Toward a Product Search Engine based on User Reviews

Paolo Fosci and Giuseppe Psaila
Università di Bergamo, Viale Marconi 5, I-24044 Dalmine, BG, Italy

Keywords: Product Search Engine, User Reviews, Itemset Mining.

Abstract: We address the problem of developing a method for retrieving products exploiting product user-reviews that can be found on the internet. For this purpose, we introduce a ranking model based on the concept of itemset mining of frequent terms. The prototype search engine that implements the proposed retrieval model is illustrated, and a preliminary evaluation on a real data set is discussed.

1 INTRODUCTION

"What is a movie with great funny hilarious jokes?" among a list of thousands of movies that could be watched on an on demand TV? Similar questions could be stated by a user for any kind of product or service, but an incredible source of opinions about products is represented by user reviews, i.e., comments posted by other people.

In the era of Web 2.0, user reviews become an important source of reputation, that other users could exploit to find out the best product they are looking for, or to avoid the worst ones. Therefore, a novel search engine could be built, in order to exploit user reviews to rank products: a person that wants to find products with given characteristics or opinions might submit a text query containing the words that better characterize the properties and related opinions he/she is looking for; as a result, the search engine produces a ranked list, where the ranking measure is based on the content of reviews.

This paper is a first step toward the development of a product search engine based on users reviews. The main idea behind our work is the way user reviews are processed to summarize their content. Common carrying opinion words are often used by review writers when they have more or less the same opinion about a product; thus, the extraction of sets of words that are frequently used by writers of reviews concerning the same product can be seen as a problem of *frequent itemset mining*.

Thus, the set of extracted itemsets gives a summarized view of the reviews. The set of extracted itemset can be queried to find the best products that match the text query.

In this paper, we present the main ideas about the exploitation of itemsets to summarize product reviews and query the collection of products. In particular, we present a novel retrieval model based on this concept, whose goal is to define a suitable ranking measure for products based on how the user query matches the content of product reviews. At this point, an evaluation is performed, where we discuss the effectiveness of our approach.

This paper is organized as follow. Section 2 presents a short survey of related works. Section 3 adapts the concept of itemset mining to our context. Section 4 introduces the proposed retrieval model. Section 5 presents a preliminary evaluation on a real data set. Finally Section 6 draws the conclusions.

2 RELATED WORK

Our research work is in the middle of several research areas: data mining, recommendation systems, search engines. In the last 2 decades, an incredible amount of research works have been published in the above mentioned areas. We do no not want to report an exhaustive review of the last twenty years, but focus on the topic of our work, considering the recent works that, we think, better represent the current state of the art of related work.

In the area of recommendation systems, several works exploited data mining techniques for building a model of user preferences and use this model to recommend products to potential customers. In particular, concerning recommendation systems that incorporate data mining technique, we can cite (Sandvig et al., 2007), that exploits association rule mining.

Fosci P. and Psaila G..
Toward a Product Search Engine based on User Reviews.
DOI: 10.5220/0004051602230228
In *Proceedings of the International Conference on Data Technologies and Applications* (DATA-2012), pages 223-228
ISBN: 978-989-8565-18-1
Copyright © 2012 SCITEPRESS (Science and Technology Publications, Lda.)

Several works (Hu and Liu, 2004b; Liu et al., 2005; Kim et al., 2009) adopt association rule mining for analyzing customer reviews and extract opinion from them. In (Hu and Liu, 2004b), association rule mining is used to extract, from within customer reviews, relevant features that characterize opinions of users about products. In (Liu et al., 2005), a system to compare opinions about products is presented, where product reviews reports PROs and CONs; in particular, association rule mining is exploited to assign a positive or negative polarity to words (namely, adjectives) in product reviews, and use this polarity to rank the opinion about products (note that we are not interestd in polarity, itemsets are a mean to summarize text). The work in (Hu and Liu, 2004a) extracts, by means of an association rule mining technique, relevant features that summarize product reviews.

