Organização e Recuperação de Informação: Aprendizado de ranking

Marcelo K. A.

Faculdade de Computação - Universidade Federal de Uberlândia

22 de janeiro de 2015

Overview

Índice

Índice

- Ideia de aprendizado de ranking: aprender a pontuação de relevância de documentos em relação a consultas

Índice

- Ideia de aprendizado de ranking: aprender a pontuação de relevância de documentos em relação a consultas
- Pontuação de campos: método simples

Índice

- Ideia de aprendizado de ranking: aprender a pontuação de relevância de documentos em relação a consultas
- Pontuação de campos: método simples
- Pontuação por aprendizado de máquina

Índice

- Ideia de aprendizado de ranking: aprender a pontuação de relevância de documentos em relação a consultas
- Pontuação de campos: método simples
- Pontuação por aprendizado de máquina
- Pontuação usando SVMs

Conteúdo

Ideia principal

- A meta do uso de pesos para termos (por exemplo, tf-idf) é medir a importância deles

Ideia principal

- A meta do uso de pesos para termos (por exemplo, tf-idf) é medir a importância deles
  - A soma dos pesos de termos é uma medida de relevância de um documento e a base para montar rankings

Ideia principal

- A meta do uso de pesos para termos (por exemplo, tf-idf) é medir a importância deles
  - A soma dos pesos de termos é uma medida de relevância de um documento e a base para montar rankings
- Esse problema consiste em "aprender a atribuir pesos", ou seja, aprender a montar rankings.

Ideia principal

- A meta do uso de pesos para termos (por exemplo, tf-idf) é medir a importância deles
  - A soma dos pesos de termos é uma medida de relevância de um documento e a base para montar rankings
- Esse problema consiste em "aprender a atribuir pesos", ou seja, aprender a montar rankings.
  - Pesos de termos podem ser aprendidos usando exemplos de treino previamente julgados

Ideia principal

- A meta do uso de pesos para termos (por exemplo, tf-idf) é medir a importância deles
  - A soma dos pesos de termos é uma medida de relevância de um documento e a base para montar rankings
- Esse problema consiste em "aprender a atribuir pesos", ou seja, aprender a montar rankings.
  - Pesos de termos podem ser aprendidos usando exemplos de treino previamente julgados
- Essa metodologia pertence à área de pesquisa de Aprendizado de Máquina aplicado a ranking

Aprendendo pesos

Metodologia principal

- Dado um conjunto de exemplos de treinamento, cada qual é uma tupla de: uma consulta q, um documento d e um julgamento de relevância de d para q

Aprendendo pesos

Metodologia principal

- Dado um conjunto de exemplos de treinamento, cada qual é uma tupla de: uma consulta q, um documento d e um julgamento de relevância de d para q
  - Caso simples: $R(d, q)$ é ou relevante (1) ou não-relevante (0)

Aprendendo pesos

Metodologia principal

- Dado um conjunto de exemplos de treinamento, cada qual é uma tupla de: uma consulta q, um documento d e um julgamento de relevância de d para q
  - Caso simples: $R(d, q)$ é ou relevante (1) ou não-relevante (0)
  - Mais sofisticado: julgamentos avaliados em níveis (por exemplo, de 0 a 10)

Aprendendo pesos

Metodologia principal

- Dado um conjunto de exemplos de treinamento, cada qual é uma tupla de: uma consulta q, um documento d e um julgamento de relevância de d para q
  - Caso simples: $R(d, q)$ é ou relevante (1) ou não-relevante (0)
  - Mais sofisticado: julgamentos avaliados em níveis (por exemplo, de 0 a 10)
- Aprendizado de pesos a partir desses exemplos, de tal forma que as pontuações aprendidas aproximam os julgamentos de relevância observados nos exempos de treino

Modelo de independência binária (BIM)

- BIM é uma forma de aprendizado de ranking?

Modelo de independência binária (BIM)

- BIM é uma forma de aprendizado de ranking?
- BIM:

Modelo de independência binária (BIM)

- BIM é uma forma de aprendizado de ranking?
- BIM:
  - Estimar classificador de probabilidade de relevância em conjunto de treino

Modelo de independência binária (BIM)

- BIM é uma forma de aprendizado de ranking?
- BIM:
  - Estimar classificador de probabilidade de relevância em conjunto de treino
  - Aplicar para todos os documentos

Modelo de independência binária (BIM)

- BIM é uma forma de aprendizado de ranking?
- BIM:
  - Estimar classificador de probabilidade de relevância em conjunto de treino
  - Aplicar para todos os documentos
  - Rankear docs de acordo com probabilidade de relevância

Aprendizado de ranking vs. classificação de textos

- Ambos são abordagens de aprendizado de máquina

Aprendizado de ranking vs. classificação de textos

- Ambos são abordagens de aprendizado de máquina
- Classificação de textos, BIM e retorno de relevância são específicos para consultas

Aprendizado de ranking vs. classificação de textos

- Ambos são abordagens de aprendizado de máquina
- Classificação de textos, BIM e retorno de relevância são específicos para consultas
  - Necessitamos um conjunto de treino específico para consultas para aprender o rankeador

Aprendizado de ranking vs. classificação de textos

- Ambos são abordagens de aprendizado de máquina
- Classificação de textos, BIM e retorno de relevância são específicos para consultas
  - Necessitamos um conjunto de treino específico para consultas para aprender o rankeador
  - Precisamos aprender um novo rankeador para cada consulta

Aprendizado de ranking vs. classificação de textos

- Ambos são abordagens de aprendizado de máquina
- Classificação de textos, BIM e retorno de relevância são específicos para consultas
  - Necessitamos um conjunto de treino específico para consultas para aprender o rankeador
  - Precisamos aprender um novo rankeador para cada consulta
- Aprendizado de ranking normalmente refere-se para ranking independente a consultas

Aprendizado de ranking vs. classificação de textos

- Ambos são abordagens de aprendizado de máquina
- Classificação de textos, BIM e retorno de relevância são específicos para consultas
  - Necessitamos um conjunto de treino específico para consultas para aprender o rankeador
  - Precisamos aprender um novo rankeador para cada consulta
- Aprendizado de ranking normalmente refere-se para ranking independente a consultas
- Aprendemos um único classificador

Aprendizado de ranking vs. classificação de textos

- Ambos são abordagens de aprendizado de máquina
- Classificação de textos, BIM e retorno de relevância são específicos para consultas
  - Necessitamos um conjunto de treino específico para consultas para aprender o rankeador
  - Precisamos aprender um novo rankeador para cada consulta
- Aprendizado de ranking normalmente refere-se para ranking independente a consultas
- Aprendemos um único classificador
- Podemos então rankear docs para uma consulta que não temos qualquer julgamento de relevâcia

Aprendizado de ranking como um problema de classificação

- Uma abordagem para aprendizado de ranking é representar cada par consulta-documento como um vetor

Aprendizado de ranking como um problema de classificação

- Uma abordagem para aprendizado de ranking é representar cada par consulta-documento como um vetor
- Temos duas classes

Aprendizado de ranking como um problema de classificação

- Uma abordagem para aprendizado de ranking é representar cada par consulta-documento como um vetor
- Temos duas classes
  - Classe 1: a consulta é relevante para o doc

Aprendizado de ranking como um problema de classificação

- Uma abordagem para aprendizado de ranking é representar cada par consulta-documento como um vetor
- Temos duas classes
  - Classe 1: a consulta é relevante para o doc
  - Classe 2: a consulta não relevante para o doc

Aprendizado de ranking como um problema de classificação

- Uma abordagem para aprendizado de ranking é representar cada par consulta-documento como um vetor
- Temos duas classes
  - Classe 1: a consulta é relevante para o doc
  - Classe 2: a consulta não relevante para o doc
- Este é um problema de classificação padrão, exceto que os vetores são pares consulta-documento (e não somente documentos)

Aprendizado de ranking como um problema de classificação

- Uma abordagem para aprendizado de ranking é representar cada par consulta-documento como um vetor
- Temos duas classes
  - Classe 1: a consulta é relevante para o doc
  - Classe 2: a consulta não relevante para o doc
- Este é um problema de classificação padrão, exceto que os vetores são pares consulta-documento (e não somente documentos)
- Docs são rankeados de acordo com a probabilidade de relevância de correspondência de pares documento-consulta

Aprendizado de ranking como um problema de classificação

- Uma abordagem para aprendizado de ranking é representar cada par consulta-documento como um vetor
- Temos duas classes
  - Classe 1: a consulta é relevante para o doc
  - Classe 2: a consulta não relevante para o doc
- Este é um problema de classificação padrão, exceto que os vetores são pares consulta-documento (e não somente documentos)
- Docs são rankeados de acordo com a probabilidade de relevância de correspondência de pares documento-consulta
- Quais características/dimensões deve-se usar para representar um par consulta-documento?

