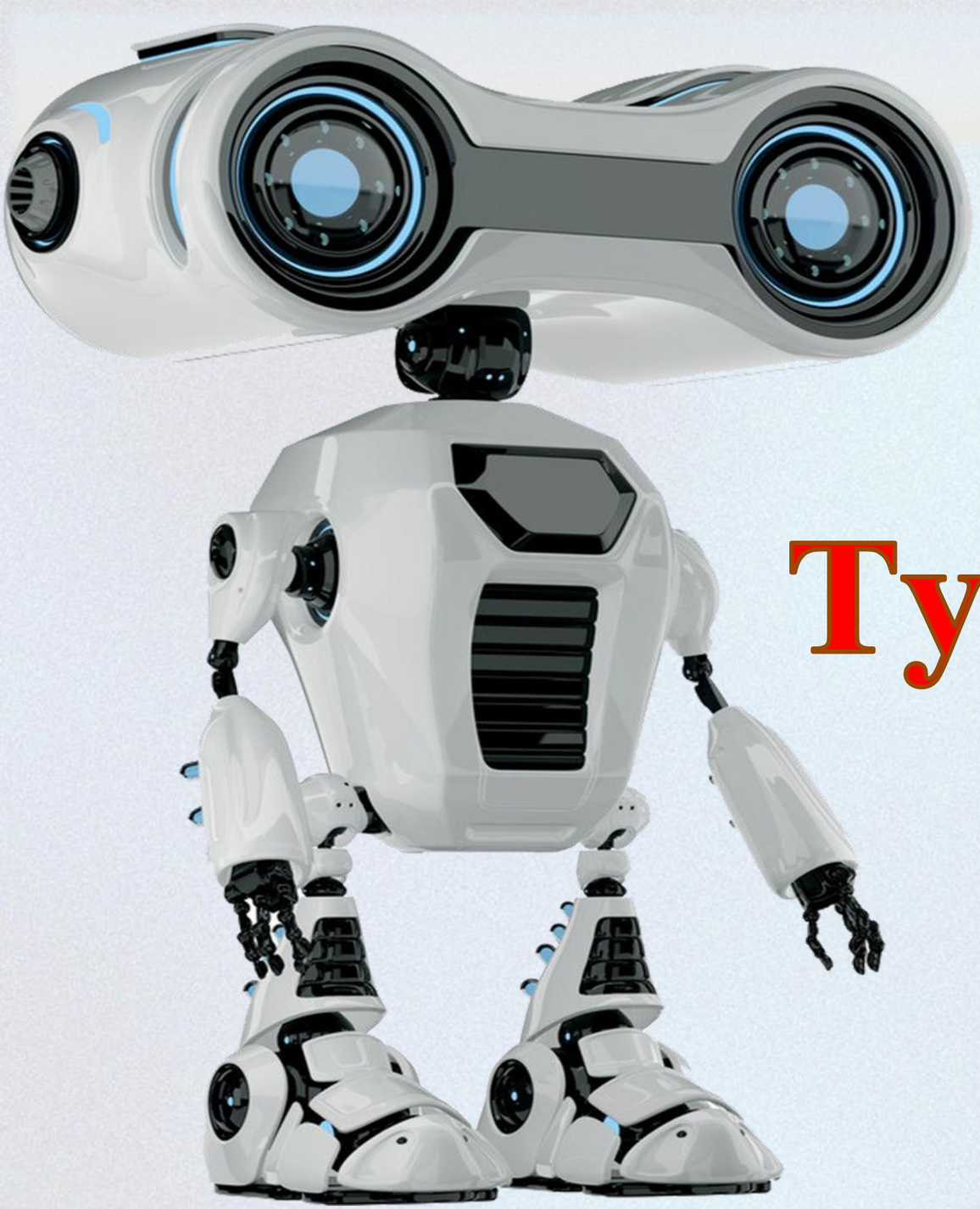
AIE111 :

Artificial Intelligence

Dr. Noha El-Sayad

Lecture 3:

Learning



Type of Learning

Learning Definition

Learning in AI (particularly machine learning (ML) models) refers to how artificial intelligence systems acquire knowledge or improve performance from data or experience, rather than being explicitly programmed.

The Difference between machine learning algorithms and machine learning models, which are often mistakenly used interchangeably.

- ***Machine Learning Algorithm***: This is the *method* - a set of mathematical procedures or rules (e.g., linear regression, decision tree, gradient descent) - that guides how learning from data should occur.
- ***Machine Learning Model:*** This is the *result* of applying a machine learning algorithm to a dataset. It is the trained artifact that can make **predictions or classifications** based on new data.

Think of it this way:

- **Algorithm = instructions for cooking**
- **Data = raw ingredients**
- **Model = final dish**

The **algorithm** processes the **data** to produce a **model**.

Type of Learning

- **Supervised Learning**
  - **Definition**: The model learns from a **labeled** dataset (i.e., input-output pairs).
  - **Examples**: Email spam detection, image classification, disease diagnosis.
  - **Algorithms**: Linear regression, decision trees, support vector machines, neural networks.

- **Unsupervised Learning**
  - **Definition**: The model finds patterns or structures in **unlabeled data**.
  - **Examples**: Customer segmentation, anomaly detection, topic modeling.
  - **Algorithms**: K-means clustering, PCA, autoencoders.

- **Semi-Supervised Learning**
  - **Definition**: Combines a small amount of **labeled data with a large amount of unlabeled data.**
  - **Example**: Improving performance when labeling data is expensive or time-consuming.

- **Reinforcement Learning (RL)**
  - **Definition**: An agent learns to make decisions by **interacting with an environment and receiving rewards/punishments**.
  - **Examples**: Game playing (like AlphaGo), robotics, self-driving cars.
  - **Key terms**: Agent, environment, reward, policy, value function.

Learning Definition

Machine learning models and their training algorithms

Supervised learning

Data scientists provide input, output and feedback to build model (as the definition).

EXAMPLE ALGORITHMS:

Linear regressions
- Sales forecasting.
- Risk assessment.

Support vector machines
- Image classification.
- Financial performance comparison.

Decision trees
- Predictive analytics.
- Pricing.

Unsupervised learning

Use deep learning to arrive at conclusions and patterns through unlabeled training data.

EXAMPLE ALGORITHMS:

Apriori
- Sales functions.
- Word associations.
- Searcher.

K-means clustering
- Performance monitoring.
- Searcher intent.

Artificial neural networks
- Generate new, synthetic data.
- Data mining and pattern recognition.

Semi-supervised learning

Builds a model through a mix of labeled and unlabeled data, a set of categories, suggestions and exampled labels.

EXAMPLE ALGORITHMS:

Generative adversarial networks
- Audio and video manipulation.
- Data creation.

Self-trained Naïve Bayes classifier
- Natural language processing.

Reinforcement learning

Self-interpreting but based on a system of rewards and punishments learned through trial and error, seeking maximum reward.

EXAMPLE ALGORITHMS:

Q-learning
- Policy creation.
- Consumption reduction.

Model-based value estimation
- Linear tasks.
- Estimating parameters.

Supervised Learning

- Prediction
- Classification (discrete labels), Regression (real values)
  - Neural Networks
  - Deep Neural Network
  - Decision Trees
  - Ensemble Tree Methods

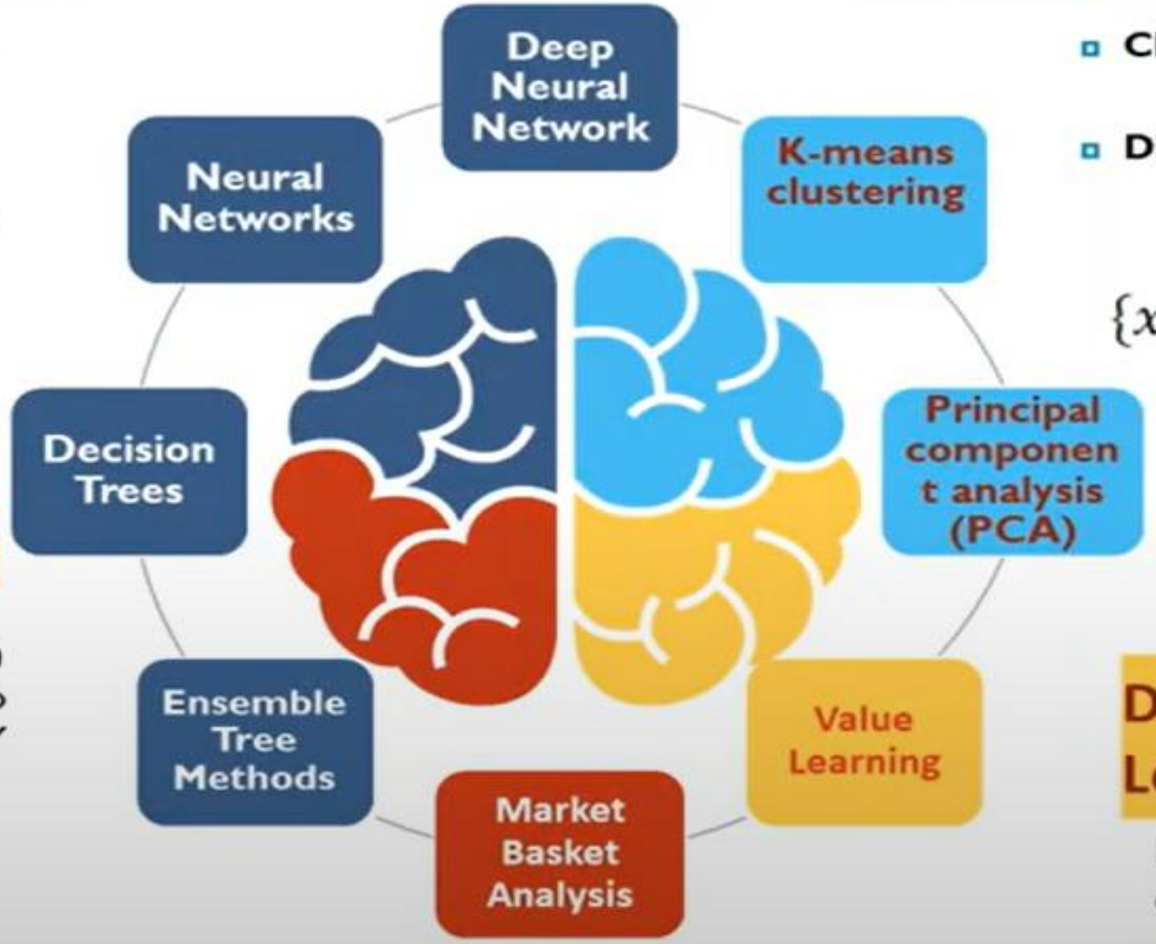
$$\{x_n \in R^d, y_n \in R\}_{n=1}^N$$

Association Analysis

Basket analysis:

P (Y | X)

probability that somebody who buys X also buys Y where X and Y are products/services.



Non Supervised Learning

- Clustering
  - K-means clustering
- Dimension reduction
  - Principal component analysis (PCA)

$$\{x_n \in R^d\}_{n=1}^N$$

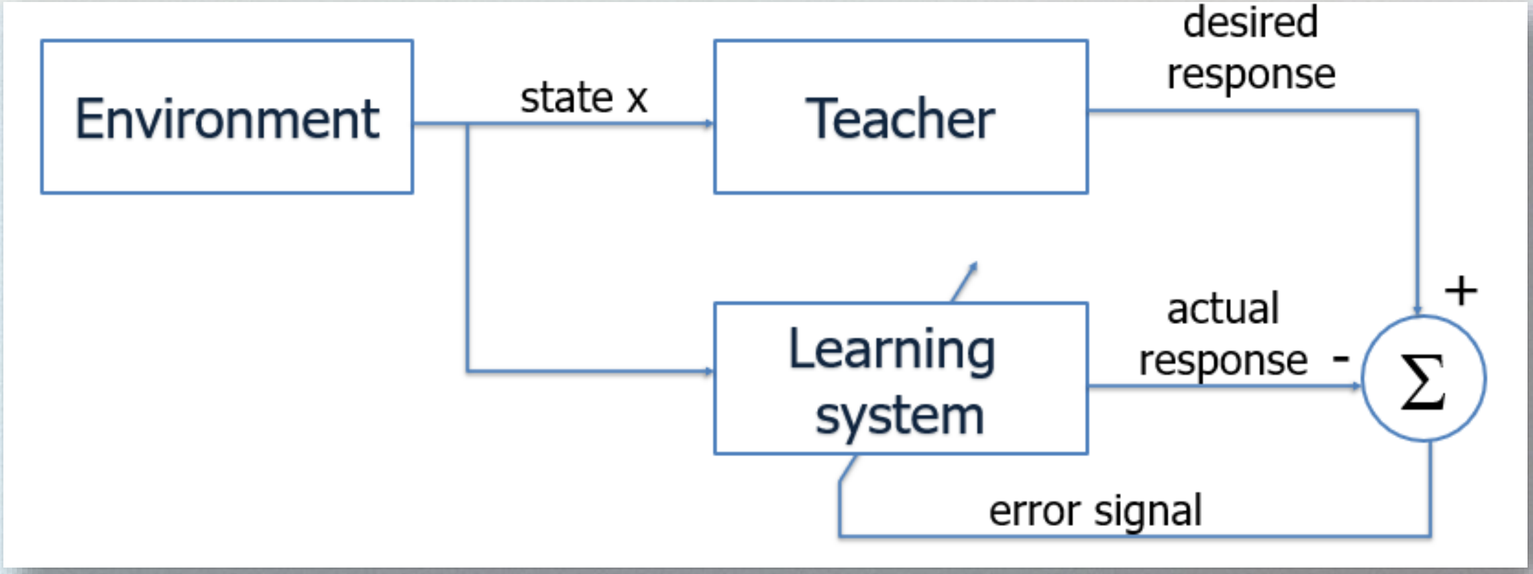
Deep Reinforcement Learning

Decision making (robot, chess machine)

- Game Playing
- Robot in Maze

supervised learning

- Knowledge represented by a set of **input-output examples (x_i, y_i)**
- Minimize the error between the actual response of the learner and the desired response



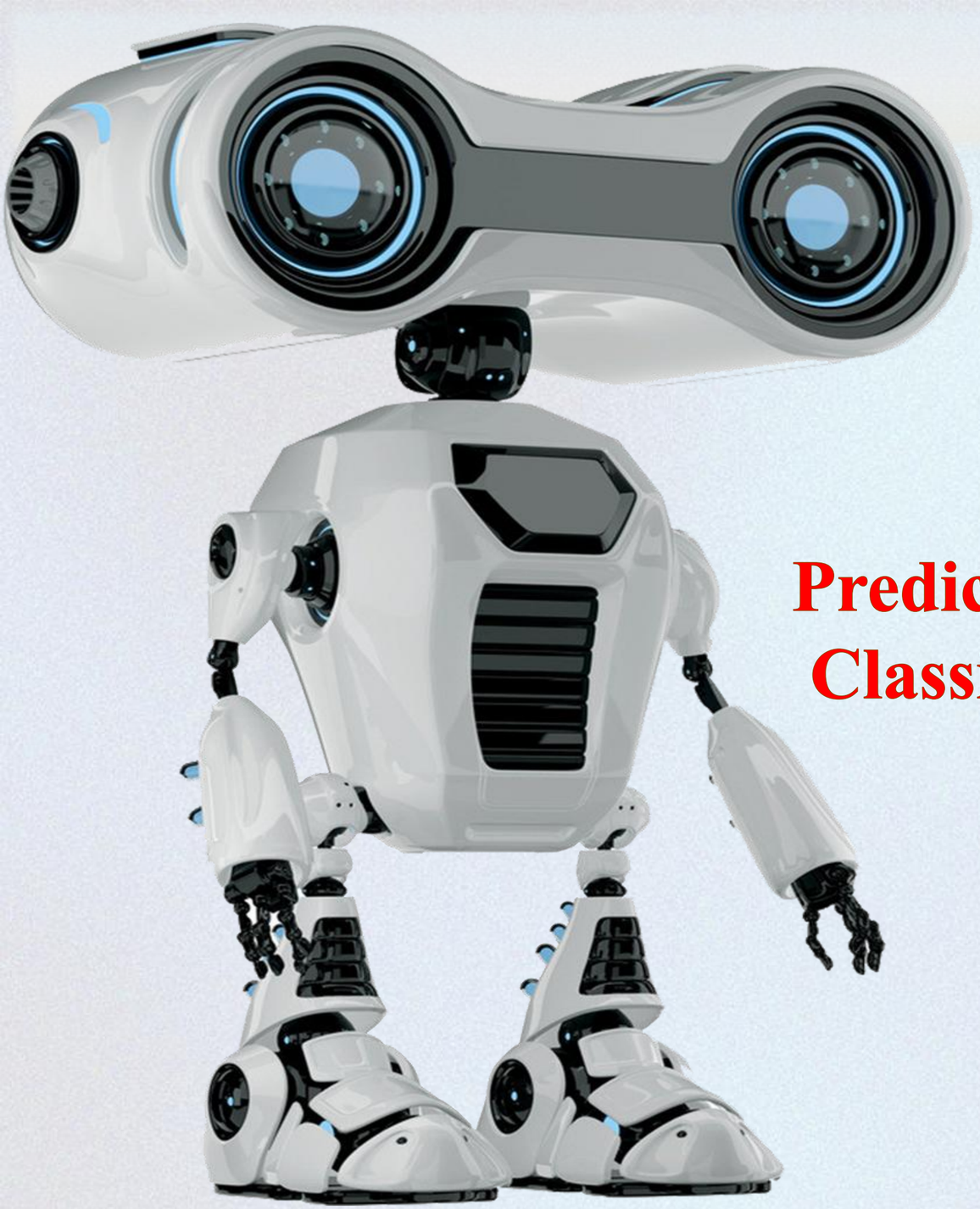
Unsupervised learning

- Self-organized learning
- No teacher
- Task independent quality measure
- Identify regularities in the data and discover classes automatically

Learning Definition

Objective: Learning an Unknown Function

- **Problem Setup**
  - We are given a training set of input–output pairs: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
  - These pairs are generated by an unknown function: $y = f(x)$
- **Goal**
  - Discover a function $\hat{f}(x)$ that approximates the true function $f(x)$.
  - The outputs y are referred to as the ground truth.
- **Key Concepts**
  - Algorithm: Learns from training data to build the model.
  - Model: A learned approximation of f(x) that maps inputs x to outputs $\hat{y}$.
  - Training Data: Used to build the model.
  - Test Data: Used to evaluate model's performance.
- **Evaluation**
  - After training, we use a test set to evaluate generalization on unseen data.



