MSiA 421 Data Mining - HW#1

Team Name:	DataRobots, Username3 (on Kaggle)	
Members:	Rishabh Joshi, Daniel Lutolf-Carroll	
Files:	submission_file.Rmd	(includes data cleaning, EDA, first wave of models, CVs)
	xgb4.Rmd	(second wave of models using xgboost)

We first did some rudimentary EDA and data cleaning and noticed some abnormalities (e.g. price = 0, and category 99) as well as extreme values (high qty) in the test set.

Based on our EDA and conceptual understanding of the data set, we built features to capture recency, frequency, and monetary profiling of clients. A noteworthy feature was ***activity*** which is a ratio of the amount of lifetime a client was an active purchaser (difference of lifetime and last purchase over lifetime). See code comments for further documentation on all the features and their creation.

We seperated the training and test sets, then ran a basic linear regression on the training using our features. We predicted values for the test set, but realized that estimates were conceptually not reasonable for a small proportion of the ids, where ***qty*** was huge. We optimized our approach by building a logistic model to classify the ***ids*** where the response was not zero, and a linear model that would estimate the magnitude of ***logtarg*** values. The predicted ***yhat*** is then a product of the two models (***probabilities * estimates***).

We optimized each model through feature selection methods (fwd/bwd stepwise and lasso) and through several cross validation approaches.

A final modeling approach was to use ***xgboost***. We tried both a pure regression and an aggregation of a classifier plus regression using ***xgboost***. Our cross validations showed that ***xgboost*** outperformed the traditional models. Fine tuning the tree parameters delivered a model that did well both on cv-train set and the public test set (30%), but due to the training set being a poor representation of the private test set (70%), we opted out from submitting it.

Our final submission included a few tweaks. We turned all negative values to zero and set a maximum threshold on the estimate before multiplying it with the corresponding probability. This was done to compensate due to the nature of the test predictor values (high outliers). Of note is how the automatic fine tuning of xgboost picked up on this as well (gamma really low).

List of all the features

1. ***tot_price***
2. ***tot_qty***
3. ***slstyr***
4. ***ordtyr***
5. ***slslyr***
6. ***ordlyr***
7. ***sls2ago***
8. ***ord2ago***
9. ***slsbfr***
10. ***ordbfr***
11. ***tot_orders***
12. ***avg_items***
13. ***avg_price***
14. ***tot_cat***
15. ***first_pur***
16. ***last_pur***
17. ***pur_time_avg***
18. ***activity***
19. ***last_ord_wt***