2IMM20 - Foundations of datamining

Assignment 1

Students:

Joris van der Heijden	(0937329)
Bram van der Pol	()

Email addresseses:
j.j.m.v.d.heijden@student.tue.nl
Bram email adres here

Supervisors:
Dr.ir. Joaquin Vanschoren

Eindhoven, March 4, 2018

Contents

Nepalese character recognition

1

Evaluate k-Nearest Neighbors, Logistic Regression and RandomForests with their default settings.

- *Take a stratified 10% subsample of the data.*
- *Use the default train-test split and predictive accuracy. Is predictive accuracy a good scoring measure for this problem?*
- *Try to build the same models on increasingly large samples of the dataset (e.g. 10%, 20%,...). Plot the training time and the predictive performance for each. Stop when the training time becomes prohibitively large (this will be different for different models).*

The pyhton code is shown in appendix 1.1.

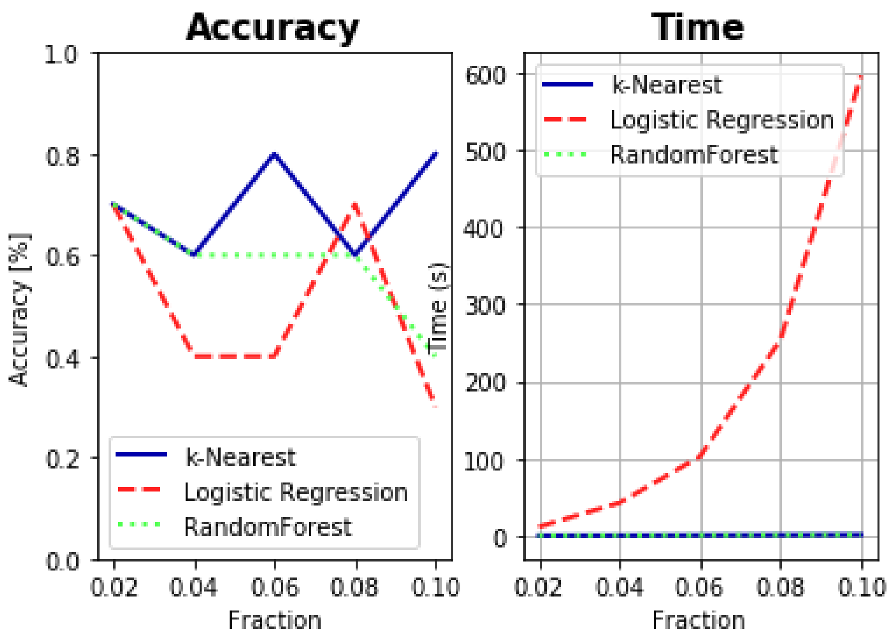
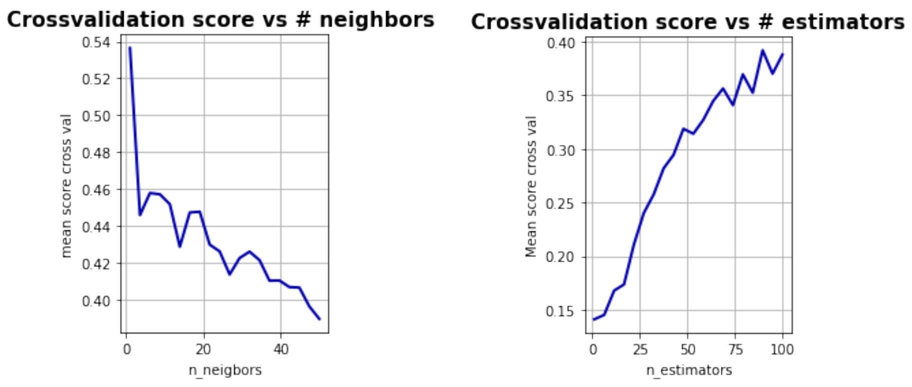


Figure 1: *Predictive accuracy as a function of the subsample data.*

Figure 1 shows that the accuracy of the k-Nearest and Logistic regression algorithm does not show a clear trend. This can be due to used randomness of the train and test samples. The predictive accuracy is due to this not a good measure for this problem.

2

Optimize the value for the number of neighbors k (keep $k < 50$*) and the number of trees (keep* $n_{estimators} < 100$*) on the stratified 10% subsample. Use 10-fold crossvalidation and plot k and* $n_{estimators}$ *against the predictive accuracy. Which value of* $k, n_{estimators}$ *should you pick? Because the simulation took too long for the 10% subsample only 1% is taken as the subsample.*



(a) (b)

Figure 2

As can been seen in figure 2 is that the number of neigbors should be set to 1 and the number of trees to 100.

