Optimal Number of Choices in Rating Contexts

Sam Ganzfried and Farzana Yusuf
Florida International University, Miami, FL
School of Computing and Information Sciences
http://www.ganzfriedresearch.com/
sam.ganzfried@gmail.com

Human rating systems

- Humans rate items or entities in many important settings
  - Physical attractiveness on dating websites and apps
  - Teachers rate students' work
  - Reviewers rate conference submissions
  - Etc.
- In these settings, the users assign a numerical (integral) score to each item from a small discrete set. However, the number of options in this set can vary significantly between applications, and even within different instantiations of the same application.

Are you hot or not?

Figure 1: Hot or Not users rate attractiveness 1–10.

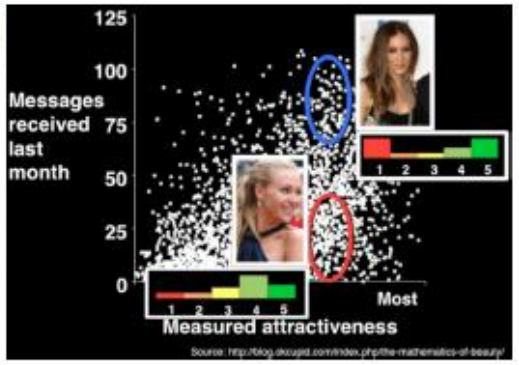

Figure 2: OkCupid users rate attractiveness 1–5.

Figure 3: Tinder users rate attractiveness 1–2.

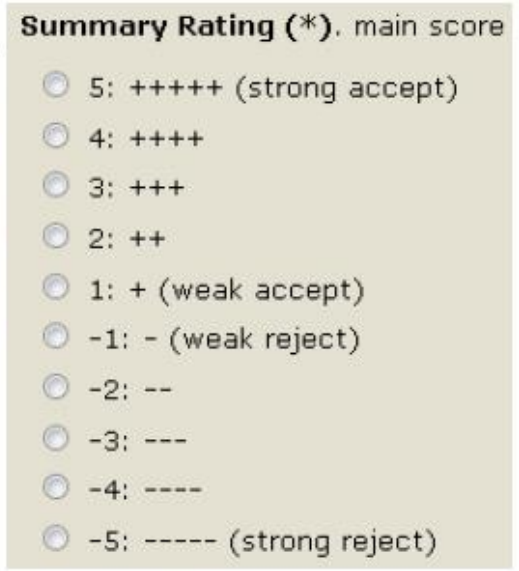

Figure 4: IJCAI reviewers rate papers -5–5.

Compression model

- We study model where users have underlying integral ground truth score for each item in {1,…,n} and are required to submit an integral rating in {1,…,k}, for k << n.
- Two generative models:
  1. Uniform: the fraction of scores for each value from 1 to n is chosen uniformly at random
  2. Gaussian: the scores are chosen according to a Gaussian distribution with a given mean and variance
- We then compute "compressed" score distribution by applying

  $s \leftarrow \text{floor}(s / (k/n))$ to map from {0,..,n-1} to {0,…,k-1}.
  - We also consider a rounding approach that maps each score to closest "midpoint:" $m^k_i = n(2i-1)/(2k)$.
- We compute the average "compressed" score a_k and its error

$e_k = |a_f - [(n-1)/(k-1)]*a_k|$, where a_f is ground truth average score.

- The goal is to pick $\text{argmin}_k e_k$.

- One could argue that this model is somewhat “trivial” in the sense that it would be optimal to set k = n to permit all the possible scores, as this would result in the “compressed” scores agreeing exactly with the full scores. However, there are many reasons that we would like to select k << n in practice, thus making this “thought experiment” worthwhile. It is much easier for a human to assign a score from a small set than from a large set, particularly when rating many items under time constraints.

- We could have made model more complex by adding in a cost function that explicitly penalizes larger values of k. This would be somewhat arbitrary, and leave this direction for future study.

- So, in our simple model, increasing k will always decrease e_k right?

