Modeling Mortgage Customer Retention

Objective

A bank is looking to understand the possible reasons why mortgage customers may choose to leave after the 'lock-in' period of their mortgage deal ends. They wish to understand what could be done differently in an effort to ultimately improve their customer retention in the future. The problem is multifaceted due to the range of customer personal profiles as well as the aspects pertaining to their specific mortgage deals.

Provided with the personal data and bank details of 399 of their customers, a request was made to analyze the given data to determine if any personal circumstances are significantly associated with a customer's decision to stay or leave the bank. Limited to the application of generalized linear models, various combinations of the 22 personal attributes are modeled using binary logistic regression to test their relationship with the account status as the response variable.

Methods

With the scope of this analysis limited to the use of generalized linear models, a binomial distribution was chosen as representative of the bank data given that the response variable has a binary outcome. As the explanatory variables for this analysis are both discrete and continuous, certain assessment tools do not apply in some cases, specifically goodness-of-fit test statistics and p-values in relation to continuous data. In addition, as this analysis has a wide range of variables under consideration, a means to compare models that are not nested and do not share common variables is of great importance.

Therefore, the four main criteria for assessing model fit here are Akaike Information Criterion, Bayesian Information Criterion, McFadden pseudo-R-Squared and model accuracy determined from classification tables. In the event a model consists of only categorical variables, the residual deviance and p-values are also assessed to determine fit.

In addition to assessment criteria, the problem of overfitting the data is a strong concern when selecting a model. There is a simple rule to prevent overfitting and it refers to the number of parameters included in the selected model. When looking at the data, the general rule is to have ten to fifteen times as many observations with the outcome of interest as there are parameters in the model. For example, in this analysis, there are 93 cases with the outcome of interest which means the final model must have less than nine parameters.

Analysis

Data

The data provided by the bank include information for 399 of their customers with 23 attributes describing the personal circumstances of the customer. There are four attributes deemed redundant in this analysis: Scheme Details, Account Number, Start Date and End Date. The information described by Scheme Details is included in both Scheme Name (Name) and Scheme Code (Code), so its inclusion would not provide any additional information. Account Number is an identifier and unique to each account and is therefore irrelevant. Finally, the applicable information associated with the Start and End Dates is already included in the analysis within the Loan Term (Term) attribute. With the exclusion of those four attributes, eighteen potential explanatory variables and one response variable are left, consisting of both discrete and continuous variables.

Let it be noted that the continuous variables could be grouped together, but for this analysis, those variables were left untouched.

```r
#display structure of the mortgage data set
mortgage <- read.csv("mortgage.csv", header=TRUE)
mortgage$Code <- as.factor(mortgage$Code)
str(mortgage)
```

```
## 'data.frame':    399 obs. of  19 variables:
##  $ Status    : Factor w/ 2 levels "D","L": 1 1 2 2 2 2 2 2 2 2 ...
##  $ Holder    : Factor w/ 2 levels "One","Two": 1 2 1 1 2 1 2 1 1 1 ...
##  $ Post      : Factor w/ 195 levels "B 735LY","BD164NQ",..: 158 158 116 26 26 16 16 40 102 103 ...
##  $ Type      : Factor w/ 4 levels "","D","I","P": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Amount    : num  46000 46000 36000 0 0 0 0 0 0 0 ...
##  $ Name      : Factor w/ 10 levels "DISC","DMAT",..: 10 10 2 2 2 3 3 4 2 10 ...
##  $ Code      : Factor w/ 10 levels "1","2","3","4",..: 10 10 4 3 3 2 2 3 3 10 ...
##  $ Balance   : num  29849 29849 23418 20309 20309 ...
##  $ Term      : int  25 25 20 25 25 25 25 25 17 23 ...
##  $ Age       : int  40 39 51 47 46 38 34 44 57 38 ...
##  $ Region    : Factor w/ 10 levels "EM","L","N","NW",..: 4 4 3 3 3 1 1 5 3 3 ...
##  $ Membership: int  10 10 10 12 12 12 12 12 12 8 ...
##  $ Gender    : Factor w/ 2 levels "F","M": 2 1 1 2 1 2 1 2 2 2 ...
##  $ Branch    : Factor w/ 2 levels "B","N": 1 2 1 1 1 1 1 1 2 1 ...
##  $ Direct    : Factor w/ 2 levels "D","N": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Internet  : Factor w/ 2 levels "I","N": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Agent     : Factor w/ 2 levels "A","N": 2 2 1 1 1 2 2 1 1 1 ...
##  $ Acron     : Factor w/ 43 levels "A0101","A0102",..: 12 12 30 34 34 33 33 23 40 34 ...
##  $ Salary    : int  0 0 0 0 0 0 0 0 0 65040 ...
```

