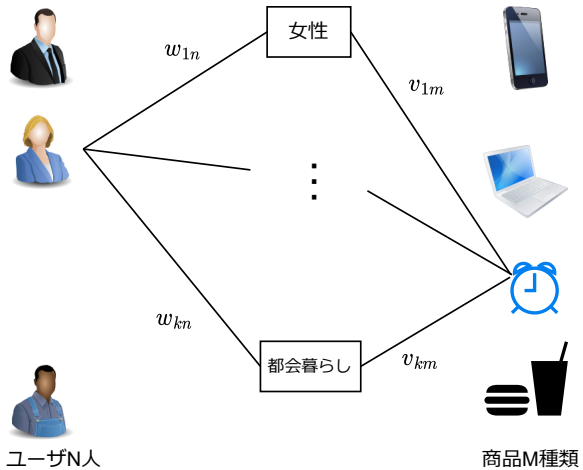
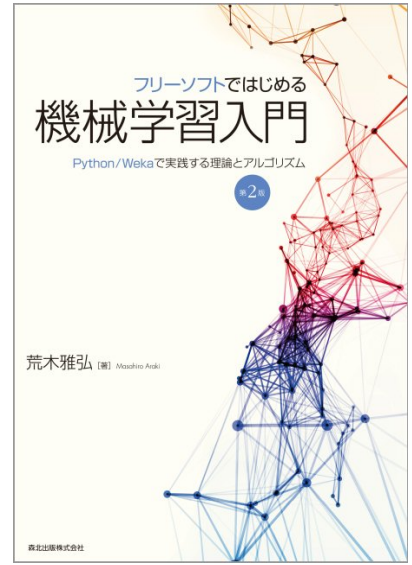
12. パターンマイニング



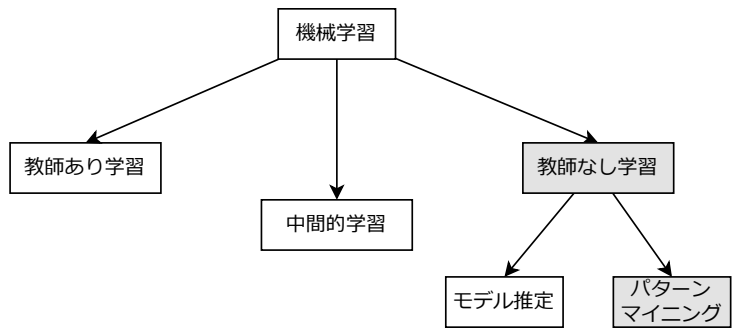
- 12.1 カテゴリ特徴・教師なし・パターンマイニングの問題設定
- 12.2 頻出項目抽出
- 12.3 連想規則抽出
- 12.4 FP-Growthアルゴリズム
- 12.5 推薦システムにおける学習
- 12.6 まとめ



- 荒木雅弘:『フリーソフトではじめる機械学習入門(第2版)』(森北出版, 2018年)
- スライドとJupyter notebook
- サポートページ

12. パターンマイニング

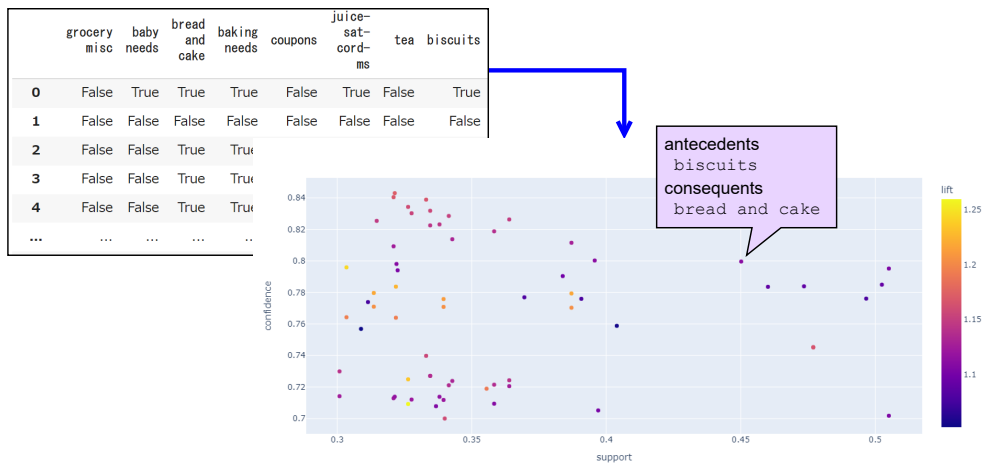
- 問題設定
  - 教師なし学習
    - (疎な)ベクトル $\rightarrow$ 規則性
    - 規則性の例
      - 頻出項目、連想規則、低次元ベクトル