3

For the RandomForest, optimize both $n_{estimators}$ and $max_{features}$ at the same time on the entire dataset. - Use a nested cross-validation and a random search over the possible values, and measure the accuracy. Explore how fine-grained this grid/random search can be, given your computational resources. What is the optimal performance you find? - Hint: choose a nested cross-validation that is feasible. Dont use too many folds in the outer loop. - Repeat the grid search and visualize the results as a plot (heatmap) $n_{estimators} \cdot max_{features} \rightarrow ACC$ with ACC visualized as the color of the data point. Try to make the grid as fine as possible. Interpret the results. Can you explain your observations? What did you learn about tuning RandomForests?

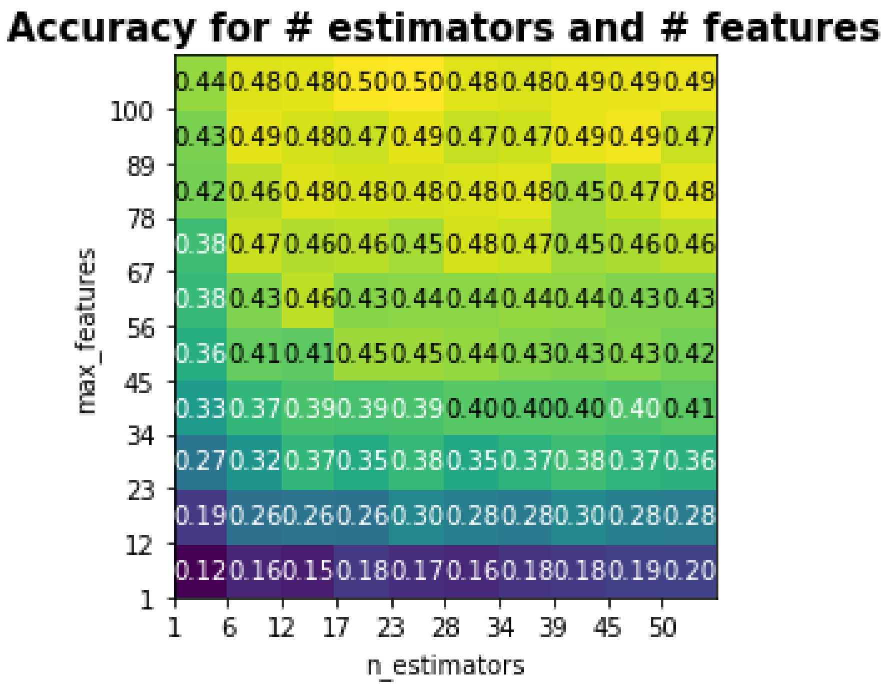


Figure 3: *Accuracy for the random forest for different maximum features and number of estimators numbers. This is done for 1% of the total data set (due to the lack of computer power and time).*

The percentage has a big influence for the accuracy prediction. The

4

1

Do a bias-variance analysis of both algorithms. For each, vary the number of trees on a log scale from 1 to 1024, and plot the bias error (squared), variance, and total error (in one plot per algorithm). Interpret the results. Which error is highest for small ensembles, and which reduced most by each algorithm as you use a larger ensemble? When are both algorithms under- or over-fitting? Provide a detailed explanation of why random forests and gradient boosting behave this way. - See lecture 3 for an example on how to do the bias-variance decomposition - To save time, you can use a 10% stratified subsample in your initial experiments, but show the plots for the full dataset in your report. In [ ]: from sklearn.model

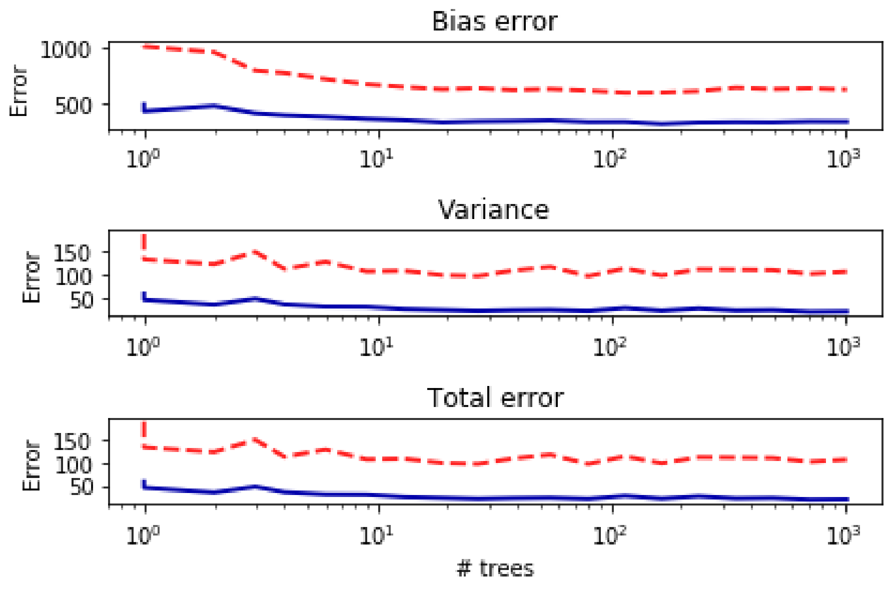


Figure 4:

1 Appendix

1.1 Code question 2.1

1.2 Code question 2.2