As shown above, the data set consists of 13 discrete variables, labeled factor, and 6 continuous variables, labeled int or num.

Null Model

The first step in the analysis is to establish the null model for the data which includes no explanatory variables, just an intercept. The null deviance, degrees of freedom (Df), Akaike Information Criteria (AIC), Bayesian Information Criteria (BIC), the McFadden pseudo-R-Squared, and the accuracy of the model are noted below for comparison with the rest of the potential models.

```r
#build the null model
null <- glm(Status ~ 1, data=mortgage, family=binomial)

#extract relevant information from null model including residual deviance,
#residual degrees of freedom, AIC, BIC, and pseudo r-squared into array
null_model <- cbind(deviance(null), df.residual(null), AIC(null),
                    BIC(null), pR2(null)[4])
colnames(null_model) <- c("Deviance","Df","AIC", "BIC","R_Sq")
rownames(null_model) <- "Null"

#convert array to dataframe
null.df <- as.data.frame(null_model)

#predict results of null model on full data set
predict_null <- predict(null, type='response')

#calculate accuracy of the predicted results
Accuracy <- sum(mortgage$Status == 'L' & predict_null > 0.5, mortgage$Status == 'D'
```

```r
                & predict_null < 0.5)/nrow(mortgage)
null.df <- cbind(null.df, Accuracy)
null_min_max <- as.matrix(null.df)
null_min_max <- null_min_max[1,3:6]
null.df
```

```
##      Deviance  Df      AIC      BIC R_Sq  Accuracy
## Null 433.2936 398 435.2936 439.2826    0 0.7669173
```

Single Predictor Models

Beginning with a single predictor, a set of models are created by adding each explanatory variable to the null model to determine whether or not their presence improves the fit of the model.

