StarTrack: The Next Generation (of Product Review Management Tools)

Stefano Baccianella, Andrea Esuli and Fabrizio Sebastiani

Istituto di Scienza e Tecnologia dell'Informazione
Consiglio Nazionale delle Ricerche
Via Giuseppe Moruzzi 1 – 56124 Pisa, Italy
{*firstname.lastname*}@isti.cnr.it

Abstract. Online product reviews are increasingly being recognized as a goldmine of information for determining product and brand positioning, and more and more companies look for ways of digging this goldmine for nuggets of knowledge that they can then bring to bear in decision making. We present a software system, called StarTrack, that automatically rates a product review according to a number of "stars", i.e., according to how positive it is. In other words, given a text-only review (i.e., one with no explicit star-rating attached), StarTrack attempts to guess the star-rating that the reviewer would have attached to the review. StarTrack is thus useful for analyzing unstructured word-of-mouth on products, such as the comments and reviews about products that are to be found in spontaneous discussion forums, such as newsgroups, blogs, and the like. StarTrack is based on "machine learning" technology, and as such does not require any re-programming for porting it from one product domain to another. Based on the star-ratings it attributes to reviews, StarTrack can also rank the products in a given set according to how favourably they have been reviewed by consumers. We present controlled experiments in which we evaluate, on two large sets of product reviews crawled from the Web, the accuracy of StarTrack at (i) star-rating reviews and (ii) ranking the reviewed products based on the attributed star-ratings.

E quindi uscimmo a riveder le stelle.
(Dante Alighieri, *Inferno*, XXXIV 139)

1 Introduction

Online product reviews are becoming increasingly available across a variety of Web sites, and are being used more and more frequently by consumers in order to make purchase decisions from among competing products [7, 10, 16]. For example, according to a study performed by Gretzel and Yoo [10] on TripAdvisor[1], one of the most popular online review sites for tourism-related activities, with over 10 million travel reviews all posted by travelers, travel review readers perceive reviews posted by other consumers as having several advantages over information obtained from travel service providers. Almost two thirds of the readers think that reviews written by consumers contain up-to-date, enjoyable and reliable information. The same study also highlights the fact that, among the users that use

[1] http://www.tripadvisor.com/

the TripAdvisor online booking system, 97.7% are influenced by other travelers' reviews, and among them 77.9% use the reviews as a help to choose the best place to stay. Almost all respondents taking part in this survey answered that reviews (i) are a good way to learn about travel destinations and products, (ii) help with the evaluation of alternatives, and (iii) help to avoid places they would not enjoy. A clear majority of them also think that reviews increase confidence and help reduce risk by making it easier to imagine how a place will be like[2].

The importance of harnessing the information contained in online product reviews, and using it as a tool for decision making, is now widely recognized [5, 6, 8, 9]. Monitoring online product reviews in order to check how a product is perceived, in order to generate sales forecasts, and in order to steer the design, production, and marketing strategies of the company accordingly, is also recognized as essential in order to react dynamically to the evolving needs of the market. As a result, software tools that crawl the Web for product reviews, analyze them automatically, and generate out of them indicators useful to analysts and researchers, are going to become increasingly important for applications such as brand positioning, revenue forecasting, and the like.

Among the issues that the designers of these software tools need to address are (a) content aggregation, such as in pulling together product reviews from sources as disparate as e-magazines, newsgroups, blogs, and community Web sites; (b) content validation, as in filtering out fake reviews authored by people with vested interests [11]; and (c) content organization, as in automatically ranking competing products of similar type and price in terms of the satisfaction of consumers that have purchased the product before.

We describe a software tool that we have recently built and that addresses a problem related to issue (c), namely, rating (i.e., attributing a numerical score of satisfaction to) consumer reviews based on a fully automatic analysis of their textual content. This is akin to guessing, based on an analysis of the textual content of the review, the score the reviewer herself would attribute to the product. This problem arises from the fact that, while some online product reviews (especially those to be found in specialized product review sites such as Epinions.com[3], Amazon[4], Ratingz.net[5], or Rotten Tomatoes[6]) consist of a textual evaluation of the product *and* a score expressed on some ordered scale of values, many other reviews (especially those to be found in newsgroups, blogs, and other venues for spontaneous discussion) contain a textual evaluation only, with no score attached. These latter reviews are difficult for an automated system to manage, especially when a comparison among them is needed in order to determine, based on the

[2] See also: Andrew Lipsman, *Online Consumer-Generated Reviews Have Significant Impact on Offline Purchase Behavior*, `http://www.comscore.com/Press_Events/Press_Releases/2007/11/Online_Consumer_Reviews_Impact_Offline_Purchasing_Behavior`, November 29, 2007.

