Empirical Study of Random Forest Algorithm for Used Car Price Prediction

Jou-Ying Lee [1], Shirui Huang [2], Yunze Xie [3]
[1] Halıcıoğlu Data Science Institute
[2] Department of Cognitive Science
[3] Department of Economics
[1 2 3] University of California San Diego, CA, 92093, USA
[1] jol067@ucsd.edu; [2] shh025@ucsd.edu; [3] yux028@ucsd.edu;

1. Introduction

In the market of vehicles, it is in actuality the case that first-hand car trading makes up only a small segment of automobile trading. Buys and sells on used cars are equally or a lot more relatable and therefore more important to the general public. With this idea in mind, this study hopes to address the difficulty on determining used cars' sales price. In specific, we leverage machine learning approaches to study price predictions on used cars. This following paper will examine Random Forest Algorithm's performance on learning the particular task, and will evaluate the condition upon multiple sets of hyper-parameters with the use of Grid Search.

This study relies on the use of the dataset: Used Cars Price Prediction with 6018 observations made available on *Kaggle*, and proposes a hypothesis as follows:

Based on the 11 used car's features -- including its make, location of sale, year manufactured, kilometers driven, fuel type, transmission, owner type, mileage, engine, power, and number of seats – a valid used cars' resale price is able to be predicted.

2. Preliminaries

2.1 Dataset

Table 1 includes feature descriptions on this dataset made available by *kaggle.com*. There is a total of 11 predictors and 1 target factor (*Price*) for prediction. The dataset contains of 6018 observations in total. In specific, obtained in the dataset's original form from *Kaggle*, there are 8 categorical and 3 numerical columns.

Table 1: Dataset Description

Column	Description
Name	The brand and model of the car.
Location	The location in which the car is being sold or is available for purchase.
Year	The year or edition of the model.
Kilometers_Driven	The total kilometres driven in the car by the previous owner (s) in KM.
Fuel_Type	The type of fuel used by the car. (Petrol / Diesel / Electric / CNG / LPG)
Transmission	The type of transmission used by the car. (Automatic / Manual)
Owner_Type	Whether the ownership is Firsthand, Second hand or other.
Mileage	The standard mileage offered by the car company in kmpl or km/kg
Engine	The displacement volume of the engine in cc.
Power	The maximum power of the engine in bhp.
Seats	The number of seats in the car.
Price	The price of the used car in INR Lakhs.

2.1.1 Exploratory Data Analysis

Exploratory data analysis (EDA) over the entire dataset is performed before feature engineering in order to better understand the dataset we are working with. As Price is the target factor for prediction, we first look into its correlations with other numerical predictors.

