Query Likelihood Model

Jaime Arguello
INLS 509: Information Retrieval
jarguell@email.unc.edu

Outline

Introduction to language modeling

Language modeling for information retrieval

Query-likelihood retrieval model

Smoothing

Document priors

What is a language model?

"The goal of a language model is to assign a probability to a sequence of words by means of a probability distribution"

--Wikipedia

What is a language model?

- To understand what a language model is, we have to understand what a probability distribution is
- To understand what a probability distribution is, we have to understand what a discrete random variable is

What is a discrete random variable?

- A is a discrete random variable if:
  - A describes an event with a finite number of possible outcomes (this property makes the random variable discrete)
  - A describes an event whose outcome has some degree of uncertainty (this property makes the variable random)

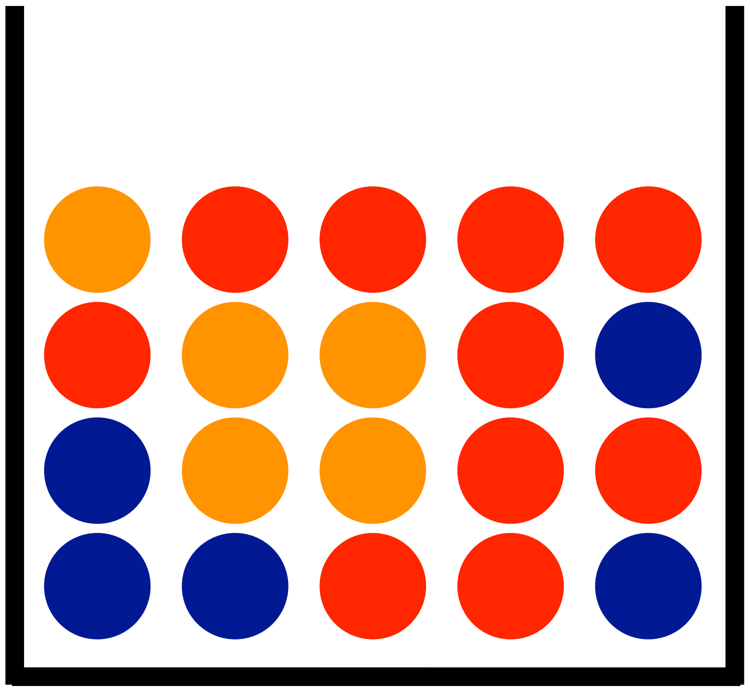
Discrete Random Variables

examples

- A = it will rain tomorrow
- A = the coin-flip will show heads
- A = you will win the lottery in your lifetime
- A = you will find the next couple of slides fascinating

What is a probability distribution?

- A probability distribution gives the probability of each possible outcome of a random variable
- P(RED) = probability that you will reach into <u>this</u> bag and pull out a red ball
- P(BLUE) = probability that you will reach into <u>this</u> bag and pull out a blue ball
- P(ORANGE) = probability that you will reach into <u>this</u> bag and pull out an orange ball



What is a probability distribution?

- For it to be a probability distribution, two conditions must be satisfied:
  - the probability assigned to each possible outcome must be between 0 and 1 (inclusive)
  - the <u>sum</u> of probabilities across outcomes must be 1

$$0 \leq P(\text{RED}) \leq 1$$

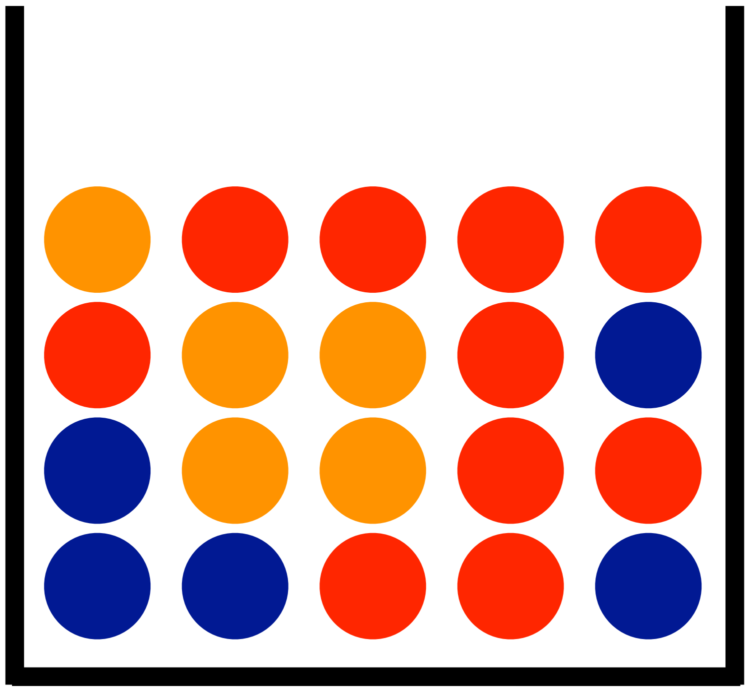
$$0 \leq P(\text{BLUE}) \leq 1$$

$$0 \leq P(\text{ORANGE}) \leq 1$$

$$P(\text{RED}) + P(\text{BLUE}) + P(\text{ORANGE}) = 1$$

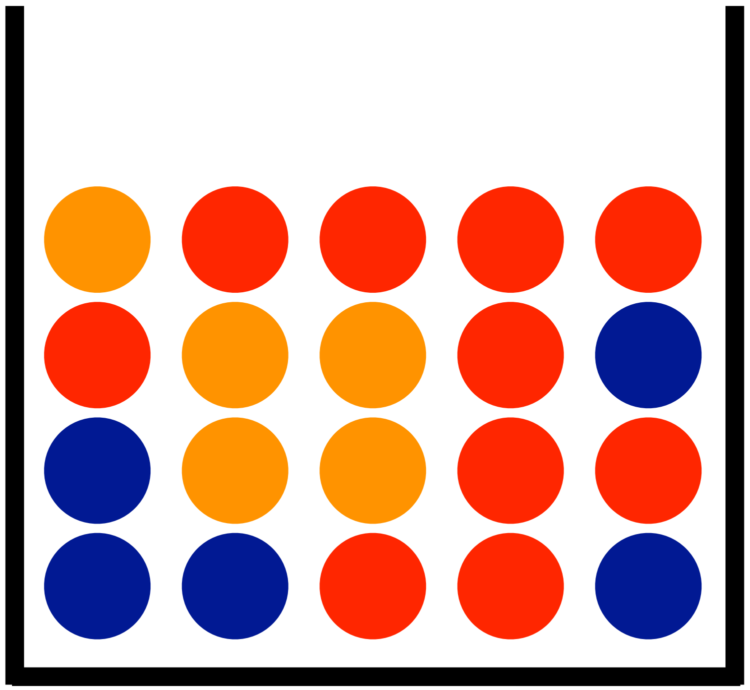
Estimating a Probability Distribution

- Let's estimate these probabilities based on what we know about the contents of the bag
- P(RED) = ?
- P(BLUE) = ?
- P(ORANGE) = ?

