Recommendation Systems Overview

Qianchao Liu
Oct. 2017

Outline

- Introduction
- Recommendation System
  - Collaborative Filtering
  - Feature based Ranking Model
    - Pointwise (CTR Models)
      - Linear Model
        - GLMMs, FFM
      - Tree Model
        - GBDT
    - Pairwise / Listwise
      - RankNet
      - LambdaMART
      - ListNet

Introduction

- Background
  - Netflix: 80% of the movies watched are recommended
  - Google News: recommendation generate 38% more click-through
  - Amazon: 30% page views from recommendations

- Model Classification
  - Collaborative Filtering
  - Feature based Recommendation(Learning to rank)
    - Pointwise
    - Pairwise/Listwise

Ref: Two Decades of Recommender Systems at Amazon.com(Brent Smith, Greg Linden 2017)

Collaborative Filtering

Collaborative Filtering

- *What is CF:* recommendations are based on user's <u>past</u> behaviour
  - User based CF
    - Represent each User as a *N*-dimensional vector
    - Find top k similar Users for each target user
    - select recommendations from the similar users' items (count/score)

  - Item based CF (2003 Amazon.com)
    - Generate Item to Item co-rated matrix and compute the similarity
    - Predict ratings for the target item

$$\cos(\vec{w}_c, \vec{w}_s) = \frac{\vec{w}_c \cdot \vec{w}_s}{\|\vec{w}_c\|_2 \times \|\vec{w}_s\|_2}$$

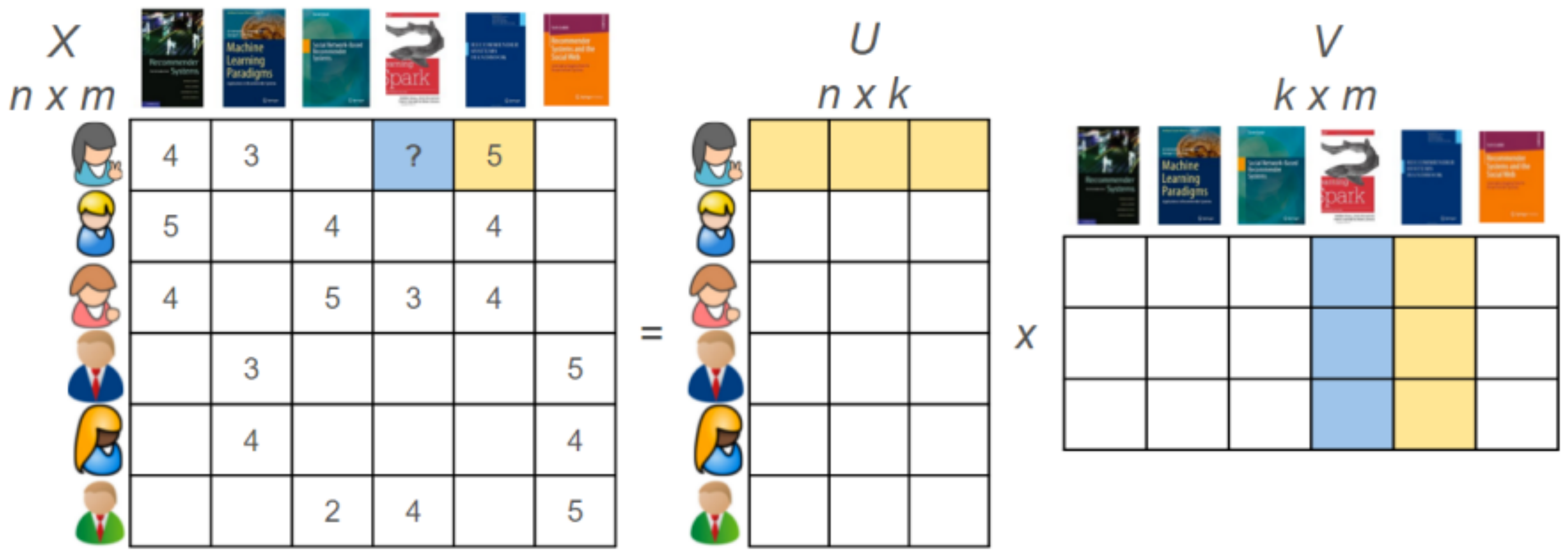
Cosine Similarity

$$sim(u,v) = \frac{\sum_{i \in I_{uv}} (y_{u,i} - \hat{y}_u)(y_{v,i} - \hat{y}_v)}{\sqrt{\sum_{i \in I_{uv}} (y_{u,i} - \hat{y}_u)^2 \sum_{i \in I_{uv}} (y_{v,i} - \hat{y}_v)^2}}$$

