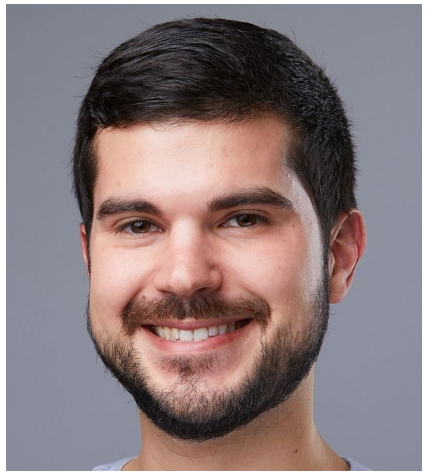
Decentralized & Collaborative AI on Blockchain

Crowdsourcing and Machine Learning Models on the Blockchain

Justin Harris
Bo Waggoner

TEAM



Justin D. Harris
(justin.harris@microsoft.com)
Senior Software Engineer
Microsoft Research, Montreal

Bo Waggoner (bwag@colorado.edu)
Former Post-doc, Microsoft Research, NYC
Assistant professor, University of Colorado, Boulder.

Microsoft

OUTLINE

- Explanation
- Why Blockchain
- ML Models
- Economics
- Demo

THE PROBLEMS

- Difficult to set up AI/ML systems
  - hardware constraints → cost constraints
  - scaling
- Difficult to keep an up-to-date model deployed
- Advancements are centralized
  - datasets are not shared
  - charged per-query

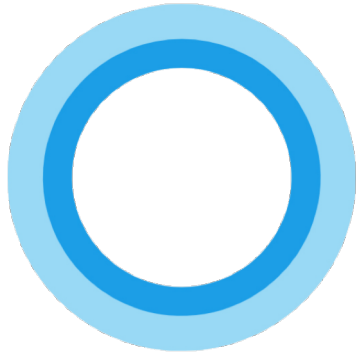
THE SOLUTION

Microsoft

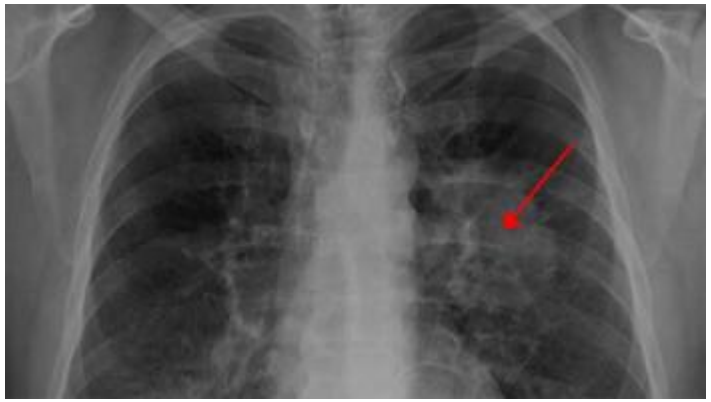
Store models on a blockchain.

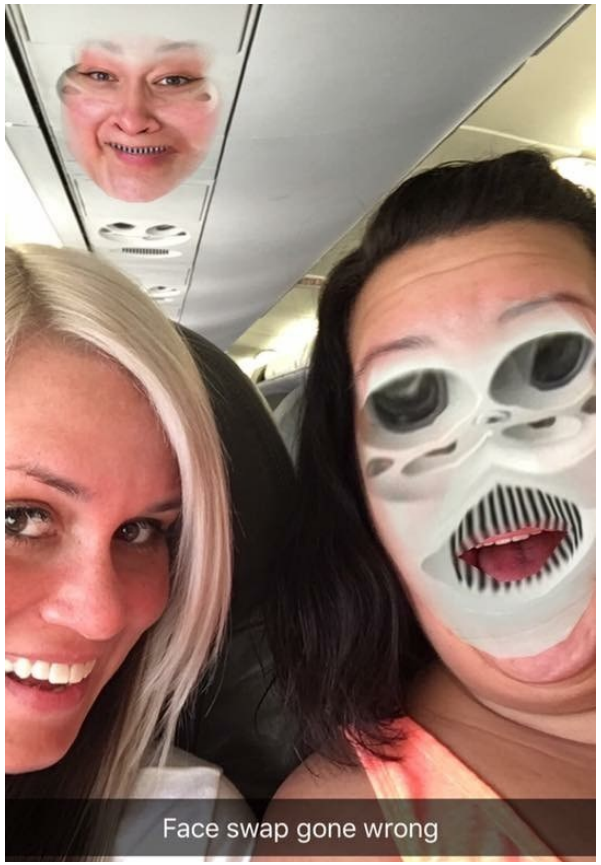
MOTIVATIONS

- Improve models you use as data evolves
- Crowdsourcing: access people + data



Hi. I'm Cortana.

