

# Learning to Rank for Faceted Search

## Bridging the gap between theory and practice

Agnes van Belle @ Berlin Buzzwords 2017

powered by

textkernel

# Job-to-person search system

Recent job titles ▾

Job group ▾

Job class ▾

City ▲

Postal/ZIP code or city

 +25 miles ▾

US Arizona City AZ @ 25 miles

Nice to have    Must have

Employers ▾

Years of experience ▾

IT skills ▾

Language skills ▾

Education level ▲

- Master (3)
- Post-Master
- Secondary Education (9)

City   Years of experience   Job class

Recent job titles   Job group

Projects

**t** (37)

Actions ▾   37 results

<input type="checkbox"/>	<b>Ricky</b> / <b>Customer Service Representative</b> / Customer Service Representative	☆ <input type="button" value="📊"/>
<input type="checkbox"/>	<b>Jeffrey</b> / Technical Support Coordinator, <b>Customer Service Representative</b> , Customer Service R	☆ <input type="button" value="📊"/>
<input type="checkbox"/>	<b>Christina</b> / Operations Manager Environmental Services, Director of Food Service / Assistant Food	☆ <input type="button" value="📊"/>
<input type="checkbox"/>	<b>Jerry</b> / Order Picker, Customer Service, Warehouseman/	☆ <input type="button" value="📊"/>
<input type="checkbox"/>	<b>Joe</b> / Well Tester/Frac Support, Stinger Welding / Well Te	☆ <input type="button" value="📊"/>
<input type="checkbox"/>	<b>Sylvia</b> / Supervisor, Dispatcher / Supervisor	☆ <input type="button" value="📊"/>

City: US Arizona City AZ @ 25 miles ✓

Job class: Hospitality ✓

Job group: Customer Service Personnel ✗

Recent job titles: Customer service representative ✗

Years of experience ✗

# Generated query

All Positions [Technical Account Manager] x Profession Account Manager (Technical Products) x

Job Category Sales and Trading x Job Type Articles and Products Representatives x City US Portland OR ⊕ 75 miles x

IT Skills SQL +8 x & WSDL +13 x & web service x & Excel +10 x & Troubleshooting x & CRM x & ITIL +3 x & OEM +57 x

Years of Experience 3 to 5 years x / 6 to 10 years x

Full text client business x & sales +14 x & pricing +5 x & scheduling +10 x & Customer Service +13 x & Account Management +7 x & SoapUI +34 x & web service calls x

Education Master's x / Doctoral or Phd x Languages name: English x & name: Spanish x

Last Employer Bank of America x

# Match indicator

/ AirWatch by VMware / US Atlanta GA	☆	📶	▼
	Education: Master's, Doctoral or Phd	✓	
	IT Skills: SQL	✓	
	IT Skills: Excel	✓	
/ Fiserv	IT Skills: CRM	✓	▼
	IT Skills: OEM	✓	
	Job Category: Sales and Trading	✓	
	Languages: name: English	✓	
or Associate	Languages: name: Spanish	✓	▼
	Years of Experience	✓	
	sales	✓	
	scheduling	✓	
artment of A	Customer Service	✓	
	Account Management	✓	▼
	SoapUI	✓	
	All Positions: [Technical Account Manager]	✗	
	City: US Portland OR ⊕ 75 miles	✗	
it Consultan	IT Skills: WSDL	✗	▼
	IT Skills: web service	✗	
	IT Skills: Troubleshooting	✗	
	IT Skills: ITIL	✗	
Specialist /	Job Type: Articles and Products Representatives	✗	▼
	Last Employer: Bank of America	✗	
	Profession: Account Manager (Technical Products)	✗	
	client business	✗	
	pricing	✗	
er / SunGar	web service calls	✗	▼

# Faceted search

- Multiple explicit and independent dimensions, called facets
- Lets users refine search by choosing values
- No candidate is ideal: many should-have clauses

# Scoring of search results

- Term-frequency based metric  
e.g. **BM25, TF-IDF**

- Facet weights

$$\begin{aligned} \text{TF-IDF}(\text{jobtitle}) &\cdot 1.5 && + \\ \text{TF-IDF}(\text{skill}) &\cdot 2.0 && + \\ \text{TF-IDF}(\text{location}) &\cdot 0.7 && + \\ \text{TF-IDF}(\text{languages}) &\cdot 0.25 && = \text{score} \end{aligned}$$

# Tuning the system: objectives

- “If I search for a skill ‘Java’ I want the candidates that also have ‘Java’ in their Jobtitle field to be weighted higher”
- “Education will be a less important match, the more years of experience a candidate has”
- “We should weight location matches less when finding candidates in IT”

# Learning to rank

- Learn a parameterized ranking model
- That optimizes ranking order
- Re-learn for personalization or preference change



# Learning to rank by tuning facet weights

- Do exhaustive search for optimal weights to set
- Improved our retrieval by **6%** (NDCG metric)

$$\left[ \text{TF-IDF}(\text{jobtitle}), \text{TF-IDF}(\text{skill}) \dots \text{TF-IDF}(\text{language}) \right] \cdot \begin{bmatrix} 1.5 \\ 2.0 \\ \vdots \\ 0.25 \end{bmatrix} = \text{score}$$

# Tuning facet weights: limitations

- Cannot consider interdependency of facet field dimensions
- Cannot take into account the actual *content* of fields
  - only match indicators

# Learning objectives

- Take into account facet field content
- Model facet field interdependencies

# Learning to rank

- Machine Learning from user feedback
- Input: set of {query, lists of assessed documents}
  - Each document has a relevance indication from feedback

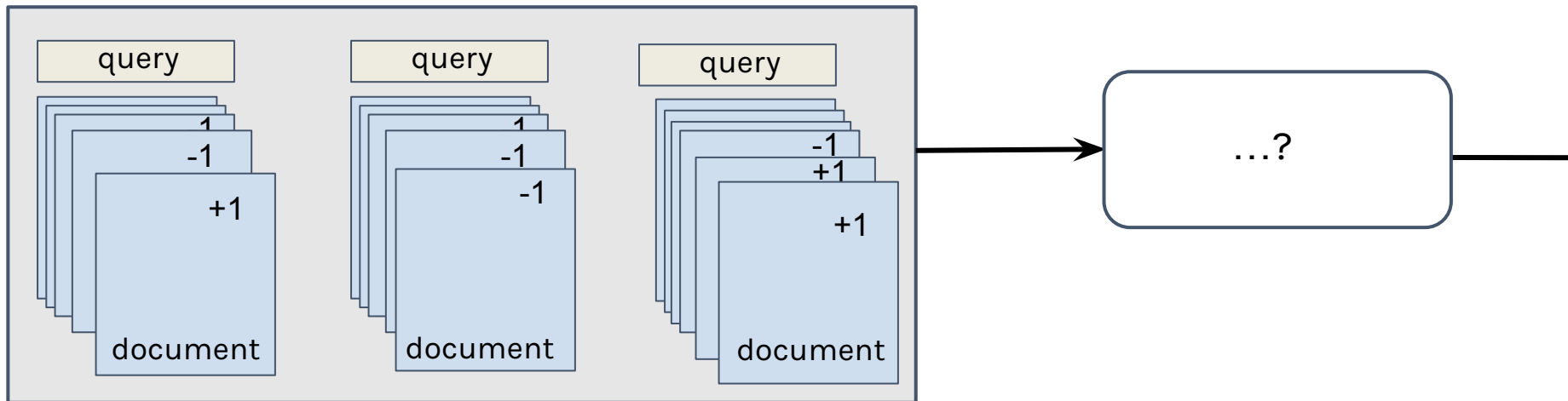
The screenshot shows a job search interface with the following filters: Employer: Cambridge Women's Resources Centre Cambridge, Jobtitle: teaching assistant, Full text: bristol, and Age: 1980 to 1984. Three job listings are visible:

- GARGI AKINMULERO** / Teaching Assistant / Bristol
- ELISA Davis** / Regional Operations Manager / COVENTRY
- Bhupesh Das** / Senior Accounts Clerk, Accounts Administrator / London

Each listing has a set of feedback icons: a speech bubble, a thumbs up, a star, and a bar chart. An orange callout box labeled "explicit feedback" points to the star icon for the first listing. Another orange callout box labeled "implicit feedback" points to the mouse cursor hovering over the checkbox for the second listing.

