Predicting Extrovert vs. Introvert Behavior

Blake Simpson

2025-06-11

Contents

1 Introduction

This project explores a behavioral dataset aimed at distinguishing between extroverted and introverted personality types. The analysis involves exploratory visualization, data preprocessing, and the application of classification models, including Random Forest and Elastic Net, with additional emphasis on threshold tuning and performance evaluation. Metrics such as Accuracy, Precision, Recall, F1 Score, and AUC are used to compare model performance. The goal is to develop a reliable pipeline for classifying personality traits based on social behavior while also assessing each model's sensitivity to changes in classification thresholds. The dataset contains 2,900 rows and 8 columns and can be found in the dataset folder or from the original Kaggle source.

1.1 Features/variables

```
  - Time_spent_Alone: Hours spent alone daily (0-11).
  - Stage_fear: Presence of stage fright (Yes/No).
  - Social_event_attendance: Frequency of social events (0-10).
  - Going_outside: Frequency of going outside (0-7).
  - Drained_after_socializing: Feeling drained after socializing (Yes/No).
  - Friends_circle_size: Number of close friends (0-15).
  - Post_frequency: Social media post frequency (0-10).
  - Personality: Target variable (Extrovert/Introvert).
```

2 Importing libraries and loading in data

```
# Setup
library(dplyr)
library(ggplot2)
library(reshape2)
library(caret)
library(randomForest)
library(glmnet)
library(pROC)
library(MLmetrics)
library(knitr)
library(kableExtra)

personality_df <- read.csv("personality_dataset.csv")
head(personality_df)
```

```
##   Time_spent_Alone Stage_fear Social_event_attendance Going_outside
## 1                4         No                       4             6
## 2                9        Yes                       0             0
## 3                9        Yes                       1             2
## 4                0         No                       6             7
## 5                3         No                       9             4
## 6                1         No                       7             5
##   Drained_after_socializing Friends_circle_size Post_frequency Personality
## 1                        No                  13              5   Extrovert
## 2                       Yes                   0              3   Introvert
## 3                       Yes                   5              2   Introvert
## 4                        No                  14              8   Extrovert
## 5                            No                    8              5    Extrovert
## 6                            No                    6              6    Extrovert
```

```
# Set random seed
set.seed(483)
```

3 Exploratory Data Analysis

We will look at the basic information and graphs of our dataset and look for any problems that will need to be addressed during data preprocessing.

3.1 Basic Info

```
# Display basic information
summary(personality_df)
```

```
##  Time_spent_Alone  Stage_fear        Social_event_attendance Going_outside
##  Min.   : 0.000   Length:2900        Min.   : 0.000          Min.   :0
##  1st Qu.: 2.000   Class :character   1st Qu.: 2.000          1st Qu.:1
##  Median : 4.000   Mode  :character   Median : 3.000          Median :3
##  Mean   : 4.506                      Mean   : 3.963          Mean   :3
##  3rd Qu.: 8.000                      3rd Qu.: 6.000          3rd Qu.:5
##  Max.   :11.000                      Max.   :10.000          Max.   :7
##  NA's   :63                          NA's   :62              NA's   :66
##  Drained_after_socializing Friends_circle_size Post_frequency
##  Length:2900               Min.   : 0.000      Min.   : 0.000
##  Class :character          1st Qu.: 3.000      1st Qu.: 1.000
##  Mode  :character          Median : 5.000      Median : 3.000
##                            Mean   : 6.269      Mean   : 3.565
##                            3rd Qu.:10.000      3rd Qu.: 6.000
##                            Max.   :15.000      Max.   :10.000
##                            NA's   :77          NA's   :65
##  Personality
##  Length:2900
##  Class :character
##  Mode  :character
##
##
##
##
```

