CS 765 Assignment 3
Decentralized Fact-Checker

Isha Arora
210050070

Karan Godara
210050082

Indian Institute of Technology, Bombay

April 14, 2024

Contents

1 Part A 2
1.1 Explanation of our DApp 2
1.1.1 Registering on the DApp 2
1.1.2 Deregistering from the DApp 2
1.1.3 Voting on the DApp 2
1.1.4 Genre of news 3
1.1.5 Genre list management 3
1.1.6 Compututation and re-computation of trustworthiness of voters 4
1.1.7 Uploading of news 5
1.1.8 Distribution of Funds from Verification Requests 5
1.1.9 Reaching the final decision 6
1.1.10 Minimum votes to reach a decision 7
1.2 How our DApp handles the issues given in the problem statement 7
1.2.1 Sybil attack 7
1.2.2 Method to evaluate or re-evaluate the trustworthiness of voters 7
1.2.3 Assigning trustworthiness score to voters 8
1.2.4 Incentivisation of rational voters 8
1.2.5 Uploading a news item 8
1.2.6 Bootstrapping 8
2 Part B 10
3 Part C 10
3.1 Effect of varying p and q on our DApp 11
3.1.1 Keeping q fixed as 0.1 11
3.1.2 Keeping q fixed as 0.2 13
3.1.3 Keeping q fixed as 0.3 15
3.2 Robustness of the System 18
3.3 Varying the Total Number of Voters 19
3.4 Alternate Strategy for Malicious Voters 22

1 Part A

1.1 Explanation of our DApp

1.1.1 Registering on the DApp

1. **Procedure:**

   Each new node, identified by its Public Key, intending to join the voting committee must undergo a registration process. During registration, the node is required to submit a certain amount of Ether (ETH), denoted as “x”, as collateral.

2. **Collateral Purpose:**

   The submitted Ether serves as collateral and acts as a deterrent against Sybil attacks, as elaborated further.

1.1.2 Deregistering from the DApp

1. **Voluntary Withdrawal:**

   Nodes have the option to reclaim their initially submitted Ether by voluntarily de-registering from the voting committee. This ensures a flexible and voluntary participation model.

2. **Maintenance of History:**

   The DApp maintains a history of the trustworthiness of nodes (identified by their public keys) that have been part of the voting committee. This is done so that even after de-registering, a node’s trustworthiness score remains intact within the system. This retention of trustworthiness scores ensures that a node’s previous contributions are acknowledged and accounted for, irrespective of their current participation status and thus facilitates smooth transitions for nodes that choose to de-register and potentially re-join the voting committee in the future. One need not keep their Ethers deposited in the DApp in case they need them somewhere else, to be able to reap benefits out of the past contributions to the system in the future. This makes the system easy to join and leave.

1.1.3 Voting on the DApp

1. **Binary Voting System:**

   The voting mechanism within the DApp adopts a binary system, where participants can cast votes of either 0 or 1. The votes are limited to only 0 and 1 and not to a range such as 1 to 10 for reason explained in the following point.

2. **Mitigation of Malicious Influence:**

   A binary voting system helps mitigate the risk of malicious actors manipulating the outcome by consistently voting at extreme ends of a wider voting scale. An honest node being cautious might vote for a 2 or 3(if they think the news is fake) or a 7 or 8(if they think the news is true) unless they are 100 percent sure. Whereas on the other hand, a dishonest node would vote for a 0 or 10 (to rig the system) which would have a higher weight than the honest node’s non extreme votes. Thus if voters were to vote on a non-binary scale, malicious entities’ extreme votes would exert a greater influence on the final result which is not desirable. Hence, voters can only cast binary votes.

1.1.4 Genre of news

1. **Genre Specification:**

   When submitting a news article to the DApp, users are prompted to classify the news into one of the genres from a pre-existing list maintained within the contract.

2. **Flexibility of new genres:**

   In instances where a user wishes to classify news under a genre not present in the predefined list, the DApp provides a mechanism for creating new genres which is explained below. This ensured that the classification system remains adaptable to evolving news categories.

1.1.5 Genre list management

1. **Initial Configuration:**

   The genre list is initially established by the creator of the contract, providing a foundational structure for categorizing news articles.

2. **Mechanism to add a new genre:**

   General score (not genre wise trustworthiness) of a user is computed as the total number of correct votes casted by the user minus the total number of incorrect votes cast by a user. Only voters with a certain minimum general score (set as 50 * (slow moving average of the total votes received per day for all genres)) are eligible to introduce a new genre on depositing a certain amount of ether. These ethers are non-refundable. For the new genre, each voter would have a minimum rating (1 in our case) in the start. To increase the robustness of the system even further, when a voter with required credentials and ethers submits the request to create a new genre, a poll might be conducted on whether the topic should be introduced or not in which only the users with a certain minimum general score can vote.

3. **Reason for the adopted mechanism:**

   We enforce restrictions on the initiation of new genre creation to prevent the proliferation of countless genres, many of which may overlap or be subsets of existing genres. This is done because:

   (a) Allowing unrestrained genre creation would pose challenges in managing the trustworthiness records associated with each genre(could be a lot in number), potentially leading to memory overflow issues.

   (b) Even in the absence of memory concerns, granting universal rights to create genres would be detrimental. This is because for a newly created genre, the trustworthiness score of each voter is initialised as 1(and not set high for truthful voters and low for malicious voters) and thus the reliability of the DApp would decrease drastically in the presence of malicious voters.

   (c) Also, lots and lots of overlapping or subsets-supersets of genres would lead to a lesser number of news articles in each genre thus leading to lesser chances to update the trustworthiness score of the voters in each genre and decreasing the reliability of the system.

   (d) Overlapping genres would make it tough for the user who wants to query the DApp with a news to pinpoint the exact genre his news belongs to.

   (e) Additionally, the creation of redundant genres could dilute the expertise recognition within related fields. For instance, say a new genre like “rotational mechanics” were introduced when “mechanics,” “Physics,” and “Sciences” already exist. Due to the creation of a new genre, all of the voters would be given the same rating and the benefit of having high trustworthiness score of people skilled in the fields of 'mechanics' and 'physics' would be lost (this trustworthiness score could have helped greatly in predicting the news correctly as there is a high chance a person having expertise in mechanics or physics has an expertise in rotational mechanics but due to the creation of a new genre, is assigned a trustworthiness score of 1 like any other voter). Therefore, the creation of genres which are a subset or superset of other genres would lead to people skilled in the field not being weighed more as their trustworthiness score of the related field wont matter in the newly created field. This would lead to less reliable estimates and thus the reliability of DApp would go for a toss.

Therefore to prevent all of the fore-stated problems, we allow only reliable members of the system to propose a new genre by providing some ethereum. If the creation of a new genre is absolutely required, money can be pooled in by the reliable voters of the DApp to create it. A real life closest example of this is 'Stack overflow'. On that platform, only experienced members of the community and not everybody can create a new topic(genre for us) to ensure quality and expertise within the community but everybody can ask or answer a question (voting or querying for us) to ensure inclusivity.

1.1.6 Compututation and re-computation of trustworthiness of voters

We adopt a dynamic adjustment of trustworthiness ratings based on voting behavior and genre-specific factors allows for a dynamic and responsive evaluation of voter reliability.

1. **Genre-Specific Trustworthiness Score:**

   (a) Each voter is assigned a genre-specific trustworthiness score, denoting their reliability in assessing news articles within a particular genre.

   (b) The maximum rating per genre is initially set as 100, starting from 1 as specified earlier.

   (c) Genre specific ratings are computed to account for the differing levels for expertise of a voter in different fields. For example, someone may give excellent opinions about news related to Physics but is not so trustworthy on topics related to Politics or Economics. If the same trustworthiness score is maintained for the voter for each genre, then an accurate judgement of the actual reliability of the voter can't be made

2. **Re-evaluation of genre-specific trustworthiness scores**

   (a) Increase in rating initially would be easier for lower ratings than for higher rating, to prevent saturation of large numbers of people at higher ratings.

   (b) The varying adjustment rates across different rating ranges, with higher rates for lower ratings and lower rates for higher ratings, ensure a balanced and gradual adjustment process.

