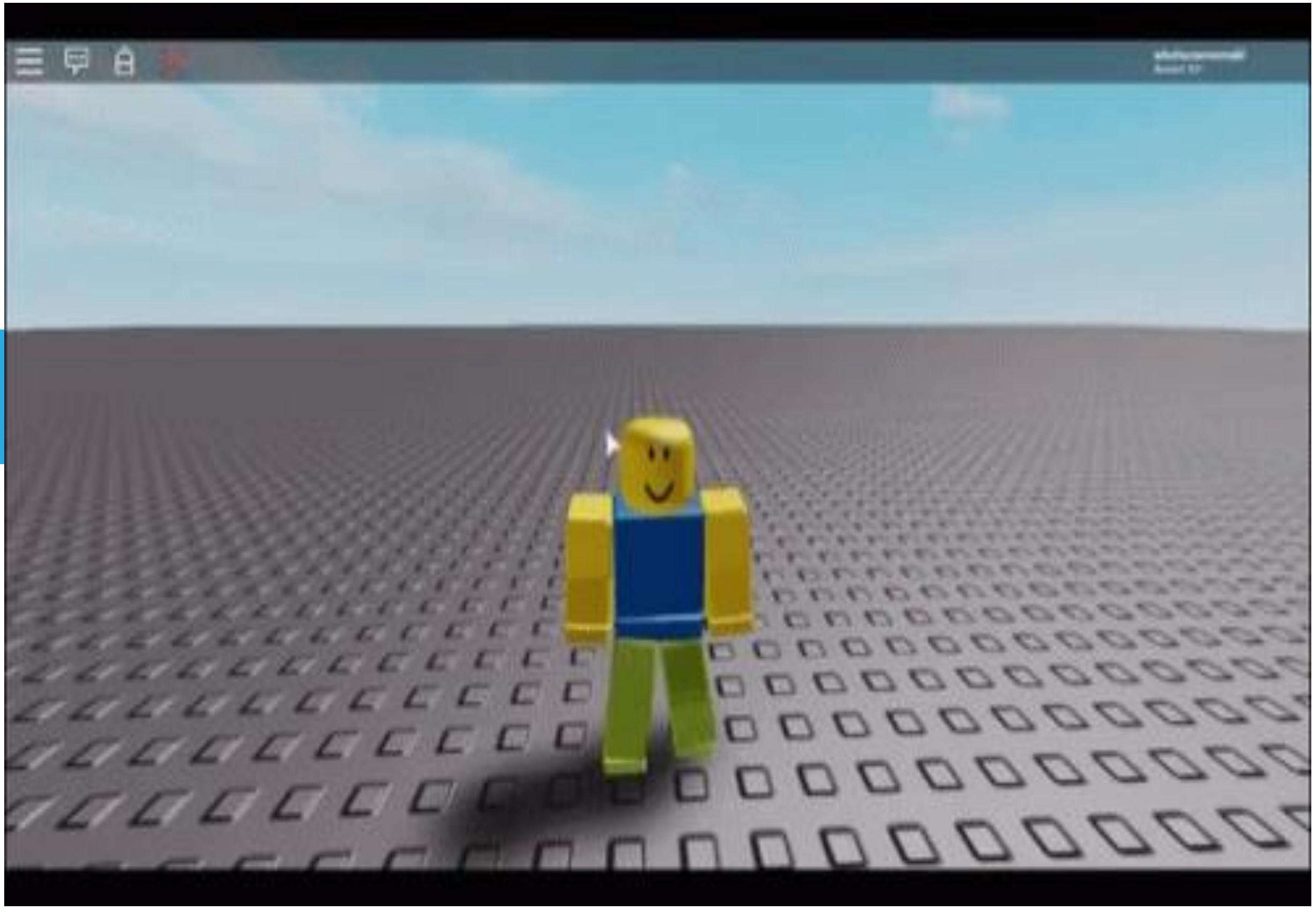
With Great Power Comes Great Responsibility: Exploring Administration in the Decentralized Web

