COMP7404 Computational intelligence and machine learning

Assignment 4 Part A

ZHANG Hongyi

A1

Consider a Perceptron with 2 inputs and 1 output. Let the weights of the Perceptron be $w_1 = 1$ and $w_2 = 1$ and let the bias be $w_0 = -1.5$. Calculate the output of the following inputs:
$(0,0), (1,0), (0,1), (1,1)$

Answer

For input $(0,0)$, $z = w_0 + x_1 w_1 + x_2 w_2 = -1.5$, and $\varphi(-1.5) = -1$. So the output is -1.

For input $(1,0)$, $z = w_0 + x_1 w_1 + x_2 w_2 = -1.5 + 1 \times 1 = -0.5$, and $\varphi(-0.5) = -1$. So the output is -1.

For input $(0,1)$, $z = w_0 + x_1 w_1 + x_2 w_2 = -1.5 + 1 \times 1 = -0.5$, and $\varphi(-0.5) = -1$. So the output is -1.

For input $(1,1)$, $z = w_0 + x_1 w_1 + x_2 w_2 = -1.5 + 1 \times 1 + 1 \times 1 = 0.5$, and $\varphi(0.5) = 1$. So the output is 1.

In summary, the outputs are: $-1, -1, -1, 1$.

A2

Define a perceptron for the following logical functions: AND, NOT, NAND, NOR

Answer

For all the perceptrons below, we define an activation function:

$$\varphi(z) = \begin{cases} 1 \text{ if } z \geq 0 \\ 0 \text{ if } z < 0 \end{cases}$$

AND

$w_0 = -1.5, w_1 = 1, w_2 = 1$

Input (x_1, x_2)	Output
$(0,0)$	0
$(0,1)$	0
$(1,0)$	0
$(1,1)$	1

NOT

$w_0 = 0.5, w_1 = -1$

Input x_1	Output
1	0

0	1

NAND

$w_0 = 1.5, w_1 = -1, w_2 = -1$

Input (x_1, x_2)	Output
$(0, 0)$	1
$(0, 1)$	1
$(1, 0)$	1
$(1, 1)$	0

NOR

$w_0 = 0.5, w_1 = -1, w_2 = -1$

Input (x_1, x_2)	Output
$(0, 0)$	1
$(0, 1)$	0
$(1, 0)$	0
$(1, 1)$	0

A3

The parity problem returns 1 if the number of inputs that are 1 is even, and 0 otherwise. Can a perceptron learn this problem for 3 inputs?

Answer

No. Perceptrons cannot learn this problem for 3 input since the problem is not *linearly separable*. There is no single hyperplane that can separate the inputs that should map to 1 from the inputs that should map to 0.

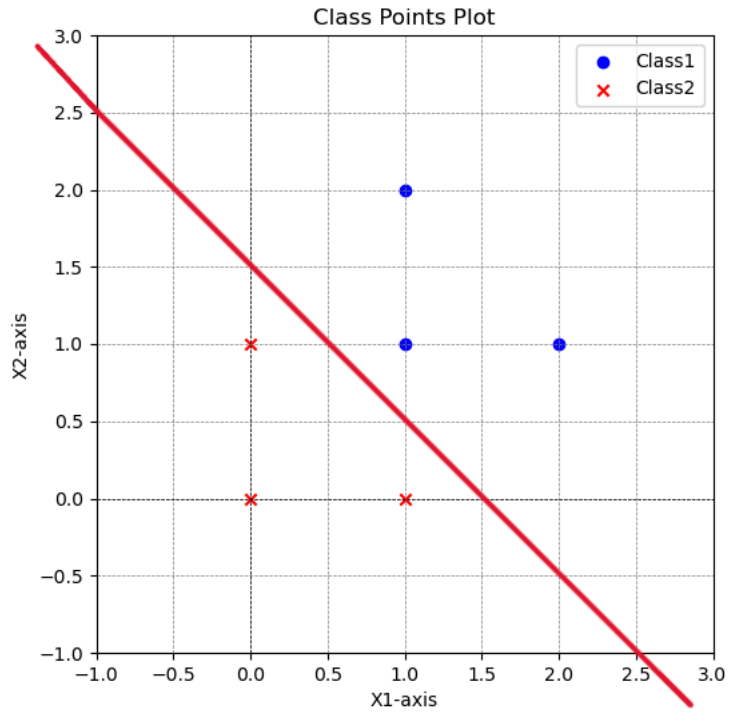
A4

Suppose that the following are a set of point in two classes:

- Class1: $\binom{1}{1}^\top, (1, 2), (2, 1)$
- Class2: $(0, 0), (1, 0), (0, 1)$

Plot them and find the optimal separating line. What are the support vectors, and what is the meaning?

