Forecasting demand with limited information using Gradient Tree Boosting

Stephan Chang and **Felipe Meneguzzi**

Abstract

Demand forecasting is an important challenge for industries seeking to optimize service quality and expenditures. Generating accurate forecasts is difficult because it depends on the quality of the data available to train predictive models, as well as on the model chosen for the task. We evaluate the approach on two datasets of varying complexity and compare the results with three machine learning algorithms, Results show our approach can outperform these approaches.

1 Introduction

Demand forecasting is an important challenge that many businesses face when assessing their sales strategy (Lu 2014). This is an important problem because it helps in decision making processes that involve the business's resource availability. One such process is the fabrication of products for sale and delivery, in which time to delivery can affect customer satisfaction. Demand forecasting therefore helps deciding whether to produce items before they are purchased, to fasten delivery times, or to withhold production and avoid having items stuck in warehouses. Demand forecasting spans a wide variety of industry sectors, such as energy demand, ATM cash demand and product retailer demands. (Ramos, Santos, and Rebelo 2015; Ma et al. 2013; Catal et al. 2015) In such research, authors use time-series analysis on historical demand data to predict demand values for future time periods. Although effective, these methods only take into consideration temporal information related to the period in which demand occurred. As expected, the predictions are also given in terms of time only. In the case of product fabrication and delivery, this means that, although we obtain a prediction of how much demand originates at a given time, we do not know where it comes from. A logistic challenge therefore remains, in which we must determine how to distribute items to distribution centers closest to customer locations, thus improving delivery times.

In recent years, researchers have attempted to use Machine Learning (Mitchell 1997) methods to solve the problem of demand forecasting (Lu and Kao 2016). Such work shows us that we can use algorithms such as Multi-Layer Perceptron, Support Vector Regression (Chang and Lin 2011) and clustering to predict sales from historical data with results similar to or better than statistical methods. In this work, we experiment with the Gradient Tree Boosting (GTB) (Chen and Guestrin 2016) algorithm on a computer retailer sales dataset. Our objective is to evaluate how well this algorithm can predict sales using limited contextual data, while also considering geographical data with respect to demand origin points. We run a benchmark between GTB, three other Machine Learning algorithms (Linear Regression, Multi-Layer Perceptron (Haykin 2009) and Support Vector Regression) and one basic forecasting technique, and provide a discussion on the results we have obtained with each model.

2 Background

Demand forecasting is the act of using any reproducible method to guess the correct amount of demand for a future time period. Classical demand forecasting methods include state space models, Moving Average and Exponential Smoothing, which have been effectively employed on time series data (Ramos, Santos, and Rebelo 2015; Ma, Fildes, and Huang 2016). Apart from those, there is one basic forecasting technique called *naïve forecasting*, which consists in using the demand value for the current period as the forecast for the next period under the assumption that demand does not change substantially from one period to the next. Due to the simplicity of this method, we use it as a reference to evaluate the performance of other prediction models.

We use Gradient Tree Boosting to solve the regression problem, particularly the *eXtreme Gradient Boosting* (*XGB*) implementation developed by Chen and Guestrin (2016). The idea of Gradient Boosting is to use smaller prediction models to build a more general model that fits the dataset used for training. In XGB, we use decision trees as the smaller prediction model. Decision trees, as the name implies, are useful for separating a dataset into different categories, or *leaves*, according to some decision criteria related to the dataset's distinct column values. Since we are working with a regression problem instead of classification, the leaves of the decision tree are not categories, but continuous valued scores. The value predicted by the model, then, is the sum of the scores pertaining to the leaves that were activated according to the predictor variables.

Consider a dataset $\mathcal{D}$, with predictor variables $\mathbf{X}$ and tar-

gets $\mathbf{y}$. The Tree Boosting prediction model is a function ϕ that maps a set of features $\mathbf{x_i}$ to a prediction $\hat{y}_i$. In turn, the ϕ function is a sum of the weights w of the predictor trees q given the input $\mathbf{x_i}$, given by $\hat{y}_i = \sum_{k=1}^{K} f_k(\mathbf{x}_i)$, where $f_k(\mathbf{x}) = w_{q(x)}$. The idea is that each tree has its own structure q and number T of leaves. According to the input $\mathbf{x_i}$, each tree returns a weight value w, and the algorithm sums all returned weights to compose a prediction. The structure of the tree, as well as the number of trees in the model, depend on an algorithm to find the best splits in the dataset. Although this is an important aspect of building an accurate prediction model using Gradient Tree Boosting, it is outside the scope of this paper to review these split finding algorithms. We do note, however, that due to the dimensions of the dataset we use, we were able to run an exact greedy algorithm, which chooses splits by enumeration.

