CLV Model v3, June 22, 2020

By Pablo Rodriguez Bertorello

Customer Lifetime Value Model

Rewrite from version 2 to version 3

Progress

- New version of the CLV model is rewritten from scratch. Model is trained for first time since almost a year ago
- New choice of Deep Neural Network model loss function better fits current business needs. Model error is reduced by up to 75%+. Compute needs are reduced by 99%+ via simpler model
- Now running on Spark, compute limitations of the past are overcome. Training an epoch is estimated to be 10x+ faster than on Domino/ADW, from hours to minutes. Simple queries, i.e. count, perform up to 400x faster. All available data is now leveraged for the best model fit
- Engineering practices are enhanced by leveraging GitHub branch policies, pull request code reviews
- Work concentrated on BF US, generalizes for all brands, and models beyond CLV

Modeling Transformation

CLV Version 2

The version 2 model currently in production was tuned to predict the expectation of net revenue. It was tuned with the intention to better fit high-end customers.

$$CLV = E[NetRevenue \mid specific\ customer] =$$

$$PurchaseRegression(customer) * ProbabilityOfPurchase(customer) +$$

$$ReturnRegression(customer) * ProbabilityOfReturn(customer)$$

The production model suffers several drawbacks:

- By not building a single end-to-end model, no measure of total model error was estimated — the upper bound error is not known. BF US lower-bound mean_absolute_error is large at $229, for purchase regression
- A major oversight is that error estimated against the training set, instead of a held out dataset, likely leading to substantial underestimation of model error
- Thus, it is difficult to invest marketing dollars in new customers, because the ROI would be too uncertain. Plots of by quantile hide possible large % differences between predicted and actual CLV
- Binary classification probability of purchase/return models are utilized, with “fair” ROC AUC under 79% provided as evidence of sufficient fit. This approach is conceptually flawed, as a binary predictor’s “logit” is not a reliable estimation of probability density function. How much compounding regression error with classification cross-entropy compromises the model’s fitness goes unquantified
- A total of almost 50 sub-models make up a single brand’s model. For instance, for each regression model, 20+ quantile models were trained, relying on Tilted Loss
- Found training code does not perform any hyper parameter search. For example, it lacks any explicit declaration of optimizer parameters, instead utilizing unstated defaults
- The above is made more conspicuous by the lack of loss and error plots
- Human bias is inserted into the training set by arbitrarily filling null value with zero, and dropping 1% of customers as extreme

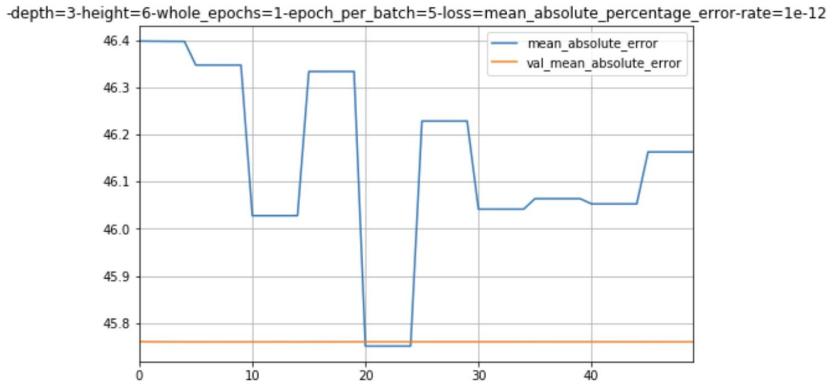
CLV Version 3

The new version 3 of the CLV model overcomes the issues above, better suiting current business need, by evenly fitting customers at all lifetime value levels:

$$CLV = \text{Regression}(\text{Purchases} - \text{Returns} \mid \text{customer})$$

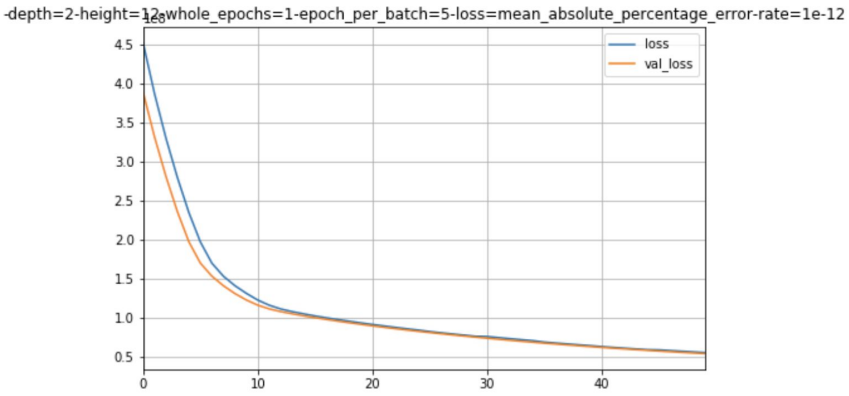
By Pablo Rodriguez Bertorello

- Total model error is known, under $47 when measured as mean_absolute_error. This is a reduction of 75%+, when compared to version 2



- To minimize error for customers with different spend/return levels, the loss function chosen is mean_absolute_percentage_error. Importantly, mean_absolute_error is dominated by high-spending customers

- A single single end-to-end regression model is trained for CLV, rather than up to 50 models in version 2.
- A grid of hyper parameters is searched, and a best model chosen, which minimizes test loss. Validation performance is measure on a held out data set



- No human bias is entered by filling values, nor by dropping customer segments
- Work concentrated on BF US, generalizes for all brands, and models beyond CLV

Compute Transformation

CLV Version 2

- One-off code manages low-performance batching of data for training/inference. Reliance on ADW results in simple queries to take hours to evaluate, when they actually take minutes to run
- Data is discarded to fit within memory constraints, potentially impacting model fit
- A different flavor of the code for each brand multiplies the burden on the team: for data preparation, training, and inference

CLV Version 3

- All data processing is performed per the Spark framework. Training an epoch is estimated to be 10x+ faster than on Domino/ADW, from hours to minutes. Simple queries, i.e. count, perform up to 400x faster
- Compute needs are reduced by 99%+ via simpler model
- The entire dataset is leveraged to achieve the best model fit. No data is discarded

- A single code base can be applied to train for all brands
- Work concentrated on BF US, generalizes for all brands, and models beyond CTV

Next Steps

- Consider simplifying all brand models into a single one
- Migrate DBO training set generation script to Spark
- Enhance DNA with necessary features
- Add a central dashboard for all model trainings
- Extract test code as separate scripts, run on cron
- Monitoring of production performance
- Data training parallelism with Horovod