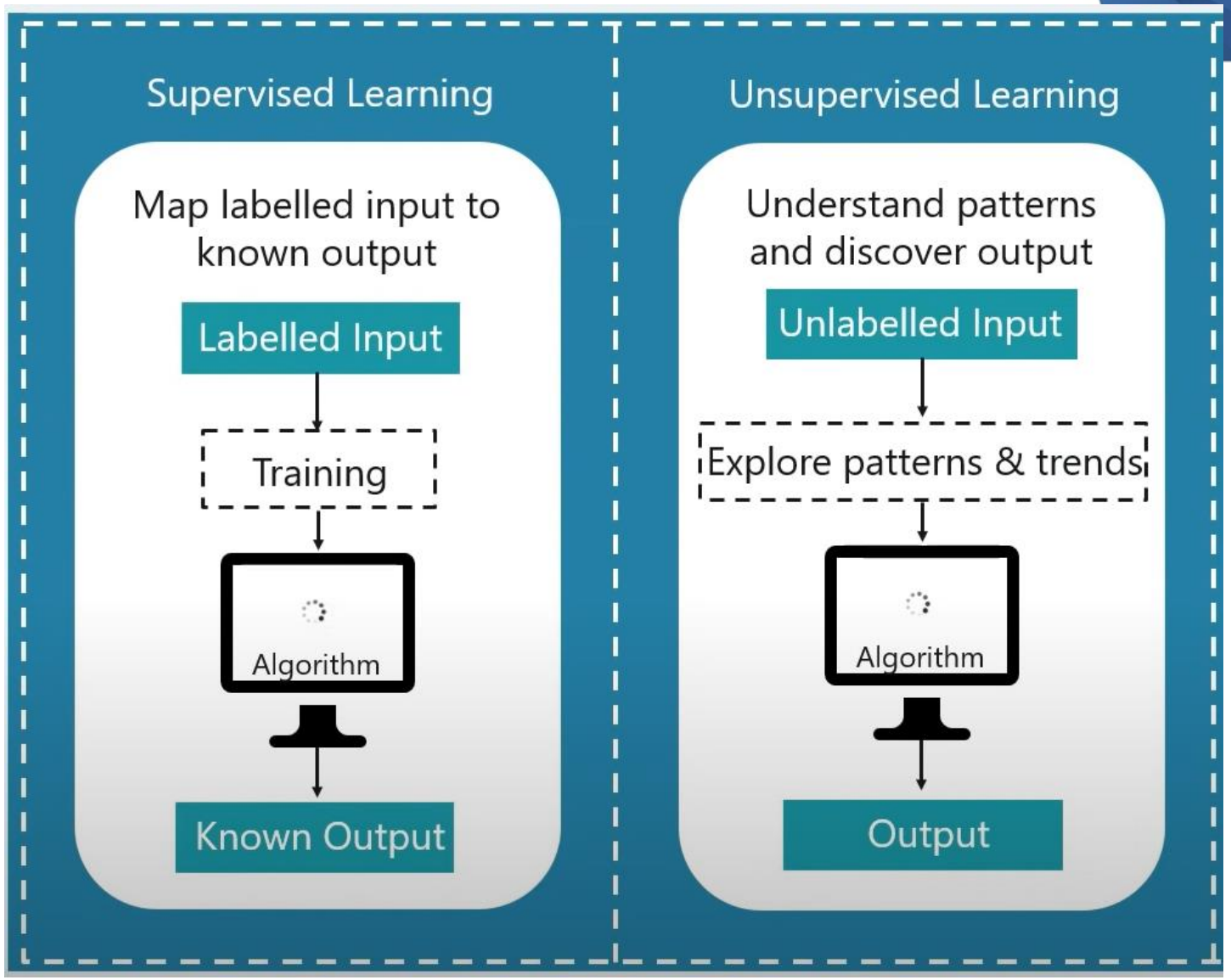
# Decision Trees, Random Forests and Performance Measures for Classification problems

**Instructor: Dr. Priyanka D Pantula**

Assistant Professor, Indian Institute of Technology (ISM) Dhanbad

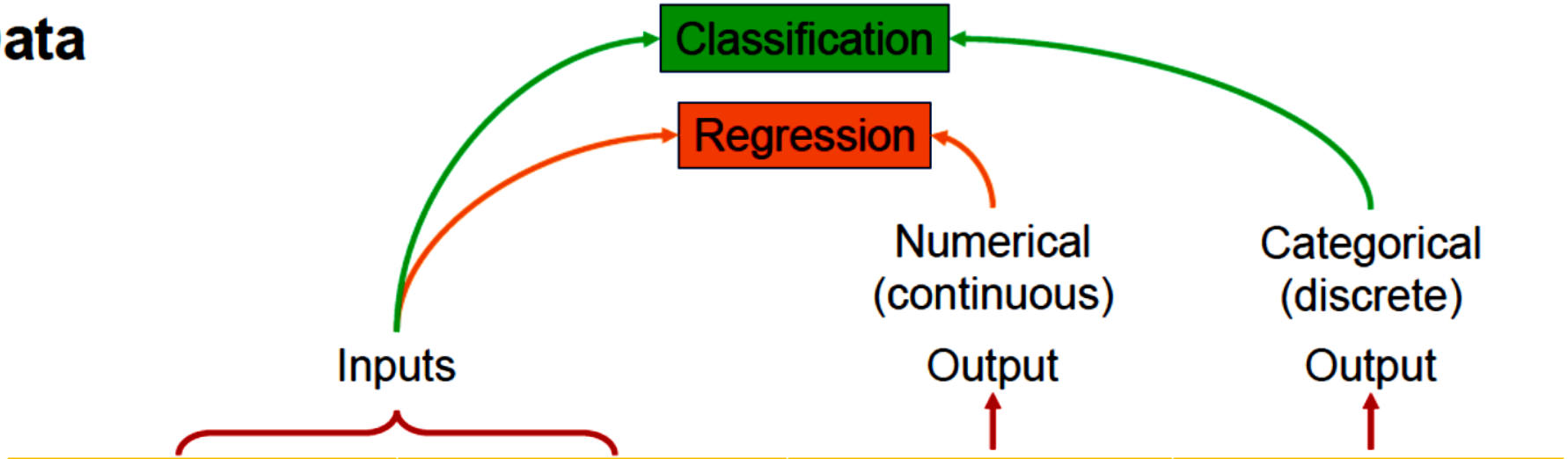
**(Email: pantula@iitism.ac.in)**

# Approach in Supervised & Unsupervised Machine Learning



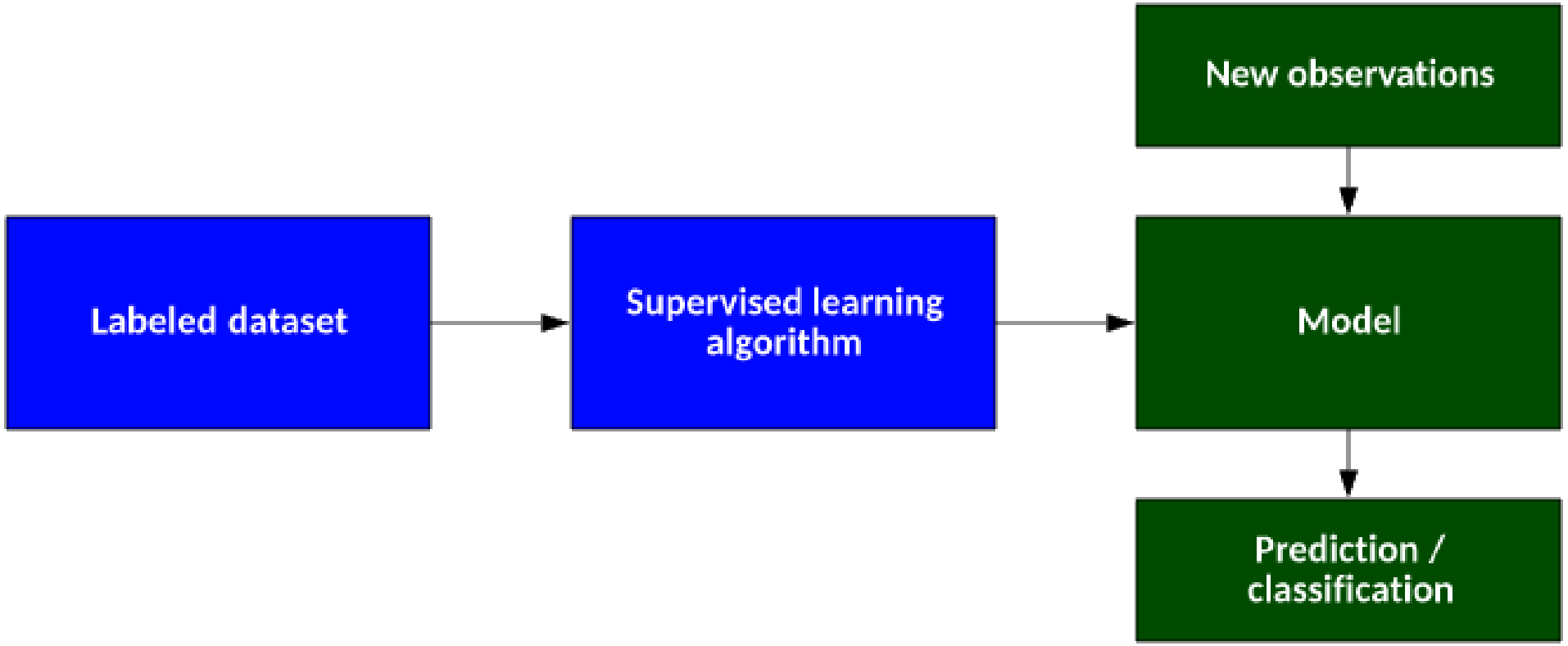
# Supervised Learning

❖ Data




| Temperature | Pressure | Ammonia Yield | Operational Risk |
|---|---|---|---|
| $T_1$ | $P_1$ | $A_1$ | High |
| $T_2$ | $P_2$ | $A_2$ | Low |
| $T_3$ | $P_3$ | $A_3$ | High |
| ... | ... | ... | ... |
| $T_{200}$ | $P_{200}$ | $A_{200}$ | Medium |

# Supervised Learning



# Decision Trees

# What is a decision tree?

Predictive model in the tree form that maps items to its target value, starting from the root to leaf (conclusion)

There are two types,

- **Classification tree:** When the final target value belongs to finite set (leaves are labels)
- **Regression tree:** When the final target value belongs to continuous set (different values based on features)

There are many algorithms like ID3, CART, CHAID etc.

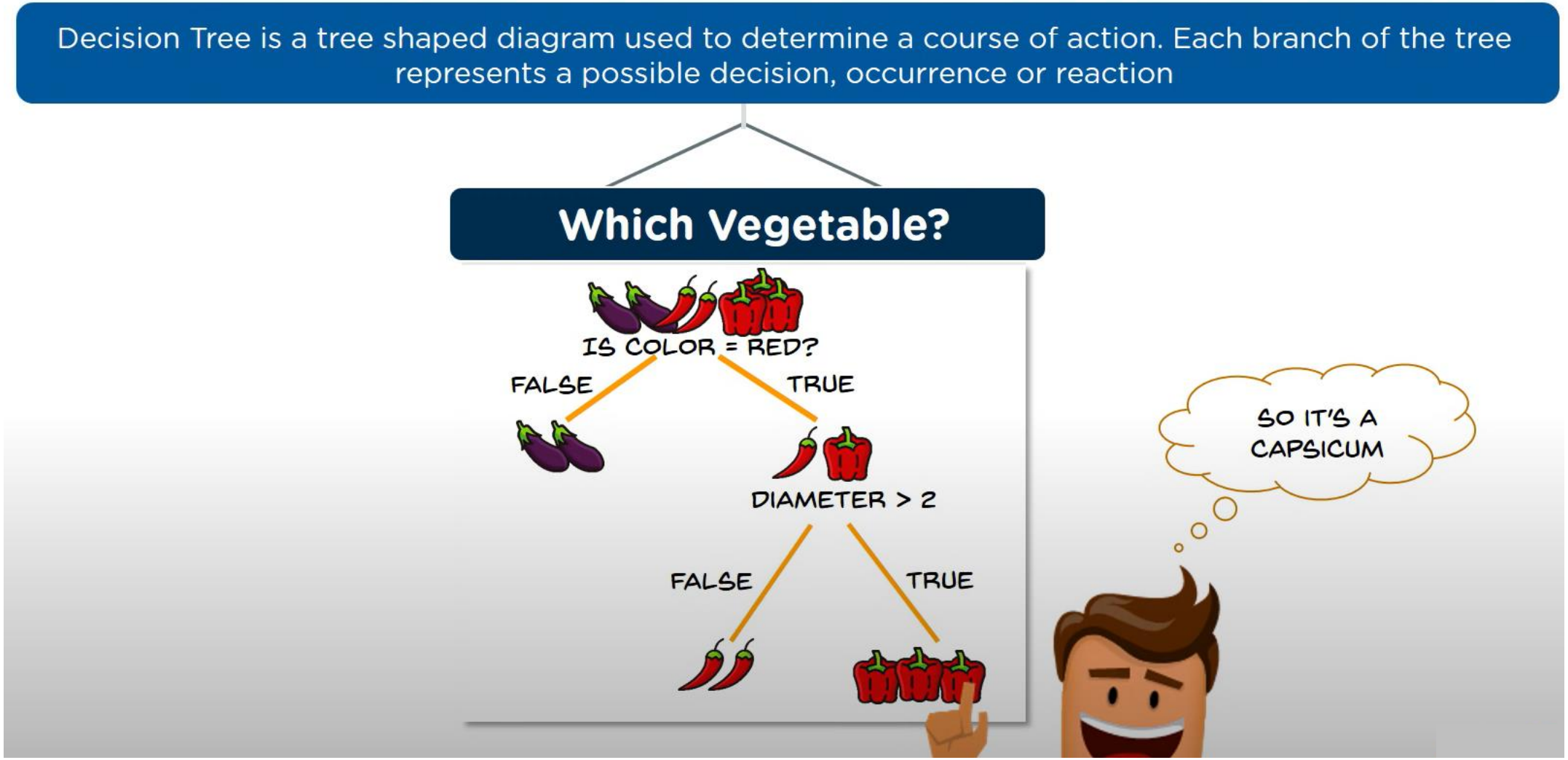
Metrics for the above mentioned algorithms can be

