Analysis

Juan Carlos Ramirez Prado

June 2019

1 Exploratory analysis

To rank the sensors according to predictive power I first define which is the best sensor. The best sensor is the one that you would choose if you could only choose access to the data of one sensor. The next-best sensor would be the one that you would choose if you couldn't choose the best sensor. In general ,the k-th sensor in the ranking would be the one that I would choose if I couldn't choose any of the $0 \dots k-1$ best sensors.

To do prediction using $sensor_i$,we would apply some classifier method to the data $(sensor_i, label)$. Of course, in order to know how I would use the chosen sensor for prediction, it is best to do some exploratory analysis. First, it is useful to see that both class labels have 200 elements (equally-sized classes).

Next, What I do is I split each sensor feature into 10 different bins of equal width, and see what is the amount elements of class 1 in bin_i as a proportion of the total amount of elements in bin_i.

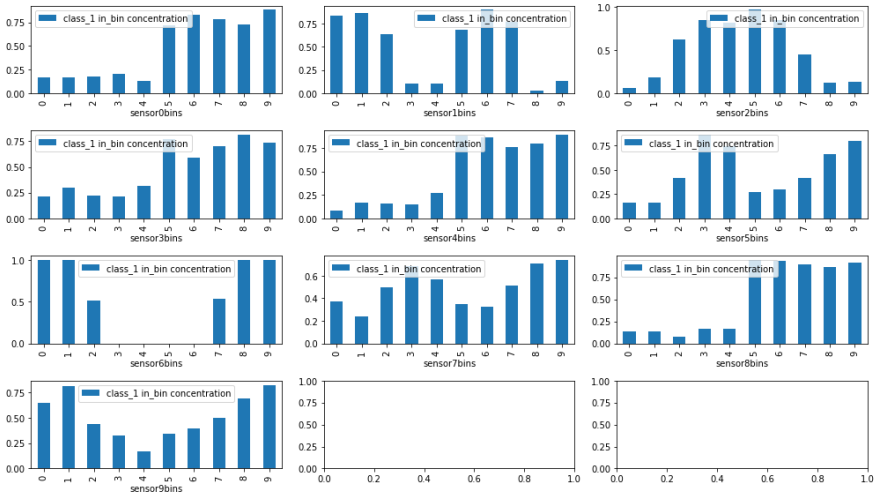

Figure 1: Sensors, proportion of class label 1 within equal-with bin

I represent the information about the proportion of ones in each bin in Figure 1.The best sensors have bar charts that look like piecewise 0-1 functions. A property of the sensor data is that a lot of these bar charts can be thresholded into up to 4 different regions where they look like piecewise functions. Intuitively, the best sensors are the ones were each bar is far away from 0.5 (random classifier), or put in another way, the ones were each bar is either close to 1 or close to 0. In fact, these bar charts could be seen as a crude classifier, where to predict a new element x first I would figure out which bin bin_i to put it in, and the predicted class would be given by the formula

$$y_{pred}(x) = 2(round(proportion(bin_i))) - 1.$$

2 Classifier

This crude classifier actually inspired my approach using k-neighbors classifier: for a data point x_{val}, create a "bin" made up of the k closest nearest neighbors from the provided data, and they vote according to which class they belong in the data. I discount the voting power by the inverse of the distance to x_{val} (not all votes are worth the same).

To choose the k for each sensor, I do parameter tuning through K-fold cross-validation, choosing the k gives us the maximum accuracy when averaged across the folds. Once I have a model for each of the sensors, I rank the sensors according to what their model prediction accuracy is. When I double-check against the test data, the rankings of the sensors are reasonably similar (the Kendall-tau statistic rejects the hypothesis of no association between the training and test rankings).

3 Strengths and weaknesses of classifier

One of the strengths of the k-nearest neighbors classifier is its simplicity. In particular, it doesn't require a lot of model structure, and is easily explainable (you choose the class of the points closest to you). Another advantage is that it is well suited for the small amount of data. Also, the interpolation is reasonable since the sensor values don't have big gaps, and are constrained to an interval.

The classifier can provide additional info: it doesn't just provide a ranking, we can measure our "confidence" of the rankings based on the difference between the accuracy of a sensor i and its next-ranked sensor. Finally, we can extend the classifier so that it doesn't just output a class, but it also outputs the probability of belonging to the class by using the predict probability method that sci-kit makes available.

One of its main weaknesses is that the classifiers need to keep in memory the training data, which is fine for a small amount of data but is not scalable to large training sets. Another weakness is that if we try to get the probabilities as mentioned above, they don't have as nice an interpretation as the probabilities outputted by a logistic regression classifier. I should also note that

tuning the k parameter is somewhat expensive, compared with models with no hyperparameters.

Although it is not a weakness, the ranking is produced by keeping the sensors independent. If we wanted to use a different notion of ranking (the second best sensor is the one that best "compliments" sensor one), we would best do something like orthogonal matching pursuit,where we progressively increase the amount of sensors used.