SVM untuk Ranking

Dr. rer. nat. Hendri Murfi

Intelligent Data Analysis (IDA) Group

Departemen Matematika, Universitas Indonesia – Depok 16424

Telp. +62-21-7862719/7863439, Fax. +62-21-7863439, Email. hendri@ui.ac.id

Model Linear

- Model dasar yang akan digunakan adalah model linear, yaitu model yang merupakan kombinasi linear dari fungsi basis, yaitu:

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x})$$

  Dimana $\mathbf{x} = (x_1, x_2, ..., x_D)^T$ adalah variabel input, dan $\mathbf{w} = (w_0, w_1, ..., w_D)^T$ adalah parameter, $\phi(\boldsymbol{x})$ adalah *fungsi basis*, M adalah jumlah fungsi basis

- Pada banyak metode $\phi_0(\boldsymbol{x}) = 1$, sehingga w_0 berfungsi sebagai bias

- Ada banyak pilihan yang mungkin untuk fungsi basis $\phi(\boldsymbol{x})$, misal fungsi polinomial, fungsi gaussian, fungsi sigmoidal, fungsi pemetaan, dll

Model Linear

Kutukan Dimensi

- Model linear memiliki sifat-sifat yang penting baik dari aspek komputasi maupun analitik. Penggunaan model linear dengan pendekatan parametrik pada metode klasik memiliki keterbatasan pada aplikasi praktis disebabkan oleh kutukan dimensi *(curse of dimensionality)*

$$y(x,w)=w_0+w_1x+w_2x^2+w_3x^3$$

$$y(\mathbf{x},\mathbf{w}) = w_0 + \sum_{i=1}^{D} w_i x_i + \sum_{i=1}^{D}\sum_{j=1}^{D} w_{ij} x_i x_j + \sum_{i=1}^{D}\sum_{j=1}^{D}\sum_{k=1}^{D} w_{ijk} x_i x_j x_k$$

1

D^3

untuk model beorde M, maka pertumbuhan jumlah parameter w proposional dengan D^M

Model Linear

Solusi Praktis

- Untuk menerapkan metode ini pada masalah skala besar, pendekatan umum yang dilakukan adalah membuat fungsi basis dapat beradaptasi dengan data pembelajaran, dan mengeset data pembelajaran tersebut sebagai pusat fungsi basis. Selanjutnya, memilih sebagian dari data pembelajaran tersebut selama proses training (nonparametrik).

- Dasarnya adalah bahwa data real biasanya memiliki sifat mulus, artinya perubahan sedikit pada data input hanya akan memberikan sedikit perubahan pada output

- Penggunaan fungsi kernel sebagai fungsi basis adalah salah satu contoh pendekatan seperti ini yang banyak digunakan saat ini.

Metode Kernel

Fungsi Kernel: Definisi

- Fungsi kernel adalah suatu fungsi *k* yang mana untuk semua vektor input ***x,z*** akan memenuhi kondisi

$$k(\mathbf{x},\mathbf{z}) = \phi(\mathbf{x})^T \phi(\mathbf{z})$$

dimana $\phi(.)$ adalah fungsi pemetaan dari ruang input ke ruang fitur

- Dengan kata lain, fungsi kernel adalah fungsi perkalian dalam (*inner product*) pada ruang fitur.

Metode Kernel

Fungsi Kernel: Contoh

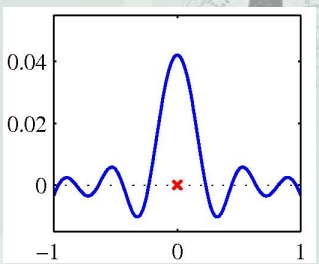
- Salah satu contoh fungsi kernel yang banyak digunakan adalah fungsi Gaussian, yaitu:

$$k(\mathbf{x},\mathbf{x}') = \exp(-||\mathbf{x}-\mathbf{x}'||^2 / 2\sigma^2)$$

dimana ***x'*** adalah „inti“ yang biasanya dipilih dari data pembelajaran.

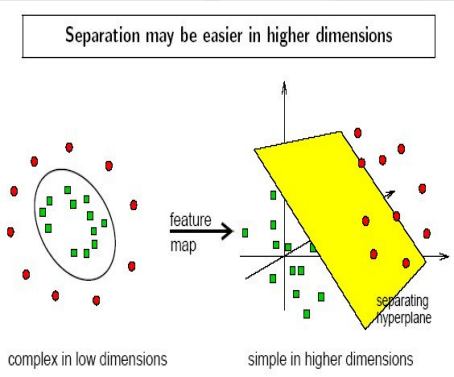
- Misal untuk data x=5 dengan target t=0.04, maka data tsb dapat digambarkan dengan fungsi basis sbb:

$$y(x) = 0.04 * \exp(-||x-5||^2 / 2\sigma^2)$$



Metode Kernel

Fungsi Kernel: Keuntungan



- Fungsi kernel memungkinkan kita untuk mengimplementasikan suatu model pada ruang dimensi lebih tinggi (ruang fitur) tanpa harus mendefinisikan fungsi pemetaan dari ruang input ke ruang fitur
- Sehingga, untuk kasus yang *non-linearly separable* pada ruang input, diharapkan akan menjadi *linearly separable* pada ruang fitur
- Selanjutnya, kita dapat menggunakan *hyperplane* sebagai *decision boundary* secara efisien

