CARVANA: IS A BAD BUY? A SURVEY THROUGH DIFFERENT CLASSIFICATION METHODS

Alessandro Cudazzo
alessandro@cudazzo.com
Department of Computer Science
University of Pisa

Giulia Volpi
giuliavolpi25.93@gmail.com
Department of Computer Science
University of Pisa

Data Mining Project: 9 CFU

Academic Year: 2019/2020

ABSTRACT

In a previous work [1] the authors have analyzed the Carvana dataset provided by Kaggle in order to find out if there was a way to predict if a car was a bad or a good purchase. The data provided by Carvana was really unbalanced and different analyses were performed: data understanding, clustering, pattern mining, and classification. In particular, a decision tree was used as a classification model. It turns out that was really difficult to achieve a good model that was able to solve the task. The aim of this paper is to go further and provide a survey with other classification algorithms as Random Forest, Naive Bayes, and K-NN in order to check if a better classification algorithm is able to generalise the data or if Carvana needs to collect more reasonable information.

***K*eywords** Data Mining, Machine Learning, Classification Task, Random Forest, Naive Bayes, K-NN

1. Introduction

It is well known that Decision Tree has the advantage of being simple to understand, interpret and visualise but it can be unstable, meaning that a small change in the data can lead to a large change in the structure of the optimal decision tree. Indeed can happen that many other classification perform better with similar data. One way to solve this type of problem is to use ensemble methods as Bootstrap Aggregation or Bagging techniques, which, in the case of the decision tree, is equivalent to employing a Random Forest classifier that is one of the most popular and most powerful machine learning algorithms. On the contrary, a Random Forest is not as easy to interpret as a single decision tree. Of course there are a lot of machine learning algorithms that could be used in order to compare them with the Decision Tree, for example Naive Bayes and K-NN classifier. Each of them has advantages and disadvantages and could work better or worse on a dataset.
Carvana is a start-up business launched by a well-established American company and has the goal to build a model to advise future customers whether a purchase could be a good or bad buy. That's why they launched a Kaggle competition [2] to achieve this goal. For these reasons it is interesting to continue the analysis started in [1] and try to understand if it is possible to improve the results, obtained previously by using the Decision Tree, with other classification algorithms as `Random Forest`, `Naive Bayes` and `K-NN`.

The remaining text in the paper is structured as follows. In Section 2 all the experimental settings relative to model selection and data set preprocessing are presented. The analyses related to the three chosen machine learning algorithms with their results are exposed in Section 3, whereas Section 4 is devoted to conclusions.

2. Experimental Setting and Dataset

The dataset provided by Carvana [2] (Training and Test set) are highly unbalanced with respect to the target attribute (`IsBadBuy`) (see [1]). We've decided to continue with all the assumptions done in [1]. Therefore, the features selected are:

- **Categorical**: `Auction`, `Size`, `VNST`, `WheelTypeID`
- **Numerical**: `VehicleAge`, `VehOdo`, `WarrantyCost`, `VehBCost`, `MMRAcquisitionAuctionAveragePrice`, `MMRCurrentRetailAveragePrice`, `MMRAcquisitionRetailAveragePrice`, `MMRCurrentAuctionAveragePrice`

Moreover, the validation scheme used is the same, we divided the data set (original training set) into random training (70% of data) and validation (30% of data) subsets by using a stratified splitting that allowed us preserving the percentage of samples for each class. From now on we will refer to these sets as external training and validation sets. The validation set will not be used initially but its purpose is to compare the models found. Since this dataset is highly unbalanced with respect to the target attribute object of the classification (`IsBadBuy`), we have to use some sampling method to try to solve this problem. In [1] the decision tree had the best result using the **undersampling** technique, therefore we will use this for all the other classifiers in this paper.
As a common goodness criterion among the models we decided to use the value of *ROC AUC* because the accuracy alone is not a good evaluation option with class-imbalanced data sets.
For the model selection of each classifier, except for the Naive Bayes, we performed a *random grid search with 3-folds stratified cross-validation* on the external training set. This will split the external training set in an internal training and validation set (the undersampling will be performed only on the internal training set). So, in order to compare all models that will be found, we will retrain the models with the external training set (by applying undersampling on it) and compared them with the external validation set (with a class balance similar to the original dataset).

