

Using Text Mining to Infer Semantic Attributes for Retail Data Mining

Rayid Ghani and Andrew E. Fano
Accenture Technology Labs,
161 N Clark St, Chicago, IL 60601
Rayid.Ghani, Andrew.E.Fano@accenture.com

Abstract

Current Data Mining techniques usually do not have a mechanism to automatically infer semantic features inherent in the data being “mined”. The semantics are either injected in the initial stages (by feature construction) or by interpreting the results produced by the algorithms. Both of these techniques have proved effective but require a lot of human effort. In many domains, semantic information is implicitly available and can be extracted automatically to improve data mining systems. In this paper, we present a case study of a system that is trained to extract semantic features for apparel products and populate a knowledge base with these products and features. We show that semantic features of these items can be successfully extracted by applying text learning techniques to the descriptions obtained from websites of retailers. We also describe several applications of such a knowledge base of product semantics that we have built including recommender systems and competitive intelligence tools and provide evidence that our approach can successfully build a knowledge base with accurate facts which can then be used to create profiles of individual customers, groups of customers, or entire retail stores.

1 Introduction

Current Data Mining techniques usually do not automatically take into account the semantic features inherent in the data being “mined”. In most data mining applications, a large amount of transactional data is analyzed without a systematic method for “understanding” the items in the transactions or what they say about the customers who purchased those items. The majority of algorithms used to analyze transaction records from retail stores treat the items in a market basket as objects and represent them as categorical values with no associated semantics. For example, in an apparel retail store transaction, a basket may contain a shirt, a tie and a jacket. When data mining algorithms such as association rules, decision trees, neural networks

etc. are applied to this basket, they completely ignore what these items “mean” and the semantics associated with them. Instead, these items could just be replaced by distinct symbols, such as A, B and C or with apple, orange and banana for that matter, and the algorithms would produce the same results. The semantics of particular domains are injected into the data mining process in one of the following two stages: In the initial stage where the features to be used are constructed e.g. for decision trees or neural networks, feature engineering becomes an essential process that uses the domain knowledge of experts and provides it to the algorithm. The second instance where semantics are utilized is in interpreting the results. Once association rules or decision trees are generated and A and B are found to be correlated, the semantics are then used by humans to interpret them. Both of these methods of injecting semantics have been proved to be effective but at the same time are very costly and require a lot of human effort.

In many domains, semantic information is implicitly available and can be automatically extracted. In this paper, we describe a system that extracts semantic features for apparel products and populates a knowledge base with these products and features. We use apparel products and show that semantic features of these items can be successfully extracted by applying text learning techniques to the product names and descriptions obtained from websites of retailers. We also describe several applications of such a knowledge base of product semantics including recommender systems and competitive intelligence and provide evidence that our approach can successfully build a knowledge base with accurate facts which can then be used to create profiles of individual customers, groups of customers, or entire retail stores.

The work presented in this paper was motivated by discussions with CRM experts and retailers who currently analyze large amounts of transactional data but are unable to systematically understand the semantics of an item. For example, a clothing retailer would know that a particular customer bought a shirt and would also know the SKU, date, time, price, size, and color of a particular shirt that was pur-

chased. While there is some value to this data, there is a lot of information not being captured that would facilitate understanding the tastes of the customer and enable a variety of applications. For example, is the shirt flashy or conservative? How trendy is it? Is it casual or formal? These "softer" attributes that characterize products and the people who buy them tend not to be available for analysis in a systematic way.

In this paper, we describe our work on a system capable of inferring these kinds of attributes to enhance product databases. The system learns these attributes by applying text learning techniques to the product descriptions found on retailer web sites. This knowledge can be used to create profiles of individuals that can be used for recommendation systems that improve on traditional collaborative filtering approaches and can also be used to profile a retailer's positioning of their overall product assortment, how it changes over time, and how it compares to their competitors. Although the work described here is limited to the apparel domain and a particular set of features, we believe that this approach is relevant to a wider class of data mining problems and that extracting semantic clues and using them in data mining systems can add a potentially valuable dimension which existing data mining algorithms are not explicitly designed to handle.

