# Learning “Learning to Rank”

Sophie Watson

@sophwats

Strata
DATA CONFERENCE

# Learning “Learning to Rank”

Sophie Watson

@sophwats

Strata
DATA CONFERENCE

Search Wikipedia

NETFLIX

WolframAlpha computational intelligence.

Enter what you want to calculate or know about

Browse Examples Surprise Me

Google Scholar

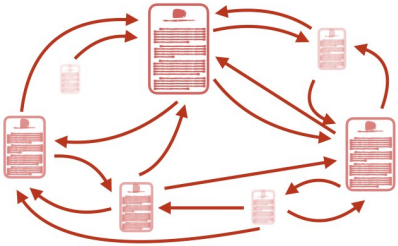
Articles Case law

Bing

Ask

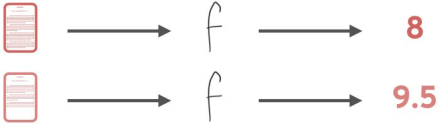
Search Twitter

amazon



dmlc
XGBoost

$f = (\sim, \sim, \sim, \sim, \cdots, \sim, \sim)$



$q = (\sim\ \sim\ \sim\sim)$ **query**

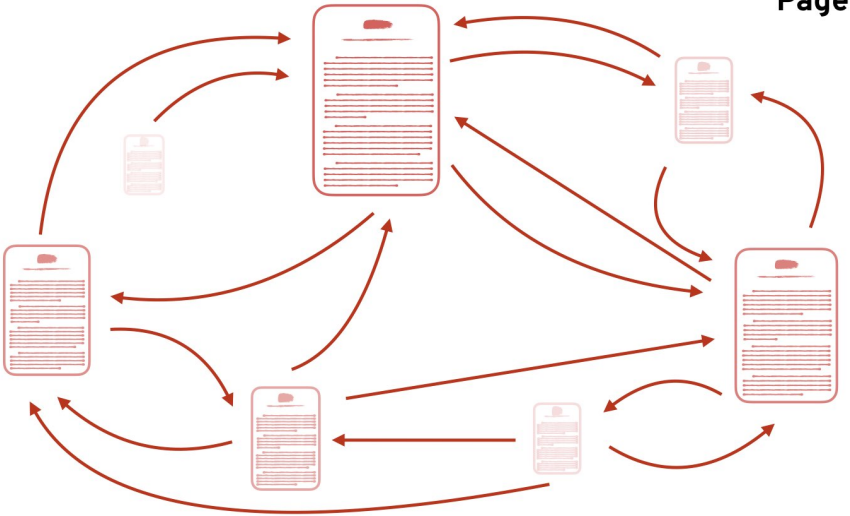
**documents**

$q = (\sim\ \sim\ \sim\sim)$ **query**

