In [5]:
```
import pandas as pd
car = pd.read_csv("autos.csv", encoding="Latin-1")
```

In [6]:
```
car.columns
```

Out[6]:
```
Index(['dateCrawled', 'name', 'seller', 'offerType', 'price', 'abtest',
       'vehicleType', 'yearOfRegistration', 'gearbox', 'powerPS', 'model',
       'kilometer', 'monthOfRegistration', 'fuelType', 'brand',
       'notRepairedDamage', 'dateCreated', 'nrOfPictures', 'postalCode',
       'lastSeen'],
      dtype='object')
```

In [7]:
```
car.columns =["date_crawled","name","seller","offer_type","price","ab_test","vehi
                  "power_ps","model","kilometer","registration_month","fuel_type"
                  "picture_number","postalcode","last_seen"]
```

In [ ]:
```
car
```

In [5]:
```
car.loc[0:5]
```

Out[5]:

	date_crawled	name	seller	offer_type	price	ab_
0	24-03-16 11:52	Golf_3_1.6	privat	Angebot	480.0	
1	24-03-16 10:58	A5_Sportback_2.7_Tdi	privat	Angebot	18300.0	
2	14-03-16 12:52	Jeep_Grand_Cherokee_"Overland"	privat	Angebot	9800.0	
3	17-03-16 16:54	GOLF_4_1_4__3TÜRER	privat	Angebot	1500.0	
4	31-03-16 17:25	Skoda_Fabia_1.4_TDI_PD_Classic	privat	Angebot	3600.0	
5	04-04-16 17:36	BMW_316i___e36_Limousine___Bastlerfahrzeug__Ex...	privat	Angebot	650.0	

In [6]:
```
car.isnull().sum()
```

Out[6]:
```
date_crawled              0
name                      0
seller                    1
offer_type                1
price                     1
ab_test                   1
vehicle_test          37870
registration_year         2
gear_box              20210
power_ps                  1
model                 20485
kilometer                 1
registration_month        2
fuel_type             33388
brand                     2
not_repaired_damage   72062
date_created              2
picture_number            2
postalcode                2
```

In [ ]:
```
car["vehicle_test"].isnull().sum()
```

In [ ]:
```
car.describe(include="all")
```

In [ ]:
```
car["name"].nunique()
```

In [28]:
```
car
```

Out[28]:

	date_crawled	name	seller	offer_type	price	ab
0	24-03-16 11:52	Golf_3_1.6	privat	Angebot	480.0	
1	24-03-16 10:58	A5_Sportback_2.7_Tdi	privat	Angebot	18300.0	
2	14-03-16 12:52	Jeep_Grand_Cherokee_"Overland"	privat	Angebot	9800.0	
3	17-03-16 16:54	GOLF_4_1_4__3TÜRER	privat	Angebot	1500.0	
4	31-03-16 17:25	Skoda_Fabia_1.4_TDI_PD_Classic	privat	Angebot	3600.0	
...	...	...	...	...	...	
371534	14-03-16 17:48	Suche_t4___vito_ab_6_sitze	privat	Angebot	2200.0	
371535	05-03-16 19:56	Smart_smart_leistungssteigerung_100ps	privat	Angebot	1199.0	
371536	19-03-16 18:57	Volkswagen_Multivan_T4_TDI_7DC_UY2	privat	Angebot	9200.0	
371537	20-03-16 19:41	VW_Golf_Kombi_1_9l_TDI	privat	Angebot	3400.0	
371538	07-03-16 19:39	BMW_M135i_vollausgestattet_NP_52.720____Euro	privat	Angebot	28990.0	c

371539 rows × 20 columns

In [8]:
```
import numpy as np
bridge = ["seller", "offer_type", "picture_number"]
car[bridge]
```