   (c) To compute the increase in ratings, we will maintain a slow-moving average of requests per day(in the genre being considered), let's call this average "r". This increase in rating can be controlled in the following manner:

      i. **Initial Adjustment Range (1 to 5):**
      Within the rating range of 1 to 5, every max(1,0.1r) correct votes result in an increase by 1 rating whereas if the voter happens to cast max(1, 0.1*r) incorrect votes first, it would result in a decrease by 1 rating (capped below by 1).

ii. **Intermediate Adjustment Range (6 to 50):**
If the rating lies in the range [(5*k)+1, 5*(k+1)] (for k belonging to [1, 9]), every max(1, k*r) correct votes result in an increase in 1 rating whereas if max(1, k*r) incorrect votes happen to be cast first by the voter, the voters rating would decrease by 1.

iii. **Advanced Adjustment Range (51 to 100):**
For ratings in the range [51, 100], every max(1, 10*r) correct votes increase by 1 rating whereas if max(1, 10*r) incorrect votes happen to be cast first by the voter, its rating would decrease by 1.

3. **Cap for High Ratings ($>= 51$):**

   (a) When a voter's trustworthiness score reaches or exceeds 51, their maximum trustworthiness score is capped by the ratio of the number of correct votes to the total number of votes cast by the voter in that specific genre.

   (b) Capping the maximum rating based on the accuracy of voting behavior ensures that only consistently accurate voters achieve the highest trustworthiness scores, enhancing the reliability of the trustworthiness estimation process. It ensures that only voters who always give the correct answer can reach a rating of 100, while those with lower accuracy are capped accordingly. A voter with say 70 percent accuracy would have a max rating of 70 only. This ensures an accurate estimate of the trustworthiness of voters.

1.1.7 Uploading of news

1. **Submission Requirement:**

   Users posting news articles for fact-checking must provide a designated amount of Ether (ETH), denoted as "y," to access the verification service.

2. **Purpose of Fee:**

   The Ethereum fee serves as a financial commitment from users to utilize the fact-checking service, ensuring a level of investment in the verification process. Thus, the fee requirement encourages users to submit legitimate and substantive news articles for verification, as they have invested resources in the process.

3. **Verification Process of the News Article:**

   When initiating a request for news verification, users are required to submit the entire news article string to the DApp. Voters will subsequently vote on the hash of the news article to identify and evaluate its authenticity.

1.1.8 Distribution of Funds from Verification Requests

1. **Allocation of Funds:**

   The Ethereum funds submitted by the user requesting the fact-checking of a news article, denoted as "y," are distributed among the voters who align with the majority decision in the decision-making process. For example, if the news article is deemed fake, the allocated funds are distributed among the voters who voted in favor of the fake designation.

2. **Proportional Distribution:**

(a) The distribution of funds is proportional to the trustworthiness of voters within the genre of the news article.

(b) Each voter receives a share of the allocated funds based on their individual trustworthiness score relative to the total trustworthiness of all voters who voted in the majority.

(c) **Formula:**

$$\text{Reward for voter } vi = \frac{p_i}{\sum_{j=1}^{q} p_j} \times y$$

where:

i. p_i is the trustworthiness of voter vi,

ii. $\sum_{j=1}^{q} p_j$ is the sum of trustworthiness scores of all voters who voted in the majority,

iii. y is the total amount of money submitted for the verification request.

3. **Reasoning behind the distribution method adopted**:

   Distributing funds among voters who correctly assess the news article ensures fairness and incentivizes accurate participation in the fact-checking process.

1.1.9 Reaching the final decision

1. **Timestamp Tracking:**

   The DApp maintains a mapping from the hash of each news article to its corresponding timestamp, tracking the initiation of verification requests.

2. After a designated period, such as 20 units of time or block equivalents, the votes cast for the news article are evaluated.

3. **Weighted Sum Calculation:**

   For each news article, a weighted sum is computed by aggregating the votes cast by voters, where each vote (0 or 1) is weighted by the voter's trustworthiness. The result is stored in a variable, such as 'weighted_sum'.

4. **Max Weighted Sum Calculation:**

   Additionally, the maximum possible weighted sum that could have been achieved if all voters who participated in the verification process had voted the news article as real (1) is computed. This value is stored in a variable, such as 'max_sum'.

5. **Decision Criteria:**

   If the calculated 'weighted_sum' is greater than or equal to half of the 'max_sum', the news article is classified as real. Otherwise, if 'weighted_sum' is less than half of 'max_sum', the news article is classified as fake.

6. **Indecision Condition:**

   In cases where only a limited number of low-trustworthiness voters participate in the voting process, potentially indicating malicious intent, a decision may not be reached. The condition for indecision and the steps taken when such a situation is encountered are further elaborated in the subsequent point.

1.1.10 Minimum votes to reach a decision

1. **Threshold Determination:**

   The DApp sets a minimum threshold for the sum of trustworthiness scores of voters required to reach a decision on the authenticity of a news article. This threshold is defined as at least 25% of the total sum of trustworthiness of the users(not voters for a particular article) in a genre to ensure a sufficient level of participation from trustworthy voters.

2. **Decision-Making Process:**

   (a) If the sum of trustworthiness scores of participating voters meets or exceeds the established threshold, the decision is determined based on the majority of votes.

   (b) If the sum of trustworthiness scores falls below the 25% threshold, indicating insufficient participation from trustworthy voters, the DApp refunds the submitted Ethereum back to the user who initiated the verification request and declares that the DApp cannot confidently determine the authenticity of the news article due to the lack of reliable voter participation.

1.2 How our DApp handles the issues given in the problem statement

1.2.1 Sybil attack

Problem Statement

A malicious person can create multiple identities and vote to skew the result in any direction.

How our system handles it

As discussed in 1.1.1, all the voters need to submit a collateral amount to be registered and to be able to vote in the DApp. By imposing a financial cost on registration, the DApp discourages malicious actors from creating numerous fake identities, as doing so would require a significant investment of ether. Thus, Sybil Attack is prevented.

1.2.2 Method to evaluate or re-evaluate the trustworthiness of voters

Problem Statement

The Dapp should evaluate how trustworthy different voters are based on how they vote. Note that someone might game the system to get a higher trustworthy rating. A method that is more robust to such gaming of the system, is preferable.

How our system handles it

The trustworthiness of the voters is computed and re-computed as explained in great detail in 1.1.6. To get a high trustworthy rating in our system, a person needs to have voted correctly. If a voter always votes incorrectly, his rating would remain 1 and he would never get any reward. If the person tries to game the system by initially giving correct answers and gaining a high rating and then after he gets a high rating, he would start giving incorrect votes. But this won't work in his favor as till the majority of voters with high trustworthiness remain honest, whenever this malicious voter casts an incorrect vote, he would receive no reward for that news. Additionally, his rating would fall. This can also be seen empirically as we simulated this strategy of malicious voters in 3.4. It can be seen clearly in our simulations that initially as the malicious acts as an honest voter, its rating increases in the first half of the simulation. Then in the second half, when the malicious voter starts to game the system, gets a 0 reward and his rating also decreases, and by the end of the simulation, the rating becomes 1(which is the lowest possible rating). Thus, our method is robust to the gaming of the system by malicious attackers.

1.2.3 Assigning trustworthiness score to voters

Problem Statement

The opinions of more trustworthy voters should be given more weight. However, we must keep in mind that someone may be more trustworthy for certain types of news and not others. For example, someone may give excellent opinions about news related to Physics but is not so trustworthy on topics related to Politics or Economics.

How our system handles it

We assign genre-wise trustworthiness scores to all of the voters. The list of genres is maintained as described in 1.1.5. The user who upload the news articles needs to assign a genre to it as described in 1.1.4 and if the user thinks a new genre might need to be created, there is a mechanism to create a new genre 1.1.5. The trustworthiness of all users for any new genre is initially set as 1. the genre wise trustworthiness scores of all the users are further computed and recomputed as described in great detail in 1.1.6. The trustworthiness scores are then accounted for in the process of decision-making so that more trustworthy voters should be given more weight. The final decision is taken using weighing by trustworthiness scores as described in 1.1.9.

