Technische Universiteit
Eindhoven
University of Technology

2IMM20 - Foundations of datamining

Assignment 2

Students:
Joris van der Heijden (0937329)
Bram van der Pol (0780042)

Email addresseses:
j.j.m.v.d.heijden@student.tue.nl
a.f.v.d.pol@student.tue.nl

Supervisors:
Dr.ir. Joaquin Vanschoren

Eindhoven, March 16, 2018

Contents

1 A data mining challenge

This sections shows the steps that are done to improve the model for the datamining challange. This is done in a small report to show the different steps with the generated figures.

1.1 Detects accents in speech data

First three previously used models are fitted on the data to check which model works best in this case.

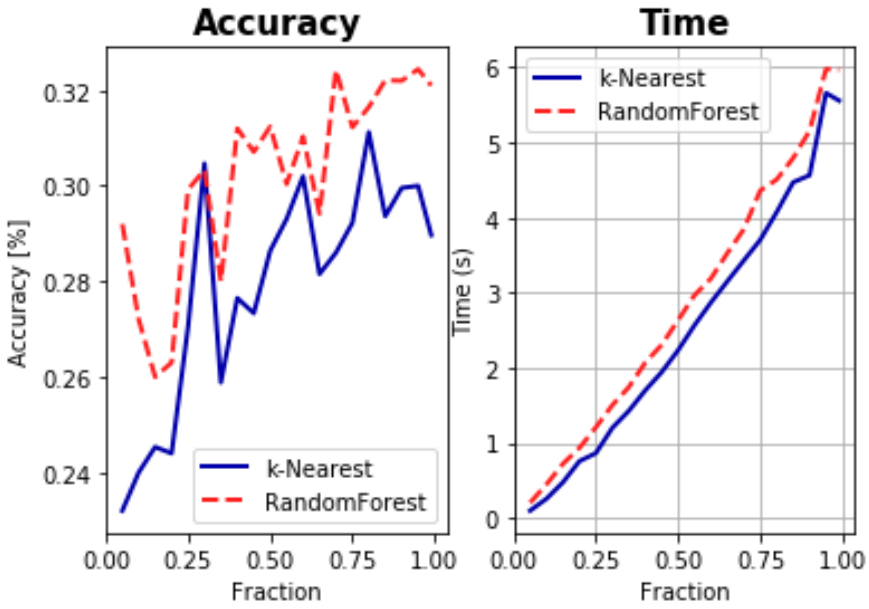

Figure 1: *Performance of the k-Nearest and Random forest algorithms.*

The RandomForest classifier is chosen because this gives the best accuracy with the default parameters. The computing time for both algorithms is

almost equal for bot algorithms. The logistic regression (not shown) has a large computation time with low performance, so therefore this model is not chosen.
In order to save computation time chosen is to start with a coarse grid and make this mesh finer and finer to come to the optimal solution. The grid search is done for only 1% of the data to save computing time. After the optimal solution is found the model is fitted on the entire data set using the obtained hyperparameters. The coarse grid search for 1% if the data is shown is figure 2.