Other data mining techniques are used to analyze customer reviews. For instance, in (Ly et al., 2011) a sentence clustering technique is adopted. In (Chaovalit and Zhou, 2005), both supervised and unsupervised approaches are evalauted. A similar work is done in (Lee et al., 2008), where key classification techniques for opinion mining are discussed. The work in (Dave et al., 2003) presents a technique for semantic classifications of product reviews.

W.r.t. all these works, that adopts data mining, and association rule mining in particular, to summarize reviews or to summarize/discover/synthesize opinions about products, we exploit itemset mining for a different purpose. In our proposal, itemset mining is the basis for a retrieval model: words in the query are matched against product reviews to find out the products with reviews that better matches the query words. The ranking of products is performed by evaluating a small subset of all possible itemsets and their support is aggregated in a relevance value, used to produce the final ranked list of products.

For the best of our knowledge, no similar work has been done yet.

3 ITEMSET MINING AND PRODUCT REVIEWS

Our approach to product retrieval is based on the widely exploited notion of *itemset mining*. Originally (Agrawal and Srikant, 1994), itemset mining is the basic and computationally difficult step for association rule mining. Here, we are not interested in association rule mining, but only in itemset mining for analyzing sets of words that frequently occur together in reviews. In this Section, we first present the basic notions of itemset mining. Then, we adapt the concept to our context.

3.1 Basic Notions on Itemset Mining

The notion of *itemset mining* was introduced as a fundamental part of the process for mining association rules (Agrawal et al., 1993; Agrawal and Srikant, 1994). In a *transaction* database $Z = \{z_1, \dots, z_n\}$, a transaction z is a set of items $z = \{i_1, \dots, i_k\}$ sold together in transaction z.

A $k-$itemset $I = \{i_1, \dots, i_k\}$ is a set of k items. The support of I, denoted as $s(I)$ is the frequency with which the itemset occurs in the transaction database, i.e., $s(I) = |Z(I)|/|Z|$ (where $Z(I) \subseteq Z$ is the set of transactions $z \in Z$ such that $z \cap I = I$, i.e., the transactions that contain itemset I).

Given a *minimum support threshold* $\overline{s}$, an itemset I is said *large* if its support is no less than the threshold $\overline{s}$, i.e., $s(I) \geq \overline{s}$.

The problem known as *itemset mining* is the problem of extracting large itemsets from the transaction database Z, provided that a minimum threshold for itemset support is defined.

Notice that this problem has been widely and extensively studied in the last two decades. As far as the main results about efficient algorithms are concerned, we just cite the fundamental paper (Agrawal and Srikant, 1994), in which the *Apriori* algorithm was defined, that has enabled itemset mining on large databases.

3.2 Product Reviews and Itemsets

Consider now a product p (a movie, a camera, etc.) and a set of reviews $R(p) = \{r_1, \dots, r_k\}$, where each review is a text, i.e., a sequence of term occurrences $r_i =< t_1, \dots, t_s >$.

A set of reviews for a product p can be seen as a transaction database, where each review $r_i \in R(p)$ can be seen as a transaction, and terms t_j corresponds to items. Given a minimum threshold for support, a large $k-$itemset $I = \{t_1, \dots, t_k\}$ summarizes a relevant number of reviews in $R(p)$. Consequently, $R(p)$ can be represented (and summarized) by means of the set of large itemsets $IS(R(p))$ extracted from within reviews, and composed of terms in reviews.

Moving from the general idea described above, we now define the way to decline the concept for our context.

Definition 1: Consider the set of terms $T(R(p)) = \{t\}$ appearing in reviews $r \in R(p)$ and the set SW of stopwords. Consider also a minimum support threshold for single terms $\overline{s}_t \in (0, 1)$.
The set $\overline{T}(R(p)) \subseteq T(R(p))$ is the set of *Relevant*

Terms, such that $\forall t \in \overline{T}(R(p))$, it is $s(t) \geq \overline{s}_t$, $s(t) \times |R(p)| \geq 3$ and $t \notin SW$. □

In other words, we consider relevant a term if it is not a stopword and it is reasonably frequent in the reviews. Notice that the set *SW* of stopwards is prior knowledge for our approach.
Notice that condition $s(t) \times |R(p)| \geq 3$ discards terms that are present in less than 3 reviews, even though their support is greater than $\overline{s}_t$; in fact, terms that appear in only one single review are sporadic and not sufficiently mediated by the community to characterize the product (typical situation with a small number of reviews for a product).