Metode Kernel

Fungsi Kernel: Penggunaan

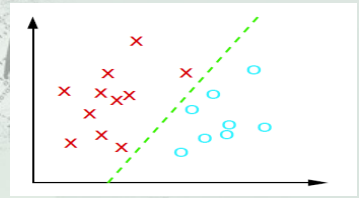
- Secara umum, ada dua cara penggunaan metode kernel pada *machine learning*, yaitu:
  - Penggunaan langsung, yaitu fungsi kernel digunakan sebagai fungsi basis dari model machine learning tersebut, contoh: *radial basis function*
  - Penggunaan tidak langsung melalui *kernel trick*, yaitu merepresentasikan suatu model ke dalam representasi dual yang mengandung *inner product* dari fungsi pemetaan, contoh: ***support vector machine***, *kernel linear regression*, *kernel Perceptron*, *kernel PCA*, dll

Support Vector Machine

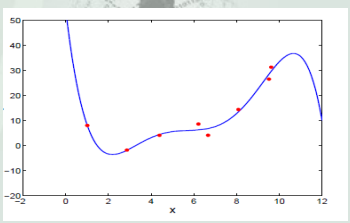
Klasifikasi dan Regresi

Diberikan data pembelajaran $\{\boldsymbol{x}_n, t_n\}$, $n = 1$ sd N

- Klasifikasi
  - t_n bernilai diskrit (kelas) berhingga tidak terurut (skala nominal)
  - Permasalahan: menentukan *decision boundary* yang dapat mengklasifikasi data dengan benar



- Regresi
  - t_i bernilai kontinu (bilangan real)
  - Permasalahan: menentukan fungsi regresi yang dapat memprediksi nilai data dengan benar

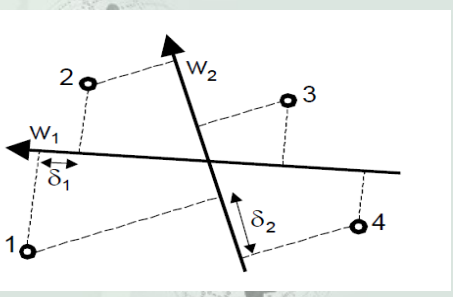


Support Vector Machine

Regresi Ordinal

Diberikan data pembelajaran $\{\boldsymbol{x}_n, t_n\}$, $n = 1$ sd N

- Regresi Ordinal
  - t_n bernilai diskrit (kelas) berhingga seperti klasifikasi
  - Terdapat urutan diantara elemen t_n (skala ordinal) seperti regresi
  - Permasalahan: menentukan fungsi regresi ordinal yang dapat mengklasifikasikan data dengan benar

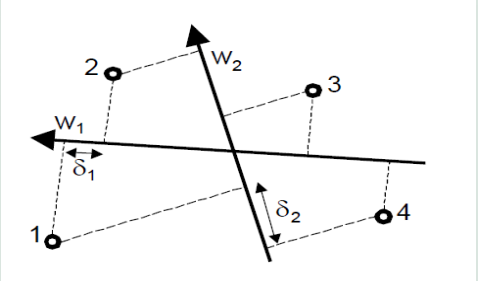


Support Vector Machine

Ranking

Diberikan data pembelajaran $\{\boldsymbol{x}_n, t_n\}$, $n = 1$ sd N

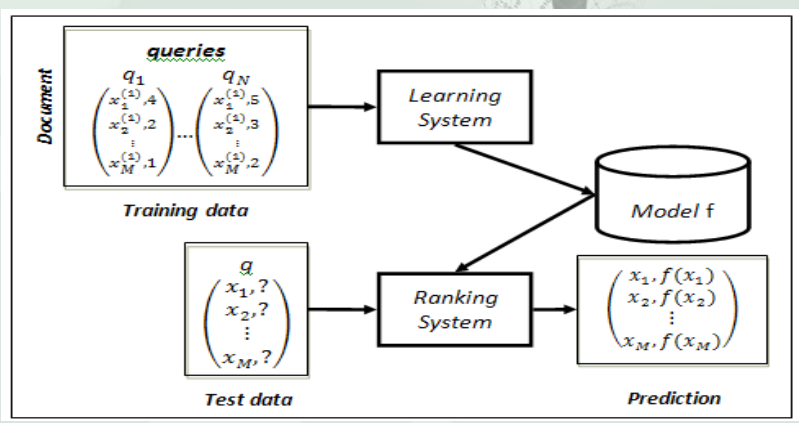
- Ranking
  - t_n bernilai real yang menyatakan urutan (ranking) dari data
  - Permasalahan: menentukan fungsi ranking yang dapat mengurutkan data dengan benar



