Big Data - Used Cars

CONTENT\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* Introduction
* Aim
* Data Acquisition
* Data Pre-processing
* Data Visualisation
* Price Estimation with WEKA
* Conclusions

# Introduction:

Very recently my family has bought a used car from the internet. We have gone through a lot of debate over the quality of the cars related to the price, and I think we viewed in total over 200 ads until we made a decision. We were always arguing about what aspects matter more, whether it’s the year of fabric, the mileage, if the car was involved in accidents or what not. This was a futile discussion since nobody really knew the right answer. Buying a used car feels like lottery sometimes, and you don’t know if you are wasting your money or having a great bargain. A tool that could tell you how good a car is, by its statistics, related t its price, would save people a lot of time when buying a used car and would help them feel better about how they spend their money.

# Aim and Objectives:

What I am trying to do is gather data about past used cars that have been sold online. I will use that dataset to explore some information and look for what makes a used car sellable. Hopefully this exercise has the capacity to make uninformed discussions about the quality of a used car less prevalent.

**OBJECTIVES:**

* Gather the data
* Analyze the attributes and whether they are important to what we are doing
* Eliminate bad data
* Visualise in order to explore the attributes and how they link
* Create a regression model to give best price for future used cars

# Gather the Data:

Unfortunately not many car sale websites are willing to give their data away, probably because they are using hidden attributes to better estimate the quality of a car, in order to rank them and decide which of them should take the first page. However, I found a free dataset that is available online and it comprises of 300.000+ instances of cars sold through ebay in Germany. (data 2016)

Further on, we will analyse the attributes and see if we can do any alterations that would better suit our aim.

# Attribute analysation

All initial attributes:

1. dateCrawled: when the ad was first crawled
2. name: title of the ad
3. seller: private or dealer
4. offerType: no documentation
5. price: the price of the ad to sell the car
6. abtest: no documentation
7. vehicleType: several buckets that represent the type of the car
8. yearOfRegistration: when the car was first registered
9. gearbox: No documentation
10. powerPS: power of the car in PS
11. model: model of the car
12. kilomenter: mileage of the car
13. monthOfRegistration: in which month the car was registered
14. fuelType: diesel, oil
15. brand: brand of the car
16. notRepairedDamage: if the car has a damage which is not repaired yet
17. dateCreated: date when ad was created
18. nrOfPictures: nr of pictures in the ad
19. postalCode:
20. lastSeenOnline: when the crawler seen this ad last online

Since most deals usually take place over the phone or email, we have no guarantee that all the cars were sold, but for the sake of the project and the free dataset, we will assume that lastSeenOnline will represent the date the car was sold.

Also, some negotiation usually takes place and the price shown is not guaranteed to be the price the car was sold for. However, we will again make an assumption the price is correct, because negotiations can only really lower the price by a couple of hundreds max.

**FIRST STEP:**

When trying to upload the dataset to WEKA, we get errors that mostly have to do with the column name. I immediately decided that the column name is of no use to my aim, since it only represents a string of characters inputted by the ad owner, and all of them are distinct. So we eliminate the column name.

**Python:**

**import csv**

**#from datetime import date**

**f=open("c:/Student/220CT/Task4/autosData/autosTest.csv",'r')**

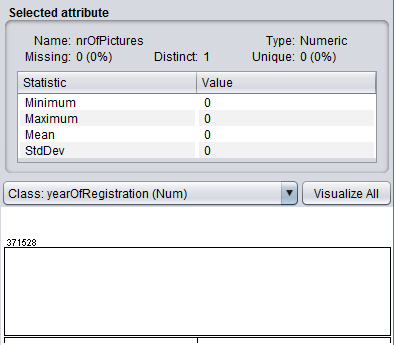
**reader=csv.DictReader(f)**

**l=list(reader)**

**for doc in l:**

**doc.pop("name",None)**

**STEP TWO:**

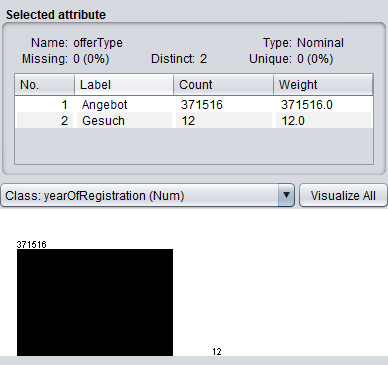
Now that we can load our dataset into WEKA, we are going to analyse each attribute and see what values it gets.

