Statistical Learning Assessed Practical 2

Julian Bara-Mason | sID: 201674483

2023-04-28

Task 1

Consider a logistic regression model for the data. Use the model with all inputs to find which input variables are relevant for explaining the output by interpreting the model. Identify the direction and magnitude of each input effect from the fitted coefficients. Which inputs would you say have strong effects? Order the inputs in terms of decreasing effect. Justify your reasoning. Compare your findings with the plots shown on the Guardian website. Do your findings agree with these plots? Comment on your findings.

The model is defined as follows:

$$voteBrexit = \beta_0 + \beta_1(abc1) + \beta_2(medianIncome) + \beta_3(medianAge) + \beta_4(withHigherEd) + \beta_5(notBornUK) + \epsilon, \epsilon \sim N(0, \sigma^2),$$

where $voteBrexit$ is the output, and $\beta_1(abc1)$, $\beta_2(medianIncome)$, $\beta_3(medianAge)$, $\beta_4(withHigherEd)$ & $\beta_5(notBornUK)$ are the inputs.

```
brexit <- read.csv("brexit.csv")

# Converts the response variable to a binary numeric variable, where 1 represents "yes" and 0 represents
"no"
brexit$voteBrexit <- as.numeric(brexit$voteBrexit == "TRUE")

set.seed(123) # for reproducibility

# Randomly selects 80% of the data for training and 20% for testing
train_index <- sample(nrow(brexit), round(0.8*nrow(brexit)), replace=FALSE)
train_data <- brexit[train_index, ]
test_data <- brexit[-train_index, ]

# Fits logistic regression model using the training data
model <- glm(voteBrexit ~ abc1 + medianIncome + medianAge + withHigherEd + notBornUK, data=train_data, fa
mily=binomial)

# Calculates the z-score for a 95% confidence interval
zc <- qnorm(0.975)


# Calculates the estimates, standard errors, and confidence intervals for each input variable
abc1_estimate <- summary(model)$coefficients[2, 1]
abc1_ste <- summary(model)$coefficients[2,2]
abc1_CI <- c((abc1_estimate - zc*abc1_ste), (abc1_estimate + zc*abc1_ste))

medianIncome_estimate <- summary(model)$coefficients[3, 1]
medianIncome_ste <- summary(model)$coefficients[3, 2]
medianIncome_CI <- c((medianIncome_estimate - zc*medianIncome_ste), (medianIncome_estimate + zc*medianInc
ome_ste))

medianAge_estimate <- summary(model)$coefficients[4, 1]
medianAge_ste <- summary(model)$coefficients[4, 2]
medianAge_CI <- c((medianAge_estimate - zc*medianAge_ste), (medianAge_estimate + zc*medianAge_ste))

withHigherEd_estimate <- summary(model)$coefficients[5, 1]
withHigherEd_ste <- summary(model)$coefficients[5, 2]
withHigherEd_CI <- c((withHigherEd_estimate - zc*withHigherEd_ste), (withHigherEd_estimate + zc*withHighe
rEd_ste))

notBornUK_estimate <- summary(model)$coefficients[6, 1]
notBornUK_ste <- summary(model)$coefficients[6, 2]
notBornUK_CI <- c((notBornUK_estimate - zc*notBornUK_ste), (notBornUK_estimate + zc*notBornUK_ste))

summary(model)$coefficients
```

```
##                Estimate Std. Error    z value     Pr(>|z|)
## (Intercept)   -0.5456499  0.9433889 -0.5783934 5.629986e-01
## abc1          16.4429287  3.1406356  5.2355417 1.645016e-07
## medianIncome  -5.2369703  2.0677492 -2.5326912 1.131906e-02
## medianAge      6.3646618  1.5335454  4.1502925 3.320507e-05
## withHigherEd -25.7200270  3.8320858 -6.7117565 1.922953e-11
## notBornUK      5.6805093  1.9377381  2.9315156 3.373125e-03
```

From the coefficients, we can see the four input variables have statistically significant effects on the likelihood of voting for Brexit with p-values all less than 5%.

Input variables with negative coefficients suggest that per unit increase in the variable, the probability of the output variable (voteBrexit) decreases by the input's coefficient.

Conversely, the probability of voting for Brexit increases by the input's coefficient if the input variable has a negative coefficient.

Based on the magnitude of the coefficients, the inputs with the strongest to lowest effects on an electoral ward's brexit vote are as follows:

1. withHigherEd has a negative coefficient of **25.72** with a confidence interval of [-33.23, -18.21]
2. abc1 has a positive coefficient of **16.44** with a confidence interval of [10.29, 22.60]
3. medianAge has a positive coefficient of **6.36** with a confidence interval of [3.36, 9.37]
4. notBornUK has a positive coefficient of **5.68** with a confidence interval of [1.88, 9.48]
5. medianIncome has a negative coefficient of **5.24** with a confidence interval of [-9.29, -1.18]

The input variables with the strongest effects are "withHigherEd" and "abc1", suggesting that a higher proportion of residents with higher education and a higher proportion of residents in the abc1 social classes have a lower probability of a ward voting for Brexit.

