Time Dependant Trustworthiness: An Iterative Algorithm

University of New South Whales Sydney
Elliot Griffiths
z3332776
e.griffiths@unsw.edu.au

Abstract.

Exponential growth in e-commerce markets has continued throughout the decade and shows no signs of slowing. The potential of scalability combined with the reduced cost to consumers which e-commerce stores provide has proven to out way the convenience of brick and mortar alternatives. Without the ability to inspect items before purchasing, consumers heavily rely on the community census, leading to strong monetary motivations for owners to synthesise false or manipulate item ratings and reviews. In this paper I present an adapted algorithm which is robust to smart collusion attacks. The algorithm is based on building trust in users feedback dependent on temporal analysis and convergence to the group consensus. I have implemented and tested the iterative algorithm on several simulated situations with results demonstrating realistic ratings.

1 Introduction

Consumers are opting to obtain a growing number of services, products and recommendations through online marketplaces more than ever before. As such, in todays world virtually all businesses have an online presence of some description. Companies who do not offer online access are left at a severe disadvantage. This can be seen in all fields from media to e-commerce. Rates of cable-cutting are increasing [2] while the growth of e-commerce and online entertainment industry is booming [1].
With this rapid expansion, consumers are spoilt for choice with literally hundreds of thousands of e-commerce stores [3] each of which can offer a vast variety of items. The US Amazon[1] store alone has over 200 million products [4] resulting in extremely high levels of competition. Without the ability to see the product in person, it can be difficult for a potential buyer to judge the quality of the product or character of the service staff. In addition to the quality of the item, the consumer must also take a risk, hoping for a smooth and pleasant customer service experience which she has no way of knowing prior to parting with her money. To minimise these risks consumers can use the crowd consensus of the online community to set apart one product or service from another. This kind of data aggregation we call an online rating system. Business and store owners frequently implement an online rating system into their marketplace in the hopes of giving potential consumers the confidence they need to make the final purchase. Platforms which do no sell items or services also include rating system to bring additional value to their users, for example, internet giants Google Maps[2] and Facebook[3] provide a rating system for businesses.

1 . 1 Problem Statement

Here lies the major issue, as the online rating systems can play such a large role in consumer choice, they are inevitably directly linked to the profit of online businesses. Thus, coupled with the intense competition, it has been proven to motivate dishonest ratings being made with the sole interest of self gain [5, 6, 7]. Dishonest votes can come in the form of an individual giving a false positive rating to their own business or giving a false negative rating to a business in competition. With a large number of ratings, these kind of attacks are less impactful. However, if an organised group engage in the same activity it can be much more damaging to the trustworthiness for the overall final product rating score. This is what is know as collusion.

[1] amazon.com
[2] google.com/maps
[3] facebook.com

1 . 2 Existing Models

Addressing this issue of dishonest reviews has been attempted using graph problems to find clusters of dishonest reviews [8], using iterative filtering [9] and evaluating rating through voting [10, 11].

1 . 3 Proposed Adaptation

I have attempted to improve the accuracy of the first version of the The Rating Through Voting (RTV) algorithm [11] by including context data in the calculation. My adapted model follows the same method as RTV by first reducing the task to an election where users ratings are seen as 'votes' where they choose the most fitting option. For all possible options the algorithm iteratively calculates to what degree each users adheres to the community consensus and assigns a trustworthiness value to said user. Additionally the algorithm also takes into account the time each user submitted their vote for each item and adjusts the trustworthiness value.

The consensus vote is then recalculated base on the new trustworthiness values. The algorithm continues to iterate calculating new trustworthiness and community consensus values until it reaches a point where changes between rounds a smaller than a set variable. This results in the simultaneous evaluation of the trust level of each user and the community consensus vote (the final most likely rating of each item).

Later we will see for a user to build high trust they will need to submit votes early in the lifetime of the product while also agreeing with the other honest voters.

2 Time Dependant Trustworthiness

In this section I will extend the RTV algorithm [11] to take into account further context information of each vote.

