Machine Learning (184.702)
Assignment 3
Large-Scale Evaluation: Analysis of Predicted Generalisation Power / Stability
Report

Gastegger Mario - 0726289
Götzinger Maximilian - 0826279
Hiess Irene - 1326056

January 2018

1 Task

In this assignment, we have to conduct a large-scale evaluation. Whereas, the execution and evaluation of experiments should be automated. The implemented system should provide a big number of algorithms and datasets in order to produce interesting results. All results should be stored in the `arff`-format and contain:

- Metrics
- Graphs
- Meta-data such as
  - Date and time of the execution
  - Experiment configuration
  - Execution time

The specific task is to analyze the predicted generalization power and stability of various classifiers. The comparison of the performances of two or more different test sets (hold-out method) gives information about the generalization power. Afterward, the different algorithms are compared concerning generalization power to analyze their reliability. To find the right test size, evaluation should be performed with a number of different test set sizes. Furthermore, experiments should be performed on several different datasets, with different characteristics.

2 Experimentation Environment

The system is written in Python[1] using `scikit-learn`[2]. For loading, processing and visualizing the data, the packages `pandas`[3], `liac-arff`[4], and `matplotlib`[5] are required.

A JSON-formatted file configures the system. An example is shown in Listing 1. The available parameters are explained in Table 1, Table 3 shows the datasets used, and Table 2 shows the algorithms used. When the program is executed by `python3 main.py config.json`, the experiment is conducted according to the specified configuration, and all results are stored in a directory named after the experiment (see Table 1). Graphics are stored in the sub-directory "plots".

The generated files are named after the dataset and algorithm (with index according to the order in the configuration). The `d<index>_<dataset>_e<index>_<algorithm>_results.arff` files contain the attributes:

- fit time
- score time on the train split
- score time on each of the test splits
- the accuracy for each split
- f1 score for each split
- precision and recall (each with macro averaging) for each split

[1]Tested with version 3.5.3
[2]Version 0.19.1
[3]Tested with version 0.22.0
[4]Tested with version 2.1.1
[5]Tested with version 2.0.2

These measurements are done for each train split size, for the number of repetitionsThey are stored in `evaluation_scores.arff` and `evaluation_times.arff`. The file `evaluation_scores_per_algo_dataset.arff` contains statistical calculations for metrics and the mean absolute deviation of the metrics over the test splits. The file `evaluation_scores_per_repetiti` contains the mean absolute deviation between the test splits for every repetition.

The program automatically generates several graphs. It creates error bar plots and boxplots. For the error bars the line values are means and for the error values we used standard deviation. We created graphs from classification accuracies and from the mean absolute deviations (mad) of the accuracies of the used test sets. Equation 1 shows how the mad for a repetition r is calculated.

$$mad_r = \frac{1}{T} * \sum_{i=1}^{T} |accuracy(r, test_i) - \frac{1}{T} * \sum_{t=1}^{T} (accuracy(r, test_t))| \tag{1}$$

Where r is the repetition and T is the number of test sets.

The following graphs are generated:

- `d<index>_<dataset>_e<index>_<algorithm>_acc_errorbar.png`
  - Generated for every dataset and classifier
  - Shows an error bar plot of accuracies on different train set sizes
- `d<index>_<dataset>_e<index>_<algorithm>_mad_errorbar.png`
  - Generated for every dataset and classifier
  - Shows an error bar plot of accuracy mad values on different train set sizes
- `d<index>_<dataset>_train<train size in %>_acc_boxplot.png`
  - Generated for every dataset and train set size
  - Shows a boxplot of accuracies for every classifier
- `d<index>_<dataset>_train<train size in %>_mad_boxplot.png`
  - Generated for every dataset and train set size
  - Shows a boxplot of accuracy mad values for every classifier
- `train<train size in %>_mad_boxplot.png`
  - Generated for every train size
  - Shows a boxplot of accuracy mad values for every classifier
- `e<index>_<algorithm>_acc_errorbar.png`
  - Generated for every classifier
  - Shows an error bar plot of accuracy values on different train sizes
  - Not used in the report
- `e<index>_<algorithm>_acc_errorbar.png`
  - Generated for every classifier
  - Shows an error bar plot of accuracy mad values on different train sizes
  - Not used in the report

The configuration of the actual experiment and the algorithm used is also JSON-formatted. The file `metadata.json` is also JSON-formatted and contains the name of the experiment, start date, total duration of execution and the name of the experiment configuration file.

Listing 1: Example configuration for an experiment.

```
{ "experiment": "example",
  "stratify": true,
  "repetitions": 2,
  "datasets": [
    "iris.arff",
    "diabetes.arff"
  ],
  "random_state": 12345,
```

Table 1: Explanation of the configuration parameters.

Parameter	Explanation
experiment	The title of the experiment. Name of the output directory.
stratify	Whether or not to create stratified splits.
repetitions	Number of times an algorithm is applied to a dataset.
datasets	List of datasets files (.arff).
random_state	Random state to be used for the splits.
train_size	List of train sizes. The algorithm is applied to a dataset with each train size.
test_splits	Number of test sets. The size of each split is determined by the size of the train split and the number of test splits.
estimators	List of algorithms to be applied to every dataset.
estimators.estimator	Name of the algorithm.
estimators.params	Algorithm parameters (only non-object values are allowed). The random_state should be specified where-ever available to ensure reproducibility.

Table 2: Characteristics of the available datasets.

