CS181 Practice Questions: Neural Networks and Decision Trees

1. **Multiclass Classification Error Function (Bishop 5.5)**

Consider a K-class supervised classification scenario with training data $\{\boldsymbol{x}_n, \boldsymbol{t}_n\}_{n=1}^N$, where the $\boldsymbol{t}_n$ are 1-hot binary vectors, with t_{nk}=1 iff $\boldsymbol{x}_n$'s true class is k. Assume we model this problem using a neural network with K output-units, where the interpretation of the k'th output unit, denoted $y_k(\boldsymbol{x}_n, \boldsymbol{w})$, is $y_k(\boldsymbol{x}_n, \boldsymbol{w}) = p(t_{nk}{=}1 \mid \boldsymbol{x}_n)$. Show that maximizing the (conditional) likelihood of such a model is equivalent to minimizing the cross-entropy loss function given by

$$E(\boldsymbol{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \ln y_k(\boldsymbol{x}, \boldsymbol{w}) \qquad \text{(Bishop 5.24)}$$

Using the usual binary exponent trick, the likelihood of a single example is

$$p(\boldsymbol{t}_n|\boldsymbol{x}_n, \boldsymbol{w}) = \prod_{k=1}^{K} y_k(\boldsymbol{x}_n, \boldsymbol{w})^{t_{nk}}.$$

So, the likelihood of all the data is

$$p(\boldsymbol{T}|\boldsymbol{X}, \boldsymbol{w}) = \prod_{n=1}^{N}\prod_{k=1}^{K} y_k(\boldsymbol{x}_n, \boldsymbol{w})^{t_{nk}}.$$

Taking logs, we get $\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \ln y_k(\boldsymbol{x}, \boldsymbol{w})$, which is the negative of the error function above, and clearly maximizing a function is the same as minimizing its negative. Note that this is the same objective as in multiclass logistic regression.

2. **Activation Gradients (Bishop 5.6)**

Consider a single-output network solving a binary classification problem, and trained with the following cross-entropy error function

$$E(\boldsymbol{w}) = -\sum_{n=1}^{N} \left[t_n \ln y(\boldsymbol{x}_n, \boldsymbol{w}) + (1 - t_n) \ln(1 - y(\boldsymbol{x}_n, \boldsymbol{w})) \right], \quad \text{(Compare Bishop 5.21)}$$

where $y(\boldsymbol{x}_n, \boldsymbol{w}) = \sigma(a_n) = \frac{1}{1+\exp(-a_n)}$ for an activation a_n. Show that the derivative of the error function above wrt a particular a_n satisfies

$$\frac{\partial E}{\partial a_n} = y(\boldsymbol{x}_n, \boldsymbol{w}) - t_n \qquad \text{(Compare Bishop 5.18)}$$

For a particular a_n we have (by the chain rule)

$$\begin{aligned}
\frac{\partial E}{\partial a_n} &= -t_n \frac{1}{y(\boldsymbol{x}_n, \boldsymbol{w})} \frac{\partial y(\boldsymbol{x}_n, \boldsymbol{w})}{\partial a_n} - (1 - t_n) \frac{1}{1 - y(\boldsymbol{x}_n, \boldsymbol{w})} \frac{\partial (1 - y(\boldsymbol{x}_n, \boldsymbol{w}))}{\partial a_n} \\
&= -t_n \frac{1}{\sigma(a_n)} \frac{\partial \sigma(a_n)}{\partial a_n} - (1 - t_n) \frac{1}{1 - \sigma(a_n)} \frac{\partial (1 - \sigma(a_n))}{\partial a_n} \\
&= -t_n \frac{1}{\sigma(a_n)} \sigma(a_n)(1 - \sigma(a_n)) + (1 - t_n) \frac{1}{1 - \sigma(a_n)} \sigma(a_n)(1 - \sigma(a_n) \\
&= -t_n (1 - \sigma(a_n)) + (1 - t_n) \sigma(a_n) \\
&= \sigma(a_n) - t_n,
\end{aligned}$$

Where in the second line we noted that $y(\boldsymbol{x}_n, \boldsymbol{w}) = \sigma(a_n)$, and in the third we made use of equation (4.88) in Bishop, viz., $\frac{\partial \sigma(a)}{\partial a} = \sigma(a)(1 - \sigma(a))$.

