STAT 149 - Generalized Linear Models (Spring 2016)
Final Project: US CHESS Membership Retention

Team **Zugzwang**: Amy Lee (60984077) & Kendrick Lo (70984997)

May 5, 2016

The goal of this project was to analyze a data set and predict the probability that a member in good standing of US Chess would have a lapsed membership. The training set consisted of 43,436 observations from a data set of 57,915 records. This document summarizes our modeling approaches and tracks the course of our analysis throughout the project. Detailed process workbooks in R can be found online on Github[1].

We achieved first place on the leaderboard with a log-loss score of 0.52925 on the 50% public test data sample, and first place in the final standings with a score of 0.53861 on the full test data. While we held as high as third place early in the competition using a Random Forest model (based on features sourced from the original data set and accompanied by additional ones generated based on age, membership months, all games played, ratings, and types of tournaments played), we began making headway to first place after drawing inspiration from a blogpost[2] on model ensembling techniques frequently deployed by Kaggle competition winners. Ultimately, our best results were achieved using a "stacking" technique that involved adding the prediction results of our best classifiers as new features to a subset of the original data set, and using a Generalized Additive Model (GAM) as the final stacking model.

A summary of the key models that will be discussed in this report along with the associated log-loss score from Kaggle can be found in tabular format in Appendix A. A plot showing progression of decreasing Kaggle log-loss scores over the course of the competition is provided in Figure 1, along with all the other figures referenced in this report in Appendix B. We will describe the models we tried in the sections below, along with the observations we made that led to the largest gains in performance.

1 Technical Approach

1.1 Exploratory Data Analysis

We began by examining the summary statistics, quality, and distribution of the provided data. The variable names and descriptions of the data can be found in Appendix D. Figure 4 shows that of the 43,436 training observations, about 60% are classified as lapsed memberships. A large portion of the data is missing ranking data, whereas a smaller subset is missing *age* and *sex* (see Figure 8). Histograms of age and membership months in Figures 5 and 6 indicate highly right-skewed

[1] https://github.com/ppgmg/chess
[2] http://mlwave.com/kaggle-ensembling-guide

distributions. We also noted that one member had played 625 tournament games in a year! Figure 7 shows that three regions in particular are much more strongly represented in US Chess than others.

Plots of the lapsed/not lapsed membership distribution of categorical variables in Figure 2 and of quantitative variables in Figure 3 suggest that *hasemail*, *age*, *memmonths*, and ratings-related variables will likely be important predictors for the probability of a lapsed membership.

1.2 Missing Data

To handle the missing data, we inferred a regression-adjusted effect for a predictor's missing values. For categorical variables, we created a new level indicating the value was missing from the original data set. For quantitative variables, we created a new binary feature indicating that the value was missing; in addition, where a value was missing we imputed the mean value of each respective quantitative feature.

The major simplifying assumption behind this technique is that the missing data mechanism is "ignorable" for each variable with missing data. However, given that a summary of the logistic regression model suggested that the *.na* binary indicators were significant, further exploration into potentially more effective approaches for dealing with missing values, such as multiple imputation, would appear warranted. That said, in the interest of time, and given that we experienced a sizable improvement in score with this imputation method in any event, we used the data sets with values imputed as noted earlier in this paragraph (hereinafter the "clean data set") as the basis for further modeling.

1.3 Feature Engineering

We applied domain knowledge to create new features from the existing data to improve the fit of the model. First, we included interactions and higher-order terms on features we believed to have highest relevance to a person's membership status: the number of months since joining US chess (*memmonths*), the number of tournament games played in the past year ($allgames1yr$), and the player's age (*age*). We created new binary variables corresponding to ranges of these predictors. After adding similar first-order interactions to the previous model, we obtained a slight improvement to the Kaggle score.

In addition, for certain models we considered features for the member's percentage of tournaments played that were blitz, medium or regular-speed, the percent change in rating from now vs. 1 and 2 years ago, and the percent change in number of tournament games played in the past year compared to the annual average of the four previous years.

1.4 Cross Validation

We implemented 5-fold cross-validation on our best models to obtain a measure out-of-sample error, which helped prevent overfitting, determine best hyperparameters such as optimial number of trees in a Random Forest given the data set, and better compare different models. We understood that using Kaggle scores as the sole source of feedback would be dangerous since we would be making decisions on the test set, and would thus effectively be training on the test set.

2 Modeling

2.1 Logistic Regression

Multi-collinearity As a first model on the clean data set, we attempted to reduce the number of variables for use in a Generalized Linear Model (GLM). First, we posited that some of the predictors would be nearly linearly related to each other. Though collinearity can cause problems with the estimation of model coefficients, when the goal is prediction and not interpretation, it becomes less critical to remedy the collinearity problems. In any event, in order to demonstrate some of the techniques learned in class, we removed $allgames5yr$ and $r3$ from our "full model" after an examination of computed **variance inflation factors (VIFs)**.

Variable Reduction and Interactions Next, we performed an **analysis of deviance** to further reduce the number of variables, using two different methods: starting from a full model and dropping features one at a time, and starting from an intercept-only model and adding features one at a time. We ended up with the same reduced model, although we note this will not necessarily occur as a matter of course. The resulting model did not yield an improvement in the Kaggle score, but when we added **interactions** to the reduced set of main effects, we were able to obtain a slight improvement as shown in the table in Appendix A.

With respect to interactions, given the high number of predictors even in our reduced model (21 in total), we strategically prioritized certain interactions rather than test all pairwise comparisons and combinations of interactions. We decided to identify the first 7 terms that were added to the previous model (arguably in order of "importance") and added pairwise interactions involving these terms. We then conducted a series of **likelihood ratio tests** (LRTs) to determine which interactions to keep. We added these first-order interactions to both the reduced model, and to the full data set; adding the interactions to the full data set resulted in an improved score.

