Bella Lin, Emilie Baek DA 6813
Emily Bates, Makenzie Kalebick Spring 2024

Bank Marketing Case Study

Executive Summary

This case study addresses the challenge of predicting term deposit subscriptions for clients of a Portuguese banking institution, utilizing logistic regression and linear discriminant analysis (LDA) models. The aim is twofold: to comprehend the factors influencing clients' subscription decisions and to develop accurate predictive models for forecasting subscription outcomes. Through an examination of related literature, various supervised machine learning algorithms were explored, with the Random Forest Classifier identified as the most effective predictor in prior studies. Methodologically, logistic regression and LDA models were applied to tackle the binary classification problem posed by the outcome variable. Data preprocessing involved addressing missing values, transforming variables, and creating dummy variables. Models were initially trained and tested on unbalanced datasets before being resampled to achieve balance, enhancing their predictive performance.

Following a comprehensive evaluation, the balanced logistic regression with predictors from a stepwise selection (M2_bal) and the balanced LDA model emerged as the top performers, exhibiting superior predictive accuracy and positive predictive value when applied to balanced datasets. Specifically, M2_bal is recommended as the optimal choice for predicting term deposit subscriptions while mitigating false positives. Its balanced performance across metrics positions it as the preferred model for predicting term deposit subscriptions. Leveraging the balanced M2 enables informed decision-making, tailored marketing efforts, and improved business performance, ultimately fostering stronger client relationships and maximizing subscription rates for term deposits.

Problem

The task we were provided with for this case study was to predict if the client of a Portuguese banking institution will subscribe (yes or no) to a term deposit (y). Because it is a classification problem, we want to build a logistic model as well as a linear discriminant analysis model to identify which is the best predictor in the context of this case study. Our problem encompasses two main aspects: first, to understand the factors that influence a client's decision to subscribe to a term deposit, and second, to develop robust predictive models that accurately forecast these subscription outcomes. The outcome variable, y, serves as the focal point for prediction, while a set of predictor variables offer insights into client demographics, past interactions with the bank, economic indicators, and campaign-specific details. Ultimately, the successful development and validation of predictive models can equip the banking institution with valuable insights for optimizing marketing strategies, enhancing client engagement, and maximizing the efficiency of future direct marketing campaigns.

Review of Related Literature

In a previous study by Singh et al. (2023)[1] that aimed to predict if a client will subscribe to a term deposit, various supervised machine learning algorithms were employed. First, the study experiments with logistic regression, which will be described further in *Methodology*. Secondly, the study utilizes a decision tree, a training model used to "forecast the value based on the decision rules from the training data" (p. 91). Benefits of using a decision tree includes no assumptions about the structure of the data and that it can detect non-linear effects. Next, the study employs a Random Forest Classifier, which simply put combines a number of decision trees to make predictions. This approach is often used for large datasets, and it can minimize the variance in the model (p. 90). Finally, the study experiments with a Support Vector Machine (SVM). An SVM is used to solve classification tasks with two groups by creating hyperplanes to separate the two classes (p. 91).

The study compared the models' performances on accuracy and found that the Random Forest Classifier is the algorithm that best predicted whether a client will subscribe to a term deposit (p. 91).

Methodology

The models and methods that will be accessed in our analysis will be appropriate to the nature of the problem and to the type of response and predictors variables. As described earlier, the predictor is binary and our goal was to predict if a client will subscribe to a term deposit, making this a two-group classification problem. The predictors in the data set consisted of 10 categorical variables and 10 numeric variables. In the *Data* section below, we will elaborate on how each of these variables were handled during the data cleaning and data preprocessing stages. In these stages, we also resampled our data to not include missing values and we created a new variable using information from an existing variable. The final

[1] https://books.google.com/books?id=VujNEAAAQBAJ&lpg=PA82&ots=CZdayPH6vU&dq=predict%20if%20the%20client%20will%20subscribe%20to%20a%20term%20deposit&lr&hl=da&pg=PA91#v=onepage&q=predict%20if%20the%20client%20will%20subscribe%20to%20a%20term%20deposit&f=false (Accessed Feb 9, 2024)

step we took to prepare the data for modeling, was to split the preprocessed data into a training set and a test set using an 80/20 ratio.

The first method we employed was logistic regression, a statistical model used for classification and predictive modeling (IBM, n.d.).[2] Logistic regression can also be used for inference analysis, however, this will not be the main focus of this study. The model returns a probability of an event occurring. In our case, an event is when *y* is "yes". While logistic regression can be applied to a response variable with three or more possible outcomes (IBM, n.d.), we will apply *binomial* logistic regression appropriate to our binary response variable. Logistic regression is a parametric model, meaning that it entails assumptions about the data. These assumptions include 1) linearity in the logit function for continuous variables, 2) no multicollinearity among the variables, and 3) no highly influential outliers. If these fundamental assumptions are violated, the model may produce inaccurate results. For this reason, we will evaluate these four assumptions in our analysis. Another limitation of logistic regression is that the model requires a larger representative sample to have "sufficient statistical power to detect a significant effect" (IBM, n.d.) compared to a linear regression.

The second and final method we used was linear discriminant analysis (LDA). LDA is a classification technique that finds the linear combinations of features that best separate the classes (IBM, 2023).[3] The coefficients obtained from LDA can be interpreted as the weights assigned to each feature in this linear combination, indicating their contribution to the discriminant function. Consequently, the model assumes that the data is linearly separable (IBM, 2023).

Comparing logistic regression and LDA to the methods described in *Review of Related Literature*, decision trees, random forest classifiers, and SVMs are generally more flexible due to their ability to capture complex nonlinear relationships and handle high-dimensional data without making strong assumptions about the underlying data distribution. On the other hand, logistic regression and LDA are generally more interpretable. For instance, the coefficients of logistic regression represent the change in the log-odds of the outcome for a one-unit change in the corresponding predictor (IBM, n.d.), making them interpretable in terms of the effect of each predictor on the outcome (i.e. inference) as briefly mentioned previously.

In our analysis, both methods were initially fitted and tested using the unbalanced training and testing set, after which we resampled the data to produce a balanced training and

[2] https://www.ibm.com/topics/logistic-regression. Accessed Feb 10, 2024.
[3] https://www.ibm.com/topics/linear-discriminant-analysis. Accessed Feb 10, 2024.

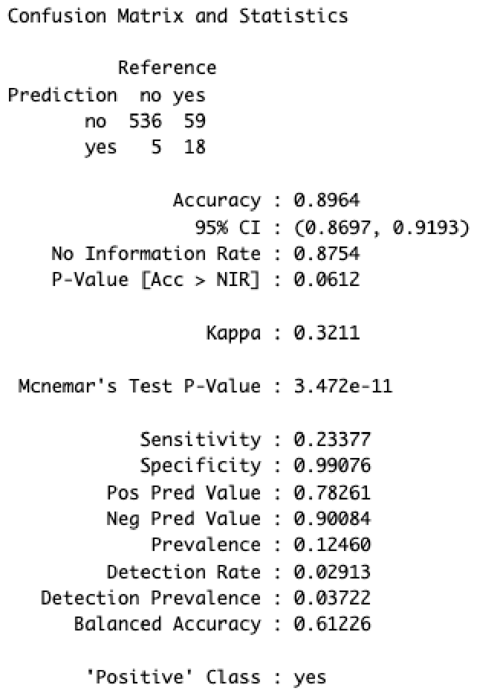
testing set. Additionally, the logistic models were accessed using all predictors followed by a model using the predictors resulting from a stepwise selection.