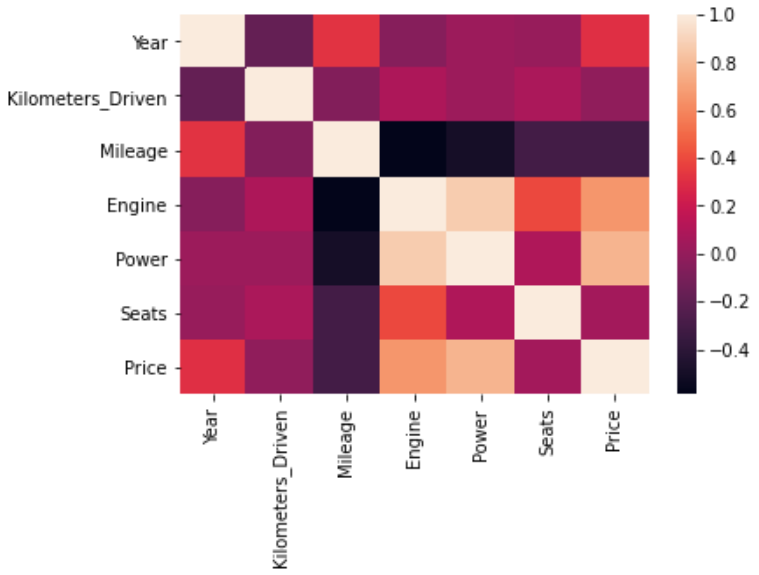

Figure 1: Heatmap of Numerical Predictors' Correlation

Figure 1 shows a correlation map among numerical predictors in the dataset. As shown in the scale to the right where the lighter the value is the greater it signifies the correlation, it seems that used cars' *Power*, *Engine* and (*manufactured*) *Year* have a relatively greater correlation with *Price*. However, it is also important to keep in mind that this correlation map only demonstrates linear

correlation among the features, and therefore does not say further of higher degree or nonlinear relations among the parameters.

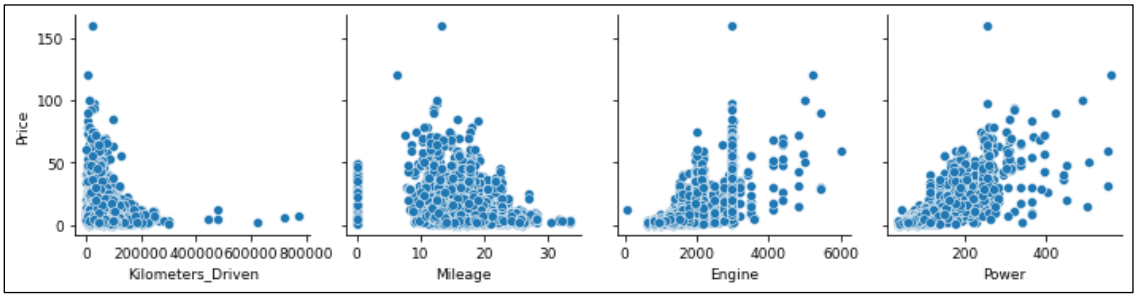

Figure 2: Pairwise Relationships of Numerical Predictors with Price

Figure 2 shows the pairwise relationships among numerical predictors in our dataset with *Price*. Each plot is a scatter plot that visualizes *Price* verses *kilometers driven*, *mileage*, *engine displacement*, and *power*.

- Overall, we have most used cars driven distance less than 400,000 kilometers, and as we might think of intuitively, our dataset does displays that the lesser the *kilometers driven*, the greater the price.
- *Mileage* however does not seem to show a clear correlation with *Price* if inspecting this scatter plot directly. Most entries scatter around a Mileage of 10-20 with prices from 0-100.
- *Engine* and *Power* on the other hand, displays a general trend of having most points at relatively lower to average qualities, but having a higher price as the quality gets greater (i.e. positive correlation between the two variables). This is reasonable because larger engine displacement usually results in higher power output and therefore priced costlier.

Figure 7.1 (please see Appendix 7.1, figure is too big to be included here) is a pie chart which displays that there is a valid variation for all the brands in our dataset. Most of them are in the mid-price range, such as Hyundai, Maruti and Honda. This therefore eliminates the problem that bias from cars' original price might influence the result.

2.1.2 Data Preprocessing

The dataset is imputed before feature engineering. Specifically, steps for imputation are carried out as follows:

1. For the column: *Mileage*, a simple mean imputation is performed since there are only 2 entries with missing values.

2. Missing data in the rest of the columns: *Engine*, *Power*, and *Seats* are done using k-Nearest Neighbor imputation. This is because each of the features contains more than 30 missing entries, and as when we think of these features intuitively – it is usually the case that these three features have some sort of relations with other properties in the dataset. With these assumptions in mind, KNN imputation is performed.

The following are steps performed for feature engineering:

1. *Name* column originally contains detailed "Makes" of a car (e.g. "Maruti Wagon R LXI CNG", "Toyota Corolla Altis GAT", etc.). For there are too many unique values in this case, *Name* is cleaned up by leaving only the "brands" of each car, and the column *Name* is therefore renamed as *Brand*. (See Figure 7.1 in Appendix 5.1 for visualization).