- **Gini Impurity (IG)**
- **Information entropy (IE)**
- **Variance reduction**

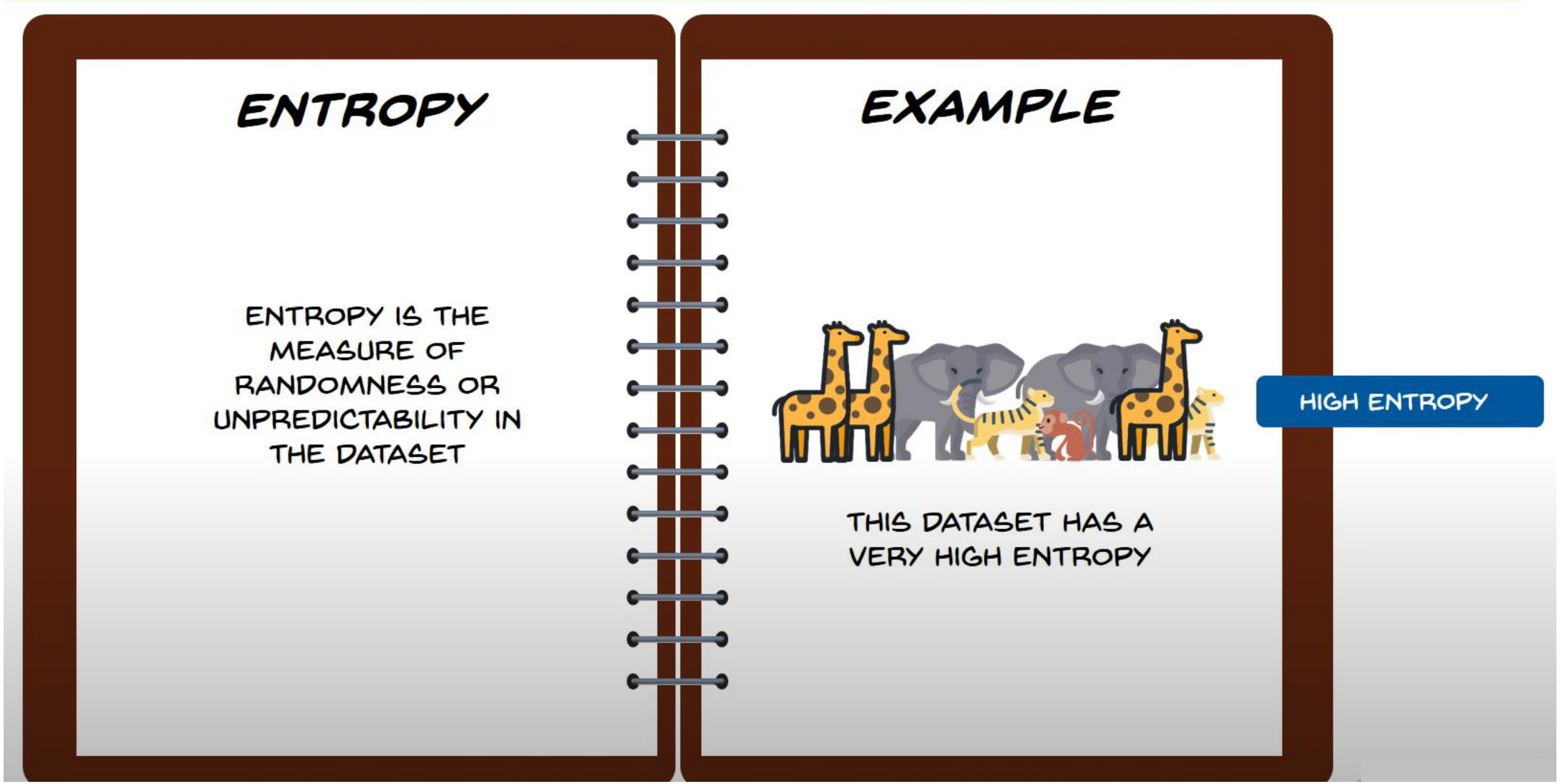
# What is a decision tree?

Decision Tree is a tree shaped diagram used to determine a course of action. Each branch of the tree represents a possible decision, occurrence or reaction

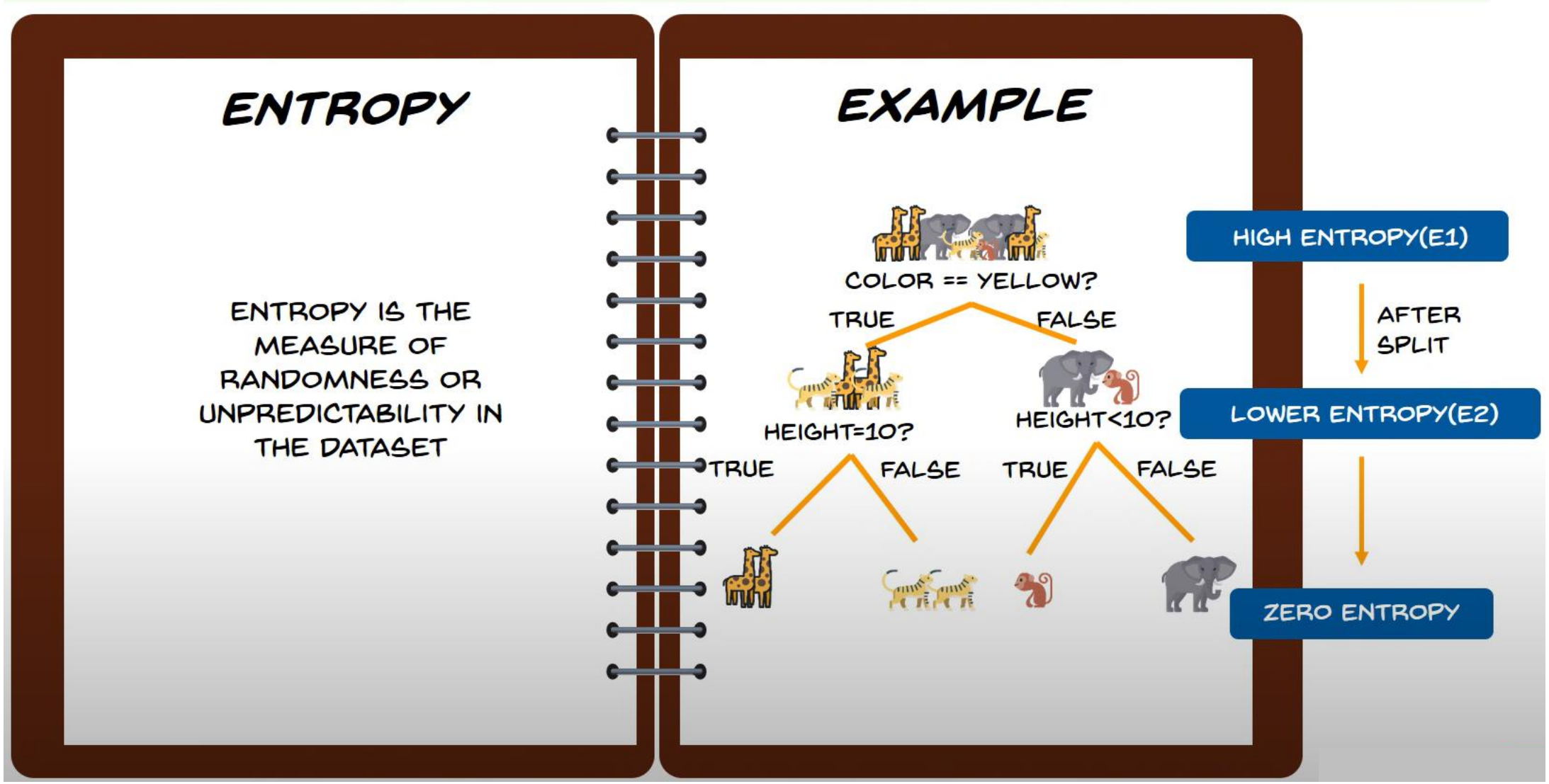
# What is a decision tree?



