Recommender Systems
Project: study on the introduction of a bias term in the cosine similarity for KNN recommender systems

Paolo Cudrano

May 15, 2019

1 Introduction

The cosine similarity is a measure often used in KNN recommender systems, applied to both ICMs in content-based filtering (CBF) systems and URMs in collaborative filtering (CF) techniques.

This similarity measure evaluates the cosine of the angular distance between two vectors in a specified feature space.

Preliminary results behind this study showed that, at least in particular cases, the introduction of a constant bias term, added to each feature, could increase the performances of the recommendations provided by the system. Thus, we set out to investigate this hypothesis and analyze the effects of such bias on the similarity measure and, ultimately, on the overall performances of the recommender system adopted.

Notice that in the rest of this section we will refer for simplicity only to content-based systems, where the cosine similarity is applied to feature vectors belonging to the ICM. Nevertheless, every computation can be carried analogously on URM vectors for collaborative filtering systems.

2 Previous work

The literature on the cosine similarity and its usage ranges over a variety of topics and applications, from information retrieval to natural language processing, and from data mining to recommender systems. However, despite such large body of research, still no interest has been put towards the analysis of how a bias term could influence its property as a similarity measure.

Among the papers we consulted, the most pertinent to our research is [1], which analyzes the relation between the cosine similarity and the Euclidean distance, together with its implications in several data mining tasks.

Another noticeable resource is [2], which explicitly reports among the properties of the cosine similarity that of being *translation non-invariant.*

3 Theoretical analysis

3.1 Cosine similarity and bias term

Given two vectors $\mathbf{a}$ and $\mathbf{b}$, their cosine similarity is computed as

$$cos\_sim\left(\mathbf{a},\mathbf{b}\right) = \frac{\mathbf{a} \bullet \mathbf{b}}{\|\mathbf{a}\|_2 \|\mathbf{b}\|_2} \tag{1}$$

We consider adding a constant bias term to each of the values of the ICM, thus translating each feature vector it contains of a constant amount along *each* dimension of the feature space. In general, this translation causes changes to both the module and the direction of each feature vector, through a non-linear map. As a direct consequence, also the angles between each pair of feature vectors is modified, and with them their cosine similarities.

Assuming the feature space has n dimensions, we can represent the bias term as a vector $\boldsymbol{\delta} = (\delta, \delta, \dots \delta) \in \mathbb{R}^n$. Thus, feature vectors from the ICM $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ would become $\mathbf{a} + \boldsymbol{\delta}, \mathbf{b} + \boldsymbol{\delta} \in \mathbb{R}^n$, and their similarity measure

$$cos\_sim\left(\mathbf{a}+\boldsymbol{\delta},\mathbf{b}+\boldsymbol{\delta}\right) = \frac{\left(\mathbf{a}+\boldsymbol{\delta}\right) \bullet \left(\mathbf{b}+\boldsymbol{\delta}\right)}{\|\left(\mathbf{a}+\boldsymbol{\delta}\right)\|_2 \|\left(\mathbf{b}+\boldsymbol{\delta}\right)\|_2} \tag{2}$$

3.2 Analytical comparison

Manipulating the components of Eq. 2 we can find

$$\begin{aligned}
\left(\mathbf{a}+\boldsymbol{\delta}\right) \bullet \left(\mathbf{b}+\boldsymbol{\delta}\right) &= \sum_i (a_i + \delta_i)(b_i + \delta_i) \\
&= \sum_i a_i b_i + \delta \sum_i a_i + \delta \sum_i b_i + \sum_i \delta^2 \\
&= \sum_i a_i b_i + \delta \left(\sum_i a_i + \sum_i b_i \right) + \sum_i \delta^2 \\
&= \mathbf{a} \bullet \mathbf{b} + \delta \left(\|\mathbf{a}\|_1 + \|\mathbf{b}\|_1 \right) + n\delta^2
\end{aligned} \tag{3}$$

and for $\mathbf{v} \in \mathbb{R}^n$

$$\begin{aligned}
\|\left(\mathbf{v}+\boldsymbol{\delta}\right)\|_2 &= \sqrt{\sum_i (v_i + \delta)^2} \\
&= \sqrt{\sum_i \left(v_i^2 + 2\delta v_i + \delta^2\right)} \\
&= \sqrt{\sum_i v_i^2 + 2\delta \sum_i v_i + n\delta^2} \\
&= \sqrt{\|\mathbf{v}\|_2^2 + 2\delta \|\mathbf{v}\|_1 + n\delta^2}
\end{aligned} \tag{4}$$

therefore

$$
\begin{aligned}
\mathit{cos\_sim}\left(\mathbf{a}+\boldsymbol{\delta},\mathbf{b}+\boldsymbol{\delta}\right) &= \frac{\left(\mathbf{a}+\boldsymbol{\delta}\right)\bullet\left(\mathbf{b}+\boldsymbol{\delta}\right)}{\|\left(\mathbf{a}+\boldsymbol{\delta}\right)\|_2\|\left(\mathbf{b}+\boldsymbol{\delta}\right)\|_2} \\
&= \frac{\mathbf{a}\bullet\mathbf{b}+\delta\left(\|\mathbf{a}\|_1+\|\mathbf{b}\|_1\right)+n\delta^2}{\sqrt{\|\mathbf{a}\|_2^2+2\delta\|\mathbf{a}\|_1+n\delta^2}\sqrt{\|\mathbf{b}\|_2^2+2\delta\|\mathbf{b}\|_1+n\delta^2}} \\
&= \frac{\mathbf{a}\bullet\mathbf{b}}{\|\left(\mathbf{a}+\boldsymbol{\delta}\right)\|_2\|\left(\mathbf{b}+\boldsymbol{\delta}\right)\|_2}+\frac{\delta\left(\|\mathbf{a}\|_1+\|\mathbf{b}\|_1\right)+n\delta^2}{\|\left(\mathbf{a}+\boldsymbol{\delta}\right)\|_2\|\left(\mathbf{b}+\boldsymbol{\delta}\right)\|_2} \\
&= \frac{\mathbf{a}\bullet\mathbf{b}}{\|\left(\mathbf{a}+\boldsymbol{\delta}\right)\|_2\|\left(\mathbf{b}+\boldsymbol{\delta}\right)\|_2}+\delta\frac{\|\mathbf{a}\|_1+\|\mathbf{b}\|_1+n\delta}{\|\left(\mathbf{a}+\boldsymbol{\delta}\right)\|_2\|\left(\mathbf{b}+\boldsymbol{\delta}\right)\|_2} \\
&= \frac{\mathbf{a}\bullet\mathbf{b}}{\|\left(\mathbf{a}+\boldsymbol{\delta}\right)\|_2\|\left(\mathbf{b}+\boldsymbol{\delta}\right)\|_2}+\delta\frac{\sum_i a_i+\sum_i b_i+\sum_i\delta}{\|\left(\mathbf{a}+\boldsymbol{\delta}\right)\|_2\|\left(\mathbf{b}+\boldsymbol{\delta}\right)\|_2} \\
&= \frac{\mathbf{a}\bullet\mathbf{b}}{\|\left(\mathbf{a}+\boldsymbol{\delta}\right)\|_2\|\left(\mathbf{b}+\boldsymbol{\delta}\right)\|_2}+\delta\frac{\sum_i\left(a_i+b_i+\delta\right)}{\|\left(\mathbf{a}+\boldsymbol{\delta}\right)\|_2\|\left(\mathbf{b}+\boldsymbol{\delta}\right)\|_2}
\end{aligned}
\tag{5}
$$

Unfortunately however, even after these simplifications it is not trivial to study the behavior of this function. We can clearly see how the first highlighted component has the same form of the original $\mathit{cos\_sim}\left(\mathbf{a},\mathbf{b}\right)$, shrunk by the presence of $\boldsymbol{\delta}$ in the norms at the denominator. If this addend is smaller than $\mathit{cos\_sim}\left(\mathbf{a},\mathbf{b}\right)$ however, we could not find any non-trivial bound for the second term, and even more importantly we were not able to determine how much this term would impact on the final result with respect to the shrinkage present in the first component. For this reason, we could not establish analytically whether (and when) $\mathit{cos\_sim}\left(\mathbf{a},\mathbf{b}\right) \lesseqgtr \mathit{cos\_sim}\left(\mathbf{a}+\boldsymbol{\delta},\mathbf{b}+\boldsymbol{\delta}\right)$.