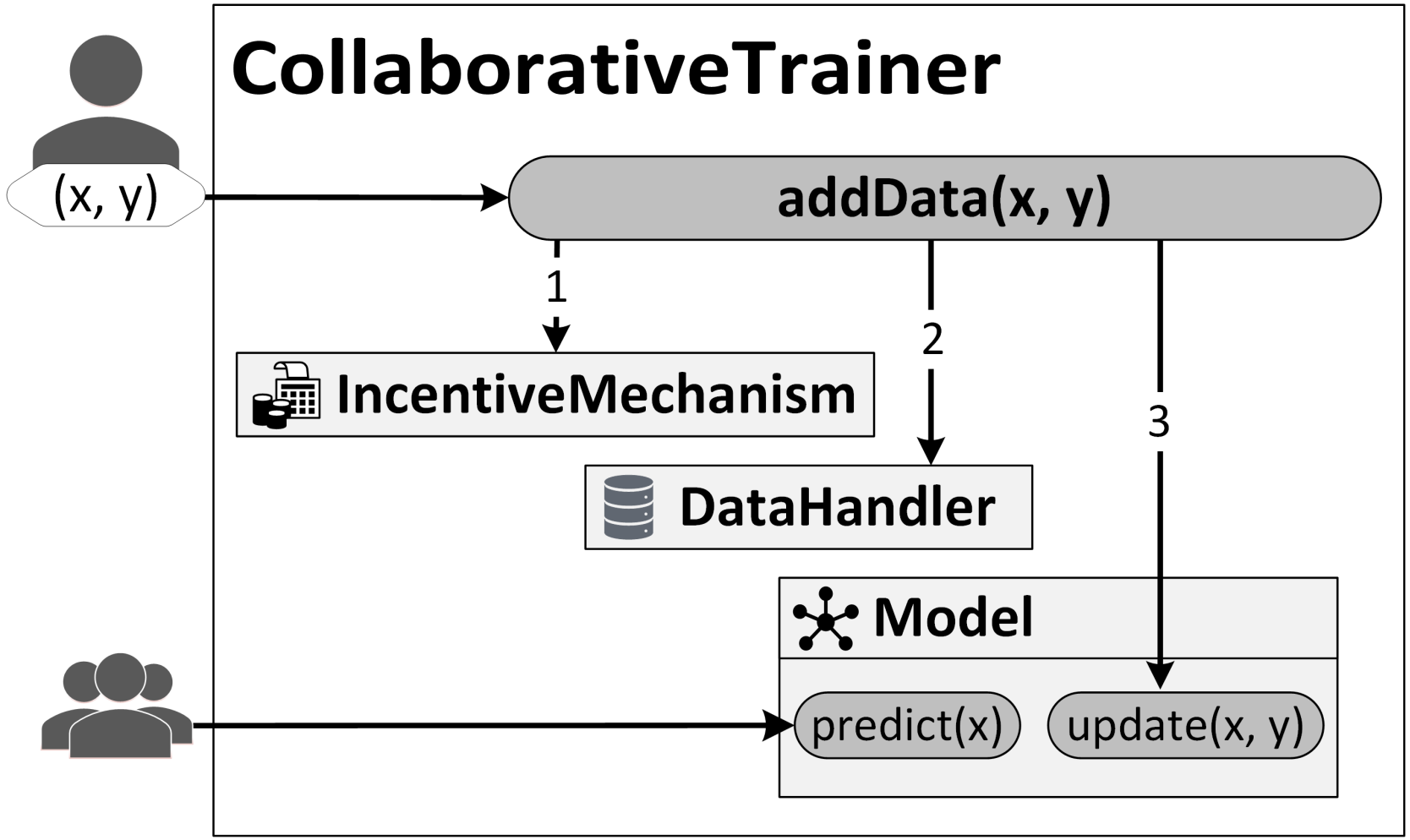




OVERVIEW

Adding data consists of 3 steps.

- Boxes are smart contracts
- Rounded boxes are methods



BLOCKCHAIN

- Public, Persistent, and Decentralized
- Versioned
- Models evolve over time
- Transparency & Trust
- Simplified payment

MODELS

- Supervised machine learning: data with labels
- Minimize gas costs → efficient to train models
  - E.g. Perceptron, Nearest Centroid Classifier, or Naïve Bayes
- Encoding "off-chain" then fine-tuning "on-chain"
  - E.g. Image recognition

Ethereum Gas Costs

Action	Gas Cost	USD[1]
Deploy model contract of Perceptron with 100 weights	3,845,840	$4.71
Add data with 15 words (model agrees)[2]	177,693	$0.22
Add data with 15 words (model disagrees) [2]	249,037	$0.30

[1]Approximate costs in July 2019 with a modest gas price of 4gwei.
[2]Perceptron models are only updated when the model disagrees.

MODELS

- Perceptron:
  - Only update when expected class ≠ predicted class
  - Easy to update: $w(t+1) = w(t) + r \cdot (y - \hat{y}) \cdot x$
    - $w(t)$: weights at time t
    - r : learning rate
    - $\hat{y} = w(t) \cdot x + b$: current classification
    - y : expected classification

- Nearest Centroid Classifier
  - Easy to update moving average: $\text{avg}(t+1) = \frac{x + n \cdot \text{avg}(t)}{t+1}$
  - Enforce normalized: no one can move a centroid too much
  - Encode "off-chain" using a known encoder: tested with 512 dimensions

INCENTIVIZING QUALITY DATA

There are many ways to encourage contributors to submit good quality data.