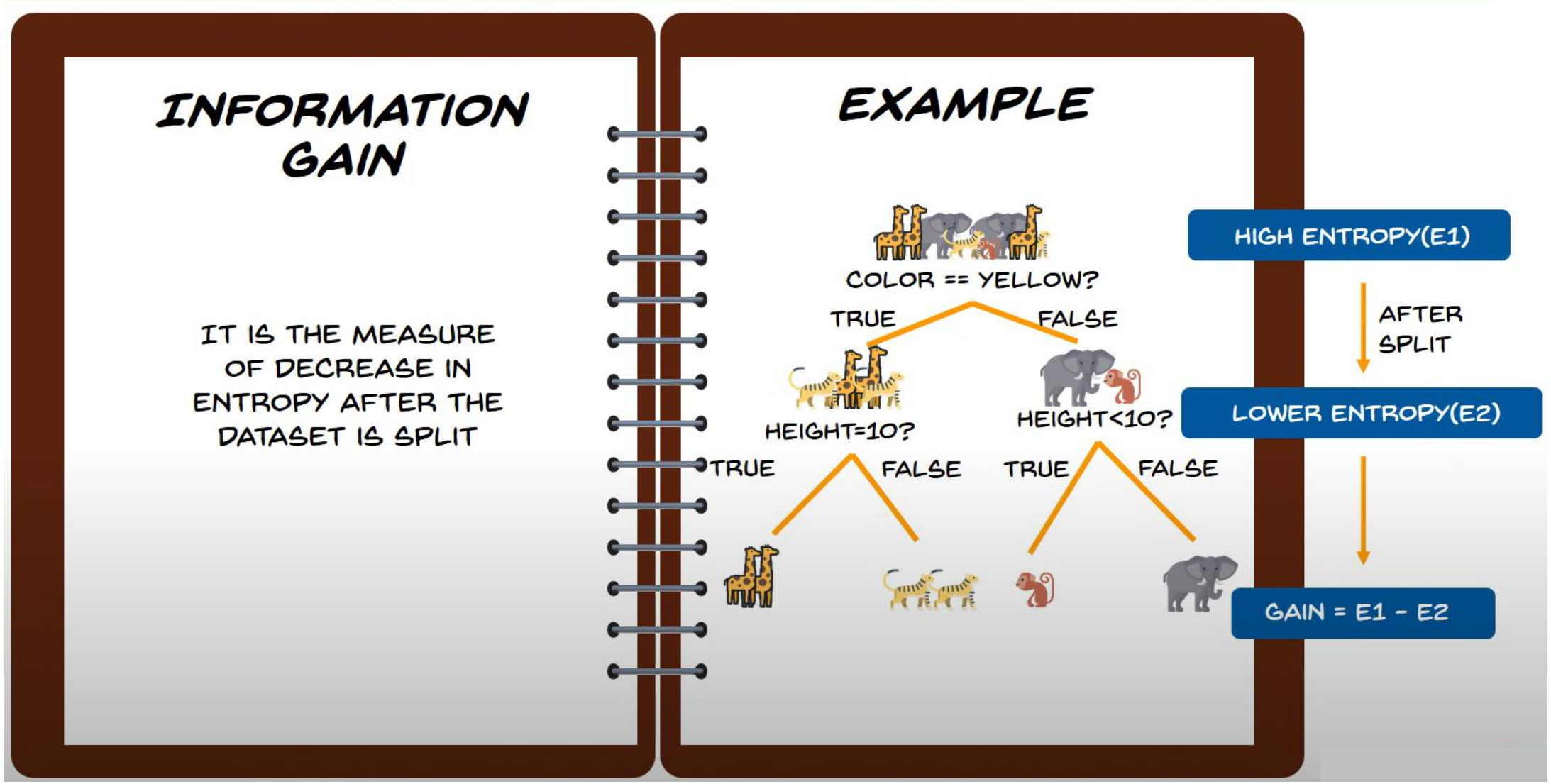
# Decision Tree – Important Terms



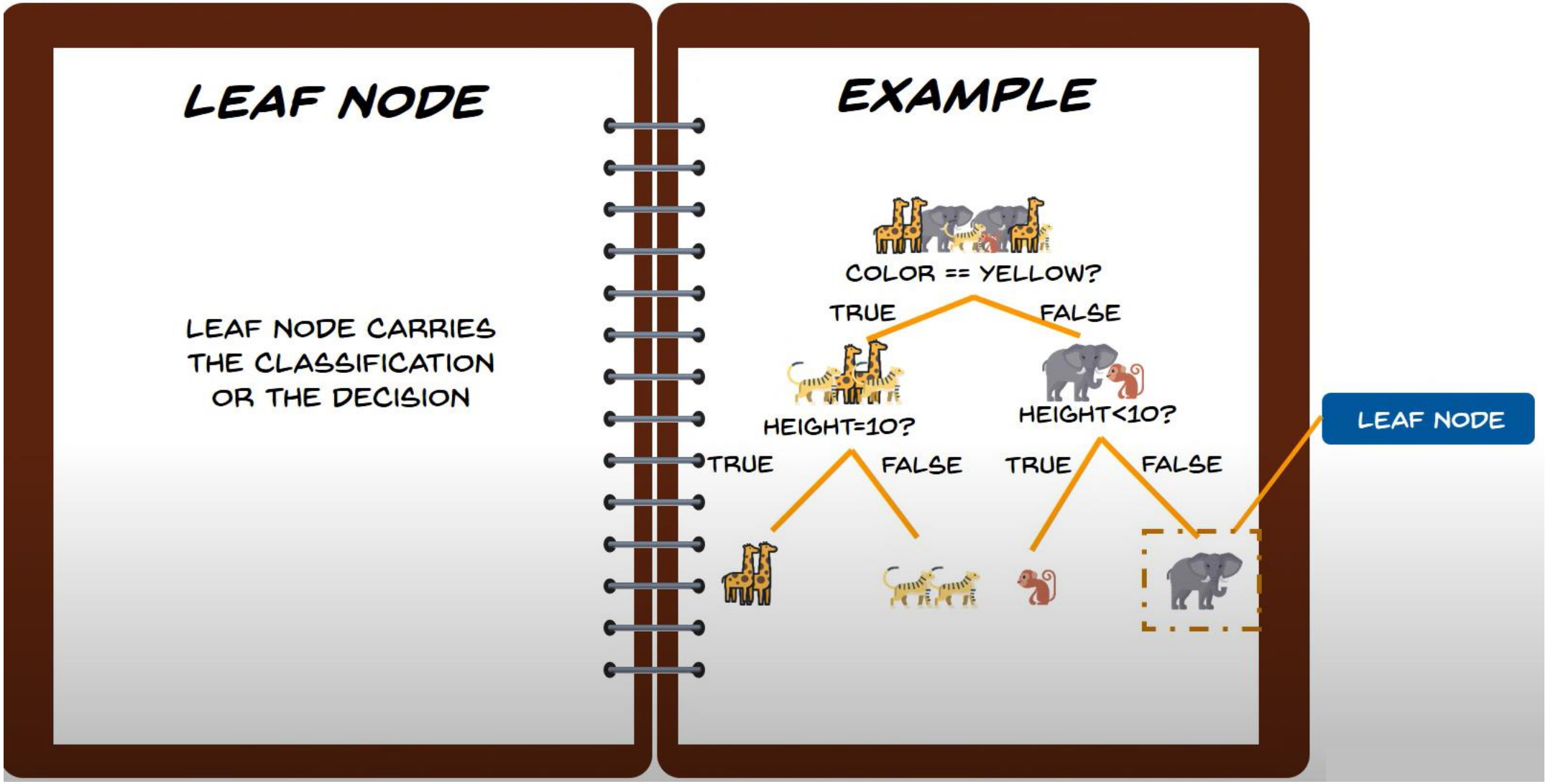
# Decision Tree – Important Terms



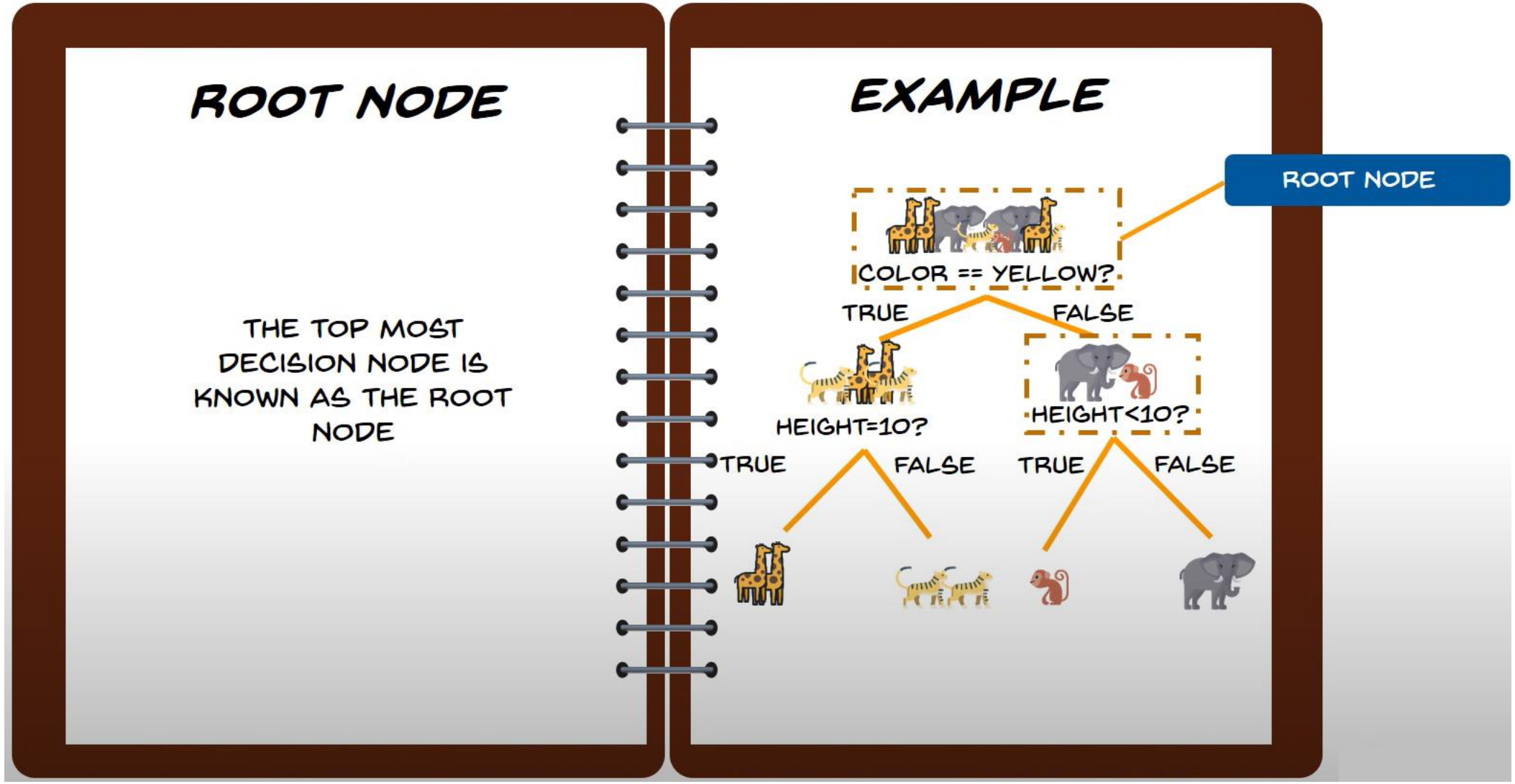
# Decision Tree – Important Terms



# Decision Tree – Important Terms

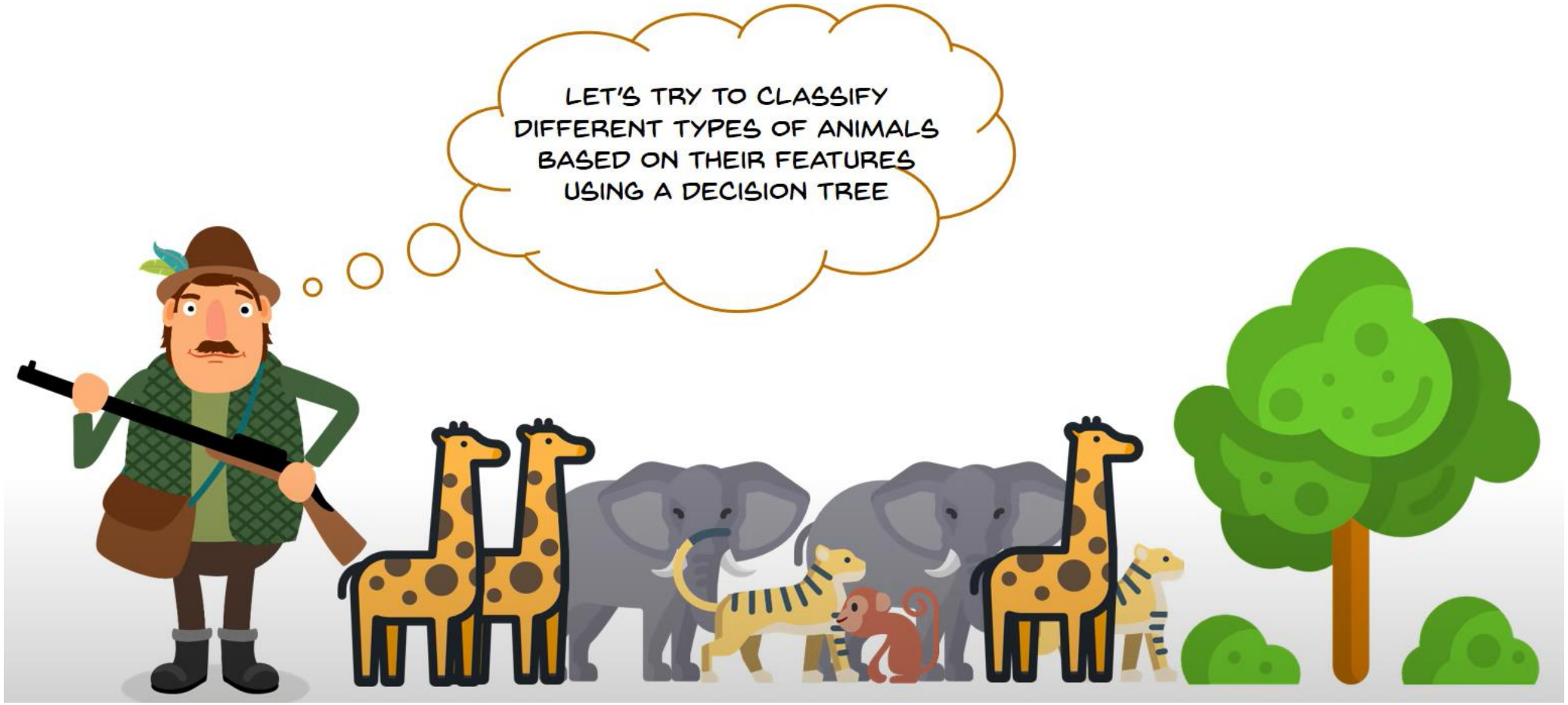


# Decision Tree – Important Terms



# How does a Decision Tree work?

# How does a Decision Tree work?



# How does a Decision Tree work?

**PROBLEM STATEMENT**

TO CLASSIFY THE DIFFERENT TYPES OF ANIMALS BASED ON THEIR FEATURES USING DECISION TREE

THE DATASET IS LOOKING QUITE MESSY AND THE ENTROPY IS HIGH IN THIS CASE

**TRAINING DATASET**

| COLOR | HEIGHT | LABEL |
|---|---|---|
| GREY | 10 | ELEPHANT |
| YELLOW | 10 | GIRAFFE |
| BROWN | 3 | MONKEY |
| GREY | 10 | ELEPHANT |
| YELLOW | 4 | TIGER |