We must point out anyway that, from a graphical analysis, it is clear how, in most cases, the introduction of a bias would make the vector shrink toward the center, leading to a shrinkage of their angular distance and, therefore, a raise of their cosine similarity. Such line of reasoning can be better analyzed through the study of special or boundary cases, as the ones described below, together with the observation of the pictures provided.

3.2.1 Special case 1: orthogonal versors

We consider the simple case of two vectors oriented along two different axes of the feature space and having both unitary module, as depicted in Figure 1.

We can represent them *wlog* as $\mathbf{a}=(1,0,\ldots,0),\mathbf{b}=(0,1,\ldots,0)\in\mathbb{R}^n$. Since they are orthogonal, their cosine similarity is trivially

$$
\mathit{cos\_sim}\left(\mathbf{a},\mathbf{b}\right)=0 \tag{6}
$$

while for $\mathbf{a}+\boldsymbol{\delta}=(1+\delta,\delta,\ldots,\delta),\mathbf{b}+\boldsymbol{\delta}=(\delta,1+\delta,\ldots,\delta)$ we have

$$
\begin{aligned}
\mathit{cos\_sim}\left(\mathbf{a}+\boldsymbol{\delta},\mathbf{b}+\boldsymbol{\delta}\right) &= \frac{\left(\mathbf{a}+\boldsymbol{\delta}\right)\bullet\left(\mathbf{b}+\boldsymbol{\delta}\right)}{\|\left(\mathbf{a}+\boldsymbol{\delta}\right)\|_2\|\left(\mathbf{b}+\boldsymbol{\delta}\right)\|_2} \\
&= \frac{\sum_i(a_i+\delta)(b_i+\delta)}{\sqrt{\sum_i a_i^2+2\delta\sum_i a_i+n\delta^2}\sqrt{\sum_i b_i^2+2\delta\sum_i b_i+n\delta^2}} \\
&= \frac{2\delta(1+\delta)+(n-2)\delta^2}{\sqrt{1+2\delta++n\delta^2}\sqrt{1+2\delta++n\delta^2}} \\
&= \frac{n\delta^2+2\delta}{n\delta^2+2\delta+1}
\end{aligned}
\tag{7}
$$

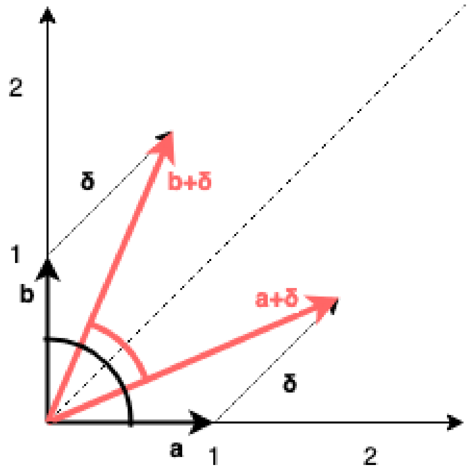

Figure 1: 2D example of special case 1

As depicted also in Figure 2, the similarity is trivially 0 when $\delta = 0$ and it increases rapidly until it soon saturates at 1. We can notice also how the saturation occurs faster for increasing number of dimensions of the feature space.

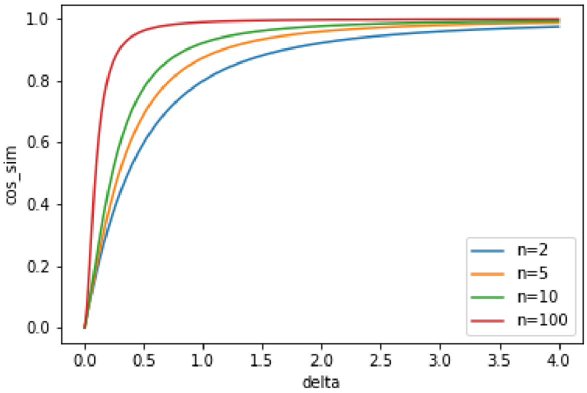

Figure 2: Plot of the cosine similarity for special case 1, with different values of n

3.2.2 Special case 2: versor along an axis and pure diagonal (all-ones) vector

We consider here the situation where one of the vectors is the pure diagonal (or all-ones) vector, while the other remains on one of the axis. As in the previous special case, the two vectors are normalized. The situation is depicted in Figure 3.

We can represent *wlog* the two vectors as $\mathbf{a} = (1, 0, \ldots, 0), \mathbf{b} = (1, 1, \ldots, 1) \in \mathbb{R}^n$. Their cosine similarity is trivially

$$cos\_sim\left(\mathbf{a}, \mathbf{b}\right) = \frac{\mathbf{a} \bullet \mathbf{b}}{\|\mathbf{a}\|_2 \|\mathbf{b}\|_2} = \frac{1}{\sqrt{1} \cdot \sqrt{n}} = \frac{1}{\sqrt{n}} \tag{8}$$

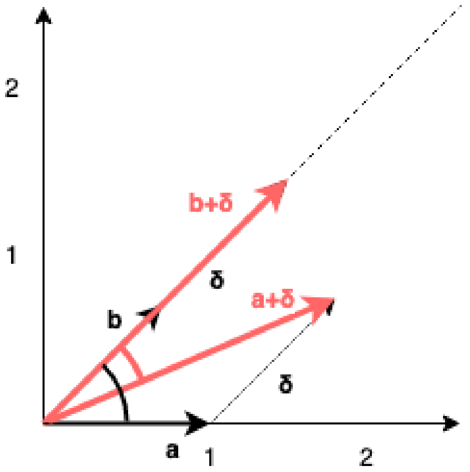

Figure 3: 2D example of special case 2

Then, with $\mathbf{a}+\boldsymbol{\delta} = (1+\delta, \delta, \ldots, \delta), \mathbf{b}+\boldsymbol{\delta} = (1+\delta, 1+\delta, \ldots, 1+\delta)$ we have

$$
\begin{aligned}
cos\_sim\left(\mathbf{a}+\boldsymbol{\delta}, \mathbf{b}+\boldsymbol{\delta}\right) &= \frac{(\mathbf{a}+\boldsymbol{\delta}) \bullet (\mathbf{b}+\boldsymbol{\delta})}{\parallel (\mathbf{a}+\boldsymbol{\delta}) \parallel_2 \parallel (\mathbf{b}+\boldsymbol{\delta}) \parallel_2} \\
&= \frac{1 \cdot (1+\delta^2) + (n-1) \cdot \delta(1+\delta)}{\sqrt{1 \cdot (1+\delta)^2 + (n-1) \cdot \delta^2} \cdot \sqrt{n \cdot (1+\delta)^2}} \\
&= \frac{1 + 2\delta + \delta^2 + n\delta + n\delta^2 - \delta^2 - \delta^2}{\sqrt{1 + 2\delta + \delta^2 + (n-1)\delta^2} \cdot \sqrt{n}(1+\delta)} \\
&= \frac{1}{\sqrt{n}} \cdot \frac{n\delta^2 + (n+1)\delta + 1}{(1+\delta)\sqrt{n\delta^2 + 2\delta + 1}} \\
&= cos\_sim\left(\mathbf{a}, \mathbf{b}\right) \cdot \frac{n\delta^2 + (n+1)\delta + 1}{(1+\delta)\sqrt{n\delta^2 + 2\delta + 1}}
\end{aligned}
\tag{9}
$$

From Eq. 9 we can see how the second factor acts as a magnifier and amplifies the magnitude of the original cosine similarity. We can observe this effect in Figures 4 and 5.

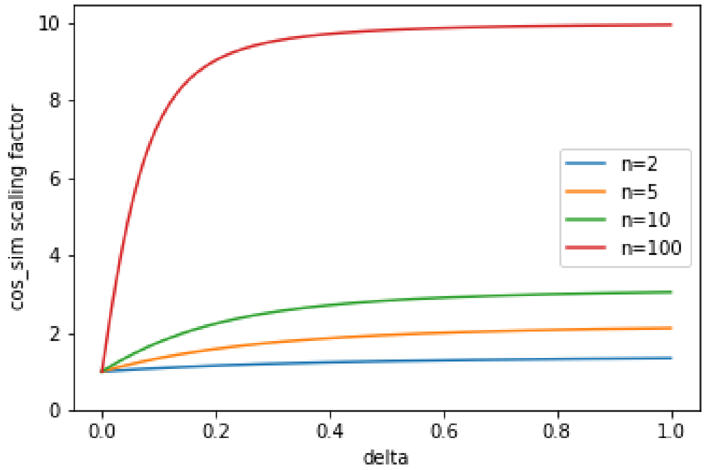

Figure 4: Scaling factor introduced in the cosine similarity with the bias term. We can clearly see how the value of the cosine similarity is amplified significantly as the number of dimensions increases.

