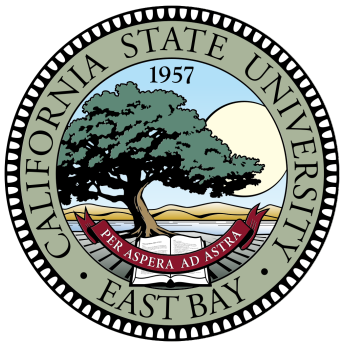


CS 6820 – Machine Learning

Lecture 9

Instructor: Eric S. Gayles, PhD.

Feb 12, 2018

Preliminaries

Total Probability

Marginalization and Law of Total Probability

- Marginalization (Sum Rule)

$$p(x) = \sum_y p(x, y)$$

- Law of Total Probability

$$p(x) = \sum_y p(x \mid y) \cdot p(y)$$

Expected Value

- If X is a random variable that takes value v_i with a probability of $p(v_i)$ and has a sample space of size n, then the expected value is:

$$E(X) = \sum_n v_i * p(v_i)$$

Bernoulli and Binomial Distributions

- We have a binary random variable $x \in \{0,1\}$ (i.e. dom$(x) = \{0,1\}$)
  The *Bernoulli* distribution is parameterized by a single scalar μ,

$$P(x=1 \,|\, \mu) = \mu \,, \quad P(x=0 \,|\, \mu) = 1 - \mu$$
$$\text{Bern}(x \,|\, \mu) = \mu^x (1-\mu)^{1-x}$$

- We have a data set of random variables $D = \{x_1, .., x_n\}$, each $x_i \in \{0,1\}$. If each $x_i \sim \text{Bern}(x_i \,|\, \mu)$ we have

$$P(D \,|\, \mu) = \prod_{i=1}^n \text{Bern}(x_i \,|\, \mu) = \prod_{i=1}^n \mu^{x_i} (1-\mu)^{1-x_i}$$

$$\operatorname*{argmax}_\mu \; \log P(D \,|\, \mu) = \operatorname*{argmax}_\mu \; \sum_{i=1}^n x_i \log \mu + (1-x_i) \log(1-\mu) = \frac{1}{n} \sum_{i=1}^n x_i$$

- The *Binomial distribution* is the distribution over the count $m = \sum_{i=1}^n x_i$

$$\text{Bin}(m \,|\, n, \mu) = \binom{n}{m} \mu^m (1-\mu)^{n-m} \,, \quad \binom{n}{m} = \frac{n!}{(n-m)!\, m!}$$

Marc Toussaint
University of Stuttgart

Multinomial Distribution

- We have an integer random variable $x \in \{1, .., K\}$
  The probability of a single x can be parameterized by $\mu = (\mu_1, .., \mu_K)$:

$$P(x=k \,|\, \mu) = \mu_k$$

  with the constraint $\sum_{k=1}^K \mu_k = 1$ (probabilities need to be normalized)
- We have a data set of random variables $D = \{x_1, .., x_n\}$, each $x_i \in \{1, .., K\}$. If each $x_i \sim P(x_i \,|\, \mu)$ we have

$$P(D \,|\, \mu) = \prod_{i=1}^n \mu_{x_i} = \prod_{i=1}^n \prod_{k=1}^K \mu_k^{[x_i=k]} = \prod_{k=1}^K \mu_k^{m_k}$$

  where $m_k = \sum_{i=1}^n [x_i=k]$ is the count of $[x_i=k]$. The ML estimator is

$$\operatorname*{argmax}_\mu \; \log P(D \,|\, \mu) = \frac{1}{n}(m_1, .., m_K)$$

- The *Multinomial distribution* is this distribution over the counts m_k

$$\text{Mult}(m_1, .., m_K \,|\, n, \mu) \propto \prod_{k=1}^K \mu_k^{m_k}$$

Marc Toussaint
University of Stuttgart

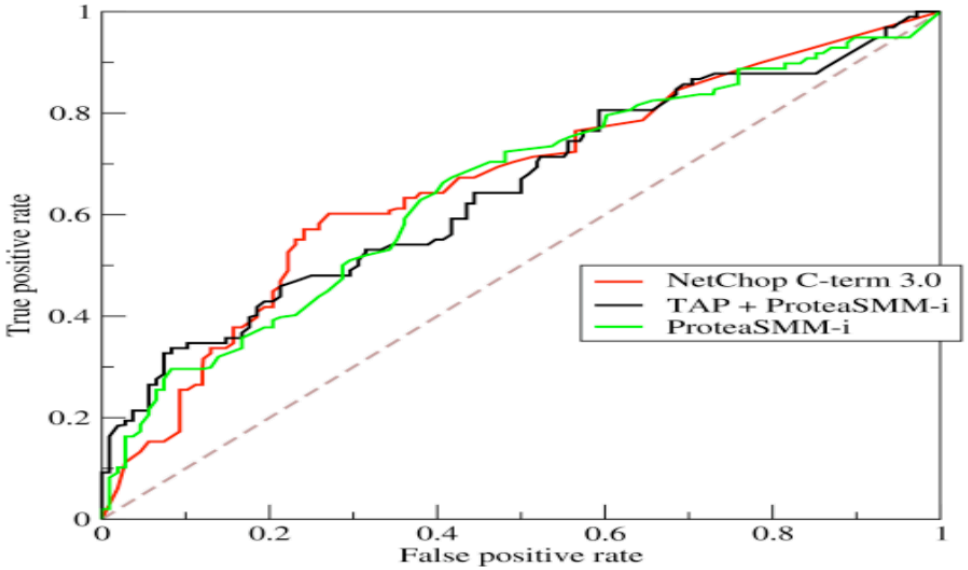
Classification Measures – F Measure

- Comparing different approaches is difficult when using two evaluation measures (e.g. Recall and Precision)
- F-measure combines recall and precision into a single measure:

$$f_\beta = \left(1 + \beta^2\right) \frac{P \cdot R}{\left(\beta^2 \cdot P\right) + R}$$

Classification Measures – ROC Curves

Receiver Operator Characteristic (ROC) curves plot **true positive** rates against **false positive** rates



- Can be achieved by e.g. varying decision threshold of a classifier
- **Area under the curve** is an often used measure of goodness
- Two-forced alternative choice (**2AFC**) score is an easy to compute approximation of the area under the ROC curve

Moving on …

Decision Trees

- Decision trees are powerful and popular tools for classification and prediction.

- Decision trees represent intuitive rules that can be used in knowledge systems such as databases and ML solutions.

