Machine learning: Learning challenge

Group number 22, account name Jrzi
Tom van den Broek - u299223
Nicky van Bruggen – u190658
Jeroen van Buren – u490632

Data exploration

At the start of the learning challenge the written data, spoken data and match data was explored. The shape of all datasets was inspected. After running the first row of the written data it confirmed that it consisted of a 28x28 matrix of grayscale pixels which had a value between 0 and 255. By plotting the written data, noise was discovered in the background. Also the spoken data was inspected. As described, the spoken data consisted of a three dimensional array of 45.000 x N x 13 instances. Therefore, the data was transformed to a two dimensional matrix by aggregating the columns. Furthermore, it was remarkable that the match data contained less than 10% true matched which could cause implications in further analysis.

Feature engineering and model selection

Feature engineering and model selection procedure were done simultaneously. After pre-processing a variety of un-optimized base models were analysed. To internally test the performance and optimize the predictive models, a double hold out method was used, this resulted in a training set, validation set and test set. Cross-validation was preferred, but by means of efficiency it was decided to use a hold out method. It could provide less reliable results because of a selection bias, but this method was much faster. The learning algorithms used were: Logistic Regression (LR), Stochastic Gradient Descent (SGD), Perceptron, Naïve Bayes (NB), K-Nearest Neighbours (KNN), Random Forest (RF) and Multi-Layer Perceptron (MLP). The learning algorithms were trained on an internal training set and predicted on an internal validation set to measure the performance of the learning algorithms and different stages of feature engineering. The first four learning algorithms are linear algorithms. NB is an optimal classification model if the independence assumption holds, which in this case it does not. RF is an assembly method. KNN is a non-linear algorithm. MLP is a deep neural network, which can find non-linear complex relations.

As evaluation metric the F1 score was used to compare the predictions with the true Y's of the validation set. A classification report extensively described the harmonic average of precision and recall which together make up the F1 score. This model selection procedure is repeated after every step of optimization which will be described in the next paragraphs; these steps were almost always cumulative.

As mentioned before, to prepare the three dimensional spoken data for further analysis aggregate measures were used (mean, minimum, maximum, standard deviation, skewness and kurtosis) for every observation and were concatenated. Only the columns/channels were applicable to summarize, because the rows had unequal lengths. Then the written data was merged with the summarized spoken data. After this step, this data was validated on the learning algorithms. It was noticed that the base models performed worse on all, but small exceptions were observed in the MLP and NB. Both models scored 0.12 on the F1 score.

Because the values of the pixels were between 0 and 255 and the summary data were on a lower scale, z-transformation was a next logical step. Therefore, the train data was fit and standardized and the values were applied to z-transform the validation set. SGD, Perceptron and KNN all slightly improved between 0.07 and 0.15. MLP significantly improved to a F1 score of 0.43. All the other models stayed the same.

After studying the classification report of the previous trained models, findings showed that the models performed very well on the false prediction and a lot worse on the true predictions. Thus, because of the low amount of true labels, these were upsampled to a fifty-fifty ratio, merged and shuffled the data. Upsampling had a reasonable effect on SGD and Perceptron and especially on LR and KNN. All other models improved minimal.

The following step was removing the noise in the background. The result was visualised and it was confirmed that the noise in the background was gone. The grayscale threshold chosen by trial and error was 50. RF and MLP improved reasonably. All other models remained approximately equal. Since, this proved to be seemingly promising. Thus, afterwards we binarized the written data with the same chosen threshold, due the problem of standardizing binarized data, we did not use the standardization step. As expected, KNN improved significantly to 0.50 and had a little effect on RF and MLP, SGD performed worse. In the next run, the standardization step was again conducted, but only for the spoken summary data. The only improvement was for MLP and SGD. As a final part of feature engineering, frames were added to the summary of spoken data. The minimum number of frames is four. Therefore, the minimum number of frames was added for all observation. This resulted in a lower F1 score for KNN. The F1 scores for RF and MLP increased.

For all the results and a visualization, see the appendix and the extra evaluation excel file. the final pre-processing results, see the table in the appendix.

Optimization

In the optimization phase the best performing base models after feature engineering were selected for optimization. Well-performing models were KNN and MLP. Also RF was selected, because it performed third best and offers a lot of optimization options.

For KNN the parameters metrics, weights and n_neighbours were selected for optimization. For RF the parameters criterion , n_estimators, max_depth, min_samples_split and min_impurity_decrease were selected for optimization. Both optimization models did not perform well enough compared to the MLP. Therefore, MLP was extensively optimized as our final model. For MLP the parameters hidden_layer_sizes, alpha, activation, learning_rate and learning_rate_init if learning_rate is 'constant' were selected for optimization.