Pearson Correlation

Collaborative Filtering

- Matrix Factorization (2008 Netflix Competition)
  - learning: Gradient Descent with Square Loss



$$\hat{y}_{ij} = U_i^T V_j$$

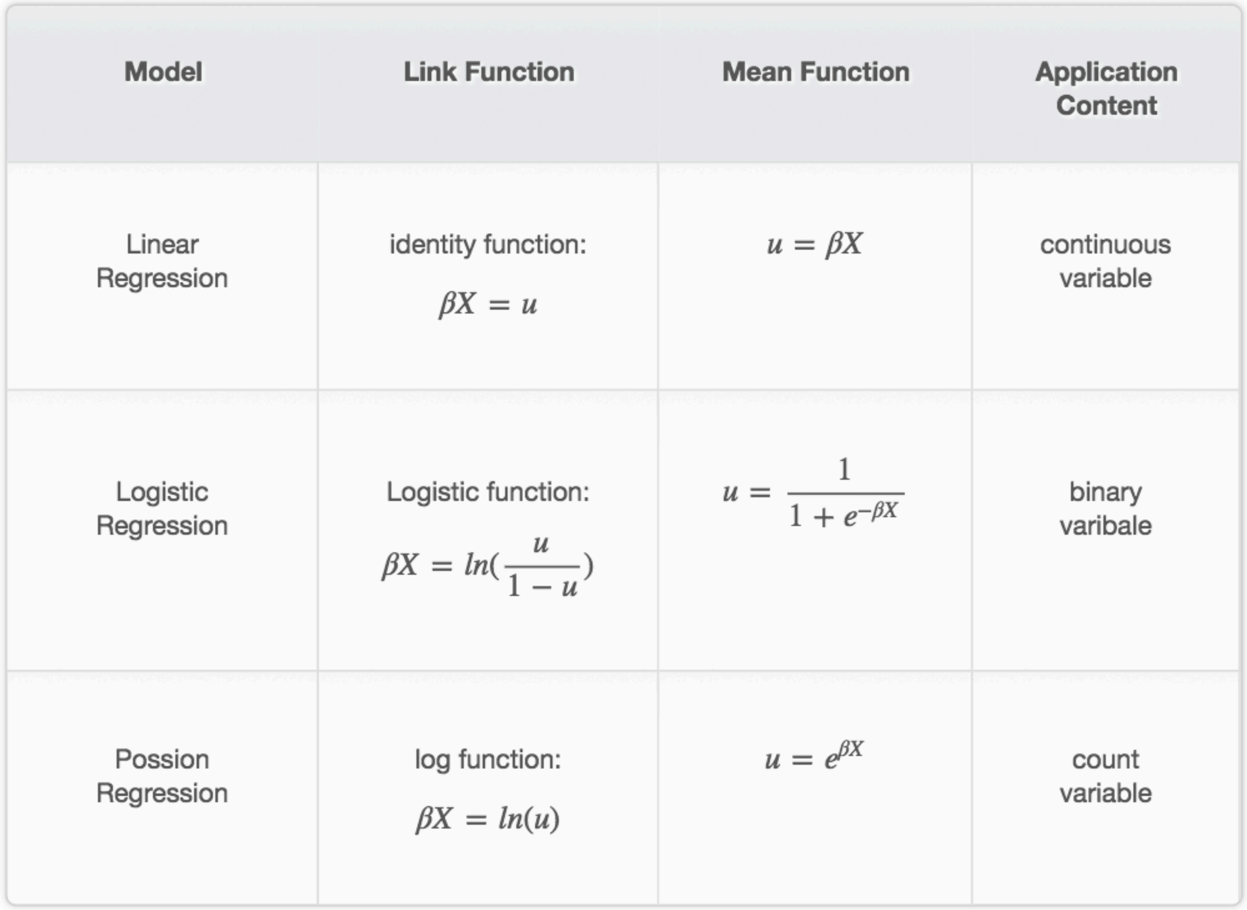
$$e_{ij} = \frac{1}{2}(y_{ij} - \hat{y}_{ij})^2 + \frac{\lambda}{2}\sum_{p=1}^{P}(u_{pi}^2 + v_{pj}^2)$$