Ishaku Anaobi

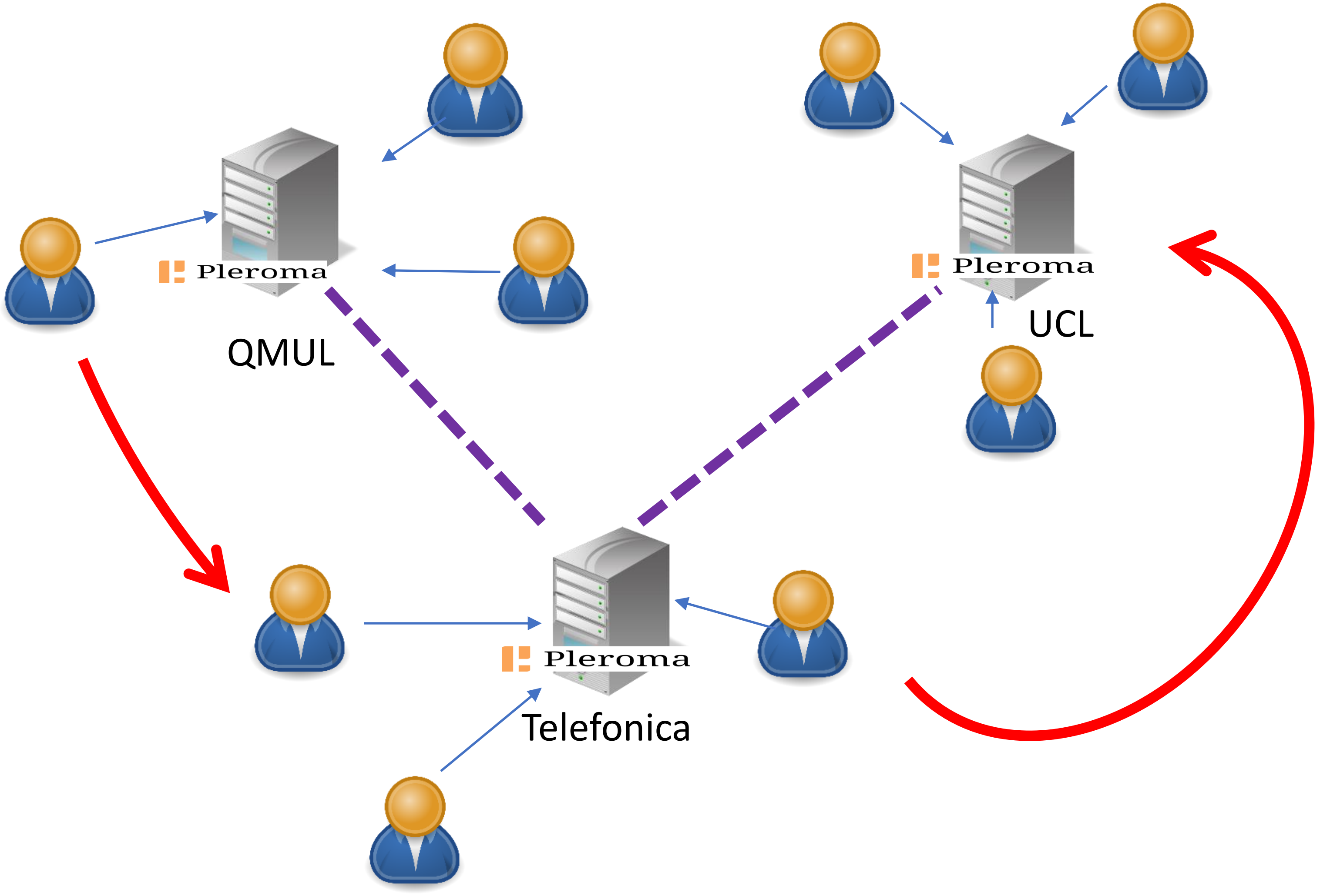
Queen Mary
University of London

Decentralized Web



- Imagine breaking Twitter into multiple parts
- Each part operates as a mini twitter
- Dweb has multiple implementations e.g Pleroma

Decentralized Setting



- Segmented communities
- Independent servers called instances
- Users register on their community servers and exchange information
- Federation
- Fediverse

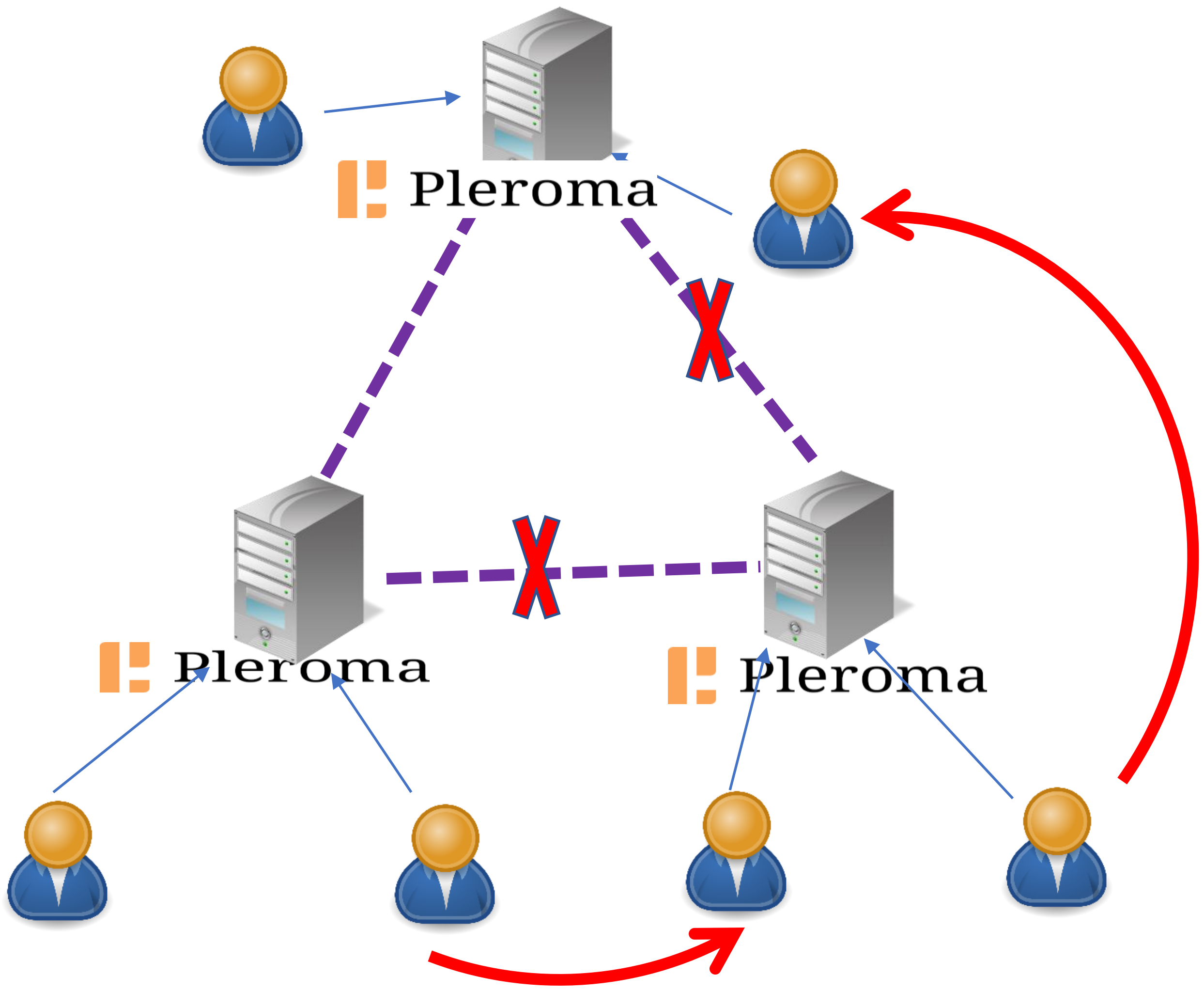
Why is this cool?