  - 応用例
    - 推薦システム

12.1 カテゴリ特徴・教師なし・パターンマイニングの問題設定 (1/2)

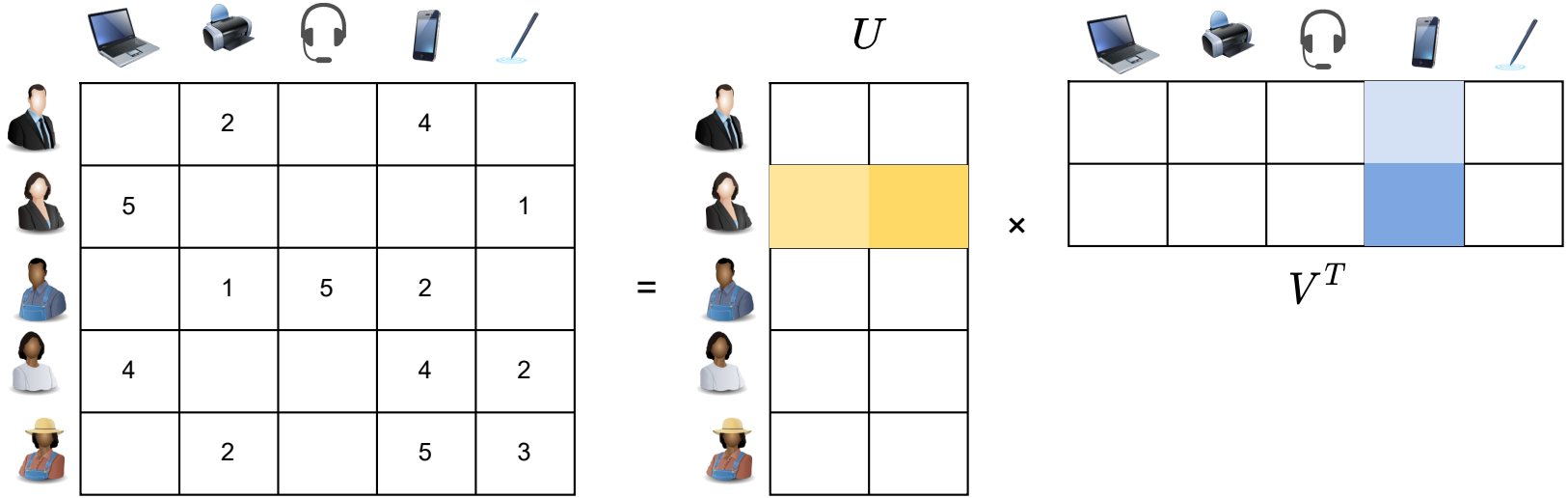
- データセット(正解なし)
  - (疎な)d次元のカテゴリまたは数値ベクトル　　$\{\boldsymbol{x}_i\}\quad i=1,\ldots,N$
    - 値が数値の場合でも、順序尺度(例:5段階評価値)のような実質的にはカテゴリとみなせるもの
- 問題設定1
  - データセット中で一定頻度以上で現れる特徴の組み合わせや規則を抽出



	grocery misc	baby needs	bread and cake	baking needs	coupons	juice-sat-cord-ms	tea	biscuits
0	False	True	True	True	False	True	False	True
1	False	False	False	False	False	False	False	False
2	False	False	True	Tru				
3	False	False	True	Tru				
4	False	False	True	Tru				
...	...	...	...	..				

12.1 カテゴリ特徴・教師なし・パターンマイニングの問題設定（2/2）

- 問題設定2
  - 似ているデータを参考にして、空所の値を予測する



12.2 頻出項目抽出

12.2.1 頻出の基準と問題の難しさ（1/2）

- 例題：バスケット分析（1件分のデータをトランザクションとよぶ）

	Milk	Bread	Butter	Magazine
0	t	t		
1		t		
2				t
3		t	t	
4	t	t	t	
5	t	t		

- バスケット分析の目的
  - トランザクション集合中で、一定割合以上出現する項目集合を抽出

12.2.1 頻出の基準と問題の難しさ（2/2）

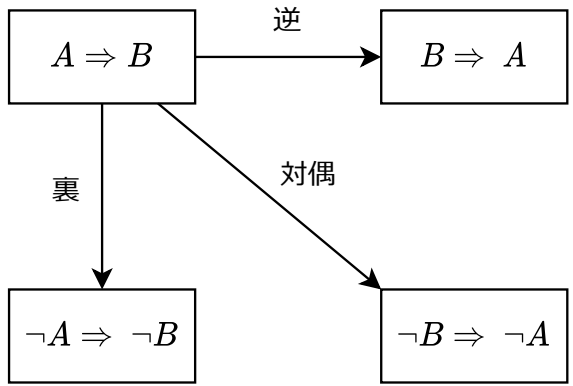
- 頻出の基準：支持度（support）
  - 全トランザクション数 T に対する、項目集合 $items$ が出現するトランザクション数 T_{items} の割合

