Group1_VictorTuytte_LucaVisser

Victor Tuytte, Luca Visser

Binary Response

Setup

Helper Functions

```r
# Function to compute ROC curve and AUC
compute_roc_auc <- function(actual, predicted_probs) {
  roc_curve <- roc(actual, predicted_probs, levels = c(0, 1), direction = "<")
  auc_value <- auc(roc_curve)
  return(list(roc_curve = roc_curve, auc_value = auc_value))
}

# Function to plot ROC curves for train and test
plot_roc_curve <- function(roc_curve_train, roc_curve_test, auc_value_train, auc_value_test) {
  # Set up plot layout to have extra space for text
  par(mfrow = c(1, 1), mar = c(5, 4, 4, 6))  # Adjust margins for extra space on the right

  # Plot ROC curves for both training and testing sets
  plot(roc_curve_train, col = "blue", main = "", lwd = 2, xlim = c(0, 1), ylim = c(0, 1))
  lines(roc_curve_test, col = "red", lwd = 2)

  # Add AUC and other metrics as a legend on the plot
  legend("bottomright",
         legend = c(paste("Train AUC =", round(auc_value_train, 2)),
                    paste("Test AUC =", round(auc_value_test, 2))),
         col = c("blue", "red"), lwd = 2, bty = "n", cex = 0.8)
}

evaluate_binary_model_performance <- function(model, train_data, test_data, response_var, confidence_le
  # Predict probabilities on the training and test sets
  train_pred_probs <- predict(model, newdata = train_data, type = "response")
  test_pred_probs <- predict(model, newdata = test_data, type = "response")

  # Convert probabilities to binary predictions
  train_pred_binary <- ifelse(train_pred_probs > tau, 1, 0)
  test_pred_binary <- ifelse(test_pred_probs > tau, 1, 0)

  # Convert actual and predicted to factors with consistent levels
  actual_train <- factor(train_data[[response_var]], levels = c(0, 1))
  predicted_train <- factor(train_pred_binary, levels = c(0, 1))

  actual_test <- factor(test_data[[response_var]], levels = c(0, 1))
  predicted_test <- factor(test_pred_binary, levels = c(0, 1))
```

```r
  # Generate confusion matrix
  cm <- confusionMatrix(predicted_test, actual_test, positive = "1")

  # Calculate sensitivity and specificity with confidence intervals
  sensitivity <- cm$byClass["Sensitivity"]
  specificity <- cm$byClass["Specificity"]
  n_positive <- sum(actual_test == 1)
  n_negative <- sum(actual_test == 0)

  # Confidence intervals for sensitivity and specificity
  sensitivity_ci <- binom.test(round(sensitivity * n_positive), n_positive, conf.level = confidence_leve
  specificity_ci <- binom.test(round(specificity * n_negative), n_negative, conf.level = confidence_leve

  # Display results with confidence intervals
  cat("Sensitivity:", sensitivity, "\n")
  cat("Sensitivity CI:", sensitivity_ci, "\n")
  cat("Specificity:", specificity, "\n")
  cat("Specificity CI:", specificity_ci, "\n")

  # Compute ROC and AUC for training and testing data
  train_roc_auc <- compute_roc_auc(actual_train, train_pred_probs)
  test_roc_auc <- compute_roc_auc(actual_test, test_pred_probs)

  # Plot ROC curve and add metrics
  plot_roc_curve(train_roc_auc$roc_curve, test_roc_auc$roc_curve,
                 train_roc_auc$auc_value, test_roc_auc$auc_value)
}
```

Data Import

```r
support <- as.data.frame(read_dta("support.dta")) %>%
  mutate(sex = as.factor(sex), dzclass = as.factor(dzclass), hospdead = as.factor(hospdead))

key_predictors <- c("age", "sex", "dzclass", "num_co", "edu", "slos")
head(support)
```

