Price Arbitrage for Mom & Pop Store Product Traders

BADM Final Project Report

Group 7B:

Abhinav Verma (PGID: 61310725)

Bhaskar Vedula (PGID: 61310424)

Raphael Schilling (PGID: 61319001)

Rhianna Prabhu (PGID: 61310319)

Vivek Aranha (PGID: 61310554)

Table of Contents

EXECUTIVE SUMMARY

Problem Description

Currently Mom & Pop Stores do not have much insight into the price changes of electronic products. This information is useful for the stores to plan for inventory purchases prior to the price increases and generate profits on the products due to the increased price. Currently, ***this decision to buy is based on intuition and there is no real science or reasoning behind it***. Traders' buying patterns currently ***lead to increased inventory or missed opportunities*** due to incorrect price change assumptions.

Business Goal

We are consultants to Mom & Pop Stores, and our goal is to help them get more reliable information regarding the price changes thus resulting in increased profits and lower inventory holding costs. The Mom & Pop stores could make profits by buying products through Bargain.in that are likely to have a price increase within the next 24 hours. These product traders can utilize information asymmetry to sell at high prices to their customers and would thus be able to take advantage of the price arbitrage opportunity.

Data Mining Goal

The data mining goal is to ***recommend the top 10 brands, across various products, to the Mom & Pop Stores for them to sell in the offline mode***. We would ***analyze various data mining models and select the model with the least error rate*** while also keeping the costs in mind. The task would be a supervised task with prediction of 'Price Up' and classification based on various input parameters like Brand, Day of Week, Average rating, to name a few.

Data Description

The raw data includes the price listings of 30 brands of mobile phones and cameras. The fields of this data are given below with their respective description:

Name	Description
RowID	index number
name	Product name
brand	Brand of the Product
color	Color of the Product
freeShipping	1 = YES, 2 = NO, 0 = Data Unavailable
inStock	1 = YES, 2 = NO, 0 = Data Unavailable
avRating	Average rating of the product
reviewCount	No. of users who rated the product
listPrice	Price of the product on "date"
shippingPeriod	Shipping period of the product
date	Timestamp of the product and price information (mm/dd/yyyy hh:mm)
siteName	Name of the website from which the product is sold
category	Category of the product
TimeNextPrice	number of days until the next available price information
group	always a value of 1 (required for administrative reasons).
PriceUp	whether the product price went up ("1") or not ("0") on the next time the price information is recorded ("TimeNextPrice")

The rows in our data are differentiated by the 'date' column as there would be multiple price checks for the same product on different dates and times.

RowID	name	brand	color	freeShipping	inStock	avRating	reviewCount	listPrice	shippingPeriod	date	siteName	category	TimeNextPrice	group	PriceUp
291	Apple iPhone 5 - 16GB (Black)	Apple	Black	1	1	4	5	45500	5-7 business days	11/7/2012	Infibeam	Mobiles	1.00	1	0
504	LG Optimus L3 E400 (Black)	LG	Black	1	1	5	5	7129	3-5 business days	10/16/2012	Infibeam	Mobiles	0.58	1	0
1445	Karbonn K 9	Karbonn	Black & Red	1	1	3	4	1710	3 Business Days	10/17/2012	Saholic	Mobiles	7.95	1	1
1656	Lava ARC04	Lava	Black	1	1	3	3	1485	3 Business Days	10/17/2012	Saholic	Mobiles	10.95	1	0

After obtaining this data from Kaggle, we followed a few steps to process it further

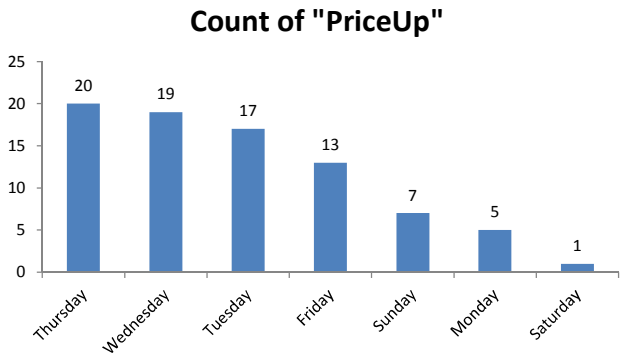
Step 1 –Filling Missing Values: Given below is the treatment for each of the columns which had missing values

