# Information Theory (UAI/500)

## Lecture Notebook

Lecturers: Ivo Bukovský & Ladislav Beránek

**Basic Reading:**

[1] MACKAY, David J. C. Information theory, inference, and learning algorithms. Cambridge: Cambridge University Press, 2003. ISBN 978-0-521-64298-9, https://www.inference.org.uk/itprnn/book.pdf, (Copyright Cambridge University Press 2003. On-screen viewing permitted. Printing not permitted. http://www.cambridge.org/0521642981)

**Basic Viewing**

[2] “Course on Information Theory, Pattern Recognition, and Neural Networks - VideoLectures - VideoLectures.NET.” http://videolectures.net/course_information_theory_pattern_recognition/.

**Recommended Reading:**

[3] COVER, T. M. and Joy A. THOMAS. Elements of information theory. 2nd ed. Hoboken: Wiley-Interscience, c2006. ISBN 978-0-471-24195-9..
[4] HOST, S. Information and Communication Theory. Hoboken, NJ: Wiley-IEEE Press, 2019. ISBN 978-1119433781..
[5] EL-GAMAL, A. and YOUNG-HAN, K. Network information theory. Primera. Cambridge: Cambridge University Press, 2011. ISBN 978-1-107-00873-1..

## UAI/500 Grades and Assesment Conditions

In the end of semester, there will be compulsory test, that you have to passs before exam.

The points can be collected as follows:

a) **max 40 points** ... from optional assignments during semester that you upload to Moodle within 2 weeks from their introduction in the class, this is not compulsory, you do are not obliged to submit your asignments to get assesment. In Jupyter Notebooks, keep looking for tags **[## points]**, these are assignments that you can submit and get points for.

b) **max 70 points** ... for final exam itself

**Alternatively/aditionally to a), b):**
If you prefere to work on a closely related project to the class subjects, it can add or replace

points you. For really good projects (and the report), you might get up to 40 points instead of a) and b).
In advance, make sure, you let your teacher (Ivo or Ladislav) know about the topic of your project for this class and ideally get it confirmed by your teacher in advance - thank you!

**In total, you may collect up to 110 points, and the course grading is to be as follows:**

**Total points collected:**
$90 < \dots$ **A**
$80 \leq 90$ **B**
$70 \leq 80$ **C**
$60 \leq 70$ **D**
$50 \leq 60$ **E**
$\leq 50$ **F**

## Lecture 1

## Fundamentals for Information Theory (probability, information)

- combinatorics, binomial coefficients
- probability, random variable, probability distributions (binomial distribution), probability distribution (discrete random variable) vs. probability density function (continuous random variable)

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

### Minimum to see for this week

- from [1], read about: Example 1.1., Appendix starting on page 598, then read about (A1,A2,A5,A6,A8,C1,C2)
- recap this: “5.3: Mean and Standard Deviation of Binomial Distribution,” Statistics LibreTexts, May 05, 2019. https://stats.libretexts.org/Bookshelves/Introductory_Statistics/Book%3A_Statistics_Using_Technology_(Kozak)/05%3A_Discrete_Probability_Distributions/5.03%3A_Mean_and_Standard_Deviation_of_Binomial_Distribution"
- See et least about first 30 minutes of the legendary Lecture 1: Introduction to Information Theory

```
Start coding or generate with AI.
```

## Combinatorics, Binomial Coefficients

### Permutations

There is $n$ various elements in the set and their order matters.
$n$ ... the number of elements in the set.

$\Psi(n)$ ... the number of permutations (**# of variants of arrangement**).