Data

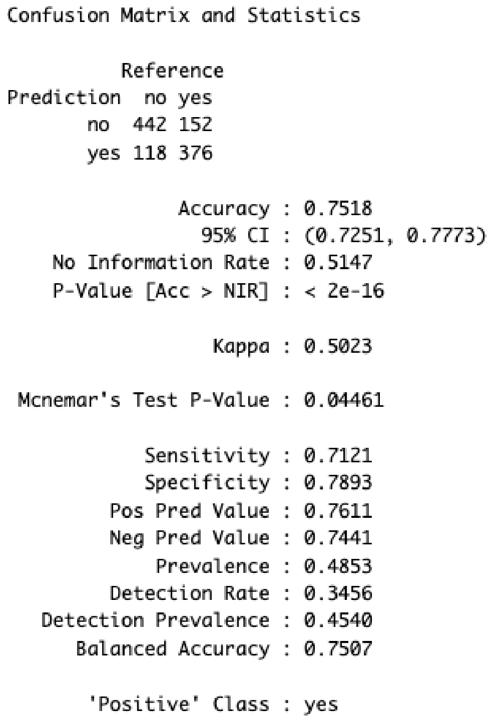
The dataset provided for this project contains data from a Portuguese banking institution that relates to direct marketing campaign phone calls. It originally contained 4119 objects and 21 variables. After removing missing values, our number of observations decreased to 3090. The variables age, duration, campaign, pdays, previous, emp.var.rate, cons.price.idx, cons.conf.idx, euribor3m, and nr.employed are all numeric in the initial dataset. We changed the remaining character variables into factors. In reviewing the documentation, we found an issue with the variable pdays. While it is a numeric variable, the number 999 means that the client was not previously contacted. To fix this issue, we categorized the values as "less than 1 week" for values between 0 and 7, "between 1 and 2 weeks" for values between 7 and 14, and "more than 2 weeks" for values between 14 and 998. Additionally, we created a new variable to account for whether they had been contacted or not. The ones previously coded as 999 became "no" and those that had a real pdays value became "yes". Upon examining our variables further, we discovered that the variable 'default' only had one 'yes' value. Therefore, we removed 'default'. The documentation points out that the variable 'duration' highly affects the output target. For example, if duration is 0, y is no. Because this variable is directly related to the outcome (y), we must also remove it from our model. To include 'marital', 'housing', 'loan', 'contact', 'poutcome', 'and 'contacted' in our models, we needed to create dummy variables that contained either a 1 or 0. Marital_dummy1 was coded for divorce, marital_dummy2 was coded for married, housing_dummy was coded for yes, loan_dummy was coded for yes, contact_dummy1 was coded for cellular, poutcome_dummy1 was coded for failure, poutcome_dummy2 was coded for success, and contacted_dummy was coded for yes. This concluded our variable manipulation.

Results

Given the task of predicting term deposit subscription, we believe that models with balanced sensitivity and specificity would be the most ideal since it will offer a well-rounded approach to identifying both subscribers and non-subscribers effectively. Using both logistics regression and LDA methods, we have created 6 models for this task. For the first model, M1, we started with all predictors and removed predictors with highest VIF value until all predictors have a VIF value less than 10 to account for multicollinearity issues. The final predictors chosen for M1 are: age, marital_dummy1 (divorce), marital_dummy2 (married), housing_dummy (yes), loan_dummy (yes), contact_dummy (cellular), campaign, previous, poutcome_dummy1 (failure), poutcome_dummy2 (success), cons.price.idx, cons.conf.idx, and nr.employed. Then we created models for both original(unbalanced) dataset and balanced dataset.



```
Confusion Matrix and Statistics

          Reference
Prediction  no yes
       no  536  59
       yes   5  18

               Accuracy : 0.8964
                 95% CI : (0.8697, 0.9193)
    No Information Rate : 0.8754
    P-Value [Acc > NIR] : 0.0612

                  Kappa : 0.3211

 Mcnemar's Test P-Value : 3.472e-11

            Sensitivity : 0.23377
            Specificity : 0.99076
         Pos Pred Value : 0.78261
         Neg Pred Value : 0.90084
             Prevalence : 0.12460
         Detection Rate : 0.02913
   Detection Prevalence : 0.03722
      Balanced Accuracy : 0.61226

       'Positive' Class : yes
```

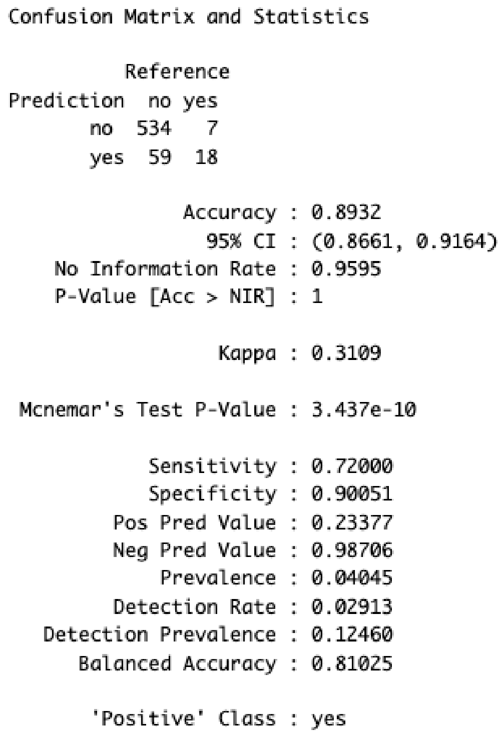


```
Confusion Matrix and Statistics

          Reference
Prediction  no yes
       no  442 152
       yes 118 376

               Accuracy : 0.7518
                 95% CI : (0.7251, 0.7773)
    No Information Rate : 0.5147
    P-Value [Acc > NIR] : < 2e-16

                  Kappa : 0.5023

 Mcnemar's Test P-Value : 0.04461

            Sensitivity : 0.7121
            Specificity : 0.7893
         Pos Pred Value : 0.7611
         Neg Pred Value : 0.7441
             Prevalence : 0.4853
         Detection Rate : 0.3456
   Detection Prevalence : 0.4540
      Balanced Accuracy : 0.7507

       'Positive' Class : yes
```