1.2.4 Incentivisation of rational voters

Problem Statement

Rational voters are to be incentivised to participate and vote truthfully to the best of their ability.

How our system handles it

Rational voters are incentivized to participate and vote truthfully to the best of their abilities as the more they vote correctly(their opinion aligns with the final decision of the DApp), the higher their trustworthiness scores would become. The exact details about how the trustworthiness of voters is updated as per their votes are described in 1.1.6. The user who wants to get a piece of news fact-checked needs to submit a fee as described in 1.1.7. This money would be distributed among the voters who voted correctly in proportion to their trustworthiness scores. The way this is done is described in 1.1.8. Thus, the more voters vote correctly, the higher their trustworthiness scores would be and thus the more rewards they would get for giving the correct answer, and rational voters would be incentivized to vote truthfully.

1.2.5 Uploading a news item

Problem Statement

Some efficient method should be used to identify a news item (which is to be evaluated) in the Dapp.

How our system handles it

The process of uploading and identifying a news item is described in 1.1.7. The news intially entered by the user is passed through hash function to compute its hash digest. This hash digest is then used to refer to the news from this point. Hence, during voting, checking result of the news etc one uses hash digest of the news rather than the news itself. This helps in making sure not too much bandwidth is used to identify news itself and thus the process is efficient.

1.2.6 Bootstrapping

Problem Statement

If the Dapp does not have any trustworthy rating of different initial voters, then how to get started with fact-checking news?

How our system handles it

Initially, the trustworthiness of all of the voters for all of the genres is set as 1 and all of the voters have equal say in the fact-checking process. The decision of the DApp is the decision of the majority of the voters and as per the votes, the trustworthiness scores of all of the voters are updated as explained in 1.1.6. If the majority decision is the right decision, then correct updation of trustworthiness of voters would take place(the ratings of malicious would stay at 1 and the rating of honest voters would increase and thus the DApp would start estimating news more reliably and the system would move in the right direction). However if in the initial scenario if the malicious voters form a majority ($> 50\%$), then the rating of malicious voters would increase, they would have a greater say in future decision making process (even greater than $> 50\%$ due to their higher trustworthiness) and the system would go for a toss.

An in-depth analysis of the necessary condition for DApp to function correctly is as follows:

Let:

- p: fraction of honest votes with an accuracy of 0.9
- q: fraction of malicious voters
- N: total number of voters who vote

Thus, $p(1-q)(0.1)N$ is the expected number of incorrect votes cast by honest voters with an accuracy of 0.9. Similarly, $(1-p)(1-q)(0.3)N$ is the expected number of incorrect votes cast by honest voters with an accuracy of 0.7. $q \cdot N$ is the expected number of incorrect votes cast by malicious users. As initially, all the voters have a weight of 1, the 'max_sum' of trustworthiness which we require to make the final decision is also N.

At the beginning of the system when the trustworthiness of all the voters is the same (1) and thus all have equal weight, for the DApp to give the correct decision, we require that the number of incorrect votes $< 0.5 \cdot N$.

$$p(1-q)(0.1)N + (1-p)(1-q)(0.3)N + q \cdot N < 0.5 \cdot N$$

After some simplification:

$$\begin{aligned}
&p(1-q)(0.1)N + (1-p)(1-q)(0.3)N + q \cdot N < 0.5 \cdot N\\
&0.1p(1-q)N + 0.3(1-p)(1-q)N + qN < 0.5N\\
&0.1pN - 0.1pqN + 0.3N - 0.3p(1-q)N + qN < 0.5N\\
&0.1pN + 0.3N - 0.1pqN - 0.3pN + 0.3pqN + qN < 0.5N\\
&(0.1p - 0.3p + 0.3q + q)N < 0.5N\\
&(0.3q - 0.2p + 0.3pq)N < 0.5N\\
&0.3q - 0.2p + 0.3pq < 0.5\\
&7q - 2p + 2pq < 2
\end{aligned}$$

or

$$q < \frac{2(1-p)}{7+2p}$$

To get a sense of the numbers obtained, we can see that on setting p as some values as shown in the table below, the maximum value that q can take is:

p	q
0	0.28
0.1	0.3
0.3	0.34
0.5	0.375
0.7	0.40
1	0.44

Thus, if all honest voters vote with an accuracy of 70 percent ($p = 0$), the DApp would work correctly in the presence of malicious voters up to 28 percent of the total voters. Similarly, if all the honest voters vote with an accuracy of 90 percent ($p = 1$), the DApp would work correctly in the presence of malicious voters up to 44 percent of the total voters.

Also, it is to be noted that initially, as all of the voters have equal reliability, these stricter bounds need to be imposed to prevent a 51% attack. Later, as the trustworthiness of voters is updated, these bounds are loosened, and the honest people would have greater weight than the malicious ones. This is empirically observed in simulations done in 3.2.

2 Part B

The pseudo code written in solidity syntax is submitted. The name of the file is **"pseudo_code.sol"**. Please refer to it, to map the ideas written in Part A, to their actual implementations.

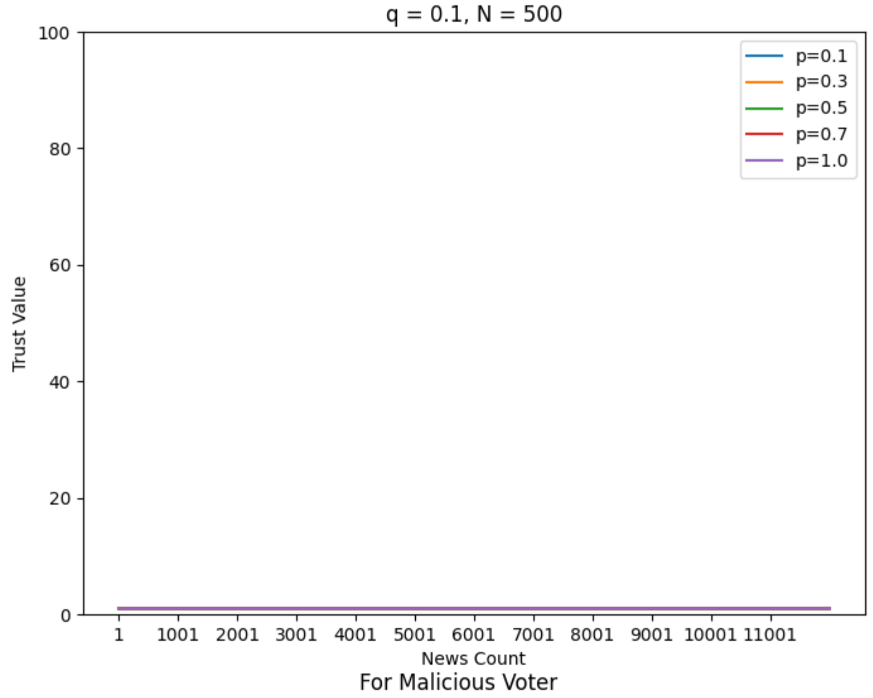
3 Part C

Terminology Used:

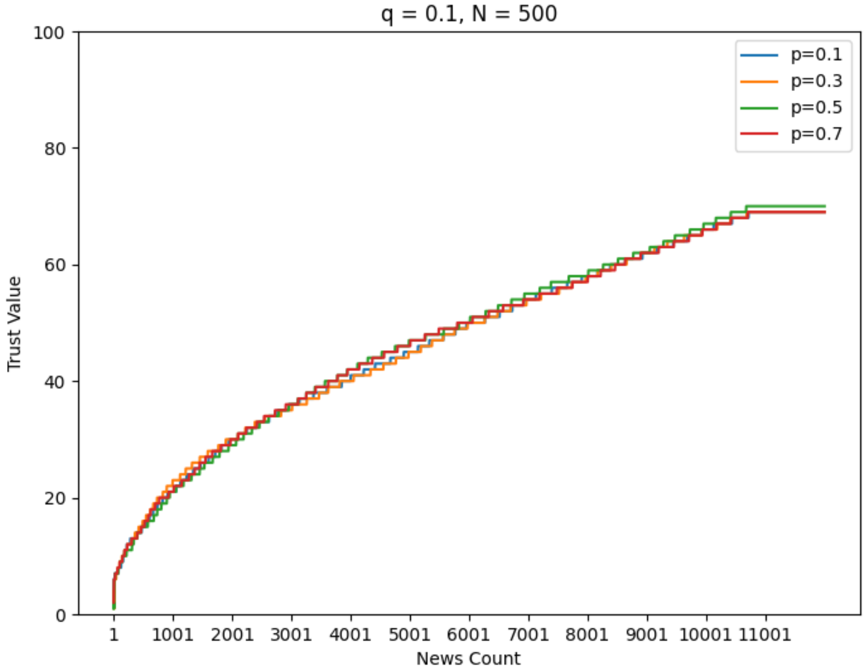
- N: Total number of voters.
- q: Fraction of malicious voters. Out of a total of N voters, q represents the fraction of voters who are malicious and always vote for the incorrect option.
- p: Fraction of honest voters who are very trustworthy and give the correct vote with a probability of 0.9.
- Num Correct: Number of news items correctly classified by the DApp
- Num Incorrect: Number of news items incorrectly classified by the DApp
- Total News Count: Total number of news items given to DApp for fact-checking in the simulation

