Predicting Product Choice with Symbolic Regression and Classification

Philip Truscott
Southwest Baptist University
Michael F. Korns
Korns Associates

Abstract

Market researchers often conduct surveys to measure how much value consumers place on the various features of a product. A good choice model should enable managers to combine these utility values in different ways to predict the market share of a product with a new configuration of features. Researchers assess the accuracy of these choice models by measuring the extent to which the summed utilities can predict actual market shares when respondents choose from sets of complete products. The current paper includes data from 201 consumers who gave ratings to 18 cell phone features and then ranked eight complete cell phones. A simple summing of the utility values predicted the correct product on the ranking task for 22.8% of respondents. Another accuracy measurement is to compare the market shares for each product using the ranking task and the estimated market shares based on summed utilities. This produced a mean absolute difference between ranked and estimated market shares of 7.8%. The current paper applied two broad strategies to improve these prediction methods. Various evolutionary search methods were used to classify the data for each respondent to predict one of eight discrete choices. The fitness measure of the classification approach seeks to minimize the Classification Error Percent (cep) which minimizes the percent of incorrect classifications. The forms of classification included classification and regression trees (CART), neural networks, decision trees and non-linear discriminant analysis. A regression approach treated the dependent variable as a continuous variable rather than a set of categories. This produced a significantly better fit with the hit rate rising from 22.8% to 35.8%. The mean absolute deviation between actual and estimated market shares declined from 7.8% to 6.1% ($p. < 0.01$).

Introduction

Market research has a long history of attempting to evaluate the importance of product features so that brand managers can predict the popularity of new feature combinations. Some of these methods require respondents to consider a set of products and place them into a rank ordering. Since these survey methods require the respondents to assess all of a product's features jointly the methodology has been termed 'Conjoint' (Green & Rao, 1971). After the respondents have ranked the complete configurations the utility values of the individual product features are calculated using multinomial logit.

Another approach requires respondents to explain how much value they place on each feature separately. This 'self-explicated' approach generates utility values directly (Marder, 1997; Srinivasan & Park, 1997).

Both conjoint and self-explication methods then incorporate the utility values into predictive models. The utilities are re-combined to play 'what-if' games that predict the market share of future product offerings. The most common technique for assessing the accuracy of these choice models is to follow the conjoint or self-explication survey task with a validation task. This involves asking the respondents to rank order a set of products. The goal is to find methodology that is able to predict a high proportion of the top products in the validation task. A high 'hit rate' substantiates the accuracy of a given methodology.

The Experiment

For the current research, a self-explication survey collected data from 201 Indian consumers. The specific form of rating has been termed the Un-bounded Write-in Scale (UWS) because respondents may give rating numbers without upper or lower limits(Marder, 1997). A Web page tells them to click:

a) a plus sign button as many times as they want to show how much they like a feature
b) a minus sign button as many times as they want to show how much the dislike a feature
c) a zero button to indicate that they are neutral about a feature

Table 1: Illustrative Mobile Phone Attributes

Diagonal Screen Size	Price	Operating System
Less than 3 Inch	5000 Rupees or less	Android
3.0 - 3.4 Inches	5001 - 10000 Rupees	Symbian
3.5 - 3.9 Inches	10001 - 18000 Rupees	Windows
4.0 - 4.4 Inches	18001 - 35000 Rupees	Blackberry
4.5 - 4.9 Inches	35001 Rupees and Above	iOS (iPhone OS)
5 Inches and over		

The chief proponent of this methodology, Marder (1997), argues that the resultant ratings are superior to bounded ratings because they lead to normal distributions. Ratings scales, for example, that limit choices from one to ten often produce 'cliff distributions' where the values cluster at the minimum or maximum value. Respondents were required to evaluate 18 attributes of a mobile phone. Table 1shows three of the 18 attributes.

The attributes did not all have the same number of levels. As can be seen from Table 1the screen size attribute had six levels, while price had only five levels. To complete the Web based survey respondents were required to enter ratings for every level of every attribute. The full list of attributes and levels is shown in Appendix A.

