PROVA FINALE

Pros and cons on controvertial issues through crowdsourcing

candidato
Fela Winkelmolen

relatore
Marina Ribaudo

UNIVERSITÀ DEGLI STUDI DI GENOVA

Marzo 2013

Contents

Chapter 1

Ranking by vote

1.1 Bayesian average

The objective when ranking elements is to show the most relevant and interesting information at the top. This is done by using the votes as an indication of how interesting the information was for past users, and assuming that what holds for past users is likely to hold for future users[1].

The simplest and most widespread ranking schemes suffer from well known weaknesses ([5]). Zhang et al. propose two axioms that should reasonably be respected by a good ranking algorithm: first, up and downvotes should respectively (strictly) increase and (strictly) decrease the score, and second, if a vote of the same type gets added its weight should be strictly smaller than the previous vote. They find three ranking algorithm consistent with these axioms, their most general result uses what they call Dirichlet prior smoothing. Let $n_{\uparrow}$and $n_{\downarrow}$ be the number of up and down-votes, they will be ranked by the following score

$$\frac{n_{\uparrow} + \mu p_{\uparrow}}{n_{\uparrow} + n_{\downarrow} + \mu}$$

where $p_{\uparrow}\epsilon(0,1)$ and μ are fixed parameters that will determine how the ranking behaves in the case of few available votes. Intuitively, as the total number of votes increases, the score assigned by the algorithm will tend towards the ratio *positive votes/total votes*, if there are few votes it will tend towards the fixed "default" score $p_{\uparrow}$; the higher μ the higher the weight of

[1] When different users have very different opinions this might not hold, an example is on controversial topic, where polarization could make opinions diverge widely. Although this is exactly the setting of this thesis, the main component of this polarization is removed by raking the claims of the two main positions separately.

$p_\uparrow$. To better understand this, the score could be rewritten as the weighted average between $p = positive\,votes/total\,votes$ and the user defined $p_\uparrow$

$$\frac{w_1 p + w_2 p_\uparrow}{w_1 + w_2}$$

where the weight w_1 is equal to the number of total votes $n_\uparrow + n_\downarrow$, and w_2 is equal to μ, the weight given to the prior information.

Here is an example of the result on random imaginary votes, the first column shows the resulting score, followed by number of upvotes and the number of downvotes:

97	+119	−3
88	+12	−1
85	+85	−14
82	+7	−1
80	+2	−0
68	+11	−5
67	+0	−0
57	+50	−39
48	+13	−15
38	+1	−4
23	+7	−29
10	+0	−18

There is a generalization of this algorithm for the case where the votes can assume more than two values, this version currently in use by a number of websites. IMDB, for example, uses it for its top 250 list and calls it true bayesian estimate. It has also been variously referred to online as bayesian average and bayesian ranking, but without ever giving much explanation. [3], although without giving the above formula, sketches the basic reasoning from which the result can be derived.

The model rests on the assumption that there is a certain population of potential voting users, and that we would like to estimate the percentage q of them that, if given the chance, would vote positively. A detailed derivation of the final formula is given in appendix A.

I will use bayesian average to denote the score calculated this way, and bayesian ranking to denote the ranking process that uses that score.

1.2 Limitations and assumptions

"All models are wrong, but some are useful" – George E. P. Box

Every model is just an approximation of reality, but it is hoped to capture the main interactions that are of interest. Very often approximations or assumptions are made without giving them much thought. I believe that a honest analyses should give them some space, to be able to better assess how well and under which condition a model might is warranted. This is specially important in the case where where there are no obvious benchmarks to assess quality. I will also note that assuming that the result of a benchmark will correlate to usefulness in real life is in itself a model.

1.2.1 Justification of the model

The probabilistic setting assumes there is a population of voters from which a number of individuals selected at random cast their vote. To start to understand the possible weaknesses of the model it is worth asking how much this is an accurate description of what happens in practice.

There are at least two ways in which the independence assumption might fail. First, the users that visit the webpage might not really be randomly chosen from the population of interest, for a number of reasons, for example if somebody likes a comment very much he might refer other people to it, which might have similar opinions. Second, users might be influenced by the current score[2].

Another implicit assumption which subtly breaks, is that the total number of votes has been assumed as being given, where in fact it is unlikely to be totally independent of q. The score will influence the position that an element will have on the page, which in turn influences the number of votes.