Example

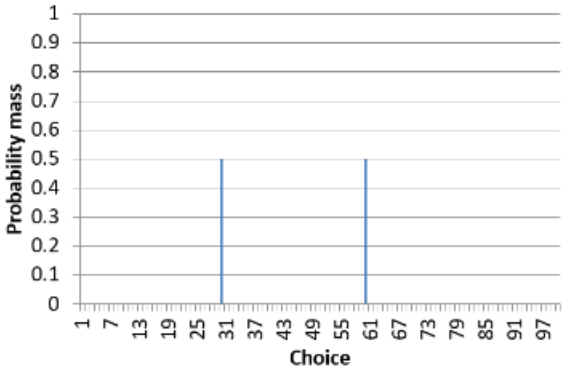

Figure 5: Example distribution for which compressing with $k = 2$ produces lower error than $k = 3$.

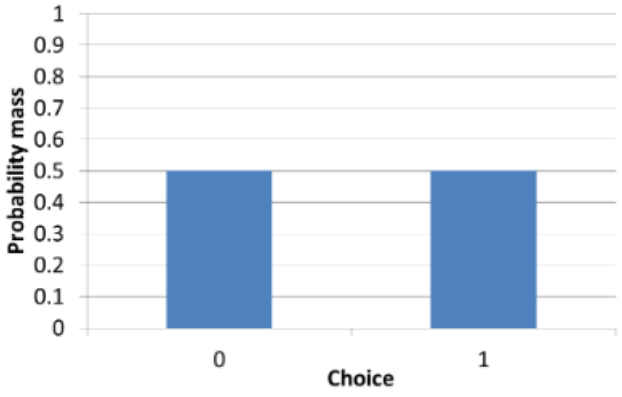

Figure 6: Compressed distribution using $k = 2$.

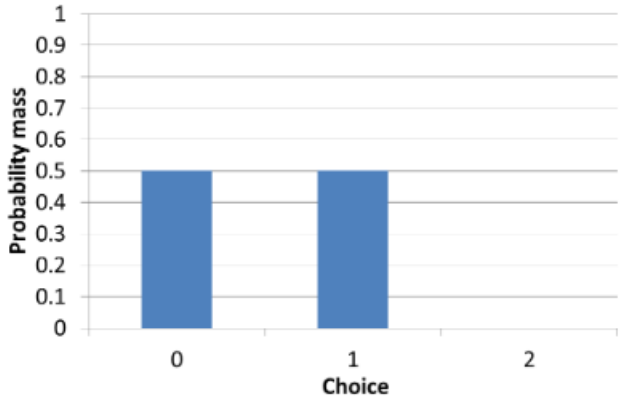

Figure 7: Compressed distribution using $k = 3$.

Example where k=2 outperforms k=3

- $a_f = E[X] = 0.5 * 30 + 0.5 * 60 = 45$.
- If we use k = 2, then the mass at 30 will be mapped down to 0 (since 30 < 50) and the mass at 60 will be mapped up to 1 (since 60 > 50).
- So $a_2 = 0.5 * 0 + 0.5 * 1 = 0.5$, $e_2 = |45 - 100\ (0.5)| = |45 - 50| = 5$.
- If we use k = 3, then the mass at 30 will also be mapped down to 0; but the mass at 60 will be mapped to 1, since 100/3 < 60 < 200/3.
- So again $a_3 = 0.5 * 0 + 0.5 * 1 = 0.5$, but now using normalization of n/k = 50 we have $e_2 = |45 - 50\ (0.5)| = |45 - 25| = 20$.

Theoretical characterization

Suppose scores are given by continuous pdf f (with cdf F) on $(0, 100)$, and we wish to compress them to two options, $\{0, 1\}$. Scores below 50 are mapped to 0, and scores above 50 are mapped to 1.

The average of the full distribution is

$$a_f = E[X] = \int_{x=0}^{100} x f(x) dx.$$

The average of the compressed version is

$$\begin{aligned} a_2 &= \int_{x=0}^{50} 0 f(x) dx + \int_{x=50}^{100} 1 f(x) dx = \int_{x=50}^{100} f(x) dx \\ &= F(100) - F(50) = 1 - F(50). \end{aligned}$$

