Science and Information Conference 2015
July 28-30, 2015 | London, United Kingdom

A New Perspective on Recommender Systems: *a Class Path Information Model*

Roberto Saia, Ludovico Boratto, Salvatore Carta
Department of Mathematics and Computer Science
University of Cagliari, Italy

OUTLINE

1. Introduction
2. Implementation
3. Experiments
4. Conclusions

RECOMMENDER SYSTEMS

A **Recommender System** suggests items that might be interesting for a user

RECOMMENDER SYSTEMS

A **Recommender System** suggests items that might be interesting for a user

Then it has to perform a **rating prediction** [1,2,3]

RECOMMENDER SYSTEMS

A **Recommender System** suggests items that might be interesting for a user

Then it has to perform a **rating prediction** [1,2,3]

However, there are some widely-known problems that afflict this operation, such as the **Overspecialization** [4], and the ***Preference Stability*** [5,6,7]

OPEN PROBLEMS

Overspecialization/Serendipity

- *Recommender systems usually suggest items that have a strong similarity with the user profile*
- *Then users never receive suggestions for unexpected, surprising, and novel items*

OPEN PROBLEMS

Overspecialization/Serendipity

- *Recommender systems usually suggest items that have a strong similarity with the user profile*
- *Then users never receive suggestions for unexpected, surprising, and novel items*

Preference Stability

- *Users tend to choose items of the same classes*
- *For example, movies of the same genres*

PROPOSED SOLUTION

1

Information Source Preprocessing

PROPOSED SOLUTION

1
Information Source Preprocessing

We build a set of binary filters able to describe the user preferences

2
User Model Definition

PROPOSED SOLUTION

1
Information Source Preprocessing

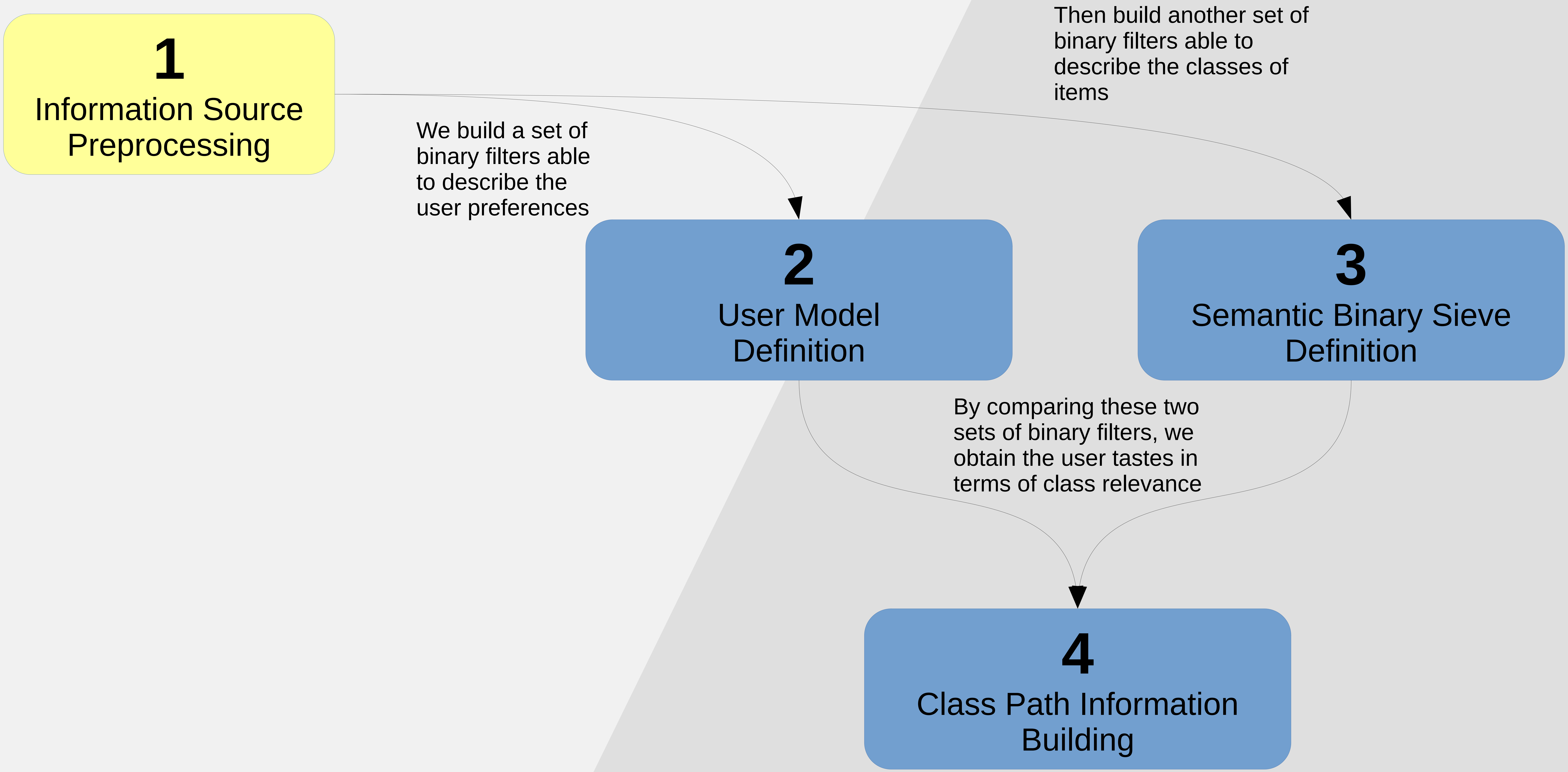
We build a set of binary filters able to describe the user preferences

2
User Model Definition

Then build another set of binary filters able to describe the classes of items

3
Semantic Binary Sieve Definition

PROPOSED SOLUTION



PROPOSED SOLUTION

1
Information Source Preprocessing

We build a set of binary filters able to describe the user preferences

Then build another set of binary filters able to describe the classes of items

2
User Model Definition

3
Semantic Binary Sieve Definition

By comparing these two sets of binary filters, we obtain the user tastes in terms of class relevance

4
Class Path Information Building

CPI
Model

The **CPI model** could be employed by a recommender system to exploit the class preference information

1. SOURCE PREPROCESSING

1 We preprocess the textual description of each item, to remove the useless elements

1. SOURCE PREPROCESSING

1. We preprocess the textual description of each item, to remove the useless elements
2. then we convert each word in its base form (e.g., *walk* instead of *walking*)

1. SOURCE PREPROCESSING

1. We preprocess the textual description of each item, to remove the useless elements
2. then we convert each word in its base form (e.g., *walk* instead of *walking*)
3. and assign the right word sense (**1**=noun, **2**=verb: we use only nouns and verbs)

In the near future, a computer hacker named Neo discovers that all life on Earth may be nothing more than an elaborate illusion

Future/**1** computer/**1** hacker/**1** nam/**2** neo/**1** discover/**2** life/**1** earth/**1** illusion/**1**

1. SOURCE PREPROCESSING

1. We preprocess the textual description of each item, to remove the useless elements
2. then we convert each word in its base form (e.g., *walk* instead of *walking*)
3. and assign the right word sense (1=noun, 2=verb: we use only nouns and verbs)

Future/**1** computer/**1** hacker/**1** nam/**2** neo/**1** discover/**2** life/**1** earth/**1** illusion/**1**

