Visualizable and explicable recommendations obtained from price estimation functions

Claudia Becerra and
Fabio Gonzalez
Intelligent Systems Research Laboratory (LISI)
Universidad Nacional de Colombia, Bogota
[cjbecerrac,fagonzalezo]@unal.edu.co

Alexander Gelbukh
Center for Computing Research (CIC)
National Polytechnic Institute (IPN)
Mexico DF, 07738, Mexico
gelbukh@gelbukh.com

ABSTRACT

Collaborative filtering is one of the most common approaches in many current recommender systems. However, historical data and customer profiles, necessary for this approach, are not always available. Similarly, new products are constantly launched to the market lacking historical information. We propose a new method to deal with these "cold start" scenarios, designing price-estimation functions used for making recommendations based on cost-benefit analysis. Experimental results, using a data set of 836 laptop descriptions, showed that such price-estimation functions can be learned from data. Besides, they can also be used to formulate interpretable recommendations that explain to users how product features determine its price. Finally a 2D visualization of the proposed recommender system was provided.

Categories and Subject Descriptors

H.1.2 [**User/Machine Systems**]: Human information processing; H.4.2 [**Types of Systems**]: Decision support

General Terms

Experimentation

Keywords

Apriori recommendation, Cold-start recommendation, Price estimation functions

1. INTRODUCTION

The internet and e-commerce grow exponentially. As a result, decision-making process about products and services is becoming increasingly complex. These processes involve hundreds and even thousands of choices and a growing number of heterogeneous features for each product. This is mainly due to the introduction and constant evolution of new markets, technologies and products.

Unfortunately, human capacity for decision-making is too limited to address the complexity of this scenario. Studies in psychology field have shown that human cognitive capacities are limited from five to nine alternatives for simultaneous comparison [17, 14]. Consequently, making a purchasing decision at an e-commerce store that does not provide tools to assist decision-making, is a task that largely overwhelms human capacities. Moreover, several studies have shown that this problem generates adverse effects on people such as: regret due to the selected option, dissatisfaction due to poor justification for the decision, uncertainty about the idea of "best option", and overload of time, attention and memory (see [20]).

Many recommender systems approaches have addressed this problem through collaborative filtering [6] based on product content (i.e. descriptions) and on customer information [2, 9]. This approach recommends products similar to those chosen by similar users. On the other hand, latent semantics approaches [10] have been successfully used to build affinity measures between products and users. Most of the aforementioned approaches have been applied in domains with products such as books and movies that remain available long enough to collect enough historical data to build a model [4, 12].

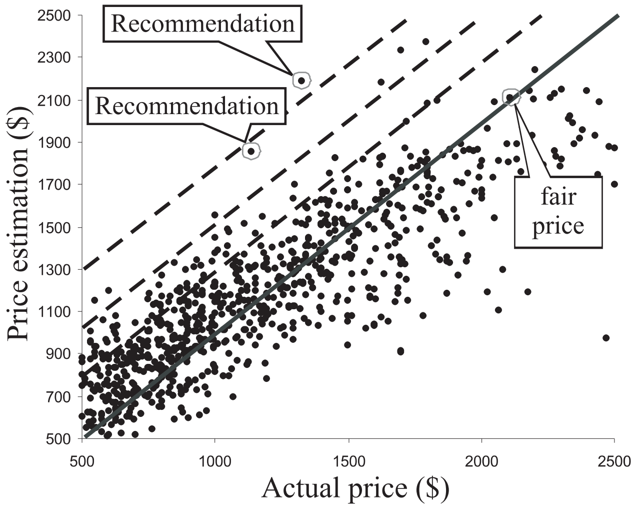
While impressive progresses have been made in the field using collaborative filtering, the relevance of current approaches in domains with frequent changes in products is still an open question [8]. For example, customer-electronics domain is characterized by products with a very short life cycle in the market and a constant renewal of technologies and paradigms. Collaborative approaches face two major problems in this scenario [13]. First, product features are constantly redefined, making difficult for users to identify relevant attributes. Second, historical sales product data become obsolete very quickly due to the frequent product substitution. This problem of making automatic recommendations without historical data is known as *cold-start recommendation* [18].

In this paper, we propose a new cold-start method based on an estimate of the benefit to the user when purchasing a product. This function is formulated as the difference between estimated and real prices. Therefore, our approach recommends products with high benefit-cost ratio to find "best-deals" on a data set of products. Figure 1 shows an example of such recommendations based on utility functions displaying 900 laptop computers. In this figure, the features of laptops below the line in bold, indicating fair prices, do not justify prices of laptops.

The rest of the paper is organized as follows. In Section 2, the necesary background and proposed method are presented. In Section 3, an evaluation methodology and some data refinements are proposed and applied to the model. Finally, in Section 4, some concluding remarks are briefly discussed.

RecSys'11, October 23–27, 2011, Chicago, Illinois, USA.

