Experiment

Creator **Olusanmi Hundogan** Created **Jan 12, 2024, 01:37** Last updated **Jan 17, 2024, 14:26**

Files

- experiments_missing_value_imputers.py
- analyze_missing_value_imputers.py

Motivation

Missing value imputation is a crucial component of the overall model as the data is highly sparse. For many of the methods used, we need to handle 'nan' values in the dataset. As we assume that a better imputation method will lead to a better prediction result, we want to investigate a multitude of imputation methods for the dataset. These methods are:

- BaselineImputer: Imputes values based on a simple heuristic
- DummyImputer: Imputes values with random numbers
- MVEImputers: Column-wise RegressionImputers
  - KNN
  - Ridge
  - Decision Tree
  - LGBM(learning rate=0.1, num estimators = 50)
- OldOxariImputer: Same as MVEImputater but with RandomForest as regressor
- K-Bucket Imputers: Imputations based on an instance based K algorithm
  - KNN (K=9)
  - Kmedian (K=13)
  - KMeans (K=7)
- Categorical Statistics Imputer: Computes imputation values based on categories
  - Country
  - Sector
  - Industry
- Numerical Statistics Imputer: Similar to categorical but based on discretized numerical values
  - TotalAssets(num_buckets=11)
  - TotalLiabilities(num_buckets=11)
  - Revenue(num_buckets=11)

Design

We evaluate the imputer by removing values from a dataframe gradually and measuring the sMAPEof reconstructing the values. We test difficulties from 0.1 to 0.9 in even steps. A difficulty of 0.1 corresponds to the artificial removal of 10% of known values. (The number is not completely accurate because the 10% applies to the dataframe, but the dataframe already contains missing values.)

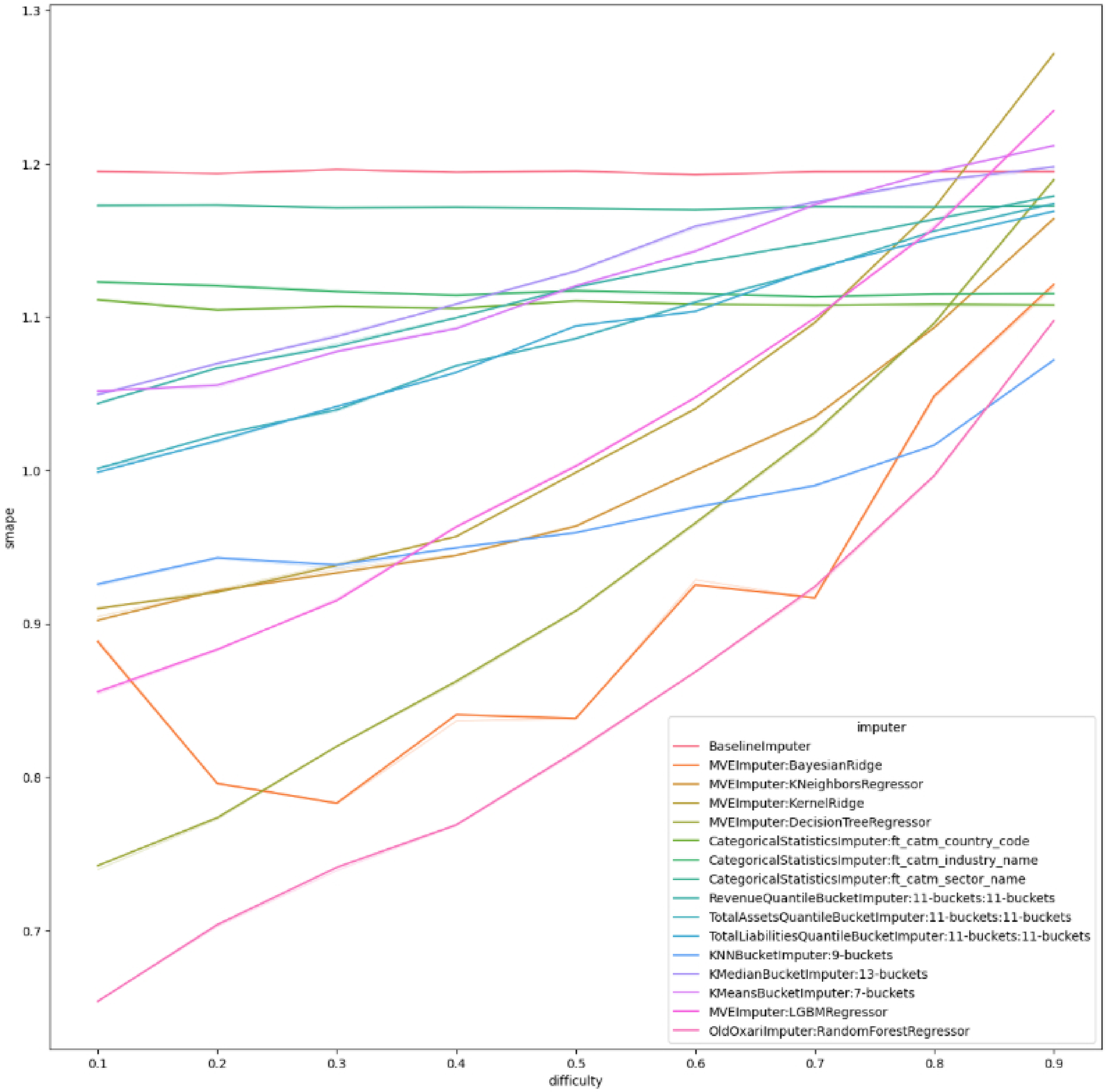
Furthermore, we only impute on 3 modes of feature sets:

- realistic: contains all features
- mid_missingness Only features that have a rate of missingness below 50% across the data set.
- low_missingness Only features that have a rate of missingness below 30% across the data set.

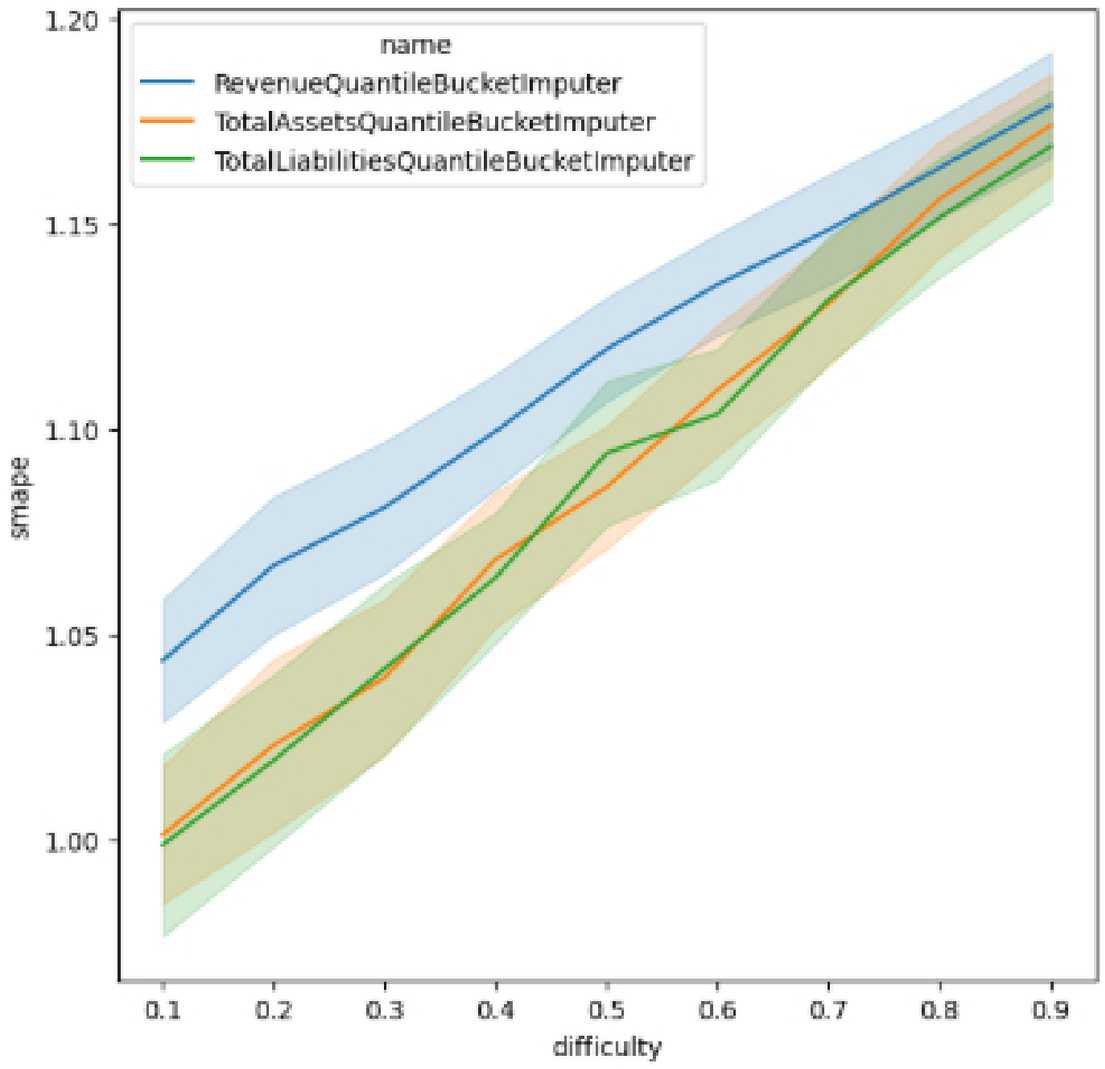
Note: Within the low_missingnessmode the data rows have a missingness of below 10%. (It's important to understand that the missingness that a feature exudes is different from the missingness per row. A feature might be entirely missing but the rows have on average 95% of there data filled)

