Alzheimer's Detection Using Handwriting Analysis

Braeden Cullen

April 29th, 2024

Contents

1 Introduction and Problem Statement

10% of all people over the age of 65 have Alzheimer's disease, and as many as 50% of people over 85 have it. The number of people with the disease doubles every 5 years beyond age 65. Alzheimer's is a progressive neurodegenerative disease that affects millions of people worldwide. It is the most common cause of dementia and is characterized by memory loss, cognitive decline, and behavioral changes. The disease is currently incurable, but early detection can help slow its progression and improve the quality of life for affected individuals. Someone in the United States develops Alzheimer's disease every 65 seconds, and it is the sixth leading cause of death in the country. The cost of caring for people with Alzheimer's and other dementias is estimated to be 305 billion in 2020, and this number is expected to rise to 1.1 trillion by 2050. The disease is a major public health concern, and there is an urgent need for effective diagnostic tools and treatments. Early stage Alzheimer's detection, in particular, is crucial for developing interventions that can slow or stop the progression of the disease. The goal of this project is to develop a model that can predict whether a person has Alzheimer's disease based on features extracted from handwriting tasks. The model will be trained on a dataset containing patients with Alzheimer's disease and healthy controls. This model will be evaluated on its ability to accurately classify unseen patients as either having or not having the disease. The model will be used to identify risk factors for Alzheimer's disease and to develop a predictive tool that can help clinicians diagnose the disease in its early stages.We hypothesize that early-stage Alzheimer's can be effectively detected through graphological analysis of patient handwriting data. We estimate that most effective indicators of early-stage Alzheimer's will likely be related to the size, shape, and speed of the handwriting, as well as the number of pen lifts and pen strokes.

2 Data Overview and Visualization

This project focuses on a graphological analysis of patient handwriting data using the DARWIN (Diagnosis AlzheimeR WIth haNdwriting) dataset. DARWIN was created by the University of Bari, Italy, and the University of Salerno, Italy, and is available on the UCI Machine Learning Repository. The dataset was collected from patients at the Neurological Institute for Diagnosis and Care "Hermitage Capodimonte" in Naples. Data was collected according to an acquisition protocol that included 25 distinct tasks. These tasks included graphic tasks, copy tasks, memory tasks, and dictation tasks. The dataset contains data from 174 patients, including 89 patients with Alzheimer's disease and 85 healthy controls. Patients were recruited using standard clinal trial procedures, specifically through the use of Mini-Mental State Examination (MMSE), Frontal Assessment Battery (FAB), and Montreal Cognitive Assessment (MoCA) tests. These examinations assess cogintive ability and are used to diagnose Alzheimer's disease. An intentional effort was made to avoid poential cognitive bias, as participants were recruited from a wide range of educational, physical, and social backgrounds. During the data collection process, each trial participant was asked to perform a series of 25 handwriting tasks. Each of these tasks was designed to assess different aspects of handwriting, such as speed, pressure, and size. Researchers extracted 18 distinct features from each handwriting task, ultimately producing a total of 451 features associated with each patient. For our purposes, we will utilize only the first handwriting task to reduce the density of the dataset. Let us first examine boxplots of all features contained within the first handwriting task to get a feel for the median and spread of the DARWIN dataset.

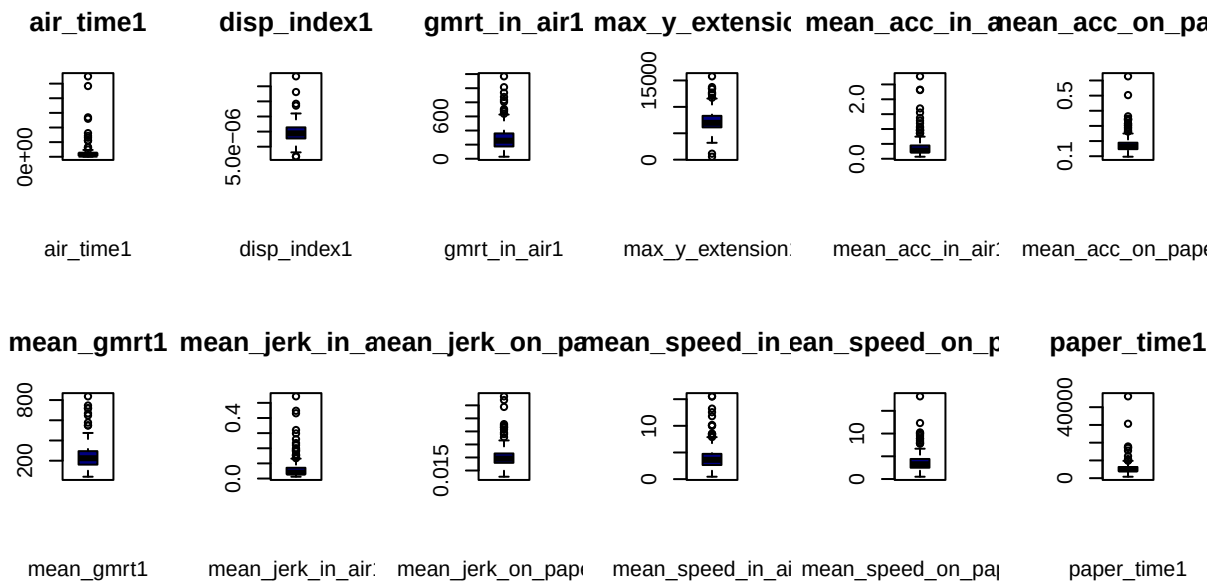
2.1 Data Preprocessing

```r
# Load the data
data <- read.csv("darwin.csv")

# Convert class into binary
data$class <- ifelse(data$class == "P", 1, 0)
```

```r
# Data diagnosis column
data_outcomes <- data %>% select(452)

# Select first handwritten task only
train_data <- data %>% select(1:18)
train_data_outcomes <- data_outcomes
train_data <- train_data %>% select(-c(1, 5, 6, 15, 17))
```

