STA314 Project

2024-12-02

1. Identifying Alzheimer's Early: A Statistical Model for Public Awareness and Timely Diagnosis

Alzheimer's Disease dataset

Aaron Tran - 1006071144, Fuzo Yoshikawa (1006239982), Adeline Wang (1007327135)

Project 27 - (ranking here) - (Prediction score here)

2. Problem Statement

Dementia is an umbrella term encompassing a variety of symptoms resulting from brain disorders, which impact memory, cognition, behavior, and emotions. Alzheimer's disease, the most common form of dementia, accounts for more than half of all cases.

Dementia faces a significant challenge with underdiagnosis. According to Alzheimer's disease international, in high-income countries, only 20-50% of cases are diagnosed, while in low-income countries, this figure rises to around 90%. Globally, approximately 75% of people with dementia remain undiagnosed.

While Alzheimer's disease is currently incurable, an early diagnosis is far from futile. The 2011 World Alzheimer Report highlighted the benefits of early diagnosis and support. Early diagnosis enables families to better prepare to support their loved ones, providing time to process their initial grief and fostering a sense of empowerment. Most individuals with early-stage dementia want to know their diagnosis. Early therapeutic interventions can delay the onset of Alzheimer's symptoms, delaying the need for long-term care, and improving the patient's quality of life.

Our research aims to identify symptoms that help people recognize the early signs of Alzheimer's and spread awareness to encourage individuals to seek timely diagnosis and treatment for their loved ones. By promoting early recognition, we hope to empower families to take proactive steps, improving outcomes and enhancing the quality of life for those affected.

3. Statistical Analyses

The dataset we are using includes information on 2,149 patients. Of these, 1,504 will be used to train our model, while 646 will serve as the validation set. The dataset encompasses a diverse range of variables, including demographic details, lifestyle factors, medical history, clinical measurements, cognitive and functional assessments, and reported symptoms.

The first step of our data cleaning would be checking for any variables that are NAs, and how to proceed with any observations that contain then. Fortunately our data did not contain any.

The next step involved removing variables that were unlikely to contribute meaningfully to our study's goal. Specifically, we excluded the MMSE (Mini-Mental State Examination score), FunctionalAssessment (Functional Assessment score), and ADL (Activities of Daily Living score). The MMSE and Functional Assessment were removed because they require evaluation by a licensed physician, which does not align with our objective of empowering individuals to recognize potential early signs of Alzheimer's and seek a physician's assessment. Similarly, the ADL score was excluded due to its subjective nature and its reliance on clinical evaluation. While similar concerns could be raised for variables such as diet quality and sleep

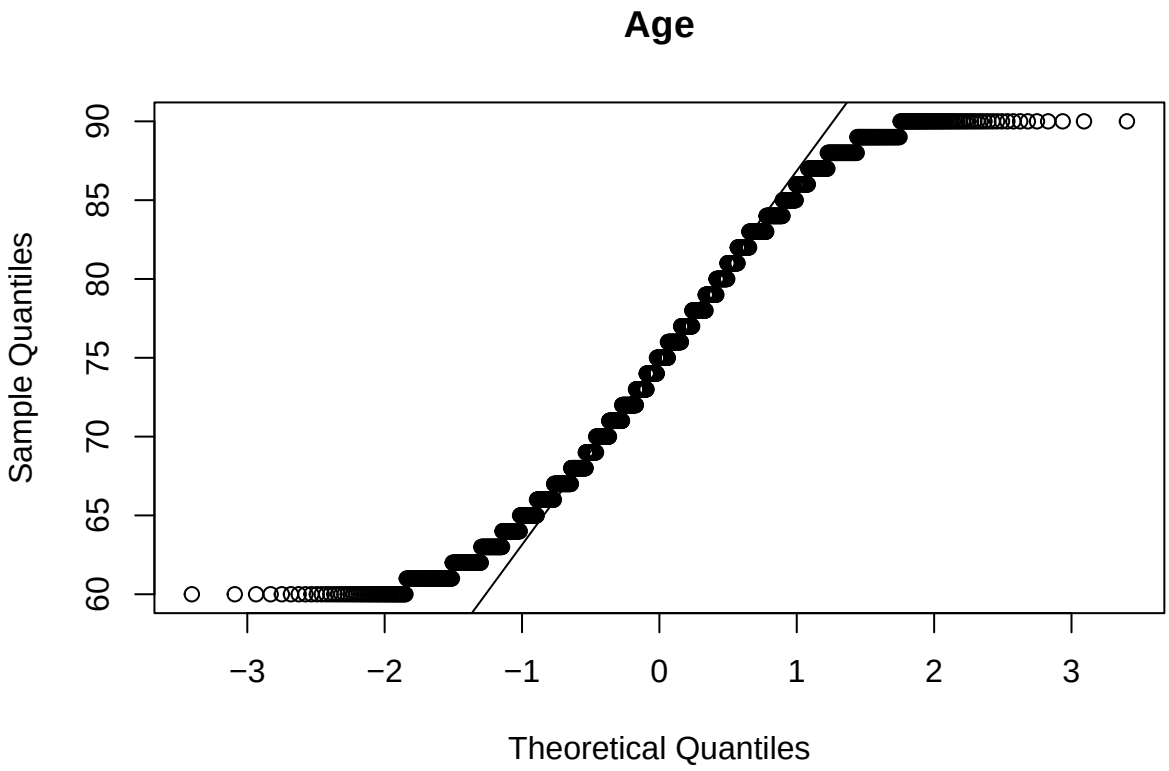
quality, we retained these as they can be measured more objectively through metrics like hours of sleep or specific dietary choices making them more relevant to self-assessment. We also chose to retain clinical measurements. Although this may seem counterintuitive for a self-assessment-focused study, these measurements are often easily accessible in everyday life. Small heart monitors, for example, are increasingly common in households, and blood pressure measurements can be readily obtained at drugstores. Additionally, cholesterol measurements are typically available from annual blood test results, making these variables practical for inclusion.

