CLEANING STEPS

1. First of all, since strings are case sensitive, I converted all factor strings to uppercase, in order to standardize the strings.
2. Since the data file is in German, I had to replace some column names and elements with English translations.
3. Then I wrote a function to fill missing brand data by looking at name column (which is actually the title of the ad inserted by the advertiser - can be very arbitrary), and retrieve brand information if brand element of the same row is NaN.
   This proved to be in vain because brand column has no missing value!
4. Then I dropped "offerType", "postalCode", "nrOfPictures", "dateCreated", "lastSeen" columns which I believe will be absolutely unnecessary when developing a predicting model, since they are no factor in price.
5. Then I developed an algorithm for filtering the outliers. There are known luxurious brands that are very expensive. If a price value is on the luxurious range, but the brand is a regular brand, this is an erroneous row. I developed 3 stage filter to filter out all the unwanted data in this case.
6. I also had to write code to address the problem where all model names starting with “Grand” were only named “Grand”, therefore “Grand Cherokee”, “Grand Vitara” and “Grand Caravan” all have the same model name:”Grand”. This had to be corrected.
7. There were also entries with price 0, or 1 €, which are obviously erroneous. I deleted those data points together with those less than 100€.
8. Some of the ads are not about selling the car, but renting it monthly. These ads are around 500€, all the way up to 1000€. It is hard to detect if ads at this price range are rental or real preowned data. If the vehicle is old and/or has had an accident these values may be of value. Developing a filtering algorithm in this case needs attention and actually it could be smarter to remove those rows from the beginning. I decided to remove all vehicles with price less than 500.
9. I generated brands and models list from unique values and wrote a function for populating a brandsmodels dictionary where the function finds all entered model values for the corresponding brand.
10. Some brand names are not entered instead they are grouped under “sonstige autos” name, which means “other cars”. At first I thought it would be very strenuous to retrieve data from name element since model info was also missing.Therefore data points that had “sonstige autos “value had to be deleted. However, later, I decided to have another approach, since there were considerable number of American vehicles, and in order to keep that data I augmented my coded dictionary by manually adding missing brands that made up “other vehicles” and their models. Later, I used this augmented dictionary to fill missing values, which are “other vehicles(sonstige_autos)”, “others (andere)”, and simply null values.
11. Some of the engines have lower than expected horsepower - as low as 0-, and some are unrealistically high. These have to be removed.

12. Registration year of the car value is as low as 1800, and some values are higher than 2016, which suggest that these values are wrong because the dataset was created on 2016. These erroneous values have to be removed too. I used a two stage filter: first one removing data points with registration date later than 2016, and second stage comparing data crawl month to registration month and removing erroneous data for 2016 vehicles. Also, registration dates lower than 1950 have been removed.
13. I also thought that retrieving the trim data, engine capacity, and other strings of importance would be valuable. This, however, exceeds the limits and scope of this project and requires data mining algorithms and NLP.
14. After the code written to retrieve data from the name column to fill missing values in brand and model columns is executed, there will be still some missing values. These are in transmission, vehicle type, fuel type. There are no intermediate values for these columns,so interpolation would not work. Forward or backward filling would create erroneous data. I reached a decision that omitting those rows is the best for the sake of the model.
15. Missing values at vehicle type column at first appears like fillable through coding, since a line of code like this would reveal the most prominent type for each model:
    df[df.model=='CIVIC'].groupby("vehicleType").count()

vehicleType	dateCrawled	name	seller	price	abtest	yearOfRegistration	gearbox	powerPS	kilometer	monthOfRegistration	fuelType	notRepairedDamage	br
andere	12	12	12	12	12	12	12	12	12	12	12	9	
bus	1	1	1	1	1	1	1	1	1	1	1	0	
cabrio	10	10	10	10	10	10	9	10	10	10	9	9	
coupe	168	168	168	168	168	168	161	168	168	168	153	131	
kleinwagen	240	240	240	240	240	240	236	240	240	240	218	179	
kombi	36	36	36	36	36	36	35	36	36	36	34	31	
limousine	664	664	664	664	664	664	653	664	664	664	645	587	

    However, there is a problem here: The model selected for EDA here is Honda Civic and this vehicle, just like many other similar brands and models, has many body types. A vehicle can be coupe, sedan or station wagon. The only way to figure out what the actual type of the vehicle should be is to have the trim information and a dictionary of model-trim to vehicle types. As discussed previously, this process does not fall within this course's remit at his time. Rows with missing values in this column can be omitted or this column can be removed completely. I decided to omit all missing data.
16. Once all erroneous data points and non-fillable missing values have been deleted, I have 232 thousand data points remaining out of 377 thousand. This is approximately 62% of the original data. At first glance this may seem to be too much data loss, however, keeping in mind that this dataset is derived from a car sales ads on a website, and that was not created by professionals, but by individuals who enter data arbitrarily, the amount of data loss is acceptable.