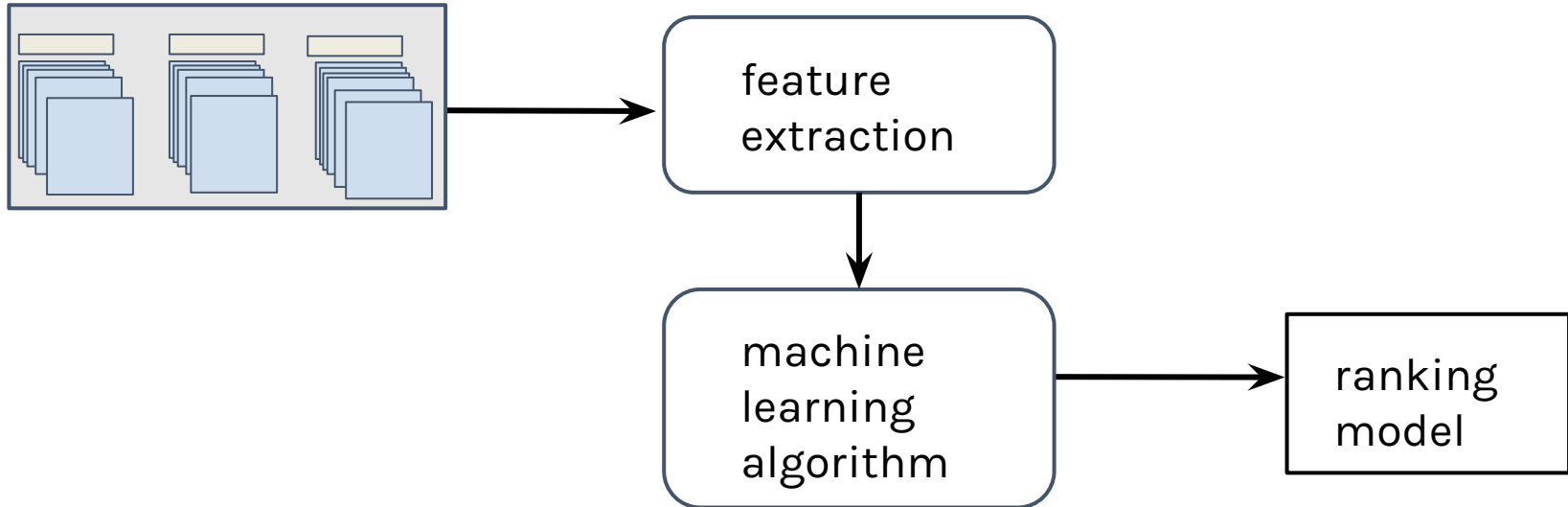
# Learning to rank

- Machine Learning from user feedback
- Input: set of {query, list of assessed documents}
  - Each document has a relevance label from feedback



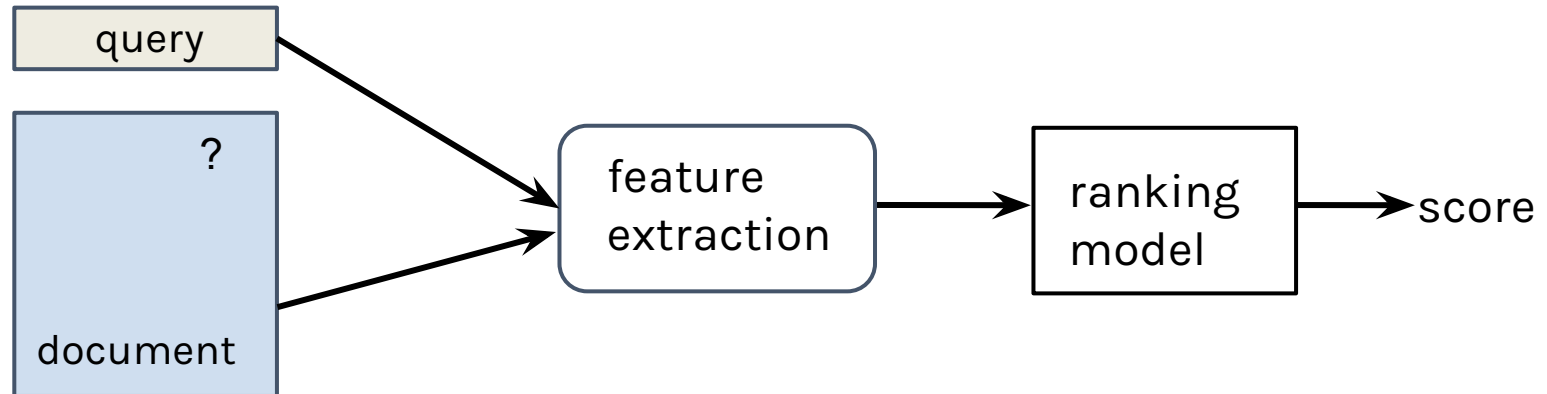
# Learning to rank

- Algorithm learns how to combine query & document content to optimize ordering considering relevance labels

