You may complete the following entirely in a Jupyter notebook. Ensure that the notebook has your name on it. Save the notebook as a PDF and submit the PDF through Canvas.

1) **Decision boundary of logistic regression.**

Let's practice plotting the linear decision boundary learned by a logistic regression model. Assume you are given the following fitted model:

$$P(y = 1|\mathbf{x}) = \frac{1}{1 + e^{-(32-9.4x_1+8x_2)}}$$

(a) Show that

$$\text{if } P(y = 1|\mathbf{x}) \geq \frac{1}{2} \implies 32 - 9.4x_1 + 8x_2 \geq 0$$

$$\text{if } P(y = 1|\mathbf{x}) < \frac{1}{2} \implies 32 - 9.4x_1 + 8x_2 < 0$$

This shows that to get the equation of the separating hyperplane, we can take the exponent from the exponential function and set it equal to 0:

$$32 - 9.4x_1 + 8x_2 = 0$$

Next, if we re-arrange the equation, we obtain:

$$-8x_2 = 32 - 9.4x_1$$

which simplifies to:

$$x_2 = -4 + 1.175x_1$$

We can now use NumPy's `linspace` function to generate the x_1 values and then plug them into the line's equation to generate the values for x_2. Lastly, we can plot the line defined by the two arrays of x_1 and x_2 values.

Your turn! Find the equation of the line/plane for each fitted model below and plot the line/plane.

(a)

$$P(y = 1|\mathbf{x}) = \frac{1}{1 + e^{-(-20.96-123.08x_1+40.46x_2)}}$$

(b) *Optional*

$$P(y = 1|\mathbf{x}) = \frac{1}{1 + e^{-(259.55-27.15x_1-30.49x_2)}}$$

(c) *Optional*

$$P(y = 1|\mathbf{x}) = \frac{1}{1 + e^{-(-26.7-284.7x_1+284.9x_2-64.6x_3)}}$$

2) **Logistic likelihood function**

A logistic regression model is trained by finding weights which optimize the logistic likelihood function:

$$L = \prod_{j=1}^{N} P(y^{(j)} = 1|\mathbf{x}^{(j)})^{y^{(j)}}(1 - P(y^{(j)} = 1|\mathbf{x}^{(j)}))^{1-y^{(j)}}$$

where $\mathbf{x}^{(j)}$ represents the feature vector of the j^{th} sample and $y^{(j)}$ represents the label of the j^{th} sample.

Logistic regression computes the probability that each sample $\mathbf{x}^{(j)}$ belongs to class 1. Formally, the logistic regression model is referred to as $P(y^{(j)} = 1|\mathbf{x}^{(j)})$. Since logistic regression only assumes two possible outcomes, the probability that the sample is not in class 1 is given by $1 - P(y^{(j)} = 1|\mathbf{x}^{(j)})$. Depending on which class (designated 0 or 1) the sample is in, the exponent of one term will be 1 and the other will be 0. So only one term is used per sample in the calculation of the likelihood.

For example, assume we are trying to distinguish between "setosa" and "non-setosa" samples. If a sample $j = 5$ has relatively small petal lengths and widths, there is a high probability (e.g., 99%) that the sample belongs to the "setosa" class. We would write this as $P(y^{(5)} = setosa|\mathbf{x}^{(5)}) = 0.99$. Assume that the sample genuinely does belong to the "setosa" species so $y^{(5)} = 1$. Since $1 - y^{(5)} = 1 - 1 = 0$, the $1 - P(y^{(5)} = 1|\mathbf{x}^{(5)})$ is reduced to 1. We then calculate the rest to get the likelihood for that sample:

$$\begin{aligned}
&L_j = P(y^{(j)} = 1|\mathbf{x}^{(j)})^{y^{(j)}}(1 - P(y^{(j)} = 1|\mathbf{x}^{(j)}))^{1-y^{(j)}} \\
&L_5 = 0.99^1(1 - 0.99)^{1-1} \\
&L_5 = 0.99
\end{aligned}$$

Now, let's look at the case where the species of sample 7's species is not setosa but the model predicts that it is. Assume the model gives us a probability of 75% of being in setosa: $P(y^{(7)} = setosa|\mathbf{x}^{(7)}) = 0.75$. The true class of the sample is 0 so $y^{(7)} = 0$.

$$\begin{aligned}
&L_j = P(y^{(j)} = 1|\mathbf{x}^{(j)})^{y^{(j)}}(1 - P(y^{(j)} = 1|\ \mathbf{x}^{(j)}))^{1-y^{(j)}} \\
&L_7 = 0.75^0(1 - 0.75)^{1-0} \\
&L_7 = 1 - 0.75 \\
&L_7 = 0.25
\end{aligned}$$

(a) Calculate the likelihood for each sample and overall when all of the model's estimated probabilities are consistent with the true class.

