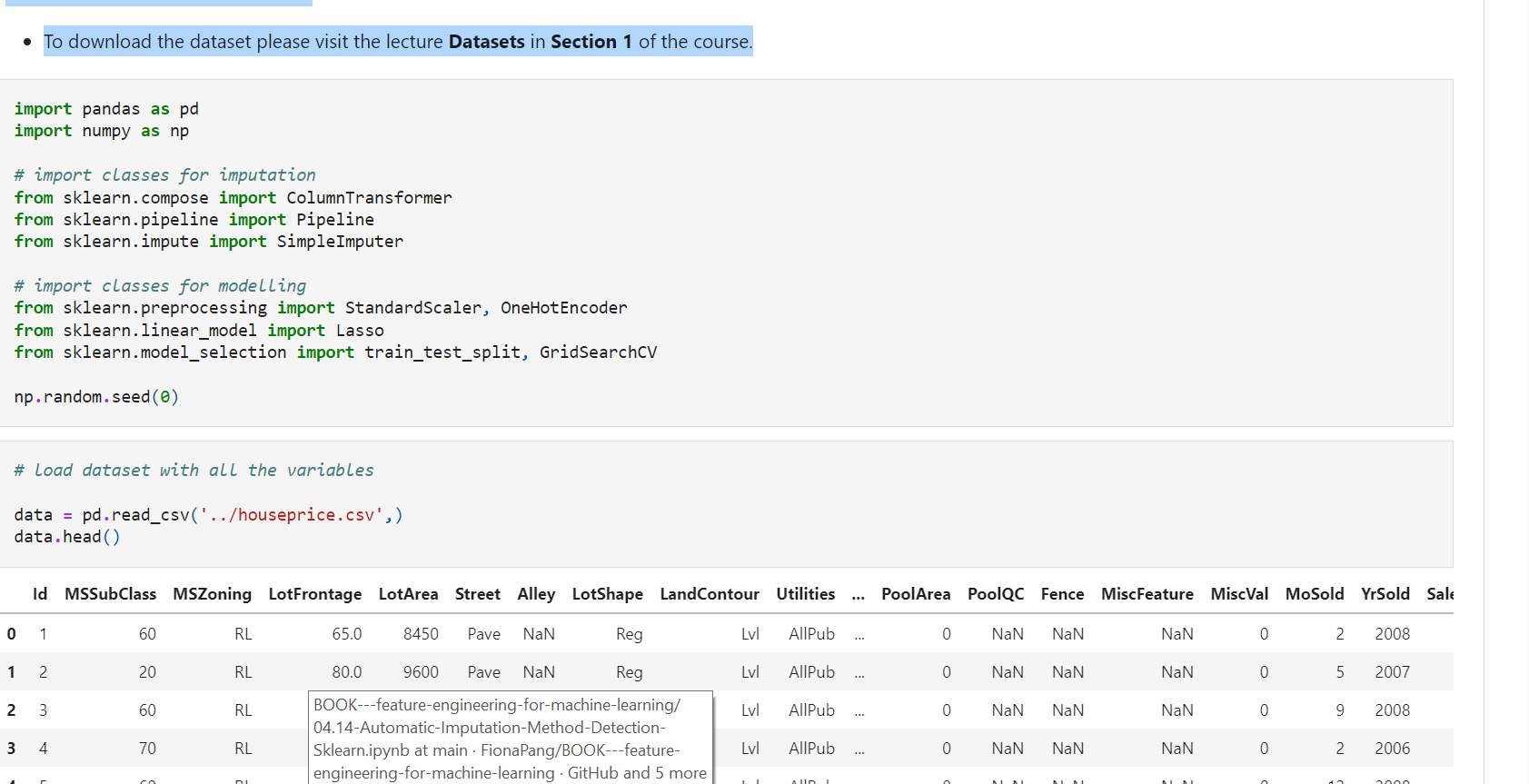
**Automatic selection of best imputation technique with Sklearn**

In this notebook we will do a **grid search over the imputation methods** available in Scikit-learn to determine which imputation technique works best for this dataset and the machine learning model of choice.

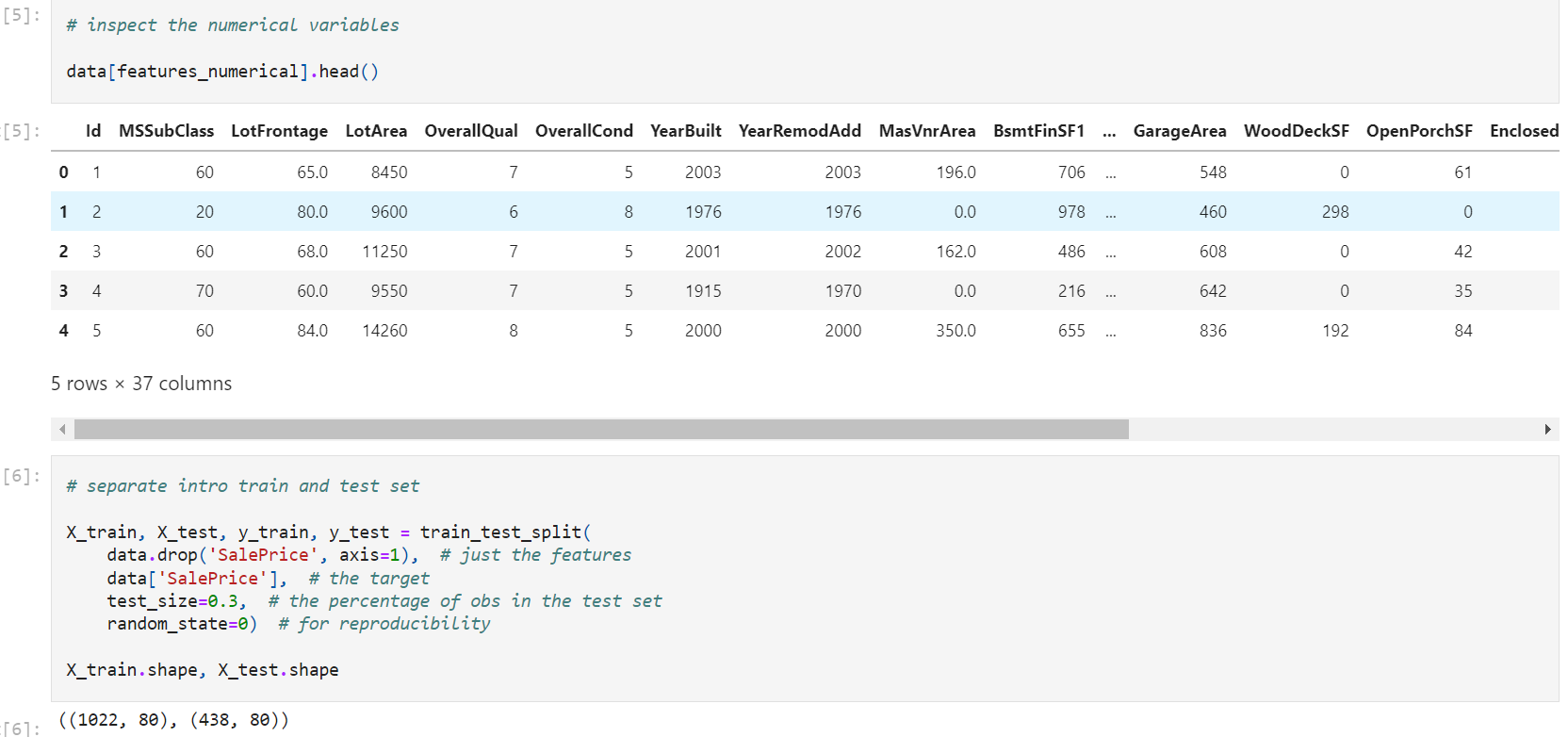
We will also train a very simple machine learning model as part of a small pipeline.

We will use the House Price dataset.

* To download the dataset please visit the lecture **Datasets** in **Section 1** of the course.