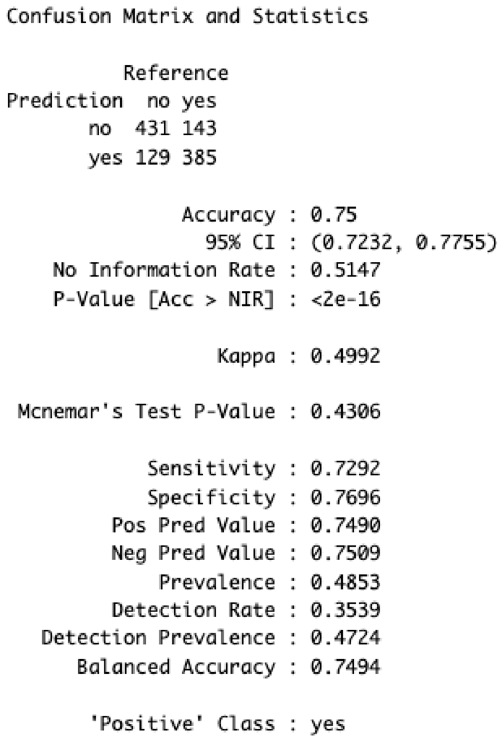
(M1 model with unbalanced dataset) (M1 model with balanced dataset)

According to the result above, the unbalanced M1 model prioritizes specificity over sensitivity, achieving a high true negative rate but a low true positive rate. However, the balanced M1 model maintains a better balance between sensitivity and specificity, resulting in improved performance in correctly identifying positive cases while still maintaining a respectable true negative rate. While the unbalanced model may have higher overall accuracy, its compromised sensitivity could limit the effectiveness in identifying term deposit subscribers accurately. Therefore, the balanced M1 model appears more suitable for this predictive task due to its better balance between sensitivity and specificity.

Next, we used stepwise model selection to identify the most significant predictors for our M2 models. The selected predictors were: nr.employed, contact_dummy (cellular), poutcome_dummy1 (failure) + cons.conf.idx, campaign , cons.price.idx, and poutcome_dummy2 (success). Again, we want to create models for both balanced and unbalanced data.



```
Confusion Matrix and Statistics

          Reference
Prediction  no yes
       no  534   7
       yes  59  18

               Accuracy : 0.8932
                 95% CI : (0.8661, 0.9164)
    No Information Rate : 0.9595
    P-Value [Acc > NIR] : 1

                  Kappa : 0.3109

 Mcnemar's Test P-Value : 3.437e-10

            Sensitivity : 0.72000
            Specificity : 0.90051
         Pos Pred Value : 0.23377
         Neg Pred Value : 0.98706
             Prevalence : 0.04045
         Detection Rate : 0.02913
   Detection Prevalence : 0.12460
      Balanced Accuracy : 0.81025

       'Positive' Class : yes
```

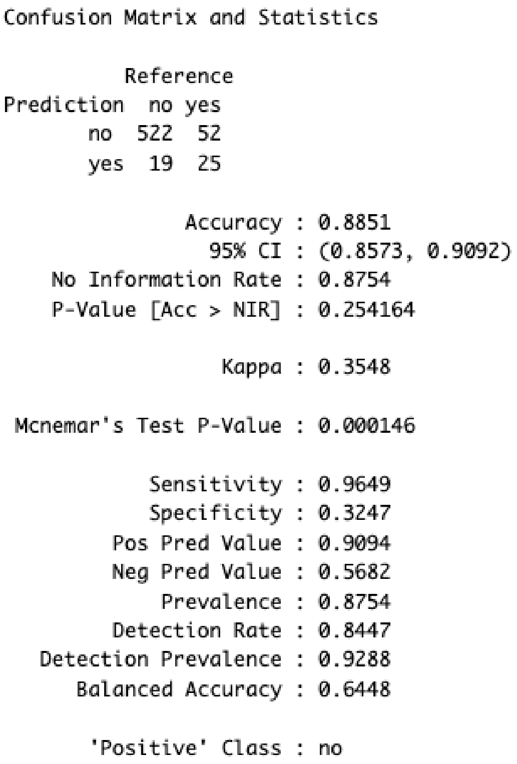


```
Confusion Matrix and Statistics

          Reference
Prediction  no yes
       no  431 143
       yes 129 385

               Accuracy : 0.75
                 95% CI : (0.7232, 0.7755)
    No Information Rate : 0.5147
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.4992

 Mcnemar's Test P-Value : 0.4306

            Sensitivity : 0.7292
            Specificity : 0.7696
         Pos Pred Value : 0.7490
         Neg Pred Value : 0.7509
             Prevalence : 0.4853
         Detection Rate : 0.3539
   Detection Prevalence : 0.4724
      Balanced Accuracy : 0.7494

       'Positive' Class : yes
```

(M2 model with unbalanced dataset) (M2 model with balanced dataset)

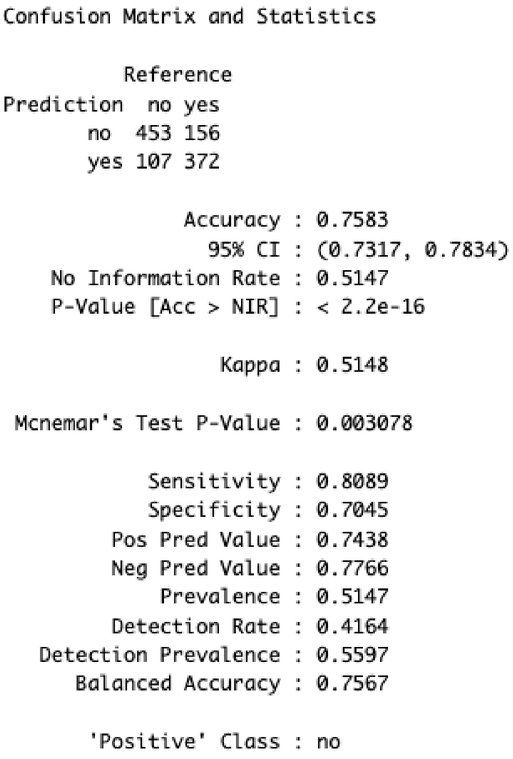
Comparing the M2 models, the unbalanced model demonstrates strong sensitivity and specificity but falls short in positive predictive value, indicating a higher rate of false positives. In contrast, the balanced M2 model maintains good sensitivity and specificity while significantly improving positive predictive value, making it more adept at accurately predicting term deposit subscriptions while minimizing false positives. Therefore, the balanced M2 model appears more effective for this predictive task due to its superior balance between sensitivity and positive predictive value.