Out[8]:

	seller	offer_type	picture_number
0	privat	Angebot	0.0
1	privat	Angebot	0.0
2	privat	Angebot	0.0
3	privat	Angebot	0.0
4	privat	Angebot	0.0
...	...	...	...
371534	privat	Angebot	0.0
371535	privat	Angebot	0.0
371536	privat	Angebot	0.0
371537	privat	Angebot	0.0
371538	privat	Angebot	0.0

371539 rows × 3 columns

In [9]:
```
car["seller"].value_counts()
```

Out[9]:
```
privat        371534
gewerblich         3
golf               1
Name: seller, dtype: int64
```

In [10]:
```
car["offer_type"].value_counts()
```

Out[10]:
```
Angebot    371525
Gesuch         12
150000          1
Name: offer_type, dtype: int64
```

In [11]:
```
car["picture_number"].value_counts()
```

Out[11]:
```
0.0    371537
Name: picture_number, dtype: int64
```

In [12]:
```
cars = car.drop(bridge, axis=1)
```

In [13]:
```
cars
```

Out[13]:

	date_crawled	name	price	ab_test	vehicle_test
0	24-03-16 11:52	Golf_3_1.6	480.0	test	NaN
1	24-03-16 10:58	A5_Sportback_2.7_Tdi	18300.0	test	coupe
2	14-03-16 12:52	Jeep_Grand_Cherokee_"Overland"	9800.0	test	suv
3	17-03-16 16:54	GOLF_4_1_4__3TÜRER	1500.0	test	kleinwagen
4	31-03-16 17:25	Skoda_Fabia_1.4_TDI_PD_Classic	3600.0	test	kleinwagen
...	...	...	...	...	...
371534	14-03-16 17:48	Suche_t4___vito_ab_6_sitze	2200.0	test	NaN
371535	05-03-16 19:56	Smart_smart_leistungssteigerung_100ps	1199.0	test	cabrio
371536	19-03-16 18:57	Volkswagen_Multivan_T4_TDI_7DC_UY2	9200.0	test	bus
371537	20-03-16 19:41	VW_Golf_Kombi_1_9l_TDI	3400.0	test	kombi
371538	07-03-16 19:39	BMW_M135i_vollausgestattet_NP_52.720____Euro	28990.0	control	limousine

371539 rows × 17 columns

In [14]:
```
cars["price"].head(10)
```

Out[14]:
```
0      480.0
1    18300.0
2     9800.0
3     1500.0
4     3600.0
5      650.0
6     2200.0
7        0.0
8    14500.0
9      999.0
Name: price, dtype: float64
```

In [ ]:
```
autos.rename({"odometer": "odometer_km"}, axis=1, inplace=True)
```

In [72]:
```
cars["kilometer"].value_counts(dropna=False)
```

Out[72]:
```
150000    219434
125000     34673
150000     21368
100000     14517
90000      11457
80000      10054
70000       8954
60000       7910
50000       6977
5000        6443
40000       5841
30000       5530
20000       5205
125000      3394
10000       1776
100000      1403
90000       1067
80000        999
70000        819
60000        759
50000        639
5000         627
40000        536
30000        511
20000        471
10000        173
NaN            2
Name: kilometer, dtype: int64
```

In [73]:
```
cars["kilometer"].value_counts()
```

Out[73]:
```
150000    219434
125000     34673
150000     21368
100000     14517
90000      11457
80000      10054
70000       8954
60000       7910
50000       6977
5000        6443
40000       5841
30000       5530
20000       5205
125000      3394
10000       1776
100000      1403
90000       1067
80000        999
70000        819
60000        759
50000        639
5000         627
40000        536
30000        511
20000        471
10000        173
Name: kilometer, dtype: int64
```

