Using Logistic Regression to Predict Bank Telemarketing Campaign Success

Katherine Botz, Jessica Páez Bonilla, Manuel Jordán Expósito

Abstract

The aim of this Predictive Modeling Project is to find an accurate model for predicting the success or failure of a bank telemarketing campaign. We build several logistic regression models to predict the binary Success response variable and select the best fitting model. We split the dataset into Train and Test data, then train each model using the former and calculate the model accuracy using the latter. We first train a logistic model using the significant variables found by fitting a 10-fold Cross-Validation Lasso model, then perform a stepwise model selection by minimizing the Akaike Information Criterion. In this way, we compare the variable selection methods of the Lasso model compared to the stepwise AIC model selection. We select the best model in the end by balancing the R^2, AIC, multicollinearity, and calculated accuracy of the Success prediction in the Test data. Our final model consists of 6 predictor variables and has 92.13% accuracy for prediction.

Introduction

We use an open source dataset with 4119 instances and 21 variables describing a bank telemarketing campaign. The dataset includes descriptive information about the clients themselves (including age, marital status, job type, any loans, and education), as well as information about the previous campaign, current campaign, and statistics related to the social and economic context at the time of the campaign. The response is a binary variable that describes the success or failure of the campaign. The goal is to build a logistic regression model to predict the success of future telemarketing campaigns, and determine which variables are significant in this prediction.

Methods

In our last project we found a Linear Regression Model to predict a continuous response variable. Since this dataset presents a binary classification problem, we will have to generalize the method of finding a Linear Model where the response is the continuous *probability* of being either one of the classes.

In other words, when in the linear model we fit the coefficients $\beta_0, \ldots, \beta_p$ for continuous response Y and predictors $X_1, \ldots, X_p$ such that:

$$E[Y|X_1 = x_1, \ldots, X_p = x_p] = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$$

then for this case we will need to assume Y is a Bernoulli variable and calculate the conditional expectation as follows:

$$E[Y|X_1 = x_1, \ldots, X_p = x_p] = P[Y = 1|X_1 = x_1, \ldots, X_p = x_p]$$

where $P[Y = y] = p^y(1-p)^{1-y}$, for $y = 0, 1$.

However, this conditional expectation η can be any range of values, so we need to map it to the 0 to 1 range that we'd expect for probabilities. For a binary response as in our case, we can take the logistic(η):

$$E[Y|X_1 = x_1, \ldots, X_p = x_p] = logistic(\eta) = \frac{1}{1+e^{\eta}}$$

where $\eta = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$.

Now we can interpret the response as a probability between 0 and 1, and assign the binary response as follows:

- If $\eta \geq 0$, then $P[Y = 1|X_1 = x_1, \ldots, X_p = x_p] \geq 0.5$, and classify $Y = 1$
- If $\eta < 0$, then $P[Y = 1|X_1 = x_1, \ldots, X_p = x_p] < 0.5$, and classify $Y = 0$

Therefore we use logistic regression and the `glm` function in R to find a generalized linear model for predicting the success of bank telemarketing campaigns in our dataset.

Least Absolute Shrinkage and Selection Operator (LASSO) is a regression method that involves penalizing the absolute size of the regression coefficients, whereby in penalization, some of the parameter estimates may be exactly zero. The larger the penalty applied means that this coefficient is not "important," and this importance is measured by the penalization. This particular type of regression is well-suited for models showing high levels of multicollinearity too. Lasso regression performs L1 regularization, which adds a penalty equal to the absolute value of the magnitude of coefficients, on the other hand, we can also find L2 regularization, or Ridge regression.

We will use the significant variables selected from the Lasso function to compare to another model generated using the `stepAIC` function (just as in the linear case) that performs a stepwise model selection with different combinations of variables. This function analyzes each stepwise model for the best fit by finding the one with the smallest Akaike Information Criterion. The AIC is a number that estimates the model fitness compared to its complexity. Therefore the `stepAIC` function gives us a set of variables with the most influence on the response variable while minimizing the multivariate noise.

We will use the Deviance of our logistic regression models as one measure of the fitness of the model. The Deviance measures the difference between the example model and the saturated model, or the one that fits every instance with 100% accuracy. As an analogy to the Linear Model case, we can calculate the R2 from the Deviance:

$$R^2 = 1 - \frac{D}{D_0} \sim 1 - \frac{SSE}{SST}$$

where SSE is the Residual Sum of Squares, SST is the Total Sum of Squares, and D_0 is the Null Deviance; a model where each instance is fit exactly *incorrectly*.

