Data Mining Project 2019/20
Carvana

Sara Bertoldo 588361 - Valerio Bonsignori 509061 - Federica Currao 494795

Introduction

The data under investigation come from Carvana’s resale’s cars process and describe the characteristics of each vehicle bought at auction by the dealer. The goal is to predict whether the car purchased is or not a bad buy to decrease Carvana’s loss. This last information is directly defined with a binary feature in the training dataset and therefore it’s part of the process evaluate and understand which features are the most discriminating in the choice.

Data Understanding

Data Semantics

The dataset contains 58386 entries composed by 33 features each. The dataset has categorical, numerical and date types attributes. Each entry has a unique **RefID** field used as primary key. **IsBadBuy** identifies if the purchased vehicle has been kicked. The feature of date type is **PurchDate**, **VehYear** is a related feature describing the year the car was produced, on which depend the numerical feature **VehicleAge** and the name of the auction where the car was purchased is defined in **Auction**. The specific characteristic of the vehicle are expressed in the features: **Make**, the brand of the car; **Model** and **SubModel** specify the name of the model of the car; **Trim** is the level of decorative features and **VehOdo** is the odometer value of the car. Then we have other categorical features: **Color**, **Transmission**, **Nationality**, **Size**, **WheelType** and **WheelTypeID** (which depends on **WheelType** and vice versa). **TopThreeAmericanName** is the feature that specifies if the maker is one of the top three manufacturer in the US.

Other 8 features are relative to the costs associated to each vehicle:

- **MMRAcquisitionAuctionAveragePrice,**
- **MMRAcquisitionAuctionCleanPrice,**
- **MMRAcquisitionRetailAveragePrice,**
- **MMRAcquisitonRetailCleanPrice,**
- **MMRCurrentAuctionAveragePrice,**
- **MMRCurrentAuctionCleanPrice,**
- **MMRCurrentRetailAveragePrice,**
- **MMRCurrentRetailCleanPrice**

These names are composed by different affixes whose meaning will be explained in the following lines: **MMR** stands for Manheim Market Report, a tool offering pricing and market information, **Acquisition** stands for the price at which the vehicle was purchased at auction and **Current** is the price of it at current day, **Auction** is the expected price of the vehicle at the auction meanwhile **Retail** is the expected price at which the customer will pay the vehicle. Lastly, **Average** is the price of the vehicle in normal conditions while **Clean** is the price of vehicle in above average conditions.

Finally there are some other features regarding additional info: **VNST** and **VNZIP1** describe the codes of the state and the region where the vehicle was purchased, which are highly correlated. **BYRNO** is the code of the employee that *conducted* the purchase, **VehBCost** is the true price paid from Carvana at time of purchase, **IsOnlineSale** is whether the sale was done online and **WarrantyCost** is the price of the warranty. **PRIMEUNIT** describes with a YES/NO value if the vehicle would have a higher demand with respect to a standard purchase and **AUCGART** describes how much guarantee the seller is willing to give expressed by RED/GREEN values.

All of these details where found in *Carvana_Data_Dictionary.txt* file, which lists also **FieldName**, **AcquisitionType** and **KickDate** but that are not present in our dataset.

Distribution of the variables and statistics

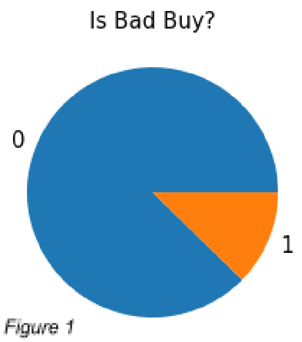


Figure 1

Since the goal of the project is to predict whether a car will be or not a bad buy, the first feature that we are going to analize, and the one that will be compared with the relevant other features is precisely **IsBadBuy.**

IsBadBuy represent whether a purchase was satisfying (value 0) or not (value 1): 88% of the entries were a good purchase meanwhile 12% were not, the classes are unbalanced (*Figure 1)*.

According to **PurchDate** all the procurements were made in 2009 and 2010. The month with least number of sales is January while the month with the most is October, but the distribution of IsBadBuy doesn't change regarding the dates.

The **auction houses** are three: Manheim with 56% of sales, Adesa with 20% and Other with 24% (*Figure 2*), we don't have further knowledge on which houses this last group includes.

Proportionally Adesa auction house has an higher percentage of bad buy (15%) compared to Manheim and the others that provide a bad buy with a probability of 11% approximately.

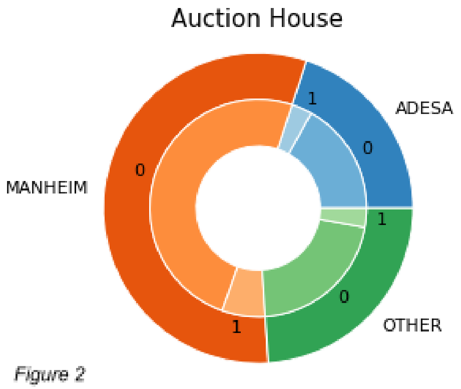


Figure 2

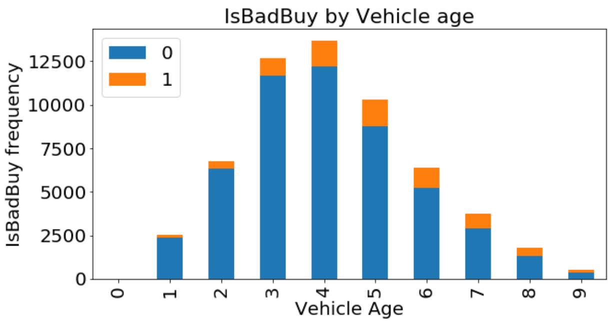


Figure 3A

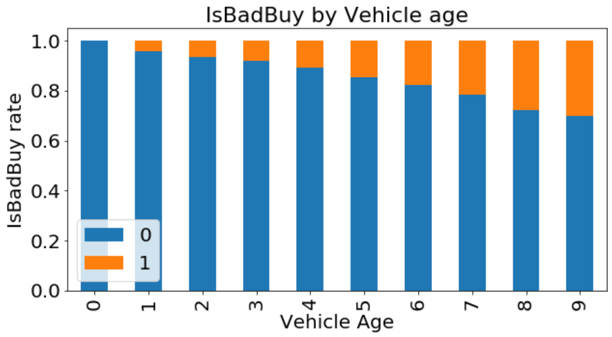


Figure 3B

It's interesting to see that **vehicle age** has a normal distribution, with median at 4 years: in percentage, as older the vehicle gets, the more it is signed as bad buy, meaning that a *younger* vehicle is a more trustable purchase for Carvana.

The most purchased vehicles have a medium **size** (42%). The most reliable vehicles are the large sized ones: 9% have been kicked; the less reliable are the sport vehicles (17% kicked).
Just the 2% of the purchases is an **online sale**, however there is not a particular evidence in this feature to bias the decision on **IsBadBuy**.

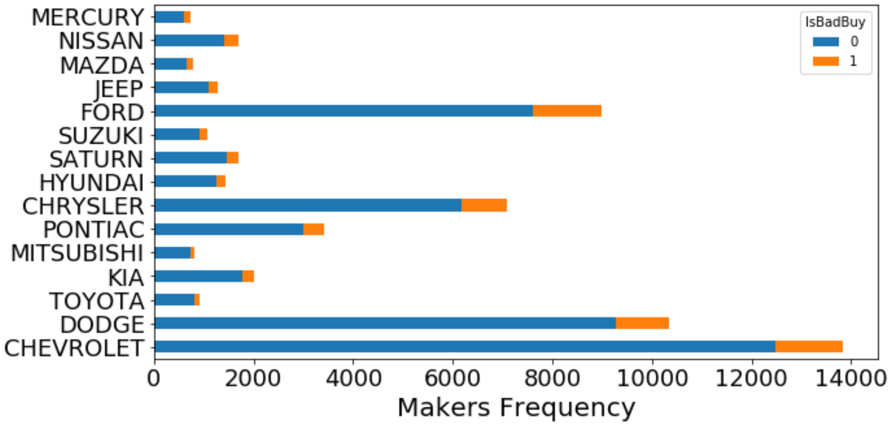


Figure 4

The most frequent **makers** are Chevrolet, Dodge, Ford and Chrysler. The most kicked vehicles belongs to Mercury, Nissan and Mazda brands: on average 16% of their cars have been kicked compared with 9% of the more trustable Chevrolet maker. Makers with less than 25 sold vehicles have not been included in the graph of *Figure 4*.