"medianAge" has a positive coefficient of 6.36, indicating that a higher median age is likely to have a higher probability of a ward voting for Brexit. "notBornUK"s positive coefficient of 5.68 suggets that a higher proportion of residents who were born outside the UK is likely to have a higher probability of a ward voting for Brexit.

"medianIncome" has the smallest but negative magnitude of 5.24, indicating that a higher median income is likely to have a lower probability of a ward voting for Brexit.

These findings are consistent with the plots on the Guardian website. However, the plots don't provide any quantitative measures of the effects, so we can't compare the magnitudes of the effects directly. Therefore, we'll lead with the assumption that the arrangement of the plots indicate the order of magnitudes from strongest to weakest: withHigherEd > abc1 > noFormalQualifications > medianAge > medianIncome > notBornUK.

The plots suggest that a higher proportion of residents with higher level education and a higher median income are likely to have a lower probability of a ward voting for Brexit. The direction of the predictors on the plots are consistent with the negative coefficients for "withHigherEd" and "medianIncome" in the logistic regression model. Similarly, the positive coefficients of the model are consistent with the direction on the plots.

Task 2

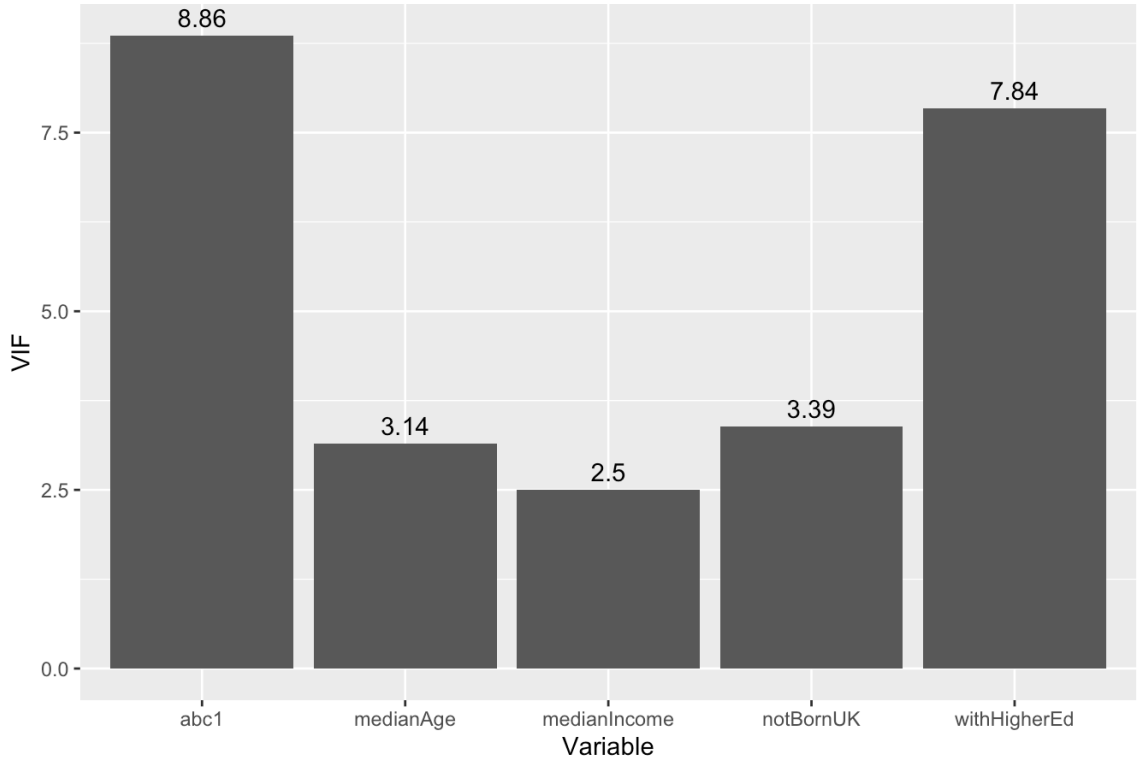
Discuss factors that may affect interpretability of the regression coefficients of the fitted model. Based on your discussion, explain whether you can reliably determine which inputs are relevant for modelling the output and order the input variables based on their relevance in decreasing order. Justify your reasoning.

