Deep Learning for Customer Lifetime Value Estimation

(Swapnil Manchanda, Sean Soutar, Robert Smillie)

Abstract

Customer lifetime value modelling is used to predict the future value of a customer from a company's perspective. Given there are many industries, each with a different customer base, applying a generalised modelling framework across all industries and their respective customer groups is challenging. Each existing modelling technique has its own pros and cons. In this paper we supply an amalgamation of deep learning and probabilistic methods focused on an e-commerce data set to model customer lifetime value. We developed a custom zero-inflated inverse gamma loss function specifically tailored to our data. Furthermore, we also develop a deep learning architecture inspired by random forests. We succeed in showing that the current, out of the box methods can be vastly improved upon with a tailored well thought out approach. By customising our deep learning components to the underlying customer lifetime value density distribution of the data set, we ensured our models outperformed Google's deep learning model for customer lifetime value prediction.

1. Introduction

How can a company build long term relationships with their customers, promote effective marketing and investment strategies? Understanding the economic value of a customer's relationship with a firm can help answer this question. One way of measuring the economic value of a customer's relationship with a firm, is by measuring the customer lifetime value (CLV) (Andon et al., 2001). CLV is a statistic which represents the expected total amount a customer will spend in the future, by purchasing goods or services from a given company, until they cease to become a customer. From a company's perspective, one can never truly be certain of how much money a customer will spend in the future where there is no contractual obligations between customers and the company. The best action a company can do is try to predict future income. Predicting CLV accurately is challenging as you have to deal with cash flow being stochastic (Calciu, 2009) in nature due to a large number of factors such as human behaviour, political, environmental, economic and much more. The high variance and possible temporal evolution found within customer relationship data makes predicting future CLV challenging.

Naive approaches for CLV prediction are formula based (Gupta et al., 2004; 2006), with a limited set of variables and parameters. Challenge with the naive approach is that it can severely over or underestimate. We hypothesis creating non-linear functions with a substantially large number of parameters and variables can help predict more accurate CLV values. Since deep learning models typically have a large number of parameters, one can attempt to leverage deep learning to create a non-linear regression model in order to predict how much business a customer may provide in the future.

We find Wang et al. paper on deep probabilistic model for CLV prediction to be a reasonable starting point when it comes to understanding the properties a deep probabilistic neural network should consider when trying to predict CLV values for customers. For simplicity we shall term their proposed architecture as 'Google F.F.'. Their results showcased competitive performance on the KDD Cup 1998 Data Set, where around 95% of customers are one-time purchasers. The heavily skewed distribution poses challenges for mean-square error (MSE) loss function. The MSE loss function can over-penalise prediction errors for the 5% of customers that purchased. Therefore, they proposed a novel zero-inflated lognormal loss function which counteracts the issue faced by the MSE loss function. However, they have not showcased how their model performs in a different industry/setting, where the behaviour of customers can be vastly different.

The Chamberlain et al. paper states that when it comes CLV prediction, state of the art resides within systems using Random Forest regressors as they have been shown to perform well in highly stochastic problems. Researchers have aimed to create deep learning models which outperform tree-based systems such as ones using the Random Forest. This paper's aim is to create deep learning model(s) which outperforms the Random Forest.

In this paper we shall use the Online Retail Data Set (e-commerce data set) from the UCI Machine Learning Repository [1], for training and evaluating our models. The first stage of the experiments evaluates Google F.F. and Random Forest on the e-commerce data set. Our observation and background research motivated us to develop a custom loss function and a custom deep neural network architecture with the aim of outperforming our baseline. Finally, we apply an over-sampling method called SMOTE (Chawla et al., 2002), to see if we could improve our results further. Our experimental results are based on how well the models generalises through minimising test set error.

Our contributions are:

- A novel custom loss function which we term as Zero-inflated Inverse Gamma (ZIIG), for modelling customer lifetime value distributions with extreme skewness to the left and low kurtosis.
- A deep neural network architecture which provides competitive performance in estimating customer lifetime value. We name this network Dropnet.

[1] https://archive.ics.uci.edu/ml/datasets/Online%20Retail

2. Background and Related Work

In the introduction we define CLV as the 'total amount a customer will spend across their life', however when it comes to modelling, this definition creates many challenges. For example, 'how long will each customer live for?' and 'what if a person ceases to be a customer in the future?' are non-trivial concerns. It is very likely that not each and every person will remain a customer for the same amount of duration, therefore when modelling, taking a finite duration of time T into the future given the present defined by time $t = 0$, will make our models more robust through simplifying the modelling objective.

Theoretically the expected CLV for each customer k, given the company c, data related to a specific customer $\boldsymbol{D_k}$, model parameters $\boldsymbol{\theta_k}$, from time $t = 0$ to T can be represented by the following:

$$CLV(k \mid c, t = 0 : T, \boldsymbol{D_k}, \boldsymbol{\theta_k}) = \mathbb{E}_{x_k \sim p(x_k \mid c, t=0:T, \boldsymbol{D_k}, \boldsymbol{\theta_k})}[x_k] \quad (1)$$

Initial CLV prediction models were formulae based approach known as recency, frequency and monetary (RFM) formula's. These models were specifically developed for target marketing programs (Gupta et al., 2004). The drawback of these models are their inability in capturing extremely varied customer level behaviour.

The next evolution in CLV modelling came in the form of a probabilistic approach by applying Bayesian models (Fader et al., 2010). Bayesian models observed many different features of customer behaviour and represented it as a process which is governed by latent behavioural characteristics. A popular probabilistic CLV approach is the use of Pareto/Negative Binomial distribution conjugates, where predicts if the customer will purchase a product in the future and if so, how much will they spend.

The challenge of predicting CLV using probability theory is that you have to specify a distribution for each customer. This has experimentally been shown to not be flexible when it comes to capturing the wide range of customer behaviour in CLV prediction (Chen et al., 2018).