Figure 1: Graphic recommender based on price estimates



2. APRIORI RECOMMENDATIONS USING UTILITY FUNCTIONS

The general intuition of method is led by the lexicographical criterion [22]. That is, users prefer products that offer more value for their money. Clearly, this approach is not applicable to all circumstances, but it is general enough when customer profiles are not available in cold-start scenarios.

When a user purchases a product x_i, a utility function $utility(x_i)$ provides an estimation of the difference between the estimated price $f(x_i)$ and the market price y_i, that is $utility(x_i) = f(x_i) - y_i$. Thus, the products in the market are represented as a set $X = \{x_1, x_2, ..., x_n, ..., x_N\}$, where each product x_i is a vector characterized in a feature space $\mathbb{R}^{\mathrm{M}}$. With these data, a regression model, learned from X and the vector of prices y, generates price estimations $f(x_i)$ required for calculation of the utility. Finally, the utility function is computed on all products thus providing an ordered list with the top-n apriori recommendations.

Estimates of price $f(x_i)$ can be obtained by a linear-regression model as:

$$f(x_i) = \beta_o + \sum_{m \in \{1,...,M\}} \beta_m x_{im}. \tag{1}$$

This model is equivalent to an additive value function used in the decision-making model *SAW* (simple additive weighting) [5], but with coefficients β_m learned automatically. Clearly, the recommendations obtained from these estimates can be explained to users, since each term $\beta_m x_{im}$ represents the money contribution to the final price estimate provided by the m-th feature of the i-th product.

The quality of the apriori recommendations obtained with the proposed method depends primarily on three factors: the amount of training data, the accuracy of price estimates $f(x_i)$, and the ability to extract user-understandable explanations from the regression model. Certainly, linear models, such as that of eq. 1, offer good interpretability, but in many cases, these models generate high rates of error in their predictions when the interactions among features are complex. These models are known as *weak regression models*. On the other hand, discriminative approaches, such as *support vector regression* [19], provide better models with lower error rates but also with lower interpretability.

This trade-off can be overcome with a hybrid regression model as *3FML* (three-function meta-learner) [3]. This meta-regressor combines two different regression methods in a new improved combined model in a way similar to other meta-algorithms such as *voting*, *bagging* and *AdaBoost* (see [1]). Unlike these methods, 3FML uses one regression method to make price predictions and another to predict the error. As long as the former regression method is weak, stable and interpretable, the latter can be any other regression method regardless its interpretability. As a result, the combined regression preserves the same interpretability level of the first regressor but with lower error rate.

A linear regression model can be trained to learn parameters β_m by minimizing the least squared error from data [15]. This first model can be used by 3FML to build a base regression model $f_0(x)$ with the full dataset. Then, this model is used to divide the data into two additional groups depending on whether the obtained price predictions were below or above the training price, given a difference threshold θ. Next, using the same base-regression method, two additional models $f_{+1}(x)$ and $f_{-1}(x)$ are trained with the pair of subsets called respectively, *upper model* and *lower model*. Figure 3 illustrates *upper*, *base* and *lower* models compared to the target function, which is the price in a data set of laptop computers. The three resulting models are combined using an aggregation mechanism – called *mixture of experts* [11] – with the following expression:

$$\hat{f}(x_i) = \sum_{l \epsilon \mathbb{H}} w_l(x_i) f_l(x_i), \tag{2}$$

having

$$\sum_{l \epsilon \mathbb{H}} w_l(x_i) = 1, i \in \{1 \ldots n\} \tag{3}$$

$\mathbb{H}$ is a dictionary of experts consisting on the base model and two additional specialized models, $\mathbb{H} = \{f_{-1}(x), f_0(x), f_{+1}(x)\}$. The gating coefficient w_{li} establishes the level of relevance of the l model into the final price prediction for the i-th product.

In 3FML model, coefficients $w_{li} = w_l(x_i)$ are obtained by chaining a membership function w_l for each regression model to a function α that depends on the errors of the three models, $w_l(x_i) = w_l(\alpha(f_{-1}(x_i), f_0(x_i), f_{+1}(x_i), y_i))$. These membership functions $w_l(\alpha)$ are similar to those used in fuzzy sets [23] but these satisfy the constraint given by eq. 3. Three examples of those functions are shown in Figure 2; one triangular and two Gaussian. Clearly, the range of the error function α must agree with the domain of the membership functions. For instance, if the domain of the membership functions is $[0, 1]$, an appropriate function α_i must return a value close to 0.5 when y_i is better modeled by $f_0(x_i)$. Similarly, reasonable values for α_i, if y_i is better modeled by $f_{-1}(x_i)$ or $f_{+1}(x_i)$, are respectively 0.0 and 1.0.