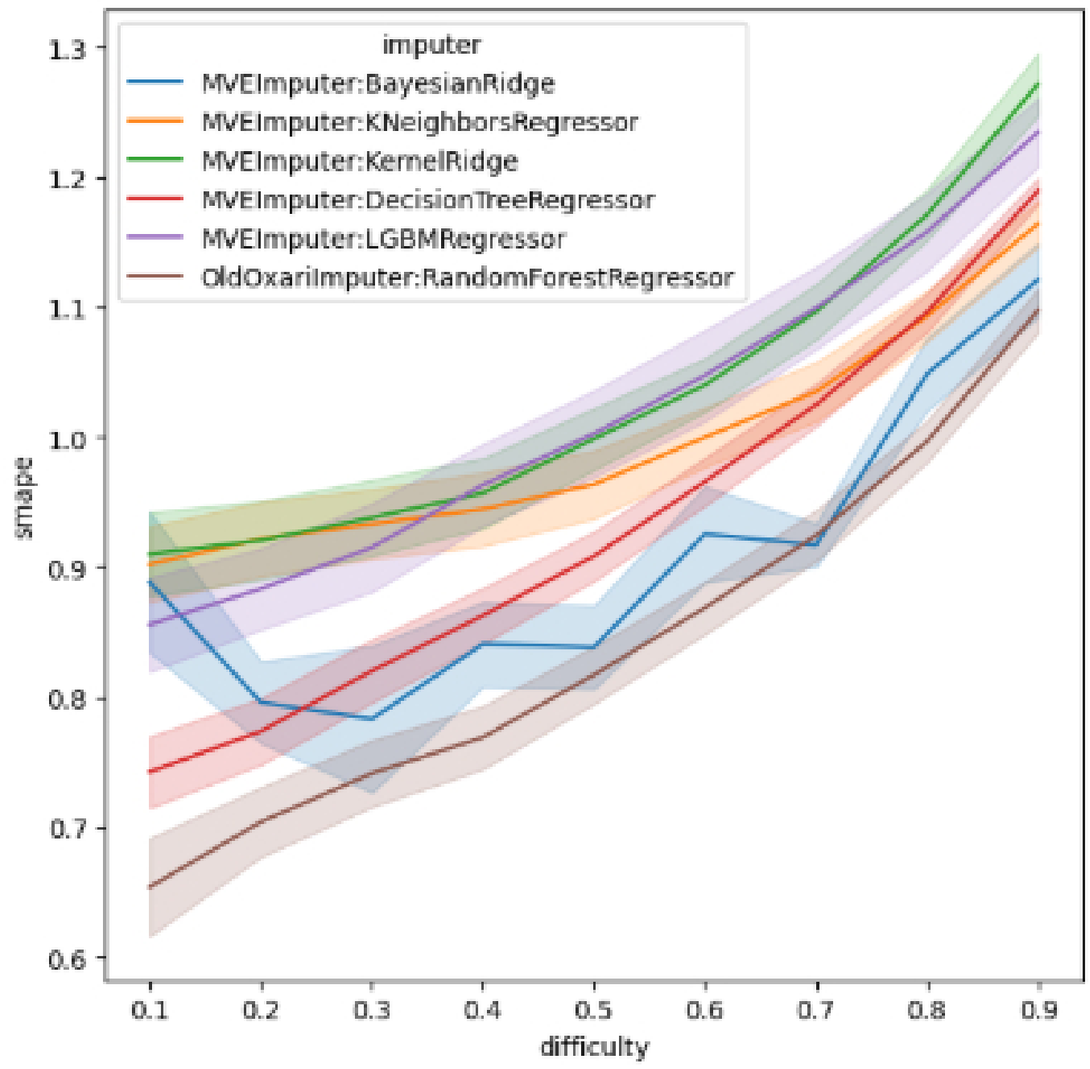
Result and Insights

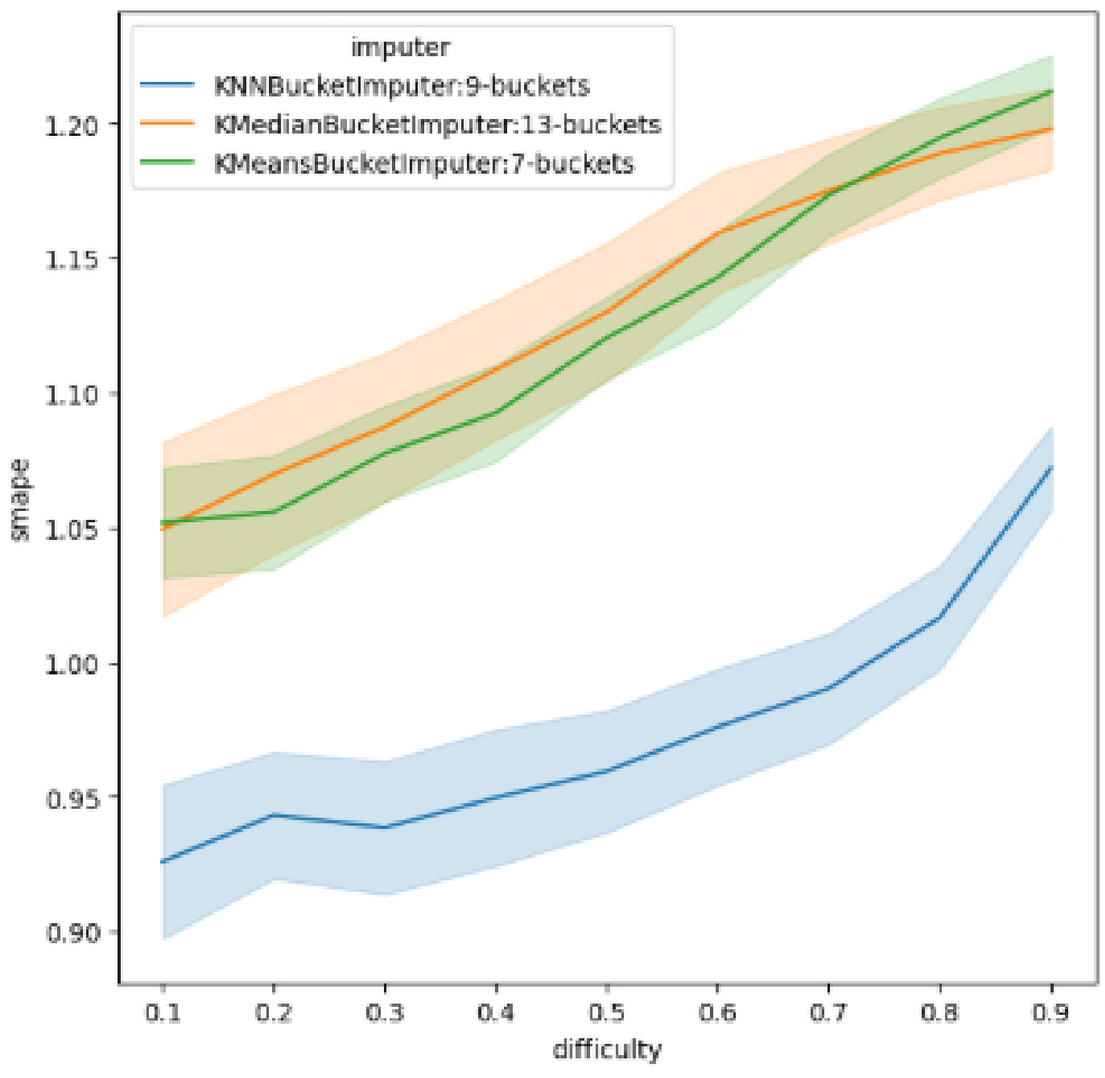
In the result plot we see the performance of all the configurations tested in the experiment. Here, the best performing method seems to be the OldOxariImputer. We also see patterns paterning to different categories of imputers. For instance, the CategoryStatisticsImputers' performance remains constant.

