Learning User Preferences for Sets of Objects

Marie desJardins, Eric Eaton, Kiri L. Wagstaff

October 24, 2006

Presented By:
Yisong Yue

Introduction

The Goal:

- finding an optimal *set* of items

The Problem:

- can't just pick the k individual best items (the deepest set)
- optimal set requires diversity

The Solution

- DD-PREF!
- models trade-off between diversity and depth
- can learn this model given examples of good sets

Portfolio Effect

Definition

Portfolio Effect : the *Portfolio Effect* occurs when the valuation of a set is not equal to the sum of its component item valuations.

Examples:

- choosing foods representative of a healthy diet
- recommending a list of songs or TV shows

DD-PREF

DD-PREF (Diversity and Depth PREFerences) - a language for specifying diversity and depth of individual features.

Represented as tuples

$$P = \langle \mathbf{q}, \mathbf{d}, \mathbf{w}, \alpha \rangle$$

For each feature $f \in V_f$,

- $q_f \ : \ V_f \rightarrow [0,1]$ computes the depth value of feature f.
- $d_f \in [0,1]$ denotes the desired diversity of feature f.
- $w_f \in [0,1]$ denotes the relative importance of feature f.
- α denotes tradeoff between diversity and depth.

Computing Depth

Given set of items s, and DD-PREF model P, depth computed as

$$\mathcal{V}_{dep}(s|P) = \sum_{f} \left(P.w_f \frac{1}{|s|} \sum_{x \in s} P.q_f(x^f) \right),$$

where x^f is the value of for feature f of item x.

Computing Skew

Let $Y =< y_{min}, \ldots, y_{max} >$ be a list of k sorted values, then *skew* computed as

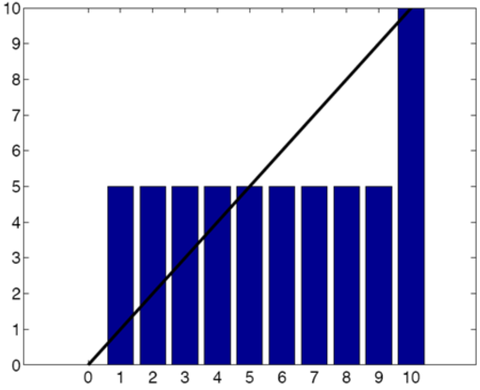
$$\sigma(Y) = \frac{\sum_{i=1}^{k}(y_i - y'_i)^2}{M(Y)},$$

where y'_i is the ith value in an evenly distributed list:

$$y'_i = y_{min} + (y_{max} - y_{min})\frac{i-1}{k-1},$$

and $M(Y)$ is a normalizing factor.

Computing Skew



Skew: 0.21   Diversity: 0.79

Computing Diversity

Given set of items s, and model P, diversity computed as

$$\mathcal{V}_{div}(s|P) = \sum_{f} P.w_f (1 - (P.d_f - \text{div}_f(s))^2)$$

where

$$\text{div}_f = 1 - \sigma(sort(< x_i^f \mid i = 1, \ldots, k >)).$$

Computing Value

The value of a set s given a model P is then computed as a weighted average of depth and diversity:

$$\mathcal{V}_P(s) = (1 - P.\alpha)\mathcal{V}_{dep}(s|P) + P.\alpha\mathcal{V}_{div}(s|P).$$

Finding Best Set

Finding the best set of items (computing arg max):

- **Basic-Greedy** - iteratively select best item for $\mathcal{V}_P$. Requires a non-empty base set due to how div_f works.
- **Wrapper-Greedy** - try Basic-Greedy with each item as base set, pick best.
- **Lookahead-Greedy** - searches over all subsets of size 2 for best base set.
- **Exhaustive** - only works for small datasets.

Learning a DD-PREF Model

Material present thus far due to original paper on DD-PREF (desJardins & Wagstaff, 2005).

Main contribution of (desJardins et al., 2006) is method for learning a DD-PREF model given training sets.

- Learning depth parameters, q_f.
- Learning diversity parameters, d_f.
- Learning relative importance of features, w_f.
- Learning α not addressed, $\alpha = 0.5$.

Learning Depth Parameters

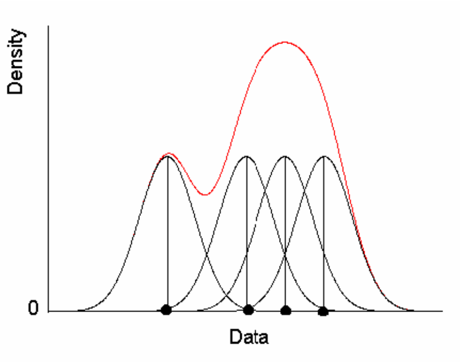
Use kernel density estimation (KDE):

$$q_f(x) = \frac{P_{KDE}(x)}{\max_{x'} P_{KDE}(x')},$$

where $P_{KDE}(x)$ is the density estimate of x from a mixture of Gaussian distributions centered at each data point.

Gaussian width parameter σ selected automatically (?)

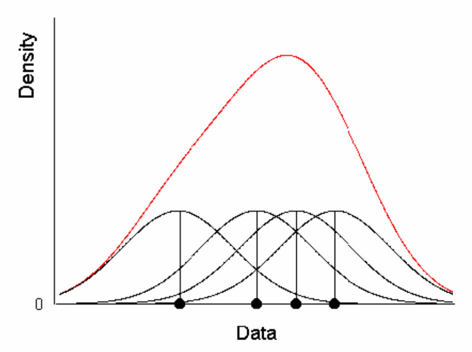
Kernel Density Estimate



P_{KDE} with small σ.