So $e_2 = |a_f - 100(1 - F(50))| = |E[X] - 100 + 100F(50)|$. For three options,

$$\begin{aligned} a_3 &= \int_{x=0}^{100/3} 0 f(x) dx + \int_{x=100/3}^{200/3} 1 f(x) dx \\ &+ \int_{x=200/3}^{100} 2 f(x) dx \\ &= F(200/3) - F(100/3) + 2(1 - F(200/3)) \\ &= 2 - F(100/3) - F(200/3) \\ e_3 &= |a_f - 50(2 - F(100/3) - F(200/3))| \\ &= |E[X] - 100 + 50F(100/3) + 50F(200/3)| \end{aligned}$$

In general for n total and k compressed options,

$$\begin{aligned} a_k &= \sum_{i=0}^{k-1} \int_{x=\frac{ni}{k}}^{\frac{n(i+1)}{k}} i f(x) dx \\ &= \sum_{i=0}^{k-1} \left[i \left(F\left(\frac{n(i+1)}{k}\right) - F\left(\frac{ni}{k}\right) \right) \right] \\ &= (k-1)F(n) - \sum_{i=1}^{k-1} F\left(\frac{ni}{k}\right) \\ &= (k-1) - \sum_{i=1}^{k-1} F\left(\frac{ni}{k}\right) \end{aligned}$$

$$\begin{aligned} e_k &= \left| a_f - \frac{n}{k-1} \left((k-1) - \sum_{i=1}^{k-1} F\left(\frac{ni}{k}\right) \right) \right| \\ &= \left| E[X] - n + \frac{n}{k-1} \sum_{i=1}^{k-1} F\left(\frac{ni}{k}\right) \right| \end{aligned} \quad (3)$$

Equation 3 allows us to characterize the relative performance of choices of k for a given distribution f. For each k the characterization requires only knowing k statistics of f (the $k-1$ values of $F\left(\frac{ni}{k}\right)$ plus $E[X]$). In practice these could likely be closely approximated from historical data for small values of k.

- From this characterization, we see, for example, that $e_2 < e_3$ iff $|E[X] - 100 + 100\ F(50)| < |E[X] - 100 + 50F(100/3) + 50F(200/3)|$.
- If we happened to be in the case where both $a_2 <= a_f$ and $a_3 <= a_f$, then we could remove the absolute values and reduce the expression to see that $e_2 < e_3$ iff integral of f(x) from 100/3 to 50 is smaller than the integral from 50 to 200/3.
- Can perform more comprehensive analysis considering all cases to obtain better characterization and intuition for the optimal value of k for distributions with different properties.

Rounding compression

- Can modify the compression function s ← floor(s / (k/n)) to round s/(k/n) to nearest value.
- For example, for n = 100, k = 2, instead of dividing s by 50 and taking the floor, we could instead partition the points according to whether they are closest to $t_1 = 25$ or $t_2 = 75$. This would produce a compressed average score of $a_1 = 0.5*25 + 0.5*75 = 50$. No normalization would be necessary, and this would produce error of $e_2 = | a_f - a_2| = |45-50| = 5$, as the floor approach did as well. Similarly, for k = 3 the region midpoints will be $q_1 =100/6$, $q_2 = 50$, $q_3 = 500/6$. The mass at 30 will be mapped to q1, and the mass at 60 will be mapped to 2. This produces a compressed average score of $a_3= 0.5*100/6 + 0.5*50 = 100/3$. This produces an error of $|45-100/3| = 35/3 = 11.67$. Although the error for k=3 is smaller than for the floor case, it is still significantly larger than for k=2.

In general, this approach would create k "midpoints" $\{m_i^k\}$: $m_i^k = \frac{n(2i-1)}{2k}$. For $k = 2$ we have

$$\begin{aligned} a_2 &= \int_{x=0}^{50} 25 + \int_{x=50}^{100} 75 = 25F(50) + 75(1 - F(50)) = 75 - 50F(50) \\ e_2 &= |a_f - (75 - 50F(50))| = |E[X] - 75 + 50F(50)| \end{aligned}$$

One might wonder whether the floor approach would ever outperform the rounding approach (in the example above the rounding approach produced lower error $k = 3$ and the same error for $k = 2$). As a simple example to see that it can, consider the distribution with all mass on 0. The floor approach would produce $a_2 = 0$ giving an error of 0, while the rounding approach would produce $a_2 = 25$ giving an error of 25. Thus, the superiority of the approach is dependent on the distribution. We explore this further in the experiments.

