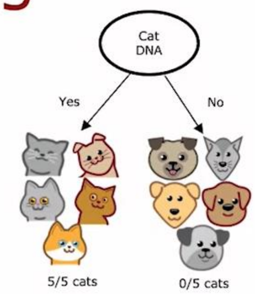
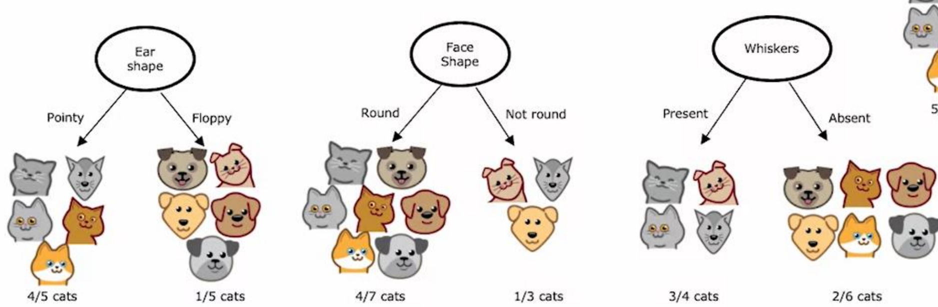
Decision Tree Learning

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

Maximize purity (or minimize impurity)



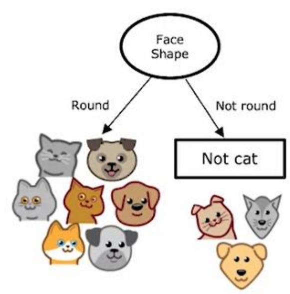


to split on on each node.

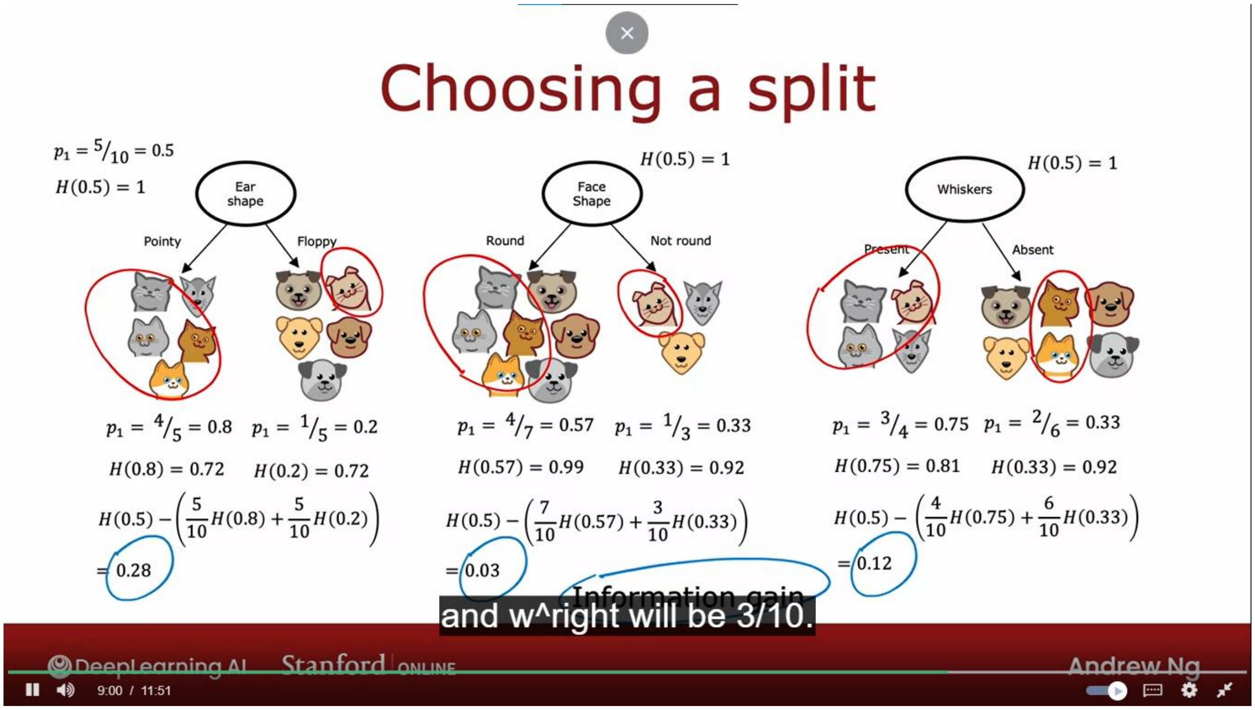
Decision Tree Learning

Decision 2: When do you stop splitting?

