CS600.475: Proposal
Machine Learning Systems: Personalized Web Search

Biman Gujral (bgujral1, bgujral1@jhu.edu) Rujuta Deshpande (rdeshpa3,rdeshpa3@jhu.edu)
October 31, 2014

1 Abstract

This project aims to improve web search results through personalization. It involves re-ranking of results on the basis of user profiles comprising their past searches and actions.
The features used are based upon the short term and long term user search history. Short term history is the user's search queries and actions(dwell time for URL and clicks on URL) in the same search session. Long term refers to a user's search behavior observed over multiple sessions over a long term period.This problem lies in the Learning to Rank class of problems.

2 Methods

2.1 Learning and Prediction

The problem of re-ranking web search results to obtain personalized results falls under Learning to Rank which involves algorithms to generate rankings for information retrieval based on rankings in training data. These algorithms are usually implemented using either Pointwise, Pairwise or Listwise approach. We will start by implementing LambdaMART, a listwise approach, as it has been observed to perform the best in the paper that we are initially following.

2.2 Evaluation

The evaluation metric used is NDCG (Normalized Discounted Cumulative Gain). It is a metric between 0 and 1 that evaluates the ranking order. It is given by:

$$NDCG_k = \frac{DCG_k}{IDCG_k}$$

where k denotes documents uptil rank k and DCG is Discounted Cumulative Gain, given by:

$$DCG_k = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{log_2(i+1)}$$

where rel is the actual relevance of the document provided by labels and i is the rank given by the algorithm. Thus, a highly relevant document ranked later in the list will result in penalization and a low DCG. The Ideal DCG or IDCG gives the best ranking in accordance with the relevance values. Therefore, NDCG = 1 when the ideal ranking is obtained.

3 Resources

3.1 Data

Our project idea is obtained from a Kaggle competition. Hence, data is taken from Kaggle. It consists of web search logs of Yandex, a Russian search engine, collected over a period of 30 days. The first 27 days form the train data and the remaining 3 days form the test data.

The training data is a stream of logs. Each log line includes a SESSION ID, a type of record - which indicates if it is information about a query(Q), a click(C) or meta data(M). Based on the type of record, there are other fields like *UserID, QueryID and terms, URLID-DomainID and SERPID*. The field *TimePassed*, indicates the time that has passed since sessions start until the click was made which helps compute the dwell time.

3.2 Tools

We plan to use Python for algorithm implementation and scikit-learn for any machine learning libraries we may need. We may also use a database to work with the large datasets and store our learned parameters - SQLite.

4 Milestones

4.1 Must achieve

Since NDCG is used as evaluation metrics, we aim to achieve ordering of the URLs sorted according to decreasing relevance for a particular user. This would be reflected by an NDCG score close to 1. We will attempt to achieve this using the LambdaMART algorithm. We will use the ranklib library's Java implementation of LambdaMART for our task.

4.2 Expected to achieve

The Kaggle competition winners have experimented with various features - both user specific and global. Some of them are the non-personalized rank of the URL, the frequency of query, the position of the query within the user's session, whether the URL has been missed or skipped, the relevance of the snippet displayed in search results and the like. In total, they have used about 90 features. Many features are obtained as a conditional probability of getting a URL conditioned on some predicate. This predicate is a function of the a number of global and user specific variables.
Our aim is to experiment with a subset of these features and compare the accuracy we achieve. We would be able to determine which features influence ranking greatly and whicj don't. We also plan to use not just LambdaMART but a few other Learning to Rank algorithms - RankNet, AdaRank, Random Forests and compare their results to the ones returned by LambdaMART.

4.3 Would like to achieve

One of the papers we read, spoke about whether personalization is necessary for every query. We would like to improve upon the search results, by only personalizing when needed. Thus, at all times, we would get optimum results.

5 Final Writeup

The final writeup will disuss in detail the algorithms and experimental setup and the results will be presented. Some broad headers will be:

- Abstract and General Approach
- Related work and our approach in comparison to them
- Explanation of Feature Engineering, Algorithm, Evaluation metrics and Result

6 Bibliography

1. David Sontag, Kevyn Collins-Thompson, Paul N. Bennett, Ryen W. White, Susan Dumais, and Bodo Billerbeck. Probabilistic models for personalizing web search. In Proceedings of the fifth ACM international conference on Web search and data mining (WSDM '12).
2. Zhicheng Dou, Ruihua Song, and Ji-Rong Wen. A large-scale evaluation and analysis of personalized search strategies. In Proceedings of the 16th international conference on World Wide Web (WWW '07).
3. Learning to Rank `http://en.wikipedia.org/wiki/Learning_to_rank`
4. Dataiku's winning Solution to the problem `http://research.microsoft.com/en-us/um/people/nickcr/wscd2014/papers/wscdchallenge2014dataiku.pdf`