Machine Learning Ex. 2

Regression

Group 30

Georg Faustmann
Maximilian Moser
Wolfgang Weintritt

Agenda

- Overview
- Details
- Conclusions

Overview

Used Technology

scikits learn
machine learning in Python

matplotlib

pandas

https://matplotlib.org/1.3.1/_static/logo2.png
http://www.scipy-lectures.org/_images/scikit-learn-logo.png
https://qph.ec.quoracdn.net/main-thumb-t-348902-200-dgbemwvkzzlbbhaeukyjituzjzgudxmh.jpeg

Used Datasets

- KDD Cup 1998
- Auto MPG
- Challenger USA Space Shuttle O-Ring

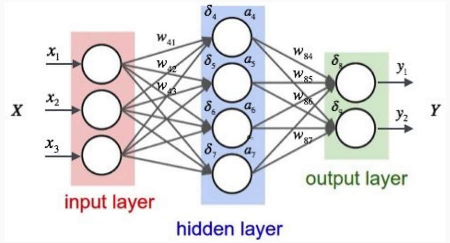
  5 columns, 23 rows, no missing values
- Appliances energy prediction

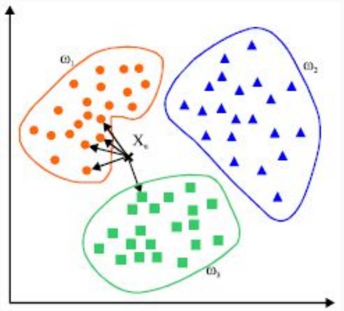
  29 columns, 19735 rows, no missing values

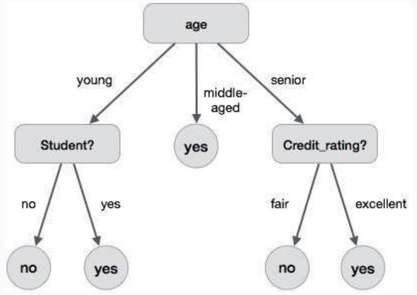
http://freakonomics.com/wp-content/uploads/2011/06/space-shuttle.jpg
https://www.energysavings.com/images/appliances.jpg

Used Regressors

- K Nearest Neighbours
- Random Forests
- Bagging Regressor
- Neural Networks (Multi-Layer Perceptron)





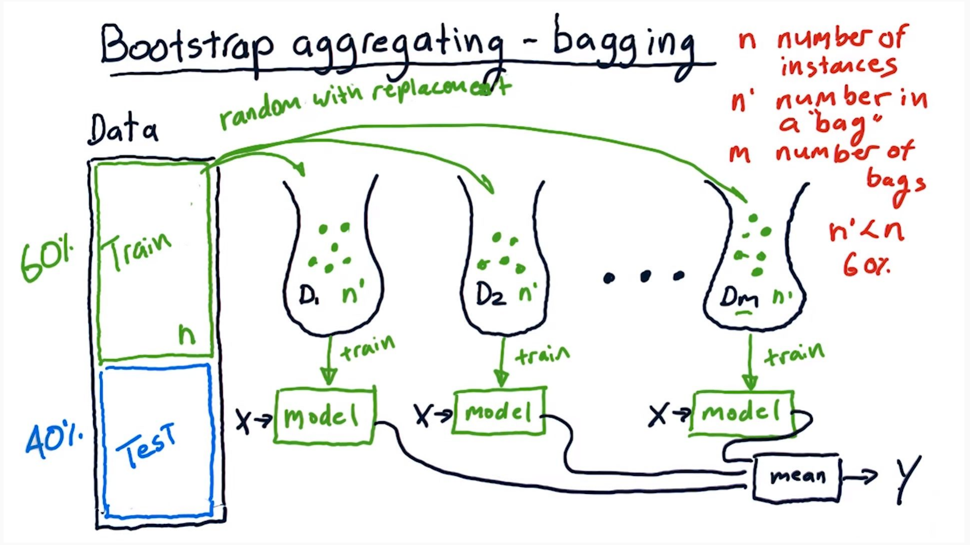


https://www.tutorialspoint.com/data_mining/images/dm_decision_tree.jpg
https://cdn-images-1.medium.com/max/479/1*QVIyc5HnGDWTNX3m-nIm9w.png
https://www.mathworks.com/matlabcentral/mlc-downloads/downloads/submissions/63621/versions/1/screenshot.gif

Bagging Regressor

Since this regressor was not explained in the lecture, here is an overview how it works:

Choose randomly n' instances of the whole dataset. Repeat this m times. These m sets forms bags. For each bag a model is trained, e.g a Decision tree. To predict new data the outputs of these m models are combined, e.g with the mean.

