Machine Learning

by - Andrew Ng

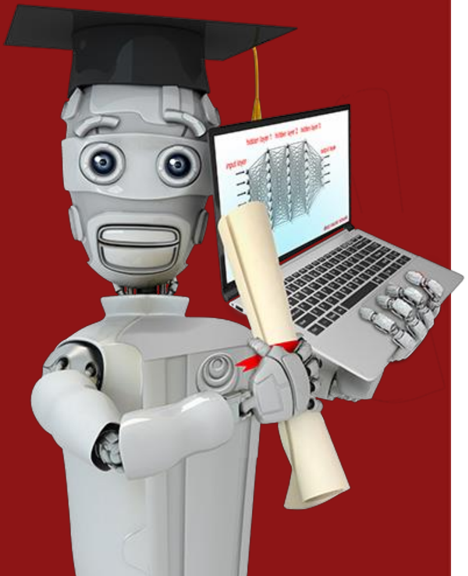
Tue 19 Jul

week 4

Copyright Notice

DeepLearning.AI

Stanford ONLINE



Decision Trees

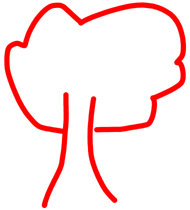
Decision Tree Model

Cat classification example

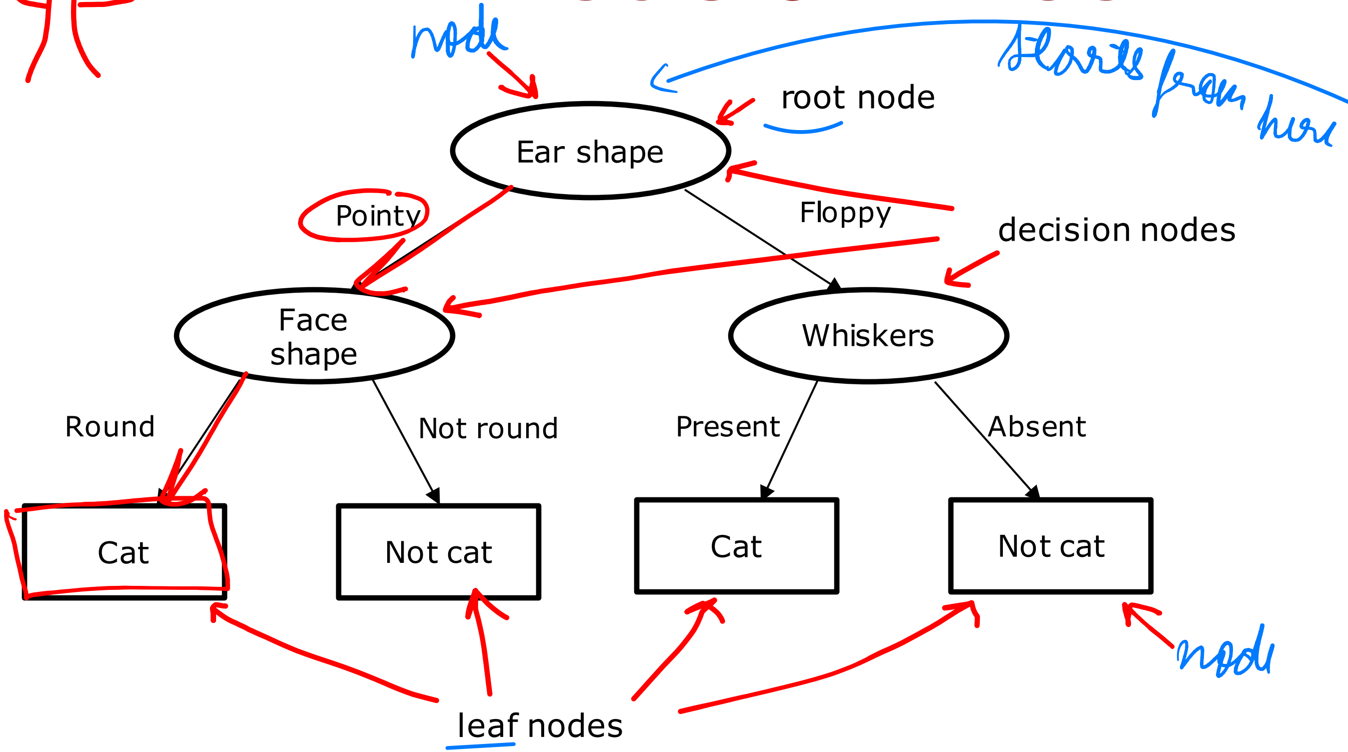
Ear shape (x_1)	Face shape (x_2)	Whiskers (x_3)	Cat
Pointy	Round	Present	1
Floppy	Not round	Present	1
Floppy	Round	Absent	0
Pointy	Not round	Present	0
Pointy	Round	Present	1
Pointy	Round	Absent	1
Floppy	Not round	Absent	0
Pointy	Round	Absent	1
Floppy	Round	Absent	0
Floppy	Round	Absent	0

X　　y

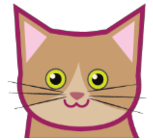
Categorical (discrete values)



Decision Tree

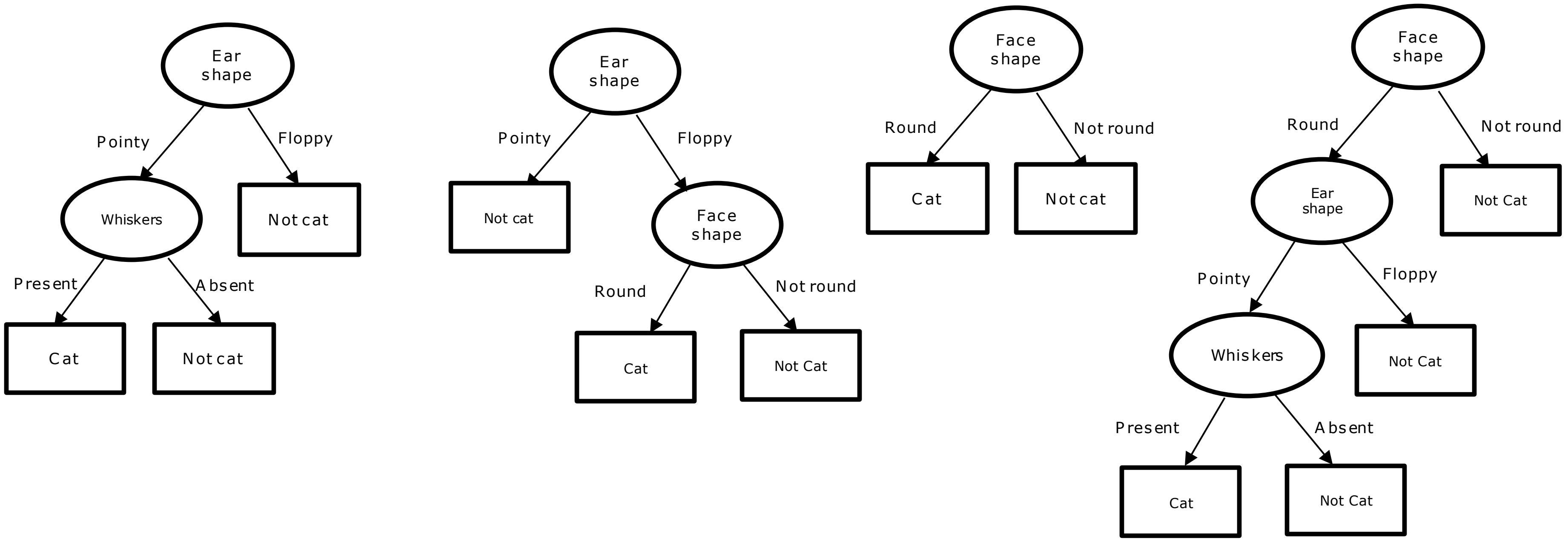


New test example



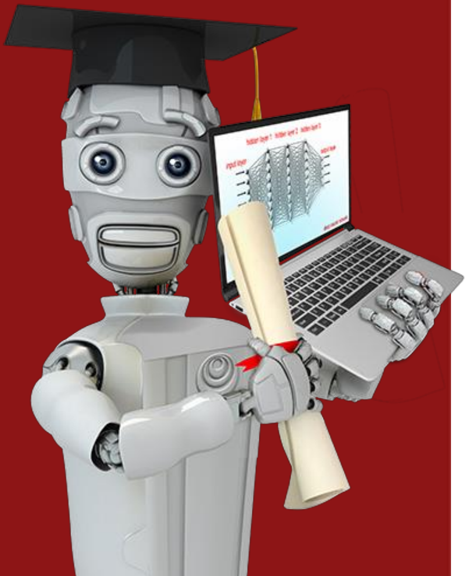
Ear shape: Pointy
Face shape: Round
Whiskers: Present

