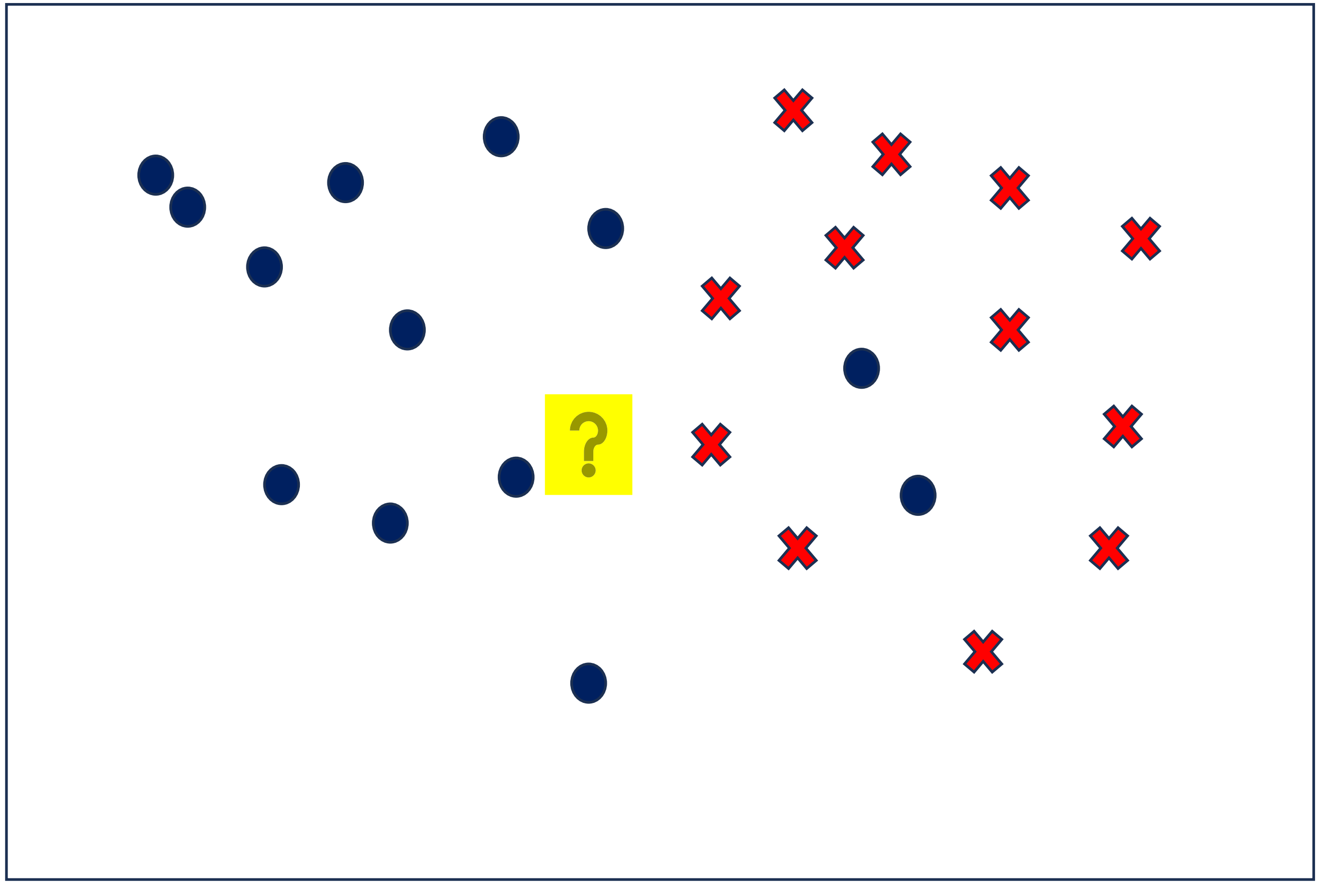
Supervised Learning Algorithm

K-Nearest Neighbor Algorithm

Supervised Learning algorithm: Instance based algorithm

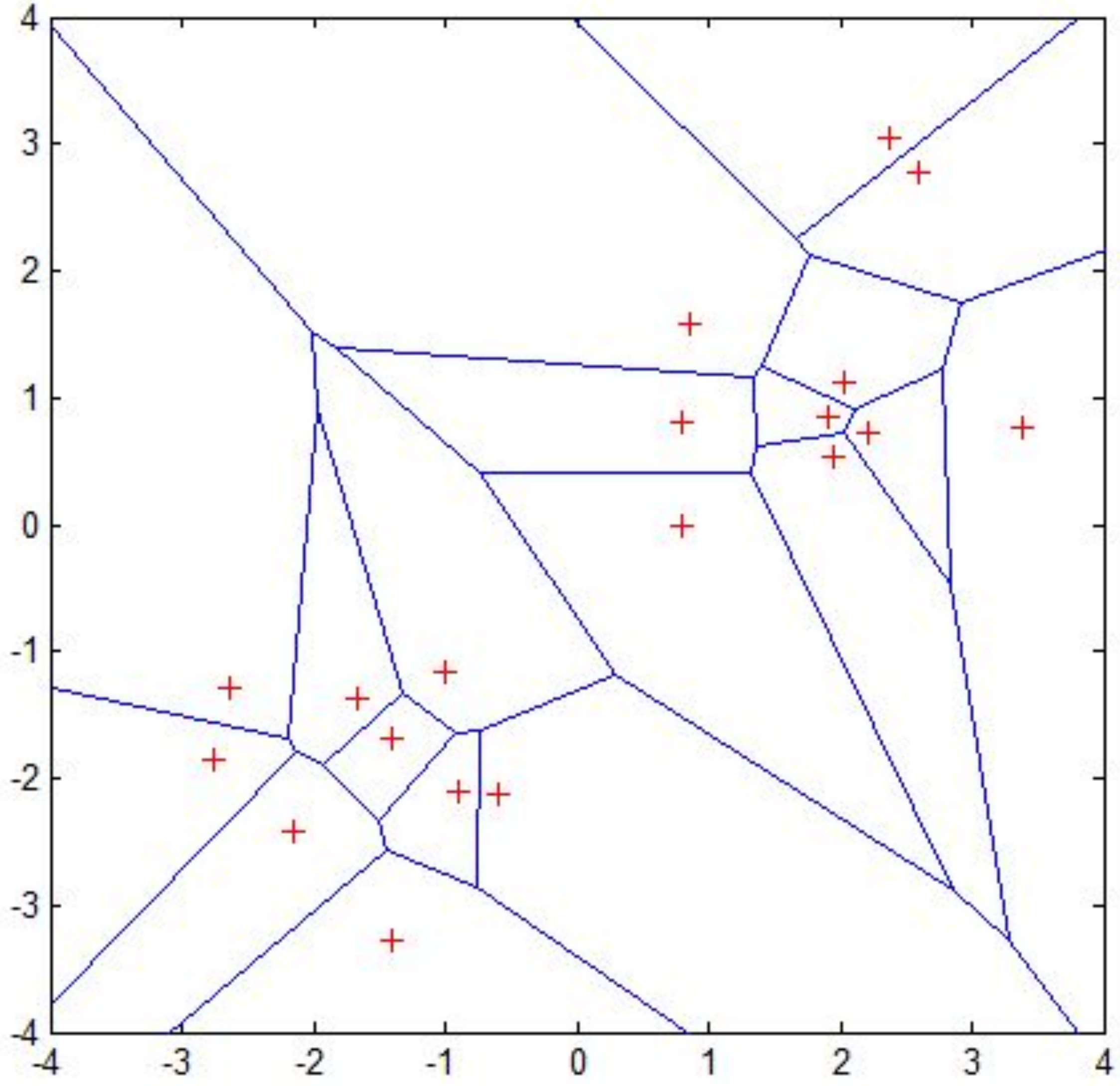
- $given\ (x^i, y^i)\ it\ tried\ to\ estimate\ f(x^i)$
- In instance based algorithm (Lazy learning model) instead of approximation of $f(x^i)$, we do store the training examples.
- It means that lazy learning model will not come up with the model priory, but store the instance of the training examples into a memory.

Instance based learning



K-nearest neighbors of a record ? are data points that have the k smallest distance to ?

Voronoi Diagram: Decision Boundary



Properties:

1) All possible points within a sample's Voronoi cell are the nearest neighboring points for that sample
2) For any sample, the nearest sample is determined by the closest Voronoi cell edge

Basic K-Nearest Neighbor Algorithm

- Training Phase
  - Store training examples into a memory
- Testing Phase / Prediction time
  - Find the k training examples $(x_1, y_1), (x_2, y_2), \ldots \ldots \ldots (x_k, y_k)$which are nearest to test instance (x_t)
- Classification problem:
  - Take the nearest y out of k nearest neighbor
- Regression problem
  - Take a average of y's of k nearest neighbor.

K-NN algorithm

- Compute distance matric between test data point and all the labelled data points.
- Sort labelled data points in ascending order based on computed distance
- Select top K labelled points and observe the class label.
- Assign the majority class label to test data

Standard Distance function for K-NN

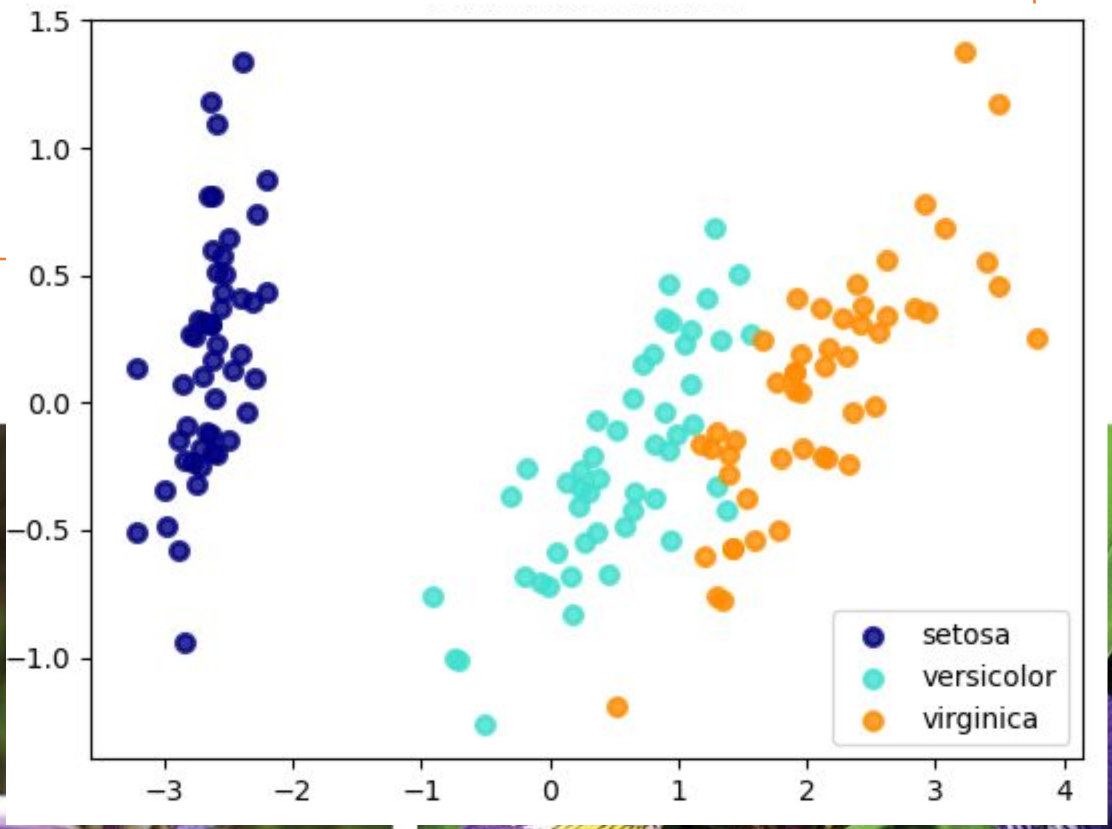
- Finding the closest point is the important part for the implementation of K-NN algorithm.
- Here, we use **Euclidian distance** to find the distance between two training examples.
- Given $x_i = (x_{i1}, x_{i2}, x_{i2} \dots \dots .., x_{in})$ and $x_j = \left(x_{j1}, x_{j2}, x_{j2} \dots .., x_{jn}\right)$
- $D\left(x_i, x_j\right) = \sqrt{\sum_{a=1}^{n} (x_{ia} - x_{ja})^2}$

