Booking predictions

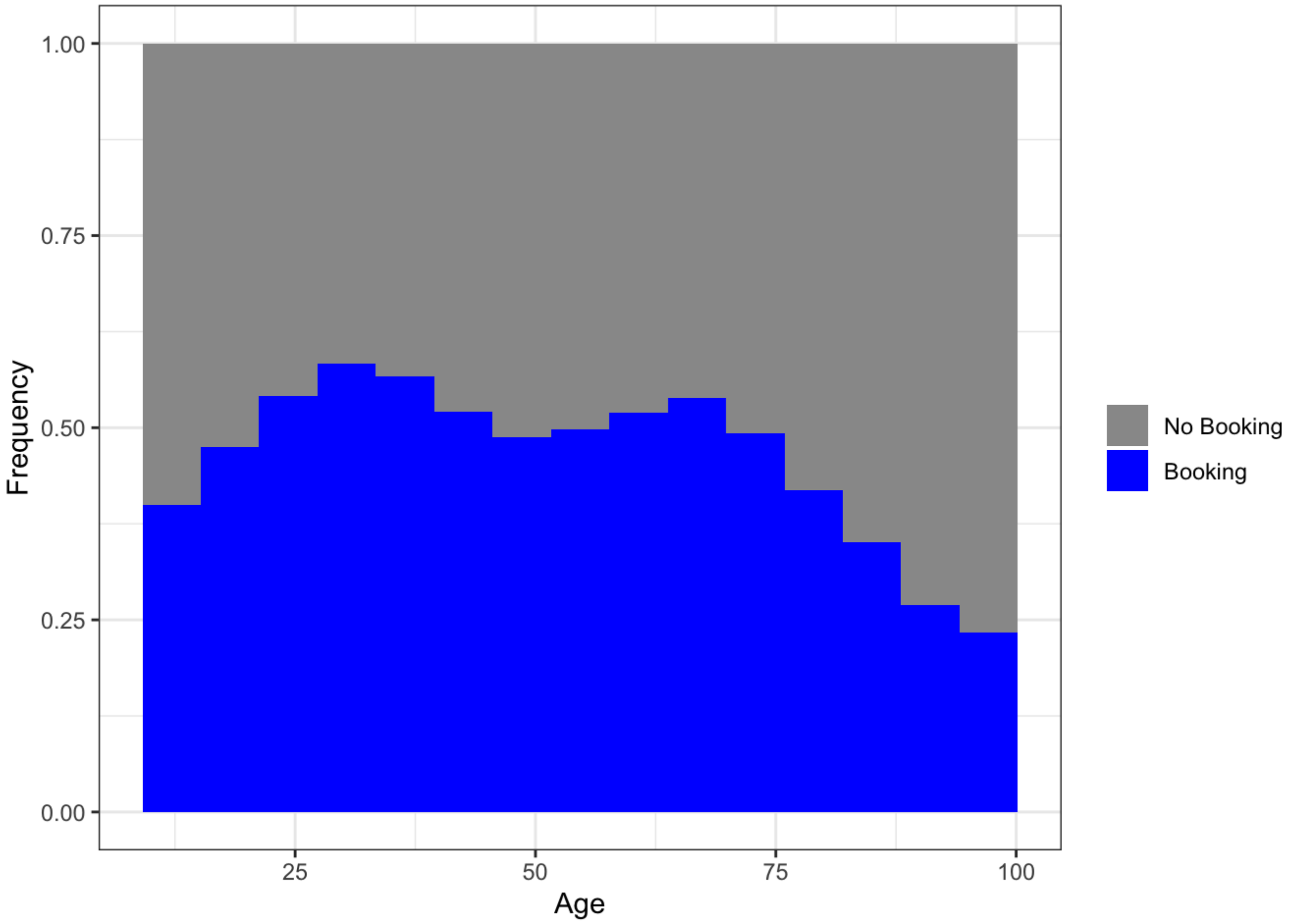
The best team

30/11/2019

Preprocessing

```r
# Read and pre-porocess the datasets
users_train <- read.csv("full_train.csv")
users_test <- read.csv("full_test.csv")

users_train$signup_flow = factor(users_train$signup_flow)
users_test$signup_flow = factor(users_test$signup_flow)

users_train$booking = factor(users_train$booking)
users_test$booking = factor(users_test$booking)

users_train_trip <- users_train %>% select(-c("id","date_first_booking", "country_destination"))
users_test_trip  = users_test %>% select(-c("id","date_first_booking", "country_destination"))
users_train_trip <- na.omit(users_train_trip)
users_test_trip <- na.omit(users_test_trip)

#this is needed to equalize the levels between training and testing sets so we can do predictions
levels(users_test_trip$gender) <- levels(users_train_trip$gender)
levels(users_test_trip$signup_method) <- levels(users_train_trip$signup_method)
levels(users_test_trip$signup_flow) <- levels(users_train_trip$signup_flow)
levels(users_test_trip$language) <- levels(users_train_trip$language)
levels(users_test_trip$affiliate_channel) <- levels(users_train_trip$affiliate_channel)
levels(users_test_trip$affiliate_provider) <- levels(users_train_trip$affiliate_provider)
levels(users_test_trip$first_affiliate_tracked) <- levels(users_train_trip$first_affiliate_tracked)
levels(users_test_trip$signup_app) <- levels(users_train_trip$signup_app)
levels(users_test_trip$first_device_type) <- levels(users_train_trip$first_device_type)
levels(users_test_trip$first_browser) <- levels(users_train_trip$first_browser)
levels(users_test_trip$age_generation) <- levels(users_train_trip$age_generation)
levels(users_test_trip$age_quantil) <- levels(users_train_trip$age_quantil)
levels(users_test_trip$millenials) <- levels(users_train_trip$millenials)
levels(users_test_trip$device_type_colapsed) <- levels(users_train_trip$device_type_colapsed)

