
# FACULTY OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY

## WQD 7004 – Programming for Data Science

## SEMESTER 2, 2024/2025

## Group Project

## Project Title: Prediction of Thyroid Hormone Levels and Diagnosis of Thyroid Disorders

## Group Members:


| No. | Name | Matric Number |
|---|---|---|
| 1. | MUHAMMAD NAQIB SYAHMI BIN AB RAZAK | 23073950 |
| 2. | MD TIPU SULTAN | 24072590 |
| 3. | AIEDYL DAVA AKBARI | 23070275 |
| 4. | TANISYA PRISTI AZRELIA | 24088031 |
| 5. | MARINA A/P MANIDAR | 24075325 |


## Supervised by: Dr. Ong Sim Ying

# Contents

# Introduction

Thyroid disorders, including **hypothyroidism** and **hyperthyrodism**, are **prevalent endocrine conditions** that can significantly impact **metabolic health**. Globally, the **incidence of thyroid dysfunction** is increasing, with studies indicating that thyroid disorders affect a substantial portion of the population [1]. **Thyroid-stimulating hormone (TSH)**, produced by the pituitary gland, plays a pivotal role in regulating thyroid function and is a primary marker for assessing thyroid health [2].

In this project, we are utilizing the Hyperthyroid dataset, originally sourced from the UCI Machine Learning Repository and made available via Kaggle [3], to develop **predictive models** with two primary objectives. First, we are constructing a regression model to **estimate TSH levels** based on **clinical and demographic features**. Second, we are developing a **binary classification model** to determine whether an individual exhibits signs of thyroid dysfunction. By integrating both regression and classification, this projects aims to **enhance early diagnosis** and **support effective clinical decision-making** for thyroid health management.

# Project Objectives

1. To build a regression model that estimates thyroid-stimulating hormone (TSH) levels based on clinical and demographic features.
2. To develop a binary classification model that distinguishes between individuals with thyroid dysfunction and those with normal thyroid function.
3. To evaluate the effectiveness of both models in providing accurate predictions that can aid in medical decision-making and early intervention.

# Project Questions

1. Can we accurately predict an individual's TSH (thyroid-stimulating hormone) level based on clinical and demographic data?
2. Can we determine whether an individual has thyroid dysfunction (e.g., hypothyroidism) or is in a normal state using available features in the dataset?

## Step 1: Preliminaries

In this preliminary step, we set up the working environment by loading the necessary R packages that will be used throughout the project and importing the thyroid dataset.

This includes loading essential libraries for data manipulation and visualization, and reading the dataset into a data frame. We are also previewing the first few rows to confirm that the data has been successfully imported and to understand its structure before proceeding with data preprocessing.

```
# List of required packages
packages <- c("tidyverse", "ggplot2", "dplyr", "corrplot", "randomForest",
              "Metrics", "smotefamily", "caret", "xgboost")

# Install any missing packages
installed <- packages %in% rownames(installed.packages())
if (any(!installed)) {
  install.packages(packages[!installed])
}
```

```
# Load required libraries

# Data Manipulation and Visualization
library(tidyverse)

# Plotting
library(ggplot2)

# Data Wrangling
library(dplyr)

# Correlation Plots
library(corrplot)
```

```r
# Load thyroid data
thyroid_data_df <- read.csv("../data/hypothyroid.csv")

# Preview the dataset
head(thyroid_data_df)
```

```
##   age sex on.thyroxine query.on.thyroxine on.antithyroid.medication sick
## 1  41   F            f                  f                         f    f
## 2  23   F            f                  f                         f    f
## 3  46   M            f                  f                         f    f
## 4  70   F            t                  f                         f    f
## 5  70   F            f                  f                         f    f
## 6  18   F            t                  f                         f    f
##   pregnant thyroid.surgery I131.treatment query.hypothyroid query.hyperthyroid
## 1        f               f              f                  f                  f
## 2        f               f              f                  f                  f
## 3        f               f              f                  f                  f
## 4        f               f              f                  f                  f
## 5        f               f              f                  f                  f
## 6        f               f              f                  f                  f
##   lithium goitre tumor hypopituitary psych TSH.measured  TSH T3.measured  T3
## 1       f      f     f             f     f            t  1.3           t 2.5
## 2       f      f     f             f     f            t  4.1           t   2
## 3       f      f     f             f     f            t 0.98           f   ?
## 4       f      f     f             f     f            t 0.16           t 1.9
## 5       f      f     f             f     f            t 0.72           t 1.2
## 6       f      f     f             f     f            t 0.03           f   ?
##   TT4.measured TT4 T4U.measured  T4U FTI.measured FTI TBG.measured TBG
## 1            t 125            t 1.14            t 109            f   ?
## 2            t 102            f    ?            f   ?            f   ?
## 3            t 109            t 0.91            t 120            f   ?
## 4            t 175            f    ?            f   ?            f   ?
## 5            t  61            t 0.87            t  70            f   ?
## 6            t 183            t  1.3            t 141            f   ?
##   referral.source binaryClass
## 1            SVHC           P
## 2           other           P
## 3           other           P
## 4           other           P
## 5             SVI           P
## 6           other           P
```

## Step 2: Data Preprocessing

In this data preprocessing step, we clean and prepare the thyroid dataset for analysis.

