To sell or ~~not~~ at what price to sell!

Understanding customer pricing expectation in the pre-owned car market

I. Introduction

1.1 Motivation

Automotive industry is facing unprecedented upheaval. Iconic brands all across the world be it GM in US, whose chief executive once claimed that GM is indistinguishable from America; TATA Motors in India which was once considered a showcase for Asian engineering know-how, PSA group in Europe which once held a place of pride in France and considered an emblem of its recovery and resurgence after the second world war, all of these groups suffered either bankruptcies or the credit restructuring in the last decade. These are but few examples of an industry suffering an identity crisis.

The reasons for present state of affairs are many and varied but consumer dissatisfaction must be counted among the main causes. [see 4, 5, 6 to mention a few] Considering that after home, car is the most expensive purchase that most consumer will ever make, it is rather startling to note that in independent surveys, the consumers have repeatedly ranked their experience of buying a car very poorly.

When analysing the causes of dissatisfaction we first note the segmentation and frequency of purchase. As the industry can provide stable, middle income jobs every big nation tries to produce a national champion in automotive sector. The upside is a crowded market. China has 70 car brands while US has close to 50. This when combine with the infrequency of purchase has resulted in a situations where manufacturers don’t always have the keenest insight into their customers.

When the industry was transitioning from those clunky slower than horse vehicles to the age formula one races, the sheer magnitude of improvement from one generation of vehicles to the next largely compensated for these problems. However now that the industry has matured, improving customer experience is critical for the industry.

Consumers have repeatedly named the price negotiation as the top reason for the dissatisfactions. It’s not that the consumers want lower prices, owing to the saturation of the market, competitive models in every segment are easy to find, but that the consumers find the whole process to be very opaque. Applying the methods of statistical learning on the pricing is the key focus of this project.

1.2 On the choice of the dataset:

At the outset we encountered several hurdles. To begin with, owing to the fragmentation of the industry, standardise data beyond the basic surveys is very hard to come by. In addition, impressive videos of the automated assembly lines belies the challenges of the industry with the digitisation especially in the area of customer relationship management.

Our focus was to find a dataset that gives us insight into the consumer behavior and experience vis-à-vis car price. Upon additional analysis we discovered that 75% of the market is of the second hand vehicles[3]. We were therefore pleased to find a dataset containing the classified advertisement for selling pre-owned on the German site of eBay.[7]

Germans are known for their passionate attachment to their vehicles. In addition, Germany is also the fourth biggest car market and, among the five big markets, it is also the second wealthiest in terms of per capita income. These factors combine to produce a richly diverse dataset. Therefore the insights gained from this dataset can be readily generalized to other markets.

This dataset in essence provides us with the customer’s view of the value of the vehicle after owning it for a few years. The fact that we were able to predict with upto 96% accuracy the price that a customer would demand is evidence of the fact that there is an underlying pattern to the customer’s expectation and getting an insight into this pattern can be lucrative for the industry.

1.3 Previous work:

The manufacturer interacts with the customer through the third party dealers and the dealer’s incentive focuses more on selling the vehicles rather than doing the tedious and non-remunerative work of maintaining the customer database. This along with the culture of fierce and adversarial copyright protection and the high segmentation in the market which results in absence dominant player across the different geographic market means that industry standard datasets which are critical for developing generally valid and applicable insights are not readily available. This militate against the kind of consumer focused (as opposed to that focused on improving manufacturing) research prevalent in FMCG and IT industries.

The most useful work that we did find was done in the insurance market. Insurance providers do regularly interact with the customer directly and cater to the customers from different automotive brands. However the main focus of the insurance providers is assessing the various kind of risk associated with the consumers. [See for instance 1 & 2]

On the webpage of the dataset shared with this report, some users have undertaken analysis of the data. While most such work concerns with the data cleaning or data visualization, a handful of users have tried to drive useful insights by using statistical learning techniques. Such work occasionally provided us with a headstart in many areas, although our work was distinct in its focus on consumer price expectations and its utility in the wider industry in that context.

II Running the project

2.1. Data exploration and preparation

The dataset used for the statistical learning task was retrieved from Kaggle datasets. It initially contained 20 variables with 371528 observations. This dataset, however, had numerous observations with missing entries and unreasonable prices (such as 999999 EUR for VW Golf) and we did the initial cleaning for the data to remove such observations with outliers. We didn't try to input the most likely values for missing ones ourselves, as many of them were categorical and had no direct relation to other features. In addition, we still had 190454 observations left (>51% of original), so the reduced sample had a lot of data points to work on.

