UNIVERSITY of GUELPH

School of Engineering

ENGG*6500: Introduction to Machine Learning
Fall 2021

Assignment 1
Due: Friday October 15, 2021

Notes:

- The questions require thought, but do not require long answers so try to be concise.
- You can consult with your colleagues or check for solutions so long as what is finally submitted is your individual interpretation and not a copy paste.
- For problems that require programming, please include in your submission a copy of your code (with comments) and any figures that you were asked to plot.
- You should write your own code and not copy from any resource on the internet. If you consult any resource online before writing your own code, you must cite the source, add a comment to your code with a link to the source you got it from.
- All students must submit an electronic PDF version.
- There will be a 5% penalty for submissions which are merely scanned versions of handwritten solutions.
- Submit the assignment pdf file in Assignment 1 dropbox of Courselink, before due date. (20% deduction per day for delayed submission).

Include a cover page indicating your name, title of work, course, instructor's name and date. Assignments will be judged on the basis of visual appearance, the grammatical correctness and quality of writing, and the visual appearance and readability of any graphics, as well as their contents. Please make sure that the text of your report is well-structured, using paragraphs, full sentences, and other features of a well-written presentation. Use itemized lists of points where appropriate. Text font size should be 12 point.

I. Theory

1. Define Machine Learning and name four types of problems where it excels.
2. What is happening if your model performs great on the training data but generalizes poorly to new instances? Name three possible solutions to this problem.
3. What is a test set and a validation set and what are their usage and purpose.
4. Let us say we are building an OCR and for each character, we store the bitmap of that character as a template that we match with the read character pixel by pixel. Explain when such a system would fail. Why do you think are barcode readers still used?
5. If a face image is a 100×100 pixel image, written in row-major, this is a 10,000-dimensional vector. If we shift the image one pixel to the right, this will be a very different vector in the 10,000-dimensional space. How can we build face recognizers robust to such distortions?
6. In estimating the price of a used car, it makes more sense to estimate the percent depreciation over the original price than to estimate the absolute price. Why?
7. In light of the training dataset size, when is a simpler model better and when is a complex model better?
8. What is the reason to have different train and test accuracies? What should you do if while experimenting in the real world and you find a substantial difference between the train and test accuracies?
9. In the KNN algorithm, explain how to select a K value that results a high performance of the classifier.
10. Does the Decision Tree Algorithm guaranteed to terminate? Explain your reasoning.

II. Math

On Gradients and Hessians:
Recall that a matrix $A \in \mathbb{R}^{n\times n}$ is symmetric if $A^T = A$, that is, $A_{ij} = A_{ji}$
Also recall that the gradient $\bigtriangledown f(x)$ of a function $f : \mathbb{R}^n \to \mathbb{R}$, which is the n-vector of partial derivatives. $\bigtriangledown f(x) = \begin{bmatrix} \frac{\partial}{\partial x_1} f(x) \\ \vdots \\ \frac{\partial}{\partial x_n} f(x) \end{bmatrix}$ where $x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$

The Hessian $\bigtriangledown^2 f(x)$ of a function $f : \mathbb{R}^n \to \mathbb{R}$ is the $n \times n$ symmetric marix of twice partial derivatives.

$$\bigtriangledown^2 f(x) = \begin{bmatrix} \frac{\partial^2}{\partial {x_1}^2} f(x) & \frac{\partial^2}{\partial x_1 \partial x_2} f(x) & \cdots & \frac{\partial^2}{\partial x_1 \partial x_n} f(x) \\ \frac{\partial^2}{\partial x_2 \partial x_1} f(x) & \frac{\partial^2}{\partial {x_2}^2} f(x) & \cdots & \frac{\partial^2}{\partial x_1 \partial x_n} f(x) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2}{\partial x_n \partial x_1} f(x) & \frac{\partial^2}{\partial x_n \partial x_2} f(x) & \cdots & \frac{\partial^2}{\partial {x_n}^2} f(x) \end{bmatrix}$$

1. Let $f(x) = \frac{1}{2} x^T A x + b^T x$, where A is a symmetric matrix and $b \in \mathbb{R}^n$ is a vector. What is $\bigtriangledown f(x)$

2. Let $f(x) = g(h(x))$, where $g : \mathbb{R} \to \mathbb{R}$ is differentiable and $h : \mathbb{R}^n \to \mathbb{R}$ is differentiable. What is $\bigtriangledown f(x)$

III. Programming

The objectives of the programming assignment include:
1. Gaining familiarity with programming in Python
2. Becoming familiar working with Jupiter notebook
3. Becoming comfortable with classification algorithms
4. Learning to work with data, choose appropriate variables

Getting started:

The recommended way of getting started follows. If you already have a great deal of experience with Python, you may already have a specific setup in place and should feel free to continue with this. If this is entirely new to you the simplest method is to visit this site:
http://jupyter.org/install.html
Here it is recommended that you install an up to date Anaconda distribution with Anaconda's latest stable Python 3 version. There are instructions for both Linux and Windows users.

One simple method of working on the notebook itself is to do so within your web browser. Launching an Anaconda prompt (if windows), and the jupyter notebook command will typically take you to the browser directly. From here, you can create new notebooks (*.ipynb file).
This is a relatively simple way of getting off the ground. For any missing packages, these can be installed at the prompt (or in shell) using pip.
There are IDEs to work with Python, and typically your bits of Python could be broken down into individual .py files, however the notebook will allow you to work interactively, and produce your assignment report in the same place as your code.
If you are looking for a richer experience, you could consider running jupyter lab (there is also the option of suppressing the web interface entirely if you look deeper).

Requirements:

1. Write a Python program to implement a KNN classifier. The implementation is according to the data points in the file Iris.xlxs which can be downloaded from CourseLink. Appropriately divide the data in the file to obtain a testing set. Find the accuracy of the tested data. Select two value of K. For each value, implement the code for two distance criteria. Compare your results and provide a detailed explanation.

2. Write a Python program to implement a decision tree classifier. The implementation is according to the data points in the file heart.xlxs which can be downloaded from CourseLink. Compute the generalization error rate of the tree.

   a. Plot the resulting decision tree based on Gini index computations
   b. Plot the resulting decision tree based on entropy computations

   A completed version of the .ipynb file should be submitted.