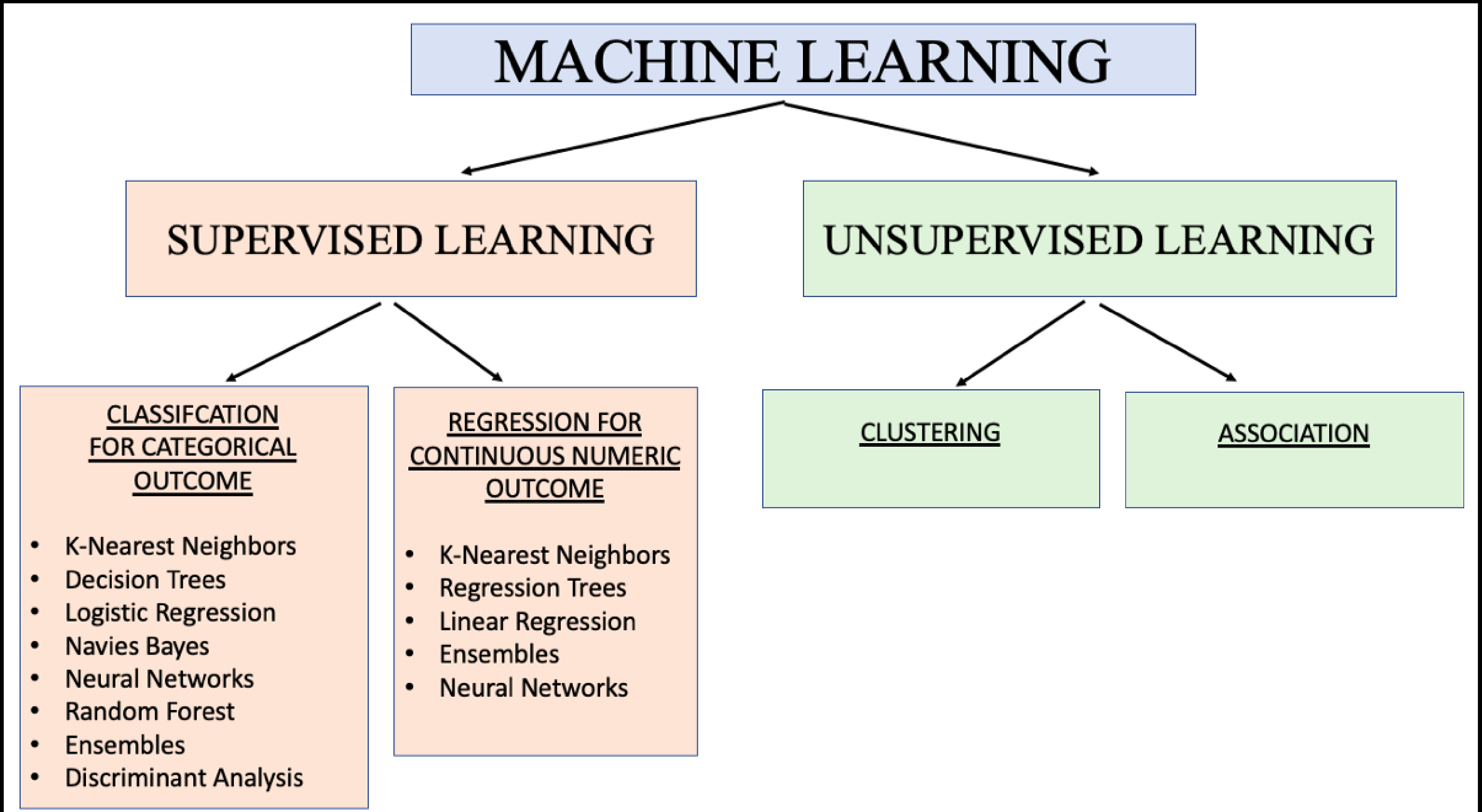
Pattern Recognition

Classification

Introduction



