EXPERIMENT 3

Taniya Ahmed

21BDS0059

Q1. Write R programs to read data from keyboard and transform it to various ranges.

Code:

1. **Installing packages**

```
install.packages("dlookr")
library("dlookr")
install.packages("dplyr")
library("dplyr")
install.packages("tidyr")
library("tidyr")
install.packages("magrittr")
library("magrittr")
```

2. **Loading dataset, imputing missing values and summarising.**

```
df = read.csv("D:\\Downloads\\DS1_Diabetes1.csv")
df$SkinThickness[df$SkinThickness == -Inf] = NA
df$SkinThickness[is.na(df$SkinThickness)] = mean(df$SkinThickness, na.rm = TRUE)
head(df)
print("Taniya Ahmed 21BDS0059")
summary(df)
print("Taniya Ahmed 21BDS0059")
dim(df)
max(df$Insulin, na.rm = TRUE)
sum(is.na(df$Insulin))
print("Taniya Ahmed 21BDS0059")
```

```
> df = read.csv("D:\\Downloads\\DS1_Diabetes1.csv")
> df$SkinThickness[df$SkinThickness == -Inf] = NA
> df$SkinThickness[is.na(df$SkinThickness)] = mean(df$SkinThickness, na.rm = TRUE)
> head(df)
  Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI DiabetesPedigreeFunction Age Outcome
1           6     148            72            35       0 33.6                    0.627  50       1
2           1      85            66            29       0 26.6                    0.351  31       0
3           8     183            64             0       0 23.3                    0.672  32       1
4           1      89            66            23      94 28.1                    0.167  21       0
5           0     137            40            35     168 43.1                    2.288  33       1
6           5     116            74             0       0 25.6                    0.201  30       0
> print("Taniya Ahmed 21BDS0059")
[1] "Taniya Ahmed 21BDS0059"
> summary(df)
  Pregnancies        Glucose      BloodPressure    SkinThickness      Insulin           BMI
 Min.   : 0.000   Min.   : 71.0   Min.   :  0.00   Min.   : 0.00   Min.   :  0.0   Min.   : 0.0
 1st Qu.: 2.000   1st Qu.:103.0   1st Qu.: 66.00   1st Qu.: 0.00   1st Qu.:  0.0   1st Qu.:27.4
 Median : 5.000   Median :119.0   Median : 72.00   Median :23.00   Median :  0.0   Median :31.6
 Mean   : 5.388   Mean   :127.8   Mean   : 70.59   Mean   :19.45   Mean   : 83.2   Mean   :31.9
 3rd Qu.: 8.000   3rd Qu.:147.0   3rd Qu.: 82.00   3rd Qu.:33.00   3rd Qu.:115.0   3rd Qu.:37.6
 Max.   :13.000   Max.   :197.0   Max.   :110.00   Max.   :47.00   Max.   :846.0   Max.   :45.8
 DiabetesPedigreeFunction      Age           Outcome
 Min.   :0.1340           Min.   :21.00   Min.   :0.0000
 1st Qu.:0.2540           1st Qu.:29.00   1st Qu.:0.0000
 Median :0.4200           Median :33.00   Median :1.0000
 Mean   :0.5252           Mean   :37.55   Mean   :0.5102
 3rd Qu.:0.5870           3rd Qu.:48.00   3rd Qu.:1.0000
 Max.   :2.2880           Max.   :60.00   Max.   :1.0000
> print("Taniya Ahmed 21BDS0059")
[1] "Taniya Ahmed 21BDS0059"
```

3. Applying some statistical methods

```
subset_df = df[1 : 5, ]
print(subset_df)

max_insulin = max(df$Insulin)
print(max_insulin)

mean_skin_thick = mean(df$SkinThickness)
print(mean_skin_thick)

sum_preg = sum(df$Pregnancies)
print(sum_preg)

std_dev_diabetes_predig = sd(df$DiabetesPedigreeFunction)
print(std_dev_diabetes_predig)

grouped_blood_press = df %>% group_by(Age) %>% summarize(Avg_BP =
mean(BloodPressure))
print(head(grouped_blood_press, 5))
print("Taniya Ahmed 21BDS0059")
```

```
> subset_df = df[1 : 5, ]
> print(subset_df)
  Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI DiabetesPedigreeFunction Age Outcome
1           6     148            72            35       0 33.6                    0.627  50       1
2           1      85            66            29       0 26.6                    0.351  31       0
3           8     183            64             0       0 23.3                    0.672  32       1
4           1      89            66            23      94 28.1                    0.167  21       0
5           0     137            40            35     168 43.1                    2.288  33       1
>
> max_insulin = max(df$Insulin)
> print(max_insulin)
[1] 846
>
> mean_skin_thick = mean(df$SkinThickness)
> print(mean_skin_thick)
[1] 19.44898
>
> sum_preg = sum(df$Pregnancies)
> print(sum_preg)
[1] 264
>
> std_dev_diabetes_predig = sd(df$DiabetesPedigreeFunction)
> print(std_dev_diabetes_predig)
[1] 0.4316859
>
> grouped_blood_press = df %>% group_by(Age) %>% summarize(Avg_BP = mean(BloodPressure))
> print(head(grouped_blood_press, 5))
# A tibble: 5 x 2
    Age Avg_BP
  <int>  <dbl>
1    21   66
2    22   64.7
3    25   66
4    26   57
5    27   78
> print("Taniya Ahmed 21BDS0059")
[1] "Taniya Ahmed 21BDS0059"
```