2 Related Work

There has been some work in using textual sources to create knowledge bases consisting of objects and features associated with these objects. Craven et al.[3], as part of the WebKB group at Carnegie Mellon University built a system for crawling the Web (specifically, websites of CS departments in universities) and extract names of entities (students, faculty, courses, research projects, departments) and relations (course X is taught by faculty Y, faculty X is the advisor of student Y, etc.) by exploiting the content of the documents, as well as the link structure of the web. This system was used to populate a knowledge base in an effort to organize the Web into a more structured data source but the constructed knowledge base was not used to make any inferences. Recently, Ghani et al. [6] extended the WebKB framework by creating a knowledge base consisting of companies and associated features such as address, phone numbers, employee names, competitor names etc. extracted from semi-structured and free-text sources on the Web. They applied association rules, decision trees, relational learning algorithms to this knowledge base to infer facts about companies. Nahm & Mooney [11] also report some experiments with a hybrid system of rule-based and instance-based learning methods to discover soft-matching rules from textual databases automatically constructed via information extraction.

3 Overview of our approach

At a high level, our system deals with text associated with products to infer a predefined set of semantic features for each product. These features can generally be extracted from any information related to the product but in this paper, we only use the descriptions associated with each item. The features extracted are then used to populate a knowledge base, which we call the product semantics knowledge base. The process is described below.

1. Collect information about products
2. Define the set of features to be extracted
3. Label the data with values of the features
4. Train a classifier/extractor to use the labeled training data to now extract features from unseen data
5. Extract Semantic Features from new products by using the trained classifier/extractor
6. Populate a knowledge base with the products and corresponding features

4 Data Collection

We constructed a web crawler to visit web sites of several large apparel retail stores and extract names, urls, descriptions, prices and categories of all products available. This was done very cheaply by exploiting regularities in the html structure of the websites and manually writing wrappers¹. We realize that this restricts the collection of data from websites where we can construct wrappers and although automatically extracting names and descriptions of products from arbitrary websites would be an interesting application area for information extraction or segmentation algorithms[14], we decided to take the manual approach. The extracted items and features were placed in a database and a random subset was chosen to be labeled.

5 Defining the set of features to extract

After discussions with domain experts, we defined a set of features that would be useful to extract for each product. We believe that the choice of features should be made with particular applications in mind and that extensive domain knowledge should be used. We currently infer values for 8 kinds of attributes for each item but are in the process of identifying more features that are potentially interesting. The features we use are Age Group, Functionality,

¹In our case, the wrappers were simple regular expressions that took the html content of web pages into account and extracted specific pieces of information

Price point, Formality, Degree of Conservativeness, Degree of Sportiness, Degree of Trendiness, and Degree of Brand Appeal. More details including the possible values for each feature are given in Table 1.

The last four features (conservative, sportiness, trendiness, and brand appeal) have five possible values 1 to 5 where 1 corresponds to low and 5 is the highest (e.g. for trendiness, 1 would be not trendy at all and 5 would be extremely trendy).

6 Labeling Training Data

The data (product name, descriptions, categories, price) collected by crawling websites of apparel retailers was placed into a database and a small subset (600 products) was given to a group of fashion-aware people to label with respect to each of the features described in the previous section. They were presented with the description of the predefined set of features and the possible values that each feature could take (listed in Table 1).

7 Verifying Training Data

Since the data was divided into disjoint subsets and each subset was labeled by a different person, we wanted to make sure that the labeling done by each expert was consistent with the other experts and there were no glaring errors. One way to do that would be to now swap the subsets for each person and ask the other labelers to repeat the process on the other set. Obviously, this can get very expensive and we wanted to find a cheaper way to get a general idea of the consistency of the labeling process. For this purpose, we decided to generate association rules on the labeled data. We pooled all the data together and generated association rules between features of items using the apriori algorithm[1]. The particular implementation that we used was [2]. By treating each product as a transaction (basket) and the features as items in the basket, this scenario becomes analogous to the traditional market-basket analysis. For example, a product with some unique ID, say Polo V-Neck Shirt, which was labeled as Age Group: Teens, Functionality: Loungewear, Pricepoint: Average, Formality: Informal, Conservative: 3, Sportiness: 4, Trendiness:4, Brand Appeal: 4 becomes a basket with each feature value as an item. By using Apriori algorithm, we can derive a set of rules which relate multiple features over all products that were labeled. We ran apriori with both single and two-feature antecedents and consequents. Table 2 shows some sample rules that were generated.