We analyze several examples in our paper:

1. **Gamified** (non-financial, points + badges like Stackoverflow)
2. Based on established theory in **Prediction Markets**
3. Deposit, Refund, and Take: **Self-Assessment** (screenshot demo later)

INCENTIVIZING QUALITY DATA
GAMIFIED (NON-FINANCIAL)

- Points + badges like Stackoverflow
  - Milestones for number of contributions
  - Points for submitting diverse data
  - Badges for using different labels
  - Extra points for submitting data frequently
- Can be tracked on-chain or off-chain by 3rd parties

INCENTIVIZING QUALITY DATA

BASED ON PREDICTION MARKETS

Prediction Market: Bet or contribute a belief on the outcome of an event.

E.g. Winner of a soccer game or an election.

Here we use ideas from prediction markets to incentivize good data contributions such as in "A Collaborative Mechanism for Crowdsourcing Prediction Problems" (Abernethy et al., NeurIPS 2011) and "A Market Framework for Eliciting Private Data." (Waggoner et al., NeurIPS 2015).

Phases

1) Commitment
2) Participation
3) Reward

INCENTIVIZING QUALITY DATA
BASED ON PREDICTION MARKETS

1) Commitment Phase

- A generous **provider** stakes a **bounty** to be split and rewarded to contributors.
- Now the provider must prove they have a valid **test set** but without revealing all of it yet.[1]
  - Provider shares hashes for portions of their test set: $h_1, h_2, \ldots, h_N$
  - Provider reveals a portion of the test set randomly chosen by a smart contract: $H = \{h_i : 1 \leq i \leq N\}, |H| < N$

[1]Similar to the DanKu Protocol: https://algorithmia.com/research/ml-models-on-blockchain

INCENTIVIZING QUALITY DATA
BASED ON PREDICTION MARKETS

2) Participation Phase

- **Participants** submit one training **data** sample at time along with a small **deposit** of funds.
- The shared **model** is **updated** using the provided data sample.

INCENTIVIZING QUALITY DATA
BASED ON PREDICTION MARKETS

3) Reward Phase

- The **provider reveals** the rest of the **test set** and the smart contract validates that it matches the hashes they originally gave in the Commitment Phase.
- Participants are **rewarded** based on how much their data contribution helped the model **improve** its **accuracy** on the test set:

  change in loss (error rate): $L(h_{t-1}, D) - L(h_t, D)$

  change in accuracy: $\mathrm{A}(h_t, D) - A(h_{t-1}, D)$

INCENTIVIZING QUALITY DATA
BASED ON PREDICTION MARKETS

B = bounty

Amount distributed $\leq B \cdot [L(h_0, D) - L(h_T, D)]$

Let $b_t = 1$ for all t *// balance initially equals stake*
Let list $S = (1, \ldots, T)$ *// list initially contains everyone*
for $i = 1, \ldots, B$ **do**
  for each participant t in S **do**
    Let t' be previous participant in S, or 0 if none.
    Participant t's balance is changed:

$$b_t \leftarrow b_t + L(h_{t'}, D) - L(h_t, D)$$

  Let list $S = (t \in S : b_t \geq 1)$. *// all who can re-stake 1 stay in S*
Each participant t is paid b_t.

