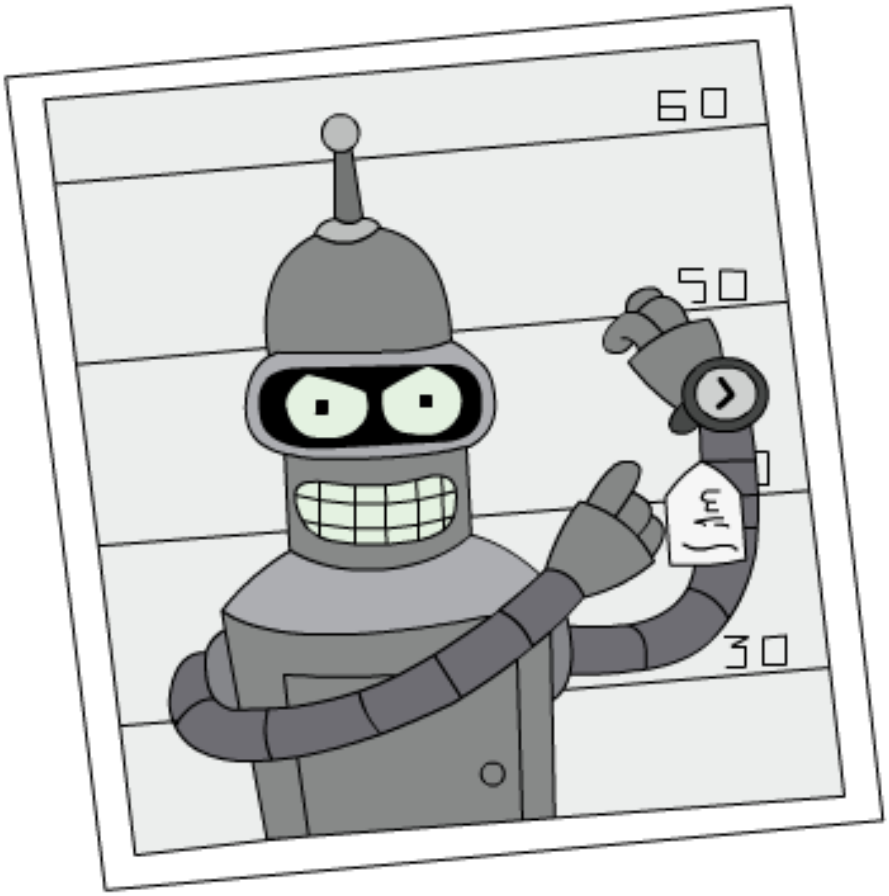
ECE4179 - Deep Learning and Neural Networks

Assignment 1

Due: 20/08/2021, 6:00pm



General Comments.

- Please submit your report along with the code (Jupyter notebook is preferred) as a single zip file.
- Include your name and student number in the filename for both the zip file and PDF. Do not send doc/docx files.
- Include your name and email address on the report and use a single column format when you prepare your report.
- Please ensure all of your results are included in your report. Marking is based on what is in the report. This includes plots, tables, code screen-shots etc.
- Make sure you answer questions in full and support any discussions with relevant additional information/experiments if necessary.

Late submission. Late submission of the assignment will incur a penalty of 10% for each day late. That is with one day delay, the maximum mark you can get from the assignment is 90 out of 100, so if you score 99, we will (sadly) give you 90. Assignments submitted with more than a week delay will not be assessed. Please apply for special consideration for late submission as soon as possible (, documented serious illness).

Note from ECE4179/5179 Team. The nature of assignments in ECE4179/5179 is different from many other courses. Here, we may not have a single solution to a problem. Be creative in your work and feel free to explore beyond questions. Creativity will be awarded by bonus points.

Good Luck

Question:	1	2	3	4	Total
Points:	10	35	40	15	100
Score:					

1. [10 points] Would it be possible to train a perceptron using a variant of the perceptron training algorithm in which the bias weight is left unchanged, and only the other weights are modified?