This includes handling missing values by replacing "?" with NA, removing duplicate and irrelevant columns, converting character columns to numeric types, and filtering out outliers (e.g., age >= 100).

We are also imputing missing numeric values using the column mean and encode categorical variables (e.g., sex, binaryClass) into numerical format to prepare the data for model training.

```
# Dimension of the dataset
cat("The thyroid dataset has", nrow(thyroid_data_df), "rows and",
    ncol(thyroid_data_df), "columns.\n")
```

```
## The thyroid dataset has 3772 rows and 30 columns.
```

```
# List all column names in the dataset
for (idx in seq_along(thyroid_data_df)) {
  cat(idx, ":", colnames(thyroid_data_df)[idx], "\n")
}
```

```
## 1 : age
## 2 : sex
## 3 : on.thyroxine
## 4 : query.on.thyroxine
## 5 : on.antithyroid.medication
## 6 : sick
## 7 : pregnant
## 8 : thyroid.surgery
## 9 : I131.treatment
## 10 : query.hypothyroid
## 11 : query.hyperthyroid
## 12 : lithium
## 13 : goitre
## 14 : tumor
## 15 : hypopituitary
## 16 : psych
## 17 : TSH.measured
## 18 : TSH
## 19 : T3.measured
## 20 : T3
## 21 : TT4.measured
## 22 : TT4
## 23 : T4U.measured
## 24 : T4U
## 25 : FTI.measured
## 26 : FTI
## 27 : TBG.measured
```

```
## 28 : TBG
## 29 : referral.source
## 30 : binaryClass
```

```
# Statistical summary for each column in the dataset
summary(thyroid_data_df)
```

```
##      age                sex             on.thyroxine       query.on.thyroxine
##  Length:3772        Length:3772        Length:3772        Length:3772
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##  on.antithyroid.medication     sick              pregnant
##  Length:3772               Length:3772        Length:3772
##  Class :character          Class :character   Class :character
##  Mode  :character          Mode  :character   Mode  :character
##  thyroid.surgery    I131.treatment     query.hypothyroid  query.hyperthyroid
##  Length:3772        Length:3772        Length:3772        Length:3772
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##    lithium             goitre             tumor           hypopituitary
##  Length:3772        Length:3772        Length:3772        Length:3772
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##     psych           TSH.measured           TSH            T3.measured
##  Length:3772        Length:3772        Length:3772        Length:3772
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##       T3            TT4.measured           TT4            T4U.measured
##  Length:3772        Length:3772        Length:3772        Length:3772
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##      T4U            FTI.measured           FTI            TBG.measured
##  Length:3772        Length:3772        Length:3772        Length:3772
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##      TBG            referral.source    binaryClass
##  Length:3772        Length:3772        Length:3772
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
```

```
# Structure of the dataset
str(thyroid_data_df)
```

```
## 'data.frame':    3772 obs. of  30 variables:
##  $ age                      : chr  "41" "23" "46" "70" ...
##  $ sex                      : chr  "F" "F" "M" "F" ...
##  $ on.thyroxine             : chr  "f" "f" "f" "t" ...
```

```
##  $ query.on.thyroxine       : chr  "f" "f" "f" "f" ...
##  $ on.antithyroid.medication: chr  "f" "f" "f" "f" ...
##  $ sick                     : chr  "f" "f" "f" "f" ...
##  $ pregnant                 : chr  "f" "f" "f" "f" ...
##  $ thyroid.surgery          : chr  "f" "f" "f" "f" ...
##  $ I131.treatment           : chr  "f" "f" "f" "f" ...
##  $ query.hypothyroid        : chr  "f" "f" "f" "f" ...
##  $ query.hyperthyroid       : chr  "f" "f" "f" "f" ...
##  $ lithium                  : chr  "f" "f" "f" "f" ...
##  $ goitre                   : chr  "f" "f" "f" "f" ...
##  $ tumor                    : chr  "f" "f" "f" "f" ...
##  $ hypopituitary            : chr  "f" "f" "f" "f" ...
##  $ psych                    : chr  "f" "f" "f" "f" ...
##  $ TSH.measured             : chr  "t" "t" "t" "t" ...
##  $ TSH                      : chr  "1.3" "4.1" "0.98" "0.16" ...
##  $ T3.measured              : chr  "t" "t" "f" "t" ...
##  $ T3                       : chr  "2.5" "2" "?" "1.9" ...
##  $ TT4.measured             : chr  "t" "t" "t" "t" ...
##  $ TT4                      : chr  "125" "102" "109" "175" ...
##  $ T4U.measured             : chr  "t" "f" "t" "f" ...
##  $ T4U                      : chr  "1.14" "?" "0.91" "?" ...
##  $ FTI.measured             : chr  "t" "f" "t" "f" ...
##  $ FTI                      : chr  "109" "?" "120" "?" ...
##  $ TBG.measured             : chr  "f" "f" "f" "f" ...
##  $ TBG                      : chr  "?" "?" "?" "?" ...
##  $ referral.source          : chr  "SVHC" "other" "other" "other" ...
##  $ binaryClass              : chr  "P" "P" "P" "P" ...
```

```r
# Check for duplicate rows
duplicate_values <- sum(duplicated(thyroid_data_df))

cat("\nDuplicate values in the dataset: \n")
print(duplicate_values)

# Remove duplicate rows
thyroid_data_df <- thyroid_data_df[!duplicated(thyroid_data_df), ]

cat("\nDuplicate values after removal: \n")
print(sum(duplicated(thyroid_data_df)))

thyroid_data_df

# Loop through all columns and print value counts
for (colname in names(thyroid_data_df)) {
  cat("\nColumn:", colname, "\n")
  print(table(thyroid_data_df[[colname]], useNA = "ifany"))
}
```

```r
# Removing irrelevant columns
thyroid_data_df$TBG <- NULL
thyroid_data_df$referral.source <- NULL

for (idx in seq_along(thyroid_data_df)) {
  cat(idx, ":", colnames(thyroid_data_df)[idx], "\n")
}

# Replace data points '?' with NA
thyroid_data_df[thyroid_data_df == "?"] <- NA
na_counts <- data.frame(
  Column = names(thyroid_data_df),
  Missing_Values = colSums(is.na(thyroid_data_df)),
  row.names = NULL
)
print(na_counts)

# Data encoding
thyroid_data_df$binaryClass <- ifelse(thyroid_data_df$binaryClass == "P", 0,
                                ifelse(thyroid_data_df$binaryClass == "N", 1, NA))

thyroid_data_df$sex <- ifelse(thyroid_data_df$sex == "F", 1,
                       ifelse(thyroid_data_df$sex == "M", 0, NA))

thyroid_data_df[thyroid_data_df == "t"] <- 1
thyroid_data_df[thyroid_data_df == "f"] <- 0

thyroid_data_df

str(thyroid_data_df)

# Converting character columns to numeric
char_cols <- sapply(thyroid_data_df, is.character)

thyroid_data_df[char_cols] <- lapply(thyroid_data_df[char_cols],
                                     function(x) as.numeric(as.character(x)))

str(thyroid_data_df)

# Replace data points '?' with NA
thyroid_data_df[thyroid_data_df == "?"] <- NA
na_counts <- data.frame(
  Column = names(thyroid_data_df),
  Missing_Values = colSums(is.na(thyroid_data_df)),
  row.names = NULL
)
```

```r
print(na_counts)

# Impute missing numeric values using the column mean
thyroid_data_df$age[is.na(thyroid_data_df$age)] <- mean(thyroid_data_df$age,
                                                        na.rm = TRUE)
thyroid_data_df$sex[is.na(thyroid_data_df$sex)] <- mean(thyroid_data_df$sex,
                                                        na.rm = TRUE)

# Impute missing numeric values using the column mean
cols_to_impute <- c("TSH", "T3", "TT4", "T4U", "FTI")

# Mean imputation
thyroid_data_df <- thyroid_data_df %>%
  mutate(across(all_of(cols_to_impute), ~ ifelse(is.na(.), mean(., na.rm = TRUE), .)))

# Replace data points '?' with NA
thyroid_data_df[thyroid_data_df == "?"] <- NA
na_counts <- data.frame(
  Column = names(thyroid_data_df),
  Missing_Values = colSums(is.na(thyroid_data_df)),
  row.names = NULL
)
print(na_counts)

# Filtering out outliers
thyroid_data_df <- subset(thyroid_data_df, age < 100)

# Preview cleaned dataset
thyroid_data_df

# Preview standardised structured of the dataset
str(thyroid_data_df)
```