2.1 Setup

Assume there are nv many voters $V = \{V_1 \,..\, V_{nv}\}$ each of whom vote on nl many lists $L = \{L_1 \,..\, L_{nl}\}$. Each list L_j contains ni many items I where, $L_j = \{I_1^j \,..\, I_{ni}^j\}$. In our implementation the items are analogous to star ratings and the lists are the products being reviewed. ie. The most popular item according to the algorithm will be the most likely rating for the product which is our list L.

For each vote cast by voters V_i $(i = 1..nv)$, on list L_j $(j = 1..nl)$ we store the number of time units passed between L's release date (the product) and vote cast time in table θ where $\theta_{i,j} \leq TP$.

Here TP is the maximum number of time units passed between the product release (list L) and the time of its latest submitted vote, item I. For each voter V_i we also keep track of her trustworthiness at each iteration $T_i^{(p)}$

2.2 Algorithm

Initialisation:

$\epsilon > 0$	Precision threshold,
$\alpha \geq 1$	Compliance discrimination parameter
$\beta > 0$	Cast time discrimination parameter

$T_i^{(0)} = 1,$ Initial trustworthiness for $i = 1..nv$

$$R_{item,list}^{(0)} = \sum_{V_i : V_i \rightarrow item,list}^{V_{nv}} (T_{V_i}^{(0)})^{\alpha}$$

where R is the ranking of each item on each list, calculated by summing up all trustworthiness values ($\wedge\alpha$) of voter V_i if she voted for the item on list

$$\rho^{(0)}_{item,list} = \frac{R^{(0)}_{item,list}}{\sqrt{\sum_{i=1}^{ni} (R^{(0)}_{i,list})^2}}$$

We then normalise the R value to a value ρ between 0 and 1 by dividing the items scores R by the sum in quadrature of all the scores in that particular list

We now iteratively repeat the following steps updating the trustworthiness and ranking scores until the change in the magnitude of vector $\overrightarrow{\rho} < \epsilon$

$$T^{(p+1)}_{V_i} = \sum_{list=1}^{nl} \frac{\rho^{(p-1)}_{V_i \to item,list}}{(\theta_{list,V_i})^{\beta}}$$

To update the trustworthiness scores of each voter we take the sum of the normalised ranking of each vote they made (over all lists) divided by the 'age' of their vote cast in time units raised to the power of β.

$$R^{(p+1)}_{item,list} = \sum_{V_i : V_i \to item,list}^{V_{nv}} (T^{(p+1)}_{V_i})^{\alpha}$$

$$\rho^{(p+1)}_{item,list} = \frac{R^{(p+1)}_{item,list}}{\sqrt{\sum_{i=1}^{ni} (R^{(p+1)}_{i,list})^2}}$$

Continue until:
$||\overrightarrow{\rho}^{(p+1)} - \overrightarrow{\rho}^{(p)}||^2 < \epsilon$

2 . 3 Motivation

The intuition follows the RTV algorithm [11]. I aim to calculate the rankings of items by judging the voters themselves based on their votes. To influence the final rankings a voter must build high trustworthiness. Trustworthiness is built by voting early and consistently with the consensus of the honest community members.
The idea of proportionally building trust in earlier votes stems from the motivations of colluding attackers. This follows in two parts:

(1) Attackers will aim to target competitors who threaten their profits. For a competitor to impact revenue or show signs of rapid market share growth it follows that they must already have achieved positive feedback from the community. Thus, once they appear on the 'radar' of the attackers they will already have a rating which can not be impacted by a large number of late coming new attacking votes.

(2) Attackers who are aiming to build trust in the community (preparing for a new collusion attack) can not do so by agreeing with community consensus on already determined ratings. The only way to build trust is to submit an accurate review early before the community has come to a consensus. This is only consistency possible by purchasing a product or service and submitting an honest review early in the produces lifespan. This is not possible on a large scale with automated vote bots.

