**Cognitive Science Questions**

**Part 1**

1. Compare the errors of MLP and linear regression for the exponential data and explain in your own words why it makes sense that the MLP error is lower than the linear error. Compare the prediction of the MLP and the linear regression models or have a look at the definitions of both models to make your argument. (2 Points)

I have observed that the Linear regression yields more errors than the Multilayer Perceptron.

Linear Regression is a linear model for predicting classifications on datasets while the Multilayer Perceptron (MLP) is exponential (non-linear). Since the dataset we were presented with is non-linear, it is not surprising that the Linear Regression yielded a higher error (Linear Regression works best with linear data). Since the Multilayer Perceptron is exponential, it is therefore more flexible and is able to manoeuvre towards the outlier data. No amount of training (or how well-trained the linear model is) can make the linear model shaped more like the exponential (MLP) model, making it less flexible and more prone to errors.

1. Have a look at the errors and predictions for both linear and exponential data for MLP and linear regression.

Would you expect both models to perform equally well for some data? What kind of data should be well modeled by a linear regression and MLP alike? (1 point)

The Linear Regression and Multilayer Perceptron could both perform equally well on noiseless linear data but not for anything else, because Linear Regression works best with linear data (dataset has very few outliers and are clustered in a linear form), whilst the Multilayer Perceptron is much more flexible can work with of datasets of all shapes and sizes.

1. From our error plots and the predictions it seems as if MLP always does an equal or better job than linear regression. Does that mean that we should always use MLPs?

There are many different forms of MLP or deep neuronal networks and many different techniques to train them. For this question only consider the MLPs encountered in the lecture or the labs. (3 points)

The Multilayer Perceptron is able to learn non-linear models, hence making them much more flexible and powerful than the Linear Regression. However, if you have a relatively linear dataset, it would be more efficient to use Linear Regression because it is faster (MLP has to work with more data and try to fit the data into the model), remains consistent and takes up less memory.

**Part 2**

1. Briefly discuss the classification results.

Train = [1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 0. 0.

0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Predict = [1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 0. 0.

0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Mean Squared Error = 0.0

Mean Accuracy = 1.0

The Mean Squared Error is 0.0.

We have trained the model from a set of data. When we tested the model with same set of data that we trained it with, it gives us an MSE of 0.0 because the model had learned from the data.