We also experimented with second-order interactions, but noticed that the incremental improvements over the previous models were negligible, and the residual deviance/degrees of freedom ratio did not more closely approach 1. The fact that the level of improvement seemed to plateau might indicate, among other things, that we are missing some variables needed to better explain the data. Accordingly, for our subsequent model analysis, we settled on the simpler model (i.e. the full model with first-order interactions) as our "best" logistic regression model.

Goodness-of-Fit To evaluate our best logistic regression model, we started with the Hosmer-Lemeshow test. This coarse test has low power, and yet the null hypothesis being rejected suggested that we had a fit problem! We constructed a plot of **Cook's Distance** (see Figure 9); this plot did not reveal anything particularly noteworthy as none of the values had a value greater than 1.

We also constructed several residual plots (see Figures 10 and 11); we noted quite a few observations outside of the (+/- 2) bands, although this might be somewhat expected given the large volume of observations. The binned plots were particularly interesting; it appeared the binned residuals have some pattern to them. We hypothesized we might be missing some higher-order term, or that important predictors were missing from the model. We explore these issues further in the "Generalized Additive Models" and "Feature Engineering" sections that follow.

Interpretation of Coefficients and Confidence Intervals A summary of the final logistic regression model can be found in Appendix C. As an exercise, we interpreted select coefficients:

- **quantitative predictor**: for *age*, an increase of 1 year in age corresponds to a 0.114 increase in the probability of lapsing, on the logit scale, holding all other predictors fixed.
- **categorical predictor**: for *hasemail*, the odds of a person's membership lapsing for members having an e-mail address on file is $\exp(0.88) = 2.41$ times smaller compared to members not having an e-mail address on file, holding all other predictors fixed.
- **first order interaction term**: for $memtypeN : r1$, 0.179 is the additional effect of a 1 unit increase in the transformed regular chess rating from 2 years prior on the probability of lapsing (on the logit scale) for being a normal member compared to being an affiliate member.

The standard errors in the summary can be used to calculate Wald confidence intervals (e.g. for *age*, this may be given by [0.08, 0.15]), or we could have used the $confint$ command in R for greater accuracy (which uses profile likelihoods).

2.2 Generalized Additive Models

A generalized additive model (GAM) is a GLM in which the linear predictor depends linearly on unknown smooth functions of some predictor variables. In general, GAMs allow for some flexibility in modeling the relationship between the response variable and the predictors in the model. For comparison purposes, we left in the first order interactions from our logistic regression model, and built up the GAM. A likelihood ratio test suggested the GAM better fit the data compared to the corresponding GLM, and this result was corroborated by a jump in score on Kaggle.

2.3 Classification Trees

We also experimented with tree-based non-parametric models. We built some basic models for classification trees along the lines of the models constructed in class. This model building included the task of pruning trees using the "1-SE rule", to prevent us from overfitting the training data and hopefully allow for greater generalization (see Figure 12.) An example of the pruned tree using $cp = 0.0013$, a parameter associated with the classification model, is provided in Figure 13. Notably, the accuracy of predictions on the test set for this model seemed poor relative to that of previous models.

2.4 Random Forest

As with other ensemble learning methods, Random Forest models tend to do very well in model fitting where accuracy of fit (and predictive power) is prioritized over model interpretability. While decision trees typically over-fit the training set, random forest models reduce variance by training on subsets of both the training data and the design features and, consequently, usually generalize better to out-of-sample data. We can use the out-of-bag error (OOB), which R provides as output, as a measure of how well the model will perform on unseen data. The Random Forest package in R also provides the facility to construct Variable Importance Plots, an example of which can be found in Figure 14. In one example model fit, the OOB estimate for the error rate obtained

was approximately 27.7%. It was interesting to see that this model placed *r*3 as a very important variable.

Although we obtained a slight improvement in Kaggle score, the good-but-not-exceptional performance of the Random Forest classifier was somewhat disappointing – we had heard that this model class tends to do very well in Kaggle competitions. Nevertheless, this model did turn out to perform better than the previous models we had tried thus far using techniques taught in the course. A summary of our best Random Forest model can be found in Figure 15.

2.5 Neural Networks and Gradient Boosting Models

As we reached the top 3 positions on the leaderboard relatively early on in the competition, it became clear that a little more firepower would be needed if we were to try to win. Our biggest early gain with these other methods came from the *nn* package; a 3-stage neural network (that was surprisingly easy to set up in R) gave a bump in our Kaggle score, which resulted in our first debut at the top of the leaderboard. After being overtaken for the lead, we then moved back into the lead position using an extension to Random Forest, a Gradient Boosting Model (GBM). In contrast to random forests, boosting methods are based on a different, constructive strategy: in boosting, we add new models to construct an ensemble sequentially. At each particular iteration, a new model is trained with respect to the error of the whole ensemble learned so far.

3 Final Models: Putting It All Together

Near the end of competition we tried a number of ensemble methods. First, it was interesting to note how well simply **averaging the predictions** of good models (at the time, our best GLM, GAM, Random Forest, and neural network models) worked. We had anticipated getting a result between the best and worst scores for models we averaged, but we ultimately obtained a score that exceeded that of any model considered individually. Inspired by the Kaggle blogpost cited earlier, we took advantage of the fact that the predictions provided by models of different types were not perfectly correlated; by averaging these non-correlated predictions we benefit from certain error-correction effects, leading to improved predictions.

We also explored "stacking": we obtained four sets of predictions on the training and test sets, using our best GAM, Random Forest, Neural Network, and GBM, respectively. We then treated these four sets of predictions as new features that were combined with the top 5 features from the original data set (*age*, *memmonths*, *r*3 along with *r*3.*na*, *allgames*1*yr*, and *hasemail*) to produce a new feature set. A GAM was then used to train and predict on this set, resulting in final scores of 0.52925 and 0.53861 on the public and private test sets, respectively, and a first place ranking.