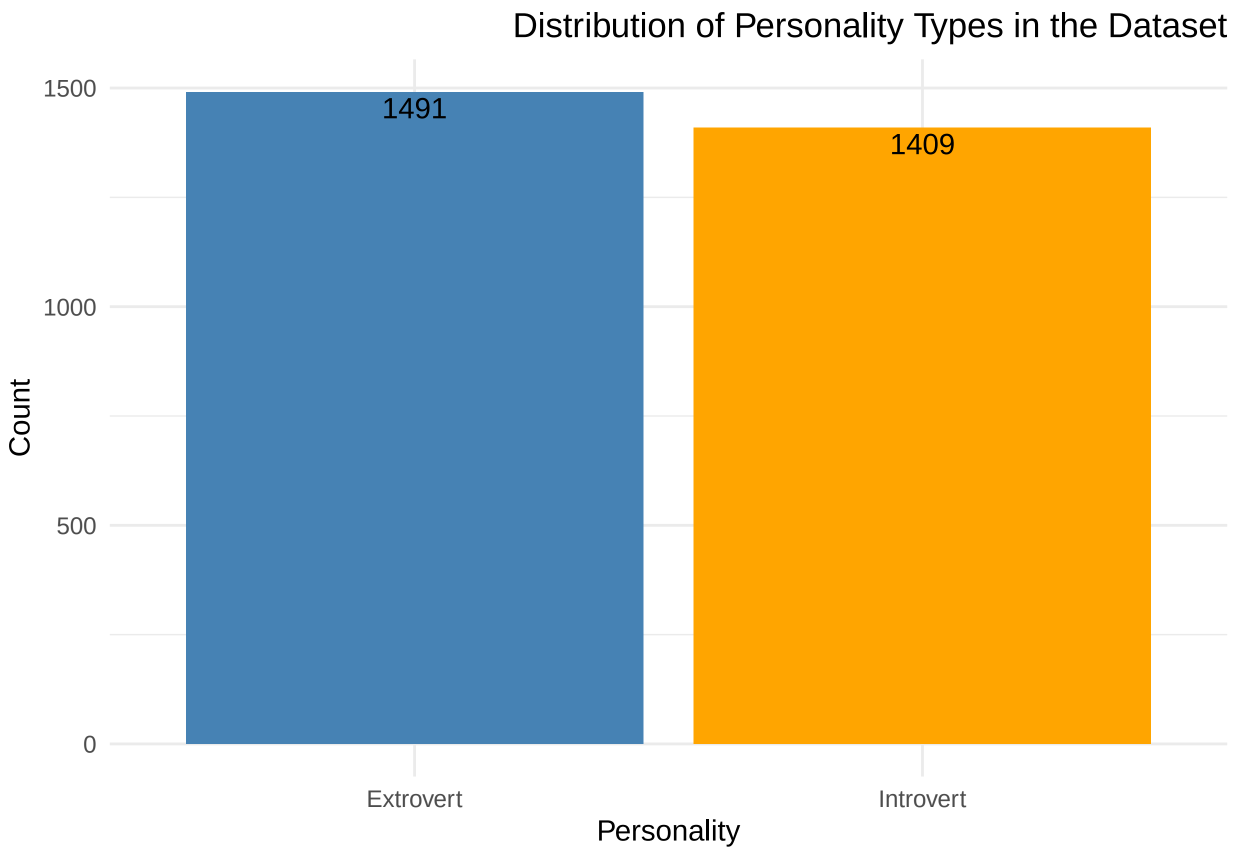
```
# Setting any empty string to NA
personality_df[personality_df == ""] <- NA
# Looking for NA values in each feature
colSums(is.na(personality_df))
```

```
##          Time_spent_Alone                Stage_fear   Social_event_attendance
##                        63                        73                        62
##             Going_outside Drained_after_socializing       Friends_circle_size
##                        66                        52                        77
##            Post_frequency               Personality
##                        65                         0
```

As we can see, we have some NA values in each feature that will need to be fixed during the data preprocessing phase.

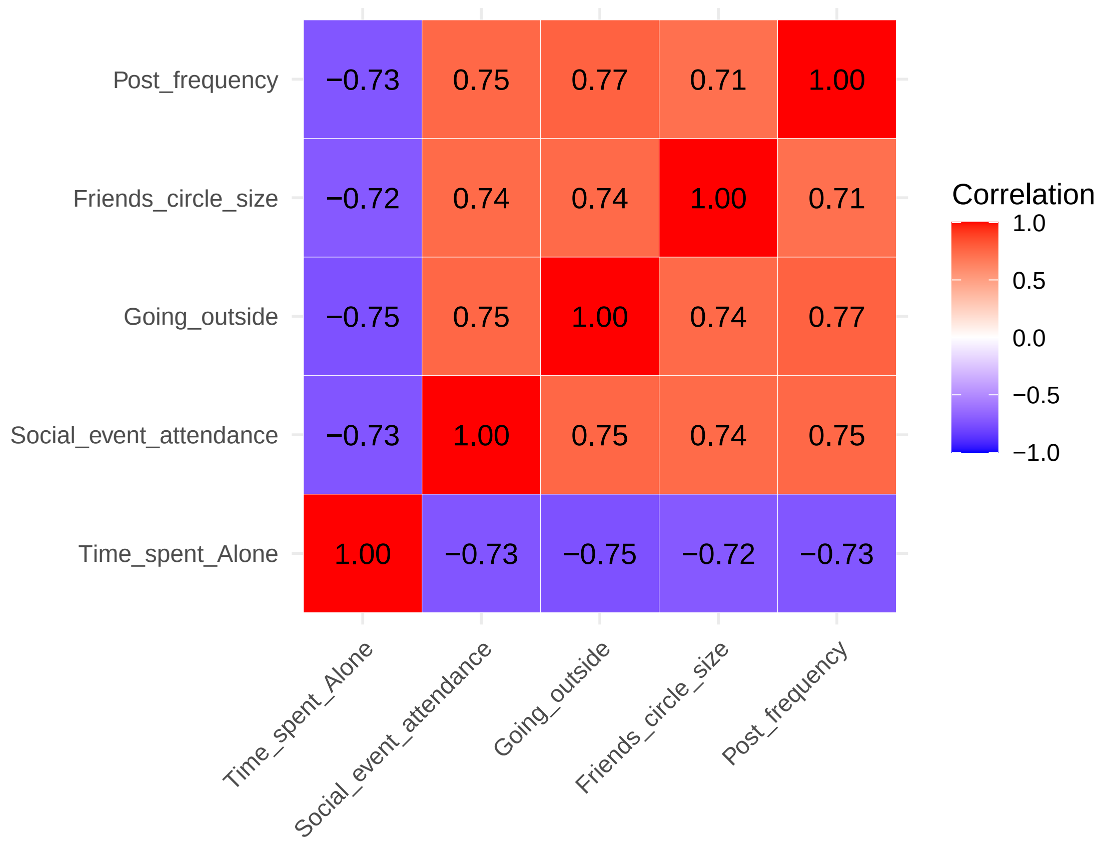
3.2 Class distribution graph

```r
# Bar Graph of Class distribution
ggplot(personality_df, aes(x = Personality, fill = Personality)) +
  geom_bar() +
  geom_text(stat = "count", aes(label = ..count..), vjust = 1.25) +  # add counts on top
  labs(title = "Distribution of Personality Types in the Dataset", x = "Personality", y = "Count") +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 1),  # center the title
    legend.position = "none"                # optional: hide legend
  ) +
  scale_fill_manual(values = c("Extrovert" = "steelblue", "Introvert" = "orange"))
```



