Probability theory

Prof.dr.hab. Viorel Bostan

Technical University of Moldova

viorel.bostan@adm.utm.md

Post Lecture 5

There is a magician and an assistant.

Magic trick

There is a magician and an assistant.

The assistant goes into audience with a deck of 52 cards and asks the audience to select 5 cards without magician seeing them.

Magic trick

There is a magician and an assistant.

The assistant goes into audience with a deck of 52 cards and asks the audience to select 5 cards without magician seeing them.

The assistant then shows four cards to the magician, one by one, and then magician correctly names the fifth secret card.

Magic trick

There is a magician and an assistant.

The assistant goes into audience with a deck of 52 cards and asks the audience to select 5 cards without magician seeing them.

The assistant then shows four cards to the magician, one by one, and then magician correctly names the fifth secret card.

How this is done?

Magic trick

There is a magician and an assistant.

The assistant goes into audience with a deck of 52 cards and asks the audience to select 5 cards without magician seeing them.

The assistant then shows four cards to the magician, one by one, and then magician correctly names the fifth secret card.

How this is done?

Clearly, the assistant somehow communicates to the magician the fifth card.

Magic trick

There is a magician and an assistant.

The assistant goes into audience with a deck of 52 cards and asks the audience to select 5 cards without magician seeing them.

The assistant then shows four cards to the magician, one by one, and then magician correctly names the fifth secret card.

How this is done?

Clearly, the assistant somehow communicates to the magician the fifth card.

Let's identify first the ways the assistant could communicate with the magician.

There is a magician and an assistant.

The assistant goes into audience with a deck of 52 cards and asks the audience to select 5 cards without magician seeing them.

The assistant then shows four cards to the magician, one by one, and then magician correctly names the fifth secret card.

How this is done?

Clearly, the assistant somehow communicates to the magician the fifth card.

Let's identify first the ways the assistant could communicate with the magician.

Assistant can announce the four cards in any order.

There is a magician and an assistant.

The assistant goes into audience with a deck of 52 cards and asks the audience to select 5 cards without magician seeing them.

The assistant then shows four cards to the magician, one by one, and then magician correctly names the fifth secret card.

How this is done?

Clearly, the assistant somehow communicates to the magician the fifth card.

Let's identify first the ways the assistant could communicate with the magician.

Assistant can announce the four cards in any order.

The number of orderings of four cards is $4! = 24$. This alone is not sufficient to announce the remaining card since there are 48 cards left.

Magic trick

There is a magician and an assistant.

The assistant goes into audience with a deck of 52 cards and asks the audience to select 5 cards without magician seeing them.

The assistant then shows four cards to the magician, one by one, and then magician correctly names the fifth secret card.

How this is done?

Clearly, the assistant somehow communicates to the magician the fifth card.

Let's identify first the ways the assistant could communicate with the magician.

Assistant can announce the four cards in any order.

The number of orderings of four cards is $4! = 24$. This alone is not sufficient to announce the remaining card since there are 48 cards left.

The assistant can also choose which four cards to reveal in $\binom{5}{4} = 5$ different ways.Still not sufficient since the magician does not know which of these 5 possibilities the assistant selected because of the secret card.

Trick:

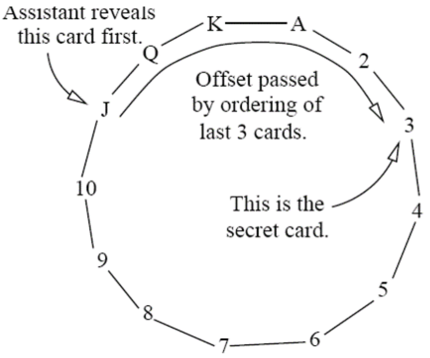
Magic trick revealed

Trick: The secret card has the same suit as the first card!

Magic trick revealed

Trick: The secret card has the same suit as the first card!

The value of the secret card is offset from the value of the first card announced by the value of the remaining three cards:



Offset 5 has to be communicated.

How offset can be communicated?

How offset can be communicated? Magician will see the remaining three cards. A number between 1 and 6 will be the offset.

Magic trick revealed

How offset can be communicated? Magician will see the remaining three cards. A

number between 1 and 6 will be the offset.