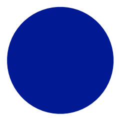
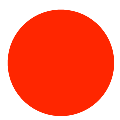
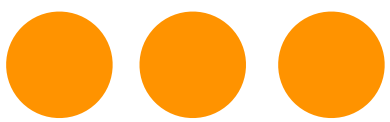
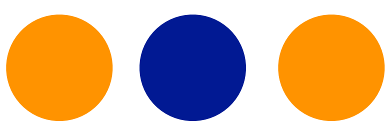
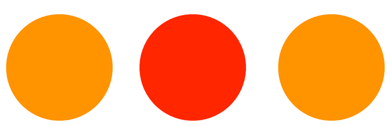
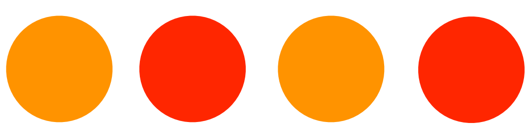


Estimating a Probability Distribution

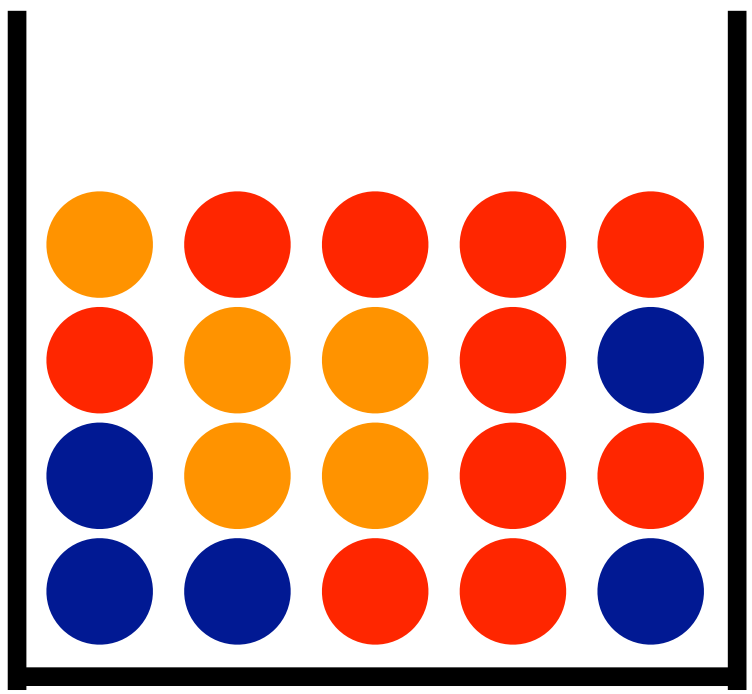
- Let's estimate these probabilities based on what we know about the contents of the bag
- P(RED) = 10/20 = 0.5
- P(BLUE) = 5/20 = 0.25
- P(ORANGE) = 5/20 = 0.25
- P(RED) + P(BLUE) + P(ORANGE) = 1.0



What can we do with a probability distribution?

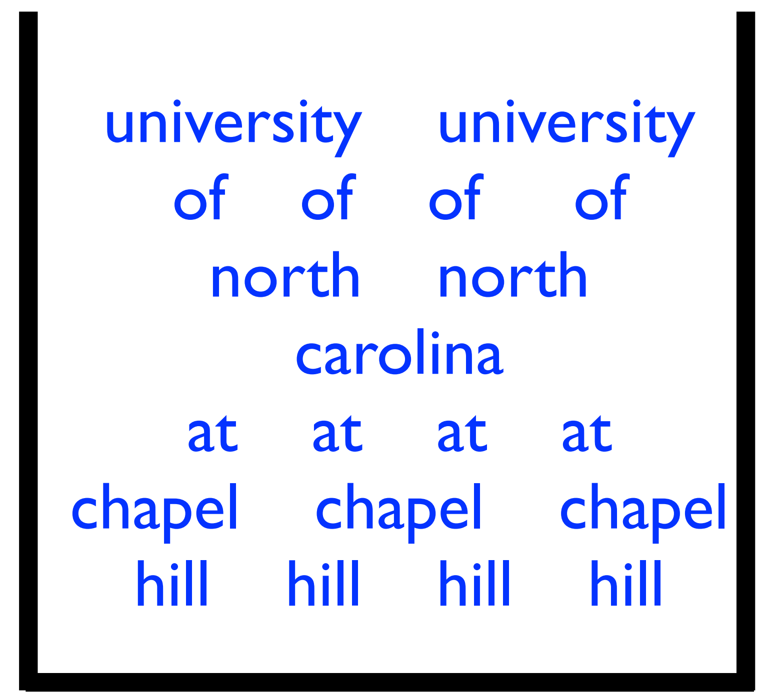
- P() = 0.25
- P() = 0.5
- P() = 0.25 x 0.25 x 0.25
- P() = 0.25 x 0.25 x 0.25
- P() = 0.25 x 0.50 x 0.25
- P() = 0.25 x 0.50 x 0.25 x 0.50

P(RED) = 0.5
P(BLUE) = 0.25
P(ORANGE) = 0.25



Unigram Language Model

- Defines a probability distribution over <u>individual</u> words
  - P(university) = 2/20
  - P(of) = 4/20
  - P(north) = 2/20
  - P(carolina) = 1/20
  - P(at) = 4/20
  - P(chapel) = 3/20
  - P(hill) = 4/20



Unigram Language Model

- It is called a unigram language model because we estimate (and predict) the likelihood of each word independent of any other word
- Assumes that words are independent!
    - The probability of seeing “tarheels” is the same, even if the previously sampled word is “carolina”
- Other language models take context into account
- Those work better for applications like speech recognition or automatic language translation
- Unigram models work well for information retrieval

Unigram Language Model

- Sequences of words can be assigned a probability by multiplying their <u>individual</u> probabilities:

P(university of north carolina) =

P(university) x P(of) x P(north) x P(carolina) =

(2/20) x (4/20) x (2/20) x (1/20) = 0.0001

P(chapel hill) =

P(chapel) x P(hill) =

(3/20) x (4/20) = 0.03

Unigram Language Model

- There are two important steps in language modeling
  - estimation: observing text and estimating the probability of each word
  - prediction: using the language model to assign a probability to a span of text

