Churn Analysis with logistic regression and random forests

Daniel Moscoe

May, 2021

https://rpubs.com/dmoscoe/768184

Data collection

Kaggle Customer Churn Prediction 2020

```
k1.dat <- read_csv("https://raw.githubusercontent.com/dmoscoe/SPS/main/churn_train.csv")
str(k1.dat)
```

```
## spec_tbl_df [4,250 x 20] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ state                        : chr [1:4250] "OH" "NJ" "OH" "OK" ...
##  $ account_length               : num [1:4250] 107 137 84 75 121 147 117 141 65 74 ...
##  $ area_code                    : chr [1:4250] "area_code_415" "area_code_415" "area_code_408" "area_code_415" ...
##  $ international_plan           : chr [1:4250] "no" "no" "yes" "yes" ...
##  $ voice_mail_plan              : chr [1:4250] "yes" "no" "no" "no" ...
##  $ number_vmail_messages        : num [1:4250] 26 0 0 0 24 0 0 37 0 0 ...
##  $ total_day_minutes            : num [1:4250] 162 243 299 167 218 ...
##  $ total_day_calls              : num [1:4250] 123 114 71 113 88 79 97 84 137 127 ...
##  $ total_day_charge             : num [1:4250] 27.5 41.4 50.9 28.3 37.1 ...
##  $ total_eve_minutes            : num [1:4250] 195.5 121.2 61.9 148.3 348.5 ...
##  $ total_eve_calls              : num [1:4250] 103 110 88 122 108 94 80 111 83 148 ...
##  $ total_eve_charge             : num [1:4250] 16.62 10.3 5.26 12.61 29.62 ...
##  $ total_night_minutes          : num [1:4250] 254 163 197 187 213 ...
##  $ total_night_calls            : num [1:4250] 103 104 89 121 118 96 90 97 111 94 ...
##  $ total_night_charge           : num [1:4250] 11.45 7.32 8.86 8.41 9.57 ...
##  $ total_intl_minutes           : num [1:4250] 13.7 12.2 6.6 10.1 7.5 7.1 8.7 11.2 12.7 9.1 ...
##  $ total_intl_calls             : num [1:4250] 3 5 7 3 7 6 4 5 6 5 ...
##  $ total_intl_charge            : num [1:4250] 3.7 3.29 1.78 2.73 2.03 1.92 2.35 3.02 3.43 2.46 ...
##  $ number_customer_service_calls: num [1:4250] 1 0 2 3 3 0 1 0 4 0 ...
##  $ churn                        : chr [1:4250] "no" "no" "no" "no" ...
```