An order of all 52 cards is agreed in advance, for example,

$$A\clubsuit, 2\clubsuit, 3\clubsuit, \ldots, K\clubsuit, A\diamondsuit, 2\diamondsuit, \ldots, K\diamondsuit, \ldots K\spadesuit$$

How offset can be communicated? Magician will see the remaining three cards. A

number between 1 and 6 will be the offset.

An order of all 52 cards is agreed in advance, for example,

$$A\clubsuit, 2\clubsuit, 3\clubsuit, \ldots, K\clubsuit, A\diamondsuit, 2\diamondsuit, \ldots, K\diamondsuit, \ldots K\spadesuit$$

With this order, one of the cards will be the smallest (S), the next one will be the middle (M), and the third card will be the largest (L).

Magic trick revealed

How offset can be communicated? Magician will see the remaining three cards. A

number between 1 and 6 will be the offset.

An order of all 52 cards is agreed in advance, for example,

$$A\clubsuit, 2\clubsuit, 3\clubsuit, \ldots, K\clubsuit, A\diamondsuit, 2\diamondsuit, \ldots, K\diamondsuit, \ldots K\spadesuit$$

With this order, one of the cards will be the smallest (S), the next one will be the middle (M), and the third card will be the largest (L).

The offset is encoded by the order of these three cards,

$$\begin{aligned} SML &= 1, \quad SLM = 2, \quad MSL = 3 \\ MLS &= 4, \quad LSM = 5, \quad LMS = 6 \end{aligned}$$

Magic trick revealed

For example, audience selects:

$$3\heartsuit, 8\diamondsuit, A\spadesuit, J\heartsuit, 6\diamondsuit$$

Magic trick revealed

For example, audience selects:

$$3\heartsuit, 8\diamondsuit, A\spadesuit, J\heartsuit, 6\diamondsuit$$

Assistant picks the two cards with the same suit: $3\heartsuit$ and $J\heartsuit$.

For example, audience selects:

$$3\heartsuit, 8\diamondsuit, A\spadesuit, J\heartsuit, 6\diamondsuit$$

Assistant picks the two cards with the same suit: $3\heartsuit$ and $J\heartsuit$.

The $3\heartsuit$ is offset 5 clockwise from $J\heartsuit$. So the first shown card is $J\heartsuit$.

Magic trick revealed

For example, audience selects:

$$3\heartsuit, 8\diamondsuit, A\spadesuit, J\heartsuit, 6\diamondsuit$$

Assistant picks the two cards with the same suit: $3\heartsuit$ and $J\heartsuit$.

The $3\heartsuit$ is offset 5 clockwise from $J\heartsuit$. So the first shown card is $J\heartsuit$.

Magician knows that the secret card has suit $\heartsuit$. Offset 5 should be communicated.

Magic trick revealed

For example, audience selects:

$$3\heartsuit, 8\diamondsuit, A\spadesuit, J\heartsuit, 6\diamondsuit$$

Assistant picks the two cards with the same suit: $3\heartsuit$ and $J\heartsuit$.

The $3\heartsuit$ is offset 5 clockwise from $J\heartsuit$. So the first shown card is $J\heartsuit$.

Magician knows that the secret card has suit $\heartsuit$. Offset 5 should be communicated.

The remaining cards shown are $A\spadesuit, 6\diamondsuit, 8\diamondsuit$. Meaning order $LSM = 5$.

Magic trick

Suppose that the audience selects only 4 cards.

Magic trick

Suppose that the audience selects only 4 cards.

Can the trick be pulled of? In other words, if three cards are revealed, can magician guess the fourth card?

Suppose that the audience selects only 4 cards.

Can the trick be pulled of? In other words, if three cards are revealed, can magician guess the fourth card?

Let X be the set of four cards that the audience might select and let Y be the set of all sequences of three cards that the assistant can reveal.

Magic trick

Suppose that the audience selects only 4 cards.

Can the trick be pulled of? In other words, if three cards are revealed, can magician guess the fourth card?

Let X be the set of four cards that the audience might select and let Y be the set of all sequences of three cards that the assistant can reveal.