2. Column *Owner Type* includes 4 discrete values as that shown on the x-axis in Figure 3. As we might suspect that the less times a car is transferred, the higher the sales price might be, Figure 3 reaffirms this intuition by displaying price distribution across each owner type. Regardless of the transmission being manual or automatic, first-hand used cars are generally more pricier than second-hand, than third hand, than fourth and above.

   While manual cars' of fourth or above owner types displays a higher count of higher price than third, this is also reasonable when we think of how antique cars, especially manual antique cars are a lot of times very expensive and hard to attain in real-life. As this phenomena is explainable and close to real life, we proceed with engineering this column by encoding the values backwards – meaning encoding value "Fourth & Above" as 0, "Third" as 1, "Second" as 2 and "First" as 3. This approach is taken in order to preserve the ordinal importance in these values.

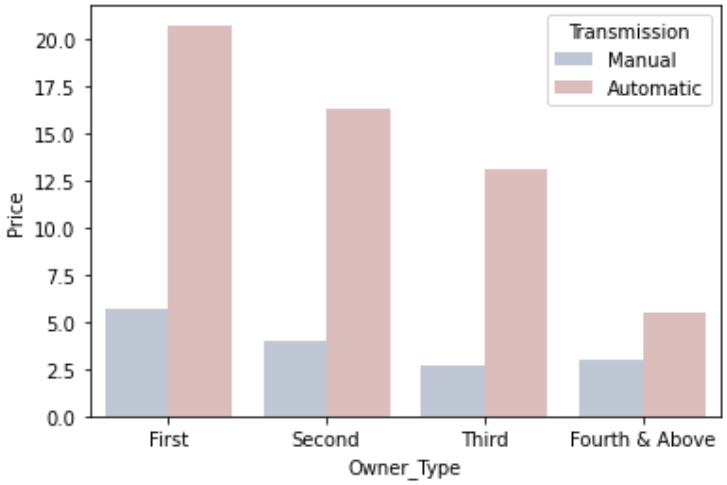

Figure 3: Used Car Prices vs Owner Type across Different Transmissions

3. For values in column *Year* are discrete, it is transformed with a MinMaxScaler to scale between feature range [0, 1].

4. All remaining numerical columns are scaled with StandardScaler, so each feature column scales to their corresponding unit variance following the Standard Normal Distribution.

5. All categorical features are one-hot encoded.

2.2 Methods

The entire dataset is split to include 70% data in training set and the remaining 30% in testing. That is 4213 listings in the training dataset and 1806 entries in the testing dataset.

2.2.1 Grid Search (with Cross Validation)

"GridSearchCV" is an approach provided in Python's package: scikit-learn. "For given values, it exhaustively considers all parameter combinations" over inputted hyper-parameter space (3.2 Tuning the hyper-parameters of an estimator). By changing the parameter "cv" in GridSearchCV's implementation, the cross-validation splitting strategy may be specified. The best model is then selected as the estimator with the parameter set that gives the best mean cross-validated score in terms of the "scoring" metric specified for optimization in GridSearchCV.

2.2.2 Random Forest

Ensemble-based method is implemented for our prediction task. Specifically, Random Forest is chosen in hope of addressing possible overfitting issues with our training. As described above there are around ~4000 observations for training, which is not a lot. Algorithm learned on limited amount of data can easily result in overfitting on unseen data. With this in mind, we decide to leverage Random Forest which introduces randomness to decrease variance of the forest estimator. Random forest "bags" for data and "samples" for features, therefore achieves a reduced variance by combining diverse trees.

Random Forest also comes with a strong property: the impurity-based feature importances. Provided as an attribute after fitting the algorithm, "the importance of a feature is computed as the (normalized) total reduction of the criterion brought by that feature" (RandomForestRegressor). Enabled by information given by this attribute, this study also looks into comparison on Random Forest performance on that fitted on the entire training set as well as that fitted on "most important features".

