Dataset description:

This dataset is designed for heart disease prediction and combines traditional clinical indicators with an innovative engineered feature called QuantumPatternFeature. It includes seven attributes: Age, Gender, Blood Pressure, Cholesterol, Heart Rate, QuantumPatternFeature, and HeartDisease (target variable). Each feature contributes to understanding cardiovascular risk: *Age* represents the patient's age in years; *Gender* is binary (0 = Female, 1 = Male); *Blood Pressure* and *Cholesterol* are key metrics in cardiovascular health; *Heart Rate* indicates cardiac function; and the *QuantumPatternFeature* captures complex, non-linear relationships that traditional features may miss, enabling enhanced modeling potential, especially for advanced machine learning and quantum computing applications. The target variable, *HeartDisease*, indicates whether a patient has heart disease (1) or not (0), making the dataset suitable for both classification tasks and exploratory health analytics.

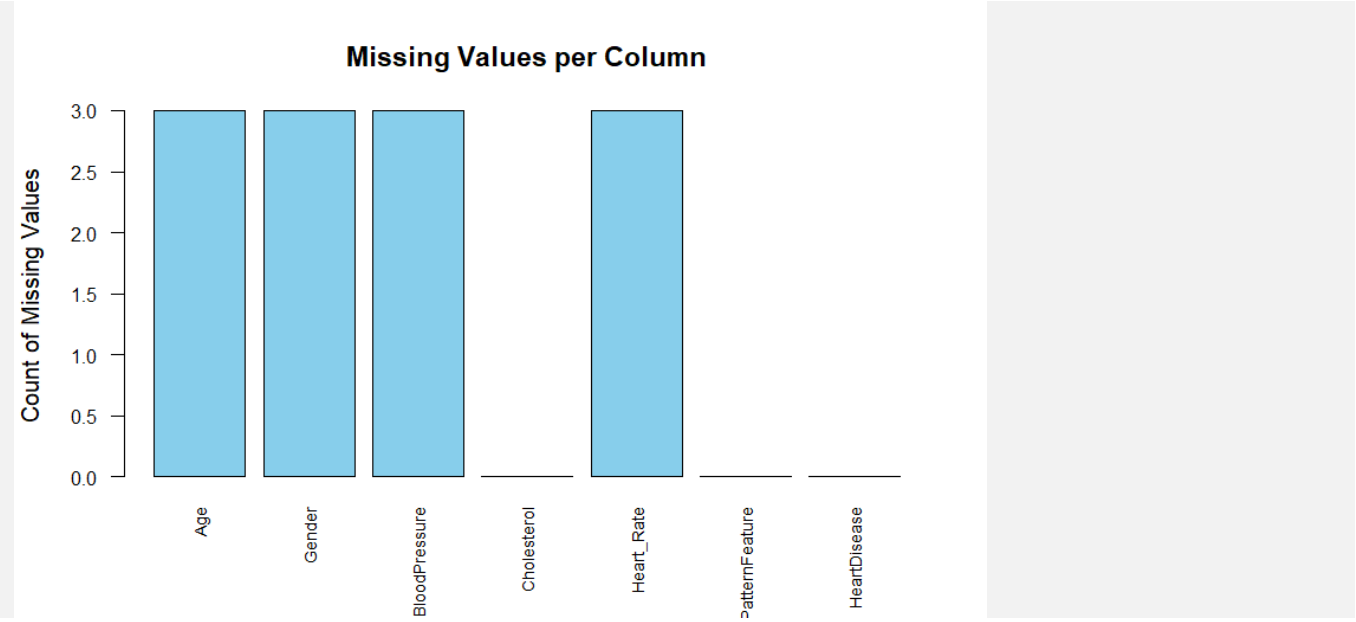
1. We can see missing values on a graph

Description: Used a barplot() to show missing values in each column, making it easier to see which features need cleaning. is.na()for missing values, colSums() for sum column wise.

Code:

```
missing_value <- colSums(is.na(file))
barplot(
  missing_value,
  names.arg = names(missing_value),
  las = 2,  # Rotate x-axis labels
  col = "skyblue",
  main = "Missing Values per Column",
  ylab = "Count of Missing Values"
)
```

Output:



2. Remove Noise Data from Age

Description:

Fixed negative age values by converting them to positive numbers, as negative ages don't make sense. abs() for absolute values,

Code:

```
file$Age <- sapply(file$Age, function(x){
  if(!is.na(x) && x< 0){
    abs(x)
  } else {x}})
```

Output:

Before Handle Noise:

```
> summary(file$Age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
-65.00   41.00   53.50   55.45   68.00  260.00       3
```

After Handle Noise:

```
> summary(file$Age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
  30.00   42.50   54.00   56.32   68.00  260.00       3
```

3. If there are any missing values in the dataset, we should apply all applicable methods from the available options to handle the missing values.

