ORES Preparation

Tom Gülenman

6. Dezember 2018

Disclaimer: No guarantee for the correctness of information / explanations / sources is given.

1 Sources

1.1 ORES direct sources

- ORES: Facilitating re-mediation of Wikipedia's socio-technical problems. (2018)
- Blogpost (https://blog.wikimedia.org/2015/11/30/artificial-intelligence-x-ray-specs/)
- Wikimedia Page (https://www.mediawiki.org/wiki/ORES)
- ORES FAQ (https://www.mediawiki.org/wiki/ORES/FAQ)

1.2 Other possibility to use ORES

- ORES Experiment (https://en.wikipedia.org/wiki/User:Fuzheado/ORES_experiment)

1.3 Current ORES UI (→ work on an improvement here)

- ORES GUI (https://ores.wikimedia.org/ui/)

1.4 Also interesting in that context

- Google (not sure if useful, https://research.google.com/bigpicture/attacking-discrimination-in-ml/)

2 ORES: Facilitating re-mediation of Wikipedia's socio-technical problems.

2.1 Abstract

2.1.1 point of departure:

- Wikipedia's challenges and community's value change, but algorithmic support systems remain stagnant
- Conversation about quality of control and what place algorithms have remains exclusive to few experts

2.1.2 proposed solution: ORES...

- ... to open up socio-technical conversations → and enable theoretical mechanisms of social change

2.2 Introduction

- As every other major platform, that is based on user-generated content, moderation has been relying on artificial intelligence
- But Wikipedia's approach is different in terms of *Fairness*, *Accountability*, *Transparency in ML* ← strong ideological principles of openness of the volunteer community
- **ORES** pushes innovations in openness in ML even further:
  - volunteers *curate labeled training data* from variety of sources for particular prupose
  - commission production of machine classifier
  - make classifier available via API → classifying on criteria like "not / damaging", "good/bad faith", quality scale

2.2.1 Audiences for this work

- ORES is also the implementation of dominant recommendations for algorithmic system builders around transparency and community consent
- "automation in Wikipedia has generally made it more difficult for newcomers"

- "ORES is an advanced algorithmic prediction service for Wikipedians that is designed to democratize the development of work process support tools to a wider audience" → not **directly** solve quality/newcomer problem

2.2.2 Genre: A systems paper and a work study paper

-

2.3 Related Work

2.3.1 The politics of algorithms

- Relevance of algorithms is increasing in social and political life → new questions of fairness and transparency
- In this paper: Wikipedia's algorithmic quality control and socialization practices

2.3.2 Machine prediction in support of open production

State-of-the-art:

- Vandalism detection
  1. Automated bots reverting most obvious vandalism
  2. "vandal fighers": not fully automated bots for less obvious vandalism
  3. Watchlists of experienced Wikipedia editors
- Task routing: SuggestBot routes attention to important, but low quality articles

2.3.3 The Rise and Decline: Wikipedia's socio-technical problems

- Problem context
  - Wikipedia grew exponentially between 2005 and 2007
  - Scaling was a struggle → efficiency of quality and damage control > positive experience of newcomers
  - Prioritizing efficiency by using ML → decline of good-faith newcomers and overall editors

- Countermeasures
  - multiple initiatives, e.g. "The Teahouse", a newcomer help space (meet exp. users)
- Results
  - The dominant quality control systems remained unchanged
  - Most efforts did not show any gains in newcomer retention
  - Experiments have been done with control that highlights good editors who run into trouble

2.4 Design rationale

Wikipedia's problems are systemic (necessity for ML with its conseq.) but with no apparent system-level solutions.

How does Wikipedia function (processes, policies etc.)? → How to use ML to adress problems?

2.4.1 The context: Wikipedia as a genre ecology

- Processes not centrally planned
- "Bots and human-computation tools mediate the meaning of policies and how Wikipedia enacts quality management"

2.4.2 The problem: Wikipedia's problems with automated mediation

Again: trading human newcomer socialization for efficient quality control → decline in retention of new editors.