By analyzing the association rules, we found a few inconsistencies in the labeling process where the labelers misunderstood the features. As we can see from Table 2, the

Table 2. Sample Association Rules

Rule	Support	Confidence
Informal <- Sportswear	24.5%	93.6%
Informal <- Loungewear	16.1%	82.3%
Informal <- Juniors	12.1%	89.4%
PricePoint=Ave <- BrandAppeal=2	8.8%	79.0%
BrandAppeal=5 <- Trendy=5	16.3%	91.2%
Sportswear <- Sporty=4	9.0%	85.7%
AgeGroup=Mature <- Trendy=1	9.4%	78.8%

association rules match our general intuition e.g. Apparel items labeled as informal were also labeled as sportswear and loungewear. Items with average prices did not have high brand appeal - this was probably because items with high brand appeal are usually more expensive. An interesting rule that was discovered was that items that were labeled as belonging to Mature Age group were also labeled as being not trendy at all.

Using association rules over the entire labeled data proved to be very useful in verifying the consistency of the labeling process done by several different labelers and we believe would be a useful tool for data verification in general where the labeling is performed by multiple people.

8 Training from the Labeled Data

We treat the learning problem as a traditional text classification problem and create one text classifier for each "semantic feature". For example, in the case of the Age Group feature, we classify the product into one of five classes (Juniors, Teens, GenX, Mature, All Ages). The initial algorithm used to perform this classification was Naive Bayes and a description is given below.

8.1 Naive Bayes

Naive Bayes is a simple but effective text classification algorithm[10, 9]. Naive Bayes defines an underlying generative model where, first a class is selected according to class prior probabilities. Then, the generator creates each word in a document by drawing from a multinomial distribution over words specific to the class. Thus, this model assumes each word in a document is generated independently of the others given the class. Naive Bayes forms maximum a posteriori estimates for the class-conditional probabilities for each word in the vocabulary V from labeled training data D . This is done by counting the frequency that word w_i occurs in all word occurrences for documents d_i in class c_j , supplemented with Laplace smoothing to avoid probabilities of zero:

Table 1. Details of features extracted from each product description

Feature Name	Possible Values	Description
Age Group	Juniors, Teens, GenX, Mature, All Ages	For what ages is this item most appropriate?
Functionality	Loungewear, Sportswear, Eveningwear, Business Casual, Business Formal	How will the item be used?
Pricepoint	Discount, Average, Luxury	Compared to other items of this kind is this item cheap or expensive?
Formality	Informal , Somewhat Formal, Very Formal	How formal is this item?
Conservative	1(gray suits) to 5 (Loud, flashy clothes)	Does this suggest the person is conservative or flashy?
Sportiness	1 to 5	
Trendiness	1 (Timeless Classic) to 5 (Current favorite)	Is this item popular now but likely to go out of style? or is it more timeless?
Brand Appeal	1(Brand makes the product unappealing) to 5 (high brand appeal)	Is the brand known and makes it appealing?

$$\Pr(w_t|c_j) = \frac{1 + \sum_{i=1}^{|\mathcal{D}|} N(w_t, d_i) \Pr(c_j|d_i)}{|V| + \sum_{s=1}^{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{D}|} N(w_s, d_i) \Pr(c_j|d_i)}, \quad (1)$$

where $N(w_t, d_i)$ is the count of the number of times word w_t occurs in document d_i , and where $\Pr(c_j|d_i) \in \{0, 1\}$ as given by the class label.

At classification time we use these estimated parameters by applying Bayes' rule to calculate the probability of each class.

$$\begin{aligned} \Pr(c_j|d_i) &\propto \Pr(c_j) \Pr(d_i|c_j) \\ &= \Pr(c_j) \prod_{k=1}^{|d_i|} \Pr(w_{d_i,k}|c_j). \end{aligned} \quad (2)$$

8.2 Incorporating Unlabeled Data using EM

In our initial data collection phase, we collected names and descriptions of thousands of women's apparel items from websites. Since the labeling process was expensive, we only labeled about 600 of those, leaving the rest as unlabeled. Recently, there has been much recent interest in supervised learning algorithms that combine information from labeled and unlabeled data. Such approaches include using Expectation-Maximization to estimate maximum a posteriori parameters of a generative model [12], using a generative model built from unlabeled data to perform discriminative classification [7], and using transductive inference for support vector machines to optimize performance on a

specific test set [8]. These results have shown that using unlabeled data can significantly decrease classification error, especially when labeled training data are sparse.

For the case of textual data in general, and product descriptions in particular, obtaining the data is very cheap. A simple crawler can be build and large amounts of unlabeled data can be collected for very little cost. Since we had a large number of product descriptions that were collected but unlabeled, we decided to use the Expectation-Maximization algorithm to combine labeled and unlabeled data for our task.