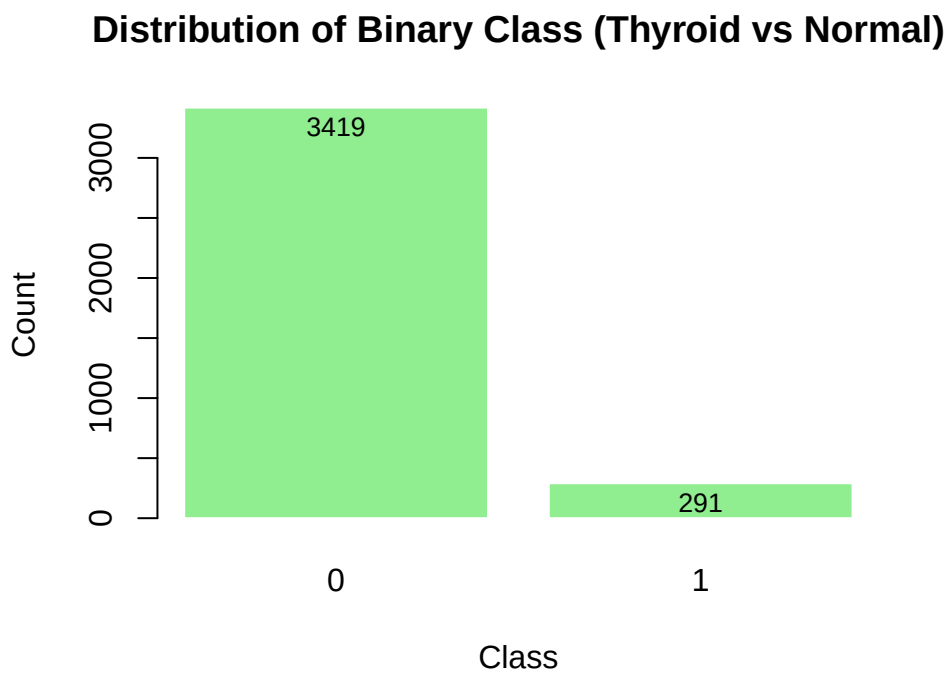
## Step 3: Exploratory Data Analysis (EDA)

In this EDA step, we explore the thyroid dataset to uncover patterns, trends, and relationships between variables.