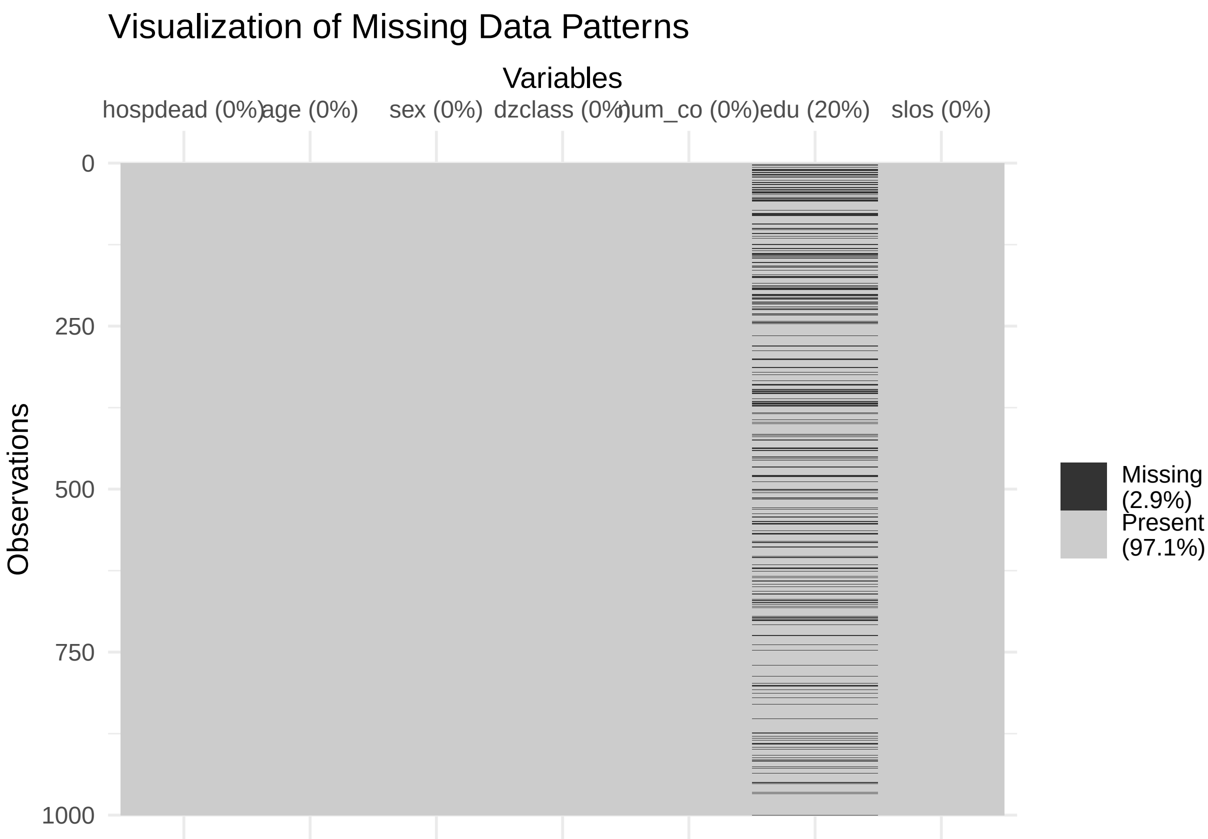
```
##        age death sex hospdead slos d_time dzgroup dzclass num_co edu income
## 1 85.65594     1   2        0   12     63       7       4      2  12  NA(z)
## 2 42.25897     1   1        0    8    370       6       4      0  11      3
## 3 43.53998     0   1        0  115   2022       1       1      1  NA  NA(z)
## 4 45.41800     1   2        0    7    827       7       4      2  NA  NA(z)
## 5 63.66299     1   1        1   14     14       1       1      0  22      3
## 6 41.52197     1   2        1   21     21       8       1      2  18      4
##   scoma charges   totcst totmcst  avtisst race meanbp       wblc hrt resp
## 1    26      NA       NA      NA  8.50000    2     97  9.6992188  56   20
## 2     0    9914       NA      NA  8.00000    5     84 11.2988281  94   20
## 3    26  706577 390460.5      NA 38.25000    1     67 24.5976562 172   20
## 4     0    8400       NA      NA  7.00000    1     97 10.7988281 100   24
## 5    26  283303 156674.1      NA 40.00000    1     69 30.0976562 108   22
## 6    26  320843 165178.9      NA 32.66666    1     66  0.1999817 130   18
##       temp     pafi      alb      bili      crea sod       ph glucose bun
## 1 36.59375 357.1250       NA 0.3999634 1.0000000 143 7.449219      NA  NA
## 2 38.19531       NA 4.699219 0.1999817 0.7999268 139       NA      NA  NA
```

```
## 3 38.79688 113.3281       NA         NA 0.5999756 134 7.399414      NA  NA
## 4 37.29688 585.1250       NA  0.2999878 1.0998535 137 7.489258      NA  NA
## 5 36.69531 155.5312 2.899902 14.0000000 2.8999023 130 7.449219      NA  NA
## 6 37.50000 315.0000 1.899902  3.7998047 7.2998047 134 7.359375      NA  NA
##   urine adlp adls sfdm2     adlsc
## 1    NA   NA    7 NA(z) 7.0000000
## 2    NA    0   NA NA(z) 0.4947999
## 3    NA   NA   NA     3 2.7641602
## 4    NA   NA   NA NA(z) 3.3515625
## 5    NA   NA    0     5 0.0000000
## 6    NA   NA    0     5 0.0000000
```

Data Exploration

Missing Data

```
# Visualize missingness in combinations
vis_miss(support[, c("hospdead", key_predictors)]) +
  labs(title = "Visualization of Missing Data Patterns",
       x = "Variables",
       y = "Observations") +
  theme_minimal()
```



Of the key predictors, only `edu` displays missingness (20% of observations).

```
support$edu_missing <- is.na(support$edu)

# Fit a logistic regression model to predict missingness in 'edu'
```

