Online Dating Applications: Combatting Catfishing using Reputation System Design

John Tucker and Halle Clottey

December 19, 2021

1 Introduction

Catfishing is a term that defines when someone intentionally misrepresents themselves online in order to deceive others. It has become significantly problematic in the last few years. According to Psychology Today, approximately 23 percent of women admitted that they had perpetrated catfishing in a study that sampled 917 women in early 2021 [Ph.21]. 38 percent of men in a sample of 190 men confessed that they had perpetrated catfishing, which is remarkably higher than the percentage of women. Even though both men and women fall prey to catfishing, women are more likely to be victims, making online dating incredibly difficult for them.

Some people may out-right catfish (completly misrepresent themselves), but others may just misrepresent themselves a little bit. According to VOX, a group of computer science PhD students released an application called Lightricks in 2013 that allowed ordinary people to edit photos of their faces [Jen19]. Within the span of two years, their company generated about $18 million in revenue from the 4.5 million downloads of Facetune. Since its founding, Lightricks has received at least $70 million in funding, and in 2017, Facetune was Apple's most popular paid app and has been growing profitably ever since.

This drastic growth shows the pressure many individuals have in altering their faces and/or bodies to appear more "attractive" on these dating applications. Some even go as far as to morph their entire physical self by manipulating the shade of their skin tones, contorting the size and shape of their facial features and various body parts, as well as many other distortions. Many do so to seek validity, regain confidence, and mask their insecurities at the expense of those they form close bonds with online.

1.1 What other dating apps are doing to combat catfishing

Bumble, a dating app where women make the first move, requires users to upload an image of themselves in one of 100 poses suggested by the app; this picture is then vetted by Bumble moderators to ensure that users are indeed who they say they are. Other lesser known dating apps also incorporate tactics to deter users from catfishing. Badoo, for instance, allows users to change their settings to only see verified profiles, meaning that potential catfishers won't be seen as frequently. Another application called Coy uses video footage to verify users' identities. Most dating apps also have a reputation system where users can report others. We will be proposing our own reputation system in this paper

1.2 Our Proposition

We seek to create a dating app that matches people on dates in a series of rounds. Each round, a person is matched with one other person, and we assume that they can go on a date. The app also has a way to report catfishing after a date (the profile was not representative of their true self). This report (along with any previous reports) will be considered by the reputation system with a possibility that the person reported for catfishing will be kicked out the next round (or the next couple of rounds, depending on their history).

Note: This algorithm could be useful in many different contexts. For example, suppose we are trying to match researchers together for collaboration on a research paper. They may misreport their expertise on a subject just so they can collaborate with an expert in the field.

1.3 Reputation Design Systems

In his chapter on Reputation Systems, David Parkes defines a reputation system as a framework that ensures reputable agents in the market [Par21a]. It automates "word-of-mouth" feedback which is then collected and made available to users; this feedback ultimately helps guide users' decisions about whether or not to engage in a particular action. A successful reputation system can be used to rapidly identify bad actors and provide incentives to promote good behavior. For this paper in particular, we will explore ways in which we can combat catfishing by building a system of trust that guards against extreme levels of catfishing.

2 Our Process

Variable definitions:
x_score: agent x's actual score
x_report: what agent x reported on their profile

2.1 Agent score

Each agent has a score 0 through 1, which represents some desirable attribute (such as attractiveness). We know that a person cannot be prescribed to a one dimensional figure and that using one attribute is an oversimplification, but the interesting parts of our results don't have anything to do with how an agent is attributed.

2.2 Agent report/profile

Every round, an agent makes a report of what their score is. This report may be their score (if it's truthful) or they may report a higher score. Report = agent_score + amt_lying. For example, this may represent how much photo shopping a person does.

2.3 Matching Algorithm

One downfall of our dating matching algorithm is that it assumes that the people participating are cisgender and heterosexual. We separate everyone into two groups (men and women) and each man will be matched with one woman. Each person will only have one date. Preference scores are based on potential dates' reports (not scores, because no one can tell what they are actually like on a profile).

