Bayesian Learning

Probability notions

Independence

Two events A and B are independent if $P(A|B) = P(A)$ and $P(B|A) = P(B)$.

Conditional independence

There are situations where three or more events may be dependent by each other, but can be expressed as independent if expressed in a certain way.

For example:

$$P(catch|toothache, cavity) = P(catch|cavity)$$

In this case the information about toothache is redundant, thus it can be considered conditionally independent.

In general, we say that X is conditionally independent from Y given Z if:

$$P(X|Y, Z) = P(X|Z)$$

The union of conditional independence and chain rule result in the following formula:

$$P(X, Y, Z) = P(X|Y, Z)P(Y, Z) = P(X|Y, Z)P(Y|Z)P(Z) = P(X|Z)P(Y|Z)P(Z)$$

The application of both rules simplifies a lot the number of computations needed, reducing them from an exponential (2^n) to a linear $(2 \cdot n)$ complexity.

Bayes' rule

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

It is useful to express this formula as the dependency between cause and effect:

$$P(cause|effect) = \frac{P(effect|cause)P(cause)}{P(effect)}$$

What if we want to study the probability of a certain cause to be caused by more effects?

If conditional independence can be applied, we can express it as:

$$P(cause|effect_1, ..., effect_n) = \alpha P(cause) \prod_i P(effect_i|cause)$$

Where α is a normalization factor that can be generalized for this purpose.

Bayesian learning

The foundation of Bayesian learning is that we can express a machine learning problem as a probabilistic one.

In other words, we define our target function $f : X \to V$, with X being our input domain and V our output (let's imagine it as classes). We condider D as the dataset and x' a new instance. The best prediction will be $\hat{f}(x') = v^*$, where:

$$v* = argmax_{v \in V} P(v|x', D)$$

> 💡 Conceptually we are saying that the best prediction is the one that maximizes the probability that it was generated given our prior knowledge of the dataset, and the knowledge of the new instance.

When training, we want to compute the following probability distribution:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

That is the probability that h is the solution given the dataset D, where:

- $P(D|h)$ is the probability that the dataset was generated from the hypothesis.
- $P(h)$ is the prior probability of the hypothesis h.
- $P(D)$ is the prior probability of training data D.

The MAP and ML hypothesis

The foundation of Bayesian learning says that the best the hypothesis is the *Maximum A Posteriori* (MAP) hypothesis, meaning the one that maximizes the probability that it was generated given the training dataset:

$$h_{MAP} = argmax_{h \in H} P(h|D) = argmax_{h \in H} P(D|h)P(h)$$

D will be our input, and we will have to search inside the hypothesis space H the optimal one. With the current formulation of this problem, we would have to know $P(h)$, namely the a-priory probability of h to happen. That is not always the case. In order to address the problem, we could simply assume that *all a-priori probabilities for our hypothesis are the same*, thus eliminating that term.

The problem formulated in this is called finding the *Maximum Likelihood* (ML) hypothesis:

$$h_{ML} = argmax_{h \in H} P(D|h)$$

Making predictions

Once we obtained our hypothesis, how do we make predictions?

We might think that applying $h_{MAP}(x')$ will give the best prediction, but is it actually the most probable one?

Let's consider the following case: we have three hypotesis h_1, h_2, h_3. Upon computation we find that:

$$P(h_1|D) = 0.4 \;\; P(h_2|D) = 0.3 \;\; P(h_3|D) = 0.3$$

Let's suppose that given a new instance x' we obtain the following predictions:

- $h_1(x') = positive$
- $h_2(x') = negative$
- $h_3(x') = negative$

If we just cosidered the MAP hypothesis, we would say that our prediction is positive. But we can observe that h_2 and h_3 give a compound contribution of 0.6, which is heavier than h_1.

We can elaborate a strategy to take in account this problem, by considered a weighted probability of our hypothesis. This approach is called the *Bayes' Optimal Classifier*.

Consider a target function $f : X \to V$, with $V = \{v_1, ..., v_n\}$ (the set of classes), the dataset D and a new instance x'.

The probability for each class will be:

$$P(v_j|x', D) = \sum_{h_i \in H} P(v_j|x', h_i) P(h_i|D)$$

Meaning that the probability for each class is the average of the probability of the class given each hypothesis weighted by the probability of that hypothesis was generated by the training dataset.

The best prediction will be:

$$v_{OB} = argmax_{v_j \in V} \sum_{h_i \in H} P(v_j|x', h_i) P(h_i|D)$$

The Optimal Bayes' Classifier always finds the optimal solution.

Naive Bayes Classifier

If we consideral the general case for a machine learning problem, we'll have to deal with two major problems: the space of hypothesis might be very big and instances can be described by a set of many attributes. In this cases finding a solution with the Bayes Optimal Classifier method might be unfeasable.

In order to deal with the general cases we use the Naive Bayes Classifier, which exploits the notion of conditional independence.

Let's assume that each instance x is described by a set of attributes $< a_1, ..., a_n >$. We want to compute:

$$argmax_{v_j \in V} P(v_j | x, D) = argmax_{v_j \in V} P(v_j | a_1, ..., a_n, D)$$

The MAP prediction will be (using Bayes' rule):

$$v_{MAP} = argmax_{v_j \in V} P(v_j | a_1, ..., a_n, D) = argmax_{v_j \in V} \alpha P(a_1, ..., a_n | v_j, D) P(v_j | D)$$

The naive Bayes assumtion is that all attributes are conditionally independent. Applying this consideration we wil obtain:

$$v_{NB} = argmax_{v_j \in V} P(v_j | D) \prod_i P(a_i | v_j, D)$$

What we obtained is a Naive Bayes Classifier. This kind of classifier does not guarantee to find the optimal solution, but it is still able to have good performances.