$$\mathrm{support}(items) = \frac{T_{items}}{T}$$

- バスケット分析の問題点
  - すべての可能な項目集合について支持度を計算することは現実的には不可能
    - 商品の種類数 1,000 の店なら、項目集合の数は 2^{1000}
  - 高頻度の項目集合に絞って計算を行う必要がある

12.2.2 Aprioriアルゴリズムによる頻出項目抽出 (1/7)

- 命題論理
  - 「AならばB」が成り立つなら、その対偶である「¬Bならば¬A」は必ず成り立つ
    - $A \Rightarrow B$ は $\neg A \vee B$ と定義されている
    - この定義より $\neg B \Rightarrow \neg A$ は $\neg(\neg B) \vee (\neg A)$ なので、$B \vee \neg A$ となり、$\neg A \vee B$ と等しい

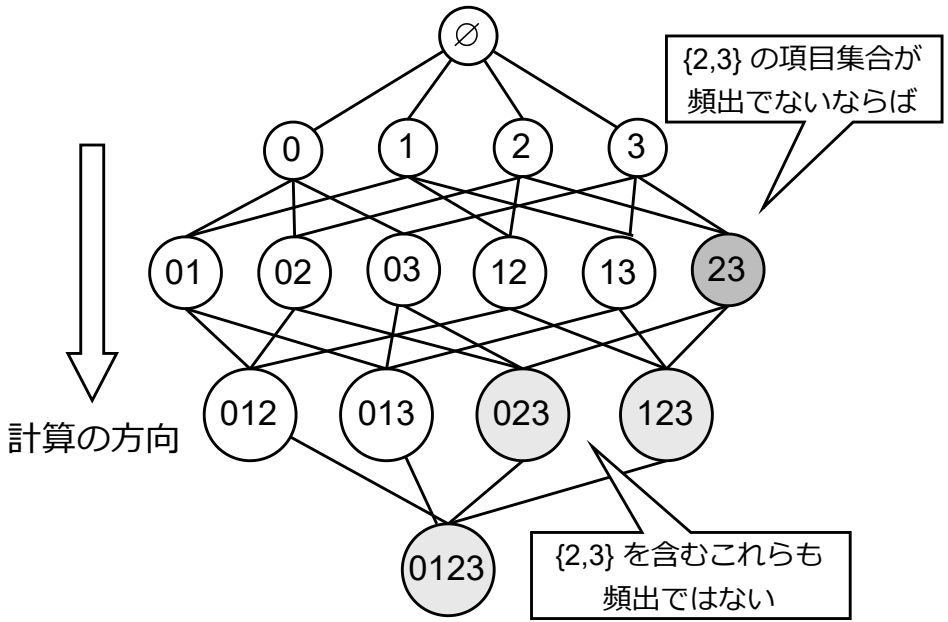


12.2.2 Aprioriアルゴリズムによる頻出項目抽出（2/7）

- a prioriな原理:「ある項目集合が頻出」ならば「その部分集合も頻出である」
  - 例）「パン・ミルク」が頻出ならば「パン」も頻出
- 対偶:「ある項目集合が頻出でない」ならば「その項目集合を含む上位集合も頻出ではない」
  - 例）「バター・雑誌」が頻出でないならば「バター・雑誌・パン」も頻出でない

12.2.2 Aprioriアルゴリズムによる頻出項目抽出 (3/7)

- a prioriな原理の対偶を用いて頻出項目集合の候補を削減



12.2.2 Aprioriアルゴリズムによる頻出項目抽出 (4/7)

- 頻出項目集合抽出の手順
  1. 項目数1の集合 C_1を求める
  2. C_1 から支持度が閾値以下の要素を削除し、集合 F_1 を求める
  3. $k = 2$ から始め、$F_k = \varnothing$(空集合)になるまで以下を繰り返す
     1. F_{k-1} の要素を組合せ、項目数 k の集合 C_k を作成する
     2. C_k の要素で、その部分集合が F_{k-1} に含まれないものを削除する
     3. C_k から支持度が閾値以下の要素を削除し、F_k とする