The matching algorithm we'll use is Deferred action matching algorithm. Because people are one dimensional, it doesn't matter if it's women-proposing or man-proposing, because the most attractive man will be matched with the most attractive woman, second most attractive man will be matched with the second most attractive woman, etc. This isn't an interesting matching algorithm, but again, the interesting parts of our results don't have anything to do with the matching algorithm.

Once everyone is matched, according to everyone's reports, each person has a utility for the date.

2.4 Utility function definition

- Suppose person a and person b go on a date.
- Person a's utility is: b_score - opportunity_scale * opportunity_cost

where **opportunity_cost** = the opportunity cost if b did not lie. This will be 0 if person b did not lie. We subtract the opportunity cost here because person a could have had a better date if person b was truthful. Thus this utility function represents the utility a person gets from the actual date subtracted by the opportunity cost if their date didn't lie.

More specifically, we define opportunity_cost to be opportunity_scale *(report_a - b_score). An agent can expect to go on a date with someone who has a similar report as them (because the x^{th} most attractive man will be paired with the x^{th} most attractive woman). So the opportunity cost is the difference between what they could have expected if every person of the opposite sex was relatively truthful and the date they actually got.

If we don't include an opportunity_scale, it's very possible that the utility of an agent for going on a date could be negative. This may make sense in some cases (like if the date was really bad compared to the person they could have dated). However, it seems to make more sense that a person's utility for going on a date won't be negative for most of the time (though it still could be if they were better off never going on the date).

As a result, utility of a = b_score - opportunity_cost = b_score - opportunity_scale * (report_a - b_score)

Let's look at an example to illustrate this. Suppose person a and b go on a date. Let opportunity_scale = 1, a_score = .7, a_report = .8 and b_score = .5 and b_report = .8

- Utility to a: .5 - (.8 - .5) = .2

- Utility to b: .7 - (.8 - .7) = .6

Here, player b lied more than player a and player a has a higher score than player b. Because of this, player a has a lower utility than player b.

2.5 Reporting

Each round, a person has some chance of being reported.

Probability user gets reported = "base_line_report" + report_probability * (agent_report - agent_score)

Explanation:
There is always going to be misreports of catfishing in which a catfish-reporter's perception of the difference between the profile and the actual person is more than it actually is (there may also be spiteful misreporting if the date went back). So, just because a person gets a report, doesn't necessarily mean they are catfishing to a great degree. To accommodate this, we have a "base_line_report" variable, which is the baseline probability that a user is reported (probably would be within the range .01 through .25) The more a user lies, the more likely they are going to be reported for catfishing. In our model, we represented this as report_probability * (agent_report - agent_score). The higher the report_probability, the more likely lying will get someone reported.

2.6 Reputation system

Each player has a bad_reputation_score (between 0 and amt_of_rounds). The greater the bad_reputation_score, the worse the player's reputation is.

Every time an agent gets reported, their bad_reputation_score increases by 1.

An agent has a chance to prove their reputation through "forgive_difficulty". For every forgive_difficulty amount of rounds, the player's reputation score decreases by 1.

If someone is kicked out for a round, they cannot be reported (or not reported). In the reputation records, we'll say (for their benefit) that they were not reported for that round. This is because we assume that their behavior was fixed (at least a little bit) from them getting kicked out.

Reputation system kicks agents out: A person has a harshness*(bad_reputation_score / amt_rounds) probability of being kicked out in the next round. Harshness represents how harsh the reputation system is—the more harsh, the more likely someone is going to be kicked out. This formulation can be thought of as harshness * (proportion of rounds player has been reported over all of the rounds).

2.7 Incentives and strategies

- **Agent objective**: To increase their own individual utility.

- **Reputation System objective**: To increase the overall utility (the sum of all players' utility).