We immediately spot that nrOfPictures is always 0!

This means this column is useless so we can eliminate it!

**PYTHON:**

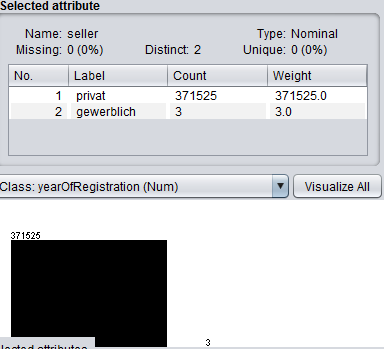
**doc.pop("nrOfPictures",None)**



We now check offerType: There are only two distinct values (sell, buy). Only 12 people want to buy a specific car, and they do not reflect our aim. Therefore, we can scrap this attribute too since all other cars are for sale.

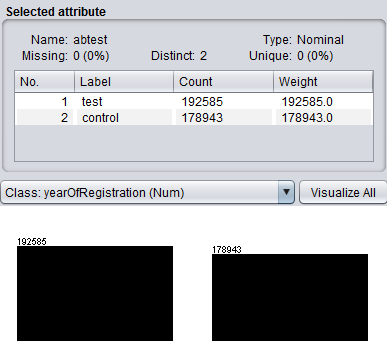
**Python:**

**doc.pop("nrOfPictures",None)**

****The seller attribute is in the same situation as typeOffer. Therefore, we don’t need this column as well. Let’s scrap it!

**Python:**

**doc.pop("seller",None)**

****The abtest attribute is well split among 2 values. However, we have no documentation on it and we don’t know what it means, so we might as well just scrap it!

**Python:**

**doc.pop("abtest",None)**

We got rid of 5 attributes so far: name, nrOfPictures, offerType, seller, abtest.

We have many dates in our dataset, all of those attributes having a lot of distinct values which can screw up our models. We have to do something about it. I decided that we do not need to know the dateCrawled attribute, since it doesn’t relate to the instance very well. lastSeen and dateCreated represent the ad lifespan. If we subtract dateCreated from lastSeen, we will get the average amount of days the car was up on the site. That is a better statistic than 3 columns with datetimes on them.