2 . 4 Convergence

The requirement of convergence is conserved from the RTV algorithm [11] to the adaptation, Time Dependant Trustworthiness (TDT) algorithm.
The TDT algorithm will terminate for all twice differentiable strictly increasing functions. The division of the trustworthiness function by a constant will not affect this proof as the fraction is still differentiable;

$$\frac{\partial F(\overrightarrow{\rho})}{\partial \rho_{li}} = (\alpha + 1) \sum_{r:r \to li} T_r^{\alpha}(\overrightarrow{\rho})$$

For a rigorous convergence proof of RTV algorithm see [11].

3 Experimentation

In this section I will first show how a relatively small group of intelligent attackers are able to manipulate the RTV algorithm. Then I will implement the TDT algorithm and compare the outcomes. Later I will look at the effect of adjusting the α and β parameters and attempt to break the algorithm with several other complex attacks.

3 . 1 Intelligent bot Attack: RTV vs TDT

RTV Setup:

An implantation of the Voting Through Rating algorithm [1] is attached in the supporting documents. (See file RTV_Intelligent_Attack.nb) This implementation is the work of Aleks Ignjatovic with an added intelligent bot attack. The original file can be found on the COMP4121 Advanced Algorithms - Session 2, 2017 Webpage[4].

In this test we have:

- 25 honest voters who agree that product 7 should receive a rating of 9/9.
- 10 intelligent attacking voters who aim to decrease the score of item 7 to a 1/9.
- 75 unintelligent attackers trying to change the score of 7 to 5/9.

Our 10 intelligent attackers all vote 1/9 on item 7 and on all remaining items they vote to match the community consensus. This is to show they agree with the community and thus must be trustworthy voters. In reality this would be easy to do with a bot which votes on many items on amazon with the same score as the individual item' s average rating.
Our 75 unintelligent attackers all vote 5/9 on item 7 and vote randomly on all remaining items to show they are active in the community.
Our 25 honest voters vote honestly on all items. From this we can see that the true score of item 7 should be 9/9.

[4] www.cse.unsw.edu.au/~cs4121/

RTV Results:

NOTE: All following matrixes represent items 1 to 7 on the vertical axis and star ratings 1 to 9 on the horizontal.

Iteration 0:

$$\begin{array}{ccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\left(\begin{matrix}
0.723175 & 0.451985 & 0.135595 & 0.112996 & 0.271191 & 0.225992 & 0.135595 & 0.158195 & 0.271191 \\
0.215365 & 0.172292 & 0.366121 & 0.172292 & 0.129219 & 0.796851 & 0.279975 & 0.150756 & 0.0861461 \\
0.169714 & 0.127286 & 0.127286 & 0.424285 & 0.784928 & 0.169714 & 0.0636428 & 0.254571 & 0.212143 \\
0.254238 & 0.1849 & 0.27735 & 0.27735 & 0.1849 & 0.231125 & 0.115563 & 0.762713 & 0.254238 \\
0.26295 & 0.310759 & 0.286855 & 0.717137 & 0.167332 & 0.239046 & 0.310759 & 0.167332 & 0.167332 \\
0.228934 & 0.366295 & 0.297615 & 0.73259 & 0.0457869 & 0.206041 & 0.160254 & 0.206041 & 0.274721 \\
0.125491 & 0. & 0. & 0. & 0.941184 & 0. & 0. & 0. & 0.313728
\end{matrix}\right)
\end{array}$$

After the first round we can see that the 75 unintelligent attacking voters have won. They caused the rating of the 7th item to be 5/9.

Iteration 12:

$$\begin{array}{ccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\left(\begin{matrix}
0.99551 & 0.0942886 & 0.00126143 & 0.000713382 & 0.0034558 & 0.00253583 & 0.00392271 & 0.00453724 & 0.00364661 \\
0.056125 & 0.000816285 & 0.126642 & 0.00332984 & 0.00113855 & 0.990306 & 0.00794177 & 0.00241365 & 0.00484201 \\
0.00714845 & 0.000849649 & 0.000587266 & 0.193916 & 0.980961 & 0.00490574 & 0.000756615 & 0.00511078 & 0.00311536 \\
0.0310004 & 0.00294882 & 0.00443858 & 0.00420888 & 0.00243891 & 0.00230629 & 0.00285326 & 0.997979 & 0.0548756 \\
0.0565176 & 0.0670505 & 0.0356098 & 0.995495 & 0.000490574 & 0.00313031 & 0.00379999 & 0.00284893 & 8.28569\times10^{-6} \\
0.00806661 & 0.259532 & 0.00193818 & 0.965653 & 0.000273736 & 0.00728339 & 0.00471051 & 0.00250527 & 0.00278573 \\
0.804965 & 0. & 0. & 0. & 0.0456377 & 0. & 0. & 0. & 0.591564
\end{matrix}\right)
\end{array}$$