P(y = 1)	**y**	**Per-Sample Likelihood**
0.99	1	
0.99	1	
0.01	0	
0.01	0	
0.99	1	

Overall likelihood:

(b) *Optional* Calculate the likelihood for each sample and overall when none of the model's estimated probabilities are consistent with the true class.

P(y = 1)	**y**	**Per-Sample Likelihood**
0.01	1	
0.01	1	
0.99	0	
0.99	0	
0.01	1	

Overall likelihood:

(c) Calculate the likelihood for each sample and overall for the following five samples.

P(y = 1)	**y**	**Per-Sample Likelihood**
0.75	1	
0.99	1	
0.25	0	
0.01	0	
0.99	1	

Overall likelihood:

(d) What is the best possible likelihood value? What is the worst possible likelihood value?

3) **Interpretability of Logistic Regression**

Linear models such as linear regression and logistic regression are considered interpretable machine learning models.

1. In linear regression, a continuous outcome is predicted as a weighted sum of the feature inputs:

   $$y = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_m x_m$$

   The weights w_1, w_2, ... , w_m help us understand the contribution of each feature to the predicted outcome. For instance, if we keep all features constant and we increase x_1 by 1 unit, how does y change?

2. Your answer to the last question is the way we interpret the weights in linear regression. In logistic regression, the interpretability of the weights differ. A logistic regression model predicts the probability that an instance belongs to class 1 (assuming here a binary classification, $y \in \{0, 1\}$):

   $$P(y = 1|\mathbf{x}) = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_m x_m)}} \quad (1)$$

   To interpret the weights correctly for a logistic model, we need to reformulate the above equation.

   (a) Use (1) to express the following ratio in terms of features and their weights:

   $$\frac{P(y = 1|\mathbf{x})}{P(y = 0|\mathbf{x})}$$

   The ratio (probability of $y = 1$ divided by probability of $y = 0$) is called the **odds**. For example, if you have odds of 2, it means that the probability for $y = 1$ is twice as high as the probability for $y = 0$.

   (b) If we keep all features constant and we increase x_1 by 1 unit, how does the **odds** change?

   (c) *Optional* - Assume that the features are standardized, and that the features in (1) are the scaled features. If we keep all features constant and we increase the unscaled version of the first feature by 1 unit, how does the **odds** change ?

3. A logistic regression model was fitted on heart failure clinical records data set (extracted from here). The data consists of clinical features and the target is to predict heart failure (1 means heart failure and 0 means no heart failure). The features and the weights obtained are the following:

Features)	**Description**	**Weights**
Age	Age of the patient (years)	0.0538
Anaemia	Decrease if red blood cells or hemoglobin (boolean)	0.2287
Diabetes	If the patient has diabetes (boolean)	0.1611
Ejection Fraction	Percentage of blood leaving the heart at each contraction (%)	-0.0723
High Blood Pressure	If the patient has hypertension (boolean)	0.3520
Creatinine	Level of serum creatinine in the blood (mg/dL)	0.6723
Gender	(1) Woman or (0) Man	-0.23

How do you intrepret the weights for the following features: gender and serum creatinine? (How does the odds for heart failure vs no heart failure change if we change each of those features individually?)

4) **Growing Trees - Gini Index**

There are various algorithms to construct a decision tree. The classification and regression trees (CART) algorithm is the most popular algorithm. It consists of a top-down greedy approach known as *recursive binary splitting* that aims at dividing the feature space into non-overlapping regions: it begins at the root or top of the tree, finds the best split at the root and then successively repeats the same steps for each child node (top-down: it starts from the root, greedy: it chooses the best split at each step). When constructing a classification tree, the best split is chosen according to a measure that indicates the node impurity - node impurity refers to how much the data in each node or region is mixed. An example of such measure is the Gini index.

Steps of CART algorithm (classification):

(a) Start at the root

(b) Determine the best feature and threshold to split: for each feature x_j, find the best cutpoint or threshold v such that the sum of the node impurities of the two regions (or two nodes) defined by $x_j < v$ and $x_j \geq v$ is minimized. Then choose the best feature.

