Chapter 1

Handling Missing Data in Training Sets

1.0.1 Experimental Setup

To compare the existing algorithms with our proposed approach we use five, very different, datasets: the *Digits* dataset, containing 1797 instances with handwritten digits with 64 variables per instance, from the *Scikit-learn* Python package [3]. The Digits dataset is relatively small, and some attributes are definitely less important than others (every attribute is a gray value of a pixel in an $8 * 8$ image). Most of the outer pixels are often 0. The target class of the dataset is the digit that the instances represents, so $0 - 9$.

As a second dataset we use the Cover Type dataset from the *UCI Machine Learning Repository* [2], containing 581012 instances with 54 attributes and a target ranging from 1 to 7 assigning a certain forest cover type to a region. This dataset is much larger than the Digits dataset. In most of our test runs we do not use the entire Cover Type dataset but only the first 40.000 instances, this is for time-complexity reasons.

The third dataset we use is the *House 16H* dataset from the *Machine Learning Data* repository [1]. This dataset was designed on the basis of data provided by US Census Bureau and consist mostly of cumulated counts at different survey levels. The regression task is to predict the median price of the house in the region on demographic composition and a state of housing market in the region. The dataset contains a broad variety of continues values. The dataset has 22784 records, each with 16 attributes.

The fourth dataset we use is the *Page Blocks* dataset from the *UCI Machine Learning Repository*. This is a classification problem and the goal is to classify all the blocks of the page layout of a document that has been detected by a segmentation process. There are 5473 records, each with 10 attributes.

The last dataset we use is the *Concrete Compressive Strength* dataset from the *UCI Machine Learning Repository*. This dataset contains 1030 instances with 9 attributes. The regression task is to predict the compressive strength of the concrete.

For each dataset we run several experiments with 75% of the attributes containing missing values and 25% of the attributes (randomly chosen) containing no missing values. The amount of missing values in the attributes with missing

data, is set to $10, 20, 30, 40, 50, 60$ percent and for each set-up we run 20 experiments using different random seeds. Both the *MAR* and the *MNAR* case are considered, by running our test also with the datasets where we delete values only with an observed value higher than the median of its attribute.

1.0.2 Performance Indicators

We measured two aspects of the quality of the imputation. First, we estimated, with help of cross-validation, the accuracy of the final model that was trained on the repaired dataset. The accuracy was measured either by the ratio of correctly classified cases (in case of classification) or by the *coefficient of determination*, R^2, (in case of regression):

$$R^2 = 1 - \frac{\sum_i (p_i - y_i)^2}{\sum_i (y_i - \bar{y})^2}$$

where y_i denotes the target value and p_i the predicted value.

This score indicates how well the model fits the test data. The maximal value of R^2 is 1, meaning the perfect fit, lower (even negative) values reflect the error.

Second, we measured the quality of the approximation of the imputed values. As all the imputed variables are numeric, we used the *Root Mean Squared Error*, *RMSE*, to measure the difference between the observed and imputed values:

$$\mathrm{RMSE} = \sqrt{\frac{\sum (v_{\mathrm{observed}} - v_{\mathrm{imputed}})^2}{n}}$$

As a last indicator of algorithm's performance we measured the execution time. For bigger datasets the cpu time might be an issue to consider. Each experiment was repeated 10 times and the results of measurements were averaged.

1.0.3 Results

Each dataset is normalized before further processing. For each dataset, $12 * 20$ tests are executed with all the algorithms and a test dataset with values MAR or values MNAR and an amount of missing values ranging from 10 to 60 percent per variable. For each dataset the accuracy of the final models (R^2 or the ratio of correctly classified cases) and the accuracy of imputation ($RMSE$) can be found in the following subsections. In each table row, the scores are shown averaged over 10 runs with the same settings except for the random seed. The amount of missing values and the type of missing values (MAR or MNAR) are shown as well. Note that for the MNAR test cases we could not always delete the $X\%$ per attribute, since not all attributes have $X\%$ values above the median (where X is ranging from 10 to 60). In these cases the percentage that is shown is the average percentage of deleted values per attribute.

Furthermore, the R^2 and accuracy scores of each dataset are given for different final model algorithms: Random Forests, Support Vector Machines and Gradient Boosting Forests. This is to show that the R^2 and accuracy scores are dependent on the regressor or classifier that is being used and that this not always reflect the quality of the imputation itself.

For the R^2 and accuracy scores, also a Ref. score is given. This score is calculated by training a model on the complete dataset without missing values. The results from the algorithms after the Ref. score are from left to right; Imputation by Mean, Imputation by Median, Imputation by Most Frequent, Predictive Value Imputation using 2-Nearest Neighbour over a dataset imputed by the Mean, Regression Imputation using Random Forests and last but not least, our own algorithm: IARI. Entries in bold are significantly better than all other entries with the same settings. The significance is tested using a T test where a result is significant if $p < 0.5$. If there is no bold entry in the row this means that none of the results where significantly better than the others.

