CustomerAcquisitionRetention

Visha Arumugam, Michael Grogan,Sanyogita Apte

11/12/2021

I - Executive Summary

The analysis of the data from the Acquisition/Retention set from the Customer Relationship Management data collection shows that a simple linear and logistic model performs better than decision trees or random forest models for predicting the duration of a customer's time with the firm, and predicting the acquisition of a new customer is similar across the different types of models that were tested in this analysis. A firm with access to this kind of information about their customers will be able to use these predictions to better optimize their spending on acquisition and retention in order to maximize profit from customers they know they can keep and avoiding wasting money on those customers they will not be able to keep.

II - The Problem

Firms need to optimally allocate their resources in pursuing new clients as well as keeping their profitable current clients for as long as they can. The balancing act can be made easier by making predictions for which clients are likely to be acquired/retained given their associated data and spending money on those clients. We will analyze the data that the firm does have available in order to determine the clients that should receive the focus.

III - Review of Related Literature

For any business, customers are the basis for its success and revenue and that is why companies become more aware of the importance of gaining customers' satisfaction. Customer relationship management (CRM) supports marketing by selecting target consumers and creating cost-effective relationships with them. CRM is the process of understanding customer behavior in order to support organization to improve customer acquisition, retention, and profitability.

There were many predictive analysis happened in CRM for customer retention using various dataset and based upon the usage of various tools and technologies,various exploration and various prediction techniques, different analysis came with different conclusion and Recommendation.

As per the Journal "Machine-Learning Techniques for Customer Retention" by Sahar F. Sabbeh using telecom industry Data set, executed various prediction analysis to find out the Best customer retention rate. Based on the results and findings ensemble methods such as both Random forest and Ad boost models gave the best accuracy.

There were many CRM Analysis happened using various industry data set. In most of the Analysis , ensemble methods provides the best accuracy with quite satisfying prediction rate.

IV - Methodology

We are going to use few prediction algorithm techniques to which customers will be acquired and for how long (duration) based on a feature set. The prediction algorithms are as follows Linear Regression, Logistic Decision tree and Random Forest.

Linear Regression: Linear regression attempts to model the relationship between dependent and independent variables by fitting a linear equation to observed data. The most common method for fitting a regression line is the method of least-squares. This method calculates the best-fitting line for the observed data by minimizing the sum of the squares of the vertical deviations from each data point to the line.

Logistic Regression: Logistic Regression is a parametric classification method in which is used to model the probability of a certain class or event existing based upon the independent variables.In Logistic Regression, we don't directly fit a straight line to our data like in linear regression. Instead, we fit a S shaped curve, called Sigmoid, to our observations.

Decision Tree Decision Tree (DT) is a model that generates a tree-like structure that represents set of decisions. DT returns the probability scores of class membership. DT is composed of: a) internal Nodes: each node refers to a single variable/feature and represents a test point at feature level; b) branches, which represent the outcome of the test and are represented by lines that finally lead to c) leaf Nodes which represent the class labels. That is how decision rules are established and used to classify new instances. DT is a flexible model that supports both categorical and continuous data. Due to their flexibility they gained popularity and became one of the most commonly used models for churn prediction.

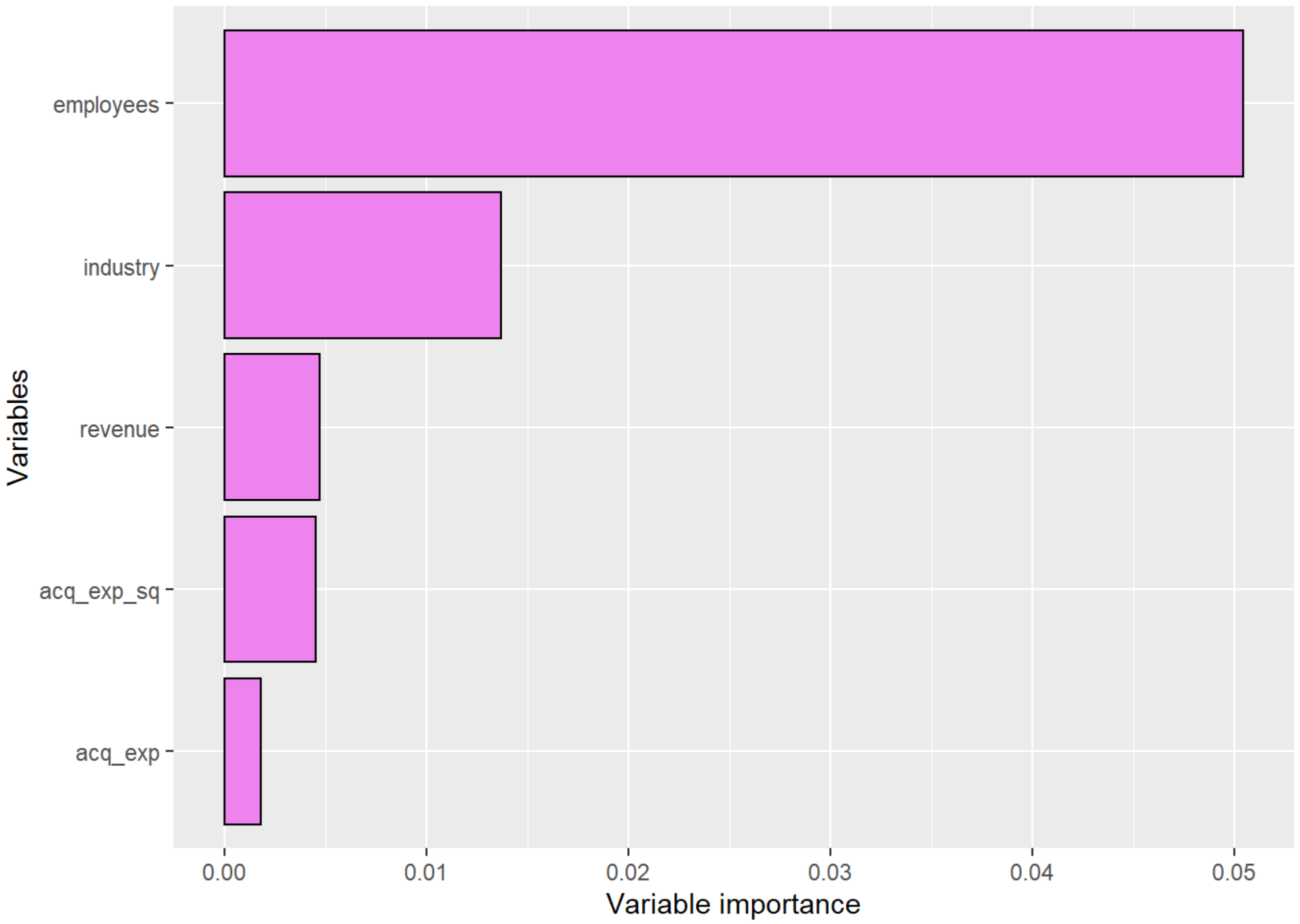
Random Forest Random forests (RF) are an ensemble learning technique that can support classification and regression. It extends the basic idea of single classification tree by growing many classification trees in the training phase. To classify an instance, each tree in the forest generates its response (vote for a class), the model choses the class that has receive the most votes over all the trees in the forest. One major advantage of RF over traditional decision trees is the protection against overfitting which makes the model able to deliver a high performance