- Offers full control
- Freedom
- However, content moderation could be an issue

Who manages these instances?

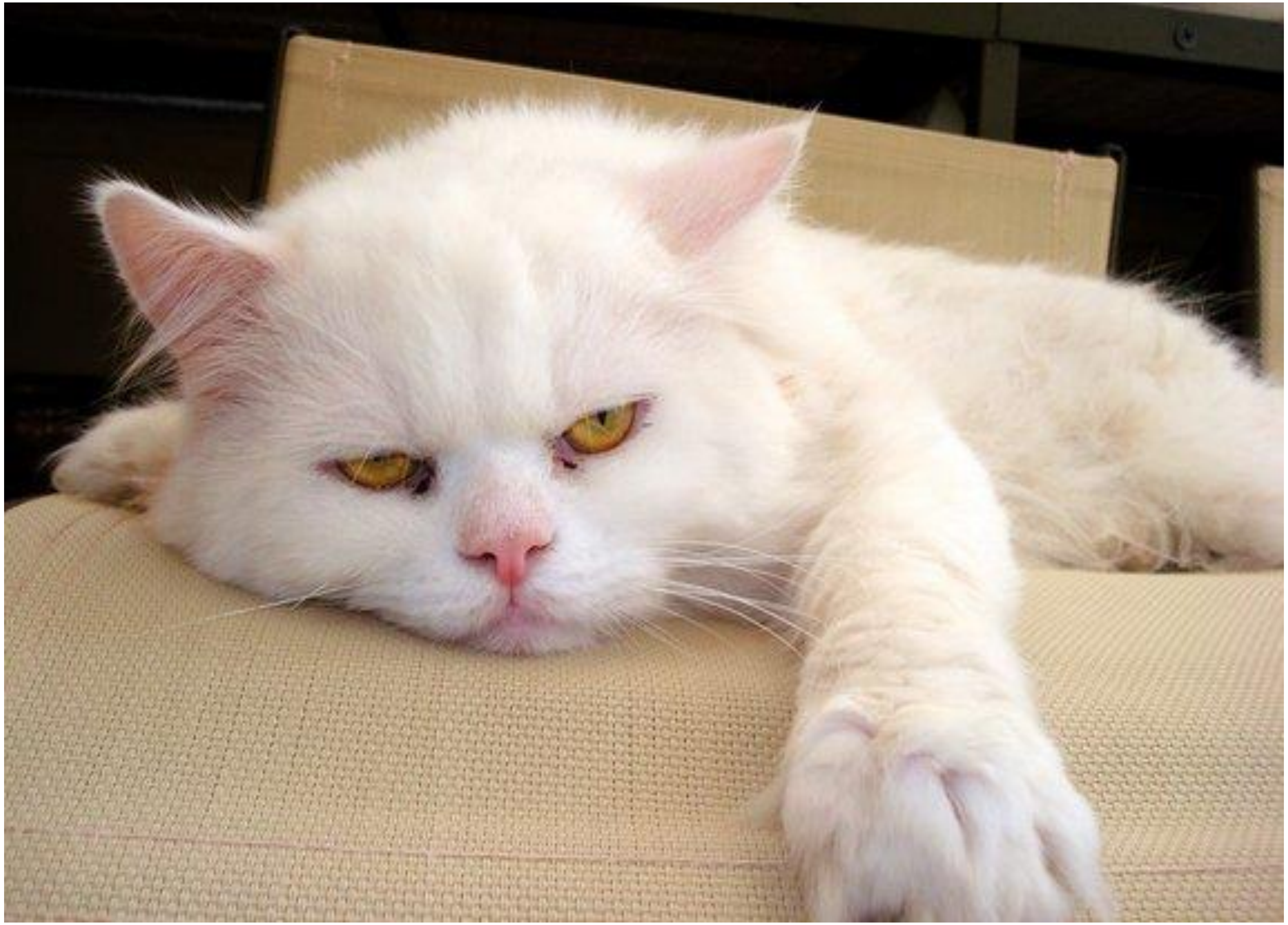
- Specialized users responsible for the day-to-day administrative tasks on the instances called **Administrators**
- Usually, volunteers
- By default, the creator of an instance will take on the role of the administrator
- Can delegate such responsibilities to multiple others.