Decision Trees

- The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

Decision Trees

- **Classification:** Decision trees classify instances or examples by starting at the root of the tree and moving through it until a leaf node.
- **Decision node:** specifies a test on a single attribute
- **Leaf node:** indicates the value of the target attribute
- **Arc/edge:** split of one attribute
- **Path:** a disjunction of tests to make the final decision

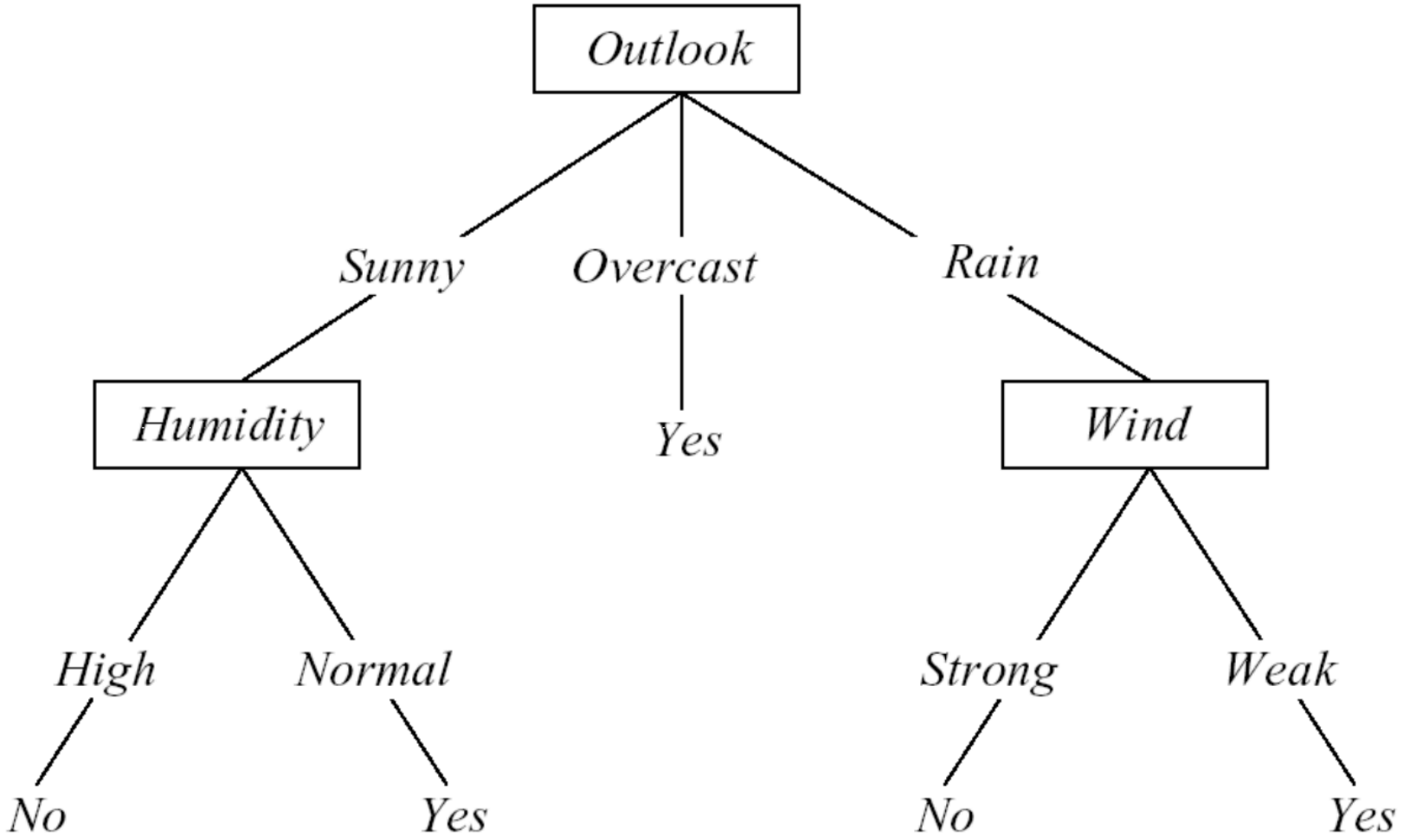
Decision Trees - Advantages

- Simple to understand and to interpret.
- Trees can be visualized.
- The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.
- Able to handle both numerical and categorical data.
- Able to handle multi-output problems.
- Easy to explain the results
- Possible to validate a model using statistical tests. That makes it possible to assess the reliability of the model.

