S&DS 220 Report Template

Braeden Cullen

April 21st, 2024

Contents

1 Introduction **1**

2 Problem Statement **1**

3 Hypothesis **2**

4 Data Overview **2**

5 Data exploration and visualization **2**

6 Methodology **3**

7 Results **3**

8 Modeling/Analysis **3**

9 Visualization and interpretation of the results **3**

10 Conclusions and recommendations **4**

11 Code **4**

11.1 Data Preprocessing 4

11.2 Variable Selection 5

11.3 Modelling 9

1 Introduction

2 Problem Statement

- a non-technical description of the problem you are trying to solve or the question you are trying to answer, and why you are trying to answer that question

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. Alzheimers 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 machine learning model that can predict whether a person has Alzheimer's disease based on demographic and clinical data. The model will be trained on a dataset of patients with and without Alzheimer's disease and will be evaluated on its ability to accurately classify new 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.

3 Hypothesis

- a non-technical description of the hypothesis you are testing

Early-stage alzehimers can be effectively detected through graphological analysis of patient handwriting data. The most effective indicators of early-stage alzherimers will likely be related to the size, shape, and speed of the handwriting, as well as the number of pen lifts and pen strokes. Using a series of metrics, we can determine which features are most indicative of early-stage alzheimers, and use these features to develop a predictive model that can accurately classify patients as either having or not having the disease.

4 Data Overview

- a non-technical description of the data, where it came from, and what it contains, including the predictors, the outcome, and the observations

This project focuses on using graphological analysis of patient handwriting data, provided through the DARWIN dataset, to determine indicators of early-stage Alzheimers. This dataset contains data from 174 patients, including 89 patients with Alzheimer's disease and 85 healthy controls. The data includes wether the patient has Alzheimer's disease or not, as well as 451 features carefully collected during the participant handwriting process. These features include the number of pen lifts and the number of pen strokes, as well as various measures of the size, shape, and speed of the handwriting.

5 Data exploration and visualization

- This section will give an overview of the data. It should include descriptive statistics and visualizations of the raw data. Reveal to the reader any interesting relationships in the data, and if you are doing multiple regression, convince the reader that the predictors are related to the outcome. Visualizations are one of the most powerful ways to communicate information to the reader, so it is important to spend time producing clear, descriptive, eye-catching visualizations.
- again, this should be nontechnical

6 Methodology

- a non-technical description of what kind of analysis you did and how to interpret the results of the model

7 Results

- a non-technical description of the results of the model and main takeaways.

8 Modeling/Analysis

Describe the statistical inference/hypothesis testing/regression model(s) used and the analysis that was performed. Discuss

- Any assumptions that are made
- The observations (the rows of the data), the predictors (non-outcome columns of the data), and the outcome (one of the column of the data)
- Interpret the results of the model
  - If regression:
    - What the coefficients mean and how this is related to your problem
    - Appropriate measures of the performance of the model, such as adjusted R^2
    - How easy/hard it is to interpret the results and explain them to either a technical or non-technical audience. For example, do the coefficients have the expected sign? Are the sizes (magnitudes) of the coefficients reasonable, and can you put them in real world terms?
  - For other kinds of analysis, what you give is highly dependent on the type of analysis you do. But in general, talk about assumptions, if they are appropriate, how they might not be appropriate, and why you chose this type of analysis.
- Whether or not you think the model is appropriate for this kind of data, and why.

9 Visualization and interpretation of the results

Create visualizations of the results, focusing on visualizations that

- help describe aspects of the results that have real-world interpretation
- help the reader understand how the model addresses the problem you are studying.

Visualizations are one of the most powerful ways to communicate information to the reader, so it is important to spend time producing clear, descriptive, eye-catching visualizations.

Discuss the results of the model or models you chose, and describe how they are related to the problem statement or question that you were trying to answer in the project.

If you build multiple models or perform multiple types of analysis, compare the measures of performance and the ease of interpretability across models or types of analysis, stating which model or models performed best, and which model or models were most interpretable. Finally, decide which model or type of analysis is best for your particular problem based on some combination of performance and interpretability.

10 Conclusions and recommendations

A few sentences stating conclusions, recommendations, and ideas for future work and improvements.