3.1 Effect of varying p and q on our DApp

3.1.1 Keeping q fixed as 0.1

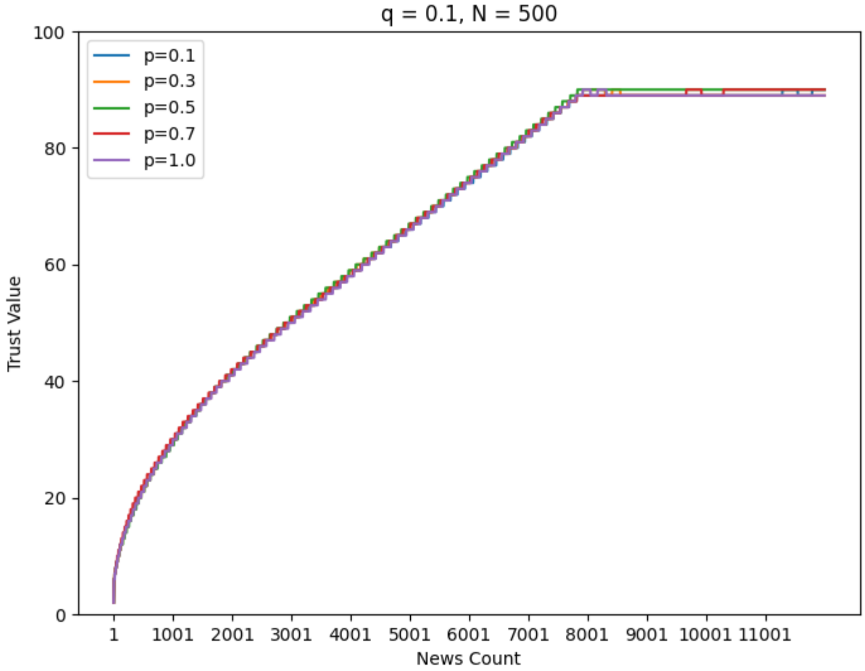


As we can see in 1, our DApp is able to estimate the trustworthiness of malicious voters really well for all values of p. As the voter never votes correctly, his trustworthiness remains 1 (the lowest possible value of trustworthiness for our system) throughout the simulation.



For Honest Voter with 70% correctness

As we can see in 2, our DApp is able to estimate the trustworthiness of honest voters with 70 percent accuracy really well for all values of p. As explained in the working of the DAPP, the trustworthiness scores are increased at a fast rate in the beginning and the rate of change becomes gradual as the trustworthiness score of the voter increases. This can be easily observed in the graph as the slope is high in the beginning and gradual as the trustworthiness score increases. The score saturated to 70 by the end of the simulation as one would want to as the the voter answers with 70 percent accuracy.



For Honest Voter with 90% correctness

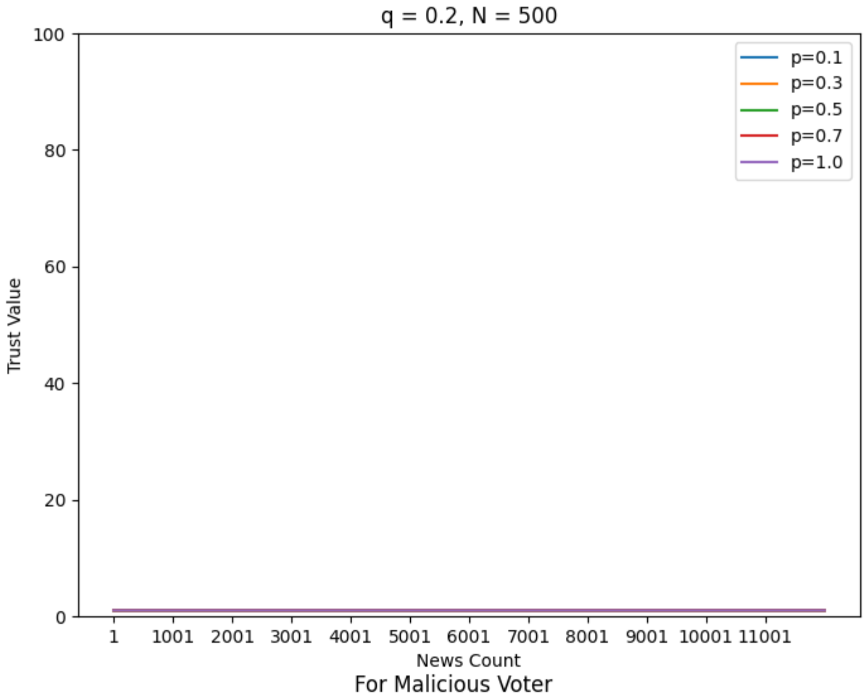
As we can see in 3, our DApp is able to estimate the trustworthiness of honest voters with 90 percent accuracy really well for all values of p. As explained in the working of the DAPP, the trustworthiness scores are increased at a fast rate in the beginning and the rate of change becomes gradual as the trustworthiness score of the voter increases. This can be easily observed in the graph as the slope is high in the beginning and gradual as the trustworthiness score increases. The score saturated to 90 by the end of the simulation as one would want as the the voter answers with 90 percent accuracy.

p	Num Correct	Num Incorrect	Total News Count
0.1	12000	0	12000
0.3	12000	0	12000
0.5	12000	0	12000
0.7	12000	0	12000
1	12000	0	12000

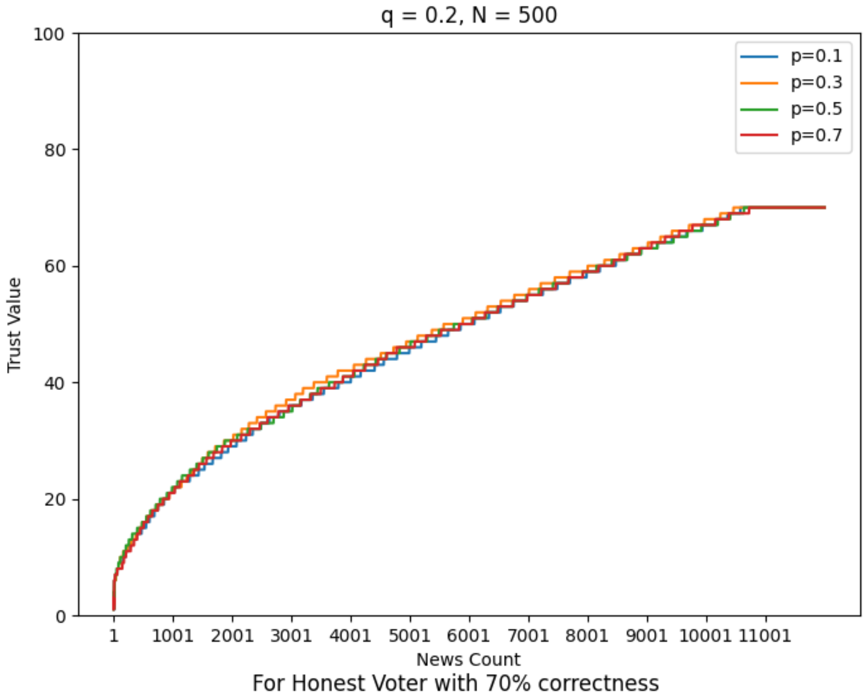
Table 1: News statistics of the DApp for $q = 0.1$

