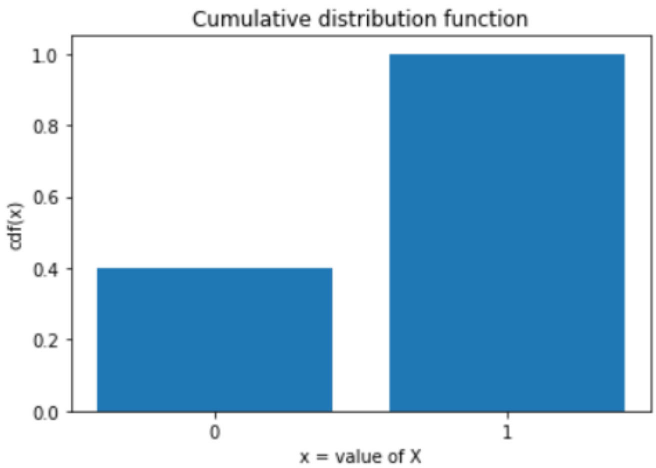
Midterm. Practical exercises

Total: 5p. Minimum: 1.25p

1. (0.3p+0.2p) A fair die is thrown twice. A is the event "sum of the throws equals 4", B is "at least one of the throws is a 3".

a) Calculate P(A|B).
b) Are A and B independent events?

2. (0.2p+0.25p+0.25p) The cumulative distribution function of a discrete random variable is represented below.



If the random variable is called X and P(X = 1) is stored in a code variable called *p*, what does the following code print? Justify your answers!

Code for E1, E2 groups	Code for E3, E4 groups

`import tensorflow_probability as tfp`
`tfd = tfp.distributions`
`X = tfd.Bernoulli(probs=p)`
`print(X.prob(0).numpy())`
`print(X.prob(1).numpy())`
`print(X.mean().numpy())`
`print(X.variance().numpy())`
	`from scipy.stats import bernoulli`
	`print(bernoulli.pmf(0,p))`
	`print(bernoulli.pmf(1,p))`
	`print(bernoulli.mean(p))`
	`print(bernoulli.var(p))`

3. (0.3p) For the dataset below, you are trying to find the variable that is most helpful in predicting Y, according to information gain. Replace variables "a" to "n" with correct values in the code below such that it answers your question.

Code for E1, E2 groups	Code for E3, E4 groups

```
import pandas as pd
data = pd.DataFrame(
     [
        (0,  0,  1),
        (0,  0,  1),
        (1,  1,  1),
        (0,  1,  0),
        (1,  0,  0),
        (0,  1,  0),
        (0,  1,  0),
        (1,  0,  0)
     ], columns=["X1", "X2", "Y"]
)
```

Code for E1, E2 groups:

```
import math
import tensorflow_probability as tfp
tfd = tfp.distributions

def f_entropy(probs):
  X = tfd.Categorical(probs=probs)
  return X.entropy().numpy()/math.log(2)
```

Code for E3, E4 groups:

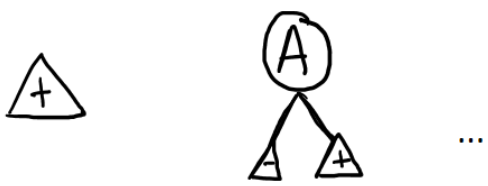
```
from scipy.stats import entropy

def f_entropy(probs_or_counts):
  return entropy(probs_or_counts, base=2)
```

```
H_y = f_entropy([a,  b])
H_y_X1_0 = f_entropy([c, d]) # H(y|X1=0)
H_y_X1_1 = f_entropy([e, f]) # H(y|X1=1)
H_y_X2_0 = f_entropy([g, h]) # H(y|X2=0)
H_y_X2_1 = f_entropy([i, j]) # H(y|X2=1)
IG_X1_y = H_y - k*H_y_X1_0 - l*H_y_X1_1 # IG(y;X1)
IG_X2_y = H_y - m*H_y_X2_0 - n*H_y_X2_1 # IG(y;X2)
print(f"IG(X1;y)={IG_X1_y}")
print(f"IG(X2;y)={IG_X2_y}")
```