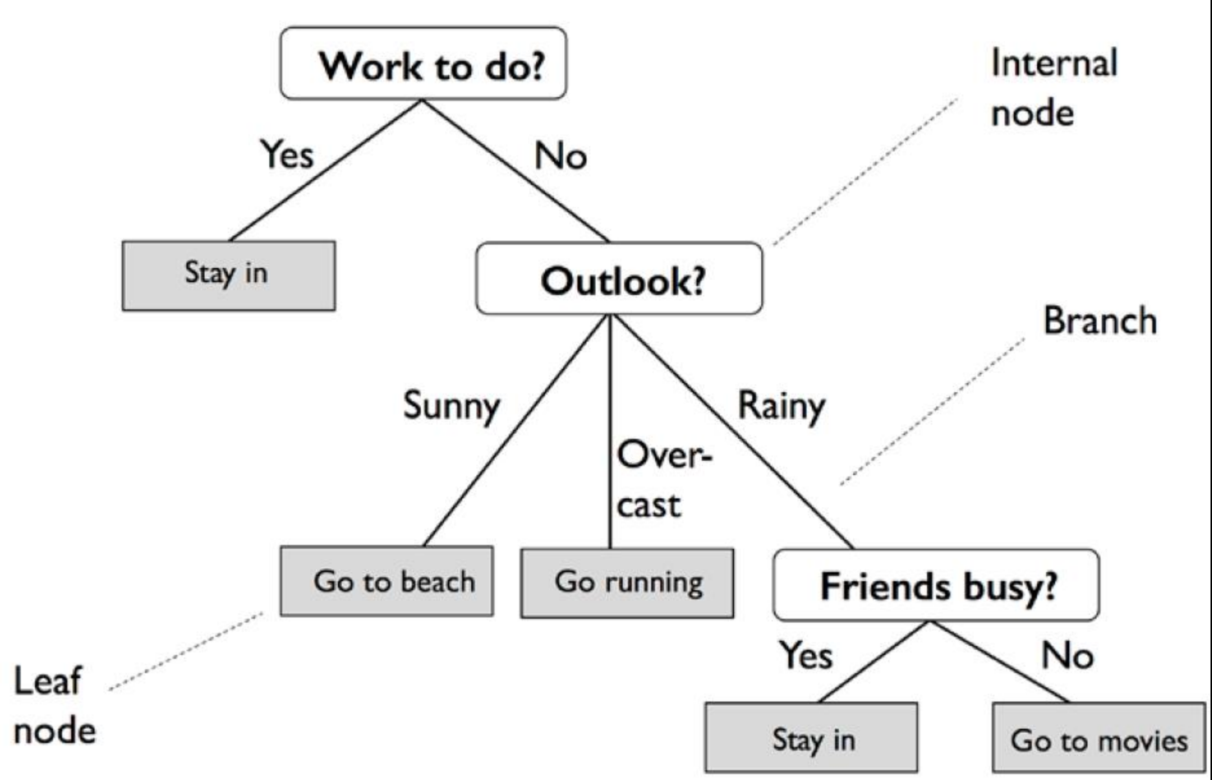
https://miro.medium.com/max/1200/1*dqeortojRQDxvA_um2d8uA.png