However, this is only an analogy to the linear case, as this R^2 measures the fit of the model. We will use the following function to calculate the R^2 to use during the model evaluation, with caution since the R^2 increases with more predictors in the model. Therefore we will need a few more statistics to determine the best model.

```
r2glm <- function(model) {
  summaryLog <- summary(model)
  1 - summaryLog$deviance / summaryLog$null.deviance
}
```

We use multicollinearity as another statistic for model selection. Even though the response is a logistic expression, the predictors still comprise a linear interaction. Therefore there is a chance that the variables in our model are linearly dependent, and we will try to minimize any dependence in our final selection. We run the `vif` function in R to diagnose multicollinearity and try to find a model where the `vif` is smaller than 5 for all variables.

The goal of fitting our model is to make accurate predictions of the response variable, so we select three final models and use the Test data to predict the campaign success and then compare to the actual values to calculate the model accuracy. In this way, based on accuracy, best fit, and low multicollinearity we are able to select the best model.

Statistical Analysis

To start, we manipulate the dataset to eliminate missing values and update the response variable with 1 for "Yes," or a successful campaign, and 0 for "No," or an unsuccessful one. We also train the model with 80% of the data and reserve 20% to compare the predicted values to the actual values with our final selected model.

In order to minimize the set of predictors, we generate a Lasso model using 10-fold Cross Validation on the entire dataset and identify the significant variables.

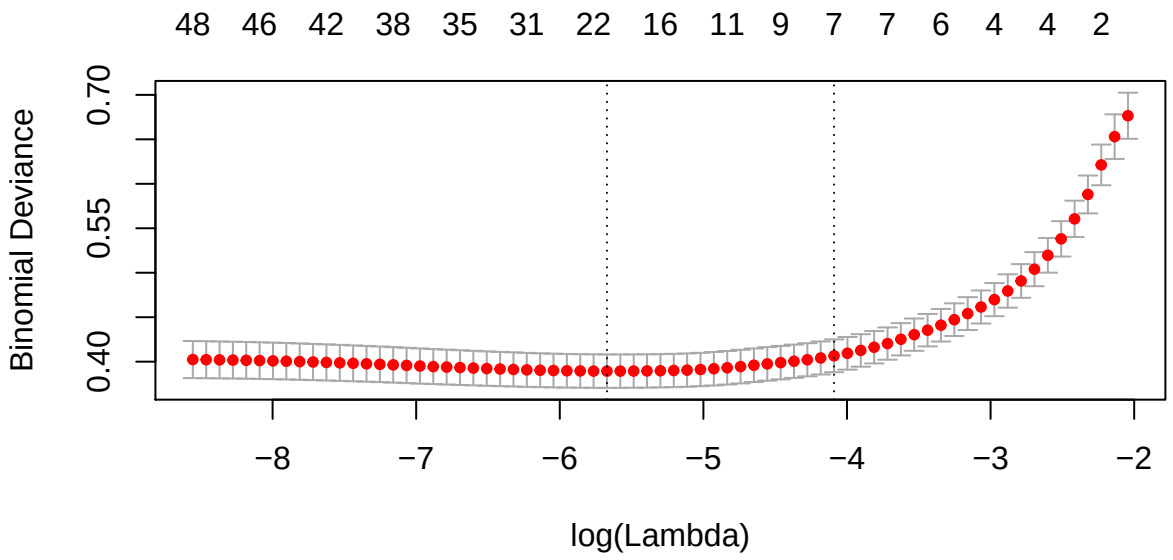
```
lassoMod <- cv.glmnet(x = x, y = y, alpha = 1, nfolds = 10, family = "binomial")
```

But first, let's analyze the percentage of deviance explained and the selection of lambda in the standard Lasso model. One criterion for selecting the optimal value of lambda with a penalized regression is to examine a plot of the deviance against the range of lambda and select lambda when deviance is minimized.

Deviance measures how close the model comes to perfection. It is a measure of goodness of fit, where the smaller the deviance, the better. As previously mentioned, we will later calculate the R^2 from the deviance in the logistic regression model, as an analogy to the linear model.