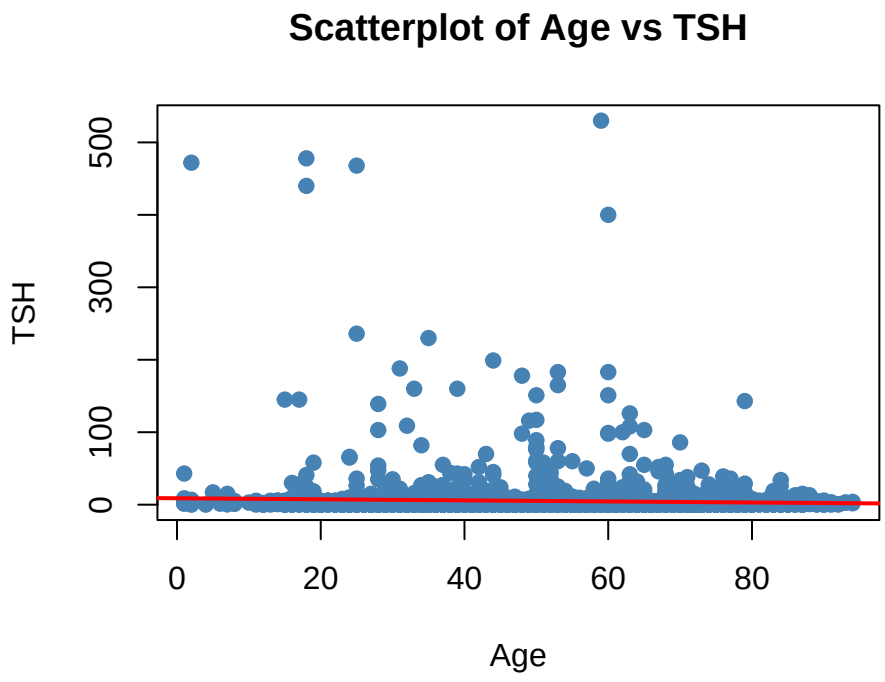
We begin by examining the distribution of individual features, such as binary class labels, using a bar plot. Relationships between numerical features (e.g., Age vs TSH levels) are explored using a scatter plot.

Lastly, a correlation matrix is generated for key thyroid-related features (e.g., T3, TT4, TSH) to visualize interdependencies and identify potential predictor variables for TSH levels.

```
# Distribution of Binary Class (Thyroid vs Normal)
counts <- table(thyroid_data_df$binaryClass)

bar_positions <- barplot(counts,
                 main = "Distribution of Binary Class (Thyroid vs Normal)",
                 xlab = "Class",
                 ylab = "Count",
                 col = "lightgreen",
                 border = "white")

text(x = bar_positions,
     y = counts - 400,
     label = counts,
     pos = 3,
     cex = 0.8)
```



```r
# Scatterplot between Age and TSH levels
plot(thyroid_data_df$age, thyroid_data_df$TSH,
     main = "Scatterplot of Age vs TSH",
     xlab = "Age", ylab = "TSH",
     pch = 19, col = "steelblue")

# Add a linear regression line to observe trend
abline(lm(TSH ~ age, data = thyroid_data_df), col = "red", lwd = 2)
```

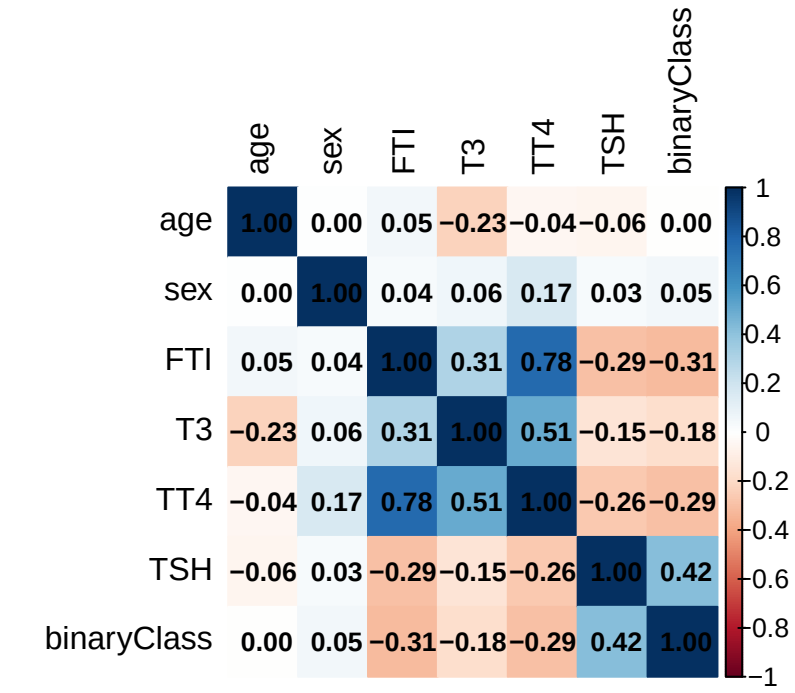



```r
correlation <- cor(thyroid_data_df$age, thyroid_data_df$TSH, use = "complete.obs")
cat("The correlation between Age and TSH is: ", round(correlation, 3), "\n")
```

```
## The correlation between Age and TSH is:  -0.059
```

```r
# Heatmap of Thyroid-Related Features
selected_cols <- c("age", "sex", "FTI", "T3", "TT4", "TSH", "binaryClass")
subset_data <- thyroid_data_df[, selected_cols]

# Compute correlation matrix
cor_matrix <- cor(subset_data, use = "complete.obs")

library(corrplot)

corrplot(cor_matrix,
```

```
        method = "color",
        tl.col = "black",
        addCoef.col = "black",
        number.cex = 0.8,
        title = "Correlation Matrix of Thyroid-Related features",
        mar = c(0, 0, 2, 0))
```



