Autoautomod

Project Specifications

## General:

Machine learning script that can automatically report/remove bad comments. Two part process that examines mod history to learn what types of comments typically get removed, then it makes predictions on new comments made and will report bad comments or remove them (if certainty is high enough).

## Tools:

Python (language), Anaconda(env)

APIs: PRAW(Reddit), Scikit-Learn, Numpy, Pandas, matplotlib, NLTK(language processing)

## Features:

All features will be stored in a uint8 array. Scaled to 0~256

#### Comment Body:

* Count of top 3000(?) words/symbols most unique to each removed and allowed comments
  + |Freqingood – Freqinbad| / Freqingeneral to find which symbols
  + More, 10k? If doing a feature selection step. Cut down to 4k? Useful #.
    - Can do w/ RF still but slow af
    - Could use elastic net but this is non-linear (probably fine anyways)
  + Preprocessing (occurs AFTER collecting some other features):
    - Strip punctuation, lowercase
    - Chop up by spaces
    - Convert smilies, formatting, links, numbers, dates, times to uniqueidentifiers like ‘a\_time’, ‘a\_date’
    - Stemmer? Worth trying with a CV
    - Convert everything to numbers/a count in an indexed list
      * Get a count for # of appearances in good and bad dataset
      * Keep only the most relevant ‘words’. 3000 or so? CV this.
* Link\_count, numb\_count, exclamation\_count, emoji\_count etc.
  + These will be converted to ‘words’ in the preprocessing stage
* Num\_words – how many words in the comment
* Num\_chars – how many chars in the comment
* Longest word
* Max\_inarow, max\_inarowchar
  + Finds stuff like !!!!!!!!!!!!!!! Waaahhhhhh
* num\_caps
  + More caps info? Words w/ 1st letter cap vs multicaps
* ?Pct\_validwords (by character count) pct\_spaces pct\_specialchars
  + Is this redundant w/ other features?
* Proper\_punct – 1st sentence uses correct caps/punctuation?
* Proper\_grammar - 1st sentence uses proper grammar?

#### Comment Meta:

* Comment\_location (0 – top level)
  + Careful about bias of mods typically missing more deeply nested comments AND ignoring heavily replied to comments
* Co\_age …. Comment age at time of post
  + Careful about using co\_age since mods are less likely to police old posts, is this something we want the bot to have?...
* num\_reports …. and content?
  + not possible to check how many reports a comment had in past ☹
* Num\_children
  + Children comments won’t exist in test set. Is this bad? Pointless? Weighting training more on comments without children IS good. It clears up a mod bias potentially. But this would just be thrown out if everything is always 0…. So a RF won’t do this
* Diff w/ parent comment
* Diff w/ thread title
  + Seems hard to do quickly

#### Comment Parent:

* Comment\_parent\_score
* Comment\_parent\_state (removed?)

#### Commenter Meta

* User\_karma, user\_age,
* user\_num\_rem
  + Would need to keep track or scan user history… both suck

#### Submission Meta

* Pct removed comments in thread
* post\_karma
  + Does post\_karma introduce a bias we want to avoid? (mods are more lenient on big threads)
* post\_num\_comments

## Stages (Learner)

### Setup

* Check config for refreshkey
  + If no valid refreshkey, guide user through getting one/use user login
* What other config settings?

### Data Collection

* Collect comments from last 1000 threads (to get around the 1000 comment cap? By timestamp easier?)
  + Allow for restart to add to collected data? Or always fresh?
* Ignore:
  + Use an ignore list regex from the config file
  + Removed threads, meta threads, party threads, media threads, question threads
  + Special users (bots)
  + ITAR violations shouldn’t be removed by this bot without notifying the mod team… this is hard since it involves comment content
  + IF learning from # comment reports, then MUST ignore ‘mod required’ type comments to update sidebar etc.
  + Comments <24hrs old in training data ….
  + Posts >96hrs older than the post in training data. (Mods don’t check that old shit)

### Pre-processing

* DEV - Save raw data to disk for future use if needed.
  + Likely pointless in release.
  + Needed to try different processes mayyyybe.
  + Could try on only a few hundred at a time to avoid this step.
  + Compression needed?
* Preprocessing PER COMMENT should be in its own function/file
* Collect Features described in that section …
* Save processed data to disk (Hopefully under 5GB) maybe need compression here too.

### Feature-Selection

* Do round for feature selection, either hand selecting words or running a RF on a subset to trim?

### Learn

* Fit the RF.
  + DEV - Look at CVing an SVM or ADA solution as well. Or even something multiclass (hard). Definitely CV the number of trees (estimators) in our Random Forest
    - Default max\_features = sqrt(n\_features)
  + DEV – Also CV the stemmer, lowercase … other things?
  + DEV – If we’re doing feature selection for the hardcoded features, now is the time to look at that.
  + DEV – print out some nice graphs, get some stats, understanding of how well it is working to find smart default settings for other subs … some of this will still be useful after Dev for mods. Comparison to automod is neat.
* Save the resultant classifier to a file
* Could at this point run an unsupervised algorithm to classify our removal types

## Stages (Decider)

### Running

* Log in, load up the classifier
* Get new comments as they come in
* Run prediction on them using our classifier to get a pct
* Do action based on pct (report, remove, notify mods)
* Keep stats log
  + Modmail stats updates (comparison to automod and stuff)
  + Could go super fancy and have this upload graphs to imgur
* Online learning?
  + Running the learner every few months will improve the confidence levels of the classifier as fewer comments are missed each round, and get checked by mods. Not quite ‘online’ but pretty good.

## Notes/Concerns:

* u/CAM-Gerlach is insanely helpful. Give him more credit
* How are we handling the misclassification rate in the training data? GIGO?
* Keep it generalized to allow use on any sub
* Ideas for > binary classification? Meaningful reports/removal notes.
  + Would have to give up on RF?
  + Could use an unsupervised learning algorithm to help give rough titles for the removal types
    - Could feed this into a k classifier algo and let each classifier diversify inc acc…. At heavy costs. Probably not worth? Unless there are several radically different types of comments that need removal, in which case, perhaps a different classifier would be better.
* Config allows targeted # reports/hr or confidence level %
* Avoiding biasing against users?
* Keep automod in mind, dual system… lets not conflict with each other.
  + Use BOTH for several months until you feel confident that automod is out classed
  + Can measure automod classification rate compared to autoautomod :D
    - ‘was reported by the bot’ / ‘was removed’ (if old reports are visible in API)
* NLP preprocessing steps?
* Best of luck keeping this solution to something that will fit in ram… I can probably handle <13GB
* Have different configs/ignore lists for training and predicting