Answer



Let's assume the support vectors are: $x_+ = \begin{pmatrix}1\\1\end{pmatrix}, x_- = \begin{pmatrix}1\\0\end{pmatrix}$ and $\begin{pmatrix}0\\1\end{pmatrix}$. Then the margin is $\frac{(x_+ - x_-)^\top w}{\| w \|} = \sqrt{2}$. Thus, the optimal seperating line is $x_1 + x_2 = 1.5$.

The support vectors are the most "informative" points in the dataset. They are the points that directly affect the position of the optimal separating hyperplane. If you were to remove these points, the separating boundary could shift. In this case, the support vectors are $x_+ = \begin{pmatrix}1\\1\end{pmatrix}, x_- = \begin{pmatrix}1\\0\end{pmatrix}$ and $\begin{pmatrix}0\\1\end{pmatrix}$. These points are the closest to the separating hyperplane, and removing them would change the decision boundary.

A5

Suppose that the probability of five events are $P(\text{first}) = 0.5, P(\text{second}) = P(\text{third}) = P(\text{fourth}) = P(\text{fifth}) = 0.125$. Calculate the entropy and write down in words what this means.

Answer

$$
\begin{aligned}
H &= \sum_i p_i \log_2 \frac{1}{p_i} \\
&= 0.5 \times \log_2 2 + 4 \times 0.125 \log_2 8 \\
&= 0.5 + 1.5 = 2
\end{aligned}
$$

The entropy is 2.

Entropy is a numerical measure that quantifies the concept of uncertainty. The larger entropy demonstrates the higher uncertainty the situation has. It is usually used in Binary Decision Tree to select the tests used in a decision tree. Typically, we always choose the test that can reduce the entropy the most.

A6

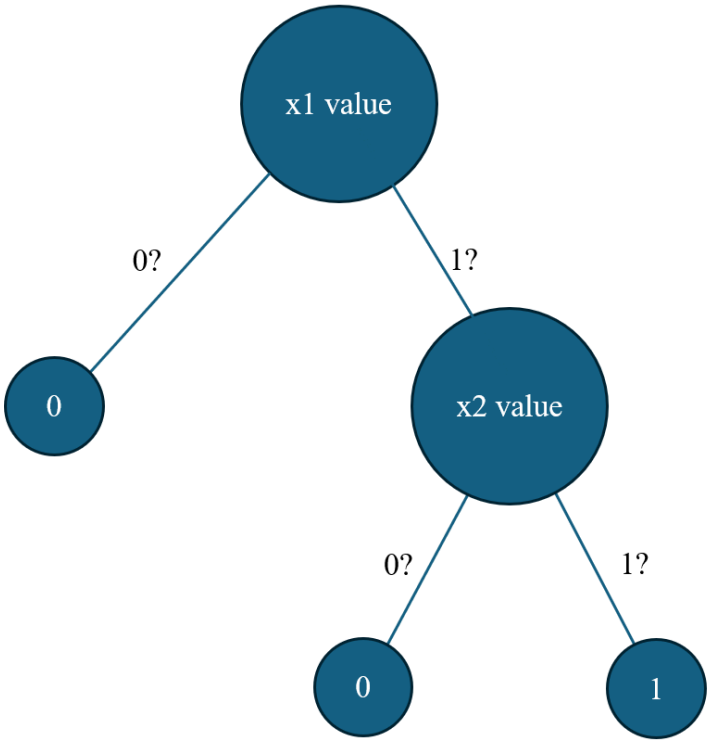
Design a decision tree that computes the logical AND function. How does it compare to the Perceptron solution?