For three options,

$$\begin{aligned} a_3 &= \int_0^{\frac{100}{3}} \frac{100}{6} f(x) + \int_{\frac{100}{3}}^{\frac{200}{3}} 50 f(x) + \int_{\frac{200}{3}}^{100} \frac{500}{6} f(x) = \frac{500}{6} - \frac{100}{3} F\left(\frac{100}{3}\right) - \frac{100}{3} F\left(\frac{200}{3}\right) \\ e_3 &= \left| E[X] - \frac{500}{6} + \frac{100}{3} F\left(\frac{100}{3}\right) + \frac{100}{3} F\left(\frac{200}{3}\right) \right| \end{aligned}$$

For general n and k, analysis as above yields

$$\begin{aligned} a_k &= \sum_{i=0}^{k-1} \int_{x=\frac{ni}{k}}^{\frac{n(i+1)}{k}} m_{i+1}^k f(x) dx = \frac{n(2k-1)}{2k} - \frac{n}{k} \sum_{i=1}^{k-1} F\left(\frac{ni}{k}\right) \\ e_k &= \left| a_f - \left[\frac{n(2k-1)}{2k} - \frac{n}{k} \sum_{i=1}^{k-1} F\left(\frac{ni}{k}\right) \right] \right| = \left| E[X] - \frac{n(2k-1)}{2k} + \frac{n}{k} \sum_{i=1}^{k-1} F\left(\frac{ni}{k}\right) \right| \end{aligned} \quad (4)$$

Computational Simulations and Analysis

- Used n = 100, and k = 2,3,4,5,10.
- For Gaussian model used s = 1000 (number of samples), $\mu = 50$, $\sigma = 50/3$.
- For each set of simulations we computed the errors for all considered values of k for m = 100,000 “items” (each corresponding to a different distribution generated according to the specified model).
- The main quantities we are interested in computing are the number of times that each value of k produces the lowest error over the m items (i.e., the “number of victories”), and the average value of the errors over all items for each k value.

Rounding compression simulations

- Not surprisingly, we see that the number of victories increases monotonically with the value of k. while the average error decreased monotonically (recall that we would have zero error if we set k = 100). However, using a smaller number of compressed scores produced the optimal error in a far from negligible number of the trials. For the uniform model, using 10 scores minimized error only around 53% of the time, while 5 scores minimized error 17% of the time, and even using 2 scores minimized it 5.6% of the time. The results were similar for the Gaussian model, though a bit more in favor of larger values of k, which is what we would expect because the Gaussian model is less likely to generate “fluke” distributions that could favor the smaller values.

Simulations for flooring compression

	2	3	4	5	10
Uniform # victories	5564	9265	14870	16974	53327
Uniform average error	1.32	0.86	0.53	0.41	0.19
Gaussian # victories	3025	7336	14435	17800	57404
Gaussian average error	1.14	0.59	0.30	0.22	0.10

Table 1: Number of times each value of k in {2,3,4,5,10} produces minimal error and average error values, over 100,000 items generated according to both models.

	2	3
Uniform number of victories	36805	63195
Uniform average error	1.31	0.86
Gaussian number of victories	30454	69546
Gaussian average error	1.13	0.58

Table 2: Number of times each value of k in {2,3} produces minimal error and average error values, over 100,000 items generated according to both generative models.

	2	10
Uniform number of victories	8253	91747
Uniform average error	1.32	0.19
Gaussian number of victories	4369	95631
Gaussian average error	1.13	0.10

Table 3: Number of times each value of k in {2,10} produces minimal error and average error values, over 100,000 items generated according to both generative models.

	2	10
Uniform number of victories	32250	67750
Uniform average error	1.31	0.74
Gaussian number of victories	10859	89141
Gaussian average error	1.13	0.20

Table 4: Number of times each value of k in {2,10} produces minimal error and average error values, over 100,000 items generated according to both models. For $k = 10$, we only permitted scores between 3 and 6 (inclusive). If a score was below 3 we set it to be 3, and above 6 to 6.

	2	10
Uniform number of victories	93226	6774
Uniform average error	1.31	0.74
Gaussian number of victories	54459	45541
Gaussian average error	1.13	1.09

Table 5: Number of times each value of k in {2,10} produces minimal error and average error values, over 100,000 items generated according to both generative models. For $k = 10$, we only permitted scores between 3 and 7 (inclusive). If a score was below 3 we set it to be 3, and above 7 to 7.

