LEARNING TO RANK WEB DATA USING MULTIVARIATE ADAPTIVE REGRESSION SPLINES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

GÜLŞAH ALTINOK

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
STATISTICS

SEPTEMBER 2018

Approval of the thesis:

LEARNING TO RANK WEB DATA USING MULTIVARIATE ADAPTIVE REGRESSION SPLINES

submitted by **GÜLŞAH ALTINOK** in partial fulfillment of the requirements for the degree of **Master of Science in Statistics Department, Middle East Technical University** by,

Prof. Dr. Halil KALIPÇILAR ______________
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Ayşen AKKAYA ______________
Head of Department, **Statistics**

Prof. Dr. İnci BATMAZ ______________
Supervisor, **Statistics Department, METU**

Prof. Dr. Pınar KARAGÖZ ______________
Co-supervisor, **Computer Engineering Department, METU**

Examining Committee Members:

Assoc. Prof. Dr. Ceren VARDAR ACAR
Statistics Department, METU ______________

Prof. Dr. İnci BATMAZ
Statistics Department, METU ______________

Assist. Prof. Dr. Fatma YERLİKAYA ÖZKURT
Industrial Engineering Department, ATILIM UNIVERSITY ______________

Date: ______________

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: GÜLŞAH ALTINOK

Signature :

ABSTRACT

LEARNING TO RANK WEB DATA USING MULTIVARIATE ADAPTIVE REGRESSION SPLINES

Altınok, Gülşah
M.S., Department of Statistics
Supervisor : Prof. Dr. İnci BATMAZ
Co-Supervisor : Prof. Dr. Pınar KARAGÖZ

September 2018, 97 pages

A new trend, called learning to rank, has recently come to light in a wide variety of applications in Information Retrieval (IR), Natural Language Processing (NLP), and Data Mining (DM), to utilize machine learning techniques to automatically build the ranking models. Typical applications are document retrieval, expert search, definition search, collaborative filtering, question answering, and machine translation. In IR, there are three approaches used for ranking. The one is traditional model approaches such as Boolean Model (BM), Vector Space Model (VSM) and classical Probabilistic Model (classical PM). The second approach is called Language Model (LM). Such models are n-gram Model, Query Likelihood Model (QLM). The final method is namely system model including Support Vector Model (SVM) and Artificial Neural Network (ANN). In this study, we adopted the system model approach and compared the performance measures of those widely used models, SVM and ANN with those Multivariate Adaptive Regression Splines (MARS) and its variant Conic Multivariate Adaptive Regression Splines (CMARS). Results indicate that MARS performs slightly better than the others considered in this study.

Keywords: Learning to rank, Information Retrieval, Data Mining, Multivariate Adaptive Regression Splines

ÖZ

ÇOK DEĞİŞKENLİ UYARLANABİLİR REGRESYON EĞRİLERİ İLE WEB VERİLERİNİ SIRALAMAYI ÖĞRENME

Altınok, Gülşah
Yüksek Lisans, İstatistik Bölümü
Tez Yöneticisi : Prof. Dr. İnci BATMAZ
Ortak Tez Yöneticisi : Prof. Dr. Pınar KARAGÖZ

Eylül 2018 , 97 sayfa

Son zamanlarda sıralama öğrenme adında yeni bir yaklaşım, Bilgi Edinme (BE), Doğal Dil İşleme (DDİ) ve Veri Madenciliği (VM) uygulamaları gibi geniş bir yelpazede otomatik olarak sıralama modeli oluşturmak amacıyla makine öğrenme teknikleri kullanmak için ortaya çıkmıştır. Bu tekniğin tipik uygulama alanları arasında belge alma, uzman arama, tanımlama arama, işbirlikçi filtreleme, soru yanıtlama ve makine çevirisi gibi pratikler mevcuttur. Bilgi edinme alanında, sıralama için kullanılan üç yaklaşım vardır. Bu yaklaşımlardan biri, Boole Modeli (BM), Vektör Uzay Modeli (VUM) ve klasik Olasılık Modeli (klasik OM) gibi geleneksel modelleri kapsayan bir yaklaşımdır. Bilgi edinmede kullanılan bir diğer yaklaşım, doğal Dil Modeli (doğal DM) olarak adlandırılır. Bu yaklaşım n-gram Modelleri ve Sorgu Olabilirlik Modellerini (SOM) kapsamaktadır. Üçüncü yaklaşım ise, Destek Vektör Modeli (DVM) ve Yapay Sinir Ağı (YSA) gibi makine öğrenim yöntemlerini içeren sistem modeli yaklaşımıdır. Bu çalışmada, sistem modeli yaklaşımının benimsenmiş ve yaygın olarak kullanılan bu modellerinin, DVM ve YSA'nın, performans ölçümlerinin, Çok-değişkenli Uyarlanabilir Regresyon Eğrileri (ÇURE) modelinin ve Konik ÇURE modelinin (KÇURE) performans değerleri ile karşılaştırması çalışılmıştır. Sonuçlar ÇURE modelin bu çalışmada göz önüne alınan diğerler modellerden biraz daha iyi olduğunu göstermektedir.

Anahtar Kelimeler: Sıralama Öğrenme, Veri Madenciliği, Çok Değişkenli Uyarlanabilir Regresyon Eğrileri

To my beloved family, my husband Mustafa Altınok and Firuze

ACKNOWLEDGMENTS

I would like to thank my supervisors Prof. Dr. İnci Batmaz and Prof. Dr. Pınar Karagöz for their devoted supports, guidance and friendship. It was a great honour to work with them. I also would like to express my gratitude to Ms. Ceyda Yazıcı, research assistant at the Department of Statistics, Middle East Technical University, Ankara, Turkey, for her guidance in developing MARS and CMARS models.

I also would like to thank my family for the mental support and motivation they have provided. My beloved, husband Mustafa Altınok, has provided an invaluable support. Without him, this work could not have been completed. He always make me feel loved and cared and strong. I am also thankful for all the love and support by Cande Kurt, Z. Ebru Öztürk, Z. Özge Tüzer. They've given me the whole motivation during this study and taught me real meaning of friendship. And there is my second family that were and will be with me many years. They defined me, they made me who I am, they are true owners of this work. Those valuable people are Emre Erdoğan, Görkem Erdoğan and Burak Özkum. Tolga Atalay is also my best supporter for this study. Finally, I would like to thank Firuze. She was with me until late hours, she became tired with me, she gave much support to me.

TABLE OF CONTENTS

CHAPTERS

APPENDICES

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ANN	Artificial Neural Network
ANOVA	Analysis of Variance
AUC	Area Under the Curve
BF	Basis Function
BM	Boolean Model
CMARS	Conic Multivariate Adaptive Regression Splines
CQP	Conic Quadratic Problem
DCG	Discounted Cumulative Gain
DM	Data Mining
DL	Document Length
GCV	Generalized Cross Validation
HMM	Hidden Markov Model
idf	Inverse Document Frequency
IR	Information Retrieval
KL-divergence	Kullback- Leibler Divergence
LETOR	Learning to Rank
LM	Language Model
LMIR	Language Model in IR
LSE	Least Square Estimation
MAP	Mean Average Precision
MARS	Multivariate Adaptive Regression Splines
MLE	Maximum Likelihood Estimation
MS	Mean Square
NDCG	Normalized Discounted Cumulative Gain
NLP	Natural Language Processing
NN	Neural Network
PM	Probabilistic Model
PRP	Probability Ranking Principle

PRSS	Penalized Residual Sum of Squares
QLM	Query Likelihood Model
RANOVA	Repeated measure Analysis of Variance
ROC	Receiver Operating Characteristic
SS	Sum of Squares
SSE	Sum of Square Estimation
SVM	Support Vector Machine
tf	Term Frequency
tf-idf	Term Frequency Inverse Document Frequency
TREC	Text Retrieval Conference
URL	Uniform Resource Locator
VSM	Vector Space Model
WWW	World Wide Web

CHAPTER 1

INTRODUCTION

In 1989, Tim Berners-Lee invented a tool that becomes an indispensable part of our lives, *World Wide Web* (WWW). Even though this event has been one of milestones of the Information Age, in early years, to get the required information is like looking for a needle in a haystack. Thus, people were frustrated of web search. Link analysis appeared in 1998 and saved the WWW. Later, Web search made a significant progress. This, domino effect finally leads a discipline to find material and get an information need from within large collections. This discipline is called web information retrieval (web IR) (Langville and Meyer, 2011).

IR deals with retrieving of a piece of information from immense and complex sources on the internet. In other words, there exists a database of documents such that $C = \{D_1, D_2, ..., D_N\}$, and the elements of it are ranked from the most relevant to irrelevant with respect to the given query, Q. In IR, relevance denotes how well a retrieved document or set of documents meets the information need of the user. Moreover, ranking is one of the significant tasks for many applications in IR. To score documents for ranking, retrieval function, *s*, is used such that s(Q, $D_i) \in$ R, and it is also called *similarity degree*. In ranking, word order is important to get right documents.

The search result ranking evaluation is done either by actual users' datasets or by developing and testing a system. To put it more explicitly, there are two major categories to challenge the ranking problems in IR: one is the traditional non-learning approaches and the other one is the learning to rank approach (LETOR) (Li, Wu and Burges, 2007). The main difference between the two approaches is that in learning to rank, machine can automatically learn the parameters of the ranking function. In other words ranking parameters are learnt via the training dataset. On the other hand, the traditional method usually determines the parameters heuristically. Moreover, it has the disadvantage that if a user search a query in `system A` first, then s/he becomes familiar with the documents and do a bias search in `system B`. Thus, the learning to rank approach is more appropriate especially in academic research (Zhai, 2008).

The contributions of this thesis work can be summarized as follows:

- To show that MARS and CMARS can be used as alternative to baseline models.
- To see how effective MARS and CMARS are on high-dimensional datasets.
- To see whether MARS and CMARS have effect on different sizes of a dataset.

In Chapter 2, a background information is given about three known IR model types; traditional models, language Models (LMs) and system models (Ceri et al., 2013). The ranking process of traditional models are based on a user decision. Boolean Models (BMs), Vector Space Models (VSMs) and classical Probabilistic Models (classical PMs) are well-known traditional IR models. BMs provides precise match according to the given query, Q (Allan, 2006). In VSMs, a similarity value is calculated for each (D, Q) pair. Then, the documents are sorted by this value (Ceri et al., 2013). On the other hand, classical PMs represent probability of relevance (Ceri et al., 2013). Another common studies on traditional approach is language Models (LMs). The LMs usually aim to generate a word sequence according to a given query. Thus, it is widely used in linguistic research such as speech recognition, machine translation (Li, 2014). There are two models in this category. One is N-gram models used for word prediction and Query Likelihood Models (QLMs) for (D, Q) pairs (Chen et al., 1998). The system models, also known as learning to rank models, aim to provide a ranking model by teaching a machine. In IR, Support Vector Machine (SVM) and Artificial Neural Network (ANN) are two widely known system models. In addition, some related studies about learning to rank approaches are discussed in this chapter.

Chapter 3 describes the four methods used in this thesis. Two of them, SVM and ANN are widely used models while the other two, Multivariate Adaptive Regression Splines (MARS) and Conic MARS (CMARS), are novel methods in this field. SVM is the most frequent approach used in IR. Since SVM is a supervised learning technique, it is used for classification, ranking based on a training dataset of observations. In this technique, given a set of instances belonging to two classes as Vectors in a dimensional space, an optimal hyperplane separating the two classes is found. Because of this feature, SVM is an extremely efficient technique for supervised classification due to their convergence properties and this makes the SVM applicable for ranking (Joachims, 2002). In ANN model, the network of the model is 3-layered; queries (Q), terms (T), documents (D). Q's are connected to T's to D's asymmetrically and these connections work only forward directions, feed-forward ANN. Ranking a set of objects by an ANN model, that are sorted with respect to a given criterion, is based on its relevancy measure. The main objective of an ANN model is to contribute an ideal ranking document pool with respect to queries (Kwok, 1989). MARS and its variant CMARS are considered in this study due to the success in modelling high-dimensional large datasets. MARS model, developed by Friedman (1991), is formed by basis functions (BFs) and to get the best model, two stages are applied to the model. Those are the forward stage which shapes the BF pairs and the backward stage which eliminates the terms with minimum effect. CMARS, on the other hand, deals the process as an optimization problem and use Tikhonov regularization instead of the backward stage (Weber et al., 2012). This chapter also defines the performance measures in order to compare those four models and repeated measure ANOVA (RANOVA) method to test if there exists a significant difference between these measures. In Chapter 4, two datasets used in this thesis, Microsoft Bing Data and LETOR 4.0 are introduced. Tools and its related packages that used to construct the models, are also stated. Moreover, the experimental results for five folds of each datasets are discussed in this chapter. The results indicates that MARS method outperforms the others while ANN is the second best with respect to accuracy criterion. ANN, on the other hand, is more stable and also has the most robust stability. Besides, all methods are almost performing the same with respect to the robustness criterion. So, it can be concluded that MARS and ANN are two prominent methods in learning to rank applications.

This thesis organized as follows: Part 2 provides a background information about IR and three basic models used in IR. Part 3 provides a brief description about methods used in this study. Part 4 gives information about datasets, tools used in this study and the experimental results. Finally, the last part concludes the thesis.

CHAPTER 2

BACKGROUND AND RELATED STUDIES

In this chapter, we explain basic IR models- traditional models, language models- as background info (Ceri et al., 2013). In addition, we summarize similar works for the system model that we are working on in the thesis.

2.1 Traditional Models (Actual Users' Models)

There are three traditional IR models; Boolean, Vector Space, Probability. These models provide the basics for query evaluation, the process that retrieves the relevant documents, $D_i(i = 1, ..., N)$ from a document collection, C=$\{D_1, D_2, ..., D_N\}$, upon a user's query, Q. The three models represent documents and compute their relevance to the user's query in very different ways (Ceri et al., 2013).

2.1.1 Boolean Model

BM is the most basic model which provides exact matching. The fact that a query must specify a Boolean condition is a must for this model. In other words, queries must be Boolean expressions, involving {AND, OR, and NOT} of keywords. For a given query, all the matched documents have the same degree but term satisfaction may differ (Allan, 2006). Thus, in BM, ranking is not provided via calculated score. Instead, it is obtained by users' interest. An example is given as follows:
Suppose that there is a set of documents, D_i, with some specific terms, T_i. The

number of T's that the documents includes is given Table 2.1[1].

Table2.1: Document - Query table of the BM example

	nuclear	**nonproliferation**	**treaty**	**Iran**
D1	0	0	0	0
D2	1	0	0	1
D3	0	0	1	0
D4	0	0	1	1
D5	1	1	0	0
D6	0	0	1	1
D7	1	0	1	0
D8	0	1	1	1

From the Table 2.1, we would like to retrieve best documents through the given query, Q=(nuclear **AND** treaty) **OR** ((**NOT** treaty) **AND** (nonproliferation **OR** Iran)). The solution set for each Boolean expression is as follows:

- (nuclear **AND** treaty) $\Rightarrow$ {D7}
- (**NOT** treaty) $\Rightarrow$ {D1, D2, D5}
- (nonproliferation **OR** Iran) $\Rightarrow$ {D2, D4, D5, D6, D8}

Thus, the algebra becomes Q= {D7} **OR** ({D1, D2, D5} **AND** {D2, D4, D5, D6, D8}). After simplification, the retrieved documents are {D7, D5, D2}.

2.1.2 Vector Space Model

In the VSM, documents and queries are two terms in a high-dimensional space. It is based on similarity degree. The output documents are ranked according to this similarity value. VSM is more flexible than Boolean. The importance is reflected by binary weights (w), that is 1 if the term occurs in the document and 0 if it does not. Besides that, frequency has also an effective role; the more frequent terms in

[1] Following example is retrieved from the website, http://www.ccs.neu.edu/home/jaa/CSG339.06F/Lectures/boolean.pdf on August 8^{th}, 2018.

a document are more important. Therefore weights are usually calculated by using tf-idf (term frequency- inverse document frequency) weighting. For this method, three basic factors, *"local frequency in a doc/query"*, *"global frequency of term in collection"* and *"document's length"*, are essential.
Let f_{ij} be the frequency of the term T_i in the document D_j for a given query, Q. We can normalize the *term frequency* (tf) through the whole body as given in Eq. (2.1).

$$tf_{ij} = \frac{f_{ij}}{max\{f_{ij}\}}. \tag{2.1}$$

Let Df_i be the number of documents containing the term T_i and let idf_i be the idf of the term T_i as given in Eq. (2.2).

$$idf_i = log(\frac{N}{Df_i}), \tag{2.2}$$

where N is the number of documents. Finally, the typical weighting formula for the document, D_j, is given in Eq. (2.3).

$$w_{ij} = tf_{ij}.idf_i = \frac{f_{ij}}{max\{f_{ij}\}}.log(\frac{N}{df_i}). \tag{2.3}$$

Note that the weight of the query, Q, is also computed by the same formula. Additionally, the similarity degree between the document D_j and the query Q can be computed as the vector inner product, $sim(\vec{\boldsymbol{D_j}}, \vec{\boldsymbol{Q}}) = \vec{\boldsymbol{D_j}} \bullet \vec{\boldsymbol{Q}}$. However, the inner product measures how many terms are matched but not how many terms are not matched. The cosine similarity, normalized by the length of the documents, D_j, and the length of the query, Q, is a better option because cosine measures the angles between the two vectors. It is calculated by Eq. (2.4) given below:

$$cosSim(\vec{\boldsymbol{D_j}}, \vec{\boldsymbol{Q}}) = \frac{\vec{\boldsymbol{D_j}} \bullet \vec{\boldsymbol{Q}}}{|\vec{\boldsymbol{D_j}}| \bullet |\vec{\boldsymbol{Q}}|} = \frac{\sum_{i=1}^{t} w_{ij} w_{iq}}{\sqrt{\sum_{i=1}^{t} w_{ij}^2 \sum_{i=1}^{t} w_{iq}^2}}. \tag{2.4}$$

VSM has some positive aspects as well as negative ones (Ceri et al., 2013):

- It is simple
- It considers existence frequencies
- It is used for partial matching and ranking
- It allows efficient implementation

- It is more flexible than Boolean

Its disadvantages are also itemized as follows:

- It is weak for semantic and synthetic information
- It has weak control on frequencies of two or more query

As an example, consider a very small collection, C, that consists of the following three documents (Manning et al., 2008)[2]:

D1: “new york times”

D2: “new york post”

D3: “los angeles times”

For all documents, *tf* scores tables, calculated by Eq. (2.1), are given as follows:

Table2.2: Document - Query table of the VSM example

	angeles	**los**	**new**	**post**	**times**	**york**
D1	0	0	1	0	1	1
D2	0	0	1	1	0	1
D3	1	1	0	0	1	0

Total number of documents is N=3 and *idf* scores obtained from Eq. (2.2) are as follows:

• angeles	log(3/1) = 1.584	• post	log(3/1) = 1.584
• los	log(3/1) = 1.584	• times	log(3/2) = 0.584
• new	log(3/2) = 0.584	• york	log(3/2) = 0.584

Now calculate the weights of i_{th} terms in j_{th} document, w_{ij}, by using the formula given in Eq. (2.3). Results are indicated in Table 2.3.

[2] Following example retrieved from http://www.site.uottawa.ca/~diana/csi4107/ on August 8th, 2018.