Such function α can be arithmetically constructed (see [3] for triangular and Gaussian cases) and α_i can be obtained for every x_i. 3FML makes use of a second regression method to learn a function for α_i. This function is called α-learner, which seeks to predict the same target y_i but indirectly through the errors obtained by f_{-1}, f_0 and f_{+1}. The estimates obtained with α-learner are used in com-

Figure 2: Triangular and Gaussian membership functions

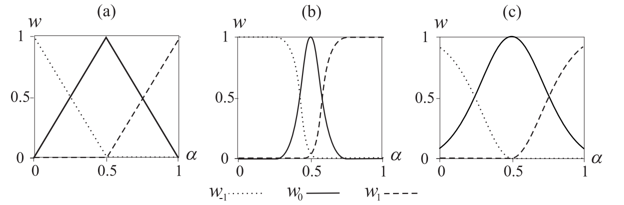
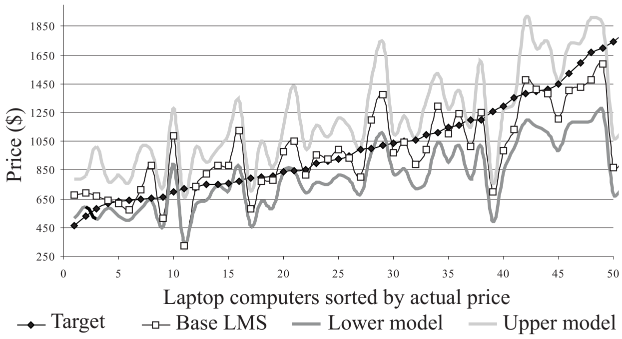


Figure 3: 3FML's three regression models graph



bination with the membership functions to get coefficients $w_l(x_i)$. Therefore, final predictions are obtained with a different linear model for each target price y_i. The resulting model is also linear, but different for each product instance in function to x_i:

$$\hat{f}(x_i) = \hat{\beta}_0(x_i) + \sum_{m \epsilon \{1, ..., M\}} \hat{\beta}_m(x_i) x_{im}, \quad (4)$$

where

$$\hat{\beta}_o(x_i) = \sum_{l \in \mathbb{H}} \beta_{lo} w_l(x_i) \; ; \; \hat{\beta}_m(x_i) = \sum_{l \in \mathbb{H}} \beta_{lm} w_l(x_i).$$

Clearly, the model in eq. 4 is as user-explainable as that of eq. 1.

The effect of α-learner in eq. 4 is that the entire data set is clustered into three latent classes. These classes can be considered as market segments namely: high-end, mid-range and low-end products. Many commercial markets exhibit this segmentation, e.g. computers, mobile phones, cars, etc.

3. EXPERIMENTAL VALIDATION

The aim of experiments is to build a model that provides a cost-benefit ranking of a set of products where each product is represented as a vector of features. To assess the quality of this ranking, two factors are observed. First, the error of the price-estimation regression should be low to make sure that this function provides a reasonable explanation of the data set. Second, the model must be interpretable and discovered knowledge must be consistent with market data. For example, if a proposed model discovers a ranking of how much money each operating system contributes to laptop prices, this ranking should be in agreement the prices of retail versions of the same operating systems.

In addition, the full features set of the top-10 recommended products is provided along with a 2D visualization of the entire data set. These resources allow the reader – guided by a brief discussion – to qualitatively evaluate the recommendations obtained with the proposed method.

Table 1: Attributes in *Laptops_17_836* data set

Feature name	Type	% missing
Manufacturer	Nominal	0.00%
Processor Speed	Numeric	0.40%
Installed Memory	Numeric	1.90%
Operating System	Nominal	0.00%
Processor	Nominal	0.20%
Memory Technology	Nominal	7.20%
Max Horizontal Resolution	Numeric	7.90%
Warranty-Days	Numeric	15.50%
Infrared	Nominal	0.00%
Bluetooth	Nominal	0.00%
Docking Station	Nominal	0.00%
Port Replicator	Nominal	0.00%
Fingerprint	Nominal	0.00%
Subwoofer	Nominal	0.00%
External Battery	Nominal	0.00%
CDMA	Nominal	0.00%
Price	Numeric	0.00%

3.1 Data

The data is a set of 836 laptop computers each represented by a vector of 69 attributes including price, which is the attribute to be estimated. Data were collected by Becerra[1] from several U.S. e-commerce sites (e.g. Pricegrabber, Cnet, Yahoo, etc.), during the second half of 2007 within a month. A subset of 17 features was selected using the correlation-based selection method proposed by Hall [7]. We call this dataset *Laptops_17_836*; all its features and percentage of missing values are shown in Table 1.

3.2 Price estimation results

For the construction of the price-estimation function, several regression methods were used, namely: least mean squares linear regression (LMS) [15], M5P regression tree [21, 16], support vector regression (SVR) [19] and three-function meta-learner (3FML, described in previous section). 3FML provides three interpretable linear models: *upper*, *base* and *lower* models, which can be associated with product classes. Finally, estimated price for each laptop was obtained with the combination of these three models using eq. 4 with the weights obtained from α-Learner and Gaussian membership functions.

The performance of each method was measured using root-mean-square error (RMSE) defined as:

$$RMSE = \sqrt{\frac{\sum_i \left(\hat{f}(x_i) - y_i\right)^2}{|X|}}.$$

The data set was randomly divided into 75% for training and 25% for testing. Ten different runs of this partition ratio were used for each method. These ten RMSE results were averaged and reported. Table 2 shows the results, their standard deviation (in parentheses) and some model parameters.

