Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

Is the Price Right

Bachelor thesis

Christopher Signer

signerc@student.ethz.ch

Distributed Computing Group
Computer Engineering and Networks Laboratory
ETH Zürich

Supervisors:
Laura Peer, Philipp Brandes
Prof. Dr. Roger Wattenhofer

March 23, 2016

Acknowledgements

I thank Laura and Philipp for supporting me with this project. They took their time to meet up regularly and discuss the findings and especially gave input on how to progress when I struggled or the results were unsatisfactory. I would also like to express my thanks to Prof. Dr. Wattenhofer, who has built up a diverse research group which also tackles everyday problems. He held a course on distributed programming which inspired me to look into projects of his group.

Abstract

While some people go to car dealerships when they are interested in buying a car, other people prefer to comfortably browse countless car listings on a number of available Internet platforms, for example autoscout24, car4you or autoportal24. At the time of writing, the online car portal autoscout24 boasts over 150'000 car offers. The users who visit these online car marketplaces can narrow their search by applying specific criteria, for example brand, model, make or year, but finding outstanding offers in this large collection of results is a tedious task, which is currently done manually and by the user's own recollection.

The goal of this thesis is to crawl car offers from online resources and use the gathered data to develop a model that estimates the price of a car based on several key indicators. The result of this thesis is a web application that allows the user to specify a car with specific attributes and returns a selection of similar offers and an estimated price.

Contents

Chapter 1

Introduction

Switzerland has about 5.9 million registered civilian vehicles[1]. Many of these cares are resold and find a new owner. To prevent car dealerships from taking a percentage of the selling price, online platforms and other publications allow or even promote advertisements of private car listings. The prices are set by the seller, which can potentially be a private person who does not have much experience selling cars. How can an inexperienced seller figure out a reasonable price, and how can a potential buyer check if the suggested price is appropriate?

In both situations, the other car listings provide a lot of information. See another five listed cars which are similar? Compare their prices and set yours accordingly. But what makes the cars comparable? Does the color matter? Or can less mileage make up for being a year older? Especially with cars that are not too common, it might suddenly take a lot of time or even prove impossible to find similar cars according to the price deciding factors.

1.1 Related Work

Comparis[2] has a car search tool with listings from multiple Swiss websites. They host them on their website and give a calculated market value as comparison to rate the price. They also offer estimating the price of a car based on a set of basic information that includes mileage, licensing date and information to identify the car. Either the type number (manufacturer number) or car brand, model and car type (an accumulation of multiple factors as for example horse power, number of doors) can be provided. The estimated car price is based on a statistical model, credited to a former ETH scientist Andreas Keller[3]. There is no more detailed information given besides that they are actively collecting more data and that they cannot take into account special conditions and optional equipment in detail.

The Kelley Blue Book[4], autotrader.co.uk[5], carsales[6] and a variety of other websites offer to estimate the value of a car. Some base it on a set of basic information and others on detailed forms that include optional equipment, color and more. Some websites (for example eurotaxglass[7]) even offer their price estimation as a paid service. None of the mentioned services go into detail on how their price estimate was derived.

1.2 Objective

The objective of this thesis is to create a tool that can estimate the prices of cars and return similar cars based on the features found in car listings to help buyers and sellers. At first information on existing car listings is collected. In a second step, the information is evaluated to create a price estimator based on features and parameters which also returns similar cars. As final product of this thesis we integrate the estimator into a simple user interface.

Chapter 2

Price checker

2.1 Dataset

We were unable to find a publicly available list of secondhand car purchases. Therefore, car offers from people who intend to sell their cars are the most obvious and easiest accessible sources for data on cars. There are many online platforms where secondhand cars can be sold in Switzerland. The largest among them are Ricardo[8], car4you[9] and autoscout24.ch[10]. They offer a choice of 60'000 to almost 120'000 cars. We collect data from the largest portal: autoscout24.ch.

Since Germany has a large secondhand car market, we considered using their online platforms as another source to gather a larger number of car data. The large data set would allow to price check German cars and could perhaps offer more data on uncommon cars and help with cross validation. Among the largest with about 1 and 2 million offers are mobile.de[11] and autoscout24.de[12], respectively.

Unfortunately, autoscout24.ch and autoscout24.de do not offer an API which lists car offers and mobile.de sells access to their API for commercial use only. Crawling the websites to extract the car offers from the HTML pages is thus the chosen approach.

For each of the three car selling platforms, we implemented a crawler which finds the car offers by searching for cars of all models and traverses the search results. This provides a definite classification of the brand and model of each search result on the granularity used by the websites. Search results, among other things, consist of links to car detail pages. For each car detail page we store the vehicleID, a unique value given to every car listing on each website, which later allows access to the car offer for as long as it is online. All the information given on the detail websites for car offers is then checked for possible user input errors and then stored in a database. The database schema is given in Figure 2.1 and shows all of the car properties stored. Notably, every time we find a preexisting car, we add an entry to the table *times*. This provides a time window for every listed car to determine how long it was online, until it is either sold or taken down due to being unsuccessful. The *buckets* table stores group names of car model versions as will be explained in more detail later.

2.2 Car Features

The car features can be separated into two types: Categorical features and continuous features. Categorical features contain boolean features which are "all or nothing". A car has for example either a navigation device or it does not. The car's equipment (navigation, ISOFIX, xenon headlights, etc.), is thus transformed into categoric features. The brand and model are also considered to be categories. The equipment features are sorted by their number of occurrence in Figure 2.2.

The continuous features are numerical columns in the database which theoretically have an infinite number of possible values each, for example the mileage, car price and fuel consumption or the licensing date, which is turned into the number of months since the current date to get a value that can be processed well. The logarithm is taken for licensing date and mileage as the price decreases non-linearly for these values.

The car model is arguably one of, if not the most important feature. There is a huge variety of car types from luxury cars, sports cars, family vans, to station wagons and more. The model describes the car type, car brand and often many features such as approximate production year and engine. A car model is given in the search form of car selling platforms but only at a coarse granularity. A VW Golf with a licensing in 2008 could be any of about sixteen versions (e.g. 1.4, 1.4 TSI, GTI, 2.0 TDI, etc.) and this is only out of the then current sixth generation of this specific car model. The VW Golf in question might still be a car from the 2007 product generation. A search box for the edition allows the user to narrow it down more, but it is a text search on whatever description the seller deemed important. More often than not, it might include irrelevant words to advertise the car.

The mileage of a car is a good indicator for the condition of a car. A more used car can potentially require more repairs and is more likely to break down and often shows more signs of use like dents, scratches and general uncleanliness. Figure 2.3 shows a log-log plot of the mileage and price, in comparison to 2.4 which is a semi-log plot but with the logarithm of the mileage on the x-axis. The mileage also correlates strongly with the licensing. The licensing date indicates the age of a car and as such is shown as relevant in Figure 2.5. An older car once again is more costly to maintain. Generally, the licensing date correlates as mentioned with the mileage but also with the car model and thus indirectly with many other features. Electric cars were only introduced in recent years, and as such are not contained in cars that are for example over twenty years old. The

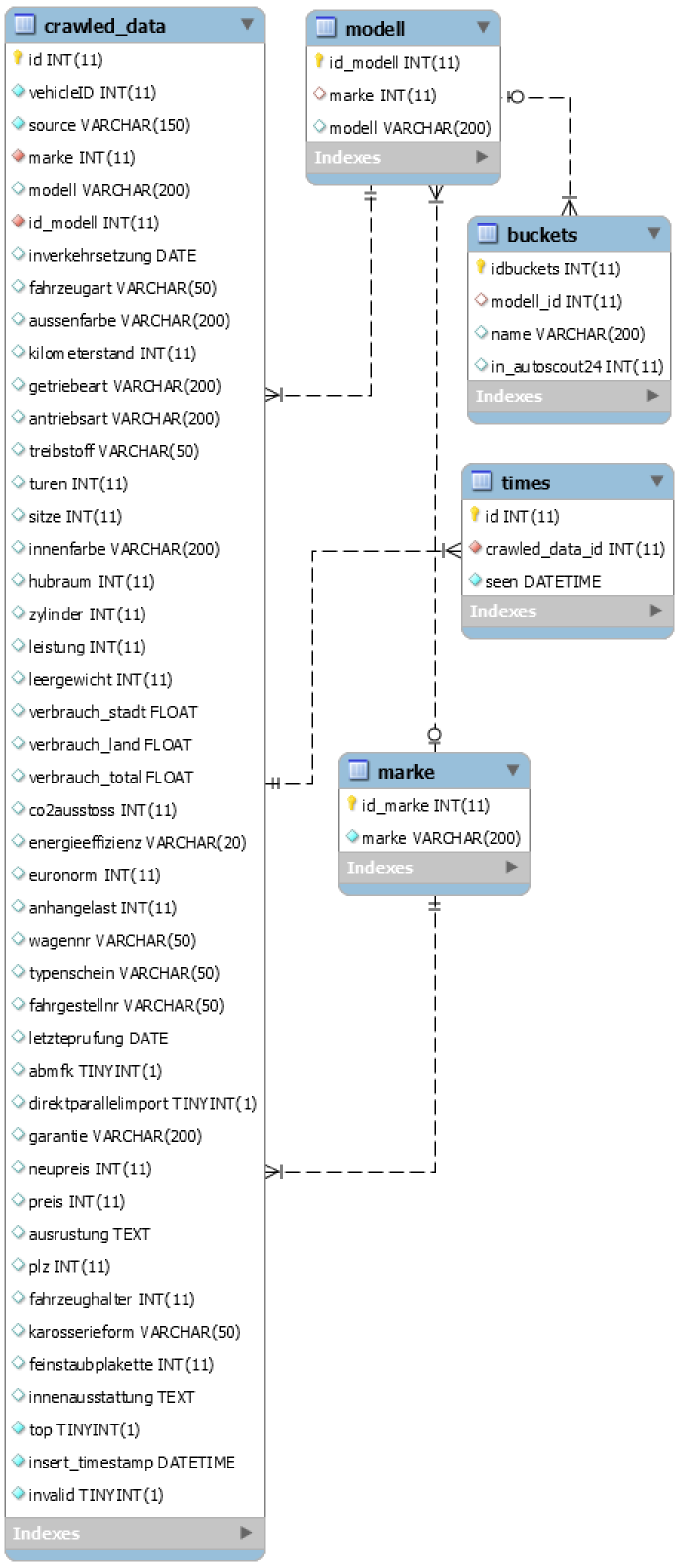

Figure 2.1: Database schema storing the car offers

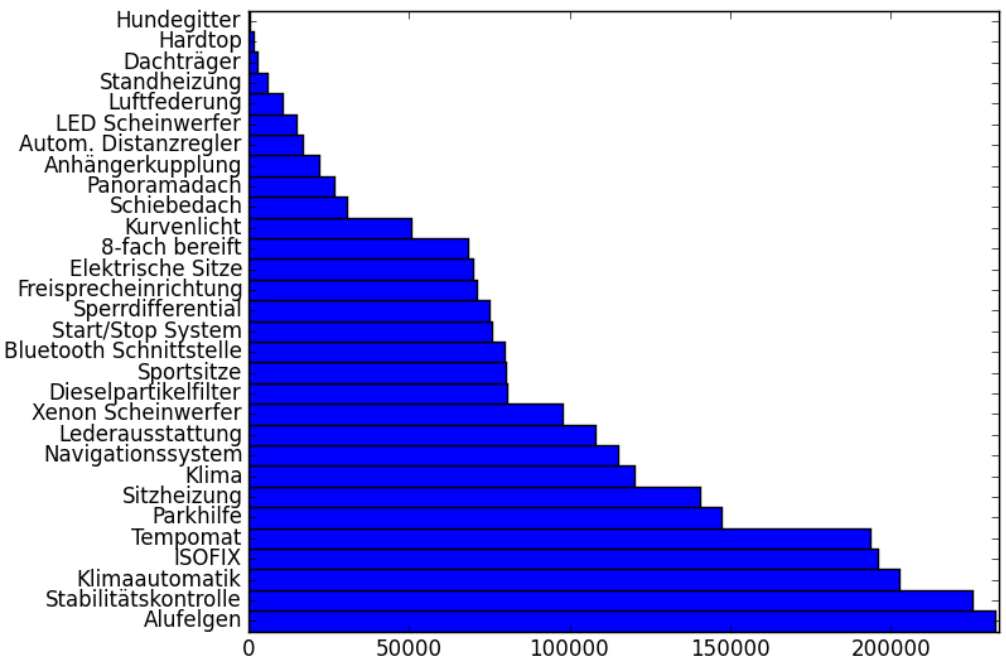

Figure 2.2: Equipment occurrences in car descriptions

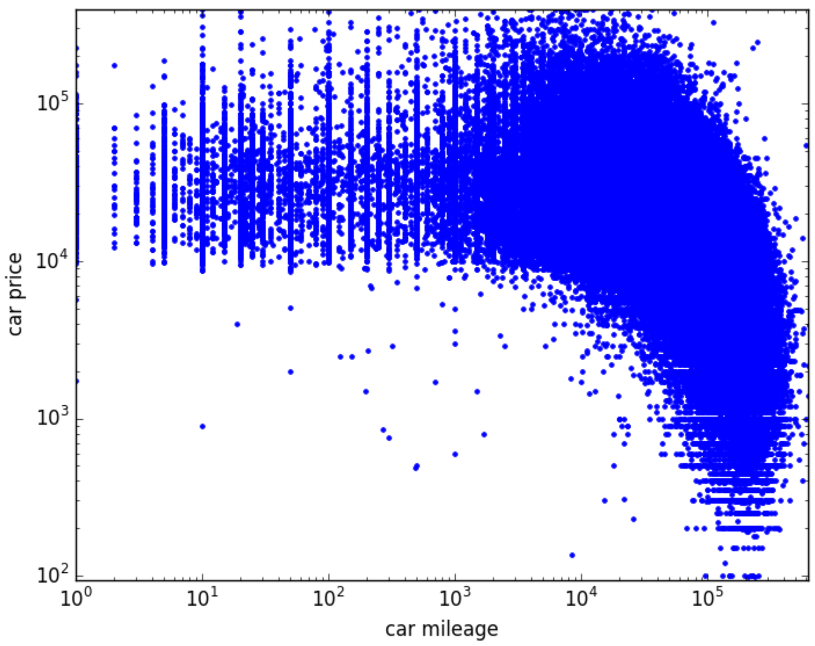

Figure 2.3: log-log plot of price vs mileage

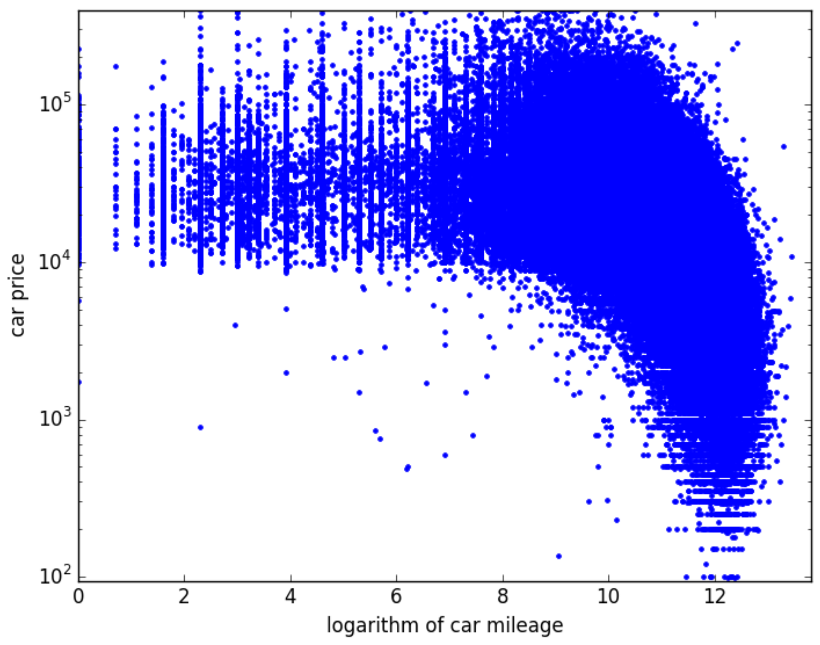

Figure 2.4: semi-log plot of price vs the logarithm of the mileage

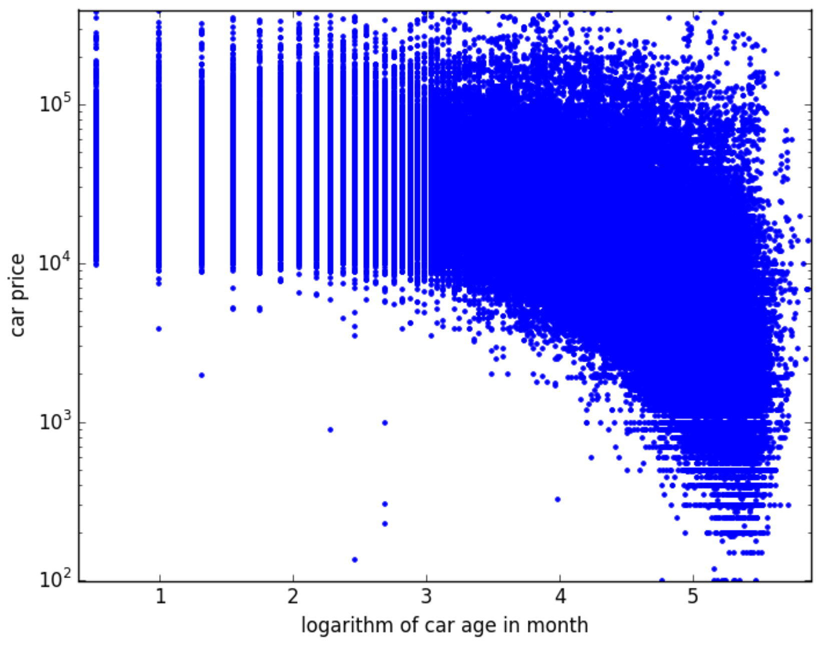

Figure 2.5: semi-log plot of price vs the logarithm of the mileage

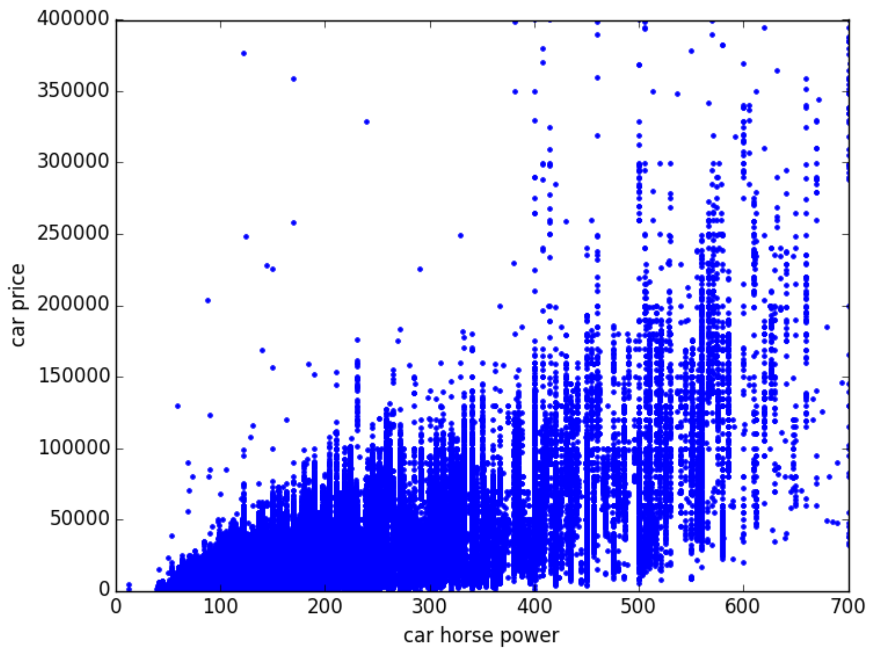

Figure 2.6: Price vs horse power

listed horse power of a car is an alternate indicator for the engine type, besides the seller's description of the model version. Many car models are sold with a variety of engines which usually do not have the exact same horse power. Usually more horse power also correlates with a higher price as can be seen in Figure 2.6. The fuel consumption behaves analogous in that it is greatly dependent on the type of engine and thus has a similar impact on the price. Figure 2.7 shows the car price vs fuel consumption.

2.3 Model

Figure 2.8 shows an overview of the process described in this chapter.

Usually, the dataset used for a single price estimation consists of all cars of a certain model (see Chapter 3 for some exceptions). At this point, after the feature processing, a linear regression still leads to relatively bad price estimations, which is why we split a car model into buckets of car model editions. The cars are initially sorted into natural buckets based on the model and exact matches on the the user given model name, which is often more descriptive as it contains the car edition. Since many of these buckets tend to be too small for any further evaluation, the smallest buckets are merged with larger ones based on whether all

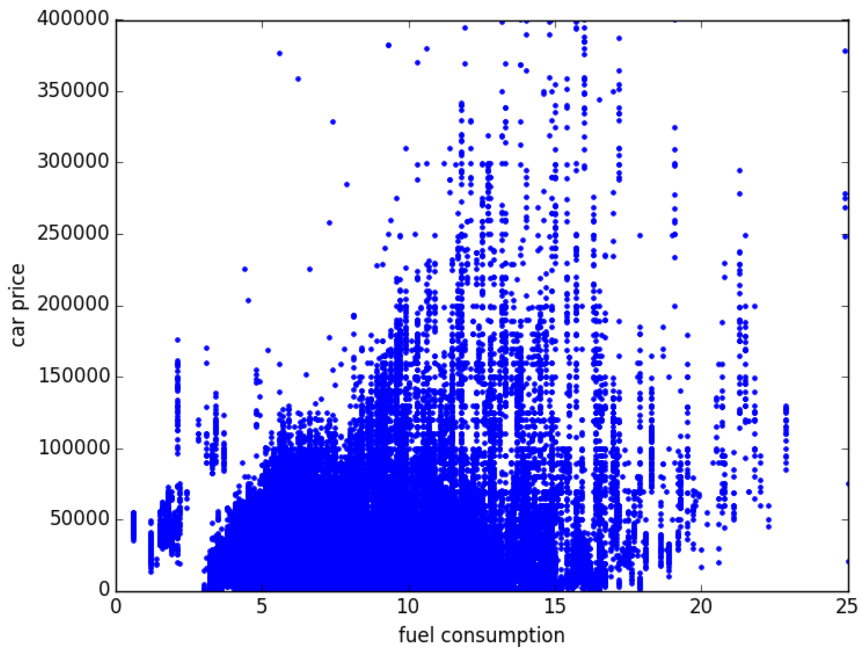

Figure 2.7: Price vs fuel consumption

words of one bucket name are contained in the other if and only if the minimum number of words that is equal in both names is above a set threshold bucket match (see Section 5.1). This threshold is in place to prevent really vague model names such as "Golf" to be merged with practically any other bucket of "VW Golf" cars. Cars are merged into buckets until no buckets smaller than a certain minimum bucket size (see Section 5.2) can be matched with larger buckets anymore, as merging is generally undesirable if it can be avoided to keep a more precise distinction of car models.

Within the bucket, we might still find cars that differ too much in age, equipment or engine. We therefore use a nearest neighbor rating on a select few features with numerically determined weight parameters (see Section 5.3). For a price check of some car A we compare it to all cars in the bucket it belongs to. The following Equation 2.1 shows the calculation for the rating of a car compared to A. The difference d of a categorical *cat.* feature *feat.* is 1 if the cars do not belong to the same category and 0 otherwise. The difference of a continuous *cont.* feature is a value in the range of 0 to 1, derived by normalizing the absolute value of all differences of that feature between all cars in the bucket and A. The weight parameter w expresses the weight for each feature.

$$R = \Big(\sum_{\forall cat.feat.} w_{cat.} * d_{cat.}\Big) + \sum_{\forall cont.feat.} w_{cont.} * d_{cont.} \tag{2.1}$$

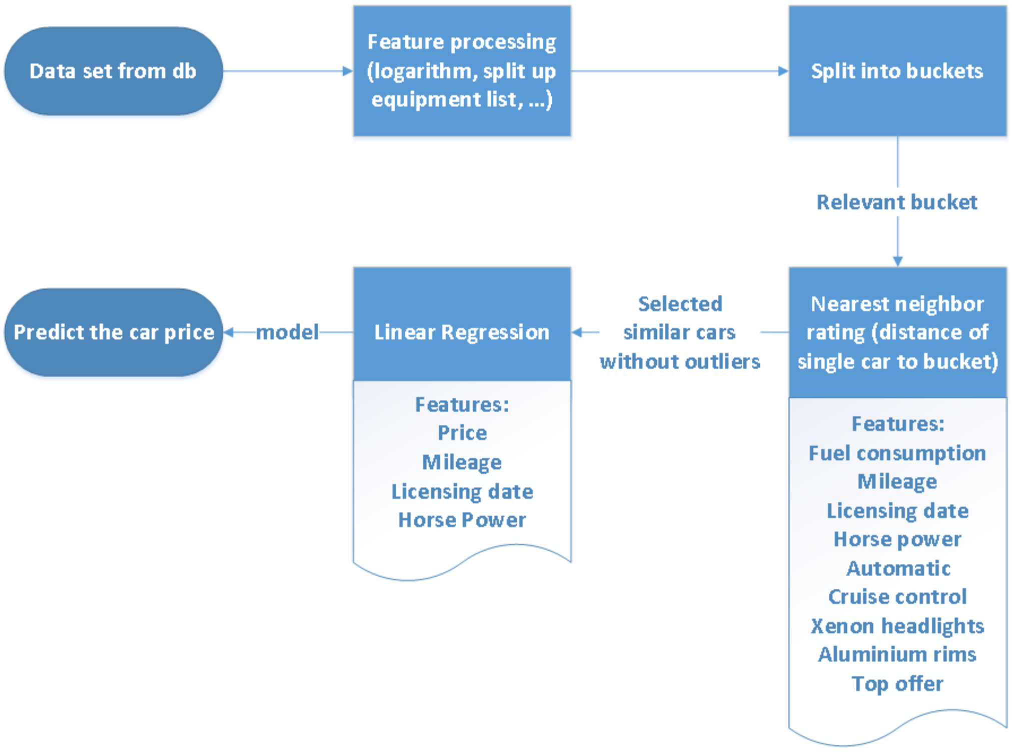

Figure 2.8: Processing an estimation

The smaller the rating, the more similar a car is to A. A perfect match in all features would thus be rated with a 0.

As described in Section 5.4, the bucket is reduced to the most similar 24 cars according to the nearest neighbor rating and further reduced by removing any cars which have a price that varies more than 3.5 times the standard deviation from the mean price as detailed in Section 5.5. This is the list of cars returned on the web application as similar (except for cars which are known not to be listed anymore). Finally we estimate a price using a multiple linear regression based on the price, horse power, mileage and licensing.

Chapter 3

Web Application

The price checker is integrated in a web application as final result, as searching listed cars, comparing and also buying or posting new cars to be listed is usually done on a computer. A computer offers better possibilities to do visual side by side comparison than mobile devices.

The web application provides a simple user interface where the user can paste an autoscout24.ch vehicleID or URL, or enter car features manually. On submit, the input is processed by the web app, which returns the estimated price and similar cars.

The user can choose a brand, car model and a model bucket, to specify best what car he is looking for. Only large enough buckets are shown. If a bucket is picked, the procedure is as described in Section 2.3. If no bucket is picked, the web application will estimate the price using all cars of the specified model as if it was one bucket which can produce a less accurate result. Suppose no car model is picked either or there are not enough cars of that model (less than the minimum bucket size), then all cars of the specified brand are used as if it was one bucket. And lastly, if there are not enough cars of the brand (less than the minimum bucket size) or no brand is specified, all cars in our database are used to estimate the car price. These generalizations are provided so that we can always return a result, even if the estimates and car similarities can be less accurate.

Chapter 4

Implementation

Python 2.7 is generally the programming language of choice for this project due to the readily available packages on machine learning, statistical models, data processing and model plotting. Jupyter Notebook[13] is used as IDE.

The Scrapy[14] library enables easy crawling to store the data in a MySQL[15] database. The crawler runs every two to four days as cron job for each of the three websites and collected over the course of the project data (including incomplete entries) of nearly 3.5 million cars. The database schema is shown in Figure 2.1. The column *ausrustung* notably contains the equipment which was previously mentioned (see 2.2) as a semi-colon separated list. The data is loaded from the database into a Pandas [16] dataframe with the Mysql-Python connector [17]. Numpy [18] is used to perform operations on series as such as the logarithm on a dataframe column and also internally by the Pandas library. The linear regression is done with statsmodel[19], more specifically using the formula api, which allows easy differentiating of categorical and continuous features. The web application uses flask[20]. The library matplotlib[21] is used to plot the graphs.

Chapter 5

Results

All evaluations are performed with the listed cars from autoscout24.ch. We mostly used the car model VW Golf, as it is the most frequently listed car on that platform as can be seen in Figure 5.1. To measure the quality of price estimates, they are compared to the listed prices. Estimates within ±20% of the listed price are considered to be ok. The percentage of all cars we have estimated within that 20% range is used for parameter tuning. It is interesting to note, that the calculated linear regression parameters do not necessarily correlate to the features in the way which would trivially be expected. If for example a bucket by chance contains mainly two car model versions, one slightly older than the other, it still might be that the older model is a more luxurious and expensive car. Similarly a car with less horse power might be electric and thus more expensive than the stronger gasoline-driven car.

5.1 Determining the parameter bucket match

A small model bucket can be merged with a larger bucket if the model names have three words in common. This number is derived by observing the usual model name patterns. For a VW Golf an excerpt of the names is shown in Figure 5.2. Suppose the parameter was two, the name "Golf 1.6" could be merged with any bucket as Golf 1.6 Comfortline, Golf 1.6 16V, Golf 1.6 TDI Comformt 4M, and so on. Figure 5.3 shows the results for values two, three and four together with how many cars are in buckets for each. Three was derived as compromise between better results and more cars per bucket.

5.2 Determining the parameter minimum bucket size

The minimum bucket size is set to 50. Due to time constraints and the large dataset size the parameter is not numerically derived for the best result. Considering the number of cars per model (see Figure 5.5) it is deliberate though, as more would lead to more buckets being merged that do not belong together

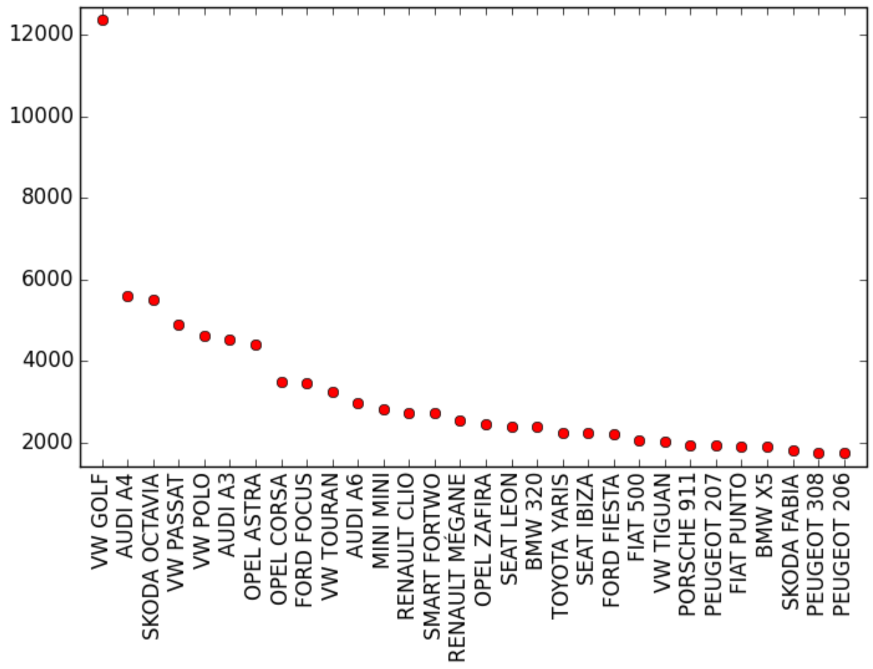

Figure 5.1: Number of cars of the 30 most crawled car models

```
GOLF 1.2 TSI COMFORT             GOLF 1.4 TSI COMFORT
GOLF 1.6                         GOLF 2.0 TSI R 4MOTION
GOLF 1.4 TSI LOUNGE              GOLF 1.4 TSI TEAM
GOLF 1.6 COMFORTLINE             GOLF 2.0 TSI GTI
GOLF 1.6 TDI COMFORT             GOLF 1.4 TSI HIGH
GOLF 2.0 TDI GTD                 GOLF 2.0 TFSI GTI
GOLF 1.9 TDI COMFORT             GOLF 2.0 TDI COMFORT
GOLF 2.0 TDI HIGH                GOLF R32 4MOTION
GOLF 2.0 TDI COMFORT 4M.         GOLF 1.8 T GTI
GOLF 1.4 TSI TREND               GOLF 1.4 TSI CUP
GOLF 1.6 16V                     GOLF 1.2 TSI TREND
GOLF 1.6 TDI COMFORT 4M          GOLF 2.0 TSI GTI PERFORM.
GOLF 2.0 COMFORTLINE             GOLF 1.6 16V COMFORTLINE
GOLF 1.4 TSI                     GOLF 1.4 TSI GTE
GOLF 2.0 TSI GTI TEAM            GOLF 1.6 FSI COMFORT
```

Figure 5.2: Excerpt of VW Golf bucket names

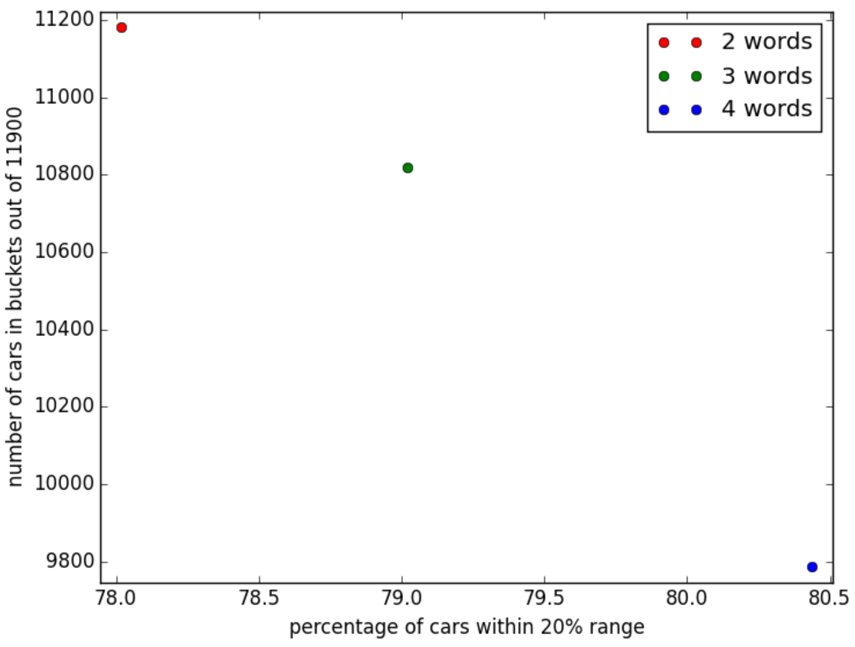

Figure 5.3: The number of cars in large enough buckets for different minimum bucket match parameters vs the percentage of cars within the 20% range.

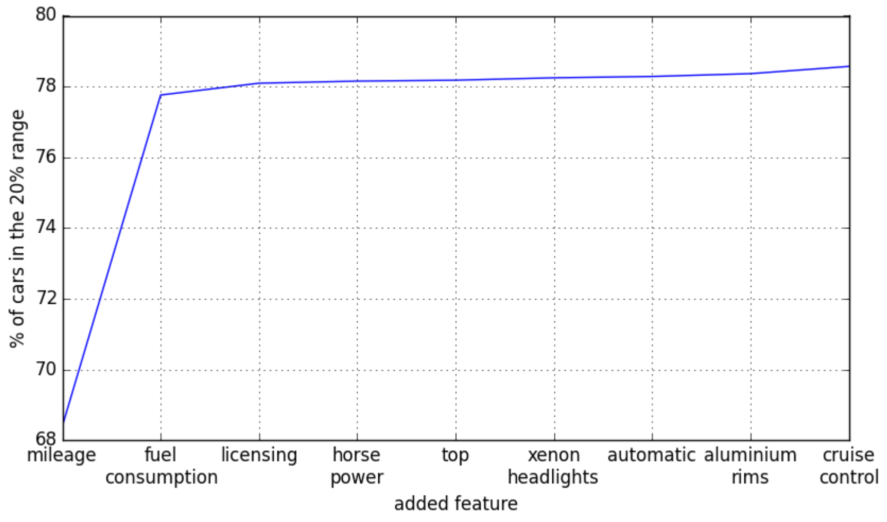

Figure 5.4: Features for the nearest neighbor rating and the 20% range accuracy they achieve by stacking up from left to right

and less would shrink the number of cars left over for the linear regression to a concerningly small number.

5.3 Determining the nearest neighbor weight parameters

The nearest neighbor rating has a weight parameter for each feature used. The logarithm of the mileage is given a weight of 1 as baseline, since that and the licensing date are two features which correlate with the price as seen already in Figure 2.3 and Figure 2.5. The difference in continuous features from a single car is calculated against the dataset (usually a bucket) and normalized so that the largest difference is given a value of one and no difference is a zero. On the other hand, the categorical features are either the same or not, thus the weight is either added to the difference or not at all. One by one the parameters are set by iteratively testing features with a reasonable range of weights, adding the best and repeating the process with the remaining features. Many of the added features do not lead to improvements, but the ones which do and their parameter can be seen in figure 5.4 together with the overall estimation accuracy. Table 5.1 shows the weight parameters for these features.

Feature	**Weight Parameter**
Mileage	1
Fuel Consumption	0.8
Licensing	0.3
Horse power	0.4
Top	0.6
Xenon headlights	0.1
Automatic	0.1
Aluminium rims	0.3
Cruise control	0.1

Table 5.1: Features and their weight parameter for the nearest neighbor rating

5.4 Determining the parameter nearest neighbor filtering

The 24 most similar cars in a bucket are used for further processing. The size 50 turned out to be best for the VW Golf, but buckets of size 100 as used to derive that number are unpractical for other car models as too many cars would end up not in large enough buckets. Figure 5.5 shows for each model how many cars it has in a log-log plot. The two intercepting black lines mark the model sizes of 50 and 100 for reference. Figure 5.6 shows the car price estimations within the 20% range for the VW Golf with varying parameter size, with 50 as maximum and 24 as a local maximum.

5.5 Filtering outliers

A car from the list of similar cars is considered an outlier if the difference of price to the mean of the list is more than a factor times the standard variance. By iterating over a range of factors and running the price estimator for each the best is determined. 3.5 is the best factor as can be seen in Figure 5.7.

5.6 Price checker accuracy

The results for testing all cars against the implementation if they are within the +-20% price variation are shown in the Table 5.2, where category 1 refers to cars tested against all cars of the brand, category 2 are cars tested against the whole car model and category 3 are cars tested against the bucket only. The exception are 199 cars which would be tested against all cars. The Table 5.3 is analogous to Table 5.2. It shows similar results but slightly improved. They are derived by excluding the initial crawl from the data set. This is likely due to

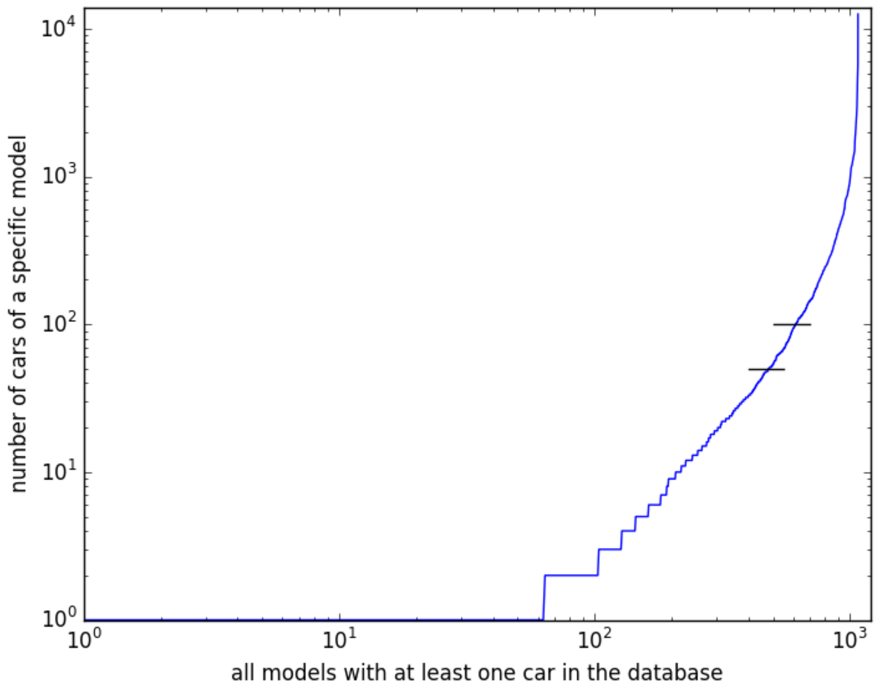

Figure 5.5: Number of cars per model as log-log plot with marks at model sizes 50 and 100

Category	**Number of cars**	**Percentage of cars**	**Percentage in +-20%**
1	8308	3.81%	22.75%
2	98546	45.23%	45.78%
3	111017	50.96%	63.94%
Total	217871	100.00%	54.15%

Table 5.2: Number of cars and accuracy of our price estimation model shown together and separated in the three categories (category 1: cars tested against all cars of the brand, category 2: cars tested against the whole car model, category 3: cars tested against the model version bucket only).

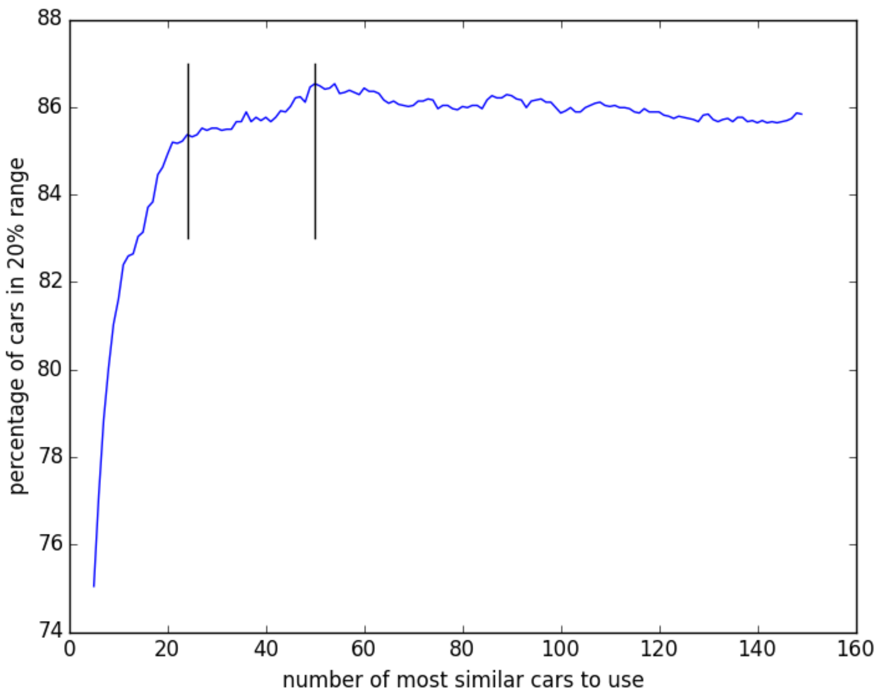

Figure 5.6: Car price estimations for the VW Golf with a variable number of most similar cars and a minimum bucket size of 100. 24, the local maximum which is in use and 50, the global maximum are marked.

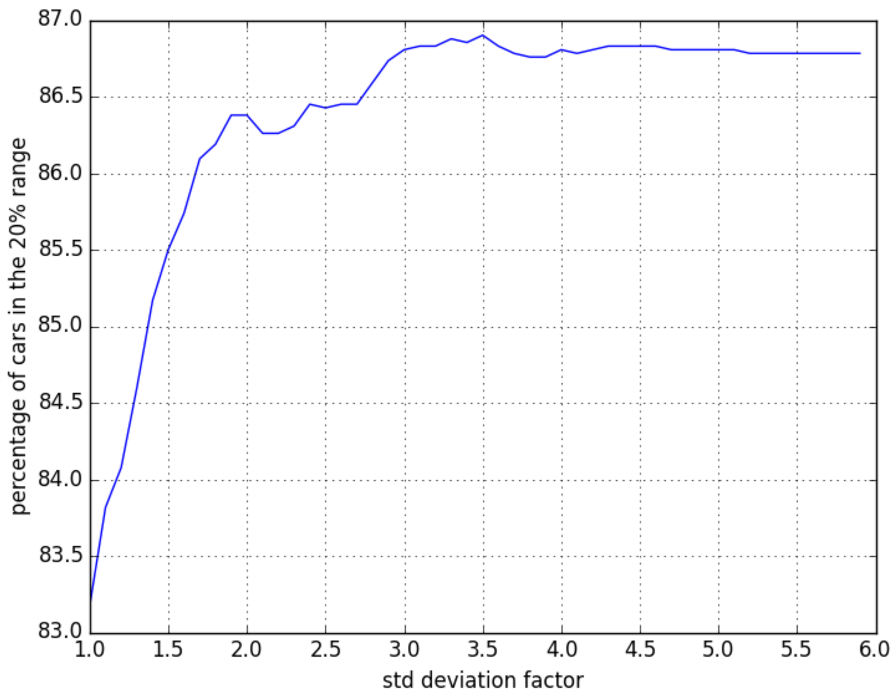

Figure 5.7: Factor of the standard deviation for cutting off outliers vs the estimation accuracy

Category	Number of cars	Percentage of cars	Percentage in +-20%
1	8760	6.71%	31.42%
2	73737	56.47%	53.69%
3	48069	36.82%	68.33%
Total	130566	100.00%	57.59%

Table 5.3: Number of cars and accuracy of our price estimation model shown together and separated in the three categories (category 1: cars tested against all cars of the brand, category 2: cars tested against the whole car model, category 3: cars tested against the model version bucket only). The dataset used here excludes the initial crawl.

excluding a group of cars that sell rarely and are online for a long time due to being overpriced or special in some other way. Considering only a single model as the VW Golf we can show the amount of cars in the 20% range nicely by graphing the accumulation of cars in the y-axis and the percentage by which the estimation is off in the x-axis, as shown in Figure 5.8.

5.7 Correlation of price and the time that an offer is online

A timestamp is added when a car is crawled and another timestamp entry is added whenever a car is seen again. We can therefore show how our price estimate tends to be too high for cars which sell quickly. Figure 5.9 is a density map which shows on the x-axis how long an offer is online with a variation of plus 0-3 days as we can only estimate when it is taken offline by not seeing it during the next crawl process. The y-axis indicates the difference in percent that the price estimate is higher or lower than the actual price. The density is shown by the color with white indicating no car, the predominant dark blue that there is only one car there and the lighter blue to green, yellow and red indicates a gradually higher density with dark red having the highest. The first instance of crawled cars is excluded because they might have been placed online long before. It stands out that many of our estimates are off by multiple 100 or even 1000 percent, especially also negative values larger than 100%. For the web application any negative price estimates are displayed as 0.

Figure 5.10 shows the same as figure 5.9 but with an enlarged y-axis. The highest density is clearly above the y=0 axis, which indicates our too high estimates for fast selling cars.

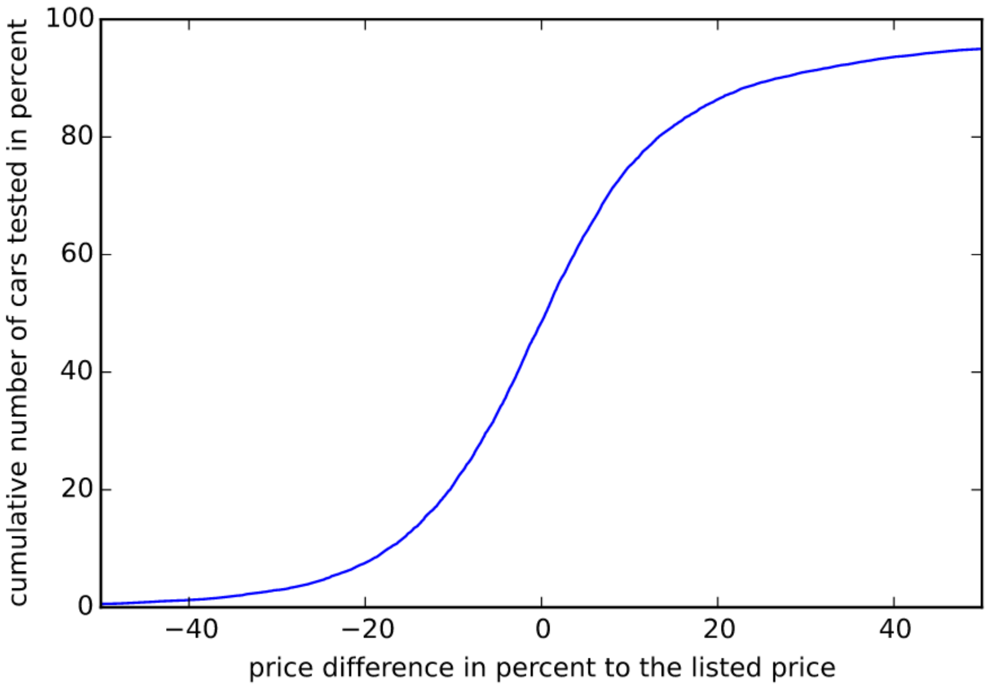

Figure 5.8: Price difference of the VW Golf estimates in percent to the listed price vs the percentage of cars cumulative.

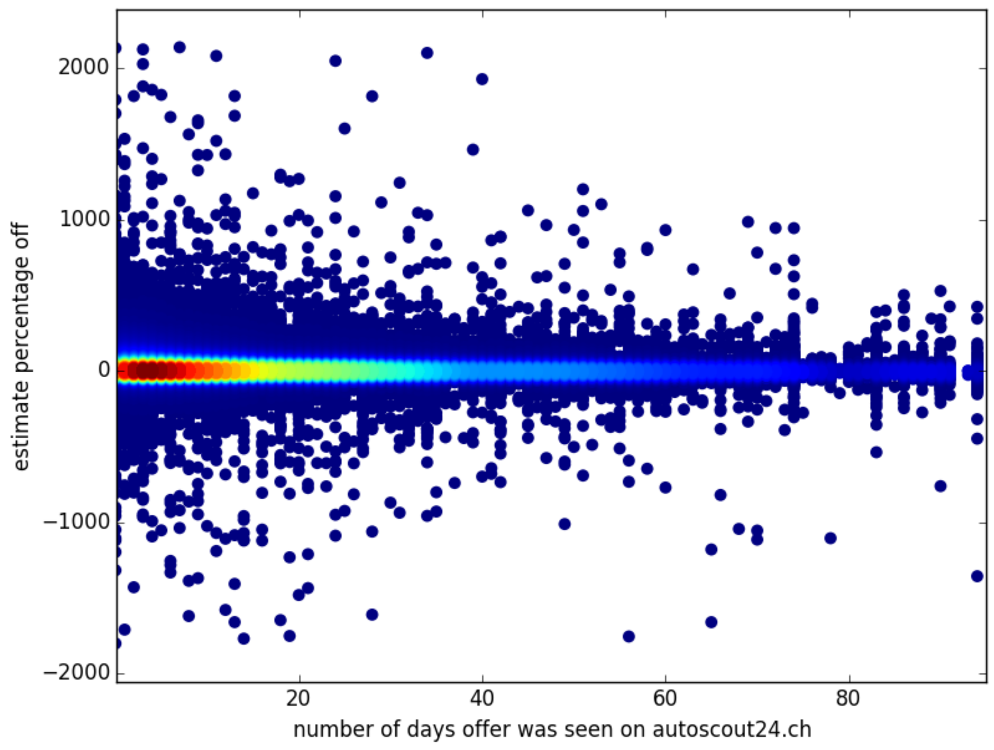

Figure 5.9: The number of days a car is listed on autoscout24.ch vs the percentage by which the estimate is off from the listed car price, shown as density map. Dark red indicates the highest density, white indicates no car to show there and the dark blue which is shown predominantly indicates only one car there.

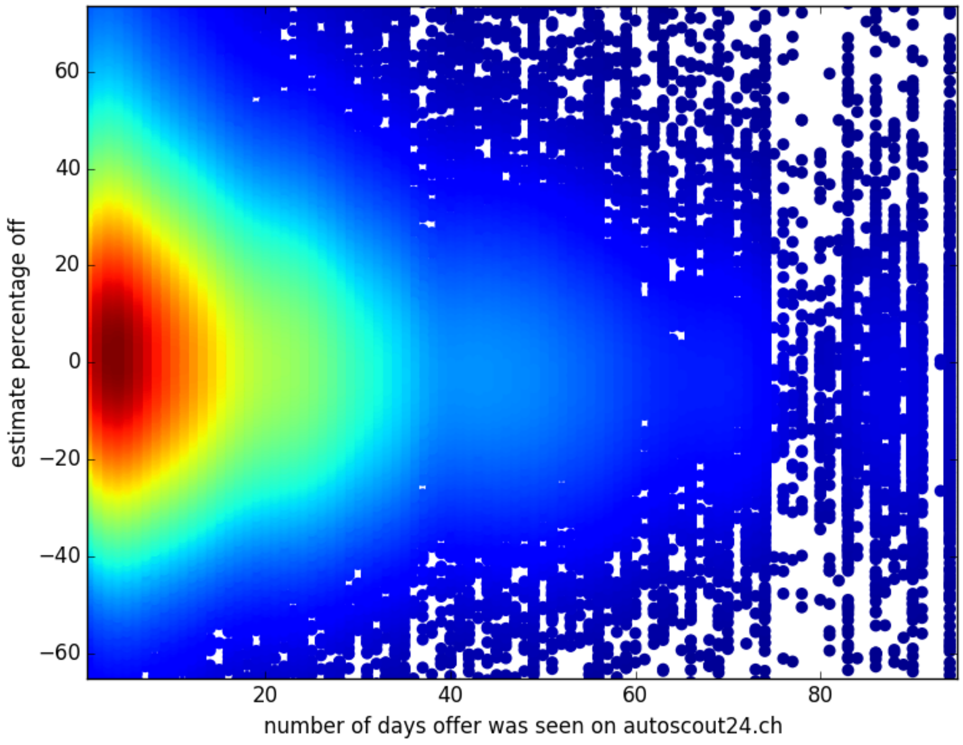

Figure 5.10: The number of days a car is listed on autoscout24.ch vs the percentage by which the estimate is off from the listed car price, shown as density map zoomed in on the y-axis. Dark red indicates the highest density, white indicates no car to show there and the dark blue which is shown predominantly at the right border for example indicates only one car there.

Chapter 6

Summary & Outlook

The web application provides users the possibility to look up an estimated car price and similar cars based on a few features or a link to a car listed on autoscout24.ch. 63% of the estimates lie within a range of 20% from the listed price. The functionality of returning also similar cars can provide better productivity and for some car models which occur frequently in the dataset, the price estimates are significantly better. Exactly these car models can bring up also another use case for this tool, when there are too many search results to get a good overview and find appropriate offers directly on a platform. Estimating car prices for models which are rare does not become much easier with this tool.

6.1 Possible problems and improvements

- Fit the parameters for all cars and less for specific popular models.
- Come up with better models possible for the situations where currently the whole car model, car brand or all cars are used like a bucket. They would probably use more features to find similar cars across all models and brands.
- The web application is currently relatively slow on processing an estimate as it loads most data directly from the database instead of only updating it in regular intervals. As the data only changes slowly in the database and it is not important to have live data, caching could improve the performance significantly without impeding accuracy.
- The web application could crawl a new car offer right away given an autoscout24 vehicle ID or URL to a car which is not in the database, instead of not being able to provide a result. The list of similar cars could also be checked to see if they are still online before returning them as result in the web application.
- Sellers can change the price of a listed car without creating a new listing. These changed prices should be stored in the database.

- Integrate collected data about German cars to predict not only German car prices separately but maybe also to make up for insufficient data about certain rare cars in Switzerland.

Bibliography

[1] : Swiss Statistics, level of motorisation. http://www.bfs.admin.ch/bfs/portal/en/index/themen/11/03/blank/02/01/01.html Accessed: 2016-03-18.

[2] : Comparis. https://www.comparis.ch Accessed: 2016-03-18.

[3] : Comparis car value finder by Andreas Keller. https://www.comparis.ch//carfinder/info/glossar/comparis-bewertung.aspx Accessed: 2016-03-23.

[4] : Kelley Blue Book. http://www.kbb.com/whats-my-car-worth/ Accessed: 2016-03-18.

[5] : autotrader.co.uk. http://www.autotrader.co.uk/car-valuation Accessed: 2016-03-18.

[6] : carsales. http://www.carsales.com.au/car-valuations/ Accessed: 2016-03-18.

[7] : eurotaxglass. http://www.eurotaxglass.ch/ Accessed: 2016-03-18.

[8] : auto ricardo. https://auto.ricardo.ch Accessed: 2016-03-18.

[9] : car4you. https://www.car4you.ch/ Accessed: 2016-03-18.

[10] : autoscout24.ch. http://www.autoscout24.ch/ Accessed: 2016-03-18.

[11] : mobile.de. http://www.mobile.de Accessed: 2016-03-18.

[12] : autoscout24.de. https://www.autoscout24.de/ Accessed: 2016-03-18.

[13] : Jupyter Notebook. http://jupyter.org/ Accessed: 2016-03-22.

[14] : Scrapy. http://scrapy.org/ Accessed: 2016-03-18.

[15] : MySQL. https://www.mysql.com/ Accessed: 2016-03-18.

[16] : Pandas. http://pandas.pydata.org/ Accessed: 2016-03-18.

[17] : Mysql-Python connector. https://dev.mysql.com/doc/connector-python/en/ Accessed: 2016-03-18.

[18] : Numpy. http://www.numpy.org/ Accessed: 2016-03-18.

[19] : Statsmodels. http://statsmodels.sourceforge.net/ Accessed: 2016-03-18.

[20] : Flask web framework. http://flask.pocoo.org/ Accessed: 2016-03-18.

[21] : Matplotlib. http://matplotlib.org/ Accessed: 2016-03-22.