Final Project - Machine Learning

Or Gabay 314923681 & Danielle Shrem 208150433 & Shachar Oron 322807231

2023-08-15

Introduction:

Endometriosis is a chronic progressive inflammatory disease that affects about 1 in 10 women in the population. In this disease, tissue similar to the lining of the uterus grows outside the uterus. This leads to inflammation and scar tissue forming in the pelvic region and (rarely) elsewhere in the body. There is no known way to prevent endometriosis. There is no cure, but its symptoms can be treated with medicines or, in some cases, surgery. For our final project in the "Machine Learning" course, we used a database that contains information on 144 female patients who are suspected of being diagnosed with endometriosis, of which 56 actually are. Our data contained all kinds of comprehensive details about the clinical characteristics of all the women, treatment background, daily life, and personal background. Our data is relatively limited because it takes an average of 10 years to diagnose the disease for each woman and today there is a lack of data concerning this disease. We chose to work with this database because two of the members of the group, Daniel and Shahar, are sick with the disease, and the issue is very close to our hearts. Our main goal in the project is to use machine learning algorithms in order to shed light on the disease, to understand whether there are prominent characteristics or whether the combination of them contributes to the condition. Because it is a "transparent" disease on which there is not enough data, we consider it an important task to do research on it and, among other things, we will also strive to understand what are the significant symptoms that help diagnose the disease.

Due to lack of space and page limitation, we will list here the main conclusions and the main graphs that helped us to analyze the data, while more detailed and in-depth information will be included in the annexed pages at the end of the file.

Cleaning the data:

As a reminder, our data contains information on 143 female patients. For each patient we have 54 different features of characteristics.Our data contains a lot of missing values. We have dicided that columns that has more then half of NA values- we will delete it. There will be still some NA values in the data, so, we installed the 'mice' library that replace the NA values with other values according to the distribution of the other values in the data.

```
data <- read.csv("../FinalProjectML/data.csv")
col_names <- as.character(data[6, ])
new_table <- data[15:nrow(data), ]
colnames(new_table) <- col_names
# Remove rows with NA values:
# Create a logical vector indicating rows with any non-NA value
non_na_rows <- apply(new_table, 1, function(row) any(!is.na(row)))
new_table_clean <- new_table[non_na_rows, ]
column_number <- which(colnames(new_table_clean) == "Diagnosed with endo (ie at least one biopsy positve)")

new_table_cut <- new_table_clean[, 2:58]
new_table_cut <- new_table_cut[, -c(55, 56)]
n_counts <- colSums(new_table_cut == "n", na.rm = TRUE)
total_rows <- nrow(new_table_cut)
# Identify columns with more than half "n" values
columns_to_drop <- names(n_counts[n_counts > total_rows/2])
# Drop the columns from the table
new_table_cut_filtered <- new_table_cut[, !(names(new_table_cut) %in% columns_to_drop)]
new_table_cut_filtered[, 3][new_table_cut_filtered[, 3] == "n"] <- "5"
```

```
# Count the number of 'n' values in each column
n_values <- colSums(new_table_cut_filtered == "n", na.rm = TRUE)
library(mice)
# Replace "n" values with NA
new_table_cut_filtered[new_table_cut_filtered == "n"] <- NA
new_table_cut_filtered[new_table_cut_filtered == "N"] <- NA
new_table_cut_filtered <- as.data.frame(lapply(new_table_cut_filtered, as.numeric))# Convert to numeric
imputation_object <- mice(new_table_cut_filtered, m = 1)

# Access the imputed data without modifying column names
imputed_data <- complete(imputation_object, action = "long", include = FALSE)
```

The code helped us identify the number of "n" values present in each column of the `new_table_cut` data frame. By using `colSums`, you can efficiently calculate these counts across all columns. Now when we got our data cleaned. We will jump in and start by loading the clean data and analyze it.

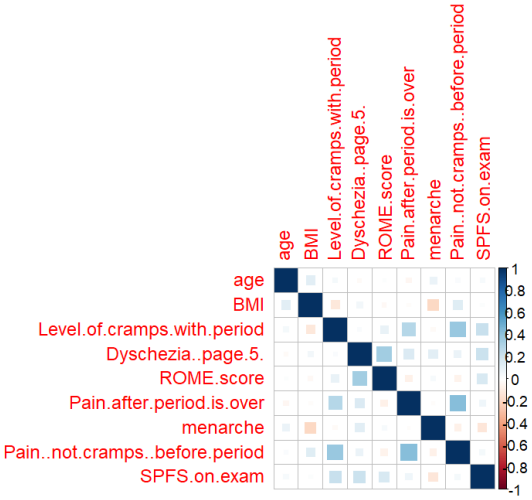
EDA:

```
mydata <- read.csv("../FinalProjectML/imputed_data.csv")))# first, get the data from the csv
```

Correlation plot:

A positive correlation indicates that as one variable (e.g., dyschezia) increases, the other variable (e.g., ROME score) also tends to increase. A negative correlation, on the other hand, means that as one variable increases, the other variable tends to decrease. A correlation of zero suggests no linear relationship between the variables.

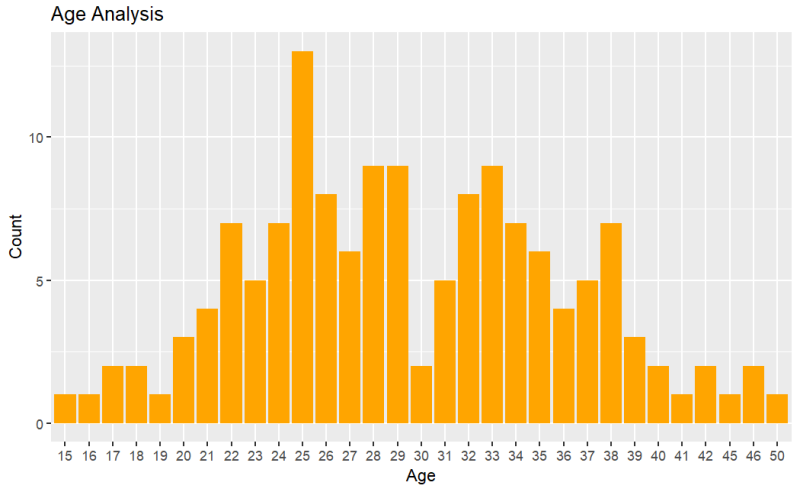
```
# Select the desired columns
selected_columns <- mydata[, c("age", "BMI",
                               "Level.of.cramps.with.period",
                               "Dyschezia..page.5.", "ROME.score", "Pain.after.period.is.over", "menarche", "
Pain..not.cramps..before.period", "SPFS.on.exam")]
cor_matrix <- cor(selected_columns)# Compute the correlation matrix
corrplot(cor_matrix, method = "square") # Create the correlation plot
```



It can be concluded that there is a strong relationship between dyschezia and ROME score in endometriosis patient. The strong relationship suggests a connection between endometriosis lesions affecting the gastrointestinal tract and pelvic discomfort. Dyschezia, a common symptom in endometriosis, is associated with inflammation, adhesions, or scarring caused by these lesions, leading to pain during bowel movements . In the other hand, ROME score is a diagnostic classification system used to evaluate and classify functional gastrointestinal disorders.What make sense because endometriosis, is associated with inflammation that can be in in the digestive system and cause gastrological disorders. In addition, it can be seen that there is a strong connection between the pain features in the study.

Age Analysis:

```
mydata %>%
  group_by(age) %>%
  count() %>%
  ggplot(aes(x = as.factor(age), y = n)) +
  geom_bar(stat = "identity", fill = "orange") +
  ggtitle("Age Analysis") +
  xlab("Age") +
  ylab("Count")
```

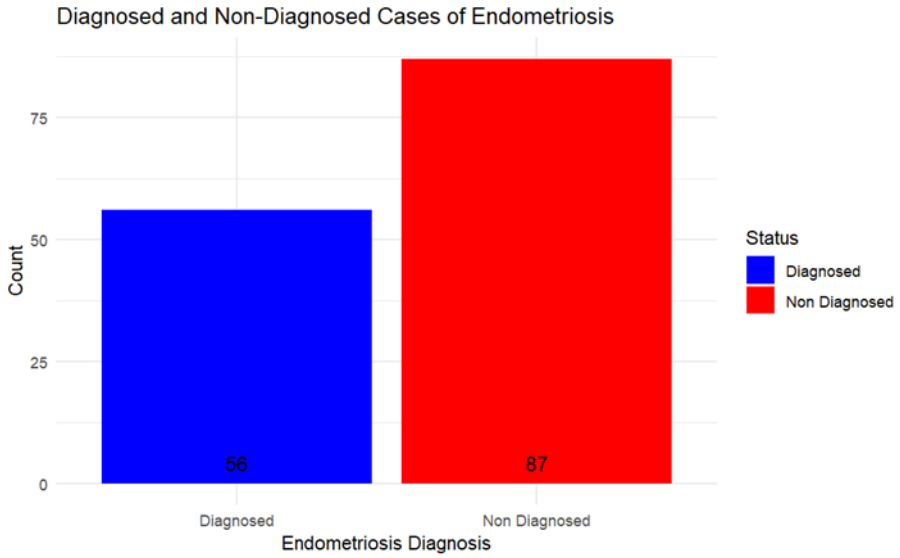


It can be seen that the patients are between the ages of 15 and 50, when there is a higher concentration of patients in the middle of this age range (except for the age of 30), we assume that this is logical because this is also the fertile period of the woman, and in the middle of the age range the fertile period reaches its hormonal peak.