All previous assumptions will be used for the methods in the next section.

3. Methods

This section will contain all the analyses carried out with the three machine learning models chosen for this survey: `Random forest`, `Naive Bayes` and `K-NN`.

3.1 Random Forest

We performed the random grid search illustrated in Sec. 2 with parameters ranges showed in Tab. 3.1 and with *ROC AUC* as goodness criterion. As a result, we have achieved three best models and all the information about them are reported in Tab. A.1 in the Appendix A. So, in order to compare all three models found, we evaluated them with the external validation set. Tab. 3.2 shows the results of the comparison. Finally, we have chosen `Model 2` as best model for this methods because has the best `ROC AUC` equal to 0.644 on the external validation. The *confusion matrix* and *ROC Curve* are reported, respectively, in Fig. 3.1a and in Fig. 3.1b. It can be seen from the confusion matrix that `Model 2` is able to ensure the same percentage of

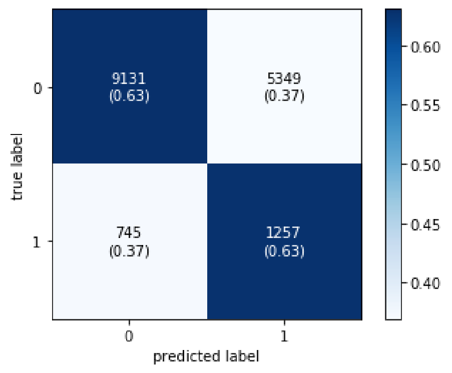
True Positive and True Negative (0.63%) and it seems to be very balanced in predicting whether a car is a good or a bad purchase.

Hyperparameters	**Values range**
n_estimators	2, 100
criterion	gini, entropy
max_depth	None, 2 - 100
min_sample_split	2 - 100
min_sample_leaf	1 - 100

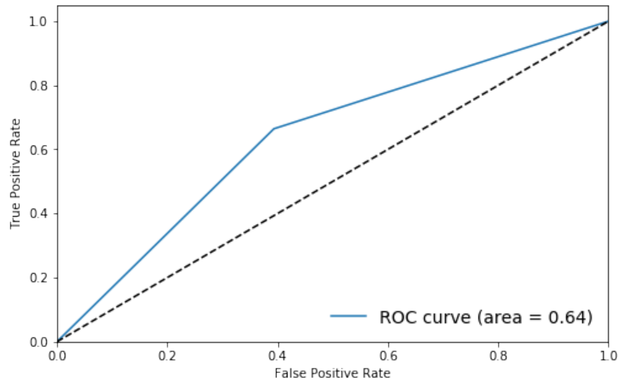
Table 3.1: Range of Hyper-parameters for the random grid search of the Random Forest Classifier.

Scoring on the common Validation set						
Model	**TR - ROC auc**	**TR - Accuracy**	**TR - F1**	**VL - ROC auc**	**VL - Accuracy**	**VL - F1**
Model 1	0.697	0.699	0.705	0.636	0.617	0.295
Model 2	0.822	0.822	0.822	0.644	0.624	0.302
Model 3	0.721	0.721	0.728	0.640	0.613	0.298

Table 3.2: Evaluation of the three models of Random Forest Classifier on the external training and validation set.



(a) Normalized Confusion Matrix.