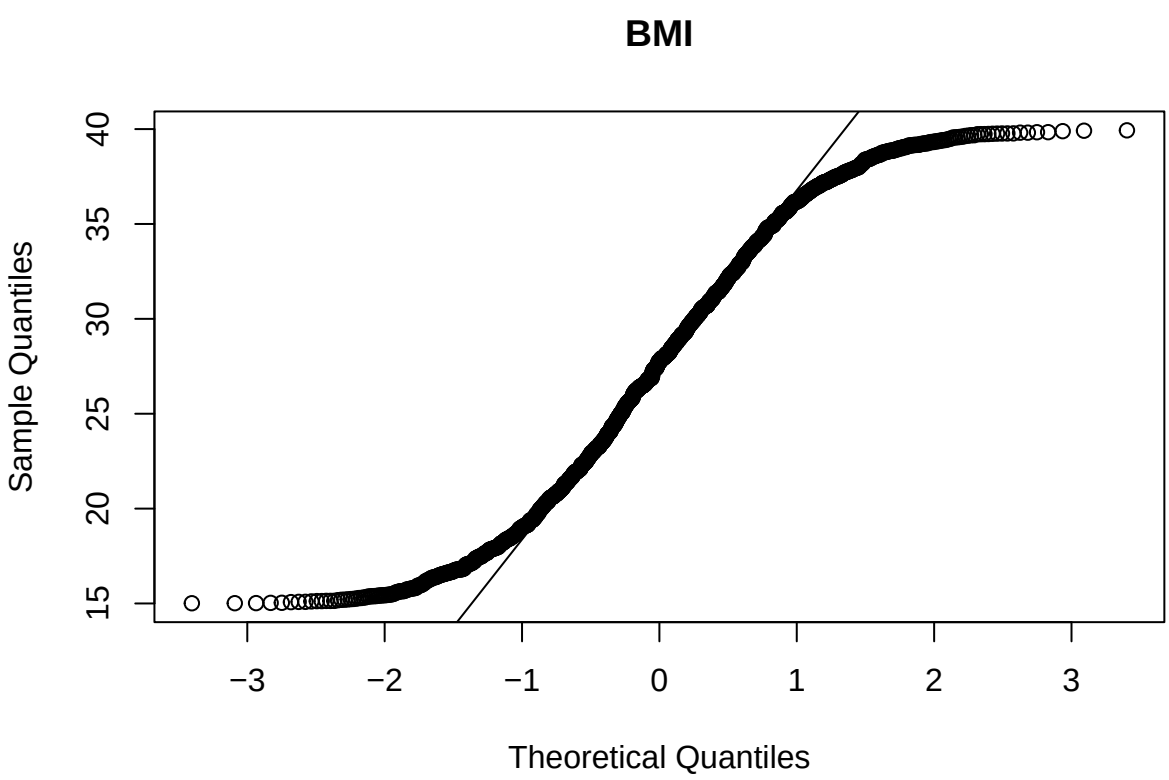
The following step was one-hot encoding any categorical variables we had. This refers to the process of converting categorical variables into multiple binary variables, each representing one unique category. For example, we applied this to the ethnicity variable, which had values ranging from 0 to 3, where 0 represented Caucasians, 1 represented African Americans, and so on. We created separate variables, such as Eth_Caucasian, which equals 1 if ethnicity is 0, and 0 otherwise. Without doing so the model would assume ethnicity was a continous variable, and would not represent each category properly.