[1] http://unal.academia.edu/claudiabecerra/teaching

The method with lowest RMSE was SVR with a complexity parameter $C = 100$ using radial basis functions (RBF) as kernel. However, interpretability of this model is quite limited, given the embedded feature space induced by the kernel. On the other hand, LMS and 3FML provide straightforward interpretation of β coefficients, which represent the amount of the contribution of each feature to the product estimated price. Clearly, 3FML was the method that better coped with this interpretability-accuracy trade-off.

Table 2: 10 runs average RMSE results for price estimates obtained with several regression methods

Regression model	Avg. RMSE
M5P regression tree	239.70(21.57)
Least Mean Squares (LMS)	259.87(17.90)
ε-SVR, $C = 100$, linear kernel	258.93(16.93)
3FML (LMS as base model)	233.48(14.76)
3FML (ε-SVR, $C = 100$, linear kernel)	**223.76(8.57)**
ε-SVR, $C = 100$, RBF kernel $\sigma = 7.07$	230.23(12.27)

Table 3: Proportions of operating systems ocurrences in *Laptops_17_836* data set

Operating System	#	%
Vista Home Premium™ (WinVHP)	251	30.02%
WinXP Pro™ (WinXPP)	208	24.88%
WinXP™ (WinXP)	151	18.06%
Win. Vista Business™ (WinVB)	137	16.39%
Win. Vista Home Basic™ (WinVHB)	44	5.26%
Mac OS™ (MacOS)	34	4.07%
Win. Vista Ultimate™ (WinVU)	11	1.32%
Total	836	100%

Table 4: Retail prices for different editions of *Windows Vista*™

O.S.→	WinVHB	WinVHP	WinVB	WinVU
Retail price*	\$199.95	\$259.95	\$299.95	\$319.95

*http://www.microsoft.com/windows/windows-vista/compare-editions (site consulted in September 2007)

3.3 Evaluation and feedback

In this section the price estimation function obtained using 3FML is manually analyzed checking coherence of β coefficients with real facts of the market. Particularly, coefficients for attributes *operating system*, *processor* and numerical features are reviewed, and – when necessary – some refinements are proposed to the data sets to deal with discussed issues.

3.3.1 Operating System *attribute analysis*

Table 3 shows the distribution of the different operating systems into the entire data set of laptops and the abbreviations that we use to refer them at Table 5 and Table 4.

In order to evaluate the portion of the price estimation model related to *operating system* (OS) attribute, coefficients of this feature are compared with related Microsoft's retail prices. Table 4 shows public retail prices for *Windows Vista*™ published at 2007-3Q. In spite that at that date, *Windows Vista*™ operating system had already six months of launched, many brand new laptops still had pre-installed previous *Windows XP*™. Thus, we consider for analysis *Windows XP Pro*™ equivalent to *Windows Vista Business*™, as well as, *Windows XP*™ equivalent to *Windows Vista Home Premium*™. This assumption is also coherent with the observed behavior in Microsoft's price policy that keeps prices of previous product releases invariable during version transition periods.

It is interesting to highlight the behavior of 3FML model with *Windows Vista Ultimate*™. Although this OS version occurs only at 1.32% of instances (see Table 3), it is correctly recognized as the most expensive OS (see Table 4) by the upper model. This fact corrects an erroneous tendency recognized by base and lower models. In general terms, for other OS versions, 3FML managed to predict similar ordering as that of retail prices.

3.3.2 Processor *attribute coefficients*

As shown in Table 1, *Laptops_17_836* data set has two features to describe the main processors of laptops , they are: *Processor Speed* (numeric) and *Processor* (nominal). The former is the processor clock rate and the latter is a text string that contains — in most of cases – the manufacturer, the product family and the model (e.g. "Intel Core 2 Duo Mobile T7200"). Unlike OS attribute, which has only seven possible alternatives, *Processor* attribute has 133 possible processor models. Moreover, the frequencies of occurrence of each processor model exhibit a Zipf-type distribution (see Figure 4). Thus, approximately half of the 836 laptops have only 8 different processors and more than 80 processors occur only in one laptop. Part of this sparseness is due to missing information, abbreviations and formatting.

The *Processor* attribute, as found in the data set, can generate a detrimental effect on the price-estimation function. Besides, β coefficients could hardly be explained and their evaluation against market facts could lead to misleading results. Thus, the model was withdrawn from *Processor* attribute and it was renamed as *Proc. Family*. In addition, the data set was enriched manually adding the following four processor related attributes:

- *L2-Cache*: processor cache in Kibibytes (2^{10} bytes).
- *Hyper Transport*: frontal bus clock rate in Mhz.
- *Thermal Design*: maximum dissipated power in watt.
- *Process Technology*: CMOS technology in nanometre.

This new data set is referred as *Laptops_21_836* data set. Performance results of new price-estimation functions are shown in Table 6. Clearly, SVR and 3FML obtained substantial improvements using this new data set.