When setting the grid parameters, this is how we indicate the parameters:

preprocessor**numerical**imputer\_\_strategy': ['mean', 'median'],

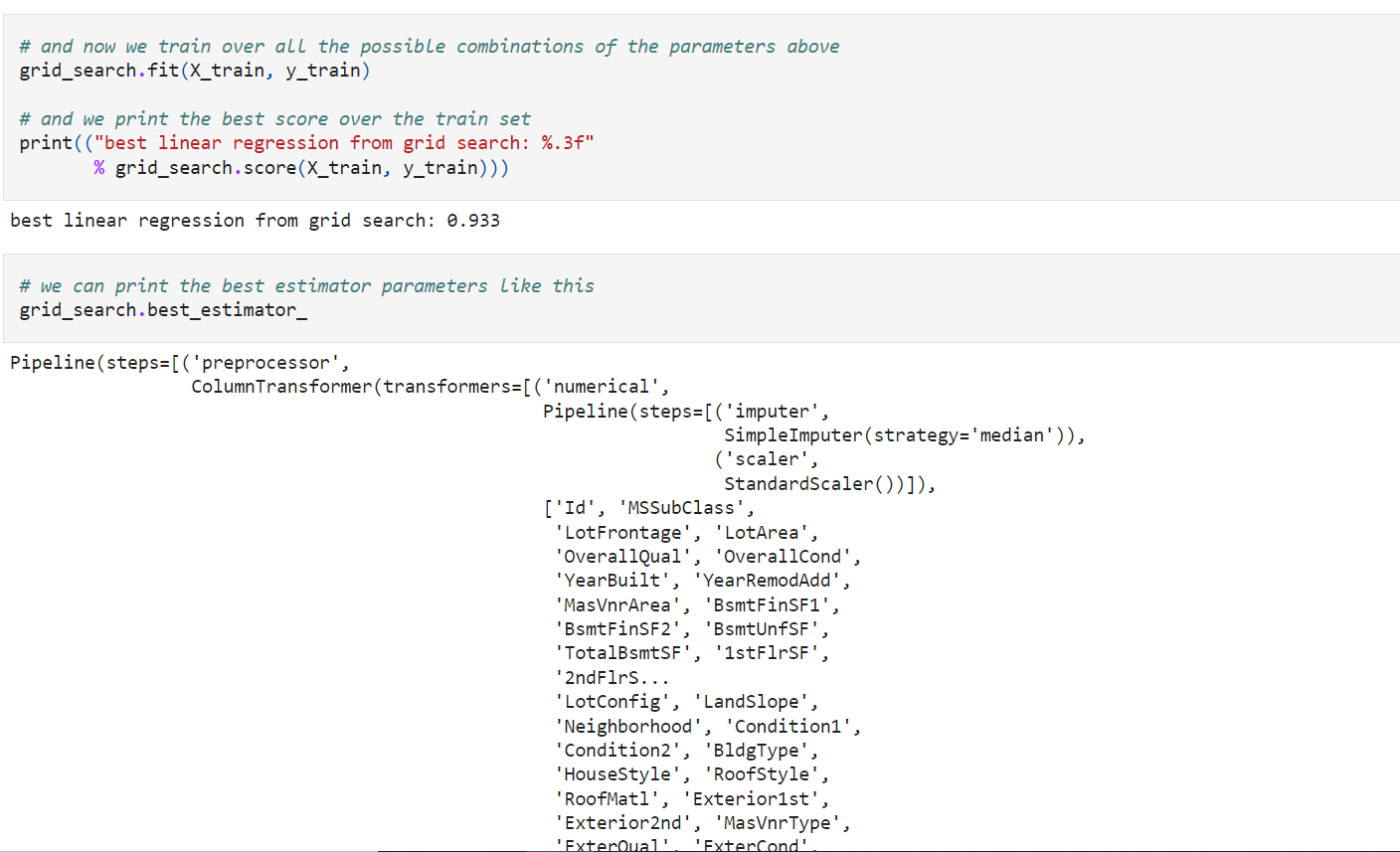
the above line of code indicates that I would like to test the **mean and the median** in the imputer step of the numerical processor.

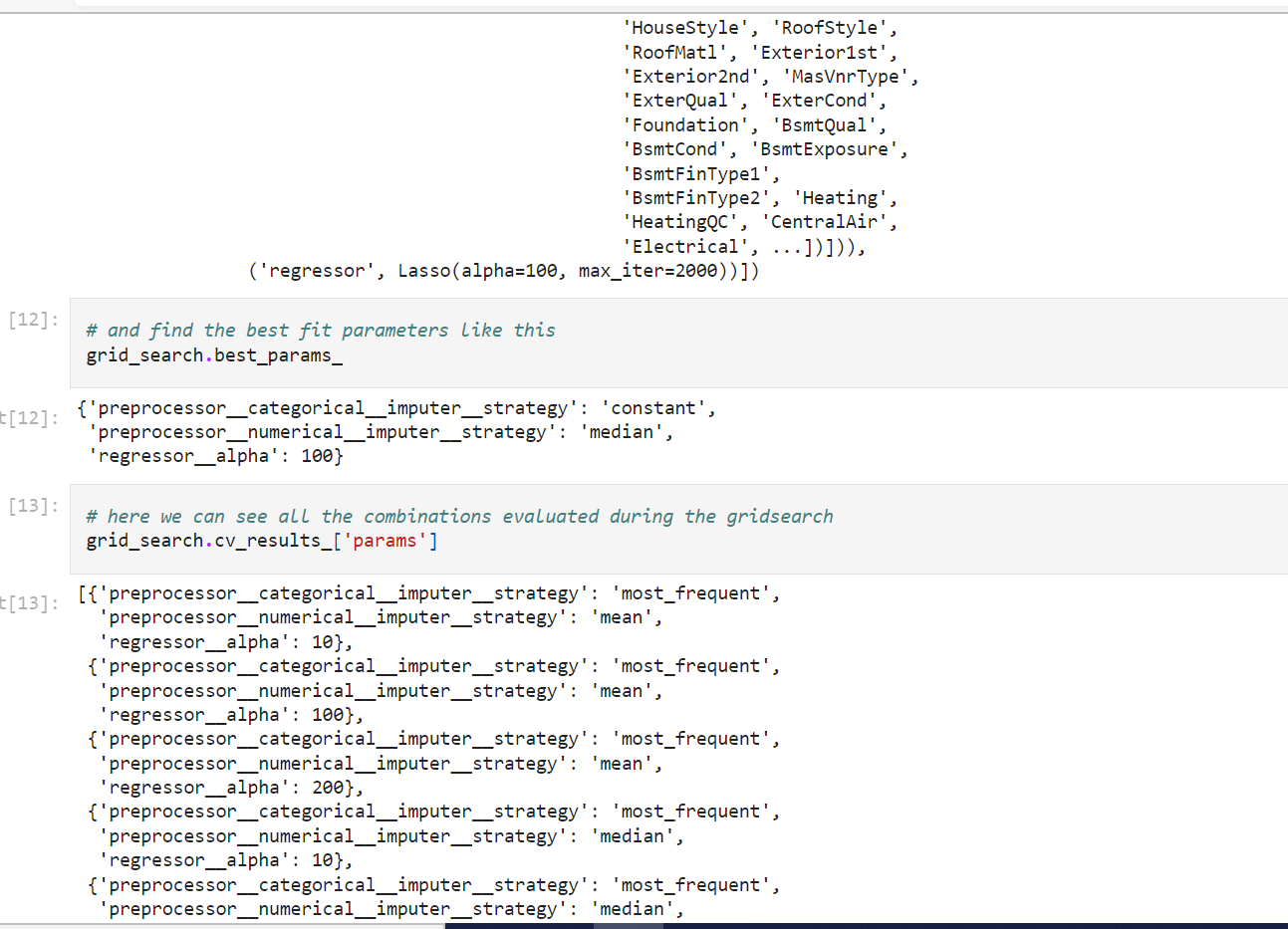
preprocessor**categorical**imputer\_\_strategy': ['most\_frequent', 'constant']

the above line of code indicates that I would like to test the most frequent or a constant value in the imputer step of the categorical processor

classifier\_\_**alpha': [0.1, 1.0, 0.5]**

the above line of code indicates that I want to test those 3 values for the alpha parameter of Lasso. Note that Lasso is the 'classifier' step of our last pipeline







**Mean / Median Imputation ==> Feature-engine**

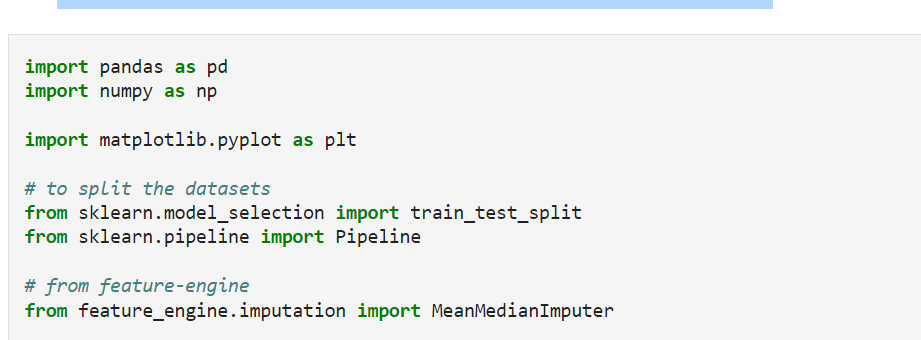
**Feature-engine** is an **open source Python package originally designed to support this course**, but has increasingly gained popularity and now supports transformations beyond those taught in the course. It was launched in 2017, and since then, several releases have appeared and a growing international community is beginning to lead the development.

