

# RFscorer

Recency-Frequency に基づく

商品推薦スコアリング Python パッケージ

---

```
pip install rfscorer
```

GitHub: [github.com/jiro-iwanaga/rfscorer](https://github.com/jiro-iwanaga/rfscorer) PyPI: [pypi.org/project/rfscorer](https://pypi.org/project/rfscorer)

**rfscorer (v0.5.3) : MIT License** / **本資料 (v0.1.2) : CC BY 4.0**

© 2026 Erdos Inc.

# 目次

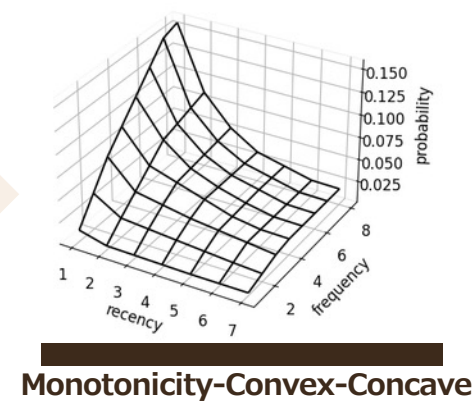
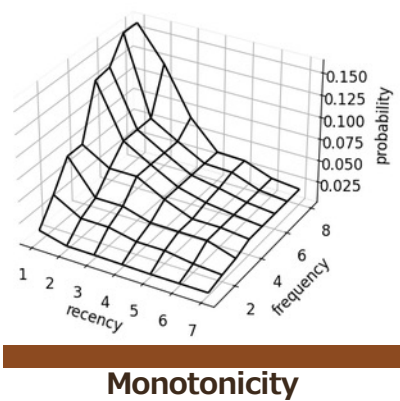
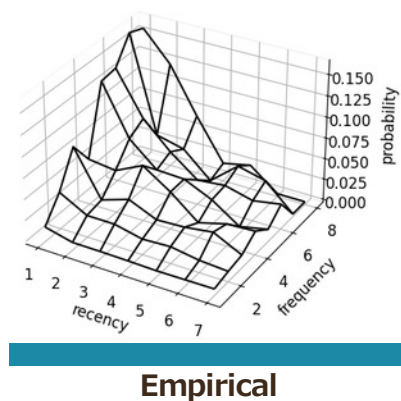
- RFスコアリングによる商品推薦
- 推薦システムの課題
- RFスコアリングを用いた推薦システム構成
- 対象ユーザー
- インストールと使い方
- 入出力データ
- パッケージの特徴
- 引用方法
- まとめ
- Appendix : 数理モデル・API・チュートリアル・開発者

# RFスコアリングによる商品推薦

**rfscorer** は、最新度（**Recency**）と頻度（**Frequency**）に基づいて、ユーザーが過去に接触した商品の推薦スコア（商品選択確率）を推定する Python パッケージです。

- 最新度（**Recency**）：最近接触した商品ほど関心が高い傾向
- 頻度（**Frequency**）：何度も接触した商品ほど関心が高い傾向

数理最適化により最新度と頻度の単調性を満たす商品選択確率を推定することで、過去に接触した商品を推薦するタスクで、直感に合う推薦順位を実現します。



Based on the paper:

Jiro Iwanaga, Naoki Nishimura, Noriyoshi Sukegawa, and Yuichi Takano, "Estimating product-choice probabilities from recency and frequency of page views," *Knowledge-Based Systems*, Vol. 99, 2016, pp. 157–167.