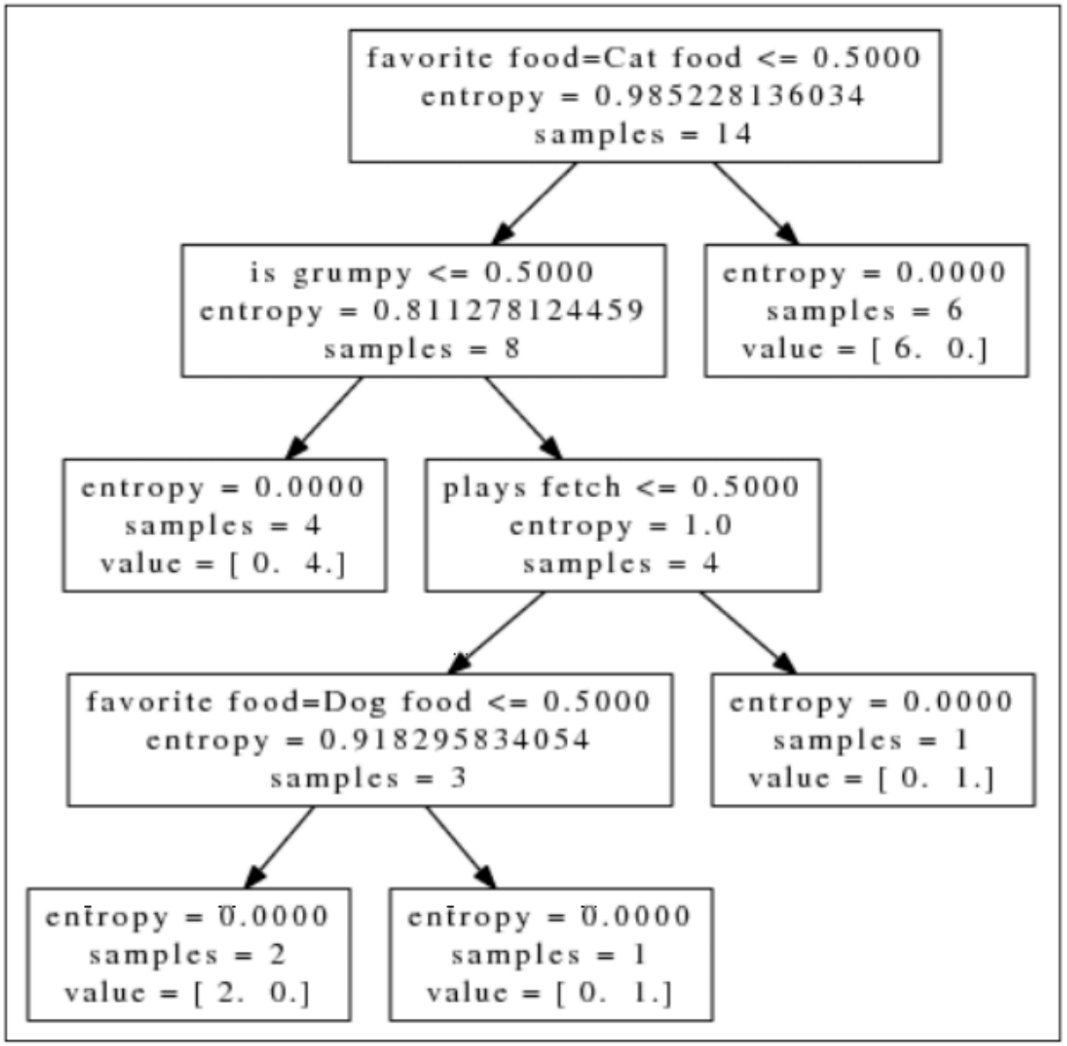
Decision Trees - Disadvantages

- Decision-tree learners can create overly-complex trees that do not generalize the data well – essentially overfitting.
- Decision trees can be unstable because small variations in the data might result in a completely different tree being generated.
- The problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts.
  - Consequently, practical decision-tree learning algorithms are based on heuristic algorithms such as the greedy algorithm where locally optimal decisions are made at each node.
  - Such algorithms cannot guarantee to return the globally optimal decision tree.
  - This can be mitigated by training multiple trees in an ensemble learner, where the features and samples are randomly sampled with replacement.
- Decision tree learners can create biased trees if some classes dominate.

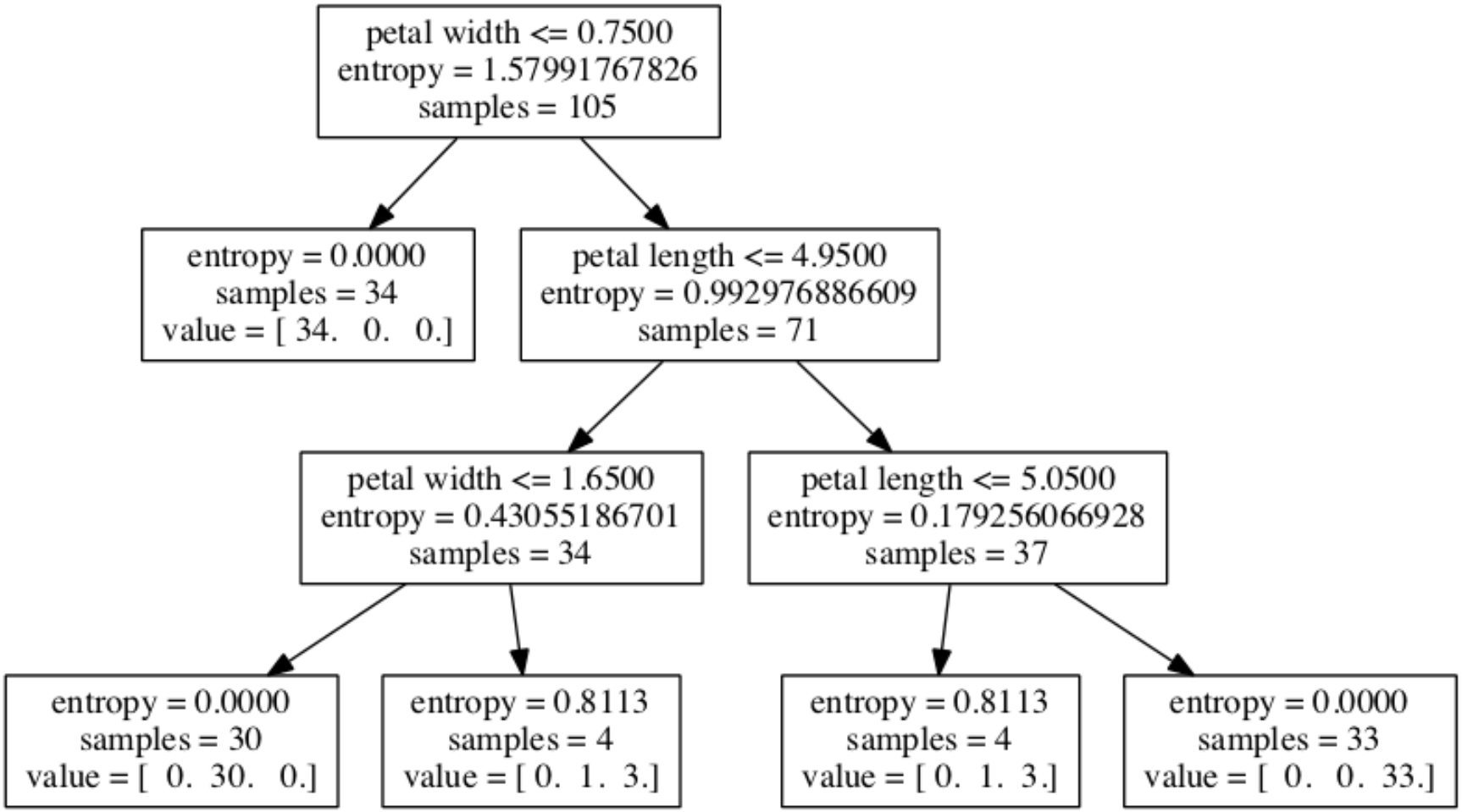
Decision Trees



Decision Trees



Decision Trees



Decision Trees

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - Examples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)

- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left

Decision Trees

- Main loop:
  1. $A \leftarrow$ the "best" decision attribute for next *node*
  2. Assign A as decision attribute for *node*
  3. For each value of A, create new descendant of *node*
  4. Sort training examples to leaf nodes
  5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

Decision Trees

- **USAGE:** To classify new animals, the decision tree examines an explanatory variable at each node.
- For example, the first node might ask whether or not the animal likes to play fetch.
- If the animal does, we will follow the edge to the left child node;
- if not, we will follow the edge to the right child node.
- Eventually an edge will connect to a leaf node that indicates whether the animal is a cat or a dog.