How do admins moderate?



- Implementations rely on federation policies
- Admins create rules and apply ***"actions"***
- Mostly applied instance-wide
- Moderation is mostly manual

Can admins cope?

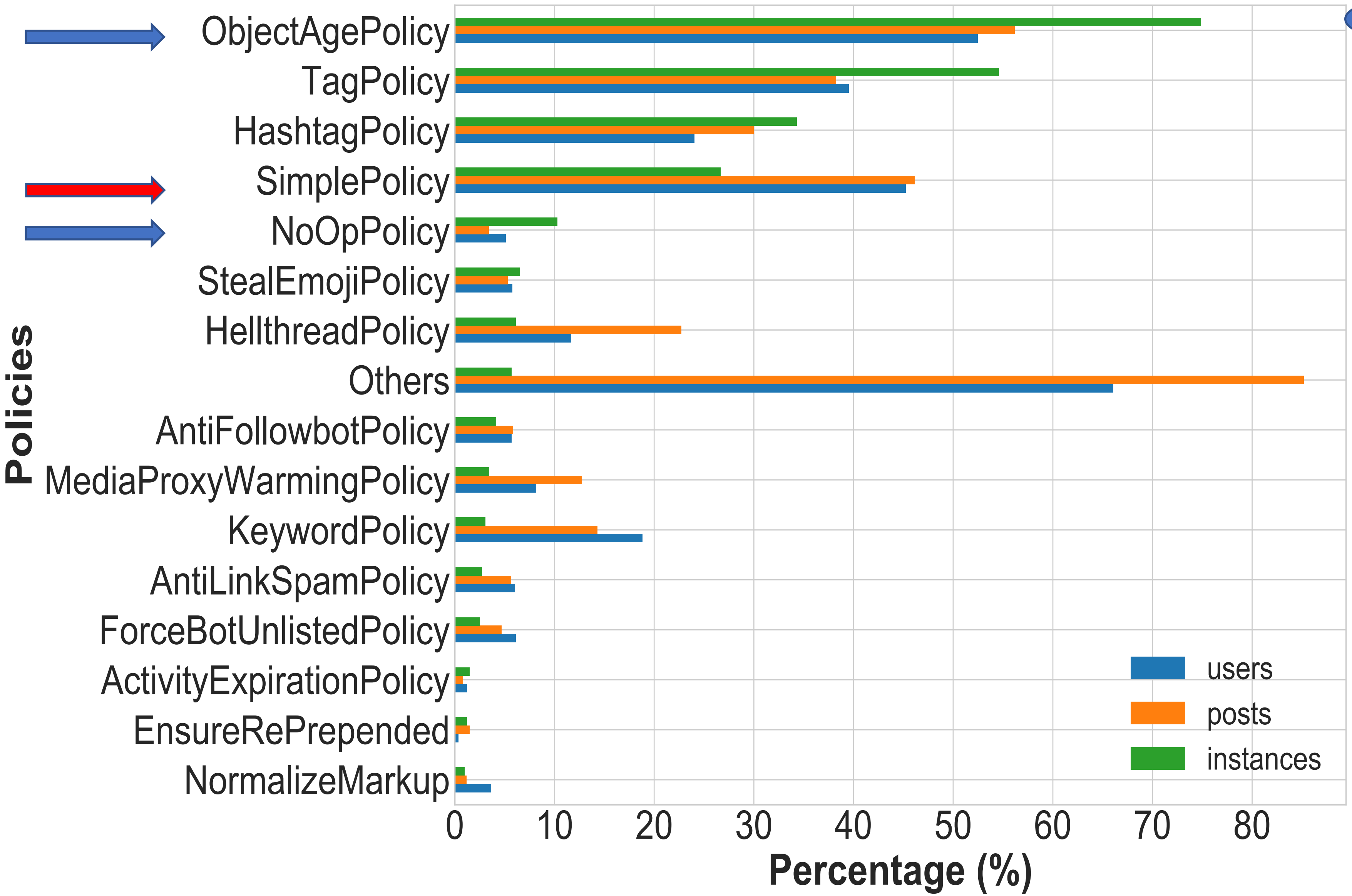


We conjecture that admins as volunteers could possibly get overwhelmed

Dataset

Date	instances	users	posts	policies
16-Dec-2020 – 19-Oct-2021	1,740	133.8k	29m	49

What policies do admins apply?