Prediction Problems:
Classification vs. Numeric Prediction

Classification

- The output variable is **categorical** (discrete/ nominal classes) Labels.
- classifies data based on the training set and the values (class labels) in a classifying attribute and uses it in classifying new data.
- **Example**: Email → {Spam, Not Spam}
- **Algorithms**: Decision Trees, SVM, k-NN, Neural Networks

Numeric Prediction (Regression)

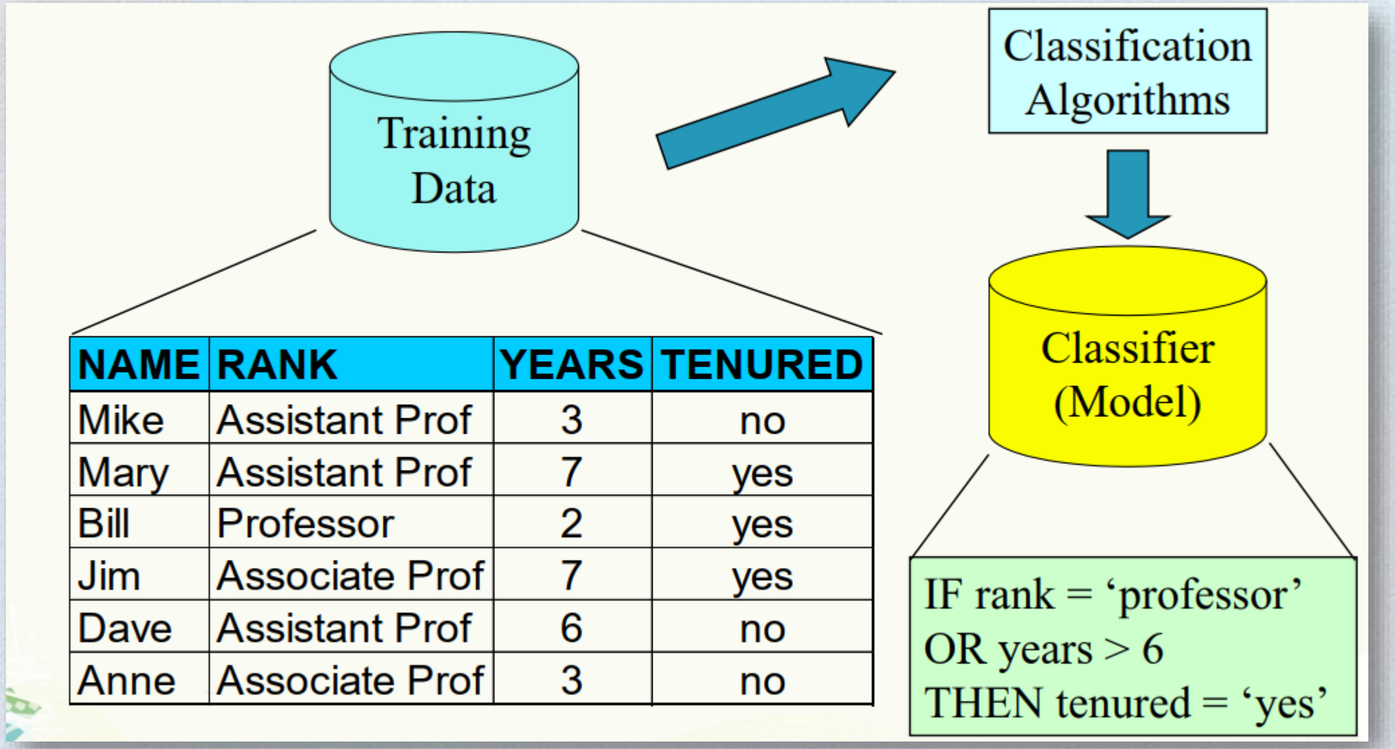
- The output variable is **continuous** (real numbers).
- Goal: Predict a unknow or missing numerical value from input features.
- **Example**: Predicting house prices, temperature, or stock prices.
- **Algorithms**: Linear Regression, SVR, Random Forest Regression

🧠 **Key Difference**
- Classification → Discrete labels
- Prediction → Continuous values

Classification – A Two-Step Process

Step 1: Model Construction

- Use a training set with known class labels (class label attribute)
  Each example belongs to one of the predefined classes
- Build a model using rules, decision trees, or formulas

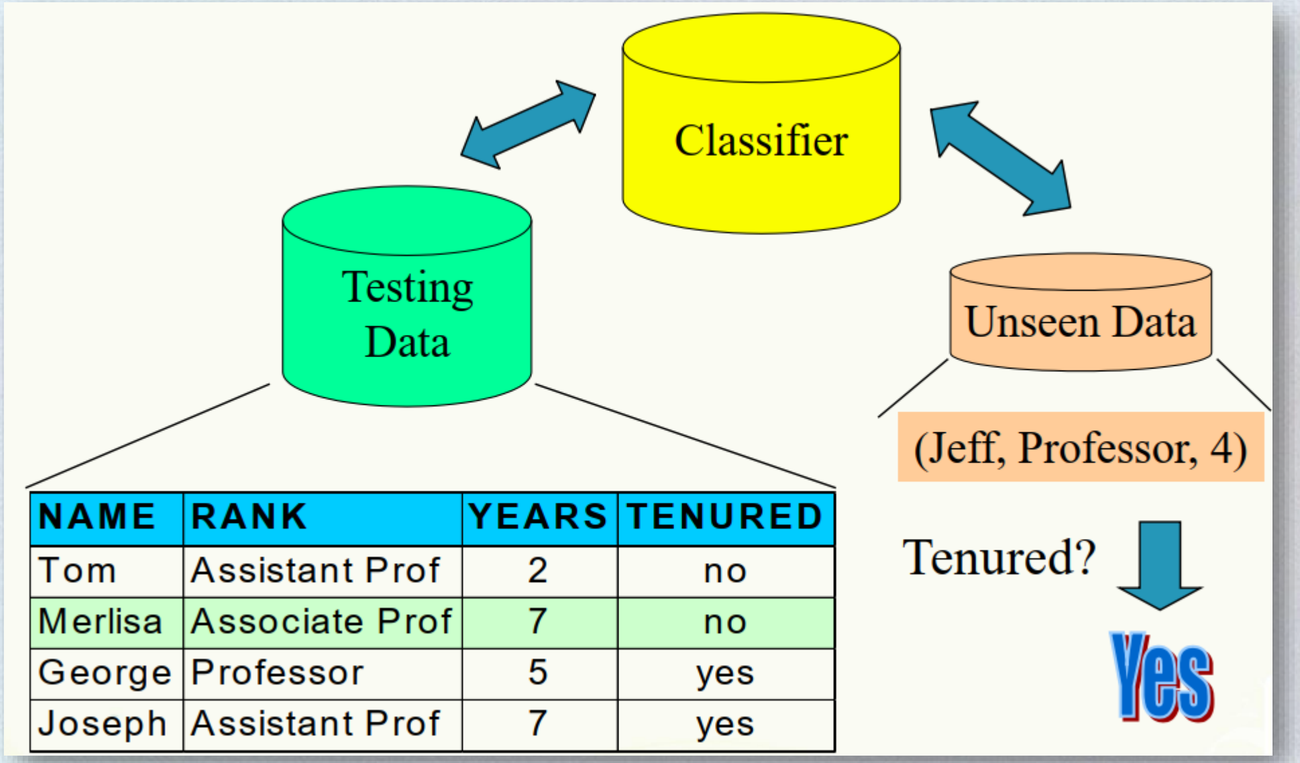


NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

Classification – A Two-Step Process

Step 2: Model Usage

- Use the model to classify new, unseen data
- Compare model output with real labels from test set
- Accuracy = % of correct predictions
- Test set must be different from training set (to avoid overfitting)
- If accuracy is good, use the model in real-world cases
- If test data is used to choose the model, it's called a validation set



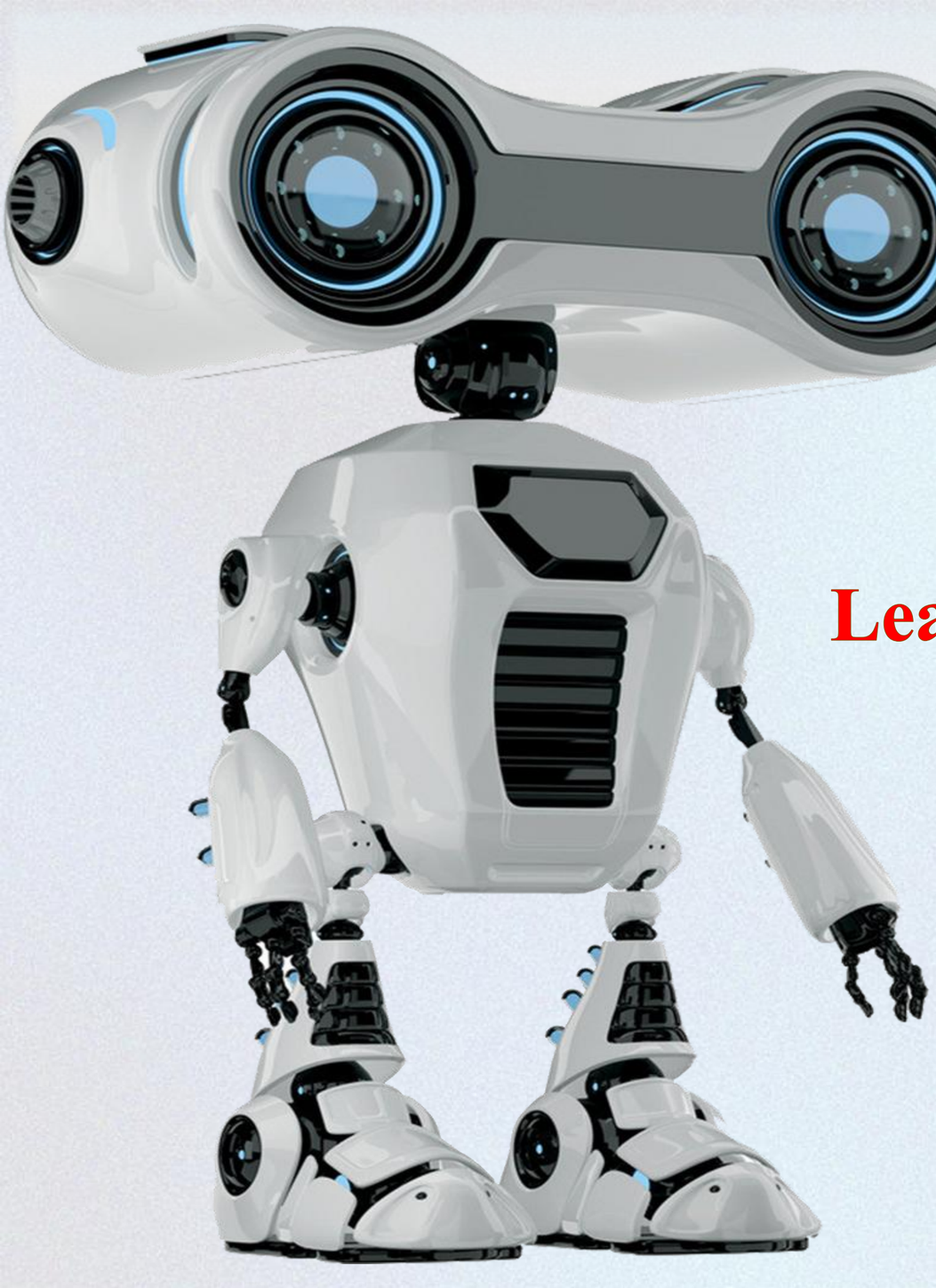
NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

Test

Training and Validation

Data Preprocessing

Read input Data



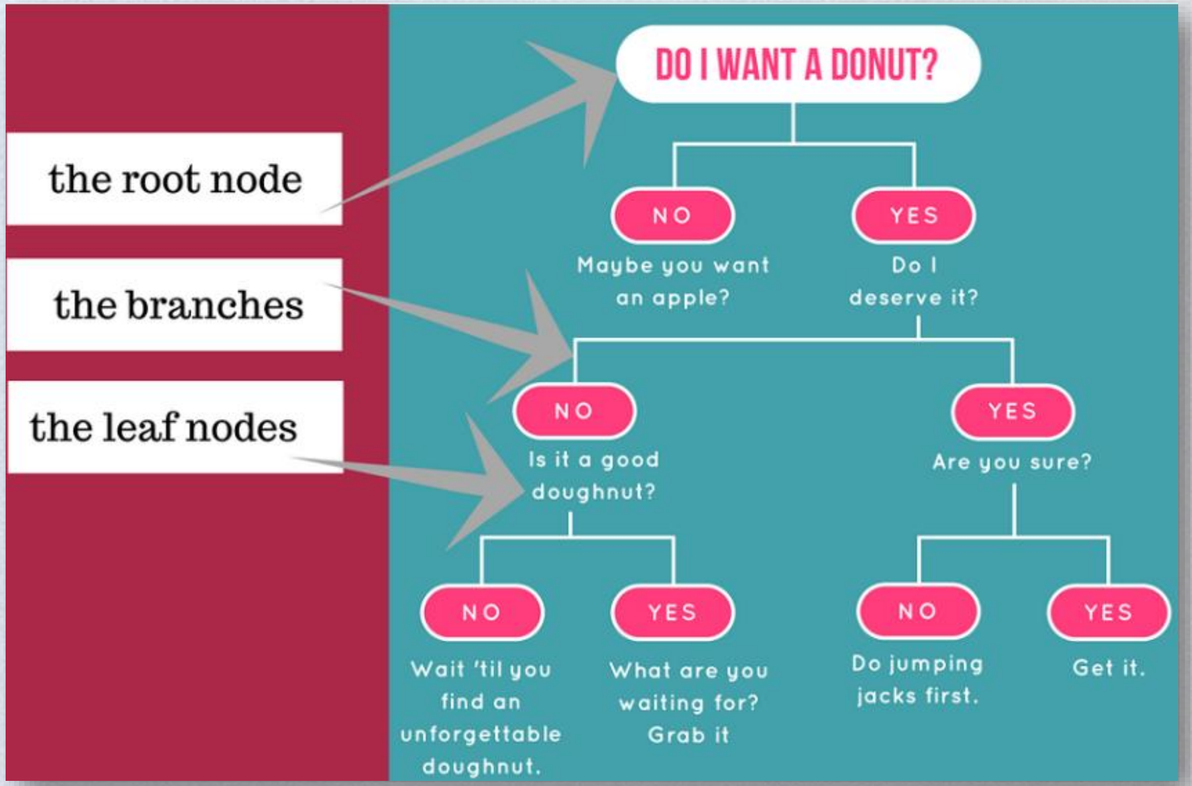
Learning Decision Trees

Decision Tree

- ✓ A **decision tree** is a model that helps make decisions by learning from data.
- ✓ It takes input values (called attributes) and gives one output (a decision).
- ✓ The process starts at the **root** and follows a path of **yes/no questions** (tests).
- ✓ At each step, it chooses a branch based on the answer, until it reaches a **leaf** (the final decision).

The Structure of Decision Tree

- ✓ diagram representation of possible solutions to a decision.
- ✓ It shows different outcomes from a set of decisions.
- ✓ The diagram is a widely used decision-making tool for analysis and planning.

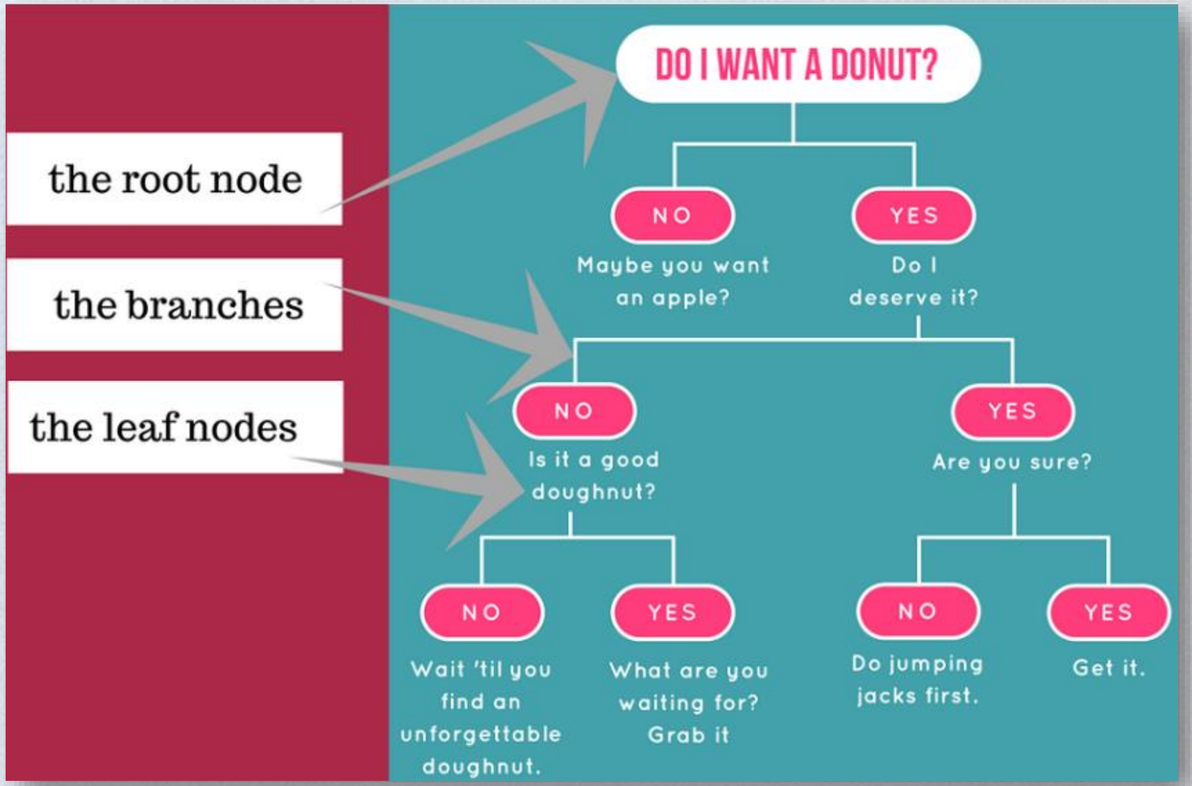


Decision Tree

- ✓ A **decision tree** is a model that helps make decisions by learning from data.
- ✓ It takes input values (called attributes) and gives one output (a decision).
- ✓ The process starts at the **root** and follows a path of **yes/no questions** (tests).
- ✓ At each step, it chooses a branch based on the answer, until it reaches a **leaf** (the final decision).

The Structure of Decision Tree

- ✓ diagram representation of possible solutions to a decision.
- ✓ It shows different outcomes from a set of decisions.
- ✓ The diagram is a widely used decision-making tool for analysis and planning.



Popular because very flexible and easy to interpret.