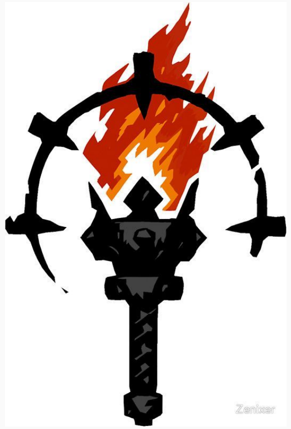


https://www.youtube.com/watch?v=2Mg8QD0F1dQ

Details

Energy Prediction: Overview

- 19735 rows, 29 columns (including Timestamp and two meaningless Random Variables)
- Predict: Energy use of Appliances (in Wh), Range: 10 - 1080
  - We left out the Energy use of light fixtures (in Wh)
- Features: Weather-related
  - Humidity, Temperature in different Rooms of the House
  - Pressure, Wind Speed, Visibility (in km) from a Weather Station
  - One Row Entry for every 10 Minutes (for about 4.5 Months → 19735 Rows)


Predicted Energy Consumption: 0 Wh

http://media.pinterest.com.s3.amazonaws.com/1200x/b2/e0/b0/b2e0b031cdcca9d96a0af988da7db9f3.jpg

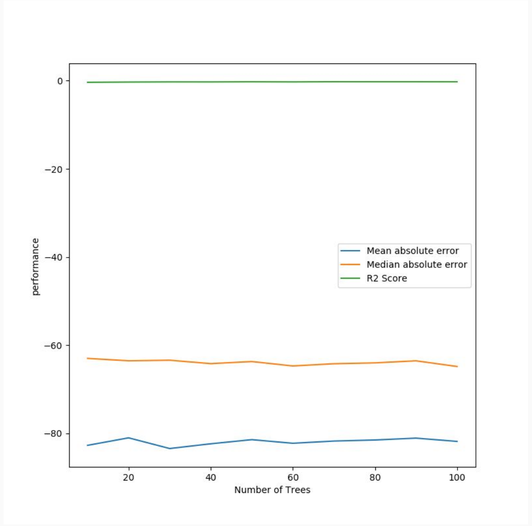
Energy Prediction: Overview

We tried the different Regressors under the following different circumstances:

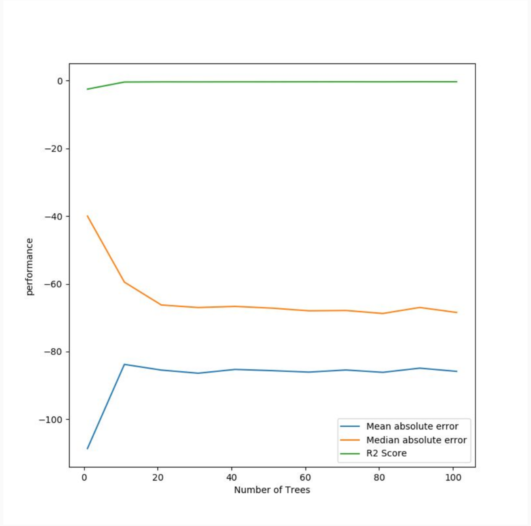
- Keeping and removing the Random Variables
- With and without Pre-Processing (normalization and scaling)
- Different Parameters for each Regressor
- With and without Feature Selection (sklearn: SelectFromModel)

Note: Due to the high values for median and average errors, the R2 score looks like a constant value on the following graphs.

Energy Prediction: Random Forest Dataset with/without Random Variables



Without Random Variables

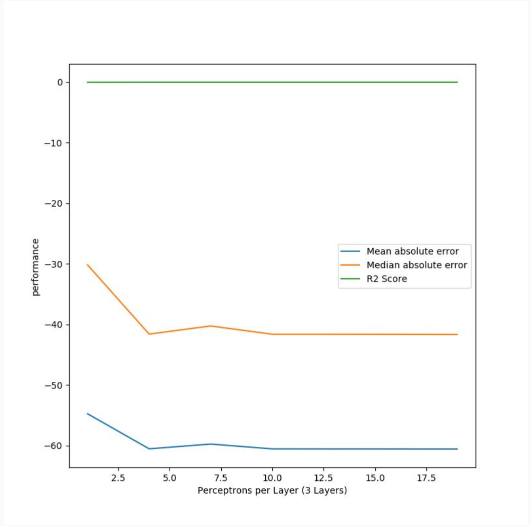


With Random Variables

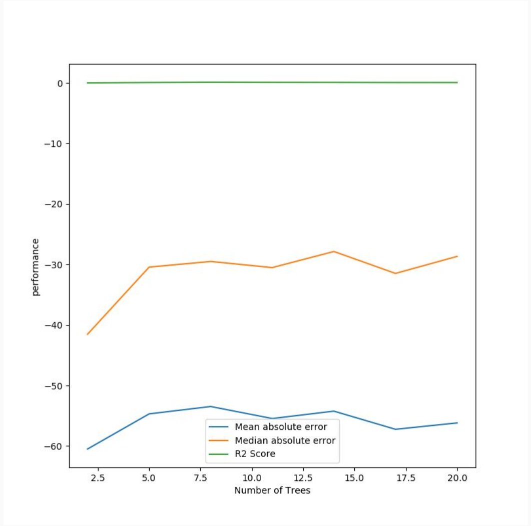
As one can see, the Random Variables do not seem to make very much of a difference (except for a peak at low parameter settings, both graphs remain at roughly the same values for the KPIs)

The same holds for the other regressors (not shown here)

