca_result, col.var = "contrib", gradient.cols =
c("blue", "red", "green"), repel = TRUE)
```

Print or plot the biplot

```
print(biplot)
```



Kaiser-Meyer-Olkin (KMO)

Test for Sampling Adequacy

```
kmo_results <- KMO(data)
print(kmo_results)
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = data)
## Overall MSA =  0.91
## MSA for each item =
##   E1   E2   E3   E4   E5   E6   E7   E8   E9  E10   N1   N2   N3   N4   N5
N6
## 0.94 0.93 0.96 0.95 0.95 0.94 0.94 0.90 0.92 0.95 0.93 0.91 0.92 0.90 0.96
0.93
##   N7   N8   N9  N10   A1   A2   A3   A4   A5   A6   A7   A8   A9  A10   C1
C2
## 0.87 0.87 0.93 0.94 0.91 0.94 0.90 0.89 0.92 0.90 0.91 0.95 0.90 0.96 0.92
0.86
##   C3   C4   C5   C6   C7   C8   C9  C10   O1   O2   O3   O4   O5   O6   O7
O8
## 0.90 0.94 0.91 0.88 0.89 0.94 0.89 0.91 0.77 0.84 0.81 0.81 0.86 0.83 0.91
0.75
```

```
##   O9  O10
## 0.90 0.85
```

The Kaiser-Meyer-Olkin (KMO) test results indicate an overall measure of sampling adequacy (MSA) of 0.91, which suggests that the dataset is suitable for factor analysis. Individual MSA values for each item range from 0.75 to 0.96, with most items showing high adequacy, thereby confirming that the data structure is appropriate for identifying underlying factors.

Bartlett's Test of Sphericity

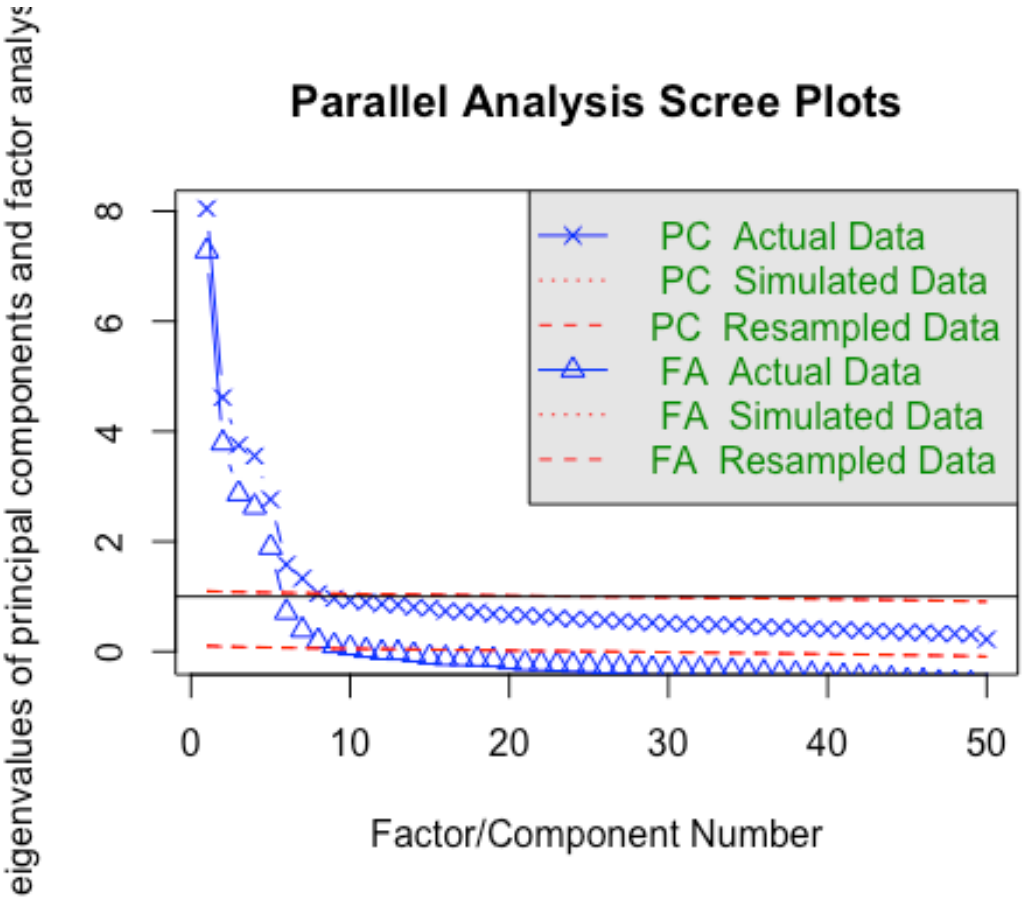
```
library(REdaS)

## Loading required package: grid

bart_spher(data)

##  Bartlett's Test of Sphericity
##
## Call: bart_spher(x = data)
##
##      X2 = 376850.453
##      df = 1225
## p-value < 2.22e-16
```

Bartlett's Test of Sphericity checks whether the observed variables in dataset are uncorrelated and hence, the identity matrix is an appropriate covariance matrix for the data. The extremely small p-value (< 2.22e-16) indicates that there is significant evidence to reject the null hypothesis of the identity covariance matrix, suggesting that the variables are related and therefore suitable for factor analysis or PCA.

Parallel Analysis

Determines the number of factors/components.