Algorithm	Based on
RandomForestClassifier	Decision Tree
ExtraTreesClassifier	Decision Tree
DecisionTreeClassifier	Decision Tree
AdaBoostClassifier	Decision Tree
GaussianNB	Naive Bayes
KNeighborsClassifier	Nearest Neighbor
SVC	Support Vector Machines
GaussianProcessClassifier	Laplace Approximation
MLPClassifier	Perceptron
Perceptron	Perceptron

```
  "train_size": [ 0.6 ],
  "test_splits": 2,
  "estimators": [
    { "estimator": "RandomForestClassifier",
      "params": {
        "random_state": 12345,
        "n_estimators": 500
      }
    },
    { "estimator": "RandomForestClassifier",
      "params": {
        "random_state": 12345,
        "n_estimators": 500
      }
    }
  ]
}
```

2.1 Classifiers

To compare the generalization power of different algorithms we made 10 classification algorithms available for experimentation. Table 2 shows the available algorithms and on what they are based. In particular, we implemented several decision trees based ensemble classifiers, to compare their generalization power.

2.2 Datasets

In our experimentation software, we implemented 12 different datasets. All datasets can be obtained from the OpenML [1][6] (Some of the datasets are also available on the UCI Machine Learning Repository [2]). The preprocessing for each dataset was kept to the minimum. Nominal features are label encoded and missing values are either replaced by zero (speeddating) or instances are dropped (hepatitis). The number of instances is ranging from 80 to 11.000, the number of attributes is ranging from 4 to 122 and the number of classes is ranging from 2 to 7.

[6] https://www.openml.org/

Table 3: Characteristics of the available datasets.

Dataset	# instances		# features	Nominal attributes	Missing values	# classes
	Original	After prep.				
speeddating[7]	8378	8378	122	Yes	Yes	2
mammography[8]	11183	11183	6	No	No	2
iris[9]	150	150	4	No	No	3
climate-model-simulation-crashes [3][10]	540	540	20	No	No	2
diabetes[11]	768	768	8	No	No	2
ilpd[12]	583	583	10	Yes	No	2
kc2[13]	522	522	21	No	No	2
steel-plates-fault[14]	1941	1941	27	No	No	7
segment[15]	2310	2310	19	No	No	7
hepatitis[16]	155	80	19	Yes	Yes	2
ringnorm[17]	7400	7400	20	No	No	2
credit-g[18]	1000	1000	20	Yes	No	2

Table 4: Sizes of train and test sets used for experiment 1

size of train set	number of test sets	size of test sets
30%	2	35%
40%	2	30%
50%	2	25%
60%	2	20%

Table 3 summarizes some characteristics of the available datasets.

3 Experiment1

3.1 Experiment Setup

In experiment 1 we evaluated all classifiers listed in Section 2.1 on all datasets listed in Section 2.2. For this experiment, we used the default configurations of the classifiers; only the random seed was set if necessary. For every step of the experiment, we split the datasets into one training set and two test sets. For the training sets, we used four different sizes such as 30%, 40%, 50%, and 60%. For the test sets we divided the remaining samples into two equally sized sets. The test set sizes corresponding to the training set sizes can be seen in Table 4. For every dataset and train set size we did ten repetitions, where we used different random seeds for the creation of the train and test sets. The exact configuration of the experiment can be found in the file `experiment_2-test-sets/config.json`.

3.2 Results and Analysis

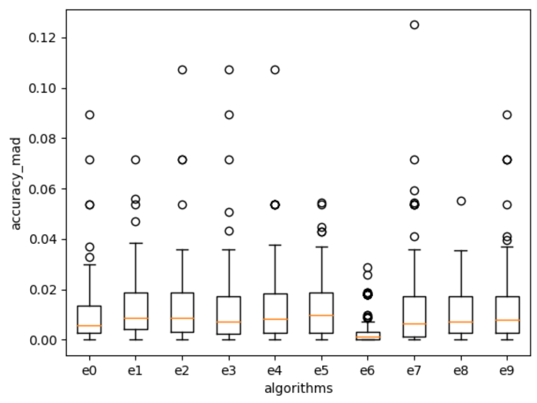
Figure 1 shows boxplots of the mad of the accuracies of the used test sets, of all repetitions on all datasets. The figure shows a boxplot for every train set size and every algorithm.

It can be seen that the classifier e6 (SVC) has by far the lowest deviations on all train set sizes. Algorithm e8 (MLPClassifier) also seems to be reliable. The mean deviation of MLPClassifier is as high as the mean deviation of the other classifiers, but with MLP there are no outliers with higher mad values. The highest maximum deviation between the test sets occurred with a train set size of 0.5 where algorithm e4 (GaussianNB) has a maximum mad value of about 0.2. With a train set size of 0.3 the highest mad value was about 0.125 with algorithm e7 (GaussianProcessClassifier).

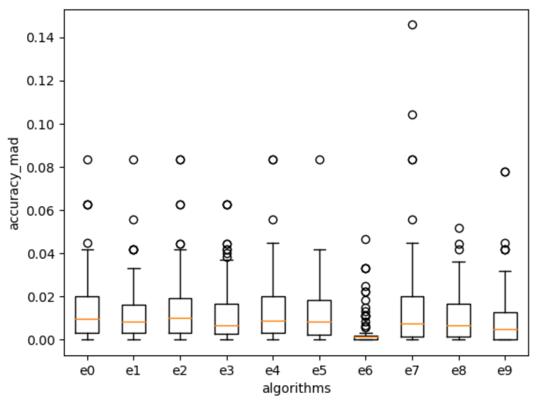
Which training set size is best seems to depend on both the classifier and the dataset. For us, there is no clear pattern recognizable, but apparently, the choice of the dataset has a stronger impact on the best train set size than the choice of the algorithm.

Figure 2 shows the mean mad of the test set accuracies of the ExtraTreesClassifier on some of the datasets. While a training set size of 0.3 works very good regarding generalization power for the speed dating dataset, it seems to be a bad choice for the other datasets. A training set size of 0.5 is fine for the mammography and steel-plates-fault datasets, but bad for the speed dating dataset. For other algorithms there is also no identifiable best training set size. Furthermore, a better generalization power often corresponds to a worse accuracy. Figure 3 shows the accuracies for the datasets used in Figure 2, whereas a training set size of 0.6 leads to the highest accuracies. However, Figure 2 shows the generalization errors are higher with that training set size.