Energy Prediction: MLP
Dataset with/without Preprocessing



Without Preprocessing

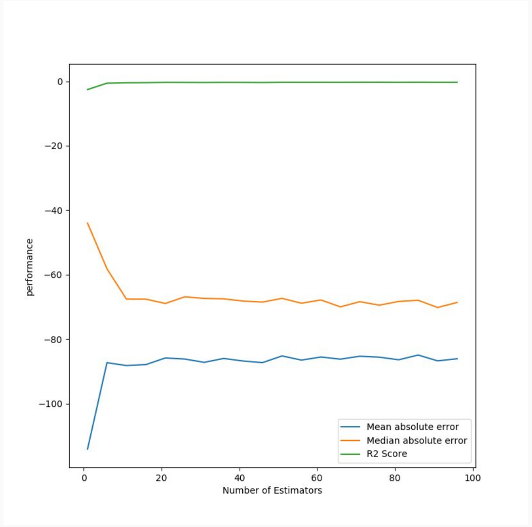


With Preprocessing

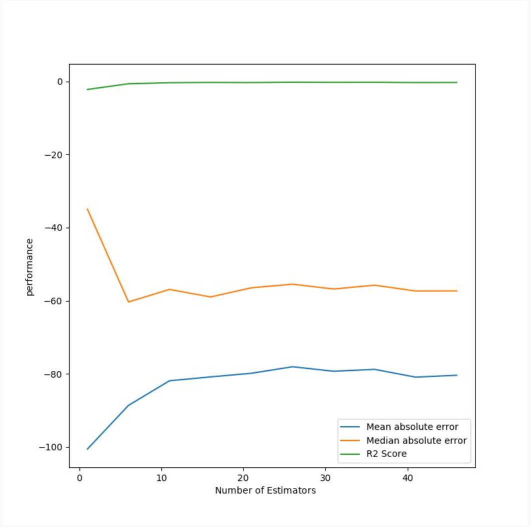
With Pre-Processing (normalization and scaling), the results tended to get better, as can be seen on the example of MLP (parameters here: Logistic / Adam / Adaptive)

This holds true even for kNN (not shown here), where the distance measure might have been impaired

Energy Prediction: MLP Dataset with/without Feature Selection



Without Feature Selection



With Feature Selection

With Feature Selection, you can see that the Bagging Regressor yields slightly better results than without Feature Selection

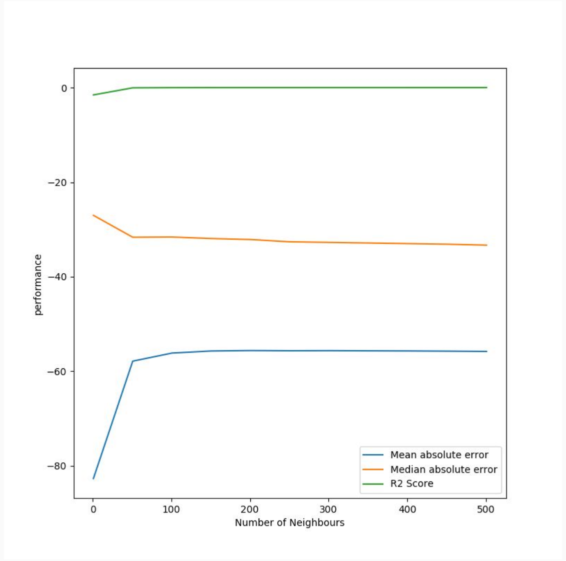
This holds true even for kNN (not shown here), where the distance measure might have been impaired

Energy Prediction: Comparison of Regressors

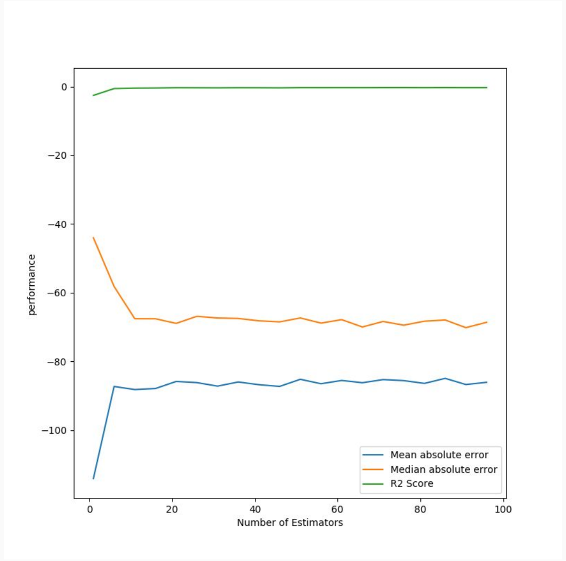
Note:

The following slides contain comparisons of the different used Regressors, with their applications all based on the same preconditions (i.e. we used pre-processing via scaling and normalization as well as removal of the Random Variables, but **no** feature selection for the runs that produced the following graphs)

Energy Prediction: KNN and Bagging Regressor



kNN

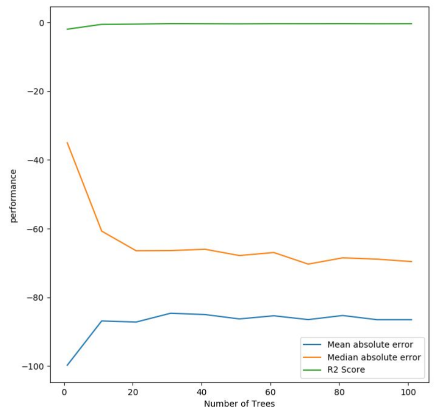


Bagging Regressor

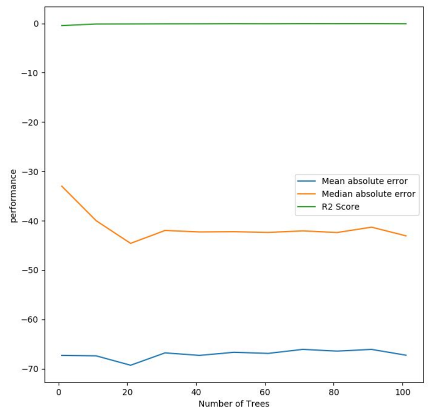
For this Dataset, the kNN regressor performed remarkably well (and consistently well over variation of the parameter)

