MGSC 310 Final

Justin Lewinski

2023-05-16

library(readr)  
library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──  
## ✔ ggplot2 3.4.1 ✔ dplyr 1.1.0  
## ✔ tibble 3.1.8 ✔ stringr 1.5.0  
## ✔ tidyr 1.3.0 ✔ forcats 1.0.0  
## ✔ purrr 1.0.1   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

library(ggplot2)  
library(caret)

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

library(plotROC)  
library(ggcorrplot)  
library(ISLR)  
library(yardstick)

## For binary classification, the first factor level is assumed to be the event.  
## Use the argument `event\_level = "second"` to alter this as needed.  
##   
## Attaching package: 'yardstick'  
##   
## The following objects are masked from 'package:caret':  
##   
## precision, recall, sensitivity, specificity  
##   
## The following object is masked from 'package:readr':  
##   
## spec

library(tidyverse)  
library(rsample)  
library(glmnet)

## Loading required package: Matrix  
##   
## Attaching package: 'Matrix'  
##   
## The following objects are masked from 'package:tidyr':  
##   
## expand, pack, unpack  
##   
## Loaded glmnet 4.1-7

library(glmnetUtils)

##   
## Attaching package: 'glmnetUtils'  
##   
## The following objects are masked from 'package:glmnet':  
##   
## cv.glmnet, glmnet

library(forcats)  
library(randomForestExplainer)

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

library(ggplot2)  
library(randomForest)

## randomForest 4.7-1.1  
## 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

# 2

data <- read\_csv('datasets/booking-1.csv')

## Rows: 2632 Columns: 24  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## dbl (23): srch\_id, site\_id, visitor\_country\_id, hotel\_country\_id, hotel\_id,...  
## dttm (1): date\_time  
##   
## 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.

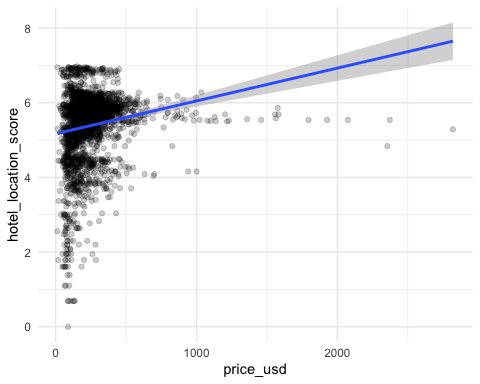
data %>% summary()