Diagnosed VS. non diagnosed patients:

Our data contains 143 patients, some of them are Diagnosed with Endometriosis but not every one. We would like to check the relationship between the number of those diagnosed and those who are not so that we can know things to continue such as how to normalize the data.

```
diagnosed_data <- filter(mydata, `Diagnosed.with.endo..ie.at.least.one.biopsy.positve.` == 1)
non_diagnosed_data <- filter(mydata, `Diagnosed.with.endo..ie.at.least.one.biopsy.positve.` == 0)
# Create a new column for diagnosed status
diagnosed_data$Status <- "Diagnosed"
non_diagnosed_data$Status <- "Non Diagnosed"
combined_data <- rbind(diagnosed_data, non_diagnosed_data) # Combine the diagnosed and non-diagnosed data
count_data <- combined_data %>% # Calculate the counts for each diagnosis status
  group_by(Status) %>%
  summarise(Count = n())
# Plot the data
ggplot(combined_data, aes(x = Status, fill = Status)) +
  geom_bar() +
  geom_text(data = count_data, aes(label = Count), vjust = -0.5, color = "black", size = 4, stat = "count") +
  xlab("Endometriosis Diagnosis") +
  ylab("Count") +
  ggtitle("Diagnosed and Non-Diagnosed Cases of Endometriosis") +
  scale_fill_manual(values = c("Diagnosed" = "blue", "Non Diagnosed" = "red")) +
  theme_minimal()
```

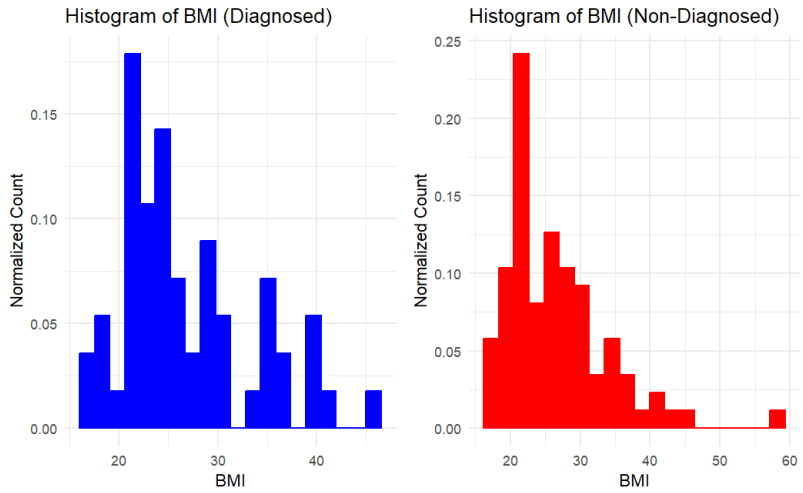


It can be seen that there are 56 diagnosed with endometriosis compared to 87 who are not diagnosed.

Correlation between BMI and Endometriosis:

We needed to normalize the counts in the histogram for better comparison between diagnosed and non-diagnosed patients, by using the position = "identity" argument in the geom_histogram() function and divide the counts by the total number of patients in each group.

```
# Calculate the total number of diagnosed and non-diagnosed patients
total_diagnosed <- nrow(diagnosed_data)
total_non_diagnosed <- nrow(non_diagnosed_data)
# Create a histogram of BMI for diagnosed patients with normalized counts
histogram_diagnosed <- ggplot(diagnosed_data, aes(x = BMI)) +
  geom_histogram(aes(y = after_stat(count) / total_diagnosed), color = "blue", fill = "blue", bins = 20, posi
tion = "identity") +
  labs(x = "BMI", y = "Normalized Count", title = "Histogram of BMI (Diagnosed)") +
  theme_minimal()
# Create a histogram of BMI for non-diagnosed patients with normalized counts
histogram_non_diagnosed <- ggplot(non_diagnosed_data, aes(x = BMI)) +
  geom_histogram(aes(y = after_stat(count) / total_non_diagnosed), color = "red", fill = "red", bins = 20, po
sition = "identity") +
  labs(x = "BMI", y = "Normalized Count", title = "Histogram of BMI (Non-Diagnosed)") +
  theme_minimal()
# Combine the histograms into a single plot
combined_plot <- cowplot::plot_grid(histogram_diagnosed, histogram_non_diagnosed, ncol = 2)
print(combined_plot) # Display the combined plot
```

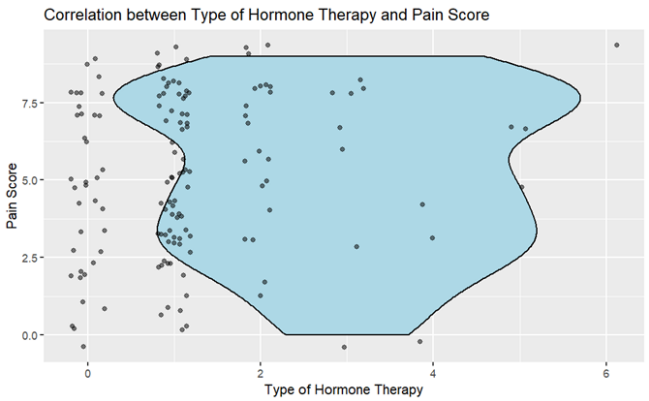


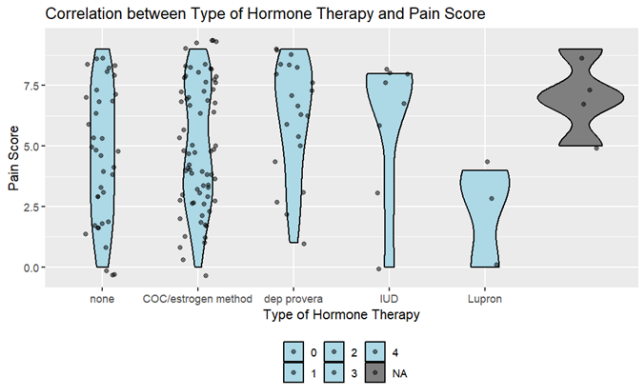
Explanation about the BMI values: Underweight - BMI less than 18.5. Normal weight - BMI ranges from 18.5 to 25. Overweight - BMI ranges from 25 to 30. Obesity - BMI greater than 30. we can infer that both diagnosed and non diagnosed BMI values are mostly normal. The first thought was that most endo patients will have an abnormal BMI due to the effects the disease has on the digestive system.In conclusion, compared to healthy people, the BMI of endo patients is not much different.