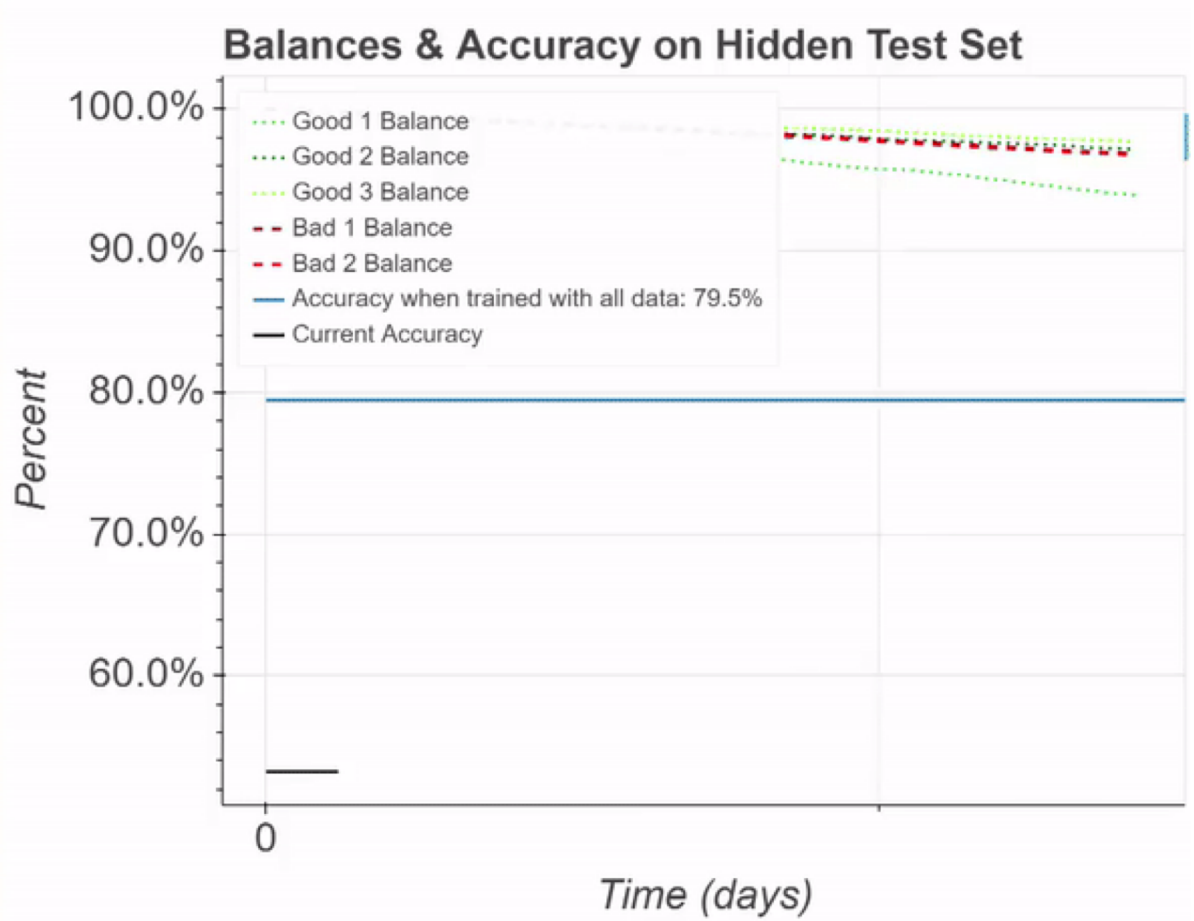
BASED ON PREDICTION MARKETS

Let $b_t = 1$ for all t *// balance initially equals stake*
Let list $S = (1, \ldots, T)$ *// list initially contains everyone*
for $i = 1, \ldots, B$ **do**
  for each participant t in S **do**
    Let t' be previous participant in S, or 0 if none.
    Participant t's balance is changed:

$$b_t \leftarrow b_t + L(h_{t'}, D) - L(h_t, D)$$

  Let list $S = (t \in S : b_t \geq 1)$. *// all who can re-stake 1 stay in S*
Each participant t is paid b_t.

Amount distributed $\leq B \cdot [L(h_0, D) - L(h_T, D)]$



Participants submit data and deposits.