get_stats <- function (predictions, true_data) {
  matrix = table(true_data, predictions)
  accuracy = (matrix[1,1]+matrix[2,2])/length(true_data)
  TPR = (matrix[2,2])/sum(matrix[2,])
  FPR = (matrix[1,2])/sum(matrix[1,])
  return(c(accuracy, TPR, FPR))
}

get_auc <- function(predicted_probabilities, true_data) {
  rocr.pred.rf <- prediction(predicted_probabilities, true_data)
  performance(rocr.pred.rf, "auc")@y.values[[1]]
}
```

```
library(ggplot2)
ggplot(data=users_train) +
  geom_histogram(aes(x=age,fill=(booking==1)),position='fill', bins = 15) +
  theme_bw() +
  xlab('Age') +
  ylab('Frequency') +
  scale_fill_manual (name='', labels=c('No Booking','Booking'), values=c('grey60','bl
ue'))
```



Random Forest

```
# control <- trainControl(method="repeatedcv", number=5, repeats=1, search="random")
#
# train.rf.oob <- train(booking ~ .,
#                        data = users_train_trip,
#                        method="rf",
#                        tuneGrid=data.frame(mtry=1:sqrt(ncol(users_train_trip))),
#                        nodesize = 25,
#                        ntree = 100,
#                        trControl=control)

# mtry_chosen <- train.rf.oob$bestTune[[1]]
### mtry_chosen after cv = 4
mtry_chosen <- 4

model_RF = randomForest(booking ~ . ,
                                  data = users_train_trip,
                                  ntree = 500,
                                  mtry = mtry_chosen,
                                  nodesize = 25,
                                  na.action=na.exclude)
importance.rf <- data.frame(imp=importance(model_RF))
importance.rf
```

	MeanDecreaseGini <dbl>
gender	713.09795
age	1435.98847
signup_method	2727.67816
signup_flow	638.49527
language	445.69879
affiliate_channel	799.80260
affiliate_provider	557.67241
first_affiliate_tracked	604.78668
signup_app	190.79557
first_device_type	458.29991