Below is the code for the eighteen single predictor models and their associated assessment criteria.

```r
model_predictor <- function(predictors,num,interact) {
  #Input  - predictors is an array of variable names used in the model
  #       - num is the number of variables included in the model
  #       - interact is a boolean operator for the inclusion of interaction terms
  #       in the model
  #Output - the dataframe including all variable models with their associated residual
  #       deviance, residual degrees of freedom, AIC, BIC, pseudo-r-squared,
  #       accuracy percentage, and error percentage

  if (interact == TRUE) {
    #determine combinations of variables including interaction terms
    comb <- combinations(n=length(predictors), r=num, v=predictors)
    comb2 <- combinations(n=length(predictors), r=num-1, v=predictors)
    form <- array(0, dim=c(nrow(comb2),1))
    for (i in c(1:nrow(comb2))) {
      form1 <- paste(comb, collapse=" + ")
        form2 <- paste("+", paste(comb2[i,], collapse=":"))
        form3 <- paste(form1, paste(form2))
        form[i] <- paste("Status ~ ", (paste(form3)))
    }
  } else {
    #determine combinations of variables excluding interaction terms
      comb <- combinations(n=length(predictors), r=num, v=predictors)
      form <- array(0, dim=c(nrow(comb),1))
      for (i in c(1:nrow(comb))) {
        form[i] <- paste("Status ~ ", (paste(comb[i,], collapse= " + ")))}
      }

#builds a model for each variable included in predictors array
    model <- lapply(form, function(x) {
        glm(substitute(i, list(i=as.formula(x))), family=binomial, data=mortgage)
    })
#extract relevant information from null model including residual deviance,
#residual degrees of freedom, AIC, BIC, and pseudo-r-squared into array
    DEV <- as.numeric(lapply(model, deviance))
    Df <- as.numeric(lapply(model, df.residual))
    AIC <- as.numeric(lapply(model, AIC))
    BIC <- as.numeric(lapply(model, BIC))
```

```r
    r2 <- lapply(model, pR2)

    R2 <- array(0, dim=c(nrow(form),1))
    for (i in c(1:nrow(form))){
        R2[i] <- r2[[i]][4]
    }
#combine relevant info into array and convert to dataframe
    final <- cbind(DEV, Df, AIC, BIC, R2)
    colnames(final) <- c("Deviance", "Df", "AIC", "BIC", "R_Sq")
    rownames(final) <- form
    models.df <- as.data.frame(final)

#predict results of each model on full data set
    predictor <- lapply(model, function(x) {predict(x, type='response')})

#calculate accuracy of the predicted results for each model
    accuracy <- lapply(predictor, function(x) {
            accuracy <- sum(mortgage$Status == 'L' & x > 0.5, mortgage$Status == 'D'
                            & x < 0.5)/nrow(mortgage)})

    accuracy <- as.data.frame(as.numeric(accuracy), col.names="Accuracy")
  rownames(accuracy) <- rownames(final)

#calculate associated error and add accuracy and error percentages to output dataframe
    error <- 1 - accuracy
    acc_err <- cbind(accuracy, error)
    colnames(acc_err) <- c("Accuracy", "Error")
    models.df <- cbind(models.df, acc_err)
    return(models.df)
}
#build and predict all single predictor models
single_model <- model_predictor(names(mortgage)[2:19],1,FALSE)
single_model
```

```
##                        Deviance  Df      AIC       BIC         R_Sq
## Status ~  Acron       315.57242 356 401.5724  573.0978 2.716892e-01
## Status ~  Age         423.06846 397 427.0685  435.0464 2.359868e-02
## Status ~  Agent       319.59336 397 323.5934  331.5713 2.624093e-01
## Status ~  Amount      432.95572 397 436.9557  444.9336 7.798390e-04
## Status ~  Balance     431.06587 397 435.0659  443.0438 5.141428e-03
## Status ~  Branch      420.05841 397 424.0584  432.0363 3.054560e-02
## Status ~  Code        280.00990 389 300.0099  339.8995 3.537641e-01
## Status ~  Direct      426.49837 397 430.4984  438.4763 1.568278e-02
## Status ~  Gender      433.29361 397 437.2936  445.2715 2.929963e-08
## Status ~  Holder      433.29362 397 437.2936  445.2715 7.570472e-09
## Status ~  Internet    431.69447 397 435.6945  443.6724 3.690689e-03
## Status ~  Membership  429.38182 397 433.3818  441.3597 9.028067e-03
## Status ~  Name        278.80084 389 298.8008  338.6905 3.565545e-01
## Status ~  Post         17.31513 204 407.3151 1185.1626 9.600383e-01
## Status ~  Region      402.12397 389 422.1240  462.0136 7.193655e-02
## Status ~  Salary      425.87273 397 429.8727  437.8507 1.712671e-02
## Status ~  Term        425.45800 397 429.4580  437.4359 1.808386e-02
## Status ~  Type        422.36776 395 430.3678  446.3236 2.521584e-02
##                        Accuracy      Error
```

```
## Status ~  Acron      0.7994987 0.20050125
## Status ~  Age        0.7619048 0.23809524
## Status ~  Agent      0.7669173 0.23308271
## Status ~  Amount     0.7669173 0.23308271
## Status ~  Balance    0.7669173 0.23308271
## Status ~  Branch     0.7669173 0.23308271
## Status ~  Code       0.8195489 0.18045113
## Status ~  Direct     0.7669173 0.23308271
## Status ~  Gender     0.7669173 0.23308271
## Status ~  Holder     0.7669173 0.23308271
## Status ~  Internet   0.7669173 0.23308271
## Status ~  Membership 0.7669173 0.23308271
## Status ~  Name       0.8195489 0.18045113
## Status ~  Post       0.9874687 0.01253133
## Status ~  Region     0.7744361 0.22556391
## Status ~  Salary     0.7593985 0.24060150
## Status ~  Term       0.7669173 0.23308271
## Status ~  Type       0.7669173 0.23308271
```