(c) If a stopping criterion is reached, exit. Otherwise, repeat for each child node.

In this exercise, you will explore the Gini index and work on finding the root node of a small classification dataset.

1. When a split is performed while constructing a classification tree, two regions in the feature space are created, where each region contains a subset of the training data. To measure how much this subset is impure (or mixed) in a given region, we find the Gini index. If the data consists of C classes, the Gini index (G) is defined as:

$$G = \sum_{c=1}^{C} \text{prop}(c)(1 - \text{prop}(c))$$

where prop(c) represents the proportion of training observations that belong to the c^{th} class in the considered region. When there are two classes (0 and 1), the Gini index reduces to the following:

$$G = 2 \text{ prop(of class 1)}(1 - \text{prop(of class 1)})$$

which can be also shown as

$$G = 1 - \text{ prop(of class 1) prop(of class 1)} - \text{ prop(of class 0) prop(of class 0)}$$

(a) Find the Gini index for each of the following three cases:

- the data points in a region all belong to class 1
- the data points in a region all belong to class 0

- half of the data points in a region belong to class 1

(b) Plot the function of Gini index with respect to prop(of class 1).

2. Consider the following training dataset:

cigsPerDay	**prevalentHyp**	**TenYearCHD**
0	True	0
0	True	1
0	True	0
0	True	1
3	False	1
10	False	0
20	True	1
20	False	0
30	False	0
30	False	0

Features: `cigsPerDay`: number of cigarettes per day, `prevalentHyp`: whether or not the patient was hypertensive.

Target: 10 year risk of coronary heart disease)

To find the root of the tree, we need to check for each feature the possible thresholds and then choose the best threshold for each feature and finally the best feature.

(a) Find the Gini index for the overall training dataset.

(b) Let's take first the feature `prevalentHyp`. It is a categorical feature. In this case, we only have one possible cutpoint or threshold: prevalentHyp = True or = False. This split creates two regions: a region where `prevalentHyp=True` (R_1) and a region where `prevalentHyp=False` (R_2). Compute the Gini index for each region and then find the weighted average of the two Gini indices:

$$\frac{N_1}{N}G(R_1) + \frac{N_2}{N}G(R_2)$$

where N_1 is the total number of data points in R_1 and N_2 is the total number of data points in R_2.

(c) Consider now the feature `cigsPerDay`. The possible values that this feature has taken are: 0, 3, 10, 20 and 30. Then the possible thresholds are: 1.5, 6.5, 15, 25 (middle points between each possible value). For the cutpoint 1.5, compute the Gini index of the two region it creates (region where `cigsPerDay<1.5` and region where `cigsPerDay>=1.5`) and then compute their weighted average. For the remaining threholds (6.5, 15, 25), the weighted average of the Gini indices are respectively: 0.4, 0.417, 0.375. Which threshold for the feature `cigsPerDay` does have the lowest impurity?

(d) Which feature should be chosen for the root node?

5) *Optional* - **Given Trees**

We created two decision tree models to classify the wine data set (see Figure 1). The trees use two different subsets of features. DT 1 uses proline, color intensity, flavonoids, and alcohol. DT 2 uses malic_acid, ash, alcalinity_of_ash, magnesium.

(a) Create scatter plots using the proline and color intensity, flavanoids and alcohol, malic_acid and ash, and alcalinity_of_ash and magnesium feature pairs. The points should be colored by class. *Hint: Look up the `load_wine` function in scikit learn.*

(b) Create a table with 4 rows and 2 columns. List the features used in each plot in one column and whether the classes can be easily separated using the features in that plot.

(c) Trace DT 1 to classify the following samples:

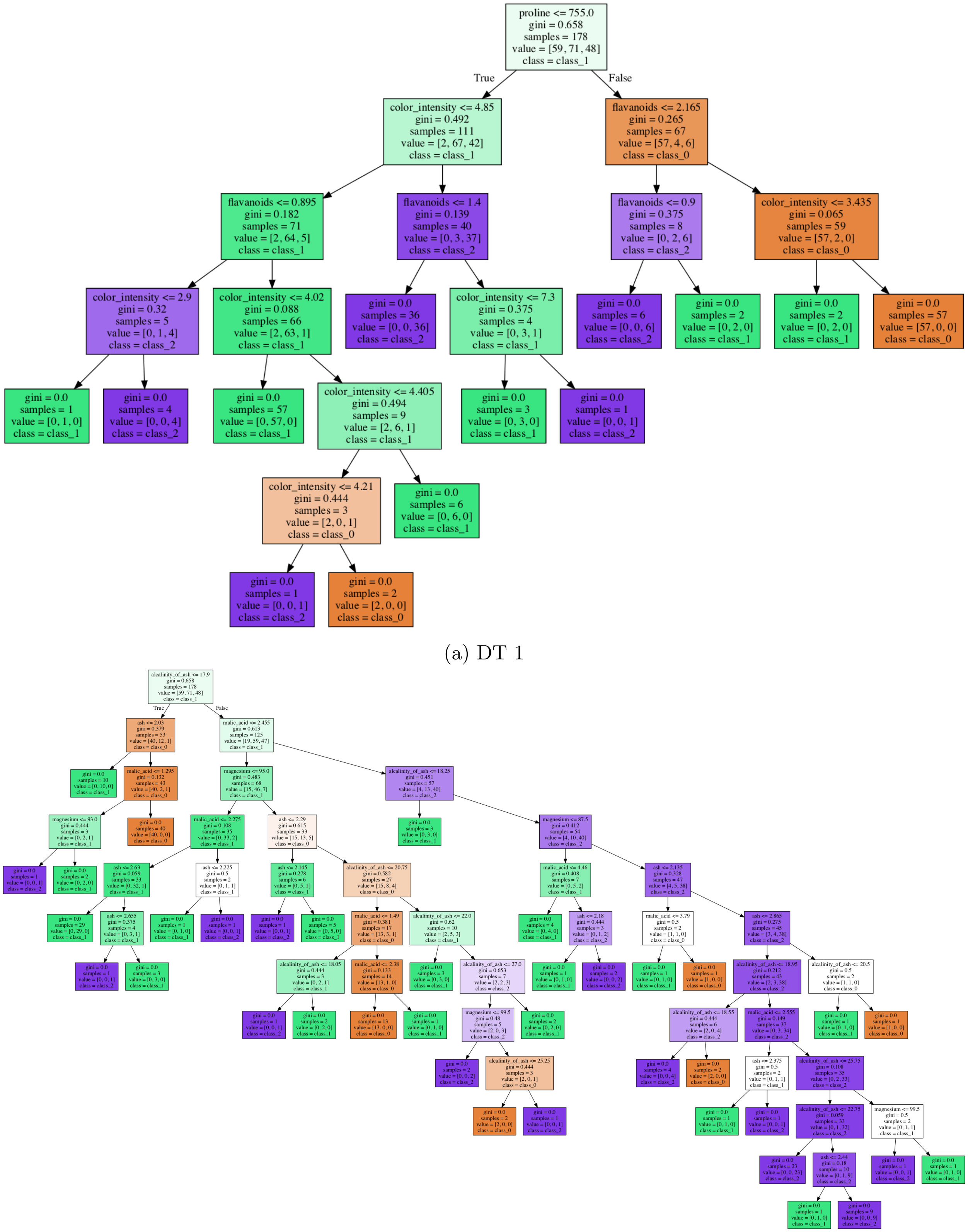
Alcohol	**Color Intensity**	**Flavanoids**	**Proline**	**Predicted Class**
0.5	5	1	500	
0.75	4.25	0.75	525	
0	4	3	800	

(d) For samples in (c), trace the path from the root to the leaf and provide the corresponding Boolean expression.

(e) On the proline and color intensity and flavanoids and alcohol feature pair scatter plots from (a), plot the decision lines for every non-leaf node of the tree.

(f) The two trees are of very different sizes. Why do you think that is?