After giving their ratings to the separate product features, respondents saw a product-ranking screen. The lower left part of the screen contained a list of eight mobile phones. The survey software randomized the list in a different way for each respondent. The lower right hand side of the screen showed an empty list which respondents were required to fill. They had to rank the eight products from 'least likely to buy' to 'most likely to buy'. The software prevented respondents from proceeding to the final screens until they had completed the ranking task. After completing the survey, respondents were sent electronic money in the form of a 500 electronic gift certificate which could be spent on the Web site Flipkart.com (India's closest equivalent to Amazon.com).

Results from Utility Summation

The features of the eight mobile phones researched to find the configuration of features and prices for each of the eight products. Appendix B shows the sources for the product feature information.

Table 2: Market Shares from Direct Choice Task and Utility Summation

iPhone 5 with 32 GB	Samsung Galaxy Note 2	Black-berry Curve 9220	XOLO Q1000	Spice Mi-495	Micro-max Canvas 4 A210	Nokia Lumia 520	Lava Iris 504Q	Prediction Method
22.4%	50.2%	2.0%	6.0%	3.0%	6.5%	9.5%	0.5%	Summed Utilities
41.3%	21.4%	6.0%	4.0%	2.5%	6.5%	12.9%	5.5%	Ranking Task
18.9%	-28.9%	4.0%	-2.0%	-0.5%	0.0%	3.5%	5.0%	Difference
Mean Absolution Deviation: 7.8% Hit Rate: 22.9% N: 201								

Table 2 shows implied market shares based on the two types of data collected. The first row shows the number of products that achieved the highest score based on summing the utilities for the 18 attributes of the various products. Formula 1 below describes the scoring process that determines the market shares in row 1 of Table 2.

$$PV_{pi} = \sum_{a=1}^{amax} U_{ai} * F_{ap} \quad (1)$$

In formula 1 above, U_{ai} is the utility value of the *ith* respondent for the *ath* attribute level. F_{ap} represents a matrix describing the configuration of a specific product. It contains Boolean (0,1) values that indicate which level of a given attribute a product has. To take the example of our mobile phone survey, if the largest screen size takes the value '1' and the iPhone has this screen size then it will take the value '1' for this level and all other levels of the screen size attribute will be zero. F_{ap} is the presence of feature *F* for

the attribute-level *a* for product *p*. The total product value, $PV_{pi,}$ is the sum of a product's utilities for the *ith* respondent for product *p* (given the feature configuration F_p).

Row 2 of Table 2 shows the proportion of respondents who put each of the products at the top of the list in the 'probability to purchase' ranking task. The mean absolute deviation between the estimated and direct choice markets shares was 7.8%. The summed utility method predicted the top ranked products for 22.8% of the respondents (a proportion commonly called the 'hit rate' in market research literature).

Fitness Measures and Classification Problems

For non-logit regression models, predicted values are continuous variables. The evolutionary search process results in sets of such variables that minimize the error between their predicted values and those of the dependent variable. For classification searches, the predicted values are categorical variables. The predicted values for our product search process are discrete categorical values between one and eight that represent one of the eight products.

The classification search required a suitable database of training data. The 18 utility scores were the independent variables. Each of the 201 respondents had eight sets of utility scores to represent the eight products in the ranking task. For each person the eight rows of product data were based on:

a) the utility score the respondent gave to each of the 18 attributes
b) the specific utility value that was relevant to each product's feature configuration

The evolutionary search was conducted using Abstract Regression Classification (ARC) software (Korns, 2007, 2011a, 2011b). The dependent variable was the rank order number of the eight products where the number zero represented the product the respondents were least likely to buy and seven was the product they were most likely to buy.

