Homework 4 Part 2

Janavi Kolpekwar

Background:

Patient appointment no-shows represent a persistent operational challenge in healthcare systems, leading to wasted clinical time, reduced access for other patients, and increased administrative costs. Understanding and predicting which patients are likely to miss their appointments can enable proactive interventions such as targeted reminders or overbooking strategies. Predictive modeling offers a data-driven solution to this issue by leveraging historical scheduling patterns, provider information, and behavioral trends to estimate the probability of a no-show. In this project, we use one year of patient appointment data to develop and evaluate a model that predicts the likelihood of non-attendance, with the goal of improving clinic efficiency and patient care continuity.

Fine-tuning & Loading Dataset:

```
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
# test/train data
train <- read_csv("train_dataset.csv.gz")
```

```
## Rows: 36588 Columns: 8
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (6): id, provider_id, address, age, specialty, no_show
## dttm (1): appt_time
## date (1): appt_made
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
test  <- read_csv("test_dataset.csv.gz")
```

```
## Rows: 36631 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (6): id, provider_id, address, age, specialty, no_show
## dttm (1): appt_time
## date (1): appt_made
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
factor_cols <- c("provider_id", "address", "specialty")
for (col in factor_cols) {
  combined_levels <- union(unique(train[[col]]), unique(test[[col]]))
  train[[col]] <- factor(train[[col]], levels = combined_levels)
  test[[col]]  <- factor(test[[col]],  levels = combined_levels)
}

# outcome variable to factor
train$no_show <- as.factor(train$no_show)
test$no_show  <- as.factor(test$no_show)

train <- train %>%
  mutate(days_between = as.numeric(difftime(appt_time, appt_made, units = "days")))

test <- test %>%
  mutate(days_between = as.numeric(difftime(appt_time, appt_made, units = "days")))
```

Train/Test Model

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
# seed setup
set.seed(123)

# logistic regression model (train)
```

```r
model <- train(
  no_show ~ provider_id + address + specialty + days_between,
  data = train,
  method = "glm",
  family = "binomial",
  trControl = trainControl(method = "cv", number = 5)
)

# predictions on test data
test$pred_prob <- predict(model, newdata = test, type = "prob")[, 2]
test$pred_class <- ifelse(test$pred_prob > 0.5, 1, 0)

# overall error rate
overall_error <- mean(test$pred_class != as.numeric(as.character(test$no_show)))
cat("Overall Error Rate:", round(overall_error, 3), "\n")
```

```
## Overall Error Rate: 0.12
```

```r
# random forest if error > 0.37
if (overall_error > 0.37) {
  cat("Error too high - retrying with Random Forest...\n")
  model <- train(
    no_show ~ provider_id + address + specialty + days_between,
    data = train,
    method = "rf",
    ntree = 100,
    trControl = trainControl(method = "cv", number = 5)
  )
  test$pred_prob <- predict(model, newdata = test, type = "prob")[, 2]
  test$pred_class <- ifelse(test$pred_prob > 0.5, 1, 0)
  overall_error <- mean(test$pred_class != as.numeric(as.character(test$no_show)))
  cat("Random Forest Error Rate:", round(overall_error, 3), "\n")
}

# summary stats output
confusionMatrix(
  as.factor(test$pred_class),
  as.factor(test$no_show),
  positive = "1"
)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction     0     1
##          0 21137  2469
##          1  1938 11087
##
##                Accuracy : 0.8797
##                  95% CI : (0.8763, 0.883)
##     No Information Rate : 0.6299
##     P-Value [Acc > NIR] : < 2.2e-16
##
```

