Week 6: Decision Trees and Ensembles

March 13, 2018

1 Decision trees

Imagine you do a newspaper round to help you get through these lean times. On your round, you encounter a number of dogs that either bark or (try to) bite. The dogs are described by the following binary features: *Heavy*, *Smelly*, *Big* and *Growling*. Consider the following set of examples:

Heavy	Smelly	Big	Growling	Bites
No	No	No	No	No
No	No	Yes	No	No
Yes	Yes	No	Yes	No
Yes	No	No	Yes	Yes
No	Yes	Yes	No	Yes
No	No	Yes	Yes	Yes
No	No	No	Yes	Yes
Yes	Yes	No	No	Yes

question 1: What is the entropy of the target value *Bites* in the data?
question 2: Which attribute would the ID3 algorithm choose to use for the root of the tree (without pruning)?
question 3: What is the information gain of the attribute you chose in the previous question?
question 4: Draw the full decision tree that would be learned for this data using ID3 without pruning.

question 5: Suppose three new dogs appear in your round as listed in the table below. Classify them using the decision tree from the previous question.

Dog	Heavy	Smelly	Big	Growling	Bites
Buster	Yes	Yes	Yes	Yes	?
Pluto	No	Yes	No	Yes	?
Zeus	Yes	Yes	No	No	?

question 6: Someone proposes a new scheme to prevent overfitting: she suggests to set a pre-defined maximum depth for the decision trees. When the standard algorithm reaches this depth, it terminates. Could this help to prevent overfitting? Why (not)?
question 7: In the maximum depth scheme introduced above, how would you determine a good value for the maximum depth for a given data set?

question 8: Why can't we apply L1-regularization to this decision tree learning problem?

2 Variational autoencoders

The maximum likelihood principle tells us to optimize the quantity $p(x \mid \theta)$ as a function of θ (the model parameters).

For complex models, this does not usually lead to a closed form solution and applying gradient descent to θ directly tends to lead to mode collapse. Instead we rely on the following equality:

$$\ln p(\mathbf{x} \mid \theta) = L(q, \theta) + KL(q, p)$$

with

$$\begin{aligned} q(z) &\text{ any distribution on } z \\ p &= p(z \mid \mathbf{x}, \theta) \\ KL(q, p) &\text{ the Kullback-Leibler divergence} \\ L(q, \theta) &= \mathbb{E}_q \ln \frac{p(\mathbf{x}, z \mid \theta)}{q(z)} \end{aligned}$$

We will first prove that this equality holds. We start with the right hand side, fill in the components, and derive the left-hand side.
question 9: Fill in the blanks. We have written everything in terms of *expectations* $\mathbb{E}$ to simplify the notation. The expectation is over the random variable z, while $\mathbf{x}$ has some definite value. Note that $\mathbb{E}f(z) + \mathbb{E}g(z) = \mathbb{E}[f(z) + g(z)]$.

$$
\begin{aligned}
L(q, \theta) + KL(q, p) &= \dots \\
&= \mathbb{E}_q \ln \frac{p(x, z \mid \theta)}{q(z)} - \mathbb{E}_q \ln \frac{p(z \mid x, \theta)}{q(z)} \\
&= \dots \\
&= \mathbb{E}_q \ln p(x, z \mid \theta) - \mathbb{E}_q \ln p(z \mid x, \theta) \\
&= \mathbb{E}_q \ln \frac{p(x, z \mid \theta)}{p(z \mid x, \theta)} = \dots \\
&= \mathbb{E}_q \ln p(x \mid \theta) = \ln p(x \mid \theta)
\end{aligned}
$$

In the EM algorithm, we search by alternately optimizing $L(q \mid \theta)$ with respect to θ and setting q equal to p (so that the KL term becomes zero).
question 10: For the variational autoencoder, we cannot (easily) perform this last step. Why not?

Instead, we *approximate* $p(z \mid x, \theta)$ with a neural network $q(z \mid x)$ that produces a distribution on z given some x. This gives us an auto-encoder-like structure. An input is mapped to a distribution $q(z \mid x)$ by the *encoder*. We sample a single z from this distribution and pass it through the *decoder* $p(x \mid z, \theta)$ to produce a distribution on x (see the slides for diagrams).

To find a way to train such an architecture, we turn again to our decomposition of the likelihood:

$$\ln p(x \mid \theta) = L(q, \theta) + KL(q, p) \ .$$

Since the KL divergence is always positive, this tells us that

$$\ln p(x \mid \theta) \geqslant L(q, \theta)$$

for *any* q we choose. This is why L is called the variational *lower bound*.[1] If we choose our parameters θ and our function q to maximize L, we are also, indirectly, maximizing $p(x \mid \theta)$.[2]

To do so, we rewrite it into two separate terms: a KL divergence and an expectation:

[2] The word *variational* comes from the fact that one of its arguments, q, is a function (the calculus of functions is called *variational* calculus). For our purposes, this distinction doesn't matter much, since the function q is defined by a set of parameters, so ultimately we will take the derivative over those parameters, as we are used to.

[2] How close the lower bound L comes to the true value $p(x \mid \theta)$ depends on how well our encoder network q approximates the true conditional distribution on z: $p(z \mid x, \theta)$.

$$L(q, \theta) = -KL(q(z \mid x), p(z)) + \mathbb{E}_q \ln p(x \mid z, \theta)$$

question 11: Show that this equation holds (i.e. rewrite the left part into the right).

Thus, to optimize our variational autoencoder, we should maximize L. In other words, $-L$ is our loss function. The only problem left to solve is that the second term is an expectation (which we cannot compute explicitly). **question 12:** How is this solved in practice?