Answer

Decision Tree

- **Root Node**: Check the value of x_1.
  - If $x_1 = 0$, output 0,
  - If $x_1 = 1$, move to the next node.
- **Second Node**: Check the value of x_2.
  - If $x_2 = 0$, output 0,
  - If $x_2 = 1$, output 1.

The overall structure can be illustrated as:



Comparison to the Perceptron Solution

- **Decision Tree**: The decision tree doesn't rely on a linear decision boundary. Instead, it uses binary splits, which make it intuitive and transparent for computing discrete functions like AND. It can be used to solve problems which are not linear separable such as XOR.
- **Perceptron**: The Perceptron defines a linear decision boundary between classes based on the weights and bias. For AND, this boundary will separate the $(1, 1)$ case (output 1) from the other cases (output 0). This approach works well for binary logic functions that can be separated linearly, like AND. However, perceptron cannot handle problems that are not linear separable.

A7

Turn the following politically incorrect data into a decision tree to classify which attributes make a person attractive, and then extract the rules. Use the Gini Impurity.

Height	Hair	Eyes	Attractive?
Small	Blonde	Brown	No
Tall	Dark	Brown	No
Tall	Blonde	Blue	Yes
Tall	Dark	Blue	No
Small	Dark	Blue	No
Tall	Red	Blue	Yes
Tall	Blonde	Brown	No
Small	Blonde	Blue	Yes

Answer

First, we calculate the Gini Impurity of the entire dataset.

$$I_G = 1 - \sum_{i=1}^{c} p_i^2 = 1 - \left(\frac{5}{8}\right)^2 - \left(\frac{3}{8}\right)^2 = 0.46875$$

If we split the dataset by Height,

- Tall: $I_G(\text{Tall}) = 1 - \left(\frac{2}{5}\right)^2 - \left(\frac{3}{5}\right)^2 = 0.48$
- Small: $I_G(\text{Small}) = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2 = 0.444$

So the weighted Gini Impurity is: $\left(\frac{3}{8}\right) \times 0.44 + \left(\frac{5}{8}\right) \times 0.48 = 0.467$.

If we split it by Hair,

- Blonde: $I_G(\text{Blonde}) = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2 = 0.444$
- Dark: $I_G(\text{Dark}) = 1 - 1 = 0$
- Red: $I_G(\text{Red}) = 1 - 1 = 0$

So the weighted Gini Impurity is: $\left(\frac{1}{2}\right) \times 0.444 = 0.222$.

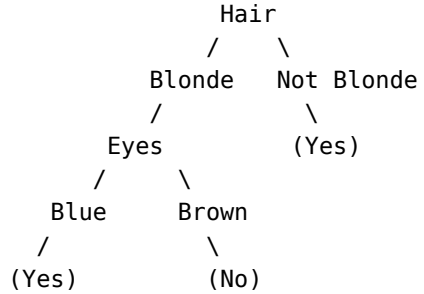
If we split it by Eyes,

- Brown: $I_G(\text{Brown}) = 1 - 1 = 0$
- Blue: $I_G(\text{Blue}) = 1 - \left(\frac{3}{5}\right)^2 - \left(\frac{2}{5}\right)^2 = 0.48$

So the weighted Gini Impurity is: $\frac{5}{8} \times 0.48 = 0.3$.

Splitting by Hairs achieved the lowest Gini Impurity. So we choose Hair as the root node.

We then split the data set as described above, and we can get the final decision tree:



Extracted rules

- If Hair is Dark, then Attractive = No
- If Hair is Red, then Attractive = Yes
- If Hair is Blonde and Eyes are Blue, then Attractive = Yes
- If Hair is Blonde and Eyes are Brown, then Attractive = No

A8

Suppose we collect data for a group of students in a postgraduate machine learning class with features x_1 = hours studies, x_2 = undergraduate GPA and label y = receive an A. We fit a logistic regression and produce estimated weights as follows: $w_0 = -6$, $w_1 = 0.05$, $w_2 = 1$.

1. Estimate the probability that a student who studies for 40h and has an undergraduate GPA of 3.5 gets an A in the class
2. How many hours would the student in part 1. need to study to have a 50% chance of getting an A in the class?

