CAP4770 Assignment 1

- There is a written component for this assignment (P1). Please see a separate file named 'A1-written.pdf' for the written assignment. You need to do that first.
- Please turn in your written assignment as one pdf file. For your programming assignment, you can create a pdf file containing all your code and output. For instance, a pdf file generated from the Jupyter notebook will be great.

P0. Set up your Python Machine Learning Environment
(Nothing to turn in for this part)

Step 1: Install Python+Conda distribution.

Your choice is either Miniforge or Miniconda. If you use a Mac with the M1 chip, you should install Miniforge.

Step 2: Create a basic Python environment for projects in this course and install needed Python packages.

You should install the following Python packages to the newly created environment: matplotlib, numpy, pandas, scipy, and scikit-learn. For editing Python code and running Python interactively, jupyter notebook is recommended, and in that case, you will add jupyter to the above package list for installation.

Resources:
- The lecture slides contain many details that can guide the setup process.
- There is a great deal of Internet resources. Here are some of the videos and links I found useful in the past.

Warning: These are for information only. Some of the instructions may be out of date. For package installations, follow our lecture slides or search for the most recent instructions on the web.

For Mac computers with the M1 chip, I recommend the following two videos. You don't need to follow their steps to the end. But, the videos give you some idea about the big picture. For most of the projects in this course, we don't need Tensorflow and you don't have to install that for now.

1. Jeff Heaton, Mac M1 Monterey Installing Miniforge and Anaconda/Miniconda Side-by-Side
   https://www.youtube.com/watch?v=w2qlou7n7MA

2. Daniel Bourke, Setup Apple Silicon Mac for Machine Learning in 13 minutes (TensorFlow edition)
   https://www.youtube.com/watch?v=_1CaUOHhI6U

Other Introductory Resources on Internet:
David Chong, How I Set Up My MacBook Pro as A ML Engineer in 2022
https://towardsdatascience.com/how-i-set-up-my-macbook-pro-as-a-ml-engineer-in-2022-88226f08bde2

Zolzaya Luvsandorj, Introduction to Conda virtual environments
https://towardsdatascience.com/introduction-to-conda-virtual-environments-eaea4ac84e28

Machine Learning libraries (NumPy, SciPy, matplotlib, scikit-learn, pandas)
https://www.dotnetlovers.com/article/217/machine-learning-libraries-numpy-scipy-matplotlib-scikit-learn-pandas

P1. (30 points) Work on the written part of the assignment. See the file name 'A1-written.pdf'. You will need the solution for the programming part.

P2. (20 points) Load the data set named 'lin_df.csv' on Canvas. You can use DataFrame to load it. Check it out and you will see it contains two columns of data. The first column contains input X. The second column contains output Y. You will use the entire data set as the training set. In other words, we don't worry about generalization in this exercise.

(a) Plot the data points and inspect it.
(b) Write your own linear regression code to find the best fit (don't use the scikit-learn linear regression package). You will need the result from the written part of the assignment. Plot the learned linear function together with the training data points and see how it fits.
You may find it convenient to convert the columns of the DataFrame into numpy arrays and work with the arrays.
(c) What are the results of θ_0 and θ_1 of your linear regression? Assume the linear function has the form $y = \theta_0 + \theta_1 x$.

P3. (15 points) Load the data set named 'nonlin_df.csv' from Canvas. Repeat the steps in P2.
The data is generated by $Y = X^{2.5} + \epsilon$, where ϵ is a random noise independent of X and has zero mean. You should superimpose the function $y = x^{2.5}$ in your lot. It is the best prediction function because $E[Y|X = x] = x^{2.5}$.

P4. (20 points) You will see that for the 'nonlin_df.csv' data set, linear regression does not give a good fit. Now, implement your own K-Nearest-Neighbors (KNN) code. Plot the result of learning for three cases: $K = 4$, $K = 8$, and $K = 16$. You will see that although KNN provides a good fit, it does not yield a smooth function.

P5. (10 points) For this part, you will use the data in the file 'lin_df.csv'. In your written part of the assignment, you derive the function $h(\theta_0, \theta_1)$, which is a quadratic function of θ_0 and θ_1. Calculate the required coefficients using the training data. Plot the function $h(\theta_0, \theta_1)$ in 3D using matplotlib. Please try to show the minimum in your plot, if you can. If the function is hard to visualize in 3D, you may supplement it with a sequence of 2D plots, one for each chosen (fixed) value for θ_1.

P6. (5 points) Plot the function $g(\theta_0, \theta_1) = \theta_0^2 - \theta_1^2$ in 3D around the point (0,0). You should see (0,0) is a saddle point.