Technical Report

Machine Learning Project

University of Piraeus x NCSR Demokritos

1 Introduction

This is a technical report about a simple machine learning application developed in the context of machine learning course of MSc in AI at University of Piraeus x NCSR Demokritos. The application consists of two parts, a regression and a clustering one. They are explained in later sections. The application is deployed on Streamlit.

2 Data Collection

In this phase data was collected via two API calls at auto.dev. With the first call we collected listings of cars in the American market. With the second API call we got more detailed features of cars from the former call by using their VIN number. In total, the variables in the first dataset were 33 and 62, respectively.

Both of the calls were made once every day with the use of a cron job. As far as the first call is concerned, the same listing could be collected more than once because the listing updated one of its fields. In the end, we kept the most recent listing of each car. The total size of each dataset, with respect to the amount of rows, is 14.992.

3 Modeling

The first action made in this phase was to drop columns from both datasets that is clear that don't play any role in the price of cars or their the similarity between them.

3.1 Price Prediction Model Development

Below one can see the distribution of prices in the listings dataset. It's clear that any machine learning model trained on the whole dataset would have greater variance the closer we get to the edges, because of the big volume of cars in mid range area. Regardless of that assumption, we tested training a single model on the whole dataset and the assumption turned out to be true. More details can be found at in subsection 4.1 of the Appendix.

So we split the dataset into eight parts as seen in the figure below. That way the dataset is split in eight buckets of 12.500$ each, starting from zero. The last bucket turns out to be for cars listed at 87.500$ or more. A different model was trained for each segment, eight models in total. The size of each split can be found in subsection 4.2 of the Appendix.

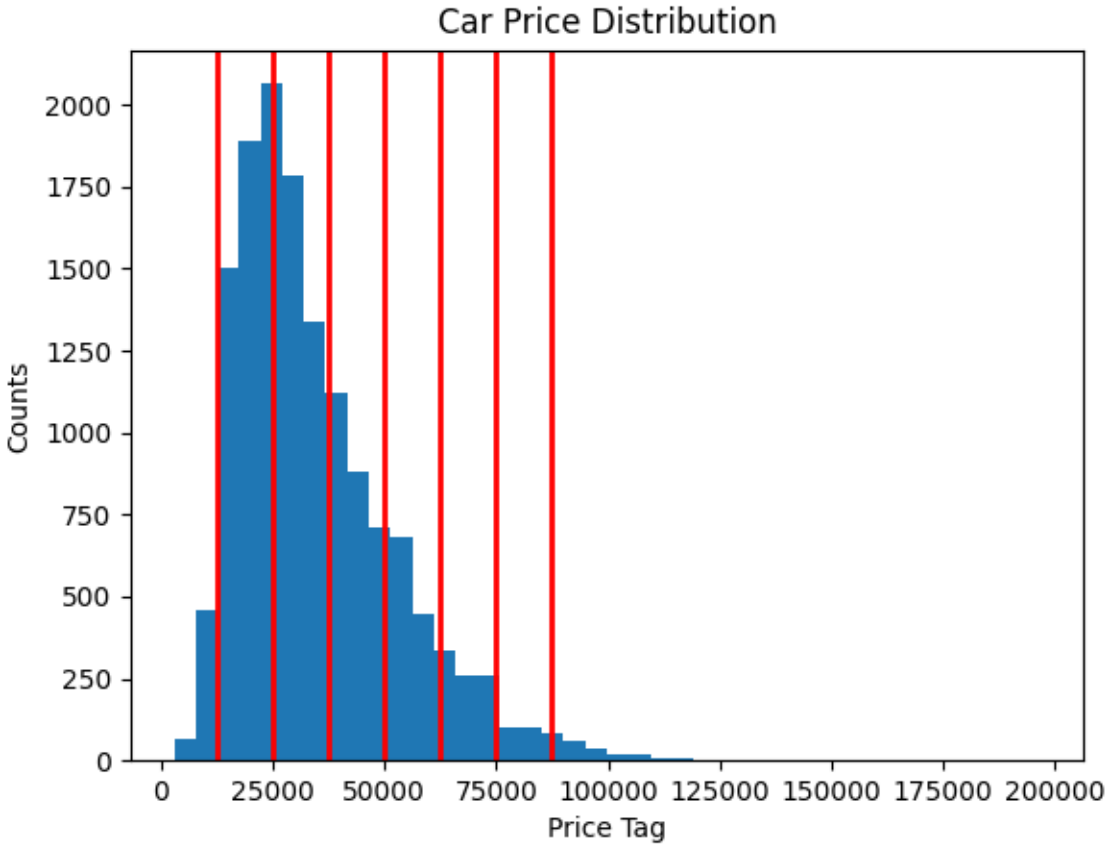

Figure 1: Distribution of car prices in the listings dataset.