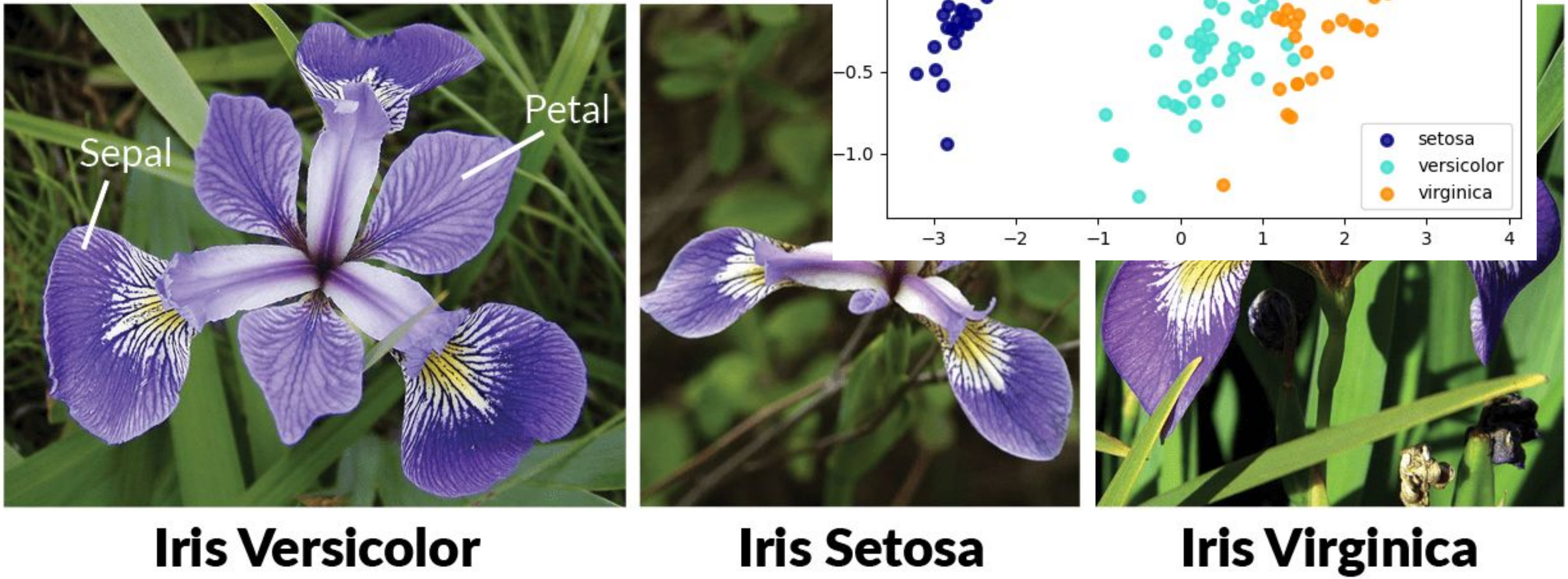
Other Distance Measure

- Minkowski Distance (p=p)
- Manhattan Distance (Taxicab or City Block) (p=1)
- Hamming Distance (Binary numbers)
- Euclidian Distance (p=2)

$$\left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

Example: Iris Dataset

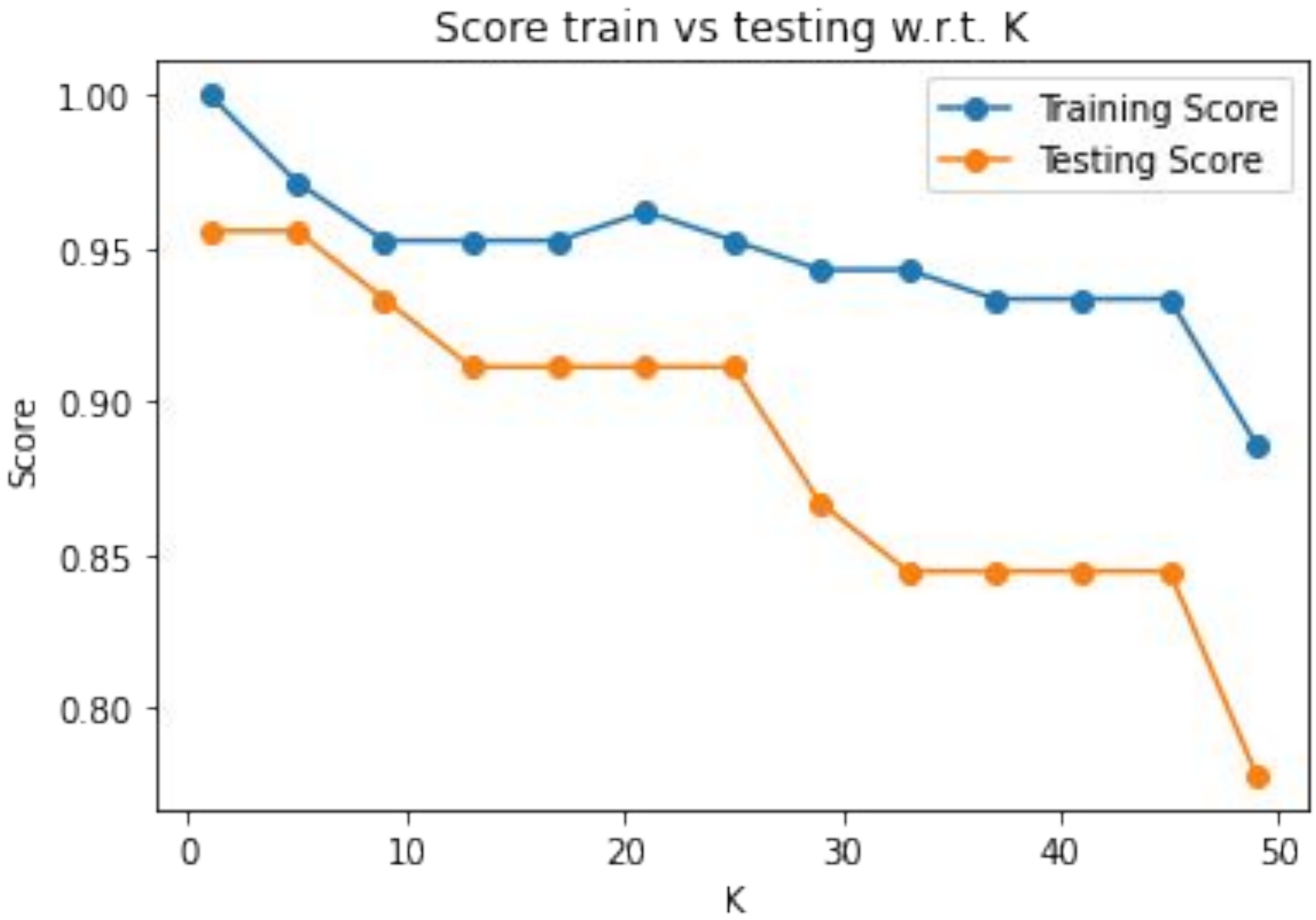




```python
from sklearn.datasets import load_iris
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
iris = load_iris()
```

```python
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier

X_train,X_test, y_train,y_test = train_test_split(iris.data,iris.target, test_size=0.30)
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train,y_train)
x= knn.score(X_test,y_test)
print(x)
```



- Different values of K impact the training and testing scores.
- Optimal value of k can also be identified using k-fold cross validation.

Some imp points about KNN

- Use equal weight to all features if;
  - No noise in the attributes
  - Attributes are at the same scale
  - Equal importance of all the features
- What if there is a noise/unscaled features/ unequal importance?
  - Use Large K value
  - Use weighted Euclidean distance algorithm
- Small K Captures fine decision boundary between the classes. Useful for small dataset.
- Large K less sensitive to noise (Class noise), larger training set

Weighted Euclidean distance

- $D(x_i, x_j) = \sqrt{\sum_{a=1}^{n} w_a (x_{ia} - x_{ja})^2}$
- Large weight → if feature is more important
- Small weight → if feature is less important
- Zero weight → if features has no importance

Supervised Machine Learning Algorithm

Decision Tree Classification

Tree Ensemble

Decision Tree

- A decision tree is **a non-parametric supervised learning algorithm for classification and regression tasks**.
- A decision tree is a **hierarchical model** used in decision support that depicts decisions and their potential outcomes.
- The tree structure is comprised of a root node, branches, internal nodes, and leaf nodes, forming a hierarchical, tree-like structure.

