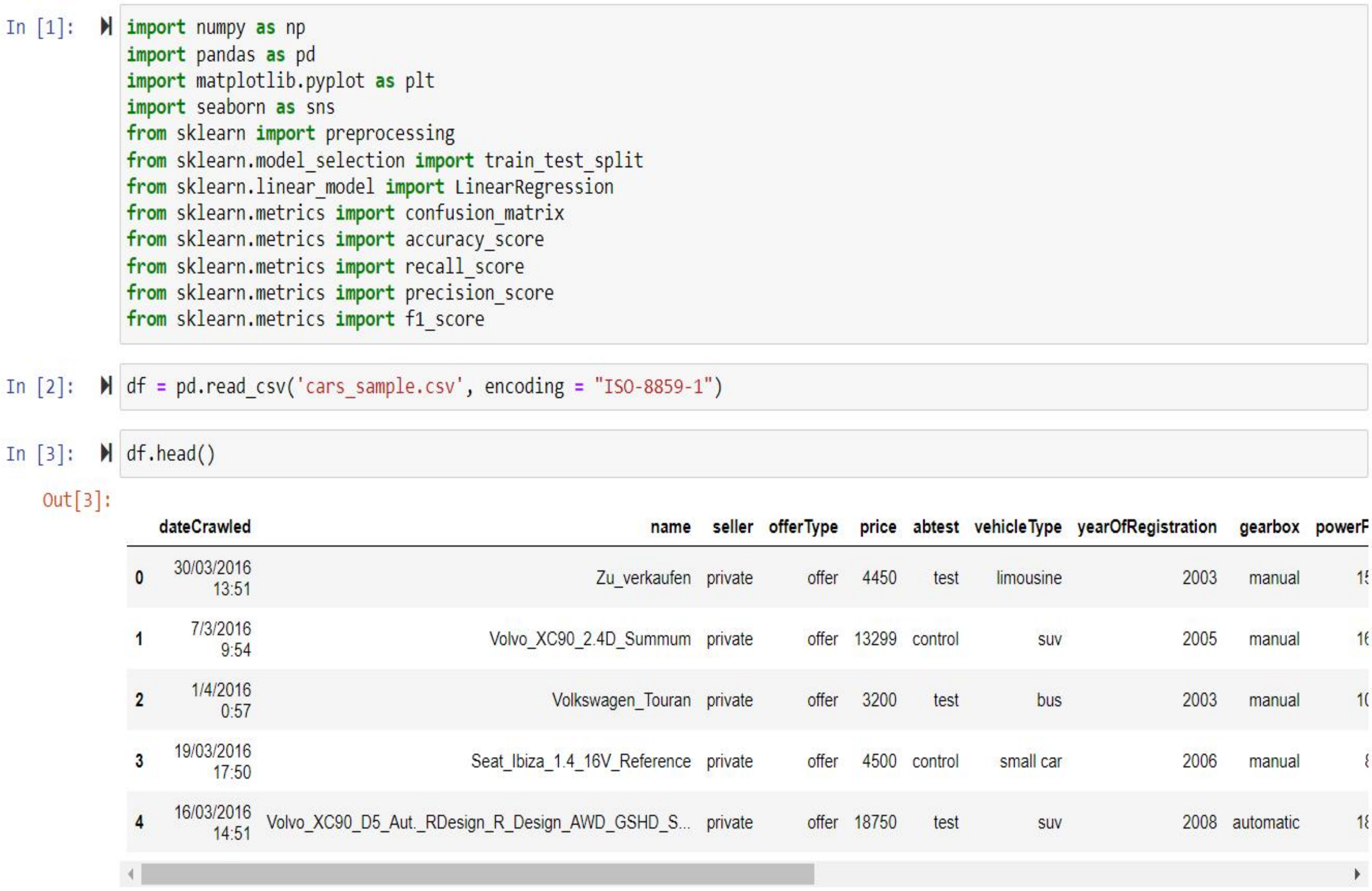
Pooja Agarwal
1905330

Lab 8

Q1: Find out if following variables are significant or insignificant and need to be dropped.
i) Seller-insignificant
ii) offerType-insignificant
iii) abtest-insignificant
Iv)vehicleType-significant
V)gearbox,
Vi)Model
Vii)Kilometer
Viii)Fueltype
Ix)Brand
X)notRepairedDamage



```
In [1]:  import numpy as np
         import pandas as pd
         import matplotlib.pyplot as plt
         import seaborn as sns
         from sklearn import preprocessing
         from sklearn.model_selection import train_test_split
         from sklearn.linear_model import LinearRegression
         from sklearn.metrics import confusion_matrix
         from sklearn.metrics import accuracy_score
         from sklearn.metrics import recall_score
         from sklearn.metrics import precision_score
         from sklearn.metrics import f1_score
```

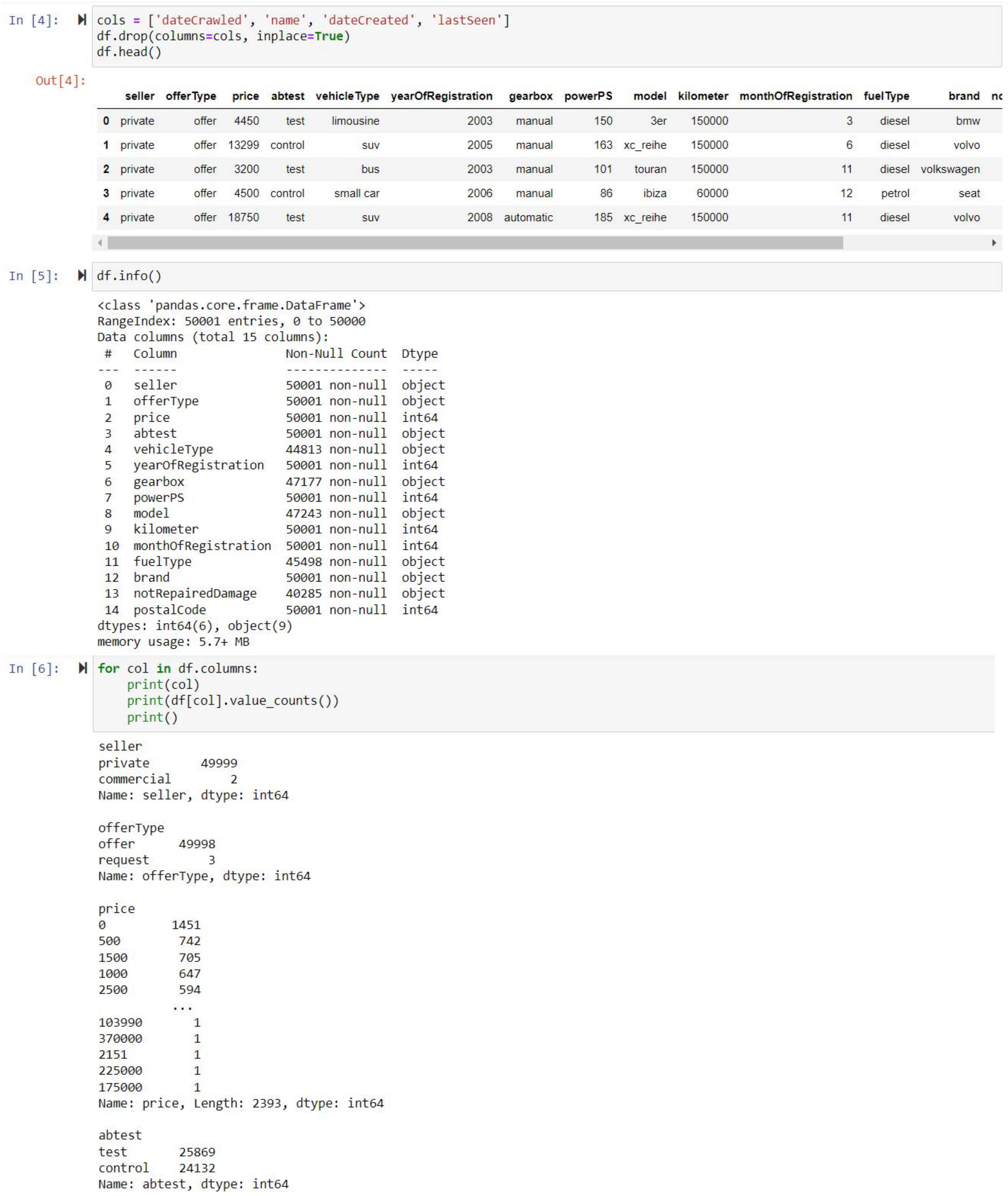
```
In [2]:  df = pd.read_csv('cars_sample.csv', encoding = "ISO-8859-1")
```

```
In [3]:  df.head()
```

Out[3]:

	dateCrawled	name	seller	offerType	price	abtest	vehicleType	yearOfRegistration	gearbox	powerF
0	30/03/2016 13:51	Zu_verkaufen	private	offer	4450	test	limousine	2003	manual	1
1	7/3/2016 9:54	Volvo_XC90_2.4D_Summum	private	offer	13299	control	suv	2005	manual	1
2	1/4/2016 0:57	Volkswagen_Touran	private	offer	3200	test	bus	2003	manual	1
3	19/03/2016 17:50	Seat_Ibiza_1.4_16V_Reference	private	offer	4500	control	small car	2006	manual	
4	16/03/2016 14:51	Volvo_XC90_D5_Aut._RDesign_R_Design_AWD_GSHD_S...	private	offer	18750	test	suv	2008	automatic	1

Pooja Agarwal
1905330



```
In [4]:  cols = ['dateCrawled', 'name', 'dateCreated', 'lastSeen']
         df.drop(columns=cols, inplace=True)
         df.head()
```

Out[4]:

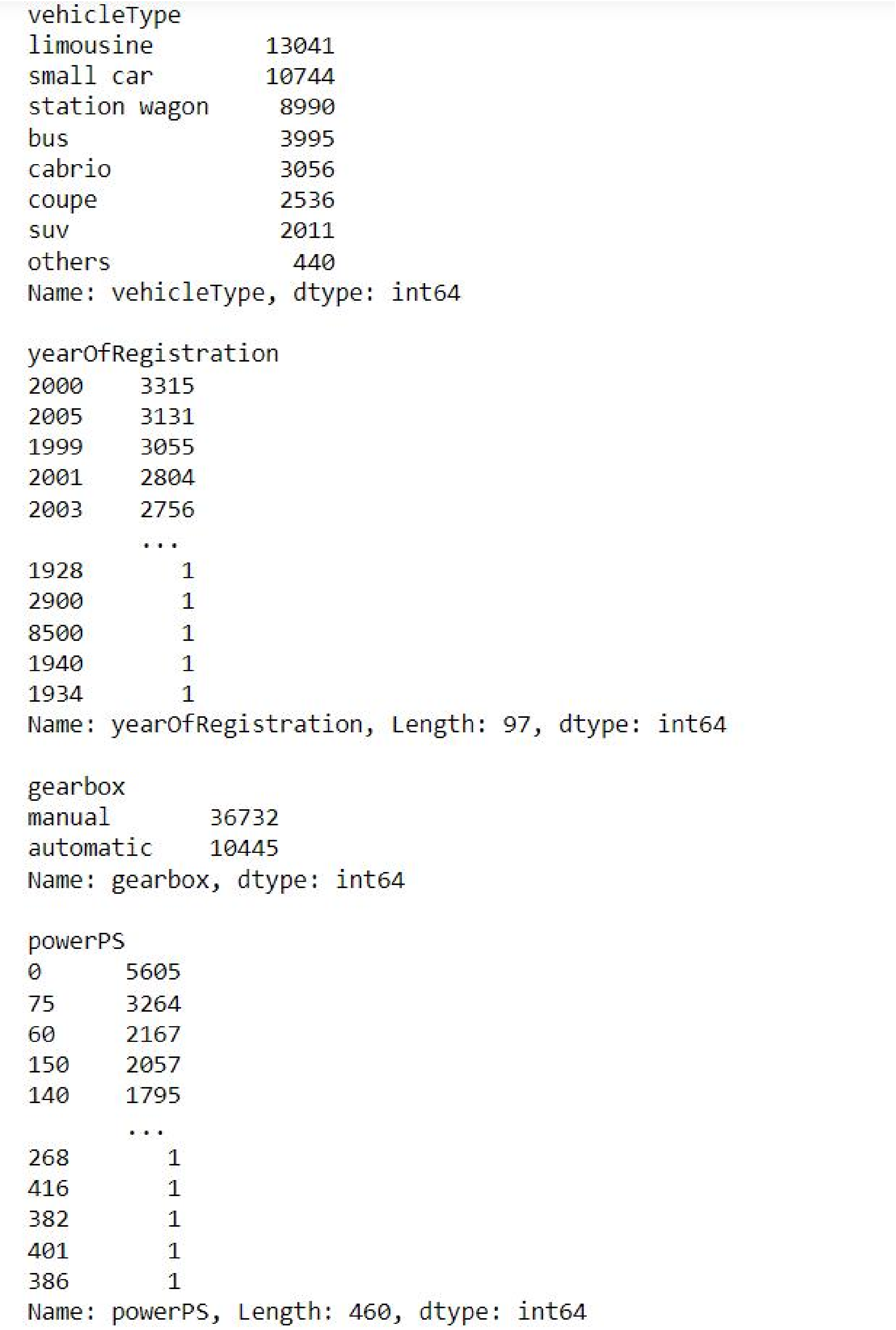
	seller	offerType	price	abtest	vehicleType	yearOfRegistration	gearbox	powerPS	model	kilometer	monthOfRegistration	fuelType	brand	nc
0	private	offer	4450	test	limousine	2003	manual	150	3er	150000	3	diesel	bmw	
1	private	offer	13299	control	suv	2005	manual	163	xc_reihe	150000	6	diesel	volvo	
2	private	offer	3200	test	bus	2003	manual	101	touran	150000	11	diesel	volkswagen	
3	private	offer	4500	control	small car	2006	manual	86	ibiza	60000	12	petrol	seat	
4	private	offer	18750	test	suv	2008	automatic	185	xc_reihe	150000	11	diesel	volvo	

```
In [5]:  df.info()
```

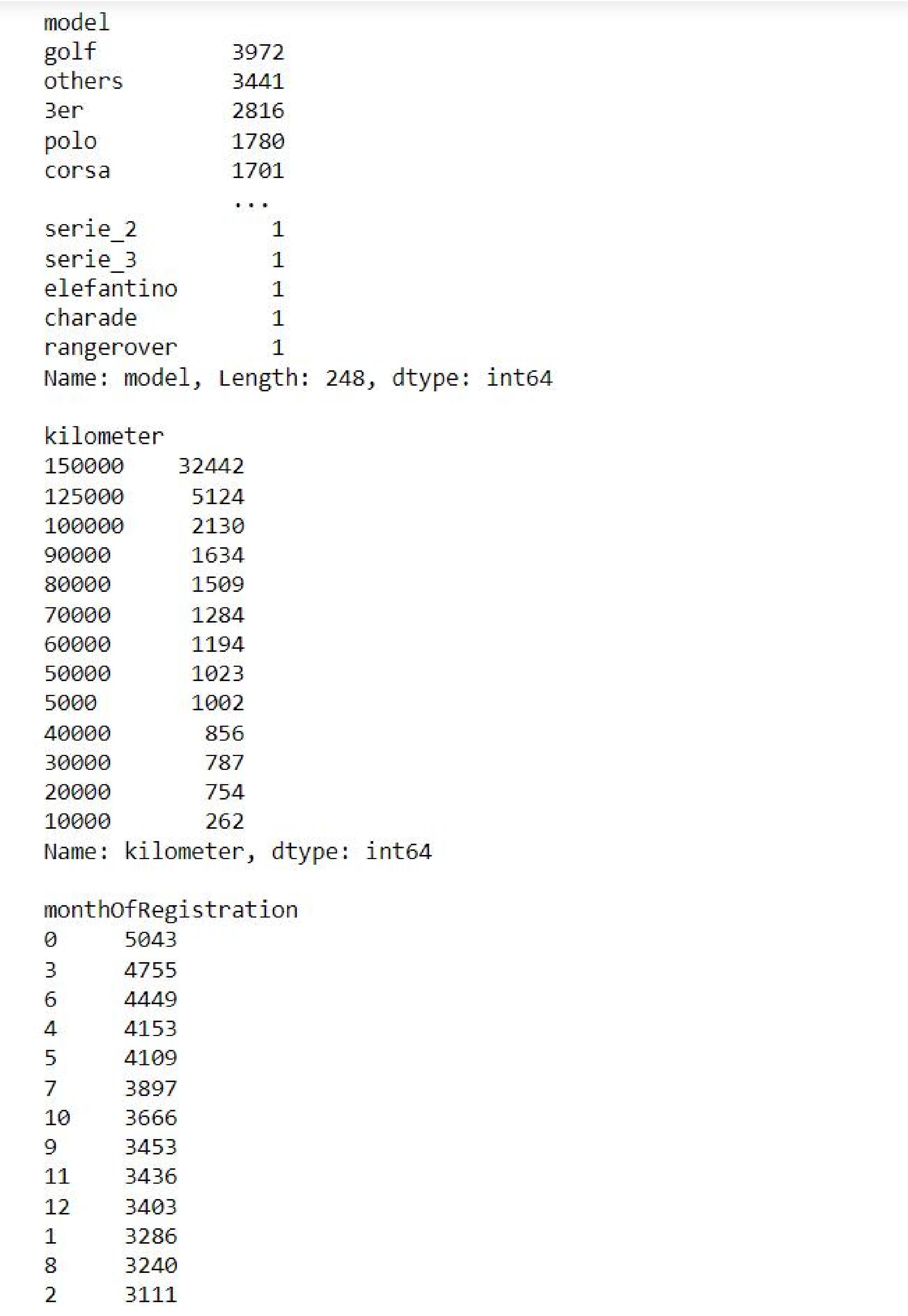
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50001 entries, 0 to 50000
Data columns (total 15 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   seller               50001 non-null  object
 1   offerType            50001 non-null  object
 2   price                50001 non-null  int64 
 3   abtest               50001 non-null  object
 4   vehicleType          44813 non-null  object
 5   yearOfRegistration   50001 non-null  int64 
 6   gearbox              47177 non-null  object
 7   powerPS              50001 non-null  int64 
 8   model                47243 non-null  object
 9   kilometer            50001 non-null  int64 
 10  monthOfRegistration  50001 non-null  int64 
 11  fuelType             45498 non-null  object
 12  brand                50001 non-null  object
 13  notRepairedDamage    40285 non-null  object
 14  postalCode           50001 non-null  int64 
dtypes: int64(6), object(9)
memory usage: 5.7+ MB
```

```
In [6]:  for col in df.columns:
             print(col)
             print(df[col].value_counts())
             print()
```

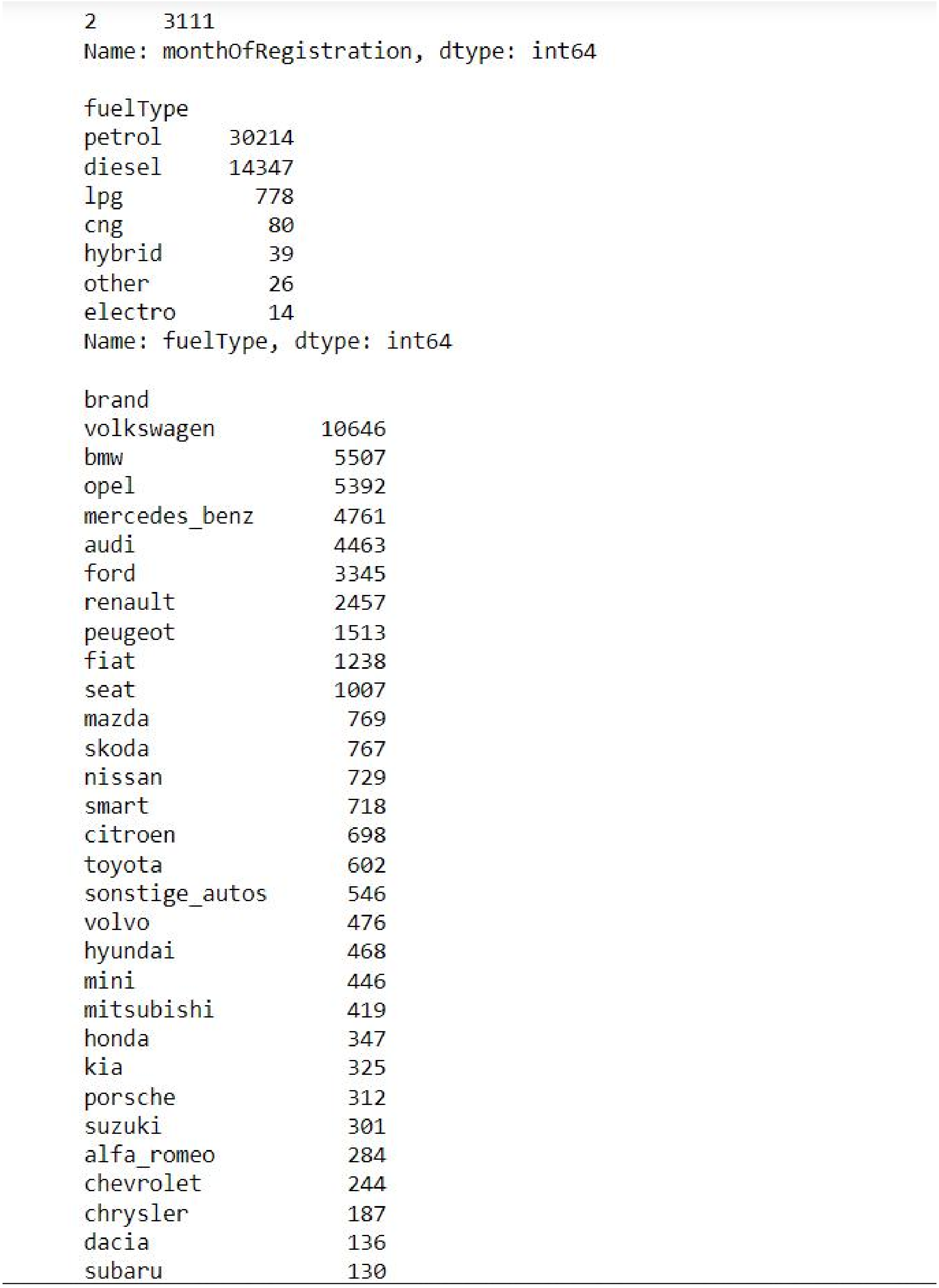
```
seller
private       49999
commercial        2
Name: seller, dtype: int64

offerType
offer      49998
request        3
Name: offerType, dtype: int64

price
0         1451
500        742
1500       705
1000       647
2500       594
          ... 
103990       1
370000       1
2151         1
225000       1
175000       1
Name: price, Length: 2393, dtype: int64

abtest
test       25869
control    24132
Name: abtest, dtype: int64
```



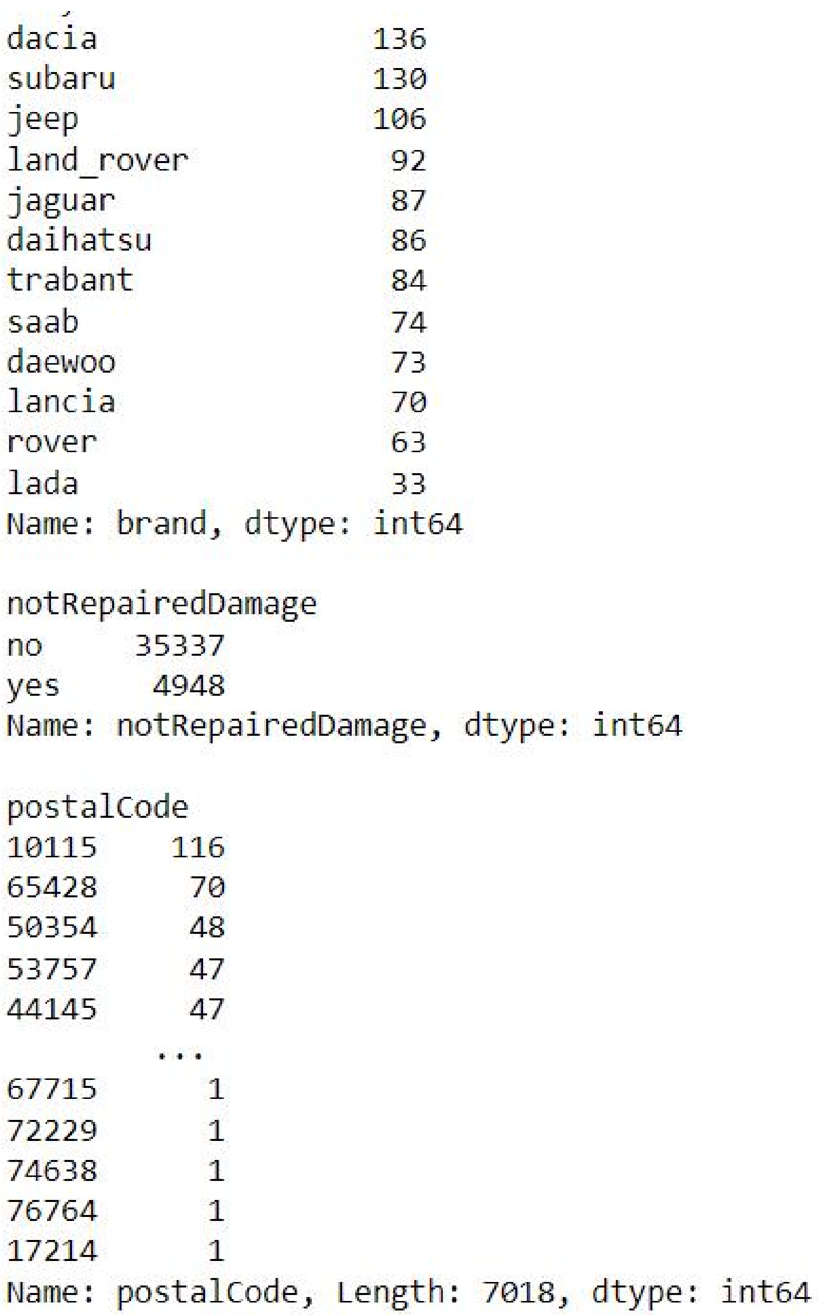
```
vehicleType
limousine        13041
small car        10744
station wagon     8990
bus               3995
cabrio            3056
coupe             2536
suv               2011
others             440
Name: vehicleType, dtype: int64

yearOfRegistration
2000    3315
2005    3131
1999    3055
2001    2804
2003    2756
        ...
1928       1
2900       1
8500       1
1940       1
1934       1
Name: yearOfRegistration, Length: 97, dtype: int64

gearbox
manual       36732
automatic    10445
Name: gearbox, dtype: int64

powerPS
0      5605
75     3264
60     2167
150    2057
140    1795
       ...
268       1
416       1
382       1
401       1
386       1
Name: powerPS, Length: 460, dtype: int64
```



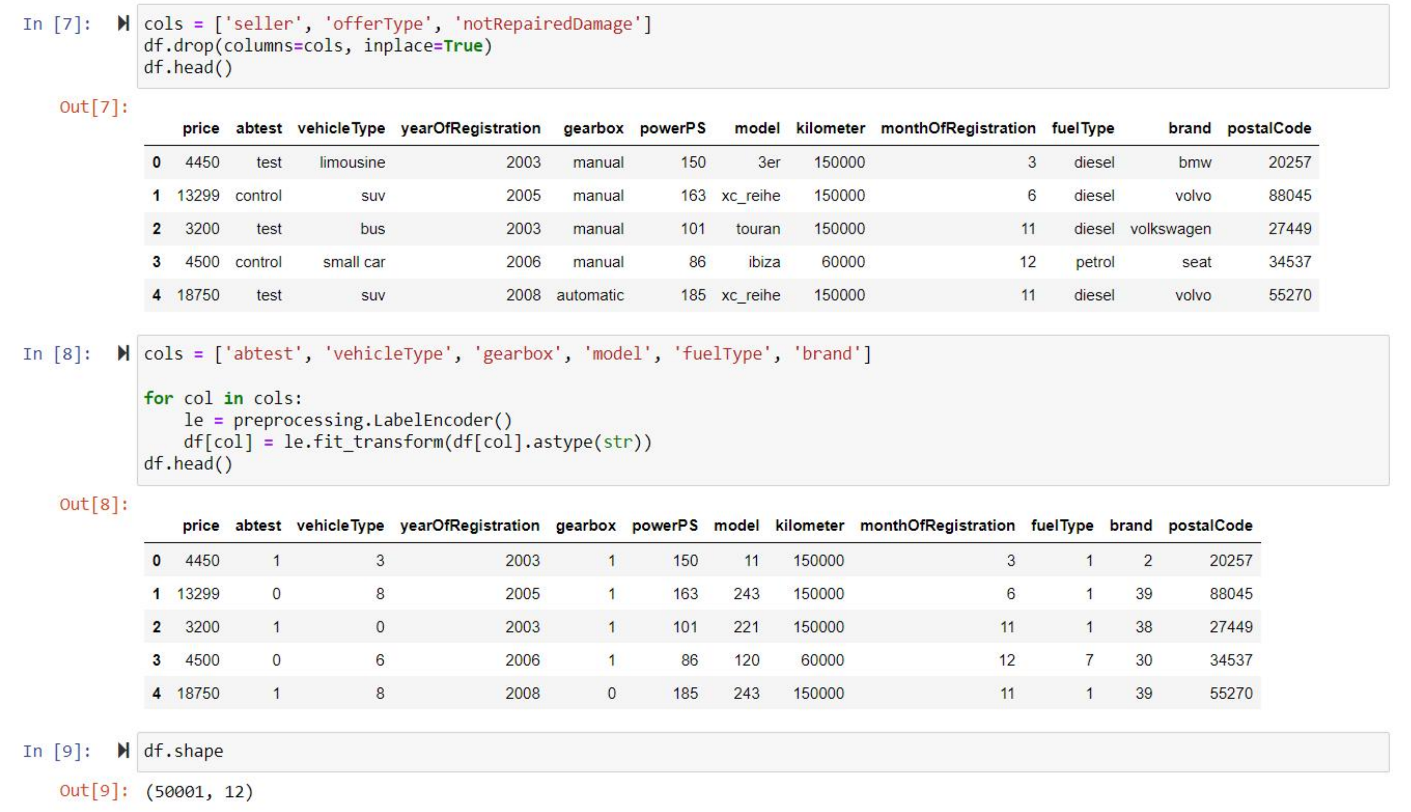
```
model
golf              3972
others            3441
3er               2816
polo              1780
corsa             1701
                  ...
serie_2              1
serie_3              1
elefantino           1
charade              1
rangerover           1
Name: model, Length: 248, dtype: int64

kilometer
150000    32442
125000     5124
100000     2130
90000      1634
80000      1509
70000      1284
60000      1194
50000      1023
5000       1002
40000       856
30000       787
20000       754
10000       262
Name: kilometer, dtype: int64

monthOfRegistration
0     5043
3     4755
6     4449
4     4153
5     4109
7     3897
10    3666
9     3453
11    3436
12    3403
1     3286
8     3240
2     3111
```



```
2     3111
Name: monthOfRegistration, dtype: int64

fuelType
petrol     30214
diesel     14347
lpg          778
cng           80
hybrid        39
other         26
electro       14
Name: fuelType, dtype: int64

brand
volkswagen        10646
bmw                5507
opel               5392
mercedes_benz      4761
audi               4463
ford               3345
renault            2457
peugeot            1513
fiat               1238
seat               1007
mazda               769
skoda               767
nissan              729
smart               718
citroen             698
toyota              602
sonstige_autos      546
volvo               476
hyundai             468
mini                446
mitsubishi          419
honda               347
kia                 325
porsche             312
suzuki              301
alfa_romeo          284
chevrolet           244
chrysler            187
dacia               136
subaru              130
```



```
dacia               136
subaru              130
jeep                106
land_rover           92
jaguar               87
daihatsu             86
trabant              84
saab                 74
daewoo               73
lancia               70
rover                63
lada                 33
Name: brand, dtype: int64

notRepairedDamage
no     35337
yes     4948
Name: notRepairedDamage, dtype: int64

postalCode
10115    116
65428     70
50354     48
53757     47
44145     47
        ...
67715      1
72229      1
74638      1
76764      1
17214      1
Name: postalCode, Length: 7018, dtype: int64
```



In [7]:
```
cols = ['seller', 'offerType', 'notRepairedDamage']
df.drop(columns=cols, inplace=True)
df.head()
```

Out[7]:

	price	abtest	vehicleType	yearOfRegistration	gearbox	powerPS	model	kilometer	monthOfRegistration	fuelType	brand	postalCode
0	4450	test	limousine	2003	manual	150	3er	150000	3	diesel	bmw	20257
1	13299	control	suv	2005	manual	163	xc_reihe	150000	6	diesel	volvo	88045
2	3200	test	bus	2003	manual	101	touran	150000	11	diesel	volkswagen	27449
3	4500	control	small car	2006	manual	86	ibiza	60000	12	petrol	seat	34537
4	18750	test	suv	2008	automatic	185	xc_reihe	150000	11	diesel	volvo	55270

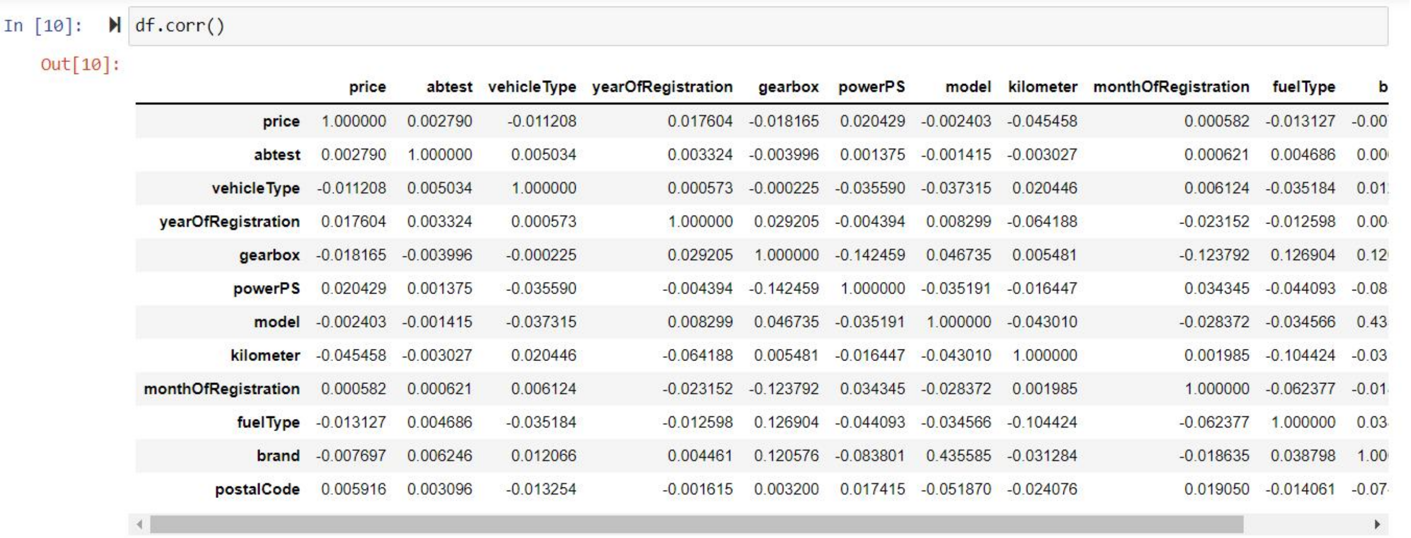
In [8]:
```
cols = ['abtest', 'vehicleType', 'gearbox', 'model', 'fuelType', 'brand']

for col in cols:
    le = preprocessing.LabelEncoder()
    df[col] = le.fit_transform(df[col].astype(str))
df.head()
```

Out[8]:

	price	abtest	vehicleType	yearOfRegistration	gearbox	powerPS	model	kilometer	monthOfRegistration	fuelType	brand	postalCode
0	4450	1	3	2003	1	150	11	150000	3	1	2	20257
1	13299	0	8	2005	1	163	243	150000	6	1	39	88045
2	3200	1	0	2003	1	101	221	150000	11	1	38	27449
3	4500	0	6	2006	1	86	120	60000	12	7	30	34537
4	18750	1	8	2008	0	185	243	150000	11	1	39	55270

In [9]:
```
df.shape
```

Out[9]:
```
(50001, 12)
```

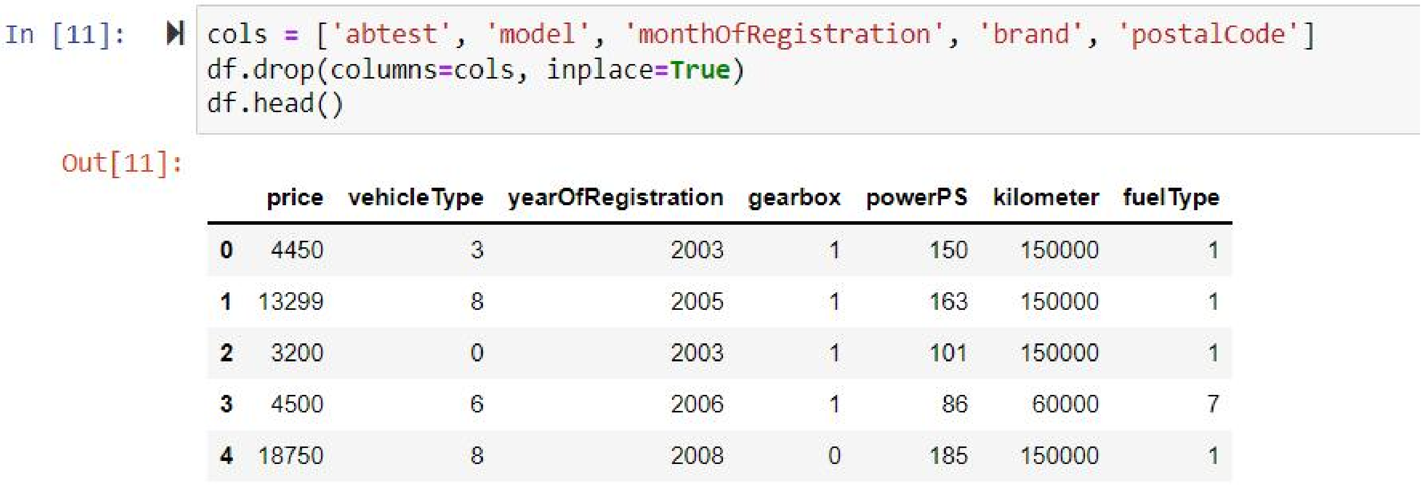


In [10]:
```
df.corr()
```

Out[10]:

	price	abtest	vehicleType	yearOfRegistration	gearbox	powerPS	model	kilometer	monthOfRegistration	fuelType	b
price	1.000000	0.002790	-0.011208	0.017604	-0.018165	0.020429	-0.002403	-0.045458	0.000582	-0.013127	-0.00
abtest	0.002790	1.000000	0.005034	0.003324	-0.003996	0.001375	-0.001415	-0.003027	0.000621	0.004686	0.00
vehicleType	-0.011208	0.005034	1.000000	0.000573	-0.000225	-0.035590	-0.037315	0.020446	0.006124	-0.035184	0.01
yearOfRegistration	0.017604	0.003324	0.000573	1.000000	0.029205	-0.004394	0.008299	-0.064188	-0.023152	-0.012598	0.00
gearbox	-0.018165	-0.003996	-0.000225	0.029205	1.000000	-0.142459	0.046735	0.005481	-0.123792	0.126904	0.12
powerPS	0.020429	0.001375	-0.035590	-0.004394	-0.142459	1.000000	-0.035191	-0.016447	0.034345	-0.044093	-0.08
model	-0.002403	-0.001415	-0.037315	0.008299	0.046735	-0.035191	1.000000	-0.043010	-0.028372	-0.034566	0.43
kilometer	-0.045458	-0.003027	0.020446	-0.064188	0.005481	-0.016447	-0.043010	1.000000	0.001985	-0.104424	-0.03
monthOfRegistration	0.000582	0.000621	0.006124	-0.023152	-0.123792	0.034345	-0.028372	0.001985	1.000000	-0.062377	-0.01
fuelType	-0.013127	0.004686	-0.035184	-0.012598	0.126904	-0.044093	-0.034566	-0.104424	-0.062377	1.000000	0.03
brand	-0.007697	0.006246	0.012066	0.004461	0.120576	-0.083801	0.435585	-0.031284	-0.018635	0.038798	1.00
postalCode	0.005916	0.003096	-0.013254	-0.001615	0.003200	0.017415	-0.051870	-0.024076	0.019050	-0.014061	-0.07

Q2: Drop insignificant variables from dataframe ‘cars’.



In [11]:

```
cols = ['abtest', 'model', 'monthOfRegistration', 'brand', 'postalCode']
df.drop(columns=cols, inplace=True)
df.head()
```

Out[11]:

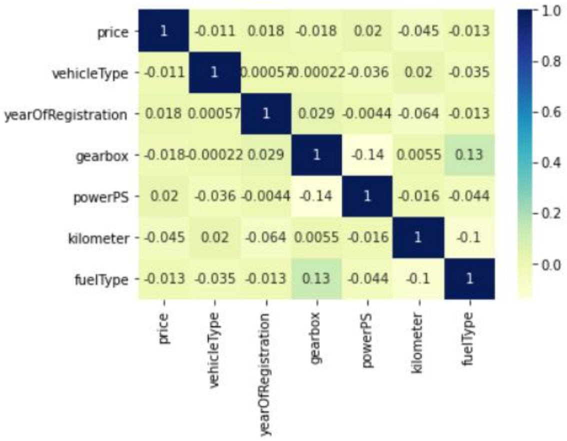
	price	vehicleType	yearOfRegistration	gearbox	powerPS	kilometer	fuelType
0	4450	3	2003	1	150	150000	1
1	13299	8	2005	1	163	150000	1
2	3200	0	2003	1	101	150000	1
3	4500	6	2006	1	86	60000	7
4	18750	8	2008	0	185	150000	1

Pooja Agarwal
1905330

Q3: Find correlation between all numerical variables and find which variable has the highest correlation with price.

```
In [12]:    cor = df.corr()
```

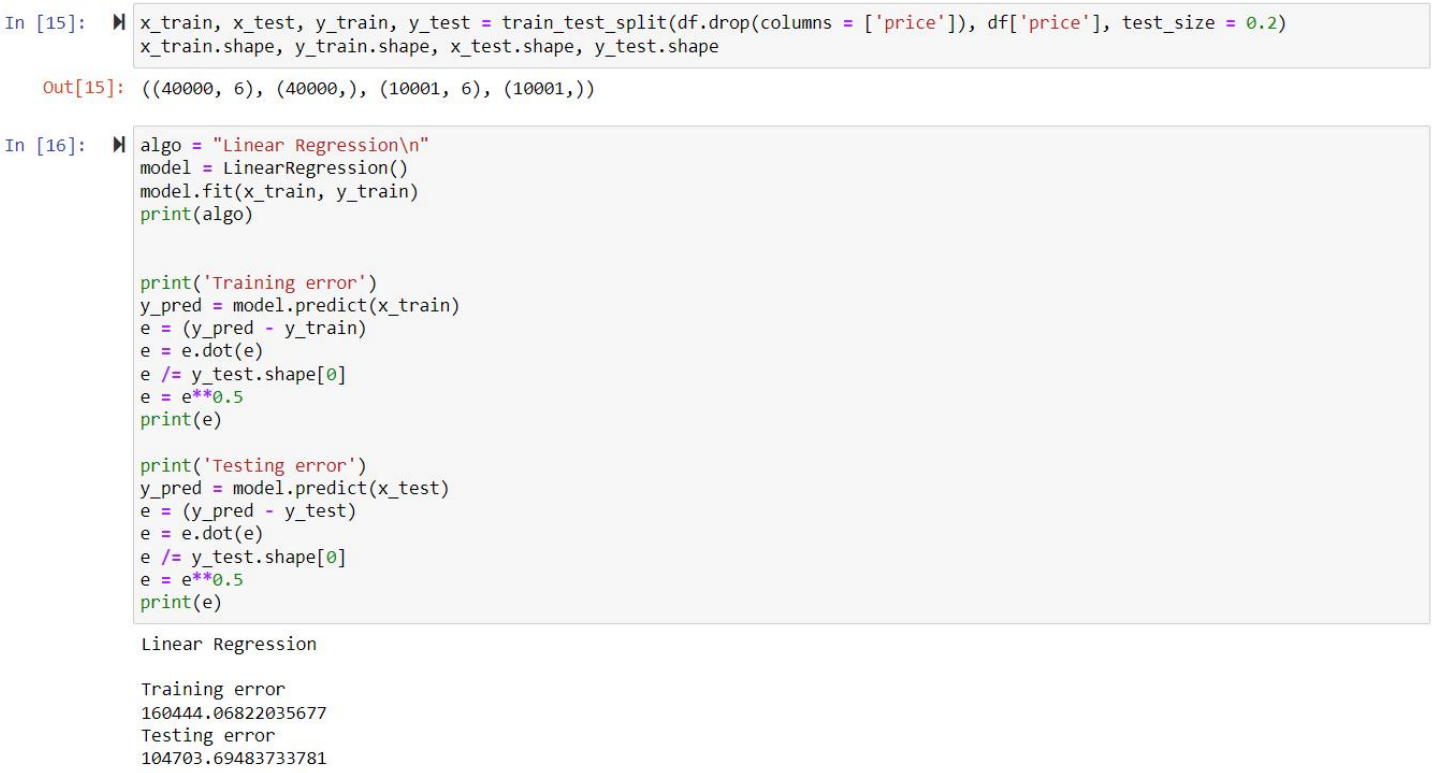
```
In [13]:    sns.heatmap(cor, cmap="YlGnBu", annot=True)
            plt.show()
```



```
In [14]:    print(cor['price'])
            print('Highest: kilometer (abs vale of 0.045458)')
```

```
price                  1.000000
vehicleType           -0.011208
yearOfRegistration     0.017604
gearbox               -0.018165
powerPS                0.020429
kilometer             -0.045458
fuelType              -0.013127
Name: price, dtype: float64
Highest: kilometer (abs vale of 0.045458)
```

Q4: Calculate the training data and testing data score using a linear regression model.



```
In [15]:  x_train, x_test, y_train, y_test = train_test_split(df.drop(columns = ['price']), df['price'], test_size = 0.2)
          x_train.shape, y_train.shape, x_test.shape, y_test.shape

Out[15]:  ((40000, 6), (40000,), (10001, 6), (10001,))
```

```
In [16]:  algo = "Linear Regression\n"
          model = LinearRegression()
          model.fit(x_train, y_train)
          print(algo)


          print('Training error')
          y_pred = model.predict(x_train)
          e = (y_pred - y_train)
          e = e.dot(e)
          e /= y_test.shape[0]
          e = e**0.5
          print(e)

          print('Testing error')
          y_pred = model.predict(x_test)
          e = (y_pred - y_test)
          e = e.dot(e)
          e /= y_test.shape[0]
          e = e**0.5
          print(e)
```

```
Linear Regression

Training error
160444.06822035677
Testing error
104703.69483733781
```