Predicting Customer Response with Boosting: Logit and Gradient Descent Models

Patrick Sweeney

02/02/2019

Background

Predicting human behaviour has a rich history. Many theorists have attempted to model the complex, nonlinear and dynamic nature of humans with varying degrees of success.

Nascent behavioural theorists of the 18th and 19th centuries favoured qualitative models and tended to avoid mathematics (Dewey, 1938; Kuhn, 1962). Whether they questioned the utility of reductionist quantitative models is unknown, but it is worth noting that the two godfathers of modern psychology described themselves as having 'infamously low capability for visualising spatial relationships' (Freud) and 'never having dreamt of adding anything to mathematics.' (Jung) (Young-Bruehl, 1992; Evans, 1964).

In the 20th century, this aversion to formal mathematic rigour in the social sciences changed and quantitative modelling breathed new excitement into the fields of economics, anthropology, sociology and psychology. This trend has continued into the present day, where quantitative behaviour models informed by theory are the norm (Abraham & Hassanien, 2009: Cioffi-Revilla, 2014).

Since the 1990s, an analogous paradigm shift has occurred in the fields of statistical learning, data mining and machine learning. The recent development of powerful statistical methods has exploded into the social sciences and has far reaching implications for the prediction of human behaviour in commercial contexts (Breiman, 2001).

Data analytics methods within the context of business can be can be broadly categorised into three distinct types: descriptive, predictive and prescriptive. In this context, predictive does not refer to an accurate forecast of the future, but rather an estimate of some output variable given a change in one or many input variables. Similarly, prescriptive modelling uses observed data to provide an estimation of a potential outcome likelihood as a basis for decision making (Lilien, 2013; Katsov, 2017).

For use in both the predictive and prescriptive domains, statistical learning is unique in that it requires few theory-driven a priori assumptions, allowing it to flourish in business contexts where causal inference methods such as classical econometric modelling may have failed (Grigsby, 2016; Hastie & Tibshirani, 2008).

Executive Summary

Dataset Description

This report describes the application of several statistical learning methods to a dataset of 45,211 observations of 17 variables. The dependent variable y is a binary outcome response indicating whether the client has subscribed to a Portuguese bank's bank term deposit. Predictive features include the customer's age, job, marital status, education, bank balance and housing loan status, as well as relevant information about their response previous marketing campaigns.

Project Goal

The classification model is a market response model intended to maximise the impact and conversion lift of a new direct marketing campaign to the bank's customers. Success will be measured by lift over random chance mailing.

Key Steps

- Wrangling: conversion of factors to dummy variables
- Exploration: summary statistics, feature correlation analysis and visualisation
- Modeling: data stratification, model specification, training and testing
- Evaluation: accuracy & AUC

Analysis

Data Cleaning

Although the bank_full dataset contains no missing values, many class variables were converted to dummies to permit analysis. This was using the fastDummies package, however extraneous 'No' outcome features were deleted to avoid perfect multicollinearity.

```
#Count missing values
missing <- sapply(bank_full, function(x) sum(is.na(x)))
missing <- t(missing)
```

Table 1: Missing Values

age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

```
#Convert classes to dummies for use as needed
bankdummies <- dummy_cols(bank_full, select_columns = c("job", "poutcome", "marital",
                                                        "education", "default",
                                                        "housing", "loan", "month", "y"))
```

Data Exploration

Summary statistics describe the basic distribution and features of the dataset, providing the foundation and context for further analysis.

```
#Summary statistics
summary <- describe(bankdummies)
summarytrimmed <- summary %>%
select(mean, min, max, sd, se)
summaryround <- round(summarytrimmed, digits = 2)
```

```
#Find nummber of rows and columns in dataset
dimensions <- dim(bank_full)
dimensions
```

```
#Find feature correlations with outcome variable
corrs <- cor(bankdummies, use = "complete.obs")
corrs <- as.data.frame(corrs)
ycorrs <- corrs %>% select(y_yes)
ycorrsround <- round(ycorrs, digits = 2)
```

Table 3: Feature Correlations with y

	y_yes
age	0.03
balance	0.05
day	-0.03
duration	0.39
campaign	-0.07
pdays	0.10
previous	0.09
job_management	0.03
job_technician	-0.01
job_entrepreneur	-0.02
job_blue-collar	-0.07
job_unknown	0.00
job_retired	0.08
job_admin.	0.01
job_services	-0.03
job_self-employed	0.00
job_unemployed	0.02
job_housemaid	-0.02
job_student	0.08
poutcome_unknown	-0.17
poutcome_failure	0.01
poutcome_other	0.03
poutcome_success	0.31
marital_married	-0.06
marital_single	0.06
marital_divorced	0.00
education_tertiary	0.07
education_secondary	-0.04
education_unknown	0.01
education_primary	-0.04
default_yes	-0.02
housing_yes	-0.14
loan_yes	-0.07
month_may	-0.10
month_jun	-0.02
month_jul	-0.03
month_aug	-0.01
month_oct	0.13
month_nov	-0.01
month_dec	0.08
month_jan	-0.01
month_feb	0.04
month_mar	0.13
month_apr	0.07
month_sep	0.12
y_yes	1.00

```r
#Find  count and proportion of y outcomes
proptable <- bankdummies %>% freq(y_yes)
proptable
```

Table 4: Count and Proportion of y outcomes

item	count	percent	cum_count	cum_percent
0	39922	0.8830152	39922	0.8830152
1	5289	0.1169848	45211	1.0000000

```r
#Find number of classes
agescount <- length(unique(bank_full$age))
jobcount <- length(unique(bank_full$job))
maritalcount <- length(unique(bank_full$marital))
educationcount <- length(unique(bank_full$education))
classes <- c("Age Count" = agescount, "Job Count" = jobcount,
             "Marital Count" =  maritalcount, "Education" = educationcount)
classes
```