Definition 2: Consider the set $\overline{T}(R(p))$ of *relevant words*, and a minimum support threshold for compound itemsets $\overline{s}_c$.
A $k-$itemset I with length $k \geq 2$ is said *relevant* if $I \cap \overline{T}(R(p)) = I$ and $s(I) \geq \overline{s}_c$.
The set of relevant itemsets for product reviews $R(p)$ is denoted as $\overline{IS}(R(p))$. □

In other words, we focus on itemsets composed only of relevant words whose support is greater than a minimum support threshold $\overline{s}_c$ that is possibly different w.r.t. the minimum support threshold $\overline{s}_t$ used for determining relevant terms.

The reason why we consider two different minimum support thresholds for single terms and compound itemsets is the following. Terms must be relevant alone, thus they must be sufficiently representative of community opinions. But compound itemsets may be rarer than single terms in reviews, so a different minimum support threshold for compound itemsets is considered.[1]

Therefore, we can state that given two minimum support thresholds $\overline{s}_t$ and $\overline{s}_c$ for, respectively, single terms and compound itemsets, a set $R(p)$ of product reviews for a product p is summarized by the set of single relevant terms $\overline{T}(R(p))$ an by the set of relevant compound itemsets $\overline{IS}(R(p))$.

4 RETRIEVAL MODEL

The key element of our proposal is the *Retrieval Model*, since it is used to rank products in order to let most significant products to emerge, i.e., the products whose reviews better match the query terms.

Consider a query $q = \{t_1, \ldots, t_n\}$ containing a number of terms $n \geq 1$. The query q itself is a $n-$itemset.

With $I_l(q)$ we denote a $l-$itemset composed by l terms in q (it is $1 \leq l \leq n$). We denote with D_q the set of all itemsets that can be obtained with terms in q, and with $D_q(l)$ the subset of itemsets of length l; notice that cardinality of $D_q(l)$ is $|D_q(l)| = \binom{n}{l}$ and notice that $|D| = 2^n - 1$, i.e., D is the power set of q without the empty itemset.

In order to define a rank for a product based on its reviews, we define the concept of *weight* for an itemset.

Definition 3: The weight of a $l-$itemset is denoted as $w_q(l)$. The weight of the single $n-$itemset q is, by definition, $w_q(n) = 1$, while for $1 \leq l < n$ is $w_q(l) = w_q(l+1)/\binom{n}{l}$. □

The rationale behind Definition 3 is the following. The topmost itemset, corresponding to the whole query, is the most important one, and it gives the full contribution to rank the product. In contrast, lower levels must contribute a little, but not as much as the topmost itemset; in particular, each non-topmost level contributes as one of the itemset in the upper level.

Notice, that this way, the overall weight of itemsets of level $(n-1)$ is 1, exactly as the single topmost itemset; reducing the size of itemsets, the contribution of each level quickly decreases.

Example 1: To illustrate, consider the following example query over a movie reviews collection: *great funny hilarious jokes*. Based on the previous definitions, the weights for itemsets are determined. In particular, the 4-*itemset* that coincides with the whole query is assigned weight 1. For each of the four 3-*itemsets*, the assigned weight is $w_q(3) = w_q(4)/\binom{4}{3} = 1/4 = 0.25$. For each 2-*itemset*, the weight is $w_q(2) = w_q(3)/\binom{4}{2} = 0.25/6 = 0.042$; the weight at this level dramatically decreases, so that the presence of only two terms together in the review gives a very little contribution to the overall product ranking.
Finally, 1-*itemsets* are assigned with weight $w_q(1) = w_q(2)/\binom{4}{1} = 0.042/4 = 0.01$. Notice that even though single terms are less than 2-*itemsets*, their contribution to the overall ranking is very low, in order to compensate the possible very high supports of single terms.
Figure 1 graphically illustrates the itemsets levels. □

We are now ready to define the *Product Relevance Value (PRV)*.

