

Московский государственный технический университет им. Н.Э. Баумана
Факультет «Информатика и системы управления»
Кафедра «Системы обработки информации и управления»



Лабораторная работа №1
по дисциплине «Методы машинного обучения»
«Создание истории о данных»

ИСПОЛНИТЕЛЬ:

Алехин С.С.
Группа ИУ5-23М

ПРОВЕРИЛ:

Балашов А.М.

Задание

- Создать «историю о данных» в виде юпитер-ноутбука, с учетом следующих требований:
- Создать "историю о данных" в виде юпитер-ноутбука, с учетом следующих требований:
- История должна содержать не менее 5 шагов (где 5 - рекомендуемое количество шагов). Каждый шаг содержит график и его текстовую интерпретацию.
- На каждом шаге наряду с удачным итоговым графиком рекомендуется в юпитер-ноутбуке оставлять результаты предварительных "неудачных" графиков.
- Не рекомендуется повторять виды графиков, желательно создать 5 графиков различных видов.
- Выбор графиков должен быть обоснован использованием методологии data-to-viz. Рекомендуется учитывать типичные ошибки построения выбранного вида графика по методологии data-to-viz. Если методология Вами отвергается, то просьба обосновать Ваше решение по выбору графика.
- История должна содержать итоговые выводы. В реальных "историях о данных" именно эти выводы представляют собой основную ценность для предприятия.
- Сформировать отчет и разместить его в своем репозитории на github.

Lab1

June 20, 2023

```
[ ]: import pandas as pd
import seaborn as sns
```

```
[ ]: # Enable inline plots
%matplotlib inline

# Set plot style
sns.set(style="ticks")

# Set plots formats to save high resolution PNG
from IPython.display import set_matplotlib_formats
set_matplotlib_formats("retina")
```

```
/var/folders/fs/5xh23h99763f_blp7m50x23h0000gq/T/ipykernel_3732/2802708398.py:9:
DeprecationWarning: `set_matplotlib_formats` is deprecated since IPython 7.23,
directly use `matplotlib_inline.backend_inline.set_matplotlib_formats()`
    set_matplotlib_formats("retina")
```

```
[ ]: pd.set_option("display.width", 70)
```

```
[ ]: data = pd.read_csv("cars.csv")
```

```
[ ]: data.dtypes
```

```
[ ]: Unnamed: 0      int64
price             int64
brand             object
model            object
year             int64
title_status      object
mileage          float64
color            object
vin              object
lot              int64
state            object
country          object
condition        object
dtype: object
```

```
[ ]: data.head()
```

```
[ ]:      Unnamed: 0  price      brand  model  year  title_status \
0          0      6300    toyota  cruiser  2008  clean vehicle
1          1      2899      ford      se  2011  clean vehicle
2          2      5350    dodge    mpv  2018  clean vehicle
3          3     25000      ford    door  2014  clean vehicle
4          4     27700  chevrolet    1500  2018  clean vehicle

      mileage  color      vin      lot      state \
0  274117.0  black  jtezu11f88k007763  159348797  new jersey
1  190552.0  silver  2fmdk3gc4bbb02217  166951262  tennessee
2   39590.0  silver  3c4pdcgg5jt346413  167655728   georgia
3   64146.0   blue  1ftfw1et4efc23745  167753855  virginia
4    6654.0   red   3gcpcrec2jg473991  167763266   florida

      country      condition
0      usa  10 days left
1      usa   6 days left
2      usa   2 days left
3      usa 22 hours left
4      usa 22 hours left
```

```
[ ]: data.shape
```

```
[ ]: (2499, 13)
```

```
[ ]: data.describe()
```

```
[ ]:      Unnamed: 0      price      year      mileage \
count  2499.000000  2499.000000  2499.000000  2.499000e+03
mean    1249.000000  18767.671469  2016.714286  5.229869e+04
std      721.543484  12116.094936    3.442656  5.970552e+04
min         0.000000    0.000000  1973.000000  0.000000e+00
25%       624.500000  10200.000000  2016.000000  2.146650e+04
50%      1249.000000  16900.000000  2018.000000  3.536500e+04
75%      1873.500000  25555.500000  2019.000000  6.347250e+04
max      2498.000000  84900.000000  2020.000000  1.017936e+06

      lot
count  2.499000e+03
mean    1.676914e+08
std     2.038772e+05
min     1.593488e+08
25%     1.676253e+08
50%     1.677451e+08
75%     1.677798e+08
```

```
max    1.678055e+08
```