Outline

Introduction to language modeling

Language modeling for information retrieval

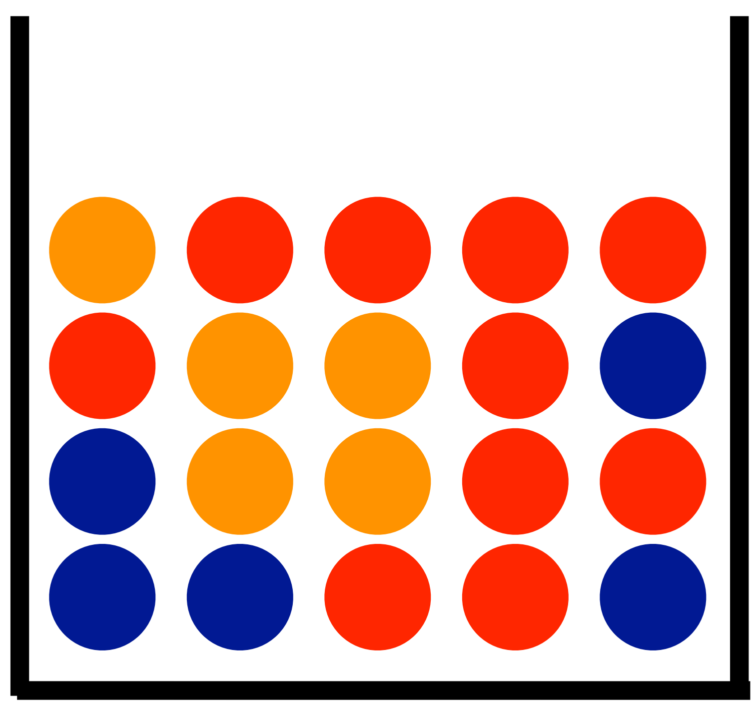
Query-likelihood Retrieval Model

Smoothing

Pseudo-relevance feedback and priors

Language Models

- A language model is a probability distribution defined over a particular vocabulary
- In this analogy, each color represents a vocabulary term and each ball represents a term occurrence in the text used to estimate the language model



P(RED) = 0.5
P(BLUE) = 0.25
P(ORANGE) = 0.25

Document Language Models

- Estimating a document's language model:
  1. tokenize/split the document text into terms
  2. count the number of times each term occurs ($tf_{t,D}$)
  3. count the total number of term occurrences (N_D)
  4. assign term t a probability equal to:

$$\frac{tf_{t,D}}{N_D}$$

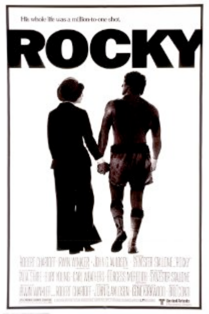
Document Language Models

- The language model estimated from document ***D*** is sometimes denoted as:

$$\theta_D$$

- The probability given to term ***t*** by the language model estimated from document ***D*** is sometimes denoted as:

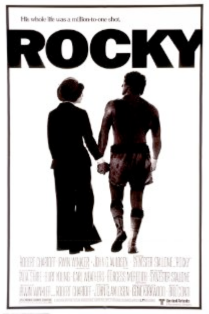
$$P(t|D) = P(t|\theta_D) = \frac{tf_{t,D}}{N_D}$$



Document Language Models

- Movie: Rocky (1976)
- Plot:

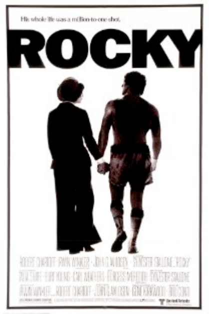
  Rocky Balboa is a struggling boxer trying to make the big time. Working in a meat factory in Philadelphia for a pittance, he also earns extra cash as a debt collector. When heavyweight champion Apollo Creed visits Philadelphia, his managers want to set up an exhibition match between Creed and a struggling boxer, touting the fight as a chance for a "nobody" to become a "somebody". The match is supposed to be easily won by Creed, but someone forgot to tell Rocky, who sees this as his only shot at the big time. Rocky Balboa is a small-time boxer who lives in an apartment in Philadelphia, Pennsylvania, and his career has so far not gotten off the canvas. Rocky earns a living by collecting debts for a loan shark named Gazzo, but Gazzo doesn't think Rocky has the viciousness it takes to beat up deadbeats. Rocky still boxes every once in a while to keep his boxing skills sharp, and his ex-trainer, Mickey, believes he could've made it to the top if he was willing to work for it. Rocky, goes to a pet store that sells pet supplies, and this is where he meets a young woman named Adrian, who is extremely shy, with no ability to talk to men. Rocky befriends her. Adrain later surprised Rocky with a dog from the pet shop that Rocky had befriended. Adrian's brother Paulie, who works for a meat packing company, is thrilled that someone has become interested in Adrian, and Adrian spends Thanksgiving with Rocky. Later, they go to Rocky's apartment, where Adrian explains that she has never been in a man's apartment before. Rocky sets her mind at ease, and they become lovers. Current world heavyweight boxing champion Apollo Creed comes up with the idea of giving an unknown a shot at the title. Apollo checks out the Philadelphia boxing scene, and chooses Rocky. Fight promoter Jergens gets things in gear, and Rocky starts training with Mickey. After a lot of training, Rocky is ready for the match, and he wants to prove that he can go the distance with Apollo. The 'Italian Stallion', Rocky Balboa, is an aspiring boxer in downtown Philadelphia. His one chance to make a better life for himself is through his boxing and Adrian, a girl who works in the local pet store. Through a publicity stunt, Rocky is set up to fight Apollo Creed, the current heavyweight champion who is already set to win. But Rocky really needs to triumph, against all the odds...