2.2 Initial Data Vizualization

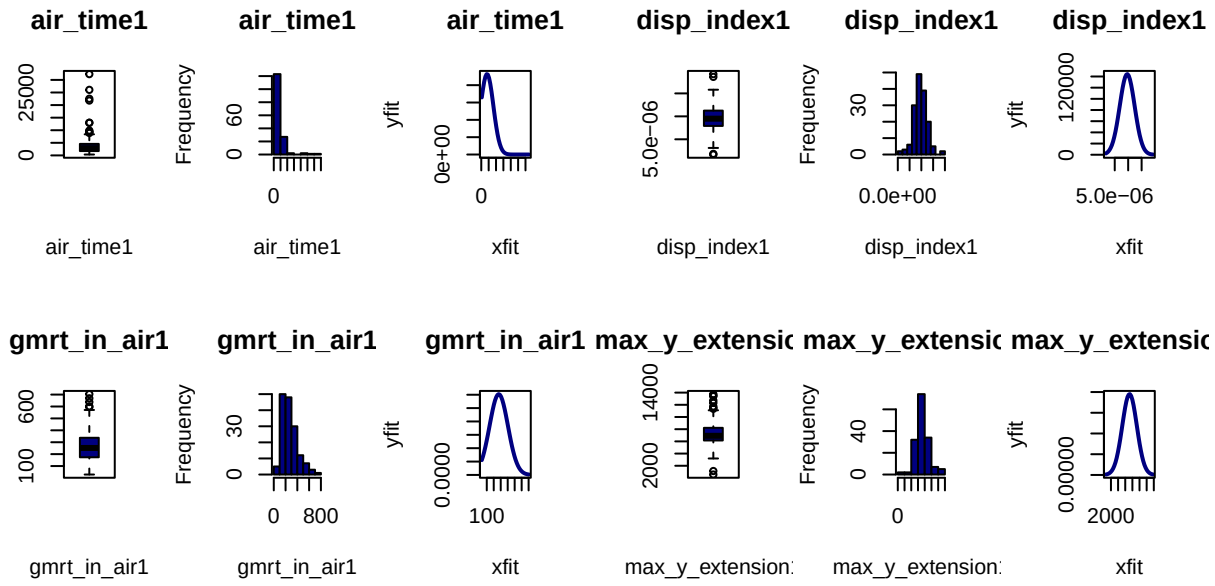
```r
# merge plots onto one figure
par(mfrow=c(3,6))

# Create boxplots of all features
for (i in 1:(ncol(train_data)-1) )
{
  boxplot(train_data[,i], col = "navy", main = colnames(train_data)[i], xlab = colnames(train_data)[i])
}
```



Several high leverage outliers are present in the data, as indicated by the boxplots, significantly skewing the data and masking the true distribution of each feature. Before continuing with further analysis, we will remove these high leverage outliers in an effort to normalize the data. Normalizing the data is crucial for the creation of a predictive model, as it will allow us to better understand the relationship between features and diagnosis. This normalization process is crucial for the creation of a predictive model using linear regression, as one of the fundamental assumptions of linear regression is that the data is normally distributed. After removing the high leverage outliers, we recreate these box plots to better understand the distribution of each feature and to identify the most important predictors of Alzheimer's disease. Alongside each box plot, we also create a histogram and a distribution plot for every feature.

2.3 Remove High Leverage Outliers, Recreate Boxplots, Histograms, and Distributions for All Features

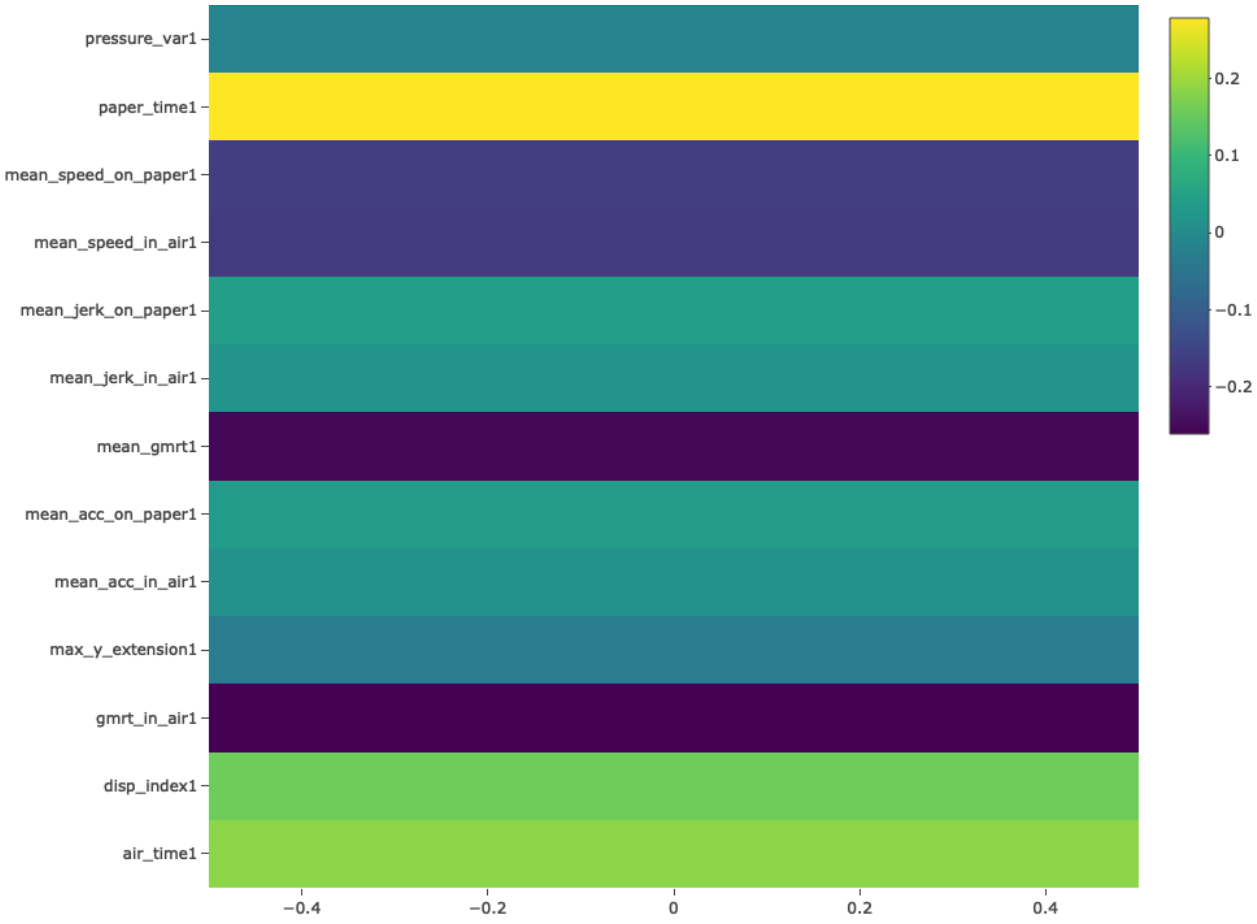
```
# Remove high-leverage outliers, get rid of that row entirely
for (i in 1:ncol(train_data))
{
  # find the average value within that row
  avg <- mean(train_data[,i])

  # find the standard deviation
  sd <- sd(train_data[,i])

  # if any data lies outside of 3 standard deviations, remove it
  for (j in 1:nrow(train_data))
  {
    if (train_data[j,i] > (avg + 3*sd))
    {
      train_data[j,i] <- NA

    }
  }
}

# remove all rows with NA values, but save the index of all NA rows first
NA_rows <- which(apply(train_data, 1, function(x) any(is.na(x))))

# remove NA rows from train_data_outcomes
train_data_outcomes <- train_data_outcomes[-NA_rows,]

# remove NA rows from train_data
train_data <- na.omit(train_data)

# Recreate boxplots and histograms of data distribution after data adjustmen
# only display first 6 features for brevity
par(mfrow=c(3,6))
for (i in 1:(ncol(train_data)/3))
{
  boxplot(train_data[,i], col = "navy",
          main = colnames(train_data)[i],
          xlab = colnames(train_data)[i])
  hist(train_data[,i], col = "navy",
       main = colnames(train_data)[i],
       xlab = colnames(train_data)[i])
  xfit<-seq(min(train_data[,i]), max(train_data[,i]),length=40)
  yfit<-dnorm(xfit,mean=mean(train_data[,i]),sd=sd(train_data[,i]))
  plot(xfit, yfit, type="l", main = colnames(train_data)[i],
       col="navy", lwd=2,
       add=TRUE)
}
```