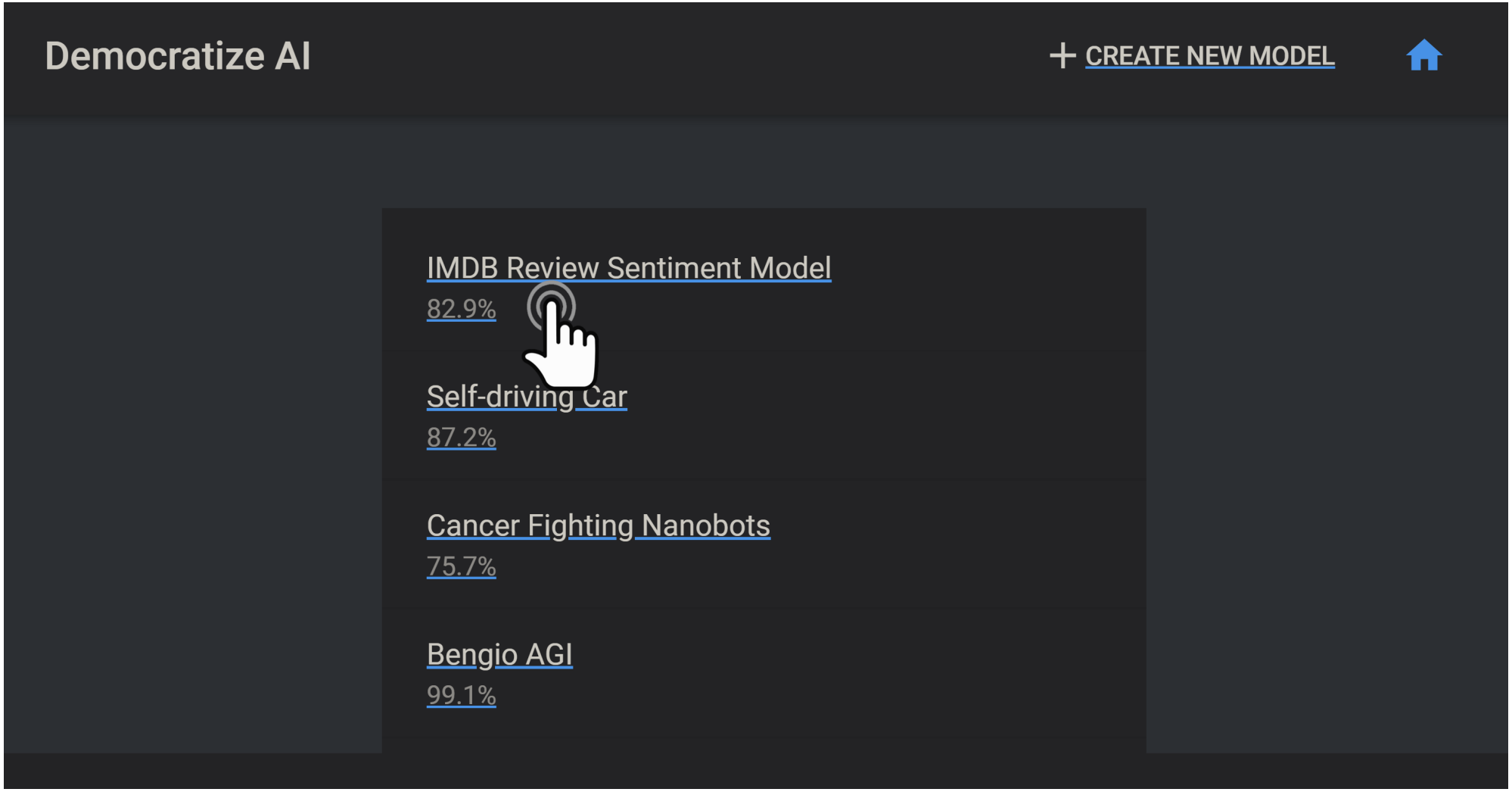
INCENTIVIZING QUALITY DATA

DEPOSIT, REFUND, AND TAKE: SELF-ASSESSMENT

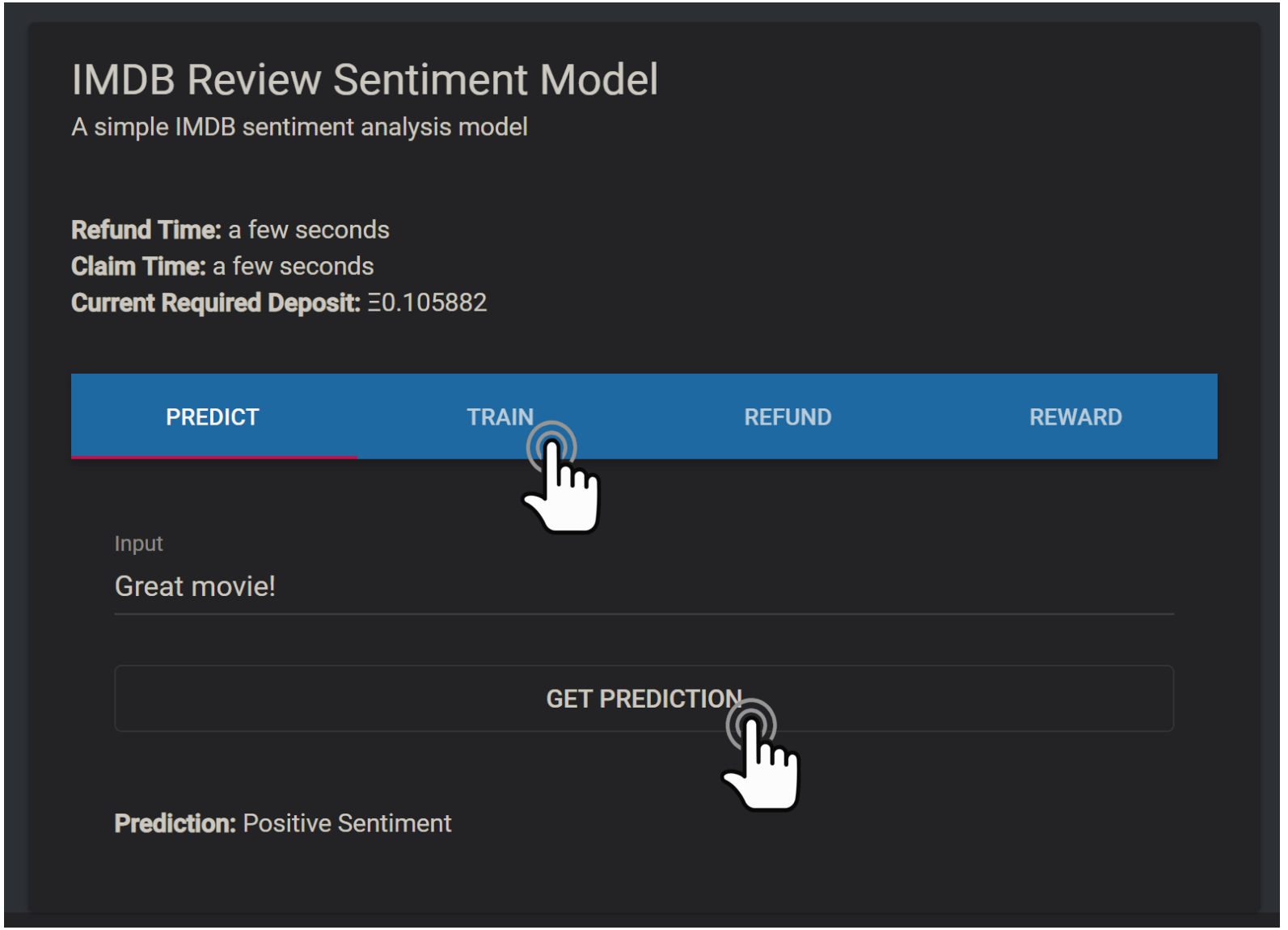
Demo

- Predict
- Deposit
- Refund
- Take

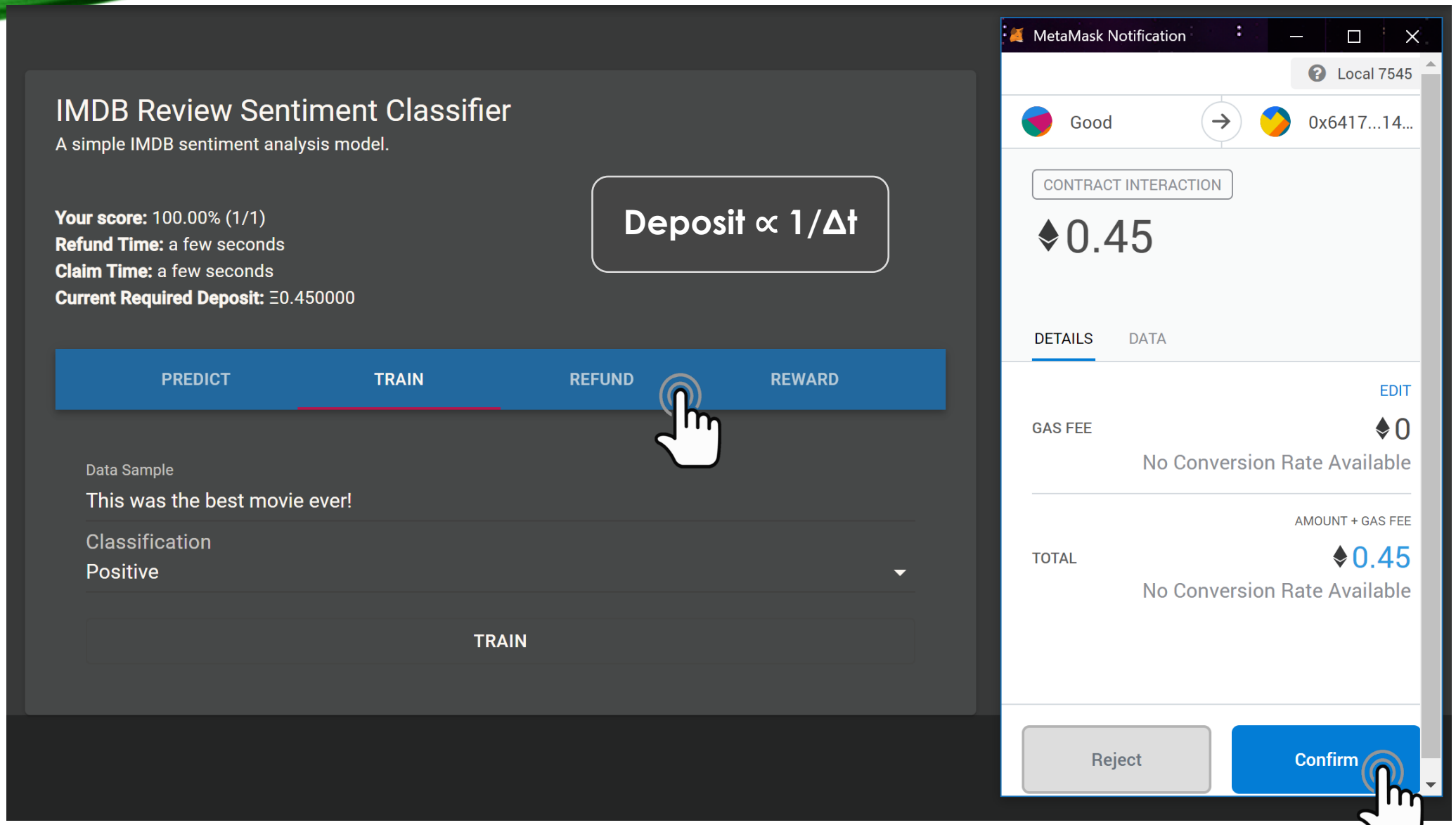
DEMO


Democratize AI
+ CREATE NEW MODEL
IMDB Review Sentiment Model
82.9%
Self-driving Car
87.2%
Cancer Fighting Nanobots
75.7%
Bengio AGI
99.1%

DEMO


IMDB Review Sentiment Model
A simple IMDB sentiment analysis model
Refund Time: a few seconds
Claim Time: a few seconds
Current Required Deposit: Ξ0.105882
PREDICT
TRAIN
REFUND
REWARD
Input
Great movie!
GET PREDICTION
Prediction: Positive Sentiment