## Step 4: Model Training

### Regression

In this step, we build and evaluate predictive models to answer our regression research question. The goal is to assess whether applying machine learning can effectively predict TSH

We train **three different regression models**:

1. **Random Forest** - a robust ensemble model that captures non-linear relationships.
2. **Linear Regression** - a simple baseline model assuming linear relationships.
3. **XGBoost** - a high-performance gradient boosting model

```
# Data preparation
rf_thyroid_data <- thyroid_data_df[!is.na(thyroid_data_df$TSH), ]

# Remove all columns that contains 'measured'
rf_thyroid_data <- rf_thyroid_data[, !grepl("measured", names(rf_thyroid_data))]

# Remove the 'binaryClass' column
rf_thyroid_data <- subset(rf_thyroid_data, select = -binaryClass)

rf_thyroid_data

# Splitting the dataset into training and test sets
set.seed(42)
sample_index <- sample(1:nrow(rf_thyroid_data), 0.7 * nrow(rf_thyroid_data))

thyroid_regress_train_data <- rf_thyroid_data[sample_index, ]
thyroid_regress_test_data <- rf_thyroid_data[-sample_index, ]

# Random Forest - Training
library(randomForest)

rf_model <- randomForest(TSH ~ ., data = thyroid_regress_train_data,
                         ntree = 100, importance = TRUE)
rf_preds_regress_train <- predict(rf_model, thyroid_regress_train_data)
rf_preds_regress_test <- predict(rf_model, thyroid_regress_test_data)

# Linear Regression - Training
lm_model <- lm(TSH ~ ., data = thyroid_regress_train_data)
lm_preds_regress_train <- predict(lm_model, thyroid_regress_train_data)
lm_preds_regress_test <- predict(lm_model, thyroid_regress_test_data)

# XGBoost - Training
library(xgboost)
```

```
train_matrix <- xgb.DMatrix(data = as.matrix(thyroid_regress_train_data
                            [, -which(names(thyroid_regress_train_data) == "TSH")]),
                             label = thyroid_regress_train_data$TSH)

test_matrix <- xgb.DMatrix(data = as.matrix(thyroid_regress_test_data
                           [, -which(names(thyroid_regress_test_data) == "TSH")]),
                           label = thyroid_regress_test_data$TSH)

xgb_model <- xgboost(data = train_matrix,
                     label = thyroid_regress_train_data$TSH,
                     objective = "reg:squarederror",
                     nrounds = 100,
                     verbose = 0)

xgb_preds_regress_train <- predict(xgb_model, train_matrix)
xgb_preds_regress_test <- predict(xgb_model, test_matrix)
```

### Classification

In this step, we build a classification model to answer our classification research question. The goal is to predict whether an individual has thyroid dysfunction (binaryClass) based on clinical and demographic features.

We use the **Random Forest Classifier** to train the model and evaluate its performance using metrics such as accuracy, sensitivity, specificity, and confusion matrix.

To address the class imbalance in the dataset, **SMOTE** is applied to the training set, ensuring that the model is **not biased toward the majority class**.

```
# Data preparation
thyroid_data_df$binaryClass <- as.factor(thyroid_data_df$binaryClass)

# Remove all columns that contains 'measured'
clf_thyroid_data <- thyroid_data_df[, !grepl("measured", names(thyroid_data_df))]

# Remove TSH as its also the target variable
clf_thyroid_data <- clf_thyroid_data[, setdiff(names(clf_thyroid_data), "TSH")]

clf_thyroid_data

# Splitting the dataset into training and test sets
set.seed(42)
sample_index <- sample(1:nrow(clf_thyroid_data), 0.7 * nrow(clf_thyroid_data))

thyroid_classify_train_data <- clf_thyroid_data[sample_index, ]
thyroid_classify_test_data <- clf_thyroid_data[-sample_index, ]
```

```r
# Apply SMOTE to balance class distribution by oversampling class 1 (Negative Thyroid)
library(smotefamily)

smote_result <- SMOTE(X = thyroid_classify_train_data
                      [, -which(names(thyroid_classify_train_data) == "binaryClass")],
                      target = thyroid_classify_train_data$binaryClass,
                      K = 5)
smote_classify_train <- smote_result$data
smote_classify_train$binaryClass <- as.factor(smote_classify_train$class)
smote_classify_train$class <- NULL

# Random Forest - Training
library(randomForest)

rf_clf <- randomForest(binaryClass ~ ., data = smote_classify_train,
                       ntree = 100, importance = TRUE)
rf_preds_classify_train <- predict(rf_clf, thyroid_classify_train_data)
rf_preds_classify_test <- predict(rf_clf, thyroid_classify_test_data)
```

# Step 5: Model Evaluation

### Regression

In this step, each model were trained on a subset of the dataset (training set) and evaluated on unseen data (test set) to assess generalizability.