(g) For logistic regression, we need to use a one-versus-all scheme with multiple models to handle multi-class problems. How are multi-class problems handled with decision trees?


proline <= 755.0 gini = 0.658 samples = 178 value = [59, 71, 48] class = class_1
True
False
color_intensity <= 4.85 gini = 0.492 samples = 111 value = [2, 67, 42] class = class_1
flavanoids <= 2.165 gini = 0.265 samples = 67 value = [57, 4, 6] class = class_0
flavanoids <= 0.895 gini = 0.182 samples = 71 value = [2, 64, 5] class = class_1
flavanoids <= 1.4 gini = 0.139 samples = 40 value = [0, 3, 37] class = class_2
flavanoids <= 0.9 gini = 0.375 samples = 8 value = [0, 2, 6] class = class_2
color_intensity <= 3.435 gini = 0.065 samples = 59 value = [57, 2, 0] class = class_0
color_intensity <= 2.9 gini = 0.32 samples = 5 value = [0, 1, 4] class = class_2
color_intensity <= 4.02 gini = 0.088 samples = 66 value = [2, 63, 1] class = class_1
gini = 0.0 samples = 36 value = [0, 0, 36] class = class_2
color_intensity <= 7.3 gini = 0.375 samples = 4 value = [0, 3, 1] class = class_1
gini = 0.0 samples = 6 value = [0, 0, 6] class = class_2
gini = 0.0 samples = 2 value = [0, 2, 0] class = class_1
gini = 0.0 samples = 2 value = [0, 2, 0] class = class_1
gini = 0.0 samples = 57 value = [57, 0, 0] class = class_0
gini = 0.0 samples = 1 value = [0, 1, 0] class = class_1
gini = 0.0 samples = 4 value = [0, 0, 4] class = class_2
gini = 0.0 samples = 57 value = [0, 57, 0] class = class_1
color_intensity <= 4.405 gini = 0.494 samples = 9 value = [2, 6, 1] class = class_1
gini = 0.0 samples = 3 value = [0, 3, 0] class = class_1
gini = 0.0 samples = 1 value = [0, 0, 1] class = class_2
color_intensity <= 4.21 gini = 0.444 samples = 3 value = [2, 0, 1] class = class_0
gini = 0.0 samples = 6 value = [0, 6, 0] class = class_1
gini = 0.0 samples = 1 value = [0, 0, 1] class = class_2
gini = 0.0 samples = 2 value = [2, 0, 0] class = class_0

