# Introduction to data mining lab2: Intro to scikit learn

Renfei Wang (wang.10876) Qi Song(song.1602)

Sept.29 2-19

In this project we studied on lab2.csv, a dataset for a binary classification problem, in which each datapoint has two dimensions of attributes. We used visualization tools to show some key characters of the data. We then train-test split to separate the test set and training set as the subset of the dataset. And N-neighbors algorithm was applied to solve this classification problem. We used default hyperparameters (n\_neighbors=5) and grid search and finally compare them on the testing set. We discovered that models with hyperparameters tuned via grid search have minor performance boost with accuracy on training set.

1. Load and Visualize Data (Visualization from part 1)

After load the data into dataframe, we plotted the data with different color by their class, as shown in figure 1, the data points that are labeled as class 1 are rendered in yellow, and the data points labeled as class 0 are rendered as purple. As we can see from the figure, the label 0 class are tend to appear on the top half of the plot to the right, and the label 1 class are tend to appear on the bottom part of the figure to the left and they slightly overlapped with each other. As we can see in the figure 2(a) that the distribution of a1 for different class, though not completely same, are very similar; and from figure 2(b) that the distribution of a2 for different class are different, for class 0, the range of a2 attribute tends to be larger and for class 1, it tends to be smaller, which reveals the separability along y-axis in figure 1.

Figure 3(a) is the boxplot of the attributes, and figure 3(b) provides kernel density of the attributes, through these two figure we can find out that, though the two attributes are in the about the same range, there are not exactly the same, to promote the performance of the classifier, we applied a min-max scaling to the dataset before feeding it into the classifier.

1. Prepare Data

We first set columns a1 and a2 as attributes, which should be the input of the model, and class column as the label, which is the desired output given specific input pattern. We use the sklearn.model\_selection.train\_test\_split method to split the data into training set and test set with test set size at 0.2. The resulted training set contains 400 data entries and the test set contains 100 data entries.

We further use sklearn.preprocessing.MinMaxScaler to transform the attribute into the range of 0 to 1. The scaler was fit\_transformed on training set and then applied the transform function to the testing set. The attributes of testing set and training set after rescaling are shown in figure 4. Note that since the scaler was fitted on the training set, for in reality we should not have the knowledge of test data when we are applying the rescaling. So, it’s guaranteed that the max value of each attribute in training set is 1 and minimum is 0, as shown in figure 4(a), which is not true for testing set, as shown in figure 4(b).

1. Training the Model (Visualization from part 2)

We first trained a 5 NN classifier on the training set, also employed grid search to try different numbers of n\_neighbors with 10-fold validation on the training set. The 5NN model has a 0.905 accuracy on the training dataset, and the grid search suggests a 24 NN model, and after retrain it on the entire training set it yields 0.9075 accuracy on the training set. After that we have the models tested on the test set obtained earlier, with 5-NN at 0.90 accuracy and 24-NN at 0.9 accuracy. Figure 5 shows the prediction result of 5-NN classifier, Figure 6 shows the prediction result of 24-NN classifier.

1. Summary

After completing the above experiment, we have acquired basic knowledge of a pipeline of applying machine learning algorithms to a set of data. We also tried some techniques that help us to understand the data better and help model selection. We observed that for K nearest neighbors classifier, as long as the number of n\_neighbors are in a reasonable range, the classifier may tends to have reasonable output; and from figure 5 and 6 we can observe that, most classification error that k-NN classifier make is the overlapping area of the two classes.

Compare the accuracy figure you got for the training set with that of the predictions on the test set. Were the predictions better or worse? Why?

In some trials we observed that there might be high accuracy on test set compared to training set, but in lots of case accuracy on test set is slightly lower than the training set. Such phenomenon is foreseeable since KNN algorithm doesn’t actually “trained” on the training set, so if randomization gives more separable testing set, the loss of test set might be lower. But in most of the case, the performance on test set is slightly lower than on training set, which is a common situation in all machine learning tasks. We conclude that such phenomenon might be a result of random split of test and training set.