## srch\_id date\_time site\_id   
## Min. : 93 Min. :2012-11-06 05:18:44.00 Min. : 1.00   
## 1st Qu.:214708 1st Qu.:2013-02-10 17:09:01.00 1st Qu.: 5.00   
## Median :218049 Median :2013-04-26 03:07:51.00 Median :12.00   
## Mean :259787 Mean :2013-04-04 15:55:42.73 Mean :11.96   
## 3rd Qu.:237124 3rd Qu.:2013-05-24 21:51:53.00 3rd Qu.:16.00   
## Max. :665537 Max. :2013-06-30 19:02:59.00 Max. :34.00   
## visitor\_country\_id hotel\_country\_id hotel\_id hotel\_stars   
## Min. : 2.0 Min. : 31.0 Min. : 223 Min. :0.000   
## 1st Qu.: 59.0 1st Qu.: 99.0 1st Qu.: 31304 1st Qu.:3.000   
## Median :129.0 Median :129.0 Median : 63445 Median :4.000   
## Mean :141.3 Mean :137.8 Mean : 66169 Mean :3.776   
## 3rd Qu.:219.0 3rd Qu.:219.0 3rd Qu.:101677 3rd Qu.:4.000   
## Max. :228.0 Max. :219.0 Max. :140694 Max. :5.000   
## hotel\_review\_score hotel\_chain hotel\_location\_score  
## Min. :0.000 Min. :0.0000 Min. :0.000   
## 1st Qu.:4.000 1st Qu.:0.0000 1st Qu.:5.110   
## Median :4.000 Median :1.0000 Median :5.660   
## Mean :4.022 Mean :0.5129 Mean :5.387   
## 3rd Qu.:4.500 3rd Qu.:1.0000 3rd Qu.:5.900   
## Max. :5.000 Max. :1.0000 Max. :6.980   
## hotel\_historical\_price search\_ranking price\_usd promotion   
## Min. :0.000 Min. : 1.00 Min. : 12.0 Min. :0.0000   
## 1st Qu.:4.940 1st Qu.: 6.00 1st Qu.: 140.7 1st Qu.:0.0000   
## Median :5.450 Median :13.00 Median : 216.0 Median :0.0000   
## Mean :4.598 Mean :14.73 Mean : 246.1 Mean :0.3891   
## 3rd Qu.:5.790 3rd Qu.:24.00 3rd Qu.: 301.1 3rd Qu.:1.0000   
## Max. :6.210 Max. :36.00 Max. :2820.0 Max. :1.0000   
## length\_of\_stay booking\_window adults\_count children\_count   
## Min. : 1.000 Min. : 0.00 Min. :1.000 Min. :0.0000   
## 1st Qu.: 1.000 1st Qu.: 5.00 1st Qu.:1.000 1st Qu.:0.0000   
## Median : 3.000 Median : 20.00 Median :2.000 Median :0.0000   
## Mean : 2.803 Mean : 27.99 Mean :1.874 Mean :0.3131   
## 3rd Qu.: 4.000 3rd Qu.: 46.00 3rd Qu.:2.000 3rd Qu.:1.0000   
## Max. :13.000 Max. :173.00 Max. :7.000 Max. :3.0000   
## room\_count saturday\_night random\_sort comp\_rate   
## Min. :1.000 Min. :0.0000 Min. :0.0000 Min. :-1.0000   
## 1st Qu.:1.000 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.: 0.0000   
## Median :1.000 Median :0.0000 Median :0.0000 Median : 1.0000   
## Mean :1.267 Mean :0.4734 Mean :0.2272 Mean : 0.5258   
## 3rd Qu.:1.000 3rd Qu.:1.0000 3rd Qu.:0.0000 3rd Qu.: 1.0000   
## Max. :7.000 Max. :1.0000 Max. :1.0000 Max. : 1.0000   
## comp\_inv booking   
## Min. :-1.000 Min. :0.0000   
## 1st Qu.: 0.000 1st Qu.:0.0000   
## Median : 0.000 Median :0.0000   
## Mean : 0.019 Mean :0.3625   
## 3rd Qu.: 0.000 3rd Qu.:1.0000   
## Max. : 1.000 Max. :1.0000

data\_clean <-   
 data %>%   
 mutate(reviewscore2 = hotel\_review\_score\*\*2,  
 hotel\_stars2 = hotel\_stars\*\*2,  
 price2 = price\_usd\*\*2,  
 booking\_window\_2 = booking\_window\*\*2)

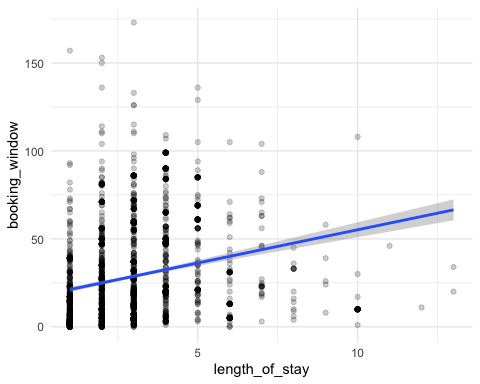
ggplot(data\_clean, aes(x = price\_usd, y = hotel\_location\_score)) + geom\_point(alpha = 0.2) + geom\_smooth(method = lm) + theme\_minimal()

## `geom\_smooth()` using formula = 'y ~ x'



ggplot(data\_clean, aes(x = length\_of\_stay, y = booking\_window)) + geom\_point(alpha = 0.2) + geom\_smooth(method = lm) + theme\_minimal()