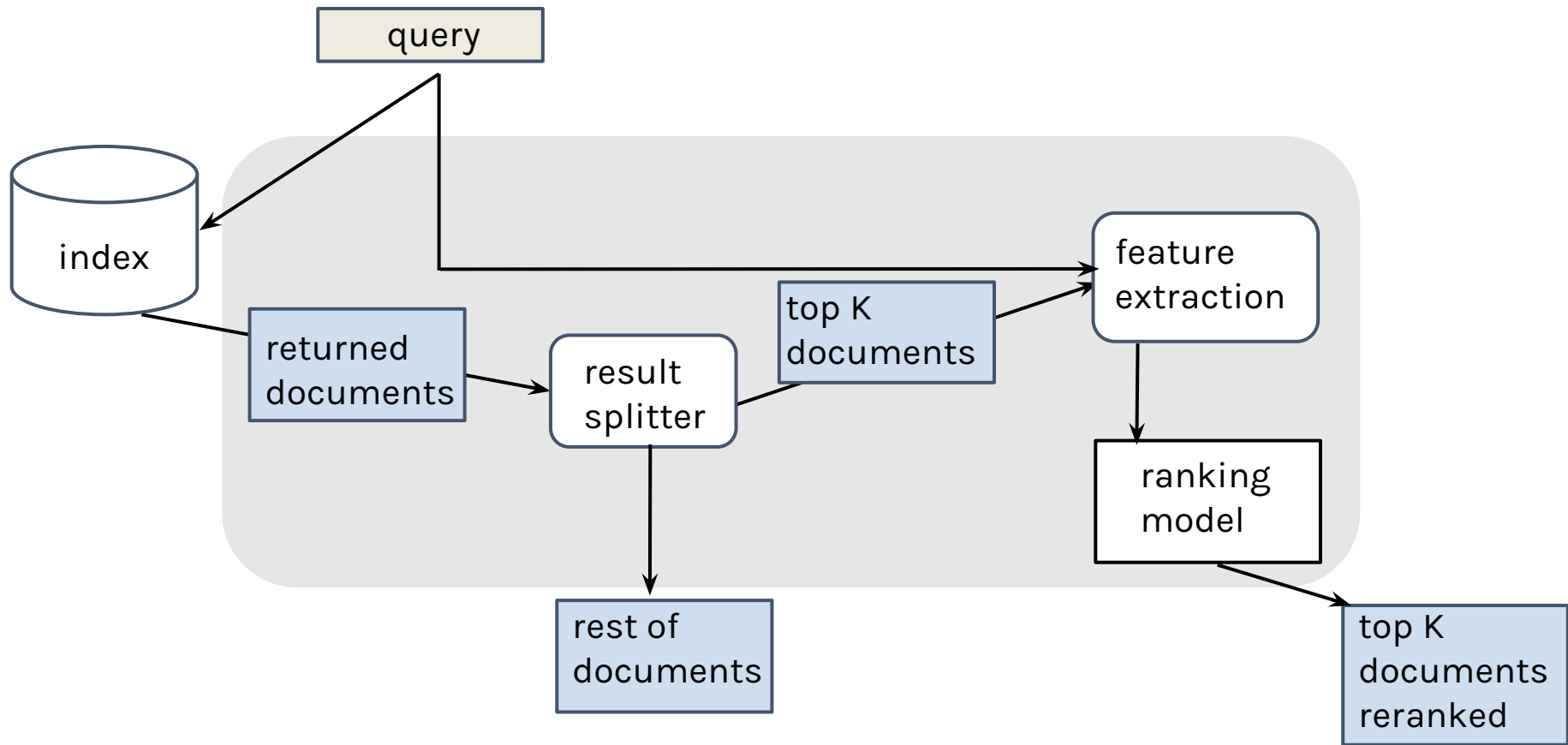


# Learning to rank

- Output: model that gives a relevance score given a query and document

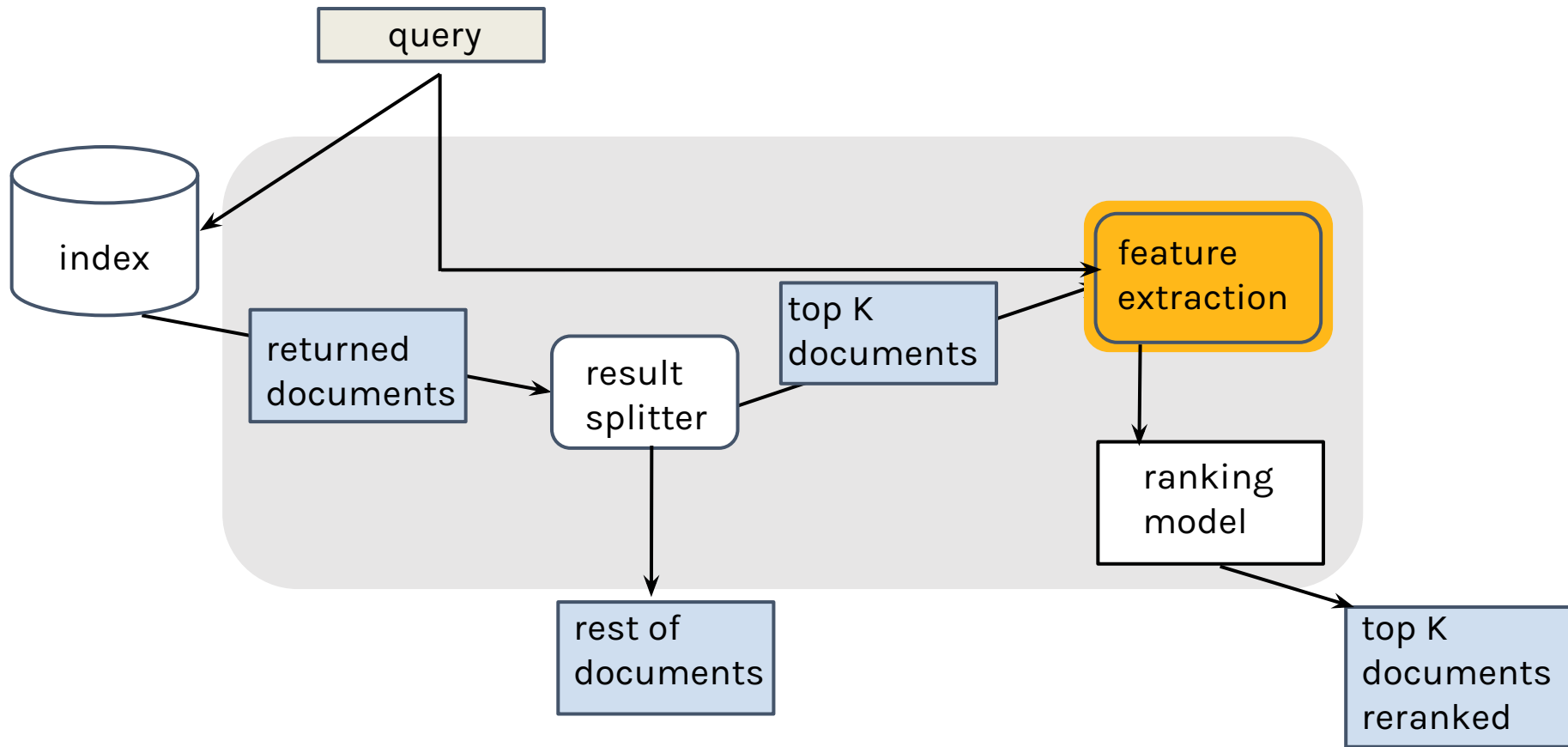


# Dynamic top K reranking





# Dynamic top K reranking



# Typical features

“In learning to rank, each query-document pair is represented by a multi-dimensional feature vector, and **each dimension of the vector is a feature indicating how relevant or important the document is with respect to the query.**”<sup>\*</sup>

Used in LTR papers: <sup>1, 2, 3</sup>

- TF-IDF, BM25, DFR, Language Model, cosine similarity, rank in other engines, etc.
- Match-indicator between *whole* query & *whole* document

\* "LETOR: A Benchmark Collection for Research on Learning to Rank for Information Retrieval", T. Qin, T. Liu, J. Xu, Jun and H. Li, 2010

1 "Optimizing Search Engines using Clickthrough Data", T. Joachims, 2003

2 "AdaRank: A Boosting Algorithm for Information Retrieval", J. Xu and H. Li, 2007

3 "Multileave Gradient Descent for Fast Online Learning to Rank", A. Schuth, H. Oosterhuis, S. Whiteson and M. de Rijke, 2016

# Bag of words

software engineer data mining java amsterdam english

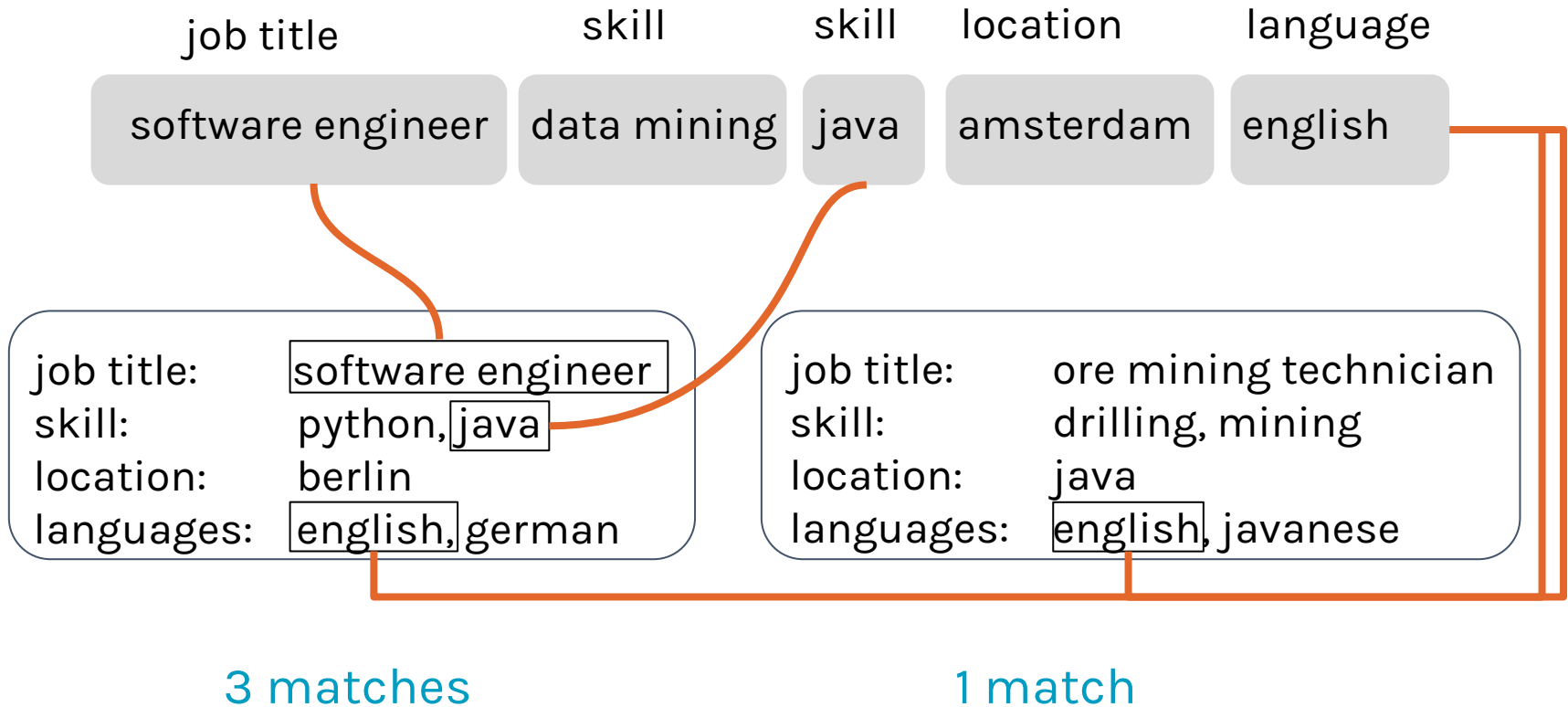
job title: software engineer  
skill: python, java  
location: berlin  
languages: english, german

