Recommending Products using Preference Based Modeling

Hui Li HUI.LI@PEGA.COM
Pegasystems Inc. Vinoly building. 8th floor, Claude Debussylaan 20b, 1082MD Amsterdam, The Netherlands

Peter van der Putten PETER.VAN.DER.PUTTEN@PEGA.COM
Pegasystems Inc. Vinoly building. 8th floor, Claude Debussylaan 20b, 1082MD Amsterdam, The Netherlands and LIACS, Leiden University. P.O Box 9512, 2300 RA Leiden, The Netherlands

Maarten Keijzer MAARTEN.KEIJZER@PEGA.COM
Pegasystems Inc. Vinoly building. 8th floor, Claude Debussylaan 20b, 1082MD Amsterdam, The Netherlands

Keywords: product recommendations, online learning, preference modeling, ROC, Naive Bayes

Abstract

Common approaches to product recommendations are offer level propensity models and collaborative filtering. These methods are typically used for respectively a low or a high number of products, and come with specific limitations. In this paper we introduce preference based approaches to product recommendations, aimed at filling the gap between these methods. This approach can be used for any expected value decision problem, with choices across alternatives with common sets of attributes.

1. Introduction

A common way to approach product or offer recommendation problems is to construct a propensity model for every product predicting the likelihood of a positive response. This approach becomes unmanageable when the number of products grows large, changes dynamically at run time or when outcomes are sparse. Alternative approaches are collaborative filtering or recommender systems, but these methods are typically limited to interaction or purchase history, and can't deal with so called 'cold start' situations when a given customer hasn't built up any history yet.

In this paper we present two preference based approaches to this problem. The first method uses explicit models to model customer preferences for specific product attributes, for example a brand or price range. The likelihood to accept an offer is then derived from the likelihood of accepting specific product or offer attributes, given customer, contextual and history data. In the second method simply a single model is used, and product attributes are added to the other inputs for the model. The preference based approach is not limited to product recommendations, but can apply to any decision problem leveraging an expected value framework with a large and dynamic number of alternatives to choose from.

The paper is organized as follows. In section 2 we review relevant background concepts. Section 3 outlines the proposed algorithms, followed by an experimental evaluation in section 4. We end the paper with a discussion (section 5) and a conclusion (section 6).

2. Background

Before presenting our proposed preference based methods in later sections, this section reviews some relevant concepts from decision making under uncertainty as well as the reference approaches for making product recommendations that we are aiming to complement.

2.1. Decision making under uncertainty

From an application point of view the scope of our problem domain is larger than classical ecommerce recommendations. We focus on so called Next Best Action (NBA) systems: engines that provide real time recommendations in all customer interactions, across digital channels such as the web, mobile and social networks, as well as in more physical domains such as contact centers, shops, ATMs etc. These recommenda-

Appearing in *Proceedings of BENELEARN 2013*.

tions may cover more purposes than just recommending products, and there are more factors that go into the decision than just the likelihood of an offer being accepted. In other words, the methods presented in later sections provide a simplified view of both the problem space and approach, but can be further generalized.

The concept of NBA may be new, but a classical framework can be used to model some of the decisions in NBA: expected value, later generalized to expected utility; an early example is Blaise Pascal in 1669 (Pascal, 1995). The idea is that when a choice needs to be made across alternative courses of action, the alternative needs to be chosen with the highest expected value, the summation of likelihood times value for each of the outcomes. For example, to decide between offering A, B or C you choose the offer with the highest likelihood to be accepted times the value (for example profit) on an accept. There are some issues with this approach, outside the scope of this paper, for example it assumes that a decision now does not impact future decisions. In practice it most certainly will; customers and particularly employees will lose trust in these systems if offers are recommended that are high value but low probability to be accepted.

In the scope of this paper the focus is on the likelihood of positive outcomes. Negative outcomes are ignored, as well as value and other business rules, strategies or priorities driving the decision. This can easily be added in real world implementations.