Content

Content
Decision Trees
K-Nearest Neighbors
Logistic Regression

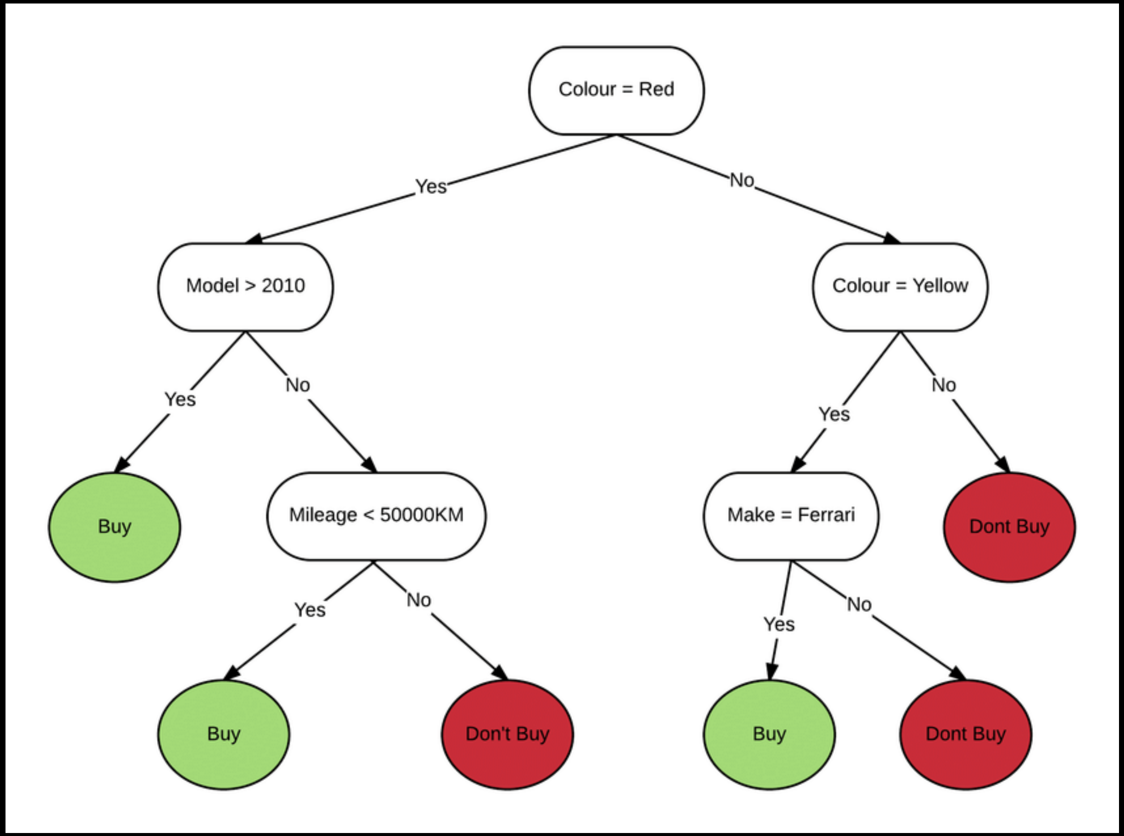
Decision Trees

- We can think of this model as breaking down our data by making a decision based on asking a series of questions.



Decision Trees

- Based on the features in our training dataset, the decision tree model learns a series of questions to infer the class labels of the examples.



Decision Trees

Building a decision tree is an iterative process.

1. We start with a root node and split the data based on the features that results in the largest **information gain (IG)**.
2. Repeat the splitting process at each child node until the leaves gives the target value.
3. The tree can be very deep, which results in **overfitting**, so we **prune** the tree by setting a limit to the maximum depth.

Decision Trees

- To split the nodes at the most informative features, we need to define an objective function to optimize.
- Here, our objective function is to maximize the **IG** at each split

$$IG(D_p, f) = I(D_p) - \sum_{j=1}^{m} \frac{N_j}{N_p} I(D_j)$$

  - f is the feature to perform the split.
  - D_p is the dataset at the parent, D_j is the dataset at the child.
  - I is an **impurity** measure.
  - N_p is the total number of training examples at the parent node.
  - N_j is the total number of training examples at the child node.