Decision Tree



DeepLearning.AI

Stanford ONLINE

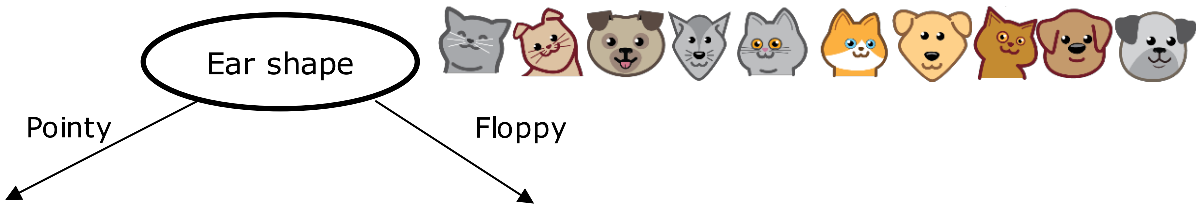


Decision Trees

Learning Process

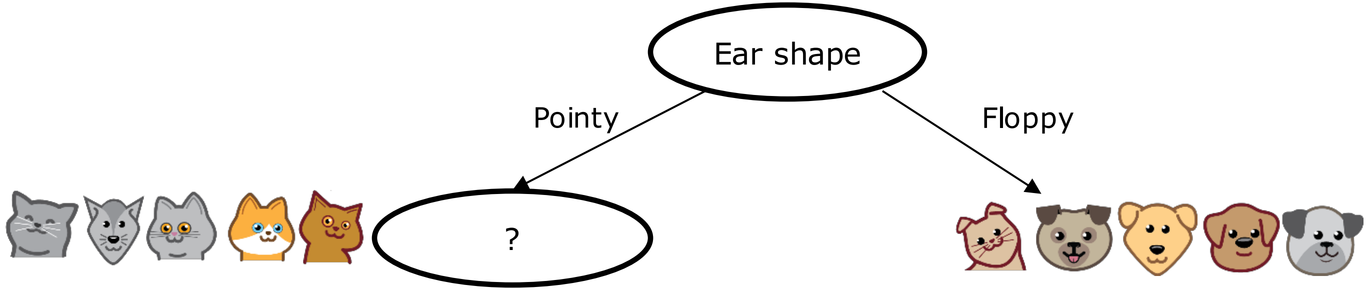
Decision Tree Learning

Decision Tree Learning

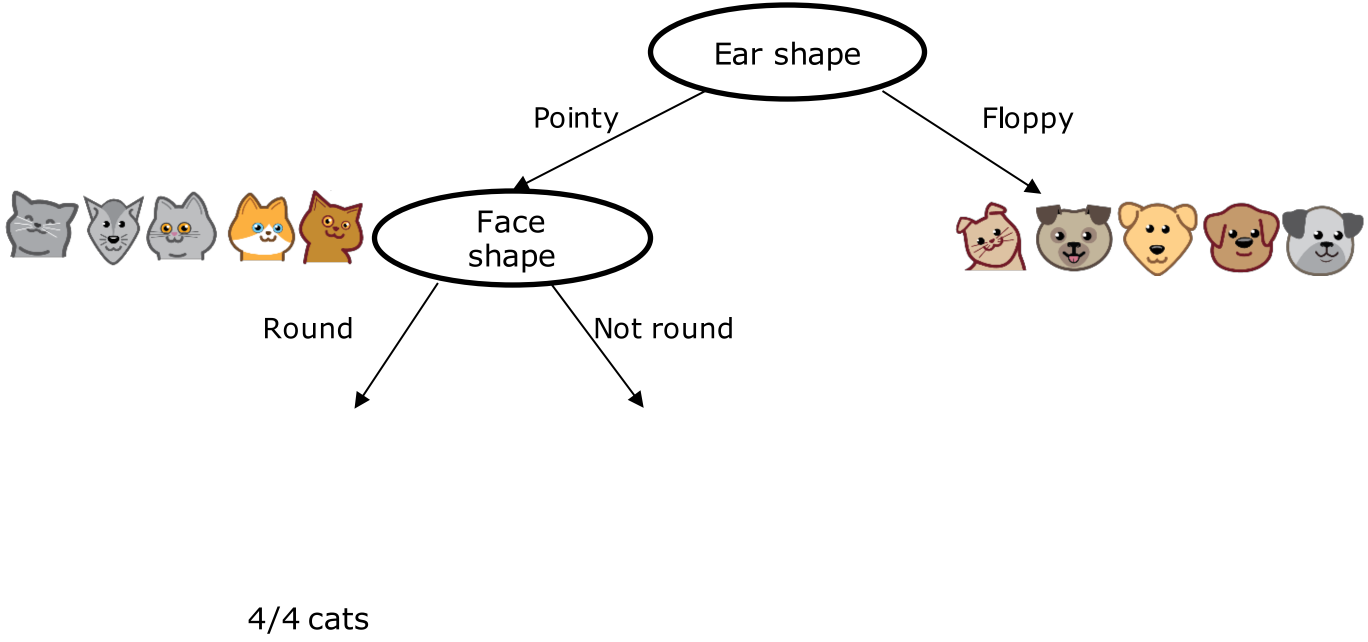


Decision Tree Learning

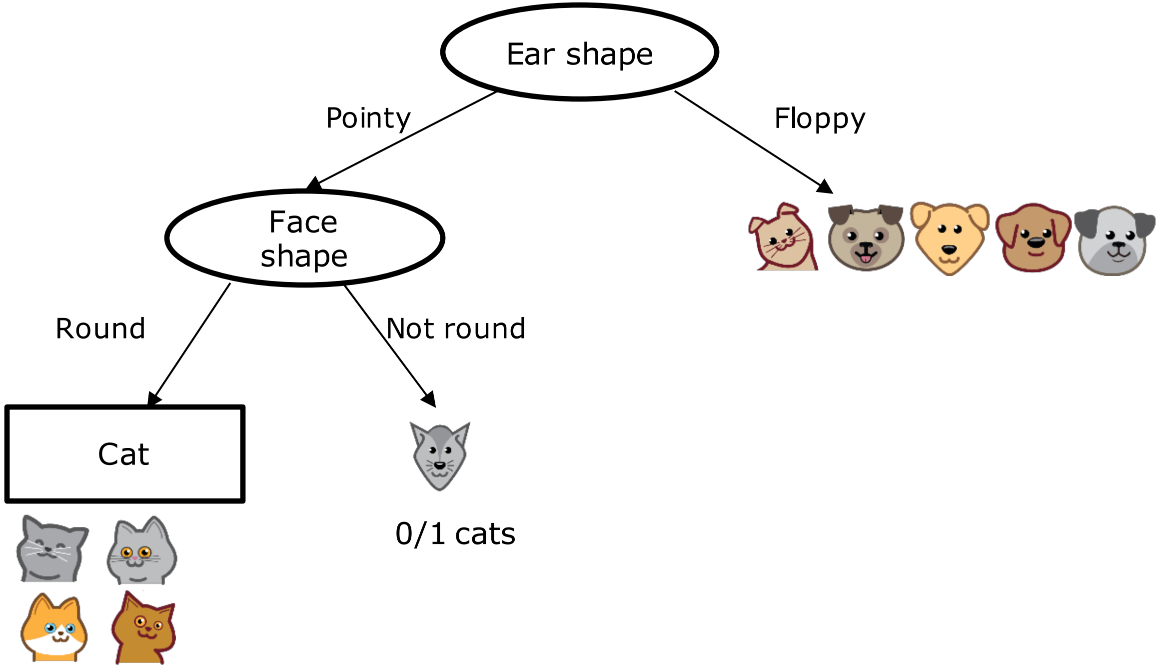
Decision Tree Learning



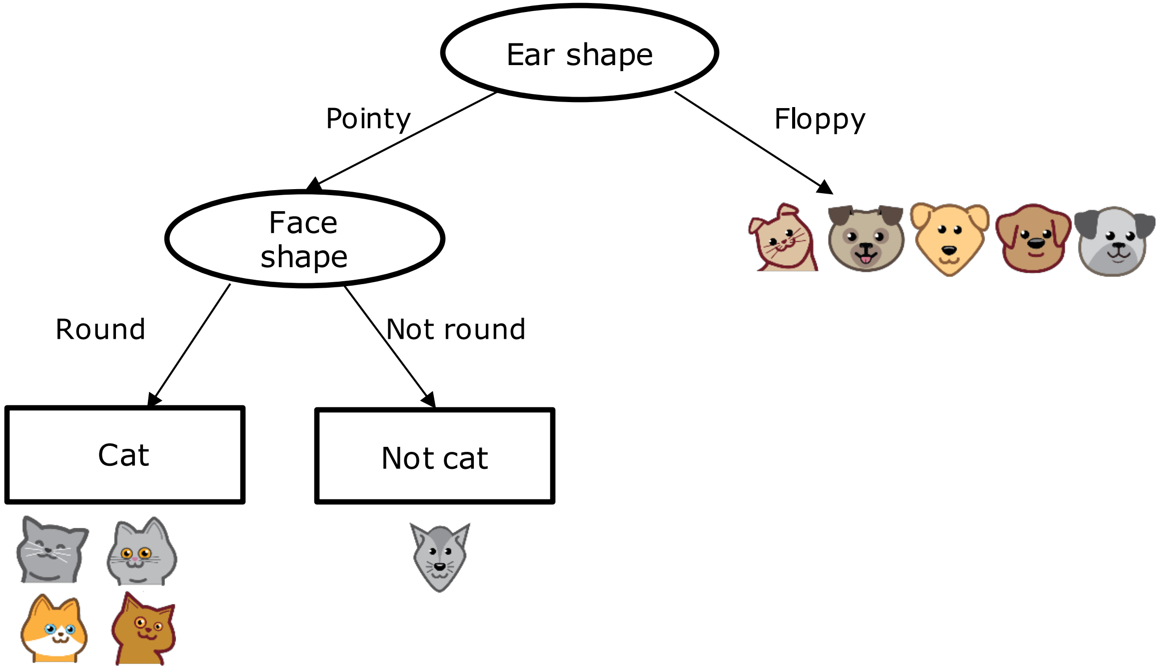
Decision Tree Learning



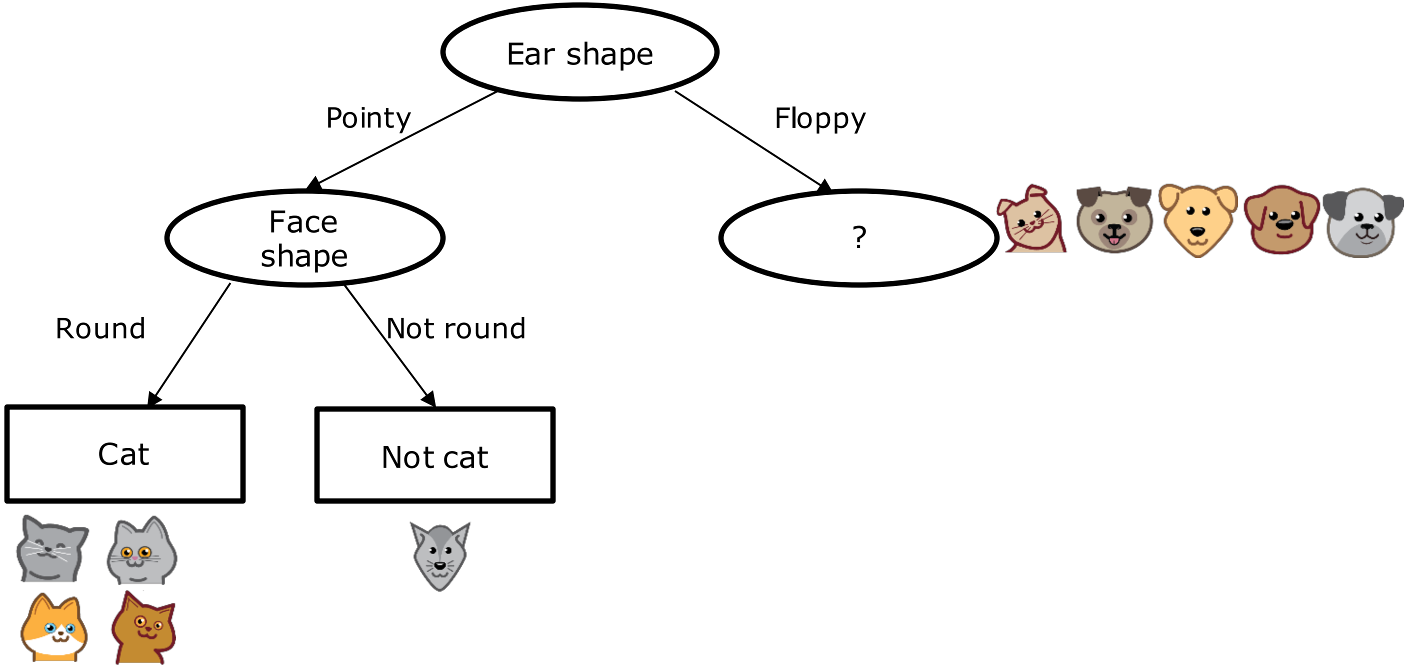
Decision Tree Learning



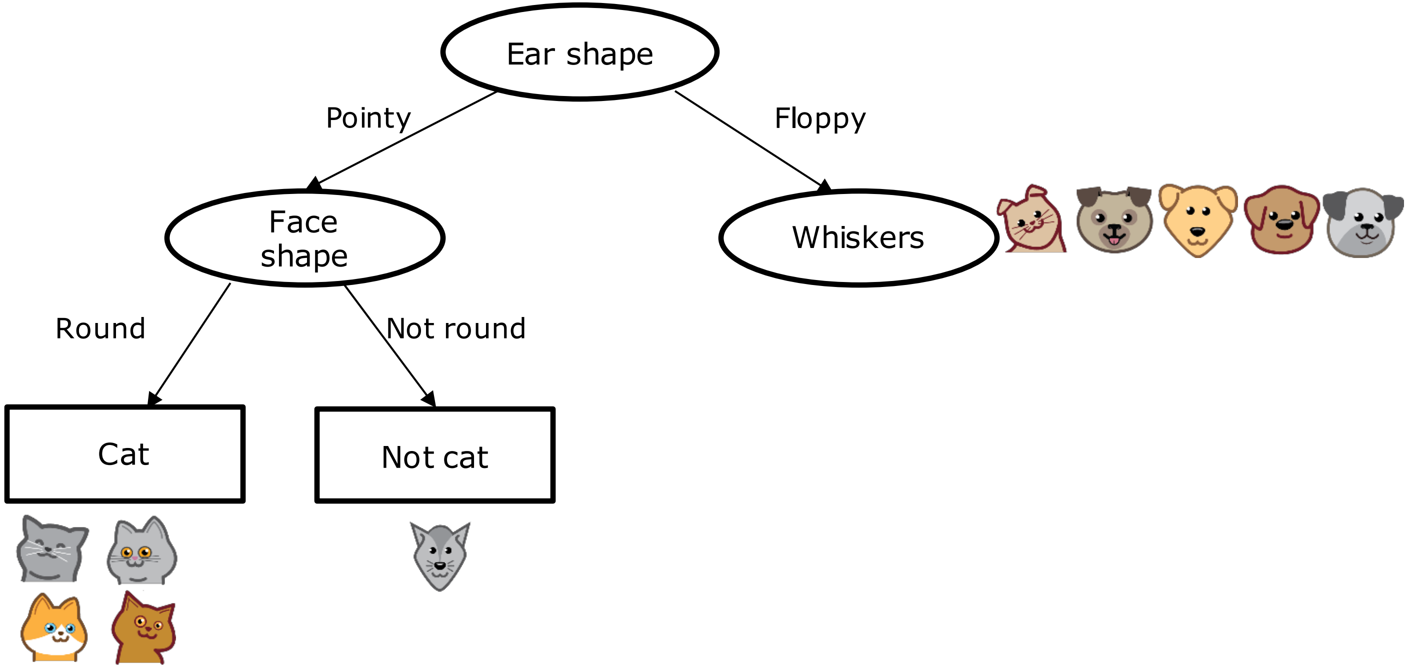
Decision Tree Learning



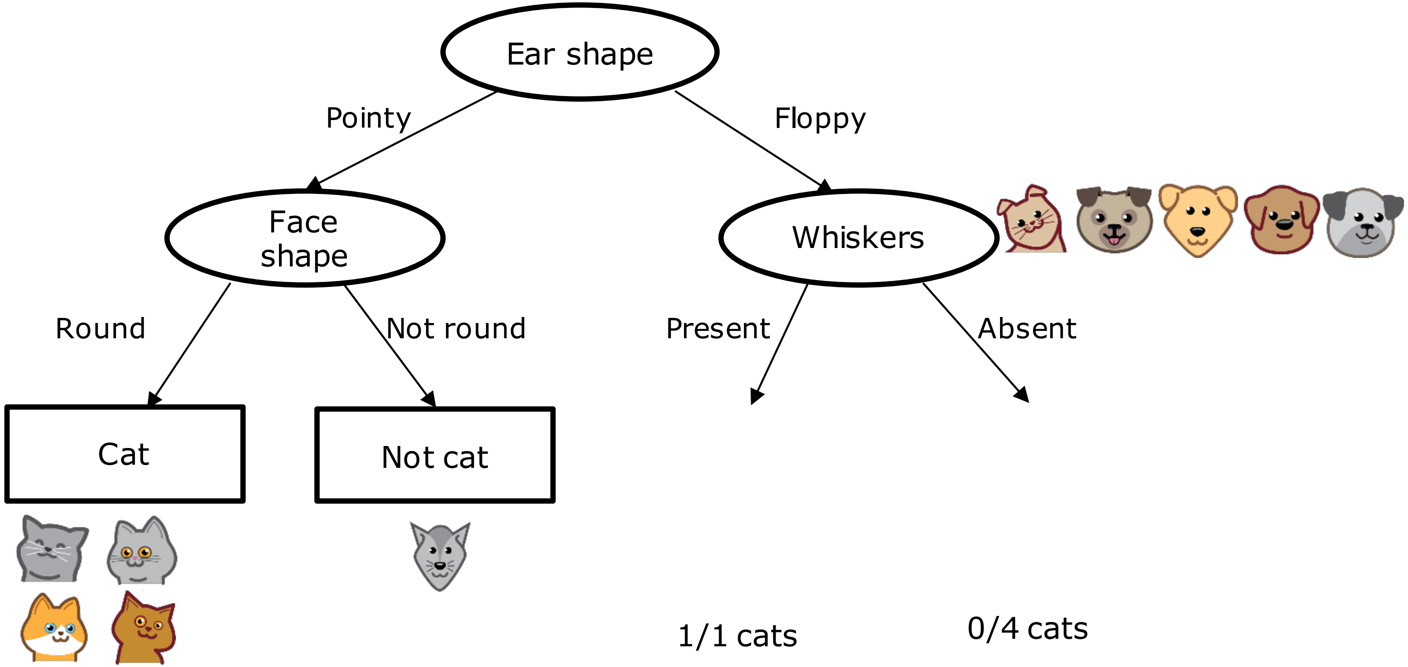
Decision Tree Learning



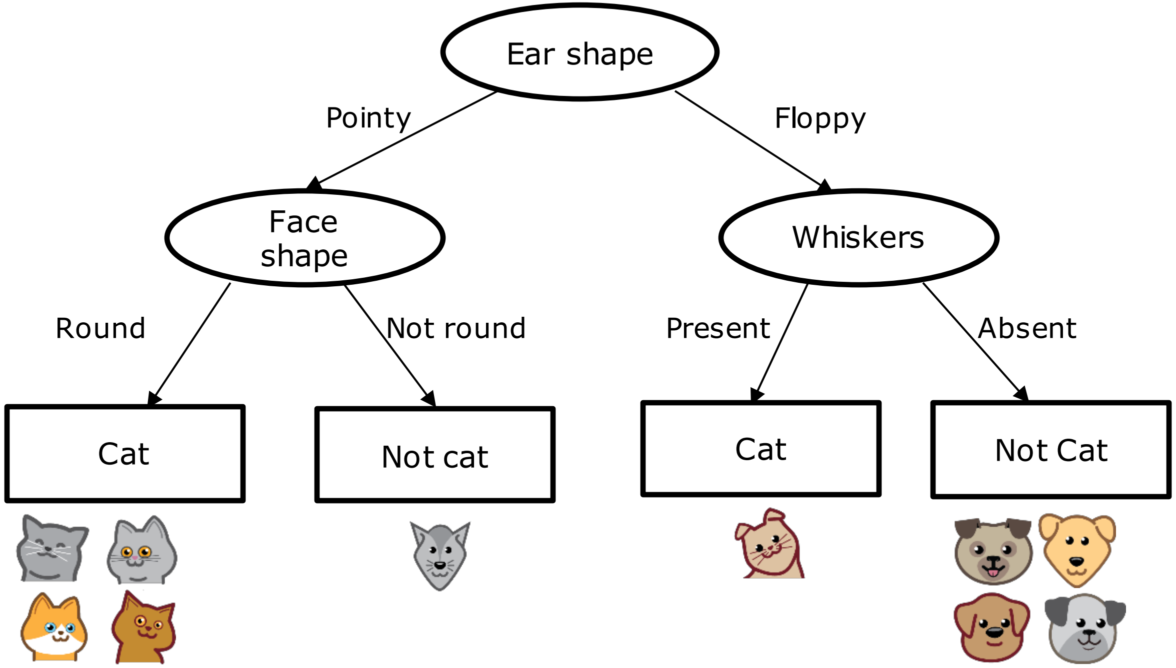
Decision Tree Learning



Decision Tree Learning



Decision Tree Learning



Decision Tree Learning

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

?

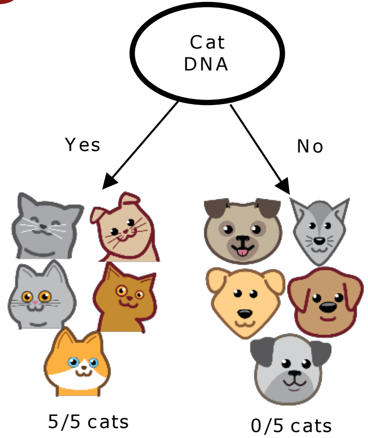
Maximize purity (or minimize impurity)

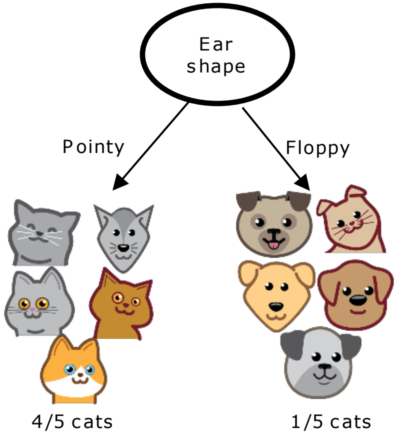
all cats or all dogs

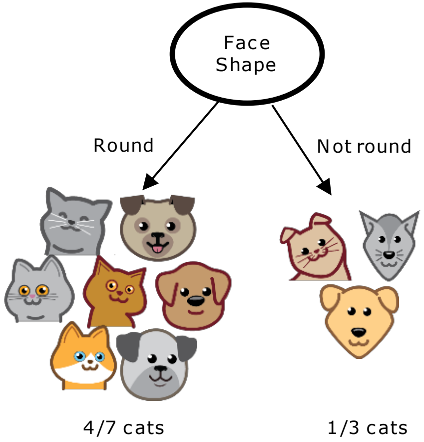
Decision Tree Learning

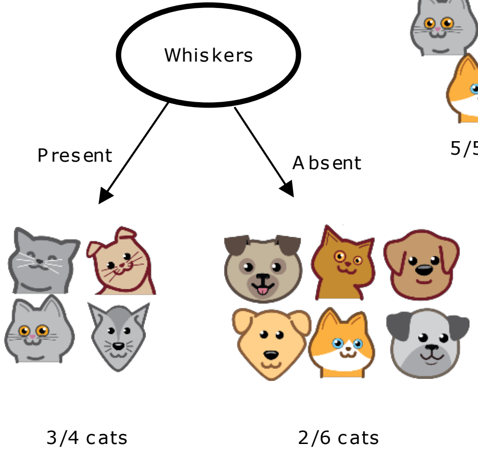
Decision 1: How to choose what feature to split on at each node?

Maximize purity (or minimize impurity)





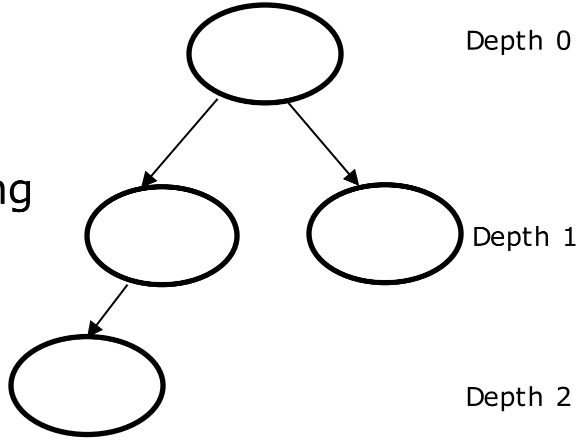




Decision Tree Learning

Decision 2: When do you stop splitting?

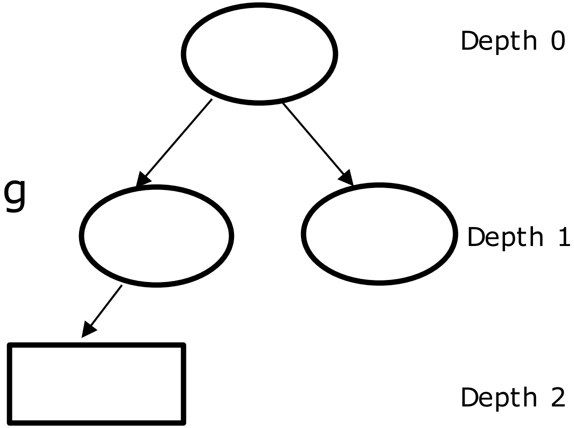
- When a node is 100% one class
- When splitting a node will result in the tree exceeding a maximum depth



Decision Tree Learning