4 matches

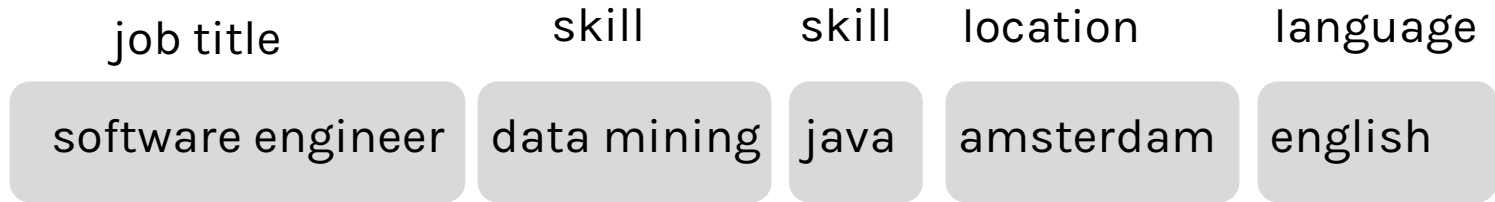
job title: ore mining technician  
skill: drilling, mining  
location: java  
languages: english, javanese

4 matches

# Split up in facet fields



# One feature per field

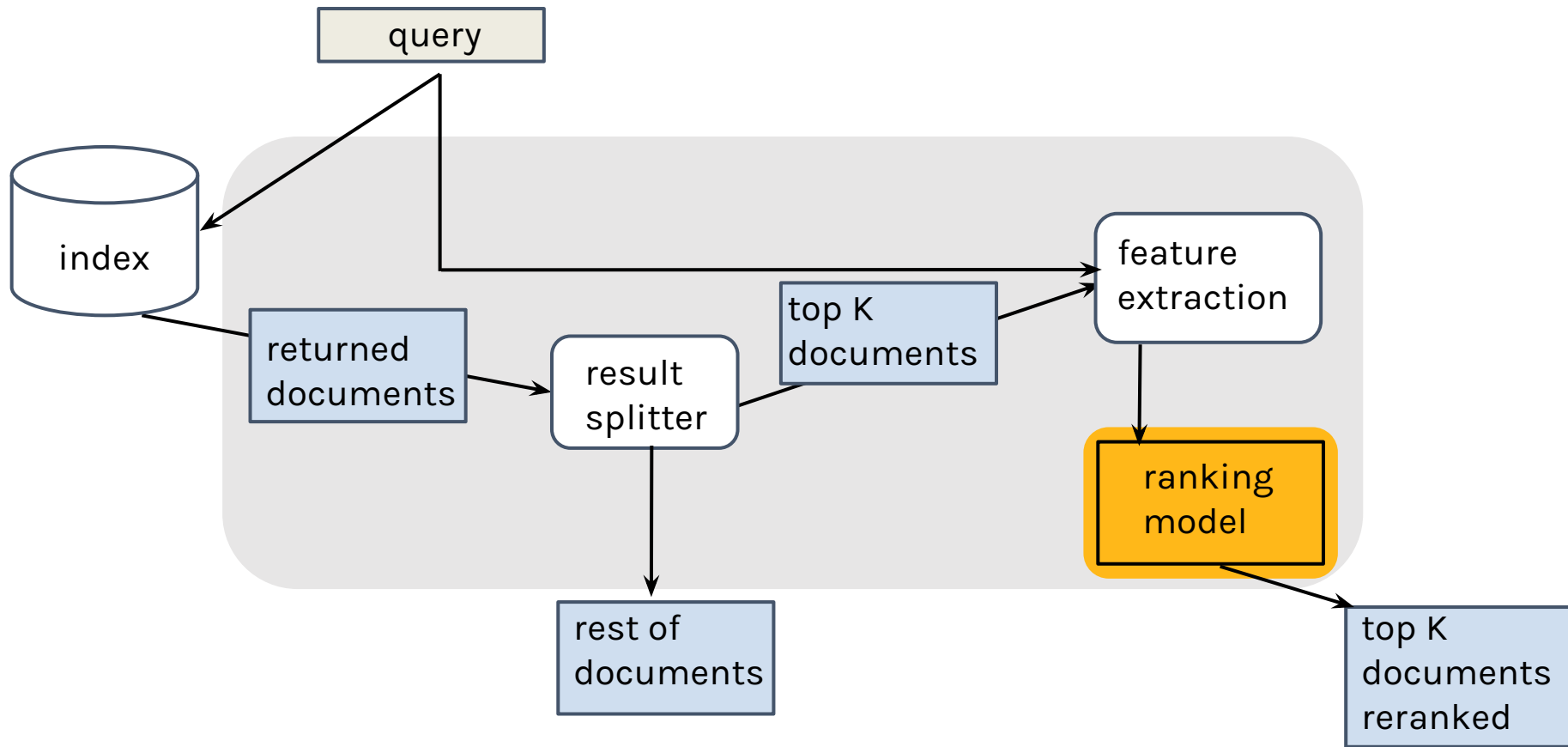


job title: software engineer  
skill: python, java  
location: berlin  
languages: english, german

feature vector

jobtitle	1/1
skill	1/2
location	0
language	1/1

# Dynamic top K reranking



# Linear models

- Used in many papers:
  - seminal papers<sup>1</sup>,
  - papers about leveraging user preferences<sup>2</sup>
  - papers about online learning / interleaving<sup>3</sup>
- Also in e.g. documentation about Solr's LTR contrib module

1 "Optimizing Search Engines using Clickthrough Data", T. Joachims, 2003

2 "A contextual-bandit approach to personalized news article recommendation", L. Li, W. Chu, J. Langford, and R. E. Schapire, 2010.

3 "Balancing exploration and exploitation in listwise and pairwise online learning to rank for information retrieval", K. Hofmann, S. Whiteson, M. de Rijke, 2013

# Linear models

End up with weight vector you can multiply with feature vectors.

$$[f_1 \quad f_2 \quad f_3 \quad \dots \quad f_n] \cdot \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix} = \text{score}$$



# Linear models

End up with weight vector you can multiply with feature vectors.