The Bagging Regressor, on the other hand, offered lower performance when compared to the others

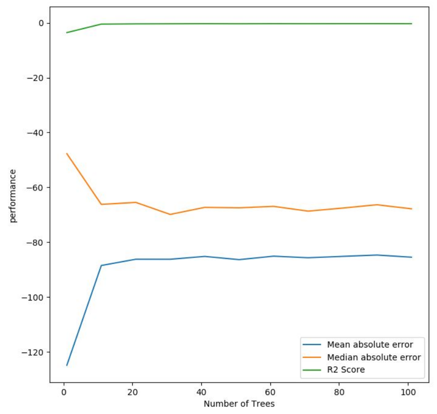
Energy Prediction: Random Forest Comparison of diff. Max. Tree Depths



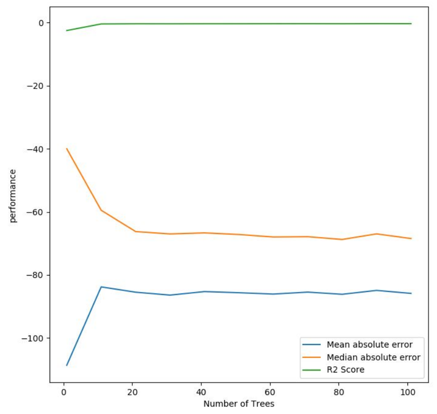
Unlimited Depth



Max. 10

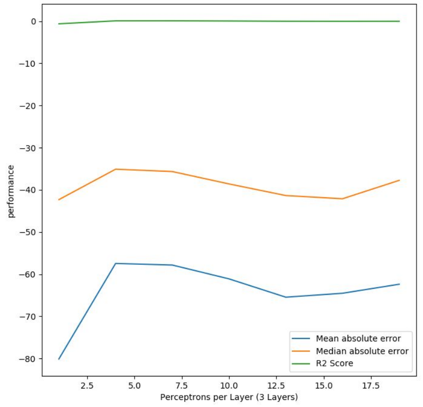


Max. 30

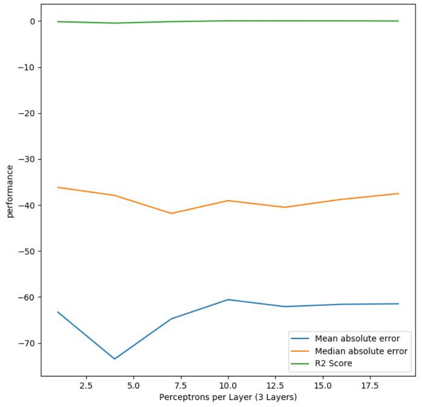


Max. 100

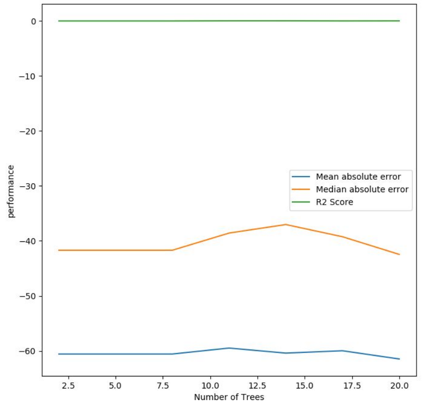
Energy Prediction: Multi-Layer Perceptrons Comparison of diff. Parameters



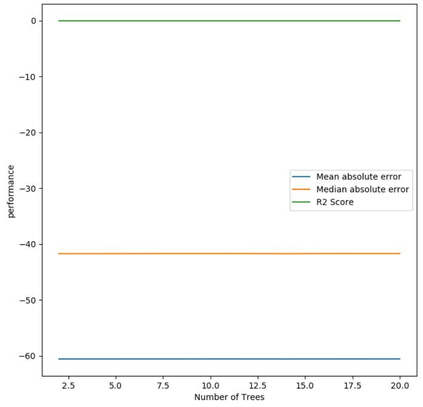
Relu / Adam / Constant



Relu / Adam / Adaptive



Tanh / LBFGS / Adaptive



Tanh / SGD / Adaptive

Format: Activation Function / Solver / Learning Rate

Energy Prediction: Findings

- The MLP Solver SGD (Stochastic Gradient Descent) does not seem to learn more if you assign more perceptrons
- The Random Variables don’t seem to have much impact (except for (better) peaks at low parameter settings)
- kNN performed pretty well (after all, “close neighbours” should yield a similar outcome too)
- Pruning the Trees in Random Forest can yield increased performance over unpruned Trees
- Once again, Pre-Processing brings a delicate bonus