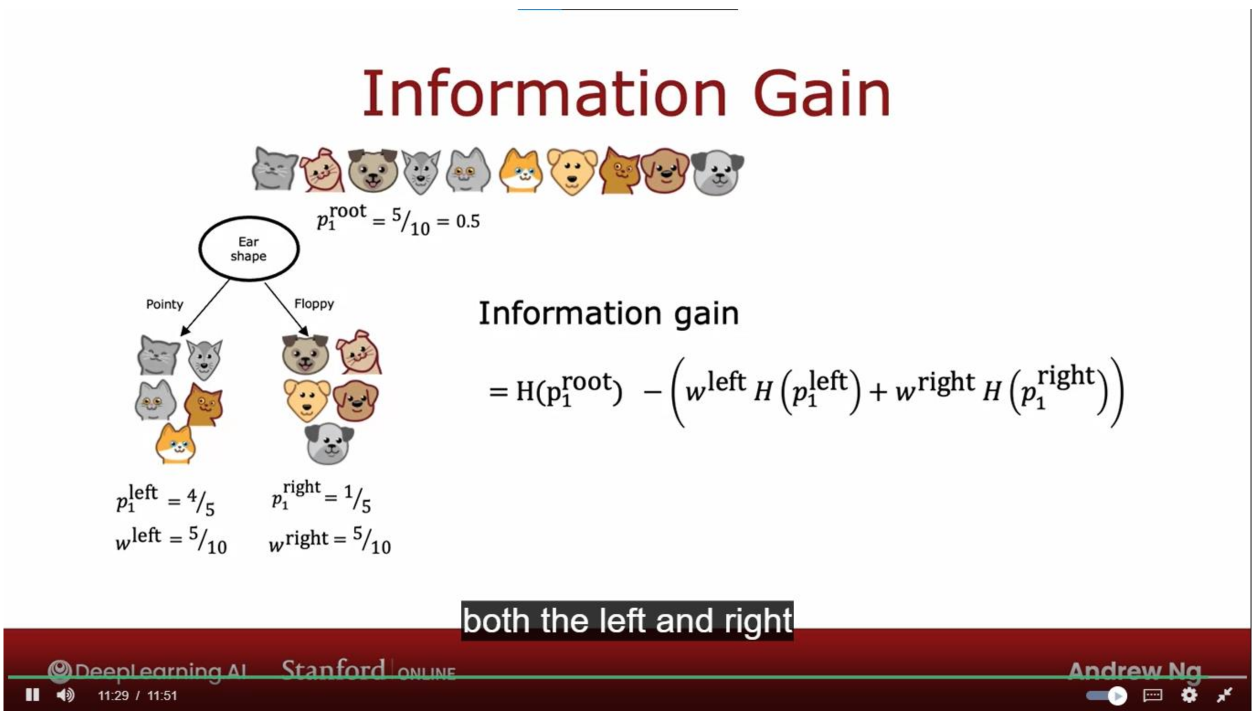
- When a node is 100% one class
- When splitting a node will result in the tree exceeding a maximum depth
- When improvements in purity score are below a threshold
- When number of examples in a node is below a threshold



smaller and to avoid overfitting.



Choosing a split
$p_1 = 5/10 = 0.5$
$H(0.5) = 1$
Ear shape
Pointy
Floppy
$p_1 = 4/5 = 0.8$ $p_1 = 1/5 = 0.2$
$H(0.8) = 0.72$ $H(0.2) = 0.72$
$H(0.5) - \left(\frac{5}{10}H(0.8) + \frac{5}{10}H(0.2)\right)$
$= 0.28$
$H(0.5) = 1$
Face Shape
Round
Not round
$p_1 = 4/7 = 0.57$ $p_1 = 1/3 = 0.33$
$H(0.57) = 0.99$ $H(0.33) = 0.92$
$H(0.5) - \left(\frac{7}{10}H(0.57) + \frac{3}{10}H(0.33)\right)$
$= 0.03$
$H(0.5) = 1$
Whiskers
Present
Absent
$p_1 = 3/4 = 0.75$ $p_1 = 2/6 = 0.33$
$H(0.75) = 0.81$ $H(0.33) = 0.92$
$H(0.5) - \left(\frac{4}{10}H(0.75) + \frac{6}{10}H(0.33)\right)$
$= 0.12$
Information gain
and w^right will be 3/10.
DeepLearning.AI
Stanford ONLINE
Andrew Ng
9:00 / 11:51