The data in Table 2 appears to be an ideal candidate for a classification search. The independent variables are 18 product feature utilities for each of the eight products. The dependent variables are eight discrete values that represent one of the eight products. This requires a fitness measure to replace the Normalized Least Squares Error (NLSE) commonly used in regression models where the dependent variable is a quantitative variable. For this reason, the current classification search used ARC's Classification Error Percent (*cep*) fitness measure. This minimizes the percent of observations where the prediction was *not* an exact match.

The Select() Command

Since the classification search requires a prediction in the form of discrete values, several of the goal specifications below needed a command to constrain predicted values to be in this form. For this reason a *select()* command was used to transform continuous results into one of eight values representing one of the eight products. The following example illustrates the use of the select command. The neuralnet command (described below) can be specified to produce a certain number of outputs. In the illustrative command below the final numeric parameter (the number 8) specifies that the neuralnet goal will have eight outputs.

neuralnet(0,18,4,8,n)

In order to constrain this goal to produce a discrete value between one and eight it was embedded within a select command as follows:

select(neuralnet(0,18,4,8,n))

The select() command will analyze the vector of eight output values and return the position of the highest value. The resulting value was an integer from one to eight.

Training and Testing Data

Initial training runs used at data records for all respondents, however this meant using the records for those products that received a lower (2nd and below) choice rankings. This process resulted in fitness scores close to level that would be produced by chance. Since classification models could expect to produce random hits 12.5% of the time, cep error levels close to 87.5% were similar to the results of chance.

An alternative search strategy involved using only the data records for the top ranked products during the training stage, but then applying the resulting model to the full data set during testing. This procedure was followed in the searches described below.

A Decision Tree Search

A form of decision tree searching is described in Breiman et al. (1984) where the predicted outcome variable is a category. This form of their search process has been termed a classification tree (as distinction from a regression tree described below). A tree search can be specified in ARC using the 'tree' code-expression generator in the following form:

tree(categories, node-depth, tree-depth, c | v | f)

The final parameter takes the following values:

a) 'c' signifies that there is a constant at the decision node
b) 'v' signifies that there is an abstract variable at the decision node
c) 'f' signifies that there is a function at the decision node

In the case of our cell phone search task the goal was specified as follows:

model(tree(8,2,3,f))

Thus, eight categories were specified. The node-depth was two. The tree depth was three and functions were at the decision node.

Table 3: Market Shares from Direct Choice Task and a Decision Tree Search

iPhone 5 with 32 GB	Samsung Galaxy Note 2	Black-berry Curve 9220	XOLO Q1000	Spice Mi-495	Micro-max Canvas 4 A210	Nokia Lumia 520	Lava Iris 504Q	Prediction Method
0.0%	0.0%	0.0%	0.0%	30.5%	23.8%	0.0%	45.7%	Decision Tree
41.3%	21.4%	6.0%	4.0%	2.5%	6.5%	12.9%	5.5%	Ranking Task
-41.3%	-21.4%	-6.0%	-4.0%	28.0%	17.4%	-12.9%	40.2%	Difference
Mean Absolute Deviation: 21.4%								
Hit Rate: 2.2%								
N:201								

After running for three hours and evaluating 142,000 formulas, the champion formula produced the data in Table 3. The fitness percent error on the training data was 50%. In terms of the metrics used by the market research industry, the product hit rate worsened from 22.8% under summed utilities to only 2.2%. The Mean Absolute Deviation between actual and estimated choice shares also deteriorated from 7.8% to 21.4%.

A Non-Linear Discriminant Analysis (NLDA) Search

The next evolutionary search involved Linear Discriminant Analysis (LDA) as proposed by Fisher (1936). Since this search was conducted at a node-depth of two (see below) this was technically Non-Linear Discriminant Analysis (NLDA). The 'net' code-expression generator for LDS searches takes the following form:

net(node-depth, inputs, outputs, x | v, n | h | s)

The penultimate parameter was introduced to hand extremely large numbers of input variables. Its two values have the following meanings:

- 'x' signifies concrete features (when there are fewer than 250 independent variables)
- 'v' signifies abstract variables (when there are more than 250 independent variables)