Another relevant decision theoretical concept is multi attribute utility theory (MAUT) (Dyer, 2005). As outlined in the introduction we aim to lift limitations of common propensity modeling and collaborative filtering by modeling customer preferences. The core idea of MAUT is that when choosing from alternatives, once can consider a common set of criteria, with a utility or preference function defined for each criterion. In our preference based methods we use product attributes, either as outcomes (for example predicting the likelihood of accepting an expensive product) or as inputs (predict offer acceptance based on price, in addition to customer and other contextual data), thus modeling preferences for each customer. This can be seen as a special case of a MAUT approach, in which attribute level utility functions are automatically learned.

2.2. Product recommendation algorithms

The preference based methods we will present in the next section onwards are meant to complement other common methods used for product recommendations, such as offer level propensity modeling and collaborative filtering.

In offer level propensity modeling one model is used per product to predict the likelihood of a positive outcome (click, accept, conversion), based on customer data, real time contextual data and past interaction and outcome history. This is then combined with value factors, exclusion rules and other business heuristics to priority rank recommendations. This lends itself well to domains with deep, offer specific decisioning such as financial services, but becomes impractical when there are thousands of products or more. This could lead to an explosion of models, the lack of outcome data for certain products could be an issue and the relationships and similarities between products are not leveraged.

An alternative approach that is often used for recommending across many products (up to hundreds of thousands or millions) is a collaborative filtering recommender system approach. Amazon is one of the classical examples: 'people that bought these books also bought...' This approach has limitations in the sense that only interaction history is leveraged, not necessarily other inputs such as customer characteristics or contextual data. It also suffers from the cold start problem: if there is no interaction history yet for a given customer or history is sparse, it will be hard to make relevant recommendations. We have previously experimented with a hybrid approach that combines customer and offer level modeling and decisioning with collaborative filtering, based on Mahout, which includes a Hadoop and MapReduce based recommender system (Pegasystems, 2012).

The preference based methods are targeting problem domains that are in between these two approaches, i.e. product recommendations for products with thousands to tens of thousands of product instances and single set of product attributes. This can be generalized to any decision problem with similar number of alternatives and a common set of criteria attributes, with automated learning of preferences over these criteria.

3. Preference Based Product Recommendations

In this section we describe the proposed algorithms for preference based product recommendations, which are using an online learning propensity model. We will introduce the online learning method and then show how such a model can be used for composing the two preference based approaches.

3.1. Adaptive models

Our solution to self learning models is a closed-loop system that automates the model creation, deployment, and monitoring process (Walker & Khoshafian, 2012). It is powered by self-learning algorithms that are able to learn from customer feedback in each interaction, therefore establishing customer preferences incrementally while collecting historical data. The adaptive model can use various sources of information about the customer as predictors, such as customer demographics information like age, gender, house, credit, balance, etc. and real time contextual data. It is essentially a scoring model, predicting the likelihood of a positive outcome such as a click on a banner, acceptance of an offer or actual conversion. The adaptive model covers the following steps in the learning and prediction life cycle:

- Data Analysis and pre-Processing: for all candidate predictors, the adaptive model keeps the counts of positive and negative outcomes as sufficient statistics, at a granular level. For numeric predictors this is done at the level of a large number of numeric bins (f.e. age=18-20, 21-23, etc), for nominal predictors for each of most frequent occuring nominal values (f.e. product code = A, B, C etc). We periodically group the numeric bins and nominal values into larger categories, by testing whether there is any significant difference in the likelihood of a positive outcome between bins or nominals. These larger categories will be used by the model, to ensure that the model is not just accurate but also robust, even when it is just starting to learn.

- Sub set feature selection: this is done by analyzing the correlation between candidate predictors. We use an online algorithm to keep track of correlations between candidate predictors, and only the best ones out of each group are used for the model. A correlation threshold can be configured to control the degree of correlations for filtering. Any other real time sub set feature selection method could have been used in this step.