Pointwise Feature-based Recommendation System

Generalized Linear Model(GLM)

- Starting with GLM
    - X: Feature Vector
    - β: Coefficient Vector
    - $g(\cdot)$: Link Function

$$g(E[y]) = \beta X$$



Model	Link Function	Mean Function	Application Content
Linear Regression	identity function: $\beta X = u$	$u = \beta X$	continuous variable
Logistic Regression	Logistic function: $\beta X = ln(\frac{u}{1-u})$	$u = \frac{1}{1+e^{-\beta X}}$	binary varibale
Possion Regression	log function: $\beta X = ln(u)$	$u = e^{\beta X}$	count variable

Generalized Linear Mixed Models(GLMMs): GLMMs

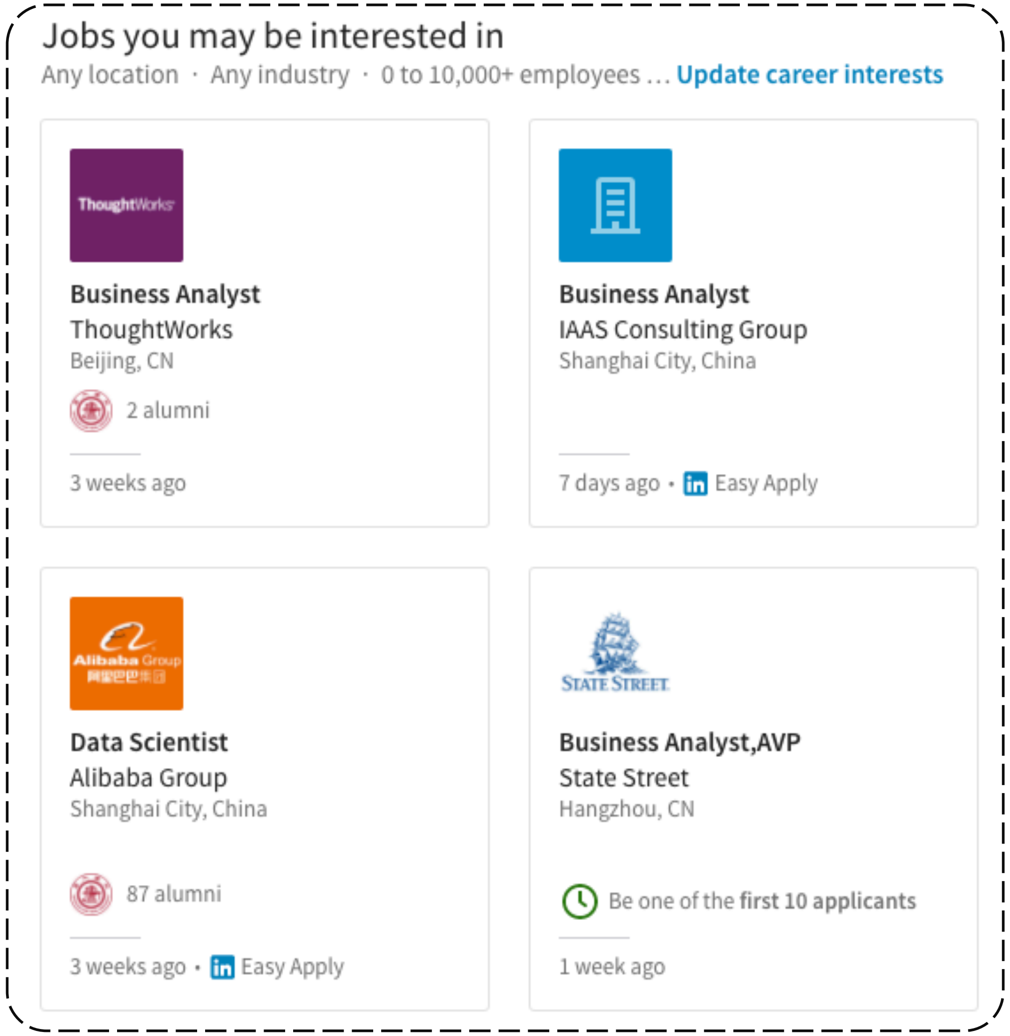
- GLMMs = Fixed Effect (GLM) + Random Effect
- GLMMs allows the response to include information at the group level

