Personalized Web Search Re-ranking

Santanu Dey Conrad Roche

Abstract

The intent of a search is context dependent and cannot be fully (realized) by the search term used by the user for performing the search. Adding more context by using a user's search history and additional details can help us to personalize the ranking of the search output and help the search results to be more relevant for the user. Search engines should not rank the results the same for its entire user base, but must factor in and personalize the ranking based on the user.

Introduction

The source of the project is a Kaggle challenge by Yandex and is publicly available [1].

The challenge is to personalize the rankings of the URLs of a search result based on the search history of user. The search history considered would be both the long-term user search history and also the short-term current search session context history. The purpose of the re-ranking is to present the search results most relevant to the user based on the user's intent.

Related Works

Several researchers have worked on techniques to personalize web search result ranking. Many of the related works are listed on Kaggle [2].

Data Set Description

Yandex has provided datasets containing user session information - including the user queries, URLs, URL rankings and clicks. The relevance of a search result is measured by the user's dwell time on the page. If the dwell time exceeds a specific threshold, the result is considered relevant. The data contains about 160M records.

The dwell time is the time between a click and the subsequent click or query. A higher dwell time is proportional to the result relevant. A dwell time higher than a specific threshold (400 time units in this case) signifies a satisfied click.

The search data is sampled from users of a single large city and is around two years old. The top-k (k unknown) most popular queries are removed from the data. Additionally, queries with commercial intent are also removed from the data set. The training data contains 27 days of search and the test data contains 3 days of activity.

The data is provided as a user activity log - which contains session, query and click action data for all users. The data is anonymous to maintain user privacy and only contains numeric identifiers. Each type of record uses a different format adding to the challenge of extracting the data.

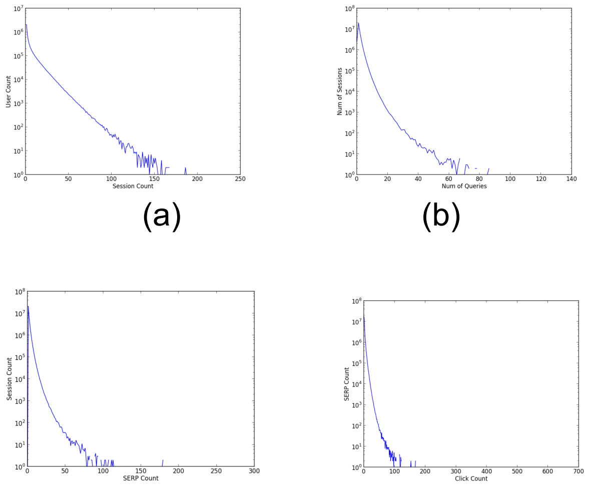
Analysis of Dataset

Basic characteristics of the dataset:

Characteristic	**ALL**	**Train**	**Test**
Days	30	27	3
Session Cnt	35,371,497	34,573,630	797,867
Unique users	5,736,333	5,659,229	797,867
Query Count	20,840,402	20,588,928	468,739
Unique terms	4,750,000	4,701,603	254,848
Unique domains	5,233,657	5,199,927	557,069
Unique URLs	69,791,340	69,134,718	3,190,886
Click Count	65,325,593	64,693,054	632,539
Total records	**167,413,039**	**164,439,537**	**2,973,502**

In order to understand that dataset, the figures below show

a) Number of sessions against the number of users
b) Number of sessions against number of queries
c) Number of sessions against the number of search engine results page (SERP)
d) Number of SERPs against the number of clicks.



(a) (b)

(c) (d)

We observed that more than half of the queries (67%) have three or less terms. Around 35% of the sessions contain more than two queries.

Methods

We used multiple machine learning techniques to determine the best ranking for the user's current search. Some of the approaches we thought of are:

- Determine users' historical engagements with respect to a domain and URLs within the domain. The goal is to represent user as a vector of interests where each element represents interest in some aspect such as Domain, URL.
- Determine the association between query and domain and URLs within the domain. Historical performance is manifestation of intrinsic quality of the domain and URL.
- Finally we would like to come up a set of feature vectors based on everything mentioned above. We define the loss function based on clicks as feedback.
- Learning to rank, we plan to use both pairwise and point wise approach with loss function modifies for ranking. For point-wise approach, we plan to employ Logistic Regression and SVM in subset ranking[7]. For pairwise ranking, we would use SVM based approach such as Ranking SVM. List-wise ranking is out of scope.
- Given that the training and test data volume is large, we would want to scale our models and algorithms to be run efficiently. Whenever applicable, data will be sampled randomly.

The ranking result is evaluated by Yandex internally using the NDCG (Normalized discounted cumulative gain) score which uses the ranking of each URL for each query against the known grade and then averaged over all the queries.