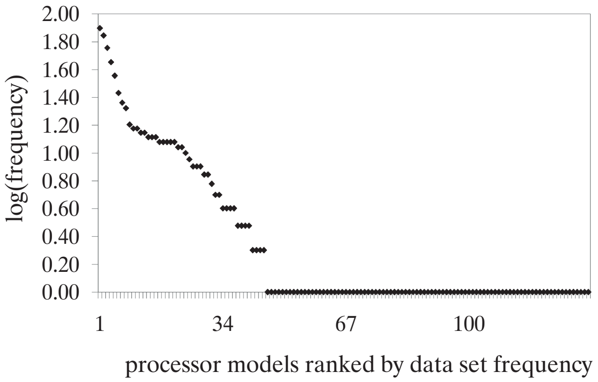
Similarly to the analysis made for OS attribute, processors families also have a consumer-value ranking given by their technology, which can be compared to a ranking taken from an interpretable price-estimation function. The technology ranking of Intel processors is: (1st) *Core 2 Duo*™, *Core Duo*™, *Core Solo*™, *Pentium Dual Core*™ and *Celeron*™. Same for AMD's processors: (1st) *Turion*™, *Athlon*™ and *Sempron*™ [2]. We extracted a ordering for processor fami-

[2]see http://www.notebookcheck.net/Notebook-Processors.129.0.html for a short description of mobile processor families (site consulted in June 2011)

Table 5: 3FML base, upper and lower model coefficients $\beta_{s.o}$ for *operating system* attribute

Base model		Upper model		Lower model	
S.O.	$\beta_{s.o}$	S.O.	$\beta_{s.o}$	S.O.	$\beta_{s.o}$
WinVU	323.3	WinVB	185.6	WinVB	127.3
WinVB	260.3	WinXPP	184.5	WinXPP	127
WinXPP	249.8	MacOS	169.2	MacOS	95.2
MacOS	245.9	WinVU	96.4	WinVHP	24.7
WinVHP	116.7	WinVHP	57	WinVU	0.0
WinXP	94.3	WinXP	27.8	WinVHB	0.0
WinVHB	0.0	WinVHB	0.0	WinXP	-7.1

Figure 4: Distribution of *Processor* attribute



lies by their corresponding β coefficients from 3FML models. Results for this ranking – means and standard deviation – making 10 runs with different samples of 75% training and 25% test are shown in Table 7.

Results in Table 7 show how *upper model* better ordered processor families with high technological ranking. Similarly, *lower model* does a similar work recognizing *Sempron*™ family at the lowest rank.

3.3.3 Numerical attributes coefficients

This subsection present a brief discussion on the interpretation of β coefficients extracted from the price-estimation function for some numeric attributes (shown in Table 8). Although this interpretation is clearly subjective, it reveals some laptop-market facts, which were extracted in an unsupervised way from the data.

For instance, consider *Thermal Design* attribute. Negative values in the β coefficients reveal a fact: the lesser power the CPU dissipates, the higher the laptop's price.

Table 6: RMSE for regression price estimates in *Laptops_21_836* data set

Regression model	RMSE
ε-SVR ($C = 1$, lineal kernel)	254.56(11.75)
ε-SVR ($C = 100$, RBF kernel,)	219.16(9.88)
3FML*	220.91(10.97)

* Base model: ε-SVR, $C = 1$, lineal kernel. α-Learner: ε-SVR, $C = 100$, RBF kernel.

Table 7: Processor families rankings obtained from 3FML price-estimation function

Upper model		Lower model	
Intel Core2 Duo	7.4(0.8)	Intel Core2 Duo	7.6(0.5)
Intel Core Duo	7.2(1.2)	Intel Core Solo	6.2(1.7)
Intel Core Solo	5.3(2.1)	AMD Athlon	5.7(3.8)
Intel Celeron	5.1(1.7)	Intel Core Duo	4.8(2.1)
PowerPC	4.6(2.6)	AMD Turion	4.8(1.8)
Pent DualCore	3.7(1.4)	Pent DualCore	4.3(2.1)
AMD Sempron	3.4(2.4)	PowerPC	4.3(3.1)
AMD Turion	3.3(1.8)	Intel Celeron	3.3(1.3)
AMD Athlon	1.8(1.4)	AMD Sempron	2.8(2.3)

Base model	
Intel Core Solo	8.5(0.7)
Intel Core2 Duo	8.3(0.7)
Intel Core Duo	6.8(0.9)
Pent DualCore	5.1(1.4)
Intel Celeron	5.0(0.7)
AMD Turion	3.7(1.1)
PowerPC	2.9(2.1)
AMD Sempron	2.6(1.8)
AMD Athlon	2.1(1.3)

Table 8: β coefficients for numerical attributes from 3FML model with *Laptops_21_836* data set

Feature name	Upper	Base	Lower
β_0	0.23	0.12	0.06
Warranty Days	0.04	0.01	-0.01
Installed Memory	-0.11	0.17	0.10
Max. Horizontal Resolution	0.12	0.37	0.15
Processor Tech.	0.30	0.08	0.05
Thermal Desing	-0.01	-0.37	-0.27
Hyper Transport	-0.02	0.25	0.05
L2-Cache	0.08	0.16	0.11
Processor Speed	-0.03	0.25	0.17

Besides, these coefficients also shows that this effect affects prices more at mid-range and low-end laptop-market segments. Similarly, *Max. Horizontal Resolution* attribute reveals that this feature has greater impact on the mid-range laptop market prices.