# How does a Decision Tree work?

**HOW TO SPLIT THE DATA**

WE HAVE TO FRAME THE CONDITIONS THAT SPLIT THE DATA IN SUCH A WAY THAT THE INFORMATION GAIN IS THE HIGHEST

**NOTE**

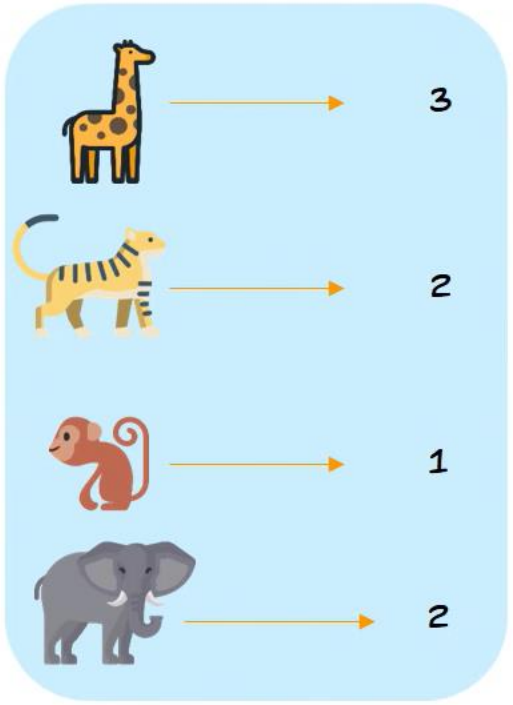
GAIN IS THE MEASURE OF DECREASE IN ENTROPY AFTER SPLITTING

**FORMULA FOR ENTROPY**

$$-\sum_{i=1}^{k} P(valuei).log_2(P(value_i)$$

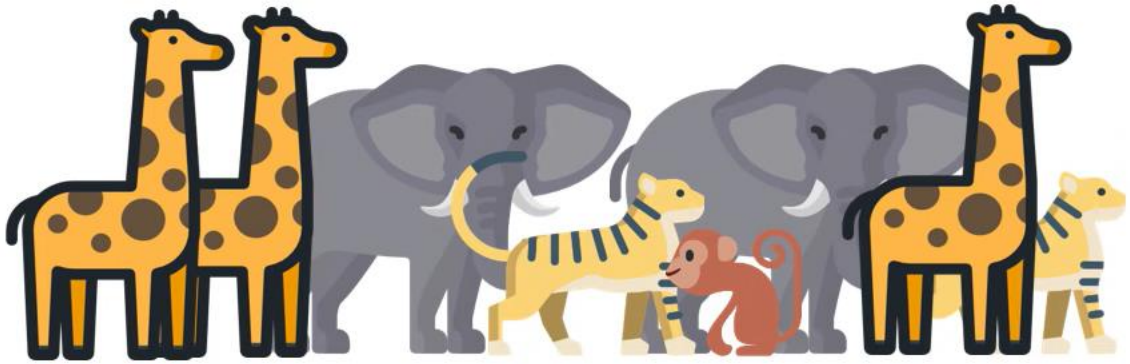
LET'S TRY TO CALCULATE THE ENTROPY FOR THE CURRENT DATASET

# How does a Decision Tree work?




LET'S USE THE FORMULA

$$-\sum_{i=1}^{k} P(valuei).log_2(P(value_i))$$



ENTROPY = $(\frac{3}{8})\ log_2(\frac{3}{8}) + (\frac{2}{8})\ log_2(\frac{2}{8}) + (\frac{1}{8})\ log_2(\frac{1}{8}) + (\frac{2}{8})\ log_2(\frac{2}{8})$

Entropy =1.904

WE WILL CALCULATE THE ENTROPY OF THE DATASET SIMILARLY AFTER EVERY SPLIT TO CALCULATE THE GAIN

GAIN CAN BE CALCULATED BY FINDING THE DIFFERENCE OF THE SUBSEQUENT ENTROPY VALUES AFTER SPLIT

# How does a Decision Tree work?

NOW WE WILL TRY TO CHOOSE A CONDITION THAT GIVES US THE HIGHEST GAIN

WE WILL DO THAT BY SPLITTING THE DATA USING EACH CONDITION AND CHECKING THE GAIN THAT WE GET OUT THEM.

THE CONDITION THAT GIVES US THE HIGHEST GAIN WILL BE USED TO MAKE THE FIRST SPLIT

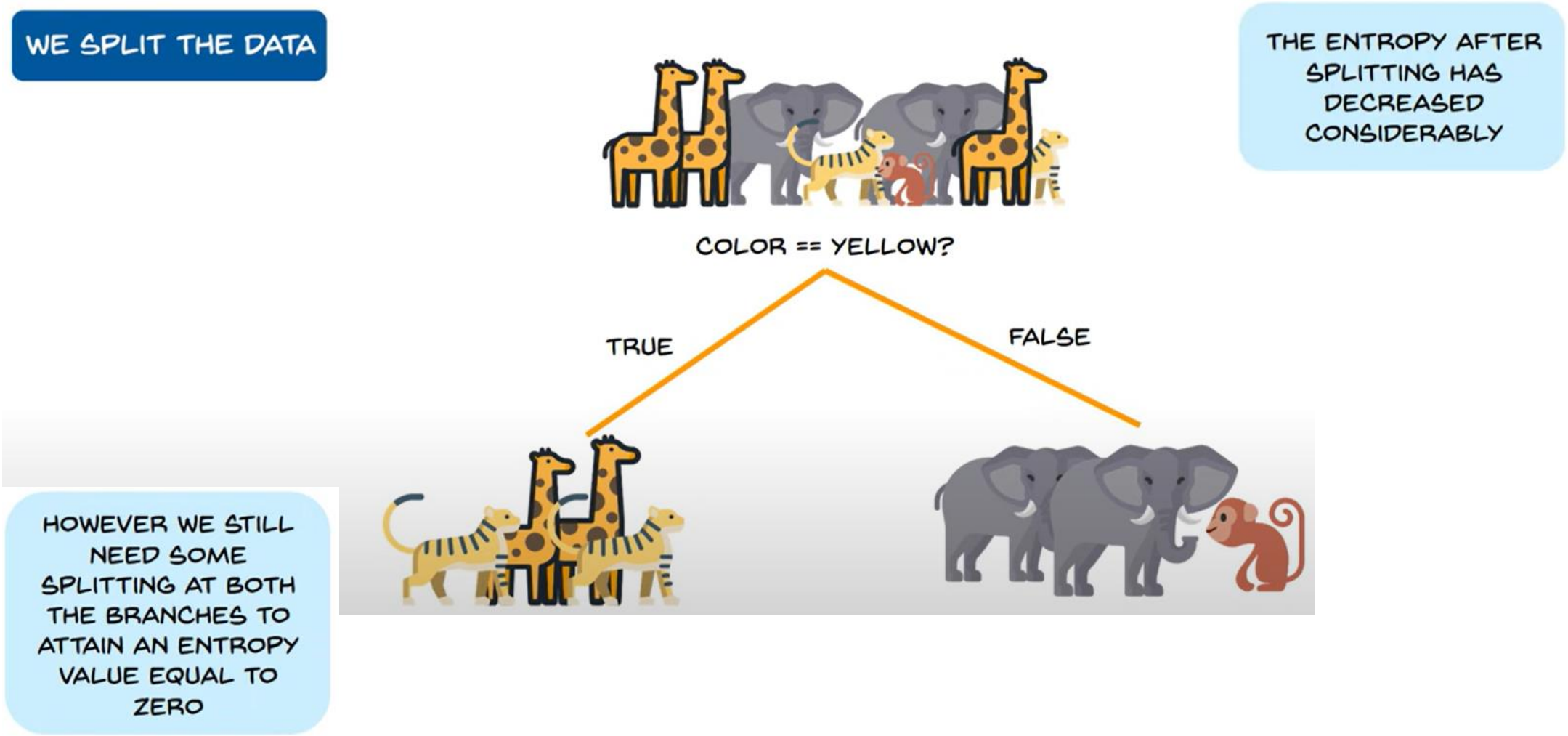
**CONDITIONS**

- COLOR== YELLOW?
- HEIGHT>=10
- COLOR== BROWN?
- COLOR==GREY
- DIAMETER<10

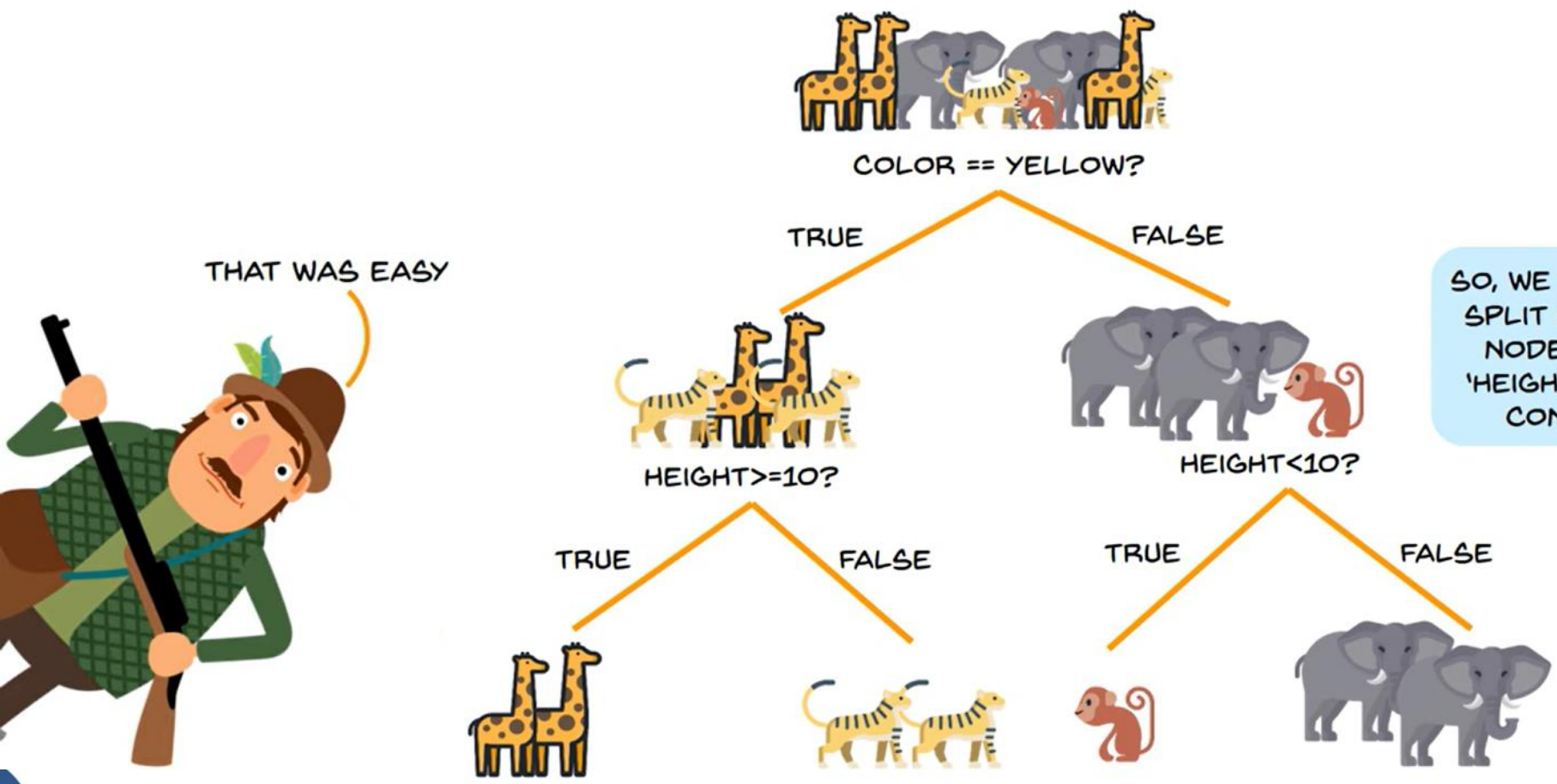
**TRAINING DATASET**

| COLOR | HEIGHT | LABEL |
|---|---|---|
| GREY | 10 | ELEPHANT |
| YELLOW | 10 | GIRAFFE |
| BROWN | 3 | MONKEY |
| GREY | 10 | ELEPHANT |
| YELLOW | 4 | TIGER |

# How does a Decision Tree work?



# How does a Decision Tree work?



# Metric for the decision tree

Let us consider Gini impurity as the metric,

Gini Impurity (GI) = $1 - \sum_{i=1}^{m} f_i^2$

Gini split index = GI(s) – $p_1$ * GI($s_1$) – p2* GI($s_2$)

Information entropy (IE) = $-\sum_{i=1}^{m} f_i * \log(f_i)$

where $f_i$ = fraction of class label $i$,

s – parent node, $s_1$ and $s_2$ are child nodes

$p_1$ & $p_2$ are split fractions

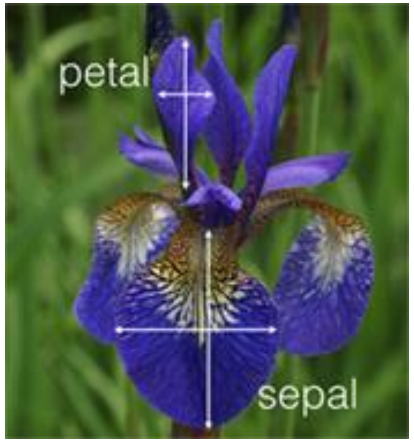
# Important rules for constructing trees

- Every parent node of higher Gini impurity / information entropy is split based on features in order to lower its Gini impurity (or information entropy or variance reduction in the case of regression trees).
- Gini impurity of pure sets = 0

- The split which corresponds to higher Gini split index is always preferred.
- Example: If split index 1 = 0.5 and split index 2 = 0.25, then split corresponding to split index 1 will be chosen.

