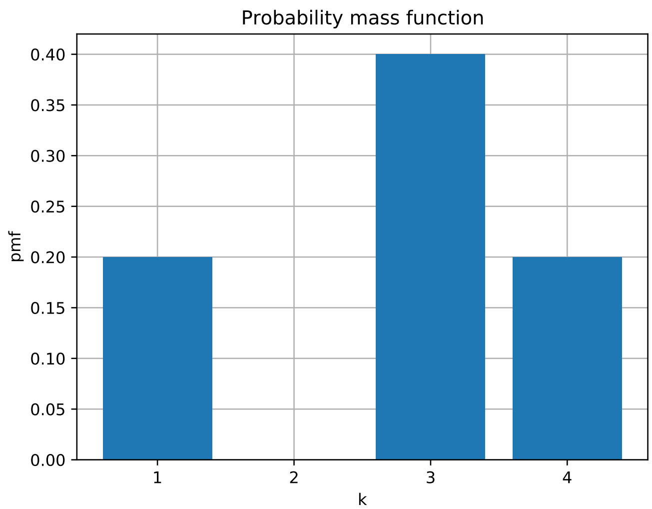
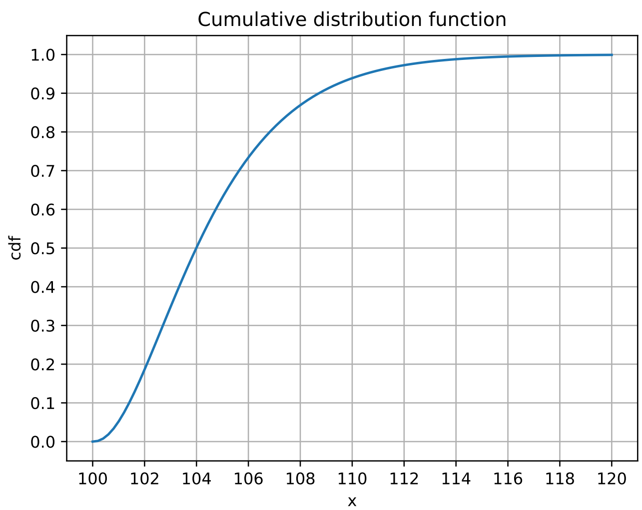
Ex. 1 — Below is an incomplete probability mass function (pmf) chart for the randon variable X.

1. What is the value of the missing probability for $k = 2$?
2. What is the expected value $E(X)$?



Ex. 2 — What is the median value of a random variable with the following cumulative distribution function (cdf)?



Ex. 3 — What is the result of executing the following code?

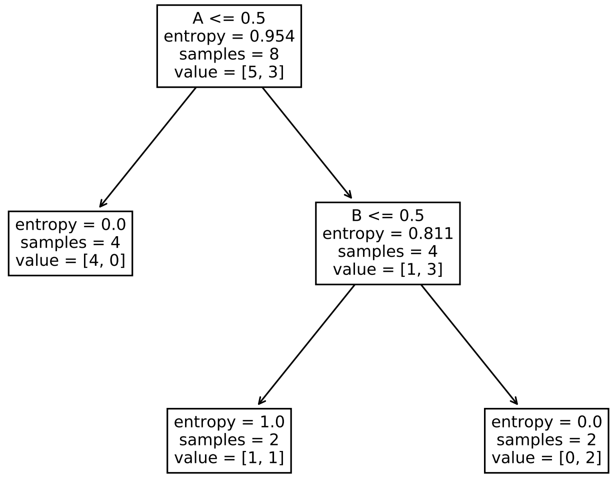
```
from scipy.stats import entropy
entropy([0.25, 0.25, 0.5], base=2)
```

Ex. 4 — Given the followind dataset, the code below traines a pruned decision tree.