Different treatments:

First of all we would like to check the effectiveness of the hormonal treatment. In our data there is a column 'Initial.visual.analog.score'. A visual analog scale (VAS) is a measurement tool commonly used in medical and research settings to assess subjective experiences or perceptions, such as pain intensity.

```
# create a frequency table of Company.Location
freq_table <- table(mydata$Initial.visual.analog.score)
# create a dataframe from the frequency table
freq_df <- data.frame(location = names(freq_table), frequency = as.numeric(freq_table))
# order the dataframe by frequency and select top 40 countries
top_locations <- freq_df[order(freq_df$frequency, decreasing = TRUE), ][1:40, "location"]
# filter the data to only include the top 40 countries
mydata_filtered <- mydata[mydata$Initial.visual.analog.score %in% top_locations, ]
mydata_filtered <- mydata[complete.cases(mydata$Initial.visual.analog.score) & complete.cases(mydata$prior.ho
rmone.therapy..page.4.), ]
# Create the violin plot
ggplot(data = mydata_filtered, aes(x = prior.hormone.therapy..page.4., y = Initial.visual.analog.score)) +
  geom_violin(fill = "lightblue", color = "black") +
  geom_jitter(color = "black", width = 0.2, alpha = 0.5) +
  xlab("Type of Hormone Therapy") +
  ylab("Pain Score") +
  ggtitle("Correlation between Type of Hormone Therapy and Pain Score")

ggplot(data = mydata_filtered, aes(x = factor(prior.hormone.therapy..page.4., levels = 0:4), y = Initial.visu
al.analog.score, fill = factor(prior.hormone.therapy..page.4., levels = 0:4))) +
  geom_violin(color = "black") +
  geom_jitter(color = "black", width = 0.2, alpha = 0.5) +
  scale_x_discrete(breaks = 0:4, labels = c("none", "COC/estrogen method", "dep provera", "IUD", "Lupron")) +
  scale_fill_manual(values = c("lightblue", "lightblue", "lightblue", "lightblue", "lightblue")) +
  xlab("Type of Hormone Therapy") +
  ylab("Pain Score") +
  ggtitle("Correlation between Type of Hormone Therapy and Pain Score") +
  theme(legend.title = element_blank(), legend.position = "bottom")
```





The violin plot will display the "type of hormone therapy" on the x-axis and the "Pain Score" on the y-axis.The light blue area represents the density or frequency of data points at different values along the y-axis (in this case, the pain score). The width of the violin at each y-value indicates the estimated density of data points at that specific value. The wider the violin, the higher the density of data points at that particular value. Each data point represents an individual observation, and the color of the data points represents the different categories of "type of hormone

therapy". The color scale used ranges from red to green. the different type of hormones are: 0 = none 1 = COC/estrogen method- The combined oral contraceptive pill (COC) is a tablet that contains two hormones, progestogen and estrogen, and is taken daily to prevent pregnancy. 2 = dep provera - An injectable contraceptive. This is a hormonal treatment based on progestin, which is injected into the muscle and is released slowly, thus delaying ovulation in the uterus. 3 = IUD - Intrauterine device 4 = Lupron - is used to prevent premature ovulation in cycles of controlled ovarian stimulation for in vitro fertilization (IVF).

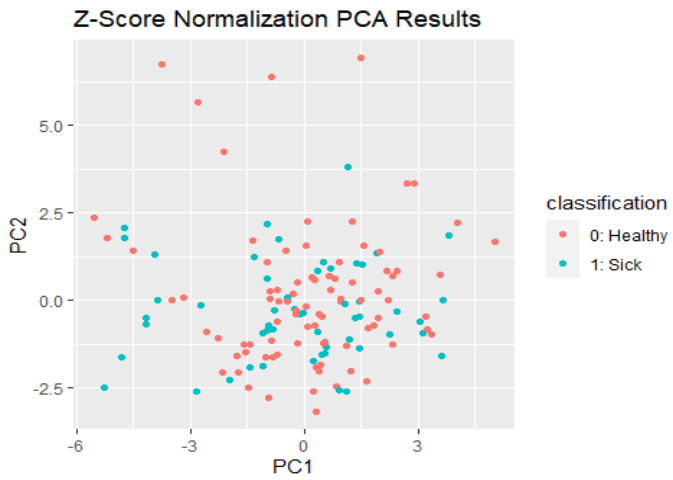
We can infer important information from this plot. for example: (1) The ones that are using the 4th type of hormone therapy, Lupron, have a strong pain score. From the research we have done, we discovered that this drug has serious side effects so in order to use the drug, the medical condition is probably really serious. (2) There is no difference in the distribution of the different pain scores between the different hormonal treatments. In all of them the pain levels are varied in the same way. (3) The majority of individuals in the dataset have either "none" (0) or "COC/estrogen method" (1) as their type of hormone therapy.

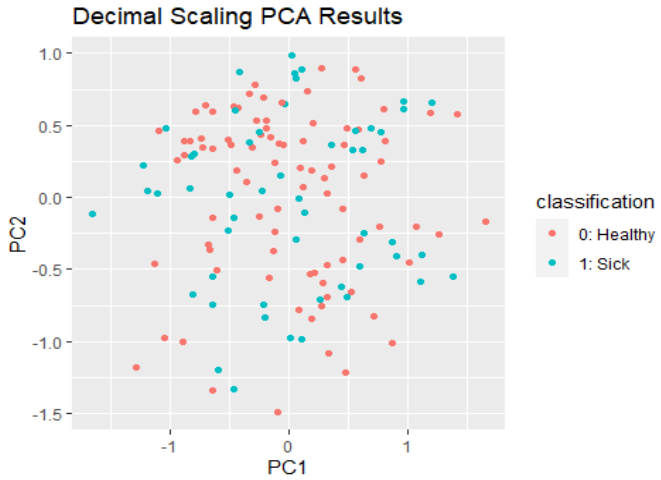
PCA:

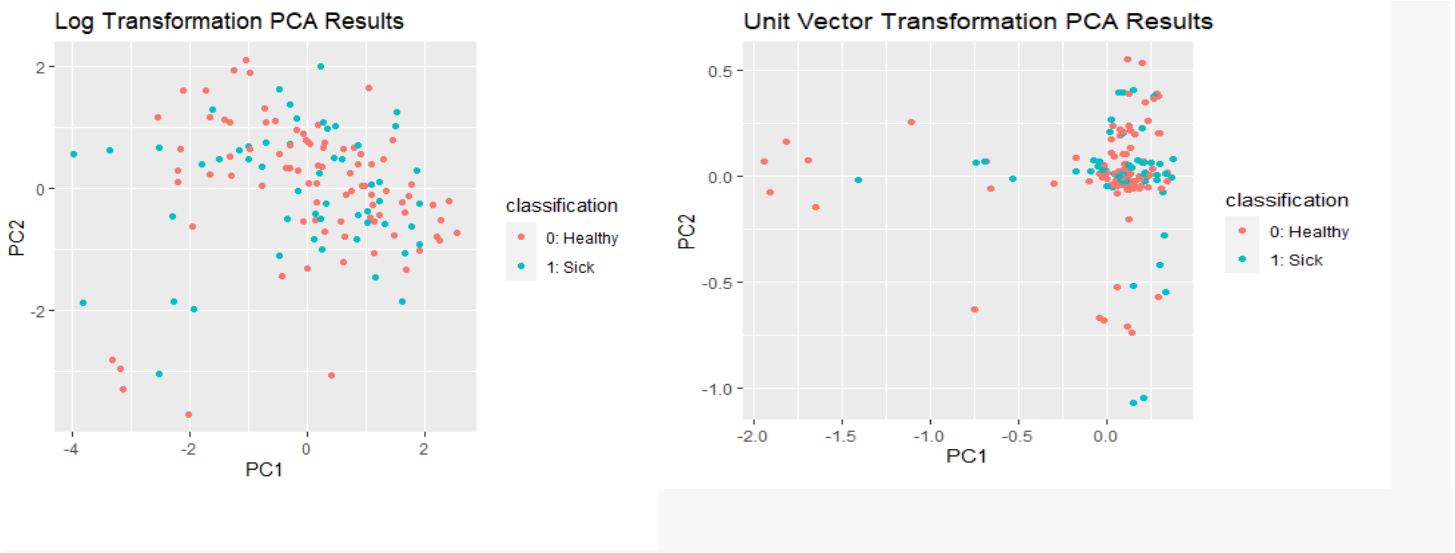
This code performs Principal Component Analysis (PCA) on a dataset to explore its patterns and variability. The PCA contains some inportant steps: (1) Data preprocessin: removing the "Intraoperative.complication" column from the original dataset, resulting in mydata_clean, checks for constant and zero columnsand retains only the valid columns that have varying values. (2) Data normalization: Four different normalization techniques-z-score normalization,min-max normalization, decimal scaling normalization and log transformation normalization. (3) PCA computation: Principal component scores and loadings for each dataset are stored. (4) Data frame creation for plotting and plotting.

```
# Remove the "Intraoperative.complication" column
mydata_clean <- mydata[1:(nrow(mydata)-3), ]
data_clean <- mydata_clean[, -which(names(mydata_clean) == "Intraoperative.complication")]
# Remove the label column
data_without_label <- data_clean[, -which(names(data_clean) == "Diagnosed.with.endo..ie.at.least.one.biopsy.p
ositve.")]
# Check for constant and zero columns
non_constant_cols <- sapply(data_without_label, function(x) !all(x == x[1]))
non_zero_cols <- sapply(data_without_label, function(x) any(x != 0))
valid_cols <- non_constant_cols & non_zero_cols
data_without_label <- data_without_label[, valid_cols]# Select only valid columns
zscore_data <- scale(data_without_label)# Z-score normalization
# Min-max normalization
minmax_data <- apply(data_without_label, 2, function(x) (x - min(x)) / (max(x) - min(x)))
# Decimal scaling normalization
decimal_data <- apply(data_without_label, 2, function(x) x / 10^ceiling(log10(max(abs(x)))))
# Log transformation normalization
log_data <- log1p(abs(data_without_label)) * sign(data_without_label)
# Unit vector transformation normalization
unit_vector_data <- sweep(data_without_label, 2, sqrt(rowSums(data_without_label^2)), FUN = "/")
# Perform PCA on z-score, min-max, decimal scaled data, log transformed data, unit vector transformed data

pca_result_zscore <- prcomp(zscore_data)
pca_result_minmax <- prcomp(minmax_data)
pca_result_decimal <- prcomp(decimal_data)
pca_result_log <- prcomp(log_data)
pca_result_unit_vector <- prcomp(unit_vector_data)
# Create a data frame for plotting (z-score normalization)
pca_graph_zscore <- data.frame(
  PC1 = pca_result_zscore$x[, 1],
  PC2 = pca_result_zscore$x[, 2],
  label = data_clean$Diagnosed.with.endo..ie.at.least.one.biopsy.positve.,
  classification = factor(data_clean$Diagnosed.with.endo..ie.at.least.one.biopsy.positve., levels = c(0, 1),
labels = c("0: Healthy", "1: Sick"))
)
pca_graph_decimal <- data.frame(
```

```r
  PC1 = pca_result_decimal$x[, 1],
  PC2 = pca_result_decimal$x[, 2],
  label = data_clean$Diagnosed.with.endo..ie.at.least.one.biopsy.positve.,
  classification = factor(data_clean$Diagnosed.with.endo..ie.at.least.one.biopsy.positve., levels = c(0, 1),
labels = c("0: Healthy", "1: Sick"))
)
pca_graph_log <- data.frame(
  PC1 = pca_result_log$x[, 1],
  PC2 = pca_result_log$x[, 2],
  label = data_clean$Diagnosed.with.endo..ie.at.least.one.biopsy.positve.,
  classification = factor(data_clean$Diagnosed.with.endo..ie.at.least.one.biopsy.positve., levels = c(0, 1),
labels = c("0: Healthy", "1: Sick"))
)
pca_graph_unit_vector <- data.frame(
  PC1 = pca_result_unit_vector$x[, 1],
  PC2 = pca_result_unit_vector$x[, 2],
  label = data_clean$Diagnosed.with.endo..ie.at.least.one.biopsy.positve.,
  classification = factor(data_clean$Diagnosed.with.endo..ie.at.least.one.biopsy.positve., levels = c(0, 1),
labels = c("0: Healthy", "1: Sick"))
)
# Plot PCA results: z-score normalized data,decimal scaled data,log transformed data,unit vector transformed
data
library(ggplot2)
ggplot(pca_graph_zscore, aes(x = PC1, y = PC2, col = classification)) +
  geom_point() + labs(title = "Z-Score Normalization PCA Results")

ggplot(pca_graph_decimal, aes(x = PC1, y = PC2, col = classification)) +
  geom_point() + labs(title = "Decimal Scaling PCA Results")
ggplot(pca_graph_log, aes(x = PC1, y = PC2, col = classification)) +
  geom_point() + labs(title = "Log Transformation PCA Results")

ggplot(pca_graph_unit_vector, aes(x = PC1, y = PC2, col = classification)) +
  geom_point() + labs(title = "Unit Vector Transformation PCA Results")
```