```
parallel_results <- fa.parallel(data, fa="both")
```




```
## Parallel analysis suggests that the number of factors =  10  and the
number of components =  7

print(parallel_results)

## Call: fa.parallel(x = data, fa = "both")
## Parallel analysis suggests that the number of factors =  10  and the
number of components =  7
##
##  Eigen Values of
##    Original factors Resampled data Simulated data Original components
## 1              7.28           0.10           0.10                 8.05
## 2              3.79           0.09           0.09                 4.62
## 3              2.87           0.08           0.08                 3.75
## 4              2.63           0.08           0.08                 3.55
## 5              1.89           0.07           0.07                 2.76
## 6              0.72           0.07           0.07                 1.58
## 7              0.39           0.06           0.06                 1.33
## 8              0.19           0.06           0.06                 1.05
## 9              0.10           0.06           0.06                 0.97
## 10             0.06           0.05           0.05                 0.93
##    Resampled components Simulated components
## 1                  1.10                 1.10
```

```
## 2                   1.09                   1.09
## 3                   1.08                   1.08
## 4                   1.08                   1.08
## 5                   1.07                   1.07
## 6                   1.07                   1.07
## 7                   1.06                   1.06
## 8                   1.06                   1.06
## 9                   1.05                   1.05
## 10                  1.05                   1.05
```

The plot shown visualizes the eigenvalues of principal components and factor analysis for both the actual data and data obtained through resampling and simulation. The plot suggests choosing 7 components for PCA (as their eigenvalues are above the eigenvalues of the corresponding simulated data) and 10 factors for factor analysis based on the same criterion.

QUESTION A]

Load necessary libraries.

```
library(readr)
library(ggplot2)
library(factoextra)
```

Read in the dataset

```
data <- read.csv("/Users/mrunalipatil/Downloads/BIG5.csv")
```

Checking missing values

```
missing_values <- sum(is.na(data))
print(paste("There are", missing_values, "missing values in the dataset."))
```

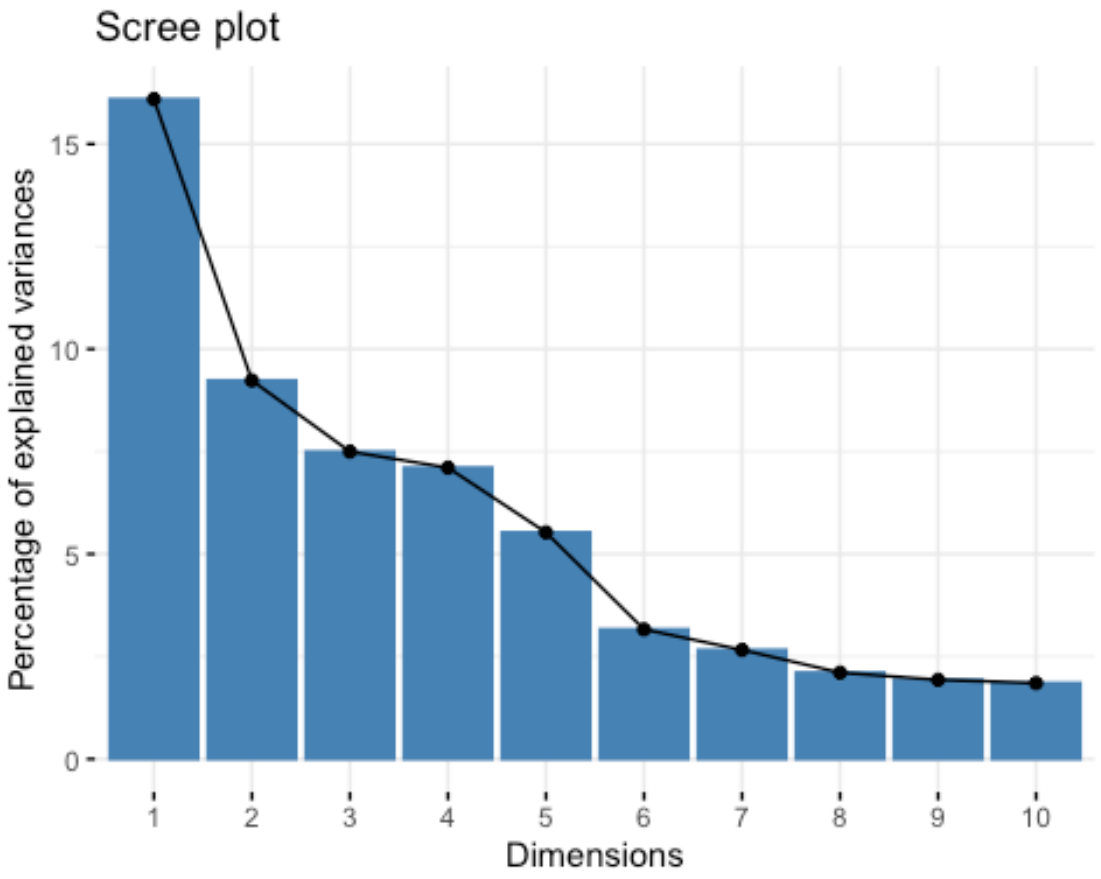
```
## [1] "There are 0 missing values in the dataset."
```

Performing PCA assuming no missing values