Definition 4: Consider a product p, the set of reviews $R(p)$ and the set of relevant terms and compound itemsets $F(p) = \overline{T}(R(p)) \cup \overline{IS}(p)$ that it is possible to extract from $R(p)$. Consider now a query q and the set of itemsets D_q included in q.

[1]In principle, we could think about different minimum thresholds for compound $k-$itemsets that somehow depend on itemset length k. We will consider this aspect in our future work.

#	*weight*	*l*	*itemsets*
1	1.0000	4	{funny,great,hilarious,jokes}
4	0.2500	3	{funny,great,hilarious} {funny,great,jokes} {funny,hilarious, jokes} {great,hilarious, jokes}
6	0.0417	2	{funny,great} {funny,hilarious} {funny,jokes} {great,hilarious} {great,jokes} {hilarious, jokes}
4	0.0104	1	{funny} {great} {hilarious} {jokes}

Figure 1: Itemsets levels for query *great funny hilarious jokes* and corresponding weights.

If we denote with $\overline{D}_q(p) = F(p) \cap D_q$ the set of itemsets included in q that can be actually extracted from within reviews of product p, the *Product Relevance Value (PRV)* for product p is defined as

$$PRV_q(p) = \sum_{I \in \overline{D}_q(p)} (w_q(|I|) \times s(I))$$

□

The rationale of the above definition is the following. For each itemset included in the query q and actually relevant in the reviews, its contribution to the overall relevance value is its support value multiplied by its weight. Notice that the longer the matched itemset, the stronger its contribution; the shorter the matched itemset, the smaller its contribution.

However, the definition of *PRV* does not take the number of reviews for each product into account. For this reason, we introduce the *Adaptive Product Relevance Value*.

Definition 5: Consider a product p, the set of reviews $R(p)$, the query q and the *Product Relevance Value* $PRV_q(p)$. The *Adaptive Product Relevance Value APRV* is defined as

$$APRV_q(p) = PRV_q(p) \times log_{10}(|R(p)|)$$

□

The rationale behind this variation of the relevance measure is that the same itemset I with a given support $s(I)$ is more effective in summarizing a product with a large number of reviews than a product with a very small number of reviews. The logarithm allows us to avoid that products with very large sets of reviews excessively domintae the other ones.

Consequently, based on this concepts, the goal of the retrieval task is to provide a ranked list of products, based either on the *Product Relevance Value* or on the *Adaptive Product Relevance Value*.

5 EVALUATION

We performed a preliminary evaluation of the proposed retrieval method. Our approach has been to design and then populate a database on our servers with data derived from user-reviews taken from the internet. Then we built a search engine that, performing a query on the database, retrieves the information about the items involved in the query and builds on them the itemsets to apply our model. Below we describe the source data set, discuss the query results based either on *PRV* or on *APRV*, observe the effects of different settings for minimum support thresholds.

5.1 Source Data Set

Products are described by a set of user-reviews grabbed from the internet. The web site source is `epinion.com`, from which we took only products with at least five reviews. In particular, in this experiment we considered user reviews concerning *movies*.

Our dataset is composed by a corpus of 324 products (movies). Each product has a number of reviews that varies from 5 to 509. The total number of analyzed reviews is 9701, and the average number of reviews per movie is 30.

While collecting terms, we do not matter about lower or upper case, and we do not consider puntuaction or stop-words. We have a dictionary with a list of about 600 distinct stop-words. Moreover, for each product, we collect only terms with support greater than or equal to threshold $\overline{s}_t = 0.1$. The support of terms varies from 0.1 to 1, with 0.33 as average value.