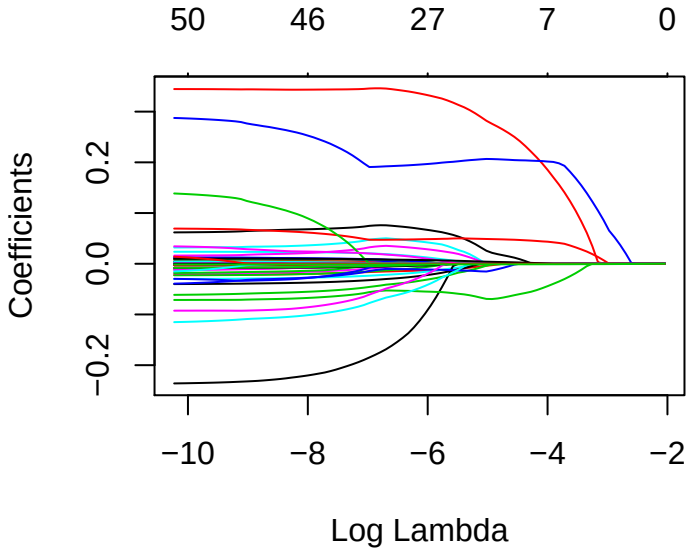
The optimal lambda is found via cross-validation and the resulting models can be compared with various measures. With the next plot we can conclude which lambda has the lowest deviance.

```
plot(lassoMod, label = TRUE, xvar = "lambda")
```

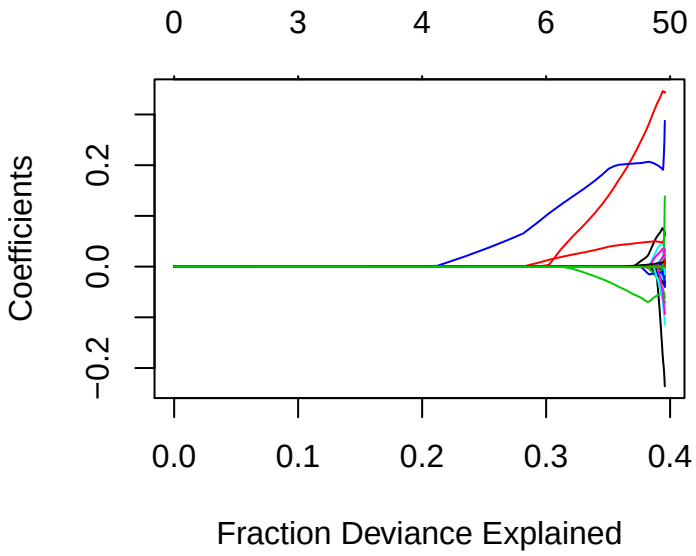


In the plots below we can visualize the coefficients vs. log of lambda and the coefficients vs. the fraction of deviance explained. In the first one we can see how many attributes we have with each value of lambda. In the second one we can see that we can explain 30% of the deviance with just 3 attributes.

```
lassoMod1 <- glmnet(x = x, y = y, alpha = 1)
plot(lassoMod1, label = TRUE, xvar = "lambda")
```



```
plot(lassoMod1, label = TRUE, xvar = "dev")
```



We then look at the sparse coefficient matrix to determine which variables are significant. The nonsignificant variables are penalized with zero coefficients, so we are interested in the nonzero coefficients.

```
predict(lassoMod, type = "coefficients", s = lassoMod$lambda.1se)
```

```
## 53 x 1 sparse Matrix of class "dgCMatrix"
##                                          1
## (Intercept)                   47.974275356
## age                                    .
## jobblue-collar                         .
## jobentrepreneur                        .
## jobhousemaid                           .
## jobmanagement                          .
## jobretired                             .
## jobself-employed                       .
## jobservices                            .
## jobstudent                             .
## jobtechnician                          .
## jobunemployed                          .
## jobunknown                             .
## maritalmarried                         .
## maritalsingle                          .
## maritalunknown                         .
## educationbasic.6y                      .
## educationbasic.9y                      .
## educationhigh.school                   .
```

```
## educationilliterate           .
## educationprofessional.course  .
## educationuniversity.degree    .
## educationunknown              .
## defaultunknown                .
## housingunknown                .
## housingyes                    .
## loanunknown                   .
## loanyes                       .
## contacttelephone              .
## monthaug                      .
## monthdec                      .
## monthjul                      .
## monthjun                      .
## monthmar                      1.226448474
## monthmay                     -0.351366521
## monthnov                      .
## monthoct                      .
## monthsep                      .
## day_of_weekmon                .
## day_of_weekthu                .
## day_of_weektue                .
## day_of_weekwed                .
## duration                      0.003736585
## campaign                      .
## pdays                         .
## previous                      0.133592825
## poutcomenonexistent           .
## poutcomesuccess               1.073576821
## emp.var.rate                 -0.041389839
## cons.price.idx                .
## cons.conf.idx                 .
## euribor3m                     .
## nr.employed                  -0.009981545
```