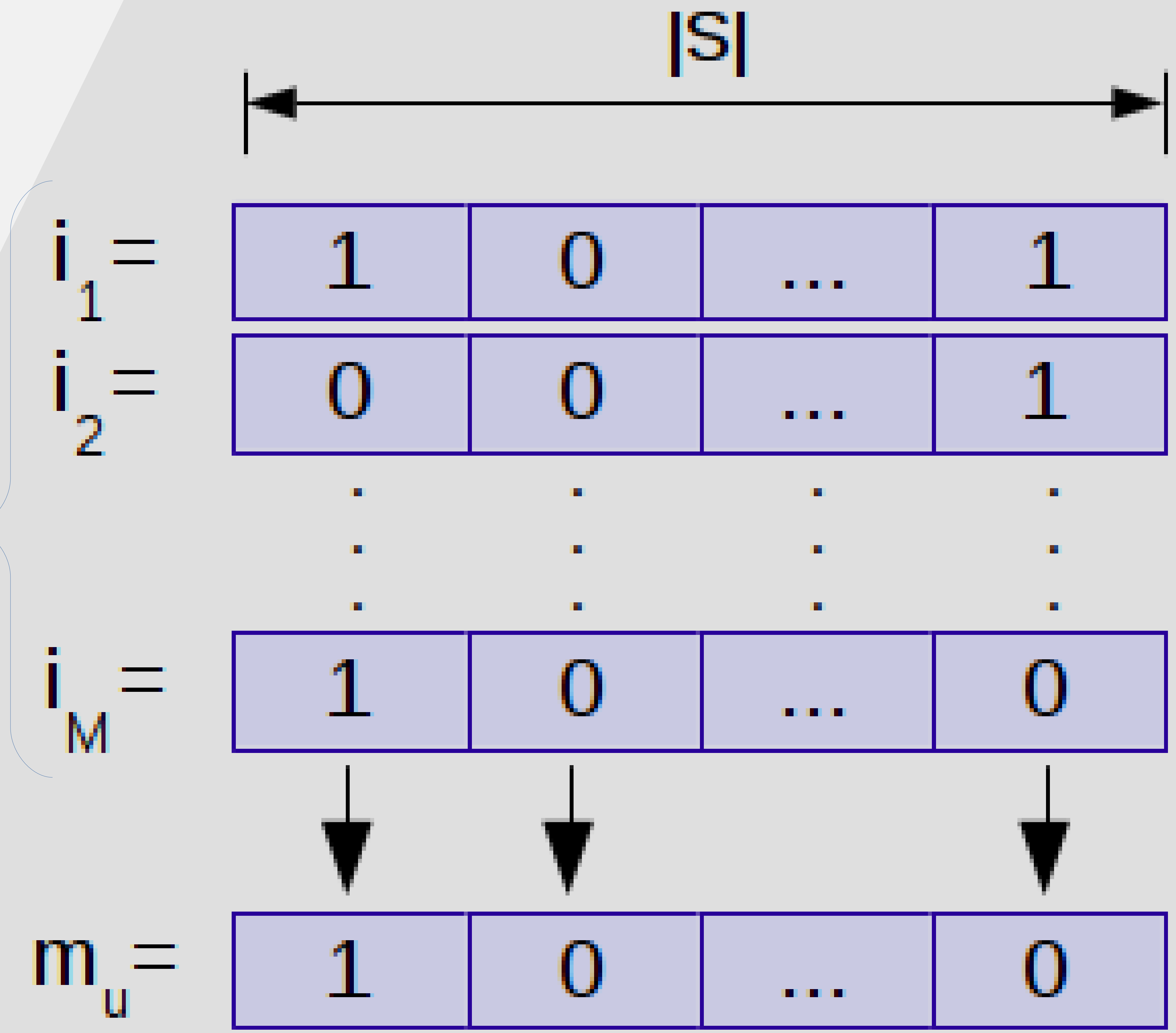
We map each of this term in a *Wordnet* **synset** (*Wordnet* is a lexical database that groups the words into sets of synonyms called *synsets*)

2. USER MODEL DEFINITION

$$m_u[w] = \begin{cases} 1, \ if \ ds_i[w] = 1 \\ m_u[w], \ otherwise \end{cases}$$

1

*For each user **u** we consider the set of items* **I_u** *she/he likes*

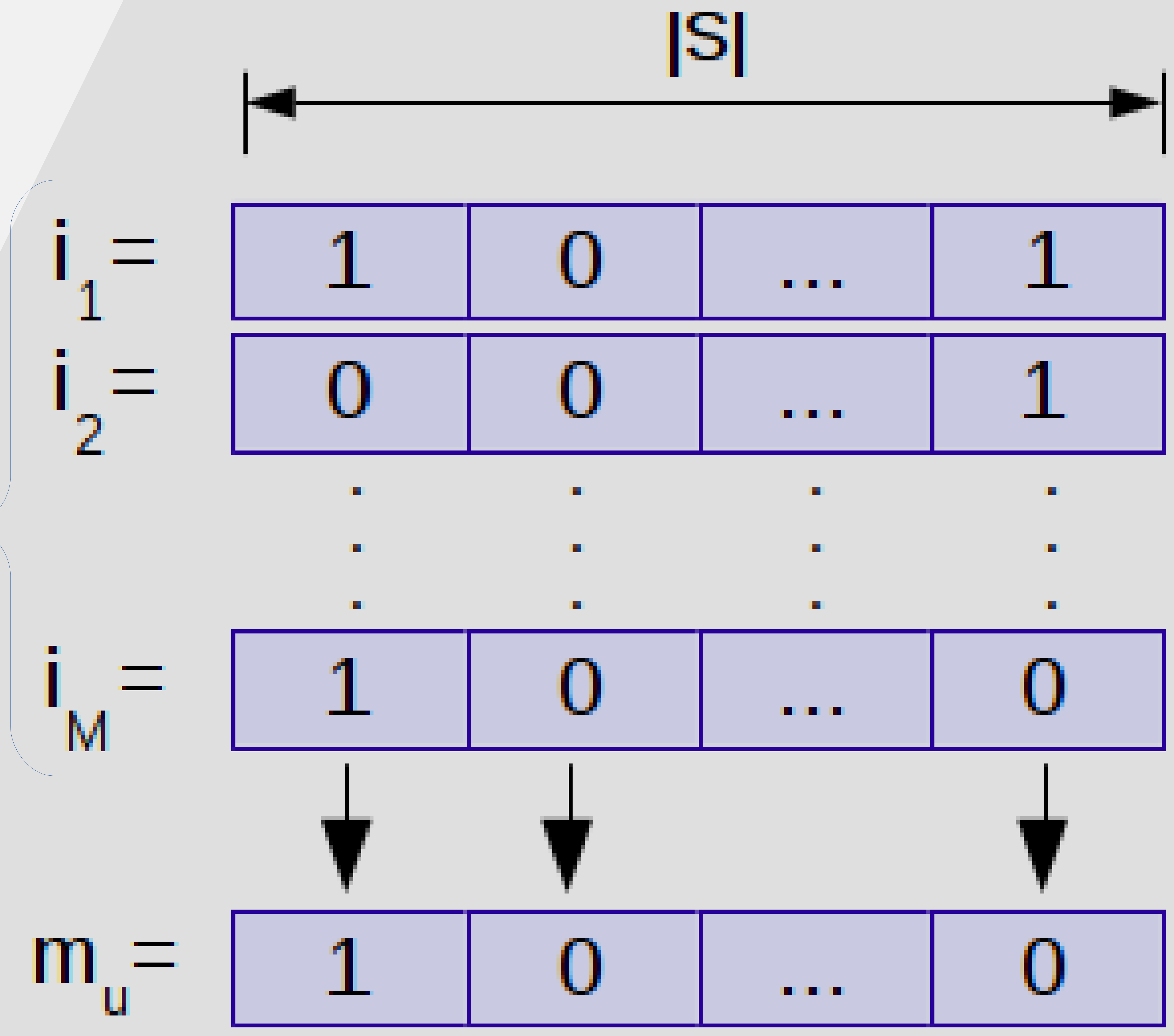


2. USER MODEL DEFINITION

$$m_u[w] = \begin{cases} 1, \ if \ ds_i[w] = 1 \\ m_u[w], \ otherwise \end{cases}$$

1. *For each user* ***u*** *we consider the set of items* $\mathbf{I_u}$ *she/he likes*