Image from Technical Report AMC Technical Brief No. 4

Kernel Density Estimate



P_{KDE} with large σ.

Image from Technical Report AMC Technical Brief No. 4

Learning Diversity Parameters

Use maximum *a posteriori* (MAP) estimate by calculating the mean observed diversity:

$$d_f = \frac{1}{|S|} \sum_{s_i \in S} \mathrm{div}_f(s_i)$$

One interpretation: recall that

$$\mathcal{V}_{div,f}(s|P) = P.w_f(1 - (P.d_f - \mathrm{div}_f(s))^2).$$

To compute MAP estimate, minimize diversity "loss"

$$d_f = \arg\min_{d'} \sum_{s \in S} (d - \mathrm{div}_f(s))^2,$$

take the derivative, blah blah blah, the result is the mean.

Learning Relative Weights

Let P denote true model. Let Q denote model we train. Weights $Q.\mathbf{w}$ computed via gradient descent to minimize

$$\sum_{s \in S} (\mathcal{V}_P(s) - \mathcal{V}_Q(s))^2 = \sum_{s \in S} (1 - \mathcal{V}_Q(s))^2.$$

Datasets

- Artificial Blocks
- Classical Music

Metrics

- Retrieval Similarity
- Functional Similarity
- Preference Precision

Artificial Blocks Preference Models

- Castle - large and small, similar colors, few sides.
- Child - multi-colored, medium-sized, few sides, close together.
- Mosaic - similar colors, smaller size, few sides.
- Tower - large, similar-sized, uniform color.

Artificial Blocks Dataset

Blocks are defined with attributes

- size - real valued in [0,100]
- color - integer in [0,6]
- sides - integer in [3,20]
- bin (location) - integer in [0,100]

Model for Child Blocks

Feature	q_f	d_f	w_f
size	[10,50]	1.0	1.0
color	1.0	1.0	0.8
sides	[3,6]	1.0	0.8
bin	1.0	0.2	0.4

Music Dataset

Collected 2353 classical music songs, represented by 28 features:

Feature	Type/Units
Composition date	Year
Composer birthdate	Year
Duration	Minutes
Tempo	Beats per minute
Bark spectrum	Loudness at 24 frequencies

Training examples are randomly selected playlists.

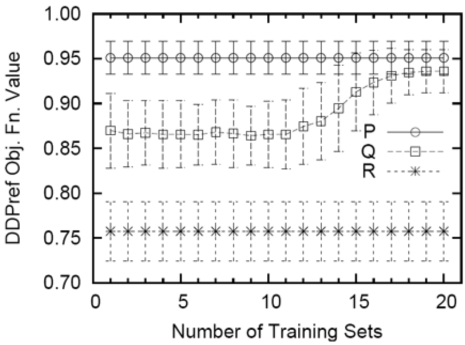
Methodology

- split data into f disjoint folds
- given true P, use wrapper-greedy to compute argmax set for each fold
- use LOO CV to train on all except one argmax set
- test on held-out data

Note that true P not available for Music dataset, so P learned from random playlist.

Retrieval Similarity

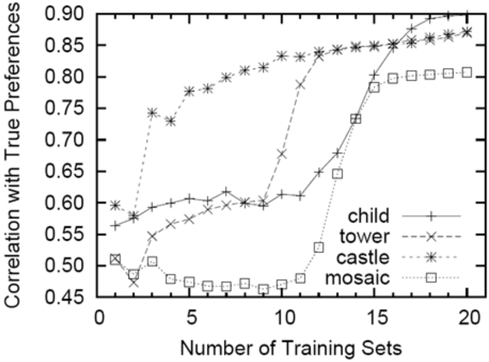
Retrieval Similary Results for Child Blocks Experiments



Comparison of argmax from true P, model Q, and random R.

Functional Similarity

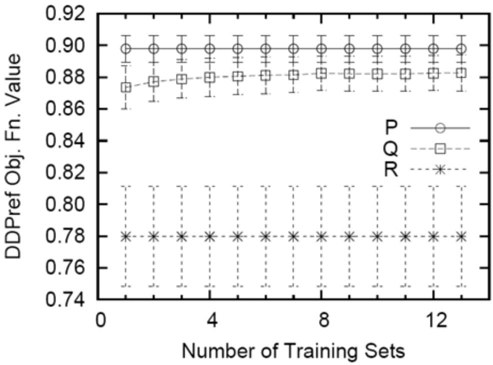
Functional Similary Results for Artificial Blocks Experiements



Correlation of true P and learned Q over training sets.

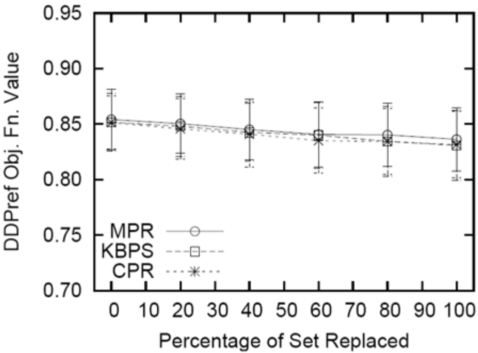
Retrieval Similarity

Retrieval Similary Results for Music Experiments



Comparison of argmax from true P, model Q, and random R.

Preference Precision Results for Music Experiments



Analysis of how fast objective function decays with random replacements to argmax set.

Conclusion

Conclusions & Future Work

- given appropriate features, can model many set-preference learning problems
- addresses portfolio effect by modeling tradeoff of depth and diversity across all features
- tradeoff parameter α unlearned
- plan on examining over more heterogeneous music collections