The data used for the price prediction model was created by a merge between listings and features datasets. Variables used can be found in subsection 4.3 of the Appendix. Our data had a lot of missing values, that were difficult to be imputed, because of the nature of the problem, so we needed a classifier that can handle them. BaggingRegressor was used and as his base estimator we used a DecisionTreeRegressor. The same model parameters were used for every split, which can be seen in the tables below. As far as categorical features are concerned, they were all transformed to numerical with the use of LabelEncoder.

BaggingRegressor	
Parameter	**Value**
estimator	DecisionTreeRegressor
n_estimators	10
max_samples	0.4
max_features	0.4
bootstrap	False

Table 1: Model parameters for BaggingRegressor.

DecisionTreeRegressor	
Parameter	**Value**
criterion	friedman_mse
splitter	best
min_samples_leaf	15
min_samples_split	30
bootstrap	False

Table 2: Model parameters for DecisionTreeRegressor.

We used this algorithm because of its bagging feature and its ability to deal with missing data. By using a single tree, depending on its depth, it's possible that not all features will be used for prediction. So by sub setting columns and rows with a bagging algorithm it is more possible that every feature was used at least once. For each split, we performed 10-fold cross validation and calculated a 95% confidence interval based on the RMSE of each fold. The interval are shown in the table below. In the end, the models were trained on the whole split, to account for the small size of the data.

Confidence Intervals	
Split Interval	**Error Range**
split 1 interval	[1595.71, 1872.86]
split 2 interval	[2746.98, 2883.70]
split 3 interval	[2900.58, 3077.98]
split 4 interval	[3245.07, 3434.17]
split 5 interval	[3276.74, 3467.67]
split 6 interval	[3409.35, 3608.02]
split 7 interval	[3367.30, 3892.83]
split 8 interval	[10518.12, 18649.26]

Table 3: Model error ranges by split.

In test, when the user inputs features of his car the system searches how many cars exist like this and in which price segments. If cars are found in multiple segments, then more than one regressors are used to make a prediction. The percentage that each regressor contributes to the final result is decided by how many cars out of the total were found in the corresponding segment (weighted average). For example, if for a certain input of features, we get back 10 cars out of which 6 cars are from segment 2 and 4 are from segment 3 then the predicted price is calculated as:

$$Price = \frac{6}{10} \cdot split\_2\_regressor + \frac{4}{10} \cdot split\_3\_regressor$$

But the actual thing the user gets back is a range for his car value. To do that, we do the same for the upper and lower bound of the range using the intervals we computed earlier. At last, we subtract and add the lower and upper bound, respectively to the price prediction, in order to get the range and this is what the user finally sees.

3.2 Recommendation Model Development

The recommendation model corresponds to the second feature of the application, where the user selects a certain car model and gets back up to 5 similar cars. This feature was implemented using a traditional machine learning algorithm like HDBSCAN, which is an unsupervised hierarchical clustering algorithm.

HDBSCAN	
Parameter	**Value**
min_cluster_size	40
metric	euclidean
algorithm	auto
n_jobs	4
max_cluster_size	100
cluster_selection_method	leaf
leaf_size	50

Table 4: Model parameters for HDBSCAN.

The data used for the recommendation model was also created by a merge between listings and features datasets. Variables used can be found in subsection 4.4 of the Appendix. The distance between instances in the data are calculated using Euclidean distance, so every numerical feature was scaled to to zero mean and unit variance. Also, categorical columns were one hot encoded.

A problem that arises from the data is that the same car model with different characteristics can exist in multiple clusters. To deal with this, when the user selects a candidate model, the system recognizes the clusters it belongs to and computes an average representation of the model by taking the mean across all its entries in the dataset. After that, it computes cosine similarity with every other car in the clusters. The user gets back up to five cars with the highest cosine similarity to the one selected.

4 Appendix

4.1 One Model Assumption

For this test, one model was trained on the whole dataset in order to analyze its performance on the edges. Below there are three graphs showing the predicted price against the target for three different segments. The distributions between predicted and target prices are barely overlapping for 1^{st} and 3^{rd} segments, while they overlap by a large degree on the 2^{nd}. So it is clear that the model performs better in the second segment where the majority of the cars exist.