Information Gain
$p_1^{\text{root}} = 5/10 = 0.5$
Ear shape
Pointy
Floppy
$p_1^{\text{left}} = 4/5$ $p_1^{\text{right}} = 1/5$
$w^{\text{left}} = 5/10$ $w^{\text{right}} = 5/10$
Information gain
$= H(p_1^{\text{root}}) - \left(w^{\text{left}} H\left(p_1^{\text{left}}\right) + w^{\text{right}} H\left(p_1^{\text{right}}\right)\right)$
both the left and right
DeepLearning.AI
Stanford ONLINE
Andrew Ng
11:29 / 11:51

Decision Tree Learning

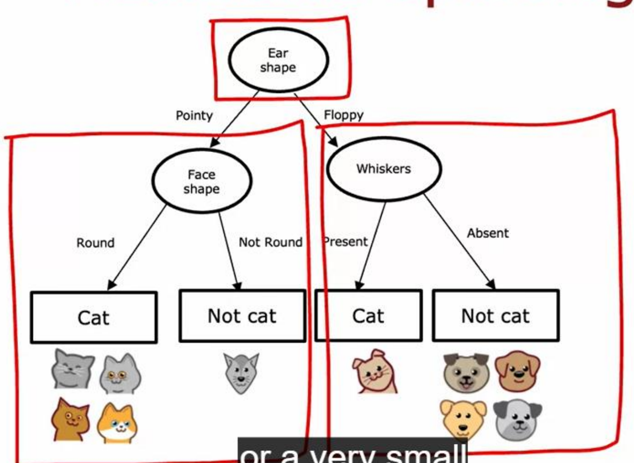
- Start with all examples at the root node
- Calculate information gain for all possible features, and pick the one with the highest information gain
- Split dataset according to selected feature, and create left and right branches of the tree
- Keep repeating splitting process until stopping criteria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