3.3 Correlation matrix (numeric features)

```r
# Compute correlation matrix and Convert matrix to long format for ggplot
cor_matrix <- melt(cor(personality_df[sapply(personality_df, is.numeric)], use = "complete.obs"))

# Plot correlation heatmap with numbers
ggplot(cor_matrix, aes(Var1, Var2, fill = value)) +
  geom_tile(color = "white") +                    # heatmap tiles
  geom_text(aes(label = sprintf("%.2f", value)),  # numbers with 2 decimals
            color = "black", size = 4) +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                       midpoint = 0, limit = c(-1,1), space = "Lab",
                       name = "Correlation") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust=1),
        axis.title.x = element_blank(),
        axis.title.y = element_blank()) +
  coord_fixed()
```



4 Data Preprocessing

We will handle missing values by inputting the mean for numeric features and the mode for categorical feature.

4.1 Handle missing values

```r
# Function to find the mode
get_mode <- function(x) {
  ux <- na.omit(unique(x))
  ux[which.max(tabulate(match(x, ux)))]
}
```

```
# Handle missing values
personality_df <- personality_df %>%
  # Numeric features: replace with median
  mutate(across(where(is.numeric), ~ ifelse(is.na(.), median(., na.rm = TRUE), .))) %>%
  # Categorical features: replace with mode
  mutate(across(where(is.character), ~ ifelse(is.na(.), get_mode(.), .)))

# Check to make sure that there are no missing values
colSums(is.na(personality_df))
```

```
##          Time_spent_Alone                Stage_fear   Social_event_attendance
##                         0                         0                         0
##             Going_outside Drained_after_socializing       Friends_circle_size
##                         0                         0                         0
##            Post_frequency               Personality
##                         0                         0
```

5 Model Training and Evaluation

We will train a Random Forest and a Elastic Net and report on important metrics such as Accuracy, Precision, Recall, F1 Score, and AUC.

5.1 Spliting the data (Train/Test)

We will split the data into 80% training and 20% testing

```
# Create train/test indices
train_index <- createDataPartition(personality_df$Personality, p = 0.8, list = FALSE)

# Split the data
train_df <- personality_df[train_index, ]
test_df <- personality_df[-train_index, ]
```

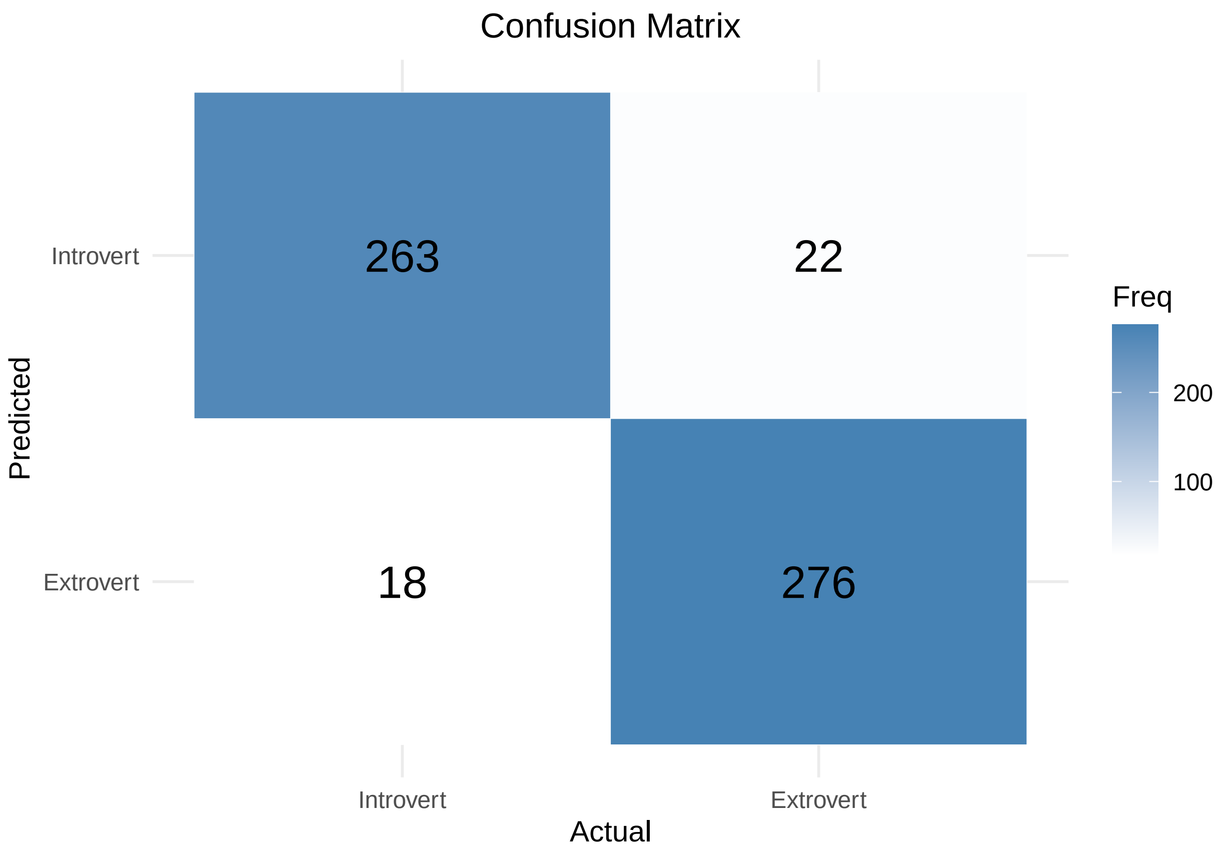
5.2 Random Forest

```
# Define training control with 10-fold CV
train_control <- trainControl(method = "cv", number = 10)

# Train random forest model with cross-validation
rf_model <- train(
  Personality ~ .,
  data = train_df,
  method = "rf",
  trControl = train_control,
  importance = TRUE
)

# Print results
print(rf_model)
```

```
## Random Forest
```

```
##
## 2321 samples
##    7 predictor
##    2 classes: 'Extrovert', 'Introvert'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 2089, 2089, 2089, 2089, 2089, 2088, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##   2     0.9353797  0.8707528
##   4     0.9181568  0.8362476
##   7     0.9129788  0.8258643
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