8.2.1 Expectation-Maximization

If we extend the supervised learning setting to include unlabeled data, the naive Bayes equations presented above are no longer adequate to find maximum a posteriori parameter estimates. The Expectation-Maximization (EM) technique can be used to find locally maximum parameter estimates.

EM is an iterative statistical technique for maximum likelihood estimation in problems with incomplete data [4]. Given a model of data generation, and data with some missing values, EM will locally maximize the likelihood of the parameters and give estimates for the missing values. The naive Bayes generative model allows for the application of EM for parameter estimation. In our scenario, the class labels of the unlabeled data are treated as the missing values.

EM is an iterative two-step process. Initial parameter estimates are set using standard naive Bayes from just the labeled documents. Then we iterate the E- and M-steps. The E-step calculates probabilistically-weighted class labels, $\Pr(c_j|d_i)$, for every unlabeled document using Equa-

tion 2. The M-step estimates new classifier parameters using all the documents, by Equation 1, where $\Pr(c_j|d_i)$ is now continuous, as given by the E-step. We iterate the E- and M-steps until the classifier converges.

9 Experimental Results

In order to evaluate the effectiveness of the algorithms described above for building an accurate knowledge base, we calculated classification accuracies using the labeled product descriptions and 5 fold cross-validation. The evaluation was performed for each attribute and the table below reports the accuracies. The first row in the table (baseline) gives the accuracies if the most frequent attribute value was predicted as the correct class. The experiments with Expectation-Maximization were run with the same amount of labeled data as Naive Bayes but with an additional 3500 unlabeled product descriptions.

Looking at Table 3, we can see that Naive Bayes outperforms our baseline for all the attributes. Using unlabeled data and combining it from the initially labeled product descriptions with EM helps improve the accuracy even further. To get a qualitative and intuitive feel for the performance of these algorithms and for the effectiveness of our approach, Table 4 gives a list of words which had high weights for some of the features that we used the naive bayes classifier to extract. These words were selected by scoring all the words according to their log-odds-ratio scores and picking the top 10 words. Looking at the words gives us a qualitative and intuitive idea of what type of words are indicative of each attribute and verifies our initial hypothesis that the marketing language associated with product does correspond to these softer attributes that we are trying to infer.

9.1 Results on a new test set

The results reported earlier in Table 3 are extremely encouraging but are indicative of the performance of the algorithms on a test set that follows a similar distribution as the training set. Since we first extracted and labeled product descriptions from a retail website and then used subsets of that data for training and testing (using 5 fold cross-validation), the results may not hold for test data that is drawn from a different distribution or a different retailer.

The results we report in Table 5 are obtained by training the algorithm on the same labeled data set as before but testing it on a small (125 items) new labeled data set collected from a variety of retailers. As we can observe, the results are consistently better than baseline and in some cases, even better than in Table 3. This results enables us to hypothesize that our system can be applied to a wide variety of data and can adapt to different distributions of test sets using the unlabeled data.

10 Applications

The knowledge base (KB) constructed by labeling unseen products has several applications. In this section, we describe some concrete applications that we have developed. The KB can be used to create profiles of individuals that can be used for recommender systems that improve on traditional collaborative filtering approaches and can also be used to profile a retailer’s positioning of their overall product assortment, how it changes over time, and how it compares to their competitors.

10.1 Recommender systems

Being able to analyze the text associated with products and map it to the set of predefined semantic features in real-time gives us the ability to create instant profiles of customers shopping in an online store. As the shopper browses products in a store, the system running in the background can extract the name and description of the items and using the trained classifiers, can infer semantic features of that product. This process can be used create instant profiles based on viewed items without knowing the identity of the shopper or the need to retrieve previous transaction data. This can be used to suggest subsequent products to new and infrequent customers for whom past transactional data may not be available. Of course, if historical data is available, our system can use that to build a better profile and recommend potentially more targeted products. We believe that this ability to engage and target new customers tackles one of the challenges currently faced by commercial recommender systems [13] and can help retain new customers.

We have built a prototype of a recommender system for women’s apparel items by using our knowledge base of product semantics. More details about the recommender system can be found in [5]. The knowledge base is populated with thousands of items and their associated semantic attributes inferred by the learning algorithm described in earlier sections. Our system monitors the browsing behavior of user browsing a retailer’s website and in real-time, extracts names and descriptions of products that they browse. The description text is then passed through our learned models and the semantic attributes of the products are inferred. For each product browsed, our system calculates $P(A_{i,j} | Product) \forall i, j$, where $A_{i,j}$ is the j th value of the i th attribute. The attributes are the semantic features described in Table 1 and the possible values for each attribute are also listed in the table. The user profile is constructed by combining these probabilities for each product browsed: User Profile =

$$\Pr(U_{i,j} | PastNItems) = \frac{1}{N} \sum_{k=1}^N \Pr(A_{i,j} | Item_k) \quad (3)$$

Table 3. Classification accuracies for each attribute using 5 fold cross-validation. Naive Bayes uses only labeled data and EM uses both labeled and unlabeled data.

