Applying model checking in recommender system

- ➢ The main steps of the recommendation system and the problems solved
- ➢ Scenarios of application
- ➢ overall architecture of the recommender system
- ➢ perspective of Applying model checking
- ➢ the Automata generation
- ➢ the Model Verification
- ➢ case study
- ➢ conclusion

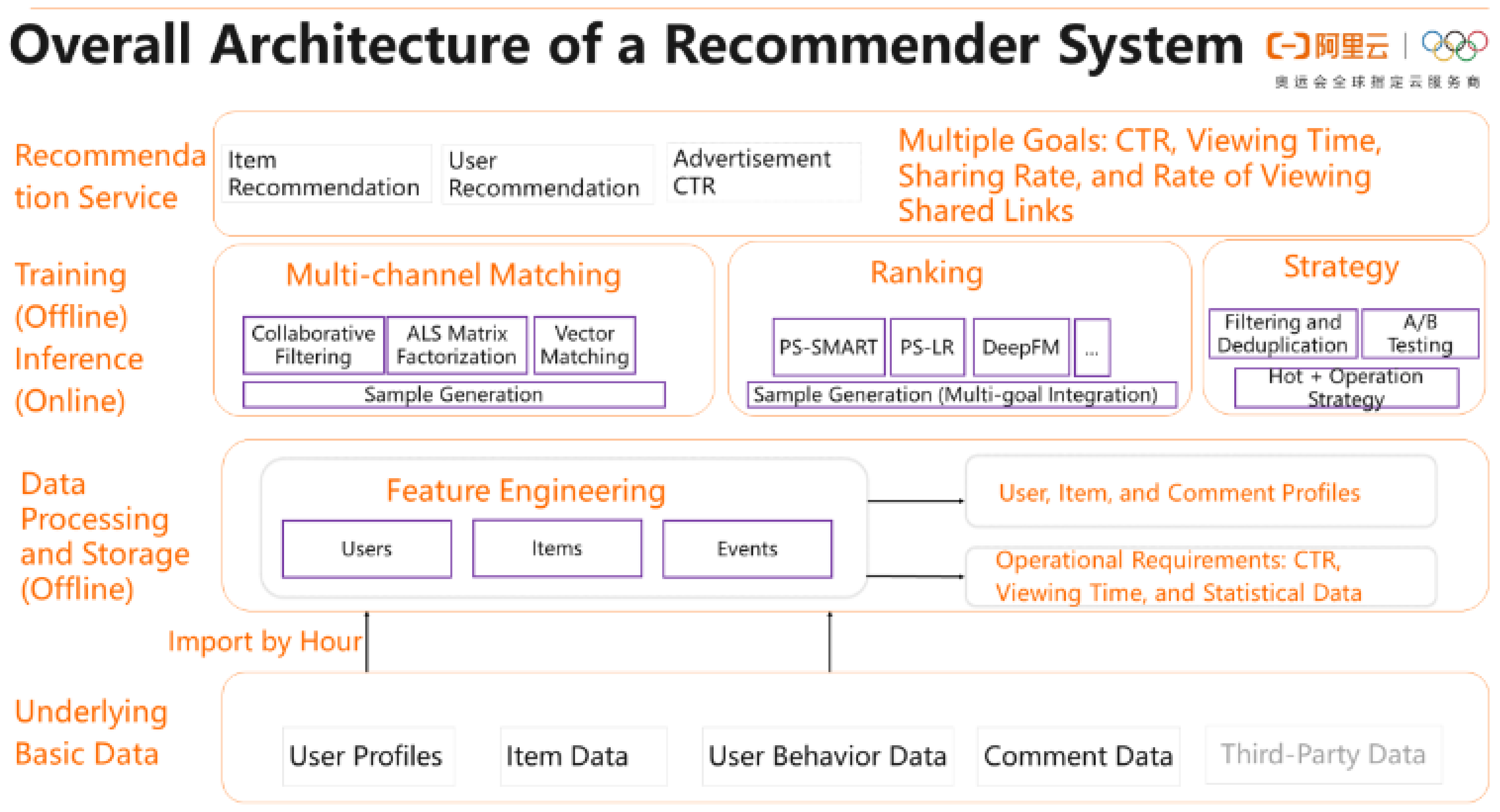
The main steps of the recommender system and the problems solved