3.1.2 Keeping q fixed as 0.2

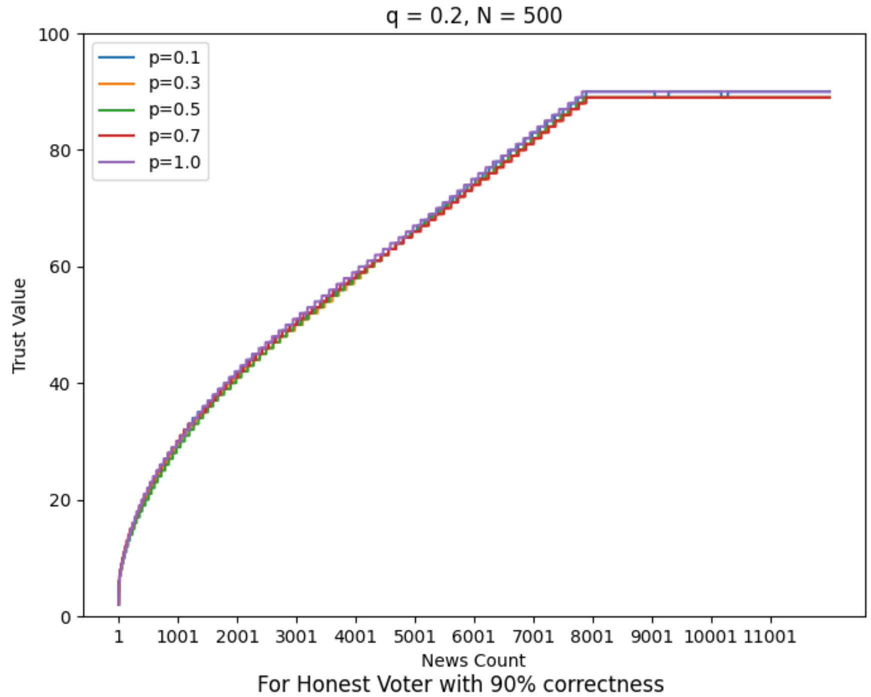
The results obtained for the value of q being kept as 0.2 are very similar to the ones obtained for q being kept as 0.1 and the same observations and conclusions follow.



For Malicious Voter



For Honest Voter with 70% correctness



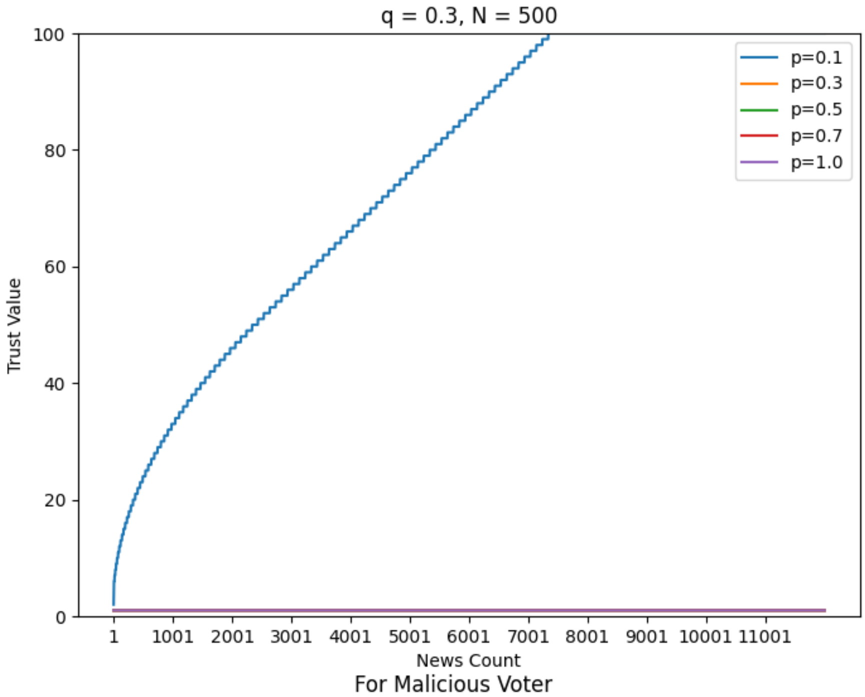
For Honest Voter with 90% correctness

p	Num Correct	Num Incorrect	Total News Count
0.1	12000	0	12000
0.3	12000	0	12000
0.5	12000	0	12000
0.7	12000	0	12000
1	12000	0	12000

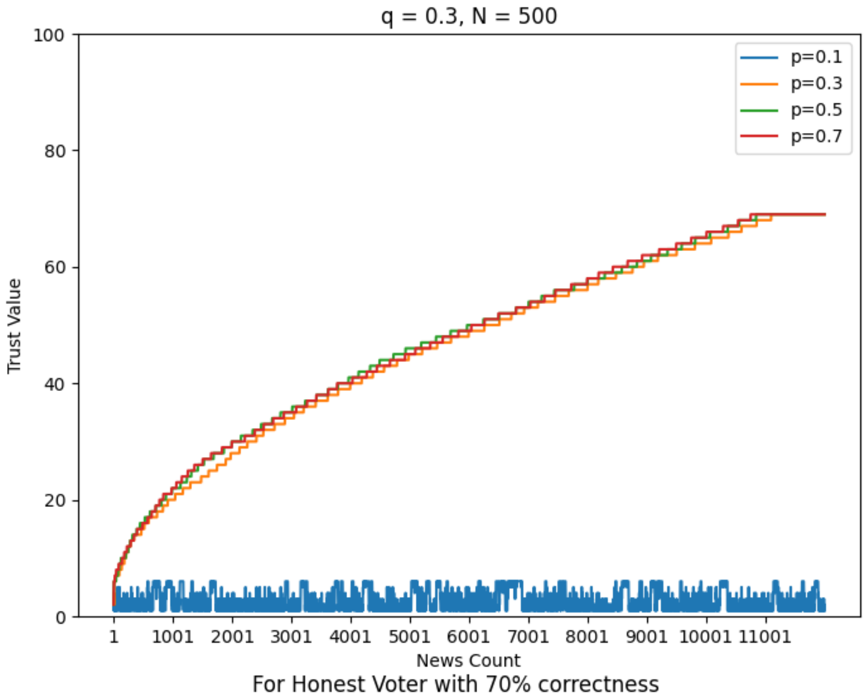
Table 2: News statistics of the DApp for $q = 0.2$

3.1.3 Keeping q fixed as 0.3

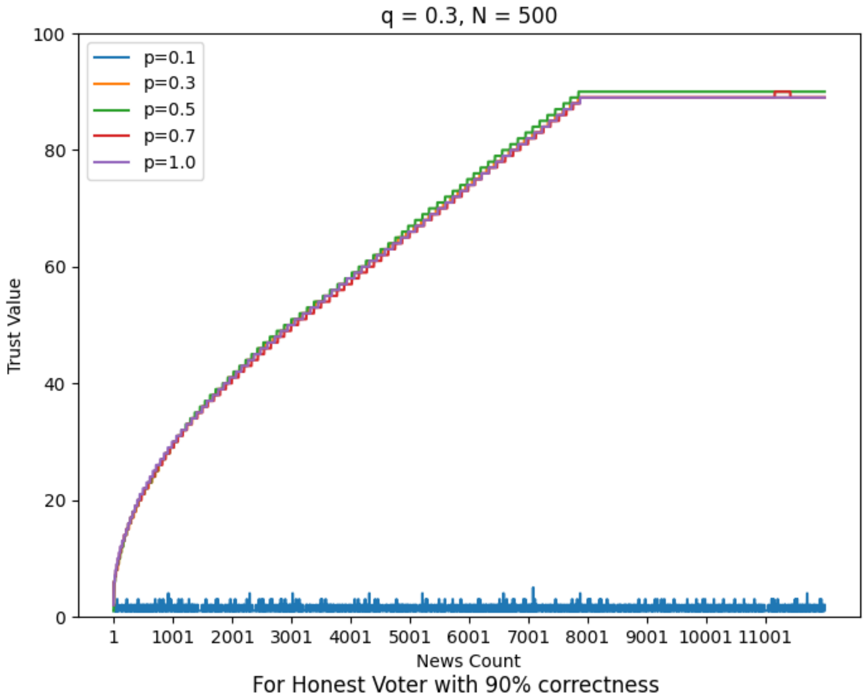
For the values of p as 0.3, 0.5, 0.7, the results are very similar to the ones reported earlier and our DApp works really well in these cases. An interesting case arises when p is 0.1. In 1.2.6, we established the condition required for our DApp to start functioning (51 percent attack does not take place in the starting of the simulation). In the table 1.2.6, it can be seen that for the value of p as 0.1, the value of q should be less than 0.3. At the value of p as 0.1 and q as 0.3, incorrect votes being cast at the start of simulation are about 50 percent. If the incorrect votes are slightly greater than 50 percent, then the rating of malicious voters would increase in the subsequent rounds and the system would go for a havoc. This is what happens in this case as shown in the simulation. By the end of the simulation, the trustworthiness of malicious voter becomes 100, whereas those of honest voters remains around 1.