Any models over the null AIC and BIC and below the null R-Squared and accuracy are eliminated from further analysis. Out of the remaining models, the minimum AIC and BIC, as well as the maximum R-Squared and calculated accuracy percentages, are noted and used as the comparison values for the next set of models.

```
subset_model <- function(mod, comp) {
  #Input  - mod is the output from the model_predictor function
  #       - comp is the array of values for comparison
  #Output - subset dataframe with only models below the minimum AIC and BIC thresholds
  #       as well as the maximum r-squared and accuracy percentages

  new_model <- subset(mod, AIC <= comp[1] & BIC <= comp[2] &
                      R_Sq >= comp[3] & Accuracy >= comp[4])

  #assure models avoid overfitting
    new_model <- subset(new_model, (null.df$Df - new_model$Df) < 9)
    return(new_model)
}

min_max <- function(new_mod) {
  #Input  - output from subset_model function
  #Output - matrix of new minimum AIC and BIC and maximum r-squared and accuracy
    minmax <- cbind(min(new_mod$AIC), min(new_mod$BIC), max(new_mod$R_Sq),
                    max(new_mod$Accuracy))
    colnames(minmax) <- c("AIC", "BIC", "R_Sq", "Accuracy")
    rownames(minmax) <- "Min/Max"
    minmax <- as.matrix(minmax)
    return(minmax)
}
#subset the single predictor models based on null model AIC, BIC, r-squared
#and accuracy percentage
new_single_model <- subset_model(single_model, null_min_max)

#calculate the new minimums and maximums for two-predictor model comparison
min_max_single <- min_max(new_single_model)
new_single_model
```

```
##                  Deviance  Df      AIC      BIC       R_Sq  Accuracy
## Status ~  Agent  319.5934 397 323.5934 331.5713 0.26240928 0.7669173
## Status ~  Branch 420.0584 397 424.0584 432.0363 0.03054560 0.7669173
## Status ~  Direct 426.4984 397 430.4984 438.4763 0.01568278 0.7669173
## Status ~  Term   425.4580 397 429.4580 437.4359 0.01808386 0.7669173
##                      Error
## Status ~  Agent  0.2330827
## Status ~  Branch 0.2330827
## Status ~  Direct 0.2330827
## Status ~  Term   0.2330827
```

```r
min_max_single
```

```
##              AIC      BIC      R_Sq  Accuracy
## Min/Max 323.5934 331.5713 0.2624093 0.7669173
```

As a result of the assessment criteria and the concern with overfitting, fourteen models are eliminated from further consideration, leaving four possible explanatory variables. Of these four variables, Agent provides the minimum AIC and BIC as well as the maximum R-Squared and accuracy which will be used as the benchmark in assessing the two-predictor models.

Two-Predictor Models

Two-predictor models are then created using all combinations of the remaining single predictor models. Given four explanatory variables, six combinations are possible in constructing two-predictor models.

```r
#vector of predictor variable names
x2 <- c("Agent", "Branch", "Direct", "Term")

#build and predict all two variable predictor models
double_model <- model_predictor(x2,2,FALSE)
double_model
```

```
##                           Deviance  Df      AIC      BIC       R_Sq
## Status ~  Agent + Branch  211.9087 396 217.9087 229.8756 0.51093500
## Status ~  Agent + Direct  307.3715 396 313.3715 325.3384 0.29061621
## Status ~  Agent + Term    319.1233 396 325.1233 337.0902 0.26349410
## Status ~  Branch + Direct 414.1958 396 420.1958 432.1626 0.04407602
## Status ~  Branch + Term   407.6901 396 413.6901 425.6569 0.05909056
## Status ~  Direct + Term   419.2157 396 425.2157 437.1826 0.03249041
##                            Accuracy      Error
## Status ~  Agent + Branch  0.9223058 0.07769424
## Status ~  Agent + Direct  0.7669173 0.23308271
## Status ~  Agent + Term    0.7669173 0.23308271
## Status ~  Branch + Direct 0.7669173 0.23308271
## Status ~  Branch + Term   0.7669173 0.23308271
## Status ~  Direct + Term   0.7669173 0.23308271
```