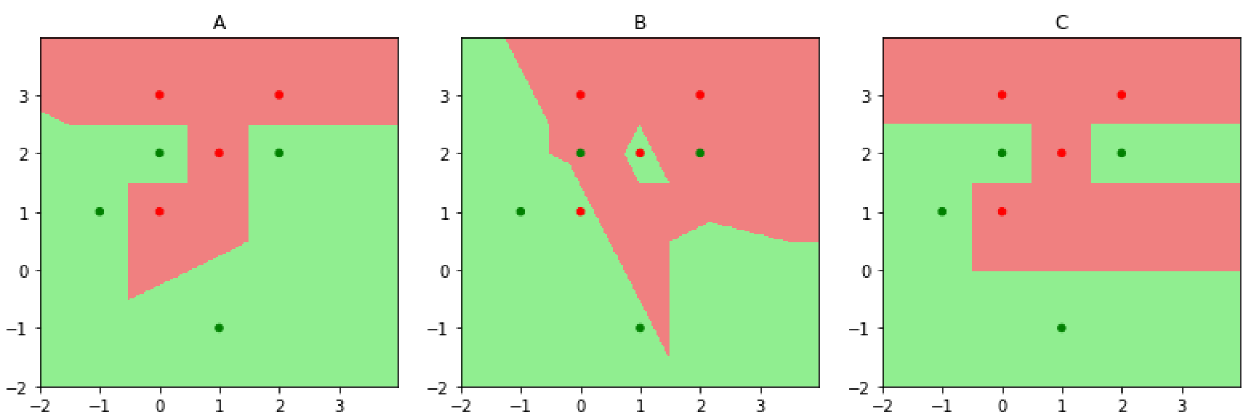
1. How should the code be changed to disable pruning?
2. How will the tree look like with no prunning (draw the full tree exactly as it would be plotted by `plot_tree`).

	A	B	C	Y
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	0
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

```
from sklearn import tree
import matplotlib.pyplot as plt
dt = tree.DecisionTreeClassifier(criterion='entropy',
                                 max_depth=2).fit(X,y)
fig, ax = plt.subplots(1, 1)
f = tree.plot_tree(dt, ax=ax,
                   fontsize=10,
                   feature_names=['A','B','C'])
```



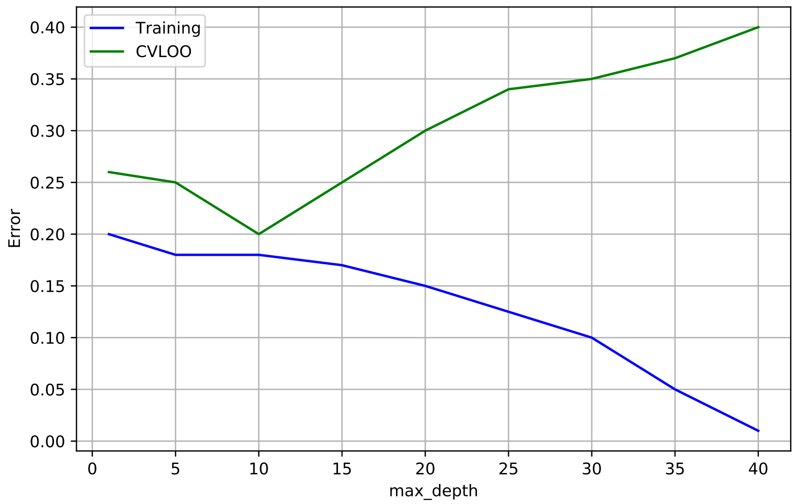
Ex. 5 — Which of the decision surfaces below corresponds to which of the following algorithms: (1) ID3, (2) 1-NN, (3) 3-NN?



Ex. 6 — Given the following graph that shows the training and CVLOO error for a decision tree with various levels of pruning:

1. What is the likely reason for the large difference at `max_depth` = 40?

2. What is the optimum value for `max_depth` if we're aiming for the most accurate predictions?



Ex. 7 — What is the result of executing the following code?

```
import pandas as pd

d = pd.DataFrame({
    'X1': [-1, 0, 1],
    'X2': [0, 1, 0],
    'Y': [0, 1, 2]
})
X, Y = d[['X1', 'X2']], d['Y']
knn = KNeighborsClassifier(n_neighbors=1).fit(X, Y)
knn.predict([[0, 0.5]])
```