$\Psi(n) = n!$

Solve, complete, and calculate the results bellow:

- How many various 9-digit numbers we can generate from 9 digits 1,2,3,4,5,6,7,8,9, provided each digit appears only once?

$$n = 9$$
$$\Psi(n) = 9! = 9.8.7...$$

---

- How many valid 9-digit numbers we can generate from 9 digits 0,1,2,3,4,5,6,7,8, provided each digit appears only once?

$$\dots$$

---

```
from math import factorial as fa
N=fa(9)
print("We can create",N,"different 9-digit numbers.")

#task:
8*fa(8)
```

```
We can create 362880 different 9-digit numbers.
322560
```

```
Start coding or generate with AI.
```

### Permutation with Repetition

The set has total of $n$ elements, the order matters, and the elements may appear multiple-times (in $m$ groups of multiple appeareance).
$n$ ... the total length of the set,
$r_i$ ... the size of $i^{th}$ group, i.e., $i^{th}$ element is repeated $r_i$-times,
where $i = 1, \dots, m$ .

$$\Psi_{r_1,r_2,\dots,r_m}(n) = \frac{n!}{r_1! \cdot r_2! \dots \cdot r_m!} = \frac{\dots}{\dots} = \dots$$

Solve, complete, and calculate the results bellow:

- How many 9-digit numbers we can generate from digits 1,1,2,2,2,3,3,3,3?
- How many 9-digit numbers we can generate from digits 1,1,5,5,5,0,0,0,0? **[0.2 points]**

$$\dots$$

$$n = 9$$

$$r_1 = 2, r_2 = 3, r_3 = 4,$$

---

```python
from math import factorial as fa
# 1)How many 9-digit numbers we can generate from digits 1,1,2,2,2,3,3,3,3?
n = 9
r1 = 2
r2 = 3
r3 = 4
N = fa(n)/(fa(r1) * fa(r2) * fa(r3))
print("We can generate",N,"different 9 digit numbers from digits 1,1,2,2,2,3,3,3,3")

# 2) How many 9-digit numbers we can generate from digits 1,1,5,5,5,0,0,0,0?
# Total number of permutations is 1260 for the above given digits. Subtracting the inv
#  which start with zero as 1st number.
n = 8
r1 = 2
r2 = 3
r3 = 3
IN = fa(n)/(fa(r1) * fa(r2) * fa(r3))
N1 = N - IN
print("We can generate",N1,"different 9 digit numbers from digits 1,1,5,5,5,0,0,0,0")
```

```
We can generate 1260.0 different 9 digit numbers from digits 1,1,2,2,2,3,3,3,3
We can generate 700.0 different 9 digit numbers from digits 1,1,5,5,5,0,0,0,0
```

## Variations

i.e., variations without repetition.

There is $n$ of various elements. We make selections of $r$ out of $n$ elements, and selections can vary only in ordering.

Then, the number of possible selections is

$$V_r(n) = \frac{n!}{(n-r)!} = \binom{n}{r} \cdot r! \quad .$$

Solve, complete, and calculate the results bellow:

- How many valid 3-digit numbers, where each digit is unique, can be generated from digits 1,2,3,4,5,6,7,8,9?
- How many valid 3-digit numbers, where each digit is unique, can be generated from digits 0,1,2,3,4,5,6,7,8? **[0.2 Points]**

...

$n = \ \ldots\ ?, \ r = \ \ldots\ ?$

```
# How many valid 3-digit numbers, where each digit is unique, can be generated from di
n = 9
r = 3
N = fa(n)/(fa(n - r))
print("We can generate", N, "valid 3-digit numbers from digits 1,2,3,4,5,6,7,8,9")

# How many valid 3-digit numbers, where each digit is unique, can be generated from di
# 1st digit cannot be zero, therefore we have 8 digits to choose from
# 2nd digit can be zero, therefore we have 8 digits to choose from
# 3rd digit can be zero, therefore we have 7 digits to choose from
n1 = 8
n2 = 8
n3 = 7
N = n1*n2*n3
print("We can generate",N, "different 3 digit numbers from digits 0,1,2,3,4,5,6,7,8")
```

```
We can generate 504.0 valid 3-digit numbers from digits 1,2,3,4,5,6,7,8,9
We can generate 448 different 3 digit numbers from digits 0,1,2,3,4,5,6,7,8
```

## Variations with Repetition

There is $n$ of various elements.

We make selections of $r$ out of $n$ various elements, selections can vary only in ordering, and the elements in selection may repeat.