Lastly, linear discriminant analysis was used to create our final two models, using the unbalanced and balanced data respectively. All predictors were included in the LDA models: age, marital_dummy1 (divorce), marital_dummy2 (married), housing_dummy (yes), loan_dummy (yes), contact_dummy (cellular), campaign, previous, poutcome_dummy1 (failure), poutcome_dummy2 (success), emp.var.rate, euribor3m, cons.price.idx, cons.conf.idx, and nr.employed.



```
Confusion Matrix and Statistics

          Reference
Prediction  no yes
       no  522  52
       yes  19  25

               Accuracy : 0.8851
                 95% CI : (0.8573, 0.9092)
    No Information Rate : 0.8754
    P-Value [Acc > NIR] : 0.254164

                  Kappa : 0.3548

 Mcnemar's Test P-Value : 0.000146

            Sensitivity : 0.9649
            Specificity : 0.3247
         Pos Pred Value : 0.9094
         Neg Pred Value : 0.5682
             Prevalence : 0.8754
         Detection Rate : 0.8447
   Detection Prevalence : 0.9288
      Balanced Accuracy : 0.6448

       'Positive' Class : no
```

(LDA model with unbalanced dataset)



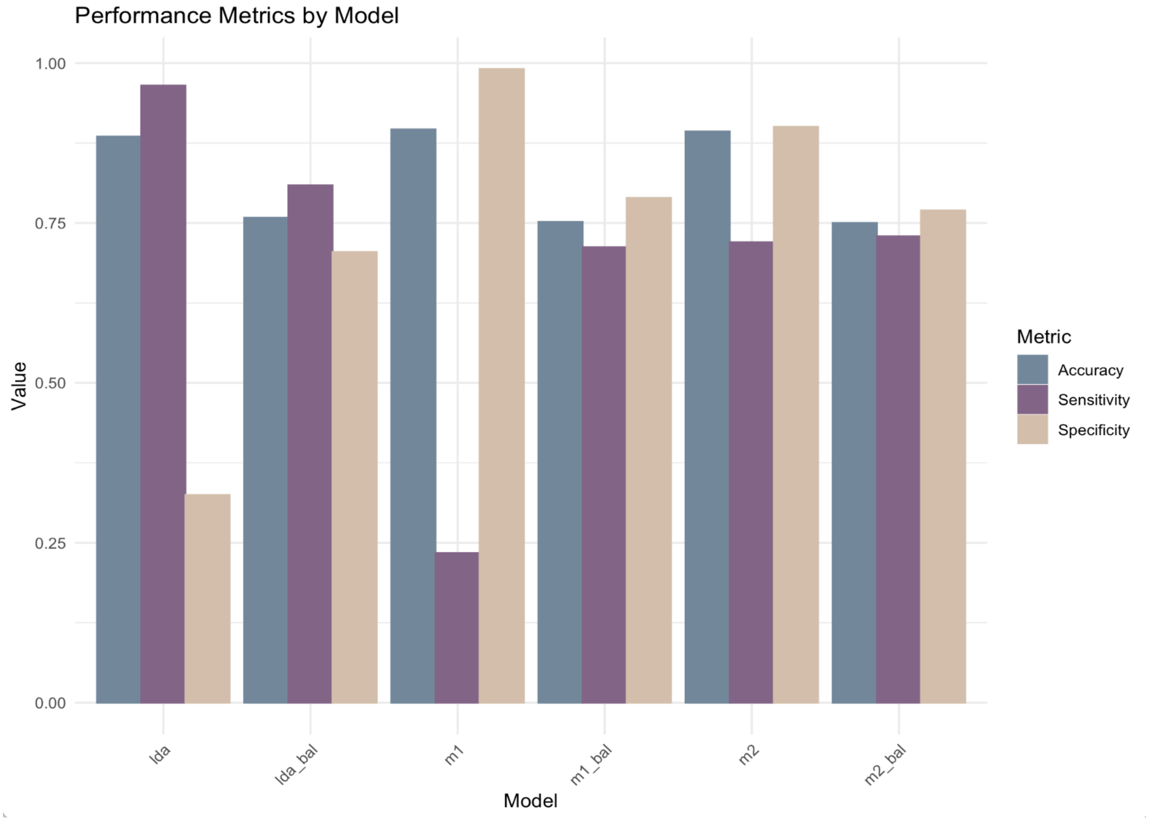
```
Confusion Matrix and Statistics

          Reference
Prediction  no yes
       no  453 156
       yes 107 372

               Accuracy : 0.7583
                 95% CI : (0.7317, 0.7834)
    No Information Rate : 0.5147
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5148

 Mcnemar's Test P-Value : 0.003078

            Sensitivity : 0.8089
            Specificity : 0.7045
         Pos Pred Value : 0.7438
         Neg Pred Value : 0.7766
             Prevalence : 0.5147
         Detection Rate : 0.4164
   Detection Prevalence : 0.5597
      Balanced Accuracy : 0.7567

       'Positive' Class : no
```

(LDA model with balanced dataset)

Comparing the results above, the unbalanced model shows exceptionally high sensitivity but low specificity, leading to a higher rate of false positives. On the other hand, the balanced LDA model achieves a better balance between sensitivity and specificity, resulting in improved predictive accuracy with a notable positive predictive value. Therefore, the balanced LDA model appears more suitable for accurately predicting term deposit subscriptions while minimizing false positives.

Looking back at all the models we created, we can see that models with balanced dataset improved balance between sensitivity and specificity compared to their unbalanced counterparts. The balanced models generally achieve higher sensitivity while maintaining respectable levels of specificity, making them more adept at identifying both true positives and true negatives effectively.

As seen on the graph on the following page, among the balanced models, LDA_bal stands out for producing higher positive prediction values. On the other hand, M2_bal achieved the best balance between sensitivity and specificity, therefore making it the best suitable model for accurately predicting term deposit subscriptions while minimizing false positives.