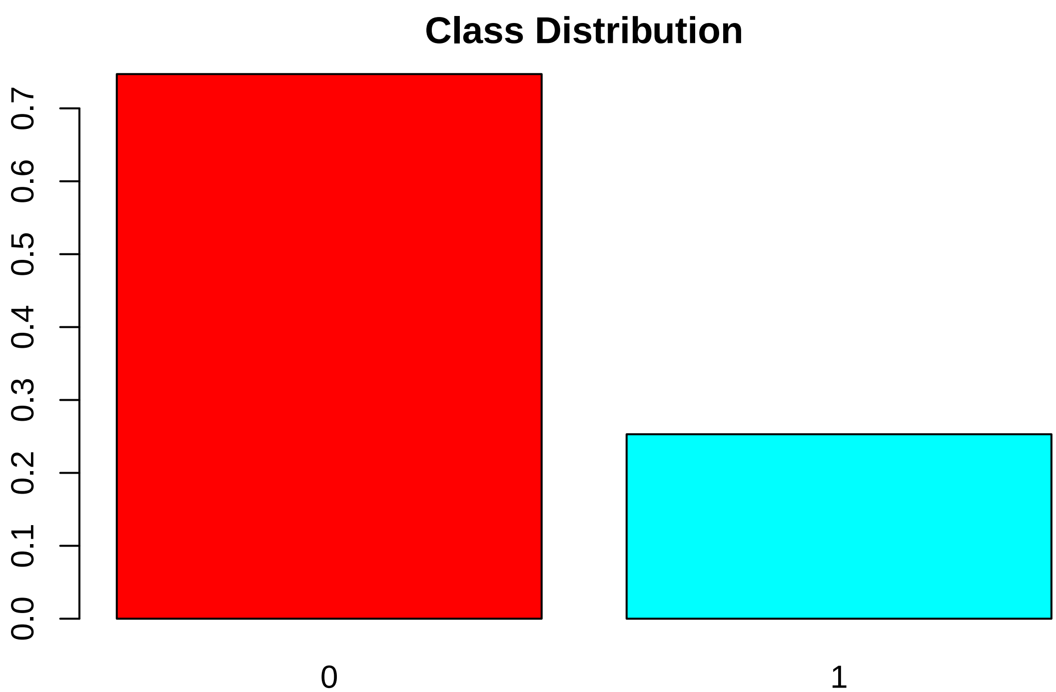
```r
logistic_model <- glm(edu_missing ~ age + sex + dzclass + num_co + slos + hospdead,
                      data = support,
                      family = binomial)

# Summarize the logistic regression model
summary(logistic_model)
```

```
##
## Call:
## glm(formula = edu_missing ~ age + sex + dzclass + num_co + slos +
##     hospdead, family = binomial, data = support)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.091799   0.376248  -5.560  2.7e-08 ***
## age          0.011026   0.005077   2.172   0.0299 *
## sex2         0.156057   0.161927   0.964   0.3352
## dzclass2    -0.309697   0.213968  -1.447   0.1478
## dzclass3     0.166344   0.314920   0.528   0.5974
## dzclass4    -0.281503   0.257413  -1.094   0.2741
## num_co      -0.053630   0.066620  -0.805   0.4208
## slos         0.002179   0.003480   0.626   0.5312
## hospdead1    0.388449   0.183562   2.116   0.0343 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1006.33  on 999  degrees of freedom
## Residual deviance:  986.84  on 991  degrees of freedom
## AIC: 1004.8
##
## Number of Fisher Scoring iterations: 4
```

Based on the model fit, the missingness seems not to be completely at random, but rather dependent slightly dependent on the `hospdead` and `age` variables. Nevertheless, we opt to delete the observations associated with missing values.

Class Imbalance

```r
barplot(prop.table(table(support$hospdead)),
        col = rainbow(2),
        ylim = c(0, 0.7),
        main = "Class Distribution")
```



Evidently, the data is inbalanced. If we were to train a model on this data using a random split (without stratified sampling), we would obtain a model highly biased towards predicting the control label (many false negatives). Oversampling offers a solution to this problem.

Predictor Importance

```r
# Summarize baseline characteristics by `hospdead` groups
summary_table <- support %>%
  group_by(hospdead) %>%
  summarise(
    age_mean = mean(age, na.rm = TRUE),
    age_sd = sd(age, na.rm = TRUE),
    sex_male_prop = mean(sex == 2, na.rm = TRUE),
    num_co_mean = mean(num_co, na.rm = TRUE),
    num_co_sd = sd(num_co, na.rm = TRUE),
    edu_mean = mean(edu, na.rm = TRUE),
    edu_sd = sd(edu, na.rm = TRUE),
    slos_mean = mean(slos, na.rm = TRUE),
    slos_sd = sd(slos, na.rm = TRUE),
  ) %>%
  mutate_if(is.numeric, round, digits = 2)

print(summary_table)
```

```
## # A tibble: 2 x 10
##   hospdead age_mean age_sd sex_male_prop num_co_mean num_co_sd edu_mean edu_sd
##   <fct>       <dbl>  <dbl>         <dbl>       <dbl>     <dbl>    <dbl>  <dbl>
## 1 0            62.5   15.7          0.56        1.94      1.34     11.6   3.62
## 2 1            62.5   17.3          0.57        1.74      1.36     12.3   3.52
## # i 2 more variables: slos_mean <dbl>, slos_sd <dbl>
```

```r
# Perform t-tests for continuous variables between `hospdead` groups
age_ttest <- t.test(age ~ hospdead, data = support)
```

```r
num_co_ttest <- t.test(num_co ~ hospdead, data = support)
edu_ttest <- t.test(edu ~ hospdead, data = support)
slos_ttest <- t.test(slos ~ hospdead, data = support)

# Perform chi-square tests for categorical variables between `hospdead` groups
sex_chisq <- chisq.test(table(support$sex, support$hospdead))
dzclass_chisq <- chisq.test(table(support$dzclass, support$hospdead))

# Extract p-values and test names into a data frame
results <- data.frame(
  Variable = c("Age", "Number of Comorbidities", "Education", "Sex", "Disease Class", "SLOS"),
  p_value = c(age_ttest$p.value, num_co_ttest$p.value, edu_ttest$p.value,
              sex_chisq$p.value, dzclass_chisq$p.value, slos_ttest$p.value)
)

# Format p-values to 2 decimal places
results$p_value <- round(results$p_value, 3)

results$p_value <- format.pval(results$p_value)

# Rank variables by p-value (ascending)
results <- results[order(results$p_value), ]

# Display the results
print(results)
```

```
##                  Variable p_value
## 5           Disease Class  <2e-16
## 3               Education   0.020
## 2 Number of Comorbidities   0.041
## 6                    SLOS   0.733
## 4                     Sex   0.847
## 1                     Age   0.971
```