The diagram illustrates a linear model equation. On the left, there are four orange rounded rectangular boxes containing the labels: "jobtitle match", "skill match", "location match", and "language match". Arrows point from "jobtitle match" to the value 1.0, from "skill match" to 0.5, from "location match" to 0.0, and from "language match" to 1.0. The feature vector is shown as  $[1.0 \ 0.5 \ 0.0 \ \dots \ 1.0]$ . This is multiplied by a weight vector  $\begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix}$ , which is represented by a vertical column of values in square brackets. The result of the multiplication is an equals sign followed by the word "score".

$$\begin{bmatrix} 1.0 & 0.5 & 0.0 & \dots & 1.0 \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix} = \text{score}$$

# Tuning facet weights: limitations 🐱

- Cannot consider interdependency of facet field dimensions
- Cannot take into account the actual *content* of fields
  - only match indicators

The diagram illustrates the calculation of a score based on match indicators and weights. It features four orange rounded rectangular boxes: 'jobtitle match' and 'skill match' at the top, and 'location match' and 'language match' at the bottom. Arrows point from 'jobtitle match' to the value '1.0' in a vector, from 'skill match' to '0.5', from 'location match' to '0.0', and from 'language match' to '1.0'. The vector is  $[1.0 \ 0.5 \ 0.0 \ \dots \ 1.0]$ . This vector is multiplied by a column vector of weights  $\begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix}$ , resulting in the equation:  $[1.0 \ 0.5 \ 0.0 \ \dots \ 1.0] \cdot \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix} = \text{score}$ .

$$\begin{bmatrix} 1.0 & 0.5 & 0.0 & \dots & 1.0 \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix} = \text{score}$$

# Objectives

- “If I search for a skill ‘Java’ I want the candidates that also have ‘Java’ in their Jobtitle field to be weighted higher”
- “Education will be a less important match, the more years of experience a candidate has”
- “We should weight location matches less when finding candidates in IT”

# Learning objectives

- Take into account facet field content
- Model facet field interdependencies

# Take into account facet field content

jobclass:IT was in document

jobclass:Retail was not in document

the query asked for 5+ years of experience

[0.0 1.0 0.0 0.0 0.0 0.5 0.0 ...]

5 possible "jobclass" categories in document

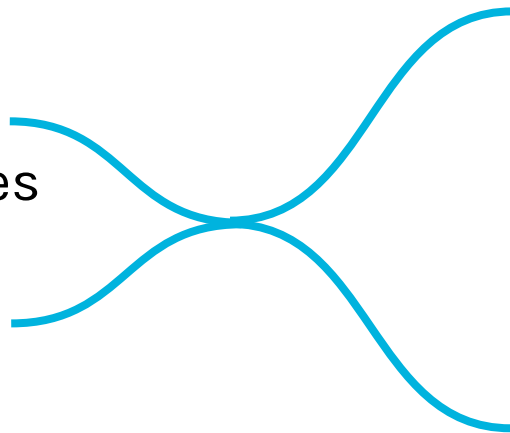
how many minimum years of experience in the query, normalized between 0 and 1

Categorical feature

Interval feature

# Take into account facet field content

- Query-document match features
- Document features
- Query features



Categorical: e.g. denoting job-class, skill etc.

Interval: e.g. years of experience

# Model facet field interdependencies

jobclass:IT was in document

jobclass:Retail was not in document

$$[0.0 \quad 1.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.5 \quad 0.0 \quad \dots] \cdot \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix} = \dots$$

# Model facet field interdependencies

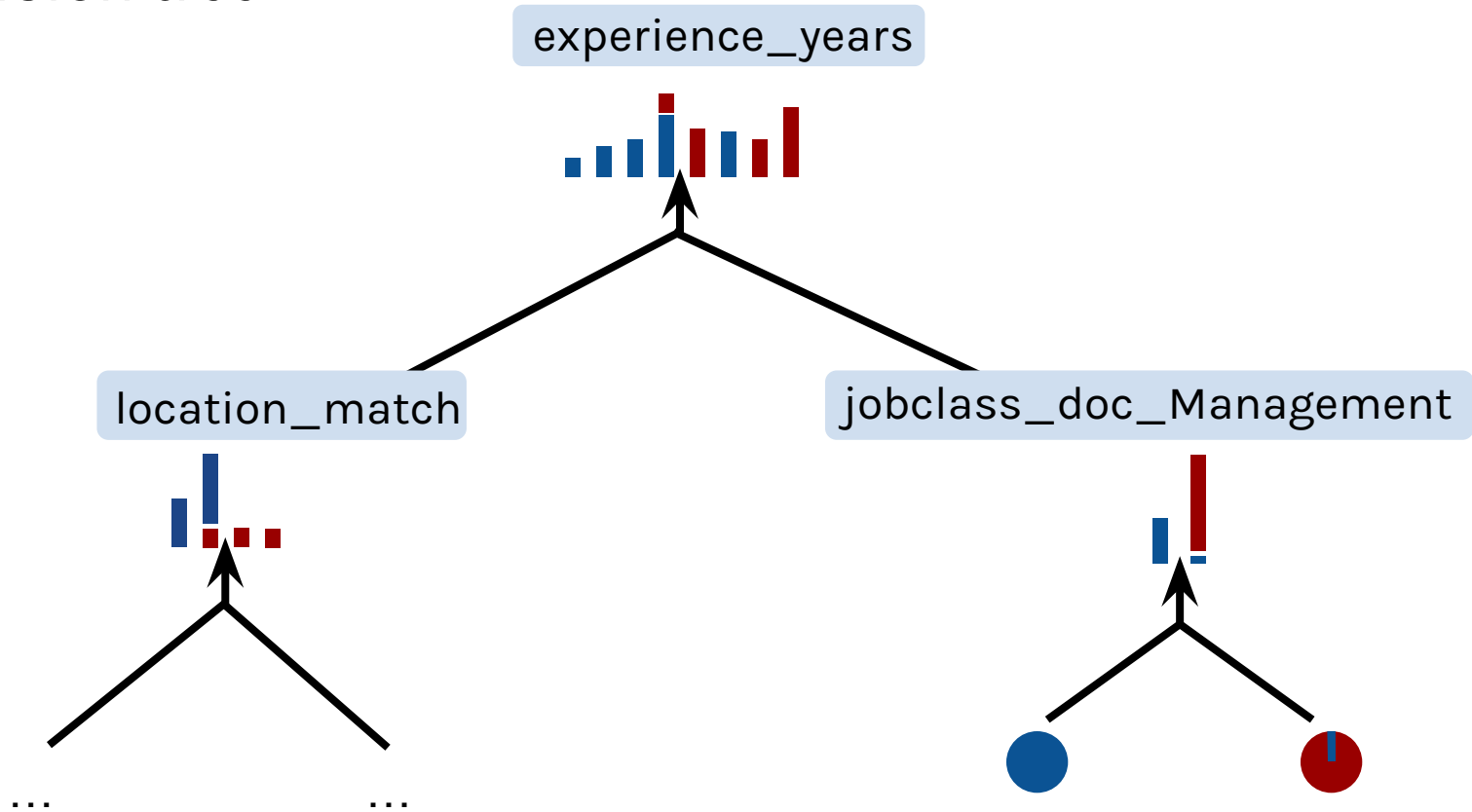
Use nonlinear ranking model based on e.g.