Data collection

Google Trends for wireless carrier names, March 2020. `library(gtrendsR)`

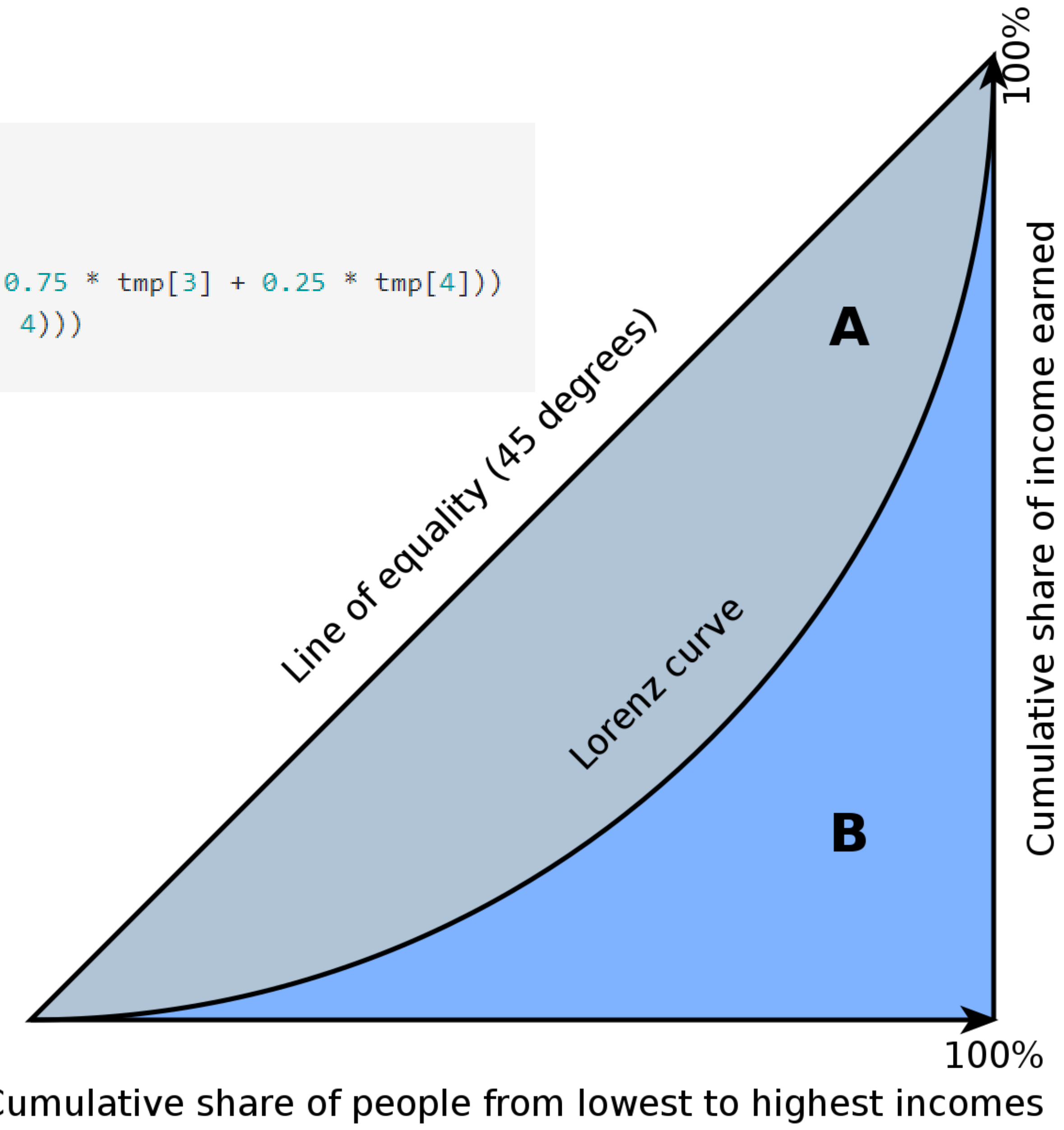
```
gtrends(keyword = gtrends_search_terms[1:4], geo = states[i], time = "2020-03-01 2020-03-30")
```

```
## 'data.frame':    120 obs. of  7 variables:
##  $ date    : POSIXct, format: "2020-03-01" "2020-03-02" ...
##  $ hits    : int  44 20 84 34 36 76 32 38 69 59 ...
##  $ keyword : chr  "att" "att" "att" "att" ...
##  $ geo     : chr  "US-AL" "US-AL" "US-AL" "US-AL" ...
##  $ time    : chr  "2020-03-01 2020-03-30" "2020-03-01 2020-03-30" "2020-03-01 2020-03-30" "2020-03-01 2020-03-30" ...
##  $ gprop   : chr  "web" "web" "web" "web" ...
##  $ category: int  0 0 0 0 0 0 0 0 0 0 ...
```

The Gini Index

```r
ginis <- data.frame("geo" = "x", "gini" = "-1")
for(i in seq(nrow(tmp_query_summaries))) {
  tmp <- sort(as.integer(tmp_query_summaries[i,2:5]))
  tmp.gini <- 1 - ((1/sum(tmp)) * (1.75 * tmp[1] + 1.25 * tmp[2] + 0.75 * tmp[3] + 0.25 * tmp[4]))
  ginis <- rbind(ginis, c(tmp_query_summaries[i,1], round(tmp.gini, 4)))
}
```

- Computed from Google Trends data
- $0 \leq A/(A+B) \leq 1$ (perfect equality to perfect inequality)
- ***Inequality in searches implies inequality in number of customers across providers***



Logistic Regression

- Address class imbalance
- Prune model with backward selection

```
## Call:
## glm(formula = churn ~ total_day_minutes + total_eve_minutes +
##     total_night_minutes + number_customer_service_calls + international_plan +
##     voice_mail_plan, family = binomial(link = "logit"), data = k4_train.dat)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.55164  -0.78148  -0.03148   0.82985   2.67961
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                   -6.060681   0.612370  -9.897  < 2e-16 ***
## total_day_minutes              0.014621   0.001472   9.936  < 2e-16 ***
## total_eve_minutes              0.006237   0.001579   3.951 7.79e-05 ***
## total_night_minutes            0.003483   0.001645   2.117   0.0343 *
## number_customer_service_calls  0.630234   0.061077  10.319  < 2e-16 ***
## international_planTRUE         2.487112   0.255333   9.741  < 2e-16 ***
## voice_mail_planTRUE           -1.356577   0.217122  -6.248 4.16e-10 ***
```

Interpreting coefficients of the logistic regression

$$\frac{p}{1-p} = \sum_{i,j} \exp(\beta_i x_{ij})$$

A unit increase in x_i increases the odds of churn by a *factor* of $\exp(\beta_i)$.