For Malicious Voter



For Honest Voter with 70% correctness

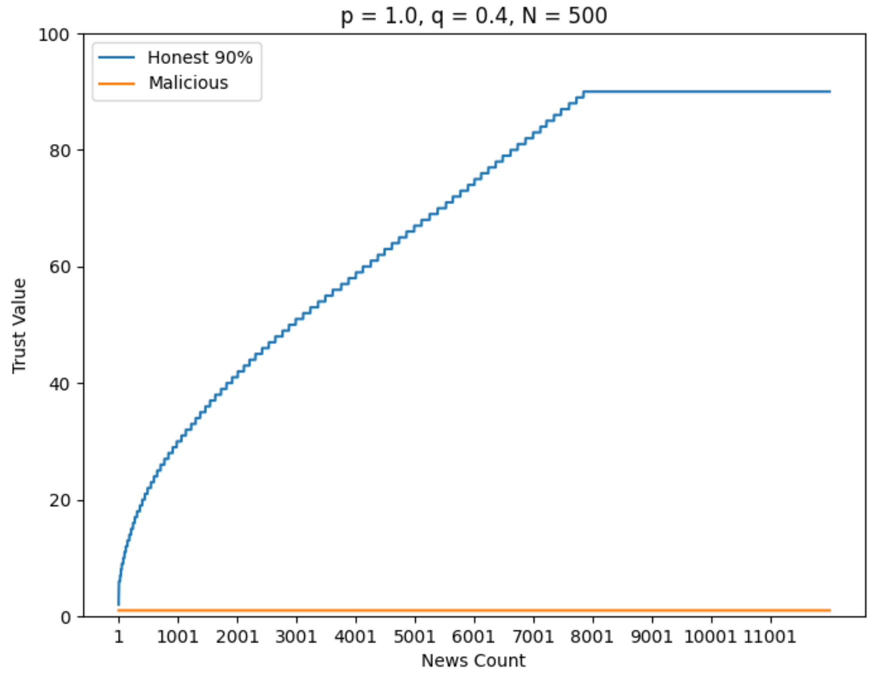


p	Num Correct	Num Incorrect	Total News Count
0.1	0	12000	12000
0.3	12000	0	12000
0.5	12000	0	12000
0.7	12000	0	12000
1	12000	0	12000

Table 3: News statistics of the DApp for $q = 0.3$

Thus, it can be established from the above graphs that our DApp performs really well as estimating the trustworthiness of voters for varying values of p and q.

As we can see in 10, even for the percentage of malicious voters being as high as 40 percent, our DAPP estimates the trustworthiness of all voters really well.

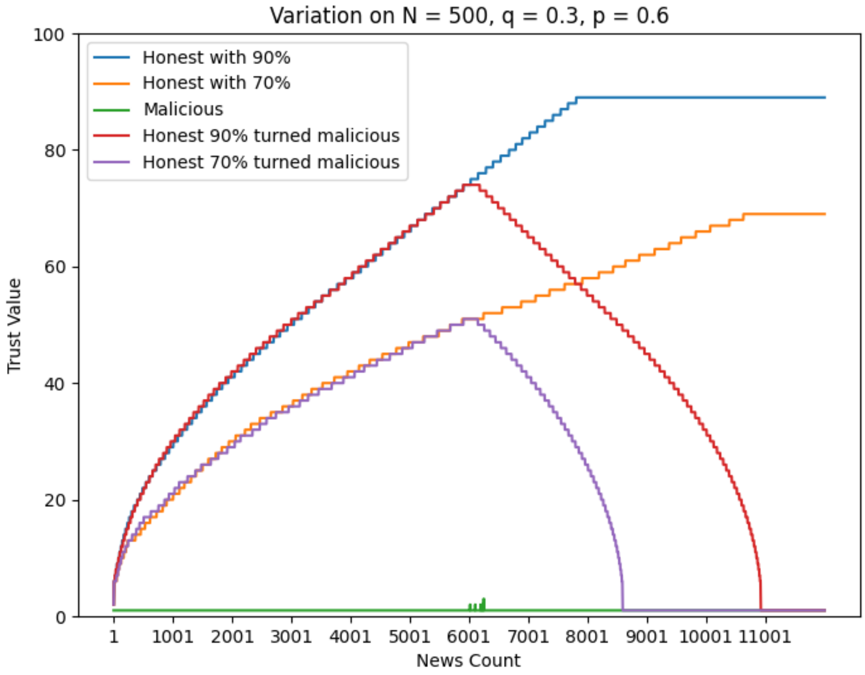


Number of correctly classified news are 12000 / 12000

3.2 Robustness of the System

It is to be noted that the constraints established in 1.2.6, hold for when the system first comes into action. Later when the trustworthiness scores of the voters have been updated, the system is more robust to attack by malicious parties as the votes are weighed by the trustworthiness scores of the voters and the trustworthiness scores of the voters are updated as per their responses. The simulation setup was as follows:

1. p = 0.6
2. q = 0.3
3. The malicious voters always voted incorrectly since the start of the simulation but starting at halftime, 25 percent of the honest voters also starting voting incorrectly. So at starting at halftime, there were 55 (25 + 30) percent of malicious voters. Additionally the honest voters who were still honest also didn't have perfect accuracies(some had an accuracy of 70 percent and some had an accuracy of 90 percent.



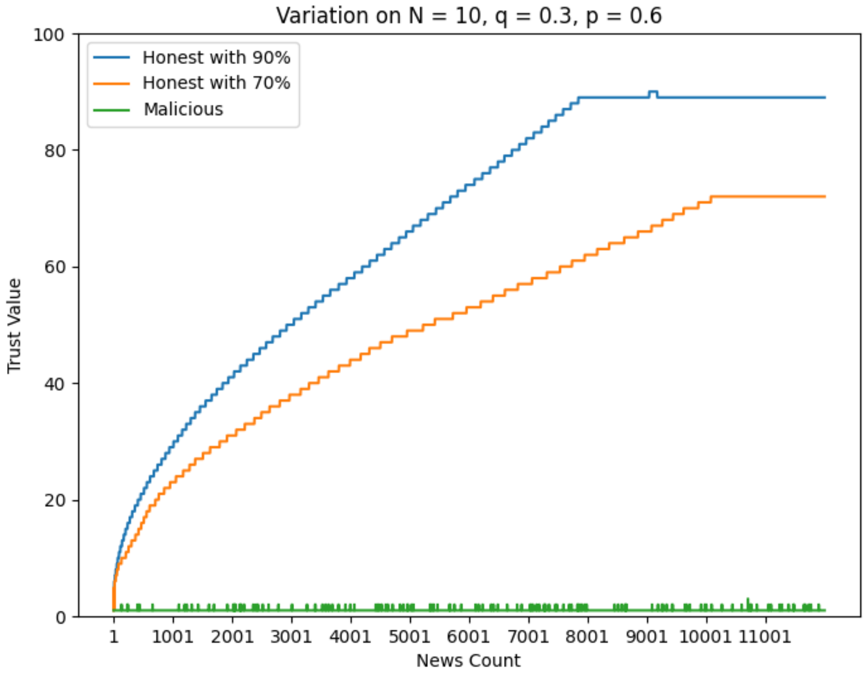
Number of correctly classified news are 11993 / 12000

As seen in the above graph, the system is able to accurately estimate the trustworthiness well even when the percentage of malicious voters exceeds 50 percent. As we can see by the end of the simulation, the trustworthiness of the honest voters who turned malicious falls to 1 and the trustworthiness of the malicious voters remain almost 1 throughout the simulation. Thus this shows that the robustness of the system increases as time progresses and the system is even capable of avenging 51 percent attack.