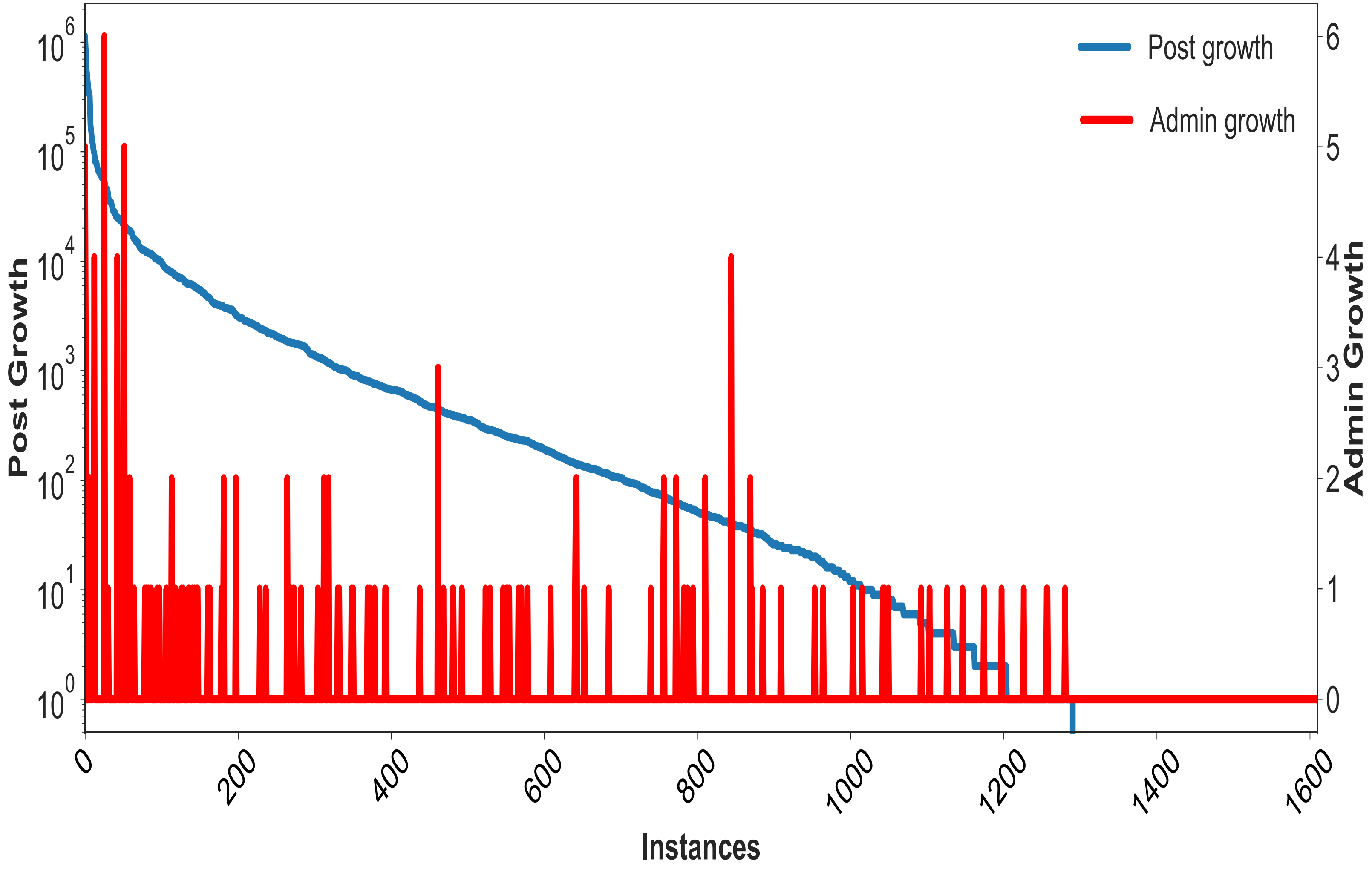


Default policies

SimplePolicy with wide range of actions

18.2% of instances run only on the default policies alone

Do admins seek help?