DEMO


IMDB Review Sentiment Classifier
A simple IMDB sentiment analysis model.
Your score: 100.00% (1/1)
Refund Time: a few seconds
Claim Time: a few seconds
Current Required Deposit: Ξ0.450000
Deposit ∝ 1/Δt
PREDICT
TRAIN
REFUND
REWARD
Data Sample
This was the best movie ever!
Classification
Positive
TRAIN
MetaMask Notification
Local 7545
Good
0x6417...14...
CONTRACT INTERACTION
0.45
DETAILS
DATA
EDIT
GAS FEE
0
No Conversion Rate Available
AMOUNT + GAS FEE
TOTAL
0.45
No Conversion Rate Available
Reject
Confirm

DEMO

IMDB Review Sentiment Model

A simple IMDB sentiment analysis model

Your score: 100.00% (2/2)
Refund Time: a few seconds
Claim Time: a few seconds
Current Required Deposit: Ξ0.036735

PREDICT | TRAIN | REFUND | REWARD

Data	Classification	Initial Deposit	Date Added	
"this was the best movie i've ever seen"	Positive Sentiment	Ξ0.087805	Wed Nov 21 2018 13:59:34 GMT-0500 (Eastern Standard Time)	Already refunded or completely claimed.
"great film"	Positive Sentiment	Ξ0.900000	Wed Nov 21 2018 14:00:24 GMT-0500 (Eastern Standard Time)	Already refunded or completely claimed.
"best movie ever"	Positive Sentiment	Ξ0.720000	Wed Nov 21 2018 14:00:05 GMT-0500 (Eastern Standard Time)	REFUND Ξ0.720000

DEMO

IMDB Review Sentiment Model

A simple IMDB sentiment analysis model

Refund Time: a few seconds
Claim Time: a few seconds
Current Required Deposit: Ξ0.036364

PREDICT | TRAIN | REFUND | REWARD

Data	Classification	Initial Deposit	Date Added	
"bad movie"	Positive Sentiment	Ξ0.257143	Wed Nov 21 2018 14:21:23 GMT-0500 (Eastern Standard Time)	Classification doesn't match. Got "Negative Sentiment".
"most amazing drama ever"	Negative Sentiment	Ξ0.112500	Wed Nov 21 2018 14:18:03 GMT-0500 (Eastern Standard Time)	Already refunded or completely claimed.

DEMO

IMDB Review Sentiment Model

A simple IMDB sentiment analysis model

Your score: 100.00% (2/2)
Refund Time: a few seconds
Claim Time: a few seconds
Current Required Deposit: Ξ0.036000

PREDICT | TRAIN | REFUND | REWARD

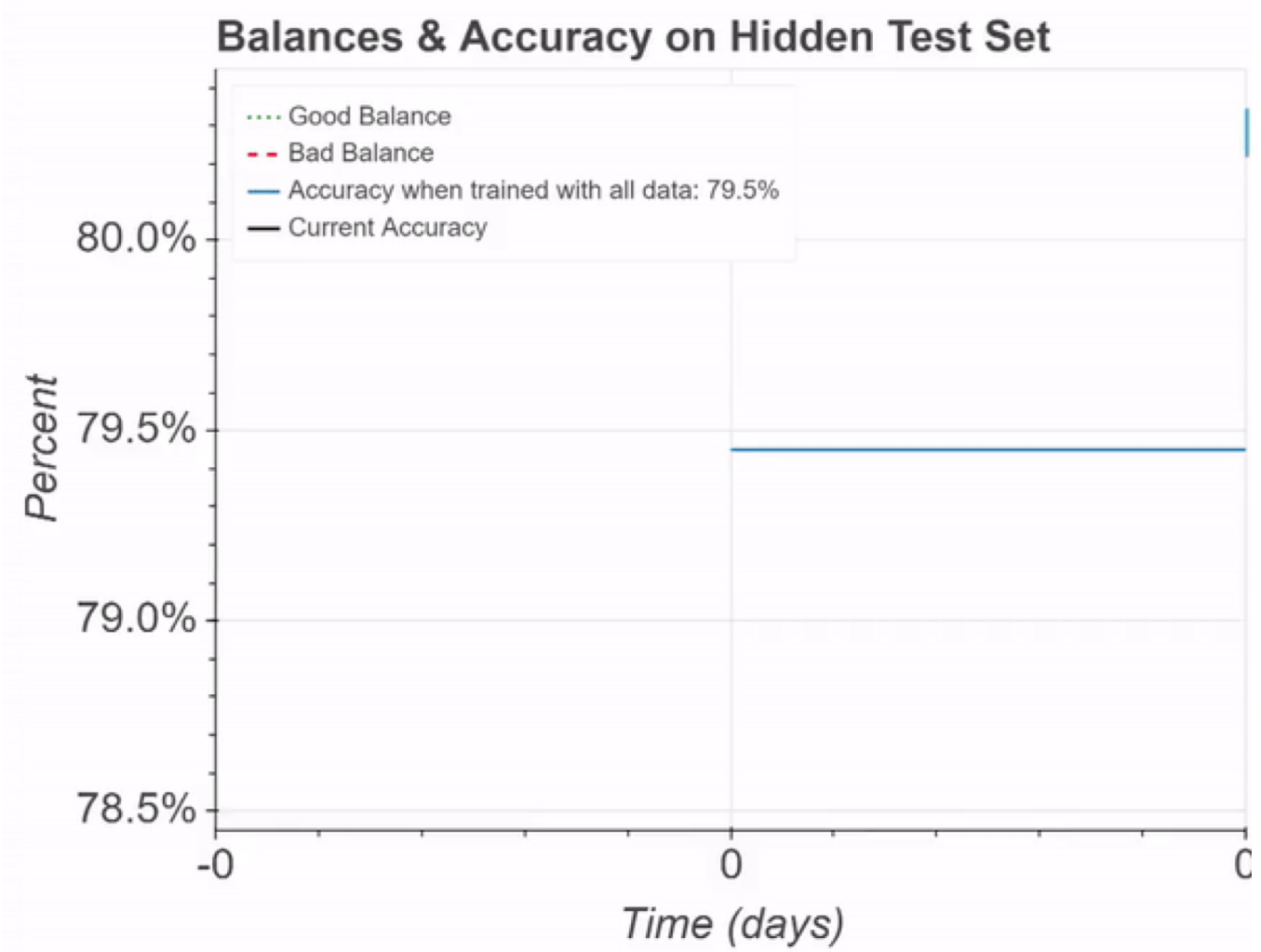
Data	Classification	Initial Deposit	Date Added	
"most amazing drama ever"	Negative Sentiment	Ξ0.112500	Wed Nov 21 2018 14:18:03 GMT-0500 (Eastern Standard Time)	Already refunded or completely claimed.
"bad movie"	Positive Sentiment	Ξ0.257143	Wed Nov 21 2018 14:21:23 GMT-0500 (Eastern Standard Time)	TAKE Ξ0.257143