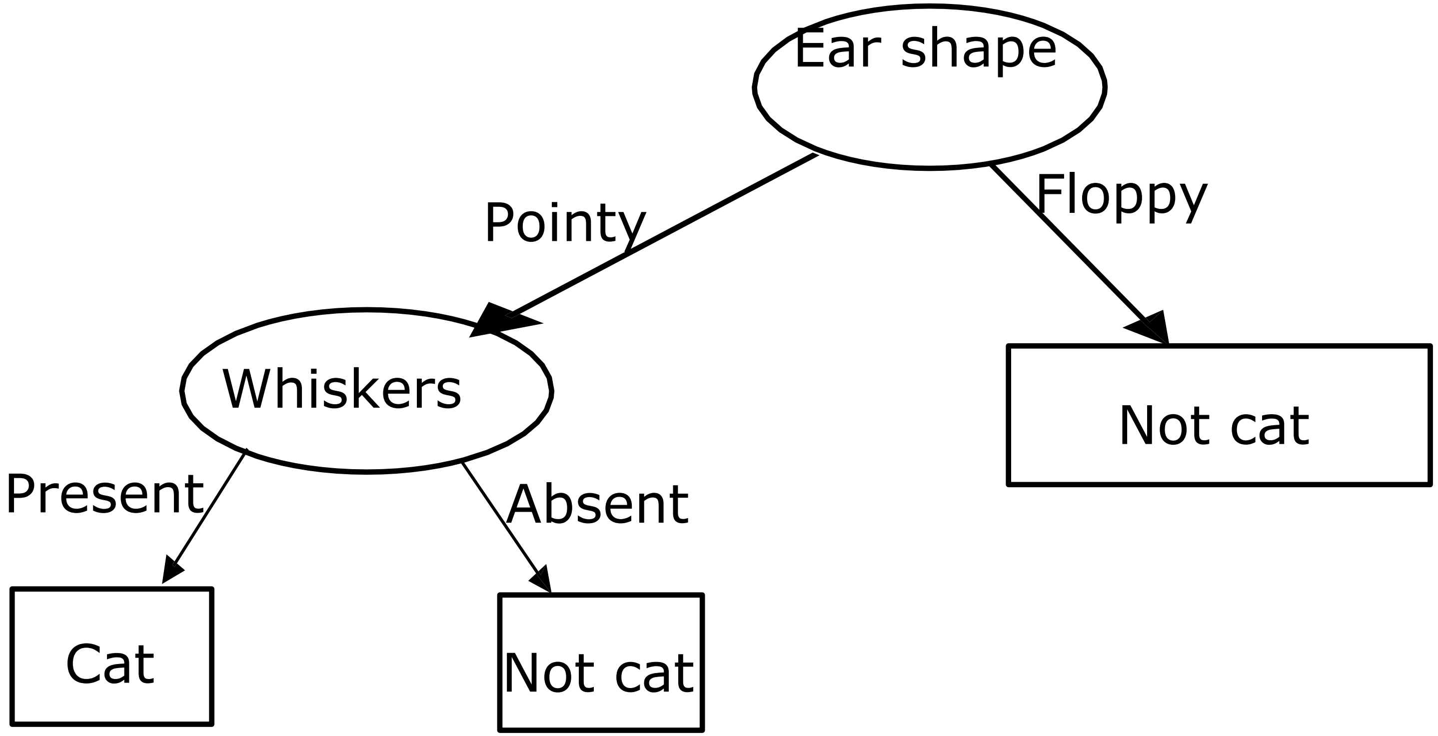
Cat classification example

Ear shape (x_1)	Face shape (x_2)	Whiskers (x_3)	Cat
Pointy	Round	Present	1
Floppy	Not round	Present	1
Floppy	Round	Absent	0
Pointy	Not round	Present	0
Pointy	Round	Present	1
Pointy	Round	Absent	1
Floppy	Not round	Absent	0
Pointy	Round	Absent	1
Floppy	Round	Absent	0
Floppy	Round	Absent	0

Categorical (discrete values)

Source: Sandford Online

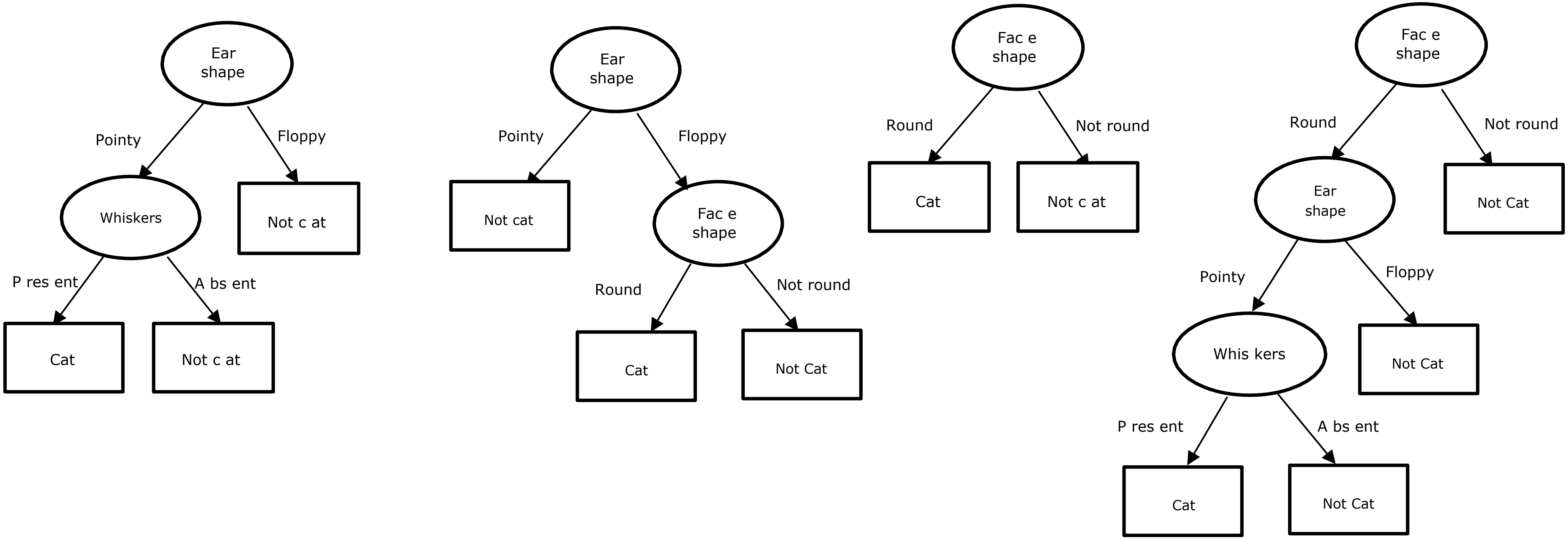
Decision Tree



New Test Example

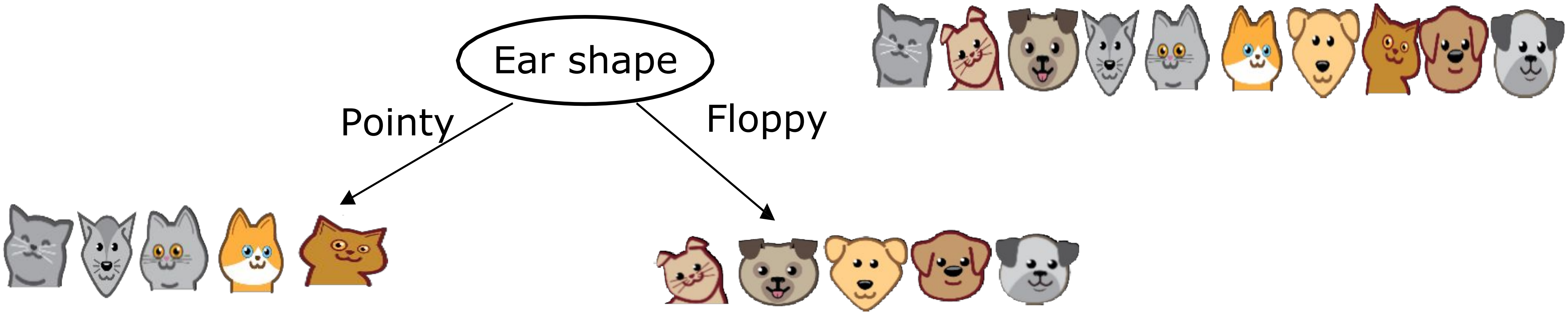
Ear shape: Pointy
Face shape: Round
Whiskers: Present