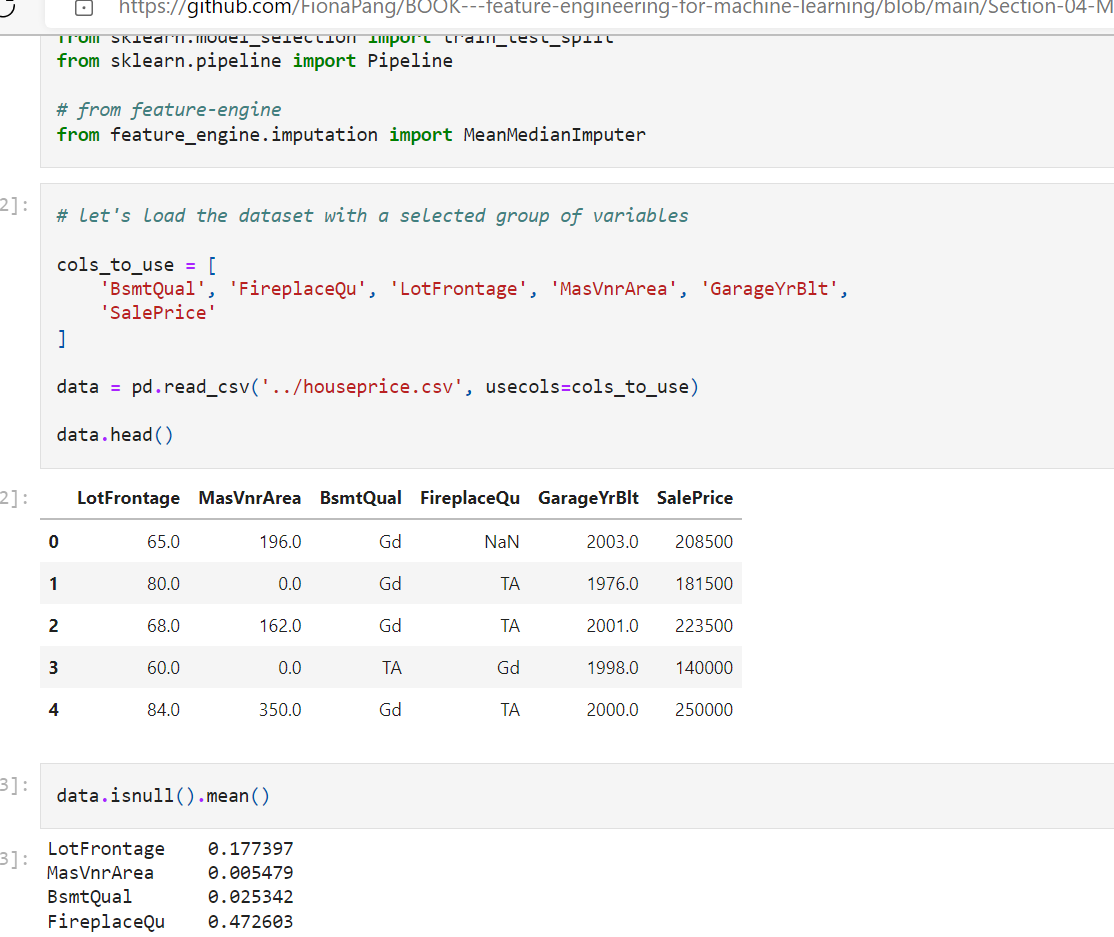
* Feature-engine **works like to Scikit-learn**, so it is easy to learn
* Feature-engine allows you to **implement specific engineering steps to specific feature subsets**
* **Feature-engine can be integrated with the Scikit-learn pipeline** allowing for smooth model building
* **Feature-Engine allows you to design and store a feature engineering pipeline with different procedures for different variable groups.**
* Make sure you have installed feature-engine before running this notebook.

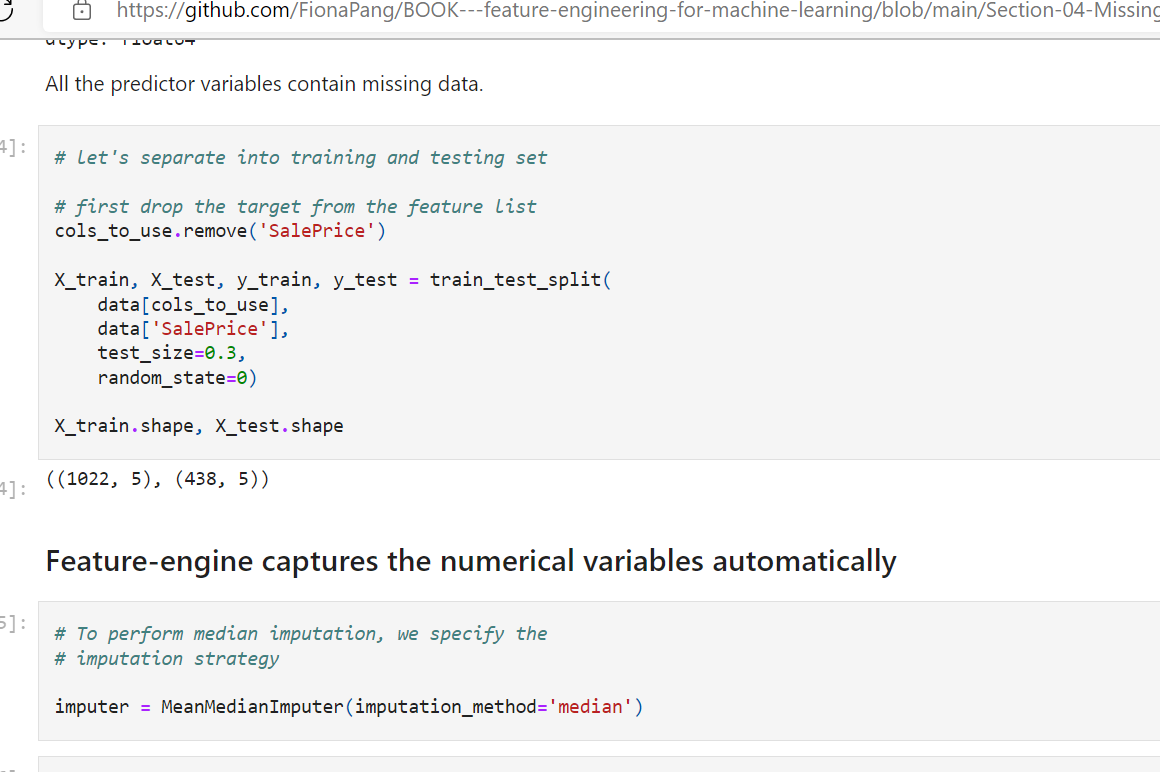
**In this demo**

We will use Feature-engine to perform mean or median imputation using the Ames House Price Dataset.

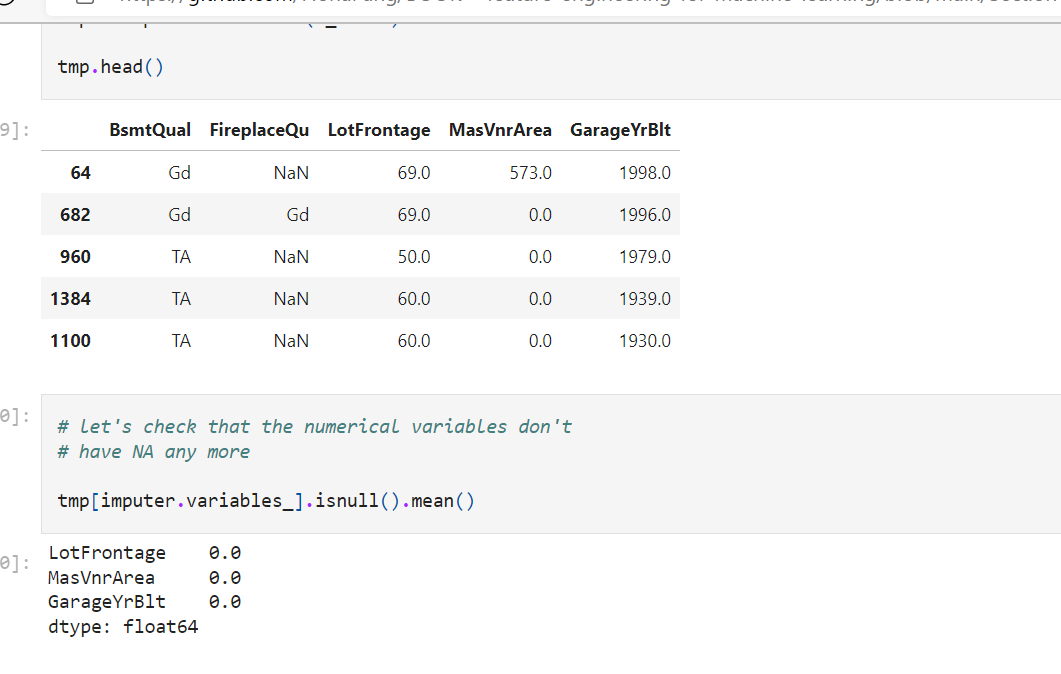
* To download the dataset visit the lecture **Datasets** in **Section 1** of the course.