```
pca_results <- prcomp(data, scale. = TRUE)
```

Create a scree plot to visualize the variance explained by each principal component

```
fviz_eig(pca_results)
```



Determine the number of components needed to explain 100% of the variation

```
total_components <- ncol(data)
print(paste("To explain 100% of the variation, we need", total_components,
"components."))
```

```
## [1] "To explain 100% of the variation, we need 50 components."
```

Determine the number of components to use based on cumulative variance explained

```
explained_variance <- summary(pca_results)$importance[2,]
cumulative_variance <- cumsum(explained_variance)
num_components_to_use <- which(cumulative_variance >= 0.95)[1]  # for 95%
variance explained
print(paste("Number of components to use for 95% variance explained:",
num_components_to_use))
```

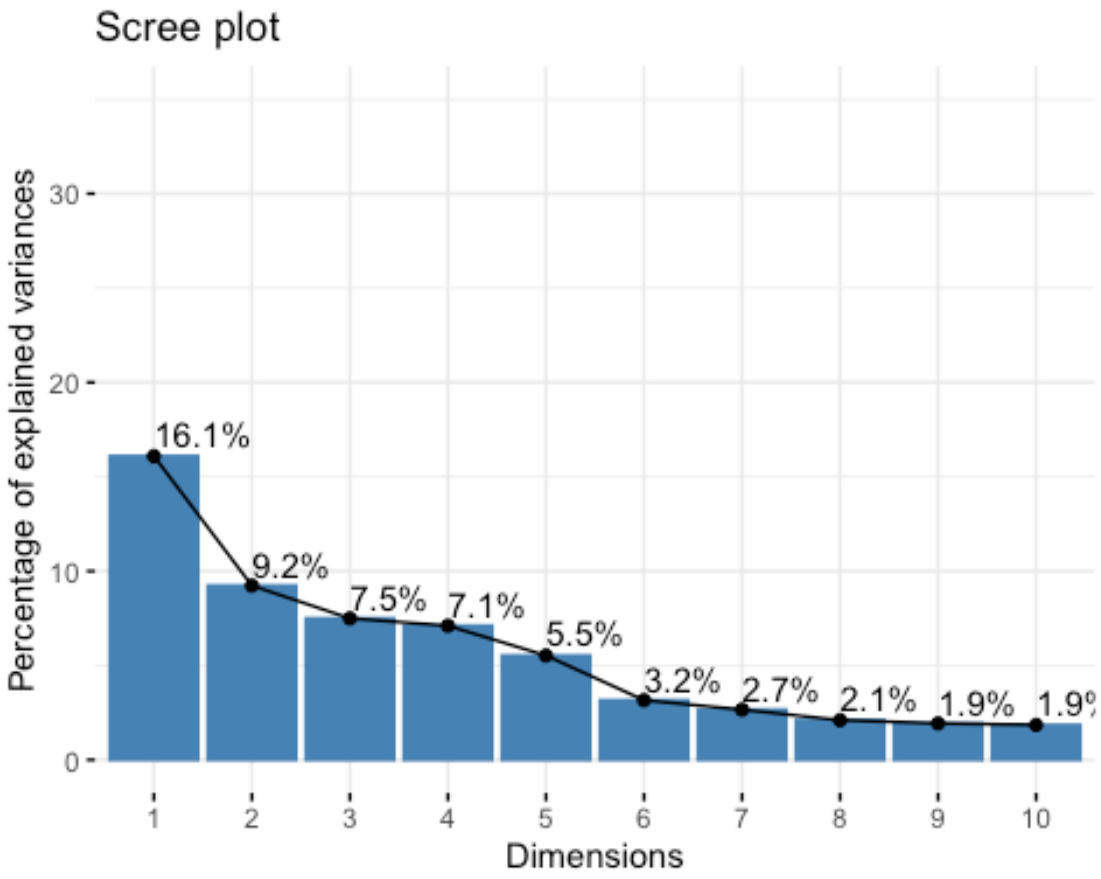
```
## [1] "Number of components to use for 95% variance explained: 43"
```

Number of Components for 100% Variation: To explain 100% of the total variation in the data, all the original variables are needed. Since there are 50 variables in the dataset, we need 50 components to explain 100% of the variation.

The number of components is determined by looking for the "elbow" in the scree plot, which is the point where the explained variance levels off and additional components do not contribute significantly to explaining more variance. In the plot, the "elbow" appears to be around the 5th component, as the drop in the percentage of explained variance becomes less steep after this point hence the number of components to use for 95% variance explained is 43.

Creating another scree plot to visualize the variance explained by each principal component