The final parameter allows the user to constrain the output to be in one of three forms:

- 'n' signifies 'no operator' (results unconstrained)
- 'h' signifies hyperbolic tangent (results in the range -1 to +1)
- 's' signifies sigmoid (results in the range 0 to 1)

The specific goal for the product search was

select(net(2,18,8,x,n))

Two represented the node-depth. The 18 utility scores were the inputs, and the eight outputs corresponded to the eight product choices. The 'x' parameter implies concrete features rather than abstract variables. The final parameter indicates the output values were unconstrained but since the select command was wrapped around the goal specification, the outputs were constrained to be in the range of 1

to 8. After 24,000 well-formed formulas, the NLDA search produced a champion on the testing data with a cep error of 49%.

Table 4: Market Shares from Direct Choice Task and Non-Linear Discriminant Analysis

iPhone 5 with 32 GB	Samsung Galaxy Note 2	Black-berry Curve 9220	XOLO Q1000	Spice Mi-495	Micro-max Canvas 4 A210	Nokia Lumia 520	Lava Iris 504Q	Prediction Method
7.8%	1.2%	32.1%	0.0%	0.0%	16.9%	0.0%	41.9%	LDA
41.3%	21.4%	6.0%	4.0%	2.5%	6.5%	12.9%	5.5%	Ranking Task
-33.5%	-20.1%	26.1%	-4.0%	-2.5%	10.4%	-12.9%	36.4%	Difference
Mean Absolute Deviation: 18.3%								
Hit Rate: 8.0%								
N: 201								

A Weighted Search

The weighted() command differs from the net() command above in that it does not guarantee coverage of all the features. The net() command guarantees coverage due to the deterministic nature of the. Due to this determinism, all the independent variables must be included in every evolution of the formula.

The general form of the weighted code-expression generator is:

Weighted (node-depth, base-functions, n | h | s)

The final parameter has the same meanings as described above under the LDA search section. The specific form of the weighted search used for the mobile phone search was:

model(select(weighted(5,8,s)))

This implied a node-depth of five, eight base functions and outputs constrained to be in sigmoid form.

Table 5: Market Shares from Direct Choice Task and Weighted Search

iPhone 5 with 32 GB	Samsung Galaxy Note 2	Black-berry Curve 9220	XOLO Q1000	Spice Mi-495	Micro-max Canvas 4 A210	Nokia Lumia 520	Lava Iris 504Q	Prediction Method
11.4%	10.7%	1.9%	0.0%	0.0%	5.2%	24.4%	46.4%	Weighted
41.3%	21.4%	6.0%	4.0%	2.5%	6.5%	12.9%	5.5%	Ranking Task
-29.9%	-10.7%	-4.1%	-4.0%	-2.5%	-1.3%	11.5%	40.9%	Difference
Mean Absolute Deviation: 13.1%								
Hit Rate: 14.%								
N: 201								

After evaluating 23,000 formulas this search produced the champion associated with Table 5. Compared to the NLDA search the hit rate and the difference between actual and estimated choice shares improved.

However, both were still worse than the simple process of summing utilities shown in Table 2. The champion model had a cep error of 48%.

An Artificial Neural Network (ANN) Search

Pitts and McCulloch (1943) proposed that neural events and relationships could be represented by propositional logic. Since then various algorithms have been proposed to mimic neural activity that fall into the class of Artificial Neural Networks (ANN).

ARC's neural net code-generator can create a classification goal as follows:

neuralnet(node-depth, inputs, hidden, outputs, x | v, n | h | s)

The penultimate parameter (x | v) has the same meaning as in the case of the LDA search above. The final parameter values (n | h | s) have the same meanings as they do in LDA goal specification. The specific form of the goal for the product search was:

select(neuralnet(0,18,4,8,n))

This indicates a node-depth of zero. The 18 inputs were the 18 product feature utility variables. There were four hidden layers. Eight output values represented the eight product choices. The select command wrapped around this goal constrained these outputs to be integers from one to eight.