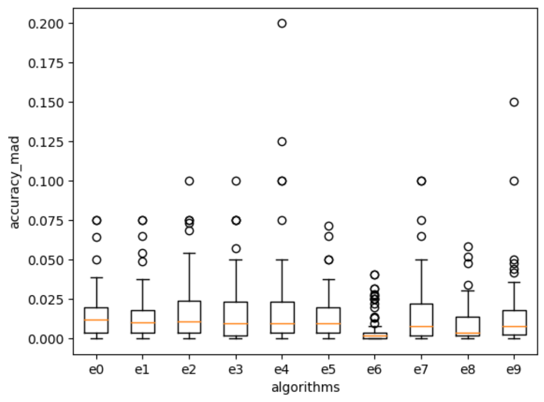
Figure 4 shows the error bar plots of the mean absolute deviation of the accuracies, of all repetitions, of the mammography dataset with different algorithms. It is again impossible to choose the best training set size. While a training set size of 0.4 is best for the RandomForestClassifier, a size of 0.5 is much better for the DecisionTreeClassifier. The most algorithms seem less reliable on this dataset when the training set has a size of 0.3 and 0.6. Figure



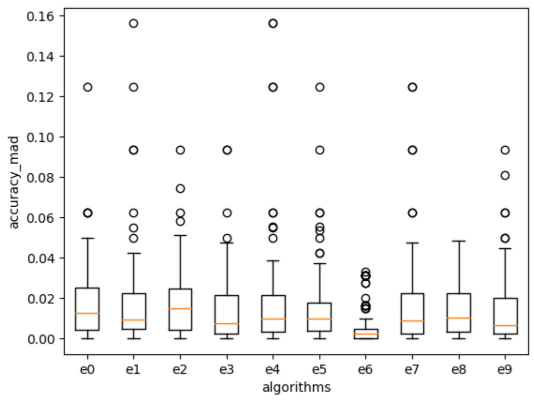
(a) Train size 0.3



(b) Train size 0.4

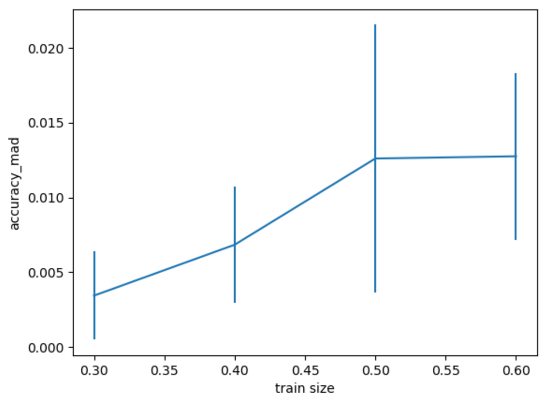


(c) Train size 0.5

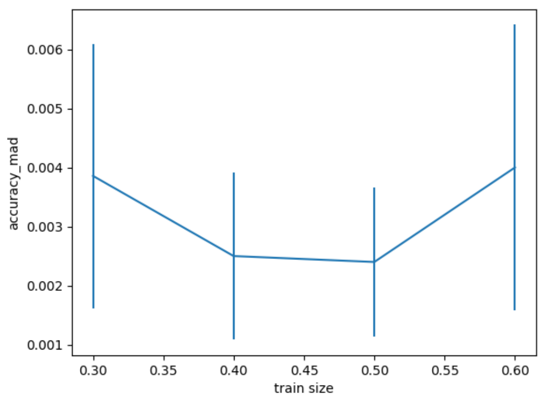


(d) Train size 0.6

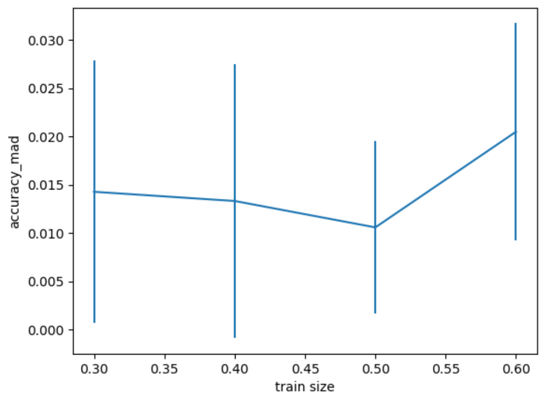
Figure 1: Boxplots of mad between the accuracies of the test sets for experiment 1. There are boxplots for every used train set size and for every used algorithm. The boxplots contain the results from all datasets.