Now we can use those significant variables as identified by Lasso to create and train a logistic regression model on the training dataset.

```
lassoModglm <- glm(y ~ month+duration+previous+poutcome+emp.var.rate+nr.employed,
           data = train, family = "binomial")
summary(lassoModglm)
```

```
##
## Call:
## glm(formula = y ~ month + duration + previous + poutcome + emp.var.rate +
##     nr.employed, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -4.8301  -0.2738  -0.1737  -0.1295   2.9562
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         54.9604058 10.3373974   5.317 1.06e-07 ***
## monthaug             0.8663637  0.3710878   2.335 0.019561 *
## monthdec             0.4439696  0.7761547   0.572 0.567314
## monthjul             0.7405351  0.3862663   1.917 0.055217 .
## monthjun             1.0530544  0.3566396   2.953 0.003150 **
## monthmar             2.4993008  0.5461141   4.577 4.73e-06 ***
## monthmay            -0.4004478  0.3380964  -1.184 0.236247
## monthnov            -0.1656893  0.4024321  -0.412 0.680545
## monthoct             0.3437212  0.4948450   0.695 0.487304
## monthsep             0.3019569  0.5058194   0.597 0.550530
## duration             0.0050574  0.0002944  17.177  < 2e-16 ***
## previous             0.7546672  0.3343718   2.257 0.024010 *
## poutcomenonexistent  2.2965054  1.1715008   1.960 0.049959 *
## poutcomesuccess      2.9573845  0.8713818   3.394 0.000689 ***
## emp.var.rate        -0.1853310  0.0904517  -2.049 0.040467 *
## nr.employed         -0.0120108  0.0020012  -6.002 1.95e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1999.2  on 2945  degrees of freedom
## Residual deviance: 1115.4  on 2930  degrees of freedom
## AIC: 1147.4
##
## Number of Fisher Scoring iterations: 6
```

To quantify the model fitness, we look at the R^2, AIC, and multicollinearity.

```
r2glm(lassoModglm)
```

```
## [1] 0.4420971
```

```
AIC(lassoModglm)
```

```
## [1] 1147.386
```

```
vif(lassoModglm)
```

```
##                  GVIF Df GVIF^(1/(2*Df))
## month        2.039299  9        1.040383
## duration     1.272090  1        1.127870
## previous     6.099206  1        2.469657
## poutcome     6.439411  2        1.592984
## emp.var.rate 4.080366  1        2.019992
## nr.employed  4.995113  1        2.234975
```

As a comparison, we generate the logistic regression model on the full set of variables.

```
mod <-glm(y ~ ., data = train, family = "binomial")
r2glm(mod)
```

```
## [1] 0.456357
```

```
AIC(mod)
```

```
## [1] 1190.877
```

As you can see, the R^2 and AIC are both higher in this model compared the previous model; this is most likely due to the inclusion of more predictor variables in the model.

As an interesting comparison to the model with significant variables selected by the Lasso model, we use `stepAIC` to select the best model from the one with all variables, on the Training dataset.

```
modAIC <- stepAIC(mod, trace = 0)
summary(modAIC)
```

```
##
## Call:
## glm(formula = y ~ contact + month + duration + pdays + previous +
##     poutcome + cons.conf.idx + nr.employed, family = "binomial",
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.6378  -0.2685  -0.1740  -0.1305   3.2431
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          7.160e+01  7.142e+00  10.025  < 2e-16 ***
## contacttelephone    -6.743e-01  2.458e-01  -2.743  0.00608 **
## monthaug             1.103e-01  4.373e-01   0.252  0.80088
## monthdec            -9.842e-02  8.079e-01  -0.122  0.90305
## monthjul             3.316e-01  3.980e-01   0.833  0.40475
## monthjun             1.047e+00  3.777e-01   2.772  0.00556 **
## monthmar             2.606e+00  5.677e-01   4.591 4.41e-06 ***
## monthmay            -4.729e-01  3.503e-01  -1.350  0.17705
## monthnov            -4.931e-01  4.155e-01  -1.187  0.23529
## monthoct            -2.917e-01  5.482e-01  -0.532  0.59464
## monthsep            -4.480e-01  5.405e-01  -0.829  0.40721
## duration             5.053e-03  2.938e-04  17.201  < 2e-16 ***
## pdays               -1.215e-01  6.453e-02  -1.883  0.05970 .
## previous             7.442e-01  3.488e-01   2.134  0.03287 *
## poutcomenonexistent  1.225e+02  6.378e+01   1.921  0.05469 .
## poutcomesuccess      2.315e+00  9.355e-01   2.474  0.01334 *
## cons.conf.idx        5.640e-02  1.983e-02   2.844  0.00446 **
## nr.employed         -1.447e-02  1.374e-03 -10.533  < 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: 1999.2  on 2945  degrees of freedom
## Residual deviance: 1103.0  on 2928  degrees of freedom
## AIC: 1139
##
## Number of Fisher Scoring iterations: 6
```