```
pca_result <- prcomp(data, scale. = TRUE)
data <- PCA(data, graph = FALSE)

screePlot <- fviz_screeplot(data, addlabels = TRUE, ylim = c(0,35))
screePlot
```



Components Determined from Scree Plot: Based on the second scree plot provided, which shows a more gradual drop-off in explained variance after the seventh component, it

suggests that 7 components are sufficient to capture the most significant variance in the dataset without including all 50 components.

Number of Components to Use in the Model: Although 50 components would explain all the variation, it's practical to use fewer to simplify the model and focus on the most informative aspects of the data. The second scree plot indicates that using 7 components strikes a balance between capturing a substantial amount of the variance and maintaining model parsimony. Therefore, 7 components would be used in the model.

QUESTION B]

Load the necessary library

```
library(psych)
library(FactoMineR)
library(factoextra)
```

Perform PCA and apply a varimax rotation with 7 factors

```
data <- read.csv("/Users/mrunalipatil/Downloads/BIG5.csv")
pca_results <- principal(data, rotate = "varimax", nfactors = 7, scores =
TRUE)
```

Perform PCA with Varimax rotation using the principal() function

```
pca_res_rotated <- principal(data, nfactors = 7, rotate = "varimax", scores =
TRUE)
```

Print the rotated loadings with a cutoff of 0.4, sorting the loadings for easier interpretation

```
print(pca_res_rotated$loadings[,1], cutoff = 0.4, sort = TRUE)
```

```
##          E1          E2          E3          E4          E5          E6
##  0.69067347 -0.73468725  0.65491483 -0.75430890  0.74103372 -0.63479671
##          E7          E8          E9         E10          N1          N2
##  0.74400079 -0.62794924  0.65290608 -0.69843892 -0.10186070  0.06870283
##          N3          N4          N5          N6          N7          N8
## -0.14057811  0.11310589 -0.06110445 -0.07324613 -0.01919875 -0.03212181
##          N9         N10          A1          A2          A3          A4
## -0.05323870 -0.24746303 -0.04284674  0.35257635  0.11934795  0.02416537
##          A5          A6          A7          A8          A9         A10
## -0.15636455 -0.03746727 -0.33776486  0.10568785  0.09506841  0.32110112
##          C1          C2          C3          C4          C5          C6
##  0.03729276  0.03575751 -0.05546603 -0.07972766  0.07561215 -0.02595174
##          C7          C8          C9         C10          O1          O2
```