In [74]:
```
kil_name = ["name", "kilometer"]
cars[kil_name].sample(10)
```

Out[74]:

	name	kilometer
217826	Mazda_MX_5_2.0_MZR_Niseko_inkl._Hardtop__Top_g...	80000
225139	Volkswagen_Golf_1.6	150000
280557	Audi_A4_Avant_2.0_TDI_DPF_multitronic	150000
105953	nissan_micra_5_tuerer_kein_rost_nix_kein_tuev	125000
150491	Seat_Ibiza_ST_1.6_TDI_CR_Sport	80000
244666	Skoda_Octavia_Combi_1.4_TSI_IMPULS_EDITION	50000
183668	Volkswagen_Passat_2.0_Turbo_FSI_Autom._Highlin...	150000
110310	Langstrecke_Technisch_gut!	150000
190108	Ford_fiesta	150000
161080	BMW_118	125000

In [75]:
```
comb = ["name", "model"]
inv_kil = cars["kilometer"]
inv_kil_bool = inv_kil == "30-03-16 0:44"
bad_row = cars.loc[inv_kil_bool, "kilometer"]
bad_row
```

Out[75]:
```
Series([], Name: kilometer, dtype: object)
```

In [76]:
```
comb = ["name", "model"]
inv_kil = cars["kilometer"]
inv_kil_bool = inv_kil == "30-03-16 0:44"
cars.loc[inv_kil_bool, "kilometer"] = np.nan
```

In [77]:
```
inv_kil = cars["kilometer"].notnull()
clean_car = cars.loc[inv_kil]
```

In [78]:
```
clean_car["kilometer"].value_counts()
```

Out[78]:
```
150000    219434
125000     34673
150000     21368
100000     14517
90000      11457
80000      10054
70000       8954
60000       7910
50000       6977
5000        6443
40000       5841
30000       5530
20000       5205
125000      3394
10000       1776
100000      1403
90000       1067
80000        999
70000        819
60000        759
50000        639
5000         627
40000        536
30000        511
20000        471
10000        173
Name: kilometer, dtype: int64
```

In [79]:
```
clean_car["price"].value_counts()
clean_car["price"].describe()
```

Out[79]:
```
count    3.715370e+05
mean     1.729549e+04
std      3.587910e+06
min      0.000000e+00
25%      1.150000e+03
50%      2.950000e+03
75%      7.200000e+03
max      2.147484e+09
Name: price, dtype: float64
```

In [80]:
```
clean_car["price"].nunique()
```

Out[80]:
```
5597
```

In [81]:
```
clean_car["price"].value_counts().sort_index
```

Out[81]:
```
<bound method Series.sort_index of 0.0         10778
500.0          5670
1500.0         5394
1000.0         4649
1200.0         4594
               ...
349000.0          1
8889.0            1
3440.0            1
1997.0            1
10985.0           1
Name: price, Length: 5597, dtype: int64>
```

In [82]:
```
clean_car["price"].value_counts().sort_index(ascending=False).head(15)
```

Out[82]:
```
2.147484e+09     1
1.000000e+08    15
9.900000e+07     1
7.418530e+07     1
3.254546e+07     1
2.732222e+07     1
1.400050e+07     1
1.234568e+07     9
1.111111e+07    10
1.001001e+07     1
1.000000e+07     8
9.999999e+06     3
3.895000e+06     1
3.890000e+06     1
2.995000e+06     1
Name: price, dtype: int64
```

In [83]:
```
clean_car = clean_car[clean_car["price"].between(1,351000)]
```

In [84]:
```
clean_car.shape
```

Out[84]:
```
(360644, 17)
```

In [89]:
```
dates = ["date_crawled", "registration_year", "registration_month", "date_created
clean_car[dates].sample(10)
```

Out[89]:

	date_crawled	registration_year	registration_month	date_created	last_seen
314524	29-03-16 19:45	1999.0	8.0	29-03-16 0:00	06-04-16 5:44
203176	15-03-16 15:56	2000.0	3.0	15-03-16 0:00	17-03-16 11:18
332204	07-03-16 14:46	2000.0	9.0	07-03-16 0:00	07-04-16 5:45
136590	22-03-16 17:56	2002.0	7.0	22-03-16 0:00	22-03-16 17:56
288708	16-03-16 1:00	2000.0	0.0	16-03-16 0:00	31-03-16 7:16
64471	02-04-16 12:38	2000.0	6.0	02-04-16 0:00	06-04-16 10:17
95518	27-03-16 19:57	2003.0	5.0	27-03-16 0:00	27-03-16 19:57
274484	20-03-16 11:50	1999.0	7.0	20-03-16 0:00	21-03-16 14:16
209163	28-03-16 0:58	2003.0	8.0	28-03-16 0:00	28-03-16 9:42
206929	27-03-16 17:58	1989.0	1.0	27-03-16 0:00	01-04-16 18:24