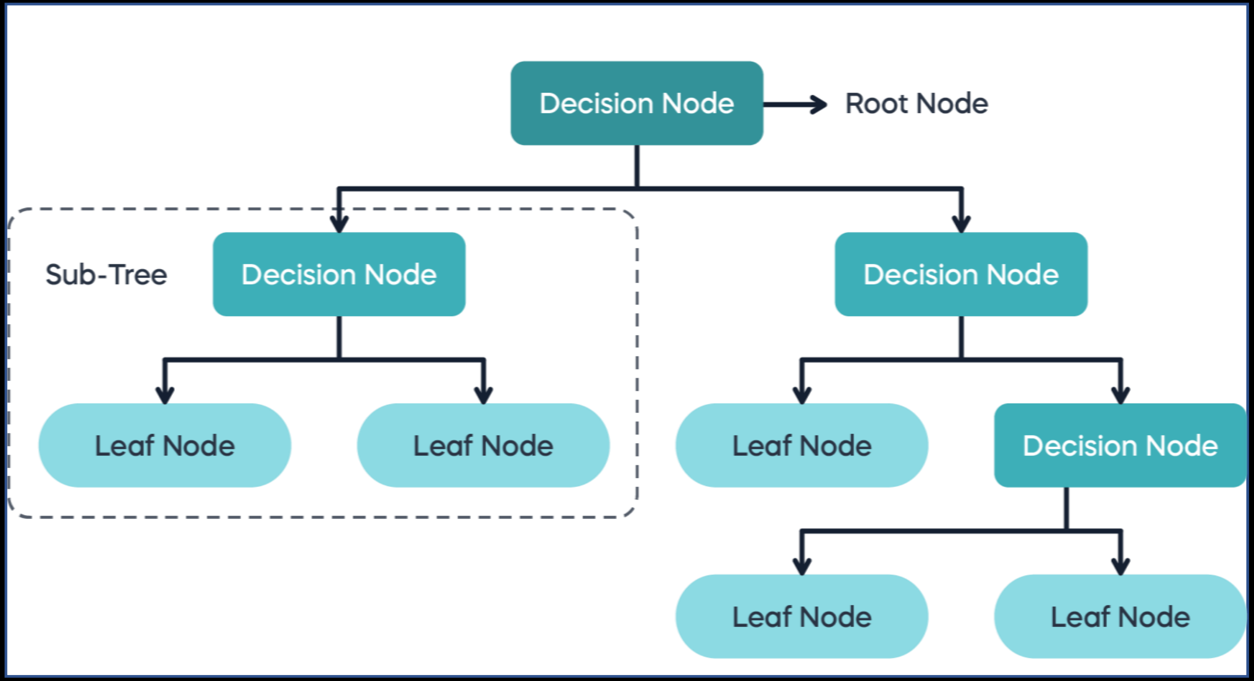
Decision Trees

$$IG(D_p, f) = I(D_p) - \sum_{j=1}^{m} \frac{N_j}{N_p} I(D_j)$$

- The information gain is the difference between the impurity of the parent node and the sum of the child node impurities.
- The lower the impurities of the child nodes, the larger the information gain.

Decision Trees

- Impurity measures allows identifying which feature (attribute) to consider as a root node at each level.
- There are three impurity measures
  - Entropy (I_H)
  - Gini impurity (I_G)
  - Classification error (I_E)



Decision Trees

Entropy

- Entropy measures the expected (i.e., average) amount of information conveyed by identifying the outcome of a random trial.
- For example, casting a die has higher entropy than tossing a coin because each outcome of a die toss has smaller probability ($\frac{1}{6}$) than each outcome of a coin toss ($\frac{1}{2}$)
- Therefore, we can say that the entropy criterion attempts to maximize the mutual information in the tree.

Decision Trees

Gini impurity

- It can be understood as a criterion to minimize the probability of misclassification.
- In practice, both the Gini impurity and entropy typically yield very similar results.
  - It is often not worth spending much time on evaluating trees using different impurity criteria.

Decision Trees

Classification error

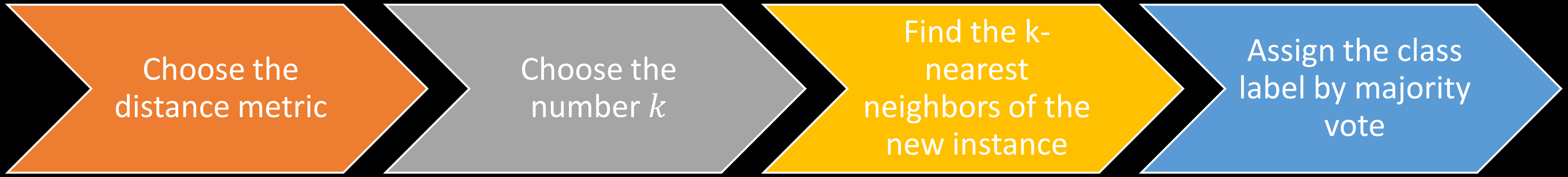
- This is a useful criterion for pruning, but not recommended for growing a decision tree, since it is less sensitive to changes in the class probabilities of the nodes.