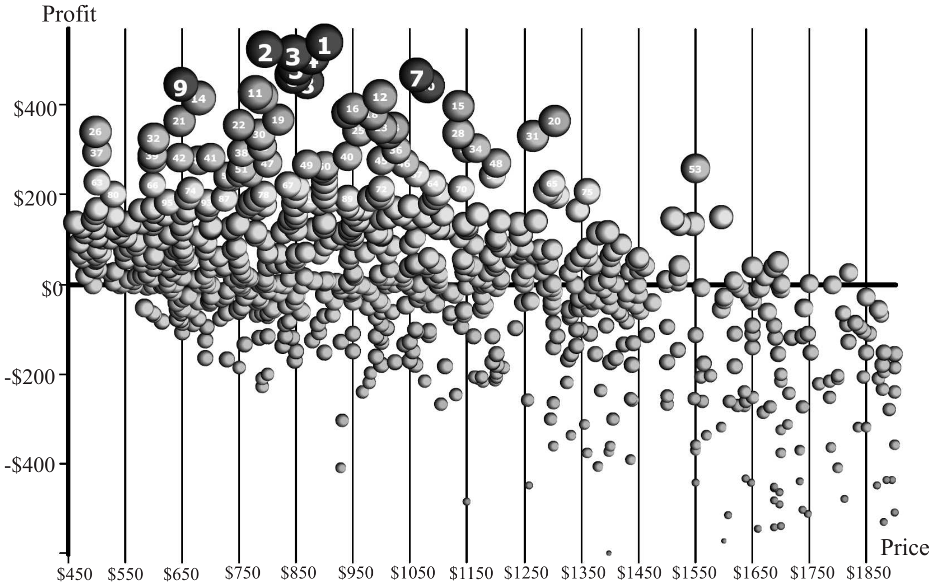
Interestingly, there is a phenomenon revealed by the features that are easy perceived by users, such as *Installed Memory*, *Max. Horizontal Res.* (number of horizontal pixels on screen), *L2-Cache* and *Processor Speed*. That is: those features have considerably less effect on prices in high-end than in mid-range and low-end market segments. This phenomenon can be explained by the fact that "luxury" goods justify their price more by attributes such as brand-label, exclusive features and physical appearance rather than for their configuration.

3.4 Recommendations for users

3.4.1 Top-10 recommendations

After the quantitative evaluation (i.e. regression error) and qualitative assessment (i.e. agreement with market facts) using 3FML model, the resulting functions provided reasonable estimates of price and support elements to explainable

Figure 5: Visualization of 836 laptops recommendation ranking



recommendations such as rankings and weights of attributes. After obtaining the estimates of prices, the profit for each laptop is calculated from the difference between this estimate and real price. Table 9 shows the top-10 recommendations with the highest profit among all 836 laptops.

The first and second top-ranked laptops have similar configurations, but even small differences make comparison difficult at first sight. The second laptop has better price, more memory, docking station and ports replicator slots. Unlike, the former has higher screen resolution and a fingerprint sensor. These differences can be compared quantitatively with the help of β coefficients provided by the model. However, a better explanation of the #1 recommended choice is a market fact extracted from the obtained manufacturer ranking showed in Table 10. The three regression models identify the *Lenovo*™ brand better ranked than *HP*™. Therefore, the first recommended laptop becomes a "best deal" given the standard prices of Lenovo at the time. Similarly, recommendations #7, #9 and #10 seem to get their high user profit not because of their configuration features, but because of their label *Sony*™, which do better positions on the ranking of manufacturers than its counterparts.

Second and third recommendations only differ in *Processor Speed* attribute. Clearly, the estimated cost of that differentia is the numerical difference between their estimated prices, which is $42. Nevertheless, their real price difference is $50. This explains the order of position in the ranking assigned by the recommender system to the #2 and #3 recommendations. More pair-wise comparisons and evaluations could be made but are omitted due to space limitations.

These paired comparisons become cognitively more difficult when the number of features, differences and instances increases. However, the proposed recommender method provides reasonable explanations no matter how much data is involved, and these can be provided by user request. This is important because cold-start recommender systems need to establish trust in users due of the lack of collaborative support.

3.4.2 2D Visualization

Ordered lists are the most common format to present recommendations to users. However, despite having such an ordination, establish the most appropriated choice for a particular user is a difficult task. Therefore, we propose a novel visualization method for our recommender system. The purpose of this is to enable users to build a mental model of the market. When users do not have a clear aim or a defined budget, this tool provides a rough idea of the number of options and prices. In addition, visualization can help the short-term memory decreasing cognitive load and highlight the recommended options.