Algorithm	AgeGroup	Functionality	Formality	Conservative	Sportiness	Trendiness	BrandAppeal
Baseline	29%	24%	68%	39%	49%	29%	36%
Naive Bayes	66%	57%	76%	80%	70%	69%	82%
EM	78%	70%	82%	84%	78%	80%	91%

Table 4. For each class, the table shows the ten words that are most highly weighted by one of our learned models. The weights shown represent the weighted log-odds ratio of the words given the class.

Brand Appeal=5(high)	Conservative=5(high)	Conservative=1(low)	Formality=Informal	Somewhat Formal
lauren ralph dkny kenneth cole imported	lauren ralph breasted seasonless trouser jones sport classic blazer	rose special leopard chemise straps flirty spray silk platform	jean tommy jeans denim sweater pocket neck tee hilfiger	jacket fully button skirt lines york seam crepe leather

agegroup=juniors	Functionality=Loungewear	Functionality=Partywear	Sportiness=5(high)	Trendiness=1(low)
jrs dkny jeans tee collegiate logo tommy polo short sneaker	chemise silk kimono calvin klein august lounge hilfiger robe gown	rock dress sateen length: skirt shirtdress open platform plaid flower	sneaker camp base rubber sole white miraclesuit athletic nylon mesh	lauren seasonless breasted trouser pocket carefree ralph blazer button

Table 5. Classification accuracies when trained on the same labeled data as before but tested on a new set of test data that is collected from a new set of retailers

Algorithm	AgeGroup	Functionality	Formality	Conservative	Sportiness	Trendiness	BrandAppeal
Naive Bayes	83%	45%	61%	70%	81%	80%	87%

The user profile is stored in terms of probabilities for each attribute value which allows us flexibility to include mixture models in future work in addition to being more robust to changes over time.

As the user browses products, the system compares the evolving profile against the products in the knowledge base, which has products classified into the same taxonomy of semantic features, and recommends the closest matching ones. Currently, we give equal weight to all products browsed when constructing the profile. In future work, we plan to experiment with different weighting schemes such as weighting recent items more than older ones.

There are two prevalent approaches to building recommender systems : Collaborative Filtering and Content-based. Collaborative Filtering systems work by collecting user feedback in the form of ratings for items in a given domain and exploit similarities and differences among profiles of several users to recommend an item. It recommends other items bought by people who also bought the current item of interest and completely ignores "what" the current item of interest was. Collaborative Filtering approaches suffer from two main problems: the "sparsity" problem that most customers do not browse or buy most products in a store and the "New Item" problem that a new product cannot be recommended to any customer until it has been browsed by a large enough number of customers. On the other hand, content-based methods provide recommendations by comparing representations of content contained in an item to representations of content that interests the user. A main criticism of content-based recommendation systems is that the recommendations provided are not very diverse. Since the system is powered solely by the user's preferences and the descriptions of the items browsed, it tends to recommend items "too" similar to the previous items of interest.

This type of recommender system improves on collaborative filtering as it would work for new products which users haven't browsed yet and can also present the user with explanations as to why they were recommended certain products (in terms of the semantic attributes). We believe that our system also performs better than standard content-based systems. Although content-based systems also use the words in the descriptions of the items, they traditionally use those words to learn one scoring function. For example, a classical content-based recommendation engine takes the text from the descriptions of all the items that user has browsed or bought and learns a model (usually a binary target function: "recommend" or "not recommend"). In contrast, our system changes the feature space from words (thousands of features) to the eight semantic attributes. This still enables us to recommend a wide variety of products unlike most content-based systems. Another potential advantage of our system is its ability to suggest products across categories (i.e. apparel styles may be predictive of furniture,

for example) which content-based systems are not able to do.