12.2.2 Aprioriアルゴリズムによる頻出項目抽出（5/7）

- コーディング
  - mlxtendライブラリで実装されている `apriori`, `association_rules` を利用
  - 例題12.1 のデータで動作確認

```
import numpy as np
import pandas as pd
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori, association_rules, fpgrowth

dataset = [
    ['Milk', 'Bread', 'Butter'],
    ['Milk', 'Bread', 'Jam'],
    ['Milk', 'Magazine'],
    ['Bread', 'Butter'],
    ['Milk', 'Bread', 'Butter', 'Jam'],
    ['Magazine'],
    ['Milk', 'Bread', 'Jam', 'Magazine'],
    ['Jam']]
```

12.2.2 Aprioriアルゴリズムによる頻出項目抽出（6/7）

- 前処理 `TransactionEncoder`
  - 疎行列形式の表現を、真偽値を値とする行列に変換

```
te = TransactionEncoder()
te_ary = te.fit_transform(dataset)
df = pd.DataFrame(te_ary, columns=te.columns_)
```

	Bread	Butter	Jam	Magazine	Milk
0	True	True	False	False	True
1	True	False	True	False	True
2	False	False	False	True	True
3	True	True	False	False	False
4	True	True	True	False	True
5	False	False	False	True	False
6	True	False	True	True	True
7	False	False	True	False	False

12.2.2 Aprioriアルゴリズムによる頻出項目抽出（7/7）

- `apriori`メソッドのパラメータ
  - `min_support`：支持度の最小値。デフォルトは0.5
  - `use_colnames`：結果の表示が列名か（True）、列番号か（False:デフォルト）を指定

```
freq = apriori(df, min_support= 3/len(df), use_colnames=True)
```

	support	itemsets
0	0.625	(Bread)
1	0.375	(Butter)
2	0.500	(Jam)
3	0.375	(Magazine)
4	0.625	(Milk)
5	0.375	(Butter, Bread)
6	0.375	(Bread, Jam)
7	0.500	(Milk, Bread)
8	0.375	(Milk, Jam)
9	0.375	(Milk, Bread, Jam)

12.3 連想規則抽出

- 規則抽出
  - 正解付きデータに対して、正解を目的変数とみなし、それに対する入力変数の関係を記述
- 連想規則抽出
  - 正解なしデータに対して、特徴間の強い相関関係を記述
    - 例 :「商品Aを買った人は商品Bも買う傾向がある」というような規則性を抽出したい
  - 評価値の高い規則を抽出
    - 確信度: 前提部Aが起こったときに結論部Bが起こる割合

$$\mathrm{confidence}(A \to B) = \frac{\mathrm{support}(A \cup B)}{\mathrm{support}(A)} = \frac{T_{A \cup B}}{T_A}$$

    - リフト値: Bだけが単独で起こる割合とAが起こったときにBが起こる割合との比

$$\mathrm{lift}(A \to B) = \frac{\mathrm{confidence}(A \to B)}{\mathrm{support}(B)}$$