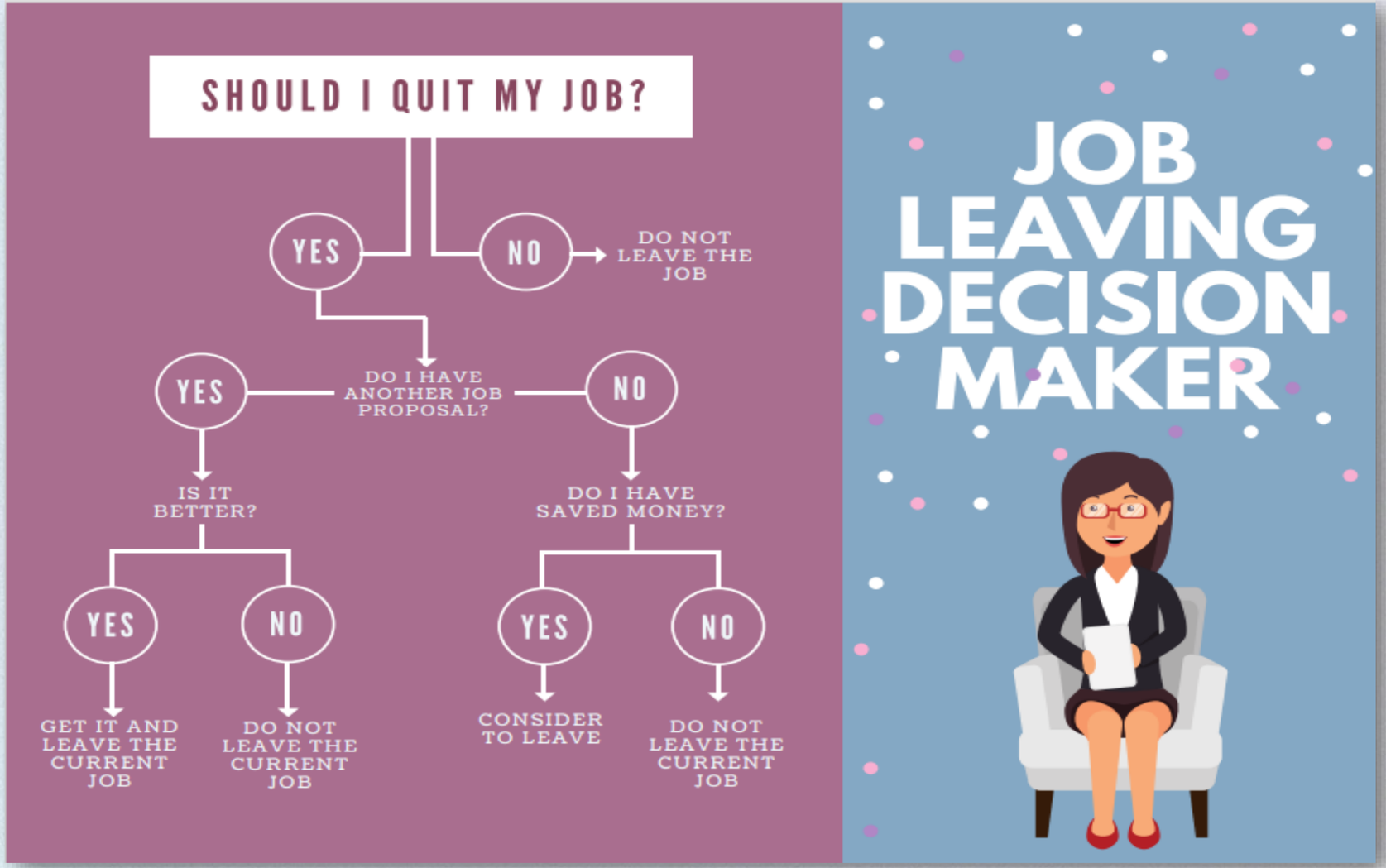
Learning a decision tree =

finding a tree with small error on the training set.

Steps

1. Start with the root node.
2. At each step split one of the leaves
3. Repeat until a termination criterion

Simple example



Which node to split?

We want the children to be more "pure" than the parent.

Example:

- Parent node is 50%+, 50%-.
- Child nodes are (90%+,10%-),(10%+,90%-)

How can we quantify improvement in purity?

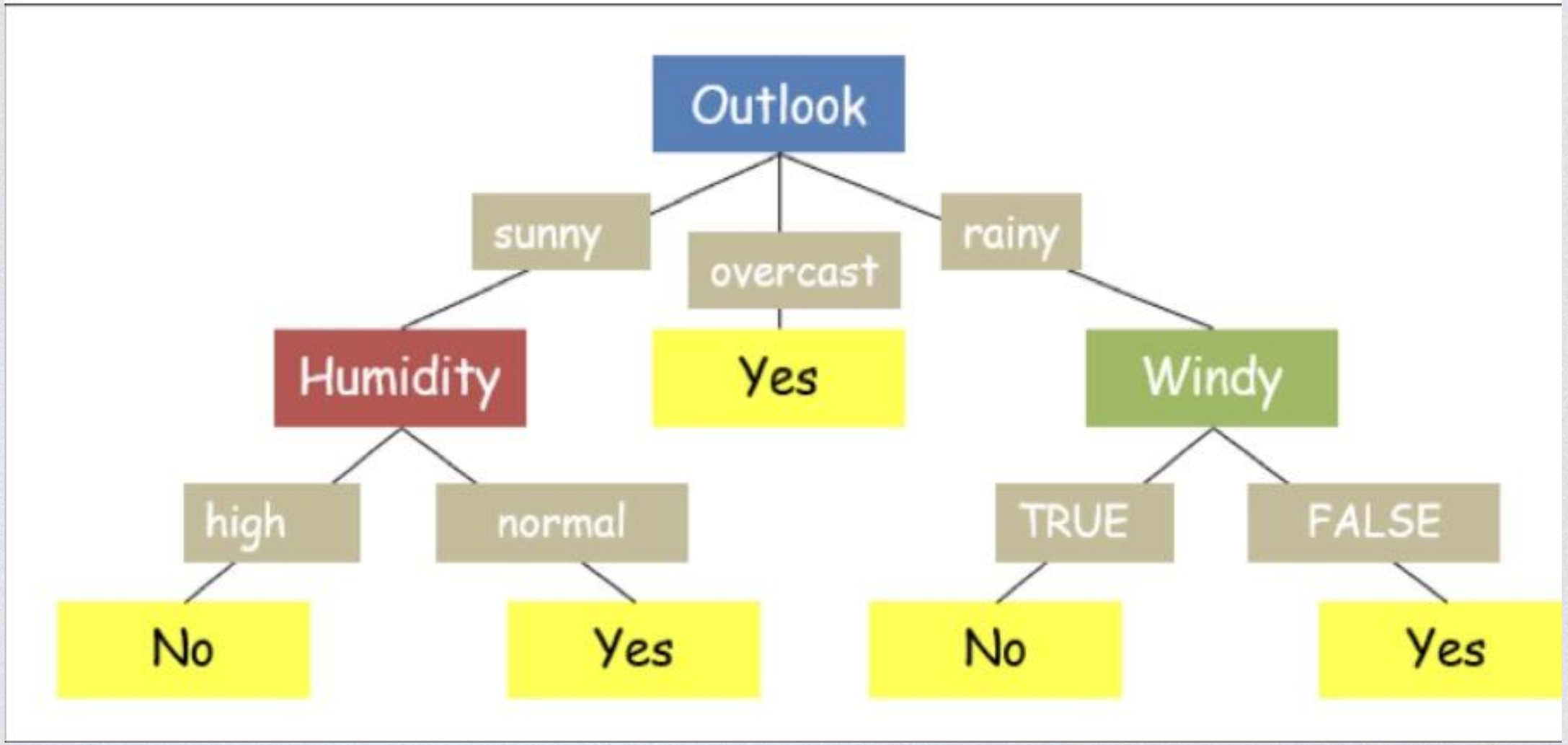
How to determine the Best Split

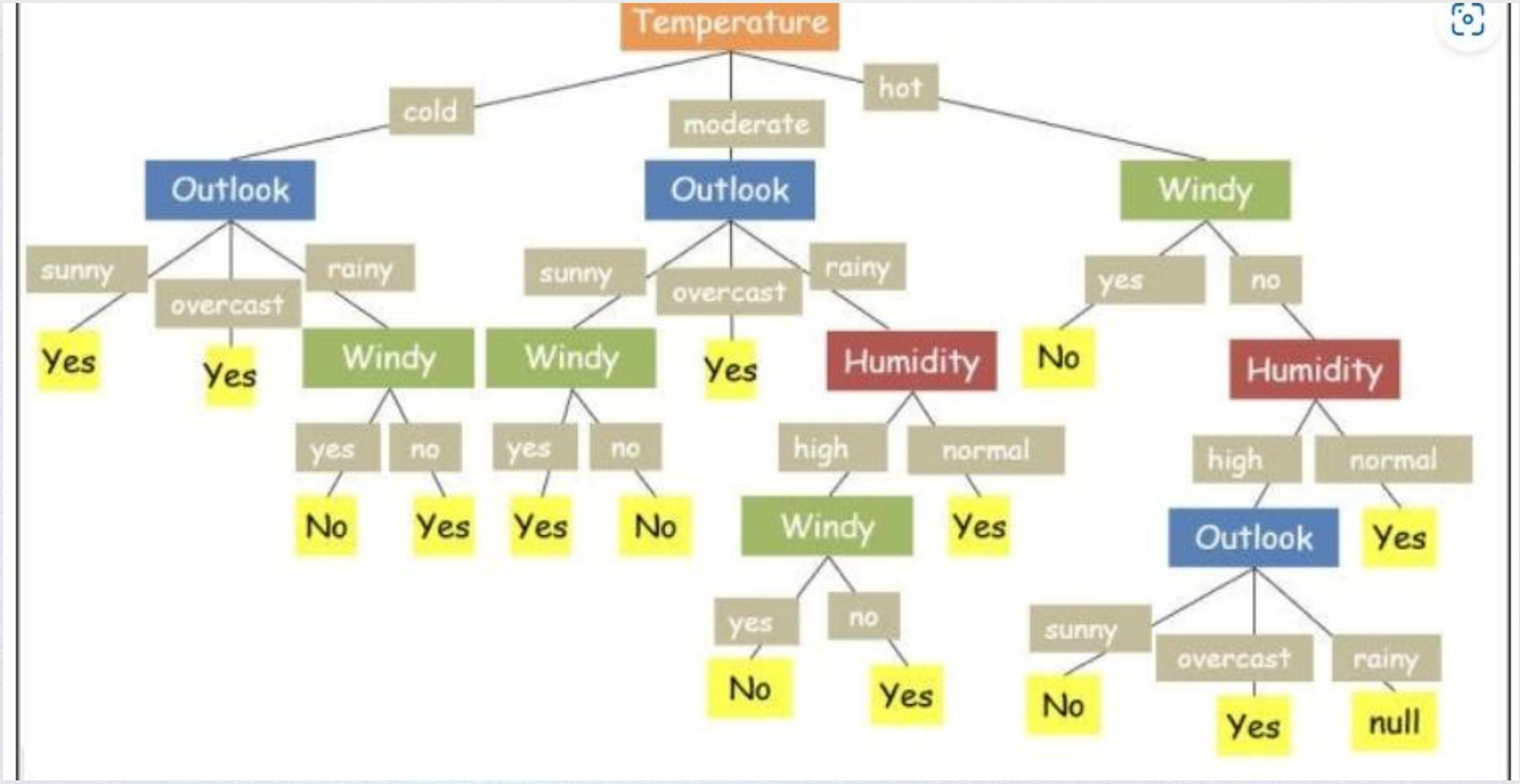
Will Nadal Play Tennis?

Rafael Nadal

Features

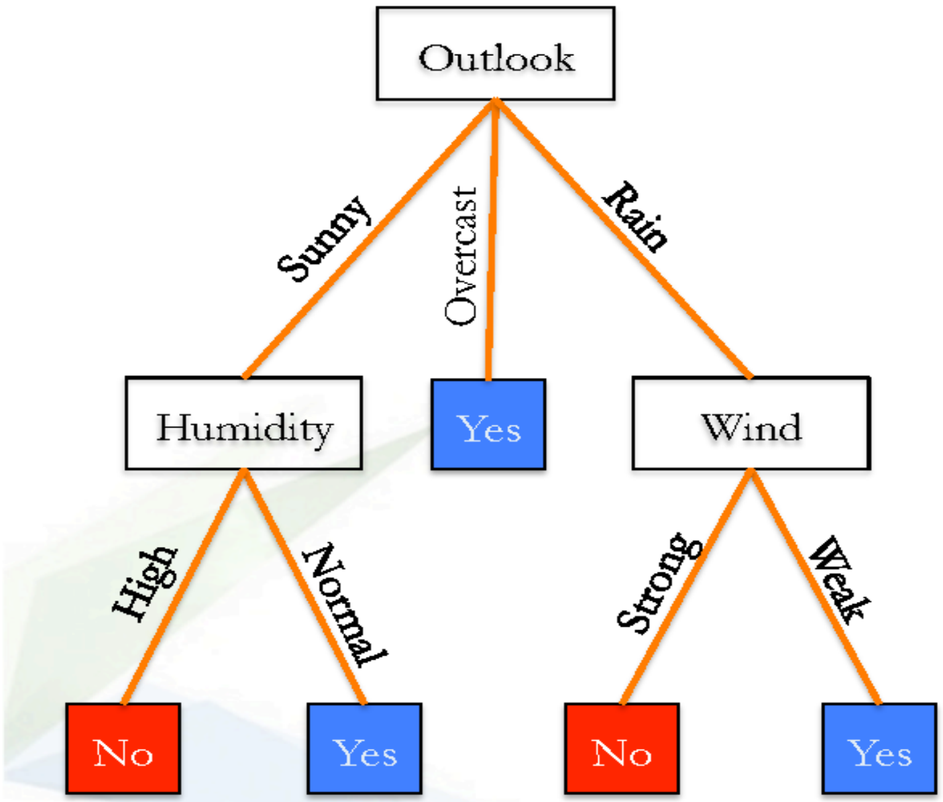
Day	Outlook	Temp	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Weak	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Strong	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No


Outlook
sunny
overcast
rainy
Humidity
Yes
Windy
high
normal
TRUE
FALSE
No
Yes
No
Yes


Temperature
cold
moderate
hot
Outlook
Outlook
Windy
sunny
overcast
rainy
sunny
overcast
rainy
yes
no
Yes
Yes
Windy
Windy
Yes
Humidity
No
Humidity
yes
no
yes
no
high
normal
high
normal
No
Yes
Yes
No
Windy
Yes
Outlook
Yes
yes
no
sunny
overcast
rainy
No
Yes
No
Yes
null

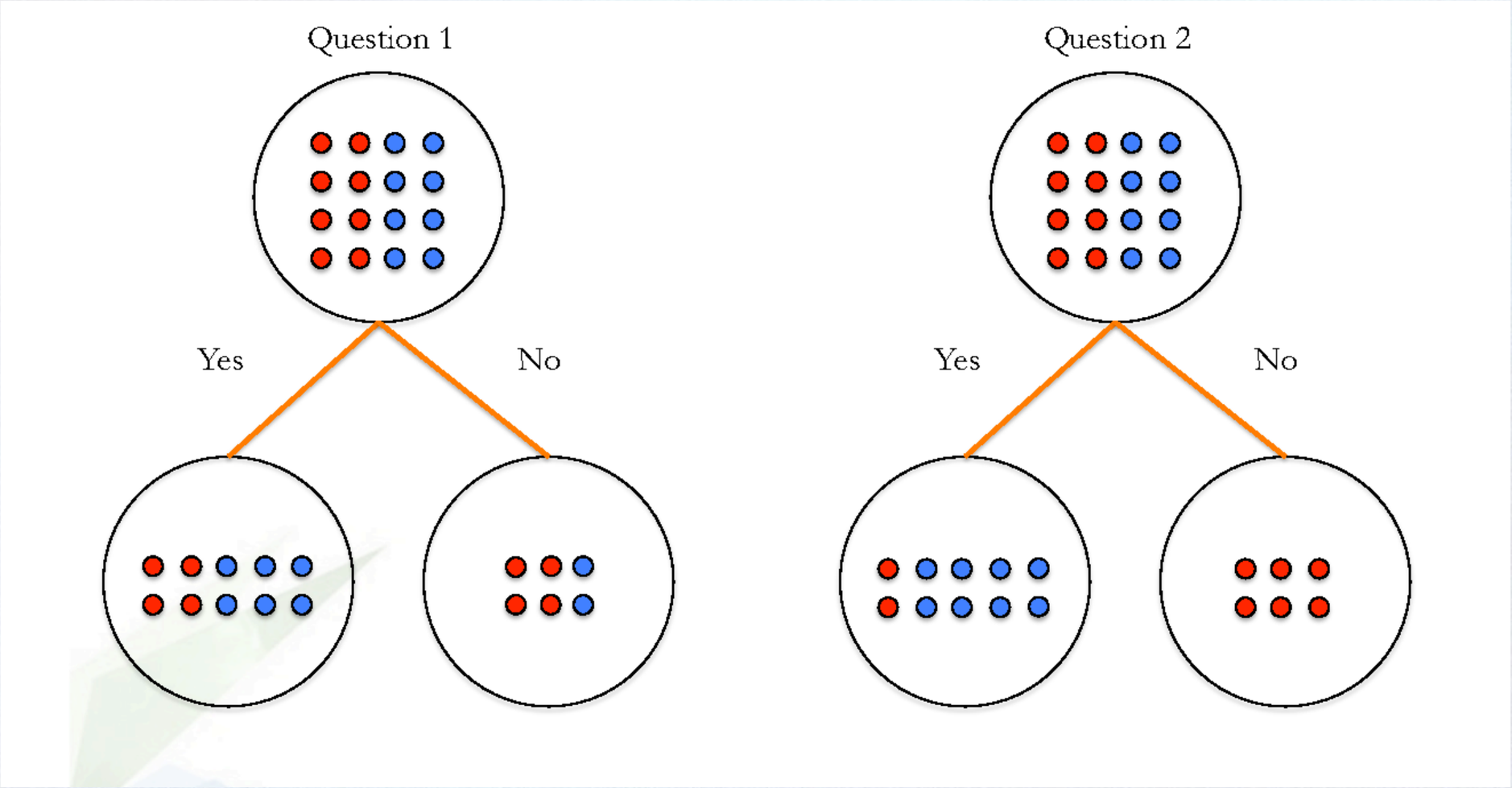
How to determine the Best Split

Will Nadal Play Tennis?



Day	Outlook	Temp	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Weak	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Strong	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

How to determine the Best Split



How to determine the Best Split

1. **Entropy:** measures the degree of uncertainty, impurity, or disorder.