- Model Scoring: adaptive models use a Naive Bayes like technique to combine the selected predictors to generate a score, thus adaptive model generates robust and highly predictive scoring models. The quality measure used for evaluating a scoring model is the Coefficient of Concordance (CoC) (Lin, 1989). Rank concordance methods are preferred because outcome distributions may be highly unbalanced.

- Post-processing: typically uncalibrated scores produced by the scoring model cannot be used directly as probabilities, as underlying model assumptions are rarely met by real world data. Naive Bayes is known to perform surprisingly well for ranking based on model scores, even if the independence assumption of inputs is violated. However, the absolute estimate of probability may suffer in this case (Domingos & Pazzani, 1997; van der Putten & van Someren, 2004), and a proper estimate is key for comparing recommendations properly. The adaptive model thus implements an algorithm to transform model scores to propensities, by binning on the model score range and estimating the likelihood of a positive event for each of the bins. This concludes the adaptive life cycle from the raw predictor inputs to propensities.

3.2. Turning Real-time Contextual Data and Interaction History into Predictors

We have mentioned that adaptive models are able to use customer demographics as input data. During the customer interaction, real-time contextual data can also be taken into account as predictors for the adaptive model. We list a number of potentially predictive data points during a customer interaction:

- Customer intent information such as reason for a call into a contact center (leaving, enquiring, complaining)

- Channel specifics such as the the type of agent that handles the call, browsing history, device used for the interaction

- History about previous recommendations, interactions, outcomes and behavior

In many situations, real time contextual information turns out to be very predictive. The experiments will show how important such attributes can be.

3.3. Preference based methods

As discussed the goal of this paper is to provide preference based approaches, complementing the reference approaches of either using one model per product or using a collaborative filtering recommender system approach.

The first preference based approach is the so-called "*Single Adaptive*" approach. Rather than using one model per product learning from customer and contextual predictors (or leveraging collaborative filtering), only a single model is used for all products in a

product category that also takes product attributes as an input, so that we can scale towards many product instances that can change dynamically at run time.

The second preference based approach is the "*Combined Adaptive*" approach. It is a two-layer method that combines a set of propensity models predicting the likelihood of certain product attributes to be accepted (f.e. price ranges, brands etc), as opposed to products, thus modeling the "preferences" of customers towards certain product features. This approach is elaborated as follows.

We reuse the binning and grouping capabilities in the adaptive model to determine a small number of binned product attributes to focus on as outcomes. For example, the product attribute "Color" has three resulting bins: "Red", "Blue", and "Others". We then create adaptive models to track a user's propensity towards accepting these product attributes (likelihood of accepting a "Red" product etc). Note that the predictors now contain only customer and contextual data, but no product attributes.

Second we combine the propensities of all product attributes to generate a final propensity for each product instance, since one product is uniquely characterized by the set of product attributes. The combination methods can include a simple average over all propensities, or a weighted average in which weights are based on the performance of the adaptive models.

The binning and grouping of predictor values are critical to keep the number of models tractable in this approach. For example, with 10 attributes and 10 possible bins for each attribute, we will define $10 * 10 = 100$ adaptive models. In turn we could in theory model $10^{10} = 1$ *billion* different products, though in practice we will need to score all eligible product instances for a single customer interaction.

4. Empirical Evaluation

In this section we present the empirical evaluation of the proposed approaches for customer behavior prediction, based on an Event Recommendation Dataset. We first describe the data and the experimental setup in detail. Then we further analyze and compare the performance of predictors and algorithms under study.

4.1. Experimental Setup

We use a publicly available dataset from the Event Recommendation Engine Challenge, which is a data mining competition hosted on Kaggle (Kaggle, 2013)[1]. The task is to predict what cultural events users will be interested in based on events they bought tickets for in the past, user demographic information, and what events they have seen and clicked on. The data that we used for experiment include the following information:

- **Interaction data** contains which user is interested in which event. It has the following columns: *userid*, *eventid*, *invited*, *timestamp*, *interested*. "Interested" is a binary variable indicating whether a user has seen and clicked on the event.