11 Code

Guiding question: Can we predict the existence of Alzheimer's disease based on patient handwriting data?

11.1 Data Preprocessing

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

# Print Dimmensions of Face Data
print(dim(data))
```

```
[1] 174 452
```

```
# Check for missing values
if(sum(is.na(data)) != 0) {
  print("There are missing values in the data")
}

# Check for duplicates
if(sum(duplicated(data)) != 0) {
  print("There are duplicates in the data")
}

# Convert diagnosis into a binary value, P = 1 HC = 0
data$class <- ifelse(data$class == "P", 1, 0)

# Indicies, we will be limiting our analysis to the first 100 features
x1_idx = 2;
x2_idx = 100;
diagnosis_idx = 452;

data_condensed <- data %>% select(x1_idx:x2_idx, diagnosis_idx)

# Create train / test splits, save trustworthiness data
set.seed(123)
train_index <- sample(1:nrow(data_condensed), 0.7*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(x2_idx - x1_idx + 2)
test_data_outcomes <- test_data %>% select(x2_idx - x1_idx + 2)

# Remove diagnosis category
```

```r
train_data_removed <- train_data %>% select(1:(x2_idx - x1_idx + 1))
test_data_removed <- test_data %>% select(1:(x2_idx - x1_idx + 1))
```

11.2 Variable Selection

```r
# Linear Regression Model
set.seed(42)
lm_model <- lm(unlist(train_data_outcomes) ~ ., data=train_data_removed)

# Model Evaluation
# summary(lm_model)
predictions_train <- predict(lm_model, newdata=train_data_removed)
predictions_test <- predict(lm_model, newdata=test_data_removed)
predictions_list <- unname(as.list(as.data.frame(predictions_train)))
train_data_outcomes_numeric <- as.numeric(unlist(train_data_outcomes))

# Find the statistically significant coefficents of the model
significant_coefficients <- summary(lm_model)$coefficients[summary(lm_model)$coefficients[,4] < 0.05,]
print(significant_coefficients)
```

```
                        Estimate   Std. Error   t value    Pr(>|t|)
mean_speed_in_air2  1.472096e-01 6.420982e-02  2.292634 0.028589879
gmrt_in_air3        1.477457e-03 6.316382e-04  2.339087 0.025736608
mean_acc_in_air5   -6.317344e+00 2.158515e+00 -2.926709 0.006257505
mean_jerk_in_air5   2.571136e+01 1.017056e+01  2.528017 0.016604131
pressure_var5       7.049308e-06 2.250710e-06  3.132037 0.003697171
```

```r
# Calculate the adjusted R^2 value
adjusted_r_squared <- summary(lm_model)$adj.r.squared
print(adjusted_r_squared)
```

```
[1] 0.4312823
```

```r
# Create a correlation test for the linear regression model
cor.test(unlist(test_data_outcomes), predictions_test)
```

