ML

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import math
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
df=pd.read_csv(r"C:\Users\Nishant kmr\Downloads\ML Data - data.csv")
```

In [3]:
```
pd.set_option('display.max_rows', 100)

pd.set_option('display.max_columns', 100)
```

In [ ]:
```

```

In [4]:
```
(df.isnull().mean()*100).sort_values()
```

Out[4]:
```
appointmentId                                          0.000000
fuel_type                                              0.000000
engineTransmission_gearShifting_value                  0.000000
engineTransmission_clutch_value                        0.000000
engineTransmission_engineBlowByBackCompression_cc_value_0      0.000000
engineTransmission_engineBlowByBackCompression_value           0.000000
engineTransmission_exhaustSmoke_value                  0.000000
engineTransmission_engineSound_value                   0.000000
engineTransmission_engineMounting_value                0.000000
odometer_reading                                       0.000000
engineTransmission_coolant_value                       0.000000
engineTransmission_engine_value                        0.000000
rating_engineTransmission                              0.000000
inspectionStartTime                                    0.000000
year                                                   0.000000
month                                                  0.000000
engineTransmission_battery_value                       0.000000
engineTransmission_engineOil                           0.000000
engineTransmission_engineoilLevelDipstick_value        0.000000
engineTransmission_engineSound_cc_value_0             27.471776
engineTransmission_engineOil_cc_value_0               29.459840
engineTransmission_engineSound_cc_value_1             48.663854
engineTransmission_clutch_cc_value_0                  55.281864
engineTransmission_engineOil_cc_value_1               58.170829
engineTransmission_coolant_cc_value_0                 61.850458
engineTransmission_engineMounting_cc_value_0          64.697609
engineTransmission_engine_cc_value_0                  65.522485
engineTransmission_engineOil_cc_value_2               74.938229
engineTransmission_engineSound_cc_value_2             76.918691
engineTransmission_exhaustSmoke_cc_value_0            78.880146
engineTransmission_engine_cc_value_1                  80.674345
engineTransmission_clutch_cc_value_1                  83.639336
engineTransmission_comments_value_0                   84.544038
engineTransmission_engineOil_cc_value_3               85.775649
engineTransmission_gearShifting_cc_value_0            86.832402
engineTransmission_battery_cc_value_0                 86.931235
engineTransmission_engine_cc_value_2                  90.975786
engineTransmission_coolant_cc_value_1                 91.017600
engineTransmission_engineSound_cc_value_3             91.819668
engineTransmission_engineOil_cc_value_4               93.264150
engineTransmission_comments_value_1                   95.027939
engineTransmission_engine_cc_value_3                  96.563652
engineTransmission_clutch_cc_value_2                  96.970388
engineTransmission_gearShifting_cc_value_1            97.069221
engineTransmission_engineSound_cc_value_4             97.236477
engineTransmission_engineOil_cc_value_5               97.685027
engineTransmission_battery_cc_value_1                 98.365454
engineTransmission_engineOilLevelDipstick_cc_value_0  98.437678
engineTransmission_comments_value_2                   98.517505
engineTransmission_engine_cc_value_4                  98.874824
```

```
engineTransmission_coolant_cc_value_2                99.106702
engineTransmission_engineSound_cc_value_5            99.300566
engineTransmission_clutch_cc_value_3                 99.395598
engineTransmission_engineOil_cc_value_6              99.540046
engineTransmission_engine_cc_value_5                 99.650283
engineTransmission_battery_cc_value_2                99.726309
engineTransmission_gearShifting_cc_value_2           99.749116
engineTransmission_comments_value_3                  99.775725
engineTransmission_clutch_cc_value_4                 99.836545
engineTransmission_engine_cc_value_6                 99.859353
engineTransmission_battery_cc_value_3                99.939180
engineTransmission_comments_value_4                  99.946782
engineTransmission_engineOil_cc_value_7              99.958186
engineTransmission_engine_cc_value_7                 99.969590
engineTransmission_coolant_cc_value_3                99.969590
engineTransmission_clutch_cc_value_5                 99.973391