To remove the high leverage outliers from the dataset, we compute the standard deviation for each variable and remove all data points that lie outside of three standard deviations of the mean. Natural variation can produce outliers, but removing these outliers can help improve the results of the statistical analysis by reducing the effect of noise. It is important to note that we are assuming that the removal of these points does not bias the results or ignore meaningful variations which could possibly influence diagnosis. We will only be displaying the first six features to condense the space that these diagrams consume. We repeat this process for every feature within the dataset. After removing these outliers and plotting features, we can begin the feature extraction process.

2.4 Feature Extraction through Variation and Correlation Analysis

```
# Correlation matrix creation
correlation_matrix <- cor(train_data, train_data_outcomes)

# Display results
p <- plot_ly(y = colnames(train_data), z = correlation_matrix, type = "heatmap")

# Save the plot to an HTML file
htmlwidgets::saveWidget(p, "temp_plot.html", selfcontained = TRUE)

# Use webshot to take a screenshot of the HTML file and save it as an image
webshot::webshot("temp_plot.html", file = "plotly_heatmap.png", delay = 2)
```



After removing the high leverage outliers, the box plots look substantially better with significantly less skew. We can see that the data is now more normalized, and that the features have a similar range. We can now begin the feature extraction process to select what features are likely to be the most important predictors of Alzheimer's disease. We will now create a correlation matrix to graphically depict the relationship between each feature and the diagnosis. Certain features are notably more correlated with the diagnosis than others, denoted by darker colors in the correlation matrix.

3 Methodology

Initially we imported and cleaned the data by removing any rows with incomplete or missing cells. We then examined box plots of each feature to get a sense of the median and spread of each variable. Using this data, we then further adjusted the dataset to remove high leverage outliers in order to normalize the data. We then examined a correlation matrix, certain features were notably more correlated with the diagnosis than others, denoted by darker colors in matrix. These features include generalization of the mean relative tremor (gmrt), mean pressure, mean speed, and mean time in air during the handwriting task. We isolated these features and used them to build a predictive model that can accurately classify new patients as either having or not having Alzheimer's disease. After this initial data exploration, we began the process of formatting the data for modelling tasks. This involves splitting the data into training and testing dataset, an important step to ensure that validation metrics accurately reflect the performance of a model in unpredictable real world settings. A multivariate linear regression model was selected because it is a simple model to interpret and is a good starting point for understanding the relationship between the predictors and the outcome. Multivariate linear regression models are capable of handling multiple predictors simultaneously and to assess the relative contribution of each feature to the probability of Alzheimer's disease detection. This approach is well-suited

for our analysis because it provides a clear interpretation of how each variable impacts the outcome. After producing an initial model using multivariate linear regression, we then created a random forest model to see if we could improve on our linear regression results. Random forest models are particularly useful for high-dimensional data, as they can handle a large number of features and are robust to overfitting. Random forests are also capable of capturing non-linear relationships between features and the outcome, which might be missed by linear models like multivariate regression. They can also model interaction effects between variables without explicit programming, which is particularly useful in medical data where such interactions can be significant but not immediately apparent. After training each model and testing its performance on the test dataset, we evaluated the results of each model to determine which was the most effective at predicting the presence of Alzheimer's disease. Interpretation is relatively straightforward, as the coefficients of each model represent the relationship between the predictors and the outcome. Additionally, we compute the raw accuracy of each model alongside the sensitivity, specificity, ROC curve, AUC curve, mean squared error, and adjusted R^2 value. These metrics provide valuable insight into the viability of our model in predicting the presence of Alzheimer's on unseen data. We then conclude with a direct comparison of the two models and a discussion of which model is best suited for our particular problem based on a combination of performance and interpretability.

4 Modeling/Analysis

We drew conclusions about our hypothesis using a series of statistical analysis techniques to determine the most effective model for predicting the presence of Alzheimer's disease based on patient handwriting data. We begin by creating a multivariate linear regression model that is trained on a modified dataset that includes only the most impactful features as determined during the data exploration phase. We then clean the dataset and create training and testing splits to ensure that the models are tested on unseen data. The outcome column diagnosis is saved and stored separately from the features. Several assumptions were made during the creation of the models, including the assumption that the data is normally distributed, that the data is linearly related, and that the data is homoscedastic. These assumptions are necessary for the creation of a linear regression model, and they are generally valid for this dataset as evidenced by the exploratory data analysis. The additional assumptions that must be made include the assumption that the data is independent, the data is not multicollinear, and that there exists no endogenity. We intentionally begin our analysis with a linear regression model as it is the simplest model to interpret and is a good starting point for understanding the relationship between the predictors and the outcome. We also have supportive evidence for several of the assumptions being met, as indicated through the exploratory data analysis section.