```

    Pearson's product-moment correlation

data:  unlist(test_data_outcomes) and predictions_test
t = 2.9309, df = 51, p-value = 0.005047
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.1218904 0.5894750
sample estimates:
      cor
0.3796754
```

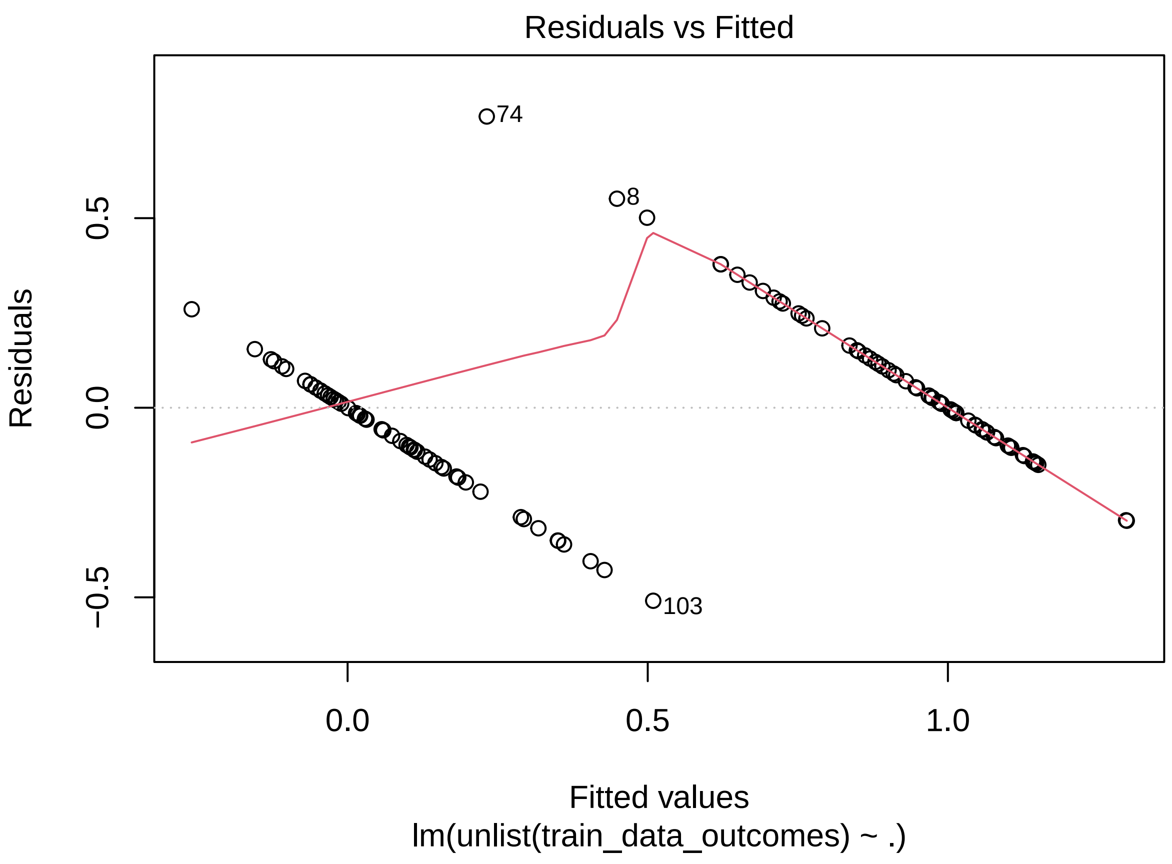
```
cor.test(unlist(train_data_outcomes), predictions_train)
```

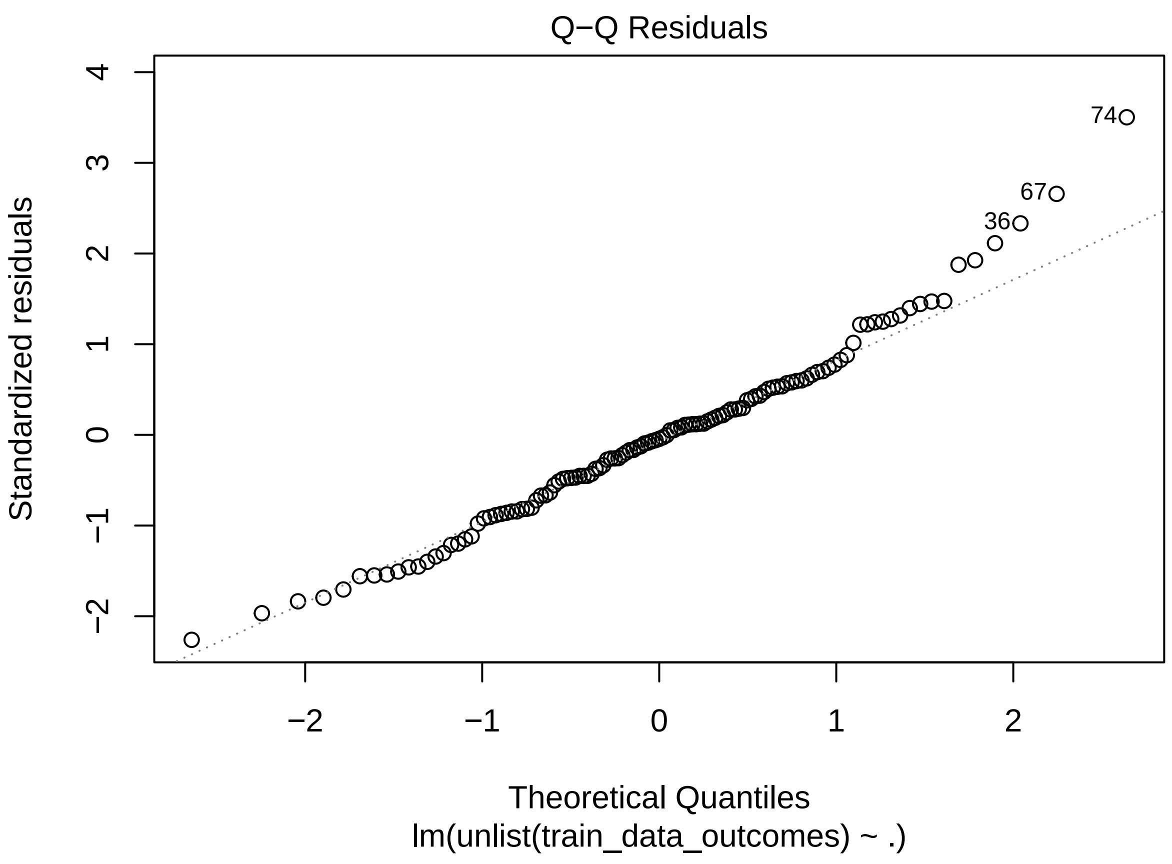
```