The used variables were:

- dateCrawled : when this ad was first crawled, all field-values are taken from this date
- name : "name" of the car
- seller : private or dealer
- offerType
- price : the price on the ad to sell the car
- abtest
- vehicleType
- yearOfRegistration : at which year the car was first registered
- gearbox
- powerPS : power of the car in PS
- model
- kilometer : how many kilometers the car has driven
- monthOfRegistration : at which month the car was first registered
- fuelType
- brand
- notRepairedDamage : if the car has a damage which is not repaired yet
- dateCreated : the date for which the ad at ebay was created
- nrOfPictures : number of pictures in the ad (unfortunately this field contains everywhere a 0 and is thus useless (bug in crawler))
- postalCode
- lastSeenOnline : when the crawler saw this ad last online

We dropped several features almost instantly due to either lack of unique values, most of which belong to only one particular value (seller, offerType) or lack of relevance to the prediction task (nrOfPictures, lastSeen, dateCreated, dateCrawled, monthOfRegistration). As for the variable 'postalCode', though it was interesting to find the influence of region, we decided to remove it to follow the suggestion from research article by Haan and Boer(2010). The latter argued that the differences between the prices of used cars based on the country regions tends to decrease more and more due to the increasing proliferation of online sales platforms' usage.

To deal with unreasonable observations we looked at the summary statistics of numerical features. The following three features in the table presented certain problems:

Summary stats	price	powerPS	yearOfRegistration
Min	0	0	1000
Max	2.147e+09	120000	9999

Mean	17295	115.55	2004
Median	2950	105	2003

As can be seen above, the minimum and maximum values for three variables are evident indicators of outliers and erroneous observations. Most certainly, car could not be registered in year 1000 or any year above 2017. It could neither have 0 PH of engine power, nor 120000. As for the prices, although technically they might be true, we avoided those observations, as most probably the extremely high or low prices cannot represent their true market values. Thus, to avoid discrepancy in the data we restricted the lower-, upper-bounds for these features. The resulting summary statistics presented as follows are more adequate:

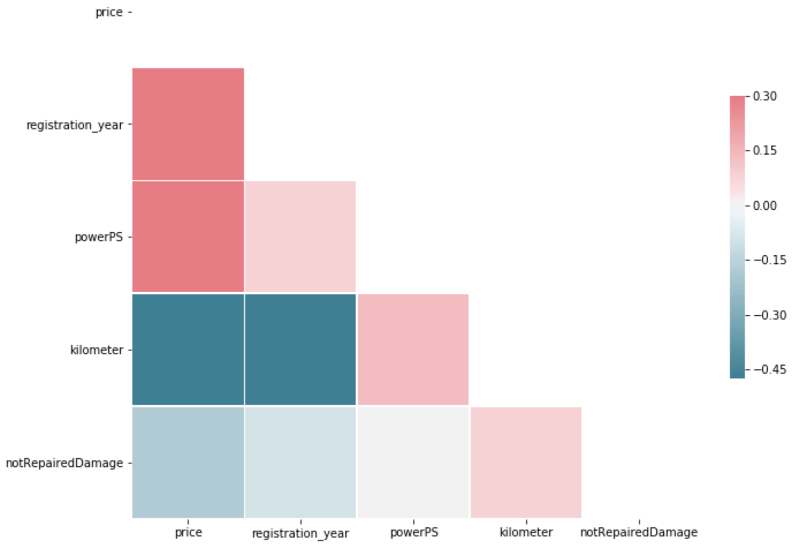
Summary stats (ed.)	price	powerPS	yearOfRegistration
Min	555	59	1975
Max	21980	227	2016
Mean	5593.3	122.2	2004
Median	3999	116	2004

2.2. Feature selection

The second issue with the rest of variables was that many of them were categorical and needed to be turned into numerical ones. Using one hot encoding method we turned the following variables into indicators (except for ‘model’), which is convenient as we received an orthogonal vector space. As for the ‘model’ variable, we altered it using label encoding instead in order to avoid high-dimensionality in the vector space.