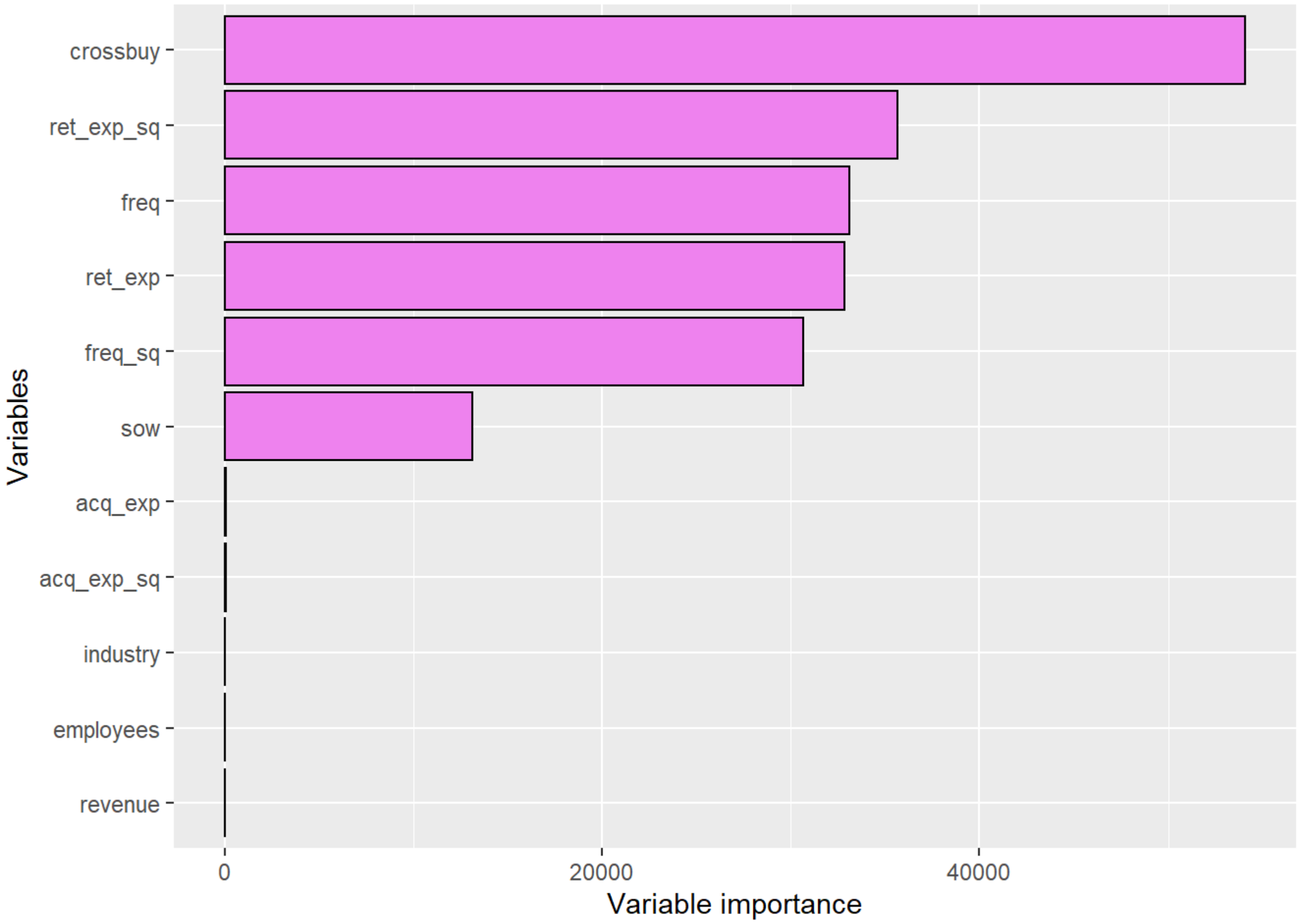
V - Data

The values available to a firm trying to determine which clients they are likely to acquire would not include values such as purchase frequency or duration of partnership, because these data would be unknown when trying to predict the potential of a client to become an acquisition. Therefore the only variables available for predicting acquisition are expenditure towards acquisition, whether the potential client is in the industry, the revenue of the client, and the number of employees of the client.

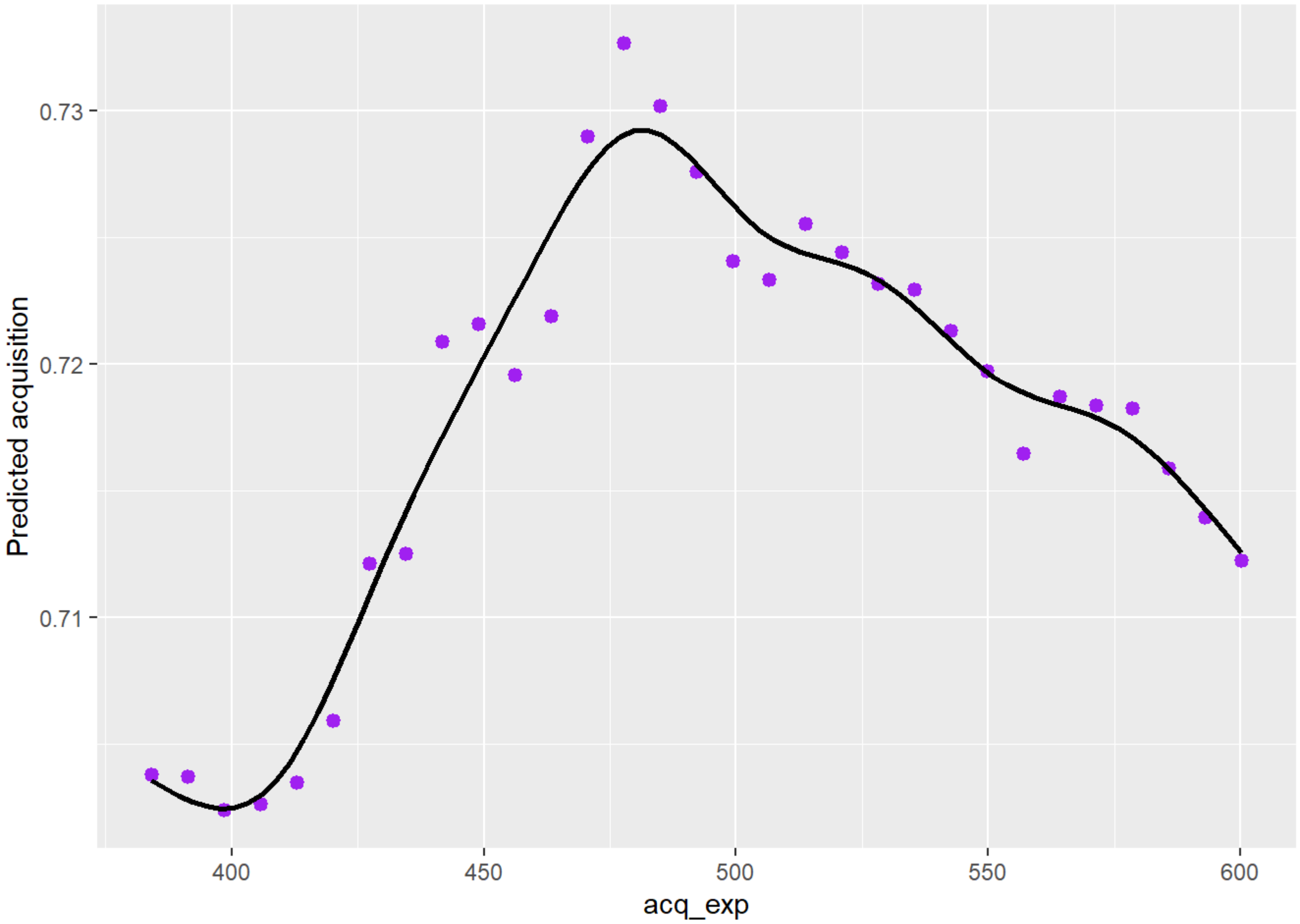
The variables used in the duration model are those tracking expenditure towards retention, frequency of purchase, share of wallet (amount of budget spent with the firm) and crossbuy (the number of production categories the client is purchasing from the firm). Also included for the linear model of the duration is the inverse mills ratio from the acquisition model to account for sample selection bias that is present due to the fact that the only customers that have frequency and crossbuy etc. to measure are those clients that chose to partner with the firm in the first place.

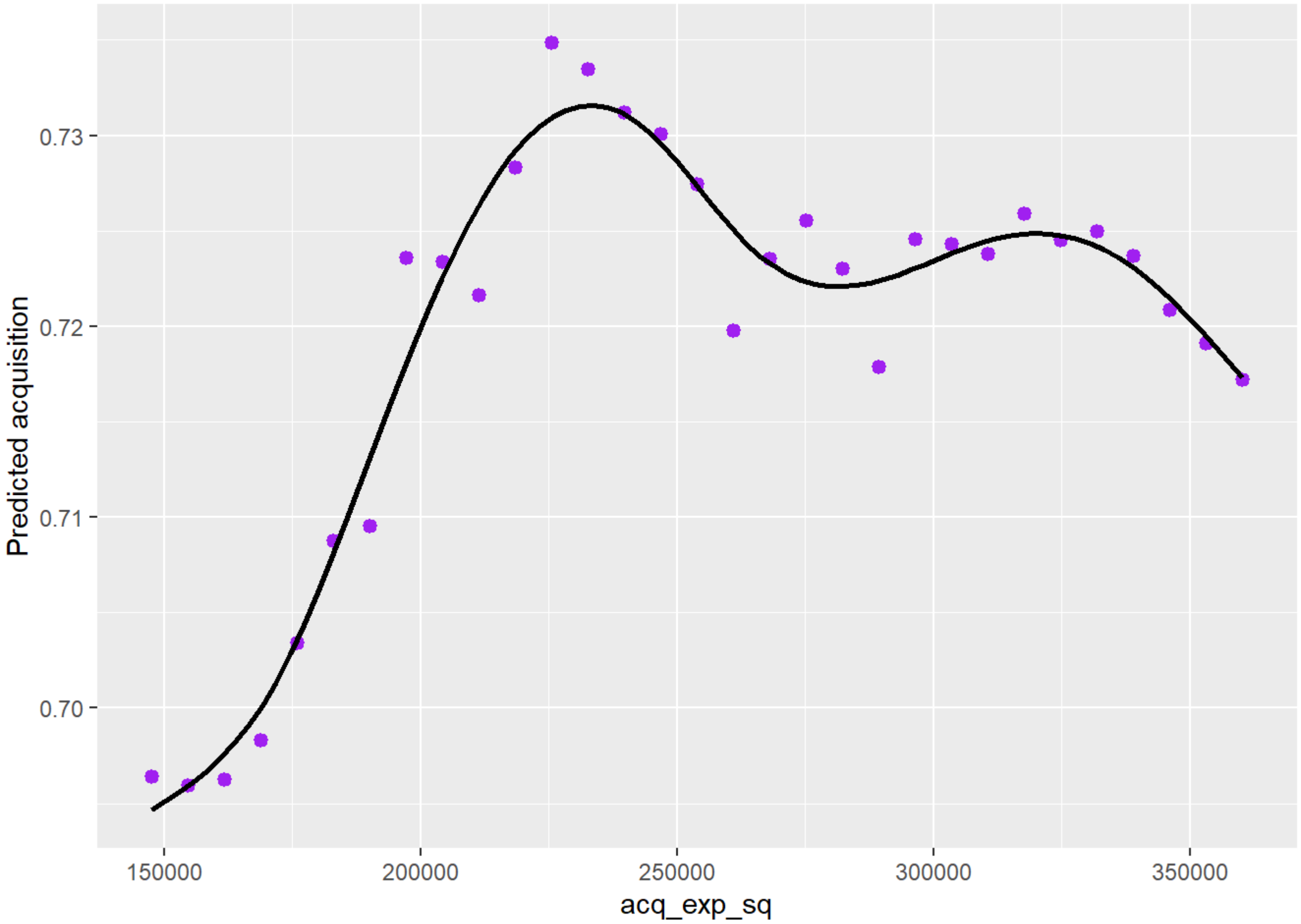
We see that when measuring the variable importance in the random forest model that includes both the variables used in the acquisition model and the duration model, the variables from the acquisition model are not important as the variables mentioned for duration.