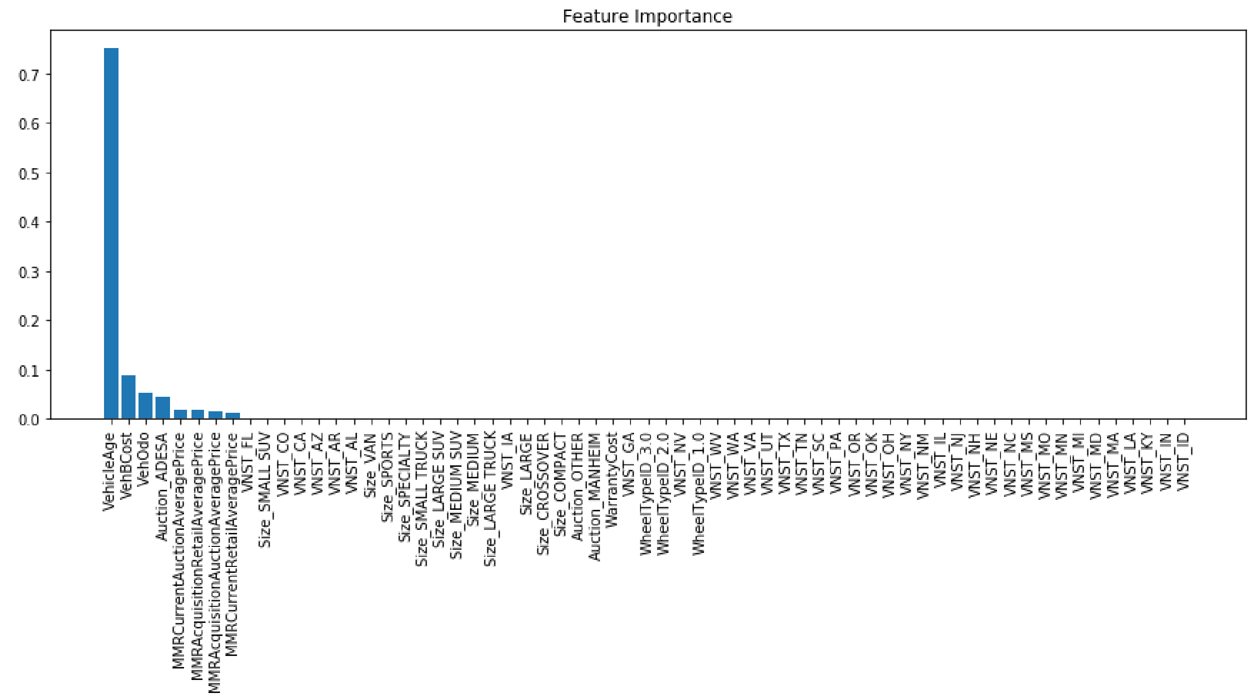
(b) ROC Curve.

Figure 3.1: Confusion Matrix and ROC curve evaluated for the best Random Forest model on the external validation set.

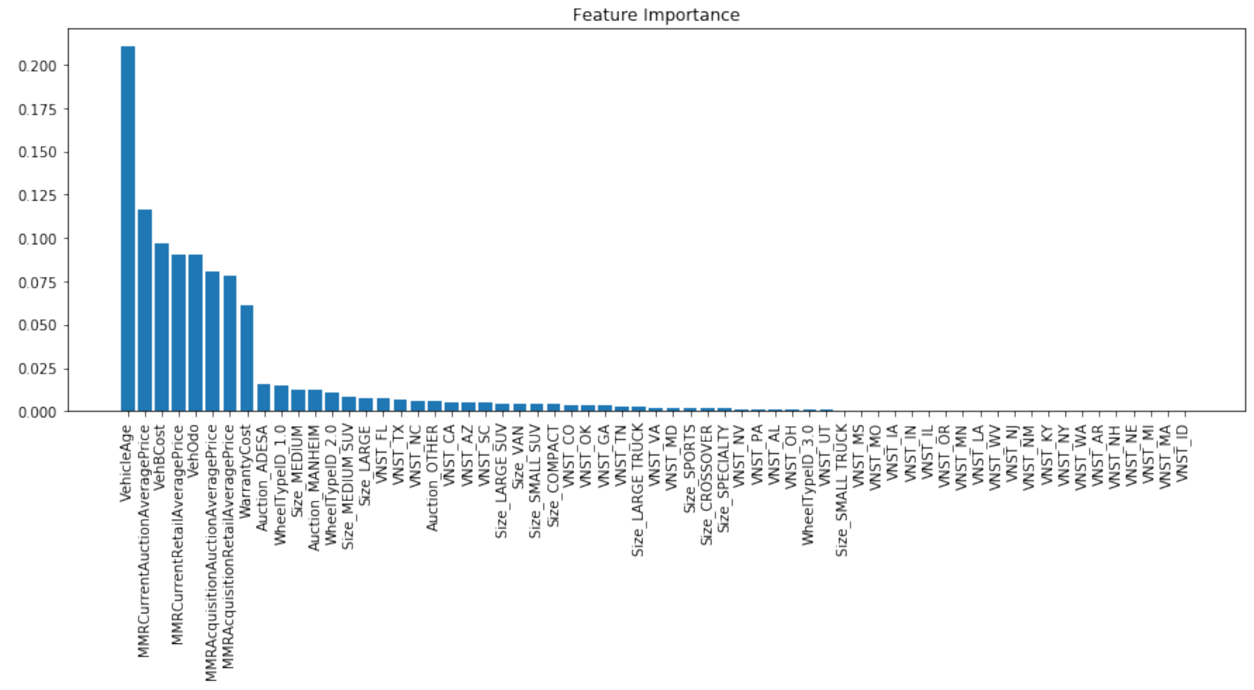
In addition, in Fig. 3.2, we also compared the weight of the features in the decision tree with those used in the random forest. As we can see, the decision tree almost always bases its decision on only few features (`MMRAcquisitionAuctionAveragePrice`, `VehicleAge`, `VehBCost`, `VehOdo`, `WarrantyCost` and `Auction` and other with low relevance), while the random forest makes use of other features that have a greater weight than the decision tree (`Size`, `WheelTypeID`, `MMRCurrentRetailAveragePrice`, `MMRAcquisitionAuctionAveragePrice`, `MMRAcquisitionRetailAveragePrice`, and other with low relevance).