Description:

Applied different techniques for each column – removing rows consisting missing values and replacing by mean values for Age, mode imputation for Gender and Heart Rate, and median imputation for Blood Pressure.

Code:

Remove Null From Age

```
sum(is.na(file$Age))
mean_age <- ceiling(mean(file$Age, na.rm = TRUE))
file$Age[is.na(file$Age)] <- mean_age
sum(is.na(file$Age))
```

Remove Null From Gender

```
sum(is.na(file$Gender))
unique(file$Gender)
max_gender <- names(which.max(table(file$Gender)))
file$Gender[is.na(file$Gender)] <- max_gender
sum(is.na(file$Gender))
table(file$Gender)
```

Remove NULL from Blood Pressure

```
sum(is.na(file$BloodPressure))
median_BloodPressure <- median(file$BloodPressure,
na.rm = TRUE)
file$BloodPressure[is.na(file$BloodPressure)] <-
median_BloodPressure
sum(is.na(file$BloodPressure))
```

Remove NULL from Heart_Rate

```
sum(is.na(file$Heart_Rate))
max_heartRate <-
names(which.max(table(file$Heart_Rate)))
file$Heart_Rate[is.na(file$Heart_Rate)] <- max_heartRate
sum(is.na(file$Heart_Rate))
max_heartRate
colSums(is.na(file))
```

Output:

Before Handle NULL Value

```
> colSums(is.na(file))
                Age              Gender       BloodPressure
                  3                   3                   3
        Cholesterol          Heart_Rate QuantumPatternFeature
                  0                   3                   0
      HeartDisease
                  0
```

After Handle NULL Value

```
> colSums(is.na(file))
                Age              Gender       BloodPressure
                  0                   0                   0
        Cholesterol          Heart_Rate QuantumPatternFeature
                  0                   0                   0
      HeartDisease
                  0
```

4. **Detect outliers in the data set and use the appropriate approach to handle those values.**

Description:

Removed age values above 120 years as they're unrealistic for human ages.

Code:

```
count <- sum(file$Age > 120, na.rm = TRUE)
file <- file %>%
filter(is.na(Age) | (Age >= 0 & Age <= 120))
count(file)
```

Output:

Find Outliers:

```
> summary(file$Age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  30.00   43.00   54.00   56.34   68.00  260.00
```

Handle Outliers Noise:

```
> summary(file$Age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  30.00   43.00   54.00   54.07   67.00   79.00
```

5. **We can convert attributes from numeric to categorical or categorical to numeric.**

Description:

Changed Gender from numeric (0,1) to categorical (Male/Female) and Heart Rate from categorical (High/Low) to numeric (1/0).

Code:

Gender Converted from Number to Category

```
file$Gender <- recode(file$Gender,  `0` = "Male",`1` = "Female")
```

Heart Rate converted from Category to Number

```
file$Heart_Rate <- recode(file$Heart_Rate,` High` = 1, `Low` = 0)
```

Output:

Before Convert

```
> head(file,10)
# A tibble: 10 × 7
     Age Gender BloodPressure Cholesterol Heart_Rate QuantumPatternFeature
   <dbl> <chr>          <dbl>       <dbl> <chr>                      <dbl>
 1    68 1                105         191 High                        8.36
 2    58 0                 97         249 Low                         9.25
 3    44 0                 93         190 Low                         7.94
 4    72 1                 93         183 High                        6.50
 5    37 0                145         166 High                        7.65
 6    50 1                114         271 Low                         8.63
 7    68 0                156         225 Low                         7.56
 8    57 0                156         236 Low                         9.15
 9    52 0                134.        266 High                        9.15
10    40 1                121         255 Low                         9.68
# ℹ 1 more variable: HeartDisease <dbl>
```