Based on the results of the PCA plots, it can be concluded that the data is not linearly separable. The data points belonging to the Healthy and Sick classes overlap and do not form distinct clusters in the PCA space, and this can be seen in each of the four different normalization methods we used .This suggests that a linear classification approac h may not be sufficient to accurately classify the data. Non-linear classification algorithms or additional feature eng ineering may be necessary to improve the separation between the two classes.

Algorithms:

We used four algorithms: KNN, SVM, k-means, decision tree. We will note that even though we have concluded that the data is not linearly separable, we will still use linear algorithms in order to try to draw further conclusions and we will also adjust the algorithms so that they work on our data, despite the problematic conditions.

KNN:

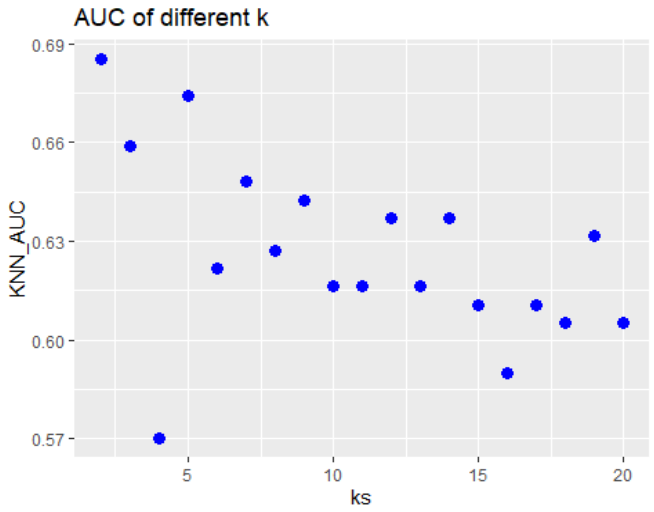
KNN can be effective for non-linearly separable data because it considers the local structure of the data. It assigns a label to a new data point based on the labels of its nearest neighbors. If there are distinct clusters or groups within the data, KNN can capture those patterns and make accurate predictions.

```
set.seed(112233)
# Split the data into training and testing sets
train_indices <- sample(1:nrow(mydata), 0.7 * nrow(mydata))  # 70% for training
train_data <- mydata[train_indices, ]
test_data <- mydata[-train_indices, ]
# Scale numeric columns
train_data[, 3:46] <- scale(train_data[, 3:46])
test_data[, 3:46] <- scale(test_data[, 3:46])
# Extract labels for train_data and test_data
train_labels <- train_data$Diagnosed.with.endo..ie.at.least.one.biopsy.positve.
test_labels <- test_data$Diagnosed.with.endo..ie.at.least.one.biopsy.positve.
train_size <- nrow(train_data)
class_size <- length(train_labels)
KNN_DF <- data.frame(number = 1:dim(test_data)[1], stringsAsFactors = TRUE)
KNN_AUC <- c()
ks <- c()
if (train_size != class_size) {
  stop("Mismatch in the lengths of training data and class labels.")
}
```

We did a test which K should we choose according to the accuracy values that each K yields.

```
library(pROC)
library(mice)
```

```r
# Combine train_data and test_data for imputation
combined_data <- rbind(train_data, test_data)
# Remove the "Intraoperative.complication" column
combined_data <- combined_data[, !(colnames(combined_data) %in% "Intraoperative.complication")]
imputed_data <- mice(combined_data, method = "mean") # Perform mean imputation

imputed_data_complete <- complete(imputed_data, 1) # Extract the imputed data
# Split the imputed data back into train_data and test_data
train_data_imputed <- imputed_data_complete[1:nrow(train_data), ]
test_data_imputed <- imputed_data_complete[(nrow(train_data) + 1):nrow(imputed_data_complete), ]
# Extract the class labels for training data
train_labels <- train_data_imputed$Diagnosed.with.endo..ie.at.least.one.biopsy.positve.
KNN_AUC <- c()
ks <- c()
for (K in 2:20) {
  predictions <- knn(train = train_data_imputed[, -1],  # Exclude the label column
                     test = test_data_imputed[, -1],    # Exclude the label column
                     cl = train_labels, k = K)
  KNN_DF[toString(K)] <- as.numeric(predictions)
  roc_auc <- auc(as.numeric(test_labels), as.numeric(predictions))
  KNN_AUC <- append(KNN_AUC, roc_auc)
  ks <- append(ks, toString(K))
  #print(paste("AUC for k ", toString(K), " : ", toString(roc_auc), sep = ""), quote = FALSE)
}
# Create a data frame with ks and KNN_AUC for plotting
roc_data <- data.frame(ks = as.numeric(ks), KNN_AUC)
roc_data <- roc_data[order(roc_data$ks), ] # Sort the data frame by ks in ascending order
ggplot(roc_data, aes(x = ks, y = KNN_AUC)) +
  ggtitle("AUC of different k") +
  geom_point(color = "blue", size = 3)
```