We consider deep learning to model tabular data for CLV prediction. Deep learning models are known for their flexibility in modelling complex processes. Additional advantages include the simple incorporation of different data types into the model, the avoidance of excessive feature engineering through utilizing representation learning and deep learning can be training in an online way if desired. The work of Chen et al. compared the performance of stochastic distributions such as Pareto & Negative Binomial with using convolutional neural networks (CNN) to predict future CLV of individual customers. Their research found that CNN outperformed stochastic models. The challenge of using stochastic models was they were not able to predict who the top spenders were, whereas it was found that CNN were able to better predict who the top spenders were.

Deep neural networks based on sequential convolutional layers or multilayer perceptrons run the risk of being overly parametrized (Arik & Pfister, 2019). This leads to a lack of inductive bias, which often causes such models difficulties in finding optimal solutions on tabular data. Canonical research into self-supervised learning on tabular data modelling was performed by Arik & Pfister. Tabnet utilises sequential attention mechanism during learning, in order to choose which features are most meaningful at each decision step. Tabnet was shown in their research to outperform decision tree methods and other deep learning approaches for modelling tabular data.

Defining the customer to company relationship provides insight into the behaviour of purchasing pattern. This relationship can either be contractual or non-contractual. If a contract has been signed between the company and a customer, then there is a legal obligation. One can predict with higher confidence when the company will see a flow in cash, for example a bank providing mortgage on a house to a customer. On the other hand, in a non-contractual setting the customer has no legal obligation and can make transaction to his or her needs, for example a brick and mortar store purchase. Furthermore, transactions can be either discrete or continuous. Discrete transactions are ones where cash flow can only occur at a set of discrete times. On the other hand, if a transaction is continuous, the customer is free to make a transaction at any point in time. In this paper we will only look to define CLV models for non-contractual customers who can make continuous transactions.

Since in this research we will be modelling for a non-contractual setting, often it can be observed that a small share of users often encompasses the largest part of the revenue. What this typically leads to in the data set is a class imbalance with respect to the financial value of customer segments. Given the entire data set, only a small subset of data would be classed as customers who drive company revenue, whereas the remaining majority data would be classed as customers who have contributed very little or nothing. Malthouse & Blattberg proposed that the top 20% of customers tend to drive a majority of total revenue. Sifa et al. showed that applying a synthetic minority oversampling technique (SMOTE) on their experimental training data allowed for better generalisation when training machine learning models especially on high-value customers.

Chamberlain et al. made use of product embeddings to boost CLV predictive accuracy. These embeddings have been shown to give better results than using the sparse representations produced by one-hot encodings and similar methods.

Finally, whilst Recency, Frequency and Monetary (RFM) models are deficient in themselves, using this framework in constructing informative customer features has been shown to improve deep learning performance (Sifa et al., 2018).

3. Data Set

The data set we used in this research is a transactional data which outline transactions that occurred between 01/12/2010 and 09/12/2011 for a UK-based, online retail company. This company sells gifts and items for events as well as other celebrations. Both the general public and wholesalers brought from this company.

Some key features include:

- Unique Customers: 4290
- Number of Transactions: 21,856
- Number of Products: 4520

The data is arranged in market basket format initially (Figure 2). Each transaction is accompanied by an InvoiceNo which is uniquely tied to a single customer via the CustomerID. Each unique item in a transactional basket is spread across multiple rows as seen in the example invoice in Figure 2.

3.1. Data Modification

The original data has a mapping of many transactions to one customer as seen in Figure 2. Therefore, data pre-processing was applied in order to map all transactions related to a single customer and have that information condensed into a single row.

We removed any rows that indicated a cancellation or amendment. Then by grouping by each unique customer, we calculated aggregated features. For each customer we took the historic CLV to be the sum of all transaction values in the data up until a given cut-off date in time. Our cut off dates were constructed such that our training set grew from 8 months to 10 months' worth of data successively. Transaction values beyond a cut-off date is reserved for calculating the target CLV value. The square root of these values were returned.

$$sqrtCLV = \sqrt{\sum_{\forall invoiceNo} Quantity_{invoiceNo} \times UnitPrice_{invoiceNo}} \tag{2}$$

The square rooted values were used because upon analysing the distribution of values in Figure 1, the values exhibit an intense positive skewness as shown by the left plot. The square-root distribution results in an easier distribution to model without extreme tail behaviour. The log transformation was not used because at different cut-off dates, the target CLV values for certain customers could be zero as they did not buy again beyond the cut-off date. We created three different data sets with cut-off dates of 2010-09-01, 2010-10-01 and 2010-11-01.

The total number of transactions and days since last transaction for each customer are calculated. In yellow (Figure 2), the two distinct invoice numbers give a number of transactions of two. In orange and red the historic and target CLVs are calculated respectively and square rooted according to an example cut-off date of 2010-10-01. In green the days from last purchase before the cut-off date is calculated. These features calculated lie within each category of the famous recency frequency and monetary (RFM) framework.

3.2. Historic Basket Description Embeddings

Additionally, we wanted to include semantic information on the product descriptions to boost predictive performance. In Figure 2 the product descriptions in blue are used to create 10-dimensional numeric embeddings. Only information before the cut-off date is used.

There are many options for embedding these descriptions. One option is to use a pre-trained model embeddings such as GloVe. The descriptions found in the data were quite esoteric therefore we opted to train our own embeddings using the FastText algorithm (Joulin et al., 2016; Bojanowski et al., 2016).

The FastText algorithm is different to other word to vector algorithms in that it uses sub-sequences within the word strings to provide extra information. This is useful in a suffix and prefix heavy language such as English. We trained this on the descriptions in the training data and fit a numeric vector of length ten to each description.

Now that each product description is embedded into a 10-dimensional numeric space, how should this information be summarized per customer? We elected to *add* all of the numeric vectors for each item that a customer has bought across his relationship with the company. The average of this resultant vector is calculated elementwise. What is left is a 10-dimensional average vector which shows the average product description vector associated with each customer. The idea is to reflect further information about the purchasing habits of this customer in this new space. For example, it is theorized that a customer who buys items for large corporate events will have a very different average vector than someone who bought party items for her young children this year.