Decision 2: When do you stop splitting?

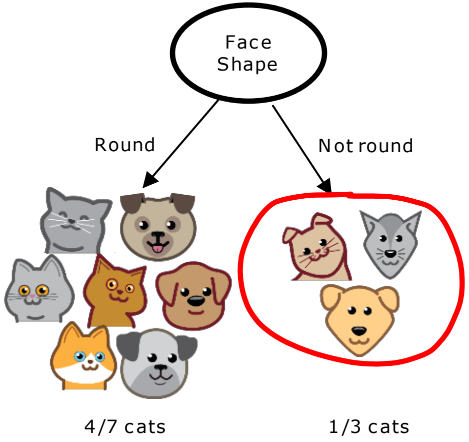
- When a node is 100% one class
- When splitting a node will result in the tree exceeding a maximum depth {To not make a very big tree & reduce the risk of overfitting}



Decision Tree Learning

Decision 2: When do you stop splitting?

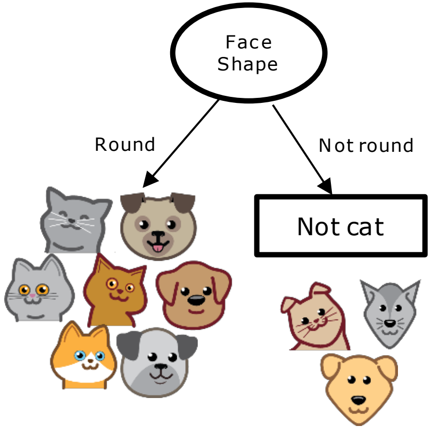
- When a node is 100% one class
- When splitting a node will result in the tree exceeding a maximum depth
- When improvements in purity score are below a threshold
- When number of examples in a node is below a threshold



Decision Tree Learning

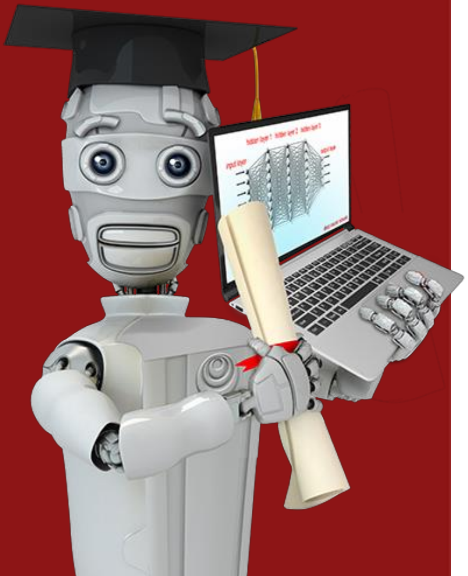
Decision 2: When do you stop splitting?

- 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



DeepLearning.AI

Stanford ONLINE



Decision Tree Learning

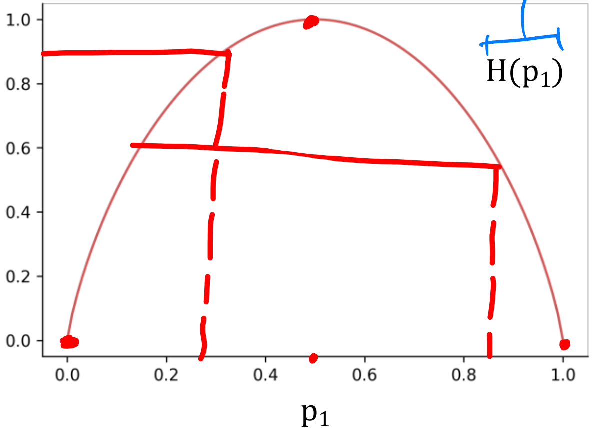
Measuring purity

Entropy

Entropy as a measure of impurity

$p_1 =$ fraction of examples that are cats

entropy



$p_1 = 0 \quad H(p_1) = 0$

$p_1 = 2/6 \quad H(p_1) = 0.92$

$p_1 = 3/6 \quad H(p_1) = 1$

more pure

$p_1 = 5/6 \quad H(p_1) = 0.65$

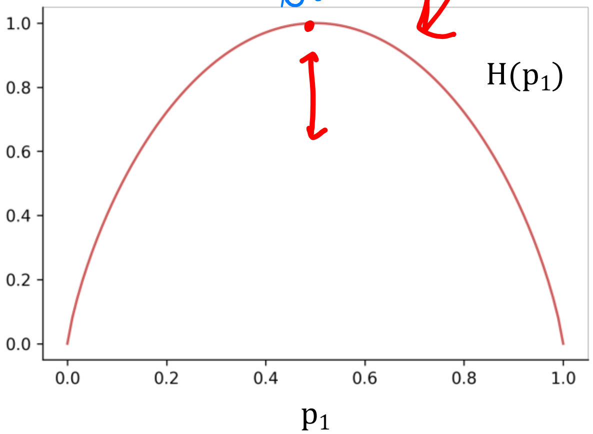
$p_1 = 6/6 \quad H(p_1) = 0$

$H(P_1) \propto$ Pure

Entropy as a measure of impurity

p_1 = fraction of examples that are cats

base 2 makes the peak at 1



$$p_0 = 1 - p_1$$

$$H(p_1) = -p_1 log_2(p_1) - p_0 log_2(p_0)$$

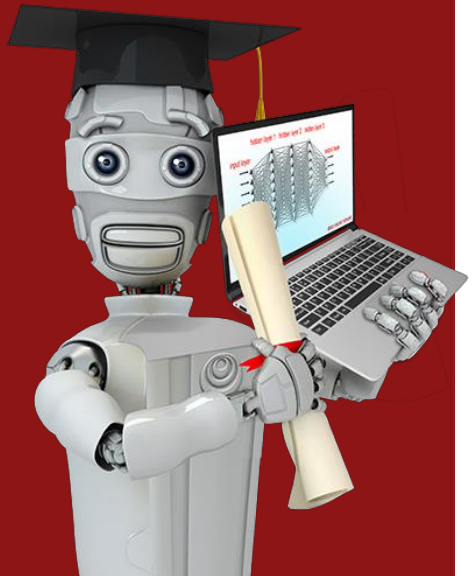
$$\boxed{H(p_1) = -p_1\ log_2(p_1) - (1-p_1) log_2(1-p_1)}$$

