ECE C147/C247, Winter 2024 **Homework #2**
Neural Networks & Deep Learning Prof. J.C. Kao
UCLA ECE TAs: T. Monsoor, Y. Liu, S. Rajesh, L. Julakanti, K. Pang

Due Monday, 29 Jan 2024, by 11:59pm to Gradescope.
100 points total.

1. (10 points) **Noisy linear regression**

   A real estate company have assigned us the task of building a model to predict the house prices in Westwood. For this task, the company has provided us with a dataset $\mathcal{D}$:

$$\mathcal{D} = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(N)}, y^{(N)})\}$$

   where $x^{(i)} \in \mathbb{R}^d$ is a feature vector of the i^{th} house and $y^{(i)} \in \mathbb{R}$ is the price of the i^{th} house. Since we just learned about linear regression , so we have decided to use a linear regression model for this task. Additionally, the IT manager of the real estate company has requested us to design a model with small weights. In order to accommodate his request, we will design a linear regression model with parameter regularization. In this problem, we will navigate through the process of achieving regularization by adding noise to the feature vectors. Recall, that we define the cost function in a linear regression problem as:

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - {(x^{(i)})}^T\theta)^2$$

   where $\theta \in \mathbb{R}^d$ is the parameter vector. As mentioned earlier, we will be adding noise to the feature vectors in the dataset. Specifically, we will be adding zero-mean gaussian noise of known variance σ^2 from the distribution

$$\mathcal{N}(0, \sigma^2\mathbf{I})$$

   where $\mathbf{I} \in \mathbb{R}^{d\times d}$ and $\sigma \in \mathbb{R}$. With the addition of gaussian noise the modified cost function is given by,

$$\tilde{\mathcal{L}}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - (x^{(i)} + \delta^{(i)})^T\theta)^2$$

   where $\delta^{(i)}$ are i.i.d noise vectors with $\delta^{(i)} \sim \mathcal{N}(0, \sigma^2\mathbf{I})$.

   (a) (6 points) Express the expectation of the modified loss over the gaussian noise, $\mathbb{E}_{\delta\sim\mathcal{N}}[\tilde{\mathcal{L}}(\theta)]$, in terms of the original loss plus a term independent of the dataset $\mathcal{D}$. To be precise, your answer should be of the form:

$$\mathbb{E}_{\delta\sim\mathcal{N}}[\tilde{\mathcal{L}}(\theta)] = \mathcal{L}(\theta) + R$$

where R is not a function of $\mathcal{D}$. For answering this part, you might find the following result useful:

$$\mathbb{E}_{\delta \sim \mathcal{N}}[\delta\delta^T] = \sigma^2 \mathbf{I}$$

.

(b) (2 points) Based on your answer to (a), under expectation what regularization effect would the addition of the noise have on the model?

(c) (1 point) Suppose $\sigma \longrightarrow 0$, what effect would this have on the model?

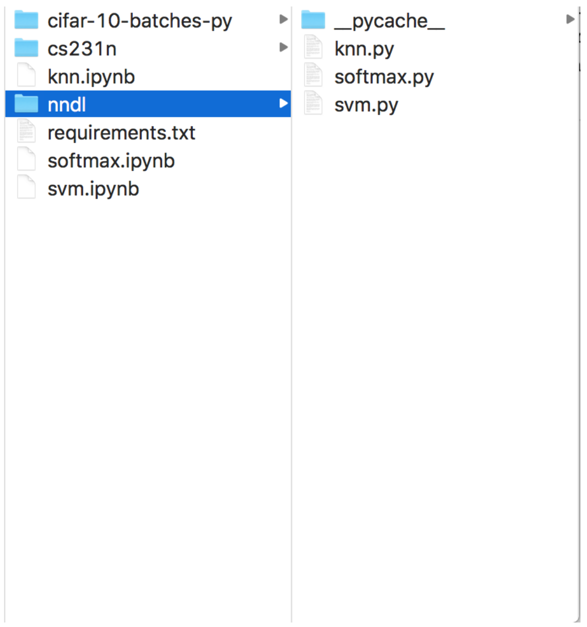
(d) (1 point) Suppose $\sigma \longrightarrow \infty$, what effect would this have on the model?

2. (20 points) ***k*-nearest neighbors.** Complete the k-nearest neighbors Jupyter notebook. The goal of this workbook is to give you experience with the CIFAR-10 dataset, training and evaluating a simple classifier, and k-fold cross validation. In the Jupyter notebook, we'll be using the CIFAR-10 dataset. Acquire this dataset by running:

```
wget http://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
tar -xzvf cifar-10-python.tar.gz
rm cifar-10-python.tar.gz
```

If you don't have `wget` you can simply go to: `https://www.cs.toronto.edu/~kriz/cifar.html` and download it manually.