SVM untuk Ranking

Learning to Rank

- *Learning to Rank* adalah suatu bidang penelitian yang bertujuan membangun fungsi ranking dengan mengunakan machine learning
- Misal $\boldsymbol{x}_i$ adalah suatu item/dokumen, $\boldsymbol{q}_j$ adalah suatu query, f adalah suatu fungsi ranking, maka *Learning to Rank* dapat diilustrasikan sbb:



SVM untuk Ranking

Bentuk Umum

- Model linear yang akan digunakan sebagai fungsi regresi ordinal memiliki bentuk umum sbb:

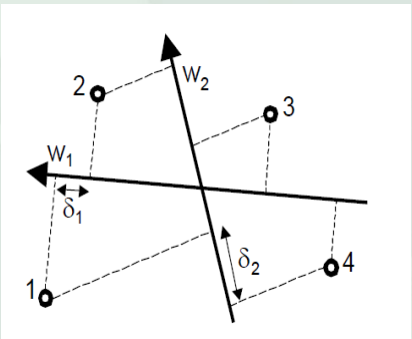
$$y(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x})$$

dimana $\mathbf{x}$ adalah vektor input, $\mathbf{w}$ adalah parameter bobot, dan $\phi(\mathbf{x})$ adalah fungsi pemetaan

SVM untuk Ranking

Formulasi Masalah

- Diberikan data pembelajaran $\{\mathbf{x}(\mathbf{q}_k, \mathbf{d}_n), t\}$, dimana $\mathbf{x}(\mathbf{q}_k, \mathbf{d}_n) \in R^D$ adalah suatu vektor dimana dimensi/fitur merupakan parameter-parameter kemiripan antara query $\mathbf{q}_k \in R^S$, $k=1..K$ dan item/dokumen $\mathbf{d}_n \in R^S$, $n=1..N$, dan $t \in \{r_1, r_2, ..., r_P\}$ adalah suatu skala ordinal.



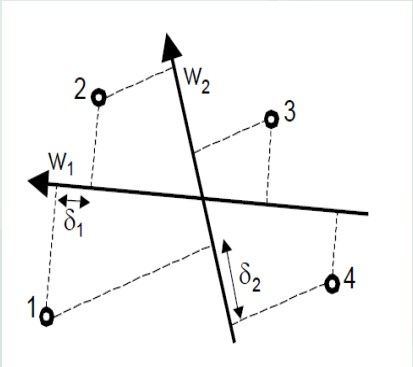
- Permasalahan adalah menentukan fungsi $y(\mathbf{x})$ sedemikian sehingga untuk sembarang pasangan data training $(\mathbf{x}_i, t_i)$ dan $(\mathbf{x}_j, t_j)$ maka

$$y(\mathbf{x}_i) > y(\mathbf{x}_j) <==> t_i > t_j$$

Contoh: Diberikan data training $(\mathbf{x}_1, 4)$, $(\mathbf{x}_2, 3)$, $(\mathbf{x}_3, 2)$, $(\mathbf{x}_4, 1)$, maka $\mathbf{w}_1$ adalah fungsi yang benar

SVM untuk Ranking

Formulasi Masalah



- Misal P_k adalah himpunan dari pasangan *(i,j)* untuk suatu query $\boldsymbol{q}_k$, dimana $\mathbf{x}_i$ memiliki ranking yang lebih tinggi dari $\mathbf{x}_j$:

$$P_k = \{(i,j) : t_i > t_j\}$$

Contoh: dari contoh sebelumnya, maka $P_1 = \{(1,2), (1,3), (1,4), (2,3), (2,4), (3,4)\}$

SVM untuk Ranking

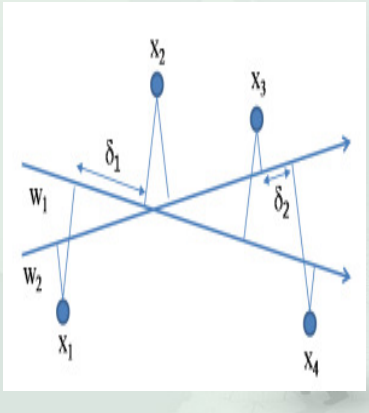
Formulasi Masalah

- Maka penentuan fungsi ***y(x)*** harus memenuhi kondisi berikut ini:

$$\forall (i,j) \in P_1 : w^T \phi(x(q_1, d_i)) \geq w^T \phi(x(q_1, d_j))$$
$$\forall (i,j) \in P_2 : w^T \phi(x(q_2, d_i)) \geq w^T \phi(x(q_2, d_j))$$
$$\vdots$$
$$\forall (i,j) \in P_k : w^T \phi(x(q_k, d_i)) \geq w^T \phi(x(q_k, d_j))$$