		Agent	
		Truthful Reporting	Non-Truthful Reporting
Reputation System	Agent is In	$(3, 3)$	$(-1, 5)$
	Agent is Kicked Out	$(0, 0)$	$(0, 0)$

Table 1: Payoff Matrix.

- **Agent Incentives**: When an agent lies, they get matched with someone who has a better report than if they didn't lie. Because our reputation system punishes people who catfish a lot, the report is positively correlated with a person's score (we'll see this in the results section). Because of this, when an agent lies, they get a higher immediate reward than if they were truthful. However, they don't want to lie too much because then they'll be kicked out.
- **Reputation incentives**: The reputation system wants to punish agents who catfish a lot by kicking them out of the next round in hopes of fixing their behavior. However, they don't want to be too harsh because then a lot of people will be kicked out and the overall utility will decrease (if an agent is kicked out, they cannot contribute to the overall utility).

The market we have created can be seen as a collection of **noisy iterated prisoner's dilemma games**. There are amt_agents different iterated prisoner dilemma games; each game is between a different agent and the singular reputation system, and each iteration is a game (so there are amt_rounds iterations). However, this iterated prisoner's dilemma is fundamentally different than the iterated prisoner's dilemma we have studied thus far. One reason is because the agent has no way of "punishing" the reputation system (without also harming themselves). In general, the reputation system has all of the power (they can kick agents out), and the agents are just trying to lie as much as they can get away with (without being kicked out frequently).

The overall utility for a given round is the sum of the system's payoff for all of the amt_agents games for the round.

An (approximate) example payoff matrix for each round is defined at the top of the page.

There are many other variables that may affect this payoff matrix:

- What proportion of other agents lied in a given round? If a lot of people lied, being in the system may not actually grant an agent that much utility.
- Whether the overall utility will be increased slightly or decreased by an agent whose report was very different from their score depends on the round. Sometimes this non-truthful agent may cause negative utility to their date (if opportunity cost is high enough compared to their score) and sometimes it may be slightly positive (if opportunity cost is not more than their score). We'll assume that an agent who lies a lot decreases the overall utility.

- Agent didn't lie a lot, reputation system doesn't kicks agent out:

  The agent gets some utility (less than if they lied). The reputation system has the most utility they could possibly have because it's best for overall utility if the agent is truthful.
- Agent lied a lot, reputation system doesn't kicks agent out:

  The overall utility will be decreased (according to our reasonable assumption described above). The agent gets a lot of utility because when they lie, they are more likely to matched with a higher score person (as explained above).
- Agent didn't lie a lot, reputation kicks agent out:

  The agent is not part of the system so neither the overall utility nor the agent's utility increases.
- Agent lied a lot, reputation kicks agent out:

  The agent is not part of the system so neither the overall utility nor the agent's utility increases.

One important feature of how we formulated our algorithm is that there may be misresports (and thus accidental kick-outs). Thus, this setup is analogous to the **noisy iterated prisoner's dilemma**.

2.8 If the system didn't have any noise (and the payoff matrix was exactly correct)

The only nash equilibrium is (agent is out, non-truthful reporting). This is not ideal for either agent. (agent is in, truthful reporting) is an enforceable action profile (and the only one) because it has higher utility for both the agent and the reputation system. According to the nash-threat folk theorem (Theorem 4.30), there must be a strategy for the reputation system and for the agent such that it is a subgame-perfect equilibrium to play the enforceable action profile (agent is in, truthful reporting) [Par21b].

Because it is noisy, a reputation system strategy like grim-reaper wouldn't work because the reputation system may punish indefinitely a player who was mis-reported (which is bad for overall utility). So, we'll develop our own strategy for both the reputation system and the agent and explain our reasoning.

2.9 Reputation system strategy

As described before, the reputation system will kick people out with probability harshness*(bad_reputation_score / amt_rounds). This way, a misreport will not necessarily result in someone being kicked out. However, the more reports they have, it becomes exponentially less likely that they are being mis-reported.