3. Results

3.1 Model Selection

For our implementation on GridSearchCV, we specially specified "cv" to be 3, which makes use of 3-fold cross validation with no shuffling on data splits. Each distinct estimator (model with each unique hyper-parameter set) is fitted and scored on the same three validation sets which together constitutes the 4213 entries of training data. For the "scoring" parameter is also specified to include both "r2" and "Mean Squared Error" scores, these two scores across all folds and validation sets are available in the training summary outputted by GridSearchCV. By our specification, the "best model" is determined by the one with the highest mean "r2" performance across the 3 folds.

For we have a regression task on predicting car price, our implementation leverages use of the Random Forest Regressor. In specific, the following three hyper-parameters are specially tuned to reduce tree complexity and size to avoid overfitting, and to reduce memory consumption in training. Values in brackets are the respective values tried out for training. (Please note that some of the descriptions below are from scikit-learn's API writeup).

- n_estimators: The number of trees in the forest [20, 50, 75].
- max_depth: The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure [7, 8, 9].
- min_samples_leaf: The minimum number of samples required to be at a leaf node [1, 2, 3].

For there are 3 values in each parameter to try out, there are 27 candidate parameter sets. Fitting 3 folds for each of these 27 candidates therefore totals 81 fits in our grid search process.

3.2 Model Estimation

The best model we acquired in the end is the Random Forest Regressor with 20 trees in the forest, each with a minimum samples of 2 at the leaf node, and each with a maximum depth of 9. With this parameter set we have a best r2 score over the entire training set of around 0.879, and that over the test set of around 0.878 – which is pretty close and a quite satisfying outcome. Considering that the test set is completely unseen from the algorithm during training phase, but not only did the "best" random forest did not overfit, it displayed a difference in learning score of only 0.01.

Below we show specific model learning process as well as our algorithm's performance on the entire training and testing data. Each candidate parameter set's r2 and negative mean squared error (MSE) score on each of the three data splits are recorded and made available by GridSearchCV in a comprehensive summary table. Below, we display key visualizations from this table to understand our model learning performance.

Note that the scoring metric "Negative MSE" is displayed instead of ordinary MSE. In order to correctly understand our algorithm's performance, it is important to know that the actual MSE is simply the positive version of the number displayed, i.e., the higher return values are better than lower return values.

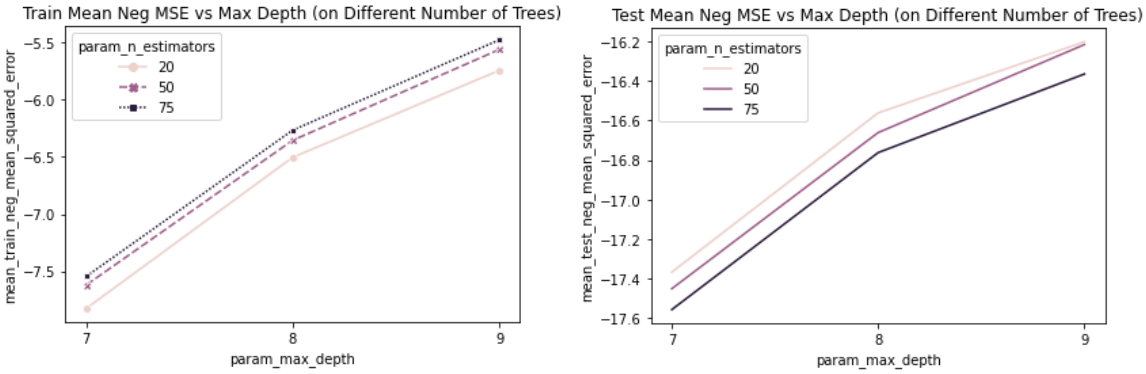

Figure 4: Training and Testing Neg MSE vs Max Depth (on Different Number of Trees)