SVM untuk Ranking

Formulasi Masalah



- Asumsikan semua data dapat di ranking dengan benar (*linearly rankable*), maka jarak antara proyeksi dua data **x(*q*$_k$, *d*$_i$)** dan **x(*q*$_k$, *d*$_j$)** pada ***w*** adalah:

$$\frac{w^T(\phi(x(q_k,d_i))-\phi(x(q_k,d_j)))}{||w||}$$

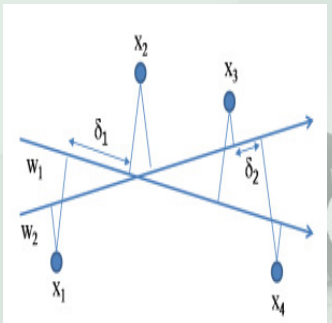
- Margin (δ_i) adalah jarak terdekat dari proyeksi dua data training pada vektor bobot **w.** Sehingga, memaksimumkan margin dapat dideskripsikan sbb:

$$\arg\max_w \{\frac{1}{||w||} \min_{i,j}[w^T(\phi(x(q_k,d_i))-\phi(x(q_k,d_j)))]\}$$

SVM untuk Ranking

Formulasi Masalah

- Solusi langsung dari masalah optimasi sebelumnya akan sangat kompleks, sehingga perlu dikonversi ke masalah yang ekivalen yang lebih mudah diselesaikan



- Salah satu metode adalah menggunakan bentuk kanonik, yaitu:

$$w^T(\phi(x(q_k,d_i))-\phi(x(q_k,d_j))) = 1$$

untuk data yang terdekat. Selanjutnya, semua data pembelajaran harus memenuhi kondisi berikut ini:

$$w^T(\phi(x(q_k,d_i))-\phi(x(q_k,d_j))) \geqslant 1$$

SVM untuk Ranking

Formulasi Masalah

- Sehingga masalah penentuan fungsi ranking *y(**x**)* dapat ditulis sbb:

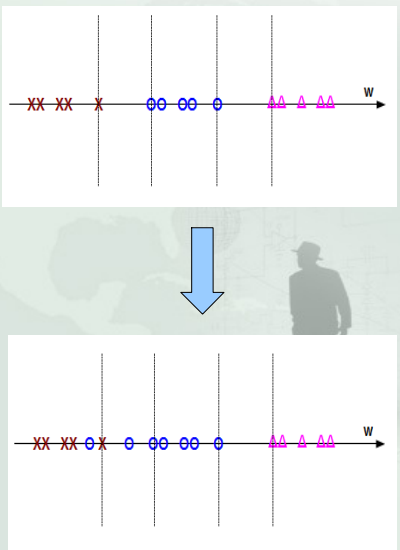
$$\arg\min_{w} \ \frac{1}{2}\|w\|^2$$

$$s.t \quad \forall (i,j) \in P_1 : w^T \phi(x(q_1, d_i)) \geqslant w^T \phi(x(q_1, d_j))$$
$$\forall (i,j) \in P_2 : w^T \phi(x(q_2, d_i)) \geqslant w^T \phi(x(q_2, d_j))$$
$$\vdots$$
$$\forall (i,j) \in P_k : w^T \phi(x(q_k, d_i)) \geqslant w^T \phi(x(q_k, d_j))$$

- Dengan kata lain, penentuan fungsi ranking *y(**x**)* menjadi masalah pemrograman kuadrat (*quadratic programming*), yaitu meminimumkan suatu fungsi kuadrat dengan kendala pertidaksamaan linear

SVM untuk Ranking

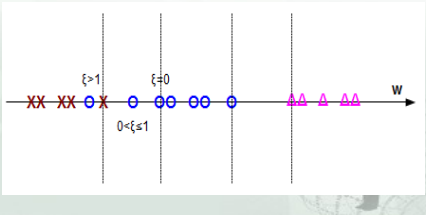
Soft Margin: Landasan



- Diberikan data pembelajaran *{**x**(**q**$_k$,**d**$_n$), t}*
- Pada formulasi sebelumnya, semua data diasumsikan *linearly rankable* pada ruang fitur *ϕ(**x**)*.
- Dalam aplikasi praktis, kondisi tersebut sering tidak dapat terpenuhi bahkan setelah data di transformasi ke ruang fitur *ϕ(**x**)*.
- Masalah ini diatasi dengan membuat margin lunak (*soft margin*) yang memungkinkan beberapa data pada „urutan yang salah“

SVM untuk Ranking

Soft Margin: Formulasi Masalah



- Untuk merealisasikan *soft margin* ini, diperkenalkan variabel *slack*, $\xi_{ijk} \geq 0$ untuk masing-masing pasangan data pembelajaran
- Variabel *slack* tersebut bernilai

$$\forall (i,j) \in P_k : w^T[\phi(x(q_k,d_i)) - w^T\phi(x(q_k,d_j))] = 1 - \xi_{ijk}$$