- RS are software tools and related technologies that suggest projects t hat are considered relevant to specific users, often in scenarios or app lications where the project space is very large and project search and selection is difficult for the user.
- In general, RSs follow a process that encompasses three main steps:
  - 1. Collect related user information.
  - 2. Learning from the collected information to build user profiles;
  - 3. Applying a function or a previously built model to select and rank the items that the user is most likely to prefer.

Scenarios of application

- Recommendations based on user dimensions: recommendations base d on users' historical behaviors and interests, such as Guess Your Favo rite on the home page of shopping websites, and home page recomm endations on video and music websites.
- Recommendations based on item dimensions: Recommendations are made based on the target objects currently browsed by the user. For example, when opening the product details page of a shopping APP, p roducts related to the main product will be recommended to you.

The overall architecture of the recommender system



- **Data sources**: the various data sources on which the recommender algorithm relies, including item data, user data, behavior logs, other available business data, and even data from third parties.
- **Computing platform**: responsible for cleaning and processing of various data at the bottom, offline and real-time computing.
- **Data storage layer**: storage of data processed by the computing platform, which can be landed in different storage systems as needed.
- **Recall layer**: including various recommendation strategies or algorithms, such as classic collaborative filtering, content-based recall, etc.
- **Fusion filtering layer**: triggering multi-way recall, because each recall source of the recall layer will return a candidate set, so this layer needs to be fused and filtered.
- **Sorting layer**: Use machine learning or deep learning models, and richer features for reordering to filter out a smaller, more accurate set of recommendations to return to the upper layer of business.

The perspective of Applying model checking

- From the data storage layer to the recall layer, to the fusion filtering la
yer and the sorting layer, the candidate set decreases layer by layer, b
ut the accuracy requirement is getting higher and higher, so it also bri
ngs the increase of computational complexity layer by layer, which is t
he biggest challenge of the recommender system.
- Data cleansing and processing is often required between the data sou
rce and the data storage layer. Data plays a very important role for the
whole recommender system. Apply model checking in the data cleans
ing phase to better make the data work for the entire architecture.

- Simply put a labeled transformation system is used to represent data user behavior and model checking techniques are used to demonstrate whether data anomalies can be successfully detected.
- This is a data-driven approach designed to extract data changes directly from the database by observing modifications as they occur. This approach focuses on recurring processes that occur periodically and modify the same set of data fields each time they are executed. My goal is to capture these recurring features of data field changes by mining the changes that occur on the database using a formal approach. The data lifecycle of each user is modeled by automata. Data anomalies are detected through model checking technology.
- A "data anomaly" can occur when a change in a data field goes against the corresponding process. In fact, data usually evolves according to a specific process. Anomalies, i.e. illegal or unexpected changes to data fields, can be determined by several possible causes：
    - 1. Altering the database.
    - 2. cyber attacks on systems belonging to the database.
    - 3. failure of the operating system of the infrastructure.
    - 4. Errors in the application that updates the database

Two main phases

The Automata generation

Construct an automaton for each user.

In the figure, as an example, a set of four users (User A, User B, User C and User D) is considered. Each user is able to perform read, update and write operations on the data. Each user operation is stored through a log: a se t of information is collected from the log, enabling the proposed approach to construct an automaton for each user. In the considered example (automaton A, automaton B, automaton C and automaton D, representing the u ser behavior, have been generated. In addition, a cluster automaton has been constructed, responsible for ensuri ng synchronization between other automata.