After evaluating 11,000 formulas this goal produced the champion with the results shown in Table 6. This champion had a cep error of 44%.

Table 6: Market Shares from Direct Choice Task and Neural Net Search

iPhone 5 with 32 GB	Samsung Galaxy Note 2	Black-berry Curve 9220	XOLO Q1000	Spice Mi-495	Micro-max Canvas 4 A210	Nokia Lumia 520	Lava Iris 504Q	Prediction Method
17.3%	13.0%	0.0%	0.0%	6.8%	3.1%	6.2%	53.5%	Neural Net
41.3%	21.4%	6.0%	4.0%	2.5%	6.5%	12.9%	5.5%	Ranking Task
-24.0%	-8.4%	-6.0%	-4.0%	4.4%	-3.4%	-6.7%	48.1%	Difference
Mean Absolute Deviation: 13.1% Hit Rate: 15.6% N: 201								

A Classification and Regression Tree (CART) Search

ARC allows for a search based on the Classification and Regression Tree technique described by Breiman et al. (1984). The general form of the goal specification is:

cart(node-depth, tree-depth, c | v | f)

The final parameter takes the following values:

a) 'c' signifies that there is a constant at the decision node
b) 'v' signifies that there is an abstract variable at the decision node

c) 'f' signifies that there is a function at the decision node

The specific goal for our product classification search was:

model(cart(2,3,c))

This the goal specified a node-depth of 2 at the leaf level, a tree-depth of 3 and constants at the decision node. Since the select command was not used, the results are not constrained to be continuous values. This goal produced a regression model with a continuous variable as its predicted values.

The CART search evaluated 988,000 formulas. Its training score is not comparable to the fitness percentages above because the output from CART is a regression formula rather than a category. Its error of 97% appears to be larger than the fitness percent errors quoted above but it produced the best metrics in terms of hit rate and mean absolute deviation between actual and estimated choice shares.

It is interesting to note that the hit rate is an improvement on that based on summed utilities in Table 2 but the Mean Absolute Deviation is worse.

Table 7: Market Shares from Direct Choice Task and CART Search

iPhone 5 with 32 GB	Samsung Galaxy Note 2	Black-berry Curve 9220	XOLO Q1000	Spice Mi-495	Micro-max Canvas 4 A210	Nokia Lumia 520	Lava Iris 504Q	Prediction Method
91.5%	0.0%	7.5%	1.0%	0.0%	0.0%	0.0%	0.0%	CART
41.3%	21.4%	6.0%	4.0%	2.5%	6.5%	12.9%	5.5%	Ranking Task
50.2%	-21.4%	1.5%	-3.0%	-2.5%	-6.5%	-12.9%	-5.5%	Difference
Mean Absolute Deviation: 12.9%								
Hit Rate: 37.8%								
N: 201								

An NLSE Search

The favorable hit rate from the CART search suggested the possibility of using a regression model search rather than classification. Normalized Least Squares Error (NLSE) was chosen as the fitness measure. For any given respondent the full data set was used in training (the winning products and the lower ranked products) because all rankings were considered to have information value in the NLSE search process.

ARC's universal code-expression generator has the following general format:

universal(node-depth, base-functions, v | t)

The first parameter specifies the grammar depth of the expression allowed. The second parameter specifies the number of base functions. The final parameter has the following meanings:

- 'v' means only variables may compose the base functions
- 't' means variables or constants may compose the base functions

The specific goal for the mobile phone search was defined as follows:

regress(universal(1,14,v))

This specified a grammar depth of one, 14 basis functions, and only variables within them. After various combinations of operators and evolution durations no champion model improved on the summed utility approach in Table 2.

It was not clear how best to balance the importance of the winning product and the lower ranked products. Since market shares depend only on a person's top-ranked product, it was attractive to privilege them in the search process. However, the lower ranked products represented seven eighths of all the available data. A hybrid approach was selected. The dependent variable was squared. Since the top ranked product had the highest value this meant its ranking had more importance but the data from the lower ranked products was retained. This eventually produced a champion that improved on both the hit rate and the mean absolute deviation between the actual and estimated choice shares. Table 8 shows the results of this champion.