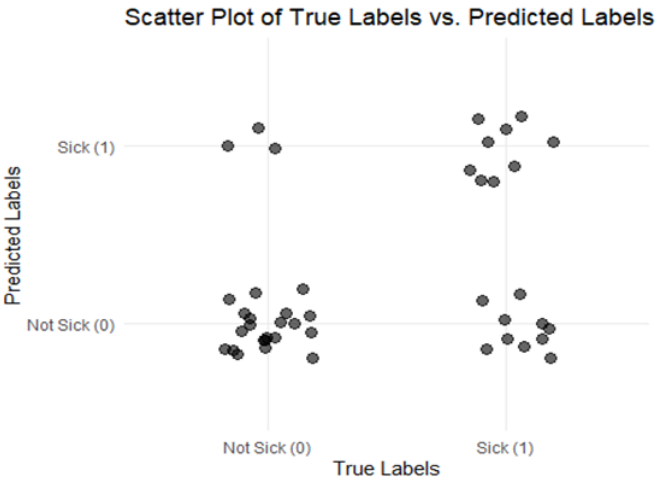

```r
KNN_AUC
```

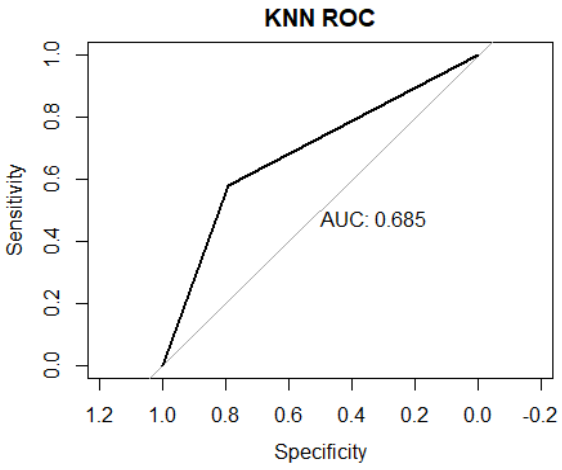
```
##  [1] 0.6853070 0.6589912 0.5701754 0.6743421 0.6217105 0.6480263 0.6271930
##  [8] 0.6425439 0.6162281 0.6162281 0.6370614 0.6162281 0.6370614 0.6107456
## [15] 0.5899123 0.6107456 0.6052632 0.6315789 0.6052632
```

One of the best models of KNN is when K=2 there both AUC score is 0.685. Now we visualize the distribution of predicted labels vs. true labels of K=2.

```r
# Load required libraries
library(ggplot2)
predictions <- knn(train = train_data_imputed[, -1],  # Exclude the label column
                     test = test_data_imputed[, -1],    # Exclude the label column
                     cl = train_labels, k = 4)
# Create a data frame with predicted labels and true labels for the test_data_imputed
```

```
test_results <- data.frame(True_Labels = as.numeric(test_labels),
                           Predicted_Labels = as.numeric(predictions))
# Create a scatter plot to visualize the distribution of predicted labels vs. true labels
ggplot(test_results, aes(x = as.factor(True_Labels), y = as.factor(Predicted_Labels))) +
  geom_jitter(width = 0.2, height = 0.2, size = 3, alpha = 0.6, color = "black") +
  labs(x = "True Labels", y = "Predicted Labels", title = "Scatter Plot of True Labels vs. Predicted Labels")
+
  scale_x_discrete(labels = c("Not Sick (0)", "Sick (1)")) +
  scale_y_discrete(labels = c("Not Sick (0)", "Sick (1)")) +
  theme_minimal()

# Plot the ROC curve
plot.roc(test_data$Diagnosed.with.endo..ie.at.least.one.biopsy.positve., KNN_DF$'2', main = "KNN ROC", print.
auc = TRUE)
```





From the prediction distribution diagram for K=2 and the ROC diagram we see that the percentage of accuracy is not high enough, that is, the algorithm fails to be really accurate, especially since 2 neighbors is not a high number in itself. This is not good enough, we will try other algorithms that will provide us better results.

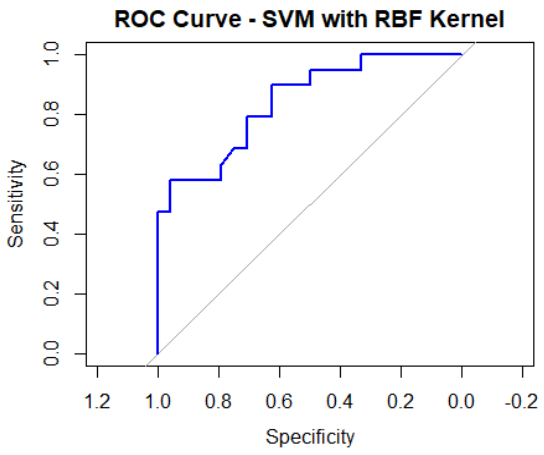
SVM:

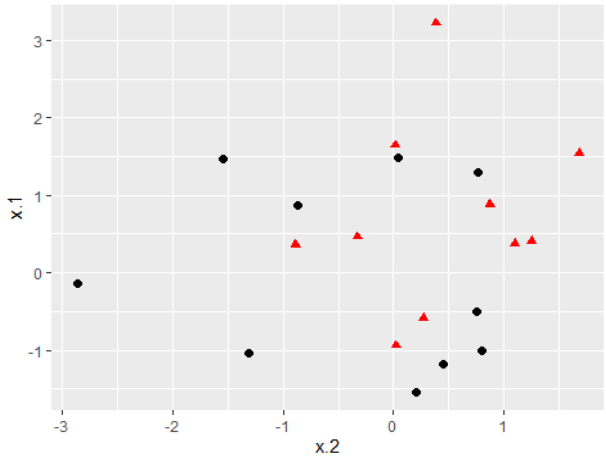
We need to use the kernel trick because our data is non linear separable.

```
library(pROC)
library(mice)
library(e1071)
# Combine train_data and test_data for imputation
combined_data <- rbind(train_data, test_data)
# Remove the "Intraoperative.complication" column
combined_data <- combined_data[, !(colnames(combined_data) %in% "Intraoperative.complication")]
imputed_data <- mice(combined_data, method = "mean")# Perform mean imputation

imputed_data_complete <- complete(imputed_data, 1)# Extract the imputed data
# Split the imputed data back into train_data and test_data
train_data_imputed <- imputed_data_complete[1:nrow(train_data), ]
test_data_imputed <- imputed_data_complete[(nrow(train_data) + 1):nrow(imputed_data_complete), ]
# Extract the class labels for training data
train_labels <- train_data_imputed$Diagnosed.with.endo..ie.at.least.one.biopsy.positve.
# Train the SVM model with a radial basis function (RBF) kernel
svm_model <- svm(Diagnosed.with.endo..ie.at.least.one.biopsy.positve. ~ .,
                 data = train_data_imputed[, -1], # Exclude the label column
                 kernel = "radial")
# Make predictions on test data using the trained SVM model
predictions <- predict(svm_model, test_data_imputed[, -1]) # Exclude the label column
roc_auc <- roc(test_labels, predictions, levels = c(0, 1), auc = TRUE)# Calculate AUC
print(paste("AUC:", roc_auc$auc))
```

```
## [1] "AUC: 0.841008771929825"
roc_auc <- auc(as.numeric(test_labels), as.numeric(predictions))# Calculate AUC
print(paste("AUC: ", roc_auc, sep = ""))# Print the AUC value
```

```
## [1] "AUC: 0.841008771929825"

roc <- roc(as.numeric(test_labels), as.numeric(predictions))# Plot the ROC curve
plot(roc, col = "blue", main = "ROC Curve - SVM with RBF Kernel")

set.seed(457)
# Construct sample data set - not completely separated
x <- matrix(rnorm(20*2), ncol = 2)
y <- c(rep(-1,10), rep(1,10))
x[y==1,] <- x[y==1,] + 1
dat <- data.frame(x=x, y=as.factor(y))
# Plot data set
ggplot(data = dat, aes(x = x.2, y = x.1, color = y, shape = y)) +
  geom_point(size = 2) +
  scale_color_manual(values=c("#000000", "#FF0000")) +
  theme(legend.position = "none")
```



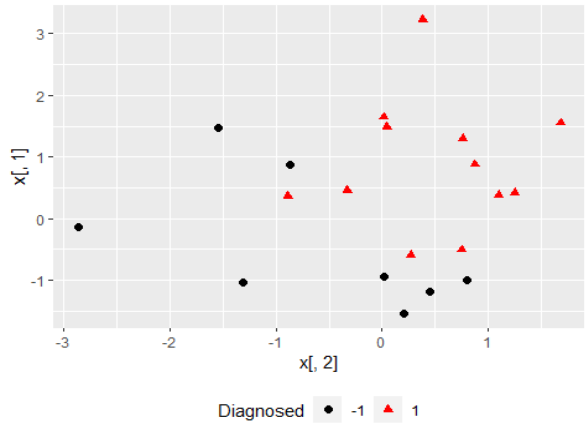


The results of SVM (Support Vector Machine) algorithm with an RBF (Radial Basis Function) kernel are indicative of a common behavior when dealing with non-linearly separable data. This means that the data points cannot be separated by a straight line or a hyperplane in the original feature space, but they can be transformed into a higher-dimensional space where separation becomes possible. ROC Curve Behavior: A well-behaved ROC curve for a good classifier should show a smooth, upward-sloping curve. However, in some cases, especially when dealing with complex data and non-linearities, we might observe "jumps" or "steps" in the ROC curve like we got in our results. This can happen due to the nature of the decision boundary learned by the SVM in the transformed feature space.These "steps" in the ROC curve are an indication of how the SVM's decision boundary is affecting its classification decisions. The steps occur when the threshold for classification changes, leading to a sudden shift in the true positive rate (sensitivity) and false positive rate (1-specificity). In summary, the steps we're observing in the ROC curve with an SVM using an RBF kernel are not necessarily a problem, but they do indicate that the decision boundary of the SVM in the transformed space is causing shifts in classification decision.