**documents**

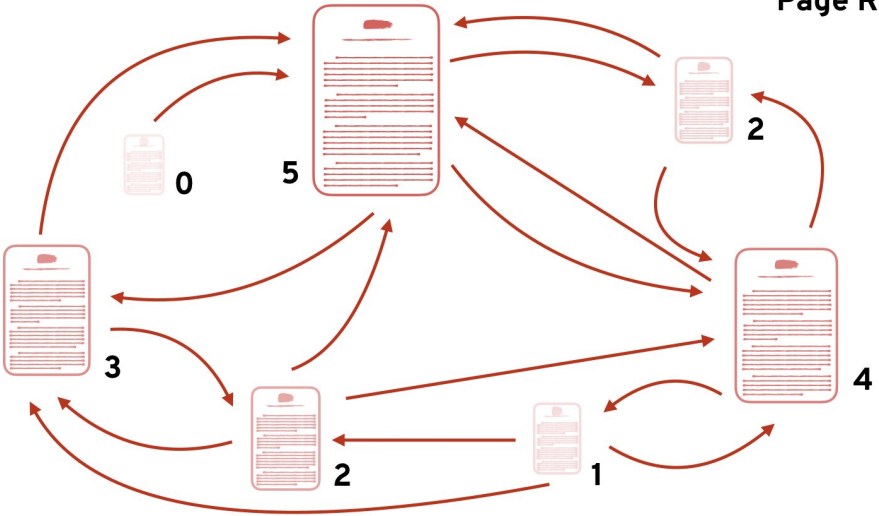
q = (~ ~ ~~~) **query**

# Page Rank

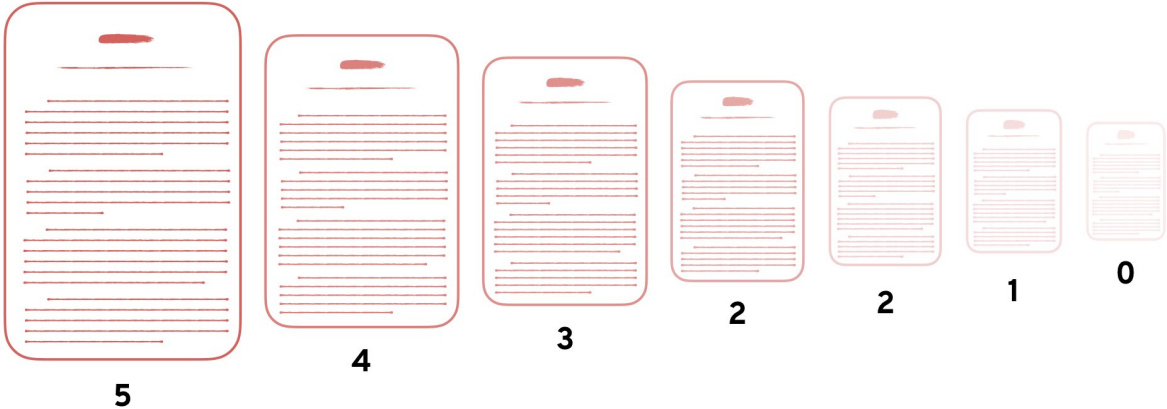


q = (~ ~ ~~~) query

# Page Rank



q = (~ ~ ~~~) **query**

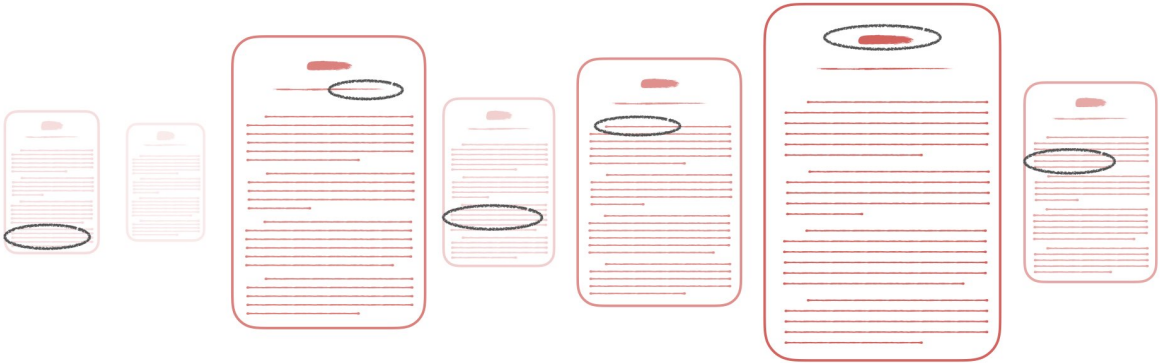


q = ( ) **query**

# Full Text Search

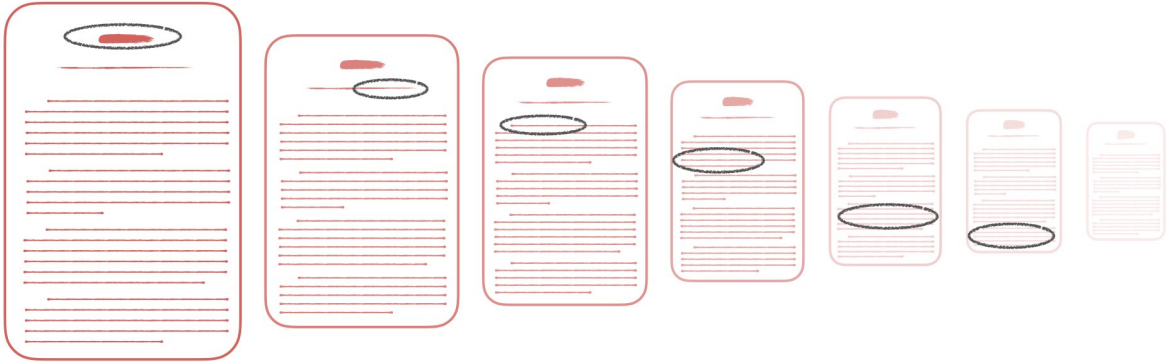