Then, the number of possible selections is

$V_r(n) = n^r.$

### Solve, complete, and calculate the results bellow:

- How many valid 3-digit numbers can be generated from digits 1,2,3,4,5,6,7,8,9?
- How many valid 3-digit numbers can be generated from digits 0,1,2,3,4,5,6,7,8?**[0.2 Points]**

$n = \ \ldots\ ?, \ r = \ \ldots\ ?$

```
# How many valid 3-digit numbers can be generated from digits 1,2,3,4,5,6,7,8,9?
n = 9
r = 3
N = n**r
print("We can generate",N, "different 3 digit numbers from digits 1,2,3,4,5,6,7,8,9")

""" How many valid 3-digit numbers can be generated from digits 0,1,2,3,4,5,6,7,8?
1st digit cannot be zero, therefore we have 8 digits to choose from
2nd digit can be zero, therefore we have 9 digits to choose from
3rd digit can be zero, therefore we have 9 digits to choose from """
n1 = 8
```

```
n2 = 9
n3 =9
N = n1*n2*n3
print("We can generate",N, "different 3 digit numbers from digits 0,1,2,3,4,5,6,7,8")
```

```
We can generate 729 different 3 digit numbers from digits 1,2,3,4,5,6,7,8,9
We can generate 648 different 3 digit numbers from digits 0,1,2,3,4,5,6,7,8
```

## Combinations

i.e., combinations without repetitions.

There is $n$ of various elements, we make selections of $r$ out of all $n$ elements, i.e. $n$ choose $r$, ordering does not matter, and elements do not repeat.

Then, the number of such possible selections is

$$C_r(n) = \binom{n}{r} = \frac{n!}{r! \cdot (n-r)!}.$$

### Solve, complete, and calculate the results bellow:

In a lotterry bag, there are numbers 0,1,2,...,9. The ticket is for guessing four numbers, and the first price is if you correctly guess all four numbers.

- What is the probability that you win the first price?
- If one ticket costs 1,-Eur, :
  - What is your chance to win 1st price for 20,-Eur? **[0.2 Points]**
  - How much money you have to spend to have 10% chance to win the 1st price? **[0.2 Points]**

$$r = \ \ \ldots \ \ , \ \ n = \ \ 10 \ \ ?$$

```
n = 10 #lottery bag = 0 to 9
r = 4 #ticket = 4 numbers
# Total number of combinations is
N = fa(n)/(fa(r)*fa(n-r))
print("Total number of Tickets is",N)
print("Probability of winning first prize is:",(1/N))
print("Probability of winning 1st prize for 20 euros is:",(20/N)*100,"Percent")
print("Money spent to have 10% chance to win the 1st price:",N*(10/100),"Euros")
```

```
Total number of Tickets is 210.0
Probability of winning first prize is: 0.004761904761904762
Probability of winning 1st prize for 20 euros is: 9.523809523809524 Percent
Money spent to have 10% chance to win the 1st price: 21.0 Euros
```

## Combinations with Repetition

There is $n$ of various elements, we make selections of $r$ out of all $n$ elements, ordering does not matter, and elements may repeat.

Then, the number of such possible selections is

$$\bar{C}_r(n) = \binom{n+r-1}{r}$$

## Solve, complete, and calculate the results bellow:

In the lottery bag, there are numbers 0,1,2,...,9 and the number is returned to the bag after each drawing. The ticket is for guessing four numbers, and the first price is if you correctly guess all four numbers.

- What is the probability that you win the first price?
- If one ticket costs 1,-Eur:
    - What is your chance to win 1st price for 20,-Eur?**[0.2 Points]**
    - How much money you have to spend to have 10% chance to win the 1st price?**[0.2 Points]**

$r = \quad \ldots \quad , \; n = \quad \ldots \; ?$

```
n = 10
r = 4
N = (fa(n+r-1)/(fa(r)*fa(n-1)))
print("Total number of Tickets is",N)
print("Probability of winning first prize is:",(1/N))
print("Probability of winning 1st prize for 20 euros is:",(20/N)*100,"Percent")
print("Money spent to have 10% chance to win the 1st price:",N*(10/100),"Euros")
```

```
Total number of Tickets is 715.0
Probability of winning first prize is: 0.0013986013986013986
Probability of winning 1st prize for 20 euros is: 2.797202797202797 Percent
Money spent to have 10% chance to win the 1st price: 71.5 Euros
```

## Binomial Coefficients

$(a+b)^k = \sum_{i=0}^{k} \binom{k}{i} \cdot a^{k-i} \cdot b^i$ ... e.g.. for binomial distribution (of probability)

$(a-b)^k = \sum_{i=0}^{k} (-1)^i \binom{k}{i} \cdot a^{k-i} \cdot b^i$ ... e.g. for k-th central (mean) moments of a random variable via its moments, but do not worry too much about this formula for now.