- Nonlinear neural networks
- Nonlinear SVM
- Decision trees



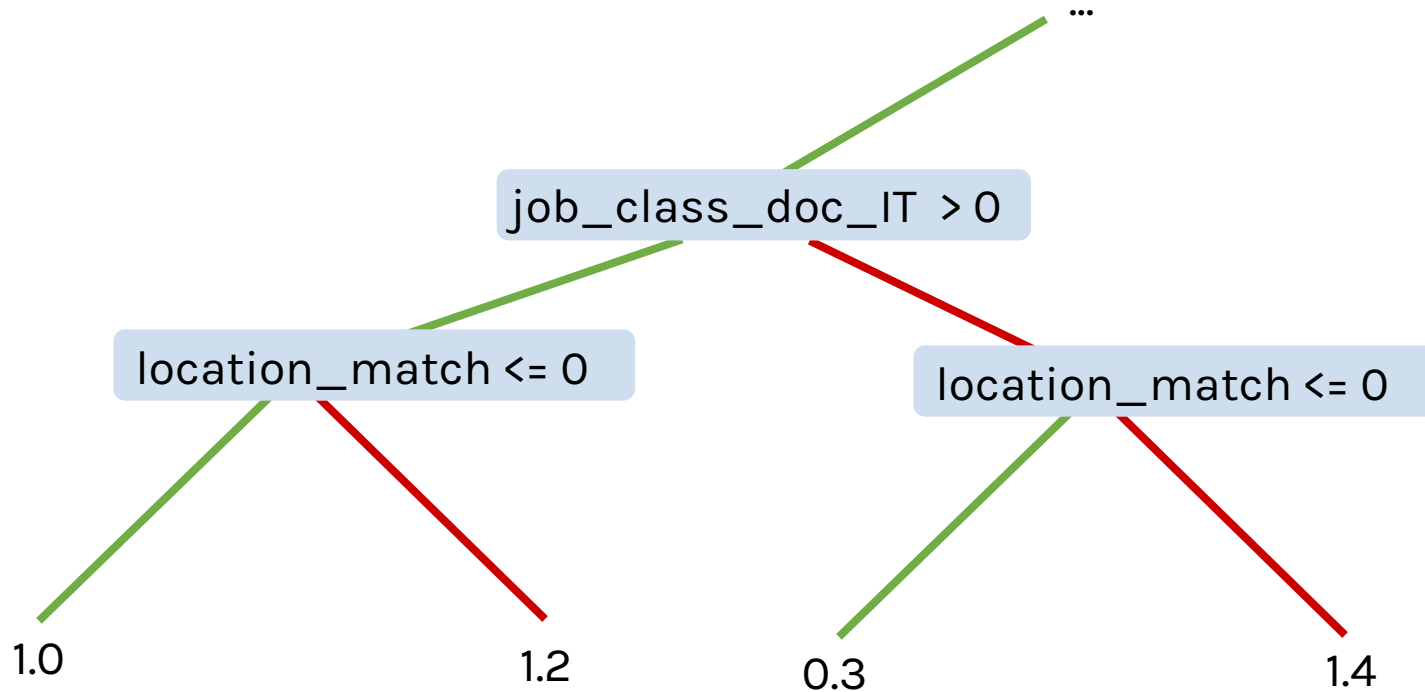
# Model facet field interdependencies

Decision tree



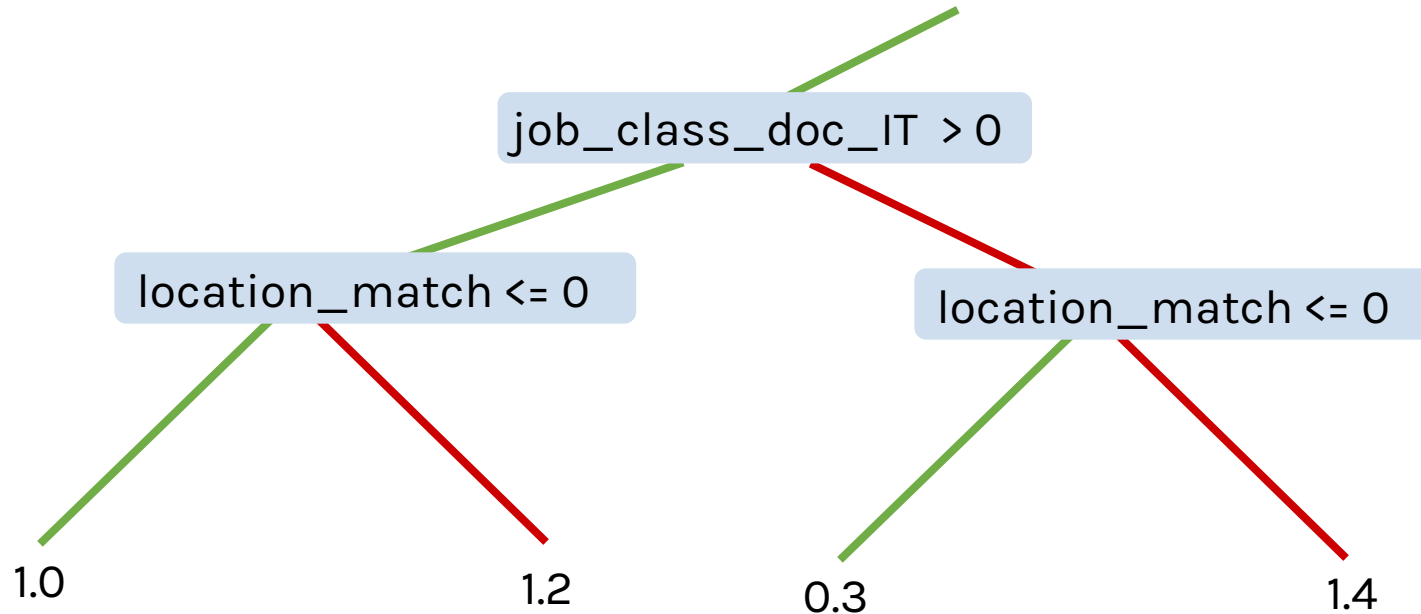
# Model facet field interdependencies

Decision tree



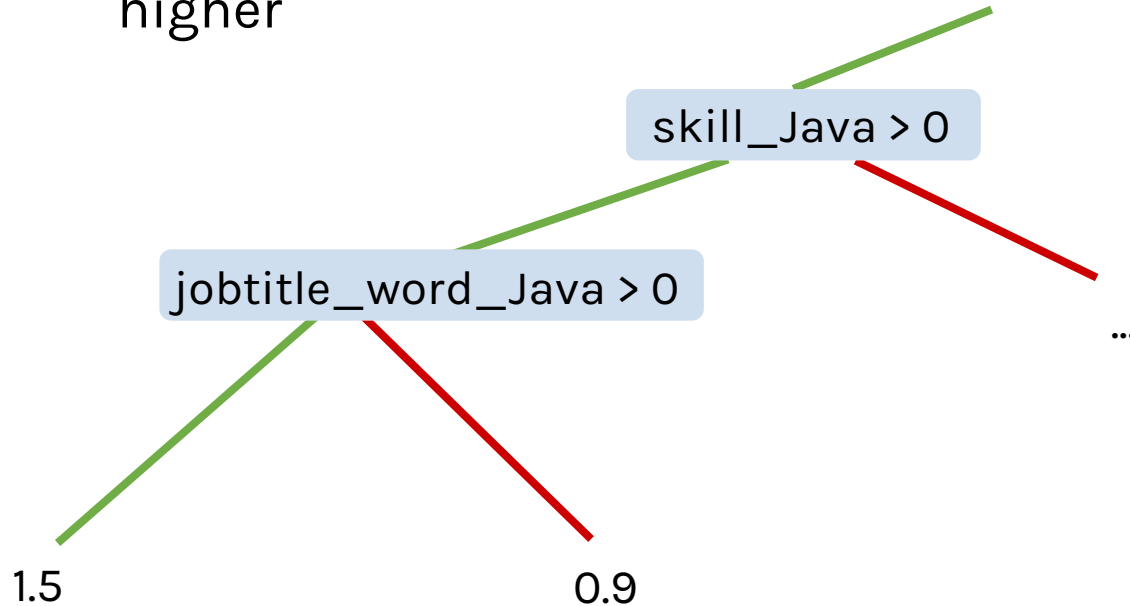
# Model facet field interdependencies

- “We should weight location matches less when finding candidates in IT”



# Model facet field interdependencies

- “If I search for a skill ‘Java’ I want the candidates that also have ‘Java’ in their Jobtitle field to be weighted higher”