## 推薦システムの課題

推薦システムは、過去の行動履歴を用いて、未来に推薦する商品の優先順位を決める問題と捉えられます。

### 過去の行動履歴

商品	7月5日	7月6日
A		1 閲覧
B		2 閲覧
C	1 閲覧	
D	2 閲覧	

### 未来（7月7日）に推薦する商品の検討

- 【頻度判断】 1回閲覧したA vs 2回閲覧したB ▶ **たくさん閲覧した B**
- 【最新度判断】 1日前に閲覧したA vs 2日前に閲覧したC ▶ **直近で閲覧した A**
- 【トレードオフ】 1日前に1回閲覧したA vs 2日前に2回閲覧したD ▶ **直感では判断できない**

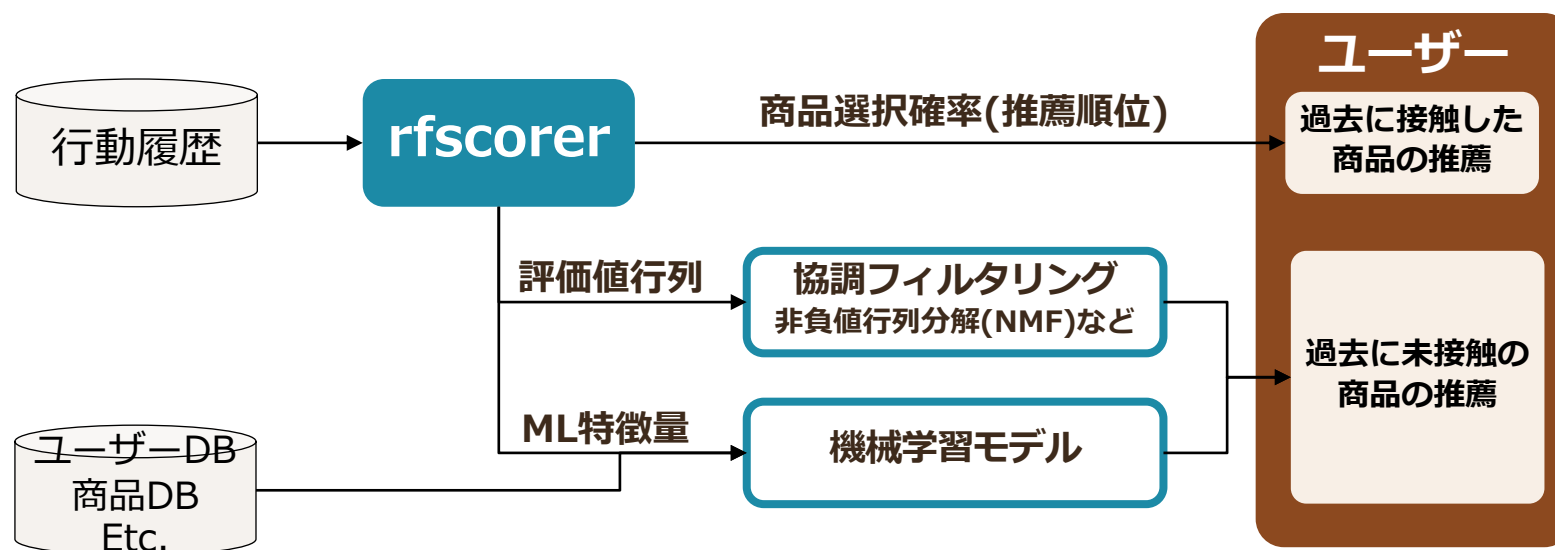
**rfscorer** を用いることで、頻度と最新度にトレードオフが発生する場合でも、

**最新度と頻度の交互作用**を考慮しつつ、

**最新度と頻度の単調性**を満たすような自然な商品推薦を実現します。

# RFスコアリングを用いた推薦システム構成

**rfscorer** が出力する商品選択確率は単独で推薦順位として利用できるだけでなく、下流のモデルの入力（協調フィルタリングの評価値行列・機械学習モデルの特徴量）としても有効です。**最新度と頻度の交互作用**が反映された有用な特徴量によるモデル構築ができます。



Based on the paper:

Jiro Iwanaga, Naoki Nishimura, Noriyoshi Sukegawa, and Yuichi Takano, "Improving collaborative filtering recommendations by estimating user preferences from clickstream data," *Electronic Commerce Research and Applications*, Volume 37, Article 100877, 2019.