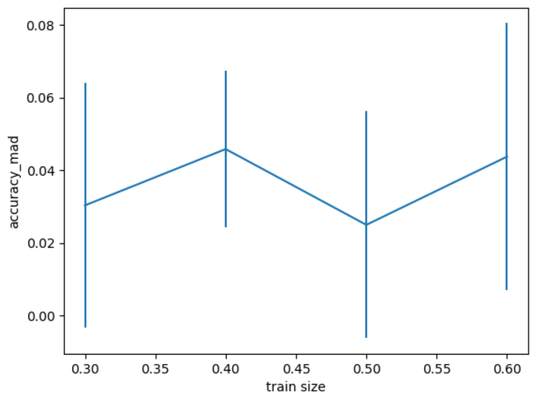
(a) Speeddating dataset



(b) Mammography dataset

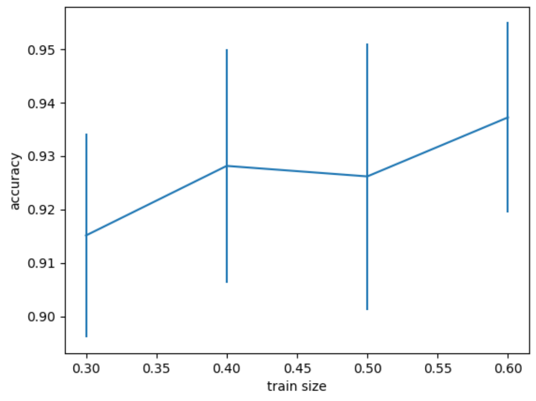


(c) Steel-plates-fault dataset

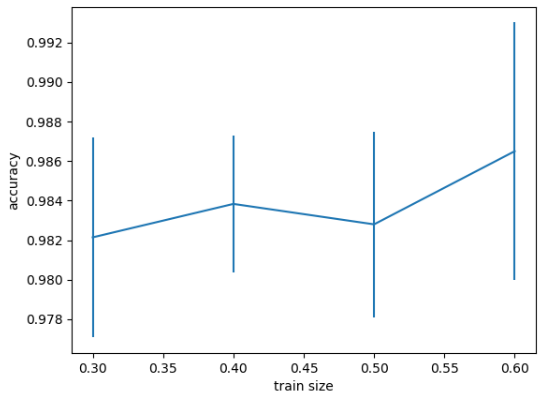


(d) Hepatitis dataset

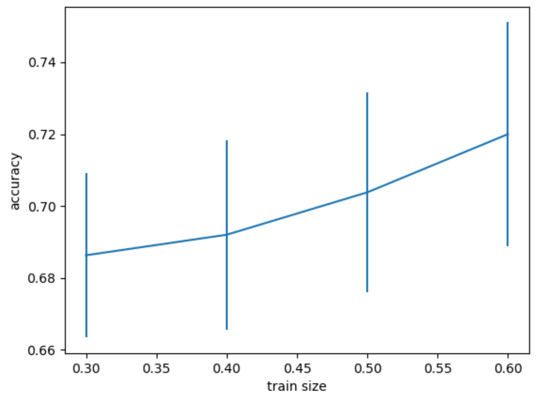
Figure 2: Error bar plots of mad between the accuracies of the test sets achieved by the ExtraTreesClassifier on a subset of the used datasets in experiment 1.



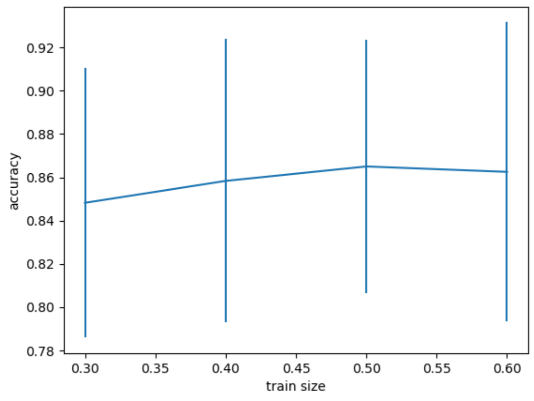
(a) Speeddating dataset



(b) Mammography dataset



(c) Steel-plates-fault dataset



(d) Hepatitis dataset

Figure 3: Error bar plots of accuracies of the test sets achieved by the ExtraTreesClassifier on a subset of the used datasets in experiment 1.

Table 5: Sizes of train and test sets used for experiment 2

size of train set	number of test sets	size of test sets
30%	4	17.5%
40%	4	15%
50%	4	12.5%
60%	4	10%

5 shows the accuracies for the classifiers on the mammography dataset. Again, the highest accuracies often correspond to the highest generalization error (e.g., ExtraTreesClassifier and RandomForestClassifier). For Perceptron the train size 0.4 with the highest accuracy also gives the lowest mad between the accuracies on the mammography dataset, but on other datasets that is not the case.

Figure 1 contains the performance measurements for all datasets combined.It shows that algorithm e6 (SVC) generalizes better than the other classifiers and that the mean accuracy mad values of the other algorithms are more or less the same over all datasets. It is necessary to have a look at single datasets to compare the other classifiers. Figure 6 shows the accuracies as well as the deviations between the accuracies of the two test sets for the speeddating dataset with a train set size of 0.4. On this dataset algorithms, e2 (DecisionTreeClassifier) and e3 (AdaBoostClassifier) perform best. They achieve a classification accuracy of 1 in every repetition, and therefore, they have a deviation of 0. Algorithm e4 (GaussianNB) also performs highly accurate, but the deviation is greater than 0. The classifier SVC performed much worse, but it still has only small deviations between the test sets. So although the accuracy of SVC is lower than the accuracy of GaussianNB, it is still more reliable. Another noticeable algorithm is e9 (Perceptron). The Perceptron classifier seems to be the least reliable of all classifiers tested. Figure 6 (a) shows that the Perceptron classifier has the biggest box out of the boxplots. That means that the accuracy results of the Perceptron algorithm were further apart than the accuracies of the other algorithms. Interestingly, the Perceptron classifier is not the classifier with the highest mean mad values, that is classifier e7 (GaussianProcessClassifier). Although the accuracies of the Perceptron classifier were further apart than the accuracies of the GaussianProcessClassifier, the deviations between the two created test sets were smaller. In contrast, Figure 7 shows the boxplots for the accuracy and mad values concerning the mammography dataset. The most classifiers achieved a mean accuracy of 0.98 when processing the mammography dataset. Only e2 (DecisionTreeClassifier) and e4 (GaussianNB) performed worse. These classifiers also show the highest mean deviations of accuracies. Although classifier e2 seemed very reliable on the speed dating dataset, it is one of the least reliable classifiers on the mammography dataset. The algorithms e3 (AdaBoostClassifier), e8 (MLPClassifier), and e9 (Perceptron) also show slightly higher deviations of test set accuracies than the other classifiers, but these deviations are all still below 0.01. The scale of the mad values in Figure 6 goes up to 0.04 while the scale of the mad values in Figure 7 only goes up to 0.014, so altogether the mammography dataset produces more reliable results than the speed dating dataset. For other datasets, the best and worse algorithms are again different.