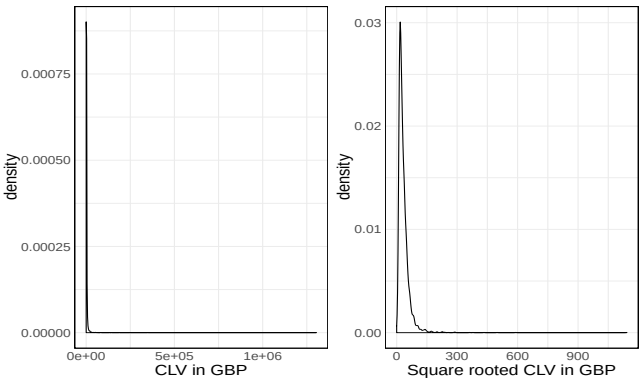

Figure 1. Density plot of unmodified CLV distribution and CLV distribution after square root transform.

3.3. Zero Historic CLV Customers

There are some customers in the data set which have zero historic CLV and then go on to purchase new items beyond a given cut-off date and so have a positive future CLV. The first thought when dealing with these customers is to remove them from the data as they are not representative of a general customer with no previous purchase. There is a bias in that the data set does not include customers who have not bought previously and will not buy in the future, this means that for any customer with no past transactions there will be a future transaction, obviously this will not hold true in the real world. However, we saw some value in retaining these customers even though it is likely to reduce our models' accuracy. For example, if a company knows that on average it attracts ten extra customers per month then our models can estimate these customers' CLVs. CLV estimation for these customers form a best guess estimate without any information. A way of interpreting it is that the company should not spend more than this best guess in additional marketing spend on these customers. Also, growth in this prediction estimate could indicate that on average, the business is attracting more valuable customers. It seems clear that this would be of some value to the company. To encode these customers into the data we calculated everything else in the same way as described in §3.1. We assigned them a 'days since last purchase' of -1 and we set their product description vector to zero (which is the mean value).

3.4. Data Splitting

The challenge we foresaw was an inadequate volume of training data therefore, we elected to use multiple data splitting to create five sets of training, validation and test sets per cut-off date. The training data comprised of 90% of the data in each split, the validation data consisted of 5% and the test data consisted of 5%. This allows us to account for variation within each time period via the 5 splits per cut-off date and across time through altering the cut-off dates to allow for a robust

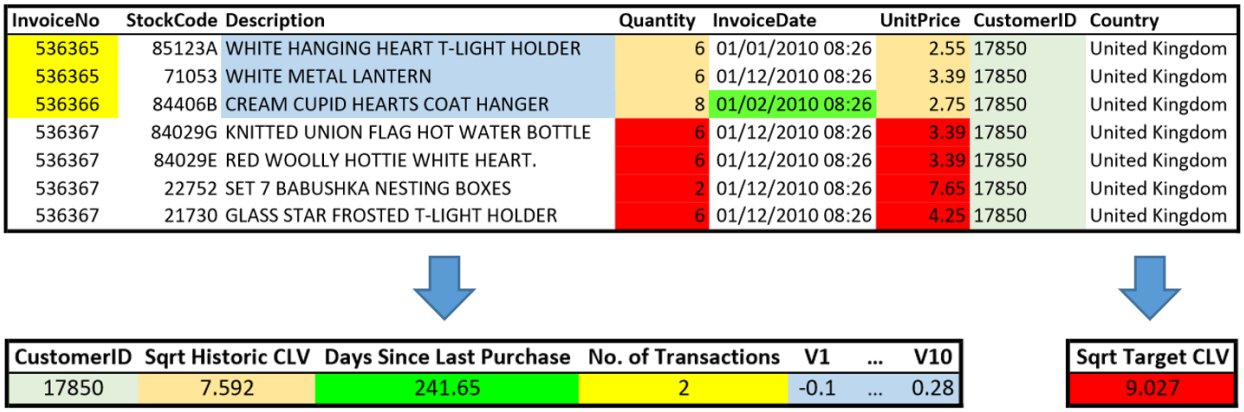

Figure 2. Visualization aide of data process. Precise process described in §3.1

measure of model performance.

4. Methodology

There are three key components to our experimental methodology. A set of loss functions, neural network architectures and a data augmentation strategy. We aim to outperform our baseline through a series of experiments that systematically place together these three key components.

4.1. Loss Functions

In this chapter we shall outline the different loss functions used to train the various models. The baseline loss functions we used were mean square error (MSE) and zero-inflated lognormal (ZILN) as outlined by our baseline paper (Wang et al., 2019).

Mean squared error (MSE):

$$L_{MSE} = \frac{1}{T}\sum_{k=1}^{T}\left(C\hat{LV}_k - CLV_k\right)^2 \tag{3}$$

Zero-inflated lognormal (ZILN):

$$L_{ZILN} = L_{BCE}(\mathbb{1}_{\{x>0\}}; p_k) + \mathbb{1}_{\{x>0\}}L_{LNRV}(x;\mu_k,\sigma_k) \tag{4}$$

$$L_{LNRV}(x;\mu_k,\sigma_k) = log(x\sigma_k\sqrt{2\pi}) + \frac{(logx-\mu_k)^2}{2\sigma_k^2} \tag{5}$$

$$L_{BCE}(\mathbb{1}_{\{x>0\}}; p_k) = -\mathbb{1}_{\{x=0\}}log(1-p_k) - \mathbb{1}_{\{x>0\}}logp_k \tag{6}$$

Furthermore, we propose to the best of our understanding a novel loss function which we shall term as zero-inflated inverse gamma (ZIIG). The ZIIG loss function is composed of the binary cross entropy loss and negative log likelihood of an inverse gamma distribution. The reason for proposing this loss function is that the inverse gamma probability distribution captures the tail behaviour of the online retail data set's CLV distribution better then the negative lognormal distribution as seen in the ZILN function (discussed in §5.1).

