- Saving and loading notebooks in GitHub
- Interactive forms
- Interactive widgets

Working with data

- Loading data: Drive, Sheets and Google Cloud Storage
- Charts: visualising data
- Getting started with BigQuery

Machine learning crash course

These are a few of the notebooks from Google's online machine learning course. See the full course website for more.

- Intro to Pandas DataFrame
- Linear regression with tf.keras using synthetic data

Using accelerated hardware

- TensorFlow with GPUs
- TensorFlow with TPUs

Featured examples

- NeMo voice swap: Use Nvidia NeMo conversational AI toolkit to swap a voice in an audio fragment with a computer-generated one.
- Retraining an Image Classifier: Build a Keras model on top of a pre-trained image classifier to distinguish flowers.
- Text Classification: Classify IMDB film reviews as either *positive* or *negative*.
- Style Transfer: Use deep learning to transfer style between images.
- Multilingual Universal Sentence Encoder Q&A: Use a machine-learning model to answer questions from the SQuAD dataset.
- Video Interpolation: Predict what happened in a video between the first and the last frame.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
df=pd.read_csv("Automobile_data.csv")
df.head()
```

	symboling	normalized-losses	make	fuel-type	body-style	drive-wheels	engine-location	width	height	engine-type	engine-size	horsepower	city-mpg	highway-mpg
0	3	?	alfa-romero	gas	convertible	rwd	front	64.1	48.8	dohc	130	111	21	27
1	3	?	alfa-romero	gas	convertible	rwd	front	64.1	48.8	dohc	130	111	21	27
2	1	?	alfa-romero	gas	hatchback	rwd	front	65.5	52.4	ohcv	152	154	19	26

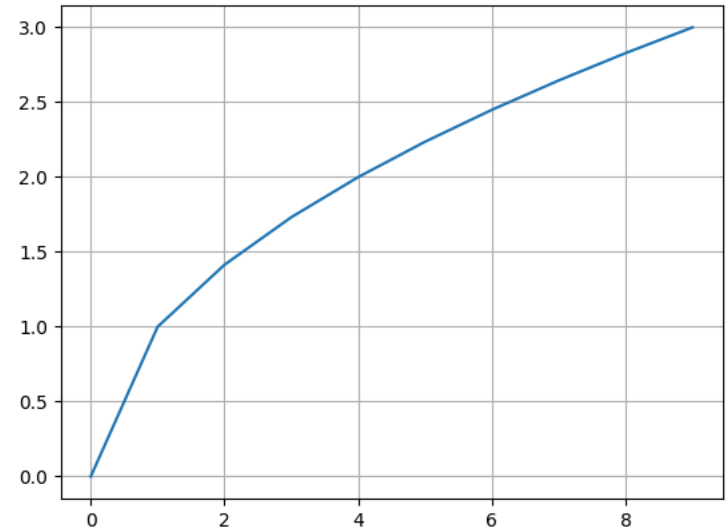
Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 15 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   symboling          205 non-null    int64
 1   normalized-losses  205 non-null    object
 2   make               205 non-null    object
 3   fuel-type          205 non-null    object
 4   body-style         205 non-null    object
 5   drive-wheels       205 non-null    object
 6   engine-location    205 non-null    object
 7   width              205 non-null    float64
 8   height             205 non-null    float64
 9   engine-type        205 non-null    object
 10  engine-size        205 non-null    int64
 11  horsepower         205 non-null    object
 12  city-mpg           205 non-null    int64
```