In closing, we considered how our analysis might translate into actionable items. In an observational study, we have to be careful about drawing conclusions that certain features might be the **cause** of a lapse in membership. Moreover, the models we built involve complex interactions between multiple predictors, and so the inferred effect of any one predictor may be different depending on what other predictors are in a given model. That said, there seemed to be consensus that, at least, *age* and *hasemail* were important features. It may be worth exploring campaigns to retain younger players who might otherwise allow their membership to lapse in their late teens to early twenties (e.g. allow youth who have been members for some time to continue at a lower rate as an incentive), and certainly, to ensure valid e-mail addresses for members are kept on file.

4 Appendices

A Key Kaggle Submissions

Table 1: Key Kaggle Submissions in Chronological Order

Features	Model Class	Kaggle
best baseline	logistic regression	0.66486
age * *allgames*1*yr*	logistic regression	0.59509
imputed missing values	logistic regression	0.56551
all features with 1st order interactions	logistic regression	0.56513
feature engineering with interactions	logistic regression	0.55723
all features with 1st order interactions	generalized additive model (GAM)	0.54542
pruning parameter (cp) = 0.0013	classification tree	0.57371
3000 trees, tuned model	random forest (RF)	0.54365
3-stage, full feature set	neural network (NN)	0.53759
sklearn	gradient boosting model (GBM)	0.53411
averaging best GLM/NN/RF/GAM	ensemble	0.53115
GAM stacked + 5 top features	ensemble	0.52925

(Log-Loss error)