After Convert:

```
> head(file,10)
# A tibble: 10 × 7
     Age Gender BloodPressure Cholesterol Heart_Rate QuantumPatternFeature
   <dbl> <chr>          <dbl>       <dbl>      <dbl>                  <dbl>
 1    68 Male             105         191          1                   8.36
 2    58 Female            97         249          0                   9.25
 3    44 Female            93         190          0                   7.94
 4    72 Male              93         183          1                   6.50
 5    37 Female           145         166          1                   7.65
 6    50 Male             114         271          0                   8.63
 7    68 Female           156         225          0                   7.56
 8    57 Female           156         236          0                   9.15
 9    52 Female           134.        266          1                   9.15
10    40 Male             121         255          0                   9.68
# ℹ 1 more variable: HeartDisease <dbl>
```

6. We can apply the normalization method for any continuous attribute.

Description:

Scaled Blood Pressure and Age to values between 0 and 1 to improve model performance.
Normalize=origin(x)-min(x)/max(x)-min(x)

Code:

Normalize Blood Pressure

```
file$BloodPressure  <- (file$BloodPressure  -
min(file$BloodPressure)) /
  (max(file$BloodPressure) -
min(file$BloodPressure ))
```

Normalize Age

```
file$Age  <- (file$Age  - min(file$Age)) /
          (max(file$Age) - min(file$Age ))
file$Age
```

Output:

Before Normalize:

```
> head(file[, c("Age", "BloodPressure")], 10)
# A tibble: 10 × 2
     Age BloodPressure
   <dbl>         <dbl>
 1    68           105
 2    58            97
 3    44            93
 4    72            93
 5    37           145
 6    50           114
 7    68           156
 8    57           156
 9    52           134.
10    40           121
```

After Normalize:

```
> head(file[, c("Age", "BloodPressure")], 10)
# A tibble: 10 × 2
     Age BloodPressure
   <dbl>         <dbl>
 1 0.776         0.169
 2 0.571         0.078Z
 3 0.286         0.033Z
 4 0.857         0.033Z
 5 0.143         0.618
 6 0.408         0.270
 7 0.776         0.742
 8 0.551         0.742
 9 0.449         0.489
10 0.204         0.348
```

7. We can find and remove duplicate values.

Description:

Found and removed duplicate records to prevent bias in the analysis.

Code:

Duplicate Count and Remove

```
sum(duplicated(file))
count(file)
file <- distinct(file)
```

Output:

Before Duplicate:

```
> sum(duplicated(file))
[1] 2
```

After Duplicate:

```
> sum(duplicated(file))
[1] 0
```

8. We can apply some filtering methods to filter the data.

Description:

Filtered the dataset to include only valid Gender values (Male or Female).

Code:

```
file <- file[file$Gender %in% c("Male", "Female"),]
```

Output:

	Age	Gender	BloodPressure	Cholesterol	Heart_Rate
1	68	Female	105	191	High
2	58	Male	97	249	Low
3	44	Male	93	190	Low
4	72	Female	93	183	High
5	37	Male	145	166	High
6	50	Female	114	271	Low
8	NA	Male	156	236	Low
9	52	Male		266	High
10	40	Female	121	255	Low
11	40	Female	139	235	Low
12	53	Female	150	176	
13	65	Male	140	206	High
14	69	Female	108	180	Low
15	53	Female	110	283	High
16	32	Female	94	247	High
18	31	Female	131	202	Low
19	53	Female	150	287	High
20	73	Male	111	294	Low
21	59	Female	110	271	Low
22	67	Female	159	195	Low

9. Detect invalid data in the data set and use the appropriate approach to handle those values.

Description:

Cleaned Blood Pressure values by removing non-numeric characters.

Code:

Handle Invalid

```
file$BloodPressure <- gsub("[^0-9]", "",
file$BloodPressure)
file$BloodPressure <-
as.numeric(file$BloodPressure)
```

Output:

Before Invalid:

```
> sapply(file, class)
                Age              Gender        BloodPressure
          "numeric"         "character"         "character"
        Cholesterol          Heart_Rate QuantumPatternFeature
          "numeric"         "character"             "numeric"
      HeartDisease
          "numeric"
```

After Handle Invalid:

```
> sapply(file, class)
                Age              Gender        BloodPressure
          "numeric"         "character"             "numeric"
        Cholesterol          Heart_Rate QuantumPatternFeature
          "numeric"         "character"             "numeric"
      HeartDisease
```

10. We can convert the imbalanced data set into the balanced data set.

Description:

Used both undersampling (reducing majority class) and oversampling (increasing minority class) to address class imbalance in heart disease cases.