Space Shuttle O-Ring: Overview

- 23 rows, 4 columns
- No missing values
- Predict: number of O-rings that will experience thermal distress
- Features:
  - #O-rings at risk on a given flight, Launch temperature, Leak-check pressure, Temporal order of flight

this

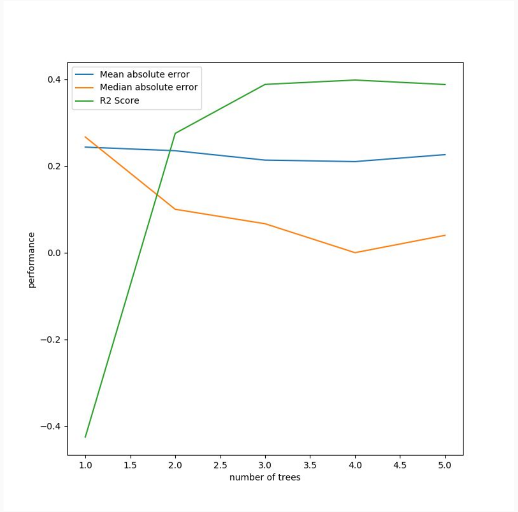
not this

https://archive.ics.uci.edu/ml/assets/MLimages/Large92.jpg
http://static.tvtropes.org/pmwiki/pub/images/the_thin_man_3650.jpg

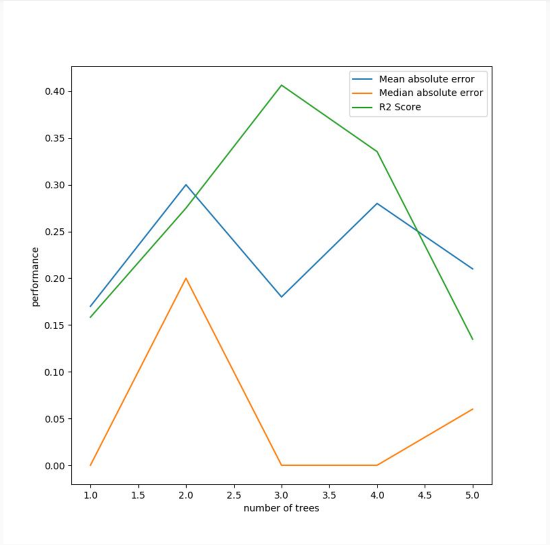
Space Shuttle O-Ring: Characteristics of the Dataset

- Low feature dimensionality and a low number of instances. → It is interesting how the different ML techniques can handle it.
- All the models were evaluated with a 5-fold-cross validation.
- Feature selection was used, so that the model considers only the important features.
  - default feature selector in sklearn (SelectFromModel) rates the feature "temperature" as the only important feature.
  - → compare the performance of these resulting models against the models with all features

Space Shuttle O-Ring: Regressors



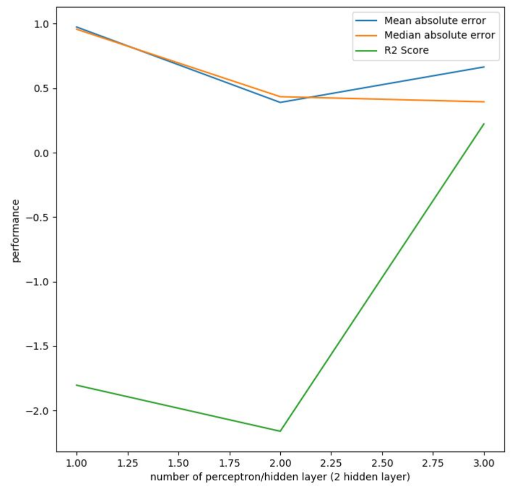
Random Forest with feature selection



Random Forests

Considering the R2 Score feature selection does not increase the max performance significantly. But feature selection makes the performance more stable over the range of the different #trees.

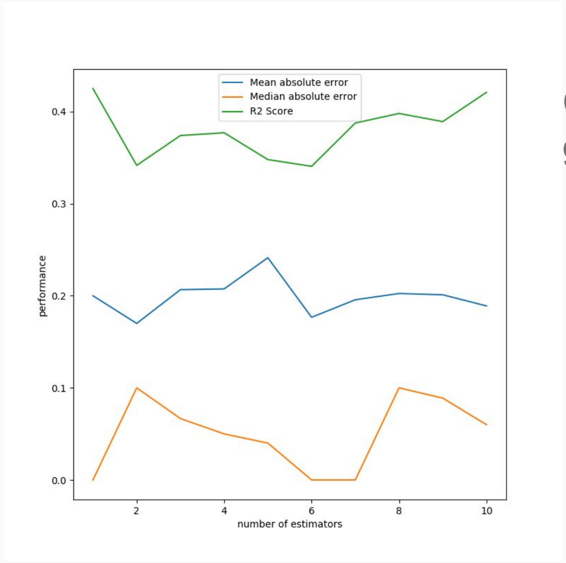
Space Shuttle O-Ring: Regressors



Multilayer perceptron

The left picture depicts the best performance we can achieve with a multilayer perceptron (=MLP) approach. It uses tanh as activation function and uses only the feature “temperature”. The dataset is scaled and normalized. Although MLP is pretty hyped these days this result shows that MLP is not superior to all other ML techniques. We think the reason for the weak performance on this dataset is because it contains not enough instances for the MLP approach.