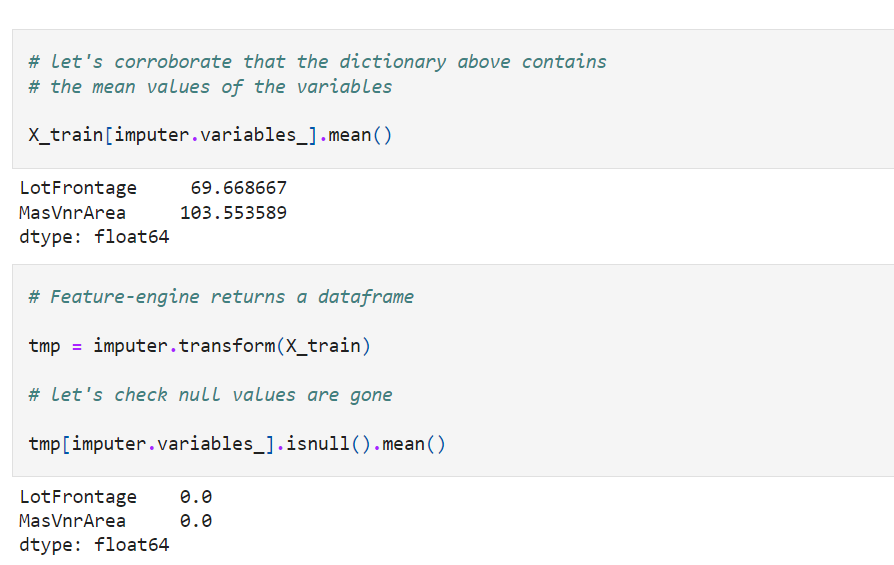


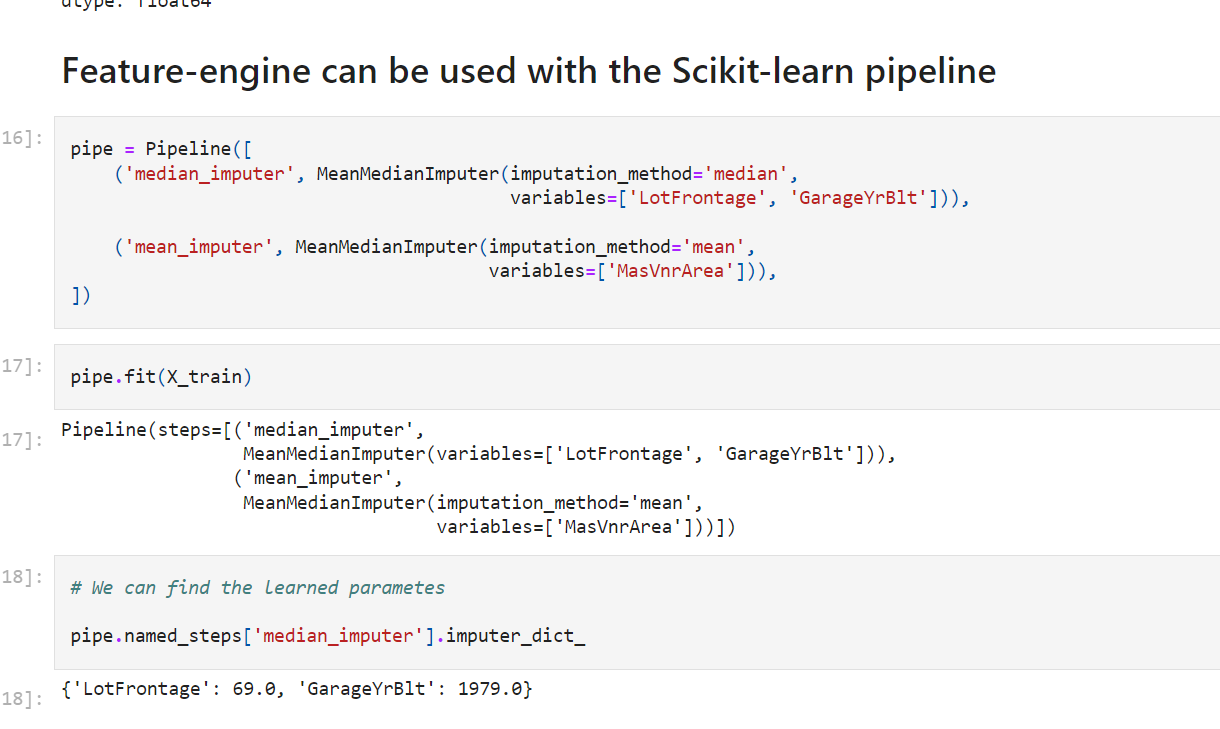


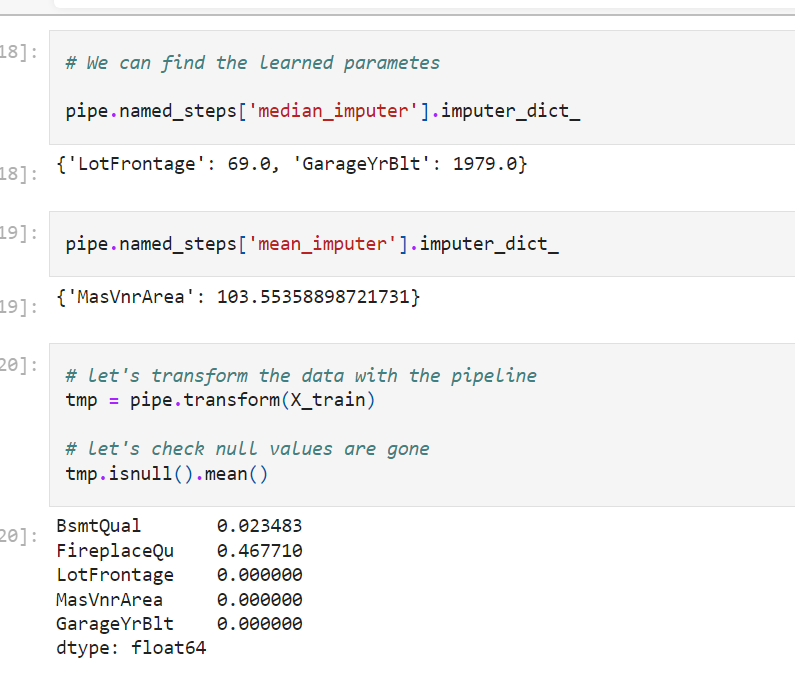












[**GitHub - solegalli/feature-engineering-for-machine-learning: Code repository for the online course Feature Engineering for Machine Learning**](https://github.com/solegalli/feature-engineering-for-machine-learning)

Univariate imputation:

We get the value statistically derived from that variable only

Mean, median,mode

Random sample

Multivariate imputation: we use other variables to estimate missing values of one variables

KNN

MICE(multivariate imputation of chained equation)

MissForest: extend MICE using random forest

Extension of mice, uses random forest

Aim :

Predict as accurate as possible real values

Instead of actually a real value , accurate prediction of target

One model to predict missing value to one variable

3 variables: 3 models

This cause complexity

KNN:

Determine missing data point value as the weighted avg value of its k nearest neighour

Most likes the missing value similar to that observation

* First find knn with help of other neighbours similar to it

NA replacement= wi\*value+w2\*value2-----/k(nearest neighbour)

