Here are some extra practice problems taken from past semesters. You are welcomed to check your solutions with me or discuss them in the telegram group chat.

1. (AY 24/25 Sem 1 Final Exam) You are minimizing the cost function $J(w) = \frac{1}{2}w^2 + 4w$ using gradient descent, what is the **largest** integer learning rate α that can be used so that it always finds the optimal value regardless of the initial value?

   Solution:
   We first calculate its derivative: $\frac{\partial J(w)}{\partial w} = w + 4$. Hence, the update rule of gradient descent would be $w \leftarrow w - \alpha(w + 4)$. We then get $w \leftarrow (1 - \alpha)w - 4\alpha$.

   The value w that minimizes $J(w)$ can be derived by setting the partial derivative $\frac{\partial J(w)}{\partial w} = w+4$ to 0, i.e. $w = -4$.

   When $\alpha = 1$, the update rule simplifies to $w \leftarrow -4$ so it gives the optimal solution in one shot. Therefore, $\alpha = 1$ clearly works.

   When $\alpha = 2$, the update rule simplifies to $w \leftarrow -8 - w$. Considering the initial value $w = 0$, gradient descent with proceed with $w = 0 \rightarrow -8 \rightarrow 0 \rightarrow -8 \rightarrow \cdots$. It does not converge in this case.

2. (AY 24/25 Sem 1 Final Exam) Consider a logistic regression model for multi-class classification with three classes: Pizza, Burger, and Sushi. The weight vectors for our multi-class (One vs One) classifiers where the $h_{\text{A/B}}(x)$ represents the probability of the class A. The weight vectors for each classifier include the bias term as the first element in each weight vector (2 is the bias for $w_{\text{Pizza/Burger}}$).
   $w_{\text{Pizza/Burger}} = \begin{bmatrix} 2 & -0.5 & 0.3 \end{bmatrix}$
   $w_{\text{Sushi/Pizza}} = \begin{bmatrix} -1 & 0.2 & -0.4 \end{bmatrix}$
   $w_{\text{Burger/Sushi}} = \begin{bmatrix} 0 & 0.4 & 0.1 \end{bmatrix}$
   Given the input $\begin{bmatrix} 3 & 2 \end{bmatrix}$, determine which class the model predicts. Justify your answer.

   Solution:
   For Pizza/Burger,
   - $\boldsymbol{w} \cdot \boldsymbol{x} = \begin{bmatrix} 2 & -0.5 & 0.3 \end{bmatrix}^\top \cdot \begin{bmatrix} 1 & 3 & 2 \end{bmatrix}^\top = 1.1$.
   - $h_{\boldsymbol{w}}(\boldsymbol{x}) = \frac{1}{1 + e^{-1.1}} = 0.75$.
   - The model predicts $P(\text{Pizza}|\boldsymbol{x}) = 0.75$ and $P(\text{Burger}|\boldsymbol{x}) = 0.25$. Pizza wins.

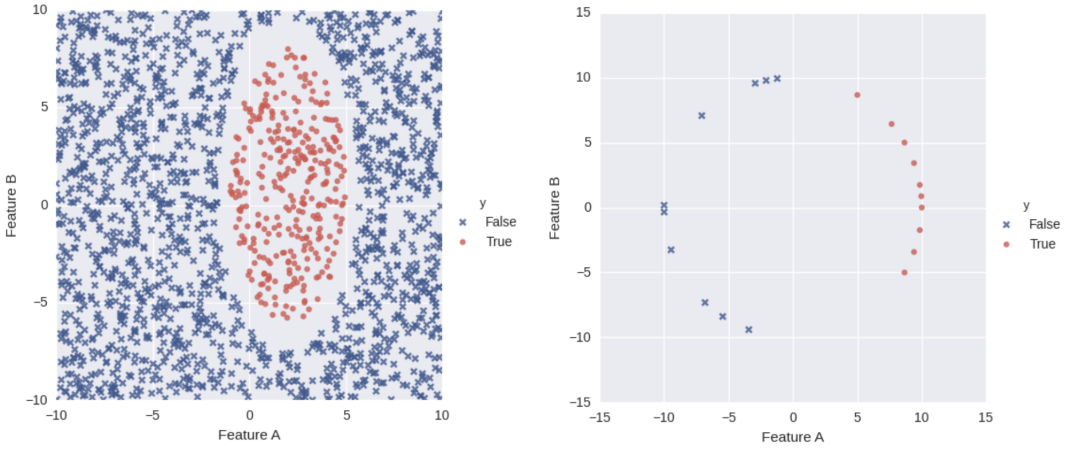
   For Sushi/Pizza,
   - $\boldsymbol{w} \cdot \boldsymbol{x} = \begin{bmatrix} -1 & 0.2 & -0.4 \end{bmatrix}^\top \cdot \begin{bmatrix} 1 & 3 & 2 \end{bmatrix}^\top = -1.2$.
   - $h_{\boldsymbol{w}}(\boldsymbol{x}) = \frac{1}{1 + e^{1.2}} = 0.23$.
   - The model predicts $P(\text{Sushi}|\boldsymbol{x}) = 0.23$ and $P(\text{Pizza}|\boldsymbol{x}) = 0.77$. Pizza wins.

