Steam® inspired recommandation system

CHIMIENTI Francesco
PROIETTO Mattia

TELECOM Paris

IP PARIS

Working method

→ **01**

Analysis of Steam® market platform: understanding peculiarities and product placement stategies.

→ **02**

Engineering and implementing a lifelike environment.

→ **03**

Experimenting and comparing different agents.

Environment

01

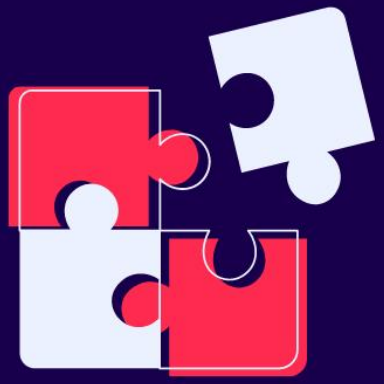
Users don't always rate a game:
this is modeled by a Bernoulli distribution
with `p = rating_probability`.

02

Agents need rewards to understand which action to take: a proper measure must be designed taking into account implicit ratings.

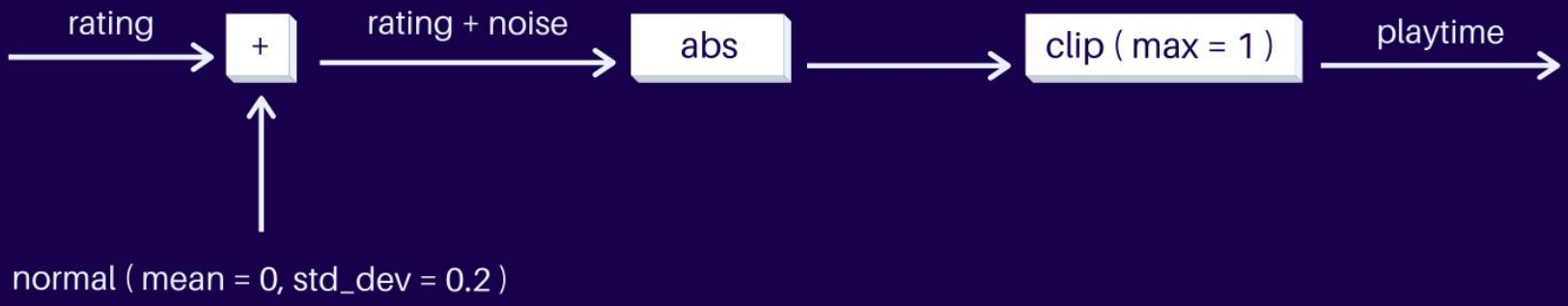
03

Users may ask for a refund if they play a game for less than 10 hours: we would like to avoid this snag in order to maximize the incomes.



Playtimes

Generally, it can be assumed that playtimes roughly reflect
how much a game is enjoyable for a given user.
Therefore, for simulation purposes, it is reasonable
to generate them starting from the rating somehow,
as it represents the similarity between user's profile and game's features.



The reward

Suppose that both rewards and playtimes $\in [0, 1] \subseteq \mathbb{R}$.
To simplify, let's say that every game lasts 100 hours.

Agents

→ **01**

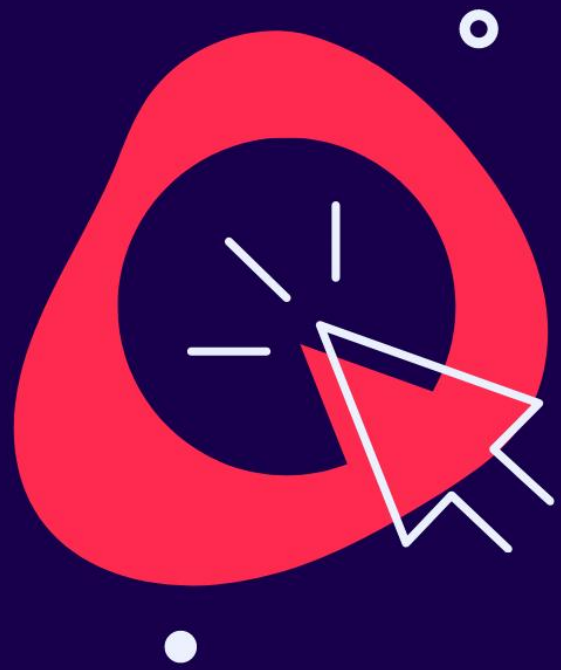
Implementing non-contextual strategies following a greedy behavior.

→ **02**

Training models to learn user-specific preferences and features.

→ **03**

Sketching out a collaborative filtering solution to infer users' interests.



→ **01**

ε-greedy
It is based on a ε parameter given a-priori.

→ **02**

Thompson sampling
The sample is made on a given Beta distribution.

→ **03**

UCB
It build several confidence intevals and chooses the best one.

Not contextual

These agents make predictions without taking into account users and their very specific preferences.
Indeed they are not aware of the existence of any context and try to abstract from the individuals inherently.
They suggest the most rated/played games of the moment.

