Jedha Bootcamp - Fundamentals course final project:

Using Machine Learning to value a second-hand car

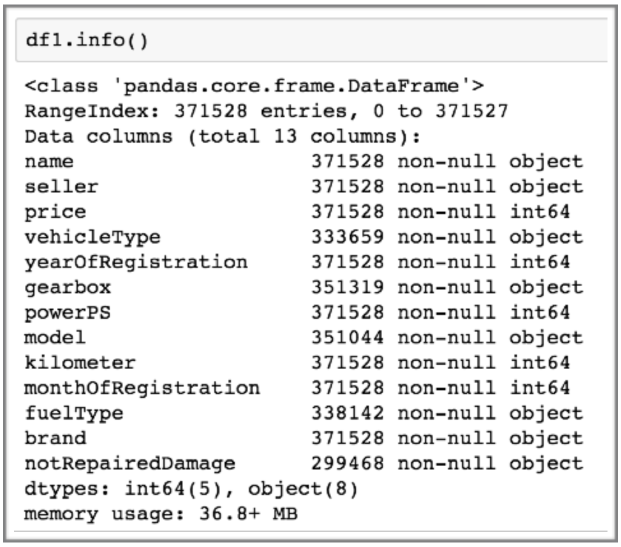
By Agata Gramatyka
October 2018

Introduction

Most cars sold each year around the world are second-hand. They represent 70% of the total number of cars sold in France, or 5.6 million in 2016 (according to *Le Figaro*). The market is very big and largely unregulated, with the price being very often established through negotiation. As such, obtaining a fair price for their car or buying one without overpaying is a concern for most people at some point in their lives. Thanks to vast and growing quantities of data collected about our vehicles, we could remove the doubt over fair valuation through application of Machine Learning.

The dataset

The raw dataset used for this exercise is available from Kaggle (https://www.kaggle.com/orgesleka/used-cars-database). It contains 371,528 records of car sale ads scraped from the German ebay in March 2016.



```
df1.info()
```

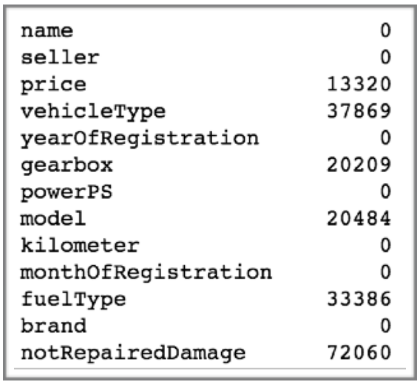
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 371528 entries, 0 to 371527
Data columns (total 13 columns):
name                   371528 non-null object
seller                 371528 non-null object
price                  371528 non-null int64
vehicleType            333659 non-null object
yearOfRegistration     371528 non-null int64
gearbox                351319 non-null object
powerPS                371528 non-null int64
model                  351044 non-null object
kilometer              371528 non-null int64
monthOfRegistration    371528 non-null int64
fuelType               338142 non-null object
brand                  371528 non-null object
notRepairedDamage      299468 non-null object
dtypes: int64(5), object(8)
memory usage: 36.8+ MB
```

Data cleaning

Due to the origin of the data as well as its vast quantity, the dataset is rather “unclean” : there are many missing values, false values, outliers etc. That’s why data cleaning was the most complicated and time consuming step of the process, however it was crucial for the quality of the future predictions.

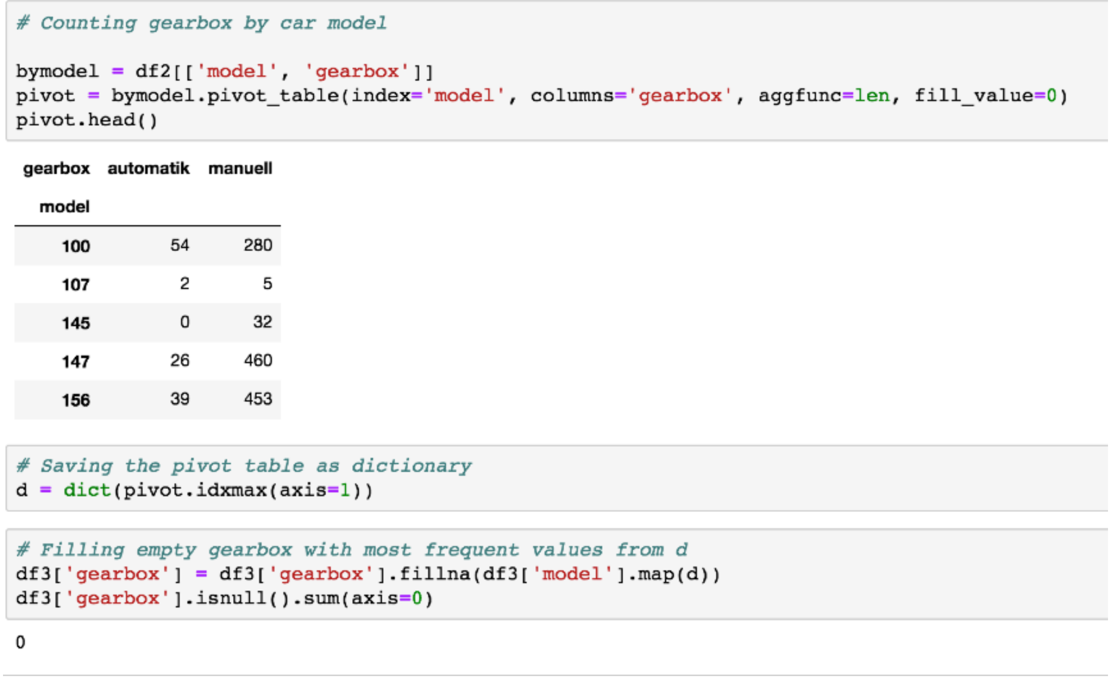
1) Missing values

Thousands of missing values were detected across 6 different columns:



```
name                         0
seller                       0
price                    13320
vehicleType              37869
yearOfRegistration           0
gearbox                  20209
powerPS                      0
model                    20484
kilometer                    0
monthOfRegistration          0
fuelType                 33386
brand                        0
notRepairedDamage        72060
```

I started by removing observations where it was impossible or tricky to impute the missing values. This was the case for `notRepairedDamage` , where it was impossible to know if the car was damaged or not; and `price`, which is the dependent variable that the Machine Learning model is trying to predict.
Imputation was implemented in the case of `vehicleType`, `gearbox` and `fuelType`. To do this, the most common value for a given car model was assigned to each null value.



```
# Counting gearbox by car model

bymodel = df2[['model', 'gearbox']]
pivot = bymodel.pivot_table(index='model', columns='gearbox', aggfunc=len, fill_value=0)
pivot.head()
```

gearbox	automatik	manuell
model		
100	54	280
107	2	5
145	0	32
147	26	460
156	39	453

```
# Saving the pivot table as dictionary
d = dict(pivot.idxmax(axis=1))
```

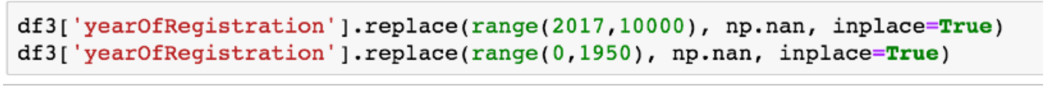
```
# Filling empty gearbox with most frequent values from d
df3['gearbox'] = df3['gearbox'].fillna(df3['model'].map(d))
df3['gearbox'].isnull().sum(axis=0)
```

```
0
```

2) False values and outliers

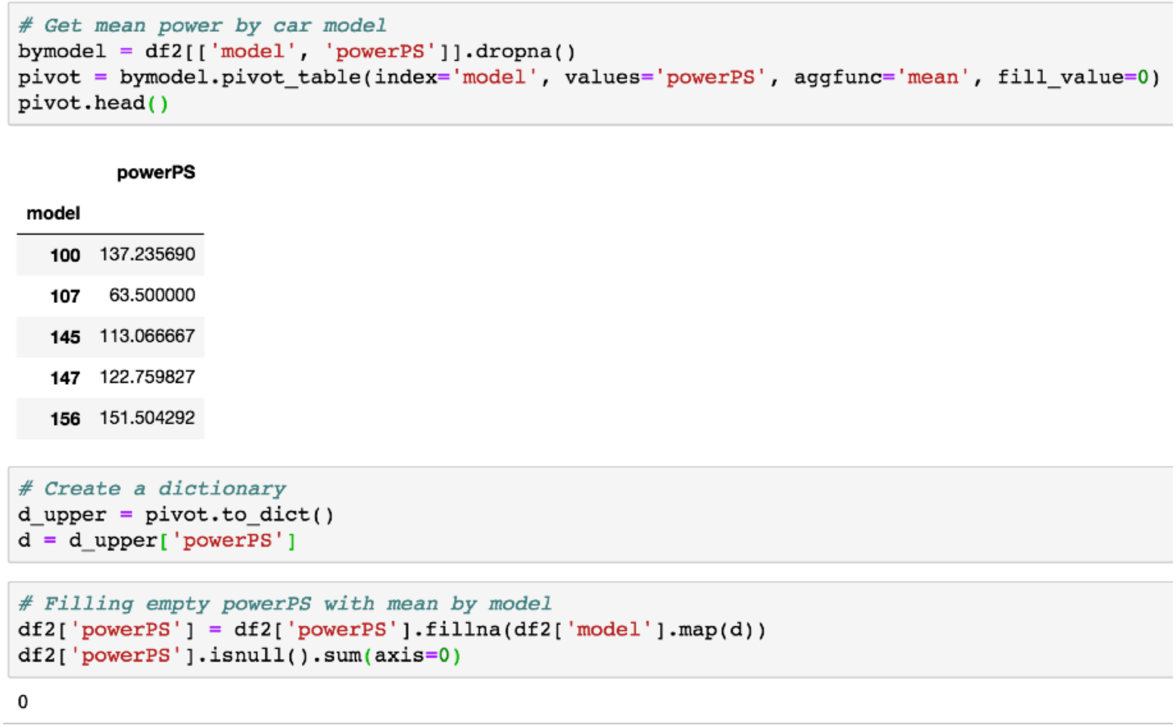
After closer inspection, it became apparent that the numerical variables `price`, `yearOfRegistration` and `powerPS` contain false or zero values.
To deal with these, and to narrow down the scope of the study I discarded all records where:

- the price is less than 100 or over 100,000 EUR
- the year of registration is before 1950 or after 2016.



```
df3['yearOfRegistration'].replace(range(2017,10000), np.nan, inplace=True)
df3['yearOfRegistration'].replace(range(0,1950), np.nan, inplace=True)
```

The false brake horsepower values can be easily replaced by applying a similar technique to the one I used for imputation, which is inserting average BHP values for a given model. I have done this for records where the BHP values were zero or greater than 600.



```
# Get mean power by car model
bymodel = df2[['model', 'powerPS']].dropna()
pivot = bymodel.pivot_table(index='model', values='powerPS', aggfunc='mean', fill_value=0)
pivot.head()
```

model	powerPS
100	137.235690
107	63.500000
145	113.066667
147	122.759827
156	151.504292

```
# Create a dictionary
d_upper = pivot.to_dict()
d = d_upper['powerPS']
```

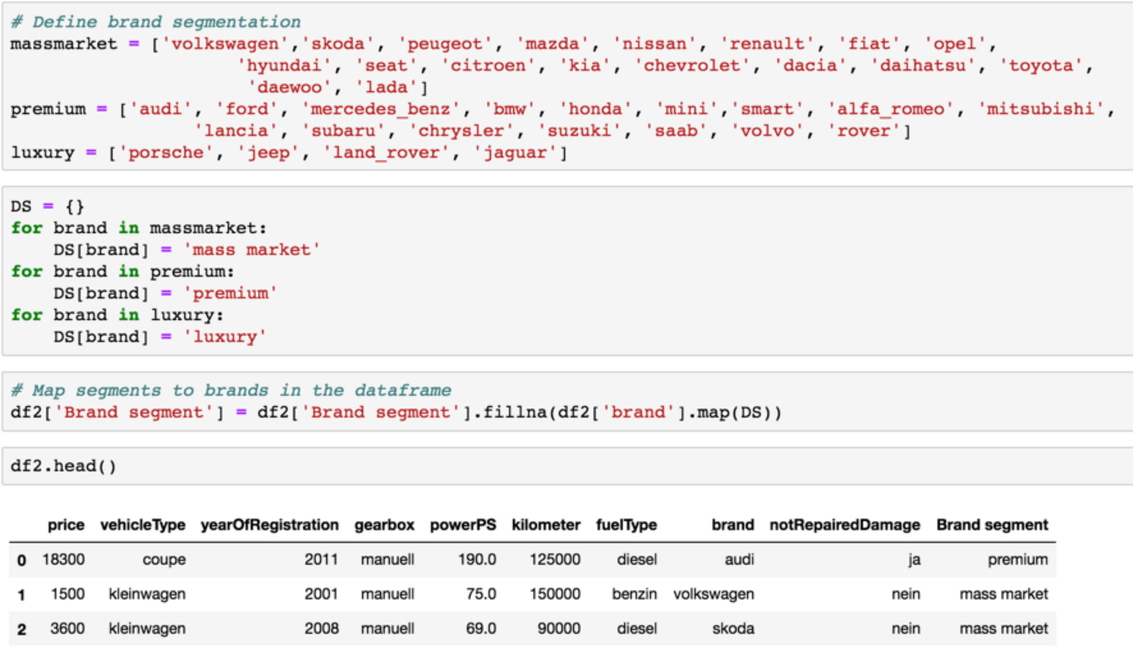
```
# Filling empty powerPS with mean by model
df2['powerPS'] = df2['powerPS'].fillna(df2['model'].map(d))
df2['powerPS'].isnull().sum(axis=0)
```

```
0
```

I have then narrowed the sample further by keeping only cars registered in the 2000-2016 period. The reason I've done this is because the valuation and depreciation of classic cars, luxury or sportscars follow completely different rules. If kept in the model, those outliers would negatively affect the quality of my predictions.

Brand segmentation

The total number of brands in the dataset is 40. I have decided to group them by market segment, which would simplify the model by reducing the number of dummy variables. The brands were assigned to three segments: mass market, premium and luxury.



```
# Define brand segmentation
massmarket = ['volkswagen','skoda', 'peugeot', 'mazda', 'nissan', 'renault', 'fiat', 'opel',
              'hyundai', 'seat', 'citroen', 'kia', 'chevrolet', 'dacia', 'daihatsu', 'toyota',
               'daewoo', 'lada']
premium = ['audi', 'ford', 'mercedes_benz', 'bmw', 'honda', 'mini','smart', 'alfa_romeo', 'mitsubishi',
           'lancia', 'subaru', 'chrysler', 'suzuki', 'saab', 'volvo', 'rover']
luxury = ['porsche', 'jeep', 'land_rover', 'jaguar']
```

```
DS = {}
for brand in massmarket:
    DS[brand] = 'mass market'
for brand in premium:
    DS[brand] = 'premium'
for brand in luxury:
    DS[brand] = 'luxury'
```

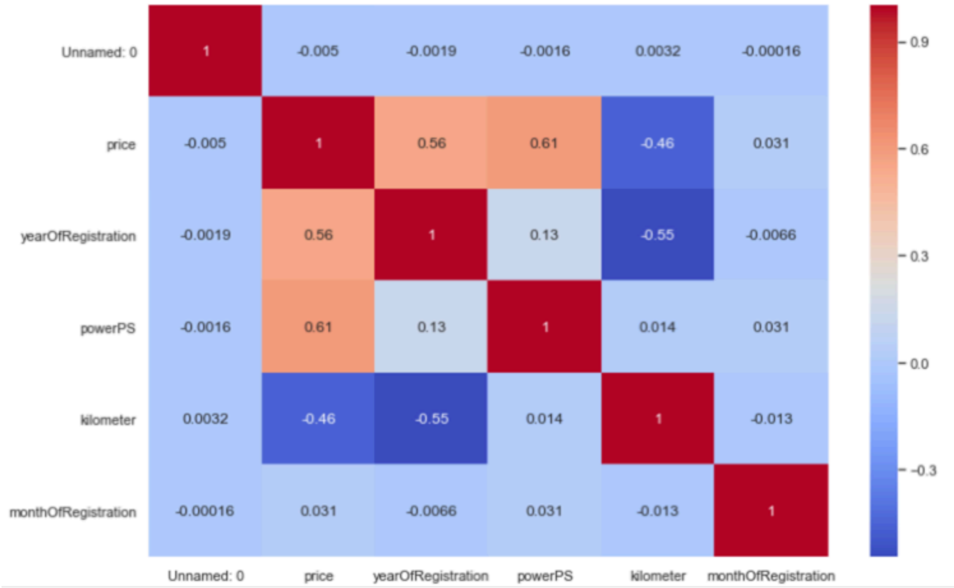
```
# Map segments to brands in the dataframe
df2['Brand segment'] = df2['Brand segment'].fillna(df2['brand'].map(DS))
```

```
df2.head()
```

	price	vehicleType	yearOfRegistration	gearbox	powerPS	kilometer	fuelType	brand	notRepairedDamage	Brand segment
0	18300	coupe	2011	manuell	190.0	125000	diesel	audi	ja	premium
1	1500	kleinwagen	2001	manuell	75.0	150000	benzin	volkswagen	nein	mass market
2	3600	kleinwagen	2008	manuell	69.0	90000	diesel	skoda	nein	mass market

Modelling

The exploratory data analysis suggested negative correlation between the age and mileage of a vehicle and its price, and positive correlation between brake horsepower and price.



Based on data visualisation which showed linear relationship between age and vehicle price, I have selected the Multiple Linear Regression model. The model was trained on 70% of the dataset and yielded the following coefficients:

Independent variable	Coefficient
Intercept	11428
brake horsepower	64
mileage (km)	-0,06
Age (years)	-578
unrepaired damage	-2063
Brand segment	
luxury	8396
premium	568
mass market	0
Body type	
cabriolet	1774
coupe	1179
SUV	783
van	493
hatchback	290
saloon	-463
estate	-829
Transmission	
automatic	0
manual	-784
Fuel type	
hybrid	2202
diesel	1990
petrol	0
elektro	-706
lpg	-1118

We can interpret the above as follows:

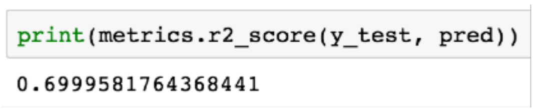
For a starting value of 11,428€,

- every unit of brake horsepower increases the price by 64€,
- every 100km driven reduce the price by 60€,
- every year since registration reduces the price by 578€,
- the presence of any unrepaired damage reduces the price by 2,063€,

and so on.

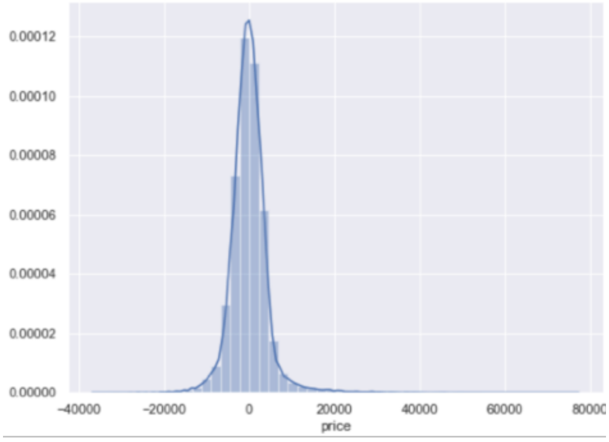
Model evaluation

The coefficient of determination (R-squared) is a widely used indicator of performance of regression models. The closer it is to 0, the closer the given model's predictions are to reality. Our model's R-squared coefficient is 0.70, indicating good fit.



```
print(metrics.r2_score(y_test, pred))
0.6999581764368441
```

The difference between the model's prediction and the true value is known as error. The below histogram shows that our model's error is normally distributed and concentrated around zero, which in statistical terms is very satisfactory.



Conclusion

Given a limited scope of this project, a simplified Multiple Linear Regression model trained on static, historical data has fulfilled the project's goals.

Thanks to the great power of Machine Learning, the idea could potentially be taken much further. For example, the model could be made dynamic and use more granular data for even more precise results. Another interesting aspect which could be explored is whether and how the rate of depreciation differs among car brands and models. Machine Learning could help us choose a car which loses value less fast than other, comparable models.