## `geom\_smooth()` using formula = 'y ~ x'



# 3

set.seed(2370404)  
initial\_split\_hotel <- initial\_split(data\_clean, prop = .75)  
hotel\_train <- training(initial\_split\_hotel)  
hotel\_test <- testing(initial\_split\_hotel)

hotel\_train %>% glimpse

## Rows: 1,974  
## Columns: 28  
## $ srch\_id <dbl> 312781, 218049, 215038, 237161, 322621, 86424, …  
## $ date\_time <dttm> 2013-05-13 20:38:38, 2013-01-31 15:29:28, 2013…  
## $ site\_id <dbl> 15, 24, 5, 7, 24, 15, 16, 24, 15, 14, 5, 32, 15…  
## $ visitor\_country\_id <dbl> 129, 216, 219, 219, 216, 55, 31, 99, 55, 100, 2…  
## $ hotel\_country\_id <dbl> 129, 56, 219, 219, 181, 132, 60, 99, 99, 99, 99…  
## $ hotel\_id <dbl> 62765, 79465, 89359, 68487, 110250, 32831, 1357…  
## $ hotel\_stars <dbl> 5, 4, 2, 4, 4, 4, 4, 3, 3, 3, 3, 4, 3, 4, 3, 4,…  
## $ hotel\_review\_score <dbl> 4.5, 4.0, 2.5, 4.0, 4.0, 4.0, 4.0, 4.0, 3.5, 4.…  
## $ hotel\_chain <dbl> 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0,…  
## $ hotel\_location\_score <dbl> 4.66, 4.82, 5.79, 5.61, 4.33, 5.54, 4.47, 5.10,…  
## $ hotel\_historical\_price <dbl> 5.46, 5.26, 5.48, 5.73, 0.00, 5.49, 4.77, 5.28,…  
## $ search\_ranking <dbl> 8, 9, 22, 2, 1, 2, 2, 20, 15, 18, 9, 8, 16, 8, …  
## $ price\_usd <dbl> 183.57, 121.26, 335.00, 286.88, 106.64, 134.06,…  
## $ promotion <dbl> 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,…  
## $ length\_of\_stay <dbl> 3, 6, 3, 7, 3, 1, 2, 1, 5, 3, 1, 10, 1, 1, 1, 2…  
## $ booking\_window <dbl> 4, 5, 21, 23, 20, 32, 6, 0, 69, 76, 3, 108, 23,…  
## $ adults\_count <dbl> 2, 2, 2, 1, 1, 2, 1, 2, 1, 2, 4, 3, 3, 2, 4, 2,…  
## $ children\_count <dbl> 1, 0, 3, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1,…  
## $ room\_count <dbl> 1, 1, 1, 2, 1, 1, 2, 1, 1, 1, 2, 1, 2, 1, 2, 2,…  
## $ saturday\_night <dbl> 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0,…  
## $ random\_sort <dbl> 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0,…  
## $ comp\_rate <dbl> 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,…  
## $ comp\_inv <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,…  
## $ booking <dbl> 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1,…  
## $ reviewscore2 <dbl> 20.25, 16.00, 6.25, 16.00, 16.00, 16.00, 16.00,…  
## $ hotel\_stars2 <dbl> 25, 16, 4, 16, 16, 16, 16, 9, 9, 9, 9, 16, 9, 1…  
## $ price2 <dbl> 33697.945, 14703.988, 112225.000, 82300.134, 11…  
## $ booking\_window\_2 <dbl> 16, 25, 441, 529, 400, 1024, 36, 0, 4761, 5776,…

logit <- glm(booking ~ hotel\_stars + hotel\_review\_score + hotel\_chain +   
 hotel\_location\_score + hotel\_historical\_price + search\_ranking  
 + price\_usd + promotion + length\_of\_stay + booking\_window +   
 adults\_count + children\_count + room\_count + saturday\_night  
 + comp\_rate + comp\_inv, data = hotel\_train, family = binomial)  
logit %>% summary()