[3] `http://www.ratingz.net/`

[4] `http://www.amazon.com/`

[5] `http://www.epinions.com/`

[6] `http://www.rottentomatoes.com/`

reviews alone, whether product x is considered by the reviewers to be better than product y, or in order to identify the best perceived product in the lot. The availability of tools capable of interpreting a text-only product review and scoring it according to how positive the review is, is thus of the utmost importance.

In particular, our system addresses the problem of rating a review when the value to be attached to it must range on an *ordinal* (i.e., discrete) scale. This scale may be in the form either of an ordered set of *numerical* values (e.g., one to five "stars"), or of an ordered set of *non-numerical* labels (e.g., Disastrous, Poor, Good, VeryGood, Excellent). In mathematics, this rating task is usually called *ordinal regression*, or *ordinal classification*[7]. The difference between numerical and non-numerical values is inessential to our purposes, since we assume a scale of non-numerical labels to be easily mappable onto one of numerical values[8]. Since rating a product according to a number of "stars" is commonplace for many types of products (including movies, records, wines, hotels, etc.), hereafter we will assume this to be the rating model. We have accordingly called our software system StarTrack.

As hinted above, the basic operation that StarTrack is capable of performing is "star-rating" (i.e., attributing a certain number of stars, e.g., from one to five, to) a product review based on a fully automatic analysis of its textual content. However, the usefulness of StarTrack is not limited to this. Based on this capability, StarTrack can compute the *average* rating obtained by a given product (as resulting from star-rating different reviews of the product written by different consumers) and *rank* all the products that fulfil a given set of constraints (e.g., all stereo amplifiers in the 500 to 800 Euro range; all horror movies released since 2006 to 2008 and produced in the US; etc.) according to the computed average star-rating.

This latter ability thus allows a user to rank a set of comparable products by average reviewer satisfaction; it goes by itself that this allows a researcher to monitor consumer attitudes towards a given product / service / brand easily, so that the company may then respond by revising its production and marketing strategies accordingly. How well does product x fare against competing products? How high is brand y positioned in the reviewers' opinions? Did product z soar in the ranking as a result of last month's massive advertising campaign?

The rest of the paper is organized as follows. Section 2 describes StarTrack, analyzing its underlying philosophy and describing its main functionality. Section 3 presents instead the results of a laboratory evaluation, in which we measure the ability of StarTrack at guessing the correct star-ratings of a set of manually rated

[7] Ordinal regression is intermediate between the task of *classification* (in which there is no order defined on the non-numerical labels) and *metric regression* (in which objects are rated by real numbers).

[8] This assumption entails a further assumption, i.e., that the "distances" between two subsequent non-numerical labels are always the same; i.e., is the conceptual distance between Poor and Good equal to the distance between VeryGood and Excellent? This assumption may or may not be satisfied, depending on the context; however, for the purpose of this paper we will ignore this issue.

reviews (Section 3.1) and at guessing their correct ranking (Section 3.2). Section 4 concludes, discussing the results obtained and their value for market research.

2 StarTrack: An automatic tool for making sense of product reviews

How does StarTrack work? StarTrack is a system built according to a sophisticated mathematical model, that has its roots in disciplines such as information retrieval, machine learning, and computational linguistics; it is outside the scope of this paper to describe this model in detail, and we leave it to the mathematically conscious reader to check [3, 4] for details. In this paper, only a high-level description will be given that mostly tries to appeal to intuition.

The most important feature of StarTrack is that it does not need complex rules, based on natural language processing or other, to be manually written by an expert in order to analyze the reviews in a given problem domain. Rather, StarTrack is based on a "learning" metaphor, according to which StarTrack learns, from a set of manually star-rated reviews, the characteristics a given review should have in order to be attributed a given number of stars. Therefore, StarTrack does not need to be programmed with explicit rating rules: it only needs to be "trained" to star-rate reviews through exposure to a representative set of correctly star-rated reviews (therefore called *training reviews*). This approach is thus conceptually akin to the one championed by the VCS system described by Macer et al. [14], in which a verbatim coding system learns, from properly coded verbatim answers, the characteristics a verbatim answer should have in order to be attributed a given code. The main differences between StarTrack and VCS are that

1. StarTrack analyzes product reviews, while VCS analyzes verbatim comments returned to open questions in a survey;
2. given a review, StarTrack needs to make an exhaustive choice from an *ordered* set of *several* scores, while VCS, given a verbatim answer and a code, needs to make an exhaustive choice from an *unordered* set of *two* scores (Yes – meaning that the code should be given - or No – meaning it should not).