   For Burger/Sushi,
   - $\boldsymbol{w} \cdot \boldsymbol{x} = \begin{bmatrix} 0 & 0.4 & 0.1 \end{bmatrix}^\top \cdot \begin{bmatrix} 1 & 3 & 2 \end{bmatrix}^\top = 1.4$.
   - $h_{\boldsymbol{w}}(\boldsymbol{x}) = \frac{1}{1 + e^{-1.4}} = 0.80$.
   - The model predicts $P(\text{Burger}|\boldsymbol{x}) = 0.80$ and $P(\text{Sushi}|\boldsymbol{x}) = 0.20$. Burger wins.

   Under the One vs One rule, Pizza wins twice and hence the classifier concludes Pizza.

   Note: You can avoid computing the sigmoid function by simply comparing whether $\boldsymbol{w} \cdot \boldsymbol{x}$ is positive/negative, since $\sigma(0) = 0.5$.

3. For the following two cases, define a minimal set of features that will perfectly classify the data. Here are some examples: $(A), (B), (AB, A^{10})$.



Solution:
For the first case, we can draw an ellipse to classify the positive and negative samples. The equation of the ellipse would be

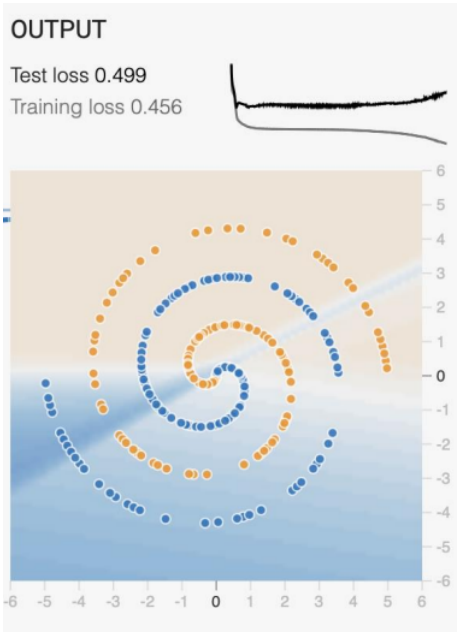
$$\frac{(A-x)^2}{a^2} + \frac{(B-y)^2}{b^2} = 1$$

where (x, y) is the center of the ellipse, a and b are scaling factors to turn the circle into an ellipse.

Expanding this, we get $\frac{1}{a^2}A^2 + \frac{1}{b^2}B^2 - \frac{2x}{a^2}A - \frac{2y}{b^2}B + \left(\frac{x^2}{a^2} + \frac{y^2}{b^2} - 1\right) = 0$, which implies that we need the feature set (A^2, B^2, A, B).

For the second case, drawing a vertical line at $A = 0$ already perfectly classifies the data (a value slightly larger than 0 might give a better separation – this is what SVMs are for, more in lecture 7). Therefore, the minimal set of features is (A).

4. (AY 24/25 Sem 1 Final Exam) Suppose you are training a classifier with stochastic gradient descent.



(a) Which evaluation metric is the **most** appropriate to evaluate the model's performance?

A. Accuracy

B. Precision and recall

C. Mean squared error

D. Weighted binary cross entropy loss

Solution:
It's important to differentiate between **evaluation metrics** and **loss functions** – evaluation metrics are used to judge the performance of a model (after the training is completed), so should only depend on the final predictions made by the model (the process, e.g. the output probabilities for logistic regression, should not matter). On the other hand, loss functions are used to train the model so they should be descriptive to the model – a wrong prediction can be penalized differently according to the output probability.

Options C and D are loss functions so they are not suitable here. Option B is more descriptive than Option A so it would be more appropriate.

(b) The result shown in the figure is after 1300 epochs with step size 0.03. Which advice(s) will you give to your colleague?

A. The test loss is too high, so they should use more test examples.

B. Keep running for more epochs.

C. Use a less complex model.

D. Use transformed features.

Solution:

A. Training examples are seen by the model, so having more training examples will benefit training. Test/validation samples are only used to validate the model's performance (determine if the model overfits) and the model cannot potentially benefit from them. So, increasing the amount of test samples do not improve the performance of the model – as the test samples are drawn from the population, the test loss will likely be similar even after we increased test samples.

B. The model should fit the training data as much as possible first (if it overfits, we should use a less complex model/hypothesis). Since the training loss still shows a clearly decreasing trend, it will be a good idea to keep training for more epochs so that the model fits the training data better.
But the test loss starts increasing! Note that a slight difference between train loss and test loss is perfectly normal – the model cannot see the exact test data but it can see the training data, so it's expected to capture training data slightly better. We should try to fit the model to the training data first, then decide to use a less complex model if it overfits.

C. At this point, we cannot tell if overfitting or underfitting occurs. So the correctness of this option can not be determined with the given information.

D. The data is not linearly separable, so we should definitely try transformed features.