##   
## Call:  
## glm(formula = booking ~ hotel\_stars + hotel\_review\_score + hotel\_chain +   
## hotel\_location\_score + hotel\_historical\_price + search\_ranking +   
## price\_usd + promotion + length\_of\_stay + booking\_window +   
## adults\_count + children\_count + room\_count + saturday\_night +   
## comp\_rate + comp\_inv, family = binomial, data = hotel\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.9258 -0.8535 -0.4898 0.9502 2.7506   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 1.2750113 0.4747510 2.686 0.007239 \*\*   
## hotel\_stars 0.1433822 0.0654857 2.190 0.028559 \*   
## hotel\_review\_score 0.0418149 0.0756897 0.552 0.580639   
## hotel\_chain 0.0729350 0.1129255 0.646 0.518365   
## hotel\_location\_score 0.1205084 0.0570104 2.114 0.034533 \*   
## hotel\_historical\_price -0.1177641 0.0255086 -4.617 3.90e-06 \*\*\*  
## search\_ranking -0.0799179 0.0056578 -14.125 < 2e-16 \*\*\*  
## price\_usd -0.0052525 0.0005691 -9.230 < 2e-16 \*\*\*  
## promotion 0.0586445 0.1143186 0.513 0.607957   
## length\_of\_stay -0.0498322 0.0345273 -1.443 0.148944   
## booking\_window 0.0008761 0.0021523 0.407 0.683965   
## adults\_count 0.0974074 0.0683817 1.424 0.154311   
## children\_count 0.4020869 0.1068958 3.761 0.000169 \*\*\*  
## room\_count -0.5048533 0.1202061 -4.200 2.67e-05 \*\*\*  
## saturday\_night 0.0141528 0.1131327 0.125 0.900445   
## comp\_rate -0.0018489 0.0855885 -0.022 0.982765   
## comp\_inv -0.5303007 0.3260470 -1.626 0.103853   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 2592.5 on 1973 degrees of freedom  
## Residual deviance: 2130.7 on 1957 degrees of freedom  
## AIC: 2164.7  
##   
## Number of Fisher Scoring iterations: 5

exp(logit$coefficients)

## (Intercept) hotel\_stars hotel\_review\_score   
## 3.5787418 1.1541709 1.0427015   
## hotel\_chain hotel\_location\_score hotel\_historical\_price   
## 1.0756606 1.1280703 0.8889057   
## search\_ranking price\_usd promotion   
## 0.9231921 0.9947612 1.0603982   
## length\_of\_stay booking\_window adults\_count   
## 0.9513890 1.0008765 1.1023094   
## children\_count room\_count saturday\_night   
## 1.4949413 0.6035941 1.0142534   
## comp\_rate comp\_inv   
## 0.9981528 0.5884280

scores\_train <- predict(logit, typ = "response")  
scores\_test <- predict(logit, type = "response", newdata = hotel\_test)

predicted\_train<- ifelse(scores\_train>0.5,"1","0")  
head(predicted\_train)

## 1 2 3 4 5 6   
## "1" "1" "0" "0" "1" "1"

predicted\_test <- ifelse(scores\_test>0.5,"1","0")  
head(predicted\_test)

## 1 2 3 4 5 6   
## "0" "0" "0" "0" "0" "0"

results\_train <- data.frame(  
 true = factor(hotel\_train$booking),  
 predicted = factor(predicted\_train),  
 score = scores\_train)  
sum(results\_train$predicted == 1)

## [1] 604

sum(results\_train$predicted == 0)

## [1] 1370

results\_test <- data.frame(  
 true = factor(hotel\_test$booking),  
 predicted = factor(predicted\_test),  
 score = scores\_test)  
results\_test %>% glimpse()

## Rows: 658  
## Columns: 3  
## $ true <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, …  
## $ predicted <fct> 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, …  
## $ score <dbl> 0.3066496687, 0.2589455048, 0.3094074926, 0.0007639443, 0.02…

cm\_trainlogit <- conf\_mat(results\_train,  
 truth = true,  
 estimate = predicted)  
  
cm\_testlogit <- conf\_mat(results\_test,  
 truth = true,  
 estimate = predicted)  
  
print(cm\_trainlogit)

## Truth  
## Prediction 0 1  
## 0 1022 348  
## 1 230 374

print(cm\_testlogit)

## Truth  
## Prediction 0 1  
## 0 353 98  
## 1 73 134

TN\_trainlogit = 1022  
TP\_trainlogit = 374  
FN\_trainlogit = 348  
FP\_trainlogit = 230  
  