$$g(E[y]) = \underbrace{\beta X}_{\text{Fixed}} + \underbrace{uZ}_{\text{u} \sim \mathcal{N}(0, \sigma^2 I)} + e$$

- Multi-task Learning(Parameter based Transfer Learning)
  - For a user with many responses, it's able to accurately estimate personal coefficient
  - For a user doesn't have much past data, u is close to 0 and model will fall back to the global fixed effect

Generalized Linear Mixed Models(GLMMs): GLMMs

- GLMix: Generalized Linear Mixed Models For Large-Scale Response Prediction 2015
  - Application in *Linkedin* (with 400 million members)



Jobs you may be interested in

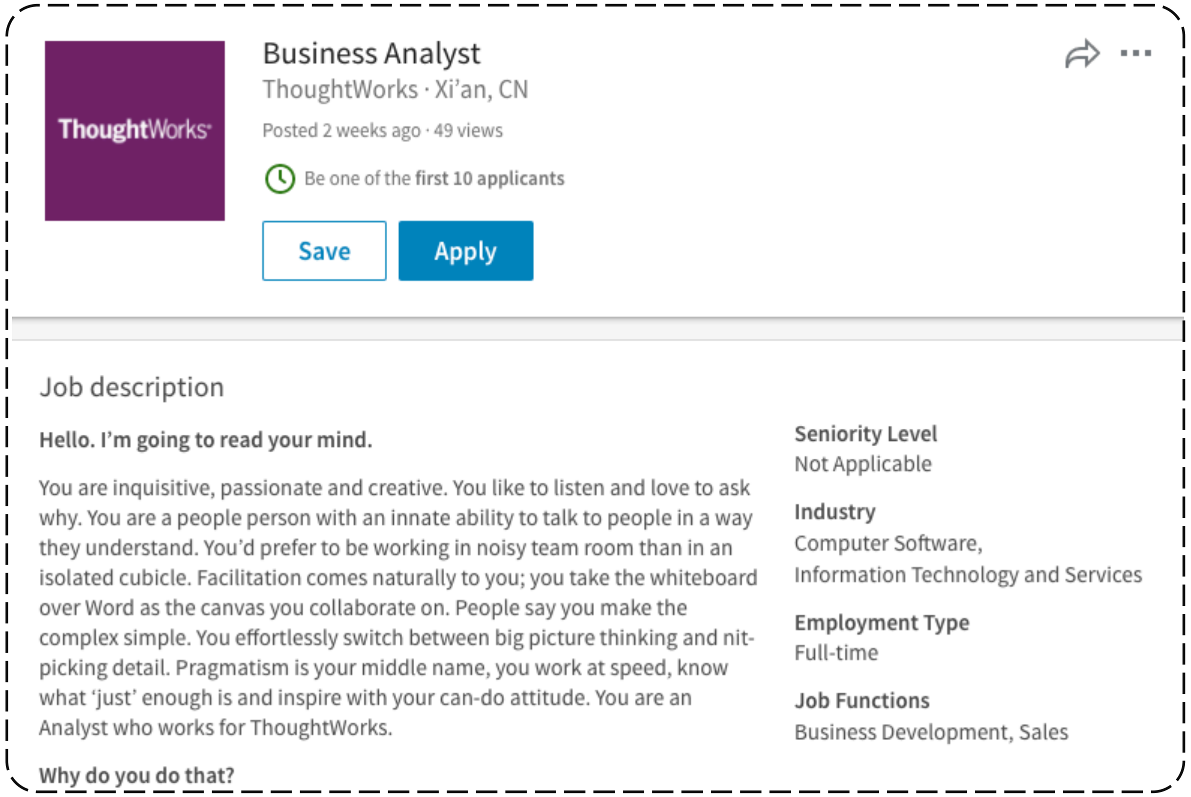
Any location · Any industry · 0 to 10,000+ employees … **Update career interests**