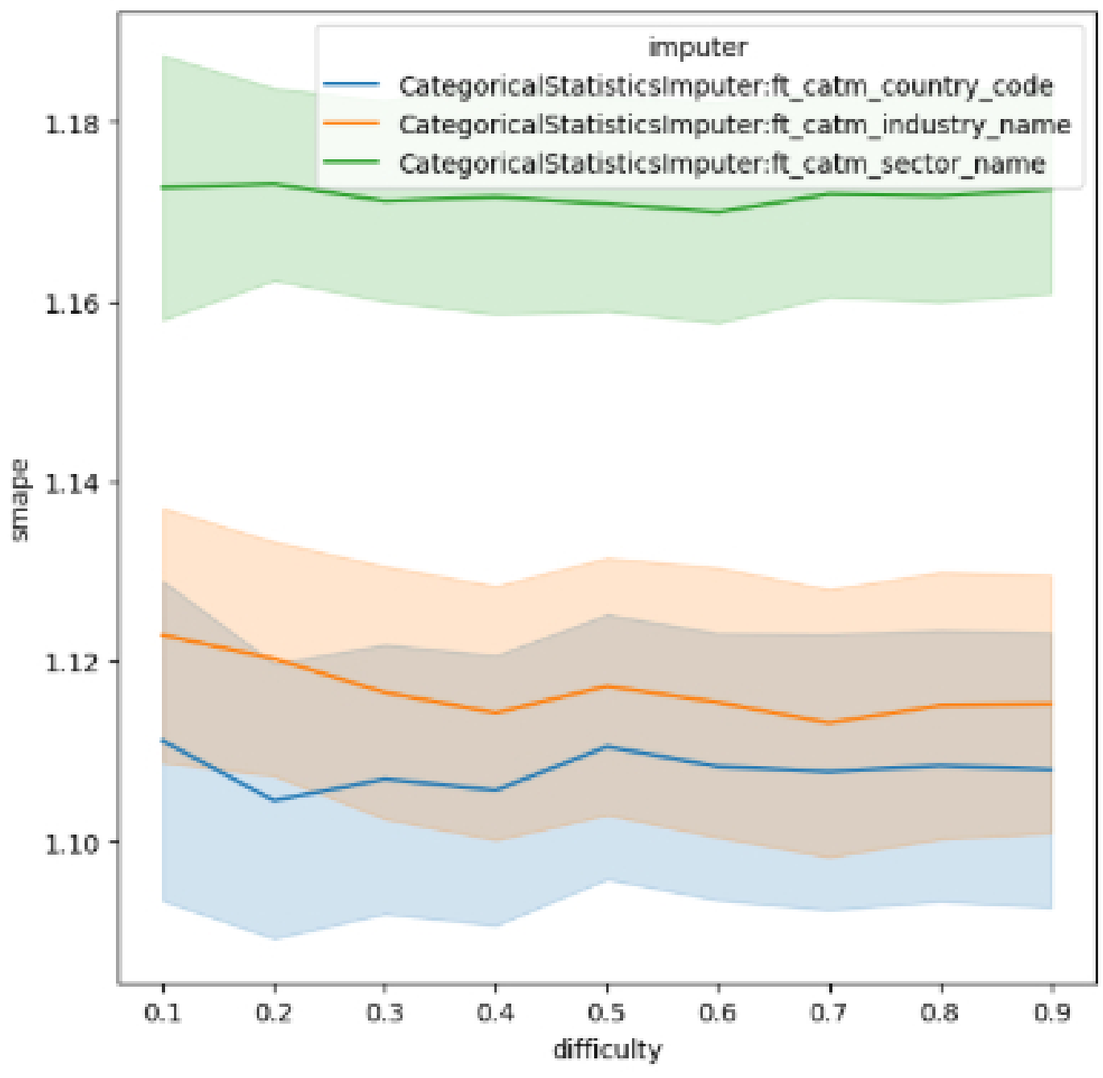


This figure shows the performances per group. Noteworthy is the tie between the TotalAssetsand TotalLiability based NumericalStatisticsImputer methods. We also see that KNNImputation performs better than the other K-methods. Furthermore, the Decision Treebased MVEImputer performs better than most other imputers except the OldOxariImputer. The BayesianRidgeimputer plot behaves eratic due to many predictions being non-finite. At last, the best CategoricalStatisticsImputer appears to be the one based on country codes.