The same procedure is performed, only retaining the combinations of variables which meet all four criteria, eliminating all others and updating the new minimums and maximums for comparison in the next set.

```r
#subset the two variable predictor models based on single model AIC, BIC,
#r-squared and accuracy percentage
new_double_model <- subset_model(double_model, min_max_single)
new_double_model
```

```
##                          Deviance  Df      AIC      BIC      R_Sq
```

```
## Status ~  Agent + Branch 211.9087 396 217.9087 229.8756 0.5109350
## Status ~  Agent + Direct 307.3715 396 313.3715 325.3384 0.2906162
##                           Accuracy      Error
## Status ~  Agent + Branch 0.9223058 0.07769424
## Status ~  Agent + Direct 0.7669173 0.23308271
```

```r
#calculate the new minimums and maximums for three predictor model comparison
min_max_double <- min_max(new_double_model)
min_max_double
```

```
##              AIC      BIC     R_Sq  Accuracy
## Min/Max 217.9087 229.8756 0.510935 0.9223058
```

Of the six combinations of two-predictor models, only two models meet the assessment criteria as shown above.

Three-Predictor Models

With only three variables left, there is only one possible combination of a three-predictor model which is found below.

```r
#vector of predictor variable names
x3 <- c("Agent", "Branch", "Direct")

#build and predict all three variable predictor models
triple_model <- model_predictor(x3,3,FALSE)
triple_model
```

```
##                                   Deviance  Df      AIC      BIC      R_Sq
## Status ~  Agent + Branch + Direct 205.5426 395 213.5426 229.4984 0.5256275
##                                    Accuracy      Error
## Status ~  Agent + Branch + Direct 0.9223058 0.07769424
```

```r
#subset the three variable predictor models based on double model AIC, BIC,
#r-squared and accuracy percentage
new_triple_model <- subset_model(triple_model, min_max_double)

#calculate the new minimums and maximums for future model comparison
min_max_triple <- min_max(triple_model)
```

Using the final three-predictor model, all variations of interaction terms are added to determine if an interaction between any of the two variables is significant. With three variables, there are three different possible interaction terms.

```r
#build and predict all three variable predictor models with interaction terms
triple_model_int <- model_predictor(x3,3,TRUE)
triple_model_int
```

```
##                                                 Deviance  Df      AIC
## Status ~  Agent + Branch + Direct + Agent:Branch  177.0313 394 187.0313
## Status ~  Agent + Branch + Direct + Agent:Direct  205.4553 394 215.4553
## Status ~  Agent + Branch + Direct + Branch:Direct 203.1151 394 213.1151
##                                                       BIC      R_Sq
## Status ~  Agent + Branch + Direct + Agent:Branch  206.9761 0.5914288
## Status ~  Agent + Branch + Direct + Agent:Direct  235.4001 0.5258288
## Status ~  Agent + Branch + Direct + Branch:Direct 233.0599 0.5312300
##                                                   Accuracy      Error
```

```
## Status ~  Agent + Branch + Direct + Agent:Branch  0.9223058 0.07769424
## Status ~  Agent + Branch + Direct + Agent:Direct  0.9223058 0.07769424
## Status ~  Agent + Branch + Direct + Branch:Direct 0.9223058 0.07769424
```

The nested models with interaction terms are evaluated based on the assessment criteria against the three-predictor model.

```
#subset the models with three variable predictors and interaction terms based on
#three predictor model AIC, BIC, r-squared and accuracy percentage
new_triple_model_int <- subset_model(triple_model_int, min_max_triple)
new_triple_model_int
```

```
##                                                   Deviance  Df      AIC
## Status ~  Agent + Branch + Direct + Agent:Branch 177.0313 394 187.0313
##                                                        BIC      R_Sq
## Status ~  Agent + Branch + Direct + Agent:Branch 206.9761 0.5914288
##                                                    Accuracy      Error
## Status ~  Agent + Branch + Direct + Agent:Branch 0.9223058 0.07769424
```