Pontuação de campos

- Temos: uma coleção onde docs tem três campos na qual documentos tem três campos: `autor`, `título`, `corpo`

Pontuação de campos

- Temos: uma coleção onde docs tem três campos na qual documentos tem três campos: `autor`, `título`, `corpo`
- Pontuação de campos com pesos requer um peso separado para cada campo, por exemplo, g_1, g_2, g_3

Pontuação de campos

- Temos: uma coleção onde docs tem três campos na qual documentos tem três campos: `autor`, `título`, `corpo`
- Pontuação de campos com pesos requer um peso separado para cada campo, por exemplo, g_1, g_2, g_3
- Nem todos os campos são igualmente importantes: exemplo: `autor` $<$ `título` $<$ `corpo`
  $\rightarrow g_1 = 0.2, g_2 = 0.3, g_3 = 0.5$ (soma igual a 1)

Pontuação de campos

- Temos: uma coleção onde docs tem três campos na qual documentos tem três campos: `autor`, `título`, `corpo`
- Pontuação de campos com pesos requer um peso separado para cada campo, por exemplo, g_1, g_2, g_3
- Nem todos os campos são igualmente importantes: exemplo: `autor` $<$ `título` $<$ `corpo`
  $\rightarrow g_1 = 0.2, g_2 = 0.3, g_3 = 0.5$ (soma igual a 1)
- Pontuação para um campo é 1 se o termo da consulta ocorre no campo e 0 caso contrário (Booleano)

Pontuação de campos

- Temos: uma coleção onde docs tem três campos na qual documentos tem três campos: `autor`, `título`, `corpo`
- Pontuação de campos com pesos requer um peso separado para cada campo, por exemplo, g_1, g_2, g_3
- Nem todos os campos são igualmente importantes: exemplo: `autor` $<$ `título` $<$ `corpo`
  $\rightarrow g_1 = 0.2, g_2 = 0.3, g_3 = 0.5$ (soma igual a 1)
- Pontuação para um campo é 1 se o termo da consulta ocorre no campo e 0 caso contrário (Booleano)

Exemplo

Termo da consulta aparece no `título` e `corpo` somente
Pontuação de documento: $(0.3 \cdot 1) + (0.5 \cdot 1) = 0.8$.

Forma geral de pontuação de campos com pesos

Para uma consulta q e um doc d, a pontuação de pesos por campo atribue ao par (q, d) uma pontuação no intervalo [0,1] ao computar uma combinação linear das pontuações dos campos do doc, onde cada campo contribui com um valor.

- Considere um conjunto de docs, que tem L campos

Forma geral de pontuação de campos com pesos

Para uma consulta q e um doc d, a pontuação de pesos por campo atribue ao par (q, d) uma pontuação no intervalo [0,1] ao computar uma combinação linear das pontuações dos campos do doc, onde cada campo contribui com um valor.

- Considere um conjunto de docs, que tem L campos
- Seja $g_1, ..., g_L \in [0, 1]$, tal que $\sum_{i=1}^{L} g_i = 1$

Forma geral de pontuação de campos com pesos

Para uma consulta q e um doc d, a pontuação de pesos por campo atribue ao par (q, d) uma pontuação no intervalo [0,1] ao computar uma combinação linear das pontuações dos campos do doc, onde cada campo contribui com um valor.

- Considere um conjunto de docs, que tem L campos
- Seja $g_1, ..., g_L \in [0, 1]$, tal que $\sum_{i=1}^{L} g_i = 1$
- Para $1 \leq i \leq L$, seja s_i a pontuação booleana denotando uma correspondencia (ou não) entre q e o i-ésimo campo i

Forma geral de pontuação de campos com pesos

Para uma consulta q e um doc d, a pontuação de pesos por campo atribue ao par (q, d) uma pontuação no intervalo [0,1] ao computar uma combinação linear das pontuações dos campos do doc, onde cada campo contribui com um valor.

- Considere um conjunto de docs, que tem L campos
- Seja $g_1, ..., g_L \in [0, 1]$, tal que $\sum_{i=1}^{L} g_i = 1$
- Para $1 \leq i \leq L$, seja s_i a pontuação booleana denotando uma correspondencia (ou não) entre q e o i-ésimo campo i
  - $s_i = 1$ se um termo da consulta ocorre no campo i, 0 caso contrário

Forma geral de pontuação de campos com pesos

Para uma consulta q e um doc d, a pontuação de pesos por campo atribue ao par (q, d) uma pontuação no intervalo [0,1] ao computar uma combinação linear das pontuações dos campos do doc, onde cada campo contribui com um valor.

- Considere um conjunto de docs, que tem L campos
- Seja $g_1, ..., g_L \in [0, 1]$, tal que $\sum_{i=1}^{L} g_i = 1$
- Para $1 \leq i \leq L$, seja s_i a pontuação booleana denotando uma correspondencia (ou não) entre q e o i-ésimo campo i
  - $s_i = 1$ se um termo da consulta ocorre no campo i, 0 caso contrário

Pontuação de campos com pesos, ou seja, Recuperação booleana com ranking

Ordenar docs de acordo com $\sum_{i=1}^{L} g_i s_i$

Aprendizado de pesos para pontuação de campos com pesos

- Pontuação de campos com pesos pode ser visto como o aprendizado de uma função linear das pontuação de correspondência booleana contribuídas por vários campos

Aprendizado de pesos para pontuação de campos com pesos

- Pontuação de campos com pesos pode ser visto como o aprendizado de uma função linear das pontuação de correspondência booleana contribuídas por vários campos
- Custo: montagem dos julgamentos de relevância exige bastante trabalho de especialistas/usuários

Aprendizado de pesos para pontuação de campos com pesos

- Pontuação de campos com pesos pode ser visto como o aprendizado de uma função linear das pontuação de correspondência booleana contribuídas por vários campos
- Custo: montagem dos julgamentos de relevância exige bastante trabalho de especialistas/usuários
  - Especialmente em uma coleção dinâmica como a Web

Aprendizado de pesos para pontuação de campos com pesos

- Pontuação de campos com pesos pode ser visto como o aprendizado de uma função linear das pontuação de correspondência booleana contribuídas por vários campos
- Custo: montagem dos julgamentos de relevância exige bastante trabalho de especialistas/usuários
  - Especialmente em uma coleção dinâmica como a Web
  - Os principais buscadores investem bastante na criação de grandes conjuntos de treino para aprendizado de ranking

Aprendizado de pesos para pontuação de campos com pesos

- Pontuação de campos com pesos pode ser visto como o aprendizado de uma função linear das pontuação de correspondência booleana contribuídas por vários campos
- Custo: montagem dos julgamentos de relevância exige bastante trabalho de especialistas/usuários
  - Especialmente em uma coleção dinâmica como a Web
  - Os principais buscadores investem bastante na criação de grandes conjuntos de treino para aprendizado de ranking
- Por outro lado: uma vez temos um conjunto de treino grande o suficiente, o problema de aprendizado de pesos g_i reduz para problemas de otimização simples