Cover Type Dataset Results

In Tables 1.1 to 1.3 and 1.4 the accuracy of the models (Accuracy Score) and the quality of imputation ($RMSE$) are shown for the imputation algorithms on 40.000 instances of the Cover Type dataset.

Table 1.1: Model Accuracy Score on the Cover Type Dataset with 40000 instances using Random Forests

Miss.%	Type	Ref.	Mean	Median	Freq.	PVI NN	RI	IARI
4	MNAR	0.911	0.887	0.879	0.876	0.886	0.886	**0.893**
6	MNAR	0.911	0.871	0.864	0.860	0.868	0.868	**0.881**
8	MNAR	0.911	0.850	0.841	0.835	0.844	0.845	**0.864**
10	MNAR	0.911	0.815	0.809	0.803	0.806	0.805	**0.839**
12	MNAR	0.911	0.670	0.678	0.656	0.657	0.663	**0.693**
10	MAR	0.911	0.893	0.899	0.900	0.900	0.896	**0.907**
20	MAR	0.911	0.874	0.887	0.886	0.883	0.880	**0.899**
30	MAR	0.911	0.857	0.874	0.874	0.866	0.863	**0.889**
40	MAR	0.911	0.834	0.859	0.858	0.845	0.845	**0.878**
50	MAR	0.911	0.808	0.841	0.838	0.819	0.822	**0.864**
60	MAR	0.911	0.776	0.824	0.822	0.787	0.799	**0.847**

Table 1.2: Model Accuracy Score on the Cover Type Dataset with 40000 instances using Support Vector Machines

Miss.%	Type	Ref.	Mean	Median	Freq.	PVI NN	RI	IARI
4	MNAR	0.819	0.746	0.741	0.724	0.746	0.747	0.748
6	MNAR	0.819	0.723	0.717	0.696	0.723	0.727	**0.736**
8	MNAR	0.819	0.704	0.689	0.675	0.704	0.708	**0.735**
10	MNAR	0.819	0.672	0.666	0.662	0.671	0.678	**0.731**
12	MNAR	0.819	0.613	**0.621**	0.593	0.612	0.612	0.602
10	MAR	0.819	0.813	0.813	0.810	0.815	0.815	**0.819**
20	MAR	0.819	0.804	0.806	0.799	0.807	0.810	**0.819**
30	MAR	0.819	0.795	0.797	0.785	0.797	0.803	**0.817**
40	MAR	0.819	0.787	0.788	0.769	0.788	0.794	**0.814**
50	MAR	0.819	0.778	0.778	0.755	0.779	0.784	**0.809**
60	MAR	0.819	0.765	0.764	0.747	0.767	0.776	**0.802**

Table 1.3: Model Accuracy Score on the Cover Type Dataset with 40000 instances using Gradient Boosting Trees

Miss.%	Type	Ref.	Mean	Median	Freq.	PVI NN	RI	IARI
4	MNAR	0.787	0.778	0.776	0.761	0.778	0.777	0.779
6	MNAR	0.787	0.774	0.769	0.747	0.772	0.771	**0.778**
8	MNAR	0.787	0.767	0.758	0.745	0.767	0.765	**0.773**
10	MNAR	0.787	0.753	0.748	0.743	0.751	0.750	**0.764**
12	MNAR	0.787	0.695	0.694	**0.717**	0.687	0.690	0.691
10	MAR	0.787	0.785	0.785	0.773	0.785	0.784	0.785
20	MAR	0.787	0.782	0.783	0.771	0.783	0.778	**0.784**
30	MAR	0.787	0.778	0.778	0.765	0.779	0.777	**0.783**
40	MAR	0.787	0.774	0.774	0.763	0.775	0.774	**0.780**
50	MAR	0.787	0.770	0.770	0.760	0.770	0.766	**0.778**
60	MAR	0.787	0.766	0.767	0.755	0.767	0.758	**0.774**

Table 1.4: Imputation Quality (RMSE) of each Imputation Algorithm on the CoverType dataset with 40000 instances

Miss.%	Type	Mean	Median	Freq.	PVI NN	RI	IARI
4	MNAR	0.755	0.762	0.774	0.755	0.745	**0.721**
6	MNAR	0.786	0.795	0.813	0.786	0.776	**0.760**
8	MNAR	0.814	0.823	0.842	0.814	0.802	**0.771**
10	MNAR	0.848	0.852	0.867	0.847	0.838	**0.791**
12	MNAR	0.894	0.884	0.889	0.894	0.894	**0.877**
10	MAR	0.271	0.277	0.294	0.261	0.225	**0.176**
20	MAR	0.380	0.389	0.414	0.370	0.330	**0.266**
30	MAR	0.468	0.479	0.509	0.460	0.420	**0.347**
40	MAR	0.540	0.552	0.588	0.533	0.496	**0.422**
50	MAR	0.602	0.615	0.654	0.597	0.564	**0.493**
60	MAR	0.661	0.676	0.718	0.658	0.630	**0.564**

Table 1.5: Execution time of Imputation Algorithms on the Cover Type Dataset with values 50% MAR in seconds.