As a result, StarTrack's job is no doubt more complex than that of VCS. Figure 1 illustrates the basic process according to which StarTrack works.

StarTrack can thus potentially work as a building block for other larger systems that implement more complex functionality. For instance, given a community Web site containing product reviews whose users only seldom rate their own reviews, StarTrack can be used in order to learn, from the few rated reviews, to rate the others. Given another community Web site containing only unrated product reviews, StarTrack can be used to learn, from rated reviews of different provenance, to rate the Web site's own reviews. Web sites can also be set up

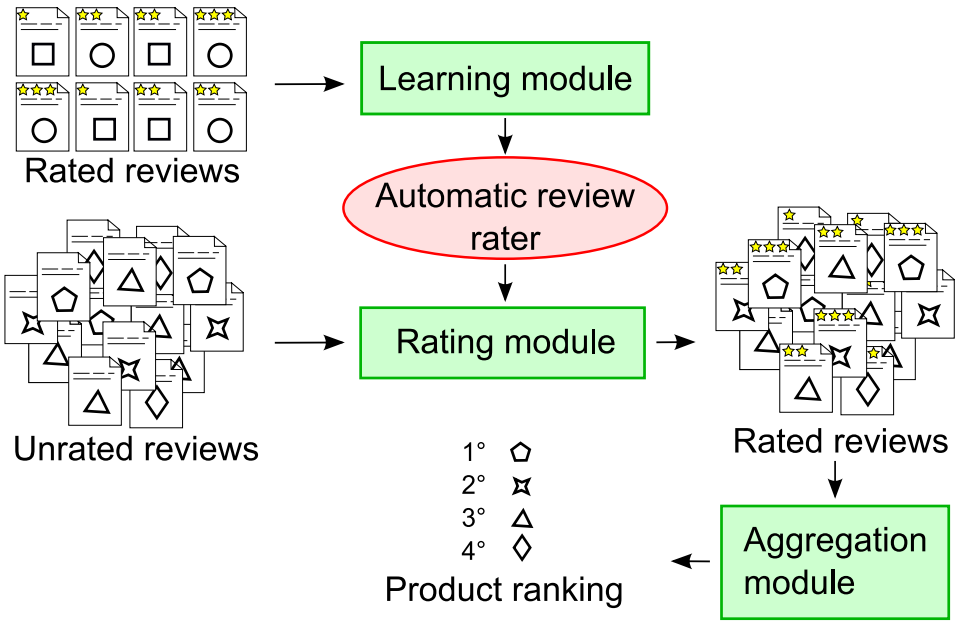

Fig. 1. The basic process according to which StarTrack works.

that act as "meta" review sites, i.e., as aggregators of the reviews contained in other Web sites (prominent examples are Metacritic[9] or the Movie Review Query Engine[10]); for such a meta-site it may be necessary to rate all the reviews according to its own ordered scale, irrespectively of the possibly different ordered scales used by the contributing sites.

The section that follows takes a slightly more detailed look at the internal workings of StarTrack; however, the less technically inclined reader may skip it without hindering her comprehension of the rest of the paper.

2.1 A quick look at the internals of StarTrack: machine learning and sentiment analysis

How does StarTrack learn from manually rated reviews? StarTrack is endowed with a sophisticated linguistic analysis module that, given a review, extracts *linguistic patterns* from it; in essence, this module transforms the review into an internal representation that consists of the set of patterns extracted from the review. Patterns may vary widely in nature. Some patterns are of a lexical nature (i.e., they are simple words), some are of a syntactic nature (e.g., noun phrases, verb phrases, clauses, entire sentences), some have a semantic nature (in which semantic categories have been substituted for actual words), and some are of yet different types. More importantly, once the patterns have been extracted from

[9] http://www.metacritic.com/
[10] http://www.mrqe.com/

the training reviews, StarTrack checks how each pattern is correlated to a given rating. For instance, if a given pattern mostly occurs in training reviews rated 4 Stars, with the possible exception of a few training reviews rated 3 Stars or 5 Stars, then it is deemed to be highly correlated with the 4 Stars rating. Such a pattern is useful, since when it is discovered in a yet unrated review, it will bring evidence that the rating of this review might be close to or exactly 4 Stars. Actually, an enormous number of patterns is encountered at least once in the set of training reviews; as a result, a "filtering" phase follows in which only the patterns that are highly correlated with a specific rating are retained (this is typically around 3% of the entire lot), while all others are discarded from consideration.