Table 5: Number of Classes

	x
Age Count	77
Job Count	12
Marital Count	3
Education	4

```r
#Find highly correlated features
library(corrplot)
correlationMatrix <- cor(bankdummies)
highlyCorrelated <- findCorrelation(correlationMatrix, cutoff=0.6, names = TRUE)
highlyCorrelated
```

Table 6: Features with Multicollinearity > 0.6

x
poutcome_unknown
pdays
education_tertiary
marital_single

Visualisation

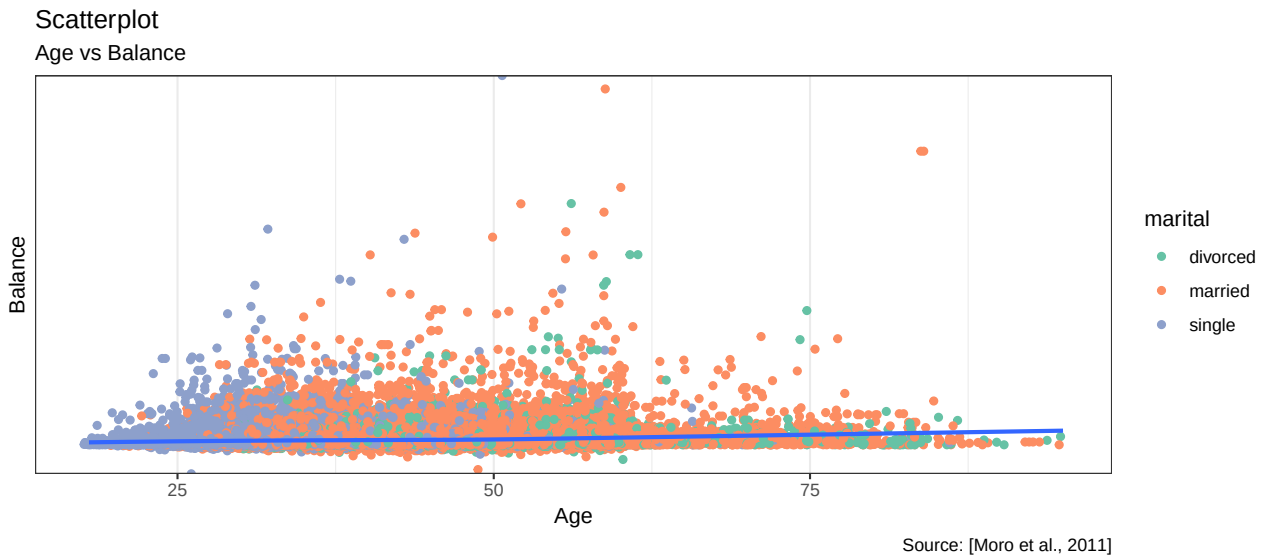
```
options(scipen=999)  # turn-off scientific notation like 1e+48
  theme_set(theme_bw())  # pre-set the bw theme.

  #Duration vs y
  bankdummies$loan_yes <- as.factor(bankdummies$loan_yes)
  gg1 <- ggplot(bankdummies, aes(x=duration, y=y_yes)) +
    geom_jitter(aes(col=loan_yes)) +
    labs(subtitle="Campaign Outcome vs Last Contact Duration",
         y="Campaign Outcome",
         x="Last Contact Duration (secs)",
         title="Scatterplot",
         caption = "Source: [Moro et al., 2011]")+
         scale_color_brewer(palette="Set2")
```


```
  #Poutcome Success vs y
  gg2 <- ggplot(bankdummies, aes(x=poutcome_success, y=y_yes)) +
    geom_jitter(aes(col=loan_yes)) +
    labs(subtitle="Campaign Outcome vs Previous Campaign Outcome",
         y="Campaign Outcome",
         x="Previous Campaign Outcome",
         title="Scatterplot",
         caption = "Source: [Moro et al., 2011]")+
           scale_color_brewer(palette = "Set2")
```

