Case Study V

James Vasquez, Javier Saldana, Sara Zaheri

Table of Contents

Table of Figures

List of Tables

1. Introduction

Albeit usually seen as nuisance more than a problem, the hidden cost of data may be a price too high to pay. Missing data has the potential to reduce robust datasets down to a sample size too small for the model to utilize for considerable validation. In situations where the sample size is still large enough for modeling, missing data may provide insignificant results based on the stripped results. In addition, it could generate misleading results that may then cause significant harm in the business. Other unintended consequence of missing data could be misrepresentation of variables, bias in parameters, and reduction in statistical power.

Rubin separates missing data into three groups: missing completely at random, missing at random, and missing not at random [1]. Missing completely at random is defined as missing data that has no relation to the value of the variable or the instance of record. An example of data missing completely at random could be the unanswered responses in 10-question surveys that have been submitted. Missing at random data is data that it is missing based on a set of observations. Keeping in line with our 10-question survey example, missing at random would be the missing data of Question #5 based on the way the question was presented. Finally, there is data missing not at random, which is data that is intentionally removed/missing. In our survey example, missing not at random would be the data if we decided to remove Question #9 entirely from observations. This is by far the most problematic as it presents a strong opportunity for bias in the data.

There are several techniques commonly utilized to handle missing data. For our study, we will be utilizing multiple imputation as our preferred method. In this method, the missing values are substituted by plausible values and utilized to then train and test the model. For our study, we selected the Boston Housing Price dataset, which is publicly available. We create a baseline linear regression model without any missing data and proceed by expanding our imputation technique to account from 1% up to 50% of the data. We hypothesize that our models will perform worse than our baseline model since we are replacing the real data with generated data.

2. Methods

2.1. Data Source

In this case study we use Boston housing price dataset which is automatically readable from sklearn by calling:

```
sklearn.datasets.load_boston()
```

It is a dataset containing 506 records having 12 features and one target which is the median price. The target is between 5 and 50. These features are related to some geographical, social, and economical factors together with some home features such as the number of rooms or age. We will

see that the number of rooms per dwelling has the maximum regression coefficient when predicting the price. This dataset was provided by StatLab library. A detailed description of the data variables are provided below:

Table 1. Description of columns in Boston Housing Price dataset

	Description
CRIM	per capita crime rate by town
ZN	proportion of residential land zoned for lots over 25,000 sq.ft.
INDUS	proportion of non-retail business acres per town
CHAS	Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
NOX	nitric oxides concentration (parts per 10 million)
RM	average number of rooms per dwelling
AGE	proportion of owner-occupied units built prior to 1940
DIS	weighted distances to five Boston employment centres
RAD	index of accessibility to radial highways
TAX	full-value property-tax rate per $10,000
PTRATIO	pupil-teacher ratio by town
B	$1000(Bk - 0.63)^2$ where Bk is the proportion of blacks by town
LSTAT	% lower status of the population
MEDV	Median value of owner-occupied homes in $1000's

2.2. Exploratory Data Analysis (EDA)

First we present a quick description of each feature:

Table 2. A brief description of Boston housing data

	CRIM	**ZN**	**INDUS**	**CHAS**	**NOX**	**RM**	**AGE**	**DIS**	**RAD**	**TAX**	**PTRATIO**	**B**	**LSTAT**	**MEDVAL**

count	506.0	506.0	506.0	506.0	506.0	506.0	506.0	506.0	506.0	506.0	506.0	506.0	506.0	506.0
mean	3.61	11.36	11.14	0.07	0.55	6.28	68.57	3.8	9.55	408.24	18.46	356.67	12.65	22.53
std	8.6	23.32	6.86	0.25	0.12	0.7	28.15	2.11	8.71	168.54	2.16	91.29	7.14	9.2
min	0.01	0.0	0.46	0.0	0.38	3.56	2.9	1.13	1.0	187.0	12.6	0.32	1.73	5.0
25%	0.08	0.0	5.19	0.0	0.45	5.89	45.02	2.1	4.0	279.0	17.4	375.38	6.95	17.02
50%	0.26	0.0	9.69	0.0	0.54	6.21	77.5	3.21	5.0	330.0	19.05	391.44	11.36	21.2
75%	3.68	12.5	18.1	0.0	0.62	6.62	94.07	5.19	24.0	666.0	20.2	396.22	16.96	25.0
max	88.98	100.0	27.74	1.0	0.87	8.78	100.0	12.13	24.0	711.0	22.0	396.9	37.97	50.0

The target is MEDVAL. We plot its distribution to understand better how it varies from 5 to 50.

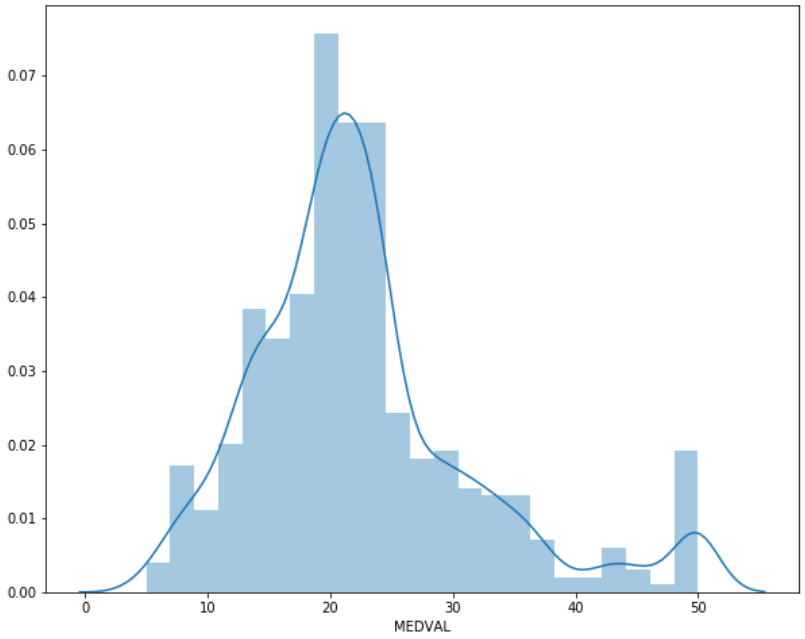

Figure 1.MEDVAL distribution

Another helpful tool to realize the importance of each feature is the correlation matrix. We use Seaborn heatmap to illustrate this matrix.

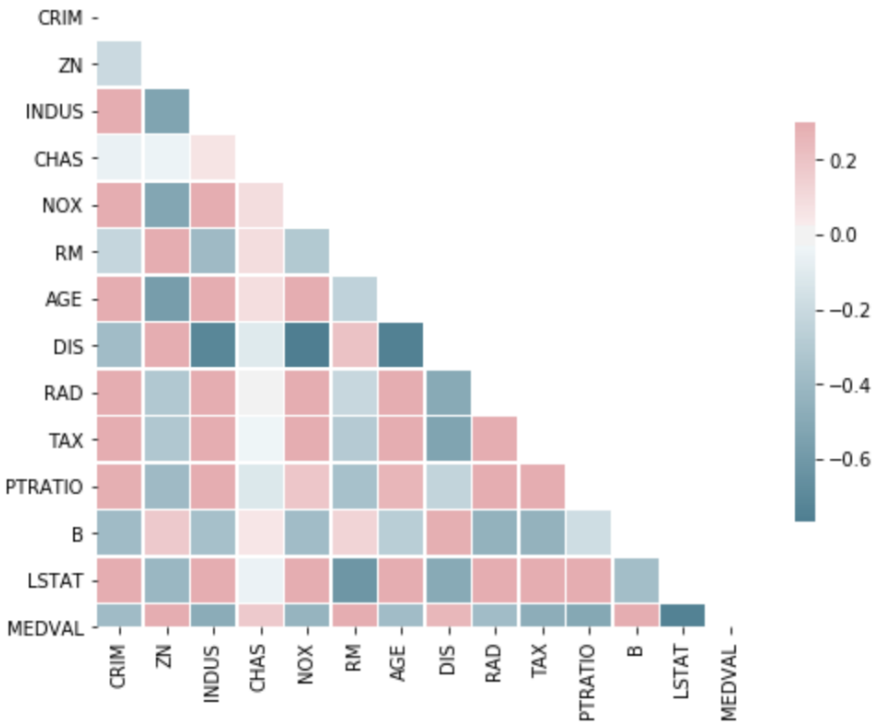

Figure 2. Correlation matrix

Now we apply regression to find the most promising features according to their regression coefficients. The sorted regression coefficients are as below:

[('RM', 3.81), ('CHAS', 2.687), ('RAD', 0.306), ('ZN', 0.046), ('INDUS', 0.021), ('B', 0.009), ('AGE', 0.001), ('TAX', -0.012), ('CRIM', -0.108), ('LSTAT', -0.525), ('PTRATIO', -0.953), ('DIS', -1.476), ('NOX', -17.767)]

As a result the most important positive feature is 'RM' and the most important negative feature is 'NOX'. The intercept is also:36.459. The mean square error when fitting the model is 21.895. Thus if we use the regression as our baseline model, the loss of mean square error is the mentioned value and The R2 score is 0.74.

2.3. Analysis

Analysis of the data set was done at random meaning that the code produced randomly selects columns that are used for creating NULLs in the data set. This was done to provide a random selection of how the data is interpreted.

The scenarios are as followed:

- Create a baseline with no missing data
- Select a features and NULL out 1%, 5%, 10%, 20%, 33% & 50% of the data
- Use a conditional statement to NULL out 10%, 20%, & 30% of the data for 2 features
- Create a pattern to NULL out 25% of a selected feature

To assess the data a linear regression model was built for each scenario, which totaled ten separate models. Metrics that are captured include the Mean Absolute Error (MAE), Mean Square Error (MSE), Root Mean Square Error (RMSE) and R^2.

- MAE
    - Calculate the residuals for every data point and use the absolute value of each point
- MSE
    - Calculate the residuals for every data point and squares difference between actual and predicted
- RMSE
    - Square Root of the MSE
- R^2
    - Goodness of fit measure

MAE is a general metric but due to using the absolute value we are unable to determine underperformance or overperformance. MSE will be larger than the MAE and cannot be used to compare against, however the MSE can be used to compare competing models. Total error in the MSE can be driven by outliers. RMSE provides an absolute measure of fit, which is the standard deviation of the variance in the model. R2 is a relative measure of fit, which identifies how well the data fits the model. We will use RMSE for analysis.

Following the metrics from each model the baseline model results were subtracted to find the difference.

In order to ensure each model with each scenario provided the same NULLs and incremented to each percentage that was assign to be NULL out, a function in NumPy was used ‘ediff1d’[2] this allowed the same NULLs to be NULL when incrementing to the next percentage of NULLs to be used. The same imputation method of using the median value was used for each model.

3. Results

Results are from run 1:

Table 3. Results of run 1, 10 models, row 0 is baseline

	frac	mae	mse	rmse	r2	mae_diff	mse_diff	rmse_diff	r2_diff	col_for_NULLs	question
0	0.00	3.347030	24.609093	4.960755	0.720811	NaN	NaN	NaN	NaN	PTRATIO	1
1	0.01	3.471674	26.469148	5.144818	0.728160	-0.124644	-1.860055	-0.184062	-0.007349	PTRATIO	2
2	0.05	3.180363	20.507386	4.528508	0.721483	0.166667	4.101706	0.432247	-0.000671	PTRATIO	2
3	0.10	3.392329	20.045118	4.477177	0.711063	-0.045299	4.563975	0.483578	0.009748	PTRATIO	2
4	0.20	3.206405	19.526042	4.418828	0.753610	0.140625	5.083050	0.541927	-0.032798	PTRATIO	2
5	0.33	3.499708	23.180748	4.814639	0.721299	-0.152678	1.428345	0.146116	-0.000487	PTRATIO	2
6	0.50	3.258062	21.836552	4.672960	0.741213	0.088968	2.772541	0.287796	-0.020402	PTRATIO	2
7	0.10	3.466904	21.615498	4.649247	0.727559	-0.119874	2.993595	0.311508	-0.006748	LSTAT, INDUS	3
8	0.20	3.176836	18.402038	4.289760	0.768076	0.170194	6.207055	0.670996	-0.047264	LSTAT, INDUS	3
9	0.30	3.664395	31.111902	5.577804	0.688859	-0.317366	-6.502810	-0.617049	0.031953	LSTAT, INDUS	3
10	0.25	3.263214	24.808219	4.980785	0.688303	0.083816	-0.199126	-0.020030	0.032509	PTRATIO	4

When plotting the RMSE run 1 against all other scenarios:

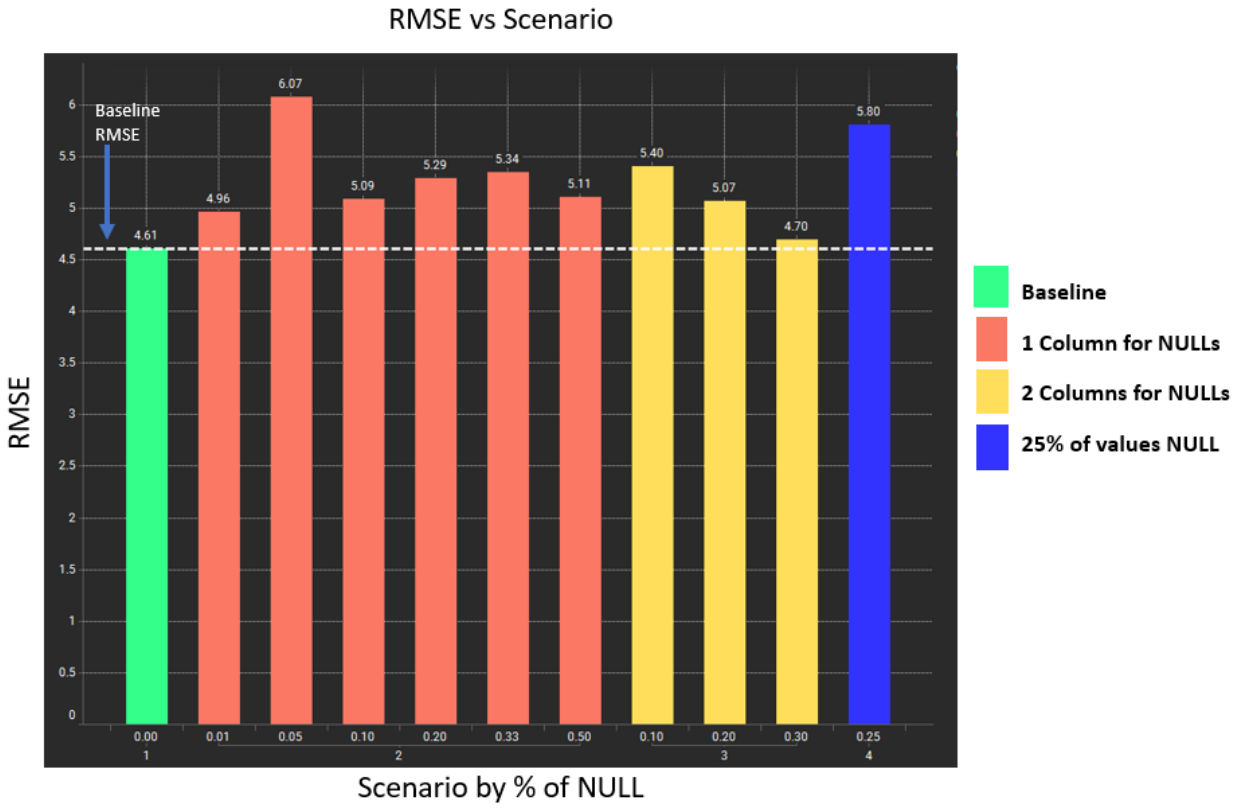

Figure 3. RMSE Plots run 1

Results from run 2:

Table 4. Results of run 2, 10 models, row 0 is baseline

	frac	mae	mse	rmse	r2	mae_diff	mse_diff	rmse_diff	r2_diff	col_for_NULLs	question
0	0.00	3.432729	23.205297	4.817188	0.737367	NaN	NaN	NaN	NaN	RM	1
1	0.01	3.455842	25.693831	5.068908	0.727701	-0.023112	-2.488534	-0.251721	0.009666	RM	2
2	0.05	3.611163	33.850334	5.818104	0.544227	-0.178434	-10.645037	-1.000916	0.193140	RM	2
3	0.10	3.731713	29.151386	5.399202	0.682847	-0.298984	-5.946089	-0.582015	0.054521	RM	2
4	0.20	3.344940	21.778527	4.666747	0.746946	0.087789	1.426770	0.150441	-0.009579	RM	2
5	0.33	3.684412	28.228408	5.313041	0.726803	-0.251682	-5.023111	-0.495854	0.010564	RM	2
6	0.50	3.619973	28.128313	5.303613	0.690898	-0.187243	-4.923017	-0.486426	0.046470	RM	2
7	0.10	3.651740	27.589057	5.252529	0.674472	-0.219011	-4.383760	-0.435341	0.062896	RAD, TAX	3
8	0.20	3.311663	23.695397	4.867792	0.715141	0.121066	-0.490100	-0.050604	0.022226	RAD, TAX	3
9	0.30	3.424800	22.804380	4.775393	0.730828	0.007930	0.400917	0.041794	0.006539	RAD, TAX	3
10	0.25	3.400514	26.374798	5.135640	0.687403	0.032215	-3.169501	-0.318452	0.049964	RM	4

When plotting the RMSE run 2 against all other scenarios:

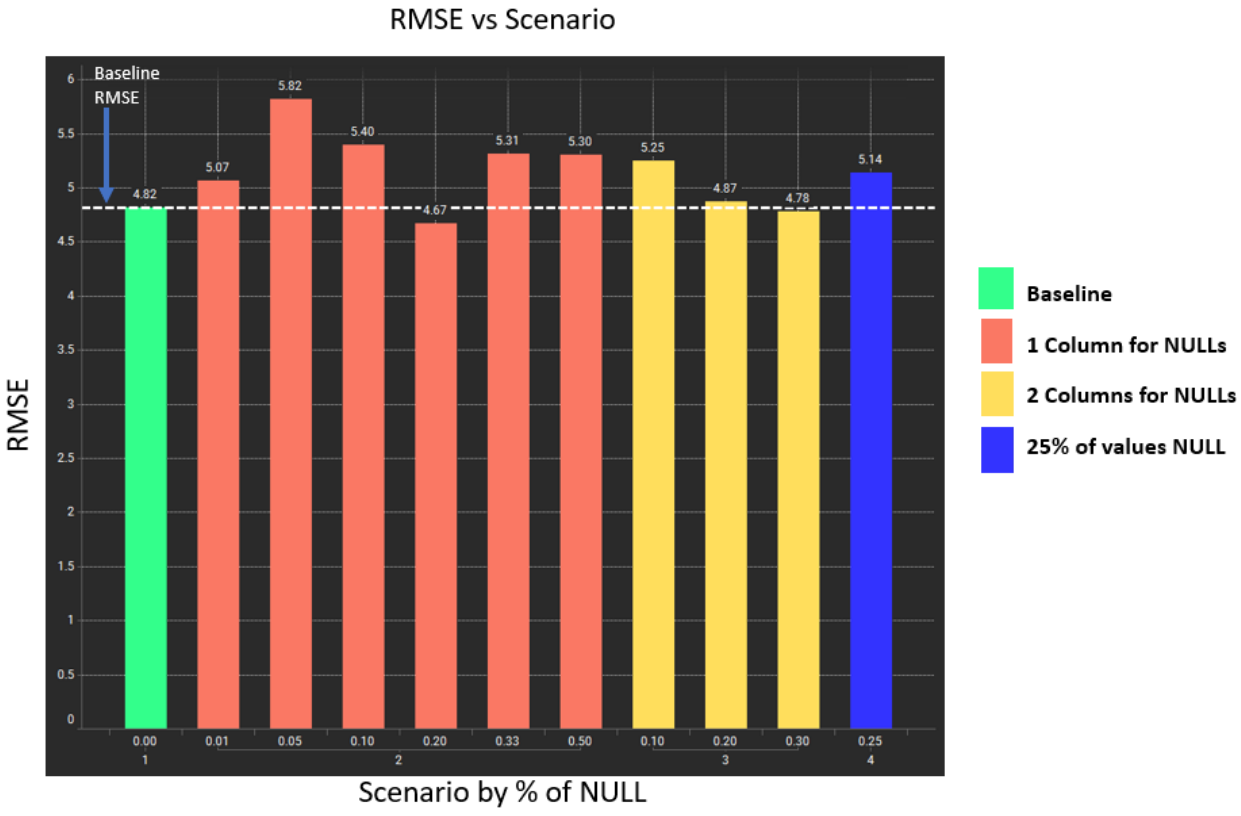

Figure 4. RMSE Plots run 2

What we find when different features provide NULLs are that the RMSE used to determine the quality of the model changes in each different run. In run 1 no other scenario beat the baseline RMSE. In run 2 we find that several scenarios were able to beat the baseline RMSE.

- Scenario 2 with 20% NULL of a single feature
- Scenario 3 where 2 features at 30% were NULL

4. Conclusion

Based on the results, we see the baseline outperform all other models. However, in the second attempt two other models perform slightly better than the baseline model. The model from scenario two with 20% of a single feature being null performed considerably worse in the first run than the second run. However, the model from scenario 3 where two features have 30% of null data, we see the model performs slightly better or worse than the baseline. This finding appears to go against our initial hypothesis in that higher nulls value impact the performance of the model. However, there is a cause for caution moving forward considering 30% of the data is missing.

What the model is then doing is replace those nulls with the median values of the column. These median values are reducing the spread of the variable and helps bring the response to the center. In this scenario, the model is likely to generate values closer to the median which there is a greater concentration of values. The issue rises that when the model is then tested using new data. The new data may not reflect the same spread being imputed into the model and as a result the model may perform worse in deployment. This bring us to a critical point mentioned initially with regards to the costly side effects of missing data. Models may be trained on the data sets with missing values and those results may be inaccurate or misleading.

The best way to circumvent this issue would be to attempt to minimize the missing data as much as possible. In the event that is not possible, then extensive testing and training would be needed in an effort to truly see the performance of the models. Missing data is an issue that extends well beyond the walls of data science and this case is a prime example of that. With these results, we may be tempted to present an erroneous model to the end-user if we were to base our measure solely on the performance metrics and not understand why its performing better than the baseline.

5. References

[1] Rubin DB. Inference and missind data. Biometrika. 1976, 63:581–592.

[2] numpy.ediff1d — NumPy v1.19 Manual. (2020). Retrieved 10 July 2020, from https://numpy.org/doc/stable/reference/generated/numpy.ediff1d.html

6. Appendix (Code)

The code is implemented in Jupyter notebook. We leave this part blank and submit the .ipython file together with the project.