- **User data** contains demographic data about users, including the following columns: *userid*, *locale*, *birthyear*, *gender*, *joinedAt*, *location*, and *timezone*.

- **Event data** contains around 110 columns of information about events. They include *eventid*, *starttime*, *city*, *state*, *zip*, *country*, *latitude*, and *longitude*. The other 101 columns are count1, count2, ..., count100, count-other, where countN represents the number of times the Nth most common word stem appears in the name or description of the event. count-other is a count of the rest of the words whose stem was not one of the 100 most common stems.

The interaction data has 15398 records in total, and it contains 2000+ unique users and 8000+ events. We use a 80-20 split into train and test datasets. Stratified sampling on the user's total number of events is performed for splitting.

We use "Coefficient of Concordance" (CoC) (Lin, 1989) to measure the performance of predictors and models. CoC measures how good a model in discriminating good cases from bad cases. It is a value between 0.5 (random distribution) and 1 (perfect discrimination). For binary outcomes, CoC is identical to the area under the receiver operating characteristic (ROC) curve (Fawcett, 2006) (AUC). The AUC is related to the Gini coefficient G by the formula G = 2 * AUC - 1.

We describe in detail the definition of models based on the binning of product features using the event dataset.

Name	Role	Binned Intervals	Grouped Intervals	Grouped Perf...ance
lng	Predictor	201	8	62.56
lat	Predictor	201	8	62.32
city	Predictor	201	4	57.3
country	Predictor	80	4	56.83
c_other	Predictor	200	9	56.63
c_52	Predictor	16	3	55.56
c_3	Predictor	31	4	53.97
c_28	Predictor	12	2	53.56
c_6	Predictor	73	5	53.48
c_8	Predictor	20	3	53.29

Figure 1. The predictive power of event features as predictors in Predictive Analytics Director (PAD). The binned intervals are the initial number of bins (max 200 bins + 1 bin for missing values). The grouped intervals are the number of bins after predictor grouping. Performance is measured using CoC.

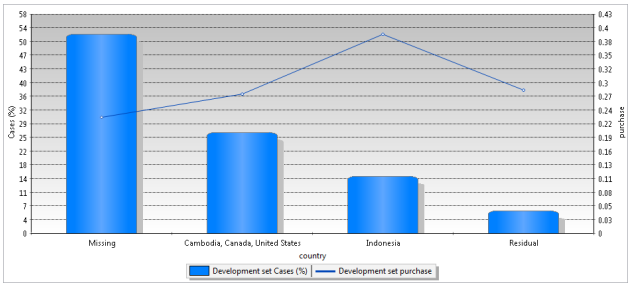

Figure 2. Grouping results for the event feature "Country". From initially 80 countries it is reduced to 4 groups. The percentage of cases in each group and the rate of clicking interest are shown.

4.2. Binning of Product Attributes

As described above, our products (or events), contain 110 features including location information and counts of popular words. If we want to define models on the values of numeric features, naturally we will have to discretize values into ranges or bins. Moreover, it is important to reduce the total number of models in the system, for which we use the predictor binning and grouping capabilities in the Predictive Analytics Director (PAD) tool. Figure 1 shows the top ten predictors ranked by the performance CoC. We can see that the location information such as latitude and longitude of the event, and some keywords (e.g. c-other, c52) are among the most predictive of all event features. After binning and further grouping of bins, the resulting bins of predictor values typically range from 2 to 10. Figure 2 shows one example binning result for feature "country". From initially 80 countries it is reduced to 4 groups. For instance, "Cambodia", "Canada", and "United States" are grouped into one statistically robust bin. Obviously these countries are very different, but from the perspective of accepting offers these are similar. As a result, we define 4 separate adaptive models to track the propensity of each bin. When making a prediction, an event's "country" value will fall into one of the bins, and the propensity of that model will be used as the likelihood for the feature "country".