Ex. 8 — How does the result of the following code change, if we change the `alpha` parameter to be 1^{-10}?

```
>>> import pandas as pd
>>> from sklearn.naive_bayes import BernoulliNB

>>> # Create the training set
>>> features = ['A', 'B']
>>> target = 'Y'
>>> messages = pd.DataFrame(
... [(1, 0, 0),
...  (0, 0, 1),
```

```
...   (1, 1, 1),
...   (1, 1, 0)],
... columns=features+[target])

>>> # Make the prediction
>>> X = messages[features]
>>> y = messages[target]
>>> cl = BernoulliNB(alpha=1).fit(X, y)
>>> print(cl.predict_proba([[0, 1]]))
[[0.33333333 0.66666667]]
```

9. (1+0.5p) **(Arbori de decizie; optimalitate, ca număr minim de noduri)**

Let us consider the function $Y = A \wedge (B \text{ XOR } C)$, which is defined over the boolean attributes / variables A, B and C.

a. Identify the decision tree(s) having the minimum number of test nodes and is (or, are) *consistent* with (i.e. correspond to) the function Y.

***Attention, please*! You shall use the *exhaustive search algorithm* (not the ID3 algorithm!). When executing this algorithm, *the search space* will be the set of all trees which can be built by using the attributes A, B and C in the internal nodes, and the values of the variable Y in the leaf nodes.**

b. *The version space* for a given concept is defined as the set of all hypotheses (here, the decision trees) which are *consistent* with that concept (in our case, the function Y). Indicate all the decision trees that made up the version space for the function Y given above.

10. (1.5p) **(Algoritmul Bayes Naiv: calculul ratei medii a erorii – exemplificare)**

Consider a binary classification problem with variable $X_1 \in \{0,1\}$ and label $Y \in \{0,1\}$. The true generative distribution $P(X_1, Y) = P(Y)P(X_1|Y)$ is shown in the following tables:

Y	0	1
$P(Y)$	0.7	0.3

$P(X_1\|Y)$	$Y=0$	$Y=1$
$X_1 = 0$	0.8	0.4
$X_1 = 1$	0.2	0.6

a. Suppose that we have trained a Naive Bayes classifier, using infinite *training data* generated according to the above tables. Please write down in the following table the *predictions* from the trained Naive Bayes for different configurations of X_1. Note that $\hat{Y}(X_1)$ in the table is the decision about the value of Y given X_1. For decision terms in the table, write down either $\hat{Y} = 0$ or $\hat{Y} = 1$; for probability terms in the table, write down the actual values (and the calculation process, e.g., $0.8 \cdot 0.7 = 0.56$).

	$P(X_1, Y=0)$	$P(X_1, Y=1)$	$\hat{Y}(X_1)$
$X_1 = 0$			
$X_1 = 1$			

b. What is the *expected error rate* of this Naive Bayes classifier on *testing examples* that are generated according to the given (first two) tables? In other words, [calculate] $P(\hat{Y}(X_1) \neq Y)$ when (X_1, Y) is generated according to the two tables.

For the next three questions, consider two variables X_1, $X_2 \in \{0,1\}$ and the label $Y \in \{0,1\}$. Y and X_1 are still generated according to the given (first two) tables, and then X_2 is created as a *negated copy* of X_1, i.e. $X_2 = 1$ iff $X_1 = 0$ and $X_2 = 0$ iff $X_1 = 1$.

c. Suppose now that we have trained a Naive Bayes classifier, using infinite *training data* that are generated according to the given (first two) tables and the *negated copy* rule. In the following table, please write down the *predictions* from the trained Naive Bayes for different configurations of (X_1, X_2).

	$\hat{P}(X_1, X_2, Y=0)$	$\hat{P}(X_1, X_2, Y=1)$	$\hat{Y}(X_1, X_2)$
$X_1 = 0, X_2 = 0$			
$X_1 = 0, X_2 = 1$			
$X_1 = 1, X_2 = 0$			
$X_1 = 1, X_2 = 1$			

d. What is the *expected error rate* of this Naive Bayes classifier on testing examples that are generated according to the given (first two) tables and the duplication rule?

e. Compared to the scenario without X_2, how does the expected error rate change (i.e., increase or decrease)? In the previous table (at point c), the decision rule $\hat{Y}$ on *which* configuration is responsible to this change? What actually happened to this decision rule?