The learning objective of Gradient Tree Boosting is the minimization of the error between predicted and actual values. In its most basic form, this minimization takes the form $L(\phi) = \sum l(\hat{y}, y) + \Omega(f_k)$, where l is the loss function that represents the prediction error, and $\Omega(f_k)$ is a function that regularizes the learning task to control overfitting. To conform to the *gradient* part of the algorithm, we modify the learning objective by removing the constant loss term and substituting it for the gradient statistics of the loss function multiplied by the prediction function at iteration step t on the input $\mathbf{x}_i$, given by

$$L^{(t)} = \sum_{i=1}^{n} [g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i f_t^2(\mathbf{x}_i)] + \Omega(f_t)$$

In this version of the minimization step, g_i is the first order gradient on the loss function $l(\hat{y}, y)$, while h_i is the second order gradient.

3 Dataset Characteristics

In this section we describe the peculiarities of the Computer Retailer dataset and discuss how we process the dataset to improve both training and the results we obtain.

3.1 Raw Demand Dataset

The raw dataset we received from a computer equipment retailer contains 5 columns: *Year*, *Week*, *Item*, *Location* and *Demand*. Both Year and Week columns come in the form of fiscal weeks, so the first fiscal week of fiscal year 2015 corresponds to the first week of October 2014, while the last week of fiscal year 2015 corresponds to the last week of September 2015. Demand is given in weeks, so the maximum amount of samples for a single year is of 52 for a given item-location pair. This represents a challenge for training, considering the short lifespan of products in the computer equipments market. This limitation extends to the available year values, which are 2015, 2016 and roughly 5 months from fiscal year 2017, amounting to 126 weeks worth of data. With respect to *Item* and *Location* data, we have a variety of 32 items and 50 locations. We can see in the scatter plot of Figure 1 that demand is concentrated on the lower range of the y axis, with a few scattered outliers on the upper range. These outliers represent peaks in demand, standing out from the normal demand distribution, and therefore a possible challenge for learning.

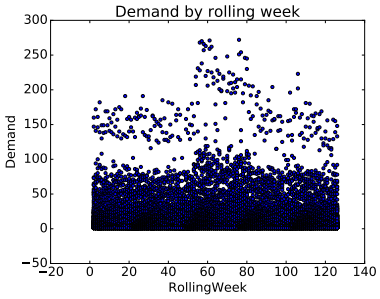

Figure 1: Demand distribution per rolling week.

3.2 Processing the dataset

To augment the dataset, we add a new feature called "Previous Demand" by looking for the last observed demand of a given item-location pair. The idea is to use the naïve forecast as a training component to improve the resulting accuracy of the prediction model by offering it a hint. If the algorithm finds no previous demand, it then removes the row from the dataset, eventually eliminating all 516 rows with missing data (approximately 1.602% of the dataset). Second, we binarize the categorical data of *Item* and *Location* for the machine learning algorithms.

Finally, we smooth the variance of demand values by using natural logarithms instead of raw values (Figure 2), since when the log transformation stabilizes the data, this transformation can improve forecasting accuracy (Lütkepohl and Xu 2012). In the raw dataset, the Variance to Mean Ratio $\frac{\sigma^2}{\mu}$ of demand values is of 37.71. This ratio falls down to 1.01 after applying log transformations, meaning that the variance scales down more than the mean after the transformation, and therefore that the data has become more stable.

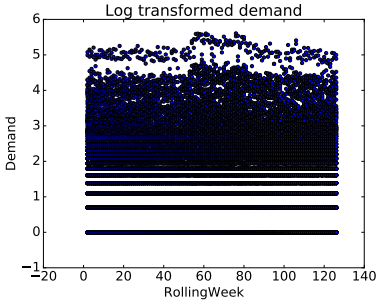

Figure 2: Log transformation of demand distribution per rolling week.

4 Experiments and Evaluation

To evaluate our prediction models, we separate the dataset into two: one part for training and one part for evaluation.

After training our model using the training portion of the dataset, we use the trained model to make predictions for the evaluation dataset. We then compute the differences between our prediction and the actual value in the dataset. In our experiments, we use two years of data for training and our prediction window is of 5 months.