Recall $\binom{k}{0} = \binom{k}{k} = 1\,.$

## Binomial Distribution

**probability, probability of independent events, random variable**

## Example L-1

**Tossing a coin**

1. head or tail only:
   - experimentaly estimate probability of head, tale
   - calculate (and validate) probability of 2xhead in two tossings, 3xtail in three tossings
   - probabality
   - out of 10 consequent independent tossings with a single coin, what is probability that you get head only at first toss?
   - when tossing 10 coins at once, what is probability that only one particular coin gets tail?
   - when tossing 10 coins, what is probability that you get 2 heads?
2. head or tail or edge (btw here we touch **probability** vs. **possibility**)
3. example of random variable: $x$ ... how many times you get tail in $N$ independent tossings

```
Start coding or generate with AI.
```

## Example L-2

**Throwing a dice**

1. possible outomes are exclusively number of spots 1,2,3,4,5, or 6:
   - experimentally estimate probability that you get number 6 in a single throw
   - calculate (and validate) probability of two-times number 6 in two tossings, three-times 6 in three throws
   - out of 10 consequent independent tossings with a single dice, what is probability that you get 6 only at first throw?
   - when throwing 10 dices at once, what is probability that only one particular coin gets 6?
   - when throwing 10 coins, what is probability that you gets 6 at least once?
2. example of random variable: $x$ ... how many times you get number 6 in $N$ independent throws

```
Start coding or generate with AI.
```

## Example L-3

(The following example is a very simple one just to introduce the notion of the binomial distribution. The field that seriously deals with system reliability is callled **Reliability Engineering**, and the strong buzzword here is the

**Predictive Maintanance**, of course.)

**There is running $N$ same independent computers. Operating system on all computers is updated at once for all computers. During previous updates, it was observed that in average about $n$ computers out of $N$ had problems that consequently required technical support.**

**Questions:**

$p = \ldots$ ...choose $p$ as the probability that a computer will need maintenance after its update, $p \in \langle 0, 1 \rangle$ , $p << 1$
$N = \ldots$ ...choose the number of all computers.

1. Compared to the coin tossing where also two outcomes can happen(head or tail), what is the main difference from the above computer failure example in terms of probability?
2. What is the probability that no computer will need technical support after update? **[0.1 Points]**
3. What is the probability that all computers will need technical support after update? **[0.1 Points]**
4. What is the probability that only one particular computer will need technical support after update? **[0.1 Points]**
5. What is the probability that one computer (anyone out of all) will need technical support after update? **[0.1 Points]**
6. What is the probability that any two computers will need technical support after update? **[0.1 Points]**
7. What is the probability that any three computers will need technical support after update? **[0.1 Points]**
8. What is variance and standard deviation of this probability distribution? i.e. how many computers is estimated to need the support plus minus? mean ± variance **[0.2 Points]**
9. What is the probability that a particular number of computers will need technical support after the update? Draw the graphs of probability distribution for your chosen value of $p << 1$ and and for $p = 0.5$. **[1.0 Point]**
10. For the achieved probability distributions in previous task 9., calculate numericaly the mean and variance, and compare the values with the values directly calculated by formulas for mean and variance of binomial distribution.**[1.0 Point]**

**Solutions:**

Let's define random variable $x$ as the number of failed computers after the update, where $x \in \langle 0, N \rangle \in \mathbb{W}$, i.e., $x = 0, 1, 2, 3, \ldots, N$ [failed computers]