Entropy

$$Entropy = -\sum_{i=0}^{c-1} p_i(t) log_2 p_i(t)$$

Entropy

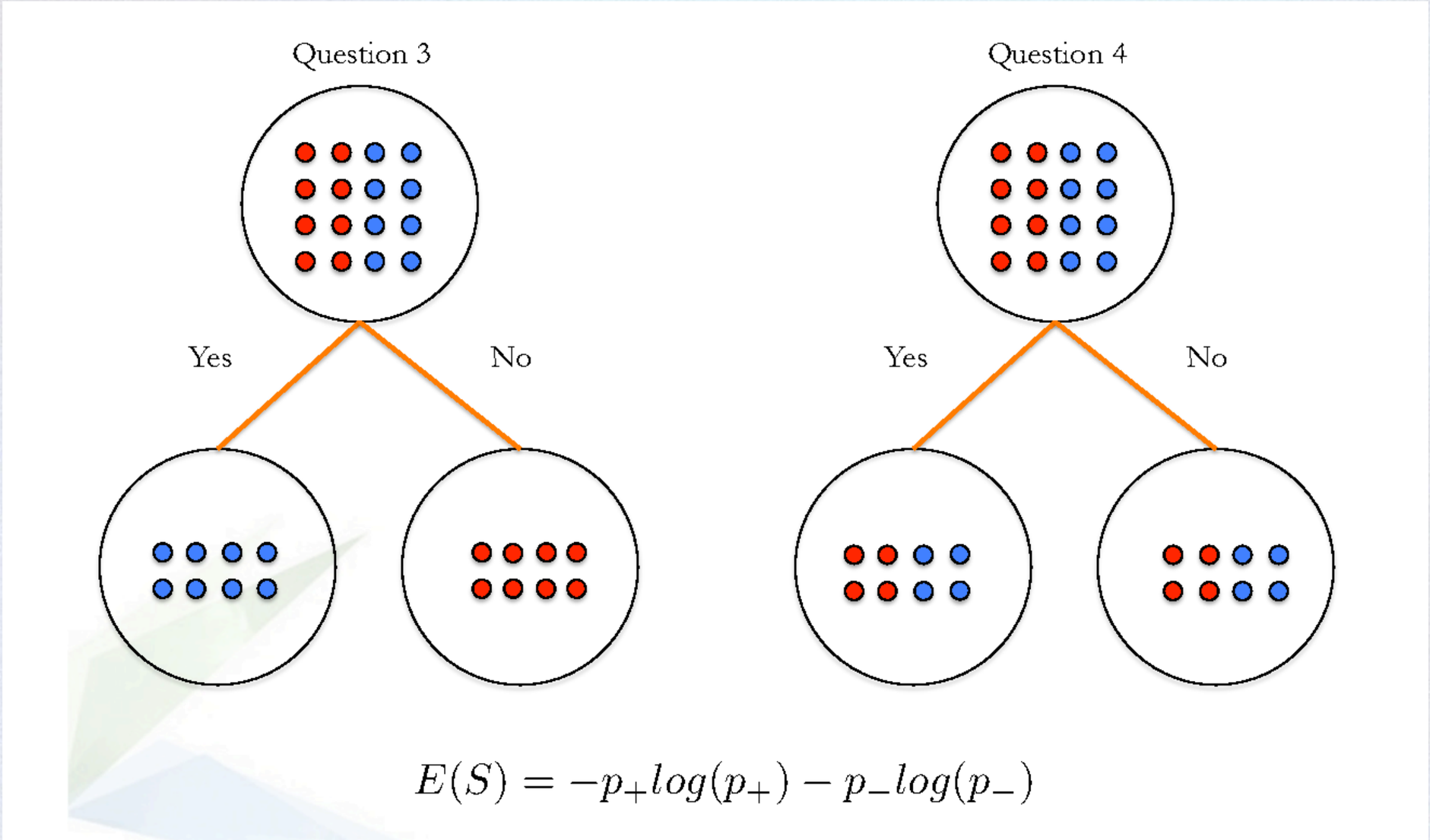
Entropy at a given node $\boldsymbol{t}$

$$Entropy = -\sum_{i=0}^{c-1} p_i(t) log_2 p_i(t)$$

Where $\boldsymbol{p_i(t)}$ is the frequency of class $\boldsymbol{i}$ at node $\boldsymbol{t}$, and $\boldsymbol{c}$ is the total number of classes

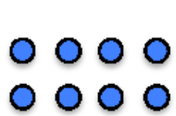
- Maximum of $\log_2 c$ when records are equally distributed among all classes, implying the least beneficial situation for classification
- Minimum of 0 when all records belong to one class, implying most beneficial situation for classification

How to determine the Best Split

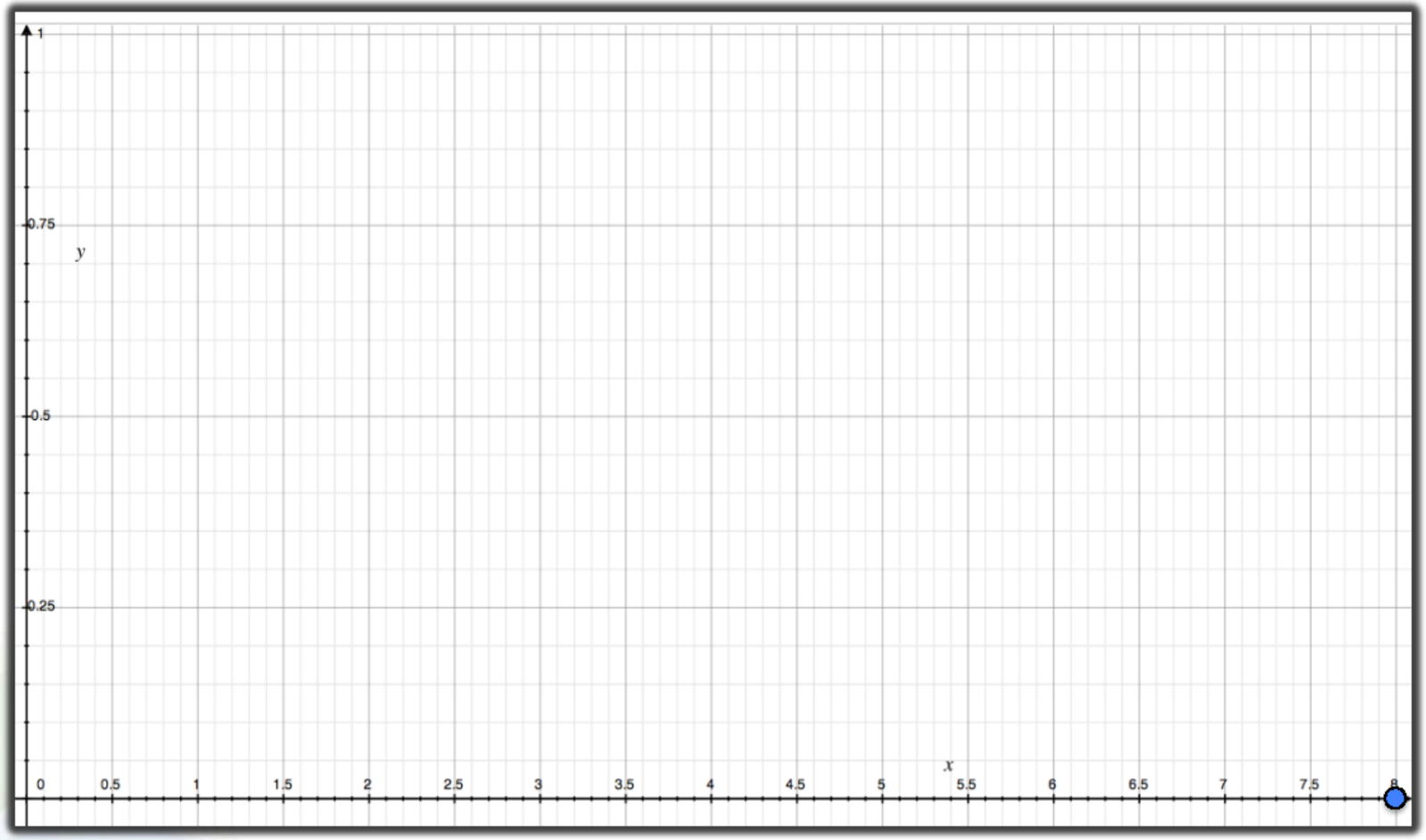


$$E(S) = -p_+ log(p_+) - p_- log(p_-)$$

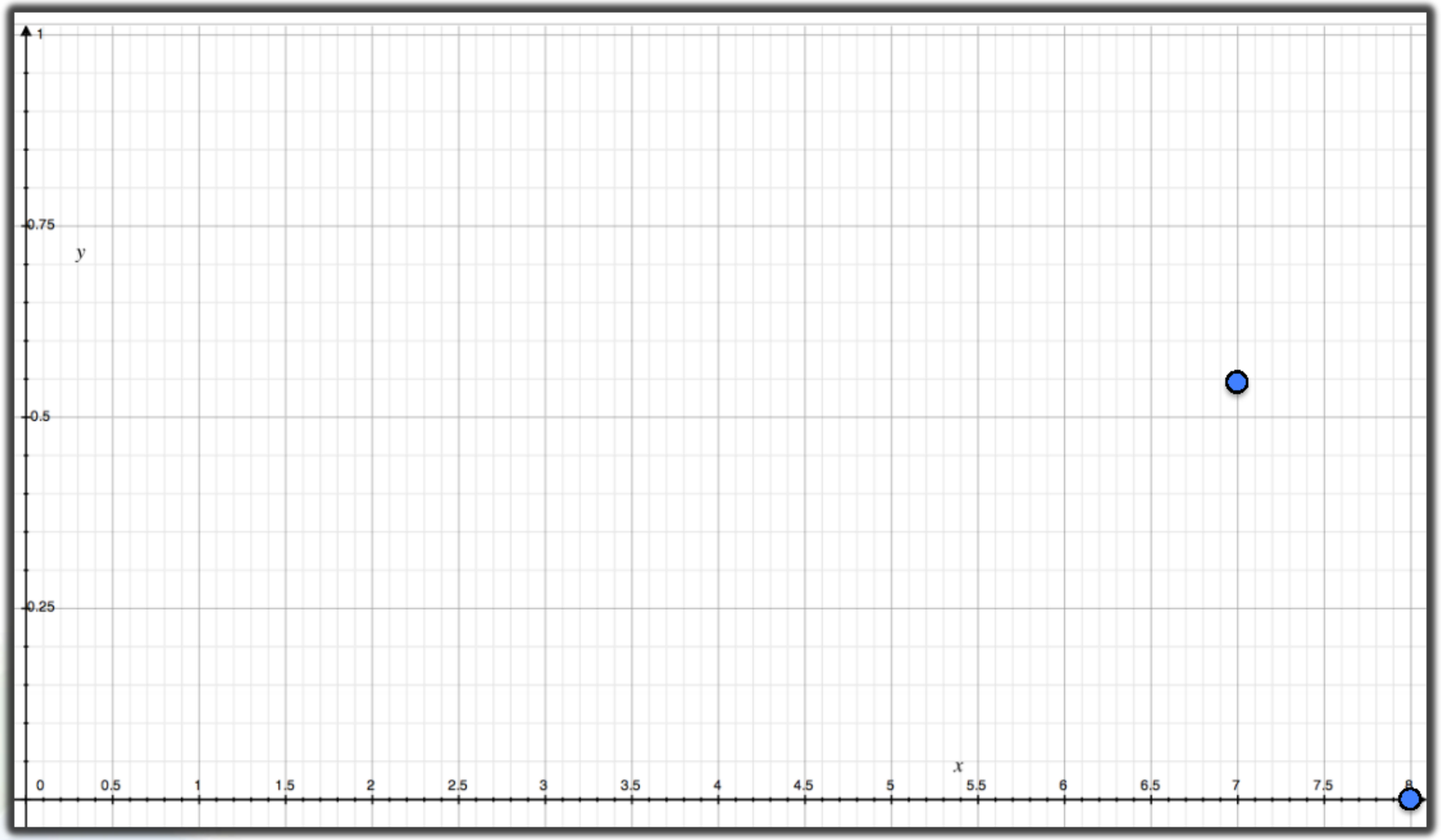
How to determine the Best Split



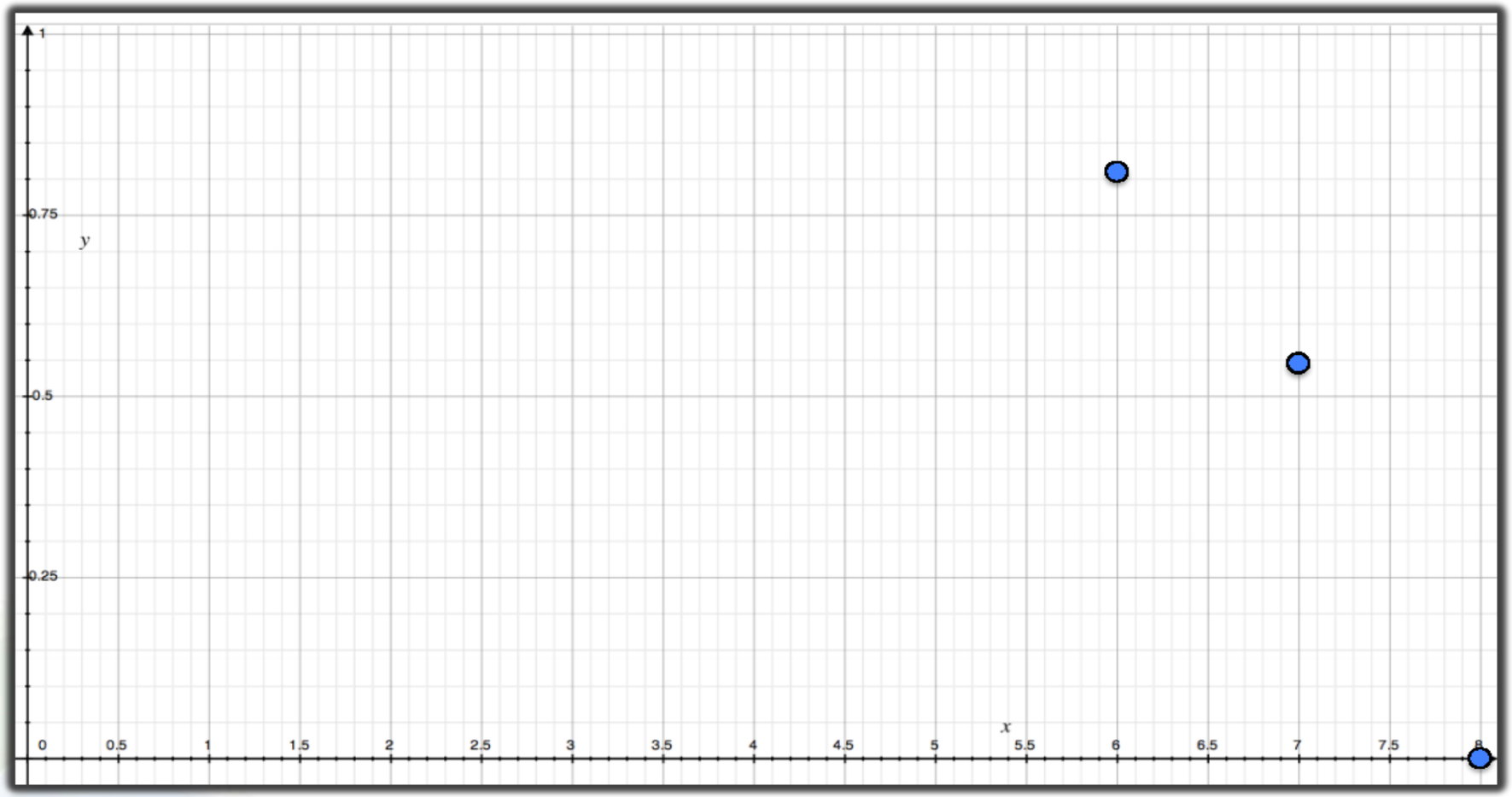
$$-\left(\frac{8}{8}\right) log_2 \left(\frac{8}{8}\right) - \left(\frac{0}{8}\right) log_2 \left(\frac{0}{8}\right) = 0$$



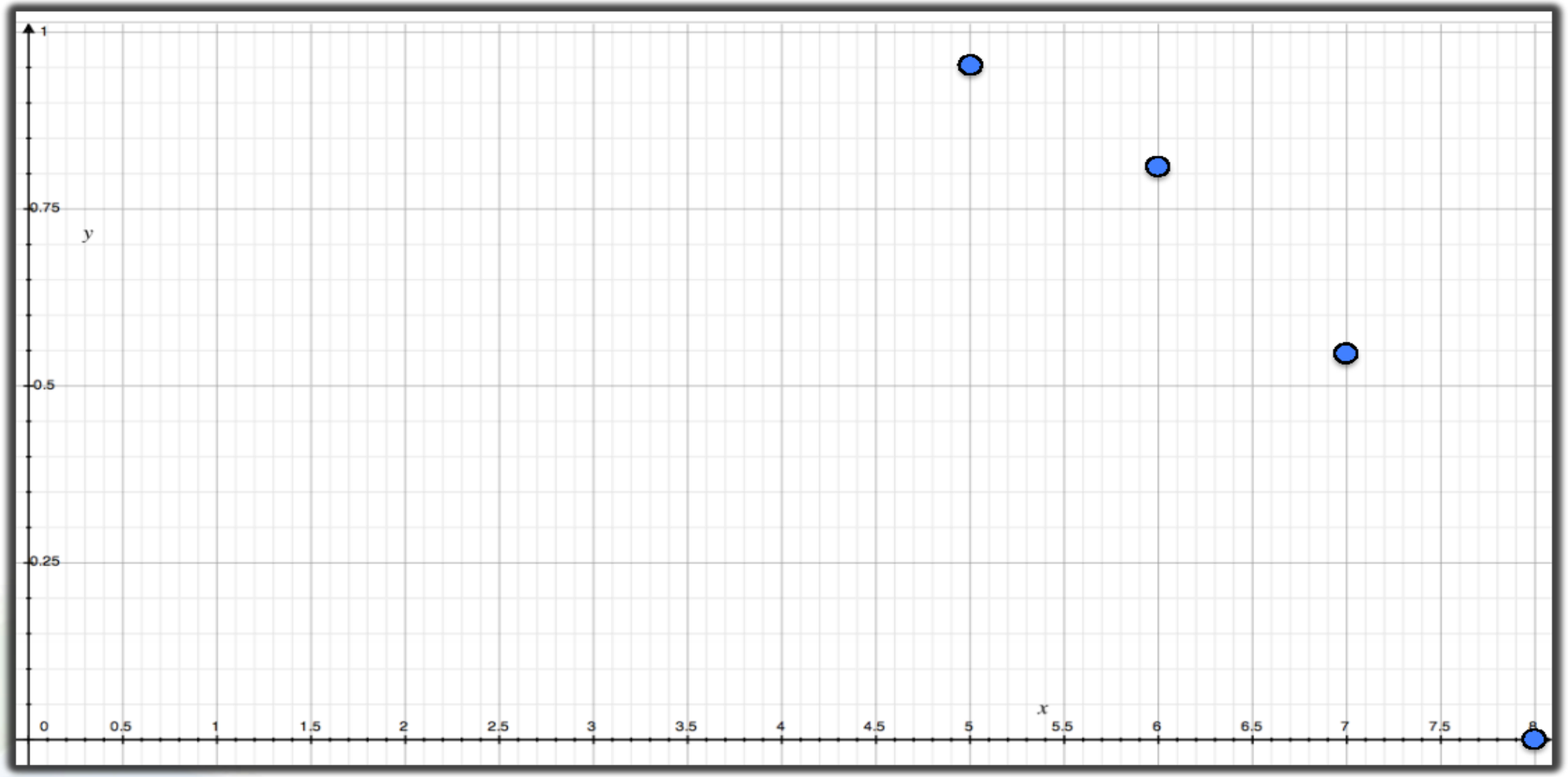
$$-\left(\frac{7}{8}\right) log_2 \left(\frac{7}{8}\right) - \left(\frac{1}{8}\right) log_2 \left(\frac{1}{8}\right) = 0.54$$