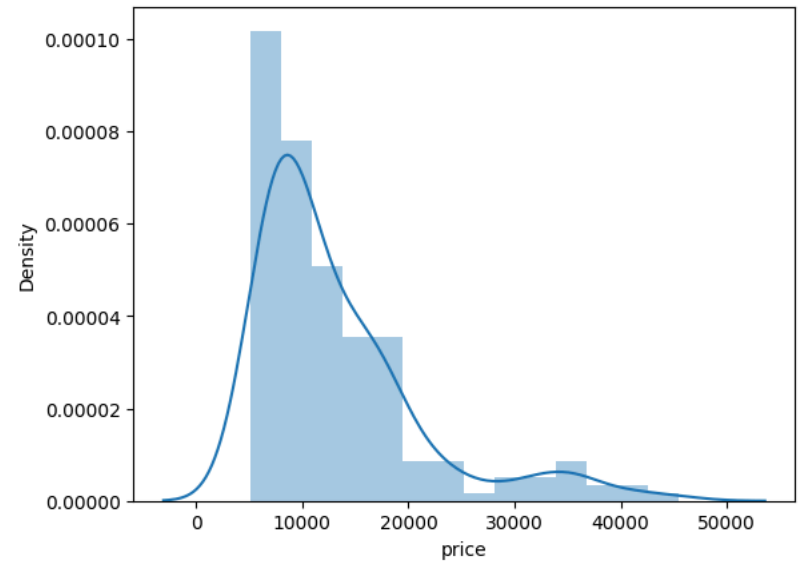
```
 13  highway-mpg        205 non-null    int64
 14  price              205 non-null    int64
dtypes: float64(2), int64(5), object(8)
memory usage: 24.2+ KB
```

```
x=np.arange(10)
y=np.sqrt(x)

sns.lineplot(y)
plt.grid(True)
plt.show()
```

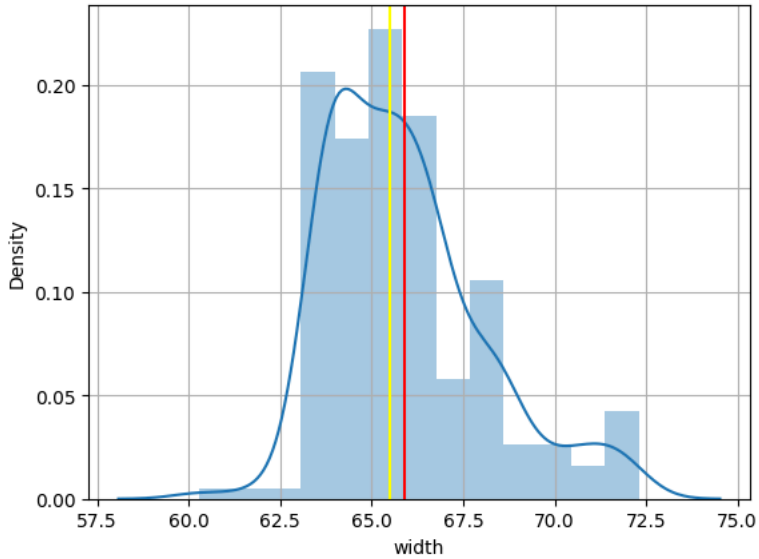



```
sns.distplot(df["price"])
```

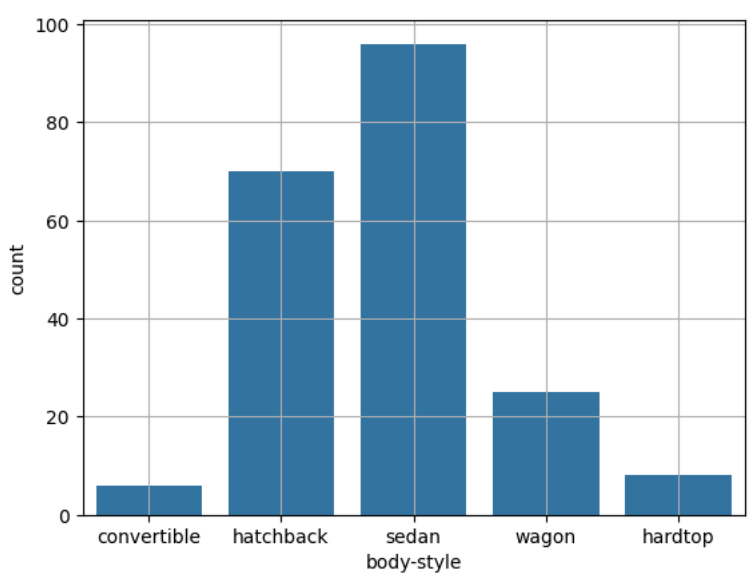
```
<Axes: xlabel='price', ylabel='Density'>
```