Work & Results

Training set generation:

The intuition is that historical click through rate and dwell time can predict how users are going to be

First we generate features to represent the search results vectors:

Variable Name	Description
User Domain Historical Click thru	This ratio represents historical click through rate of a use given the domain. This is orthogonal to query
User URL Historical Click thru	This ratio represents historical click-through rate of the user given the URL. This is orthogonal to query
Query Domain Historical Click thru	This ratio represents the historical click through rate of the query given the domain. This is orthogonal to users.
Query URL Historical Click thru	This ratio represents historical click-through rate of a query given the URL. This is orthogonal to users
User Domain Historical Average Dwell Time	Total time spent by users / count of clicks that ended in that domain
User URL Average Dwell Time	Total time spent by users / count of clicks that ended in that URL
Query Domain Average Dwell Time	Total time spent in the domain by all users / count of clicks from the query by all users
Query URL Average Dwell Time	Total time spent in the URL by all users / count of clicks from the query by all users

Mean Average Precision (MAP)

The training does not have any relevance labeling except click through information. So the binary relevance model is assumed as a basis. This means if a search results page has n clicks, the best ranking function will have those n URLs will be in the top. As part of point-wise approach, the clicks are modeled and predicted. Since binary relevance is assumed, Mean Average Precision is used to evaluate the different approaches. In the process broader metrics such as precision and F1 measures are looked into.

As a starter, MAP of the data is calculated to be 0.35 and it establishes baseline value of MAP.

Point wise Approach

As explained by Cossock, 2006 [7], a regression based approached is used to predict likelihood of clicks given the search result page. In this simplified approach, the relevance of a page is solely determined by whether the user clicked on the link or not. It is also expected that search engine will rank URLs based on likelihood of a page being clicked by users. Thus, if the click probability can be calculated, it can be used to rank the URLs.

There are 2 separate approaches implement point-wise ranking, first a logistic regression model is fit to predict click probability. Second, a maximum margin classifier (liblinear SVM) is used to separate clicks with non-clicks URLs. Then the distance from the separation line is converted into probability measure for ranking [8].

In both cases, the target was set to 1 or 0 based on click or non-clicks. If there were multiple clicks in the page, all of them are considered. Once labeled, each URLs/label combination becomes an independent observation within a page.

Evaluation:

When prediction is made on test set, first the URLs are sorted by click probability (probability for logistic regression and distance from separator for SVM[8]). Evaluation can be done as a typical classification situation, measuring predicted labels with actual labels for all the URLs in the page. Even though this approach is simple and straightforward, it has shortcoming in ranking in web search since no of clicks in a page is going to be limited even though the predicted labels can all be highly probable on the page. This is addressed in pair-wise approached.

In summary, point-wise approach is measured by the following metrics

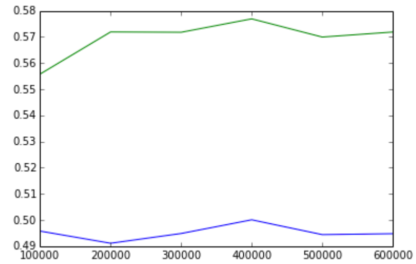
- Accuracy
- Precision/Recall
- F1 Score

Role of sample size

Given that the big data volume and high complexity in terms of space and computation time, data has been sampled for both training and testing.

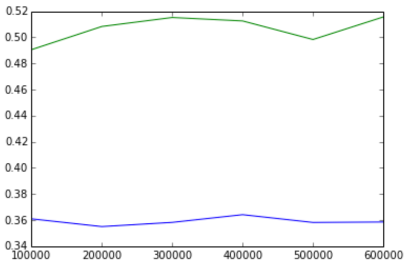
Sample Size	SVM			Logistic Regression		
	F1-Score	Precision	Recall	F1-Score	Precision	Recall
100,000	0.4957	0.7913	0.3609	0.5558	0.6411	0.4905
200,000	0.4911	0.7966	0.3550	0.5719	0.6546	0.5083
300,000	0.4948	0.8001	0.3582	0.5718	0.6438	0.5152
400,000	0.5001	0.7980	0.3641	0.5769	0.6598	0.5125
500,000	0.4944	0.7979	0.3582	0.5699	0.6665	0.4983
600,000	0.4948	0.7980	0.3585	0.5719	0.6433	0.5156

As specified below, (a) shows F1-Scores vs. sample size. Green line = Logistic regression and blue line is SVM.



(a)

As specified below, (b) shows Recall vs. sample size. Green line = Logistic regression and blue line is SVM.



(b)

Tuning hyper-parameters of the Model

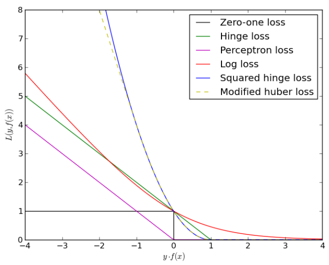
In order to find best hyper-parameters for both Logistic Regression and SVM, a grid-search has been conducted with 3-fold cross-validation. Here's the summary of the final model parameters:

Logistic Regression (Stochastic Gradient Descent)

- alpha=1.0e-05 (Constant that multiplies the regularization term)
- Loss = Modified Huber Loss (instead of Log)
- Regularization=L1

SVM (liblinear)

- Kernel= Linear
- Loss=L2 (squared hinge loss)
- Regularization=L2
- C=100.0 (Penalty of the error term)



Source : scikit-learn website

Ranking SVM was not used with default parameters.