Key evaluation metrics include:

- **R-squared (R2)** - proportion of variance in TSH explained by the model.
- **Root Mean Squared Error (RMSE)** - average prediction error magnitude.

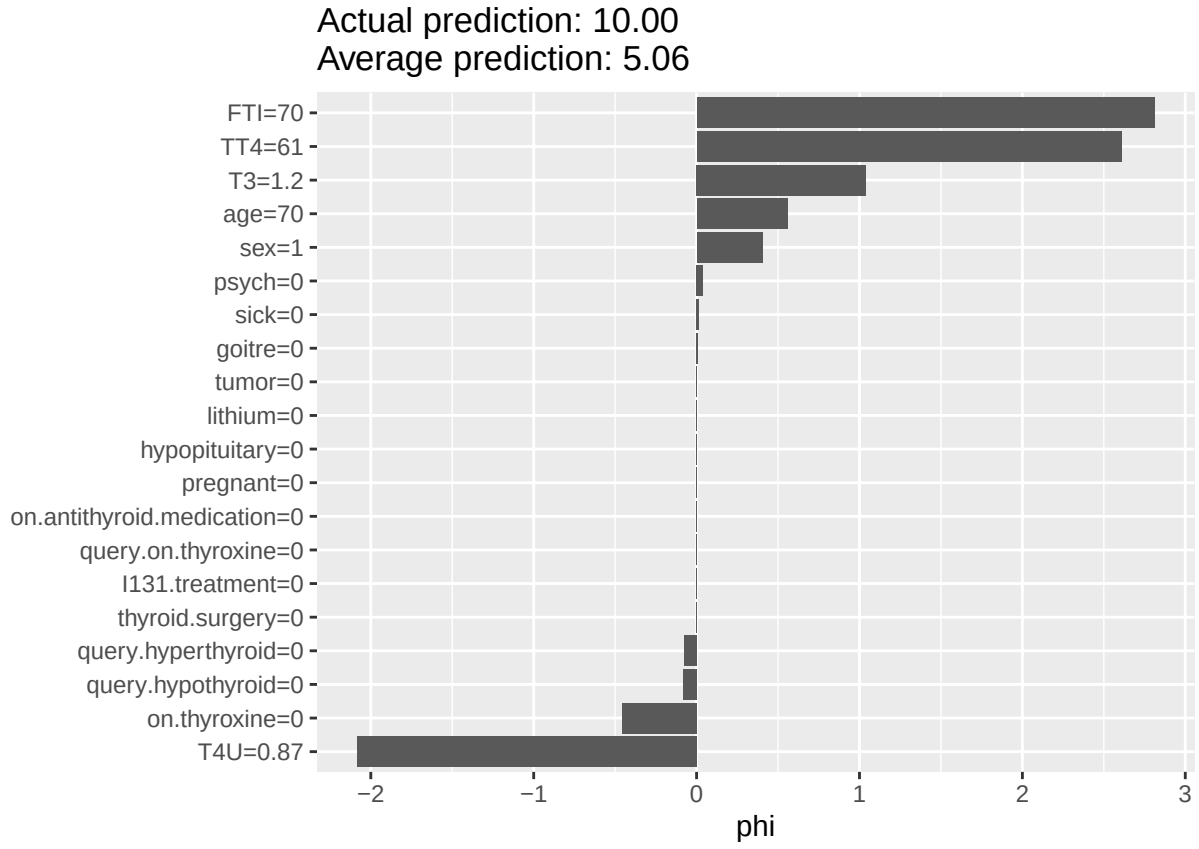
Finally, we use **SHAP (Shapley Additive Explanations)** to interpret the predictions of the best-performing model, **identifying which features most influence TSH levels**.

```
library(Metrics)

r2_score <- function(actual, predicted) {
  cor(actual, predicted)^2
}

results_df <- data.frame(
  Model = c("Random Forest", "Linear Regression", "XGBoost"),
  R_squared_Train = round(c(
    r2_score(thyroid_regress_train_data$TSH, rf_preds_regress_train),
    r2_score(thyroid_regress_train_data$TSH, lm_preds_regress_train),
    r2_score(thyroid_regress_train_data$TSH, xgb_preds_regress_train)
  ), 3),
  R_squared_Test = round(c(
    r2_score(thyroid_regress_test_data$TSH, rf_preds_regress_test),
    r2_score(thyroid_regress_test_data$TSH, lm_preds_regress_test),
    r2_score(thyroid_regress_test_data$TSH, xgb_preds_regress_test)
  ), 3),
  RMSE = round(c(
    rmse(thyroid_regress_test_data$TSH, rf_preds_regress_test),
    rmse(thyroid_regress_test_data$TSH, lm_preds_regress_test),
    rmse(thyroid_regress_test_data$TSH, xgb_preds_regress_test)
  ), 3)
)
```

```
##               Model R_squared_Train R_squared_Test   RMSE
## 1     Random Forest           0.784          0.598 17.110
## 2 Linear Regression           0.112          0.185 23.150
## 3           XGBoost           0.997          0.592 17.135
```

```
library(iml)
```

```r
# Remove TSH from features
X <- thyroid_regress_test_data[, setdiff(names(thyroid_regress_test_data), "TSH")]
y <- thyroid_regress_test_data$TSH

# Create iml Predictor object
predictor <- Predictor$new(
  model = rf_model,
  data = X,
  y = y
)

# Choose a single observation to interpret
shap <- Shapley$new(predictor, x.interest = X[1, ])
plot(shap)
```