1

```
[ ]: sns.distplot(data["price"]);
```

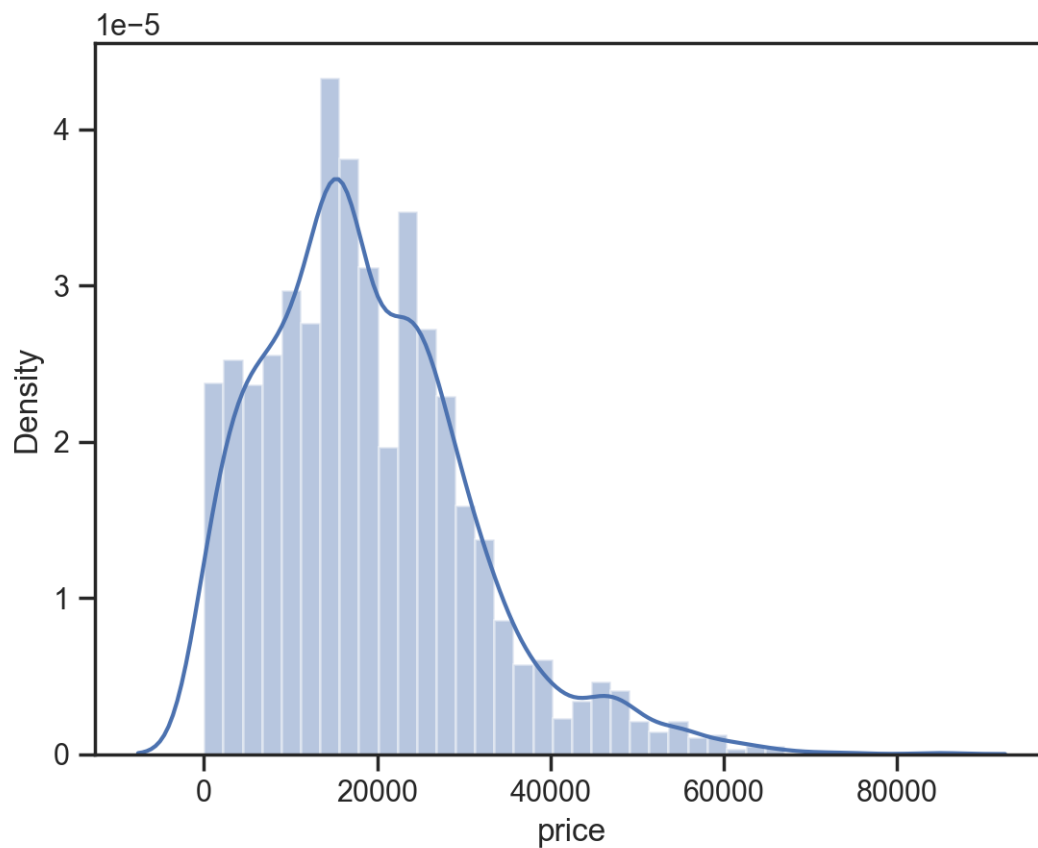
```
/var/folders/fs/5xh23h99763f_blp7m50x23h0000gq/T/ipykernel_3732/2055821964.py:1:  
UserWarning:
```

```
`distplot` is a deprecated function and will be removed in seaborn v0.14.0.
```

Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).