Pairwise Approach:

Ranking SVM (SVM-rank software developed by Joachims, 2006 [9]) has been used. SVM-rank solves the quadratic program of the ROC-area optimization.

$$\min_{\mathbf{w},\xi_{ij}\geq 0} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{m}\sum_{(i,j)\in\mathcal{P}} \xi_{ij}$$

$$s.t. \quad \forall(i,j)\in\mathcal{P}\colon\ (\mathbf{w}^T\mathbf{x}_i)\geq(\mathbf{w}^T\mathbf{x}_j)+1-\xi_{ij}$$

It minimizes the number of pairs of training examples that are swapped w.r.t. their desired order.

In this approach, prediction will limited to number of clicks in actual data. This means if we have N number of clicks in the page, there will be only N predicted labels set to 1 and those are expected to be in the top. This is similar to point-wise approach except output labels are restricted to underlying data.

Evaluation

Since only output labels are restricted to match input click counts, no of clicks and no-clicks in actual data will be same as predicted data. So in this case, precision, recall and F1 score will have same values

- precision = recall = F1-score

Information Retrieval, specifically binary relevance model, measures such as Mean Average Precision (MAP) would be better suited to determine the quality of prediction in this case. MAP takes ranked position into account which is very critical given that we are expecting all the clicks to happen starting from the top of search result page.

Then based on no of clicks in the actual data of the page, test data is labeled with clicks. For instance, if page had 2 clicks, it is expected the top 2 positions in the ranked page will have clicks label set to 1. Other will be 0.

Pairwise approach needs to distinguish between pairs of URLs in the page. A simple approach would be dividing the URLs into 2 sets, one for click and other for no-click. Then pairs are composed by taking one URL from each set. This is will be functionally same as point-wise approach as that was dealing with 2 sets of URLs and trying to find a line of separation between them in process. As it turns out a majority of pages have more than 1 click and dwell time for each click can be determined unless it's the last click in the page. So we can use dwell time as a proxy of relevance and differentiate between clicks. For the last click, it can be assumed to be relevant as the user got what is needed and left it afterwards.

So all the clicks can be ranked by dwell time and non-clicks are assumed to be all equal. With this, Ranking SVM is constructed with pairs of URLs.

Results

In case of point-wise approach (without considering click restriction), SVM seems to do better in precision and Logistic regression does better in recall.

In overall comparison, based on the results, it seems like point-wise Logistic Regression (with Modified Huber Loss) can get the best result given the dataset and the problem.

Model Type	**Model**	**Mean Average Precision**	**Precision/F 1-score**	**Accuracy**
Point-wise	Logistic Regression	0.5608	0.4672	0.8230
Point-wise	SVM(liblinear)	0.4872	0.4162	0.8060
Pair-wise	Ranking SVM	0.4788	0.4084	0.8035

Training size = 1.4 million Test size = 600k

Conclusion

As far as click-through modeling is concerned, point-wise approach gives better results than pair-wise. This would be a good thing if the goal of the search engine is to generate as many clicks as possible. But in reality, often, the goal is to provide relevant information so that the user can get what is needed in the first click. But objective function should consider broader relevant signals such as dwell time and optimize the model based on it, which is what ultimately matters to the users.

References

[1] Yandex. (n.d.). Personalized web search challenge. Retrieved from Kaggle website: http://www.kaggle.com/c/yandex-personalized-web-search-challenge

[2] Yandex. (n.d.). Personalized web search challenge – related papers. Retrieved from Kaggle website: http://www.kaggle.com/c/yandex-personalized-web-search-challenge/details/related-papers

[3] Milad Shokouhi, Ryen W. White, Paul N. Bennett, Filip Radlinski: Fighting search engine amnesia: reranking repeated results. SIGIR 2013: 273-282

[4] Carsten Eickhoff, Kevyn Collins-Thompson, Paul N. Bennett, Susan T. Dumais: Personalizing atypical web search sessions. WSDM 2013: 285-294

[5] Hongning Wang, Xiaodong He1, Ming-Wei Chang, Yang Song, Ryen W. White, Wei Chu: Personalized Ranking Model Adaptation for Web Search. SIGIR 2013.

[6] Chris Manning, Pandu Nayak, Prabhakar Raghavan: CS276 lecture notes. Retrieved from Stanford website: http://cs276.stanford.edu

[7] David Cossock ,Tong Zhang : Subset Ranking Using Regression

[8] John C. Platt : Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods (1999)

[9] T. Joachims, *Training Linear SVMs in Linear Time,* Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD), 2006.