4 Experiment2

4.1 Experiment Setup

In this experiment, we used the same configuration as in the first experiment, except for the number of test splits which we increased to 4. The resulting size of the train set and the test sets are shown in Table 5. The exact configuration of the experiment can be found in the file `experiment_4-test-sets/config.json`.

4.2 Results and Analysis

Figure 8 shows the mad boxplots for 2 test sets and for 4 test sets next to each other. There is a slight increase of the mad for all algorithms, whereas e6 (SVC) is by far the most stable one.

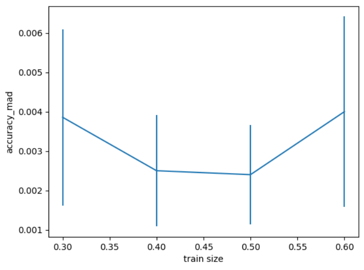
Figure 9 and Figure 10 show the mad and accuracy for the ExtraTreesClassifier, for a subset of the datasets.

For the speed dating dataset, the accuracy for 2 and for 4 test sets is very similar. Regarding mad, the values are higher for 4 test sets. It should be noted that for 4 test sets, the size of each is only half of the size as for 2 test sets. In general, the estimation of generalization power should be more realistic with 4 smaller test sets. For the speed dating dataset, the curve for both test set variants is of similar shape.

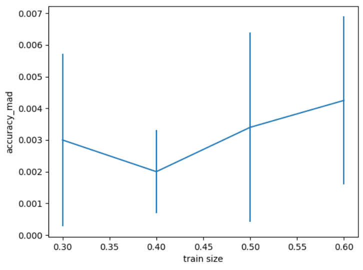
For the mammography dataset, the accuracies of both test set numbers are nearly similar. The values for the 4 test sets are again higher than for the 2 test sets. The shape of the mad looks different. The means and the standard deviation increase monotonously with 4 test sets. However, this difference is within the standard deviation and could be coincidental.

For the steel-plates-fault dataset, again, are the accuracies of both test set numbers nearly similar. The shape of the mad curves looks different. The mean values are similar (besides the, by 0.01, slightly higher mean values) except for the 50% train set size. For the train set sizes 30% and 40%, the standard deviation is similar, but for 50% and 60% the standard deviation for the 4 test sets is higher than then on for 2 test sets.

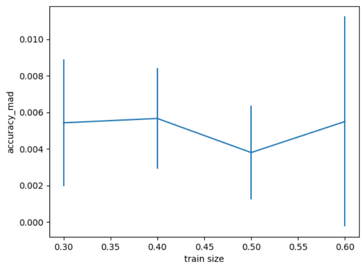
For the hepatitis dataset, both curves are similar besides the positive offset for the 4 test sets values.