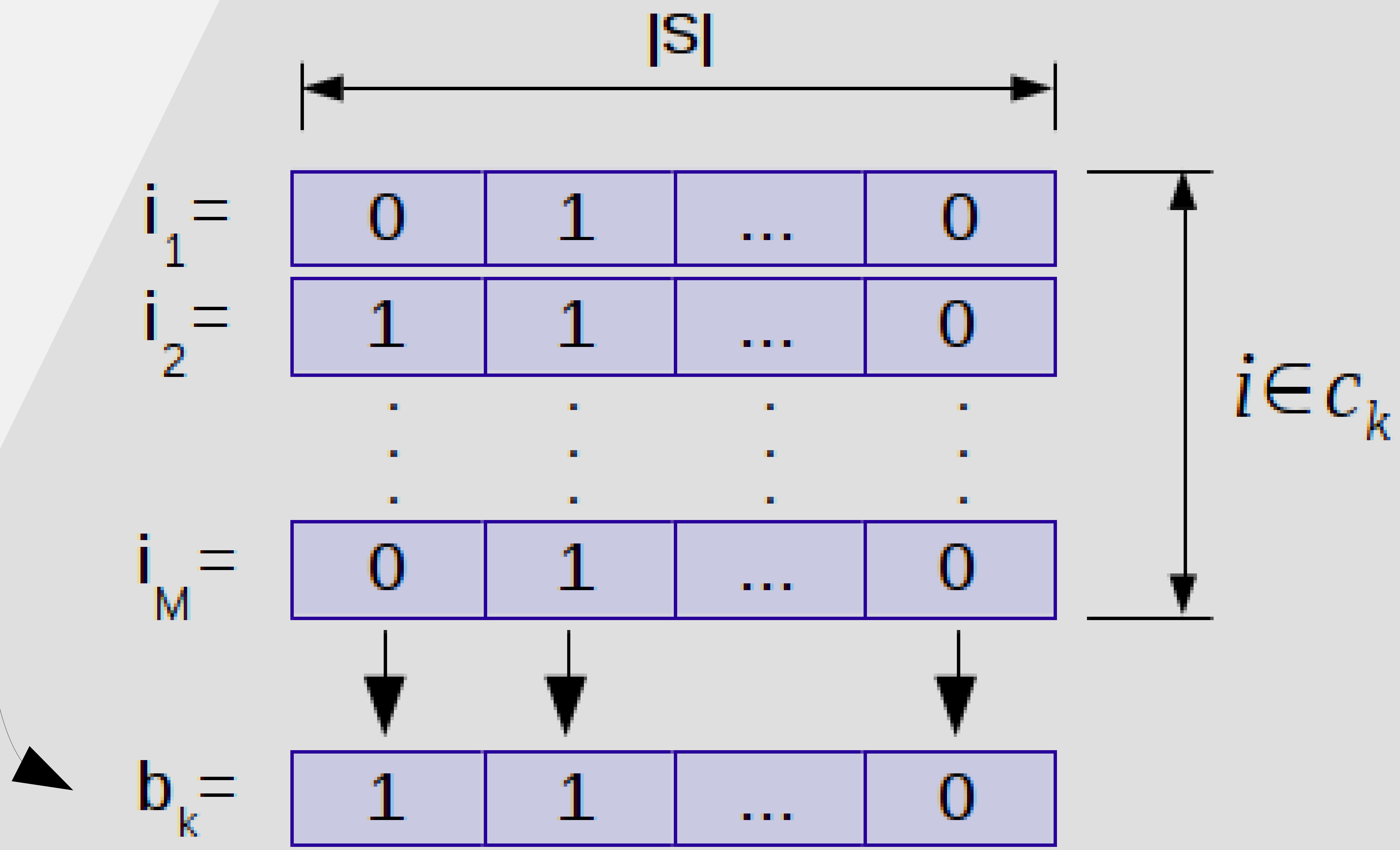
2. *Then build a model* $\mathbf{m_u}$ *that describes which synsets appear in the semantic description* $\mathbf{ds_i}$ *of each item* **i** *in her/his profile*



3. SEMANTIC BINARY SIEVE DEFINITION

$$b_k[w] = \begin{cases} 1, \ if \ ds_i[w] = 1 \land i \in c_k \\ b_k[w], \ otherwise \end{cases}$$

1. *For each class* **c ∈ C,**, *we create a binary vector* $\mathbf{b_k}$ *that reports which synsets are relevant for it*

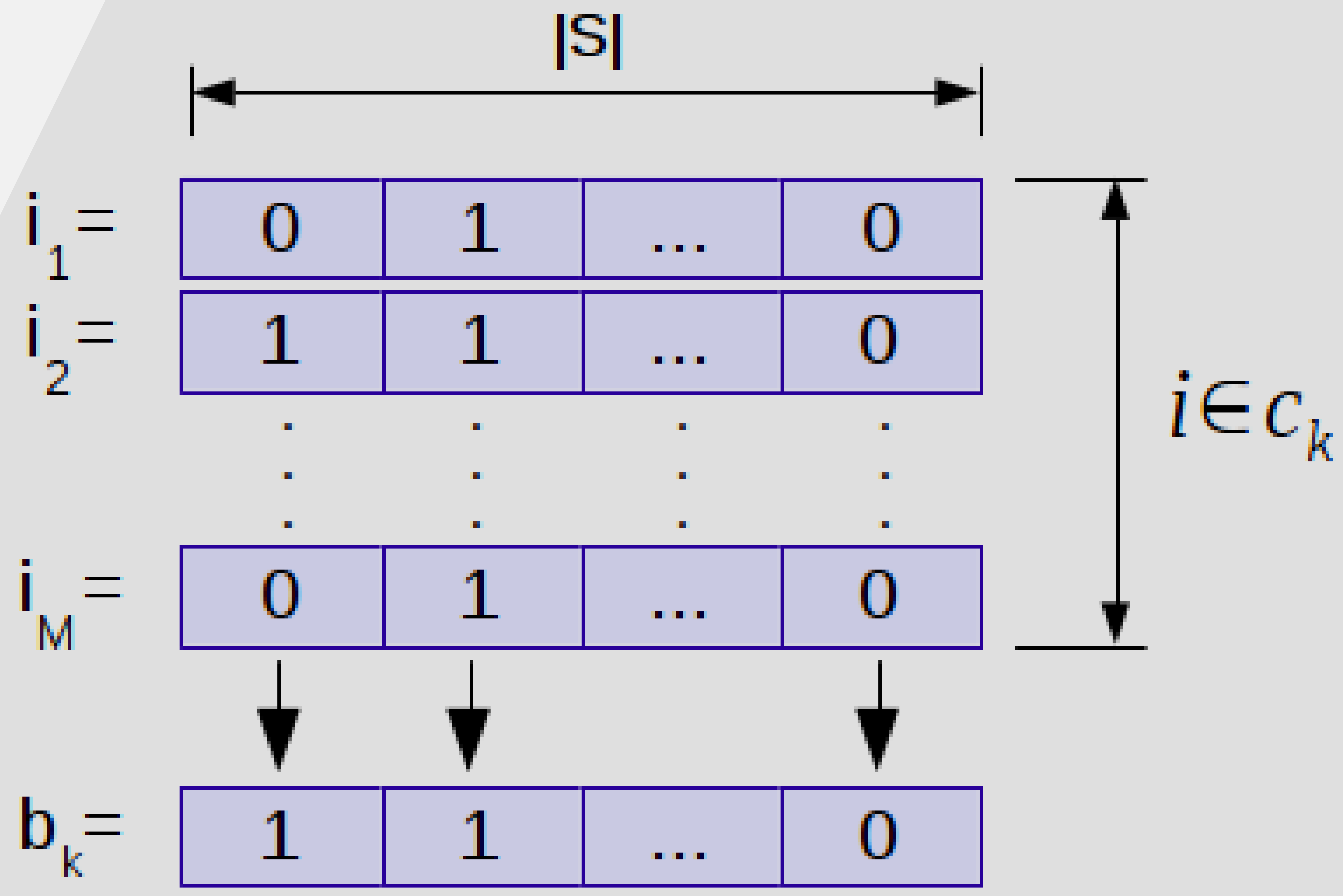


3. SEMANTIC BINARY SIEVE DEFINITION

$$b_k[w] = \begin{cases} 1, \; if \; ds_i[w] = 1 \wedge i \in c_k \\ b_k[w], \; otherwise \end{cases}$$

1. *For each class* **c ∈ C,***, we create a binary vector* **b_k** *that reports which synsets are relevant for it*