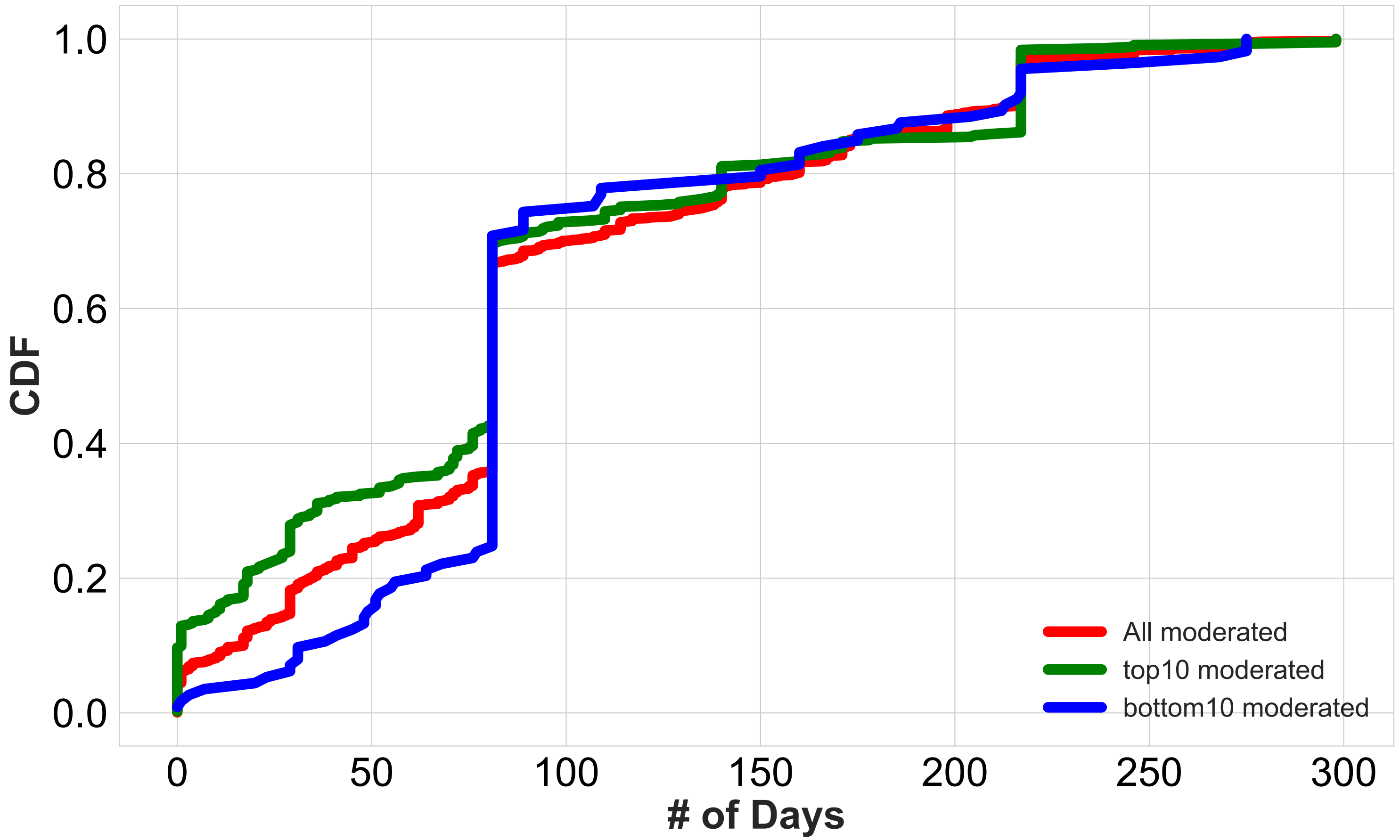


Post growth vs. Administrator growth

The number of administrators do not grow proportionately with the number of posts

Only 6.9% of instances recruit additional admins

How swift are admins?



Number of days from federation to moderation

Admins take an average of 81.2 days to apply any form of policy against other instances

Even for well-known highly controversial instances (anime.website:150d ays)

What do we propose?

WatchGen

A tool to recommend to admins
 a “watchlist” of other instances that
may require federated moderation

Developing WatchGen (Feature Selection)

- We extract multiple features from each instance (e.g number of users and posts).