3. **Positive Definiteness of Hessian (Bishop 5.10)**

Consider a hessian matrix $\boldsymbol{H} \in \mathbb{R}^D$ with eigenvector equation

$$\boldsymbol{H}\boldsymbol{u}_d = \lambda_d \boldsymbol{u}_d, \tag{Bishop 5.33}$$

where the eigenvectors $\boldsymbol{u}_d$ form an orthonormal basis of $\mathbb{R}^D$. As Bishop points out, since any vector $\boldsymbol{v} \in \mathbb{R}^D$ can be written as

$$\boldsymbol{v} = \sum_{d=1}^{D} c_d \boldsymbol{u}_d, \tag{Bishop 5.38}$$

the orthonormality of the eigenvectors implies that

$$\boldsymbol{v}^\mathsf{T}\boldsymbol{H}\boldsymbol{v} = \sum_{d=1}^{D} c_d^2 \lambda_d. \tag{Bishop 5.39}$$

(a) As a first step, show how to derive (Bishop 5.39) from Bishop (5.38).

(b) Now, recall that a matrix is positive definite if $\boldsymbol{v}^\mathsf{T}\boldsymbol{H}\boldsymbol{v} > 0$ for all nonzero $\boldsymbol{v}$. Show that $\boldsymbol{H}$ is positive definite if and only if all of its eigenvalues are positive.

(a) From (Bishop 5.38), we have

$$\begin{aligned}
\boldsymbol{v}^\mathsf{T}\boldsymbol{H}\boldsymbol{v} &= (\sum_{d=1}^{D} c_d \boldsymbol{u}_d)^\mathsf{T} \boldsymbol{H} (\sum_{d=1}^{D} c_d \boldsymbol{u}_d) \\
&= (\sum_{d=1}^{D} c_d \boldsymbol{u}_d)^\mathsf{T} (\sum_{d=1}^{D} c_d \lambda_d \boldsymbol{u}_d) \\
&= \sum_{d=1}^{D} c_d^2 \lambda_d,
\end{aligned}$$

where in the second line we brought $\boldsymbol{H}$ inside the right parentheses, and in the third line we made use of the fact that $\boldsymbol{u}_d^\mathsf{T}\boldsymbol{u}_{d'} = 0$ if $d \neq d'$ (because these vectors are orthonormal).

(b) $\Rightarrow$:
If all the eigenvalues λ_d are positive, then since the c_d's are squared, the sum $\sum_{d=1}^{D} c_d^2 \lambda_d$ consists of terms that are all non-negative. Since we only consider nonzero $\boldsymbol{v}$, at least one of the c_d^2's must be positive and so the sum is positive.
$\Leftarrow$:
If $\boldsymbol{v}^\mathsf{T}\boldsymbol{H}\boldsymbol{v} > 0$ for all nonzero $\boldsymbol{v}$, then for each $\boldsymbol{u}_d$ (which are nonzero) we must have $\boldsymbol{u}_d^\mathsf{T}\boldsymbol{H}\boldsymbol{u}_d = \lambda_d \boldsymbol{u}_d^\mathsf{T}\boldsymbol{u}_d = \lambda_d > 0$.

4. **Approximate Hessian (Bishop 5.2)**

The outer product approximation to the Hessian matrix for a neural network using a sum-of-squares error function is given by

$$\boldsymbol{H} = \sum_{n=1}^{N} \boldsymbol{b}_n \boldsymbol{b}_n^{\mathsf{T}}$$
$$\boldsymbol{b}_n = \nabla y_n = \nabla a_n$$

This is the approximation to the Hessian matrix of $E = \frac{1}{2}\sum_{n=1}^{N}(y_n - t_n)^2$. Derive an expression for an approximation to the Hessian for the case of multiple outputs. Consider the multivariate generalization of the energy function:

$$E = \frac{1}{2}\sum_{n=1}^{N}(\boldsymbol{y}_n - \boldsymbol{t}_n)^{\mathsf{T}}(\boldsymbol{y}_n - \boldsymbol{t}_n)$$

Take the partial derivative with respect to w_i and find

$$\frac{\partial E}{\partial w_i} = \sum_{n=1}^{N}(\boldsymbol{y}_n - \boldsymbol{t}_n)^{\mathsf{T}}\frac{\partial \boldsymbol{y}_n}{\partial w_i},$$

and then compute the second partials to find

$$\frac{\partial^2 E}{\partial w_i \partial w_j} = \sum_{n=1}^{N}\frac{\partial \boldsymbol{y}_n}{\partial w_j}^{\mathsf{T}}\frac{\partial \boldsymbol{y}_n}{\partial w_i} + (\boldsymbol{y}_n - \boldsymbol{t}_n)^{\mathsf{T}}\frac{\partial^2 \boldsymbol{y}_n}{\partial w_j \partial w_i}$$

Now, we can assume that the second term in the sum above vanishes, as once we've trained the neural network, the quantity $(\boldsymbol{y}_n - \boldsymbol{t}_n)$ becomes a random variable with 0 mean (same argument as the univariate case presented in Bishop). Then, we have that

$$\boldsymbol{H} \approx \sum_{n=1}^{N} \boldsymbol{B}_n \boldsymbol{B}_n^{\mathsf{T}}$$
$$(\boldsymbol{B}_n)_{ij} = \frac{\partial y_{nk}}{\partial w_l}$$