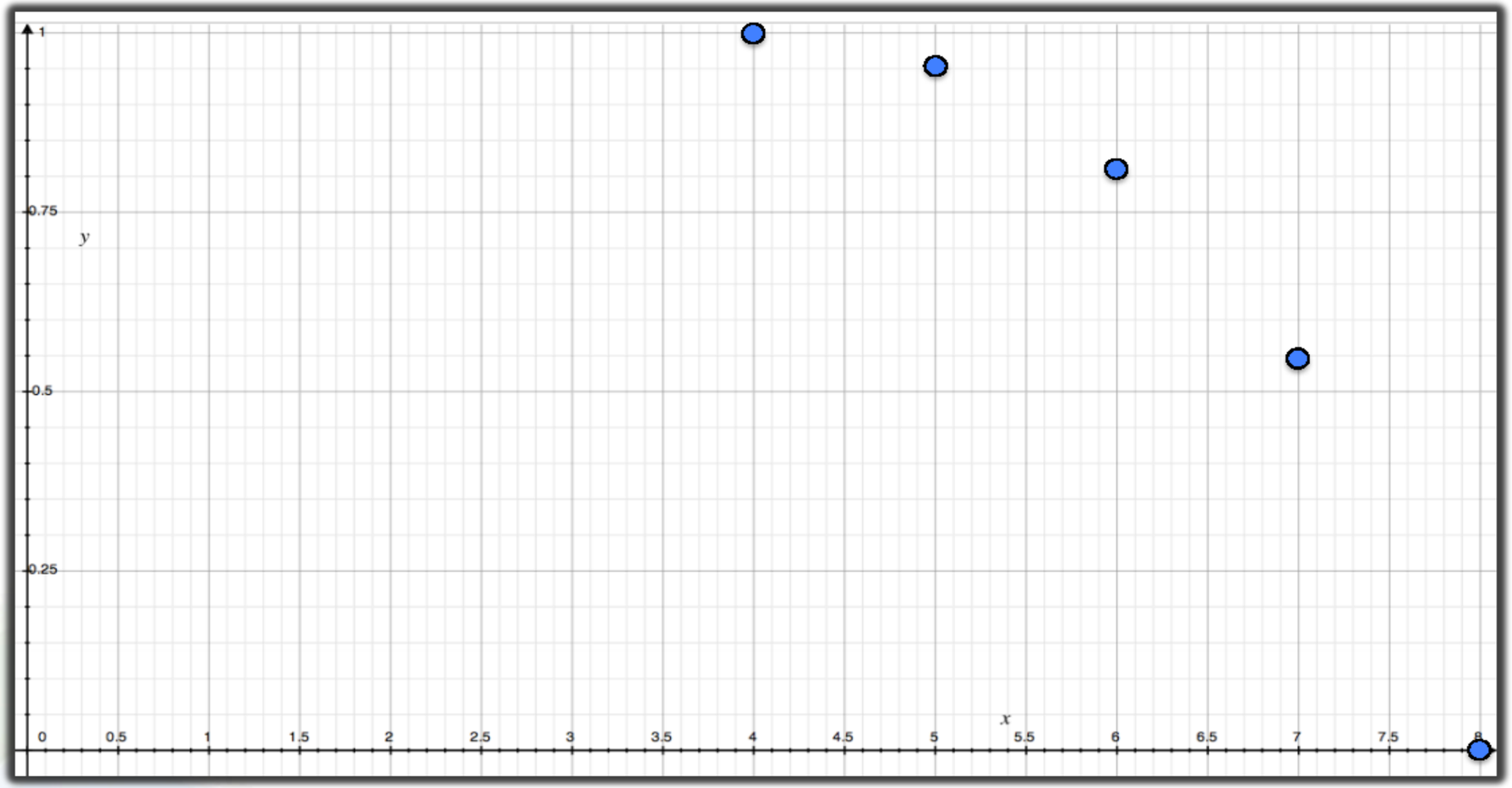
$$-\left(\frac{6}{8}\right) log_2 \left(\frac{6}{8}\right) - \left(\frac{2}{8}\right) log_2 \left(\frac{2}{8}\right) = 0.81$$



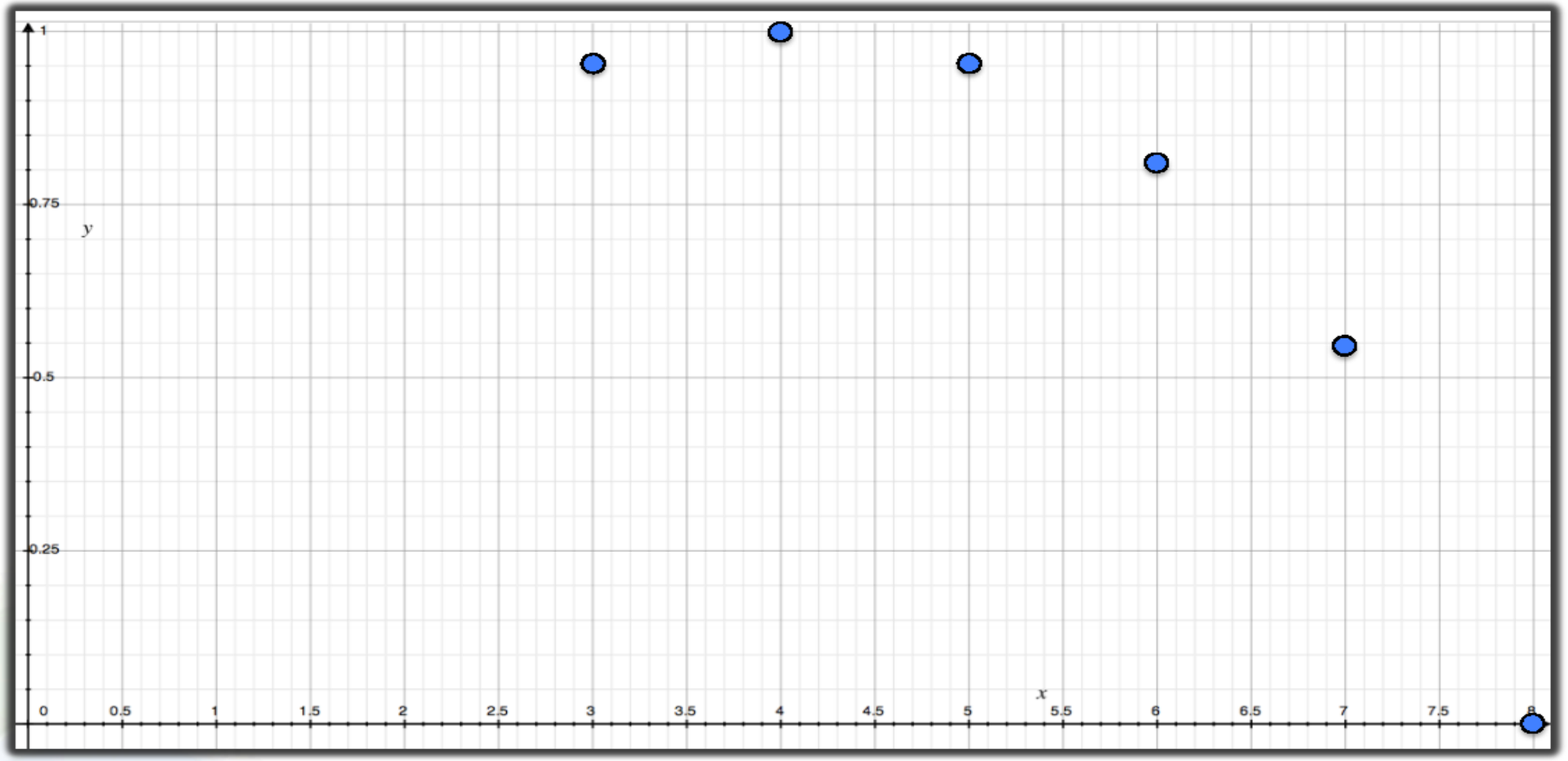
$$-\left(\frac{5}{8}\right) log_2 \left(\frac{5}{8}\right) - \left(\frac{3}{8}\right) log_2 \left(\frac{3}{8}\right) = 0.95$$



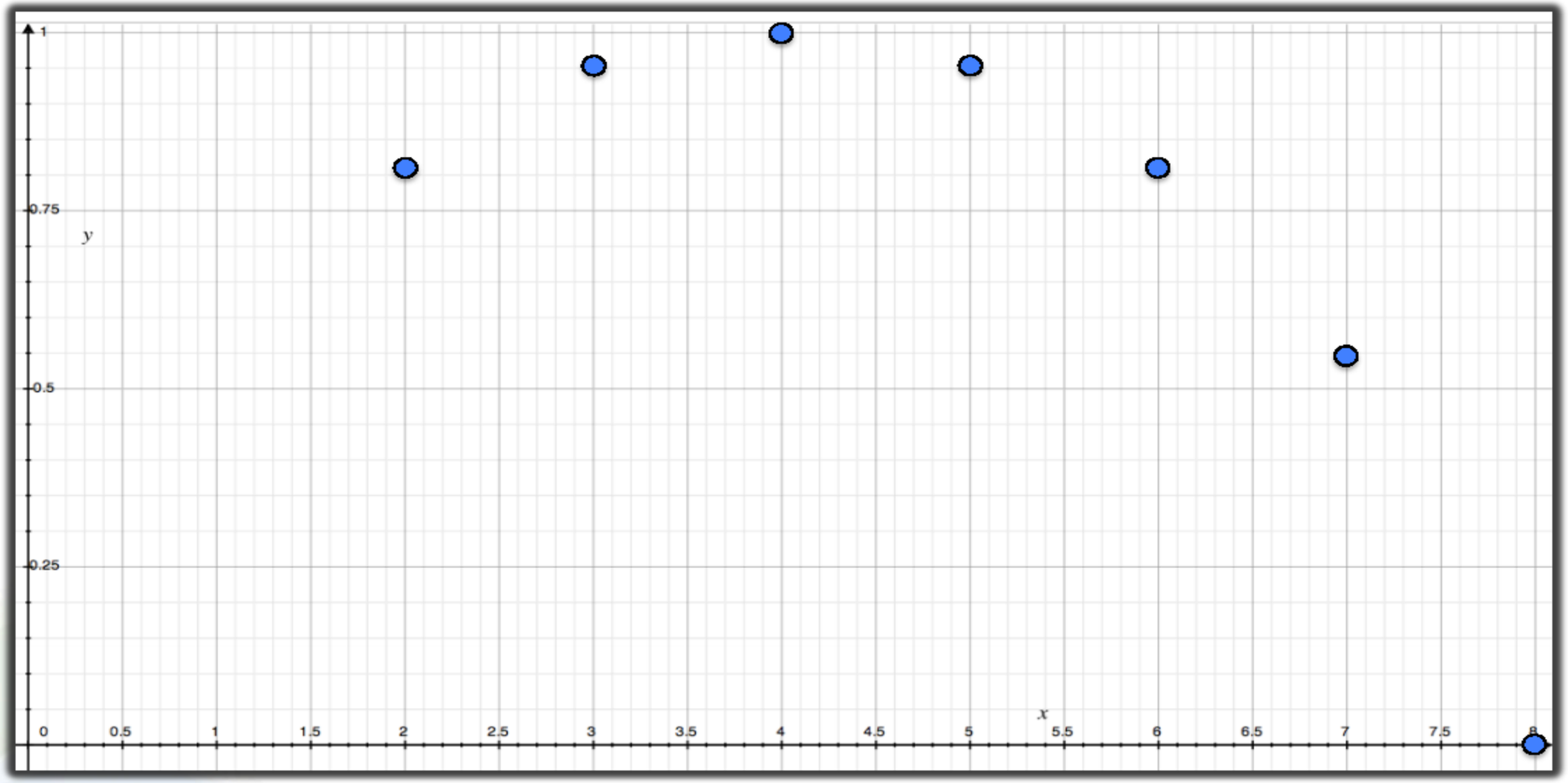
$$-\left(\frac{4}{8}\right) log_2 \left(\frac{4}{8}\right) - \left(\frac{4}{8}\right) log_2 \left(\frac{4}{8}\right) = 1$$



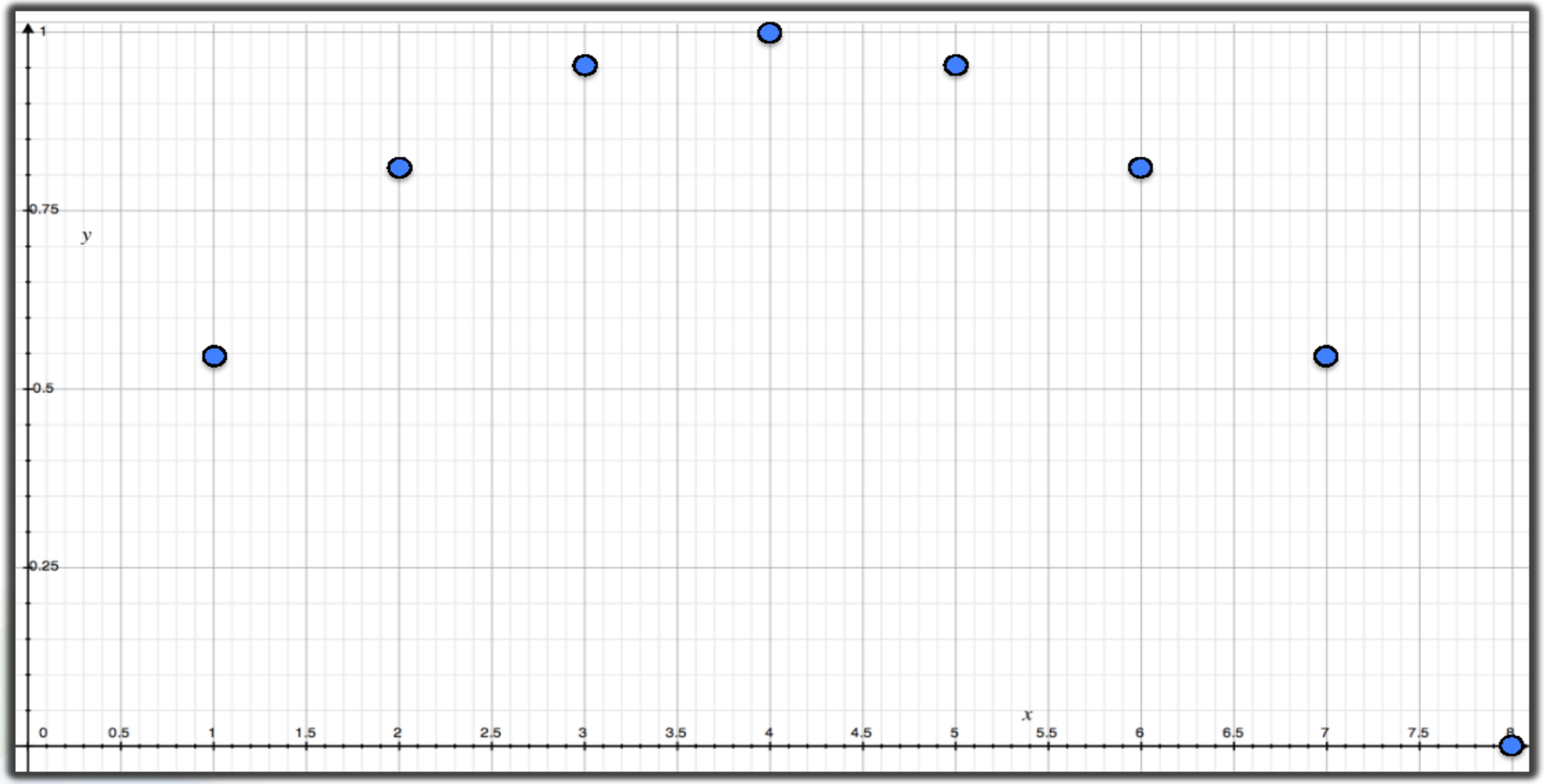
$$-\left(\frac{3}{8}\right) log_2 \left(\frac{3}{8}\right) - \left(\frac{5}{8}\right) log_2 \left(\frac{5}{8}\right) = 0.95$$



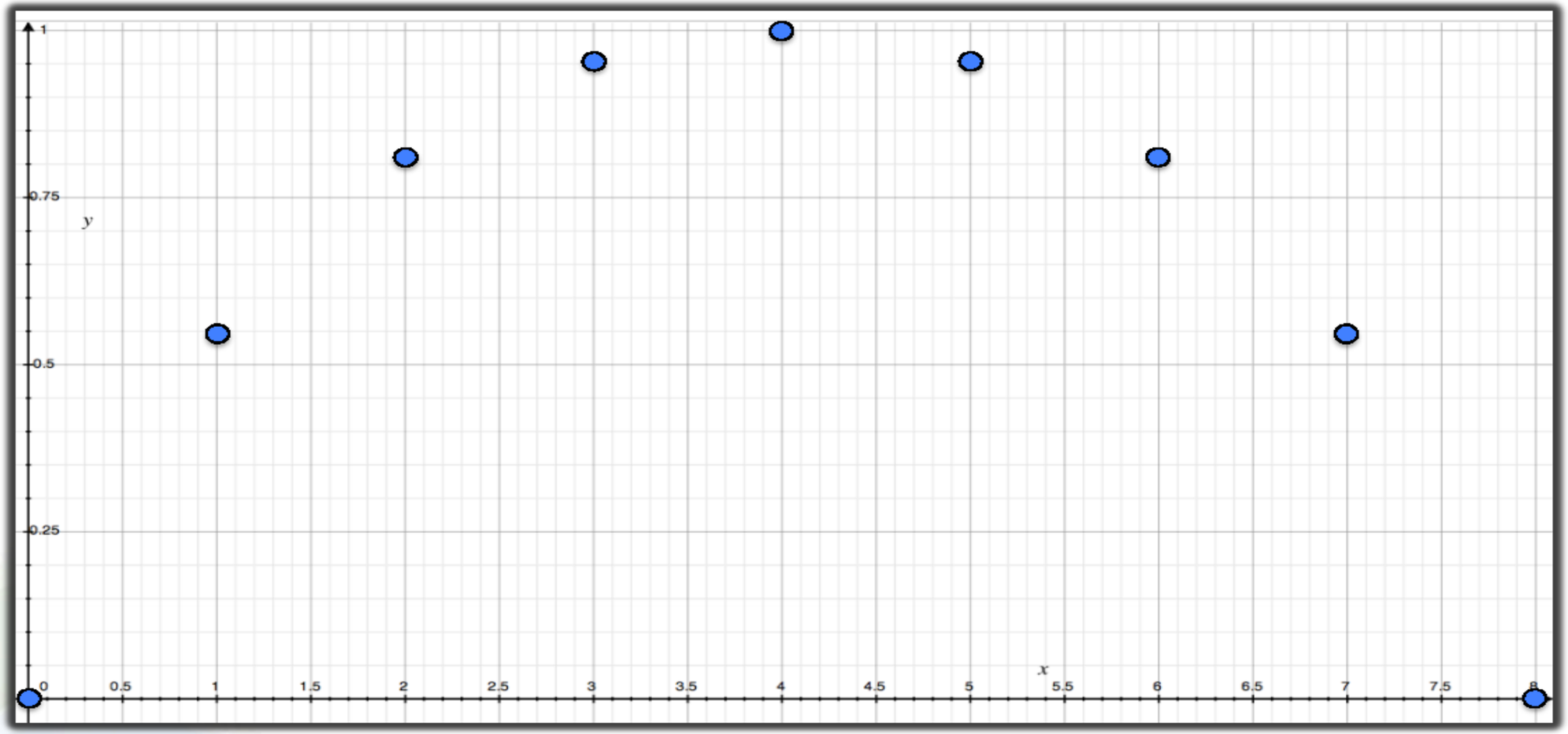
$$-\left(\frac{2}{8}\right) log_2 \left(\frac{2}{8}\right) - \left(\frac{6}{8}\right) log_2 \left(\frac{6}{8}\right) = 0.81$$



$$-\left(\frac{1}{8}\right) log_2 \left(\frac{1}{8}\right) - \left(\frac{7}{8}\right) log_2 \left(\frac{7}{8}\right) = 0.54$$

