Discussion 8 **Alex Francis**

CS 189

02/21/2017

Email: afrancis@berkeley.edu
OH: Wednesday 10:00am - 12:00pm (283E Soda)

1 Lecture on Information Theory

From Wikipedia: "Information theory studies the quantification, storage, and communication of information." The appearance of information theory in this course is rather brief, but information theory is one of the most interesting, applicable, and elegant topics in applied mathematics. The birth of the field is attributed to Claude Shannon, who was famous for his work with a variety of algorithms you may be familiar with, including the *One Time Pad* in cryptography, which he proved was unbreakable for random single-use private keys during his time at Bell Labs in the late 1940s.

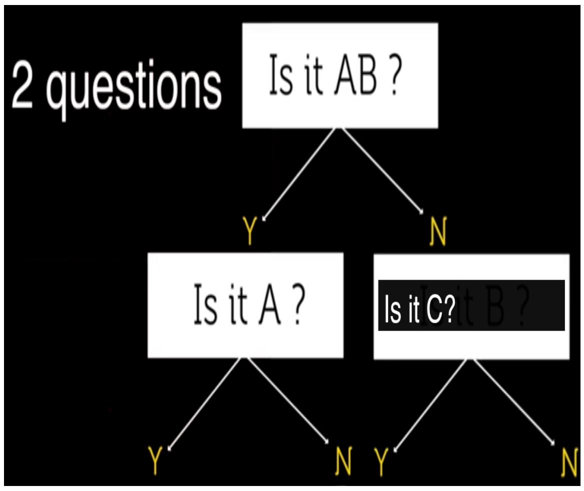
Now, imagine for a moment that we have an alphabet of characters $\mathcal{A}$. To make things more concrete, we imagine for simplicity that this alphabet contains only the first four letters of our English alphabet. More formally, we have that,

$$\mathcal{A} = \{A, B, C, D\}$$

Now, imagine that a machine generates a specific letter from this alphabet according to following probability distribution.[1]

$$P_{\mathcal{A}} = \{1/4, 1/4, 1/4, 1/4\}$$

Our goal is to gain certainty about the next letter in the sequence. Consider the following game. Let's say I am allowed to propose Yes-No questions in order to determine with certainty the form of the next letter. My goal is to minimize the number of questions I ask. It turns out that the optimal strategy in this game is to split half of the probability into one case and the other half into the other case (see the figure below).



The expected number of guesses is,

$$\mathbb{E}(\#\text{guesses}) = \sum_{i=1}^{|\mathcal{A}|} p_i(\#\text{guesses}_i)$$

[1]From this point on, I steal a fair amount of material from a terrific video here: `https://www.youtube.com/watch?v=R4OlXb9aTvQ&t=4s`

In this case, it is easy to see that the expected number of guesses is two, since the number of guesses is two for every letter in the alphabet. Now, consider the general case. It is obvious that $(\#\text{guesses}_i)$ is related to the depth at which full information is available for a particular letter in the alphabet, which is equal to $\log_2(\#\text{nodes at } \ell)$, where ℓ is the level at which full information is achieved. Under the optimal strategy, it can be shown that $(\#\text{nodes at } \ell_i) = 1/p_i$. Therefore, the expected number of guesses, or the number of bits that must be sent to guarantee the correct guess, is,

$$\mathbb{E}(\#\text{guesses}) = \sum_{i=1}^{|\mathcal{A}|} p_i \log_2(1/p_i) = -\sum_{i=1}^{|\mathcal{A}|} p_i \log_2(p_i)$$

This is less obvious for probabilities that are not of the form $1/2^k$, but it can be shown that an equivalence exists between the two cases.

This forms a measure of "uncertainty," which Shannon called *entropy.* For a more rigorous proof, please see Noah's notes.

2 Lecture on Decision Trees

Recall our initial discussion of decision trees Let's consider the pseudocode for the algorithm that was presented in lecture, to make the discussion last week more concrete. *Note that this*

Algorithm 1 Decision Trees

```
1: procedure GrowTree($S$)
2:     if $y_i = C$ for all $i \in S$ and some class $C$ or other stopping condition then
3:         return new leaf(C)
4:     else
5:         choose best splitting feature $j$ and splitting value $\beta$
6:         $S_l = \{i : X_{ij} < \beta\}$
7:         $S_r = \{j : X_{ij} \geq \beta\}$
8:         return new node($j, \beta$, GrowTree($S_l$), GrowTree($S_r$))
```

algorithm can also be adopted for use in a regression context. In the regression context, the stopping conditions are different than in the classification context. This is one of our next areas of study.

There are two details to iron out in this algorithm, namely that of "best" feature splitting and that of the convergence condition. In both classification and regression, we attempt to try all splits. In *classification* contexts, we measure the average reduction in entropy via the formula,

$$H - \left(\frac{n_1 H_1 + n_2 H_2}{n_1 + n_2}\right)$$

Using entropy makes sense in this context, since via the derivation above, it gives us a measure of the "predictability" of a set of values under a probability distribution. Of course, we desire 0 entropy. In *regression* contexts, we measure the average reduction in *mean square error* via the formula,

$$MSE - \left(\frac{MSE_1 + MSE_2}{n_1 + n_2}\right)$$

Which has an interpretation in terms of regression algorithms learned in previous parts of the course.

The following are considered convergence conditions for the decision tree algorithm. Some are better than others.

1. Next split does not reduce entropy or error enough (bad).
2. Most of node's points have the same class - deals with outliers and overlapping distributions.
3. Node contains few sample points.
4. Node covers tiny volume in space.
5. Depth too great.
6. Use (cross-)validation to compare.
7. (Post-processing step) Prune.

3 Lecture on Ensemble Learning

I will keep this short and simple, since the concept is not particularly complicated. The idea is that in *classification* contexts, we should train T learners, and then return the majority prediction for a particular point in the validation/test set, and in *regression* contexts, we return the mean or median of the predictions for a particular point in the validation/test set. This is justified by the central limit theorem, which states that the mean of independent and identically distributed random variable is normally distributed, with a variance reduction by a factor of n and no additional bias! More formally,

Theorem 1.2.1 **(The Central Limit Theorem)** Suppose that $X_1, \ldots, X_n$ is a finite sequence of independent, identically distributed random variables with common mean $\mathbb{E}(X_i) = \mu$ and finite variance $\sigma^2 = \mathrm{Var}(X_i)$. Then, if we let $S_n = \sum_{i=1}^{n} X_i$, we have that

$$\lim_{n\to\infty} \mathbb{P}\left(\frac{S_n - n\mu}{\sigma\sqrt{n}} \leq c\right) = \Phi(c) = \int_{-\infty}^{c} \frac{e^{-x^2/2}}{\sqrt{2\pi}} dx$$