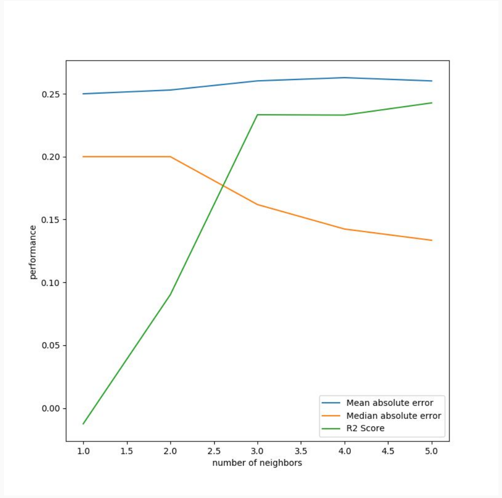
Space Shuttle O-Ring: Regressors



Bagging regressor

Considering the R2 score Bagging regressor gives the best performance on this dataset.

The performance of kNN is in the midrange. It uses only the feature “temperature” and no data-preparation steps.



kNN

Auto MPG: Overview

- 199 rows, 9 columns
- A few missing values
- Predict: MPG (Miles per Gallon) for a car
- Features:
  - Cylinders, Displacement, Horsepower, Weight, Acceleration, Year of Production, Continent of Origin

https://blondethoughtsthatflow.files.wordpress.com/2011/05/cars1.jpg

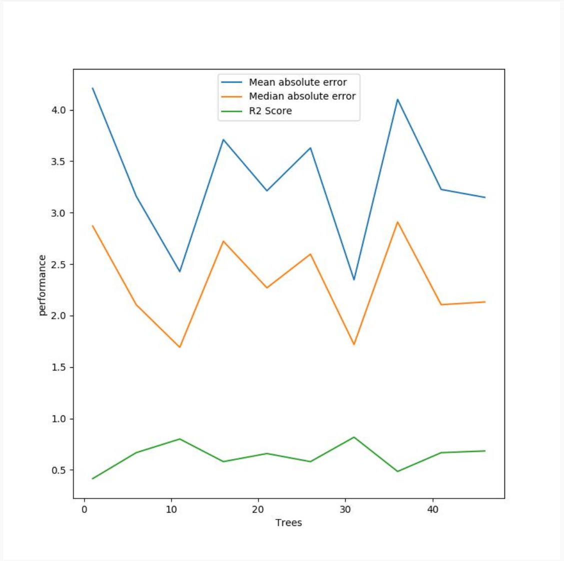
Auto MPG: Characteristics of the Dataset

- Low feature dimensionality and a low number of instances. → It is interesting how the different ML techniques can handle it.

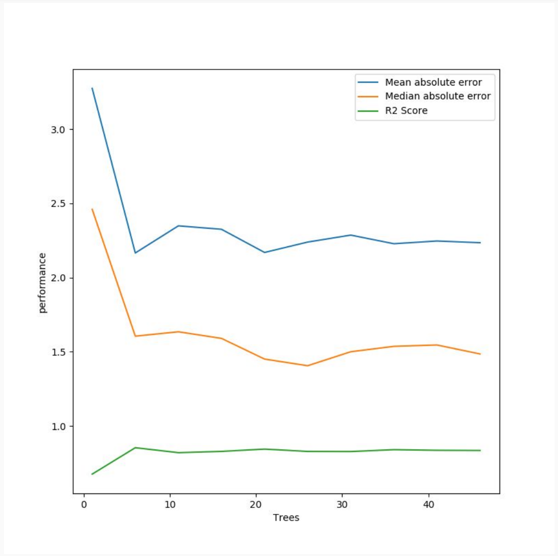
- All the models were evaluated with a 10-fold-cross validation.

- Feature selection was used, so that the model considers only the important features.
    - Rating via default feature selector in sklearn (SelectFromModel)
    - Important features: cylinders, displacement, weight, modelYear

Auto MPG: Regressors



Random Forest with feature selection

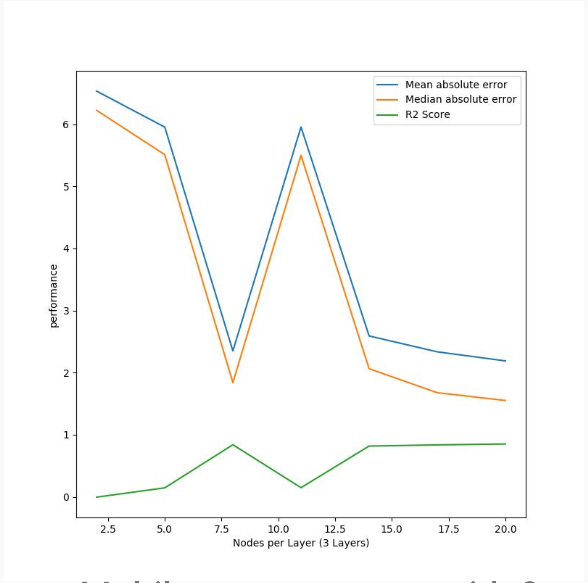


Random Forests without feature seletction

It does not matter which measure we look at, **feature selection worsens the results**. Since our dataset is rather small, the performance gain is little, so feature selection does not matter for our dataset.