Then,

$$|X| = \binom{52}{4} = 270\,725, \qquad |Y| = 52 \cdot 51 \cdot 50 = 132\,600.$$

Clearly, $|X| > |Y|$ and by the Pigeonhole Principle, the assistant must reveal the same sequence of three cards for some two different sets of four!

Suppose that the audience selects only 4 cards.

Can the trick be pulled of? In other words, if three cards are revealed, can magician guess the fourth card?

Let X be the set of four cards that the audience might select and let Y be the set of all sequences of three cards that the assistant can reveal.

Then,

$$|X| = \binom{52}{4} = 270\,725, \qquad |Y| = 52 \cdot 51 \cdot 50 = 132\,600.$$

Clearly, $|X| > |Y|$ and by the Pigeonhole Principle, the assistant must reveal the same sequence of three cards for some two different sets of four!

In other words, if magician hears a set of three cards, then there are at least two possibilities for the fourth card which he cannot distinguish.

Data compression

Stars and bars help us to solve a data compression problem:

Problem

How many bits are needed to store a **multiset** *of n arbitrary integers in the range $[0, 2n]$?*

For example, how much disk space is necessary to store one million numbers in the range from zero to two million?

Stars and bars help us to solve a data compression problem:

Problem

How many bits are needed to store a **multiset** *of n arbitrary integers in the range* $[0, 2n]$?

For example, how much disk space is necessary to store one million numbers in the range from zero to two million?

A simple (straightforward) scheme: Store each number in binary.

$$(101011)_2 = 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0.$$

Stars and bars help us to solve a data compression problem:

Problem

How many bits are needed to store a **multiset** *of n arbitrary integers in the range $[0, 2n]$?*

For example, how much disk space is necessary to store one million numbers in the range from zero to two million?

A simple (straightforward) scheme: Store each number in binary.

$$(101011)_2 = 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0.$$

The biggest binary number with m digits is

$$\underbrace{(111 \ldots 1)}_{m \text{ digits}} = 1 \cdot 2^{m-1} + 1 \cdot 2^{m-2} + \ldots + 1 \cdot 2^1 + 1 \cdot 2^0 = 2^m - 1.$$

Data compression

By storing m binary digits, the biggest decimal number that can be stored is $N = 2^m - 1$.

Data compression

By storing m binary digits, the biggest decimal number that can be stored is $N = 2^m - 1$.

Vice versa, if want to store decimal number N, then will need at most $m = \log_2(N + 1)$ binary digits.

Data compression

By storing m binary digits, the biggest decimal number that can be stored is $N = 2^m - 1$.

Vice versa, if want to store decimal number N, then will need at most $m = \log_2(N+1)$ binary digits.

Numbers to be stored are in the range $[0, 2n]$.

Data compression

By storing m binary digits, the biggest decimal number that can be stored is $N = 2^m - 1$.

Vice versa, if want to store decimal number N, then will need at most $m = \log_2(N + 1)$ binary digits.

Numbers to be stored are in the range $[0, 2n]$.

Storing each number this way will require $\log_2(2n + 1)$ bits.

By storing m binary digits, the biggest decimal number that can be stored is $N = 2^m - 1$.

Vice versa, if want to store decimal number N, then will need at most $m = \log_2(N+1)$ binary digits.

Numbers to be stored are in the range $[0, 2n]$.

Storing each number this way will require $\log_2(2n+1)$ bits.

And storing n numbers will require $n \log_2(2n+1)$ bits.

Data compression

By storing m binary digits, the biggest decimal number that can be stored is $N = 2^m - 1$.

Vice versa, if want to store decimal number N, then will need at most $m = \log_2(N+1)$ binary digits.

Numbers to be stored are in the range $[0, 2n]$.

Storing each number this way will require $\log_2(2n+1)$ bits.

And storing n numbers will require $n \log_2(2n+1)$ bits.

For example, 10^6 numbers in the range from zero to $2 \cdot 10^6$ will need

$$10^6 \cdot \log_2(2 \cdot 10^6 + 1) \approx 21 \cdot 10^6 \text{ bits!}$$

Data compression

By storing m binary digits, the biggest decimal number that can be stored is $N = 2^m - 1$.