B Figures

Figure 1: Kaggle Progression plot

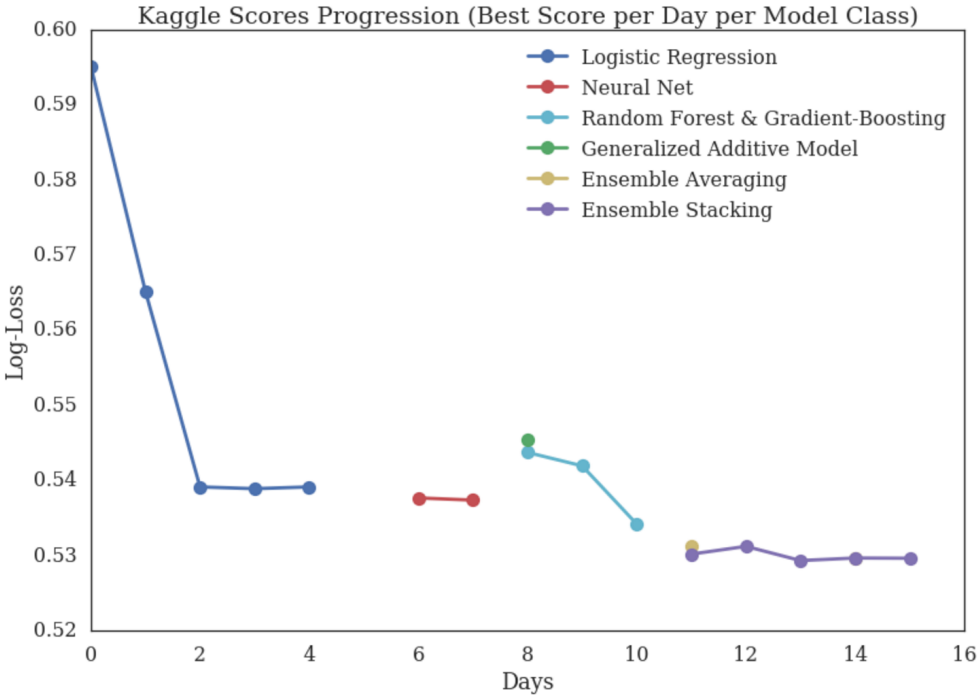

Figure 2: Indicator Variables

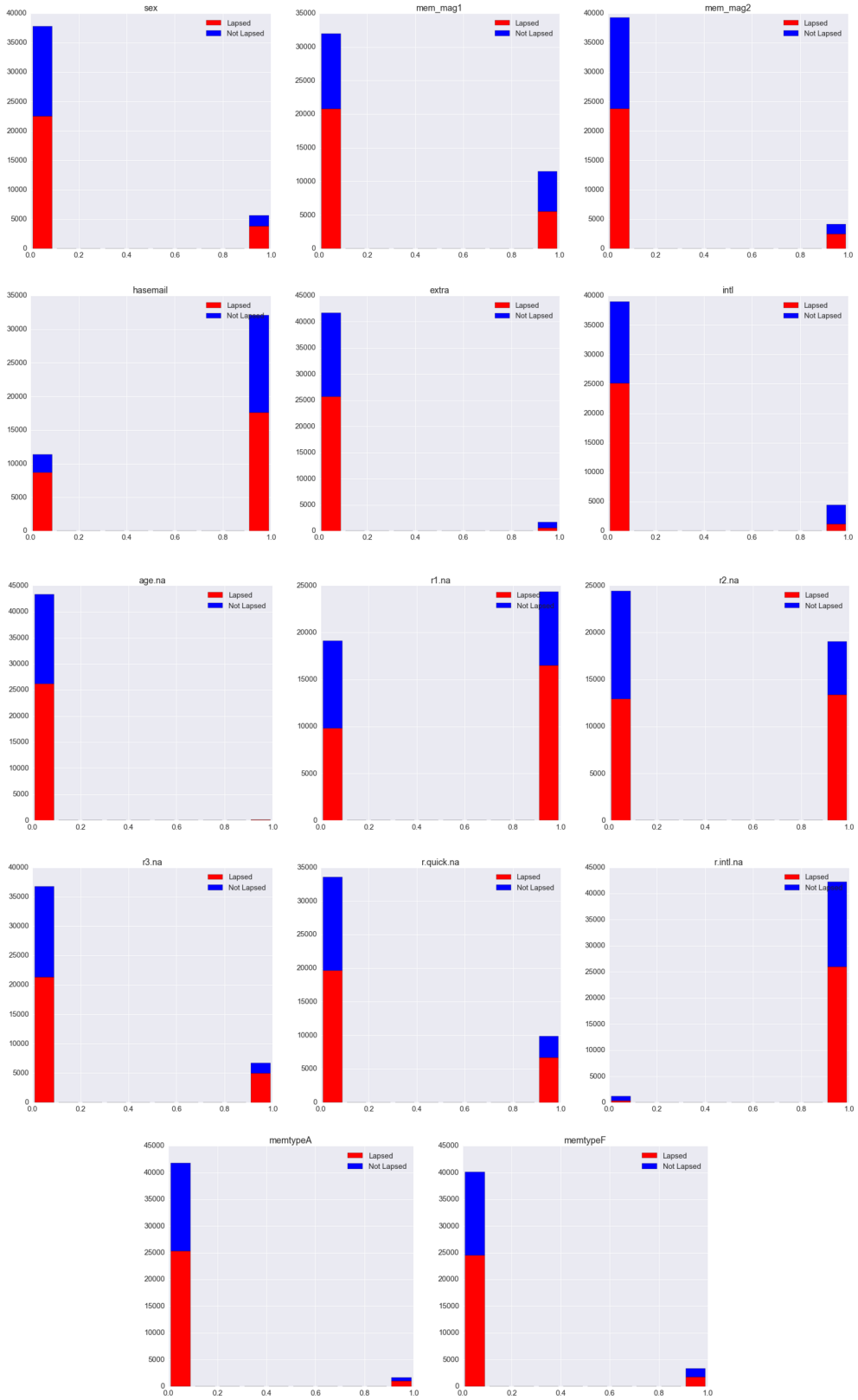

Figure 3: Continuous Variables

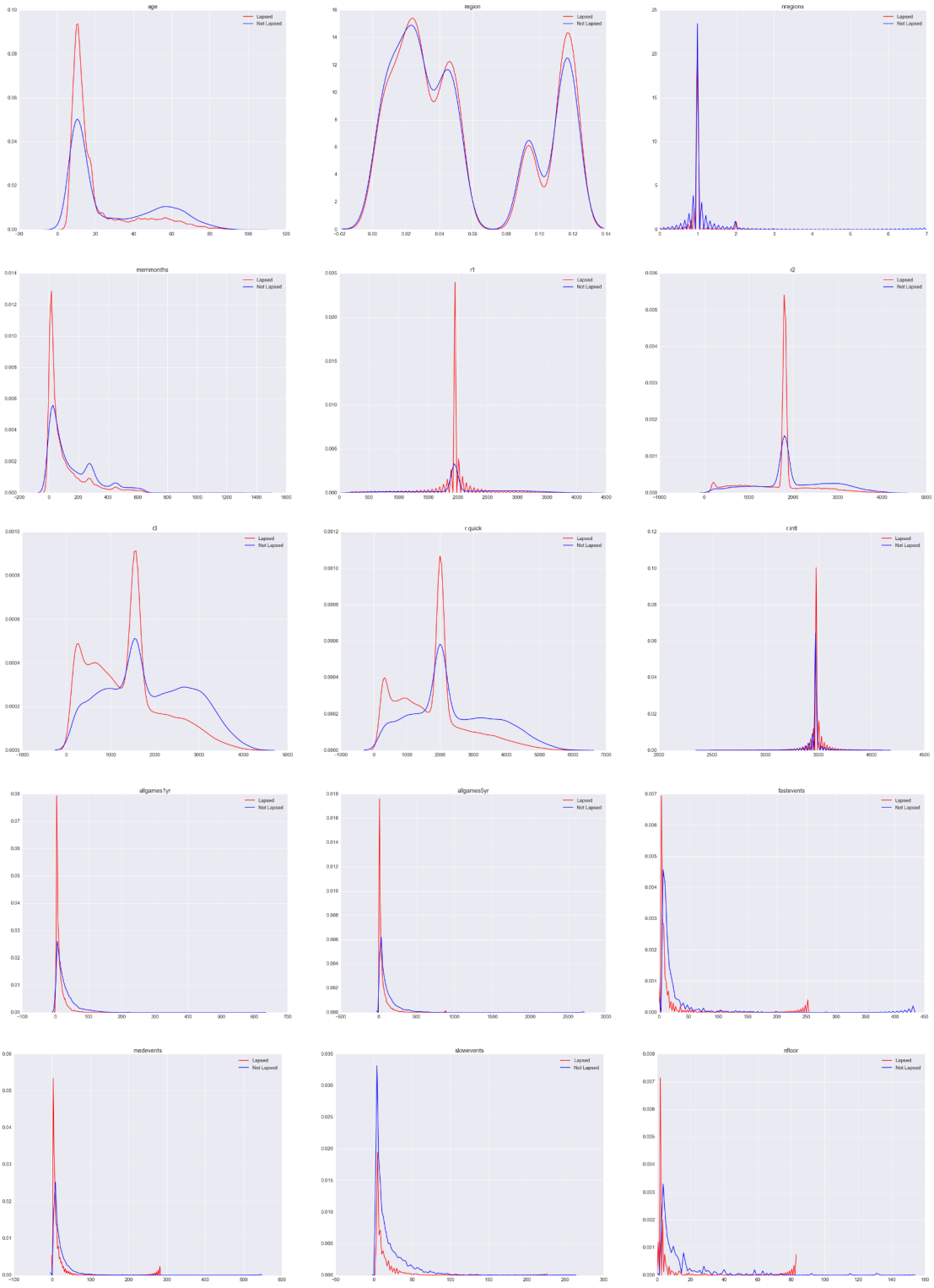

Figure 4: Percentage Lapsed

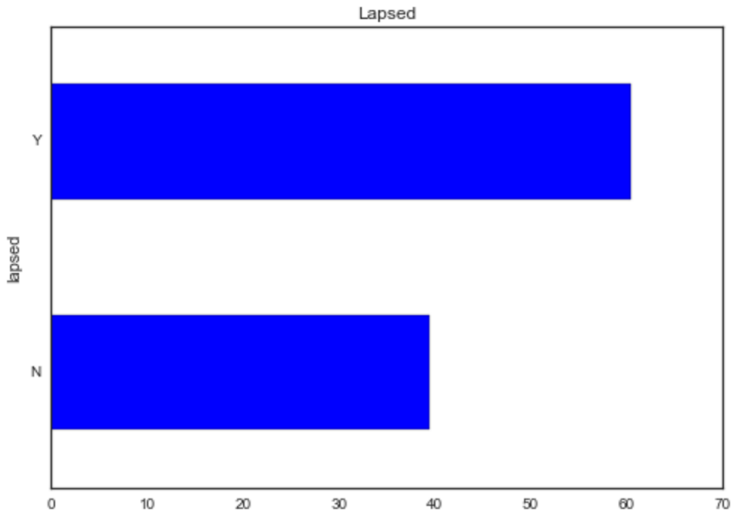

Figure 5: Distribution of Age

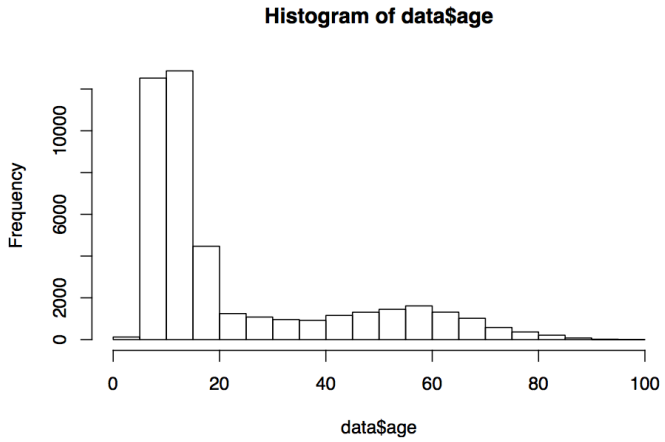

