Homework4p2

Bo Han

2025-10-29

Homework 4p2

```
library(tidyverse)
library(lubridate)

train = read.csv("train_dataset.csv.gz")
test = read.csv("test_dataset.csv.gz")
```

```
train = train %>%
  mutate(
    appt_time = ymd_hms(appt_time, tz="UTC"),
    appt_date = as.Date(appt_time),
    appt_hour = hour(appt_time),
    appt_day = wday(appt_time, label=T, abbr=T),
    diff_time = as.numeric(difftime(appt_date, as.Date(appt_made), units="days")))
test = test %>%
  mutate(
    appt_time = ymd_hms(appt_time, tz="UTC"),
    appt_date = as.Date(appt_time),
    appt_hour = hour(appt_time),
    appt_day = wday(appt_time, label=T, abbr=T),
    diff_time = as.numeric(difftime(appt_date, as.Date(appt_made), units="days")))
```

Transformation of Variables The code above standardizes the appt_time variable from the original datasets, and adds appt_date, appt_hour, appt_day, and diff_time. appt_date takes the date component from appt_date values, appt_hour takes the hour component from appt_date values, appt_day converts the date components from appt_date values into days of the week, and diff_time is the difference between appt_date and appt_made in days. These new variables will be used as predictors in our model.

```
model = glm(no_show ~ appt_day + appt_hour + diff_time,
            data=train, family=binomial())
summary(model)
```

Prediction Model

```
##
## Call:
## glm(formula = no_show ~ appt_day + appt_hour + diff_time, family = binomial(),
##     data = train)
##
## Coefficients:
```

```
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept) -30.082957   0.354237 -84.923   <2e-16 ***
## appt_day.L    0.065884   0.048728   1.352    0.176
## appt_day.Q   -0.072601   0.048746  -1.489    0.136
## appt_day.C    0.046033   0.048801   0.943    0.346
## appt_day^4   -0.056548   0.048976  -1.155    0.248
## appt_day^5    0.006353   0.048879   0.130    0.897
## appt_day^6    0.036212   0.048969   0.739    0.460
## appt_hour     0.318463   0.007760  41.041   <2e-16 ***
## diff_time     0.384404   0.004367  88.034   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 48291  on 36587  degrees of freedom
## Residual deviance: 18881  on 36579  degrees of freedom
## AIC: 18899
##
## Number of Fisher Scoring iterations: 7
```

The code above implements a model that predicts no_show by appt_day, appt_hour, and diff_time using logistic regression on our training data. Logistic regression is used because no_show is a binary outcome, logistic regression outputs probabilities, and it is simple and easy to interpret.

```r
test$pred_prob = predict(model, newdata=test, type="response")
test$pred_no_show = if_else(test$pred_prob >= 0.5, 1, 0)

error = mean(test$pred_no_show != test$no_show)
error
```

Predict No Shows

```
## [1] 0.1132647
```

The code above adds the pred_prob and pred_no_show variables to the test dataset. pred_prob includes the probabilities of patients being a no_show considering our model implemented earlier. pred_no_show categorizes each patient as a no-show or not a no-show using the pred_prob probabilities (1 and 0, respectively). The overall error rate is also calculated (0.11), which is well below the threshold (0.37).

Deviations From Proposed Design I had very little to no deviations from the design I proposed in Homework 4p1.