Bayesian Personalized Ranking from Implicit Feedback

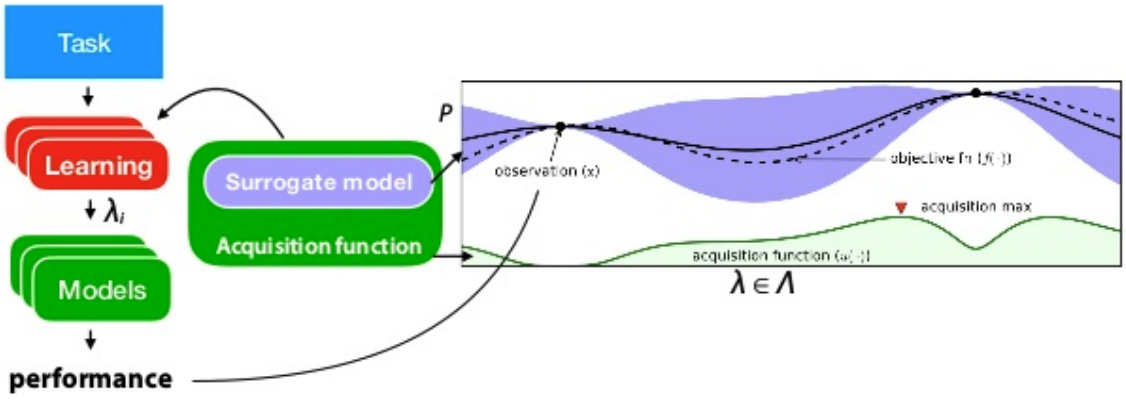
Bayesian Optimization

어떤 입력 값 x를 받는 미지의 목적 함수 f(objective function)를 상정하여, 그 함수 값 f(x)를 최대로 만드는 최적해 x를 찾는 것을 목적으로 한다. 가능한 한 적은 입력 값 후보들에 대해서만 그 함수 값을 순차적으로 조사해서, f(x)를 최대로 만드는 최적 해를 빠르고 효과적으로 찾는 것이 주요 목표다.

Rasmussen 2014

Bayesian optimization

- Consider space of all configuration options (e.g. all possible neural nets or pipelines)
- **Surrogate model**: *probabilistic* regression model of configuration performance
- **Acquisition function**: selects next configuration to try (exploration-exploitation)



13

Bayesian Optimization에는 두 가지 필수 요소가 존재한다

1. Surrogate Model - 현재 까지 조사된 입력-함수 값 점들 [(a, f(a)), (b, f(b)), (c, f(c))]을 바탕으로, 미지의 목적 함수의 형태에 대한 확률 적인 추정을 수행하는 모델
2. Acquisition Function - 목적 함수에 대한 현재까지의 확률적 추정 결과를 바탕으로, '최적 입력 값을 찾는데 가장 유용한' 다음 입력 값 후보를 추천해 주는 함수를 지칭한다.

알고리즘 Bayesian Optimization

for $t = 1, 2, \ldots$ **do**
  기존 입력값-함숫값 점들의 모음 $(x_1, f(x_1)), (x_2, f(x_2)), \ldots, (x_t, f(x_t))$ 에 대한 Surrogate Model의 확률적 추정 결과를 바탕으로, Acquisition Function을 최대화하는 다음 입력값 후보 x_{t+1}을 선정한다.
  입력값 후보 x_{t+1} 에 대한 함숫값 $f(x_{t+1})$ 을 계산한다.
  기존 입력값-함숫값 점들의 모음에 $(x_{t+1}, f(x_{t+1}))$ 를 추가하고, Surrogate Model로 확률적 추정을 다시 수행한다.
end for

Bayesian Optimization 알고리즘의 의사 코드(pseudo-code)

Surrogate Model

미지의 목적 함수의 대략적인 형태를 확률적으로 추정하는 모델. 가장 많이 사용하는 확률 모델로 Gaussian Process가 있다.