Decision Tree

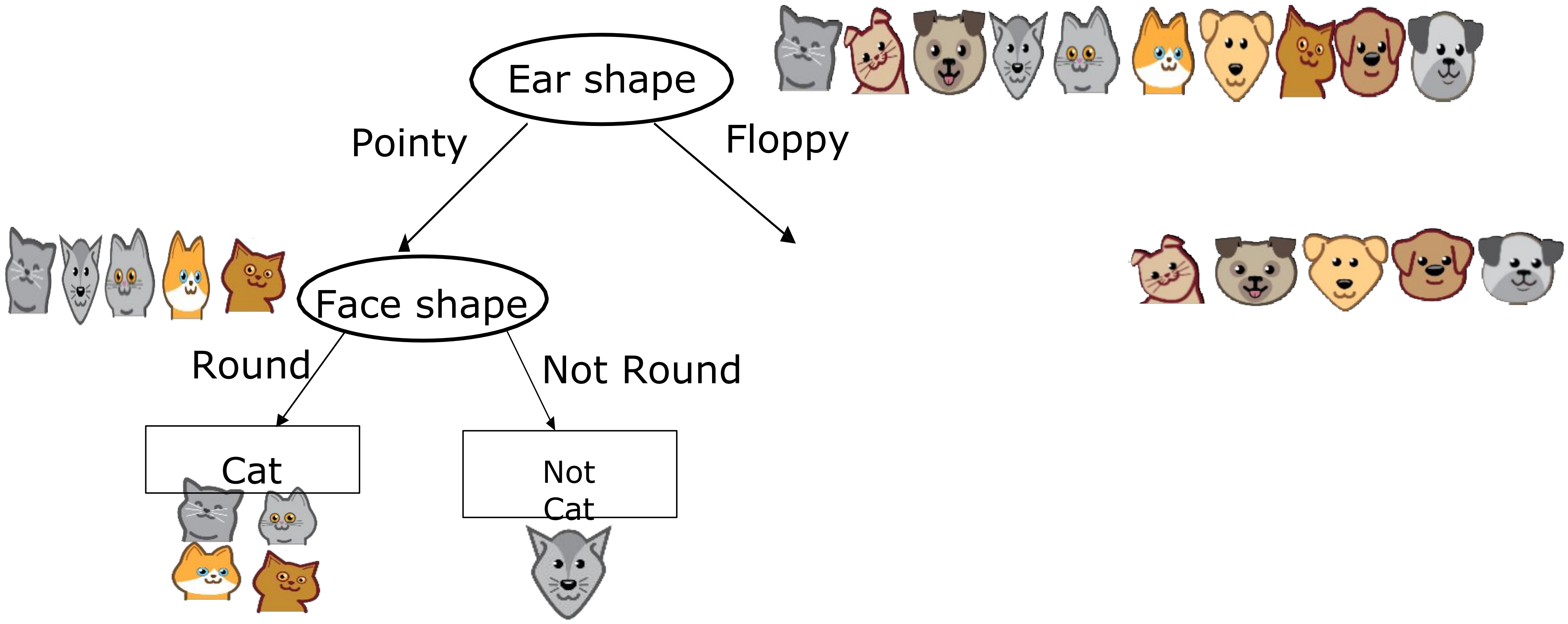


Decision Tree: Learning Process

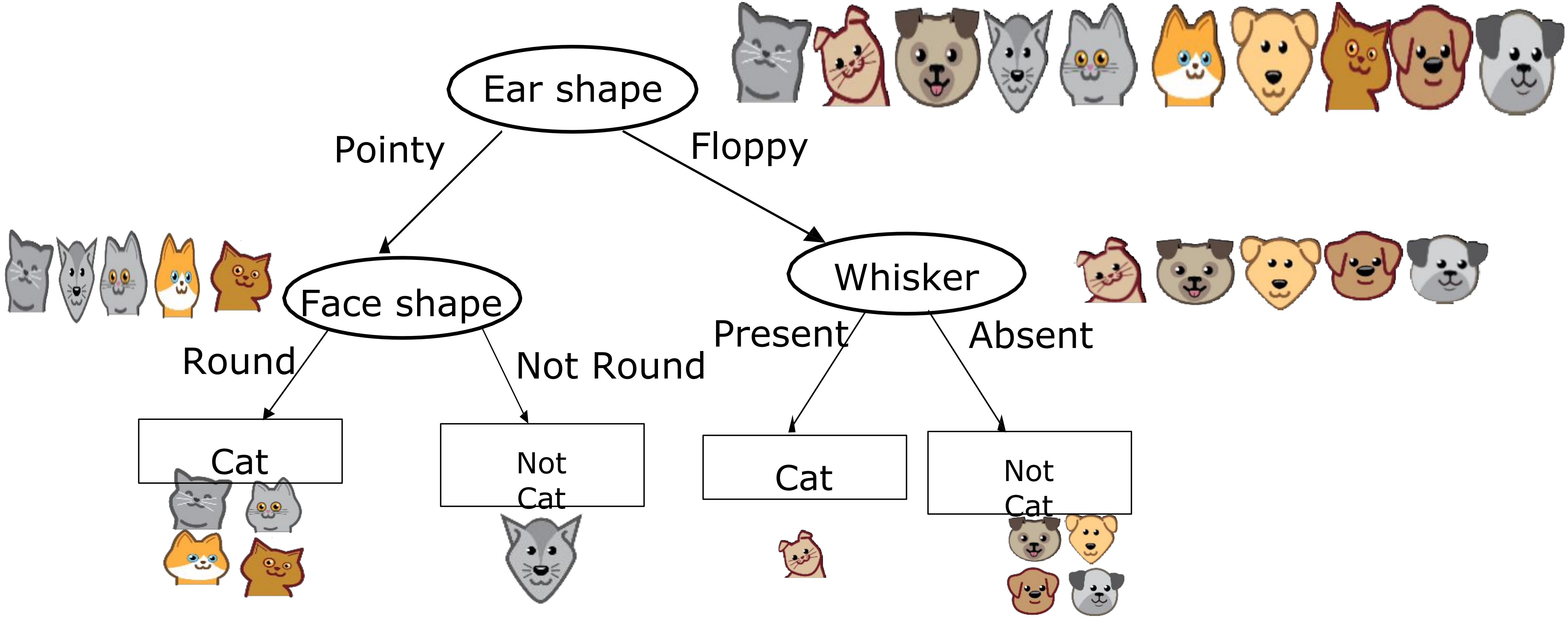
Decision Tree: Learning Process



Decision Tree: Learning Process



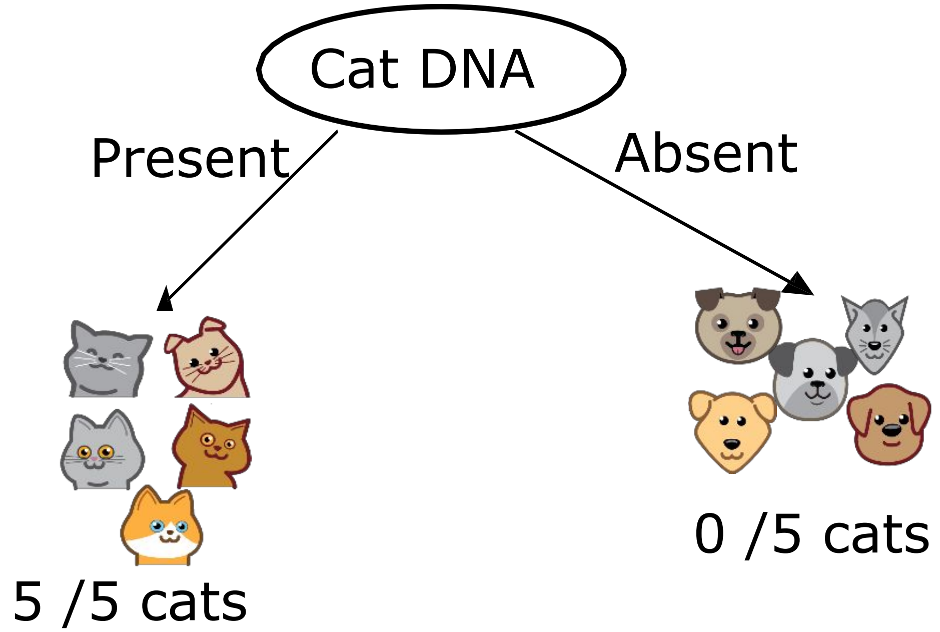
Decision Tree: Learning Process



Decision Tree: Learning Process

Decision 1: How to choose what feature to split on at each node?

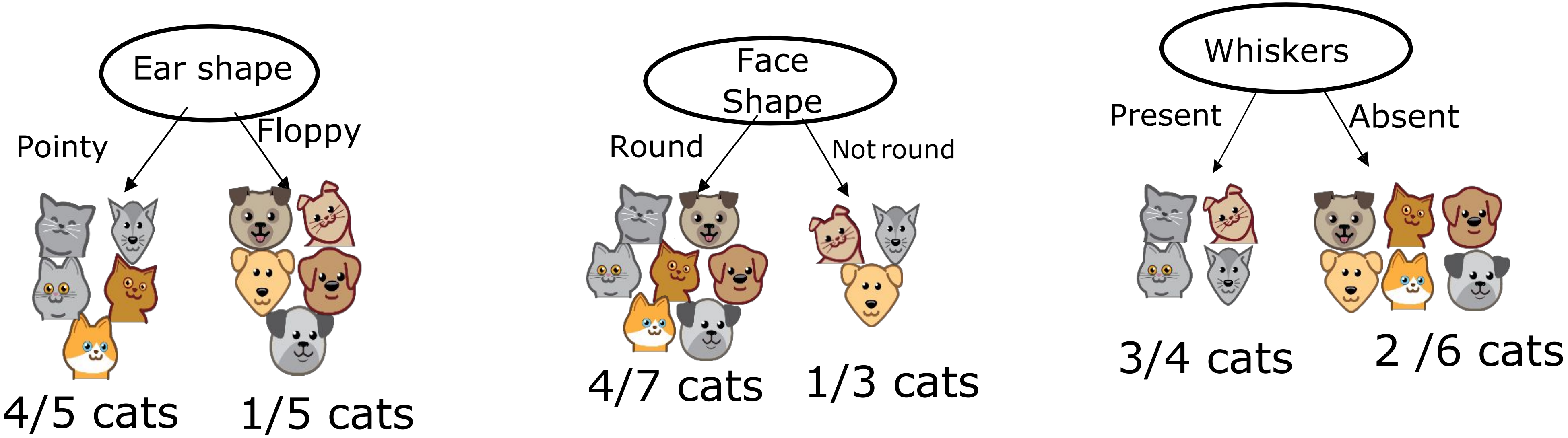
Maximize the purity (Minimize the impurity)



Decision Tree: Learning Process

Decision 1: How to choose what feature to split on at each node?

Maximize the purity (Minimize the impurity)

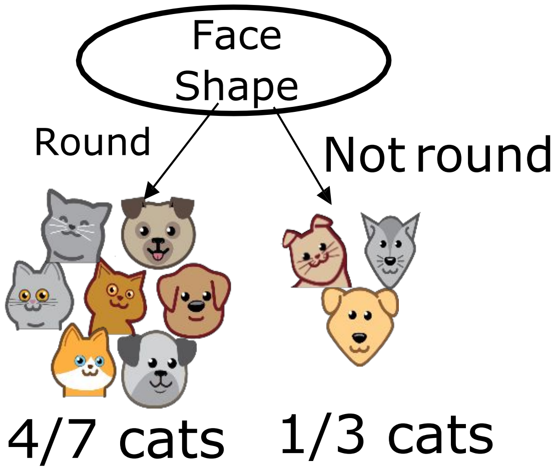


Entropy as measure of purity/impurity

Decision Tree: Learning Process

Decision 2: When do you stop Splitting?

- When a node is 100% one class
- When splitting a node will result in the tree exceeding a maximum depth.
- When improvements in purity score are below a threshold.
- When number of examples in a node is below a threshold



Decision Tree Learning

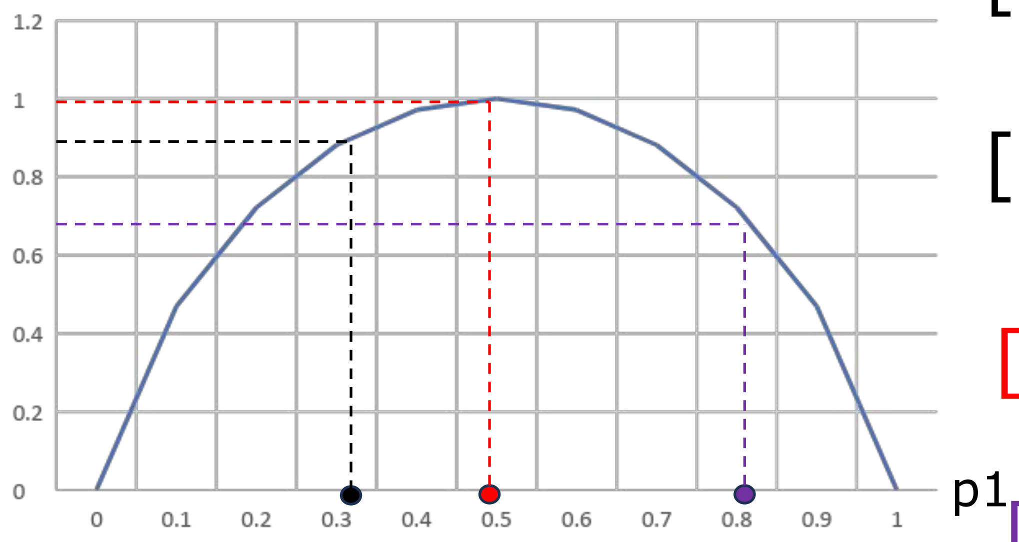
Measuring purity

Entropy as measure of impurity

P1 is fraction of examples that are cat

P0 is fraction of examples that are not cat/dog

☐ 1-p1



[dog,dog,dog,dog,dog,dog] $p_1 = \frac{0}{6} = 0$

[cat,cat,dog,dog,dog,dog] $p_1 = \frac{2}{6} = 0.33$

[cat,cat,cat,dog,dog,dog] $p_1 = \frac{3}{6} = 0.5$

[cat,cat,cat,cat,cat,dog] $p_1 = \frac{5}{6} = 0.83$

[cat,cat,cat,cat,cat,cat] $p_1 = \frac{6}{6} = 1$

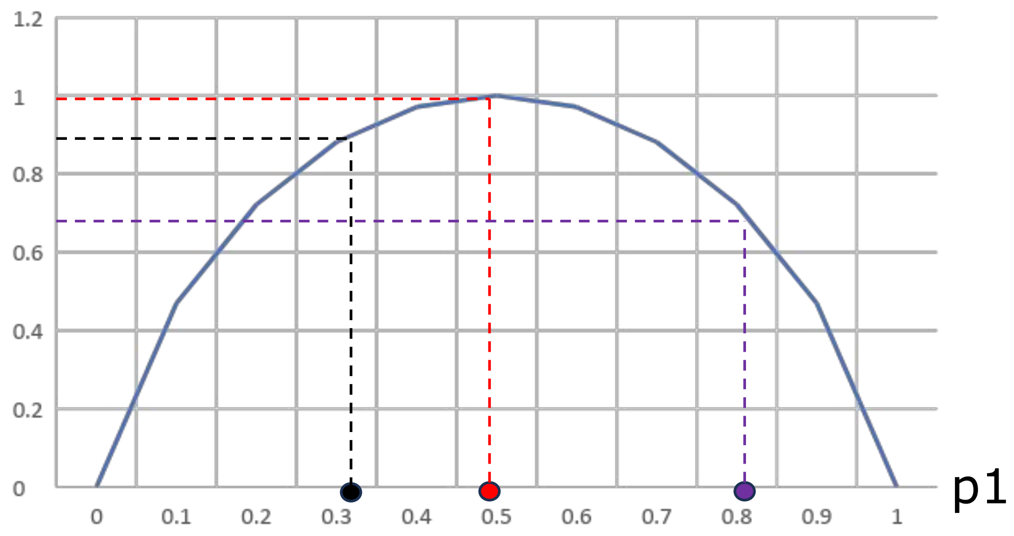
$h(p_1) = 1$ implies high impurity

$h(p_1) = 0$ implies no impurity

Entropy as measure of impurity

P1 is fraction of examples that are cat

P0 is fraction of examples that are not cat/dog
☐ 1-p1



$$h(p) = -p_1 \log_2 p_1 - p_0 \log_2 p_0$$

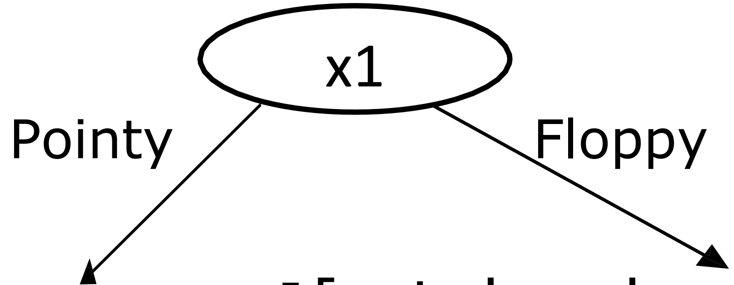
$$h(p) = -\sum_{i=0}^{N} p_i \log_2 p_i$$