The interpretability of the regression coefficients may be affected by several factors including:

1. Collinearity: When two or more input variables are highly correlated with each other, it can affect the interpretability of the model by making it difficult to determine the individual effects of each variable on the output. By using a Variance Inflation Factor (VIF) diagnostic test, we can check for collinearity. A VIF value between 1-5 suggests low correlation not enough to warrant corrective measures; VIF less than 10 but greater than 5 indicates moderate correlation; VIF above 10 suggests critical levels of collinearity: the coefficients are poorly estimated with questionable p-values. Fig. 1 below is the VIF values of the variables in the logistic regression model.

```
library(car)
library(ggplot2)
vif_values <- vif(model)

# Dataframe of VIF values
vif_df <- data.frame(variables = names(vif_values), vif = vif_values)

# Bar plot of VIF values
ggplot(data = vif_df, aes(x = variables, y = vif)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = round(vif, 2)), vjust = -0.5) +
  labs(x = "Variable", y = "VIF") +
  ggtitle("Fig. 1 | VIF values to check for collinearity")
```

Fig. 1 | VIF values to check for collinearity

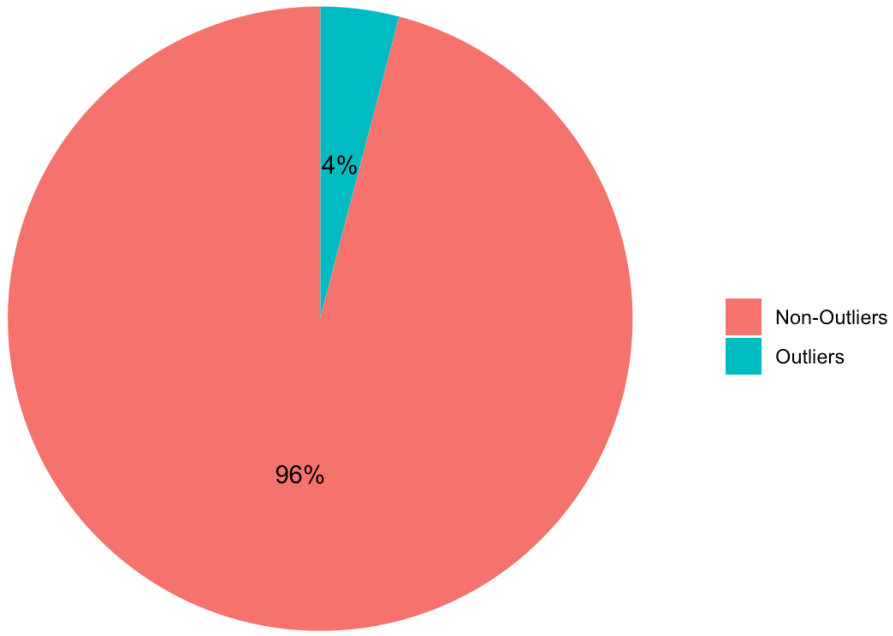


The VIF values of abc1 (9.99) and withHigherEd (8.56) are moderate levels of collinearity, indicating some collinearity might exist in the data. The other variables have low VIF values, indicating no collinearity.

2. Outliers: Datapoints that are significantly different from most datapoints in the dataset can distort the results of the model. Outliers can be detected using Mahalanobis Distance - a measurement of each observation from the center of the data. Observations with large Mahalanobis distances can be considered outliers. Fig. 2 is the proportion of outliers and non-outliers in the data.

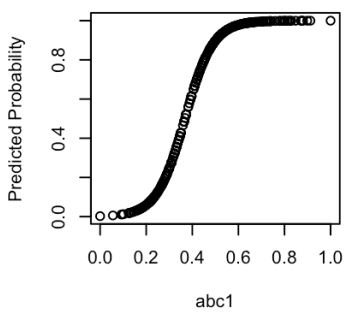
```
# Mahalanobis distance for each observation
mah_dist <- mahalanobis(model$model, center = colMeans(model$model), cov = vcov(model))

# Observations with high Mahalanobis distance
outliers <- mah_dist > quantile(mah_dist, probs = 0.95)

# Proportion of outliers
prop_outliers <- sum(outliers) / nrow(brexit)

# Proportion of non-outliers
prop_non_outliers <- 1 - prop_outliers

# Dataframe for pie chart
df <- data.frame(
  proportion = c(prop_outliers, prop_non_outliers),
  category = c("Outliers", "Non-Outliers")
)

# Pie chart
ggplot(df, aes(x = "", y = proportion, fill = category)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar("y", start=0) +
  ggtitle("Fig. 2 | Proportion of Outliers using Mahalanobis Distance") +
  geom_text(aes(label = paste0(round(proportion * 100), "%")),
            position = position_stack(vjust = 0.5)) +
  labs(fill = "") +
  theme_void()
```

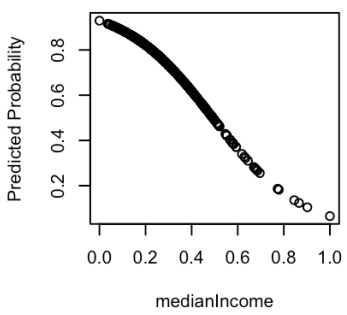
Fig. 2 | Proportion of Outliers using Mahalanobis Distance