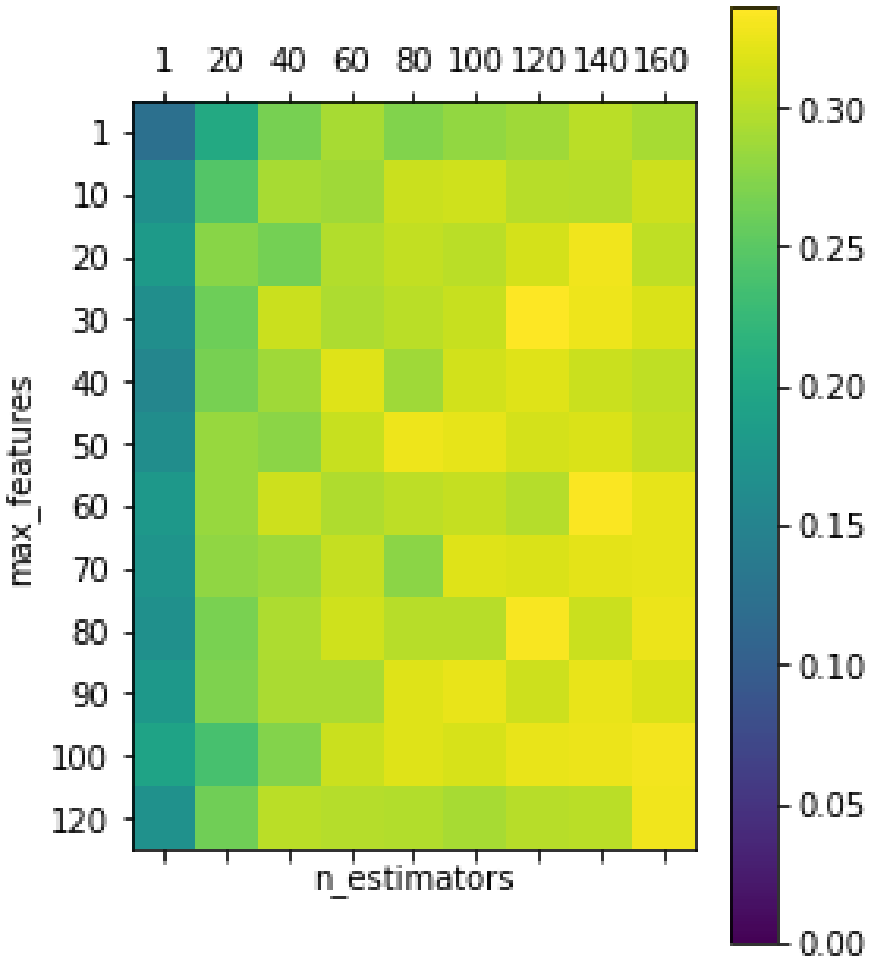

Figure 2: *A coarse grid search for 1% of the data with the Random Forest classifier. The optimal parameters are:* $n_{estimators} = 120$ *and* $max features = 30$

Now we ran a finer grid search near the domain $n_{estimators} = 120$ and $max features = 30$. The results is shown in figure 3

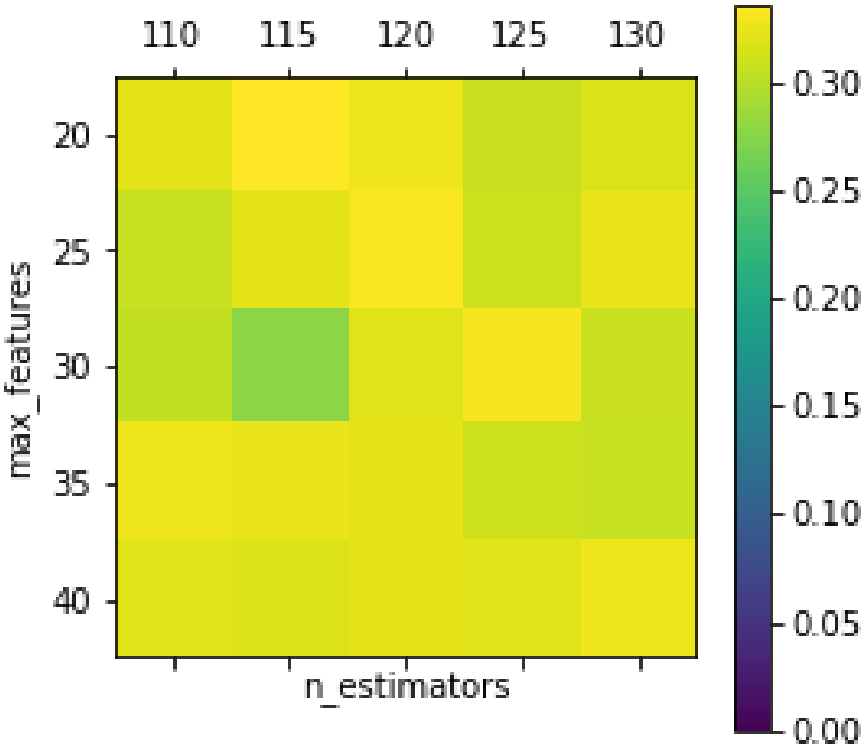

Figure 3: *A coarse grid search for 1% of the data with the Random Forest classifier. The optimal parameters are:* $n_{estimators} = 115$ *and* $max features = 30$. *This gave a score of 0.34 (for 1% of the data)*