1-10 of 18 rows Previous **1** 2 Next

Prediction and evaluation

```
#predicting binary outcomes directly
pred_RF = predict(model_RF, newdata = users_test_trip)
```

```
#confusion matrix
table(pred_RF, users_test_trip$booking)
```

```
##
## pred_RF     0     1
##       0  7157  4725
##       1  6418 11990
```

```
get_stats(pred_RF, users_test_trip$booking)
```

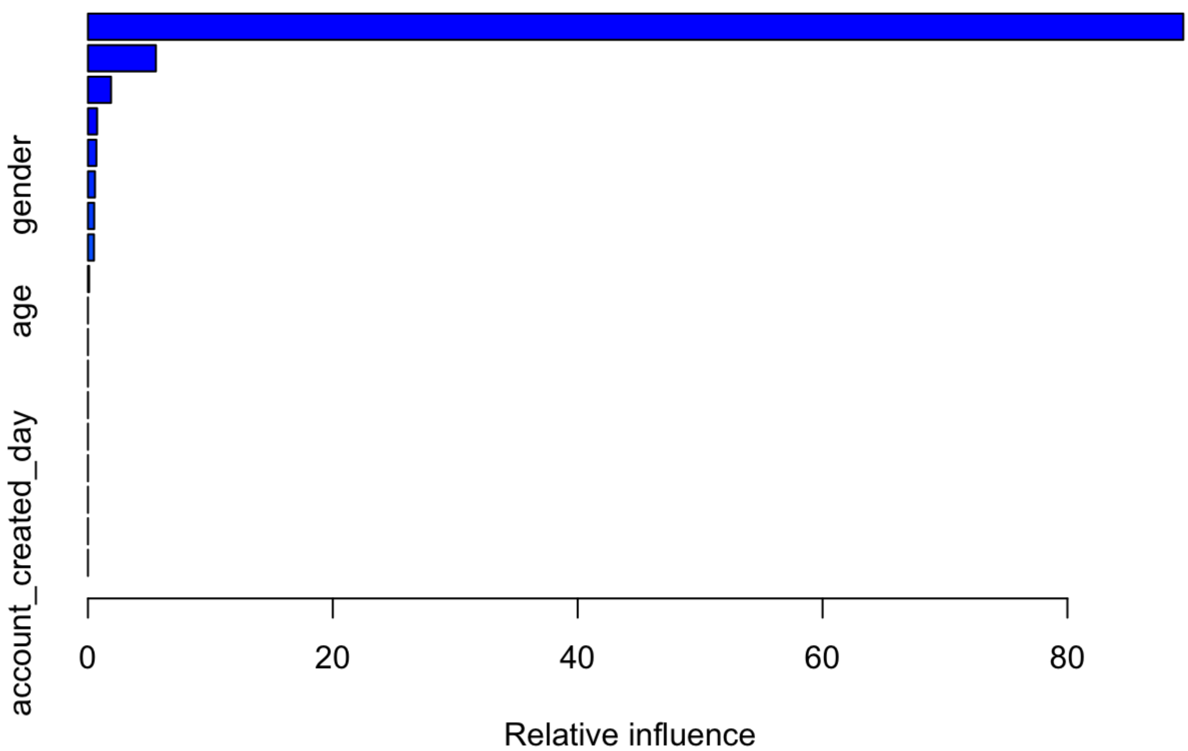
```
## [1] 0.6321228 0.7173198 0.4727808
```

```
#to get AUC, we need to predict probabilities instead
get_auc(predict(model_RF, newdata = users_test_trip, type = "prob")[, 2], users_test_
trip$booking)
```