Training instance	Plays fetch	Is grumpy	Favorite food	Species
1	Yes	No	Bacon	Dog
2	No	Yes	Dog Food	Dog
3	No	Yes	Cat food	Cat
4	No	Yes	Bacon	Cat
5	No	No	Cat food	Cat
6	No	Yes	Bacon	Cat
7	No	Yes	Cat Food	Cat
8	No	No	Dog Food	Dog
9	No	Yes	Cat food	Cat
10	Yes	No	Dog Food	Dog
11	Yes	No	Bacon	Dog
12	No	No	Cat food	Cat
13	Yes	Yes	Cat food	Cat
14	Yes	Yes	Bacon	Dog

Decision Trees

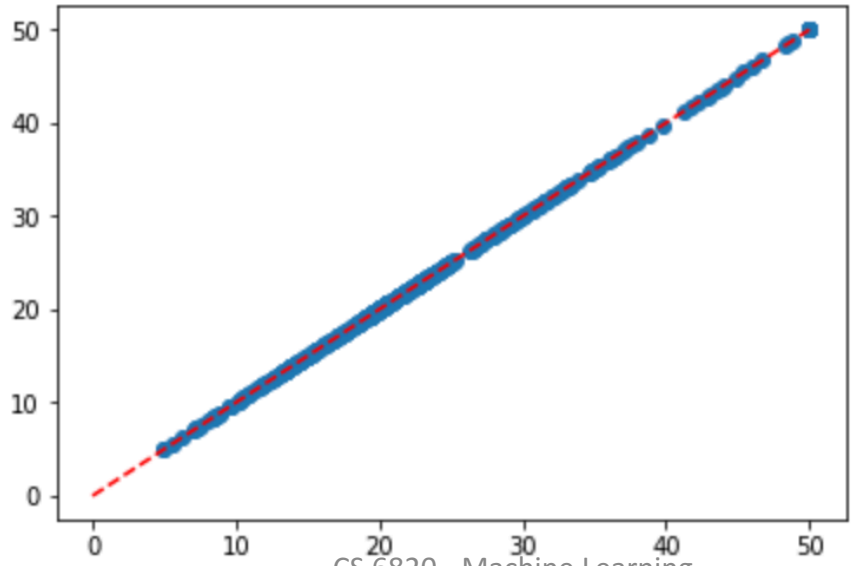
- From this data we can see that cats are generally grumpier than the dogs.
- Most dogs play fetch and most cats refuse.
- Dogs prefer dog food and bacon, whereas cats only like cat food and bacon.
- The “is grumpy” and “plays fetch” explanatory variables can be easily converted to binary-valued features.
- The “favorite food” explanatory variable is a categorical variable that has three possible values; we can use one-hot encoding to represent.

Sample – Decision Tree Classifier

In [3]:
```
from sklearn.datasets import load_boston
from sklearn.tree import DecisionTreeRegressor
import matplotlib.pyplot as plt

data = load_boston()
clf = DecisionTreeRegressor().fit(data.data, data.target)
predicted = clf.predict(data.data)
expected = data.target

plt.scatter(expected, predicted)
plt.plot([0, 50], [0, 50], '--r')
plt.show()
```



Decision Trees

- **Selection of an attribute** to test at each node: choosing the most useful attribute for classifying examples.

- **Information gain:** measures how well a given attribute separates the training examples according to their target classification

  – This measure is used to select among the candidate attributes at each step while growing the tree

Decision Trees

- **Information Gain** measures the expected reduction in entropy, or uncertainty.
- *Set* S is the collection of objects
- $A()$ maps elements s in the set S to the set of all possible values for attribute A
- S_v the subset of S for which attribute A has value v
  - $S_v = \{s \text{ in } S \mid A(s) = v\}$
- Information Gain compares the entropy of the original collection with the expected value of the entropy after S is partitioned using attribute A

Let’s Talk About Information Theory

Information Theory

- Average amount of information provided per event
- Average amount of surprise when observing a event
- Uncertainty an observer has before seeing the event
- Average number of bits needed to communicate each event

Information Theory

- **Information theory provides a mathematical framework for the quantification, compression and transmission of information.**
- We are going to talk about information as a decrease in uncertainty.
- We have some uncertainty about a process, e.g., which symbol (A, B, C) will be generated from a device.
- We learn a piece of information, e.g., that the previous symbol is B.
- How uncertain are we now about which symbol will appear?
- The more informative our information is, the less uncertain we will be.
- Entropy is a measure of unpredictability of information content.

Information Theory

- Named after Boltzmann's H-theorem, Shannon defined the entropy H (Greek letter Eta) of a discrete random variable X with possible values $\{x_1, \ldots, x_n\}$ and probability mass function $P(X)$ as:

$$\mathrm{H}(X) = \sum_i \mathrm{P}(x_i)\,\mathrm{I}(x_i) = -\sum_i \mathrm{P}(x_i)\log_b \mathrm{P}(x_i),$$

Information Theory

- Shannon, in fact, defined entropy as a measure of the average information content associated with a random outcome.
- Shannon's definition of information entropy makes this intuitive distinction mathematically precise.
- His definition satisfies these criteria:
  - The measure should be continuous — i.e., changing the value of one of the probabilities by a very small amount should only change the entropy by a small amount.
  - If all the outcomes are equally likely, then entropy should be maximal. In this case, the entropy increases with the number of outcomes.
  - if the outcome is a certainty, then the entropy should be zero.
  - The amount of entropy should be the same independently of how the process is divided into parts.

Information Theory

- Bit is a unit of information.

- 1 bit refers to the amount of information that one is uncertain about in a binary random variable that takes the value of either 0 or 1 with equal probability.

Information Theory

- Let's assume that we have a device that emits one symbol (A).
  - We have no uncertainty as to what we will see: uncertainty is zero.

- A device that emits two symbols (A, B).
  - We have one choice, either A or B: our uncertainty is one.

  - We could use one bit (0 or 1) to encode the outcome

- A device that emits four symbols (A, B, C, D).
  - If we made a decision tree, there would be two levels (starting with "Is it A/B or C/D?"):
    - Uncertainty is two.
    - We would need two bits (00, 01, 10, 11) to encode the outcome

  - What we are describing is a logarithm (base 2): log2 M, where M is the number of symbols