Let's look at an illustration

DeepLearning.AI Stanford ONLINE Andrew Ng

1:50 / 9:07

Recursive splitting

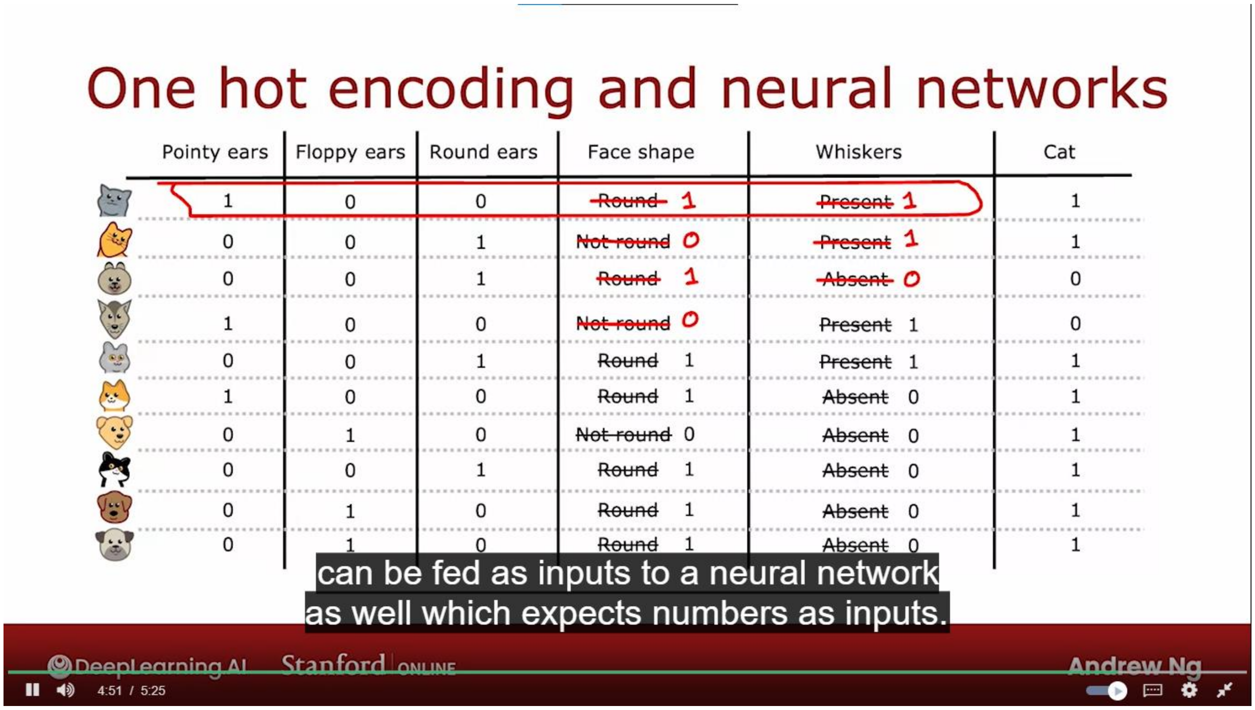


Recursive algorithm

or a very small information gain,

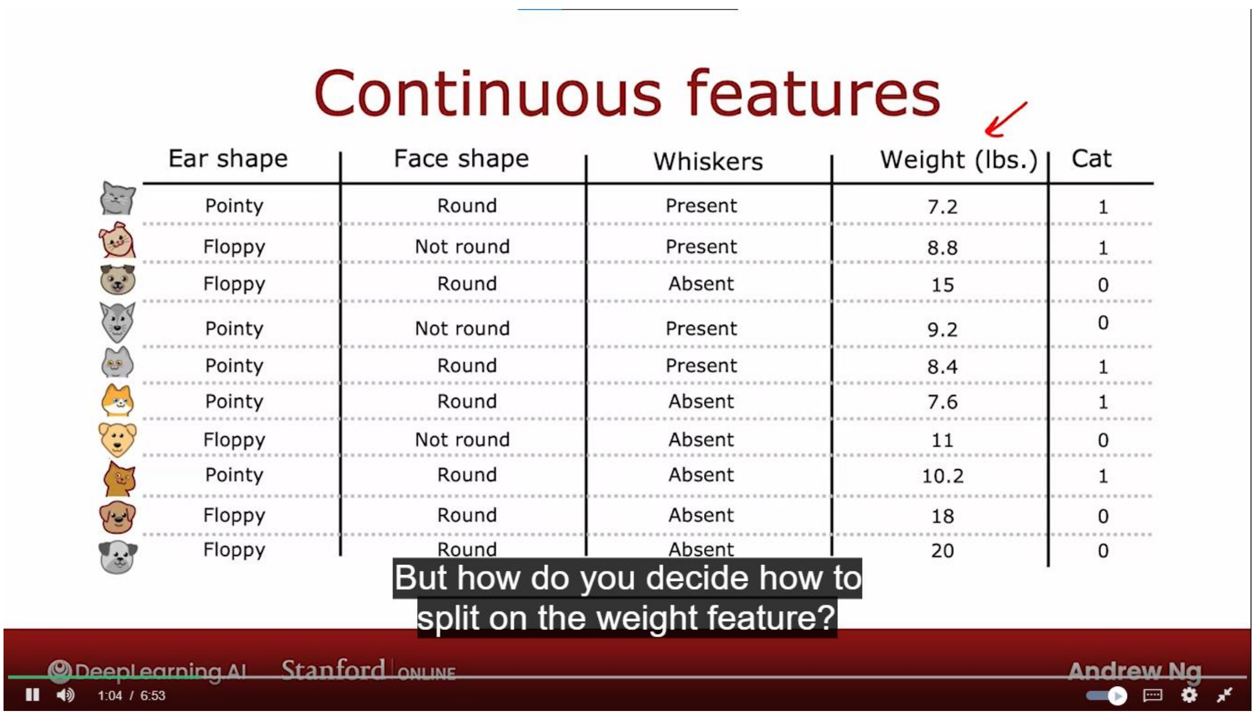
DeepLearning.AI Stanford ONLINE Andrew Ng

