score_priority_modules.md

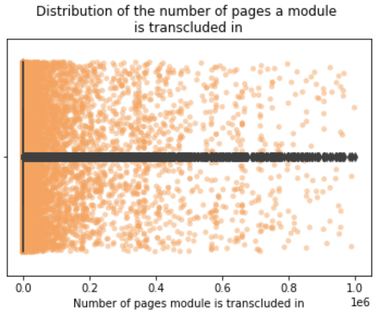
Scoring Lua Modules

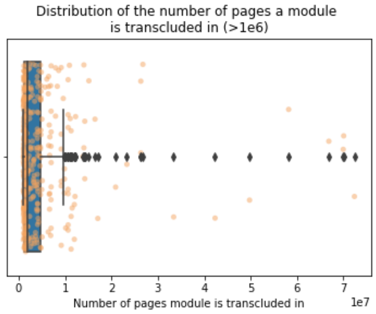
Based on the data analysis results, we found that most values (pagelinks, langlinks, transclusions etc) are highly skewed and it will be hard to identify and list important modules solely relying on these raw values. We identified a rough list of limits to help create a metric for scoring the module importance. The procedure is shown below for the feature `transcluded_in` as an example. The same process is followed for other features too. See code in `get_distribution.py` file.

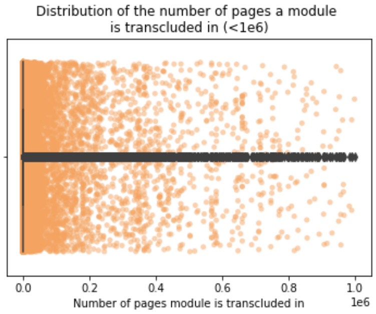
Steps to calculate score

1. Get limit value

First data is collected and analysed. The distribution is observed and a limit value is approximated. For the example below, after a couple of iteration it was evident that most modules are transcluded in less than 1 million pages. So those modules transcluded in more than 1 million pages may be worth a little more attention than others. So the limit set for the feature `transcluded_in` is 1e6.





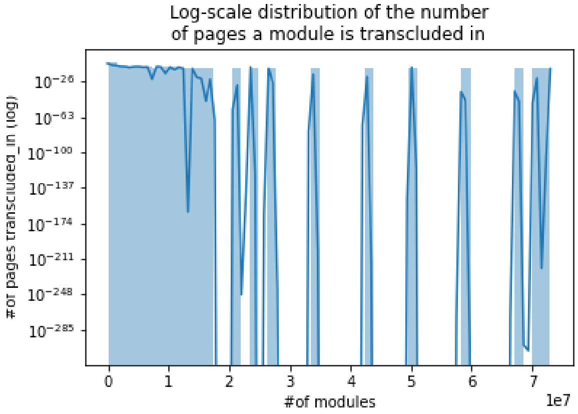


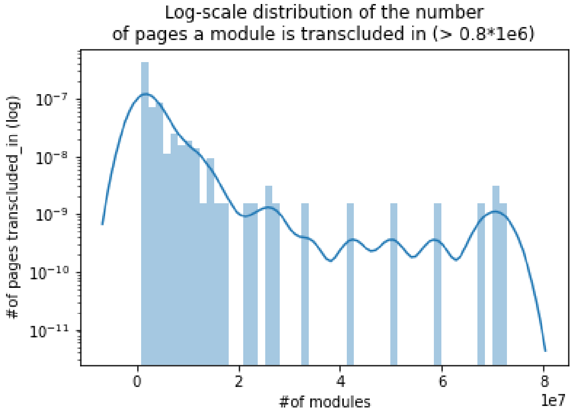
Note that these values are later normalized by wiki. So `trancluded_in = traincluded_in/(sum of trainscluded_in in this wiki)` to ensure all wikis got equal priority despite their sizes. Then the limits are expressed as percentage instead of raw values, but it is the same idea.

2. Modify distribution

Now that we got our limit value it's time to calculate scores. Each feature would be one component of the score. To normalize and turn raw values to scores we modify the original distribution of the features such that the limit we determined is somewhere below `87%`. Originally the limit would be at the 99.999999....th percentile, that way there is no way we can dig out important modules. The result of modifying the distribution is that we give less importance to modules that have *very* less values (0-10 for most features), and that's okay for us since we are aiming *high*.