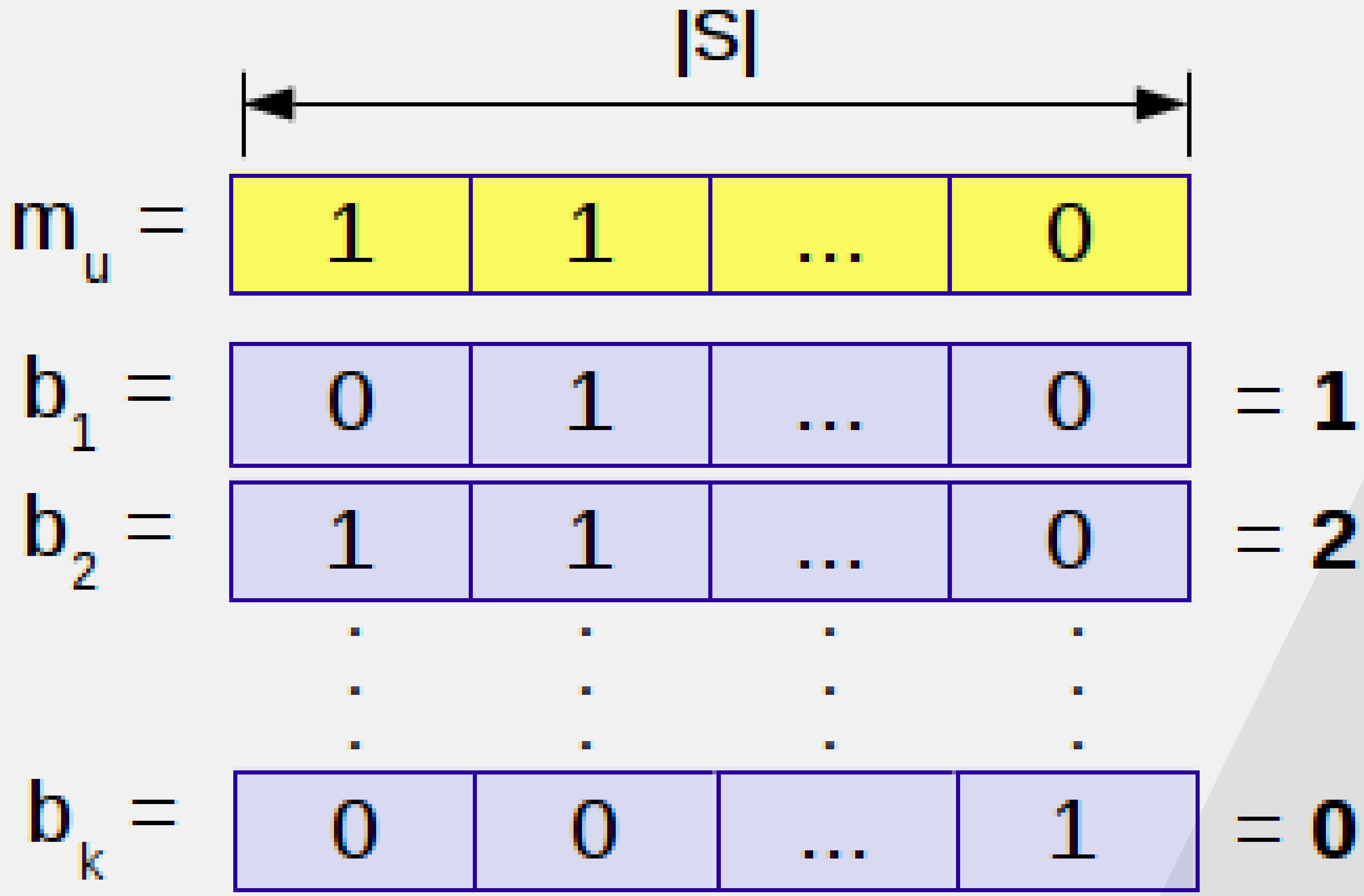
2. *Then we store each binary vector* **b_k** *in a set* **B**={**b_1, b_2, … ,b_k**} *(note that* **|B| = |C|** *)*



4. CLASS PATH VECTOR DEFINITION

$$b_k[w]\Theta m_u[w] = \begin{cases} r_u[k]++,\ if\ m_u[w]=1 \wedge b_k[w]=1 \\ r_u[k],\ otherwise \end{cases}$$

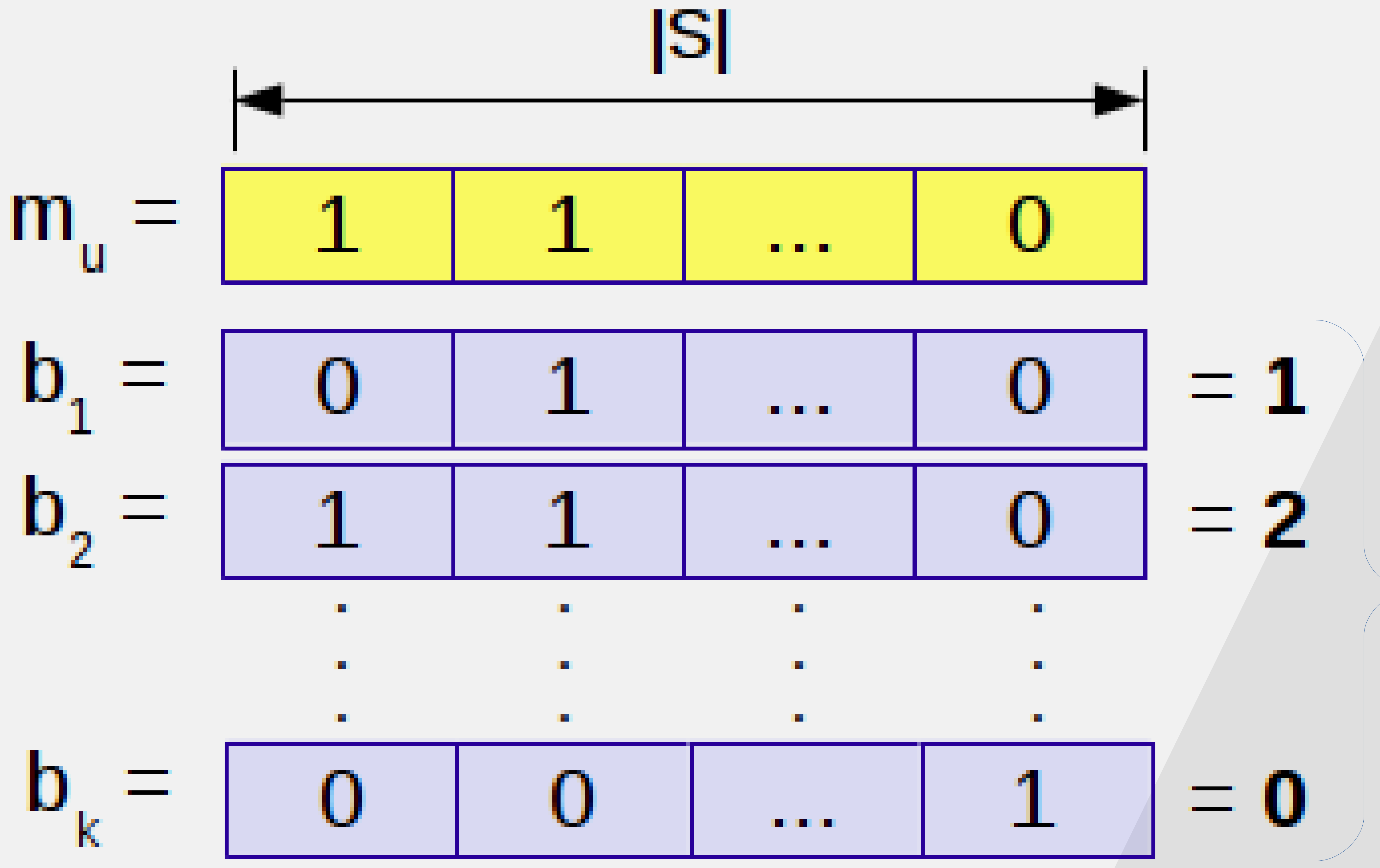
1. We compare (Θ operator) the user models **M**={$\mathbf{m_1}$, $\mathbf{m_2}$, … , $\mathbf{m_N}$} and the SBS vectors **B**={$\mathbf{b_1}$, $\mathbf{b_2}$, … , $\mathbf{b_K}$}



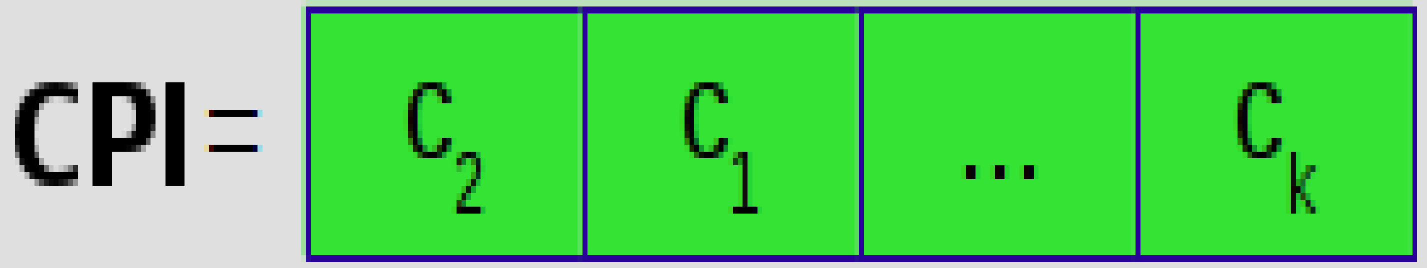
4. CLASS PATH INFORMATION DEFINITION

$$b_k[w] \Theta m_u[w] = \begin{cases} r_u[k]++, \; if \; m_u[w] = 1 \land b_k[w] = 1 \\ r_u[k], \; otherwise \end{cases}$$

1. We compare (Θ operator) the user models **M**={$\mathbf{m_1}$, $\mathbf{m_2}$, … , $\mathbf{m_N}$} and the SBS vectors **B**={$\mathbf{b_1}$, $\mathbf{b_2}$, … , $\mathbf{b_K}$}



2. Then build a Class Path Information (**CPI**) for each user, which contains the class relevance in decreasing order



EXPERIMENTS

The experiments were performed using two real-world datasets: **Yahoo! Webscope Movie (R4)** and **Movielens 10M**