Now we ran the model with the obtained parameters and achieved a score shown in figure 4.

```
**Results for RandomForest**
Best cross-validation accuracy: 0.43
Test set score: 0.43
Best parameters: {'Forest__max_features': 30, 'Forest__n_estimators': 120}
Best estimator:
Pipeline(memory=None,
     steps=[('scaler', StandardScaler(copy=True, with_mean=True, with_std=True)), ('Forest', RandomForestClassifier(bootstrap=Tru
ight=None, criterion='gini',
            max_depth=None, max_features=30, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
      ...n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False))])
clf step:
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features=30, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=120, n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
Elapsed time=  148.606
```

Figure 4: *RandomForest score for:* $n_{estimators} = 115$ *and* $maxfeatures = 30$. *This gave a score of 0.43 for 100% of the data*

In order to improve the results, the leaderboard was checked to look for the best algorithm. This was the $Keras$ classifier and therefore decided was to use this neural network API. Unfortunately this model took too much time to run, and therefore the optimization of the Keras algorithm did not have the desired outcome.

1.2 Image recognition (CIFAR-10 subsample)

As we have seen an image recognition problem in Assignment 1 which we tackled using random forest, it made sense to apply the same classifier on this dataset. To do this, first a gridsearch was run on a subset of the data to optimize hyperparameters, after which the optimum hyperparameter values were used to generate a model on the entire set.

1.2.1 Random Forests

In more detail, a 25x25, 10 fold cross validated grid search to optimize $n_{estimators}$ and $maxfeatures$ on 10% of the data was executed. This was left to run overnight as it took a long time to run this grid search. The results are visible in a heatmap in figure 5. It is recommended to view this report in a digital format such that zooming in is possible. Also please note that the axis labels are swapped, unfortunately there was no time left to regenerate this heatmap.