**documents**

# Full Text Search



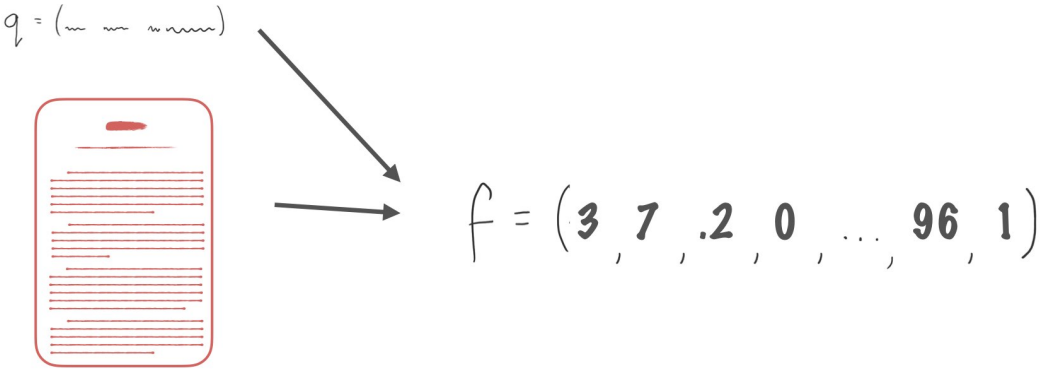
q = ( ) query

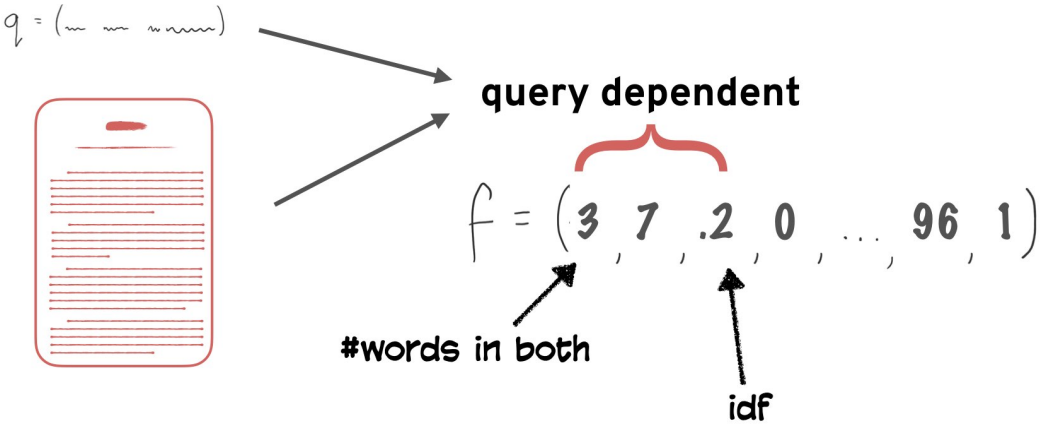
# Full Text Search

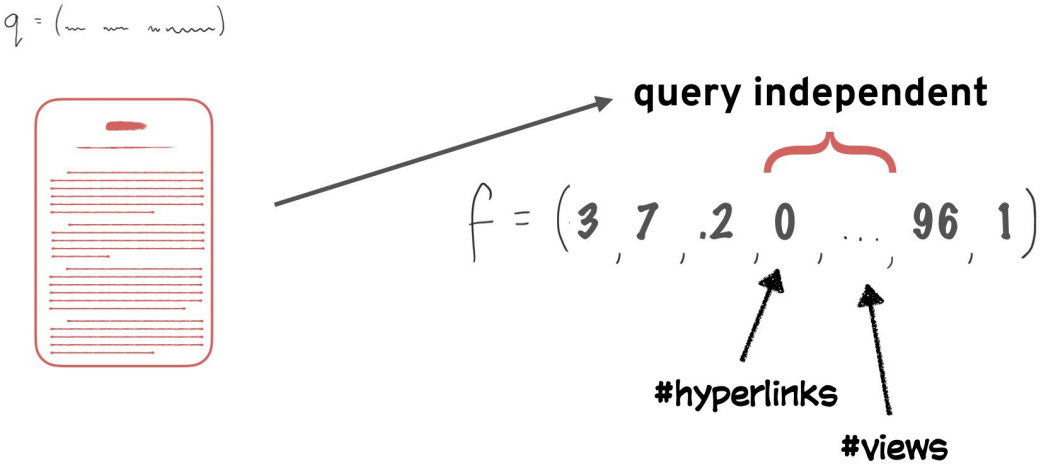


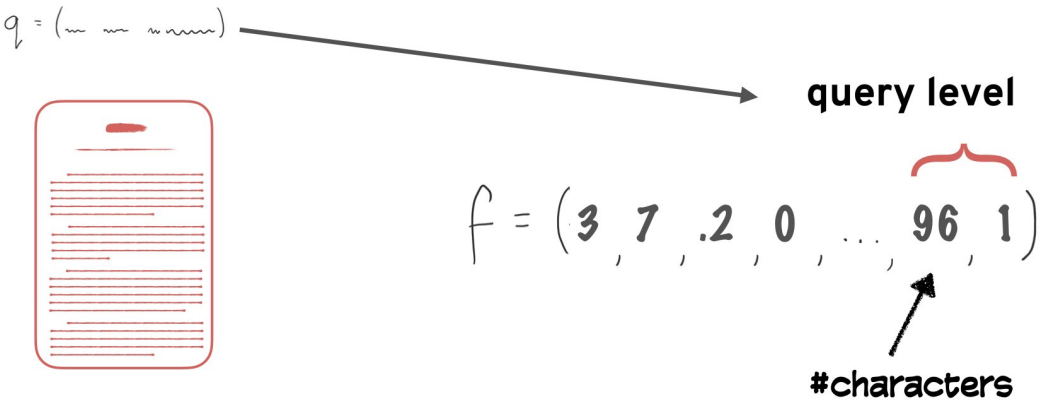
# Feature Engineering

Strata
DATA CONFERENCE


q = ( )
f = (3, 7, .2, 0, ..., 96, 1)