As you can see, the R^2 is closer to the Lasso model and the AIC is lower than all previous models. However, we can see some instances of high multicollinearity in this model, especially with pdays.

```
r2glm(modAIC)
```

```
## [1] 0.4483156
```

```
AIC(modAIC)
```

```
## [1] 1138.954
```

```
vif(modAIC)
```

```
##                      GVIF Df GVIF^(1/(2*Df))
## contact       1.740746e+00  1        1.319373
## month         4.863973e+00  9        1.091858
## duration      1.258621e+00  1        1.121883
## pdays         6.845456e+04  1      261.638229
## previous      6.234734e+00  1        2.496945
## poutcome      1.087729e+05  2       18.160599
## cons.conf.idx 2.041608e+00  1        1.428849
## nr.employed   2.315431e+00  1        1.521654
```

We now have three models trained on 80% of the dataset, so as another statistic on the fitness of the models we can predict the response variable for the remaining Test dataset, and calculate the accuracy of each model. Here accuracy is calculated by creating a Confusion Table with the Test predictions compared to actual values and then finding the ratio of correct to incorrect classifications.

```
modPred <- predict(mod, newdata = test.x, type = "response")
modlassoPred <- predict(lassoModglm, newdata = test.x, type = "response")
modAICPred <- predict(modAIC, newdata = test.x, type = "response")
```

Table 1: Top 3 Models for Fitness and Accuracy

Model	Description	R-Squared	AIC	Accuracy
mod	GLM Model with all Variables	0.4563	1190.877	91.86%
lassoModglm	GLM Model with Variables Selected by Lasso	0.4421	1147.386	92.13%
modAIC	GLM Model Selected by stepAIC	0.4483	1138.954	92.13%

Another important point to make is that this dataset is skewed, with only 10.6% in the "No" category. The trivial model of setting all new data to the "No" classification would have 89.3% accuracy, so we need to make sure our model is more accurate than that. Fortunately our most successful model has 92.13% accuracy, so this is indeed a useful model.

Conclusions

Three final models have been selected using logistic regression and the Test data partition has been used to measure the accuracy of each model. We select the best model based on accuracy (calculated using the confusion table), best fit (R^2 and AIC), and low multicollinearity. Even though the model with all variables has the highest R^2, we attribute this to the inclusion of more predictors than the other models. We have found two very good models (lassoModglm, and modAIC) that perform better than the trivial model with an accuracy of 92.13%. The best model based on AIC and accuracy is the GLM model selected by `stepAIC`, however this model also has high multicollinearity.

Therefore it has been useful to use the Lasso method to select the most relevant attributes for our model, as this is the best model overall. We can say with 92.13% certainty that this model will accurately predict the success of the bank telemarketing campaign, with Month, Duration, Previous, Previous Outcome, Employee Variability Rate, and Number Employed as the significant predictor variables.

References

Hastie, Trevor and Qian, Junyang. Glmnet Vignette, Stanford. June 26, 2014, https://web.stanford.edu/~hastie/glmnet/glmnet_alpha.html.

Jeff. "Bank Telemarketing (Moro Et Al.)." *Kaggle*, 15 June 2017, www.kaggle.com/gobert/bank-telemarketing-moro-et-al.

[Moro et al., 2014] S. Moro, P. Cortez and P. Rita. A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems, Elsevier, 62:22-31, June 2014.

Portugués, Eduardo García. *Notes for Predictive Modeling.* 21 Jan. 2018, bookdown.org/egarpor/PM-UC3M/.