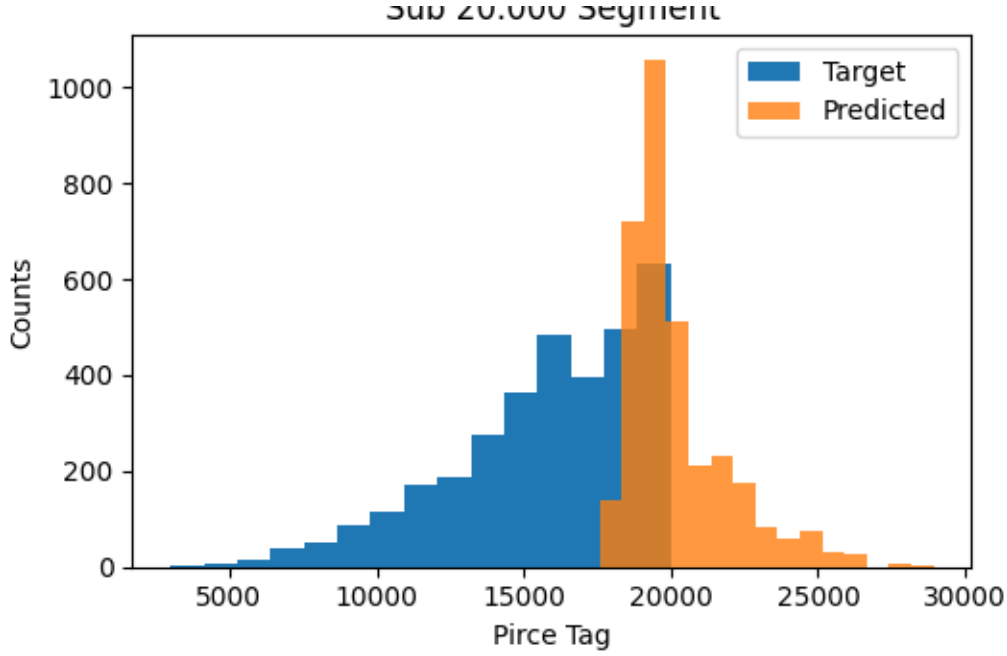

Figure 2: Predicted against Target prince for 1^{st} segment.

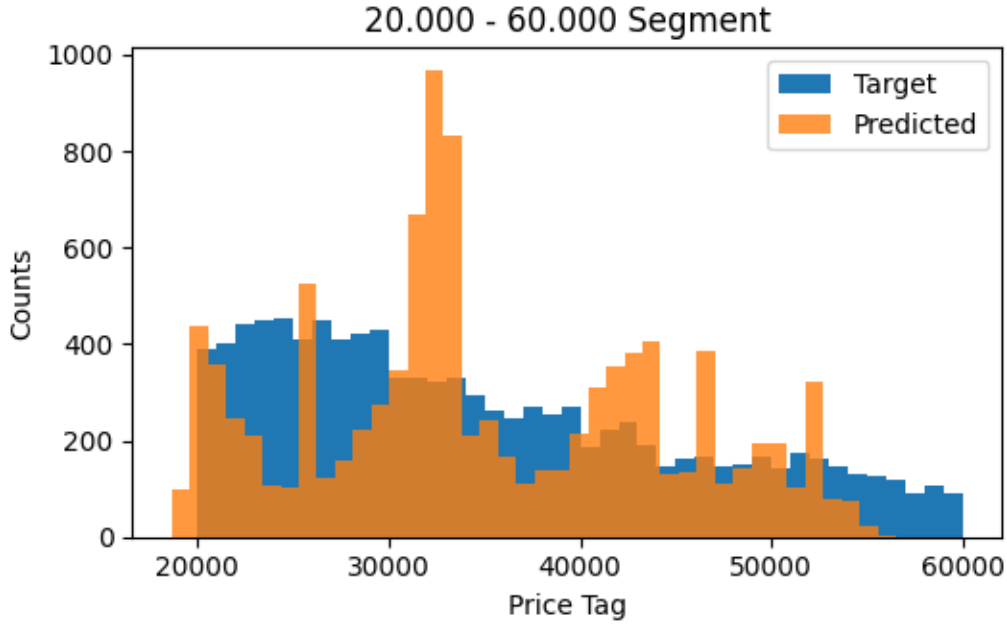

Figure 3: Predicted against Target prince for 2^{nd} segment.