Vice versa, if want to store decimal number N, then will need at most $m = \log_2(N+1)$ binary digits.

Numbers to be stored are in the range $[0, 2n]$.

Storing each number this way will require $\log_2(2n+1)$ bits.

And storing n numbers will require $n \log_2(2n+1)$ bits.

For example, 10^6 numbers in the range from zero to $2 \cdot 10^6$ will need

$$10^6 \cdot \log_2(2 \cdot 10^6 + 1) \approx 21 \cdot 10^6 \text{ bits!}$$

Better scheme: There is a much clever scheme that uses stars and bars.

Data compression

By storing m binary digits, the biggest decimal number that can be stored is $N = 2^m - 1$.

Vice versa, if want to store decimal number N, then will need at most $m = \log_2(N+1)$ binary digits.

Numbers to be stored are in the range $[0, 2n]$.

Storing each number this way will require $\log_2(2n+1)$ bits.

And storing n numbers will require $n \log_2(2n+1)$ bits.

For example, 10^6 numbers in the range from zero to $2 \cdot 10^6$ will need

$$10^6 \cdot \log_2(2 \cdot 10^6 + 1) \approx 21 \cdot 10^6 \text{ bits!}$$

Better scheme: There is a much clever scheme that uses stars and bars.

Regard the multiset of numbers to be stored as an n-combination with repetition of the $(2n+1)$-element set $\{0, 1, 2, 3, \ldots, 2000000\}$.

Data compression

By storing m binary digits, the biggest decimal number that can be stored is $N = 2^m - 1$.

Vice versa, if want to store decimal number N, then will need at most $m = \log_2(N + 1)$ binary digits.

Numbers to be stored are in the range $[0, 2n]$.

Storing each number this way will require $\log_2(2n + 1)$ bits.

And storing n numbers will require $n \log_2(2n + 1)$ bits.

For example, 10^6 numbers in the range from zero to $2 \cdot 10^6$ will need

$$10^6 \cdot \log_2(2 \cdot 10^6 + 1) \approx 21 \cdot 10^6 \text{ bits!}$$

Better scheme: There is a much clever scheme that uses stars and bars.

Regard the multiset of numbers to be stored as an n-combination with repetition of the $(2n + 1)$-element set $\{0, 1, 2, 3, \ldots, 2000000\}$.

The previous arguments show that such $n-$combinations with repetition corresponds to sequences of n stars and $2n$ bars.

Data compression

The previous arguments show that such $n-$combinations with repetition corresponds to sequences of n stars and $2n$ bars.

If represent a star with a 0 and a bar with 1, then can store such a star-bar sequence using $n + 2n = 3n$ bits.

The previous arguments show that such $n-$combinations with repetition corresponds to sequences of n stars and $2n$ bars.

If represent a star with a 0 and a bar with 1, then can store such a star-bar sequence using $n + 2n = 3n$ bits.

To make the scheme clear, consider the example of storing 5 integer numbers in the range $[0, 10]$.

The previous arguments show that such $n-$combinations with repetition corresponds to sequences of n stars and $2n$ bars.

If represent a star with a 0 and a bar with 1, then can store such a star-bar sequence using $n + 2n = 3n$ bits.

To make the scheme clear, consider the example of storing 5 integer numbers in the range $[0, 10]$.

In particular, want to store the multiset $\{2, 4, 4, 6, 7\}$.

The previous arguments show that such $n-$combinations with repetition corresponds to sequences of n stars and $2n$ bars.

If represent a star with a 0 and a bar with 1, then can store such a star-bar sequence using $n + 2n = 3n$ bits.

To make the scheme clear, consider the example of storing 5 integer numbers in the range $[0, 10]$.

In particular, want to store the multiset $\{2, 4, 4, 6, 7\}$.

Represent this multiset as a stars and bars sequence (and consequently with $3 \cdot 5 = 15$ bits) as follows:

$$\begin{aligned} \{2,4,4,6,7\} \quad & \rightarrow \,|\,|*|\,|**|\,|*|*|\,|\,| \\ & \rightarrow 110110011010111 \end{aligned}$$

The previous arguments show that such $n-$combinations with repetition corresponds to sequences of n stars and $2n$ bars.

