Jesse Backman 255240  
Stefanus Wirdatmadja 232722  
Kuisma Hannuksela 257860  
Ville Holmberg 252926

SGN-41007 Assignment 1

Project report

# PROJECT REPORT

## Data loading, preprocessing

The data given for the “TAU Robot Surface Detection” competition should be grouped based on the time domain group in order not overfitting the model. With numpy library, we use “np.load()” command to extract the X data to the corresponding parameters. This command will load data given in the competition from the files whose extension is “.npy”. And the given “groups.csv” will be used to group the data which will be fed into training data and test data. So, the data which belongs to the same series of data will be categorized to either training or test data. Finally, each string label will be encoded by unique integer.

After all the required data have been loaded, we need to extract the required features. We computes the mean of each channel and adds 10 new features which are the standard deviations of each channel and normalizes each measurement. This will generate the normalized data in "squashed" form with standard deviations of each channel as additional features. The reason of this normalization is to scale the data so the comparison among different classifiers is possible. Finally, we split the data to 80% training data and 20% test data using the command “GroupShuffleSplit()”. Next, this data will be used for the next processes.

## Classifier evaluation & selection

Classifiers selected for evaluation:

* Linear Discriminant Analysis
* SVC (linear kernel)
* SVC (RBF kernel)
* Logistic Regression
* Random Forest
* KNeighbors
* Extra Trees

The 80% of the split data was used to train each classifier. The remaining 20% was used to test the accuracy. One thing that the group noticed was that the parameter random\_state of the function GroupShuffleSplit influenced the results quite heavily. This is due to how the random\_state parameter influences the split. To evaluate the performance of the classifiers, the group used n\_splits=100 parameter for the GroupShuffleSplit to split the data 100 times. Each classifier was trained with each split of the data and the accuracy score was computed with sklearn-library’s accuracy\_score function. The average of these scores was then computed and the results are listed in Table N:

|  |  |
| --- | --- |
| **Classifier** | **Average score over 100 splits** |
| Linear Discriminant Analysis | 0.33 |
| SVC (linear kernel) | 0.37 |
| SVC (RBF kernel) | 0.36 |
| Logistic Regression | 0.32 |
| Random Forest | 0.46 |
| KNeighbors | 0.37 |
| Extra Trees | 0.36 |

The selected classifier for classifying the Kaggle data was Random Forest. The group notes here that the classifiers were trained mostly with default parameters (ensemble methods had 1000 trees), so with parameter tweaking i.e. Extra Trees could reach better results. The parameters for each classifier were as follows:

classifiers = [LinearDiscriminantAnalysis(),

SVC(kernel='linear', probability=True),

SVC(kernel='rbf', probability=True),

LogisticRegression(),

RandomForestClassifier(n\_estimators=1000, max\_depth=4),

KNeighborsClassifier(),

ExtraTreesClassifier(n\_estimators=1000, max\_depth=4)]

The initial target of the group was to reach the baseline on Kaggle. For this purpose, the Random Forest Classifier was selected to do the job. Further evaluations and i.e. feature engineering could be done if there is interest in the group and enough time.

## Final results

Our first submission score was 59,085% with only using KNeightbors and soon after that we made new submission using Random Forest and score was 67,995%. It reached the baseline on Kaggle so after that we only did little changes to the parameters and that didn’t improve our score anymore. Score with slightly changed parameters was 66,354% so our best score was still 67,995%.

**Final score: 67,995%**