```
## -0.04667736 -0.08034564  0.05326864  0.01744965  0.05813011 -0.04538369
##          O3          O4          O5          O6          O7          O8
##  0.01310030 -0.00934949  0.19745652 -0.09825913  0.07140325  0.02219380
##          O9         O10
## -0.16325462  0.18185098
```

The formula for the first component, RC1, is a linear combination of your original variables (E1 to O10), each weighted by its respective loading on RC1.

RC1 = 0.6907 * E1 - 0.7347 * E2 + 0.6549 * E3 - 0.7543 * E4 + 0.7410 * E5 - 0.6348 * E6 + 0.7440 * E7 - 0.6279 * E8 + 0.6529 * E9 - 0.6984 * E10 - 0.1019 * N1 + 0.0687 * N2 - 0.1406 * N3 + 0.1131 * N4 - 0.0611 * N5 - 0.0732 * N6 - 0.0192 * N7 - 0.0321 * N8 - 0.0532 * N9 - 0.2475 * N10 - 0.0428 * A1 + 0.3526 * A2 + 0.1193 * A3 + 0.0242 * A4 - 0.1564 * A5 - 0.0375 * A6 - 0.3378 * A7 + 0.1057 * A8 + 0.0951 * A9 + 0.3211 * A10 + 0.0373 * C1 + 0.0358 * C2 - 0.0555 * C3 - 0.0797 * C4 + 0.0756 * C5 - 0.0260 * C6 - 0.0467 * C7 - 0.0803 * C8 + 0.0533 * C9 + 0.0174 * C10 + 0.0581 * O1 - 0.0454 * O2 + 0.0131 * O3 - 0.0093 * O4 + 0.1975 * O5 - 0.0983 * O6 + 0.0714 * O7 + 0.0222 * O8 - 0.1633 * O9 + 0.1819 * O10

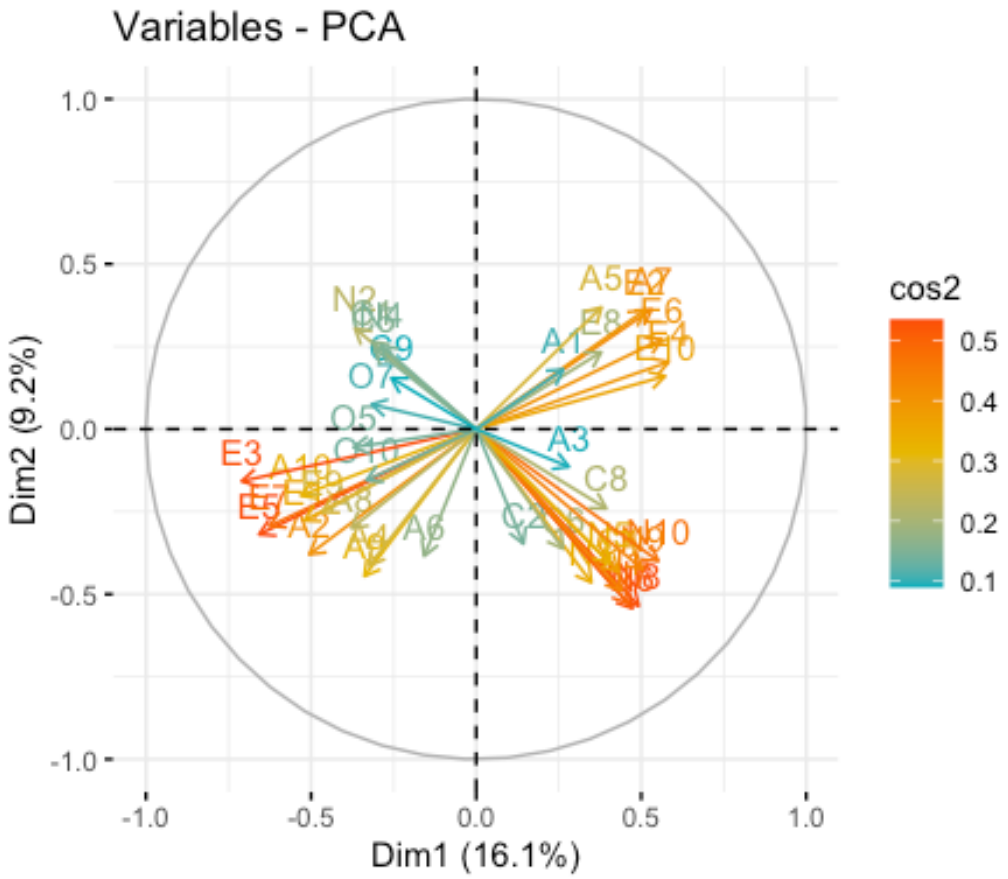
Interpretation of the First Component (RC1)
The first component is characterized by strong positive loadings on several “E” variables (E1, E3, E5, E7, E9) and strong negative loadings on others (E2, E4, E6, E8, E10). These variables are related to the “Extraversion” personality trait, suggesting that RC1 captures aspects of extraversion and introversion.

Specifically:
Positive Loadings (e.g., E1, E3, E5, E7, E9): These suggest that higher scores on RC1 are associated with higher extraversion traits, such as being sociable, energetic, and enthusiastic.

Negative Loadings (e.g., E2, E4, E6, E8, E10): These indicate that lower scores on RC1 are associated with traits opposite to extraversion, which can be interpreted as introversion traits such as being reserved, quiet, and solitary.

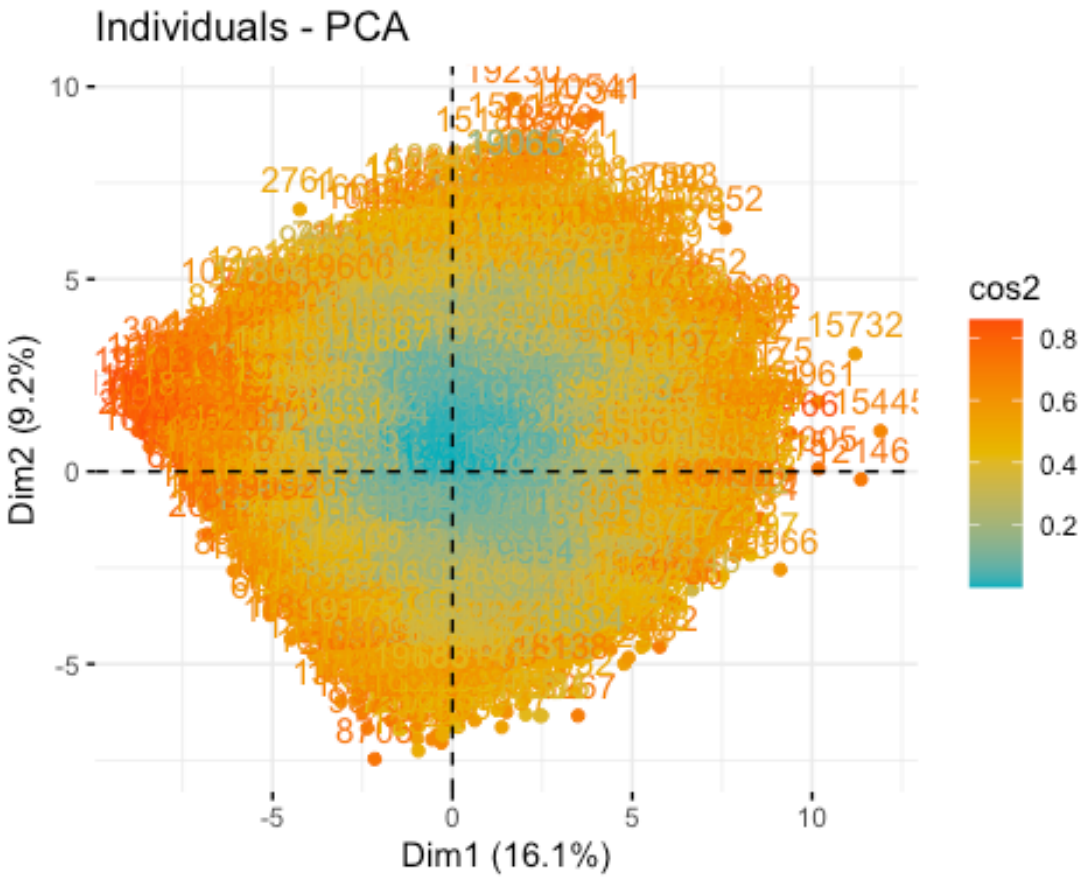
PCA Variables