```
#varImp(rf_model)

# Predict on the training data (or ideally on a test set)
rf_predictions <- predict(rf_model, newdata = test_df)

# Create confusion matrix
rf_cm <- confusionMatrix(rf_predictions, factor(test_df$Personality))
rf_cm_table <- as.data.frame(rf_cm$table)
colnames(rf_cm_table) <- c("Prediction", "Reference", "Freq")

rf_cm_table$Reference <- factor(rf_cm_table$Reference, levels = rev(levels(rf_cm_table$Reference)))

ggplot(data = rf_cm_table, aes(x = Reference, y = Prediction, fill = Freq)) +
  geom_tile(color = "white") +
  geom_text(aes(label = Freq), color = "black", size = 6) +
  scale_fill_gradient(low = "white", high = "steelblue") +
  labs(title = "Confusion Matrix", x = "Actual", y = "Predicted") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```

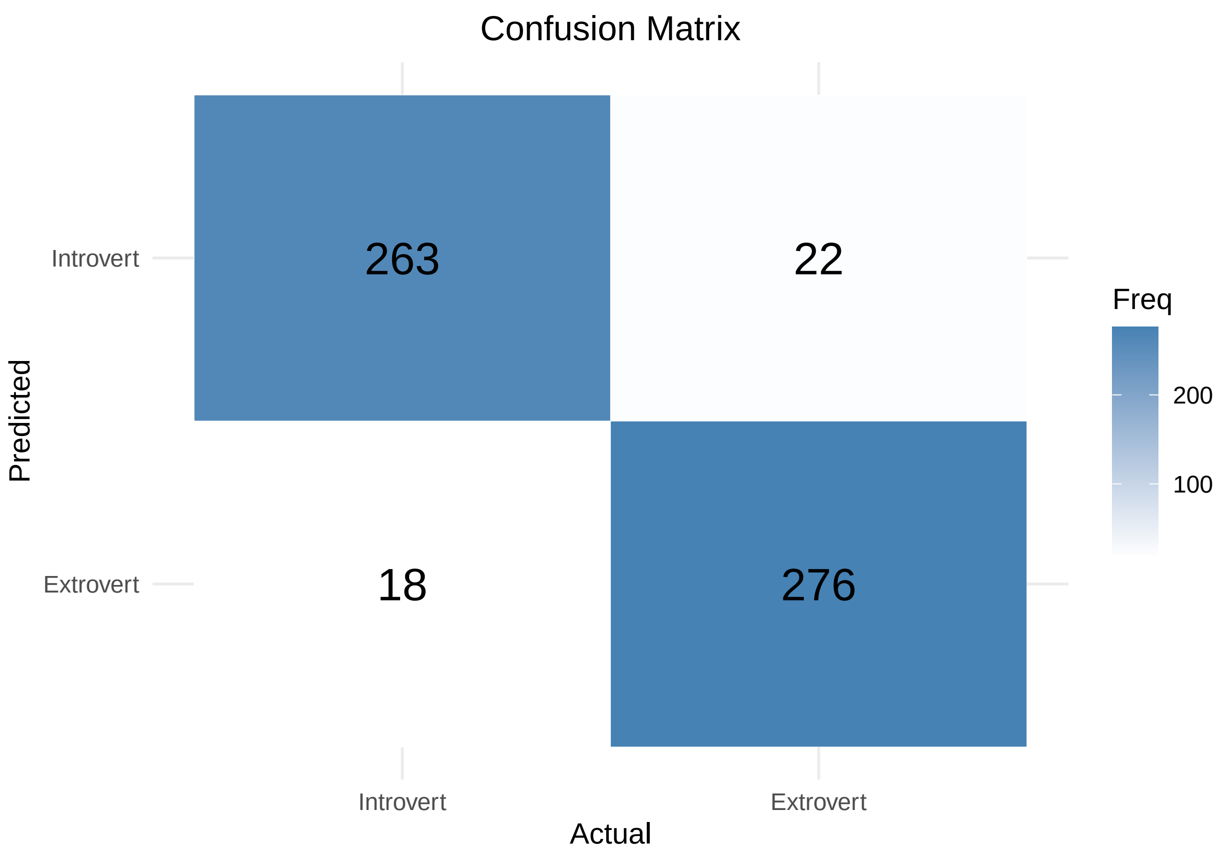


5.3 Elastic Net

```
# Define train control for 10-fold CV
train_control <- trainControl(method = "cv", number = 10)

# Train Elastic Net model
elastic_net_model <- train(
  factor(Personality) ~ .,
  data = train_df,
  method = "glmnet",     # method for elastic net
  trControl = train_control,
  tuneLength = 10        # try 10 different combinations of alpha and lambda
)

# View results
# print(elastic_net_model)

# Best tuning parameters
print(elastic_net_model$bestTune)
```

```
##    alpha    lambda
## 10   0.1 0.3674102
```