q = ( )
query dependent
f = (3, 7, .2, 0, ..., 96, 1)
#words in both
idf


q = ( )
query independent
f = (3, 7, .2, 0, ..., 96, 1)
#hyperlinks
#views


q = ( )
query level
f = (3, 7, .2, 0, ..., 96, 1)
#characters

# Established Methods

Strata
DATA CONFERENCE

# Pointwise Ranking

Strata
DATA CONFERENCE

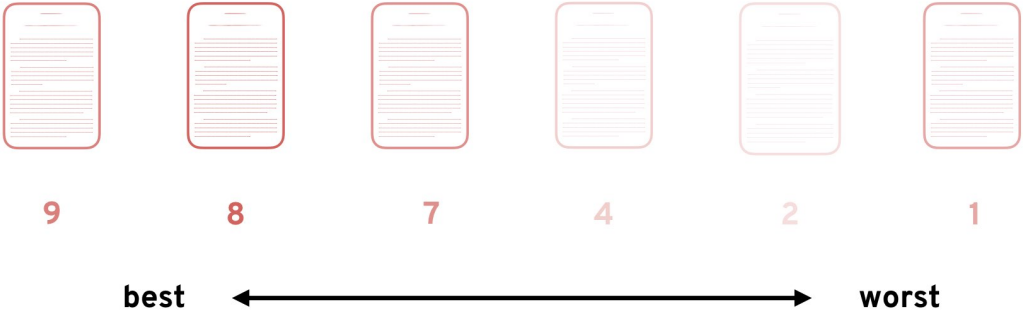
# Pointwise Ranking

Strata
DATA CONFERENCE

# Pointwise Ranking

Strata
DATA CONFERENCE

# Pointwise Ranking



# Pointwise Ranking

| query | feature vector |
| --- | --- |



Strata
DATA CONFERENCE

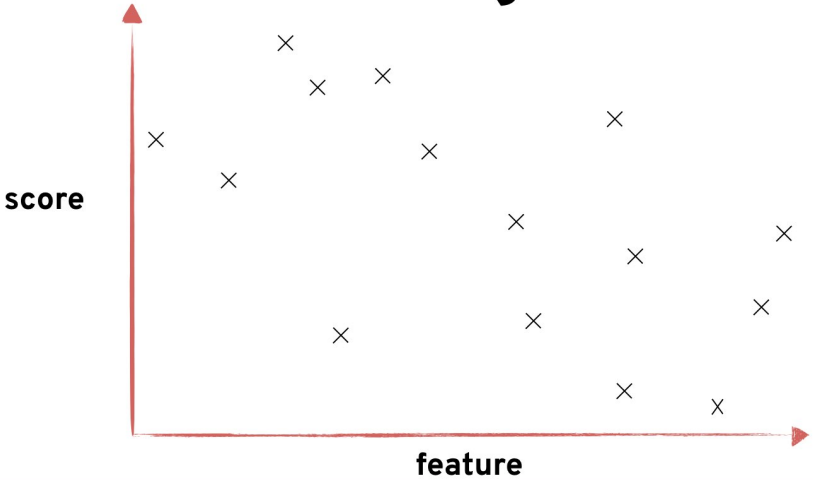
# Pointwise Ranking

| score | query | feature vector |
|---|---|---|

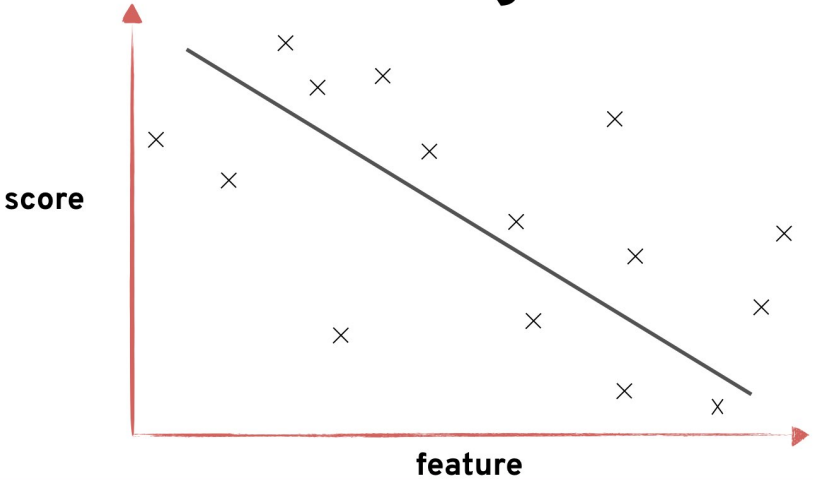


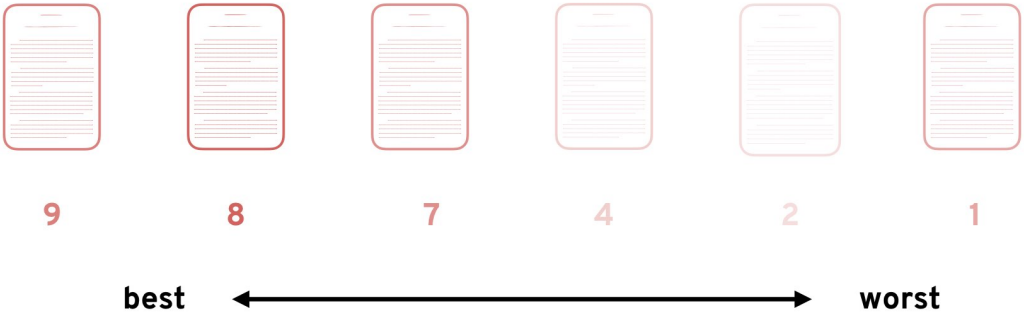