```r
fviz_pca_var(pca_result, col.var = "cos2",
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             select.var = list(contrib = 40))  # Use 40 instead of 0.4
```



PCA Individuals

```
fviz_pca_ind(pca_result,col.ind = "cos2",
                 gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
                 select.var = list(contrib = 40))
```



Naming Components:

RC1 ("Extraversion"): High positive loadings on extraversion-related items suggest this component captures outgoing, energetic, and sociable traits.

RC2 ("Neuroticism"): Dominated by "N" items, representing emotional instability and anxiety.

RC3 ("Agreeableness"): High loadings on "A" items, capturing cooperative, friendly, and compassionate traits.

RC5 ("Conscientiousness"): Strong "C" item loadings, related to being organized, dependable, and disciplined.

RC4 & RC7 ("Openness to Experience"): Mixed loadings with emphasis on "O" items, suggesting creativity and openness.

RC6 ("Emotional Openness/Reactivity"): A mix of emotional and openness traits.

QUESTION C]

Perform PCA and apply varimax rotation.

```
pca_results <- principal(data, rotate = "varimax", nfactors = 7, scores =
TRUE)
```

Extract the PCA scores

```
scores <- pca_results$scores
```

Initialize lists to store the identifiers of subjects with the highest and lowest scores

```
highest_scorers <- list()
lowest_scorers <- list()
```

Loop through each component to find the subjects with the highest and lowest scores

```
for (i in 1:7) {
  highest_score <- max(scores[, i], na.rm = TRUE)
  lowest_score <- min(scores[, i], na.rm = TRUE)

  # Assuming subjects are rows in the original dataset, their identifiers are
row numbers
  highest_scorers[[i]] <- which(scores[, i] == highest_score)
  lowest_scorers[[i]] <- which(scores[, i] == lowest_score)

  # Print out the subject identifiers and their scores for each component
  cat(sprintf("Component %d:\n", i))
  cat("Highest Scorer: Subject", highest_scorers[[i]], "with a score of",
highest_score, "\n")
  cat("Lowest Scorer: Subject", lowest_scorers[[i]], "with a score of",
lowest_score, "\n\n")
}
```