Acquisition Function

Surrogate Model이 목적 함수에 대하여 확률적으로 추정한 현재 까지 결과를 바탕으로, 다음 번에 조사할 입력 값 후보를 추천해주는 함수. 유용할 만한 다음 입력 값을 찾는데 있어서 Exploration 전략과 Exploration 전략을 조절하는 것이 성공적인 탐색에 매우 중요하다

Abstract

본 논문은 BPR-OPT를 제시해, Item들에 대한 User의 선호 강도를 반영할 수 있도록 하였다. 이를 MF 와 KNN에 적용한 결과, 기존의 것보다 우수함을 증명하였다

Contribution

ROC 커브 아래의 면적인 AUC (Area Under Curve)를 최대화 하는 문제와 BPR-OPT와 동치 임을 보였다

BPR-OPT를 최대화 하기 위한 알고리즘 LEARN BPR을 제안하고 MF, KNN에 적용하는 방법을 제시한다

Personalized Ranking

본 논문의 목표는 각 User 별 Personalized Total Ranking 을 구하는 것이다

- personalized total ranking ($>_u \subset I^2$) 이란?

예를 들면, 5개의 Item 집단이 있으면 User가 선호할 만한 Item을 순서대로 예측

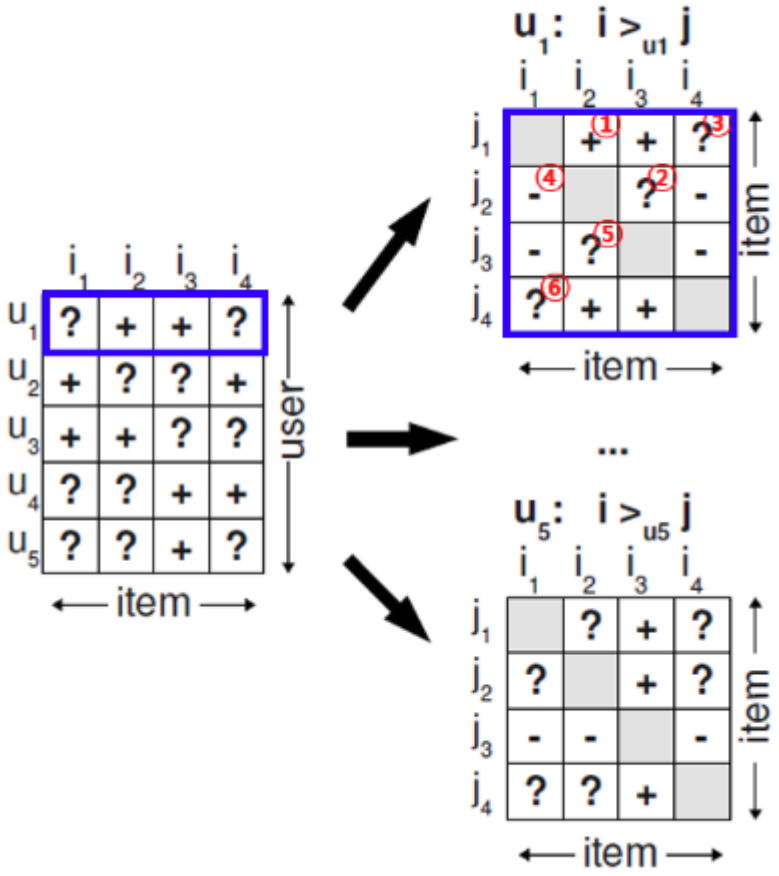
이어 $>_u$는 다음과 같은 속성을 만족한다.

$$\begin{aligned}
&\forall i,j \in I : i \neq j \Rightarrow i >_u j \ \vee \ j >_u i && \text{(totality)} \\
&\forall i,j \in I : i >_u j \ \wedge \ j >_u i \Rightarrow i = j && \text{(antisymmetry)} \\
&\forall i,j \in I : i >_u j \ \wedge \ j >_u k \Rightarrow i >_u k && \text{(transitivity)}
\end{aligned}$$

위의 속성들은 item간의 order를 정의하기 위한 조건으로, 집합 론에서 사용 되는 개념이다.

