Learning to Rank Relevance Scores of Products and User Queries

Abstract

Learning to rank refers to machine learning techniques for training the model in a ranking task. Learning to rank is useful for many applications in Information Retrieval, Natural Language Processing, and Data Mining. In this project we have explored how an LETOR (Learning to Rank) is implemented on the Dataset having Categorical attributes which describes the Product Info and User queries to predict the Relevancy Scores. Query expansion is performed to expand the queries and Feature Engineering is implemented to extract the attributes required for the Regression Models.

Introduction

Ranking is one of the major issues when it comes to Information Retrieval especially when it comes to the field of text-based retrieval or document retrieval. Now given the dataset we utilize the ranking model to create a ranked list of the objects present in the data the order which is relative in nature can be approximated or called as the degree of relevance for the objects which are present in the data list. And to do this the most relevant topic is information retrieval which will be discussed in the paper.

Learning to rank when applied to the dataset for the retrieval of the correct or expected words now the dataset used for this paper contains the products in a typical home depot shop, we are here to make a ranking system based on the information provide in the corpora. In this paper we will be exploring the LETOR procedures and using different feature extraction and data relevance fields will try to extract information and see how an LETOR works as a learning quotient of machine learning.

The first phase of the paper we merge the available data into training and testing sets this allows us to use this information for machine learning afterwards. Then the second phase was to eliminate unwanted words and tokenize needed vocabulary for the ranking procedure. In the third phase using query expansion modules such as WordNet, Mutual information and we try to retrieve common or synonymous words which will help us to add term weights so that or retrieval results are accurate. In feature engineering a set of features are obtained from the text data by computing the similarity measures between query and Product. Along with this similarity measures additional features were also obtained which will be later used for training the Model. Lastly, we are using regression methods to test out the performance of our overall algorithm and results are obtained from it.

Methodology

The methodology can be understood by following the diagram in figure 1

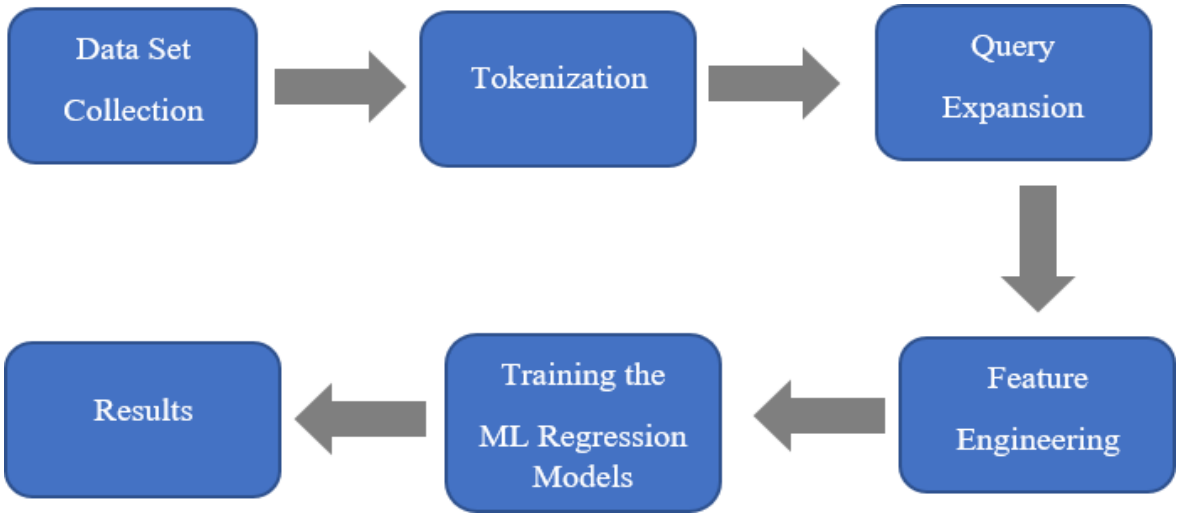

FIGURE 1

Dataset Collection and Preprocessing

In this paper We have used Home Depot dataset provided in the Kaggle competition which comprises of Train, Test, Description and Attributes files. The Train dataset has the Information about the previous user search queries and its relevance score with respect to the results (Product titles). Test Data has the New set of queries and Product titles of which relevance score has to be determined. Finally, the Descriptions have the Information of detailed descriptions of all products in the store. Since Product title is the Precise short description, we may need the detailed explanation of the product in order to find the Relevance score with respect to the Query. So, the Descriptions were added to the Train and Test data by doing the Left Join and the Instances of the data with NaN values are replaced with empty strings.