Table2.3: The weights table of (D, Q) pair

	angeles	los	new	post	times	york
D1	0	0	0.584	0	0.584	0.584
D2	0	0	0.584	1.584	0	0.584
D3	1.584	1.584	0	0	0.584	0

For the following query, Q= "new new times", we calculate the tf-idf vector, and compute the score for each document in C relative to this query, using Eq. (2.4). The weight of Q, calculated by using Eq. (2.3), is as follows:

Q	0	0	0.584	0	0.292	0

and lengths of documents and query are calculated as follows:

$$Length(D1) = \sqrt{0.584^2 + 0.584^2 + 0.584^2} = 1.011$$
$$Length(D2) = \sqrt{0.584^2 + 1.584^2 + 0.584^2} = 1.786$$
$$Length(D3) = \sqrt{1.584^2 + 1.584^2 + 0.584^2} = 2.316$$
$$Length(Q) = \sqrt{0.584^2 + 0.292^2} = 0.652$$

Once the lengths are calculated, the cosine similarities are evaluated by using Eq. (2.4). The results are as follows:

$$cosSim(D1, Q) = \frac{(0*0+0*0+0.584*0.584+0*0+0.584*0.292+0.584*0)}{(1.011*0.652)} = 0.776$$
$$cosSim(D2, Q) = \frac{(0*0+0*0+0.584*0.584+1.584*0+0*0.292+0.584*0)}{(1.786*0.652)} = 0.292$$
$$cosSim(D3, Q) = \frac{(1.584*0+1.584*0+0*0.584+0*0+0.584*0.292+0*0)}{(2.316*0.652)} = 0.112$$

According to the similarity values, the ranks of the documents present as a result of the query will be, D1, D2, D3.

2.1.3 Classical Probabilistic Models

Probabilistic models (PMs) attempt to represent the probability of relevance by computing similarity coefficient between the query and the document (Ceri et. al, 2013). To generate preliminary probability for ideal results, bootstrapping is an important issue. Relevance feedback that is similarity degree, is important. It can improve the ranking by giving better term probability estimates. Moreover, the Probability Ranking Principle (PRP) justifies ranks in descending order of probability of relevance. Two well-known PMs are given below (Soboroff, 2001):

- *Binary Independence PM:* It is a classical PM. It has traditionally been used with PRP. Binary relevance is assumed. Terms are independent of each other.
- *Okapi (BM25) PM:* It is a PM that incorporates term frequency. For modern full-text search collections, a model should pay attention to term frequency and document length. "BM" is for short catalogue, Okapi is more sensitive to these quantities. It has been succesfully applied on TREC[3] tasks.

Classical PMs have some difficulties; evidence is based on an unwell representation and computing the term probabilities exactly according to the model is intractable. Term probabilities (weights) are computed by using the *idf*. To construct a model by this probabilities, two assumptions must be satisfied:

1. $P(T_i|R)$, where R is the known relevant of D_j, is constant (usually 0.5)
2. $P(T_i|\ R)$ is approximated by the distribution of term, T_i

The probabilistic model (weight) formula without similarity degree is as given in Eq. (2.5).

$$w_T = P(T|\overline{R}) = log(\frac{N - n + 0.5}{n + 0.5}). \tag{2.5}$$

Once similarity degree is got from the users, final model is constructed by Eq. (2.6).

$$w_i = log(\frac{(r + 0.5)(N - R - n + r + 0.5)}{(n - r + 0.5)(R - r + 0.5)}), \tag{2.6}$$

where N is the total number of documents, R is "the known relevant", n_i documents containing term i, r_i is relevant.

As an example, consider a collection C of documents with the following terms (Soboroff, 2001)[4]:

[3] For detail information about **T**ext **RE**trieval **C**onference (TREC), please refer to the website https://trec.nist.gov/

[4] Following example retrieved from https://www.csee.umbc.edu/~ian/ on November 10th, 2015.

Table2.4: Document - Query table of the PM example

	col	**day**	**eat**	**hot**	**lot**	**nin**	**old**	**pea**	**por**	**pot**
D1	1			1				1	1	
D2								1	1	1
D3		1				1	1			
D4	1			1						1
D5								1	1	
D6			1		1					

The weights of D_i obtained from Eq. (2.5) is given as follows:

w_T 0.26 0.56 0.56 0.26 0.56 0.56 0.56 0.0 0.0 0.56

q1 = eat, q2 = porridge, q3 = hot porridge, q4 = eat nine day old porridge

For q3= hot porridge, weights are given below with known relevant, R=1:

Table2.5: Document - Query table of the PM example

	col	day	eat	hot	lot	nin	old	pea	por	pot
D1	1			1				1	1	
D2								1	1	1
D3		1				1	1			
D4	1			1						1
D5								1	1	
D6			1		1					

w_T -0.33 0.0 0.0 -0.33 0.0 0.0 0.0 0.62 0.62 0.95

For known parameters that are N=6 and R=1, binary relevant value, r, for each term, and number of documents containing term, n, are determined. Later, the weight are calculated by using Eq. (2.6). The terms with positive weights are “pea”, “por”, “pot” and the only D2 has all of them. Thus, the relevant document, D2 with positive weights, is highlighted for q3.

2.2 Language Models

Current language models (LMs) extend classical PMs by considering specific representations of documents and queries. LMs aim to predict the next word (wd) or rank of the document, D, related with given query, Q. Such models are vigorous for tasks like speech recognition, spelling correction, and machine translation, where the probability of a term conditioned on surrounding context is needed. In this model, it is assumed that the word sequence is generated by independent words (“bag of words”). Thus, the word sequence distribution computed as a multiplication of each word’s pdf. Parameters are estimated by maximum likelihood estimation (MLE). The probability of a sequence of words is equal to the product of the probability of each words. The LM has significant relations to traditional tf-idf models (like VSM). Query likelihood retrieval models- scoring documents models- are the basic LM approaches. The simplest models are N-gram models. Hidden Markov Model (HMM) is also considered in today’s researches. In an HMM, the state is not directly visible, while state-dependent output is visible. Each state is associated with an “initial” probability.

2.2.1 N-gram Models

The simplest form of the LM simply throws away all conditioning context and estimates each term independently by their frequency in the collection (Li, 2014). Such a model is called a *unigram language model* and the simple equation with "k" words is represented in Eq. (2.7) as follows:

$$P_{uni}(wd_1 wd_2 wd_3 wd_4 \ldots wd_k) = P(wd_1)P(wd_2)P(wd_3)P(wd_4)\ldots P(wd_k). \quad (2.7)$$

There are many more complex kinds of LMs, such as bigram language models which condition on the previous term such that the formulation states as given in Eq. (2.8).

$$P_bi(wd_1 wd_2 wd_3 wd_4 \ldots w_dk) = P(wd_1)P(wd_2|wd_1)P(wd_3|wd_2)\ldots P(wd_k|wd_{k-1}). \quad (2.8)$$

2.2.2 Query Likelihood Models

QLM is the first generation of LMs in IR (Li, 2014). The basic idea is computing the query likelihood score from estimated language model in order to give score a document.

Let Q be a query, D be a document and θ_D be an estimated LM (e.g. multinomial model, multiple Bernoulli model, multiple Poisson model) based on the document D. The score of document D with respect to the query Q is then defined as the conditional probability $p(Q|\theta_D)$. Estimating query model, θ_D, retrieval performance changes in terms of the model chosen. In other words, different models have different retrieval behaviour. Up to now, the most popular and the most successful model is the unigram multinomial LM for θ_D. Yet, Bernoulli and Poisson models are other QLMs used recently. These three models make different assumptions about the word frequencies. The multinomial model assumes that existence of each word is independent, including the multiple occurrences of the same word. The assumption of multiple Bernoulli model is that different words occur independently. In the multiple Poisson model, each word is modelled through an independent model with a flexibility in estimating the parameters for different words in different ways. In comparison of those models, multiple Bernoulli model is weaker than the multinomial model in independency statement. Additionally, the Poisson model has a positive aspect over Bernoulli in capturing the term occurrences.

In multinomial model, since it is assumed that the word occurrences are independent, θ_D have the same number of parameters as the number of words in our vocabulary set, V. Let Q = $q_1 \ldots q_m$, the likelihood model would be as given in Eq. (2.9).

$$P(Q|\theta_D) = \prod_{i=1}^{m} p(q_i|\theta_D) = \prod_{wd \in V} p(wd|\theta_D)^{c(wd,Q)}, \tag{2.9}$$

where c(wd, Q) is the count of word, wd, in the query, Q.

In *multiple Bernoulli model*, for each word wd_i, a binary random variable, X, is defined, and the presence $(X_i = 1)$ or absence $(X_i = 0)$ of every word is assumed to the independent of each other. That means, the number of parameters is equal to the number of words in the vocabulary. Multiple Bernoulli model is able to devise both presence and absence of words in the query. Thus, likelihood model would be as shown in Eq. (2.10).

$$P(Q|\theta_D) = \prod_{wd_i \in Q} p(X_i = 1|\theta_D) \prod_{wd_j \in Q} p(X_j = 0|\theta_D). \tag{2.10}$$

In *multiple Poisson model*, a Poisson random variable, X_i, is defined for every word, wd_i, and is modelled every frequency of wd_i, independently. The multiple Poisson model with λ_i - the mean rate of a Poisson process corresponding to X_i- also has the same number of parameters as the number of words in our vocabulary. To build a model for the counts of a word in the query, the query length, m, is taken as the length of the time period. Hence, the query likelihood becomes as in Eq. (2.11).

$$P(Q|\theta_D) = \prod_{wd_i \in V} \frac{e^{-\lambda_i m}(\lambda_i m)^{c(wd_i,Q)}}{c(wd_i, Q)!}. \tag{2.11}$$

To estimate θ_D, the MLE method is used under the assumption that D is a sample of θ_D. After parameter estimation, the probability of the counts of a word in the query is calculated as the score of the document.

2.3 System Models (Learning to Rank Models)

Another approach that is within the consideration of this study is learning to rank. As Lim (2016) states occurring recently, a noticeable approach to solving the ranking problem has appeared, called learning to rank, in which machine learning methods are made use of learning predictive models that can generate satisfactory rankings. Briefly stated, it is a machine learning approach in IR for ranking any material. Distinctive areas, where learning to rank approach is used, are document retrieval, expert search, collaborative filtering, question answering, document outlining and machine translation.

In learning to rank, datasets usually consist of pair of items such as query-document, query-URL etc. The target in this technique is to teach machine so as to construct ordering models. This order can be numerical score *(e.g. 5 out of 10)*, ordinal score *(e.g. irrelevant-mostly relevant- relevant-perfectly relevant)* or binary *(e.g. relevant or irrelevant)*. Model construction task is carried out as follows: First, the training data including a set of features for documents with relevance judgement are formed. Then a ranking model is constructed. In retrieval (i.e., testing) phase, given a new query, -we call it test data- the ranking model is used to create a ranked list for the documents associated with the query. The process is visualized in Figure 2.1 (adapted from Liu, 2009).

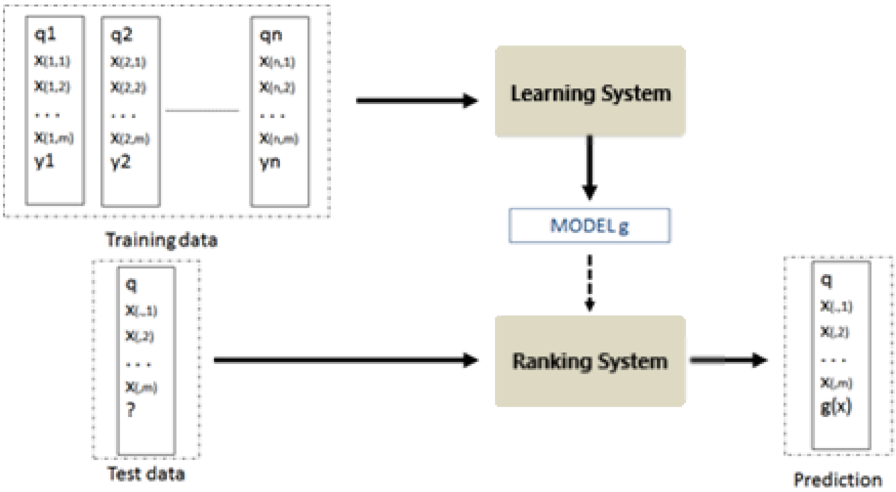

Figure 2.1: Learning to Rank process

In learning to rank, the studies are gathered around three approaches; pairwise approach, listwise approach and pointwise approach. This section introduces related work in those three methods in learning to rank.

2.3.1 The pairwise approach

The pairwise approach shapes the learning task as classification of item pairs into true and false rank. Learning to rank, particularly the pairwise approach, has been successively applied to IR. For instance, Joachims (2002) applied `Ranking SVM` to document retrieval. He developed a method for deriving document pairs for training from clicks-through data of users. Cao et al. (2006) adapted Ranking SVM to document retrieval by modifying the loss function. Furthermore, Burges et al. (2005) applied `RankNet` to large scale web search by using ANN as model, and the gradient descent to optimize the cross-entropy loss. Cao et al. (2007) has used fidelity loss function in parallel background of `RankNet` and constructed the `FRank` method. Additionally, Freund et al. (2003) has developed `RankBoost`, constructed by exponential loss function given initial distribution for document pairs. Unlike the common application in pairwise method, Cao et al. (2006) have focused query-level normalization to the pairwise loss function to get rid of document pair distribution skewness. In other words, they performed `IR-SVM`. As a result, major improvement has been observed by using the query-level normalizer.

2.3.2 The listwise approach

The listwise method proposes ranking lists in both learning stage and prediction stage. It has two significant branches. One is direct optimization of information retrieval measures and the other one is listwise loss minimization. This is all to say that in order to study listwise ranking, some needs to either try to optimize IR evaluation measures or else something correlated to the measures, or make a loss function stated permutations by considering the properties of ranking for IR and reduce to the smallest possible degree.

At the optimization concept, Taylor et al. (2008) has soften (approximate) the evaluation measure so as to make it smooth and differentiable. First, they construct a score distribution and map it to rank distribution. Later, they computed smooth and differentiable expected Normalized Discounted Cumulative Gain (NDCG), and finally optimize it by the gradient descent method. The method is called `SoftRank`. Moreover, Yue et al. (2007) use structural SVM context for optimization in their study, called `SVM-MAP`. At first, they optimize a model by using just training data constraint. Later, learning model constructed by optimization is engaged to catch violated constraints. In the meantime, most violated constraint is found. In order to do that, task is carried out as follows: relevant and irrelevant documents are sorted and rank is used to get optimal sort lists. When the documents are sorted by their scores in descending order, the second term will be maximized. If the constraint got by optimization model is more violated than the most violated constraint in the training set, it is added to the working set. Other studies at this concept are `AdaRank` of Xu and Li (2007), which is developed by training weak learner with larger weight from the initial distributions of each query and the `RankGP` model built by Yeh et al. (2007) by using single-population GP to optimize non-smoothing non-differentiable objectives.

At the minimization concept, listwise loss functions are defined depending on the interpretation of the distinctive ranking properties of IR. There are two representative algorithms: `ListNet` and `ListMLE`. Cao et al. (2007) has studied the listwise approach in which lists of objects are used as *instances* in learning. Loss function is created by Kullback–Leibler divergence (KL-divergence) between two permutation probability distributions and the model is built by ANN through the gradient descent algorithm. The study is called `ListNet`. However the complexity of algorithm is high. Xia et al. (2008) has solved this limitation by computing the likelihood of a ground truth permutation, and maximize it to learn the model parameter. Thus, `ListMLE` algorithm has been constructed.

2.3.3 The pointwise approach

The pointwise approach has the following concept: a single document is viewed as input in learning, and it describes its loss function based on individual documents. Li et al. (2007) defined the ranking problem as a multiple classification, called `McRank`. Later, they conducted another study on multiple ordinal classification, which brings about computationally manageable learning algorithms for ranking in web search with regard to relevancy. Wu et al. (2006) consider a standard quality measure in IR, the discounted cumulative gain (DCG) criterion, in their study. The fact that perfect DCG scores and the DCG errors are bounded by classification errors make good classification. As it can be seen that even though the models learned with all the approaches are evaluated by IR measures, the methods in those three approaches use different loss functions.

Recently, LETOR method has been used in various fields. Li (2014) presents comprehensive explanations on learning for ranking on language process in his book. Furthermore, Ibrahim and Murshed (2016) make a contribution to have a vast understanding of LETOR systems and its evolution from and relation to the traditional IR methods. Bhowmik and Ghosh (2017) aggregate a set of expert opinion rank order lists by studying on unsupervised rank aggregation problem and get a notable refinement in performance on the subject of rank aggregation methods with the datasets, MQ2008, MQ2007 and OHSUMED.

Dammak et al. (2015) proposed a new active learning to rank algorithm based on boosting for active ranking functions. In order to test the algorithm, they choose three algorithms of boosting from the pairwise approach: `RankBoost`, AdaRank and LambdaMART from the listwise approach. Later, those models are compared in terms of the performance measures, Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain (NDCG), using the benchmark LETOR 4.0 dataset. For more detail about those measures, please refer to "*Active Learning to Rank Method for Documents Retrieval*" (Dammak et al., 2015).

Previous research on learning to rank is usually SVM and ANN. This study has demonstrated two recently developed technique that have not been exercised in IR. They are non-parametric models called MARS and CMARS. In this study, we consider MARS as well as CMARS, an improved version of MARS, to implement learning to rank approach. Besides, their performances are compared with those of SVM and ANN. For this thesis, we adopted pairwise approach for SVM and ANN techniques to classify document pair and get the learning technique Since MARS is a regression based algorithm, it can be categorized as pointwise approach. Detailed information about these models is given in Chapter 3.

CHAPTER 3

METHODS

In this chapter, we present the techniques that we use for learning to rank on web data. The SVM and ANN are the techniques from the literature for comparison purposes. MARS and CMARS are implemented for the first time in this thesis.

3.1 Support Vector Machine (SVM)

SVM is one of the most recurrent approach used in IR. It has proven applicable for ranking and to solve certain ranking problems (via learning to rank), a variant algorithm called *Ranking SVM* is adapted (Joachims, 2002). Detailed information about this algorithm is given in Section 3.1.2. SVMs are supervised learning techniques. Therefore, it is used for classification and ranking based on a training dataset of observations. In this technique, given a set of instances belonging to two classes as vectors in a d-dimensional space, an optimal hyperplane separating the two classes is found. SVM is an effective technique for supervised classification due to their convergence properties.

3.1.1 Methods of SVM

There are two main SVM methods: linear SVM and non-linear SVM. Linearly separable set of instances of data points $\{x_i = (x_1, x_2, .., x_k)|x_i \in X\}$ related to labels, y_i=-1 or 1 (i.e. SVM model can assign either positive label y = +1 or negative label y = -1 to any x), express an optimal hyperplane by determining weight vectors,

w. Constructing the SVM model is equivalent to solving the Quadratic Optimization problem. The optimization for the maximum margin of hyperplane is obtained from the constraint definition as given in Eq. (3.1).

$$\begin{aligned} minimize\ \langle \vec{\mathbf{w}}, \vec{\mathbf{w}} \rangle + \gamma \sum_{i=1}^{k} \xi_i \\ subject\ to\ y_i(\langle \vec{\mathbf{w}}, x_i \rangle + b) \geq 1 - \xi_i \\ \xi_i \geq 0,\ i = 1, 2, ..., k, \end{aligned} \tag{3.1}$$

where γ is a parameter to be tuned during training, b is optimized bias value and ξ is the slack variable.
In many cases, observations of real-life datasets are not linear. Thus, association of inputs are applied by using a non-linear mapping function such that $\phi :$ X $\rightarrow$ F, where X represents input points space while F is the feature space. In addition to this, hyperplane can be denoted as in Eq. (3.2).

$$f(\mathbf{x}, \alpha^{\star}, b^{\star}) = \sum_{i \in SV} y_i \alpha_i^{\star} \langle \phi(x_i).\phi(\mathbf{x}) \rangle + b^{\star}, \tag{3.2}$$

where $\alpha_i^{\star}$ is an optimized value set of the i^{th} point for misclassified x_i. Note that the inner product $\langle \phi(x_i).\phi(\mathbf{x}) \rangle$ is known as a kernel function, written K(x_i,**x**).