engineTransmission_engine_cc_value_8                 99.984795
engineTransmission_battery_cc_value_4                99.984795
engineTransmission_engine_cc_value_9                 99.988596
engineTransmission_engineOil_cc_value_8              99.992397
engineTransmission_clutch_cc_value_6                 99.996199
engineTransmission_engineOil_cc_value_9             100.000000
engineTransmission_engine_cc_value_10               100.000000
dtype: float64
```

In [5]:
```
df.shape
```

Out[5]:
```
(26307, 73)
```

In [6]:
```
col=[var for var in df.columns if df[var].isnull().mean()>.45]
col
```

Out[6]:
```
['engineTransmission_battery_cc_value_0',
 'engineTransmission_battery_cc_value_1',
 'engineTransmission_battery_cc_value_2',
 'engineTransmission_battery_cc_value_3',
 'engineTransmission_battery_cc_value_4',
 'engineTransmission_engineOilLevelDipstick_cc_value_0',
 'engineTransmission_engineOil_cc_value_1',
 'engineTransmission_engineOil_cc_value_2',
 'engineTransmission_engineOil_cc_value_3',
 'engineTransmission_engineOil_cc_value_4',
 'engineTransmission_engineOil_cc_value_5',
 'engineTransmission_engineOil_cc_value_6',
 'engineTransmission_engineOil_cc_value_7',
 'engineTransmission_engineOil_cc_value_8',
 'engineTransmission_engineOil_cc_value_9',
 'engineTransmission_engine_cc_value_0',
 'engineTransmission_engine_cc_value_1',
 'engineTransmission_engine_cc_value_2',
 'engineTransmission_engine_cc_value_3',
 'engineTransmission_engine_cc_value_4',
 'engineTransmission_engine_cc_value_5',
 'engineTransmission_engine_cc_value_6',
 'engineTransmission_engine_cc_value_7',
 'engineTransmission_engine_cc_value_8',
 'engineTransmission_engine_cc_value_9',
 'engineTransmission_engine_cc_value_10',
 'engineTransmission_coolant_cc_value_0',
 'engineTransmission_coolant_cc_value_1',
 'engineTransmission_coolant_cc_value_2',
 'engineTransmission_coolant_cc_value_3',
 'engineTransmission_engineMounting_cc_value_0',
 'engineTransmission_engineSound_cc_value_1',
 'engineTransmission_engineSound_cc_value_2',
 'engineTransmission_engineSound_cc_value_3',
 'engineTransmission_engineSound_cc_value_4',
 'engineTransmission_engineSound_cc_value_5',
 'engineTransmission_exhaustSmoke_cc_value_0',
 'engineTransmission_clutch_cc_value_0',
 'engineTransmission_clutch_cc_value_1',
 'engineTransmission_clutch_cc_value_2',
 'engineTransmission_clutch_cc_value_3',
 'engineTransmission_clutch_cc_value_4',
 'engineTransmission_clutch_cc_value_5',
 'engineTransmission_clutch_cc_value_6',
 'engineTransmission_gearShifting_cc_value_0',
 'engineTransmission_gearShifting_cc_value_1',
 'engineTransmission_gearShifting_cc_value_2',
 'engineTransmission_comments_value_0',
 'engineTransmission_comments_value_1',
```

```
 'engineTransmission_comments_value_2',
 'engineTransmission_comments_value_3',
 'engineTransmission_comments_value_4']
```

In [7]:
```
df[col].head()
```

Out[7]:

	engineTransmission_battery_cc_value_0	engineTransmission_battery_cc_value_1	engineTransmission_battery_cc_value_2	engineTransmission_battery_cc_value_3	engineTransmission_battery_c
0	Weak	NaN	NaN	NaN	
1	NaN	NaN	NaN	NaN	
2	NaN	NaN	NaN	NaN	
3	NaN	NaN	NaN	NaN	
4	NaN	NaN	NaN	NaN	

In [8]:
```
df[col].shape
```

Out[8]:
```
(26307, 52)
```

In [9]:
```
df_n=df.drop(col,axis=1)
```

In [10]:
```
df_n.head(10)
```

Out[10]:

	appointmentId	inspectionStartTime	year	month	engineTransmission_battery_value	engineTransmission_engineoilLevelDipstick_value	engineTransmission_engineOil	engineTransmission_eng
0	aj_01	2/3/19 15:43	2008	8	No	Yes	No	
1	aj_02	1/16/19 13:02	2007	5	Yes	Yes	No	
2	aj_03	2/9/19 13:31	2012	5	Yes	Yes	No	
3	aj_04	1/18/19 11:02	2013	1	Yes	Yes	No	
4	aj_05	1/27/19 12:12	2011	7	Yes	Yes	No	
5	aj_06	1/31/19 11:53	2011	11	Yes	Yes	Yes	
6	aj_07	4/4/19 17:00	2012	3	Yes	Yes	No	
7	aj_08	4/8/19 12:47	2016	2	No	Yes	Yes	
8	aj_09	1/5/19 16:09	2007	4	Yes	Yes	No	
9	aj_10	1/6/19 13:38	2011	2	Yes	Yes	No	

In [11]:
```
df_n.isnull().sum()*100/len(df_n)
```

Out[11]:
```
appointmentId                                    0.000000
inspectionStartTime                              0.000000
year                                             0.000000
month                                            0.000000
engineTransmission_battery_value                 0.000000
engineTransmission_engineoilLevelDipstick_value  0.000000
engineTransmission_engineOil                     0.000000
engineTransmission_engineOil_cc_value_0         29.459840
engineTransmission_engine_value                  0.000000
engineTransmission_coolant_value                 0.000000