- Sehingga, $\xi_{ijk} = 0$ adalah pasangan data yang terletak pada *margin*, $0 < \xi_{ijk} \leq 1$ adalah pasangan data yang terletak didalam margin, dan $\xi_{ijk} > 1$ adalah pasangan data yang salah ranking

SVM untuk Ranking

Soft Margin: Formulasi Masalah

- Sehingga masalah penentuan fungsi ranking *y(**x**)* dapat ditulis sbb [1]:

$$\begin{aligned}
&\arg\min_{w} \ \frac{1}{2}\|w\|^2 + C\sum \xi_{ijk} \\
&s.t \quad \forall (i,j) \in P_1 : w^T[\phi(x(q_{1,}d_i)) - w^T\phi(x(q_{1,}d_j))] \geq 1 - \xi_{ijk} \\
&\quad \forall (i,j) \in P_2 : w^T[\phi(x(q_{2,}d_i)) - w^T\phi(x(q_{2,}d_j))] \geq 1 - \xi_{ijk} \\
&\quad \vdots \\
&\quad \forall (i,j) \in P_k : w^T[\phi(x(q_k,d_i)) - w^T\phi(x(q_k,d_j))] \geq 1 - \xi_{ijk} \\
&\quad \xi_{ijk} \geq 0
\end{aligned}$$

dimana parameter C > 0 akan mengkontrol *trade-off* antara lebar margin (kapabilitas generalisasi) dan jumlah pasangan data training yang tertukar urutannya (kesalahan urutan)

SVM untuk Ranking

Solusi Masalah: Lagrange Multipliers

- Masalah optimisasi diatas tampak ekivalen dengan masalah optimisasi dari SVM untuk klasifikasi, dimana setiap vektor diganti dengan pasangan vektor jarak $[\phi(\boldsymbol{x}(\boldsymbol{q}_{k'}\boldsymbol{d}_i)) - \phi(\boldsymbol{x}(\boldsymbol{q}_{k'}\boldsymbol{d}_k))]$. Sehingga penyelesaian masalah optimasi ini dapat mengadaptasi pendekatan yang digunakan oleh SVM untuk klasifikasi

SVM untuk Ranking

Prediksi Data Baru

- Misal diberikan himpunan data $\{\boldsymbol{x}(\boldsymbol{q}_{new,}\boldsymbol{d}_1), \boldsymbol{x}(\boldsymbol{q}_{new,}\boldsymbol{d}_2), \ldots, \boldsymbol{x}(\boldsymbol{q}_{new,}\boldsymbol{d}_N)\}$, maka ranking dari data tersebut ditentukan berdasarkan fungsi ranking $y(\boldsymbol{x})$, yaitu:

$$\boldsymbol{x}_i \propto \boldsymbol{x}_j \text{ jika } y(\boldsymbol{x}_i) > y(\boldsymbol{x}_j)$$

dimana $\propto$ didefiniskan „memiliki ranking yang lebih tinggi“

SVM untuk Ranking

Algoritma & Perangkat Lunak

- Ada beberapa algoritma dan perangkat lunak yang telah dikembangkan untuk memecahkan masalah optimasi pada SVM untuk Ranking, antara lain:
  - SVM^{light} [1][2] (http://svmlight.joachims.org)
  - SVM^{rank} [3] (http://svmlight.joachims.org)

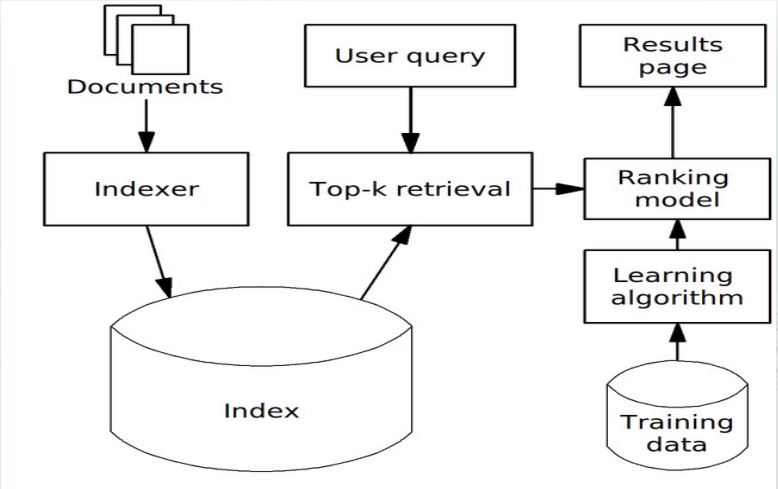
SVM untuk Ranking

Aplikasi

- ***Information Retrieval*** untuk meranking dokumen/web yang relevan dengan suatu query yang diberikan oleh pengguna
- ***Machine Translation*** untuk meranking sekumpulan hipotesis translasi
- ***Computational Biology*** untuk meranking kandidat struktur 3-D dari suatu protein
- ***Recommender System*** untuk meranking artikel berita terkait dengan berita yang sedang dibaca oleh pengguna