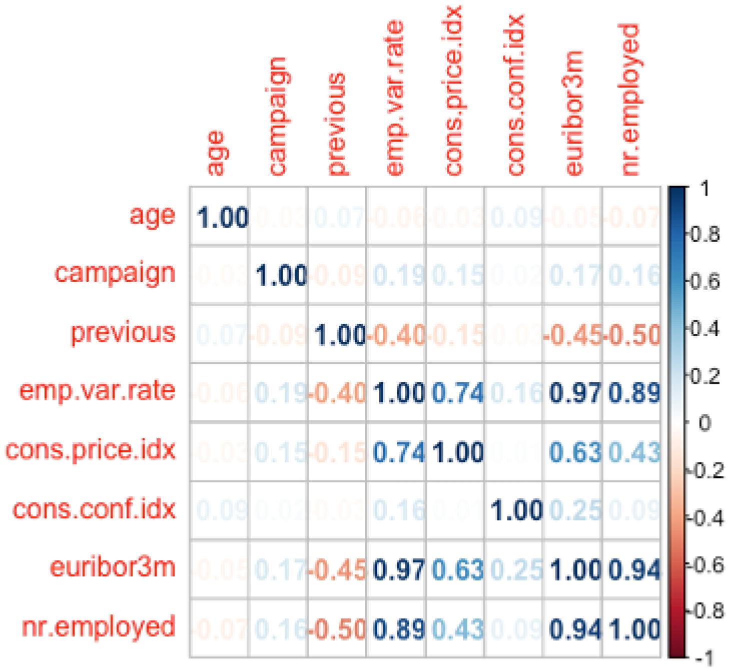
Conclusion & Recommendations

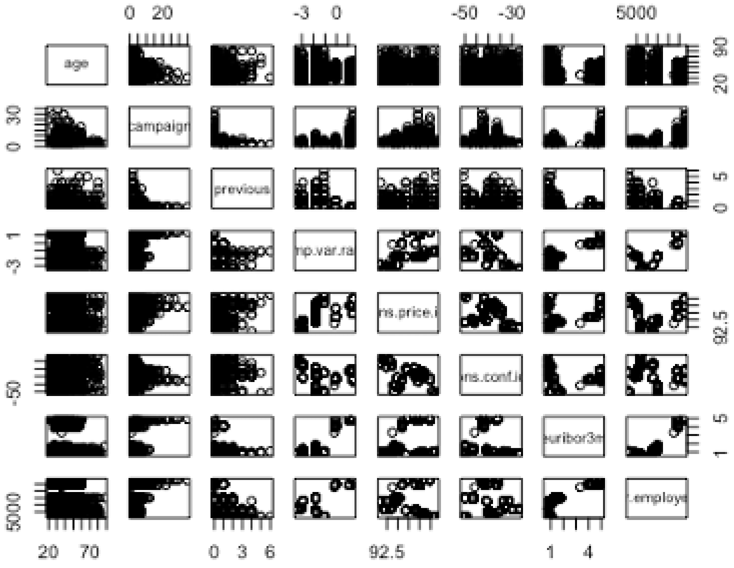
In summary, our analysis aimed to predict term deposit subscriptions for clients of a Portuguese banking institution, employing logistic regression and linear discriminant analysis (LDA). Through meticulous examination, we found that models constructed with a balanced dataset consistently outperformed their unbalanced counterparts, achieving a more optimal balance between sensitivity and specificity. These findings underscore the importance of dataset balance in improving model performance, highlighting the significance of robust data preprocessing techniques in predictive modeling.

After thorough evaluation, the balanced logistic regression model with selected predictors emerges as the most suitable choice for predicting term deposit subscriptions while effectively minimizing false positives. Its balanced performance positions it as the preferred model for guiding marketing strategies, optimizing client engagement, and informing future direct marketing campaigns within the banking institution. By leveraging M2_bal, the institution can make informed decisions, tailor marketing efforts, and enhance overall business performance, ultimately fostering stronger relationships with clients and maximizing subscription rates for term deposits.

When considering alternative methods to this business problem, we find inspiration in the study by Singh et al. as described in Review of Related Literature. This study found that the Random Forest Classifier was the best predictor of whether a client will subscribe to a term deposit. While the Random Forest Classifier is significantly harder to interpret than the methods accessed in our study, its ability to predict non-linear patterns outweighs this limitation, particularly considering the goal of this study being prediction rather than inference.

Appendix





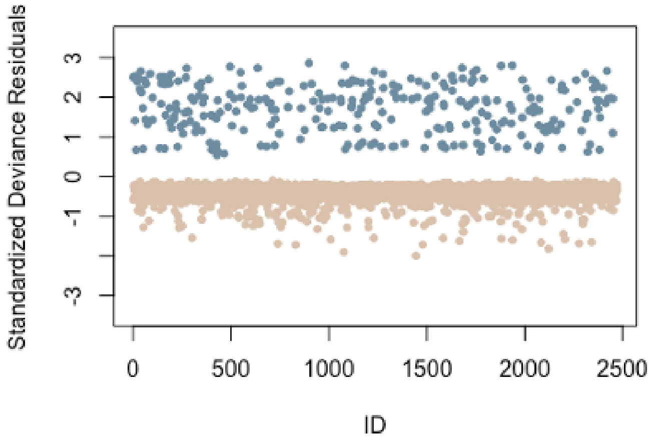
Selection of M1 predictors:

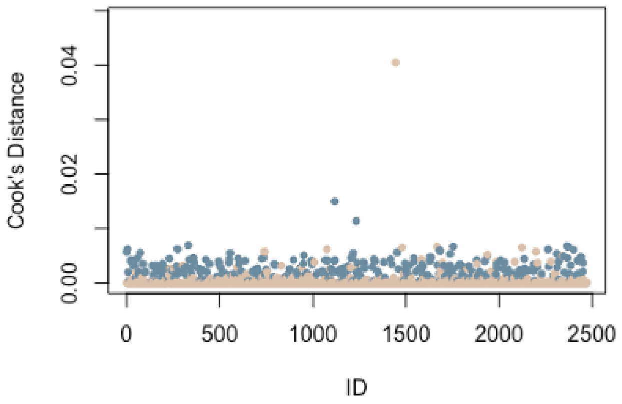
```
### Building model with unbalanced data
m1 <- glm(y ~ age+marital_dummy1+marital_dummy2+housing_dummy+loan_dummy+
          contact_dummy+campaign+previous+poutcome_dummy1+poutcome_dummy2+
          emp.var.rate+cons.price.idx+cons.conf.idx+euribor3m+nr.employed+
          contacted_dummy, data = bank_train, family = binomial)
summary(m1)
```

```
##
## Call:
## glm(formula = y ~ age + marital_dummy1 + marital_dummy2 + housing_dummy +
##         loan_dummy + contact_dummy + campaign + previous + poutcome_dummy1 +
##         poutcome_dummy2 + emp.var.rate + cons.price.idx + cons.conf.idx +
##         euribor3m + nr.employed + contacted_dummy, family = binomial,
##         data = bank_train)
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -1.132e+02  6.074e+01  -1.864   0.0624 .
## age             1.286e-02  6.705e-03   1.918   0.0551 .
## marital_dummy1  -2.828e-01  2.690e-01  -1.051   0.2931
## marital_dummy2  -1.839e-01  1.700e-01  -1.081   0.2795
## housing_dummy       1.082e-01  1.433e-01   0.755   0.4503
## loan_dummy          -1.661e-01  1.982e-01  -0.838   0.4020
## contact_dummy       1.129e+00  2.301e-01   4.908 9.22e-07 ***
## campaign     -6.336e-02  4.240e-02  -1.494   0.1352
## previous       5.799e-02  1.794e-01   0.323   0.7465
## poutcome_dummy1 -6.824e-01  3.075e-01  -2.219   0.0265 *
## poutcome_dummy2  1.727e-01  6.736e-01   0.256   0.7976
## emp.var.rate -5.985e-01  2.333e-01  -2.565   0.0103 *
## cons.price.idx   1.176e+00  3.950e-01   2.978   0.0029 **
## cons.conf.idx           4.522e-02  2.113e-02   2.140   0.0324 *
## euribor3m    -1.277e-01  2.961e-01  -0.431   0.6662
## nr.employed             3.812e-04  5.337e-03   0.071   0.9431
## contacted_dummy  1.044e+00  6.660e-01   1.567   0.1171
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##         Null deviance: 1799.5  on 2471  degrees of freedom
## Residual deviance: 1418.6  on 2455  degrees of freedom
## AIC: 1452.6
##
## Number of Fisher Scoring iterations: 6
```

```
### Checking for multicolliniarity
vif(m1)

##             age  marital_dummy1  marital_dummy2   housing_dummy      loan_dummy
##        1.318023        1.349427        1.455620        1.022916        1.016234
##   contact_dummy        campaign        previous poutcome_dummy1 poutcome_dummy2
##        1.419296        1.041171        3.986140        2.598249        8.435170
##    emp.var.rate  cons.price.idx   cons.conf.idx       euribor3m     nr.employed
##       29.013972       13.344030        3.219422       54.971137       42.227013
## contacted_dummy
##        9.218469

# Removing euribor3m
m1 <- glm(y ~ age+marital_dummy1+marital_dummy2+housing_dummy+loan_dummy+
            contact_dummy+campaign+previous+poutcome_dummy1+poutcome_dummy2+
            emp.var.rate+cons.price.idx+cons.conf.idx+nr.employed+contacted_dummy,
            data = bank_train, family = binomial)
vif(m1)

##             age  marital_dummy1  marital_dummy2   housing_dummy      loan_dummy
##        1.317928        1.349978        1.456023        1.022477        1.016419
##   contact_dummy        campaign        previous poutcome_dummy1 poutcome_dummy2
##        1.407418        1.034848        3.995827        2.601403        8.423814
##    emp.var.rate  cons.price.idx   cons.conf.idx     nr.employed contacted_dummy
##       25.503523       10.453165        1.411143       15.651066        9.206119

# Removing emp.var.rate
m1 <- glm(y ~ age+marital_dummy1+marital_dummy2+housing_dummy+loan_dummy+
            contact_dummy+campaign+previous+poutcome_dummy1+poutcome_dummy2+
            cons.price.idx+cons.conf.idx+nr.employed+contacted_dummy,
            data = bank_train, family = binomial)
vif(m1)

##             age  marital_dummy1  marital_dummy2   housing_dummy      loan_dummy
##        1.317152        1.346197        1.454807        1.020871        1.016346
##   contact_dummy        campaign        previous poutcome_dummy1 poutcome_dummy2
##        1.262986        1.030145        3.981611        2.586550        8.670405
##  cons.price.idx   cons.conf.idx     nr.employed contacted_dummy
##        1.357716        1.194942        1.381971        9.447838
```

```
# Removing contacted_dummy
m1 <- glm(y ~ age+marital_dummy1+marital_dummy2+housing_dummy+loan_dummy+
          contact_dummy+campaign+previous+poutcome_dummy1+poutcome_dummy2+
          cons.price.idx+cons.conf.idx+nr.employed,
          data = bank_train, family = binomial)
vif(m1)

##             age  marital_dummy1  marital_dummy2   housing_dummy      loan_dummy
##        1.316542        1.346295        1.454758        1.019705        1.016102
##   contact_dummy        campaign        previous poutcome_dummy1 poutcome_dummy2
##        1.261984        1.030172        3.636186        2.645323        2.337782
##  cons.price.idx   cons.conf.idx     nr.employed
##        1.358048        1.193094        1.381208
```