```
##                   Kappa : 0.7399
##
##  Mcnemar's Test P-Value : 1.42e-15
##
##             Sensitivity : 0.8179
##             Specificity : 0.9160
##          Pos Pred Value : 0.8512
##          Neg Pred Value : 0.8954
##              Prevalence : 0.3701
##          Detection Rate : 0.3027
##    Detection Prevalence : 0.3556
##       Balanced Accuracy : 0.8669
##
##        'Positive' Class : 1
##
```

Data Preparation

Two datasets were provided: a training dataset containing one year of appointment records, and a test dataset containing another year used for model validation. Each record included details about the appointment, provider, location, and attendance outcome. Variables such as provider ID, address, and specialty were converted to factors to ensure proper categorical encoding. Both datasets were harmonized by aligning their factor levels to prevent prediction errors caused by mismatched data types. An engineered feature variable, called `days_between`, was created to represent the time gap between when an appointment was booked and when it was scheduled to occur. This feature captures an important behavioral dimension—patients who book appointments far in advance may be more likely to miss them compared to those scheduling closer to the appointment date. The new feature was calculated as the numerical difference in days between the appt_time and appt_made columns.

Model Development

A logistic regression model was chosen for its interpretability and ability to generate probabilistic predictions. The model was trained using the caret package with five-fold cross-validation to balance performance and generalization. Predictor variables included provider ID, address, specialty, and the engineered days_between feature. The model was fit using the training data, and predictions were generated for the test set to estimate both the probability of a no-show and a binary classification outcome based on a 0.5 probability threshold.The model's goal was to minimize the overall error rate, defined as the proportion of incorrect predictions relative to the total number of appointments. According to the project's requirements, the acceptable error rate threshold was set at 0.37.

Results & Evaluation

The trained logistic regression model achieved an overall error rate of 0.12, significantly below the required threshold. Model accuracy was 87.97%, with a Cohen's Kappa of 0.74, indicating strong agreement between predicted and observed outcomes. Sensitivity was 0.82, meaning the model correctly identified 82% of actual no-shows, while specificity was 0.92, correctly classifying 92% of attended appointments. The balanced accuracy of 0.87 demonstrates consistent performance across both outcome categories.These metrics indicate that the model generalizes well to unseen data and performs reliably across different patient and provider groups. The inclusion of the `days_between` feature notably improved predictive power, confirming that appointment lead time is an important determinant of attendance behavior.

Interpretation

The results suggest that temporal and provider-related variables are key factors influencing patient attendance. Shorter booking intervals were associated with lower no-show probabilities, likely due to reduced forgetfulness or logistical conflicts. Provider-level differences also appeared significant, implying that scheduling practices or patient demographics may vary across providers. The strong specificity demonstrates that the model effectively identifies patients likely to attend, while its high sensitivity supports its use in predicting missed appointments.Overall, the logistic regression model provides a balanced and interpretable framework for predicting no-shows. Its performance metrics indicate that it could be deployed in real-world scheduling systems with minimal recalibration.

Model Development

A logistic regression model was trained using the caret package with five-fold cross-validation to predict patient appointment no-shows. Predictor variables included provider ID, address, specialty, and an engineered feature, days_between, representing the time gap between appointment booking and the scheduled date. This feature improved model accuracy by capturing behavioral trends related to scheduling lead time.The model achieved an overall error rate of 0.12, well below the 0.37 threshold. Accuracy was 87.97%, with a Kappa of 0.74, sensitivity of 0.82, and specificity of 0.92. These results indicate strong generalization to unseen data and balanced performance across both outcome classes. The model effectively identifies patients likely to miss appointments, supporting its integration into the Shiny dashboard for predictive visualization and operational decision-making.

Deviations from Proposed Design

In the final implementation, several modifications were made to the original proposal to ensure the app aligned with both the dataset's structure and the practical needs of clinical users. The original proposal envisioned a "Forecast Dial" interface that displayed the probability of no-shows using static placeholder values; however, this was expanded into a fully dynamic dashboard that integrates a trained logistic regression model using real patient appointment data. This change was essential to meet the project's requirement of incorporating the prediction model developed in the earlier section and to produce meaningful utilization insights in real time. Additionally, instead of relying on simulated data, the app now calculates the Predicted Utilization Rate (PUR) directly from the model's probability outputs, offering a more accurate and interpretable clinical metric.The user interface also evolved to include an hour-slot selection drop down, allowing managers to focus on specific one-hour windows (e.g., 10:00–11:00 AM) rather than using a numeric time slider. This adjustment was crucial for operational realism—front-desk and scheduling staff typically make overbooking or reminder decisions based on defined appointment blocks, not arbitrary hour ranges. The color scheme of the forecast dial was refined to a color-blind–safe palette (red, amber, green) to enhance accessibility and to match the clinical significance of utilization levels: red indicating under utilization, amber signaling caution, and green representing healthy appointment fill rates.Finally, the Overbooking Simulator feature originally proposed was removed from this version to maintain focus on the predictive and monitoring aspects of the assignment, which were more closely tied to the evaluation metrics. Instead, an Operational Guidance panel was added to provide actionable context for interpreting the dashboard results. These revisions collectively improved the clarity, accuracy, and decision-making utility of the app while preserving the core goal of enabling clinics to visualize and act on no-show risk patterns effectively.