SIMULATION

"Bad Agent" frequently adds incorrect data.

The model can still maintain accuracy.

Honest contributors can still profit.



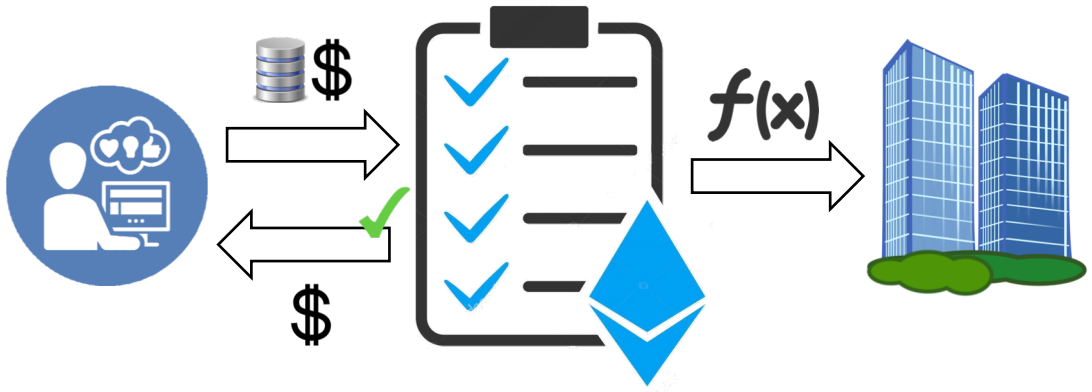
SUMMARY

Goals:

- Free to use models in smart contracts
- Build high quality datasets

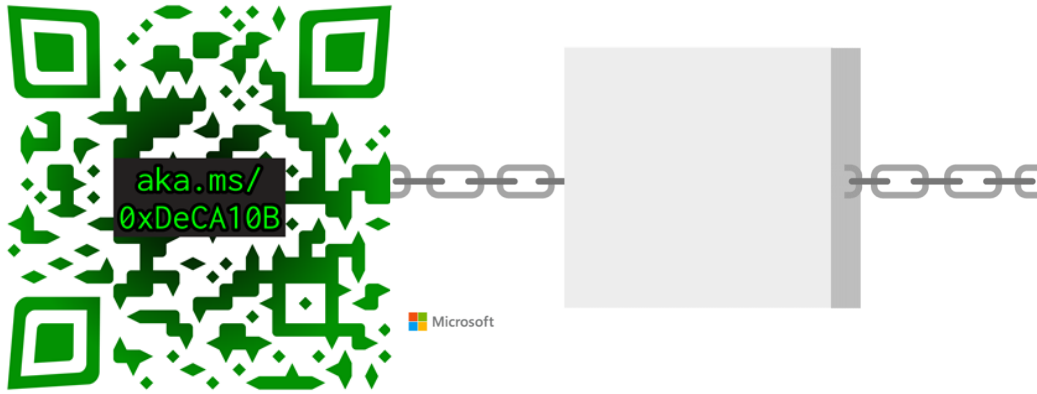
Method:

- Deploy an initial model
- Contributors submit data + deposit
- Contributors can get a reward after submitting good data
- The model remains free to use for inference



NEXT STEPS

- Analyze more incentive mechanisms
- Find the best models to use
- Privacy: handle private data?
- Off-chain models?
- 3rd Party Platforms: free to contribute
- Unsupervised techniques for filtering bad data



blog: https://aka.ms/0xDeCA10B-blog1
https://github.com/microsoft/0xDeCA10B