List Price Optimization using customized decision trees

Shashank Shekhar[1], Kiran R[1], John Kiran[1], Dr. Raghava Rau[1],Sam Pritchett[1], Anit Bhandari[1], Parag Chitalia[1]

[1] VMware Inc.
{ sshekhar, rki, kjohn, drau, spritchett, anitb, pchitalia}@vmware.com

Abstract. There are many data mining solutions in the market which cater to solving pricing problems to various sectors in the business industry. The goal of such solutions is not only to give an optimum pricing but also maximize earnings of the customer. This paper illustrates the application of custom data mining algorithms to the problem of list price optimization in B2B (Business to Business). Decision trees used are typically binary and pick the right order based on impurity measures like Gini/entropy and mean squared error (for example in CART). In our study we take a novel approach of non-binary decision trees with order of splits being the choice of business and stopping criteria being the classical. We exploit proxies for list price changes as discount %age and Special Pricing Form (SPF) discounting. We calculate transaction thresholds, anchor discounts and elasticity determinants for each Stock Keeping Unit (SKU) segment to arrive at recommended list price which gets used by pricing unit

Keywords. Non-Binary Decision Tree, List Price, SPF, B2B, Anchor Discount, classification, regression, entropy, log-loss

1 Introduction

VMware (VMW) is a virtualization, end user computing and cloud company with annual revenues of over USD 6 BB (as of 2015) and a market cap of USD 25 BB [1]. VMware sells products in the Software Defined Data Center (vSphere, VSAN, NSX for computing, storage & network virtualization respectively), end user computing (Airwatch, Horizon, and Fusion/Workstation) and cloud. These are all sold to B2B customers.

The prices of products at VMW have been rarely changed over time. However Sales representatives could offer discounts via SPF flag which a special discretionary discount was typically given to large orders. The Pricing Business Unit was keen to figure out a way to get to optimal list prices at VMW.

1.1 Objectives

The Advanced Analytics & Data Sciences team came up with the following objectives with the Pricing Business Unit.

- Analyze historical prices and related volume movements and understand major drivers of discount % and SPF (special pricing form/sales person specific discount) Flag. Since, the prices haven't changed much historically, discount and SPF have been considered as proxies to list price changes.
- To come up with recommended list price along with level of confidence for all SKUs

2 Solution Framework

2.1 Strategy

A traditional list price optimization would use price changes versus quantity changes, but we never change prices. Discounting practices are an alternative inference method but requires additional steps.

Discount% and SPF requests are useful indicators to infer the customer's assignment of value to a product. Segments will be arrived at for discount % and SPF flag. SPF flag is a discount that can be given by a sales representative on request – mostly given on large order sizes. List price is used to measure VMware's assignment of value to a product. Imbalance between assignments of value then indicate tension in List Price. Following detailed steps were planned –

- Understand segments of the order-SKUs based on discount percentage and also SPF Usage
- Understand the importance of factors that drive variation between segments and within segment
- Incorporate explicit price-volume elasticity algorithms and come up with pseudo elasticity determinant

2.2 Segmentation Approach

Objective was to arrive at segments of order-SKUs based on discount% and SPF usage (proxies to list price changes). Business required that the decision tree be non-binary in nature and in a particular order. Non-Binary decision trees were built to create segments to explain discount% and SPF flag. Product platform, Order Size, Local Currency, Industry Vertical, Partner Tier were used in the same order. Uniqueness of the tree is that non-binary tree implementation that has us determine the order of the input variables based on business perceived improvements.

Working of a normal decision tree – Decision trees [2] in standard software packages decide the list of features and the order of features based on mathematics related to mean squared error, entropy, log-loss, Gini [3] etc.

The customized decision tree made during this study was needed to be built from the leaf node and up using packages in R [4]. It takes the features in the same order required by the business and groups the order-SKUs accordingly. The standard splitting criteria is now replaced by the custom criteria required by the business. The stopping criterion is minimum number of observations. These kind of custom trees are neither available as part of any standard package nor have a custom splitting and stopping criteria. It also uses hand-built mean squared error and log-loss for regression and classification trees respectively. Price elasticity based on above didn't yield significant results and will be trying to arrive at recommendations based on segments of a single product. We will correlate discount % and SPF Usage for segments against their characteristics and use insights from the above segmentation to arrive at list price recommendations at product platform level.