- We experimented with a total of 38 features

- We distil this down to the 16 most determinant features

Model Training

We train a number of machine learning models

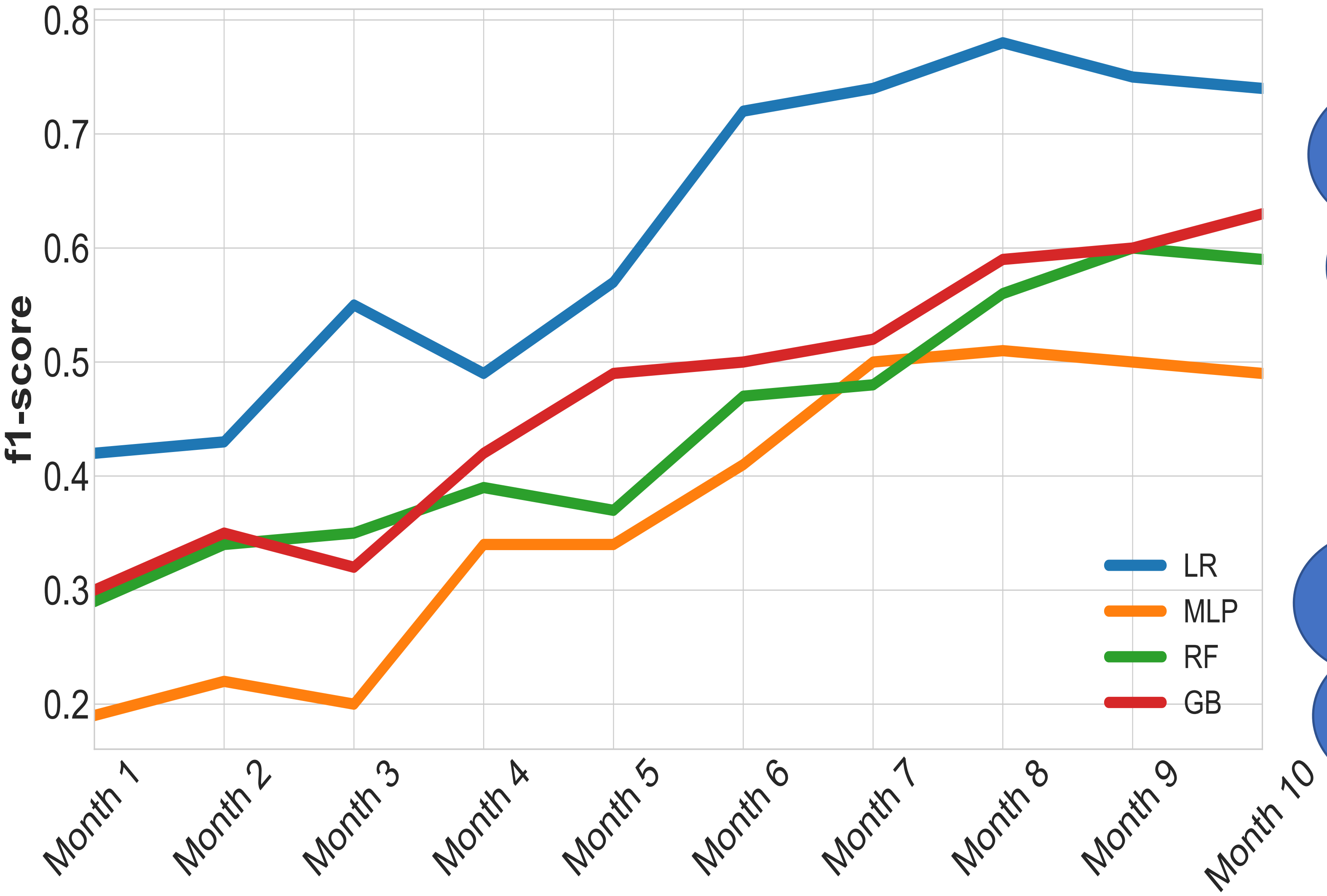
(i) Logistic Regression

(ii) Multilayer Perceptron

(iii) Random Forest

(iv) Gradient Boosted Trees

Experiment 1 (global)

