# 2020 Intro to ML Exam Solution

1a. Below is the answer to the question

i) A book distributor has a collection of books, which it has classified into different categories, e.g. “Young adults”, “Science fiction”, “Biography”, and “Horror”. It wants to use this information to build a system to classify its new products automatically.

> Supervised learning. This problem is a classification problem with the categories of each book already known, and thus one should use this to train the model, lower the loss function, and then make new predictions for unseen data

ii) A supermarket has a database of its customers, and wants to automatically discover and group its customers into different market segments to target them separately.

> Unsupervised learning. This is clearly a clustering problem where the labels/clusters of each customer is not known before constructing the model and the model will find latent structures within the data points

iii) A group of aviation companies wants to develop a machine learning algorithm which can predict the (x, y) coordinates pinpointing the location of a plane that has crashed. To develop this, the companies have collected historical plane crash data, which include the coordinates of the planes’ crash sites.

> Unsupervised learning. The label/output of each data point is unknown before training and the model will find internal patterns withn the crash data and thus learn to pinpoint the location.

1b. Below is the answer to the question

i) Using the Information Gain metric, which attribute will be selected as the root node of a decision tree classifier? Please show all calculations (including the Information Gain of all candidate nodes) to justify your answer.

> We first calculate the information entropy of the entire dataset, then calculate the entropy of selecting each attribute and using it to divide the dataset.
>
> Notice that the number of spam and not-spam emails are the same in the dataset, thus the information entropy of the entire dataset is just 1
>
> The information gain for selecting each attribute as the root node is calculated below

$$
\begin{aligned}
H_{cash} &= 1 - (\frac{4}{10} \cdot (-\frac{1}{4}\log_2\frac{1}{4} - \frac{3}{4}\log_2\frac{3}{4}) + \frac{6}{10} \cdot (-\frac{2}{6}\log_2\frac{2}{6} - \frac{4}{6}\log_2\frac{4}{6})) = 0.1245 \\
H_{win} &= 1 - (\frac{4}{10} \cdot (-0 \cdot \log_2 0 - \log_2 1) + \frac{6}{10} \cdot (-\frac{1}{6}\log_2\frac{1}{6} - \frac{5}{6}\log_2\frac{5}{6})) = 0.6099 \\
H_{debt} &= 1 - (\frac{5}{10} \cdot (-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}) + \frac{5}{10} \cdot (-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5})) = 0.0290 \\
H_{home} &= 1 - (\frac{7}{10} \cdot (-\frac{4}{7}\log_2\frac{4}{7} - \frac{3}{7}\log_2\frac{3}{7}) + \frac{3}{10} \cdot (-\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3})) = 0.0348
\end{aligned}
$$

> Based on the information gain calculated above, the “win” keyword attribute needs to be selected since it results in the largest information gain.

ii) The decision tree might grow too deep due to overfitting so that the depth is the result of trying to fit the training data perfectly, but it will not generalize well for unseen test data. Therefore, one might want to prune the decision tree to overcome overfitting. A validation set can be useful in determining whether the branch needs to be pruned: if using the validation set, the performance (e.g. accuracy) improves after pruning, then we choose to prune the branch; if the performance drops then we do not prune the branch.

1c. i) Complete the table above

| $i$ | $x^i$ | $y^i$ | $d(x^{(q)}, x^{(i)})$ | $w_q^i$ |
|---|---|---|---|---|
| 1 | 1.5 | 3.16 | 2.7 | 0.3704 |
| 2 | 2.3 | 1.45 | 1.9 | 0.5263 |
| 3 | 3.0 | 1.07 | 1.2 | 0.8333 |
| 4 | 3.8 | 2.01 | 0.4 | 2.5 |
| 5 | 4.9 | 4.51 | 0.7 | 1.4286 |

ii) Predict the output y(q) for x(q) = 4.2 using the k-nearest neighbours regression algorithm with d(x(q), x(i)) as its distance measure, and assuming k = 3. Show your calculation.

We pick the first 3 nearest neighbours and perform the calculation of their mean value:

$$y^{(q)} = \frac{1}{3}(2.01 + 4.51 + 1.07) = 2.53$$

iii) Now predict the output y(q) for x(q) = 4.2 using the locally weighted k-nearest neighbours regression algorithm. Use k = 3, the distance measure d(x(q), x(i)), and the weights w(i). Show your calculation.

Assuming k = 3, we pick the nearest three data points and calculate the weighted mean as the output value $y^{(q)}$.

$$y^{(q)} = \frac{1}{3}(2.5 * 2.01 + 1.4286 * 4.51 + 0.8333 * 1.07) = 4.1198$$

2a. Write out the necessary calculations for updating both the connection weights and bias weights in the last layer using gradient descent.

In order to update the connections weight and bias weights in the last layer using gradient descent, we need to calculate both $\frac{\partial L}{\partial W^{[2]}}$ and $\frac{\partial L}{\partial b^{[2]}}$. There calculation is as follows

$$\frac{\partial L}{\partial W^{[2]}} = \frac{\partial L}{\partial \hat{Y}} \cdot \frac{\partial \hat{Y}}{\partial Z^{[2]}} \cdot \frac{\partial Z^{[2]}}{W^{[2]}}$$
$$\frac{\partial L}{\partial b^{[2]}} = \frac{\partial L}{\partial \hat{Y}} \cdot \frac{\partial \hat{Y}}{\partial Z^{[2]}} \cdot \frac{\partial Z^{[2]}}{b^{[2]}}$$