In [70]:
```
clean_car["date_crawled"].str[:8].value_counts(normalize=True, dropna=False).sort
```

Out[70]:
```
31-03-16    0.031871
30-03-16    0.033534
29-03-16    0.034125
28-03-16    0.035062
27-03-16    0.030226
26-03-16    0.031976
25-03-16    0.032802
24-03-16    0.029916
23-03-16    0.032001
22-03-16    0.032492
21-03-16    0.035681
20-03-16    0.036402
19-03-16    0.035270
18-03-16    0.013118
17-03-16    0.031646
16-03-16    0.030207
15-03-16    0.033424
14-03-16    0.036329
13-03-16    0.015786
12-03-16    0.036241
11-03-16    0.032772
10-03-16    0.032644
09-03-16    0.034117
08-03-16    0.033468
07-04-16    0.001617
07-03-16    0.035658
06-04-16    0.003128
06-03-16    0.014482
05-04-16    0.012780
05-03-16    0.025546
04-04-16    0.037627
03-04-16    0.038814
02-04-16    0.035093
01-04-16    0.034144
Name: date_crawled, dtype: float64
```

In [119]:
```
clean_car["date_created"].value_counts(normalize=True, dropna=True).sort_values(a
```

Out[119]:
```
03-04-16 0:00    0.039002
04-04-16 0:00    0.037735
20-03-16 0:00    0.036490
12-03-16 0:00    0.036077
21-03-16 0:00    0.035775
                   ...
06-01-16 0:00    0.000003
17-11-15 0:00    0.000003
10-11-15 0:00    0.000003
17-12-15 0:00    0.000003
18-06-15 0:00    0.000003
Name: date_created, Length: 114, dtype: float64
```

In [60]:
```
clean_car.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 360644 entries, 0 to 371538
Data columns (total 17 columns):
 #   Column               Non-Null Count   Dtype  
---  ------               --------------   -----  
 0   date_crawled         360644 non-null  object 
 1   name                 360644 non-null  object 
 2   price                360644 non-null  float64
 3   ab_test              360644 non-null  object 
 4   vehicle_test         326549 non-null  object 
 5   registration_year    360644 non-null  float64
 6   gear_box             342954 non-null  object 
 7   power_ps             360644 non-null  float64
 8   model                342386 non-null  object 
 9   kilometer            360644 non-null  object 
 10  registration_month   360644 non-null  float64
 11  fuel_type            330735 non-null  object 
 12  brand                360644 non-null  object 
 13  not_repaired_damage  293917 non-null  object 
 14  date_created         360644 non-null  object 
 15  postalcode           360644 non-null  float64
 16  last_seen            360644 non-null  object 
dtypes: float64(5), object(12)
memory usage: 57.6+ MB
```

In [93]:
```
dates = ["date_crawled", "registration_year", "registration_month", "date_created
clean_car[dates].sample(10)
```

Out[93]:

	date_crawled	registration_year	registration_month	date_created	last_seen
214536	02-04-16 17:52	1991.0	6.0	02-04-16 0:00	04-04-16 15:47
290248	29-03-16 12:36	2003.0	3.0	29-03-16 0:00	05-04-16 22:46
2557	25-03-16 11:49	1995.0	10.0	25-03-16 0:00	06-04-16 12:44
75484	08-03-16 21:47	2000.0	11.0	08-03-16 0:00	06-04-16 3:45
84020	30-03-16 14:45	1996.0	9.0	30-03-16 0:00	03-04-16 4:17
336976	02-04-16 22:46	2009.0	9.0	02-04-16 0:00	06-04-16 23:45
180071	02-04-16 19:52	2004.0	1.0	02-04-16 0:00	02-04-16 19:52
100747	28-03-16 16:58	1997.0	0.0	28-03-16 0:00	06-04-16 22:17
125589	29-03-16 15:57	2002.0	2.0	29-03-16 0:00	06-04-16 2:15
351666	31-03-16 20:53	2002.0	10.0	31-03-16 0:00	04-04-16 14:45