In addition, given the model only contains categorical variables, the difference in residual deviance and the associated p-values are reviewed for the nested models.

```
#initialize arrays for difference in residual deviances and degrees of freedom
#as well as the p-values
res_diff <- array(0, dim=c(3,2))
p <- array(0,dim=c(3,1))

#calculate the difference in residual deviances and degrees of freedom for three
#predictor models and those with interaction terms and find associated p-values
for (i in c(1:nrow(triple_model_int))) {
  res_diff[i,] <- c((new_triple_model[1,1] - triple_model_int[i,1]),
                    (new_triple_model[1,2] - triple_model_int[i,2]))
  p[i,] <- pchisq(res_diff[i,1], res_diff[i,2], lower.tail=FALSE)
}
pval <- cbind(res_diff, p)
rownames(pval) <- rownames(triple_model_int)
colnames(pval) <- c("Diff Res Dev", "Diff Deg Freedom", "p-value")
pval
```

```
##                                                    Diff Res Dev
## Status ~  Agent + Branch + Direct + Agent:Branch    28.51127033
## Status ~  Agent + Branch + Direct + Agent:Direct     0.08723474
## Status ~  Agent + Branch + Direct + Branch:Direct    2.42750565
##                                                    Diff Deg Freedom
## Status ~  Agent + Branch + Direct + Agent:Branch                  1
## Status ~  Agent + Branch + Direct + Agent:Direct                  1
## Status ~  Agent + Branch + Direct + Branch:Direct                 1
##                                                         p-value
## Status ~  Agent + Branch + Direct + Agent:Branch   9.315466e-08
## Status ~  Agent + Branch + Direct + Agent:Direct   7.677225e-01
## Status ~  Agent + Branch + Direct + Branch:Direct 1.192225e-01
```

```
#test p-values at 0.05 significance level
sig <- as.matrix(as.logical(p < 0.05))
rownames(sig) <- rownames(triple_model_int)
colnames(sig) <- "Significant"
sig
```

```
##                                                   Significant
## Status ~  Agent + Branch + Direct + Agent:Branch         TRUE
## Status ~  Agent + Branch + Direct + Agent:Direct        FALSE
## Status ~  Agent + Branch + Direct + Branch:Direct       FALSE
```

Assessment based on residual deviance and p-values confirms the results from the other criteria leaving a three-predictor model with an interaction term between Agent and Branch variables.

Prediction Ability

Using the caTools library in R, the data are randomly split into training and testing sets using a specified split ratio so the final model can be assessed for its prediction ability. The model is then fitted using the training data before creating a classification table to determine training set accuracy rates. The fitted model is then used to predict the results of the testing set to determine prediction accuracy rates. The procedure is repeated multiple times for both sets, each with a new random sample, before being averaged and used as the training accuracy and testing prediction rates.

```r
#extract number of levels for all factor columns of dataframe
fac <- sapply(mortgage[,sapply(mortgage, is.factor)], nlevels)

#subset factor columns with greater than four levels
fac <- subset(fac, fac > 4)

#duplicate dataframe
mortgage.n <- mortgage

#convert factor columns with more than four levels into numeric columns
mortgage.n[names(fac)] <- sapply(mortgage[names(fac)], function(x) as.numeric(x))

#set number of test and training sets and initialize empty array for accuracy percentages
n = 10
accuracy <- array(0, dim=c(2,n))

#randomly sample training and testing sets for set number n
for (i in c(1:n)) {
  split <- sample.split(mortgage.n$Status, SplitRatio = 0.7)
  mortgage.train <- subset(mortgage.n, split == TRUE)
  mortgage.test <- subset(mortgage.n, split == FALSE)

  #fit model with training set
  model <- glm(Status ~ Agent + Branch + Direct + Agent:Branch, family=binomial,
               data=mortgage.train)

  #predict model with training set
  train_predict <- predict(model, type = 'response')

  #calculate training set accuracy
  train_acc <- sum(mortgage.train$Status == 'L' & train_predict > 0.5,
                   mortgage.train$Status == 'D' & train_predict < 0.5)/nrow(mortgage.train)

  #predict model with test set
  predictor <- predict(model, newdata=mortgage.test, type='response')

  #calculate test set accuracy
```

```r
  test_acc <- sum(mortgage.test$Status == 'L' & predictor > 0.5,
                  mortgage.test$Status == 'D' & predictor < 0.5)/nrow(mortgage.test)

  #assign accuracy values to array
  accuracy[,i] <- rbind(train_acc, test_acc)
}
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
#take the average of all sample sets
accuracy <- rowMeans(accuracy)

#combine final model accuracy with average accuracy of training and testing sets
accuracy <- c(new_triple_model[1,6],accuracy)
Accuracy <- as.data.frame(accuracy)
rownames(Accuracy) <- c("Full Set", "Training Set", "Testing Set")
Accuracy
```

```
##               accuracy
## Full Set     0.9223058
## Training Set 0.9218638
## Testing Set  0.9200000
```