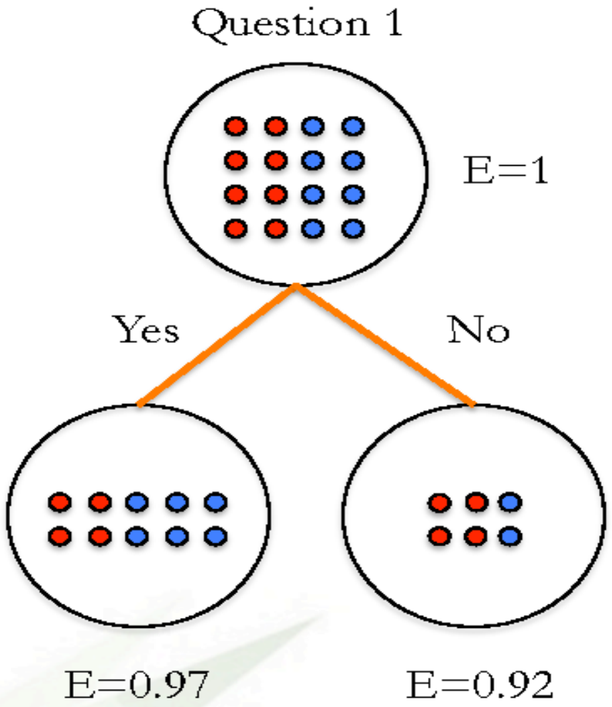


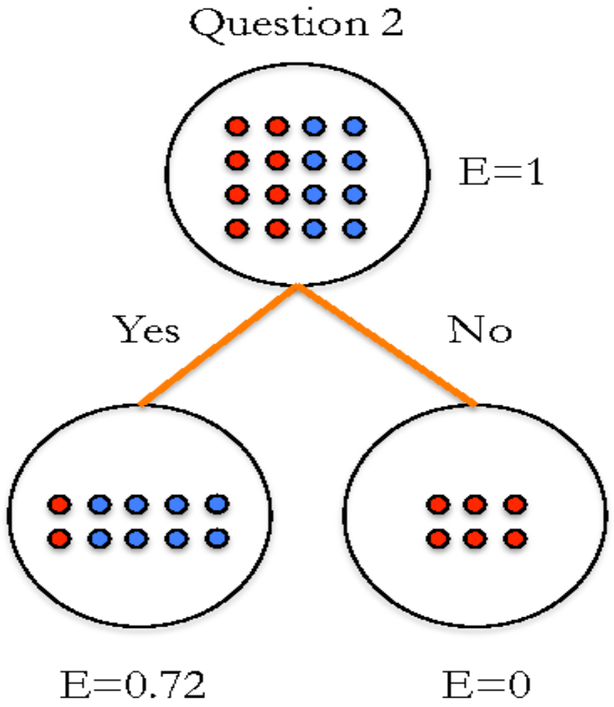
$$-\left(\frac{0}{8}\right) log_2 \left(\frac{0}{8}\right) - \left(\frac{8}{8}\right) log_2 \left(\frac{8}{8}\right) = 0$$

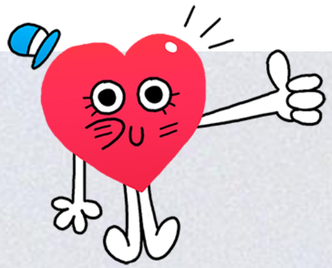


$$y = -\sum_{i=1}^{k} p_i log_k(p_i)$$

$$y = -\underbrace{\left[\left(\frac{1}{10}\right) log_4\left(\frac{1}{10}\right)\right]}_{\text{Red}} - \underbrace{\left[\left(\frac{3}{10}\right) log_4\left(\frac{3}{10}\right)\right]}_{\text{Green}} - \underbrace{\left[\left(\frac{2}{10}\right) log_4\left(\frac{2}{10}\right)\right]}_{\text{Blue}} - \underbrace{\left[\left(\frac{4}{10}\right) log_4\left(\frac{4}{10}\right)\right]}_{\text{Yellow}}$$


Question 1
E=1
Yes
No
E=0.97
E=0.92


Question 2
E=1
Yes
No
E=0.72
E=0



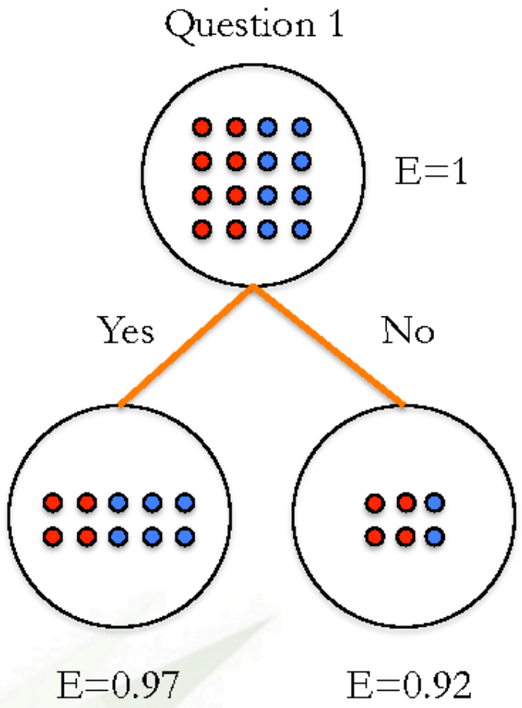
Information Gain:

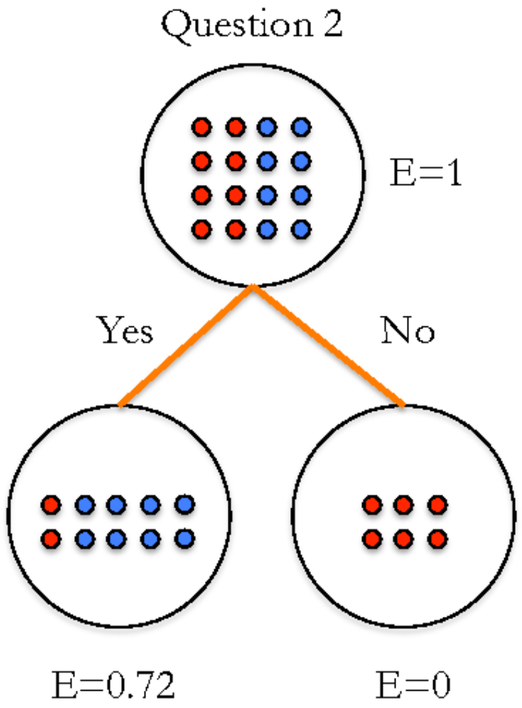
$$Gain_{split} = Entropy(p) - \sum_{i=1}^{k} \frac{n_i}{n} Entropy(i)$$

Parent Node, p is split into k partitions (children)

n_i is number of records in child node i

- Choose the split that achieves most reduction (maximizes GAIN)
- Used in the ID3 and C4.5 decision tree algorithms
- Information gain is the mutual information between the class variable and the splitting variable





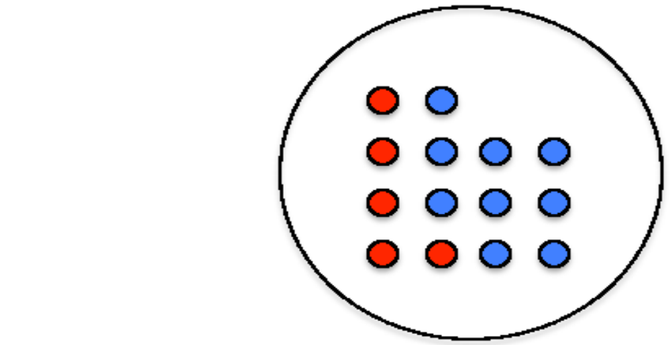
$$G(S, Q_1) = 1 - \left(\frac{10}{16}\right) \times 0.97 - \left(\frac{6}{16}\right) \times 0.92$$

$$G(S, Q_1) = 0.049$$

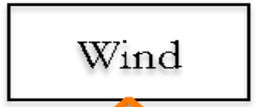
$$G(S, Q_2) = 1 - \left(\frac{10}{16}\right) \times 0.72 - \left(\frac{6}{16}\right) \times 0$$

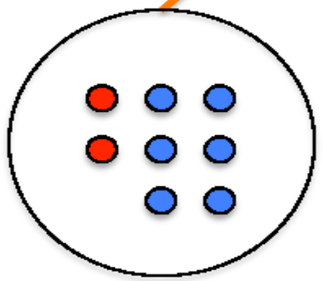
$$G(S, Q_2) = 0.55$$

Day	Outlook	Temp	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Weak	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Strong	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

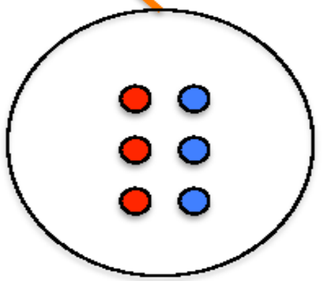


E=0.954





E=0.811

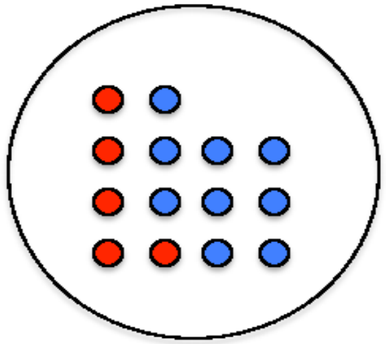


E=1

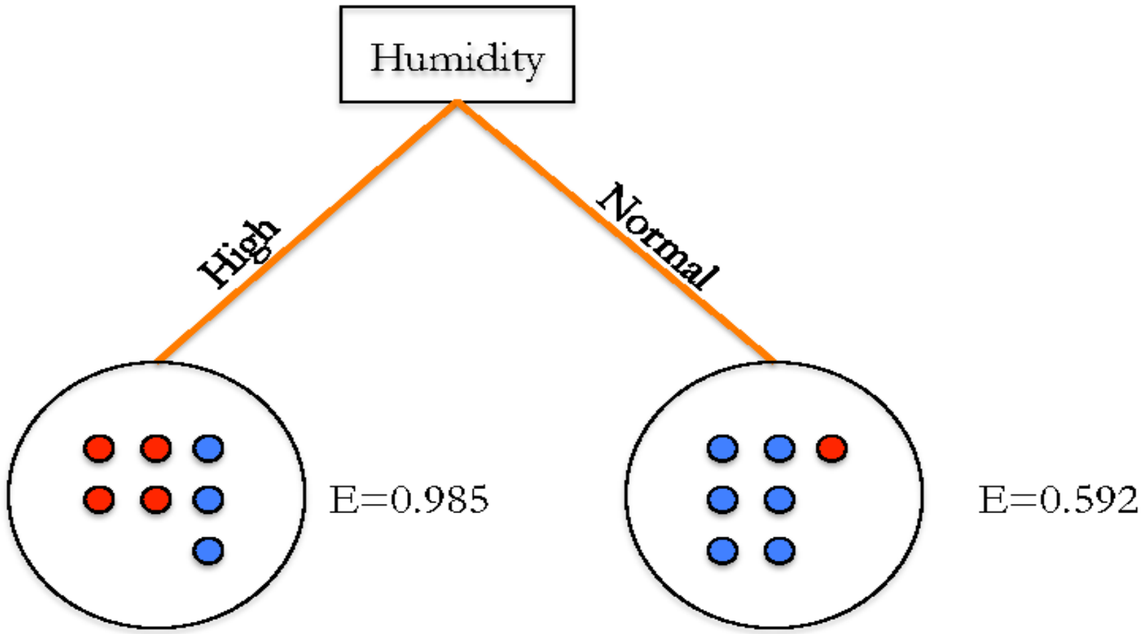
$$G(S, Wind) = 0.954 - \frac{8}{14}0.811 - \frac{6}{14}1$$

$$G(S, Wind) = 0.048$$

$G(S, Wind) = 0.048$



E=0.954



E=0.985

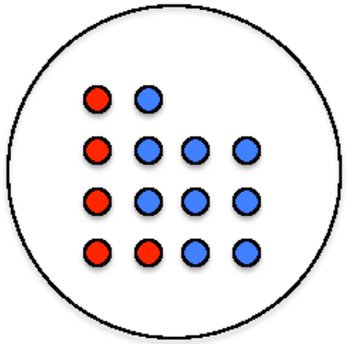
E=0.592

$$G(S, Humidity) = 0.954 - \frac{7}{14}0.985 - \frac{7}{14}0.592$$

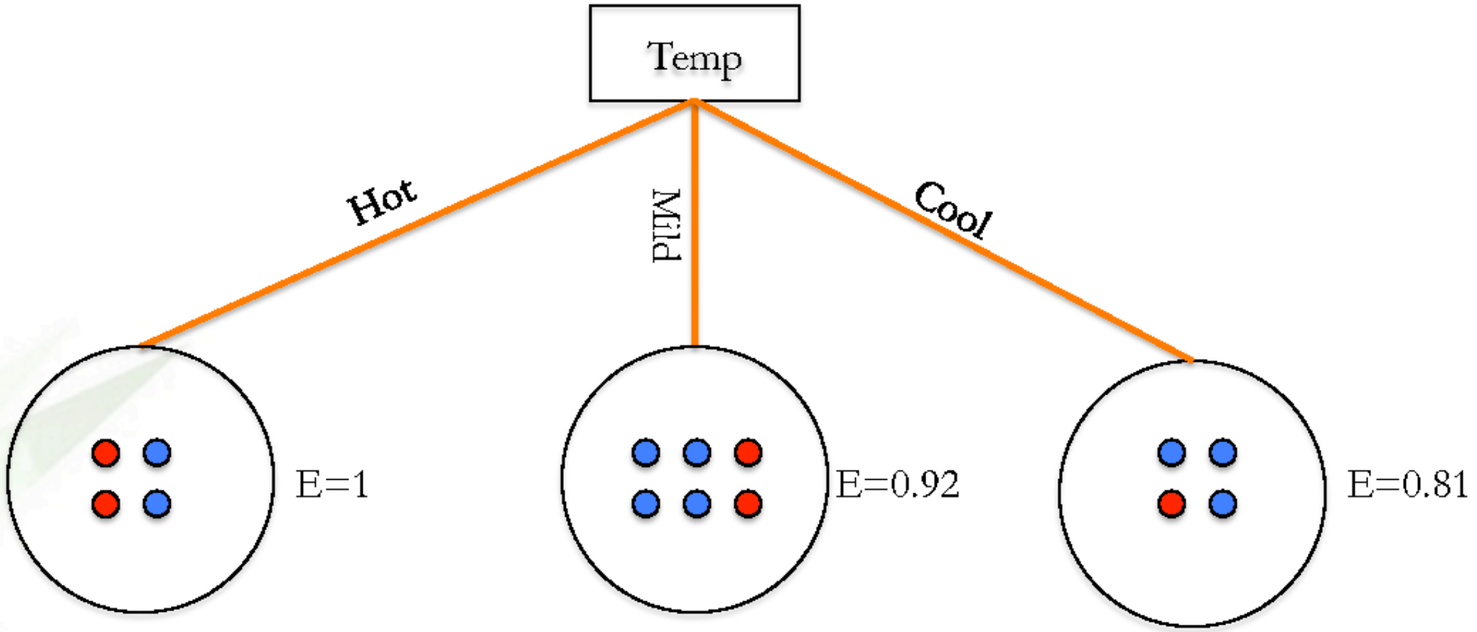
$$G(S, Humidity) = 0.151$$

$G(S, Wind) = 0.048$

$G(S, Humidity) = 0.151$



E=0.954



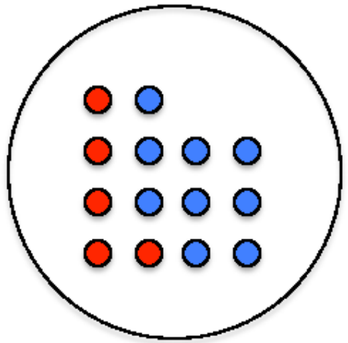
$$G(S, Temp) = 0.954 - \frac{4}{14}1 - \frac{6}{14}0.92 - \frac{4}{14}0.81$$