Tokenization

After Merging the data with the Supposed descriptions, we need to know about the Significant Words in about the product in order to train the Model. Since the product Information may have many repeated words which doesn't add any striking features to distinct the product with the others the need for finding Significant words gains importance. Hence Tokenization will be done on lower cases of both Product titles and Description and the subsequent results are filtered by certain methods to get the significant tokens. In the Filtering step we need to filter the commonly used words/Useless words such as "the", "a", "an", "in" which are known as Stop words. And also, we need to filter the Non-Alphabetical words like Punctuations, Comma etc. Even though all of these

filtering steps removes useless words we still want to consider the words/tokens such as "Stretching" and "Stretch" as the same. Stemming is the process of reducing inflected (or sometimes derived) words to their word stem, base or root form generally a written word form. And this is implemented by using Porter Stemming Algorithm.

A function **get_tokens_info** is defined in the project to do the tokenization and Filtering the tokens. Along with this the function will compute the other characteristics of tokens like Tokens frequency configuration of each product, Term Frequencies which is the raw frequency of the token appearance in all products, Inverse Document Frequencies which is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus, Document length of each product and the Associations of each token with others in the corpus. These values are generated separately with respect to the Product titles and Description.

Query expansion

This is a process in Information Retrieval which consists of selecting and adding terms to user queries with the intention or goal to minimize query-document mismatch and thereby improving the retrieval performance.
Query expansion (QE) is the process of reformulating a seed query to improve retrieval performance in information retrieval operations.
The following techniques are used for Query Expansion:

Wordnet
WordNet can be said as a superficial Thesaurus. WordNet is a lexical database in English which has Nouns Verbs Adjectives and Adverbs which are grouped in as cognitive synonyms also called Synsets. WordNet is different from all others by these features called relation that is WordNet structure is dependent on the words and its synonyms. By using WordNet for every word in the query the synonyms will be obtained out of which two synonym words will be added to the query.

Mutual Information
Mutual information is one of many quantities that measures the mutual dependence of two random variables. The concept of mutual information is intricately linked to that of entropy of a random variable, a fundamental notion in information theory that quantifies the expected amount of information held in a random variable. Association of every token is computed during the Tokenization and based on the formula mentioned below the Mutual Information values are computed.

$$I(x,y) = \log_2 \frac{P(x,y)}{P(x).P(y)}$$

For every token in the Query based on the Mutual Information the top two tokens with high values will be added to the Query.

Correlation (Association Matrix)
Association matrix are the matrices which quantify term correlations in terms of how frequently they co-occur. In this section a Document-Term matrix is formed with rows as Product Titles and Columns as the distinct tokens/terms. An intermediate term-term association matrix can be created

by multiplying the transpose of the above document-term matrix (which is a term-document matrix) by the document-term matrix, itself. This will result in a symmetric term-term matrix where each entry Cij corresponds to the dot product of the corresponding terms.

$$c_{ij} = \sum_{d_k \in D} f_{ik} \times f_{jk}$$

The final association matrix can be computed by normalizing the entries.

$$s_{ij} = \frac{c_{ij}}{c_{ii} + c_{jj} - c_{ij}}$$

For every token in the Query based on the Association/Correlation Matrix the top two tokens with high values will be added to the Query. This results in expanding queries with statistically most similar terms.

Since 114425 unique tokens were generated for Product Descriptions it is not possible to Construct a Association Matrix as it results in getting Memory issues/Errors. So, Association Matrix is constructed only for tokens (12820) in the Product titles and the Query expansion is done with respect to the Titles.