Business Analyst
ThoughtWorks
Beijing, CN
2 alumni
3 weeks ago

Business Analyst
IAAS Consulting Group
Shanghai City, China
7 days ago · Easy Apply

Data Scientist
Alibaba Group
Shanghai City, China
87 alumni
3 weeks ago · Easy Apply

Business Analyst,AVP
State Street
Hangzhou, CN
Be one of the **first 10 applicants**
1 week ago



Business Analyst
ThoughtWorks · Xi'an, CN
Posted 2 weeks ago · 49 views
Be one of the **first 10 applicants**

Save Apply

Job description

Hello. I'm going to read your mind.

You are inquisitive, passionate and creative. You like to listen and love to ask why. You are a people person with an innate ability to talk to people in a way they understand. You'd prefer to be working in noisy team room than in an isolated cubicle. Facilitation comes naturally to you; you take the whiteboard over Word as the canvas you collaborate on. People say you make the complex simple. You effortlessly switch between big picture thinking and nit-picking detail. Pragmatism is your middle name, you work at speed, know what 'just' enough is and inspire with your can-do attitude. You are an Analyst who works for ThoughtWorks.

Why do you do that?

Seniority Level
Not Applicable

Industry
Computer Software,
Information Technology and Services

Employment Type
Full-time

Job Functions
Business Development, Sales

Generalized Linear Mixed Models(GLMMs): GLMMs

- GLMix: Generalized Linear Mixed Models For Large-Scale Response Prediction 2015
  - Application in *Linkedin* (with 400 million members)

$$g(E[y]) = \overset{1}{\boxed{wX_{ij}}} + \overset{2}{\boxed{\alpha_i X_j}} + \overset{3}{\boxed{\beta_j X_i}}$$

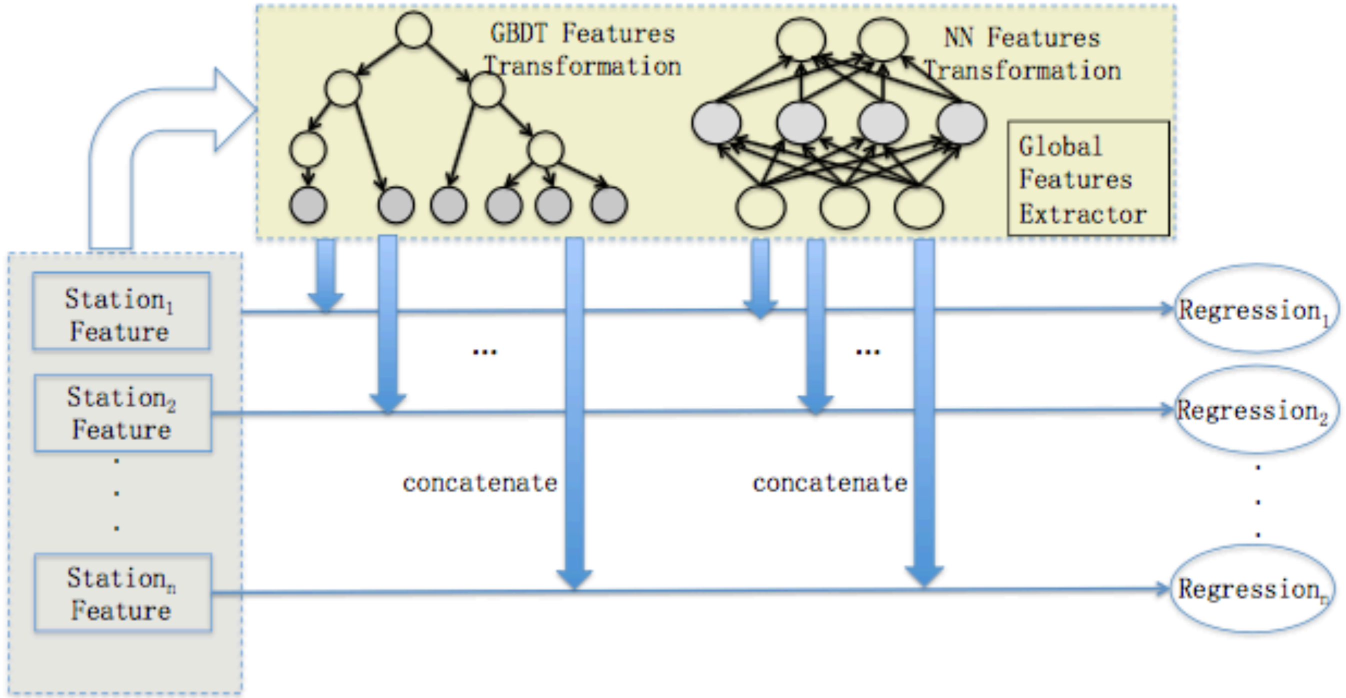
- Global Model: Linear combination between user and job profile
- Per-user Model: $x_{userid}x_{job\_title}$ if a member has applied to a job with title “Data Scientist”
- Per-item Model: $x_{jobid}x_{user\_years}$ if a job get an apply from a user with “4 year experiences”