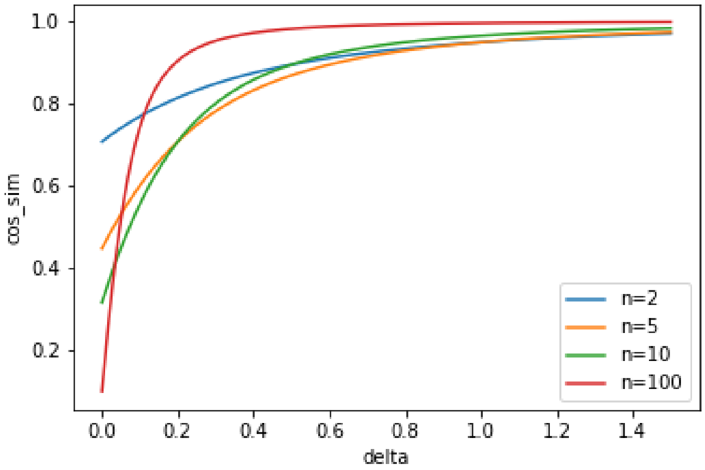

Figure 5: Value of the cosine similarity after the introduction of a bias term. Notice that the value of the cosine similarity without bias term corresponds to the value for $delta = 0$. We can see then how the introduction of the bias rapidly increases the similarity value until it soon saturates to 1.

3.3 Numerical comparison

Given the difficulties encountered in the analytical analysis and the consequent lack of results, we proceed to its study from a numerical perspective.

For the results shown in this sections as well as for further information on how they were obtained, we refer to our Python notebook `bias_effect_analysis`, which contains the python code used to generate them, together with some comments on the outcomes.

We conducted our analysis in two stages: at first, we studied the effect of a variable bias only on unitary vectors, and then, with the appropriate considerations, we relaxed such constraint and studied the most general case. The analysis is conducted only in a 2-dimensional space, to be able to graphically visualize the results.

3.3.1 Effect of bias on unitary vectors

We consider two unitary vectors with orientation varying within the first quadrant (i.e. with positive components). We then add to them a constant bias and visualize the values of their cosine similarity through a heat-map, where each row represents a possible orientation of the first vector, and each column an orientation of the second. We repeat then this experiment for different values of the bias.

Formally, we consider vectors in polar coordinate $\mathbf{v_1} = (\rho_1, \theta_1) = (1, \tilde{\theta_1}), \mathbf{v_2} = (\rho_2, \theta_2) = (1, \tilde{\theta_2})$. We then add to them bias $\boldsymbol{\delta} = (\delta, 45°)$. For $\delta \in \{0, 0.1, \ldots, 0.9, 1.0\}$, a heat-map describes the value of $cos\_sim(\mathbf{v_1} + \boldsymbol{\delta}, \mathbf{v_2} + \boldsymbol{\delta})$.

Figure 6 shows the results obtained and highlights how the introduction of the bias term increases the similarity regardless of the orientation of each vector, rapidly saturating it to values close to 1. This is as to say that the two vectors are rapidly pulled closer to each other. Since it is not clear from the plots how fast and significant this process is, we report in Table 1 the minimum and maximum values of the similarities registered in each of the heat-maps generated.

δ	cos_sim interval
0.0	[0.00, 1.00]
0.1	[0.18, 1.00]
0.2	[0.32, 1.00]
0.3	[0.44, 1.00]
0.4	[0.53, 1.00]
0.5	[0.60, 1.00]
0.6	[0.66, 1.00]
0.7	[0.70, 1.00]
0.8	[0.74, 1.00]
0.9	[0.77, 1.00]
1.0	[0.80, 1.00]

Table 1: Minimum and maximum cosine similarity registered for different values of the bias term.

3.3.2 Effect of bias on (almost) any pair of vectors

To study a more general case and still be able to easily visualize the results obtained, we have to first impose some constraints, since the variables to be otherwise considered would be 5: ρ_1, θ_1, ρ_2, θ_2 and δ.

However, since the computation of the cosine similarity involves a normalization on the norms of the two vectors, we can maintain a unitary length for one of the vectors, and vary only the norm of the other

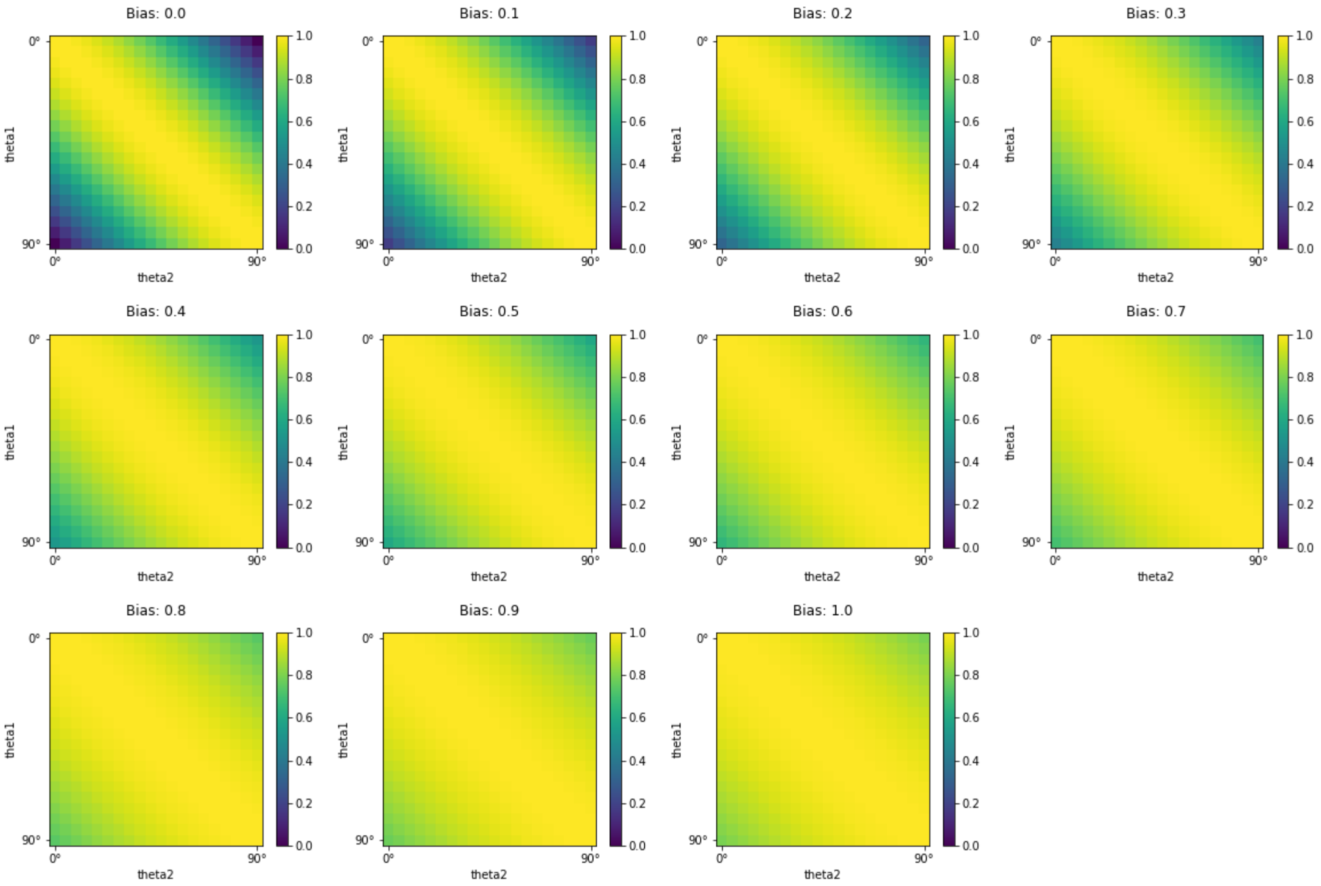

Figure 6: Numerical analysis of the effect of an increasing bias term to the cosine similarity of two unitary vectors. The similarity is shown through a heat-map, where yellow corresponds to the highest value and blue to the lowest. The two axes represent the orientation of the two vectors before the bias was added, from 0° to 90°.