> 예를 들어 antisymmetry 조건은, 어떤 user가 i 아이템 보다 j 아이템을 선호하는 동시에 j 아이템 보다 i 아이템을 선호한다면, 두 아이템은 서로 같다는 것을 의미한다

Problem Design



- (A1) user는 관측된 item을 관측되지 않은 모든 item 보다 더 선호한다
- (A2) 관측된 item들에 대해서는 선호 강도를 추론할 수 없다(어떤 item을 더 선호하는지 모름)
- (A3) 관측되지 않은 item들에 대해서도 선호 강도를 추론 할 수 없다(어떤 item을 비 선호하는지 알 수 없음)

- $(1)\ i_2 > j_1 \Leftrightarrow (4)\ i_1 < j_2 \quad \text{by A1}$
- $(2)\ i_3\ ?\ j_2 \Leftrightarrow (5)\ i_2\ ?\ j_3 \quad \text{by A2}$
- $(3)\ i_4\ ?\ j_1 \Leftrightarrow (6)\ i_1\ ?\ j_4 \quad \text{by A3}$

위의 문제 정의 방식은 아래와 같은 두 가지 특징을 가진다.

1. 관측되지 않은 item에도 정보를 부여 (by A1)해, missing value를 간접적으로 학습시킬 수 있도록 한다. (모두 0으로 간주하는 것이 아니기 때문에 더 좋은 결과가 보장 된다)
2. 관측되지 않은 item들에 대해서도 순서를 매길 수 있다.

일반적인 Bayesian Optimization이란, 사후 확률을 최대화 하는 파라미터를 찾는 것이다. 이를 Maximum A Posteriori Estimation이라고 한다.

$$p(\Theta|>_u) \propto p(>_u|\Theta)\,p(\Theta)$$

$$\because\ p(\Theta|>_u) = \frac{p(\Theta,>_u)}{p(>_u)} = \frac{p(>_u|\Theta)p(\Theta)}{p(>_u)} \propto p(>_u|\Theta)\,p(\Theta)$$

"사후 확률"이란 "사후"라는 말에서 알 수 있듯, "어떤 정보"가 고려된 파라미터에 대한 확률. user의 선호 정보가 "어떤 정보"가 고려된 파라미터에 대한 확률이 된다.

최대 사후 확률 추정의 목표는 > 정보 즉 user의 선호 정보가 주어졌을 때, 이를 최대한 잘 나타낼 수 있는 파라미터를 추정하는 것이다.

사후 확률은 베이즈 정리에 의해 likelihood와 사전 확률의 곱으로 나타낼 수 있다.

likelihood

>u 에 대한 확률 분포. 아이템 i > j, j > i 에 대한 경우는 두 가지 경우 밖에 존재하지 않으므로 베르누이 분포를 따른다고 볼 수 있다

$$>_u\ := (u,i,j) \in D_s\ \Rightarrow \delta((u,i,j) \in D_S) \overset{iid}{\sim} B(1,\ p(i >_u j))$$

$$where,\quad \delta((u,i,j) \in D_S) := \begin{cases} 1 & \text{if } (u,i,j) \in D_S \\ 0 & \text{if } (u,i,j) \notin D_S \end{cases}$$

베르누이 분포의 likelihood function은 아래와 같다.

$$p(>_u|\Theta) = p(i >_u j)^{\delta((u,i,j)\in D_S)}(1-p(i>_u j))^{\delta((u,i,j)\notin D_S)}$$

유저가 아이템 i 보다 아이템 j를 선호할 확률인 p(i >_u j) 를 정의해야 한다. x 의 sigmoid로 정의한다. x를 구하기 위해 Matrix Factorization이나 Adaptive KNN 등을 사용한다.

$$p(i >_u j) := \sigma(\hat{x}_{uij}), \quad \sigma(x) := \frac{1}{1+e^{-x}}$$

모든 user는 iid 이므로, 모든 user에 대해 고려한 likelihood function은

$$\prod_{u\in U} p(>_u|\Theta) = \prod_{(u,i,j)\in D_s} p(i>_u j|\Theta) = \prod_{(u,i,j)\in D_s} p(i>_u j)^{\delta((u,i,j)\in D_S)}(1-p(i>_u j))^{\delta((u,i,j)\notin D_S)}$$