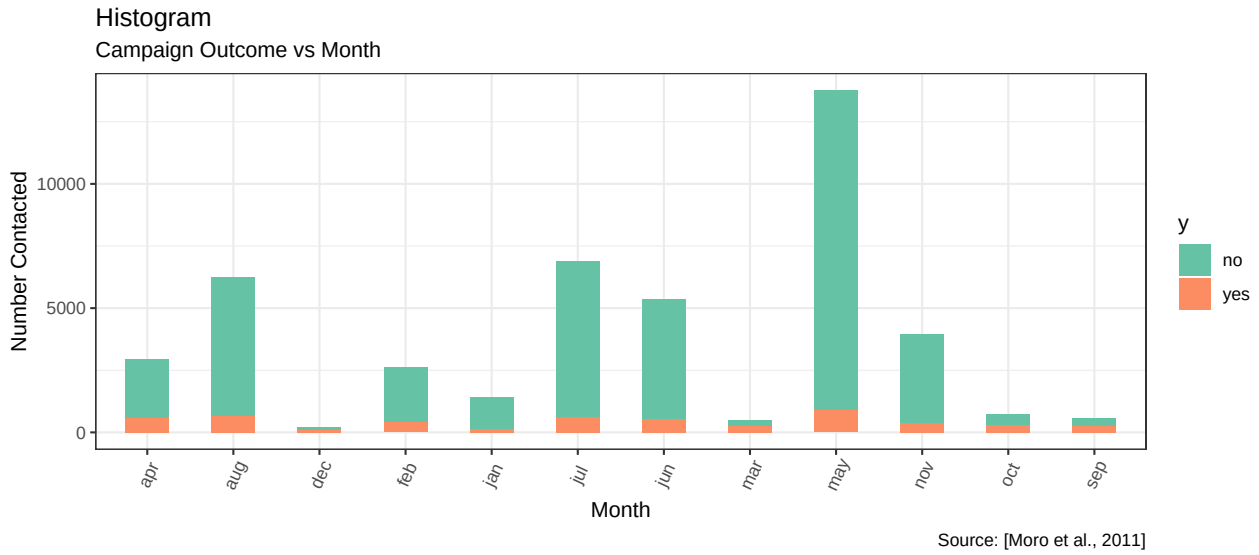

```
#Pdays vs y
gg3 <- ggplot(bank_full, aes(x=pdays, y=y)) +
  geom_jitter(aes(col=month)) +
  scale_y_discrete() +
  labs(subtitle="Campaign Outcome vs Days Since Last Contact",
       y="Campaign Outcome",
       x="Last Contact (days)",
       title="Scatterplot",
       caption = "Source: [Moro et al., 2011]")+
       scale_color_brewer(palette="Set3")
```


```
#Age vs Balance
gg4 <- ggplot(bank_full, aes(x=age, y=balance)) +
  geom_jitter(aes(col=marital)) +
  geom_smooth(method = "loess", se = F) +
  scale_y_discrete() +
  labs(subtitle="Age vs Balance",
       y="Balance",
```

```
        x="Age",
        title="Scatterplot",
        caption = "Source: [Moro et al., 2011]")+
        scale_color_brewer(palette="Set2")
```




```
#Monthly vs y
  gg5 <- ggplot(bank_full, aes(month))+
  geom_bar(aes(fill=y), width = 0.5) +
    theme(axis.text.x = element_text(angle=65, vjust=0.6)) +
    labs(subtitle="Campaign Outcome vs Month",
         y="Number Contacted",
         x="Month",
         title="Histogram",
         caption = "Source: [Moro et al., 2011]")+
         scale_fill_brewer(palette="Set2")
```



```
#Job vs y
gg6 <- ggplot(bank_full, aes(job)) +
  geom_bar(aes(fill=y), width = 0.5) +
  theme(axis.text.x = element_text(angle=65, vjust=0.6)) +
  labs(subtitle="Campaign Outcome vs Job",
       y="Number Contacted",
       x="Job",
       title="Histogram",
       caption = "Source: [Moro et al., 2011]")+
       scale_fill_brewer(palette="Spectral")
```

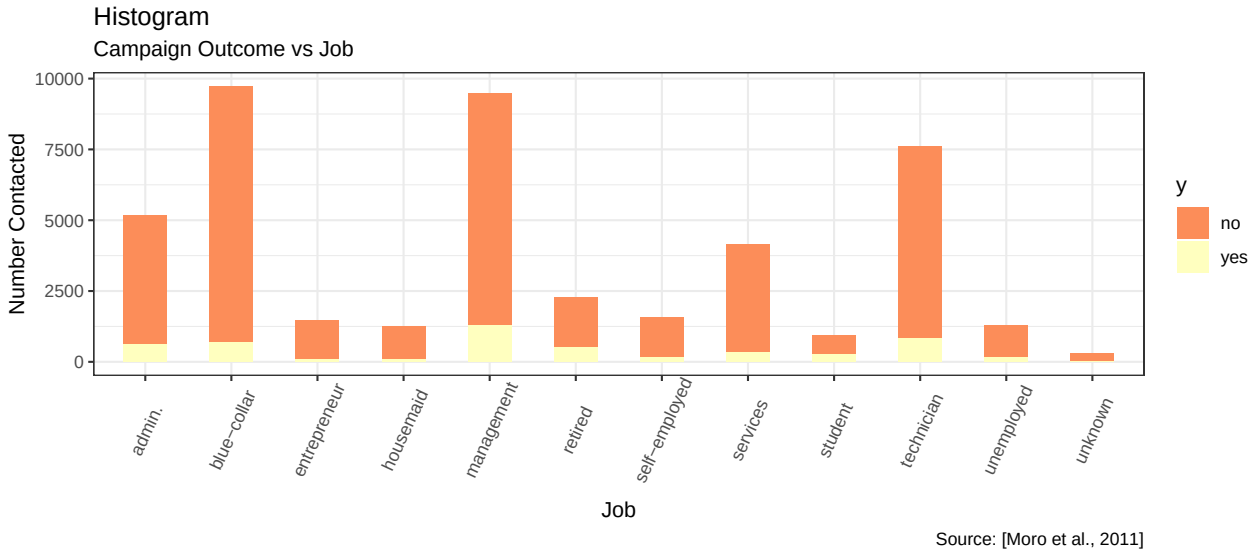
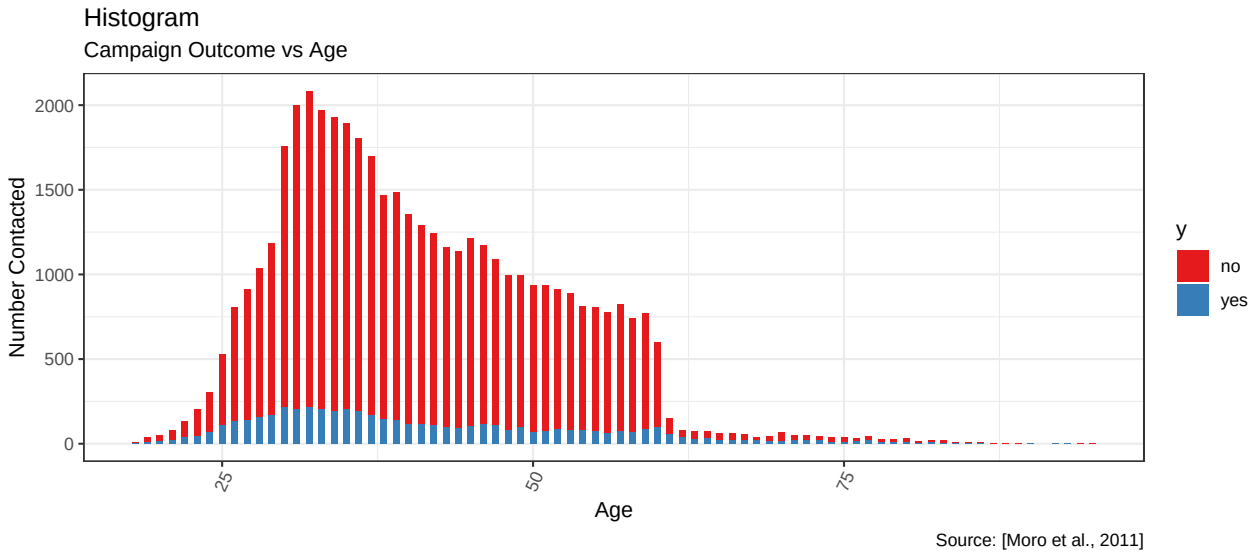