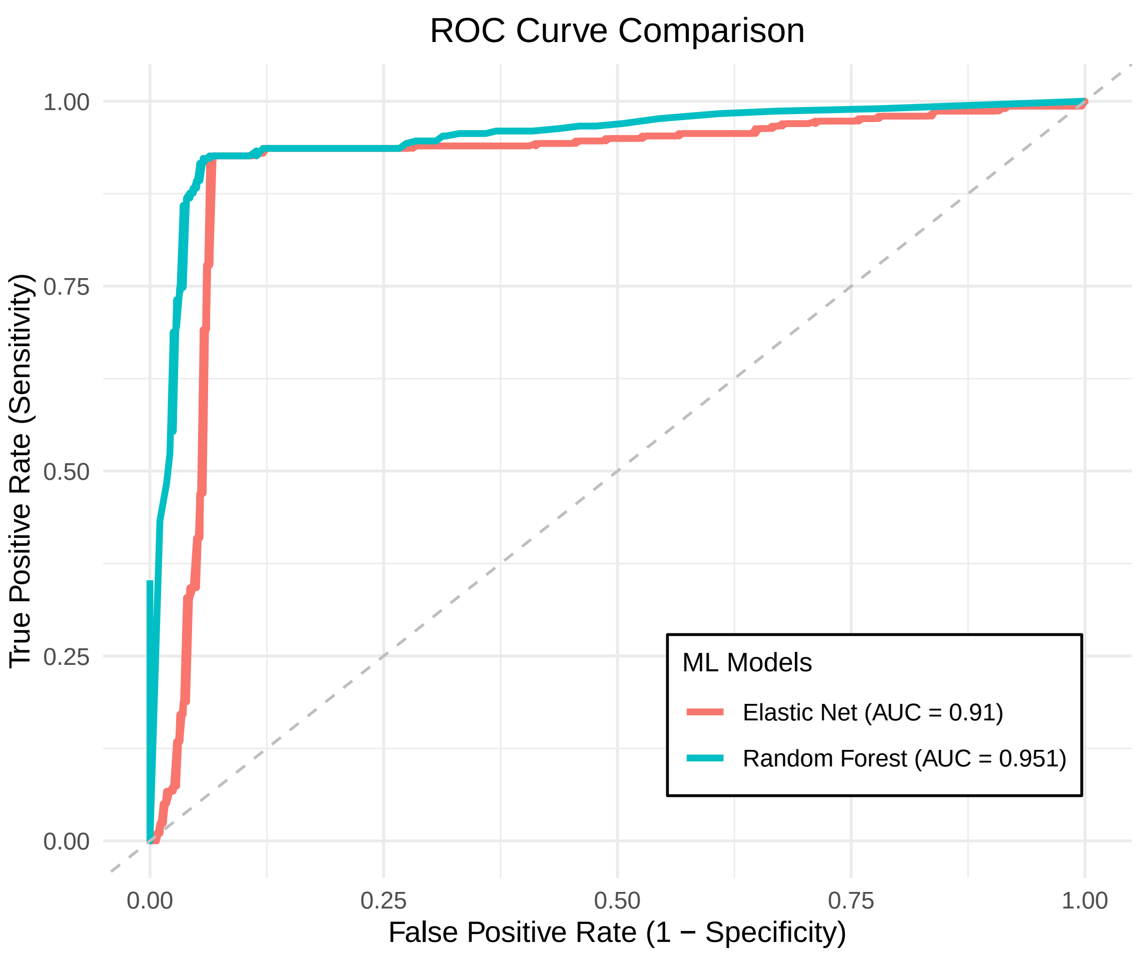
```r
# Predict on the training data (or a test set if you have one)
enet_predictions <- predict(elastic_net_model, newdata = test_df)

# Create confusion matrix
enet_cm <- confusionMatrix(enet_predictions, factor(test_df$Personality))
enet_cm_table <- as.data.frame(enet_cm$table)
colnames(enet_cm_table) <- c("Prediction", "Reference", "Freq")

enet_cm_table$Reference <- factor(enet_cm_table$Reference, levels = rev(levels(enet_cm_table$Reference)

ggplot(data = enet_cm_table, aes(x = Reference, y = Prediction, fill = Freq)) +
  geom_tile(color = "white") +
  geom_text(aes(label = Freq), color = "black", size = 6) +
  scale_fill_gradient(low = "white", high = "steelblue") +
  labs(title = "Confusion Matrix", x = "Actual", y = "Predicted") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```



5.4 ROC Curve Comparison

```r
# Calulating the probabilites for each model
rf_probs <- predict(rf_model, test_df, type = "prob")
enet_probs <- predict(elastic_net_model, test_df, type = "prob")