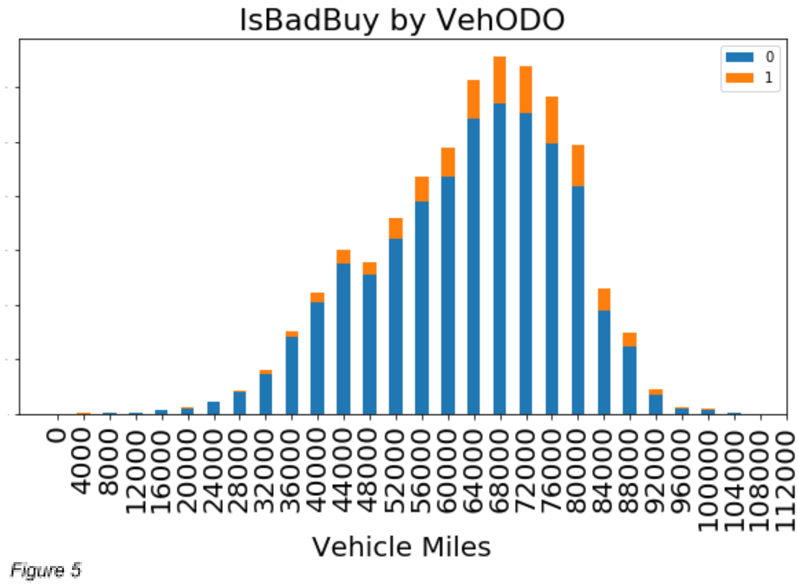


Figure 5

Regarding **VehOdo** (Vehicle Odometer), assumed to be expressed in miles, just 3 cars have more than 113k miles, each one of them have been kicked: it is never a good choice to select a car with already a long percurred distance. On both sides we have a low frequent rate, moreover, there is a drop in the shape of the curve: it correspond to the lower frequency of entries present in the dataset with a percurred distance above 99k miles.
At the center of the distribution, between 28k and 92k miles, the probability of being a Bad Buy increases proportionally to the percurred miles by the car.

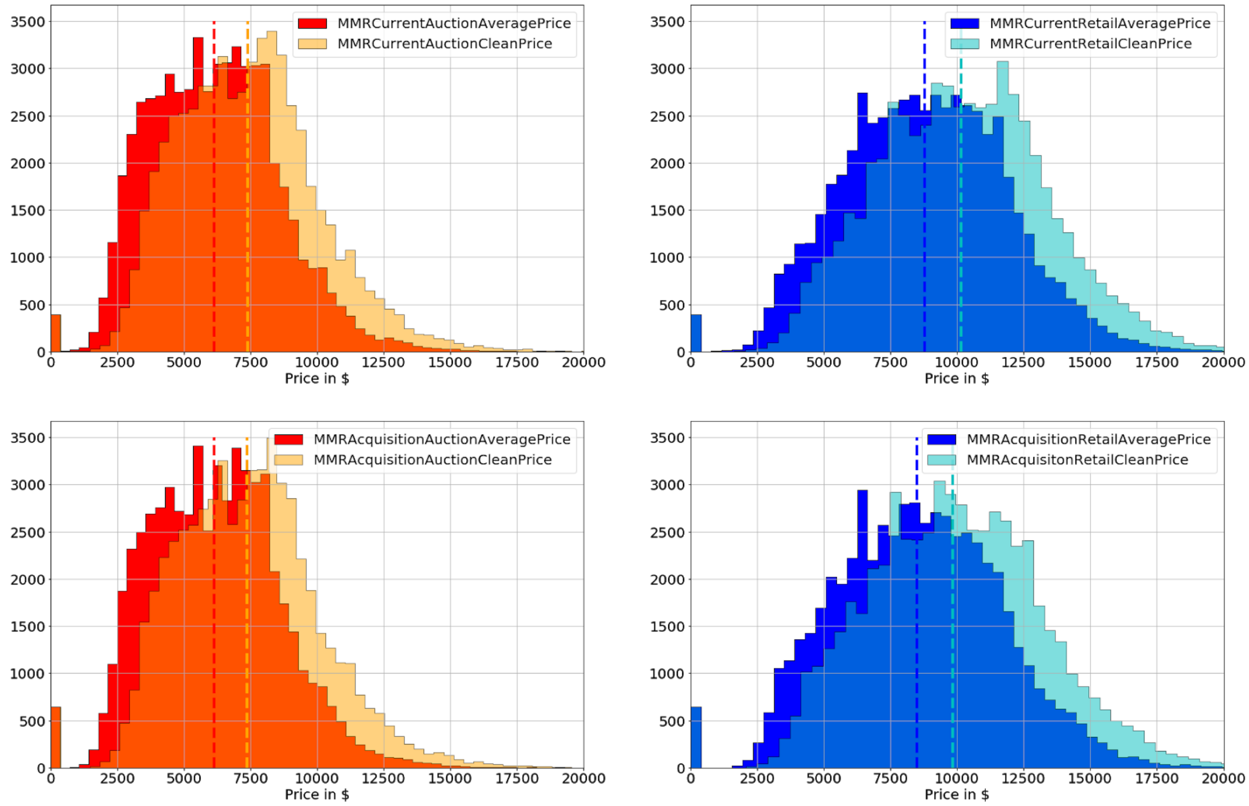


Figure 6

Looking at *Figure 6*, the frequency is shown on the y-axis while on the x-axis there is the price, that we have assumed to be in dollars. To the left, in orange, there are the **prices** spent at auction, meanwhile to the right, in blue, the prices for the client. As expected the prices for the final user are higher, there rely the main gain of Carvana. The histogram of prices for better conditions is showed with a lighter color.

In the upper part of the figure there are the *current* prices, meanwhile in the lower part there are the *acquisition* prices of the vehicle at the auction. The shapes of the curves are similar, however the mean of the *current* prices for the clients are higher than the prices spent for the vehicle by the company at the time of purchase.

In each one of the plots there are two different trends: the price if the vehicle is at average condition and the price if the vehicle is above average condition. As expected, the price for a vehicle at better condition is higher: on average a car at better condition has an higher value of 22% at auction and of 17% at retail, with respect to average condition.

The first bins at the left of every plot are outliers, they will be handled in the following part.

Vehicles bought at low price by Carvana have got the highest probability to be kicked. An higher investment is inversely proportional to the probability of getting a bad buy.

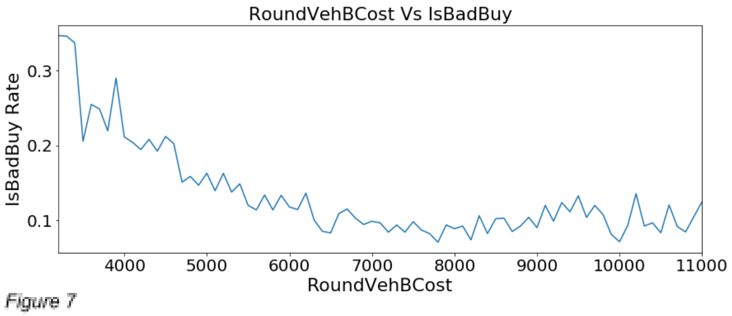


Figure 7

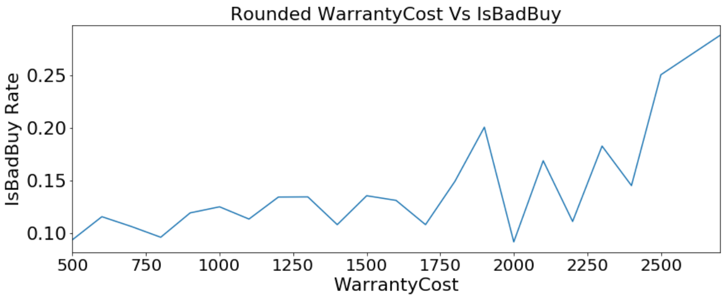


Figure 8

Reversely, the higher the **WarrantyCost**, the higher the rate of BadBuy.

No other features have shown particularly interesting distributions, characteristics or behaviours.

Data quality

Missing values and variables transformations

The feature **Transmission** has two main values: MANUAL and AUTO. There is just one entry in the dataset that has as the value “Manual” written in lower case. It has been corrected with its correspective in uppercase.

PRIMEUNIT and **AUCGUART** features are mainly null, only 4% of the dataset has a value. For this reason, these features have been excluded.

For **Size**, **Transmission**, **Nationality**, **TopThreeAmericanName** and **Color** the few missing values (7 at most), have been replaced with the mode of that feature. It has been decided to adopt this strict decision due to the low number of missing values.

2577 entries do not have any information for the **WheelType**: looking at the IsBadBuy ratio of these missing values, 71% of them are marked as kicked cars; and looking at all the kicked cars, 25% of them do not provide WheelType information; resulting in a good indicator for the target class. A new feature has been created **AugWheelInfo**, which contains the same values as WheelTypeID but in which the 2573 missing values were substituted by the value 0.

1911 entries do not have a value for the **Trim** feature: every missing value has been replaced by the mode of the Trim level according to each **Maker**. Furthermore, since Trim admits 132 different values (each of which is an optional accessory for the cars) it has been decided to reduce such multiplicity to 2: the values have been mapped into a new boolean feature stating if the Trim level is base or not, **BoolTrim**.

7 missing values were present in the **SubModel** feature; similarly to Trim**,** the lack has been solved setting such missing values as the mode for each **Maker.** Also in this case, since SubModel contains 839 different values, a simplification has been performed: just the most significant word has been taken as a simpler submodel describer, reducing the number of submodels to 32. Same reasoning has been applied to **Model**, reducing the number from 1028 to 241.

13 entries do not have any **MMR** price, since all belong to the major target class (none of them IsBadBuy) it has been decided to delete them.