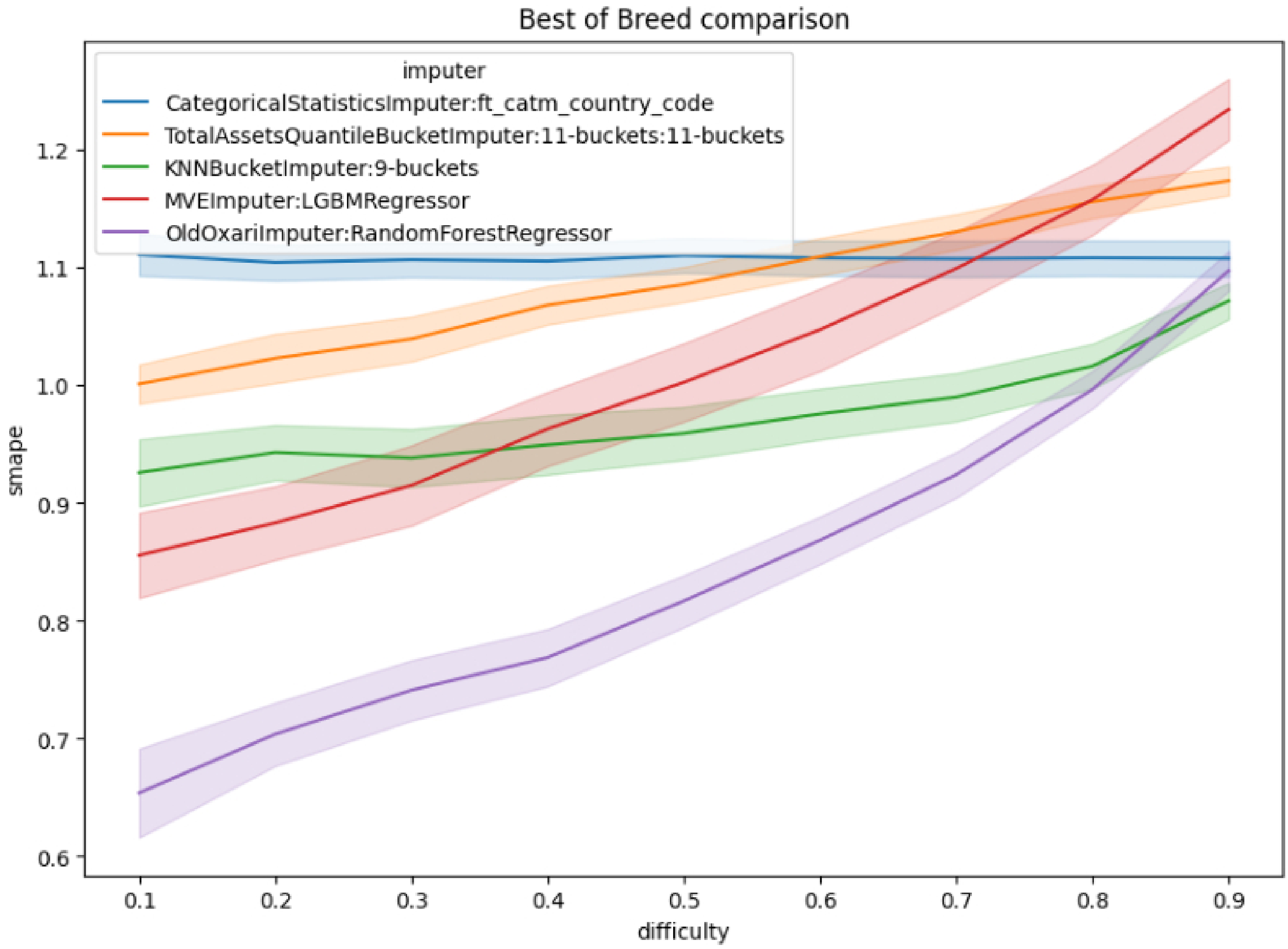








We compare all the "best" methods in each group in the following plot.



Decision

Update 17.01.2024

The overall best imputer appears to be the OldOxariImputer. However, the best approach in each cateogry will be evaluated on the main task of predicting the scope values.