DSC1107_MONFERO_SA1

John Benedict A. Monfero

March 18, 2025

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.4     ✓ readr     2.1.5
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ ggplot2   3.5.1     ✓ tibble    3.2.1
## ✓ lubridate 1.9.4     ✓ tidyr     1.3.1
## ✓ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✕ dplyr::filter() masks stats::filter()
## ✕ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
##
## Attaching package: 'kableExtra'
##
##
## The following object is masked from 'package:dplyr':
##
##     group_rows
##
##
##
## Attaching package: 'cowplot'
##
##
## The following object is masked from 'package:lubridate':
##
##     stamp
##
##
##
## Attaching package: 'stat471'
##
##
## The following object is masked from 'package:FNN':
##
##     knn
```

```
## Warning: package 'caret' was built under R version 4.4.3
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
## Warning: package 'e1071' was built under R version 4.4.3
```

```
## Warning: package 'rpart.plot' was built under R version 4.4.3
```

```
## Warning: package 'MLmetrics' was built under R version 4.4.3
```

```
##
## Attaching package: 'MLmetrics'
##
## The following objects are masked from 'package:caret':
##
##     MAE, RMSE
##
## The following object is masked from 'package:base':
##
##     Recall
```

```
## Warning: package 'randomForest' was built under R version 4.4.3
```

```
## randomForest 4.7-1.2
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:dplyr':
##
##     combine
##
## The following object is masked from 'package:ggplot2':
##
##     margin
```

Objective:

> The purpose of this summative assessment is to evaluate students' ability to apply data mining techniques, data visualization, data wrangling, and predictive modeling using R. Students will work with a provided dataset to perform exploratory data analysis, data transformation, model tuning, and regression-based methods.

> **Dataset**: Download the provided dataset `customer_churn.csv` , which contains customer demographics, service usage data, and churn labels.

```
data <- read.csv("CUSTOMER_CHURN_TIDY.CSV")

glimpse(data)
```

```
## Rows: 9,928
## Columns: 12
## $ CustomerID      <chr> "CUST00001", "CUST00002", "CUST00003", "CUST00004", "C…
## $ Gender          <chr> "Male", "Male", "Male", "Female", "Male", "Female", "F…
## $ SeniorCitizen   <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, …
## $ Partner         <chr> "No", "No", "Yes", "Yes", "No", "No", "Yes", "Yes", "Y…
## $ Dependents      <chr> "No", "No", "No", "Yes", "No", "Yes", "No", "Yes", "Ye…
## $ Tenure          <int> 65, 26, 54, 70, 53, 45, 35, 20, 48, 33, 33, 39, 6, 51,…
## $ PhoneService    <chr> "Yes", "Yes", "Yes", "Yes", "Yes", "Yes", "Yes", "Yes"…
## $ InternetService <chr> "Fiber optic", "Fiber optic", "Fiber optic", "DSL", "D…
## $ Contract        <chr> "Month-to-month", "Month-to-month", "Month-to-month", …
## $ MonthlyCharges  <dbl> 20.04, 65.14, 49.38, 31.19, 103.86, 87.34, 119.91, 69.…
## $ TotalCharges    <dbl> 1302.60, 1693.64, 2666.52, 2183.30, 5504.58, 3930.30, …
## $ Churn           <chr> "No", "No", "No", "No", "Yes", "Yes", "Yes", "Yes", "N…
```

Unit 2: Tuning Predictive Models

Model Complexity

> Fit a decision tree and logistic regression model.

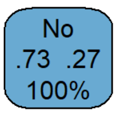
```
# Desicion Tree
  # Revise the data type of the `Churn` variable
data$Churn <- factor(data$Churn, levels = c("No", "Yes"))

  # Partitioning the data into 70% Training and  30% Test
train_index <- createDataPartition(data$Churn, p = 0.7, list = FALSE)
train_data <- data[train_index, ]
test_data <- data[-train_index, ]

  # Address missing TotalCharges in training data
train_data$TotalCharges[is.na(train_data$TotalCharges)] <- median(train_data$TotalCharges, na.rm = TRUE)

  # Fit the decision tree model based on your partitioned training data
decision_tree <- rpart(
  Churn ~ Tenure + MonthlyCharges + TotalCharges + InternetService + Contract,
  data = train_data,
  method = "class",
  control = rpart.control(minsplit = 5, cp = 0.01))