4.1 Experiments

There are different ways in which we can obtain the error measure of a prediction model, and each way has its own peculiarities. We focus on four error measures. Mean Absolute Error (MAE), which is the mean of the absolute differences between predicted and actual values, values closer to 0 are better. Root Mean Squared Error (RMSE), which emphasizes bigger differences on the result, making it more robust than MAE. Mean Absolute Percentage Error (MAPE), which is a scale-independent and easy to interpret measure, as it represents how far our predictions are from the actual value. Lower values indicate closeness to actual values. Mean Absolute Scaled Error (MASE) is a scale-independent error specific for comparing models. Its value is determined by the ratio between the MAE of a candidate model (the one we wish to evaluate) and the MAE of a reference model. A result of less than 1 indicates that the candidate model surpasses the reference model, whereas a result greater than 1 indicates that the candidate model is worse.

We experiment with Gradient Tree Boosting (XGB) on both base and augmented datasets, to see how much we can improve the training process by preprocessing the data, and summarize the results in Table 1. For the Base dataset, we can see that XGB and MLP are both competitive models in terms of their MAPE indices, with MLP showing superior performance in the MAE, RMSE and MASE indices. Although Linear Regression (LR) and Support Vector Regression (SVR) both fall behind Naïve forecasting in both MAE and RMSE, LR still performs better when considering its MAPE index, meaning that LR, on average, deviates less from the actual value. In the Augmented dataset, XGB shows increased performance, surpassing even the MLP model by approximately 5.05% on the MAPE index, and XGB's indices for the Augmented dataset are even lower than MLP's for the Base dataset. This shows that using the naïve forecast in the training process helps XGB to make better predictions. The same, however, does not happen with MLP, as its performance is penalized by the extra feature. MLP is also the only model to suffer from this slight modification to the dataset, since both LR and SVR show a slight improvement on their indices.

4.2 Bike Sharing Demand

From previous experimentation, we conclude that XGB and MLP have more potential than naïve forecasting for the Computer Retailer dataset. There is, however, no gold standard for the referred dataset. To compensate for this shortcoming, we experiment with the same algorithms on the Bike Sharing Demand dataset (Fanaee-T and Gama 2014), which is an open dataset that was used in a Machine Learning competition[1]. Our objective is to compare the models across the two datasets to assess how the quality of the data used for training can affect the models's predictive power. The bike dataset is richer than the Computer Retailer dataset in two ways: (1) it contains more examples to learn from, since it focuses on a single item; (2) there are more predictor variables for us to work with.

According to the results in Table 2, the XGB model is still the one with the best performance, but not like with the retailer datasets. In comparison with naïve forecasting, it shows an improvement of 0.31 in the MASE index. We include the RMSLE (Root Mean Squared Log Error) in these results, because it is the error measure used to score participant entries in the competition. Our RMSLE of 0.3363 is slightly lower than the first place in the public leaderboard of the competition, who won with an index of 0.33757. [2] XGB's MASE index is of 0.69, *versus* 0.72 in the retailer dataset, meaning that XGB brings about similar improvement to both datasets, even though the quality of the training data between them differs.

5 Related Work

Lu (2014) also studies the problem of forecasting computer related product sales using the Multivariable Adaptive Regression Splines (MARS) tool to find the set of predictor variables that yield the best forecasting model. He concludes that the set of ideal predictors may vary according to the product and its sales history and achieves a minimum MAPE index of 15.22% and a maximum of 24.22% for different products in the dataset. Lu's research does not consider, however, the locations from where demand originates and limits the quantity of products to five, making it more manageable to create different models for each product.

Choi et al. (2014) investigate sales forecasting in the fast fashion industry, which shares similarities with our dataset employing Extreme Learning Machines (ELM), which is a type of Neural Network used for regression, and Grey Systems Theory for forecasting time series with limited data and time. The authors separate the general forecasting task into two: forecasting the main time series (trend) using Grey Models, and forecasting the residuals using Extended Extreme Learning Machines. Their insight is that the trend series has a different behavior than that of the residual series, and so using distinct predictive models for each might improve the overall prediction accuracy by first improving the accuracy of these separate models. Their approach is able to achieve MAPE indices ranging from 43.2% to 50.7%.

[1] `https://www.kaggle.com/c/bike-sharing-demand`

[2] We should clarify, however, that our RMSLE is only a reference, because it was not tested in the same manner as in the competition. In the leaderboards, the score is obtained from the participant's predictions for a testing period, in which the competitors do not have access to the answers. Instead, we divided the training dataset and selected 80% of its instances for training, and the remaining 20% for testing, so that we could calculate our own error measures to compare with our other models.

Table 1: Forecasting error for the Computer Equipment Retailer Demand dataset.