In a univariate setting, education level, number of comorbidities, and disease class show significant associations with in-hospital death. The univariates age, sex and SLOS do not appear to have a statistically significant relationship with in-hospital death in this dataset.

Modelling

```r
processed_data <- support %>%
  select(hospdead, key_predictors) %>%
  na.omit()

# Set seed for reproducibility
set.seed(123)

# Split the filtered data into training and testing sets
train_index <- as.vector(createDataPartition(processed_data$hospdead, p = 0.8, list = FALSE))
train_data <- processed_data[train_index, ]
test_data <- processed_data[-train_index, ]

prop.table(table(train_data$hospdead))
```

```
##
##         0         1
## 0.7683881 0.2316119
```

```r
prop.table(table(test_data$hospdead))
```

```
##
##         0         1
## 0.7672956 0.2327044
```

```r
# Get the size of the training dataset
train_size <- nrow(train_data)

# Perform oversampling on the training data
over <- ovun.sample(hospdead ~ num_co + age + sex + dzclass + edu + slos,
                    data = train_data,
                    method = "over",
                    N = 2 * sum(train_data$hospdead == 0))$data  # Target 50/50 distribution
prop.table(table(over$hospdead))
```

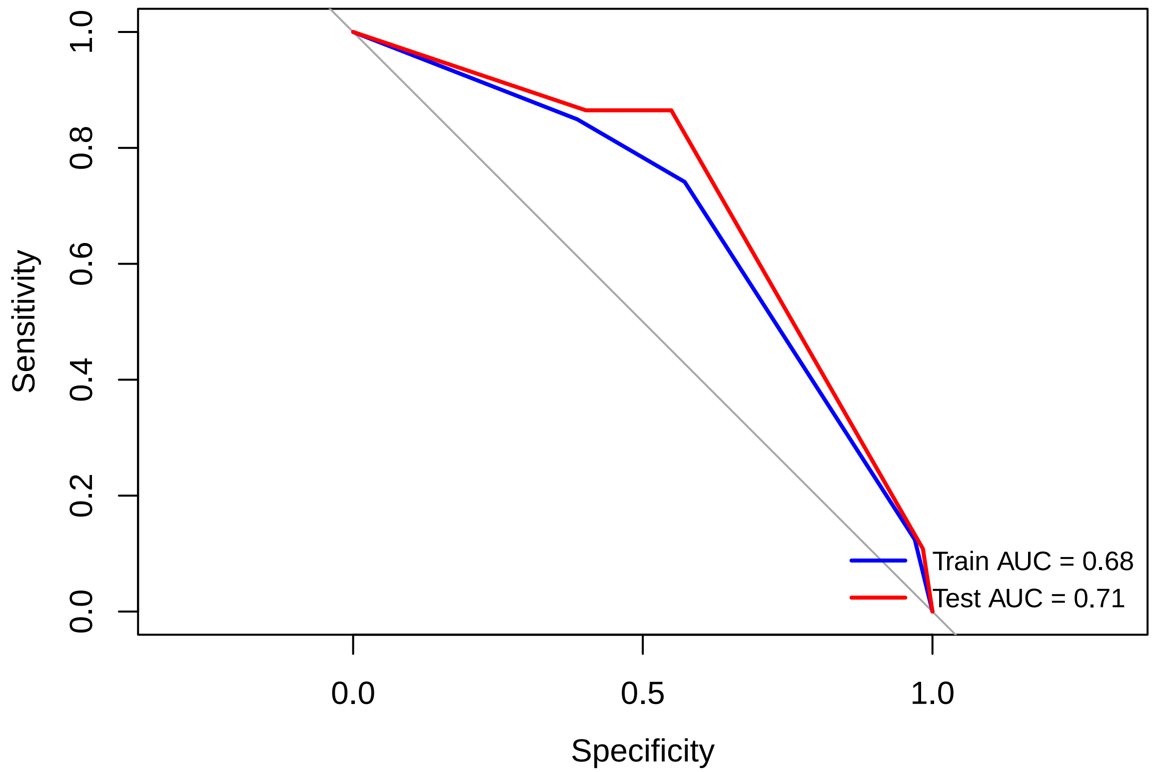
```
##
##   0   1
## 0.5 0.5
```

```r
model1_binary <- glm(hospdead ~ dzclass, data = over, family = binomial)
summary(model1_binary)
```

```
##
## Call:
## glm(formula = hospdead ~ dzclass, family = binomial, data = over)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.44073    0.09181   4.801 1.58e-06 ***
## dzclass2    -1.38369    0.16494  -8.389  < 2e-16 ***
## dzclass3     0.96209    0.30247   3.181  0.00147 **
## dzclass4    -0.98130    0.19567  -5.015 5.30e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1361.3  on 981  degrees of freedom
## Residual deviance: 1245.0  on 978  degrees of freedom
## AIC: 1253
##
## Number of Fisher Scoring iterations: 4
```

```r
evaluate_binary_model_performance(model1_binary, over, test_data, "hospdead")
```

```
## Sensitivity: 0.8648649
## Sensitivity CI: 0.7122522 0.954628
## Specificity: 0.5491803
## Specificity CI: 0.4565288 0.6393882
```