  # Visualize the decision tree
rpart.plot(decision_tree, type = 2, extra = 104, fallen.leaves = TRUE)
```




```
cat("Having a decision tree yeilds a single node result predicting `Churn = \"No\"` with Gini's Impurity of 0.73 True
Positives and 0.27 False Positive Results, this overcomplicated intepretation suggests that the tree could not determi
ne the best split fits in order to have selected predictors even if four (4) variables already considered: {`Tenure +
MonthlyCharges + TotalCharges + InternetService + Contract`}. This induces the following:", "\n\n\n")
```

```
## Having a decision tree yeilds a single node result predicting `Churn = "No"` with Gini's Impurity of 0.73 True Posi
tives and 0.27 False Positive Results, this overcomplicated intepretation suggests that the tree could not determine t
he best split fits in order to have selected predictors even if four (4) variables already considered: {`Tenure + Mont
hlyCharges + TotalCharges + InternetService + Contract`}. This induces the following:
```

```
cat("The variables produces Homogenous Data: lack of distinctions or clear seperation (as provided in Unit 1: Data Vis
ualization) having no outstanding basis as predictors")
```

```
## The variables produces Homogenous Data: lack of distinctions or clear seperation (as provided in Unit 1: Data Visua
lization) having no outstanding basis as predictors
```

```
cat("Class Imbalances: Most of the customers (as provided in Unit 1: Data Visualization) are in favor `Churn = \"No
\"`. The factor is highly skewwed, the overfitting of the desicion tree is highly observable in this model.")
```

```
## Class Imbalances: Most of the customers (as provided in Unit 1: Data Visualization) are in favor `Churn = "No"`. Th
e factor is highly skewwed, the overfitting of the desicion tree is highly observable in this model.
```

```
# Fit logistic regression
logistic_model <- glm(Churn ~ Tenure + MonthlyCharges + TotalCharges + InternetService + Contract,
                      data = train_data,
                      family = binomial)

# Summarize the model
summary(logistic_model)
```

```
##
## Call:
## glm(formula = Churn ~ Tenure + MonthlyCharges + TotalCharges +
##     InternetService + Contract, family = binomial, data = train_data)
##
## Coefficients:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -1.083e+00  1.484e-01  -7.299  2.9e-13 ***
## Tenure                     3.791e-03  3.452e-03   1.098   0.2721
## MonthlyCharges             1.679e-03  1.872e-03   0.897   0.3698
## TotalCharges              -3.887e-05  4.610e-05  -0.843   0.3992
## InternetServiceFiber optic -2.614e-02  5.988e-02  -0.436   0.6625
## InternetServiceNo         -1.622e-01  7.565e-02  -2.145   0.0320 *
## ContractOne year          -1.266e-01  7.120e-02  -1.778   0.0753 .
## ContractTwo year           2.098e-02  6.858e-02   0.306   0.7596
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8122.2  on 6949  degrees of freedom
## Residual deviance: 8112.0  on 6942  degrees of freedom
## AIC: 8128
##
## Number of Fisher Scoring iterations: 4
```

```r
# Visualize Logistic Regression Lines:

# Tenure vs Churn
plot1 <- ggplot(train_data, aes(x = Tenure, y = as.numeric(Churn))) +
  geom_point(alpha = 0.5) +
  stat_smooth(method = "glm", method.args = list(family = "binomial"), se = FALSE) +
  labs(title = "Tenure vs Churn") +
  theme_minimal()

# Monthly Charges vs Churn
plot2 <- ggplot(train_data, aes(x = MonthlyCharges, y = as.numeric(Churn))) +
  geom_point(alpha = 0.5) +
  stat_smooth(method = "glm", method.args = list(family = "binomial"), se = FALSE) +
  labs(title = "Monthly Charges vs Churn") +
  theme_minimal()

# Total Charges vs Churn
plot3 <- ggplot(train_data, aes(x = TotalCharges, y = as.numeric(Churn))) +
  geom_point(alpha = 0.5) +
  stat_smooth(method = "glm", method.args = list(family = "binomial"), se = FALSE) +
  labs(title = "Total Charges vs Churn") +
  theme_minimal()