Code:

Imbalanced handle using Under sampling

```
minority_n <- nrow(filter(file, HeartDisease ==
0))
majority_sample <- file %>%
  filter(HeartDisease == 1) %>%
  sample_n(minority_n)
count(majority_sample)
undersampled_data <-
bind_rows(majority_sample, file %>%
filter(HeartDisease == 0))
```

Output:

Imbalanced Data

```
> table(file$HeartDisease)

 0  1
59 88
```

Balanced data

```
> table(undersampled_data$HeartDisease)

 0  1
59 59
```

Imbalanced handle using Over sampling

```
majority_n <- nrow(filter(file, HeartDisease ==
1))

minority_sample <- file %>%
  filter(HeartDisease == 0) %>%
  sample_n(majority_n, replace = TRUE)
oversampled_data <- bind_rows(
  file %>% filter(HeartDisease == 1),
  minority_sample
)
```

Imbalanced Data

```
> table(file$HeartDisease)

 0  1
59 88
```

Balanced data

```
> table(oversampled_data$HeartDisease)

 0  1
88 88
```

11.Split the dataset for Training and Testing.

Description:

Divided data into 80% training and 20% testing sets while maintaining class distribution.

Code:

Split Dataset

```
set.seed(123)
index <- sample(1:nrow(oversampled_data), size
= 0.8 * nrow(oversampled_data))
train_data <- oversampled_data[index, ]
test_data  <- oversampled_data[-index, ]
table(train_data$HeartDisease)
table(test_data$HeartDisease)
```

Output:

After split Trainset

```
> table(train_data$HeartDisease)

 0  1
70 70
```

After split Testset

```
> table(test_data$HeartDisease)

 0  1
18 18
```

12. Compare the central tendencies (mean, median, mode) of Age across different groups of Gender and interpret the results

Description:

Compared mean, median, and mode of Age between genders.

Code:

Split Dataset

```
age_stats_gender <- file %>%
  group_by(Gender) %>%
  summarise(
    Mean_Age = mean(Age, na.rm = TRUE),
    Median_Age = median(Age, na.rm = TRUE),
    Mode_Age = as.numeric(names(sort(table(Age), decreasing = TRUE)[1]))
  )
print(age_stats_gender)
```

Output:

```
> print(age_stats_gender)
# A tibble: 2 × 4
  Gender Mean_Age Median_Age Mode_Age
  <chr>     <dbl>      <dbl>    <dbl>
1 Female    0.488      0.490    0.163
2 Male      0.491      0.480    0.143
```

13. Compare Age's central tendencies (mean, median, mode) across Heart Rate and interpret the results.

Description:

Examined age statistics between different heart rate groups.

Code:

```
age_stats_by_hr <- file %>%
  group_by(Heart_Rate) %>%
  summarise(
    Mean_Age = mean(Age, na.rm = TRUE),
    Median_Age = median(Age, na.rm = TRUE),
    Mode_Age = as.numeric(names(sort(table(Age), decreasing = TRUE)[1]))
  )
print(age_stats_by_hr)
```

Output:

```
> print(age_stats_by_hr)
# A tibble: 2 × 4
  Heart_Rate Mean_Age Median_Age Mode_Age
       <dbl>    <dbl>      <dbl>    <dbl>
1          0    0.503      0.510    0.878
2          1    0.466      0.469    0.265
```

14. Compare the Spread (Range, IQR, Variance, Standard Deviation) of Age across different groups of Gender and interpret the results.

Description:

Calculated range, IQR, variance, and standard deviation of Age across gender groups to understand variability.

Code:

```
age_spread_gender <- file %>%
  group_by(Gender) %>%
  summarise(
    Min_Age = min(Age, na.rm = TRUE),
    Max_Age = max(Age, na.rm = TRUE),
```

```
  Range_Age = Max_Age - Min_Age,
  IQR_Age = IQR(Age, na.rm = TRUE),
  Variance_Age = var(Age, na.rm = TRUE),
  SD_Age = sd(Age, na.rm = TRUE)
 )

print(age_spread_gender)
```

Output:

```
> print(age_spread_gender)
# A tibble: 2 x 7
  Gender Min_Age Max_Age Range_Age IQR_Age Variance_Age SD_Age
  <chr>    <dbl>   <dbl>     <dbl>   <dbl>        <dbl>  <dbl>
1 Female       0       1         1   0.480       0.0827  0.288
2 Male         0       1         1   0.510       0.0903  0.300
```