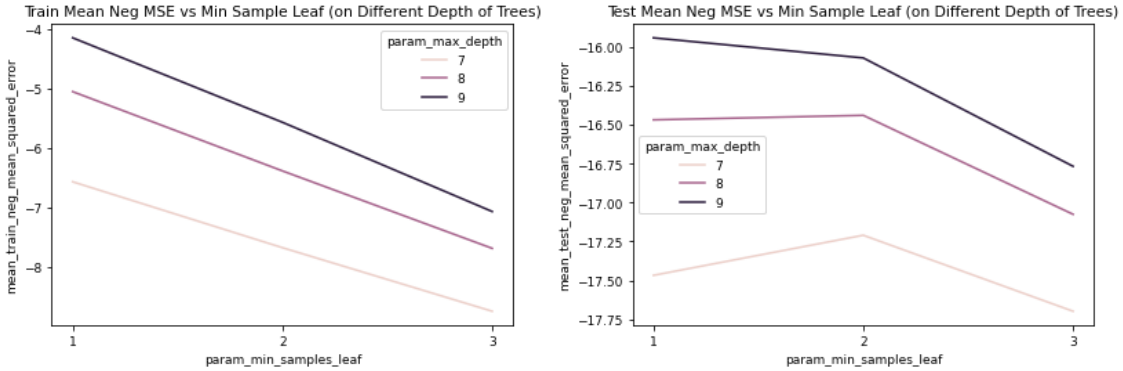

Figure 5: Training and Testing Neg MSE vs Min Sample Leaf (on Different Depth of Trees)

Graphs in Figure 4 are the mean negative MSE vs max depth of trees on different numbers of trees for both training and testing datasets. This can be interpreted that the models with trees of max depth of 9 and a total of 20 trees in the forest have the highest mean MSE on both the training and testing sets. This set of parameters therefore seems desirable to be used for best model prediction.

Graphs in Figure 5 are the mean negative MSE vs minimum number of sample leaves on different depth of trees for both training and testing datasets. This can be interpreted that the models with trees of max depth of 9 and minimum number of sample leaves of 1 generally displays to have the highest mean MSE on both the training and testing sets. Notice that though testing graph in Figure 5 shows that the negative MSE of a model with 1 minimum sample leaf is higher, it does not mean that the model with 1 minimum sample leaf is the better performer. This may be caused by the encounter of one of the three cross validation folds which happens to fit the model with 1 minimum sample leaf having a slightly better performance. However, the overall performance, which is the

mean of all the cross validation folds, is still better on the models with 2 minimum sample leaves. This set of parameters therefore also seems desirable to be used for best model prediction.