Strata
DATA CONFERENCE

# Pointwise Ranking



# Pointwise Ranking




9
8
7
4
2
1
best
worst

best ⟷ worst

# Pairwise Ranking

Strata
DATA CONFERENCE

# Pairwise Ranking

Rank Net

# Pairwise Ranking

Rank Net

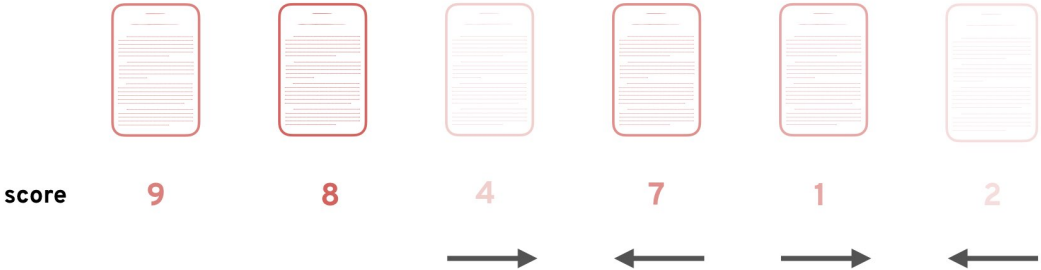
# RankNet

score 9 8 7 4 2 1

# RankNet

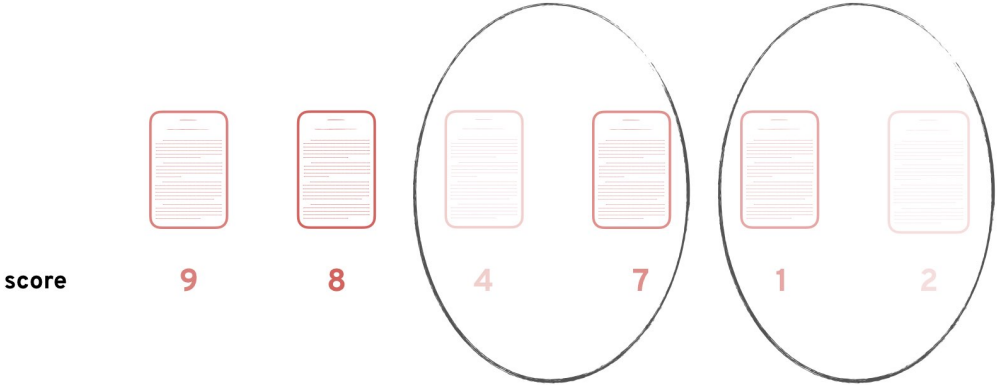
| score | 9 | 8 | 4 | 7 | 1 | 2 |
|---|---|---|---|---|---|---|

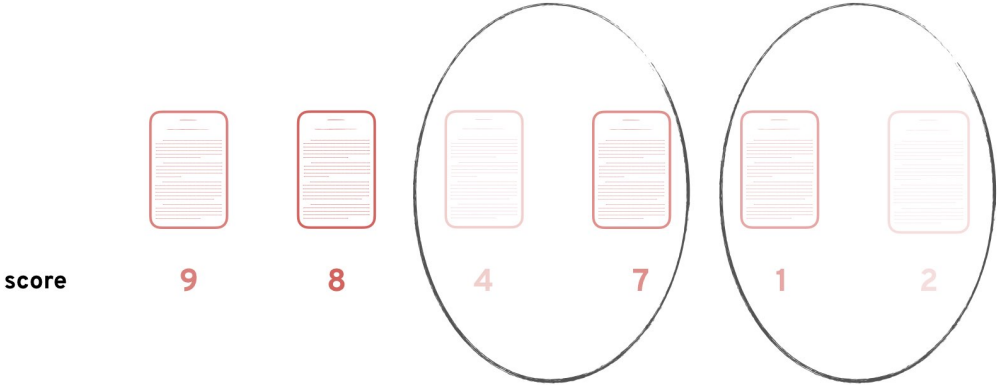
# RankNet



# Normalised Discounted Cumulative Gain

Strata
DATA CONFERENCE


score
9
8
4
7
1
2


score
9
8
4
7
1
2

# ~~Normalised~~ Discounted Cumulative Gain

$$\sum_{\text{docs} \in \text{testing set}} \frac{\text{true score}}{\log_2 \text{predicted rank} + 1}$$