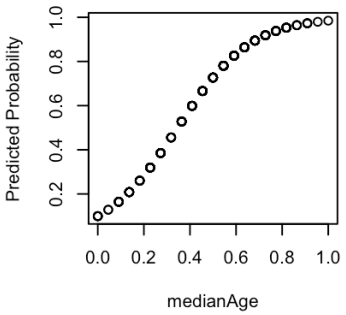


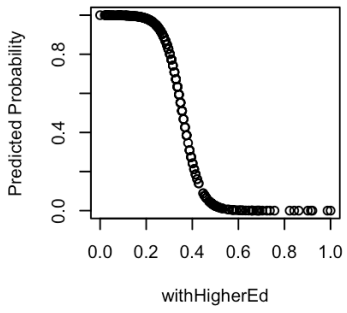
3. Non-monotonic predictors: Monotonic predictors are the input variables whose relationship with the output variable is always in the same direction. They improve the interpretability of the regression model and help avoid high collinearity. Non-monotonic predictors may lead to overfitting or underfitting the model. All input variables are monotonic, as the 5 charts below show.

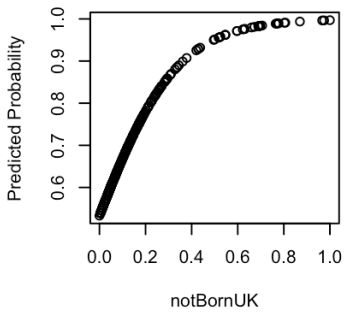
```
plot_predictor <- function(var_name) {
  # Dataframe with the predictors and corresponding probabilities
  new_data <- data.frame(
    abc1 = rep(mean(brexit$abc1), nrow(brexit)),
    medianIncome = rep(mean(brexit$medianIncome), nrow(brexit)),
    medianAge = rep(mean(brexit$medianAge), nrow(brexit)),
    withHigherEd = rep(mean(brexit$withHigherEd)),
    notBornUK = rep(mean(brexit$notBornUK), nrow(brexit))
  )

  new_data[[var_name]] <- brexit[[var_name]]
  new_data$prob <- predict(model, newdata = new_data, type = "response")

  plot(new_data[[var_name]], new_data$prob, xlab = var_name, ylab = "Predicted Probability")
}

# Calls the plot_predictor function for each input variable
par(mfrow = c(2, 3))
plot_predictor("abc1")
plot_predictor("medianIncome")
plot_predictor("medianAge")
plot_predictor("withHigherEd")
plot_predictor("notBornUK")
```











Task 3

Based on your discussion for Task 2, present and carry out an alternative approach to carry out the analysis for Task 1. Discuss benefits and disadvantages of your approach.

From Task 2, we saw that the logistic regression model was fairly interpretable but had some issues with moderate collinearity and outliers. On the other hand, all predictors are monotonic, so a logistic regression model was applicable.

To tackle the moderate collinearity, a simple decision tree was used as the alternative approach. A Random Forest algorithm is another possible alternative but the data we're working with does not appear complex enough for a full-fledged Random Forest.

A Decision Tree is beneficial in this case because it is not affected by collinearity as it's based on a hierarchical split of the input variables. The decision tree is also not sensitive to outliers because outliers don't cause much reduction in the Residual Sum of Squares as thy are never involved in the split.

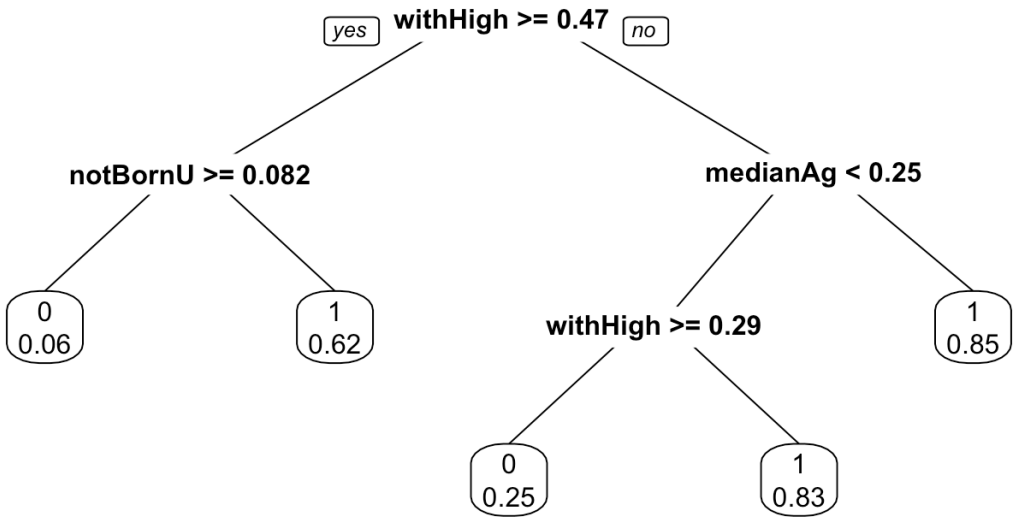
However, decision trees are prone to overfitting if the data has too many variables. In the context of this analysis, we want to know the magnitude of each predictor variable and their direction, in order to rank the variables. Decision trees are not of great help here. Although they provide a Variable Importance metrics, these metrics are considered unreliable and not a great way of statistically inferring the important predictor variables.