If represent a star with a 0 and a bar with 1, then can store such a star-bar sequence using $n + 2n = 3n$ bits.

To make the scheme clear, consider the example of storing 5 integer numbers in the range $[0, 10]$.

In particular, want to store the multiset $\{2, 4, 4, 6, 7\}$.

Represent this multiset as a stars and bars sequence (and consequently with $3 \cdot 5 = 15$ bits) as follows:

$$\begin{aligned} \{2, 4, 4, 6, 7\} \quad &\rightarrow | \, | * | \, | * * | \, | * | * | \, | \, | \\ &\rightarrow 110110011010111 \end{aligned}$$

Could store 10^6 numbers in the range $\left[0, 2 \cdot 10^6\right]$ in only $3 \cdot 10^6$ bits.

Data compression

Let's compare both schemes. In the first scheme, $21 \cdot 10^6$ bits are used, while in the second scheme need only $3 \cdot 10^6$ bits. An improvement factor of 7 .

Data compression

Let's compare both schemes. In the first scheme, $21 \cdot 10^6$ bits are used, while in the second scheme need only $3 \cdot 10^6$ bits. An improvement factor of 7 .

It is surprising that only 3 bits per number on average are necessary, no matter how large n becomes.

Data compression

Let's compare both schemes. In the first scheme, $21 \cdot 10^6$ bits are used, while in the second scheme need only $3 \cdot 10^6$ bits. An improvement factor of 7 .

It is surprising that only 3 bits per number on average are necessary, no matter how large n becomes.

The stars and bars approach is simple and efficient, but not quite optimal! In any scheme, must associate every multiset of n numbers with a distinct binary sequence with n zeros.

Let's compare both schemes. In the first scheme, $21 \cdot 10^6$ bits are used, while in the second scheme need only $3 \cdot 10^6$ bits. An improvement factor of 7 .

It is surprising that only 3 bits per number on average are necessary, no matter how large n becomes.

The stars and bars approach is simple and efficient, but not quite optimal! In any scheme, must associate every multiset of n numbers with a distinct binary sequence with n zeros.

There are $\binom{3n}{n}$ distinct binary sequences with n zeros, so there are $\binom{3n}{n}$ distinct possible sets of n numbers. So, we must be prepared to store $\binom{3n}{n}$ distinct binary sequences.

Data compression

Let's compare both schemes. In the first scheme, $21 \cdot 10^6$ bits are used, while in the second scheme need only $3 \cdot 10^6$ bits. An improvement factor of 7 .

It is surprising that only 3 bits per number on average are necessary, no matter how large n becomes.

The stars and bars approach is simple and efficient, but not quite optimal! In any scheme, must associate every multiset of n numbers with a distinct binary sequence with n zeros.

There are $\binom{3n}{n}$ distinct binary sequences with n zeros, so there are $\binom{3n}{n}$ distinct possible sets of n numbers. So, we must be prepared to store $\binom{3n}{n}$ distinct binary sequences.

Using k bits, we can store at most 2^k different binary sequences.

Data compression

Let's compare both schemes. In the first scheme, $21 \cdot 10^6$ bits are used, while in the second scheme need only $3 \cdot 10^6$ bits. An improvement factor of 7 .

It is surprising that only 3 bits per number on average are necessary, no matter how large n becomes.

The stars and bars approach is simple and efficient, but not quite optimal! In any scheme, must associate every multiset of n numbers with a distinct binary sequence with n zeros.

There are $\binom{3n}{n}$ distinct binary sequences with n zeros, so there are $\binom{3n}{n}$ distinct possible sets of n numbers. So, we must be prepared to store $\binom{3n}{n}$ distinct binary sequences.

Using k bits, we can store at most 2^k different binary sequences.

This means that we need at least k bits, where $2^k \geq \binom{3n}{n}$.

Let's compare both schemes. In the first scheme, $21 \cdot 10^6$ bits are used, while in the second scheme need only $3 \cdot 10^6$ bits. An improvement factor of 7 .

It is surprising that only 3 bits per number on average are necessary, no matter how large n becomes.

The stars and bars approach is simple and efficient, but not quite optimal! In any scheme, must associate every multiset of n numbers with a distinct binary sequence with n zeros.