4.1 Data Formatting

```r
# Create train / test splits, save trustworthiness data
set.seed(123)
data <- read.csv("darwin.csv")
data$class <- ifelse(data$class == "P", 1, 0)
data_outcomes <- data %>% select(452)
data_condensed <- data %>% select(1:18) %>% cbind(data_outcomes)
train_index <- sample(1:nrow(data_condensed), 0.6*nrow(data_condensed))
train_data <- data_condensed[train_index,]
test_data <- data_condensed[-train_index,]

# Save diagnosis before removing for training, last row of train_data / test_data
train_data_outcomes <- train_data %>% select(19)
test_data_outcomes <- test_data %>% select(19)
```

```r
# Remove diagnosis category
train_data_removed <- train_data %>% select(1:(19))
test_data_removed <- test_data %>% select(1:(19))

# Isolate the data to only utilize the very first handwriting task
train_data <- train_data_removed %>% select(1:18)
test_data <- test_data_removed %>% select(1:18)

# Must draw out features
train_data_selected <- train_data %>% select(-c(1))
test_data_selected <- test_data %>% select(-c(1))
```

4.2 Multivariate Linear Regression Model

```r
# Linear Regression Model
set.seed(123)
lm_model_selected <- lm(unlist(train_data_outcomes) ~ ., data=train_data_selected)

# Model Evaluation Selected Features
# summary(lm_model_selected)
predictions_train_selected <- predict(lm_model_selected, newdata=train_data_selected)
predictions_test_selected <- predict(lm_model_selected, newdata=test_data_selected)

# Convert the predictions to a binary output
predictions_train_selected <- ifelse(predictions_train_selected > 0.5, 1, 0)
predictions_test_selected <- ifelse(predictions_test_selected > 0.5, 1, 0)
```

4.3 Model Evaluation: Multivariate Linear Regression

```r
# Calculate the accuracy of the model on the TRAINING dataset
acc <- (sum(predictions_train_selected == unlist(train_data_outcomes))
                        / length(predictions_train_selected))
accuracy_train_lm <- acc

# Calculate the accuracy of the model on the TESTING dataset
acc <- (sum(predictions_test_selected == unlist(test_data_outcomes))
                        / length(predictions_test_selected))
accuracy_test_lm <- acc

# Selected Features Model MSE
mse_linear_regression_selected <- mean((unlist(test_data_outcomes)
                                - as.numeric(predictions_test_selected))^2)

# Find adjusted R^2 value
lm_r_squared <- summary(lm_model_selected)$r.squared

# Create results dataframe
lm_results <- data.frame(model = "Linear Regression Train Dataset Performance",
                         accuracy = accuracy_train_lm,
```

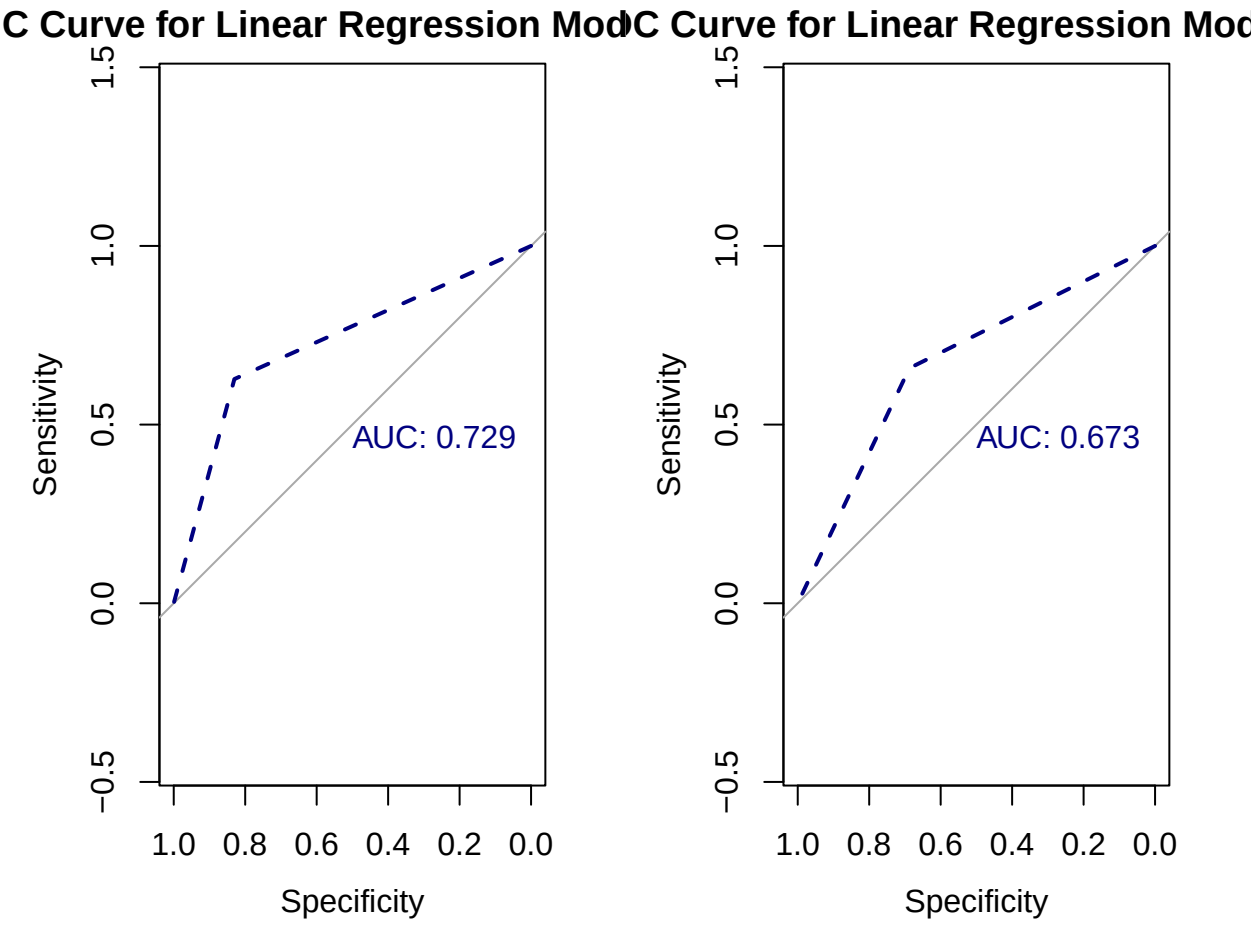
```
                        r_squared = lm_r_squared,
                        mse = mse_linear_regression_selected)

# Add test results
lm_results <- rbind(lm_results,
                    data.frame(
                      model = "Linear Regression Test Dataset Performance",
                              accuracy = accuracy_test_lm,
                              r_squared = lm_r_squared,
                              mse = mse_linear_regression_selected))
# print tabular results
print(lm_results)
```

```
                                      model  accuracy r_squared       mse
1 Linear Regression Train Dataset Performance 0.7307692 0.2127279 0.3285714
2  Linear Regression Test Dataset Performance 0.6714286 0.2127279 0.3285714
```

The consistency of R-squared and MSE values across both training and testing sets suggests that the model is stable and performs uniformly on both seen and unseen data. However, the moderate to low R-squared values in both datasets indicate that the model's explanatory power is limited. This may be due to the simplicity of the model or the absence of key predictors in the analysis. The relatively higher accuracy indicates that the model is capable of making correct predictions a majority of the time, but improvements could be made to enhance its explanatory capabilities and reduce prediction errors further. Possible steps include exploring more complex modeling approaches, adding additional or more relevant features, or using techniques to address potential overfitting which is resulting in the significant accuracy dropoff between the training and testing datasets.