Table 7: Outcomes by Job

item	count	percent	cum_count	cum_percent
blue-collar	9732	0.2152573	9732	0.2152573
management	9458	0.2091969	19190	0.4244542
technician	7597	0.1680343	26787	0.5924886
admin.	5171	0.1143748	31958	0.7068634
services	4154	0.0918803	36112	0.7987437
retired	2264	0.0500763	38376	0.8488200
self-employed	1579	0.0349251	39955	0.8837451
entrepreneur	1487	0.0328902	41442	0.9166353
unemployed	1303	0.0288204	42745	0.9454558
housemaid	1240	0.0274270	43985	0.9728827
student	938	0.0207472	44923	0.9936299
unknown	288	0.0063701	45211	1.0000000

```
#Job vs y
gg7 <- ggplot(bank_full, aes(age)) +
  geom_bar(aes(fill=y), width = 0.5) +
  theme(axis.text.x = element_text(angle=65, vjust=0.6)) +
  labs(subtitle="Campaign Outcome vs Age",
       y="Number Contacted",
```

```
        x="Age",
        title="Histogram",
        caption = "Source: [Moro et al., 2011]")+
        scale_fill_brewer(palette="Set1")
```



Insights

Only 11% of customers subscribed to the bank's feature deposit. No individual features show extreme positive correlation with this outcome variable. The most likely recipient of the bank's mailing efforts is a 30 - 40 year old blue collar or management worker. Unsurprisingly, customers who have not been contacted in two years are highly unlikely to subscribe, and customers who have previously subscribed are more likely to subscribe.

Modeling Approach

There are several possible approaches to maximising the impact of direct mail campaigns. The traditional analytic solution is logistic regression with evaluation via a confusion matrix and lift charts (Grigsby, 2016). Popular domain-specific solutions include demographic sorting, propensity modelling, RFM (recency, frequency, monetary value), modelling, lifetime value modelling, Markov chain models and choice modelling (Lilien 2013) (Katsov, 2016).

In the marketing context, response models are broadly similar in that they are attempting to maximise an economic optimisation function. This can be expressed formally as follows:

$$s_{opt} = \underset{s \in S}{argmax}\, G(s, D)$$

where D is the data available for analysis, S is the space of actions and decisions, G is an economic model mapping actions and data to the economic outcome and S_opt is the optimal strategy. This general optimisation function can also be made more specific to direct marketing applications:

$$U_{opt} = \underset{U \subseteq P}{argmax}\, G(U)$$

where P is the entire population of consumers, U is the subset reached by the campaign and $G(U)$ is the campaign's expected profit; a function of the targeting model selecting U from P.

One popular category of classification targeting is ensemble-tree methods. Methods such as bagging, random forests and boosting produce multiple decision trees which are then aggregated to yield a single census prediction (Hastie). Boosting can be explained simply as a form of sequential probabilistic prediction by committee. It is particularly popular for its ability to combine several weak learning models into a single strong learning model, overcoming the noise, variance and bias of weak models by iteratively re-weighting each learner's 'opinion' according to its accuracy. The two boosting models detailed below were used in the classification of customer response.

Boosted Logit Models.

The LogitBoost model was formulated by Stanford statisticians Friedman, Hastie & Tibshirani and evolved from the popular adaboost model. All supervised learning algorithms work by defining a loss function and attempting to minimise it. In the case of LogistBoost, the algorithm predicts outcomes by minimising the logistic loss, which theoretically reduces sensitivity to outliers (as compared with Adaboost's exponential loss function). The logistical loss function can be expressed formally:

$$\sum_i log(1 + e^{y_i f(x_i)}$$

Gradient Boosting Models

Gradient boosting was also developed by Friedman, evolving from stochastic gradient descent (SGD). Intuitively speaking, SGD repetitively leverages patterns in the residuals of a weak model to improve it. Once no such pattern exists after many random iterations, the algorithm stops modelling residuals to avoid overfitting. However, where SGD trains a single complex model, gradient boosting trains an ensemble of simple models. The gradient boosting loss function is expressed as:

$$\sum_i (y_i - y_i^p)^2$$

where y_i is the ith target value, y_i^p is the ith prediction. XGB is also known for its speed and performance accuracy, making it an attractive choice for modelling response outcomes.