SVC - Support Vector Classifiers:

```
library(e1071)
library(caret)
library(ggplot2)
set.seed(10)
# Split the data into training and test sets
train_indices <- createDataPartition(dat$y, p = 0.7, list = FALSE)
train_data <- dat[train_indices, ]
test_data <- dat[-train_indices, ]
```

```
# Train the SVM model
svmModel <- svm(y ~ ., data = train_data, kernel = "radial", probability = TRUE)
svmPrediction <- predict(svmModel, test_data) # Make predictions on the test set
svmPredictionProb <- attr(predict(svmModel, test_data, probability = TRUE), "probabilities")[, 2]
# Convert the predicted values to a factor with the same levels as the test data
svmPrediction <- factor(svmPrediction, levels = levels(test_data$y))
svmConfMat <- confusionMatrix(svmPrediction, test_data$y) # Evaluate the model performance
dat$predicted <- predict(svmModel, dat) # Plot data set with predicted classes
ggplot(data = dat, aes(x = x[, 2], y = x[, 1], color = predicted, shape = predicted)) +
  geom_point(size = 2) +
  scale_color_manual(values = c("#000000", "#FF0000")) +
  labs(color = "Diagnosed", shape = "Diagnosed") +
  theme(legend.position = "bottom")
```

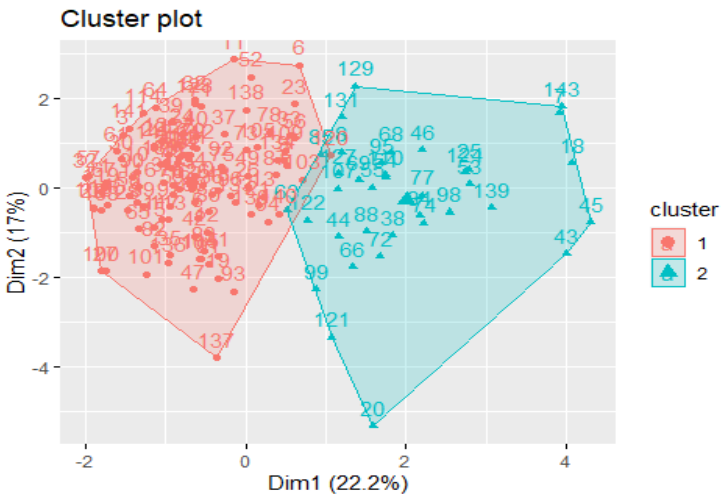


We tried to do another method of svm and see if we are capable of improving the results. You can see a certain separation between the red triangles and the black dots (which represent the sick versus the healthy), although the separation is not exact and not the best there is, but there is an improvement from the previous graph.The results were not good so we stayed with this method at the end.

K-means:

Now we will want to run the k-means algorithm, which will use the various features we have on the data. We tried to perform the algorithm on all features or on selected features and finally we decided to perform it on the 6 most important and significant features in the data. In the first step we checked what the 6 most important features are and used them (the code how we checked and the features themselves were detailed in the appendices). Now we will perform the k-means algorithm with two centers, sick and healthy.

```
set.seed(123)
endoNorm <- as.data.frame(scale(endo))
endos_K2 <- kmeans(endoNorm, centers = 2, nstart = 25) # apply kmeans algorithm
cluster_1_samples <- endoNorm[endos_K2$cluster == 1, ] # Get the samples in cluster 1
cluster_2_samples <- endoNorm[endos_K2$cluster == 2, ] # Get the samples in cluster 2
# Now we visualize the clusters we have created:
fviz_cluster(endos_K2, data = endoNorm)
```




```
original_data <- read.csv("imputed_data.csv")
labels <- original_data[, 49]# Get the labels from column 49
# Compare original labels with cluster assignments
comparison <- table(labels, endos_K2$cluster)
accuracy <- sum(diag(comparison)) / sum(comparison)# Calculate accuracy
print(comparison)# Print the comparison and accuracy

##
## labels  1  2
##      0 72 15
##      1 35 21

print(paste("Accuracy:", accuracy))

## [1] "Accuracy: 0.65034965034965"

# Identify which cluster is which
if (comparison[1, 1] > comparison[2, 2]) {
  cluster1_label <- "Healthy"
  cluster2_label <- "Sick"
} else {
  cluster1_label <- "Sick"
  cluster2_label <- "Healthy"
}
# Print the cluster labels
print(paste("Cluster 1 is", cluster1_label))

## [1] "Cluster 1 is Healthy"

print(paste("Cluster 2 is", cluster2_label))

## [1] "Cluster 2 is Sick"
```

You can see from the graph that we got that there is a certain and relatively good separation of the points, so that the red points are the healthy women while the blue triangles are the sick women. However, there is still an overlap of quite a few of the sands, so the percentage of accuracy is still not that high and around 65%. Therefore, it can be concluded that we got some improvement after selecting these 6 particular features and using k-means.

Decision tree:

In this updated code, we are creating a new data frame called “endometriosis_data” by excluding the last 3 rows from the original “mydata” data frame. These last 3 rows are considered non-informative as they contain various statistical summary information that is not relevant for further analysis.

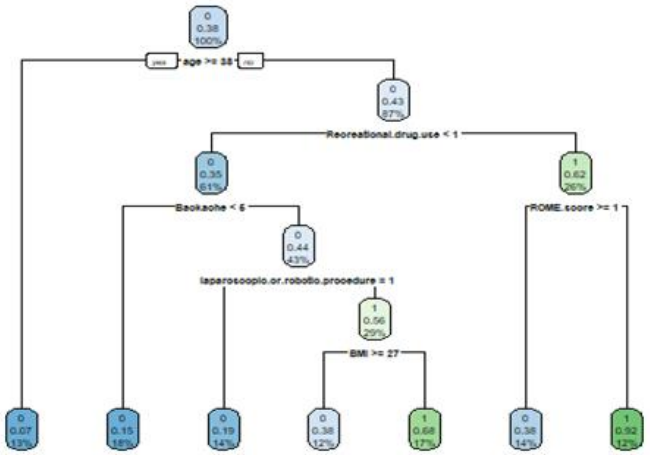
First we tried to perform the decision tree algorithm on the data as it is but we got a bad result, of a small and uninformative decision tree, as detailed in the appendices.

The short decision tree with just one feature is due to the fact that the algorithm did not find other informative features to create meaningful splits and improve the prediction. The decision tree's structure is inherently limited by the available data and the relationships between features and the target variable.

Since the feature representing the number of affected sites does not provide sufficient information for predictive modeling and is more suitable for classification rather than prediction, the decision was made to exclude this feature from the analysis. By removing this feature, we aim to train the algorithm without relying on a non-informative predictor, which can lead to a more robust and accurate model.