4. Normalizing data in various ways

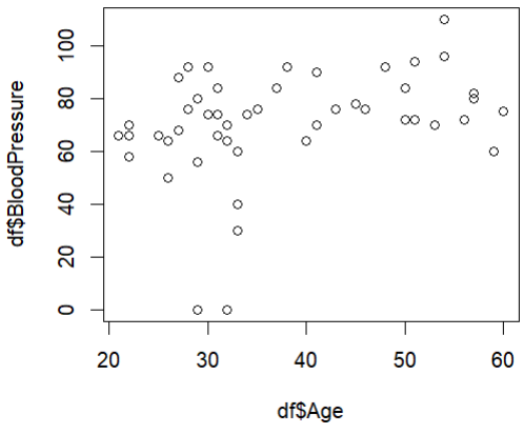
```
# normalizes to [0, 1]
# numerator shifts the min value to 0
# denominator scales data in the range of 0 to 1
#normalize_0_1 = function(x){
#  if((max(x) - min(x)) == 0){
#    return (0)
#  }
#  return (x - min(x)) / (max(x) - min(x))
#}

# [-1, 1] requires application of linear transformation
# numerator scales the range from [0, 1] to [0, 2]
# subtracting 1 shifts range from [0, 2] to [-1, 1]
normalize_1_1 = function(x){
 return (2 * ((x- min(x)) / (max(x) - min(x))) - 1)
}
# range helps in getting the max and min values
# numerator scales the range from [0, 1] to [0, 6]
# subtracting 3 shifts the scale to [-3, 3]
normalize_3_3 = function(x){
 r = range(x)
 return (6 * ((x - r[1]) / (r[2] - r[1])) - 3)
}
```

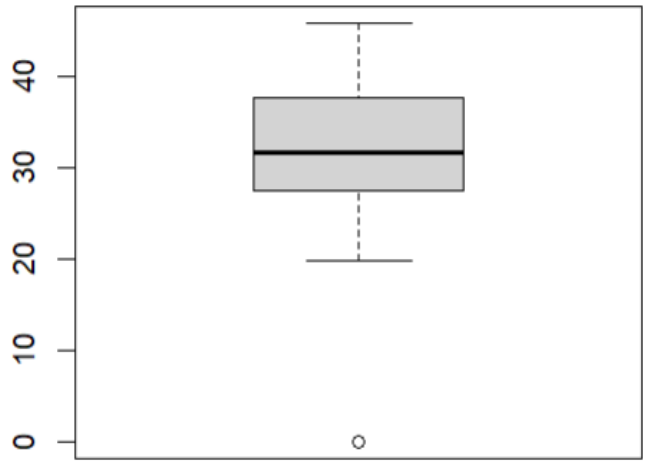
```
numeric_data = df[, sapply(df, is.numeric)]
# normalized0_1_df = as.data.frame(lapply(numeric_data$Insulin, normalize_0_1))
# normalized0_1_df
print("Taniya Ahmed 21BDS0059")
normalize_1_1_insulin = normalize_1_1(df$Insulin)
normalize_1_1_insulin
print("Taniya Ahmed 21BDS0059")
normalize_3_3_glucose = normalize_3_3(df$Glucose)
normalize_3_3_glucose
print("Taniya Ahmed 21BDS0059")
```