Bigger contribution from start of list

# Normalised Discounted Cumulative Gain

score 9 8 4 7 1 2

$$\text{nDCG} = \frac{\text{discounted cumulative gain}}{\text{optimal discounted cumulative gain}}$$

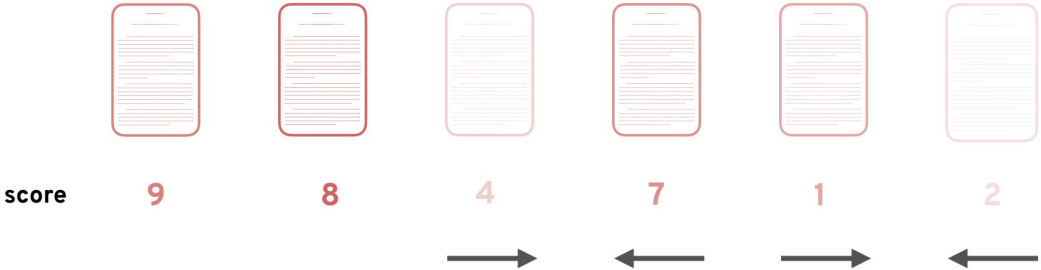
nDCG = 1 is perfect!

nDCG = 0 means it couldn't be worse!

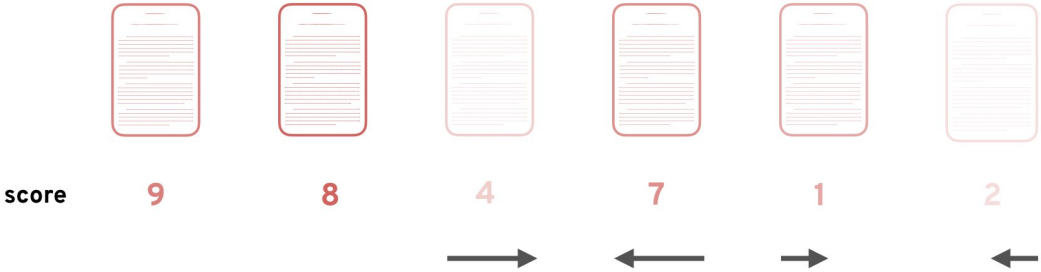
# Normalised Discounted Cumulative Gain

$$\text{nDCG@4} = \frac{\text{discounted cumulative gain for top 4}}{\text{optimal discounted cumulative gain for top 4}}$$