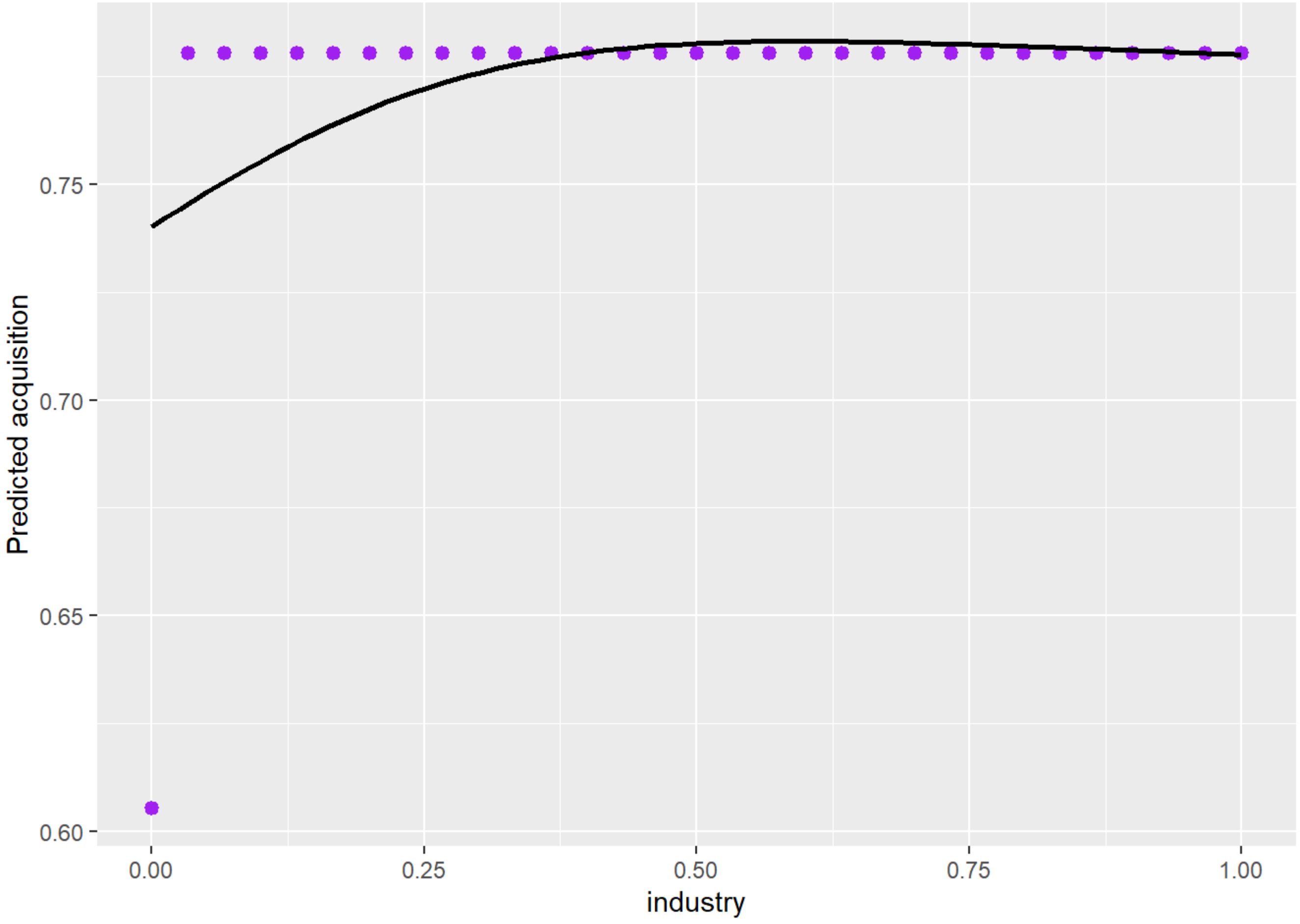

employees
industry
revenue
acq_exp_sq
acq_exp
Variables
0.00
0.01
0.02
0.03
0.04
0.05
Variable importance

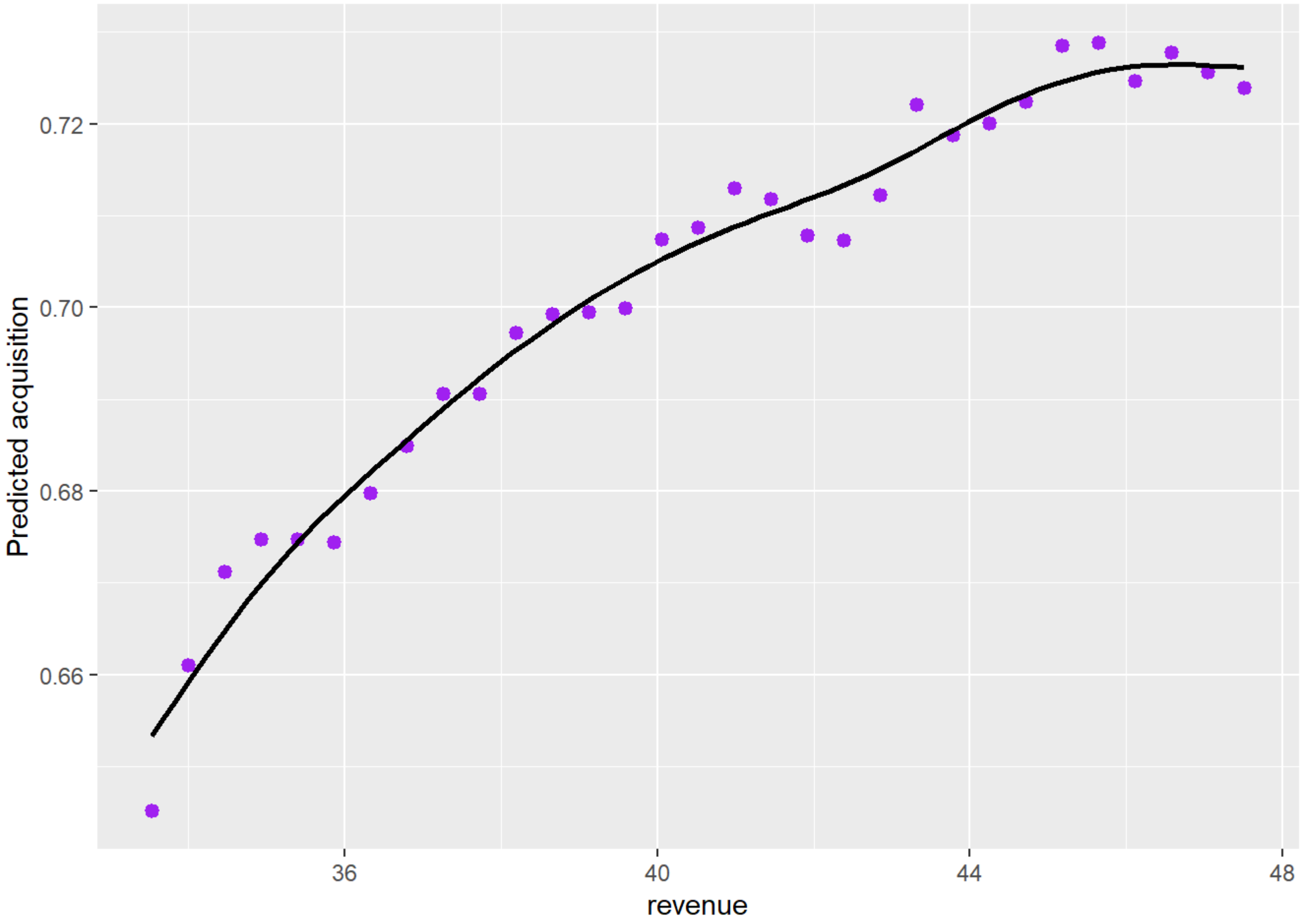

crossbuy
ret_exp_sq
freq
ret_exp
freq_sq
sow
acq_exp
acq_exp_sq
industry
employees
revenue
Variables
0
20000
40000
Variable importance