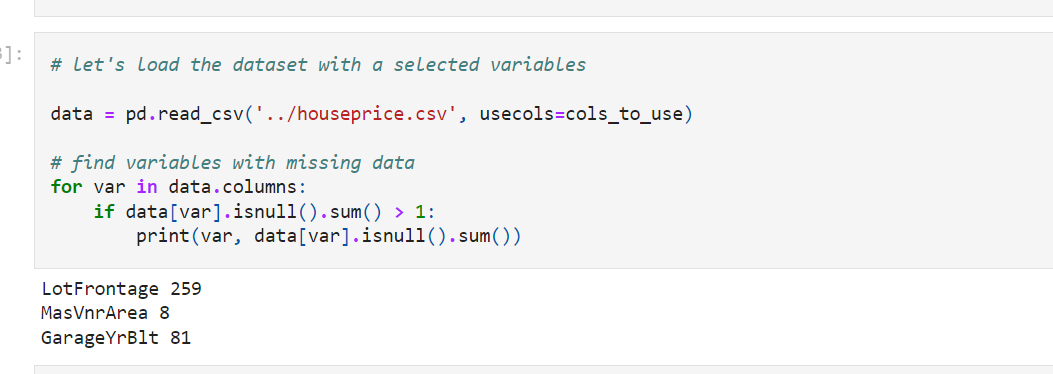
Skikitlearn used 2 methods for weight

1. All neighbour equal weight- uniform
2. Distance=w1=i/Euclidean distance(neighbour-distance)

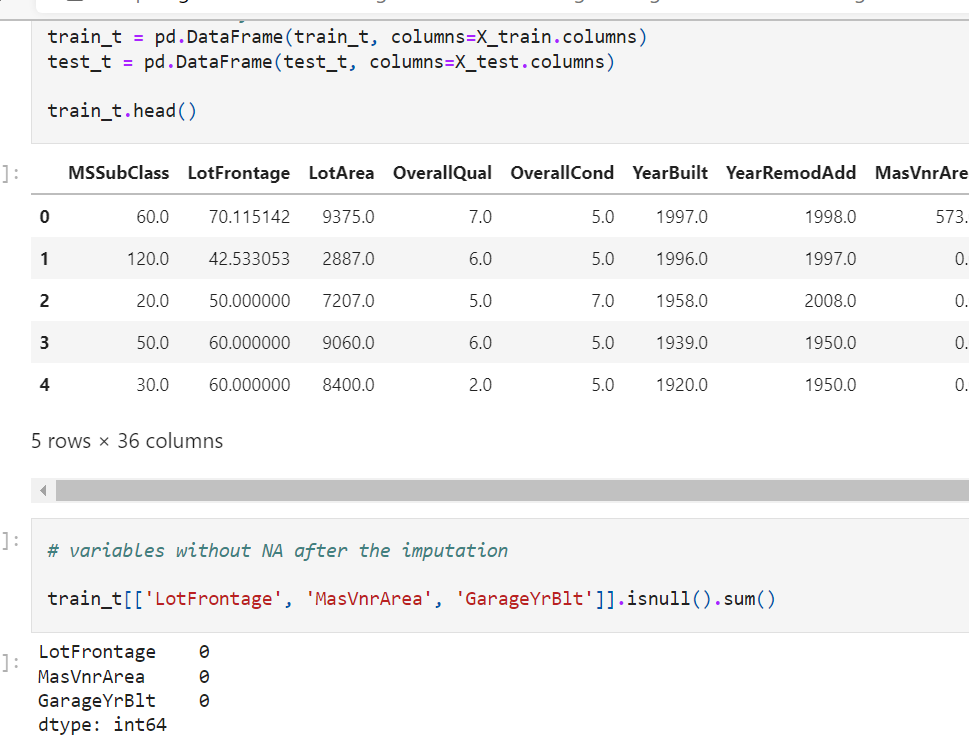
Find K(neighbours)?

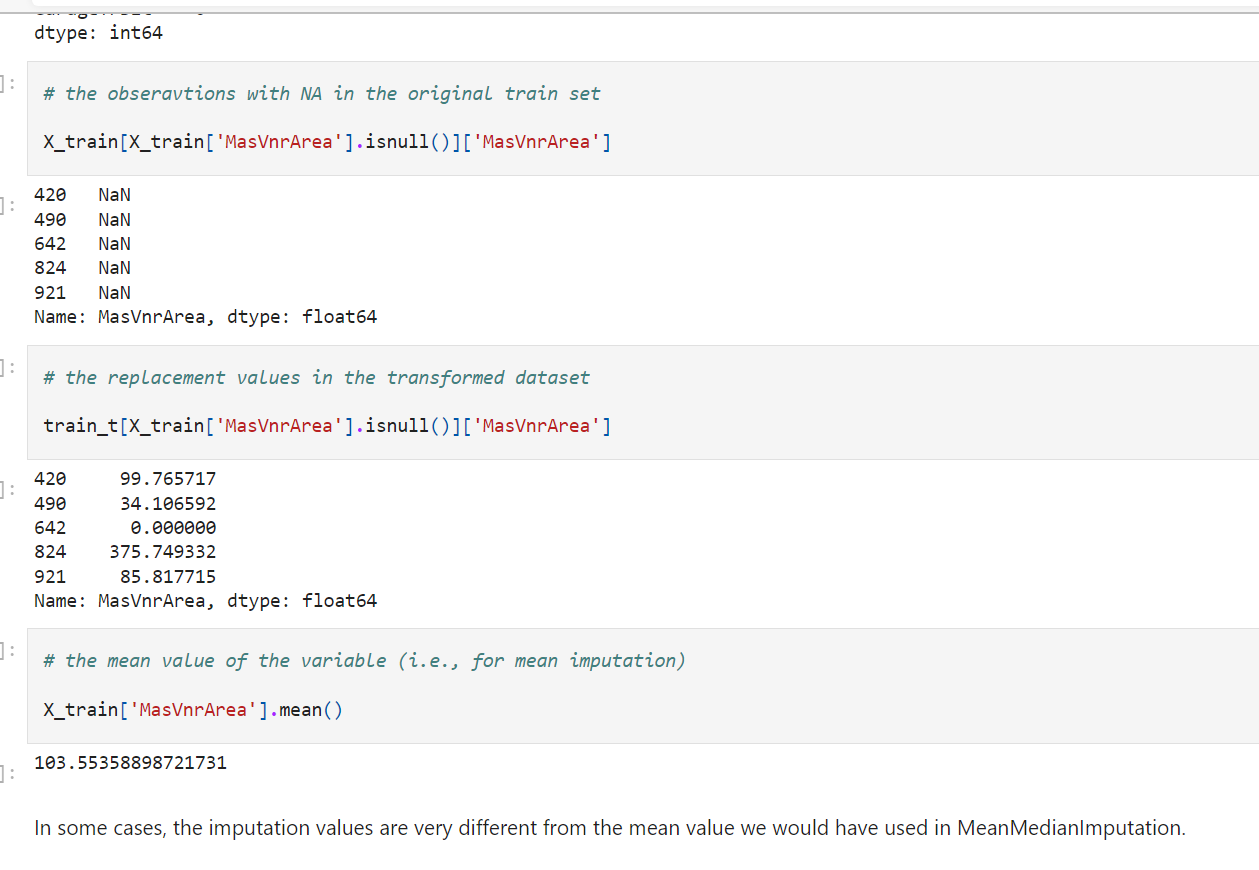
Upto 20 percentage misiing value, can use k vlue between 10 and 20

