Carl Wilhjelm

Machine Learning

Final Project

Introduction:

For the purposes of my project I elected to use the programming language which I felt would provide the most useful and relevant experience for any future machine learning projects which I may engage in, Python. Python is particularly chosen for its reliability, versatility, and already available libraries. I coded this project entirely in PyCharm, and utilized libraries such as pandas, numpy, and scikit-learn to complete it.

Part 1: Classification

The classification problem in machine learning represents one of the most fundamental challenges that can be approached by a machine learning algorithm. As such, a wide variety of options were available in attempting to solve this problem. As a first attempt in programming a machine learning algorithm, my first attempt to solve this problem revolved around the simplest option I was aware of, KNN.

The KNN algorithm essentially calculates the distance between all shared values between any two rows and uses this value to calculate the label. An implementation of this algorithm already existed as part of the sci-kit learn libraries, and rather than reinvent the wheel I elected to use this library when applicable. I attempted to use the method I developed for Part two in order to solve for the missing values, however this proved unsuccessful because all rows were missing some values. For that reason, I also used a pre-existing library to fill in missing values with mean, median, and mode.

KNN provided adequate results, which were difficult to improve upon. For the purposes of part one my efforts were divided primarily between methods using KNN and Random Forest. However, as I do not have a good grasp of the ideas behind random forest. I avoided its use unless the results were significantly better. Modulating between various values for K and replacing average with median also improved results somewhat. Overall results varied wildly from 30% to 90%.

Part 2: Missing value estimation

The missing value estimation portion of the final project is where I personally found the most contemplative portion of this project to be. After finding very little success in employing libraries to solve this problem I elected to write the code entirely from scratch.

Based on the weighted KNN example we discussed in class, I was able to design an algorithm based entirely on that principle. The algorithm first replaces all erroneous values with the numpy ‘nan’ value. Then uses the existence of these values to eliminate candidates for nearest neighbor. This process is based on the following algorithm.

1) given the row for the element which is missing and sought after, remove from all consideration the columns in this row which do not have values

2) search all other rows for ‘nan’ values and if the row contains a ‘nan’ value in any column which is being considered, remove this row from consideration for nearest neighbors

3) once all possible neighbors are found, sort by distance and calculate value of missing element using weights

I further attempted to develop a method for calculating the effectiveness of this algorithm using known values but was unable to come up with a solution which provided a logical percent grade. Therefore, the ultimate success or failure of this algorithm remains unknown.

Conclusion:

Overall, the project was a healthy combination of libraries and personally developed algorithms. Given more time, I’m sure even more progress might have been made and look forward to pursuing such work in the future.