# Search Engines

Gertjan van Noord

October 7, 2022

# Overview

- Some notes regarding assignments
- Pagerank

# Kappa

$$\mathsf{kappa} = \frac{A - E}{1 - E}$$

*... border line case ...*

# Kappa

$$\text{kappa} = \frac{A - E}{1 - E}$$

What if $A$ equals 1? The result is 1.

# Kappa

$$\text{kappa} = \frac{A - E}{1 - E}$$

What if $A$ equals 1? The result is 1. Except if $E$ equals 1 as well...

In that case, we obtain $0/0$.

$0/0$ is not defined. So, it is wrong to state that kappa is 0 (or 1) in such cases.

# Some Python notes

- Python and typing of variables
- Updating the value of a dict

# Python and typing of variables

Some purists despise Python because of its dynamic typing

# Python and typing of variables

wrong.py

```
import sys
for terms in sys.stdin:
    terms = terms.split()
    print(terms)
```

better.py

```
import sys
for terms in sys.stdin:
    term_list = terms.split()
    print(term_list)
```

Both variants work, but only the second variant is acceptable.

# Python and typing of variables

It is good practice to use "static types" where possible. You can check with the `mypy` tool.

```
$ mypy wrong.py
wrong.py:4: error: Incompatible types in assignment
  (expression has type "List[str]", variable has type "str")
Found 1 error in 1 file (checked 1 source file)
$ mypy better.py
Success: no issues found in 1 source file
```

# Updating the value of a dict

This works:

```
if el in my_dict:
    my_dict[el] =+ 1
else:
    my_dict[el] = 1
```

# Updating the value of a dict

This works:

```
if el in my_dict:
    my_dict[el] =+ 1
else:
    my_dict[el] = 1
```

Where possible, this idiom is preferred:

```
my_dict[el] = my_dict.get(el,0) + 1
```

# Last week: Evaluation

- Precision, Recall, F-measure (F-score)
- Precision and recall at rank
- Interpolated precision
- n-pt average precision, p@n, r@n, R-precision
- Mean Average Precision (MAP)
- Annotator Agreement, Kappa-score

# This week: Pagerank

Some "documents" are more important, popular, authorative than others

Pagerank applied to web search engine: Google

# This week: Pagerank

Some “documents” are more important, popular, authorative than others

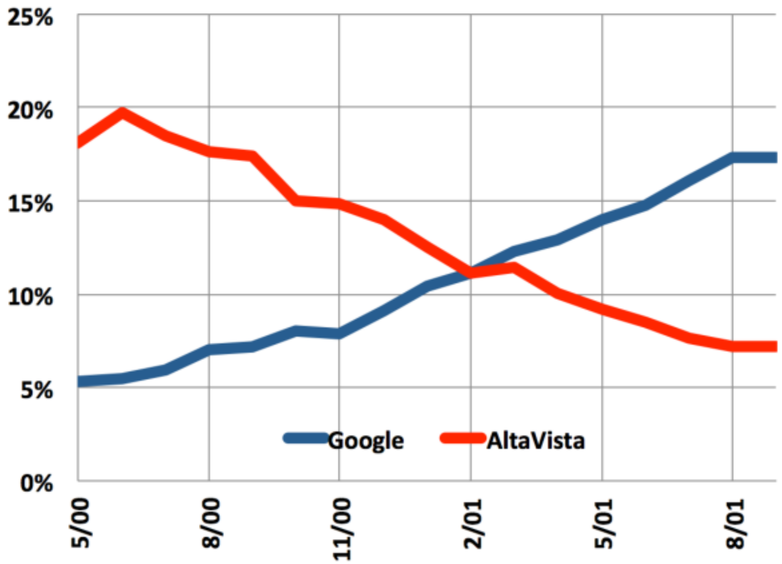
Pagerank applied to web search engine: Google

Ranking of documents not only on their contents, but also on their importance/popularity/authority/. . .

# Search engines on the web

- The first search engine only indexed a few million pages
- In 1995, AltaVista arrives (indexes 20 million pages)
- Altavista was extremely popular from 1995-2000
- In 2000, it suddenly lost almost all its visitors to a new search engine: Google

# Search engine popularity



# What did Google do that Altavista did not do?

It implemented a technique so that “better” web sites are preferred

Using an existing algorithm, PageRank, developed in the field of citation studies

# Citation analysis

How to rank scientific articles?