Upon encountering a yet unrated review, StarTrack extracts all possible patterns from it and checks, for each of them, if the pattern has been found to be correlated to a given rating, and how strong this correlation is. Therefore, all patterns extracted from the review constitute small pieces of evidence, each of which "votes" for a specific ratings; the rating for which the collective evidence is maximum is picked as the predicted rating for the review.

It is important to notice that "sentiment analysis" [15] plays a major role in the pattern extraction phase. Indeed, a product review management tool could hardly do without a sentiment analysis phase, since product reviews are mostly about reviewers *opinions*, which are heavily sentiment-laden. In StarTrack, sentiment analysis is mainly (although not exclusively) performed by mapping (via the use of a sentiment-specific lexical resource) patterns into ones in which the sentiment conveyed is made explicit. For instance, a noun phrase such as `helpful staff` (as to be found, say, in hotel reviews) is mapped into the pattern `[Virtue] [Positive] staff` (in which it is recorded that `helpful` is a term indicating an assessment of moral approval), and then into the simpler pattern `[Positive] staff`. Likewise, a noun phrase such as `friendly staff` is mapped into the pattern `[Emot] [Positive] staff` (in which it is recorded that `friendly` is a term indicating an assessment of emotional approval), and then into the simpler pattern `[Positive] staff`. Note that this process has generated patterns that reveal both the commonalities between the two expressions (`[Positive] staff`) and their differences (`[Virtue] [Positive] staff` and `[Emot] [Positive] staff`).

3 Evaluating the accuracy of StarTrack on real-world product review data

In this section we present the results of a laboratory evaluation of StarTrack we have conducted on datasets of real review data. Specifically, we have conducted this evaluation according to a *train-and-test* evaluation methodology, according to which a set of manually rated textual reviews is split into two non-overlapping sets:

Dataset	$\|Tr\|$	$\|Te\|$	1 Star	2 Stars	3 Stars	4 Stars	5 Stars	MAE^M
Amazon(MP3)-15071	9,998	5,073	16.76%	8.37%	9.33%	26.30%	39.24%	0.704
Amazon(MP3)-15071(s)	5,073	9,998	22.14%	8.69%	8.85%	22.85%	37.47%	0.736
TripAdvisor-15763	10,508	5,255	4.10%	7.16%	10.01%	34.69%	44.04%	0.722
TripAdvisor-15763(s)	5,255	10,508	4.55%	7.19%	9.97%	34.84%	43.40%	0.750

Table 1. Main characteristics of the datasets used in this paper. The 2nd and 3rd columns indicate the number of training and test reviews in the dataset, respectively. Columns from 4th to 8th indicate the fraction of training reviews that have a given number of "stars". The 9th column indicates the accuracy (measured by MAE^M) with which StarTrack has rated the test reviews in the dataset.

1. A *training set* of reviews, from which StarTrack learns to rate reviews. It is by analyzing the reviews in the training set that StarTrack learns the characteristics that a yet unrated review should have in order to have a certain rating. In other words, by analyzing the training reviews StarTrack builds an internal "mental model" of what it takes for a review to have a certain star-rating. This is called the *training phase* of StarTrack.
2. A *test set* of reviews, on which the ability of StarTrack at correctly guessing the star-ratings of textual reviews, and at correctly ranking products based on their guessed star-ratings, is tested. Specifically, the star-rating manually attached to the test reviews are assumed correct (as is the product ranking deriving from them), and hidden from StarTrack. This latter tries to guess both the correct star-ratings and the correct product ranking, based on the "mental model" acquired during the training phase. Evaluation consists in checking how closely the true and the predicted star-ratings / product rankings match. Here, the exact meaning of "close match" is mathematically specified by an *evaluation function*; two different such functions need to be employed, one for evaluating the correctness of star-ratings, and one for evaluating the correctness of product ranking.

As the first dataset for conducting our experiments we have chosen a set (that we here call Amazon(MP3)-15071) consisting of 15,071 reviews of MP3 players from the Amazon Web site (see Table 1, 1st row). Amazon(MP3)-15071 is actually a small subset of the dataset (consisting of more than 5 million reviews from the Amazon site) originally crawled by Jindal and Liu for spam review detection purposes [11], and consists of *all* the reviews of MP3 players contained in it. We have randomly picked 9,998 reviews to be used for training, and use the remaining 5,073 reviews for test. Altogether, the reviewed products are about 1,102 (on average, this means 13.68 reviews per product); 295 products have a single review while one product has 298 reviews.