Variable	Estimate	Odds change by factor of
total_day_minutes	0.015	1.015
total_eve_minutes	0.0062	1.006
total_night_ minutes	0.0035	1.004
number_customer_ service_calls	0.6302	1.878
international_plan	2.4871	12.026
voicemail_plan	-1.357	0.257

Logistic Regression Cutoffs

Maximize detection of true positives:
Cutoff = 0.0857

```
k4_train_preds <- ifelse(k4_train_preds >= optimal_cutoff, TRUE, FALSE)
k4_train_preds_table <- table(k3_test.dat$churn, k4_train_preds)
k4_train_preds_table
```

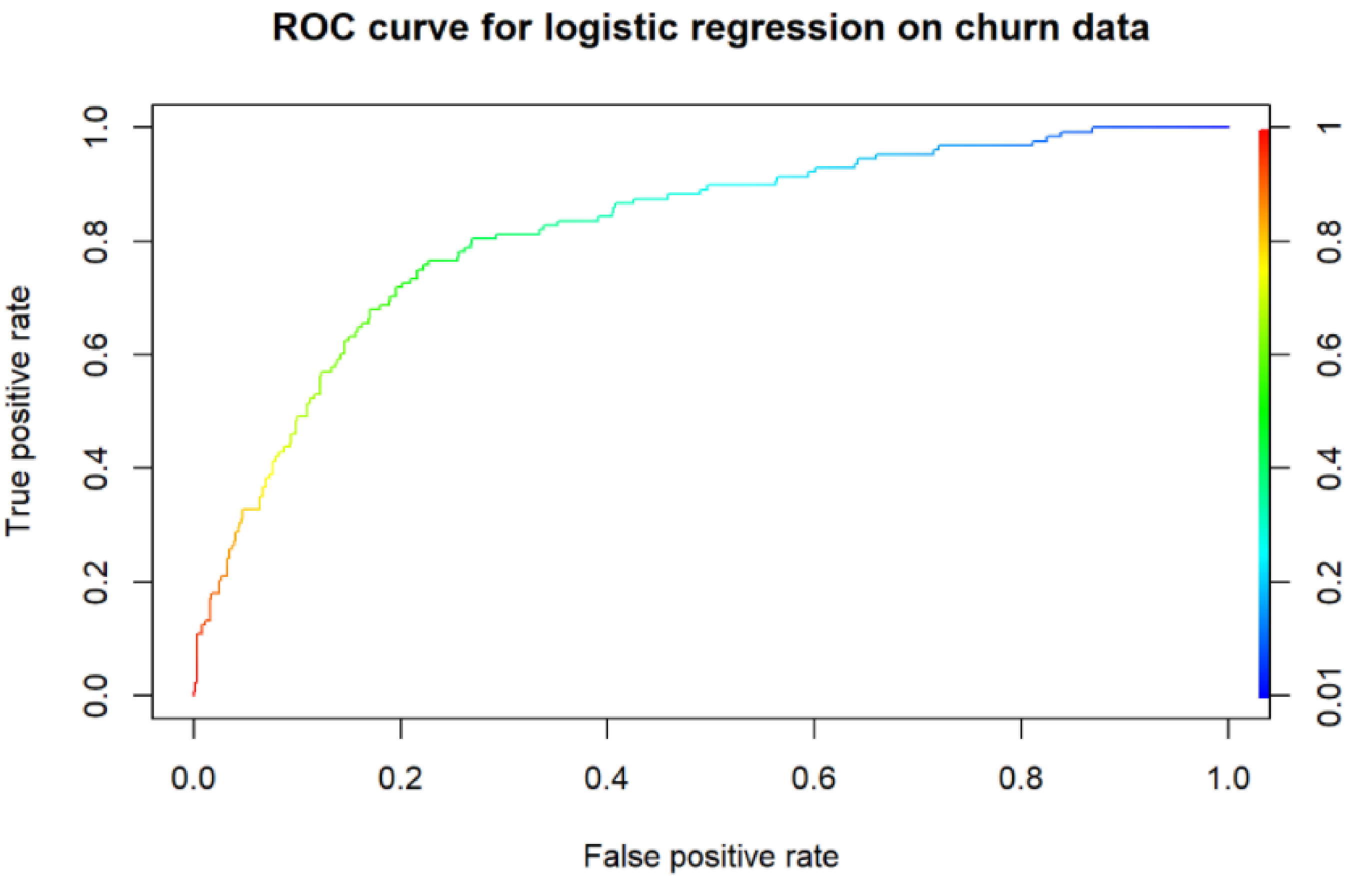
```
##        k4_train_preds
##         FALSE TRUE
##   FALSE    95  627
##   TRUE      0  128
```