$$G(S, Temp) = 0.042$$

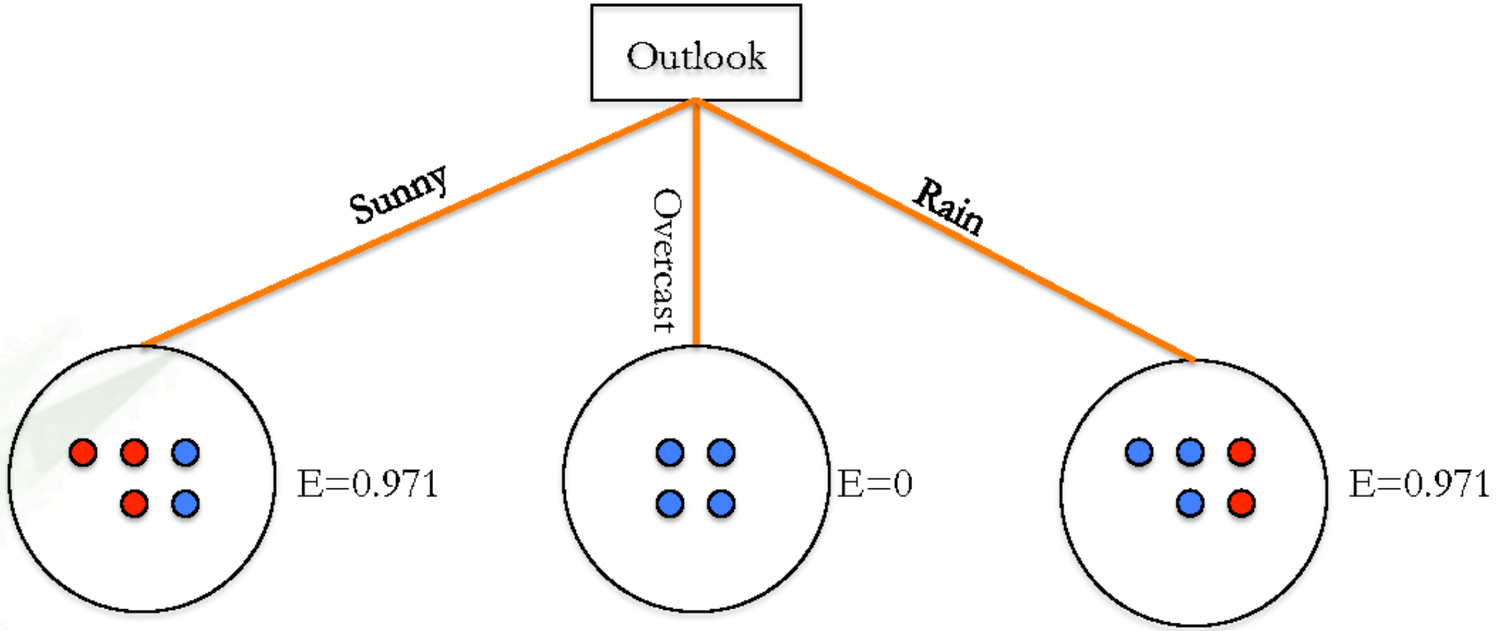
$G(S, Wind) = 0.048$

$G(S, Humidity) = 0.151$

$G(S, Temp) = 0.042$

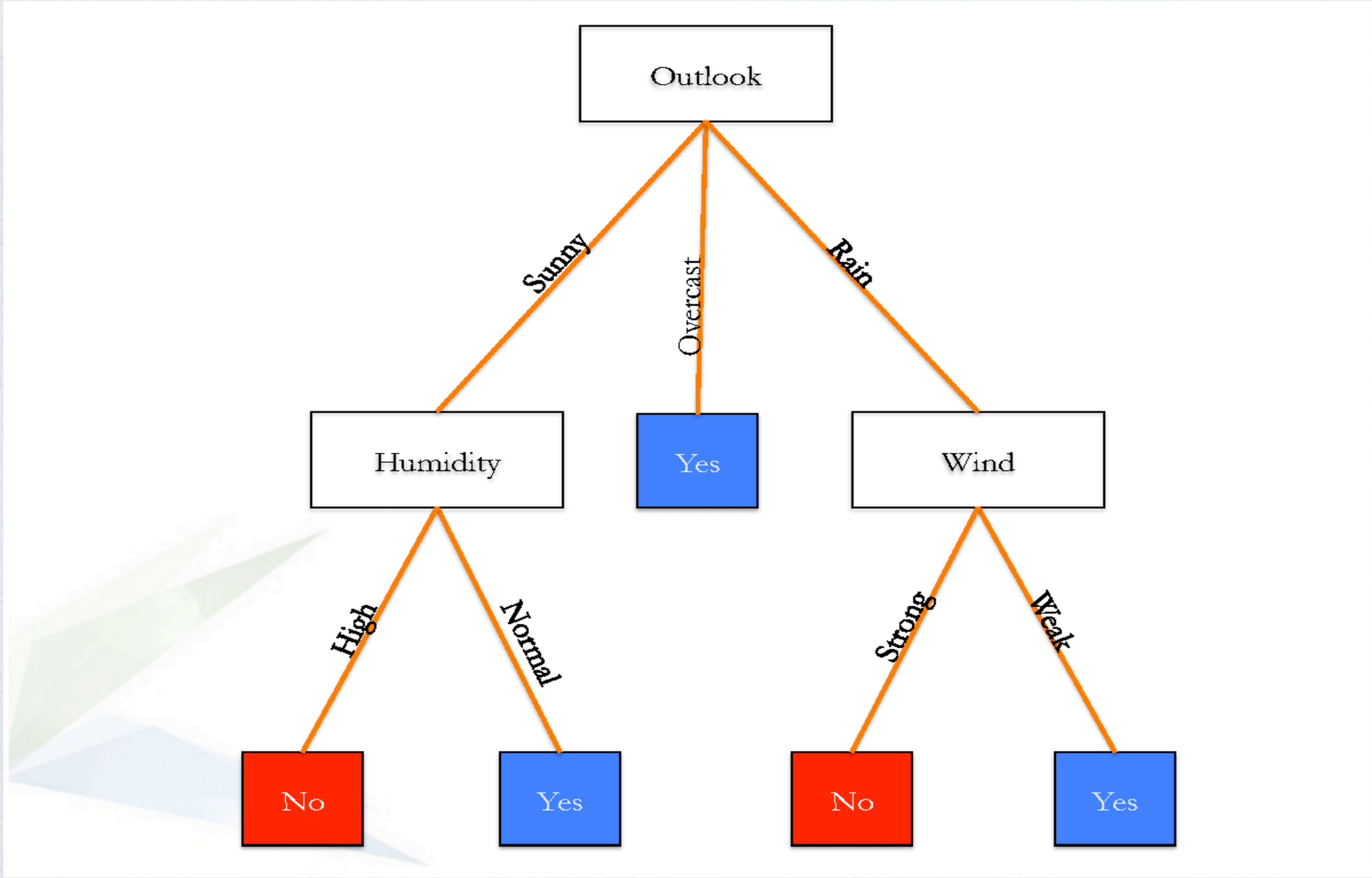


E=0.954



$$G(S, Outlook) = 0.954 - \frac{5}{14}0.971 - \frac{4}{14}0 - \frac{5}{14}0.971$$

$$G(S, Outlook) = 0.247$$


Outlook
Sunny
Overcast
Rain
Humidity
Yes
Wind
High
Normal
Strong
Weak
No
Yes
No
Yes

Decision Tree

Advantages:

•It is very easy to understand and interpret.

•The data for decision trees require minimal preparation.

•They force you to find many possible outcomes of a decision.

•Can be easily used with many other decision tools.

•Helps you to make the best decisions and best guesses on the basis of the information you have.

•Helps you to see the difference between controlled and uncontrolled events.

•Helps you estimate the likely results of one decision against another.

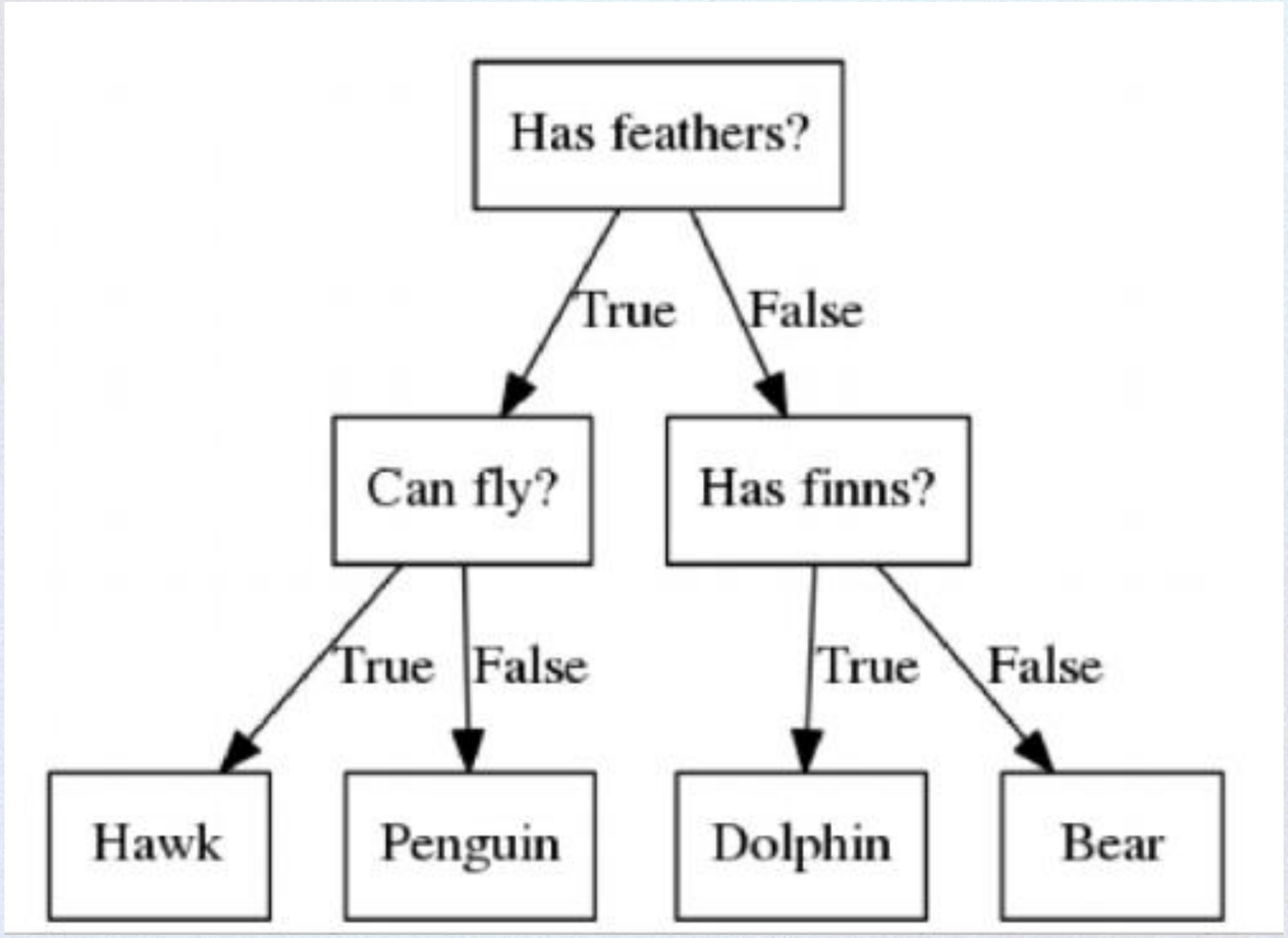
Disadvantages:

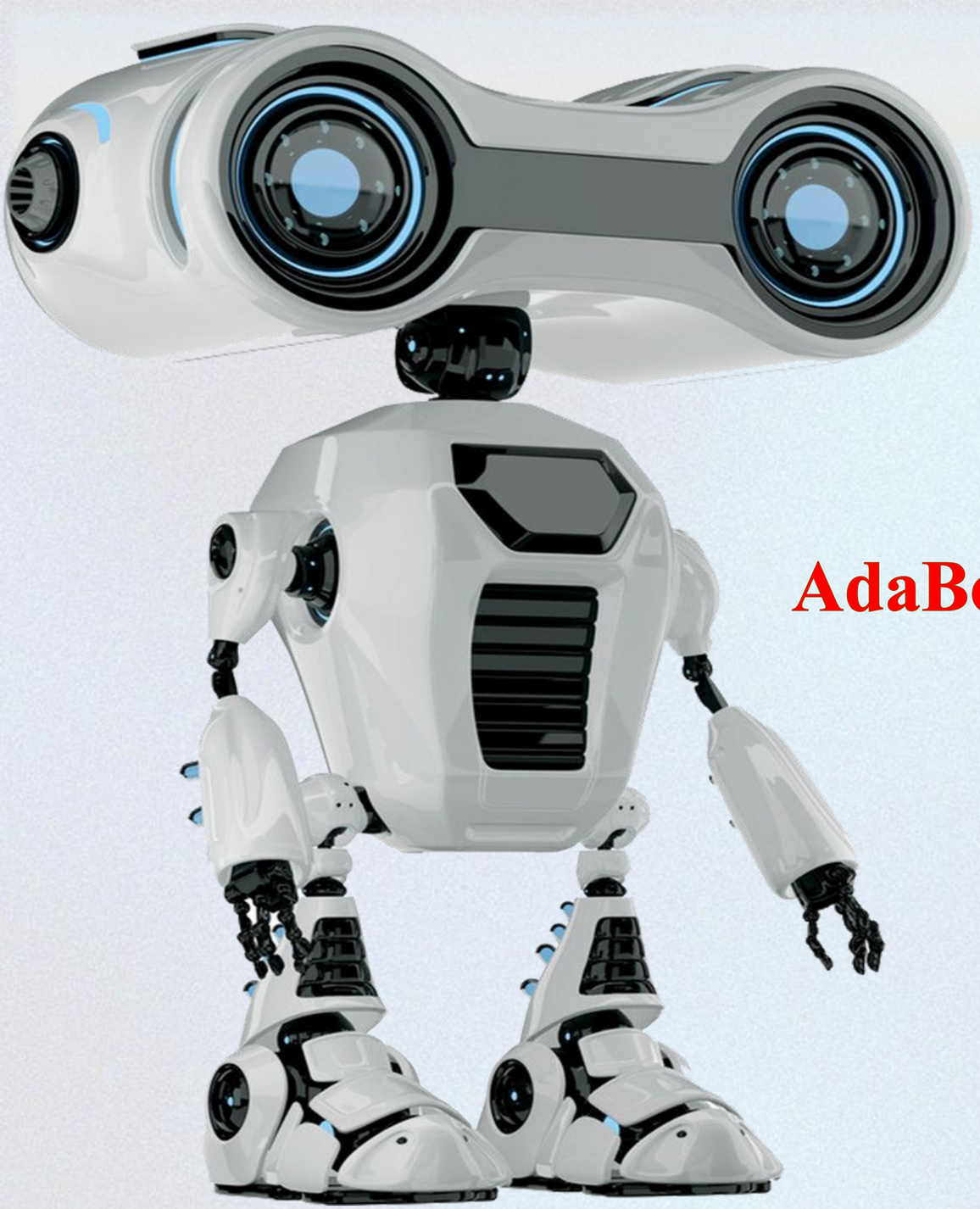
•Sometimes decision trees can become too complex.

•The outcomes of decisions may be based mainly on your expectations. This can lead to unrealistic decision trees.

•The diagrams can narrow your focus to critical decisions and objectives.

Assignment


Has feathers?
True
False
Can fly?
Has finns?
True
False
True
False
Hawk
Penguin
Dolphin
Bear

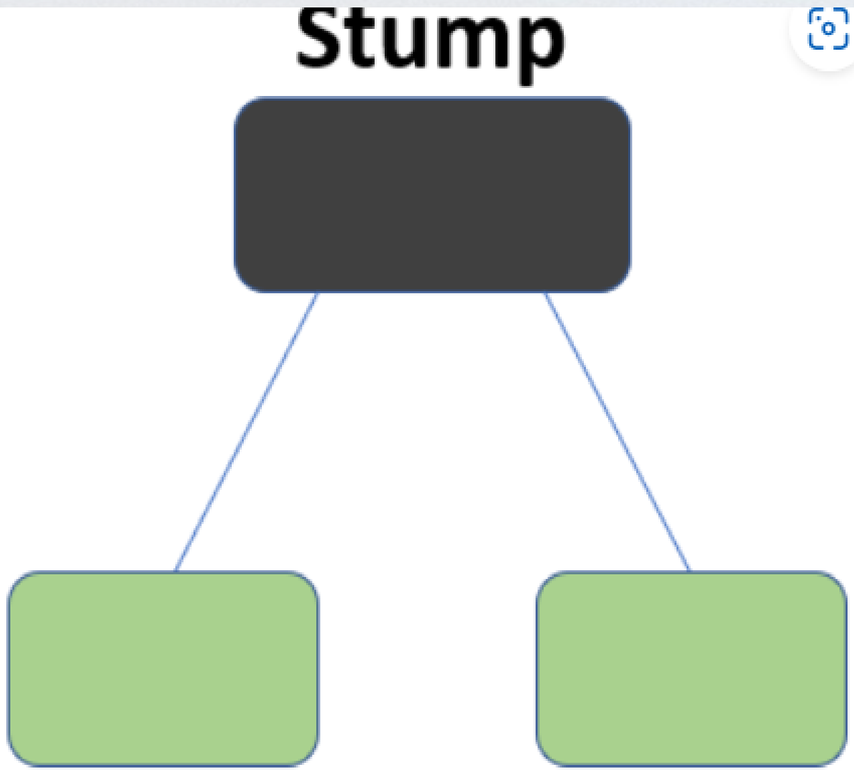


AdaBoost Algorithm

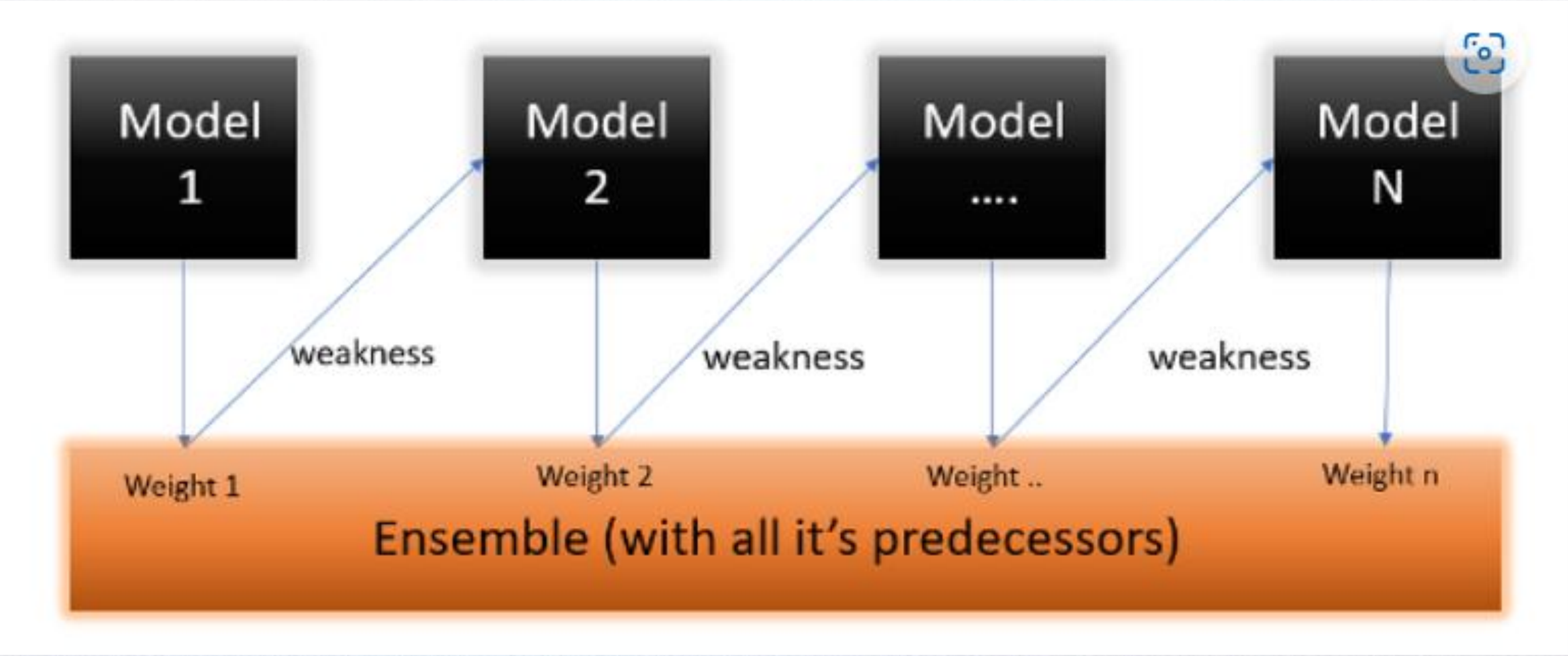
What Is the AdaBoost Algorithm?