Figure 6: Distribution of Membership Months

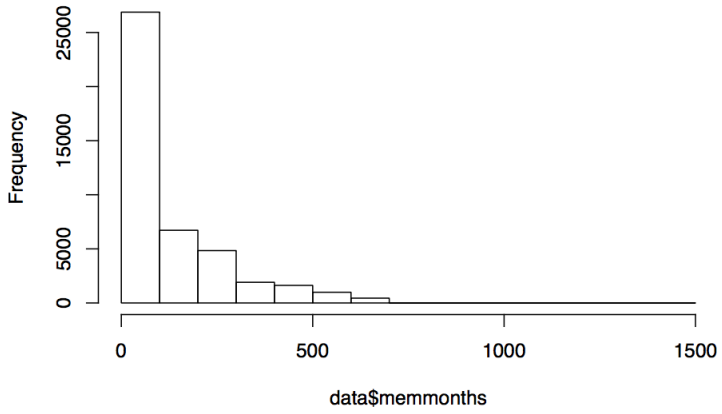

Figure 7: Distribution of Regions

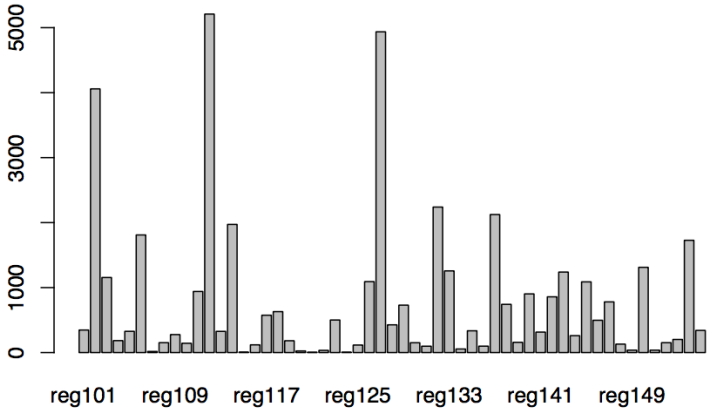

Figure 8: Percentage of Missing Data by Feature

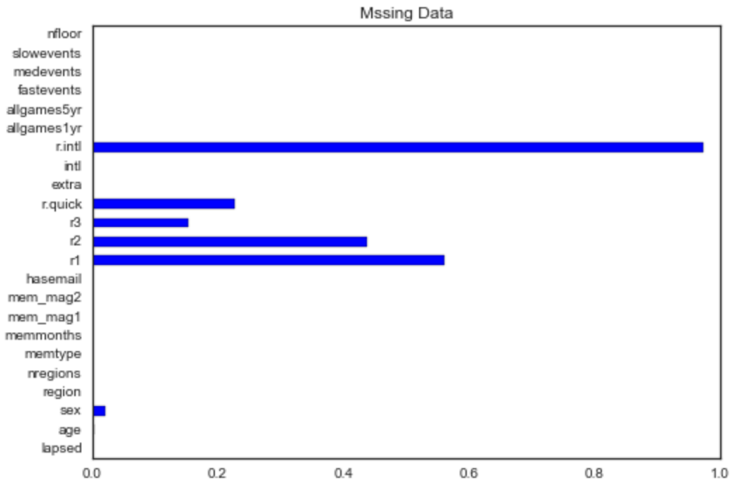

Figure 9: Cook's Distances

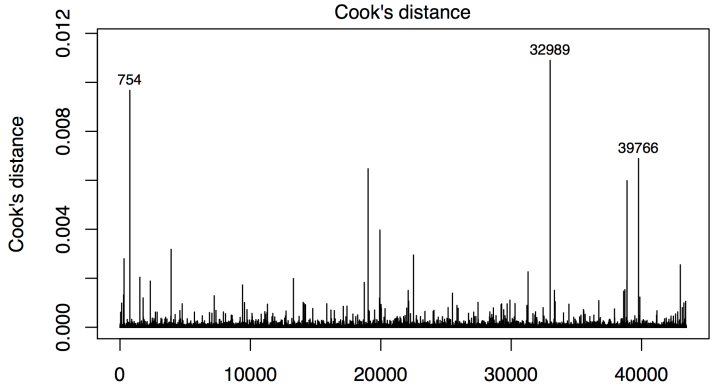

Figure 10: Fitted vs. Jackknifed Residual Plot