3.1.2 Ranking SVMs

Joachims (2002) improved the ranking SVM algorithm, called `RankNet`, which is a variant of SVM model proposed by Herbrich (1999). The algorithm rates exemplars, instead of classifying them. Given a training dataset of observations where category is annotated by users, M^{*} and a document, d_i is preferred over another one, d_j, i.e. $d_i > d_j$, $\{d_i, d_j\} \in M^{*}$. When a linear learning function, f(d) = $\vec{\mathbf{w}}$.d is applied, the followings holds:

$$\begin{aligned} d_i > d_j \Rightarrow f(di) > f(dj) \\ f(d_i) > f(d_j) \Leftrightarrow \vec{\mathbf{w}}.d_i > \vec{\mathbf{w}}.d_j, \end{aligned}$$

and the vector can be gotten by the same SVM method given in Eq. (3.1).

$$\begin{aligned} minimize\ \langle \vec{\mathbf{w}}, \vec{\mathbf{w}} \rangle + \gamma \sum_{i,j \in |M|} \xi_{i,j} \\ subject\ to\ \forall (d_i, d_j) \in M^* : \vec{\mathbf{w}}.d_i \geq \vec{\mathbf{w}}.d_j + 1 - \xi_{i,j} \\ \forall i, j : \xi_{i,j} \geq 0. \end{aligned} \tag{3.3}$$

Thus, the vector for the hyperplane, called ranking vectors, is obtained in a different way from the ranking SVM generalization. That is to say, for ranking vectors, the data is assumed as linear and vectors are points, closest to each other on the hyperplane. However, generalization is realized by computing the weight, w, to make the distance between these nearest points as large as possible.

3.2 Artificial Neural Network (ANN)

In the early years, many studies have been conducted to overcome the challenging artificial intelligence (AI) problems such as IR. For this purpose, an approach that imitates the structure and operations of the brain on machines to solve conventional AI complications has been enquired. Since it is inspired by biological neural networks (NN), it is called ANN or NN.

One of the first studies about NN approach to IR has carried out by Kwok (1989). He aimed to develop an ANN model to supply an ideal ranking document pool with respect to queries. The network of Kwok's model is 3-layered; *Queries (Q), Terms (T), Documents (D)*. Queries are connected to terms to documents asymmetrically. In addition, these connections work in two directions. A representation of Kwok's NN design is shown in Fig. 3.1.

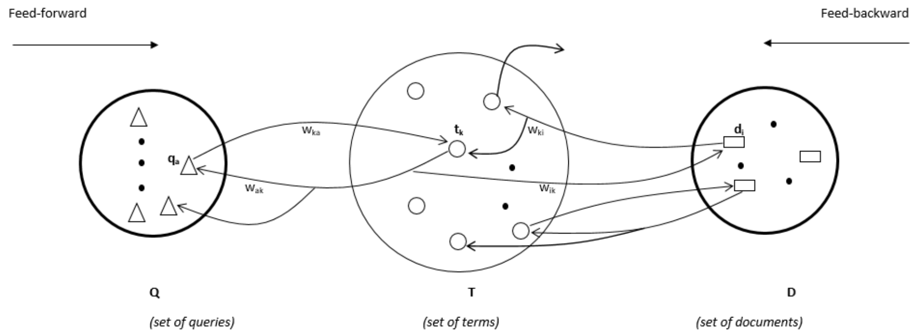

Figure 3.1: 3-Layer Network Connection for IR.

To develop neural information retrieval system, input layer (set of queries) is created of N input neurons q_1, ..., q_N, and each neuron represents one character of a query. Set of terms is considered as hidden layer with M neurons such that y_1, ..., y_M. Finally, output layer (set of documents) is created by L neurons d_1, ..., d_L.
Let y_j be in a sigmoid transition function such that:

$$y_j = \frac{1}{1 + e^{-\varphi_{y_j}}} \tag{3.4}$$

where φ is hidden layer based on the following formula:

$$\varphi_{y_j} = \sum_{i=1}^{N} w_{ij} q_i(t) + \vartheta_{y_j}, j = 1, ..., M \tag{3.5}$$

3.2.1 Feed-forward Neural Networks

Feedforward NNs are the first invented type of ANN, and they are more basic than their counterparts.

As it can be seen from Fig. 3.2 (McGonagle, 2018), the connections between units proceed forward in the network, therefore there is no loop. They go through output nodes from inputs in a one way (left to right or right to left).
As in this study, the feed-forward method is used for supervised learning. To put it another way, the feedforward NN build a model/function with a set of independent variables such that $y \approx f(x)$ for training pairs (x,y).

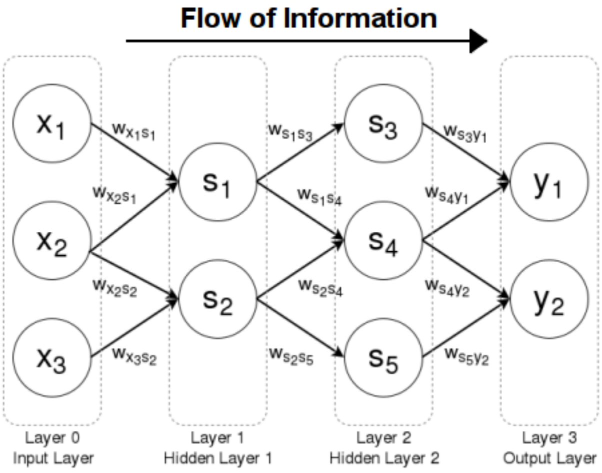

Figure 3.2: A feedforward neural network example

3.2.2 SortNet

As stated in Section 2.1, ranking a set of objects, that are sorted with respect to a given criterion, is based on its relevancy measure. On the other hand, in some cases, such as personalized retrieval systems, different users have different relevance criteria which may be undefined. Rigutini et al. (2011) present a learning to rank approach called `SortNet` by using ANN application. `SortNet` is a study which aims to get the most useful patterns form the training set. In order to improve the ranking performance, a comparative ANN is used. A preference function is implemented as a standard into a typical ranking algorithm for comparison. By this way, it provides an overall ranking of a set of objects.

3.3 Multivariate Adaptive Regression Splines (MARS)

MARS, made known by Friedman (1991), is a technique to find a solution to regression-type problems. Its goal is to estimate what will be the point of a dependent variable from a group of independent variables. MARS algorithm is a methodology developed by Friedman as a non-parametric method for multiple regression. It uses adaptively selected spline functions. Due to the fact that MARS is adaptable, it is often used in constructing model where functions represent high variation in one region of the predictor space and are smoother in other parts. In other words, MARS produces continuous models for high-dimensional data that can have multiple partitions and predictor variable interactions.

Due to the fact that MARS does not suppose or enforce any presumption, it is considered useful technique in DM. In addition to this, it does not take on or impose any specific type or class of relationship such as logistic, linear etc., between predictor and outcome data. As an alternative, suitable models (i.e., models that yield accurate predictions) can be derived even in situations where the relationship between the predictors and the dependent variable is non-monotone and difficult to approximate with parametric models. Thus, being a flexible regression model for high-dimensional data, MARS has many positive aspects. As Nisbet et al. (2009) states, MARS is an easy technique to apply classification problems due to the fact that it can handle multiple dependent variables. Furthermore, it is appropriate to use for modelling large datasets. Another advantage of this fresh methodology is that MARS can cope with categorical data as well as continuous data.

3.3.1 Theory of MARS

Let y be the target dependent variable and **x** = $(x_1, \ldots, x_z)^T$ be a multiple z input vectors. Then, it is assumed the data are generated based on an unknown "true" model. To get the response values, the model would be as in Eq. (3.6).

$$y = f(\mathbf{x}) + \epsilon, \tag{3.6}$$

in which ϵ is the fitting error with zero mean and finite variance and f, represented by a linear combination, is the MARS model built from basis functions (BFs) given in Eq. (3.7).

$$\hat{y} = \hat{f}(\mathbf{x}) = c_0 + \sum_{i=1}^{k} c_i B_i(\mathbf{x}). \tag{3.7}$$

MARS model formula is demonstrated as an approximation of recursive partitioning regression in the shape of augmented set of BFs. c_i is the coefficient of the BF that is calculated cooperatively to get the best fit of data or the constant one (i = 0), and $B_i(\mathbf{x})$ is the BF with the two-sided truncated form that is piecewise linear function, and it is demonstrated as:

$$max(0, x-t) = \begin{cases} x\text{-}t, & \text{if } x>t \\ 0, & \text{otherwise.} \end{cases} \tag{3.8}$$

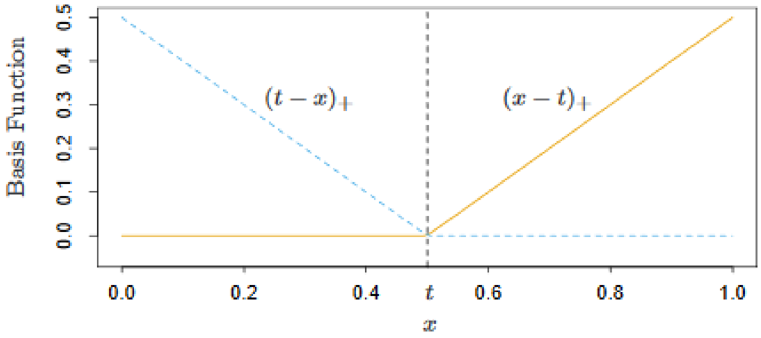

Figure 3.3: The basis functions of $(x-t)_+$ and $(t-x)_+$ by MARS

In Eq. (3.8), and Fig. 3.3 (Hastie, et al., 2009, Figure 9.9), parameter t is the knot point of the BFs. Knot values define the "pieces" of the piecewise linear regression as given in Eq. (3.9), and they are determined from the data.

$$c^{+}(x;t) = [+(x-t)]_{+}; c^{-}(x;t) = [-(x-t)]_{+} \tag{3.9}$$

By the algorithm, all knot values, all BFs and their combinations, i.e interactions, are obtained from the observations. Due to the fact that algorithm determined a massive amount of entity, it needs to be trimmed. The process is achieved by least square estimation method (LSE).

3.3.2 Algorithm of MARS

Friedman (1991) states that the MARS method for estimating the "true model" consists of forward and backward stepwise algorithms. In the forward part, MARS starts with a model consisting of the mean of the response values as the intercept terms. Then, it adds BF in pairs to the model. After every BF addition, it determines the pair of BFs that gives the maximum reduction in the sum of squares residual error (SSE). This process of adding terms continues until the change in SSE is too small to continue or until the pre-specified maximum number of terms is reached. Once the forward process is done, the backward part runs. It is the phase where reduction is applied on the model. Thus, the model gets simpler. The backward pass removes terms one by one, deleting the least effective term at each step until it finds the best submodel. Model subsets are compared using the generalized cross validation (GCV) criterion

$$GCV = \frac{\frac{1}{m}\sum_{j=1}^{m}[y_j - \hat{f}(\mathbf{x}_j)]^2}{[1 - \frac{\tilde{E}}{m}]^2}, \tag{3.10}$$

where $\tilde{E}$ is the effective number of parameters of m measurements in the model. The model with the smallest GCV value is accepted to be the best fitted model. Here note that GCV criterion provides a form of regularization by trading off the model goodness-of-fit against its complexity.

The MARS method and its algorithm have also been proposed to cope with classification problems. Stone et al. (1997) developed a model called `PolyMARS`, and it is intended to be used as a solution for classification problem, especially. It uses the multiple logistic framework, and runs the model in a forward step phase like MARS. However, unlike MARS, a quadratic approximation is utilized to search for the next BF pair at each stage usage. Once reached, the model is fit by the MLE, and the process is repeated.

3.4 Conic Multivariate Adaptive Regression Splines (CMARS)

As stated in Section 3.3, the MARS method is performed in two phases: *forward and backward steps*. By these algorithms the method aims to achieve getting a good-fitted simple model. In CMARS, an alternative to backward algorithm is adapted by Weber et al. (2012). That is, a penalized residual sum of squares (PRSS) for MARS shown in Eq. (3.11) is treated as a *Tikhonov regularization problem*, and carried on this with continuous optimization technique, *conic quadratic programming* (CQP). The process is as given in Eq. (3.11).

$$PRSS = \sum_{i=1}^{N}(y_i - f(\tilde{\mathbf{x}}_i))^2 + \sum_{m=1}^{M_{max}} \lambda_m \sum_{|\boldsymbol{\alpha}|=1}^{2} \sum_{\tau<s} c_m^2 \int [D_{\tau,s}^{\boldsymbol{\alpha}} B_m(\mathbf{t}^m)]^2 d\mathbf{t}^m, \tag{3.11}$$

for $\boldsymbol{\alpha} = (\alpha_1, \alpha_2)$, $\alpha_1, \alpha_2 \in \{0, 1\}$ and τ, s$\in V_m$, where V_m is the independent variables associated with m^{th} BF, $B_m(\mathbf{t}^m)$. Furthermore, the integration part of the Eq.

(3.11) (adapted from Weber et al., 2012) is represented in Eq. (3.12).

$$D^{\boldsymbol{\alpha}}_{\tau,s} B_m(\mathbf{t}^m) = \frac{\partial^{|\boldsymbol{\alpha}|} B_m}{\partial^{\alpha_1} t^m_\tau \partial^{\alpha_2} t^m_s}(\mathbf{t}^m). \tag{3.12}$$

3.4.1 Tikhonov Regularization

Tikhonov Regularization (*or Tikhonov-Phillips regularization*) is a common way to handle linear ill-posed problems. Regularization makes ill-posed problems stable by providing accurate approximate solutions. The formulation is shown in the Eq. (3.13) to construct the model that calculates the minimum-norm least squares (Kaipio, 2005).

$$min_{x \in X} \|Tx - y\|_2^\gamma + \rho \|x\|_X^2, \tag{3.13}$$

where $\rho > 0$ is a given constant. In order to solve the problem given in Eq. (3.11), PRSS is adapted to Tikhonov regularization problem, by Eq. (3.13) and describes as in Eq. (3.14).

$$PRSS \approx \|\mathbf{y} - \mathbf{B}(\tilde{\mathbf{d}})\mathbf{c}\|_2^2 + \lambda \|\mathbf{L}\mathbf{c}\|_2^2 \tag{3.14}$$

3.4.2 Conic Quadratic Optimization (CQP)

CQP, also known as *second-order cone optimization*, is a straightforward generalization of linear optimization. Many convex sets can be modelled using conic quadratic formulations. Here, Euclidean norm defined in Eq. (3.15) (MOSEK Modelling Cookbook 3.0, 2018) is used:

$$\|x\|_2^2 \preceq t \Longleftrightarrow (1/2, x, t) \in Q_{n+1}. \tag{3.15}$$

Finally, Weber et al. (2012) reformulated Eq. (3.14) as a *CQP* given in Eq. (3.16).

$$\begin{aligned} &\min_{t,\mathbf{c}} t, \\ subject\ to\ &\|\mathbf{y} - \mathbf{B}(\tilde{\mathbf{d}})\mathbf{c}\|_2 \preceq t, \\ &\|\mathbf{L}\mathbf{c}\|_2 \preceq \sqrt{\tilde{M}}. \end{aligned} \tag{3.16}$$

Once the optimization problem given in Eq. (3.16) is solved by the optimization software MOSEK, CMARS has risen as a new approach to MARS.

3.5 Performance Measures

To evaluate the performance of an IR system, one needs to measure how far down the ranked list of results are related to some or all of documents that a user needs. Several numerical evaluation measures can be computed for each query. As a common practice in supervised learning, a confusion matrix is constructed in order to summarize the accuracy performance of the model, as shown in Table 3.1.

Table3.1: Confusion Matrix

		True Condition	
		Condition positive	*Condition negative*
Predicted condition	*Predicted condition positive*	True Positive (TP)	False Positive (FP)
	Predicted condition negative	False Negative (FN)	True Negative (TN)

If the documents which are relevant are retrieved, they are called true positives (TP), and those that are not retrieved are named false negatives (FN). On the other hand, non-relevant documents which are retrieved are called false positives (FP), and those not retrieved are called true negatives (TN).

Following is the list of measures that are used for comparing the performances of the models developed in this study (Sokolova et al., 2009). The definition of the measures in the IR framework and their formulas are presented below.

Precision. Precision (P) measures the ability to retrieve top-ranked documents that are mostly relevant. In other words, it is a measure of how many selected items are relevant (Ceri et al., 2010). It is calculated by using the set of retrieved document that are relevant to a given complete set of document, i.e, true positive results, TP and the set of false positives, FP, that are not relevant to documents. The formulation is as given in Eq. (3.17).

$$P = \frac{|TP|}{|TP| + |FP|}. \tag{3.17}$$

In web search, high precision value is generally considered more essential. Thus, the more the fraction in Eq. (3.17) gets close to 1, the better the ranking is.

Recall (Sensitivity). Recall (Rc) measures the ability of the search to find all of the relevant items in the search. That means, it is a measure of how many relevant items are selected (Ceri et al., 2010). The difference from precision formula is that the denominator is the summation of true positive results, TP, and false negative results, FN, which is the set of documents that are related but not retrieved. The formula is given in Eq. (3.18).

$$Rc = \frac{|TP|}{|TP| + |FN|}. \tag{3.18}$$

The advantage of having both precision and recall values is that one is more important than the other in many circumstances.

F-measure. The more documents that are retrieved mean the more relevant documents which rises the recall measure. Conversely, precision value decreases because results are weakened by non-relevant documents. Likewise, fewer documents cause higher P and lower Rc value. Since both values are significant for a system, an effective measure, called F-score, may satisfy the need. It is defined as the harmonic mean of P and Rc represented in Eq. (3.19).

$$F = 2 \times \frac{P \times Rc}{P + Rc}. \tag{3.19}$$

G-measure (Fowlkes–Mallows index). It is the geometric mean of P and RC. The higher values indicates the greater similarity in document classification.

$$G = \sqrt{P \times Rc}. \tag{3.20}$$

ROC. A graphical plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied.In ROC curve, the *TP rate* is plotted in function of the *FP rate* for different cut-off points. Each point on the ROC curve represents a TP rate/FP rate pair corresponding to a particular decision threshold. A test with perfect discrimination has a ROC curve that passes through the upper left corner. Therefore the closer the ROC curve is to the upper left corner, the higher the overall accuracy of the test (Zweig and Campbell, 1993).

AUC. The area under the ROC curve (AUC) is the accuracy of the model. It is a statistics used in the model comparison. It is calculated by

$$AUC = \int_{\infty}^{-\infty} TP(x)FP(x)\, dx, \tag{3.21}$$

where TP is the true positive rate and FP is the false positive rate of the minimum value, x, of the observations. It is related with the accuracy. A test with no better accuracy than chance has an AUC of 0.5, a test with perfect accuracy has an AUC of 1.