Zero-inflated inverse gamma (ZIIG):

$$L_{ZIIG} = L_{BCE}(\mathbb{1}_{\{x>0\}}; p_k) + \mathbb{1}_{\{x>0\}}L_{IGRV}(x;\alpha_k,\beta_k) \tag{7}$$

$$L_{Igrv}(x;\alpha_k,\beta_k) = \frac{\beta_k}{x} + (\alpha_k+1)logx - log\left(\frac{\beta_k^{\alpha_k}}{\Gamma(\alpha_k)}\right) \tag{8}$$

4.2. Baseline Model: Google F.F.

Our baseline was a deep probabilistic neural network as outlined by Xiaojing Wang et al. We did not modify their original proposed structure, for when we ran experiments using both mean square error (MSE) loss function L_{MSE} and zero-inflated lognormal (ZILN) loss function L_{ZILN}. When the deep probabilistic neural network is configured to minimise L_{MSE} for each customer k during training, the predicted output $C\hat{LV}_k$ from the network in the final layer is from one neuron. CLV_k is the actual CLV value for the customer.

In the case of using a ZILN loss function, the output of the final layer is 3 neurons. We arbitrarily assign each one of the three neurons to a sigmoid, identity and softplus activation functions, this allows the network to output a probability of a customer having a non-zero CLV value, a lognormal random variable L_{Lnrv} with mean μ_k and a standard deviation σ_k parameter , we pass the output neurons through sigmoid, identity and softplus activation functions respectively. Therefore, for each customer we obtain a vector consisting of $[p_k, \mu_k, \sigma_k]^T$. Using this vector, we can determine the CLV for each customer $C\hat{LV}_k$.

$$C\hat{LV}_k = p_k exp\left(\mu_k + \frac{\sigma_k^2}{2}\right) \tag{9}$$

Where the following conditions must be met, $p_k \in [0,1]$, $\mu_k > 0$ and $\sigma_k > 0$.

4.3. Google F.F. With ZIIG Loss Function

After initial experimentation using the MSE and ZILN loss function as defined by the Google F.F. architecture. We propose a novel and zero-inflated inverse gamma (ZIIG) loss function L_{ZIIG}. The output of the final layer is still three neurons. We arbitrarily assign each one of the three neurons to activation functions of a sigmoid, softplus offset by one unit and another softplus offset by one unit respectively. This allows the network to output a probability p_k of the customer having a non-zero CLV, alpha α_k and beta β_k for an inverse gamma random variable. Therefore, for each customer we obtain a vector consisting of $[p_k, \alpha_k, \beta_k]^T$.

$$C\hat{LV}_k = p_k\frac{\beta_k}{\alpha_k - 1} \tag{10}$$

Where the following conditions must be met, $p_k \in [0,1]$, $\alpha_k > 1$ and $\beta_k > 0$.

4.4. Tabnet Model

From the architecture perspective you can find the intricate details of Tabnet in the original paper (Arik & Pfister, 2019). Holistically, in Tabnet each building block implements a decision tree like output manifold. Tabnet uses a soft feature selection to identify which features are important given the data set. By applying a sequential multistep decision mech-

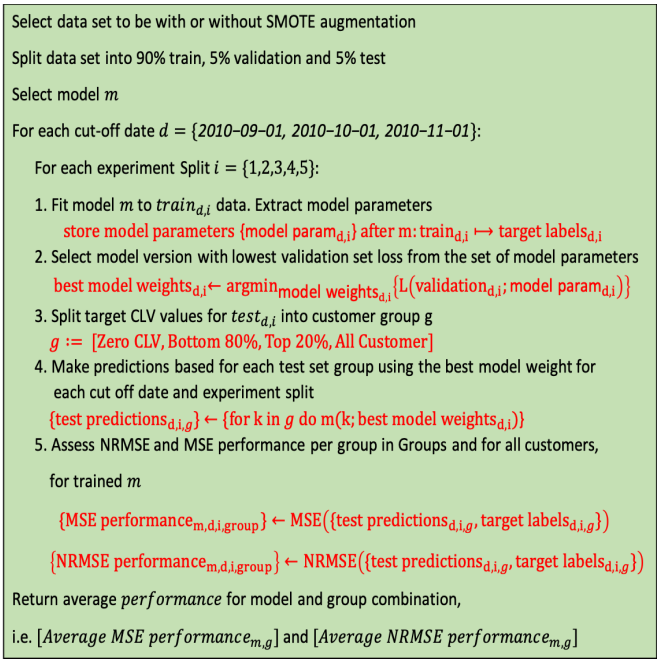

Figure 3. Performance evaluation process for a model

anism, the information is processed in a top down approach. The architecture contains set of transformer blocks which act as a way to apply sequential attention.

4.5. Dropnet Model

Dropnet was created by taking on inspiration from both the Random Forest and Tabnet architectures. The exact specification can be found in Figure 5. Both Tabnet and Random Forest consider subsets of input features at a given point. After the initial input layer, Dropnet has a dropout (Srivastava et al., 2014) layer which masks a proportion of inputs being fed into the network for each mini-batch pass. This is similar to a Random Forest which randomly selects a subset of features for each tree grown.

The reasoning behind this initial dropout layer is to focus the model to locally interrogate subsets of inputs features at a time as well as to prevent over-fitting. Subsequent filters at varying depths will learn interactions between the un-masked features and their subsequent representations. Filters consist of 1-dimensional convolutional layers. These were chosen because of their ability to locally scan and look for interactions between subsets of elements as they scan through a feature map. The sharing of weights in convolutional layers is theorized to allow the network to look for consistent patterns in the data and to avoid the issue of over-parameterization.

In our experimental methodology we perform experiments to observe the performance of Dropnet with MSE and ZIIG loss function individually. When predicting using MSE loss function the predicted output $C\hat{L}V_k$ from the network in the final layer is from one neuron. When predicting using ZIIG loss function the predicted output $C\hat{L}V_k$ follows equation 10.