Document Language Models

language model estimation (top 20 terms)

term	$tf_{t,D}$	N_D	$P(term\|D)$	term	$tf_{t,D}$	N_D	$P(term\|D)$
a	22	420	0.05238	creed	5	420	0.01190
rocky	19	420	0.04524	philadelphia	5	420	0.01190
to	18	420	0.04286	has	4	420	0.00952
the	17	420	0.04048	pet	4	420	0.00952
is	11	420	0.02619	boxing	4	420	0.00952
and	10	420	0.02381	up	4	420	0.00952
in	10	420	0.02381	an	4	420	0.00952
for	7	420	0.01667	boxer	4	420	0.00952
his	7	420	0.01667	s	3	420	0.00714
he	6	420	0.01429	balboa	3	420	0.00714

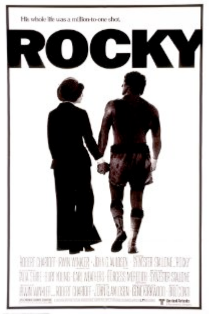


Document Language Models

language model estimation (top 20 terms)

term	$tf_{t,D}$	N_D	*P(term|D)*	*term*	$tf_{t,D}$	N_D	*P(term|D)*
a	**22**	**420**	**0.05238**	creed	5	420	0.01190
rocky	**19**	**420**	**0.04524**	philadelphia	5	420	0.01190
to	18	420	0.04286	has	4	420	0.00952
the	17	420	0.04048	pet	4	420	0.00952
is	**11**	**420**	**0.02619**	boxing	4	420	0.00952
and	10	420	0.02381	up	4	420	0.00952
in	10	420	0.02381	an	4	420	0.00952
for	7	420	0.01667	**boxer**	**4**	**420**	**0.00952**
his	7	420	0.01667	s	3	420	0.00714
he	6	420	0.01429	balboa	3	420	0.00714

- What is the probability given by this language model to the sequence of text “rocky is a boxer”?

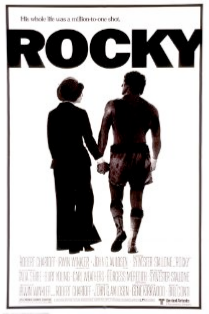


Document Language Models

language model estimation (top 20 terms)

term	$tf_{t,D}$	N_D	P(term|D)	term	$tf_{t,D}$	N_D	P(term|D)
a	**22**	**420**	**0.05238**	creed	5	420	0.01190
rocky	19	420	0.04524	philadelphia	5	420	0.01190
to	18	420	0.04286	has	4	420	0.00952
the	17	420	0.04048	**pet**	**4**	**420**	**0.00952**
is	**11**	**420**	**0.02619**	boxing	4	420	0.00952
and	10	420	0.02381	up	4	420	0.00952
in	10	420	0.02381	an	4	420	0.00952
for	7	420	0.01667	**boxer**	**4**	**420**	**0.00952**
his	7	420	0.01667	s	3	420	0.00714
he	6	420	0.01429	balboa	3	420	0.00714

- What is the probability given by this language model to the sequence of text “a boxer is a pet”?



Document Language Models

language model estimation (top 20 terms)

term	$tf_{t,D}$	N_D	*P(term|D)*	*term*	$tf_{t,D}$	N_D	*P(term|D)*
a	**22**	**420**	**0.05238**	creed	5	420	0.01190
rocky	19	420	0.04524	philadelphia	5	420	0.01190
to	18	420	0.04286	has	4	420	0.00952
the	17	420	0.04048	pet	4	420	0.00952
is	**11**	**420**	**0.02619**	boxing	4	420	0.00952
and	10	420	0.02381	up	4	420	0.00952
in	10	420	0.02381	an	4	420	0.00952
for	7	420	0.01667	**boxer**	**4**	**420**	**0.00952**
his	7	420	0.01667	s	3	420	0.00714
he	6	420	0.01429	balboa	3	420	0.00714

- What is the probability given by this language model to the sequence of text "a boxer is a dog"?

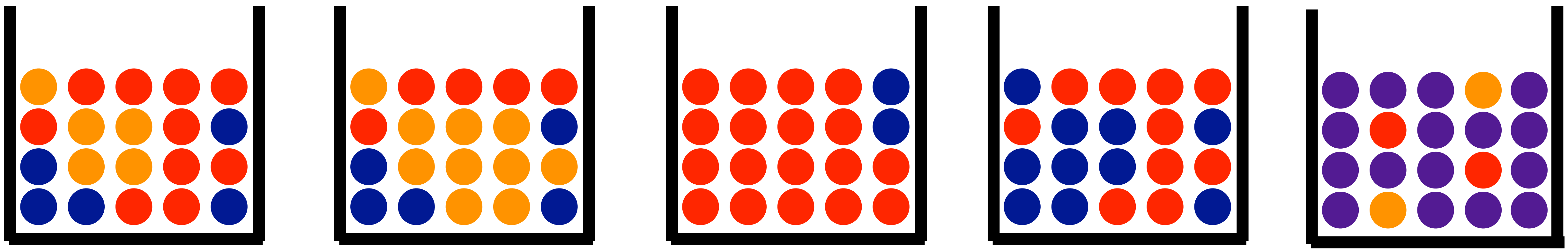
Query-Likelihood Retrieval Model

- Objective: rank documents based on the probability that they are on the same topic as the query
- Solution:
  - Score each document (denoted by ***D***) according to the probability given by its language model to the query (denoted by ***Q***)
  - Rank documents in descending order of score

$$score(Q, D) = P(Q|\theta_D) = \prod_{i=1}^{n} P(q_i|\theta_D)$$

Query-Likelihood Model

back to our analogy

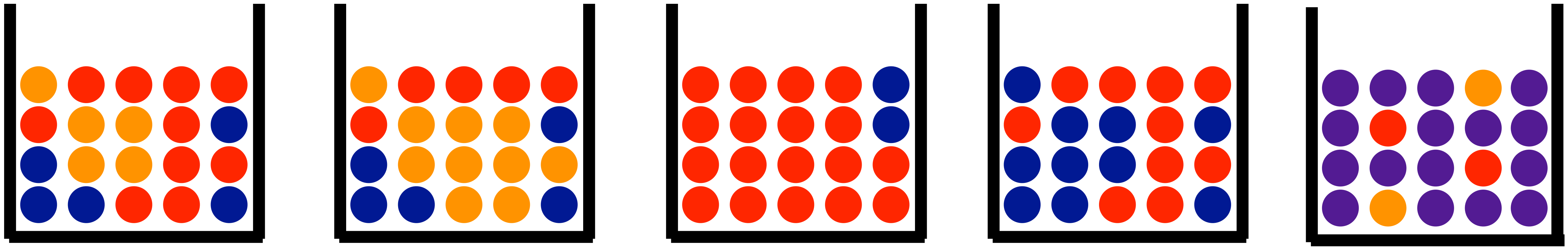


	D_1	D_2	D_3	D_5	D_6
P(RED)	0.50	0.25	0.90	0.50	0.10
P(BLUE)	0.25	0.25	0.10	0.50	0.80
P(ORANGE)	0.25	0.50	0.00	0.00	0.10

- Each document is scored according to the probability that it “generated” the query
- What does it mean for a document to “generate” the query?
- Sample query terms with replacement