4-6. For the exercises from 4 to 6, consider the following code snippet. **Notice that there are TWO print statements at the end on the code.**

```
import pandas as pd
from sklearn import tree, naive_bayes, neighbors

features = ['A', 'B', 'C', 'Y']
data = pd.DataFrame([
   (0, 0, 0, 0),
   (0, 0, 1, 0),
   (0, 1, 0, 0),
   (0, 1, 1, 0),
   (1, 0, 0, 0),
   (1, 0, 1, 1),
   (1, 1, 0, 1)
],
columns=features)

X = data[['A', 'B', 'C']]
y = data['Y']

model = ...
model.fit(X,y)

new_instance =  pd.DataFrame([
     (1, 1, 1)
],
columns=['A', 'B', 'C'])
print(model.predict(new_instance))
print(model.predict_proba(new_instance))
```

4. (0.8p+0.2p) Look at the following image to recall how the depth of a tree is computed.



depth = 0   depth = 1

What does the code above print if the line

`model = ...`

is replaced by

```
model = tree.DecisionTreeClassifier(criterion='entropy',max_depth=1)
```

? Justify your answer! You may use the fact that:

H([0+,n-])=H([n+,0-])=0, H([1+,2-])=H([2+,1-])=0.9182, H([1+,3-])= H([3+,1-])=0.8112.

5. (0.8p+0.2p) Recall that `alpha=1` in `BernoulliNB(alpha=1)` refers to the add-**one** Laplace smoothing rule. What does the code above print if the line

```
model = ...
```

is replaced by

```
model = naive_bayes.BernoulliNB(alpha=1)
```

? Justify your answer!

6. (0.8p+0.2p) What does the code above print if the line

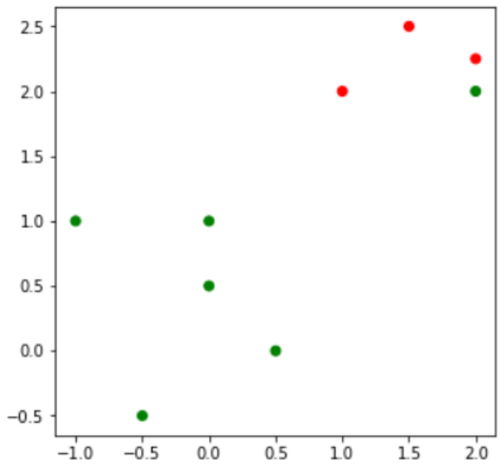
```
model = ...
```

is replaced by

```
model = neighbors.KNeighborsClassifier(n_neighbors=3)
```

? Justify your answer!

7. (0.5p) For the following dataset, which algorithm works best (according to CVLOO)? Is it 1-NN, 3-NN or 5-NN? Justify your answer by calculating the error in each case (no need to create tables for each point)!



Teorie: 5p. Minim: 1.25

Învăţare automată

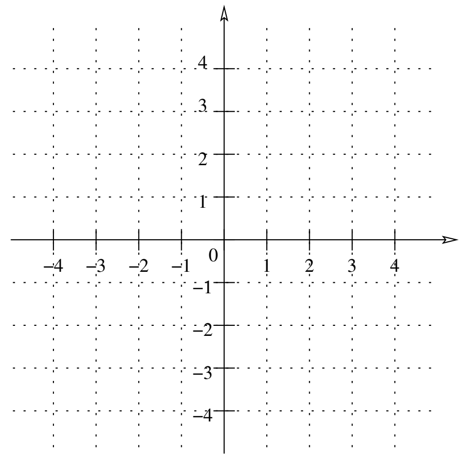
— Licenţă, anul III, 2021-2022, examenul parţial I —

Nume student: **Grupa:**

1. 1.5p **(Algoritmul k-NN, cu aplicare pe date din $\mathbb{R}^2$, respectiv $\mathbb{R}$: desenarea de diagrame Voronoi / identificarea zonelor de decizie; calcularea erorii la CVLOO)**

A. Se dă un set de date de antrenament din planul euclidian ($\mathbb{R}^2$), format din instanţele negative $(-1, 0)$, $(2, 1)$ şi $(2, -2)$ şi instanţele pozitive $(0, 0)$ şi $(1, 0)$.

a. Pe reperul de axe de coordonate ortogonale alăturat marcaţi aceste exemple de antrenament. Veţi folosi *convenţia* noastră de notare: simbolul $\bullet$ desemnează instanţe pozitive $(+)$, iar simbolul $\circ$ instanţe negative $(-)$. Desenaţi apoi graniţele de decizie determinate de algoritmul 1-NN. Veţi haşura zona / zonele de decizie corespunzătoare instanţelor pozitive.



b. Cum va clasifica instanţa de test $(1, -1.01)$ de către algoritmul 1-NN? Dar de către algoritmul 3-NN? Justificaţi, în ambele cazuri.