	Pearson's product-moment correlation

data:  unlist(train_data_outcomes) and predictions_train
t = 25.8, df = 119, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.8886582 0.9443031
sample estimates:
      cor 
0.9210548 
```

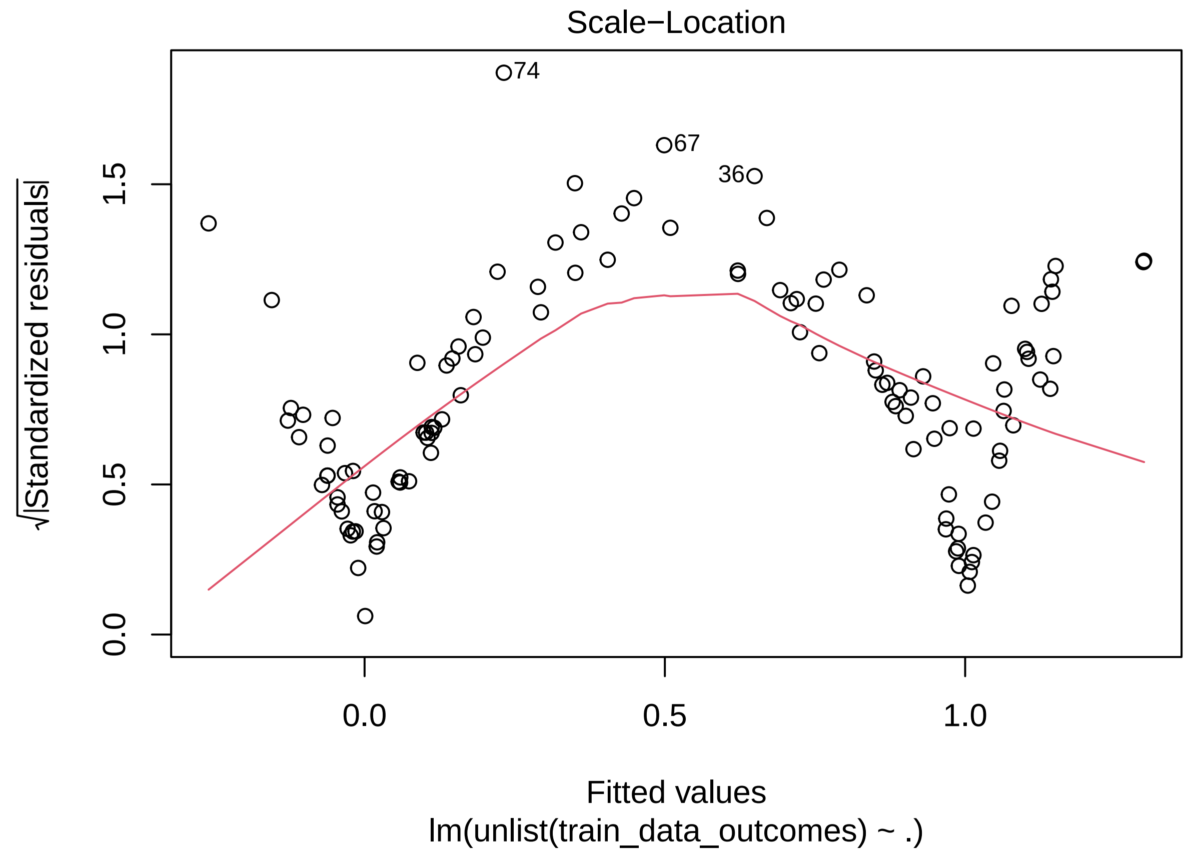
```
# Store significant variables for future training within a new dataframe
significant_variables <- data.frame()
for (i in 1:nrow(significant_coefficients)) {
  significant_variables <- rbind(significant_variables, data.frame(variable = rownames(significant_coef
}
significant_variables
```

