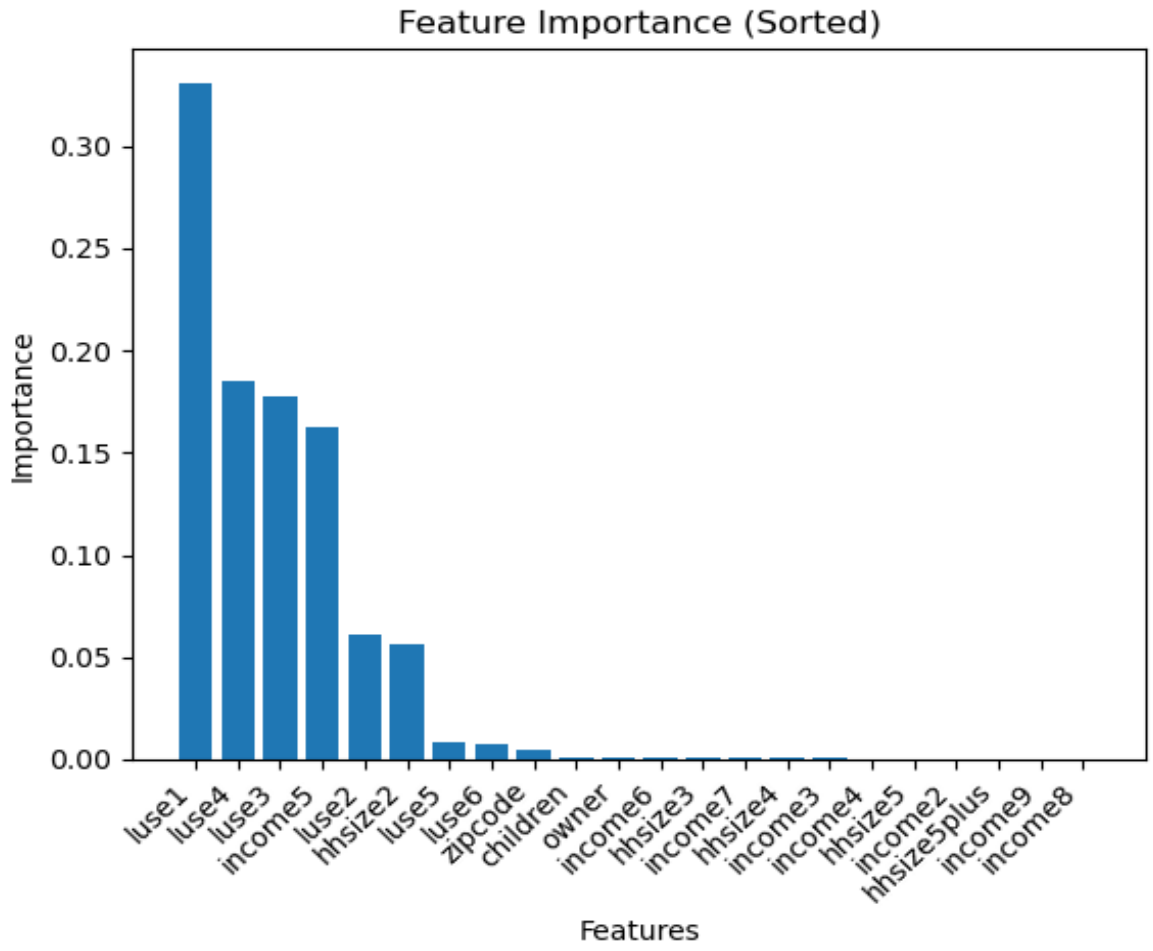
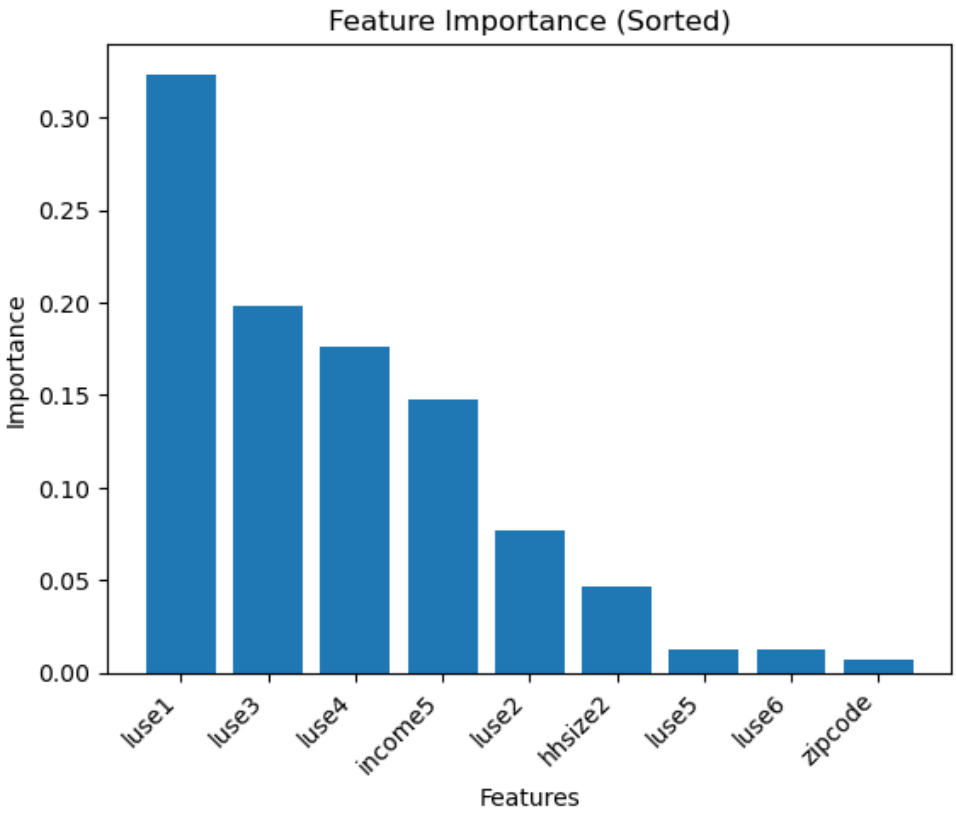
Explanations