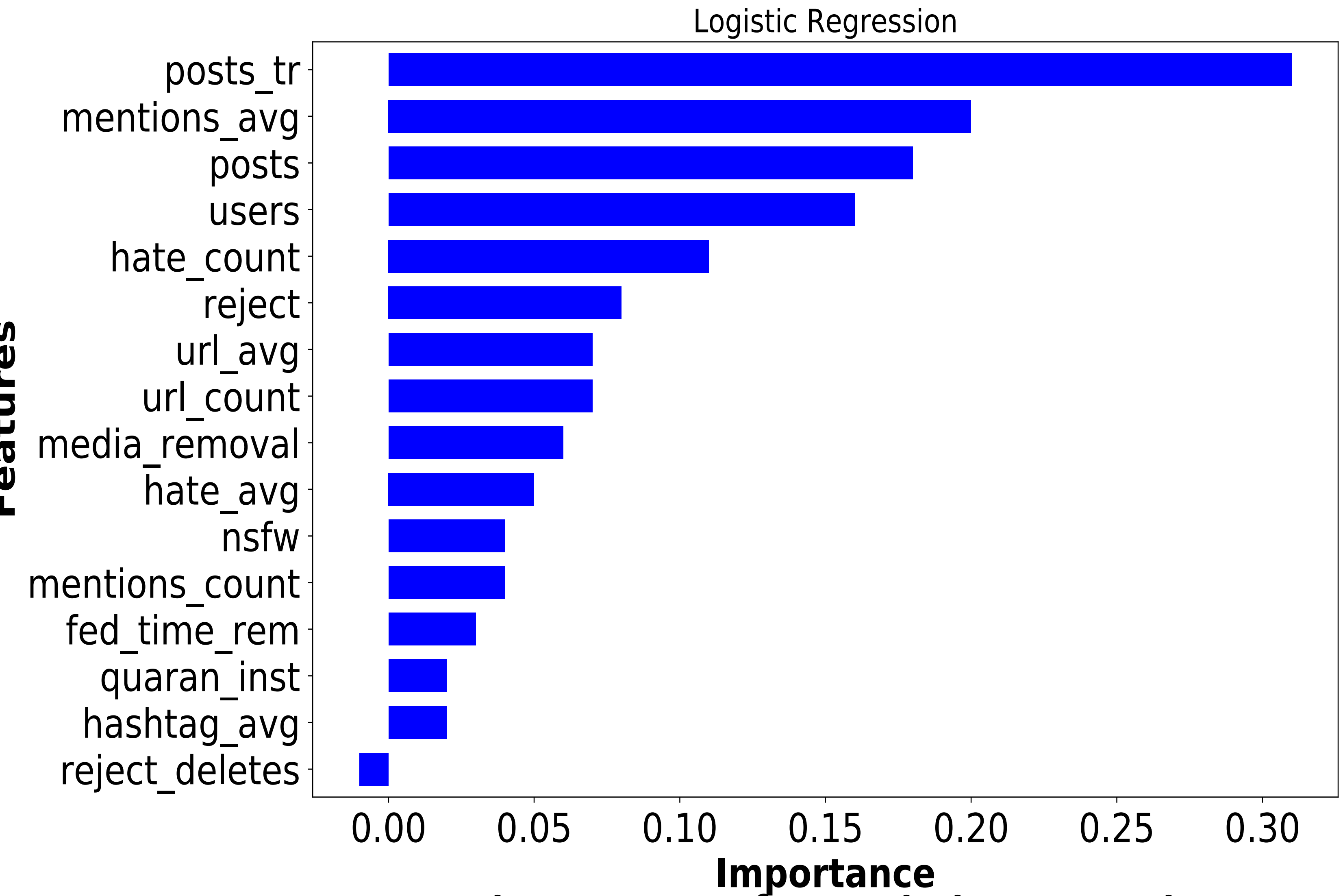


Entire pool of data predict if a given instance will be subject to any policy

Best performing model is Logistic Regression with 8 months of training data (f1=0.78)

F1 scores for our 4 models

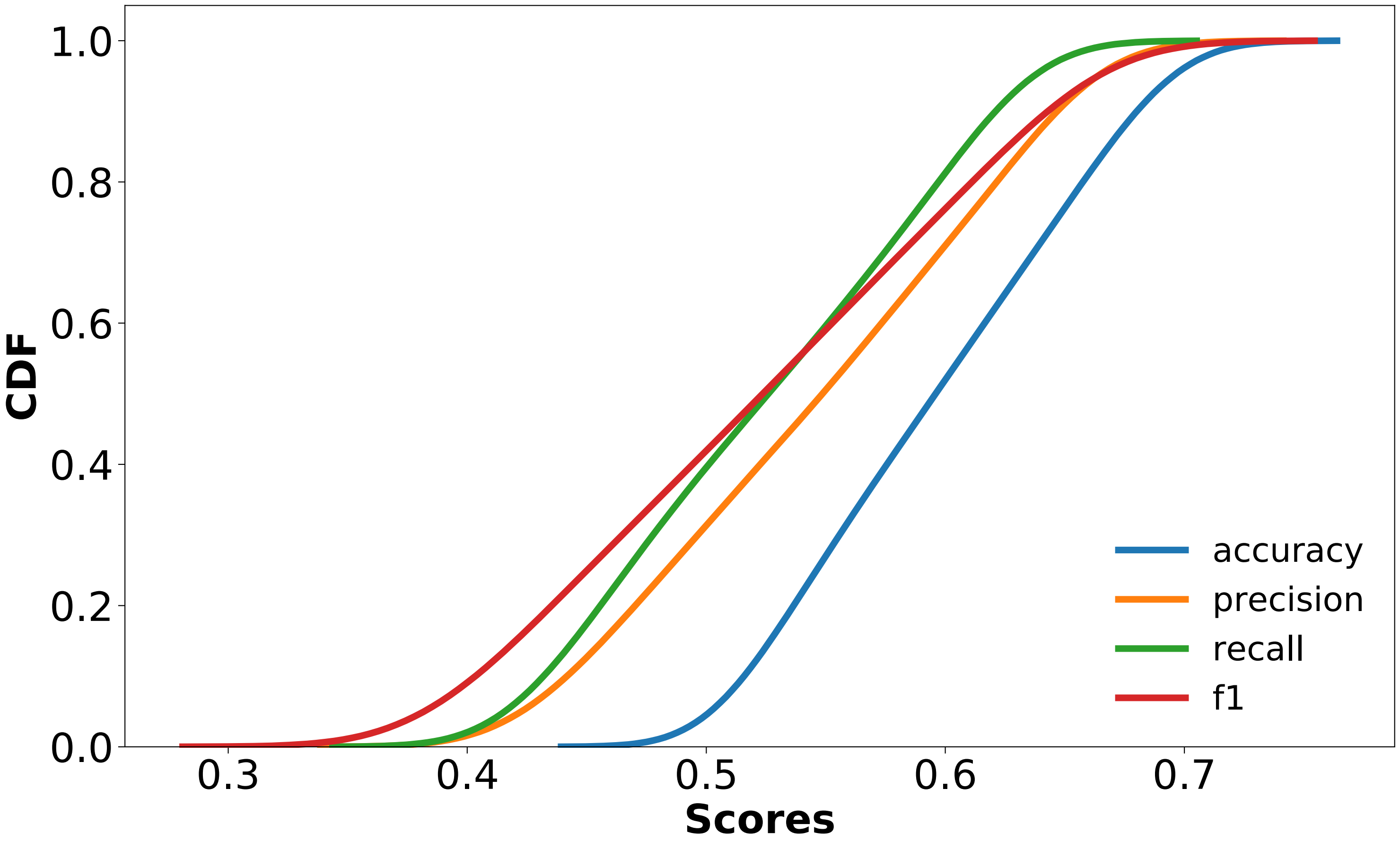
What features are most Important?



Feature importance for Logistic Regression

Top 3 features are all related to the posts on an instance

Experiment 2 (Local)



Per-instance performance metrics for Logistic Regression

Avg f1-score of 0.52. 23.9% of instances gain above 0.6

Performance is impacted by size

Questions?