Content

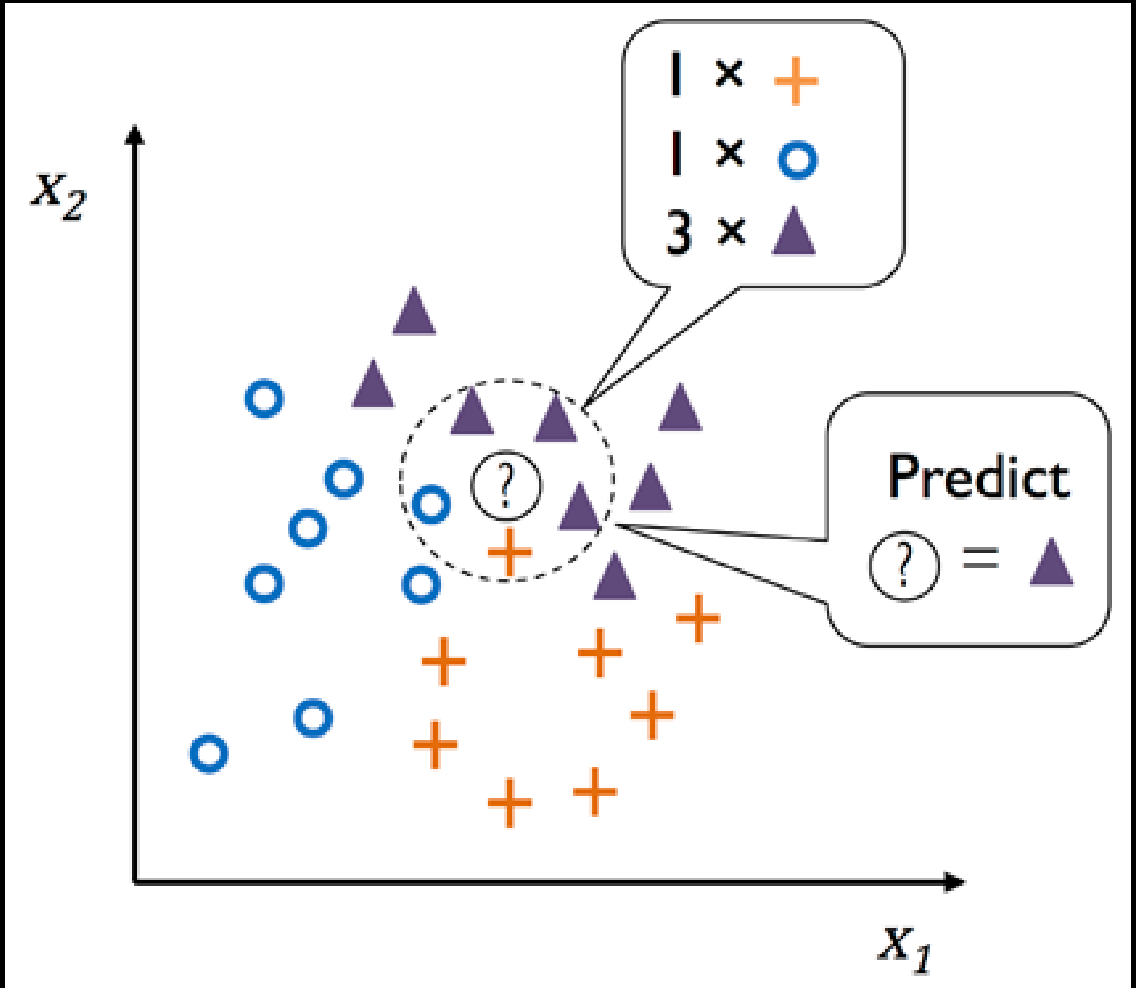
Content
Decision Trees
K-Nearest Neighbors
Logistic Regression

K-Nearest Neighbors

- KNN is a supervised classifier that is referred to as a “lazy learner”.
  - It doesn’t learn a discriminative function from the training data but memorizes the training dataset instead.
- KNN works by computing the distances between the data points and classify the new instance based on the nearest k point.
- Algorithm:



K-Nearest Neighbors



K-Nearest Neighbors

Distance functions

Euclidean $\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$

Manhattan $\sum_{i=1}^{n}|x_i - y_i|$

Minkowski $\left(\sum_{i=1}^{n}(|x_i - y_i|)^q\right)^{1/q}$

Here:

n = no of dimensions (2 for our data)

x = datapoint from dataset

y = new data point(to be predicted)

https://miro.medium.com/max/1400/1*FlMiuoENrq52tMV4S6LSZg.png

Content

Content
Decision Trees
K-Nearest Neighbors
Logistic Regression

Logistic Regression

- Logistic regression is a linear classification model used for binary classification.
    - It also can be extended to multiclass classification.
- Logistic Regression predicts the probability of occurrence of a binary event utilizing a logit function.
- Algorithm:
    1. Compute the net input – apply linear regression equation.
    2. Apply sigmoid function to the net input.
    3. Threshold the probability output to get class label (0 or 1).