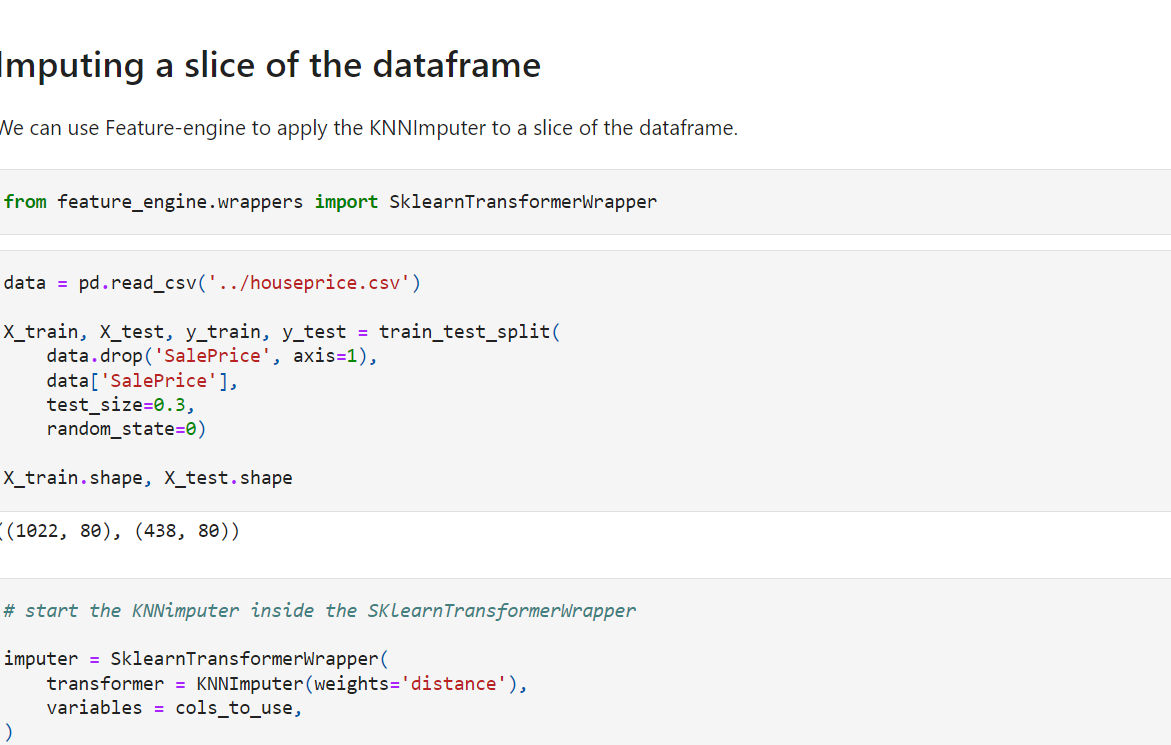


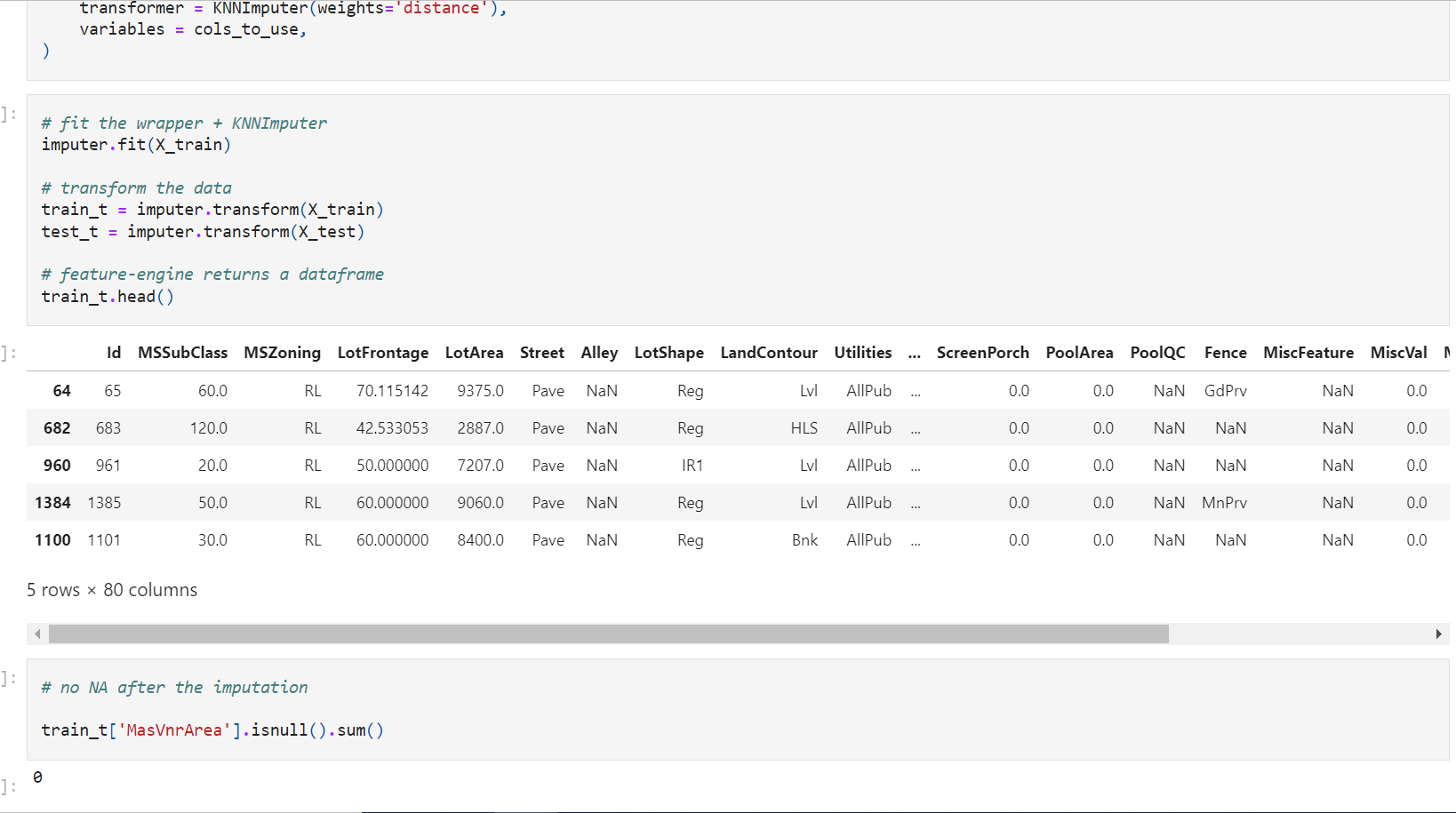


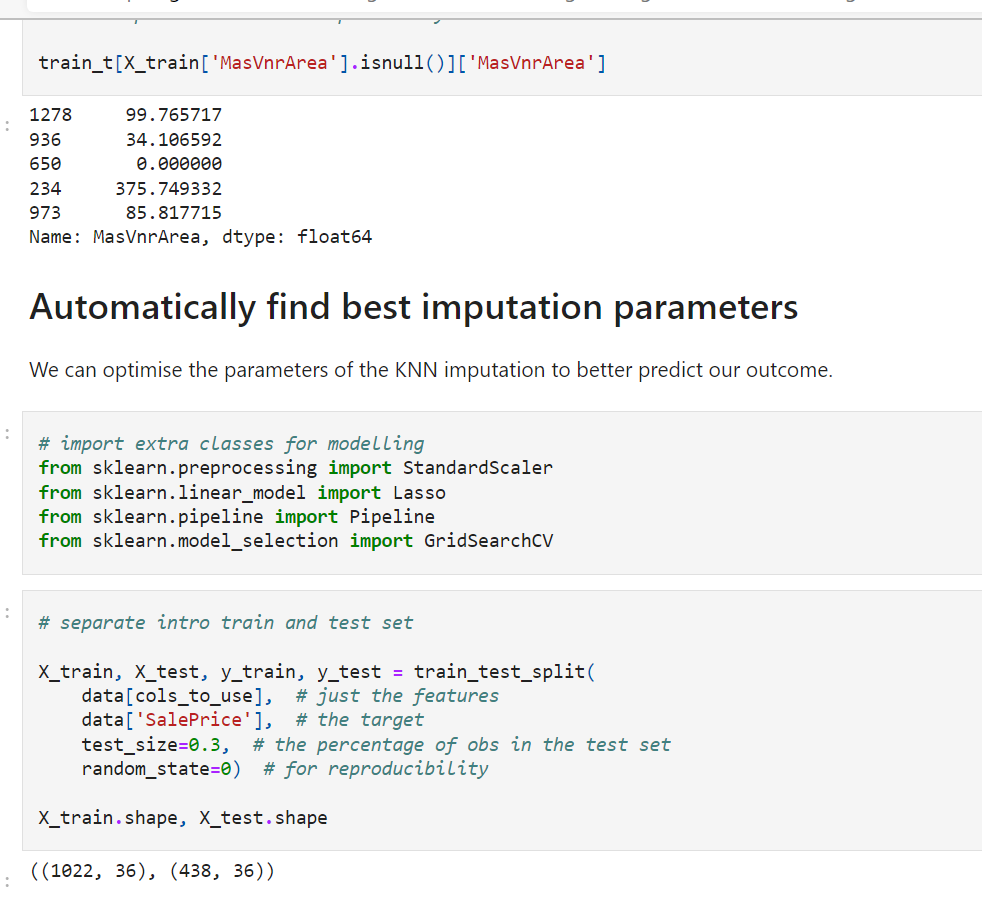




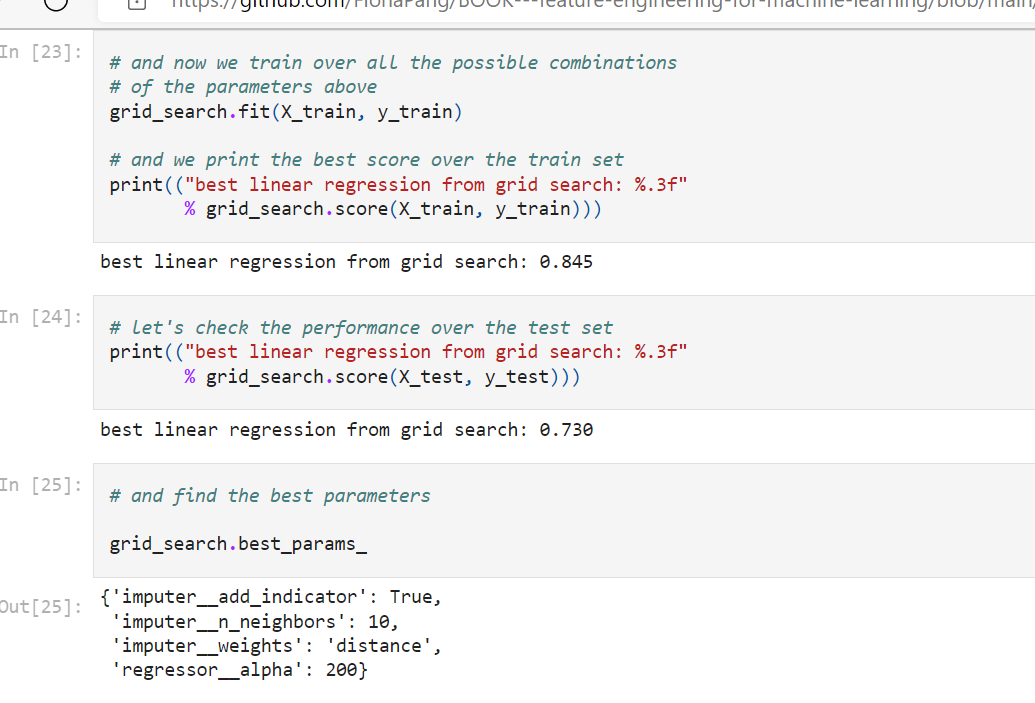




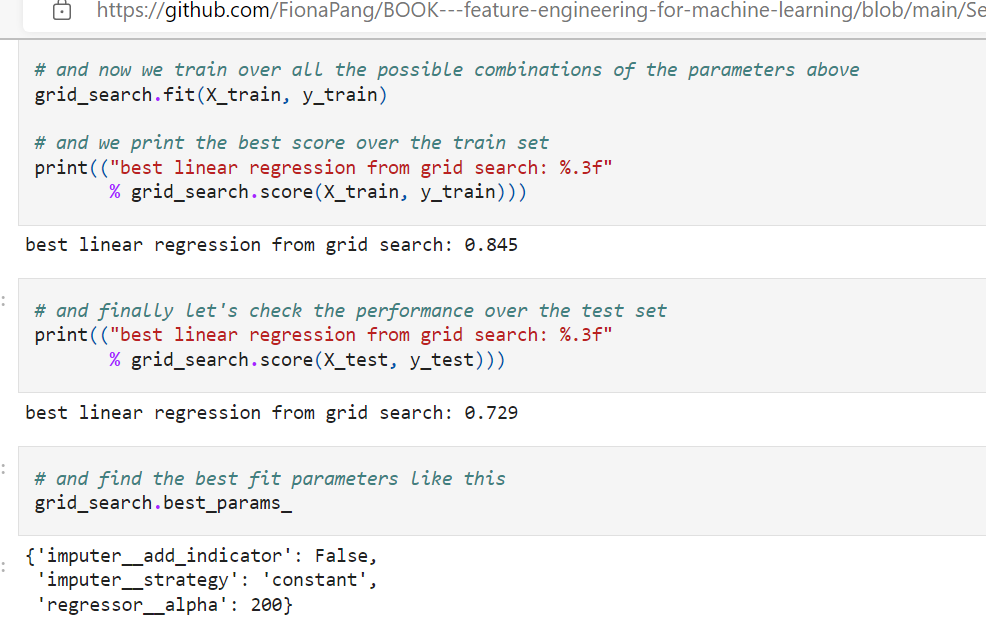












--

MICE: Multivariate imputation of Chained Equation

A series of models where each variable is modelled – conditional opon other variables in the data

Each incomplete variable is imputed by a separate model

--

More a frame work then imputation tech

Establishes procedure with witch we can deal those models

The variables are modelled upon other variables

Na are replaced by model predictions

Step;1

We impute all variables with simple imputation tech

We take one variables are reverted to null

The variables modelled upon the other variables –

Na will be replaced by model prediction

Now—same for second variables

Once all variables models, missing values will be replaced

One round of imputation completed

Procedure repeats for n times

Usually 10 imputations to find sample parameters

When we need multiple rounds of imputation:

In first round we are modelling the variables based on others (those may also contain missing data)

Predictions may be biased

As we continue regress one variable upon other

We obtain better estimate for NA

These estimates are use to regress the other variables

Thus returning more accurate predictions

Assumptions:

MAR(missing at random)

NA in the variables modelles by other variables in dataset- doent depend on external sourses

Mice consideration:

Variables may have linear or non linear relationship

We can find best model to predict missing data i.e linear regression, bayes, tree based algorithm

Optimize model performance

--depending on variable nature:

We should use different models

Binary : claasification models

Continuous : regression algorithm

Discrete : poison

Which variables should we use as predictors:

Authors suggest that using every available bit of available information – yields multiple imputation- which has min bias and max certainty

Num of predictors should be as large as possible

Include all variables that will be used in final model

MISSFOREST:

Extension of MICE, where random forests are used to regress the NA to other variables

Works well with missed data types

Robust and accurate

Handles non -linear relationships and variable iterations

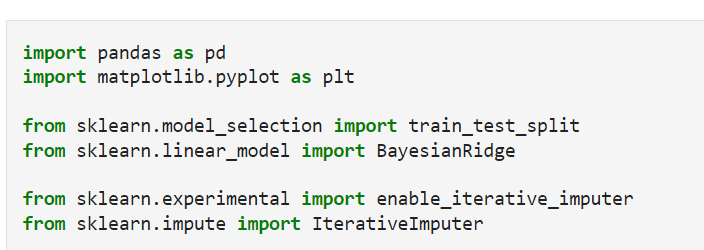
**Multivariate imputation of Chained Equations**

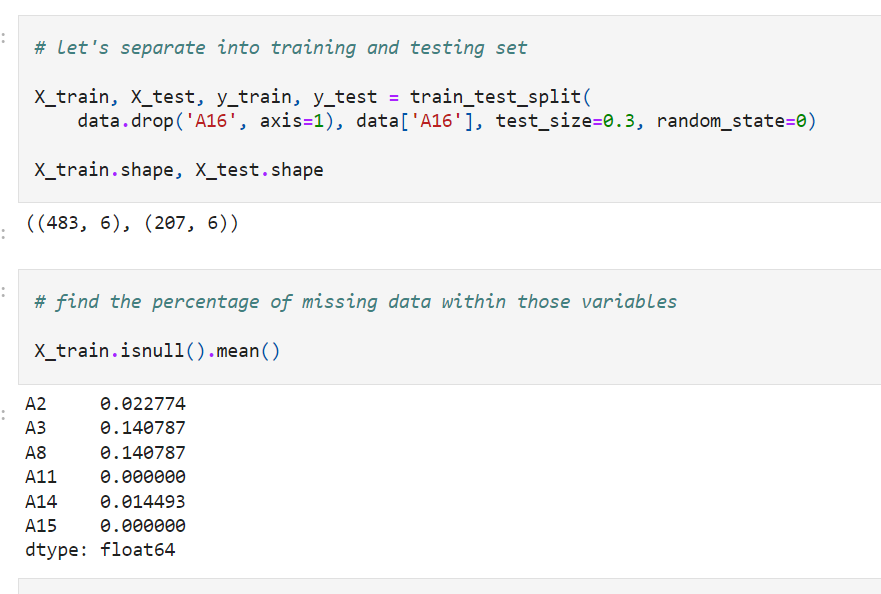
In this notebook we will implement MICE using various machine learning models to estimate the missing values.

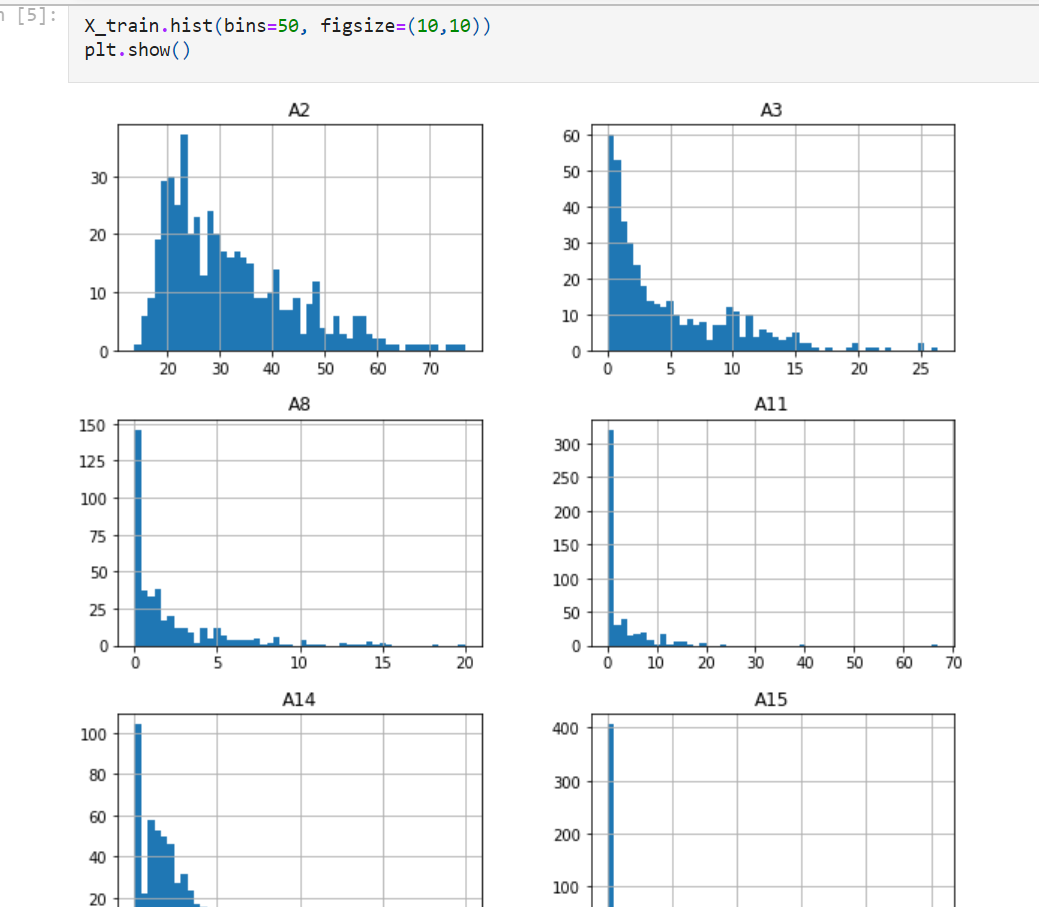
[IterativeImputer from Sklearn](https://scikit-learn.org/stable/modules/generated/sklearn.impute.IterativeImputer.html#sklearn.impute.IterativeImputer)