# 想定ユーザー

## 実務家

- データサイエンティスト・ML エンジニア
- マーケター・アナリスト
- EC・コンテンツ基盤の推薦システム担当者

## 研究者

- 推薦システム・情報検索
- マーケティングサイエンス・消費者行動論
- オペレーションズリサーチ・数理最適化
- 認知心理学（記憶・単純接触効果）

# インストールと使い方

## pip インストール

```
$ pip install rfscorer
```

## 使い方 (fit → optimize → transform の3ステップで推薦スコアを算出)

```
from rfscorer import RecencyFrequencyScorer, split_by_date

# 行動履歴（カラム: user, item, datetime）を基準日で、観測データ7日分、正解データ1日分に分割
df_obs, df_gt = split_by_date(df, "2026-07-07", 7, 1)

scorer = RecencyFrequencyScorer()
scorer.fit(df_obs, df_gt)          # モデル構築
scorer.optimize(kind="mono")      # RF単調性で最適化

# テストデータに推薦スコアを付与
df_scores = scorer.transform(df_test_obs, "2026-07-07", kind="mono")
```

# 入出力データ

入力：3 カラムの行動履歴

user	item	datetime
u011	i144	2026-07-05
u042	i091	2026-07-06
u044	i077	2026-07-07
u044	i072	2026-07-07

出力：各 user × item に推薦スコアと推薦順位を付与したテーブル

user	item	recency	frequency	probability	order
u011	i144	3	4	0.0767	1
u042	i091	2	3	0.0789	1
u044	i077	1	2	0.0621	1
u044	i072	1	1	0.0248	2

各ユーザーに対して商品選択確率（probability）が高い順に商品を推薦



## パッケージの特徴

- 01 【scikit-learn ライク】 fit() / transform() のシンプルなAPI
- 02 【最小限のデータ要件】 user / item / datetime の3カラムだけ
- 03 【説明可能性】 数理最適化でRF単調性を満たし推薦理由を説明
- 04 【安定した確率推定】 最新度・頻度から商品選択確率を直接推定
- 05 【下流モデルへの活用】 協調フィルタリング・MLの入力に
- 06 【豊富な診断と可視化】 統計量出力と可視化で実務・研究に対応

## 01

/ 06

## sklearn-style API

# scikit-learn ライク

fit() / optimize() / transform() という馴染みのインターフェースを提供  
既存の機械学習ワークフローに自然に対応

## HIGHLIGHTS

- fit() → optimize() → transform() の3ステップで完結
- save() / load() でモデルを保存・再利用

```
scorer = RecencyFrequencyScorer()  
scorer.fit(df_obs, df_gt)  
scorer.optimize(kind="mono")  
df_scores = scorer.transform(df_test_obs, "2026-07-07", kind="mono")  
  
scorer.save("rfscorer.pkl")  
scorer_loaded = RecencyFrequencyScorer.load("rfscorer.pkl")
```

## 最小限のデータ要件

必要な入力は user・item・datetime の 3 カラムを持つ行動履歴  
特別な前処理や追加の特徴量がなくとも利用可能

### HIGHLIGHTS

- 多くのサービスが user・item・datetime をもつ行動履歴を保有
- 追加の特徴量エンジニアリングが不要

user	item	datetime
u011	i144	2026-07-07
u042	i091	2026-07-06
u044	i077	2026-07-07
u044	i072	2026-07-07

## 説明可能性

数理最適化で RF 単調性を満たすスコアを推定  
ブラックボックス化せず、推薦理由を説明可能

### HIGHLIGHTS

- 「最近閲覧した商品やたくさん閲覧した商品ほど選ばれやすい」という直感に一致した商品推薦が可能
  - 最新度（Recency）：最近接触した商品ほど関心が高い傾向
  - 頻度（Frequency）：何度も接触した商品ほど関心が高い傾向
- 商品選択確率の根拠を、最新度と頻度に基づいて説明可能