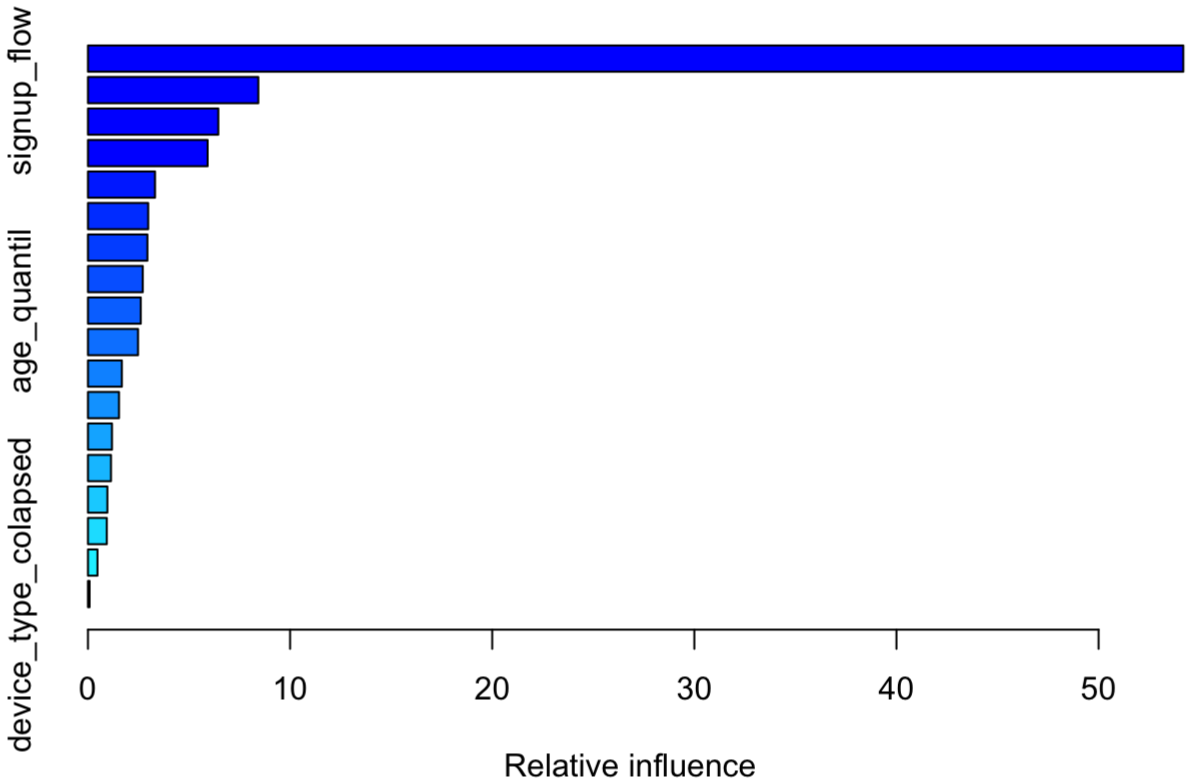
```
## [1] 0.663925
```

Boosting

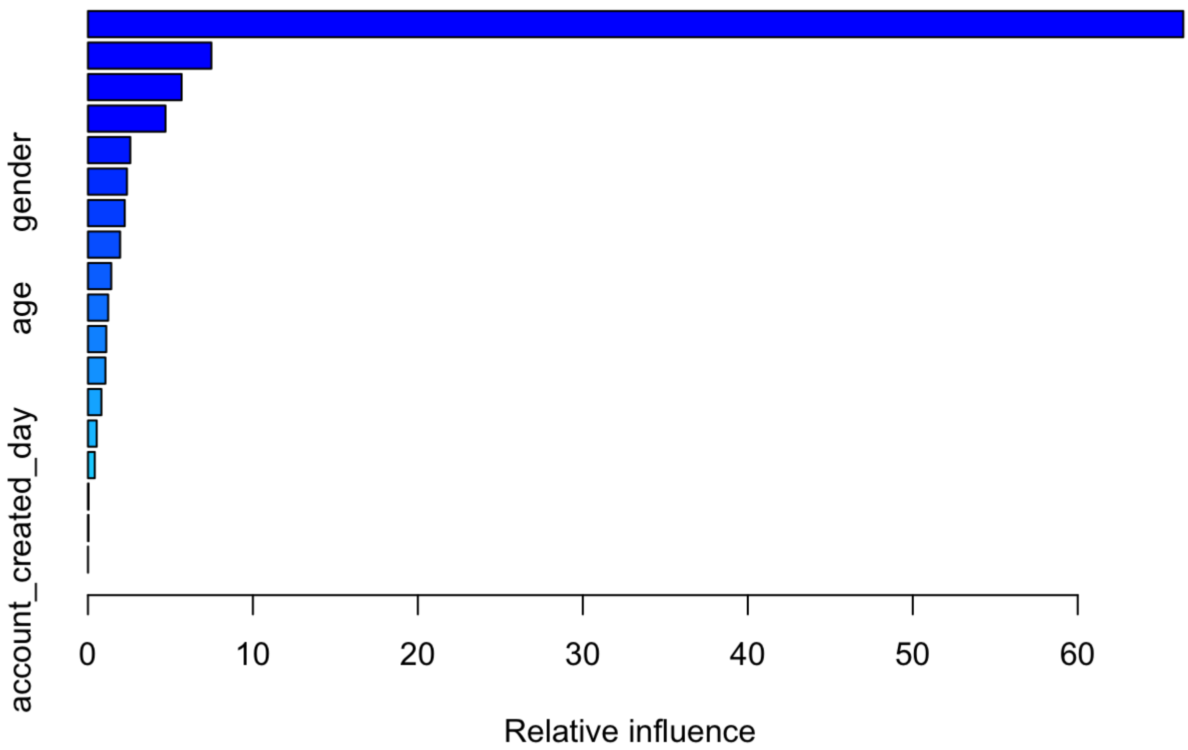
```
#simplest model
model_GBM_1 = gbm(booking ~ .,
                  data=users_train_trip, distribution = "multinomial",
                  n.minobsinnode = 50, n.trees = 100, shrinkage = 0.001, interaction.
depth = 3)
influence = summary(model_GBM_1)
```



```
#a bit more complex: more and deeper trees, but faster learning rate
model_GBM_2 = gbm(booking ~ .,
                  data=users_train_trip, distribution = "multinomial",
                  n.minobsinnode = 25, n.trees = 500, shrinkage = 0.01, interaction.d
epth = 7)
influence = summary(model_GBM_2)
```




```
#the most complex model: small learning rate, lots of deep trees
model_GBM_3 = gbm(booking ~ .,
                  data=users_train_trip, distribution = "multinomial",
                  n.minobsinnode = 25, n.trees = 1000, shrinkage = 0.001, interactio
n.depth = 10)
influence = summary(model_GBM_3)
```



Prediction and evaluation

```
pred_GBM_1_prob <-  predict(model_GBM_1, newdata = users_test_trip, n.trees = 100, ty