Aprendizado de pesos para pontuação de campos com pesos: caso simples

- Assuma que docs tem dois campos: `título, corpo`

Aprendizado de pesos para pontuação de campos com pesos: caso simples

- Assuma que docs tem dois campos: `título, corpo`
- Fórmula de pontuação de campos com pesos que vimos antes:

$$\sum_{i=1}^{L} g_i s_i$$

Aprendizado de pesos para pontuação de campos com pesos: caso simples

- Assuma que docs tem dois campos: `título, corpo`
- Fórmula de pontuação de campos com pesos que vimos antes:

$$\sum_{i=1}^{L} g_i s_i$$

- Para q, d, temos $s_T(d, q) = 1$ se um termo de consulta ocorre no título, 0 caso contrário ; $s_B(d, q) = 1$ se um termo de consulta ocorre no corpo, 0 caso contrário

Aprendizado de pesos para pontuação de campos com pesos: caso simples

- Assuma que docs tem dois campos: `título, corpo`
- Fórmula de pontuação de campos com pesos que vimos antes:

$$\sum_{i=1}^{L} g_i s_i$$

- Para q, d, temos $s_T(d, q) = 1$ se um termo de consulta ocorre no título, 0 caso contrário ; $s_B(d, q) = 1$ se um termo de consulta ocorre no corpo, 0 caso contrário
- Computamos uma pontuação entre 0 e 1 para cada par (d, q) usando $s_T(d, q)$ e $s_B(d, q)$ usando uma constante $g \in [0, 1]$:

$$pontuacao(d, q) = g \cdot s_T(d, q) + (1 - g) \cdot s_B(d, q)$$

Aprendizado de pesos: obter g usando exemplos de treino

Φ_j	d_j	q_j	s_T	s_B	$r(d_j, q_j)$
Φ_1	37	linux	1	1	relevante
Φ_2	37	penguin	0	1	não-relevante
Φ_3	238	system	0	1	relevante
Φ_4	238	penguin	0	0	não-relevante
Φ_5	1741	kernel	1	1	relevante
Φ_6	2094	driver	0	1	relevante
Φ_7	3194	driver	1	0	não-relevante

Aprendizado de pesos: obter g usando exemplos de treino

Φ_j	d_j	q_j	s_T	s_B	$r(d_j, q_j)$
Φ_1	37	linux	1	1	relevante
Φ_2	37	penguin	0	1	não-relevante
Φ_3	238	system	0	1	relevante
Φ_4	238	penguin	0	0	não-relevante
Φ_5	1741	kernel	1	1	relevante
Φ_6	2094	driver	0	1	relevante
Φ_7	3194	driver	1	0	não-relevante

- Exemplos de treino: tuplas da forma: $\Phi_j = (d_j, q_j, r(d_j, q_j))$

Aprendizado de pesos: obter g usando exemplos de treino

Φ_j	d_j	q_j	s_T	s_B	$r(d_j, q_j)$
Φ_1	37	linux	1	1	relevante
Φ_2	37	penguin	0	1	não-relevante
Φ_3	238	system	0	1	relevante
Φ_4	238	penguin	0	0	não-relevante
Φ_5	1741	kernel	1	1	relevante
Φ_6	2094	driver	0	1	relevante
Φ_7	3194	driver	1	0	não-relevante

- Exemplos de treino: tuplas da forma: $\Phi_j = (d_j, q_j, r(d_j, q_j))$
- Temos docs de treino d_j e consultas de treino q_j avaliados por especialistas que decidem $r(d_j, q_j)$ (relevante ou não-relevante)

Aprendizado de pesos: obter g a partir dos exemplos de treino

Exemplo	DocID	Consulta	s_T	s_B	Julgamento
Φ_1	37	linux	1	1	relevante
Φ_2	37	penguin	0	1	não-relevante
Φ_3	238	system	0	1	relevante
Φ_4	238	penguin	0	0	não-relevante
Φ_5	1741	kernel	1	1	relevante
Φ_6	2094	driver	0	1	relevante
Φ_7	3194	driver	1	0	não-relevante

Aprendizado de pesos: obter g a partir dos exemplos de treino

Exemplo	DocID	Consulta	s_T	s_B	Julgamento
Φ_1	37	linux	1	1	relevante
Φ_2	37	penguin	0	1	não-relevante
Φ_3	238	system	0	1	relevante
Φ_4	238	penguin	0	0	não-relevante
Φ_5	1741	kernel	1	1	relevante
Φ_6	2094	driver	0	1	relevante
Φ_7	3194	driver	1	0	não-relevante

- Para cada exemplo de treino Φ_j temos valores booleanos $s_T(d_j, q_j)$ e $s_B(d_j, q_j)$ que usamos para computar uma pontuação dependente do valor g:

$$pontuacao(d_j, q_j) = g \cdot s_T(d_j, q_j) + (1 - g) \cdot s_B(d_j, q_j)$$

Aprendizado de pesos

- Comparamos a $pontuacao(d_j, q_j)$ com julgamentos de especialistas para o mesmo par documento-consulta (d_j, q_j).

Aprendizado de pesos

- Comparamos a $pontuacao(d_j, q_j)$ com julgamentos de especialistas para o mesmo par documento-consulta (d_j, q_j).
- Definimos o erro da função de pontuação com peso g a seguir:

$$erro(g, \Phi_j) = (r(d_j, q_j) - pontuacao(d_j, q_j))^2$$

Aprendizado de pesos

- Comparamos a $pontuacao(d_j, q_j)$ com julgamentos de especialistas para o mesmo par documento-consulta (d_j, q_j).
- Definimos o erro da função de pontuação com peso g a seguir:

$$erro(g, \Phi_j) = (r(d_j, q_j) - pontuacao(d_j, q_j))^2$$

- Então o erro total nos exemplos de treino é dado por

$$\sum_j erro(g, \Phi_j)$$

Aprendizado de pesos

- Comparamos a $pontuacao(d_j, q_j)$ com julgamentos de especialistas para o mesmo par documento-consulta (d_j, q_j).
- Definimos o erro da função de pontuação com peso g a seguir:

$$erro(g, \Phi_j) = (r(d_j, q_j) - pontuacao(d_j, q_j))^2$$

- Então o erro total nos exemplos de treino é dado por

$$\sum_j erro(g, \Phi_j)$$

- O problema de aprendizado de ranking é resolvido pela escolha do valor g que minimiza o erro total considerando os exemplos de treino.

Minimizar o erro total *erro*: Exemplo (1)

Exemplos de treino

Exemplo	DocID	Consulta	s_T	s_B	Julgamento
Φ_1	37	linux	1	1	1 (relevante)
Φ_2	37	penguin	0	1	0 (não-relevante)
Φ_3	238	system	0	1	1 (relevante)
Φ_4	238	penguin	0	0	0 (não-relevante)
Φ_5	1741	kernel	1	1	1 (relevante)
Φ_6	2094	driver	0	1	1 (relevante)
Φ_7	3194	driver	1	0	0 (não-relevante)

Minimizar o erro total *erro*: Exemplo (1)

Exemplos de treino

Exemplo	DocID	Consulta	s_T	s_B	Julgamento
Φ_1	37	linux	1	1	1 (relevante)
Φ_2	37	penguin	0	1	0 (não-relevante)
Φ_3	238	system	0	1	1 (relevante)
Φ_4	238	penguin	0	0	0 (não-relevante)
Φ_5	1741	kernel	1	1	1 (relevante)
Φ_6	2094	driver	0	1	1 (relevante)
Φ_7	3194	driver	1	0	0 (não-relevante)

- Computar pontuação:
  $pontuacao(d_j, q_j) = g \cdot s_T(d_j, q_j) + (1 - g) \cdot s_B(d_j, q_j)$

Minimizar o erro total *erro*: Exemplo (1)

Exemplos de treino

Exemplo	DocID	Consulta	s_T	s_B	Julgamento
Φ_1	37	linux	1	1	1 (relevante)
Φ_2	37	penguin	0	1	0 (não-relevante)
Φ_3	238	system	0	1	1 (relevante)
Φ_4	238	penguin	0	0	0 (não-relevante)
Φ_5	1741	kernel	1	1	1 (relevante)
Φ_6	2094	driver	0	1	1 (relevante)
Φ_7	3194	driver	1	0	0 (não-relevante)