Model	LR-Cosine	LR-Full	MF	GLMix-Member	GLMix-Job	GLMix-Full	GLMix-Full+MF
AUC	0.664	0.723	0.772	0.780	0.758	0.799	**0.801**

Offline model performance for LinkedIn Job Recommendation data.

Related Work: Capturing Features on Individuals Level

- Improving Demand Prediction in Bike Sharing System by Learning Global Features (2016)
  - Regression by individuals level + Global Features(GBDT, NNs)



The station-centric model with global feature transformation framework. The decision tree and neural network are trained on the entire dataset. Then the global features (gray) are transformed by the models from the station-centric features and concatenated to the original features. The regression model will be trained for each station individually. In this way, we can integrate the station-centric features and global features.

FM & FFM (Factorization Machine & Field-aware FM)

- Introduce Feature Combination to Linear Model

$$y(X) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} w_{ij} x_i x_j$$

- Using latent vector

$$y(X) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} < v_i, v_j > x_i x_j$$

FM

- Extending feature to field-feature

$$y(X) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} < v_{i,f_j}, v_{j,f_i} > x_i x_j$$

FFM

Introduce to GBDT: Classification and Regression Tree(CART)

- Definition
  - Decision rules same as in decision tree
  - Contains continuous/categorical value in each leaf for regression/ classification problem
- How to build a tree?
  - Branching criteria (decision stump/subset for different feature types)
    - Leaf value
      - classification: Majority (regression: Mean)
    - Evaluated by purifying
      - classification: Gini $1 - \sum_{k=1}^{K} \left(\frac{\sum_{n=1}^{N} [\![y_n = k]\!]}{N} \right)^2$ (regression: SSE)
  - Termination criteria
    - All y_n the same / Height limitation
    - Pruning when growing or after fully-grown by $Error(G) + \lambda\Omega(G)$

GBDT Model

- Functional Gradient Descent + Tree Model = Gradient Boosting Decision Tree

- Gradient Descent

$$f(x+x') \approx f(x) + f'(x)x' \longrightarrow \theta_{k+1} = \theta_k - \eta g_k$$

Taylor' formula

Gradient Descent

- Functional Gradient Descent

$$\begin{aligned}
&F_0(x) = f_0(x)\\
&F_1(x) = f_0(x) + f_1(x) = F_0(x) + f_1(x)\\
&F_2(x) = f_0(x) + f_1(x) + f_2(x) = F_1(x) + f_2(x)\\
&\ldots\\
&F_m(x) = f_0(x) + \cdots + f_m(x) = F_{m-1}(x) + f_m(x)
\end{aligned}
\longrightarrow$$

$$\begin{aligned}
L\big(y, F_m(x)\big) &= L\big(y, F_{m-1}(x) + f_m(x)\big)\\
&\approx L\big(y, F_{m-1}(x)\big) + \frac{\delta L\big(y, F_{m-1}(x)\big)}{\delta F_{m-1}(x)} f_m(x)
\end{aligned}$$