사전 확률

사전 확률은 파라미터에 대한 확률 분포를 가정하는 것이다. 특별한 사전 정보가 없다면 일반적으로 uniform이나, normal을 사용한다.

$$p(\Theta) \sim N(\mathbf{0},\ \lambda_\Theta I)$$

$$N(\Theta|\mu,\Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} exp(-\frac{1}{2}(\Theta-\mu)^T\Sigma^{-1}(\Theta-\mu))$$

$$\propto exp(-\frac{1}{2}\Theta^T(\frac{1}{\lambda_\Theta}I)\Theta)$$
$$= exp(-\frac{1}{2\lambda_\Theta}\Theta^T\Theta) \simeq exp(-\lambda_\Theta\|\Theta\|^2)$$

BPR - OPT

likelihood 와 사전 확률 정의 했다면, 사후 확률을 최대화하는 파라미터를 구할 차례다.

$$\text{BPR-OPT:}\ \arg\max_\Theta\ p(\Theta|>_u)$$

Matrix Factorization에 적용을 했을 때

(1) $\Theta = \{\mathbf{p}_u, \mathbf{q}_i\}$ (여기서 $\mathbf{p}_u$, $\mathbf{q}_i$는 각각 user, item latent vector)

(2) $\hat{x}_{uij} := \hat{x}_{ui} - \hat{x}_{uj} = \mathbf{p}_u\mathbf{q}_i^T - \mathbf{p}_u\mathbf{q}_j^T$

Matrix Factorization은 user와 item의 latent vector를 예측하는 것이므로 p_u, q_i 가 파라미터가 된다. 다음 user와 item i, item j 간의 관계는 user의 item i 에 대한 점수와 item j에 대한 점수의 차이로 정의된다. 차이는 양수이고, 클수록 아이템 i 보다 아이템 j를 선호할 확률이 1에 가까워 진다.

optimization function은 다음과 같다

$$
\begin{aligned}
\log p(\Theta| >_u) &= \log p(>_u|\Theta)p(\Theta) \\
&= \log\Big(\prod_{(u,i,j)\in D_s} \sigma(\hat{x}_{uij})\, p(\Theta)\Big) \\
&= \sum_{u,i,j} \log\sigma(\hat{x}_{uij}) + \log p(\Theta) \\
&= \sum_{u,i,j} \log\sigma(\hat{x}_{uij}) - \lambda_\Theta\|\Theta\|^2 \\
&= \sum_{u,i,j} \log\sigma(\hat{x}_{ui} - \hat{x}_{uj}) - \lambda_\Theta\|\Theta\|^2 \\
&= \sum_{u,i,j} \log\sigma(\mathbf{p}_u\mathbf{q}_i^T - \mathbf{p}_u\mathbf{q}_j^T) - \lambda_\Theta\|\mathbf{p}_u\|^2 - \lambda_\Theta\|\mathbf{q}_i\|^2 - \lambda_\Theta\|\mathbf{q}_j\|^2 \\
&= \sum_{u,i,j} \log\Big(\frac{1}{1+e^{-(\mathbf{p}_u\mathbf{q}_i^T - \mathbf{p}_u\mathbf{q}_j^T)}}\Big) - \lambda_\Theta\|\mathbf{p}_u\|^2 - \lambda_\Theta\|\mathbf{q}_i\|^2 - \lambda_\Theta\|\mathbf{q}_j\|^2
\end{aligned}
$$

BPR Learning Algorithm

파라미터를 업데이트 하는 방법에 대해 살펴보자.

$$
\Theta \leftarrow \Theta - \alpha\frac{\partial \log p(\Theta| >_u)}{\partial\Theta}
$$

각 파라미터에 대한 gradient는 다음과 같다.