Data Preparation

Before model training, data must be partitioned into stratified training and test sets.

```
#Sample stratified 10% subset
sample <- stratified(bankdummies, "y_yes", size = 0.1)

#Check that proprtions are representative
prop.table(table(bankdummies$y_yes))
prop.table(table(sample$y_yes))

#Make y outcome a factor
sample$y_yes <- as.factor(sample$y_yes)

#Create test and train datasets
set.seed(123)
indexes <- createDataPartition(sample$y_yes,
                               times = 1,
                               p = 0.7,
                               list = FALSE)
sample.train <- sample[indexes,]
sample.test <- sample[-indexes,]
```

```
#Check stratification proportions of test and train are representative
prop.table(table(sample$y_yes))
prop.table(table(sample.train$y_yes))
prop.table(table(sample.test$y_yes))
```

Table 8: Original Sample Distribution

item	count	percent	cum_count	cum_percent
0	3992	0.8829905	3992	0.8829905
1	529	0.1170095	4521	1.0000000

Table 9: Train Subset Distribution

item	count	percent	cum_count	cum_percent
0	2795	0.8828174	2795	0.8828174
1	371	0.1171826	3166	1.0000000

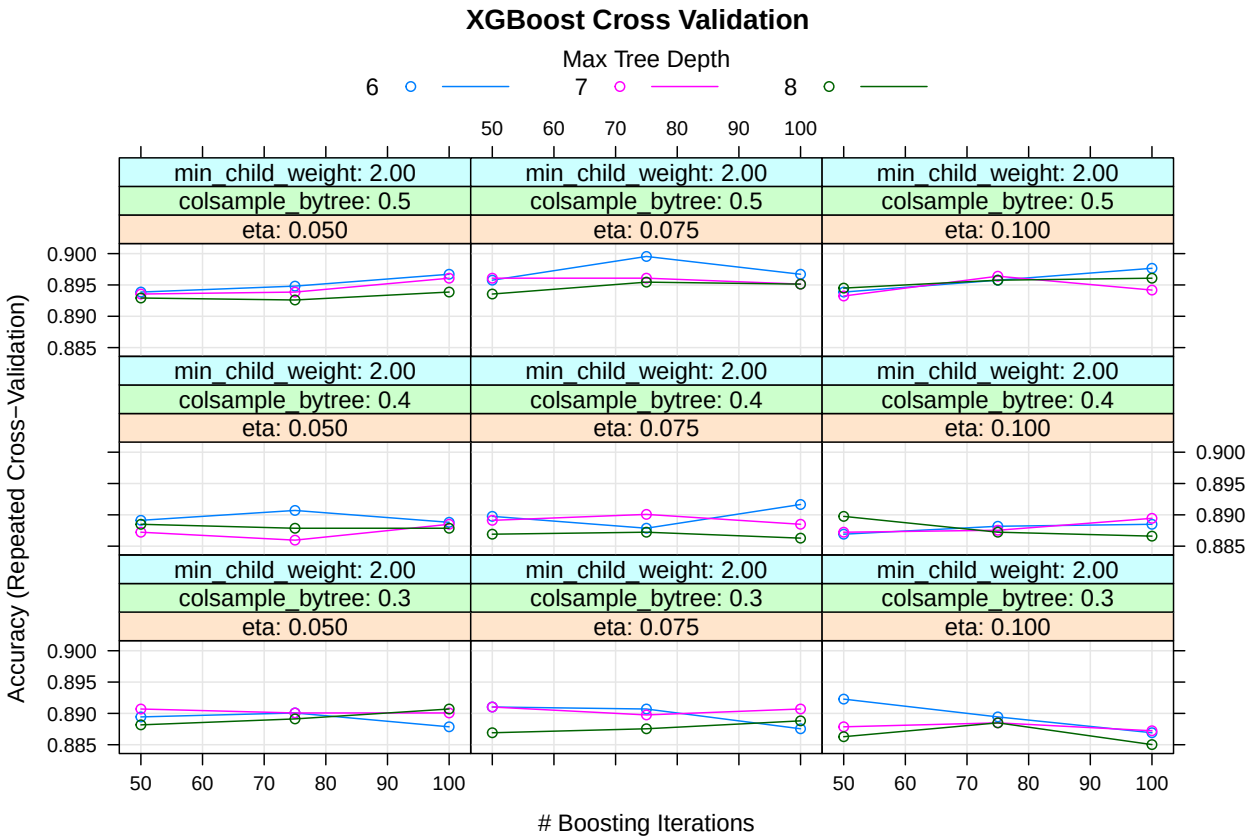
Table 10: Test Subset Distribution

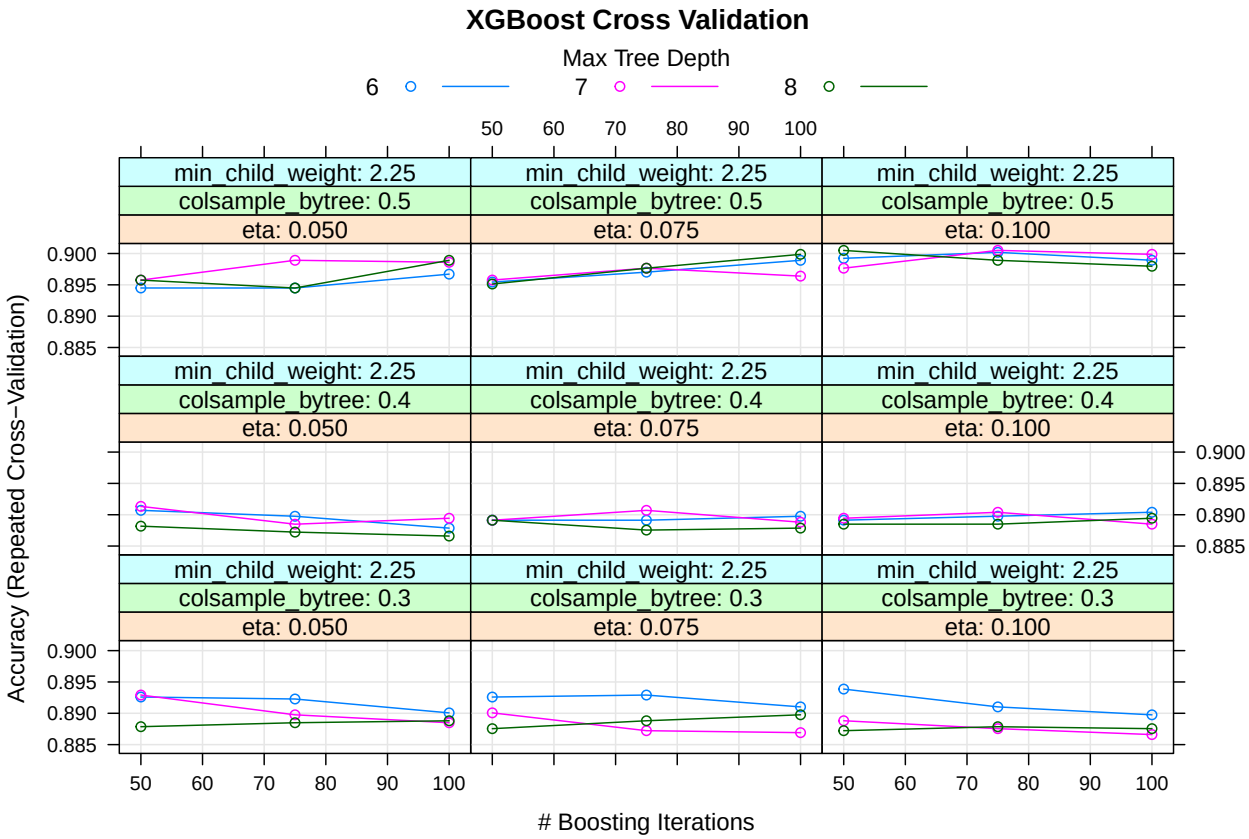
item	count	percent	cum_count	cum_percent
0	1197	0.8833948	1197	0.8833948
1	158	0.1166052	1355	1.0000000