3.2 Naive Bayes

As Naive Bayes classifier we have used the Gaussian Naive Bayes where the likelihood of the features is assumed to be Gaussian. It has no hyperparameters to tweak, so there is no need to set a GridSearch. We



(a)



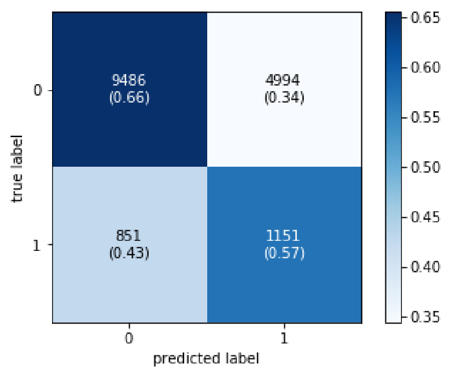
(b)

Figure 3.2: (a) Features importance for the Decision Tree Classifier. (b) Feature importance for the Random Forest Classifier.

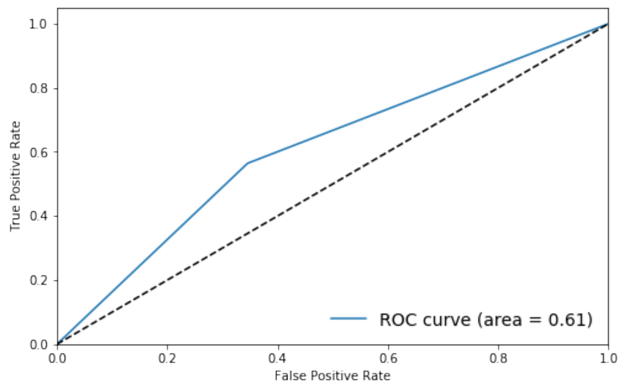
simply trained the classifier on the external training set and then evaluated it on the external validation set. We obtained 35.7% miss classification errors on the external validation set and `ROC AUC` equal to 0.609. In Table 3.3 we can see the results of the Naive Bayes classifier on the external training and the validation set. The *confusion matrix* and *ROC Curve* are reported, respectively, in Fig. 3.3a and Fig. 3.3b.

TR - ROC auc	TR - Accuracy	TR - F1	VL - ROC auc	VL - Accuracy	VL - F1
0.617	0.617	0.601	0.609	0.643	0.278

Table 3.3: Naive Bayes evaluation on the external training and validation set.



(a) Normalized Confusion Matrix.



(b) ROC Curve.

Figure 3.3: Confusion Matrix and ROC curve evaluated for the final Naive Bayes model on the external validation set.