- Computar pontuação:
  $pontuacao(d_j, q_j) = g \cdot s_T(d_j, q_j) + (1 - g) \cdot s_B(d_j, q_j)$
- Computar erro total: $\sum_j erro(g, \Phi_j)$, onde
  $erro(g, \Phi_j) = (r(d_j, q_j) - pontuacao(d_j, q_j))^2$

Minimizar o erro total *erro*: Exemplo (1)

Exemplos de treino

Exemplo	DocID	Consulta	s_T	s_B	Julgamento
Φ_1	37	linux	1	1	1 (relevante)
Φ_2	37	penguin	0	1	0 (não-relevante)
Φ_3	238	system	0	1	1 (relevante)
Φ_4	238	penguin	0	0	0 (não-relevante)
Φ_5	1741	kernel	1	1	1 (relevante)
Φ_6	2094	driver	0	1	1 (relevante)
Φ_7	3194	driver	1	0	0 (não-relevante)

- Computar pontuação:
  $pontuacao(d_j, q_j) = g \cdot s_T(d_j, q_j) + (1 - g) \cdot s_B(d_j, q_j)$
- Computar erro total: $\sum_j erro(g, \Phi_j)$, onde
  $erro(g, \Phi_j) = (r(d_j, q_j) - pontuacao(d_j, q_j))^2$
- Escolher o valor de g que minimiza o erro total

Minimizar o erro total *erro*: Exemplo (2)

- Computar pontuação $pontuacao(d_j, q_j)$

$$pontuacao(d_1, q_1) = g \cdot 1 + (1 - g) \cdot 1 = g + 1 - g = 1$$
$$pontuacao(d_2, q_2) = g \cdot 0 + (1 - g) \cdot 1 = 0 + 1 - g = 1 - g$$
$$pontuacao(d_3, q_3) = g \cdot 0 + (1 - g) \cdot 1 = 0 + 1 - g = 1 - g$$
$$pontuacao(d_4, q_4) = g \cdot 0 + (1 - g) \cdot 0 = 0 + 0 = 0$$
$$pontuacao(d_5, q_5) = g \cdot 1 + (1 - g) \cdot 1 = g + 1 - g = 1$$
$$pontuacao(d_6, q_6) = g \cdot 0 + (1 - g) \cdot 1 = 0 + 1 - g = 1 - g$$
$$pontuacao(d_7, q_7) = g \cdot 1 + (1 - g) \cdot 0 = g + 0 = g$$

Minimizar o erro total *erro*: Exemplo (2)

- Computar pontuação $pontuacao(d_j, q_j)$

$$pontuacao(d_1, q_1) = g \cdot 1 + (1-g) \cdot 1 = g + 1 - g = 1$$
$$pontuacao(d_2, q_2) = g \cdot 0 + (1-g) \cdot 1 = 0 + 1 - g = 1 - g$$
$$pontuacao(d_3, q_3) = g \cdot 0 + (1-g) \cdot 1 = 0 + 1 - g = 1 - g$$
$$pontuacao(d_4, q_4) = g \cdot 0 + (1-g) \cdot 0 = 0 + 0 = 0$$
$$pontuacao(d_5, q_5) = g \cdot 1 + (1-g) \cdot 1 = g + 1 - g = 1$$
$$pontuacao(d_6, q_6) = g \cdot 0 + (1-g) \cdot 1 = 0 + 1 - g = 1 - g$$
$$pontuacao(d_7, q_7) = g \cdot 1 + (1-g) \cdot 0 = g + 0 = g$$

- Computar erro total $\sum_j erro(g, \Phi_j)$

$$(1-1)^2+(0-1+g)^2+(1-1+g)^2+(0-0)^2+(1-1)^2+(1-1+g)^2+(0-g)^2 = 0+(-1+g)^2+g^2+0+0+g^2+g^2 = 1-2g+4g^2$$

Minimizar o erro total *erro*: Exemplo (2)

- Computar pontuação $pontuacao(d_j, q_j)$

  $pontuacao(d_1, q_1) = g \cdot 1 + (1-g) \cdot 1 = g + 1 - g = 1$
  $pontuacao(d_2, q_2) = g \cdot 0 + (1-g) \cdot 1 = 0 + 1 - g = 1 - g$
  $pontuacao(d_3, q_3) = g \cdot 0 + (1-g) \cdot 1 = 0 + 1 - g = 1 - g$
  $pontuacao(d_4, q_4) = g \cdot 0 + (1-g) \cdot 0 = 0 + 0 = 0$
  $pontuacao(d_5, q_5) = g \cdot 1 + (1-g) \cdot 1 = g + 1 - g = 1$
  $pontuacao(d_6, q_6) = g \cdot 0 + (1-g) \cdot 1 = 0 + 1 - g = 1 - g$
  $pontuacao(d_7, q_7) = g \cdot 1 + (1-g) \cdot 0 = g + 0 = g$

- Computar erro total $\sum_j erro(g, \Phi_j)$

  $(1-1)^2+(0-1+g)^2+(1-1+g)^2+(0-0)^2+(1-1)^2+(1-1+g)^2+(0-g)^2 = 0+(-1+g)^2+g^2+0+0+g^2+g^2 = 1-2g+4g^2$

- Escolher o valor de g que minimiza o erro total

  Obtendo a derivada igual a 0, obtém-se o mínimo em $g = \frac{1}{4}$.

Peso g que minimiza o erro no caso geral

$$g = \frac{n_{10r} + n_{01n}}{n_{10r} + n_{10n} + n_{01r} + n_{01n}}$$

Peso g que minimiza o erro no caso geral

- $$g = \frac{n_{10r} + n_{01n}}{n_{10r} + n_{10n} + n_{01r} + n_{01n}}$$
- $n_{...}$ são as contagem de linhas do conj. de treino com as seguintes propriedades:

n_{10r}	$s_T = 1$	$s_B = 0$	documento relevante
n_{10n}	$s_T = 1$	$s_B = 0$	documento não-relevante
n_{01r}	$s_T = 0$	$s_B = 1$	documento relevante
n_{01n}	$s_T = 0$	$s_B = 1$	documento não-relevante

Peso g que minimiza o erro no caso geral

- $$g = \frac{n_{10r} + n_{01n}}{n_{10r} + n_{10n} + n_{01r} + n_{01n}}$$
- $n_{...}$ são as contagem de linhas do conj. de treino com as seguintes propriedades:

n_{10r}	$s_T = 1$	$s_B = 0$	documento relevante
n_{10n}	$s_T = 1$	$s_B = 0$	documento não-relevante
n_{01r}	$s_T = 0$	$s_B = 1$	documento relevante
n_{01n}	$s_T = 0$	$s_B = 1$	documento não-relevante

- Notar que ignoramos docs que tem pontuação 0 para os dois campos ou 1 correspondia para os dois campos – o valor de g não muda a pontuação final.

Exercício: Computar g que minimiza o erro

	DocID	Consulta	s_T	s_B	Julgamento
Φ_1	37	linux	0	0	relevante
Φ_2	37	penguin	1	1	não-relevante
Φ_3	238	system	1	0	relevante
Φ_4	238	penguin	1	1	não-relevante
Φ_5	238	redmond	0	1	não-relevante
Φ_6	1741	kernel	0	0	relevante
Φ_7	2094	driver	1	0	relevante
Φ_8	3194	driver	0	1	não-relevante
Φ_9	3194	redmond	0	0	não-relevante

Conteúdo

Metodologia de pontuação por aprendizado de máquina

- Até agora usamos indicadores booleanos de relevância

Metodologia de pontuação por aprendizado de máquina

- Até agora usamos indicadores booleanos de relevância
- Agora veremos sobre o uso de características mais gerais tais como

Metodologia de pontuação por aprendizado de máquina

- Até agora usamos indicadores booleanos de relevância
- Agora veremos sobre o uso de características mais gerais tais como
  - similaridade cosseno

Metodologia de pontuação por aprendizado de máquina

- Até agora usamos indicadores booleanos de relevância
- Agora veremos sobre o uso de características mais gerais tais como
  - similaridade cosseno
  - distância entre termos da consulta

Metodologia de pontuação por aprendizado de máquina

- Até agora usamos indicadores booleanos de relevância
- Agora veremos sobre o uso de características mais gerais tais como
  - similaridade cosseno
  - distância entre termos da consulta
  - PageRank etc.