Genre ecology here $=^{?}$ mediation of policy and the design of software

2.4.3 The complication: Making change in an decentralized ecology

- General interest in balancing quality/efficiency and diversity/welcomingness more effectively exists
- Where are the designers able to make such a change?

2.4.4 The goal: Expanding the margins of the ecology

- The barrier: needing to know about ML classification models for quality control
- Two options to lower the barrier to create quality control:
  1. increase general literacy around ML
  2. minimize need to deeply understand ML ← ORES
- How does ORES do that?
  - Support multiple classifiers at scale
  - Design accessible interfaces to engage with those classifiers
  - engage in basic outreach efforts

2.5 The ORES System

Architectural overview

2.5.1 Conceptual architecture

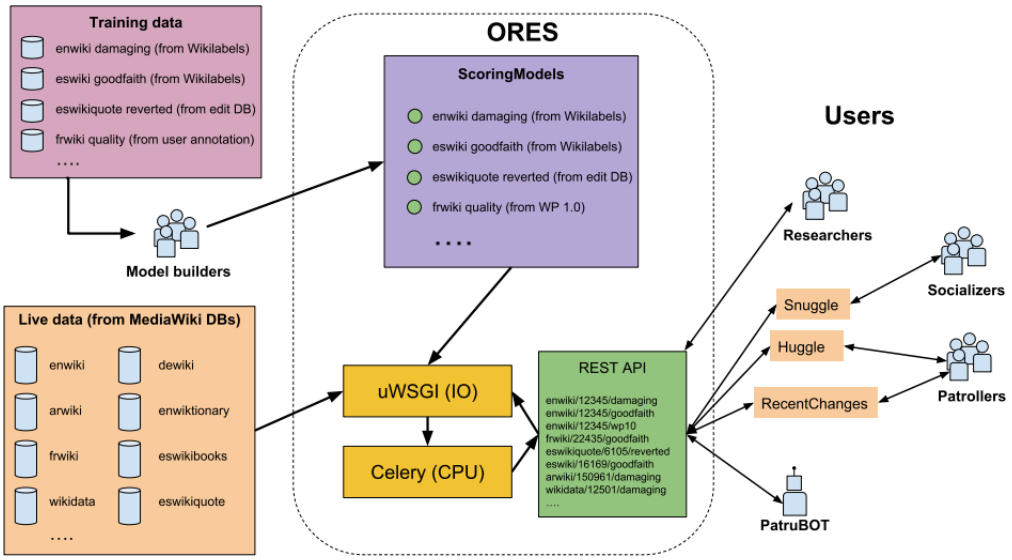
Components:

- Collection of machine classifier models (using varied sources of training data)
  - For Wikipedia: models related to damage-detection, quality assessment and topic-routing
  - But adaptable to wide range of other models

Making models available:

- "Container service"
- Containers = Models referred to as *ScoringModel*s
- *ScoringModel*s contain metadata:
  - When model was trained/tested
  - Which features are necessary for making prediction
  - Predictions are JSON documents

- Access *ScoringModel*s via RESTful HTTP interface and show users JSON documents = predictions



2.5.2 Scaling & robustness

ORES uses distributed computation:

- IO workers (**uwsgi**) ← requests are split between those, necess. data is gathered using external APIs
- Computation workers (**celery**) ← data is then split into job queue

⇒ Efficient use of resources and scalable: IO and CPU workers can be added in multiple datacenters if needed

⇒ Robust (fault tolerant): servers can fail without failing the whole service

2.5.3 Real-time processing

ORES's most common use case: real-time processing of edits, e.g. counter-vandalism withing seconds of when edits are made (tools like Huggle)

- Single score. ORES generates score from scratch if requested in real-time. 1.1 - 1.9 seconds.
- Caching and precaching. Scores of edits that have been recently generated: 20X faster (50ms)