```
            variable   coefficient
1 mean_speed_in_air2  1.472096e-01
2       gmrt_in_air3  1.477457e-03
3   mean_acc_in_air5 -6.317344e+00
4  mean_jerk_in_air5  2.571136e+01
5      pressure_var5  7.049308e-06
```

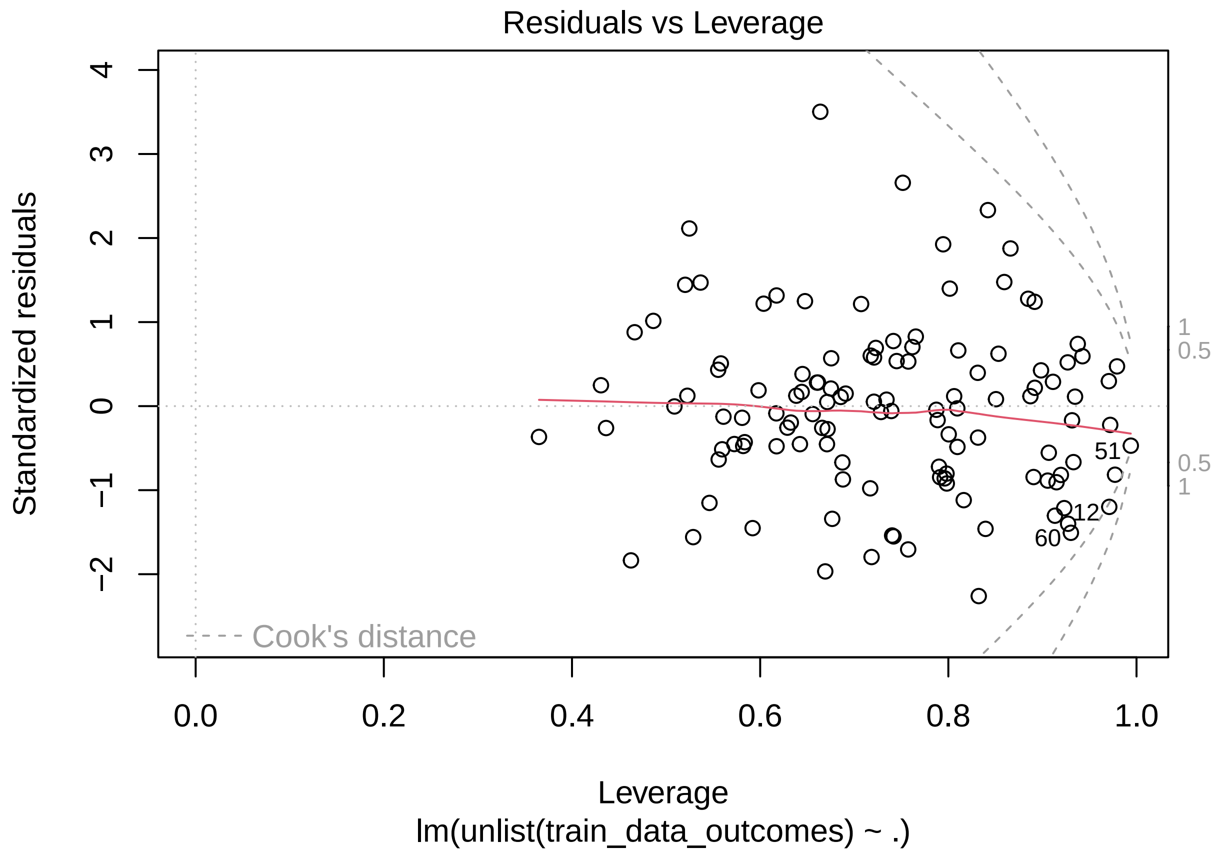
```
# Visualizing the linear regression model
plot(lm_model)
```


Residuals vs Fitted
Residuals
Fitted values
lm(unlist(train_data_outcomes) ~ .)
74
8
103
0.5
0.0
−0.5
0.0
0.5
1.0


Q−Q Residuals
Standardized residuals
Theoretical Quantiles
lm(unlist(train_data_outcomes) ~ .)
74
67
36
4
3
2
1
0
−1
−2
−2
−1
0
1
2



Scale−Location
74
67
36
√|Standardized residuals|
0.0
0.5
1.0
1.5
Fitted values
lm(unlist(train_data_outcomes) ~ .)



Residuals vs Leverage
Standardized residuals
Leverage
lm(unlist(train_data_outcomes) ~ .)
Cook's distance
51
12
60
1
0.5
0.5
1
0.0
0.2
0.4
0.6
0.8
1.0