After the 12 iterations the algorithm has terminated. We can see that the RTV algorithm has managed to reduce the trustworthiness of the 75 unintelligent attacking voters to a point where their attack on item 7 is negligible. However, the 10 intelligent attacking voters have managed to out weigh the 25 honest voters as item 7 has received a rating of 1 when it deserved a 9.

TDT Setup:

Here all voters keep the same voting strategy however we have increased the number of intelligent attacking voters from 10 to 75 to demonstrate the robustness of the algorithm.

In this test we have:

- 25 honest voters who agree that product 7 should receive a rating of 9/9.
- 75 intelligent attacking voters who aim to decrease the score of item 7 to a 1/9.
- 75 unintelligent attackers are trying to change the score of 7 to 5/9.

We need to generate cast times for each vote as the TDT algorithm takes into account these times.

Cast time generation:

- Honest voters: Randomised 1 - 10
- Intelligent attacking voters: Randomised 8 - 10
- Unintelligent attacking voters: Randomised 1 - 10

For the intelligent attacking voters they are randomised between 8 to 10 because in order to vote in line with the consensus of honest voters they need to know the outcome of all other votes thus they can only cast their votes after the majority of honest voters.

TDT Results:

An implantation of the Time Dependant Trustworthiness algorithm is attached in the supporting documents. (See file TDT_Intelligent_Attack.nb) This implementation has been adapted from the work of Aleks Ignjatovic with modified algorithm and an added intelligent bot attack. The original file can be found on the COMP4121 Advanced Algorithms - Session 2, 2017 Webpage[5].

[5] www.cse.unsw.edu.au/~cs4121/

Iteration 0:

$$\begin{array}{ccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\left(\begin{array}{c} 0.950333 \\ 0.139395 \\ 0.0697113 \\ 0.159182 \\ 0.204862 \\ 0.00967053 \\ 0.688247 \end{array}\right. &
\begin{array}{c} 0.180063 \\ 0.0796543 \\ 0.139423 \\ 0.0497445 \\ 0.143403 \\ 0.17407 \\ 0. \end{array} &
\begin{array}{c} 0.150053 \\ 0.0995678 \\ 0.0497938 \\ 0.139285 \\ 0.122917 \\ 0.0967053 \\ 0. \end{array} &
\begin{array}{c} 0.100035 \\ 0.109525 \\ 0.139423 \\ 0.0994889 \\ 0.942364 \\ 0.957382 \\ 0. \end{array} &
\begin{array}{c} 0.080028 \\ 0.159309 \\ 0.956041 \\ 0.0895401 \\ 0.13316 \\ 0.125717 \\ 0.688247 \end{array} &
\begin{array}{c} 0.0900315 \\ 0.955851 \\ 0.109546 \\ 0.0696423 \\ 0.0409723 \\ 0.0967053 \\ 0. \end{array} &
\begin{array}{c} 0.0700245 \\ 0.0696975 \\ 0.0597525 \\ 0.0596934 \\ 0.0512154 \\ 0.0967053 \\ 0. \end{array} &
\begin{array}{c} 0.110039 \\ 0.0597407 \\ 0.119505 \\ 0.955094 \\ 0.0921878 \\ 0.0676937 \\ 0. \end{array} &
\left.\begin{array}{c} 0.020007 \\ 0.0696975 \\ 0.0995876 \\ 0.119387 \\ 0.0614585 \\ 0.0676937 \\ 0.229416 \end{array}\right)
\end{array}$$

After the first round we can see that the 75 unintelligent attacking voters and the 75 intelligent attacking voters have equally swayed the vote in their preferred direction. 5/9 and 1/9 respectively.