3 Modeling

3.1 Dataset Creation

The data matrix for the model was aggregated at an order-SKU level using Greenplum and Hadoop. This is because our entity is the combination of an order and a SKU. Specific business level judgements were applied like:

- dataset for a time period of FY13Q1 to FY14Q2 with license only transactions for only partner channel
- Specific products like vCloud Air that had a subscription model/enterprise license agreements (ELAs) that have pre-determined prices transactions while creating this dataset
- Given large amounts of data, the data was picked from Hadoop, where we took last 5 years of dataset related to site made available by IT in a flat tabular structure at order-SKU level.
- The continuous variables were subjected to coarse classing and fine classing making all the features categorical.

The key independent variables were Product platform, order size group, industry vertical currency code and partner type tier and the dependent variables were Discount percentage (for custom regression tree) and Special Pricing Form (SPF) flag (for custom classification tree).

The goal in our analysis is to arrive at elasticity of list prices by using the following as proxy:

- By using discount %age as a proxy
- Using the special discounting that sales reps give as an indication of the list price elasticity

Subsequently regression trees and classification trees were built to arrive at the above. We developed a unique algorithm that builds non-binary trees and shows all possible combinations of the features to the business. Businesses like white-box models and also control the order when the mathematical uplift generated by a feature is marginal. Our supervised algorithms custom-built allow this

3.2 Custom Regression Tree for Discount Percentage

Features for the regression tree were chosen in the same order of importance as following: product platform, order size group, industry vertical, currency code, and partner tier.

We found *248* segments with their measures which were used to identify orders where discount % is very different from the segment mean. It was also used to group segments. The tree was a non-binary which used *mean-squared error minimization* [5]. It also can split the dataset using an order of variables/features that business desire. The attributes used to identify segments are mean discount %, proportions of order with SPF flag, # transactions, average booking amount and average unit list price in USD.

Table 1. Mock-up Representation of Regression/Classification tree data

Segment	Order Number	Product SKU	Actual Discount %	Segment Mean Discount %
VCLOUD SUITE UPGRADED	13499196	CLX-VXXX-STD-UG-AXXX	-20%	-34%
VCLOUD SUITE UPGRADED	13471982	CLX-VXXX-STD-UG-BXXX	-20%	-34%

3.3 Custom Classification Tree for SPF flag

The feature order of importance was same as in the custom regression tree. We found *188* segments with their measures which were used to identify orders where proportion of SPF is high/low. It was also used to group segments. The tree was a non-binary which used *entropy for minimizing classification error*. It also can split the dataset using an order of variables/features that business desire. The attributes used to identify segments are mean discount %, proportions of order with SPF flag, # transactions, average booking amount and average unit list price in USD.

The outputs based on the decision tree approach using classification and regression [6] were joined back to our original dataset. Hence, we have two additional fields in our dataset – Discount Segment and SPF Segment. These were utilized in later part of the project

3.4 Determining Relative Importance

The relative importance of the factors in the trees were determined by running custom regressions [7] on the predictions from the two decision trees. The relative importance of factors in the decision tree based on discount % was determined using linear regression while logistic regression was used for the decision tree based on SPF usage. The relative importance of the variable were determined for all the depth of the tree and for each splitting rule. Below are the order and magnitude of relative importance of the independent variables at the final depth of the tree:

Table 2. Relative Importance of Factors

Splitting Variable	**Average Varlogloss**	**Relative Importance**
Product Platform	0.22312171	30%
Order Size Group	0.22527636	27%
Industry Vertical	0.20068196	15%
Currency Code	0.18544909	14%
Partner Type Tier	0.22435609	14%

3.5 Comparison with Binary Decision trees

Table 3. Overview of comparison of Customized Decision Tree with Standard Binary Tree

Bucket	**SI#**	**Binary Tree**	**Our custom Non-Binary Tree**
Development	1	Only 2 splits at each level	Multiple splits at each level
	2	Same feature might get repeated many times from root to leaf	Same feature cannot get repeated from root to leaf more than once. This explains the business logic better
	3	The features are chosen based on impurity reduction	Full tree is exhaustively built for all combinations and presented as a choice to the business
	4	No overriding of what the algorithm choses	Allows business user to override feature splitting when the mathematical efficiency from split is marginal

	5	Stopping criteria is # of observations OR impurity criteria	Same here
Performance	1	Traditionally binary trees provide better performance than non-binary trees	Our algorithm is able to provide comparable accuracy of binary trees despite being non-binary with the added benefit of business overriding
Implementations	1	Custom implementations exist in R	No such implementation exists in R or in any other software package