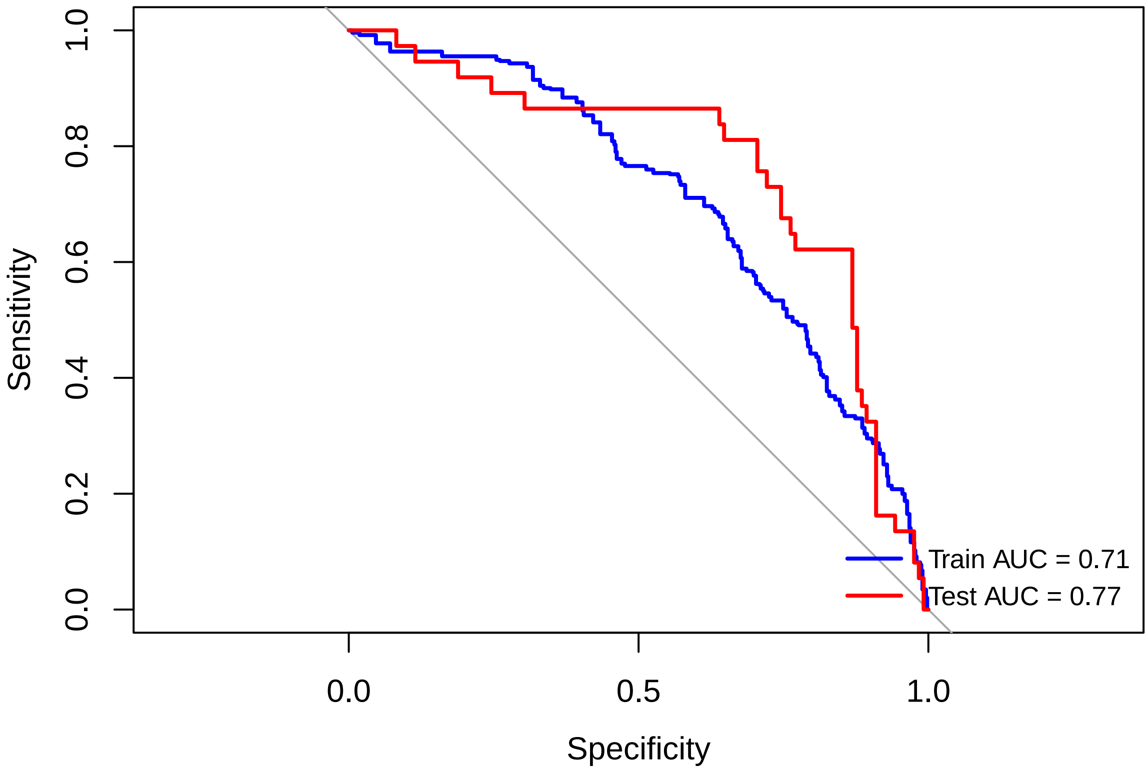



```r
model2_binary <- glm(hospdead ~ num_co + age + sex + dzclass + edu, data = over, family = binomial)
summary(model2_binary)
```

```
##
## Call:
## glm(formula = hospdead ~ num_co + age + sex + dzclass + edu,
##     family = binomial, data = over)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.614802   0.403044  -1.525 0.127160
## num_co       0.183256   0.053323   3.437 0.000589 ***
## age          0.001062   0.004217   0.252 0.801071
## sex2        -0.039035   0.138569  -0.282 0.778172
## dzclass2    -1.552074   0.178133  -8.713  < 2e-16 ***
## dzclass3     1.019978   0.305683   3.337 0.000848 ***
## dzclass4    -0.977581   0.197919  -4.939 7.84e-07 ***
## edu          0.058346   0.020046   2.911 0.003607 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1361.3  on 981  degrees of freedom
## Residual deviance: 1226.8  on 974  degrees of freedom
## AIC: 1242.8
##
## Number of Fisher Scoring iterations: 4
```

```r
evaluate_binary_model_performance(model2_binary, over, test_data, "hospdead")
```

```
## Sensitivity: 0.8648649
## Sensitivity CI: 0.7122522 0.954628
## Specificity: 0.5901639
## Specificity CI: 0.4974962 0.6783489
```