## 安定した確率推定

最新度と頻度から商品選択確率を直接推定  
機械学習モデルの出力を確率スケールへ変換する際の不安定さを回避

### HIGHLIGHTS

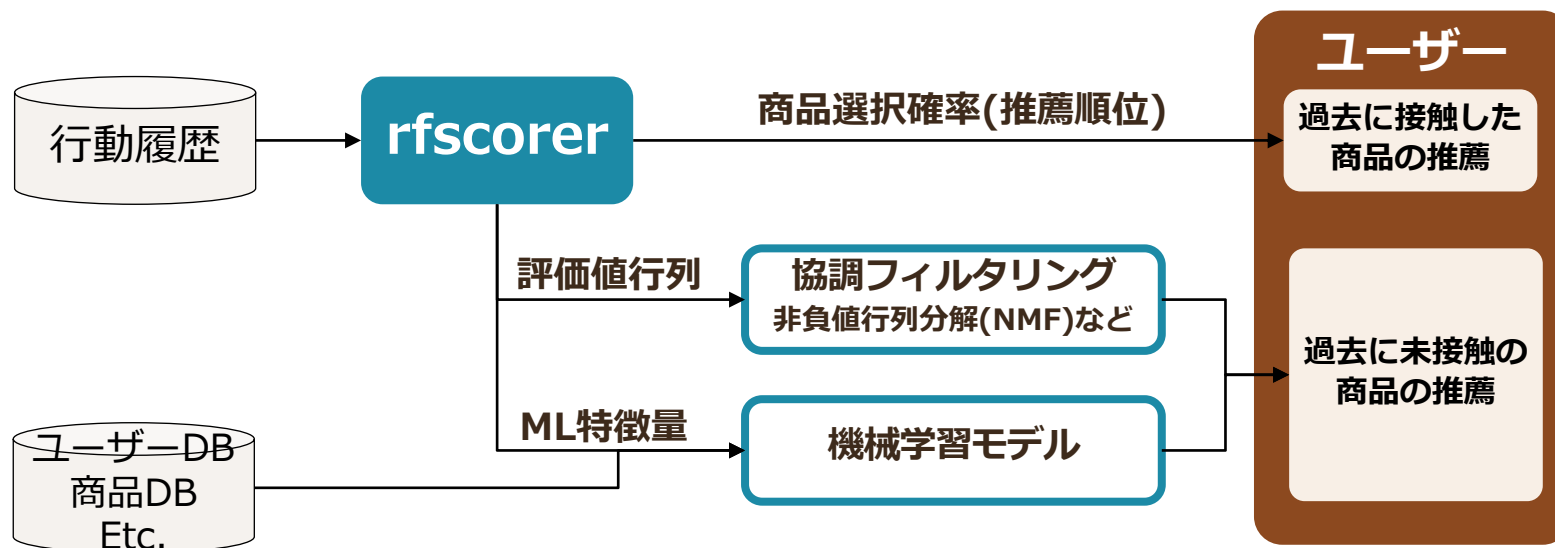
- 不安定な確率スケール変換が不要なため、確率値として扱いやすく、期待収益などの期待値計算にそのまま利用可能
- `fit_rolling()` で複数基準日を集計し経験的商品選択確率を安定化