# Describing the metric through an example

- Consider the following classification problem
- Discriminate between three different species of Iris flower
- The training data contains 49-setosa, 50-versicolor and 50- virginica species
- The features that are available are sepal length, sepal width, petal length and petal width
- The ranges for these feature values (in cm) are

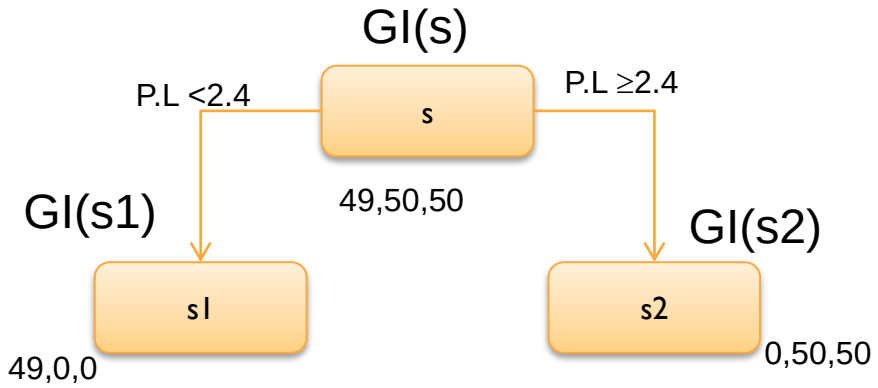




| | setosa | versicolor | virginica |
|---|---|---|---|
| S.L | [4.3,5.8] | [4.9,7] | [4.9,7.9] |
| S.W | [2.3,4.4] | [2,3.4] | [2.2,3.8] |
| P.L | [1,1.9] | [3,5.1] | [4.5,6.9] |
| P.W | [0.1,0.6] | [1,1.8] | [1.4,2.5] |

# Construction of nodes (level 1)

The training data contains 49-setosa, 50-versicolor and 50- virginica species, the root node could start with versicolor

**Possibility 1**




| | setosa | versicolor | virginica |
|---|---|---|---|
| S.L | [4.3,5.8] | [4.9,7] | [4.9,7.9] |
| S.W | [2.3,4.4] | [2,3.4] | [2.2,3.8] |
| P.L | [1,1.9] | [3,5.1] | [4.5,6.9] |
| P.W | [0.1,0.6] | [1,1.8] | [1.4,2.5] |

By choosing petal length as splitting feature,

2.4 is considered as the mid point and the splitting criteria.

In doing so we can split setosa into a completely pure data set.

$GI(s) = 1- (49/149)^2 – (50/149)^2 –(50/149)^2 = 0.66$

$GI(s1) = 1- (49/49)^2 = 0$

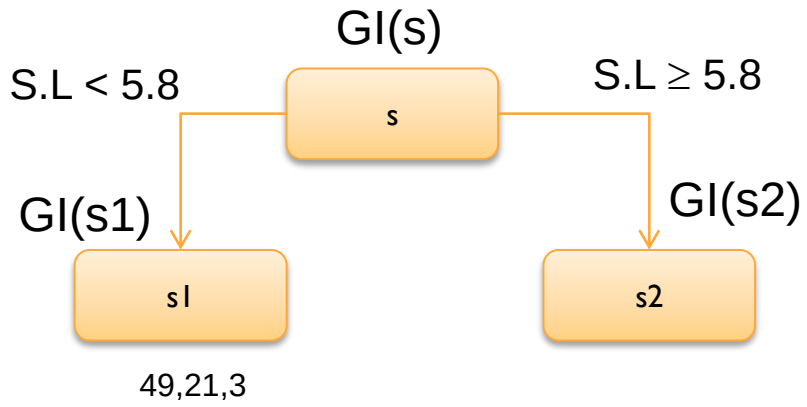
$GI(s2) = 1 – (50/100)^2 – (50/100)^2 = 0.5$

Gini split index = $0.66 – (49/149)(0) – (100/149)\ (0.5) = 0.324$

where $f_i$ = fraction of class label $i$,
s – parent node, $s_1$ and $s_2$ are child nodes, $p_1$ & $p_2$ are split fractions

# Construction of node (level1) contd.

**Possibility 2**: What happens if we split based on sepal length,




Possibility 2

| | setosa | versicolor | virginica |
|---|---|---|---|
| S.L | [4.3,5.8] | [4.9,7] | [4.9,7.9] |
| S.W | [2.3,4.4] | [2,3.4] | [2.2,3.8] |
| P.L | [1,1.9] | [3,5.1] | [4.5,6.9] |
| P.W | [0.1,0.6] | [1,1.8] | [1.4,2.5] |

By choosing sepal length as splitting feature,

It is observed that range values overlap, so in this case consider the edge value that corresponds to high split index. Therefore split value can be 5.8 or 4.9

$GI(s) = 1 - (49/149)^2 - (50/149)^2 - (50/149)^2 = 0.66$

$GI(s1) = 1 - (49/73)^2 - (21/73)^2 - (3/73)^2 = 0.4650$

$GI(s2) = 1 - (29/76)^2 - (47/76)^2 = 0.4720$

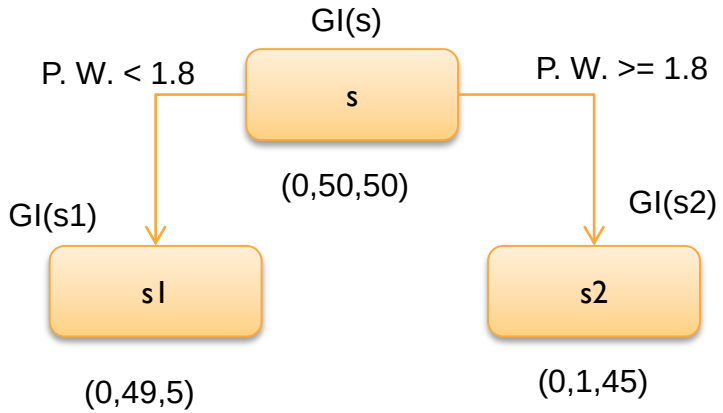
Gini split index = $0.66 - (73/149)(0.465) - (76/149)(0.472) = 0.1915$

**If split value is S.L =4.9, the split index = 0.0746**

# Construction of nodes (level 2)

Considering all such possible splitting, for level 1, we identify that splitting according to P.L is the best

**Level 2**




Possibility 1

By choosing petal width as splitting feature,

It is observed that range values overlap, so in this case consider the edge value that corresponds to high split index. Therefore split value can be 1.8 or 1.4

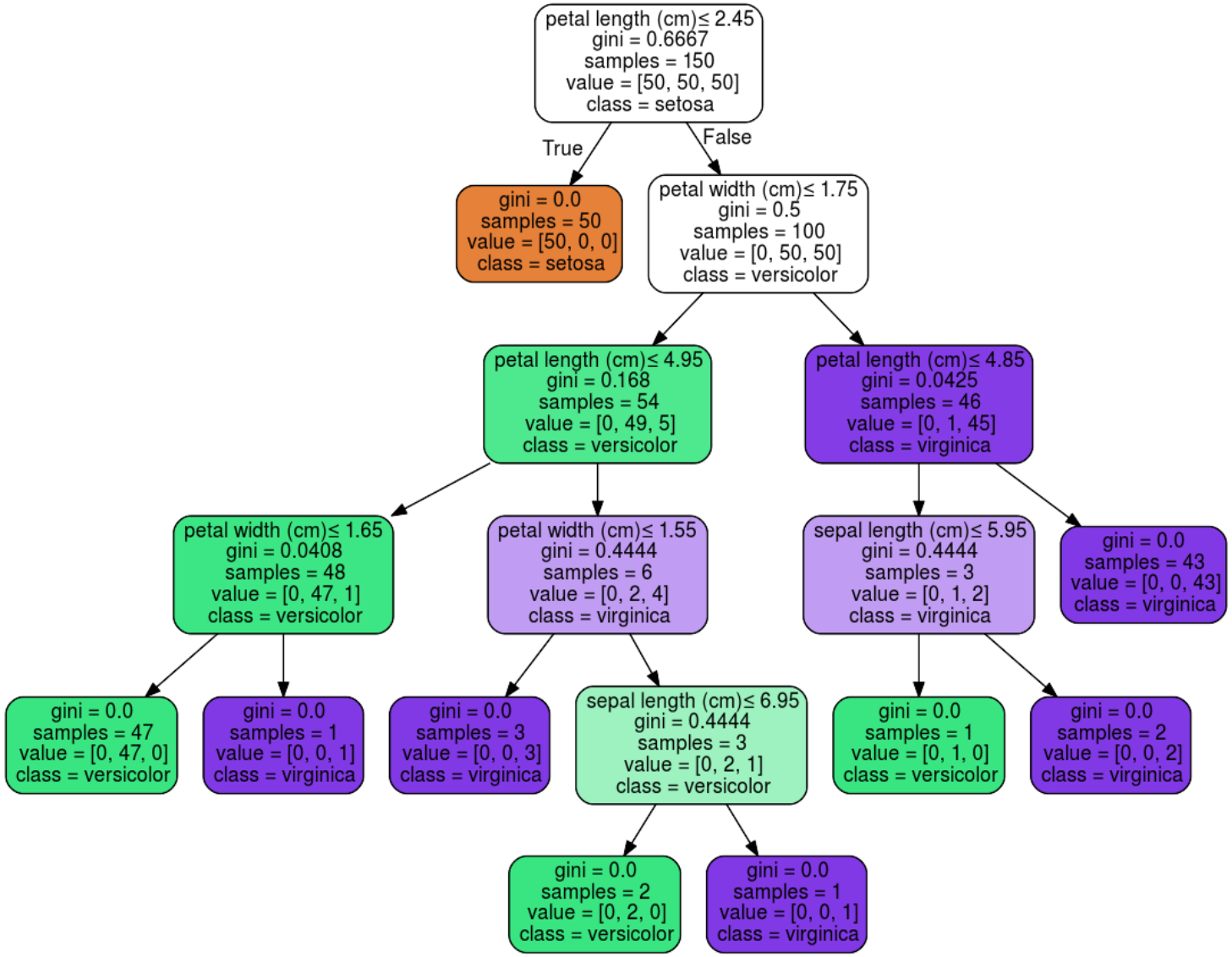
$GI(s) = 1 - (50/100)^2 - (50/100)^2 = 0.5$

$GI(s1) = 1 - (49/54)^2 - (5/54)^2 = 0.168$

$GI(s2) = 1 - (1/46)^2 - (45/46)^2 = 0.0425$

Gini split index $= 0.5 - (54/100)(0.168) - (46/100)(0.0425) = 0.3897$

# More deeper tree



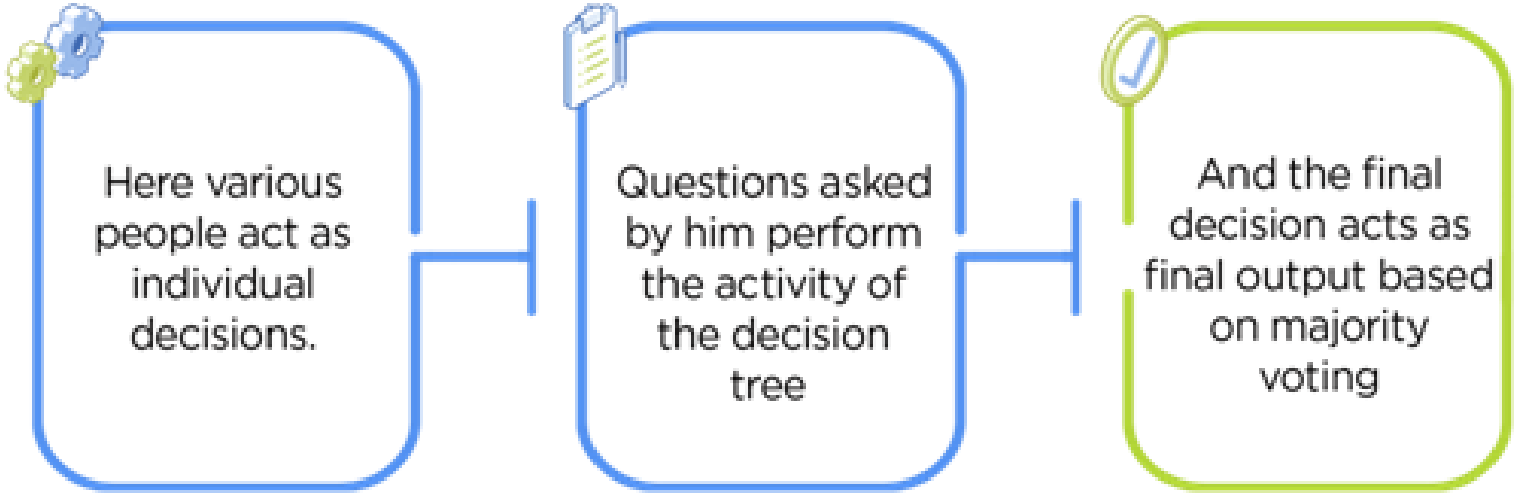
# Random Forests

# What is a random forest ?

- One of the ensemble techniques that bags decision trees from multiple subsets of given data.
- It is used for regression / classification problems.
- It aims to reduce overfitting to the training data set.
- The algorithm consists of 2 parts:
  - Split the data set into many subsets based on its features and then build a decision tree classifier
  - Bag all the classifiers obtained from every subset and classify the test data
  - Based on voting or average method classify the data

# Real Life Analogy

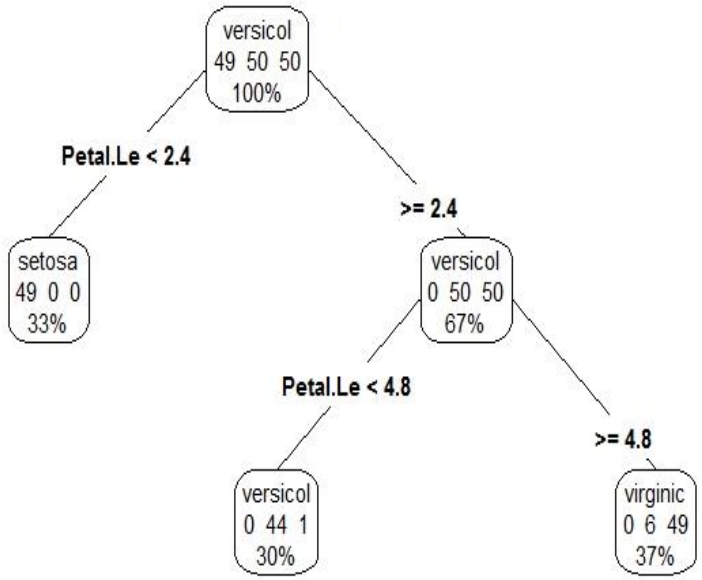
- A student named X wants to choose a course after his 10+2, and he is confused about the choice of course based on his skill set.
- So he decides to consult various people like his cousins, teachers, parents, degree students, and working people.
- He asks them varied questions like why he should choose, job opportunities with that course, course fee, etc. Finally, after consulting various people about the course he decides to take the course suggested by most of the people.