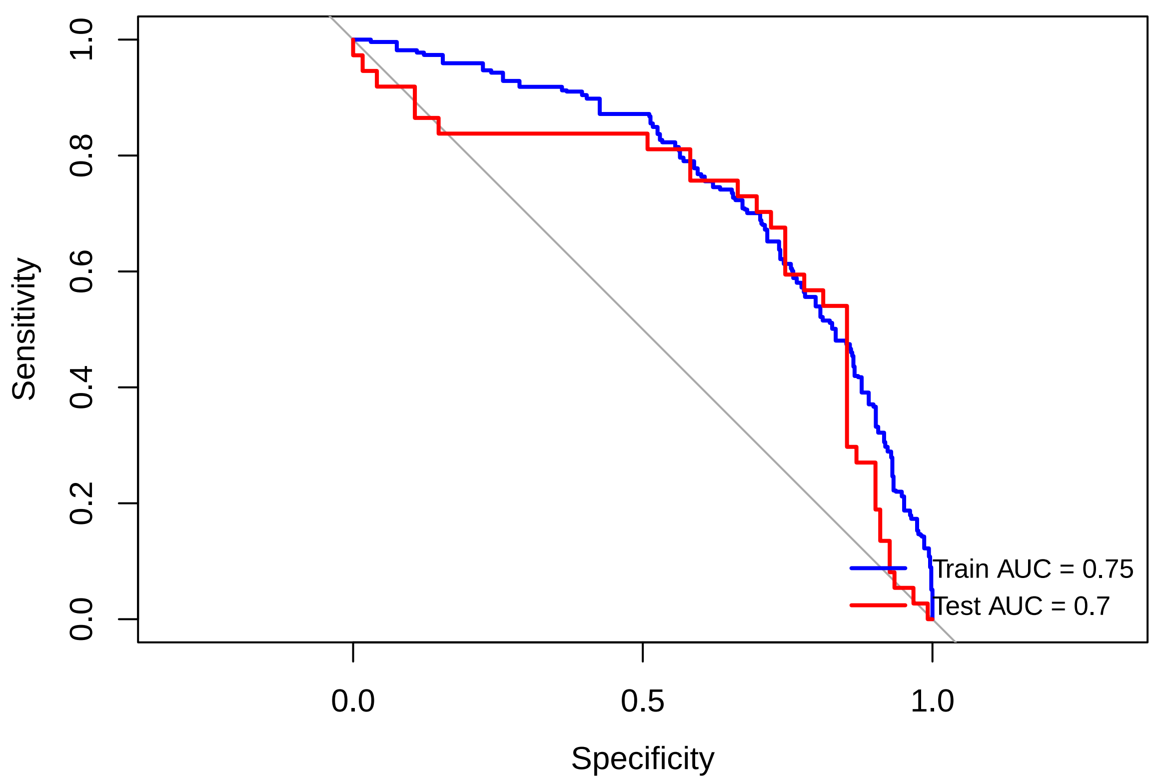

```r
model3_binary <- glm(hospdead ~ (num_co + age + sex + dzclass + edu)^2, data = over, family = binomial)
summary(model3_binary)
```

```
##
## Call:
## glm(formula = hospdead ~ (num_co + age + sex + dzclass + edu)^2,
##     family = binomial, data = over)
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -4.181186   1.712183  -2.442  0.01461 *
## num_co            0.759238   0.346343   2.192  0.02837 *
## age               0.069021   0.022479   3.070  0.00214 **
## sex2              0.400893   0.882249   0.454  0.64954
## dzclass2         -0.587090   1.207814  -0.486  0.62691
## dzclass3          4.221239   2.123060   1.988  0.04678 *
## dzclass4          2.845646   1.513814   1.880  0.06014 .
## edu               0.118149   0.118691   0.995  0.31953
## num_co:age       -0.009648   0.003549  -2.719  0.00655 **
## num_co:sex2      -0.134557   0.121404  -1.108  0.26772
## num_co:dzclass2 -0.083284   0.133192  -0.625  0.53178
## num_co:dzclass3 -0.046616   0.415776  -0.112  0.91073
## num_co:dzclass4  0.032627   0.274481   0.119  0.90538
```

```
## num_co:edu       0.013393   0.018696   0.716  0.47377
## age:sex2        -0.022727   0.009191  -2.473  0.01341 *
## age:dzclass2    -0.027354   0.012370  -2.211  0.02702 *
## age:dzclass3    -0.091098   0.021978  -4.145  3.4e-05 ***
## age:dzclass4    -0.029158   0.017258  -1.690  0.09111 .
## age:edu         -0.002172   0.001540  -1.410  0.15848
## sex2:dzclass2    0.150215   0.387813   0.387  0.69851
## sex2:dzclass3    2.466939   0.824838   2.991  0.00278 **
## sex2:dzclass4   -0.764105   0.430731  -1.774  0.07607 .
## sex2:edu         0.101845   0.046120   2.208  0.02723 *
## dzclass2:edu     0.065364   0.054994   1.189  0.23461
## dzclass3:edu     0.118250   0.140806   0.840  0.40102
## dzclass4:edu    -0.129639   0.062066  -2.089  0.03673 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1361.3  on 981  degrees of freedom
## Residual deviance: 1149.3  on 956  degrees of freedom
## AIC: 1201.3
##
## Number of Fisher Scoring iterations: 5
```

```r
evaluate_binary_model_performance(model3_binary, over, test_data, "hospdead")
```

```
## Sensitivity: 0.7567568
## Sensitivity CI: 0.5880083 0.8822748
## Specificity: 0.6147541
## Specificity CI: 0.5223685 0.701431
```



```r
# Extract the summary
model_summary <- summary(model3_binary)

# Get names of significant coefficients with p-value < 0.05, excluding "(Intercept)"
significant_terms <- rownames(model_summary$coefficients)[
  model_summary$coefficients[, 4] < 0.05 & rownames(model_summary$coefficients) != "(Intercept)"
]

print(significant_terms)
```

```
##  [1] "num_co"        "age"           "dzclass3"      "num_co:age"
##  [5] "age:sex2"      "age:dzclass2"  "age:dzclass3"  "sex2:dzclass3"
##  [9] "sex2:edu"      "dzclass4:edu"
```