The second dataset we use is the TripAdvisor-15763 dataset that we have built ourselves in a previous work [4], consisting of 15,763 hotel reviews from the

TripAdvisor Web site (see Table 1, 3rd row)[11]. We use the same split between training and test reviews of [4], resulting in 10,508 reviews used for training and 5,255 used for test. Altogether, the reviewed hotels are 323 (on average, this means 48.80 reviews per hotel); 6 hotels have a single review while one hotel has 368 reviews.

Both datasets consist of reviews scored on a scale from 1 Star to 5 Stars (scores with "half stars" are not allowed) and, as clear from Table 1, are highly imbalanced, with positive and very positive reviews by far outnumbering mild and negative reviews (this is especially true for TripAdvisor-15763). Both datasets contain a large set of test reviews, and the conclusions drawn from them can thus be considered fairly reliable.

3.1 Evaluating the ability of StarTrack at rating product reviews

We evaluate the ability of StarTrack at correctly predicting the star-rating of a product review by a mathematical measure called *macroaveraged mean absolute error* (noted MAE^M); this measure, which is more fully discussed in [2], is presented here only briefly.

Essentially, a "good" software system for star-rating reviews is a system that, as often as possible, guesses the correct rating of a review either exactly *or approximately*. This means that, if a given review's true rating is 5 Stars, predicting that its rating is 4 Stars is a better guess than predicting that its rating is 2 Stars. In other words, evaluating such a system should take into consideration the numerical distance between the true and the predicted rating. This distance is called *absolute error*; for instance, predicting 2 Stars when the true rating is 5 Stars incurs into an absolute error of 3, and predicting 5 Stars when the true rating is 2 Stars also incurs into an absolute error of 3 (that is, overrating and underrating are equally penalized). *Mean* absolute error (MAE) refers to the fact that absolute error is computed for all reviews in the test set that have a certain true rating (say, 2 Stars) and the mean of the absolute errors is computed. *Macroaveraged* MAE refers to the fact that, for each possible rating (e.g., 1 Star, 2 Stars, etc.) the corresponding mean is computed as described above, and the average of these means is computed to yield the final value.

MAE^M presents a global view of how accurate a system for guessing star-ratings is. Lower values are better, and the perfect system has $MAE^M = 0$. However, such a result is not attainable in practice, since the star-rating a hypothetical human annotator would attribute to a given textual review is highly subjective; this is just another facet of the well-known phenomenon of *inter-rater (dis)agreement* (see e.g., [13, pp. 219–250]).

In our experiments on Amazon(MP3)-15071 StarTrack obtained a MAE^M result of 0.704. In other words this means that, for an average review, StarTrack's

[11] The dataset is available for download from `http://patty.isti.cnr.it/~baccianella/reviewdata/`

predicted rating is 0.704 stars away from the true rating of the review. On the analogous experiment on TripAdvisor-15763, StarTrack obtained a MAE^M result of 0.722. While there are certainly margins of improvement, these results are noteworthy, and for several reasons:

1. Because analyzing product reviews is a hard task, since the language used by reviewers is often ungrammatical, colloquial, and ridden with typos and abbreviations.
2. Because *rating* product reviews is a hard task. Guessing the number of stars that the reviewer has attributed (or would attribute) to it would be difficult also for a human coder. It is not clear at all that, given a set of test reviews, the human coder would obtain a much better level of MAE^M when guessing the true star-ratings of these reviews. One reason for this difficulty is that different reviewers may use very similar language to express different ratings, depending on how conservative or liberal they are in their ratings; two reviewers writing approximately the same rebuttal of a given product might rate it 1 Star and 2 Stars, respectively.
3. Because the average accuracy of the system is heavily influenced by ratings (such as 2 Stars and 3 Stars) that are infrequent, and as such have few training reviews. In fact, each of the five different ratings counts the same (by design – see [2]) when computing MAE^M.

Table 3.1 provides another look at the same results, in the form of *contingency tables* which display, for each pair of ratings r_1 and r_2, how many reviews whose true rating is r_1 have been misrated as r_2. Table 3.1 shows that StarTrack performs quite well, as witnessed by the fact that high numbers of reviews tend to be concentrated on the diagonal (which represents perfectly correct decisions - more than half of the total number of test reviews in each dataset are in this category) or in the vicinity of the diagonal (which represents venial errors which any human coder trying to manually rate the review might potentially make), and by the fact that the cells far away from the diagonal (which represent blatant mistakes) are very scarcely populated.