5. **Likelihood Function of Conditional (Bishop 5.2)**

Show that maximizing the likelihood function under the conditional distribution given by

$$p(\boldsymbol{t}|\boldsymbol{x}, \boldsymbol{w}) = \mathcal{N}(\boldsymbol{t}|\boldsymbol{y}(\boldsymbol{x}, \boldsymbol{w}), \beta^{-1}\boldsymbol{I})$$

for a multioutput neural network is equivalent to minimizing the sum of squares error function given by

$$E(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N} ||\boldsymbol{y}(\boldsymbol{x}_n, \boldsymbol{w}) - \boldsymbol{t}_n||^2$$

We first write the likelihood function for an i.i.d. data set under the conditional distribution, which is given by

$$\prod_{n=1}^{N} \mathcal{N}(\boldsymbol{t}_n|\boldsymbol{y}(\boldsymbol{x}_n, \boldsymbol{w}), \beta^{-1}\boldsymbol{I})$$

We can take the logarithm of this to get

$$\begin{aligned} &= -\frac{1}{2}\sum_n (\boldsymbol{t}_n - \boldsymbol{y}(\boldsymbol{x}_n, \boldsymbol{w}))^\mathsf{T}(\beta\boldsymbol{I})(\boldsymbol{t}_n - \boldsymbol{y}(\boldsymbol{x}_n, \boldsymbol{w})) + \text{const.} \\ &= -\frac{\beta}{2}\sum_n ||\boldsymbol{t}_n - \boldsymbol{y}(\boldsymbol{x}_n, \boldsymbol{w})||^2 + \text{const.} \end{aligned}$$

so we see that the terms dependent on $\boldsymbol{w}$ (the first term) is proportional to the negative of the sum-of-squares error function, so maximizing it is equivalent to minimizing the sum-of-square function.

6. **Degrees of Freedom of the Hessian (Bishop 5.13)**

Determine that the number of independent elements (degrees of freedom) in the quadratic error function, given by the Taylor Expansion of the error function around the point $\hat{\boldsymbol{w}}$

$$E(\boldsymbol{w}) = E(\hat{\boldsymbol{w}}) + (\boldsymbol{w} - \hat{\boldsymbol{w}})^{\mathsf{T}}\boldsymbol{b} + \frac{1}{2}(\boldsymbol{w} - \hat{\boldsymbol{w}})^{\mathsf{T}}\boldsymbol{H}(\boldsymbol{w} - \hat{\boldsymbol{w}})$$

is given by $\frac{W(W+3)}{2}$. Hint: What properties of the Hessian matrix restrict its number of independent elements?

Since the Hessian matrix is symmetric (as partial with respect to x_i then x_j is the same as the partial with respect to x_j then x_i), we have that there are $1 + 2 + \cdots W = \frac{W(W+!)}{2}$ free elements of $\boldsymbol{H}$ (just take the diagonal and everything above it). Furthermore, we have W independent elements from the gradient vector $\boldsymbol{b}$. Summing these gives the desired result of $W(W+3)/2$.

7. **Information Theory**

Calculate the Shannon entropy of a fair coin, and the Shannon entropy of an unfair coin that comes up heads 80% of the time.

The definition of entropy is:

$$H(X) = -\sum p(x) \log_2 p(x)$$

So, we have for the fair coin X,

$$H(X) = \left(-\frac{1}{2}\log_2 \frac{1}{2}\right) + \left(-\frac{1}{2}\log_2 \frac{1}{2}\right) = 1$$

For the unfair 80% coin Y,

$$H(Y) = (-0.8 \log 0.8) + (-0.2 \log 0.2) = 0.2068$$

Both of these are in units of bits.

8. **Choosing Attributes (Dublin University)**

ID	Age	Income	Student	Credit	Buys
1	< 31	high	no	bad	no
2	< 31	high	no	good	no
3	$31 - 40$	high	no	bad	yes
4	> 40	med	no	bad	yes
5	> 40	low	yes	bad	yes
6	> 40	low	yes	good	no
7	$31 - 40$	low	yes	good	yes
8	< 31	med	no	bad	no
9	< 31	low	yes	good	yes
10	> 40	med	yes	bad	yes
11	< 31	med	yes	good	yes
12	$31 - 40$	med	no	good	yes
13	$31 - 40$	high	yes	bad	yes
14	> 40	med	no	good	no

A dataset collected in an electronics shop showing details of customers and whether or not they responded to a special offer to buy a new laptop is shown in the table above. We plan to use this dataset to build a decision tree to predict which customers will respond to future special offers. We are trying to decide on the root node of our decision tree with ID3. We are given that the information gain of the feature *Age* at the root node of the tree is 0.247. A colleague has suggested that *Student* would be a better feature to consider for the root node. Show that this is not the case.