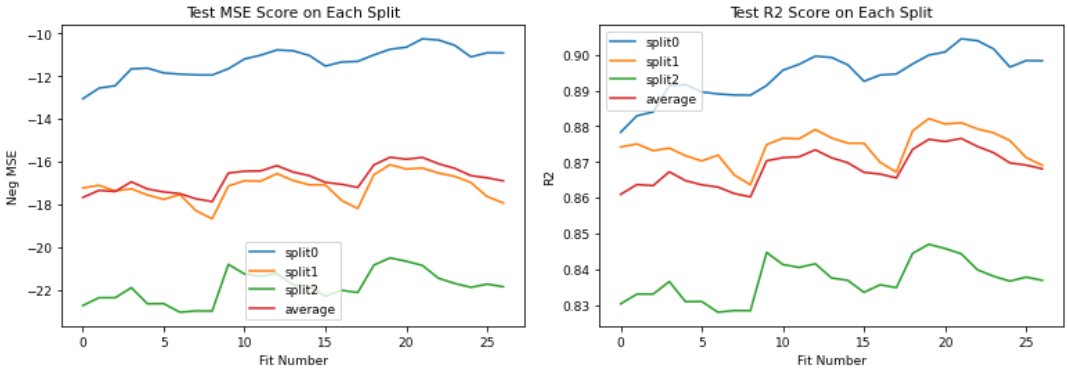

Figure 6: Testing Neg MSE and r2 Score on Each Test Set Split

Graphs in Figure 6 are the testing negative MSE score of each split performed on the test set during cross validation. The blues are the first split, orange the second and green the third. As described previously, each data split in our cross-validation process is specifically tuned to be the same. Therefore comparing the two scores across, it seems to be telling that the first data split are comparably easier to learn than the second or third as it also has a comparably higher score on both negative MSE and r2. The red lines in the middle of both graphs are the mean performance across the 3 splits. Relating back to our best score achieved, the red line does seem to average between 0.87 and 0.88 on r2.

4. Extension (Model Refitting on Most Important Features)

As an extended study into the main subject of this project's discussion, we look into a key attribute enabled by Random Forest: *Feature Importance* -- which is the impurity learned by the algorithm during training phase to determine the splitting criteria at each split.

There are 56 features in total after feature engineering. Each of their impurity-based feature importances after model fitting is obtained, and the top 20 attributes with the highest value is selected as the most important features we intend to try out for refitting.

The exact same model selection process including grid search and cross validation described above is carried out on this shrunk dataset i.e., less features. The exact same best hyper-parameter set is returned by the algorithm, with 20 trees in the forest, each with a minimum sample of 2 at the leaf node, and each with a maximum depth of 9. However, this model has a slightly lower training r2 score of 0.876 (previously 0.879), however a higher testing score of 0.88 (previously 0.878). Although this might not be a big difference, it does say a lot about this later model being more robust than the prior. It is usually not the case that testing has an even better performance than

training, but this algorithm not only generalizes to unseen data better, it even does it by more than 1% better.

5. Conclusion and Discussion

Demands and supplies for used cars are abundant in real life settings. Being able to be educated and informed of used cars' sales price is therefore just as essential as being able to tell costs of new cars. This paper makes use of a dataset which includes extensive car-related features, leverages random forest for pattern learning and examines the algorithm's prediction performances against numerous parameters and scoring criterions.

With a specially tuned model coming at a best performance of 0.878 on unseen dataset, and one further developed on smaller number of input features coming at a best 0.88 testing r2 score, we are excited to respond to our hypothesis, that:

Based on 11 used car's features -- including its make, location of sale, year manufactured, kilometers driven, fuel type, transmission, owner type, mileage, engine, power, and number of seats – a valid used cars' resale price is able to be predicted.

Used car price predictions made by algorithms in real life may more possibly be used as references for people to look at when reflecting over their ideal sales price. Therefore a nearly 0.9 r2 score performance on price prediction is believed to be a considerably convincible and credible reference before decision making.

Considering future directions or steps this study can be extended, it is believed that with more amount of data combined with more extensive features, this current ~0.9 performance can be further improved. Secondly, there are numerous other algorithms that can be tried out for this prediction task. Last but not least, it is also believed that with applications or websites that enable such "used or secondary" product resale, more people especially ones with possibly lower level of income, may be benefitted and be able to collectively participate in such activities of product trading.

6. Works Cited

"GridSearchCV", *scikit-learn*, https://scikitlearn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

"RandomForestRegressor", scikit-learn,https://scikitlearn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

"scikit-learn", *scikit-learn*, https://scikit-learn.org/stable/index.html

"Used Cars Price Prediction", *Kaggle*, https://www.kaggle.com/avikasliwal/used-cars-price-Prediction

"3.2 Tuning the hyper-parameters of an estimator", *scikit-learn*, https://scikit-learn.org/stable/modules/grid_search.html

7. Appendix

7.1 Makes of Used Cars

The following is a pie chart on the makes of used cars included in our dataset observations.