2.5.4 Batch processing

- Wikipedia's bots rely on batch processing → ORES needs to support **sudden, high-intensity querying**
- Splitting IO (gather data) and CPU operations into distinct stages → 5X increase in time for large requests
- Worst case: 1 million revisions in ¡ 24 hours

2.5.5 Empirical access patterns

- ORES supports 78 different models and 37 different language-specific wikis
- Generally 50-125 requests per minute from tools that use ORES predictions
- Sometimes up to 400-500 /min
- Precaching requests: (from graphic) 600-800 /min up to 1.1k /min

2.6 Innovations in Openness

- Keep the system open for review, critique, iteration
- System = flow of data from random samples through model training and evaluation ⇒ enable flexibility in the use of ORES

2.6.1 Collaboratively labeled data

Two ways of gathering labeled data for ORES' models:

1. Found traces
    - Taking sample of edits and labeling them as *reverted_for_damage* if reverted within 48 hours, reverting editor ≠ author, edit not restored
    - Only make use of this when manually labeled data is not available (because *reverted_for_damage* is problematic: no information about good/bad faith, content dispute etc)
    - Other case of found traces: Article quality assessment ("*wp10*")
2. Manual labeling

- Asking Wikipedians judgements on examples from their wikis
- High up-front expense of human labor but ⇒ models that give predictions that make sense to these communities
- Wikipedians are also asked to distinguish btw. *damaging/good* and *good faith/vandalism*
- Result: two different models → Users can search for good-faith mistakes, vandalism or all damaging edits

2.6.2 Threshold optimization

- Operational concerns (examples):
  - Catch "all" vandalism before it is allowed to stick in Wikipedia for long (→ concern around **recall** of a damage prediction model) (recall: catching damaging edits)
  - Review as few edits as possible (→ concern about filter rate)

Important: find threshold of prediction likelihood that optimizes filter-rate at high level of recall.

⇒ Find best trade-off btw efficiency and reliability of filter

= **threshold optimizations**

- Filter-rate: workload of vandal-fighters is reduced by 88% Q1.: are we still talking about the bots here? do they only check 12% of edits now?
- Recall: Catching 75% of (the most) damaging edits
- Likelihood: 0.299 Q2.: ?
- Precision: 0.215 → "How often is it that when an edit is predicted to be "damaging" it actually is?" Q3.: Does this really mean that here, 5 times as many edits are marked as damaging as should be?

2.6.3 Dependency injection and interrogability

ORES is designed to host machine classifiers but is suited to other scoring paradigms as well.

- Dependency injection
  - Serve multiple scores in same request: e.g. "edit type", "damaging" etc. scores;

  - These models depend on features related to edit *diff*
  - ORES combines features required for models and extracts them with their shared dependencies together

    ⇒ **Efficiency** of multiple requests is roughly the same as for one req

- Flexibility
  - Interrogability: exposing how ORES makes predictions; inject own features as user, e.g. what if this same edit was made by an unknown user (Fig.4 "damaging" false 0.938 % → 0.912%)
  - ORES is in that way also used to suggest work to new editors (by asking what change one more image / header / citation etc make)

2.7 Adoption Patterns

ORES: make edit quality predictions available → lower barrier to experimentation

2.7.1 Showcase tools

1. **ScoredRevisions**
   - Javascript-based gadget
   - Submits requests to ORES to score edits
   - Then highlights Edits as "damaging", worth reviewing or not at all in color.
   - Limits: requesting 50-500 edits at once (all the edits on a page!) → 30s-2m; not possible to filter edits to only show damaging f.ex.
2. **The ORES Review Tool**
   - MediaWiki extension in PHP
   - Offline process scoring all recent edits to Wikipedia; storing them in a table for querying and quick access
   - Similar to **ScoredRevisions**, but pre-caching scores → rendering highlights as the page loaded and enabling filtering
   - Successful: around 26k/70k editors manually enabled this feature (April 2017)

2.7.2 Adoption in current tools

Many developers quickly adopted ORES in their tools for counter-vandalism, but also article-quality. E.g. SuggestBot included ORES predictions in their tables of recommendations.