The binning and grouping capabilities are essential to the Combined Adaptive approach. We choose the top 10 predictors and have in average 5 bins per predictor, therefore around 50 Adaptive models are created. These models can in turn be used to track propensities for a large number of events (e.g. there are 3 million events in the data set).

4.3. Analysis of Predictor Performance

Figure 3 shows the top 15 predictors by performance out of all predictors. We have stated in Section 3 that real-time contextual data gathered during the customer interactions are typically predictive information. In this particular experiment, we have derived a number of attributes from interaction history, for example:

pneg: count of negative responses in previous interactions with the user.

lastoffer_pos: whether the user's response to the last offer is positive or not.

ts_lastoffer: time duration since the last interaction with the user.

We can see that the derived attributes from interaction history are the best performing predictors, with CoC up to 70% (*pneg*). On the other side, the best predictor from user demographics is "locale" (53% CoC).

Figure 3 also shows the predictor grouping results after PAD correlation analysis. The correlated predictors are grouped together and we can choose to use only the best predictor in the group. For example, the set of derived attributes are largely correlated with each other, so we can select *pneg* from this group in the scoring model. There is a parameter called correlation threshold to control the granularity of grouping. Predictor grouping can be used to reduce the total number

[1]The notions of "Products" and "Events" are used exchangeable in this section. The events can be treated as products to be recommended to the users.

Group	Use group	Use p...ictor	Predictors	Grouped...ormance
Group 1	☑	☑	pneg	70.22
		☑	pcount	69.88
		☑	lastpos_id	69.78
		☑	ts_lastpos	69.7
		☑	ppos	69.56
		☑	ts_lastoffer	68.99
		☑	lastoffer_id	68.9
Group 2	☑	☑	lastoffer_pos	63.14
Group 3	☑	☑	lng	62.56
		☑	lat	62.32
Group 4	☑	☑	city	57.3
		☑	country	56.83
Group 5	☑	☑	c_other	56.63
Group 6	☑	☑	c_52	55.56
Group 7	☑	☑	c_3	53.97

Figure 3. Performance of the top 15 predictors and the predictor grouping results by PAD correlation analysis.

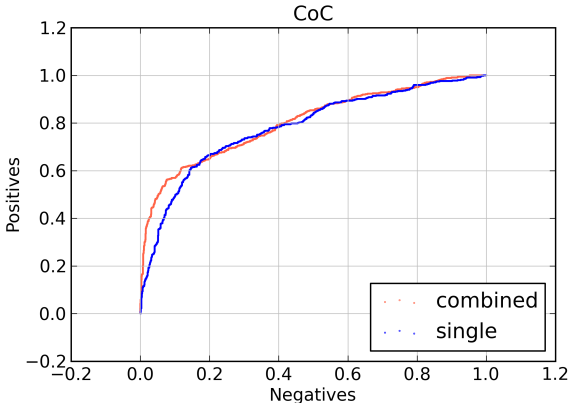

Figure 4. ROC Analysis and Comparison of the Single Adaptive and the Combined Adaptive approach. The ratio of total Negatives vs total Positives is 3 vs 1. CoCs of the Single Adaptive and Combined Adaptive are 0.783 and 0.803, respectively. The Combined approach outperforms clearly on the lower-left region of ROC Curve.

Table 1. Performance comparison of Naive Bayes, NBTree, Random Forests (WEKA), and the adaptive model approaches. The model build time is the time elapsed for model training. CoC is calculated on the test set.

Algorithm	CoC	Model Build Time
Naive Bayes	0.729	1.3 sec
NBTree	0.786	322 sec
Single Adaptive	0.783	13.6 sec
Combined Adaptive	0.803	13.6 sec
Random Forests	0.821	6.6 sec

of predictors used in scoring.