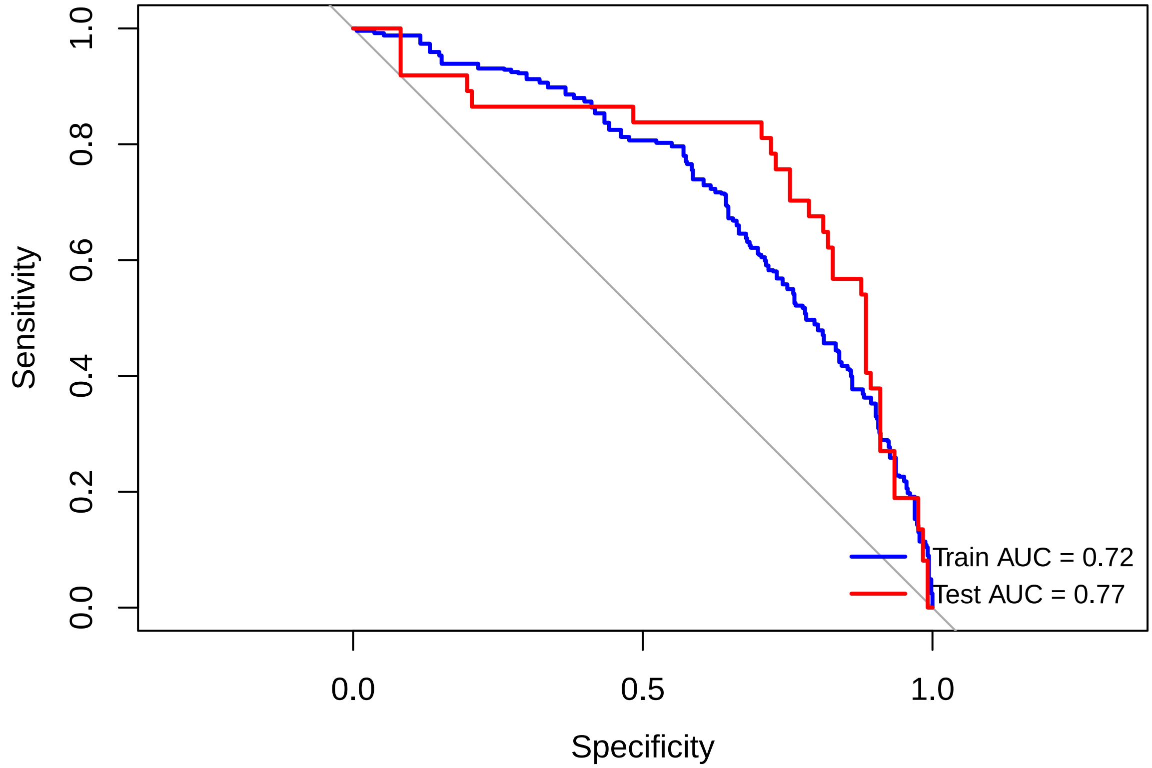
```r
# Create a new formula with only significant terms
new_formula <- as.formula(paste("hospdead ~", paste(c("sex", "edu", "age", "num_co", "age*dzclass"), co

# Refit the model with only significant terms
final_binary_model <- glm(new_formula, family = binomial, data = over)

# Display summary of the final model
summary(final_binary_model)
```

```
##
## Call:
## glm(formula = new_formula, family = binomial, data = over)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.593958   0.453046  -3.518 0.000434 ***
## sex2         -0.019225   0.141386  -0.136 0.891839
## edu           0.055398   0.020428   2.712 0.006691 **
## age           0.017039   0.005343   3.189 0.001428 **
## num_co        0.205035   0.054685   3.749 0.000177 ***
## dzclass2      1.213337   0.690438   1.757 0.078859 .
## dzclass3      4.823372   1.211399   3.982 6.84e-05 ***
## dzclass4      0.401548   1.035458   0.388 0.698166
## age:dzclass2 -0.045021   0.010923  -4.122 3.76e-05 ***
## age:dzclass3 -0.062310   0.017848  -3.491 0.000481 ***
## age:dzclass4 -0.022589   0.016692  -1.353 0.175950
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1361.3  on 981  degrees of freedom
## Residual deviance: 1198.7  on 971  degrees of freedom
## AIC: 1220.7
##
## Number of Fisher Scoring iterations: 4
```