12.3.3 規則の有用性

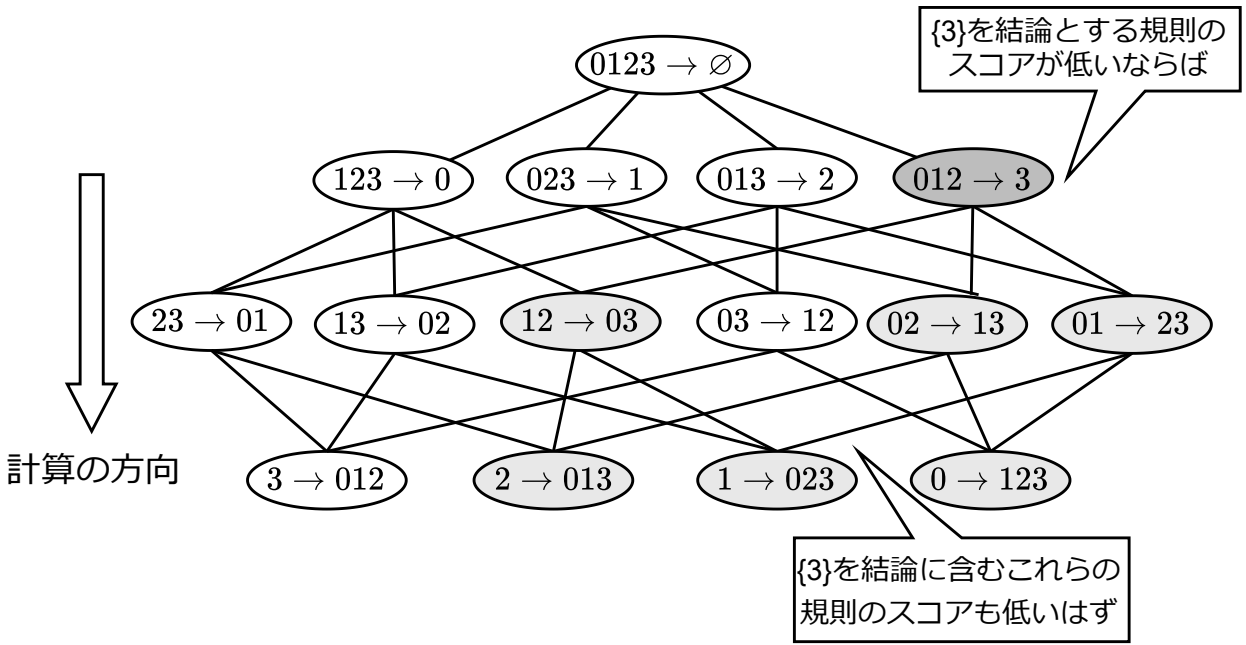
- 支持度・確信度・リフト値の意味
  - support({ハム, 卵}) : 0.1
    - 顧客全体のうち、10%の顧客がハムと卵を一緒に購入している
  - confidence(ハム⇒卵) : 0.7
    - ハム購入者の70%が卵も購入している
  - lift(ハム⇒卵): 5
    - ランダムに選んだ顧客が卵を買う確率に対して、ハムを買った顧客が卵を買う確率は5倍大きい

12.3.4 Aprioriアルゴリズムによる連想規則抽出（1/4）

- a prioriな原理:「ある項目集合を結論部に持つ規則」が頻出ならば、「その部分集合を結論部に持つ規則」も頻出である
  - 例）結論部が「パン・ミルク」の規則が頻出ならば、結論部が「パン」の規則も頻出である
- 対偶:「ある項目集合を結論部に持つ規則」が頻出でないならば、「その上位集合を結論部に含む規則」も頻出ではない
  - 例） 結論部が「雑誌」の規則が頻出でないならば、結論部が「パン・雑誌」の規則も頻出ではない

12.3.4 Aprioriアルゴリズムによる連想規則抽出（2/4）

- a priori原理に基づく探索



12.3.4 Aprioriアルゴリズムによる連想規則抽出（3/4）

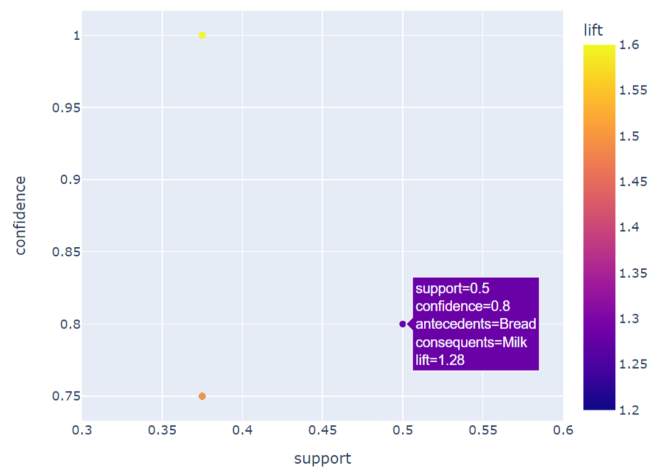
- 連想規則抽出の手順
  1. 頻出項目集合を求める
  2. 求めた頻出項目集合の要素のそれぞれについて、その要素中のひとつの要素を結論部、残りの要素を前提部とした規則集合 H_1 を作成する。そして、H_1 からスコア（確信度またはリフト値）が閾値以下の要素を削除
  3. $k = 2$ から始め、$H_k = \varnothing$（空集合）になるまで以下を繰り返す
     1. H_{k-1} の各要素について、前提部から結論部へ項目を1つ移動した規則を作成して H_k とする
     2. H_k から結論部が H_{k-1} の結論部を組み合わせたものでない要素を削除
     3. H_k からスコアが閾値以下の要素を削除