As a result, we will use other model evaluation techniques to decide on the better model between the Logistic Regression and the Decision Tree.

Below is the Decision Tree we developed, with withHighEd being the starting node.

```
library(rpart.plot)
# Fits decision tree model
tree_model <- rpart(voteBrexit ~ abc1 + medianIncome + medianAge + withHigherEd + notBornUK, data=train_d
ata, method="class")

# Plots decision tree model
prp(tree_model, extra=6, main="Fig. 8 | Decision Tree")
```

Fig. 8 | Decision Tree



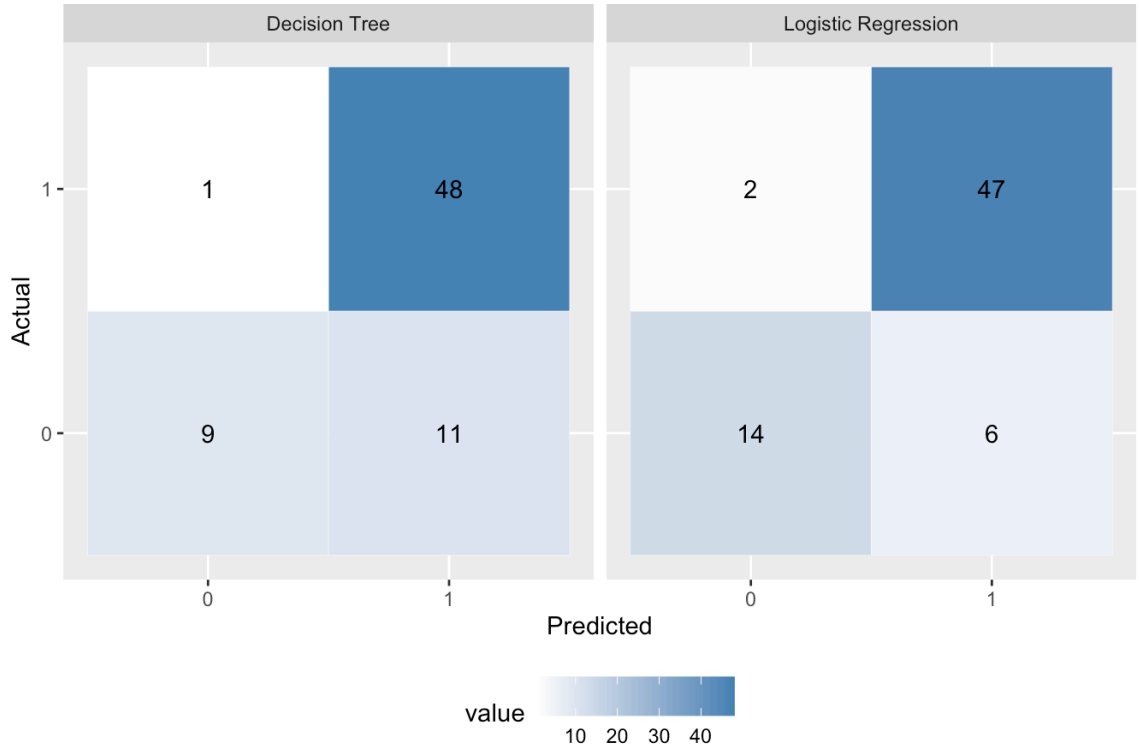
```
#install.packages("pROC")
library("pROC")
library(reshape2)

log_pred <- predict(model, newdata= test_data, type="response")
log_pred_binary <- ifelse(log_pred > 0.5, 1, 0)

tree_pred <- predict(tree_model, newdata=test_data, type="class")


# Prediction Comparison
# ROC AUC
log_auc <- (roc(test_data$voteBrexit, log_pred))
tree_auc <- roc(test_data$voteBrexit, as.numeric(tree_pred))

# Model Accuracy
log_accuracy <- round(sum(log_pred_binary == test_data$voteBrexit)/length(test_data$voteBrexit), 2)
tree_accuracy <- round(sum(tree_pred == test_data$voteBrexit)/length(test_data$voteBrexit), 2)

# Dataframe for accuracy
accuracy_df <- data.frame(model = c("Logistic Regression", "Decision Tree"),
                          accuracy = c(log_accuracy, tree_accuracy))


# Confusion Matrices
log_confusion <- table(log_pred_binary, test_data$voteBrexit)
tree_confusion <- table(tree_pred, test_data$voteBrexit)

# Converts confusion matrices to data frames and reshape to long format
log_df <- melt(as.data.frame(log_confusion))
tree_df <- melt(as.data.frame(tree_confusion))

colnames(log_df) <- c("Var1", "Var2", "variable", "value")
colnames(tree_df) <- c("Var1", "Var2", "variable", "value")

# Adds column with model name
log_df$model <- "Logistic Regression"
tree_df$model <- "Decision Tree"

# Combines dataframes
combined_df <- rbind(log_df, tree_df)
```