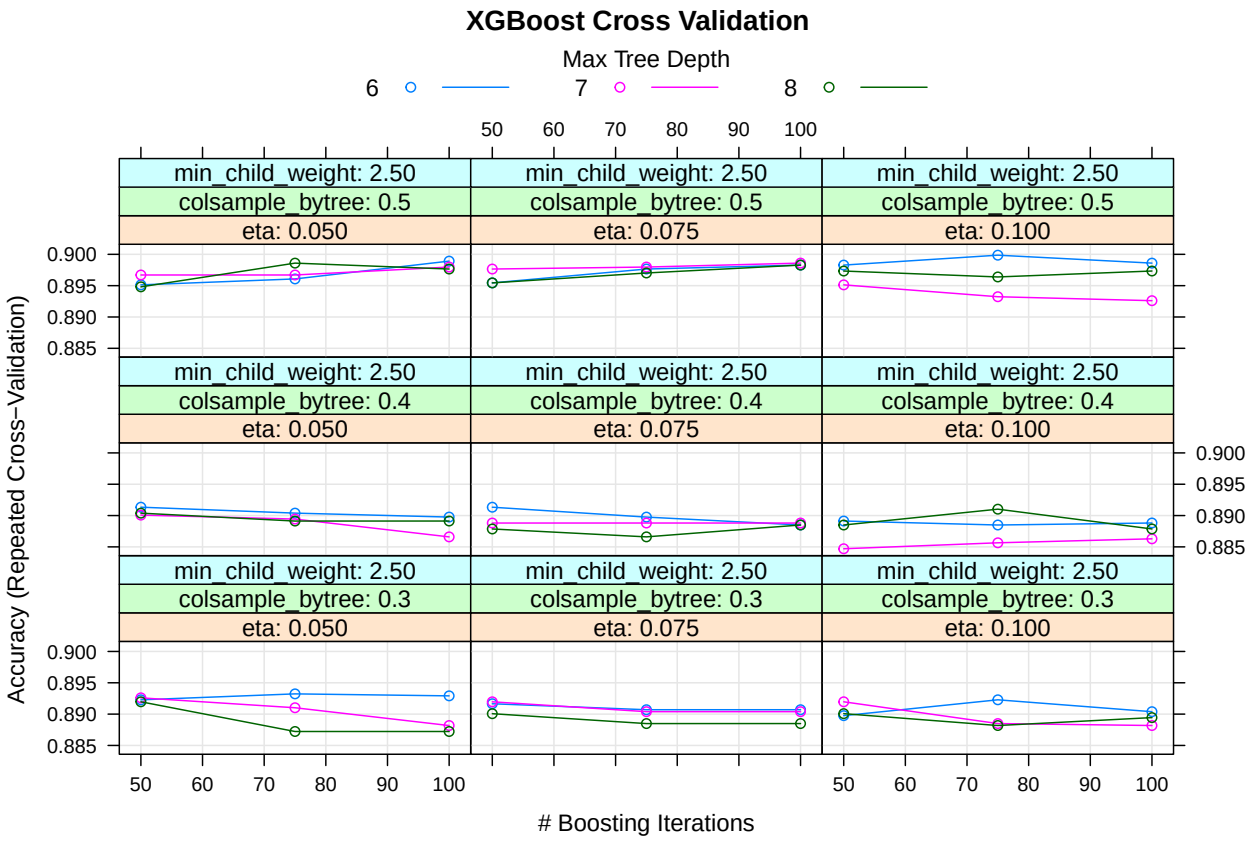
The caret Package

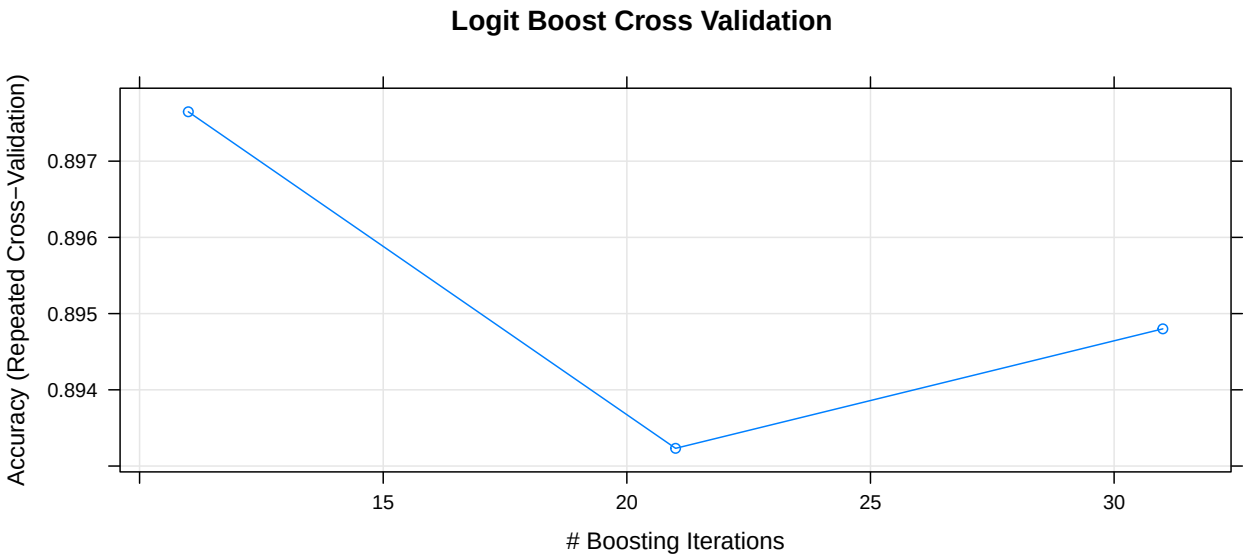
Kuhn's caret package provides standardised syntax for hyper-parameters, cross validation and training of both models (Kuhn, 2013).

```r
#Set parameters: 10-fold cross validation with 2 repeats.
train.control <- trainControl(method = "repeatedcv",
                              number = 10,
                              repeats = 1,
                              verboseIter = TRUE,
                              search = "grid")

#Set up tuning hyperparameters
tune.grid <- expand.grid(eta = c(0.05, 0.075, 0.1),
                         nrounds = c(50, 75, 100),
                         max_depth = 6:8,
                         min_child_weight = c(2.0, 2.25, 2.5),
                         colsample_bytree = c(0.3, 0.4, 0.5),
                         gamma = 0,
                         subsample = 1)

#Fix factors
levels(bankdummies$y_yes) <- list("0" = "1", "1" = "2")

#Create and plot extreme gradient boosting model
xgbmodel <- train(y_yes ~ ., data = sample.train, method = "xgbTree", trControl = train.control, tuneGr

#Compare XGB model with boosted logit model
logitmodel <- train(y_yes ~ ., data = sample.train, method = "LogitBoost", trControl = train.control, ou
```




XGBoost Cross Validation
Max Tree Depth
6
7
8
min_child_weight: 2.25
colsample_bytree: 0.5
eta: 0.050
eta: 0.075
eta: 0.100
colsample_bytree: 0.4
colsample_bytree: 0.3
Accuracy (Repeated Cross−Validation)
Boosting Iterations


XGBoost Cross Validation
Max Tree Depth
6
7
8
min_child_weight: 2.50
colsample_bytree: 0.5
eta: 0.050
eta: 0.075
eta: 0.100
colsample_bytree: 0.4
colsample_bytree: 0.3
Accuracy (Repeated Cross−Validation)
Boosting Iterations




```