- Comparing just k = 2 vs. k =3, as expected k = 3 generally performed better, but surprisingly k = 2 produced a lower error 37% of the time. As before, the larger k value performs relatively better in the Gaussian model. We also looked at results for the most extreme comparison k = 2 vs k = 10. Using k = 2 outperformed 10 8.3% of the time in the uniform setting, which is larger than we expected.
- Next slide gives a specific distribution for which k = 2 particularly outperformed k = 10. Full distribution has mean 54.188, while k = 2 compression has mean 0.548 (54.253 after normalization) and k = 10 has mean 5.009 (55.009 after normalization). The normalized errors between the means were 0.906 for k=10 and 0.048 for k = 2, yielding a difference of 0.859 in favor of k=2.

Example where k=2 significantly outperforms k=10

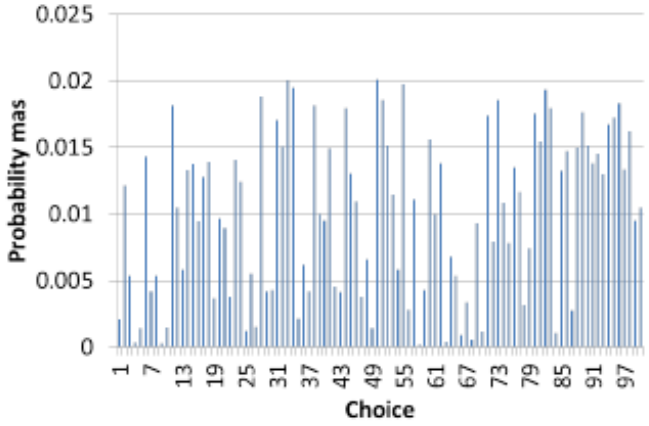

Figure 8: Example distribution for which compressing with $k = 2$ produces significantly lower error than $k = 10$. The full distribution has mean 54.188, while the $k = 2$ compression has mean 0.548 (54.253 after normalization) and the $k = 10$ compression has mean 5.009 (55.009 after normalization). The normalized errors between the means were 0.906 for $k = 10$ and 0.048 for $k = 2$, yielding a difference of 0.859 in favor of $k = 2$.

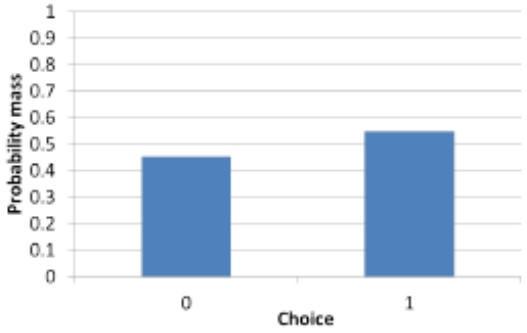

Figure 9: Compressed distribution for $k = 2$.

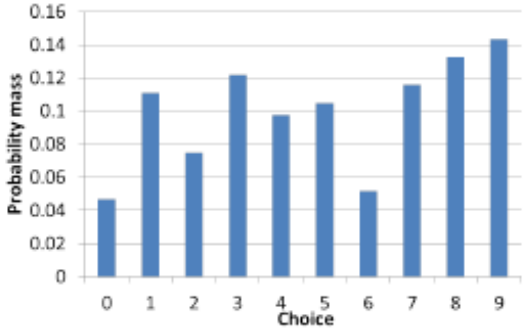

Figure 10: Compressed distribution for $k = 10$.

- We next repeated the extreme k = 2 vs. 10 comparison, with a restriction that k = 10 could not give a score below 3 or above 6 (if below 3 set to 3 and if above 6 set to 6). For instance for paper reviewing, extreme scores are very uncommon, and we suspect the vast majority of scores are in the middle range.
- Some possible explanations are that reviewers who give extreme scores may be required to put in additional work to justify their scores, and are more likely to be involved in arguments with the other reviewers (or with the authors in the rebuttal). Reviewers could also experience higher regret or embarrassment for being "wrong" and possibly off-base in the review by missing an important nuance.
- In this setting using k=2 outperforms k=10 nearly 1/3 of the time in the uniform model.