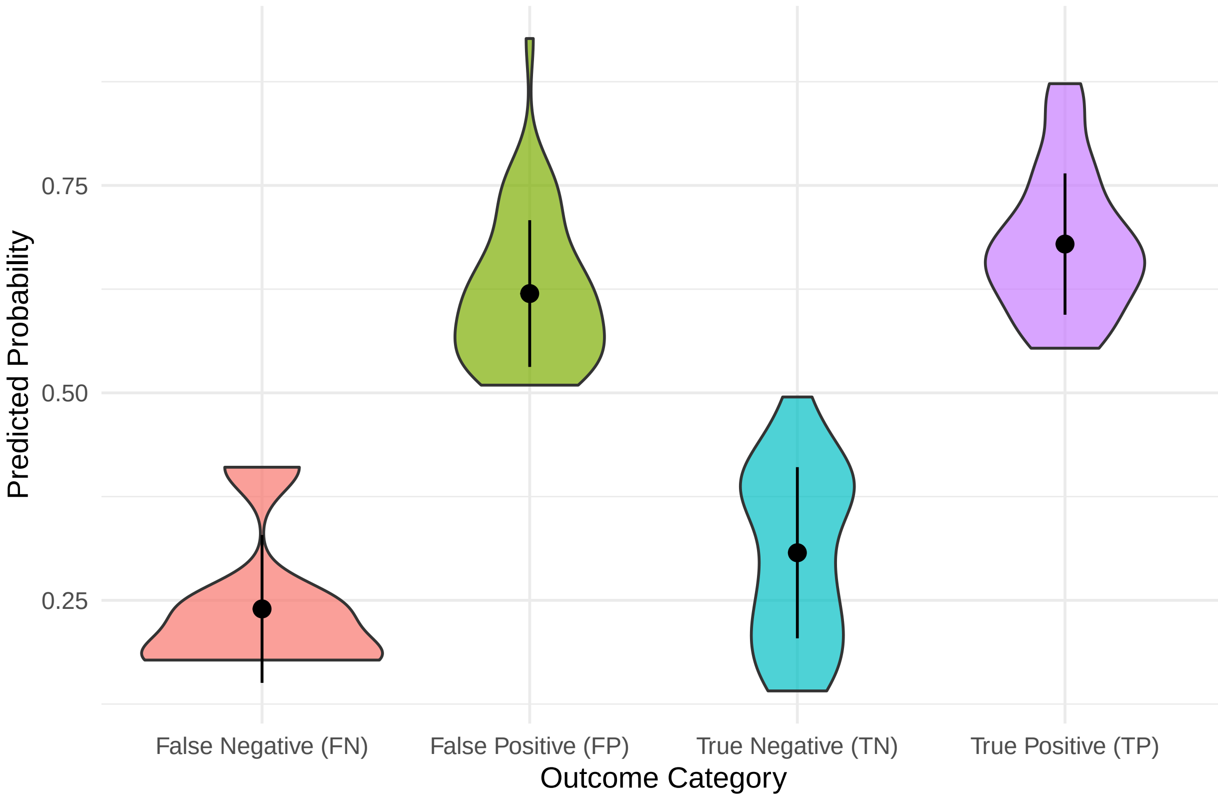
```r
evaluate_binary_model_performance(final_binary_model, over, test_data, "hospdead")
```

```
## Sensitivity: 0.8378378
## Sensitivity CI: 0.6798629 0.9380743
```

```
## Specificity: 0.5901639
## Specificity CI: 0.4974962 0.6783489
```




```r
# Predict probabilities on the test dataset
test_pred_probs <- predict(final_binary_model, newdata = test_data, type = "response")

# Categorize observations based on predicted probabilities and actual outcomes
test_data <- test_data %>%
  mutate(
    predicted = ifelse(test_pred_probs > 0.5, 1, 0),
    category = case_when(
      hospdead == 1 & predicted == 1 ~ "True Positive (TP)",
      hospdead == 0 & predicted == 0 ~ "True Negative (TN)",
      hospdead == 0 & predicted == 1 ~ "False Positive (FP)",
      hospdead == 1 & predicted == 0 ~ "False Negative (FN)"
    )
  )

# Plot the distribution of predicted probabilities for each category
ggplot(test_data, aes(x = category, y = test_pred_probs, fill = category)) +
  geom_violin(trim = TRUE, alpha = 0.7) +
  stat_summary(fun.data = "mean_sdl", fun.args = list(mult = 1), geom = "pointrange", color = "black") +
  labs(
    title = "",
    x = "Outcome Category",
    y = "Predicted Probability"
  ) +
  theme_minimal() +
  theme(legend.position = "none")
```




```r
binary_model_with_slos <- glm(update(new_formula, . ~ . + slos), data = over, family = binomial)
summary(binary_model_with_slos)
```

```
##
## Call:
## glm(formula = update(new_formula, . ~ . + slos), family = binomial,
##     data = over)
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -1.419940   0.469194  -3.026 0.002475 **
## sex2          -0.036205   0.142086  -0.255 0.798869
## edu            0.055777   0.020445   2.728 0.006369 **
## age            0.016160   0.005387   3.000 0.002701 **
## num_co         0.200536   0.054844   3.656 0.000256 ***
## dzclass2       1.158597   0.691887   1.675 0.094023 .
## dzclass3       4.738132   1.220532   3.882 0.000104 ***
## dzclass4       0.287419   1.039751   0.276 0.782218
## slos          -0.004883   0.003439  -1.420 0.155714
## age:dzclass2 -0.044755   0.010937  -4.092 4.27e-05 ***
## age:dzclass3 -0.061622   0.017961  -3.431 0.000601 ***
## age:dzclass4 -0.021650   0.016726  -1.294 0.195523
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1361.3  on 981  degrees of freedom
## Residual deviance: 1196.6  on 970  degrees of freedom
```

```
## AIC: 1220.6
##
## Number of Fisher Scoring iterations: 4
```

```r
evaluate_binary_model_performance(binary_model_with_slos, train_data, test_data, "hospdead")
```

```
## Sensitivity: 0.8378378
## Sensitivity CI: 0.6798629 0.9380743
## Specificity: 0.5983607
## Specificity CI: 0.5057622 0.6860679
```