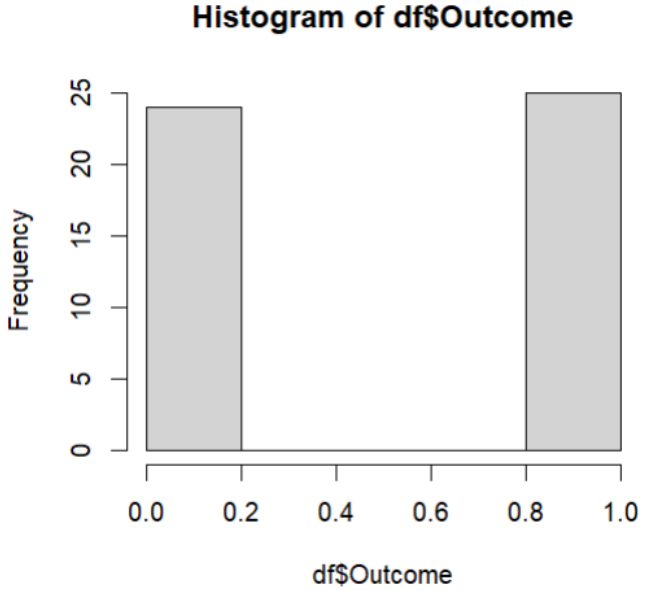
```
> numeric_data = df[, sapply(df, is.numeric)]
> # normalized0_1_df = as.data.frame(lapply(numeric_data$Insulin, normalize_0_1))
> # normalized0_1_df
> print("Taniya Ahmed 21BDS0059")
[1] "Taniya Ahmed 21BDS0059"
> normalize_1_1_insulin = normalize_1_1(df$Insulin)
> normalize_1_1_insulin
  [1] -1.0000000 -1.0000000 -1.0000000 -0.7777778 -0.6028369 -1.0000000 -0.7919622 -1.0000000  0.2836879 -1.0000000
 [11] -1.0000000 -1.0000000 -1.0000000  1.0000000 -0.5862884 -1.0000000 -0.4562648 -1.0000000 -0.8037825 -0.7730496
 [21] -0.4444444 -1.0000000 -1.0000000 -1.0000000 -0.6548463 -0.7281324 -1.0000000 -0.6690307 -0.7399527 -1.0000000
 [31] -1.0000000 -0.4208038 -0.8723404 -1.0000000 -1.0000000 -0.5460993 -1.0000000 -1.0000000 -1.0000000 -0.5106383
 [41] -0.8345154 -1.0000000 -1.0000000 -0.4326241 -1.0000000 -1.0000000 -1.0000000 -1.0000000 -1.0000000
> print("Taniya Ahmed 21BDS0059")
[1] "Taniya Ahmed 21BDS0059"
> normalize_3_3_glucose = normalize_3_3(df$Glucose)
> normalize_3_3_glucose
  [1]  0.66666667 -2.33333333  2.33333333 -2.14285714  0.14285714 -0.85714286 -2.66666667 -0.90476190  3.00000000
 [10] -0.42857143 -1.14285714  1.61904762  0.23809524  2.61904762  1.52380952 -1.61904762 -0.76190476 -1.28571429
 [19] -1.47619048 -0.90476190 -0.38095238 -1.66666667  2.95238095 -0.71428571  0.42857143 -0.42857143  0.61904762
 [28] -1.76190476  0.52380952 -0.80952381 -1.19047619  1.14285714 -2.19047619 -2.00000000 -0.57142857 -1.47619048
 [37]  0.19047619 -1.52380952 -2.09523810 -1.09523810  2.19047619 -0.04761905 -1.33333333  1.76190476  1.19047619
 [46]  2.19047619  0.57142857 -3.00000000 -1.47619048
> print("Taniya Ahmed 21BDS0059")
[1] "Taniya Ahmed 21BDS0059"
>
```

5. **Plotting a few variables.**

```
boxplot(df$BMI)

hist(df$Outcome)

plot(df$Age, df$BloodPressure)

print("Taniya Ahmed 21BDS0059")
```







6. Finding and imputing outliers

```
# finding outlier
# method 1 - using interquartile range
iqr = IQR(df$Insulin)
lower_bound = quantile(df$Insulin, 0.25)
upper_bound = quantile(df$Insulin, 0.75) + 1.5 * iqr
outlier = df %>% filter((Insulin < lower_bound) | (Insulin > upper_bound))
outlier
# imputing of the outlier
median_norm = median(df$Insulin[df$Insulin >= lower_bound & df$Insulin <=
upper_bound])
print(median_norm)
# method 2 – boxplot
```