3.3 K-Nearest Neighbors

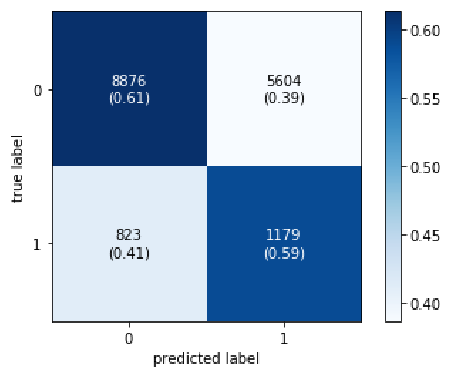
The K-NN in order to perform the classification of an object (x) finds the Ks neighbours to that object, compute the distance between x and the other neighbours. Usually *Euclidean distance* distance is used, but other types of distance can also be used. For example the distance to *Manhattan distance*. Therefore, we performed two random grid searches as illustrated in Sec. 2 with parameters ranges showed in Tab. 3.4, one using the *Euclidean distance* and the other using *Manhattan distance*. In both cases we used *ROC AUC* as goodness criterion. For each of grid we extracted the three best models, shown in Tab. A.2 in the Appendix A. So, to compare them we performed the same approach used for Random Forest models and all results are shown in tables Tab. 3.5. Finally, we have chosen `Model 4` as final model because has the best `ROC AUC` value equal to 0.610 on the external validation. The *confusion matrix* and *ROC Curve* are reported, respectively, in Fig. 3.4a and in Fig. 3.4b.

Random Grid Search		
Hyperparameters	**Euclidean distance**	**Manhattan distance**
n_neighbors	2 - 200	2 - 200
algorithm	auto	auto
weights	uniform, distance	uniform, distance
leaf_size	2 - 200	2 - 200
p	2	1

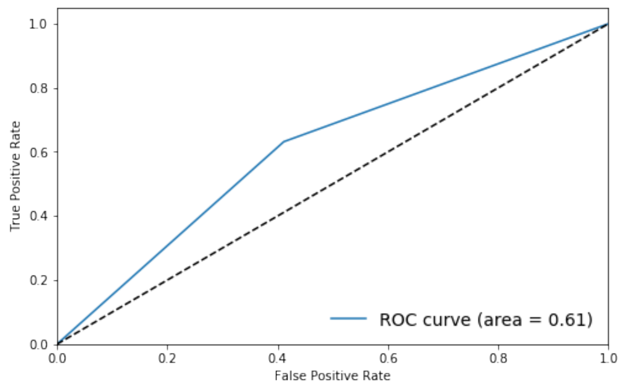
Table 3.4: Hyper-parameters for the random grid search of the K-NN Classifier

Scoring on the common Valdiation set						
Model	**TR - ROC auc**	**TR - Accuracy**	**TR - F1**	**VL - ROC auc**	**VL - Accuracy**	**VL - F1**
Model 4	1.0	1.0	1.0	0.610	0.594	0.274
Model 5	1.0	1.0	1.0	0.610	0.592	0.274
Model 6	1.0	1.0	1.0	0.608	0.591	0.272
Model 7	1.0	1.0	1.0	0.593	0.589	0.261
Model 8	1.0	1.0	1.0	0.594	0.593	0.262
Model 9	1.0	1.0	1.0	0.592	0.590	0.261

Table 3.5: Evaluation of the six models of K-NN Classifier, with Euclidean distance (p=2) and with Manhattan distance (p=1), on training and validation set



(a) Normalized Confusion Matrix.



(b) ROC Curve.

Figure 3.4: Confusion Matrix and ROC curve evaluated for the final K-NN model on the Validation Set.

3.4 Results

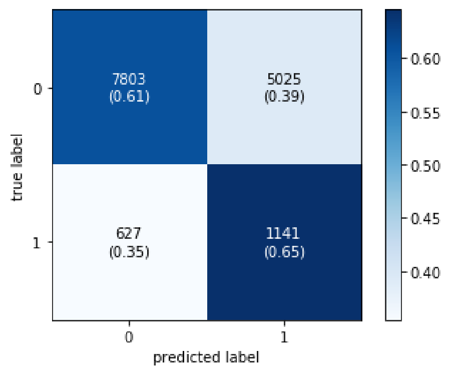
Tab. 3.6 shows the results of the best models found in this survey on the common validation set (the external one), also for comparison the result found on the decision tree in [1] has been added. From this it emerged that the best model was found through the use of the Random Forest classifier, so we choose this model as final ones. To measure the quality of the final model we proceeded to estimate the prediction error on the test set provided by Carvana and the result is shown in the Tab. 3.7. The *confusion matrix* and *ROC Curve* are reported, respectively, in Fig. 3.5a and in Fig. 3.5b