**Python:**

**created=doc["dateCreated"][:doc["dateCreated"].find(" ")]**

**seen=doc["lastSeen"][:doc["lastSeen"].find(" ")]**

**aDate=list(map(int,seen.strip().split("-")))**

**bDate=list(map(int,created.strip().split("-")))**

**a=date(aDate[0],aDate[1],aDate[2])**

**b=date(bDate[0],bDate[1],bDate[2])**

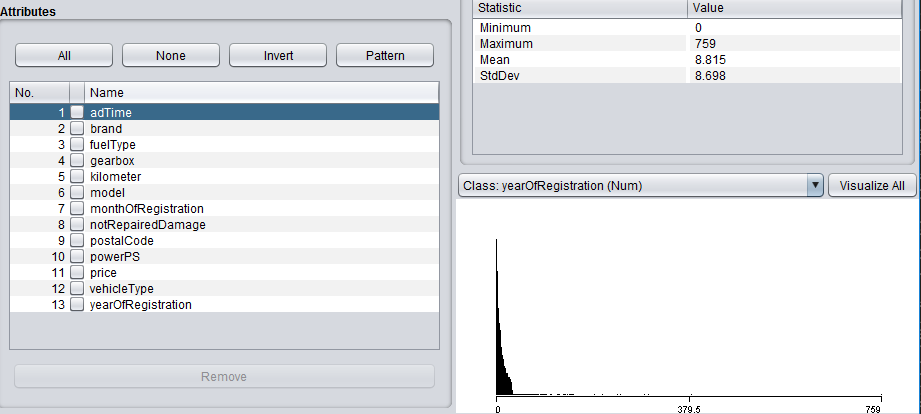
**doc["adTime"]=(a-b).days**

**doc.pop("dateCreated",None)**

**doc.pop("dateCrawled",None)**

**doc.pop("lastSeen",None)**

After applying all these python code to our initial csv file, we create a new csv file, upload it to WEKA and see what it looks like:



This is now our dataset.

**STEP THREE:**

Now, we have to look through the data and detect all the values that just don’t make sense and try to normalize it a bit.

We currently have 371528 instances, however we have to filter out all instances with missing values.

First we take the attribute with the most missing values, which is notRepairedDamage with 72000 instances missing. We have to eliminate these. We will use WEKA.

In WEKA we use the filter RemoveWithValues and set parameters -S -1.0 -C 8 -L first-last -V -M. We now have 299468 instances.

We will now remove from vehicleType the missing values in the same way and we will have approx. 282000. We do the same process until we have no more missing values and we end up with: **260900** instances.

We will now look at each attribute and remove very strange or isolated values.

* For adTime, we have several instances with more than 200 days. Those are very isolated case and we do not need them. We will remove them with the RemoveWithValues filter, this time mentioning we want to remove all instances over a certain value. We eliminated 6 instances, and now our maximum adTime is 150 which is reasonable
* For kilometre, the values range from 5000 to 150000 which is perfectly reasonable for used cars so we will not change anything
* For powerPS, we have removed all instances less than 40 and higher than 600, since they don’t make any sense.
* For prices, we have removed all instances with price lower than 50 and higher than 60000
* For yearOfRegistration, we have removed all instances with values less than 1980

We now have **244897** instances, which means more than 100k were removed. However, our dataset is now much better for analysis since we eliminated the outliers and the missing values. We are now ready to visualise some data.

# Data visualisation:

I am going to use Excel to visualise some of the data. I have inserted all the data into a MongoDB collection and I am going to use python to query the data I am concerned about and then plot the graphs using Excel.

1. We are first going to plot the price against the number of cars with that price, to see how this line is distributed.

Python:

cursor=db.usedcars.aggregate([

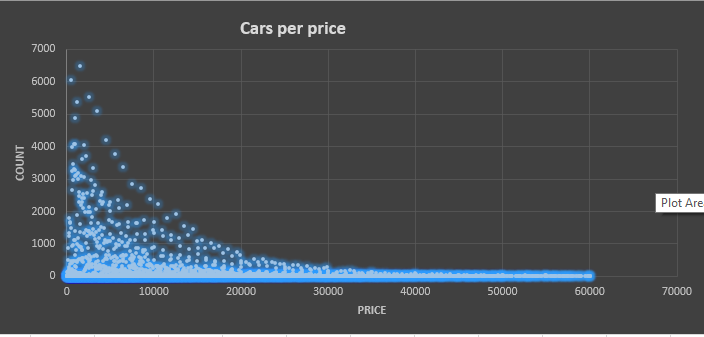
{

"$group":{

"\_id":"$price",

"count":{"$sum":1}}}])

Plot:



We can see that more than 60% of the cars are below 10000, which is very common for used cars.