8:08 / 9:07



One hot encoding and neural networks

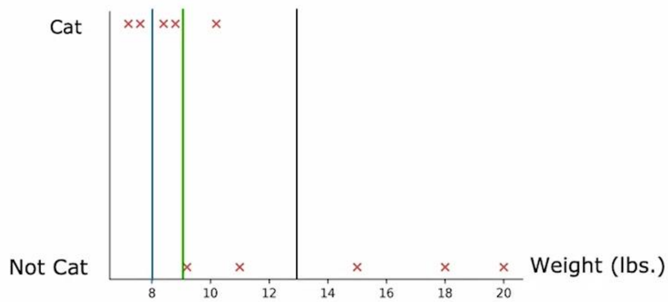
Pointy ears	Floppy ears	Round ears	Face shape	Whiskers	Cat
1	0	0	~~Round~~ 1	~~Present~~ 1	1
0	0	1	~~Not round~~ 0	~~Present~~ 1	1
0	0	1	~~Round~~ 1	~~Absent~~ 0	0
1	0	0	~~Not round~~ 0	~~Present~~ 1	0
0	0	1	~~Round~~ 1	~~Present~~ 1	1
1	0	0	~~Round~~ 1	~~Absent~~ 0	1
0	1	0	~~Not round~~ 0	~~Absent~~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

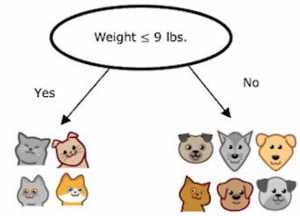


Continuous features

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



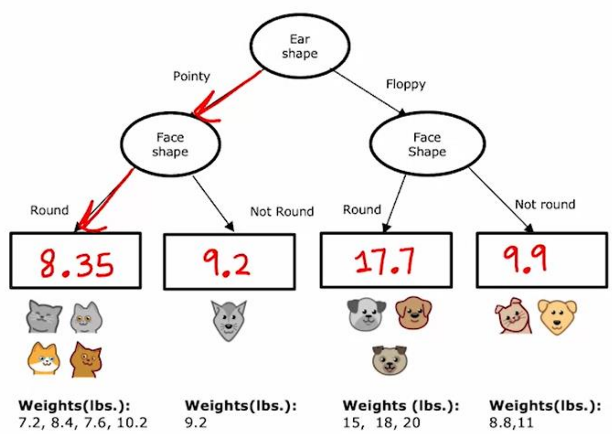


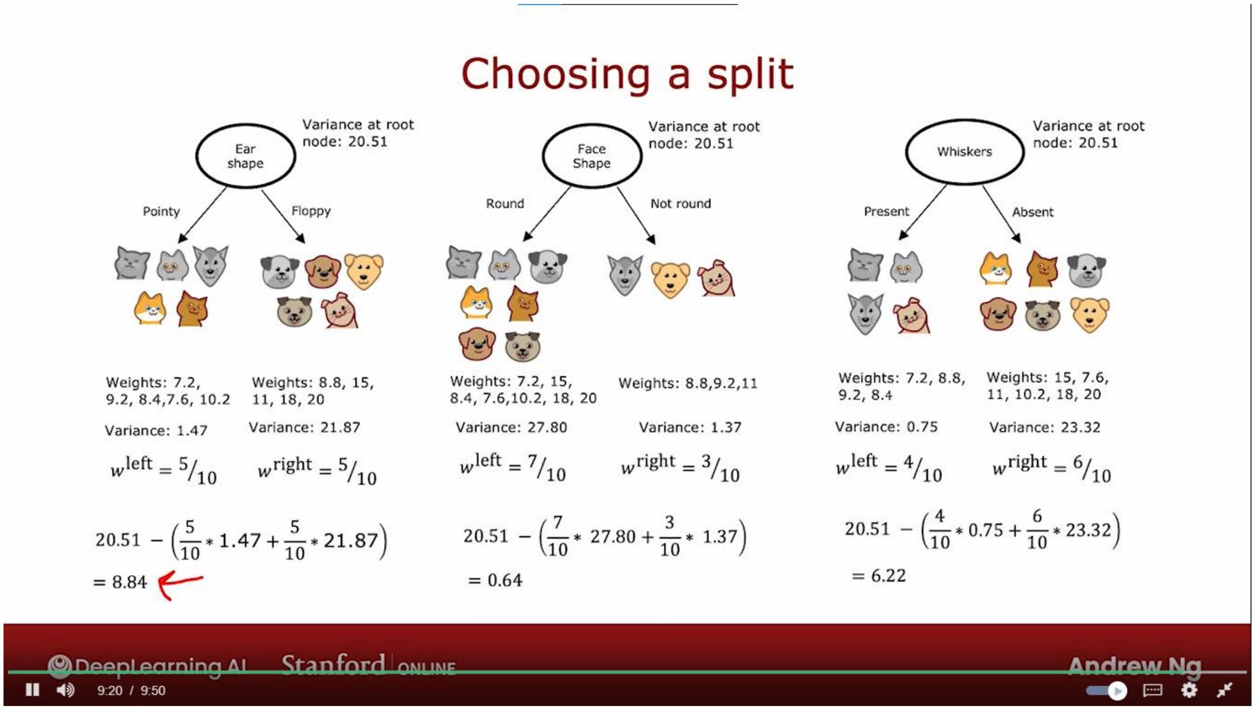
$$H(0.5) - \left(\frac{2}{10}H\left(\frac{2}{2}\right) + \frac{8}{10}H\left(\frac{3}{8}\right)\right) = 0.24$$