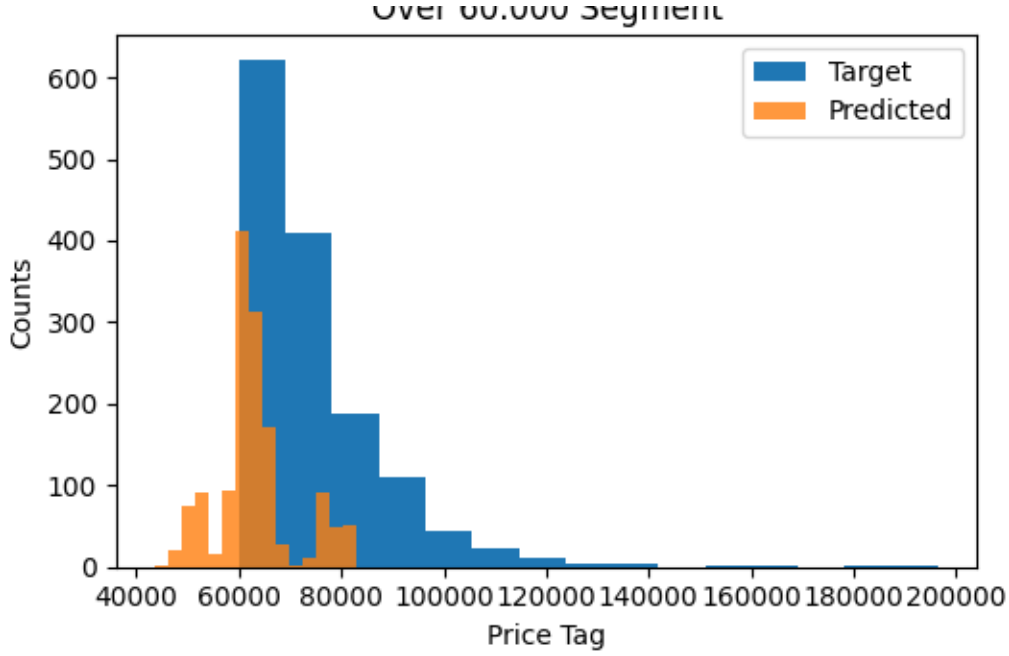

Figure 4: Predicted against Target prince for 3^{rd} segment.

4.2 Price Model Splits

Split	**Range** ($)	**Size**
1	0 - 12.500	548
2	12.500 - 25.000	4908
3	25.000 - 37.500	4348
4	37.500 - 50.000	2475
5	50.000 - 62.500	1481
6	62.500 - 75.500	730
7	75.500 - 87.500	299
8	87.500 - ∞	203

Table 5: Splits based on which the price prediction models were trained.

4.3 Price Model Variables

Variable	Descriptiion
year	Model year of the car
make	Make of the car
condition	Used or New
modelId	Car model in numerical form
bodyStyle	Car type (Sedan, SUV, etc.)
mileageUnformatted	Car mileage
size_engine	Engine displacement in litres
cylinder_engine	Number of cylinders in the engine
configuration_engine	Shape of engine (inline, V-shape, etc.)
horsepower_engine	Engine power in HP
torque_engine	Engine torque in $lb \cdot ft$
type_engine	Fuel (gas, diesel, etc.)
name_transmission	Car transmission (8A, 5M, etc.)
drivenWheels	Movement axle (FWD, RWD, etc.)
market_categories	Market Categories (Luxuty, Crossover, etc.)
highway_mpg	Fuel consumption on highway, measured in MPG
city_mpg	Fuel consumption in the city, measured in MPG
priceUnformatted	Listed car price (Target variable)

Table 6: Variables used for the price prediction model along with the target variable, priceUnformatted.

4.4 Clustering Model Variables

Variable	Descriptiion
modelId	Car model in numerical form
priceUnformatted	Listed car price
compressionRatio_engine	Compression ratio of the engine
size_engine	Engine displacement in litres
configuration_engine	Shape of engine (inline, V-shape, etc.)
horsepower_engine	Engine power in HP
torque_engine	Engine torque in $lb \cdot ft$
type_engine	Fuel (gas, diesel, etc.)
compressorType_engine	Air compressors (turbo, supercharger, etc.)
transmissionType_transmission	Transmission type (automatic, manual, etc.)
drivenWheels	Movement axle (FWD, RWD, etc.)
market_categories	Market Categories (Luxuty, Crossover, etc.)
epaClass_categories	Car size category (Compact Cars, Large Cars, etc.)
highway_mpg	Fuel consumption on highway, measured in MPG
city_mpg	Fuel consumption in the city, measured in MPG

Table 7: Variables used for the recommendation model.