## **1. Task 2: Model Development & Evaluation**

### **1.1 Algorithm Selection**

Based on the binary classification nature of churn prediction and the need for interpretability, we recommend using **Random Forest** or **Gradient Boosting Machines**:

* **Random Forest**: Robust to overfitting, interpretable via feature importances, handles nonlinear relationships.
* **Gradient Boosting (e.g., XGBoost / LightGBM)**: Often achieves higher accuracy, can be tuned for class imbalance

**1.2 Model Training**

### # Xgboost Model

### set.seed(220)

### *# Split into training and testing sets*

### split <- sample.split(data$ChurnStatus, SplitRatio = 0.8)

### train <- subset(data, split == TRUE)

### test <- subset(data, split == FALSE)

*# Make column names safe*

names(train) <- make.names(names(train))

names(test) <- make.names(names(test))

train$ChurnStatus <- factor(ifelse(train$ChurnStatus == "1", "yes", "no"))

test$ChurnStatus <- factor(ifelse(test$ChurnStatus == "1", "yes", "no"))

*# Apply both over- and under-sampling*

both\_sampled <- ovun.sample(ChurnStatus ~ ., data = train, method = "under")$data

control <- trainControl(method="cv", number=10, classProbs = TRUE, summaryFunction = twoClassSummary)

*# Prediction with no warning printed in console*

xgb\_model <- train(ChurnStatus~., data=both\_sampled, method=”xgbTree”, metric=”ROC”, trControl=control, verbose = 0)

xgb\_model\_pred <- predict(xgb\_model, test, type = “prob”)

xgb\_model\_pred\_class <- as.factor(ifelse(xgb\_model\_pred[, “yes”] > 0.51, “yes”, “no”))

confusionMatrix(xgb\_model\_pred\_class, test$ChurnStatus, positive = “yes”)

*# Extract raw xgb.Booster from caret model*

booster\_model <- xgb\_model$finalModel

feature\_importance <- xgb.importance(model = booster\_model)

print(feature\_importance)

xgb.plot.importance(feature\_importance, xlab = "Relative importance (Gain)",col= "maroon", top\_n = 15)

### **1.3 Performance Metrics & Feature Importance**

* **ROC-AUC**: Measures discrimination ability across thresholds.
* **Sensitivity (Recall / True Positive Rate)**: Focus on identifying churners (positive class) accurately.
* **Specificity (True Negative Rate)**: Specificity (also called the True Negative Rate) is the proportion of actual negative cases that are correctly identified by the model.
* A screenshot of a computer

  AI-generated content may be incorrect.**Confusion Matrix**: Visualise true positives, false positives, etc.

### A graph with a bar graph AI-generated content may be incorrect.A diagram with blue squares AI-generated content may be incorrect.

### **A graph of a number of different colored lines AI-generated content may be incorrect.**

### **1.4. Recommendations for Business Application**

* **Real-time Scoring**: Integrate the model into CRM to score customers on churn risk at each login.
* **Targeted Retention Campaigns**: Use risk scores to trigger personalized offers or outreach for high-risk segments.
* **Feature Monitoring**: Continuously monitor key drivers (e.g., login frequency drops) and recalibrate model as behaviors evolve.

### **1.5 Future Improvements**

* **Ensemble Approaches**: Combine multiple algorithms (e.g., RF + GBM) for robust predictions.
* **Additional Data Sources**: Incorporate customer support interactions, sentiment analysis, or external macroeconomic indicators.
* **Explainability Tools**: Use SHAP or LIME to provide transparency into individual predictions for stakeholders.