pe = "response")
pred_GBM_1 <- as.numeric(pred_GBM_1_prob[, 2, 1] <= pred_GBM_1_prob[, 1, 1])


pred_GBM_2_prob <-  predict(model_GBM_2, newdata = users_test_trip, n.trees = 500, ty
pe = "response")
pred_GBM_2 <- as.numeric(pred_GBM_2_prob[, 2, 1] <= pred_GBM_2_prob[, 1, 1])


pred_GBM_3_prob <-  predict(model_GBM_3, newdata = users_test_trip, n.trees = 1000, t
ype = "response")
pred_GBM_3 <- as.numeric(pred_GBM_3_prob[, 2, 1] <= pred_GBM_3_prob[, 1, 1])
```

```
get_stats(pred_GBM_1, users_test_trip$booking)
```

```
## [1] 0.3662265 0.3434640 0.6057459
```

```
get_auc(pred_GBM_1_prob[, 2, ], users_test_trip$booking)
```

```
## [1] 0.6632239
```

```
get_stats(pred_GBM_2, users_test_trip$booking)
```

```
## [1] 0.4074282 0.5504637 0.7686924
```

```
get_auc(pred_GBM_2_prob[, 2, ], users_test_trip$booking)
```

```
## [1] 0.6724686
```

```
get_stats(pred_GBM_3, users_test_trip$booking)
```

```
## [1] 0.3659624 0.3539336 0.6192265
```

```
get_auc(pred_GBM_3_prob[, 2, ], users_test_trip$booking)
```

```
## [1] 0.6713134
```