232 entries of the dataset do not provide a value for all the **MMRCurrent** price features and in **MMR** price features, there were also 0 and 1 values, that represent distorted or symbolic values that we are not aware of. To solve this problem, 0 and 1 have been set to NaN and to fill all

these missing values, according to the distribution already analysed, it has been decided to use MMRAcquisition values to multiply them in a way to keep proportions among prices features and create a value for the missing entries. Since some entries did not provide information about any MMR feature, the means of SimpleSubmodel grouped by Maker have been assigned to them, remaining with only 5 missing values (if a model didn't have any price, the mean couldn't be done and the missing value couldn't be filled). These last entries have been deleted.

The eight prices features have been joined in pairs (Acquisition with Current), doing the mean of each couple and obtaining four new features (**AuctionAverage, AuctionClean, RetailAverage, RetailClean**), since these prices were very similar two by two.

Outliers

In the boxplots of *Figure 9*, we studied the distribution of the variables related to the prices. As shown, on the left plot there are a lot of outliers at high price.

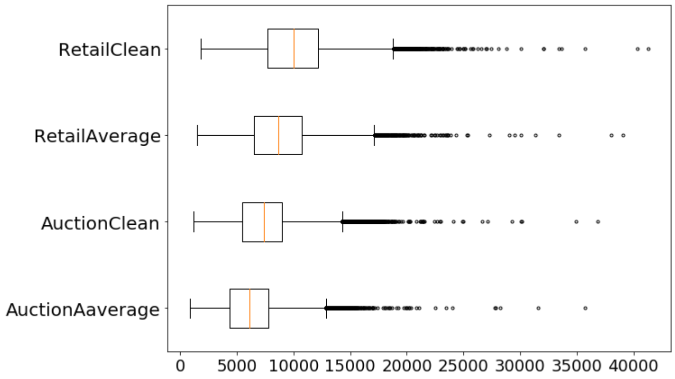


Figure 9A

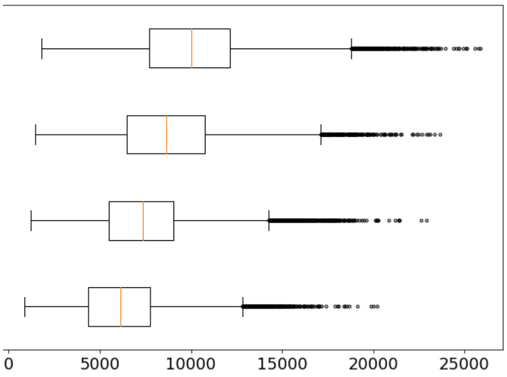


Figure 9B

Decision was taken to delete the four maximum values of each price feature, to remove the outliers with the aim of improving the normalization and the K-means algorithm.

Figure 9B shows the new distribution.

In **VehOdo** the minimum two values have been removed. **VehBCost** contained one entry with value 1, that has been replaced with the mean of that model; the values greater than 14000 have been removed.

Pairwise correlations and eventual elimination of redundant variables

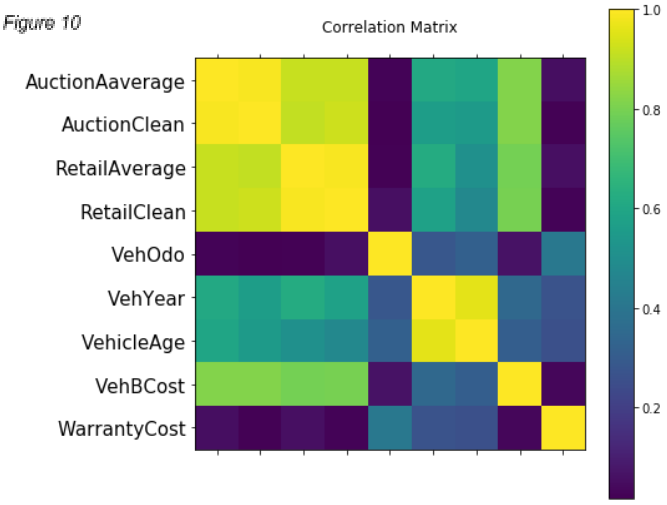


Figure 10

Categorical features have been excluded from the correlation analysis, therefore obtaining the correlation matrix among numerical attributes shown in *Figure 10.*

The correlation among every price feature is high, while a slightly lower correlation is present also with **VehBCost**.

VehicleAge and **VehYear** are negatively correlated but in the figure they have been plotted in absolute value, showing an high correlation. They also have a low correlation with the modified prices features.

VehicleAge is derived from **VehYear**, it is redundant to keep both of them, therefore **VehYear** will not be considered anymore.

WheelTypeID and **WheelType** provide the same information, so **WheelTypeID** will be held.

Clustering

Clustering Analysis by K-means

Choice of attributes, distance function and K value

In order to perform K-means clustering analysis the numerical features **AuctionAverage, AuctionClean, RetailAverage, RetailClean**, **VehBCost**, **VehOdo** and **WarrantyCost** have been log scaled. These features and **VehicleAge** have been taken into account**.** Every attribute have been then normalized in range [0,1] subtracting the minimum and dividing by the maximum.

The distance function used is the Euclidean distance because is the most reasonable with the K-means algorithm.

K-means algorithm has run with the previously listed features. The best sse scores, out of 10 initializations, for each K ranging from 2 to 30 are shown in *Figure 11*, upper plot. The lower one represent the difference between each consequent sse.

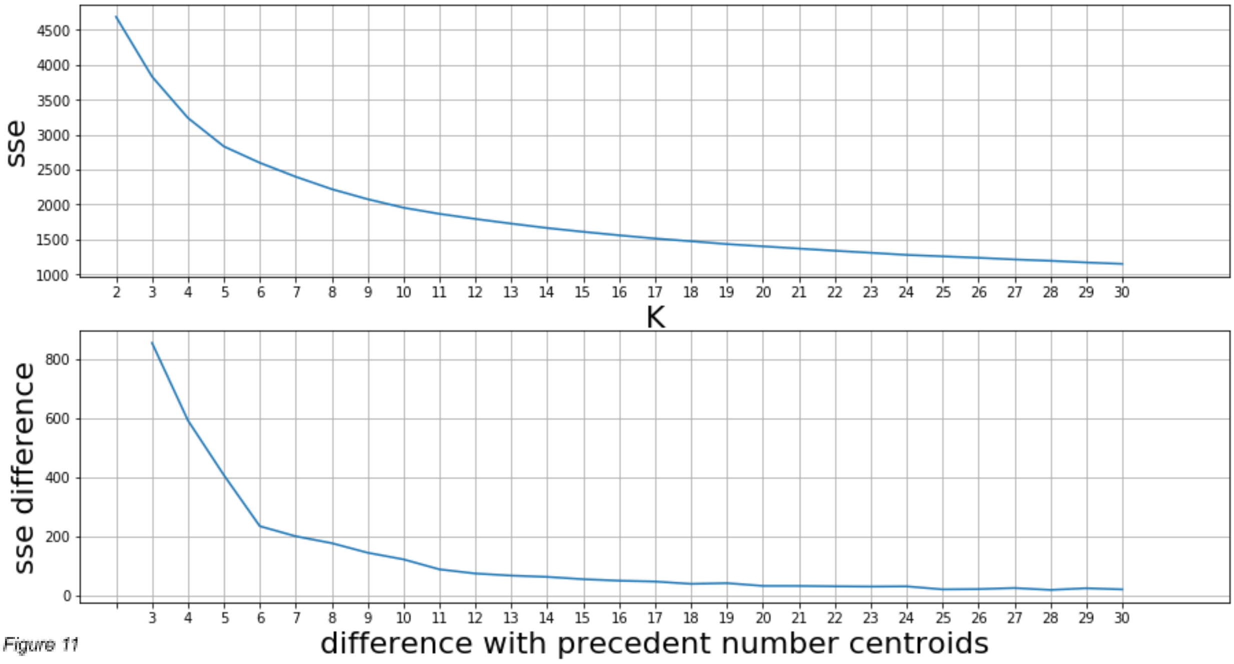


Figure 11

Willing to obtain the lowest K significant value, the k values greater than 30 have not been taken into account. From the lower plot we can notice two interesting points: 6 and 11. At K equal to 11, the increase of the value of K will cause a very little difference of the sse, meanwhile a decrease of the value of K will increase the sse.

We choose 11 as K value because it gives us the higher purity on each cluster: the clusters are more polarized, meaning that majority of clusters have either percentage greater than 14% or lower than 10% of Bad Buy. In some clusters there are mostly reliable cars and in others there are on average more kicked cars than the whole dataset.

The study of the silhouette does not help in the K identification, that is because data overlaid and clusters are very close amongst them: entries are very similar and very close to each other thus they are not well separated.

K equals to 11 gives the best level of purity within the clusters, nevertheless, by the reason of the data available, K-means does not work in an optimal way, since the shape of the clusters is not globular.