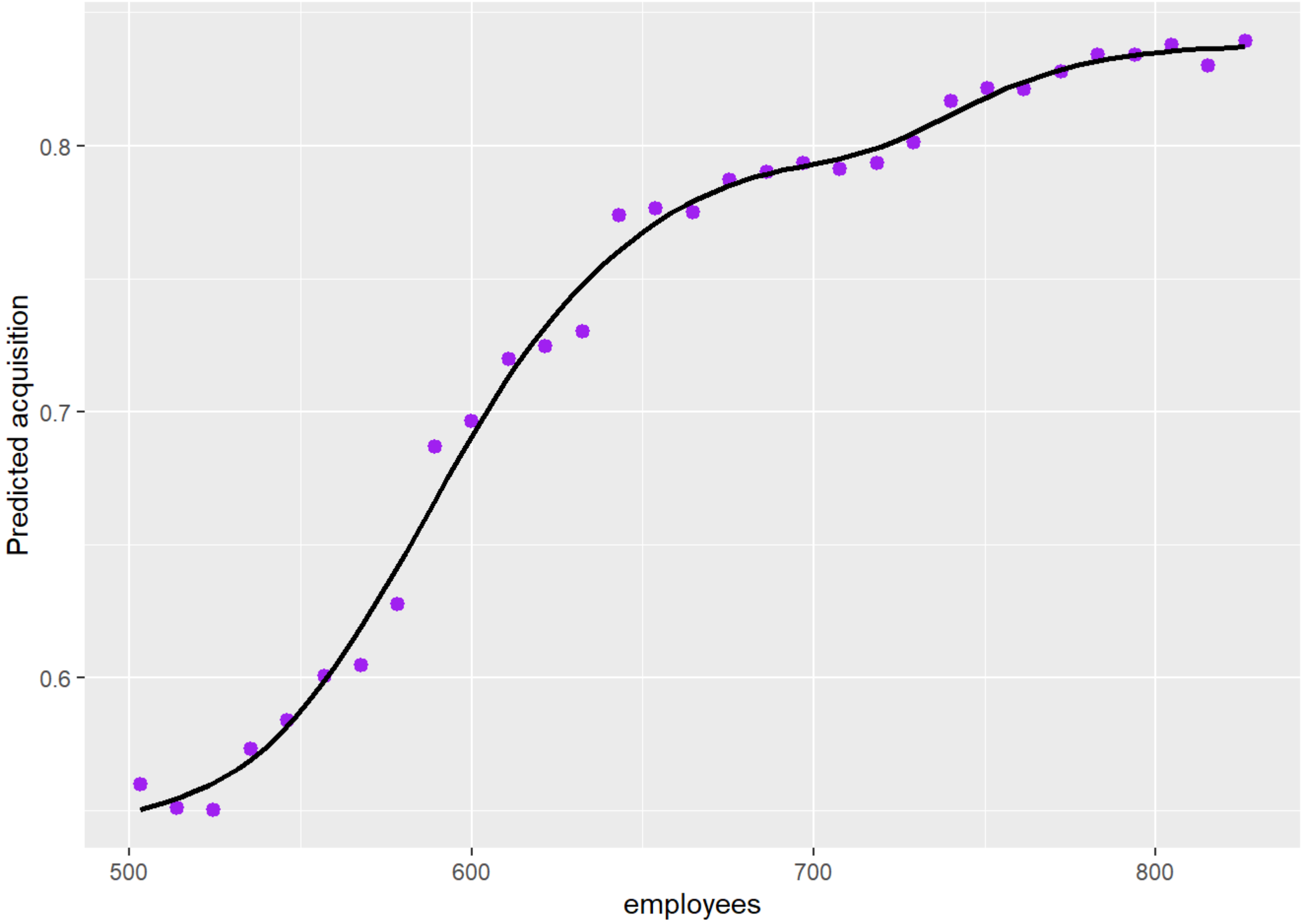
We can also see the effect each of these variables has on the target variables of acquisition and duration by plotting partial dependence of these variables. The following charts show the effect on acquisition.



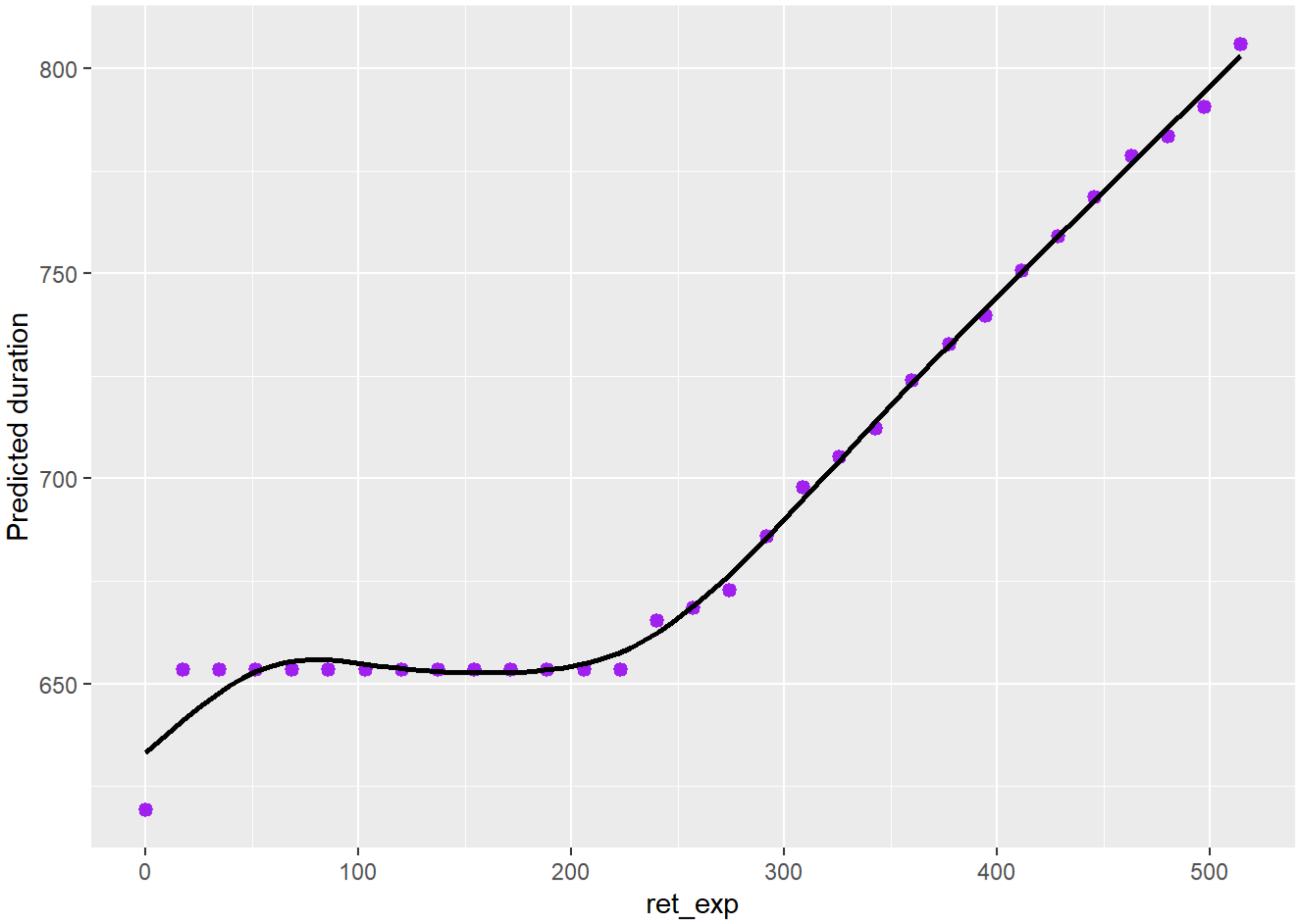

Predicted acquisition
0.70
0.71
0.72
0.73
150000
200000
250000
300000
350000
acq_exp_sq