Answer

Part 1

$$z = w_0 + w_1 x_1 + w_2 x_2 = -6 + 0.05 \times 40 + 1 \times 3.5 = -6 + 2 + 3.5 = -0.5$$

$$p(A \mid x) = \varphi(z) = \frac{1}{1 + e^{0.5}} = 0.378$$

So, the probability that a student who studies for 40h and has an undergraduate GPA of 3.5 gets an A in the class is **37.8%**.

Part 2

Let's assume the student need to study t hours to have a 50% chance of getting an A.

$$\frac{1}{1 + e^{-z}} = 0.5$$

$$z = 0$$

$$w_0 + w_1 t + w_2 x_2 = 0$$

$$t = 50$$

So, the student need to study **50 hours** to have a 50% chance of getting an A.

A9

Suppose that we take a data set, divide it into equally-sized training and test sets, and then try out two different classification procedures. First we use logistic regression and get an error rate of 20% on the training data and 30% on the test data. Next we use 1-nearest neighbors (i.e., K=1) and get an average error rate (averaged over both test and training data sets) of 18%. Based on these results, which method should we prefer to use for classification of new observations? Why?

Answer

Based on the result aforementioned, I prefer to use logistic regression for classification.

For the training and test error, logistic regression has a lower training error than its test error. This suggests that the model is likely generalizing reasonably well, though there is some performance drop on unseen data. On the other hand, 1-NN typically has a perfect training error rate of 0%

(because 1-NN memorizes the training data exactly). This means that the 1-NN test error must be significantly higher than the training error, given that the average error rate is 18%. This suggests that 1-NN is likely overfitting the training data and performing poorly on the test set. Therefore, logistic regression should be prefered.

A10

Suppose the features in your training set have very different scales. Which algorithms discussed in class might suffer from this, and how? What can you do about it?

Answer

1. K-Nearest Neighbors (KNN), Support Vectors Machine (SVM): These algorithm rely on distance calculations to find nearest neighbors or the optimal hyperplane. If features are on different scales, the distance calculation can be dominated by features with larger values, affecting the margin and decision boundary.
2. Logistic Regression and Linear Regression: These algorithms calculate weights for each feature to fit the data, but if features vary widely in scale, features with larger scales can disproportionately influence the model's coefficients. This can lead to poor interpretability and convergence issues, especially for gradient-based optimization.

For the issue above, scaling features to have a consistent range using standardization (e.g., transforming features to have a mean of 0 and a standard deviation of 1) or normalization (scaling features to fall between 0 and 1) can help these algorithms perform better.

A11

If your AdaBoost ensemble underfits the training data, which hyperparameters should you tweak and how?

Answer

Underfitting basically means the model is not complex enough to capture the pattern in the training data well. To solve this problem, we can try to tweak these hyperparameters:

1. **The number of estimators**: A small number of estimators can lead to an underfit model since there may not be enough iterations to capture the data's complexity. Increasing the number of estimators allows AdaBoost to build more weak learners and better adapt to the training data.
2. **Use a More Complex Base Estimator**: By default, AdaBoost uses decision stumps (one-level decision trees) as weak learners, which are simple models. If the dataset has more complex patterns, consider using a slightly deeper decision tree as the base estimator (for example, trees with a max depth of 2 or 3).

A12

What is the benefit of out-of-bag evaluation?

Answer

According to the idea of bootstrap sampling, there will be about 33% of the training examples that will never be used to train decision trees. And out-of-bag evaluation is a technique of using these data to evaluate the model without requiring a separate validation set. This can effectively increase data utilization.

This technique can also provide reliable error estimation, since OOB error is often a reliable estimate of the model's generalization error on unseen data, as each training example is only tested on

models that did not include it in their training set. This leads to a realistic assessment of the model's accuracy.

A13

What is the difference between hard and soft voting classifiers?

Answer

- Hard voting uses a majority vote on discrete class labels, while soft voting averages class probabilities.
- Soft voting generally gives better results when models output reliable probability estimates, as it factors in prediction confidence.
- Hard voting is simpler and useful when model probabilities are not available or reliable.