#Predictions on test data using XGB model and logit model
predsxgb <- predict(xgbmodel, sample.test)
predslogit <- predict(logitmodel, sample.test)
```

Results

```
# Test data metrics for XGB
cmX <- confusionMatrix(predsxgb, sample.test$y_yes)

Xconfusionmatrix <- as.matrix(cmX, what = "overall")
Xconfusionmatrix2 <- as.matrix(cmX, what = "classes")


#Teset data metrics for boosted logit
cmL <- confusionMatrix(predslogit, sample.test$y_yes)

Lconfusionmatrix <- as.matrix(cmL, what = "overall")
Lconfusionmatrix2 <- as.matrix(cmL, what = "classes")
```

Table 11: XGBoost Confusion Matrix (Test

Accuracy	0.8988930
Kappa	0.3775838
AccuracyLower	0.8815972
AccuracyUpper	0.9144353
AccuracyNull	0.8833948
AccuracyPValue	0.0393226
McnemarPValue	0.0000000

Table 12: LogitBoost Confusion Matrix (Test)

Accuracy	0.8937269
Kappa	0.3097105
AccuracyLower	0.8760863
AccuracyUpper	0.9096378
AccuracyNull	0.8833948
AccuracyPValue	0.1258296
McnemarPValue	0.0000000

```
#Predictions on full set using XGB and boosted logit
bankdummies$y_yes <- as.factor(bankdummies$y_yes)
predsfullxgb <- predict(xgbmodel, bankdummies)
predsfulllogit <- predict(logitmodel, bankdummies)


#Full set data metrics for XGB
levels(bankdummies$y_yes) <- list("0" = "1", "1" = "2")

cmX2 <- confusionMatrix(predsfullxgb, bankdummies$y_yes)
X2confusionmatrix <- as.matrix(cmX2, what = "overall")

#Full data set metrics for boosted logit
cmL2 <- confusionMatrix(predsfulllogit, bankdummies$y_yes)
L2confusionmatrix <- as.matrix(cmL2, what = "overall")
```

Table 13: XGBoost Confusion Matrix (Full Sample)

Accuracy	0.9305258
Kappa	0.0000000
AccuracyLower	0.9281432
AccuracyUpper	0.9328528
AccuracyNull	1.0000000
AccuracyPValue	1.0000000
McnemarPValue	0.0000000

Table 14: LogitBoost Confusion Matrix (Full Sample)