After running many of different options for the parameter tuning for the MLP, the optimal solution given in the 'optimization' gave the best performance. More hidden layers than 250 gave a minor decrease in F1 score, although they remained stable, whereas in a decrease of hidden layers, the F1 score dropped a lot. The default value for alpha at 0.0001 appeared to be too low for accurate predictions. Because 'invscaling' and 'adaptive' performed worse, we used a 'constant' learning rate, we tried different values for the initial learning rate used. The default learning rate was 0.001. Here we found a smaller optimal value of 0.0001. These parameters had the most impact on the performance. On the validation set, the best performing model scored 0.797 as F1 score.

Discussion

Evaluation

For the final internal evaluation as train data (the train and validation set) was used and the split test data as internal test set. We applied all pre-processing steps to make it consistent. Thereafter, the trainings set was trained with the optimal settings and predicted the on the internal test set, the F1 score was: 0.793. It performed slighty less than the optimal performance on the validation set.

As final step, we then used all of the available train data was used (earlier the train, validation and test data). We then loaded the internal test data and applied all pre-processing steps. Again, trained the whole trainings set with the optimal settings and predicted the on the internal test set, the F1 score was: 0.831

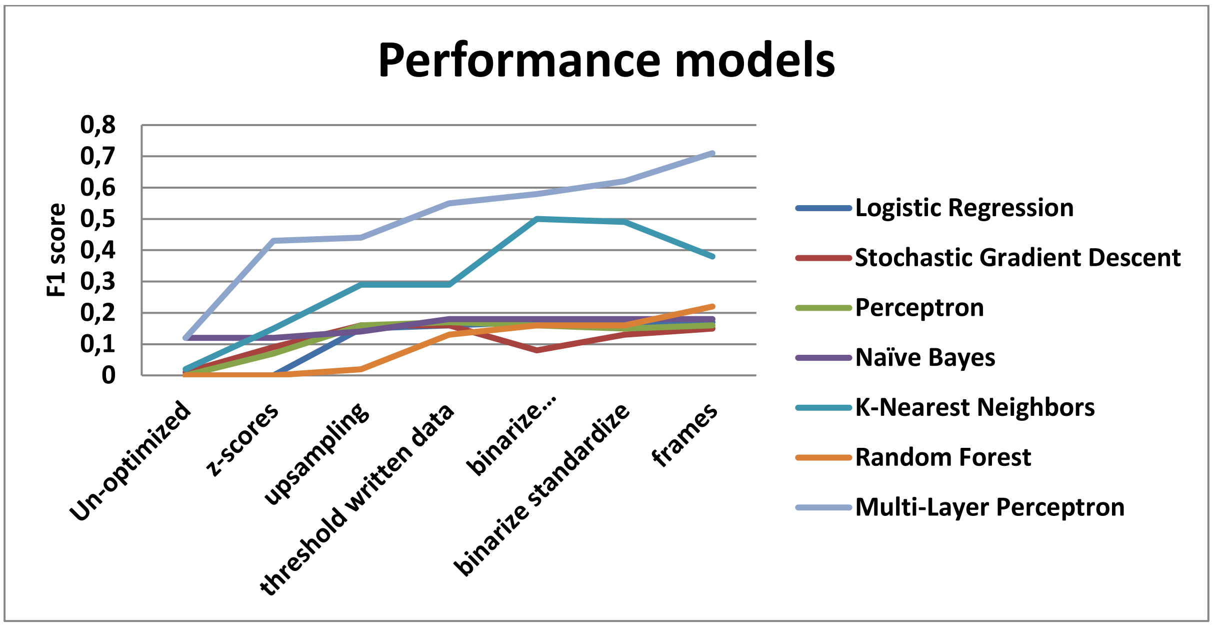
Further discussion

The performance on the Codalab was slightly better than the validation set. Therefore, we suspect that more data result might improve performance. Furthermore, probably the use of recurrent neural networks would have also increased the performance.

Division of work

During the learning challenge was attempted to work as much as possible all together. Unfortunately Nicky van Bruggen was because of a resit not able to join all the sessions. However, after the resit he spent additional time at home to validate models and in this way compensate it. It must be said that Jeroen van Buren had somewhat more experience with Numpy, Scipy and Sci-kit learn and therefore took the lead in programming.

Appendix



LR	SGD	Perceptron	NB	KNN	RF	MLP
0.17	0.15	0.16	0.18	0.38	0.22	0.71

Table 1: Performance