Machine Learning For Appointments Capstone Project

Background

This problem is a prediction of missed medical appointments. The goal is to predict whether the appointment will be kept or missed, and the probability of each class, which is a supervised classification problem.

The models chosen to test the missed appointment predictions are glm and random forest, which will both be used with the caret package. The glm model is chosen because this is a binary classification problem, where it is also useful to predict the probabilities of each outcome, and the glm is well suited for these tasks.

Random forest will be used because it will be able to pick up on some non-linearities that the glm can't since it is a linear model. LIke the glm, it also has the ability to predict the class probabilities.

The chosen features are summarized below:

```
## # A tibble: 14 x 2
##    ml_columns           ml_descriptions
##    <chr>                <chr>
##  1 appt_length          Appointment length in Minutes
##  2 patient_age          Patient Age in Years
##  3 patient_gender       Patient gender
##  4 billing_type         Billing type
##  5 patient_distance     Patient distance from office in miles
##  6 provider_specialty   Provider primary specialty code
##  7 remind_call_result   Reminder Call result
##  8 hour                 Hour of Day appointment occurs
##  9 prior_percent_missed Percent of prior appointments missed by patient
## 10 appt_lead_time       Lead time between day appt was booked and when it~
## 11 appt_weekday         Weekday Appointment Occurs
## 12 is_new_patient       Is the patient new or existing?
## 13 weekday_scheduled    Weekday appointent was booked
## 14 county_code          county where the medical office is located
```

Set Up Model Parameters

```
control <- caret::trainControl(method = "cv", number = 2, classProbs = TRUE)
seed <- 7
metric <- "Accuracy"
set.seed(seed)
mtry <- 3
tunegrid <- expand.grid(.mtry = mtry)
```

Logistic Regression Model

```
glm_model <- caret::train(
    kept_status ~ .,
    data = train_balanced,
    method = "glm",
    trControl = control
)
```

Random Forest Model

```
rf_model <- caret::train(
    kept_status ~ ., data = train_balanced, method = "rf", metric = metric,
    tuneGrid = tunegrid, trControl = control)
```

Model Selection

To select the best model, I will first make predictions for each model on the validation set using the `predict` function in caret. This will allow me to have caret produce a confusion matrix and calculate the F1 score. The F1 score, which is a harmonic average of precision and recall, will be used to select the best model. A high recall is desired because when the actual appointment is missed, we want the model to predict it, while a high precision is desired because when the model predicts missed, we want the actual appointment to be missed.

```
pred_glm <- predict(glm_model, validate)

conf_mat_glm <- caret::confusionMatrix(
    pred_glm, validate$kept_status, positive = "Missed")

conf_mat_glm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  Kept Missed
##     Kept   41656   3329
##     Missed 15428   8127
##
##                Accuracy : 0.7263
##                  95% CI : (0.723, 0.7297)
##     No Information Rate : 0.8329
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.3088
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7094
##             Specificity : 0.7297
##          Pos Pred Value : 0.3450
##          Neg Pred Value : 0.9260
##              Prevalence : 0.1671
##          Detection Rate : 0.1186
##    Detection Prevalence : 0.3437
##       Balanced Accuracy : 0.7196
##
##        'Positive' Class : Missed
##
```

```
conf_mat_glm$byClass["F1"]
```

```
##        F1
## 0.4642541
```

```
pred_rf <- predict(rf_model, validate)

conf_mat_rf <- caret::confusionMatrix(
    pred_rf, validate$kept_status, positive = "Missed")

conf_mat_rf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  Kept Missed
##     Kept   40187   2442
##     Missed 16897   9014
##
##                Accuracy : 0.7178
##                  95% CI : (0.7145, 0.7212)
##     No Information Rate : 0.8329
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.3263
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7868
##             Specificity : 0.7040
##          Pos Pred Value : 0.3479
##          Neg Pred Value : 0.9427
##              Prevalence : 0.1671
##          Detection Rate : 0.1315
##    Detection Prevalence : 0.3780
##       Balanced Accuracy : 0.7454
##
##        'Positive' Class : Missed
##
```

```
conf_mat_rf$byClass["F1"]
```

```
##        F1
## 0.4824578
```

Looking at the F1 scores based on the model predictions on the validation set, the random forest model outperforms the glm model, with a score of 0.4824578 compared to the glm model with a score of 0.4642541. The difference is fairly small and likely indicates that there are some non-linearities that the random forest is better able to pick up. Because of this advantage, random forst will be the chosen model. The next step is to evaluate the expected performance on new data.

Expected Model Performance

Now that the random forest model has been selected, I will see how well it performs on the test data. Since the test data is the most recent, the performance on this data should be the best representation of how well the model will perform on future unseen information.

```
final_pred_rf <- predict(rf_model, test)

final_conf_mat_rf <- caret::confusionMatrix(
    final_pred_rf, test$kept_status, positive = "Missed")
```

```
final_conf_mat_rf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  Kept Missed
##     Kept   41596   2671
##     Missed 14838   9415
##
##                Accuracy : 0.7445
##                  95% CI : (0.7412, 0.7477)
##     No Information Rate : 0.8236
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.3698
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7790
##             Specificity : 0.7371
##          Pos Pred Value : 0.3882
##          Neg Pred Value : 0.9397
##              Prevalence : 0.1764
##          Detection Rate : 0.1374
##    Detection Prevalence : 0.3540
##       Balanced Accuracy : 0.7580
##
##        'Positive' Class : Missed
##
```

```
final_conf_mat_rf$byClass["F1"]
```

```
##        F1
## 0.5181761
```

The F1 score is 0.4824578 on the test set, which is higher than it was based on the validation set. This is a fairly significant improvement over the performance on the validation set. I'm not sure why it performs better on the test data, but my speculation is there is some time dependence that works in favor of the test set.

The positive predictive value of 0.3881994 means that when the model predicts a miss, it is expcted to be correct 38.82 % of the time and incorrect 61.18 % of the time. Predictions of "Miss" are expected to be incorrect more often than they are expected to be correct. While this is a little disappointing, it helps to look at the improvement over having no model at all, and it also helps to remember that the misses the model is trying to predict can be caused by manny things that the data doesn't directly capture. With no model, predicting an appointment as missed is expected to be correct approximately 15.93 % of the time, based on the overall rate of missed appointments. 61.18 % incorrect is a significant improvement over 84.07 % incorrect.