Mean	Median	Freq.	PVI NN	RI	IARI
0.03	0.11	0.48	61.47	381.75	119.12

From our test results we can observe that the maximum average amount of MNAR values we can delete from each attribute is around the 12%. Which implies that approximately 88% of the dataset is filled with values below or equal the median of each attribute (probably 0). In Table 1.5 the execution time for each algorithm is shown for the case of 50% values MAR, which is representative for all the tests on this dataset. Our approach is not the fastest, Replace by Median, Replace by Mean and Replace by Most Frequent are almost instant while PVI, RI and IARI are more complex and take some time. The

execution time is mostly dependent on the size of the dataset and in particular the amount of attributes, and not so much on the amount of missing values.

Digits Dataset Results

In the tables below (Tables 1.6 to 1.8) the *accuracy* of the models created using the different imputed datasets as training set are shown.

Table 1.6: Model Accuracy Score on the Digits Dataset using Random Forests

Miss.%	Type	Ref.	Mean	Median	Freq.	PVI NN	RI	IARI
8	MNAR	0.972	0.971	**0.972**	0.966	0.972	0.970	0.972
16	MNAR	0.972	**0.969**	0.967	0.953	0.967	0.967	0.968
23	MNAR	0.972	**0.966**	0.964	0.937	0.962	0.963	0.962
25	MNAR	0.972	**0.967**	0.951	0.904	0.954	0.961	0.947
27	MNAR	0.972	**0.950**	0.923	0.815	0.934	0.932	0.944
10	MAR	0.972	0.969	**0.971**	0.967	0.970	0.971	0.971
20	MAR	0.972	0.964	0.963	0.961	0.967	0.968	**0.970**
30	MAR	0.972	0.962	0.961	0.956	0.963	0.963	**0.967**
40	MAR	0.972	0.954	0.957	0.952	0.959	0.960	**0.962**
50	MAR	0.972	0.950	0.952	0.945	0.953	0.957	**0.957**
60	MAR	0.972	0.944	0.943	0.934	0.944	0.948	**0.953**

Table 1.7: Model Accuracy Score on the Digits Dataset using Support Vector Machines

Miss.%	Type	Ref.	Mean	Median	Freq.	PVI NN	RI	IARI
8	MNAR	0.980	0.966	0.966	0.964	0.967	0.967	**0.969**
16	MNAR	0.980	0.955	0.952	0.946	0.958	0.958	**0.963**
23	MNAR	0.980	0.943	0.934	0.916	0.943	0.945	**0.955**
25	MNAR	0.980	0.914	0.897	0.841	0.917	0.920	**0.933**
27	MNAR	0.980	0.816	0.754	0.553	0.815	0.823	**0.891**
10	MAR	0.980	0.977	0.978	0.977	0.979	0.978	**0.980**
20	MAR	0.980	0.974	0.973	0.969	0.977	0.976	**0.979**
30	MAR	0.980	0.967	0.966	0.962	0.974	0.971	**0.979**
40	MAR	0.980	0.962	0.960	0.954	0.966	0.965	**0.976**
50	MAR	0.980	0.953	0.950	0.946	0.959	0.958	**0.974**
60	MAR	0.980	0.941	0.938	0.929	0.947	0.947	**0.967**

Table 1.8: Model Accuracy Score on the Digits Dataset using Gradient Boosting Trees

Miss.%	Type	Ref.	Mean	Median	Freq.	PVI NN	RI	IARI
8	MNAR	0.960	0.960	**0.960**	0.959	0.958	0.958	0.957
16	MNAR	0.960	**0.959**	0.954	0.945	0.956	0.954	0.956
23	MNAR	0.960	**0.956**	0.949	0.934	0.952	0.953	0.943
25	MNAR	0.960	0.949	0.936	0.892	0.940	**0.950**	0.918
27	MNAR	0.960	0.915	0.858	0.746	0.882	0.901	**0.924**
10	MAR	0.960	0.962	**0.962**	0.960	0.958	0.960	0.959
20	MAR	0.960	0.957	0.958	0.956	0.955	**0.960**	0.956
30	MAR	0.960	0.954	0.952	0.953	**0.958**	0.953	0.953
40	MAR	0.960	0.950	0.949	0.944	0.949	**0.952**	0.947
50	MAR	0.960	**0.944**	0.944	0.941	0.944	0.944	0.944
60	MAR	0.960	0.937	0.938	0.928	0.937	0.934	0.930

Table 1.9: Imputation Quality (RMSE) of each Imputation Algorithm on the Digits Dataset

Miss.%	Type	Mean	Median	Freq.	PVI NN	RI	IARI
8	MNAR	0.499	0.524	0.577	0.464	0.466	**0.419**
16	MNAR	0.608	0.649	0.752	0.566	0.565	**0.479**
23	MNAR	0.723	0.775	0.919	0.691	0.680	**0.534**
25	MNAR	0.858	0.903	1.037	0.841	0.829	**0.646**
27	MNAR	0.974	0.994	1.103	0.960	0.963	**0.850**
10	MAR	0.265	0.279	0.358	0.193	0.211	**0.154**
20	MAR	0.380	0.399	0.511	0.299	0.316	**0.231**
30	MAR	0.462	0.486	0.623	0.384	0.395	**0.289**
40	MAR	0.537	0.564	0.721	0.470	0.472	**0.345**
50	MAR	0.602	0.632	0.807	0.548	0.543	**0.400**
60	MAR	0.658	0.692	0.883	0.619	0.608	**0.451**

Table 1.10: Execution time of Imputation Algorithms on the Digits Dataset with values 50% MAR in seconds.