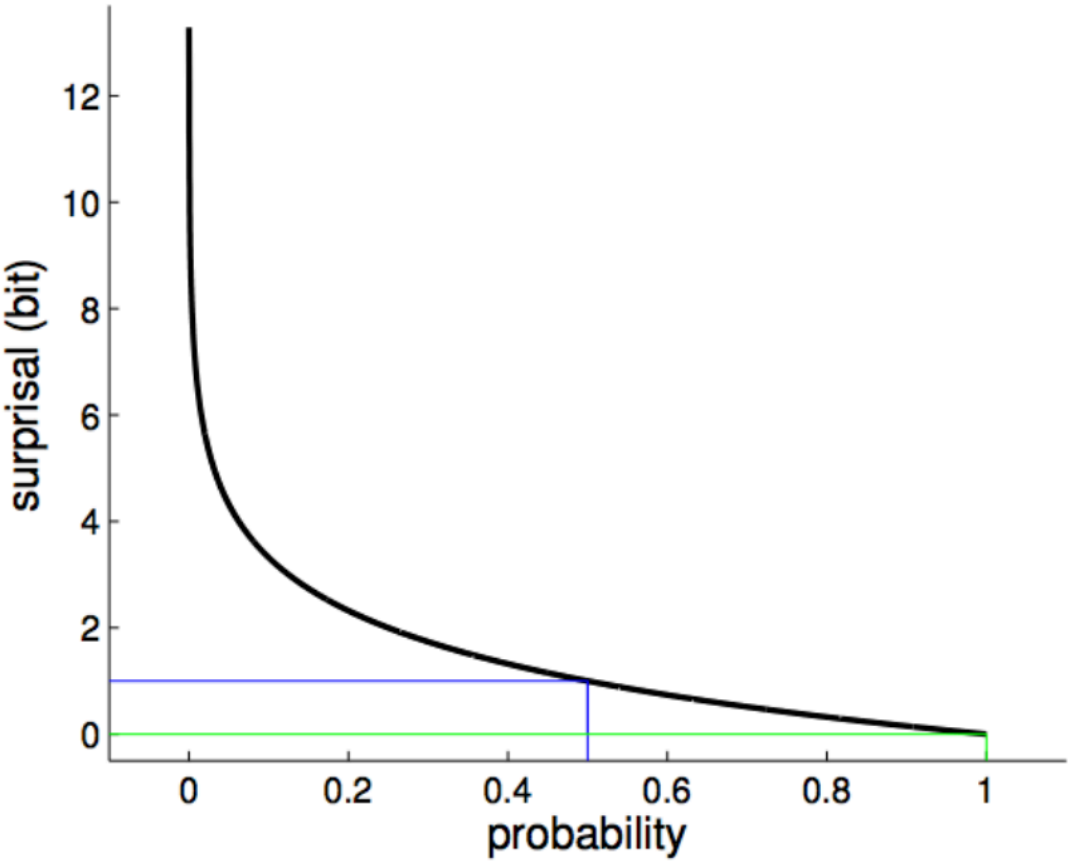
Information Theory

- An event with zero probability would have an infinite level of **SUPRISAL**, because one would be maximally uncertain about the outcome of this event (and consequentially, one would be completely surprised).

- An event with probability of 0.5, e.g. a fair coin toss, would have a surprisal value of 1 bit.

- A sure event would have a surprisal of 0, because (intuitively) the outcome would always be within one's expectation, meaning no surprise.

Information Theory



Information Theory

- Surprisal $s()$ quantifies the uncertainty in a random variable X taking a certain value x based on its probability of occurrence $p(X = x)$ or $p(x)$.

- Surprisal is measured in bits when the base of the logarithm is 2.

$$s(x) = \log_2 \frac{1}{p(x)} = -\log_2 p(x).$$

Information Theory

- Let X be a random variable with distribution $p(X)$.
- We want to quantify the information provided by each possible outcome.
- Specifically we want a function which maps the probability of an event $p(x)$ to the information $I(x)$
- Our metric $I(x)$ should have the following properties:
  - $I(x_i) \geq 0 \quad \forall i.$
  - $I(x_1) > I(x_2)$ if $p(x_1) < p(x_2)$
  - $I(x_1, x_2) = I(x_1) + I(x_2)$

Information Theory

$$I(x) = f(p(x))$$

- We want $f()$ such that $I(x_1, x_2) = I(x_1) + I(x_2)$
- We know $p(x_1, x_2) = p(x_1)p(x_2)$
- The only function with this property is $\log()$: $\log(ab) = \log(a) + \log(b)$
- Hence we define:

$$I(X) = \log(\frac{1}{p(X)})$$

Information Theory

- Suppose we have a sequence of observations of a random variable X.
- A natural question to ask is what is the average amount of information per observation.

$$H(X) = E[I(X)] = E[\log(\frac{1}{p(X)})]$$

Information Theory

But $-\log_2 p(x)$ just tells us how surprising one particular symbol is. On average, though, how surprising is our random variable?

That is where the summation (a weighted average) comes in:

(2) $H(X) = -\sum_{x \in X} p(x) \log_2 p(x) = E(\log_2 \frac{1}{p(X)})$

Sum over all possible outcomes, multiplying the surprisal $(-\log_2 p(x))$ by the probability of seeing that surprising outcome $(p(x))$

- The amount of surprisal is the amount of information we need in order to not be surprised.
  - $H(X) = 0$ if the outcome is certain
  - $H(X) = 1$ if out of 2 outcomes, both are equally likely

*Advanced NLP

Information Theory

- **Putting it all together**

- Suppose we have a sequence of observations of a random variable X.
- A natural question to ask is what is the average amount of information per observation.
- This quantitity is called the *Entropy* and denoted $H(X)$

$$H(X) = E[I(X)] = E[\log(\frac{1}{p(X)})]$$

$$H(X) = \sum_x p(x)s(x) = \sum_x p(x)\log_2\frac{1}{p(x)} = -\sum_x p(x)\log_2 p(x).$$

Information Theory

- The entropy of the variable $\boldsymbol{X}$ is the sum, over all possible outcomes $\boldsymbol{x_i}$ of $\boldsymbol{X}$, of the product of the probability of outcome $\boldsymbol{x_i}$ times the log of the inverse of the probability of $\boldsymbol{x_i}$ (which is also called $\boldsymbol{x_i}$'s surprisal) - the entropy of $\boldsymbol{X}$ is the expected value of its outcome's surprisal.

- We can also apply this to a general probability distribution, rather than a just discrete-valued event.

Information Theory

- A single random variable X has a *Marginal Distribution*

$$p(X)$$

- This distribution has an associated *Marginal Entropy*

$$H(X) = \sum_i p(x_i) \log \frac{1}{p(x_i)}$$

- Marginal entropy is the average information provided by observing a variable X

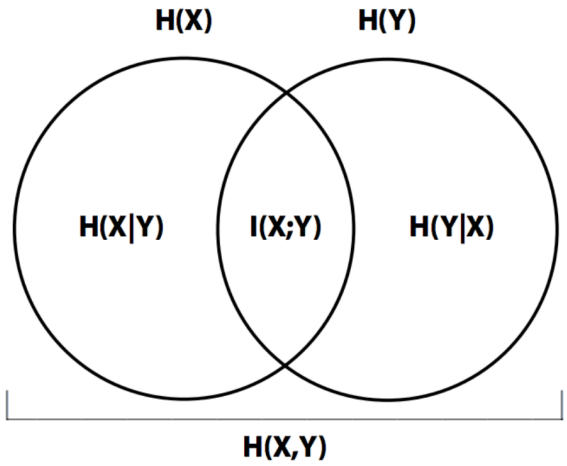
Information Theory

- **Source Coding Theorem** - It is impossible to compress an input variable (or a data source) at a rate less than its entropy without any loss of information.