## 下流モデルへの活用

推薦スコアは単独利用だけでなく、  
協調フィルタリングの評価値行列や機械学習モデルの特徴量に活用可能

### HIGHLIGHTS

- 最新度と頻度の交互作用が反映された有用な特徴量によるモデル構築

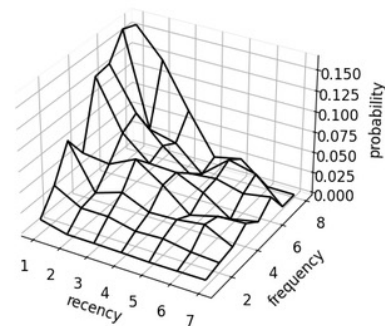


## 豊富な診断と可視化

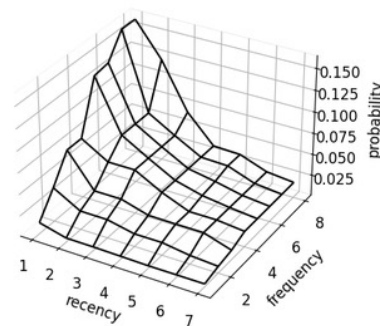
相関係数などの各種統計量の出力と可視化機能が充実  
実務では結果を説明しやすく、研究では分析結果を報告しやすい

### HIGHLIGHTS

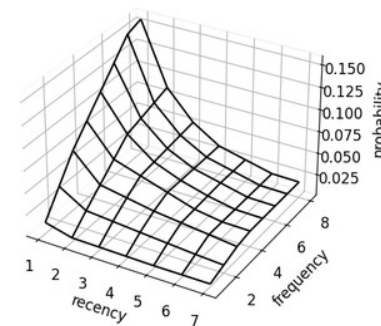
- 相関係数など各種統計量を出力
- $\text{recency} \times \text{frequency} \times \text{商品選択確率}$ を3Dサーフェスで直感的に把握



Empirical



Monotonicity



Monotonicity-Convex-Concave

# 引用方法

学術論文で rfscorer を利用する場合、本文中で次のように引用できます。

We used rfscorer (Iwanaga et al., 2016), a Python library for Recency-Frequency-based recommendation scoring.<sup>1</sup>

<sup>1</sup> <https://github.com/jiro-iwanaga/rfscorer>

## 参考文献

- Iwanaga, Nishimura, Sukegawa, Takano (2016) “Estimating product-choice probabilities from recency and frequency of page views,” Knowledge-Based Systems, Vol.99, pp.157–167.
- Iwanaga, Nishimura, Sukegawa, Takano (2019) “Improving collaborative filtering recommendations by estimating user preferences from clickstream data,” Electronic Commerce Research and Applications, Vol.37, 100877.
- 岩永二郎, 石原響太, 西村直樹, 田中一樹, Pythonではじめる数理最適化 ケーススタディでモデリングのスキルを身につけよう, オーム社, 2021 (第7章)



# まとめ

- ✓ 最新度（Recency）と頻度（Frequency）から商品選択確率を推定するPythonパッケージ
- ✓ 数理最適化で RF 単調性を満たし、解釈可能で自然な推薦順序を実現
- ✓ user / item / datetime の 3 カラムだけで導入でき、scikit-learn ライクに使える
- ✓ 単独の推薦スコアにも、協調フィルタリング・ML の入力にも活用できる
- ✓ 豊富な診断・可視化で実務の説明と研究の報告を支援

```
pip install rfscorer
```

GitHub: [github.com/jiro-iwanaga/rfscorer](https://github.com/jiro-iwanaga/rfscorer)   PyPI: [pypi.org/project/rfscorer](https://pypi.org/project/rfscorer)   MIT License

# 数理最適化モデル

- Set
  - $R = \{1, 2, 3, \dots, R_{\max}\}$
  - $F = \{1, 2, 3, \dots, F_{\max}\}$
- Variable
  - $x_{r,f} \in [0, 1] \quad (r \in R, f \in F)$
- Constant
  - $p_{r,f} \in [0, 1] \quad (r \in R, f \in F)$
  - $N_{r,f} \in \mathbb{Z}_{\geq 0} \quad (r \in R, f \in F)$
  - $\epsilon \geq 0$
- Constraint
  - Recency Monotonicity
    - $x_{r,f} \geq x_{r+1,f} + \epsilon \quad (f \in F, r, r+1 \in R)$
  - Frequency Monotonicity
    - $x_{r,f} + \epsilon \leq x_{r,f+1} \quad (r \in R, f, f+1 \in F)$
  - Recency Convex
    - $x_{r,f} - x_{r+1,f} \geq x_{r+1,f} - x_{r+2,f} \quad (f \in F, r, r+1, r+2 \in R)$
  - Frequency Concave
    - $x_{r,f+1} - x_{r,f} \geq x_{r,f+2} - x_{r,f+1} \quad (r \in R, f, f+1, f+2 \in F)$
- Objective (Minimize)
  - $\sum_{r \in R, f \in F} N_{r,f} \cdot (p_{r,f} - x_{r,f})^2$