EXPERIMENTS

The experiments were performed using two real-world datasets: **Yahoo! Webscope Movie (R4)** and **Movielens 10M**

Considering the absence of the textual description of the items in the **Movielens 10M** dataset, we use both datasets to study the *Preference Stability* phenomenon, but only the **Yahoo! Webscope Movie (R4)** to test our strategy

EXPERIMENTS

We compare our strategy with **SVD++** [8], the *Koren's* version of **SVD** [9], one of the most accurate approaches at the state of the art

EXPERIMENTS

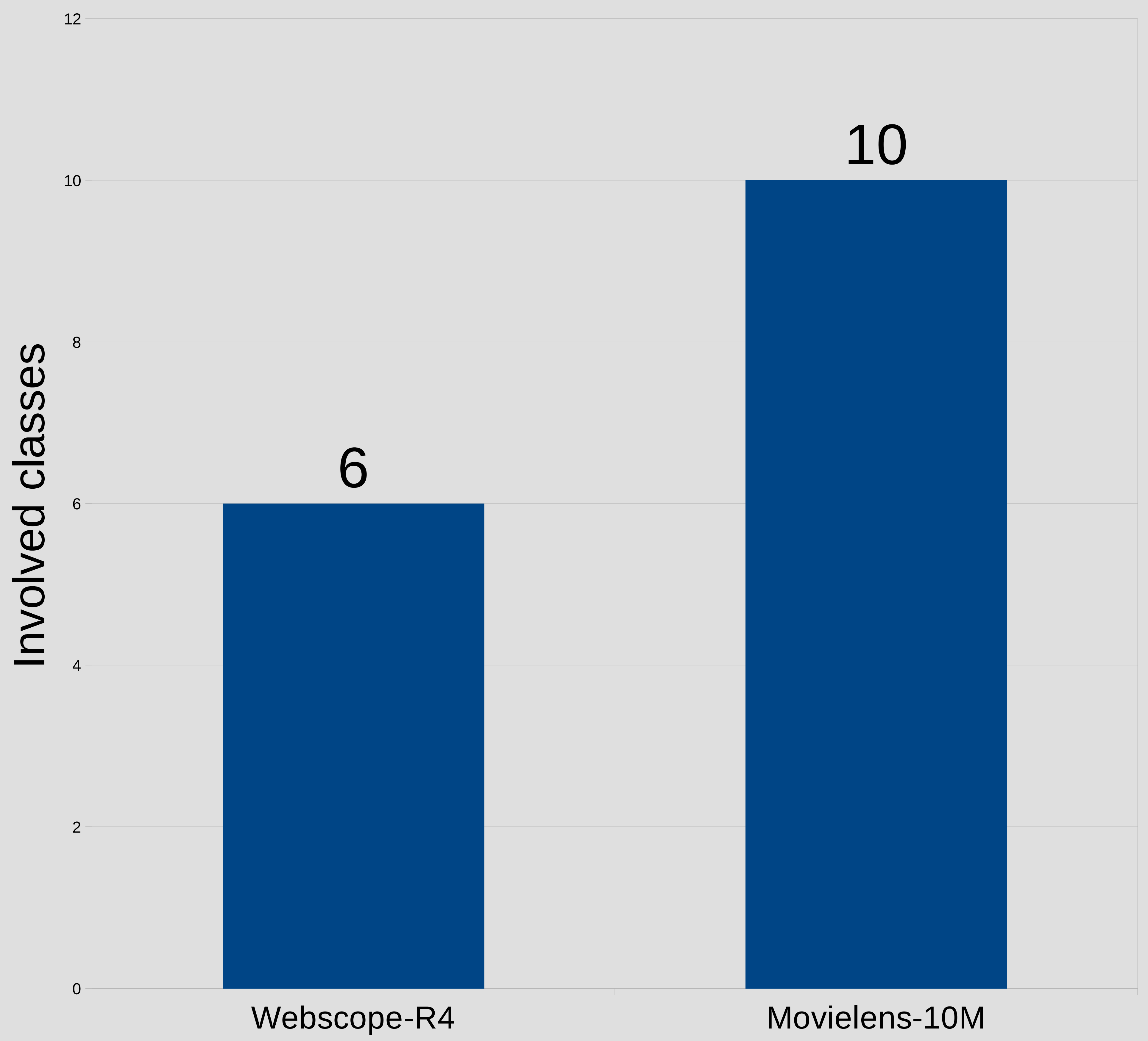
We compare our strategy with **SVD++** [8], the *Koren's* version of **SVD** [9], one of the most accurate approaches at the state of the art

We evaluate the results in order to check if:

- Our approach is able to reduce the effects of the **Preference Stability**
- The classes in the adopted **CPI Model** are significant

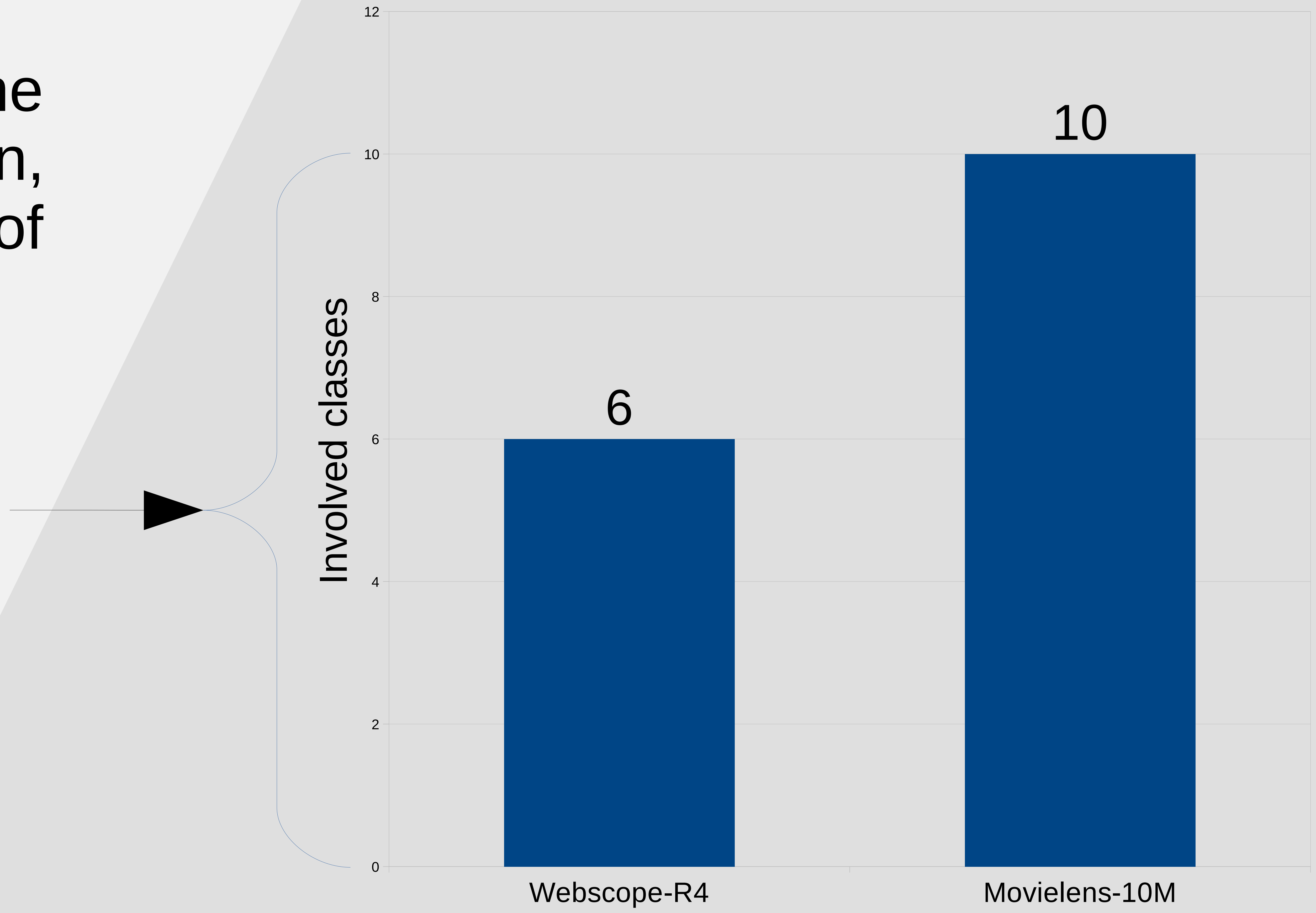
1. PRELIMINARY STUDY

- This study, aimed to investigate on the **Preference Stability** phenomenon, showed its existence in terms of classes