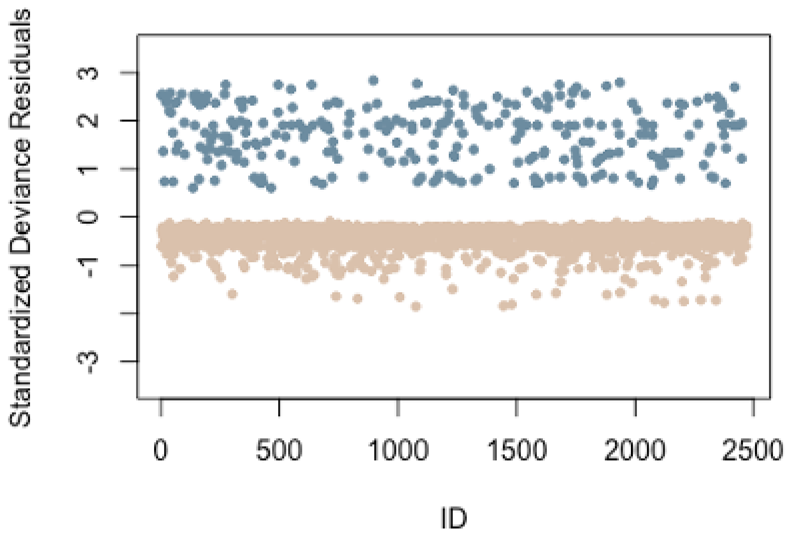
```
# Which observation has cook's d larger than 0.25?
(inf.id=which(cooks.distance(m1)>0.25)) # No influential points

## named integer(0)
```

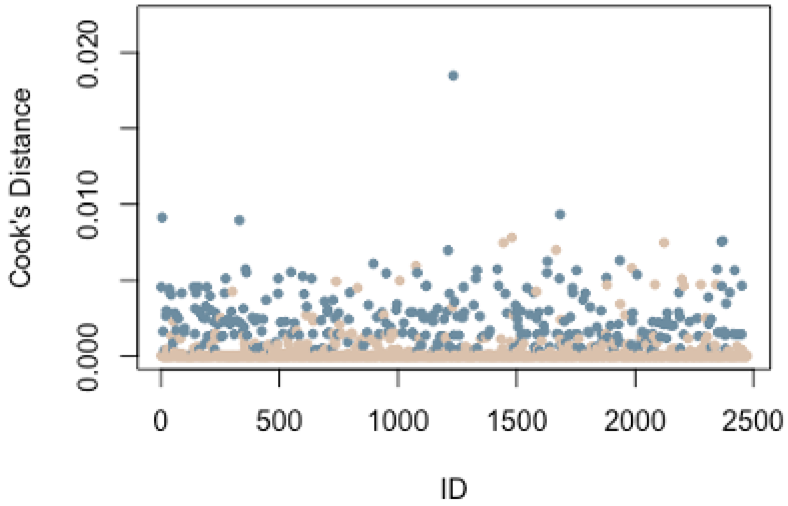
Selection of M2 predictors:

```
#####  Stepwise Selection with AIC
null_model = glm(y ~ 1, data = bank, family = binomial)
full_model = m1

step.model.AIC = step(null_model, scope = list(upper = full_model),
                      direction = "both", test = "Chisq", trace = F)
summary(step.model.AIC)

##
## Call:
## glm(formula = y ~ nr.employed + poutcome_dummy2 + contact_dummy +
##     cons.conf.idx + poutcome_dummy1 + campaign + cons.price.idx,
##     family = binomial, data = bank)
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)     36.5688923 10.8029977   3.385 0.000712 ***
## nr.employed     -0.0106389  0.0008146 -13.060  < 2e-16 ***
## poutcome_dummy2  1.3294750  0.2272942   5.849 4.94e-09 ***
## contact_dummy    0.9138821  0.1905973   4.795 1.63e-06 ***
## cons.conf.idx    0.0344833  0.0113136   3.048 0.002304 **
## poutcome_dummy1 -0.4111162  0.1826831  -2.250 0.024421 *
## campaign        -0.0679744  0.0380607  -1.786 0.074108 .
## cons.price.idx   0.1803267  0.1095550   1.646 0.099765 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2264.4  on 3089  degrees of freedom
## Residual deviance: 1808.3  on 3082  degrees of freedom
## AIC: 1824.3
##
## Number of Fisher Scoring iterations: 6
```

```r
# Best model based on stepwise selection
m2 <- glm(y ~ contact_dummy+campaign+poutcome_dummy1+poutcome_dummy2+
          cons.price.idx+cons.conf.idx+nr.employed, bank_train, family = binomial)
summary(m2)
```

```
##
## Call:
## glm(formula = y ~ contact_dummy + campaign + poutcome_dummy1 +
##     poutcome_dummy2 + cons.price.idx + cons.conf.idx + nr.employed,
##     family = binomial, data = bank_train)
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)     36.9040478 12.1040453   3.049  0.00230 **
## contact_dummy    1.0009818  0.2181666   4.588 4.47e-06 ***
## campaign        -0.0689990  0.0426900  -1.616  0.10603
## poutcome_dummy1 -0.5589138  0.2060582  -2.712  0.00668 **
## poutcome_dummy2  1.2893870  0.2528174   5.100 3.40e-07 ***
## cons.price.idx   0.1861229  0.1227445   1.516  0.12943
## cons.conf.idx    0.0275472  0.0125167   2.201  0.02775 *
## nr.employed     -0.0108784  0.0009326 -11.664  < 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: 1799.5  on 2471  degrees of freedom
## Residual deviance: 1434.3  on 2464  degrees of freedom
## AIC: 1450.3
##
## Number of Fisher Scoring iterations: 6
```





```
# Which observation has cook's d larger than 0.25?
(inf.id=which(cooks.distance(m1)>0.25)) # No influential points

## named integer(0)
```