Scientific articles cite other scientific articles

A scientific article that is cited a lot is a better scientific article

The number of citations indicates the quality of a scientific article

# Citation analysis: pagerank

The number of citations indicates the quality of a scientific article

Why stop there?

Rather than count citations, you can weigh the citations: a citation from an article that is cited a lot counts more heavily

# Pagerank for Web pages

Web pages do not cite other pages, but they refer to other pages through hyperlinks

A web page that many other pages link to is (assumed to be) a “better” page

A web page is better if better web pages link to it

# Pagerank more precise

- Imagine a browser doing a random walk on web pages
  - Start at a random page
  - At each step, select one of the hyperlinks randomly
- If you visit a page often, then apparently there are many ways to get to that page
- In the "steady state", each page has a long-term visit rate – this is the score of that page

# Pagerank more precise

- Imagine a browser doing a random walk on web pages
- In the "steady state", each page has a long-term visit rate – this is the score of that page

Problem: there may be web pages without any outgoing links ("dead ends"). Solution: Teleporting

# Teleporting

- At a dead end, jump to a random web page
- At any non-dead end:
  - with probability $\alpha$ (say 0.1), jump to a random web page, and
  - with probability $1 - \alpha$, take one of the links randomly

# Random walk with teleporting

- cannot get stuck
- there is a long-term rate at which any page is visited
- can we compute this? Yes!

# Markov Chain

- Markov Chain consists of $n$ states (here: web pages)
- For each state $i$ and $j$, we know the probability of going to state $j$ if we are in state $i$
- These probabilities form a *transition probability matrix* **P**.
- The matrix entry $\mathbf{P}_{ij}$ indicates the probability that, if your are in $i$, you now move to $j$.

# Probability Matrix example

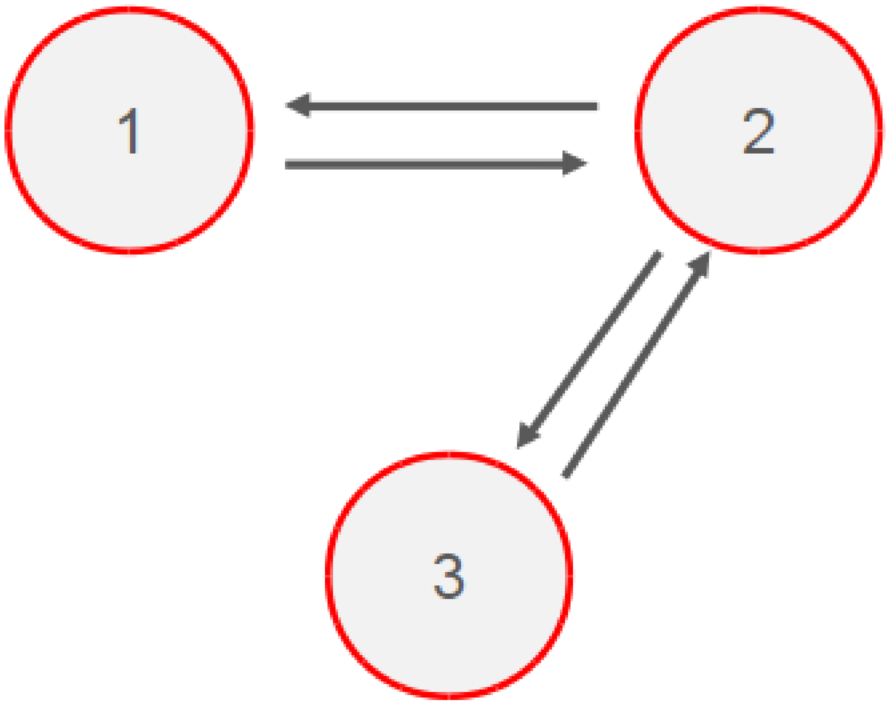
| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0.1 | 0.3 | 0.3 | 0.2 | 0.1 |
| 2 | 0.1 | 0.2 | 0.5 | 0.1 | 0.1 |
| 3 | 0.4 | 0.3 | 0.1 | 0.1 | 0.1 |
| 4 | 0.3 | 0.2 | 0.2 | 0.1 | 0.1 |
| 5 | 0.1 | 0.3 | 0.2 | 0.3 | 0.1 |

Note: the **rows** of the matrix always sum to 1.