Logistic Regression

1. Compute the net input – apply linear regression equation.

$$z = logit(p) = w_1 x_1 + \cdots w_m x_m + b = \sum_{i=j} w_j x_j + b = \boldsymbol{w}^{\boldsymbol{T}} \boldsymbol{x} + b$$

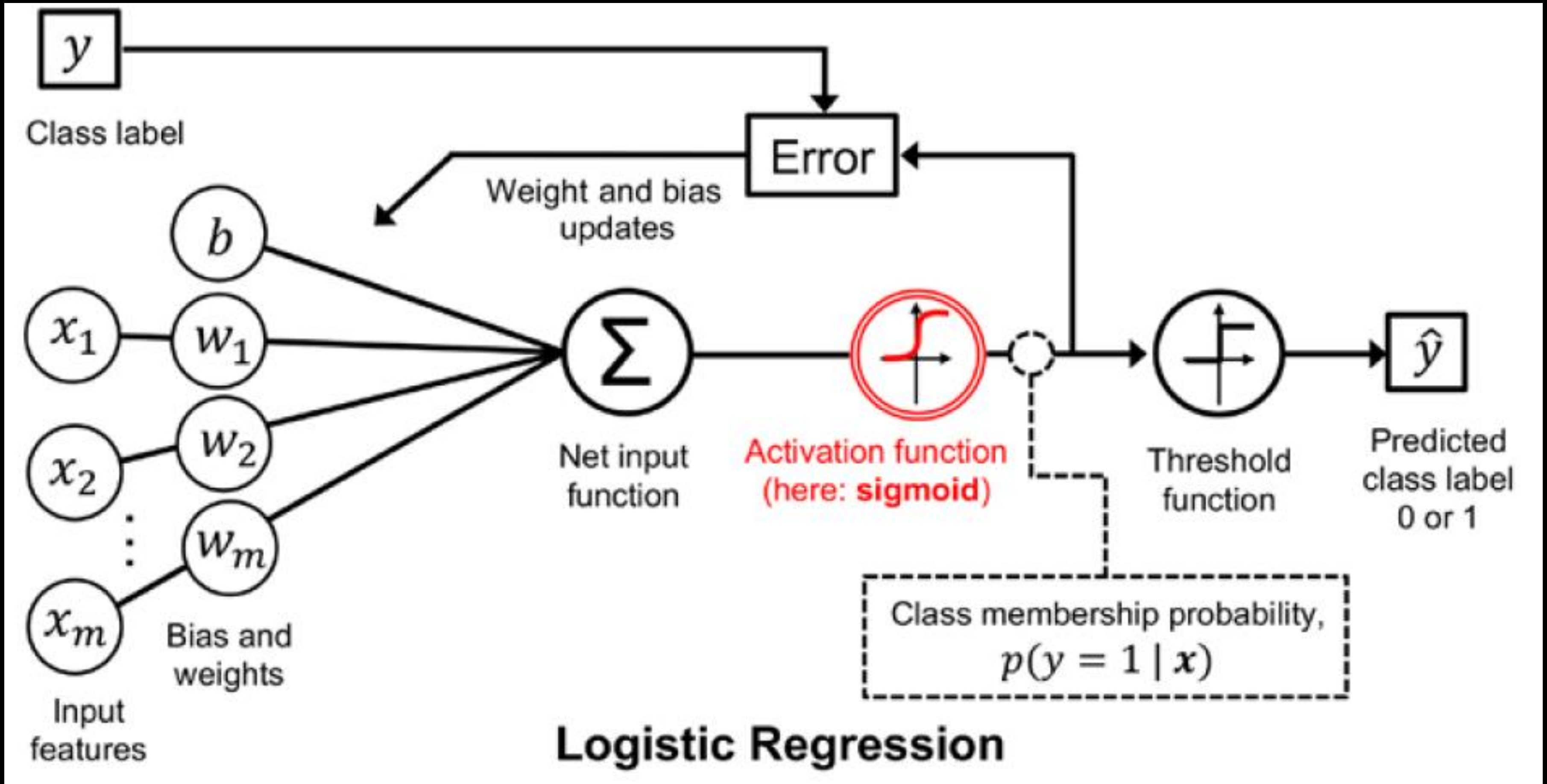
2. Apply sigmoid function to the net input.

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

3. Threshold the probability output to get class label (0 or 1).

$$\hat{y} = \begin{cases} 1 & if\ \sigma(z) \geq 0.5 \\ 0 & otherwise \end{cases}$$

Logistic Regression



TASK

- What are parametric and non-parametric models? Give examples for each.
- What is a Random Forest? What is Bagging? What is Boosting?
- What is XGBoost classifier?

References

- Machine Learning with PyTorch and Scikit-Learn by Sebastian Raschka