In [123]:
```
clean_car["registration_year"].value_counts(normalize=True, dropna=False).sort_va
```

Out[123]:
```
8200.0    0.000003
5600.0    0.000003
7800.0    0.000003
7777.0    0.000003
2222.0    0.000003
             ...
2001.0    0.054747
2006.0    0.055476
2005.0    0.060239
1999.0    0.061091
2000.0    0.064116
Name: registration_year, Length: 145, dtype: float64
```

In [166]:
```
clean_car["registration_year"].between(1900, 2016)
```

Out[166]:
```
0         True
1         True
2         True
3         True
4         True
          ...
371534    True
371535    True
371536    True
371537    True
371538    True
Name: registration_year, Length: 360644, dtype: bool
```

In [128]:
```
clean_car["registration_year"].between(1900, 2016).sum() / clean_car.shape[0]
```

Out[128]:
```
0.9612498752232118
```

In [127]:
```
clean_car["date_created"].describe()
```

Out[127]:
```
count            360644
unique              114
top       03-04-16 0:00
freq              14066
Name: date_created, dtype: object
```

In [126]:
```
clean_car["registration_year"].describe()
```

Out[126]:
```
count    360644.000000
mean       2004.433020
std          81.015993
min        1000.000000
25%        1999.000000
50%        2004.000000
75%        2008.000000
max        9999.000000
Name: registration_year, dtype: float64
```

In [131]:
```
the_car = clean_car["registration_year"].between(1900, 2016)
```

In [135]:
```
newly_cleaned = clean_car[the_car]
```

In [137]:
```
newly_cleaned["registration_year"].describe()
```

Out[137]:
```
count    346669.000000
mean       2002.896411
std           7.244917
min        1910.000000
25%        1999.000000
50%        2003.000000
75%        2008.000000
max        2016.000000
Name: registration_year, dtype: float64
```

In [142]:
```
newly_cleaned["brand"].value_counts(normalize=True).sort_values(ascending=False)
```

Out[142]:
```
volkswagen        0.211695
bmw               0.109868
opel              0.106407
mercedes_benz     0.096850
audi              0.089541
ford              0.068919
renault           0.047521
peugeot           0.030153
fiat              0.025690
seat              0.018660
skoda             0.015686
mazda             0.015384
smart             0.014331
citroen           0.013950
nissan            0.013598
toyota            0.012935
hyundai           0.009972
sonstige_autos    0.009493
mini              0.009384
volvo             0.009147
mitsubishi        0.008236
honda             0.007532
kia               0.006914
suzuki            0.006363
alfa_romeo        0.006309
porsche           0.006211
chevrolet         0.005022
chrysler          0.003862
dacia             0.002495
jeep              0.002192
land_rover        0.002169
daihatsu          0.002161
subaru            0.002117
jaguar            0.001734
saab              0.001465
daewoo            0.001457
trabant           0.001408
lancia            0.001301
rover             0.001272
lada              0.000597
Name: brand, dtype: float64
```

In [169]:
```
brand_count = newly_cleaned["brand"].value_counts(normalize=True)
brand_count_bool = brand_count[brand_count < 100]
common_brand = brand_count_bool.index
common_brand
```

Out[169]:
```
Index(['volkswagen', 'bmw', 'opel', 'mercedes_benz', 'audi', 'ford', 'renault',
       'peugeot', 'fiat', 'seat', 'skoda', 'mazda', 'smart', 'citroen',
       'nissan', 'toyota', 'hyundai', 'sonstige_autos', 'mini', 'volvo',
       'mitsubishi', 'honda', 'kia', 'suzuki', 'alfa_romeo', 'porsche',
       'chevrolet', 'chrysler', 'dacia', 'jeep', 'land_rover', 'daihatsu',
       'subaru', 'jaguar', 'saab', 'daewoo', 'trabant', 'lancia', 'rover',
       'lada'],
      dtype='object')
```

In [170]:
```
price_mean = {}
for checks in common_brand:
    the_brand = newly_cleaned["brand"]
    the_bool = the_brand == checks
    price = newly_cleaned.loc[the_bool, "price"]
    price_mean[checks] = int(price.mean())
price_mean
```

Out[170]:
```
{'volkswagen': 5400,
 'bmw': 8449,
 'opel': 2971,
 'mercedes_benz': 8558,
 'audi': 9086,
 'ford': 3696,
 'renault': 2437,
 'peugeot': 3267,
 'fiat': 2892,
 'seat': 4541,
 'skoda': 6530,
 'mazda': 4076,
 'smart': 3632,
 'citroen': 3734,
 'nissan': 4708,
 'toyota': 5340,
 'hyundai': 5567,
 'sonstige_autos': 14288,
 'mini': 10080,
 'volvo': 5238,
 'mitsubishi': 3407,
 'honda': 4005,
 'kia': 5855,
 'suzuki': 4044,
 'alfa_romeo': 4291,
 'porsche': 42258,
 'chevrolet': 7117,
 'chrysler': 4121,
 'dacia': 5922,
 'jeep': 11213,
 'land_rover': 17060,
 'daihatsu': 1775,
 'subaru': 4386,
 'jaguar': 13765,
 'saab': 3955,
 'daewoo': 1027,
 'trabant': 1900,
 'lancia': 3289,
 'rover': 1600,
 'lada': 3191}
```

Type *Markdown* and LaTeX: α^2

In [162]:
```
bmp_series = pd.Series(price_mean)
pd.DataFrame(bmp_series, columns=["mean_price"])
```

Out[162]:

	mean_price
volkswagen	5400
bmw	8449
opel	2971
mercedes_benz	8558
audi	9086
ford	3696

In [167]:
```
newly_cleaned
```

Out[167]:

	date_crawled	name	price	ab_test	vehicle_test
0	24-03-16 11:52	Golf_3_1.6	480.0	test	NaN
1	24-03-16 10:58	A5_Sportback_2.7_Tdi	18300.0	test	coupe
2	14-03-16 12:52	Jeep_Grand_Cherokee_"Overland"	9800.0	test	suv
3	17-03-16 16:54	GOLF_4_1_4__3TÜRER	1500.0	test	kleinwagen
4	31-03-16 17:25	Skoda_Fabia_1.4_TDI_PD_Classic	3600.0	test	kleinwagen
...	...	...	...	...	...
371534	14-03-16 17:48	Suche_t4___vito_ab_6_sitze	2200.0	test	NaN
371535	05-03-16 19:56	Smart_smart_leistungssteigerung_100ps	1199.0	test	cabrio
371536	19-03-16 18:57	Volkswagen_Multivan_T4_TDI_7DC_UY2	9200.0	test	bus
371537	20-03-16 19:41	VW_Golf_Kombi_1_9l_TDI	3400.0	test	kombi
371538	07-03-16 19:39	BMW_M135i_vollausgestattet_NP_52.720____Euro	28990.0	control	limousine

346669 rows × 17 columns

In [168]:
```
newly_cleaned["price"].value_counts().sort_values()
```

Out[168]:
```
10985.0       1
30790.0       1
18555.0       1
3425.0        1
5277.0        1
             ...
2500.0     4244
1200.0     4332
1000.0     4366
1500.0     5093
500.0      5468
Name: price, Length: 5500, dtype: int64
```

In [ ]:
```

```