Considering users

→ **01**

Regression

It incorporates the semantic of the similarity, therefore it can be used as a reference.

→ **02**

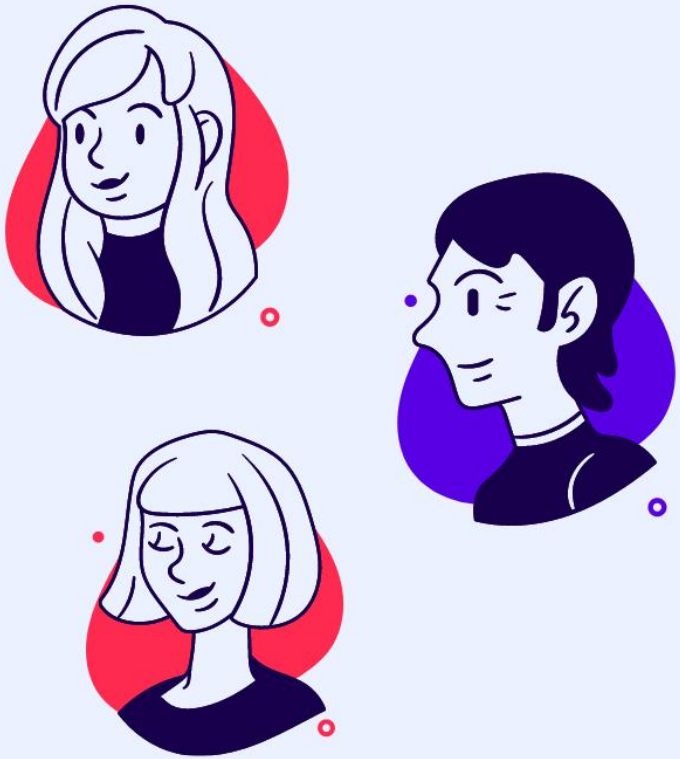
Deep regression

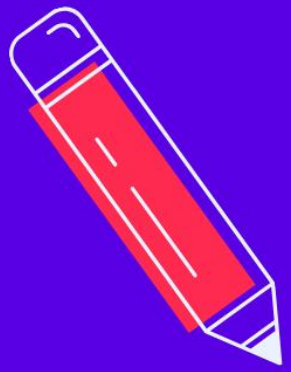
2 fully-connected layers, taking an additional feature vector as input.

→ **03**

SVD++

Only the matrix factorization is relevant for the purposes of recommender systems.





History-based prediction

An experiment is run by using a random agent to generate some data to feed the models.

generating history → **training the model** → **predicting the game**

SVD++

Each item can be represented by a vector *qi*. Similarly each user can be represented by a vector *pu* such that the dot product of those 2 vectors is the expected rating.
In order to generalize well, a penalty term is introduced.
To reduce the error between the predicted and actual value, the algorithm make use of some characteristics of the dataset as biases.
The *yj* terms are a set of item factors that capture implicit ratings.

$$\sum_{r_{ui} \in R_{train}} (r_{ui} - \hat{r}_{ui})^2 + \lambda \left(b_i^2 + b_u^2 + ||q_i||^2 + ||p_u||^2 \right)$$

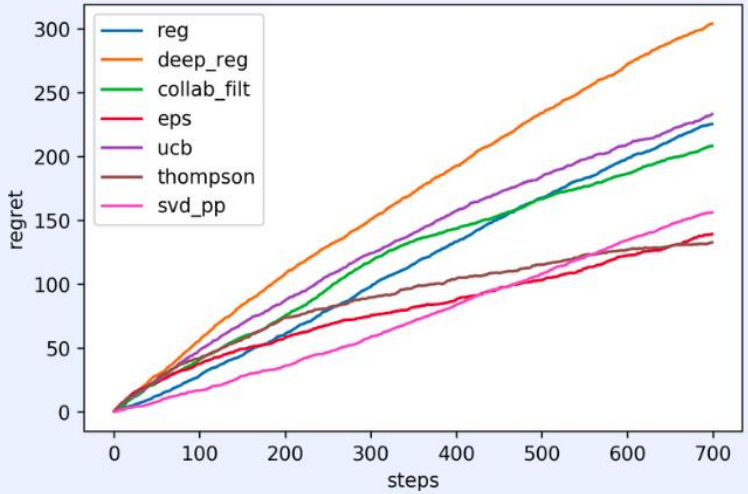
$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T \left(p_u + |I_u|^{-\frac{1}{2}} \sum_{j \in I_u} y_j \right)$$