Query-Likelihood Model

back to our analogy

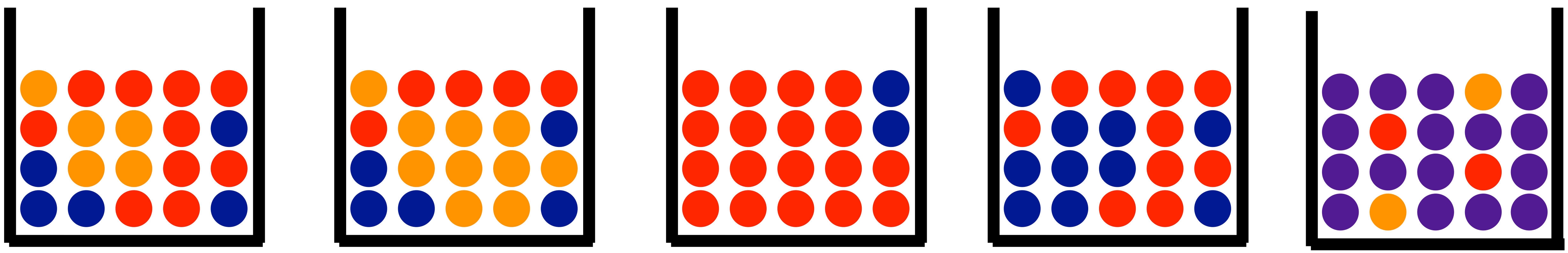


D_1	D_2	D_3	D_5	D_6
P(RED) = 0.50	P(RED) = 0.25	P(RED) = 0.90	P(RED) = 0.50	P(RED) = 0.10
P(BLUE) = 0.25	P(BLUE) = 0.25	P(BLUE) = 0.10	P(BLUE) = 0.50	P(BLUE) = 0.80
P(ORANGE) = 0.25	P(ORANGE) = 0.50	P(ORANGE) = 0.00	P(ORANGE) = 0.00	P(ORANGE) = 0.10

- Query = ●●●
- Which would be the top-ranked document and what would be its score?

Query-Likelihood Model

back to our analogy

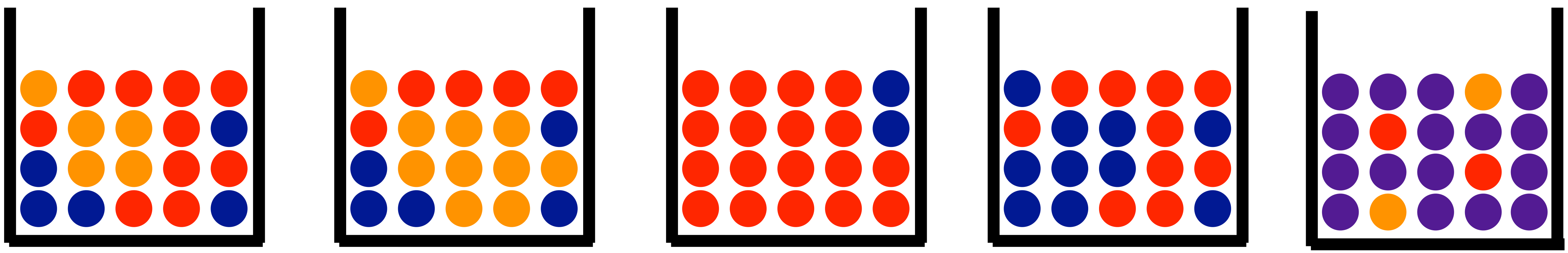


D_1	D_2	D_3	D_5	D_6
P(RED) = 0.50	P(RED) = 0.25	P(RED) = 0.90	P(RED) = 0.50	P(RED) = 0.10
P(BLUE) = 0.25	P(BLUE) = 0.25	P(BLUE) = 0.10	P(BLUE) = 0.50	P(BLUE) = 0.80
P(ORANGE) = 0.25	P(ORANGE) = 0.50	P(ORANGE) = 0.00	P(ORANGE) = 0.00	P(ORANGE) = 0.10

- Query =
- Which would be the top-ranked document and what would be its score?

Query-Likelihood Model

back to our analogy



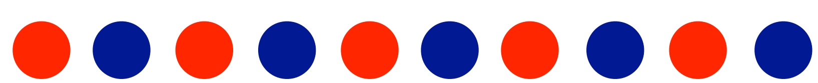
D_1
P(RED) = 0.50
P(BLUE) = 0.25
P(ORANGE) = 0.25

D_2
P(RED) = 0.25
P(BLUE) = 0.25
P(ORANGE) = 0.50

D_3
P(RED) = 0.90
P(BLUE) = 0.10
P(ORANGE) = 0.00

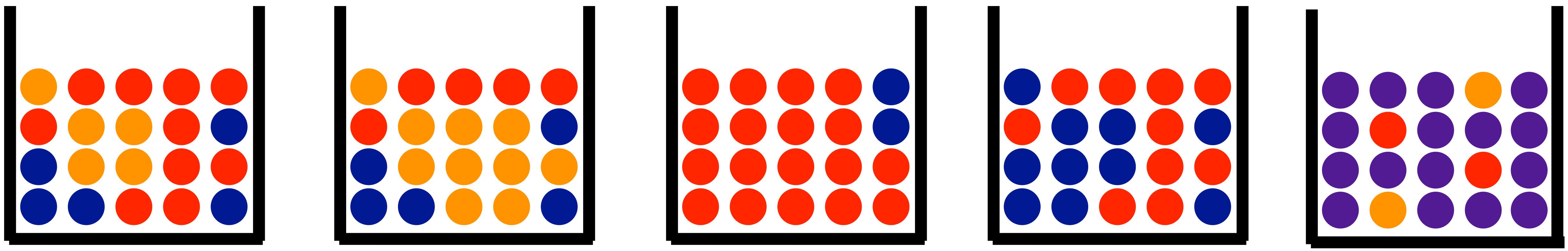
D_5
P(RED) = 0.50
P(BLUE) = 0.50
P(ORANGE) = 0.00

D_6
P(RED) = 0.10
P(BLUE) = 0.80
P(ORANGE) = 0.10

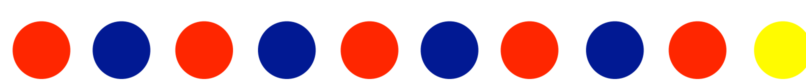
- Query = 
- Which would be the top-ranked document and what would be its score?

Query-Likelihood Model

back to our analogy



D_1	D_2	D_3	D_5	D_6
P(RED) = 0.50	P(RED) = 0.25	P(RED) = 0.90	P(RED) = 0.50	P(RED) = 0.10
P(BLUE) = 0.25	P(BLUE) = 0.25	P(BLUE) = 0.10	P(BLUE) = 0.50	P(BLUE) = 0.80
P(ORANGE) = 0.25	P(ORANGE) = 0.50	P(ORANGE) = 0.00	P(ORANGE) = 0.00	P(ORANGE) = 0.10

- Query = 
- Which would be the top-ranked document and what would be its score?