Scoring on the common Valdiation set						
Classifier	**TR - ROC auc**	**TR - Accuracy**	**TR - F1**	**VL - ROC auc**	**VL - Accuracy**	**VL - F1**
Random Forest	0.697	0.699	0.705	0.636	0.617	0.295
Naive Bayes	0.617	0.617	0.601	0.609	0.643	0.278
K-NN	1.0	1.0	1.0	0.610	0.594	0.274
Decision Tree	0.627	0.627	0.628	0.621	0.610	0.284

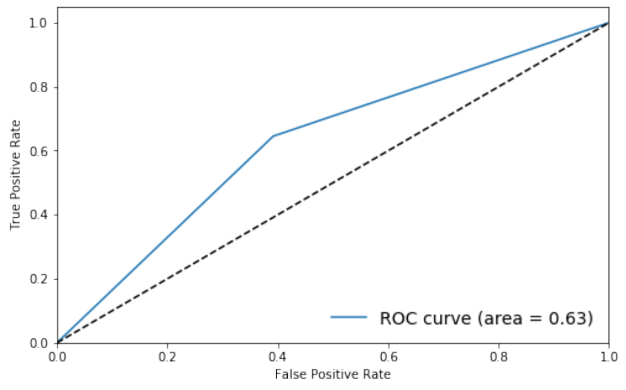
Table 3.6: Evaluation of all the models on the external training and validation set.

	ROC AUC score	**Accuracy**	**F1 Score**	**Precision**	**Recall**
Random Forest	0.627	0.613	0.288	0.19	0.65

Table 3.7: Random Forest evaluation on the test set.



(a) Normalized Confusion Matrix.



(b) ROC Curve.

Figure 3.5: Confusion Matrix and ROC curve evaluated for the final Random Forest model on the Tset Set.

In addition, we have compared the running time at training and predicting phase for the final model for each methods.

Runtime at Training and Predicting Phase

We have analysed, for every model, including the Decision Tree classifier, the runtime at training and predicting phase. In order to do that, we have calculated the time needed to perform the training of individual models with the external training set (runtime at training) and the time needed to evaluated the external validation set (runtime at predicting). All the operation have been performed on an Intel I7-8565U with 4 core (8 Threads) at 1.80GHz (Intel Turbo Boost 4,60 GHz) and 16GB of RAM and the results are shown in Tab. 3.8. By looking at the running times we can immediately see that the Random Forest as longest time

Classifier	**Time at Training Phase**	**Time at Predicting Phase**
Random Forest	$1.93\,s \pm 18.7\,ms\ per\ loop$	$456\,ms \pm 1.11\,ms\ per\ loop$
Naive Bayes	$15.2\,ms \pm 1.38\,ms\ per\ loop$	$30.9\,ms \pm 2.82\,ms\ per\ loop$
K-NN	$79.4\,ms \pm 1.88\,ms\ per\ loop$	$1.8\,s \pm 21.1\,ms\ per\ loop$
Decision Tree	$33.8\,ms \pm 1.95\,ms\ per\ loop$	$4.74\,ms \pm 947\,\mu s\ per\ loop$

Table 3.8: Times required for the models for the training and predicting phases, calculated with 7 runs with 1 loop each.

on training respect all the others. This happens because the Random Forest classifier constructs a multitude of decision trees at training time. Instead, regarding the predicting time, the K-NN has the highest runtime and this result reflects the characteristics of the algorithm. Indeed, during the classification phase, the model calculates the distances between all the points and then goes to find, for each point, its k-nearest neighbours. On the contrary, if we take into consideration the shorter time, we see that the Naive Bayes has the lowest training time. This because, during the training phase, the Naive Bayes simply calculates the *Prior Probabilities* of different classes and the *Likelihood* of different features for each class. In fact, the Naive Bayes tends to be faster when applied to big data and the K-NN is usually slower for large amounts of data. As for the predicting phase, the decision tree has the lowest time because it simply scrolls the tree to classify a new instance.