```
## Component 1:
## Highest Scorer: Subject 4177 with a score of 2.894111
## Lowest Scorer: Subject 629 with a score of -2.842958
##
## Component 2:
## Highest Scorer: Subject 12089 with a score of 2.748238
## Lowest Scorer: Subject 19065 with a score of -3.578815
##
## Component 3:
```

```
## Highest Scorer: Subject 17760 with a score of 2.337412
## Lowest Scorer: Subject 9864 with a score of -4.405143
##
## Component 4:
## Highest Scorer: Subject 16225 with a score of 2.837716
## Lowest Scorer: Subject 15237 with a score of -3.443489
##
## Component 5:
## Highest Scorer: Subject 445 with a score of 2.929062
## Lowest Scorer: Subject 2464 with a score of -4.678958
##
## Component 6:
## Highest Scorer: Subject 13095 with a score of 3.96649
## Lowest Scorer: Subject 19065 with a score of -5.702946
##
## Component 7:
## Highest Scorer: Subject 2794 3329 7128 8210 8224 8850 11343 13094 16592
17686 19654 with a score of 7.151124
## Lowest Scorer: Subject 19065 with a score of -10.57518
```

Principal Components Score are as follows:

Defining the indices for the highest and lowest scorers for each component

```
highest_scorers_indices <- c(4177, 12089, 17760, 16225, 445, 13095, c(2794,
3329, 7128, 8210, 8224, 8850, 11343, 13094, 16592, 17686, 19654))
lowest_scorers_indices <- c(629, 19065, 9864, 15237, 2464, 19065, 19065)
```

Retrieve scores for the highest and lowest scorers

```
highest_scorers_scores <- scores[highest_scorers_indices, ]
lowest_scorers_scores <- scores[lowest_scorers_indices, ]
```

Print the scores with the indices

```
cat("PCA scores for the highest scorers:\n")

## PCA scores for the highest scorers:

for (i in seq_along(highest_scorers_indices)) {
  cat("Subject", highest_scorers_indices[i], "Scores:",
highest_scorers_scores[i, ], "\n")
}

## Subject 4177 Scores: 2.894111 1.38598 -4.264436 -0.2924879 -0.07657847
1.069631 2.818563
## Subject 12089 Scores: 1.490764 2.748238 0.1697074 0.8189701 -0.1498882 -
2.769093 0.355551
## Subject 17760 Scores: -0.5719914 -1.04946 2.337412 -1.936645 -2.819714 -
0.3372327 1.341597
```

```
## Subject 16225 Scores: 2.702665 1.90507 -3.501147 2.837716 0.4461282 -
2.561388 1.783662
## Subject 445 Scores: -1.287302 1.370156 -0.333341 -0.7433972 2.929062 -
0.6602222 3.400583
## Subject 13095 Scores: 0.4635365 -0.4261685 0.3100429 0.00929272 -3.308426
3.96649 0.50021
## Subject 2794 Scores: -0.070383 2.106021 0.191434 0.6398735 -1.130667
2.681237 7.151124
## Subject 3329 Scores: -0.070383 2.106021 0.191434 0.6398735 -1.130667
2.681237 7.151124
## Subject 7128 Scores: -0.070383 2.106021 0.191434 0.6398735 -1.130667
2.681237 7.151124
## Subject 8210 Scores: -0.070383 2.106021 0.191434 0.6398735 -1.130667
2.681237 7.151124
## Subject 8224 Scores: -0.070383 2.106021 0.191434 0.6398735 -1.130667
2.681237 7.151124
## Subject 8850 Scores: -0.070383 2.106021 0.191434 0.6398735 -1.130667
2.681237 7.151124
## Subject 11343 Scores: -0.070383 2.106021 0.191434 0.6398735 -1.130667
2.681237 7.151124
## Subject 13094 Scores: -0.070383 2.106021 0.191434 0.6398735 -1.130667
2.681237 7.151124
## Subject 16592 Scores: -0.070383 2.106021 0.191434 0.6398735 -1.130667
2.681237 7.151124
## Subject 17686 Scores: -0.070383 2.106021 0.191434 0.6398735 -1.130667
2.681237 7.151124
## Subject 19654 Scores: -0.070383 2.106021 0.191434 0.6398735 -1.130667
2.681237 7.151124
```

```
cat("\nPCA scores for the lowest scorers:\n")
```

```
##
## PCA scores for the lowest scorers:
```

```
for (i in seq_along(lowest_scorers_indices)) {
  cat("Subject", lowest_scorers_indices[i], "Scores:",
lowest_scorers_scores[i, ], "\n")
}
```