engineTransmission_engineMounting_value          0.000000
engineTransmission_engineSound_value             0.000000
engineTransmission_engineSound_cc_value_0       27.471776
engineTransmission_exhaustSmoke_value            0.000000
engineTransmission_engineBlowByBackCompression_value     0.000000
engineTransmission_engineBlowByBackCompression_cc_value_0    0.000000
engineTransmission_clutch_value                  0.000000
engineTransmission_gearShifting_value            0.000000
fuel_type                                        0.000000
odometer_reading                                 0.000000
rating_engineTransmission                        0.000000
dtype: float64
```

In [12]:
```
df_n[['engineTransmission_engineOil_cc_value_0','engineTransmission_engineSound_cc_value_0']]
```

Out[12]:

	engineTransmission_engineOil_cc_value_0	engineTransmission_engineSound_cc_value_0
0	Leaking	Alternator Brg Noise
1	Leaking	Timing Noise
2	Dirty	Alternator Brg Noise
3	Dirty	NaN
4	Leaking	Timing Noise
...	...	...
26302	Leaking	Alternator Brg Noise
26303	Dirty	Alternator Brg Noise
26304	Leaking	Timing Noise
26305	Dirty	NaN
26306	Dirty	Tappet Noise

26307 rows × 2 columns

In [13]:
```
df_n['engineTransmission_engineOil_cc_value_0'].value_counts()
```

Out[13]:
```
Leaking                             7892
Dirty                               6906
Level Low                           2739
Leakage from Tappet Cover            716
Leakage from Side cover              221
Leakage from Sump/chamber             50
Leakage from Turbo Charger            31
Low Pressure warning light glowing     1
Mixed with Coolant                     1
Name: engineTransmission_engineOil_cc_value_0, dtype: int64
```

In [14]:
```
#df_n['engineTransmission_coolant_cc_value_0'].value_counts()
```

In [15]:
```
#df_n['engineTransmission_engineMounting_cc_value_0'].value_counts()
```

In [16]:
```
df_n['engineTransmission_engineSound_cc_value_0'].value_counts()
```

Out[16]:
```
Timing Noise             8170
Tappet Noise             3934
Alternator Brg Noise     3700
Whistling Noise-Turbo    1685
Engine Auxiliary Noise    936
Water Pump Brg Noise      408
Injector Noise            247
Name: engineTransmission_engineSound_cc_value_0, dtype: int64
```

In [17]:
```
#df_n['engineTransmission_engineSound_cc_value_1'].value_counts()
```

In [18]:
```
#df_n['engineTransmission_clutch_cc_value_0'].value_counts()
```

In [19]:
```
cm=[ab for ab in df_n.columns if df_n[ab].isnull().sum()>0]
cm
```

Out[19]:
```
['engineTransmission_engineOil_cc_value_0',
 'engineTransmission_engineSound_cc_value_0']
```

In [20]:
```
df_n[cm].head(10)
```

Out[20]:

	engineTransmission_engineOil_cc_value_0	engineTransmission_engineSound_cc_value_0
0	Leaking	Alternator Brg Noise
1	Leaking	Timing Noise
2	Dirty	Alternator Brg Noise
3	Dirty	NaN
4	Leaking	Timing Noise
5	NaN	Tappet Noise
6	Dirty	Timing Noise
7	NaN	Timing Noise
8	Level Low	Alternator Brg Noise
9	Dirty	Timing Noise

In [21]:
```
df_n[cm]=df_n[cm].fillna('missing')
```

In [22]:
```
df_n['engineTransmission_engineOil_cc_value_0'].fillna('missing',inplace=True)
```

In [23]:
```
df_n.isnull().sum()
```

Out[23]:
```
appointmentId                                        0
inspectionStartTime                                  0
year                                                 0
month                                                0
engineTransmission_battery_value                     0
engineTransmission_engineoilLevelDipstick_value      0
engineTransmission_engineOil                         0
engineTransmission_engineOil_cc_value_0              0
engineTransmission_engine_value                      0
engineTransmission_coolant_value                     0
engineTransmission_engineMounting_value              0
engineTransmission_engineSound_value                 0
engineTransmission_engineSound_cc_value_0            0
engineTransmission_exhaustSmoke_value                0
engineTransmission_engineBlowByBackCompression_value 0
engineTransmission_engineBlowByBackCompression_cc_value_0    0
engineTransmission_clutch_value                      0
engineTransmission_gearShifting_value                0
fuel_type                                            0
odometer_reading                                     0
rating_engineTransmission                            0
dtype: int64
```

In [24]:
```
df_n.shape
```

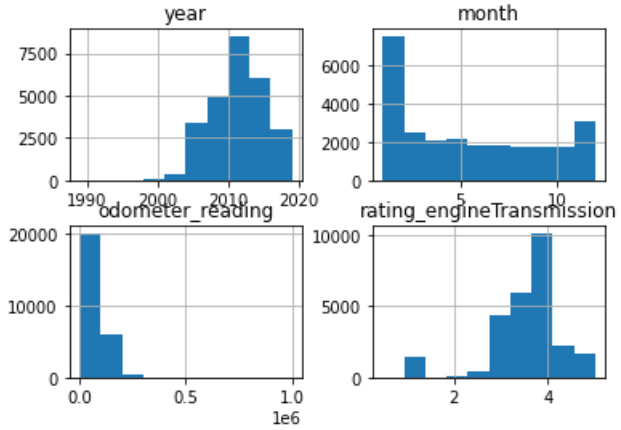
Out[24]:
```
(26307, 21)
```

In [25]:
```
df_n.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26307 entries, 0 to 26306
Data columns (total 21 columns):