Dois exemplos de características típicas

- Similaridade cosseno (do modelo vetorial entre consulta e documento (denotada por α)

Dois exemplos de características típicas

- Similaridade cosseno (do modelo vetorial entre consulta e documento (denotada por α)
- Tamanho da janela mínimo dentro do qual que os termos da consulta estão (denotado por ω)

Dois exemplos de características típicas

- Similaridade cosseno (do modelo vetorial entre consulta e documento (denotada por α)
- Tamanho da janela mínimo dentro do qual que os termos da consulta estão (denotado por ω)
  - Consulta por proximidade de termos é frequentemente um indicativo de relevância

Dois exemplos de características típicas

- Similaridade cosseno (do modelo vetorial entre consulta e documento (denotada por α)
- Tamanho da janela mínimo dentro do qual que os termos da consulta estão (denotado por ω)
  - Consulta por proximidade de termos é frequentemente um indicativo de relevância
- Então, temos uma característica que captura a similaridade geral consulta-documento e uma característica que captura a proximidade de termos de consulta no documento

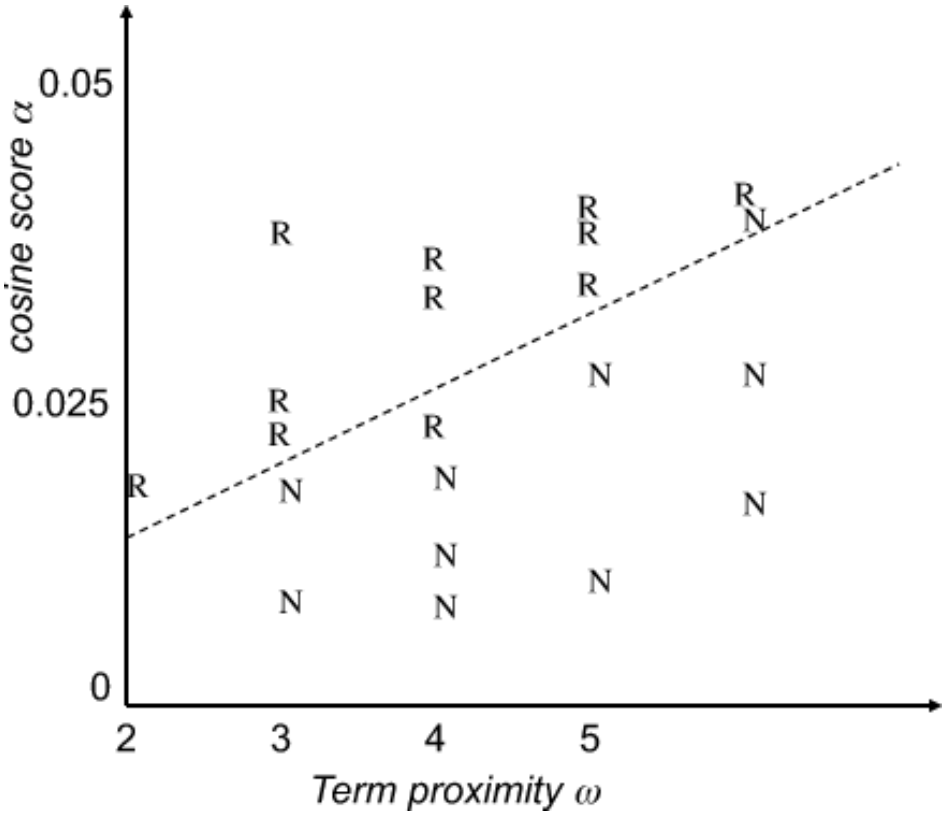
Aprendizado de ranking para essas duas características

Exemplo	DocID	Consulta	α	ω	Julgamento
Φ_1	37	linux	0.032	3	relevante
Φ_2	37	penguin	0.02	4	não-relevante
Φ_3	238	operating system	0.043	2	relevante
Φ_4	238	runtime	0.004	2	não-relevante
Φ_5	1741	kernel layer	0.022	3	relevante
Φ_6	2094	device driver	0.03	2	relevante
Φ_7	3191	device driver	0.027	5	não-relevante

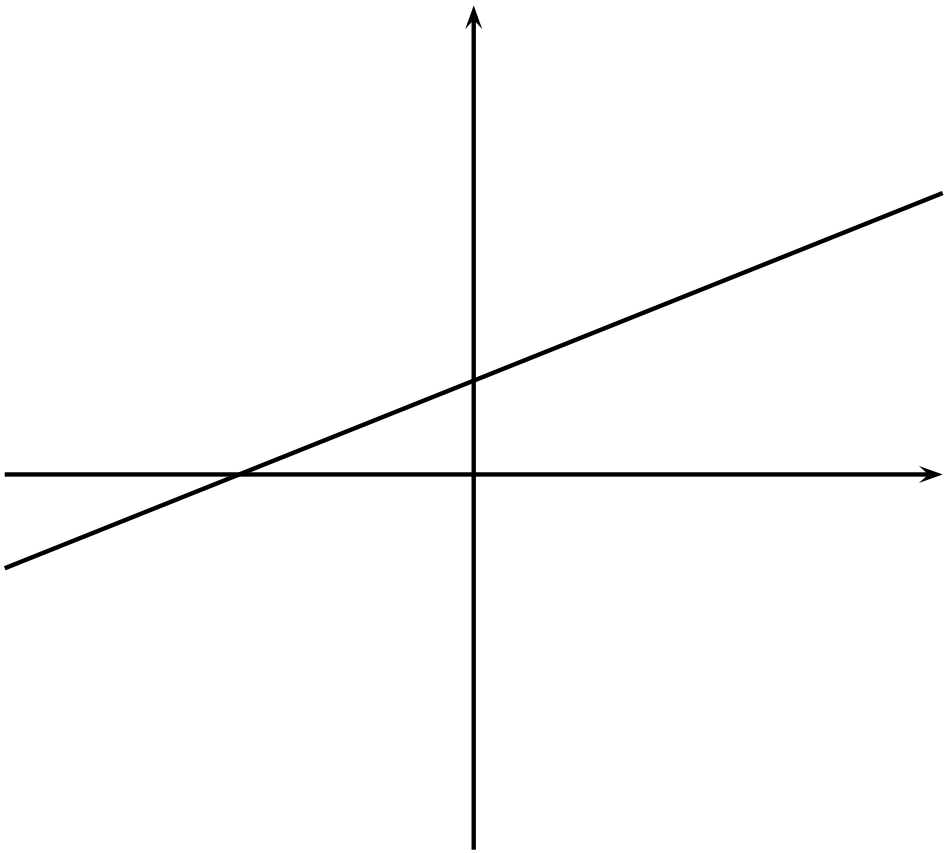
α é a pontuação cosseno ω é o tamanho da janela

Isso é a mesma metodologia para pontuação de campos exceto que agora temos características mais complexas que capturam se um doc é relevante para uma consulta