Notwithstanding all these assumptions, most of which are unlikely to have a big effect overall, if the only information available is the number of upvotes and downvotes, estimating q is the best that can be done, and the bayesian approach yield optimal results *if* the prior and loss function are chosen correctly.

1.2.2 Prior

When applying bayesian inference a few additional assumptions have been made that merit further consideration.

[2]The effect of peer pressure on judgment, also known as Bandwagon effect, has been extensively documented, see for example [1]

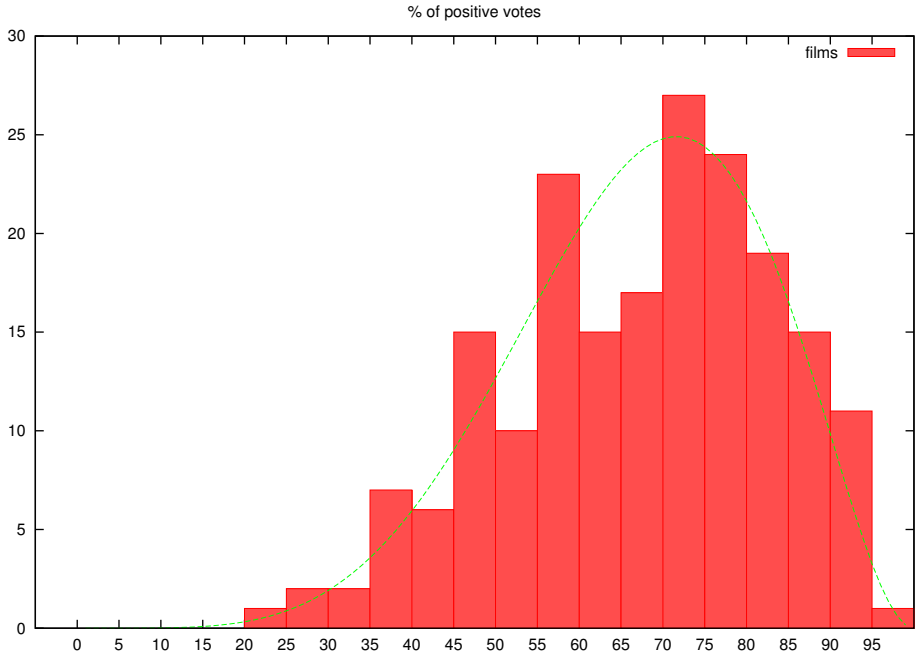

Figure 1.1: The graph shows, in red, a number of random films grouped by their Rotten Tomatoes score, which is equal to the percentage of positive user ratings. The green line shows an ideal distribution, assuming a beta distributed prior.

When calculating the posterior probability the prior has been assumed to be known and be equal to the beta distribution, without giving much justification other than algebraical convenience. The beta distribution does however have a number of desirable properties. It is worth noting that the assumption doesn't force a single prior, but rather allows for a family of distribution functions, by varying $p\uparrow$ and μ. $p_\uparrow$ is equal to the mean of the distribution, and will be the "default" value in the case when no information is available, the obvious way to chose it is to set it equal to the mean of all historical votes in the application. μ is inversely proportional to the variance, and indicates how much the score tent to vary in historical data. The higher μ the more votes will be required before the score will differ significantly from $p_\uparrow$.

Too see how realistic such a prior is, I analyzed the distribution in a real examples, to do this I chose the the Rotten Tomatoes user scores. Rotten Tomatoes is a website containing information and reviews about movies; the users can score the movies and the final score will be equal to the percentage of positive votes. Given the large number of votes, from many hundreds to many thousands in the case of the movies I've analyzed, this percentage

can be assumed a good approximation of q. The films analyzed have been collected from available lists such as movies currently playing in the theaters and recent DVD releases. Lists such as best and worst movies have not been used as obviously biased for our use. The API doesn't allow to directly access the number of user votes, so the number of critics reviews has been used to exclude obscure movies which might not have enough votes to make the data reliable. This might add a slight bias, however the fact that the distribution is still approximately beta distributed is an indication of the good properties of this distribution. The result can be seen in Figure 1.1, the overlayed green line represents a beta distribution with parameters set to match the data.