# Internet Service vs Churn
plot4 <- ggplot(train_data, aes(x = InternetService, y = as.numeric(Churn))) +
  geom_jitter(width = 0.2, alpha = 0.5) +
  stat_smooth(method = "glm", method.args = list(family = "binomial"), se = FALSE) +
  labs(title = "Internet Service vs Churn") +
  theme_minimal()

plot_grid(plot1, plot2, plot3, plot4, ncol = 2, nrow = 2)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Failed to fit group -1.
## Caused by error:
## ! y values must be 0 <= y <= 1
```

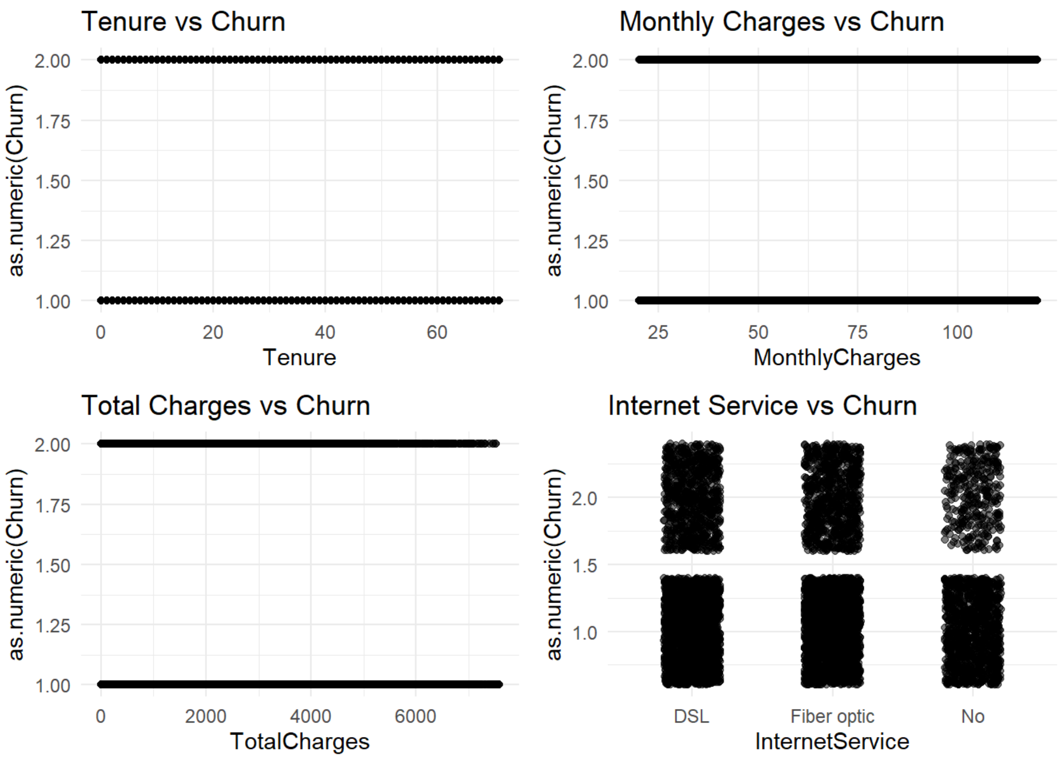
```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Failed to fit group -1.
## Caused by error:
## ! y values must be 0 <= y <= 1
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Failed to fit group -1.
## Caused by error:
## ! y values must be 0 <= y <= 1
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
cat("All independent variables failed to find best fit based on the distribution")
```

```
## All independent variables failed to find best fit based on the distribution
```

> Compare their complexities and explain trade-offs.

```
# Display the image
include_graphics("Quack.png")
```

Feature	**Decision Tree**	**Logistic Regression**
Complexity	Non-Parametric Model	Parametric Model
Interpretability	Easy to Interpret	Build Efforts to Interpret
Computational Time	Training is intensive for very deep trees	Training is less intensive since it solves an optimization problem
Handling of Non-Linearity	Non-Linear Relationships needed	Assumes the distributions follows linearity
Scalability	Struggles as the dataset is way larger or not optimized	Became resilent and scales well from scarce to large dataset
Risk	Overfitting	Underfitting

Bias-Variance Trade-Off

> Explain the concept of bias-variance trade-off in the context of the models trained.