Information Theory

- **Joint Entropy** - Joint entropy of two discrete variables X and Y is the total amount of uncertainty in the outcomes of these events:
  - Note: Integrate for the case of Continuous Variables

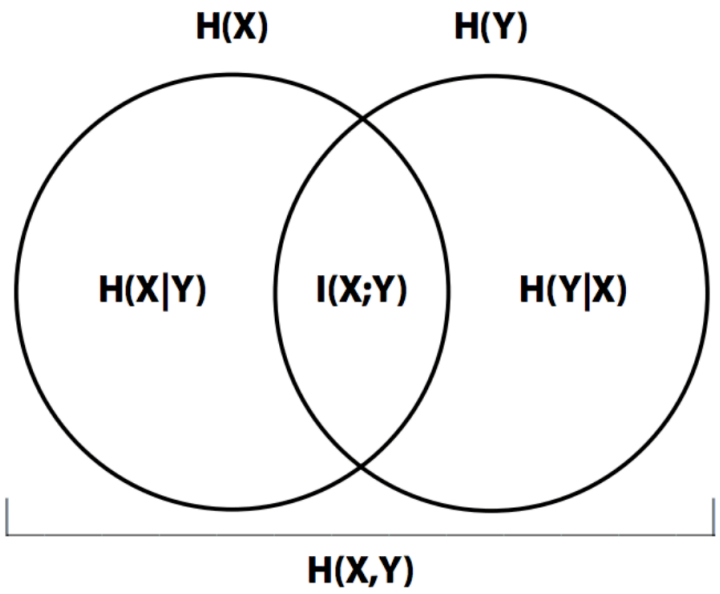
$$H(X,Y) = -\sum_{x,y} p(x,y) \log_2 p(x,y).$$



The area contained by both circles is the Joint Entropy *H(X,Y).*

Information Theory

- **Conditional entropy** - Conditional entropy $H(X|Y)$ is the amount of uncertainty in X given what one knows about Y, and vice versa for $H(Y|X)$.



$$H(X|Y) = -\sum_{x,y} p(x,y) \log_2 p(x|y);$$

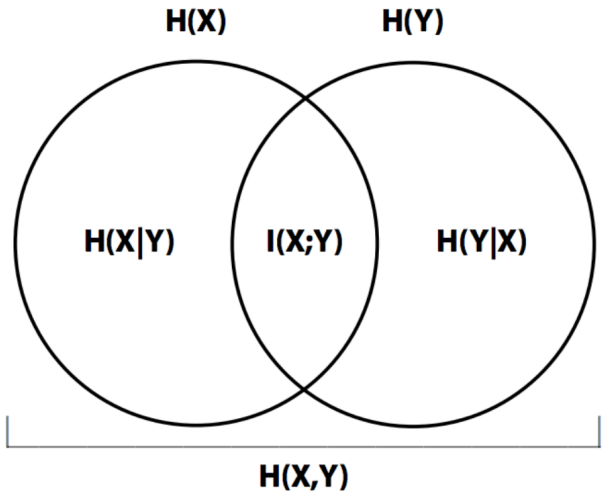
$$H(Y|X) = -\sum_{x,y} p(x,y) \log_2 p(y|x).$$

Pictorially, $H(X|Y)$ is equivalent to the area occupied by $H(X,Y)$ (i.e. joint entropy of X and Y) with the area under $H(Y)$ excluded in Figure 2, and similarly, $H(Y|X)$ is equivalent to the area under $H(X,Y)$ with the area under $H(X)$ subtracted.

Information Theory

- Pictorially, $H(X|Y)$ is equivalent to the area occupied by $H(X,Y)$ (i.e. joint entropy of X and Y) with the area under $H(Y)$ excluded in Figure 2, and similarly, $H(Y|X)$ is equivalent to the area under $H(X,Y)$ with the area under $H(X)$ subtracted.

- Thus, the conditional entropies can be also formulated in terms of the joint and individual entropies:

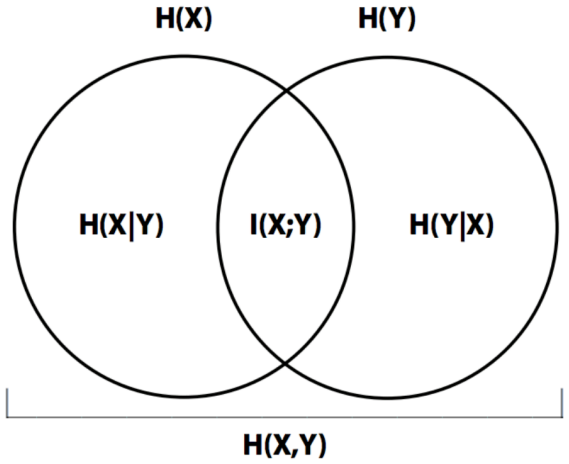


$$H(X|Y) = H(X,Y) - H(Y);$$
$$H(Y|X) = H(X,Y) - H(X).$$

Information Theory

- **Mutual Information** - Mutual Information $I(X;Y)$ between two variables X and Y quantifies the amount of information that is "shared" between the variables, or the degree of dependence between two variables. Pictorially, MI is the area where $H(X)$ and $H(Y)$ overlap in the figure.

- When two variables are independent, their mutual information is 0 since $p(x,y) = p(x)p(y)$.



$$
\begin{aligned}
I(X;Y) &= H(X) - H(X|Y) = H(Y) - H(Y|X) \\
&= H(X) + H(Y) - H(X,Y) = H(X,Y) - H(X|Y) - H(Y|X) \\
&= \sum_{x,y} p(x,y) \log_2 \frac{p(x,y)}{p(x)p(y)}.
\end{aligned}
$$

Information Theory

$$H(s) = \sum_{k=1}^{M} p_k \log_2 \frac{1}{p_k}$$

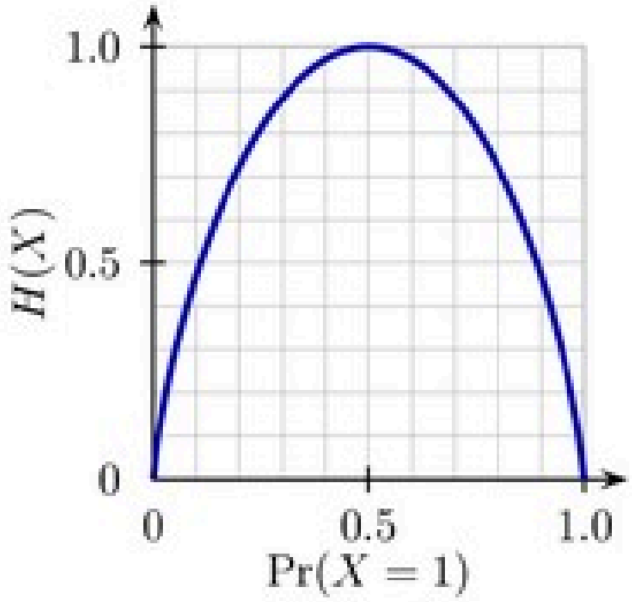
$$= M . \frac{1}{M} \log_2 \frac{1}{1/M}$$

$$= \log_2 M$$

- **This is maximum value of entropy and thus it is maximum when all symbols have equal probability of occurrence**

Entropy Intuition

- The entropy is 0 if the outcome is ``certain''.
- The entropy is maximum if we have no knowledge of the system (or any outcome is equally possible).