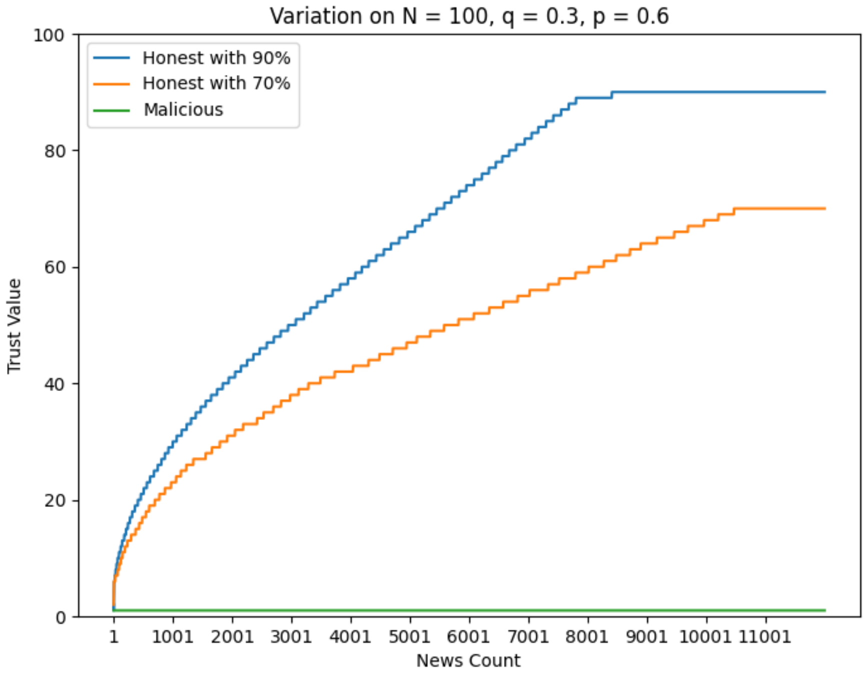
3.3 Varying the Total Number of Voters

Now we test the scalability of our system. We set the configuration parameters as follows:

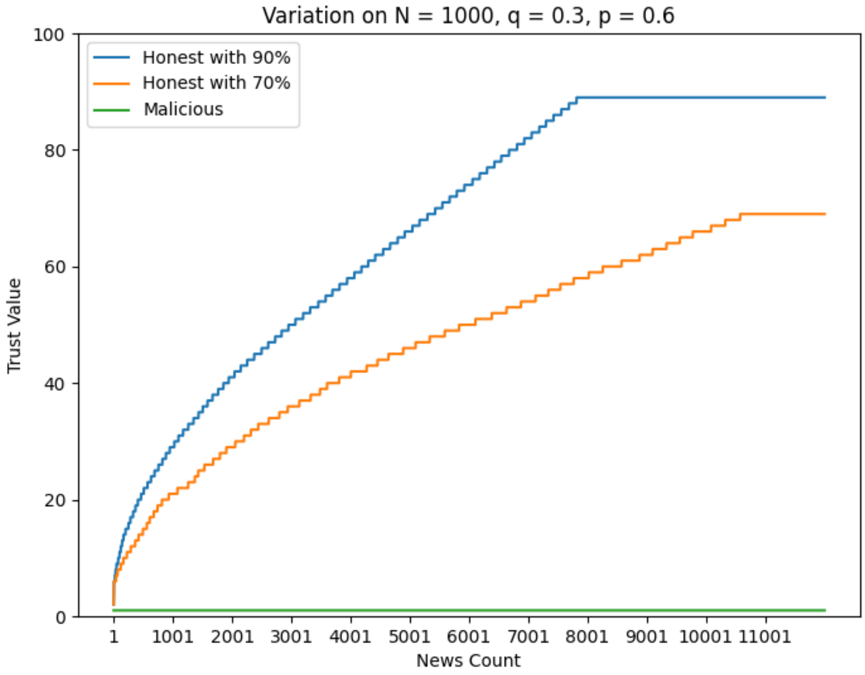
1. p = 60
2. q = 30
3. N for all the values in 10, 100, 1000, 5000 to check the scalability of the system



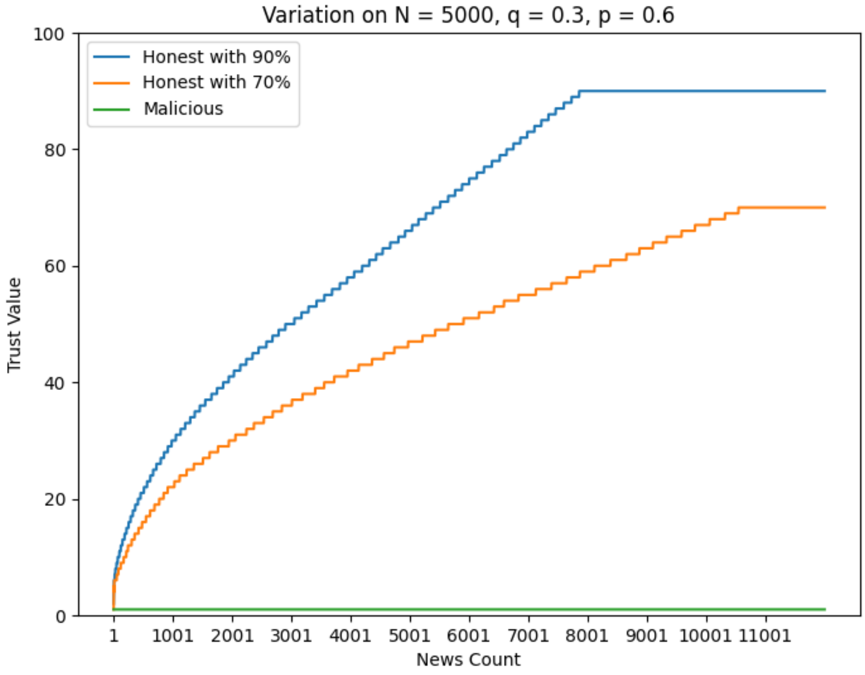
Number of correctly classified news are 11866 / 12000



Number of correctly classified news are 12000 / 12000



Number of correctly classified news are 12000 / 12000



Number of correctly classified news are 12000 / 12000

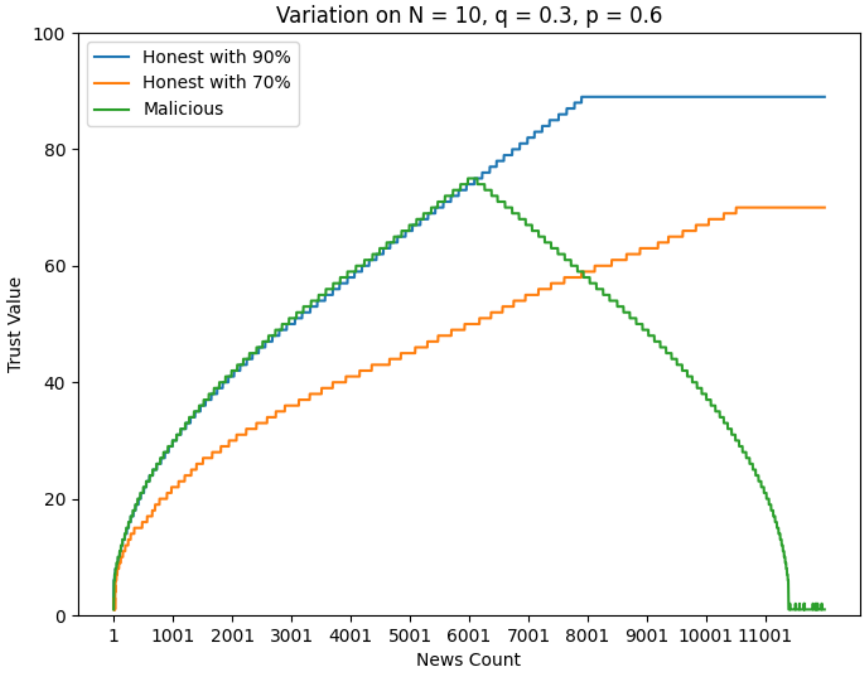
As we can see in the above graphs, our system estimates the trustworthiness of all of the voter accurately as thus is scalable. Also it is to noted that **for all of the values of N, the trustworthiness scores of all of the voters saturate at the same number of news thus highlighting how scalable our system is**. It is NOT the case that when the number of voters increase, the system has a hard time estimating the trustworthiness of voters. Rather, the system is able to estimate the trustworthiness of voters at the same rate irrespective of the total number of voters.