Model Evaluation

1. Confusion Matrix: A table of the true positives, false positives, false negatives, and true negatives of a prediction. Fig. 9 below is the heatmap of the Decision Tree & Logistic Regression models. The diagonal elements are the correct predictions, while the off-diagonal elements are the incorrect predictions. In the output below, the tree prediction (tree_pred) has more incorrect predictions (12) than the logistic regression prediction (8). This leads us to model accuracy

```
# Heatmap
ggplot(combined_df, aes(x = Var1, y = Var2, fill = value)) +
  geom_tile(color = "white") +
  geom_text(aes(label = value), color = "black") +
  facet_wrap(~ model, nrow = 1) +
  scale_fill_gradient(low = "white", high = "steelblue") +
  labs(title = "Fig. 9 | Confusion Matrices", x = "Predicted", y = "Actual") +
  theme(legend.position = "bottom")
```

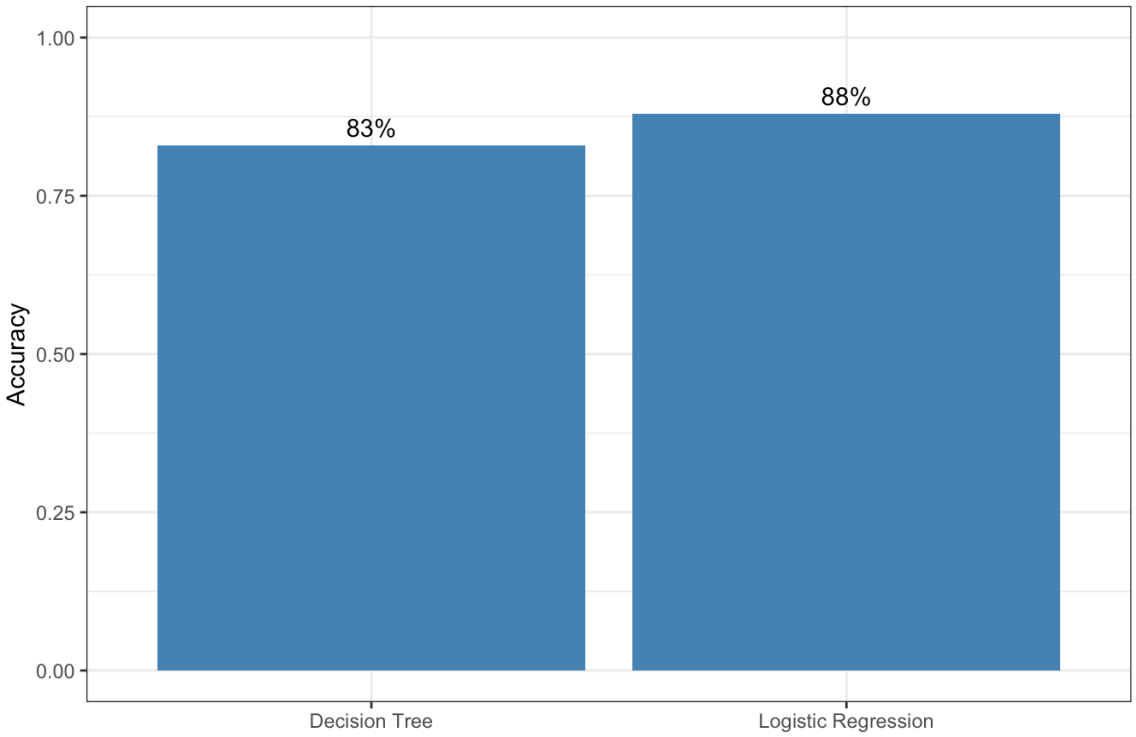
Fig. 9 | Confusion Matrices



2. Model Accuracy: This is the proportion of correct predictions against the total number of predictions. The log regression (88%) is more accurate than the decision tree (83%) as seen in Fig. 10 below.

```
# Model accuracy Bar chart
ggplot(accuracy_df, aes(x = model, y = accuracy)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  geom_text(aes(label = paste0(accuracy*100, "%")), vjust = -0.5) +
  ylim(0, 1) +
  labs(title = "Fig. 10 | Model Accuracy", y = "Accuracy", x = "") +
  theme_bw()
```