We have attached a screenshot of the paths the files ought to be in, in case helpful (though it should be apparent from the Jupyter notebook).



Print out the entire workbook and related code sections in knn.py, then submit them as a pdf to gradescope.

3. (30 points) **Softmax classifier gradient.** For softmax classifier, derive the gradient of the log likelihood.

   Concretely, assume a classification problem with c classes

   - Samples are $(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(m)}, y^{(m)})$, where $\mathbf{x}^{(j)} \in \mathbb{R}^n$, $y^{(j)} \in \{1, \ldots, c\}$, $j = 1, \ldots, m$
   - Parameters are $\theta = \{\mathbf{w}_i, b_i\}_{i=1,\ldots,c}$
   - Probablistic model is

   $$\Pr\left(y^{(j)} = i \mid \mathbf{x}^{(j)}, \theta\right) = \text{softmax}_i(\mathbf{x}^{(j)})$$

   where

   $$\text{softmax}_i(\mathbf{x}) = \frac{e^{\mathbf{w}_i^T \mathbf{x} + b_i}}{\sum_{k=1}^{c} e^{\mathbf{w}_k^T \mathbf{x} + b_k}}$$

   Derive the log-likelihood $\mathcal{L}$, and its gradient w.r.t. the parameters, $\nabla_{\mathbf{w}_i} \mathcal{L}$ and $\nabla_{b_i} \mathcal{L}$, for $i = 1, \ldots, c$.

   Note: We can group $\mathbf{w}_i$ and b_i into a single vector by augmenting the data vectors with an additional dimension of constant 1. Let $\tilde{\mathbf{x}} = \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}$, $\tilde{\mathbf{w}}_i = \begin{bmatrix} \mathbf{w}_i \\ b_i \end{bmatrix}$, then $a_i(\mathbf{x}) = \mathbf{w}_i^T \mathbf{x} + b_i = \tilde{\mathbf{w}}_i^T \tilde{\mathbf{x}}$. This unifies $\nabla_{\mathbf{w}_i} \mathcal{L}$ and $\nabla_{b_i} \mathcal{L}$ into $\nabla_{\tilde{\mathbf{w}}_i} \mathcal{L}$.

4. (10 points) **Hinge loss gradient.**

   Owing to the drastic changes in climate throughout the world, a weather forecasting organization wants our help to build a model that can classify the observed weather patterns as severe or not severe. They have accumulated data on various attributes of the weather pattern such as temperature, precipitation, humidity, wind speed, air pressure, and geographical location along with severity of weather. However, the contribution of the attributes to the classification of weather as severe or not is unknown.

   We choose to use a binary support vector machine (SVM) classification model. The SVM model parameters are learned by optimizing a hinge loss. The company has provided us with a data-set

   $$\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \cdots, (\mathbf{x}^{(K)}, y^{(K)})\}$$

   where $\mathbf{x}^{(i)} \in \mathbb{R}^d$ is a feature vector of the i^{th} data sample and $y^{(i)} \in \{-1, 1\}$. We define the hinge loss per training sample as

   $$\text{hinge}_{y^{(i)}}(\mathbf{x}^{(i)}) = \max\left(0, 1 - y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b)\right) \tag{1}$$

   , where $\mathbf{w} \in \mathbb{R}^d$ and bias $b \in \mathbb{R}$ are the model parameters. With the hinge loss per sample defined, we can then formulate the average loss for our model as:

   $$\mathcal{L}(\mathbf{w}, b) = \frac{1}{K} \sum_{i=1}^{K} \text{hinge}_{y^{(i)}}(\mathbf{x}^{(i)}) \tag{2}$$

Find the gradient of the loss function $\mathcal{L}(\mathbf{w}, b)$ with respect to the parameters i.e $\nabla_{\mathbf{w}}\mathcal{L}$ and $\nabla_b\mathcal{L}$.

Hint: An Indicator function, also known as a characteristic function, takes on the value of 1 at certain designated points and 0 at all other points. Mathematically, we can represent it as follows:

$$\mathbb{1}_{\{p<1\}} = \begin{cases} 1, & \text{if } p < 1 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

5. (30 points) **Softmax classifier.** Complete the Softmax Jupyter notebook. Print out the entire workbook and related code sections in softmax.py, then submit them as a pdf to gradescope.