Since our goal was not to build the best recommendation system but rather to demonstrate the potential of a knowledge base of product semantics, we did not explore many approaches to building a user's profile. In future work, we plan to tackle the cases where a user's profile consists of a number of separate profiles. For example, if a user is looking for something for herself and also for her son, our system should be able to recognize that the items that the user is buying or browsing are inherently different. This could be done through mixture models where we construct a profile using a mixture of different profiles. Another potential solution is to monitor the users profile as they browse more and more products. Since each product can be thought of as a point in an n -dimensional Euclidean space (where n is the number of features, in our case, 8), we can calculate the distance of a new product from the current profile of the user. If a new product is "very" different from the current profile of the user (using thresholds based on cross-validation), it can be placed in a separate profile or treated as an outlier. We also plan to conduct user studies to validate the effectiveness of such a recommendation system based on these intermediate-level semantic features.

10.2 Store Profiling

Product recommendations are just one application that we have built so far. We also have a prototype that profiles retailers to build competitive intelligence applications. For example, by closely tracking the product offerings we can notice changes in the positioning of a retailer. We can track changes in the industry as a whole or specific competitors and compare it to the performance of retailers. By profiling their aggregate offerings, our system can enable retailers to notice changes in the positioning of product lines by competitor retailers and manufacturers. This ability to profile retailers enables strategic applications such as competitive comparisons, monitoring brand positioning, tracking trends over time, etc.

11 Conclusions and Future Work

We described our work on a system capable of inferring semantic attributes of products enabling us to enhance product databases for retailers. The system learns these attributes by applying supervised and semi-supervised learning techniques to the product descriptions found on retailer web sites. One of the main assumptions we make is the descriptions associated with the products accurately convey the semantic attributes. We believe that this assumption is justified because in most cases these descriptions are written by marketers to position the product in the consumer's mind

in a manner that implicitly suggests these softer attributes. The system can be bootstrapped from a small number of labeled training examples utilizes the large number of cheaply obtainable unlabeled examples (product descriptions) available from retail websites.

In the prototype we have built at Accenture Technology Labs, we currently have several applications for this type of a knowledge base. We use it to create profiles of individuals that can be used for recommendation systems that improve on traditional collaborative filtering approaches. The ability to infer the semantics of products can also be used to profile a retailer's positioning of their overall product assortment, how it changes over time, and how it compares to their competitors. Although the work described here is limited to the apparel domain and a particular set of features, we believe that this approach is relevant to a wider class of data mining problems. We believe that by going beyond the immediately available data, such as the fact that a customer is looking at or bought a product, and paying attention to what these products mean, we can increase the effectiveness of data mining applications.

References

- [1] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo. Fast discovery of association rules. In U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/The MIT Press, pages 307–328, 1996.
- [2] C. Borgelt. apriori. <http://fuzzy.cs.Uni-Magdeburg.de/~borgelt/>.
- [3] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery. Learning to construct knowledge bases from the world wide web. *Artificial Intelligence*, 118(1-2):69–114, 2000.
- [4] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.
- [5] R. Ghani and A. E. Fano. Building recommender systems using a knowledge base of product semantics. In *Proceedings of the Workshop on Recommendation and Personalization in ECommerce at the 2nd International Conference on Adaptive Hypermedia and Adaptive Web based Systems*, 2002.
- [6] R. Ghani, R. Jones, D. Mladenic, K. Nigam, and S. Slattery. Data mining on symbolic knowledge extracted from the web. In *Workshop on Text Mining at the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000.
- [7] T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in NIPS 11*, 1999.
- [8] T. Joachims. Transductive inference for text classification using support vector machines. In *Machine Learning: Proceedings of the Sixteenth International Conference*, 1999.
- [9] D. D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In *Machine Learning: ECML-98, Tenth European Conference on Machine Learning*, pages 4–15, 1998.
- [10] A. McCallum and K. Nigam. A comparison of event models for naive Bayes text classification. In *Learning for Text Categorization: Papers from the AAAI Workshop*, pages 41–48, 1998. Tech. rep. WS-98-05, AAAI Press.
- [11] U. Y. Nahm and R. J. Mooney. Text mining with information extraction. In *AAAI 2002 Spring Symposium on Mining Answers from Texts and Knowledge Bases*, 2002.
- [12] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134, 2000.
- [13] J. Schafer, J. Konstan, and J. Riedl. Electronic commerce recommender applications. *Journal of Data Mining and Knowledge Discovery*, 5:115–152, 2000.
- [14] K. Seymore, A. McCallum, and R. Rosenfeld. Learning hidden Markov model structure for information extraction. In *Machine Learning for Information Extraction: Papers from the AAAI Workshop*, 1999. Tech. rep. WS-99-11, AAAI Press.