Table 8 Market Shares from Direct Choice Task and CART Search

iPhone 5 with 32 GB	Samsung Galaxy Note 2	Black-berry Curve 9220	XOLO Q1000	Spice Mi-495	Micro-max Canvas 4 A210	Nokia Lumia 520	Lava Iris 504Q	Prediction Method
62.7%	11.4%	9.0%	2.0%	1.5%	1.5%	11.4%	0.5%	Summed Utilities
41.3%	21.4%	6.0%	4.0%	2.5%	6.5%	12.9%	5.5%	Ranking Task
21.4%	-10.0%	3.0%	-2.0%	-1.0%	-5.0%	-1.5%	-5.0%	Difference
Mean Absolute Deviation: 6.1% Hit Rate: 35.8% N: 201								

The Table 8 results show that the hit rate increased from 22.8% to 35.8% and the mean absolute deviation fell from 7.8% to 6.1%. The Bowker-McNemer test is a variation on the Chi-square test where the same respondents are measured twice. This test indicated that the NLSE champion formula not only produced improved results but that they were different from the summed utility results ($p. < 0.01$).

The special circumstances of product choice modeling imply that the ideal search evolutionary search process would involve an customized fitness measure which progressively decreases the mean absolute deviation between actual and estimated choice shares.

Summary

Given that the cell phone data involved eight discrete choices, it was logical to assume that a predictive model could follow a classification approach. It is interesting to note, that the same training data format could be used for a wide variety of different classification search strategies. CART, decision tree learning, neural nets, non-linear discriminant analysis and it non-deterministic variant 'weighted()'. Code-generators allowed these searches to be undertaken with minimal effort to specify each search goal.

Ultimately, the product prediction process was improved by a regression approach using least squares rather than classification error as the fitness measure. Even though classification proved not to win the hunt, the ease with which hunters can use different sets of dogs must be counted as one of ARC's strengths. This should be counted as an argument for the standardization of regression and classification search languages so that this flexibility becomes commonplace.

Bibliography

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees.* Pacific Grove, CA: Wadsworth & Brooks.

Fisher, R. A. (1936). The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, *7*, 179–188.

Green, P. E., & Rao, V. R. (1971). Conjoint Measurement for Quantifying Judgmental Data. *Journal of Marketing Research*, *8*(3), 355–363. http://doi.org/10.2307/3149575

Korns, M. F. (2007). Large-Scale, Time-Constrained Symbolic Regression. In R. Riolo, T. Soule, & B. Worzel (Eds.), *Genetic Programming Theory and Practice IV* (pp. 299–314). Springer US. Retrieved from http://link.springer.com/chapter/10.1007/978-0-387-49650-4_18

Korns, M. F. (2011a). Abstract Expression Grammar Symbolic Regression. In R. Riolo, T. McConaghy, & E. Vladislavleva (Eds.), *Genetic Programming Theory and Practice VIII* (pp. 109–128). Springer New York. Retrieved from http://link.springer.com/chapter/10.1007/978-1-4419-7747-2_7

Korns, M. F. (2011b). Accuracy in Symbolic Regression. In R. Riolo, E. Vladislavleva, & J. H. Moore (Eds.), *Genetic Programming Theory and Practice IX* (pp. 129–151). Springer New York. Retrieved from http://link.springer.com/chapter/10.1007/978-1-4614-1770-5_8

Marder, E. (1997). *The laws of choice : predicting customer behavior*. New York: Free Press.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*(4), 115–133. http://doi.org/10.1007/BF02478259

Srinivasan, V., & Park, C. S. (1997). Surprising Robustness of the Self-Explicated Approach to Customer Preference Structure Measurement. *Journal of Marketing Research*, *34*(2), 286–291. http://doi.org/10.2307/3151865