Accuracy	0.9507642
Kappa	0.0000000
AccuracyLower	0.9487296
AccuracyUpper	0.9527406
AccuracyNull	1.0000000
AccuracyPValue	1.0000000
McnemarPValue	0.0000000

```
#Plot ROC curve for XGB
bankdummies$y_yes <- as.numeric(bankdummies$y_yes)
xgbROC <- roc(predsfullxgb, bankdummies$y_yes)
xgbAUC <- auc(xgbROC)
```

```
#Plot ROC curve for boosted logit
bankdummies$y_yes <- as.numeric(bankdummies$y_yes)
logitROC <- roc(predsfulllogit, bankdummies$y_yes)
logitAUC <- auc(logitROC)
```

Conclusion

LogitBoost and XGB have predictive accuracy levels of ~90%. However, the LogitBoost model has an AUC of 0.5, implying that it does not outperform random chance or blanket mailing. Fortunately, the XGB model has a predictively useful AUC of 0.77 which could likely be improved by further tweaking hyperparameters. Given the cost saving implications of effective direct mailiing predictive models, it is clear that machine learning techniques have a rightful place in the marketing mix, g. Additionally, gven the increasing popularity of big data storage and access to larger sample sizes, it is likely that predictive capabilities can only improve.

References

Abraham, A., Hassanien, A. E., & Snášel, V. (Eds.). (2009). Computational social network analysis: Trends, tools and research advances. Springer Science & Business Media.

Breiman, L. (2001). Statistical modeling: The two cultures (with comments and a rejoinder by the author). Statistical science, 16(3), 199-231.

Cioffi-Revilla, C. (2014). Introduction to computational social science. New York.

Dewey, J. (1938). Logic - The theory of inquiry. Read Books Ltd.

Evans, R. I., Jung, C. G., & Jones, E. (1964). Conversations with Carl [Gustav] Jung and reactions from Ernest Jones (Vol. 23). Princeton, NJ, Van Nost.

Grigsby, M. (2016). Advanced Customer Analytics: Targeting, Valuing, Segmenting and Loyalty Techniques. Kogan Page Publishers.

Hastie, T. and Tibshirani, R., & Friedman, J.(2008). The Elements of Statistical Learning; Data Mining, Inference and Prediction. New York: Springer.

Katsov, I. (2017). Introduction to Algorithmic Marketing: Artificial Intelligence for Marketing Operations.

Kuhn, M. and Johnson, K., (2013). Applied predictive modeling (Vol. 26). New York: Springer.

Kuhn, T. S. (1962). The structure of scientific revolutions. University of Chicago Press.

Lilien, G. L., Rangaswamy, A., & De Bruyn, A. (2013). Principles of marketing engineering. DecisionPro.

Miller, T. W. (2015). Marketing data science: modeling techniques in predictive analytics with R and Python. FT Press.

Moro, S., Laureano, R., and Cortez, P. Using Data Mining for Bank Direct Marketing: An Application of the CRISP-DM Methodology. Proceedings of the European Simulation and Modelling Conference - ESM'2011.

Young-Bruehl, E, Freud, S. (1992). Freud on women: A reader. WW Norton & Company.

Zheng, Alice. (2015). "Evaluating machine learning models: a beginner's guide to key concepts and pitfalls."

Table 2: Summary Statistics

	mean	min	max	sd	se
age	40.94	18	95	10.62	0.05
balance	1362.27	-8019	102127	3044.77	14.32
day	15.81	1	31	8.32	0.04
duration	258.16	0	4918	257.53	1.21
campaign	2.76	1	63	3.10	0.01
pdays	40.20	-1	871	100.13	0.47
previous	0.58	0	275	2.30	0.01
job_management	0.21	0	1	0.41	0.00
job_technician	0.17	0	1	0.37	0.00
job_entrepreneur	0.03	0	1	0.18	0.00
job_blue-collar	0.22	0	1	0.41	0.00
job_unknown	0.01	0	1	0.08	0.00
job_retired	0.05	0	1	0.22	0.00
job_admin.	0.11	0	1	0.32	0.00
job_services	0.09	0	1	0.29	0.00
job_self-employed	0.03	0	1	0.18	0.00
job_unemployed	0.03	0	1	0.17	0.00
job_housemaid	0.03	0	1	0.16	0.00
job_student	0.02	0	1	0.14	0.00
poutcome_unknown	0.82	0	1	0.39	0.00
poutcome_failure	0.11	0	1	0.31	0.00
poutcome_other	0.04	0	1	0.20	0.00
poutcome_success	0.03	0	1	0.18	0.00
marital_married	0.60	0	1	0.49	0.00
marital_single	0.28	0	1	0.45	0.00
marital_divorced	0.12	0	1	0.32	0.00
education_tertiary	0.29	0	1	0.46	0.00
education_secondary	0.51	0	1	0.50	0.00
education_unknown	0.04	0	1	0.20	0.00
education_primary	0.15	0	1	0.36	0.00
default_yes	0.02	0	1	0.13	0.00
housing_yes	0.56	0	1	0.50	0.00
loan_yes	0.16	0	1	0.37	0.00
month_may	0.30	0	1	0.46	0.00
month_jun	0.12	0	1	0.32	0.00
month_jul	0.15	0	1	0.36	0.00
month_aug	0.14	0	1	0.35	0.00
month_oct	0.02	0	1	0.13	0.00
month_nov	0.09	0	1	0.28	0.00
month_dec	0.00	0	1	0.07	0.00
month_jan	0.03	0	1	0.17	0.00
month_feb	0.06	0	1	0.23	0.00
month_mar	0.01	0	1	0.10	0.00
month_apr	0.06	0	1	0.25	0.00
month_sep	0.01	0	1	0.11	0.00
y_yes	0.12	0	1	0.32	0.00