  
  
TN\_testlogit = 353  
TP\_testlogit = 134  
FN\_testlogit = 98  
FP\_testlogit = 73

print('train\_scores')

## [1] "train\_scores"

acc\_trainlogit = (TN\_trainlogit +TP\_trainlogit ) / (TN\_trainlogit + TP\_trainlogit + FN\_trainlogit + FP\_trainlogit)  
print(acc\_trainlogit)

## [1] 0.7071935

sen\_trainlogit = TP\_trainlogit/(TP\_trainlogit +FN\_trainlogit)  
print(sen\_trainlogit)

## [1] 0.5180055

spe\_trainlogit = TN\_trainlogit/(TN\_trainlogit + FP\_trainlogit)  
  
print(spe\_trainlogit)

## [1] 0.8162939

print('test scores')

## [1] "test scores"

acc\_testlogit = (TN\_testlogit +TP\_testlogit ) / (TN\_testlogit + TP\_testlogit + FN\_testlogit + FP\_testlogit)  
print(acc\_testlogit)

## [1] 0.7401216

sen\_testlogit = TP\_testlogit/(TP\_testlogit +FN\_testlogit)  
print(sen\_testlogit)

## [1] 0.5775862

spe\_testlogit = TN\_testlogit/(TN\_testlogit + FP\_testlogit)  
  
print(spe\_testlogit)

## [1] 0.8286385

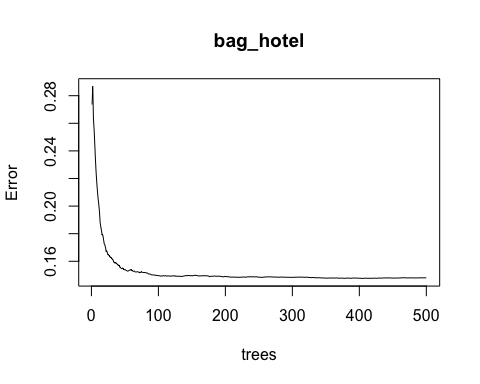
bag\_hotel <- randomForest(booking ~ hotel\_stars + hotel\_review\_score + hotel\_chain +   
 hotel\_location\_score + hotel\_historical\_price + search\_ranking  
 + price\_usd + promotion + length\_of\_stay + booking\_window +   
 adults\_count + children\_count + room\_count + saturday\_night  
 + comp\_rate + comp\_inv, data = hotel\_train)

## Warning in randomForest.default(m, y, ...): The response has five or fewer  
## unique values. Are you sure you want to do regression?

print(bag\_hotel)

##   
## Call:  
## randomForest(formula = booking ~ hotel\_stars + hotel\_review\_score + hotel\_chain + hotel\_location\_score + hotel\_historical\_price + search\_ranking + price\_usd + promotion + length\_of\_stay + booking\_window + adults\_count + children\_count + room\_count + saturday\_night + comp\_rate + comp\_inv, data = hotel\_train)   
## Type of random forest: regression  
## Number of trees: 500  
## No. of variables tried at each split: 5  
##   
## Mean of squared residuals: 0.1480358  
## % Var explained: 36.19

plot(bag\_hotel)



scores\_trainbag <- predict(bag\_hotel, typ = "response")  
scores\_testbag <- predict(bag\_hotel, type = "response", newdata = hotel\_test)  
  
predicted\_trainbag<- ifelse(scores\_trainbag>0.5,"1","0")  
head(predicted\_trainbag)

## 1 2 3 4 5 6   
## "0" "0" "0" "0" "1" "1"

predicted\_testbag <- ifelse(scores\_testbag>0.5,"1","0")  
head(predicted\_testbag)

## 1 2 3 4 5 6   
## "0" "0" "0" "0" "0" "0"

results\_trainbag <- data.frame(  
 true = factor(hotel\_train$booking),  
 predicted = factor(predicted\_trainbag),  
 score = scores\_trainbag)  
sum(results\_trainbag$predicted == 1)