 #   Column                                                Non-Null Count  Dtype  
---  ------                                                --------------  -----  
 0   appointmentId                                         26307 non-null  object 
 1   inspectionStartTime                                   26307 non-null  object 
 2   year                                                  26307 non-null  int64  
 3   month                                                 26307 non-null  int64  
 4   engineTransmission_battery_value                      26307 non-null  object 
 5   engineTransmission_engineoilLevelDipstick_value       26307 non-null  object 
 6   engineTransmission_engineOil                          26307 non-null  object 
 7   engineTransmission_engineOil_cc_value_0               26307 non-null  object 
 8   engineTransmission_engine_value                       26307 non-null  object 
 9   engineTransmission_coolant_value                      26307 non-null  object 
 10  engineTransmission_engineMounting_value               26307 non-null  object 
 11  engineTransmission_engineSound_value                  26307 non-null  object 
 12  engineTransmission_engineSound_cc_value_0             26307 non-null  object 
 13  engineTransmission_exhaustSmoke_value                 26307 non-null  object 
 14  engineTransmission_engineBlowByBackCompression_value  26307 non-null  object 
 15  engineTransmission_engineBlowByBackCompression_cc_value_0  26307 non-null  object 
 16  engineTransmission_clutch_value                       26307 non-null  object 
 17  engineTransmission_gearShifting_value                 26307 non-null  object 
 18  fuel_type                                             26307 non-null  object 
 19  odometer_reading                                      26307 non-null  int64  
 20  rating_engineTransmission                             26307 non-null  float64
dtypes: float64(1), int64(3), object(17)
memory usage: 4.2+ MB
```

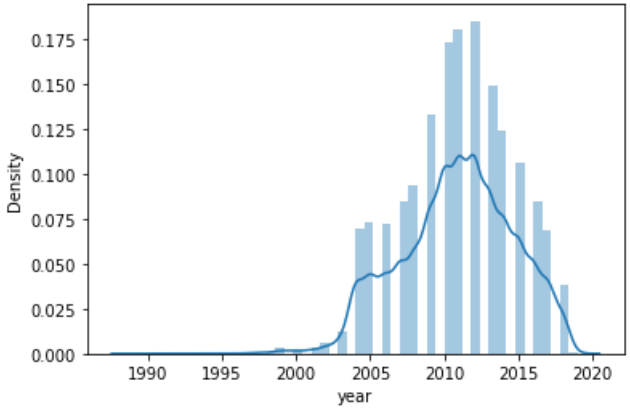
In [26]:
```
df_n.hist()
plt.show
```

Out[26]: <function matplotlib.pyplot.show(close=None, block=None)>



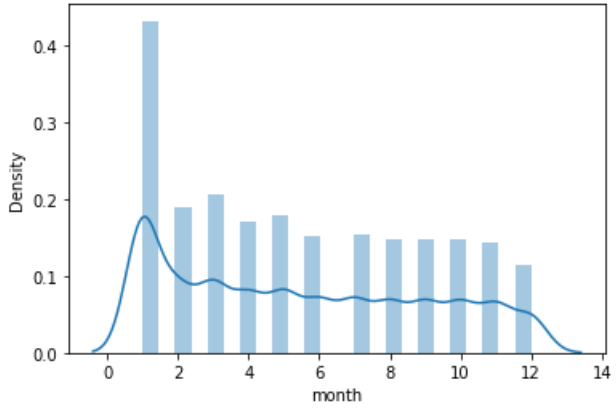
In [27]:
```
sns.distplot(df_n['year'])
```

Out[27]: <AxesSubplot:xlabel='year', ylabel='Density'>



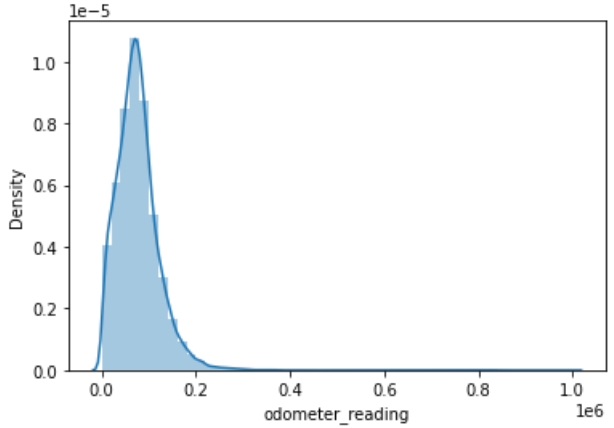
In [28]:
```
sns.distplot(df_n['month'])
```

Out[28]:
```
<AxesSubplot:xlabel='month', ylabel='Density'>
```



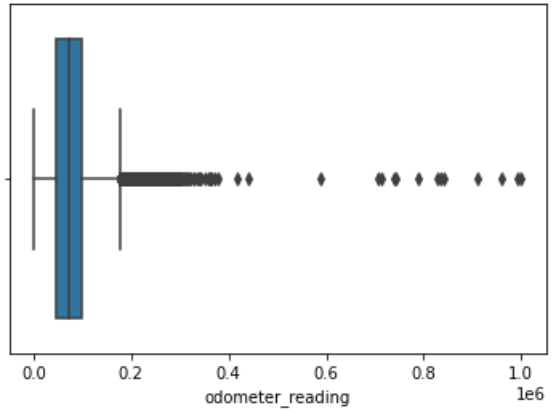
In [29]:
```
sns.distplot(df_n['odometer_reading'])
```

Out[29]:
```
<AxesSubplot:xlabel='odometer_reading', ylabel='Density'>
```



In [30]:
```
sns.boxplot(df_n['odometer_reading'])
```

Out[30]: <AxesSubplot:xlabel='odometer_reading'>



In [31]:
```
df_n['odometer_reading'].describe()
```

Out[31]:
```
count     26307.000000
mean      76460.143764
std       46762.524489
min           1.000000
25%       46396.000000
50%       72013.000000
75%       98289.500000
max      999999.000000
Name: odometer_reading, dtype: float64
```

In [ ]:
```

```

In [32]:
```
percentile_25=df_n['odometer_reading'].quantile(0.25)
percentile_75=df_n['odometer_reading'].quantile(0.75)
```

In [33]:
```
iqr=percentile_75-percentile_25
iqr
```

Out[33]: 51893.5

In [34]:
```
upper_limit=percentile_75+1.5*iqr
lower_limit=percentile_25-1.5*iqr
```

In [35]:
```
df_n[df_n['odometer_reading']>upper_limit]
```

Out[35]:

	appointmentId	inspectionStartTime	year	month	engineTransmission_battery_value	engineTransmission_engineoilLevelDipstick_value	engineTransmission_engineOil	engineTransmission_
57	aj_58	1/17/19 13:21	2007	12	Yes	Yes	No	
203	aj_204	2/17/19 15:58	2006	3	Yes	Yes	No	
237	aj_238	3/9/19 16:22	2009	12	Yes	Yes	No	
322	aj_323	1/24/19 10:07	2004	1	Yes	Yes	Yes	
337	aj_338	4/8/19 15:30	2009	11	Yes	Yes	No	
...	...	...	...	...	...	...	...	
26143	aj_26144	3/24/19 17:40	2013	11	Yes	Yes	No	
26145	aj_26146	2/17/19 15:53	2010	1	Yes	Yes	No	
26261	aj_26262	1/5/19 16:02	2006	8	No	Yes	No	
26274	aj_26275	1/14/19 16:26	2009	9	Yes	Yes	Yes	
26279	aj_26280	4/7/19 11:43	2015	4	No	Yes	No	

718 rows × 21 columns

In [36]:
```
new_df_n=df_n[df_n['odometer_reading']<=upper_limit]
```

In [37]:
```
new_df_n
```

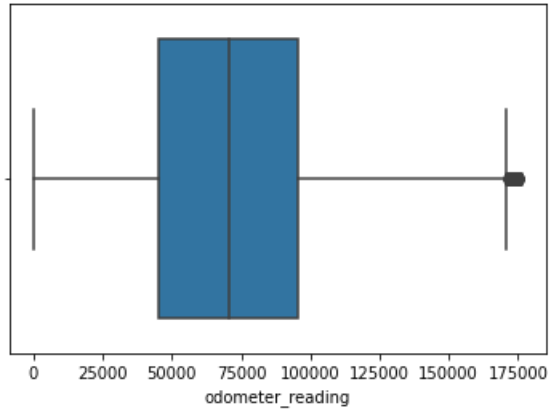
Out[37]:

	appointmentId	inspectionStartTime	year	month	engineTransmission_battery_value	engineTransmission_engineoilLevelDipstick_value	engineTransmission_engineOil	engineTransmission_
0	aj_01	2/3/19 15:43	2008	8	No	Yes	No	
1	aj_02	1/16/19 13:02	2007	5	Yes	Yes	No	
2	aj_03	2/9/19 13:31	2012	5	Yes	Yes	No	
3	aj_04	1/18/19 11:02	2013	1	Yes	Yes	No	
4	aj_05	1/27/19 12:12	2011	7	Yes	Yes	No	
...	...	...	...	...	...	...	...	