$h(p_1) = 1$ implies high impurity

$h(p_1) = 0$ implies no impurity

Choosing a Split: Information Gain

Ear shape (x_1)	Face shape (x_2)	Whiskers (x_3)	Cat
Pointy	Round	Present	1
Floppy	Not round	Present	1
Floppy	Round	Absent	0
Pointy	Not round	Present	0
Pointy	Round	Present	1
Pointy	Round	Absent	1
Floppy	Not round	Absent	0
Pointy	Round	Absent	1
Floppy	Round	Absent	0
Floppy	Round	Absent	0



[cat,dog,cat,cat,cat] [cat,dog,dog,dog,dog]

$$p_1 = \frac{4}{5} = 0.8$$

$$h(0.8) = 0.72$$

$$p_1 = \frac{1}{5} = 0.2$$

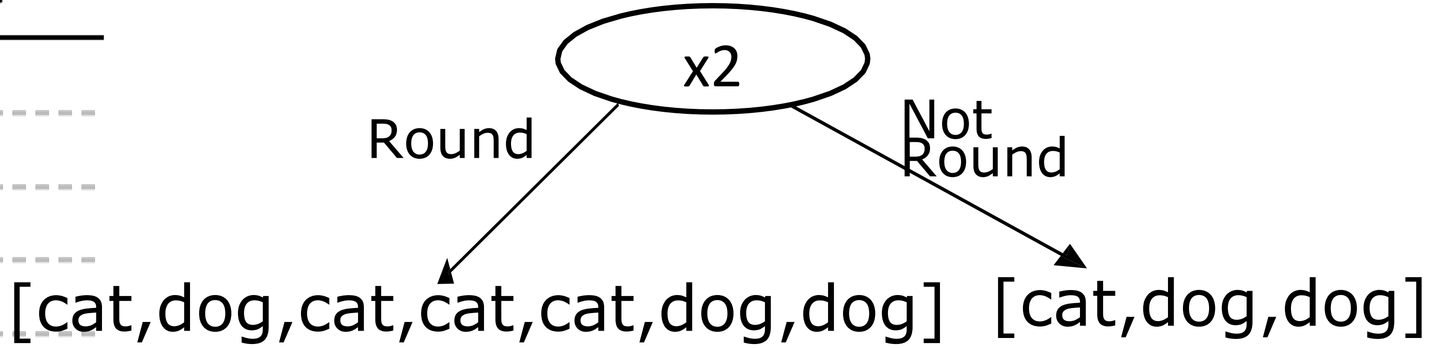
$$h(0.2) = 0.72$$

$$h(0.8) = -0.8 * \log_2 0.8 - 0.2 * \log_2 0.2$$

$$h(0.2) = -0.2 * \log_2 0.2 - 0.8 * \log_2 0.8$$

Choosing a Split: Information Gain

Ear shape (x_1)	Face shape (x_2)	Whiskers (x_3)	Cat
Pointy	Round	Present	1
Floppy	Not round	Present	1
Floppy	Round	Absent	0
Pointy	Not round	Present	0
Pointy	Round	Present	1
Pointy	Round	Absent	1
Floppy	Not round	Absent	0
Pointy	Round	Absent	1
Floppy	Round	Absent	0
Floppy	Round	Absent	0



$$p_1 = \frac{4}{7} = 0.57 \qquad p_1 = \frac{1}{3} = 0.33$$

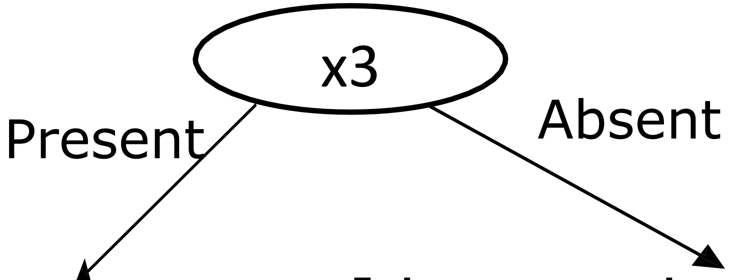
$$h(0.57) = 0.99 \qquad h(0.33) = 0.91$$

$$h(0.57) = -0.57 * \log_2 0.57 - 0.43 * \log_2 0.43$$

$$h(0.33) = -0.33 * \log_2 0.33 - 0.67 * \log_2 0.67$$

Choosing a Split: Information Gain

Ear shape (x_1)	Face shape (x_2)	Whiskers (x_3)	Cat
Pointy	Round	Present	1
Floppy	Not round	Present	1
Floppy	Round	Absent	0
Pointy	Not round	Present	0
Pointy	Round	Present	1
Pointy	Round	Absent	1
Floppy	Not round	Absent	0
Pointy	Round	Absent	1
Floppy	Round	Absent	0
Floppy	Round	Absent	0



[cat,cat,dog,cat]

$$p_1 = \frac{3}{4} = 0.75$$

$$h(0.75) = 0.81$$

[dog,cat,dog,cat, dog,dog]

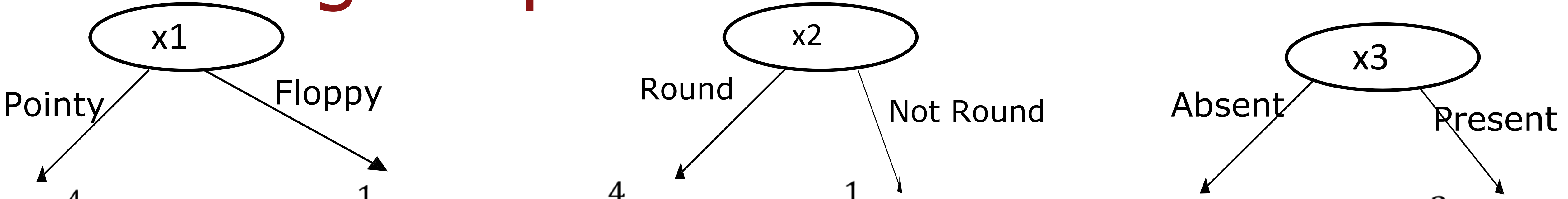
$$p_1 = \frac{2}{6} = 0.33$$

$$h(0.33) = 0.91$$

$$h(0.75) = -0.75 * \log_2 0.75 - 0.25 * \log_2 0.25$$

$$h(0.33) = -0.33 * \log_2 0.33 - 0.67 * \log_2 0.67$$

Choosing a Split: Information Gain



$p_1 = \frac{4}{5} = 0.8$ $h(0.8) = 0.72$

$p_1 = \frac{1}{5} = 0.2$ $h(0.2) = 0.72$

$p_1 = \frac{4}{7} = 0.57$ $h(0.57) = 0.99$

$p_1 = \frac{1}{3} = 0.33$ $h(0.33) = 0.91$

$p_1 = \frac{3}{4} = 0.75$ $h(0.75) = 0.81$

$p_1 = \frac{2}{6} = 0.33$ $h(0.33) = 0.91$

$$h(0.5) - \left(\frac{5}{10}h(0.80) + \frac{5}{10}h(0.20)\right) = 0.28$$

$$h(0.5) - \left(\frac{7}{10}h(0.57) + \frac{3}{10}h(0.33)\right) = 0.03$$

$$h(0.5) - \left(\frac{4}{10}h(0.75) + \frac{6}{10}h(0.33)\right) = 0.12$$

Information Gain $= h(p_1^{prev}) - \left(w^{left}h(p_1^{left}) + w^{left}h\left(p_1^{right}\right)\right)$

In decision trees, **entropy** is a measure of impurity used to evaluate the homogeneity of a dataset. It helps determine the best split for building an informative decision tree model.

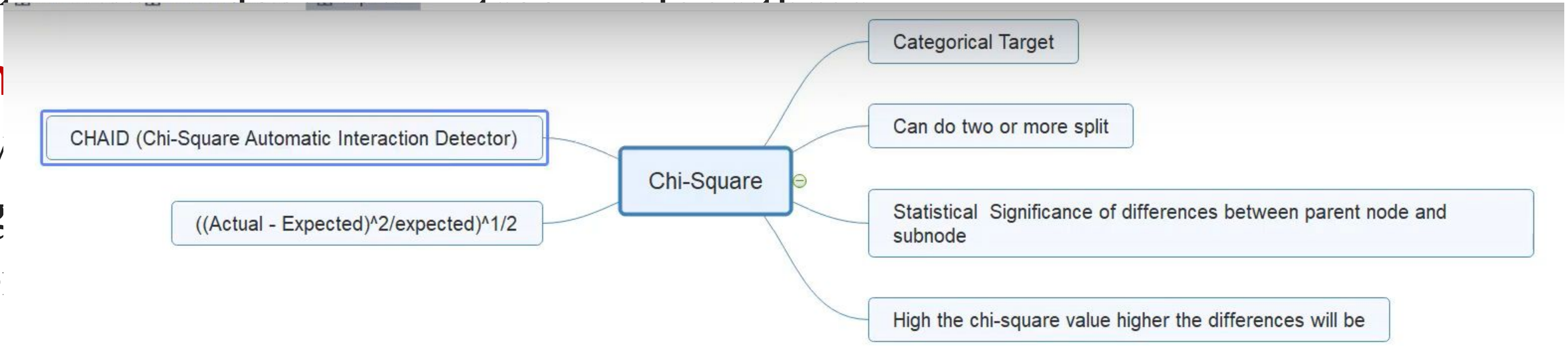
Information Gain and entropy are related concept