Our study focuses on the interpretability of the model and its results, which excludes methods like random forests and boosting. We are also not considering the KNN Classifier for a similar reason. While the interpretability of the classifier is straightforward—for example, 'You were classified as having Alzheimer's because 80% of people with these symptoms had Alzheimer's' it does not allow us to identify the specific factors at play, which is the main focus of our analysis.

The same applies to SVM and QDA, which are not only non-linear but also create decision boundaries. Although LDA is more interpretable because it is a linear combination of variables, it still relies on several assumptions to perform well. One of these assumptions, multivariate normality, is violated by our data. The results of the Henze-Zirkler test for multivariate normality confirm that it does not hold. Using QQ plots to visualize a distribution of age and BMI, we observe that if the data were normally distributed, the points would align closely with the straight reference line. However, the points deviate significantly from this line, indicating that the distributions of age and BMI are not normal.

```
##            Test       HZ p value MVN
## 1 Henze-Zirkler 1.020252       0  NO
```


Age
Sample Quantiles
60
65
70
75
80
85
90
−3
−2
−1
0
1
2
3
Theoretical Quantiles



A possible solution to this would be transforming our data to be normally distributed however, this would make the results even more difficult to interpret. For example, if we apply a log transformation to blood pressure data to make it normally distributed, it might become harder for our target audience to understand, as the transformed values are not directly interpretable in terms of the original blood pressure units.

When checking other assumptions, such as homogeneity of covariance using the Box's M-test, we observe that the dataset fails this assumption, as indicated by the extremely low p-value. This issue can be addressed by selecting a set of variables that exhibit lower correlation during variable selection. For the Durbin-Watson test, the value is close to 2, along with a high p-value, this suggests no significant autocorrelation, supporting the assumption of independence.

```
##
##  Box's M-test for Homogeneity of Covariance Matrices
##
## data:  numeric_data
## Chi-Sq (approx.) = 166.53, df = 120, p-value = 0.003199

##
##  Durbin-Watson test
##
## data:  model
## DW = 2.0078, p-value = 0.5603
## alternative hypothesis: true autocorrelation is greater than 0
```

Upon checking the correlation in our data, we observe little to no multicollinearity, with the highest correlation being only 0.0823. This low level of correlation makes logistic regression a compelling choice for our model. It also has the benefit of being more interpretable than LDA.

```
## [1] 0.08231193
```

To fit our logistic and LDA model we first performed lasso for our variable selection, choosing the lambda with the lowest cross-validation error rate. The reason we chose lasso is because it tends to yield sparse solutions. Less variables not only make the model more interpretable but also make it more likely to satisfy the assumptions we previously failed such as homogeneity of covariance and multivariate normality. Lasso left us with twelve variables, three of which are continuous: BMI, SleepQuality, and CholesterolTriglycerides.

Revisting our previous assumptions, we observe that while multivariate normality still does not hold, homogeneity of covariance now does. This indicates that both models will perform reasonably well given the conditions.

```
##            Test      HZ p value MVN
## 1 Henze-Zirkler 8.07515       0  NO
```

```
##
##  Box's M-test for Homogeneity of Covariance Matrices
##
## data:  lasso_selected_variables
## Chi-Sq (approx.) = 1.4052, df = 6, p-value = 0.9655
```

Our next choice of algorithm is a decision tree. Not only are they highly interpretable and easy to follow for laypeople, but they also do not have as many assumptions as other algorithms. It is well-suited for our study's goal, as the decision tree can easily be converted into a diagram or poster that people can follow to assess their risk for Alzheimer's. This requires much less explanation and makes the results more interpretable. It aligns perfectly with our goal of encouraging individuals to see a doctor for an official diagnosis. Furthermore, one of the benefits of using a decision tree is the ability to apply a loss matrix, assigning a higher penalty to false negatives. This approach encourages the tree to select variables and classifications that improve the recall rate.