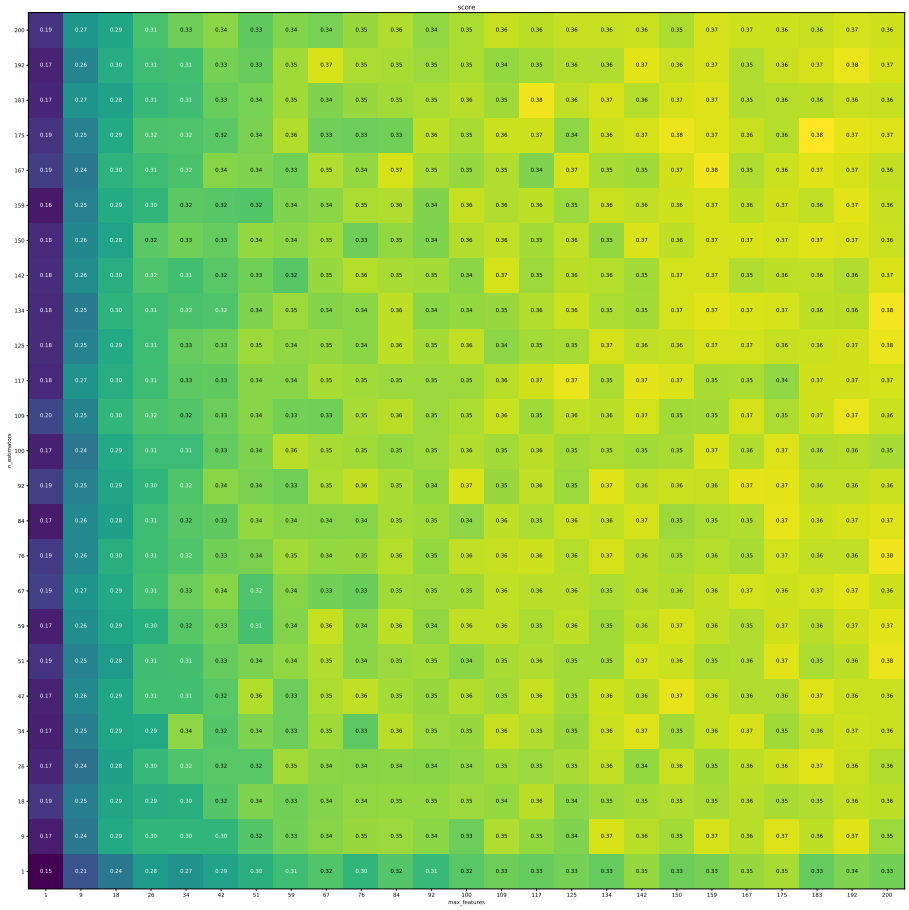

Figure 5: *25x25 grid search for* $n_{estimators}$ *and* $maxfeatures$

The scores overall were quite low, with a max score of 0.38. This may be explained by the amount of data used to run the gridsearch. Also, there is not a clear area were high scores are focused unfortunately. A large part of the grid had scores in the [0.35-0.38] range. Because the sets of hyperparameters leading to a high score in the grid, a few highest scoring hyperparameter value pairs were used to fit models on the entire data set. The pairs were chosen based on a high score and a relatively low value for the parameters in an effort to reduce the change of overfitting and to reduce computation time.

The scores generated from fitting a model on the entire set did yield higher scores than seen in the grid (0.44 fpr the entire set vs 0.38 in the grid search), but unfortunately the scores were still quite low. We do not like to see negative r2 scores here, but the obtained r2 scores are around -0.3. To

Table 1: $n_{estimators} = 192, maxfeatures = 9$

	no scaling	StandardScaler	MinMaxScaler
score	0.4326	0.4382	0.4368
r2 score	-0.289	-0.267	-0.269

Table 2: $n_{estimators} = 100, maxfeatures = 92$

	no scaling	StandardScaler	MinMaxScaler
score	0.4244	0.4250	0.4304
r2 score	-0.2956	-0.3166	-0.3080

try and improve the score, models using the previously acquired optimal hyperparameters were run in several pipelines using different preprocessing steps. The results are visible in table 1, 2 and 3. As can be seen scaling the data on average improved the r2 score marginally but still the values are negative. There is no set of hyperparameters that clearly produces the best model, but $n_{estimators} = 192$ and $maxfeatures = 9$ produces a model that is sleightly better than using other hyperparameters.

1.2.2 SVM

As random forests on this dataset did not yield satisfactory results, next we tried using SVM on the data. Again, first a grid search was executed to find optimal hyperparameters for C and $gamma$. For SVM, this was done in three stages. First a coarse grid over a wide value range was found, and then increasingly fine searches were done in areas of local optima. A heatmap for these grids can be found in figures 6, 7 and 8.

Table 3: $n_{estimators} = 134, maxfeatures = 9$

	no scaling	StandardScaler	MinMaxScaler
score	0.4312	0.4294	0.4304
r2 score	-0.3199	-0.2990	-0.2839