Maximize accuracy:
Cutoff = 0.9457

```
k4_train_preds <- ifelse(k4_train_preds >= optimal_cutoff, TRUE, FALSE)
k4_train_preds_table <- table(k3_test.dat$churn, k4_train_preds)
k4_train_preds_table
```

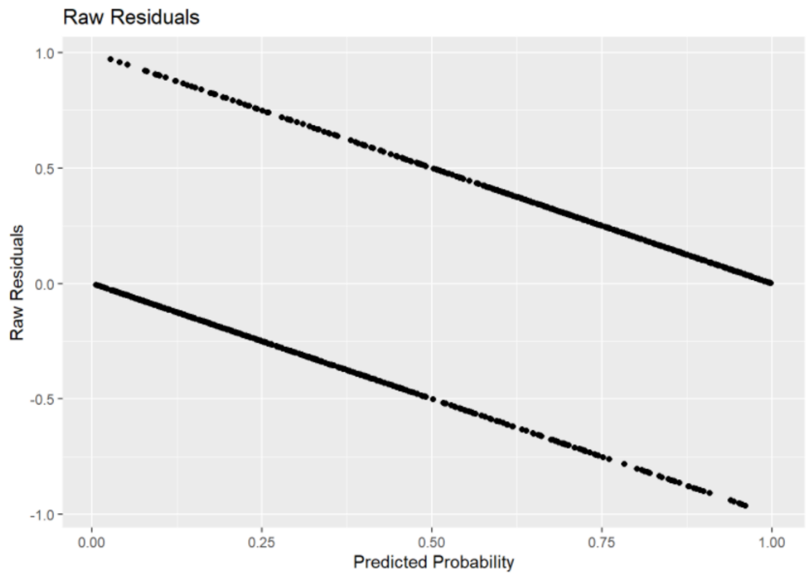
```
##        k4_train_preds
##         FALSE TRUE
##   FALSE   719    3
##   TRUE    114   14
```

```
k4_train_preds <- predict(k4_train.glm, k3_test.dat, type = "response")
pred <- ROCR::prediction(k4_train_preds, k3_test.dat$churn)
perf <- ROCR::performance(pred, "tpr", "fpr")
plot(perf, colorize = TRUE, main = "ROC curve for logistic regression on churn data")
```