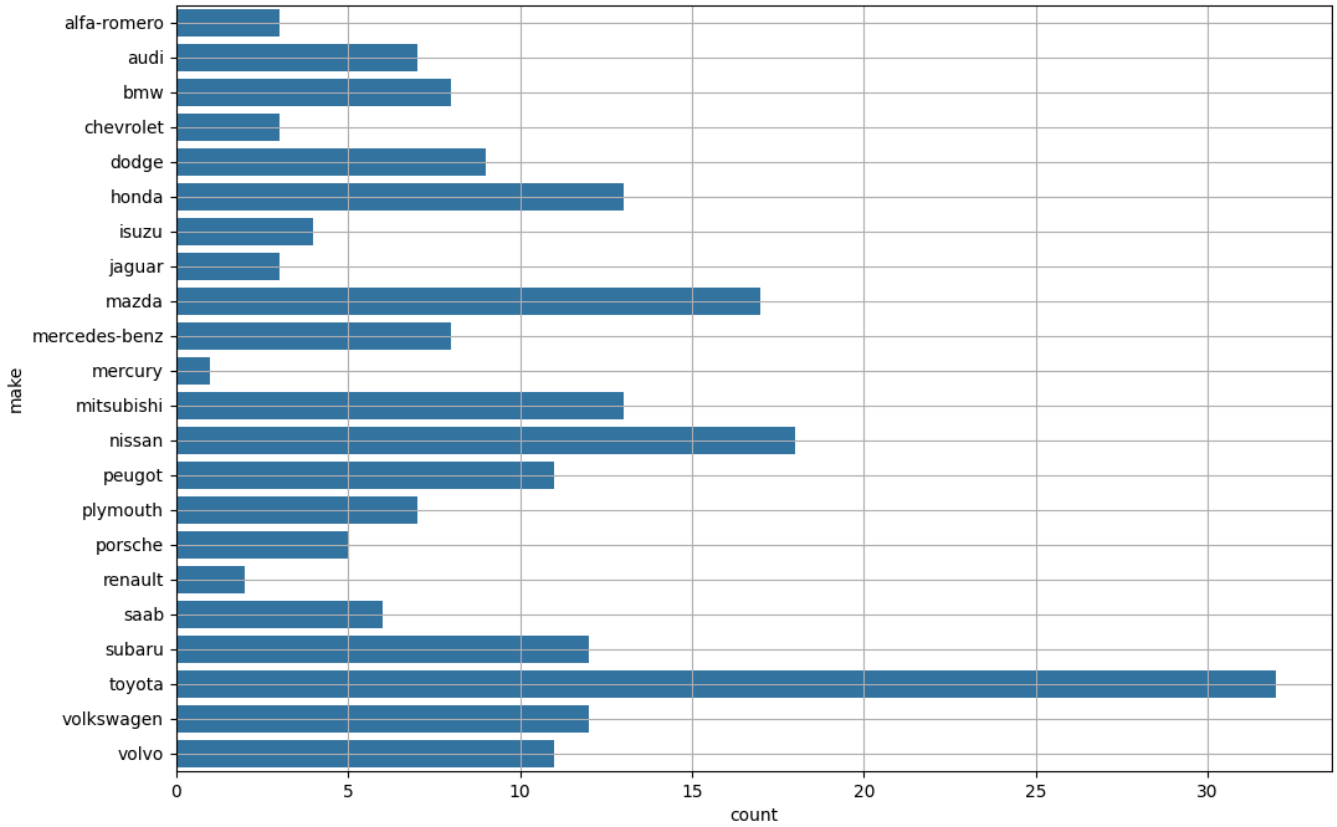

```
mwidth=df["width"].mean()
dwidth=df["width"].median()
sns.distplot(df["width"])
plt.axvline(mwidth, color="red")
plt.axvline(dwidth, color="yellow")
plt.grid(True)
```