Cat.variable	abtest	vehicleType	model	gearbox	fuelType	brand
Number of unique values	2	8	245	2	7	38

However, we decided to drop the ‘model’ feature as well to make the model more parsimonious. The reason, we believe it’s not going to give added value to prediction, as we assume that brand and the class (vehicleType) of the car already captures the differences between different cars’ prices. As concerns the rest of the variables (except for categorical), as displayed in the following correlation matrix, the features have quite significant correlation with price variable, which is why we left them for the prediction task.



III Models

As we can see, there are more variables which were originally categorical than numerical ones. By turning the categorical variables into dummies or by allocating them with values can have different impact on our model and makes the model more complex. In order to find the best model, we intend to implement both linear and non-linear methods to build our models. The methods include: Linear Regression, Ridge Regression, Lasso Regression, Ensemble Random Forest, Ensemble Gradient Boosting.

3.1. Linear Regression

Linear Regression is a linear approach for modelling the relationship between a scalar dependent variable and one or more independent variables. Linear models help to find the unknown model parameters by using linear predictor function.

A simple regression model contains independent (explanatory) variable, x_i, for i 1, . . ., n subjects, and is linear with respect to both the regression parameters and the dependent variable. The model is expressed as

$$y_i = a + bx_i + \varepsilon_i$$

where the regression parameter a is the intercept , and the regression parameter b is the slope of the regression line . The random error term e_i is assumed to be uncorrelated, with a mean of 0 and constant variance.

In our case, the output variable is the price of the car and independent variables are the features such as vehicle type, registration year, powerPS, brand and others. We created an initial model dropping the price variable and splitted the entire data into training and test set. We kept the test size to be 20% of data and allocated the random state parameter to 2.

The coefficient R^2 is defined as (1 - u/v), where u is the residual sum of squares ((y_true - y_pred) ** 2).sum() and v is the total sum of squares ((y_true - y_true.mean()) ** 2).sum(). The best possible score is 1.0 and it can be negative (because the model can be arbitrarily worse). A constant model that always predicts the expected value of y, disregarding the input features, would get a R^2 score of 0.0.

We fitted our Linear Regression models on dataset using the least square approach but they can be fitted in other ways such as Lasso, Ridge Regression which we will discuss further. We got a variance explanation of 59.62% accuracy using least square approach and then we used K-fold to check the RMSE. We initially tried K fold cross validation with K=3 and realised the best results for RMSE was with K=10. We also took into consideration, with higher value of K, computational time was increasing significantly, so we stopped at K=10.

R squared	Adjusted R squared	RSE	RMSE
0.7096	0.7092	2562.18	0.4565

3.2. Ridge Regression:

Ridge Regression penalises the size of the regression coefficient ,

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_j\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = \mathsf{RSS} + \|\beta\|_{\ell_2}^2$$

Here, the tuning parameter λ controls the relative impact of these two terms on the regression coefficient estimates. Hence, choosing a right value of lambda was critical and we used the Scikit learn implementation of kernel ridge expression. Kernel ridge regression (KRR) combines ridge regression (linear least squares with L2-norm regularization) with the kernel trick. It thus learns a linear function in the space induced by the respective kernel and the data.

From KRR, we set the value of alpha to be 1 and fitted the model on the training data to make our predictions on the test data. We did not receive any large improvement in our score from ridge regression so we tuned in to Lasso regression.

3.3. Lasso Regression

As an alternative to Ridge model, we shifted to Lasso which is similar to Ridge model as it produces simpler and more interpretable models that involve only a subset of the predictors.

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_j\right)^2+\lambda\sum_{j=1}^{p}|\beta_j| = \mathrm{RSS}+\|\beta\|_{l_1}$$

It uses an L1 penalty instead of an L2 penalty used in the Ridge model which has the effect of forcing some of the coefficient estimates to be exactly equal to zero when the tuning parameter λ is sufficiently large. We implemented the Scikit learn implementation of Lasso, and fitted the model on the training data to make our predictions on the test data.
In our case, we did not observe a huge difference in all the regression model (Linear Regression,Lasso,Ridge) so we shifted to more elaborate models like Random Forest, Gradient boosting which we will discuss further.