The metrics we will use to evaluate our models performance are accuracy and recall. We will derive these metrics from the 10-fold cross-validation error of our models. Afterward, we will train our model on the entire training data and then evaluate it on the test set as an additional measure of accuracy. Accuracy will be used as an overall metric to evaluate our models predicting ability. However, we will also be focusing on recall as it is crucial for our study goal. Recall is the ratio of true positives to the sum of true positives and false negatives, and in our context, it is vital because the cost of false negatives where individuals with Alzheimer's are not identified and fail to seek medical help is high. On the other hand, false positives, where individuals are referred to a doctor unnecessarily, are far less harmful, as it only leads to additional consultations rather than missed diagnoses. By prioritizing recall, we aim to ensure that as many at-risk individuals as possible are identified and encouraged to seek a diagnosis.

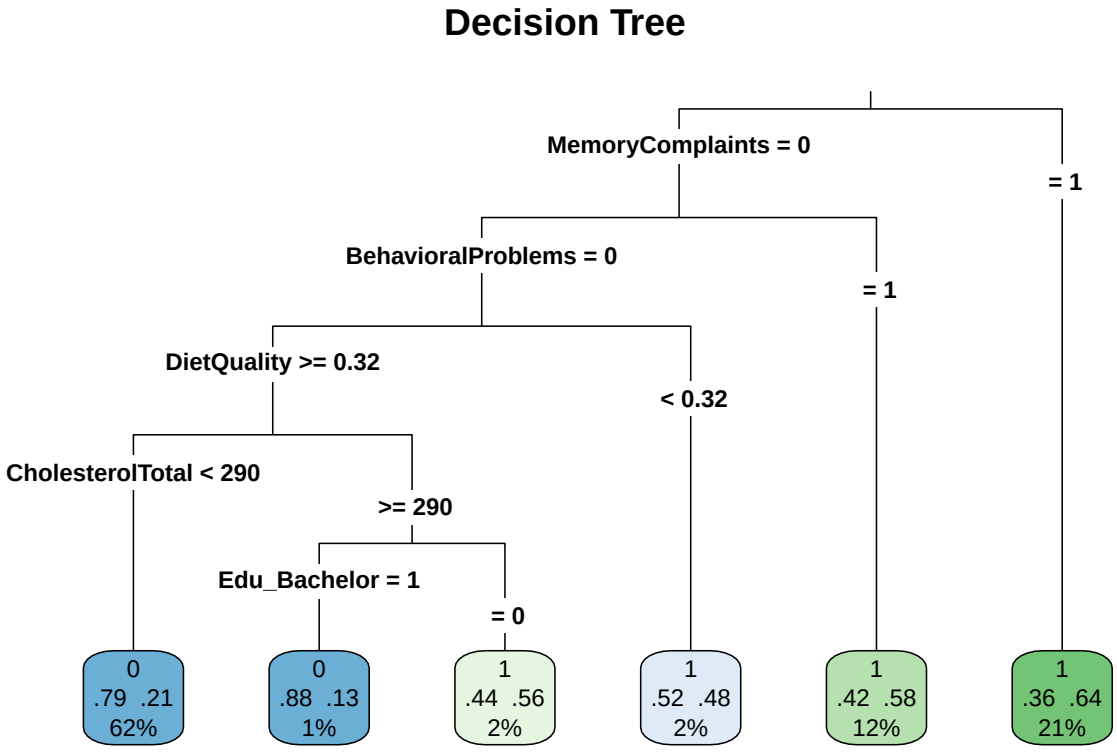
4. Results and Conclusion

From our metrics, we can see that in terms of accuracy, all the models were nearly identical. However, the decision tree performed slightly worse on the validation set compared to the other models. Where the decision tree stands out is in its recall rate, it significantly outperforms the other models. Additionally, even after applying Lasso, the other models still have 12 variables, whereas our decision tree has only 10 branches. Considering the priority of our study goal of encouraging early diagnosis for those at risk of Alzheimer's false negatives, as mentioned earlier, are particularly problematic, whereas false positives pose minimal risk. As mentioned previously, our study's goal of encouraging early diagnosis for those at risk makes a high rate of false negatives catastrophic, whereas false positives do not pose the same risk The decision tree's high recall rate makes it well-suited for our goals, as it allows individuals to easily assess whether they might be at risk. Its ability to flag more potential cases aligns with our goal of promoting early diagnosis. Its lower complexity, higher recall, and easier interpretability made it our model of choice.

