Practical Machine Learning

Guidelines and Submission Instructions:

1. The project is worth 50% of your overall module grade. You will produce a **.py file** containing all your code and a project **document** detailing your findings.

2. Upload your solution python file (**.py file**) and report as a single .zip file before **20:00 on Tuesday Nov 6th**.

3. Go to the "IBL Project" folder to upload your file.

4. Once you have submitted your files you should verify that you have correctly uploaded them. It is your responsibility to make sure you upload the correct files.

5. Please make sure you **fully comment your code**. You should clearly explain the operation of important lines of code.

6. Please use **NumPy** where possible throughout the assignment.

The marks for this assignment will be distributed as follows:

- Part 1: kNN Algorithm **(50%)**
- Part 2: Investigating kNN variants and hyper-parameters **(25%)**
- Part 3: Implementation of kNN for Regression **(25%)**

Part 1 – Development of k-NN Algorithm

For parts 1 and 2 you will be using a version of the Mammographic Mass Data Set from the UCI dataset. The following is the UCI description of the dataset.

"Mammography is the most effective method for breast cancer screening available today. However, the low positive predictive value of breast biopsy resulting from mammogram interpretation leads to approximately 70% unnecessary biopsies with benign outcomes. To reduce the high number of unnecessary breast biopsies, several computer-aided diagnosis (CAD) systems have been proposed in the last years. These systems help physicians in their decision to perform a breast biopsy on a suspicious lesion seen in a mammogram or to perform a

short term follow-up examination instead. This data set can be used to predict the severity (benign or malignant) of a mammographic mass lesion from BI-RADS attributes and the patient's age."

Feature Information:

There are 5 numerical features listed 1-5 and a binary class (number 6 on the list below)

1. BI-RADS assessment: 1 to 5 (ordinal, non-predictive!)
2. Age: patient's age in years (integer)
3. Shape: mass shape: round=1 oval=2 lobular=3 irregular=4 (nominal)
4. Margin: mass margin: circumscribed=1 microlobulated=2 obscured=3 ill-defined=4 spiculated=5 (nominal)
5. Density: mass density high=1 iso=2 low=3 fat-containing=4 (ordinal)
6. Severity: benign=0 or malignant=1 (binominal, goal field!)

In the project folder you will find a file called cancer.zip that contains a training and test file for this exercise. Please note that dataset is a smaller version of the Mammographic Mass Data Set which has undergone some cleaning.

The objective of part 1 of this assignment is to build a k-Nearest Neighbour algorithm where k=3, which takes as input a training and test dataset and will predict the appropriate class (Severity - benign or malignant) with a certain degree of accuracy.

Your k-Nearest Neighbour algorithm should use standard Euclidean distance for calculating the distance between query instances and the instances in the training data. It should report the overall accuracy of the algorithm as the percentage of test (query) instances classified correctly.

As part of the code for your kNN you should include a function called *calculateDistances*. The primary objective of this function is to calculate the Euclidean distances between a query point and a collection of other data points in space (your training instances).

The distance formula you should use is the standard Euclidean distance

$$\sqrt{(a_2 - a_1)^2 + (b_2 - b_1)^2 + \ldots}$$

The function calculateDistances should accept as arguments a single NumPy 2D array containing all the feature training data points as well as a 1D NumPy array containing a single query instance (please note that no 'for loops' should be used in this function).

Using the standard Euclidean distance formula as outlined above. The function should calculate the distance between the query point and all data points in the NumPy array.

The function should return a NumPy array containing the distance from the query point to each of the individual data points. It should also return a second NumPy array containing the indices that would sort the distances array. (Note: You may find the method argsort useful).

Part 2 – Investigating kNN variants and hyper-parameters

The objective of this section is to investigate the performance of k-NN variants and parameters and to compose a report documenting your findings.

More specifically, you should implement a distance-weighted variant of the kNN algorithm you developed in part 1. You should then compare it's performance to that of algorithm you developed in part 1 for the breast cancer dataset.

There are a range of hyper-parameters that we can use to potentially improve the performance of our k-NN algorithms (see IBL lecture notes). For example, you can alter the value of k or use alternative distance metrics. Your report should document the range of hyper-parameters that you investigated and the resulting accuracy. Please note that when incorporating additional hyper-parameters you should implement the underlying code yourself (as opposed to relying on imported functionality).

Part 3 – Developing kNN for Regression Problems

The final section of this assignment requires you to implement an IBL algorithm for tackling a regression problem.

In the project folder you will find a zip file called regressionData.zip. This folder contains both a training and test csv file. There are 12 features in the dataset. There are 6400 instances in the training set and 1600 instances in the test dataset. The target regression variable is the final column in the each dataset (13th column). You should assess the accuracy of your regression-based IBL using an R^2 metric.

$$R^2 = 1 - \frac{sum\ of\ squared\ residuals}{total\ sum\ of\ squares}$$

Where

$$sum\ of\ squared\ residuals = \sum_{i=0}^{m} (f(x^i) - y^i)^2$$

$$total\ sum\ of\ squares = \sum_{i=0}^{m} (\bar{y} - y^i)^2$$

Add a section to your report documenting the accuracy of the regression algorithm you developed along with the steps you explored to improve the overall performance.