Query-Likelihood Retrieval Model

- Because we are multiplying query-term probabilities, the longer the query, the lower the document scores (from all documents)
- Is this a problem?

Query-Likelihood Retrieval Model

- Because we are multiplying query-term probabilities, the longer the query, the lower the document scores (from all documents)
- Is this a problem?
- No, because we're scoring documents for the <u>same</u> query

Query-Likelihood Retrieval Model

$$score(Q, D) = P(Q|\theta_D) = \prod_{i=1}^{n} P(q_i|\theta_D)$$

- There are (at least) two issues with this scoring function
- What are they?

Query-Likelihood Retrieval Model

- A document with a single missing query-term will receive a score of zero (similar to boolean AND)
- Where is IDF?
  - Don't we want to suppress the contribution of terms that are frequent in the document, but not frequent in general (appear in many documents)?

Outline

Introduction to language modeling

Language modeling for information retrieval

Query-likelihood retrieval model

Smoothing

Document priors

Smoothing Probability Estimates

- When estimating probabilities, we tend to ...
  - Over-estimate the probability of observed outcomes
  - Under-estimate the probability of unobserved outcomes
- The goal of smoothing is to ...
  - Decrease the probability of observed outcomes
  - Increase the probability of unobserved outcomes
- It's usually a good idea
- You probably already know this concept!

Smoothing Probability Estimates

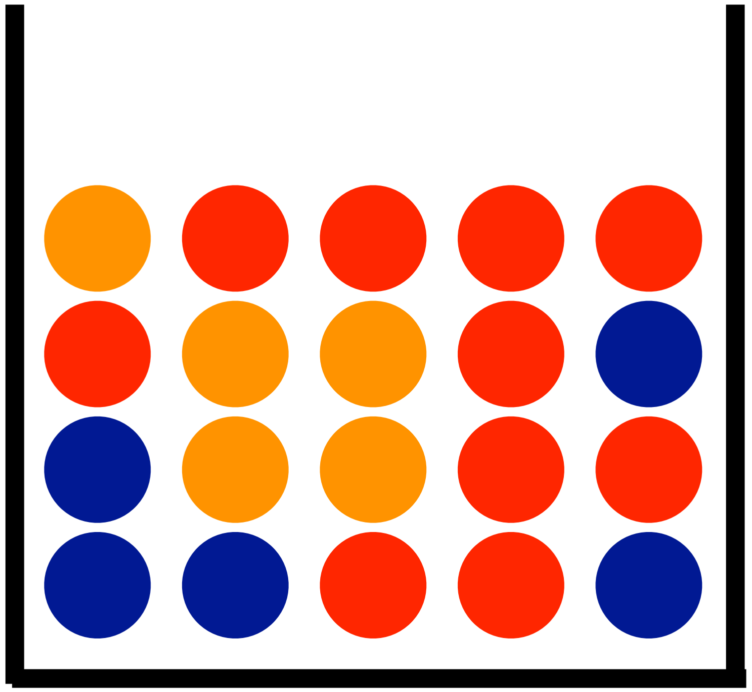
- YOU: Are there mountain lions around here?
- YOUR FRIEND: Nope.
- YOU: How can you be so sure?
- YOUR FRIEND: Because I've been hiking here five times before and have never seen one.
- YOU: ????

Smoothing Probability Estimates

- YOU: Are there mountain lions around here?
- YOUR FRIEND: Nope.
- YOU: How can you be so sure?
- YOUR FRIEND: Because I've been hiking here five times before and have never seen one.
- MOUNTAIN LION: You should have learned about smoothing by taking INLS 509. Yum!

Smoothing Probability Estimates

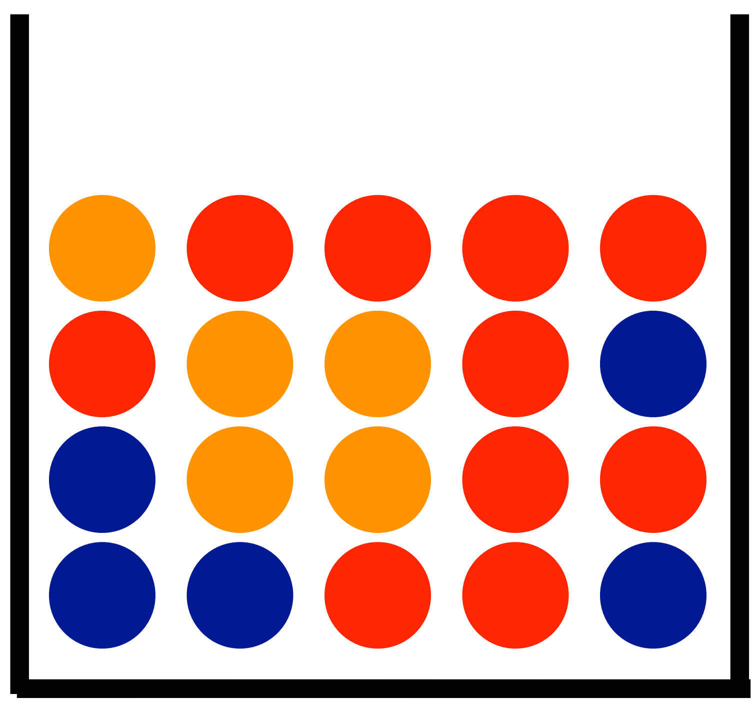
- Suppose that in reality this bag is a sample from a different, bigger bag …
- And, our goal is to estimate the probabilities of <u>that</u> bigger bag …
- And, we know that the bigger bag has red, blue, orange, yellow, and green balls.