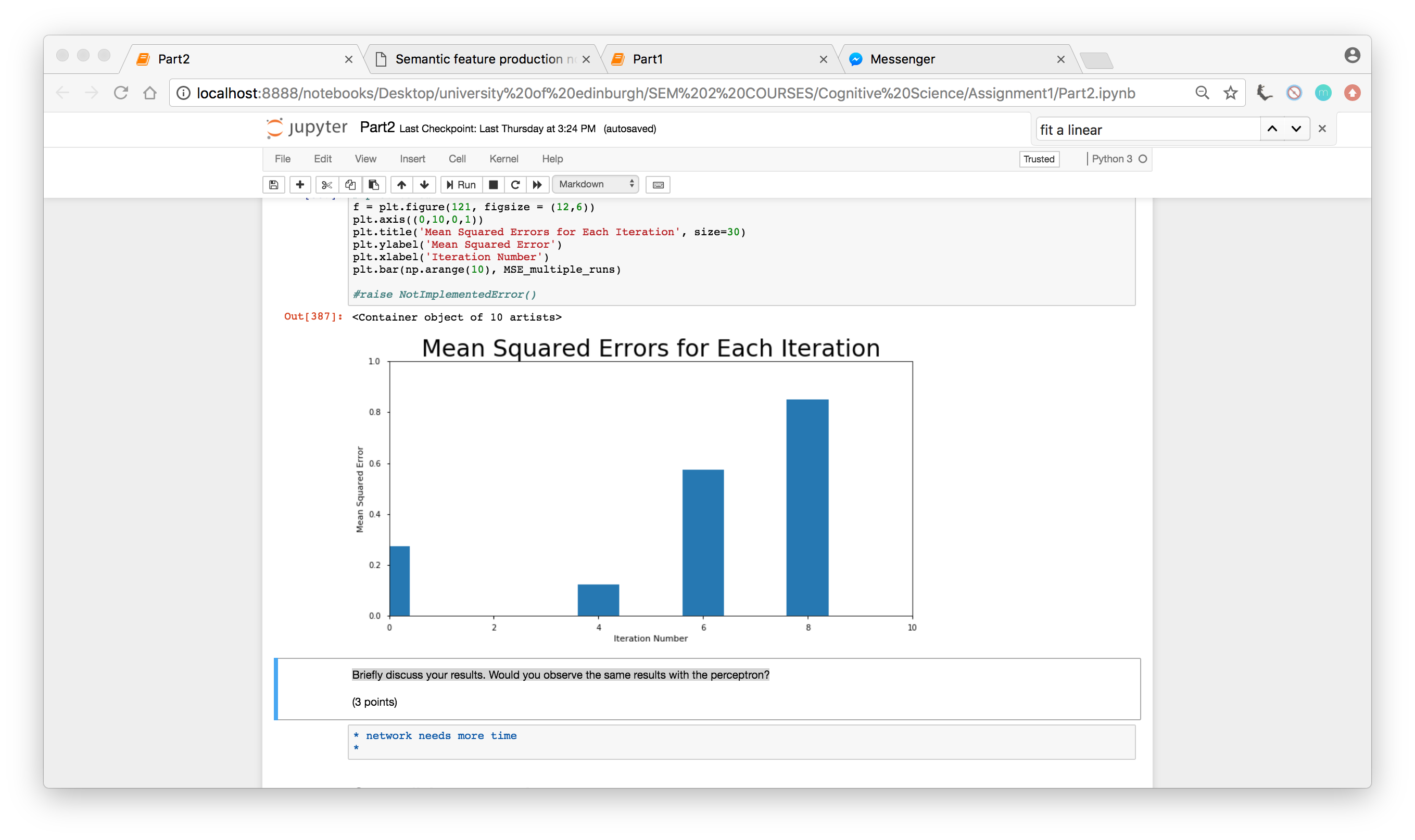
1. Did the model learn the correct classification with the set parameters? Discuss.

MSE Result = 0.0

The Mean Squared Error is 0.0.

This means that the set parameters have allowed the model to learn and understand the dataset precisely. Therefore, when we trained it with the same data it learned with, the model was able to learn the features of the animals and classify them correctly into their respective categories with no errors.

1. Briefly discuss your results. Would you observe the same results with the perceptron?



The data we are given is linearly separable, hence the perceptron is able to classify most of the animals correctly. In the exceptions where the model misclassified the data (with the animals dolphin, platypus, and bat), these data can be considered borderline data (it lies on the line that separates 'birds' and 'mammals'), and if more features were added to the feature weights, the model would be able to learn and understand the dataset better and correctly classify the data.

The Multilayer Perceptron takes more time to train compared to the Perceptron. Training the Multilayer Perceptron is set by some parameters, like the number of iterations and the learning rate, for example. In our case, we were only allowed a certain number of iterations. It may be possible that this number of iterations is not enough for the Multilayer Perceptron to learn completely. Therefore I think that the Multilayer Perceptron would yield slightly more errors.

1. Discuss the results briefly (one or two sentences).

MSE = 0.1

The Mean Squared Error is equal to 0.1.

The Mean Squared Error is not equal to 0 because we fitted the model without retraining it. In addition, it is more prone to error because the model is dealing with data it has not trained with.

1. Discuss the animals that the model failed to classify correctly using the learned weights of the model and the features of the misclassified animals.

(5 points)

Animals; Dolphin, Platypus, Bat

The weights learned by the perceptron displayed by net1.coef\_ refers to a 'bird' when it is a positive value, and refers to a 'mammal' when it is a negative value. Comparing the feature weights of the animals the model failed to classify, we can see that each animal contains a feature that is either not distinctive to either category or share features from both categories.

Dolphin

The animal 'dolphin' only has one defining feature which is 'eats'. The weight of 'eats' in the perceptron is 0.02886248, which is a relatively small positive number. This means that the feature 'eats' is not a clearly distinguishable feature (which means that 'eats' is a feature that is most probably commonly shared by most of the animals, both birds and mammals). However, since our Perceptron is linear, it is not quite flexible. Hence, despite the fact that 0.02886428 is a really small positive number, it is still a positive number, therefore the feature 'eats' is classified into a 'bird' feature. Since 'eats' is the only defining feature the animal 'dolphin' has, the Perceptron immediately classifies a 'dolphin' as a 'bird'.

