Model Results

The logistic regression and mulitple linear regression models willbe combined together to come up with a predicted response for each customer in the test set.

This will be done by first predicting the customer's liklihood to respond to the promotion via the logistic regression model. After determining response liklihood, the mulitple linear regression will be applied to predict how *much* a customer will purchase after the promotion.

In summary: E(logtargamt) = P(logtargamt > 0) * E(logtargamt | logtargamt > 0)

Logistic Regression

The logisitc model was developed by fitting a binomial generalized linear model to the test data set using a collection of predictors both natural and engineered. The predictors used in the final model are as follows:

- avgNetOrder: average amount of money each customer ever spend
- sumQty: The total quantity each customer ordered
- recency: no. of days since the last order
- frequency: number of orders
- tof: each customer's time on file
- amountPer: total past purchase amount/time on file
- oneMonth: dummy variable- 1 if the customer placed order within one month prior to 08/01/2014
- threeMonth: dummy variable- 1 if the customer placed order within three month prior to 08/01/2014
- overYear: dummy variable- 1 if the customer placed order over one year prior to 08/01/2014
- sumQtyPerTof: frequnecy of placing orders

Multiple Regression

The mulitple linear regression was produced by several iterations of feature engineering and model variation. Ultimately, a best model was obtained via stepwise regression allowed to move in both forward and backward directions. The final predictors used are as follows:

- recency: no. of days since the last order
- frequency: number of orders
- amount:total past purchase amount in euros
- amountPer: total past purchase amount/time on file
- sumQty: The total quantity each customer ordered
- oneMonth: dummy variable- 1 if the customer placed order within one month prior to 08/01/2014

Synthesis

These two models will be brought together to form predictions on the expected amount purchased for customers in the test set.

We will narrow down the full test data to just the needed variables, while also filtering out several of the significant outliers (with more than 1000 orders)

```
predictions <- test.full %>%
  select(id, avgNetOrder, sumQty, recency, frequency, tof, amount, maxNet, maxPrice,
         amountPer, oneMonth, threeMonth, sixMonth, oneYear, overYear, sumQtyPerTof,
         logtargamt) %>% filter(tof > 0, sumQty < 1000, amountPer < 15)
```

The predictions given by the logistic regression are biased towards responders becuase the training data was oversampled to better balance responses. This bias will be corrected below

```r
p2 <- predict(log.best, newdata = predictions, type = 'response')
m <- (nrow(filter(train.rebalanced, respond == 1)) / nrow(train.rebalanced)) /
  (nrow(filter(train.full, respond == 1)) / nrow(train.full))
p1 <- exp(-log(m) + log(p2 / (1 - p2)))/(1 + exp(-log(m) + log(p2 / (1 - p2))))

predictions$probRespond <- p1
summary(predictions$probRespond)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.002883 0.025395 0.034343 0.043129 0.049550 0.925039
```

The multiple regression can now be applied to the set to predict the amount purchased.

```r
predictions$predLogtargamt <- predict(ml.best, newdata = predictions)
summary(predictions$predLogtargamt)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -0.7522  3.1812  3.2487  3.3153  3.3587 19.1409
```

Lastly, we need to include the new customers in the test set. These customers do not have any data on file and thus we will not be able to form sensible predictions for their response. Instead, we will keep use the average values for new customers: 52.9% likely to respond and spending an average logtargamt of 1.753316

```r
newCustomers <- test.full %>%
  select(id, avgNetOrder, sumQty, recency, frequency, tof, amount, maxNet, maxPrice,
         amountPer, oneMonth, threeMonth, sixMonth, oneYear, overYear, sumQtyPerTof,
         logtargamt) %>% filter(tof == 0)
new.avg.response.rate <- nrow(filter(train.full, tof == 0, logtargamt > 0)) /
  nrow(filter(train.full, tof == 0))
new.avg.logtargamt <- mean(filter(train.full, tof == 0)$logtargamt)
newCustomers$probRespond <- new.avg.response.rate
newCustomers$predLogtargamt <- new.avg.logtargamt
predictions <- rbind(predictions, newCustomers)
```