```
## Subject 629 Scores: -2.842958 -0.6000083 -0.2405723 2.224126 1.438778 -
0.5558442 -0.7393839
## Subject 19065 Scores: 0.8366825 -3.578815 -3.396086 -2.060702 -1.686245 -
5.702946 -10.57518
## Subject 9864 Scores: 1.346395 1.963654 -4.405143 0.8812595 0.638137
1.369522 -0.247384
## Subject 15237 Scores: 1.252928 -2.126797 1.086574 -3.443489 -0.04847647 -
0.6002504 -2.070395
## Subject 2464 Scores: -1.165016 -0.03180801 2.260464 -0.4780488 -4.678958 -
0.5726348 0.3745229
## Subject 19065 Scores: 0.8366825 -3.578815 -3.396086 -2.060702 -1.686245 -
5.702946 -10.57518
```

```
## Subject 19065 Scores: 0.8366825 -3.578815 -3.396086 -2.060702 -1.686245 -
5.702946 -10.57518
```

QUESTION D]

Perform factor analysis with 7 factors.

```
fa_results <- fa(data, nfactors = 7, fm = "minres", rotate = "varimax")
```

Print the factor loadings

```
print(fa_results$loadings, cutoff = 0.3, sort = TRUE)
```

```
##
## Loadings:
##     MR1    MR2    MR3    MR5    MR4    MR7    MR6
## E1   0.657
## E2  -0.695
## E3   0.641
## E4  -0.721
## E5   0.722
## E6  -0.593
## E7   0.722
## E8  -0.566
## E9   0.607
## E10 -0.656
## N1          0.693
## N2         -0.534                              0.302
## N3          0.615
## N5          0.532
## N6          0.739
## N7          0.699
## N8          0.733
## N9          0.702
## N10         0.616
## A2   0.342         0.537
## A4                 0.784
## A5                -0.651
## A6                 0.595
## A7  -0.328        -0.621
## A8                 0.588
## A9                 0.697
## C1                        0.596
## C2                       -0.539
## C4          0.352        -0.564
## C5                         0.619
## C6                        -0.602
## C7                         0.534
```

```
## C9                        0.619
## O3                              0.616
## O5                              0.565
## O6                             -0.646
## O10                             0.687
## O1                                    0.676
## O8                                    0.671
## N4         -0.339
## A1                -0.429
## A3                -0.388
## A10  0.319         0.394
## C3                       0.405
## C8                      -0.478
## C10                      0.472
## O2                             -0.461
## O4                             -0.433
## O7                              0.350  0.342
## O9
##
##                  MR1   MR2   MR3   MR5   MR4   MR7   MR6
## SS loadings    5.008 4.623 3.776 3.308 2.508 1.560 0.987
## Proportion Var 0.100 0.092 0.076 0.066 0.050 0.031 0.020
## Cumulative Var 0.100 0.193 0.268 0.334 0.384 0.416 0.435
```

Factor analysis seeks to discover underlying factors explaining correlations in personality test responses, potentially offering a construct-focused interpretation that may differ from PCA's broader variance-based approach.

1)**Factor Loadings vs. PCA Loadings:** In PCA, each component is a linear combination of all the variables, where the first principal component explains the most variance, followed by the second, and so on. In factor analysis, factors represent underlying latent constructs that are presumed to cause the observed correlations among variables. The loadings indicate how strongly each measured variable is associated with each factor.

2)**Interpretation of Loadings:** PCA loadings are typically interpreted in terms of variance explained, with higher loadings indicating that a variable contributes more to the component. Factor analysis loadings are interpreted in terms of latent construct representation. A high loading of a variable on a factor suggests that the variable is a good measure of the underlying construct that the factor represents.

3)**Differences Found:** The factor analysis loadings you've provided show a clearer pattern of variables grouping together, which can correspond to the underlying latent constructs of personality traits. The variance explained by each factor (SS loadings) and the cumulative variance indicate the proportion of the total variance accounted for by the factors. Unlike PCA, factor analysis does not necessarily account for all the variance, focusing instead on common variance shared by variables.

4)**Practical Interpretation**: Factor analysis can potentially offer a more nuanced interpretation of the data, focusing on theoretical constructs rather than just variance

explained. The factors extracted can be directly named after constructs they represent (like Extraversion, Neuroticism, etc.), making them potentially more interpretable and meaningful in the context of personality assessment.

5)**Naming the Factors:** MR1 could represent Extraversion, given it has high loadings on the extraversion items (E1, E3, E5, E7, E9). MR2 appears to represent Neuroticism, with high loadings on neuroticism items (N1, N3, N6-N9). Subsequent factors (MR3-MR7) would need to be interpreted based on the items that load highly on them, potentially representing other dimensions of personality or more specific facets of the Big Five traits.

In summary, factor analysis can refine the interpretation by focusing on common variance and underlying constructs, whereas PCA is more of a data reduction method that considers total variance. This can make factor analysis particularly useful in psychological testing where the theoretical underpinnings and construct validity are crucial.