```r
# Display the image
include_graphics("Quack2.png")
```

Prone to Bias	Logistic Regression	Since it assumes a linear relationship between predictors and the logarithmic odds of the response, it has to rely on higher bias, as the variable became more non-linear, the model underfits the data
Prone to Variance	Decision Tree	Small changes within the training data would make the tree unstable and can significantly change its structure in offer to comply to the variance of the sample, the model overfits the data

> Discuss how model complexity affects performance.

- `Logistic regression` is ideal for understanding the impact of individual predictors (like `ContractOne Year`) on `Churn` . It provides clear coefficients and a probabilistic interpretation.
- `Decision trees` are more suited for capturing interactions among predictors and providing interpretable decision paths. However, they need tuning to prevent overfitting, especially when working with categorical predictors like `Contract`

Cross-Validation

> Use k-fold cross-validation (k=10) to evaluate model performance.

```r
# Define the control for k-fold cross-validation
train_control <- trainControl(method = "cv", number = 10,
                              classProbs = TRUE,
                              summaryFunction = multiClassSummary)
```

```r
# Train logistic regression model with cross-validation
logistic_cv <- train(Churn ~ Tenure + MonthlyCharges + TotalCharges + InternetService + Contract,
                     data = train_data,
                     method = "glm",
                     family = "binomial",
                     trControl = train_control)
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo,
## : There were missing values in resampled performance measures.
```

```r
# View the results
logistic_cv
```

```
## Generalized Linear Model
##
## 6950 samples
##    5 predictor
##    2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 6255, 6255, 6255, 6254, 6255, 6255, ...
## Resampling results:
##
##   logLoss    AUC        prAUC      Accuracy   Kappa  F1         Sensitivity
##   0.5849385  0.5036039  0.5044266  0.7289211  0      0.8432091  1
##   Specificity  Pos_Pred_Value  Neg_Pred_Value  Precision  Recall
##   0            0.7289211       NaN             0.7289211  1
##   Detection_Rate  Balanced_Accuracy
##   0.7289211       0.5
```

```
# Train decision tree model with cross-validation
decision_tree_cv <- train(Churn ~ Tenure + MonthlyCharges + TotalCharges + InternetService + Contract,
                          data = train_data,
                          method = "rpart",
                          trControl = train_control)
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo,
## : There were missing values in resampled performance measures.
```

```
# View the results
decision_tree_cv
```

```
## CART
##
## 6950 samples
##    5 predictor
##    2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 6255, 6254, 6256, 6256, 6255, 6256, ...
## Resampling results across tuning parameters:
##
##   cp            logLoss    AUC        prAUC      Accuracy   Kappa
##   0.0003538570  0.8311488  0.4940050  0.4910062  0.6726653  -0.017343310
##   0.0003980892  0.7522388  0.4985165  0.4419796  0.6886402  -0.009925709
##   0.0004128332  0.7522388  0.4985165  0.4419796  0.6886402  -0.009925709
##   F1         Sensitivity  Specificity  Pos_Pred_Value  Neg_Pred_Value
##   0.7975059  0.8855053    0.10028707   0.7257415       0.2449281
##   0.8104473  0.9167010    0.07534898   0.7271674       0.2240309
##   0.8104473  0.9167010    0.07534898   0.7271674       0.2240309
##   Precision  Recall     Detection_Rate  Balanced_Accuracy
##   0.7257415  0.8855053  0.6454720       0.4928962
##   0.7271674  0.9167010  0.6682098       0.4960250
##   0.7271674  0.9167010  0.6682098       0.4960250
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.0004128332.
```

> Report and interpret accuracy, precision, recall, and F1-score.

```
include_graphics("Quack3.png")
```

10-fold cross-validation	Accuracy	Precision	Recall	F1
Logistic Regression	0.7289211	0.7289211	1	0.8432091
Desicion Tree	0.7274832	0.7291961	0.9960502	0.8419681

- Consider both models obtain very high (99.6%) and perfect (100%) `Recall` , it means it successfully identifies all churners while the other might have missed less than 1%. By definition, out of all actual positives, how many were correctly predicted (reduces false negatives).
- Both models observed comparable and similar `accuracy` and `precision` Suggesting either models are effective enough to the data set it deals.
- F1 is simply the `harmonic mean` between precision and recall thus, it shows how strong and compatible really each models mentioned.

Classification

> Train a Random Forest classifier to predict customer churn.