3.4 Alternate Strategy for Malicious Voters

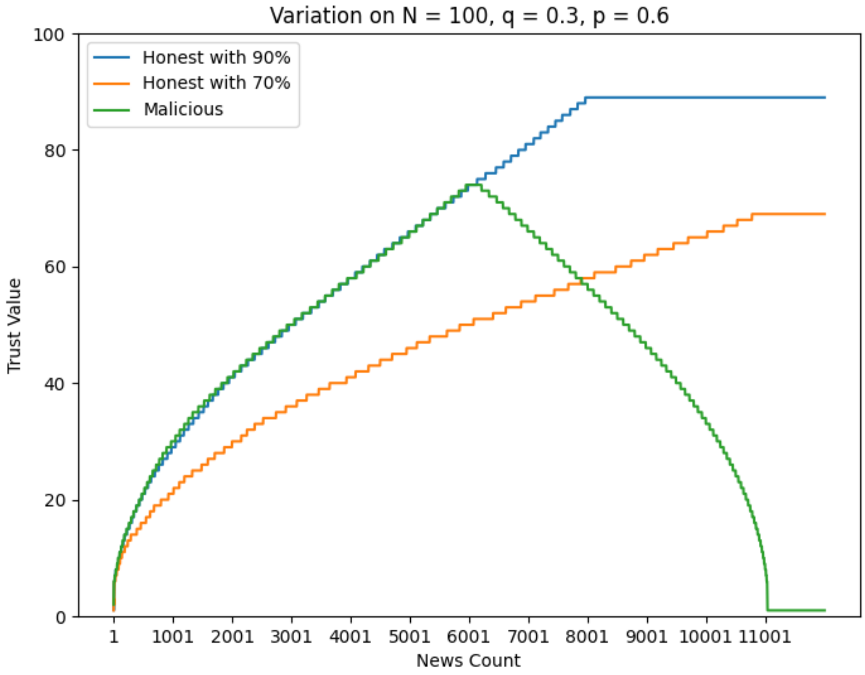
Looking at the above results, it is evident that if a malicious voter always votes incorrectly, his trurtthworthiness score would remain one and he wont be able to attack the system. But what would happen if a malicious voter votes correctly for quite a long period of time and then once the malicious voters get high trustworthiness scores, they would start voting incorrectly and try to reduce the reliability of the system. So, we simulated this strategy for the malicious voters and these are the results we obtained.

The parameters were set as follows

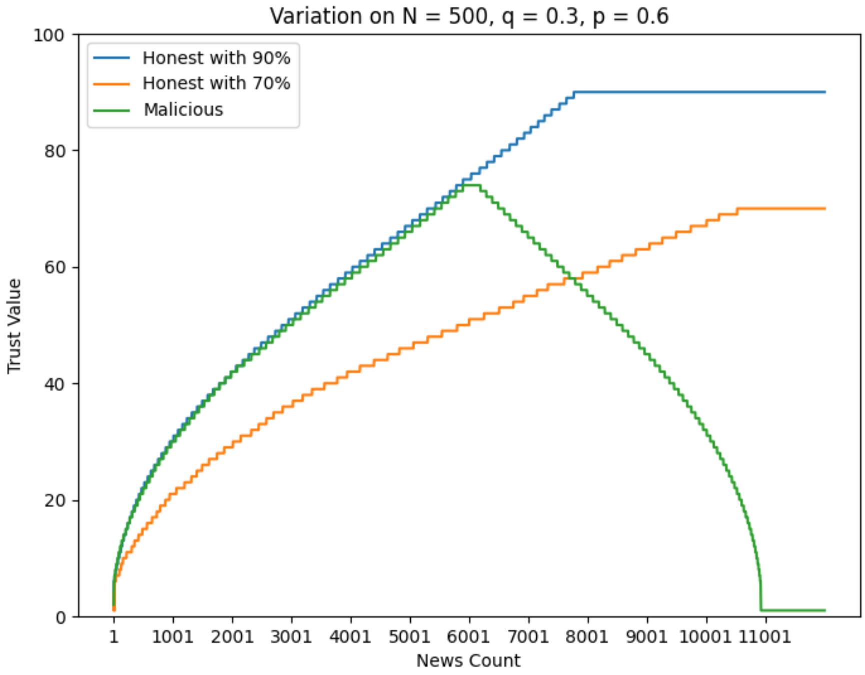
1. p = 60
2. q = 30
3. N for all the values in 10, 100, 500, 1000 to check the resistance of the system to the attack of malicious voters for varying number of voters.
4. For the first half of the simulation, the malicious voters voted correctly and they voted incorrectly in the second half



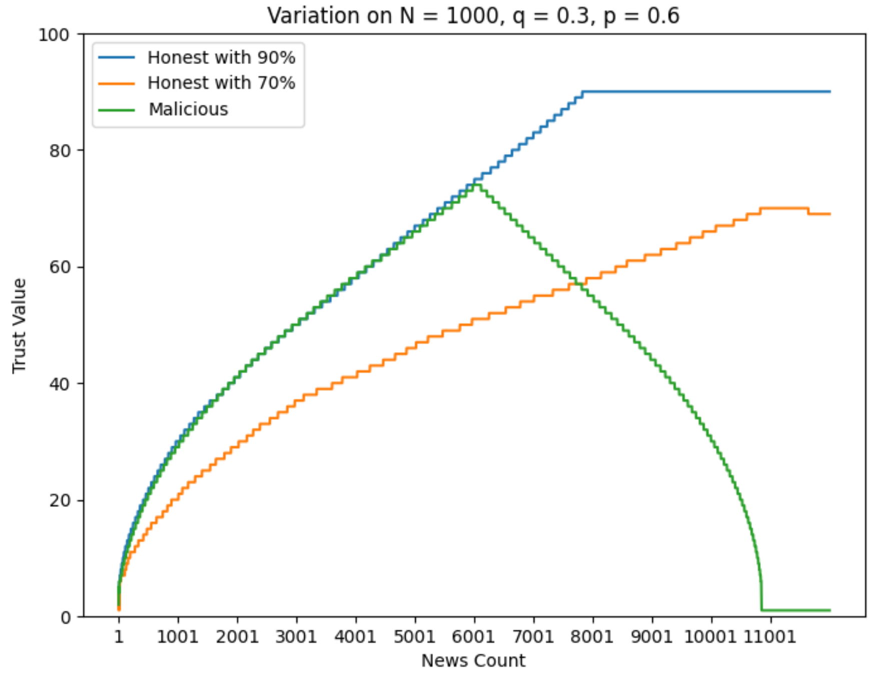
Number of correctly classified news are 11728 / 12000



Number of correctly classified news are 11918 / 12000



Number of correctly classified news are 12000 / 12000



Number of correctly classified news are 12000 / 12000

As we can see in the above graphs, our system is resistant to this attack too. When the malicious voters start voting incorrectly, their rating falls and by the end of the simulation it reaches the lowest possible score (1). Also it is to be noted that the trustworthiness score estimation of the honest voters is not affected by the attack by the malicious voters. It continues to increase or remain the same as it had been in the case where there were no malicious voters thus highlighting the **scalability** of the system.