Representação gráfica do conjunto de treino

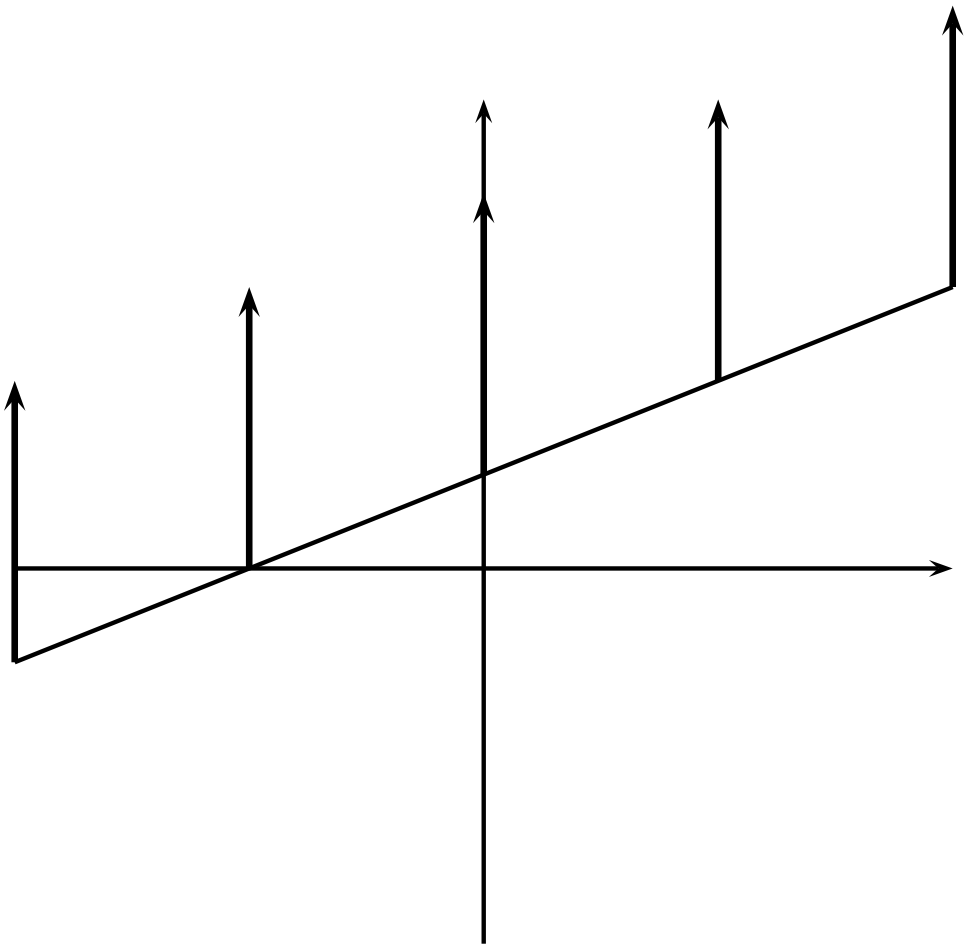


Neste caso: abordagem de Aprendizado de Ranking aprende um classificador linear em 2D



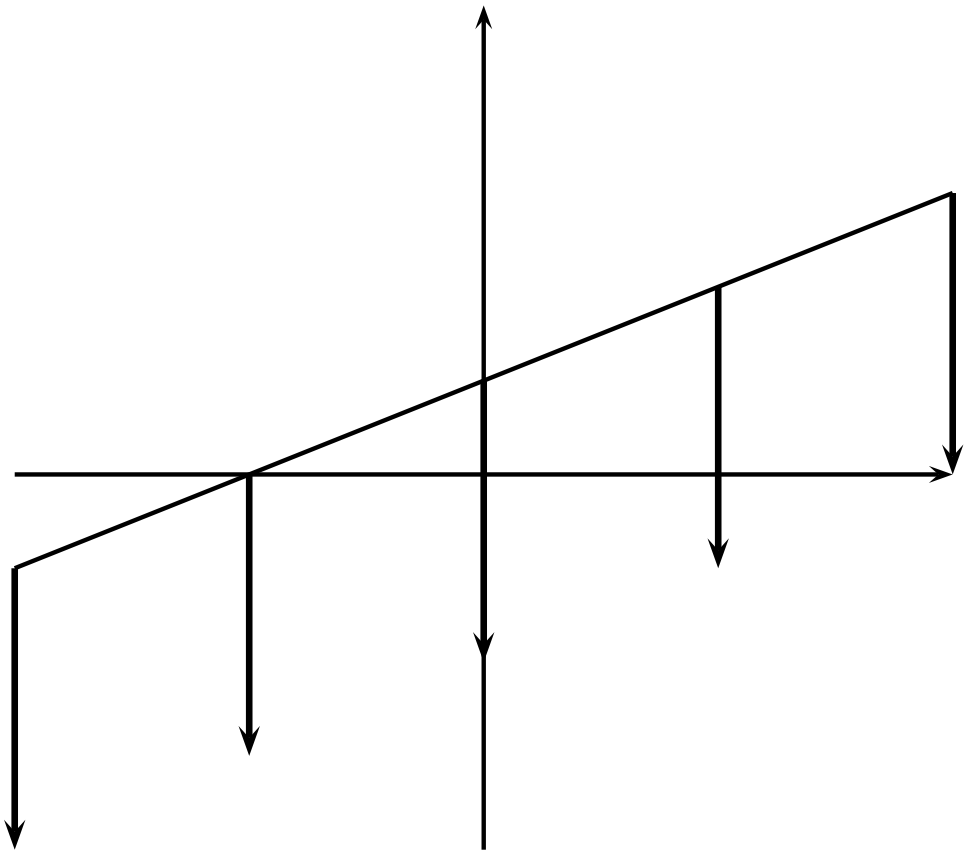
- Um classificador linear em 2D é uma linha descrita pela equação $w_1 d_1 + w_2 d_2 = \theta$
- Exemplo para um classificador linear 2D
- Pontos $(d_1\ d_2)$ com $w_1 d_1 + w_2 d_2 \geq \theta$ na classe c.
- Pontos $(d_1\ d_2)$ com $w_1 d_1 + w_2 d_2 < \theta$ estão na classe complementar $\overline{c}$.

Neste caso: abordagem de Aprendizado de Ranking aprende um classificador linear em 2D



- Um classificador linear em 2D é uma linha descrita pela equação $w_1 d_1 + w_2 d_2 = \theta$
- Exemplo para um classificador linear 2D
- Pontos $(d_1\ d_2)$ com $w_1 d_1 + w_2 d_2 \geq \theta$ na classe c.
- Pontos $(d_1\ d_2)$ com $w_1 d_1 + w_2 d_2 < \theta$ estão na classe complementar $\overline{c}$.

Neste caso: abordagem de Aprendizado de Ranking aprende um classificador linear em 2D



- Um classificador linear em 2D é uma linha descrita pela equação $w_1 d_1 + w_2 d_2 = \theta$
- Exemplo para um classificador linear 2D
- Pontos $(d_1\ d_2)$ com $w_1 d_1 + w_2 d_2 \geq \theta$ na classe c.
- Pontos $(d_1\ d_2)$ com $w_1 d_1 + w_2 d_2 < \theta$ estão na classe complementar $\overline{c}$.