(together with, of course, the value of δ). Notice that this simplification maintains generality when no bias term is added, but it can be shown instead that it induces a partial loss of generality when $\delta > 0$. Nevertheless, thanks to this single constraint, we are able to obtain clear representations of the phenomena in a large majority of cases.

For this study we proceeded as in the previous analysis, but varying also the length of $\mathbf{v_2}$ at discrete points. This generates a discrete number of cases for each value of the bias term. Formally, we consider vectors in polar coordinate $\mathbf{v_1} = (\rho_1, \theta_1) = (1, \tilde{\theta_1}), \mathbf{v_2} = (\rho_2, \theta_2) = (\tilde{\rho_2}, \tilde{\theta_2})$. We then add to them the bias $\boldsymbol{\delta} = (\delta, 45°)$. We display a series of heat-maps for $\delta \in \{0, 0.25, 0.50, 0.75, 1.0\}$. Each series represents the similarity values $cos\_sim(\mathbf{v_1} + \boldsymbol{\delta}, \mathbf{v_2} + \boldsymbol{\delta})$ for ρ_2 in an evenly distributed interval from 1 to 10 (i.e. up to 10 times larger than ρ_1).

The results are displayed in Figure 7. Notice that the scale of each picture is adapted to make the pictures comprehensible: indeed, fixing the scale would have resulted in almost homogeneous plots for high values of δ.

It is interesting to notice how, in the first row, the variation of the ratio between the length of the two vectors ($\rho_2/\rho_1 \equiv \rho_2$) does not influence their similarity, which remains distributed with the same shape. The peak similarity is, in this case, on the line $\theta_1 = \theta_2$, as expected.

With the introduction of a bias term instead, we notice at first a global increment of the similarity values over the whole heat-map, as highlighted by the first column, almost until saturation. This increment is slightly reduced when ρ_2 is increased, although the values remain close to saturation.

Furthermore, an interesting phenomena arises. When the effect of the bias combines with the difference in length ratio, it causes a rotation of the line where the similarity is maximum. Notice moreover how, for large values of ρ_2 and δ, such line tends to become vertical. As a result, in these cases, the value of the similarity between the two vectors becomes almost completely independent of the orientation of the shorter one ($\mathbf{v_1}$), and almost completely determined by $\mathbf{v_2}$ and, in particular, θ_2.

We must point out that this phenomena is potentially very dangerous, because it disrupts the original purpose of the cosine similarity and its property of being a similarity measure in the first place.

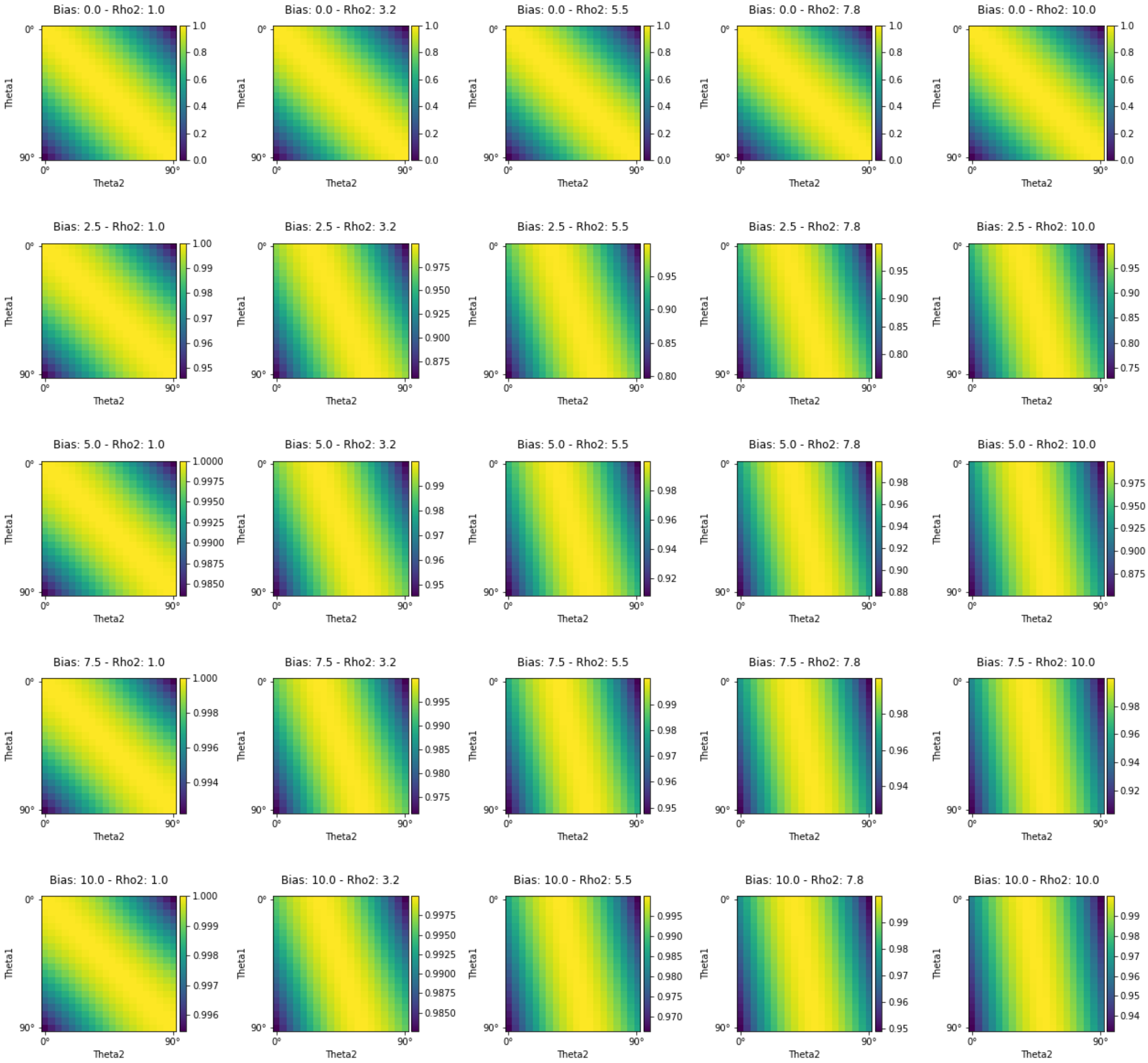

Figure 7: Numerical analysis of the effect of an increasing bias term to the cosine similarity between a unitary vector and a vector of length ρ_2, variable. The similarity is shown through a heat-map, where yellow corresponds to the highest value and blue to the lowest. The two axes represent the orientation of each vector before the bias was added, from 0° to 90°. Notice that, differently from Figure 6, each heat-map is here scaled to comfortably highlight its behavior. For this reason, each plot reports a different color-bar, which needs to be carefully minded for a correct analysis of their content.

4 Experiments and results

Given these initial observations on the behavior of the cosine similarity between two general feature vectors, we shift our attention on the real issue under analysis: the effect of the bias term on the final recommendations of a content-based KNN recommender system.

In this context, the cosine similarity is computed on the feature vectors belonging to an ICM (Item Content Matrix). The values assumed by such matrix could be of various types, depending on how they were collected and on their intrinsic meaning. Often also feature weighting techniques intended to normalize such values are applied.

For our experiments, we considered several popular datasets, some of which containing multiple ICMs. Table 2 reports a complete list of them, along with some additional information on their content.

Dataset	**ICM name**	**ICM type**	**ICM values range**
Movielens1M	ICM_genres	binary	{0,1}
Movielens10M	ICM_all	integer	{0,...,69}
Movielens10M	ICM_genres	binary	{0,1}
Movielens10M	ICM_tags	integer	{0,...,69}
Movielens20M	ICM_all	integer	{0,...,257}
Movielens20M	ICM_genres	binary	{0,1}
Movielens20M	ICM_tags	integer	{0,...,257}
BookCrossing	ICM_book_crossing	integer	{0,...,7}
LastFMHetrec2011	ICM_tags	integer	{0,...,108}
TheMoviesDataset	ICM_all	integer	{0,...,10}
TheMoviesDataset	ICM_credits	integer	{0,...,10}
TheMoviesDataset	ICM_metadata	integer	{0,...,3}
CiteULike_a	ICM_title_abstract	float	[0,1]
CiteULike_t	ICM_title_abstract	float	[0,1]

Table 2: List of datasets used in our experiments and information on the ICMs they contain. Particularly interesting for our study, the interval of values assumed by the elements of each ICM is shown.