Specificity. Specificity, also called TN rate, is the proportion of negative observations that are correctly identified. Therefore, in specificity, the smaller measure gives better solutions.

$$TNrate = specificity = \frac{TP}{TN + FP}. \tag{3.22}$$

Youden's J statistic (Informedness). It is a metric that evaluates performance in classification. It gets value between -1 and -1. If J statistic is greater than zero, the classification method works. Zero of J statistic means that the classification method has a 50% chance to predict an instance with positive class correctly, and a 50% chance to predict an instance with positive class incorrectly. The classification method works perfectly if J statistic is 1 since its sensitivity is 1 and the specificity is also 1. The J score is calculated by Eq. (3.23).

$$J = sensitivitiy + specificity - 1. \tag{3.23}$$

Fall-out. It is the proportion of non-relevant documents that are retrieved. It can be interpreted as the probability that a non-relevant document is retrieved by the query. Thus, the small fall-out values denotes a better model. In binary classification, fall-out is closely related to specificity as given in Eq. (3.24).

$$fall - out = \frac{FP}{TN + FP}. \tag{3.24}$$

Accuracy. Accuracy is the measure of how well a binary classification test correctly identifies or excludes a condition. It is the proportion of the true results. The formula is given in Eq. (3.25).

$$ACC = \frac{|TP| + |TN|}{|TP| + |TN| + |FP| + |FN|}, \tag{3.25}$$

In practice, if we have high accuracy then our has a good prediction performance. Even though accuracy is a useful measure, the statement is valid only when you have symmetric datasets where values of FP and FN are almost the same. Therefore, accuracy must be considered together with other parameters to evaluate the performance of the model.

Stability. Stability compares the performance of a model in training (MTr) and testing (MTe) data with respect to a measure M (Osei-Bryson, 2004). S value that is closer to 1 indicates a more stable model. The formula is as given in Eq. (3.26).

$$S = min[\frac{M_Tr}{M_Te}, \frac{M_Te}{M_Tr}]. \tag{3.26}$$

3.6 Repeated Measure ANOVA (RANOVA)

RANOVA is used for comparing three or more group means. Here, each group of data must be picked from the same elements. It occurs *within-subject variables*. Similar to the other ANOVA tests, the fundamental technique is a partitioning of the total sum of squares, SS, into components related to the effects used in the model. In a repeated measures design, the variability can be split into between-treatments variability and within-treatments variability. The within-treatments variability SS_w can be further partitioned into between-subjects variability (individual differences), SS_b and error, SS_{error} In addition, RANOVA also uses F-*statistic* as test statistic. The calculation of the F-statistic is as given in Eq. (3.27).

$$F = \frac{MS_{subject}}{MS_{error}} = \frac{SS_{condition}/(\daleth - 1)}{SS_{error}/(n_\varsigma - 1)}, \tag{3.27}$$

where $\daleth$ is number of conditions and n_ς is total number of cases. The $SS_{condition}$ is calculated as *between-groups variability*, SS_b, in an independent ANOVA, by Eq. (3.28).

$$SS_{condition} = SS_b = \sum_{i-1}^{\daleth} n_i(\bar{x}_i - \bar{x})^2. \tag{3.28}$$

Since, with a repeated measure ANOVA, the same subjects in each group is used, and the variability caused by the individual differences between subjects, $SS_{subjects}$, is removed from the within-groups variability, SS_w, the SS_{error} is calculated as given in Eq. (3.29).

$$\begin{aligned} SS_{subject} &= \daleth . \sum (\bar{x_i} - \bar{x})^2 \\ SS_w &= \sum_1 (\bar{x_{i1}} - \bar{x_1})^2 + \sum_2 (\bar{x_{i2}} - \bar{x_2})^2 + ... + \sum_{\daleth} (\bar{x_{i\daleth}} - \bar{x_\daleth})^2 \\ SS_{error} &= SS_w - SS_{subject} \end{aligned} \tag{3.29}$$

When the sum of squares are calculated from Eq. (3.28) and Eq. (3.29), the outcomes are divided by their corresponding degrees of freedom as seen in Eq. (3.27) to get an F-statistic.

Much the same as other ANOVA applications, in order for the test to be valid, some assumptions need to be met. These assumptions are as follows:

Normality Like the other ANOVA tests, each level of the independent variable needs to be distributed approximately normal. In order to check it, some Normality test, such as Shapiro Wilk, can be applied.

Sphericity The variance differences between all related group combinations have to be equal. To inspect this feature, the outcome of *Mauchly's test of Sphericity* is enough. For significant result, p-value must be greater than prespecificated α value which is assumed to be 0.05 as default.

In the case of failure of the assumptions, for the Normality assumption, the non-parametric Friedman test can be used instead. Moreover, the Mauchly's test has an alternative that is the p-value from the Greenhouse-Geisser correction row in the 'Tests of Within-Subjects Effects' ANOVA table.

In this study, RANOVA is used for testing whether there exists any differences between related means of the performance measures that belong to five folds among the four models. Thus, folds are treated as elements while models are predictors and performance measures are dependent variables. When RANOVA gives a significant result, we compare models in pairs to judge which of each model is preferred.

CHAPTER 4

EXPERIMENTAL EVALUATION

In this section, datasets and tools that used in the experiments are introduced. Additionally, the results of experimental analysis are discussed.

4.1 Datasets

Datasets used in this study are extracted from LETOR website[1]. It includes various datasets that are used for learning to rank approach. LETOR contains a benchmark collection in order to facilitate research in learning to rank easier. The collections are released by Microsoft Research Asia. Besides datasets, many information, documents, papers about learning to rank approach can be found in the website.

The website contains four datasets: Microsoft Learning to Rank dataset, Yahoo! Learning to Rank Challenge dataset, LETOR 3.0 dataset and LETOR 4.0 dataset. Two of these collections are used in this research. The first one is Microsoft Bing Web Search (2015) data with 136-feature[2], and the other one is called LETOR 4.0 (2015) data with 46-feature[3]. Each datasets have common specifications. These are:

- They hold in 5-folds for cross-validation approach.
- Observations are presented as in SVM^{light}[4] format.
- Each fold contains three subsets for learning:

[1] http://research.microsoft.com/en-us/um/beijing/projects/letor/default.aspx

[2] Downloaded from http://research.microsoft.com/en-us/projects/mslr/

[3] Download from http://research.microsoft.com/en-us/um/beijing/projects/letor/letor4dataset.aspx, released in 2009

[4] http://svmlight.joachims.org/

- *Training set*: It is used to build learning model for ranking,
- *Test set*: It is used to measure the performance of models,
- *Validation set*: It is used to adjust the free parameters.

Furthermore, LETOR gives not only the collection but also describes how the document corpora and query sets in LETOR are selected, and how the documents are sampled. In addition, the method that the learning features and meta information are extracted, and the way that the datasets are partitioned for comprehensive evaluation can be found on the site. It shows how datasets can be used in different kinds of research, and there exists many papers of those research.
Liu et al. (2010) also published a paper called "*LETOR*: A benchmark collection for research on learning to rank for information retrieval" to give more details about the datasets and their collection processes.

4.1.1 Microsoft Bing Data

This dataset is a collection of Bing data search results, in which queries and associated URLs are represented by IDs. The dataset consists of feature vectors extracted from query-URL pairs along with relevancy judgement.

1. The relevancy judgements are obtained from a retrieved labelling set of a commercial web search engine (Microsoft Bing), which takes on five values from 0 (irrelevant) to 4 (perfectly relevant),
2. The features are mostly extracted by Microsoft researchers.

In the data files, each row corresponds to a query-URL pair. The first column is relevance label of the pair, the second column is the query id, and the following columns are the features. The larger value the relevance label has, the more relevant the query-URL pair is. A query-URL pair is represented by a 136-dimensional feature vector.
Each query-URL pair is represented by a 136-dimensional vector. The feature list of Microsoft Learning to Rank Datasets is given in Appendix B.2. Below are sample rows from the dataset:

```
===============================
0 qid:1 1:3 2:0 3:2 4:2 ... 135:0 136:0
4 qid:1 1:2 2:1 3:2 4:4 ... 135:0 136:1
...
3 qid:4 1:3 2:3 3:0 4:0 ... 135:0 136:0
===============================
```

The data is divided into five partitions with equal sizes, marked as S1, S2, S3, S4 and S5 for 5-fold cross-validation. Folds are given in Table 4.1.

Table4.1: 5-fold cross validation table

Folds	Training Set	Testing Set	Validation Set
1	S1, S2, S3	S4	S5
2	S2, S3, S4	S5	S1
3	S3, S4, S5	S1	S2
4	S4, S5, S1	S2	S3
5	S5, S1, S2	S3	S4

Sampling. R v.3.2.3 uses the PC memory for keeping all objects. This is a problem when the data gets bigger. To overcome this problem, representative samples are taken from those datasets used in this study. By the package called "Balanced-Sampling", balanced and spatially balanced probability samples in multi-dimensional spaces are selected with prescribed inclusion probabilities by using the cube method (Brus, 2015). It is an efficient implementation method in sampling. In order to get a fixed sample size, the inclusion probabilities, which are sum to a positive integer, are chosen as a balancing variable. Table 4.2 gives the proportion of relevancy judgements of both the population and the sample.

Table4.2: Inclusion probability results

	Relevancy Judgements				
	0	1	2	3	4
Data	0.52	0.32	0.13	0.02	0.01
Sample	0.51	0.35	0.13	0.01	1e-12

After applying these proportion, representative sample is obtained using the cube sampling method. The proportion results are shown in Fig. 4.2.

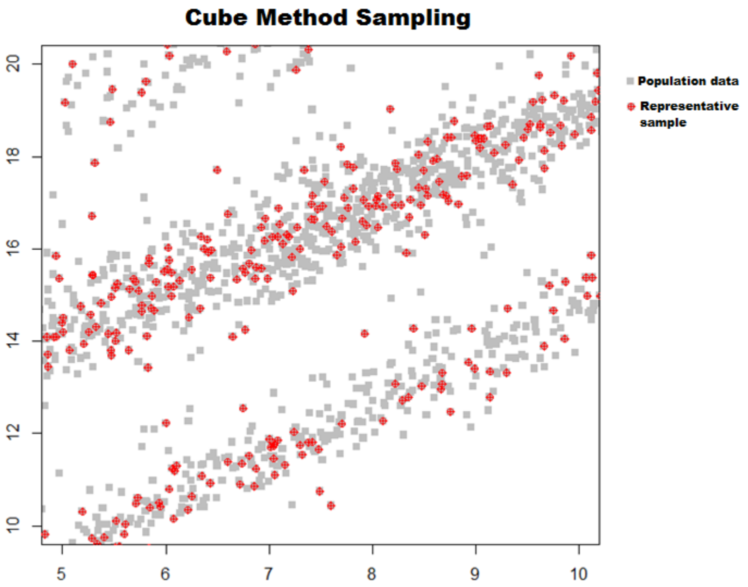

Figure 4.1: An example for a representative sample obtained by the cube method

Note that sampling is only applied to Microsoft Bing Data. Because, in LETOR 4.0, the data has fewer features than Microsoft Bing Data and its size is smaller; hence, R v.3.2.3 can perform the analysis by using all of the data without sampling. In Section 4.3, Microsoft Bing Data results both with sampling and with the original data are presented.

4.1.2 LETOR 4.0

This dataset is extracted from Million Query Track and the Gov2 web page collection. Queries are allied by documents. In this dataset, rows include observations in the form of query-document pairs.

The first column is the relevance label of this pair. Labels vary from 0 (irrelevant) to 2 (relevant). The second column is the query id of query-document pairs, and the following columns are the features, and at the end of the row, comment about the pair takes place, including id of the document. A query-document pair is denoted by a 46-dimensional feature vector. An extract from the data is given below:

```
=============================================
2 qid:10032 1:0.056537 2:0.000000 ...45:0.000000 46:0.076923
2 qid:10032 1:0.060000 2:0.000000 ...45:0.33333 46:0.069273
...
0 qid:10032 1:0.279152 2:0.000000 ...45:0.250000 46:1.000000
=============================================
```

The features have been standardized into [0, 1]. According to LETOR benchmark report written by Microsoft Asia group, normalization is done by using the following formula in Eq. (4.1).

$$\hat{x}^i_{j,k} = \frac{x^i_{j,k} - average_{1,...,N^i}(x^i_{r,k})}{max_{r=1,\ldots,N^i}(x^i_{r,k})} \tag{4.1}$$

where N^i is the size of documents, d^i_j related to i^{th} query, q_i, and $\hat{x}^i_{j,k}$ is the normalized k^{th} feature corresponding from $x^i_{j,k}$ variable of the document.

4.2 Tools

In this study, to evaluate a learning to rank model, a statistical programming tool, R v.3.2.3 (2015-12-10) – *"Wooden Christmas-Tree"* –, is used. During analysis, many packages are installed. These are listed in Appendix C.1. Furthermore, in order to construct the CMARS model, MATLAB R2015a version is used. However, due to the fact that the sizes of datasets used in this study are big, that are ~3.7G for Microsoft

Bing Dataset and $\sim$15M for LETOR 4.0 Dataset, Paralell Computing Toolbox[5] is used in order to construct CMARS models. The toolbox carries out many calculations simultaneously by splitting large sets into smaller ones, then solves them at the same time. Optimization is handled through using MOSEK v.8.0.0.52 Toolbox[6]. Moreover, for RANOVA, IBM SPSS Statistics 23.0, is used. To carry out the RANOVA, "Analyse → General Linear Model → Repeated Measures" path is utilized. Then, default configurations are applied[7].

4.3 Experiments

In the experiments conducted on Microsoft Bing Data sample and LETOR 4.0, the ANN, SVM, MARS and CMARS ranking models are developed by using the training data from each fold. Besides, in order to compare how models are affected as sample size grows, the whole Microsoft Bing Data is also analyzed. Associated validation datasets are utilized where necessary to fine tune hyper-parameters. Then, performances of the developed models are evaluated using the test dataset with respect to the performance measures.

Note that LETOR 4.0 and Microsoft Bing Data Sample experiments in the study are carried out on Samsung Ultrabook with 4.00 GB RAM, 64-bit operating system, quad-core processor using the open source statistical software R (2005). The original Microsoft Bing Data, on the other hand, is run on the MSI Computer with 8.00 GB RAM, 64-bit operating system, intel core i5.

ANN models are constructed by using the *nnet* package in R v.3.2.3 (Ripley, 2014). For this purpose, feed-forward networks with a single hidden layer is utilized. Since the response of interest is a factor, an appropriate classification network, which has one output and entropy fit and a conditional MLE is constructed. For developing the SVM ranking models, the *RSofia* package is installed to R program (King et al., 2015). The package is used for adjustment of a sofia-ml, which is a fast incremental algorithm for machine learning that can be used for training models for classification

[5] Free trial of the toolbox is downloaded from the website, https://www.mathworks.com/campaigns/products/trials.html?prodcode=DM on September 8^{th}, 2016

[6] Downloaded from the website, https://www.mosek.com/resources/downloads on September 4^{th}, 2016

[7] Free trial of the software is downloaded from the website, https://www-01.ibm.com/support/docview.wss?uid=swg24038592

or ranking. On the other side, MARS ranking models are constructed by using the *earth* package developed by Milborrow (2009). Since the response variable is ordinal categorical, the function treats factors with three or more levels in the response as multiple responses. Thus, for every relevancy judgement in both datasets, there are five models with the same basis functions with different coefficients. Minimum GCV value has been reached at different degree of interaction for each fold. MathWorks R2017a v.9.2.0.538062 performs CMARS, and MOSEK Optimization Toolbox is used to construct the model.

Note here that since datasets are in SVM^{light} format, they should be turned into a common data frame format. For that purpose, datasets are red by using `read.svm()` function in `RSofia` and then exported as a regular data format to ".txt" document. Moreover, to conduct a model for categorical data, the dependent variable is identified as factor by using `as.factor()` function while the dataset is imported to R v.3.2.3. Finally, model analysis is established via those data.

4.3.1 Microsoft Bing Data

4.3.1.1 Models

ANN models for all folds are built by using the maximum conditional likelihood fitting and SVM models have pegasos learner type which is an application of a stochastic sub-gradient method for SVM (Shalev-Shwartz et al. 2010). The only different configuration is done for building MARS and CMARS models due to the need for finding the smallest GCV criterion.

Initially, the models are computed on the sample data. All folds, except the 2^{nd} one, have the minimum GCV at 6^{th} degree. In Fold 2, it is resulted in at 4^{th} degree. However, even though Fold 1, Fold 3, Fold 4, and Fold 5 are modelled with same degree of interaction, the maximum number of terms created by the forward pass is different. The maximum term number for Fold 1 and Fold 4 is 200; where as it is 300 for the others.

The MARS models built based on the training datasets are resulted in different numbers of BFs:

- *Fold 1* has 177 terms with 56 features
- *Fold 2* has 49 basis functions, and 20 of 136 predictors
- *Fold 3* chooses 174 terms generated from 58 dependent variables
- *Fold 4* selects 100 functions with 39 features by backward pass
- *Fold 5* is modelled by 236 terms constructed 74 predictors out of 136.

On the other hand, for the original data, when the size of dataset grows, MARS models become more stable. All folds get the minimum GCV in 2^{nd} degree of interaction with 100 term number. The BFs are resulted as follows:

- *Fold 1* has 25 terms with 10 features
- *Fold 2* has 31 basis functions, and 13 of 136 predictors
- *Fold 3* chooses 31 terms generated from 14 dependent variables
- *Fold 4* selects 25 functions with 11 features by backward pass
- *Fold 5* is modelled by 21 terms constructed 9 predictors out of 136.

4.3.1.2 Performance Measures

Performances of the models developed for 5-fold by using Eq. (3.17) - Eq. (3.25) are presented in Table A.1 for the sample data and in Table A.2 for the original data. Results indicate that MARS model performance is better than CMARS and SVM even though MARS and ANN have close measures in testing data. Moreover, while the Recall and F-measure of MARS are more stable, AUC shows a better stability in the ANN model.

In addition, ROC curves are plotted in order to illustrate the performance of a binary classifier system as its discrimination threshold is varied. Here, the 0 and 1 relevance judgements are classified as irrelevant while the rest are classified as relevant. Results shows that the MARS and ANN models are more accurate models than the SVM and CMARS models. The figures that show all four models' ROC curves for each folds of the sample data and the original data are given below in Fig. 4.2 - Fig. 4.9, respectively.

For the sample data, Fig. 4.2, Fig. 4.3, Fig. 4.4, and Fig. 4.5 represent the ROC Curves for MARS, ANN, SVM, and CMARS for five folds constructed from Microsoft Bing Data sample training (right) and test (left) datasets, respectively.