Traits of the obtained clusters

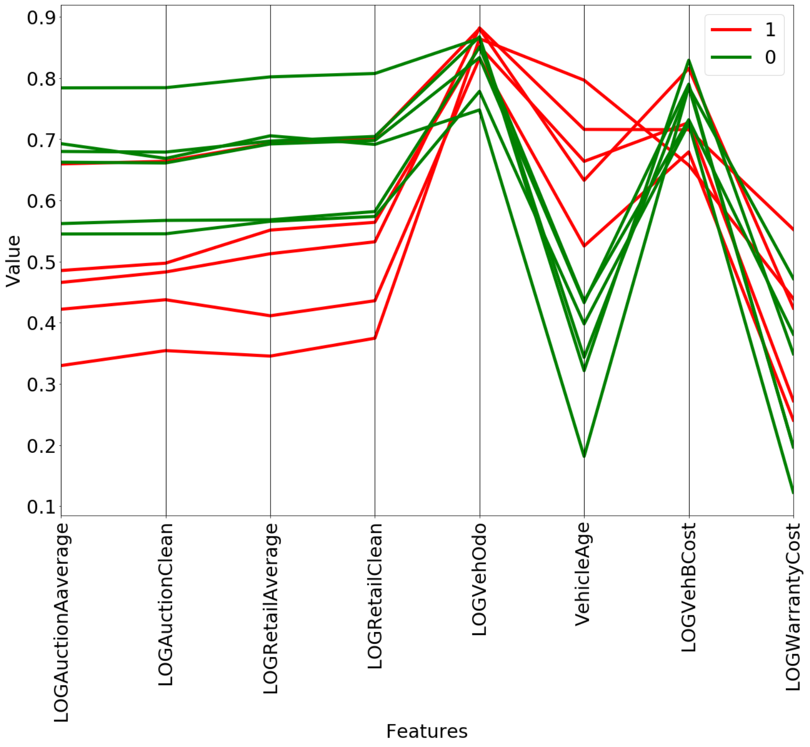


Figure 12

Red lines are used to represent centroids of clusters with a percentage of bad buy greater than the average of the dataset (12%), while green ones represent lower percentages.

As shown in Figure 12, the majority of the BadBuy centroids are characterized by low prices and high values of VehicleAge and VehOdo.

The values of LOGVehOdo and LOGVehBCost are not sparse.

The centroids reflect the same trend of the dataset as regards VehicleAge and VehOdo: the greater these values, the higher the BadBuy ratio.

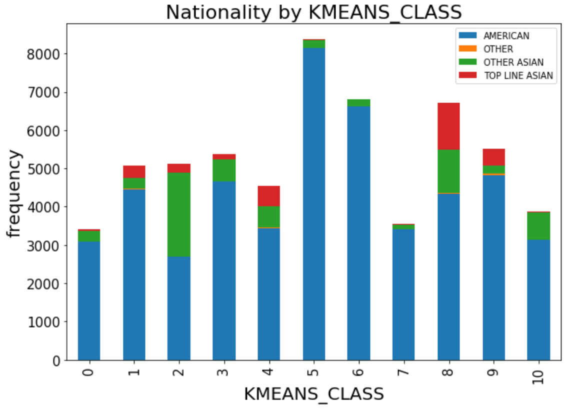


Figure 13

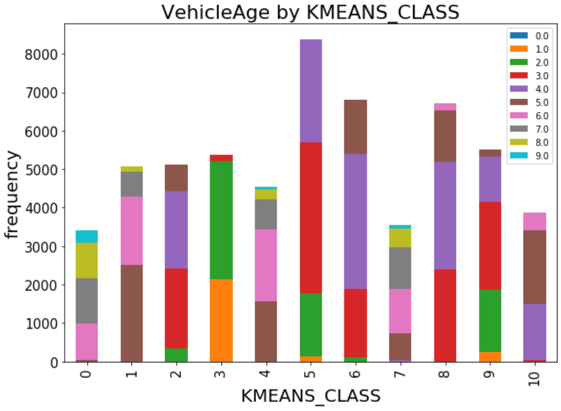


Figure 14

Figure 13 shows the predominance of the American nationality, looking deeper clusters 2, 8, and 10 contain the higher number of Asian vehicles.
As Figure 14 shows, clusters 3, 5, 9 contain the youngest cars while clusters 0, 1, 4, 7 contain the older ones.

Clustering Analysis by DBSCAN

Choice of attributes and distance function

To perform DBSCAN the used distance function is **Euclidean**; for this reason the same numerical features of K-means have been considered and treated doing the log scale and the normalization as well. In order to speed up the performance and reduce the memory used, a sampling have been performed to reduce the number of entries creating a training subset with 70% of data.

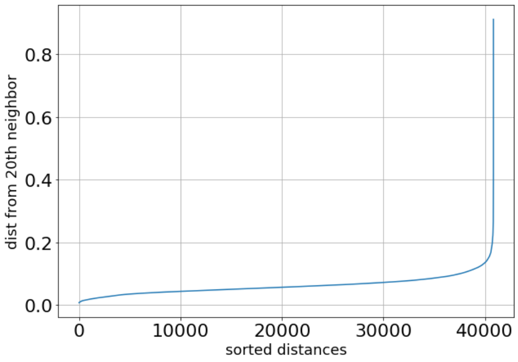


Figure 15

The distance plot obtained is shown in *Figure 15* that helps in finding the right value of epsilon.

To identify the best clustering any combination of MinPts and eps has been calculated among these values: minPts = [5, 10, 15, 20], eps = [0.07, 0.08, 0.09] checking for the higher silhouette score, not counting the noise points.
The higher silhouette obtained is **-0.005** with **MinPts 20** and **epsilon 0.09** resulting in 12 clusters.
Nevertheless, we obtained the same number of clusters as K-means algorithm.
The performances of a clustering based on density is not good since our dataset do not contain low density regions.

Characterization and interpretation of the obtained clusters

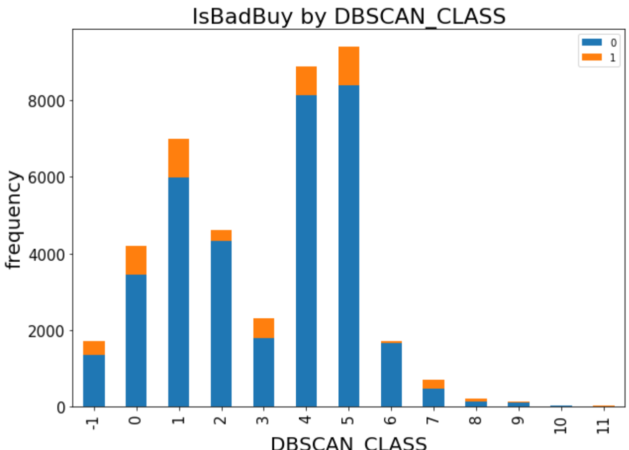


Figure 16

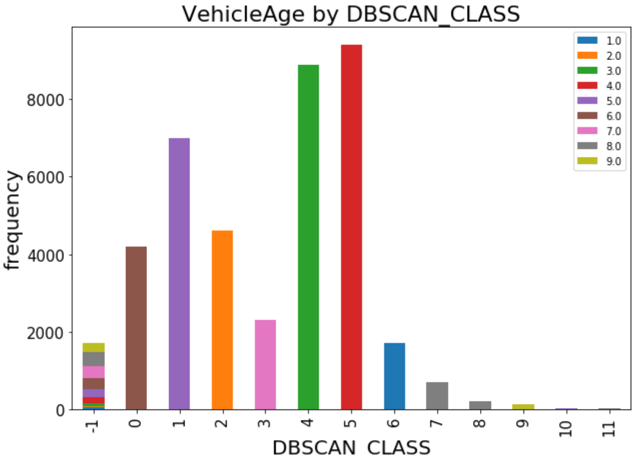


Figure 17

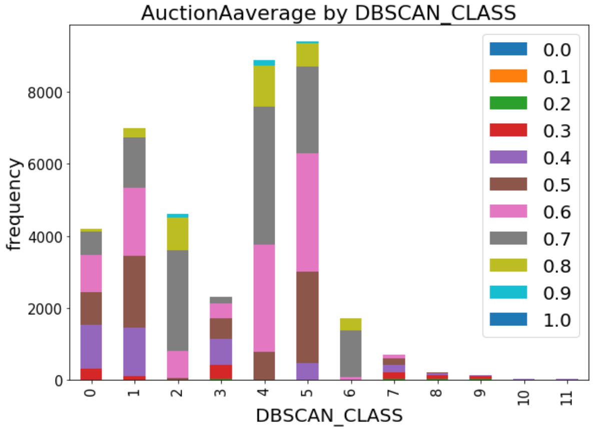


Figure 18

Figure 16 shows the distribution of BadBuy cars in the clusters: the clusters with the lower percentage of BadBuy cars are 2 and 6. The class labeled with -1 are the points considered noise.

DBSCAN perfectly splitted the cars according to their age (*Figure 17*).

The cluster 2, 4 and 5, 6 are characterized by vehicles with an higher average price at auction, meanwhile the lower priced vehicle at are in the clusters 3 (*Figure 18*).

Clustering Analysis by Hierarchical

Choice of attributes and distance function

The features have been chosen and treated in the same way as for the K-means algorithm, considering also some categorical features transformed in numerical form.

In order to speed up the performance and reduce the memory used, a sampling have been performed to reduce the number of entries creating a training subset with 80% of data.

To find the best parameters combination, we performed a grid search with different methods (complete, single, average, ward), metrics (cosine, euclidean) and threshold values.

