*Note: accuracy of these answers is not guaranteed – feel free to comment or fix any errors you spot. Also it’s recommended to open this in the desktop version of Word (in the web version, File -> Info)*

## Question 1

*Part a*

|  |  |  |  |
| --- | --- | --- | --- |
| *Instance number (validation)* | *Nearest instances* | *Prediction* | *Correct? Yes = 1, No = 0* |
| 1 | 4,5,6 | + | Yes |
| 2 | 3,4,5 | + | Yes |
| 3 | 2,4,5 | + | Yes |
| 4 | 2,3,5 | + | Yes |
| 5 | 6,1,4 | + | Yes |
| 6 | 1,5,8 | + | No |
| 7 | 10,9,8 | - | Yes |
| 8 | 9,6,1 | - | Yes |
| 9 | 8,10,7 | - | Yes |
| 10 | 7,9,3 | - | Yes |

Accuracy is hence 90% (9/10).

*Part b*

The initial entropy for the whole dataset is simply

*Note:*

First we find the information for the *income* class. There are two possibilities – low and high income (and both have an equal number of samples). This is (0.029)

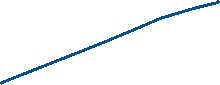
Now we find the information gain for the *stress* attribute. There are 110 samples of high stress (55%), and 90 samples of low stress (45%). Then, (0.1984)

We should choose the *stress* class because that has the highest information gain, and hence would be most useful when splitting (as will help us get the most information possible).

## Question 2

*Part a*

First we find the output (see below)



Then, we backpropagate:

Notice however that is the activation function, and hence

While,

And,

Hence,

Similarly, we can compute (for the bias):

Notice that

Hence again,

All we need to do now is to apply the gradient descent formula:

Same for the bias, which would give us .

*Note:* [*https://edstem.org/us/courses/14725/discussion/943405?answer=2143926*](https://edstem.org/us/courses/14725/discussion/943405?answer=2143926) *is an alternative using a slightly longer approach:*

Text

Description automatically generated

*Part b*

This is clearly overfitting, as the RMSE of the validation error is much higher than that of the training set (which is improving as the number of epochs increase).

Hence measures to counteract this would involve things such as adding more data or reducing the complexity of the neural net model.

## Question 3

*Note:* [*https://edstem.org/us/courses/14725/discussion/948179*](https://edstem.org/us/courses/14725/discussion/948179) *says that the K-Means algorithm has to be run until convergence – only* ***one*** *iteration has been shown below.*

Text

Description automatically generated

*Also on the desktop version of Word, double-clicking the below tables will open up an embedded Excel worksheet that you can use to see the calculation/formulas.*

1. The initial means are and .

Then we perform an assignment step. Note that we are using the L1 norm – that’s . This gets us (*see below)*



Now that we know what the clusters assigned to each index are, we can perform the update step.

These represent the locations of the new centroids (todo: this needs to be repeated until convergence)

1. The first step is to find out which cluster each index is assigned to. Note that “res” is the value of the L1 norm in the cluster that the particular index would be assigned.

*For execution 1*



*For execution 2*



As the loss function (which is simply the sum of the deviations for each index from the assigned cluster mean by the number of indices) for the second execution is less than the first, that’s the better choice.

1. Fitness function: minimise the loss (distance from each of the clusters to each index) – as evolutionary algorithms always maximise, we need to find the maximum of the *negative* of the loss.

Genotype: vector of floats of size where *n* is the number of clusters – this is an EA since the positions of the centroids are floats.

Phenotype (converting from genotype) each subarray of size 2 will correspond to the mean position of the centroid in the 2-D space. The phenotype itself would be the K-Means system with the mean cluster locations.

Mutation operator: addition of a Gaussian noise (as we’re dealing with EA)