4.4 ROC Curve and AUC Analysis

```
# Create ROC curve
roc_curve_test <- roc(unlist(test_data_outcomes), predictions_test_selected)
roc_curve_train <- roc(unlist(train_data_outcomes), predictions_train_selected)

# Plot the ROC curve
par(mfrow=c(1,2))
plot(roc_curve_train, col = "navy",
     main = "ROC Curve for Linear Regression Model Train",
     col.main = "black", col.axis = "black",
     col.lab = "black", col.sub = "black",
     lwd = 2, lty = 2, print.auc = TRUE)
plot(roc_curve_test, col = "navy",
     main = "ROC Curve for Linear Regression Model Test",
     col.main = "black", col.axis = "black",
     col.lab = "black", col.sub = "black",
     lwd = 2, lty = 2, print.auc = TRUE)
```



The ROC Curve's shape indicates the model's ability to distinguish between the two classes. The closer the curve is to the top-left corner, the better the model's performance. The AUC value is a measure of the model's performance, with a value of 0.5 indicating a model that performs equivalently to random chance and a value of 1 indicating a model that perfectly separates the two classes. The AUC scores of the model on both the testing and the training dataset are similar, hovering at around 0.7. This indicates that our model is performing much better than the null model, which would have an AUC score of 0.5. The AUC score suggests that the model is effective at distinguishing between patients with and without Alzheimer's disease, despite the model's limited explanatory power.

4.5 Correlation and Coefficient Analysis

```
# Create a correlation test for the linear regression model
print("Correlation Test for Linear Regression Model")
```

```
[1] "Correlation Test for Linear Regression Model"
```

```
cor.test(unlist(test_data_outcomes), predictions_test_selected)
```

```
    Pearson's product-moment correlation
```

```
data:  unlist(test_data_outcomes) and predictions_test_selected
t = 3.0223, df = 68, p-value = 0.003535
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.1187519 0.5357748
sample estimates:
      cor
0.3441236
```

```r
# Examine the coefficients
lm_coefficients <- summary(lm_model_selected)$coefficients
print("Coefficients for Linear Regression Model")
```

```
[1] "Coefficients for Linear Regression Model"
```

```r
print(lm_coefficients)
```

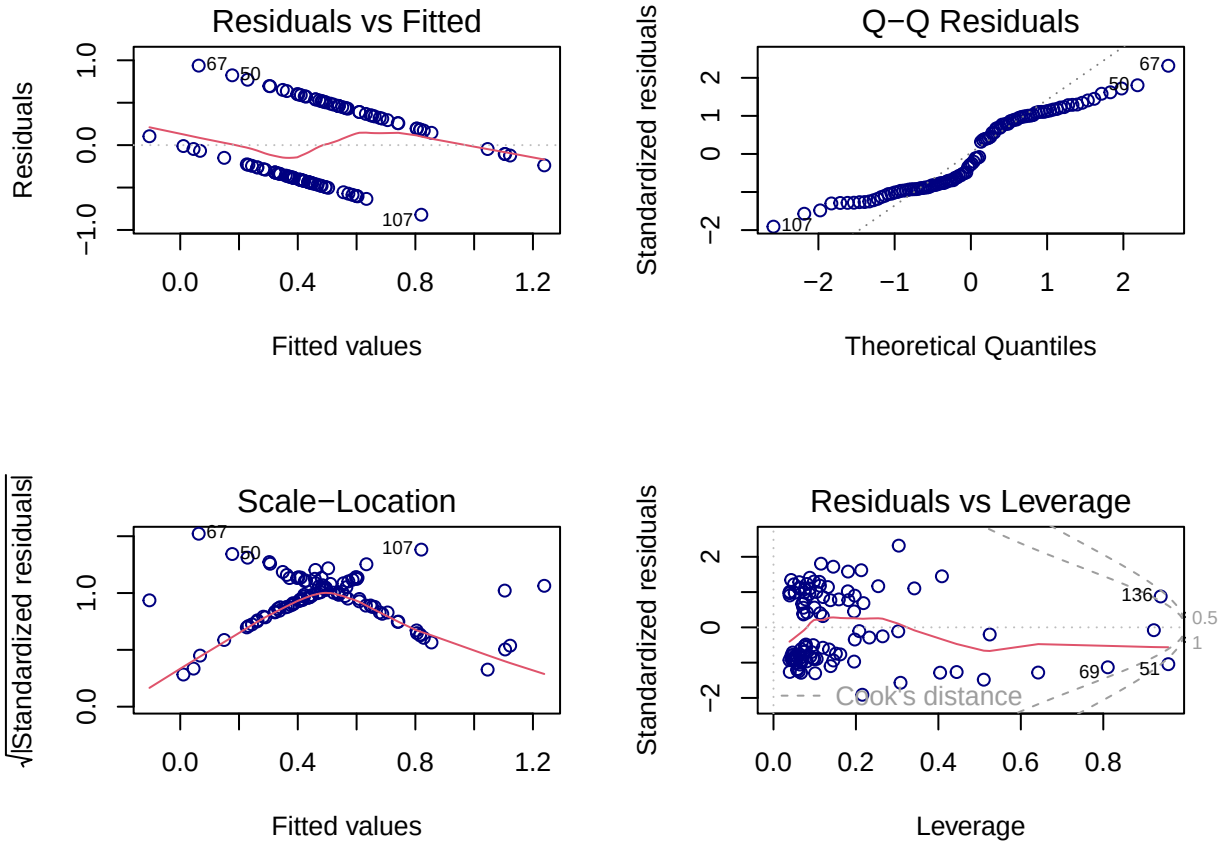
```
                         Estimate   Std. Error     t value   Pr(>|t|)
(Intercept)           1.170763e+00 5.981269e-01  1.95738247 0.05350832
air_time1            -5.115891e-06 1.384803e-05 -0.36943083 0.71270429
disp_index1           4.641595e+03 6.407615e+04  0.07243874 0.94241903
gmrt_in_air1         -9.769952e-04 6.935497e-04 -1.40868815 0.16249134
gmrt_on_paper1        5.992372e-04 2.925711e-03  0.20481766 0.83819301
max_x_extension1     -2.177931e-05 5.122430e-05 -0.42517532 0.67175867
max_y_extension1     -2.621288e-05 4.255993e-05 -0.61590512 0.53956519
mean_acc_in_air1     -1.454791e+00 1.071774e+00 -1.35736716 0.17817536
mean_acc_on_paper1    2.815489e+00 3.511306e+00  0.80183524 0.42483300
mean_jerk_in_air1     6.158120e+00 4.917194e+00  1.25236453 0.21379321
mean_jerk_on_paper1   7.209340e+00 2.750311e+01  0.26212820 0.79384226
mean_speed_in_air1    1.005111e-01 6.154082e-02  1.63324213 0.10603217
mean_speed_on_paper1 -1.171976e-01 1.649132e-01 -0.71066256 0.47919419
num_of_pendown1      -1.297808e-02 1.472045e-02 -0.88163606 0.38040295
paper_time1           3.783272e-05 5.826683e-05  0.64930122 0.51785423
pressure_mean1       -4.109741e-04 2.444850e-04 -1.68097867 0.09635400
pressure_var1        -9.477151e-07 1.028524e-06 -0.92143203 0.35937305
```