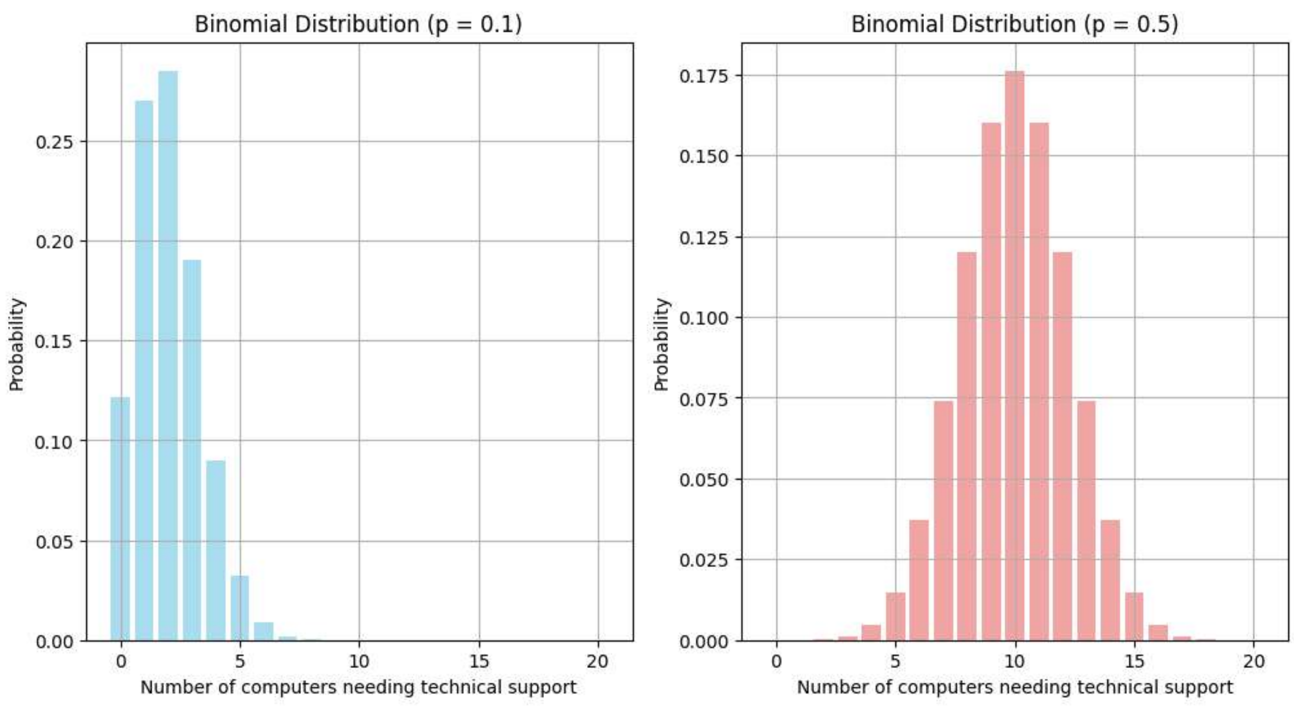
...

```
""" 2) What is the probability that no computer will need technical support after upda
    => (1-p)**N
    3) What is the probability that all computers will need technical support after up
    => p**N
    4) What is the probability that only one particular computer will need technical s
    => p*(1-p)**(N-1)
    5) What is the probability that one computer (anyone out of all) will need technic
    => Np(1-p)**(N-1)
    6) What is the probability that any two computers will need technical support afte
    => (N(N-1)/2)p**2(1-p)**(N-2)
    7) What is the probability that any three computers will need technical support af
    => (N(N-1)(N-2)/6)p**3(1-p)**(N-3)
    8) What is variance and standard deviation of this probability distribution?
    i.e. how many computers is estimated to need the support ?
    => mean = N*p, variance = N*p*(1-p), standard deviation = sqrt(N*p*(1-p)), compute

import math
p = 0.1
N = 20

print("Probability that no computer will need technical support after update is :",(1-
print("probability that all computers will need technical support after update is :",p
print("probability that only one particular computer will need technical support after
print("probability that one computer (anyone out of all) will need technical support a
print("probability that any two computers will need technical support after update is:
print("probability that any three computers will need technical support after update i
print("mean:",N*p)
print("variance:",N*p*(1-p))
print("standard deviation:",math.sqrt(N*p*(1-p)))
print("Computers estimated to need support is:",N*p,"+-",math.sqrt(N*p*(1-p)))
```

```
Probability that no computer will need technical support after update is : 0.12157665
probability that all computers will need technical support after update is : 1.000000
probability that only one particular computer will need technical support after updat
probability that one computer (anyone out of all) will need technical support after u
probability that any two computers will need technical support after update is: 0.285
probability that any three computers will need technical support after update is: 0.1
mean: 2.0
variance: 1.8
standard deviation: 1.3416407864998738
Computers estimated to need support is: 2.0 +- 1.3416407864998738
```

```
"""9)What is the probability that a particular number of computers will need technical
   Draw the graphs of probability distribution for your chosen value of  p<<1  and and
  => probability that a particular number of computers will need technical support aft

import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import binom

# Parameters
N = 20  # Total number of computers
k_values = np.arange(0, N + 1)  # Number of computers needing support (0 to N)

# Small probability (p << 1)
p_small = 0.1
probabilities_small = binom.pmf(k_values, N, p_small)
```

```

# Case 2: Moderate probability (p = 0.5)
p_mod = 0.5
probabilities_mod = binom.pmf(k_values, N, p_mod)

# Plotting the probability distributions
plt.figure(figsize=(12, 6))

# Plot for small p
plt.subplot(1, 2, 1)
plt.bar(k_values, probabilities_small, color='skyblue', alpha=0.7)
plt.title(f'Binomial Distribution (p = {p_small})')
plt.xlabel('Number of computers needing technical support')
plt.ylabel('Probability')
plt.grid(True)

# Plot for moderate p
plt.subplot(1, 2, 2)
plt.bar(k_values, probabilities_mod, color='lightcoral', alpha=0.7)
plt.title(f'Binomial Distribution (p = {p_mod})')
plt.xlabel('Number of computers needing technical support')
plt.ylabel('Probability')
plt.grid(True)

plt.show()