4.6. SMOTE: Data Augmentation Strategy

SMOTE is a statistical technique used to synthetically generate data by sampling from the feature space of the minority class and its nearest neighbour, in order to increase the data volume of the minority class.

We apply smote to each cut-off split. For example, upon analysing all 4290 unique customers as of 2010-10-01 split, we find that 2457 are zero valued CLV, 1466 customers represent bottom 80% of CLV and top 20% of CLV customers equate to 367 people. The top 20% of CLV customers only equates to 8.6% of all unique customers, and yet these customers bring the majority of revenue to the company. Therefore by applying SMOTE to the top 20% of CLV customers for each cut of date, we increase the number of unique users in this minority class by 500% to address the class imbalance.

4.7. Evaluation Procedure

As the target square root CLV distribution is skewed to the left and has a low kurtosis, choosing a sensible reporting evaluation metric was vital. The metric we have chosen is normalised root mean squared error (NRMSE) in conjunction with Mean Squared Error (MSE). The formula for NRMSE is below:

$$\text{RMSE} = \sqrt{\frac{\sum_{k=1}^{N} (\hat{y}_k - y_k)^2}{N}} \tag{11}$$

$$\text{NRMSE} = \frac{\text{RMSE}}{\text{mean}(\hat{y_k})} \tag{12}$$

Where y_k represents the actual value and $\hat{y}_k$ the predicted value, for each customer. NRMSE is used because it is more sensitive to outlying observations than MSE. Furthermore, normalizing by the predictive mean allows for comparisons across data sets which opens the opportunity for future comparative research. NRMSE was chosen over alternatives such as mean absolute percentage error (MAPE) because of its ability to handle zero valued predictions and true zero values where MAPE would be undefined.

As well as evaluating on the full test data set we have split the test data set into customer categories containing the bottom 80% and the top 20% valued customers per CLV. We do this because the top 20% of customers in terms of CLV provide 77% of the total CLV, so predicting the values for these customers is seen as vastly more important than predicting the bottom 80%. This 80-20 split has been used previously in other literature (Malthouse & Blattberg, 2005) and defining miss-classifications into these categories is seen as an important issue. We can also equate this to finding the "whales", i.e. small subset of customers that provide most of the value, for online gaming companies(Chen et al., 2018). This necessitates the act of analysing model performance per customer grouping as well. Our algorithm for assessing performance is outlined in Figure 3.

Given the algorithm in Figure 3, we aim to outperform our baseline by creating an architecture through taking a combination of a neural network, loss function and data strategy, which minimises the NRMSE across all customer groups.

5. Experimental Results

The experimental results and discussions are broken down by each model architecture. This section is concluded by a time to convergence analysis of each architecture.

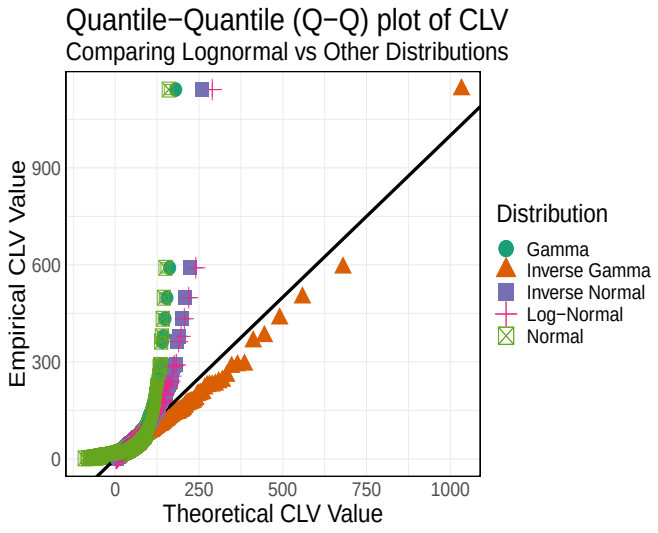

Figure 4. Unsuitability of Lognormal for CLV Response Distribution

5.1. Initial Experiments & Baseline Setup

It was found across all deep learning models that the batch size affected the predictive performance during mini-batch training. There is significant customer-level CLV variability present in the data as shown by the long right-hand tails in Figure 1. This variability meant that the batch size played an important role in model training. Very large batch sizes resulted in very slow training time and smaller batch sizes caused all deep learning models to generalize poorly to unseen data due to each mini-batch pass not seeing a sufficient range in customer behaviour. Initial experiments showed even though training times were longer, models benefit from larger batch sizes as the networks are more likely to see a wider range of customer behaviour within a mini-batch pass, while optimizing parameters. A batch size of 128 was determined to be an acceptable choice to ensure training stability. The Adam optimization algorithm was used with an initial learning rate of 10^{-3}.

The Google F.F. architecture proposed by Wang et al. included a choice of a mean squared error (MSE) loss function and zero-inflated lognormal loss (ZILN) function. Our experiment with a ZILN loss produced vastly inferior results compared to the use of MSE loss function. The authors argued for their data set the ZILN loss function fit their customer lifetime value (CLV) distribution well and hence produced superior results as opposed to the MSE loss function. This was not the case with the e-retailer data set. This architecture with the ZILN loss function produced test set MSE values which were **10x** that of the same architecture using a MSE loss function (Tab: 1). A reason for this is that the lognormal distribution does not model the CLV density well for this data set. This is illustrated in Fig. 4. A Q-Q plot measured the quantiles of the observed CLV values against what the quantiles should theoretically be under a chosen distributional assumption. The closer the points are to the 45 degree straight line, the better the fit. It can be seen in Fig. 4 that a lognormal assumption to model CLV is deficient when compared to the Inverse Gamma distribution. This is seen by a closer adherence of the Inverse Gamma points to the straight line.