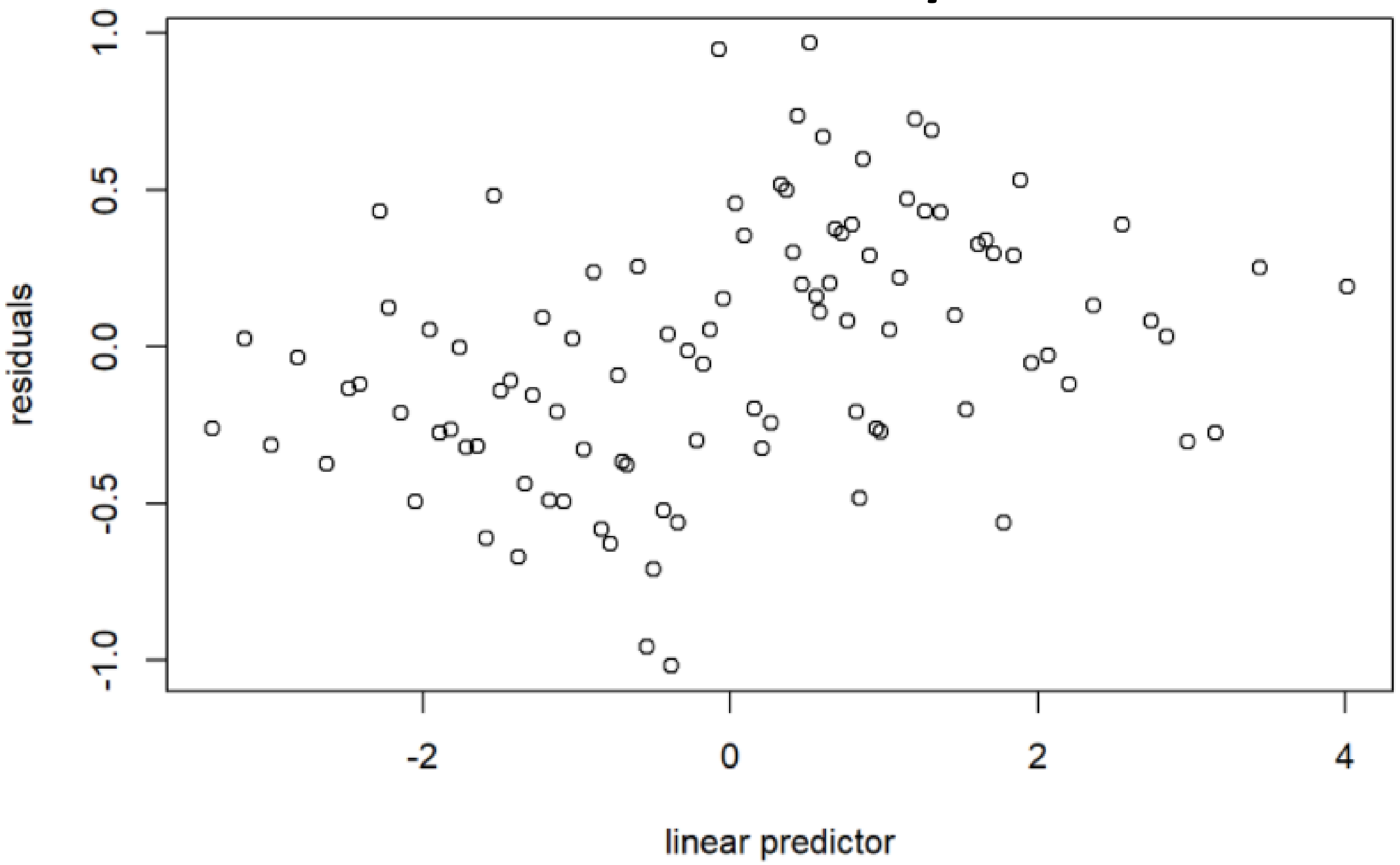


LR Residuals

```
k5_train.dat <- k4_train.dat %>%
mutate("residuals" = residuals(k4_train.glm), linpred = predict(k4_train.glm))
gdf <- group_by(k5_train.dat, cut(linpred, breaks = unique(quantile(linpred, (1:100)/101))))
diagdf <- summarise(gdf, residuals = mean(residuals), linpred = mean(linpred))
plot(residuals ~ linpred, diagdf, xlab = "linear predictor")
```



Binned residuals plot



Tuned Random Forest Model

```
k4_train.for <- randomForest(as.factor(churn) ~ ., k4_train.dat, ntree = 200, mtry = 14, importance = TRUE, proximity = TRU
E)
print(k4_train.for)
```

```
##
## Call:
##  randomForest(formula = as.factor(churn) ~ ., data = k4_train.dat,      ntree = 200, mtry = 14, importance = TRUE, proxim
ity = TRUE)
##                Type of random forest: classification
##                      Number of trees: 200
## No. of variables tried at each split: 14
##
##         OOB estimate of  error rate: 15.85%
## Confusion matrix:
##       FALSE TRUE class.error
## FALSE   386   84   0.1787234
## TRUE     65  405   0.1382979
```

Applying the new model to the test set:

```
k4_test_preds_for <- predict(k4_train.for, k3_test.dat) #predictions on the k3_test.dat data based on the model trained on k
4_train.dat.
confusionMatrix(k4_test_preds_for, as.numeric(k3_test.dat$churn)) #A confusion matrix comparing predicted values for k3_tes
t.dat with actual values.
```

```
##   FALSE TRUE
## 0   603  119
## 1    20  108
```

Choosing a model

Act.↓ Pred. →	Churn	Remain	Pro	Con
LR max true +			***Detects about 100% of churn. Use if cost of churn is much greater than cost of promo.***	***Low accuracy. Detects many false positives. Many non-churners will receive promo.*** acc = 0.26, prec = 0.17, rec = 1, spec = 0.13
Churn	0.15	0		
Remain	0.74	0.11		
LR max acc.			***High accuracy.***	***Essentially equivalent to predicting majority class for all obs's.*** acc = 0.87, prec = 0.83, Rec = 0.13, spec = 0.99
Churn	0.02	0.13		
Remain	0.004	0.85		
Random Forest			***Detects almost all churn. Detects majority of non-churn. High accuracy.***	***Some false positives.*** acc = 0.84, prec = 0.48, Rec = 0.87, spec = 0.84
Churn	0.13	0.02		
Remain	0.14	0.71		

Submission and Description	Private Score	Public Score
sub_set.csv 2 minutes ago by Daniel Moscoe	0.81333	0.80444

Conclusion

- Finding value doesn’t always mean improving accuracy.

- The LR yields “collateral insights.”