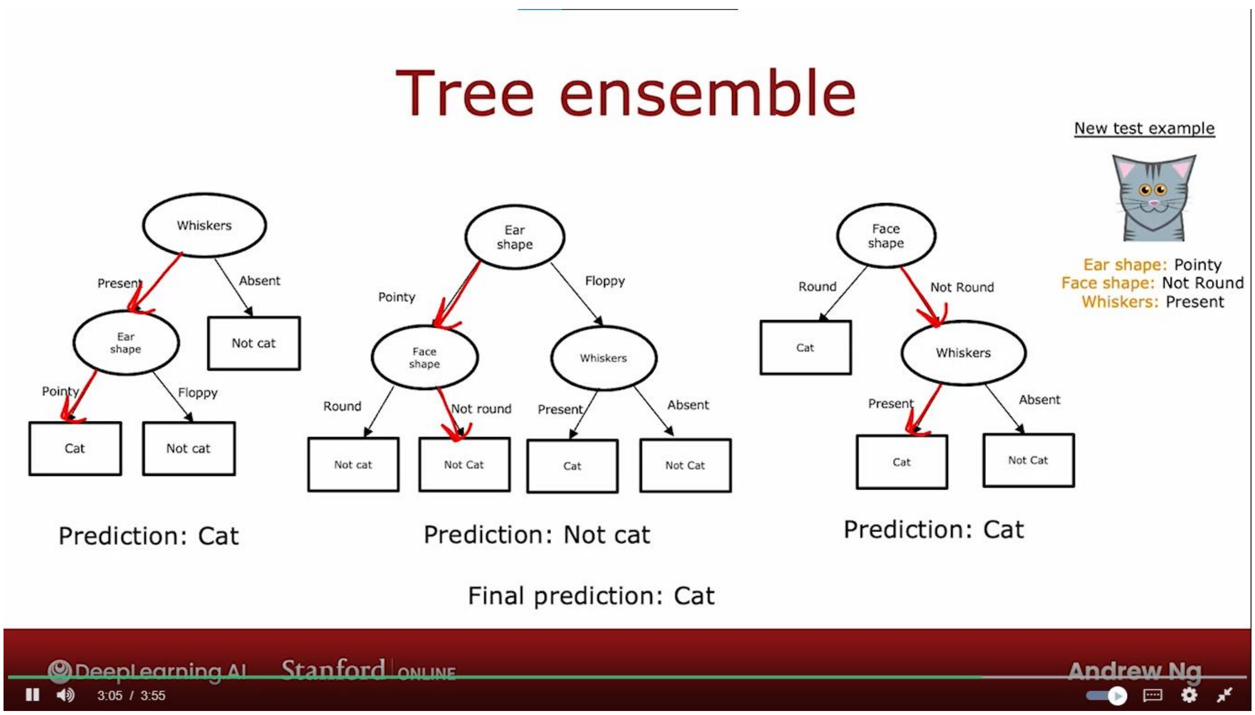
$$H(0.5) - \left(\frac{4}{10}H\left(\frac{4}{4}\right) + \frac{6}{10}H\left(\frac{1}{6}\right)\right) = 0.61$$