The results of this search are plotted in *Figure 19.*

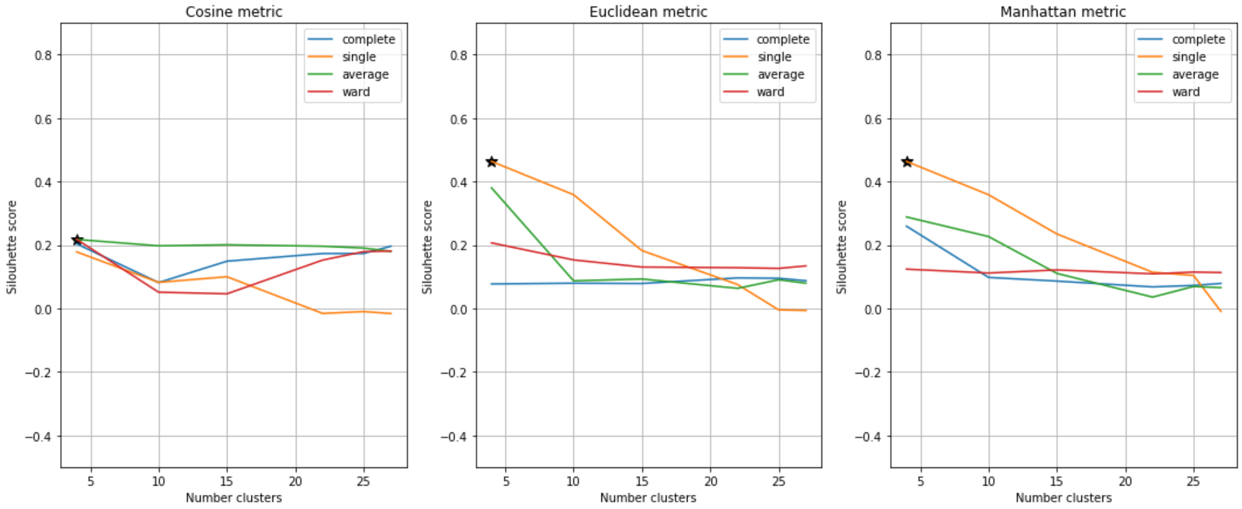


Figure 19A *Figure 19B* *Figure 19C*

From the grid search we found that the best clustering according to the higher silhouette score (0.46) uses **Euclidean** metric, with **single** method and a **threshold 0.310**.

The hierarchical clustering with the aforementioned values resulted in the following dendrogram.

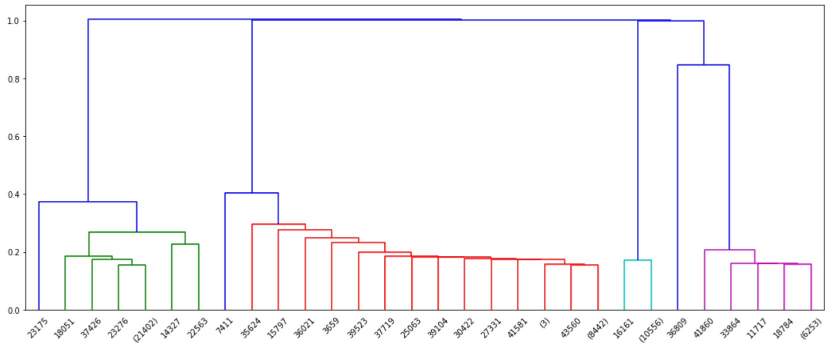


Figure 20

With the best combination for the Manhattan metric, the dendrogram is exactly the same, while the second best combination for the Euclidean metric (average method, threshold 0.791) results in the dendrogram of *Figure 21.*

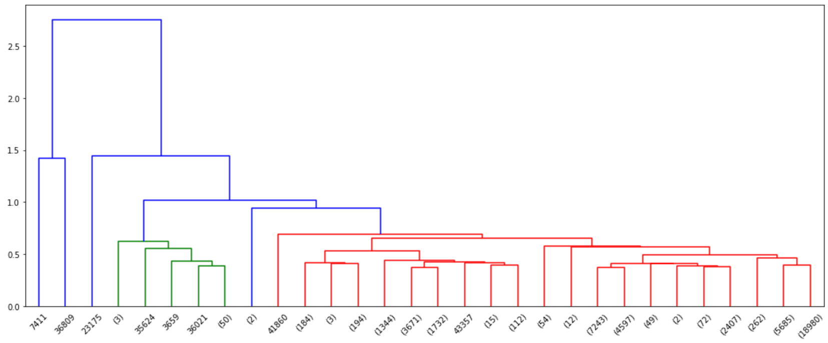


Figure 21

Final evaluation of the best clustering approach and comparison of the clustering obtained

Comparing the three clustering algorithm, according to the different silhouettes scores, the best result is obtained with K-Means.

Clustering algorithm	**Silhouettes**
K-Means	0.656
DBSCAN	-0.005
Hierarchical	0.463

Table 1

Associations Rules

In order to run the algorithm *Apriori*, there have been used the categorical features **SimpleSubModel, Make, SimpleModel, BoolTrim, Nationality, Size, VNST, AugWheelInfo, Color** together with the numerical features **VehOdo, VehBCost, WarrantyCost,**

MMRCurrentAuctionAveragePrice, MMRCurrentAuctionCleanPrice, MMRCurrentRetailAveragePrice, MMRCurrentRetailCleanPrice, MMRAcquisitionAuctionAveragePrice, MMRAcquisitionAuctionCleanPrice, MMRAcquisitionRetailAveragePrice, MMRAcquisitonRetailCleanPrice and **VehicleAge.**
The numerical features have been mapped to a reduced bin space of 7 different bins, and subsequently they have been transformed in strings.
The bounds of the bins have been decided thanks to k-means: the algorithm have run on every numerical feature (k-means run on an unidimensional feature), and each bin represented the bounds of the clusters.

Frequent pattern extractions

with different values of support count and different types of itemset extracted.

Maximal itemsets		
Min support	**Support percentages**	**Itemset**
60%	73.48% 70.06% 66.7%	‘AMERICAN Nationality', '0 IsBadBuy' 'NOBASE BoolTrim', '0 IsBadBuy' 'NOBASE BoolTrim', 'AMERICAN Nationality'
70%	73.48% 70.06%	'AMERICAN Nationality', '0 IsBadBuy' ‘NOBASE BoolTrim', '0 IsBadBuy'
80%	87.62% 83.62%	'0 IsBadBuy' 'AMERICAN Nationality'

Table 2A

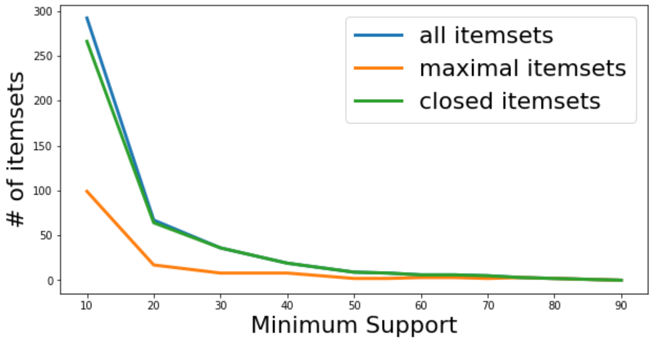
Frequent itemsets		
Min support	**Support percentages**	**Itemset**
60%	87.68% 83.62% 79.84% 73.48% 70.06% 66.7%	'0 IsBadBuy' 'AMERICAN Nationality' 'NOBASE BoolTrim' ‘'AMERICAN Nationality', '0 IsBadBuy' 'NOBASE BoolTrim', '0 IsBadBuy' 'NOBASE BoolTrim', 'AMERICAN Nationality'
70%	87.68% 83.62% 79.84% 73.48% 70.06%	'0 IsBadBuy' 'AMERICAN Nationality' 'NOBASE BoolTrim' 'AMERICAN Nationality', '0 IsBadBuy' 'NOBASE BoolTrim', '0 IsBadBuy'
80%	87.68% 83.62%	'0 IsBadBuy' 'AMERICAN Nationality'

Table 2B

The closed itemsets are exactly equal to the frequent ones, so the table is not shown.

Looking at the itemsets, the most frequent cars are a good investment and are from America. Fewer less cars have a trim level different than the basic one, and are a good purchase; even fewer cars are united by the fact that are American and are equipped with optionals features.

Number of patterns with respect to the minimum support count



It has been confirmed that the number of itemset and consequently of the rules is inversely proportional to the minimum support.

Figure 22

Association rules extraction with different values of confidence

For simplicity, it has been decided to list only few rules for every considered confidence level.