1.2.3 Loss function

As the prior, the loss was chosen mainly for its convenient algebraic properties. The loss function is the cost of estimating that an object has a percentage of approval equal to q where, if more information had been available, we would discover that the real percentage is $\tilde{q}$. In our case this cost is assumed to be $(q - \tilde{q})^2$, which penalizes outliers; this is to say that having an error of 2Δ is assumed as bad as having 4 errors of magnitude Δ. Another option would have been the absolute error. Given that there is no strong reason to prefer either, the more tractable square error has been chosen.

A case could also be made for asymmetric loss functions, for example [3] proposes a loss function that tends to give greater cost to over-ranking as opposed to under-ranking, with the result that in the case of uncertainty the score will be lower. A similar effect, without the additional computational cost, could be obtained by setting $p_\uparrow$ to a lower value in the prior distribution. There is however no conclusive justification for being conservative in sorting comments, and having uncertain scores come quickly to the top might help counter the "fastest gun in the west" problem (see below 1.2.5).

1.2.4 Using additional information

The above model takes into account only the votes. There might be other information that could be useful for ranking. For example which authors voted and the number of people who have viewed an element without voting on it. How to best use views is analyzed in the next section, while reputation systems which look into who voted are described in the next chapter.

1.2.5 Fastest gun in the west problem

A know problem often arising when ranked comments by votes is that the oldest comments tend to have a much higher chance making it to the top, purely as a consequence of being there first. This is particularly true in the case of ranking by number of votes, or by the difference of upvotes and downvotes. Newer comments will start at the bottom and thus, other things being equal, will be much less likely to collect votes, making their difference in standing relative to the top only bigger as time passes.

This shows ranking in a different perspective, because it is not just important to show the most relevant information first, but also show the most relevant information to future users. There could be cases where it is not particularly likely that an object is relevant, but it is still advantageous to make it visible to attract votes and be able to show more accurate information in the future.

Any ranking that estimates the proportion of positive votes has the following property: given a large enough number of viewers a comment will eventually reach its real score. However the number of viewers, specially for new comment, is finite, furthermore at any given time there might be a big number of of recent comment that do not yet have much votes. This has been a reason for not using score algorithms that tend to underevaluate the score in case of high uncertainty.

1.3 Number of views

An interesting piece of information that hasn't been used is the number of users who have seen an object but did not find it interesting enough to vote on it. So the current algorithm would give the same score to an item viewed by 100 people all of which voted it up, and one viewed by 1000 people only 100 of which voted it up. The next section deals with estimating the number of views, but assuming they are known, how can they be incorporated into the algorithm?

Starting from the obvious consideration that a view is better than a downvote and worse than an upvote, the easiest way to generalize the bayesian average is to suppose a view equals to $v \epsilon [0, 1]$ upvotes and $1 - v$ downvotes. This could be formally justified by assuming that if forced to vote the fraction of non voters that would cast a positive vote is equal to v.

1.3.1 Estimation of number of views

Obviously the number of page views alone is not a good measure, when a page is loaded only a fraction of the content of a page is really viewed [4]. Furthermore not everybody is as likely to vote on content they view.

Therefore, on a given page, only users that voted on at least one item get counted as having viewed its content, and will get counted only once. This solves the statistical noise that would otherwise be caused by different people behaving differently. Elements further down are less likely to be viewed, so their view count should not be increased by as much. To be able to do this the number of views can assume non discreet values, for example 0.5 views is a valid number and equals to 50% chance of a view.

It is thus needed to have a a function that maps page position to probability of being viewed. This function can, aside from a multiplication factor, be approximated by the probability of a attracting a vote given the page position. This probability can be estimated by using the historical data in the whole application. Until this data is available a rough estimate will have to do. Lets suppose the probability of obtaining a vote at position i has been calculated and is equal to

$$f(i) = \frac{votes}{pageviews}$$

This can be normalized by setting the maximum value equal to 1

$$\tilde{f}(i) = \frac{f(i)}{\max\limits_{i} f(i)}$$

At this point if elements in position $i_1, i_2, ..., i_n$ get voted by a user, the views count of element at position i will be incremented by the value assumed by the above function, scaled such that the less visible item that has been voted on equals 1:

$$\frac{\widetilde{f}(i)}{\min\limits_{j=i_1,i_2,...,i_n} \widetilde{f}(j)}$$

A more complex algorithm to estimate views could also incorporate cursor movement, which has been shown to strongly correlate with eye movement[2].