```
# Convert Churn variable to factor (if not already)
train_data$Churn <- factor(train_data$Churn)

# Handle missing values (example for TotalCharges)
train_data$TotalCharges[is.na(train_data$TotalCharges)] <- median(train_data$TotalCharges, na.rm = TRUE)
```

```
# Train the Random Forest model
random_forest_model <- randomForest(Churn ~ Tenure + MonthlyCharges + TotalCharges + InternetService + Contract,
                                    data = train_data,
                                    ntree = 500,       # Number of trees
                                    mtry = 3,          # Number of predictors sampled at each split
                                    importance = TRUE, # Enable feature importance
                                    proximity = TRUE)  # Enable proximity matrix
```

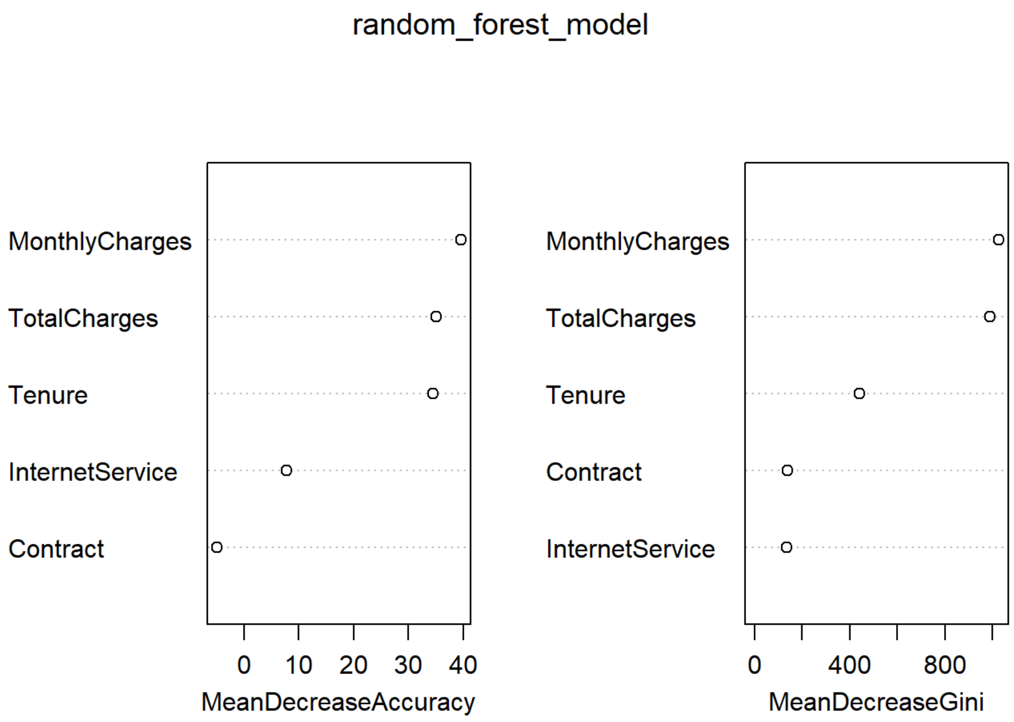
```
# Summary of the model
print(random_forest_model)
```

```
##
## Call:
##  randomForest(formula = Churn ~ Tenure + MonthlyCharges + TotalCharges +      InternetService + Contract, data = tr
ain_data, ntree = 500,      mtry = 3, importance = TRUE, proximity = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 33.19%
## Confusion matrix:
##       No Yes class.error
## No  4466 600   0.1184366
## Yes 1707 177   0.9060510
```

```
# Importance of predictors
importance(random_forest_model)
```

```
##                         No        Yes MeanDecreaseAccuracy MeanDecreaseGini
## Tenure          40.5927183 -43.966163            34.408608         440.8363
## MonthlyCharges  47.9941670 -44.267640            39.543477        1026.9583
## TotalCharges    40.8776465 -43.675620            35.009615         989.8966
## InternetService  9.9553072  -1.224511             7.851462         135.8814
## Contract        -0.8368726  -7.776821            -4.873507         136.1927
```

```
varImpPlot(random_forest_model)
```




```
# Generate predictions
rf_predictions <- predict(random_forest_model, test_data)