As a result, we collected 72802 distinct frequent terms (and a total of 709845 occurrences). The number of words in the set of review for a product p varies from a minimum of 61 words to a maximum of 36200; the average number of terms per product is 441.

From the point of view of the single-review, we have to say that the average length of each review is 73 terms; the number of terms per reviews varies from 1 to 441.

5.2 Query Results

We submitted the query: *"What is a movie with great funny hilarious jokes?"*; in this query, there are just 4 *query terms* to be searched (*"great, funny, hilarious, jokes"*), because the others are considered as stop-words.

We repeated the query processing with different minimum support thresholds $\overline{s}_c$: 0.0, 0.1, 0.2. The results of the experiments are reported in Tables 1 and Table 2.

Both Table 1.a and Table 1.b show results with $\overline{s}_c = 0.1$. For each product p we report the number of reviews (column *R*), the number of *query terms* hit

Table 1: Search result with MinSup: 0.1 according PRV ranking (a), and APRV ranking (b).

##	Product	R	H	IS	PVR	APVR
1	The Other Guys	17	4	15	1.0043	2.8454
2	Funny People	12	4	15	0.6016	1.4948
3	40-Year-Old Virgin	25	4	15	0.5179	1.6671
4	Paul	11	4	15	0.4830	1.1581
5	Eddie Murphy - Raw	10	4	13	0.2708	0.6236
6	Bridesmaids	11	4	11	0.2348	0.5631
7	Observe and Report	5	3	7	0.1792	0.2884
8	The Hangover	53	4	12	0.1773	0.7039
9	Hot Tub Time Machine	14	4	11	0.1734	0.4575
10	Get Him to the Greek	6	3	7	0.1684	0.3017
11	***Little Fockers***	***9***	***3***	***7***	***0.1644***	***0.3611***
12	Dinner for Schmucks	12	4	10	0.1432	0.3559
13	Bruno	16	4	10	0.1374	0.3809
14	Extract	5	3	7	0.1312	0.2112
15	Zombieland	22	4	9	0.1297	0.4010
16	The Proposal	28	4	11	0.1176	0.3917
17	Jennifer’s Body	8	3	7	0.1068	0.2220
18	I Love You, Man	11	3	7	0.1032	0.2475
19	Surviving Christmas	16	4	9	0.0996	0.2762
20	Kick-Ass	15	4	10	0.0875	0.2370

a)

##	Product	R	H	IS	PVR	APVR
1	The Other Guys	17	4	15	1.0043	2.8454
2	40-Year-Old Virgin	25	4	15	0.5179	1.6671
3	Funny People	12	4	15	0.6016	1.4948
4	Paul	11	4	15	0.4830	1.1581
5	The Hangover	53	4	12	0.1773	0.7039
6	Eddie Murphy - Raw	10	4	13	0.2708	0.6236
7	Bridesmaids	11	4	11	0.2348	0.5631
8	Hot Tub Time Machine	14	4	11	0.1734	0.4575
9	Zombieland	22	4	9	0.1297	0.4010
10	The Proposal	28	4	11	0.1176	0.3917
11	Bruno	16	4	10	0.1374	0.3809
12	***Little Fockers***	***9***	***3***	***7***	***0.1644***	***0.3611***
13	Anger Management	61	4	10	0.0874	0.3594
14	Dinner for Schmucks	12	4	10	0.1432	0.3559
15	Get Him to the Greek	6	3	7	0.1684	0.3017
16	When Harry met Sally	58	3	7	0.0727	0.2953
17	Observe and Report	5	3	7	0.1792	0.2884
18	Toy Story 3	28	4	8	0.0863	0.2876
19	Surviving Christmas	16	4	9	0.0996	0.2762
20	I Love You, Man	11	3	7	0.1032	0.2475

b)

Table 2: Search result according APRV ranking with MinSup=0.00 (a), and with MinSup=0.20 (b).