Random forest yields the best results for this dataset. We are **rank 3 on Kaggle**.

Auto MPG: Regressors



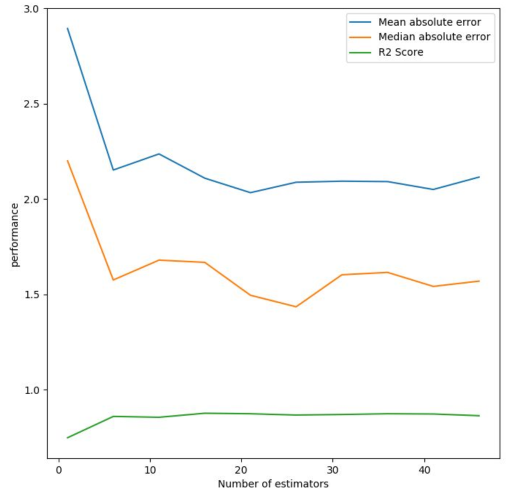
Multilayer perceptron with 3 layers

The left picture depicts the best performance we can achieve with a Multi-Layer Perceptron (=MLP) approach. The **performance of the MLP depends very much on the parameters**.

For the plot on the left side, we used “**relu**” as activation function, “**lbfgs**” as solver, and “**invscaling**” as learning rate. With the right parameters, MLP performs almost as well as random forests.

For activation functions, “logistic” and “tanh” yield very bad results. For solvers, the performance difference is not that big, but “invscaling” yields better results than “adaptive” and “constant”. For solvers, only “lbfgs” yields good results.

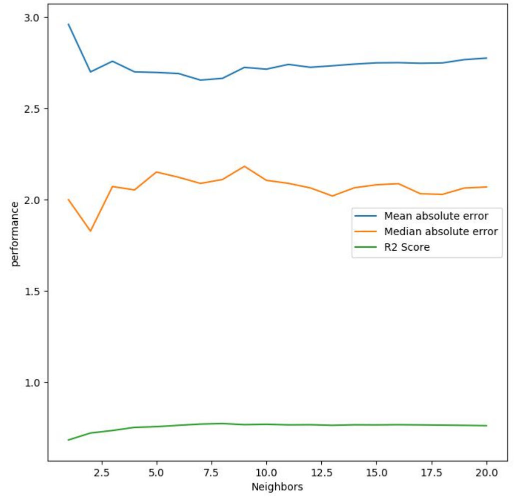
Auto MPG: Regressors



Bagging regressor

Considering the R2 score, the bagging regressor has the **second best performance** (after random forests).

kNN performance is also good for this dataset. As there are few rows in this dataset, the performance increases up until 5 neighbours and then remains almost constant.



kNN

Auto MPG: Conclusion

- Missing values: does not make much difference, since there are so few
- Important features vs. all features: **all features yields better results**
- **Normalization: big influence on the result**. helps kNN & bagging, but worsens forest & neural
- Scaling: does not change the results by much
- Algorithm parameters: very important for MLP.

KDD Cup:
Overview

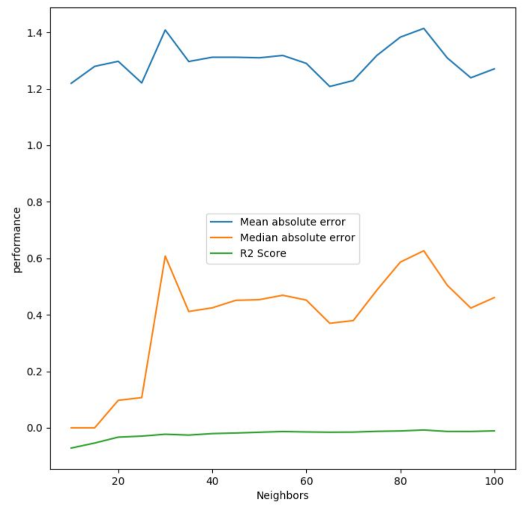
- 2500 rows, 479 columns
- Many missing values, String values and numbers mixed (also in the same column)
- Dataset is from a Data Mining Tools Competition. The task is to predict the amount of a donation.
- Features of persons, but the column names are mostly cryptic.

http://statconsulting.nazwa.pl/instalator/wordpress/wp-content/uploads/2015/08/kddlogo.png

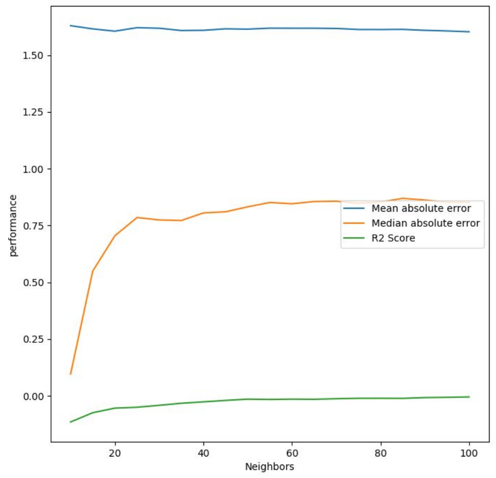
KDD Cup:
Characteristics of the Dataset

- Data quality is low, many missing values → preprocessing is important!
- Many rows + features → runtime of regressors can be significant
- All the models were evaluated with a 5-fold-cross validation.
- Feature selection was used, so that the model considers only the important features.
    - Default feature selector in sklearn (SelectFromModel)
    - Runtime can be significantly reduced!
    - → compare the performance of these resulting models against the models with all features