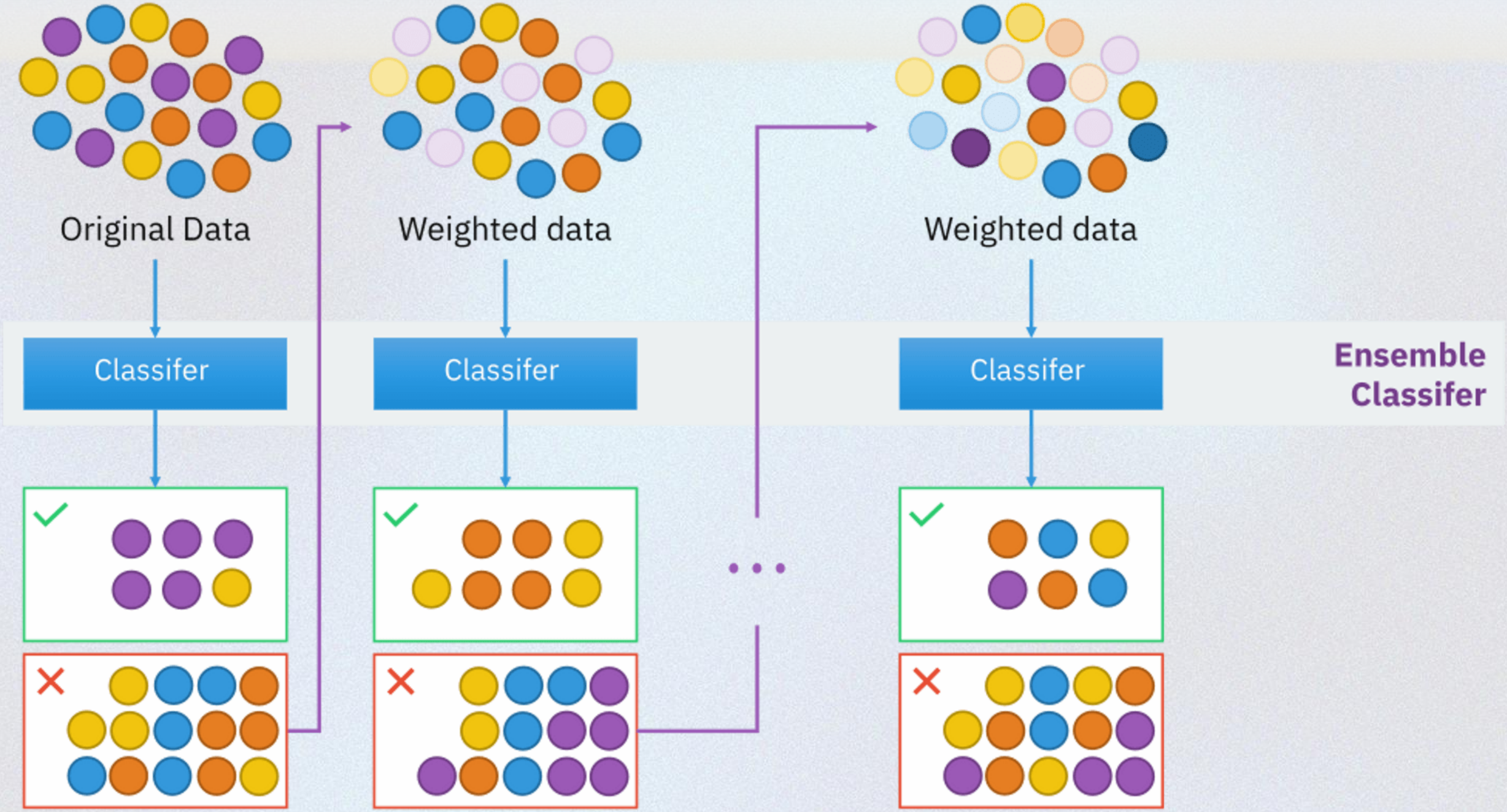
There are many machine learning algorithms to choose from for your problem statements. One of these algorithms for predictive modeling is called AdaBoost.

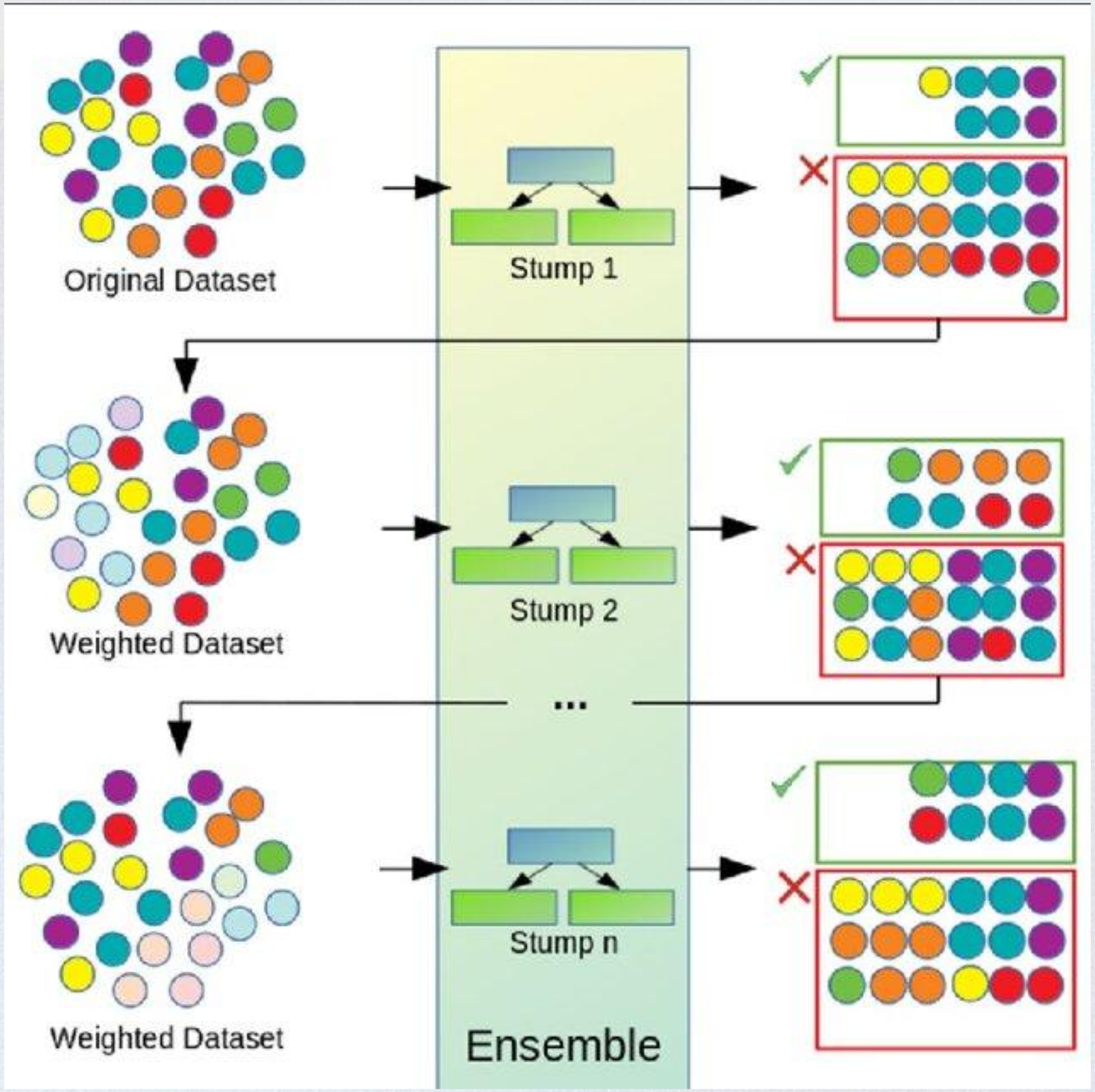
AdaBoost algorithm, short for Adaptive Boosting, is **a Boosting technique used as an Ensemble Method in Machine Learning**. It is called Adaptive Boosting as the weights are re-assigned to each instance, with higher weights assigned to incorrectly classified instances.



What this algorithm does is that it builds a model and gives equal weights to all the data points. It then assigns higher weights to points that are wrongly classified. Now all the points with higher weights are given more importance in the next model. It will keep training models until and unless a lower error is received.


Model 1
Model 2
Model ....
Model N
weakness
weakness
weakness
Weight 1
Weight 2
Weight ..
Weight n
Ensemble (with all it's predecessors)


Original Data
Weighted data
Weighted data
Classifer
Classifer
Classifer
Ensemble Classifer


Original Dataset
Stump 1
Weighted Dataset
Stump 2
...
Weighted Dataset
Stump n
Ensemble

Let’s take an example to understand this, suppose you built a decision tree algorithm on the Titanic dataset, and from there, you get an accuracy of 80%. After this, you apply a different algorithm and check the accuracy, and it comes out to be 75% for KNN and 70% for Linear Regression.

When building different models on the same dataset, we observe variations in accuracy. However, leveraging the power of AdaBoost, we can combine these algorithms to enhance the final predictions. By averaging the results from diverse models, Adaboost allows us to achieve higher accuracy and bolster predictive capabilities effectively.

Understanding the Working of the AdaBoost Algorithm

Let’s understand what and how this algorithm works under the hood with the following tutorial.

Step 1: Assigning Weights

The Image shown below is the actual representation of our dataset. Since the target column is binary, it is a classification problem. First of all, these data points will be assigned some weights. Initially, all the weights will be equal.

The formula to calculate the sample weights is:

$$w(x_i, y_i) = \frac{1}{N},\ i = 1, 2, \ldots . n$$

Where N is the total number of data points

Here since we have 5 data points, the sample weights assigned will be 1/5.

Row No.	Gender	Age	Income	Illness	Sample Weights
1	Male	41	40000	Yes	1/5
2	Male	54	30000	No	1/5
3	Female	42	25000	No	1/5
4	Female	40	60000	Yes	1/5
5	Male	46	50000	Yes	1/5

Step 2: Classify the Samples

We start by seeing how well "*Gender*" classifies the samples and will see how the variables (Age, Income) classify the samples.

We'll create a decision stump for each of the features and then calculate the ***Gini Index*** of each tree. The tree with the lowest Gini Index will be our first stump.

Here in our dataset, let's say ***Gender*** has the lowest gini index, so it will be our first stump.

Step 3: Calculate the Influence

We'll now calculate the **"Amount of Say"** or **"Importance"** or **"Influence"** for this classifier in classifying the data points using this formula:

$$\frac{1}{2}\log\frac{1-Total\ Error}{Total\ Error}$$

The total error is nothing but the summation of all the sample weights of misclassified data points.

Here in our dataset, let's assume there is 1 wrong output, so our total error will be 1/5, and th

alpha (performance of the stump) will be:

$$Performance\ of\ the\ stump\ =\ \frac{1}{2}\log_e(\frac{1-Total\ Error}{Total\ Error})$$

$$\alpha\ =\ \frac{1}{2}\log_e\left(\frac{1-\frac{1}{5}}{\frac{1}{5}}\right)$$

$$\alpha\ =\ \frac{1}{2}\log_e\left(\frac{0.8}{0.2}\right)$$

$$\alpha\ =\ \frac{1}{2}\log_e(4)\ =\ \frac{1}{2}*(1.38)$$

$$\alpha\ =\ 0.69$$

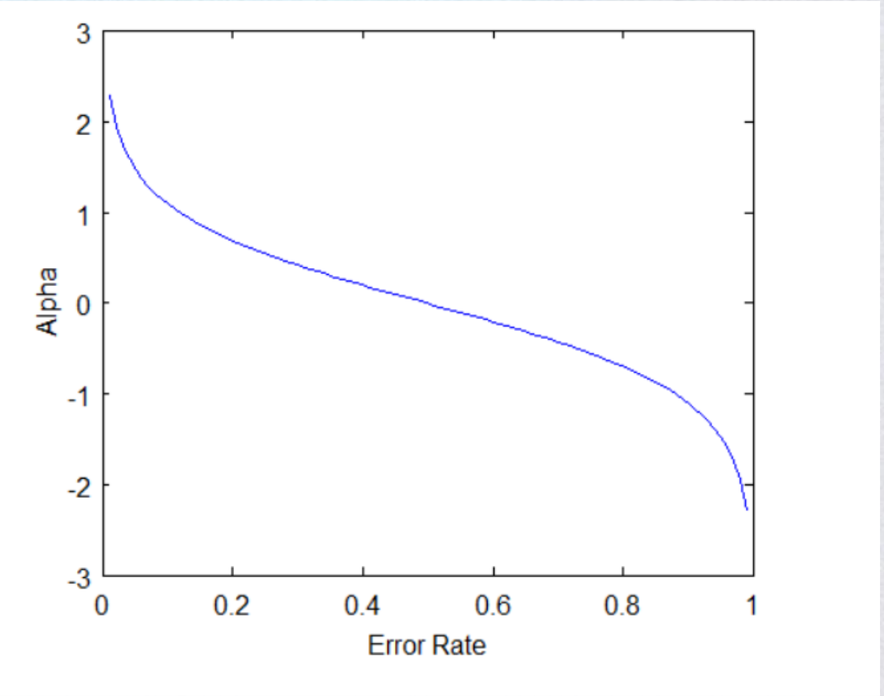
Note: Total error will always be between 0 and 1.

0 Indicates perfect stump, and 1 indicates horrible stump.

From the graph above, we can see that when there is no misclassification, then we have no error (Total Error = 0), so the "amount of say (alpha)" will be a large number.

When the classifier predicts half right and half wrong, then the Total Error = 0.5, and the importance (amount of say) of the classifier will be 0.

If all the samples have been incorrectly classified, then the error will be very high (approx. to 1), and hence our alpha value will be a negative integer.



Step 4: Calculate TE and Performance

You might be wondering about the significance of calculating the Total Error (TE) and performance of an Adaboost stump. The reason is straightforward – updating the weights is crucial. If identical weights are maintained for the subsequent model, the output will mirror what was obtained in the initial model.

The wrong predictions will be given more weight, whereas the correct predictions weights will be decreased. Now when we build our next model after updating the weights, more preference will be given to the points with higher weights.

After finding the importance of the classifier and total error, we need to finally update the weights, and for this, we use the following formula:

$$New\ sample\ weight\ =\ old\ weight\ *\ e^{\pm Amount\ of\ say\ (\alpha)}$$

The amount of, say (alpha) will be ***negative*** when the sample is **correctly classified**.

The amount of, say (alpha) will be ***positive*** when the sample is **miss-classified.**

There are four correctly classified samples and 1 wrong. Here, the ***sample weight*** of that datapoint is *1/5*, and the ***amount of say/performance of the stump*** of ***Gender*** is *0.69*.

New weights for *correctly classified* samples are:

$$New\ sample\ weight\ =\ \frac{1}{5}\ *\ \exp(-0.69)$$

$$New\ sample\ weight\ =\ 0.2\ *\ 0.502\ =\ 0.1004$$

For *wrongly classified* samples, the updated weights will be:

$$New\ sample\ weight\ =\ \frac{1}{5}\ *\ \exp(0.69)$$

$$New\ sample\ weight\ =\ 0.2\ *\ 1.994\ =\ 0.3988$$

See the sign of alpha when I am putting the values, the **alpha is negative** when the data point is correctly classified, and this *decreases the sample weight* from 0.2 to 0.1004. It is **positive** when there is **misclassification**, and this will *increase the sample weight* from 0.2 to 0.3988

Row No.	Gender	Age	Income	Illness	Sample Weights	New Sample Weights
1	Male	41	40000	Yes	1/5	0.1004
2	Male	54	30000	No	1/5	0.1004
3	Female	42	25000	No	1/5	0.1004
4	Female	40	60000	Yes	1/5	0.3988
5	Male	46	50000	Yes	1/5	0.1004

We know that the total sum of the sample weights must be equal to 1, but here if we sum up all the new sample weights, we will get 0.8004. To bring this sum equal to 1, we will normalize these weights by dividing all the weights by the total sum of updated weights, which is 0.8004. So, after normalizing the sample weights, we get this dataset, and now the sum is equal to 1.

Row No.	Gender	Age	Income	Illness	Sample Weights	New Sample Weights
1	Male	41	40000	Yes	1/5	0.1004/0.8004 =0.1254
2	Male	54	30000	No	1/5	0.1004/0.8004 =0.1254
3	Female	42	25000	No	1/5	0.1004/0.8004 =0.1254
4	Female	40	60000	Yes	1/5	0.3988/0.8004 =0.4982
5	Male	46	50000	Yes	1/5	0.1004/0.8004 =0.1254

Step 5: Decrease Errors

Now, we need to make a new dataset to see if the errors decreased or not. For this, we will remove the “sample weights” and “new sample weights” columns and then, based on the “new sample weights,” divide our data points into buckets.

Row No.	Gender	Age	Income	Illness	New Sample Weights	Buckets
1	Male	41	40000	Yes	0.1004/0.8004= 0.1254	0 to 0.1254
2	Male	54	30000	No	0.1004/0.8004= 0.1254	0.1254 to 0.2508
3	Female	42	25000	No	0.1004/0.8004= 0.1254	0.2508 to 0.3762
4	Female	40	60000	Yes	0.3988/0.8004= 0.4982	0.3762 to 0.8744
5	Male	46	50000	Yes	0.1004/0.8004= 0.1254	0.8744 to 0.9998

Step 6: New Dataset

We are almost done. Now, what the algorithm does is selects random numbers from 0-1. Since incorrectly classified records have higher sample weights, the probability of selecting those records is very high.

Suppose the 5 random numbers our algorithm take is 0.38,0.26,0.98,0.40,0.55.

Now we will see where these random numbers fall in the bucket, and according to it, we'll make our new dataset shown below.

Row No.	Gender	Age	Income	Illness
1	Female	40	60000	Yes
2	Male	54	30000	No
3	Female	42	25000	No
4	Female	40	60000	Yes
5	Female	40	60000	Yes

This comes out to be our new dataset, and we see the data point, which was wrongly classified, has been selected 3 times because it has a higher weight.

Step 7: Repeat Previous Steps

Now this act as our new dataset, and we need to repeat all the above steps i.e.

1. Assign ***equal weights*** to all the data points.
2. Find the stump that does the ***best job classifying*** the new collection of samples by finding their Gini Index and selecting the one with the lowest Gini index.
3. Calculate the ***"Amount of Say"*** and ***"Total error"*** to update the previous sample weights.
4. Normalize the new sample weights.

Iterate through these steps until and unless a low training error is achieved.

Suppose, with respect to our dataset, we have constructed 3 decision trees (DT1, DT2, DT3) in a ***sequential manner.*** If we send our **test data** now, it will pass through all the decision trees, and finally, we will see which class has the majority, and based on that, we will do predictions for our test dataset.

THANK YOU!