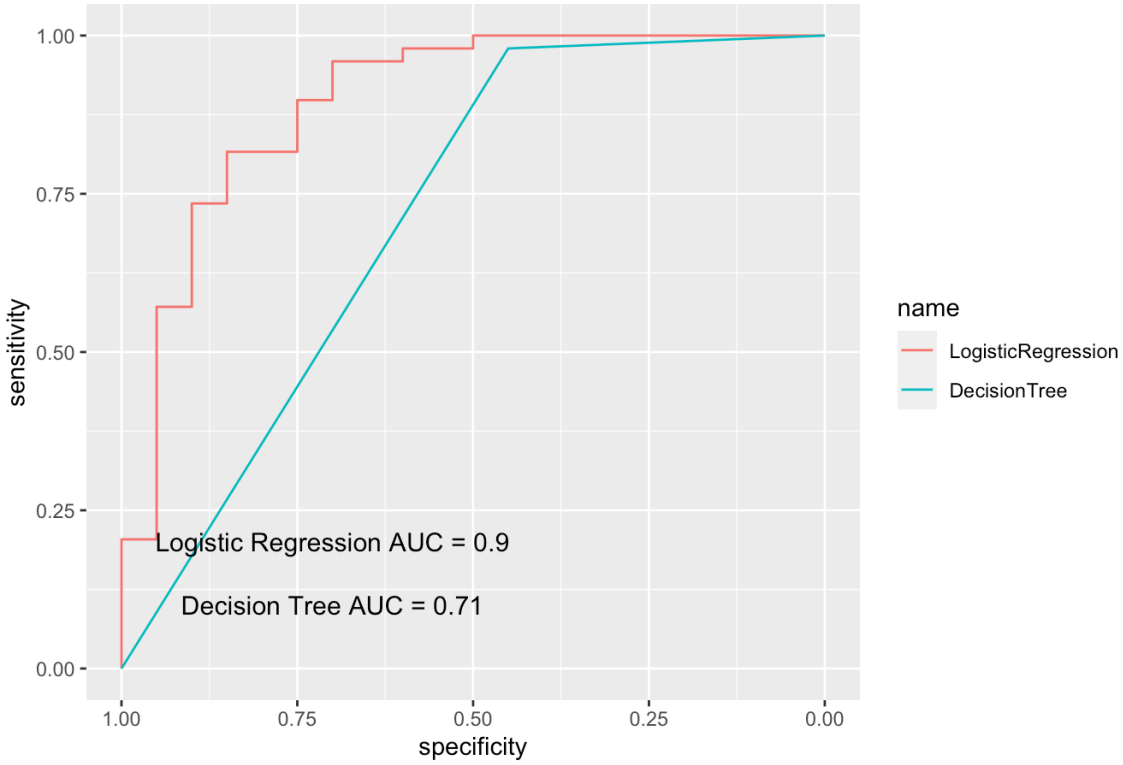
Fig. 10 | Model Accuracy



3. Receiver Operating Characteristic Area Under the Curve (ROC AUC): tells us how good a model is capable of distinguishing classes. The higher the AUC value, the better the model. The logistic regression has a higher AUC value of 0.9 compared to the Decision Tree's 0.71 as seen in Fig. 11 below.

```
#  ROC curve plot
ggroc(list(LogisticRegression = log_auc, DecisionTree = tree_auc)) +
  annotate("text", x = 0.7, y = 0.2, label = paste0("Logistic Regression AUC = ", round(log_auc$auc, 2)),
size = 4, color = "black") +
  annotate("text", x = 0.7, y = 0.1, label = paste0("Decision Tree AUC = ", round(tree_auc$auc, 2)), size
= 4, color = "black") +
  labs(title = "Fig. 11 | ROC Curves and AUC Values")
```