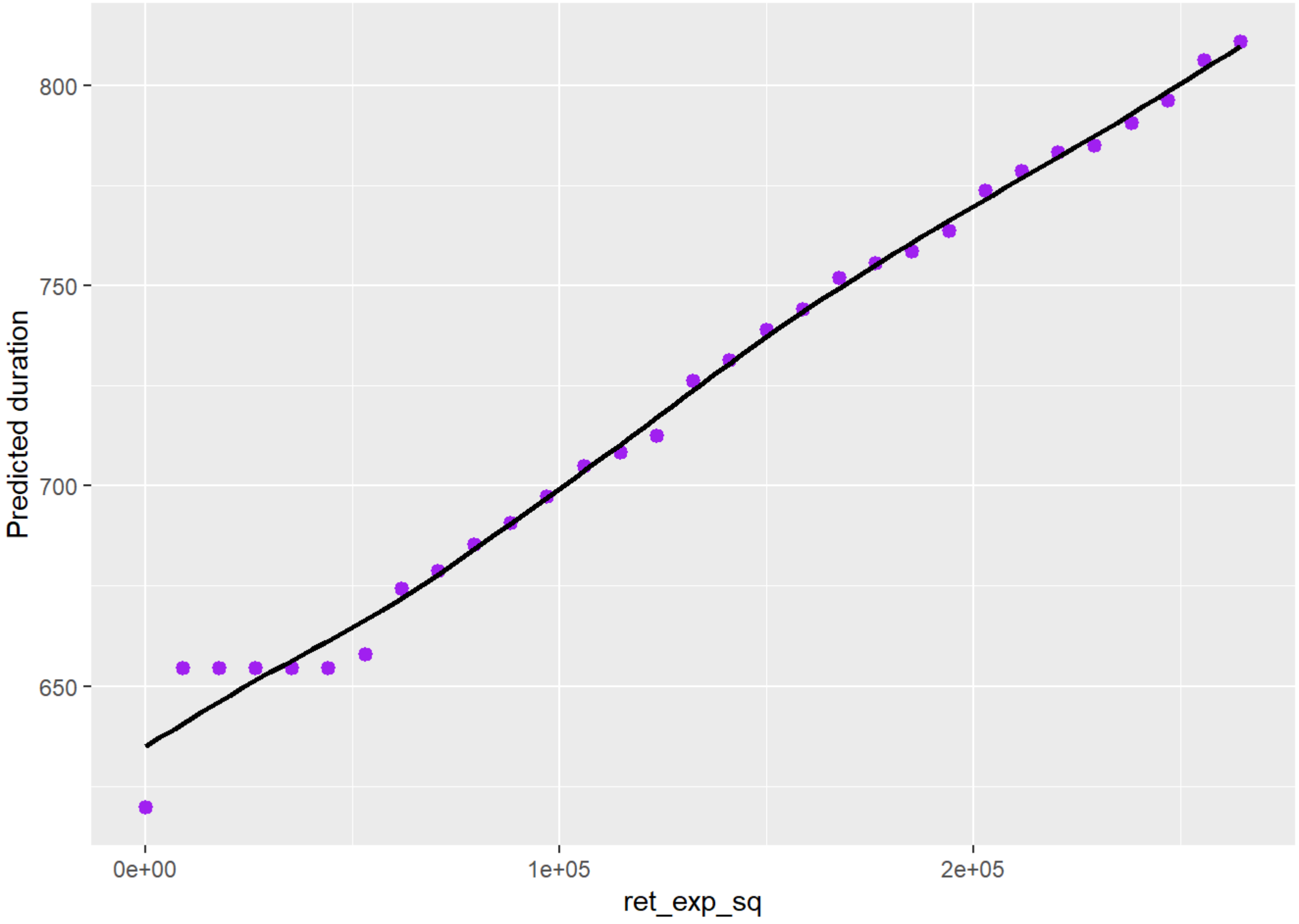

Predicted acquisition
0.60
0.65
0.70
0.75
0.00
0.25
0.50
0.75
1.00
industry

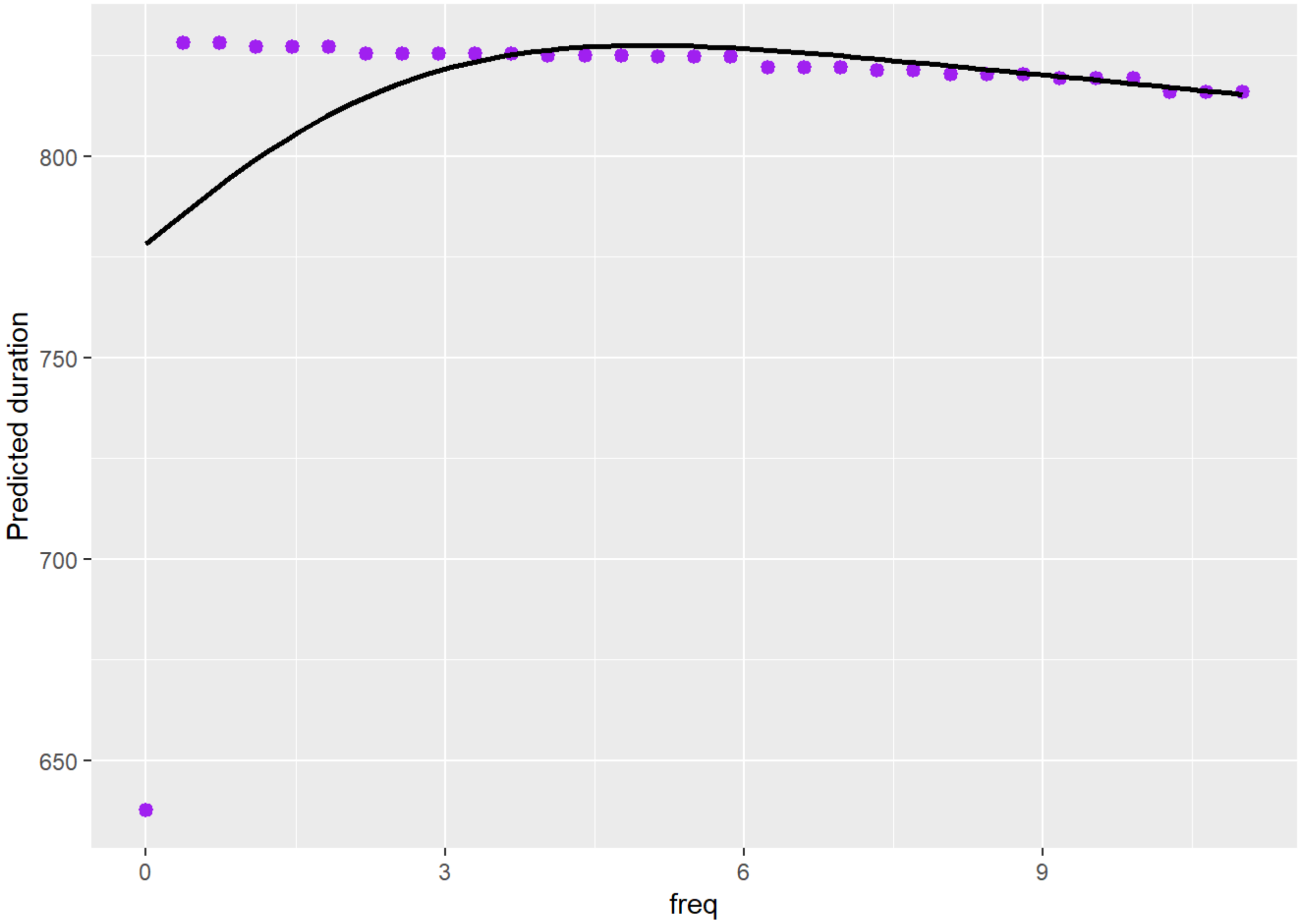

Predicted acquisition
0.72
0.70
0.68
0.66
36
40
44
48
revenue

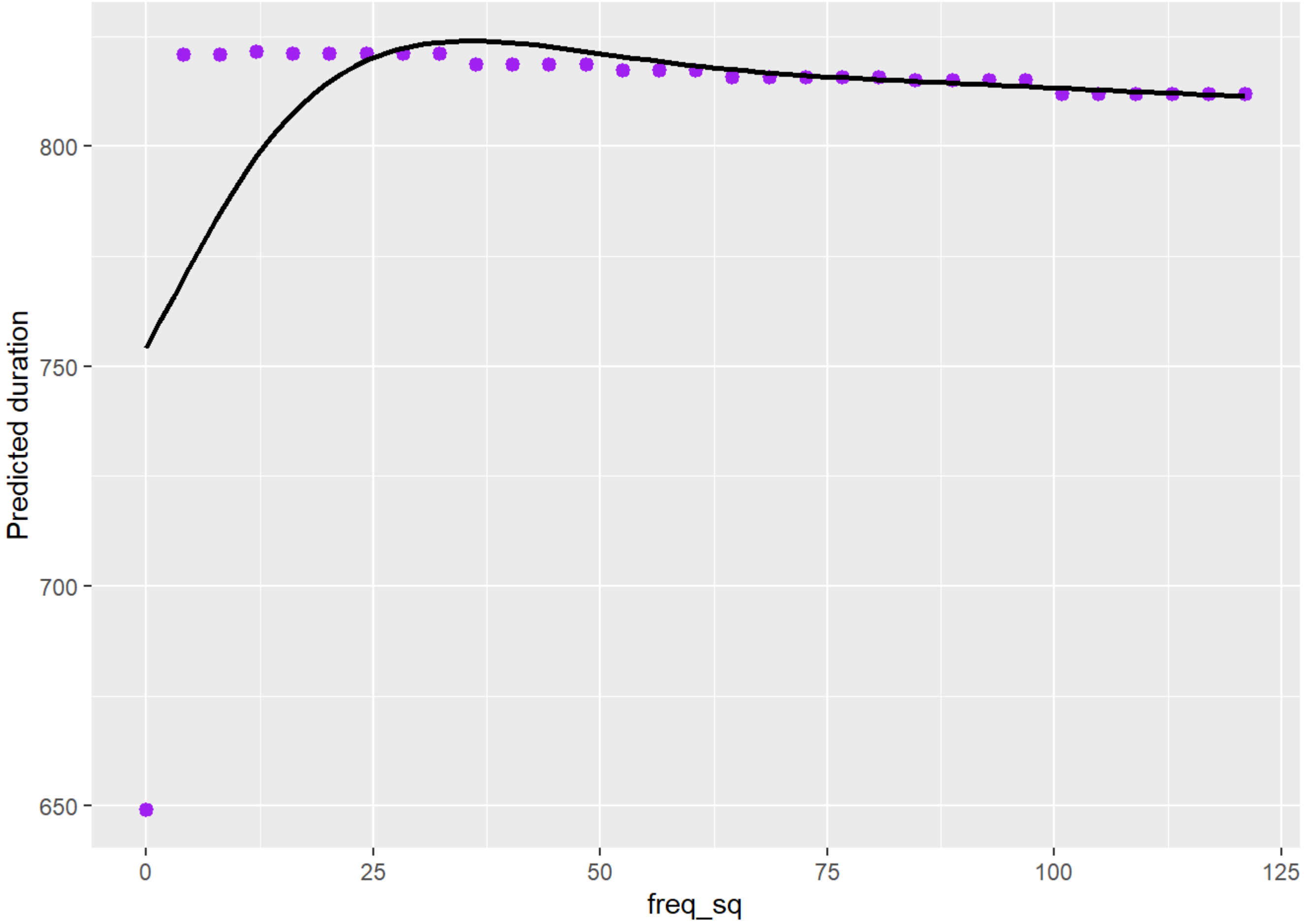

Predicted acquisition
0.8
0.7
0.6
500
600
700
800
employees

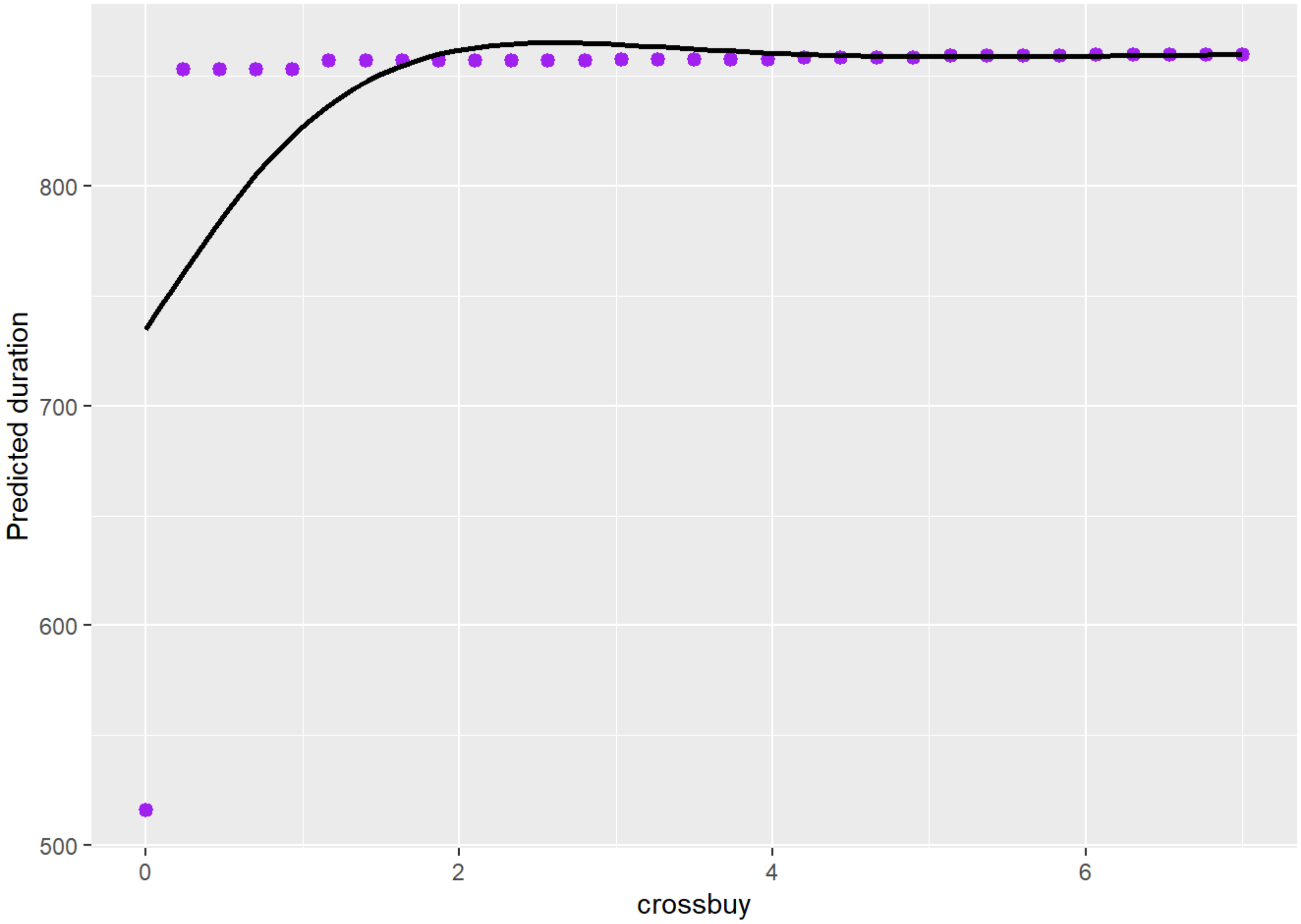
The following charts show the effect on duration

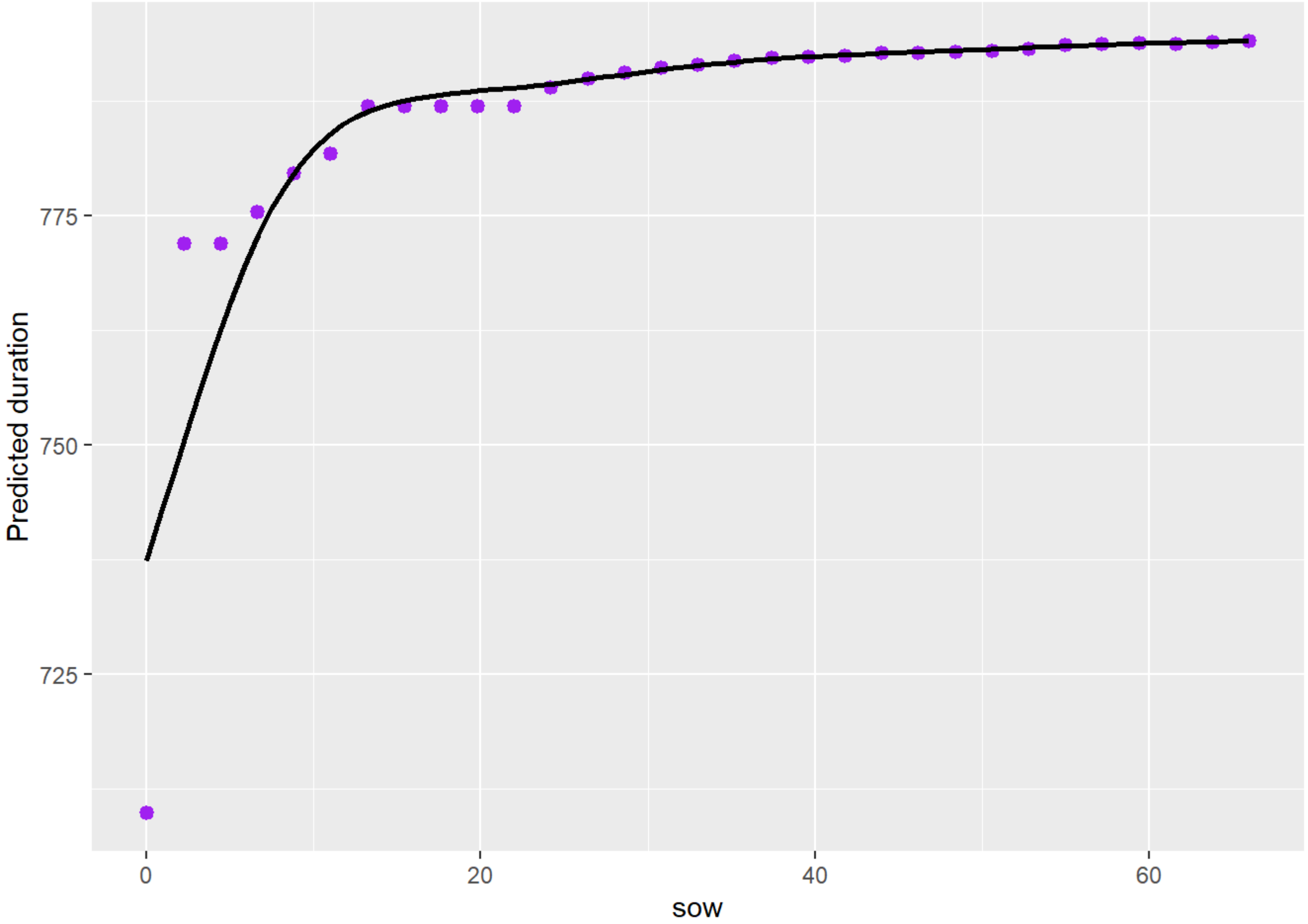





Predicted duration
800
750
700
650
0
3
6
9
freq


Predicted duration
800
750
700
650
0
25
50
75
100
125
freq_sq


Predicted duration
800
700
600
500
0
2
4
6
crossbuy


Predicted duration
775
750
725
0
20
40
60
sow

VI - Findings

We are able to determine from the table below the varying degrees of performance of the different models. The table shows the different models used, and how those models performed in accuracy for predicting whether a company would be acquired as a customer, and how long of a duration that company would remain as a customer, expressed as root mean squared error of predictions vs actuals. These statistics are given for both the training set and the test set, which was divided by 70 to 30 percent of the data. For labeling purposes, the models trained using the general linear model are labeled as linear models, but for the models focused on determining likelihood of acquisition, logistic regression was used and then the results were used in the duration model which used a typical gaussian linear regression.

```
##                   model trainaccuracies trainRMSE testaccuracies testRMSE
## 1   Tuned Random Forest       0.8742857  16.66613      0.8333333 24.49070
## 2 Untuned Random Forest       0.9314286  20.95641      0.8400000 26.88413
## 3           Full Linear       0.8000000  17.43563      0.8466667 16.18909
## 4          Linear Model       0.8000000  18.53329      0.8466667 17.35105
## 5         Decision Tree       0.8257143  67.61857      0.7400000 64.14468
```