Metodologia de Aprendizado de Ranking para duas características

- Novamente, 2 classes: relevante $= 1$ e não-relevante $= 0$

Metodologia de Aprendizado de Ranking para duas características

- Novamente, 2 classes: relevante $= 1$ e não-relevante $= 0$
- Buscamos uma função de pontuação que combina os valores das características para gera um valor que é (perto de) 0 ou 1

Metodologia de Aprendizado de Ranking para duas características

- Novamente, 2 classes: relevante $= 1$ e não-relevante $= 0$
- Buscamos uma função de pontuação que combina os valores das características para gera um valor que é (perto de) 0 ou 1
- Queremos que essa função esteja em acordo com o conj. de treino o máximo possível

Metodologia de Aprendizado de Ranking para duas características

- Novamente, 2 classes: relevante $= 1$ e não-relevante $= 0$
- Buscamos uma função de pontuação que combina os valores das características para gera um valor que é (perto de) 0 ou 1
- Queremos que essa função esteja em acordo com o conj. de treino o máximo possível
- Um classificador linear é definido por uma equação da forma:

$$pontuacao(d, q) = pontuacao(\alpha, \omega) = a\alpha + b\omega + c,$$

onde aprendemos os coeficientes a, b, c a partir dos exemplos de treino

Metodologia de Aprendizado de Ranking para duas características

- Novamente, 2 classes: relevante = 1 e não-relevante = 0
- Buscamos uma função de pontuação que combina os valores das características para gera um valor que é (perto de) 0 ou 1
- Queremos que essa função esteja em acordo com o conj. de treino o máximo possível
- Um classificador linear é definido por uma equação da forma:

$$pontuacao(d, q) = pontuacao(\alpha, \omega) = a\alpha + b\omega + c,$$

onde aprendemos os coeficientes a, b, c a partir dos exemplos de treino

- Regressão vs. classificação

Metodologia de Aprendizado de Ranking para duas características

- Novamente, 2 classes: relevante = 1 e não-relevante = 0
- Buscamos uma função de pontuação que combina os valores das características para gera um valor que é (perto de) 0 ou 1
- Queremos que essa função esteja em acordo com o conj. de treino o máximo possível
- Um classificador linear é definido por uma equação da forma:

$$pontuacao(d, q) = pontuacao(\alpha, \omega) = a\alpha + b\omega + c,$$

onde aprendemos os coeficientes a, b, c a partir dos exemplos de treino

- Regressão vs. classificação
  - Vimos até agora apenas classificação

Metodologia de Aprendizado de Ranking para duas características

- Novamente, 2 classes: relevante = 1 e não-relevante = 0
- Buscamos uma função de pontuação que combina os valores das características para gera um valor que é (perto de) 0 ou 1
- Queremos que essa função esteja em acordo com o conj. de treino o máximo possível
- Um classificador linear é definido por uma equação da forma:

$$pontuacao(d, q) = pontuacao(\alpha, \omega) = a\alpha + b\omega + c,$$

onde aprendemos os coeficientes a, b, c a partir dos exemplos de treino

- Regressão vs. classificação
  - Vimos até agora apenas classificação
  - Também podemos tratar o problema como uma questão de regressão

Metodologia de Aprendizado de Ranking para duas características

- Novamente, 2 classes: relevante = 1 e não-relevante = 0
- Buscamos uma função de pontuação que combina os valores das características para gera um valor que é (perto de) 0 ou 1
- Queremos que essa função esteja em acordo com o conj. de treino o máximo possível
- Um classificador linear é definido por uma equação da forma:

$$pontuacao(d, q) = pontuacao(\alpha, \omega) = a\alpha + b\omega + c,$$

onde aprendemos os coeficientes a, b, c a partir dos exemplos de treino

- Regressão vs. classificação
  - Vimos até agora apenas classificação
  - Também podemos tratar o problema como uma questão de regressão
  - Foi isso que fizemos para a pontuação de campos

Interpretação geométrica

Interpretação geométrica

Interpretação geométrica

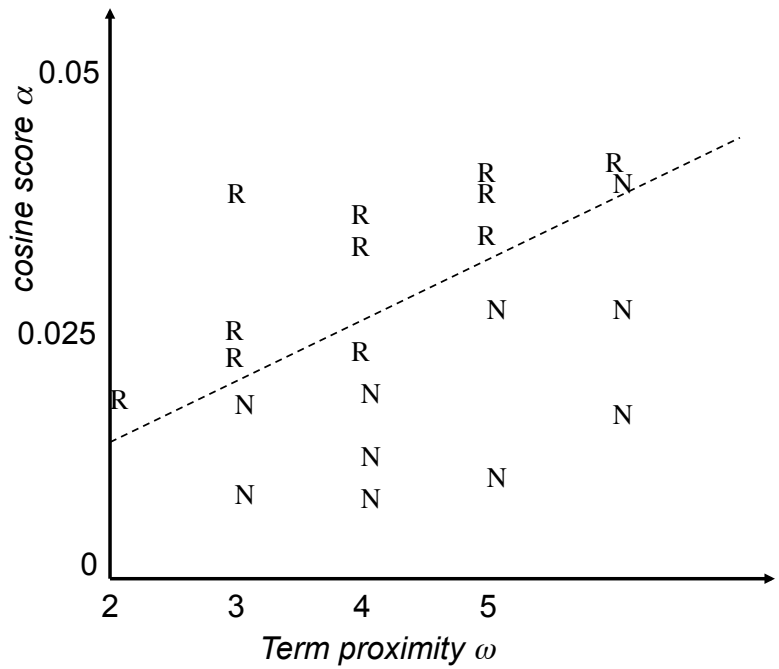
- A função *pontuacao*(α, ω) representa um plano "acima" da figura

Interpretação geométrica

- A função *pontuacao*(α, ω) representa um plano "acima" da figura
- Idealmente esse plano assume valores próximos a 1 acima dos pontos marcados R e valores próximos a 0 acima dos pontos marcados N

Interpretação geométrica

- A função *pontuacao*(α, ω) representa um plano "acima" da figura
- Idealmente esse plano assume valores próximos a 1 acima dos pontos marcados R e valores próximos a 0 acima dos pontos marcados N



Classificação linear neste caso

Classificação linear neste caso

Classificação linear neste caso

- Escolhemos um valor de limiar θ.

Classificação linear neste caso

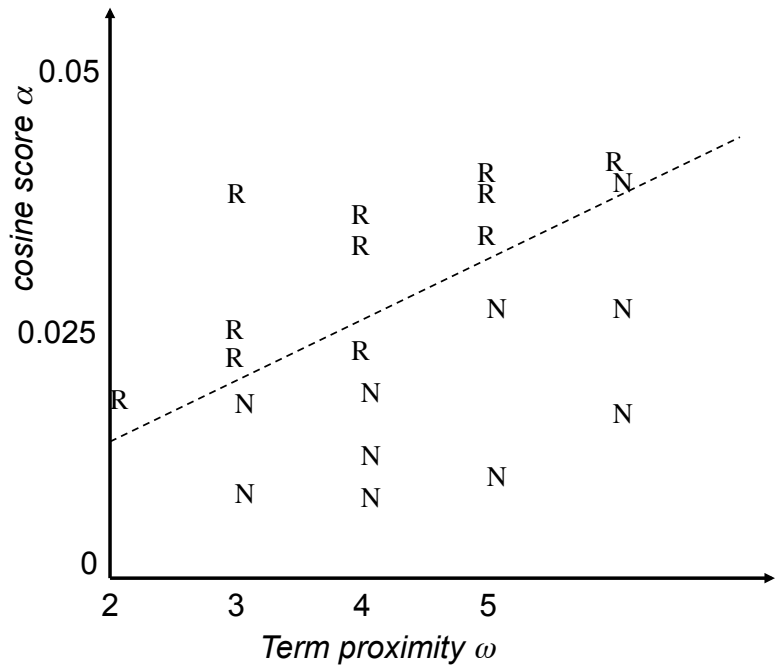
- Escolhemos um valor de limiar θ.
- Se $\mathit{pontuacao}(\alpha, \omega) > \theta$, declarmos o doc relevante, ou declaramos não-relevante.

Classificação linear neste caso

- Escolhemos um valor de limiar θ.
- Se $\mathit{pontuacao}(\alpha, \omega) > \theta$, declarmos o doc relevante, ou declaramos não-relevante.
- Todos os pontos que satisfazem $\mathit{pontuacao}(\alpha, \omega) = \theta$ formam uma linha (tracejada) $\rightarrow$ classificador linear que separa os relevantes dos não-relevantes

Classificação linear neste caso

- Escolhemos um valor de limiar θ.
- Se $pontuacao(\alpha, \omega) > \theta$, declarmos o doc relevante, ou declaramos não-relevante.
- Todos os pontos que satisfazem $pontuacao(\alpha, \omega) = \theta$ formam uma linha (tracejada) $\rightarrow$ classificador linear que separa os relevantes dos não-relevantes



Resumo

- O problema de fazer um julgamento binário relevante/não-relevante é transformado em problema de classificação ou regressão, baseado um conjunto de treino de pares de consultas-documentos e julgamentos de relevância.