Table 1: Cross-Validation and Validation Metrics by Model

Model	Metrics		
	CV Accuracy	CV Recall	Validation Accuracy
Logistic Regression	0.7240691	0.5338346	0.73273
LDA	0.7107713	0.6218487	0.73873
Decision Tree	0.7220504	0.8044919	0.69669

Observing the decision tree, we can see that memory complaints and behavior problems were the strongest indicators of Alzheimer's, accounting for the majority of the positive classifications. However, as we move further down the tree, the classifications become less conclusive. For example, individuals with a poor diet or high cholesterol levels were classified as positive, despite these nodes containing an almost equal ratio of positive and negative Alzheimer's patients. We can observe that if someone has no memory complaints, no behavioral problems, maintains a good diet, and has a cholesterol total under 290, they are classified as negative. Similarly, individuals with a high cholesterol total are classified as negative if they hold a bachelor's degree. We can understand why our recall was so high, as the terminal nodes contain a small number of false positives making up less than a third of the total. This is ideal, as we want to avoid misclassifying positive patients.



To conclude, our model suggests memory complaints and behavioral problems are among the strongest indicators of Alzheimer's disease in older adults. Even if these aren't present, factors like a poor diet or high cholesterol make seeking an official diagnosis worthwhile. However, if you exhibit none of the mentioned symptoms or have a high cholesterol total but hold a bachelor's degree, you may be at a lower risk. These

findings satisfy our study goal because they can be easily understood or made highly interpretable for the general public. A simple awareness campaign using our findings could involve posters with messages such as, 'Are you a senior experiencing memory trouble or changes in behavior? Do you have high cholesterol or a poor diet? You should seek a diagnosis for Alzheimer's.' Furthermore, by focusing on high recall, our models reduce the potential harm to the public, as false negatives are less likely.

5. Discussion

One limitation of our work is that the dataset lacks depth in certain variables, and some of these variables can be subjective. For example, behavioral problems are highly subjective, and individuals might deny that their memory has worsened, especially if self-diagnosing. Diet quality is also not well-defined. As a result, our model's accuracy can be influenced by self-diagnosis or the diagnosis of a loved one. If variables like diet quality were more detailed with objective measures, such as the frequency of meat or dairy consumption and junk food intake, our model might perform better but also more resilient to subjective diagnosis.

Another limitation of our model is the absence of cognitive and functional assessment scores, as these are clinical measures typically administered with the aid of a physician. Given our study's goal of enabling the general public to self-diagnose, replacing these clinical assessments with a memory game or similar task that can be done on a smartphone would likely improve the accuracy of our model while remaining accessible to the public.