Also, as described before, the reputation system will have a "forgiveness" policy: if an agent has not been reported for forgive_difficulty rounds (e.g. 2 rounds), then the reputation system will improve their reputation. The reputation system wants the agent to participate if they have good behavior, so they want to reward good behavior of agents. This is incredibly similar to the optimistic recovery idea (most specifically, optimistic unchoking) that we discussed in class.

2.10 Agent strategy

The agent is trying to lie as much as they can without being kicked out. Thus, for each round they do not get kicked out, they increase their report by a defined increment (which we'll call agent_report_increment).

Note, it takes two levels for an agent to be kicked out: the agent must be reported and they must be kicked out by the reputation system. It's possible that the agent gets reported a lot but they don't get kicked out (if the harshness level is low).

If they get kicked out that round, it's likely that they are lying too much and may be kicked out again in subsequent rounds. Thus, the agent will decrease their report by some defined increment (which we'll call agent_decrease_report_increment).

We test these two strategies using a simulation (with over 400 lines of code). All of the variables, hyper parameters, and hyper parameters we've discussed are present in the code. We come across very interesting results, and it indeed seems like the agents and reputation system converge to an equilibrium-of-sorts in terms of strategy.

In our code:

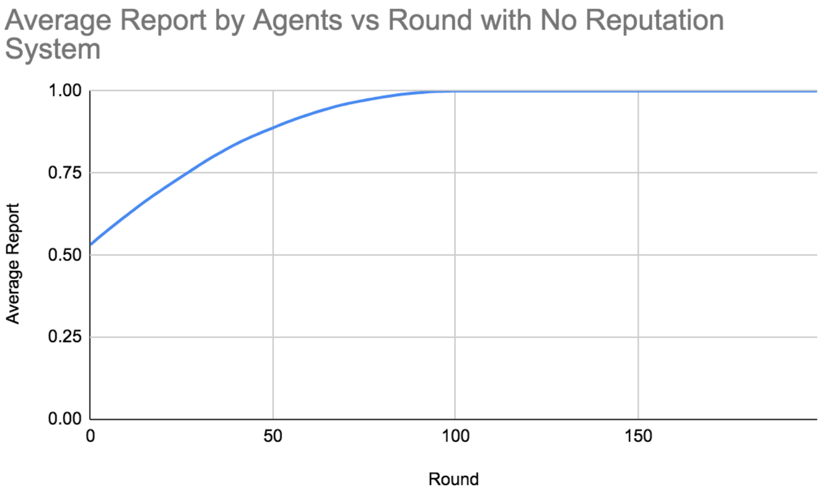
- Each agent is generated with a score that is pulled from Unif(0,1)
- amt_rounds is how many matchings there are for a given set of agents
- There was some variation for a set of rounds, so we also performed multiple "tests" where a test is a collection of rounds. This allowed us to average results across tests. amt_tests is the amount of tests
- prop_truthful is the amount of truthful agents in the pool

3 Our Results

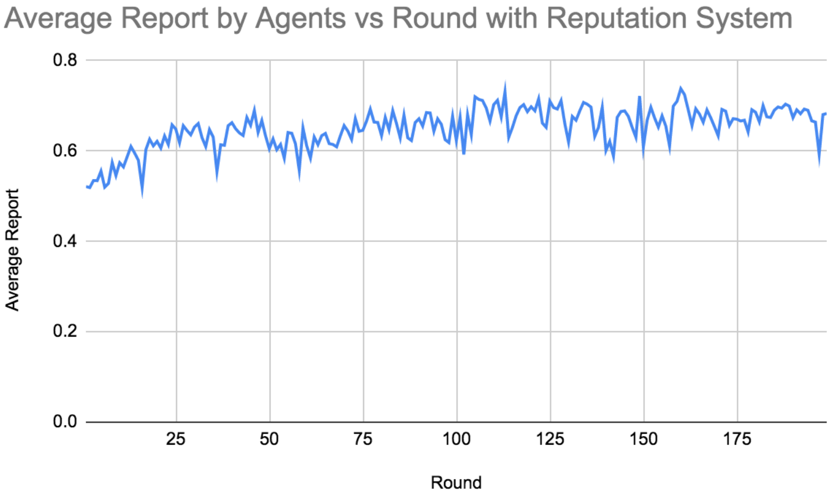
3.1 Reputation System Increases Overall Utility