Limitation of Regression models:
The linear regression models assume a straight line linear relationship between predictors and response. This assumption can significantly reduce the prediction accuracy, if true linear relationship is far from linear. In our case, when we plot the residuals versus the predicted value graph, we observed pattern in the graph. In order to further improve the score, we then took log transformation of the price but it was not useful in improving the prediction score. Also as discussed above, we found inner correlation between three categorical variables which we believe had significant effect on the accuracy of our model.

For our dataset, we find that non linear models such as Random forest, gradient boosting which as discussed below have higher prediction accuracy compared to various forms of linear applying even when the penalty for variables (Lasso) is applied.

3.4 Random Forest

Random forest is an ensemble learning method that operate by constructing a multitude of decision trees sub-sample of dataset at training time and providing as output the class that is the mode of class or mean prediction of the individual trees. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement (if the option bootstrap = True (default)). In our case, we have applied two approaches to implement random forest.

The first approach we used is “make_regression” provided by sklearn, which can generate a random regression problem. The input set can either be well conditioned or have low rank - fat tail singular profile. The output is generated by applying a (potentially biased) random linear regression model with n_informative nonzero regressors to the previously generated input and some gaussian centered noise with some adjustable scale.

parameters	*n_samples*	*n_features*	*n_informative*	*max_depth*	*Score*	*RMSE*
1 (defaut)	*100*	*100*	*10*	*None*	*0.9648*	*891.66*
2	*100*	*10*	*10*	*20*	*0.9602*	*948.42*
3	*100*	*10*	*10*	*30*	*0.9649*	*891.38*
4	*100*	*10*	*10*	*40*	*0.9649*	*891.29*
5	*500*	*10*	*30*	*30*	*0.9649*	*891.38*

n_samples: The number of trees in the forest.
n_features: The number of features.
n_informative: The number of informative features
max_depth: The maximum depth of the tree.

We tried different value combinations of the parameters, however, they didn’t have a lot of impact on the results. For example, when we decreased the number of features from 100 to 10, the result didn't change significantly. One possible explanation is that we have 61 features in total and most of which are not significant, so even 10 features are sufficient for constructing an accurate model. So finally, we chose the one with (100, 10, 10, 30) which gives the best result of 0.9037 with RMSE of 1475.77.

The second approach consists of GridSearchCV, which is an exhaustive search over specified parameter values for an estimator. The parameters of the estimator used to apply these methods are optimized by cross-validated grid-search over a parameter grid. It does the searching through a manually specified subset of the hyperparameter space of a learning algorithm.

We have defined a list of parameter settings to try as values:

param_grid	min_samples_leaf	min_samples_split	max_depth	n_estimators	Score	RMSE
1	2	2	5	100	0.7548	2337.26
2	3	3	10	100	0.8546	1813.22

3	3	3	10	300	0.8546	1812.78
4	3	3	30	100	0.9005	1499.87
5	3	3	40	100	0.9004	1500.70

Min_samples_leaf: The minimum number of samples required to be at a leaf node
Min_samples_split: The minimum number of samples required to split an internal node
Max_depth: The maximum depth of the tree.
N_estimators: The number of trees in the forest.

We set cv=2 to specify the number of fold in Kfold, n_jobs = -1 to limit the number of jobs done in parallel to be the number of cores, and verbose =1 to control the verbosity.

We notice that as we increase the value of n_estimators, the score doesn’t improve, but the training process becomes much more time consuming. The time used increased from around 40 seconds to 12 minutes for n_estimators values from 100 to 500. This is because calculation volume should be immense in order to do the exhaustive research of high dimension with many times of cross validations.One way to reduce computation expense is probably using RandomizedSearchCV.

When we increase the max_depth, we can have a much better score. Even if we decrease the value of n_estimators while keeping a high max_deep, the result is still plausible.

The reason why we applied this approach is that we want to find a more efficient method to tune the parameters. GridSearchCV allows us to define a set of parameters that we want to try with a given model and will automatically run cross validation using each of those parameters keeping track of the resulting scores. We have noticed that the most important parameter for both approaches is max_depth which define the depth of the decision trees. However, a too high value of depth may cause overfitting. With the same optimal value of depth (=30), the first approach still outperforms the GridSearchCV by almost 6%. The reason is probably that we didn’t perfectly define the list of parameters with an appropriate range.