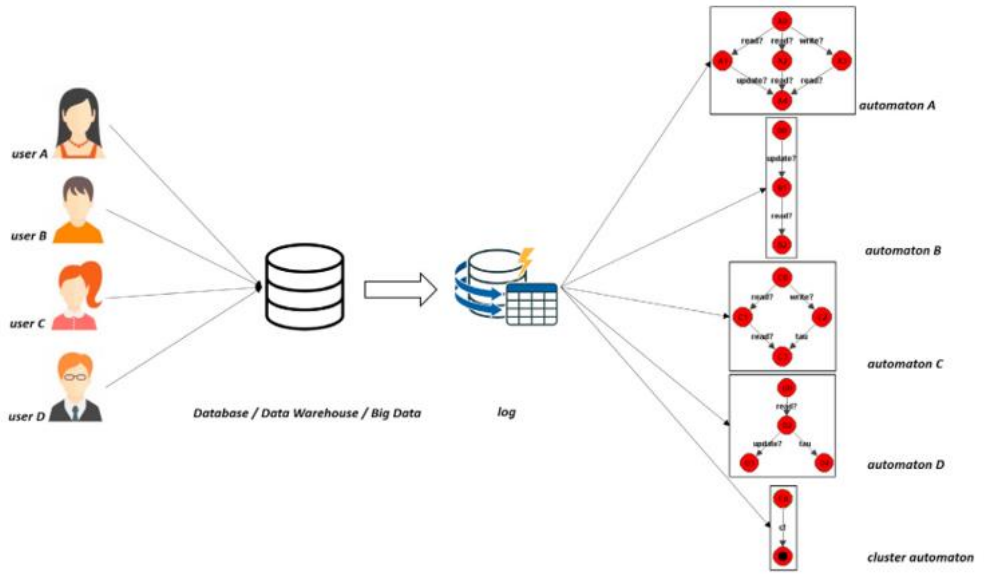


Fig. 1: Automata generation.

The Model Verification

In this phase, a model is constructed that invokes all automata in parallel, containing a restricted set of actions to ensure synchronization.

Therefore, the model checker verifies that the constructed model satisfies a set of properties: whether the model checker outputs true the model conforms to the legal behavior, otherwise (i.e., the model checker outputs false) t he model in the property analysis is not verified on the model, which shows that a departure from the legal beha vior occurred.