Balancing Data:

```
### Resample with more balanced data
bank_yes_cust = bank %>% filter(y == "yes")
bank_no_cust = bank %>% filter(y == "no")
```

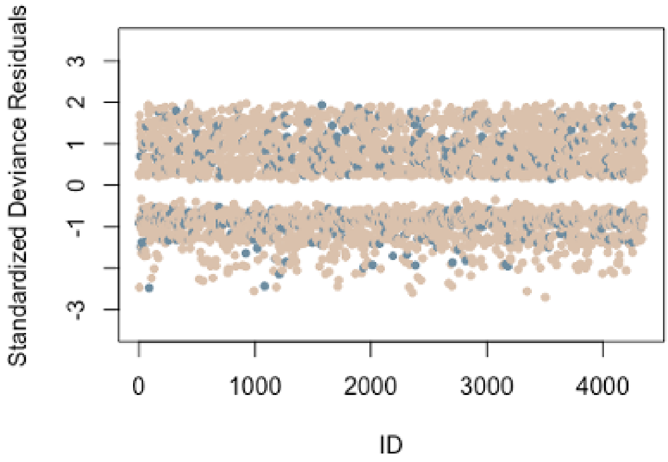
```
set.seed(1)
sample_yes_cust = sample_n(bank_yes_cust, nrow(bank_no_cust), replace = TRUE) #matching the same row number
bank_bal = rbind(bank_no_cust,sample_yes_cust) #combining both so they will have a balance data


# Split data into training and testing balanced samples
set.seed(1)
index_bal <- sample(nrow(bank_bal),0.8*nrow(bank_bal),replace = F) # 80/20 split
banktrain_bal <- bank_bal[index_bal,]
banktest_bal <- bank_bal[-index_bal,]


### Building model with balanced data
m1_bal = glm(y ~ age + marital_dummy1 + marital_dummy2 + housing_dummy + loan_dummy +
          contact_dummy + campaign + previous + poutcome_dummy1 + poutcome_dummy2 +
          cons.price.idx + cons.conf.idx  + nr.employed, data = banktrain_bal,
          family = binomial)
summary(m1_bal) # Look at results
```

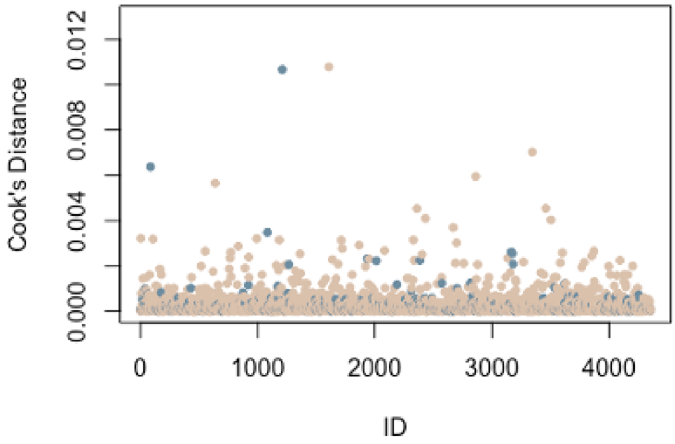
```
##
## Call:
## glm(formula = y ~ age + marital_dummy1 + marital_dummy2 + housing_dummy +
##         loan_dummy + contact_dummy + campaign + previous + poutcome_dummy1 +
##         poutcome_dummy2 + cons.price.idx + cons.conf.idx + nr.employed,
##         family = binomial, data = banktrain_bal)
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)     36.064605   6.605122   5.460 4.76e-08 ***
## age              0.012337   0.003622   3.406 0.000659 ***
## marital_dummy1  -0.392052   0.136654  -2.869 0.004119 **
## marital_dummy2  -0.207762   0.084512  -2.458 0.013957 *
## housing_dummy   -0.125710   0.071687  -1.754 0.079502 .
## loan_dummy      -0.233602   0.098108  -2.381 0.017263 *
## contact_dummy    0.705256   0.100278   7.033 2.02e-12 ***
## campaign        -0.058841   0.019098  -3.081 0.002063 **
## previous         0.318919   0.148838   2.143 0.032134 *
## poutcome_dummy1 -0.665898   0.205025  -3.248 0.001163 **
## poutcome_dummy2  0.883471   0.271856   3.250 0.001155 **
## cons.price.idx   0.180038   0.075765   2.376 0.017488 *
## cons.conf.idx    0.016525   0.007001   2.361 0.018249 *
## nr.employed     -0.010307   0.000543 -18.982  < 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: 6032.9  on 4351  degrees of freedom
## Residual deviance: 4782.4  on 4338  degrees of freedom
## AIC: 4810.4
##
## Number of Fisher Scoring iterations: 5
```

```
vif(m1_bal) # Double checking multicollinearity -> Nothing to remove
```

```
##             age  marital_dummy1  marital_dummy2   housing_dummy      loan_dummy
##        1.259346        1.344942        1.415861        1.030578        1.017507
##   contact_dummy        campaign        previous poutcome_dummy1 poutcome_dummy2
##        1.424358        1.032412        5.186456        4.362719        2.243435
##  cons.price.idx   cons.conf.idx     nr.employed
##        1.644122        1.119313        1.477398
```





```
# Which observation has cook's d larger than 0.25?
(inf.id=which(cooks.distance(m1)>0.25)) # No influential points
```

```
## named integer(0)
```