# Where do these probabilities come from?

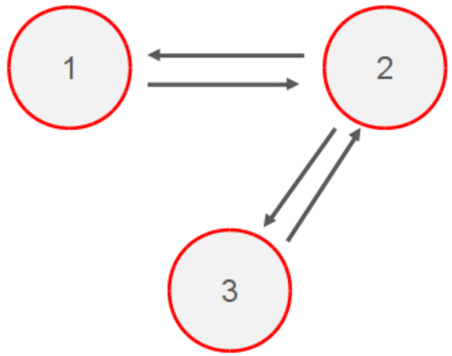
- Just assume every outgoing link is equally probable
- Take care of “teleporting”

# An Example



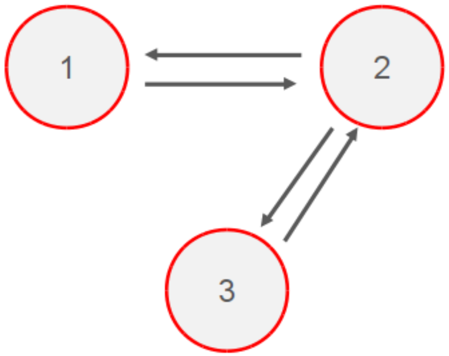

Three web pages. An arrow from 1 to 2 indicates that there is a hyperlink from web page 1 to web page 2.

# An Example




$$\mathbf{P} = \begin{pmatrix} 0 & 1 & 0 \\ 0.5 & 0 & 0.5 \\ 0 & 1 & 0 \end{pmatrix}$$

# Add Teleport




$$\mathbf{P} = \begin{pmatrix} 0 & 1 & 0 \\ 0.5 & 0 & 0.5 \\ 0 & 1 & 0 \end{pmatrix}$$

Suppose $\alpha = 0.5$

$$\mathbf{P} = \begin{pmatrix} 1/6 & 2/3 & 1/6 \\ 5/12 & 1/6 & 5/12 \\ 1/6 & 2/3 & 1/6 \end{pmatrix}$$

(Every cell now gets $1/6 + (1-\alpha) * C$ where $C$ is the probability in the previous matrix

Why 1/6? There are 3 pages in total, each is equally likely. So, the probability of each cell if teleport is used is $\alpha * 1/n$.

In general, the new value for each cell will be: $\alpha * 1/n + (1 - \alpha) * C$

# Probability Vector

A probability vector **x** $= (x_1, \ldots x_n)$ tells us where the walk is at any point.

If we only have 5 states, and we start in state 1, then the vector **x** $= (1, 0, 0, 0, 0)$.

If we are either in state 2 with probability 0.3 or in state 3 with probability 0.7, then the probability vector **x** $= (0, 0.3, 0.7, 0, 0)$.

Of course, the probability vector should always sum to 1.

# One Step

So we have a Probability Matrix $\mathbf{P}$ which tells us what the probabilities are of each step.

And we have a Probability Vector $\mathbf{x}$ which tells us where we are now.

We can then compute the Probability Vector which results if we perform one step. This is written $\mathbf{xP}$.

# One Step: example

**P**:

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0.1 | 0.3 | 0.3 | 0.2 | 0.1 |
| 2 | 0.1 | 0.2 | 0.5 | 0.1 | 0.1 |
| 3 | 0.4 | 0.3 | 0.1 | 0.1 | 0.1 |
| 4 | 0.3 | 0.2 | 0.2 | 0.1 | 0.1 |
| 5 | 0.1 | 0.3 | 0.2 | 0.3 | 0.1 |

$\mathbf{x} = (1, 0, 0, 0, 0)$.

After one step:

$\mathbf{x} = (0.1, 0.3, 0.3, 0.2, 0.1)$.