Note: "$0\log(0)$" $= 0$

if $p_1 = 0$

DeepLearning.AI

Stanford
ONLINE



Decision Tree Learning

Choosing a split: Information Gain

reducing impurity

rootnode entropy {we began with 5cats & 5 dogs}

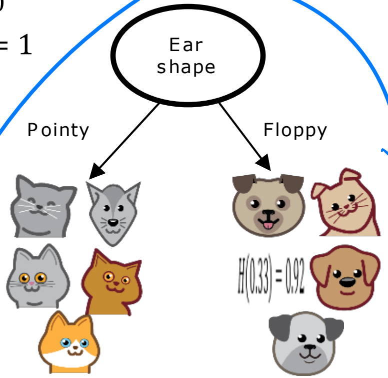
Choosing a split

$p_1 = {}^{5}/_{10} = 0.5$

$H(0.5) = 1$

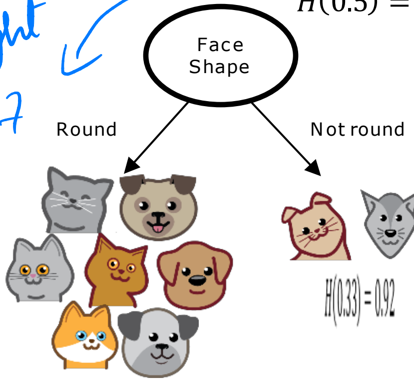
10

w^{left} 5



5

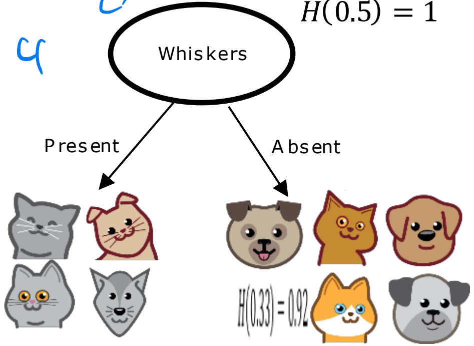
w^{right} 7



$H(0.5) = 1$

3

4



$H(0.5) = 1$

6

$p_1 = {}^{4}/_{5} = 0.8$ $\quad p_1 = {}^{1}/_{5} = 0.2$

$H(0.8) = 0.72$ $\quad H(0.2) = 0.72$

$$\underbrace{H(0.5)}_{} - \underbrace{\left(\frac{5}{10}H(0.8) + \frac{5}{10}H(0.2)\right)}_{\text{weighted avg.}}$$

$= 0.28$

info. gain

red'n in entropy

$p_1 = {}^{4}/_{7} = 0.57$ $\quad p_1 = {}^{1}/_{3} = 0.33$

$H(0.57) = 0.99$ $\quad H(0.33) = 0.92$

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

$= 0.03$

if it's too small then no splitting to avoid overfitting

$p_1 = {}^{3}/_{4} = 0.75$ $\quad p_1 = {}^{2}/_{6} = 0.33$

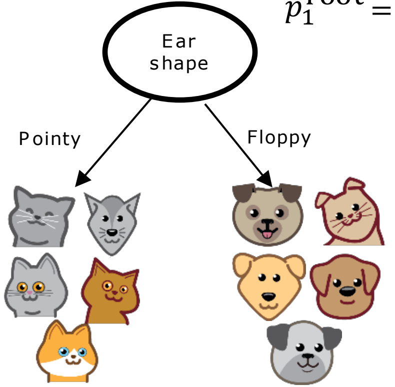
$H(0.75) = 0.81$ $\quad H(0.33) = 0.92$

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

$= 0.12$

Information Gain

$p_1^{\text{root}} = 5/10 = 0.5$



$p_1^{\text{left}} = 4/5$ $\quad$ $p_1^{\text{right}} = 1/5$

$w^{\text{left}} = 5/10$ $\quad$ $w^{\text{right}} = 5/10$

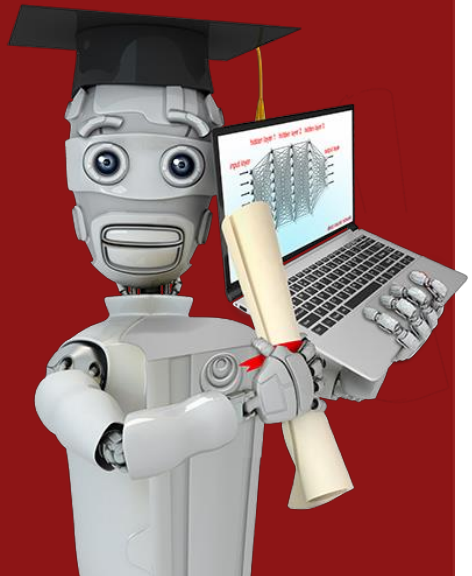
red^n in entropy

Information gain

$$= H(p_1^{\text{root}}) - \left(w^{\text{left}} H\left(p_1^{\text{left}}\right) + w^{\text{right}} H\left(p_1^{\text{right}}\right) \right)$$

DeepLearning.AI

Stanford ONLINE



Decision Tree Learning

Putting it together

Decision Tree Learning

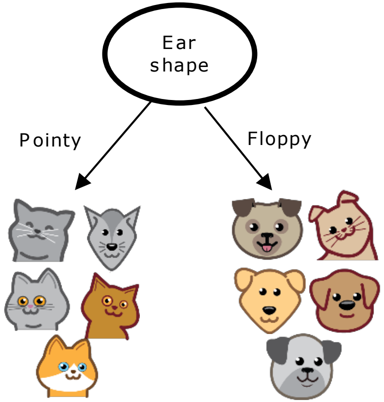
- Start with all examples at the root node
- Calculate information gain for all possible features, and pick the one with the highest information gain
- Split dataset according to selected feature, and create left and right branches of the tree
- Keep repeating splitting process until stopping criteria 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