```
# Linear regression model MSE
mse <- mean((unlist(test_data_outcomes) - as.numeric(predictions_test))^2)
mse
```

```
[1] 0.5338879
```

11.3 Modelling

```
# Begin SVM Model
svm_model <- svm(unlist(train_data_outcomes) ~ ., data=train_data_removed, kernel="linear", cost=10)

# Model Evaluation
predictions_test_svm <- predict(svm_model, newdata=test_data_removed)

# Calculate the accuracy of the model
accuracy_train_svm <- sum(predictions_test_svm == unlist(test_data_outcomes)) / length(predictions_test

# Create a confusion matrix
confusion_matrix <- table(predictions_test_svm, unlist(test_data_outcomes))

# Calculate the sensitivity and specificity of the model
sensitivity_svm <- confusion_matrix[2,2] / (confusion_matrix[2,2] + confusion_matrix[2,1])
specificity_svm <- confusion_matrix[1,1] / (confusion_matrix[1,1] + confusion_matrix[1,2])

# Calculate the ROC curve
# roc_curve_svm <- roc(unlist(test_data_outcomes), as.numeric(predictions_test_svm))

# Calculate the AUC
# auc_svm <- auc(roc_curve_svm)

# Visualize the ROC curve
# plot(roc_curve_svm)

# Calculate MSE
mse_svm <- mean((unlist(test_data_outcomes) - as.numeric(predictions_test_svm))^2)
mse
```

```
[1] 0.5338879
```

```
# Store the model results
model_results <- data.frame(model = "SVM", accuracy = accuracy_train_svm, sensitivity = sensitivity_svm


# Begin Random Forest Model
rf_model <- randomForest(unlist(train_data_outcomes) ~ ., data=train_data_removed, ntree=100)

# Model Evaluation
predictions_test_rf <- predict(rf_model, newdata=test_data_removed)

# Calculate the accuracy of the models
accuracy_train_rf <- sum(predictions_test_rf == unlist(test_data_outcomes)) / length(predictions_test_r
```

```r
# Create a confusion matrix
confusion_matrix_rf <- table(predictions_test_rf, unlist(test_data_outcomes))

# Calculate the sensitivity and specificity of the model
sensitivity_rf <- confusion_matrix[2,2] / (confusion_matrix[2,2] + confusion_matrix[2,1])
specificity_rf <- confusion_matrix[1,1] / (confusion_matrix[1,1] + confusion_matrix[1,2])

# Calculate the ROC curve
#roc_curve_rf <- roc(unlist(test_data_outcomes), as.numeric(predictions_test_rf))

# Calculate the AUC
# auc_rf <- auc(roc_curve)

# Visualize the ROC curve
# plot(roc_curve_rf)

# Calculate MSE
mse_svm <- mean((unlist(test_data_outcomes) - as.numeric(predictions_test_rf))^2)
mse
```

```
[1] 0.5338879
```

```r
# Store the model results
model_results <- rbind(model_results, data.frame(model = "Random Forest", accuracy = accuracy_train_rf,

# KNN Model ?
```