# One Step: Another example

**P**:

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0.1 | 0.3 | 0.3 | 0.2 | 0.1 |
| 2 | 0.1 | 0.2 | 0.5 | 0.1 | 0.1 |
| 3 | 0.4 | 0.3 | 0.1 | 0.1 | 0.1 |
| 4 | 0.3 | 0.2 | 0.2 | 0.1 | 0.1 |
| 5 | 0.1 | 0.3 | 0.2 | 0.3 | 0.1 |

**x** $= (0, 0.3, 0.7, 0, 0)$.

After one step:

If we had been in state 2, then the next vector would be $(0.1, 0.2, 0.5, 0.1, 0.1) * 0.3 = (0.03, 0.06, 0.15, 0.03, 0.03)$.

If we had been in state 3, then the next vector would be $(0.4, 0.3, 0.1, 0.1, 0.1) * 0.7 = (0.28, 0.21, 0.07, 0.07, 0.07)$.

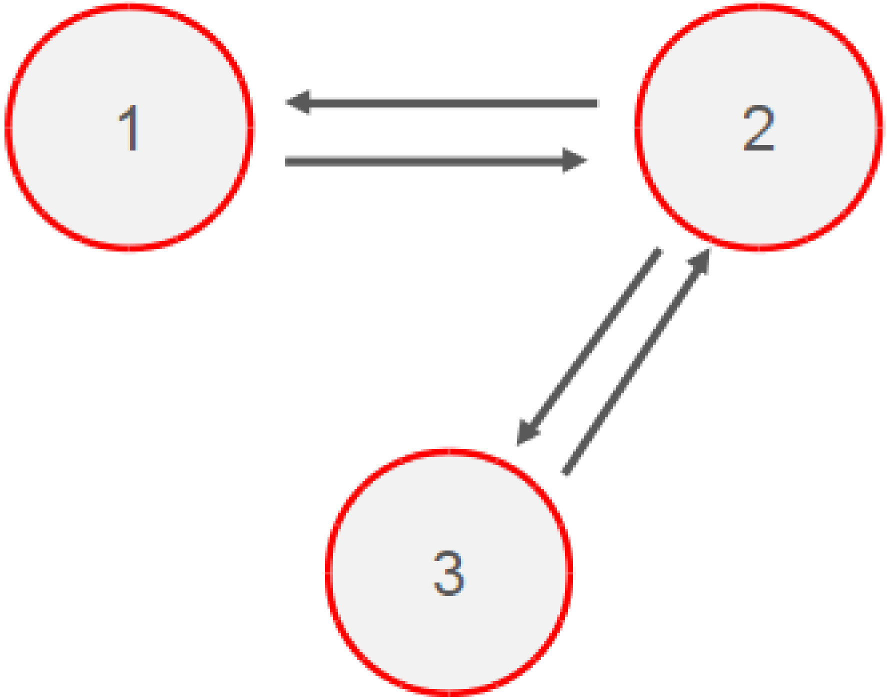
The next **x** is the sum of these:

**x** $= (0.31, 0.27, 0.22, 0.1, 0.1)$.

# One Step: In general

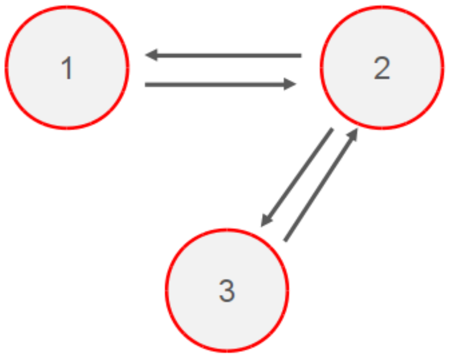
$x_1 *$ row of $x_1$ in $\mathbf{P} + x_2 *$ row of $x_2$ in $\mathbf{P} + \ldots$

# An Example



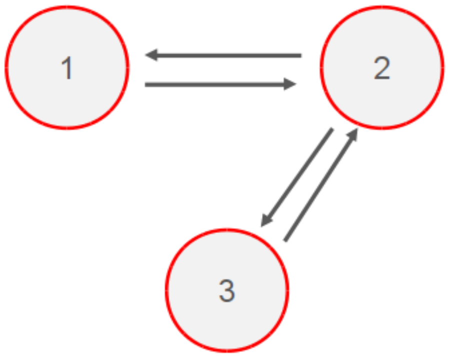

Three web pages. An arrow from 1 to 2 indicates that there is a hyperlink from web page 1 to web page 2.