To perform our analysis, for each of these datasets we tuned a content-based filtering recommender system and evaluated its performances on both a validation set and, at the end of the tuning, a test set. Such procedure was carried out first using the original ICMs contained in the datasets, and successively adding to them a variable bias term. Moreover, to take into account also the scaling introduced by feature weighting methods, each of our experiments is performed in all of the three following configurations: without feature weighting, applying TF-IDF and applying BM25. This leads to 6 possible tuning configurations for each ICM considered:

- original ICM
- original ICM + bias
- ICM weighted by TF-IDF
- ICM weighted by TF-IDF + bias
- ICM weighted by BM25
- ICM weighted by BM25 + bias

For each experiment, the parameter tuning was carried out using a Bayesian optimizer. The parameters considered were:

- `topk`: k parameter of the K-NN technique (number of nearest neighbors to consider)
- `shrink`: shrink term for the cosine similarity
- `normalize`: boolean, whether the denominator of the cosine similarity (normalizing term) has to be used or not
- `bias`: bias term, when required.

Detailed results for our experiments are listed in Table 3 and Table 4, showing respectively the performances of each recommender system registered and the parameters characterizing them.

Table 3: Results obtained with our experiments, highlighting the performances of each system built on validation and test set.

Dataset	ICM	ICM range	F.Weight	Bias	ValidMAP@10	TestMAP@10
BookCrossing	ICM_book_crossing	{0,...,7}	BM25	No	0.010251	0.009226
BookCrossing	ICM_book_crossing	{0,...,7}	BM25	Yes	0.010240	0.009875
BookCrossing	ICM_book_crossing	{0,...,7}	TF-IDF	No	0.010223	0.009853
BookCrossing	ICM_book_crossing	{0,...,7}	TF-IDF	Yes	0.010063	0.009830
BookCrossing	ICM_book_crossing	{0,...,7}	none	No	0.008640	0.008375
BookCrossing	ICM_book_crossing	{0,...,7}	none	Yes	0.009119	0.009147
CiteULike_a	ICM_title_abstract	[0,1]	BM25	No	0.051059	0.046741
CiteULike_a	ICM_title_abstract	[0,1]	BM25	Yes	0.052783	0.052097
CiteULike_a	ICM_title_abstract	[0,1]	TF-IDF	No	0.051135	0.046895
CiteULike_a	ICM_title_abstract	[0,1]	TF-IDF	Yes	0.049879	0.049084
CiteULike_a	ICM_title_abstract	[0,1]	none	No	0.048048	0.044385
CiteULike_a	ICM_title_abstract	[0,1]	none	Yes	0.047497	0.044205
CiteULike_t	ICM_title_abstract	[0,1]	BM25	No	0.032809	0.032187
CiteULike_t	ICM_title_abstract	[0,1]	BM25	Yes	0.039994	0.039119
CiteULike_t	ICM_title_abstract	[0,1]	TF-IDF	No	0.038720	0.036471
CiteULike_t	ICM_title_abstract	[0,1]	TF-IDF	Yes	0.038943	0.038781
CiteULike_t	ICM_title_abstract	[0,1]	none	No	0.030566	0.029536
CiteULike_t	ICM_title_abstract	[0,1]	none	Yes	0.039558	0.038122
LastFMHetrec2011	ICM_tags	{0,...,108}	BM25	No	0.057743	0.059243
LastFMHetrec2011	ICM_tags	{0,...,108}	BM25	Yes	0.084533	0.078435
LastFMHetrec2011	ICM_tags	{0,...,108}	TF-IDF	No	0.086732	0.084632
LastFMHetrec2011	ICM_tags	{0,...,108}	TF-IDF	Yes	0.085029	0.084686
LastFMHetrec2011	ICM_tags	{0,...,108}	none	No	0.070228	0.071444
LastFMHetrec2011	ICM_tags	{0,...,108}	none	Yes	0.072048	0.072490
Movielens10M	ICM_all	{0,...,69}	BM25	No	0.026939	0.026724
Movielens10M	ICM_all	{0,...,69}	BM25	Yes	0.033527	0.033924
Movielens10M	ICM_all	{0,...,69}	TF-IDF	No	0.033212	0.033374
Movielens10M	ICM_all	{0,...,69}	TF-IDF	Yes	0.034606	0.035339
Movielens10M	ICM_all	{0,...,69}	none	No	0.025202	0.024607
Movielens10M	ICM_all	{0,...,69}	none	Yes	0.033556	0.034330
Movielens10M	ICM_genres	{0,1}	BM25	No	0.003416	0.003064
Movielens10M	ICM_genres	{0,1}	BM25	Yes	0.002666	0.002689
Movielens10M	ICM_genres	{0,1}	TF-IDF	No	0.003258	0.003334
Movielens10M	ICM_genres	{0,1}	TF-IDF	Yes	0.002517	0.002372
Movielens10M	ICM_genres	{0,1}	none	No	0.002923	0.002685
Movielens10M	ICM_genres	{0,1}	none	Yes	0.003038	0.002833

Continued on next page...

Table 3: Results obtained with our experiments. (continued)

Movielens10M	ICM_tags	{0,...,69}	BM25	No	0.026128	0.025839
Movielens10M	ICM_tags	{0,...,69}	BM25	Yes	0.033792	0.034499
Movielens10M	ICM_tags	{0,...,69}	TF-IDF	No	0.032960	0.033134
Movielens10M	ICM_tags	{0,...,69}	TF-IDF	Yes	0.034439	0.035484
Movielens10M	ICM_tags	{0,...,69}	none	No	0.025630	0.025119
Movielens10M	ICM_tags	{0,...,69}	none	Yes	0.033599	0.034449
Movielens1M	ICM_genres	{0,1}	BM25	No	0.009126	0.008907
Movielens1M	ICM_genres	{0,1}	BM25	Yes	0.010198	0.009412
Movielens1M	ICM_genres	{0,1}	TF-IDF	No	0.010060	0.009837
Movielens1M	ICM_genres	{0,1}	TF-IDF	Yes	0.011124	0.009715
Movielens1M	ICM_genres	{0,1}	none	No	0.010956	0.010235
Movielens1M	ICM_genres	{0,1}	none	Yes	0.011045	0.008082
Movielens20M	ICM_all	{0,...,257}	BM25	No	0.031282	0.031045
Movielens20M	ICM_all	{0,...,257}	BM25	Yes	0.047448	0.047252
Movielens20M	ICM_all	{0,...,257}	TF-IDF	No	0.036214	0.035777
Movielens20M	ICM_all	{0,...,257}	TF-IDF	Yes	0.045837	0.045362
Movielens20M	ICM_all	{0,...,257}	none	No	0.021509	0.021987
Movielens20M	ICM_all	{0,...,257}	none	Yes	0.046875	0.046683
Movielens20M	ICM_genres	{0,1}	BM25	No	0.002002	0.001968
Movielens20M	ICM_genres	{0,1}	BM25	Yes	0.001750	0.001880
Movielens20M	ICM_genres	{0,1}	TF-IDF	No	0.002516	0.002732
Movielens20M	ICM_genres	{0,1}	TF-IDF	Yes	0.002309	0.002321
Movielens20M	ICM_genres	{0,1}	none	No	0.002238	0.002301
Movielens20M	ICM_genres	{0,1}	none	Yes	0.002029	0.001679
Movielens20M	ICM_tags	{0,...,257}	BM25	No	0.030855	0.030590
Movielens20M	ICM_tags	{0,...,257}	BM25	Yes	0.046330	0.045954
Movielens20M	ICM_tags	{0,...,257}	TF-IDF	No	0.036260	0.035792
Movielens20M	ICM_tags	{0,...,257}	TF-IDF	Yes	0.046428	0.045993
Movielens20M	ICM_tags	{0,...,257}	none	No	0.021511	0.021989
Movielens20M	ICM_tags	{0,...,257}	none	Yes	0.046278	0.046010
TheMoviesDataset	ICM_all	{0,...,10}	BM25	No	0.020088	0.020384
TheMoviesDataset	ICM_all	{0,...,10}	BM25	Yes	0.022559	0.022645
TheMoviesDataset	ICM_all	{0,...,10}	TF-IDF	No	0.021774	0.021757
TheMoviesDataset	ICM_all	{0,...,10}	TF-IDF	Yes	0.022649	0.022675
TheMoviesDataset	ICM_all	{0,...,10}	none	No	0.021275	0.021015
TheMoviesDataset	ICM_all	{0,...,10}	none	Yes	0.022496	0.022477
TheMoviesDataset	ICM_credits	{0,...,10}	BM25	No	0.018459	0.018736
TheMoviesDataset	ICM_credits	{0,...,10}	BM25	Yes	0.021602	0.021574

Continued on next page...