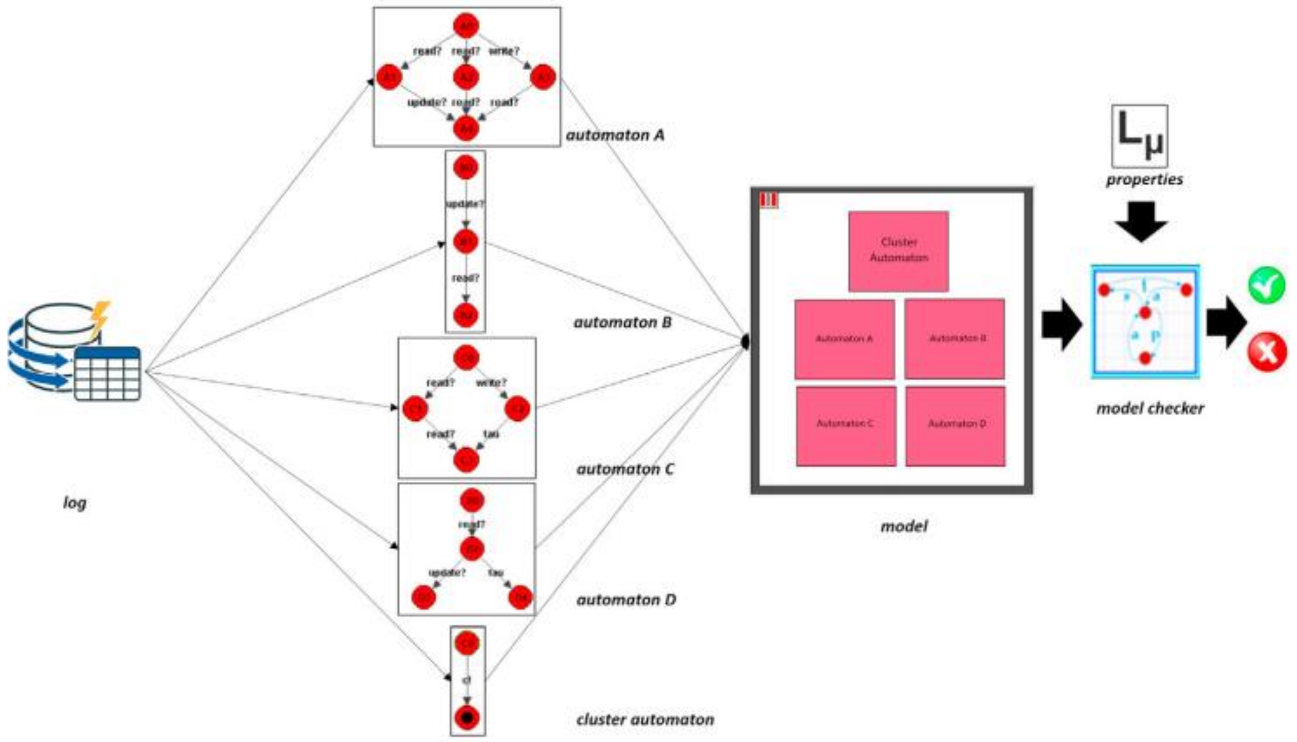
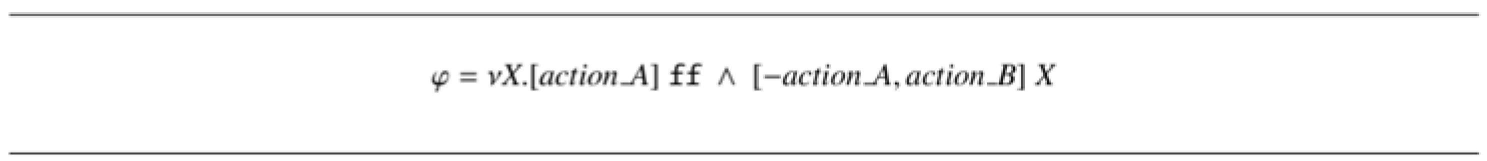


Fig. 2: Model Verification.

Temporal logic formula for detecting data anomalies

Table 2: Temporal logic formula to detect data anomalies.



$$\varphi = \nu X.[action\_A]\ \mathtt{ff}\ \wedge\ [-action\_A, action\_B]\ X$$

The ϕ property is describing following behavior: “it is not possible to perform the action A whether action B is not previously done”. This property will be considered in the case study to detect the data anomalies.

case study

- the left side of the figure shows the process system, while the right one shows the automata. To guarantee the synchronization between the automata the action we consider as restricted the actions on the cluster automata
- As shown in Figure 3 the system is composed by the parallel of A0 (i.e., the Anna user), M0 (i.e., the Mary user), F0 (i.e., the Frank user) and C0 (i.e., the cluster automaton) processes with the restriction on the c A, c M, c F actions representing the clusters for Anna, Mary and Frank plus an additional set restricted action aimed to guarantee that the various automata are properly synchronized.