# RankNet



# LambdaRank






Strata
DATA CONFERENCE

# Listwise Algorithms

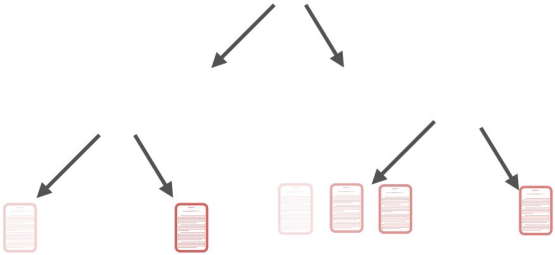
Strata
DATA CONFERENCE

# Listwise Ranking

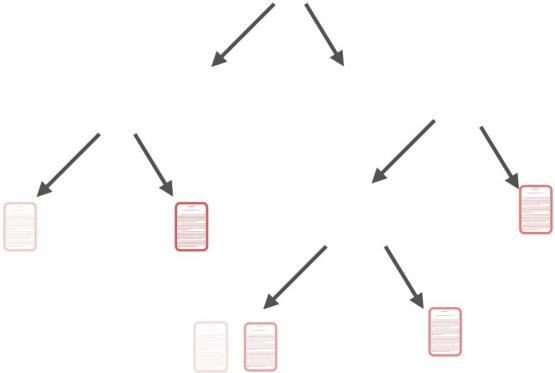
# LambdaMART

# LambdaMART

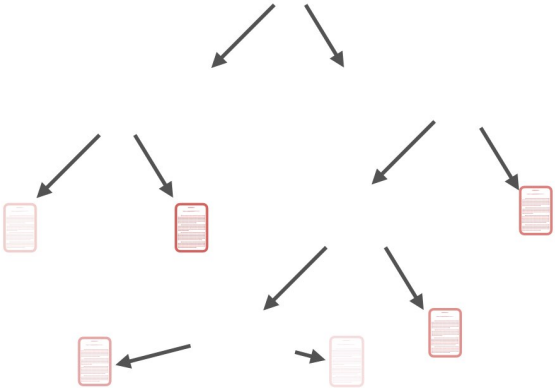
# LambdaMART



# LambdaMART



# LambdaMART



# Learning to Rank in Action

Strata
DATA CONFERENCE

# MSLR-WEB10K

- Open Source
- 10,000 queries
- scored from 0 to 4
- schema: relevance, query id, feature vector
- training, testing and validation data sets

2 qid:13 1:2 2:0 3:2 4:1 5:2 6:1 7:0 8:1 9:0.5 …

133:7 134:0 135:0 136:0

relevance

2 qid:13 1:2 2:0 3:2 4:1 5:2 6:1 7:0 8:1 9:0.5 …

133:7 134:0 135:0 136:0

relevance

2 qid:13 1:2 2:0 3:2 4:1 5:2 6:1 7:0 8:1 9:0.5 …

133:7 134:0 135:0 136:0

query id

relevance

features

2 qid:13 1:2 2:0 3:2 4:1 5:2 6:1 7:0 8:1 9:0.5 …

133:7 134:0 135:0 136:0

query id

```python
from sklearn.linear_model import LinearRegression
```

```python
reg = LinearRegression().fit(X, y)
```

```python
from sklearn.linear_model import LinearRegression
```

```python
reg = LinearRegression().fit(X, y)
```

```python
preds = reg.predict(X_test)
```

```python
preds
```

```
array([0.76923034, 0.29349399, 0.3523804 , ..., 0.80657282, 0.43
872805,
       0.19482099])
```

dmlc

# XGBoost

```python
import xgboost as xgb
```

```python
training_data = xgb.DMatrix('data/train_dat.txt')
testing_data = xgb.DMatrix('data/test_dat.txt')
```

```
[16:11:30] 723412x137 matrix with 98384032 entries loaded from data/train_dat.txt
[16:11:30] 6000 groups are loaded from data/train_dat.txt.group
[16:11:31] 241521x137 matrix with 32846856 entries loaded from data/test_dat.txt
[16:11:31] 2000 groups are loaded from data/test_dat.txt.group
```

eg. [7,8,5]

```python
import xgboost as xgb
```

```python
training_data = xgb.DMatrix('data/train_dat.txt')
testing_data = xgb.DMatrix('data/test_dat.txt')
```

```python
params = [('objective','rank:ndcg'),
          ('max_depth',2), ('eta',0.1), ('num_boost_round',4)]

model = xgb.train(params, training_data)

predictions = model.predict(testing_data)
```

```python
import xgboost as xgb
```

```python
training_data = xgb.DMatrix('data/train_dat.txt')
testing_data = xgb.DMatrix('data/test_dat.txt')
```

```python
params = [('objective','rank:ndcg'),
          ('max_depth',2), ('eta',0.1), ('num_boost_round',4)]

model = xgb.train(params, training_data)

predictions = model.predict(testing_data)
```

**rank:ndcg** LambdaMart (listwise)

**rank:pairwise** LambdaRank (pairwise)

Sophie Watson

sophie@redhat.com

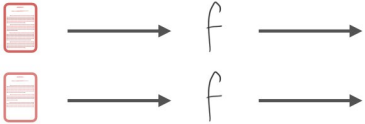
@sophwats

Red Hat

sophwats / learning-to-rank

dmlc XGBoost

$f = (\sim, \sim, \sim, \sim, \cdots, \sim, \sim)$



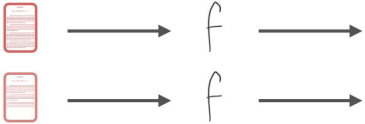
Sophie Watson

sophie@redhat.com

@sophwats

Red Hat

sophwats / learning-to-rank

dmlc XGBoost

$f = (\sim, \sim, \sim, \sim, \cdots, \sim, \sim)$