After excluding the feature, the decision tree will be constructed using other relevant features that are more informative and have a stronger relationship with the target variable, potentially resulting in a more insightful and meaningful tree.

```
endometriosis_data <- subset(endometriosis_data, select = -X..sites.with.confirmed.endo.on.histology)
# Convert the target variable "Diagnosed.with.endo..ie.at.least.one.biopsy.positve." to a factor
endometriosis_data$Diagnosed.with.endo..ie.at.least.one.biopsy.positve. <- as.factor(endometriosis_data$Diagnosed.with.endo..ie.at.least.one.biopsy.positve.)
# Split the data into training and test sets (80% training, 20% test)
set.seed(1234)
train_index <- sample(1:nrow(endometriosis_data), 0.8 * nrow(endometriosis_data))
train_data <- endometriosis_data[train_index, ]
test_data <- endometriosis_data[-train_index, ]
# Build the decision tree model with adjusted control parameters
tree_model <- rpart(Diagnosed.with.endo..ie.at.least.one.biopsy.positve. ~ .,
                    data = train_data,
                    method = "class",    # For classification task
                    control = rpart.control(minsplit = 20, minbucket = 10, cp = 0.001))
rpart.plot(tree_model) # Visualize the decision tree
```




```
predictions <- predict(tree_model, test_data, type = "class") # Make predictions on the test set
# Evaluate the model's accuracy
correct_predictions <- sum(predictions == test_data$Diagnosed.with.endo..ie.at.least.one.biopsy.positve.)
total_samples <- nrow(test_data)
accuracy <- correct_predictions / total_samples
cat("Model Accuracy:", accuracy, "\n")

## Model Accuracy: 0.4642857
```

We got a decision tree that is informative in a certain way that uses a number of significant features that will help to make a decision whether the patient is sick or healthy, as you can see in the graph. In some of the features we discussed earlier, we saw certain patterns and connections between them, such as age, BMI and the like. Note that the percentage of accuracy is still very low.

Summary and Conclusions:

In this report, we analyzed a database of endometriosis patients, in order to try to understand the important specific characteristics of endometriosis and to understand the causes of the disease in a better way. First, we cleaned the data, during which we saw that the data was not optimal, since the characteristics and backgrounds of the patients They are very diverse and there are a lot of features in our data.

In the next step, we performed an in-depth data analysis and presented it in different forms, for example, we tried to see a certain correlation between certain features in which menstrual pain stood out (which makes sense and characterizes many patients), and the relationship between dyschezia and ROME score.

Next, we used the PCA (Principal Component Analysis) algorithm and the conclusions drawn from the PCA graphs and the various normalization methods indicate that the data are not linearly separable in the transformed feature space. This began to hint to us about the difficulty of separating and distinguishing between the sick and the healthy.

Later we implemented 4 machine learning algorithms: KNN, SVM, K-means and decision tree. In all four algorithms, the understanding that the data is not a linear separation was strengthened and accordingly the percentages of accuracy were relatively low in all algorithms, even after we tried to adapt to our data. However, there were algorithms where we saw high percentages of accuracy, for example in SVM, where we were almost able to separate the sick from the healthy.

In addition, we saw that anyone who was found to have lesions of endometriosis - she is indeed sick, since this is the feature we removed in order to get more significant results in the decision tree. Furthermore, we saw that there are more significant features in the data such as menstrual pain, abdominal cramps, pain, and the number of sites in the biopsy that were found to be positive for endometriosis.

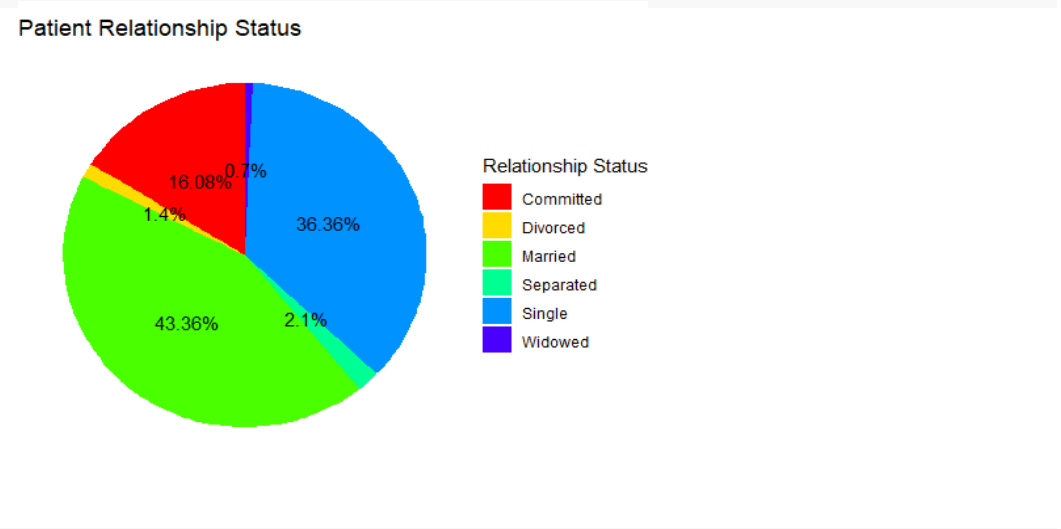
Our main insight from the findings reinforces the popular claim among doctors today in connection with endometriosis - that it is a “transparent” disease, which is very difficult to diagnose and to separate a healthy woman from a sick woman, unless the woman performs surgery in which endometriosis lesions are found, since the characteristics of the disease are very similar to the characteristics of many women, healthy, of reproductive age.

Finally, while some data exploration and machine learning classifications were performed, a comprehensive analysis of feature importance, model evaluation, and improvement strategies for classification accuracy were not explicitly addressed in the provided information. Further analysis and experimentation are needed to gain deeper insights and enhance the classification of endometriosis using the available data, or perform a new and more extensive data collection.

Supplementary Files:

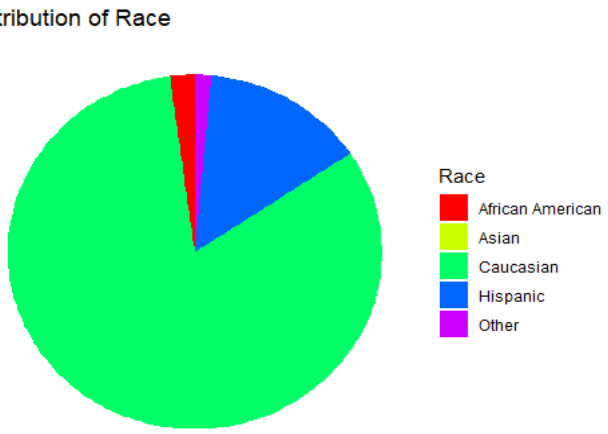
EDA - More graphs:

```
#relationship stusus
# Create a data frame for the relationship statuses and their corresponding labels
relationship_labels <- c("Single", "Married", "Widowed", "Remarried", "Separated", "Divorced", "Committed")
relationship_data <- data.frame(status = c(0, 1, 2, 3, 4, 5, 6), label = relationship_labels)
# Count the number of patients for each relationship status
relationship_counts <- mydata %>%
  count(Relationship.status) %>%
  left_join(relationship_data, by = c("Relationship.status" = "status")) %>%
  filter(!is.na(label))
# Calculate the percentages
relationship_counts <- relationship_counts %>%
  mutate(percentage = round(n / sum(n) * 100, 2))
# Create the pie chart with percentages
pie_chart <- ggplot(relationship_counts, aes(x = "", y = n, fill = label)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar("y", start = 0) +
  labs(fill = "Relationship Status",
       title = "Patient Relationship Status",
       x = NULL,
       y = NULL) +
  scale_fill_manual(values = rainbow(length(relationship_labels))) +
  theme_void() +
  geom_text(aes(label = paste0(percentage, "%")), position = position_stack(vjust = 0.5))
pie_chart
```

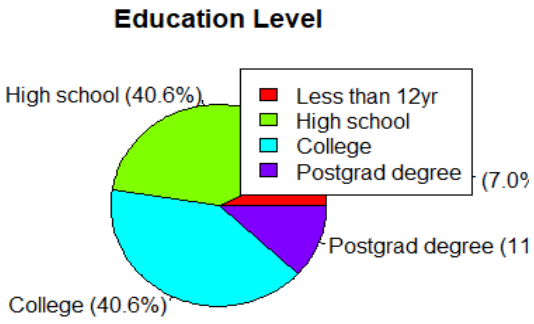



```
#race
# Create a copy of the data with the race column
race_data <- mydata[, "race"]
# Define the labels for the race categories
race_labels <- c("Caucasian", "African American", "Hispanic", "Asian", "Other")
# Convert the race column to a factor with the labels
race_data <- factor(race_data, levels = 0:4, labels = race_labels)
# Calculate the count of each race category
race_counts <- table(race_data)
# Calculate the percentages
race_percentages <- race_counts / sum(race_counts) * 100
# Create a data frame for the pie chart
race_df <- data.frame(Race = race_labels, Count = as.numeric(race_counts), Percentage = race_percentages)
# Plot the pie chart
pie_chart <- ggplot(race_df, aes(x = "", y = Count, fill = Race)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar("y", start = 0) +
  labs(fill = "Race", x = NULL, y = NULL, title = "Distribution of Race") +
```

```
  scale_fill_manual(values = rainbow(length(race_labels))) +
  theme_void()
# Display the pie chart
pie_chart
```




```
#  Education level
# Create a data frame for education levels and their counts
education_df <- data.frame(
  Education = c("Less than 12yr", "High school", "College", "Postgrad degree"),
  Count = table(mydata$Education.level)
)
# Fix the count values in the education_df data frame
education_df$Count <- education_df$Count.Freq
# Calculate the percentage
education_df$Percentage <- percent(education_df$Count / sum(education_df$Count))
# Generate the pie chart with percentages
pie(education_df$Count, labels = paste0(education_df$Education, " (", education_df$Percentage, ")"), col = ra
inbow(length(education_df$Count)))
# Add a legend
legend("topright", legend = education_df$Education, fill = rainbow(length(education_df$Count)))
# Add a title
title("Education Level")
```