6. References

https://www.alzint.org/about/dementia-facts-figures/dementia-statistics/

https://www.alzint.org/resource/world-alzheimer-report-2011/

7. Code

```
install.packages("ggplot2")
install.packages("Amelia")
install.packages("dplyr")
install.packages("tidyverse")
install.packages("GGally")
install.packages("leaps")
install.packages("MASS")
install.packages("glmnet")
install.packages("Rcpp")
install.packages("caret")
install.packages("e1071")#SVM
install.packages("car")#assumptions
install.packages("lmtest")#assumptions
install.packages("MVN") #assumptions
install.packages("biotools")#assumptions
install.packages("rpart") #DT
install.packages("rpart.plot") #DT
```

```
library(ggplot2)
library(Amelia)
library(tidyverse)
library(GGally)
library(leaps)
library(MASS)
```

```r
library(glmnet)
library(caret)
library(e1071)
library(car)
library(lmtest)
library(MVN)
library(biotools)
library(rpart)
library(rpart.plot)
```

```r
dt <- read.csv("train.csv")
dt1 <- read.csv("train.csv")
dt.test <- read.csv("test.csv")

#Hot one encoding factors

dt$Eth_Caucasian <- ifelse(dt$Ethnicity == 0, 1, 0) #0: No, 1:yes
dt$Eth_Afr_Amer <- ifelse(dt$Ethnicity == 1, 1, 0)
dt$Eth_Asian <- ifelse(dt$Ethnicity == 2, 1, 0)
dt$Eth_Other <- ifelse(dt$Ethnicity == 3, 1, 0)

dt$Edu_None <- ifelse(dt$EducationLevel == 0, 1, 0)
dt$Edu_Highschool <- ifelse(dt$EducationLevel == 1, 1, 0)
dt$Edu_Bachelor <- ifelse(dt$EducationLevel == 2, 1, 0)
dt$Edu_Higher <- ifelse(dt$EducationLevel == 3, 1, 0)

dt.test$Eth_Caucasian <- ifelse(dt.test$Ethnicity == 0, 1, 0) #0: No, 1:yes
dt.test$Eth_Afr_Amer <- ifelse(dt.test$Ethnicity == 1, 1, 0)
dt.test$Eth_Asian <- ifelse(dt.test$Ethnicity == 2, 1, 0)
dt.test$Eth_Other <- ifelse(dt.test$Ethnicity == 3, 1, 0)

dt.test$Edu_None <- ifelse(dt.test$EducationLevel == 0, 1, 0)
dt.test$Edu_Highschool <- ifelse(dt.test$EducationLevel == 1, 1, 0)
dt.test$Edu_Bachelor <- ifelse(dt.test$EducationLevel == 2, 1, 0)
dt.test$Edu_Higher <- ifelse(dt.test$EducationLevel == 3, 1, 0)

dt <- dt[,c(-4,-5,-24, -25,-28,-34,-35)]
dt.test <- dt.test[, c(-4,-5,-24, -25,-28,-34)]
dt$Diagnosis <- dt1$Diagnosis
```

```r
#variable selection
#-------------------------------------------------------------------------------#
select_features1 = c(1:36)
select_features2 = c(1:36)
alpha = 1    # Lasso (alpha = 1), ridge (alpha = 0), combination (0 < alpha < 1)
set.seed(100629982)
#-------------------------------------------------------------------------------#

x1 <- as.matrix(dt[,select_features1])
y1 <- dt$Diagnosis
x1.test <- as.matrix(dt.test[,select_features1])
# Use cross-validation to find the best lambda for Lasso regularization
kfold_cv <- cv.glmnet(x1, y1, family = "binomial", alpha = alpha)
```

```r
best_lambda <- kfold_cv$lambda.min #Optimal lambda value (using cv)
#shrinkage model selection
best_mod1 <- glmnet(x1, y1, family = "binomial", alpha = alpha, lambda = best_lambda)

#Data for Discriminant analysis
chosen_features1 <- which(coef(best_mod1) != 0)[-1]
x_selected1 <- x1[, chosen_features1-1, drop = FALSE]
x_selected1.test <- as.data.frame(x1.test[, chosen_features1-1, drop = FALSE])
train_data_selected1 <- as.data.frame(x_selected1)
train_data_selected1$target <- y1

x2 <- dt[,select_features2]
y2 <- dt$Diagnosis
x2.test <- as.matrix(dt.test[,select_features2])

#Best subset selection
best_subset <- regsubsets(y ~ ., data = cbind(x2, y = y2), nvmax = ncol(x2))
summary_best <- summary(best_subset)
#best models in each subset level features
best_adjr2_model <- which.max(summary_best$adjr2) #best model based on R^2
best_rss_model <- which.min(summary_best$rss) #best model based on RSS
#overall best model
which(summary_best$which[best_adjr2_model, ])[-1]
as.numeric(which(summary_best$which[best_adjr2_model, ]))[-1]

chosen_features2 <- as.numeric(which(summary_best$which[best_adjr2_model, ]))[-1]
x_selected2 <- x2[, chosen_features2-1, drop = FALSE]