Recursive splitting

Recursive splitting

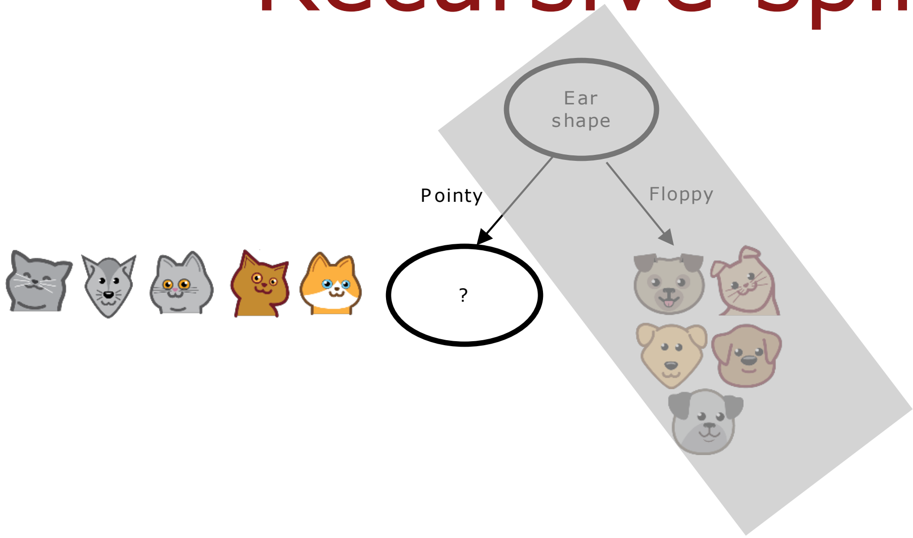
Recursive splitting

Recursive splitting

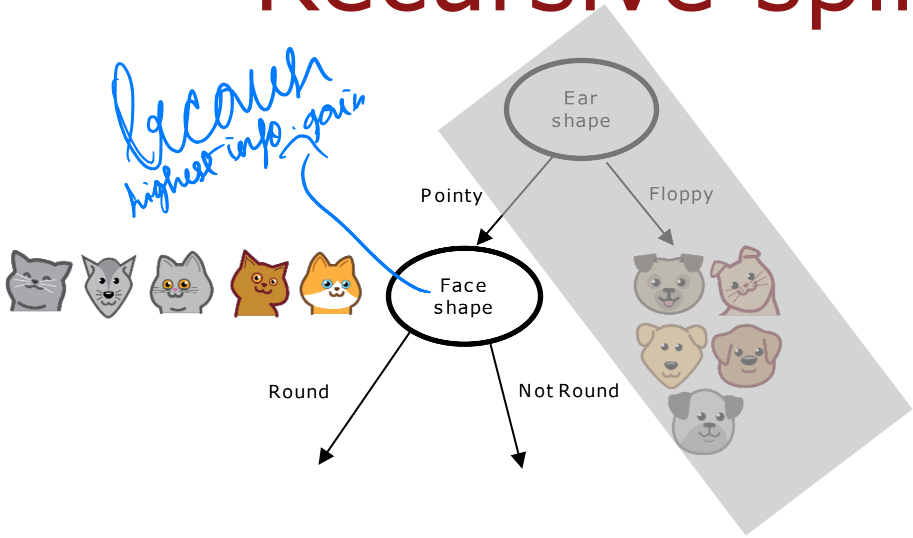


Recursive splitting

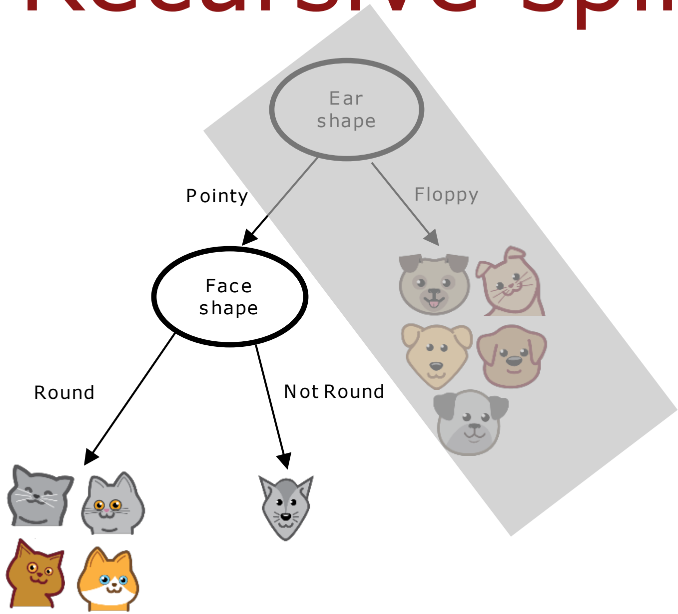
Recursive splitting



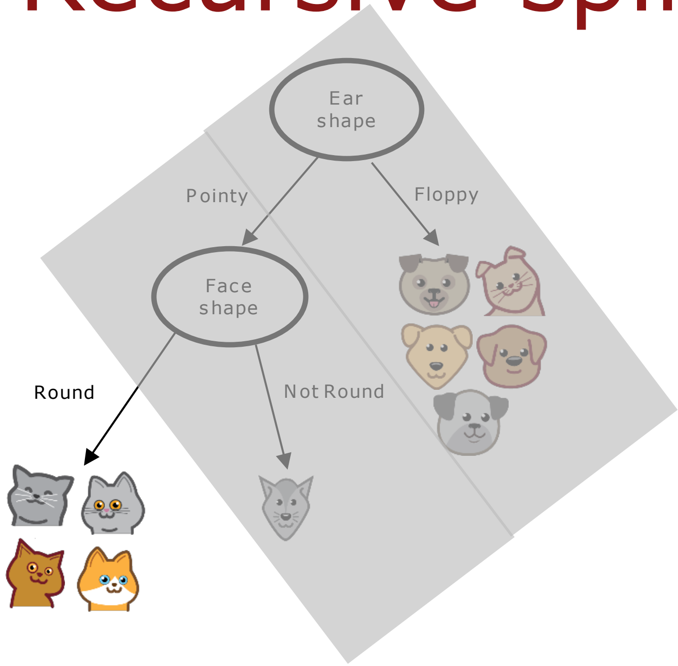
Recursive splitting



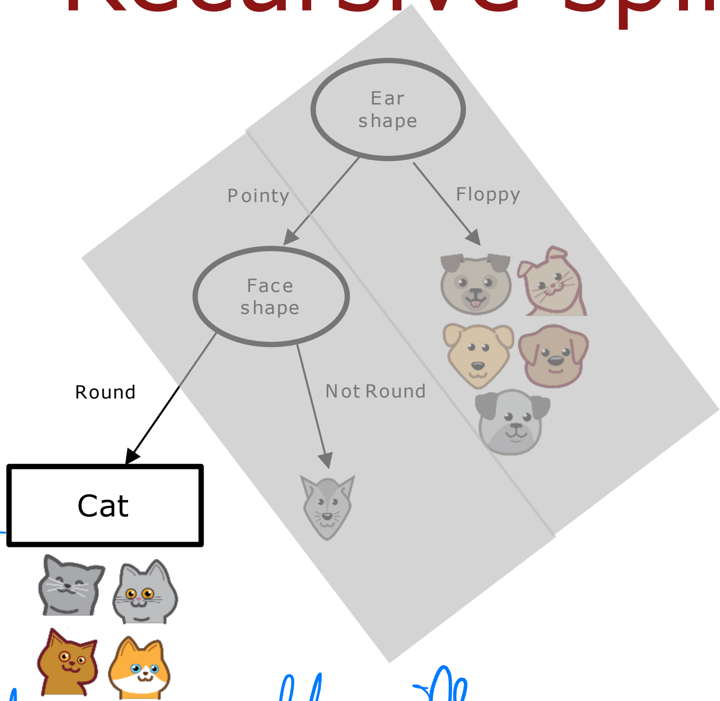
Recursive splitting



Recursive splitting

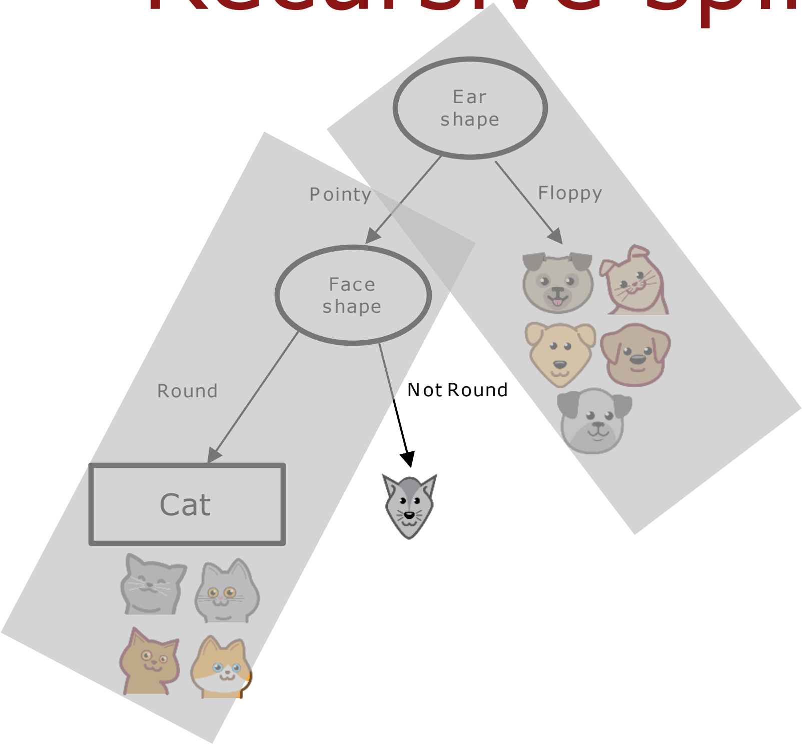


Recursive splitting

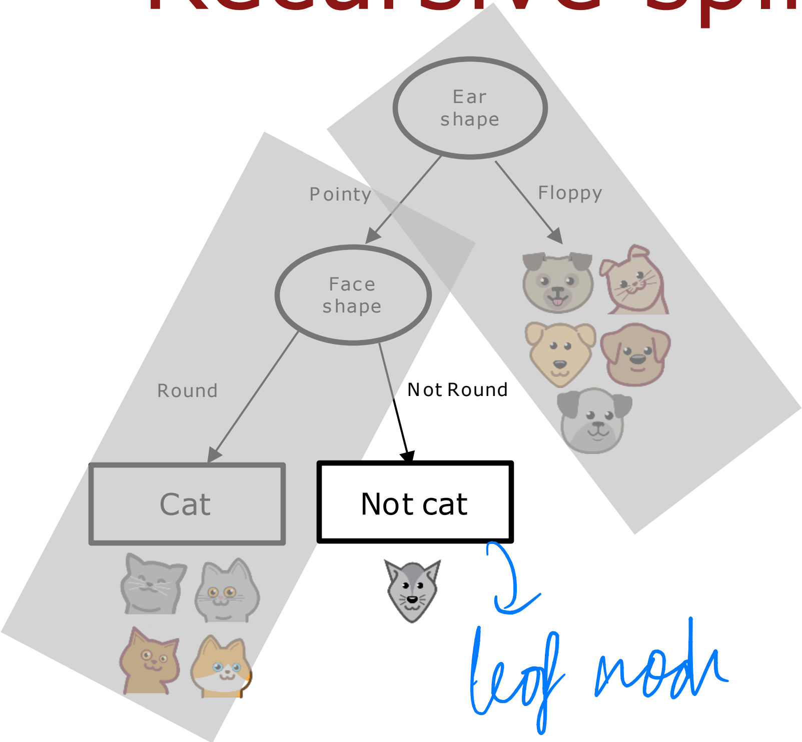


leaf node because all cats

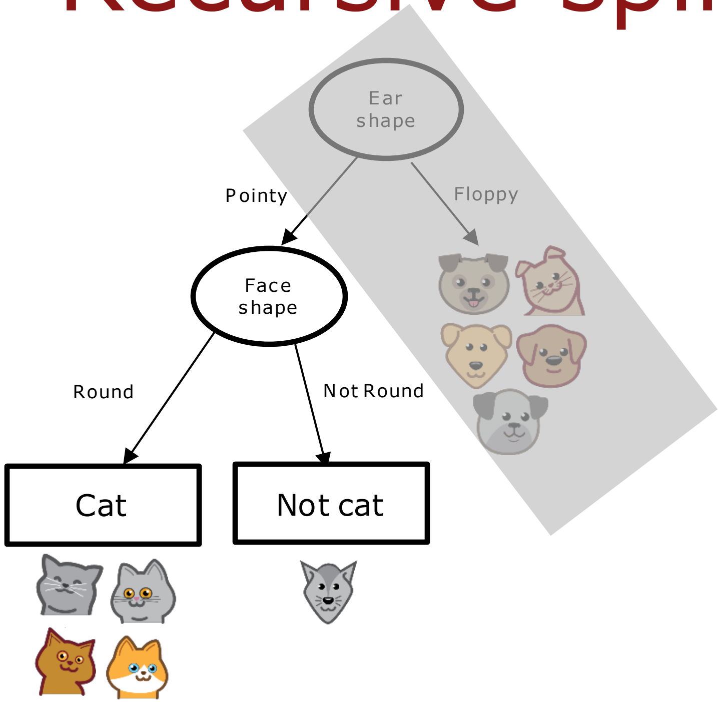
Recursive splitting



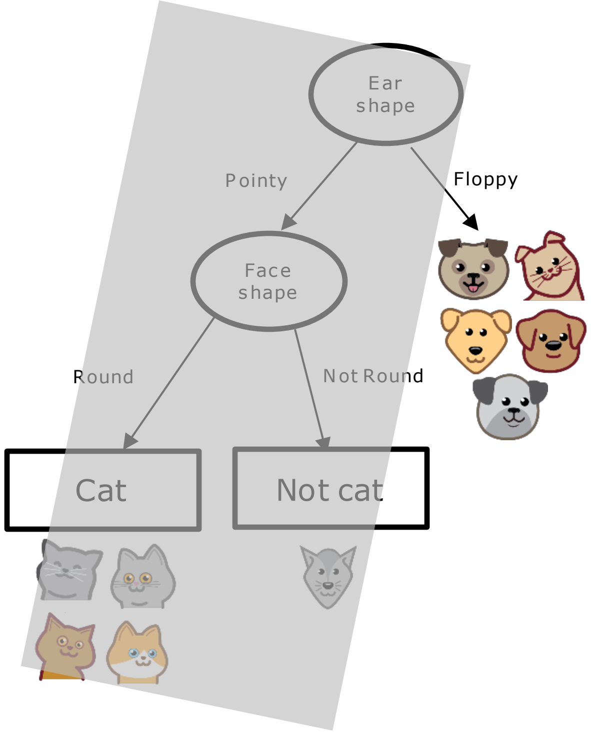
Recursive splitting



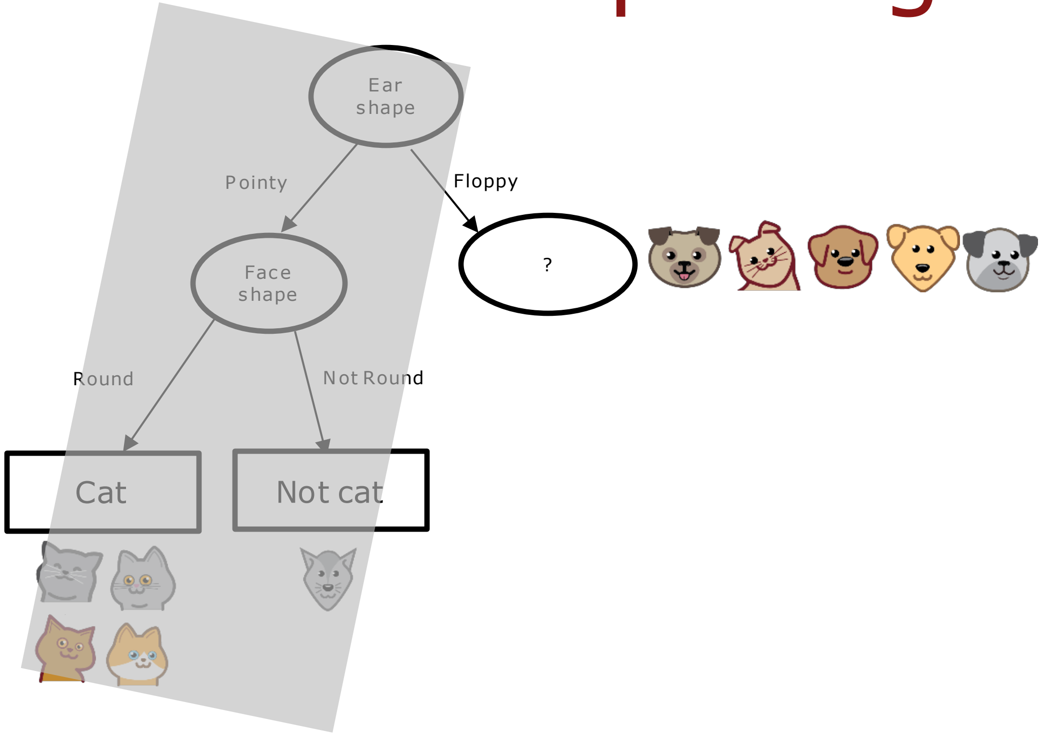
Recursive splitting



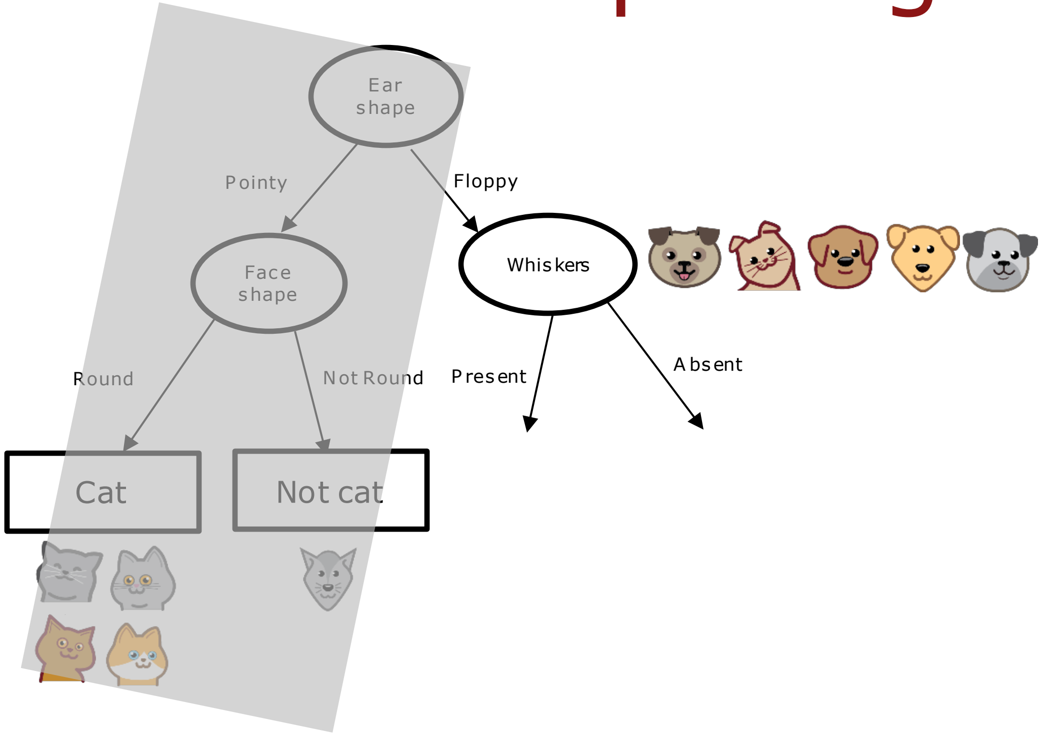
Recursive splitting



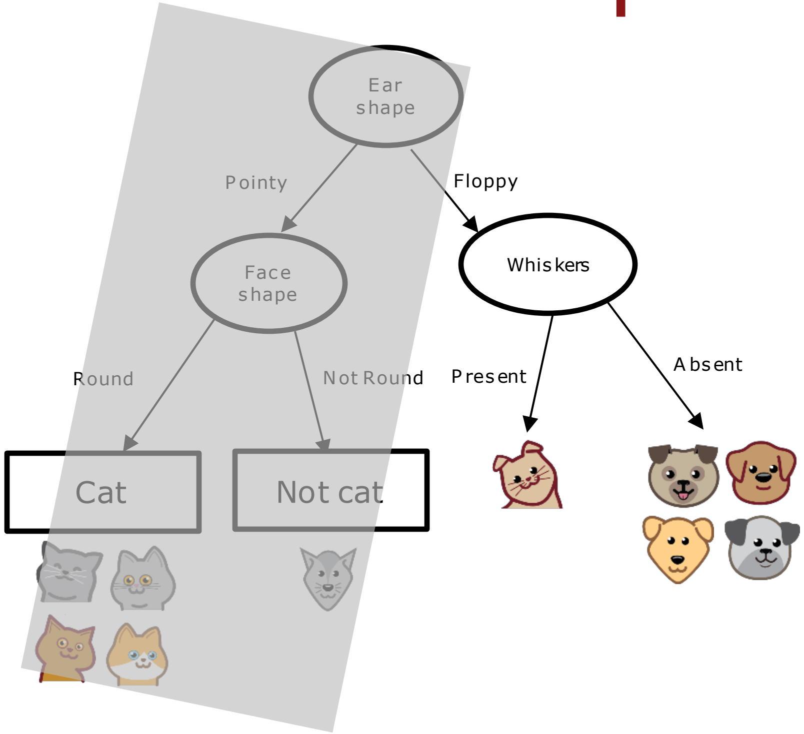
Recursive splitting