The proposed 2D visualization is shown in Figure 5. The horizontal axe represents actual price and the vertical axe represents the profit, which is the difference between the estimated and actual price. Each laptop is represented as a bubble, where larger radius and warmer colors (darker gray in the grayscale version) means higher profit-price differences. Besides, the number of ranking was included in the top-99 recommendations.

This visualization highlights other "best deals" that are hidden in the ranking list. For instance, consider recommendation #53 (see Figure 5 in the coordinates $1550 price and $260 profit). Perhaps this is an interesting option to consider if user's budget is over $1500. Similarly, recommendation #26 can be quickly identified as the best option for buyer on a low budget.

The proposed visualization also allows a qualitative assessment of the price-estimation function. For instance, consider the laptops above $1300, this function has difficulties to predict prices using the current set of features, which in turn appears to be very effective for mid-range prices. This problem could be solved indentifying and adding to the set of attributes those distinctive features of high-end laptops, namely: shockproof devices, special designs, colors, housing materials, exclusive options, etc.

Table 9: Detailed top-10 ranked recomendations

Recommendation rank →	#1	#2	#3	#4	#5
Horizontal Resol.	900 pixels	800 pixels	800 pixels	1536 pixels	768 pixels
Memory Tech.	DDR2	DDR2	DDR2	DDR2	DDR2
Inst. Memory	512 MB	1024 MB	1024 MB	1024 MB	1024 MB
Family	Core Duo	Core Duo	Core Duo	Core2 Duo	Core2 Duo
Processor Speed	1830 GHz	1830 GHz	2000 GHz	1500 GHz	2000 GHz
L2 Cache	?*	?	?	2048 kB	4096 kB
Hyper Transp	?	?	?	667 Mhz	667 Mhz
Thermal Design	?	?	?	35	34
Process Tech.	?	?	?	65nm	65nm
Manufacturer	Lenovo	HP	HP	Lenovo	HP
Op. System	WinXPP	WinXPP	WinXPP	WinVB	WinXPP
Warranty Days	1095	1095	1095	365 W	1095 W
IBDPFWC**	YNYYYNN	YYYYYNN	YYYYYNN	NNYYNNN	YYYYYNN
Actual Price	$ 899	$ 795	$ 845	$ 875	$ 849
Estimated Price	$ 1,438	$ 1,319	$ 1,361	$ 1,383	$ 1,332
Profit	$ 539	$ 524	$ 516	$ 508	$ 483

Recommendation rank →	#6	#7	#8	#9	#10
Horizontal Resol.	1050 pixels	800 pixels	800 pixels	800 pixels	800
Memory Tech.	DDR2	DDR2	DDR2	DDR2	DDR2
Inst. Memory	1024 MB	1024 MB	1024 MB	512 MB	1024 MB
Family	Core Duo	Core2 Duo	Core Duo	Core Duo	Core2 Duo
Processor Speed	2000 GHz	2160 GHz	1830 GHz	1660 GHz	2000 GHz
L2 Cache	2048 kB	4096 kB	?*	2048 kB	4096 kB
Hyper Transp	667 Mhz	667 Mhz	?	667 Mhz	800 Mhz
Thermal Design	31 W	34 W	?	31 W	35 W
Process Tech.	65nm	65nm	?	65nm	65nm
Manufacturer	Lenovo	Sony	HP	Sony	Sony
Op. System	WinXP_Pro	WinXP_Pro	WinXP_Pro	WinXP_Pro	V_Business
Warranty Days	365	365	1095	365	365
I BDPFWC**	NYNNYNN	NYYYNNN	YYYYYNN	NNYYNNNN	NYYYYNN
Actual Price	$ 845	$ 1,060	$ 868	$ 649	$ 1,080
Estimated Price	$ 1,312	$ 1,526	$ 1,319	$ 1,093	$ 1,522
Profit	$ 467	$ 466	$ 451	$ 444	$ 442

* Question mark stands for missing values.

** Initials I B D P F W C stand for *Infrared*, *Bluetooth*, *Docking Station*, *Port Replicator*, *Fingerprint*, *Subwoofer* and *CDMA*.

Table 10: Average ranking of *Manufacturer* attribute using 3FML at *Laptops_21_836* data set

Base model		Upper model		Lower model	
Asus	10.6(0.7)	Dell	10.0(0.9)	Asus	10.2(0.8)
Sony	9.6(1.2)	Fujitsu	9.2(1.0)	Fujitsu	9.6(1.5)
Fujitsu	8.4(1.4)	Sony	7.4(1.8)	Sony	8.1(1.2)
Dell	7.9(0.9)	Asus	6.8(1.8)	Dell	7.3(1.4)
Apple	7.0(2.4)	Apple	6.1(1.3)	Apple	6.7(2.0)
Lenovo	6.5(1.6)	Lenovo	4.1(2.4)	Lenovo	5.2(1.5)
Toshiba	5.0(1.2)	Gateway	4.0(2.9)	Toshiba	4.7(1.5)
Acer	4.7(1.2)	Averatec	3.8(1.3)	Acer	3.9(1.3)
HP	2.3(0.7)	Acer	3.6(1.3)	HP	3.4(1.7)
Averatec	2.1(1.3)	HP	3.3(1.9)	Gateway	1.9(0.8)
Gateway	1.9(1.0)	Toshiba	2.4(1.5)	Averatec	1.9(2.1)