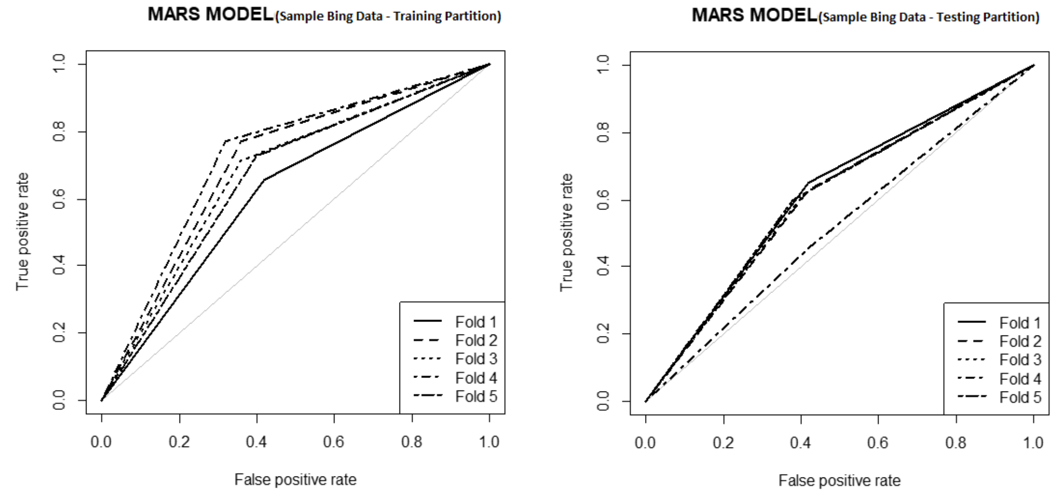

Figure 4.2: MARS Model for Sample Microsoft Bing Data

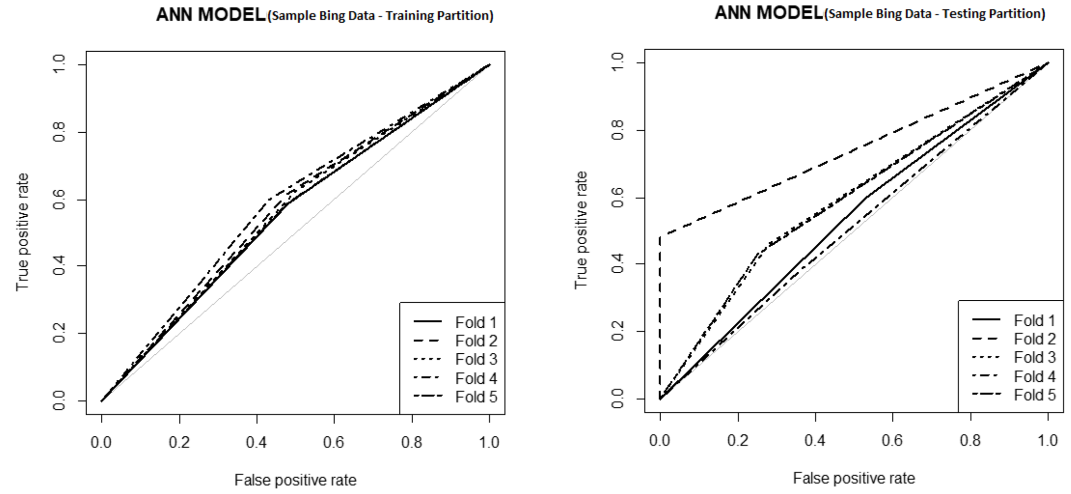

Figure 4.3: ANN Model for Sample Microsoft Bing Data

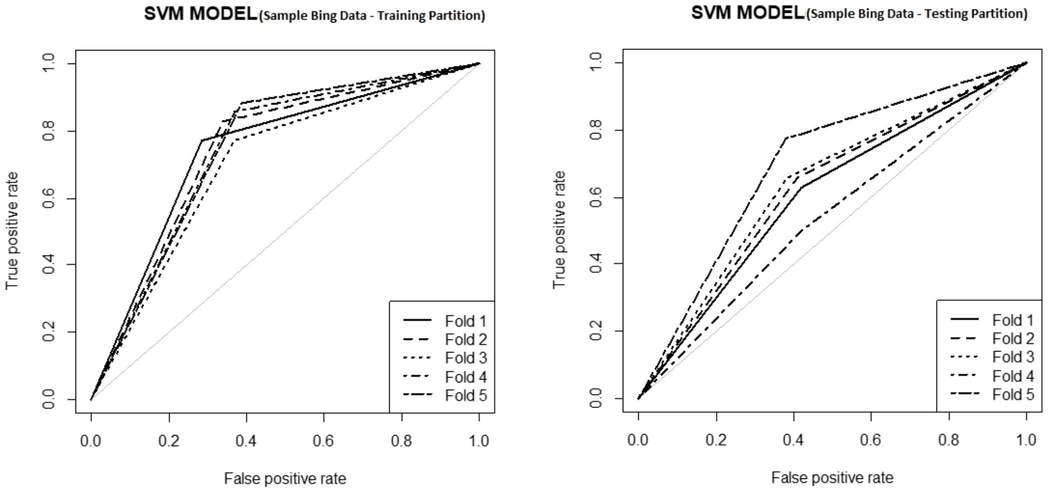

Figure 4.4: SVM Model for Sample Microsoft Bing Data

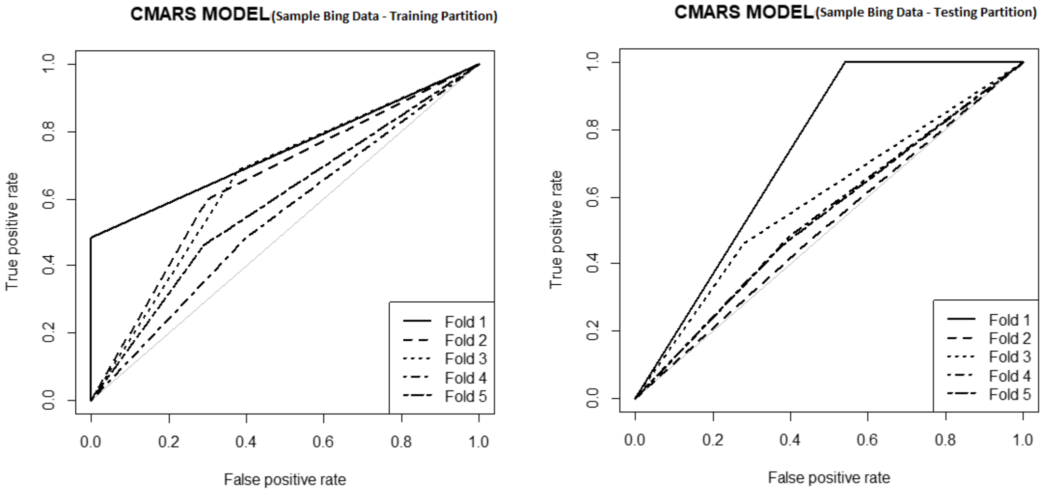

Figure 4.5: CMARS Model for Sample Microsoft Bing Data

For the original data, Fig. 4.6 - Fig. 4.9 represent the ROC Curves for MARS (1^{st} row), ANN (2^{nd} row), SVM (3^{rd} row)and CMARS (4^{th}) for five folds constructed from Microsoft Bing Data training (right) and test (left) datasets.

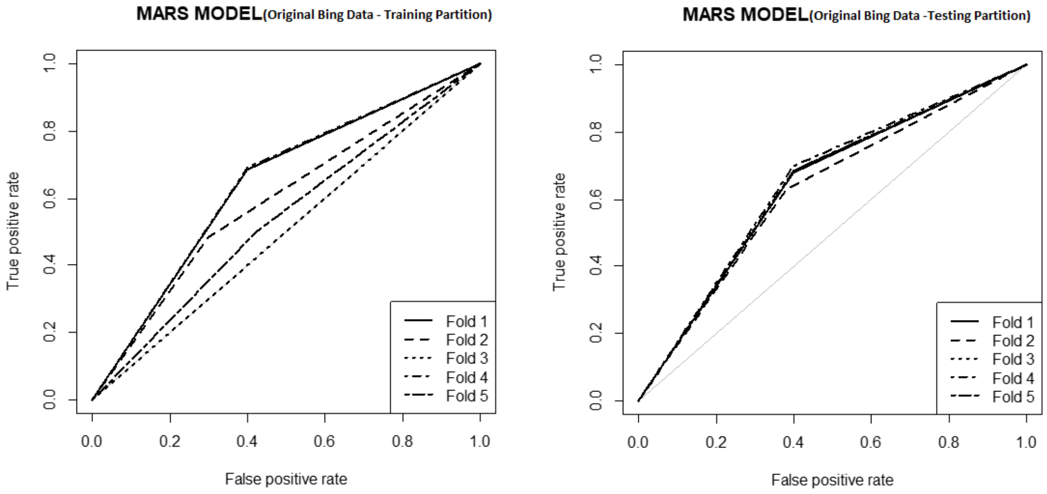

Figure 4.6: MARS Model for Microsoft Bing Data

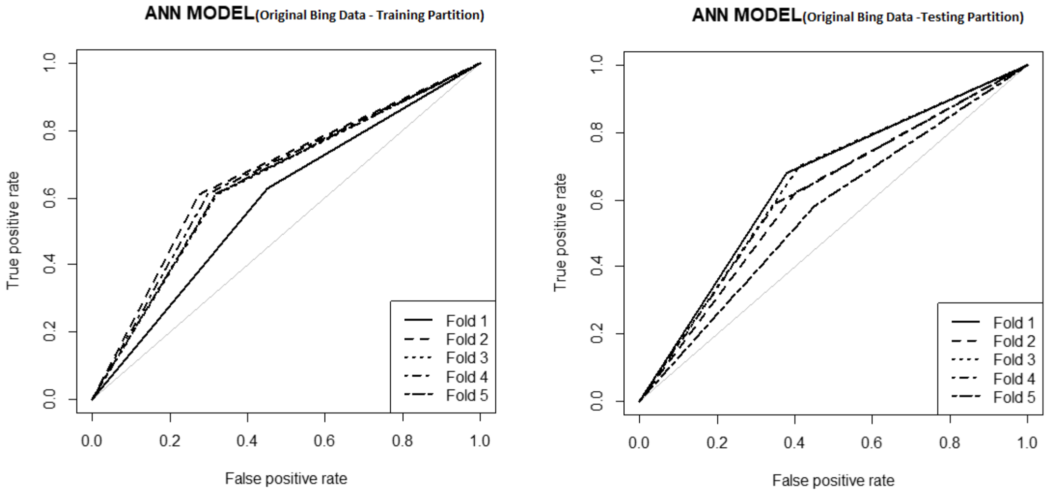

Figure 4.7: ANN Model for Microsoft Bing Data

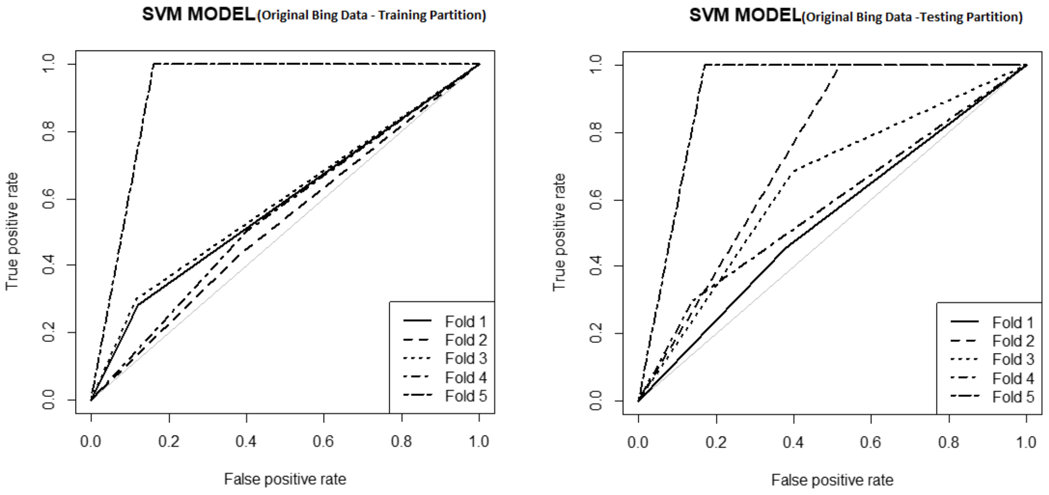

Figure 4.8: SVM Model for Microsoft Bing Data

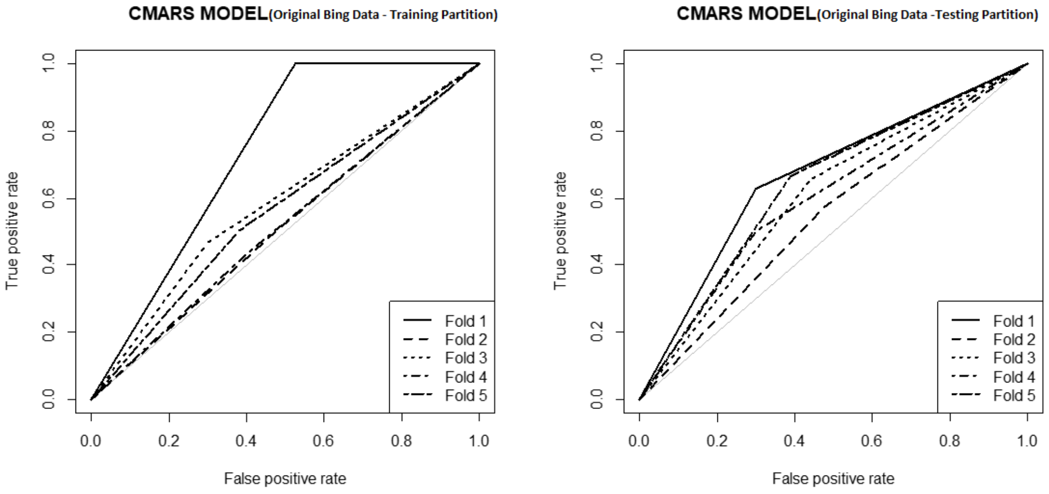

Figure 4.9: CMARS Model for Microsoft Bing Data

4.3.1.3 Repeated measures ANOVA

In order to test if there are statistically significant difference between the compared methods, repeated analysis of variance (RANOVA), described in Section 3.6 is performed for each measures with the following hypotheses:

$$H_0 : \mu_{ANN} = \mu_{SVM} = \mu_{MARS} = \mu_{CMARS}$$

vs.

H_1 : *At least one is different.*

After the assumptions are satisfied, the means of performance measures are computed by SPSS. The results are in Table 4.3 and in Table 4.4 for the original data while Table 4.5 and Table 4.6 show the outcomes for the sample data. Note here that RANOVA is applied on training and testing datasets as well as stabilities separately; p-values of significant test results are indicated within the parentheses while the greater mean results are indicated by "*". The best performances without p-value mean that test outcome is not significant. The original data outcomes are as follows:

- For training dataset:
  - ANN performs better with respect to Rc, f-measure and g-measure
  - MARS performs better with respect to P, accuracy, AUC and J
  - MARS model gives more accurate result in both training and testing datasets
  - SVM and CMARS are better only with respect to fall-out and specificity, respectively
  - ANN and MARS methods are more robust than the others with respect to standard deviations of the measures
- For testing dataset:
  - MARS outperforms with respect to the most of the performance measures (six out of nine measures).
  - SVM performs better with respect to P and fall-out

  - MARS and CMARS seem to be more robust methods than the baseline methods, SVM and ANN, according to the standard deviation of five measure out of nine ones.

- For stability:
  - MARS model has the most stable measures with respect to means and the standard deviations
  - CMARS is more stable than ANN and SVM regarding means of P, f-measure and accuracy
  - The standard deviations of P, f-measure, g-measure and J, calculated for ANN model, are more stable than the MARS model and the SVM model.

The outcomes of the sample data is given as follows:

- For training dataset:
  - SVM gives better result with respect to f-measure, g-measure and accuracy
  - CMARS is better than the other models considering P, specificity and AUC
  - MARS is only better with respect to J measure
  - ANN is the most robust method with respect to the standard deviations of the most of the performance measures (six out of nine ones)
  - CMARS is more robust than the MARS model and the SVM model regarding the standard deviations of specificity, AUC and fall-out.
- For testing dataset:
  - ANN outperforms according to four performance measures out of nine ones.
  - CMARS is better considering specificity, AUC and J
  - SVM and MARS perform better only for accuracy and P, respectively
  - ANN is the most stable model because five measures out of nine ones of it have minimum standard deviations.

  - SVM shows robustness for only P measure.

- For stability:

  - ANN and CMARS models are more stable than SVM model and MARS model
  - SVM model is more stable than the other models with respect to the standard deviations of performance measures.
  - MARS is second better model with respect to the standard deviations.

Table4.3: Means of performance measure belongs to Microsoft Bing 5-fold training data and testing data to test RANOVA (original data)

Performance Measures	Training				Testing				Stability			
	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
precision	0.598	0.581	0.660* (*0.05*)	0.487	0.682	0.705*	0.696	0.568	0.876	0.825	0.949*	0.857
recall	0.805*	0.709	0.663	0.366	0.599*	0.446	0.596	0.385	0.743	0.628	0.899	0.951*
f-measure	0.671* (*0.013*)	0.622	0.632	0.384	0.556	0.446	0.587*	0.409	0.829	0.717	0.929	0.939*
g-measure	0.686*	0.634	0.646	0.401	0.610	0.497	0.614*	0.439	0.889	0.784	0.950*	0.913
Accuracy	0.593	0.598	0.625* (*0.043*)	0.497	0.564	0.575	0.598* (*0.014*)	0.511	0.952	0.962	0.956	0.971*
Specificity	0.353	0.346	0.760	0.841*	0.340	0.428	0.845*	0.786	0.962*	0.809	0.899	0.934
AUC	0.632	0.485	0.651*	0.517	0.638	0.552	0.648* (*0.02*)	0.556	0.991	0.879	0.995*	0.930
Fall-out	0.613	0.654* (*0.029*)	0.154	0.170	0.612	0.656* (*0.011*)	0.155	0.217	0.998*	0.996	0.997	0.785
Youden's J	0.159	0.056	0.423* (*0.008*)	0.207	-0.062	-0.126	0.441* (*0.041*)	0.171	-2.578	-2.246	0.959*	0.825

Note that * indicates the best performance; statistically significant difference are associated with a p-value in parenthesis.

Table4.4: Standard deviations of performance measure belongs to Microsoft Bing 5-fold training data and testing data to test RANOVA (original data)

Performance Measures	Training				Testing				Stability			
	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
precision	0.075*	0.132	0.078	0.106	0.044*	0.095	0.085	0.153	0.580*	0.718	0.910	0.693
recall	0.173	0.151*	0.226	0.221	0.361	0.388	0.322	0.217*	0.478	0.389*	0.703	0.981
f-measure	0.018*	0.098	0.090	0.187	0.204	0.288	0.203	0.111*	0.086*	0.340	0.444	0.593
g-measure	0.033*	0.092	0.083	0.164	0.204	0.240	0.174	0.087*	0.161*	0.382	0.474	0.529
Accuracy	0.059	0.028	0.004*	0.059	0.059	0.035	0.041	0.034*	0.997	0.785	0.102*	0.581
Specificity	0.302	0.149	0.192	0.111*	0.299	0.320	0.009*	0.159	0.989	0.464	0.046*	0.698
AUC	0.030	0.303	0.006*	0.025	0.023	0.273	0.003*	0.064	0.767	0.901	0.546	0.393*
Fall-out	0.269	0.148	0.003*	0.111	0.268	0.151	0.009*	0.163	0.998	0.984	0.375*	0.683
Youden’s J	0.130*	0.179	0.231	0.297	0.504	0.318	0.322	0.297*	0.259*	0.564	0.716	0.999

Note that * indicates the best performance; statistically significant difference are associated with a p-value in parenthesis.