P(RED) = 0.5
P(BLUE) = 0.25
P(ORANGE) = 0.25
P(YELLOW) = 0.00
P(GREEN) = 0.00

Smoothing Probability Estimates

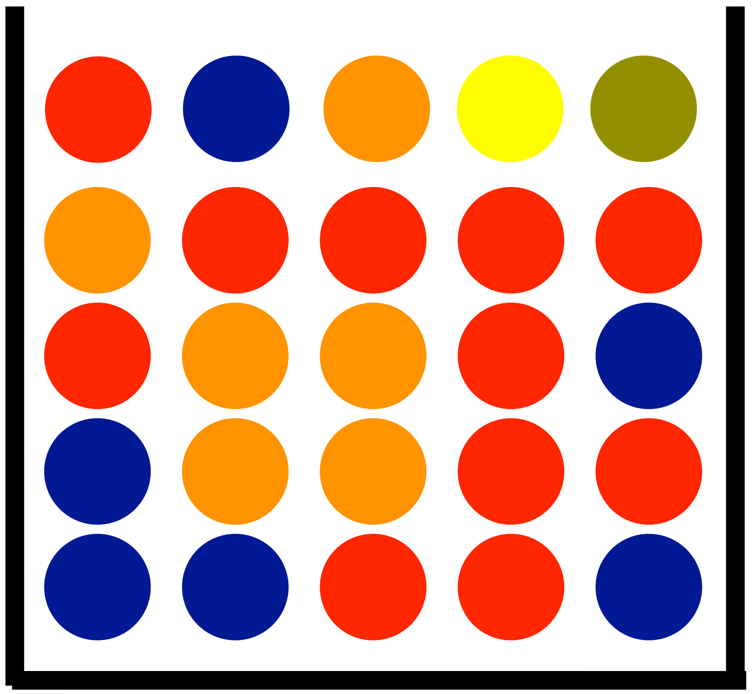
- Do we really want to assign YELLOW and GREEN balls a zero probability?
- What else can we do?



P(RED) = (10/20)
P(BLUE) = (5/20)
P(ORANGE) = (5/20)
P(YELLOW) = (0/20)
P(GREEN) = (0/20)

Add-One Smoothing

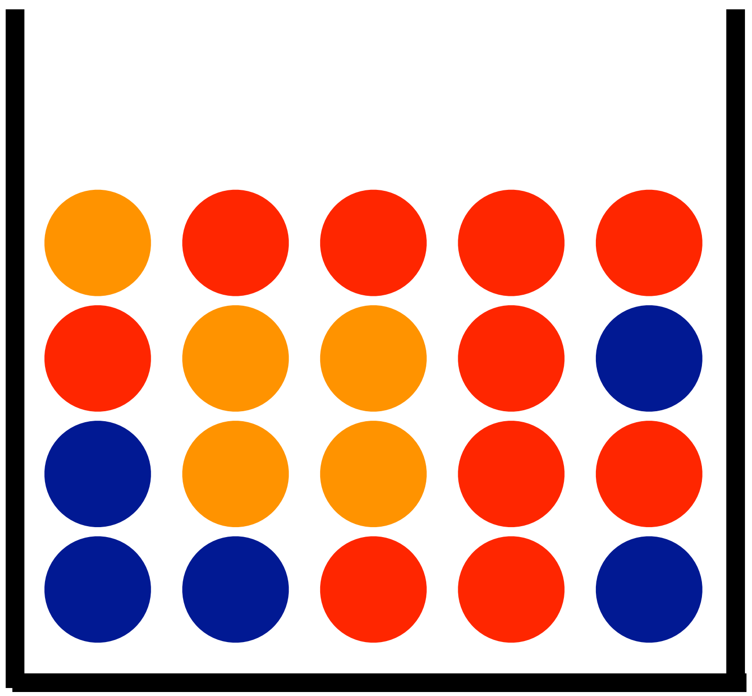
- We could add one ball of each color to the bag
- This gives a small probability to unobserved outcomes (YELLOW and GREEN)
- As a result, it also reduces the probability of observed outcomes (RED, BLUE, ORANGE) by a small amount
- Very common solution (also called ‘discounting’)



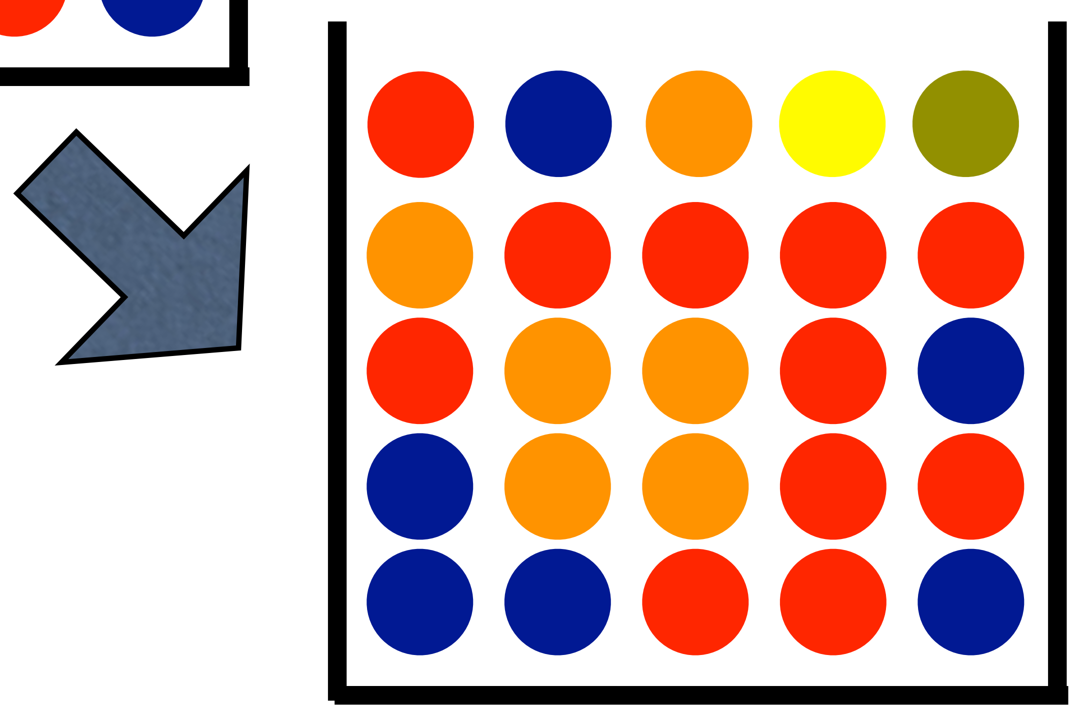
P(RED) = (11/25)
P(BLUE) = (6/25)
P(ORANGE) = (6/25)
P(YELLOW) = (1/25)
P(GREEN) = (1/25)

Add-One Smoothing

- Gives a small probability to unobserved outcomes (YELLOW and GREEN) and reduces the probability of observed outcomes (RED, BLUE, ORANGE) by a small amount



P(RED) = (10/20)
P(BLUE) = (5/20)
P(ORANGE) = (5/20)
P(YELLOW) = (0/20)
P(GREEN) = (0/20)



P(RED) = (11/25)
P(BLUE) = (6/25)
P(ORANGE) = (6/25)
P(YELLOW) = (1/25)
P(GREEN) = (1/25)

Smoothing Probability Estimates
for document language models

- In theory, we could use add-one smoothing
- To do this, we would add each indexed-term once into each document
    - Conceptually!
- Then, we would compute its language model probabilities
- In practice, a more effective approach to smoothing for information retrieval is called linear interpolation