```



```
""" 10) For the achieved probability distributions in previous task 9.,
     calculate numericaly the mean and variance, and compare the values with the val
"""

# For p = 0.1
binom_pmf = binom.pmf(k_values, N, p_small)
print(f"k: {k_values} n: {N} p: {p_small}")
# Step 2: Calculate the mean (E(X))
mean_numerical = np.sum(k_values * binom_pmf)
print(f"Numerical Mean: ",round(mean_numerical,2))

# Step 3: Calculate the variance (E((X - mean)^2))
variance_numerical = np.sum((k_values - mean_numerical)**2 * binom_pmf)
print(f"Numerical variance: ",round(variance_numerical,2))

mean = N*p_small
variance = N*p_small*(1-p_small)

print(f"Formula Mean: ",round(mean,2))
print(f"Formula variance: ",round(variance,2))

# For p = 0.5
binom_pmf = binom.pmf(k_values, N, p_mod)
print(f"k: {k_values} n: {N} p: {p_mod}")
```

```python
# Step 2: Calculate the mean (E(X))
mean_numerical = np.sum(k_values * binom_pmf)
print(f"Numerical Mean: ",round(mean_numerical,2))

# Step 3: Calculate the variance (E((X - mean)^2))
variance_numerical = np.sum((k_values - mean_numerical)**2 * binom_pmf)
print(f"Numerical variance: ",round(variance_numerical,2))

mean = N*p_mod
variance = N*p_mod*(1-p_mod)

print(f"Formula Mean: ",round(mean,2))
print(f"Formula variance: ",round(variance,2))
```

```
k: [ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20] n: 20 p: 0.1
Numerical Mean:  2.0
Numerical variance:  1.8
Formula Mean:  2.0
Formula variance:  1.8
k: [ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20] n: 20 p: 0.5
Numerical Mean:  10.0
Numerical variance:  5.0
Formula Mean:  10.0
Formula variance:  5.0
```

## Points and Assigments (week 1):

If you wish to submit your solutions to collect your optional points, solve the problems (e.g. in this notebook directly ) and upload to Moodle (Notebook + its pdf) within 2 weeks - upload your solution into Moodle by Sunday midnight that is 2nd week since the actual week of the lecture.