Rules			
Confidence	**Support**	**Lift**	**Rule**
70%	0.85 %	5.74	('0.0 AugWheelInfo', 'MEDIUM Size', 'SEDAN SimpleSubModel', 'NOBASE BoolTrim') -> 1 IsBadBuy
70%	1.3 %	5.80	('0.0 AugWheelInfo', 'MEDIUM Size') -> 1 IsBadBuy
76%	16.42%	0.91	('2.0 AugWheelInfo', 'SEDAN SimpleSubModel', 'NOBASE BoolTrim', '0 IsBadBuy') -> AMERICAN Nationality
80%	24.34 %	1.01	'MEDIUM Size', 'SEDAN SimpleSubModel', 'NOBASE BoolTrim') -> 0 IsBadBuy
88%	18.96%	1.01	('MEDIUM Size', 'SEDAN SimpleSubModel', 'NOBASE BoolTrim', 'AMERICAN Nationality') -> 0 IsBadBuy
88%	58.7 %	1.00	('NOBASE BoolTrim', 'AMERICAN Nationality') -> 0 IsBadBuy
90%	21.4 %	1.03	('CHEVROLET Make', 'AMERICAN Nationality') -> 0 IsBadBuy
90%	15.61 %	1.04	('[7496.0, 8953.0)_MMRAcquisitionAuctionCleanPrice', 'NOBASE BoolTrim') -> 0 IsBadBuy
90%	9.29%	1.56	('[782, 1103)_WarrantyCost', 'MEDIUM Size', 'NOBASE BoolTrim', 'AMERICAN Nationality', '0 IsBadBuy') -> SEDAN SimpleSubModel
100%	21.4 %	1.20	('CHEVROLET Make', '0 IsBadBuy') -> AMERICAN

			Nationality

Table 3

“0 IsBadBuy” is a consequence in the majority of the rules, due to the high presence of this value. “1 IsBadBuy” is found only with lower values of confidence (70%), these rules have a very low support and frequently contain “0.0 AugWheelInfo” as precondition. Only two of the considered rules contain prices, meaning that there is an high variety of prices between cars, but is remarkable that these two rules have 90% of confidence. The model SEDAN is a very common value in the rules and so are American nationality and NOBASE BoolTrim.

Changes the number of rules w.r.t. the min_conf parameter

As shown in *Figure 23* that shows a range from 50 to 90, the number of rules decreases proportionally with the value of confidence. For this phase the support was fixed at 20.

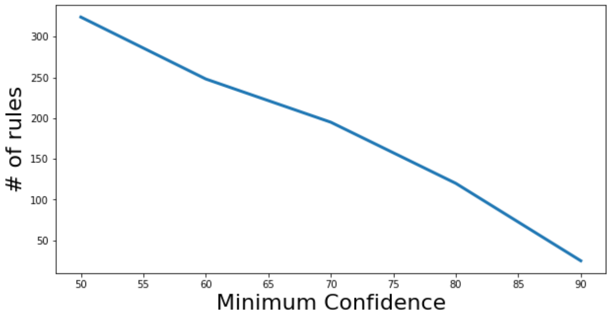


Figure 23

Histogram of rules' confidence and lift

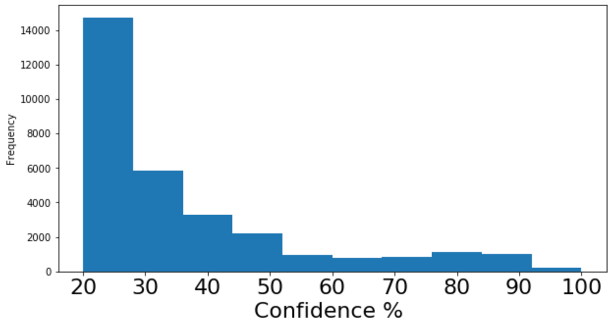


Figure 24

Starting from 20%, the frequency of each bin is halved adding 10% at every step, until reaching 60%. Then, the frequency remains roughly unchanged to be definitely reduced after 90% of confidence.

The majority of the rules have 1 as value, less than the half have 2 and drastically less rules have an higher value of lift.

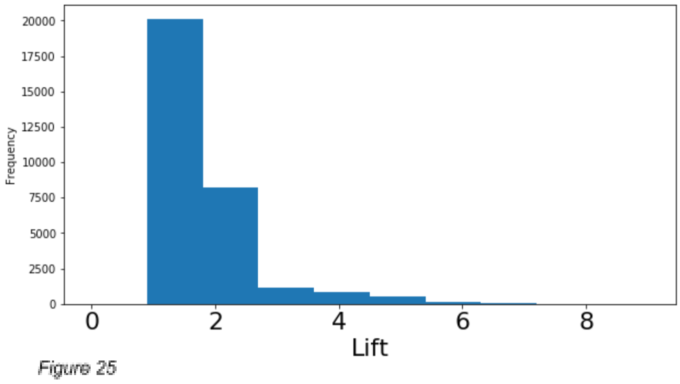


Figure 25

Replace missing values and evaluate the accuracy

First of all, the original dataset was taken into account, deleting all the entries with at least a missing value and splitting it into two parts: training set (70%) and validation set (30%).
The numerical features have been divided in bins using k-means algorithm to understand the bounds of each bin, and then transformed in strings.
The rules of the training set with at least 20% confidence have then been extracted.
The rules have been used to predict the values in the validation set and afterwards the confusion matrix have been calculated between the predicted and the real values.
This process was done on the features with the higher number and not trivial missing values: **BoolTrim**, the synthetic features that discriminate between basic and non basic trim levels (*Figure 26A)*; **SimpleSubModel** (*Figure 26B*), and **WheelTypeID** *(Figure 26C).*

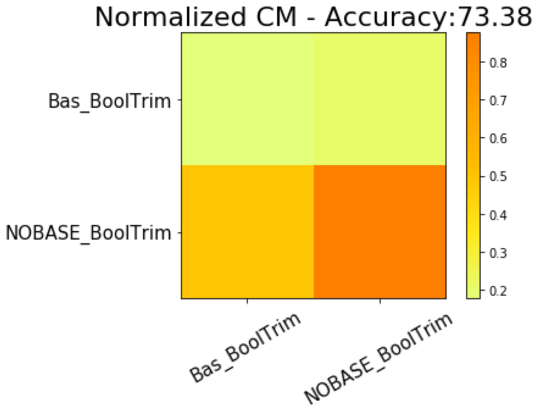


Figure 26A

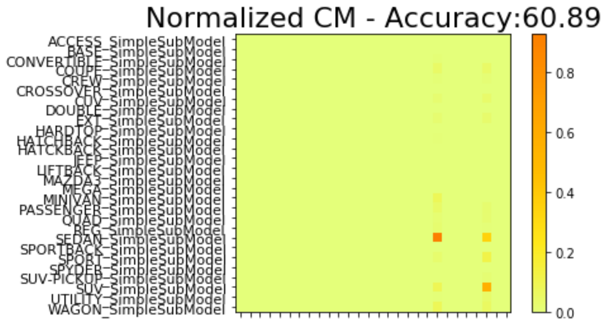


Figure 26B

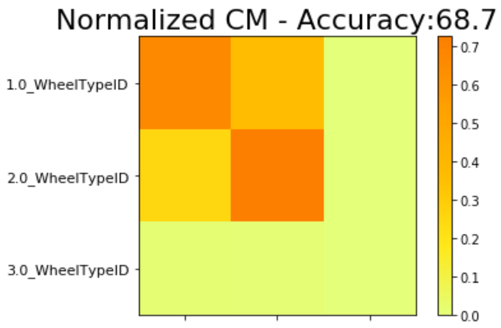
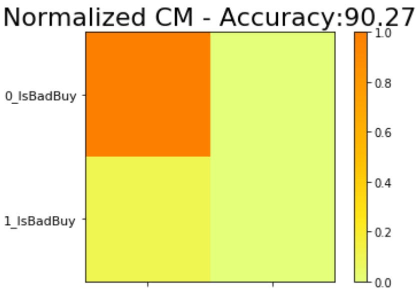


Figure 26C

Even if the accuracy values of the rules to predict the missing values is higher than a random choice, the confusion matrix shows clearly how the rules take into account just few values (the most common ones) rather than catching the latent rules behind the data that could be useful to replace the missing values.
All the rules for **SimpleSubModel** have as consequent just *SEDAN* or *SUV*. The use of those rules is not particularly accurate and detailed since they are just predicting the 2 most common simplified submodels.
The feature **Color** also contains many missing values, but independently on the premise, all the rules that have as consequences **Color** always predict the color *SILVER*: using the rules to replace would be the same as taking the mode of the feature.

Predict IsBadBuy and evaluate the accuracy



As for the missing values, using the rules to predict does not really appear to be very significative. Due to the unbalanced **IsBadBuy** feature, many rules that have as consequence BadBuy = 0 have been extracted. Therefore many entries match the premises of rules and the all of them is predicted to be a good investment.

Figure 27

Classification

In order to obtain new *significant* features to improve the performances of the classification, new sintentic features of comparison between car prices have been analyzed: ratios between prices (e. g. MMRAcquisitionAuctionAveragePrice / MMRAcquisitionAuctionCleanPrice) and differences (e. g. MMRAcquisitionAuctionAveragePrice - MMRAcquisitionAuctionCleanPrice).

These features were added to the dataframe, subsequently sampled with training size 70%.
To understand the most interesting synthetic features, those have been fed to a forest of trees, builded with a grid search. The grid search has been executed using gini criterion, min splits 2, max depth = [6,7,8,9,10] and min samples = [1,20,21,22,23]. From the grid search 25 different trees have been created and the features that appeared with the higher importance the most of the times have been included for the final classification task (*Figure 28*).