4.4. Model Performance Comparison

First we evaluate the performance of Single Adaptive compared to Combined Adaptive approach. Figure 4 shows the ROC analysis of both models. The CoCs are close with 0.783 and 0.803, respectively. When we analyze the ROC curve, we can see that the combined approach outperforms the single model more clearly on the lower-left region of the curve. This is the region of interest: we typically only show a very small number of top recommendations. The improvement by the combined approach is explained by the introduction of partitioning into the multidimensional data space. A manual partition by the product features, together with the reduction of models by binning and grouping, achieves a good balance between model complexity and predictive power.

Second we compare our adaptive approaches with a number of related classifiers in WEKA (Witten et al., 1999), and compare their performance as well as model building time. We can see that the adaptive models outperform the Naive Bayes classifier (with Entropy Discretizer) in WEKA significantly. This can be explained by the advanced data analysis functionality, especially binning, grouping, and correlation analysis in the adaptive model. Interestingly, the Combined Adaptive model based on manual partitioning of product features, outperforms the NBTree (Kohavi, 1996) approach. NBTree is a hybrid algorithm that constructs a decision tree (thus with automatic partitioning) with Naive Bayes classifiers as leaves. Overall, the Random Forests (Breiman & Schapire, 2001) classifier achieves the highest performance among the models under evaluation.

5. Discussions

The advantage of the Single Adaptive approach is its simplicity and it can be fully automated. The Combined Adaptive approach is able to achieve better performance, but requires manual work to define models based on product attributes. Both adaptive approaches learn online after every customer interaction, which is an advantage that enables real-time decisioning. Ensemble learning methods such as Random Forests are robust and perform well, but may require offline training. An interesting research direction is to experiment with online versions of these ensemble learning methods.

6. Conclusions

In this paper we propose a preference based product recommendation approach that is able to recommend across a large number of products. This approach is based on modeling the customer preferences towards product features, and combines individual preferences to form an overall recommendation for the product. The base model is an adaptive model that is able to model customer behavior online. We show that the proposed approach achieves comparable or better performance compared to other methods using a real-life web recommendation dataset. Therefore it establishes a favorable approach towards application domains that scale to thousands of product instances or more and at the same time meeting real-time learning requirements.

References

Breiman, L., & Schapire, E. (2001). Random forests. *Machine Learning* (pp. 5–32).

Domingos, P., & Pazzani, M. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, *29*, 103–130.

Dyer, J. (2005). MAUT multiattribute utility theory. In *Multiple criteria decision analysis: State of the art surveys*, vol. 78 of *International Series in Operations Research and Management Science*, 265–292. Springer New York.

Fawcett, T. (2006). An introduction to roc analysis. *Pattern Recognition Letters*, *27*, 861–874.

Kaggle (2013). Go from big data to big analytics. `http://www.kaggle.com`.

Kohavi, R. (1996). Scaling up the accuracy of naive-bayes classifiers: a decision-tree hybrid. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (pp. 202–207). AAAI Press.

Lin, L. I.-K. (1989). A concordance correlation coefficient to evaluate reproducibility. *Biometrics*, *45(1)*, 255–268.

Pascal, B. (1995). *Pensées*. Penguin Classics. English edition translated and introduced by A.J. Krailsheimer. Originally published as Pensées de M. Pascal sur La Religion et sure Quelque Autres Sujets. Guillaume Desprez, Paris, 1669.

Pegasystems (2012). *Hadoop big data support: Gain insights and take actions on big data in real-time* (Technical Report). Pegasystems Inc. Also available at `http://www.pega.com/resources/hadoop-big-data-support`.

van der Putten, P., & van Someren, M. (2004). A bias-variance analysis of a real world learning problem: The CoIL Challenge 2000. *Machine Learning*, *57*, 177–195.

Walker, R., & Khoshafian, S. (2012). *Adaptive bpm for adaptive enterprises* (Technical Report). Pegasystems Inc. See `http://www.pega.com/resources/adaptive-bpm`.

Witten, I. H., Frank, E., Trigg, L., Hall, M., Holmes, G., & Cunningham, S. J. (1999). Weka: Practical machine learning tools and techniques with java implementations.