A typical binary decision tree which works on the principle of splitting a node into two child nodes repeatedly, beginning with the root node that contains the whole learning sample uses standard splitting [6] (as an example CART includes least squares and least absolute deviation for regression trees and Gini, entropy, Symgini, twoing, ordered twoing and class probability for classification trees) and stopping criteria which will not give the analyst the ability to choose the features/attributes in the order desired by the business. The customized decision trees which were built for the project used custom splitting criteria which enables the analyst to select the order of the features used for splitting observations. As an example, when the standard binary tree was run on the same dataset, it gave order size group as the most significant variable and was the parent node whereas the business wanted the product platform to be the parent node. The rationale behind this was that discount and SPF are bound to be higher for the SKUs in bigger order size group and the business wanted to evaluate the behavior of the SKUs within the product platform. If the order size group is the parent node then the SKUs will spread across different segments and the discounting behavior cannot be evaluated at product platform level. Also, unlike a standard binary decision tree, the customized decision tree didn’t have same number of splits at each level and hence a variable (for example product platform) appeared only once and not more than once as was the case in standard binary decision tree. This applies to all the levels of the customized decision tree and the user has the flexibility to choose desired variable at any level.

The customized decision tree algorithm has been designed to handle large datasets. The algorithm utilizes the following stopping rules in order to control the tree growing process and make the mechanism efficient:

– If the size of a node is less than the user-specified minimum node size (2000), the node will not split.
– If the impurity of the child node is greater than the impurity of the parent node, the node will not split
– If the current tree depth reaches the user-specified maximum tree depth limit value, the tree growing process will stop

4 Utilizing Modeling Outputs

4.1 Identifying Anchor Product

A target SKU was selected based on overall bookings and profitability. This SKU was primarily the top selling SKU. *Selecting anchor SKU was a mutual exercise and business judgements were used to identify the SKU.*

4.2 Generating Model Parameters

Transaction Cutoff: The rationale to have a transaction cutoff was to set a minimum number of transactions required for the model to make a recommendation

$$T(Q_i) = \text{mean(transactions)} - \sigma \qquad (1)$$

Capped List Price percentage change: Sets a maximum recommendation as % change from original list price. This was set at 15% after due consultation with the pricing team on what is an acceptable limit
List Price Integer rounding: Rounds recommendations to make value appear more similar to common rounding practice. This was done in alignment to the corporate standards. In this particular case, we rounded to the nearest $5/LC5.

4.3 Calculating Anchor Discount

One of the top selling SKUs was selected as the anchor product. The mean of all the discount % was calculated for the selected SKU. The model was evaluated iteratively using the mean of discount %age. An adjustment was made/error was added to the mean to come up with the anchor discount. Adjustment was made to ensure that the anchor discount satisfies the ranges from the pseudo elasticity determinant.

$$\text{Anchor Discount} = \text{Mean of Discount \%} + \text{Adjustment Amount} \qquad (2)$$

4.4 Pseudo Elasticity Determinant

Regression of sum of quantity to gross per unit USD value for different product platform and different pricing and SPF segments. The goal was to arrive at a pseudo elasticity determinant and conform that the change in list price doesn’t result in decrease in volume for certain important segments.

4.5 SPF % Intercept and Slope

Expected SPF intercept and slope are used to determine what SPF usage should be expected based on a SKU's list price. The intercept and slope came out of a regression of SPF usage on the list price for selected representative SKUs.

$$\text{SPF Usage} = \alpha + \beta * \text{List Price} \tag{3}$$

$$\text{SPF Intercept} = \alpha \tag{4}$$

$$\text{Slope} = \beta \tag{5}$$

4.6 Expected SPF

This was calculated for each SKU based on the regression equation generated from the regression of SPF usage on the list price for selected representative SKUs.

$$\text{Exp. SPF of SKU} = \alpha + \beta * \text{List Price of SKU} \tag{6}$$

4.7 SPF Flag

This was generated as an indication to whether SPF usage for a SKU is recommended or not. There are three different cases possible for SPF Flag based on following criteria:

$$\text{Flag 'YY'} - \text{if expected SPF} > \text{Actual SPF} + 10\% \tag{7}$$

$$\text{Flag 'Y'} - \text{if expected SPF} > \text{Actual SPF} + \text{Delta } 10\% \tag{8}$$

$$\text{Flag 'N'} - \text{if expected SPF} < \text{Actual SPF} + \text{Delta } 10\% \tag{9}$$

Delta % has to be entered by the business user and is capped at 10%. For current exercise, we took it as 0%.