Table 3: Results obtained with our experiments. (continued)

TheMoviesDataset	ICM_credits	{0,...,10}	TF-IDF	No	0.022298	0.022349
TheMoviesDataset	ICM_credits	{0,...,10}	TF-IDF	Yes	0.021317	0.021234
TheMoviesDataset	ICM_credits	{0,...,10}	none	No	0.020302	0.020107
TheMoviesDataset	ICM_credits	{0,...,10}	none	Yes	0.021312	0.021344
TheMoviesDataset	ICM_metadata	{0,...,3}	BM25	No	0.015298	0.015122
TheMoviesDataset	ICM_metadata	{0,...,3}	BM25	Yes	0.015331	0.015146
TheMoviesDataset	ICM_metadata	{0,...,3}	TF-IDF	No	0.015450	0.015124
TheMoviesDataset	ICM_metadata	{0,...,3}	TF-IDF	Yes	0.015045	0.014869
TheMoviesDataset	ICM_metadata	{0,...,3}	none	No	0.014864	0.014744
TheMoviesDataset	ICM_metadata	{0,...,3}	none	Yes	0.014894	0.014722

Table 4: Parameters obtained with our Bayesian tuning for each of our experiments.

Dataset	ICM	ICM range	F.Weight	Bias	TopK	Shrink	Normalize	Bias value
BookCrossing	ICM_book_crossing	{0,...,7}	BM25	No	13	996	Yes	-
BookCrossing	ICM_book_crossing	{0,...,7}	BM25	Yes	17	14	No	847.931
BookCrossing	ICM_book_crossing	{0,...,7}	TF-IDF	No	13	996	Yes	-
BookCrossing	ICM_book_crossing	{0,...,7}	TF-IDF	Yes	17	14	No	847.931
BookCrossing	ICM_book_crossing	{0,...,7}	none	No	794	7	Yes	-
BookCrossing	ICM_book_crossing	{0,...,7}	none	Yes	782	80	Yes	1.80954
CiteULike_a	ICM_title_abstract	[0,1]	BM25	No	9	998	No	-
CiteULike_a	ICM_title_abstract	[0,1]	BM25	Yes	19	999	No	0.608829
CiteULike_a	ICM_title_abstract	[0,1]	TF-IDF	No	13	997	Yes	-
CiteULike_a	ICM_title_abstract	[0,1]	TF-IDF	Yes	6	40	Yes	0.0121279
CiteULike_a	ICM_title_abstract	[0,1]	none	No	12	17	Yes	-
CiteULike_a	ICM_title_abstract	[0,1]	none	Yes	27	74	Yes	3.49176
CiteULike_t	ICM_title_abstract	[0,1]	BM25	No	795	995	Yes	-
CiteULike_t	ICM_title_abstract	[0,1]	BM25	Yes	44	5	No	934.206
CiteULike_t	ICM_title_abstract	[0,1]	TF-IDF	No	352	196	Yes	-
CiteULike_t	ICM_title_abstract	[0,1]	TF-IDF	Yes	49	10	No	795.541
CiteULike_t	ICM_title_abstract	[0,1]	none	No	799	4	Yes	-
CiteULike_t	ICM_title_abstract	[0,1]	none	Yes	38	996	Yes	912.339
LastFMHetrec2011	ICM_tags	{0,...,108}	BM25	No	196	31	Yes	-
LastFMHetrec2011	ICM_tags	{0,...,108}	BM25	Yes	794	675	Yes	1.01154
LastFMHetrec2011	ICM_tags	{0,...,108}	TF-IDF	No	488	1000	Yes	-
LastFMHetrec2011	ICM_tags	{0,...,108}	TF-IDF	Yes	800	1000	Yes	0.01
LastFMHetrec2011	ICM_tags	{0,...,108}	none	No	777	990	Yes	-
LastFMHetrec2011	ICM_tags	{0,...,108}	none	Yes	794	675	Yes	1.01154
Movielens10M	ICM_all	{0,...,69}	BM25	No	800	981	Yes	-
Movielens10M	ICM_all	{0,...,69}	BM25	Yes	288	845	No	3.85444
Movielens10M	ICM_all	{0,...,69}	TF-IDF	No	794	994	Yes	-
Movielens10M	ICM_all	{0,...,69}	TF-IDF	Yes	43	23	No	954.063
Movielens10M	ICM_all	{0,...,69}	none	No	19	0	No	-
Movielens10M	ICM_all	{0,...,69}	none	Yes	41	13	No	976.515
Movielens10M	ICM_genres	{0,1}	BM25	No	8	978	Yes	-
Movielens10M	ICM_genres	{0,1}	BM25	Yes	40	968	No	787.718
Movielens10M	ICM_genres	{0,1}	TF-IDF	No	11	7	Yes	-
Movielens10M	ICM_genres	{0,1}	TF-IDF	Yes	15	1000	No	0.0149086
Movielens10M	ICM_genres	{0,1}	none	No	7	13	Yes	-
Movielens10M	ICM_genres	{0,1}	none	Yes	25	20	No	805.151

Continued on next page...

Table 4: Parameters obtained with our Bayesian tuning. (continued)

Movielens10M	ICM_tags	{0,...,69}	BM25	No	795	987	Yes	-
Movielens10M	ICM_tags	{0,...,69}	BM25	Yes	46	3	No	941.337
Movielens10M	ICM_tags	{0,...,69}	TF-IDF	No	797	987	Yes	-
Movielens10M	ICM_tags	{0,...,69}	TF-IDF	Yes	62	69	No	0.0100024
Movielens10M	ICM_tags	{0,...,69}	none	No	24	0	No	-
Movielens10M	ICM_tags	{0,...,69}	none	Yes	41	13	No	976.515
Movielens1M	ICM_genres	{0,1}	BM25	No	11	2	Yes	-
Movielens1M	ICM_genres	{0,1}	BM25	Yes	199	1	No	37.5414
Movielens1M	ICM_genres	{0,1}	TF-IDF	No	791	23	Yes	-
Movielens1M	ICM_genres	{0,1}	TF-IDF	Yes	199	1	No	37.5414
Movielens1M	ICM_genres	{0,1}	none	No	70	539	Yes	-
Movielens1M	ICM_genres	{0,1}	none	Yes	145	21	No	69.0231
Movielens20M	ICM_all	{0,...,257}	BM25	No	798	991	Yes	-
Movielens20M	ICM_all	{0,...,257}	BM25	Yes	60	24	Yes	998.289
Movielens20M	ICM_all	{0,...,257}	TF-IDF	No	800	1000	Yes	-
Movielens20M	ICM_all	{0,...,257}	TF-IDF	Yes	28	964	Yes	967.81
Movielens20M	ICM_all	{0,...,257}	none	No	795	14	No	-
Movielens20M	ICM_all	{0,...,257}	none	Yes	60	24	Yes	998.289
Movielens20M	ICM_genres	{0,1}	BM25	No	8	8	Yes	-
Movielens20M	ICM_genres	{0,1}	BM25	Yes	742	27	Yes	0.0100618
Movielens20M	ICM_genres	{0,1}	TF-IDF	No	388	708	No	-
Movielens20M	ICM_genres	{0,1}	TF-IDF	Yes	5	1000	Yes	0.01
Movielens20M	ICM_genres	{0,1}	none	No	17	11	No	-
Movielens20M	ICM_genres	{0,1}	none	Yes	37	19	No	767.533
Movielens20M	ICM_tags	{0,...,257}	BM25	No	798	991	Yes	-
Movielens20M	ICM_tags	{0,...,257}	BM25	Yes	40	992	Yes	962.089
Movielens20M	ICM_tags	{0,...,257}	TF-IDF	No	793	991	Yes	-
Movielens20M	ICM_tags	{0,...,257}	TF-IDF	Yes	28	964	Yes	967.81
Movielens20M	ICM_tags	{0,...,257}	none	No	791	3	No	-
Movielens20M	ICM_tags	{0,...,257}	none	Yes	28	990	Yes	950.21
TheMoviesDataset	ICM_all	{0,...,10}	BM25	No	5	13	Yes	-
TheMoviesDataset	ICM_all	{0,...,10}	BM25	Yes	9	38	No	980.25
TheMoviesDataset	ICM_all	{0,...,10}	TF-IDF	No	5	998	Yes	-
TheMoviesDataset	ICM_all	{0,...,10}	TF-IDF	Yes	5	1000	No	1000
TheMoviesDataset	ICM_all	{0,...,10}	none	No	5	25	Yes	-
TheMoviesDataset	ICM_all	{0,...,10}	none	Yes	5	1000	No	1000
TheMoviesDataset	ICM_credits	{0,...,10}	BM25	No	5	0	Yes	-

Continued on next page...