# Model facet field interdependencies

- “If I search for a skill ‘Java’ I want the candidates that also have ‘Java’ in their Jobtitle field to be weighted higher”

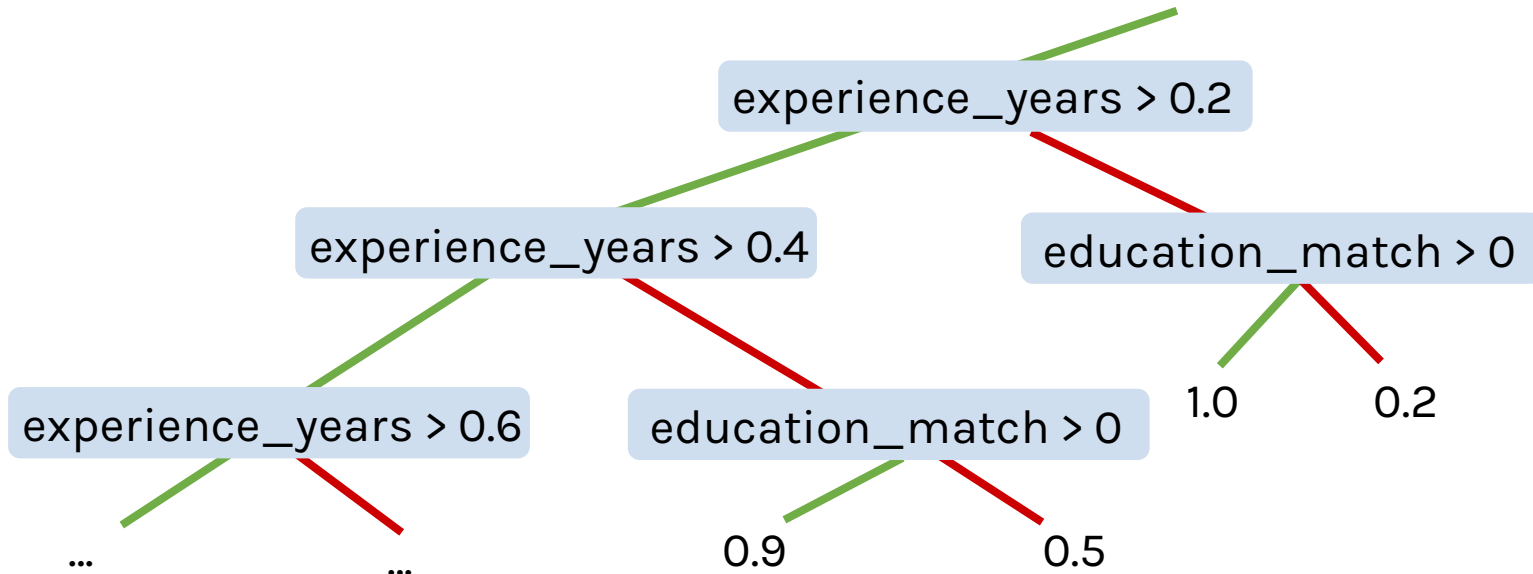
`jobtitle_contains_word_from_skill > 0`

1.4

0.8

# Model facet field interdependencies

- “Education will be a less important match, the more years of experience a candidate has”

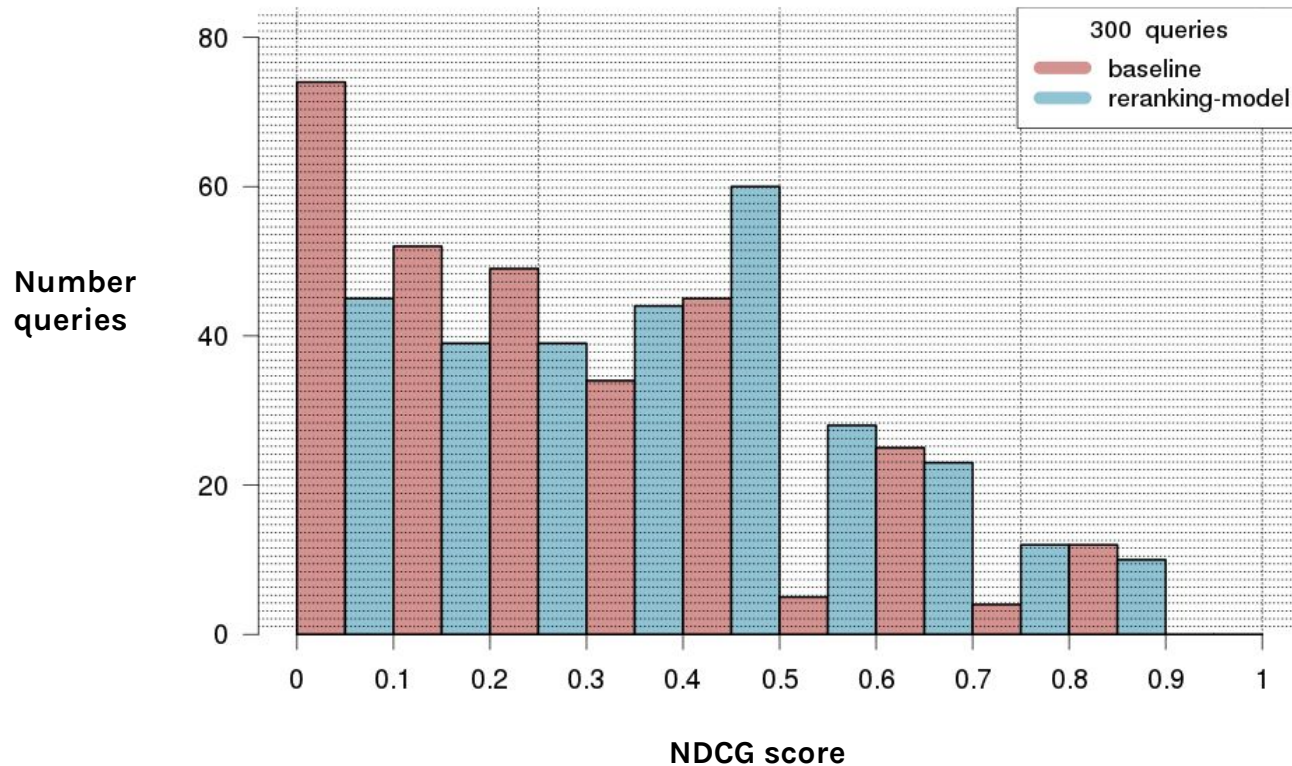


# Scores

<b>model type</b>	<b>algorithm</b>	<b>performance</b>
Linear	Ridge regression	NDCG +6%
Decision tree	LambdaMART	NDCG +16%
Decision tree	Random Forests	NDCG +22%

# Scores: risk vs. reward

"baseline" vs "reranking-model"





# Execution time

- Applying reranking on top 100
  - index: 1,000,000 documents
  - model: 1000 trees, each max. 7 leaves
- Original library: **+22%**

# Execution time

Culprit: transformation from internal API object  
to ranking-library object  
(done for each query-document pair)

feature  
extraction

```
graph LR; A[feature extraction] --> B["double[] features ->  
String features ->  
DataPoint { String relevance_label;  
String query_id;  
String description;  
float[] features}"]; B --> C[ranking model];
```