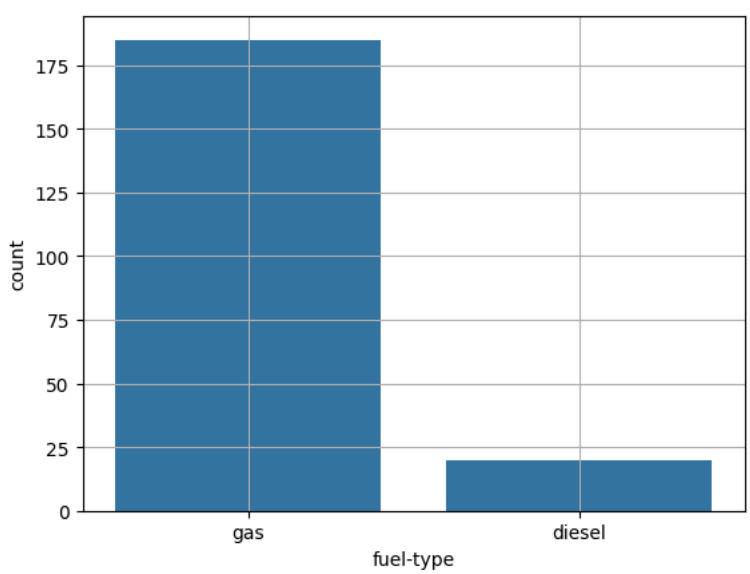

```
sns.countplot(data=df, x="body-style")
plt.grid(True)
```



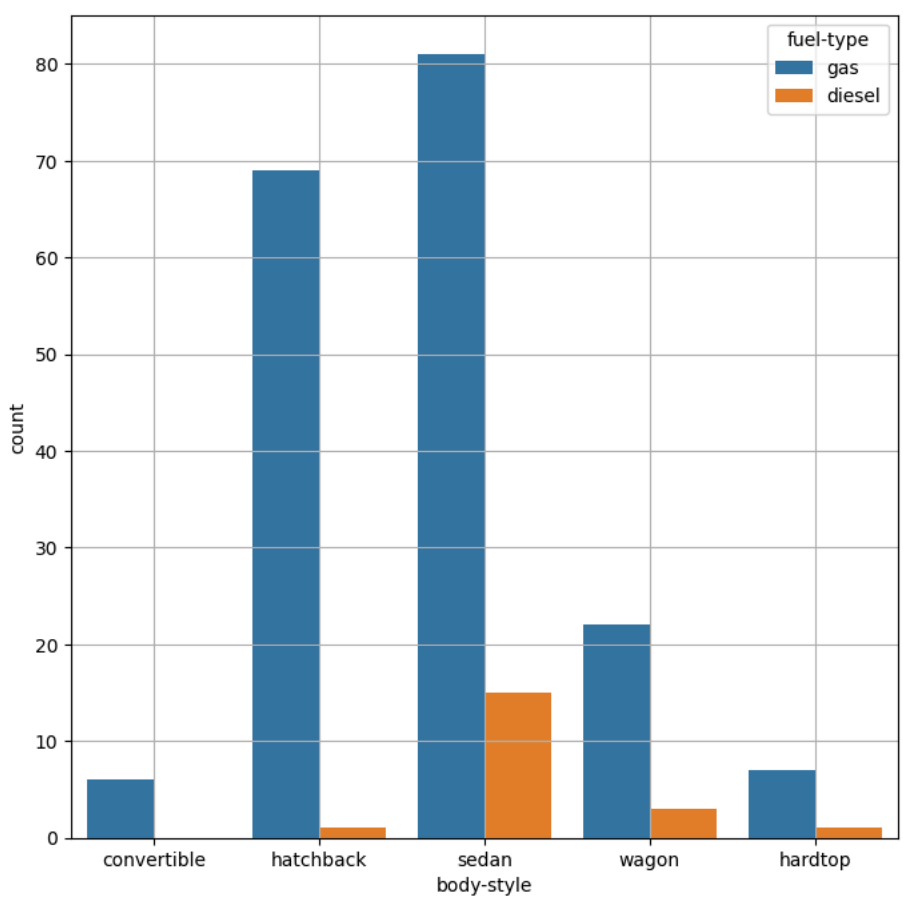

```
plt.figure(figsize=(12,8))
sns.countplot(data=df, y="make")
plt.grid(True)
```




```
sns.countplot(data=df, x="fuel-type")
plt.grid(True)
```