2. 3.5p **(Un exemplu de clasificator de tip bayesian care combină avantajele algoritmilor Bayes Naiv şi Bayes Optimal)**

Fie A, B şi C variabile aleatoare binare independente, fiecare dintre ele având posibilitatea să ia valoarea 0 cu probabilitate de 50%. Considerăm funcţia

$$Y = ((\neg B) \wedge (\neg C)) \vee (A \wedge B).$$

a. Scrieţi *tabela de adevăr* a funcţiei Y.

b. i. **Folosind tabela de adevăr a funcţiei Y — văzută acum ca un set de date de antrenament pentru un clasificator de tip bayesian — *estimaţi* valorile următoarelor probabilităţi condiţionate:**[1]

$P(A=0 \mid B=0, Y=0) = \ldots$	$P(A=0 \mid C=0, B=0, Y=0) = \ldots$ $P(A=0 \mid C=1, B=0, Y=0) = \ldots$
$P(A=0 \mid B=0, Y=1) = \ldots$	$P(A=0 \mid C=0, B=0, Y=1) = \ldots$ $P(A=0 \mid C=1, B=0, Y=1) = \ldots$
$P(A=0 \mid B=1, Y=0) = \ldots$	$P(A=0 \mid C=0, B=1, Y=0) = \ldots$ $P(A=0 \mid C=1, B=1, Y=0) = \ldots$
$P(A=0 \mid B=1, Y=1) = \ldots$	$P(A=0 \mid C=0, B=1, Y=1) = \ldots$ $P(A=0 \mid C=1, B=1, Y=1) = \ldots$

***Atenţie!* Unele dintre aceste probabilităţi condiţionate s-ar putea să nu fie definite. De *exemplu*, $P(A=0 \mid C=0, B=0, Y=0)$ este nedfinită, fiindcă $P(C=0, B=0, Y=0)$, adică (veţi vedea) nu există în tabelul de date nicio combinaţie de forma $C=0, B=0, Y=0$. În dreptul unor astfel de probabilităţi condiţionate nedfinite veţi pune semnul $*$ *(nedefinit)*.**

ii. **Analizând rezultatele estimărilor obţinute mai sus, se poate deduce o relaţie de tip *independenţă condiţională* între variabilele A, B, C şi Y. Care este această relaţie?**

c. i. **Pe baza rezultatului de la punctul $b.ii$, scrieţi *regula de decizie* pe care o poate folosi un clasificator de tip bayesian — diferit de Bayes Naiv şi Bayes Optimal / Comun; să-i spunem *New-Bayes* — pentru învăţarea conceptului / funcţiei Y, astfel încât**
– să producă eroare 0 pe setul de date [de antrenament] de la punctul a, dar
– să necesite un număr de parametri „liberi“ (engl., free) care trebuie estimaţi mai mic decât cel al algoritmului Bayes Optimal.

ii. ***Justificaţi* în mod riguros regula de decizie pe care aţi scris-o şi că eroarea la antrenare produsă de la punctul a este într-adevăr 0.**

iii. **Care sunt aceşti parametri „liberi“?**

iv. ***Comparaţi* numărul acestor parametri „liberi“ cu numărul minimal de parametri necesari de estimat de către clasificatorii Bayes Naiv şi respectiv Bayes Optimal pe acelaşi set de date.**

[1]Estimarea va fi făcută în mod clasic, adică în sensul verosimilităţii maxime (engl., Maximum Likelihood Estimation, MLE). Nu veţi aplica nicio regulă de netezire a acestor probabilităţi, cum ar fi regula “add-one” a lui Laplace.