26302	aj_26303	3/10/19 13:08	2013	3	Yes	Yes	No	
26303	aj_26304	4/12/19 13:59	2007	8	No	No	No	
26304	aj_26305	2/28/19 10:42	2004	7	Yes	Yes	No	
26305	aj_26306	4/2/19 12:21	2010	12	Yes	Yes	No	
26306	aj_26307	4/6/19 13:09	2015	11	Yes	Yes	No	

25589 rows × 21 columns

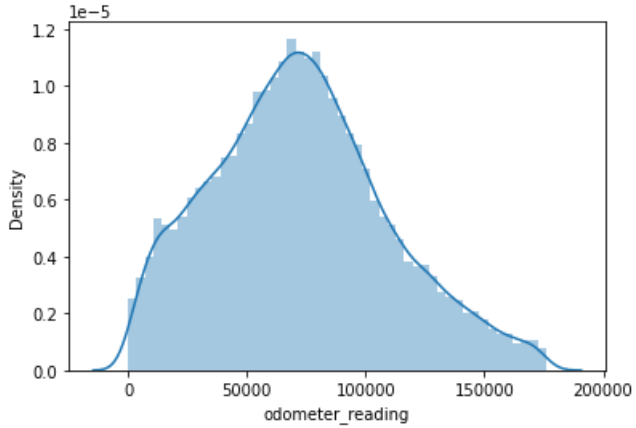
In [38]:
```
sns.boxplot(new_df_n['odometer_reading'])
```

Out[38]:
```
<AxesSubplot:xlabel='odometer_reading'>
```



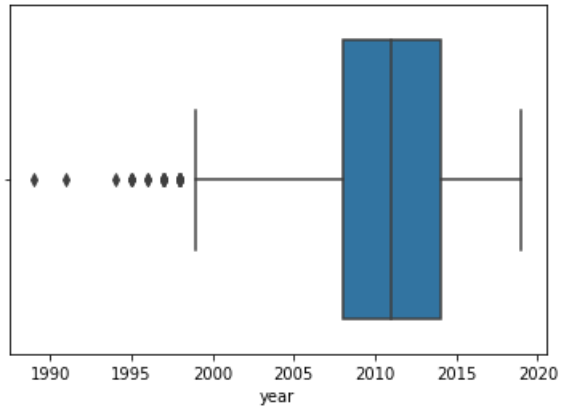
In [39]:
```
sns.distplot(new_df_n['odometer_reading'])
```

Out[39]: <AxesSubplot:xlabel='odometer_reading', ylabel='Density'>



In [40]:
```
sns.boxplot(df_n['year'])
```

Out[40]: <AxesSubplot:xlabel='year'>



In [41]:
```
def outlier(data,a):
    percentile_25=data[a].quantile(0.25)
    percentile_75=data[a].quantile(0.75)
    iqr=percentile_75-percentile_25
    upper_limit=percentile_75+1.5*iqr
    lower_limit=percentile_25-1.5*iqr
    return data[(data[a] < upper_limit) & (data[a] > lower_limit)]
```

In [42]:
```
new_df_n =outlier(new_df_n,'year')
```

In [43]:
```
new_df_n.shape
```

Out[43]:
```
(25427, 21)
```

In [44]:
```
new_df_n
```

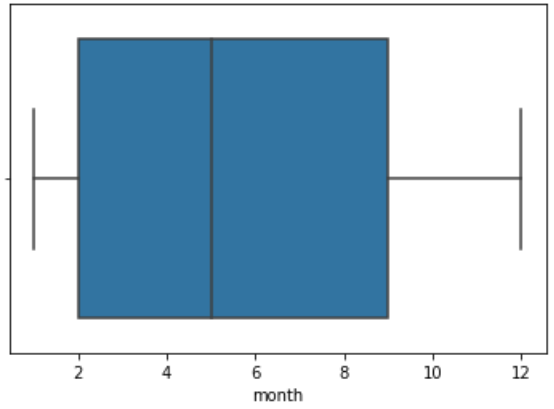
Out[44]:

	appointmentId	inspectionStartTime	year	month	engineTransmission_battery_value	engineTransmission_engineoilLevelDipstick_value	engineTransmission_engineOil	engineTransmission_
0	aj_01	2/3/19 15:43	2008	8	No	Yes	No	
1	aj_02	1/16/19 13:02	2007	5	Yes	Yes	No	
2	aj_03	2/9/19 13:31	2012	5	Yes	Yes	No	
3	aj_04	1/18/19 11:02	2013	1	Yes	Yes	No	
4	aj_05	1/27/19 12:12	2011	7	Yes	Yes	No	
...	...	...	...	...	...	...	...	
26302	aj_26303	3/10/19 13:08	2013	3	Yes	Yes	No	
26303	aj_26304	4/12/19 13:59	2007	8	No	No	No	
26304	aj_26305	2/28/19 10:42	2004	7	Yes	Yes	No	
26305	aj_26306	4/2/19 12:21	2010	12	Yes	Yes	No	
26306	aj_26307	4/6/19 13:09	2015	11	Yes	Yes	No	

25427 rows × 21 columns

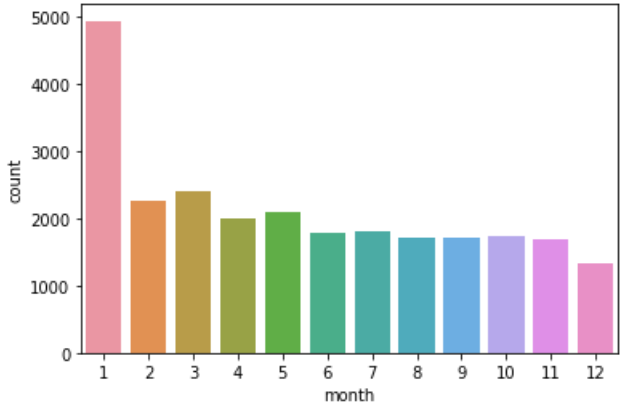
In [45]:
```
sns.boxplot(df_n['month'])
```

Out[45]:
```
<AxesSubplot:xlabel='month'>
```



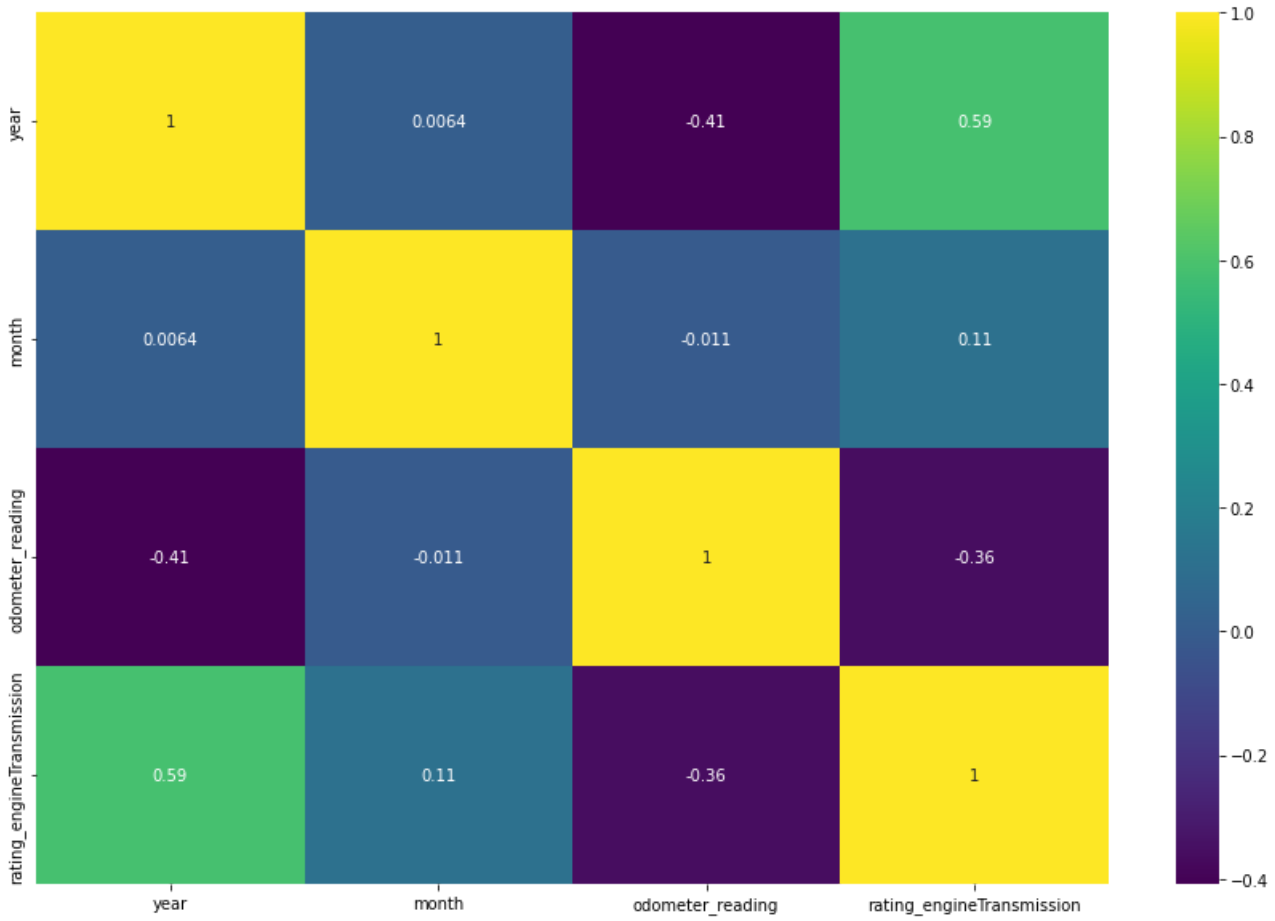
In [46]:
```
sns.countplot(new_df_n['month'])
```

Out[46]:
```
<AxesSubplot:xlabel='month', ylabel='count'>
```