##	Product	R	H	IS	PVR	APVR
1	The Other Guys	17	4	15	1.0043	2.8454
2	40-Year-Old Virgin	25	4	15	0.5179	1.6671
3	Funny People	12	4	15	0.6016	1.4948
4	Paul	11	4	15	0.4830	1.1581
5	The Hangover	53	4	15	0.2716	1.0784
6	Eddie Murphy - Raw	10	4	15	0.3958	0.9114
7	Bridesmaids	11	4	15	0.3750	0.8992
8	Anger Management	61	4	15	0.1735	0.7132
9	Bruno	16	4	15	0.2493	0.6913
10	The Proposal	28	4	15	0.1726	0.5752
11	Toy Story 3	28	4	15	0.1652	0.5504
12	Zombieland	22	4	13	0.1695	0.5240
13	Scott Pilgrim vs. …	20	4	15	0.1672	0.5008
14	Dinner for Schmucks	12	4	13	0.1884	0.4681
15	Hot Tub Time Machine	14	4	11	0.1734	0.4575
16	Aladdin	79	4	15	0.0843	0.3682
17	***Little Fockers***	***9***	***3***	***7***	***0.1644***	***0.3611***
18	Kick-Ass	15	4	13	0.1236	0.3347
19	Surviving Christmas	16	4	11	0.1178	0.3267
20	Get Him to the Greek	6	3	7	0.1684	0.3017

a)

##	Product	R	H	IS	PVR	APVR
1	The Other Guys	17	4	15	1.0043	2.8454
2	40-Year-Old Virgin	25	4	12	0.2679	0.8624
3	Funny People	12	4	11	0.3446	0.8563
4	The Hangover	53	4	8	0.1309	0.5197
5	Bridesmaids	11	4	9	0.1818	0.4360
6	Zombieland	22	4	8	0.1241	0.3835
7	***Little Fockers***	***9***	***3***	***7***	***0.1644***	***0.3611***
8	Paul	11	4	9	0.1496	0.3588
9	Eddie Murphy - Raw	10	4	9	0.1542	0.3550
10	Hot Tub Time Machine	14	4	9	0.1317	0.3476
11	Bruno	16	4	8	0.1243	0.3448
12	Dinner for Schmucks	12	4	8	0.1293	0.3214
13	Get Him to the Greek	6	3	7	0.1684	0.3017
14	Observe and Report	5	3	7	0.1792	0.2884
15	Jennifer’s Body	8	3	7	0.1068	0.2220
16	National Lampoon’s…	83	3	5	0.0383	0.1691
17	The Proposal	28	4	6	0.0491	0.1636
18	Anger Management	61	4	6	0.0396	0.1629
19	Toy Story 3	28	4	6	0.0432	0.1438
20	I Love You, Man	11	3	6	0.0578	0.1385

b)

(column *H*), the number of mined itemsets (column *IS*), and the relevance values *PRV* and *APRV*.

In particular, Table 1.a lists the best 20 movies in reverse order of *PRV*. In contrast, Table 1.b lists the best 20 product in reverse order of *APRV*, that are not exactly the same movies. Movie in position 11 in Table 1.a is now in position 12: in effect, it has a small number of reviews, w.r.t. the other movies, and it looses a position in the ranked list. For this reason, we decide to adopt the *APRV* as standard ranking measure in the next experiments.

In order to evaluate the effect of the minimum support threshold value $\bar{s}_c$, we performed other two experiments with $\bar{s}_c = 0.0$, whose results are reported in Table 2.a, and $\bar{s}_c = 0.2$, whose results are reported in Table 2.b.

In particular, setting $\bar{s}_c = 0.0$ means that every itemset is considered, even with support lower than single term minimum support threshold $\bar{s}_t$. The effect is that a large number of lowly representative itemsets are considered, and affect the results. Looking at Table 2.a, movie *Little Fockers* drops to position 17 from position 12. But looking at reviews, this movie is a good sample for the query, thus this setting is countereffective.

Considering the setting with $\bar{s}_c = 0.2$ (Table 2.b), the higher minimum support threshold obtains, as an effect, that only strongly representative itemsets actually contribute to the value of *APRV*. Consequently, movie *Little Fockers* now occupies position 7, thus significantly improving its position.