# Generate ROC curve objects
```

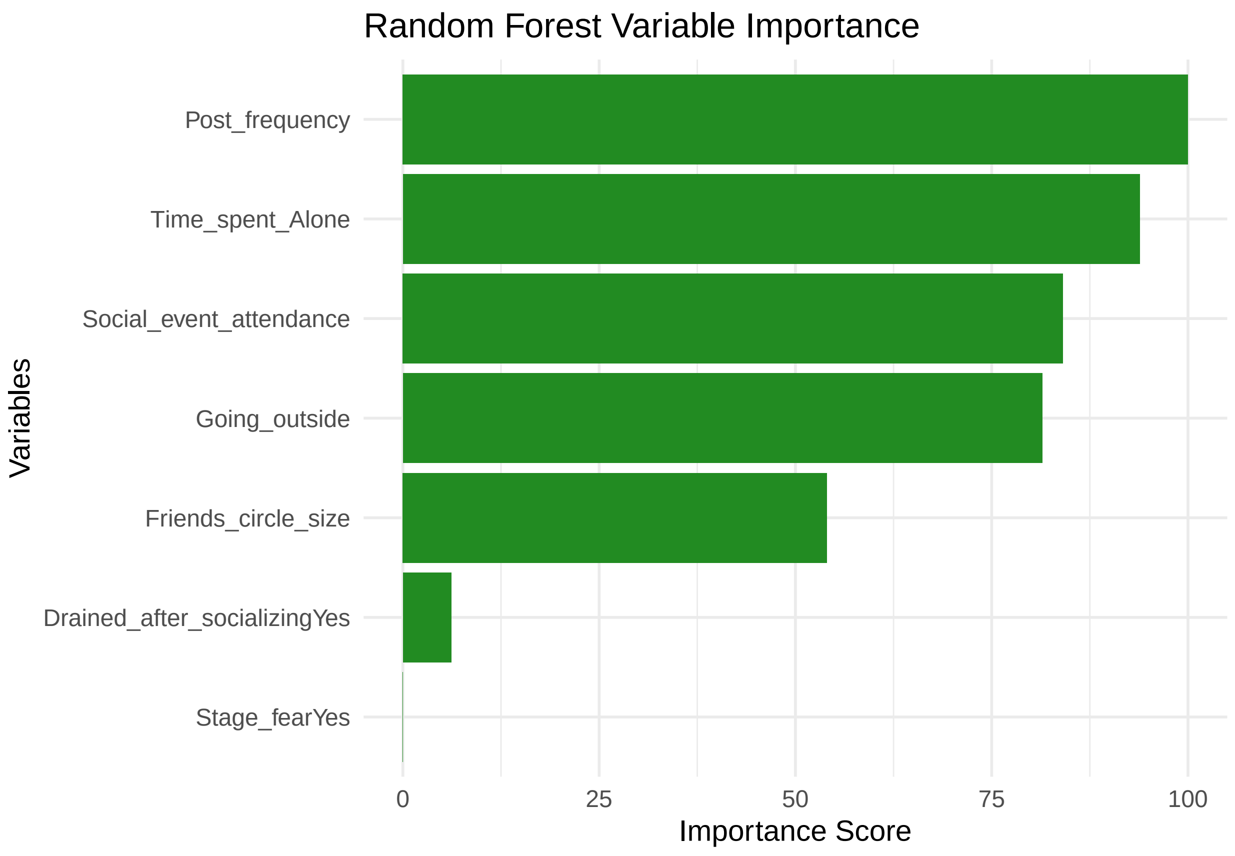
```r
rf_roc <- roc(test_df$Personality, rf_probs[,"Extrovert"], levels = c("Introvert", "Extrovert"))
enet_roc <- roc(test_df$Personality, enet_probs[,"Extrovert"], levels = c("Introvert", "Extrovert"))

# Create a ggplot-friendly data frame from both ROC curves
rf_df <- data.frame(
  fpr = 1 - rf_roc$specificities,
  tpr = rf_roc$sensitivities,
  model = "Random Forest"
)

enet_df <- data.frame(
  fpr = 1 - enet_roc$specificities,
  tpr = enet_roc$sensitivities,
  model = "Elastic Net"
)

# Calculate AUCs
rf_auc <- auc(rf_roc)
enet_auc <- auc(enet_roc)

# Create labels with AUC values
rf_label <- paste0("Random Forest (AUC = ", round(rf_auc, 3), ")")
enet_label <- paste0("Elastic Net (AUC = ", round(enet_auc, 3), ")")