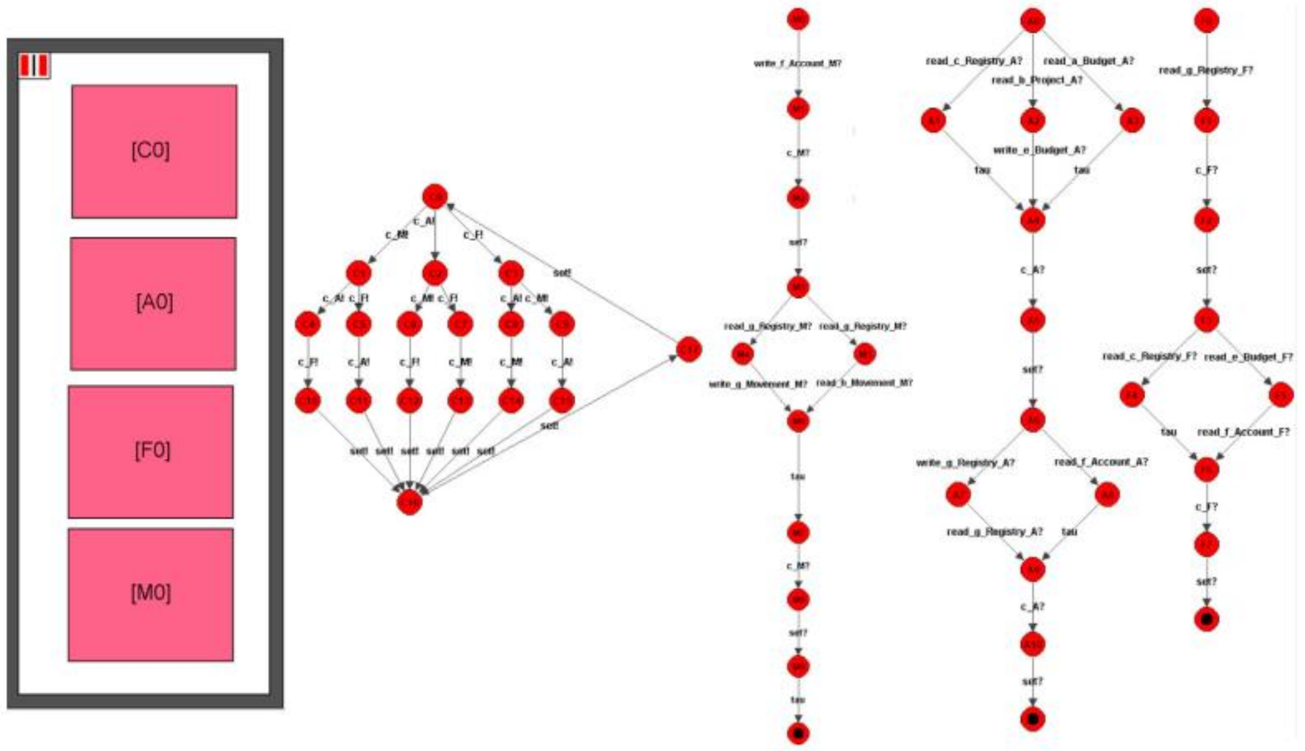
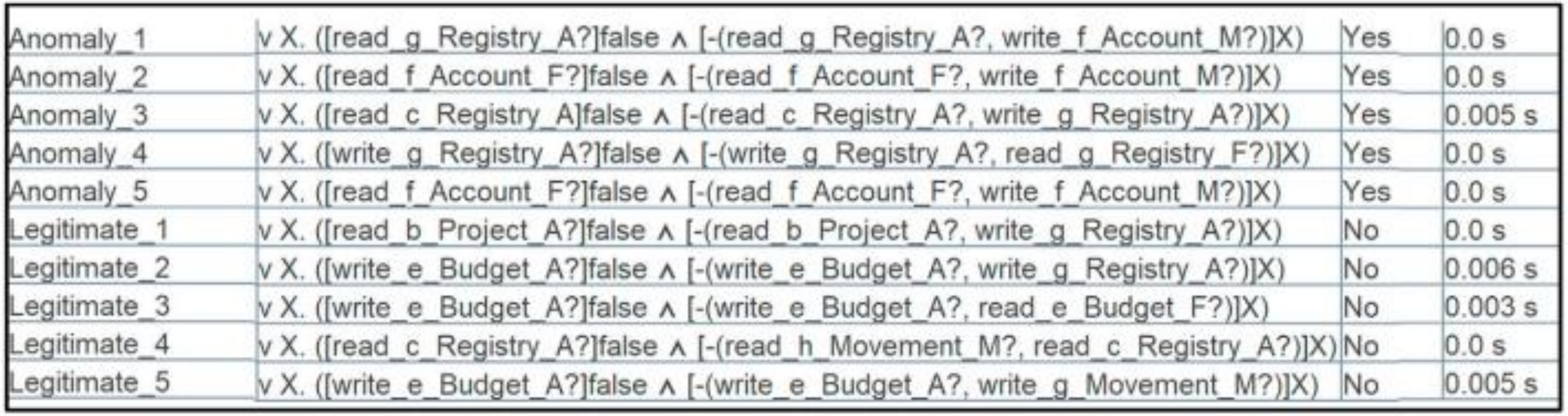


Fig. 3: Process system and automata case study.