# Example continued (subset 1)

Considering only a subset of given training data,

| Sepal.Length | Sepal.Width | Petal.Length | Species |
|---|---|---|---|
| 5.1 | 3.5 | 1.4 | setosa |
| 4.9 | 3 | 1.4 | setosa |
| 4.7 | 3.2 | 1.3 | setosa |
| 4.6 | 3.1 | 1.5 | setosa |
| . | . | . | . |
| . | . | . | . |
| 6.8 | 3.2 | 5.9 | virginica |
| 6.7 | 3.3 | 5.7 | virginica |
| 6.7 | 3 | 5.2 | virginica |
| 6.3 | 2.5 | 5 | virginica |
| 6.5 | 3 | 5.2 | virginica |
| 6.2 | 3.4 | 5.4 | virginica |
| 5.9 | 3 | 5.1 | virginica |

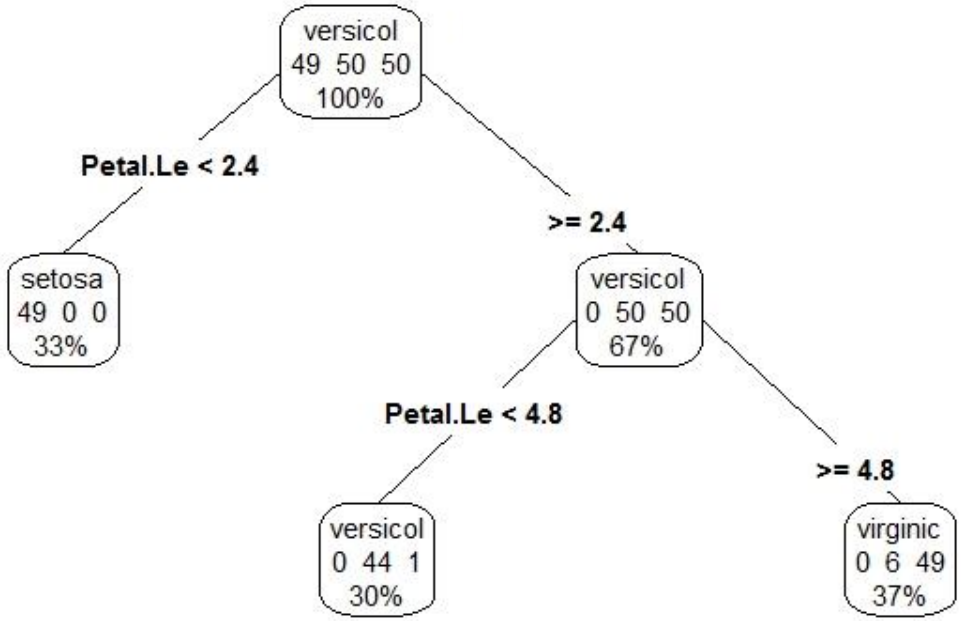



Test data = [S.L = 4.9, S.W = 3, P.L = 1.4]
Prediction of given test data is "Setosa"

# Example continued (subset 2)

Considering only a subset of given training data,

| Sepal.Length | Petal.Length | Species |
|---|---|---|
| 5.1 | 1.4 | setosa |
| 4.9 | 1.4 | setosa |
| 4.7 | 1.3 | setosa |
| 4.6 | 1.5 | setosa |
| . | . | . |
| . | . | . |
| 6.8 | 5.9 | virginica |
| 6.7 | 5.7 | virginica |
| 6.7 | 5.2 | virginica |
| 6.3 | 5 | virginica |
| 6.5 | 5.2 | virginica |
| 6.2 | 5.4 | virginica |
| 5.9 | 5.1 | virginica |



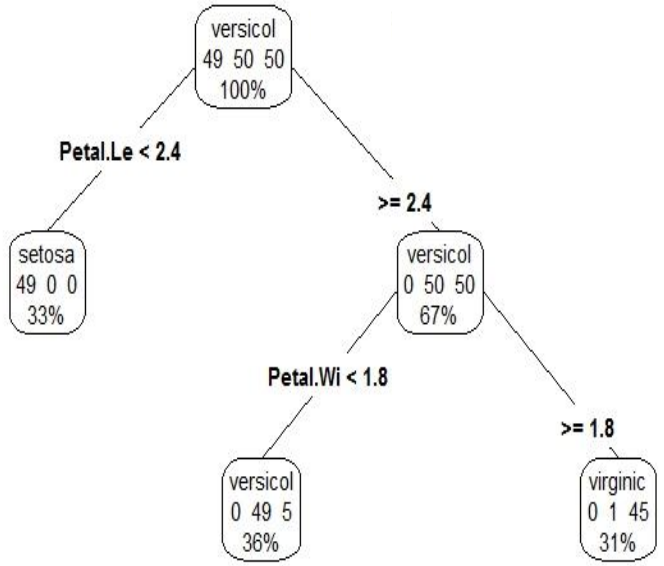

Test data = [S.L = 4.9, P.L = 1.4]

Prediction of given test data is “Setosa”

# Example continued (subset 3)

Considering only a subset of given training data,

| Petal.Length | Petal.Width | Species |
|---|---|---|
| 1.4 | 0.2 | setosa |
| 1.4 | 0.2 | setosa |
| 1.3 | 0.2 | setosa |
| 1.5 | 0.2 | setosa |
| . | . | . |
| . | . | . |
| 5.9 | 2.3 | virginica |
| 5.7 | 2.5 | virginica |
| 5.2 | 2.3 | virginica |
| 5 | 1.9 | virginica |
| 5.2 | 2 | virginica |
| 5.4 | 2.3 | virginica |
| 5.1 | 1.8 | virginica |




Test data = [P.L = 1.4, P.W = 0.2]

Prediction of given test data is "Setosa"

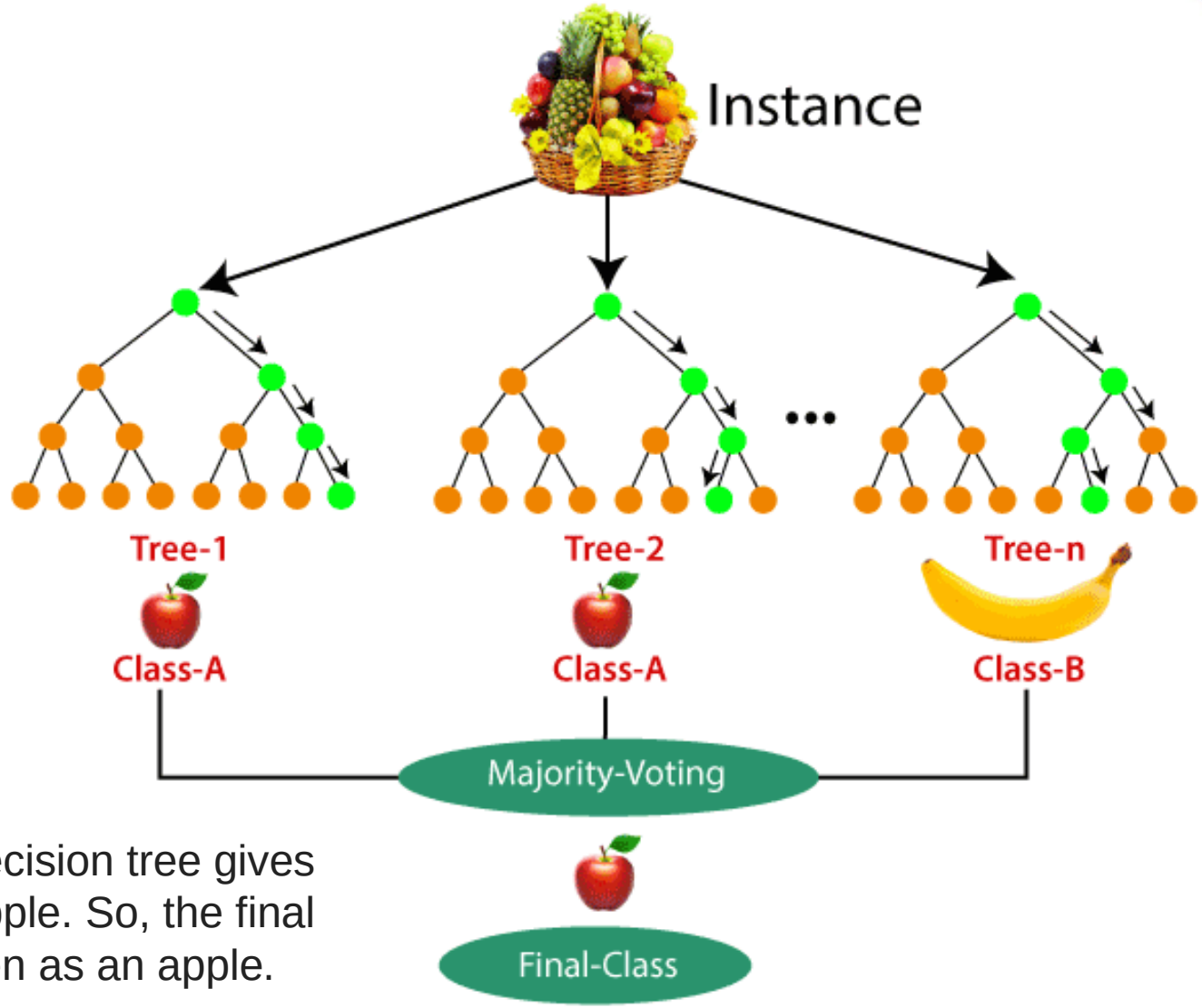
# Final Thoughts

Random forest = function(given data, subset1,..,subset 3)

- The final decision observed from 3 different subsets using gini impurity split methods are {“setosa”, “setosa”, “setosa”}
- Therefore according to voting method, random forest function classifies the test data as “Setosa”

# Example

‘n’ number of samples are taken from the fruit basket and an individual decision tree is constructed for each sample.




The majority decision tree gives output as an apple. So, the final output is taken as an apple.

# Performance measures

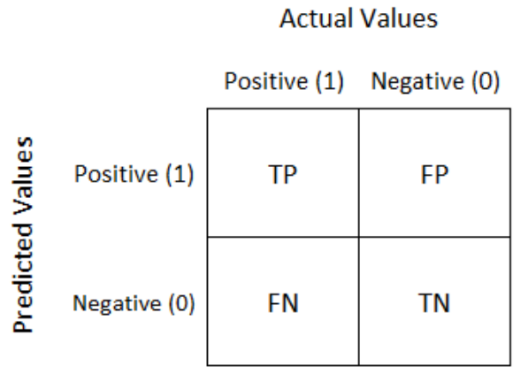
# Measures of accuracy

- Terminology
  - $TP \rightarrow$ true positives, $TN \rightarrow$ true negatives,
  - $FP \rightarrow$ false positives, $FN \rightarrow$ false negatives
  - $N = TP + TN + FP + FN$
  - TP – Correct identification of positive labels
  - TN – Correct identification of negative labels
  - FP – Incorrect identification of positive labels
  - FN – Incorrect identification of negative labels

# Confusion matrix

| | | True condition | |
|---|---|---|---|
| | Total population | Condition positive | Condition negative |
| Predicted condition | Predicted condition positive | **True positive**, Power | **False positive**, Type I error |
| | Predicted condition negative | **False negative**, Type II error | **True negative** |

Source: https://en.wikipedia.org/wiki/Receiver_operating_characteristic



# Example

## True negative


A picture of a pizza, correctly labeled as not-hotdog.