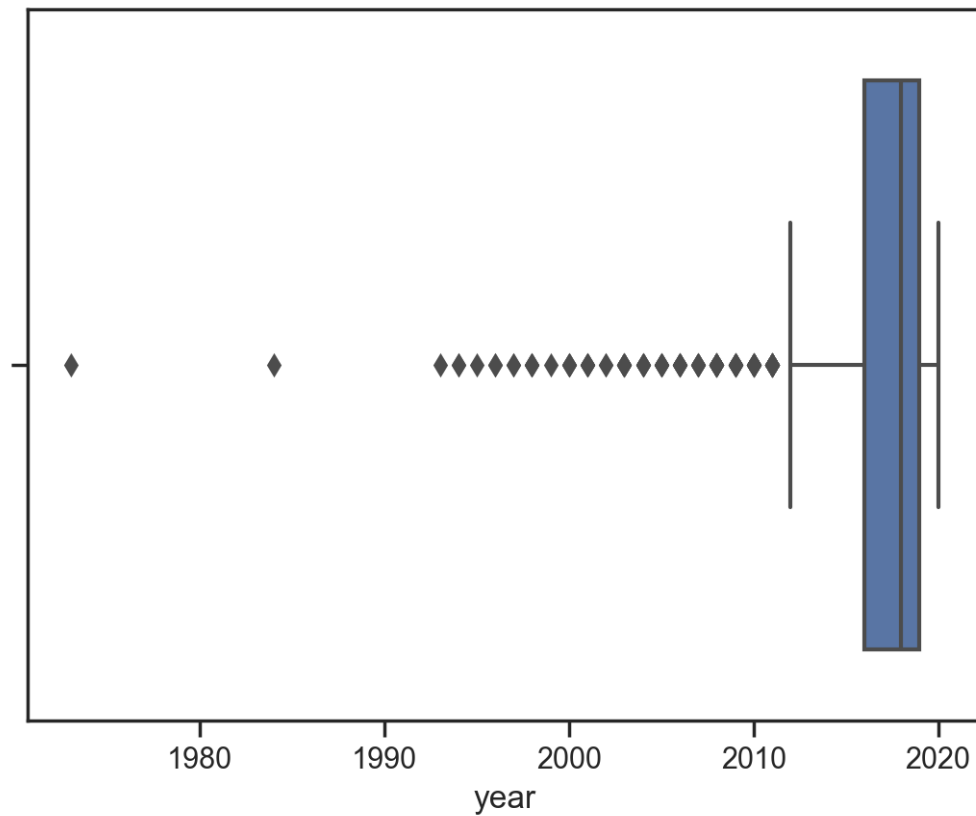
For a guide to updating your code to use the new functions, please see <https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751>

```
sns.distplot(data["price"]);
```



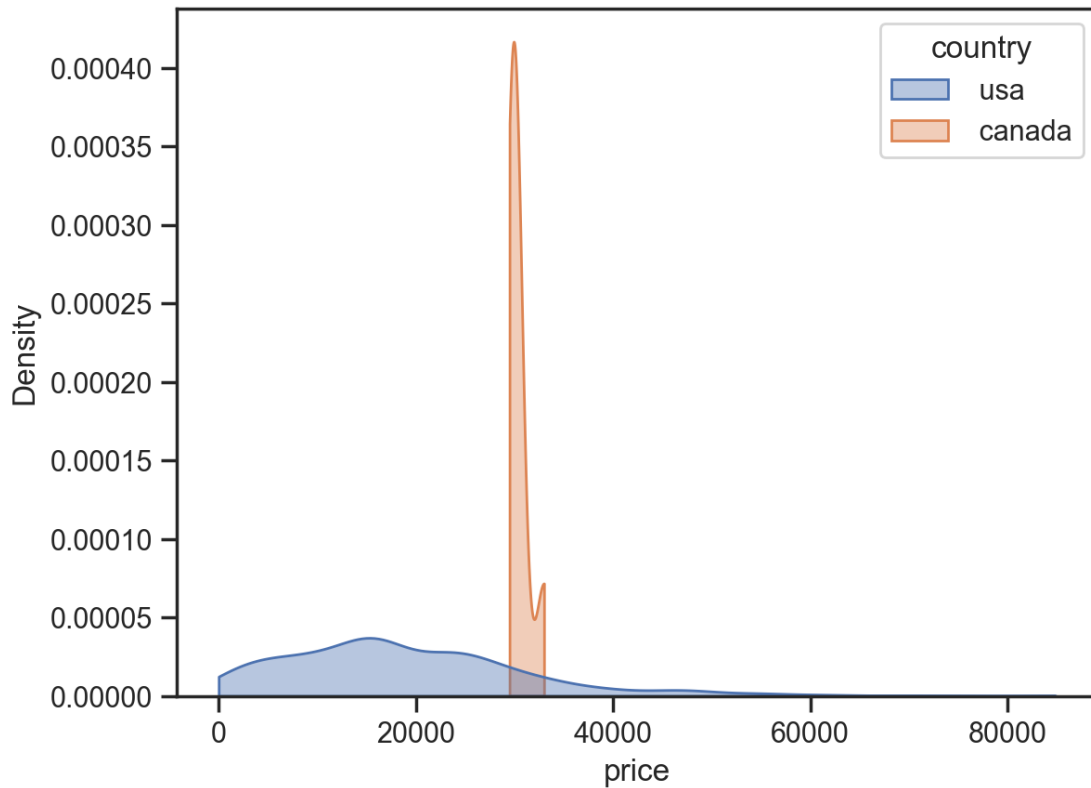
```
[ ]: sns.boxplot(x=data['year'])
```

```
[ ]: <Axes: xlabel='year'>
```



```
[ ]: sns.kdeplot(data=data, x="price", hue="country", cut=0, fill=True,
    ↪common_norm=False, alpha=0.4)
```

```
[ ]: <Axes: xlabel='price', ylabel='Density'>
```



