ECE521: Inference Algorithms and Machine Learning
University of Toronto
Assignment 1: k-NN and Linear Regression
Due date: January 20, *before* the tutorial

1 Introduction

The goal of this assignment is to become familiar with k-NN and linear regression algorithms as well as Python and NumPy. In particular, in this assignment, we will implement

- k-Nearest Neighbor algorithm that will be used for a classification task on a handwritten digit dataset.
- Linear Regression with backpropagation that will be used for fitting a function on an artificial dataset.
- Linear Regression with backpropagation and ℓ_2 weight decay that will be used for a classification task on a handwritten digit dataset.

2 Datasets

2.1 Artificial Dataset

The first dataset for this assignment is an artificial dataset with 100 training points that can be created as follows.

```
train_x = np.linspace(1.0, 10.0, num=100)[:, np.newaxis]
train_y = np.sin(train_x) +
          0.1 * np.power(train_x, 2) +
          0.5 * np.random.randn(100, 1)
```

2.2 Tiny MNIST Dataset

The second dataset that we use is a cropped and down-sampled version of a handwritten digit dataset called MNIST [1]. This dataset has 800 training images of digit 3 and 5 as well

[1] http://yann.lecun.com/exdb/mnist/

as 400 validation images. All the images are 8 by 8. You can load the dataset in numpy as follows.

```
with np.load("TINY_MNIST.npz") as data:
  x, t = data["x"], data["t"]
  x_eval, t_eval = data["x_eval"], data["t_eval"]
```

3 *k*-Nearest Neighbors

In this section, we will learn to implement the k-Nearest Neighbor (k-NN) algorithm on the Tiny MNIST dataset. We will also see the effect of the value of k as well as the training size for k-NN.

Task 1: Training Size

Set $k = 1$ and submit a 2-column table that lists the number of validation errors that you get versus the training size when the training size N is $N \in \{5, 50, 100, 200, 400, 800\}$ (Use the first N points in the complete training dataset). In one sentence, summarize your observation about the effect of N on performance and whether you think large N or small N is best.

Task 2: Overfitting and Underfitting

Submit a 2-column table that lists the number of validation errors that you get versus the values of $k \in \{1, 3, 5, 7, 21, 101, 401\}$. For this task, use the complete training set of $N = 800$. In one sentence, summarize your observation about the effect of k on performance and what the best value of k is.

4 Linear Regression

4.1 Artificial Dataset

In this section, we use linear regression to fit a function to the artificial dataset discussed in Section 2.1.

Task 3: Linear Fit

Use the linear regression algorithm to fit a line to the artificial dataset. Plot both the training data and the fitted line.

Task 4: Feature Space

We can fit a better function to the data points by introducing non-linearity in our model. One way to do so is to map each input x to a high dimensional feature space of $[1, x, x^2, x^3, x^4, x^5]$

and then train a linear regression on top of the feature space. Note that in this case, you might have to normalize each dimension of the feature space before training your model. Use the above feature space and plot both the training data and the fitted curve. In one sentence, summarize your observation about the effect of non-linearity on performance.

4.2 Tiny MNIST Dataset

In this section, we implement the linear regression algorithm and use it for classification on the Tiny MNIST dataset. The training targets of linear regression are either 0 or 1. Once the training is done, we can find the output of the trained network for the validation images and use binary thresholding to assign class labels to each validation point. We will be using the validation set to tune the hyper-parameter of the ℓ_2 weight decay. Use the following training cost function.

$$\text{Euclidean Cost} = \frac{1}{2N}\sum_{n=1}^{N}\{\mathbf{w}^T\mathbf{x}_n + b - t_n\}^2 + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w}$$

Task 5: Training Size

Use the first $N \in \{100, 200, 400, 800\}$ points of the complete training dataset and train a linear regression without regularization ($\lambda = 0$) using stochastic gradient descent. Submit a 2-column table that lists the number of validation errors for each N. In one sentence, summarize your observation about the effect of N on performance and whether you think large N or small N is best.

Task 6: Overfitting

Use the first $N = 50$ points of the complete training dataset and train a linear regression without regularization ($\lambda = 0$) using stochastic gradient descent. Plot the number of training errors and validation errors vs. the number of epochs.

Task 7: Regularization

Use the first $N = 50$ points of the complete training dataset and train a linear regression with regularization parameters of $\lambda \in \{0, 0.0001, 0.001, 0.01, 0.1, 0.5\}$ using stochastic gradient descent. Submit a 2-column table that lists the number of validation errors for each λ. In one sentence, summarize your observation about the effect of weight decay on performance and what the best value of λ is.