In the calculation above, we calculate the term $\frac{\partial L}{\partial \hat{Y}}$ by first calculating the derivative of the loss function and then plug in the individual data points from $X$, $\hat{Y}$ and $Y$, which is $\sum_{i=1}^{N}(\hat{y}^i - y^i)x^i$. Then we calculate the term $\frac{\partial \hat{Y}}{\partial Z^{[2]}}$ by finding the derivative of sigmoid function, which is $g(z) = 1 - g(z)^2$. We then apply element-wise multiplication on the term produced by $\frac{\partial L}{\partial \hat{Y}}$ and $\frac{\partial \hat{Y}}{\partial Z^{[2]}}$. The term $\frac{\partial Z^{[2]}}{\partial W^{[2]}}$ simply stands for $W^{[2]}$ since we know that $Z^{[2]} = A^{[1]}W^{[2]} + b^{[2]}$

Simply procedure follows for the calculation of $\frac{\partial L}{\partial b^{[2]}}$

2b. Overfitting is the phenomenon that during the training process, the model fits well for the training dataset but it does not perform well on unseen test dataset. The model has memorized certain patterns in the training dataset and does not generalize well for unseen data. Three methods to deal with overfitting: reduce the complexity of the model, perform dropout (e.g. randomly set 50% of the neuron weights to 0), early stop (simply stop the training process earlier than before)

2c. i) Construct the confusion matrix for this output

| | Predicted Real | Predicted Fraud |
|---|---|---|
| Actual real | 8 | 1 |
| Actual Fraud | 3 | 2 |

ii) Calculate the classification accuracy, along with precision, recall and F1 for both classes.

Classification accuracy is $\frac{8+2}{14} = 0.7143 = 71.43\%$

For the Real category:

$$P_{real} = \frac{8}{8+3} = 0.7272$$
$$R_{real} = \frac{8}{8+1} = 0.8889$$
$$F_{1,real} = \frac{2 \cdot P_{real} \cdot R_{real}}{P_{real} + R_{real}} = \frac{2 \cdot 0.7272 \cdot 0.8889}{0.7272 + 0.8889} = 0.7999$$

For the Fraud category

$$P_{fraud} = \frac{2}{2+1} = 0.6667$$
$$R_{fraud} = \frac{2}{3+2} = 0.4$$
$$F_{1,fraud} = \frac{2 \cdot P_{fraud} \cdot R_{fraud}}{P_{fraud} + R_{fraud}} = \frac{2 \cdot 0.6667 \cdot 0.4}{0.6667 + 0.4} = 0.5000$$

iii) Analyse the results. Are there any issues? If so, which metrics identify them?

There are two issues:

The first issue is that the model cannot correctly recognize fraudulent transactions, which is indicated by the low recall rate for the fraud class

The second issue is that the dataset is imbalanced: there are almost twice as many real examples as the fraud examples. This is indicated by the huge difference between the F1 scores of the two classes

3a. Using the Euclidean distance measure, we can calculate the distance between $x^{(i)}$ and each centroid

$$
\begin{aligned}
d_1 &= \sqrt{(-2+3)^2 + (0+1)^2} = \sqrt{2} \\
d_2 &= \sqrt{(-2-1)^2 + (0-2)^2} = \sqrt{13} \\
d_3 &= \sqrt{(-2+4)^2 + (0-1)^2} = \sqrt{5}
\end{aligned}
$$

Since $x^{(i)}$ is closest to the first centroid, it should be assigned to the first cluster.

3b. From the given parameters, we can calculate $p(0|\theta)$ as follows

$$
\begin{aligned}
p(0|\theta) &= 0.5 \cdot \mathcal{N}(0|-2,1) + 0.3 \cdot \mathcal{N}(0|1,4) + 0.2 \cdot \mathcal{N}(0|4,0.25) \\
&= 0.5 \cdot \frac{1}{\sqrt{2\pi}} e^{-2} + 0.3 \cdot \frac{1}{2\sqrt{2\pi}} e^{-\frac{1}{8}} + 0.2 \cdot \frac{1}{\sqrt{2\pi}} e^{-32} \\
&= \frac{1}{\sqrt{2\pi}} (0.5e^{-2} + 0.15e^{-\frac{1}{8}} + 0.2e^{-32})
\end{aligned}
$$

3c. When developing a neural network, which activation function and loss function would you use in the output layer for the following applications? Justify your decisions.

i) Predicting the temperature for tomorrow, based on the weather today.

Activation: Linear

Loss: MSE

Reason: This is clearly a regression problem with no bounded output value limit, thus a linear activation is appropriate and MSE should be used for the regression task

ii) Generating text by predicting the next word in the sequence based on the previous words.

Activation: Softmax

Loss: Cross Entropy

Reason: This is a multi-class classification problem (e.g. each word is a class and we are trying to classify the next word) and thus softmax activation + cross entropy loss is appropriate.

iii) Detecting whether the camera image from a self-driving car contains a stop sign.

Activation: Sigmoid/tanh

Loss: Binary Cross Entropy

Reason: This is a binary classification problem where the output layer will produce either yes or no; thus, sigmoid or tanh function (or any function behave like those two) can be used. Binary cross entropy should be used since it's a classification problem

3d. Hyperparameters are something that we need to deal with when designing machine learning models.

i) Explain what hyperparameters are and give 2 examples.

Hyper-parameters are the configurations of a machine learning model and those that control the learning process. For instance, the value K in the KNN algorithm and the learning rate in the neural networks are both examples of hyper-parameters.

ii) Given a dataset of 10,000 datapoints, how would you use it to find good hyperparameters?

First shuffle the data points so that they are not ordered according to any attributes. Then we perform cross-validation to choose the best hyper-parameters: divide the dataset into 10 equal folds, then select one fold for validation and one fold for test, then the rest is used for training. Run the entire process 10 times to cover different folds. Select the model with best performance and use the set of hyper-parameters associated with it. (This is not a very comprehensive process description so you guys can check the process for hyper-paraeter runing and cross-validation in the slides)