Table 4: Parameters obtained with our Bayesian tuning. (continued)

TheMoviesDataset	ICM_credits	{0,...,10}	BM25	Yes	8	308	No	141.506
TheMoviesDataset	ICM_credits	{0,...,10}	TF-IDF	No	5	1000	Yes	-
TheMoviesDataset	ICM_credits	{0,...,10}	TF-IDF	Yes	5	1000	No	0.01
TheMoviesDataset	ICM_credits	{0,...,10}	none	No	5	22	Yes	-
TheMoviesDataset	ICM_credits	{0,...,10}	none	Yes	5	1000	No	1000
TheMoviesDataset	ICM_metadata	{0,...,3}	BM25	No	5	994	Yes	-
TheMoviesDataset	ICM_metadata	{0,...,3}	BM25	Yes	5	989	Yes	0.0276268
TheMoviesDataset	ICM_metadata	{0,...,3}	TF-IDF	No	5	22	Yes	-
TheMoviesDataset	ICM_metadata	{0,...,3}	TF-IDF	Yes	5	926	Yes	599.775
TheMoviesDataset	ICM_metadata	{0,...,3}	none	No	5	0	Yes	-
TheMoviesDataset	ICM_metadata	{0,...,3}	none	Yes	5	0	Yes	1000

Given the large amount of data collected, it is hard to draw some conclusions from these tables. For this reason, we summarized the evidence obtained through several plots.

At first, Figure 8 displays a comparison of the performances with and without the addition of a bias term, including the MAP@10 (Mean Average Precision at 10) on both validation and test set. The plots are grouped by ICM type and feature weighting technique.

Since this graphical representation is still not comfortable to read, we introduced also Figure 9 and Figure 10, focusing on a metric easier to interpret: the increment in MAP determined by the introduction of the bias term. Since the values assumed by the MAP metric are small and confined in a very limited interval, the increment is first shown in absolute terms (Figure 9), i.e. formally:

$$\Delta MAP = MAP_{biased} - MAP_{original}$$

For further clarity however, the relative increment is also shown (Figure 10), computed as

$$\Delta MAP_{rel} = \frac{MAP_{biased} - MAP_{original}}{MAP_{original}}$$

and measured in percentage points. These last two plot are the most significant for our analysis, as they clearly summarize the impact the introduction of the bias had on the final recommendations.

The major result that can be deduced from the figures is that the introduction of a bias generates unpredictable results with binary and floating point data, but can determine significant advantages when applied to integer data. Moreover, we notice how, when the data are binary, the bias could often lead to overfitting, with even significant differences between the validation and test performances.

We are not able to conclude on the relation between feature weighting and the introduction of the bias. Indeed, in some cases the weighting procedure seems to degrade the improvements determined by the bias term, as it happens for the datasets Movielens10M and Movielens20M with integer features, while in other experiments it significantly improves the overall performances, as in LastFMHetrec2011.

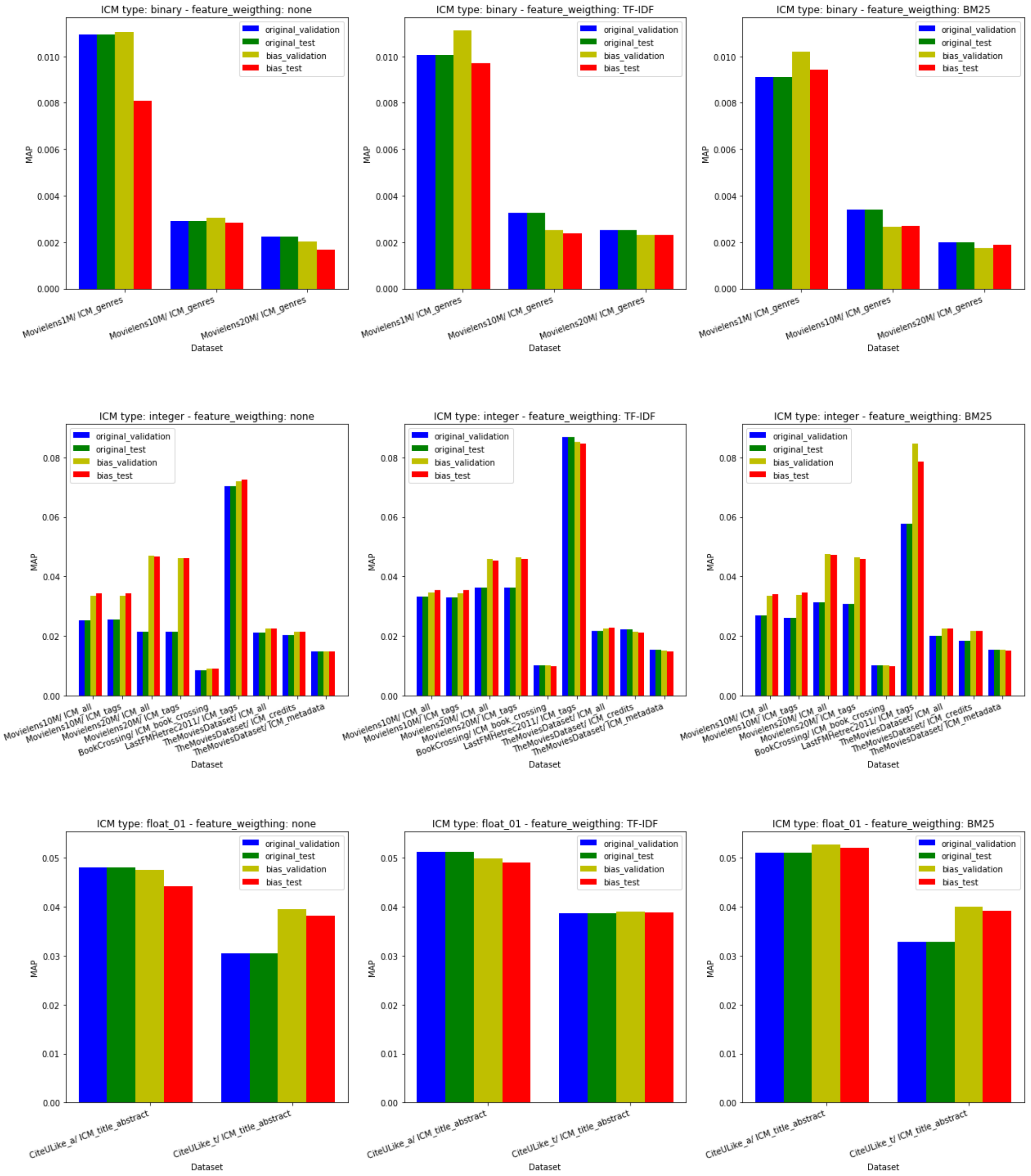

Figure 8: Comparison of the performances with and without the bias term, computed on both validation and test set.