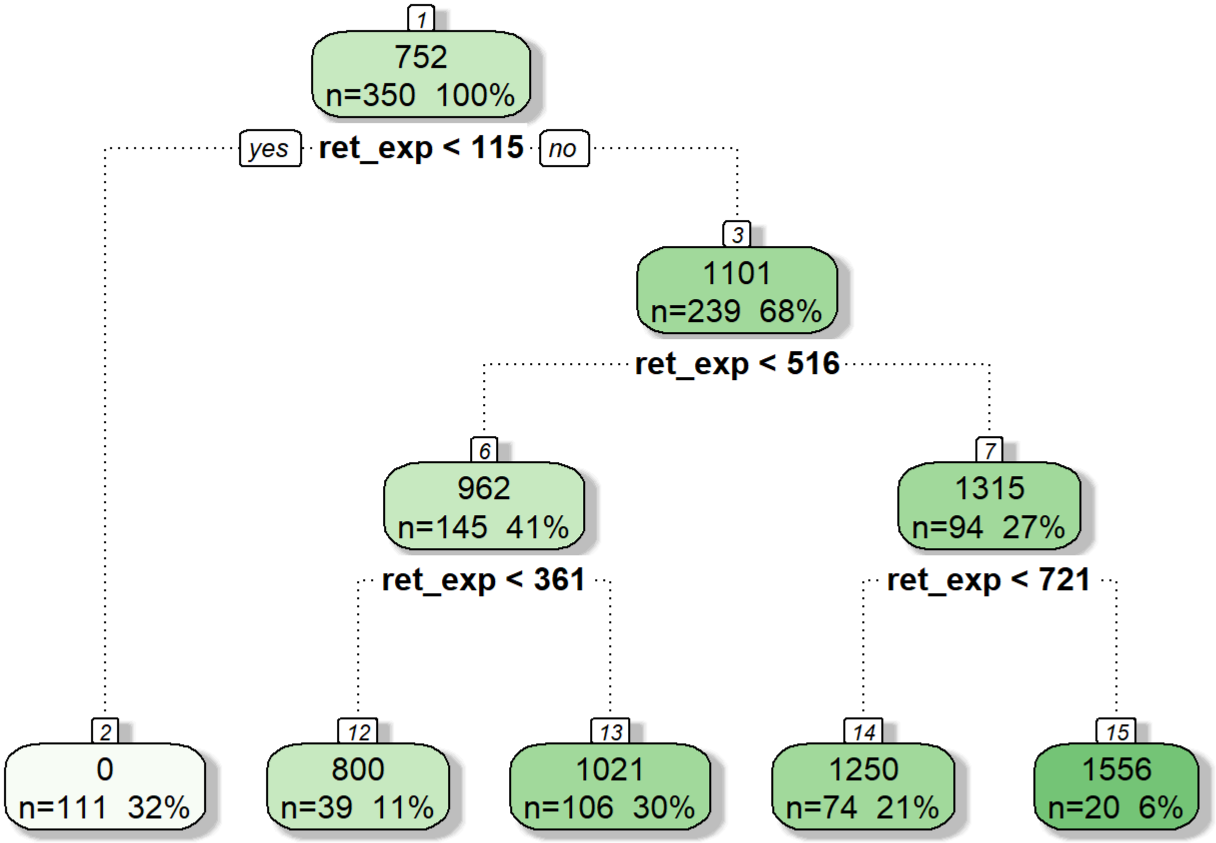
Looking first at the results for the accuracy in determining the acquisition of a company as a customer, all of the models perform similarly with accuracy between 81 and 83 percent. Not only is the accuracy very similar for the tuned random forest model as well as the linear model with only the duration-relevant variables, the confusion matrices below show that the distribution of predictions is also very similar.

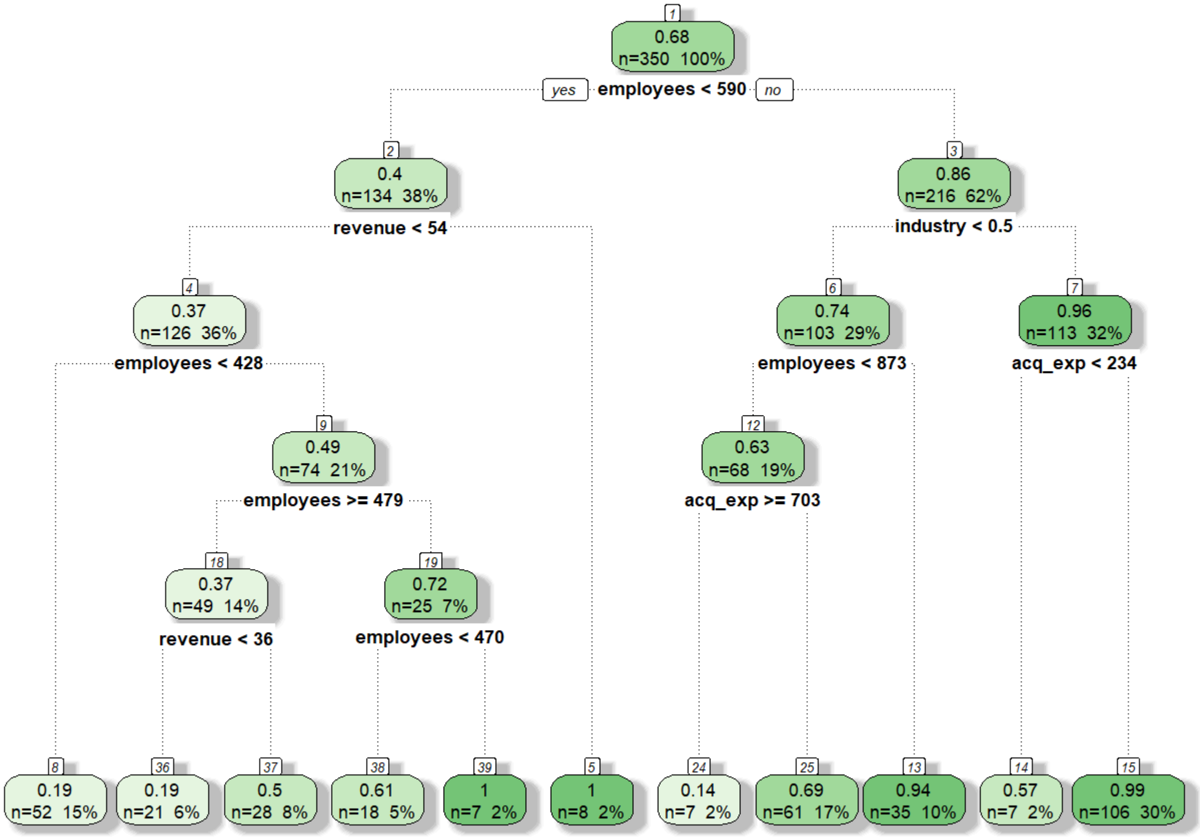
```
##    Actuals
## LM   0  1
##   0 33  5
##   1 18 94
```

```
##    Actuals
## RF   0  1
##   0 31  5
##   1 20 94
```

To confirm the implication of the chart above showing variable importances for the duration random forest including variables from the acquisition model, the inclusion of those variables in the linear model (Full Linear) does not yield a large reduction in root mean squared error of predicting duration of partnership.

With that said, the linear models are very slightly outperformed by the tuned random forest in terms of RMSE for duration values in the training data. However, when applied to the testing data, the linear model outperforms the rest of the models, including the random forest models. This swing from the training to testing data indicates that there is some degree of over-fitting of the random forest model, although the model with the tuned hyper parameters still outperforms the untuned version, and far outperforms the decision tree, both of which are shown below.