Mean	Median	Freq.	PVI NN	RI	IARI
0.00	0.01	0.02	9.21	30.39	14.25

In Table 1.9 the $RMSE$ values for every imputation algorithm are presented for all the combinations of missing data percentage and missing data type. Our IARI approach outperforms the other imputation algorithms in most of the MAR cases with respect to the accuracy. In the MNAR cases our algorithm works well but imputation by Mean sometimes has a slightly better accuracy for the Random Forest and Gradient Boosting final models. If we look at the $RMSE$ however, we see that our algorithm actually has a better $RMSE$ than the other algorithms even in the cases that another algorithm has a better

accuracy score using a Forest model. When we use Support Vector Machines to determine the accuracy scores, we see that our algorithm outperforms all the other algorithms in all cases. This would suggest that the SVM model might be a better option to use for this kind of problem than a Forest model. The higher accuracy score for the Support Vector Machine supports this assumption.

Houses 16H Dataset Results

In Tables 1.11 to 1.13 the R^2 scores are shown and in Table 1.15 the $RMSE$ results are shown for the imputation algorithms on the Houses 16H dataset.

Table 1.11: Model Accuracy Score (R^2) on the Houses Dataset using Random Forests

Miss.%	Type	Ref.	Mean	Median	Freq.	PVI NN	RI	IARI
10	MNAR	0.636	0.619	0.622	0.606	0.623	0.621	**0.630**
20	MNAR	0.636	0.604	0.598	0.580	0.606	0.603	**0.617**
30	MNAR	0.636	0.582	0.561	0.545	0.575	0.574	0.585
40	MNAR	0.636	0.534	0.491	0.485	0.511	0.520	0.531
49	MNAR	0.636	−0.277	−0.287	−0.545	−0.405	**−0.171**	−0.450
10	MAR	0.636	0.624	0.620	0.610	0.624	0.621	**0.627**
20	MAR	0.636	0.604	0.599	0.586	0.610	0.598	**0.620**
30	MAR	0.636	0.577	0.571	0.555	0.586	0.565	**0.608**
40	MAR	0.636	0.544	0.533	0.511	0.552	0.521	**0.590**
50	MAR	0.636	0.499	0.483	0.450	0.519	0.485	**0.567**
60	MAR	0.636	0.423	0.402	0.375	0.458	0.414	**0.536**

Table 1.12: Model Accuracy Score (R^2) on the Houses Dataset using Support Vector Machines

Miss.%	Type	Ref.	Mean	Median	Freq.	PVI NN	RI	IARI
10	MNAR	0.531	0.502	0.506	0.495	0.508	0.511	**0.522**
20	MNAR	0.531	0.472	0.477	0.469	0.478	0.485	**0.510**
30	MNAR	0.531	0.443	0.446	0.441	0.445	0.455	**0.493**
40	MNAR	0.531	0.403	0.404	0.403	0.402	0.414	**0.469**
49	MNAR	0.531	−0.342	−0.265	**−0.296**	−0.359	−0.341	−0.714
10	MAR	0.531	0.505	0.511	0.505	0.512	0.514	**0.526**
20	MAR	0.531	0.482	0.496	0.492	0.493	0.499	**0.522**
30	MAR	0.531	0.459	0.477	0.476	0.468	0.478	**0.517**
40	MAR	0.531	0.436	0.460	0.459	0.444	0.460	**0.511**
50	MAR	0.531	0.409	0.439	0.438	0.417	0.436	**0.506**
60	MAR	0.531	0.384	0.419	0.417	0.391	0.413	**0.495**

Table 1.13: Model Accuracy Score (R^2) on the Houses Dataset using Gradient Boosting Trees

Miss.%	Type	Ref.	Mean	Median	Freq.	PVI NN	RI	IARI
10	MNAR	0.582	0.574	0.577	0.563	0.575	0.574	0.577
20	MNAR	0.582	0.566	0.567	0.553	0.563	0.563	0.569
30	MNAR	0.582	0.553	0.553	0.541	0.547	0.550	0.551
40	MNAR	0.582	0.537	0.534	0.517	0.525	0.527	0.522
49	MNAR	0.582	−1.576	−1.161	−2.009	−1.060	−1.182	**−0.516**
10	MAR	0.582	0.568	0.574	0.565	0.574	0.572	**0.578**
20	MAR	0.582	0.559	0.562	0.551	0.561	0.559	**0.575**
30	MAR	0.582	0.547	0.548	0.536	0.550	0.544	**0.568**
40	MAR	0.582	0.530	0.531	0.515	0.532	0.525	**0.561**
50	MAR	0.582	0.510	0.507	0.490	0.512	0.500	**0.547**
60	MAR	0.582	0.486	0.483	0.464	0.490	0.471	**0.527**

Table 1.14: Execution time of Imputation Algorithms on the Houses Dataset with values 50% MAR in seconds.