5.2. Google F.F. with ZIIG Loss Function

The test set MSE and NRMSE results for all models aggregated across training splits for every cut off date are presented in Table 1. The baseline Google F.F. which uses the MSE loss function was beaten by every model except for the Dropnet using zero-inflated inverse gamma (ZIIG) loss with SMOTE data augmentation. An interesting result is the 27% and 38% improvement in MSE and NRMSE respectively when using the ZIIG loss function compared to the original author's use of MSE. The accuracy of predicting zero-value CLV customers changed nominally, but the performance in predicting the non-zero CLV customers improved dramatically. The reason for this is that the inverse gamma probability distribution captures the tail behaviour of the CLV distribution well (see Figure 4). Despite the improvement, the Google F.F. architecture did not give competitive results from an overall perspective or by customer group shown in Table 2.

5.3. Tabnet

The Tabnet architecture was the fifth best model overall. This is a surprising result as the authors Arik & Pfister concluded in their research that Tabnet outperformed decision tree based models and other neural network architectures in non-performance saturated tabular data prediction tasks. This was not the case in this scenario. We hypothesise this could be due to the fact that in the scenarios explored by Arik & Pfister, there was a higher presence of categorical variables with many levels per variable. As discussed in §3.2, the way in which categorical variables and/or product descriptions are treated in this context is different from tabular data scenarios explored by the authors of Tabnet. SMOTE was not applied in conjunction with Tabnet because Dropnet without SMOTE outperformed Tabnet in terms of overall performance (Table 1) and in all customer groups except for the zero-valued customers where only Dropnet versions 2 & 4 surpassed Tabnet. Due to resource constraints, SMOTE was only investigated with the most competitive deep learning model which was Dropnet.

5.4. Dropnet

The Dropnet model was developed specifically for this CLV estimation task in mind. A 40% dropout rate was used. This masks 40% of the input features being passed to the network for every training epoch. Originally, a dropout rate of 70% was used because this would mimic the number of random features selected for training used in Random Forests for regression problems which corresponds to $\frac{\text{InputFeatures}}{3}$ for each tree (Genuer et al., 2008). This was too aggressive and through experimentation, a dropout of 40%, which corresponds to a random selection of 60% of features for each epoch, was used. This dropout rate yielded the most stable training and best predictive performance.

Dropnet beat both Tabnet and the other neural network architectures, but Dropnet using MSE loss with SMOTE data augmentation produced the best overall deep learning based prediction results. The use of SMOTE significantly improved prediction accuracy on the vanilla Dropnet model (ID 4) by 13.5% on overall MSE and 30% on overall NRMSE as well as reducing associated standard deviations of performances. The main source of this improvement is in the improved accuracy in CLV predictions on the top 20% valued customers which occurred when our training data sets synthetically over-sampled this customer grouping. Table 2 illustrates a dramatic 24% improvement in CLV prediction accuracy for the valuable Top 20% of customers when using SMOTE with Dropnet compared to Dropnet without SMOTE. These results match up with the

work of Sifa et al. where they used SMOTE to boost prediction performance on a valuable, minority class of customers using Deep Learning methods.

The performance gains achieved by using the zero-inflated inverse gamma loss (ZIIG) function on Google F.F. (§5.2) prompted the investigation of that loss function being used on Dropnet with SMOTE data augmentation (ID 3). This resulted in the prediction on the top 20% value customers experiencing a further increase in predictive performance from 1.47 to 1.22 NRMSE. The bottom 80% error remained the same as the Random Forest Model and Dropnet with SMOTE at 0.41 NRMSE. However, the performance on zero-value CLV customers worsened from 1.11 to 1.18 NRMSE. In improving the non-zero valued CLV prediction performance the zero-valued CLV prediction accuracy suffered.

Dropnet with SMOTE (ID 2) beat the top performing model, the Random Forest, in predicting the zero-valued CLV customers and matched the Random Forest's performance in predicting the bottom 80% of customers. However, the performance of Dropnet was more consistent than Random Forest in predicting the bottom 80% grouping as shown by the lower standard deviation of NRMSE of 0.11 vs 0.2 of Random Forest. Dropnet with ZIIG achieved was the best in predicting the high valued customers, but this was offset by lower performance in other customer groupings which resulted in this version being placed third overall.

5.5. Random Forest

Despite efforts made by us and other researchers, the Random Forest came out as the most accurate model overall. Considering the relatively low dimensional input as well as the low number of unique customers (§3), this might be viewed as an unsurprising result. The Random Forest is clearly the superior model in predicting top 20% of CLV customers by having the lowest score of 1.11 NRMSE.

Decision tree methods such as Random Forests seek to group similar observations into leaf nodes based off of some sort of impurity measure such as the variance of the target variable values in a node. Actively grouping similar customers together is likely what results in the superior top 20% CLV predictions. Since regression estimates are calculated as mean averages of training-set CLV values for each leaf node, by grouping as many top 20% CLV customers together, the mean averages of grouped high valued CLV customers predictions result in higher valued predictions. This means that customers along the right-hand tail of the CLV distribution (Figure 1) are predicted better.

The data set is bootstrapped for every tree grown in a Random Forest. This allows the model to see minority classes more often across trees. This is known as "bagging" (bootstrapped aggregation). SMOTE improved performance on Dropnet in predicting the top 20% of customers and it is theorised that the more intense repeated sampling done by Random Forest allowed this model to outperform Dropnet in this customer category. Therefore, a bootstrapped ensemble of Dropnet with repeated SMOTE sampling may outperform the Random Forest. This is left as future work due to resource constraints.

5.6. Model Convergence Time

The convergence times of each model were recorded over training splits and training set cut off dates. The computer used to train the models had an Intel i7-8550 CPU utilizing 8 GB of RAM with no GPU. Convergence time is defined as the time it takes for a model to reach its lowest *validation set* loss value. It is clear by Figure 6 that the Random Forest and Google F.F. models took the least amount of time to converge. The Dropnet with SMOTE mean convergence time worsened by 15% over Dropnet without SMOTE. This is because of the fact that we generated synthetic data for the top 20% of CLV customers by 500%, due to SMOTE. The use of the ZIIG loss function improved the convergence time of Dropnet **considerably**. The mean convergence time dropped from 87 seconds to 21 seconds. This suggests that this loss function helped to cut down training time.