12.3.4 Aprioriアルゴリズムによる連想規則抽出（4/4）

- `association_rules`メソッドのパラメータ
  - `metric`：評価基準。デフォルトは`confidence`
  - `min_threshold`：抽出する最小値。デフォルトは0.8
- 抽出結果の可視化

```
association_rules(freq, metric='confidence', min_threshold=0.7)
```

	antecedents	consequents	support	confidence	lift
0	(Butter)	(Bread)	0.375	1.00	1.60
1	(Jam)	(Bread)	0.375	0.75	1.20
2	(Bread)	(Milk)	0.500	0.80	1.28
3	(Milk)	(Bread)	0.500	0.80	1.28
4	(Jam)	(Milk)	0.375	0.75	1.20
5	(Jam, Bread)	(Milk)	0.375	1.00	1.60
6	(Jam, Milk)	(Bread)	0.375	1.00	1.60
7	(Bread, Milk)	(Jam)	0.375	0.75	1.50
8	(Jam)	(Bread, Milk)	0.375	0.75	1.50



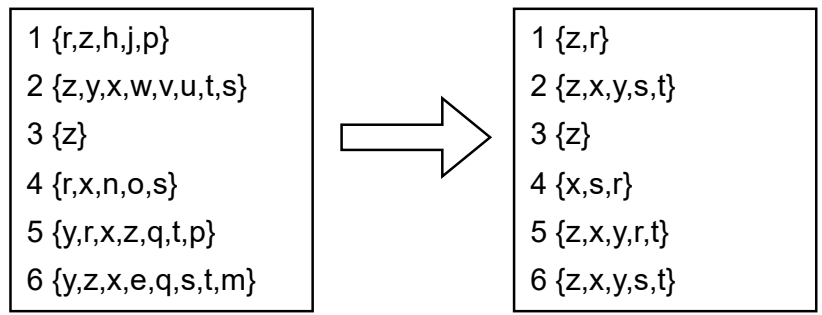
12.4 FP-Growthアルゴリズム (1/4)

- Aprioriアルゴリズムの高速化
  - トランザクションをコンパクトに表現し、重複計算を避ける

1. トランザクションの前処理
    1. トランザクションを出現する特徴名の集合に変換
    2. 各集合を出現頻度順にソート
    3. 低頻度特徴をフィルタリング
2. prefixを共有する木構造(FP木)に順次挿入
3. FP木を用いて項目集合の出現頻度を高速計算