# An Example




$$\mathbf{P} = \begin{pmatrix} 0 & 1 & 0 \\ 0.5 & 0 & 0.5 \\ 0 & 1 & 0 \end{pmatrix}$$

# An Example with Teleport




Suppose $\alpha = 0.5$

$$\mathbf{P} = \begin{pmatrix} 1/6 & 2/3 & 1/6 \\ 5/12 & 1/6 & 5/12 \\ 1/6 & 2/3 & 1/6 \end{pmatrix}$$

# Compute the visit rate

- Suppose

$$\mathbf{P} = \begin{pmatrix} 1/6 & 2/3 & 1/6 \\ 5/12 & 1/6 & 5/12 \\ 1/6 & 2/3 & 1/6 \end{pmatrix}$$

- and $\mathbf{x}_0 = (1, 0, 0)$

- $\mathbf{x}_1 =$

# Compute the visit rate

- Suppose

$$\mathbf{P} = \begin{pmatrix} 1/6 & 2/3 & 1/6 \\ 5/12 & 1/6 & 5/12 \\ 1/6 & 2/3 & 1/6 \end{pmatrix}$$

- and $\mathbf{x}_0 = (1, 0, 0)$
- $\mathbf{x}_1 = (1/6, 2/3, 1/6)$
- $\mathbf{x}_2 =$

# Compute the visit rate

- Suppose

$$\mathbf{P} = \begin{pmatrix} 1/6 & 2/3 & 1/6 \\ 5/12 & 1/6 & 5/12 \\ 1/6 & 2/3 & 1/6 \end{pmatrix}$$

- and $\mathbf{x}_0 = (1, 0, 0)$
- $\mathbf{x}_1 = (1/6, 2/3, 1/6)$
- $\mathbf{x}_2 = (1/3, 1/3, 1/3)$
- $\mathbf{x}_3 =$

# Compute the visit rate

- Suppose

$$\mathbf{P} = \begin{pmatrix} 1/6 & 2/3 & 1/6 \\ 5/12 & 1/6 & 5/12 \\ 1/6 & 2/3 & 1/6 \end{pmatrix}$$

- and $\mathbf{x}_0 = (1, 0, 0)$
- $\mathbf{x}_1 = (1/6, 2/3, 1/6)$
- $\mathbf{x}_2 = (1/3, 1/3, 1/3)$
- $\mathbf{x}_3 = (1/4, 1/2, 1/4)$
- $\mathbf{x}_n =$

# Compute the visit rate

- Suppose

$$\mathbf{P} = \begin{pmatrix} 1/6 & 2/3 & 1/6 \\ 5/12 & 1/6 & 5/12 \\ 1/6 & 2/3 & 1/6 \end{pmatrix}$$

- and $\mathbf{x}_0 = (1, 0, 0)$
- $\mathbf{x}_1 = (1/6, 2/3, 1/6)$
- $\mathbf{x}_2 = (1/3, 1/3, 1/3)$
- $\mathbf{x}_3 = (1/4, 1/2, 1/4)$
- $\mathbf{x}_n = (5/18, 4/9, 5/18)$

# Compute the visit rate

- *At some point,* $x_{n+1} = x_n$. "Fixed point"

- This fixed point is the "principal left eigenvector of **P**", and is called the "PageRank" in this context.

- *It does not matter in which web page you start!*

# Compute the visit rate

- *It does not matter in which web page you start!*

- Exercise: do the previous example with $\mathbf{x}_0 = (0, 0, 1)$

# Another example

There are four web pages a, b, c and d.

Web-page a links to b and c.
Web-page b links to c.
Web-page c links to a and b.
Web-page d links to a, b and c.

1. What is the probability matrix if teleport rate = 0 ?

2. What is the probability matrix if teleport rate = 0.85 ?

3. In that case, what is the pagerank vector?

# What to do with the pagerank vector

- higher score indicates better page
- combine this pagerank score with e.g. tfidf using some weighting scheme

# How to fool Google?

. . . or: how to do “search engine optimization”?