Resumo

- O problema de fazer um julgamento binário relevante/não-relevante é transformado em problema de classificação ou regressão, baseado um conjunto de treino de pares de consultas-documentos e julgamentos de relevância.
- No exemplo: o classificador corresponde a uma linha $\mathit{pontuacao}(\alpha, \omega) = \theta$ no plano α-ω

Resumo

- O problema de fazer um julgamento binário relevante/não-relevante é transformado em problema de classificação ou regressão, baseado um conjunto de treino de pares de consultas-documentos e julgamentos de relevância.
- No exemplo: o classificador corresponde a uma linha $\mathit{pontuacao}(\alpha, \omega) = \theta$ no plano α-ω
- Em princípio, qualquer método de aprendizado que obtém um classificador linear (incluindo regressão de mínimos quadrados) pode ser usado para obter essa linha

Resumo

- O problema de fazer um julgamento binário relevante/não-relevante é transformado em problema de classificação ou regressão, baseado um conjunto de treino de pares de consultas-documentos e julgamentos de relevância.
- No exemplo: o classificador corresponde a uma linha $\mathit{pontuacao}(\alpha, \omega) = \theta$ no plano α-ω
- Em princípio, qualquer método de aprendizado que obtém um classificador linear (incluindo regressão de mínimos quadrados) pode ser usado para obter essa linha
- Vantagem de aprendizado de ranking: podemos evitar funções de pontuação sintonizadas a mão e simplesmente aprender a partir dos exemplos de treino

Resumo

- O problema de fazer um julgamento binário relevante/não-relevante é transformado em problema de classificação ou regressão, baseado um conjunto de treino de pares de consultas-documentos e julgamentos de relevância.
- No exemplo: o classificador corresponde a uma linha $\mathit{pontuacao}(\alpha, \omega) = \theta$ no plano α-ω
- Em princípio, qualquer método de aprendizado que obtém um classificador linear (incluindo regressão de mínimos quadrados) pode ser usado para obter essa linha
- Vantagem de aprendizado de ranking: podemos evitar funções de pontuação sintonizadas a mão e simplesmente aprender a partir dos exemplos de treino
- Custo: manter um conjunto de exemplos de treino em que julgamentos de relevância devem ser feitos por humanos

Aprendizado de ranking para mais de duas características

- A metodologia apresentada pode ser generalizada para um grande número de características

Aprendizado de ranking para mais de duas características

- A metodologia apresentada pode ser generalizada para um grande número de características
- Em adição à similaridade cosseno e à janela de termos, há outros indicadores de relevância: medidas similares ao PageRank, idade do documento, contribuições de campos, comprimento do documento etc.

Aprendizado de ranking para mais de duas características

- A metodologia apresentada pode ser generalizada para um grande número de características
- Em adição à similaridade cosseno e à janela de termos, há outros indicadores de relevância: medidas similares ao PageRank, idade do documento, contribuições de campos, comprimento do documento etc.
- Se essas medidas podem ser calculadas para um conj. de documentos de treino com julgamentos de relevância, então podem ser usadas para o aprendizado de ranking

Características usadas pela Microsoft Research

- Campos: corpo, âncora, título, url, doc completo

Características usadas pela Microsoft Research

- Campos: corpo, âncora, título, url, doc completo
- Características derivadas de modelos padrões de recuperação de informação: número de termos da consulta, taxa de termos da consulta, idf, soma da frequência de termos, mínimo da frequência de termos, máximos da frequência de termos, média da frequência de termos, variância de frequência de termos, soma da frequência de termos normalizada, ..., soma de tf-idf, modelo booleano, BM25 etc.

Características usadas pela Microsoft Research

- Campos: corpo, âncora, título, url, doc completo
- Características derivadas de modelos padrões de recuperação de informação: número de termos da consulta, taxa de termos da consulta, idf, soma da frequência de termos, mínimo da frequência de termos, máximos da frequência de termos, média da frequência de termos, variância de frequência de termos, soma da frequência de termos normalizada, ..., soma de tf-idf, modelo booleano, BM25 etc.
- Características específicas para web: número de barras no url, tamanho do url, número de inlinks, número de outlinks, PageRank, SiteRank

Características usadas pela Microsoft Research

- Campos: corpo, âncora, título, url, doc completo
- Características derivadas de modelos padrões de recuperação de informação: número de termos da consulta, taxa de termos da consulta, idf, soma da frequência de termos, mínimo da frequência de termos, máximos da frequência de termos, média da frequência de termos, variância de frequência de termos, soma da frequência de termos normalizada, ..., soma de tf-idf, modelo booleano, BM25 etc.
- Características específicas para web: número de barras no url, tamanho do url, número de inlinks, número de outlinks, PageRank, SiteRank
- Características baseadas em histórico: contagem de cliques consulta-url, contagem de cliques da url, tempo na url

Características usadas pela Microsoft Research

- Campos: corpo, âncora, título, url, doc completo
- Características derivadas de modelos padrões de recuperação de informação: número de termos da consulta, taxa de termos da consulta, idf, soma da frequência de termos, mínimo da frequência de termos, máximos da frequência de termos, média da frequência de termos, variância de frequência de termos, soma da frequência de termos normalizada, ..., soma de tf-idf, modelo booleano, BM25 etc.
- Características específicas para web: número de barras no url, tamanho do url, número de inlinks, número de outlinks, PageRank, SiteRank
- Características baseadas em histórico: contagem de cliques consulta-url, contagem de cliques da url, tempo na url
- Ver: `http://research.microsoft.com/en-us/projects/mslr/`

Avaliação de aprendizado para ranking

- A ideia de aprendizado de ranking é antiga

Avaliação de aprendizado para ranking

- A ideia de aprendizado de ranking é antiga
  - Trabalho pioneiro foi feito por Norbert Fuhr e William S. Cooper

Avaliação de aprendizado para ranking

- A ideia de aprendizado de ranking é antiga
  - Trabalho pioneiro foi feito por Norbert Fuhr e William S. Cooper
- Mas só recentemente que o conhecimento em Aprendizado de Máquina e poder computacional suficientes foram obtidos para obter excelentes resultados práticos

Avaliação de aprendizado para ranking

- A ideia de aprendizado de ranking é antiga
  - Trabalho pioneiro foi feito por Norbert Fuhr e William S. Cooper
- Mas só recentemente que o conhecimento em Aprendizado de Máquina e poder computacional suficientes foram obtidos para obter excelentes resultados práticos
- Enquanto que especialistas podem fazer um bom trabalho montando função de ranking manualmente, ajustar a mão é difícil e trabalhoso e deve ser refeito com frequência para novos docs e tipos de usuários

Avaliação de aprendizado para ranking

- A ideia de aprendizado de ranking é antiga
  - Trabalho pioneiro foi feito por Norbert Fuhr e William S. Cooper
- Mas só recentemente que o conhecimento em Aprendizado de Máquina e poder computacional suficientes foram obtidos para obter excelentes resultados práticos
- Enquanto que especialistas podem fazer um bom trabalho montando função de ranking manualmente, ajustar a mão é difícil e trabalhoso e deve ser refeito com frequência para novos docs e tipos de usuários
- Quanto mais características são usadas em ranking, mais difícil é integrá-las manualmente em uma função de ranking

Avaliação de aprendizado para ranking

- A ideia de aprendizado de ranking é antiga
  - Trabalho pioneiro foi feito por Norbert Fuhr e William S. Cooper
- Mas só recentemente que o conhecimento em Aprendizado de Máquina e poder computacional suficientes foram obtidos para obter excelentes resultados práticos
- Enquanto que especialistas podem fazer um bom trabalho montando função de ranking manualmente, ajustar a mão é difícil e trabalhoso e deve ser refeito com frequência para novos docs e tipos de usuários
- Quanto mais características são usadas em ranking, mais difícil é integrá-las manualmente em uma função de ranking
- Buscadores usam um grande número de características $\rightarrow$ buscadores necessitam de Aprendizado de Máquina aplicado a Ranking