2.7.3 Development of new tools

New tools have been dev. since ORES' release that may not have been dev at all otherwise

- e.g. Redesign of MediaWiki's Special:RecentChanges interface → conclusion of the ORES Review Tool.
- Before ORES Wikipedia was (maybe) the only wiki to use ML to revert vandalism. Sincec ORES release that has changed: e.g. PatruBOT in Spanish Wikipedia
- Wiki Education Foundation: supporting classroom activities that involve editing Wikipedia

2.8 Case studies in reflection

With the first deployment of ORES came also lots of **false-positives** reports from users.

2.8.1 Report mistakes (Wikidata)

- The deployement of the first models for Wikidata was accompanied deployment of the "Report mistakes"-site for users to report **false-positives**
- ORES devs learned in what ways Wikidata editor's understandings differed from the feature extraction process
- The resulting improvements were made public in form of a table

2.8.2 Patrolling/ORES (Italian Wikipedia)

- Similar way: report sites (with categories)
- Example: "ha" (Italian *have*)

2.8.3 PatruBOT (Spanish Wikipedia)

- Another bot reverting edits using ORES prediction
- Not working very well though, currently disabled

2.8.4 Bias against anonymous editors

- Context:
  - At the time using Linear SVM (*) estimators to build classifiers
  - Considering transition towards ensemble stategies like Gradient-Boosting (*) and RandomForest estimators
- Started rec. reports that ORES dmg detection was biased against anonym. editors
- Used injection feature to look at how the prediction models changed if the same edit was made by diff. editor
- Injection analysis confirmed that point (Fig. 7 p.22)
- Improvements were made to the modeling strategy to mitigate the problem ⇒ Feedback proved very valuable

2.8.5 Discussion

- Understandings and models were refined with the help of feedback from the communities
- Spanish Wikipedians are still discussing what false discovery rate would be acceptable, or if any revert is acceptable without human intervention

2.9 Conclusion and Future Work

- ORES was created by a community that has only a fraction of the resources that for-profit organizations as Facebook and Google have
- The deployed technology will not be a black box to users, but offers classifiers to be open to skeptical reinterpretation
- ORES: socio-technical, CSCW approach to fairness, accountability and transparency in ML

2.9.1 "Hearing to speech": lowering barriers to participation

- Goal
  - not "**speaking to be heard**" (build exact technology *we* think is right for Wikipedia)
  - but "**hearing to speech**" ⇒ "What is preventing others from getting involved in this ocnversation?" and lower those barriers
- Values that ORES should help sustain: more complete balance of efficient quality control and newcomer support
- ORES targets early precursors of social change: ecological health and critical reflection

2.9.2 Ecological health

- The technological conversation around quality control represents a substantial shift from post-edit boundary to pre-review training for newcomers

2.9.3 Critical reflection

- Both case studies show that reflection over algorithms in quality control is taking place
- Surprising alternatve uses of ORES have surfaced
- Much concern from Wikipedians has surfaced about a direction of change for quality control
- Wikipedians collaborate to build information about trends in ORES' mistakes

2.9.4 Future work

- Imporved crowd-baseed auditing tools → Future devs: impl. structured means to discuss the predictions of machine models (e.g. report *false positives*)
- Make it easier to query a ORES mistakes DB → users can show that biases and the resulting problems exist and coordinate with each other and even turn the model / ORES off

- "we also see potential in allowing Wikipedians, the denizens of Wikipedia, to freely train, test, and use their own prediction models without our engineering team involved in the process" (p. 26)
- Currently only someone with strong modeling and programming background can deploy models

3 Blogpost: Artificial intelligence service "ORES" gives Wikipedians X-ray specs to see through bad edits (November 30th, 2015)

- New artificial intelligence to help discover damaging edits and "score" the quality of any Wikipedia article
- ...as an open Web service
- Hope: ORES will enable advancements in how we do quality control