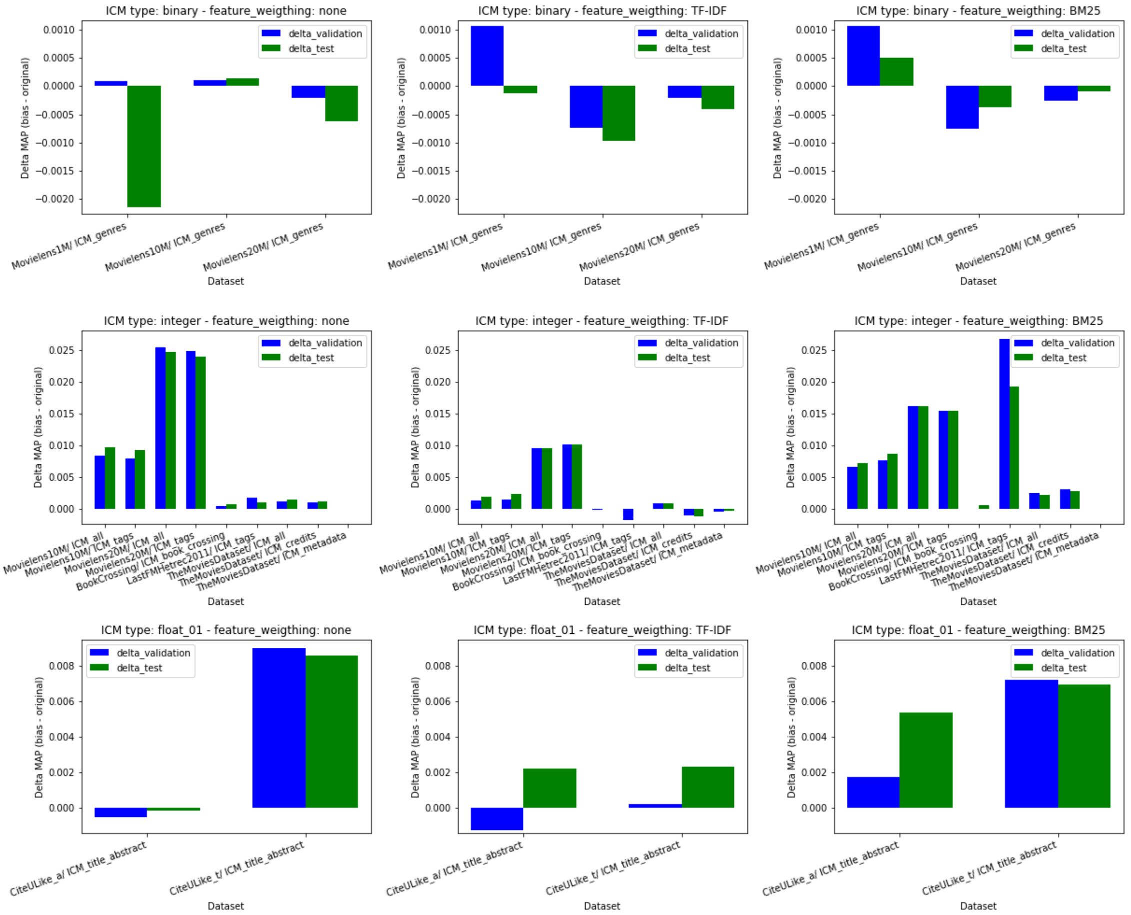

Figure 9: Comparison of the increment determined by the bias term in each dataset, on both validation and test set. The plots are grouped according to ICM type and feature weighting applied.

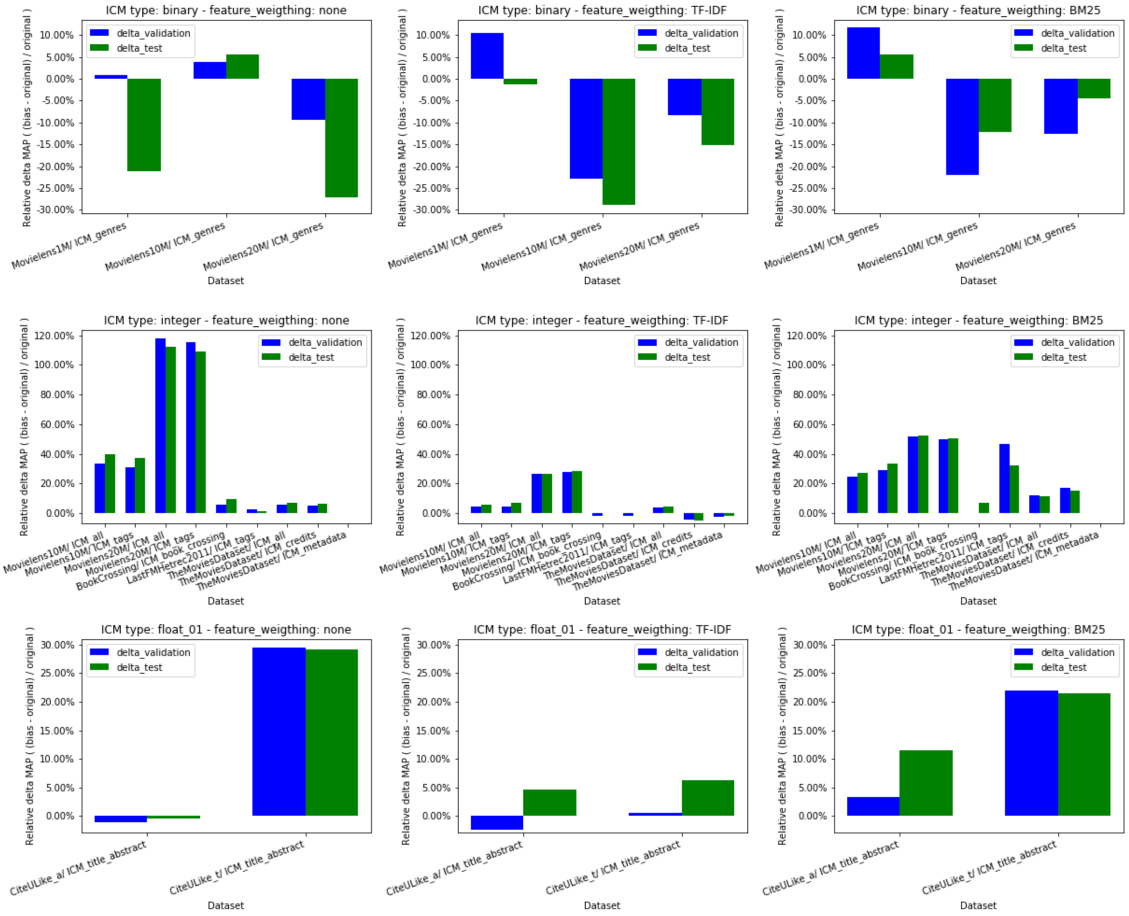

Figure 10: Comparison of the relative increment, measured in percentage points, determined by the bias term in each dataset, on both validation and test set. The plots are grouped according to ICM type and feature weighting applied.

5 Conclusions

In this work, we studied the effect a bias term can have on the performances of a content-based filtering recommender system.

To do so, we performed as a first step a theoretical analysis of its impact on the cosine similarity between two feature vectors. We conducted such analysis first analytically and then numerically, discovering that the bias term perturbs the angular distance between the vectors, significantly increasing their similarity in a non-linear manner. Moreover, we registered a distortion of the similarity measure, particularly accentuated when the lengths of the feature vectors are very dissimilar. In limit cases, the cosine similarity decouples from one of the vectors and becomes completely dependent on the longest of the two, thus disrupting its being a well-defined similarity measure.

With this first evidence, we proceeded to analyze the impact of the bias term on the overall performance of the recommender system. Despite the catastrophic risk just highlighted, we noticed how in several occasions, particularly with integer features, the introduction of the bias determines significant performance improvements. In other cases however, it results in large losses and, at times, might even lead to overfitting.

We thus conclude that no general judgment can be made about the introduction of a bias term. This technique could indeed be an effective way of increasing the performance of a content-based recommender system, especially when integer features are present. Nevertheless, we warn that such improvements are not guaranteed and underline that, because of its strong non-linearities, the behavior of this method can be particularly unpredictable, as already remarked at the end of Sections 3.3.2 and 4.

References

[1] Tuomo Korenius, Jorma Laurikkala, and Martti Juhola. "On principal component analysis, cosine and Euclidean measures in information retrieval". In: *Information Sciences* 177.22 (2007), pp. 4893–4905. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2007.05.027. URL: http://www.sciencedirect.com/science/article/pii/S0020025507002630.

[2] *ML Wiki Cosine Similarity.* URL: http://mlwiki.org/index.php/Cosine_Similarity (visited on 04/30/2019).