4. Conclusions

We presented four different classifiers, but all of them performed very poor results. Indeed, the choice to use undersampling technique to solve the problem of the unbalanced dataset has probably led to underfitting models. Maybe, if we had spent more time in research with oversampling, we would have found better models, but for reasons of lack of time and to remain consistent with the initial choices, we preferred to continue with the undersampling. Therefore, the Random Forest model obtained will not lead to a good classifier and cannot be used by the company to predict whether a car will be a good or bad purchase. However, it's interesting to note that the most significant features for the classification, concerning the Random Forest, are: `MMRAcquisitionAuctionAveragePrice`,`MMRCurrentRetailAveragePrice`, `Auction`,`VehicleAge`, `WheelTypeID`, `VehBCost`, `Size`, `VehOdo`, `WarrantyCost`, `MMRAcquisitionAuctionAveragePrice`, `MMRAcquisitionRetailAveragePrice`. This result is much more interesting than that obtained with the Decision Tree, because in this last case the classifier gave very great importance to `VehAge` and very little to the other features. Instead, for the Random Forest, `VehAge` is still the main feature for the classification, but now the others have a considerable weight. This result is consistent with our dataset analysis done in the previous work [1].

So, as regards future work: to improve the results found, it might be of interest applying different undersampling techniques (NearMiss, TomekLinks, CNN, NeighbourhoodCleaningRule, ENN) and doing more experiments to find better models also using oversampling. It would also be very interesting to see how a Neural Network behaves on this dataset. After all these further tests we could therefore understand if the unsatisfactory results of the classification are mainly related to the structure of the data set (it requires to collect additional records and better balance the data) or to the chosen models.

Appendix A. Grid Search

Random Forest - Best Models			
	Model 1	**Model 2**	**Model 3**
n_estimators	89	87	74
criterion	entropy	gini	gini
max_depth	28	59	74
min_sample_split	99	23	34
min_sample_leaf	7	3	11
TR roc_auc	0.741±0.006	0.822±0.001	0.752±0.002
VL roc_auc	0.686±0.004	0.686±0.002	0.686±0.002
TR Accuracy	0.639±0.009	0.665±0.004	0.643±0.004
VL Accuracy	0.620±0.008	0.619±0.002	0.622± 0.002
TR F1 Score	0.324±0.005	0.375±0.002	0.331±0.001
VL F1 Score	0.289±0.003	0.288±0.002	0.290±0.002

Table A.1: Best three models from the grid search for Random Forest Classifier.

K-NN - Best Models						
	Model 4	**Model 5**	**Model 6**	**Model 7**	**Model 8**	**Model 9**
n_neighbors	135	131	118	129	170	144
algorithm	auto	auto	auto	auto	auto	auto
weights	distance	distance	distance	distance	distance	distance
leaf_size	142	193	157	34	190	29
p	2	2	2	1	1	1
TR roc_auc	1.000±0.000	1.000±0.000	1.000±0.000	1.000±0.000	1.000±0.000	1.000±0.000
VL roc_auc	0.643±0.008	0.643±0.008	0.642±0.007	0.651±0.002	0.651±0.005	0.651±0.00
TR Accuracy	0.710±0.009	0.700±0.011	0.692±0.013	0.696±0.002	0.698±0.005	0.688±0.015
VL Accuracy	0.614±0.007	0.602±0.006	0.593± 0.010	0.603±0.003	0.602±0.006	0.594± 0.018
TR F1 Score	0.456±0.008	0.447±0.009	0.441±0.010	0.444±0.001	0.446±0.004	0.438±0.011
VL F1 Score	0.268±0.004	0.267±0.003	0.265±0.003	0.275±0.004	0.272±0.005	0.274±0.001

Table A.2: Best six models from the grid searches, with Euclidean distance (p=2) and Manhattan distance (p=1), for K-NN Classifier.

References

[1] F. Achena A. Maslennikova A. Cudazzo, G. Volpi. Carvana, is a bad buy?, 2020.

[2] Carvana. Data Mining 2019/2020 - Unipi. `https://www.kaggle.com/c/data-mining-20192020-unipi/`, 2019.