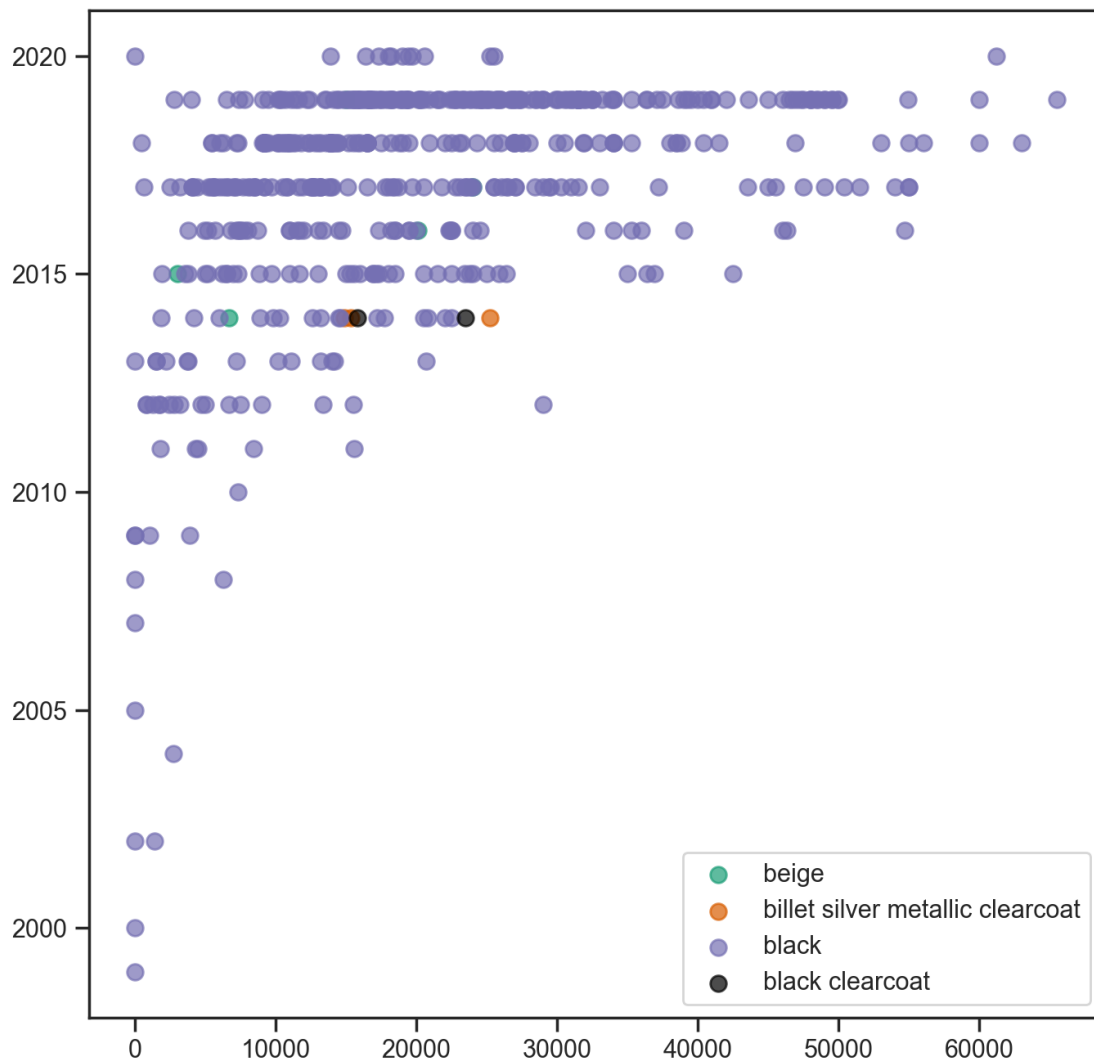
```
[ ]: import numpy as np
import matplotlib.pyplot as plt
FLIPPER_LENGTH = data["price"].values
BILL_LENGTH = data["year"].values

SPECIES = data["color"].values
SPECIES_ = np.unique(SPECIES)

COLORS = ["#1B9E77", "#D95F02", "#7570B3", "#000000"]

fig, ax = plt.subplots(figsize=(8,8))
for species, color in zip(SPECIES_, COLORS):