Name	Treatment
color	Replace 'blank' with 'NA'
avRating	Replace 'blank' with average of '3.94' (average across non blank rows)
reviewCount	Replace 'blank' with '0'
shippingPeriod	Replace '0' and 'blank' with 'NA'

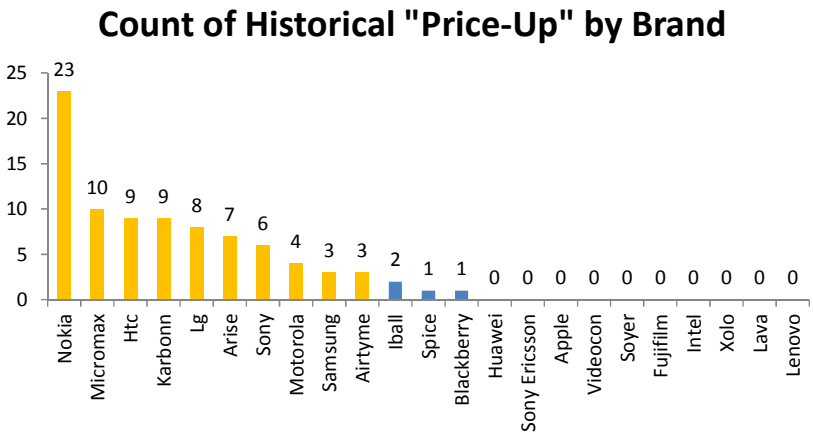
Step 2 – Filtering Data: We filtered out data which had the 'TimeNextPrice' column greater than 1. This was done to ensure that the trader is able to know about the price change as soon as possible. This will help reduce the traders' inventory holding costs, since they would not have to wait long to find out their purchased products' updated prices.

We also filtered out products with an 'instock' value of 2. This was done to ensure that the products recommended were in stock.

Step 3 – Adding Columns: We added a categorical column 'DayOfTheWeek' to include the impact of the change in price based on the day of the week in our data mining model. As is shown below, certain days showed more historical price increases than the rest:



Step 4 – Dimension Reduction: We reduced the number of brands to 10 based on the historical count of 'PriceUp'. This was done to differentiate brands with a higher chance of price increase from the rest and to reduce the number of variables in the model (to avoid over fitting). The graph below shows the brands with the number of price changes for each of them. The ones in yellow were included in the model:



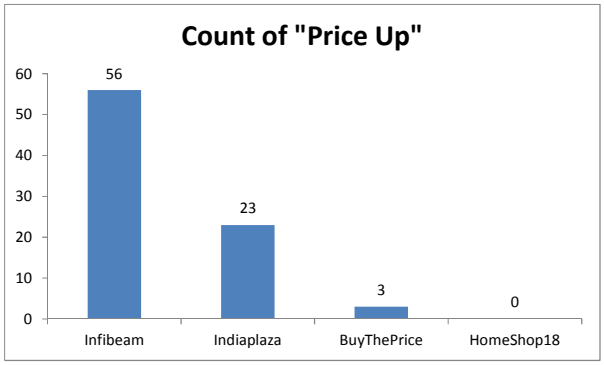
DATA MINING SOLUTION

Model Building

To achieve our data mining goal, we used a categorical variable as our output. This was defined as 1 = Price Increase and 0 = No Price Increase.

To obtain our data mining goal, we first needed to select predictor variables to determine whether the price of the shortlisted 10 brands was likely to increase. To do this, we used domain knowledge and the CART method on the numerical predictors to select the most important variables which we found to be 'ListPrice' and 'avRating'. This can be seen in the classification tree in Exhibit 1 of the Appendix.

Amongst the categorical predictors, we selected 'DayOfTheWeek' for reasons mentioned in the previous section. We also assigned numerical values to these days (1 to 7) to ensure they fit into our model. The final predictor we selected was 'siteName' since we observed that certain sites had more price changes than the others. This is shown below.



Methods Used

We used three predictor classification models (including benchmark) to determine whether or not to recommend a basket of 10 brands to product traders, and chose the best model based on lowest error rates and costs. There methods were:

1. Naïve Method: We used this method as our benchmark since it was easy to compute. Since "Price Up" = 0 was in majority in the data, we calculated the error rate of naively predicting a 0 for "Price Up", when it was actually 1. This gave us a 3% error.