Feature engineering

Feature engineering is about creating new input features from your existing ones so that the Machine Learning Algorithms will work. It is an important step as it helps us to extract some hidden information underlying the dataset by performing different operations.
In this project using domain knowledge of the data Numerical features were created from the Text data create so that Regression model is trained to predict Relevance scores.

From the table below we have given name of different features and detailed description is commented in the code when these respective features were generated.

(A)List of Similarity Features: -

Similarity Feature	Formula
Cosine Similarity	$sim(Q,D) = \frac{\sum_{j=1}^{t}(w_{q_j} \cdot w_{d_j})}{\sqrt{\sum_{j=1}^{t}(w_{q_j})^2 \cdot \sum_{j=1}^{t}(w_{d_j})^2}}$
Dice Similarity	$sim(Q,D) = \frac{2 \cdot \sum_{j=1}^{t}(w_{q_j} \cdot w_{d_j})}{\sum_{j=1}^{t}(w_{q_j})^2 + \sum_{j=1}^{t}(w_{d_j})^2}$

Dot Product	$sim(Q,D) = \sum_{j=1}^{t}(w_{q_j} \cdot w_{d_j})$
Jaccard Coefficient	$sim(Q,D) = \frac{\sum_{j=1}^{t}(w_{q_j} \cdot w_{d_j})}{\sum_{j=1}^{t}(w_{q_j})^2 + \sum_{j=1}^{t}(w_{d_j})^2 - \sum_{j=1}^{t}(w_{q_j} \cdot w_{d_j})}$

(B)List of Statistical Features: -

Statistical Feature	*Description*
Product_Length	Document Length of the Product.
Query_Length	Document Length of the Query.
Common_Words_Ratio_Query	Ratio of Common words between Query and Product to the number of words in Query.
Common_Words_Ratio_Product	Ratio of Common words between Query and Product to the number of words in Product.

Even though the Similarity Features are statistically good enough to train the Regression Model, the reason for generating Statistical features is because as the Dot product is the numerator for every similarity measure the similarity values will be greater than zero only if Dot product > 0 indicating the strong correlation between them which will lead to Overfitting of the Model. Hence to overcome that problem Statistical Features are also generated.

For computing the Similarity measures, Document lengths of Query and Product the TF_IDF values are considered as weight for the terms. Subsequently the Features are generated for Raw data (with no query expansions), for every query Expansion technique with respect to Product titles and descriptions. Since the Tokens values like IDF, TF etc. are different for Product title and Descriptions the features generated at every step will have different values.

But when the Query expansion is done using WordNet, more statistically irrelevant terms with corpus is added to the query. So, the features generated with the WordNet will not be statistically significant. Hence the features generated at every step except with WordNet will be concatenated to give the Full data which will be used to build the Machine Learning Models.

Training the Machine Learning Model

Since our goal is to predict the Relevancy scores which is continuous numerical attribute, Regression Models are required to train for the Predictions. In this project we want to analyze the results by training the models with Regression Algorithms like Random Forest, Nearest Neighbors, Ridge Regression and Gradient Boosting. Optimal parameter configuration is determined for all of these models by Hyper parameter tuning. The best algorithm low test RMSE value will then be fed to Ensemble approach to determine the optimal number estimators.

Results and Conclusion

Machine Learning Algorithm	*Optimal parameter Configuration*	*Test RMSE*
Random Forest Regression	Min_Parent_cases = 77 Min_Child_cases = 38	0.49097
Gradient Boosting Regression	Learning Rate = 0.2	0.53378
Nearest Neighbors Regression	Neighbors = 15	0.52245
Ridge Regression	Alpha = 0.05	0.50220

From the above table we can see that Random forest has low Test RMSE. Hence the Random Forest is further analyzed to find the optimal number of estimators for increasing the performance of the Model. **And with 13 estimators for Random Forest the Test_RMSE is further decreased to 0.48614.**

Future Work

As far as Future work is concerned what we have done during this paper is just a very small fragment of a larger set. We would like to emphasis many more techniques and features for our future work namely we want to explore different stemming algorithms like “snowball” and lemmatizers. And also, for Query Expansion we would like to use techniques like Word to vector algorithms and other advanced techniques in NLP. We also want to try out different Deep learning methods for Regression and NLP Regression.