x_selected2.test <- as.data.frame(x2.test[, chosen_features2-1, drop = FALSE])
train_data_selected2 <- as.data.frame(x_selected2)
train_data_selected2$target <- y2

names(train_data_selected1)
names(train_data_selected2)
```

```r
#define function for later
fn_fp <- function(a, b) {
  #FN: predicts no Alzheimer's risk for a patient who actually has early-stage symptoms or is at risk
  #FP: predicts Alzheimer's risk for a patient who is actually not at risk
  precision <- a
  recall <- b
  actual_positives <- 100  # TP + FN

  # Calculate TP and FN
  TP <- recall * actual_positives
  FN <- actual_positives - TP

  # Calculate FP
  predicted_positives <- TP / precision  # TP + FP
  FP <- predicted_positives - TP
  cat("TP:", TP, ", FN:", FN, ", FP:", FP, "\n")
}
```

```r
# Logistic regression Model
set.seed(100629982)

#EDA (linearity assumption <=> pval > 0.05)
eda1.1_linearity <- boxTidwell(target ~ BMI + SleepQuality + CholesterolTriglycerides, data = train_data
eda1.1_linearity

#EDA (multicollinearity assumption <=> vif < 5)
eda1.1 <- glm(target ~ ., data = train_data_selected1, family = "binomial")
eda1.1_mult.cor <- vif(eda1.1)
eda1.1_mult.cor

#EDA (Independence assumption <=> pval > 0.05)

eda1.1_indep <- dwtest(eda1.1)
eda1.1_indep

print("linearity holds, multicollinearity holds, independence holds")

alpha1 = 1

kfold_cv1 <- cv.glmnet(x1, y1, family = "binomial", alpha = alpha1)
best_lambda1 <- kfold_cv1$lambda.min #Optimal lambda value (using cv)

#logistic regression
mod1.1 <- glmnet(x1, y1, family = "binomial", alpha = alpha1, lambda = best_lambda1)
prob1 <- predict(mod1.1, newx = x1.test, s = best_lambda1, type = "response")
y1.1.predict <- ifelse(prob1 > 0.5, 1, 0) #with threshold 0.5
opt1.1.submission <- data.frame(dt.test[,1], y1.1.predict)
colnames(opt1.1.submission) <- c("PatientID", "Diagnosis")
table(y1.1.predict)



if (all(select_features1 %in% c(1:36))){ #<- make sure to change here if you change the 'selected_featu
  if(alpha == 0){
    write.csv(opt1.1.submission, "opt1.submission(logit_alpha1=0).csv", row.names = FALSE) #
  } else if (alpha ==1){
    write.csv(opt1.1.submission, "opt1.submission(logit_alpha1=1).csv", row.names = FALSE)
  } else {
    write.csv(opt1.1.submission, "opt1.submission(logit_alpha1=Both).csv", row.names = FALSE)
  }
}

# Statistical analysis (run cv thru selected dataset)
n <- length(dt$PatientID)
X <- as.data.frame(x_selected1)
y <- factor(y1)

folds <- createFolds(y, k = 2, list = TRUE, returnTrain = TRUE)

# Initialize a vector to store performance metrics
log_accuracy <- numeric(length(folds))
```

```r
# Perform cross-validation
for (i in seq_along(folds)) {
  # Split data into training and test sets
  train_idx <- folds[[i]]
  test_idx <- setdiff(seq_along(y), train_idx)

  train_data <- X[train_idx, ]
  test_data <- X[test_idx, ]
  train_labels <- y[train_idx]
  test_labels <- y[test_idx]

  # Train log model, predictions on test set
  log_model <- glmnet(train_data, train_labels, family = "binomial", alpha = alpha, lambda = best_lambda
  log_prob <- predict(log_model, newx = as.matrix(test_data), s = best_lambda, type = "response")
  log_pred <- ifelse(log_prob[,1] > 0.5, 1, 0)
  log_accuracy[i] <- mean(log_pred == test_labels)
}
print("for loop finished")

log_acc_mean <- mean(log_accuracy)
log_conf_matrix <- table(Predicted = log_pred, Actual = test_labels)
log_precision <- log_conf_matrix[2, 2] / sum(log_conf_matrix[2, ])
log_recall <- log_conf_matrix[2, 2] / sum(log_conf_matrix[, 2])
log_f1 <- 2 * (log_precision * log_recall) / (log_precision + log_recall)


# Mod1.2 LDA

#EDA (mulivariate normal distribution assumption <=> pval > 0.05)
mvn1.2_data <- data.frame(train_data_selected1$BMI, train_data_selected1$SleepQuality, train_data_selec
eda1.2_mvn <- mvn(data = mvn1.2_data, mvnTest = "mardia")
eda1.2_mvn

#equal (covariance matrix assumption <=> pval > 0.05)