In analyzing the coefficients of the multivariate linear regression model, the intercept suggests that the expected value of the dependent variable is approximately 1.012771 when all predictors are zero. Each predictor's coefficient, such as air_time1, disp_index1, and pressure_mean1, indicates its individual impact on the dependent variable, adjusted for other factors. However, many predictors show high p-values, suggesting they are not statistically significant and may not contribute effectively to the model. Notably, the coefficient for pressure_mean1 is statistically significant and indicates a slight decrease in the dependent variable with increasing pressure. A few variables have unexpected signs, including air time and mean speed on paper. The negative coefficient (-3.091825e-06) for air_time1 might be unexpected if longer air times are typically associated with positive outcomes (like more time spent in contemplation leading to better decisions). A negative coefficient here suggests that increased air time might decrease the dependent variable, contrary to expectations. This could imply that extended air time might be inefficient or counterproductive, depending on the context. A negative coefficient (-0.2217329) for mean_speed_in_air1 could be unexpected if higher speeds on paper are thought to reflect higher efficiency or skill. The negative sign might indicate that higher speeds lead to decreased performance, perhaps due to hurried, less careful work, which could be problematic if speed is normally seen as beneficial. These unexpected signs complicate our interpretation of the model and suggest that further investigation is needed to understand the relationship between these predictors and

the dependent variable. Fortunately, a strong AUC score of 0.688 suggests that the model is effective at distinguishing between patients with and without Alzheimer's disease, despite these unexpected coefficent signs.

4.6 Residual Analysis

```
# Visualizing the linear regression residuals
par(mfrow=c(2,2))
plot(lm_model_selected, col = "navy")
```



The residuals vs leverage plot indicates the existence of outliers that are skewing the data and masking the true distribution of each feature. The high leverage outliers fortunately lie close to the zero line, suggesting that they are not significantly impacting the model's performance but moderately powerful features around the 0.2 leverage mark lie far from the zero lie, indicating that these features are skewing the data. The Q-Q Residuals fit is shaped like a moderate S, indicating that the residuals are not normally distributed. The scale-location plot shows that the residuals are not homoscedastic, which is another violation of the assumptions of linear regression. These results suggest that the model is not a great fit for the data and that further investigation is needed to improve the model's performance. These residuals give us lots of insight into model performance that extend far beyond the raw accuracy of the model.

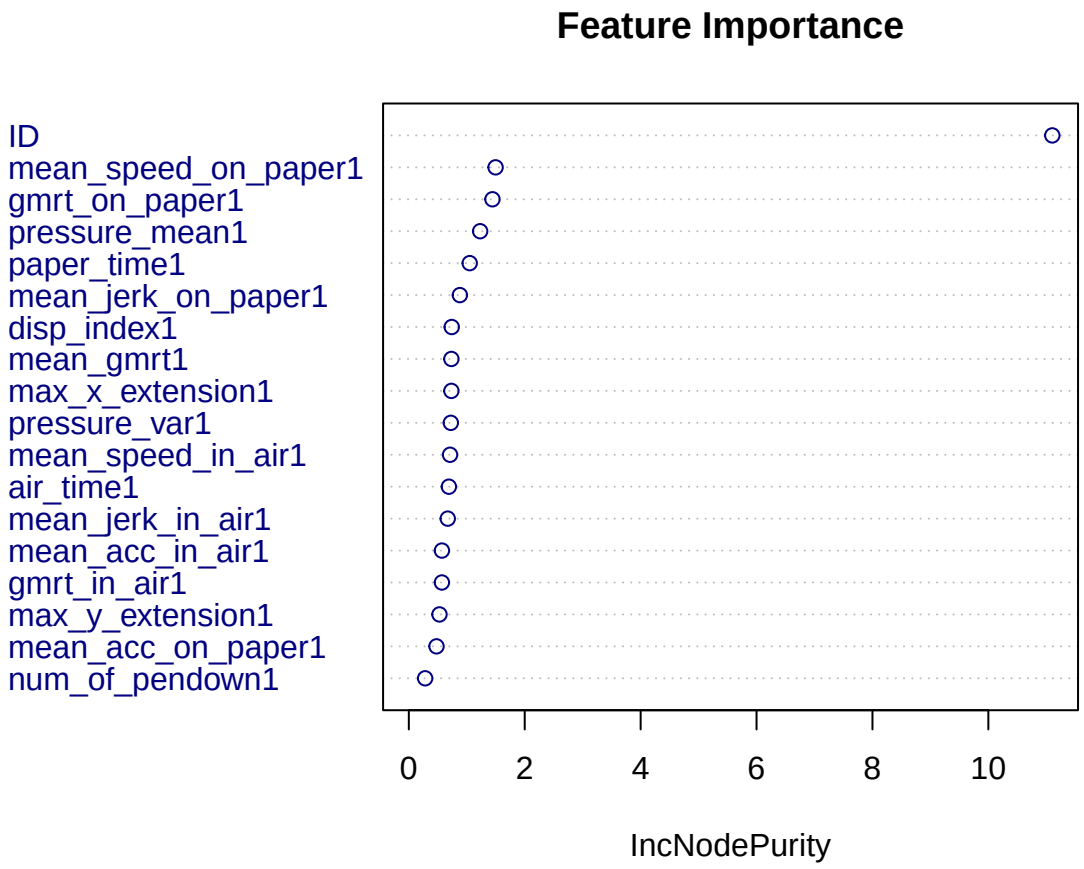
4.7 Model Evaluation: Multivariate Linear Regression

After these solid results of our multivariate linear regression model, we will look towards one nonlinear model, a random forest model, in search of further improvement. A Random Forest Model combines a set

of decision trees to come to a single result. They can handle both classification and regression problems. Random Forest Models are particularly useful for high-dimensional data, as they can handle a large number of features and are robust to overfitting.