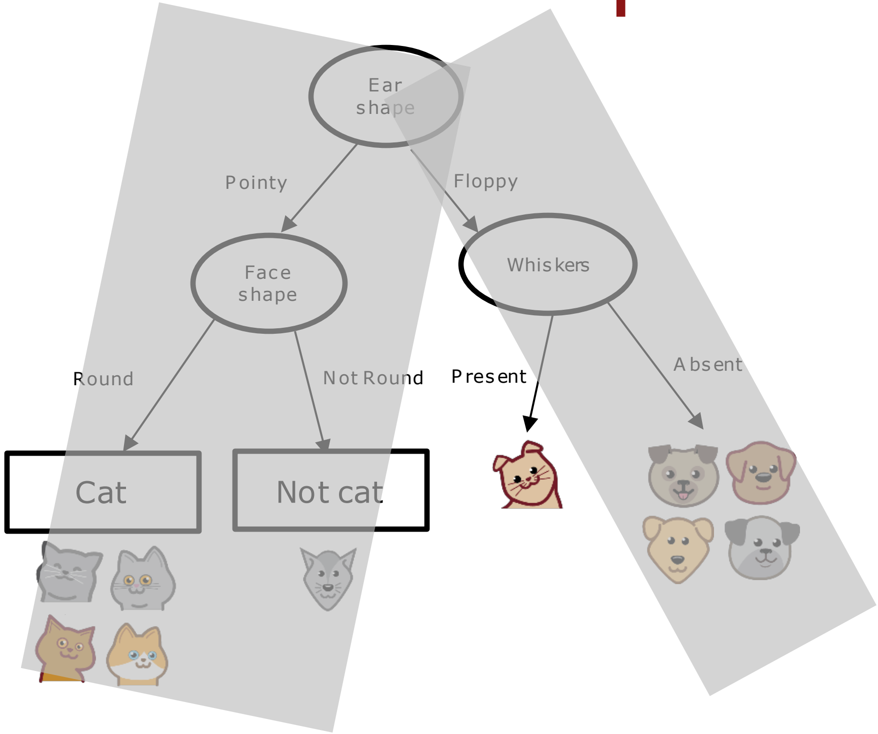
Recursive splitting



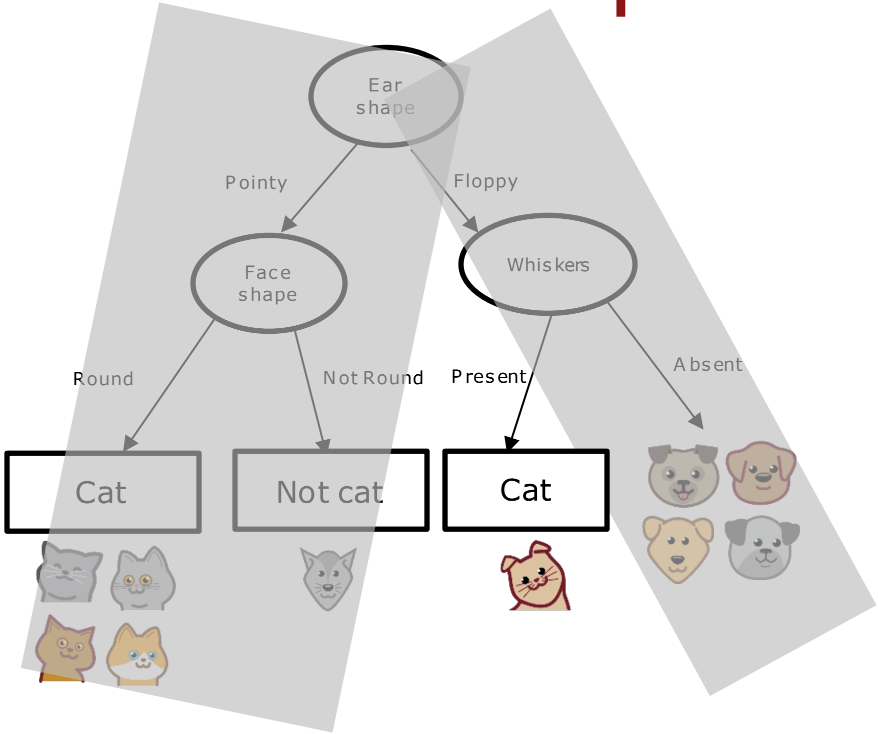
Recursive splitting



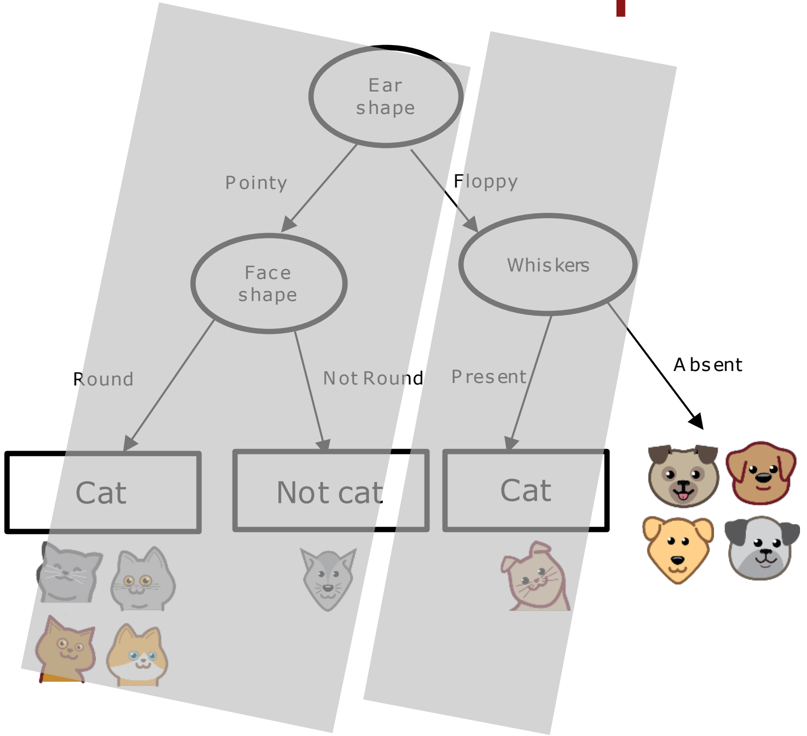
Recursive splitting



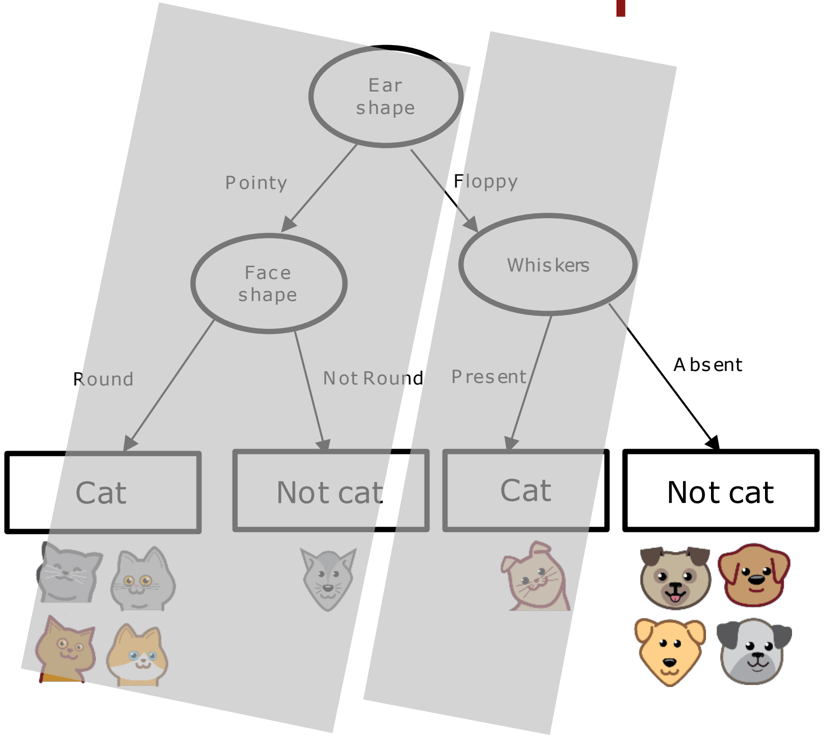
Recursive splitting



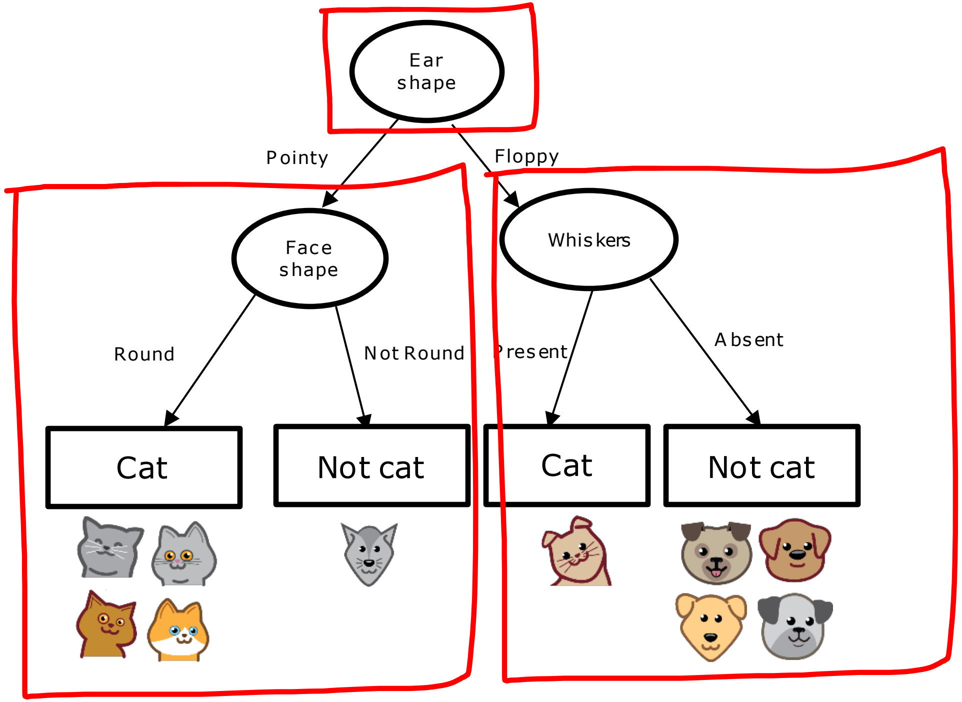
Recursive splitting



Recursive splitting



Recursive splitting



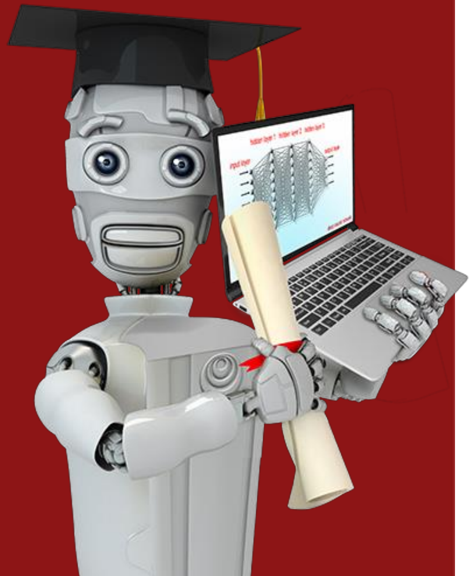
Recursive algorithm

DeepLearning.AI

Stanford ONLINE

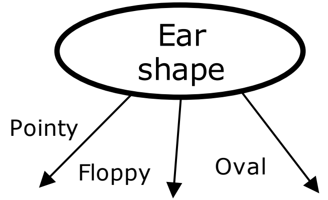
Decision Tree Learning

Using one-hot encoding of categorical features



Features with three possible values

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



3 possible values

One hot encoding

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

One hot encoding

If a categorical feature can take on k values, create k binary features (0 or 1 valued).