Fig. 11 | ROC Curves and AUC Values



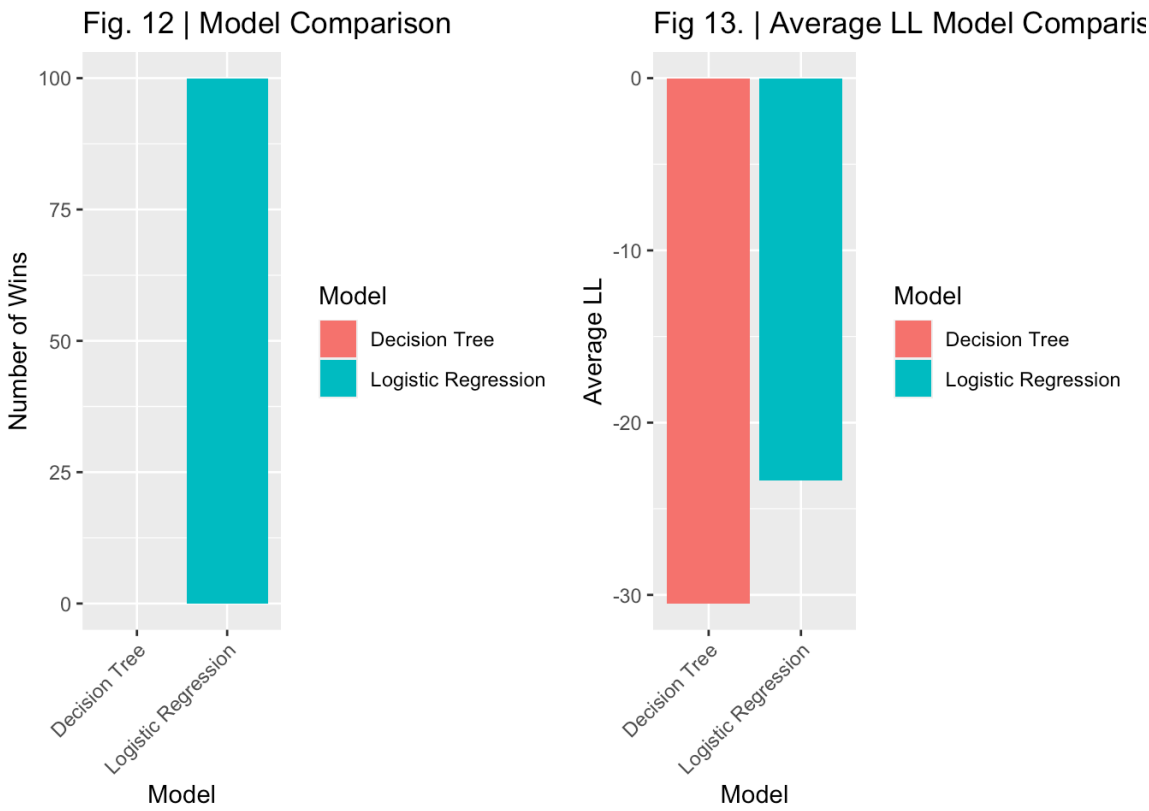
These three metrics provide a fairly comprehensive evaluation of both model's performance and so far, the logistic regression model is the better model. A final test is Cross Validation.

Cross-Validation

We cross-validated both models 100 times to find out which model performed better on each try. Fig. 11 below shows the logistic regression model performed better every single time and Fig. 12 shows the logistic regression model had the higher log-likelihood on average across all 100 times.

```
log_wins <- 0
tree_wins <- 0
log_lls <- numeric(100)
tree_lls <- numeric(100)

# Loop 100 times for cross-validation
for (i in 1:100){
  # Sample training data with replacement
  train_sample_idx <- sample(nrow(brexit), round(0.8*nrow(brexit)), replace=FALSE)
  train_sample_data <- brexit[train_index, ]
  test_sample_data <- brexit[-train_index, ]

  # Fits logistic regression model
  log_model <- glm(voteBrexit ~ abc1 + medianIncome + medianAge + withHigherEd + notBornUK, data=train_sa
mple_data, family=binomial)

  # Fits tree model
  tree_model2 <- rpart(voteBrexit ~ abc1 + medianIncome + medianAge + withHigherEd + notBornUK, data=trai
n_sample_data, method="class")


  # Makes predictions on test data
  log_preds <- predict(log_model, newdata = test_sample_data, type = "response")
  log_preds_binary <- as.numeric(log_preds > 0.5)

  tree_probs <- predict(tree_model2, newdata = test_sample_data, type = "prob")
  tree_ll <- sum(log(tree_probs[cbind(1:nrow(test_sample_data), match(test_sample_data$voteBrexit, colnam
es(tree_probs)))]))

  # Compares log-likelihood of each model
  log_ll <- sum(log(dbinom(test_sample_data$voteBrexit, size = 1, prob = log_preds)))

  # Updates winner count
  if (log_ll > tree_ll) {
    log_wins <- log_wins + 1
  } else {
    tree_wins <- tree_wins + 1
  }

  log_lls[i] <- log_ll
  tree_lls[i] <- tree_ll
}
# Wins dataframe
wins_df <- data.frame(Model = c("Logistic Regression", "Decision Tree"),
                      Wins = c(log_wins, tree_wins))

# Average LL dataframe
avg_ll_df <- data.frame(Model = c("Logistic Regression", "Decision Tree"),
                      Wins = c(mean(log_lls), mean(tree_lls)))
```

```
# Bar charts for winning model comparison
library(gridExtra)

p1 <- ggplot(wins_df, aes(x = Model, y = Wins, fill = Model)) +
  geom_bar(stat = "identity") +
  labs(title = "Fig. 12 | Model Comparison", x = "Model", y = "Number of Wins") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

p2 <- ggplot(avg_ll_df, aes(x = Model, y = Wins, fill = Model)) +
  geom_bar(stat = "identity") +
  labs(title = "Fig 13. | Average LL Model Comparison", x = "Model", y = "Average LL") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

grid.arrange(p1, p2, ncol=2)
```



The different model evaluation methods used show that the logistic regression model is the better model. It is also the better model for this analysis because we want to know how the input variables rank - the logistic regression model provides this, while the decision tree does not.