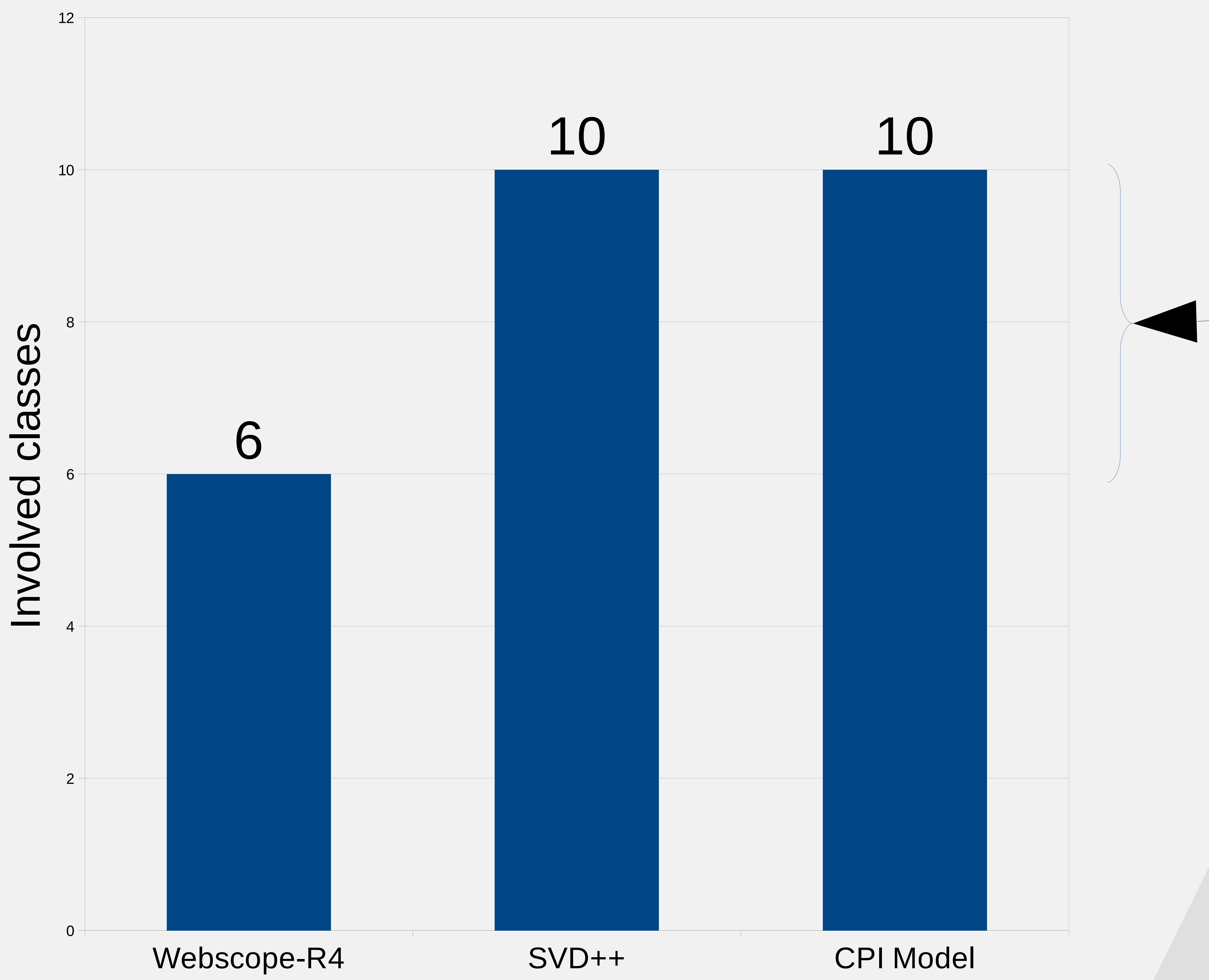


1. PRELIMINARY STUDY

- This study, aimed to investigate on the **Preference Stability** phenomenon, showed its existence in terms of classes
- The results show that the user preferences are distributed between **30** and **50** percent of the classes (*6 out of 20, and 10 out of 18*)

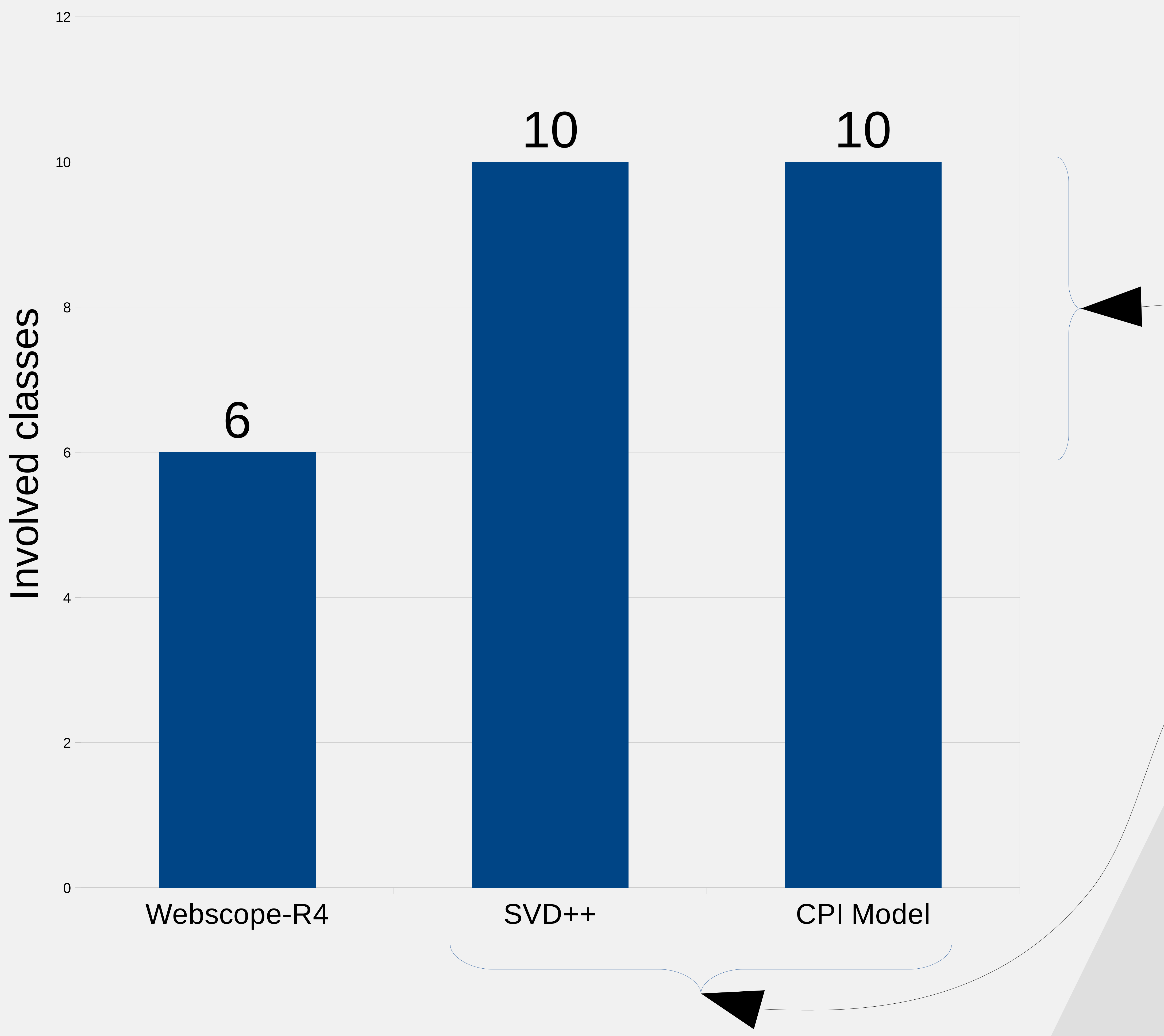


2. PREFERENCE STABILITY



The second set of experiments shows that **our strategy** is able to **extend** the ground truth, characterizing user preferences over a **larger** set of classes

2. PREFERENCE STABILITY



The second set of experiments shows that **our strategy** is able to **extend** the ground truth, characterizing user preferences over a **larger** set of classes