1.4 Normalization

The bayesian average is normalized to the interval $[0, 10]$ before being displayed to the user with one digit after the decimal point. Having one digit

after the decimal point makes sure the user understand that the score isn't a simple sum of the up and downvotes.

Bibliography

[1] SE Asch. *Effects of group pressure upon the modification and distortion of judgments.* Carnegie Press, 1951.

[2] Jeff Huang, Ryen W. White, and Susan T. Dumais. No clicks, no problem: using cursor movements to understand and improve search. In Desney S. Tan, Saleema Amershi, Bo Begole, Wendy A. Kellogg, and Manas Tungare, editors, *CHI*, pages 1225–1234. ACM, 2011.

[3] Evan Miller. Bayesian Average Ratings. online article, 2012.

[4] J. Nielsen. How Users Read on the Web, 1997.

[5] Dell Zhang, Robert Mao, Haitao Li, and Joanne Mao. How to Count Thumb-Ups and Thumb-Downs: User-Rating Based Ranking of Items from an Axiomatic Perspective. In Giambattista Amati and Fabio Crestani, editors, *ICTIR*, volume 6931 of *Lecture Notes in Computer Science*, pages 238–249. Springer, 2011.

Appendix A

Here follows the proof that the Dirichlet Prior proposed by Zhang et al. ([5]) can be obtained by bayesian inference, under the assumption of a prior following the beta distribution and a minimum square error loss function. A formalization of the voting problem that uses bayesian reasoning has been sketched in a 2012 online article by Miller ([3]). Similar results have been obtained in the literature under the name additive smoothing.

Let q denote the proportion of the population that would give a positive vote[1], i.e. the value we want to estimate. Let the prior probability of q be known and be equal to the beta-distribution in the parameters α and β

$$p(q) = Beta(\alpha, \beta) = \frac{q^{\alpha-1}(1-q)^{\beta-1}}{B(\alpha, \beta)}$$

where the normalization factor B denotes the beta function.

Every vote is assumed to come from a randomly selected user, therefore every vote will have q probability of being positive and $1-q$ probability of being negative. The probability of having $n_\uparrow$ positive votes and $n_\downarrow$ negative votes after $n = n_\uparrow + n_\downarrow$independent trials is given by the probability of a binomial experiment

$$p(n_\uparrow, n_\downarrow | q) = \binom{n_\uparrow + n_\downarrow}{n_\uparrow} q^{n_\uparrow}(1-q)^{n_\downarrow}$$

The posterior probability of q can be calculated by Bayes' theorem

$$p(q|n_\uparrow, n_\downarrow) = \frac{p(n_\uparrow, n_\downarrow|q)p(q)}{\int p(n_\uparrow, n_\downarrow|q)p(q)dq}$$

The specific prior distribution has been chosen because it is the conjugate prior of a binomial experiment, meaning that the posterior probability of

[1]Here it has been assumed the population is equal to the users that would vote, which means we leave out the users that would vote neither up nor down.

Bayes' theorem can be written in term of elementary functions, condition which does not hold in general. More specifically, the posterior probability in the case of beta-distributed prior is known to be another beta-distribution, in the specific case

$$p(q|n_\uparrow, n_\downarrow) = Beta(\alpha + n_\uparrow, \beta + n_\downarrow)$$

At this point the usual approach is to choose q by minimizing a loss function, in our case we choose to minimize the mean square error (MSE), which can be shown to be equal to finding the expected value:

$$\min_y \mathrm{MSE}[x] = \min_y \int_0^1 p(x)(x-y)^2 dx$$

$$= \min_y \int_0^1 p(x)x^2 dx + y^2 \int_0^1 p(x)dx + 2y \int_0^1 p(x)x dx$$

$$= \min_y (y + 2y\mathrm{E}[x]) = \mathrm{E}[x]$$

in the case of our distribution $Beta(\alpha + n_\uparrow, \beta + n_\downarrow)$ the expected value is equal to

$$\frac{n_\uparrow + \alpha}{n_\uparrow + n_\downarrow + \alpha + \beta}$$

from which it will suffice to replace $\mu = \alpha + \beta$ and $p_\uparrow = \alpha/(\alpha + \beta)$ to obtain the formula used by Zhang et al.

$$\frac{n_\uparrow + p_\uparrow \mu}{n_\uparrow + n_\downarrow + \mu}$$