Platypus

The animal 'Platypus' also only has one defining feature which is 'has a beak'. The weight of 'has\_a\_beak' is a positive number, and also one of the highest positive weight. This means that the property 'has\_a\_beak' is distinctive for the animal 'bird' (this feature is not commonly shared amongst mammals and is commonly shared amongst birds). Since this is the only defining feature for 'Platypus' and the Perceptron being linear, it immediately classifies 'Platypus' as a 'bird'.

Bat

The animal 'bat' is associated with both features from 'bird' (flies, has\_wings) and 'mammal' (has\_fur, is\_small). Hence the perceptron is unable to classify to which group does the animal 'bat' fall in.

1. Now discuss the MSEs, the number of erroneous classifications, and the differences/similarities between animals the network was not able to classify.

Result: The single layer perceptron has a higher mean squared error than the multilayer perceptron.

The Mean Squared Error for net2 is 0.0667 which is lower than the MSE for net1 (0.1). This means that the predictions made by net2 are closer to the test data, hence implying that net2 is better at predicting data. The Perceptron is linear whilst the Multilayer perceptron is exponential, hence the weights learned by the Multilayer Perceptron is not the same as the perceptron. The MLP learns different weights since it has a different architecture.

The Multilayer Perceptron correctly classifies 'Dolphin' whilst the single layer perceptron doesn't because the single layer perceptron isn't flexible enough to accommodate outlier data. So for the Perceptron to draw the best classification for the data, it has to learn that 'eats' is a bird feature.

Both models weren't able to distinguish 'Platypus' and 'Bat' correctly because their feature weights contain a specific weight that is a distinguishable feature of a bird (like how a platypus has a beak and how a bat is able to fly).

1. Discuss your observations briefly. (Reliabiity graph: predictions on training data is better than the test data)

I  have observed that there is a relationship between the errors in the training predictions and the corresponding test predicictions. When you do not have an error in the training predictions, the test predictions is with a constant low Mean Squared Error. However, if you have errors in the predictions of the training data, you should expect to yield a rather high Mean Squared Error for the test predictions.

The iterations with the low errors are the ones that misclassified bat and platypus, whilst the ones that yield a high error misclassified more than those two animals.

1. What is the best result you get on the training and test set, respectively? What were the corresponding parameter settings? Discuss your results and the effect of each parameter on the performance of the model.

You might want to have a look at Chapter 6 in "An Introduction to Neural Networks" (Gurney, 1997) for information.

(8 points)

It turns out that varying the parameters for the Multilayer Perceptron - such as the learning rate, number of hidden layers, number of units in the hidden layers, and the number of iterations - yields different Mean Squared Errors. By plotting a graph of each parameter change, I was able to see which settings allowed me to gain the best results (the ones with the least Mean Squared Error).

Learning Rate Variation

From the graph it is evident that the Learning Rate that yields the least Mean Squared Error is 0.1, with a Mean Squared error of 0.0 for Training data and around 0.06 for the Test data. Although it is true that a smaller learning rate yields more accuracy, the dataset we are working with is relatively small, hence a learning rate of 0.1 is enough for the model to converge to the desired minima.

Hidden Layer Variation

The least standard error is possible with a single hidden layer with a number of units of 100. A possible explanation to this is that the dataset we are working with is relatively small and not complex, hence there is no need to use more than one hidden layer.

Iteration Variation

The number of iterations that gives the best result from the graph is 100. However that was when the learning rate was se to 0.08. In the optimization graph I coded in the cell above, I set the learning rate to 0.1 and changed the value of the maximum number of iterations (max\_iter) from 20, 100, 500, 1000, and 5000, and found out that the least Mean Squared Error is obtained when the maximum number of iterations is 100. A small number of iterations (e.g. 20) is not enough for the Perceptron to learn the data. However, it is unreasonable to use the higher number of iterations (e.g. 500, 1000, and 5000) when the best result can be obtained with just 100 iterations.