    idxs = np.where(SPECIES == species)
    # No legend will be generated if we don't pass label=species
    ax.scatter(
        FLIPPER_LENGTH[idxs], BILL_LENGTH[idxs], label=species,
        s=50, color=color, alpha=0.7
    )

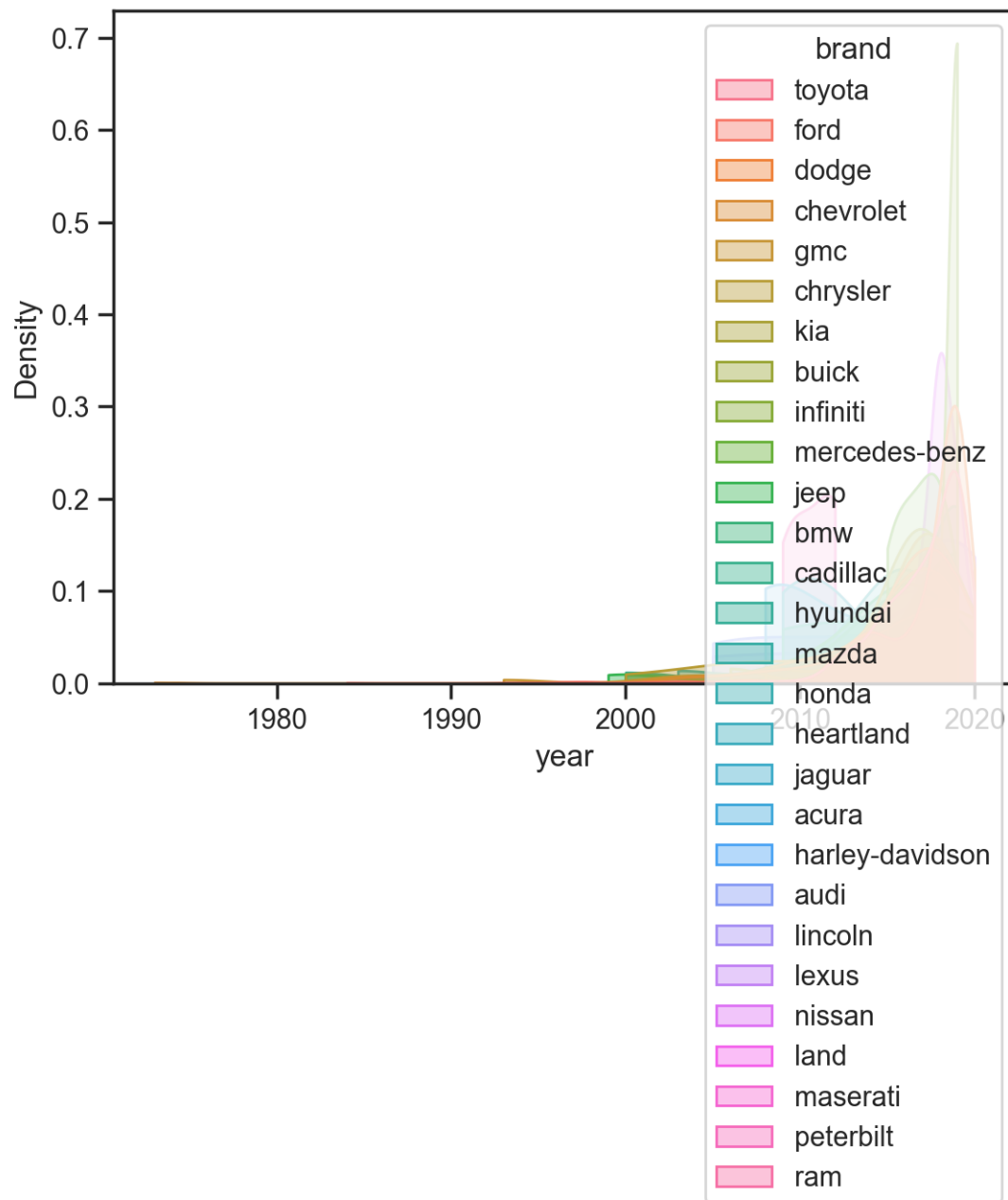
ax.legend();
```



```
[ ]: sns.kdeplot(data=data, x="year", hue="brand", cut=0, fill=True,
↳common_norm=False, alpha=0.4)
```