We calculate as follows, where pos represents the total number of positive examples and neg represents the total number of negative examples. pos_i is the number of positive examples when an attribute takes its ith value, and neg_i is the corresponding number of negative values.

$$Gain(A) = I\left(\frac{pos}{pos+neg}, \frac{neg}{pos+neg}\right) - Remainder(A)$$

$$Remainder(A) = \sum_{i=1}^{v} \frac{pos_i + neg_i}{pos+neg} \times I\left(\frac{pos_i}{pos_i+neg_i}, \frac{neg_i}{pos_i+neg_i}\right)$$

$$I\left(\frac{pos}{pos+neg}, \frac{neg}{pos+neg}\right) = -\frac{pos}{pos+neg}\log_2\frac{neg}{pos+neg} - \frac{pos}{pos+neg}\log_2\frac{neg}{pos+neg}$$

We see here that $v = 2$ since *Student* only takes on two values. When *Student* is no, we get $pos_i = 3$ and $neg_i = 4$ and when *Student* is yes, we get $pos_i = 6$ and $neg_i = 1$.

Calculating this out gives an information gain of 0.152 for *Student*, which is worse than that of *Age*.

9. **More On Choosing Attributes (Dublin University)**

In the above example, yet another colleague suggested that the ID attribute would be another good variable to consider at the root node. Would you agree with this suggestion? This should not require computation.

The ID would not be a good variable because it is arbitrary and gives no actual information about buyers' preferences. It will just lead to massive overfitting of the training data. For those interested in calculations, the information gain is 0.94.

10. **Classification Trees (Elements of Statistical Learning)**

Consider a data set on which you were training a classification tree that contained elements from two classes, call them $\mathcal{C}_1, \mathcal{C}_2$. Now, suppose that we have 200 training points $\{x_i\}_{i=1}^{200}$, with 100 belonging to the first class and 100 belonging to the second class. Let the node (a, b) contains a elements of $\mathcal{C}_1$ and b elements of $\mathcal{C}_2$. Now, suppose we produce two potential classification trees, T_1, T_2, where T_1 created a split that created nodes $(75, 25)$ and $(25, 75)$, while T_2 contained a split that created nodes $(50, 100), (50, 0)$. Explain why T_2 is a preferable decision tree even though it causes the same misclassification error.

First, we see that both trees produce a misclassification rate of 0.25 (i.e., $1/4$ of the objects are in the "wrong" nodes), but in the latter case one of our nodes contains all of the elements of $\mathcal{C}_1$ and the other contains none of the elements of $\mathcal{C}_2$, creating "pure" nodes, meaning that the split is a great characterizes of elements in $\mathcal{C}_1$. Thus, it is preferable to the classification rule that cannot discriminate between the classes as effectively.

11. **Entropy Intuition (Dublin University)**

In problem 1 you calculated the Shannon entropy of a biased coin that comes up head 80% of the time. First, calculate the Shannon entropy of a fair coin and give an explanation for why the entropy of the biased coin is less than the entropy of the fair coin,

To calculate the entropy of a fair coin, we use the formula for Shannon's entropy just as we did in the first problem to get

$$\begin{aligned} H(X) &= -\frac{1}{2}\log_2(\frac{1}{2}) - \frac{1}{2}\log_2(\frac{1}{2}) \\ &= -\frac{1}{2}(-1) + -\frac{1}{2}(-1) \\ &= 1 \end{aligned}$$

We recall from problem 1 that the entropy for the biased coin is ≈ 0.2068 bits. We can understand why the answer is less by remembering the entropy is a measure of uncertainty, and the unbiased coin is less uncertain than the unbiased coin. In other words, given the biased coin, if we simply predict that it will land heads every time we will be right 80% of the time, so it takes fewer bits to encode the most likely outcome.

12. **Decision Tree Recursion (Dublin University)**

Suppose we are given a decision tree and we generate a dataset from the decision tree. We then train a new decision tree on this dataset, generate data from the new tree, and repeat this process. As the size of the training set approaches infinity, will we eventually return the original tree using the above recursive process? Note that we only care about "logical" equivalence (i.e., a decision tree that will produce the same outputs for a given input but may have a different structure).

The algorithm will eventually return a tree that is logically equivalent, as we will eventually generate all possible combinations of input attribute during the generation of new data. Since we will eventually generate the same combination of attributes as the set that defined the original tree. This is true because any two decision trees defined on the same set of attributes that agree on all possible examples are, by definition, logically equivalent. As we noted in the problem statement, the actual form of the tree may be different, but they are equivalent in function.