4. CONCLUSIONS

We presented a novel product recommender system based on an interpretable price-estimation function, which estimates the economic benefit for the customer to buy a product in a particular market. Accurate and interpretable price estimations were obtained using the 3FML (three-function meta-learner) method. This regression method allows the combination of an interpretable regressor (e.g. LMS) to estimate prices and an uninterpretable regressor (e.g. SVR) to identify the latent class of each product. The combined model obtained better price estimates than LMS, SVR and M5P regression tree, while it kept a high level of interpretation.

The proposed method was tested with real-market data from a data set of laptops. The obtained price-estimation model was interpretable, allowing evaluation and refinement by domain experts and ensuring that price estimates are a coherent consequence of the product features. In addition, the obtained recommendations are easy to understand by users. For instance, feature rankings (e.g. ranking of CPU) and feature price contributions (e.g. cost per GB of main memory) are provided. Importantly, while the price estimates are obtained in a supervised way, other domain knowledge is extracted in a non-supervised way. Although the proposed method was tested in a particular domain (i.e. laptops), this same process can be applied to other domains that exhibit similar number of options and features.

Moreover, a user-friendly visualization method for recommendations was proposed using a 2D Cartesian metaphor and concrete variables such as cost and profit. This visualization allows users to make a quick mental map of a large market to explore and identify recommendations in different price ranges.

In conclusion, the proposed method is flexible and can be useful in e-commerce scenarios with products that allow the construction of price-estimation functions, such as customer-electronics products and others. Finally, our method fills a gap where recommender systems based on historical information fail because of the lack of such information.

5. ACKNOWLEDGEMENTS

This research is funded in part by the Bogota Research Division (DIB) at the National University of Colombia, and throught a grant from Colciencias, project 110152128465.

6. REFERENCES

[1] E. Alpaydin. *Introduction to Machine Learning*. The MIT Press, October 2004.

[2] M. Balabanovic and Y. Shoham. Fab: Content-based collaborative recommendation. *Communications of the Association for Computing Machinery*, 40(3):66–72, 1997.

[3] C. Becerra and F. Gonzalez. 3-functions meta-learner algorithm: a mixture of experts technique to improve regression models. In *DMIN08: Proceedings of the 4th international conference on data mining*, Las Vegas, NV, USA., 2008.

[4] J. Bennet and S. Lanning. The netflix prize. In *KDD Cup and Workshop*, 2007.

[5] R. T. Eckenrode. Weighting multiple criteria. *Management Sciences*, 12:180–192, 1965.

[6] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35(12):61–70, 1992.

[7] M. A. Hall. Correlation-based feature selection for discrete and numeric class machine learning. In *ICML '00: Proceedings of the 17th International Conference on Machine Learning*, pages 359–366, San Francisco, CA, USA, 2000.

[8] E. Han and G. Karypis. Feature-based recommendation system. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 446–452, 2005.

[9] Borchers A. Riedl J. Herlocker J. L., Konstan J. A. An algorithmic framework for performing collaborative filtering. In *Proc. 22nd ACM SIGIR Conference on Information Retrieval*, pages 230–237, 1999.

[10] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. on Information Systems*, pages 89–115, 2004.

[11] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Comput.*, 3(1):79–87, 1991.

[12] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

[13] M. Mandl, A. Felfernig, E. Teppan, and M. Schubert. Consumer decision making in knowledge-based recommendation. *Journal of Intelligent Information Systems*, 2010.

[14] G. A. Miller. The magical number seven: Plus or minus two: Some limits on our capability for processing information. *Psychological Review*, 63(2):81–97, 1956.

[15] D. C. Montgomery, E. A. Peck, and G. G. Vining. *Introduction to linear regression analysis*. Wiley Interscience, 2006.

[16] R. J. Quinlan. Learning with continuous classes. In *5th Australian Joint Conference on Artificial Intelligence*, pages 343–348, 1992.

[17] T. L. Saaty and M. S. Ozdemir. Why the magic number seven plus or minus two. *Mathematical and Computer Modelling*, 38(3):233–244, 2003.

[18] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock. Methods and metrics for cold-start recommendations. In *SIGIR*, 2002.

[19] B. Scholkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[20] B. Schwartz. The tyranny of choice. *Scientific American*, pages 71–75, April 2004.

[21] Y. Wang and I. H. Witten. Induction of model trees for predicting continuous classes. In *Poster papers of the 9th European Conference on Machine Learning*. Springer, 1997.

[22] K. Yoon and C. Hwang. Multiple attribute decision making: An introduction. *Sage University papers*, 7(104), 1995.

[23] L. Zadeh. *Fuzzy Sets, Fuzzy Logic and Fuzzy Systems*. World Scientific, 1996.