Experiment -1

1. **For the given data, analyze the missing values and comment on imputing or dropping the missing values. Explain how the missing values will affect the model estimation and its goodness of fit.**
2. **Influence of variance in large data set with many variables. Higher the variance of the variable will contribute the higher importance to the data set. Find the variance of method to reduce the dimension of the data set. Give justification of output.**
3. **Dimensionality reduction using PCA (Principal Component Analysis) and Feature Selection through OMP (Orthogonal Matching Pursuit)**

Aim: To deal with a dataset having missing values and analyze how the missing values will affect the model estimation and its goodness of fit.

Description:

- Create a dummy dataset.
- Use different aspects of missing data imputation
- Find out variances of all the variables and by including and excluding different variables, observe the effects.
- Apply PCA (Principal Component Analysis) and Feature Selection through OMP (Orthogonal Matching Pursuit).

Algorithm:

- Missing data imputation
- Calculate variances of all the variables.
- Apply PCA (Principal Component Analysis) and Feature Selection through OMP (Orthogonal Matching Pursuit) on the given dataset.

Program:

```
####test for missing values
#vector with missing values

 x<-c(1:4,NA,6:7,NA)
x

#test the missing values in x vector

is.na(x)

#data frame with missing values

df<-data.frame(col1=c(10:12,NA),
               col2=c("I",NA,"am","fine"),
               col3=c(TRUE,FALSE,TRUE,FALSE),
               col4=c(1.2,2.3,NA,4.3),
               stringsAsFactors=FALSE)
df

#IDENTIFY THE MISSING VALUES IN THE DATA FRAME
is.na(df)

#identify the location of NA's in the vector

which(is.na(x))

#identify the location of NA's in the data frame
which(is.na(df))

#display the sum of NA's in vector
```

```
sum(is.na(x))

#display the sum of NA's in the data frame
sum(is.na(df))

#display the count of NA's in the each column of data frame
colSums(is.na(df))

#in two ways missing values are coded i.e., NA and 99
#recode missing values with mean
#before recode x is
x
mean(x)
mean(x,na.rm=TRUE)
x[is.na(x)]<-mean(x,na.rm=TRUE)



#after recode x is
x
round(x,2)

# data frame with missing values coded as 99
df1<-data.frame(col1=c(1:3,99),col2=c(2.8,4.7,99,4.2))
df1



#change 99 to NA
df1[df1==99]<-NA
df1
```

```
#Exclude missing values
x1<-c(1:3,NA,4:5)
x1

#display mean of x1
mean(x1)

#mean of x1 after excluding NA
mean(x1,na.rm=TRUE)

#DATA FRAME WITH MISSING VALUES
df

#list of complete rows of the data frame
complete.cases(df)

#subset with complete.cases to get complete cases
df[complete.cases(df),]

#subset with ! to get incomplete cases
df[!complete.cases(df),]

#short hand way to get the complete cases of data frame by omitting
na.omit(df)
```

Output:

```
> x

[1]  1  2  3  4 NA  6  7 NA

> is.na(x)

[1] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE
```

```
#data frame with missing values

> df

  col1 col2  col3 col4
1   10    I  TRUE  1.2
2   11 <NA> FALSE  2.3
3   12   am  TRUE   NA
4   NA fine FALSE  4.3

> is.na(df)

      col1  col2  col3  col4
[1,] FALSE FALSE FALSE FALSE
[2,] FALSE  TRUE FALSE FALSE
[3,] FALSE FALSE FALSE  TRUE
[4,]  TRUE FALSE FALSE FALSE

> which(is.na(x))

[1] 5 8

> sum(is.na(x))

[1] 2

> sum(is.na(df))

[1] 3

> colSums(is.na(df))

col1 col2 col3 col4
   1    1    0    1
```

#in two ways missing values are coded i.e., NA and 99

#recode missing values with mean

#before recode x is

```
> x

[1]  1  2  3  4 NA  6  7 NA

> mean(x)

[1] NA

> mean(x,na.rm=TRUE)

[1] 3.833333

> x[is.na(x)]<-mean(x,na.rm=TRUE)

> #after recode x is

> x

[1] 1.000000 2.000000 3.000000 4.000000 3.833333 6.000000 7.000000
[8] 3.833333

> round(x,2)

[1] 1.00 2.00 3.00 4.00 3.83 6.00 7.00 3.83

> # data frame with missing values coded as 99

> df1<-data.frame(col1=c(1:3,99),col2=c(2.8,4.7,99,4.2))

> df1

  col1 col2
1    1  2.8
2    2  4.7
3    3 99.0
4   99  4.2

> #change 99 to NA

> df1[df1==99]<-NA
> df1

  col1 col2
1    1  2.8
2    2  4.7
3    3   NA
4   NA  4.2

> #Exclude missing values

> x1<-c(1:3,NA,4:5)

> x1

[1]  1  2  3 NA  4  5

> #display mean of x1
> mean(x1)

[1] NA

> #mean of x1 after excluding NA

> mean(x1,na.rm=TRUE)

[1] 3

> #DATA FRAME WITH MISSING VALUES

> df

  col1 col2  col3 col4
1   10    I  TRUE  1.2
2   11 <NA> FALSE  2.3
3   12   am  TRUE   NA
4   NA fine FALSE  4.3

> #list of complete rows of the data frame

> complete.cases(df)

[1]  TRUE FALSE FALSE FALSE
```

```
> #subset with complete.cases to get complete cases

> df[complete.cases(df),]

  col1 col2 col3 col4
1   10    I TRUE  1.2
> #subset with ! to get incomplete cases

> df[!complete.cases(df),]

  col1 col2  col3 col4
2   11 <NA> FALSE  2.3
3   12   am  TRUE   NA
4   NA fine FALSE  4.3

> #short hand way to get the complete cases of data frame by omitting

> na.omit(df)

  col1 col2 col3 col4
1   10    I TRUE  1.2
```

Experiment -1 (2)

Influence of variance in large data set with many variables. Higher the variance of the variable will contribute the higher importance to the data set. Find the variance of method to reduce the dimension of the data set. Give justification of output.

Aim: To find out the influence of variance in large data set with many variables.

Description:

- Variance has a huge impact in many different aspects of your life.
- Variance measures how distant from the mean random values are in a data set.
- A set of data with low variance (relative) is dominated at the mean, and a set of high variance is spread out and deviates significantly from the mean.

- A high variance curve will be flat relative to a low variance curve.

Algorithm:

- Variance is mathematically defined as the expected value of the squared deviation from the mean. It can also be calculated from a data set by using the following equation:

$$\sigma^2 = \frac{\sum(\chi - \mu)^2}{N}$$

```
Corrmatrix <- structure(c(1, 0.82, 0.54, 0.36, 0.85, 0.82, 1, 0.01, 0.74, 0.36,
                          0.54, 0.01, 1, 0.65, 0.91, 0.36, 0.74, 0.65, 1, 0.36,
                          0.85, 0.36, 0.91, 0.36, 1),
                        .Dim = c(5L, 5L))

Corrmatrix

library(caret)
findCorrelation(Corrmatrix, cutoff = .6, verbose = TRUE, names = F)
```

Output:

```
> Corrmatrix

     [,1] [,2] [,3] [,4] [,5]
[1,] 1.00 0.82 0.54 0.36 0.85
[2,] 0.82 1.00 0.01 0.74 0.36
[3,] 0.54 0.01 1.00 0.65 0.91
[4,] 0.36 0.74 0.65 1.00 0.36
[5,] 0.85 0.36 0.91 0.36 1.00

> findCorrelation(Corrmatrix, cutoff = .6, verbose = TRUE, names = F)

Compare row 1  and column  5 with corr  0.85
```

```
  Means:  0.642 vs 0.545 so flagging column 1
Compare row 5  and column  3 with corr  0.91
  Means:  0.543 vs 0.499 so flagging column 5
Compare row 3  and column  4 with corr  0.65
  Means:  0.33 vs 0.352 so flagging column 4

All correlations <= 0.6

[1] 1 5 4
```

Justification:

If two variables having high covariance, one of them may be removed from the data since both of them will be equally contributing individually.

Experiment -1 (3)

Dimensionality reduction using PCA (Principal Component Analysis) and Feature Selection through OMP (Orthogonal Matching Pursuit).

Aim: Performing the dimensionality reduction operation using PCA (Principal Component Analysis) and Feature Selection through OMP (Orthogonal Matching Pursuit).

Implementing PCA:

Algorithm:

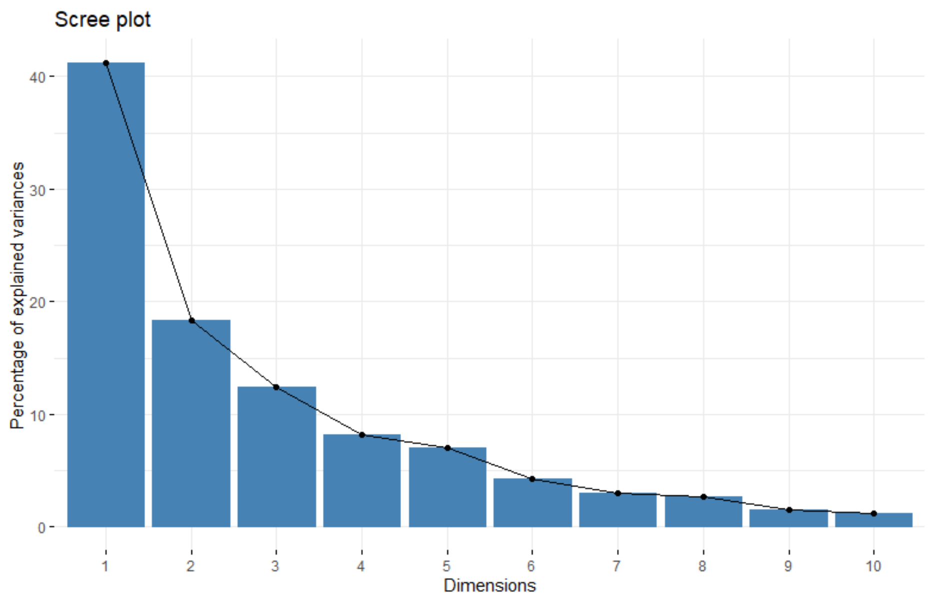
1. Load factoextra for visualization
2. Compute PCA
3. Visualize eigenvalues (scree plot). Show the percentage of variances explained by each principal component.
4. Graph of individuals. Individuals with a similar profile are grouped together.
5. Graph of variables. Positive correlated variables point to the same side of the plot. Negative correlated variables point to opposite sides of the graph.
6. Biplot of individuals and variables

```
library("factoextra")
data(decathlon2)
```

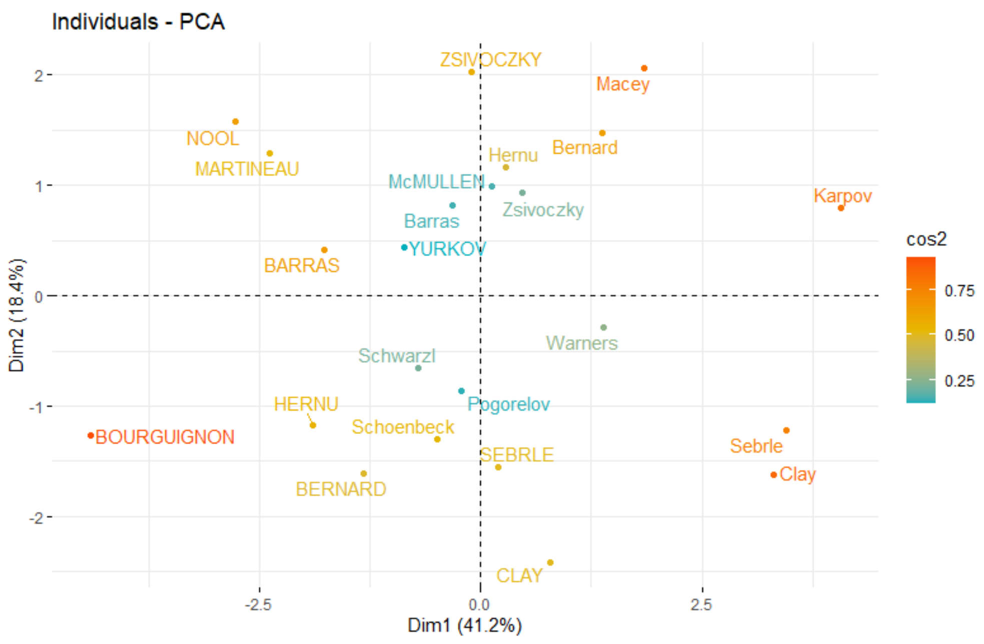
```
decathlon2.active <- decathlon2[1:23, 1:10]
head(decathlon2.active[, 1:6])

## Compute PCA
res.pca <- prcomp(decathlon2.active, scale = TRUE)

## Visualize eigenvalues (scree plot). Show the percentage of variances explained by each
principal component.
fviz_eig(res.pca)
# Graph of individuals. Individuals with a similar profile are grouped together.
Fviz_pca_ind(res.pca,
        col.ind = “cos2”, # Color by the quality of representation
        gradient.cols = c(“#00AFBB”, “#E7B800”, “#FC4E07”),
        repel = TRUE     # Avoid text overlapping
        )
## Graph of variables. Positive correlated variables point to the same side of the plot. Negative
correlated variables point to opposite sides of the graph.
fviz_pca_var(res.pca,
        col.var = "contrib", # Color by contributions to the PC
        gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
        repel = TRUE     # Avoid text overlapping
)

## Biplot of individuals and variables
fviz_pca_biplot(res.pca, repel = TRUE,
          col.var = "#2E9FDF", # Variables color
          col.ind = "#696969"  # Individuals color
)
```

Result:

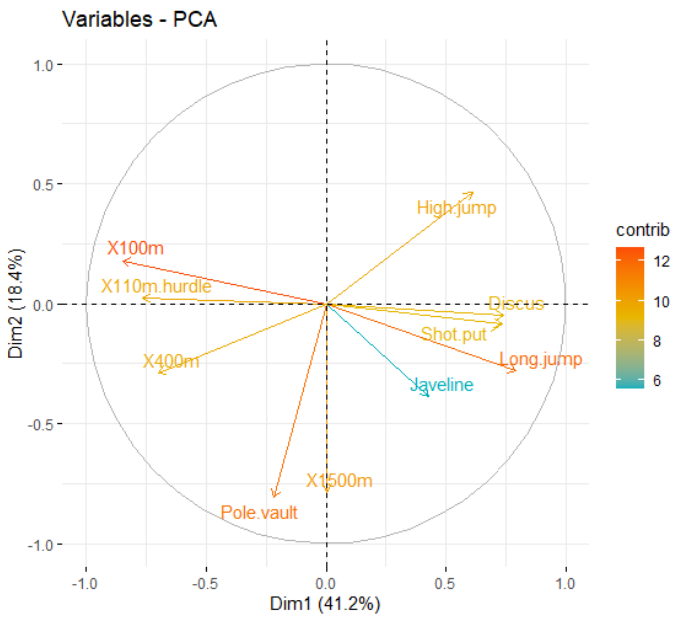
```
> head(decathlon2.active[, 1:6])

          X100m Long.jump Shot.put High.jump X400m X110m.hurdle
SEBRLE    11.04      7.58    14.83      2.07 49.81        14.69
CLAY      10.76      7.40    14.26      1.86 49.37        14.05
BERNARD   11.02      7.23    14.25      1.92 48.93        14.99
YURKOV    11.34      7.09    15.19      2.10 50.42        15.31
ZSIVOCZKY 11.13      7.30    13.48      2.01 48.62        14.17
McMULLEN  10.83      7.31    13.76      2.13 49.91        14.38

> ## Compute PCA

> res.pca <- prcomp(decathlon2.active, scale = TRUE)
```



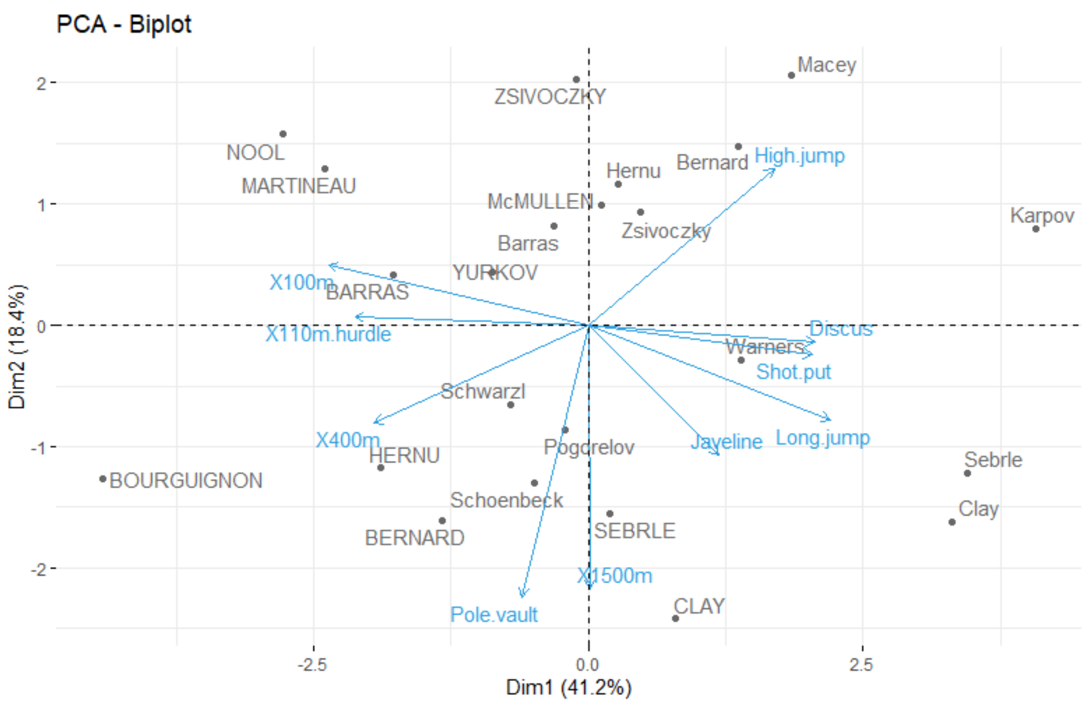
```
> ## Graph of individuals. Individuals with a similar profile are grouped
together.
```



## Graph of variables. Positive correlated variables point to the same side of the plot. Negative correlated variables point to opposite sides of the graph.



## Biplot of individuals and variables



Implementing Feature Selection through OMP (Orthogonal Matching Pursuit)