Linear Interpolation Smoothing

- Let θ_D denote the language model associated with document ***D***
- Let θ_C denote the language model associated with the entire collection
- Using linear interpolation, the probability given by the document language model to term ***t*** is:

$$P(t|D) = \alpha P(t|\theta_D) + (1 - \alpha)P(t|\theta_C)$$

Linear Interpolation Smoothing

$$P(t|D) = \alpha \underline{P(t|\theta_D)} + (1-\alpha)\underline{P(t|\theta_C)}$$

the probability given to the term by the document language model

the probability given to the term by the collection language model

Linear Interpolation Smoothing

$$P(t|D) = \underline{\alpha}\,\underline{P(t|\theta_D)} + \underline{(1-\alpha)}\,\underline{P(t|\theta_C)}$$

every one of these numbers
is between 0 and 1, so $P(t|D)$
is between 0 and 1

Query Likelihood Retrieval Model
with linear interpolation smoothing

- As before, a document's score is given by the probability that it "generated" the query
- As before, this is given by multiplying the individual query-term probabilities
- However, the probabilities are obtained using the linearly interpolated language model

$$score(Q, D) = \prod_{i=1}^{n} (\alpha P(q_i|\theta_D) + (1-\alpha)P(q_i|\theta_C))$$

Query Likelihood Retrieval Model
with linear interpolation smoothing

- Linear interpolation helps us avoid zero-probabilities
- Remember, because we're multiplying probabilities, if a document is missing a single query-term it will be given a score of zero!
- Linear interpolation smoothing has another added benefit, though it's not obvious
- Let's start with an example

Query Likelihood Retrieval Model

no smoothing

- Query: apple ipad
- Two documents (D_1 and D_2), each with 50 term occurrences

	D_1 (N_{D1}=50)	D_2 (N_{D2}=50)
apple	2/50 = 0.04	3/50 = 0.06
ipad	3/50 = 0.06	2/50 = 0.04
score	(0.04 x 0.06) = 0.0024	(0.06 x 0.04) = 0.0024

Query Likelihood Retrieval Model

no smoothing

- Query: apple ipad
- Two documents (D_1 and D_2), each with 50 term occurrences

	D_1 (N_{D1}=50)	D_2 (N_{D2}=50)
apple	2/50 = 0.04	3/50 = 0.06
ipad	3/50 = 0.06	2/50 = 0.04
score	(0.04 x 0.06) = 0.0024	(0.06 x 0.04) = 0.0024

- Which query-term is more important: apple or ipad?

Query Likelihood Retrieval Model

no smoothing

- A term is descriptive of the document if it occurs many times in the document
- But, not if it occurs many times in the document <u>and</u> also occurs frequently in the collection

Query Likelihood Retrieval Model

no smoothing

- Query: apple ipad
- Two documents (D_1 and D_2), each with 50 term occurrences

	D_1 (N_{D1}=50)	D_2 (N_{D2}=50)
apple	2/50 = 0.04	3/50 = 0.06
ipad	3/50 = 0.06	2/50 = 0.04
score	(0.04 x 0.06) = 0.0024	(0.06 x 0.04) = 0.0024

- Without smoothing, the query-likelihood model ignores how frequently the term occurs in general!

Query Likelihood Retrieval Model
with linear interpolation smoothing

- Suppose the corpus has 1,000,000 term-occurrences
- apple occurs 200 / 1,000,000 times
- ipad occurs 100 / 1,000,000 times
- Therefore:

$$P(\text{apple}|\theta_C) = \frac{200}{1000000} = 0.0002$$

$$P(\text{ipad}|\theta_C) = \frac{100}{1000000} = 0.0001$$

Query Likelihood Retrieval Model

with linear interpolation smoothing

$$score(Q, D) = \prod_{i=1}^{n} (\alpha P(q_i|\theta_D) + (1 - \alpha) P(q_i|\theta_C))$$

	D_1 $(N_{D1}$=50)	D_2 $(N_{D2}$=50)
P(apple|D)	0.04	0.06
P(apple|C)	0.0002	0.0002
score(apple)	0.0201	0.0301
P(ipad|D)	0.06	0.04
P(ipad|C)	0.0001	0.0001
score(ipad)	0.03005	0.02005
total score	0.000604005	0.000603505

$$\alpha = 0.50$$

Query Likelihood Retrieval Model
with linear interpolation smoothing

- Linear interpolation smoothing does not only avoid zero probabilities ...
- It <u>also</u> introduces an IDF-like scoring of documents
  - terms that are less frequent in the entire collection have a higher contribution to a document's score
- Yes, but we've only seen an example. Where is the mathematical proof!?

Query Likelihood Retrieval Model

with linear interpolation smoothing

$$
\begin{aligned}
p(q \mid d) &= \prod_{q_i \in q} p(q_i \mid d) \\
&= \prod_{q_i \in q} \left(\lambda p_{MLE}(q_i \mid d) + (1-\lambda) p_{MLE}(q_i \mid C)\right) && \textbf{Mixture model} \\
&= \prod_{q_i \in q} \left(\lambda p_{MLE}(q_i \mid d) + (1-\lambda) p_{MLE}(q_i \mid C)\right) \frac{(1-\lambda) p_{MLE}(q_i \mid C)}{(1-\lambda) p_{MLE}(q_i \mid C)} && \textbf{Multiply by 1} \\
&= \prod_{q_i \in q} \left(\left(\frac{\lambda p_{MLE}(q_i \mid d)}{(1-\lambda) p_{MLE}(q_i \mid C)} + 1 \right) (1-\lambda) p_{MLE}(q_i \mid C) \right) && \textbf{Recombine} \\
&= \prod_{q_i \in q} \left(\frac{\lambda p_{MLE}(q_i \mid d)}{(1-\lambda) p_{MLE}(q_i \mid C)} + 1 \right) \prod_{q_i \in q} (1-\lambda) p_{MLE}(q_i \mid C) && \textbf{Recombine} \\
&\propto \prod_{q_i \in q} \left(\frac{\lambda p_{MLE}(q_i \mid d)}{(1-\lambda) p_{MLE}(q_i \mid C)} + 1 \right) && \textbf{Drop constant}
\end{aligned}
$$

“tf”

“idf”

(slide courtesy of Jamie Callan)

Linearly Interpolated Smoothing

Review

- Doc 1: haikus are easy
- Doc 2: but sometimes they don't make sense
- Doc 3: refrigerator
- Query: haikus make sense

$$score(Q, D) = \prod_{i=1}^{n} (\lambda P(q_i|D) + (1-\lambda)P(q_i|C))$$

(source: threadless t-shirt)