Table4.5: Means of performance measure belongs to Microsoft Bing 5-fold training data and testing data to test RANOVA (sample data)

Performance Measures	Training				Testing				Stability			
	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
precision	0.528	0.653	0.661	0.747*	0.532	0.587	0.611*	0.550	0.805	0.899	0.924*	0.736
recall	0.994* (*0.028*)	0.899	0.849	0.287	0.993* (*0.013*)	0.824	0.728	0.211	0.855	0.917*	0.857	0.734
f-measure	0.689	0.754*	0.742	0.187	0.692*	0.684	0.660	0.170	0.933*	0.908	0.890	0.911
g-measure	0.724	0.765*	0.748	0.225	0.726*	0.695	0.665	0.203	0.971*	0.909	0.888	0.902
Accuracy	0.530	0.691*	0.689	0.475	0.533	0.602*	0.596	0.470	0.774	0.871	0.865	0.990*
Specificity	0.014	0.457	0.613	0.999* (*0.007*)	0.014	0.338	0.422	0.997* (*0.002*)	0.023	0.741	0.688	0.998*
AUC	0.658	0.737	0.708	0.747*	0.701	0.616	0.595	0.752* (*0.001*)	0.991	0.836	0.840	0.993*
Fall-out	0.986* (*0.004*)	0.543	0.494	0.001	0.992* (*0.002*)	0.662	0.578	0.026	0.994*	0.821	0.854	0.031
Youden's J	0.008	0.355	0.463*	0.287	0.007	0.162	0.149	0.208* (*0.037*)	0.805*	0.456	0.323	0.726

Note that * indicates the best performance; statistically significant difference are associated with a p-value in parenthesis.

Table4.6: Standard deviations of performance measure belongs to Microsoft Bing 5-fold training data and testing data to test RANOVA (sample data)

Performance	Training				Testing				Stability			
Measures	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
precision	0.016*	0.041	0.045	0.387	0.036	0.023*	0.053	0.440	0.813	0.061	0.687	0.053*
recall	0.006*	0.064	0.060	0.407	0.013*	0.097	0.086	0.303	0.226	0.238	0.157*	0.319
f-measure	0.015*	0.019	0.036	0.309	0.034*	0.047	0.034*	0.222	0.949	0.153*	0.994	0.212
g-measure	0.013*	0.019	0.035	0.298	0.029*	0.052	0.036	0.211	0.826	0.174*	0.805	0.246
Accuracy	0.014*	0.037	0.048	0.019	0.037	0.031	0.022*	0.037	0.776	0.613	0.586*	0.837
Specificity	0.016	0.149	0.180	0.001*	0.017	0.137	0.091	0.004*	0.097	0.008*	0.191	0.028
AUC	0.085	0.019	0.041	0.017*	0.093	0.056	0.027	0.025*	0.437	0.302	0.291*	0.441
Fall-out	0.016	0.149	0.180	0.001*	0.018*	0.137	0.091	0.050	0.099	0.008*	0.196	0.365
Youden's J	0.010*	0.091	0.192	0.406	0.019*	0.073	0.050	0.301	0.101*	0.180	0.388	0.242

Note that * indicates the best performance; statistically significant difference are associated with a p-value in parenthesis.

4.3.1.4 Repeated measures ANOVA for size effect

Due to the fact that the Microsoft Bing Data has been studied for different size in this thesis, we analyze if there exist a size effect on the performance measures of those four methods. The null hypothesis is as follows:

$$H_0 : \mu_{original} = \mu_{sample}$$

$$\text{vs.}$$

$$H_1 : \mu_{original} \neq \mu_{sample}$$

The analysis indicates that the 5% level of significance, the null hypothesis is failed to reject with p-value 0.3. That means, there is no difference between original dataset and sample dataset. Therefore, the size of the dataset has no effect on the performance measure analysis.

4.3.2 LETOR 4.0

4.3.2.1 Models

In this data analysis, ANN and SVM models are constructed by using the same configurations with Microsoft Bing Data (original data). In other words, an SVM model with primal estimated sub-gradient solver (pegasos) and ANN by the maximum likelihood fitting are constructed.

MARS model, on the other hand, shows a different characteristics than the other dataset. All folds have six degree of interaction and 200 forward pass terms when they reach the lowest GCV. However, as for Microsoft Bing Data, models are computed with different number of BFs consisting of different features.

- *Fold 1* has 133 terms and 34 features.
- *Fold 2* has 114 BFs constructed by using 31 predictors out of 46.
- *Fold 3* picks 113 terms with 33 independent variables.
- *Fold 4* selects 134 functions with 36 features by backward pass.

- *Fold 5* is modelled by the largest BF number which contains 163 terms and 33 of 46 predictors.

4.3.2.2 Performance Measures

Evaluated measures for both training and testing data of 5-fold and their stabilities are given in the Table A.3. Table A.3 states the following results:

- In fold 1, training dataset comparison states that the MARS model has the highest precision score. In order words, the MARS model has the highest number of correctly predicted positive elements with respect to the total predicted positive observations. Besides, the model is 83% and 81% accurate for ANN and SVM, respectively. G-measure which is higher for the MARS model indicates a greater similarity between the partitions. However, the fact that having J score closer to 0 makes the ANN a weak model. On the other hand, SVM has a higher J value. In the ranking of positive test score, MARS model and CMARS model takes third and second places. As for the evaluation on test datasets, MARS again has the highest positive prediction score. However, when the other measurements are considered, the MARS model turns out to be the second while ANN model gets better behaviour.
- SVM and CMARS again perform worse than the other two techniques.
- MARS model's accuracy is also 83% for the datasets that belong to Fold 2, and Fold 3.
- Training data performances show that MARS has better prediction performance than the other models according to the other performance measures.
- Unlike first three folds of datasets, the MARS model has the best scores within all measures except recall and fall-out in last two datasets.

Therefore, it can be concluded that the MARS model suits well to this datasets. Moreover, ROC curves of MARS (1^{st} row), ANN (2^{nd} row), SVM (3^{rd} row) and CMARS (4^{th}) constructed from LETOR 4.0 training data (right) and test data (left) are given in Fig. 4.10, Fig. 4.11, Fig. 4.12, and Fig. 4.13.

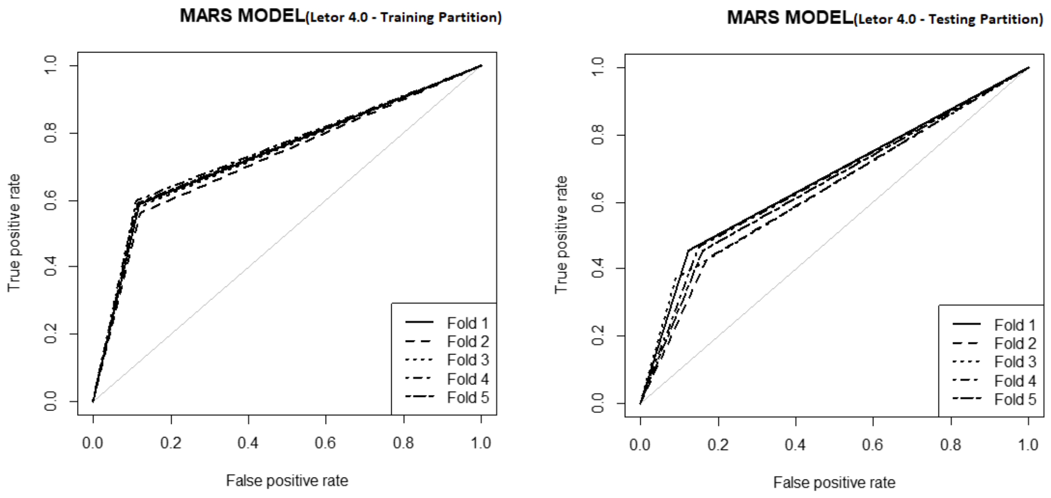

Figure 4.10: MARS Model for LETOR 4.0 Data

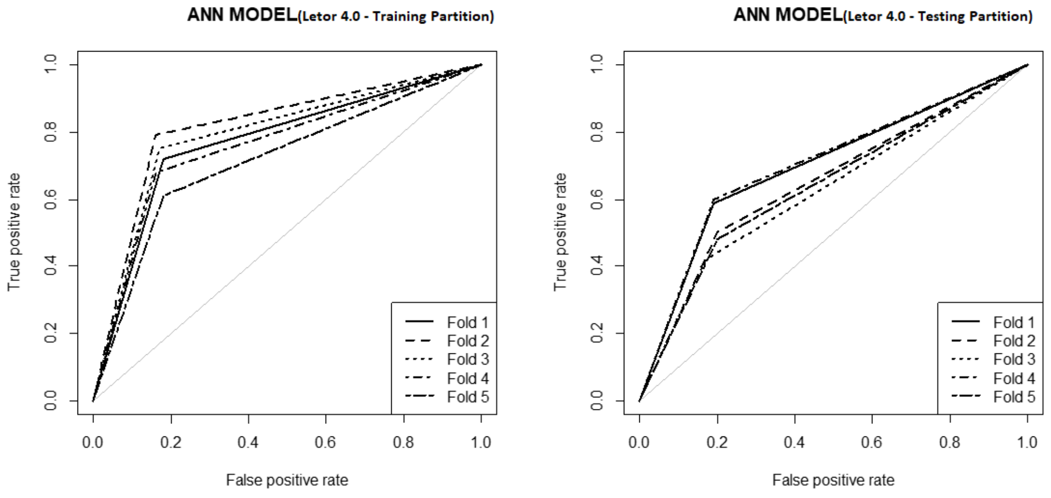

Figure 4.11: ANN Model for LETOR 4.0 Data

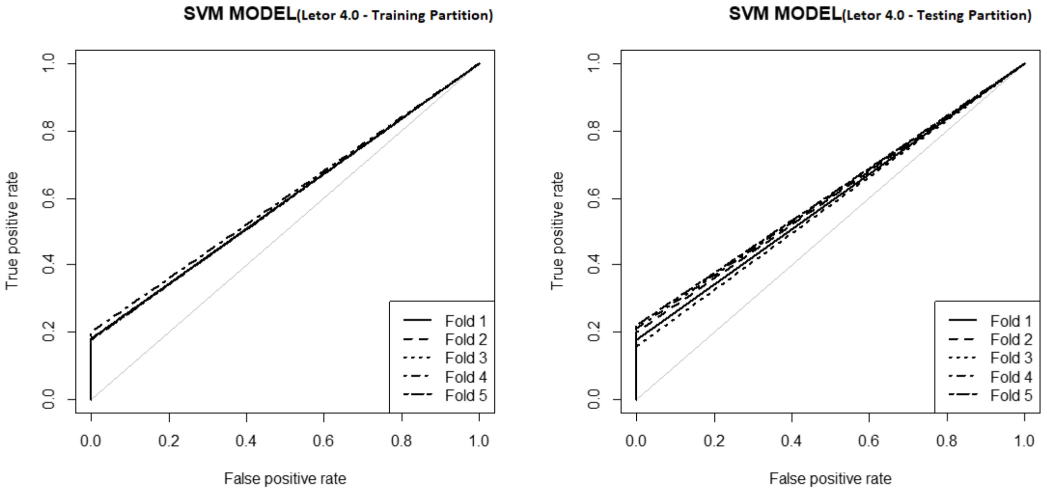

Figure 4.12: SVM Model for LETOR 4.0 Data

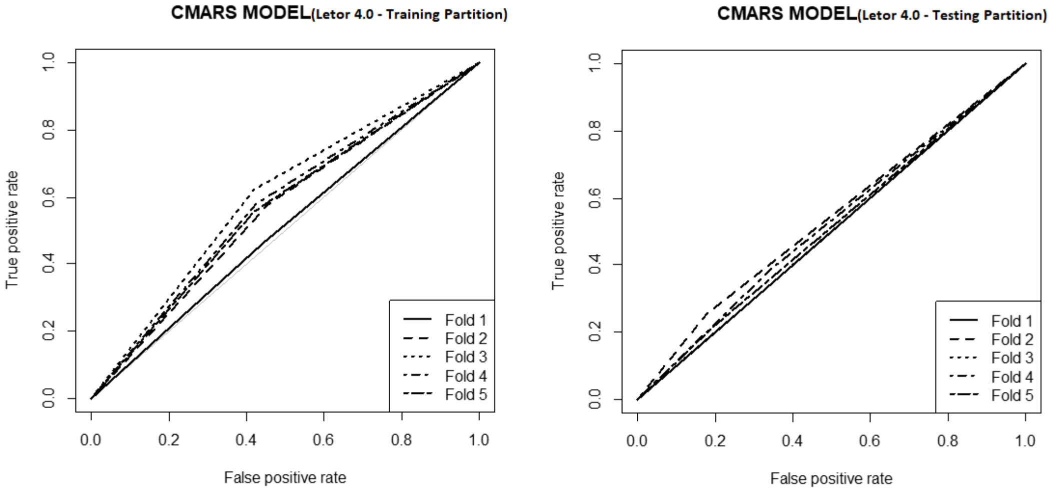

Figure 4.13: CMARS Model for LETOR 4.0 Data

4.3.2.3 Repeated measures ANOVA

For those models with several measures, RANOVA is applied again to test the following hypothesis:

$$H_0 : \mu_{ANN} = \mu_{SVM} = \mu_{MARS} = \mu_{CMARS}$$
$$\text{vs.}$$
$$H_1 : \textit{At least one is different}$$

Means of the performance measures used for RANOVA test are given in Table 4.7. The outcome of the tests is as follows:

- For training dataset:
  - MARS outperforms with respect to four performance measures out of nine
  - ANN is better according to g-measure and AUC
  - SVM is better considering Rc and fall-out
  - CMARS is only better with respect to specificity
  - ANN, MARS and SVM are the robust models with respect to the standard deviations
- For testing dataset:
  - ANN outperforms according to five performance measures out of nine.
  - MARS is better considering precision, P and J score
  - SVM and CMARS perform better only for fall-out and specificity, respectively
  - ANN, MARS and SVM are the robust models with respect to the standard deviations.
- For stability:
  - ANN and SVM are more stable than novel methods regarding the performance measures.

- SVM model is the most stable than the other models with respect to the standard deviations of performance measures.

Table4.7: Means of performance measure belongs to LETOR 4.0 5-fold training data and testing data to test RANOVA

Performance Measures	Training				Testing				Stability			
	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
precision	0.820	0.807	0.885* (*0.010*)	0.441	0.814	0.805	0.852* (*0.007*)	0.500	0.993	0.998*	0.963	0.881
recall	0.991	1.000*	0.917	0.691	0.988*	0.892	0.889	0.516	0.997*	0.892	0.970	0.746
f-measure	0.897	0.893	0.901*	0.507	0.892*	0.830	0.870	0.470	0.994*	0.930	0.966	0.927
g-measure	0.901*	0.898	0.900	0.535	0.896*	0.839	0.870	0.488	0.995*	0.935	0.966	0.912
Accuracy	0.817	0.807	0.837* (*0.007*)	0.601	0.808*	0.805	0.786	0.508	0.989	0.998*	0.940	0.845
Specificity	0.086	0.139	0.499	0.557*	0.065	0.147	0.364	0.504*	0.751	0.946*	0.728	0.905
AUC	0.753*	0.597	0.737	0.518	0.687*	0.597	0.646	0.554	0.913	0.998*	0.876	0.935
Fall-out	0.914	1.000*	0.501	0.302	0.935	1.000*	0.636	0.350	0.977	1.000*	0.787	0.861
Youden's J	0.053	0.039	0.253* (*0.0001*)	0.019	0.077	0.139	0.416* (*0.004*)	0.248	0.685*	0.282	0.608	0.077

Note that * indicates the best performance; statistically significant difference are associated with a p-value in parenthesis.

Table4.8: Standard deviations of performance measure belongs to LETOR 4.0 5-fold training data and testing data to test RANOVA

Performance	Training				Testing				Stability			
Measures	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
precision	0.005	0.010	0.004*	0.116	0.027	0.024*	0.026	0.258	0.176	0.413	0.154*	0.450
recall	0.005	0.001*	0.008	0.211	0.006*	0.241	0.010	0.344	0.803	0.004*	0.817	0.613
f-measure	0.003*	0.006	0.004	0.071	0.014	0.137	0.012*	0.248	0.223	0.044*	0.318	0.286
g-measure	0.003*	0.006	0.004	0.068	0.013	0.129	0.012*	0.251	0.243	0.045*	0.321	0.271
Accuracy	0.005	0.010	0.004*	0.182	0.024	0.025	0.017*	0.156	0.197*	0.406	0.248	0.855
Specificity	0.058	0.311	0.044*	0.209	0.037*	0.329	0.057	0.231	0.638*	0.946	0.773	0.906
AUC	0.025	0.005*	0.009	0.062	0.046	0.012*	0.015	0.018	0.555	0.409	0.619	0.292*
Fall-out	0.058	0.001*	0.044	0.340	0.037	0.001*	0.057	0.329	0.638	0.500*	0.773	0.968
Youden's J	0.053	0.311	0.037*	0.175	0.031*	0.088	0.048	0.366	0.592	0.282*	0.754	0.479

Note that * indicates the best performance; statistically significant difference are associated with a p-value in parenthesis.

CHAPTER 5

CONCLUSION

With the usage of WWW, IR has got a great significance in computer science. Ranking is one of the imported problems in this field. Building a model for ranking can be considered in three categories, which are traditional models, LMs, and the system models. The traditional models usually are applications of heuristics, learning from practical experience. In other respects, the LMs are studied for specific representation such as speech recognition, machine translation. The last category, on the other hand, aims to build machine learning models to get “true” rank. In this study, widely known models of the third category, namely ANN and SVM, are compared to two new approach, MARS and CMARS.

The objective of this thesis work is to investigate how MARS and CMARS models are adequate on learning to rank problem under IR and how it is affected by different sizes of data. The abovementioned models are applied two different large datasets, Bing Data and LETOR 4.0. The first argument that is whether MARS and CMARS model can fit for ranking in IR is discussed by evaluating the performance measures on the aforementioned datasets. In order to analyze the effect of the data set on the ranking models, we compared the performances for the original Microsoft Bing Data and the sample extracted from it. A balanced sample from Bing data is selected with prescribed inclusion probabilities.

Both datasets, Microsoft Bing Data and LETOR 4.0, have five different folds in which training dataset and testing dataset are included. In addition, four models of each dataset, which belongs to those collections and Microsoft Bing sample, are studied in this thesis. In other words, totally 120 models are built. What’s more, nine performance measures, described in Chapter 3 are calculated for those models, please refer to Table A.1, Table A.2 and Table A.3 for the results.

Once all models are created and their performance measures are calculated, RANOVA are applied to test if there exists a significant difference between models with respect to performance measures.

Results of both collections show that although performance scores are close to each other, the MARS model has better behaviour. It gives highly promising results for prediction. However, AUC and ROC curves present a better performance for ANN models. Another striking point is that the SVM model has the highest recall and fall-out scores in all folds. Specificity score is the highest for the CMARS model. Yet, the score has negative aspect, it can be said that among all comparisons, CMARS may not give the expected performance in this study. The backward stage eliminates few basis functions in MARS models. For this reason, the CMARS model may be over-fitted and not perform as the MARS does. Furthermore, analyses of repeated measures states that in Microsoft Bing Data, both μ_{ANN} has significant difference among others regarding many measures while in LETOR 4.0, the MARS model gives outstanding results with its mean value, μ_{MARS}.

In conclusion, the MARS model has been shown an effective result in ranking approach. The system performs promisingly on both Microsoft Bing and LETOR 4.0 data. Even though give different performance on differently sized retrieval sets, size does not affect on performances of models. Yet, it can be said that MARS can be used in many field of IR. Conversely, CMARS performance is overcome by the others, especially MARS. The fact that backward pass may have strong effect on CMARS in IR analysis could be the explanation for this. However, a different regularization method may be used as a potential method. Besides, the pace of learning should also be investigated by modelling different rank-learning techniques. In other respects, another study to increase the speed of learning should also be investigated.