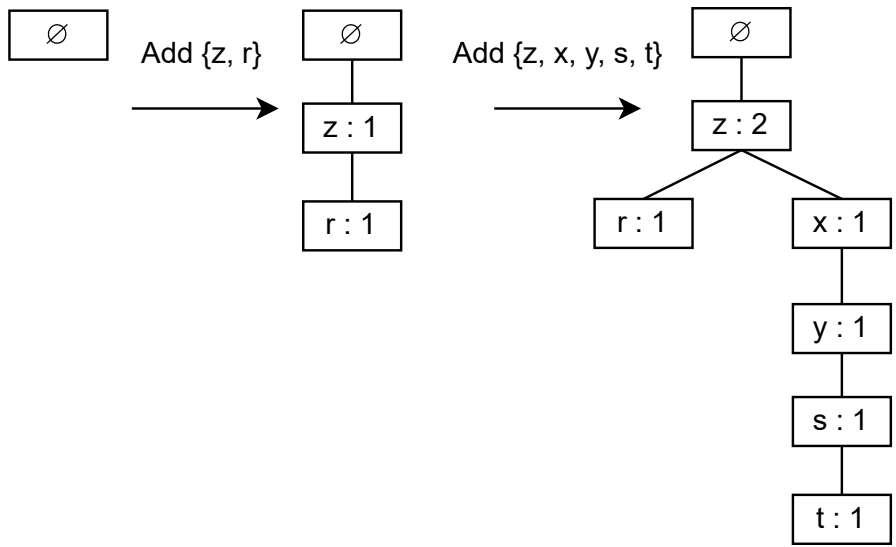
12.4 FP-Growthアルゴリズム（2/4）

- トランザクションの前処理
  - トランザクションを出現する特徴名の集合に変換
  - 各集合を出現頻度順にソート
  - 低頻度特徴をフィルタリング



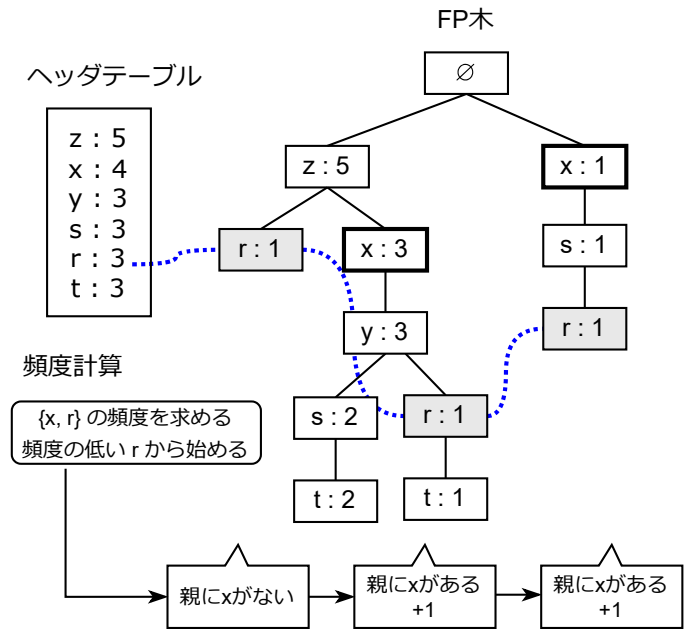
12.4 FP-Growthアルゴリズム (3/4)

- prefixを共有する木構造(FP木)を作成
  - ソート、フィルタリング後のトランザクションデータを順次FP木に挿入



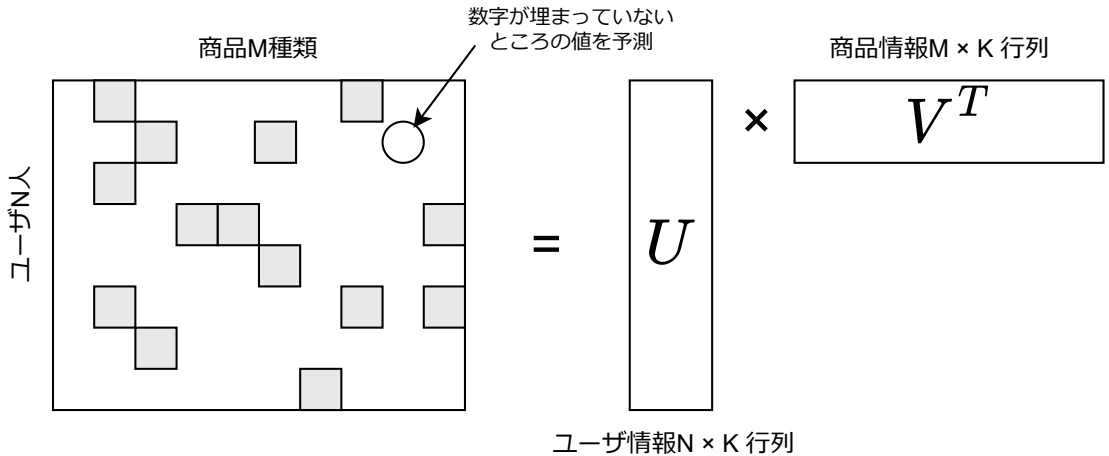
12.4 FP-Growthアルゴリズム (4/4)

- FP木を用いて項目集合の出現頻度を高速計算
  - mlxtendでの実装は メソッド名 `apriori` を `fpgrowth` に変更するだけ



12.5 推薦システムにおける学習（1/6）

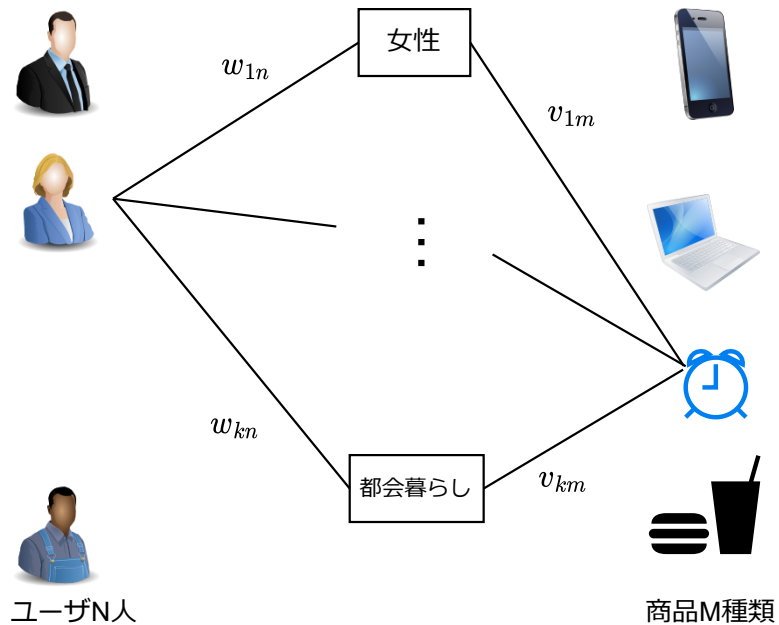
- 協調フィルタリング
  - アイデア：疎な行列は低次元の行列の積で近似できる
  - 値のある部分だけで行列分解を行う
  - 空所の値を予測する