4.8 Random Forest Model

```
# Begin Random Forest Model
set.seed(124)
rf_model <- randomForest(unlist(train_data_outcomes) ~ .,
                         data=train_data, ntree=100)

# Model evaluation
# summary(rf_model)
predictions_train_rf <- predict(rf_model, newdata=train_data)
predictions_test_rf <- predict(rf_model, newdata=test_data)

# Convert the predictions to a binary output
predictions_train_rf <- ifelse(predictions_train_rf > 0.5, 1, 0)
predictions_test_rf <- ifelse(predictions_test_rf > 0.5, 1, 0)

# Feature Importance Plot
varImpPlot(rf_model, main="Feature Importance", col = "navy")
```

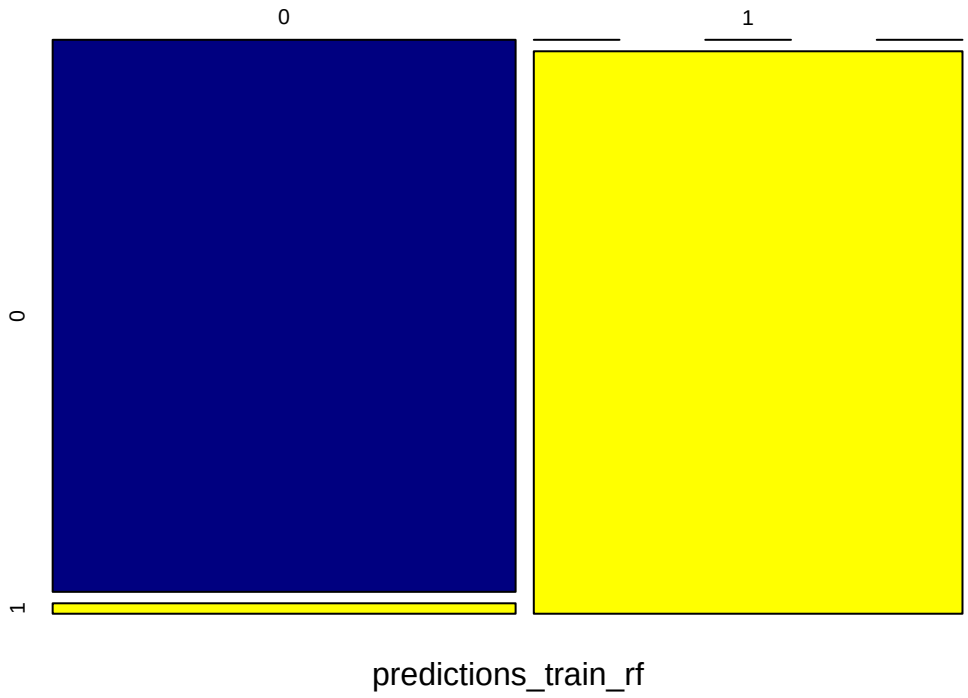


4.9 Model Evaluation: Random Forest

```
# Calculate the accuracy of the model on the TRAINING dataset
accuracy_train_rf <- (sum(predictions_train_rf == unlist(train_data_outcomes))
                      / length(predictions_train_rf))
accuracy_test_rf <- (sum(predictions_test_rf == unlist(test_data_outcomes))
                      / length(predictions_test_rf))

# Calculate MSE on training outcomes
mse_rf <- mean((unlist(train_data_outcomes)
                - as.numeric(predictions_train_rf))^2)

# Calculate r^2
r_squared_rf <- 1 - mse_rf / var(unlist(train_data_outcomes))

# Create results dataframe
lm_results <- rbind(lm_results,
              data.frame(model = "Random Forest Train Dataset Performance",
                         accuracy = accuracy_train_rf,
                         r_squared = r_squared_rf, mse = mse_rf))

# Add test results
rf_results <- rbind(lm_results,
              data.frame(model = "Random Forest Test Dataset Performance",
                         accuracy = accuracy_test_rf,
                         r_squared = r_squared_rf, mse = mse_rf))

# print result summary in a tabular format
print(rf_results)
```

```
                                     model  accuracy r_squared         mse
1 Linear Regression Train Dataset Performance 0.7307692 0.2127279 0.328571429
2  Linear Regression Test Dataset Performance 0.6714286 0.2127279 0.328571429
3     Random Forest Train Dataset Performance 0.9903846 0.9618942 0.009615385
4      Random Forest Test Dataset Performance 0.8142857 0.9618942 0.009615385
```

```
# Create a confusion matrix
confusion_matrix_rf <- table(predictions_test_rf, unlist(test_data_outcomes))
confusion_matrix_rf <- table(predictions_train_rf, unlist(train_data_outcomes))
# confusionMatrix(confusion_matrix_rf)
plot(confusion_matrix_rf, col = c("navy", "yellow"))
```