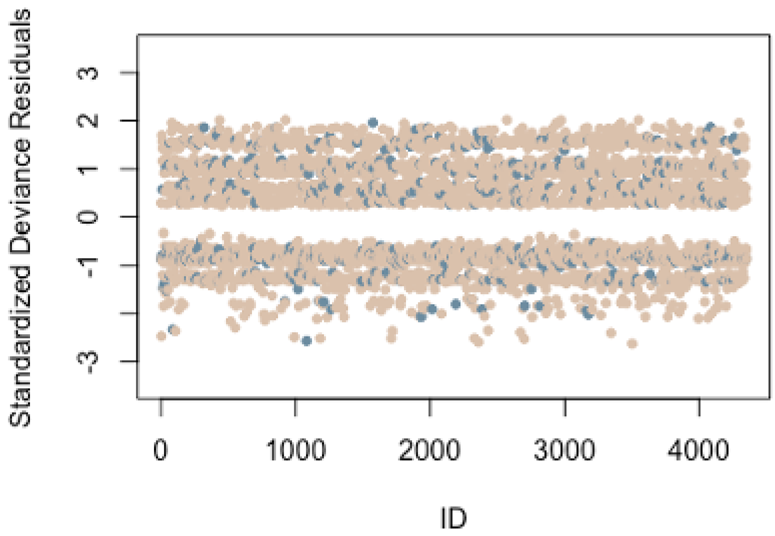
M2 balanced data:

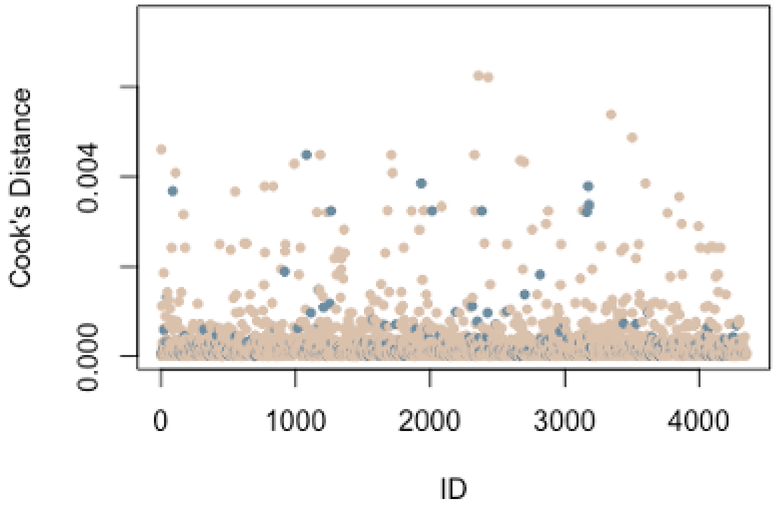
```
##### Using predictors from m2 on balanced data
# (because stepwise selection kept all predictors from m1_bal)
m2_bal = glm(y ~ nr.employed + contact_dummy + poutcome_dummy1 + cons.conf.idx
             + campaign + cons.price.idx + poutcome_dummy2,
             data = banktrain_bal, family = binomial)
summary(m2_bal) # Look at results
```

```
## 
## Call:
## glm(formula = y ~ nr.employed + contact_dummy + poutcome_dummy1 + 
##     cons.conf.idx + campaign + cons.price.idx + poutcome_dummy2, 
##     family = binomial, data = banktrain_bal)
## 
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)    
## (Intercept)     34.1428226  6.4937261   5.258 1.46e-07 ***
## nr.employed     -0.0106758  0.0005361 -19.912  < 2e-16 ***
## contact_dummy    0.7267831  0.0996437   7.294 3.01e-13 ***
## poutcome_dummy1 -0.2683346  0.1051023  -2.553  0.01068 *  
## cons.conf.idx    0.0207887  0.0069351   2.998  0.00272 ** 
## campaign        -0.0589662  0.0190741  -3.091  0.00199 ** 
## cons.price.idx   0.2250188  0.0742491   3.031  0.00244 ** 
## poutcome_dummy2  1.2927767  0.1916808   6.744 1.54e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 6032.9  on 4351  degrees of freedom
## Residual deviance: 4813.4  on 4344  degrees of freedom
## AIC: 4829.4
## 
## Number of Fisher Scoring iterations: 5
```

```
vif(m2_bal) # Double check multicollinearity -> Nothing to remove
```

```
##     nr.employed   contact_dummy poutcome_dummy1   cons.conf.idx        campaign 
##        1.452547        1.414625        1.164563        1.097616        1.031738 
##  cons.price.idx poutcome_dummy2 
##        1.589874        1.128823
```





```
# Which observation has cook's d larger than 0.25?
(inf.id=which(cooks.distance(m1)>0.25)) # No influential points

## named integer(0)
```

Creating LDA Models:

```
# LDA modeling
lda.model_bal = lda(y ~ age+marital_dummy1+marital_dummy2+housing_dummy+loan_dummy+
                    contact_dummy+campaign+previous+poutcome_dummy1+poutcome_dummy2+
                    emp.var.rate+cons.price.idx+cons.conf.idx+euribor3m+nr.employed+
                    contacted_dummy, data = banktrain_bal)
```

```
# View the output
lda.model_bal

## Call:
## lda(y ~ age + marital_dummy1 + marital_dummy2 + housing_dummy +
##     loan_dummy + contact_dummy + campaign + previous + poutcome_dummy1 +
##     poutcome_dummy2 + emp.var.rate + cons.price.idx + cons.conf.idx +
##     euribor3m + nr.employed + contacted_dummy, data = banktrain_bal)
##
## Prior probabilities of groups:
##        no       yes
## 0.4963235 0.5036765
##
## Group means:
##          age marital_dummy1 marital_dummy2 housing_dummy loan_dummy
## no  38.98611     0.11064815      0.5861111     0.5495370  0.1675926
## yes 40.53604     0.09671533      0.5437956     0.5437956  0.1368613
##     contact_dummy campaign  previous poutcome_dummy1 poutcome_dummy2
## no      0.6601852 2.560648 0.1537037       0.1157407      0.01666667
## yes     0.8640511 1.983577 0.5898723       0.1573905      0.21213504
##     emp.var.rate cons.price.idx cons.conf.idx euribor3m nr.employed
## no      0.108287       93.54820     -40.71310  3.666610    5170.566
## yes    -1.328741       93.38747     -40.05379  1.979961    5085.223
##     contacted_dummy
## no       0.01805556
## yes      0.21669708
##
## Coefficients of linear discriminants:
##                           LD1
## age              9.964134e-03
## marital_dummy1  -3.646937e-01
## marital_dummy2  -1.973452e-01
## housing_dummy   -9.440810e-02
## loan_dummy      -2.090376e-01
## contact_dummy    8.629994e-01
## campaign        -4.101191e-02
## previous         3.927696e-02
## poutcome_dummy1 -3.340020e-01
## poutcome_dummy2 -4.011142e-02
## emp.var.rate    -8.549048e-01
## cons.price.idx   1.158808e+00
## cons.conf.idx    3.073154e-02
## euribor3m        1.294793e-01
## nr.employed     -4.296745e-05
## contacted_dummy  6.722092e-01
```