Finally, one may wonder how much computer time it takes to run StarTrack on these problem sizes. In this experiment StarTrack required 21 min 29 sec for the training phase (an average of 0.12 sec per training review) and 12 min 33 sec for the rating phase (an average of 0.15 sec per test review). This latter figure means that, once trained, StarTrack is capable of rating reviews at a rate of approximately 24,000 reviews per hour, which allows it to easily tackle large or very large rating jobs. The experiments above were run on a standard consumer PC, equipped with an Intel Centrino Duo 2×2Ghz processor and 2GB RAM.

3.1.1 Rating reviews when training data are scarce In order to assess how sensitive StarTrack is to the number of training examples, for both datasets

True ratings	Predicted ratings: 1 Star	2 Stars	3 Stars	4 Stars	5 Stars	**Totals**
1 Star	544 (10.72%)	308 (6.07%)	217 (4.28%)	49 (0.97%)	5 (0.10%)	1123 (22.14%)
2 Stars	62 (1.22%)	206 (4.06%)	117 (2.31%)	49 (0.96%)	7 (0.14%)	441 (8.69%)
3 Stars	15 (0.30%)	77 (1.52%)	195 (3.84%)	135 (2.66%)	27 (0.53%)	449 (8.85%)
4 Stars	4 (0.08%)	34 (0.67%)	208 (4.10%)	556 (10.80%)	360 (7.10%)	1159 (22.75%)
5 Stars	2 (0.03%)	33 (0.65%)	146 (2.88%)	671 (13.23%)	1049 (20.68%)	1901 (37.47%)
Totals	627 (12.35%)	658 (12.97%)	883 (17.41%)	1460 (28.62%)	1448 (28.55%)	5073 (100.00%)

True ratings	Predicted ratings: 1 Star	2 Stars	3 Stars	4 Stars	5 Stars	**Totals**
1 Star	41 (0.78%)	14 (0.27%)	2 (0.04%)	0 (0.00%)	0 (0.00%)	57 (1.08%)
2 Stars	94 (1.79%)	103 (1.96%)	37 (0.70%)	9 (0.17%)	0 (0.00%)	243 (4.62%)
3 Stars	94 (1.79%)	186 (3.54%)	237 (4.51%)	171 (3.25%)	26 (0.49%)	714 (13.59%)
4 Stars	9 (0.17%)	73 (1.39%)	236 (4.49%)	1231 (23.43%)	811 (15.43%)	2360 (44.91%)
5 Stars	1 (0.02%)	2 (0.04%)	12 (0.23%)	420 (7.99%)	1446 (27.51%)	1881 (35.79%)
Totals	239 (4.55%)	378 (7.19%)	524 (9.97%)	1831 (34.84%)	2283 (43.44%)	5255 (100.00%)

Table 2. Contingency table for the experiments on the Amazon(MP3)-15071 dataset (top table) and on the TripAdvisor-15763 dataset (bottom table). Each cell contains the number and percentage of reviews with the given true rating which obtained the given predicted rating. Cells on the diagonal (white) represent perfectly correct decisions; cells near the diagonal (light grey) represent less serious mistakes while cells faraway from the diagonal (dark grey) represent more blatant mistakes.

we performed another experiment by switching the roles of training and test set: the reviews that belonged to the training set were put into the test set, and viceversa. The resulting datasets (here called Amazon(MP3)-15071(s) and TripAdvisor-15763(s), where "(s)" stands for "switched" – see Table 1, 2nd and 4th rows) have a much higher number of test reviews than the original datasets (which lends higher statistical value to the results), and a much lower number of training reviews (which gives indications as to the ability of StarTrack to perform well even when training reviews are scarce).

In these experiments (see Table 1) we obtained MAE^M results of 0.736 for Amazon(MP3)-15071(s) and 0.750 for TripAdvisor-15763(s), only marginally worse than the 0.704 and 0.722 results obtained on the original datasets. This shows that StarTrack may perform well even when training reviews do not abund.

3.2 Evaluating the ability of StarTrack at ranking products based on their reviews

We now move to evaluating the ability of StarTrack at correctly predicting the ranking of a set of products, based on the star-ratings it has attributed itself to their reviews. Generating such a ranking is accomplished by (a) taking all the reviews pertaining to a given product, (ii) averaging their (StarTrack-attributed) ratings, and (iii) ranking the products in descending order of their average ratings.

It could be objected that this evaluation exercise simply duplicates that of Section 3.1, and adds nothing to our understanding of StarTrack. Actually,

this is not true, since a software tool x might be better than another software tool y at guessing star-ratings but the opposite might be true for product ranking. To see this, assume there are three reviews A, B and C, with the (true) ratings ((A,3),(B,2),(C,1)); assume also that software tool x has returned the prediction ((A,3),(B,1),(C,2)) while software tool y has returned prediction ((A,5),(B,4),(C,3)). Tool x is obviously better than y in terms of the star-ratings they have predicted since, assuming that possible ratings range between 1 Star and 5 Stars, x has MAE^M equal to 0.4 while y has MAE^M equal to 1.2. However, y is better than x in terms of the product rankings they have predicted, since y has perfectly guessed the true ranking while x has not. So, evaluating the ability at star-rating and evaluating the ability at ranking are two independently interesting exercises, although related.