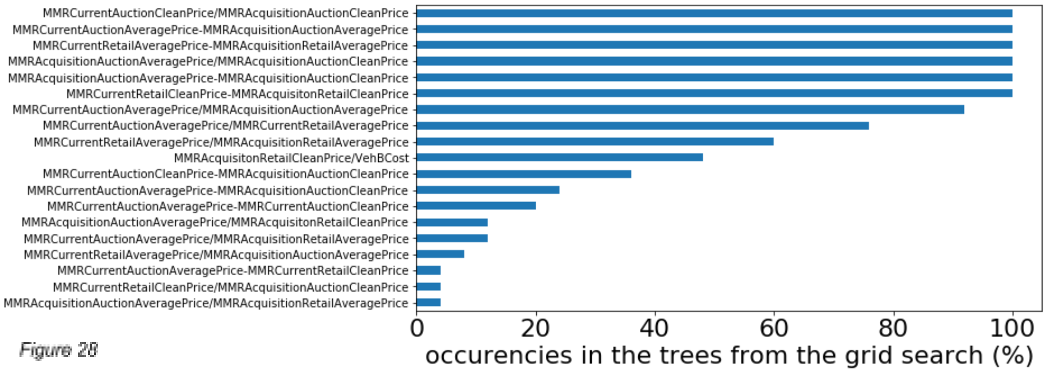


Figure 28

The prices that have been chosen all the times are:

- MMRCurrentAuctionCleanPrice / MMRAcquisitionAuctionCleanPrice
- MMRCurrentAuctionAveragePrice-MMRAcquisitionAuctionAveragePrice
- MMRAcquisitionAuctionAveragePrice/MMRAcquisitionAuctionCleanPrice
- MMRAcquisitionAuctionAveragePrice-MMRAcquisitionAuctionCleanPrice
- MMRCurrentRetailAveragePrice-MMRAcquisitionRetailAveragePrice

The dataset for the classification has been enriched with those features to improve performances.

Decision Tree

To compute the Decision Tree both categorical, numerical and synthetic features have been used.
Numerical features have been normalized between 0-1, subtracting the minimum and dividing by the maximum. Values of categorical attributes with low frequencies have been joined in “other” field for **SimpleSubModel** and **SimpleModel**.
Then, categorical features have been transformed in numerical form via dummy encoding and eventually with one hot transformation. The latter transformation allowed the algorithm to work better and therefore the one hot categorical features have been used.

The attributes used are listed in Appendix section B.
To evaluate our models, we first created a dummy classifier that assign 1 to all the values, obtaining a baseline with a f1 score value for the positive class 0.22.

Learning of different decision trees/classification algorithms with different parameters and gain formulas with the object of maximizing the performances

To create the Decision Tree with best parameter another grid search has been run with gini criterion, min splits 2, max depth = [None, 4, 5, 6, 7, 8, 9] and min samples = [1, 20, 21, 22, 23] getting values as shown in the plots of *Figure 29.*[1]

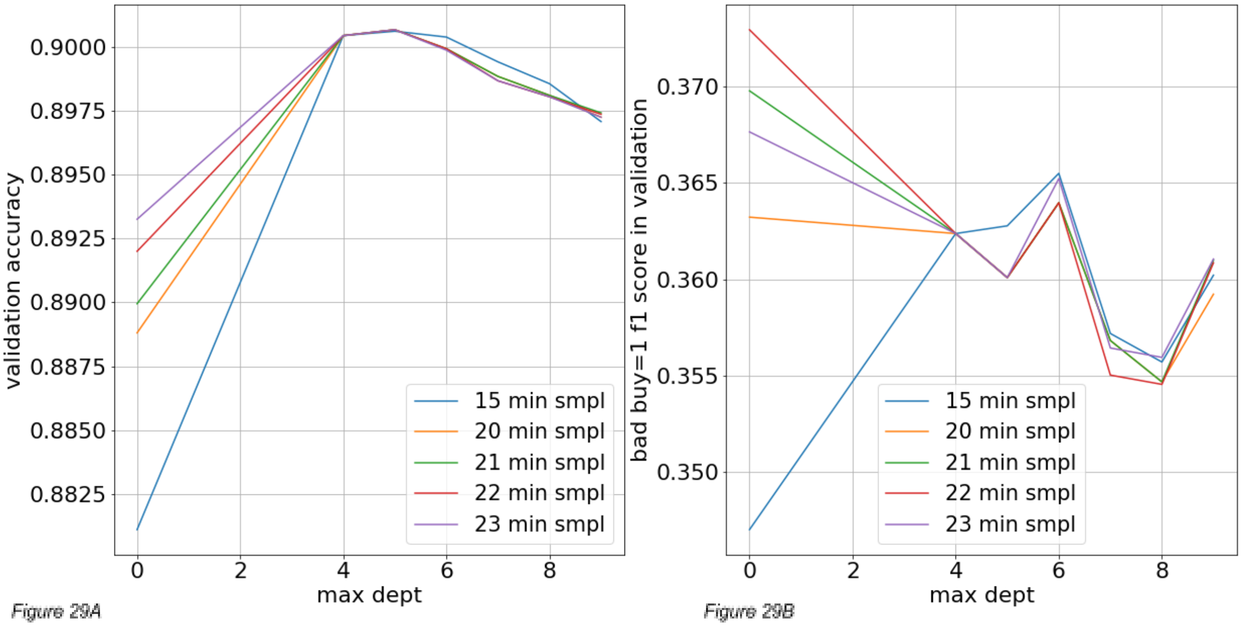


Figure 29A *Figure 29B*

Discussion of the best prediction model

To choose the best model, it was not possible to look at the accuracies on the validation set because they are overall very similar, therefore is not clear which is the best; but looking at the f1 score the best model can be more clearly identified.
The best model was produced using the gini criterion with max depth 6, min splits 2 and min samples in leaves 15.

TreeClassifier: max_dept = 6, min_split = 2 min_samples_leaf= 15		
	Train Phase	Validation Phase
accuracy	0.91	0.89
f1 for IsBadBuy = 0	0.948	0.940
f1 for IsBadBuy = 1	0.454	0.372

Table 4

[1] Max depth 0 stands for None

Our model performs better than the baseline.

Decision trees interpretation

The used Decision Tree is shown in the following images. For space reasons, the original image has been splitted but is reported in Appendix section C.

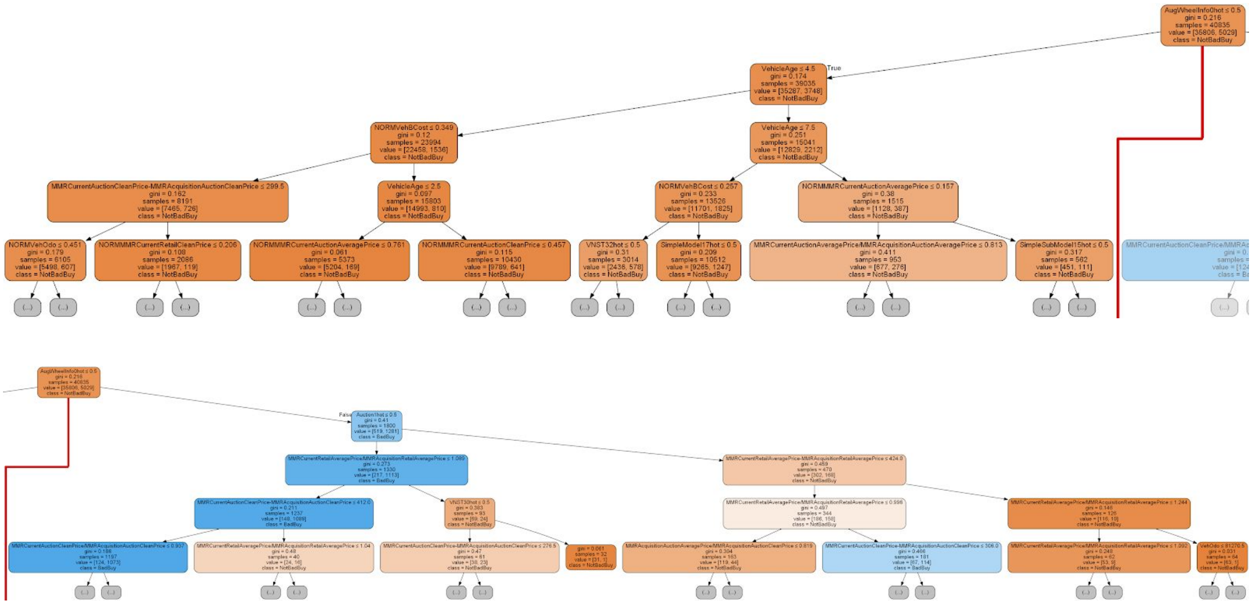


Figure 30

The root is AugWheelInfo < 0.5, which means that the tree checks whether AugWheelInfo is or not 0: this is the most important feature to consider when deciding. In the case of true answer, all the subtrees leads the inference to NotBadBuy (orange blocks).

VehicleAge and Auction1hot are very important features as well. In particular Auction1hot < 0.5 and MMRCurrentRetailAveragePrice / MMRAcquisitionRetailAveragePrice < 1.089 are the checks that mostly assign the BadBuy value to 1: the lift of the retailed average price from the acquisition time to the current day marks a car as bad buy.

Comparing different models

The classification task has been performed using various classifiers, specifically: a linear regressor, k-nearest neighbor, naïve bayes and random forest.

The dataset have been splitted in 70% training set and 30% validation set.