KDD Cup:
Regressors



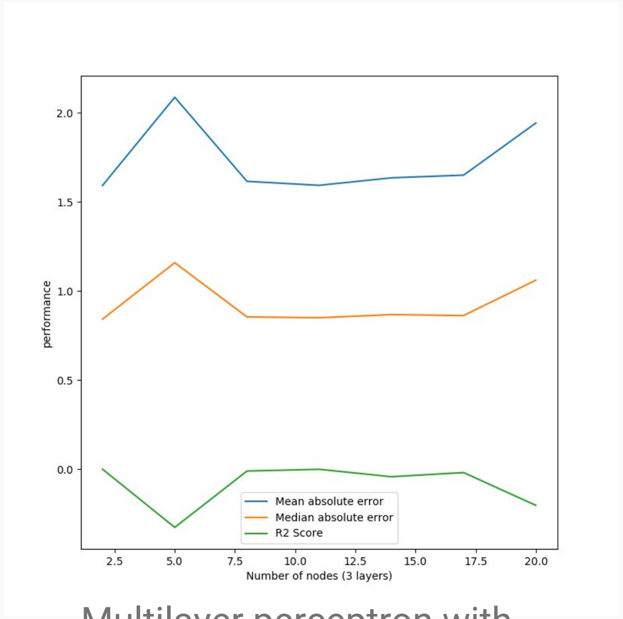
kNN with feature selection



kNN with all features

Feature selection decreases the runtime drastically. I aimed for a small number of important features compared to the original number of features (30 vs. 479).
kNN yields the best performance for this dataset. We are **rank 1 in Kaggle** with kNN!

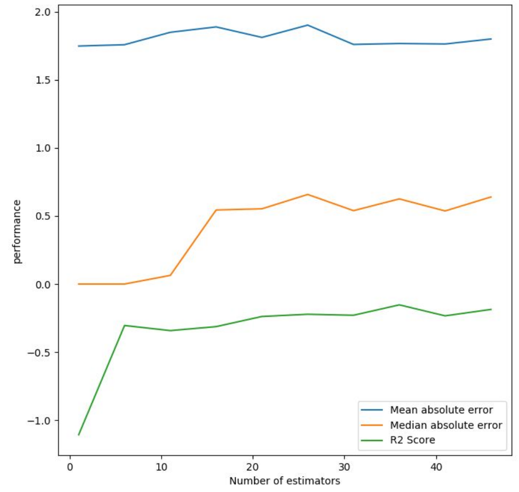
KDD Cup:
Regressors



Multilayer perceptron with feature selection

The left picture depicts the best performance we can achieve with a multilayer perceptron (=MLP) approach. We used feature selection. The activation function was “relu”, scaling was “invscaling”.
Again, the **MLP algorithm’s performance is very dependent on its parameters**.

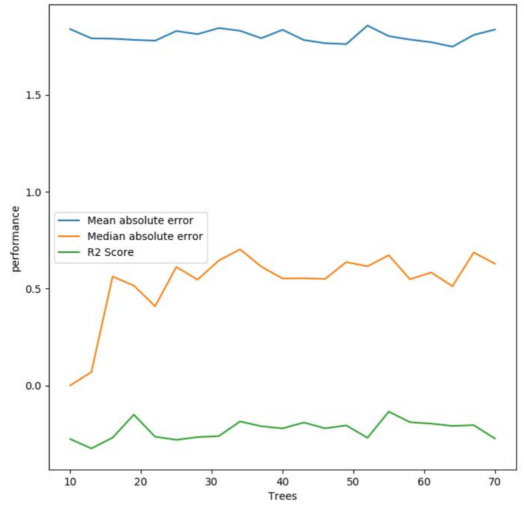
KDD Cup:
Regressors



Bagging regressor with feature selection

Considering the R2 score Bagging regressor gives an O.K. performance. The performance stagnates from ~25 estimators onwards.

The performance of random forest is good, the downside is that it takes very long. Setting the number of trees to ~30 should be high enough.



Random forest with feature selection

KDD Cup:
Conclusions

- Missing values: we **cannot delete the rows here, because every row contains missing values**. The different replacement strategies (median, mean, most_frequent) make no real difference.
- Important features vs. all features: makes almost no difference, until we reduce the features to lower than ~15. but the **speedup is massive**.
- Normalization: does not change the results by much
- Scaling: does not change the results by much
- Algorithm parameters: very important for MLP

Conclusions

Concluding Remarks

- The rankings of the various Regressors in performance differ from Dataset to Dataset
- Playing around with the parameters of a Regressor can make a big difference
- All of our Regressors had an acceptable performance, but different strengths/weaknesses (i.e. there is no silver bullet)
- Preprocessing data can improve the performance of a Regressor and even reduce the required run-time (e.g. through Feature Selection → less columns to be considered = less time spent on calculations)
- Handling missing values is not always easy, but can be the key to making a Dataset usable in the first place (e.g. KDD Cup)