2. Logistic Regression: This method was used as our output was a "categorical variable". For this model, we used 6 predictor variables (Average Rating, List Price, Day of Week, and 3 dummies for SiteName) and set the cut-off probability of success as 0.5. The output from the model is shown in Exhibit 2 of the Appendix. As can be seen, the error rate from this model on the test data was 2.51%, which was slightly better than the Naïve method.

3. Naïve Bayes Model: To check whether we could get an error rate lower than that for Logistic Regression, we ran this model on the same input and output variables. This gave us the same

error rate as Logistic Regression on the Test Data, but a slightly higher error rate than Logistic Regression on the Validation data (2.31% vs. 2.23%). For this reason, we chose the Logistic Regression as the most accurate model to recommend the short-listed brands to the trader. Having said this, the trader is also welcome to use the Naïve rule, as it might be computationally cheaper and easier to process. However, the Naïve rule would lead to higher costs as shown in Exhibit 3 of the appendix.

CONCLUSION AND RECOMMENDATIONS

Based on the error rates observed in the three models, we chose the Logistic Regression model to ***help Mom and Pop stores "buy online" and "sell offline" within a day***, thereby making profits. Out of the 10 short-listed brands that the model would classify as a "Buy", the traders can create a portfolio of products to buy depending on their customer demand. While this model will benefit the traders to a great extent, we observed certain risks in the methodology and model:

- Change in customer preferences / Products may become obsolete – While a "Samsung Galaxy" phone may have increased in price tomorrow, customers may not be available for the traders to sell to. This would increase the inventory holding cost of the trader
- Model / Error Costs – In Exhibit 4 of the Appendix, we have shown the costs associated with false-positive (recommending a purchase, when price does not actually increase) and false negative (not recommending a purchase when prices actually increase) results. In doing so, we have ***assumed an industry wide margin of 6%*** for mom & pop store traders, and electronic goods industry-wide ***average holding cost of 8%*** of listed price. As can be seen, a 2-3% error can be converted into Rs. 70,000 – Rs. 110,000 in costs for the trader, which ***while lesser than traditional trader costs***, ***are still pretty high***.
- Model Generalization Issues - Our model will need a macro to automatically select the top 10 brands based on historical "Price Up" count every time. This way, if Bargain.in expands its product portfolio going forward, the ***model can be generalized to other brands to always pick 10 brands (across products) that have the highest probability of an increase in price***
- Out of Stock Issues – There is a risk of ***recommended products being out of stock*** at Bargain.in. This can lead to lost profits for the trader