One hot encoding

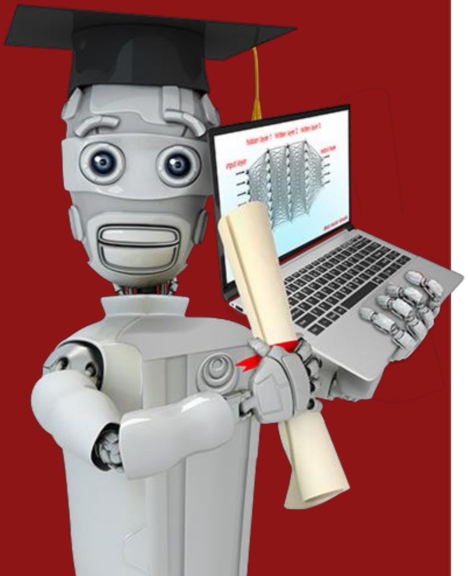
~~Ear shape~~	Pointy ears	Floppy ears	Oval ears	Face shape	Whiskers	Cat
~~Pointy~~	1	0	0	Round	Present	1
~~Oval~~	0	0	1	Not round	Present	1
~~Oval~~	0	0	1	Round	Absent	0
~~Pointy~~	1	0	0	Not round	Present	0
~~Oval~~	0	0	1	Round	Present	1
~~Pointy~~	1	0	0	Round	Absent	1
~~Floppy~~	0	1	0	Not round	Absent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

One hot encoding and neural networks

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

DeepLearning.AI

Stanford
ONLINE



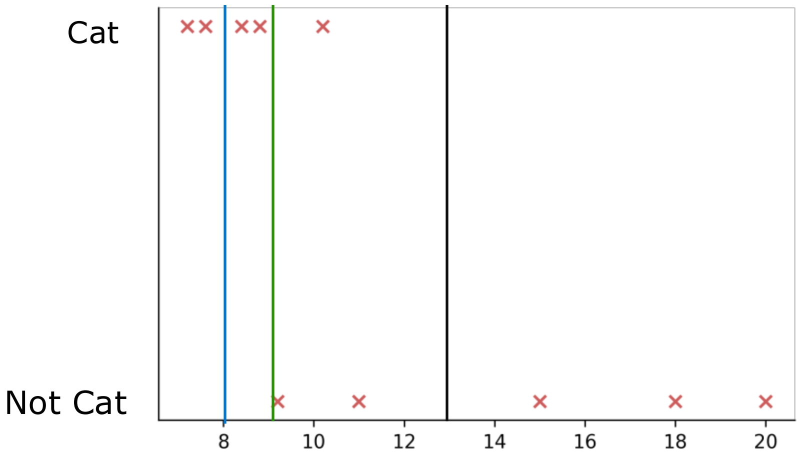
Decision Tree Learning

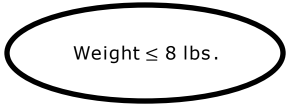
Continuous valued features

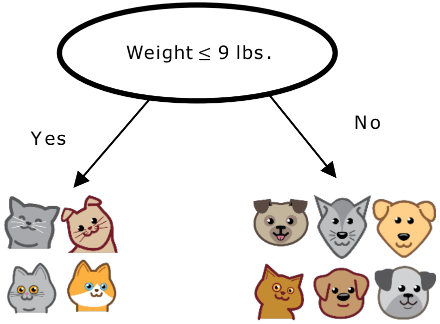
Continuous features

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

Splitting on a continuous variable







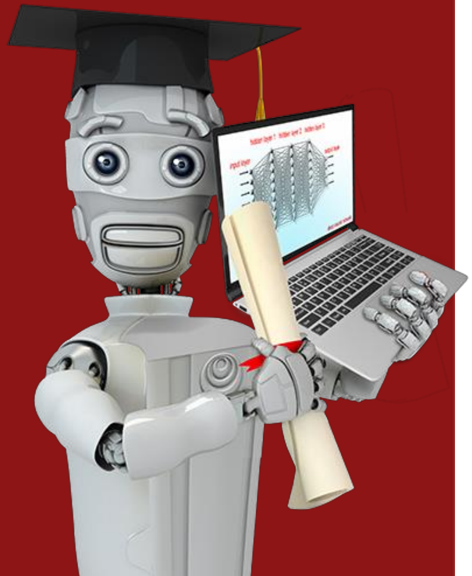
$$H(0.5) - \left(\frac{2}{10} H\left(\frac{2}{2}\right) + \frac{8}{10} H\left(\frac{3}{8}\right)\right) = 0.24$$

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

$$H(0.5) - \left(\frac{7}{10} H\left(\frac{5}{7}\right) + \frac{3}{10} H\left(\frac{0}{3}\right)\right) = 0.40$$

DeepLearning.AI

Stanford ONLINE



Decision Tree Learning

Regression Trees (optional)

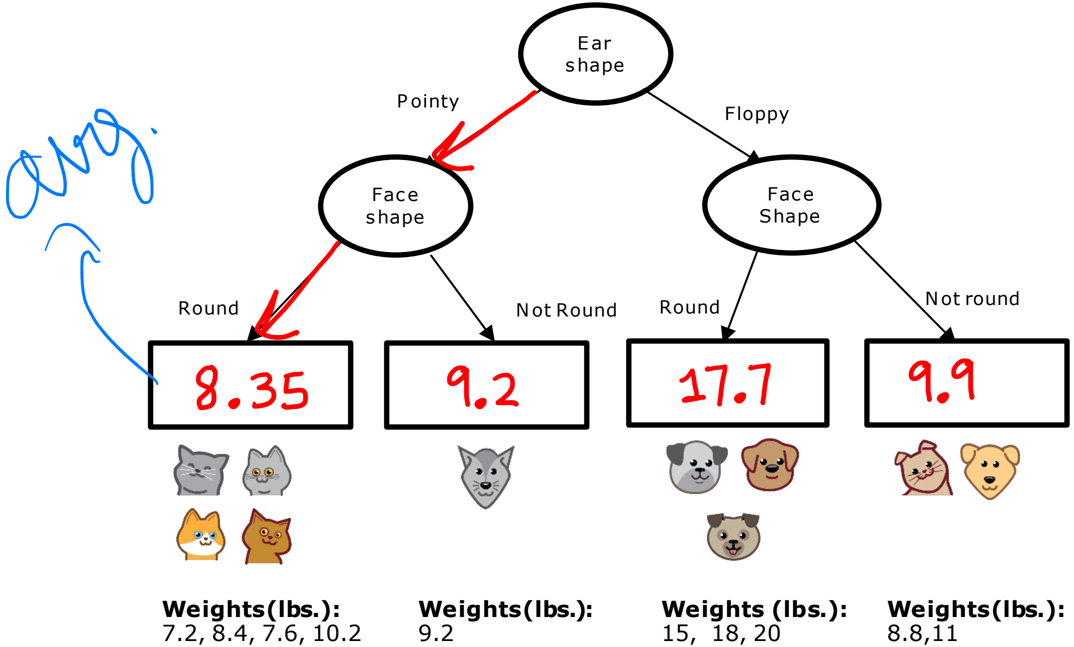
Regression with Decision Trees: Predicting a number