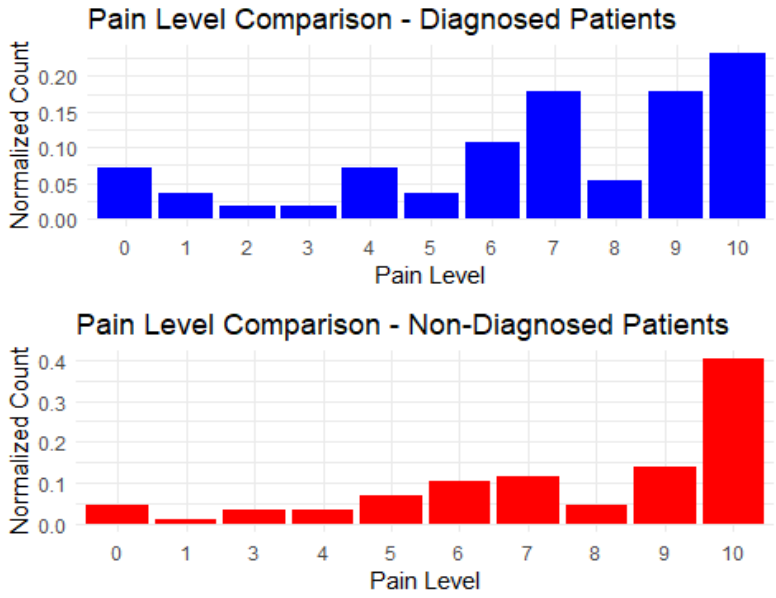


It can be concluded that most of the patients are married (43%), that most of them are Caucasian and that most of them went to college (41.8%).

Pain Level of cramps Comparison - bar plots:

Endo patients are known to suffer from pain before and during menstruation. We wanted to check the evidence given by endometriosis patients about their pain compared to those not diagnosed with endometriosis. We needed to normalize the counts in the histogram for better comparison between diagnosed and non-diagnosed patients.

```
# Calculate the total number of diagnosed and non-diagnosed patients
total_diagnosed <- nrow(diagnosed_data)
total_non_diagnosed <- nrow(non_diagnosed_data)
# Create a function to normalize counts
normalize_counts <- function(counts, total) {
  counts / total
}
# Normalize counts for diagnosed patients
diagnosed_data <- diagnosed_data %>%
  group_by(`Level.of.cramps.with.period`) %>%
  summarize(count = n()) %>%
  mutate(normalized_count = normalize_counts(count, total_diagnosed))
# Normalize counts for non-diagnosed patients
non_diagnosed_data <- non_diagnosed_data %>%
  group_by(`Level.of.cramps.with.period`) %>%
  summarize(count = n()) %>%
  mutate(normalized_count = normalize_counts(count, total_non_diagnosed))
# Create a bar plot for diagnosed patients
diagnosed_plot <- ggplot(diagnosed_data, aes(x = factor(`Level.of.cramps.with.period`, levels = c("0", "1", "
2", "3", "4", "5", "6", "7", "8", "9", "10")), y = normalized_count)) +
  geom_bar(fill = "blue", stat = "identity") +
  labs(x = "Pain Level", y = "Normalized Count", title = "Pain Level Comparison - Diagnosed Patients") +
  theme_minimal()
# Create a bar plot for non-diagnosed patients
non_diagnosed_plot <- ggplot(non_diagnosed_data, aes(x = factor(`Level.of.cramps.with.period`, levels = c("0"
, "1", "2", "3", "4", "5", "6", "7", "8", "9", "10")), y = normalized_count)) +
  geom_bar(fill = "red", stat = "identity") +
  labs(x = "Pain Level", y = "Normalized Count", title = "Pain Level Comparison - Non-Diagnosed Patients") +
  theme_minimal()
# Combine the plots into a single plot
combined_plot <- plot_grid(diagnosed_plot, non_diagnosed_plot, nrow = 2)
# Display the combined plot
print(combined_plot)
```



Since the data is normalized, it can be seen that 0.4 of the patients diagnosed with endo testified to severe menstrual pain compared to the evidence of women not diagnosed with endo (0.25). In addition, most of the

diagnosed patients testify to severe menstrual pain at strong levels compared to those who are not diagnosed, for whom the range of pain is wider.

K-means – top features:

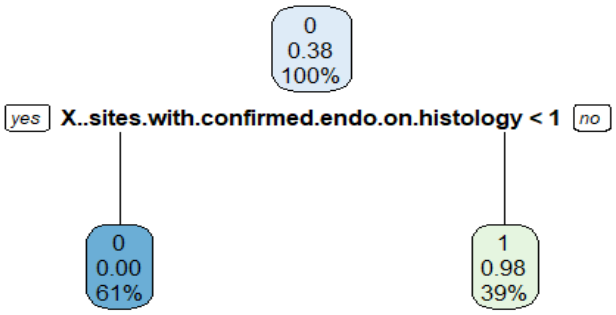
```
library(rpart)
# Feature importance:
# Load data from CSV file
data <- read.csv("imputed_data.csv")
# Split data into features and labels
features <- data[, 1:(ncol(data)-1)]
labels <- data[, ncol(data)]
# Build the decision tree model
tree <- rpart(labels ~ ., data = features, method = "class")
# Extract feature importance
feature_importance <- tree$variable.importance
# Sort features by importance in descending order
sorted_features <- sort(feature_importance, decreasing = TRUE)
# Print the top N significant features
N <- 6  # Set the number of top features you want to extract
top_features <- names(sorted_features)[1:N]
print(top_features)
```

```
## [1] "X..sites.with.confirmed.endo.on.histology"
## [2] "Pain.just.before.period"
## [3] "EBL"
## [4] "Pain..not.cramps..before.period"
## [5] "menarche"
## [6] "Initial.visual.analog.score"
```

Desicion tree:

As we wrote in the decision tree section, in the first step we tried to make a decision tree on the data as it is. In this updated code, we are creating a new data frame called “endometriosis_data” by excluding the last 3 rows from the original “mydata” data frame. These last 3 rows are considered non-informative as they contain various statistical summary information that is not relevant for further analysis.

```
# Load required libraries
library(rpart)
library(rpart.plot)
# Read the data from the CSV file
mydata <- read.csv("imputed_data.csv")
# Create a new data frame without the last 3 rows
endometriosis_data <- head(mydata, -3)
# Convert the target variable "Diagnosed.with.endo..ie.at.least.one.biopsy.positve." to a factor
endometriosis_data$Diagnosed.with.endo..ie.at.least.one.biopsy.positve. <- as.factor(endometriosis_data$Diagnosed.with.endo..ie.at.least.one.biopsy.positve.)
# Split the data into training and test sets (80% training, 20% test)
set.seed(1234)
train_index <- sample(1:nrow(endometriosis_data), 0.8 * nrow(endometriosis_data))
train_data <- endometriosis_data[train_index, ]
test_data <- endometriosis_data[-train_index, ]
# Build the decision tree model with adjusted control parameters
tree_model <- rpart(Diagnosed.with.endo..ie.at.least.one.biopsy.positve. ~ .,
                    data = train_data,
                    method = "class",    # For classification task
                    control = rpart.control(minsplit = 20, minbucket = 10, cp = 0.001))
# Visualize the decision tree
rpart.plot(tree_model)
```




```
# Make predictions on the test set
predictions <- predict(tree_model, test_data, type = "class")
# Evaluate the model's accuracy
correct_predictions <- sum(predictions == test_data$Diagnosed.with.endo..ie.at.least.one.biopsy.positve.)
total_samples <- nrow(test_data)
accuracy <- correct_predictions / total_samples
cat("Model Accuracy:", accuracy, "\n")

## Model Accuracy: 1
```