Bayesian Learning

Bayesian reasoning provides a probabilistic approach to inference. It is governed by the fact that assumptions about quantities is derived by probability distributions.

Features of Bayesian Learning

1. Each training example can decrease or increase the estimated probability that a hypothesis is correct.
2. Prior knowledge can be combined with observed data to determine the final probability. This knowledge is provided by asserting
    a. A prior probability for each candidate hypothesis
    b. A probability distribution over observed data for each possible hypothesis
3. Bayesian learning can accommodate hypotheses that make probabilistic predictions
4. New instances are classified by combining predictions of multiple hypotheses, weighted by their probabilities.
5. Bayesian methods are computationally intractable, but provide optimal decision making standards.

One difficulty of Bayesian learning is that they need a lot of knowledge about many probabilities. The second is the significant computational knowledge.

Bayes Theorem

The best hypothesis is said to be the most probable hypothesis given data D plus any initial knowledge about the prior probabilities of a hypothesis space H.

P(h) is called the prior probability of h, P(D) is the probability that the data D will be observed, P(D|h) is the probability of observing data D given a hypothesis h and P(h|D) is the posterior probability of h, which is

$P(h|D) = P(D|h)P(h) / P(D)$

In this, we find the most probable hypothesis h in H, and this is called the maximum a posteriori hypothesis (MAP).

$$h_{MAP} = \underset{h \in H}{argmax}\ P(h|D)$$
$$= \underset{h \in H}{argmax} \frac{P(D|h)\,P(h)}{P(D)}$$
$$= \underset{h \in H}{argmax} P(D|h)\,P(h)$$

P(D) is dropped as it is a constant.

In some cases, we assume that P(h) is equally probable for all h in H. So it just becomes

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

And this is called the maximum likelihood hypothesis.

Brute Force Bayes Concept Learning

For a sequence of instances X, a target concept c and D the target values for instances in X, we get a brute force MAP learning hypothesis as

For each hypothesis h in H, calculate P(h|D) = P(D|h)P(h) / P(D).

Output the MAP hypothesis with the highest posterior probability

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

In order to specify the learning problem, we make the following assumptions

1. Training data D is noise free
2. Target concept c is contained in hypothesis space H
3. No reason to believe any bias about any hypothesis

P(h) is given by 1 / |H| for all h in H.

Then,

$$P(D|h) = \begin{cases} 1\ if\ d_i = h(x_i)\ for\ all\ d_i\ in\ D \\ 0\ otherwise \end{cases}$$

Now, the learning problem is defined in a more concrete fashion where

$$P(h|D) = \begin{cases} 0 \; if \;\; h \; is \; inconsistent \; with \; D \\ \dfrac{1}{Version \; Space_{H,\,D}} \; if \;\; h \; is \; consistent \; with \; D \end{cases}$$

As P(D) = |V.S| / |H|.

This can be derived from the theorem of probability as

$$\begin{aligned} P(D) &= \sum_{h_i \in H} P\left(D|h_i\right) P\left(h_i\right) \\ &= \sum_{h_i \in V.S_{H,D}} 1.\frac{1}{|H|} + \sum_{h_i \notin V.S_{H,D}} 0.\frac{1}{|H|} \\ &= \frac{|V.S_{H,D}|}{|H|} \end{aligned}$$

MAP and Consistent Learners

Every consistent learner outputs a MAP hypothesis, if we assume a uniform prior probability distribution over H, and if we assume deterministic, noise-free training data.

Because most algorithms inherently output MAP hypotheses, the Bayesian framework allows one way to characterize the behavior of learning algorithms, even if the algorithm explicitly does not manipulate probabilities.

Maximum Likelihood and Least-Squared Error Hypotheses

Under certain assumptions, any learning algorithm that minimizes the squared error between output hypothesis predictions and the training data will output a maximum likelihood hypothesis.

For this we assume that every training example <x,d> can be categorised as d = f(x) + e, where f(x) is a noise free value of the target function and e is the error, where e belongs to a normal distribution with mean zero.