REFERENCES

[1] Allan, J.(2006). Lecture on *Boolean Retrieval*, Information Retrieval.Personal Collection of J. Aslam,Northeastern University, College of Computer and Information Science. Cited on http://www.ccis.northeastern.edu/home/jaa/CSG339.06F/Lectures/boolean.pdf on November 10^{th}, 2015.

[2] Arens, R.J. (2009): Learning to rank documents with support vector machines via active learning, Iowa Research Online.

[3] Benediktsson, J. A., Swain, P. H. and Ersoy, O. K.: Neural Network Approaches Versus Statistical Methods in Classification of Multisource Remote Sensing Data, IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, VOL 28, NO 4, JULY 1990. Cited from http://ieeexplore.ieee.org/document/572944/ on July 30^{th}, 2016.

[4] Bhowmik, A., Ghosh, J.(2017): LETOR Methods for Unsupervised Rank Aggregation. Published in WWW '17 Proceedings of the 26th International Conference on World Wide Web, pp.1331-1340. Retrieved from https://dl.acm.org/citation.cfm?doid=3038912.3052689 on July 17^{th}, 2018.

[5] Brin, S., Page, L., Motwami, R. and Winograd, T.: The PageRank citation ranking: Bringing order to the Web. Technical Report 1999-0120, Computer Science Department, Stanford University, 1999

[6] Burges, C., Shaked, T., and Renshaw, E.(2005): Learning to rank Using Gradient Descent. Appearing in Proceedings of the 22nd International Conference on Machine Learning, Bonn, Germany.

[7] Brus, D. J. (2015). Balanced sampling: A versatile sampling approach for statistical soil surveys. Geoderma, 253, 111-121.Retrieved from https://www.sciencedirect.com/science/article/pii/S0016706115001238?via%3Dihub on July 16^{th}, 2016.

[8] Cao, Y. B., Xu, J., Liu, T. Y., Li, H., Huang, Y. L., & Hon, H. W. (2006): Adapting ranking SVM to document retrieval. Proceedings of SIGIR 2006 (pp. 186–193)

[9] Cao, Z., Qin, T., Liu, T.Y., Tsai, M.F. and Li, H.(2007): Learning to rank—from pairwise approach to listwise approach. Proceedings of the 24th international conference on Machine learning, pp. 129-136.

[10] Ceri, S., Bozzon, A., Brambilla, M., Della Valle, E., Fraternali, P., and Quarteroni, S.(2013): Web Information Retrieval. Published in: Search Computing: Challenges and Directions, LNCS 5950, 2010; Search Computing: Trends and Developments, LNCS 6585, 2011; and Search Computing: Broadening Web Search, LNCS 7358.

[11] Chen, Stanley F. and Goodman, J.(1998): An Empirical Study of Smoothing Techniques for Language Modeling.

[12] Cornish,R.(2007): Lecture on Analysing repeated measures data. Collection of Mathematics Learning Support Center. Cited from http://www.statstutor.ac.uk/resources/uploaded/repeated-measures.pdf on July 24^{th}, 2018.

[13] Crowder, M. (1990). Analysis of Repeated Measures. New York: Routledge.

[14] Dammak, F., Gabsi, I., Kammoun, H. & Hamadou, A.B.(2015):Active Learning to Rank Method for Documents Retrieval. *ICIW 2015 : The Tenth International Conference on Internet and Web Applications and Services*. Retrieved from https://www.researchgate.net/profile/Faiza_Dammak2/publication/289130180_Active_Learning_to_Rank_Method_for_Documents_Retrieval/links/5689a58108ae1e63f1f8fd7a/Active-Learning-to-Rank-Method-for-Documents-Retrieval.pdf on July 24^{th}, 2018.

[15] Davies, C.S. (2003): Statistical Methods for the Analysis of Repeated Measures. Springer-Verlag, New York.

[16] Field, A.(2012):Repeated Measures ANOVA. Personel Collection of A. Field, University of Ottawa- The Faculty of Health Sciences. Cited on http://health.uottawa.ca/biomech/courses/apa6101/Repeated%20Measures%20ANOVA.pdf on July 24^{th}, 2018.

[17] Freund, Y., Iyer, R. D., Schapire, R. E. and Singer,Y. (2003): An efficient boosting algorithm for combining preferences. Journal of Machine Learning Research, 4:933-969.

[18] Friedman, J. H.(1991): Multivariate Adaptive Regression Splines. The Annals of Statistics.

[19] Gillies J. and Cailliau R.: How the Web Was Born: The Story of the World Wide Web. Oxford University Press, 2000.

[20] Grafström, A., Lisic, J. (2016). BalancedSampling: Balanced and Spatially Balanced Sampling. Retrieved from http://www.antongrafstrom.se/balancedsampling on July 16^{th}, 2016.

[21] Hastie, T., Tibshirani, R., Friedman, J., and Franklin, J.(2009): The elements of statistical learning— data mining, inference and prediction. The Mathematical Intelligencer, 27(2), 83-85.

[22] IBM SPSS Statistics 23.0

[23] Ibrahim, M.& Murshed, M.(2016): From Tf-Idf to Learning-to-Rank: An Overview.Handbook of Research on Innovations in Information Retrieval, Analysis, and Management, pp. 62-109. Retrieved from https://www.igi-global.com/chapter/from-tf-idf-to-learning-to-rank/137475 on July 18^{th}, 2018.

[24] Kaipio, J.P.(2005): Classical Regularization Methods. In: Statistical and Computational Inverse Problems. Applied Mathematical Sciences, vol 160,pp 7-48.

[25] King, M., Diaz, F.C.(2015). Rsofia: Sofia-ml models for ranking. Retrieved from https://cran.r-project.org/web/packages/RSofia/index.html on February 4^{th}, 2016.

[26] Kwok, K.L. (1989): A Neural Network for Probabilistic Information Retrieval, Dept. of Mathematics and Computer Science, Western Connecticut. Cited from https://dl.acm.org/citation.cfm?id=75338 on July 30^{th}, 2016.

[27] Langville, A.N. & Meyer, C.D: Google's PageRank and Beyond. Princeton University Press, July 1, 2011.

[28] Li, H.(2014): Learning to Rank for Information Retrieval and Natural Language Processing, Second Edition, Morgan & Claypool Publishers.

[29] Li, P., Wu, Q., and Burges, C. J.: Mcrank— Learning to rank using multiple classification and gradient boosting. In Advances in neural information processing systems,pp. 897-904, 2007.

[30] Lim, D. K. (2016): Learning to Rank for Retrieval and Recommendation. UC San Diego. Retrieved from https://escholarship.org/uc/item/5bm9g2n1 on April 10^{th}, 2017.

[31] Liu, T.Y. (2009): Learning to Rank for Information Retrieval, Foundations and Trends® in Information Retrieval: Vol. 3: No. 3, pp 225-331.

[32] Lunardon, N., Menardi, G. and Torelli, N.(2014): ROSE: A Package for Binary Imbalanced Learning, The R Journal Vol. 6/1. Cited from https://journal.r-project.org/archive/2014/RJ-2014-008/RJ-2014-008.pdf on August 12^{th}, 2017.

[33] Manning, C. D., Raghavan, P. and Schütze, H.(2008): Introduction to Information Retrieval, Cambridge University Press.

[34] MATLAB Version 7.5 (R2007b)

[35] Marshall, E. Lecture on Repeated measures (within-subjects) ANOVA. Collection of University of Sheffield. Retrieved from https://www.sheffield.ac.uk/polopoly_fs/1.531222!/file/MASH_repeated_measures_ANOVA_SPSS.pdf on July 24^{th}, 2018.

[36] McCartney, B.(April, 2005): NLP Lunch Tutorial: Smoothing. Retrieved from http://nlp.stanford.edu/~wcmac/papers/20050421-smoothing-tutorial.pdf on December 21^{st}, 2015.

[37] McGonagle, J.: Feed-forward neural networks. Brillant.org. Retrieved from https://brilliant.org/wiki/feedforward-neural-networks/ on January 10^{th}, 2017.

[38] Milborrow, S. (2009). earth: Multivariate Adaptive Regression Spline Models. R Software Package. Retrieved from http://cran.r-project.org/web/packages/earth/index.html on April 1^{st}, 2016.

[39] Mokris, I. and Skovajsova,L. (2005): Neural Network Model Of System For Information Retrieval From Text Documents In Slovak Language. Proceeding of Acta Electrotechnica et Informatica No. 3, Vol. 5.

[40] MOSEK ApS: MOSEK Modeling Cookbook Release 3.0. Cited from https://docs.mosek.com/MOSEKModelingCookbook-a4paper.pdf on May 15^{th}, 2018.

[41] MOSEK, A very powerful commercial software for CQP. Available at http://www.mosek.com

[42] Nisbet, R., Elder, J., & Miner, G. (2009): Handbook of Statistical Analysis and Data Mining Applications, 1st Edition. Academic Press, cited on https://doi.org/10.1016/B978-0-12-374765-5.00013-9 on November 25^{th}, 2015.

[43] Pei, J.: Information Retrieval and Web Search – Language Models(PowerPoint Slide). Retrieved from http://www.cs.sfu.ca/CourseCentral/456/jpei/web%20slides/L18%20-%20Language%20models.pdf on September 16^{th}, 2015.

[44] Qin, T. and Liu, T.: Introducing LETOR 4.0 Datasets. Cited from http://research.microsoft.com/en-us/um/beijing/projects/letor/letor4download on July 30^{th}, 2016.

[45] Qin, T., Liu, T.Y.(2008): Microsoft Bing Web Search. Retrieved from http://research.microsoft.com/en-us/projects/mslr/ on August 25^{th}, 2016.

[46] Qin, T., Liu, T.Y.(2008): MQ2008. Retrieved from https://www.microsoft.com/en-us/research/project/letor-learning-rank-information-retrieval/?from=http%3A%2F%2Fresearch.microsoft.com%2Fen-us%2Fum%2Fbeijing%2Fprojects%2Fletor%2F#!letor-4-0 on on August 25^{th}, 2016.

[47] R Development Core Team (2009). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. Retrieved from http://www.R-project.org

[48] Rigutini, L., Papini, T., Maggini, M. and Bianchini, M.: A neural network approach for learning object ranking https://www.researchgate.net/

profile/Leonardo_Rigutini/publication/215552260_A_neural_network_approach_for_learning_object_ranking/links/00b49518ca3e56c852000000/A-neural-network-approach-for-learning-object-ranking.pdf on July 30^{th}, 2016.

[49] Ripley, B. (2014). nnet: feed-forward neural networks. Retrieved from https://cran.r-project.org/web/packages/nnet/index.html on May 5^{th}, 2016.

[50] Soboroff,I.(2001). Lecture on *Information Retrieval*. Personal Collection of I. Soboroff, University of Maryland - Baltimore County. Cited from https://www.csee.umbc.edu/~ian/ on November 10^{th}, 2015.

[51] Sokolova, M., and Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. Information Processing and Management, 45(4), 427-437.

[52] Stone, C., Hansen, M., Kooperberg, C. and Truong, Y. (1997). Polynomial splines and their tensor products (with discussion),Annals of Statistics 25(4): 1371–1470. Retrieved from https://www.jstor.org/stable/2959054?seq=1#page_thumbnails_tab_contents on May 20^{th}, 2015.

[53] Şen, S.(2016). Lecture on REPEATED MEASURES ANOVA (Tekrarlı Ölçümler ANOVA). Personel Collection of S. Şen, Harran Üniversitesi- Osmanbey Kampüsü. Cited from https://sedatsen.files.wordpress.com/2016/11/6-sunum.pdf on July 24^{th}, 2018.

[54] Xu, J. and Li,H. (2007): AdaRank: A boosting algorithm for information retrieval. Proceedings of the Special Interest Group on Information Retrieval, 391-398.

[55] Taylor, M., Guiver, J., Robertson, S. and Minka, T. (2008): SoftRank: Optimizing non-smooth rank metrics. Proceedings of the Web Search and Data Mining, 77-86.

[56] Weber, G. W., Batmaz, I., Koksal, G., Taylan, P., and Yerlikaya-Ozkurt, F. (2012): CMARS— a new contribution to nonparametric regression with multivariate adaptive regression splines supported by continuous optimization. Inverse Problems in Science and Engineering, 20(3), 371-400.

[57] Xia, F., Liu, T. Y., Wang, J., Zhang, W., and Li, H. (2008): Listwise approach to learning to rank: theory and algorithm. In ICML 2008: Proceedings of the 25th international conference on Machine learning, New York, NY, USA, pp.1192-1199, ACM.

[58] Yeh, J.Y., Lin, J.Y., Ke, H.R and Yang, W.P. (2007): Learning to Rank for Information Retrieval Using Genetic Programming. Retrieved from https://people.cs.nctu.edu.tw/~jenyuan/publist/2007YLKY.LR4IR.Slides.pdf on August 7^{th}, 2018.

[59] Yue, Y., Finley, T., Radlinski, F. and Joachims, T.: A support vector method for optimizing average precision, Proceedings of the 30th annual international ACM SIGIR conference, pp.271-278, 2007.

[60] Zhai, C. (2008): Statistical Language Models for Information Retrieval, MORGAN & CLAYPOOL.

[61] Zweig, M.H., and Campbell, G. (1993): Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine. Clinical Chemistry 39:561-577. Retrieved from https://www.ncbi.nlm.nih.gov/pubmed/8472349 on August 26^{th}, 2018.

APPENDIX A

PERFORMANCE MEASURE TABLES

The performance measure tables of Microsoft Bing Data (sample data and original data) are given in App. A.1 and the performance measure table of LETOR 4.0 is given in App. A.2, respectively:

A.1 Microsoft Bing Data

TableA.1: 5-Fold performance measures of four models (sample data)

Folds	Performance Measures	Training				Testing				Stability			
		ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
1	precision	0.520	0.709	0.732	0.608	0.477	0.565	0.571	0.435	0.651	0.797	0.781	0.715
	recall	0.986	0.822	0.859	0.875	0.969	0.679	0.679	0.369	0.887	0.826	0.790	0.422
	f-measure	0.681	0.762	0.791	0.717	0.639	0.617	0.621	0.399	0.808	0.810	0.785	0.557
	g-measure	0.716	0.764	0.793	0.729	0.680	0.620	0.623	0.401	0.857	0.812	0.785	0.549
	Accuracy	0.524	0.735	0.766	0.484	0.476	0.596	0.602	0.521	0.622	0.811	0.787	0.929
	Specificity	0.034	0.643	0.666	0.998	0.023	0.520	0.532	1.000	0.035	0.809	0.798	0.998
	AUC	0.608	0.741	0.774	0.758	0.539	0.602	0.607	0.740	0.696	0.812	0.784	0.976
	Fall-out	0.966	0.357	0.334	0.002	1.000	0.480	0.468	0.115	0.966	0.744	0.713	0.018
	Youden's J	0.020	0.465	0.525	0.873	-0.008	0.199	0.211	0.369	-2.371	0.429	0.402	0.423
2	precision	0.534	0.678	0.642	1.000	0.517	0.582	0.577	1.000	0.805	0.858	0.899	1.000
	recall	1.000	0.910	0.873	0.002	1.000	0.818	0.794	0.006	0.873	0.898	0.909	0.312
	f-measure	0.696	0.777	0.740	0.004	0.681	0.680	0.668	0.012	0.921	0.875	0.903	0.313

Table A.1 continued from previous page

Folds	Performance Measures	Training				Testing				Stability			
		ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
	g-measure	0.731	0.785	0.749	0.043	0.719	0.690	0.677	0.077	0.960	0.878	0.904	0.559
	Accuracy	0.534	0.721	0.672	0.468	0.518	0.603	0.594	0.488	0.771	0.836	0.884	0.959
	Specificity	0.002	0.505	0.442	1.000	0.006	0.375	0.381	1.000	0.014	0.742	0.862	1.000
	AUC	0.767	0.754	0.698	0.734	0.758	0.620	0.606	0.743	0.921	0.822	0.868	0.988
	Fall-out	0.998	0.495	0.558	0.000	1.000	0.625	0.619	0.000	0.998	0.792	0.901	0.000
	Youden’s J	0.002	0.415	0.316	0.002	0.006	0.193	0.175	0.006	0.335	0.464	0.555	0.312
3	precision	0.504	0.638	0.656	1.000	0.540	0.606	0.698	0.001	0.823	0.949	0.940	0.001
	recall	0.990	0.844	0.744	0.002	1.000	0.828	0.615	0.001	0.744	0.982	0.826	0.500
	f-measure	0.668	0.727	0.697	0.004	0.701	0.700	0.654	0.001	0.994	0.963	0.938	0.251
	g-measure	0.706	0.734	0.699	0.045	0.735	0.709	0.655	0.001	0.951	0.965	0.938	0.022
	Accuracy	0.509	0.683	0.677	0.502	0.549	0.624	0.608	0.467	0.810	0.913	0.898	0.931
	Specificity	0.028	0.523	0.611	1.000	0.040	0.393	0.507	0.993	0.065	0.752	0.830	0.993
	AUC	0.620	0.705	0.681	0.750	0.770	0.638	0.607	0.766	0.884	0.905	0.891	0.979
	Fall-out	0.972	0.477	0.389	0.000	0.960	0.607	0.493	0.007	0.988	0.786	0.789	0.000
	Youden’s J	0.018	0.367	0.355	0.002	0.040	0.222	0.121	-0.006	0.449	0.604	0.342	-2.833

Table A.1 continued from previous page

Folds	Performance Measures	Training				Testing				Stability			
		ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
4	precision	0.547	0.634	0.665	1.000	0.554	0.565	0.583	1.000	0.833	0.892	0.877	1.000
	recall	0.998	0.953	0.883	0.002	1.000	0.844	0.722	0.006	0.883	0.886	0.818	0.324
	f-measure	0.707	0.761	0.759	0.004	0.713	0.677	0.645	0.011	0.940	0.890	0.851	0.326
	g-measure	0.739	0.777	0.766	0.042	0.744	0.691	0.649	0.075	0.971	0.889	0.847	0.569
	Accuracy	0.547	0.672	0.692	0.453	0.554	0.554	0.560	0.449	0.800	0.824	0.809	0.991
	Specificity	0.000	0.332	0.461	1.000	0.000	0.193	0.359	1.000	0.000	0.582	0.778	1.000
	AUC	0.726	0.744	0.715	0.726	0.724	0.533	0.546	0.724	0.988	0.716	0.764	0.997
	Fall-out	1.000	0.668	0.539	0.000	1.000	0.807	0.641	0.000	1.000	0.828	0.841	0.000
	Youden’s J	-0.002	0.285	0.344	0.002	0.000	0.038	0.081	0.006	0.000	0.132	0.235	0.324
5	precision	0.533	0.604	0.610	0.125	0.571	0.616	0.624	0.313	0.937	0.981	0.977	0.399
	recall	0.996	0.965	0.886	0.556	0.995	0.950	0.828	0.673	0.891	0.984	0.934	0.827
	f-measure	0.695	0.743	0.722	0.204	0.726	0.748	0.712	0.427	0.996	0.994	0.985	0.478
	g-measure	0.729	0.764	0.735	0.264	0.754	0.765	0.719	0.459	0.975	0.998	0.978	0.574
	Accuracy	0.534	0.645	0.637	0.466	0.569	0.633	0.616	0.425	0.894	0.982	0.967	0.911
	Specificity	0.006	0.280	0.886	0.998	0.000	0.210	0.331	0.993	0.000	0.747	0.374	0.995