4.8 Suggested List Price

This is the suggested list price without any adjustment and rounding off.

$$\text{Suggested LP} = \frac{\text{LP x (1-Actual Discount)}}{\text{(1-Anchor Discount)}} \tag{10}$$

4.9 Recommended List Price

The recommended list price was arrived at after applying the transaction cut-off constraint, capped list price percentage change constraint and the list price integer rounding.

Table 4. Model Parameter / Assumptions for Mock-up Calculation of Recommended List Price

Model Parameters/ Assumptions	Values (Product 1)
D_A = Anchor Discount	35%
T_C = Transaction Cut – Off	50
I_{LP} = Capped List Price Increase	15%
LP_R = Rounding to LP to the nearest	$5
SPF_{IN} = SPF % Intercept	0.90%
SPF_S = SPF % Slope (per $1,000)	$0.0000295
SPF_δ = Delta SPF% (capped at 10%)	0%

Table 5. Model Parameter / Assumptions for Mock-up Calculation of Recommended List Price

Model Parameters/ Assumptions	Values (Product 1)
Product Platform	ABC+
Product SKU	VS-ABC-PL-C
LP_{US} = US List Price	$3,495
Rev_{FLAT} = % Platform Revenue	83.40%
Rev_{GROSS} = Gross Revenue	$113,636,152
#tx = Transactions	9,965
Rev_L = List Revenue	$172,292,318
$Disc_E$ = Extended Discount	$58,656,166
Disc % = % Discount = $Disc_E$ / Rev_L	34.04%
SPF_w = Weighted SPF	6.0%

Table 6. Mock-Up Calculation for Estimated Value of List Price

Model Parameters/ Assumptions	Values (Product 1)
SPF_{EXP} = Expected SPF % = LP_{US} x SPF_S+SPF_{IN}	11.21%
SPF Flag	Y
$LP_{Suggested}$ = LP_{US} x (1 – Disc %) / (1-D_A)	\$3,546.37
LP_δ = List Price % Delta = ($LP_{Suggested}$ / LP_{US}) - 1	1.47%
LP_{Capped} = Round(If(LP_δ>LP_{US}, LP_{US} x (1 + I_{LP}), $LP_{Suggested}$)	\$3,546
$LP_{Recommended}$	\$3,545

5 Conclusion

The solution provided for optimizing the list price at VMware Inc. is unique and distinctive because there is hardly any instance of price change at VMware Inc. The custom non-binary decision trees that were built in the first phase of the project are piece of IP creation as there is no existing software/package which builds non-binary decision trees. Also, the implemented approach was very different from standard price optimization methodologies where historical price changes are utilized to arrive at recommended price changes.

From an algorithmic perspective, the approach is unique and there is no such implementation that involves non-binary trees, allows user to override certain decisions in tree building that explain the business processes better. It also allows to see the different combinations of tree building and customized choices. Also, a traditional list price optimization would use price elasticity (price change versus quantity change) but since the list price at VMware Inc. haven't changed significantly, we utilized the segments from the customized decision tree to compute the pseudo elasticity and preclude any decrease in quantity.

The recommendation to change the list price of about ~40% of the SKUs was given to the business. The recommended list price was calculated in local currency for each country and the business was suggested to start with modification to high confidence products. It was also recommended to make the list prices changes independent of other changes as associating a list price change with a version increase will make it harder to directly measure the customer's perception of value. One of the key endorsement was to control the discounting process and enable automated contractual discounts and clear the rules and boundaries. The current method of arriving at the list price utilizes the method of pseudo elasticity determinant. The recommendations from this project will help track the actual list price changes to actual unit quantity changes

References

1. VMware Inc. Annual Report, http://ir.vmware.com/annuals.cfm
2. Quinlan, J. R., (1986). Induction of Decision Trees. Machine Learning 1: 81-106, Kluwer Academic Publishers
3. Shannon, Claude E. (July–October 1948). "A Mathematical Theory of Communication". Bell System Technical Journal 27 (3): 379–423. doi:10.1002/j.1538-7305.1948.tb01338.x. (PDF)
4. M Dowle, A Srinivasan, T Short, S Lianoglou with contributions from R Saporta, E Antonyan https://cran.r-project.org/web/packages/data.table/data.table.pdf (PDF)
5. Hastie, T., Tibshirani, R., Friedman, J. H. (2001). The elements of statistical learning: Data mining, inference, and prediction. New York: Springer Verlag
6. Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984). Classification and Regression trees. Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software. ISBN 978-0-412-04841-8
7. Cohen, J., Cohen P., West, S.G., & Aiken, L.S. (2003). Applied multiple regression/correlation analysis for the behavioral sciences. (2nd ed.) Hillsdale, NJ: Lawrence Erlbaum Associates