We change distribution using the following formula: `feature = feature[feature>limit*multiplier]`. The multiplier is a value `(0,1]` whichever gives the max value for the percentage. Below is the distribution before and after modification.





3. Get feature score

To get the score from the feature all we have to do is find what percentile it is in the modified distribution. Is it 99th percentile? Quite important! Is it 5th percentile? Maybe not so important, so has a low score of 0.05. This also ensures that the values across features are standardized to be between 0 and 1.

4. Get score

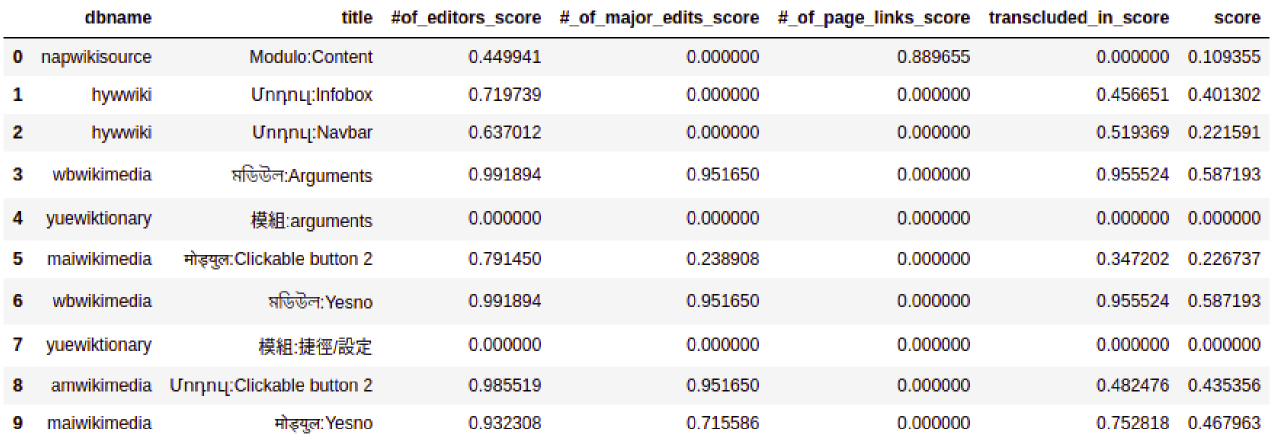
Now to get the score of the module we calculate the weighted sum of the feature scores.

```
score = feature_score1 * weight1 + feature_score2 * weight2 + ...
```

The weights here are changeable and we can increase or decrease them to get different score (and so list of modules we think are important) based on what features are more important for us to consider.

Sample scores

Here are some modules with their final scores.



	dbname	title	#of_editors_score	#_of_major_edits_score	#_of_page_links_score	transcluded_in_score	score
0	napwikisource	Modulo:Content	0.449941	0.000000	0.889655	0.000000	0.109355
1	hywwiki	Մոդուլ:Infobox	0.719739	0.000000	0.000000	0.456651	0.401302
2	hywwiki	Մոդուլ:Navbar	0.637012	0.000000	0.000000	0.519369	0.221591
3	wbwikimedia	মডিউল:Arguments	0.991894	0.951650	0.000000	0.955524	0.587193
4	yuewiktionary	模組:arguments	0.000000	0.000000	0.000000	0.000000	0.000000
5	maiwikimedia	मोड्युल:Clickable button 2	0.791450	0.238908	0.000000	0.347202	0.226737
6	wbwikimedia	মডিউল:Yesno	0.991894	0.951650	0.000000	0.955524	0.587193
7	yuewiktionary	模組:捷徑/設定	0.000000	0.000000	0.000000	0.000000	0.000000
8	amwikimedia	Մոդուլ:Clickable button 2	0.985519	0.951650	0.000000	0.482476	0.435356
9	maiwikimedia	मोड्युल:Yesno	0.932308	0.715586	0.000000	0.752818	0.467963