### Classification

In this step, the classification model selected was trained on a subset of the dataset (training set) and evaluated on unseen data (test set) to assess its ability to generalize to new cases.

Key evaluation metrics include:

- **Accuracy** - the overall proportion of correct predictions.
- **Sensitivity (Recall)** - the model's ability to correctly identify individuals with thyroid dysfunction.
- **Specificity** - the ability to correctly identify normal individuals.
- **Balanced Accuracy** - average sensitivity and specificity, useful for imbalanced data.

To enhance interpretability, we also used **SHAP (Shapley Additive Explanations)** to understand how individual features contribute to the classification decisions. This helps **identify the most influential clinical and demographic features** in classifying thyroid dysfunction.

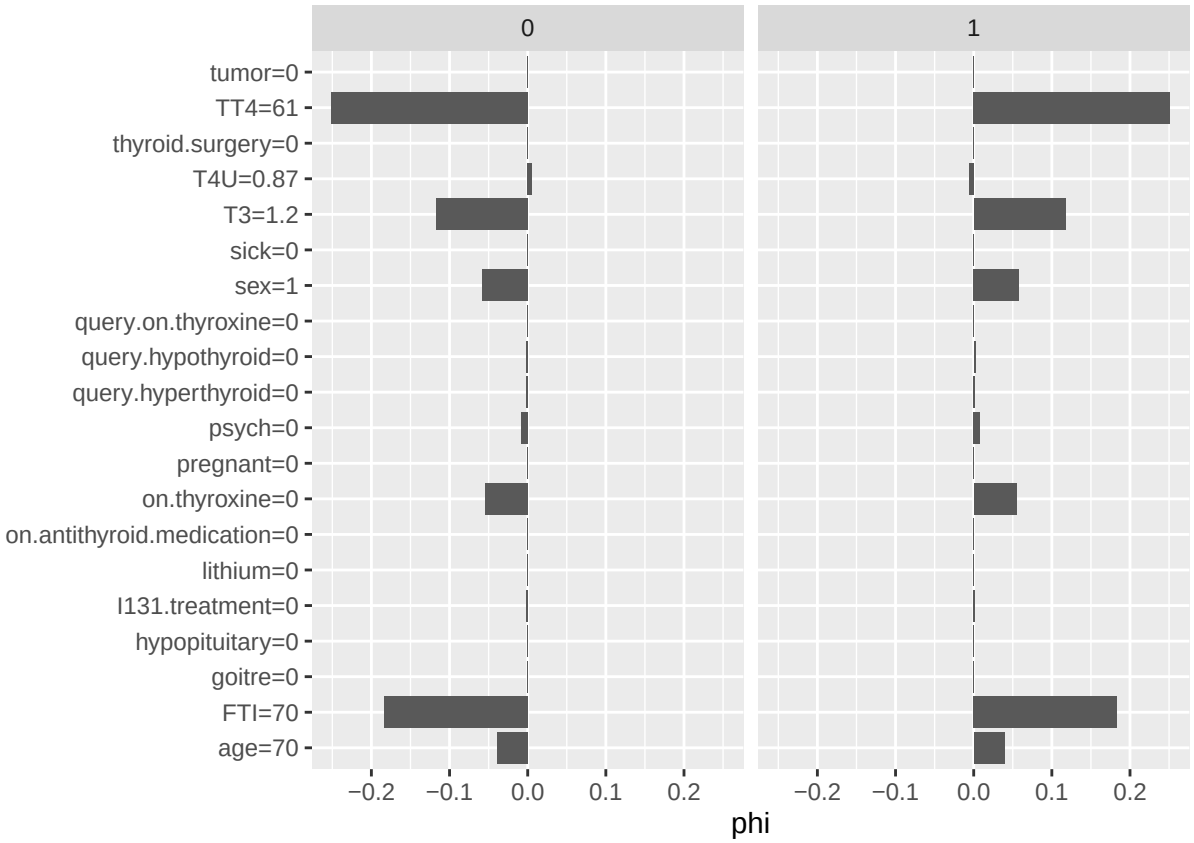
```
library(caret)

# Confusion matrix
conf_mat <- confusionMatrix(rf_preds_classify_test, thyroid_classify_test_data$binaryClass)
print(conf_mat)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 972  59
##          1  45  37
##
##                Accuracy : 0.9066
##                  95% CI : (0.8879, 0.923)
##     No Information Rate : 0.9137
##     P-Value [Acc > NIR] : 0.8187
##
##                   Kappa : 0.3653
##
##  Mcnemar's Test P-Value : 0.2024
##
##             Sensitivity : 0.9558
##             Specificity : 0.3854
##          Pos Pred Value : 0.9428
##          Neg Pred Value : 0.4512
##              Prevalence : 0.9137
##          Detection Rate : 0.8733
##    Detection Prevalence : 0.9263
##       Balanced Accuracy : 0.6706
##
##        'Positive' Class : 0
##
```

```r
cat("Test Accuracy:", round(mean(rf_preds_classify_test == thyroid_classify_test_data$binaryClass), 3), "\n")
```

```
## Test Accuracy: 0.907
```

```r
library(iml)

# Remove TSH from features
X <- thyroid_classify_test_data[, setdiff(names(thyroid_classify_test_data), "binaryClass")]
y <- thyroid_classify_test_data$binaryClass

# Create iml Predictor object
predictor <- Predictor$new(
  model = rf_clf,
  data = X,
  y = y,
  type = "prob"
)

# Choose a single observation to interpret
shap <- Shapley$new(predictor, x.interest = X[1, ])
plot(shap)
```