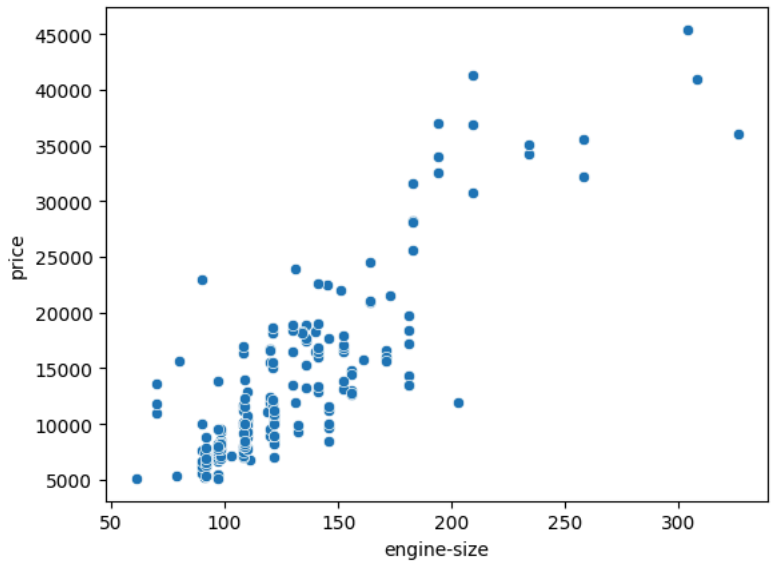



```
plt.figure(figsize=(8,8))
sns.countplot(data=df, x="body-style", hue="fuel-type")
plt.grid(True)
```



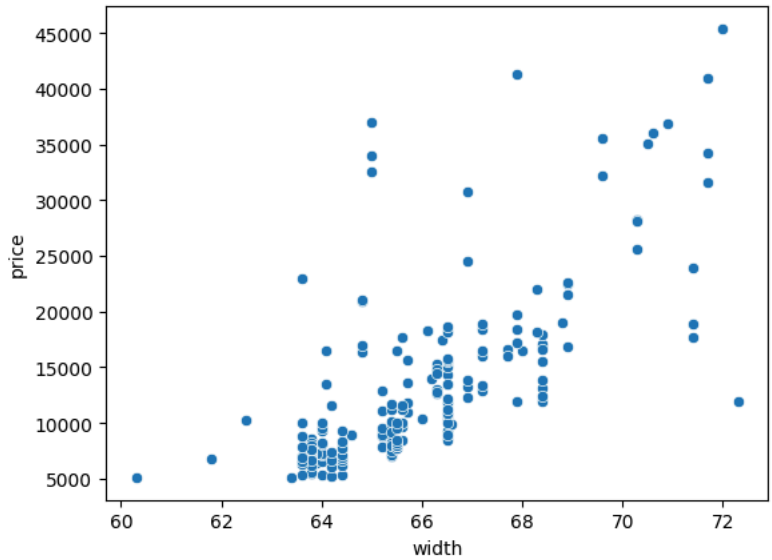

```
sns.scatterplot(data=df, x="engine-size", y="price")
```

```
<Axes: xlabel='engine-size', ylabel='price'>
```