1. We are going to calculate the average price for each brand and then plot that data.

PYTHON:

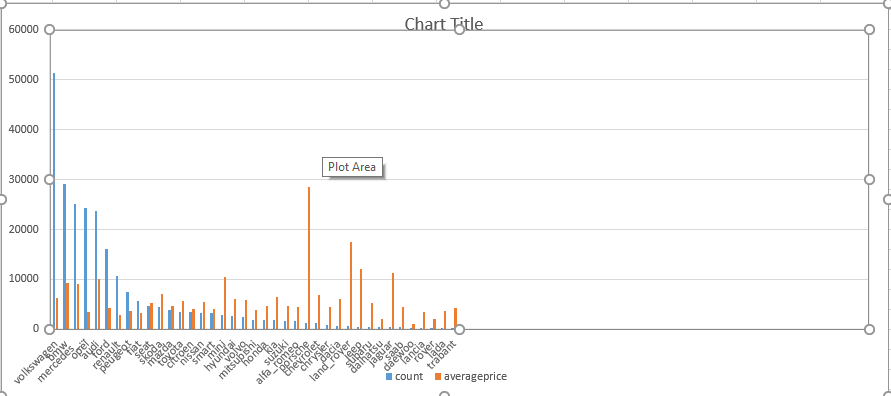
cursor=db.usedcars.aggregate([

{

"$group":{

"\_id":"$brand",

"averageprice":{"$avg":"$price"}}}])

Plot:

The most expensive average is Porsche, however theres very few Porsches on sale…the most cars are Volkswagen and they have a small average price.

1. Average price by year of Registration.

Common sense would say this should be a straight linear growing function, since you expect older cars to be cheaper than newer cars.

PYTHON:

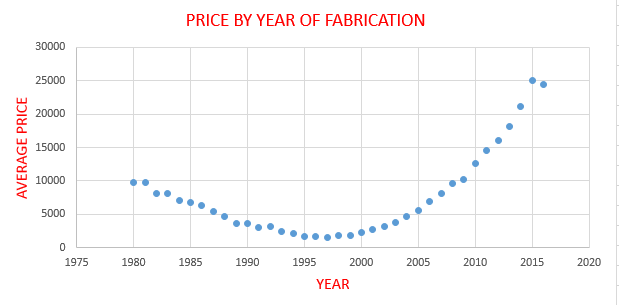
cursor=db.usedcars.aggregate([

{

"$group":{

"\_id":"$yearOfRegistration",

"averageprice":{"$avg":"$price"}}}])

PLOT:

It seemed fairly logical that newer cars would be more expensive than new cars, but surprisingly, cars older than 1990 are on average more expensive than cars of 2000!

1. **Number of cars by year of registration**

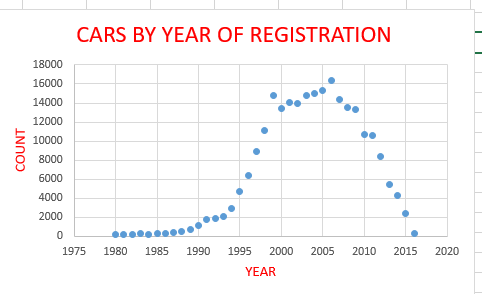
We expect to see a bunch of cars dated 2005+, and fewer that are older.

PYTHON:

cursor=db.usedcars.aggregate([

{"$group":{"\_id":"$yearOfRegistration","count":{"$sum":1}}}])

PLOT:



Indeed, we can see that most cars are from 2005-2007. There are more than 3000 cars made in 1990 or before which is impressive!

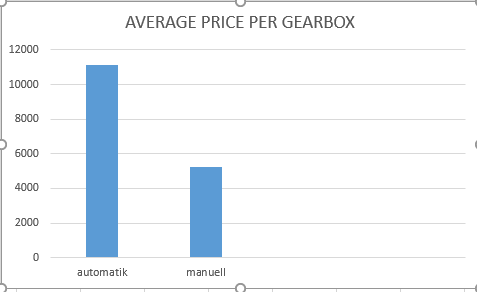
1. **Average price for cars with automatic or manual gearbox**

PYTHON:

cursor=db.usedcars.aggregate([

{"$group":{"\_id":"$gearbox","avg":{"$avg":"$price"}}}])

PLOT:



Not a surprise that automatic cars are more expensive than manual, because they are newer.