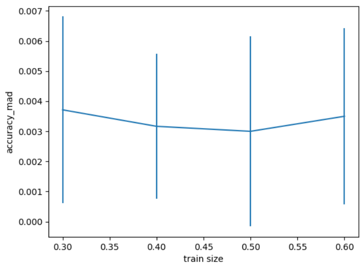
(a) ExtraTreesClassifier



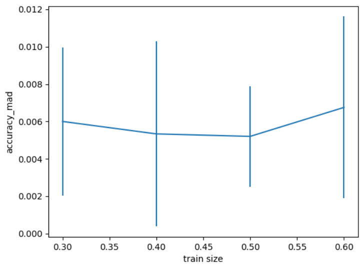
(b) RandomForestClassifier



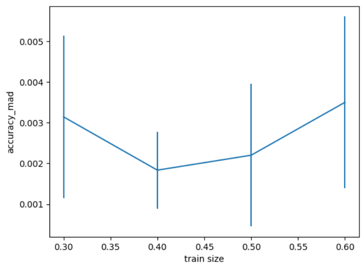
(c) DecisionTreeClassifier



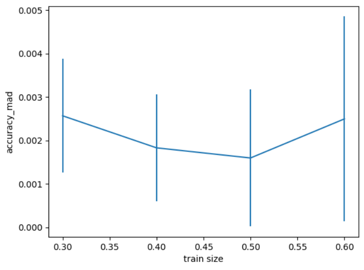
(d) AdaBoostClassifier



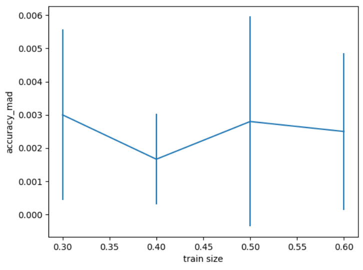
(e) GaussianNB



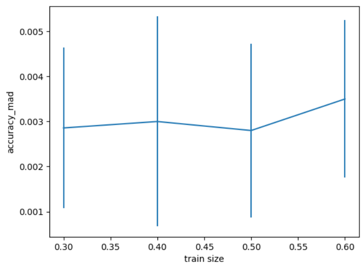
(f) KNeighborsClassifier



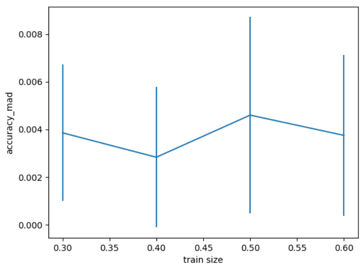
(g) SVC



(h) GaussianProcessClassifier

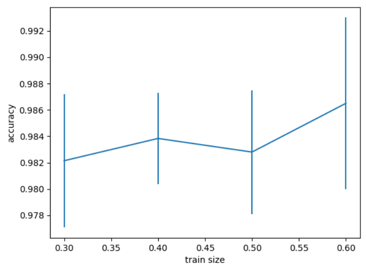


(i) MLPClassifier

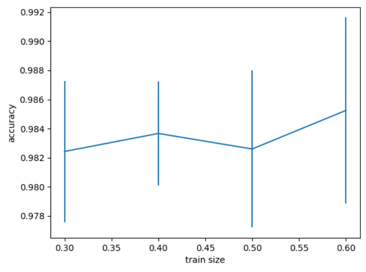


(j) Perceptron

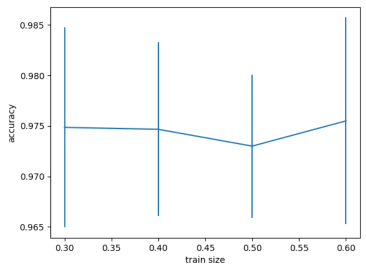
Figure 4: Error bar plots of mad between the accuracies of the test sets on the mammography dataset achieved by different classifiers in experiment 1.



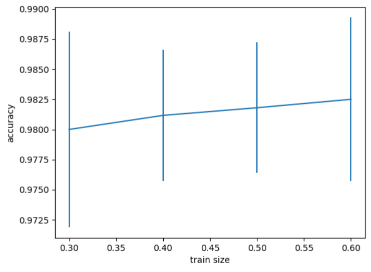
(a) ExtraTreesClassifier



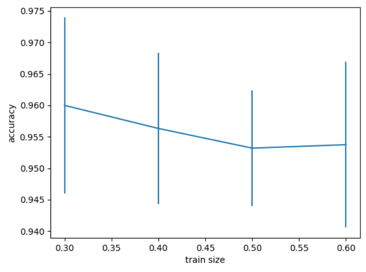
(b) RandomForestClassifier



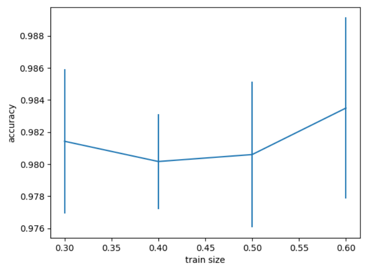
(c) DecisionTreeClassifier



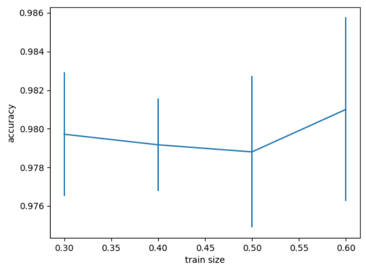
(d) AdaBoostClassifier



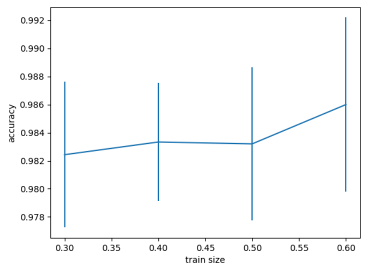
(e) GaussianNB



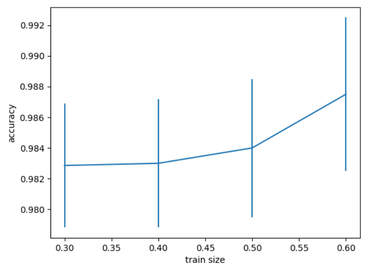
(f) KNeighborsClassifier



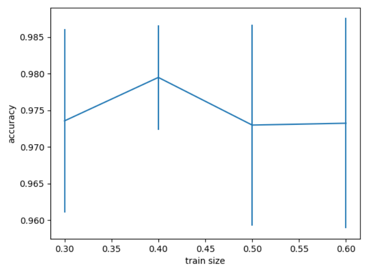
(g) SVC



(h) GaussianProcessClassifier

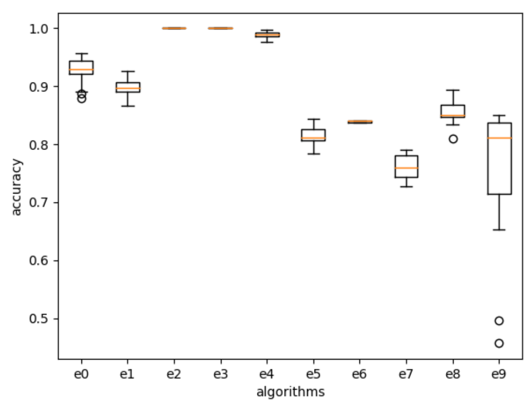


(i) MLPClassifier

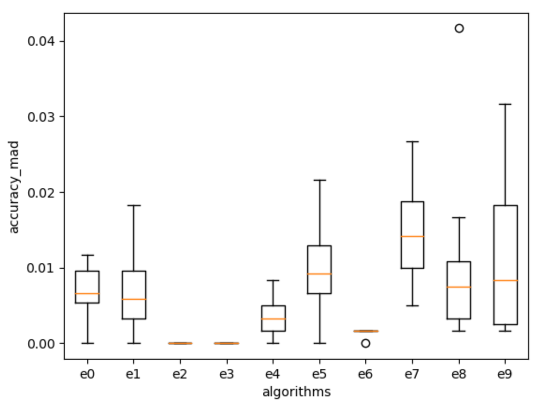


(j) Perceptron

Figure 5: Error bar plots of the accuracies of the test sets on the mammography dataset achieved by different classifiers in experiment 1.