In the final table there are listed the milliseconds needed for the training and inference phase; the accuracies for the test and the train and the f1 scores for both classes at training time and testing time.

The features used as inputs are all the numerical features, including all the MMR prices, the synthetic features, and the categorical features, unless otherwise specified. All listed in the section B of appendix. Numerical and prices features have been normalized between 0-1.

Linear regression

The regression have been performed with and without weighting the samples according to the target class **IsBadBuy**, however the performances are very similar. For the categorical features the one hot encoding was the best according to the metrics for the classifier. In the table are shown the performances metrics obtained by the regressor weightening the samples.
The best threshold to be used as decision boundary was discovered to be 0.7.

KNN

A grid search have been performed in order to choose the best parameter for the k neighbors.
The best one according to the highest f1-score for both the classes is k equals to 3.
The majority of time is spent in the inference phase and the time need grows with the parameter k.

Naïve Bayes

The input features chosen for the naïve bayes are just the dummy categorical ones since
the price features have an high correlation, the synthetic features are redundant and because the best performances are obtained using just the categorical features. The higher performances are not obtained codifying them as one hot, rather codified them as *dummy* variables.

Random Forest

The input features used for the random forest are the numerical features, included all the MMR prices features, the synthetic features, and the categorical features codified as dummy.
A grid search have been run for this classifier using as maximum dept [None,10,11,15,20] and as minimum samples [9,10,11,13,15,20]. The best resulting model is the one with None as max dept and 11 minimum leaf samples. 8 threads are used for such classifier.

The final results are in the following table.

	Training time (ms)	Test time (ms)	Train Acc.	Train negative f1-score	Train positive f1-score	Test Acc.	Test negative f1-score	Test positive f1-score
Decision Tree	4.3	0.11	0.9	0.95	0.45	0.89	0.94	0.37
Linear reg.	2.225	0.084	0.9	0.93	0.41	**0.90**	0.93	0.39
KNN	1.649	7.541	0.88	0.94	0.40	0.86	0.91	0.12
Naïve Bayes	**0.034**	**0.01**	0.90	0.94	0.37	0.89	**0.94**	0.37

Random forest	1.671	0.125	**0.95**	**0.97**	**0.81**	0.88	0.93	**0.42**

Table 5

According to *Table 5,* the knowledge-free model k-nearest neighbors has the characteristic to need a higher time for the inference step rather than its *learning* step, all the others perform faster in the predicting phase than training; Naïve Bayes is the fastest one in both phases also due to the fact that less features have been used for such classifier. NB classifier performs as the Decision Tree at test phase

The random forest is the model that predicted the positive class of a **BadBuy** more precisely and with a higher recall than the other models, on an harmonious average, is therefore the best model.

Appendix

Section A

Enumerative description of the features divided in categorical and numerical ones.

Categorical:

1. **RefID**: the primary key of the record
2. **IsBadBuy**: identifies if the vehicle purchased was a bad investment
3. **PurchDate**: the date in which the vehicle was purchased at auction
4. **Auction**: the provider of the purchased car
5. **Make**: the brand of the vehicle
6. **Model**: the model of the vehicle
7. **Trim**: the decorative features' level of the vehicle
8. **SubModel**: the submodel of the vehicle
9. **Color**: vehicle color
10. **Transmission**: vehicles transmission type
11. **WheelType**: the vehicle wheel type description (Alloy, Covers)
12. **WheelTypeID**: the type id of the vehicle wheel (redundant feature)
13. **Nationality**: the manufacturer's country
14. **Size**: the categorical size of the vehicle
15. **TopThreeAmericanName**: identifies if the manufacturer is one of the top three American manufacturers and express which one is
16. **PRIMEUNIT**: identifies if the vehicle would have a higher demand than a standard purchase
17. **AUCGUART**: the level guarantee provided by auction for the vehicle (Red/Green)
18. **BYRNO**: unique number assigned to the employee who purchased the vehicle
19. **VNZIP1**: the zipcode where the car was purchased
20. **VNST**: the state where the the car was purchased
21. **IsOnlineSale**: Identifies if the vehicle was originally purchased online

Numerical:

22. **VehYear**: the manufacturer's year of the vehicle
23. **VehicleAge**: numbers of years elapsed since the model was made
24. **VehOdo**: the vehicle odometer reading
25. **MMRAcquisitionAuctionAveragePrice**: acquisition price for this vehicle in average condition at time of purchase
26. **MMRAcquisitionAuctionCleanPrice**: acquisition price for this vehicle in the above average condition at time of purchase
27. **MMRAcquisitionRetailAveragePrice**: acquisition price for this vehicle in the retail market in average condition at time of purchase
28. **MMRAcquisitonRetailCleanPrice**: acquisition price for this vehicle in the retail market in above average condition at time of purchase
29. **MMRCurrentAuctionAveragePrice**: acquisition price for this vehicle in average condition as of current day

30. **MMRCurrentAuctionCleanPrice**: acquisition price for this vehicle in the above condition as of current day
31. **MMRCurrentRetailAveragePrice**: acquisition price for this vehicle in the retail market in average condition as of current day
32. **MMRCurrentRetailCleanPrice**: acquisition price for this vehicle in the retail market in above average condition as of current day
33. **VehBCost**: acquisition cost paid for the vehicle at time of purchase
34. **WarrantyCost**: warranty price (term=36month and millage=36K)

Section B

List of the used features for the Decision Tree.
Numerical Features:

VehicleAge,
NORMMMRCurrentAuctionAveragePrice
NORMMMRCurrentAuctionCleanPrice
NORMMMRCurrentRetailAveragePrice
NORMMMRCurrentRetailCleanPrice
NORMMMRAcquisitionAuctionAveragePrice
NORMMMRAcquisitionAuctionCleanPrice
NORMMMRAcquisitionRetailAveragePrice
NORMMMRAcquisitonRetailCleanPrice
NORMVehBCost
NORMVehOdo
NORMWarrantyCost

Synthetic Features:

MMRCurrentAuctionAveragePrice/MMRAcquisitionAuctionAveragePrice
MMRCurrentRetailAveragePrice/MMRAcquisitionRetailAveragePrice
MMRAcquisitionAuctionAveragePrice/MMRAcquisitionAuctionCleanPrice
MMRCurrentAuctionCleanPrice/MMRAcquisitionAuctionCleanPrice
MMRCurrentAuctionCleanPrice-MMRAcquisitionAuctionCleanPrice,
MMRCurrentRetailAveragePrice-MMRAcquisitionRetailAveragePrice

Categorical Features:

SimpleSubModel - hot encoding (17 features),
Auction - hot encoding (3 features)
Make - hot encoding (33 features)
SimpleModel - hot encoding (59 features)
BoolTrim - hot encoding (2 features)
Color - hot encoding (16 features)
Nationality - hot encoding (4 features)
Size - hot encoding (12 features)
TopThreeAmericanName - hot encoding (4 features)
VNST - hot encoding (37 features)
AugWheelInfo - hot encoding (4 features)

Section C

Decision Tree

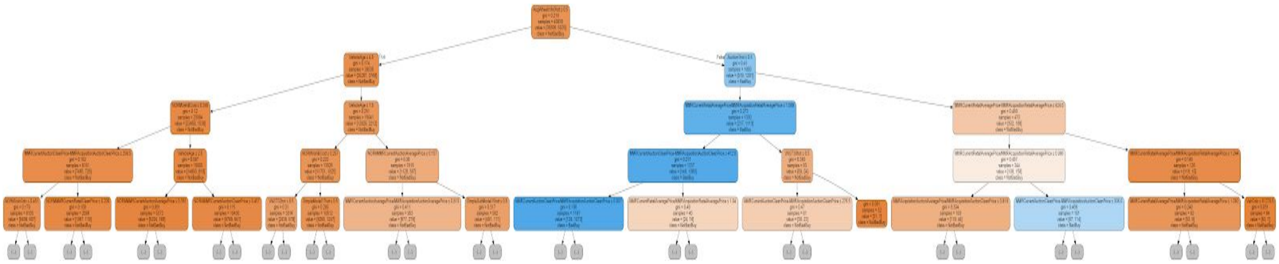


Figure I

Section D

Features for the Classifiers unlike differently specified:

Categorical Features: SimpleSubModel, Auction, Make, SimpleModel, BoolTrim, Color, Nationality, Size, TopThreeAmericanName, VNST, AugWheelInfo

Numerical Features: VehicleAge, VehOdo, VehBCost, WarrantyCost, MMRCurrentAuctionAveragePrice, MMRCurrentAuctionCleanPrice, MMRCurrentRetailAveragePrice, MMRCurrentRetailCleanPrice, MMRAcquisitionAuctionAveragePrice, MMRAcquisitionAuctionCleanPrice, MMRAcquisitionRetailAveragePrice, MMRAcquisitonRetailCleanPrice

Synthetic Features: MMRCurrentAuctionCleanPrice/MMRAcquisitionAuctionCleanPrice, MMRCurrentAuctionAveragePrice - MMRAcquisitionAuctionAveragePrice, MMRAcquisitionAuctionAveragePrice / MMRAcquisitionAuctionCleanPrice, MMRAcquisitionAuctionAveragePrice - MMRAcquisitionAuctionCleanPrice, MMRCurrentRetailAveragePrice - MMRAcquisitionRetailAveragePrice