**double[]** features ->

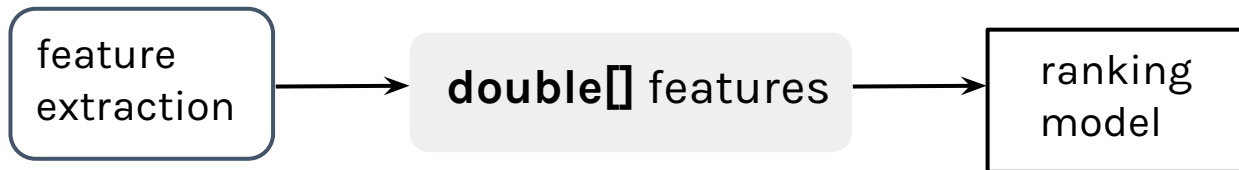
**String** features ->

```
DataPoint { String relevance_label;  
String query_id;  
String description;  
float[] features}
```

ranking  
model

# Execution time

After refactoring model application

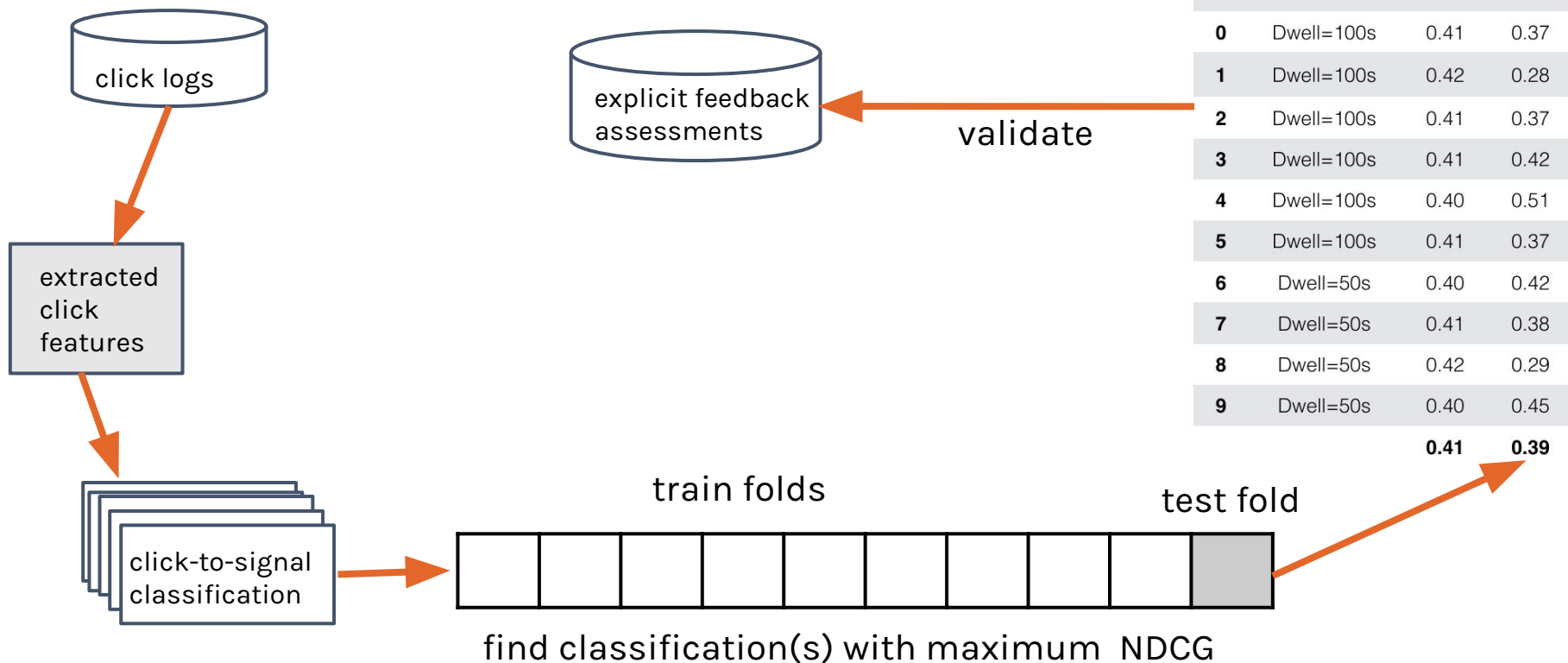


Avg. query execution time increase: **+4%**

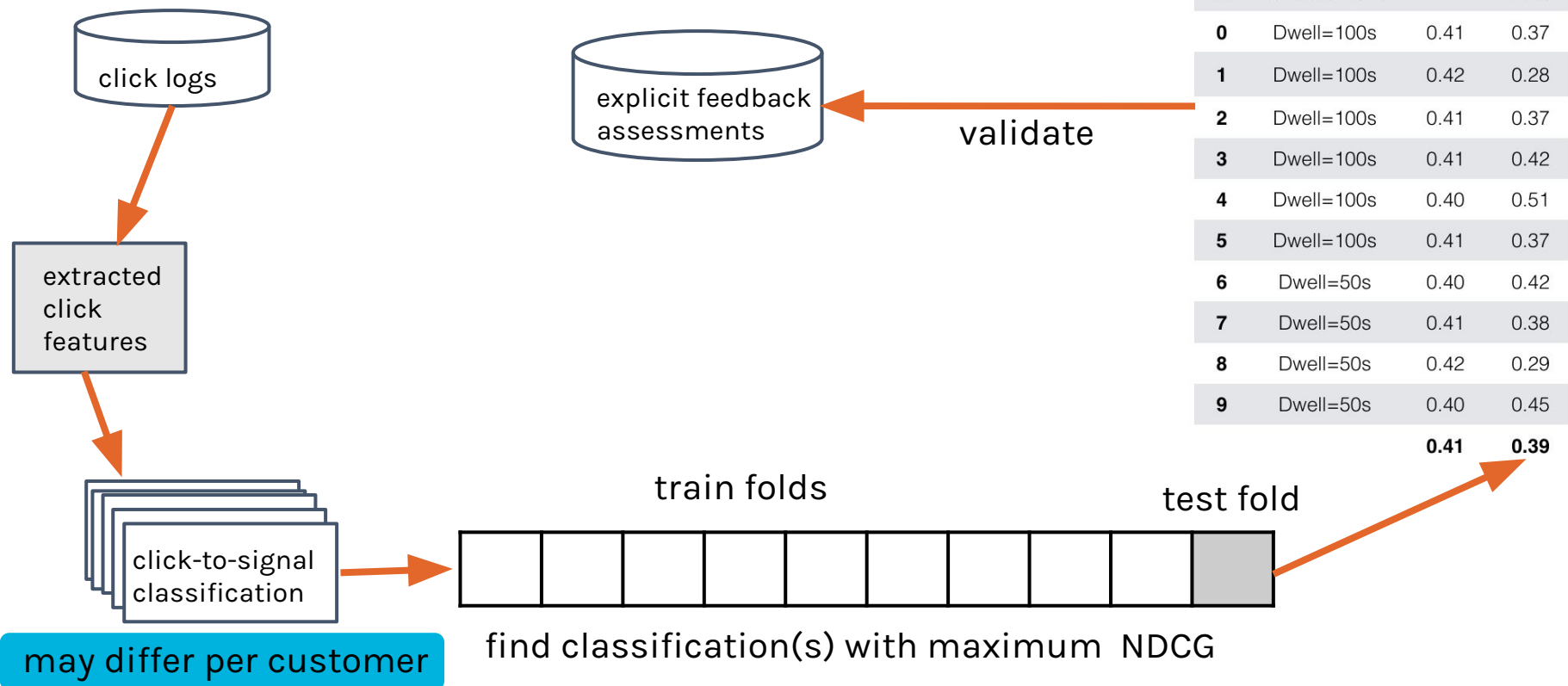
# Next steps: Implicit user feedback gathering

- Transform user actions to feedback signals
  - transformation model may differ per customer
- Avoid modeling an action loop
  - ...unless you want to optimize an action
  - validate with human-made assessments
- Avoid modeling a reinforcing feedback loop
  - deal with position / selection bias

# Implicit feedback gathering



# Implicit feedback gathering



# Conclusions

- Faceted search can be really improved by LTR
  - With minimal impact on execution times
- By determining your general learning objectives
  - Selecting features and algorithm accordingly and in harmony
- Ranking models aren't static
  - Differ in performance per query type / user



Thanks!

***Any questions?***

contact: [vanbelle@textkernel.nl](mailto:vanbelle@textkernel.nl)

join us: [textkernel.careers](https://textkernel.com/careers)