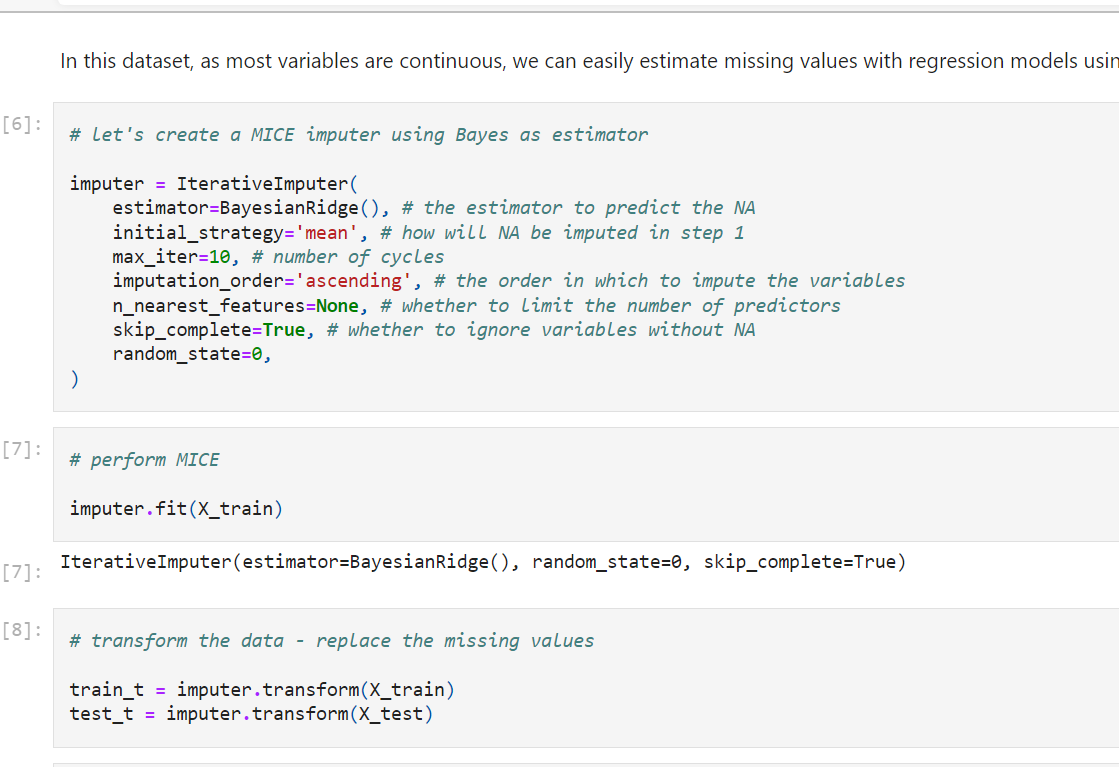
* Same model will be used to predict NA in all variables
* Can't use classification for binary variables and regression for continuous variables

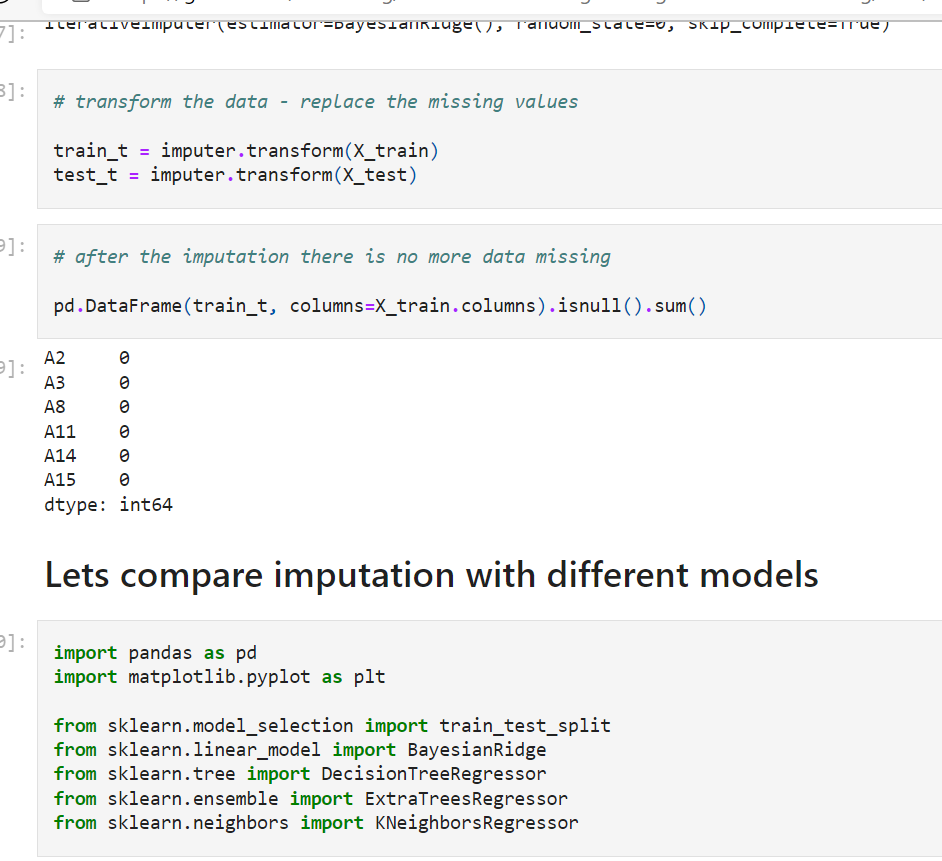
For a more sophisticated imputation, we would have to assemble the imputers / models manually.





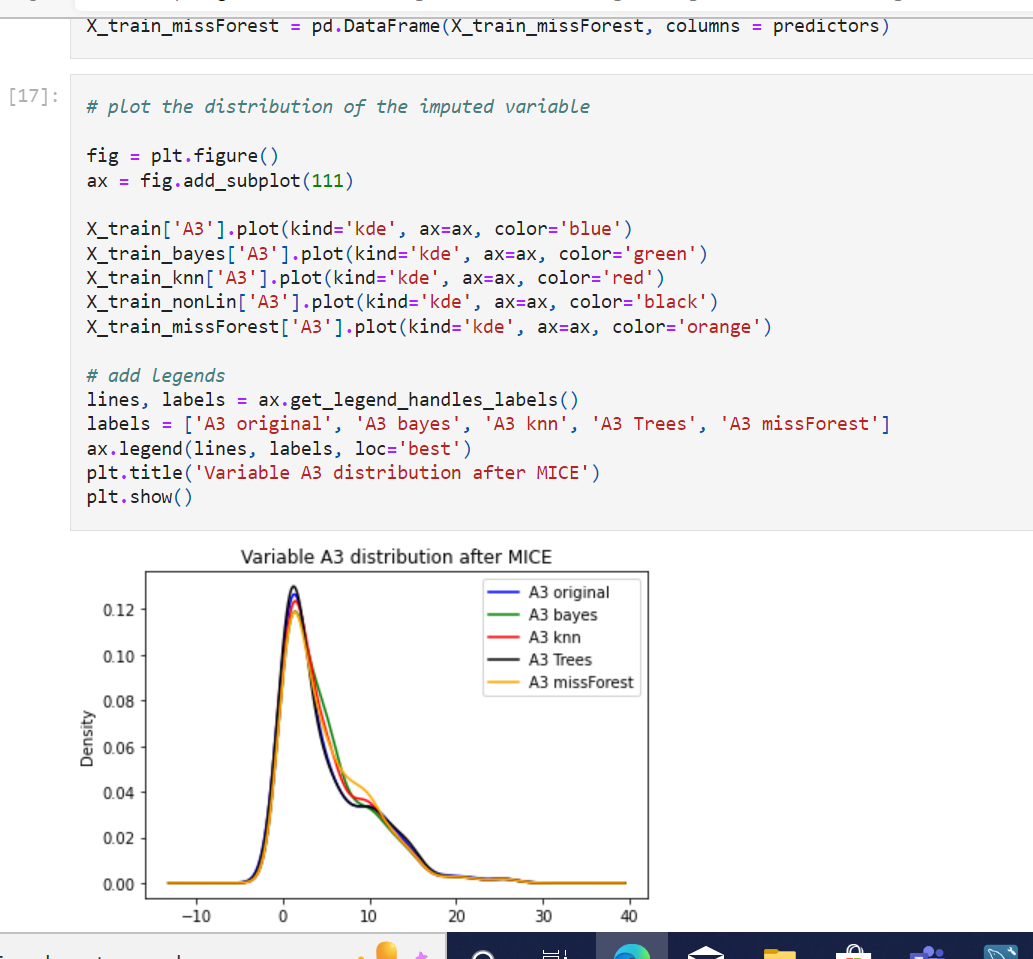












**Additional reading resources (Optional)**

Original Articles

* [mice: Multivariate Imputation by Chained Equations in R](https://stefvanbuuren.name/publications/2011%20MICE%20-%20JSS.pdf)
* [Multiple imputation by chained equations: what is it and how does it work?](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3074241/pdf/MPR-20-40.pdf)
* A Multivariate Technique for Multiply Imputing Missing Values Using a Sequence of Regression Models
* [MissForest—non-parametric missing value imputation for mixed-type data](https://academic.oup.com/bioinformatics/article/28/1/112/219101)
* [Missing value estimation methods for DNA microarrays ((KNN Impute)](https://academic.oup.com/bioinformatics/article/17/6/520/272365)

See also

* [Sklearn Documentation](https://scikit-learn.org/stable/modules/impute.html#multivariate-feature-imputation)