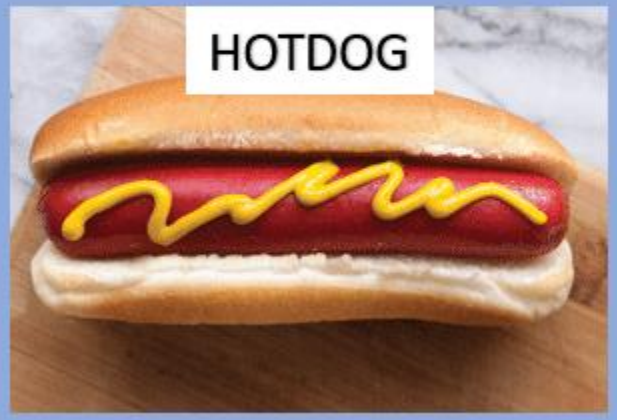
## False positive


A picture of a dog, incorrectly labeled as hotdog.

## False negative


A picture of a hotdog, incorrectly labeled as not hotdog.

## True positive




A picture of a hotdog, correctly labeled hotdog.

# Measures of accuracy

- Accuracy: Overall effectiveness of a classifier
  - $A = \frac{TP+TN}{N}$
  - Maximum value that accuracy can take is $1$
  - This happens when the classifier exactly classifies two groups (i.e., $FP = 0$ and $FN = 0$)
- Remember
  - Total number of true positive labels $= TP+FN$
- Similarly
  - Total number of true negative labels $= TN+FP$

# Some Issues

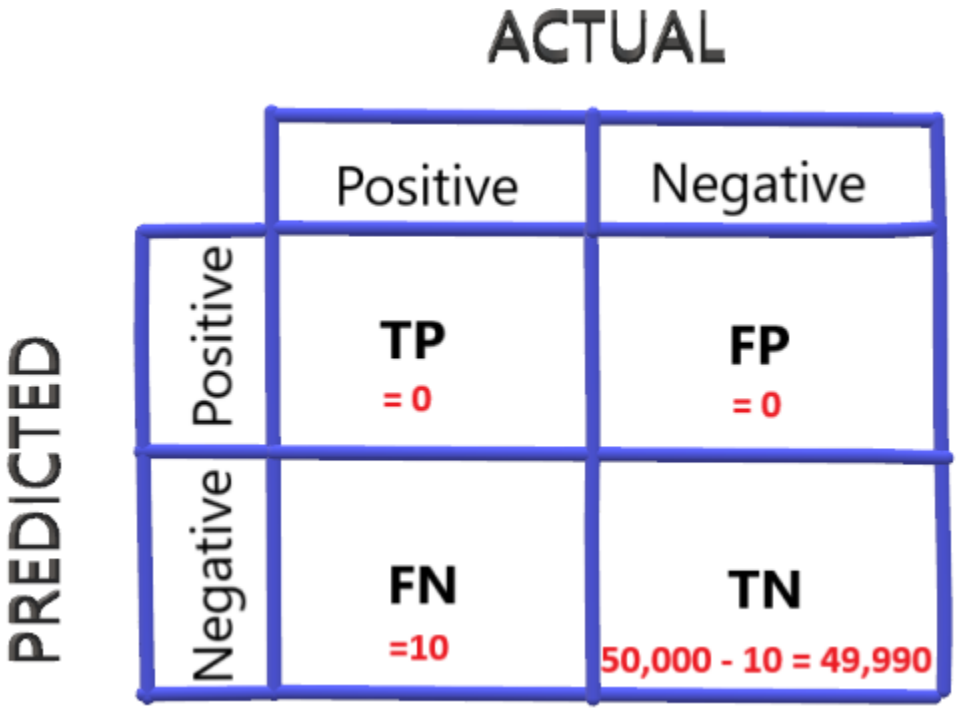
- Accuracy is a good measure when the classes in the data are nearly balanced.
  - Example: All the classes of flowers in iris have same number of data points.

- Accuracy is useful when the target class is ***well balanced*** but is not a good choice for the unbalanced classes.
  - Imagine the scenario where we had 99 images of the dog and only 1 image of a cat present in our training data. Then our model would always predict the dog, and therefore we got 99% accuracy.
  - In reality, Data is always imbalanced for example Spam email, credit card fraud, and medical diagnosis.
  - Hence, if we want to do a better model evaluation and have a full picture of the model evaluation, other metrics such as recall and precision should also be considered.

# Example

- Now, let's consider 50,000 passengers travel per day on an average. Out of which, 10 are actually COVID positive.

- One of the easy ways to increase accuracy is to **classify every passenger as COVID negative**. So our confusion matrix looks like:




*Accuracy in this context is a terrible measure.*

Accuracy for this case will be:

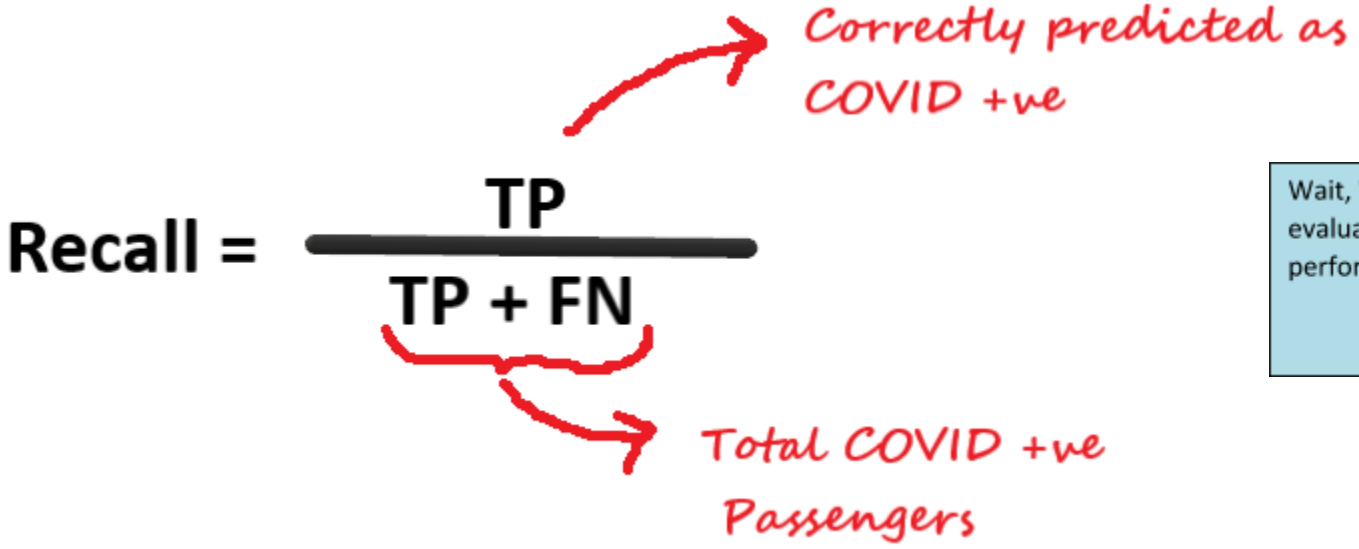
Accuracy = 49,990/50,000 = 0.9998 or 99.98%.

**Impressive!!! Right?**

Well, does that really solve our purpose of classifying COVID positive passengers correctly?

Not labeling 10 of actually positive is a lot more expensive.

# Recall (Sensitivity or True Positive rate)




Wait, Wait!! Before considering recall as a good measure for evaluation. Just think: Is recall alone good enough to evaluate the performance of a classification model?

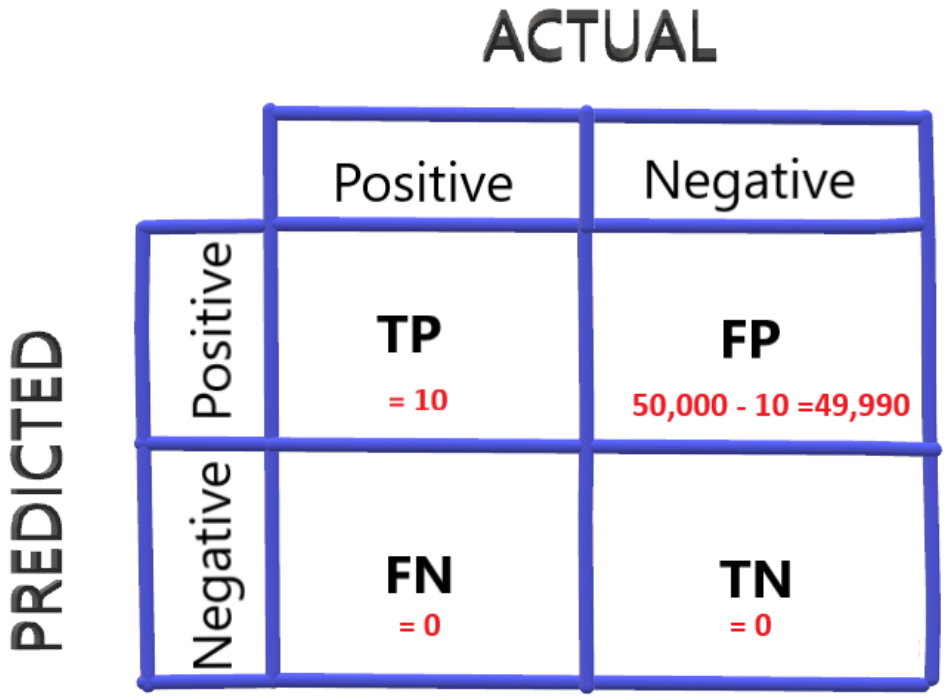
- Out of all positive passengers what fraction we identified correctly. Going back to our previous strategy of labeling every passenger as negative → will give recall of Zero.

**Recall = 0/10 = 0**

- So, in this context, Recall is a good measure. It says that the terrible strategy of identifying every passenger as COVID negative leads to zero recall. And we want to maximize the recall.

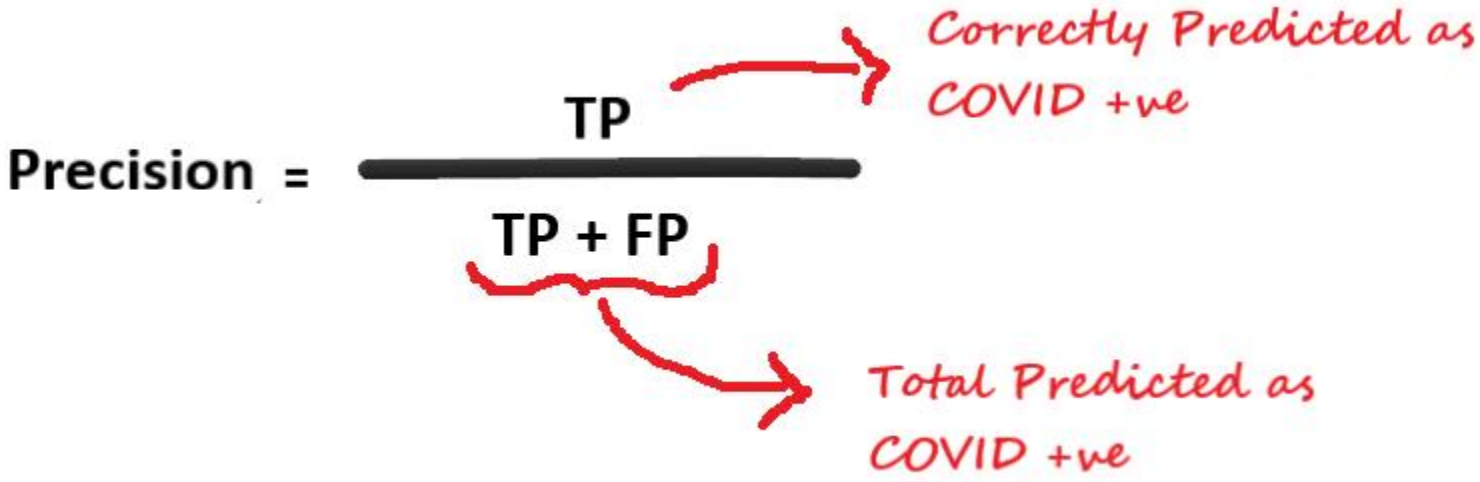
# Some Issues

- Consider another scenario of labeling every passenger as COVID positive.
- Everybody walks into the airport and the model just labels them as positive.
- Labeling every passenger as positive is bad in terms of the amount of cost that needs to be spent in actually investigating each one before they board the flight.




- **Recall = 10/(10+0) = 1**
- That's a huge problem. So concluding, it turns out that accuracy was a bad idea because labeling everyone as negative can increase the accuracy but hoping Recall will be a good measure in this context but then realized that labeling everyone as positive will increase recall as well.
- So recall independently is not a good measure.

# Precision




- Considering our second bad strategy of labeling every passenger as positive, the precision would be :

**Precision = 10 / (10 + 49990) = 0.0002**

- While this bad strategy has a good recall value of **1** but it has a terrible precision value of **0.0002** (we want maximum precision).
- This clarifies that recall alone is not a good measure, we need to consider precision value (also vice-versa).