1. **Average ad time length for cars that were damaged and cars that were not**

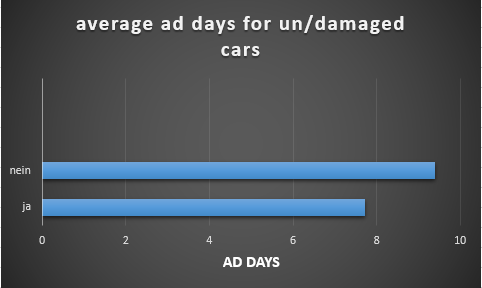
I had a feud with my mother when buying our car, she said that cars that have been damaged are not worth buying! Hopefully this graph will show her this is not an important factor

PYTHON:

cursor=db.usedcars.aggregate([

{"$group":{"\_id":"$notRepairedDamage","avg":{"$avg":"$adTime"}}}])

PLOT:



It seems like it doesn’t actually matter. Cars that have been damaged only average 2 days more until they are being sold. That is not a big gap, especially taking in consideration there are many more damaged cars than not damaged.

1. **Numbers of cars per brand**

Mouse over the areas to see which brand represents which area. Being a german website, clearly Wolkswagen, BMW, Mercedes and AUDI represent the biggest percentage of the data set, but opel also comes in with 10% which is very good.

1. **Cars sold per day, by brand**

This is the most ambitious graph so far, each line represents a brand and it is plotted on x-axis – number of days past since the ad was posted and y-axis: total number of cars sold. Because most cars are sold in the first 20 days, I selected only those days, so that we can see variance.

PYTHON:

cursor=db.usedcars.aggregate([

{"$match":{"adTime":{"$lt":20}}},

{

"$group":{

"\_id":{"brand":"$brand","adTime":"$adTime"},

"count":{"$sum":1}}}])

**import** csv

keys=["adTime","brand","count"]

**with** open('autosQuery12.csv','wt') **as** f:

dict\_writer = csv.DictWriter(f,keys)

dict\_writer.writeheader()

**for** doc **in** cursor:

doc["adTime"]=doc["\_id"]["adTime"]

doc["brand"]=doc["\_id"]["brand"]

doc.pop("\_id",None)

dict\_writer.writerow(doc)

f.close()

**PLOT:**

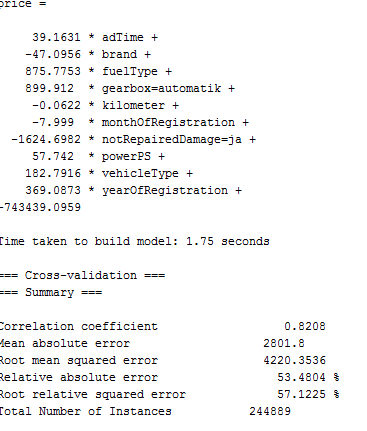
Volkswagen is by far the most sold car on the german ebay, but this graph strongly confirms that brand is one of the most important factors in determining if a car will be sold or not. We can see there are no significant intersections between the lines, meaning that this leaderboard stays constant during the days the ads are up.

# FUTURE WORK -PREDICTING BEST PRICE:

This dataset could be applied to estimate the best price a car should be sold for. The attributes can be used to form a model which would estimate the value of your car. I will try to implement this with WEKA, but for that I have to use a regression model and I have to convert all nominal attributes to numeric. We will do a trick, where we manually replace all nominal values with indices from 10 to x and then save is as an .arff, then open the .arff with notepad and edit the column definitions from a set of nominal values to numeric. For the attributes with only 2 nominal values, we can use the filter nominalToBinary to make it have 0 and 1.

Note: We remove some of the attributes, namely model and postcode.

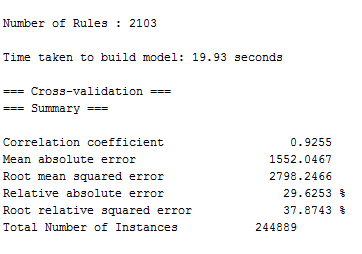
Now that we have all numeric attributes, we can try to form a linear regression that would estimate the price of a car. We will use cross-validation 10 fold for our tests. Our initial test is:



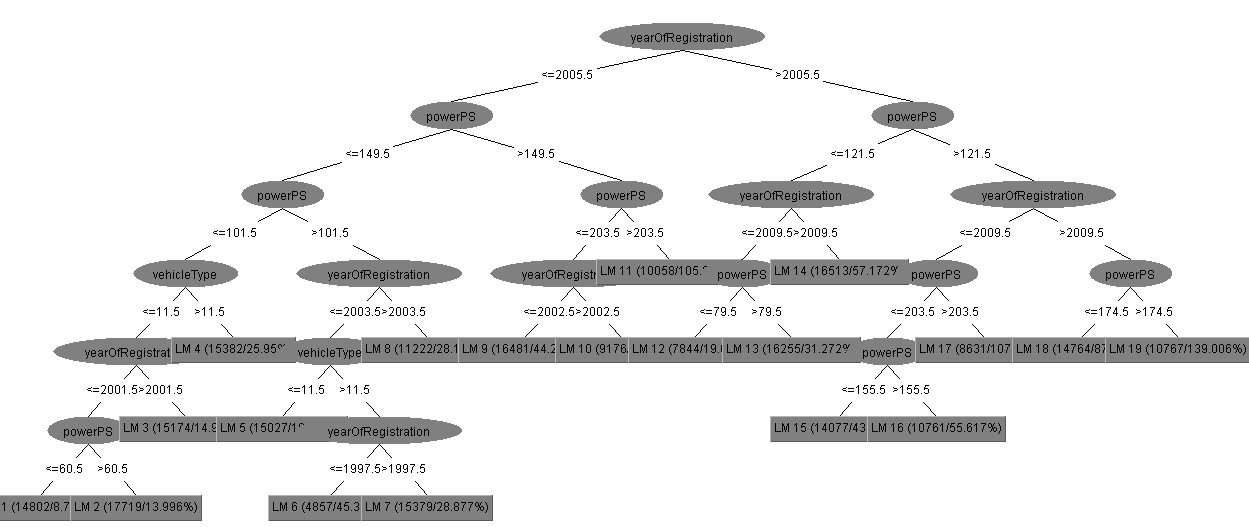
With a few alterations, we might be able to improve these results. A mean error of 4200 is pretty bad for price estimation, but that might be due to the really high values in price fro some of the cars.

Lets also remove adTime (it is not relevant to price estimation, since we only care if cars were sold, not how long it took) and monthOfRegistration (the year is more important).

We will now try to build a regression tree, we will allows 100 instances per leaf to have a smaller tree:



With 100 instances per leaf allowed, we get a whopping 2100 rules, but we have a much improved result with only 2800 root mean error. However the tree is too big to visualise. I will try one more time with 20000 instances per leaf and we get the following tree:



As we can see, because we allowed so many instances per leaf, the tree only has 19 rules and most of them have to do with yearOfRegistraton and Horse Power. We get an absolute mean error of 4200.

# Conclusion:

The model may not be perfect, but there is potential here to create a very good price estimator that would be an excellent tool for potential sellers and it would attract many customers.

The dataset can also be used by a company to carefully select which cars to use in their ad campaigns. They can choose the really successful ones that sell in the first couple of days, or they can choose the very select ones, with very good stats and expensive. The querying of the data is very straight-forward with mongoDB

mongoDB also offers excellent opportunities to store your data to an almost infinite amount. This valuable data data can then be used to further increase the accuracy of the price estimation model.