(a) DT 1

alcalinity_of_ash <= 17.9 gini = 0.658 samples = 178 value = [59, 71, 48] class = class_1
True
False
ash <= 2.03 gini = 0.379 samples = 53 value = [40, 12, 1] class = class_0
malic_acid <= 2.455 gini = 0.613 samples = 125 value = [19, 59, 47] class = class_1
gini = 0.0 samples = 10 value = [0, 10, 0] class = class_1
malic_acid <= 1.295 gini = 0.132 samples = 43 value = [40, 2, 1] class = class_0
magnesium <= 95.0 gini = 0.483 samples = 68 value = [15, 46, 7] class = class_1
alcalinity_of_ash <= 18.25 gini = 0.451 samples = 57 value = [4, 13, 40] class = class_2
magnesium <= 93.0 gini = 0.444 samples = 3 value = [0, 2, 1] class = class_1
gini = 0.0 samples = 40 value = [40, 0, 0] class = class_0
malic_acid <= 2.275 gini = 0.108 samples = 35 value = [0, 33, 2] class = class_1
ash <= 2.29 gini = 0.615 samples = 33 value = [15, 13, 5] class = class_0
gini = 0.0 samples = 3 value = [0, 3, 0] class = class_1
magnesium <= 87.5 gini = 0.412 samples = 54 value = [4, 10, 40] class = class_2
gini = 0.0 samples = 1 value = [0, 0, 1] class = class_2
gini = 0.0 samples = 2 value = [0, 2, 0] class = class_1
ash <= 2.63 gini = 0.059 samples = 33 value = [0, 32, 1] class = class_1
ash <= 2.225 gini = 0.5 samples = 2 value = [0, 1, 1] class = class_1
ash <= 2.145 gini = 0.278 samples = 6 value = [0, 5, 1] class = class_1
alcalinity_of_ash <= 20.75 gini = 0.582 samples = 27 value = [15, 8, 4] class = class_0
malic_acid <= 4.46 gini = 0.408 samples = 7 value = [0, 5, 2] class = class_1
ash <= 2.135 gini = 0.328 samples = 47 value = [4, 5, 38] class = class_2
gini = 0.0 samples = 29 value = [0, 29, 0] class = class_1
ash <= 2.655 gini = 0.375 samples = 4 value = [0, 3, 1] class = class_1
gini = 0.0 samples = 1 value = [0, 1, 0] class = class_1
gini = 0.0 samples = 1 value = [0, 0, 1] class = class_2
gini = 0.0 samples = 1 value = [0, 0, 1] class = class_2
gini = 0.0 samples = 5 value = [0, 5, 0] class = class_1
malic_acid <= 1.49 gini = 0.381 samples = 17 value = [13, 3, 1] class = class_0
alcalinity_of_ash <= 22.0 gini = 0.62 samples = 10 value = [2, 5, 3] class = class_1
gini = 0.0 samples = 4 value = [0, 4, 0] class = class_1
ash <= 2.18 gini = 0.444 samples = 3 value = [0, 1, 2] class = class_2
malic_acid <= 3.79 gini = 0.5 samples = 2 value = [1, 1, 0] class = class_0
ash <= 2.865 gini = 0.275 samples = 45 value = [3, 4, 38] class = class_2
gini = 0.0 samples = 1 value = [0, 0, 1] class = class_2
gini = 0.0 samples = 3 value = [0, 3, 0] class = class_1
alcalinity_of_ash <= 18.05 gini = 0.444 samples = 3 value = [0, 2, 1] class = class_1
malic_acid <= 2.38 gini = 0.133 samples = 14 value = [13, 1, 0] class = class_0
gini = 0.0 samples = 3 value = [0, 3, 0] class = class_1
alcalinity_of_ash <= 27.0 gini = 0.653 samples = 7 value = [2, 2, 3] class = class_2
gini = 0.0 samples = 1 value = [0, 1, 0] class = class_1
gini = 0.0 samples = 2 value = [0, 0, 2] class = class_2
gini = 0.0 samples = 1 value = [0, 1, 0] class = class_1
gini = 0.0 samples = 1 value = [1, 0, 0] class = class_0
alcalinity_of_ash <= 18.95 gini = 0.212 samples = 43 value = [2, 3, 38] class = class_2
alcalinity_of_ash <= 20.5 gini = 0.5 samples = 2 value = [1, 1, 0] class = class_0
gini = 0.0 samples = 1 value = [0, 0, 1] class = class_2
gini = 0.0 samples = 2 value = [0, 2, 0] class = class_1
gini = 0.0 samples = 13 value = [13, 0, 0] class = class_0
gini = 0.0 samples = 1 value = [0, 1, 0] class = class_1
magnesium <= 99.5 gini = 0.48 samples = 5 value = [2, 0, 3] class = class_2
gini = 0.0 samples = 2 value = [0, 2, 0] class = class_1
alcalinity_of_ash <= 18.55 gini = 0.444 samples = 6 value = [2, 0, 4] class = class_2
malic_acid <= 2.555 gini = 0.149 samples = 37 value = [0, 3, 34] class = class_2
gini = 0.0 samples = 1 value = [0, 1, 0] class = class_1
gini = 0.0 samples = 1 value = [1, 0, 0] class = class_0
gini = 0.0 samples = 2 value = [0, 0, 2] class = class_2
alcalinity_of_ash <= 25.25 gini = 0.444 samples = 3 value = [2, 0, 1] class = class_0
gini = 0.0 samples = 4 value = [0, 0, 4] class = class_2
gini = 0.0 samples = 2 value = [2, 0, 0] class = class_0
ash <= 2.375 gini = 0.5 samples = 2 value = [0, 1, 1] class = class_1
alcalinity_of_ash <= 25.75 gini = 0.108 samples = 35 value = [0, 2, 33] class = class_2
gini = 0.0 samples = 2 value = [2, 0, 0] class = class_0
gini = 0.0 samples = 1 value = [0, 0, 1] class = class_2
gini = 0.0 samples = 1 value = [0, 1, 0] class = class_1
gini = 0.0 samples = 1 value = [0, 0, 1] class = class_2
alcalinity_of_ash <= 22.75 gini = 0.059 samples = 33 value = [0, 1, 32] class = class_2
magnesium <= 99.5 gini = 0.5 samples = 2 value = [0, 1, 1] class = class_1
gini = 0.0 samples = 23 value = [0, 0, 23] class = class_2
ash <= 2.44 gini = 0.18 samples = 10 value = [0, 1, 9] class = class_2
gini = 0.0 samples = 1 value = [0, 0, 1] class = class_2
gini = 0.0 samples = 1 value = [0, 1, 0] class = class_1
gini = 0.0 samples = 1 value = [0, 1, 0] class = class_1
gini = 0.0 samples = 9 value = [0, 0, 9] class = class_2

(b) DT 2

Figure 1: Two decision trees trained on different subsets of features from the wine dataset.