Appendix A- Questionnaire Text

1	Operating system	Android
2		Symbian
3		Windows
4		Blackberry
5		iOS (iPhone OS)
6		Less than 3 Inch
7	Screen Size	3.0 - 3.4 Inches
8		3.5 - 3.9 Inches
9		4.0 - 4.4 Inches
10		4.5 - 4.9 Inches
11		5 Inches and over
12	Camera Memory	Below 2 Megapixels
13		2-4.9 Megapixels
14		5-7.9 Megapixels
15		8 Megapixels and Above
16		Below 8 GB
17	Memory	8 - 15.9 GB
18		16 - 31.9 GB
19		32 - 63.9 GB
20		64 GB or more
21	Talk Time	Less than 6 hours
22		6-11 hours
23		12-23 hours
24		24-35 hours
25		36 hours or more
26	Stand by Time	Under 50 hours
27		50-99 hours
28		100-199 hours
29		200-299 hours
30		300 hours or more
31	Price	5000 Rs or less
32		5001 - 10000 Rs
33		10001 - 18000 Rs
34		18001 - 35000 Rs
35		35001 Rs and Above
36	Phone Thickness	Less than 6 mm
37		6-7 mm
38		8-9 mm
39		10-11 mm

40		12 mm or more
41	CPU Speed	1 GHz or less
42		1.0 to 1.3 GHz
43		1.4 to 1.5 GHz
44		1.6 to 1.9 GHz
45		2.0 GHz or more
46	Warranty Length	Free Repairs for 6 months
47		Free Repairs for 1 year
48		Free Repairs for 1.5 years
49		Free Repairs for 2 years
50		Free Repairs for 2.5 years
51	GPS	Has GPS
52		No GPS
53	Wi-Fi	Has Wi-Fi
54		No Wi-Fi
55	Touchscreen	Has a touchscreen
56		No touchscreen
57	USB Connection	Has a USB Connection
58		No USB connection
59	SIM format	Single SIM
60		Dual SIM
61	3G	Has 3G Connectivity
62		No 3G Connectivity
63	Qwerty Keyboard	Has a QWERTY Keyboard
64		No QWERTY Keyboard
65	Brand Impression	Apple
66		Samsung
67		Blackberry
68		XOLO
69		Spice
70		Micromax
71		Nokia
72		Lava

Appendix B – Sources of Feature Data

All feature data for the eight mobile phones were drawn from www.Flipkart.com on September 26th, 2013 except the following items that were missing from the Flipkart comparison screens.

For the iPhone 5 with 32 GB, data was missing for the CPU speed attribute. This was taken from www.GSMArena.com on September 26th 2013.

For the Samsung Galaxy Note 2, data was missing for the talk-time and standby time attributes. This was taken from www.GSMArena.com on September 26th 2013.

For the Blackberry Curve 9220, data was missing for the GPS attribute. This was taken from www.GSMArena.com on September 26th 2013. The CPU speed attribute was missing from both these sources. It was taken from asia.cnet.com[1] on September 26th 2013.

For the XOLO Q1000, data was missing for the GPS attribute. This was taken from www.GSMArena.com on September 26th 2013.

For the Spice MI-495, data was missing for the USB connection attribute. This was taken from www.GSMArena.com on September 26th 2013. The phone thickness attribute was missing from both these sources. It was taken from comapareindia.in.com[2] on November 5th 2013.

For the Micromax Canvas 4 A210, data was missing for the GPS attribute. This was taken from www.GSMArena.com on September 26th 2013.

For the Lava Iris 504Q, data was missing for the GPS attribute. It was taken from comapareindia.in.com[3] on November 5th 2013.

[1] http://asia.cnet.com/product/blackberry-curve-9220-46114978.htm
[2] http://compareindia.in.com/specification/mobile-phones/spice-stellar-virtuoso-mi495/303332
[3] http://compareindia.in.com/specification/mobile-phones/lava-iris-504q/325242