3.5 Gradient Boosting

The following is the best description of Gradient Boosting by Hastie et al.(2009):

“Gradient Boosting is a technique which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It forms the model in a stage-wise fashion, and generalizes them by allowing optimization of an arbitrary

differentiable loss function. The idea of gradient boosting is that boosting can be interpreted as an optimization algorithm on a suitable loss function. Like other boosting methods, gradient boosting combines weak "leaners" into a single strong learner in an iterative fashion. In each stage, a regression tree is fitted on the negative gradient of the given loss function, and the loss function is gradually improved at each iteration as models with better performance will be allocated of more weight."

In our case, we have found that gradient boosting method is very time consuming compared to other methods. This can be understated as the model is iterative thus multiplied and complex. We have tried several parameters, and the corresponding results are illustrated below:

parameters	*n_estimators*	*max_depth*	*min_samples_split*	*score*
1 (Default)	*100*	*3*	*2*	*0.8543*
2	*200*	*3*	*2*	*0.8654*
3	*200*	*5*	*2*	*0.8678*
4	*200*	*5*	*4*	*0.8781*
5	*500*	*6*	*4*	*0.8939*
6	*700*	*10*	*4*	*0.90643*

n_estimators : The number of boosting stages to perform. Gradient boosting is fairly robust to overfitting so a large number usually results in better performance.
max_depth: maximum depth of the individual regression estimators. The maximum depth limits the number of nodes in the tree.
min_samles_split: The minimum number of samples required to split an internal node:

By increasing the value of each parameters, we can get a better score. However, we find that n_ estimators and max_depth are more influential than min_samples_split. And as we increase the n_estimators, the time of calculation increases rapidly. The 5th try had spent us about 10 minutes. And after then on the performance doesn't improve significantly. So we finally stopped at the 6th try, where n_estimators = 5, max_depth = 6 and min_samples_split = 4.

Comparison of random forest and gradient boosting

Both of them are improved decision tree methods which are members of ensemble learning, they use multiple "weak learners" to form a stronger one.

Random Forest use bagging method which is designed to improve the stability and accuracy, reduce variance and helps avoid overfitting. Gradient Boosting uses boosting methods which is designed to primarily reduce bias and variance in supervised learning.

Random forest fits many independent trees against different samples of data and average together. However, Gradient Boosting fits consecutive trees where each solves for net error of the prior trees, thus trees are dependent. (Mark Landry, 2015).

The advantage of random forest is that it can handle the missing values and maintains accuracy for missing data. it won't be overfitting the model and can handle large data set with high dimensions. However the downside of this method is that it can not give predictions beyond the range of training data, and they may overfit the data set that are particularly noisy. Also, we have little control of what the model does.

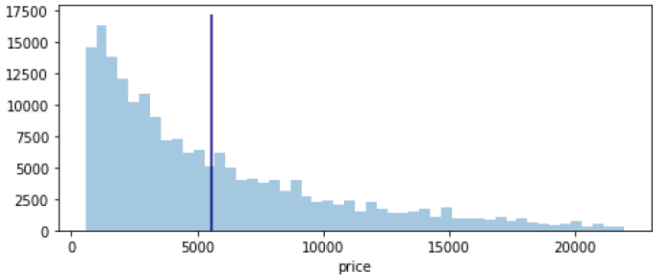
The advantages of Gradient Boosting are that it is robust and directly optimizes the cost function. However, it may cause overfitting thus needs to find proper stopping point. Also it's sensitive to noise and outliers.

In our case, Random Forest gives the best accuracy which is 0.96 versus the GB method's 0.90 and is less time consuming then the latter one. However, according to the research we have done, it is usually the GB methods outperform the random forest.

We have several probable explications for this result. One is that GBT have a few hyperparameters to tune, while owing to the specifics of our dataset the random forest is practically tuning-free, and this is confirmed by our training process. As we have a high dimension dataset, and both of the models lack of transparency, we had difficulties to calibrate the parameters of GB. For the time issue, as the GB iteratively generates the trees, when the depth of the tree is large, we can imagine that it need huge amounts of time to process.

Model limitations and evaluations

Price distribution:



We have several problems before building our models. Firstly, the target prices don't follow a normal distribution, instead, they are very right skewed. 85% percent of the values are concentrated in the range of [0,10000], while the whole range is

[0, 20000] after dropping numerous outliers. This is understandable because there exist luxury car which are extremely expensive.

So we also tried to separate the cars into “economic_car_group” and “lux_car_group” (lux stands for luxury) so as to apply different models with different important features. However, this didn’t help for the model as the accuracy is lower than before. This is maybe because we lack some sufficient background knowledge of car industries so we didn’t manage to choose the most relevant features for each group to fit a more efficient model.

Categorical variables:

Another issue is that there are more categorical variables than numerical ones. As the correlation matrix and feature importance score showed, all the numerical variables have a relative high correlationship with price, while the categorical variables have less importance. Meanwhile, this could be an illusion because after turning them into dummy, we get 57 dummy variables, which increased largely the dimensionality.

However, among the categorical variables, there are some inner correlated ones, such as “VehicleType”, “Brand” and “model”. We can imagine that each of the three features has narrow relationship with the other two, but each one can give additional information, and it’s hard to do manipulations on them such as addition, division etc.

For the feature “model”, we applied “one hot encoding” by giving them values as 1,2,3… for the reason that there are more than 150 different labels. But this might lead to misinterpretations during training because the values can not compare with each other.

Also, there are some categorical variables like “gearbox”, “abtest” on which we lack of knowledge but seem like not negligible. So we kept all of them (except the ones that we are sure of their irrelevance) in our model. This might have complicated our models and led to an insufficiency of accuracy for certain of them.

Models:

There are more detailed discussions in the methodology section for each model applied.

Here illustrates a summary of the results of each model:

Model	Accuracy	RMSE	NRMSD
Linear regression	0.70962	2562.17	0.4565

Ridge	0.70962	2562.19	0.4565
Lasso	0.70962	2564.37	0.4569
Gradient boosting	0.90643	1813.49	0.323
Random Forest	0.96494	891.38	0.1588

The one which perform best is the Random Forest, and both the ensemble methods based on decision tree outperformed linear models because of the nature of the features. We have experienced some difficulties when tuning the parameters for Random Forest and Gradient Boosting, because we don’t know them perfectly and it was time consuming for each run. As both models are sensitive to noisy data, some inappropriate manipulations in data processing phase may cause imperfections in results.

IV Conclusion:

We have applied several methods for our modeling in order to predict the selling price of used cars according to their attributes. Random Forest and Gradient Boosting provided plausible results after trying several combinations of parameter values. However, we have realised that efforts could still be done to improve our feature selection and hyperparameter tuning as well as the running time. In our further study, we will try to apply, for example, RandomizedSearchCV approach instead of GridSearchCV to reduce computation expense. Another possible direction for improvement could be to tune the parameters of each model more profoundly.

The high accuracy scores for Random Forest and GB were not only good indicators for prediction, but they also demonstrated that the features that we had in our model give a high explanatory power of the prices. Despite having a black-box model we believe we were able to achieve the main goal.

V End Notes:

1. Brockett, Patrick L., Xia, Xiaohua, and Derrig, Richard A, 1998, Using Kohonen’s Self Organizing Feature Map to Uncover Automobile Bodily Injury Claims Fraud, Journal of Risk and Insurance, 65:2 245-274.
2. Viaene, Stijn, Derrig, Richard A., Baesens, Bart and Dedene, Guido, 2002, A Comparison of State-of-the-Art Classification Techniques for Expert Automobile Insurance Claim Fraud Detection, Journal of Risk and Insurance, 69:3 373-421.

3. Manheim Used Car Market Report, https://publish.manheim.com/content/dam/consulting/2017-Manheim-Used-Car-Market-Report.pdf
4. Here's a Good Indication of How Much People Hate Car Dealerships (http://time.com/money/3826562/buying-cars-online-hate-car-dealerships/)
5. Which Is the Most Painful Shopping Experience of All? (http://business.time.com/2010/10/14/which-is-the-most-painful-shopping-experience-of-all/)
6. Will Millennials Change How Cars Are Bought and Sold? (http://business.time.com/2012/08/09/will-millennials-change-how-cars-are-bought-and-sold/)
7. Used Car Database (https://www.kaggle.com/orgesleka/used-cars-database)
8. Haan, M. and de Boer, H. (2010). Has the Internet Eliminated Regional Price Differences? Evidence from the Used Car Market. *De Economist,* 158(4), pp.373-386
9. Hastie, T.; Tibshirani, R.; Friedman, J. H. (2009). "Boosting and Additive Trees". *The Elements of Statistical Learning* (2nd ed.). New York: Springer. pp. 337–384