	Base				Augmented			
	MAE	RMSE	MASE	MAPE	MAE	RMSE	MASE	MAPE
Naïve	3.95	7.11	–	82.12%	3.95	7.11	–	82.12%
XGB	3.37	6.76	0.85	51.93%	**2.85**	**5.05**	**0.72**	**47.87%**
MLP	**3.00**	**5.77**	**0.76**	**51.33%**	3.39	9.31	0.86	50.42%
LR	6.17	15.96	1.56	73.51%	5.94	37.32	1.50	58.71%
SVR	8.93	20.21	2.26	122.31%	8.05	18.12	2.04	169.17%

Table 2: Forecasting error for the Bike Sharing Demand dataset.

	Bike Demand				
	MAE	RMSE	RMSLE	MASE	MAPE
Naïve	59.92	92.53	0.65	–	60.02%
XGB	**41.55**	**64.69**	**0.3363**	**0.69**	**24.91%**
MLP	157.00	225.51	1.09	2.62	96.81%
LR	157.00	225.46	1.09	2.62	96.83%
SVR	168.89	237.17	1.20	2.82	206.97%

6 Conclusions

In this paper, we have discussed the use of the Gradient Tree Boosting algorithm to solve the demand forecasting problem. We have experimented with this algorithm on two datasets and compared its performance with that of other forecasting techniques and Machine Learning algorithms. One of the datasets pertains to a computer equipment retailer and contains limited data, while the other dataset pertains to the Bike Sharing demand competition and provides richer data for demand prediction. We have seen that the performance of GTB is competitive in both datasets, with it showing better results on the Bike dataset. We attribute this distinctive performance to the richer quality of the data used for training. On the retailer dataset we have seen that, although there is limited information, we can still obtain better predictions from using GTB or Neural Networks than using the baseline naïve forecasting model.

As future work, we will experiment with tweaking the dataset in three ways. First, by adding more features in a similar style to Lu (2014), using statistics pulled from the dataset to increase the number of features. Second, by selecting the most frequently ordered items and creating a prediction model for each, to see if we can improve prediction accuracy for isolated products. Third, by using clustering algorithms to group products and locations with similar behavior, based on the amount of demand per time period, and build a prediction model for each cluster.

Acknowledgments

The authors would like to thank Dell Computers Brazil for funding this research project.

References

[2015] Catal, C.; Fenerci, A.; Ozdemir, B.; and Gulmez, O. 2015. Improvement of Demand Forecasting Models with Special Days. *Procedia Computer Science* 59:262 – 267.

[2011] Chang, C.-C., and Lin, C.-J. 2011. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2(3):27:1–27:27.

[2016] Chen, T., and Guestrin, C. 2016. XGBoost: A Scalable Tree Boosting System. *Computing Research Repository* abs/1603.02754.

[2014] Choi, T.-M.; Hui, C.-L.; Liu, N.; Ng, S.-F.; and Yu, Y. 2014. Fast fashion sales forecasting with limited data and time. *Decision Support Systems* 59:84 – 92.

[2014] Fanaee-T, H., and Gama, J. 2014. Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence* 2(2):113–127.

[2009] Haykin, S. 2009. *Neural Networks and Learning Machines*, volume 10 of *Neural networks and learning machines*. Prentice Hall.

[2016] Lu, C.-J., and Kao, L.-J. 2016. A clustering-based sales forecasting scheme by using extreme learning machine and ensembling linkage methods with applications to computer server. *Engineering Applications of Artificial Intelligence* 55:231 – 238.

[2014] Lu, C.-J. 2014. Sales forecasting of computer products based on variable selection scheme and support vector regression. *Neurocomputing* 128:491 – 499.

[2012] Lütkepohl, H., and Xu, F. 2012. The role of the log transformation in forecasting economic variables. *Empirical Economics* 42(3):619–638.

[2013] Ma, Y.; Wang, N.; Che, A.; Huang, Y.; and Xu, J. 2013. The bullwhip effect on product orders and inventory: a perspective of demand forecasting techniques. *International Journal of Production Research* 51(1):281–302.

[2016] Ma, S.; Fildes, R.; and Huang, T. 2016. Demand forecasting with high dimensional data: The case of SKU retail sales forecasting with intra- and inter-category promotional information. *European Journal of Operational Research* 249(1):245 – 257.

[1997] Mitchell, T. M. 1997. *Machine Learning*. New York, NY, USA: McGraw-Hill, Inc., 1 edition.

[2015] Ramos, P.; Santos, N.; and Rebelo, R. 2015. Performance of state space and ARIMA models for consumer retail sales forecasting. *Robotics and Computer-Integrated Manufacturing* 34:151 – 163.