$$\therefore -g_m f_m(x) < 0$$

$$\therefore \eta f_m(x) = g_m(x)$$

GBDT Model

- Gradient Boosting Algorithm

$$g_m = -\left[\frac{\delta L\left(y_i,\ F(x)\right)}{\delta F(x)}\right]_{F(x)=F_{m-1}(x)}$$

gradient residual

$$\sum_{i=1}^{N}(g_{im}-h_m(x_i))^2$$

week learner

$$\gamma_m = argmin_\gamma\, L(y_i, F_{m-1}(x) + \gamma h_m(x))$$

step length

A $F_m(x) = F_{m-1}(x) + \gamma_m h_m(\mathrm{x})$

- Gradient Boosting Decision Tree
  - Compute gradient residual g_m and use tree model to fit the residual
  - Compute γ_m (line search step)to re-estimate the parameters at the leaves of the tree

$$\gamma_{jm} = argmin_\gamma \sum_{x_i \in R_{jm}} L(y_i, F_{m-1}(x_i) + \gamma)$$

$$F_m(x) = F_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$$

Relationship between GBDT and XGBoost

- *XGBoost* (Extreme Gradient Boosting) is an implement of gradient boosting using C++

- Model Difference between GBDT and XGBoost
  - Use second order Taylor approximation
  - Regularization of XGBoost: $\gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$
  - Column subsampling

- Engineering Contribution
  - Support C++, Python, Java, R, etc.
  - Distributed

Summary of Pointwise Approach

- Pointwise Approach
  - GLM
    - GLMMs: Individual Level
    - FFM: Feature Combination
  - Tree and Ensemble Model
    - GBDT: Feature Combination and Regularization

- However, they doesn’t take into account the location of each alternative.

Pairwise/Listwise Recommendation System

Pairwise Ranking Model: RankNet

- *RankNet (Burges et al. 2005)* is used by Microsoft Bing Search
- Evaluating target probability through scoring function$f(x; w)$

$$\hat{P}_{i,j}(f) = \frac{1}{1 + e^{-\sigma\left(f(x_i) - f(x_j)\right)}}$$

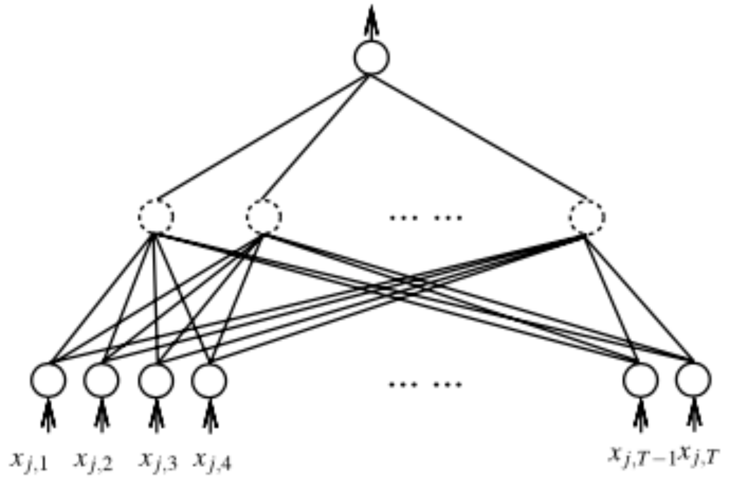
- Using Cross Entropy loss function

$$C_{ij} = -P_{i,j}\log\hat{P}_{i,j}(f) - \left(1 - P_{i,j}\right)\log(1 - \hat{P}_{i,j}(f))$$

$$C_{ij} = -P_{i,j}\log\hat{P}_{i,j}(f) - \left(1 - P_{i,j}\right)\log(1 - \hat{P}_{i,j}(f))$$

- Using two layer neural nets as scoring function

Scoring Function: $s = f(x; w)$



From RankNet to LambdaMART

- Dive into RankNet
  - Single pair

$$\frac{\delta C_{ij}}{\delta w} = \frac{\delta C_{ij}}{\delta s_i}\frac{\delta s_i}{\delta w} + \frac{\delta C_{ij}}{\delta s_j}\frac{\delta s_j}{\delta w} \qquad \text{where } \frac{\delta C_{ij}}{\delta s_i} = -\frac{\delta C_{ij}}{\delta s_j} = -\frac{\sigma}{1+e^{\sigma(s_i-s_j)}} \overset{\text{def}}{=} \lambda_{ij}$$