SVM untuk Ranking

Learning to Rank for Information Retrieval

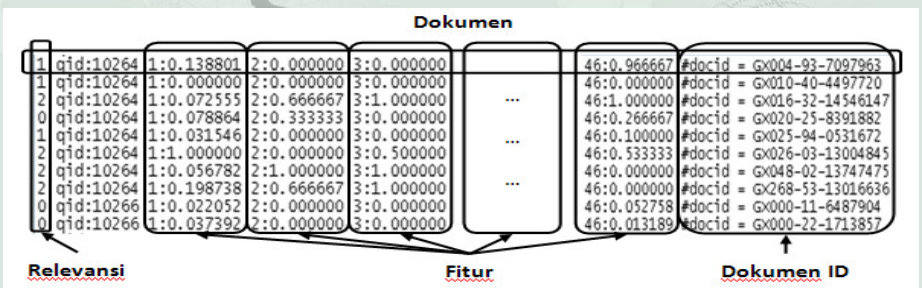


Contoh arsitektur dari suatu *machine-learned search engine* (sumber: www.wikipedia.org)

SVM untuk Ranking

Data Training

- Salah satu data training yang sering digunakan adalah LETOR, yaitu koleksi data yang diorganisasi oleh Microsoft yang menyimpan penentuan relevansi web terhadap query yang diberikan oleh pengguna pada mesin pencari Microsoft Bing (http://www.bing.com)
- Pada contoh kasus ini akan digunakan data LETOR 4.0 MQ2008 (http://research.microsoft.com/users/LETOR/)



SVM untuk Ranking

Data Training: Fitur

- Masing-masing data training terdiri dari 46 fitur yang merupakan tingkat kemiripan (similarity) antara query dan web

No	Deskripsi	No	Deskripsi	No	Deskripsi
1	TF of body	17	DL of anchor	33	BM25 of URL
2	TF of anchor	18	DL of title	34	LMIR.ABS of URL
3	TF of title	19	DL of the URL	35	LMIR.DIR of URL
4	TF of URL	20	DL of document	36	LMIR.IM of URL
5	TF of document	21	BM25 of body	37	BM25 of document
6	IDF of body	22	LMIR.ABS of body	38	LMIR.ABS of document
7	IDF of anchor	23	LMIR.DIR of body	39	LMIR.DIR of document
8	IDF of title	24	LMIR.IM of body	40	LMIR.IM of document
9	IDF of URL	25	BM25 of anchor	41	PageRank
10	IDF of document	26	LMIR.ABS of anchor	42	Inlink number
11	TF * IDF of body	27	LMIR.DIR of anchor	43	Outlink number
12	TF * IDF of anchor	28	LMIR.IM of anchor	44	Number of slash in URL
13	TF * IDF of title	29	BM25 of title	45	Length of URL
14	TF * IDF of URL	30	LMIR.ABS of title	46	Number of child page
15	TF * IDF of document	31	LMIR.DIR of title		
16	DL of body	32	LMIR.IM of title		

SVM untuk Ranking

Data Training: Fitur

Diberikan suatu himpunan dokumen $\{\boldsymbol{d}_1, \boldsymbol{d}_2, .., \boldsymbol{d}_N\}$ dan suatu query $\boldsymbol{q}$

- *Term Frequency* (TF)

  $TF(\boldsymbol{q}, \boldsymbol{d}_i)$ = jumlah kata-kata (*terms*) dari query $\boldsymbol{q}$ pada dokumen $\boldsymbol{d}_i$

- *Inverse Document Frequency* (IDF)

  $$IDF(\boldsymbol{q}, \boldsymbol{d}_i) = log\ (N/DF(\boldsymbol{q}, \boldsymbol{d}_i)$$

  dimana $DF(\boldsymbol{q}, \boldsymbol{d}_i)$ = jumlah dokumen yang mengandung kata-kata (*terms*) dari query $\boldsymbol{q}$

- *Term Frequency* * *Inverse Document Frequency* (TFIDF)

  $$TFIDF(\boldsymbol{q}, \boldsymbol{d}_i) = TF(\boldsymbol{q}, \boldsymbol{d}_i) * IDF(\boldsymbol{q}, \boldsymbol{d}_i)$$

SVM untuk Ranking

Satuan Ukuran Akurasi: P@n

- *Precision at Position n* (P@n):

$$P@n = \frac{jumlah\ web\ yang\ relevan\ pada\ posisi\ n}{n}$$

Contoh:

Misal dokumen yang relevan terhadap query $\boldsymbol{q}_1$ adalah $\{\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_3, \boldsymbol{d}_4\}$

Jika ranking yang diberikan oleh suatu fungsi ranking adalah $\{\boldsymbol{d}_1, \boldsymbol{d}_4, \boldsymbol{d}_3, \boldsymbol{d}_6, \boldsymbol{d}_2, \boldsymbol{d}_5\}$, maka:

- P@1 = 1/1, P@2 = 2/2, P@3 = 3/3, P@4 = 3/4, P@5 = 4/5, P@6 = 4/6

SVM untuk Ranking

Satuan Ukuran Akurasi: MAP

- *Mean Average Precision* (MAP):

Misal himpunan dokumen yang relevan terhadap query $\boldsymbol{q}_j \in Q$ adalah $\{\boldsymbol{d}_1, \boldsymbol{d}_2, .., \boldsymbol{d}_{mj}\}$, dan R_{jk} adalah himpunan hasil retrival yang diranking dari atas sampai ke dokumen $\boldsymbol{d}_k$, maka:

$$MAP = \frac{1}{|Q|}\sum_{j=1}^{|Q|}\frac{1}{m_j}\sum_{k=1}^{m_j} P@R_{jk}$$

dimana $P@R_{jk}$ adalah Precision dari R_{jk}, yaitu

$$P@R_{jk} = \frac{jumlah\ web\ yang\ relevan\ pada\ R_{jk}}{|R_{jk}|}$$

SVM untuk Ranking

Satuan Ukuran Akurasi: MAP

Contoh:

Misal dokumen yang relevan terhadap query $\boldsymbol{q}_1$ adalah $\{\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_3, \boldsymbol{d}_4\}$

Jika ranking yang diberikan oleh suatu fungsi ranking adalah $\{\boldsymbol{d}_1, \boldsymbol{d}_4, \boldsymbol{d}_3, \boldsymbol{d}_6, \boldsymbol{d}_2, \boldsymbol{d}_5\}$, maka:

- $P@R_{11} = 1/1$, $P@_{21} = 4/5$, $P@R_{31} = 3/3$, $P@R_{41} = 2/2$
- Sehingga *Average Precision* (AP) adalah (1+4/5+1+1)/4 = 0.95
- Dan MAP 0.95/1 = 0.95, karena hanya ada satu query

SVM untuk Ranking

Satuan Ukuran Akurasi: NDCG@n

- *Normalized Discount Cumulative Gain at Position n (NDCG@n)*

  Misal R(j,m) adalah nilai relevansi dari suatu dokumen urutan ke-m untuk query $\boldsymbol{q}_j$, maka:

$$NDCG@n = \frac{1}{|Q|}\sum_{j=1}^{|Q|} Z_n \sum_{m=1}^{n} \frac{2^{R(j,m)}-1}{\log(1+m)}$$

  dimana Z_n adalah faktor normalisasi yang dihitung untuk membuat sedemikian rupa sehingga ranking sempurna pada posisi n akan bernilai 1

SVM untuk Ranking

Satuan Ukuran Akurasi: NDCG@n

Contoh:

Diberikan nilai relevansi beberapa URL untuk suatu query sbb:

	URL	Relevansi
d_1	http://abc.go.com/	Perfect: 5
d_2	http://abcnews.go.com/sections/	Excellent: 4
d_3	http://www.abc.net.au/	Excellent: 4
d_4	http://abcnews.go.com	Excellent: 4
d_5	http://www.abcteach.com/	Fair: 2

Tentukan nilai NDCG@5 untuk urutan $\{d_1, d_5, d_2, d_3, d_4\}$?

SVM untuk Ranking

Satuan Ukuran Akurasi: NDCG@n

– Gain dan Cumulative Gain

	URL	Gain	CG
#1	http://abc.go.com/	Perfect: $31=2^5-1$	31
#2	http://www.abcteach.com/	Fair: $3=2^2-1$	34 = 31 +3
#3	http://abcnews.go.com/sections/	Excellent: $15=2^4-1$	49 = 31 + 3 + 15
#4	http://www.abc.net.au/	Excellent: 15	64 = 31 + 3 + 15 + 15
#5	http://abcnews.go.com/	Excellent: 15	79 = 31 + 3 + 15 + 15 + 15

SVM untuk Ranking

Satuan Ukuran Akurasi: NDCG@n

– Discounting Factor: Log(2) / log(1 + rank)

	URL	Gain	DCG
#1	http://abc.go.com/	Perfect: $31=2^5-1$	31 = 31x1
#2	http://www.abcteach.com/	Fair: $3=2^2-1$	32.9 = 31 + 3x0.63
#3	http://abcnews.go.com/sections/	Excellent: $15=2^4-1$	40.4 = 32.9 + 15x0.50
#4	http://www.abc.net.au/	Excellent: 15	46.9 = 40.4 + 15x0.43
#5	http://abcnews.go.com/	Excellent: 15	52.7 = 46.9 + 15x0.39