There are $\binom{3n}{n}$ distinct binary sequences with n zeros, so there are $\binom{3n}{n}$ distinct possible sets of n numbers. So, we must be prepared to store $\binom{3n}{n}$ distinct binary sequences.

Using k bits, we can store at most 2^k different binary sequences.

This means that we need at least k bits, where $2^k \geq \binom{3n}{n}$.

This inequality is satisfied if $k \geq \log_2 \binom{3n}{n}$. Let's estimate it using Stirling's formula.

Data compression

$$\binom{3n}{n} = \frac{(3n)!}{(2n)!\, n!} \sim \frac{\sqrt{2\pi 3n}\left(\frac{3n}{e}\right)^{3n}}{\sqrt{2\pi 2n}\left(\frac{2n}{e}\right)^{2n}\sqrt{2\pi n}\left(\frac{n}{e}\right)^{n}}$$
$$= \sqrt{\frac{3}{4\pi n}}\left(\frac{27}{4}\right)^{n}.$$

$$\binom{3n}{n} = \frac{(3n)!}{(2n)!\,n!} \sim \frac{\sqrt{2\pi 3n}\left(\frac{3n}{e}\right)^{3n}}{\sqrt{2\pi 2n}\left(\frac{2n}{e}\right)^{2n}\sqrt{2\pi n}\left(\frac{n}{e}\right)^{n}}$$
$$= \sqrt{\tfrac{3}{4\pi n}}\left(\tfrac{27}{4}\right)^{n}.$$

Therefore, the lower bound is

$$\log_2 \binom{3n}{n} \sim \log_2\left(\sqrt{\tfrac{3}{4\pi n}}\left(\tfrac{27}{4}\right)^{n}\right)$$
$$= n \log_2 \tfrac{27}{4} - \tfrac{1}{2}\log_2 \tfrac{4\pi n}{3} \approx 2.755n.$$

Slightly better than $3n$.

$$\binom{3n}{n} = \frac{(3n)!}{(2n)!\,n!} \sim \frac{\sqrt{2\pi 3n}\left(\frac{3n}{e}\right)^{3n}}{\sqrt{2\pi 2n}\left(\frac{2n}{e}\right)^{2n}\sqrt{2\pi n}\left(\frac{n}{e}\right)^{n}}$$
$$= \sqrt{\tfrac{3}{4\pi n}}\left(\tfrac{27}{4}\right)^{n}.$$

Therefore, the lower bound is

$$\log_2 \binom{3n}{n} \sim \log_2\left(\sqrt{\tfrac{3}{4\pi n}}\left(\tfrac{27}{4}\right)^{n}\right)$$
$$= n\log_2 \tfrac{27}{4} - \tfrac{1}{2}\log_2 \tfrac{4\pi n}{3} \approx 2.755n.$$

Slightly better than $3n$.

How to store it in this manner? Order all of the multisets in the range $[0, 2n]$. And then, store the index of multiset, instead of storing the multiset.

$$\binom{3n}{n} = \frac{(3n)!}{(2n)!\, n!} \sim \frac{\sqrt{2\pi 3n}\left(\frac{3n}{e}\right)^{3n}}{\sqrt{2\pi 2n}\left(\frac{2n}{e}\right)^{2n}\sqrt{2\pi n}\left(\frac{n}{e}\right)^{n}}$$
$$= \sqrt{\frac{3}{4\pi n}}\left(\frac{27}{4}\right)^{n}.$$

Therefore, the lower bound is

$$\log_2 \binom{3n}{n} \sim \log_2\left(\sqrt{\frac{3}{4\pi n}}\left(\frac{27}{4}\right)^{n}\right)$$
$$= n\log_2 \frac{27}{4} - \frac{1}{2}\log_2 \frac{4\pi n}{3} \approx 2.755n.$$

Slightly better than $3n$.

How to store it in this manner? Order all of the multisets in the range $[0, 2n]$. And then, store the index of multiset, instead of storing the multiset.

The index is a number from 0 to $\binom{3n}{n} - 1$, which can be stored in exactly $\log_2 \binom{3n}{n}$ bits.