Iteration 18:

$$\begin{array}{ccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\left(\begin{array}{c} 0.282011 \\ 0.00463814 \\ 0.000983159 \\ 0.928653 \\ 0.0452327 \\ 0.0160324 \\ 0.00850506 \end{array}\right. &
\begin{array}{c} 0.956868 \\ 0.0113064 \\ 0.0280894 \\ 0.0000777306 \\ 0.116043 \\ 0.99476 \\ 0. \end{array} &
\begin{array}{c} 0.0305578 \\ 0.0209938 \\ 0.011191 \\ 0.00264332 \\ 0.949644 \\ 0.0153069 \\ 0. \end{array} &
\begin{array}{c} 0.000758157 \\ 0.0125415 \\ 0.0463455 \\ 0.000455419 \\ 0.285585 \\ 0.082728 \\ 0. \end{array} &
\begin{array}{c} 0.00189018 \\ 0.0407142 \\ 0.997537 \\ 0.0135943 \\ 0.00147775 \\ 0.0125949 \\ 0.175312 \end{array} &
\begin{array}{c} 0.00164208 \\ 0.998374 \\ 0.0111307 \\ 0.0121179 \\ 0.00152116 \\ 0.0392681 \\ 0. \end{array} &
\begin{array}{c} 0.000533049 \\ 0.0197928 \\ 0.019897 \\ 0.000266312 \\ 0.000171094 \\ 0.0292044 \\ 0. \end{array} &
\begin{array}{c} 0.0614862 \\ 0.0212958 \\ 0.0280821 \\ 0.370434 \\ 0.0331565 \\ 0.0237416 \\ 0. \end{array} &
\left.\begin{array}{c} 0.0123342 \\ 0.000118725 \\ 0.023416 \\ 0.00660024 \\ 0.000916344 \\ 0.000316685 \\ 0.984476 \end{array}\right)
\end{array}$$

After the 18 iterations the algorithm has terminated. We can see that the TDT algorithm has managed to reduce the trustworthiness of the 75 unintelligent attacking voters and the 75 intelligent attacking voters to a point where their attack on item 7 is negligible. The 25 honest voters have managed to beat the 150 attacking voters. Thus item 7 received the 9/9 score it deserved.

3 . 2 TDT: Early Self Inflation

A potential issue with the TDT algorithm is product owners submitting many high self votes before other voters submit honest scores for the product. (See file TDT_Early_Self_Inflation.nb)

Setup:

In this test we have:

- 25 honest voters who agree that product 7 should receive a rating of 1/9.
- 50 intelligent attacking voters who aim to increase the score of item 7 to a 9/9.
- 75 unintelligent attackers are trying to change the score of 7 to 9/9.

Cast time generation:

- Honest voters: Randomised 1 - 10
- Intelligent attacking voters: 1 for product 7 else Randomised 8 - 10
- Unintelligent attacking voters: 1 for product 7 else Randomised 1 - 10

Iteration 7:

$$\begin{array}{ccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\left(\begin{matrix} 0.0116954 \\ 0.00278173 \\ 1.24531\times10^{-6} \\ 0.999774 \\ 0.00283224 \\ 1.15824\times10^{-6} \\ 0.999997 \end{matrix}\right. &
\begin{matrix} 0.999931 \\ 7.76126\times10^{-9} \\ 7.28752\times10^{-8} \\ 4.87968\times10^{-9} \\ 0.0117195 \\ 0.999894 \\ 0. \end{matrix} &
\begin{matrix} 1.12928\times10^{-6} \\ 2.95481\times10^{-6} \\ 2.09126\times10^{-6} \\ 0.000761273 \\ 0.999884 \\ 5.4462\times10^{-8} \\ 0. \end{matrix} &
\begin{matrix} 0.0000127911 \\ 3.18511\times10^{-9} \\ 0.0123241 \\ 3.17394\times10^{-6} \\ 0.00900972 \\ 0.0144276 \\ 0. \end{matrix} &
\begin{matrix} 6.16787\times10^{-9} \\ 2.18528\times10^{-6} \\ 0.999923 \\ 0.0000128599 \\ 0.0000147404 \\ 2.00294\times10^{-6} \\ 0. \end{matrix} &
\begin{matrix} 1.29256\times10^{-6} \\ 0.999996 \\ 0.00175484 \\ 5.28667\times10^{-8} \\ 3.25786\times10^{-9} \\ 7.06244\times10^{-9} \\ 0. \end{matrix} &
\begin{matrix} 1.16115\times10^{-14} \\ 0.0000135239 \\ 6.32931\times10^{-9} \\ 1.96817\times10^{-9} \\ 9.98706\times10^{-8} \\ 0.00176906 \\ 0. \end{matrix} &
\begin{matrix} 0.00073959 \\ 0.000733122 \\ 0.0000146628 \\ 0.0207236 \\ 0.00252475 \\ 2.06524\times10^{-6} \\ 0. \end{matrix} &
\left.\begin{matrix} 1.10864\times10^{-6} \\ 2.26891\times10^{-8} \\ 8.02521\times10^{-7} \\ 0.00460607 \\ 3.19814\times10^{-9} \\ 4.95167\times10^{-8} \\ 0.00250021 \end{matrix}\right)
\end{array}$$

After the 7 iterations the algorithm has terminated. We can see that the TDT algorithm has managed to reduce the trustworthiness of the 75 unintelligent attacking voters and the 50 intelligent attacking voters to a point where their attack on item 7 is negligible. The 25 honest voters have managed to beat the 125 attacking voters. Thus item 7 received the 1/9 score it deserved and not the inflated score by the product owners.

To achieve this the α value needed to be increased to 5 in order to further marginalise the unintelligent random voting attackers. This was because their early votes for item 7 caused their trustworthiness to increase to a point which swayed the consensus.

5 Discussion

In this section I will broadly discuss the results of the experiments and then touch on some particular points relevant to the algorithm.

5 . 1 Experimental Results

From the tests I ran it was clear to see that the TDT algorithm outperformed the RTV algorithm. We saw that 10 intelligent voters were able to out weigh 25 honest voters when using the RTV, while TDT was able to hold off an attack of 75 intelligent voters. To confirm these results we need to run tests on real world datasets. It is likely that intelligent voters could find a balance between vote accuracy and cast timing in order to maximise their trustworthiness, however this could prove to be quite difficult without knowing the code behind the algorithm.

5 . 2 Distance Between Choices

Reference [10] (An Iterative Algorithm for Reputation Aggregation in Multi-dimensional and Multinomial Rating Systems) discusses a limitation of the RTV algorithm ~

> *"The first limitation is that in RTV the order of the choices is not important and the distance between the choices is not defined. For example, when a rater chooses the Nominee1 as the most popular candidate and another rater selects the Nominee2, it does not make sense to talk about the distance between these two options. However, in a movie rating system, if one of the raters chooses 4 star rating of a movie and another chooses a 3 star rating then a distance between there ratings is well defined and might be important for rating methods. The distance between choices is not taken into account in the RTV algorithm."*

I will present a counter to this point and argue that the RTV algorithm and the TDT algorithm do in fact take into account distance between votes via the voters themselves.

Firstly distance between votes determines the degree of similarity of each choice. For example if I were to vote 10/10 on a movie I am more likely to agree with a user who voted 9/10 over a user who voted 1/10. Thus, the algorithm should account for this when calculating trustworthiness.

Although not directly, the RTV and TDT algorithms do in fact take this distance into account. We know that the two algorithms calculate trustworthiness based on the degree to which the voter conforms to the community. Lets assume that the major of honest voters where to vote 7/10. We would expect a normal (gaussian) distribution in the votes, thus there will be far greater votes of 6/10 and 8/10 than there are votes of 1/10. Therefore a user who votes 6/10 will still build some level of trustworthiness, far more than a user who votes 1/10. So we can see that the distance between the ‘correct’ vote and a ‘close’ vote is still taken into account.