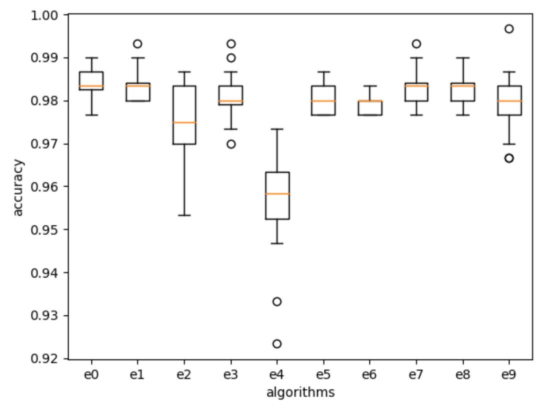


(a) Accuracy

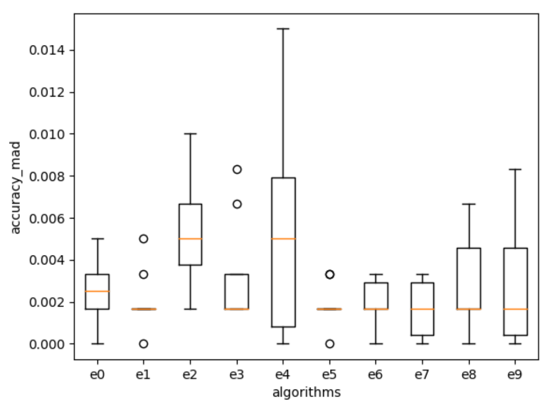


(b) Mad of accuracies

Figure 6: Left are boxplots of the accuracies of each classifier on the speeddating dataset, right are boxplots of the mad between the accuracies of the test sets of the speeddating dataset. The results are from experiment 1 for a train set size of 0.4.

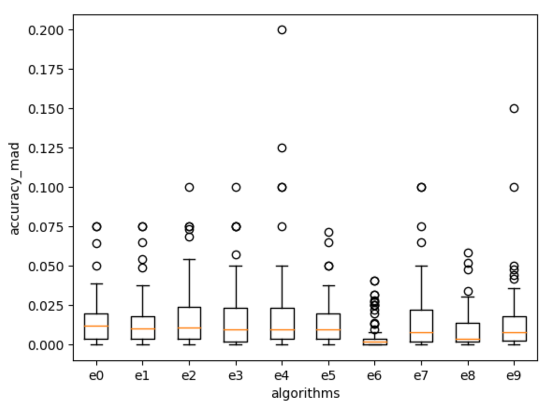


(a) Accuracy

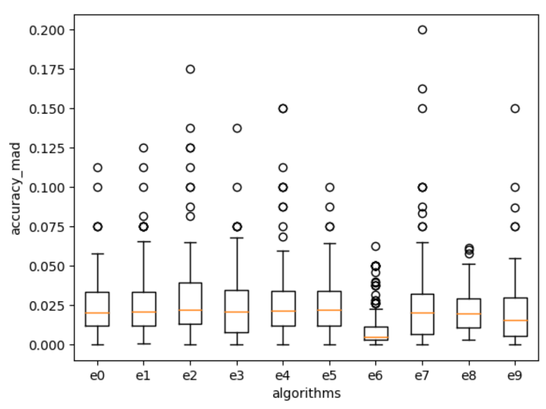


(b) Mad of accuracies

Figure 7: Left are boxplots of the accuracies of each classifier on the mammography dataset, right are boxplots of the mad between the accuracies of the test sets of the mammography dataset. The results are from experiment 1 for a train set size of 0.4.

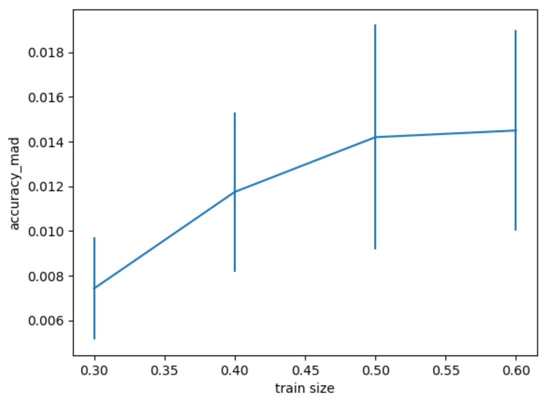


(a) Two test sets

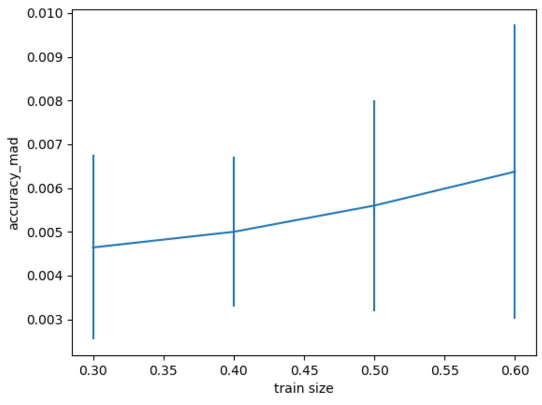


(b) Four test sets

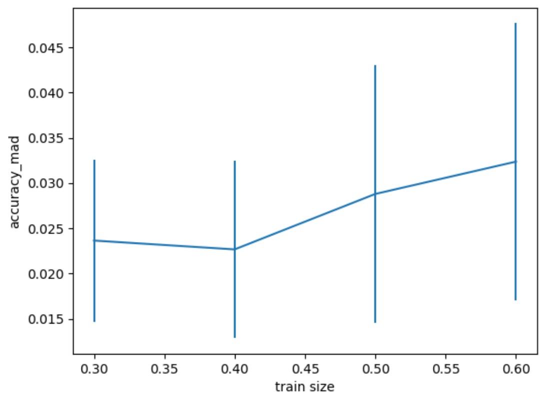
Figure 8: Boxplots of mad between the accuracies of the test sets with 50% train size. There are boxplots for every used algorithm. The boxplots contain the results from all datasets.