1
752
n=350 100%
yes
ret_exp < 115
no
3
1101
n=239 68%
ret_exp < 516
6
962
n=145 41%
ret_exp < 361
7
1315
n=94 27%
ret_exp < 721
2
0
n=111 32%
12
800
n=39 11%
13
1021
n=106 30%
14
1250
n=74 21%
15
1556
n=20 6%


1
0.68
n=350 100%
yes
employees < 590
no
2
0.4
n=134 38%
revenue < 54
3
0.86
n=216 62%
industry < 0.5
4
0.37
n=126 36%
employees < 428
6
0.74
n=103 29%
employees < 873
7
0.96
n=113 32%
acq_exp < 234
9
0.49
n=74 21%
employees >= 479
12
0.63
n=68 19%
acq_exp >= 703
18
0.37
n=49 14%
revenue < 36
19
0.72
n=25 7%
employees < 470
8
0.19
n=52 15%
36
0.19
n=21 6%
37
0.5
n=28 8%
38
0.61
n=18 5%
39
1
n=7 2%
5
1
n=8 2%
24
0.14
n=7 2%
25
0.69
n=61 17%
13
0.94
n=35 10%
14
0.57
n=7 2%
15
0.99
n=106 30%

VII - Conclusion

Given the fact that the models have a similar predictive accuracy for customer acquisition, the linear models perform the best in duration prediction, and the simpler linear model performs similarly to the model including acquisition variables, we recommend the linear model with only the duration variables as the most useful to the firm for predicting first the likelihood of a potential acquisition becoming a customer, and then the duration of that customer partnership with the firm. The next step for the firm is to optimize the amount of money spent on acquisition and retention in order to maximize the profit over the lifetime of these customers.

Appendix

```
acq_vars<-c("acq_exp","acq_exp_sq","industry","revenue","employees")
acq_target<-"acquisition"

acq_formula<-as.formula(paste(acq_target,paste(acq_vars,collapse="+"),sep="~"))


dur_vars<-c("ret_exp","ret_exp_sq","freq","freq_sq","crossbuy","sow","IMR")


dur_target<-"duration"

dur_formula<-as.formula(paste(dur_target,paste(dur_vars,collapse="+"),sep="~"))



dur_full<-c("ret_exp","ret_exp_sq","freq","freq_sq","crossbuy","sow","acq_exp","acq_exp_sq","industry","revenue","employees")
dur_full_formula<-as.formula(paste(dur_target,paste(dur_full,collapse="+"),sep="~"))
untuned_full_dur<-rfsrc(dur_full_formula,data = AR,importance = TRUE)
```

```
#Logistic Regression
###Logistic Acquisition prediction
acq_lm<-glm(acq_formula,train,family=binomial(link="probit"))


acq_lm_train_pred<-ifelse(predict(acq_lm,train,type="response")>0.5,1,0)

#Confusion Matrix
#table(acq_lm_train_pred,train$acquisition)
acq_lm_train_acc<-sum(acq_lm_train_pred==train$acquisition)/length(train$acquisition)



acq_lm_test_pred<-ifelse(predict(acq_lm,test,type="response")>0.5,1,0)

#Confusion Matrix
#table(acq_lm_test_pred,test$acquisition)
acq_lm_test_acc<-sum(acq_lm_test_pred==test$acquisition)/length(test$acquisition)



####Logistic Duration Prediction
train$IMR<-invMillsRatio(glm(acq_formula,train,family=binomial(link="probit")))$IMR1

dur_lm<-glm(dur_formula,train,family=gaussian)

dur_lm_train_RMSE<-sqrt(mean(dur_lm$residuals^2))


test$IMR<-invMillsRatio(glm(acq_formula,test,family=binomial(link="probit")))$IMR1

dur_pred<-predict(dur_lm,test)
dur_lm_test_RMSE<-sqrt(mean((dur_pred-test$duration)^2))



####Logistic Duration Prediction All variables

dur_flm<-glm(dur_full_formula,train,family=gaussian)

dur_flm_train_RMSE<-sqrt(mean(dur_flm$residuals^2))


dur_pred<-predict(dur_flm,test)
dur_flm_test_RMSE<-sqrt(mean((dur_pred-test$duration)^2))
```

```r
#Single Decision Tree
acq_dt<-rpart(acq_formula,train)

dur_dt<-rpart(dur_formula,train)

acq_dt_train_pred<-ifelse(predict(acq_dt,train)>0.5,1,0)
acq_dt_train_acc<-sum(acq_dt_train_pred==train$acquisition)/length(train$acquisition)
#Confusion Matrix
#table(acq_dt_train_pred,train$acquisition)


acq_dt_test_pred<-ifelse(predict(acq_dt,test)>0.5,1,0)
acq_dt_test_acc<-sum(acq_dt_test_pred==test$acquisition)/length(test$acquisition)
#Confusion Matrix
#table(acq_dt_test_pred,test$acquisition)


dur_dt_train_pred<-predict(dur_dt,train)
dur_dt_train_RMSE<-sqrt(mean((dur_dt_train_pred-train$duration)^2))


dur_dt_test_pred<-predict(dur_dt,test)
dur_dt_test_RMSE<-sqrt(mean((dur_dt_test_pred-test$duration)^2))
```

```r
#Un-tuned Random Forest
untuned_acq <- rfsrc(acq_formula,data = train,importance = TRUE)

acq_untunedrf_train_pred<-ifelse(predict(untuned_acq,train)$predicted>0.5,1,0)
acq_untunedrf_train_acc<-sum(acq_untunedrf_train_pred==train$acquisition)/length(train$acquisition)
#Confusion Matrix
#table(acq_untunedrf_train_pred,train$acquisition)


acq_untunedrf_test_pred<-ifelse(predict(untuned_acq,test)$predicted>0.5,1,0)
acq_untunedrf_test_acc<-sum(acq_untunedrf_test_pred==test$acquisition)/length(test$acquisition)

#Confusion Matrix
#table(acq_untunedrf_test_pred,test$acquisition)

untuned_dur <- rfsrc(dur_formula,data = train,importance = TRUE)


dur_untunedrf_train_pred<-predict(untuned_dur,train)$predicted
dur_untunedrf_train_RMSE<-sqrt(mean((dur_untunedrf_train_pred-train$duration)^2))


dur_untunedrf_test_pred<-predict(untuned_dur,test)$predicted
dur_untunedrf_test_RMSE<-sqrt(mean((dur_untunedrf_test_pred-test$duration)^2))
```

```
#Tuned Random Forest
# Establish a list of possible values for hyper-parameters
mtry.values <- seq(3,8,1)
nodesize.values <- seq(4,12,2)
ntree.values <- seq(3e3,9e3,1e3)

# Create a data frame containing all combinations
hyper_grid <- expand.grid(mtry = mtry.values, nodesize = nodesize.values, ntree = ntree.values)

# Create an empty vector to store OOB error values
oob_err <- c()

# Write a loop over the rows of hyper_grid to train the grid of models
for (i in 1:nrow(hyper_grid)) {

    # Train a Random Forest model
   model <- rfsrc(acq_formula,data = train,
                            mtry = hyper_grid$mtry[i],
                            nodesize = hyper_grid$nodesize[i],
                            ntree = hyper_grid$ntree[i])


    # Store OOB error for the model
    oob_err[i] <- model$err.rate[length(model$err.rate)]
}

# Identify optimal set of hyperparmeters based on OOB error
opt_i <- which.min(oob_err)



tuned_acq <- rfsrc(acq_formula,data = train,
                            mtry = hyper_grid[opt_i,1],
                            nodesize = hyper_grid[opt_i,2],
                            ntree = hyper_grid[opt_i,3])



acq_tunedrf_train_pred<-ifelse(predict(tuned_acq,train)$predicted>0.5,1,0)
acq_tunedrf_train_acc<-sum(acq_tunedrf_train_pred==train$acquisition)/length(train$acquisition)
#Confusion Matrix
#table(acq_tunedrf_train_pred,train$acquisition)


acq_tunedrf_test_pred<-ifelse(predict(tuned_acq,test)$predicted>0.5,1,0)
acq_tunedrf_test_acc<-sum(acq_tunedrf_test_pred==test$acquisition)/length(test$acquisition)

#Confusion Matrix
#table(acq_tunedrf_test_pred,test$acquisition)



# Establish a list of possible values for hyper-parameters
```

```
mtry.values <- seq(3,8,1)
nodesize.values <- seq(4,12,2)
ntree.values <- seq(3e3,9e3,1e3)

# Create a data frame containing all combinations
hyper_grid <- expand.grid(mtry = mtry.values, nodesize = nodesize.values, ntree = ntree.values)

# Create an empty vector to store OOB error values
oob_err <- c()

# Write a loop over the rows of hyper_grid to train the grid of models
for (i in 1:nrow(hyper_grid)) {

    # Train a Random Forest model
   model <- rfsrc(dur_formula,data = train,
                            mtry = hyper_grid$mtry[i],
                            nodesize = hyper_grid$nodesize[i],
                            ntree = hyper_grid$ntree[i])


    # Store OOB error for the model
    oob_err[i] <- model$err.rate[length(model$err.rate)]
}

# Identify optimal set of hyperparmeters based on OOB error
opt_i <- which.min(oob_err)



tuned_dur <- rfsrc(dur_formula,data = train,
                            mtry = hyper_grid[opt_i,1],
                            nodesize = hyper_grid[opt_i,2],
                            ntree = hyper_grid[opt_i,3])


dur_tunedrf_train_pred<-predict(tuned_dur,train)$predicted
dur_tunedrf_train_RMSE<-sqrt(mean((dur_tunedrf_train_pred-train$duration)^2))


dur_tunedrf_test_pred<-predict(tuned_dur,test)$predicted
dur_tunedrf_test_RMSE<-sqrt(mean((dur_tunedrf_test_pred-test$duration)^2))
```