# Prediction Measures

- Sensitivity (Recall): Effectiveness of a classifier to identify positive labels
  - $S_e = \frac{TP}{TP + FN}$ (true positive rate)
- Specificity: Effectiveness of a classifier to identify negative labels
  - $S_p = \frac{TN}{FP + TN}$
- Both $S_e$ and $S_p$ lie between $0\ and\ 1, 1$ is an ideal value for each of them
- Precision
  - Fraction of the true positives in the predicted positives
  - Precision = $\frac{TP}{TP + FP}$
  - Ratio of number of true positives to the number of predicted positives

# $F_1$ score

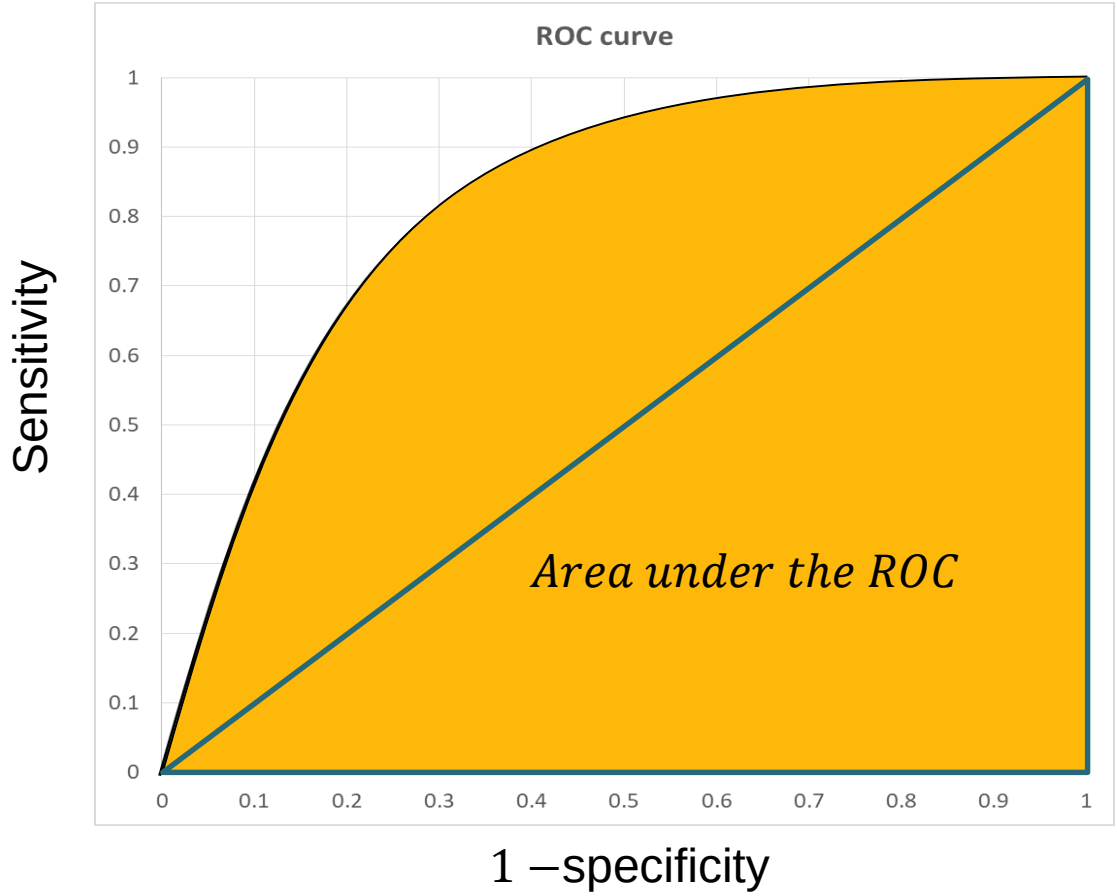
- The balance between the precision and sensitivity/recall.
- $F_1$ score = 2 * Precision * Recall / (Precision + Recall)
- The range for $F_1$ score is [0,1].
- The higher the **$F_1$ score** the better, with 0 being the worst possible and 1 being the best.

# Other Measures of accuracy

- Observed accuracy
  - $OA = \frac{a+d}{N}$
- Expected accuracy
  - $EA = \frac{(a+c)(a+b)+(b+d)(c+d)}{N}$
- Kappa $= \frac{\frac{(a+d)}{N} - \left(\frac{(a+c)(a+b)+(b+d)(c+d)}{N}\right)}{\left(1 - \left(\frac{(a+c)(a+b)+(b+d)(c+d)}{N}\right)\right)}$
  - where $a, b, c$ $and$ $d$ are $TP, FP, FN$ $and$ $TN$ respectively

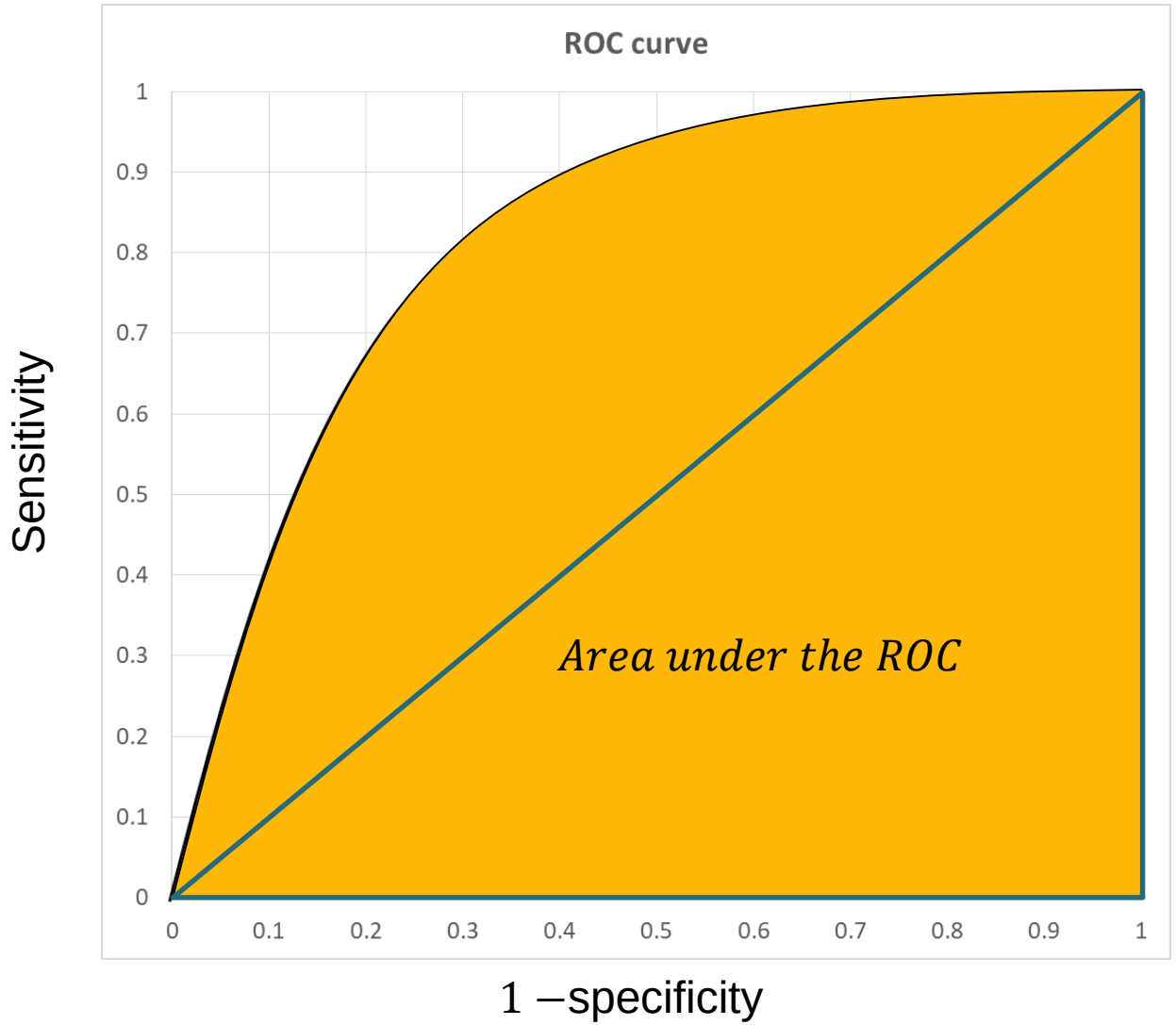
# ROC

- ROC –An acronym for Receiver Operating Characteristics
- Originally developed and used in signal detection theory
- ROC graph:
  - Sensitivity as a function of specificity
  - sensitivity (Y-axis) and 1 −specificity (X-axis)



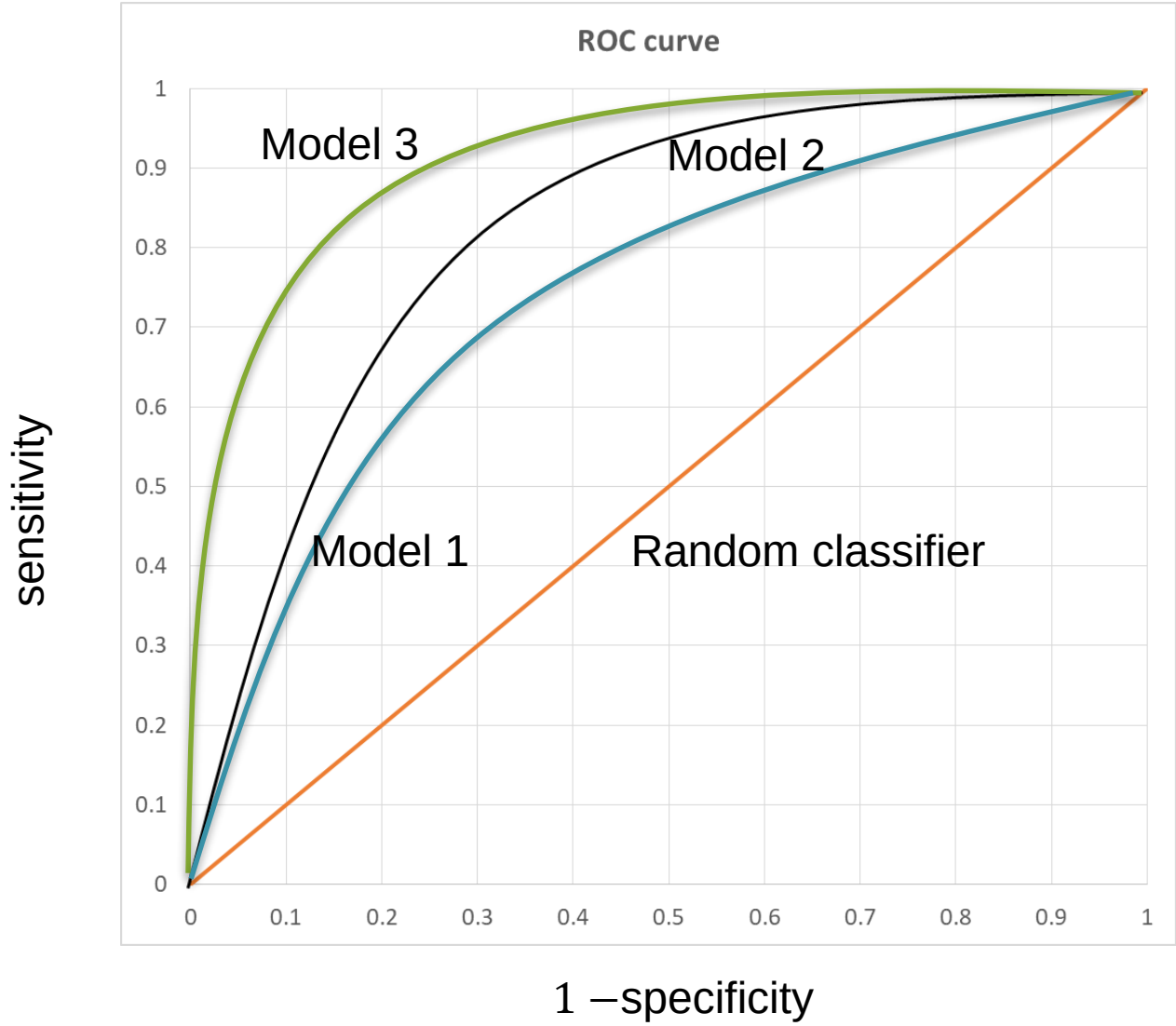
# ROC

- ROC can be used to
  - See the classifier performance at different threshold levels (from 0 to 1)
  - AUC- Area under the ROC
    - An area of 1 represents a perfect test; an area of 0.5 represents a worthless model.
    - .90 – 1 = excellent
    - .80 – .90 = good
    - .70 – .80 = fair
    - .60 – .70 = poor
  - AUC < 0.5, check whether your labels are marked in opposite



# ROC

- ROC can be used to
  - Compare different classifiers at one threshold or overall threshold levels
  - Performance
  - Model 3 > Model 2 > Model 1




operation == "MIRROR_X":
irror_mod.use_x = True
irror_mod.use_y = False
irror_mod.use_z = False
operation == "MIRROR_Y":
irror_mod.use_x = False
irror_mod.use_y = True
irror_mod.use_z = False
operation == "MIRROR_Z":
irror_mod.use_x = False
irror_mod.use_y = False
irror_mod.use_z = True
selection at the end -add
_ob.select= 1
er_ob.select=1
ntext.scene.objects.acti
("Selected" + str(modifier
irror_ob.select = 0
bpy.context.selected_ob
ata.objects[one.name].se
int("please select exact

# THANK YOU