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

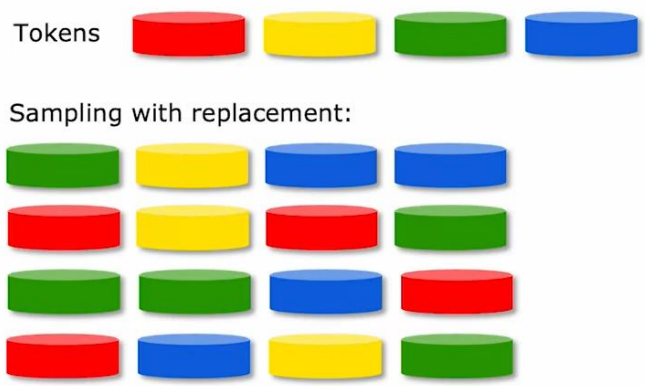
Regression with Decision Trees




Choosing a split
Ear shape
Variance at root node: 20.51
Pointy
Floppy
Weights: 7.2, 9.2, 8.4,7.6, 10.2
Weights: 8.8, 15, 11, 18, 20
Variance: 1.47
Variance: 21.87
$w^{left} = 5/10$
$w^{right} = 5/10$
$20.51 - \left(\frac{5}{10} * 1.47 + \frac{5}{10} * 21.87\right)$
$= 8.84$
Face Shape
Variance at root node: 20.51
Round
Not round
Weights: 7.2, 15, 8.4, 7.6,10.2, 18, 20
Weights: 8.8,9.2,11
Variance: 27.80
Variance: 1.37
$w^{left} = 7/10$
$w^{right} = 3/10$
$20.51 - \left(\frac{7}{10} * 27.80 + \frac{3}{10} * 1.37\right)$
$= 0.64$
Whiskers
Variance at root node: 20.51
Present
Absent
Weights: 7.2, 8.8, 9.2, 8.4
Weights: 15, 7.6, 11, 10.2, 18, 20
Variance: 0.75
Variance: 23.32
$w^{left} = 4/10$
$w^{right} = 6/10$
$20.51 - \left(\frac{4}{10} * 0.75 + \frac{6}{10} * 23.32\right)$
$= 6.22$
DeepLearning.AI
Stanford ONLINE
Andrew Ng
9:20 / 9:50


Tree ensemble
New test example
Ear shape: Pointy
Face shape: Not Round
Whiskers: Present
Whiskers
Present
Absent
Ear shape
Not cat
Pointy
Floppy
Cat
Not cat
Prediction: Cat
Ear shape
Pointy
Floppy
Face shape
Whiskers
Round
Not round
Present
Absent
Not cat
Not Cat
Cat
Not Cat
Prediction: Not cat
Face shape
Round
Not Round
Cat
Whiskers
Present
Absent
Cat
Not Cat
Prediction: Cat
Final prediction: Cat
DeepLearning.AI
Stanford ONLINE
Andrew Ng
3:05 / 3:55

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

Generating a tree sample

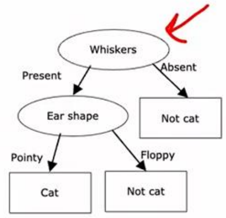
Given training set of size m

For $b = 1$ to B:

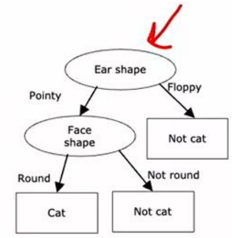
Use sampling with replacement to create a new training set of size m

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	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



...

Bagged decision tree

DeepLearning.AI Stanford ONLINE Andrew Ng

3:28 / 6:22

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.

$K = \sqrt{n}$

Random forest algorithm

DeepLearning.AI Stanford ONLINE Andrew Ng

5:31 / 6:22

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

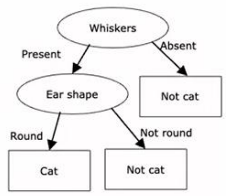
But instead of picking from all examples with equal (1/m) probability, make it more likely to pick misclassified examples from previously trained trees

Train a decision tree on the new dataset

$\underbrace{1, 2, \ldots, b-1}$

b

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	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)
```


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