Let's suppose that everyone in the dating pool has a non-truthful strategy. For each of the the graphs in this section the parameters were:

```
amt_agents = 60
amt_rounds = 200
amt_tests = 1

opportunity_scale = .2

report_probability = .9
forgive_difficulty = 2
baseline_report = .1

harshness = 21
agent_decrease_report_increment = .1
agent_report_increment = .01

prop_truthful = 0
```



Average overall utility for each test: 5023.41
In this first graph, we see that by about the 100th round, everyone just reports ”1” no matter what their actual score is.



Average overall utility for each test: 5119.06
We see that the average report floats around .7 (so it is no longer the most advantageous to always report 1). The reputation system effectively restricts people from always reporting a reputation of 1. Also, the reputation system increased the overall utility.

3.2 Reputation System Harshness

Using the below parameters, we found that harshness level that optimizes overall utility is .21.

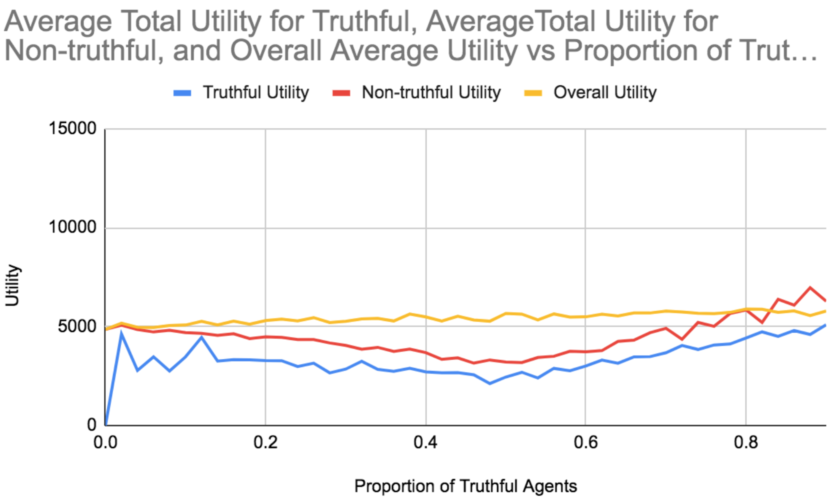
```
amt_agents = 60
amt_rounds = 200
amt_tests = 1

opportunity_scale = .2

report_probability = .9
forgive_difficulty = 2
baseline_report = .1

harshness = 21
 agent_decrease_report_increment = .1
agent_report_increment = .01

prop_truthful = 0
```

3.3 Agent strategy

We now want to show that a non-truthful agent strategy is better than a truthful agent strategy, no matter the ratio of truthful agents to non-truthful agents is. We calculated the average total truthful, non-truthful, and overall utility. If the proportion of truthful > 50, then that means there are more truthful agents so the truthful agents will have more total utility, naturally. So, to accommodate this, we increase the average total utility of the non truthful agents according to the proportion of truthful agents. This works similarly for when proportion of truthful < 50, but vice versa.
then we adjusted the average for either the truthful or non-truthful average
We ran the following simulation with these parameters:

```
amt_agents = 60
amt_rounds = 200
amt_tests = 10

opportunity_scale = .2

report_probability = .9
forgive_difficulty = 2
baseline_report = .1

harshness = 21
agent_decrease_report_increment = .1
agent_report_increment = .01

prop_truthful = x
```

where for prop_truthful = x, we change the x value.