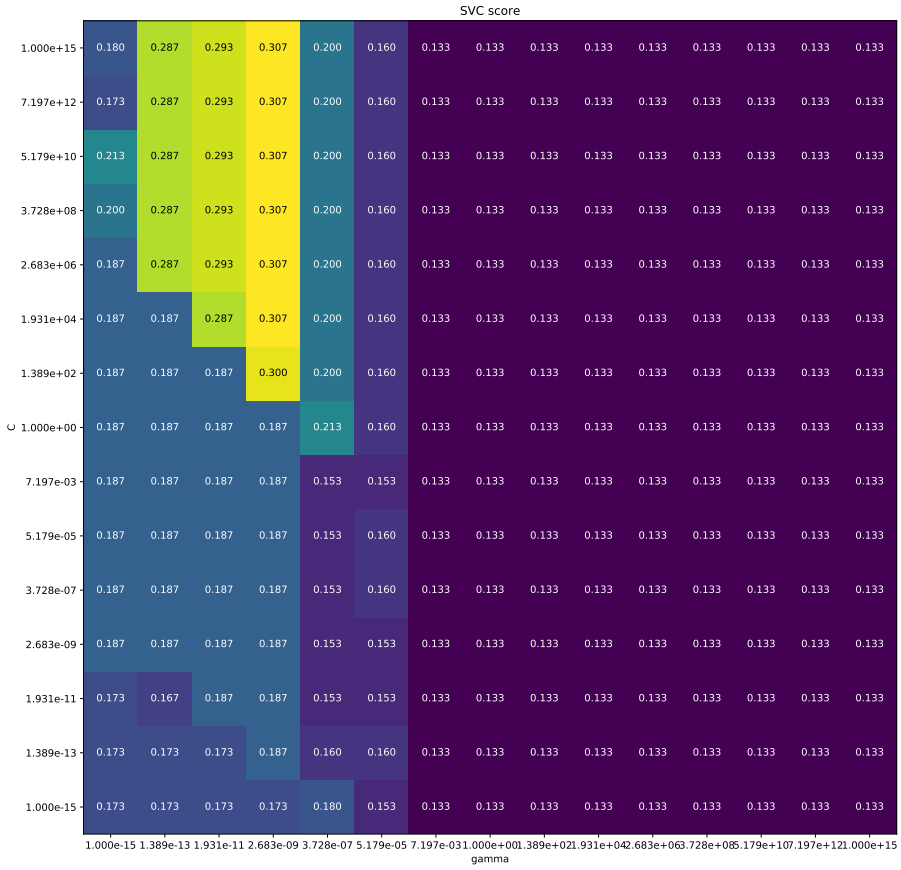

Figure 6: *15x15 grid search for* $C = [10^{-15}, 10^{15}], gamma[10^{-15}, 10^{15}]$