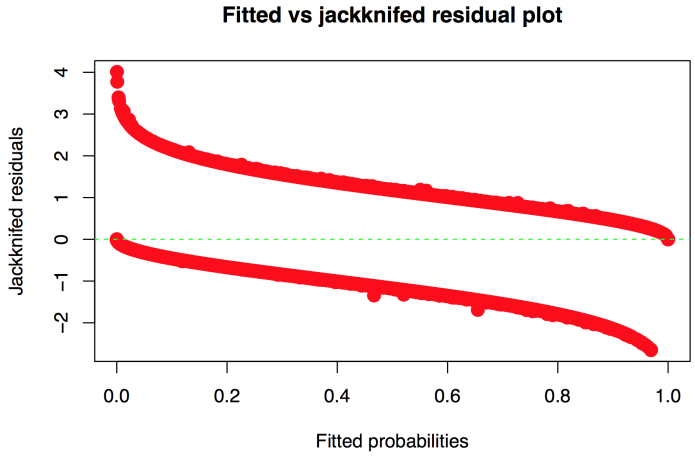

Figure 11: Average Fittend vs Jackknifed Residual Plot

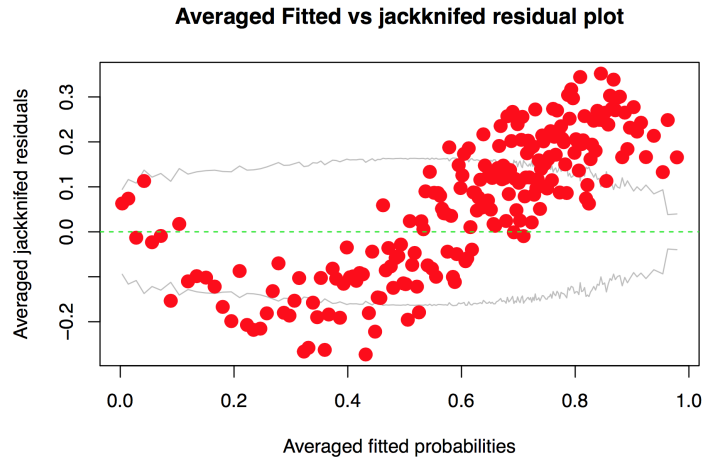

Figure 12: Classification Tree Cross Validation Error

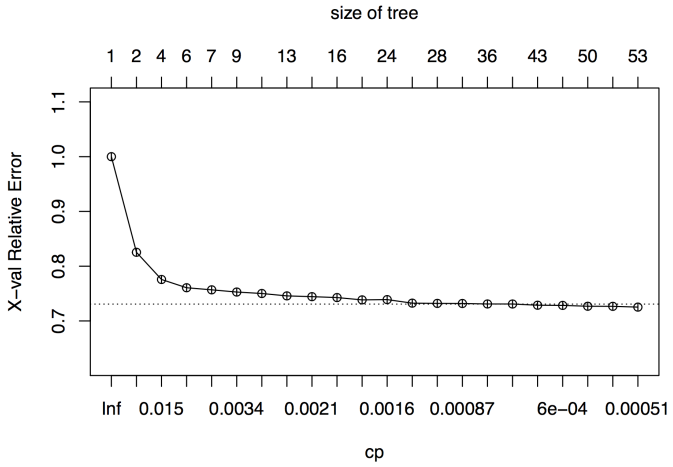

Figure 13: Pruned Classification Tree

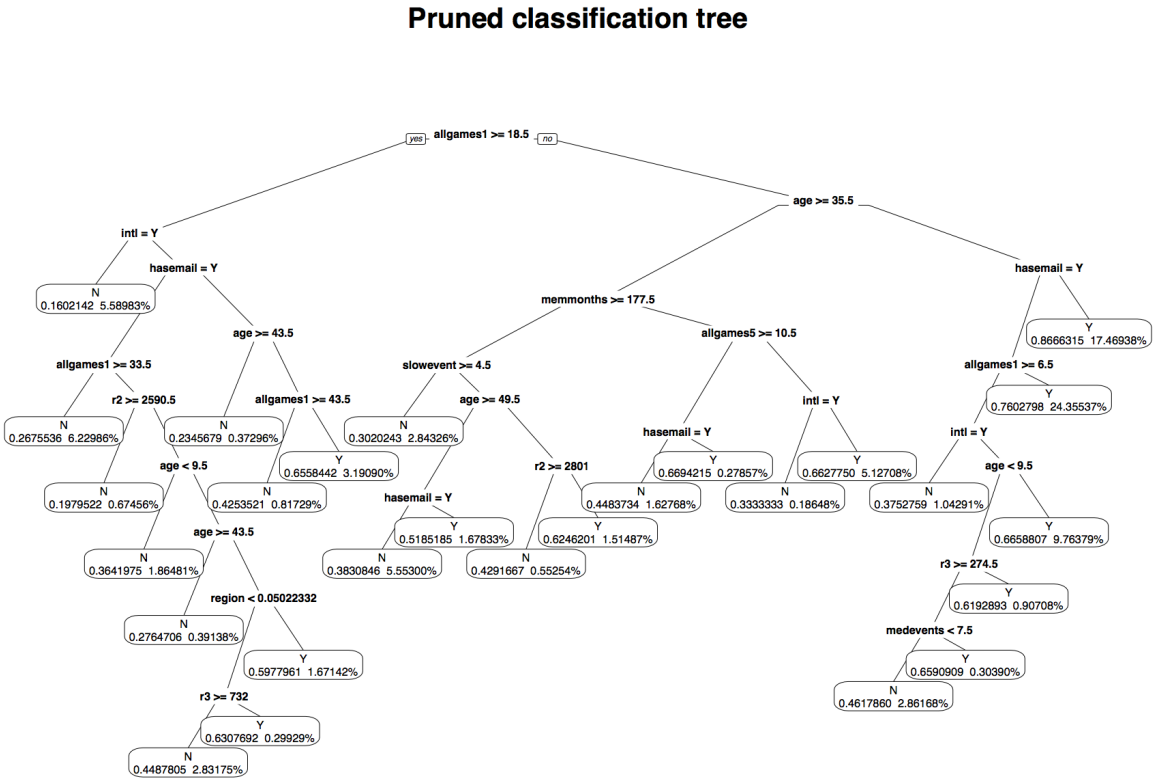

Figure 14: Variable Importance Plot