In [47]:
```
_,ax=plt.subplots(figsize=(15,10))
colormap=sns.color_palette("viridis", as_cmap=True)
sns.heatmap(df_n.corr(),annot=True,cmap=colormap)
```

Out[47]: <AxesSubplot:>



In [48]:
```
categorical_columns = new_df_n.select_dtypes('O').columns
```

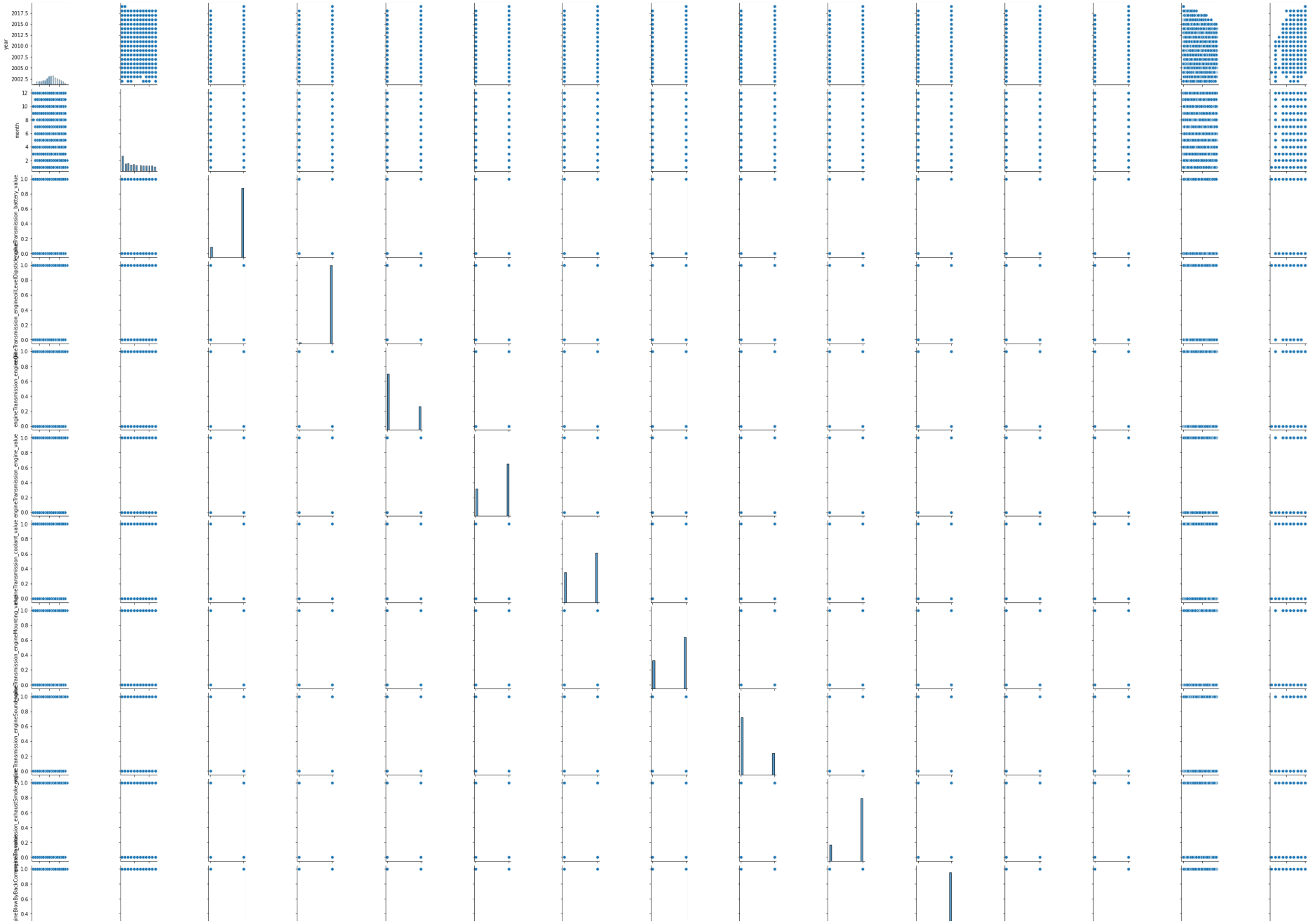
In [49]:
```
#fig = plt.figure(figsize=(50,50))
#for i, col in enumerate(categorical_columns):
    plt.subplot(9, 7, i+1)
    sns.countplot(new_df_n[col])
    plt.xticks(rotation=-45)
plt.show()
```

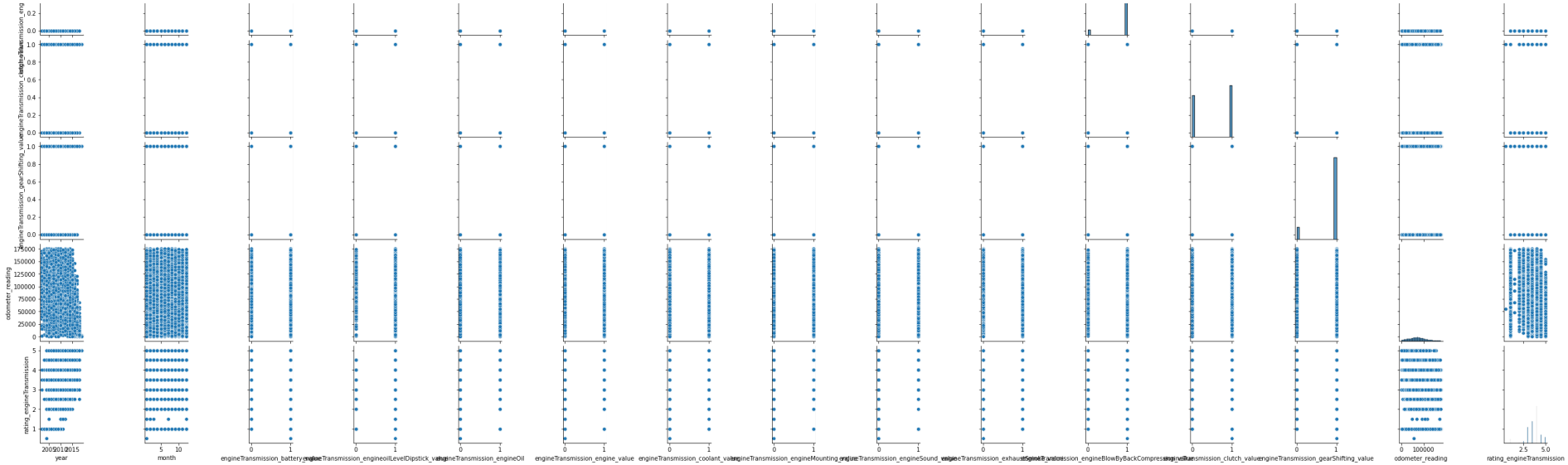
```
  Input In [49]
    plt.subplot(9, 7, i+1)
    ^
IndentationError: unexpected indent
```

In [ ]:
```
#after handling categorical data.
```

In [73]:
```
sns.pairplot(new_df_n)
```

Out[73]: <seaborn.axisgrid.PairGrid at 0x142a7433100>





In [50]:
```
new_df_n.head()
```

Out[50]:

	appointmentId	inspectionStartTime	year	month	engineTransmission_battery_value	engineTransmission_engineoilLevelDipstick_value	engineTransmission_engineOil	engineTransmission_eng
0	aj_01	2/3/19 15:43	2008	8	No	Yes	No	
1	aj_02	1/16/19 13:02	2007	5	Yes	Yes	No	
2	aj_03	2/9/19 13:31	2012	5	Yes	Yes	No	
3	aj_04	1/18/19 11:02	2013	1	Yes	Yes	No	
4	aj_05	1/27/19 12:12	2011	7	Yes	Yes	No	

In [51]:
```
new_df_n.corr()
```

Out[51]:

	year	month	odometer_reading	rating_engineTransmission
year	1.000000	-0.010301	-0.463181	0.558699
month	-0.010301	1.000000	-0.020373	0.102007
odometer_reading	-0.463181	-0.020373	1.000000	-0.393603
rating_engineTransmission	0.558699	0.102007	-0.393603	1.000000

In [52]:
```
new_df_n.drop('appointmentId',axis=1,inplace=True)
```

In [53]:
```
new_df_n.drop('inspectionStartTime',axis=1,inplace=True)
```

In [54]:
```
new_df_n.corr()
```

Out[54]:

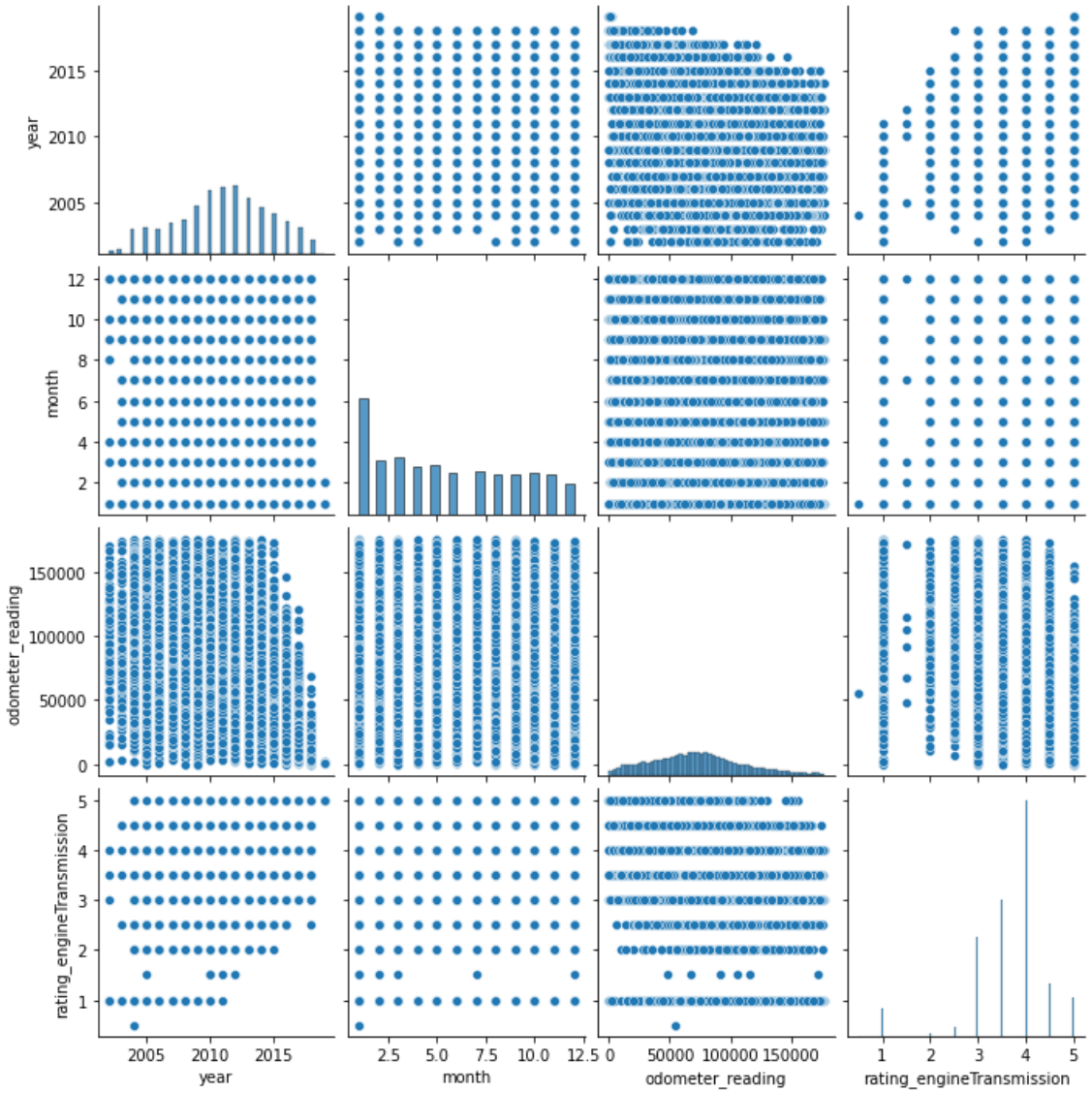
	year	month	odometer_reading	rating_engineTransmission
year	1.000000	-0.010301	-0.463181	0.558699
month	-0.010301	1.000000	-0.020373	0.102007
odometer_reading	-0.463181	-0.020373	1.000000	-0.393603
rating_engineTransmission	0.558699	0.102007	-0.393603	1.000000

In [55]:
```
new_df_n.shape
```

Out[55]:
```
(25427, 19)
```

In [56]:
```
sns.pairplot(new_df_n)
```

Out[56]: <seaborn.axisgrid.PairGrid at 0x142a8c10b50>