We evaluate the ability of StarTrack to correctly rank a set of products by a mathematical measure called *(normalized) Kendall distance with penalization* 0.5 (noted $K^{0.5}$). This measure, a variant of the well known "Kendall tau rank correlation coefficient" [12], is presented here only briefly; see e.g., [1] for a detailed discussion. $K^{0.5}$ measures the degree to which a predicted ranking of n objects coincides with the true ranking of these objects. Smaller values of $K^{0.5}$ are better, since $K^{0.5}$ returns 0 when the two rankings coincide and 1 when they are the reverse of each other. Essentially, $K^{0.5}$ returns a value proportional to the number of swaps of two objects that are needed to convert the predicted ranking into the true ranking. $K^{0.5}$ also caters for "ties" (i.e., objects that are tied either in the predicted or in the true ranking), discarding from consideration ties in the true ranking and penalizing the predicted ranking for tying two objects that are not tied in the true ranking.

It should be observed, however, that in our case ties are inevitably going to be in high numbers in both the true and the predicted ranking, given that in our datasets there are only five possible ratings and many, many more products. However, should we restrict our analysis to the products that have at least two test reviews, ties would be more infrequent, since there would now be nine values (including also values such as 1.5, 2.5, 3.5, 4.5) across which the average ratings of the various products would be distributed. And should we restrict our analysis to the products that have at least three, or four, or more, test reviews, ties would be even more infrequent. As a result, we have computed $K^{0.5}$ by restricting our analysis to the products that have at least x test reviews, for all values of x between 1 and 6 . The results of this computation are displayed in Table 3.

For instance, for $x = 1$ (i.e., the full set of 785 products is considered) on Amazon(MP3)-15071 StarTrack obtained a $K^{0.5}$ result of 0.136. Given that $K^{0.5}$ ranges between 0 (best) and 1 (worst), this indicates that StarTrack does a very good job at correctly ranking products based on how favourably they have been reviewed. Switching to Amazon(MP3)-15071(s) instead produced a notable deterioration, since $K^{0.5}$ rose to 0.182; this means that the ability of StarTrack at

Dataset	1	2	3	4	5	6
Amazon(MP3)-15071	0.136 (785)	0.053 (495)	0.031 (380)	0.019 (304)	0.013 (261)	0.008 (220)
Amazon(MP3)-15071(s)	0.182 (964)	0.093 (693)	0.055 (546)	0.036 (452)	0.029 (401)	0.020 (344)
TripAdvisor-15763	0.200 (315)	0.169 (294)	0.139 (271)	0.119 (251)	0.102 (233)	0.086 (215)
TripAdvisor-15763(s)	0.157 (321)	0.154 (314)	0.146 (304)	0.131 (289)	0.117 (274)	0.112 (268)

Table 3. Values of K^p when computed by restricting the ranking to the products that have at least x reviews, for all values of x between 1 and 6 (1 means that all products are considered). Each cell also indicates in parentheses how many products indeed satisfy the corresponding constraint.

ranking products is more sensitive to the number of available training examples than StarTrack's ability at rating reviews is.

However, Table 3 shows that results improve dramatically when x increases; for instance, if we restrict out attention to the products that have at least five test reviews, $K^{0.5}$ is equal to 0.013 for the 261 products of the original dataset and 0.029 for the 401 products of the switched dataset. Given that perfect ranking performance is denoted by $K^{0.5} = 0$, these values indicate *almost* perfect ranking performance.

The reason why StarTrack excels at ranking products that have many test reviews is that the star-ratings that reviewers assign do not always faithfully mirror what the reviewers have written in their reviews. For example, one reviewer may write a review that reads 2 Stars all the way, and instead rate it 1 Star. When there is one single test review for a given product, StarTrack may fall victim of these "mismatches" between the text of reviews and their star-ratings: in fact $K^{0.5}$ takes the ratings attributed by reviewers at face value, and penalizes StarTrack for not complying with them perfectly. When there are two or more reviews for the same product, instead, there is a smaller probability that *all* these reviews suffer from this problem: while the rating attached to the occasional review may be excessively high, the ratings of other reviews of the same product may be reasonable, and the rating of yet another review of the same product may instead be excessively low, thus compensating for the first. In other words, the high number of reviews that a given product has tends to reduce the influence of occasional "outliers"on StarTrack, and allows it to perform more correctly.