Of course, these are preliminary results and a deeper study must be performed, evaluating precision and recall of the different settings.

6 CONCLUSIONS

In this paper we presented a novel retrieval model for a product search engine based on user-reviews taken from the internet.

The two basic ideas that reside under the search engine are *(1)* itemset computation at query time from frequent terms in user-reviews and *(2)* a ranking model that permits to weight these itemsets. The work is at an early stage, but based on our experiments we can say that the whole idea seems to be quite promising.

Future Works. Our retrieval model is very context-dependent so we have to develop automatic criteria to enrich a generic *stop-words* list with those terms that are too much context frequent. A too much context frequent term has no relevant semantic value, but nonetheless it can affect ranking in a distorted way. For examples, in the case we showed in this paper the terms *movie* and *movies* are considered *stop-words*.

Details apart, the next step is to improve our ranking model including the concept of *term-closeness*. At the moment while matching a query in a user-review we do not consider term position inside the review. We do believe that matching closer terms can lead closer to the real meaning of the query. So we want to develop an index based on *term-closeness* to affect our ranking model.

A further step is to hook our search engine to a dictionary or an ontology, like *Wordnet*, in order to better characterize words.

REFERENCES

Agrawal, R., Imielinski, T., and Swami, A. (1993). Database mining: A performance perspective. *IEEE Transactions on Knowledge and Data Engineering*, 5(6):914–925.

Agrawal, R. and Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th VLDB Conference*, Santiago, Chile.

Chaovalit, P. and Zhou, L. (2005). Movie review mining: a comparison between supervised and unsupervised classification approaches. In *HICSS*. IEEE Computer Society.

Dave, K., Lawrence, S., and Pennock, D. M. (2003). Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *WWW*, pages 519–528.

Hu, M. and Liu, B. (2004a). Mining and summarizing customer reviews. In Kim, W., Kohavi, R., Gehrke, J., and DuMouchel, W., editors, *KDD*, pages 168–177. ACM.

Hu, M. and Liu, B. (2004b). Mining opinion features in customer reviews. In McGuinness, D. L. and Ferguson, G., editors, *AAAI*, pages 755–760. AAAI Press / The MIT Press.

Kim, W., Ryu, J., Kim, K. I., and Kim, U.-M. (2009). A method for opinion mining of product reviews using association rules. In Sohn, S., Chen, L., Hwang, S., Cho, K., Kawata, S., Um, K., Ko, F. I. S., Kwack, K.-D., Lee, J. H., Kou, G., Nakamura, K., Fong, A. C. M., and Ma, P. C. M., editors, *Int. Conf. Interaction Sciences*, volume 403 of *ACM International Conference Proceeding Series*, pages 270–274. ACM.

Lee, D., Jeong, O.-R., and goo Lee, S. (2008). Opinion mining of customer feedback data on the web. In Kim, W. and Choi, H.-J., editors, *ICUIMC*, pages 230–235. ACM.

Liu, B., Hu, M., and Cheng, J. (2005). Opinion observer: analyzing and comparing opinions on the web. In Ellis, A. and Hagino, T., editors, *WWW*, pages 342–351. ACM.

Liu, D. T. and Xu, X. W. (2001). A review of web-based product data manageement systems. 44(1):251–262.

Ly, D. K., Sugiyama, K., Lin, Z., and Kan, M.-Y. (2011). Product review summarization based on facet identification and sentence clustering. *CoRR*, abs/1110.1428.

Sandvig, J. J., Mobasher, B., and Burke, R. D. (2007). Robustness of collaborative recommendation based on association rule mining. In Konstan, J. A., Riedl, J., and Smyth, B., editors, *RecSys*, pages 105–112. ACM.

Zhuang, L., Jing, F., and Zhu, X. (2006). Movie review mining and summarization. In Yu, P. S., Tsotras, V. J., Fox, E. A., and Liu, B., editors, *CIKM*, pages 43–50. ACM.