Saarland University, Department of Computer Science
Neural Network Assignment 2

Deborah Dormah Kanubala (7025906) , Irem Begüm Gündüz (7026821), Anh Tuan Tran (7015463)

December 6, 2022

Exercise 4.1

4.1.a

$$MSE = \frac{1}{m}\sum_{i=1}^{m}\left\|y^{(i)} - y^{(i)}\right\|^2$$

Let $\hat{y}^{(i)} = \theta^\top x_i$ Hence; MSE $= \frac{1}{m}\sum_{i=1}^{m}\left\|\theta^\top x_i - y^{(i)}\right\|^2$ We assume;

$$y^{(1)} = \theta^\top X_i + \epsilon_i \text{ where } \epsilon_i \sim N\left(0, \sigma^2\right)$$

$\mathbb{P}\left(\epsilon_i\right) = \frac{1}{\sigma\sqrt{2\pi}} e\left(\frac{-(x-\mu)^2}{m\sigma^2}\right)$

Also, we assume Samples are i.i.d and hence the likelihood function; $l(\theta) = \prod_{i=1}^{n}\left(\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{m\sigma^2}}\right)$

$$\log l(\theta) = \log\left[\prod_{i=1}^{n}\left(\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\pi)^2}{m\sigma^2}}\right)\right.$$

$$= \log\left[\left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n\right] + \log\left[e^{-\sum_{i=1}^{n}}\left(\frac{-\left(\sigma^T x_i - y_i\right)}{m\sigma^2}\right)\right]$$

$$n\log\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \sum_{i21}^{n}\frac{1}{m\sigma^2}\left(\theta^\top x_i - y_i\right)$$

Interested in θ and maximising the problem, we take argmax.

$$\underset{\theta}{\operatorname{argmax}}\left(\sum_{i=1}^{n}\frac{1}{m}\frac{\left(\theta^\top x_i - y_i\right)^2}{\sigma^2}\right)$$

NB: Rule

$$\underset{\theta}{\operatorname{argmax}} f(\theta) = \underset{\theta}{\operatorname{argmax}} f(\theta)\alpha, \forall\alpha \in R+^*$$

also,
$\max f(\theta) = -\min(-f(\theta))$
Then we have;

$$\frac{1}{m}\sum_{i=1}^{n}\left(\theta^{\top}x_i - y_i\right)^2$$

but $\hat{y} = \theta^T x_i$

$$\Rightarrow \frac{1}{m}\sum_{i=1}^{1}\left(\hat{y} - y_i\right)^2 \gtreqless MSE$$

4.1.b

$$\Pr(x = x \mid \lambda) = \frac{\lambda^x e^{-\lambda}}{x!}$$
$$L\left(\lambda; x_1, x_2 \ldots, x_n\right) = \prod_{i=1}^{n} \exp(\Pr(x = x \mid \lambda))$$
$$= \prod_{i=1}^{n} \exp(-\lambda)\frac{1}{x!}\lambda^x$$

By taking the natural log, the log-likelihood becomes;

$$\begin{aligned}\ell\left(\lambda : x_1, x_2 \ldots, x_n\right) &= \ln\left(\prod_{i=1}^{n} \exp(-\lambda)\frac{1}{x!}\lambda^x\right)\\ &= \sum_{i=1}^{n} \ln\left(\exp(-\lambda)\cdot\frac{1}{x!}\lambda^x\right)\\ &= \sum_{i=1}^{n}[-\lambda - \ln(x!) + x\ln(\lambda)]\\ &= -n\lambda - \sum_{i=1}^{n}\ln(x!) + \ln(\lambda)\sum_{i=1}^{n} x\end{aligned}$$

MLE seeks the parameter that would maximize the likelihood of the date. Hence, we take the derivative and set it to 0.

$$\frac{\partial}{\partial\lambda}\left(-n\lambda - \sum_{i=1}^{n}\ln(x!) + \ln(\lambda)\sum_{i=1}^{n} x\right).$$
$$= -n + \frac{1}{\lambda}\sum_{i=1}^{n} x$$

Taking the first derivative to be equal to 0, we have:

$$\lambda^* = \frac{1}{n}\sum_{i=1}^{n} x$$

* $\therefore$ MLE of of parameter λ is just sample of n observation

Exercise 4.2

4.2.a

With a single bad prediction, the error is squared which therefore magnifies the error. Also, it is sensitive to outliers since the presence of outliers means there is a high chance of it being a high value and this therefore can also lead to high errors. In using MSE for binary classification problems, we will not be able to generalize to more than 2 classes. We would find some values will be out of the interval $[0, 1]$ hence making it impossible to interpret as probabilities. Also, using MSE will impose undesired ordering and distances on classes, and finally masking will occur. We will also have no guarantee to get to the minimum of the function as MSE is non-convex for binary classification problems.

4.2.b

Other methods for unsupervised learning include but are not limited to the following:

- **MDS:** Projects high-dimensional data to a lower-dimensional space on a local structure. It achieves this by defining a stress function to preserve the pairwise distance. MDS preserves the local structure of the data, unlike PCA. Nevertheless, this requires only one symmetric matrix.

- **tSNE:** This method embeds points from a higher dimension into a lower dimension while preserving the neighborhood points. It aims to retain the local structure of the data by minimizing the Kullback–Leibler divergence between the two distributions with respect to the locations of the points in the map. The problem with this is that it has a slow execution time due to the calculation of the pairwise conditional probabilities for each data point.

4.2.c

Temperature if applied to the logits to affect the final probabilities from softmax. If the temperature is below < 1 it makes the model more confident about its predictions whereas the temperature value > 1 makes the model less confident about its predictions. It might be useful to include temperature (T) when using RNN to produce text because doing that would facilitate the creation of more diverse and interesting text predictions. As one will need the model to be less confident by introducing a high value of T which would be helpful in producing non-repetitive and interesting text.

Exercise 4.3

Our function f:

$$J(x) = \frac{1}{1 + \exp\left(-\left(x_1^2 + x_2^2\right)\right)}$$

Update Rules:

$$x_1 = x_1 - \alpha \cdot \frac{d}{dx_1} \cdot J\left(x_1, x_2\right)$$

Derivatives:

$$\frac{d}{dx_1} J\left(x_1, x_2\right) = \frac{2x_1 e^{x_1^2}}{\left(e^{x_1^2} + 1\right)^2}$$

Final update rules for each value in X:

$$x_1 = x_1 - \alpha \cdot \frac{2x_1 e^{x_1^2}}{\left(e^{x_1^2} + 1\right)^2}$$

Each iteration is given as a row in the table below. Stopped at third iteration.

x_1	x_2	$f(x)$	$\nabla f(x_1)$	$\nabla f(x_1)$
1	-1	0.88	0.39	-0.39
0.92	-0.92	0.85	0.39	-0.39
0.84	-0.84	0.80	0.37	-0.37
0.77	-0.77	0.77	0.36	-0.36

If we increase the learning rate slightly higher value, then our gradients will be slightly more decreased. However, if we increase the learning rate much higher value, then the gradients will reach to zero in less iteration.