# Update data frames with labeled model names
rf_df$model <- rf_label
enet_df$model <- enet_label

# Combine both ROC data frames
roc_df <- rbind(rf_df, enet_df)

ggplot(roc_df, aes(x = fpr, y = tpr, color = model)) +
  geom_line(size = 1.2) +
  geom_abline(linetype = "dashed", color = "gray") +
  labs(
    title = "ROC Curve Comparison",
    x = "False Positive Rate (1 - Specificity)",
    y = "True Positive Rate (Sensitivity)",
    color = "ML Models"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5),
    legend.position = "inside",
    legend.position.inside = c(0.75, 0.2),  # (x, y) from bottom-left corner
    legend.background = element_rect(fill = "white", color = "black"),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 9)
  )
```



6 Feature Importance

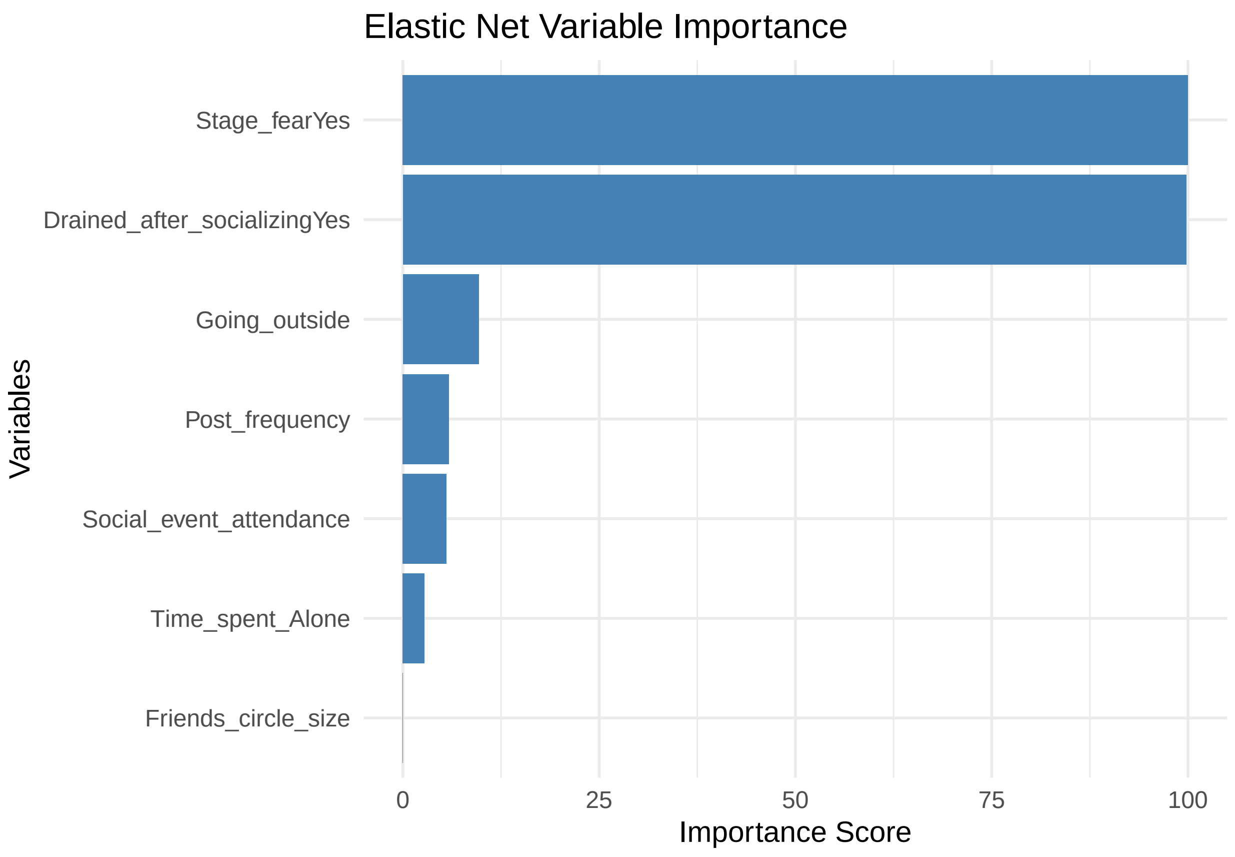
6.1 Random Forest feature importance

```r
# Extract variable importance
rf_imp <- varImp(rf_model)

# Extract importance and add variable names
rf_imp_df <- as.data.frame(rf_imp$importance)
rf_imp_df$Variable <- rownames(rf_imp_df)

# Compute row means as "Overall"
rf_imp_df$Overall <- rowMeans(rf_imp_df[, sapply(rf_imp_df, is.numeric)])

# Sort by Overall importance
library(dplyr)
rf_imp_df <- rf_imp_df %>%
  arrange(desc(Overall))

# Plot with ggplot2
ggplot(rf_imp_df, aes(x = reorder(Variable, Overall), y = Overall)) +
  geom_bar(stat = "identity", fill = "forestgreen") +
  coord_flip() +
  labs(
    title = "Random Forest Variable Importance",
    x = "Variables",
    y = "Importance Score"
  ) +
  theme_minimal()
```



6.2 Elastic Net feature importance

```r
# Extract variable importance
enet_imp <- varImp(elastic_net_model)

# Convert to data frame and add variable names
enet_df <- as.data.frame(enet_imp$importance)
enet_df$Variable <- rownames(enet_df)

# sort by importance
enet_df <- enet_df %>%
  arrange(desc(Overall))

# Plot with ggplot2
ggplot(enet_df, aes(x = reorder(Variable, Overall), y = Overall)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  coord_flip() +  # Flip axes for better readability
  labs(
    title = "Elastic Net Variable Importance",
    x = "Variables",
    y = "Importance Score"
  ) +
  theme_minimal()
```



7 Conclusion

7.1 Model Performace Table

```
# Confusion matrices
rf_cm <- confusionMatrix(rf_predictions, factor(test_df$Personality))
enet_cm <- confusionMatrix(enet_predictions, factor(test_df$Personality))

# AUC (assuming "Extrovert" is the positive class)
rf_auc <- auc(roc(test_df$Personality, rf_probs[,"Extrovert"]))
enet_auc <- auc(roc(test_df$Personality, enet_probs[,"Extrovert"]))

# F1 Scores (binary classification)
rf_f1 <- F1_Score(y_pred = rf_predictions, y_true = test_df$Personality, positive = "Extrovert")
enet_f1 <- F1_Score(y_pred = enet_predictions, y_true = test_df$Personality, positive = "Extrovert")
```

```r
# Create summary data frame
results_df <- data.frame(
  Model = c("Random Forest", "Elastic Net"),
  Accuracy = c(rf_cm$overall["Accuracy"], enet_cm$overall["Accuracy"]),
  AUC = c(rf_auc, enet_auc),
  F1_Score = c(rf_f1, enet_f1),
  Precision = c(rf_cm$byClass["Precision"], enet_cm$byClass["Precision"]),
  Recall = c(rf_cm$byClass["Recall"], enet_cm$byClass["Recall"]),
  Specificity = c(rf_cm$byClass["Specificity"], enet_cm$byClass["Specificity"])
)