It is clear that if resource constraints are significant, the use of Random Forest may be the best option. A similar conclusion was reached by Chamberlain et al. in that the slight improvement in predictive performance by deep learning methods did not outweigh the extra financial spend of training. However, the data set was quite small with only 4290 unique customers. Results may differ significantly under different conditions.

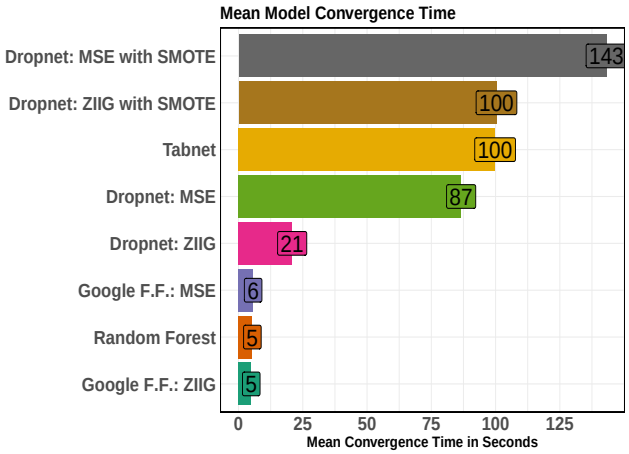

Figure 6. Computational Time for Model Architectures

6. Conclusions

Estimating Customer Lifetime Value (CLV) accurately can provide enormous benefits to business. This is a challenging problem to solve both from a data processing (§3) and a prediction (§5) standpoint. We have observed from this online retail data set that customer level CLV variability and business context differed from our existing research, thus providing a uniform deep learning architecture(s) that generalize well across all business/industry domains is a challenge. However, some important takeaways can be derived from this work.

It is concluded that repeated sampling through SMOTE or bagging significantly improves model performance for positively valued CLV customers. The use of distributional loss functions which match the actual CLV distribution found in the given context can improve both predictive power as well as time to model convergence. For this scenario, we recommend the use of our zero-inflated inverse gamma loss functions for positively skewed CLV distributions like that of Figure 1. A worthwhile avenue of research around CLV estimation would

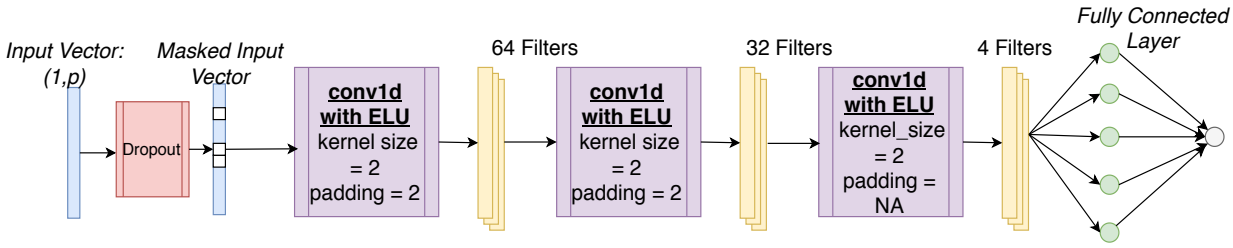

Figure 5. Dropnet Architecture

ID	Model	Cost Fn.	SMOTE	Test Set MSE ± std	Test Set NRMSE ± std	Beat Baseline?
1	Random Forest	MSE	✗	138 ± 29.29	0.76 ± 0.2	✓
2	Dropnet	MSE	✓	168.52 ± 29.63	0.84 ± 0.21	✓
3	Dropnet	ZIIG	✗	174.49 ± 31.77	0.81 ± 0.17	✓
4	Dropnet	MSE	✗	194.9 ± 31.15	1.11 ± 0.35	✓
5	Tabnet	MSE	✗	235.44 ± 45.66	1.25 ± 0.41	✓
6	Google F.F.	ZIIG	✗	243.43 ± 43.3	1.49 ± 0.58	✓
7	Google F.F.	MSE	✗	331.84 ± 55.46	2.42 ± 0.54	**BASELINE**
8	Dropnet	ZIIG	✓	709.76 ± 1944.02	1.27 ± 0.96	✗

Table 1. Overall Model Test Set Performances. ZIIG = zero-inflated inverse gamma, MSE = Mean Squared Error. Results are averaged across cut-off dates and experimental splits.

Customer Group	ID	Model	Cost Fn.	SMOTE	Test Set NRMSE ± std	Beat Baseline?
Zero-Valued CLV	4	Dropnet	MSE	✗	1.07 ± 0.04	✓
Zero-Valued CLV	2	Dropnet	MSE	✓	1.11 ± 0.07	✓
Zero-Valued CLV	5	Tabnet	MSE	✗	1.14 ± 0.09	✓
Zero-Valued CLV	1	Random Forest	MSE	✗	1.17 ± 0.09	✓
Zero-Valued CLV	3	Dropnet	ZIIG	✗	1.18 ± 0.14	✓
Zero-Valued CLV	6	Google F.F.	ZIIG	✗	1.2 ± 0.15	✓
Zero-Valued CLV	7	Google F.F.	MSE	✗	1.22 ± 0.16	**BASELINE**
Zero-Valued CLV	8	Dropnet	ZIIG	✓	1.54 ± 0.66	✗
Bottom 80%	2	Dropnet	MSE	✓	0.41 ± 0.11	✓
Bottom 80%	3	Dropnet	ZIIG	✗	0.41 ± 0.12	✓
Bottom 80%	1	Random Forest	MSE	✗	0.41 ± 0.2	✓
Bottom 80%	4	Dropnet	MSE	✗	0.58 ± 0.32	✓
Bottom 80%	8	Dropnet	ZIIG	✓	0.7 ± 1.06	✓
Bottom 80%	5	Tabnet	MSE	✗	0.78 ± 0.46	✓
Bottom 80%	6	Google F.F.	ZIIG	✗	1.1 ± 0.71	✓
Bottom 80%	7	Google F.F.	MSE	✗	2.39 ± 1.01	**BASELINE**
Top 20%	1	Random Forest	MSE	✗	1.11 ± 0.19	✓
Top 20%	3	Dropnet	ZIIG	✗	1.22 ± 0.2	✓
Top 20%	2	Dropnet	MSE	✓	1.47 ± 0.34	✓
Top 20%	8	Dropnet	ZIIG	✓	1.88 ± 0.54	✓
Top 20%	4	Dropnet	MSE	✗	1.93 ± 0.47	✓
Top 20%	5	Tabnet	MSE	✗	1.93 ± 0.49	✓
Top 20%	6	Google F.F.	ZIIG	✗	2.08 ± 0.63	✓
Top 20%	7	Google F.F.	MSE	✗	3.07 ± 0.58	**BASELINE**