Anomaly_1	ν X. ([read_g_Registry_A?]false ∧ [-(read_g_Registry_A?, write_f_Account_M?)]X)	Yes	0.0 s
Anomaly_2	ν X. ([read_f_Account_F?]false ∧ [-(read_f_Account_F?, write_f_Account_M?)]X)	Yes	0.0 s
Anomaly_3	ν X. ([read_c_Registry_A]false ∧ [-(read_c_Registry_A?, write_g_Registry_A?)]X)	Yes	0.005 s
Anomaly_4	ν X. ([write_g_Registry_A?]false ∧ [-(write_g_Registry_A?, read_g_Registry_F?)]X)	Yes	0.0 s
Anomaly_5	ν X. ([read_f_Account_F?]false ∧ [-(read_f_Account_F?, write_f_Account_M?)]X)	Yes	0.0 s
Legitimate_1	ν X. ([read_b_Project_A?]false ∧ [-(read_b_Project_A?, write_g_Registry_A?)]X)	No	0.0 s
Legitimate_2	ν X. ([write_e_Budget_A?]false ∧ [-(write_e_Budget_A?, write_g_Registry_A?)]X)	No	0.006 s
Legitimate_3	ν X. ([write_e_Budget_A?]false ∧ [-(write_e_Budget_A?, read_e_Budget_F?)]X)	No	0.003 s
Legitimate_4	ν X. ([read_c_Registry_A?]false ∧ [-(read_h_Movement_M?, read_c_Registry_A?)]X)	No	0.0 s
Legitimate_5	ν X. ([write_e_Budget_A?]false ∧ [-(write_e_Budget_A?, write_g_Movement_M?)]X)	No	0.005 s

Fig. 4: Data anomaly detection temporal logic properties.

anomaly 1: it is not possible that Mary performs a write operation on the f field on the “Account” table whether Anna did not read the b field of the “Registry” table previously: this represents an anomaly, in fact from the model in Figure 3 Mary performs a write operation on the f field on the “Account” table before the read operation of Anna on the b field of the “Registry” table for this reason the model checking outputs true;

legitimate 1: it is not possible that Anna performs a read operation on the b field on the “Project” table whether the same user did not previously performed a write operation on the g field of the “Registry” table. This represents a legitimate behavior, the model checker outputs false;

conclusion

- This model checking method for mining data anomalies. We use automata to represent user behavior and use model checking technology to detect whether deviations from expected behavior have occurred. In the recommender system, we can use Model Checking as a means of data cleaning in the computing platform (responsible for cleaning and processing the various data at the bottom, offline computing and real-time computing) to check to ensure that we get the data source we want so that we can better complete the subsequent steps of storage, filtering, and sorting and be better to recommend users.

references

- Madalina G. Ciobanu, Fausto Fasano, Fabio Martinelli, Francesco Mercaldo, Antonella Santone, Model Checking for Data Anomaly Detection, Procedia Computer Science,Volume 159,2019,Pages 1277-1286,ISSN 1877-0509, https://doi.org/10.1016/j.procs.2019.09.297
- Almonte, L., Guerra, E., Cantador , I. *et al.* Recommender systems in model-driven engineering. *Softw Syst Model* **21,** 249–280 (2022). https://doi.org/10.1007/s10270-021-00905-x
- Basic Concepts and Architecture of a Recommender System, https://www.alibabacloud.com/blog/basic-concepts-and-architecture-of-a-recommender-system_596642