6 Conclusion

The modifications on the Rating Through Voting algorithm to create the Time Dependent Trustworthiness algorithm have proved to increase the accuracy of online rating systems in the circumstances I tested. While further testing and experimentation is needed, (specifically using real world data) it is clear that the cast time of votes is a important metric which should not be overlooked when calculating community consensus. I believe wide spread implementation of online rating system algorithms based on similar metrics can be implemented to greatly improve the experience for all users.

References

[1] Small Business. 2017. *Tackling the Exponential Growth Rates from E-commerce Through Technology*. Available at: https://smallbusiness.yahoo.com/advisor/post/131183784382/e-commerce-shows-no-signs-of-abating-as-studies-show

[2] Techdirt.. 2017. *The Rate Of TV Cord Cutting Is Actually Worse Than You Think | Techdirt*. Available at: https://www.techdirt.com/articles/20170531/05304937480/rate-tv-cord-cutting-is-actually-worse-than-you-think.shtml

[3] The Data Point. 2017. *How Many Ecommerce Companies Are There? - The Data Point*. Available at: https://blog.rjmetrics.com/2014/06/18/how-many-ecommerce-companies-are-there/

[4] ExportX. 2017. *(2013) How Many Products Does Amazon Sell? | ExportX*. Available at: https://export-x.com/2013/12/15/many-products-amazon-sell/

[5] Spotting fake reviewer groups in consumer reviews. 2017. *Spotting fake reviewer groups in consumer reviews*. Available at: https://dl.acm.org/citation.cfm?id=2187863

[6] Inbound.org. 2017. *A Competitor Bought 200 1-Star Reviews For Our Facebook Page - Here's Our Story*. Available at: https://inbound.org/blog/a-competitor-bought-200-1-star-reviews-for-our-facebook-page-here-s-our-story-1

[7] Google Books. 2017. *Web Information Systems Engineering â WISE 2017: 18th International ... - Google Books*. Available at: https://books.google.com.au/books?id=UXo4DwAAQBAJ&pg=PA182&lpg=PA182&dq=Morgan,+J.,+Brown,+J.:+Reputation+in+online+auctions:+The+market+for+trust.+Calif.+Manag.+Rev.+49(1),+61%E2%80%9381+(2006)&source=bl&ots=jNmkOjg1oX&sig=RSqUJFnP1hs8srrAWXCyHBjcJz8&hl=en&sa=X&ved=0ahUKEwjo0bGw4sHXAhUGH5QKHS85A-gQ6AEIMTAB#v=onepage&q=Morgan%2C%20J.%2C%20Brown%2C%20J.%3A%20Reputation%20in%20online%20auctions%3A%20The%20market%20for%20trust.%20Calif.%20Manag.%20Rev.%2049(1)%2C%2061%E2%80%9381%20(2006)&f=false

[8] Detecting product review spammers using rating behaviors — UICollaboratory Research Profiles. 2017. *Detecting product review spammers using rating behaviors — UICollaboratory Research Profiles*. Available at: https://uic.pure.elsevier.com/en/publications/detecting-product-review-spammers-using-rating-behaviors

[9] Iterative Filtering in Reputation Systems. 2017. *Iterative Filtering in Reputation Systems*. Available at: https://dl.acm.org/citation.cfm?id=1958300

[10] An Iterative Method for Calculating Robust Rating Scores - IEEE Journals & Magazine. 2017. *An Iterative Method for Calculating Robust Rating Scores - IEEE Journals & Magazine*. Available at: http://ieeexplore.ieee.org/document/6762988/.

[11] Rating through Voting: An Iterative Method for Robust Rating | Mohammad Allahbakhsh and Aleksandar Ignjatovic - Academia.edu. 2017. *Rating through Voting: An Iterative Method for Robust Rating | Mohammad Allahbakhsh and Aleksandar Ignjatovic - Academia.edu*. [ONLINE] Available at: http://www.academia.edu/34418481/Rating_through_Voting_An_Iterative_Method_for_Robust_Rating