1 Collecting all Permission within a cluster

At first I'm collecting all permissions that all apps of one cluster want. As example the cluster Flashlight:

Permissions

0	**4**	**9**	**10**	**11**	**12**

With the permissions:

ID	**Name**
0	In-App-Purchases
4	Calender
9	Pictures/Media/Files
10	Storage
11	Camera
12	Microphone

2 Calculation of permisson weight

At next I calculate the weight of the permissions, which consists of two parts.

- Relative frequenzy of apps that don't want this permission

0.4	0.8	0.6	0.2	0.0	0.6

- Badness of the permission

0.1	0.6	0.1	0.1	0.9	0.9

This two parts are multiplied.

Permissions

0	**4**	**9**	**10**	**11**	**12**
0.04	0.48	0.06	0.02	0.0	0.54

3 Filling the matrix

Now I create an apps × permissions matrix, where each app holds the weights of the permissions, which the app needs.

Permissions

	ID	**0**	**4**	**9**	**10**	**11**	**12**
Apps	**14**	0.04	0.0	0.0	0.02	0.0	0.54
	42	0.0	0.48	0.06	0.0	0.0	0.0
	145	0.04	0.0	0.0	0.02	0.0	0.0
	465	0.04	0.0	0.06	0.02	0.0	0.54
	1010	0.0	0.0	0.0	0.02	0.0	0.0

4 Calculating the summary of the badness

At the next step I'm calculating the summary of the *badness* of an app.

		Permissions						
	ID	**0**	**4**	**9**	**10**	**11**	**12**	$\sum$
Apps	**14**	0.04	0.0	0.0	0.02	0.0	0.54	**0.6**
	42	0.0	0.48	0.06	0.0	0.0	0.0	**0.54**
	145	0.04	0.0	0.0	0.02	0.0	0.0	**0.06**
	465	0.04	0.0	0.06	0.02	0.0	0.54	**0.66**
	1010	0.0	0.0	0.0	0.02	0.0	0.0	**0.02**

5 Grouping the apps

I use the K-Means clustering algorithm to split up the app into three groups.

	ID	$\sum$
Apps	**14**	0.6
	42	0.54
	145	0.06
	465	0.66
	1010	0.02

Now I define the following HSV color ranges for the three groups.

- Good - green
  - 80 - 120 degree
  - 66.6 - 100.0 %
- Middle - yellow
  - 30 - 79 degree
  - 25.0 - 65.83 %
- Bad - red
  - 0 - 29 degree
  - 0.0 - 24.16 %

For every group I calculate the percentage distribution for each app inside a group and map them on the respective color range. If a group only contains one value, the maximum of the respective range is used.

```
# 0 - 100
value = 100 - ((app_values[i] - min_value) * 100.0) / (max_value
    - min_value)
# min_range - max_range
value = (value * (color_range[1] - color_range[0]) / 100) +
    color_range[0]
```