The results from TripAdvisor-15763 essentially confirm the observations above. However, note that in this case the improvements obtained when x increases are less dramatic, for the simple reason that in TripAdvisor-15763 most hotels have already many reviews anyway; see Table 3, rows 3 and 4 for details.

4 Conclusions

The controlled experiments we have presented show that StarTrack delivers consistently good accuracy across two very different domains (MP3 players and hotel rooms), and with no reprogramming needed for moving from one to the other.

This shows that StarTrack can be ported across a variety of domains and situations, and with no additional costs involved apart from those of obtaining the training reviews. We think this is a noteworthy accomplishment because rating product reviews is a hard task, due to the inherent subjectivity of the rating task and to the often ungrammatical and colloquial nature of the language used in these reviews.

The fact that hundreds of thousands of reviews can be rated in a few hours' computing time shows also that massive data sets can be analyzed, with all the ensuing benefits in terms of reliability of the conclusions obtained.

While the accuracy of StarTrack at star-rating product reviews (as witnessed by the MAE^M results) can be considered pretty good, its accuracy at ranking the products (as witnessed by the $K^{0.5}$ results) based on the average of the star-ratings that StarTrack has attributed to them, is no less than excellent. This is especially evident when products that have two or more reviews are ranked. This shows that StarTrack can reliably be used to detect where a given product or brand stands relative to a competitor, or relative to the rest of the pack.

Acknowledgements

We thank Nitin Jindal and Bing Liu for kindly sharing with us their "Amazon" dataset.

References

1. Fabio Aiolli, Riccardo Cardin, Fabrizio Sebastiani, and Alessandro Sperduti. Preferential text classification: Learning algorithms and evaluation measures. *Information Retrieval*, 12(5):559–580, 2009.
2. Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. Evaluation measures for ordinal text classification. In *Proceedings of the 9th IEEE International Conference on Intelligent Systems Design and Applications (ISDA'09)*, Pisa, IT, 2009.
3. Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. Feature selection for ordinal text classification. Technical Report 2009-TR-036, Istituto di Scienza e Tecnologie dell'Informazione, Consiglio Nazionale delle Ricerche, Pisa, IT, 2009.
4. Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. Multi-facet rating of product reviews. In *Proceedings of the 31st European Conference on Information Retrieval (ECIR'09)*, pages 461–472, Toulouse, FR, 2009.
5. Yubo Chen and Jinhong Xie. Third-party product review and firm marketing strategy. *Marketing Science*, 23(2):218—240, 2005.
6. Yubo Chen and Jinhong Xie. Online consumer review: Word-of-mouth as a new element of marketing communication mix. *Management Science*, 54(3):477—491, 2008.
7. Judith A. Chevalier and Dina Mayzlin. The effect of word of mouth on sales: Online book reviews. *Journal of Marketing Research*, 43(3):345–354, 2006.
8. Chrysanthos Dellarocas, Xiaoquan Zhang, and Neveen F. Awad. Exploring the value of online product reviews in forecasting sales: The case of motion pictures. *Journal of Interactive Marketing*, 21(4):23–45, 2007.
9. Guodong Gao, Bin Gu, and Mingfeng Lin. The dynamics of online consumer reviews. In *Proceedings of the Workshop on Information Systems and Economics (WISE'06)*, Evanston, US, 2006.

10. Ulrike Gretzel and Kyung Yan Yoo. Use and impact of online travel review. In *Proceedings of the 2008 International Conference on Information and Communication Technologies in Tourism*, pages 35–46, Innsbruck, AT, 2008.
11. Nitin Jindal and Bing Liu. Review spam detection. In *Proceedings of the 16th International Conference on the World Wide Web (WWW'07)*, pages 1189–1190, Banff, CA, 2007.
12. Maurice Kendall. A new measure of rank correlation. *Biometrika*, 30:81–89, 1938.
13. Klaus Krippendorff. *Content analysis: An introduction to its methodology.* Sage, Thousand Oaks, US, 2004.
14. Tim Macer, Mark Pearson, and Fabrizio Sebastiani. Cracking the code: What customers say, in their own words. In *Proceedings of the 50th Annual Conference of the Market Research Society (MRS'07)*, Brighton, UK, 2007.
15. Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1/2):1–135, 2008.
16. Sylvain Sénécal and Jacques Nantel. The influence of online product recommendations on consumers' online choices. *Journal of Retailing*, 80:159–169, 2004.