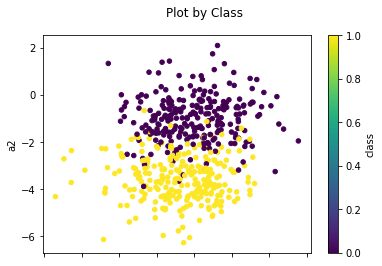


Figure 1

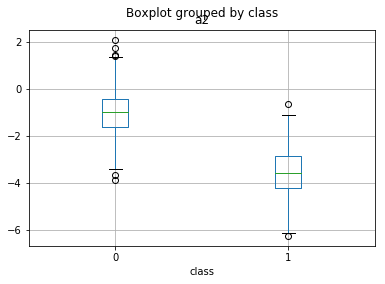
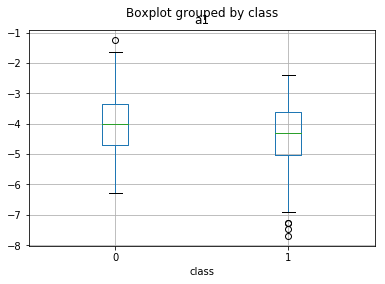


Figure 2

Figure 2(a)

Figure 2(b)

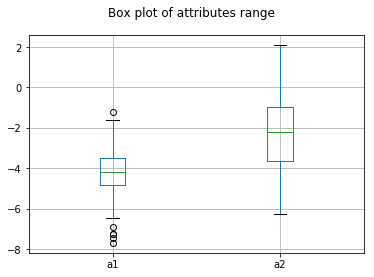


Figure 3

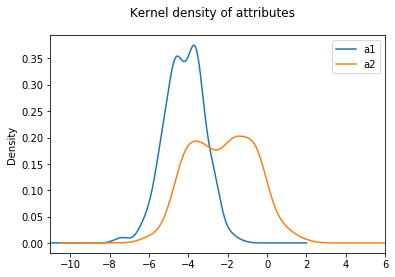


Figure 3(a)

Figure 3(b)

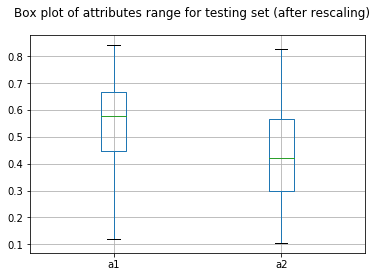
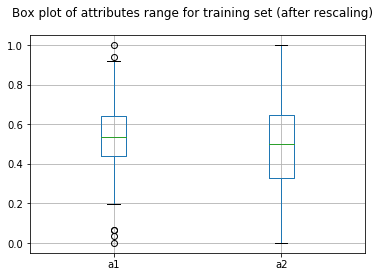


Figure 4

Figure 4(a)

Figure 4(b)

Figure 5: Prediction of 5-NN on test set

Figure 5(a)

Figure 5(b)

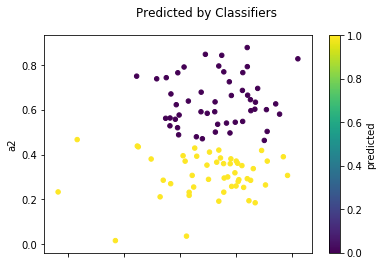
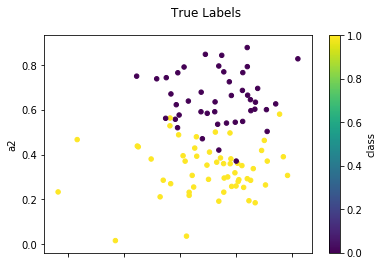
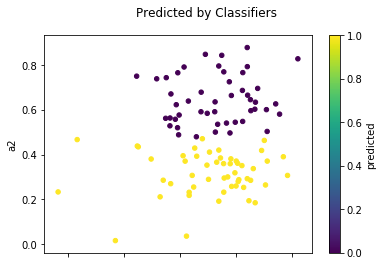
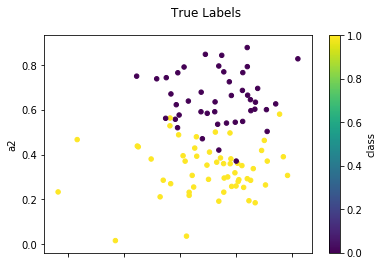


Figure 6: Prediction of 24-NN on test set

Figure 6(a)

Figure 6(b)