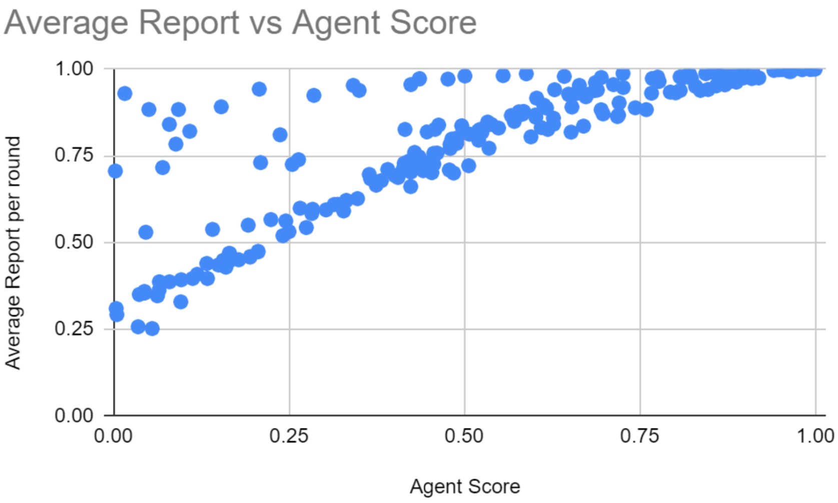


Here we see that no matter the proportion of truthful agents in the pool, non-truthful agents gain more total utility. This means that truthful reporting is not the best strategy.

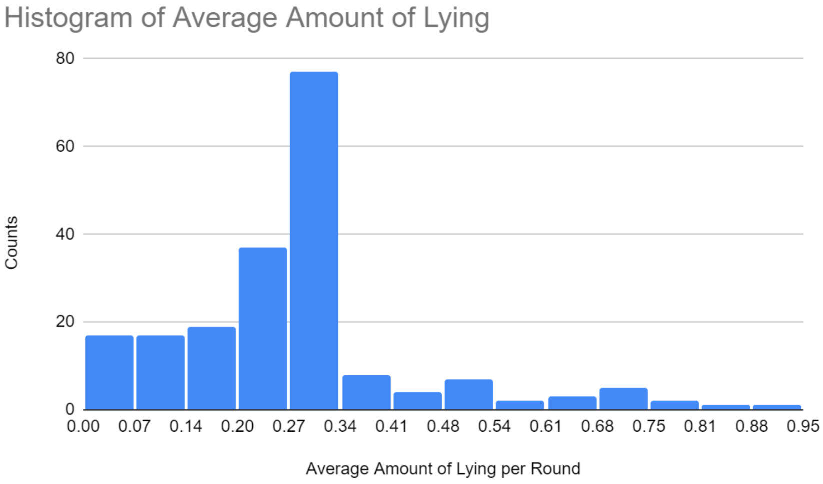
3.4 Optimal Amount of Lying

So far, we've shown that when the reputation system is in place, it's not optimal for an agent to always report 1 and truthful reporting is not optimal (a little lying gives more utility). Now we will look specifically at how much utility is optimal.
For each of the graphs in the section, we ran the simulation with these variables:

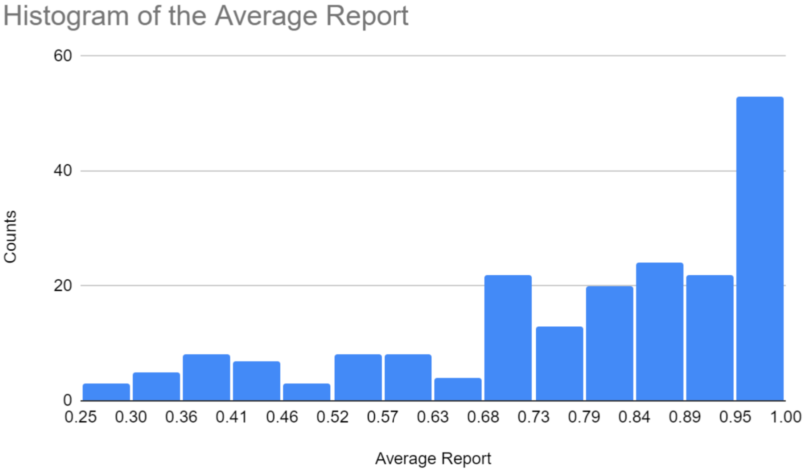
```
amt_agents = 60
amt_rounds = 200
amt_tests = 10

opportunity_scale = .2

report_probability = .9
forgive_difficulty = 2
baseline_report = .1

harshness = 21
agent_decrease_report_increment = .1
agent_report_increment = .01

prop_truthful = 0
```



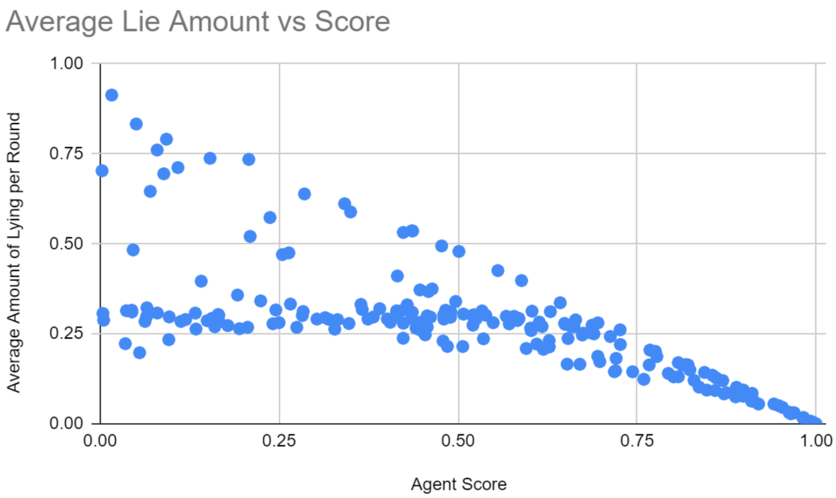
As we can see here, most agents report about .27 higher than their actual score. Some agents report higher than this .27. This makes sense. What we can see from anecdotal, empirical evidence is that many people do in fact over represent themselves on their social medias and dating apps. Also, similarly, people may commonly oversell themselves a little on their resumes (but not so much that they could be called out and fail their interview).



Here, we see that indeed most people lied between .27 and .24.



If people truthfully reported, this histogram would be evenly distributed. However, we see that it's skewed to the left, which is what we'd expect



We see that the amount of lying an agent is negatively correlated with their score. This makes sense because an agent can only lie up until their report is 1. So, if an agent's score is .8, they can only lie a max of .2. The shape of this graph (triangle) is interesting for two reasons. One, it implies that for a given agent score, there's a clear max of lying that will still allow them a reasonable amount of utility (that will allow them not to be kicked out frequently). Two, that there is a minimum amount of lying that will grant them a good utility (a little bit of lying gives good utility).

4 Conclusion

As you can see from our results above, our reputation system effectively discourages extreme catfishing. The agents and the reputation system enter a sort of equilibirum where the agents converge to the exactly right amount of lying that will get them the most utility without being kicked out from being reported. This matching algorithm could be generalized to many different situations and could be helpful to a lot of different dating apps.

References

[Jen19] Rebecca Jennings. Facetune and the internet's endless pursuit of physical perfection. *Vox*, 2019.

[Par21a] David Parkes. Chp. 20: Reputation Systems. 2021.

[Par21b] David Parkes. Chp. 4: Sequential-Move Games. 2021.

[Ph.21] Theresa E. DiDonato Ph.D. How Common is Catfishing? *Psychology Today*, 2021.