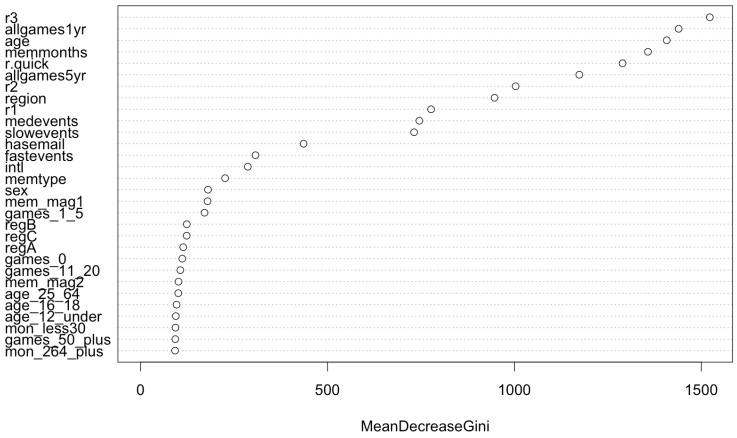

Figure 15: Random Forest Model Summary

```
##
## Call:
##  randomForest(formula = lapsed ~ ., data = train2, ntree = 500,      do.trace = 10)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 27.75%
## Confusion matrix:
##      N     Y class.error
## N 9530  7619   0.4442825
## Y 4435 21852   0.1687146
```