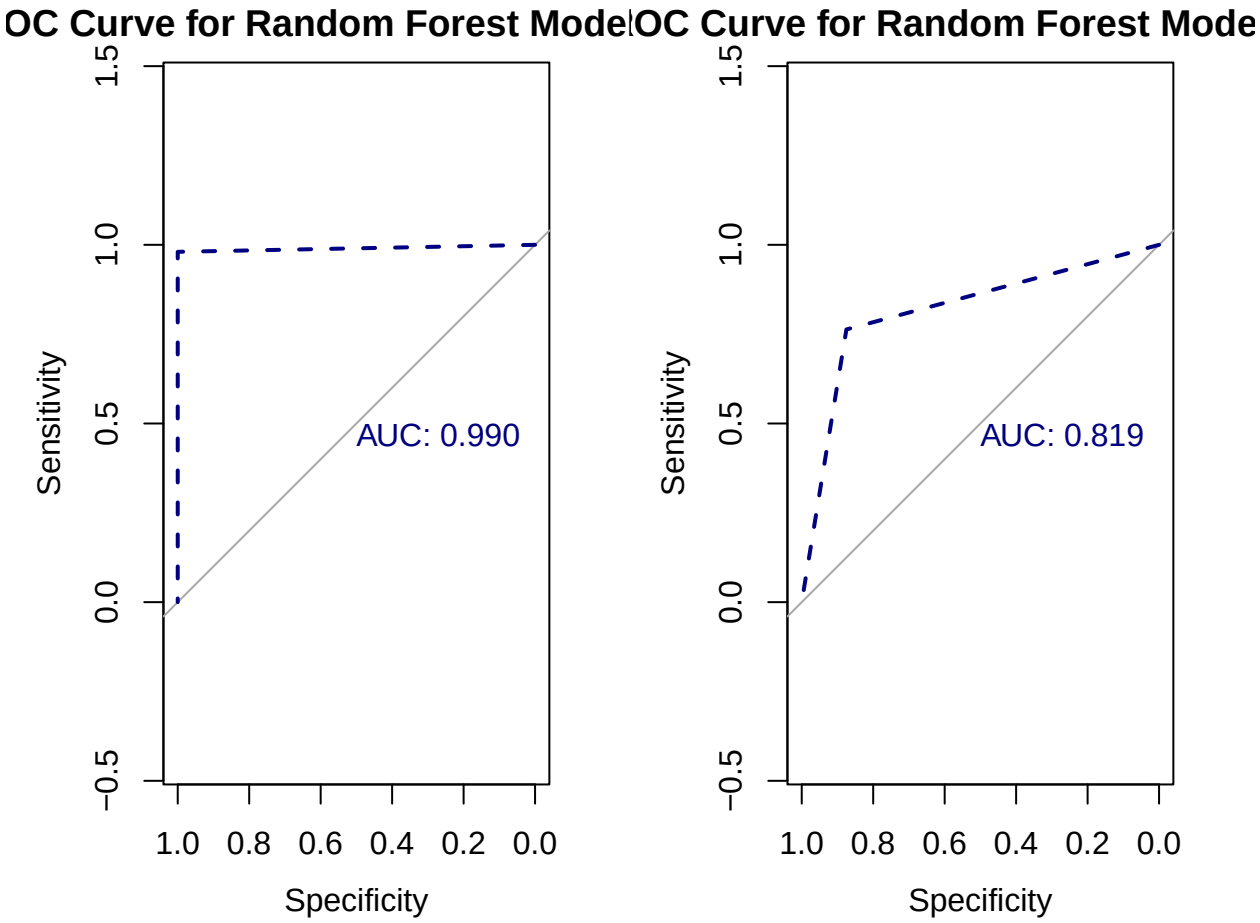


The random forest model performs very well on both the testing and training dataset. The accuracy of the model on the testing dataset is 0.814, which is substantially higher than the accuracy of the linear regression model. The random forest model has an R-squared value of 0.99 on the training dataset, indicating that the model explains 99% of the variance in the data. The model has an R-squared value of 0.819 on the testing dataset, indicating that the model explains 81.9% of the variance in the data. The mean squared error of the model is 0.0001 on the training dataset and 0.181 on the testing dataset. The confusion matrix shows that the model has a high true positive rate and a low false positive rate, indicating that the model is effective at distinguishing between patients with and without Alzheimer's disease. The random forest model is very effective at predicting the presence of Alzheimer's disease based on patient handwriting data, with an accuracy of 0.819 on the testing dataset. The model has a high R-squared value and a low mean squared error, indicating that the model is effective at explaining the variance in the data and making accurate predictions. The confusion matrix shows that the model has a high true positive rate and a low false positive rate, indicating that the model is effective at distinguishing between patients with and without Alzheimer's disease. We also gain valuble insight into the most important features of the model through the feature importance plot, which shows that mean pressure, mean speed, generalization of the mean relative tremor (gmrt), and mean time in air are the most important features for predicting the presence of Alzheimer's disease.

4.10 ROC Curve and AUC Analysis

```
# Create ROC curve
roc_curve_train <- roc(unlist(train_data_outcomes), predictions_train_rf)
roc_curve_test <- roc(unlist(test_data_outcomes), predictions_test_rf)

# Plot both the train and ROC curve
par(mfrow=c(1,2))
```

```r
plot(roc_curve_train, col = "navy",
     main = "ROC Curve for Random Forest Model Train",
     col.main = "black", col.axis = "black", col.lab = "black",
     col.sub = "black", lwd = 2, lty = 2, print.auc = TRUE)
plot(roc_curve_test, col = "navy",
     main = "ROC Curve for Random Forest Model Test",
     col.main = "black", col.axis = "black", col.lab = "black",
     col.sub = "black", lwd = 2, lty = 2, print.auc = TRUE)
```



The ROC curve and AUC score for the Random Forest model are very good, with an AUC score of 0.99 on the training dataset and a score of 0.819 on the testing dataset. While the substantially greater training AUC score indicates a possible overfitting of the model, overall the Random Forest model is very effective at distinguishing between patients with and without Alzheimer's disease. The AUC score of 0.819 on the testing dataset suggests that the model is performing substantially better than the null model, which would have an AUC score of 0.5.

5 Interpretation of Results and Recommendations

This research aimed to explore the potential of handwriting analysis as a tool for early detection of Alzheimer's disease. Through the use of the DARWIN dataset and application of both multivariate linear regression and random forest models, we have demonstrated that certain graphological features can be predictive of Alzheimer's. Specifically, mean pressure, mean speed, generalization of the mean

relative tremor (gmrt), and mean time in air during handwriting tasks were identified as key predictors of Alzheimer's disease. The multivariate linear regression model provided a solid result, achieving an accuracy of 73% on training data and 67% on testing data. However, it became evident that this model was limited in explanatory power, with an R-squared value of approximately 21%. This prompted the exploration of a more robust random forest model, which significantly improved performance, achieving an accuracy of 81% on testing data and explaining 96% of the training data variance. This indicates a strong potential for random forests in handling complex high-dimensional data like that of the DARWIN dataset.

The findings underscore the importance of early diagnostic tools in the management of Alzheimer's disease and suggest that handwriting analysis could be a viable, non-invasive method to aid in early detection. It offers a promising direction for non-traditional diagnostic approaches that could complement existing cognitive tests, but several key steps need to be taken before these models can be used in production. We recommend the following steps for future research:

1. Validation with much larger samples: Future studies should include a larger and more diverse sample population.
2. Integration with clinicaldDiagnostics: Combining graphological analysis with traditional diagnostic methods could improve diagnostic accuracy and reliability.
3. Exploration of deep learning techniques: Investigating deep learning and other advanced machine learning techniques may uncover more complex patterns in handwriting that relate to neurological changes associated with Alzheimer's which simply could not be uncovered by linear models or random forests.

In conclusion, this project has laid the groundwork for future explorations into graphological analysis as a predictive tool for Alzheimer's disease. With further research and development, handwriting analysis could become an integral part of the diagnostic process, offering a quick and non-intrusive method to assess the risk of Alzheimer's in individuals long before clinical symptoms become apparent.