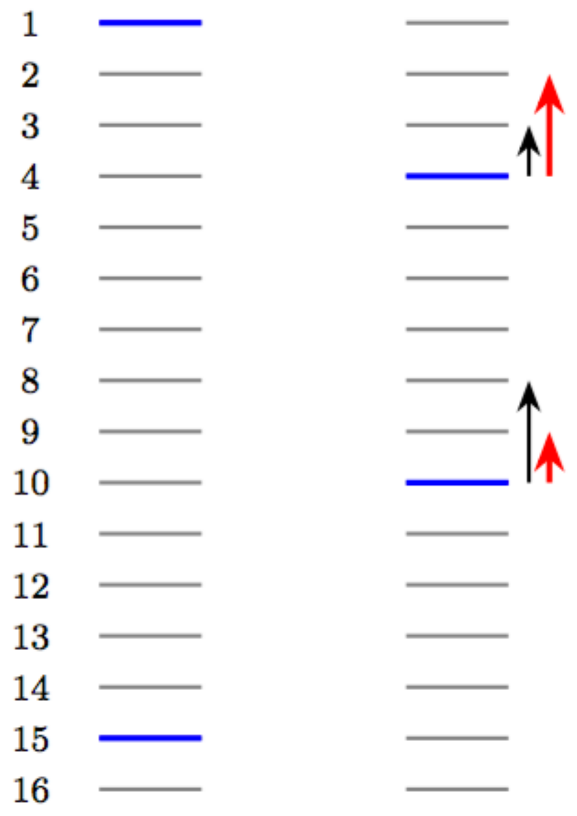
$$= \lambda_{ij}\left(\frac{\delta s_i}{\delta w} - \frac{\delta s_j}{\delta w}\right)$$

  - mini-batch

$$\frac{\delta C}{\delta w} = \sum_{\{i,j\}\in I} \lambda_{ij}\left(\frac{\delta s_i}{\delta w} - \frac{\delta s_j}{\delta w}\right) = \Big(\sum_{j:\{i,j\}\in I} \lambda_{ij} - \sum_{j:\{j,i\}\in I} \lambda_{ji}\Big)\frac{\delta s_i}{\delta w}$$

  - LambdaRank

$$\lambda'_{ij} = \lambda_{ij}|\Delta(NDCG)|$$



From RankNet to LambdaMART

- Ranking performance evaluation: Normalized Discounted Cumulative Gain(NDCG)

$$N = Z_n \sum_{j=1}^{n} \frac{\overset{\dashrightarrow \text{Info gain}}{2^{r(j)} - 1}}{\underset{\dashrightarrow \text{Discount}}{\log(j + 1)}}$$

- LambdaMART: Combining MART(Multivariate Adaptive Regression Trees) and LambdaRank

- LambdaMART won 1st at 2010 *Yahoo! Learning to Rank Challenge*

Other Listwise Ranking Model

- *ListRank(2007):* using Plackett-Luce dist to model the uncertainty of a rank

$$P(\pi|s) = \prod_{j=1}^{m} \frac{\exp(s_j)}{\sum_{u=j}^{m} \exp(s_u)}$$

evaluate the probability

$$-\sum_{i}\sum_{\boldsymbol{\pi}} p(\boldsymbol{\pi}|\mathbf{y}_i) \log p(\boldsymbol{\pi}|\mathbf{s}_i)$$

loss function

- It’s intractable since π is over $\mathrm{m}!$. We can consider permutations of top K: $\mathrm{A}_{\mathrm{m}}^{\mathrm{k}}$

- In the special case where only one document is relevant——MNL.

$$P(\pi|s) = \frac{\exp(s_j)}{\sum_{u=1}^{m} \exp(s_u)}$$

$$-\sum_{i}\sum_{\boldsymbol{\pi}} p(\boldsymbol{\pi}|\mathbf{y}_i) \log p(\boldsymbol{\pi}|\mathbf{s}_i)$$