Entropy of a 2-class problem with regard to the portion of one of the two groups

Back to Decision Trees

Decision Trees

- We will measure the **REDUCTION IN ENTROPY** using a metric called **Information Gain**.
- **Calculated with the following equation, information gain is the difference between the entropy of the parent node, *H(T)*, and the weighted average of the children nodes' entropies.**
- *T* is the set of instances, and *a* is the explanatory variable under test.
- $x_a \in vals(a)$ is the value of attribute a for instance *x*.
- $\{x \in T | x_a = v\}$ is the number of instances for which attribute *a* is equal to the value *v*.
- $H(\{ x \in T | x_a = v\})$ is the entropy of the subset of instances for which the value of the explanatory variable *a* is *v*.

$$IG(T,a) = H(T) - \sum_{v \in vals(a)} \frac{\left|\{x \in T \mid x_a = v\}\right|}{|T|} H\left(\{x \in T \mid x_a = v\}\right)$$

Decision Trees

```
INPUT: S, where S = set of classified instances
OUTPUT: Decision Tree
Require: S ≠ ∅, num_attributes > 0
 1: procedure BUILDTREE
 2:     repeat
 3:         maxGain ← 0
 4:         splitA ← null
 5:         e ← Entropy(Attributes)
 6:         for all Attributes a in S do
 7:             gain ← InformationGain(a, e)
 8:             if gain > maxGain then
 9:                 maxGain ← gain
10:                 splitA ← a
11:             end if
12:         end for
13:         Partition(S, splitA)
14:     until all partitions processed
15: end procedure
```

Decision Trees

The following table contains the information gains for all of the tests. In this case, the cat food test is still the best, as it increases the information gain the most.

Test	Parent's entropy	Child's entropy	Child's entropy	Weighted average	IG
plays fetch?	0.9852	0.7642	0.7219	0.7490 * 9/14 + 0.7219 * 5/14 = 0.7491	0.2361
is grumpy?	0.9852	0.9183	0.8113	0.9183 * 6/14 + 0.8113 * 8/14 = 0.85710.8572	0.1280
favorite food = cat food	0.9852	0.8113	0	0.8113 * 8 /14 + 0.0 * 6/14 = 0.4636	0.5216
favorite food = dog food	0.9852	0.8454	0	0.8454 * 11/14 + 0.0 * 3/14 = 0.6642	0.3210
favorite food = bacon	0.9852	0.9183	0.971	0.9183 * 9/14 + 0.9710 * 5/14 = 0.9371	0.0481

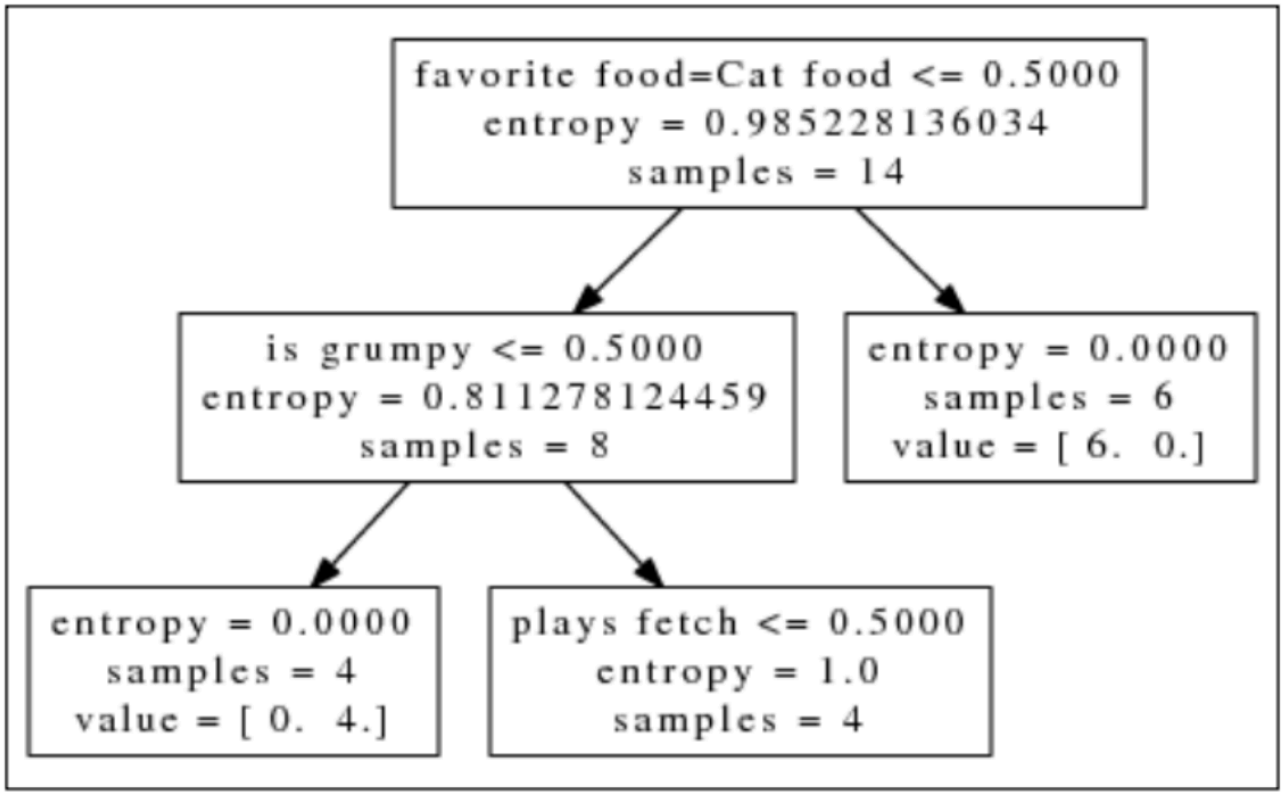
Decision Trees

Now let's add another node to the tree. One of the child nodes produced by the test is a leaf node that contains only cats. The other node still contains two cats and six dogs. We will add a test to this node. Which of the remaining explanatory variables reduces our uncertainty the most? The following table contains the information gains for all of the possible tests:

Test	Parent's entropy	Child's entropy	Child's entropy	Weighted average	IG
plays fetch?	0.8113	1	0	1.0 * 4/8 + 0 * 4/8 = 0.5	0.3113
is grumpy?	0.8113	0	1	0.0 * 4/8 + 1 * 4/8 = 0.5	0.3113
favorite food=dog food	0.8113	0.9710	0	0.9710 * 5/8 + 0.0 * 3/8 = 0.6069	0.2044
favorite food=bacon	0.8113	0	0.9710	0.0 * 3/8 + 0.9710 * 5/8 = 0.6069	0.2044