3.1 How it works

- Automated edit and article quality classification to everyone via APIs
- https://ores.wmflabs.org/scores/ enwiki / damaging / 620376896
  - Type of Wiki (English Wikipedia)
  - Model
  - Revision ID (found at the top of a page)

  ⇒ API returns results in JSON
- Revision scores can be used in "your" application by calling an available ORES endpoint (→ this link is not available anymore)

3.2 Towards better quality control

- There have been automated tools (Huggle, STiki) and bots (ClueBot NG) to help with damage detection before
- Problem with those: newcomer barriers such as encouraging rejection of new editors' changed (as if made in *bad faith*)

- ORES: newcomer support and training spaces (like *Teahouse* and *Help desk*)

  ⇒ **ORES decouples damage prediction from quality control**

  ⇒ New tools and processes that are efficient and welcoming to new editors

3.3 A feminist inspiration

- "Please exercise extreme caution to avoid ecnoding racism or other biases into an AI scheme."

  Recent research has called attention to increased technological influence on governance and power dynamics

  SW dvpmt in collab. envrmt req. application of many diff. perspectives

  ⇒ "hearing to speech" strategy

3.4 Who's using revision scores

Examples:

- Vandalism fighters like Huggle
- Snuggle uses edit quality scores to direct good-faith newcomeres to appropriate mentoring spaces
- Wiki Education Foundation: surfaces most valuable contributions made by students in an education program

3.5 What's next

- Supporting more wikis
- Edit type classification: categorize edits by the type of work performed (→ edit type model)
- Bias detection: collecting feedback on mistakes and devping strat.s for detecting bias in the models

4 Wikimedia Page: ORES

Only new information from now on...

- Two levels of support for edit quality prediction models:
  - Basic support: Build model on history of edits (bad edits have been reverted, good ones haven't); problem: reverts too many edits (reverted)
  - Advanced support: Ask editors to train ORES which edits are damaging and which are in *goodfaith* (damaging) (goodfaith)
- Curation support: Also predict if new articles will be deleted (draftquality)
- Assessment scale support
  - Large Wikipedias periodically evaluate quality of articles
  - Quality scale: FA, FL, A, GA, B, C, Start, Stub, List, Assessed, Unassessed
    - ⇒ Helps gauge the progress and identify popular articles that are low quality f.ex.
    - ⇒ Article quality model (articlequality)
  - Article quality measured by (examples):
    - Number of section
    - Infobox?
    - Number of references
    - Ref.s using "cite" template?
      - ⇒ Not evaluating the quality of writing, **but** these measured structural characteristics seem to correlate strongly with good writing → models work well in practice
- See page for more info on **How to use ORES**

5 ORES FAQ

- ORES scores: allow humans and machine to work together

  e.g. **patrollers**: human users who help determine whether edits are dmging

- How to use ORES:
  - Built-in: RecentChanges, Watchlist and Contributions pages for supported wikis

    → See Recent changes with filter interface
  - "My" editing activities: find a tool that uses ORES or directly via API (`https://ores.wikimedia.org/v3/scores/enwiki/234234320/damaging`)
  - What tools are avail. to use ORES? (`https://www.mediawiki.org/wiki/ORES/Applications`)

    Categories:
    * Integration with MediaWiki: RecentChanges, Watchlist
    * Counter-vandalism
    * New page patrolling
    * Article quality assessment
    * Task routing