Table 2. Predictive Accuracy Breakdown (NRMSE) by Customer Grouping. Results are averaged across cut-off dates and experimental splits.

thus be development of robust, automatic distribution selection procedures.

We caution against using out-of-the-box methods such as Tabnet which claims superior predictive performance over tree-based or other deep learning methods. The Dropnet architecture presented in this paper showed promise. This emphasises the importance of conducting further research into applying 1-dimensional convolutional networks on tabular data. A suggestion made in §5.5 is research around the use of ensemble methods with Dropnet architectures with a ZIIG loss for tabular prediction problems with a positive-skewed target variable. We believe that this shows great potential to deliver true, state-of-the-art performance from an overall perspective and per customer grouping.

Although the Random Forest, came out as the most accurate and fastest model overall, a recurring theme found in §5 is that of trade offs between different architectures with respect to different customer groupings. Random Forest was better for high-value customers, but Dropnet was much better for predicting the values of customers which do not buy again or very little. Clearly, different business contexts may stress the importance of different customer groups. As a final thought, we alert practitioners to the use of a single, holistic performance measure such as mean squared error for model selection and urge the analysis of model performance on business critical customer groupings.

References

Andon, Paul, Baxter, Jane, and Bradley, Graham. Calculating the economic value of customers to an organisation. *Australian Accounting Review*, 11(23):62–72, 2001. doi: 10.1111/j.1835-2561.2001.tb00181.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1835-2561.2001.tb00181.x.

Arik, Sercan O and Pfister, Tomas. Tabnet: Attentive interpretable tabular learning. *arXiv preprint arXiv:1908.07442*, 2019.

Bojanowski, Piotr, Grave, Edouard, Joulin, Armand, and Mikolov, Tomas. Enriching word vectors with subword information, 2016.

Calciu, Mihai. Deterministic and stochastic customer lifetime value models. evaluating the impact of ignored heterogeneity in non-contractual contexts. *Journal of Targeting, Measurement and Analysis for Marketing*, 17(4):257–271, 2009. doi: 10.1057/jt.2009.19. URL https://doi.org/10.1057/jt.2009.19.

Chamberlain, Benjamin Paul, Cardoso, Angelo, Liu, CH Bryan, Pagliari, Roberto, and Deisenroth, Marc Peter. Customer lifetime value prediction using embeddings. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1753–1762, 2017.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, Jun 2002. ISSN 1076-9757. doi: 10.1613/jair.953. URL http://dx.doi.org/10.1613/jair.953.

Chen, Pei Pei, Guitart, Anna, del Rio, Ana Fernandez, and Perianez, Africa. Customer lifetime value in video games using deep learning and parametric models. *2018 IEEE International Conference on Big Data (Big Data)*, Dec 2018. doi: 10.1109/bigdata.2018.8622151. URL http://dx.doi.org/10.1109/BigData.2018.8622151.

Fader, Peter S., Hardie, Bruce G. S., and Shang, Jen. Customer-base analysis in a discrete-time noncontractual setting. *Marketing Science*, 29(6):1086–1108, 2010. doi: 10.1287/mksc.1100.0580. URL https://doi.org/10.1287/mksc.1100.0580.

Genuer, Robin, Poggi, Jean-Michel, and Tuleau, Christine. Random forests: some methodological insights. *arXiv preprint arXiv:0811.3619*, 2008.

Gupta, Sunil, Lehmann, Donald R., and Stuart, Jennifer Ames. Valuing customers. *Journal of Marketing Research*, 41(1):7–18, 2004. doi: 10.1509/jmkr.41.1.7.25084. URL https://doi.org/10.1509/jmkr.41.1.7.25084.

Gupta, Sunil, Hanssens, Dominique, Hardie, Bruce, Kahn, Wiliam, Kumar, V, Lin, Nathaniel, Ravishanker, Nalini, and Sriram, S. Modeling customer lifetime value. *Journal of service research*, 9(2):139–155, 2006.

Joulin, Armand, Grave, Edouard, Bojanowski, Piotr, Douze, Matthijs, Jégou, Hérve, and Mikolov, Tomas. Fasttext.zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*, 2016.

Malthouse, Edward C and Blattberg, Robert C. Can we predict customer lifetime value? *Journal of interactive marketing*, 19(1):2–16, 2005.

Sifa, Rafet, Runge, Julian, Bauckhage, Christian, and Klapper, Daniel. Customer lifetime value prediction in non-contractual freemium settings: Chasing high-value users using deep neural networks and smote. In *Proceedings of the 51st Hawaii International Conference on System Sciences*, 2018.

Srivastava, Nitish, Hinton, Geoffrey, Krizhevsky, Alex, Sutskever, Ilya, and Salakhutdinov, Ruslan. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. URL http://jmlr.org/papers/v15/srivastava14a.html.

Wang, Xiaojing, Liu, Tianqi, and Miao, Jingang. A deep probabilistic model for customer lifetime value prediction. *arXiv preprint arXiv:1912.07753*, 2019.