Decision Trees

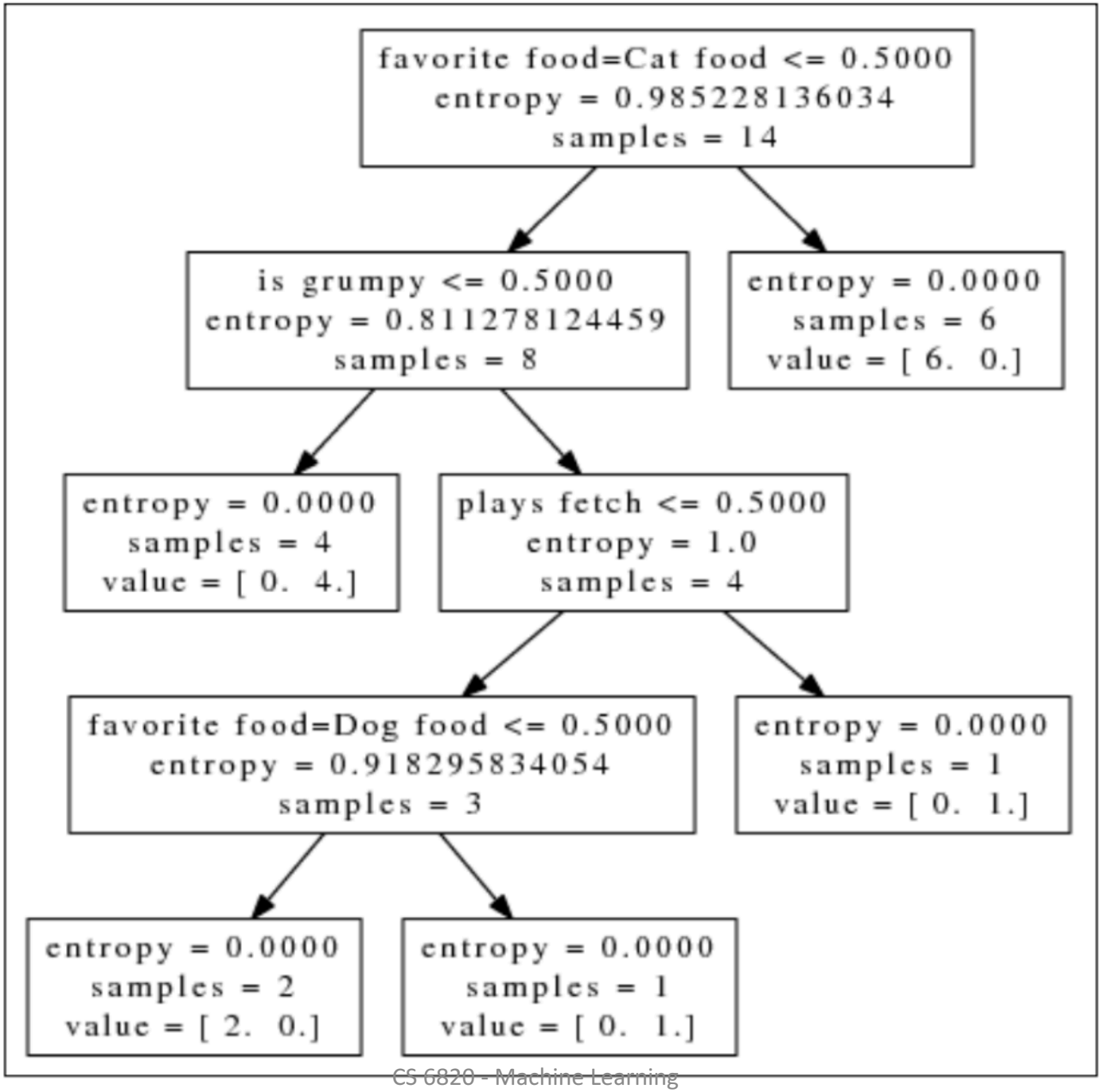
All of the tests produce subsets with 0 bits of entropy, but the `is grumpy` and `plays fetch` tests produce the greatest information gain. ID3 breaks ties by selecting one of the best tests arbitrarily. We will select the `is grumpy` test, which splits its parent's eight instances into a leaf node containing four dogs and a node containing two cats and two dogs. The following is a diagram of the current tree:



Decision Trees

We will now select another explanatory variable to test the child node's four instances. The remaining tests, `favorite food=bacon`, `favorite food=dog food`, and `plays fetch`, all produce a leaf node containing one dog or cat and a node containing the remaining animals. The remaining tests produce equal information gains, as shown in the following table:

Test	Parent's Entropy	Child Entropy	Child Entropy	Weighted Average	Information Gain
plays fetch?	1	0.9183	0	0.688725	0.311275
favorite food=dog food	1	0.9183	0	0.688725	0.311275
favorite food=bacon	1	0	0.9183	0.688725	0.311275


favorite food=Cat food <= 0.5000
entropy = 0.985228136034
samples = 14
is grumpy <= 0.5000
entropy = 0.811278124459
samples = 8
entropy = 0.0000
samples = 6
value = [6. 0.]
entropy = 0.0000
samples = 4
value = [0. 4.]
plays fetch <= 0.5000
entropy = 1.0
samples = 4
favorite food=Dog food <= 0.5000
entropy = 0.918295834054
samples = 3
entropy = 0.0000
samples = 1
value = [0. 1.]
entropy = 0.0000
samples = 2
value = [2. 0.]
entropy = 0.0000
samples = 1
value = [0. 1.]

Decision Trees

Let's classify some animals from the following test data:

Testing instance	Plays fetch	Is grumpy	Favorite food	Species
1	Yes	No	Bacon	Dog
2	Yes	Yes	Dog Food	Dog
3	No	Yes	Dog Food	Cat
4	No	Yes	Bacon	Cat
5	No	No	Cat food	Cat

Decision Trees

- You've constructed a decision tree using the ID3 algorithm.
- Other algorithms can be used to train decision trees.
- C4.5 is a modified version of ID3 that can be used with continuous explanatory variables and can accommodate missing values for features.
- C4.5 also can prune trees.
- Pruning reduces the size of a tree by replacing branches that classify few instances with leaf nodes.

Decision Trees

- We built a decision tree by creating nodes that produced the greatest information gain.

- Another common heuristic for learning decision trees is Gini impurity, which measures the proportions of classes in a set.

- Gini impurity is given by the following equation:
- *j* is the number of classes
- ***t*** is the subset of instances for the node
- ***P(i | t)*** is the probability of selecting an element of class *i* from the node's subset:

$$Gini(t) = 1 - \sum_{i=1}^{j} P(i \mid t)^2$$

Decision Trees

- Intuitively, Gini impurity is zero when all of the elements of the set are the same class, as the probability of selecting an element of that class is equal to one.

- Like entropy, Gini impurity is greatest when each class has an equal probability of being selected.

- The maximum value of Gini impurity depends on the number of possible classes, and it is given by the following equation:

$$Gini_{\max} = 1 - \frac{1}{n}$$

Impurity measures