Inform

entropy

helping

partitio

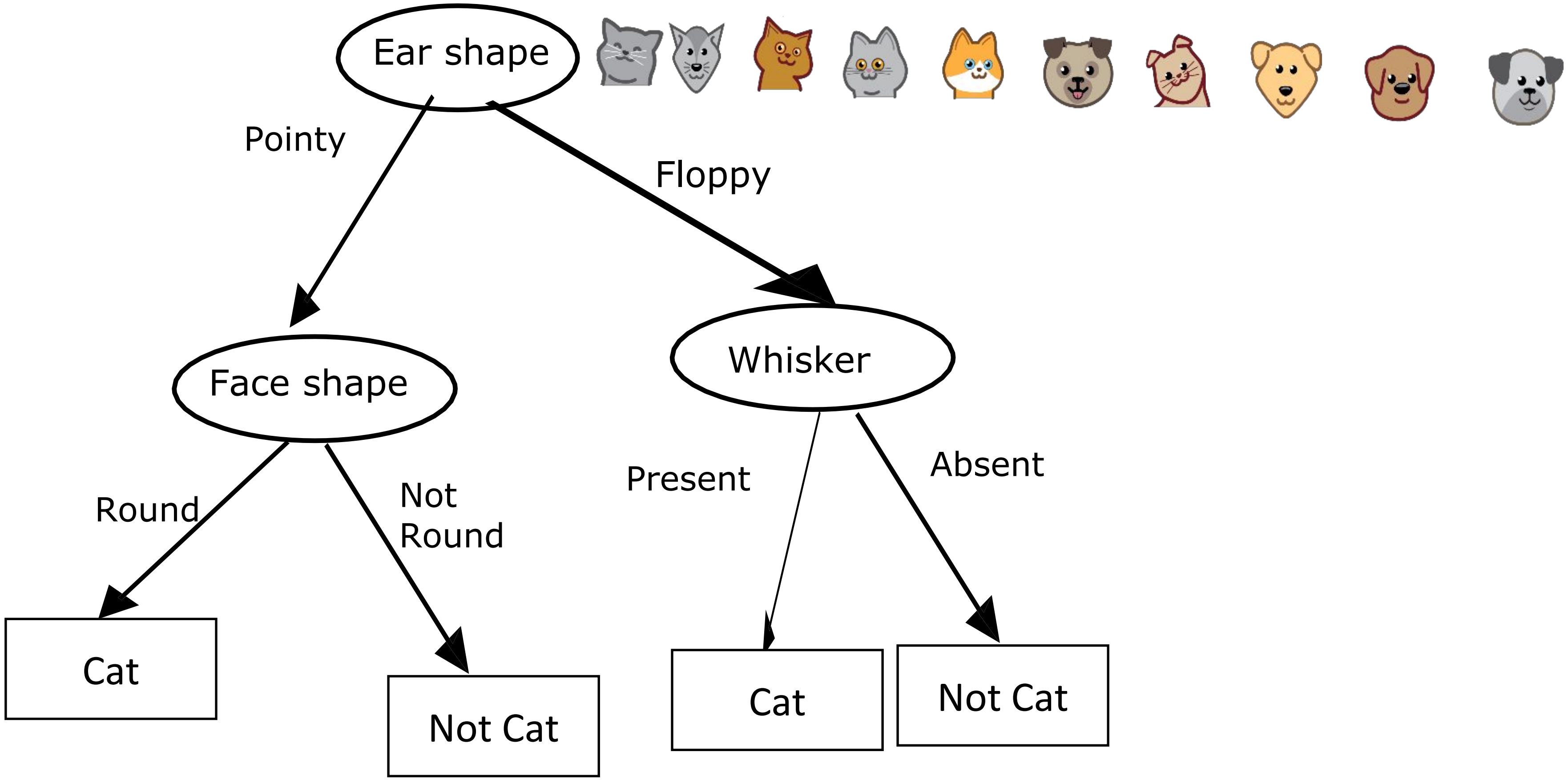


Famous Decision tree algorithm and split functions:

- ID3 Entropy
- CART Gini Index $GI(t) = 1 - \sum_{k=0}^{n} p(k)^2$
- CHAID (Chi-Squared Automatic Interaction Detection)

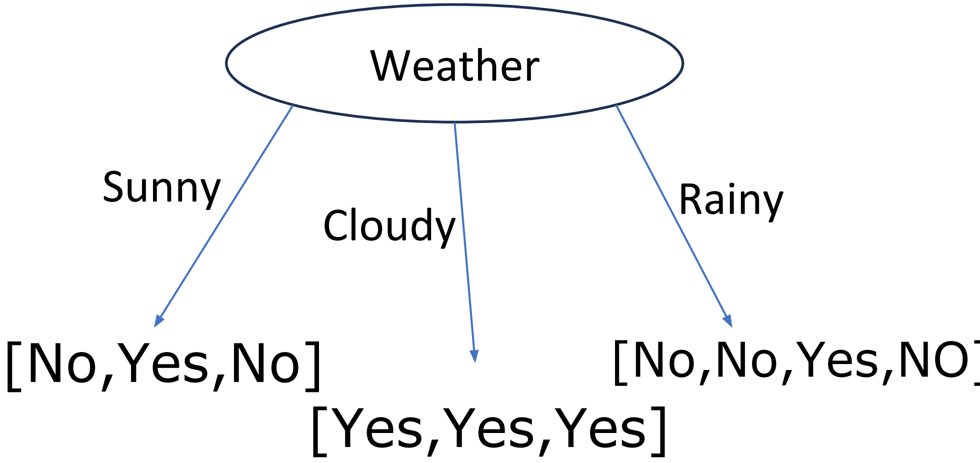
Decision Tree Algorithm

- Start with all examples at the root node
- Calculate information gain for all possible features, and pick the one with the highest information gain
- Split dataset according to selected feature, and create left and right branches of the tree
- Keep repeating splitting process until stopping criteria is met:
  - When a node is 100% one class
  - When splitting a node will result in the tree exceeding a maximum depth
  - Information gain from additional splits is less than threshold
  - When number of examples in a node is below a threshold


Ear shape
Pointy
Floppy
Face shape
Whisker
Round
Not Round
Present
Absent
Cat
Not Cat
Cat
Not Cat

Decision Tree: Tennis Play

Day	Weather	Temperature	Humidity	Wind	Play?
1	Sunny	Hot	High	Weak	No
2	Cloudy	Hot	High	Weak	Yes
3	Sunny	Mild	Normal	Strong	Yes
4	Cloudy	Mild	High	Strong	Yes
5	Rainy	Mild	High	Strong	No
6	Rainy	Cool	Normal	Strong	No
7	Rainy	Mild	High	Weak	Yes
8	Sunny	Hot	High	Strong	No
9	Cloudy	Hot	Normal	Weak	Yes
10	Rainy	Mild	High	Strong	No



One hot encoding

One hot encoding

Weather	Temperature	Humidity	Wind	Play Tennis
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

One hot encoding

With one-hot, we convert each categorical value into a new categorical column and assign a binary value of 1 or 0 to those columns. Each integer value is represented as a binary vector

Weather	W_S	W_O	W_R	Temperature	Humidity	Wind	Play Tennis
Sunny	1	0	0	Hot	High	Weak	No
Sunny	1	0	0	Hot	High	Strong	No
Overcast	0	1	0	Hot	High	Weak	Yes
Rain	0	0	1	Mild	High	Weak	Yes
Rain	0	0	1	Cool	Normal	Weak	Yes
Rain	0	0	1	Cool	Normal	Strong	No
Overcast	0	1	0	Cool	Normal	Strong	Yes
Sunny	1	0	0	Mild	High	Weak	No
Sunny	1	0	0	Cool	Normal	Weak	Yes
Rain	0	0	1	Mild	Normal	Weak	Yes
Sunny	1	0	0	Mild	Normal	Strong	Yes
Overcast	0	1	0	Mild	High	Strong	Yes
Overcast	0	1	0	Hot	Normal	Weak	Yes
Rain	0	0	1	Mild	High	Strong	No

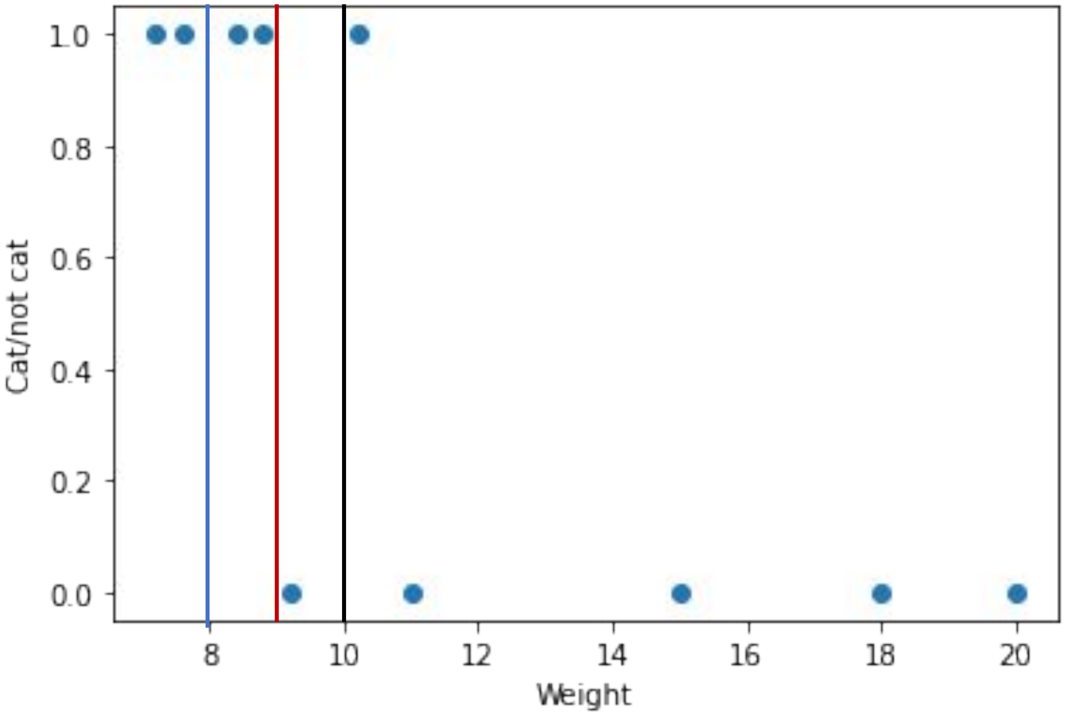
Decision Tree Learning

Continuous valued features

Continuous features

Ear shape	Face shape	Whiskers	Weight (lbs.)	Cat
Pointy	Round	Present	7.2	1
Floppy	Not round	Present	8.8	1
Floppy	Round	Absent	15	0
Pointy	Not round	Present	9.2	0
Pointy	Round	Present	8.4	1
Pointy	Round	Absent	7.6	1
Floppy	Not round	Absent	11	0
Pointy	Round	Absent	10.2	1
Floppy	Round	Absent	18	0
Floppy	Round	Absent	20	0

Continuous features give high information gain



Weight <=10

Weight <=9

$$IG = H(0.5) - \left(\frac{2}{10}(H\left(\frac{2}{2}\right) + \frac{8}{10}H\left(\frac{3}{8}\right)\right) = 0.24$$

$$IG = H(0.5) - \left(\frac{4}{10}(H\left(\frac{4}{4}\right) + \frac{6}{10}H\left(\frac{1}{6}\right)\right) = 0.61$$

$$IG = H(0.5) - \left(\frac{5}{10}(H\left(\frac{4}{5}\right) + \frac{5}{10}H\left(\frac{1}{5}\right)\right) = 0.48$$

Regression Tree

Decision Tree:

- Split Measures: Entropy and IG
- Classification: District Output

Regression Tree:

- Spit Measures: Variance, MSE
- Regression: Continuous Output

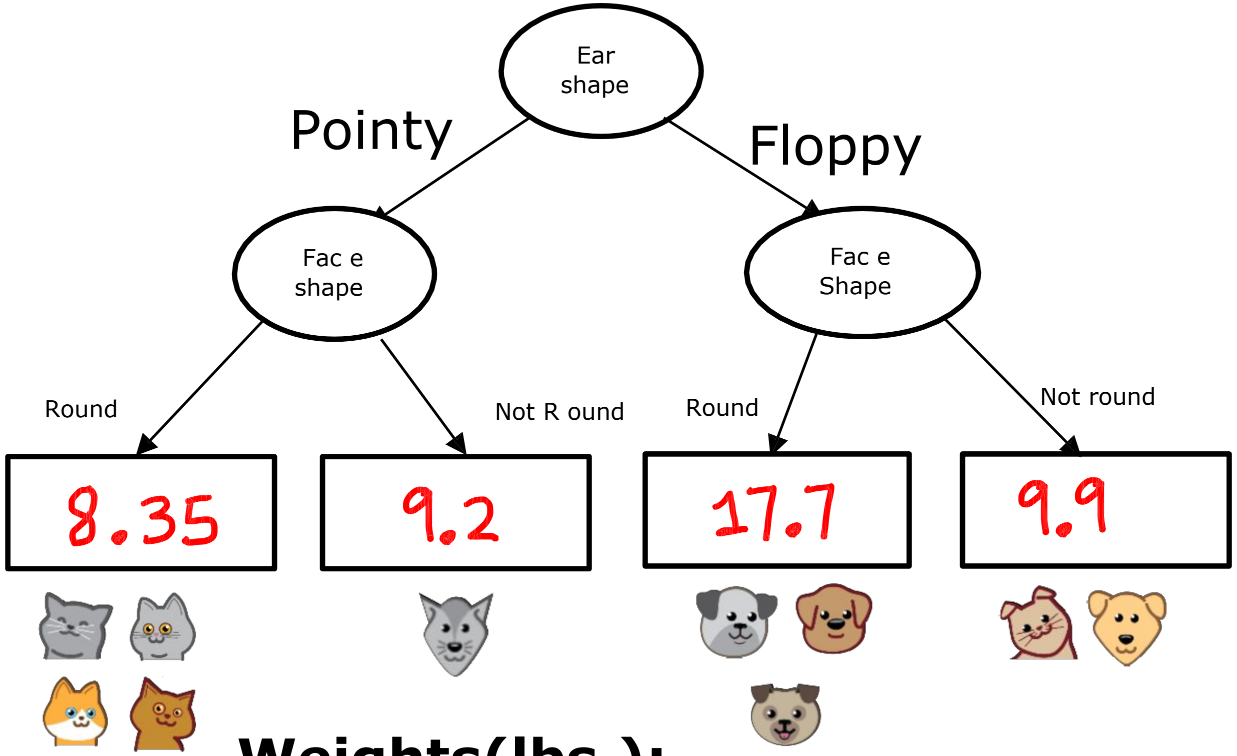
Regression with Decision Trees: Predicting a number

Ear shape	Face shape	Whiskers	Weight (lbs.)
Pointy	Round	Present	7.2
Floppy	Not round	Present	8.8
Floppy	Round	Absent	15
Pointy	Not round	Present	9.2
Pointy	Round	Present	8.4
Pointy	Round	Absent	7.6
Floppy	Not round	Absent	11
Pointy	Round	Absent	10.2
Floppy	Round	Absent	18
Floppy	Round	Absent	20

X

y

Regression with Decision Trees



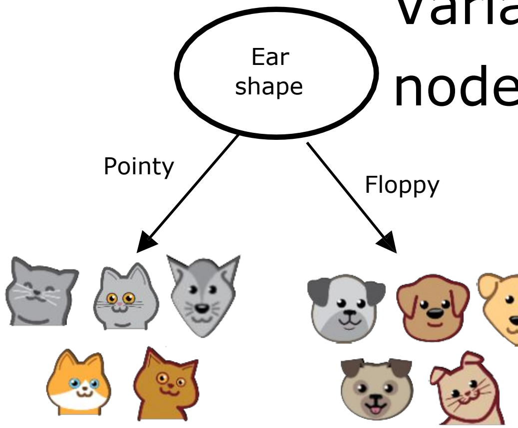
Weights(lbs.):
7.2, 8.4, 7.6, 10.2

Weights(lbs.):
9.2

Weights (lbs.):
15, 18, 20

Weights(lbs.):
8.8,11

Choosing a split



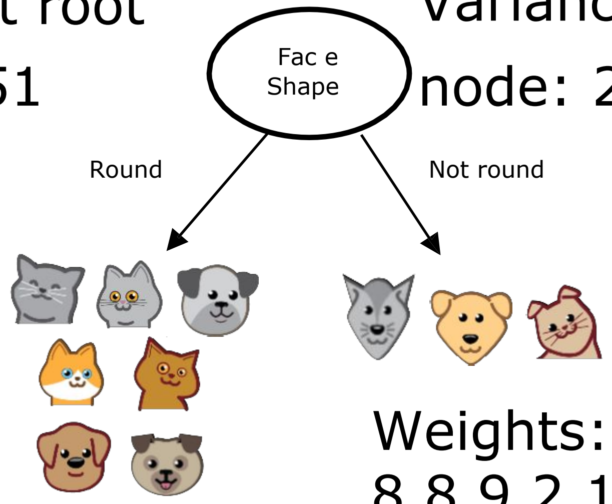
Variance at root node: 20.51

Weights: 7.2, 9.2, 8.4,7.6, 10.2

Variance: 1.47

Weights: 8.8, 15, 11, 18, 20

Variance: 21.87



Variance at root node: 20.51

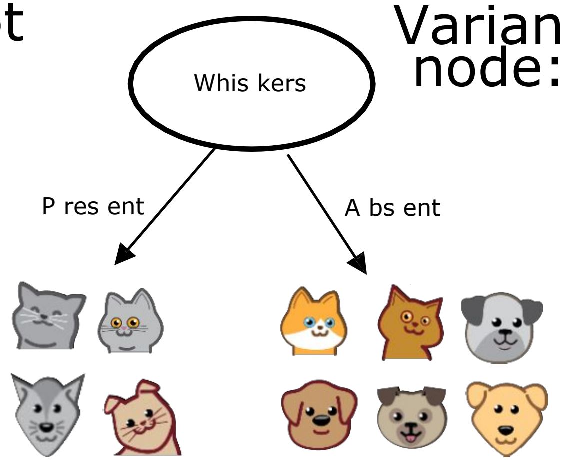
Weights: 7.2, 15,8.4,7.6,10 .2, 18, 20

Variance : 27.80

Weights: 8.8,9.2,11

Variance: 1.37

$IG = 0.64$



Variance at root node: 20.51

Weights: 7.2, 8.8, 9.2, 8.4

Variance: 0.75

Weights: 15, 7.6, 11, 10.2, 18, 20

Variance: 23.32

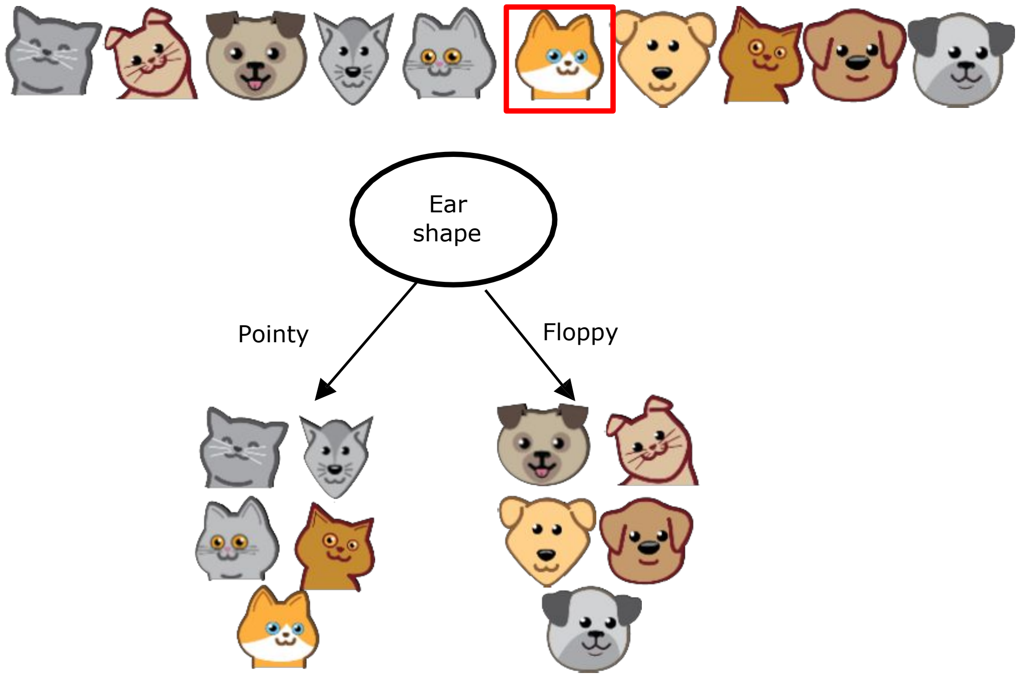
$IG = 6.22$

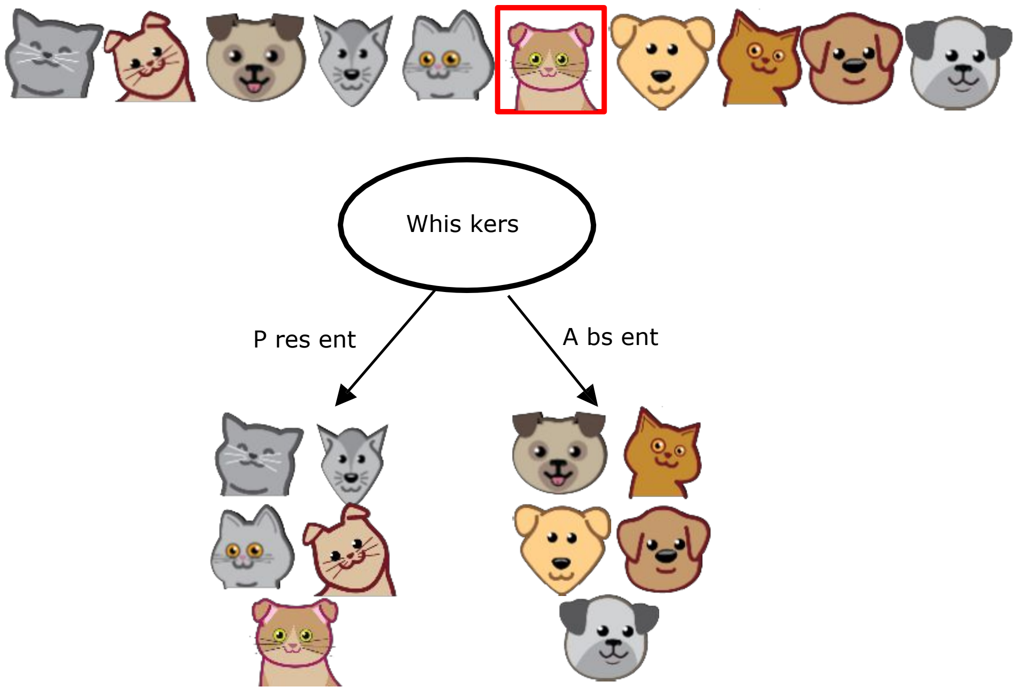
$$IG = 20.51 - \left(\frac{5}{10} * 1.47 + \frac{5}{10} * 21.87\right) = \mathbf{8.84}$$