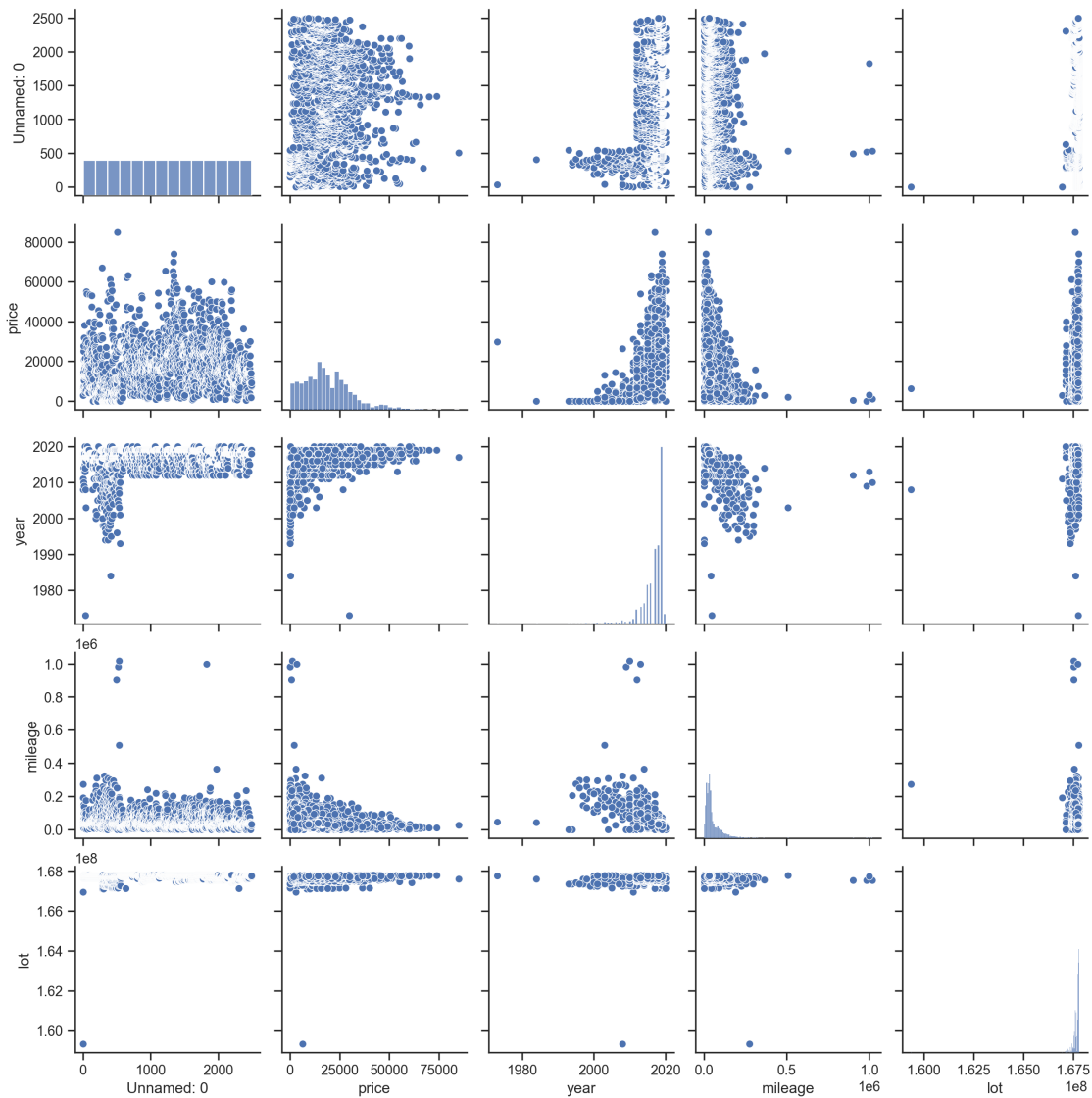
```
/var/folders/fs/5xh23h99763f_blp7m50x23h0000gq/T/ipykernel_3732/32137796.py:1:
UserWarning: Dataset has 0 variance; skipping density estimate. Pass
`warn_singular=False` to disable this warning.
  sns.kdeplot(data=data, x="year", hue="brand", cut=0, fill=True,
common_norm=False, alpha=0.4)
```

```
[ ]: <Axes: xlabel='year', ylabel='Density'>
```

```
[ ]: sns.pairplot(data)
```

```
[ ]: <seaborn.axisgrid.PairGrid at 0x284e9f090>
```