- $$\frac{\partial \log p(\Theta|>_u)}{\partial p_u} = \frac{\partial \log \sigma(\hat{x}_{uij})}{\partial p_u} - \frac{\partial}{\partial p_u}(\lambda_\Theta \|p_u\|^2)$$
$$= \frac{\partial \log \sigma(\hat{x}_{uij})}{\partial \hat{x}_{uij}} \frac{\partial \hat{x}_{uij}}{\partial p_u} - \frac{\partial}{\partial p_u}(\lambda_\Theta \|p_u\|^2)$$
$$= \frac{1}{1+e^{(p_u q_i^T - p_u q_j^T)}}(q_i - q_j) - 2\lambda_{p_u} p_u$$

- $$\frac{\partial \log p(\Theta|>_u)}{\partial q_i} = \frac{\partial \log \sigma(\hat{x}_{uij})}{\partial q_i} - \frac{\partial}{\partial q_i}(\lambda_\Theta \|q_i\|^2)$$
$$= \frac{\partial \log \sigma(\hat{x}_{uij})}{\partial \hat{x}_{uij}} \frac{\partial \hat{x}_{uij}}{\partial q_i} - \frac{\partial}{\partial q_i}(\lambda_\Theta \|q_i\|^2)$$
$$= \frac{1}{1+e^{(p_u q_i^T - p_u q_j^T)}}(p_u) - 2\lambda_{q_i} q_i$$

- $$\frac{\partial \log p(\Theta|>_u)}{\partial q_j} = \frac{\partial \log \sigma(\hat{x}_{uij})}{\partial q_j} - \frac{\partial}{\partial q_j}(\lambda_\Theta \|q_j\|^2)$$
$$= \frac{\partial \log \sigma(\hat{x}_{uij})}{\partial \hat{x}_{uij}} \frac{\partial \hat{x}_{uij}}{\partial q_j} - \frac{\partial}{\partial q_j}(\lambda_\Theta \|q_j\|^2)$$
$$= \frac{1}{1+e^{(p_u q_i^T - p_u q_j^T)}}(-p_u) - 2\lambda_{q_j} q_j$$

full gradient descent 가 아닌 stochastic gradient descent 방법을 사용하는데,

▼ Full Gradient Descent

- user 가 선호하는 i 집단과, 비 선호하는 j 집단의 비 대칭성 문제
- i 집단이 j 집단보다 개수가 적다. 따라서 optimization function에 i 집단을 포함한 항이 많아지게 되고 i 가 기울기를 지배 하게 된다.

▼ Stochastic Gradient Descent

- bootstrap sampling 방법을 통해 (u, i, j)를 randomly하게 선택하므로 i 집단과 j 집단의 개수에 대한 비대칭성 문제가 해결될 수 있다.

Adaptive K Nearest Neighbor

새로운 아이템 i 에 대한 유저 점수 x_ui는 아래와 같다

I_u^+가 user가 과거에 좋아한 item 집합

$$\hat{x}_{ui} = \sum_{l \in I_u^+ \wedge l \neq i} c_{il}$$

c_{il}은 item i와 item l 의 유사도

optimization function과 gradient는 다음과 같다.

$$\begin{aligned}\log p(\Theta|>_u) &= \sum_{u,i,j} \log \sigma(\hat{x}_{uij}) - \lambda_\Theta \|\Theta\|^2 \\ &= \sum_{u,i,j} \log\left(\frac{1}{1+e^{-(\sum c_{il} - \sum c_{jl})}}\right) - \lambda_\Theta \|c_{il}\|^2 - \lambda_\Theta \|c_{jl}\|^2\end{aligned}$$

- $\frac{\partial \log p(\Theta|>_u)}{\partial c_{il}} = \frac{1}{1+e^{(\sum c_{il} - \sum c_{jl})}}(1) - 2\lambda_+ c_{il}$
- $\frac{\partial \log p(\Theta|>_u)}{\partial c_{jl}} = \frac{1}{1+e^{(\sum c_{il} - \sum c_{jl})}}(-1) - 2\lambda_- c_{jl}$