C Logistic Regression Summary

```
Call:
glm(formula = lapsed ~ age + sex + region + nregions + memtype +
memmonths + mem_mag1 + mem_mag2 + hasemail + r1 + r2 + r3 +
r.quick + extra + intl + r.intl + allgames1yr + allgames5yr +
fastevents + medevents + slowevents + nfloor + age.na + r1.na +
r2.na + r3.na + r.quick.na + r.intl.na + age:memtype + memtype:r1 +
sex:r1 + memtype:hasemail + age:sex, family = binomial, data = data.na)

Deviance Residuals:
Min       1Q   Median       3Q      Max
-2.6369  -1.0420   0.5900   0.8764   4.0052

Coefficients:
Estimate Std. Error z value Pr(>|z|)
(Intercept)         4.096e-01  9.204e-01   0.445 0.656301
age                 1.138e-01  1.851e-02   6.147 7.87e-10 ***
sexM               -6.164e-02  1.086e-01  -0.567 0.570442
sexZ_dummy_NA       2.020e+01  1.832e+02   0.110 0.912228
regionreg102        1.114e-01  1.299e-01   0.857 0.391361
regionreg103        5.376e-02  1.410e-01   0.381 0.702993
regionreg104        2.472e-01  2.074e-01   1.192 0.233338
regionreg105        1.075e-01  1.742e-01   0.617 0.536969
regionreg106       -8.249e-02  1.357e-01  -0.608 0.543155
regionreg107        2.976e-01  5.414e-01   0.550 0.582552
regionreg108        3.012e-01  2.240e-01   1.345 0.178776
regionreg109        4.473e-01  1.851e-01   2.416 0.015672 *
regionreg110        1.426e-01  2.334e-01   0.611 0.541160
regionreg111        1.270e-01  1.443e-01   0.880 0.378788
regionreg112        2.749e-01  1.291e-01   2.129 0.033247 *
regionreg113        5.188e-01  1.829e-01   2.836 0.004569 **
regionreg114        1.441e-01  1.355e-01   1.064 0.287551
regionreg115        5.993e-01  1.120e+00   0.535 0.592450
regionreg116        4.590e-01  2.525e-01   1.818 0.069132 .
regionreg117        5.737e-02  1.552e-01   0.370 0.711678
regionreg118        2.106e-02  1.529e-01   0.138 0.890454
regionreg119       -6.542e-02  2.051e-01  -0.319 0.749723
regionreg120       -5.026e-02  4.428e-01  -0.113 0.909638
regionreg121        7.570e-01  1.255e+00   0.603 0.546367
regionreg122        8.420e-01  4.454e-01   1.891 0.058658 .
regionreg123        1.099e-02  1.588e-01   0.069 0.944837
regionreg124        5.785e-02  9.314e-01   0.062 0.950473
regionreg125        2.096e-01  2.401e-01   0.873 0.382696
regionreg126        2.785e-01  1.429e-01   1.949 0.051250 .
```

```
regionreg127        1.527e-01  1.290e-01   1.184 0.236295
regionreg128       -5.671e-02  1.637e-01  -0.346 0.729020
regionreg129        4.051e-01  1.513e-01   2.677 0.007431 **
regionreg130        1.365e-01  2.254e-01   0.605 0.544940
regionreg131       -3.051e-02  2.549e-01  -0.120 0.904728
regionreg132        3.842e-01  1.341e-01   2.865 0.004171 **
regionreg133        1.610e-01  1.395e-01   1.154 0.248558
regionreg134        1.124e-01  3.138e-01   0.358 0.720178
regionreg135       -4.280e-02  1.749e-01  -0.245 0.806647
regionreg136        1.974e-02  2.523e-01   0.078 0.937622
regionreg137        1.043e-01  1.336e-01   0.781 0.434862
regionreg138        7.177e-02  1.495e-01   0.480 0.631048
regionreg139        1.573e-01  2.145e-01   0.733 0.463494
regionreg140        4.924e-01  1.472e-01   3.344 0.000827 ***
regionreg141        2.232e-01  1.774e-01   1.259 0.208178
regionreg142        2.049e-01  1.463e-01   1.401 0.161357
regionreg143        2.610e-01  1.404e-01   1.859 0.063062 .
regionreg144       -4.689e-01  1.885e-01  -2.488 0.012863 *
regionreg145        1.143e-01  1.437e-01   0.796 0.426289
regionreg146        1.560e-01  1.606e-01   0.971 0.331368
regionreg147       -1.157e-01  1.478e-01  -0.783 0.433812
regionreg148       -4.369e-01  2.270e-01  -1.924 0.054302 .
regionreg149        3.264e-01  3.748e-01   0.871 0.383819
regionreg150        1.879e-01  1.388e-01   1.353 0.175935
regionreg151       -4.606e-01  3.761e-01  -1.225 0.220712
regionreg152        7.315e-02  2.191e-01   0.334 0.738441
regionreg153        3.618e-02  1.990e-01   0.182 0.855741
regionreg154       -1.590e-02  1.362e-01  -0.117 0.907055
regionreg155       -1.062e-01  1.718e-01  -0.618 0.536643
nregions            9.867e-02  4.969e-02   1.986 0.047051 *
memtypeF            1.622e+00  3.439e-01   4.717 2.40e-06 ***
memtypeN            2.527e+00  3.184e-01   7.936 2.09e-15 ***
memmonths          -7.961e-04  1.115e-04  -7.139 9.43e-13 ***
mem_mag1Y          -1.495e-01  3.335e-02  -4.484 7.34e-06 ***
mem_mag2Y          -3.521e-01  3.880e-02  -9.075  < 2e-16 ***
hasemailY          -8.797e-01  1.152e-01  -7.635 2.27e-14 ***
r1                  1.086e-04  1.301e-04   0.835 0.403857
r2                  7.116e-05  4.867e-05   1.462 0.143701
r3                 -2.432e-04  4.508e-05  -5.395 6.86e-08 ***
r.quick             1.069e-04  2.825e-05   3.783 0.000155 ***
extraY             -4.594e-01  5.882e-02  -7.810 5.71e-15 ***
intlY              -6.654e-01  4.976e-02 -13.372  < 2e-16 ***
r.intl             -3.055e-05  2.393e-04  -0.128 0.898410
allgames1yr        -3.834e-02  1.078e-03 -35.575  < 2e-16 ***
allgames5yr        -3.364e-03  7.044e-04  -4.776 1.79e-06 ***
```

```
fastevents          -8.834e-03  2.391e-03  -3.695 0.000220 ***
medevents            1.796e-02  2.711e-03   6.624 3.49e-11 ***
slowevents           1.077e-02  2.577e-03   4.178 2.94e-05 ***
nfloor              -2.324e-02  9.613e-03  -2.418 0.015614 *
age.na               1.311e+00  3.291e-01   3.984 6.78e-05 ***
r1.na               -1.541e-01  6.220e-02  -2.478 0.013225 *
r2.na               -1.843e-01  6.052e-02  -3.045 0.002326 **
r3.na                3.367e-01  5.545e-02   6.072 1.26e-09 ***
r.quick.na           1.566e-01  4.561e-02   3.433 0.000597 ***
r.intl.na           -3.684e-01  8.983e-02  -4.101 4.11e-05 ***
age:memtypeF        -1.245e-01  1.857e-02  -6.703 2.04e-11 ***
age:memtypeN        -1.364e-01  1.839e-02  -7.414 1.23e-13 ***
memtypeF:r1         -2.537e-04  1.320e-04  -1.922 0.054630 .
memtypeN:r1         -1.786e-04  1.159e-04  -1.541 0.123365
sexM:r1             -3.780e-05  5.842e-05  -0.647 0.517533
sexZ_dummy_NA:r1    -1.003e-02  9.434e-02  -0.106 0.915366
memtypeF:hasemailY   3.429e-01  1.468e-01   2.336 0.019501 *
memtypeN:hasemailY   1.394e-02  1.188e-01   0.117 0.906620
age:sexM             4.111e-03  2.592e-03   1.586 0.112708
age:sexZ_dummy_NA    5.789e-02  4.143e-02   1.397 0.162307
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 58278  on 43435  degrees of freedom
Residual deviance: 48811  on 43342  degrees of freedom
AIC: 48999

Number of Fisher Scoring iterations: 12
```

D Description of Features in Original Data Set

1. *lapsed*: response variable: did the membership lapse [N/Y]
2. *age*: age in years
3. *sex*: gender [F/M]
4. *region*: geographic region
5. *nregions*: number of regions of residences in the previous 6 years
6. *memtype*: N=normal, A=affiliate, F=family
7. *memmonths*: number of months since joining US Chess
8. *mem_mag*1: subscribed to membership magazine # 1 [N/Y]
9. *mem_mag*2: subscribed to membership magazine # 2 [N/Y]
10. *hasemail*: member has an email address on file [N/Y]
11. *r*1: transformed regular chess rating from 2 years prior
12. *r*2: transformed regular chess rating from 1 year prior
13. *r*3: transformed current regular chess rating
14. *r.quick*: transformed current quick-chess rating
15. *extra*: does the member play a particular variant of chess [N/Y]
16. *intl*: does the member have an international rating [N/Y]
17. *r.intl*: member's transformed international rating
18. *allgames*1*yr*: number of tournament games played in the past year
19. *allgames*5*yr*: number of tournament games played in the past 5 years
20. $fastevents$: number of quick/blitz tournaments in the past 5 years
21. *medevents*: number of medium-speed tournaments in the past 5 years
22. *slowevents*: number of regular-speed tournaments in the past 5 years
23. $nfloor$: number of times the member's rating dropped to a minimum in the past 5 years