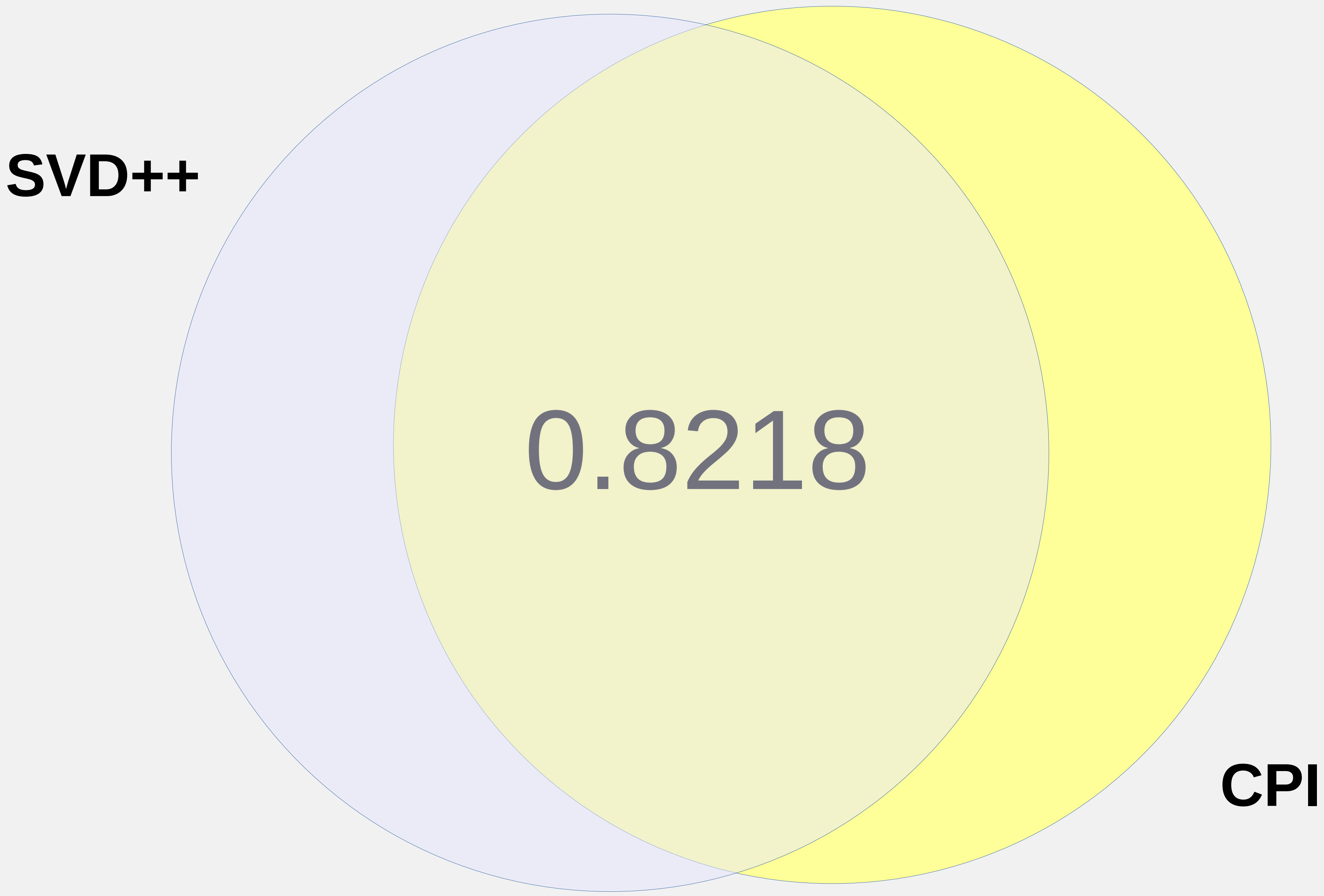
In addition, it characterizes user preferences in terms of classes, **without** producing any **comparison** with the preferences of the other users (differently from **SVD++**)

The advantages are both on the **computational load**, and on the possibility to recommend items that are **not too dissimilar** to those positively evaluated by the users

3. CLASSES IN THE CPI MODEL

1. The previous set of experiments shows that the **number of classes** involved in our **CPI** models is the same of **SVD++**

3. CLASSES IN THE CPI MODEL



$$J(C_u(CPI), C_u(SDV++)) = \frac{|C_u(CPI) \cap C_u(SDV++)|}{|C_u(CPI) \cup C_u(SDV++)|}$$

1. The previous set of experiments shows that the **number of classes** involved in our **CPI** models is the same of **SVD++**

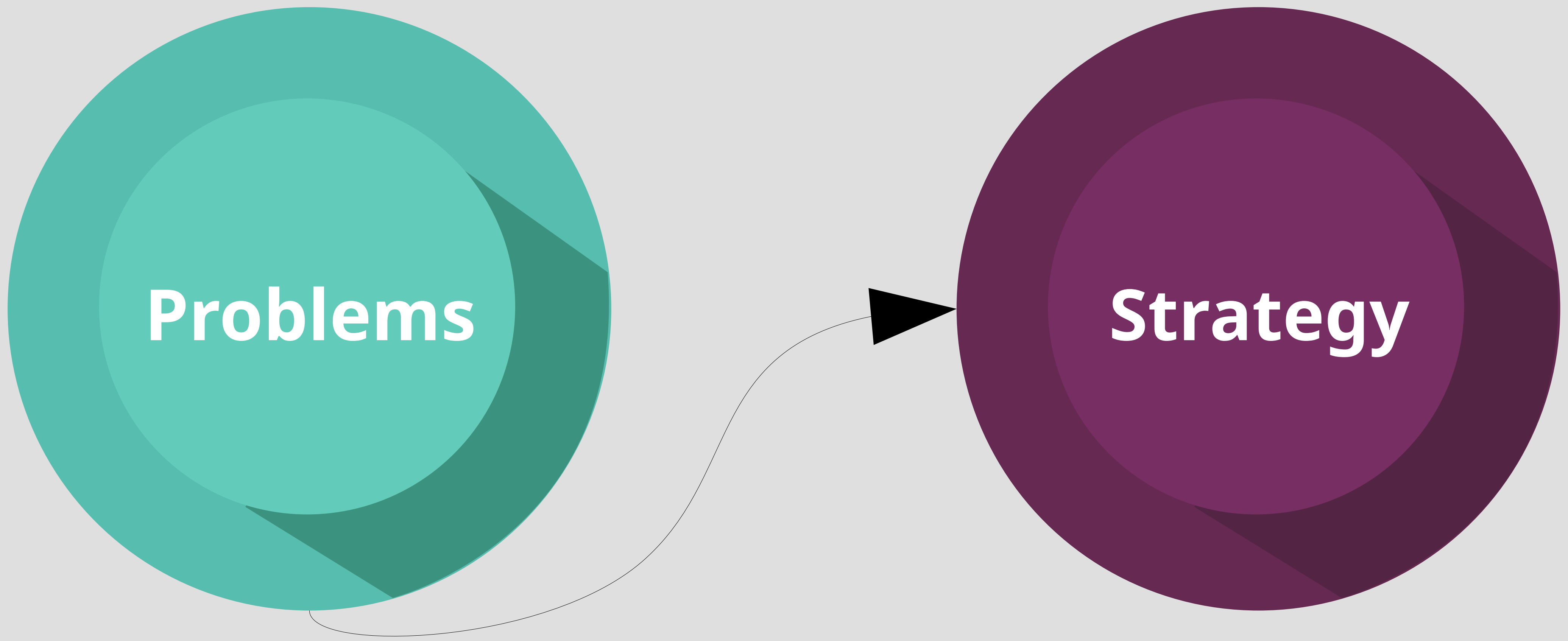
2. Using the **Jaccard index**, we verified that the **most** popular classes included in our **CPI** model, are **almost** the **same** of the **SVD++** approach

CONCLUSIONS

Problems

We wanted to face two problems: Overspecialization and Preference Stability

CONCLUSIONS



We wanted to face two problems: Overspecialization and Preference Stability

We represented the user tastes and the classes of items through two binary filters

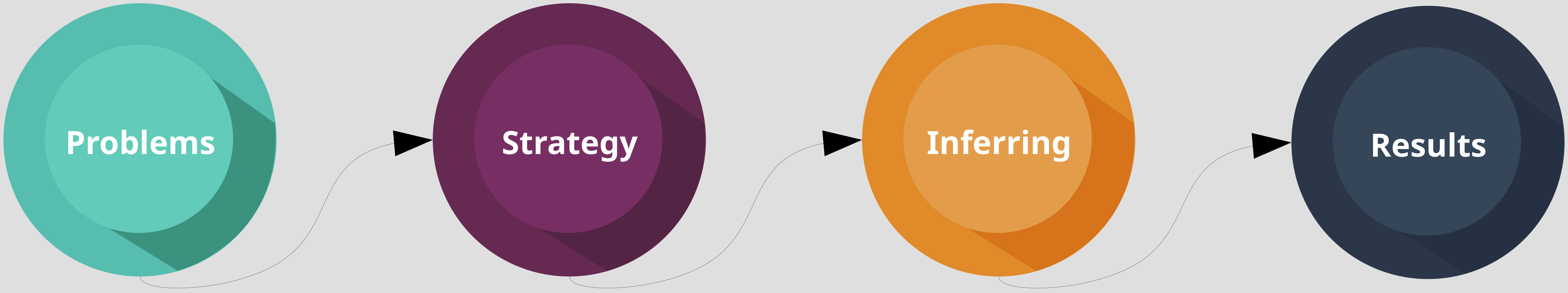
CONCLUSIONS

We wanted to face two problems: Overspecialization and Preference Stability

We represented the user tastes and the classes of items through two binary filters