# Round for cleaner display
results_df[ , -1] <- round(results_df[ , -1], 4)

# Print
kable(results_df,"latex",
 align="ccccccc",
 caption="Model Performance"
 )%>%
 kable_styling(latex_options="striped")%>%
 kable_styling(latex_options="HOLD_position")%>%
 kable_styling(latex_options="repeat_header")%>%
 kable_styling(font_size=12)
```

Table 1: Model Performance

Model	Accuracy	AUC	F1_Score	Precision	Recall	Specificity
Random Forest	0.9309	0.9512	0.9324	0.9388	0.9262	0.9359
Elastic Net	0.9309	0.9102	0.9324	0.9388	0.9262	0.9359

7.2 Different cutoff performce for each ML model

```r
enet_probs <- predict(elastic_net_model, test_df, type = "prob")

cutoffs <- seq(0.1, 0.9, by = 0.05)
enet_results <- data.frame()

for (cut in cutoffs) {
  preds <- ifelse(enet_probs$Extrovert >= cut, "Extrovert", "Introvert")
  preds <- factor(preds, levels = c("Extrovert", "Introvert"))
  reference <- factor(test_df$Personality, levels = c("Extrovert", "Introvert"))

  cm <- confusionMatrix(preds, reference)

  enet_results <- rbind(enet_results, data.frame(
  Cutoff = cut,
  Accuracy = cm$overall['Accuracy'],
  Sensitivity = cm$byClass['Sensitivity'],
  Specificity = cm$byClass['Specificity']
), row.names = NULL)
}
```

```r
rownames(enet_results) <- NULL

kable(enet_results,"latex",
 align="cccc",
 caption="Elastic Net Model Performance at Different Cutoffs"
 )%>%
 kable_styling(latex_options="striped")%>%
 kable_styling(latex_options="HOLD_position")%>%
 kable_styling(latex_options="repeat_header")%>%
 kable_styling(font_size=12)
```

Table 2: Elastic Net Model Performance at Different Cutoffs

Cutoff	Accuracy	Sensitivity	Specificity
0.10	0.5146805	1.0000000	0.0000000
0.15	0.5129534	0.9966443	0.0000000
0.20	0.6286701	0.9731544	0.2633452
0.25	0.8428325	0.9362416	0.7437722
0.30	0.9101900	0.9261745	0.8932384
0.35	0.9205527	0.9261745	0.9145907
0.40	0.9274611	0.9261745	0.9288256
0.45	0.9291883	0.9261745	0.9323843
0.50	0.9309154	0.9261745	0.9359431
0.55	0.9309154	0.9261745	0.9359431
0.60	0.9309154	0.9261745	0.9359431
0.65	0.9309154	0.9261745	0.9359431
0.70	0.9309154	0.9261745	0.9359431
0.75	0.8911917	0.8489933	0.9359431
0.80	0.7340242	0.5369128	0.9430605
0.85	0.5457686	0.1476510	0.9679715
0.90	0.4853195	0.0000000	1.0000000

```r
rf_probs <- predict(rf_model, test_df, type = "prob")
cutoffs <- seq(0.1, 0.9, by = 0.05)
rf_results <- data.frame()

for (cut in cutoffs) {
  preds <- ifelse(rf_probs$Extrovert >= cut, "Extrovert", "Introvert")
  preds <- factor(preds, levels = c("Extrovert", "Introvert"))
  reference <- factor(test_df$Personality, levels = c("Extrovert", "Introvert"))

  cm <- confusionMatrix(preds, reference)

  rf_results <- rbind(rf_results, data.frame(
  Cutoff = cut,
  Accuracy = cm$overall['Accuracy'],
  Sensitivity = cm$byClass['Sensitivity'],
  Specificity = cm$byClass['Specificity']
), row.names = NULL)
}
rownames(rf_results) <- NULL

kable(rf_results,"latex",
 align="cccc",
 caption="Random Forest Model Performance at Different Cutoffs"
 )%>%
 kable_styling(latex_options="striped")%>%
 kable_styling(latex_options="HOLD_position")%>%
 kable_styling(latex_options="repeat_header")%>%
 kable_styling(font_size=12)
```

Table 3: Random Forest Model Performance at Different Cutoffs

Cutoff	Accuracy	Sensitivity	Specificity
0.10	0.8911917	0.9362416	0.8434164
0.15	0.9153713	0.9261745	0.9039146
0.20	0.9222798	0.9261745	0.9181495
0.25	0.9257340	0.9261745	0.9252669
0.30	0.9274611	0.9261745	0.9288256
0.35	0.9274611	0.9261745	0.9288256
0.40	0.9291883	0.9261745	0.9323843
0.45	0.9309154	0.9261745	0.9359431
0.50	0.9309154	0.9261745	0.9359431
0.55	0.9309154	0.9261745	0.9359431
0.60	0.9309154	0.9261745	0.9359431
0.65	0.9291883	0.9228188	0.9359431
0.70	0.9291883	0.9228188	0.9359431
0.75	0.9326425	0.9228188	0.9430605
0.80	0.9291883	0.9127517	0.9466192
0.85	0.9188256	0.8892617	0.9501779
0.90	0.9101900	0.8590604	0.9644128

7.3 Final remarks

This project demonstrates a robust approach to personality classification using exploratory data analysis (EDA), preprocessing, and machine learning models—Elastic Net and Random Forest. Key features such as **Post_frequency** and **Time_spent_alone** were particularly influential in the Random Forest model, while **Stage_fear** and **Drained_after_socializing** were important for the Elastic Net model, as indicated by feature importance analysis. Both models achieved strong performance, with Random Forest reaching an accuracy of 0.9309 and an F1 score of 0.9324. However, when evaluating performance across different cutoffs, Random Forest exhibited significantly greater stability compared to Elastic Net, as further reflected in its higher AUC.