SVM untuk Ranking

Satuan Ukuran Akurasi: NDCG@n

– Normalized factor: nilai maksimum yang mungkin untuk suatu urutan tertentu

	URL	Gain	DCG	Max DCG
#1	http://abc.go.com/	Perfect: $31=2^5-1$	31	31 = 31x1
#2	http://www.abcteach.com/	Fair: $3=2^2-1$	32.9	40.5 = 31 + 15x0.63
#3	http://abcnews.go.com/sections/	Excellent: $15=2^4-1$	40.4	48.0 = 40.5 + 15x0.50
#4	http://www.abc.net.au/	Excellent: 15	46.9	54.5 = 48.0 + 15x0.43
#5	http://abcnews.go.com/	Excellent: 15	52.7	60.4 = 54.5 + 15x0.39

SVM untuk Ranking

Satuan Ukuran Akurasi: NDCG@n

– Normalized Discounting Cumulative Gain

	URL	Gain	DCG	Max DCG	NDCG
#1	http://abc.go.com/	Perfect: $31=2^5-1$	31	31	1 = 31/31
#2	http://www.abcteach.com/	Fair: $3=2^2-1$	32.9	40.5	0.81=32.9/40.5
#3	http://abcnews.go.com/sections/	Excellent: $15=2^4-1$	40.4	48.0	0.84=40.4/48.0
#4	http://www.abc.net.au/	Excellent: 15	46.9	54.5	0.86=46.9/54.5
#5	http://abcnews.go.com/	Excellent: 15	52.7	60.4	0.87=52.7/60.4

SVM untuk Ranking

Desain Eksperimen

- Eksperimen menggunakan 5-fold Cross Validation dengan statistik data sbb:

FOLD	TRAINING	VALIDATION	TEST
Fold 1	{S1, S2, S3}	S4	S5
Fold 2	{S2, S3, S4}	S5	S1
Fold 3	{S3, S4, S5}	S1	S2
Fold 4	{S4, S5, S1}	S2	S3
Fold 5	{S5, S1, S2}	S3	S4

FOLD	TRAINING	VALIDATION	TEST
Fold 1	9630	2707	2874
Fold 2	9404	2874	2933
Fold 3	8643	2933	3635
Fold 4	8514	3635	3062
Fold 5	9442	3062	2707

SVM untuk Ranking

Hasil Eksperimen

- Tingkat akurasi metode SVM untuk Ranking dengan menggunakan perangkat lunak SVM^{Rank}

Performance on testing set

Para	NDCG@1	NDCG@2	NDCG@3	NDCG@4	NDCG@5	NDCG@6	NDCG@7	NDCG@8	NDCG@9	NDCG@10	MeanNDCG
10	0.3611	0.3777	0.4088	0.4248	0.4469	0.4659	0.4738	0.4215	0.2078	0.2117	0.4577
2	0.3036	0.3450	0.3761	0.4018	0.4224	0.4395	0.4406	0.4230	0.1718	0.1738	0.4296
1	0.3461	0.3710	0.4082	0.4354	0.4452	0.4596	0.4664	0.4489	0.2433	0.2494	0.4686
0.2	0.4161	0.4682	0.4909	0.5078	0.5290	0.5391	0.5428	0.4955	0.2860	0.2892	0.5442
0.5	0.3864	0.4305	0.4589	0.4845	0.5042	0.5215	0.5290	0.4933	0.2107	0.2155	0.5159
Avg	0.36266	0.39848	0.42858	0.45086	0.46954	0.48512	0.49052	0.45644	0.22392	0.22792	0.4832

Para	P@1	P@2	P@3	P@4	P@5	P@6	P@7	P@8	P@9	P@10	MAP
10	0.4167	0.3942	0.3868	0.3654	0.3436	0.3216	0.2949	0.2748	0.2564	0.2423	0.4502
2	0.3631	0.3471	0.3397	0.3264	0.3121	0.2972	0.2748	0.2540	0.2385	0.2229	0.4213
1	0.4013	0.3758	0.3694	0.3487	0.3223	0.3015	0.2830	0.2675	0.2484	0.2357	0.4529
0.2	0.4968	0.4904	0.4544	0.4299	0.4089	0.3864	0.3576	0.3368	0.3171	0.2981	0.5284
0.5	0.4586	0.4268	0.4013	0.3774	0.3503	0.3259	0.3003	0.2779	0.2633	0.2465	0.4950
Avg	0.4273	0.40686	0.39032	0.36956	0.34744	0.32652	0.30212	0.2822	0.26474	0.2491	0.46956

sumber: http://research.microsoft.com/en-us/um/beijing/projects/letor/letor4baseline.aspx

Referensi

(1) T. Joachims. *Optimizing Searching Engines Using Clickthrough Data*. Proceeding of the ACM Conference on Knowledge Discovery and Data Mining, 2002

(2) T. Joachim. *Making Large-Scale SVM Learning Practical*. In B. Schoelkopf, C. J. C. Burges, and A. J. Smola (Eds), Advances in Kernel Methods – Support Vector Learning, pp. 169-184, MIT Press, 1999

(3) T. Joachims. *Training Linear SVMs in Linear Time*. Proceeding of the ACM Conference on Knowledge Discovery and Data Mining, 2006