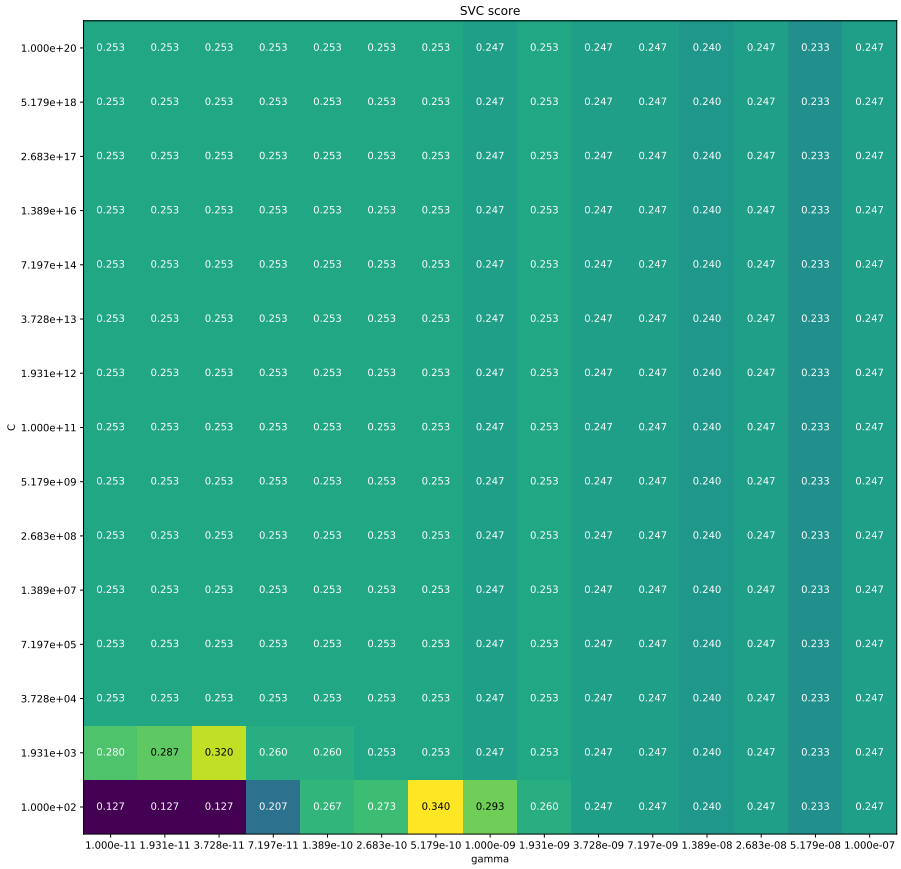

Figure 7: *15x15 grid search for* $C = [10^2, 10^{20}], gamma[10^{-11}, 10^{-7}]$

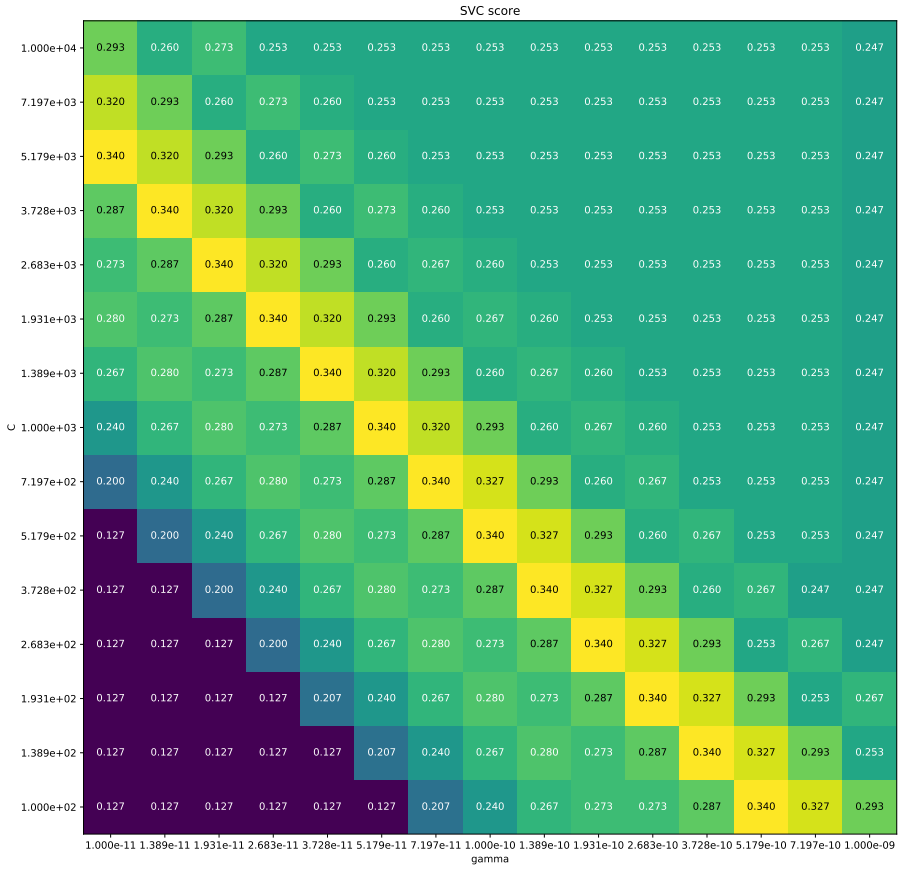

Figure 8: *15x15 grid search for* $C = [10^2, 10^4], gamma[10^{-11}, 10^{-9}]$

The obtained hyperparameter values were then used to train a model on the entire dataset. Continuing the trend we have seen up to now, fitting a model to the entire set and predicting the results once again yielded low scores, namely a score of 0.4238 and an r2 of -0.38. The best RandomForest performed sleightly better than the fitted SVM machine, but both produced badly scoring models.

At this point there was not enough time left to run large grid searches for different classifiers. As the obtained scores were so low we refrained from uploading them to OpenML. Given more time higher scoring models are certainly obtainable, but because completing assignments 1, 2 and 3 took up a long time as well not enough time was left to further optimize our models for assignment 4.