## [1] 594

sum(results\_trainbag$predicted == 0)

## [1] 1380

results\_testbag <- data.frame(  
 true = factor(hotel\_test$booking),  
 predicted = factor(predicted\_testbag),  
 score = scores\_testbag)  
results\_testbag %>% glimpse()

## Rows: 658  
## Columns: 3  
## $ true <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, …  
## $ predicted <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, …  
## $ score <dbl> 0.23620000, 0.20953333, 0.25796667, 0.07130000, 0.17780000, …

cm\_trainbag <- conf\_mat(results\_trainbag,  
 truth = true,  
 estimate = predicted)  
  
cm\_testbag <- conf\_mat(results\_testbag,  
 truth = true,  
 estimate = predicted)  
print(cm\_trainbag)

## Truth  
## Prediction 0 1  
## 0 1103 277  
## 1 149 445

print(cm\_testbag)

## Truth  
## Prediction 0 1  
## 0 379 80  
## 1 47 152

TN\_trainbag = 1108  
TP\_trainbag = 443  
FN\_trainbag = 279  
FP\_trainbag = 144  
  
TN\_testbag = 379  
TP\_testbag = 151  
FN\_testbag = 81  
FP\_testbag = 47

print('train scores forest')

## [1] "train scores forest"

acc\_trainbag = (TN\_trainbag +TP\_trainbag ) / (TN\_trainbag + TP\_trainbag + FN\_trainbag + FP\_trainbag)  
print(acc\_trainbag)

## [1] 0.7857143

sen\_trainbag = TP\_trainbag/(TP\_trainbag +FN\_trainbag)  
print(sen\_trainbag)

## [1] 0.6135734

spe\_trainbag = TN\_trainbag/(TN\_trainbag + FP\_trainbag)  
  
print(spe\_trainbag)

## [1] 0.884984

print('test scores forest')

## [1] "test scores forest"

acc\_testbag = (TN\_testbag +TP\_testbag ) / (TN\_testbag + TP\_testbag + FN\_testbag + FP\_testbag)  
print(acc\_testbag)

## [1] 0.8054711

sen\_testbag = TP\_testbag/(TP\_testbag +FN\_testbag)  
print(sen\_testbag)

## [1] 0.6508621

spe\_testbag = TN\_testbag/(TN\_testbag + FP\_testbag)  
  
print(spe\_testbag)

## [1] 0.8896714

print('Logit Model Test')

## [1] "Logit Model Test"

print(cm\_testlogit)

## Truth  
## Prediction 0 1  
## 0 353 98  
## 1 73 134

print('Bagging Model Test')

## [1] "Bagging Model Test"

print(cm\_testbag)

## Truth  
## Prediction 0 1  
## 0 379 80  
## 1 47 152

print('TP % Increase From logit to bag = ')

## [1] "TP % Increase From logit to bag = "

print(1 - (TP\_testlogit/TP\_testbag))

## [1] 0.1125828

print('TN % Increase From logit to bag = ')

## [1] "TN % Increase From logit to bag = "

print(1 - (TN\_testlogit/TN\_testbag))

## [1] 0.06860158

print('FP % decrease From logit to bag = ')

## [1] "FP % decrease From logit to bag = "

print(1 - (FP\_testbag/FP\_testlogit))

## [1] 0.3561644

print('FN % decrease From logit to bag = ')

## [1] "FN % decrease From logit to bag = "

print(1 - (FN\_testbag/FN\_testlogit))

## [1] 0.1734694

# 5

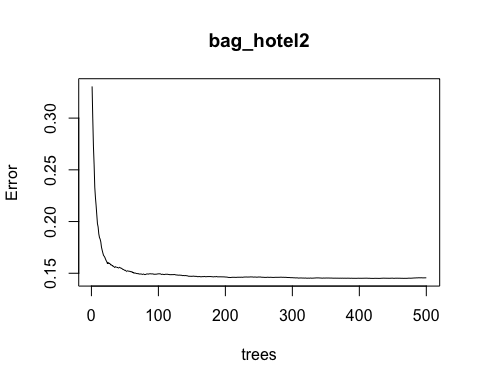
bag\_hotel2 <- randomForest(booking ~ hotel\_stars + hotel\_review\_score + hotel\_chain +   
 hotel\_location\_score + hotel\_historical\_price + search\_ranking  
 + price\_usd + promotion + length\_of\_stay + booking\_window +   
 adults\_count + children\_count + room\_count + saturday\_night  
 + comp\_rate + comp\_inv + reviewscore2 + hotel\_stars2 + price2 + booking\_window\_2, data = hotel\_train)

