Assignment 2 Report

Ex3

There are three classifiers that are used to compare with those in part a.
The filenames for each of them are below:

“rules_ZeroR”:
F1-score from Ex1 = 0.074; F1-score from Ex3 = 0.074;
“trees_RandomForest”:
F1-score from Ex1 = 0.865; F1-score from Ex3 = 0.887;
“trees_RandomTree”:
F1-score from Ex1 = 0.653; F1-score from Ex3 = 0.610;

Top 3 selected features =
Angle of BB diagonal / Bounding box height / Convex hull area ratio / Convex hull area ratio

Below is the comparison.
rules_ZeroR: This is a majority classifier. The prediction from part a is “a”, while the one from Ex3 is “r”. This is because the extended_features.csv file in EX3 is not ordered. Therefore, whichever comes first can be the answer. In this case, the number of features doesn't matter as the dummy classifier can only return the answer that is the majority in the dataset.
“trees_RandomTree”: The number of Correctly Classified Instances of dataset of 13 features is 340, while the one of dataset of 45 features is 156. This could result from over-fitting.
“trees_RandomForest”: With this classifier, F1-Score has been improved quite a bit with more features.

Based on the performance of these results from Ex1 and Ex3, it is observed that with different classifiers, the best number of features selected differ. For a given classifier and a dataset, classifiers with more features usually perform better with more data, as more trains are required to try out the complexity. In order to balance the model complexity with the overfitting or underfitting behavior, options that can be adopted are:

1. split the data into a training set and a testing set.
2. use a simpler model with feature selections.

Ex4

“rules_ZeroR”:
F1-score = 0.4;
“trees_RandomForest”:
F1-score = 0.994;
“trees_RandomTree”:
F1-score = 0.975;

Top 3 selected features =
Angle of BB diagonal / Cos of first to last / Cos of initial angle

As the results show, the majority classifier, serving as the baseline, only hits 0.4 F1-Score, which is low compared to random forest and random tree. Since the F1-Score of the random forest classifier is high enough, most likely it is unnecessary to explore a better ML algorithm.

Gesture 1: down:
Gesture 2: forward
Gesture 3: circle
Gesture 4: back

For swiping down, back and forward, simply use the cosine of the first point and the last point to determine which is which. For swiping circles, simply use the bounding box area.

Ex5
After loading the data with pandas, we shuffle it, avoiding removing only on class in a train/test split.

Then we do the feature selection part. In total, we tried 5 feature selection method. First one is variance threshold feature selection. By simply setting the threshold to 0.8 we will get 31 out of 46 feature, removing loads of similar features. Second, we tried univariate feature selection with SelectKBest, which is based on the univariate statistical test, and we just used the most common one: chi2. We also tried to use RFE and SFS as well, but owing to some Python issue that we are unfamiliar with, it did not work out will. Last but not least, we used SelectFromModel, which is also based on a Machine Learning Model estimation like RFE. Specifically, we used the ExtraTreeClassifier.

Finally and most importantly, we use RandomForest as our classifier, and its f-measure is larger than 92%.

Shape1:triangle Shape2:arc Shape3:corner Shape4:square Shape5:circle