2

```
[ ]: corr_matrix = data.corr()
```

ValueError

Traceback (most recent call last)

Cell In[221], line 1

```
----> 1 corr_matrix = data.corr()
```

File ~/BMSTU_Labs/.env/lib/python3.11/site-packages/pandas/core/frame.py:10054,
in DataFrame.corr(self, method, min_periods, numeric_only)
10052 cols = data.columns

```

10053 idx = cols.copy()
> 10054 mat = data.to_numpy(dtype=float, na_value=np.nan, copy=False)
10056 if method == "pearson":
10057     correl = libalgos.nancorr(mat, minp=min_periods)

File ~/BMSTU_Labs/.env/lib/python3.11/site-packages/pandas/core/frame.py:1838,
↳ in DataFrame.to_numpy(self, dtype, copy, na_value)
    1836 if dtype is not None:
    1837     dtype = np.dtype(dtype)
-> 1838 result = self._mgr.as_array(dtype=dtype, copy=copy, na_value=na_value)
    1839 if result.dtype is not dtype:
    1840     result = np.array(result, dtype=dtype, copy=False)

File ~/BMSTU_Labs/.env/lib/python3.11/site-packages/pandas/core/internals/
↳ managers.py:1732, in BlockManager.as_array(self, dtype, copy, na_value)
    1730     arr.flags.writeable = False
    1731 else:
-> 1732     arr = self._interleave(dtype=dtype, na_value=na_value)
    1733     # The underlying data was copied within _interleave, so no need
    1734     # to further copy if copy=True or setting na_value
    1736 if na_value is not lib.no_default:

File ~/BMSTU_Labs/.env/lib/python3.11/site-packages/pandas/core/internals/
↳ managers.py:1794, in BlockManager._interleave(self, dtype, na_value)
    1792     else:
    1793         arr = blk.get_values(dtype)
-> 1794     result[rl.indexer] = arr
    1795     itemmask[rl.indexer] = 1
    1797 if not itemmask.all():

ValueError: could not convert string to float: 'toyota'

```

```
[ ]: sns.heatmap(data.corr(), annot=True, fmt='.3f')
```

```
[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7f2cf318d040>
```