APPENDIX

Exhibit 1: CART Classification Tree

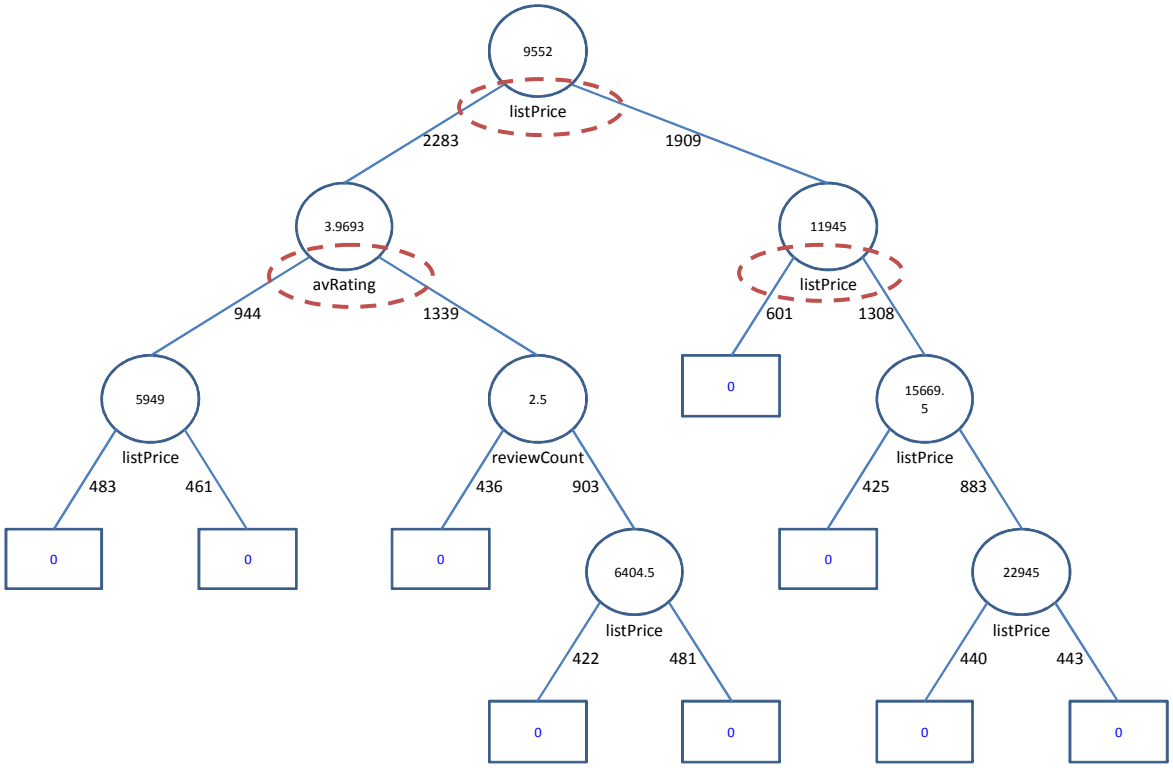


Exhibit 2: Logistic Regression Model Output

Training Data scoring - Summary Report

Cut off Prob.Val. for Success (Updatable)	0.5	(Updating the value here will NOT update value in detailed report)

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	0	33
0	0	2063

Error Report			
Class	# Cases	# Errors	% Error
1	33	33	100.00
0	2063	0	0.00
Overall	2096	33	1.57

Validation Data scoring - Summary Report

Cut off Prob.Val. for Success (Updatable)	0.5	(Updating the value here will NOT update value in detailed report)

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	0	28
0	0	1230

Error Report			
Class	# Cases	# Errors	% Error
1	28	28	100.00
0	1230	0	0.00
Overall	1258	28	2.23

Test Data scoring - Summary Report

Cut off Prob.Val. for Success (Updatable)	0.5	(Updating the value here will NOT update value in detailed report)

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	0	21
0	0	817

2.5%

Error Report			
Class	# Cases	# Errors	% Error
1	21	21	100.00
0	817	0	0.00
Overall	838	21	2.51

Exhibit 3: Naïve Bayes Model Output

Training Data scoring - Summary Report

Cut off Prob.Val. for Success (Updatable)	0.5	(Updating the value here will NOT update value in detailed report)

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	1	32
0	0	2063

Error Report			
Class	# Cases	# Errors	% Error
1	33	32	96.97
0	2063	0	0.00
Overall	2096	32	1.53

Validation Data scoring - Summary Report

Cut off Prob.Val. for Success (Updatable)	0.5	(Updating the value here will NOT update value in detailed report)

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	0	28
0	1	1229

Error Report			
Class	# Cases	# Errors	% Error
1	28	28	100.00
0	1230	1	0.08
Overall	1258	29	2.31

Test Data scoring - Summary Report

Cut off Prob.Val. for Success (Updatable)	0.5	(Updating the value here will NOT update value in detailed report)

Classification Confusion Matrix		
	Predicted Class	
Actual Class	1	0
1	0	21
0	0	817

Error Report			
Class	# Cases	# Errors	% Error
1	21	21	100.00
0	817	0	0.00
Overall	838	21	2.51

Exhibit 4: Error Cost Comparison (Based on Lost Profits and Trader Holding Costs)

Cost of Lost Profits - Logistic Reg.	Items
Data Points	4,192
Model Error	2.51%
Products the trader should have bought, but didn't buy (Units)	105
Average List Price (across 10 brands)	11,009
Trader Margin	6.0%
Lost Profits	69,503

Inventory Holding Cost - Logistic Reg.	Items
Data Points	4,192
Model Error	2.51%
Products the trader bought, but should not have bought (Units)	105
Average List Price (across 10 brands)	11,009
Inventory Holding Cost	8.0%
Lost Profits	92,670

Cost of Lost Profits - Naïve Rule	Items
Data Points	4,192
Model Error	3.00%
Products the trader should have bought, but didn't buy (Units)	125.8
Average List Price (across 10 brands)	11,009
Trader Margin	6.0%
Lost Profits	83,071

Inventory Holding Cost - Naïve Rule	Items
Data Points	4,192
Model Error	3.00%
Products the trader bought, but should not have bought (Units)	126
Average List Price (across 10 brands)	11,009
Inventory Holding Cost	8.0%
Lost Profits	110,761

Reduced Costs from using LR over Naïve	13,568

Reduced Costs from using LR over Naïve	18,091