Mean	Median	Freq.	PVI NN	RI	IARI
0.01	0.03	3.28	16.69	96.56	48.62

Table 1.15: Imputation Quality (RMSE) of each Imputation Algorithm on the Houses Dataset

Miss.%	Type	Mean	Median	Freq.	PVI NN	RI	IARI
10	MNAR	0.329	0.352	0.498	0.324	0.277	**0.228**
20	MNAR	0.486	0.517	0.709	0.485	0.428	**0.342**
30	MNAR	0.630	0.662	0.869	0.632	0.580	**0.452**
40	MNAR	0.785	0.801	1.007	0.787	0.753	**0.587**
49	MNAR	0.956	0.925	1.134	0.955	0.954	**0.927**
10	MAR	0.270	0.276	0.382	0.253	0.222	**0.190**
20	MAR	0.386	0.395	0.542	0.370	0.328	**0.280**
30	MAR	0.475	0.486	0.664	0.462	0.415	**0.352**
40	MAR	0.545	0.558	0.764	0.535	0.487	**0.412**
50	MAR	0.607	0.622	0.850	0.600	0.555	**0.466**
60	MAR	0.669	0.685	0.925	0.665	0.625	**0.531**

Page Blocks Dataset

In Tables 1.16 to 1.18 the accuracy scores are shown and in Table 1.20 the $RMSE$ results are shown for the imputation algorithms on the Page Blocks dataset.

Table 1.16: Model Accuracy Score on the Page Dataset using Random Forests

Miss.%	Type	Ref.	Mean	Median	Freq.	PVI NN	RI	IARI
10	MNAR	0.973	0.973	0.973	0.973	**0.974**	0.974	0.973
20	MNAR	0.973	0.972	0.973	0.972	**0.973**	0.972	0.972
30	MNAR	0.973	0.971	0.972	0.971	0.972	**0.974**	0.971
40	MNAR	0.973	0.971	0.970	0.970	0.971	**0.972**	0.969
49	MNAR	0.973	0.960	0.959	**0.960**	0.957	0.949	0.945
10	MAR	0.973	0.974	0.973	**0.974**	0.973	0.974	0.973
20	MAR	0.973	0.974	0.972	**0.974**	0.974	0.974	0.974
30	MAR	0.973	0.972	0.972	0.973	**0.973**	0.972	0.973
40	MAR	0.973	0.971	0.972	0.971	0.973	0.971	**0.973**
50	MAR	0.973	0.972	0.971	0.969	0.970	0.968	**0.972**
60	MAR	0.973	0.969	0.968	0.968	0.969	0.966	**0.970**

Table 1.17: Model Accuracy Score on the Page Dataset using Support Vector Machines

Miss.%	Type	Ref.	Mean	Median	Freq.	PVI NN	RI	IARI
10	MNAR	0.960	0.959	0.958	0.953	0.959	0.959	**0.960**
20	MNAR	0.960	0.958	0.957	0.952	0.958	0.959	**0.960**
30	MNAR	0.960	0.956	0.952	0.951	0.955	0.957	**0.959**
40	MNAR	0.960	0.945	0.943	0.944	0.945	0.949	**0.955**
49	MNAR	0.960	0.888	0.889	**0.895**	0.888	0.888	0.892
10	MAR	0.960	0.959	0.959	0.954	0.959	0.960	**0.960**
20	MAR	0.960	0.957	0.958	0.952	0.959	0.960	**0.960**
30	MAR	0.960	0.955	0.956	0.950	0.956	0.960	**0.960**
40	MAR	0.960	0.952	0.954	0.946	0.953	0.958	**0.961**
50	MAR	0.960	0.950	0.951	0.942	0.951	0.957	**0.959**
60	MAR	0.960	0.946	0.945	0.937	0.946	0.954	**0.960**

Table 1.18: Model Accuracy Score on the Page Dataset using Gradient Boosting Trees

Miss.%	Type	Ref.	Mean	Median	Freq.	PVI NN	RI	IARI
10	MNAR	0.971	0.971	0.972	0.970	0.971	**0.972**	0.971
20	MNAR	0.971	0.970	0.970	0.971	0.972	0.971	**0.972**
30	MNAR	0.971	0.970	0.969	0.969	0.971	**0.971**	0.970
40	MNAR	0.971	0.967	0.966	0.965	0.966	**0.970**	0.969
49	MNAR	0.971	0.943	**0.948**	0.942	0.939	0.943	0.930
10	MAR	0.971	**0.973**	0.972	0.971	0.972	0.972	0.971
20	MAR	0.971	0.972	**0.973**	0.970	0.971	0.972	0.971
30	MAR	0.971	0.971	0.971	0.970	**0.972**	0.970	0.971
40	MAR	0.971	0.968	0.968	0.967	0.969	0.968	**0.972**
50	MAR	0.971	0.966	0.964	0.964	0.967	0.967	**0.969**
60	MAR	0.971	0.961	0.959	0.962	0.962	0.965	**0.967**

Table 1.19: Execution time of Imputation Algorithms on the Page Dataset with values 50% MAR in seconds.

Mean	Median	Freq.	PVI NN	RI	IARI
0.00	0.00	0.10	1.97	35.32	19.89

Table 1.20: Imputation Quality (RMSE) of each Imputation Algorithm on the Page Dataset

Miss.%	Type	Mean	Median	Freq.	PVI NN	RI	IARI
10	MNAR	0.384	0.407	0.473	0.350	0.307	**0.220**
20	MNAR	0.543	0.574	0.649	0.517	0.455	**0.282**
30	MNAR	0.682	0.714	0.710	0.668	0.605	**0.389**
40	MNAR	0.816	0.838	0.819	0.813	0.771	**0.496**
49	MNAR	0.978	0.965	**0.939**	0.977	0.976	0.946
10	MAR	0.276	0.284	0.429	0.240	0.207	**0.126**
20	MAR	0.403	0.413	0.610	0.367	0.323	**0.230**
30	MAR	0.482	0.495	0.739	0.452	0.399	**0.287**
40	MAR	0.563	0.578	0.860	0.537	0.481	**0.317**
50	MAR	0.634	0.651	0.964	0.616	0.555	**0.367**
60	MAR	0.688	0.707	1.051	0.673	0.617	**0.417**

Concrete Dataset

In Tables 1.21 to 1.23 the R^2 scores are shown and in Table 1.25 the $RMSE$ results are shown for the imputation algorithms on the Concrete dataset.

Table 1.21: Model Accuracy (R^2) Score on the Concrete Dataset using Random Forests

Miss.%	Type	Ref.	Mean	Median	Freq.	PVI NN	RI	IARI
10	MNAR	0.906	**0.897**	0.886	0.880	0.892	0.893	0.891
20	MNAR	0.906	**0.883**	0.867	0.860	0.873	0.875	0.877
29	MNAR	0.906	**0.841**	0.839	0.839	0.824	0.823	0.825
39	MNAR	0.906	**0.821**	0.809	0.816	0.801	0.796	0.762
47	MNAR	0.906	0.727	0.720	0.726	0.712	0.701	0.642
10	MAR	0.906	0.884	0.879	0.877	0.885	0.880	**0.892**
20	MAR	0.906	0.866	0.859	0.852	0.866	0.855	**0.877**
30	MAR	0.906	0.849	0.842	0.833	0.849	0.828	**0.862**
40	MAR	0.906	0.824	0.821	0.812	0.830	0.795	**0.844**
50	MAR	0.906	0.791	0.787	0.775	0.799	0.747	**0.819**
60	MAR	0.906	0.759	0.760	0.752	0.762	0.694	**0.792**

Table 1.22: Model Accuracy (R^2) Score on the Concrete Dataset using Support Vector Machines

Miss.%	Type	Ref.	Mean	Median	Freq.	PVI NN	RI	IARI
10	MNAR	0.637	0.610	0.597	0.596	0.608	0.622	**0.630**
20	MNAR	0.637	0.564	0.549	0.542	0.556	0.583	**0.609**
29	MNAR	0.637	0.482	0.478	0.472	0.478	0.498	**0.541**
39	MNAR	0.637	0.409	0.398	0.391	0.406	0.433	**0.529**
48	MNAR	0.637	0.240	0.257	0.269	0.237	0.244	**0.318**
10	MAR	0.637	0.619	0.618	0.606	0.622	0.628	**0.634**
20	MAR	0.637	0.597	0.596	0.574	0.603	0.615	**0.631**
30	MAR	0.637	0.574	0.572	0.535	0.578	0.598	**0.628**
40	MAR	0.637	0.552	0.549	0.502	0.555	0.584	**0.625**
50	MAR	0.637	0.526	0.519	0.453	0.528	0.563	**0.620**
60	MAR	0.637	0.490	0.484	0.412	0.493	0.535	**0.610**

Table 1.23: Model Accuracy (R^2) Score on the Concrete Dataset using Gradient Boosting Trees

Miss.%	Type	Ref.	Mean	Median	Freq.	PVI NN	RI	IARI
10	MNAR	0.899	0.895	0.889	0.889	0.891	0.892	0.895
20	MNAR	0.899	**0.890**	0.872	0.869	0.881	0.880	0.883
29	MNAR	0.899	0.829	0.842	**0.844**	0.822	0.816	0.833
39	MNAR	0.899	0.820	0.824	**0.827**	0.810	0.803	0.794
47	MNAR	0.899	0.701	0.719	**0.727**	0.694	0.689	0.637
10	MAR	0.899	0.889	0.889	0.881	0.887	0.883	**0.893**
20	MAR	0.899	0.873	0.871	0.864	0.870	0.865	**0.882**
30	MAR	0.899	0.851	0.856	0.844	0.854	0.845	**0.870**
40	MAR	0.899	0.834	0.835	0.827	0.832	0.822	**0.853**
50	MAR	0.899	0.802	0.807	0.791	0.808	0.793	**0.831**
60	MAR	0.899	0.765	0.767	0.759	0.767	0.755	**0.808**

Table 1.24: Execution time of Imputation Algorithms on the Concrete Dataset with values 50% MAR in seconds.

Mean	Median	Freq.	PVI NN	RI	IARI
0.00	0.00	0.01	0.15	2.79	1.99

Table 1.25: Imputation Quality (RMSE) of each Imputation Algorithm on the Concrete Dataset

Miss.%	Type	Mean	Median	Freq.	PVI NN	RI	IARI
10	MNAR	0.343	0.400	0.469	0.331	0.271	**0.178**
20	MNAR	0.516	0.591	0.667	0.514	0.440	**0.292**
29	MNAR	0.689	0.775	0.831	0.691	0.628	**0.447**
39	MNAR	0.860	0.905	0.963	0.862	0.818	**0.589**
47	MNAR	1.058	1.036	1.092	1.058	1.055	**1.015**
10	MAR	0.277	0.295	0.378	0.240	0.209	**0.140**
20	MAR	0.389	0.415	0.527	0.360	0.311	**0.213**
30	MAR	0.477	0.509	0.653	0.456	0.399	**0.280**
40	MAR	0.549	0.585	0.746	0.534	0.476	**0.342**
50	MAR	0.613	0.653	0.843	0.603	0.547	**0.410**
60	MAR	0.674	0.717	0.909	0.668	0.619	**0.476**

1.1 Attribute Selection and Sorting Methods

For our algorithm we select the most important attribute to be imputed first by using the out-of-bag samples provided by our random forest algorithm. The question is if this is the best possible order of imputation. We compared the scores of our algorithm using this attribute ordering with our algorithm using the exact opposite attribute ordering (least important attribute first). Using this least importance first ordering the results are only slightly worse than the results from our original algorithm. This implies that the order of attribute repair does matter, but is not a key factor in our algorithm.

1.1.1 Greedy Model Accuracy Selection

Another sorting / attribute selection method would be to try to repair each attribute first using the already repaired attributes and select the attribute that improves the accuracy of the model the most. This selected attribute is then added to the training set and the procedure is repeated till all remaining attributes do not improve the accuracy any more. This greedy approach leads to an algorithm where we can decide to stop imputing the attributes once the model does not improve anymore. Effectively giving us an algorithm that not only repairs the attributes of our dataset but also selects which attributes are useful to impute and which attributes are not improving our model and so might be better left out.

Running this greedy version of our algorithm on the Digits dataset we get the R^2 results of Table 1.26.

Our greedy algorithm uses on average only 22 of the 64 attributes that are available. It is interesting to see that the results of our greedy approach are actually worse than the results of our non-greedy approach. The reason of the worse results is most probably that too many attributes get ignored (discarded) in the given dataset. It can be the case that adding an attribute to our model might not improve the models accuracy, but adding this attribute in combination with another attribute might still improve our model. We can see this clearly

Table 1.26: Model Accuracy Score for a greedy IARI approach on the Digits dataset

Miss.%	Type	Ref.	IARI	GREEDY
8	MNAR	0.964	**0.961**	0.954
16	MNAR	0.964	**0.953**	0.949
23	MNAR	0.964	**0.950**	0.912
25	MNAR	0.964	**0.926**	0.907
27	MNAR	0.964	**0.915**	0.908
10	MAR	0.964	0.956	0.952
20	MAR	0.964	0.951	0.951
30	MAR	0.964	**0.955**	0.943
40	MAR	0.964	**0.952**	0.941
50	MAR	0.964	**0.948**	0.935
60	MAR	0.964	**0.934**	0.921

Table 1.27: Model Accuracy Score (R^2) for a greedy IARI approach on the Houses Dataset

Miss.%	Type	Ref.	IARI	GREEDY
8	MNAR	0.636	0.630	**0.632**
8	MNAR	0.636	0.617	**0.624**
8	MNAR	0.636	0.585	**0.605**
8	MNAR	0.636	0.531	**0.574**
8	MNAR	0.636	−0.450	**0.016**
10	MAR	0.636	0.627	**0.635**
20	MAR	0.636	0.620	**0.631**
30	MAR	0.636	0.608	**0.624**
40	MAR	0.636	0.590	**0.614**
50	MAR	0.636	0.567	**0.599**
60	MAR	0.636	0.536	**0.580**

in the following example where we try to model $y = OR(x_1, XOR(x_2, x_3))$. If we first model our function, using only x_1, we will get an accuracy of 75% (if x_1 equals zero we either always say true or false and are wrong in half of the cases). When we now add x_2 in our training set (after fixing possible missing values) we will see that the score of the model does not improve. This is due to the fact that we do not gain any additional information by x_2 alone. However, if we also add x_3 to our training set we will suddenly improve our model to 100% accuracy (under the assumption that we have sufficient samples).

This simple example tells us that combinations of attributes might be valuable even if all the single attributes are not giving any improvement on the final model. In the case of the Digits dataset, this might well be the case since each attribute only stands for a single pixel, and conbinations of pixels create the information we need to see what digit the image represents, while single digits might not give us any information at all.

We testes our greedy algorithm also on the Houses 16H dataset, which seems

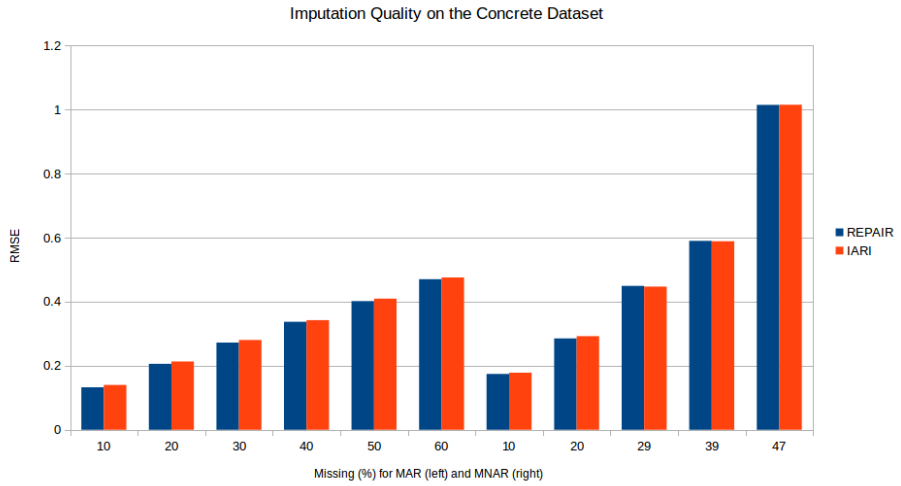

Figure 1.1: Imputation Quality ($RMSE$) on the Concrete Dataset

more suitable for such an approach. The dataset has 16 attributes and the greedy algorithm uses on average 13 of them. The R^2 scores of the original IARI and the GREEDY modification can be seen in Table 1.27. In this case the greedy algorithm performs better than the original IARI algorithm. It clearly depends on the dataset and the dependencies between the attributes if the greedy attribute selection works or not. In practice, the greedy IARI approach, tries to solve the feature selection problem while repairing the data. The focus of this Greedy algorithm is to improve the quality of our final model, which is the final goal, but we now try to do both imputation and feature selection at the same time and our imputation is not better than it was before.

1.1.2 Greedy Imputation Quality Selection

Another possibility to optimize our IARI algorithm is to select the features to be repaired on the *repairability* of each attribute. With repairability we mean how well a specific attribute can be repaired. This can be measured by the out-of-bag error provided by the Random Forest models we use to repair each attribute. The main idea behind this approach is that if we repair the attributes that can be repaired as good as possible first, we might be able to repair the remaining attributes even better and so we optimize our algorithm in minimizing the $RMSE$. In Figure 1.1 the $RMSE$ of the IARI algorithm and the repairability selection version of the IARI algorithm (named REPAIR) are shown. We see that indeed our algorithm has a tiny improvement in RMSE score, but the modified algorithm also takes more time due to the $n * (n-1)$ models it need to create.

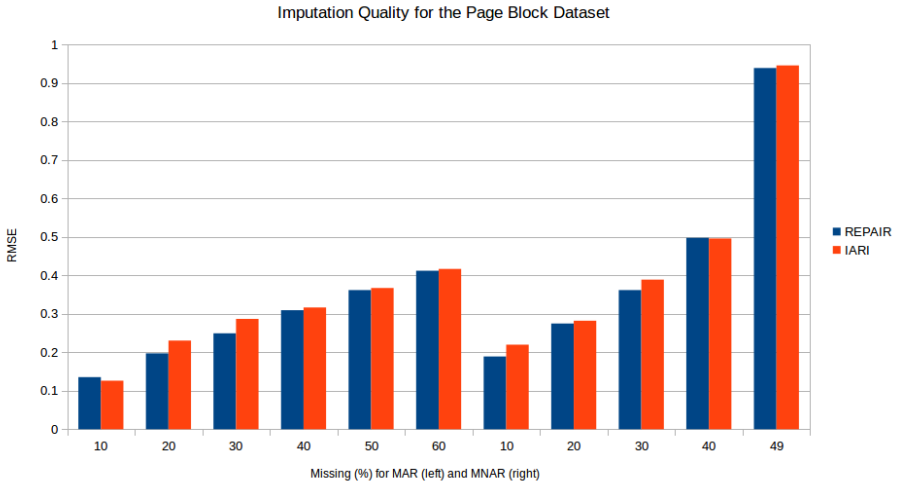

Figure 1.2: Imputation Quality ($RMSE$) on the Page Dataset

Bibliography

[1] PASCAL Machine Learning Benchmarks Repository - mldata.org. http://mldata.org/repository/data.

[2] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[3] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.