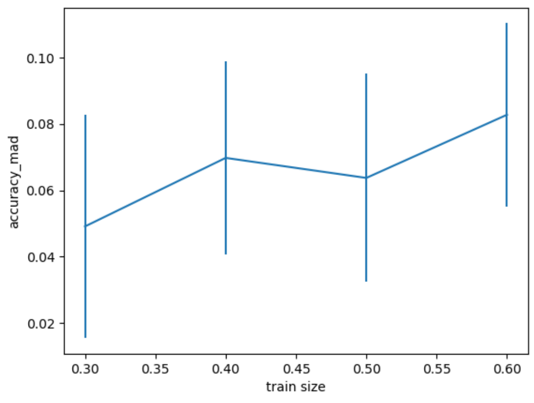
(a) Speeddating dataset



(b) Mammography dataset

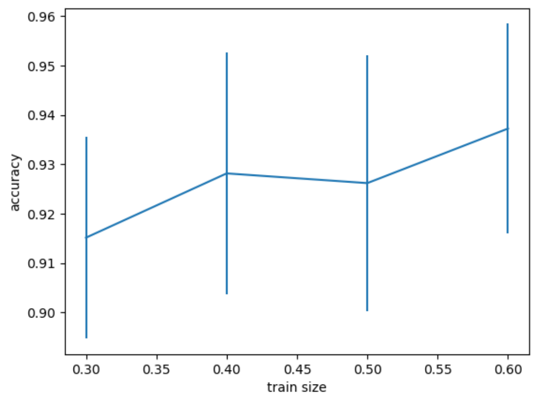


(c) Steel-plates-fault dataset

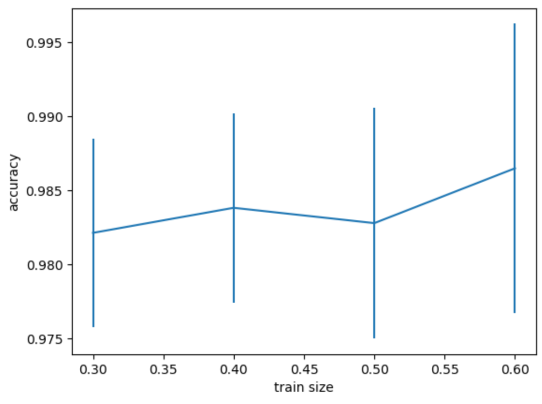


(d) Hepatitis dataset

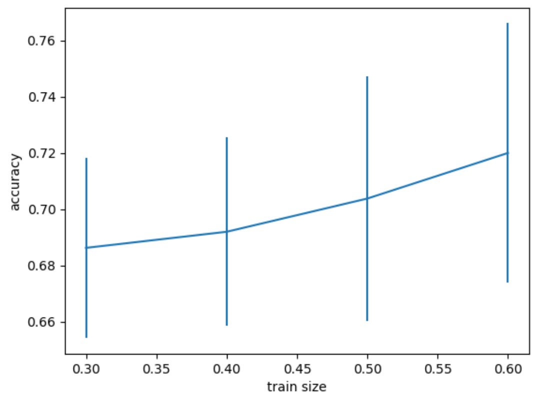
Figure 9: Error bar plots of mad between the accuracies of the test sets achieved by the ExtraTreesClassifier on a subset of the used datasets of 4 test sets.



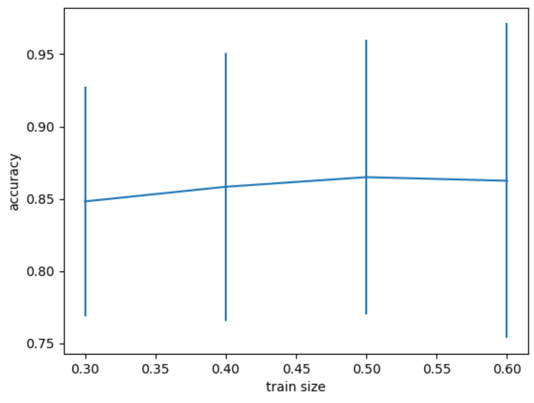
(a) Speeddating dataset



(b) Mammography dataset



(c) Steel-plates-fault dataset



(d) Hepatitis dataset

Figure 10: Error bar plots of accuracies of the test sets achieved by the ExtraTreesClassifier on a subset of the used datasets of 4 test sets.

5 Summary

Out of all the algorithms, the evaluated SVR was the only one that consistently resulted in a low accuracy deviation on the test sets. Apparently, SVR is a classifier that generalizes well on almost all datasets. The other classifiers generalized good on some datasets and bad on others.

Changing the training set size together with the test set size has an impact on the performance values, but there was no recognizable pattern identifiable.

A doubling of the number of test sets (from two to four) and a reducing of the test set sizes lead to increased accuracy mad values (mad). Mad corresponds to the deviation of the accuracies. The deviation might be higher due to the higher number of test sets (four samples instead of two). Another reason could be the decreased size of the test sets. With fewer samples, the classification of a single sample has a bigger effect on the accuracy.

References

[1] J. Vanschoren, J. N. van Rijn, B. Bischl, and L. Torgo, "Openml: Networked science in machine learning," *SIGKDD Explorations*, vol. 15, no. 2, pp. 49–60, 2013. [Online]. Available: http://doi.acm.org/10.1145/2641190.2641198

[2] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[3] D. D. Lucas, R. Klein, J. Tannahill, D. Ivanova, S. Brandon, D. Domyancic, and Y. Zhang, "Failure analysis of parameter-induced simulation crashes in climate models," *Geoscientific Model Development*, vol. 6, no. 4, pp. 1157–1171, 2013. [Online]. Available: https://www.geosci-model-dev.net/6/1157/2013/