Now that the predictions for all customers are in place, they can be combined to yield an overall expected logtargamt based on liklihood to respond and predicted logtargamt.

```r
predictions$scaledLogtargamt <- predictions$probRespond * predictions$predLogtargamt
summary(predictions$scaledLogtargamt)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -0.67824  0.08412  0.11377  0.15183  0.16532  4.91407
```

The predicted logtargamts must now be transformed back to a normal, predicted amount in euros by performing the inverse of the original log transformation ($ln(amt + 1)$)

```r
predictions$predTargamt <- exp(predictions$scaledLogtargamt) - 1
predictions$targamt <- exp(predictions$logtargamt) - 1
summary(predictions$predTargamt)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
##  -0.49249   0.08776   0.12050   0.19752   0.17977 135.19315
```

```r
SSEP <- sum((predictions$predTargamt - predictions$targamt)^2); SSEP
```

```
## [1] 2534917
```

To help find significant error contributors, we can look at the Sq. Error term for each prediction

```
bad.apples <- predictions %>% mutate(sq.error = (predTargamt - targamt) ^ 2) %>% arrange(-sq.error) %>%
```

A separate measure of performance will evaluate the financial value in our model. This will be done by looking at the top 500 prospects as identified by predicted targamt (amount spent after promotion), and then examining how much these top prospects *actually* spent following the promotion.

```
top500 <- predictions %>%
  arrange(-predTargamt) %>%
  head(500)

payoff <- sum(top500$targamt); payoff
```

```
## [1] 6672.738
```

```
actual.top.500 <- predictions %>% arrange(-targamt) %>% head(500)
percentPayoff <- payoff / sum(actual.top.500$targamt); percentPayoff
```

```
## [1] 0.2468175
```

Now we can examine the optimal number of customers to target for the promotion by looking at the short term profit margins. If the profit margin is 25% of the `targamt` spent, and cost of marketing to a customer is 1 euro, then we will maximize total profit for marketing to the top `x` customers ordered by `predTargamt`.

We will do this by ordering the customers by descending predicted targamt and then calculating the cumulative profit where $profit = 0.25 * targamt - 1$

```
predictions$profit <- predictions$targamt * 0.25 - 1
opt.profit <- max(cumsum(arrange(predictions, -predTargamt)$profit)); opt.profit
```

```
## [1] 1446.723
```

This profit is obtained by marketing to the top x customers where x is calculated below

```
opt.customers <- which.max(cumsum(arrange(predictions, -predTargamt)$profit)); opt.customers
```

```
## [1] 1516
```

This new optimal marketing strategy will be used to calculate a new payoff and percent payoff

```
top.prospects <- predictions %>%
  arrange(-predTargamt) %>%
  head(opt.customers)

opt.payoff <- sum(top.prospects$targamt); opt.payoff
```

```
## [1] 11850.89
```

```
actual.opt.prospects <- predictions %>% arrange(-targamt) %>% head(opt.customers)
opt.percentPayoff <- opt.payoff / sum(actual.opt.prospects$targamt); opt.percentPayoff
```

```
## [1] 0.3424145
```

In summary:

```
list('SSEP' = SSEP,
     'Payoff' = payoff,
     'Percent Payoff' = percentPayoff,
     'Optimal Profit' = opt.profit,
     'Optimal Target Customers' = opt.customers,
```

```r
     'Optimal Payoff' = opt.payoff,
     'Optimal Percent Payoff' = opt.percentPayoff)
```

```
## $SSEP
## [1] 2534917
##
## $Payoff
## [1] 6672.738
##
## $`Percent Payoff`
## [1] 0.2468175
##
## $`Optimal Profit`
## [1] 1446.723
##
## $`Optimal Target Customers`
## [1] 1516
##
## $`Optimal Payoff`
## [1] 11850.89
##
## $`Optimal Percent Payoff`
## [1] 0.3424145
```