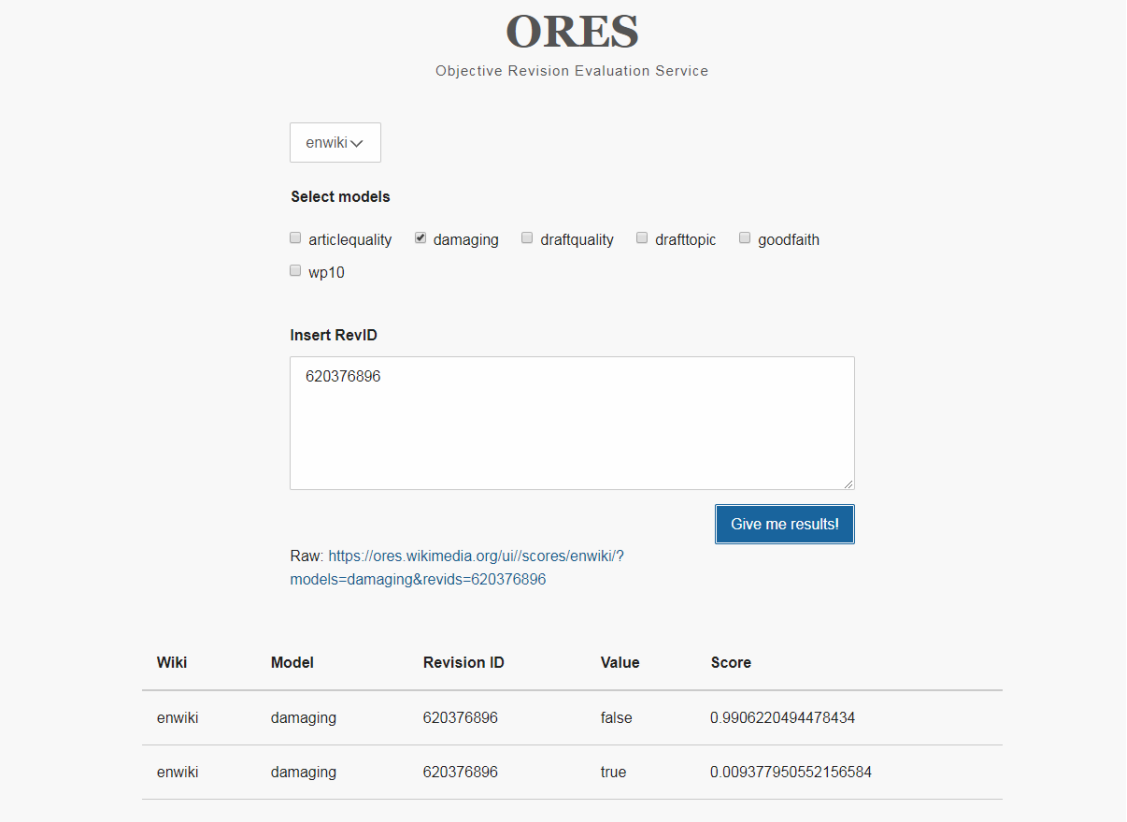
6 ORES Experiment

Experimenting with Revscore and ORES in the Classroom

- Teaching students how to edit Wikipedia
  - Before ORES: Hoping the changes stick around
  - With ORES: Getting immediate feedback (stub, start, C ...) and going back for more editing
- Experiment: 13 students (+ 1 teacher) to improve articles in 1 hour and check results with ORES
  - 8/14 articles classified as **Start**, others as **Stub**
  - Results: 9/14 jumped up 1 ranking, 1/14 jumped up 2 rankings
- Conclusion (more or less in the article): Gamification elements in form of scores and instant feedback but also through 1 hour challenge

  ⇒ Seems like an excellent method to raise motivation!

7 Current ORES GUI



8 Google: Attacking discrimination in machine learning

8.1 Attacking discrimination with smarter machine learning

- Threshold classifier: make yes/no decision, putting things in one category or another