## Warning in randomForest.default(m, y, ...): The response has five or fewer  
## unique values. Are you sure you want to do regression?

print(bag\_hotel2)

##   
## Call:  
## randomForest(formula = booking ~ hotel\_stars + hotel\_review\_score + hotel\_chain + hotel\_location\_score + hotel\_historical\_price + search\_ranking + price\_usd + promotion + length\_of\_stay + booking\_window + adults\_count + children\_count + room\_count + saturday\_night + comp\_rate + comp\_inv + reviewscore2 + hotel\_stars2 + price2 + booking\_window\_2, data = hotel\_train)   
## Type of random forest: regression  
## Number of trees: 500  
## No. of variables tried at each split: 6  
##   
## Mean of squared residuals: 0.1455956  
## % Var explained: 37.24

plot(bag\_hotel2)



scores\_trainbag2 <- predict(bag\_hotel2, typ = "response")  
scores\_testbag2 <- predict(bag\_hotel2, type = "response", newdata = hotel\_test)  
  
predicted\_trainbag2<- ifelse(scores\_trainbag>0.5,"1","0")  
head(predicted\_trainbag2)

## 1 2 3 4 5 6   
## "0" "0" "0" "0" "1" "1"

predicted\_testbag2 <- ifelse(scores\_testbag>0.5,"1","0")  
head(predicted\_testbag2)

## 1 2 3 4 5 6   
## "0" "0" "0" "0" "0" "0"

results\_trainbag2 <- data.frame(  
 true = factor(hotel\_train$booking),  
 predicted = factor(predicted\_trainbag2),  
 score = scores\_trainbag2)  
results\_testbag2 <- data.frame(  
 true = factor(hotel\_test$booking),  
 predicted = factor(predicted\_testbag2),  
 score = scores\_testbag2)  
  
results\_testbag %>% glimpse()

## Rows: 658  
## Columns: 3  
## $ true <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, …  
## $ predicted <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, …  
## $ score <dbl> 0.23620000, 0.20953333, 0.25796667, 0.07130000, 0.17780000, …

sum(results\_trainbag2$predicted == 1)

## [1] 594

sum(results\_trainbag2$predicted == 0)

## [1] 1380

cm\_trainbag2 <- conf\_mat(results\_trainbag2,  
 truth = true,  
 estimate = predicted)  
  
cm\_testbag2 <- conf\_mat(results\_testbag2,  
 truth = true,  
 estimate = predicted)  
print(cm\_trainbag2)

## Truth  
## Prediction 0 1  
## 0 1103 277  
## 1 149 445

print(cm\_testbag2)

## Truth  
## Prediction 0 1  
## 0 379 80  
## 1 47 152

TN\_trainbag2 = 1116  
TP\_trainbag2 = 445  
FN\_trainbag2 = 277  
FP\_trainbag2 = 136  
  
TN\_testbag2 = 379  
TP\_testbag2 = 160  
FN\_testbag2 = 72  
FP\_testbag2 = 47

print('test scores forest improved model')

## [1] "test scores forest improved model"

acc\_testbag2 = (TN\_testbag2 +TP\_testbag2 ) / (TN\_testbag2 + TP\_testbag2 + FN\_testbag2 + FP\_testbag2)  
print(acc\_testbag2)

## [1] 0.8191489

sen\_testbag2 = TP\_testbag2/(TP\_testbag2 +FN\_testbag)  
print(sen\_testbag2)

## [1] 0.6639004

spe\_testbag2 = TN\_testbag2/(TN\_testbag2 + FP\_testbag2)  
  
print(spe\_testbag2)

## [1] 0.8896714