PROJECT FLOW

Team ID	PNT2022TMID53349
Project Name	Car Resale value Prediction

1. DATA PREPROCESSING

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put it in a formatted way. So, for this, we use data preprocessing tasks.

Real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

It involves below steps:

• Getting the dataset

• Importing libraries

• Importing datasets

• Finding Missing Data

• Encoding Categorical Data

• Splitting dataset into training and test set

2. TRAIN-TEST SPLIT FUNCTION

“Sklearn” or “scikit-learn” is a python library that consists of various data processing features, which are used for clustering, model selection, and classification. “Model-selection” is a method to frame a blueprint. This blueprint is used to analyze data and then use that data to measure new data. While making predictions, the selection of appropriate models aids to generate accurate results. In order to test a model using a specific dataset, first that particular model must be trained against the same dataset. To split one dataset and perform training and testing, the “train-test-split” function can be used. In evaluating data mining models, the separation of data into training and testing is an important part, where the majority of data is used for training, and a small amount of data is used for the testing purposes of the model.

Scikit learn

“train-test-split” is a function in Sklearn model selection for splitting a dataset into two subsets. One is for training, and another is for testing the model. This function automatically divides data into subsets instead to perform that operation manually. By default, train-test-split takes random partitions to split into two subsets of the data. This procedure is widely used as it is fast and easy to perform. This procedure is not recommended in cases where there is a small dataset, additional configurations are needed, and the dataset is not balanced.

1. The basic syntax of train-test-split would be like: train-test-split(X, y train-size=0., test-size=0., random-state=*)

2. There are several parameters in test-train-split, they are:

- **X, y** – In general, “X” is the variable used to store the dataset for training purposes, “y” is the variable used to store the dataset for testing purposes.
- **train-size** – This parameter is used to set the size of the dataset for training purposes, “None” is used as a default argument, “int” is used when an exact number of samples is known, “float” is used which ranges from 0.1 to 1.0.
- **test-size** – This parameter is used to set the size of the dataset for testing purposes, “None” is used as a default argument, “int” is used when an absolute number of samples is known, “float” is used which ranges from 0.1 to 1.0, if train-size is also assigned “None” then 0.25 is set to complement test-size.
- **random-state** – This parameter is used to control the shuffling during the splitting of the data. “None” is used as the default argument, “int” is used to reproduce output across multiple function calls.

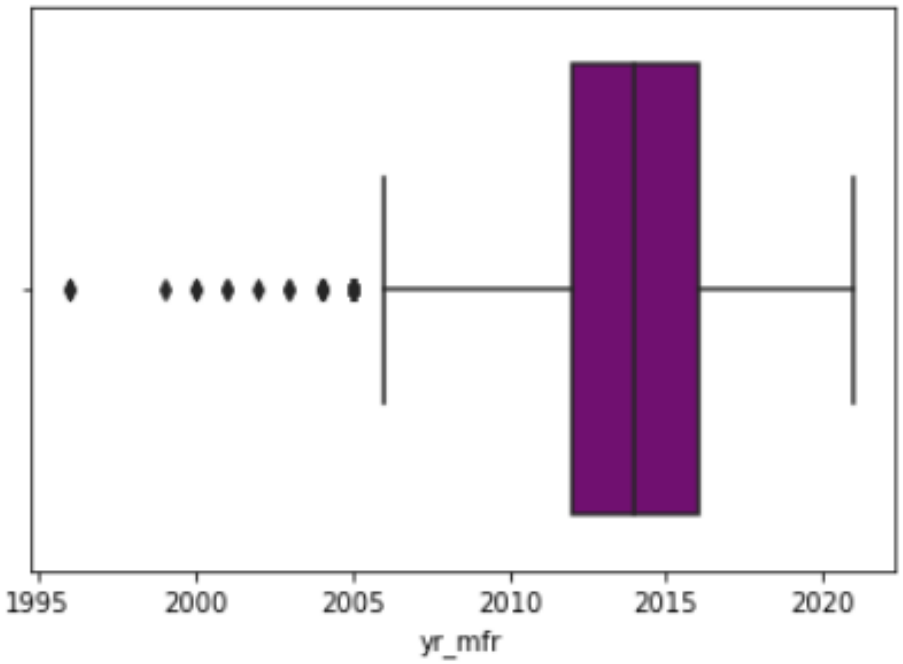
3. By default, the train-test-split function splits the data into random subsets.

4. The standard way to split is 80:20 for training and testing respectively, adjustments are recommended depending on parameter complexity and size of the dataset.

3. DATA CLEANING

The practice of correcting or deleting inaccurate, damaged, improperly formatted, duplicate, or incomplete data from a dataset is known as data cleaning. There are numerous ways for data to be duplicated or incorrectly categorised when merging multiple data sources. Even if results and algorithms appear to be correct, they are unreliable if the data is inaccurate. Because the procedures will differ from dataset to dataset, there is no one definitive way to specify the precise phases in the data cleaning process. But it is essential to create a template for your data

cleaning procedure so you can be sure you are carrying it out correctly each time.



Before Outlier Detection - Box Plot

Step 1: Remove duplicate or irrelevant observations

Remove duplicate or pointless observations as well as undesirable observations from your dataset. The majority of duplicate observations will occur during data gathering.

Step 2: Filter unwanted outliers

There will frequently be isolated findings that, at first look, do not seem to fit the data you are evaluating. Removing an outlier if you have a good reason to, such as incorrect data entry, will improve the performance of the data you are working with. But occasionally, the emergence of an outlier will support a theory you are investigating.

Step 3: Handle missing data

This can’t ignore missing data because many algorithms will not accept missing values. There are a couple of ways to deal with missing data. Neither is optimal, but both can be considered.

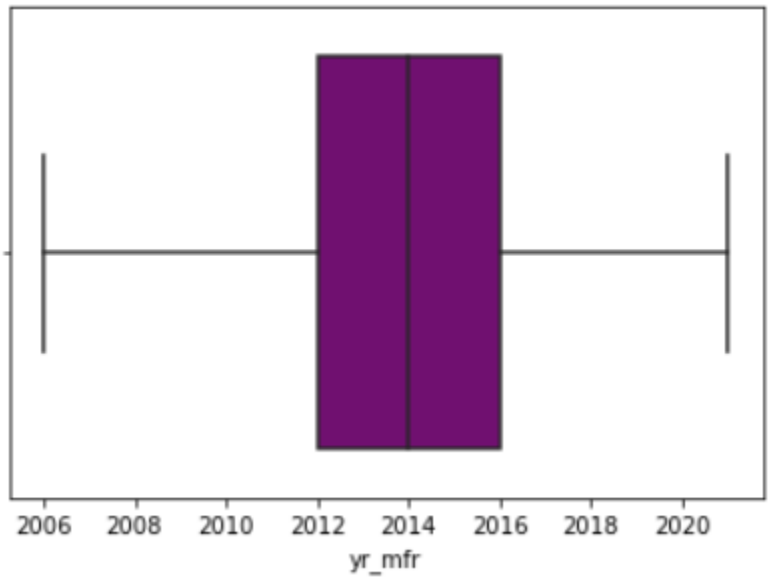
1. As a first option, you can drop observations that have missing values, but doing this will drop or lose information, so be mindful of this before you remove it.

2. As a second option, you can input missing values based on other

observations; again, there is an opportunity to lose integrity of the data because you may be operating from assumptions and not actual

observations.

3. As a third option, you might alter the way the data is used to effectively navigate null values.



After Outlier Detection - Box plot