Table A.1 continued from previous page

Folds	Performance Measures	Training				Testing				Stability			
		ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
	AUC	0.567	0.739	0.670	0.767	0.714	0.686	0.607	0.787	0.938	0.928	0.906	0.975
	Fall-out	0.994	0.720	0.648	0.002	1.000	0.791	0.669	0.007	0.994	0.910	0.968	0.314
	Youden’s J	0.003	0.245	0.773	0.554	-0.005	0.159	0.159	0.666	-1.928	0.649	0.206	0.832

TableA.2: 5-Fold performance measures of four models (original data)

Folds	Performance Measures	Training				Testing				Stability			
		ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
1	precision	0.541	0.590	0.607	0.571	0.633	0.816	0.834	0.813	0.854	0.723	0.728	0.703
	recall	0.937	0.790	0.827	0.549	0.255	0.102	0.243	0.164	0.272	0.129	0.294	0.299
	f-measure	0.686	0.676	0.700	0.560	0.363	0.181	0.376	0.273	0.530	0.268	0.538	0.487
	g-measure	0.712	0.683	0.708	0.560	0.402	0.288	0.450	0.365	0.564	0.422	0.635	0.652
	Accuracy	0.551	0.634	0.630	0.549	0.525	0.524	0.551	0.549	0.953	0.826	0.875	0.999
	Specificity	0.128	0.546	0.842	0.885	0.097	0.975	0.844	0.885	0.760	0.559	0.998	1.000
	AUC	0.596	0.237	0.647	0.549	0.657	0.660	0.647	0.665	0.907	0.359	1.000	0.826
	Fall-out	0.665	0.455	0.158	0.115	0.662	0.448	0.200	0.115	0.996	0.985	0.787	1.000
	Youden’s J	0.065	0.336	0.669	0.434	-0.648	0.077	0.087	0.049	-9.917	0.230	0.130	0.113
2	precision	0.570	0.719	0.743	0.304	0.745	0.580	0.667	0.507	0.765	0.807	0.897	0.599
	recall	0.834	0.492	0.416	0.296	0.201	0.828	0.243	0.246	0.241	0.595	0.583	0.832
	f-measure	0.677	0.584	0.533	0.300	0.317	0.682	0.356	0.332	0.468	0.857	0.667	0.905

Table A.2 continued from previous page

Folds	Performance Measures	Training				Testing				Stability			
		ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
	g-measure	0.689	0.595	0.556	0.300	0.387	0.693	0.402	0.353	0.561	0.858	0.724	0.849
	Accuracy	0.586	0.597	0.621	0.402	0.523	0.597	0.555	0.481	0.892	0.999	0.893	0.836
	Specificity	0.316	0.346	0.844	0.740	0.323	0.344	0.847	0.738	0.979	0.996	0.996	0.998
	AUC	0.603	0.756	0.657	0.531	0.643	0.244	0.654	0.510	0.938	0.323	0.995	0.960
	Fall-out	0.684	0.654	0.156	0.260	0.677	0.656	0.153	0.262	0.990	0.998	0.979	0.994
	Youden's J	0.150	-0.162	0.260	0.036	-0.476	0.172	0.090	-0.016	-3.170	-1.059	0.346	-2.288
3	precision	0.667	0.514	0.602	0.511	0.677	0.776	0.687	0.568	0.986	0.663	0.877	0.901
	recall	0.642	0.618	0.826	0.064	0.654	0.080	0.839	0.003	0.983	0.129	0.984	0.049
	f-measure	0.655	0.561	0.696	0.113	0.665	0.144	0.755	0.006	0.984	0.257	0.922	0.055
	g-measure	0.655	0.564	0.705	0.181	0.665	0.248	0.759	0.042	0.984	0.441	0.929	0.233
	Accuracy	0.650	0.617	0.628	0.485	0.652	0.617	0.628	0.473	0.996	0.999	1.000	0.976
	Specificity	0.657	0.432	0.416	0.861	0.590	0.419	0.840	0.581	0.898	0.968	0.495	0.675
	AUC	0.650	0.222	0.646	0.503	0.652	0.742	0.647	0.520	0.997	0.300	0.998	0.967
	Fall-out	0.400	0.568	0.156	0.196	0.410	0.581	0.160	0.432	0.977	0.976	0.978	0.452
	Youden's J	0.299	0.051	0.242	-0.075	0.244	-0.502	0.680	-0.416	0.814	-9.938	0.356	0.181

Table A.2 continued from previous page

Folds	Performance Measures	Training				Testing				Stability			
		ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
4	precision	0.688	0.395	0.602	0.545	0.703	0.650	0.692	0.553	0.978	0.607	0.869	0.986
	recall	0.614	0.785	0.832	0.306	0.884	0.335	0.826	0.312	0.694	0.427	0.993	0.979
	f-measure	0.649	0.525	0.698	0.392	0.783	0.442	0.753	0.399	0.828	0.841	0.927	0.982
	g-measure	0.650	0.557	0.707	0.408	0.788	0.467	0.756	0.416	0.824	0.838	0.936	0.982
	Accuracy	0.655	0.574	0.627	0.508	0.599	0.575	0.629	0.514	0.915	0.999	0.997	0.989
	Specificity	0.665	0.222	0.850	0.726	0.688	0.227	0.859	0.729	0.967	0.978	0.990	0.995
	AUC	0.657	0.347	0.647	0.519	0.638	0.278	0.648	0.525	0.971	0.802	0.998	0.989
	Fall-out	0.319	0.778	0.150	0.275	0.312	0.773	0.141	0.271	0.978	0.994	0.942	0.986
	Youden's J	0.279	0.007	0.682	0.031	0.572	-0.438	0.685	0.042	0.488	-65.433	0.996	0.750
5	precision	0.522	0.689	0.748	0.502	0.654	0.703	0.600	0.397	0.799	0.980	0.801	0.790
	recall	1.000	0.862	0.414	0.615	1.000	0.884	0.829	0.003	1.000	0.975	0.500	0.005
	f-measure	0.686	0.766	0.533	0.553	0.650	0.783	0.696	0.006	0.947	0.978	0.766	0.011
	g-measure	0.723	0.771	0.556	0.556	0.808	0.788	0.705	0.035	0.894	0.978	0.790	0.063
	Accuracy	0.522	0.569	0.621	0.539	0.523	0.563	0.626	0.483	0.999	0.990	0.993	0.896
	Specificity	0.001	0.186	0.848	0.994	0.001	0.176	0.836	0.995	0.833	0.945	0.986	0.999

Table A.2 continued from previous page

Folds	Performance Measures	Training				Testing				Stability			
		ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
	AUC	0.654	0.863	0.659	0.484	0.599	0.834	0.645	0.560	0.916	0.966	0.979	0.864
	Fall-out	0.999	0.814	0.152	0.006	0.999	0.824	0.164	0.005	1.000	0.988	0.926	0.888
	Youden’s J	0.001	0.049	0.262	0.609	0.000	0.060	0.664	0.698	0.855	0.806	0.394	0.873

A.2 LETOR 4.0

TableA.3: 5-Fold performance measures of four models (LETOR 4.0)

Folds	Performance Measures	Training				Testing				Stability			
		ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
1	precision	0.818	0.812	0.885	0.582	0.812	0.807	0.867	0.816	0.993	0.994	0.980	0.713
	recall	0.996	1.000	0.921	0.601	0.992	0.460	0.872	0.918	0.997	0.460	0.947	0.654
	f-measure	0.898	0.896	0.903	0.591	0.893	0.586	0.869	0.864	0.995	0.654	0.963	0.684
	g-measure	0.902	0.901	0.903	0.591	0.898	0.609	0.869	0.866	0.995	0.676	0.963	0.683
	Accuracy	0.816	0.812	0.839	0.464	0.809	0.807	0.789	0.767	0.991	0.993	0.941	0.605
	Specificity	0.040	0.696	0.483	0.885	0.041	0.736	0.441	0.137	0.973	0.000	0.913	0.155
	AUC	0.750	0.594	0.735	0.438	0.687	0.597	0.660	0.551	0.916	0.995	0.898	0.795
	Fall-out	0.960	1.000	0.517	0.864	0.959	1.000	0.559	0.908	0.999	1.000	0.925	0.951
	Youden’s J	0.036	0.696	0.404	0.486	0.034	0.196	0.313	0.055	0.936	0.282	0.775	0.113
	precision	0.821	0.813	0.879	0.535	0.793	0.790	0.827	0.738	0.966	0.971	0.941	0.725
	recall	0.994	1.000	0.921	0.362	0.990	1.000	0.898	0.313	0.996	1.000	0.975	0.863
	f-measure	0.899	0.897	0.899	0.432	0.881	0.882	0.861	0.439	0.979	0.984	0.957	0.984

2

Table A.3 continued from previous page

Folds	Performance Measures	Training				Testing				Stability			
		ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
	g-measure	0.903	0.902	0.900	0.440	0.886	0.889	0.862	0.480	0.981	0.986	0.958	0.916
	Accuracy	0.819	0.813	0.833	0.350	0.788	0.789	0.771	0.370	0.962	0.971	0.926	0.947
	Specificity	0.058	0.000	0.449	0.633	0.029	0.000	0.295	0.584	0.503	0.000	0.656	0.922
	AUC	0.750	0.594	0.723	0.560	0.616	0.605	0.631	0.539	0.821	0.982	0.873	0.963
	Fall-out	0.942	1.000	0.551	0.356	0.971	1.000	0.705	0.277	0.970	1.000	0.781	0.778
	Youden’s J	0.052	0.000	0.370	-0.005	0.019	0.000	0.193	-0.104	0.372	0.000	0.520	0.043
3	precision	0.824	0.796	0.884	0.303	0.861	0.847	0.889	0.261	0.958	0.940	0.995	0.862
	recall	0.984	1.000	0.912	0.787	0.979	1.000	0.892	0.245	0.995	1.000	0.979	0.311
	f-measure	0.897	0.887	0.898	0.438	0.916	0.917	0.890	0.253	0.979	0.966	0.992	0.577
	g-measure	0.901	0.892	0.898	0.489	0.918	0.920	0.890	0.253	0.981	0.969	0.992	0.517
	Accuracy	0.820	0.796	0.834	0.709	0.848	0.847	0.814	0.524	0.967	0.940	0.976	0.739
	Specificity	0.179	0.000	0.531	0.483	0.119	0.000	0.378	0.516	0.664	0.000	0.712	0.937
	AUC	0.784	0.602	0.745	0.579	0.684	0.576	0.638	0.585	0.872	0.957	0.856	0.990
	Fall-out	0.821	1.000	0.469	0.053	0.881	1.000	0.622	0.223	0.932	1.000	0.754	0.239
	Youden’s J	0.163	0.000	0.443	0.271	0.098	0.000	0.271	-0.239	0.601	0.000	0.611	-1.131

Table A.3 continued from previous page

Folds	Performance Measures	Training				Testing				Stability			
		ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
4	precision	0.812	0.796	0.889	0.368	0.804	0.792	0.846	0.376	0.990	0.995	0.952	0.979
	recall	0.989	1.000	0.905	0.856	0.986	1.000	0.892	0.863	0.997	1.000	0.985	0.992
	f-measure	0.892	0.886	0.897	0.515	0.886	0.884	0.868	0.524	0.993	0.997	0.968	0.983
	g-measure	0.896	0.892	0.897	0.561	0.890	0.890	0.868	0.569	0.993	0.997	0.968	0.986
	Accuracy	0.809	0.796	0.834	0.750	0.798	0.791	0.785	0.459	0.987	0.995	0.942	0.612
	Specificity	0.105	0.000	0.557	0.417	0.085	0.000	0.381	0.773	0.805	0.000	0.684	0.539
	AUC	0.765	0.602	0.745	0.548	0.709	0.604	0.663	0.554	0.927	0.997	0.890	0.989
	Fall-out	0.895	1.000	0.443	0.203	0.915	1.000	0.619	0.310	0.978	1.000	0.715	0.654
	Youden’s J	0.095	0.000	0.462	0.273	0.071	0.000	0.273	0.635	0.746	0.000	0.590	0.429
5	precision	0.823	0.817	0.888	0.415	0.798	0.791	0.833	0.309	0.969	0.967	0.938	0.746
	recall	0.993	1.000	0.925	0.850	0.994	1.000	0.892	0.239	0.999	1.000	0.964	0.281
	f-measure	0.900	0.899	0.906	0.558	0.885	0.883	0.861	0.269	0.983	0.982	0.951	0.483
	g-measure	0.904	0.904	0.906	0.594	0.890	0.889	0.862	0.272	0.985	0.984	0.951	0.458
	Accuracy	0.820	0.817	0.843	0.731	0.796	0.790	0.773	0.418	0.971	0.967	0.917	0.571
	Specificity	0.048	0.000	0.476	0.365	0.049	0.000	0.323	0.510	0.973	0.000	0.678	0.715

Table A.3 continued from previous page

Folds	Performance Measures	Training				Testing				Stability			
		ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS	ANN	SVM	MARS	CMARS
	AUC	0.715	0.591	0.737	0.467	0.740	0.605	0.637	0.543	0.966	0.977	0.864	0.860
	Fall-out	0.952	1.000	0.524	0.032	0.951	1.000	0.677	0.034	0.999	1.000	0.774	0.950
	Youden's J	0.041	0.000	0.401	0.215	0.043	0.000	0.215	-0.251	0.948	0.000	0.536	-1.170

APPENDIX B

FEATURE TABLES FOR DATASETS

B.1 LETOR 4.0

TableB.1: Feature List of LETOR 4.0 Learning to Rank Datasets

feature id	feature description
1	TF of body
2	TF of anchor
3	TF of title
4	TF of URL
5	TF of whole document
6	IDF of body
7	IDF of anchor
8	IDF of title
9	IDF of URL
10	IDF of whole document
11	TF*IDF of body
12	TF*IDF of anchor
13	TF*IDF of title
14	TF*IDF of URL
15	TF*IDF of whole document
16	DL of body
17	DL of anchor

Table B.1 continued from previous page

feature id	feature description
18	DL of title
19	DL of URL
20	DL of whole document
21	BM25 of body
22	BM25 of anchor
23	BM25 of title
24	BM25 of URL
25	BM25 of whole document
26	LMIR.ABS of body
27	LMIR.ABS of anchor
28	LMIR.ABS of title
29	LMIR.ABS of URL
30	LMIR.ABS of whole document
31	LMIR.DIR of body
32	LMIR.DIR of anchor
33	LMIR.DIR of title
34	LMIR.DIR of URL
35	LMIR.DIR of whole document
36	LMIR.JM of body
37	LMIR.JM of anchor
38	LMIR.JM of title
39	LMIR.JM of URL
40	LMIR.JM of whole document
41	PageRank
42	inlink number
43	outlink number
44	number of slash in URL
45	length of URL
46	number of child page

B.2 Microsoft Bing

TableB.2: Feature List of Microsoft Bing Learning to Rank Datasets

feature id	feature description	stream	comments
1	covered query term number	body	
2		anchor	
3		title	
4		url	
5		whole document	
6	covered query term ratio	body	
7		anchor	
8		title	
9		url	
10		whole document	
11	stream length	body	
12		anchor	
13		title	

Table B.2 continued from previous page

feature id	feature description	stream	comments
14		url	
15		whole document	
16	IDF	body	
17		anchor	
18		title	
19		url	
20		whole document	
21	sum of term frequency	body	
22		anchor	
23		title	
24		url	
25		whole document	
26	min of term frequency	body	
27		anchor	
28		title	

Table B.2 continued from previous page

feature id	feature description	stream	comments
29		url	
30		whole document	
31	max of term frequency	body	
32		anchor	
33		title	
34		url	
35		whole document	
36	mean of term frequency	body	
37		anchor	
38		title	
39		url	
40		whole document	
41	variance of term frequency	body	
42		anchor	

Table B.2 continued from previous page

feature id	feature description	stream	comments
43		title	
44		url	
45		whole document	
46	sum of stream length normalized term frequency	body	
47		anchor	
48		title	
49		url	
50		whole document	
51	min of stream length normalized term frequency	body	
52		anchor	
53		title	
54		url	
55		whole document	

Table B.2 continued from previous page

feature id	feature description	stream	comments
56	max of stream length normalized term frequency	body	
57		anchor	
58		title	
59		url	
60		whole document	
61	mean of stream length normalized term frequency	body	
62		anchor	
63		title	
64		url	
65		whole document	
66	variance of stream length normalized term frequency	body	

Table B.2 continued from previous page

feature id	feature description	stream	comments
67		anchor	
68		title	
69		url	
70		whole document	
71	sum of tf*idf	body	
72		anchor	
73		title	
74		url	
75		whole document	
76	min of tf*idf	body	
77		anchor	
78		title	
79		url	
80		whole document	
81	max of tf*idf	body	
82		anchor	
83		title	

Table B.2 continued from previous page

feature id	feature description	stream	comments
84		url	
85		whole document	
86	mean of tf*idf	body	
87		anchor	
88		title	
89		url	
90		whole document	
91	variance of tf*idf	body	
92		anchor	
93		title	
94		url	
95		whole document	
96	boolean model	body	
97		anchor	
98		title	
99		url	
100		whole document	

Table B.2 continued from previous page

feature id	feature description	stream	comments
101	VSM	body	
102		anchor	
103		title	
104		url	
105		whole document	
106	BM25	body	
107		anchor	
108		title	
109		url	
110		whole document	
111	LMIR.ABS	body	Language model approach for IR with absolute discounting smoothing
112		anchor	
113		title	
114		url	
115		whole document	

Table B.2 continued from previous page

feature id	feature description	stream	comments
116	LMIR.DIR	body	Language model approach for IR with Bayesian smoothing using Dirichlet priors
117		anchor	
118		title	
119		url	
120		whole document	
121	LMIR.JM	body	Language model approach for IR with Jelinek-Mercer smoothing
122		anchor	
123		title	
124		url	
125		whole document	
126	Number of slash in URL		
127	Length of URL		
128	Inlink number		
129	Outlink number		

Table B.2 continued from previous page

feature id	feature description	stream	comments
130	PageRank		
131	SiteRank		Site level PageRank
132	QualityScore		The quality score of a web page. The score is outputted by a web page quality classifier.
133	QualityScore2		The quality score of a web page. The score is outputted by a web page quality classifier, which measures the badness of a web page.
134	Query-url click count		The click count of a query-url pair at a search engine in a period
135	url click count		The click count of a url aggregated from user browsing data in a period
136	url dwell time		The average dwell time of a url aggregated from user browsing data in a period

APPENDIX C

[R] PACKAGES

C.1 R Libraries

The packages that are used during analysis are given as follows:

- `BalancedSampling`: To get representative sample from datasets. Retrieved from https://cran.r-project.org/web/packages/BalancedSampling/index.html
- `nnet`: To get feedforward neural network (NN) model. Retrieved from https://cran.r-project.org/web/packages/nnet/index.html
- `ROSE`: To calculate performance measures for comparison. Retrieved from https://cran.r-project.org/web/packages/ROSE/index.html
- `earth`: To get multivariate adaptive regression splines (MARS) model. Retrieved from https://cran.r-project.org/web/packages/earth/index.html
- `RSofia`: To get support vector machine (SVM) model. Retrieved from https://cran.r-project.org/web/packages/RSofia/index.html
- `data.table`: To have common data frame for sampling. Retrieved from https://cran.r-project.org/web/packages/data.table/index.html
- `xlsx`: To write data in xlsx format.Retrieved from https://cran.r-project.org/web/packages/xlsx/index.html Note that Java must be the updated on your PC.