1.Feature selection

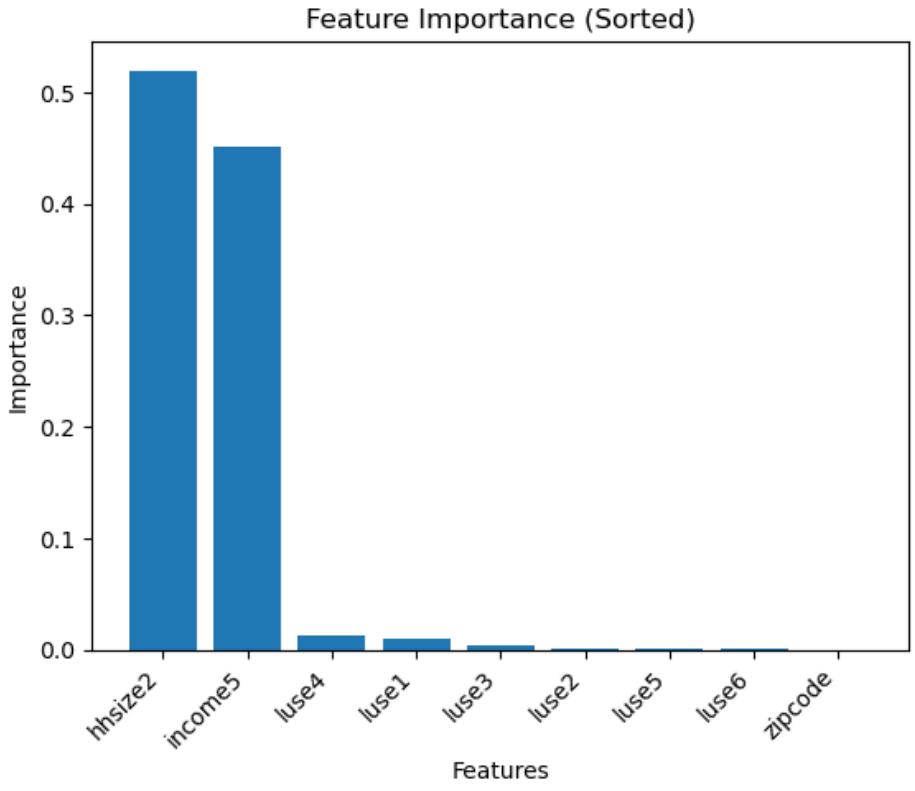
After training the model a few times, I noticed that certain features have a very small contribution, so they were eliminated.
I trained 2 different models Random Forest and XGBoost and I noticed that each has different features to predict the values, so I decided to combine them in a larger model to compensate for the weaknesses of the other.