## 数理最適化モデル

- **mono** : Monotonicity
- **mrc** : Monotonicity & Recency Convex
- **mfc** : Monotonicity & Frequency Concave
- **mcc** : Monotonicity & Convex & Concave

## 数理最適化パッケージ

- **Modeling** : CVXPY
- **Solver** : Clarabel

## API

API	説明
<code>split_by_date(df, date, obs, gt)</code>	行動履歴を観測データ・正解データに時系列分割
<code>RecencyFrequencyScorer()</code>	スコアラーを生成
<code>.fit(df_obs, df_gt)</code>	最新度×頻度ごとに経験的な商品選択確率を集計
<code>.fit_rolling(...)</code>	複数基準日でローリング集計し経験確率を安定化
<code>.optimize(kind="mono")</code>	数理最適化で単調性等の制約を満たすスコアを推定
<code>.transform(df_obs, date, kind)</code>	推薦スコア (probability / order) を算出
<code>.predict(r, f, kind)</code>	指定した (recency, frequency) の商品選択確率を返す
<code>.evaluate(df_rec, df_gt)</code>	推薦結果の精度を評価
<code>.plot_probability_surface(kind)</code>	商品選択確率の3D曲面を可視化
<code>.save(path) / .load(path)</code>	モデルの保存・読み込み

# チュートリアル

## 初級編

- `tutorial_beginner_ja.ipynb`

行動履歴の読み込み、`fit()` による経験的商品選択確率の推定、RF制約下での `optimize()`、確率曲面の可視化、`transform()` による推薦スコア算出までの `fit` → `optimize` → `transform` という流れを最短で把握できます。

## 実践編

- `tutorial_practical_ja.ipynb`

運用を想定した実践的なワークフローです。データを時系列で訓練・テストに分割し、全9種のモデル（経験的 + 最適化）を構築、`evaluate()` で推薦精度を比較し、学習済みモデルを保存・ロードして再利用します。

## 応用編

- `tutorial_advanced_fit_rolling_ja.ipynb`

`fit_rolling()` を用いて複数の基準日でローリング集計し、経験的商品選択確率を安定化させます。

- `tutorial_advanced_int_time_col_ja.ipynb`

カレンダー概念を持たない時系列データへの対応を解説します。

- `tutorial_advanced_day_freq_ja.ipynb`

頻度の上限を抑えるために、閲覧の総回数ではなく、商品を閲覧した日数を数える方法を解説します。

# 開発者

## ■ 岩永二郎 / IWANAGA Jiro

- 株式会社エルデシュ 代表取締役 / 電気通信大学 特任教授

## ■ 専門分野

- 数理最適化・推薦システム・機械学習・自然言語処理

## ■ 主要論文

- Iwanaga et al. (2016), Knowledge-Based Systems  
Estimating product-choice probabilities from recency and frequency of page views
- Iwanaga et al. (2019), Electronic Commerce Research and Applications  
Improving collaborative filtering recommendations by estimating user preferences from clickstream data

## ■ 書籍

- 岩永二郎, 石原響太, 西村直樹, 田中一樹 (2021) , オーム社, 『Pythonではじめる数理最適化』

## ■ 連絡先

- GitHub : <https://github.com/jiro-iwanaga>
- Mail : [iwanaga@erdos-the-book.com](mailto:iwanaga@erdos-the-book.com)