In [ ]:

In [57]:
```
new_df_n.replace(to_replace='Yes', value=1, inplace=True)
new_df_n.replace(to_replace='No', value=0, inplace=True)
```

In [58]:
```
cate = new_df_n.select_dtypes('O').columns
cate
```

Out[58]:
```
Index(['engineTransmission_engineOil_cc_value_0',
       'engineTransmission_engineSound_cc_value_0',
       'engineTransmission_engineBlowByBackCompression_cc_value_0',
       'fuel_type'],
      dtype='object')
```

In [59]:
```
new_df_n['engineTransmission_engineSound_cc_value_0'].value_counts()
```

Out[59]:
```
Timing Noise             7940
missing                  6928
Tappet Noise             3869
Alternator Brg Noise     3555
Whistling Noise-Turbo    1584
Engine Auxiliary Noise    919
Water Pump Brg Noise      390
Injector Noise            242
Name: engineTransmission_engineSound_cc_value_0, dtype: int64
```

In [60]:
```
new_df_n['fuel_type'].unique()
```

Out[60]:
```
array(['Petrol + CNG', 'Diesel', 'Petrol', 'Petrol + LPG', 'Hybrid',
       'Electric'], dtype=object)
```

In [61]:
```
y= new_df_n['rating_engineTransmission']
x=new_df_n.drop(['rating_engineTransmission'],axis=1)
```

In [62]:
```
from sklearn import preprocessing
from sklearn.preprocessing import LabelEncoder
le = preprocessing.LabelEncoder()
```

In [63]:
```
x['engineTransmission_engineOil_cc_value_0']=le.fit_transform(x['engineTransmission_engineOil_cc_value_0'])
#x['engineTransmission_engineOil_cc_value_1']=le.fit_transform(x['engineTransmission_engineOil_cc_value_1'])
#x['engineTransmission_engine_cc_value_0']=le.fit_transform(x['engineTransmission_engine_cc_value_0'])
#x['engineTransmission_coolant_cc_value_0']=le.fit_transform(x['engineTransmission_coolant_cc_value_0'])
#x['engineTransmission_engineMounting_cc_value_0']=le.fit_transform(x['engineTransmission_engineMounting_cc_value_0'])
x['engineTransmission_engineSound_cc_value_0']=le.fit_transform(x['engineTransmission_engineSound_cc_value_0'])
#x['engineTransmission_engineSound_cc_value_1']=le.fit_transform(x['engineTransmission_engineSound_cc_value_1'])
x['engineTransmission_engineBlowByBackCompression_cc_value_0']=le.fit_transform(x['engineTransmission_engineBlowByBackCompression_cc_value_0'])