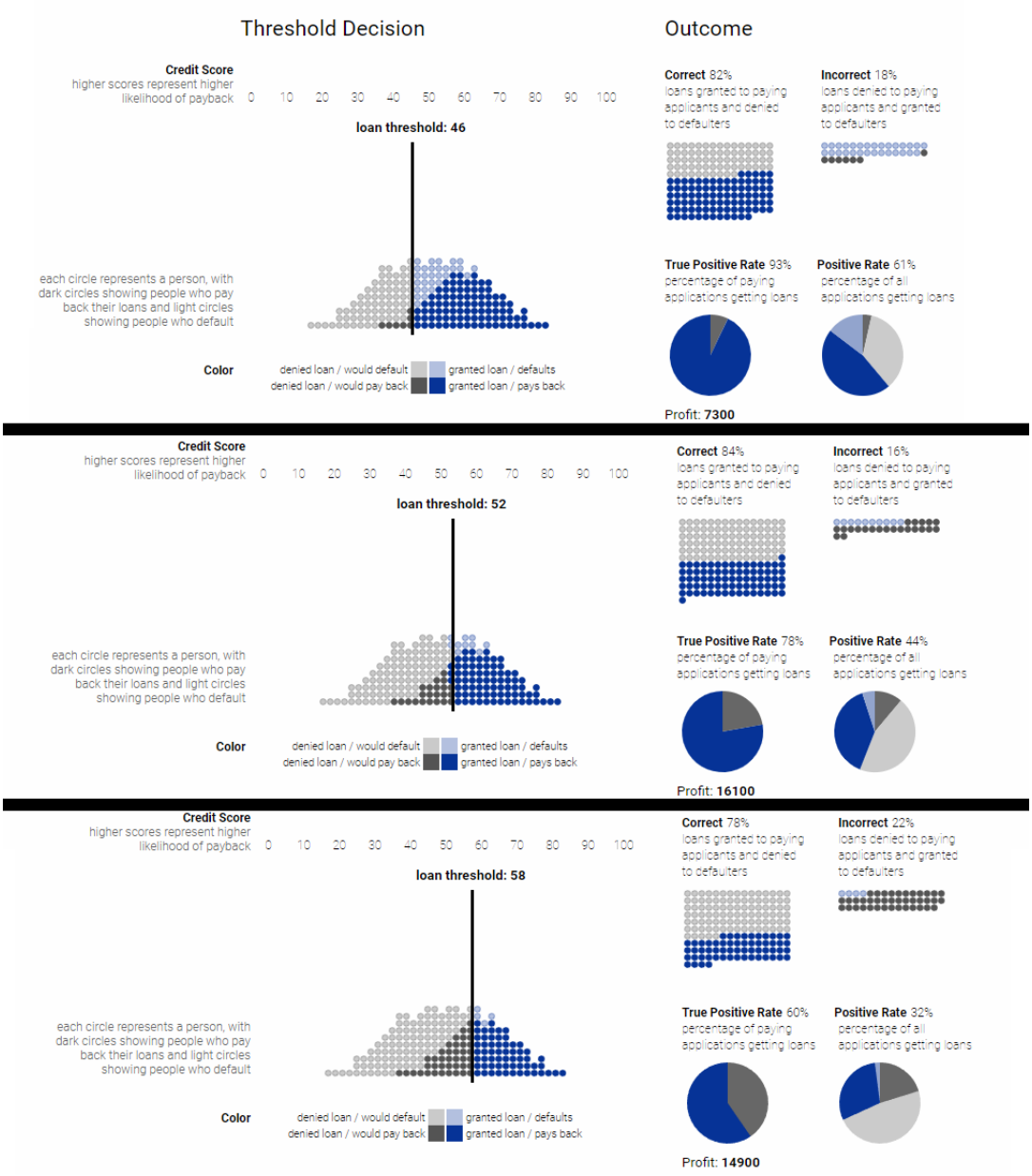
  ⇒ Look at how they work, ways they can be unfair and how to make them fair again

  ⇒ Example here: loan granting scenario

- Credit scores: representing the probability of payback

  ⇒ One number calculated from numerous factors

- Bank then picks a **threshold** and people with credit scores below that are denied the loan



Observe tradeoff:

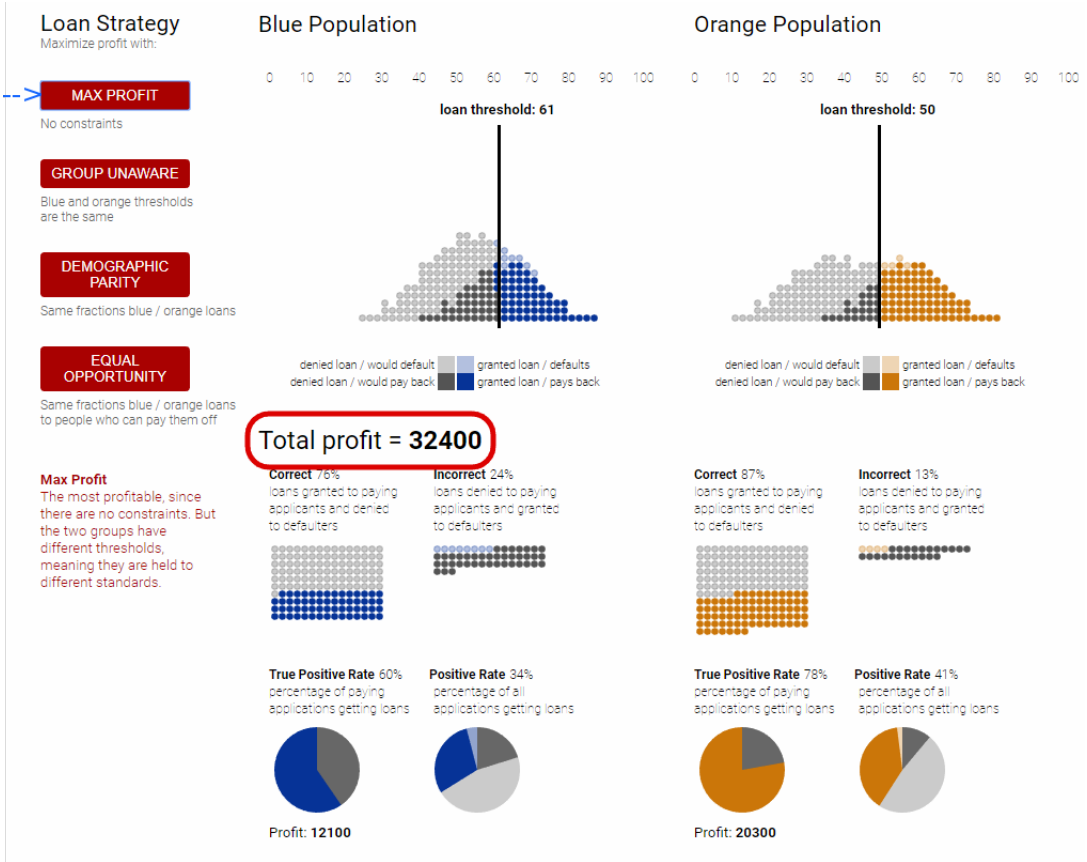
- Too low threshold: bank gives loans to many people who default
- Too high: many people who deserve a loan won't get one

- Different thresholds for different goals (maximize profit, maximize number of correct decisions, ...)
- Interesting: Threshold of 52 → Correct: 84%, profit: 16100 **but** Threshold of 54 → Correct: 83%, profit: 16600

8.2 Classification and Discrimination

The definition of "the correct" decision depends on the context and is complicated. Consider:

- Two groups of people, equally likely to pay off loans (overall)
- Different distribution
- Depending on the goal (e.g. max profit) the two groups will be held to different standards:



Other strategies:

- Group-unaware
  - Hold both groups to same standards
  - Problem: might be unfair to ignore real differences
- Demographic parity
  - Thresholds yield same fraction of loans to each group (positive rates are equal)
  - Problem: only looks at loans given, not loans paid back → fewer qualified blue people are being given loans than in the orange group
- Equal opportunity (paper by Hardt, Price, Srebro)
  - Of the oeople who can pay back a loan, the same fraction in each group should actually be granted one

8.3 Improving machine learning systems

- *Key result of the paper:* Given any scoring system it's possible to find thresholds that meet any of these criteria → always possible to attack discrimination
- Attacking discrimination in ML will require a careful, multidisciplinary approach