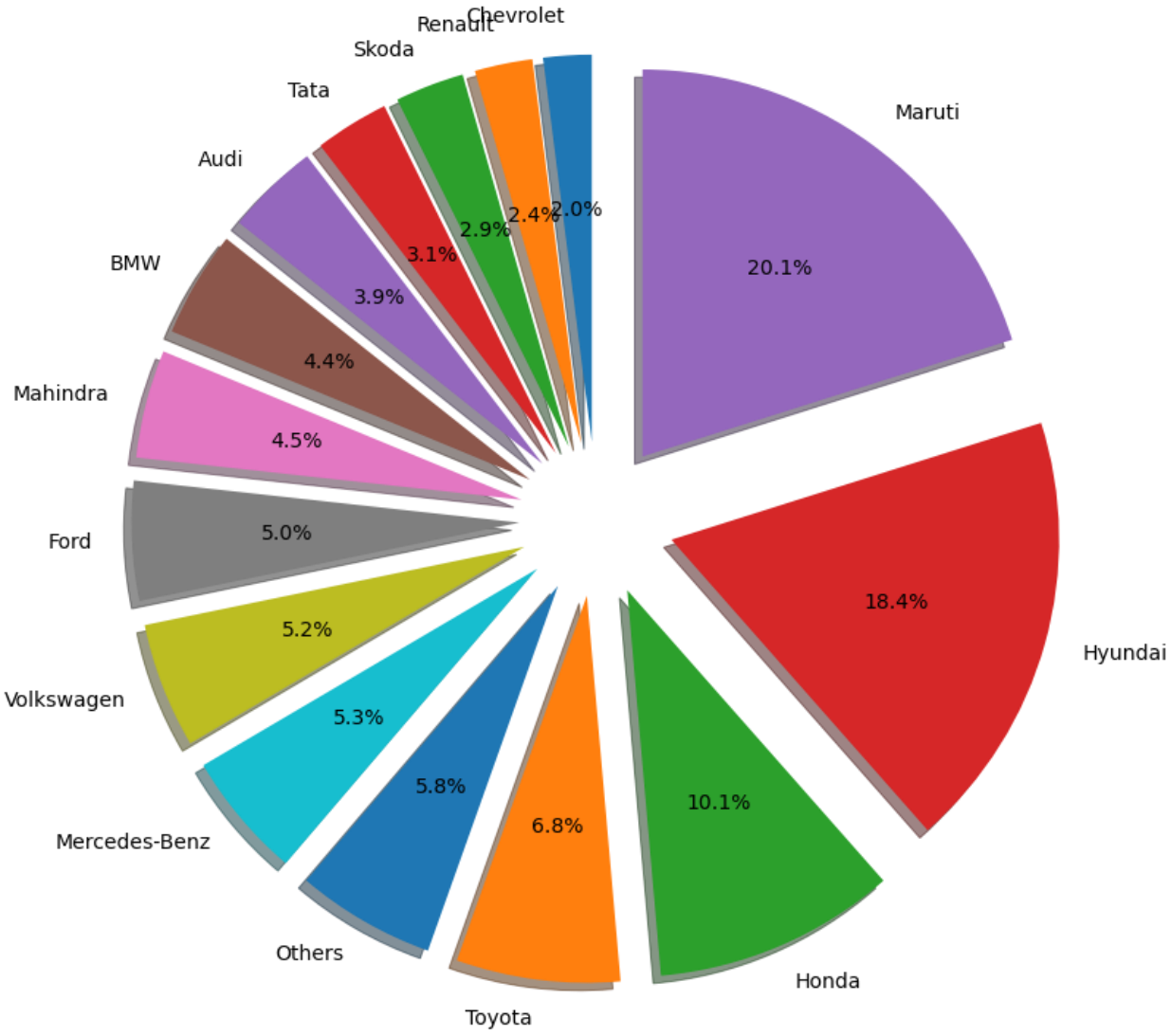

Figure 6.1: Used Cars' Makes Pie Chart