#x['engineTransmission_clutch_cc_value_0']=le.fit_transform(x['engineTransmission_clutch_cc_value_0'])
x['fuel_type']=le.fit_transform(x['fuel_type'])
x.head()
```

Out[63]:

	year	month	engineTransmission_battery_value	engineTransmission_engineoilLevelDipstick_value	engineTransmission_engineOil	engineTransmission_engineOil_cc_value_0	engineTransmissio
0	2008	8	0	1	0	5	
1	2007	5	1	1	0	5	
2	2012	5	1	1	0	0	
3	2013	1	1	1	0	0	
4	2011	7	1	1	0	5	

In [64]:
```
y1=y.values
x1=x.values
```

In [65]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x1,y1,test_size=0.30, random_state=42)
```

In [ ]:
```
```

In [66]:
```
from sklearn.ensemble import GradientBoostingRegressor
gb=GradientBoostingRegressor()
gb.fit(x_train,y_train)
gb_train_score=gb.score(x_train,y_train)
gb_test_score=gb.score(x_test,y_test)
print(gb_train_score,gb_test_score)
```

```
0.6721882839529123 0.6655870610589179
```

In [ ]:
```
```

In [67]:
```
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor , ExtraTreesRegressor
```

In [68]:
```
knn = KNeighborsRegressor()
knn.fit(x_train, y_train)
knn.predict(x_test)

knn_train_score = knn.score(x_train, y_train)
knn_test_score = knn.score(x_test, y_test)
print(knn_train_score,knn_test_score)
```

```
0.3632451449308577 0.06257174577468361
```

In [69]:
```
dt = DecisionTreeRegressor()
dt.fit(x_train, y_train)
pred=dt.predict(x_test)

dt_train_score = dt.score(x_train, y_train)
dt_test_score = dt.score(x_test, y_test)

print(dt_train_score,dt_test_score)
```

```
1.0 0.3923714275784852
```

In [ ]:

In [70]:
```
rf = RandomForestRegressor()
rf.fit(x_train, y_train)

# Print the R squared scores
rf_train_score = rf.score(x_train, y_train)
rf_test_score = rf.score(x_test, y_test)
print(rf_train_score, rf_test_score)
```

```
0.9547657939096968 0.6813502925812862
```

In [71]:
```
ext = ExtraTreesRegressor()
ext.fit(x_train, y_train)

# Print the R squared scores
ext_train_score = ext.score(x_train, y_train)
ext_test_score = ext.score(x_test,y_test)
print(ext_train_score, ext_test_score)
```

```
1.0 0.6624657450626693
```

In [ ]: