Introduction

In this assignment, we're going to construct a *classifier system*, that is, a machine learning system that can predict an outcome for a new record by indirect comparison to other records. The specific classifier we'll be building is a *Naive Bayes* classifier, that is, a *probabilistic classifier* that predicts an output for a new record based on observed outputs from similar records. Naive Bayes systems are really simple, and are useful mostly because they are exceedingly fast. Also, they are well suited to domains where you have catagorical data, that is, attributes that have multiple values that are not necessarily ordered (like the travel_to_school attribute, who's values are 'car', 'bus', 'walk', and 'skateboard').

An example should help make this algorithm clear. Let's say I told you that only 24 of 100 current US Senators are women. If I picked a Senator at random, would you guess that that Senator was a man or a woman?

We can *estimate* the probability using frequencies; 24/100 are W, 76/100 are M, so $P(W) = 0.24$ and $P(M) = 0.76$. I would certainly guess M, and be right most of the time. Note: this "estimate" is totally accurate, because we are not *sampling* from the population of Senators but actually counting the entire population. Normally, we would sample a few of the population under study and then estimate the probability in the larger population from the frequency counts obtained on the sample, thereby implicitly assuming the sample is representative of the population at large.

Now say that I point out that 53 of 100 current US Senators are Republicans, leaving 47 Democrats (we'll count the two Independents as Democrats, since Bernie Sanders and Angus King caucus with the Democrats). If I were to pick a Senator at random, would you guess that the Senator is a Democrat or a Republican? Using the same idea as before, I would be inclined to guess R, but only just slightly so, given that $P(D) = 0.47$ and $P(R) = 0.53$. But what if I told you that the Senator I picked at random was a woman. Would this change your guess as to whether the Senator was a Democrat?

It should! Of the 24 women Senators, fully 17 are Democrats, so $P(D|W) = 17/24 = 0.71$ (much larger than $P(D) = 0.47$ which I would estimate if I had no knowledge about the gender of the selected Senator). Here, we use $P(D|W)$ to represent a *conditional probability*, that is, the probability that my randomly selected Senator is a Democrat given that — or ‘‘conditioned on’’ — the fact that they are female.

Now what if I again pick a Senator at random and ask you to estimate the probability that the selected Senator is both a Democrat and a woman, $P(D\&W)$? Normally, the *joint probability* of two events happening together is the product of their individual probabilities. So, we might guess that $P(D\&W) = P(D) \times P(W) = 0.47 \times 0.24 = 0.11$. But more intuitively, we can see the answer should be 17/100, or 0.17, since there are 17 Democratic women out of a total population of 100 Senators. Why the discrepancy? Because as we have already seen, these two events are not independent: if they were, then woman Democrats and woman Republicans should occur at the same rate.

Instead, in this case where the events are not independent, the joint probability $P(D\&W)$ is related mathematically to the conditional probability $P(D|W)$ as follows:

$$P(D\&W) = P(W) \times P(D|W)$$

If we plug in the numbers, we see $0.24 \times 0.71 = 0.17$, which is the same as the intuitive answer we got from counting the 17 Democratic women in the US Senate. Likewise, $P(D\&W) = P(D) \times P(W|D) = 0.47 \times P(W|D) = 0.17$ yields a $P(W|D) = 0.36$, the probability that a known Democratic Senator is also female. These equations hint at some deeper relationship between $P(D)$, $P(W)$, $P(D\&W)$, $P(W|D)$ and $P(D|W)$.

Thomas Bayes, an 18th century English statistician and clergyman, introduced his *Bayes' Rule*, which establishes the mathematical relationships on probabilities between events when they are not *independent* one from the other:

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$$

or, in our case:

$$P(W|D) = \frac{P(D|W) \times P(W)}{P(D)}$$

Again, with our numbers, we see:

$$P(W|D) = \frac{0.71 \times 0.24}{0.47} = 0.36$$

which we can confirm numerically, as 17/47 Democratic Senators, or 36%, are women.

Another Example

Now let's consider another, more complex, example. Imagine you own a lousy car that only starts half the time. Over time, you make the following observations:

$P(S) = 0.5$ (car won't start)
$P(B) = 0.1$ (battery is dead)
$P(G) = 0.2$ (gas tank is empty
$P(S|G) = 1$ (car won't start without gas)
$P(S|B) = 1$ (car won't start with a bad battery)

Where we might think of G and B as two alternative diseases and S as an observable symptom.

We'd like to compute $P(G|S)$ and $P(B|S)$ so you can take corrective action (filling the tank or recharging the battery, respectively). Using Bayes' Rule:

$P(G|S) = (P(G) \times P(S|G))/P(S)) = 0.2 \times 1/0.5 = 0.4$
$P(B|S) = (P(B) \times P(S|B))/P(S)) = 0.1 \times 1/0.5 = 0.2$

By taking the more probable diagnosis (here, $P(G|S)$), we conclude that the car won't run because it is out of gas (note that when comparing the two diseases, we can really just compare the numerators and ignore dividing by $P(S)$ as we are only interested in which one is larger).

Now let's assume that $P(R)$ is the probability that the radio doesn't work. Note that while we know that the radio won't work if the battery is dead, we also know that the radio can still work if we are out of gas.

$P(S\&R) = 0.2$ (won't start and hinky radio -- your lucky day!)
$P(S\&R|G) = 0.3$
$P(S\&R|B) = 1$ (radio won't work without power!)

Here, the values of $P(G|S\&R)$ and $P(B|S\&R)$ would give the most probable cause of failure given the combinations of observed events.

$P(G|S\&R) = P(G) \times P(S\&R|G)/P(S\&R) = 0.2 \times 0.3/0.2 = 0.3$
$P(B|S\&R) = P(B) \times P(S\&R|B)/P(S\&R) = 0.1 \times 1/0.2 = 0.5$

In this case, its clearly more likely that the battery is the issue if both the radio is out and the car doesn't start.

Of course, the problem here is that as I add more and more observable conditions (like in HW3, where you know about age, foot length, Internet access, superpowers and so on) you need to estimate the joint and conditional probabilities for every combination of observables. And because we may have very few samples of very complicated conditions, it is impossible to estimate the above probabilities with frequencies alone. A single disease with *n* different True/False observable symptoms would require $O(2^n)$ estimates from data:

$$P(combination - of - symptoms)$$
$$P(combination - of - symptoms|D)$$

The Naive Bayes' algorithm makes a single assumption that reduces this burden.

The Naive Bayes' algorithm makes a single assumption that reduces this burden. By assuming that each symptom is an independent event, we can estimate the different values as follows:

$$P(S1\&S2) \approx P(S1) \times P(S2)$$
$$P(S1\&S2|D) \approx P(S1|D) \times P(S2|D)$$

These *independence assumptions* will make the system less reliable if they are violated; however, in practice, the result is often not sufficiently bad to compromise the relative likelihood when comparing alternative diseases.

Predicting Superpowers

In this homework, you will build a Naive Bayes system that can be used to predict one variable or attribute from a subset of the other attributes. So, for example, imagine you were asked what superpower a 160 inch tall right-handed male who likes basketball, drinks tea, and prefers English class would most likely desire? Freezing time? Flying? Invisibility? Telepathy?

To answer this question, you would compare the probability estimate for each of the possible superpowers (the "outcome") based on the combination of known attributes (the "evidence") under the conditional independence assumption. Bayes' Rule tells us that:

$$P(outcome|evidence) = \frac{P(evidence|outcome) \times P(outcome)}{P(evidence)}$$

so the approach is simply to compare $P(lotteryA|evidence)$ with $P(lotteryB|evidence)$ and then predict whichever one has the higher value. Note that when computing $P(lotteryA|evidence)$ and $P(lotteryB|evidence)$ they will both have the same denominator, $P(evidence)$, so since we are only interested in whichever one is larger, we can skip the denominator altogether and simply compare their respective $P(evidence|outcome) \times P(outcome)$ values directly.

What makes this algorithm ‘‘naive?’’ Well, the naive part comes from the assumption that P(evidence), which is really the probability of a number of individual pieces of evidence occurring together, can be approximated by the product of the probabilities of each individual piece of evidence. In other words:

$$P(evidence|outcome) = P(e_1 \wedge e_2 \wedge \cdots e_N |outcome) = P(e_1|outcome) \times \cdots \times P(e_N |outcome)$$

This is a critical assumption, because it means you don't have to consider all 2^N combinations of $e_1 \cdots e_N$ values, rather just each individual one. That's a huge savings, especially when you consider that some e_i's are not yes/no, but rather have more possible values, meaning the base of that $2N$ exponent is actually larger than 2.

What to Do

This homework consists of just two functions. Download the template file, which contains part of my solution to homework 3 (the getData() function that reads in a data file). You will implement two functions, train() and predict(). The train() function takes the output of getData() and the attribute you wish to predict (*e.g.*, ’Superpower’) and returns a dictionary containing all the elements you need to make a prediction (*e.g.*, all the $P(e_i|outcome)$ and $P(outcome)$ values). The predict() function takes the output of the train() function and a sample input consisting of a new record, and returns a list of possible outcomes ordered by their likelihood.