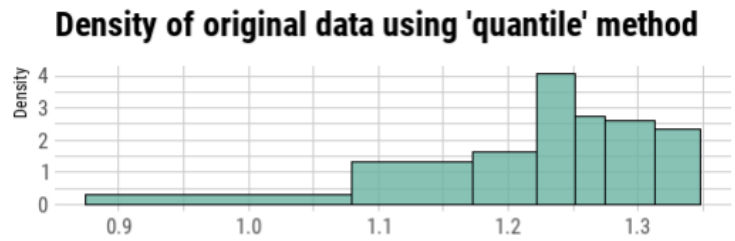
```
> # finding outlier
> # method 1 - using interquartile range
> iqr = IQR(df$Insulin)
> lower_bound = quantile(df$Insulin, 0.25)
> upper_bound = quantile(df$Insulin, 0.75) + 1.5 * iqr
> outlier = df %>% filter((Insulin < lower_bound) | (Insulin > upper_bound))
> outlier
  Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI DiabetesPedigreeFunction Age Outcome
1           2     197            70            45     543 30.5                    0.158  53       1
2           1     189            60            23     846 30.1                    0.398  59       1
> # imputing of the outlier
> median_norm = median(df$Insulin[df$Insulin >= lower_bound & df$Insulin <= upper_bound])
> print(median_norm)
[1] 0
> # Compute IQR
> iqr = IQR(df$Insulin, na.rm = TRUE)
>
> # Compute lower and upper bounds
> lower_bound = quantile(df$Insulin, 0.25, na.rm = TRUE)
> upper_bound = quantile(df$Insulin, 0.75, na.rm = TRUE) + 1.5 * iqr
>
> # Find outliers
> outlier = df %>% filter((Insulin < lower_bound) | (Insulin > upper_bound))
> print(outlier)
  Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI DiabetesPedigreeFunction Age Outcome
1           2     197            70            45     543 30.5                    0.158  53       1
2           1     189            60            23     846 30.1                    0.398  59       1
>
> # Impute outliers with median of non-outliers
> median_norm = median(df$Insulin[df$Insulin >= lower_bound & df$Insulin <= upper_bound], na.rm = TRUE)
> print(median_norm)
[1] 0
```

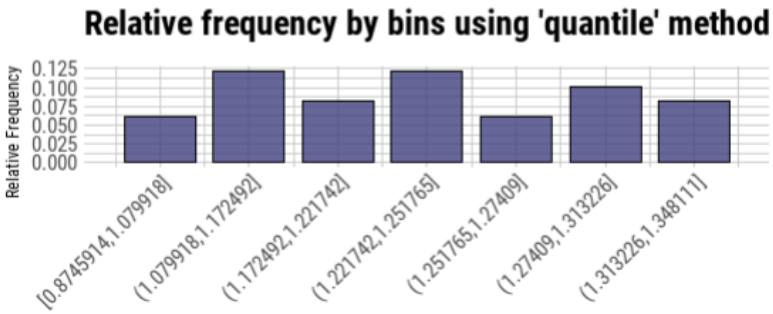
7. Skewness of the dataframe and their transformation.

```
# skewness of the dataframe
skewed_df = apply(df, 2, skewness)
skewed_df
diagnose(df)

# logarithmic transformation
df$SkinThickness = log(df$SkinThickness)

# square root transformation
df$BMI = sqrt(df$BMI)
```

```
> skewed_df = apply(df, 2, skewness)
> head(skewed_df, 5)
  Pregnancies       Glucose BloodPressure SkinThickness       Insulin
    0.1242406     0.4951351    -1.6216879           NaN     3.1193685
> diagnose(df)
# A tibble: 9 × 6
  variables               types   missing_count missing_percent unique_count unique_rate
  <chr>                   <chr>           <int>           <dbl>        <int>       <dbl>
1 Pregnancies             integer             0               0           13       0.265
2 Glucose                 integer             0               0           44       0.898
3 BloodPressure           integer             0               0           25       0.510
4 SkinThickness           numeric             0               0           24       0.490
5 Insulin                 integer             0               0           21       0.429
6 BMI                     numeric             0               0           49       1
7 DiabetesPedigreeFunction numeric            0               0           48       0.980
8 Age                     integer             0               0           29       0.592
9 Outcome                 integer             0               0            2       0.0408
>
> # logarithmic transformation
> df$SkinThickness = log(df$SkinThickness)
Warning message:
In log(df$SkinThickness) : NaNs produced
>
> # square root transformation
> df$BMI = sqrt(df$BMI)
```

```
# binning
df_binned = binning(df$SkinThickness)
df_binned

# summarising binned data
summary(df_binned)

# plotting binned data
plot(df_binned)
print("Taniya Ahmed 21BDS0059")

# transformation report
transformation_report(df, BMI)
```






```
> # binning
> df_binned = binning(df$SkinThickness)
> df_binned
binned type: quantile
number of bins: 7
x
[0.8745914,1.079918]  (1.079918,1.172492]  (1.172492,1.221742]  (1.221742,1.251765]   (1.251765,1.27409]
                   3                     6                     4                     6                    3
  (1.27409,1.313226]  (1.313226,1.348111]                  <NA>
                   5                     4                    18
>
> # summarising binned data
> summary(df_binned)
                levels freq       rate
1 [0.8745914,1.079918]    3 0.06122449
2  (1.079918,1.172492]    6 0.12244898
3  (1.172492,1.221742]    4 0.08163265
4  (1.221742,1.251765]    6 0.12244898
5   (1.251765,1.27409]    3 0.06122449
6   (1.27409,1.313226]    5 0.10204082
7  (1.313226,1.348111]    4 0.08163265
8                 <NA>   18 0.36734694
>
> # plotting binned data
> plot(df_binned)
Don't know how to automatically pick scale for object of type <table>. Defaulting to continuous.
> print("Taniya Ahmed 21BDS0059")
[1] "Taniya Ahmed 21BDS0059"
>
```

8. Applying data transformation (using dplyr and tidyr).

```
df_BMI_arranged = arrange(df, BMI)
df_BMI_arranged

df_glucose_age = select(df, Glucose, Age)
df_glucose_age

df_filtered_age = filter(df, Age > 40)
print(df_filtered_age)

df_gathered = gather(df, key = "BMI", value = "Age")
df_gathered
df
# df_spread = spread(df_gathered, key = "BMI", value = "Age")
# df_spread

df_grouped_summarized = df %>% group_by("Age") %>% summarise(mean =
mean(df$Age))
df_grouped_summarized

df_mutated = mutate(df, Age = Age + 10)
df_mutated
print("Taniya Ahmed 21BDS0059")
```

```
> # data transformation using dplyr and tidyr
> df_BMI_arranged = arrange(df, BMI)
> head(df_BMI_arranged, 5)
  Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI DiabetesPedigreeFunction Age Outcome
1           8     125            96             0       0  0.0                    0.232  54       1
2           6      92            92             0       0 19.9                    0.188  28       0
3          13     145            82            19     110 22.2                    0.245  57       0
4           7     106            92            18       0 22.7                    0.235  48       0
5           1      97            66            15     140 23.2                    0.487  22       0
>
> df_glucose_age = select(df, Glucose, Age)
> head(df_glucose_age, 5)
  Glucose Age
1     148  50
2      85  31
3     183  32
4      89  21
5     137  33
>
> df_filtered_age = filter(df, Age > 40)
> head(df_filtered_age, 5)
  Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI DiabetesPedigreeFunction Age Outcome
1           6     148            72            35       0 33.6                    0.627  50       1
2           2     197            70            45     543 30.5                    0.158  53       1
3           8     125            96             0       0  0.0                    0.232  54       1
4          10     139            80             0       0 27.1                    1.441  57       0
5           1     189            60            23     846 30.1                    0.398  59       1
>
> df_gathered = gather(df, key = "BMI", value = "Age")
> head(df_gathered, 5)
         BMI Age
1 Pregnancies   6
2 Pregnancies   1
3 Pregnancies   8
4 Pregnancies   1
5 Pregnancies   0
> # df_spread = spread(df_gathered, key = "BMI", value = "Age")
> # df_spread
>
> df_grouped_summarized = df %>% group_by("Age") %>% summarise(mean = mean(df$Age))
```

```
> df_grouped_summarized = df %>% group_by("Age") %>% summarise(mean = mean(df$Age))
> head(df_grouped_summarized, 5)
# A tibble: 1 × 2
  `"Age"`  mean
  <chr>    <dbl>
1 Age       37.6
>
> df_mutated = mutate(df, Age = Age + 10)
> head(df_mutated, 5)
  Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI DiabetesPedigreeFunction Age Outcome
1           6     148            72            35       0 33.6                    0.627  60       1
2           1      85            66            29       0 26.6                    0.351  41       0
3           8     183            64             0       0 23.3                    0.672  42       1
4           1      89            66            23      94 28.1                    0.167  31       0
5           0     137            40            35     168 43.1                    2.288  43       1
> print("Taniya Ahmed 21BDS0059")
[1] "Taniya Ahmed 21BDS0059"
>
```

9. Applying range functions.

```
# applying range function
range_df = range(df)
range_manual_df = max(df, na.rm = TRUE) - min(df, na.rm = TRUE)
print(range_df)

# range of a particular column
print(range(df$Glucose))
```

```
> # applying range function
> range_df = range(df)
> range_manual_df = max(df, na.rm = TRUE) - min(df, na.rm = TRUE)
> print(range_df)
[1]   0 846
>
> # range of a particular column
> print(range(df$Glucose))
[1]  71 197
>
```