As shown above, the lack of a significant difference between the model accuracy with the full data set and the average model accuracy with the training set shows that the final model selected produces virtually the same accuracy even when created using a fraction of the original data. The similar findings with the prediction accuracy for the testing set illustrate the consistent nature of the final model.

```r
#extract the coefficients from the final model
final <- glm(Status ~ Agent * Branch + Direct, family=binomial, data=mortgage)
c <- coefficients(final)
c
```

```
##    (Intercept)         AgentN        BranchN        DirectN AgentN:BranchN
##       5.034682      -1.295902       2.257633      -2.372094      -7.304443
```

The final model takes the form:

$logit(u) = 5.0347 - 1.2949 * AN + 2.2576 * BN - 2.3721 * DN - 7.0344 * AN : BN$

where AN = Agent No, BN = Branch No, DN = Direct No, AN:BN = Agent No and Branch No, logit(u) is the log-odds of a successful outcome and u = probability of a successful outcome.

The model shows that a customer
* lacking an agent will decrease the log-odds by 1.2949
* not being a branch user will increase the log-odds by 2.2576
* not being a direct user will decrease the log-odds by 2.3721
* lacking an agent and not being a branch user will decrease the log-odds by 7.0344

Therefore, the probability of a random customer staying with the bank at the end of the 'lock-in' period depending on all combinations of personal circumstance is calculated below.

```r
#calculate binary matrix of all combinations of personal circumstance
A <- bincombinations(3)
x <- array(1, dim=c(nrow(A),1))
A <- cbind(x, A)
x[1:6,1] <- 0
```

```r
A <- cbind(A,x)

#calculate the dot product of the personal circumstances matrix and
#the coefficients determined by the final model
b <- A %*% c
Probability <- cbind(lapply(b, function(x) {(exp(x)/(1 + exp(x)))*100}))
result <- cbind(Probability, A[,2:5] == FALSE)
colnames(result) <- c("Probability", "Agent", "Branch User", "Direct User",
                      "Agent & Branch User")
result
```

```
##      Probability Agent Branch User Direct User Agent & Branch User
## [1,] 99.35338    TRUE  TRUE        TRUE        TRUE
## [2,] 93.47826    TRUE  TRUE        FALSE       TRUE
## [3,] 99.93197    TRUE  FALSE       TRUE        TRUE
## [4,] 99.27554    TRUE  FALSE       FALSE       TRUE
## [5,] 97.67694    FALSE TRUE        TRUE        TRUE
## [6,] 79.68442    FALSE TRUE        FALSE       TRUE
## [7,] 21.28169    FALSE FALSE       TRUE        FALSE
## [8,] 2.459947    FALSE FALSE       FALSE       FALSE
```

Conclusion

According to the results of this analysis, it is in the bank's best interests to place more importance on customers having a relationship with the bank as well as initiating specific points of contact. Having an agent is the best option with a 99.27% probability of retention, while the combinations of agent/direct user and agent/branch/direct user, are the 2nd and 3rd options available with probabilities of retention of 99.93% and 99.35%, respectively. However, if a customer doesn't want an agent, the next best option is for the customer to bank by both branch and telephone which only decreases the probability of retention by 1.6%. In conclusion, the goal for the bank is to direct customers to adopt the combination of having an agent and banking directly by telephone as this results in the highest likelihood of customer retention.