# Confusion matrix
confusion_matrix_rf <- confusionMatrix(rf_predictions, test_data$Churn)
print(confusion_matrix_rf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1920  713
##        Yes  251   94
##
##                Accuracy : 0.6763
##                  95% CI : (0.6592, 0.6931)
##     No Information Rate : 0.729
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.0011
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.8844
##             Specificity : 0.1165
##          Pos Pred Value : 0.7292
##          Neg Pred Value : 0.2725
##              Prevalence : 0.7290
##          Detection Rate : 0.6447
##    Detection Prevalence : 0.8842
##       Balanced Accuracy : 0.5004
##
##        'Positive' Class : No
##
```

```r
library(MLmetrics)

# Calculate performance metrics
Accuracy <- Accuracy(rf_predictions, test_data$Churn)
Precision <- Precision(rf_predictions, test_data$Churn, positive = "Yes")
Recall <- Recall(rf_predictions, test_data$Churn, positive = "Yes")
F1 <- F1_Score(rf_predictions, test_data$Churn, positive = "Yes")

# Print metrics
cat("Random Forest Metrics:\n")
```

```
## Random Forest Metrics:
```

```r
cat("Accuracy: ", Accuracy, "\n")
```

```
## Accuracy:  0.6762928
```

```r
cat("Precision: ", Precision, "\n")
```

```
## Precision:  0.1164808
```

```r
cat("Recall: ", Recall, "\n")
```

```
## Recall:  0.2724638
```

```r
cat("F1-Score: ", F1, "\n")
```

```
## F1-Score:  0.1631944
```

> Tune hyperparameters using grid search.

```r
# Correct tuning grid for Random Forest
tune_grid <- expand.grid(mtry = c(2, 3, 4))  # Number of predictors at each split
```

```r
train_control <- trainControl(method = "cv",
                              number = 5,
                              classProbs = TRUE,
                              summaryFunction = twoClassSummary)
```

```r
# Check for missing values
colSums(is.na(train_data))
```

```
##      CustomerID          Gender   SeniorCitizen         Partner      Dependents 
##               0               0               0               0               0 
##          Tenure    PhoneService InternetService        Contract  MonthlyCharges 
##               0               0               0               0               0 
##    TotalCharges           Churn 
##               0               0
```

```r
# Impute missing values for numeric variables
train_data$TotalCharges[is.na(train_data$TotalCharges)] <- median(train_data$TotalCharges, na.rm = TRUE)
```

```r
# Perform grid search with the corrected tuning grid
rf_model <- train(Churn ~ Tenure + MonthlyCharges + TotalCharges + InternetService + Contract,
                  data = train_data,
                  method = "rf",
                  trControl = train_control,
                  tuneGrid = tune_grid,
                  ntree = 500)  # Specify number of trees
```

```
## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" was not
## in the result set. ROC will be used instead.
```

> Report final model performance.

```r
# Confusion Matrix
confusion_matrix_rf <- confusionMatrix(rf_predictions, test_data$Churn)
print(confusion_matrix_rf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  1920  713
##        Yes  251   94
##
##                Accuracy : 0.6763
##                  95% CI : (0.6592, 0.6931)
##     No Information Rate : 0.729
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.0011
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.8844
##             Specificity : 0.1165
##          Pos Pred Value : 0.7292
##          Neg Pred Value : 0.2725
##              Prevalence : 0.7290
##          Detection Rate : 0.6447
##    Detection Prevalence : 0.8842
##       Balanced Accuracy : 0.5004
##
##        'Positive' Class : No
##
```

```r
# Accuracy
Accuracy <- Accuracy(rf_predictions, test_data$Churn)

# Precision
Precision <- Precision(rf_predictions, test_data$Churn, positive = "Yes")

# Recall
Recall <- Recall(rf_predictions, test_data$Churn, positive = "Yes")

# F1-Score
F1 <- F1_Score(rf_predictions, test_data$Churn, positive = "Yes")

# Print final metrics
cat("Final Random Forest Model Performance:\n")
```

```
## Final Random Forest Model Performance:
```

```r
cat("Accuracy: ", Accuracy, "\n")
```

```
## Accuracy:  0.6762928
```

```
cat("Precision: ", Precision, "\n")
```

```
## Precision:  0.1164808
```

```
cat("Recall: ", Recall, "\n")
```

```
## Recall:  0.2724638
```

```
cat("F1-Score: ", F1, "\n")
```

```
## F1-Score:  0.1631944
```