12.5 推薦システムにおける学習（2/6）

- 潜在因子によるデータ表現の考え方

$$x_{mn} = w_{1n}v_{1m} + w_{2n}v_{2m} + \cdots + w_{kn}v_{km}$$



12.5 推薦システムにおける学習（3/6）

- 行列分解の方法
  - $\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T$ の最小化問題を解く

$$\min_{\boldsymbol{U},\boldsymbol{V}} \frac{1}{2}\|\boldsymbol{E}\|_{Fro}^2 = \min_{\boldsymbol{U},\boldsymbol{V}} \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T\|_{Fro}^2$$

    - 空欄を値0とみなしてしまっている
  - 値が存在する要素だけに限って二乗誤差を最小化

$$\min_{\boldsymbol{U},\boldsymbol{V}} \sum_{(i,j)\in\Omega} (x_{ij} - \boldsymbol{u}_i^T\boldsymbol{v}_j)^2 + \lambda_1\|\boldsymbol{U}\|_{Fro}^2 + \lambda_2\|\boldsymbol{V}\|_{Fro}^2$$

  - $\boldsymbol{U}, \boldsymbol{V}$ の要素を非負に限定したものが非負値行列因子分解（NMF：Nonnegative Matrix Factorization）

12.5 推薦システムにおける学習（4/6）

- 例題：映画評価データの非負値行列分解
  - 行がユーザ（5人）、列が映画（4作品）、数値が1-5の5段階評価で0は評価なし

```
import numpy as np
from sklearn.decomposition import NMF

X = np.array([
    [5,3,0,1],
    [4,0,0,1],
    [1,1,0,5],
    [1,0,0,4],
    [0,1,5,4]
])
```

12.5 推薦システムにおける学習（5/6）

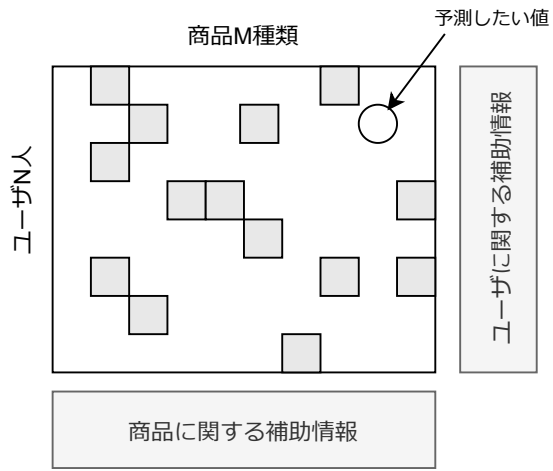
- 負値行列分解 `NMF` のパラメータ
  - `n_components`: 潜在変数の次元数。デフォルトは `None` ですべての特徴が保存される
- その積がXに近くなるような非負値行列W, Hが得られる
  - Wは人の傾向を、Hの転置は映画の傾向を表す2次元ベクトルのリストになっている

```
model = NMF(n_components = 2)
W = model.fit_transform(X)
H = model.components_
np.dot(W,H)
```

```
array([[5.2558264 , 1.99313836, 0.        , 1.45512772],
       [3.50429478, 1.32891458, 0.        , 0.9701988 ],
       [1.31294288, 0.94415991, 1.94956896, 3.94609389],
       [0.98129195, 0.72179987, 1.52759811, 3.0788454 ],
       [0.        , 0.65008935, 2.84003662, 5.21894555]])
```

12.5 推薦システムにおける学習（6/6）

- Factorization Machine
  - 補助情報を予測に取り入れることができる行列分解　$y = w_0 + w_i + w_j + \boldsymbol{v}_i^T \boldsymbol{v}_j$
    - w_0: 定数項、w_i: ユーザiのバイアス、w_j: 商品jのバイアス
    - $\boldsymbol{v}_i^T \boldsymbol{v}_j$: 補助情報を含む任意の要素間で定義可能な交互作用



12.6 まとめ

- パターンマイニングは有用な規則性を発見する
- Aprioriアルゴリズム
  - 出現頻度の高い項目集合を見つける
  - 出現頻度に基づき、有用な規則を見つける
  - FPGrowthはAprioriアルゴリズムの高速化版
- 行列分解
  - 低次元ベクトル表現を見つけることにより、未知の値の予測を行う