# Step 6: Interpretation of Results

### Result Intepretation - Regression

**Key Takeaways:**

**Model Summary:**

- **Random Forest** and **XGBoost** both performed **reasonably well on test data** (**R2 approximately 0.60**), explaining about 60% of the variance in TSH levels.
- **Linear Regression performed poorly**, indicating a **non-linear relationship** between predictors and TSH.
- **XGBoost has the best fit on training data**, but a similar test performance to Random Forest, suggesting **possible overfitting**.

**Feature Summary:**

- The top contributors to TSH prediction include **FTI**, **TT4**, **T3**, **age**, and **sex**.
- Features like **T4U** and **on.thyroxine** negatively influenced the prediction for this individual.
- This matches **clinical expectations** - hormone levels and demographics **strong indicators** of thyroid health.

Overall, it is **feasible to predict an individual's TSH level** using machine learning models trained on clinical and demographic data. The models demonstrate **good predictive performance** and **feature interpretability**, making them promising for **clinical use** and **early detection** of thyroid issues.

### Result Interpreation - Classification

**Key Takeaways:**

**Model Summary:**

- A **Random Forest Classifier** was used to predict binaryClass (0 = Positive Thyroid, 1 = Negative Thyroid)
- The model achieved a **test accuracy of 90.7%**, which reflects **strong overall performance**.
- However, due to **class imbalance**, there is a trade-off between **sensitivity** and **specificity**:
- **Sensitivity (Recall for Thyroid cases)**: **0.9558** -> Excellent at identifying individuals with thyroid dysfunction.
- **Specificity (Recall for Normal cases)**: **0.3854** -> Poor at identifying truly normal individuals.
- **Balanced Accuracy**: **0.6706** -> Moderate overall performance when accounting for imbalance.
- **Kappa**: **0.3653** -> Indicates **fair agreement** between predicted and actual classes.

**Feature Summary:**

- **FTI**, **TT4**, **T3** - Hormone levels relevant to thyroid function.
- **on.thyroxine** - Medication use is a strong signal for thyroid issues
- **age**, **sex** - Demographic features also contribute meaningfully

Overall, it is **feasible to classify individuals as having thyroid dysfunction or being in a normal state** using the features in the dataset. The Random Forest model achieved high overall accuracy and strong sensitivity for detecting thyroid conditions.

However, the **low specificity** suggests **room for improvement** in detecting truly negative thyroid individuals. To further improve the model:

- Experiment with other classifiers with threshold tuning (e.g., XGBoost, Logistic Regression).
- Perform threshold optimization to better balance sensitivity and specificity.

# Conclusion

Thyroid dysfunction has shown to be a prevalent health issue affecting a significant portion of the population. In this project, we addressed two primary questions: whether TSH levels can be accurately predicted, and whether individuals can be classified as having thyroid dysfunction using clinical and demographic features. Based on our findings, machine learning techniques show strong potential in supporting early diagnosis and clinical decision-making for thyroid health. Despite limitations such as data imbalance and missing values, the results highlight the value of predictive modeling in enhancing clinical screening. These models can help identify thyroid conditions more efficiently, reducing the wait time for lab results. Future work may focus on integrating these models into clinical workflows and validating them across larger, more diverse populations.

# References


1. Taylor, P. N., Albrecht, D., Scholz, A., Gutierrez-Buey, G., Lazarus, J. H., Dayan, C. M., & Okosieme, O. (2018). Global epidemiology of hyperthyroidism and hypothyroidism. Nature Reviews Endocrinology, 14(5), 301–316. https://doi.org/10.1038/nrendo.2018.18.
2. MedlinePlus. (2023). TSH (Thyroid-stimulating hormone) test. U.S. National Library of Medicine. https://medlineplus.gov/lab-tests/tsh-thyroid-stimulating-hormone-test/
3. Yasserhessein. (2022). Thyroid Disease Data Set. Kaggle. Retrieved from https://www.kaggle.com/datasets/yasserhessein/thyroid-disease-data-set