Tree ensembles

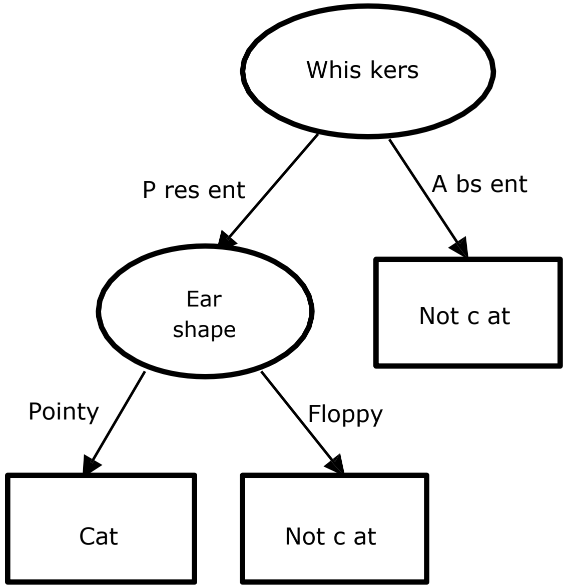
Using multiple decision trees

Trees are highly sensitive to small changes of the data

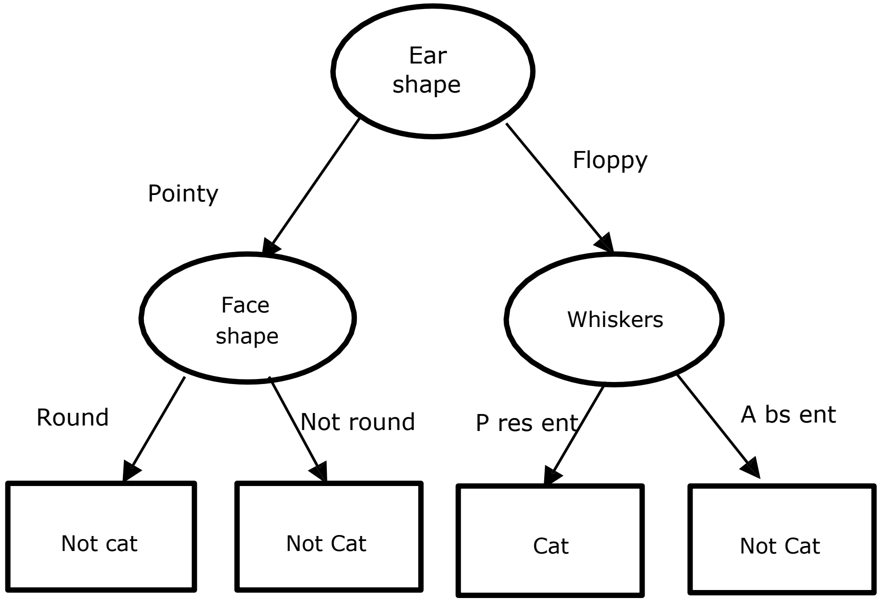




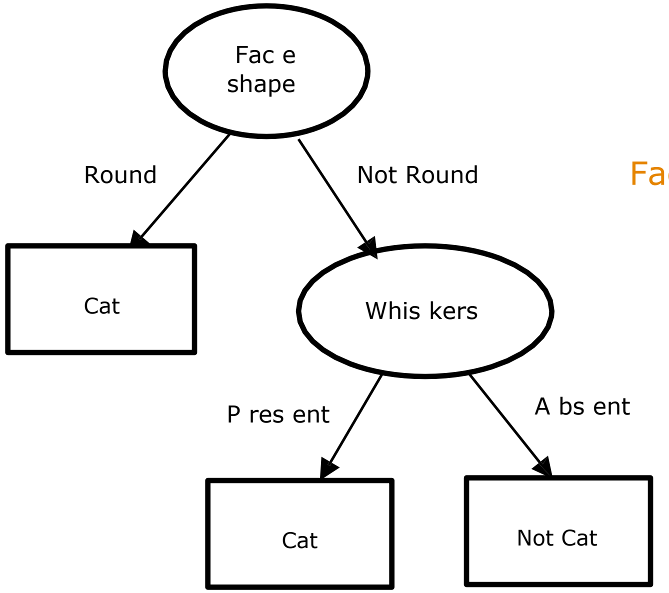
Tree ensemble



Prediction: Cat



Prediction: Not cat



Prediction: Cat

Ne w te s t e xa mple

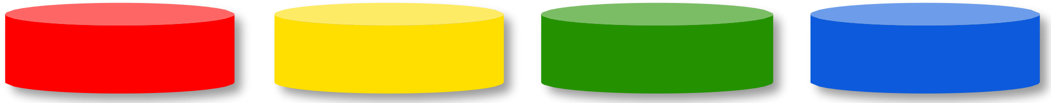
Ear shape: Pointy
Face shape: Not Round
Whiskers: Present

Final prediction: Cat (Majority Vote)

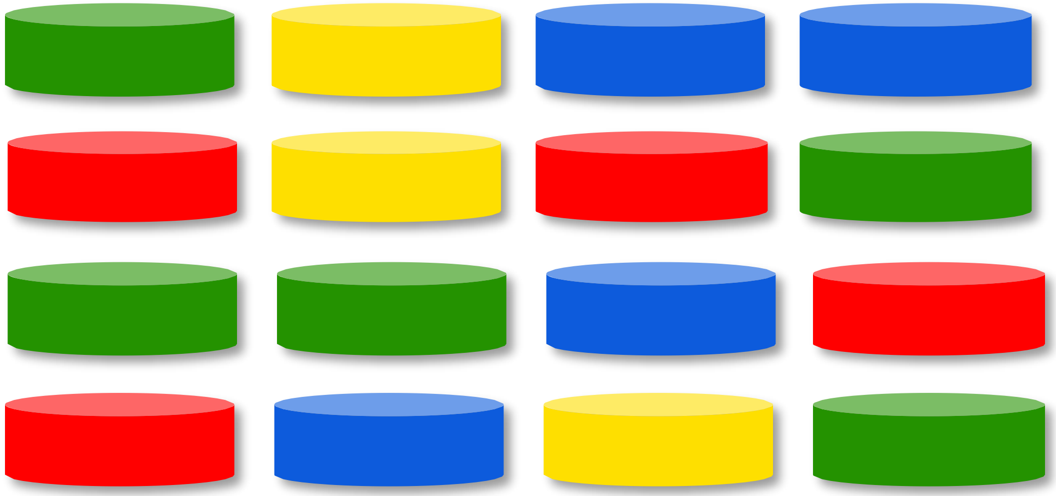
Tree ensembles
Methods

Sampling with replacement
Bootstraping

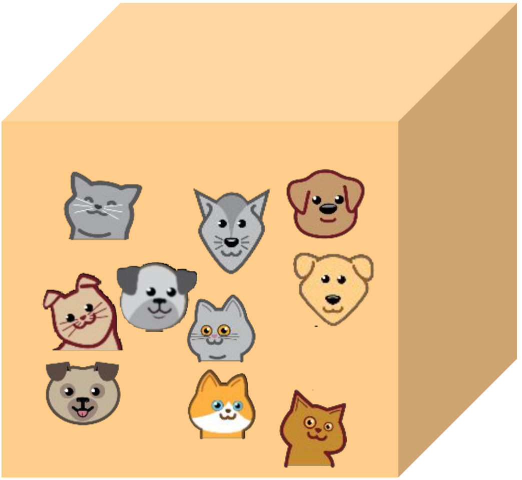
Sampling with replacement

Tokens 

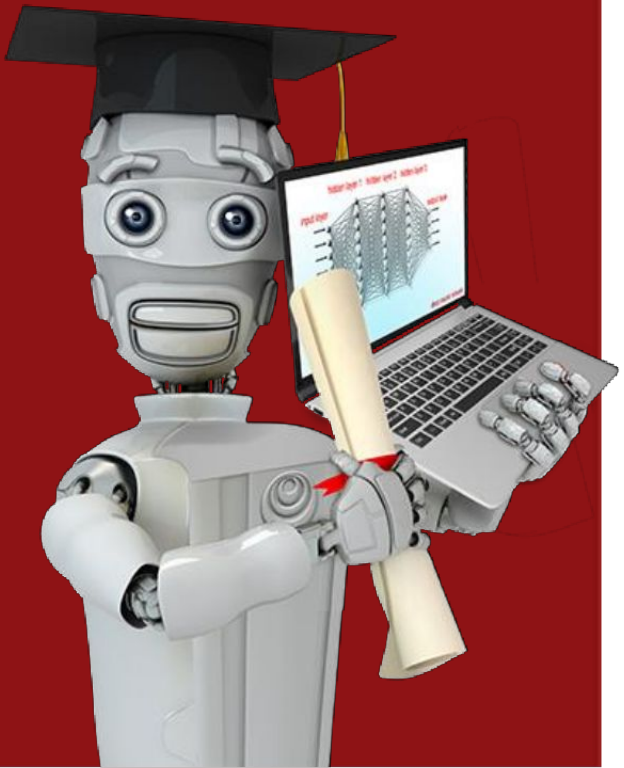
Sampling with replacement:



Sampling with replacement



Ear shape	Face shape	Whiskers	Cat
Pointy	Round	Present	1
Floppy	Not round	Absent	0
Pointy	Round	Absent	1
Pointy	Not round	Present	0
Floppy	Not round	Absent	0
Pointy	Round	Absent	1
Pointy	Round	Present	1
Floppy	Not round	Present	1
Floppy	Round	Absent	0
Pointy	Round	Absent	1



Tree ensembles

Random forest algorithm

Generating a tree sample

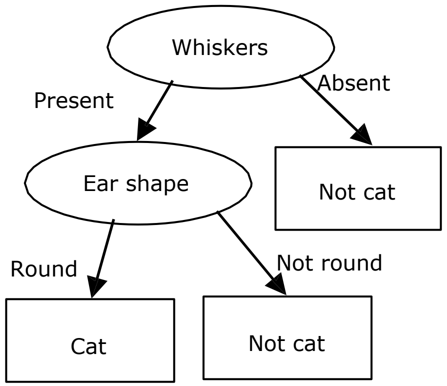
Given training set of size m

For $b = 1$ to B:

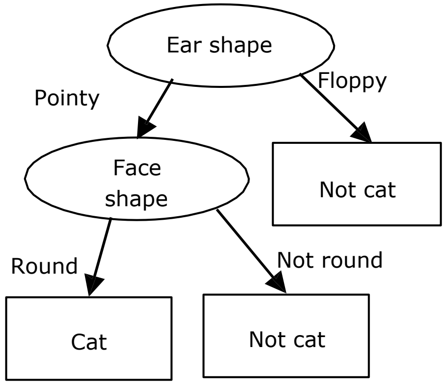
Use sampling with replacement to create a new training set of size m

Train a decision tree on the new dataset

Ear shape	Face shape	Whiskers	Cat
Pointy	Round	Present	Yes
Floppy	Round	Absent	No
Floppy	Round	Absent	No
Pointy	Round	Present	Yes
Pointy	Not Round	Present	Yes
Floppy	Round	Absent	No
Floppy	Round Not	Present	Yes
Pointy	Round Not	Absent	No
Pointy	Round Not	Absent	No
Pointy	Round	Present	Yes



Ear shape	Face shape	Whiskers	Cat
Pointy	Round	Present	Yes
Pointy	Round Not	Absent	Yes
Floppy	Round Not	Absent	No
Floppy	Round	Absent	No
Pointy	Round	Absent	Yes
Floppy	Round	Absent	No
Floppy	Round	Absent	No
Floppy	Round	Absent	No
Pointy	Not Round	Absent	No
Pointy	Round	Present	Yes



...

Bagged decision tree

Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

Decision Tree	Random Forest
A decision tree is a tree-like model of decisions along with possible outcomes in a diagram.	A classification algorithm consisting of many decision trees combined to get a more accurate result as compared to a single tree.
There is always a scope for overfitting, caused due to the presence of variance.	Random forest algorithm avoids and prevents overfitting by using multiple trees.
The results may not accurate.	This gives accurate and precise results.
Decision trees require low computation, thus reducing time to implement and carrying low accuracy.	This consumes more computation. The process of generation and analyzing is time-consuming.
It is easy to visualize. The only task is to fit the decision tree model.	This has complex visualization as it determines the pattern behind the data.