Features importance after few training



Features importance with Random Forest



Features importance with xgboost

2.Data augmentation

I also noticed that the dataset contains several duplicates, so I eliminated them, thus reducing the dataset by almost 90%. I later decided to use a tool to proportionally add an equal number of rows based on the following ranges for size: [size < 100] -> I added ~240 rows, [1000 > size>=100] I added ~1000 rows, [size > 10000] I added 10000. For [10000 > size>=1000] the diversity condition was already satisfied. Also, due to the difference in values between the columns, I decided to standardize the data before entering it in the training phase.

```
3318
dist_matrix: 100%|##########| 1323/1323 [07:07<00:00,  3.09it/s]
r_index: 100%|##########| 661/661 [00:00<00:00, 900.87it/s]
2583
dist_matrix: 100%|##########| 565/565 [01:21<00:00,  6.94it/s]
synth_matrix: 100%|##########| 565/565 [00:00<00:00, 938.58it/s]
r_index: 100%|##########| 113/113 [00:00<00:00, 888.62it/s]
4432
dist_matrix: 100%|##########| 1191/1191 [05:41<00:00,  3.48it/s]
r_index: 100%|##########| 595/595 [00:00<00:00, 975.55it/s]
6867
dist_matrix: 100%|##########| 1590/1590 [09:58<00:00,  2.66it/s]
synth_matrix: 100%|##########| 1590/1590 [00:01<00:00, 990.01it/s]
11631
```

Data augmentation for size>10000

3.Model

For the model I initially chose Random Forest and xgboost because it combines the predictions of multiple individual models (decision trees) to produce a more robust and accurate prediction. Later I used it in combination with Adaboost because It focuses on correcting the mistakes made by previous models in the ensemble and is less prone to overfitting compared to some other complex models. For the model I initially chose Random Forest and xgboost because it combines the predictions of multiple individual models (decision trees) to produce a more robust and accurate prediction. Then, I considered using another very useful model such as xgboost because it includes L1 (Lasso) and L2 (Ridge) regularization and can be used in various datasets. Later, observing the important features of each model, I had to combine them using a voting classifier thus targeting the weak points of each algorithm.

To adjust the parameters of the models, I used Grid Search, thus obtaining the best combination of parameters.

```
|: GridSearchCV(cv=5,
               estimator=Pipeline(steps=[('preprocessor',
                                          ColumnTransformer(remainder='passthrough',
                                                            transformers=[('scaler',
                                                                           StandardScaler(),
                                                                           ['lusage',
                                                                            'luse1',
                                                                            'luse2',
                                                                            'luse3',
                                                                            'luse4',
                                                                            'luse5',
                                                                            'luse6'])])),
                                         ('model', RandomForestRegressor())]),
               n_jobs=-1,
               param_grid={'model__max_depth': [None, 10, 20, 30],
                           'model__max_features': ['auto', 'sqrt', 'log2'],
                           'model__min_samples_leaf': [1, 2, 4],
                           'model__min_samples_split': [2, 5, 10],
                           'model__n_estimators': [100, 200, 250, 275, 300]},
               scoring='neg_mean_squared_error')
```

3. Conclusion

In the end, even if the MSE is quite high, I tend to think that the model can predict the "size" value quite well. I added a series of predictions for the values that contained Nan in the "size" column (predicted_size.csv) .