eda1.2_cov <- boxM(mvn1.2_data, train_data_selected1$target)
eda1.2_cov

print("MVN not hold, homogeneity covariance matrix hold, independence hold from logstic regression eda"


mod1.2 <- lda(target ~ ., data = train_data_selected1)
lda1.pred <- predict(mod1.2, newdata = x_selected1.test)
y1.2.predict <- ifelse(lda1.pred$posterior[, "1"] > lda1.pred$posterior[, "0"], 1, 0)
opt1.2.submission <- data.frame(dt.test[,1], y1.2.predict)
colnames(opt1.2.submission) <- c("PatientID", "Diagnosis")
table(y1.2.predict)

write.csv(opt1.2.submission, "opt1.submission(LDA).csv", row.names = FALSE)

# Statistical analysis (run cv thru selected dataset)
n <- length(dt$PatientID)
X <- as.data.frame(x_selected1)
y <- factor(y1)
```

```r
folds <- createFolds(y, k = 2, list = TRUE, returnTrain = TRUE)

# Initialize a vector to store performance metrics
lda_accuracy <- numeric(length(folds))

# Perform cross-validation
for (i in seq_along(folds)) {
  # Split data into training and test sets
  train_idx <- folds[[i]]
  test_idx <- setdiff(seq_along(y), train_idx)

  train_data <- X[train_idx, ]
  test_data <- X[test_idx, ]
  train_labels <- y[train_idx]
  test_labels <- y[test_idx]

  # Train LDA model, predictions on test set
  lda_model <- lda(train_labels ~ ., data = cbind(train_data, train_labels))
  lda_pred <- predict(lda_model, newdata = test_data)
  lda_accuracy[i] <- mean(lda_pred$class == test_labels)
}
print("for loop finished")
lda_acc_mean <- mean(lda_accuracy)
lda_conf_matrix <- table(Predicted = lda_pred$class, Actual = test_labels)
lda_precision <- lda_conf_matrix[2, 2] / sum(lda_conf_matrix[2, ])
lda_recall <- lda_conf_matrix[2, 2] / sum(lda_conf_matrix[, 2])
lda_f1 <- 2 * (lda_precision * lda_recall) / (lda_precision + lda_recall)


#Decision Tree

#Loss matrix penalizing false negatives
loss_matrix <- matrix(c(0, 1, 2, 0), nrow = 2, byrow = TRUE)

cvTrainingSet <- dt
cvTrainingSet$Diagnosis <- factor(dt$Diagnosis, levels = c(0, 1), labels = c("Negative", "Positive"))

#Custom summary adding accuracy
custom_summary <- function(data, lev = NULL, model = NULL) {
  out <- twoClassSummary(data, lev, model)
  accuracy <- sum(data$obs == data$pred) / nrow(data)
  out <- c(out, Accuracy = accuracy)
  return(out)
}

train_control <- trainControl(method = "cv", number = 10,
                              summaryFunction = custom_summary,
                              classProbs = TRUE)
cv_model <- train(
  Diagnosis ~ .,
  data = cvTrainingSet,
  method = "rpart",
  trControl = train_control,
```

```
  parms = list(loss = loss_matrix)
)

resampling_results <- cv_model$resample
# Rename sensitivity to recall
names(resampling_results)[names(resampling_results) == "Sens"] <- "Recall"

resampling_results <- resampling_results[, c("Accuracy", "Recall")]
averaged_results <- colMeans(resampling_results)

#cv accuracy and recall
tree_accuracy <- averaged_results["Accuracy"]
tree_recall <- averaged_results["Recall"]

# Print the averaged results
print(averaged_results)

# Train the decision tree
tree_model <- rpart(
  Diagnosis ~ .,
  data = dt,
  method = "class",
  parms = list(loss = loss_matrix)
)

#Tree plot
rpart.plot(tree_model, type = 3, extra = 104, fallen.leaves = TRUE, main = "Decision Tree")

#tree models predictions for test set
tree_model.pred <- predict(tree_model, newdata = data.frame(dt.test), type = "class")
tree_predictions<- data.frame(dt.test[,1], tree_model.pred)
colnames(tree_predictions) <- c("PatientID", "Diagnosis")
table(tree_predictions)
write.csv(tree_predictions, "tree_Predictions.csv", row.names = FALSE)
```

```
#Metrics
results <- data.frame(
  Model = c("Logistic Regression", "LDA", "Decision Tree"),
  `CV Accuracy` = c(log_acc_mean, lda_acc_mean, tree_accuracy),
  `CV Recall` = c(log_recall, lda_precision, tree_recall),
  `Validation Accuracy` = c(0.73273, 0.73873, 0.69669)
)

# Print the table
print(results)
```