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

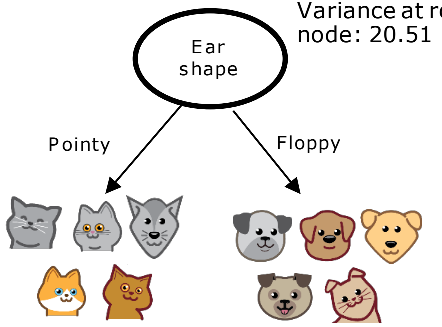
y

Regression with Decision Trees



$s^2 = \frac{\sum (x_i - \bar{x})^2}{n-1}$

Choosing a split



Variance at root node: 20.51

Weights: 7.2, 9.2, 8.4,7.6, 10.2

Variance: 1.47

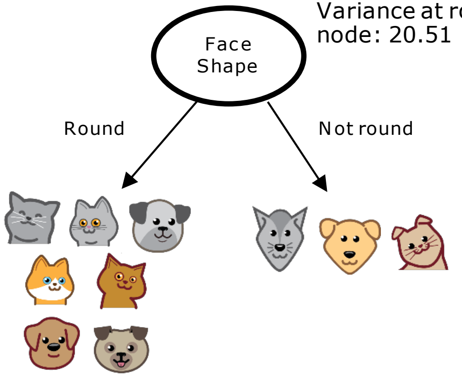
$w^{\text{left}} = 5/10$

Weights: 8.8, 15, 11, 18, 20

Variance: 21.87

$w^{\text{right}} = 5/10$

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



Variance at root node: 20.51

Weights: 7.2, 15, 8.4, 7.6,10.2, 18, 20

Variance: 27.80

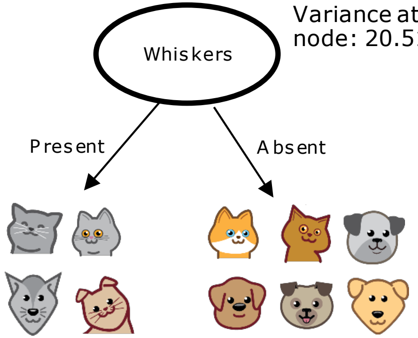
$w^{\text{left}} = 7/10$

Weights: 8.8,9.2,11

Variance: 1.37

$w^{\text{right}} = 3/10$

$$20.51 - \left(\frac{7}{10} * 27.80 + \frac{3}{10} * 1.37\right)$$
$$= 0.64$$



Variance at root node: 20.51

Weights: 7.2, 8.8, 9.2, 8.4

Variance: 0.75

$w^{\text{left}} = 4/10$

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

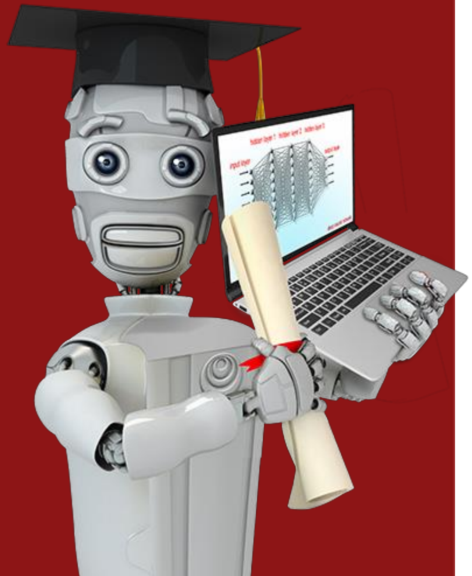
Variance: 23.32

$w^{\text{right}} = 6/10$

$$20.51 - \left(\frac{4}{10} * 0.75 + \frac{6}{10} * 23.32\right)$$
$$= 6.22$$

DeepLearning.AI

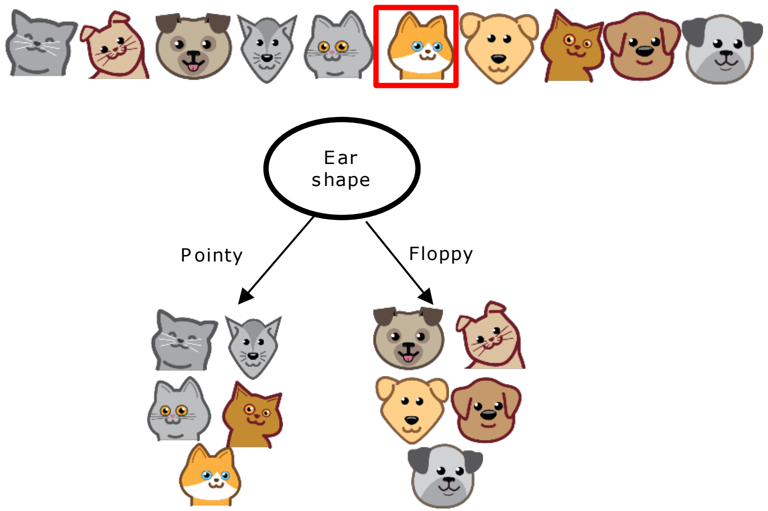
Stanford ONLINE

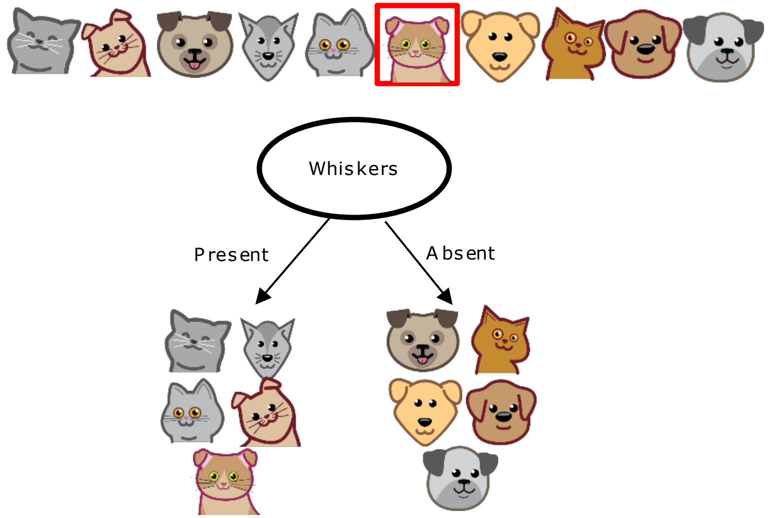


Tree ensembles

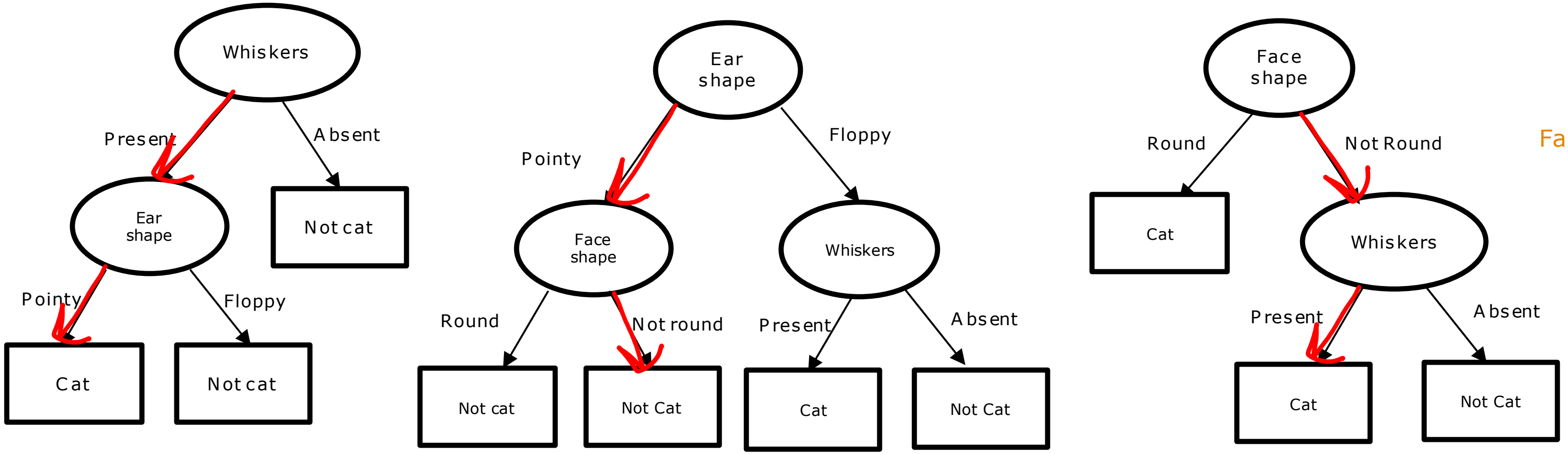
Using multiple decision trees

Trees are highly sensitive to small changes of the data





Tree ensemble



Prediction: Cat

Prediction: Not cat

Prediction: Cat

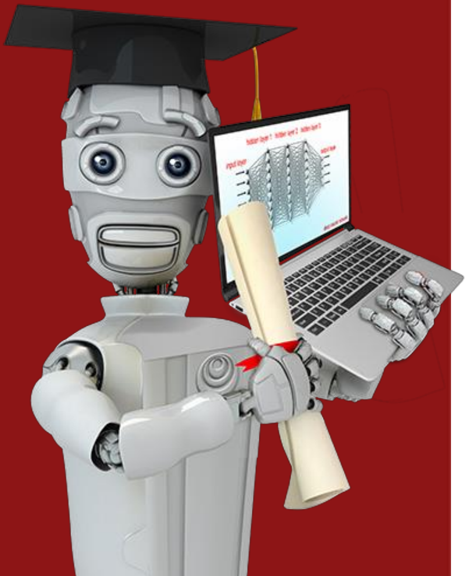
New test example

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

Final prediction: Cat

DeepLearning.AI

Stanford ONLINE



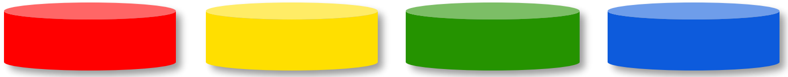
Tree ensembles

Sampling with replacement

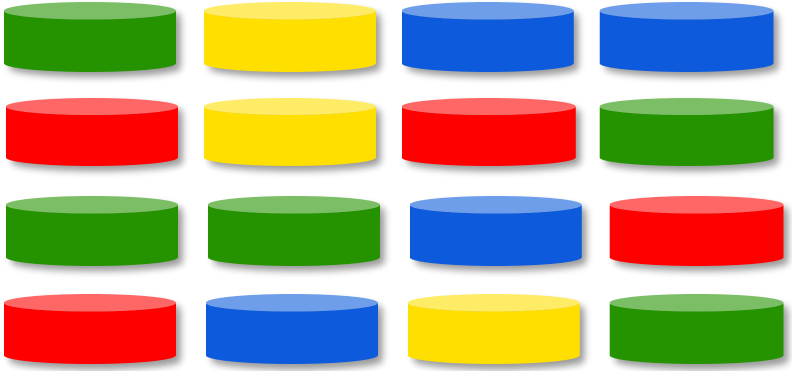
Sampling with replacement

Putting it back

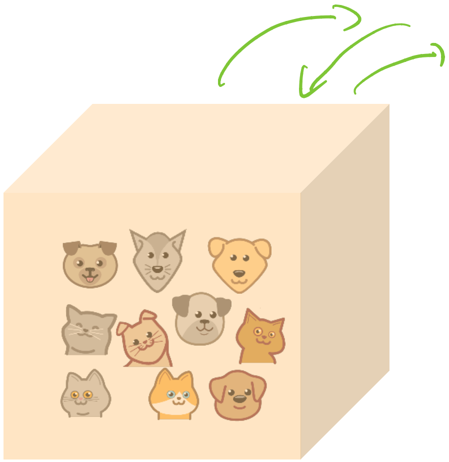
Tokens



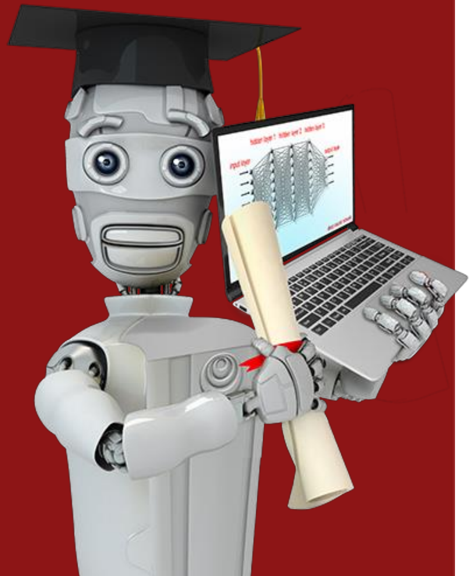
Sampling with replacement:



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

To build tree ensemble

Generating a tree sample

Given training set of size m

For b = 1 to B: no. of trees

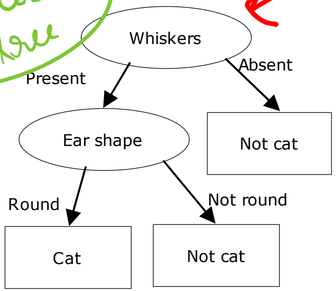
Use sampling with replacement to create a new training set of size m

Train a decision tree on the new dataset

Bag decision tree

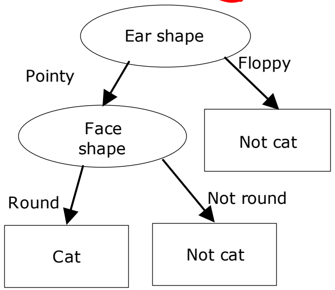
1st decision tree

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	Present	Yes
Pointy	Not Round	Absent	No
Pointy	Not Round	Absent	No
Pointy	Not Round	Present	Yes



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

2nd decision tree



...

Bagged decision tree

Randomizing the feature choice

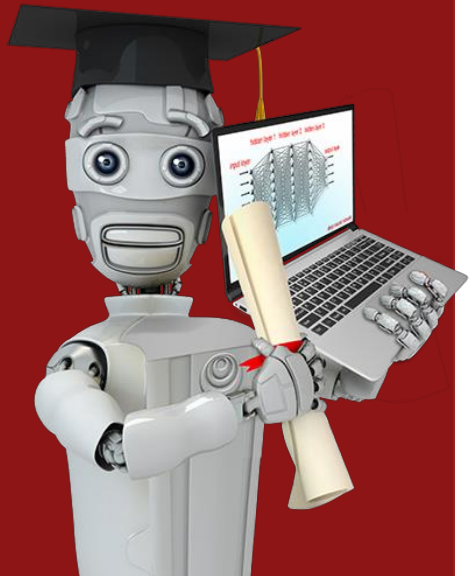
At each node, when choosing a feature to use to split, if n features are available, pick a random subset of $k < n$ features and allow the algorithm to only choose from that subset of features.

$K = \sqrt{n}$ If $n >>$

Random forest algorithm avg a lot of small no of features

DeepLearning.AI

Stanford ONLINE



Tree ensembles

much better

XGBoost

Boosted trees intuition

Given training set of size m

For $b = 1$ to B:

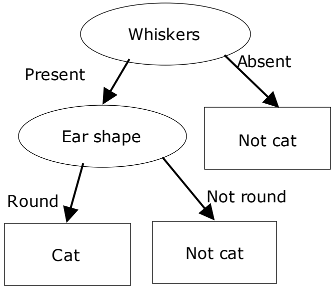
Use sampling with replacement to create a new training set of size m

But instead of picking from all examples with equal (1/m) probability, make it more likely to pick examples that the previously trained trees misclassify

Train a decision tree on the new dataset

generated 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	Present	Yes
Pointy	Not Round	Absent	No
Pointy	Not Round	Absent	No
Pointy	Not Round	Present	Yes



1, 2, ..., b-1

Higher chance to pick b

Original dataset

Ear shape	Face shape	Whiskers	Prediction	
Pointy	Round	Present	Cat	✓
Floppy	Not Round	Present	Not cat	✗
Floppy	Round	Absent	Not cat	✓
Pointy	Not Round	Present	Not cat	✓
Pointy	Round	Present	Cat	✓
Pointy	Round	Absent	Not cat	✗
Floppy	Not Round	Absent	Not cat	✓
Pointy	Round	Absent	Not cat	✗
Floppy	Round	Absent	Not cat	✓
Floppy	Round	Absent	Not cat	✓

XGBoost (eXtreme Gradient Boosting)

- Open source implementation of boosted trees
- Fast efficient implementation
- Good choice of default splitting criteria and criteria for when to stop splitting
- Built in regularization to prevent overfitting
- Highly competitive algorithm for machine learning competitions (eg: Kaggle competitions)

Using XGBoost

Classification

```
from xgboost import XGBClassifier

model = XGBClassifier()

model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

Regression

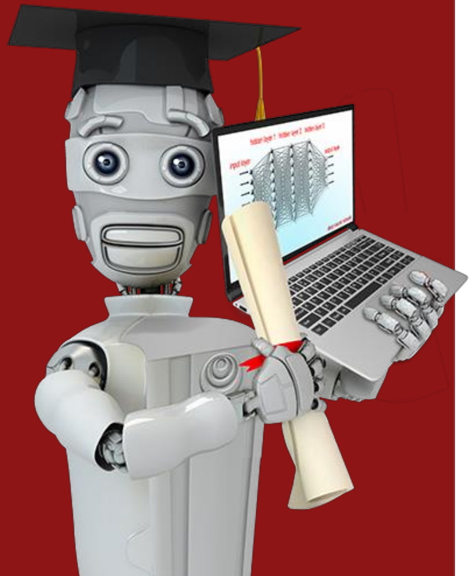
```
from xgboost import XGBRegressor

model = XGBRegressor()

model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

DeepLearning.AI

Stanford
ONLINE



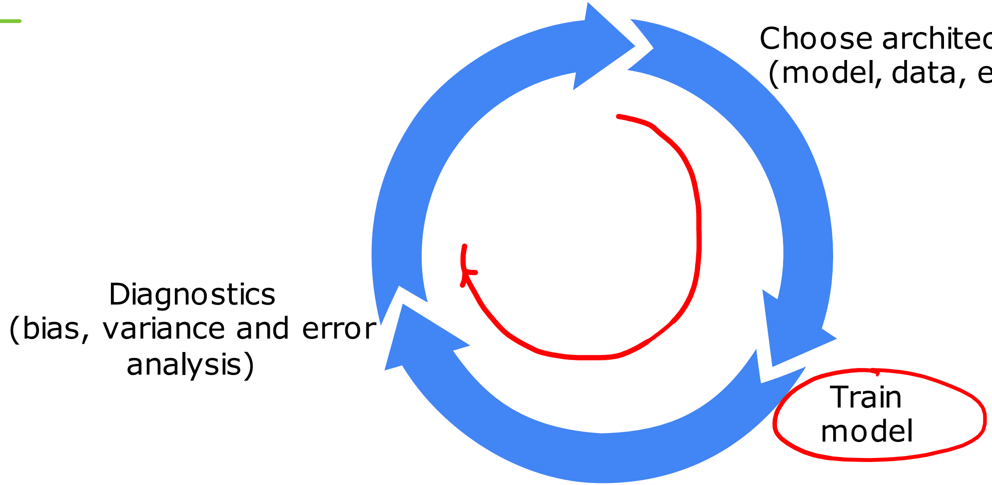
Conclusion

When to use decision trees

Decision Trees vs Neural Networks

Decision Trees and Tree ensembles

- Works well on tabular (structured) data {housing prices}
- Not recommended for unstructured data (images, audio, text)
- Fast



Decision Trees vs Neural Networks

Decision Trees and Tree ensembles

- Works well on tabular (structured) data
- Not recommended for unstructured data (images, audio, text)
- Fast
- Small decision trees may be human interpretable

Neural Networks

- Works well on all types of data, including tabular (structured) and unstructured data
- May be slower than a decision tree
- Works with transfer learning
- When building a system of multiple models working together, it might be easier to string together multiple neural networks