For continuous values, we define a probability density function as

$$p\left(x_0\right) = \lim_{e->0} \frac{1}{e} P\left(x_0 \leq x < x_0 + e\right)$$

Now, we plug this probability density in the maximum likelihood equation as

$$h_{ML} = \underset{h \in H}{argmax}\, p(D|h)$$

$$where\ D = <d_1, \ldots d_m>$$

$$This\ gives,$$

$$h_{ML} = \prod_{i=1}^{m} p\left(d_i|h\right)$$

Now since e obeys a normal distribution with 0 mean and unknown variance, each d also obeys a normal distribution around target value f(x).

Therefore, p(d|h) can also be written around an unknown variance and mean f(x).

This gives

$$h_{ML} = \underset{h \in H}{argmax} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}\left(d - h\left(x_i\right)\right)^2}$$

$$Now,\ we\ maximise\ \ln p\ as\ that\ also\ maximises\ p$$

$$h_{ML} = \underset{h \in H}{argmax} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2}\left(d_i - h\left(x_i\right)\right)^2$$

Now, discarding the first term and minimizing instead of maximising,

$$h_{ML} = \underset{h \in H}{argmin} \sum_{i=1}^{m} \frac{1}{2\sigma^2}\left(d_i - h\left(x_i\right)\right)^2$$

$$h_{ML} = \underset{h \in H}{argmin} \sum_{i=1}^{m} \left(d_i - h\left(x_i\right)\right)^2$$

This shows that the maximum likelihood hypothesis minimizes the sum of the squared errors.

We choose a normal distribution for the noise as an approximation to many types of noise in systems. The only problem with this is that it considers the noise only in the target value, but not in the attributes itself.

Bayes Optimal Classifier

The most optimal classification of a new instance is the weighted sum of the hypotheses outputs. For an example that can take on any value from set V, the probability P(v|D) that the correct classification for the instance is v is

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

The most optimal classif ication is

$$\underset{v_j \in V}{argmax} \sum_{h_i \in H} P(v_j|h_i) P(h_i|D)$$

This learner maximises the probability that a new instance is classified correctly, given the available data, the hypothesis space and the prior probabilities over the hypotheses.

Naive Bayes Classifier

The naive Bayes classifier applies to tasks where each instance x can be described as a conjunction of attribute values and where the target function can take up any value from a value set V.

If an attribute set is given as A = <a1,a2...aN>, the most probable value is given by

$$v_{map} = \underset{v_j \in V}{argmax}\ P(v_j|a_1, a_2, \ldots, a_N)$$

*Rewriting this u*sing *Bayes theorem*

$$v_{map} = \underset{v_j \in V}{argmax}\ \frac{P(a_1, a_2, \ldots, a_N|v_j) P(v_j)}{P(a_1, a_2, \ldots, a_N)}$$

$$v_{map} = \underset{v_j \in V}{argmax}\ P(a_1, a_2, \ldots, a_N|v_j) P(v_j)$$

The naive Bayes classifier is based on the assumption that the attribute values are conditionally independent of the target value, which means that they are just the product of the probabilities of each attribute containing that value.

So the naive Bayes classifier is given as

$$v_{NB} = \underset{v_j \in V}{argmax}\ P(v_j) \prod_i P(a|v_j)$$

The main difference between Naive Bayes classifiers is that the model does not search through the space of possible hypotheses.

Estimating Probabilities

When the samples to calculate an attribute's probability is too less, we use smoothing methods.

1. M-estimate

$$m - estimate\ of\ \ probability = \frac{n_c + mp}{n + m}$$

Where p is the prior estimate of the probability and m is called the equivalent sample size. p is usually set to 1/k where k is the number of distinct values the attribute can take.

2. Add-one Laplace smoothing

When mp = 1, and m is not 1, it is said to be an additive smoothing or add-one Laplace smoothing. This is usually used in text classification, where m is the size of the vocabulary.