- We also consider k = 10 scores within 3 and 7 (as opposed to 3-6). Note that the possible scores range from 0-9, so this restriction is asymmetric in that the lowest three possible scores are eliminated while only the highest two are. This is motivated by the intuition that raters may be less inclined to give extremely low scores which may hurt the feelings of an author (for the case of paper reviewing).
- In this setting, which is seemingly quite similar to the 3-6 setting, k=2 produced lower error 93% of the time in the uniform model!

- We next repeated these experiments for rounding compression. In this setting, k=3 is the clear choice, performing best in both models (by a large margin for the Gaussian model). The smaller values of k perform significantly better with rounding than flooring (as indicated by lower errors) while the larger values perform significantly worse, and their errors seem to approach 0.5 for both models.

- Taking both compressions into account, the optimal overall approach would still be to use flooring with k=10, which produced the smallest average errors of 0.19 and 0.1 in the two models, while using k=3 with rounding produced errors of 0.47 and 0.24.
- The 2 vs. 3 experiments produced very similar results for the two compressions.
- The 2 vs. 10 results were quite different, with 2 performing better almost 40% of the time with rounding, vs. less than 10% with flooring.
- In the 2 vs. 10 truncated 3-6 experiments, k=2 performed relatively better with rounding for both generative models.
- For the 2 vs. 10 truncated 3-7 experiments k=2 performed better nearly all the time.

Simulations for rounding compression

	2	3	4	5	10
Uniform # victories	15766	33175	21386	19995	9678
Uniform average error	0.78	0.47	0.55	0.52	0.50
Gaussian # victories	13262	64870	10331	9689	1848
Gaussian average error	0.67	0.24	0.50	0.50	0.50

Table 6: Number of times each value of k in {2,3,4,5,10} produces minimal error and average error values, over 100,000 items generated according to both models with rounding compression.

	2	3
Uniform number of victories	33585	66415
Uniform average error	0.78	0.47
Gaussian number of victories	18307	81693
Gaussian average error	0.67	0.24

Table 7: Number of times each value of k in {2,3} produces minimal error and average error values, over 100,000 items generated according to both generative models with rounding compression.

	2	10
Uniform number of victories	37225	62775
Uniform average error	0.78	0.50
Gaussian number of victories	37897	62103
Gaussian average error	0.67	0.50

Table 8: Number of times each value of k in {2,10} produces minimal error and average error values, over 100,000 items generated according to both generative models with rounding compression.

	2	10
Uniform number of victories	55676	44324
Uniform average error	0.79	0.89
Gaussian number of victories	24128	75872
Gaussian average error	0.67	0.34

Table 9: Number of times each value of k in {2,10} produces minimal error and average error values, over 100,000 items generated according to both models with rounding compression. For $k = 10$, we only permitted scores between 3 and 6 (inclusive). If a score was below 3 we set it to be 3, and above 6 to 6.

	2	10
Uniform number of victories	99586	414
Uniform average error	0.78	3.50
Gaussian number of victories	95692	4308
Gaussian average error	0.67	1.45

Table 10: Number of times each value of k in {2,10} produces minimal error and average error values, over 100,000 items generated according to both generative models with rounding compression. For $k = 10$, we only permitted scores between 3 and 7 (inclusive). If a score was below 3 we set it to be 3, and above 7 to 7.

Experiments

- Explored data from www.preflib.data on hotel ratings and French presidential elections.
- Trip advisor ratings in several categories from 1-5 (n = 5), we used k = 2, 3, 4.
- Average error generally decreases monotonically as k increases, as we would expect, though sometimes lower k outperformed higher k (both with respect to number of victors and average error).
- French presidential candidates rated from 0-20 (n= 20), we used k = 2, 3, 4, 5, 8, 10).
- Again the error generally decreased monotonically as k increased, though for one case it was minimized for k = 2.

Future research

- Extend theoretical characterization analysis to get better understanding of the specific distributions for which different values of k are optimal.
  - Our experimental simulation results are in aggregate over many different distributions.
- Specific application domains will have distributions with different properties, and improved understanding will allow us to determine which k is optimal for the types of distributions we expect to encounter.
- This improved theoretical understanding can be coupled with further exploration of data.