User-based collaborative filtering

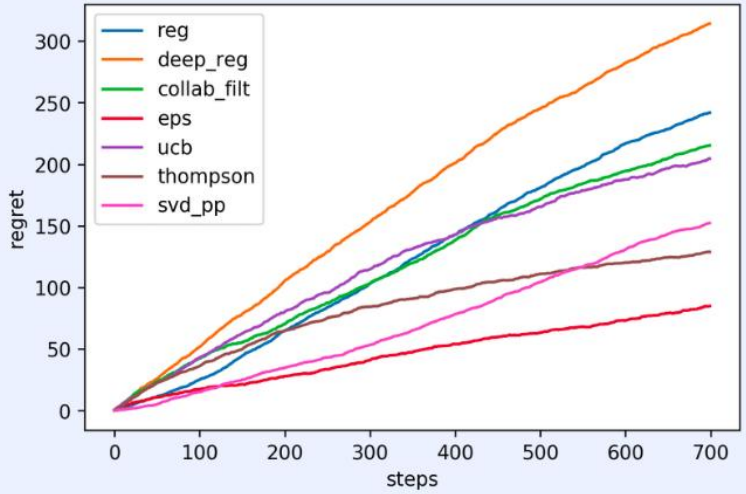
The recommender system looks at what games a given user plays and what games other people play, then makes informed suggestions based on the decisions of other people. The idea is that if players with broadly similar habits also tend to play another game that has not been tried yet, then that game is likely to be a good recommendation.

Experiments

The following experiments are run with 50 users and 20 games,
both described with context vectors of size 10 (i.e. game categories).



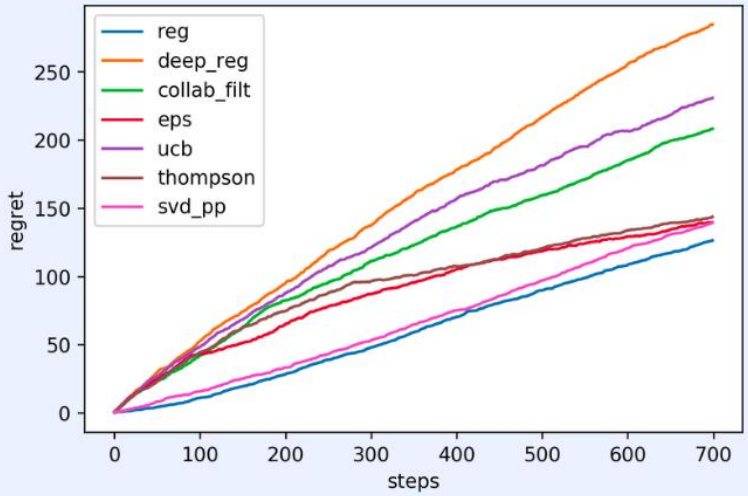
rating probability = 0.4
historical data generated = 300



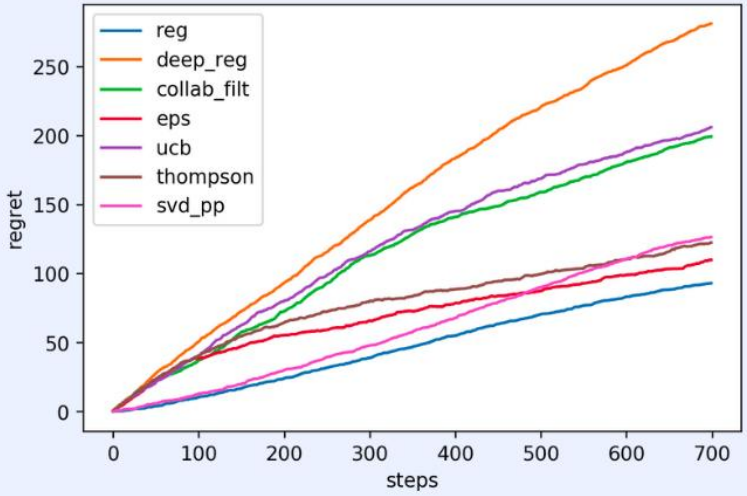
rating probability = 0.8
historical data generated = 300

Experiments

The following experiments are run with 50 users and 20 games,
both described with context vectors of size 10 (i.e. game categories).

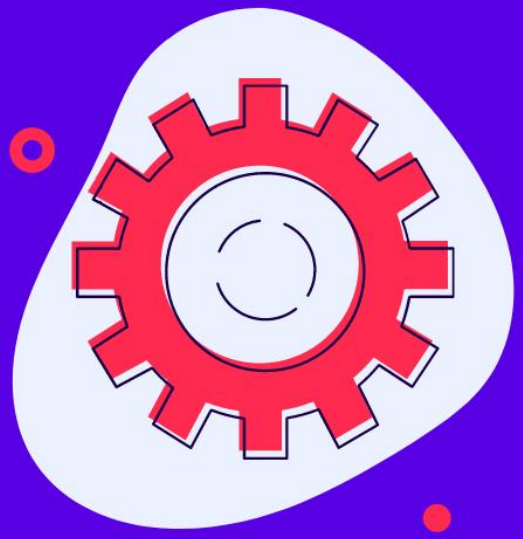


rating probability = 0.4
historical data generated = 700



rating probability = 0.8
historical data generated = 700

Further improvements



→ **01**

Re-train history-based models after a certain period of time: this might lead to great improvements in predictions.

→ **02**

Adding a database of games' features in order to model the playtime in a more realistic way.

→ **03**

Improving the reward definition, taking into account other environmental variables (i.e. price of the game, year of release, etc.).