---

2. Recall that for the logistic regression, the cross entropy loss is defined as:

$$\mathcal{L}_{\text{CE}}(\boldsymbol{w}) = -\frac{1}{m}\sum_{i=1}^{m}\left\{ y_i \log\Big(\sigma(\boldsymbol{w}^\top \boldsymbol{x}_i)\Big) + (1-y_i)\log\Big(1-\sigma(\boldsymbol{w}^\top \boldsymbol{x}_i)\Big)\right\}. \tag{1}$$

Here, $\sigma(z) = 1/\big(1+\exp(-z)\big)$ denotes the sigmoid function, and $(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_m, y_m)$ with $\boldsymbol{x}_i \in \mathbb{R}^n, y_i \in \{0,1\}$ showing m training samples. The gradient of the cross entropy loss wrt $\boldsymbol{w}$ can be written as

$$\nabla_{\boldsymbol{w}} \mathcal{L}_{\text{CE}} = \frac{1}{m}\sum_{i=1}^{m}\Big(\sigma(\boldsymbol{w}^\top \boldsymbol{x}_i) - y_i\Big)\boldsymbol{x}_i\,. \tag{2}$$

Use the starter code provided as a Jupyter notebook and implement the following functions. Pay attention to the comments and guidelines in the starter code

2.1. [5 points] a function to compute the sigmoid.

2.2. [15 points] a function to compute gradient and cost of the logistic regression

2.3. [10 points] complete the for loop that performs the gradient descent algorithm

2.4. [5 points] function that predicts the class of its inputs according to the parameters of the logistic model

---

3. Load the data file called "toy_data.npz" and use the training data (X_train, y_train) to train your logistic model. Note that each sample $\boldsymbol{x}_i$ is a point in $\mathbb{R}^2$ and each label is $y_i \in \{0,1\}$. Also note that samples are bounded in $\boldsymbol{x}_i \in [-5,5]\times[-5,5]$ (meaning that each feature is within the range [-5,5]).

In your report,

3.1. [10 points] discuss the effect of the learning rate parameter in your model. Be specific here and discuss in what range, the model performance is acceptable, whether your model becomes unstable when changing the learning rate, or if you have observed slow convergence. In your report, provide the loss curves and learning rates for the following cases;

1. if your model becomes unstable
2. if your model converges very slowly
3. if your model converges nicely.

3.2. [10 points] report the accuracy of your model on the test data (X_test, y_test). Here you need to use the predict function and see what fraction of the test samples your model predicts correctly.

3.3. [10 points] compare the solution with and without the bias term in your model. Recall that to have a bias, you need to augment your samples with "1".

3.4. [10 points] plot the decision boundary of your model for points $[-5,5]\times[-5,5]$. Here you need to scan all the points (with a reasonable gap) in the region $[-5,5]\times[-5,5]$ and check the prediction of your model. Then you can plot the results to visualize how your model predicts points in the input region.

Hint: You can use the "meshgrid" function from Numpy to create a grid of points. Also, the function "contourf" from Matplotlib will come handy for plotting the decision boundary.

---

4. Load the data file called "toy_data_two_circles.npz" and use the training data (X_train, y_train) to train a logistic model. Note that each sample $\boldsymbol{x}_i$ is a point in $\mathbb{R}^2$ and each label is $y_i \in \{0, 1\}$.

   In your report,

   4.1. [5 points] report the accuracy of your model on the test data (X_test, y_test).

   4.2. [10 points] Your friend suggests that to better classify this data, we need to use nonlinear features. In particular, (s)he suggests to map a sample $\mathbb{R}^2 \ni \boldsymbol{x} = (x_1, x_2)^\top$ to $\mathbb{R}^5 \ni \hat{\boldsymbol{x}} = (x_1, x_2, x_1^2, x_2^2, x_1 x_2)^\top$. Use the above mapping and then train a new logistic model. Compute the accuracy of the resulting model and compare it with the model from part 4.1. Plot the decision boundaries and compare it with the model in part 4.1.