We combined these two binary filters to define the CPI, a model able to infer the user preferences

CONCLUSIONS



We wanted to face two problems: Overspecialization and Preference Stability

We represented the user tastes and the classes of items through two binary filters

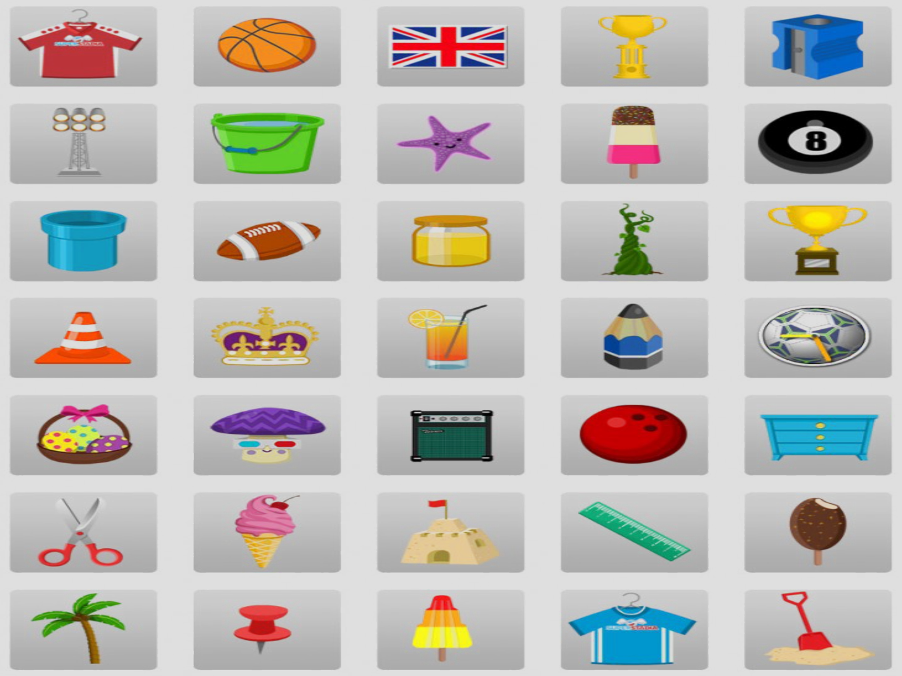
We combined these two binary filters to define the CPI, a model able to infer the user preferences

Experimental results proved the effectiveness of our strategy, comparing with an approach at the state of the art

FUTURE WORK

We will introduce our modeling technique to support a recommender system

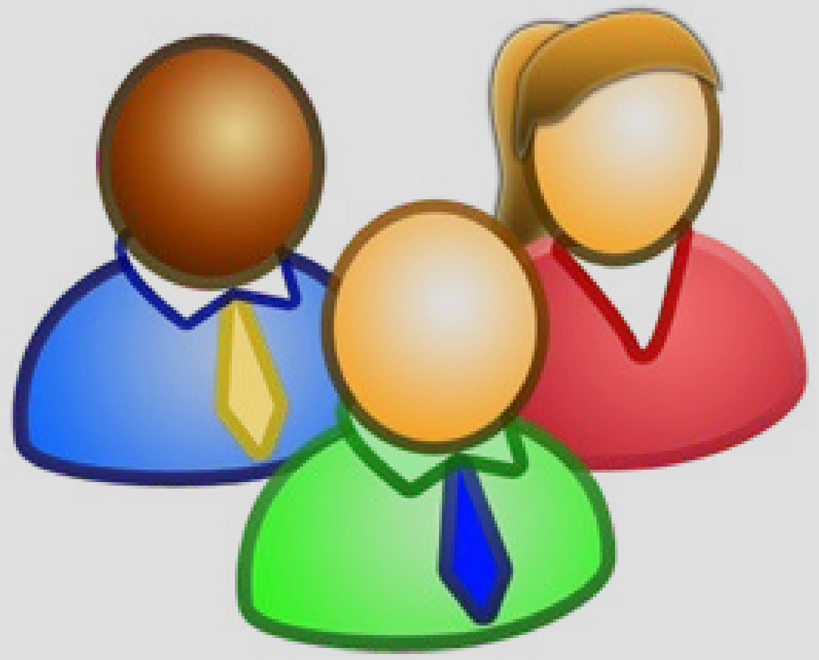
FUTURE WORK



We will introduce our modeling technique to support a recommender system

The idea is to produce the recommendations considering the available classes in the model

FUTURE WORK

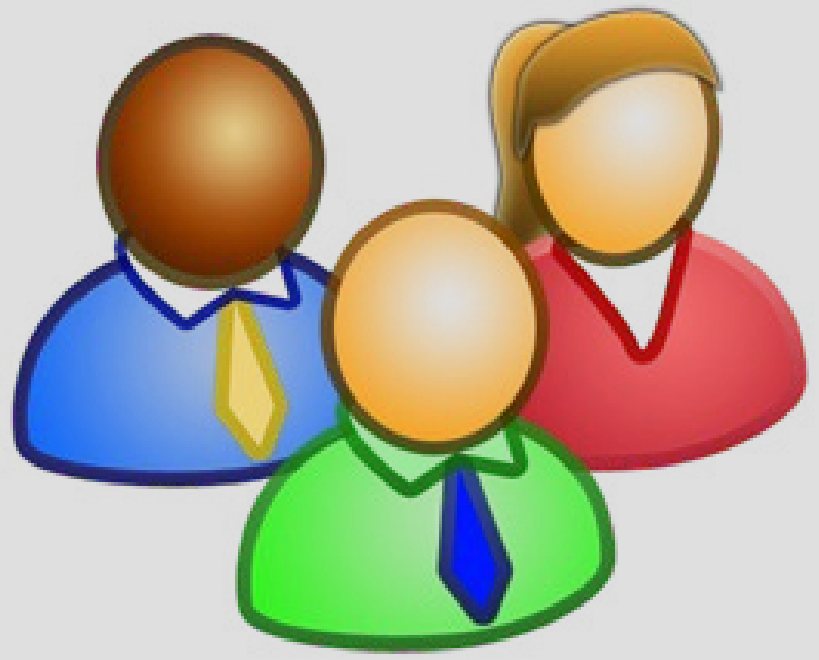


We will introduce our modeling technique to support a recommender system

The idea is to produce the recommendations considering the available classes in the model

Considering that the classes are semantically related to the items a user likes, but are not the same

FUTURE WORK



We will introduce our modeling technique to support a recommender system

The idea is to produce the recommendations considering the available classes in the model

Considering that the classes are semantically related to the items a user likes, but are not the same

This would allow to produce serendipity, without the consequences caused by diversity

REFERENCES

1. F. Ricci, L. Rokach, and B. Shapira, “Introduction to recommender systems handbook,” in Recommender Systems Handbook, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer, 2011, pp.1–35.
2. G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions,” IEEE Trans. Knowl. Data Eng., vol. 17, no. 6, 2005, pp.734–749.
3. L. Boratto and S. Carta, “The rating prediction task in a group recommender system that automatically detects groups: Architectures, algorithms, and performance evaluation,” J Intell Inf Syst, vol. To Appear, 2014.
4. G. Shani and A. Gunawardana, “Evaluating recommendation systems," in Recommender Systems Handbook, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer, 2011, pp. 257–297.
5. R. D. Burke and M. Ramezani, “Matching recommendation technologies and domains,” in Recommender Systems Handbook, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer, 2011, pp. 367–386.
6. S. A. Munson and P. Resnick, “Presenting diverse political opinions: How and how much,” in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ser. CHI ’10. New York, NY, USA: ACM, 2010, pp. 1457–1466.
7. E. Pariser, The Filter Bubble: What the Internet Is Hiding from You. Penguin Group , The, 2011.
8. C. V. Yehuda Koren, Robert M. Bell, “Matrix factorization techniques for recommender systems,” IEEE Computer, vol. 42, no. 8, 2009, pp.30–37.
9. M. J. P. Daniel Billsus, “Learning collaborative information filters,”in Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998), Madison, Wisconsin, USA, J. W. Shavlik, Ed. Morgan Kaufmann, 1998, pp. 46–54.