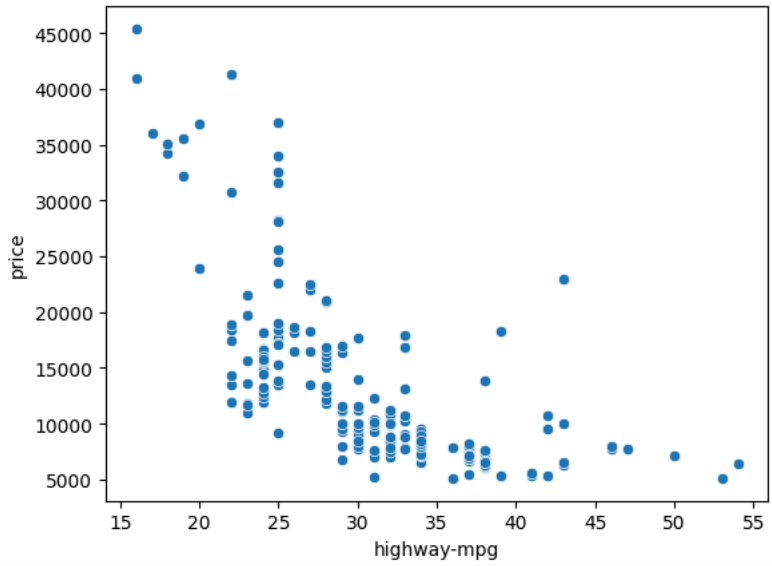

```
sns.scatterplot(data=df, x="width", y="price")
```

```
<Axes: xlabel='width', ylabel='price'>
```



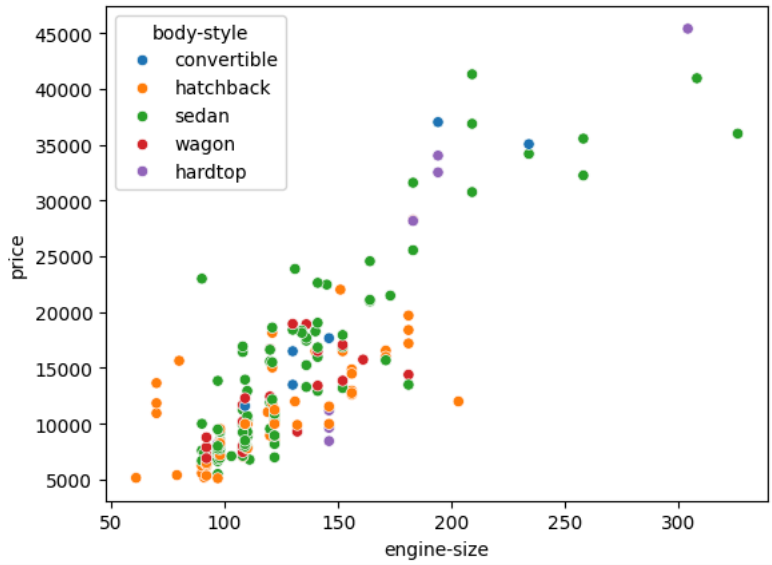

```
sns.scatterplot(data=df, x="highway-mpg", y="price")
```

```
<Axes: xlabel='highway-mpg', ylabel='price'>
```



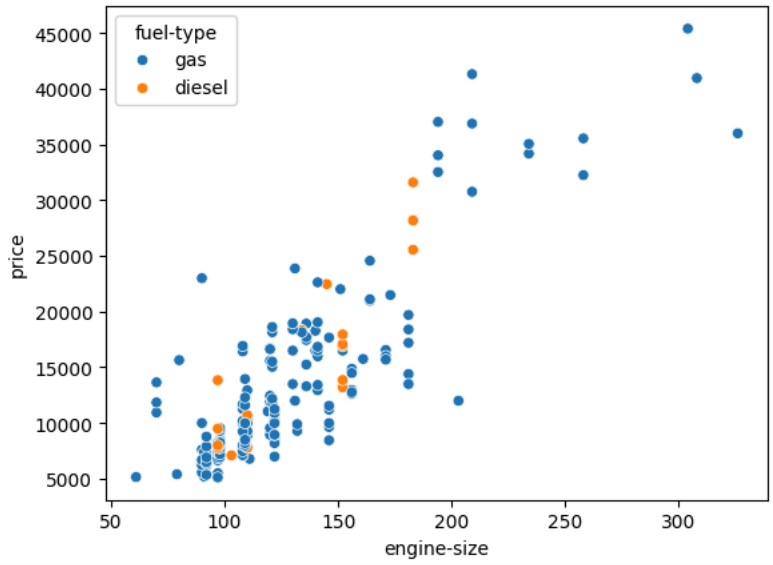

```
sns.scatterplot(data=df, x="engine-size", y="price", hue="body-style")
```

```
<Axes: xlabel='engine-size', ylabel='price'>
```



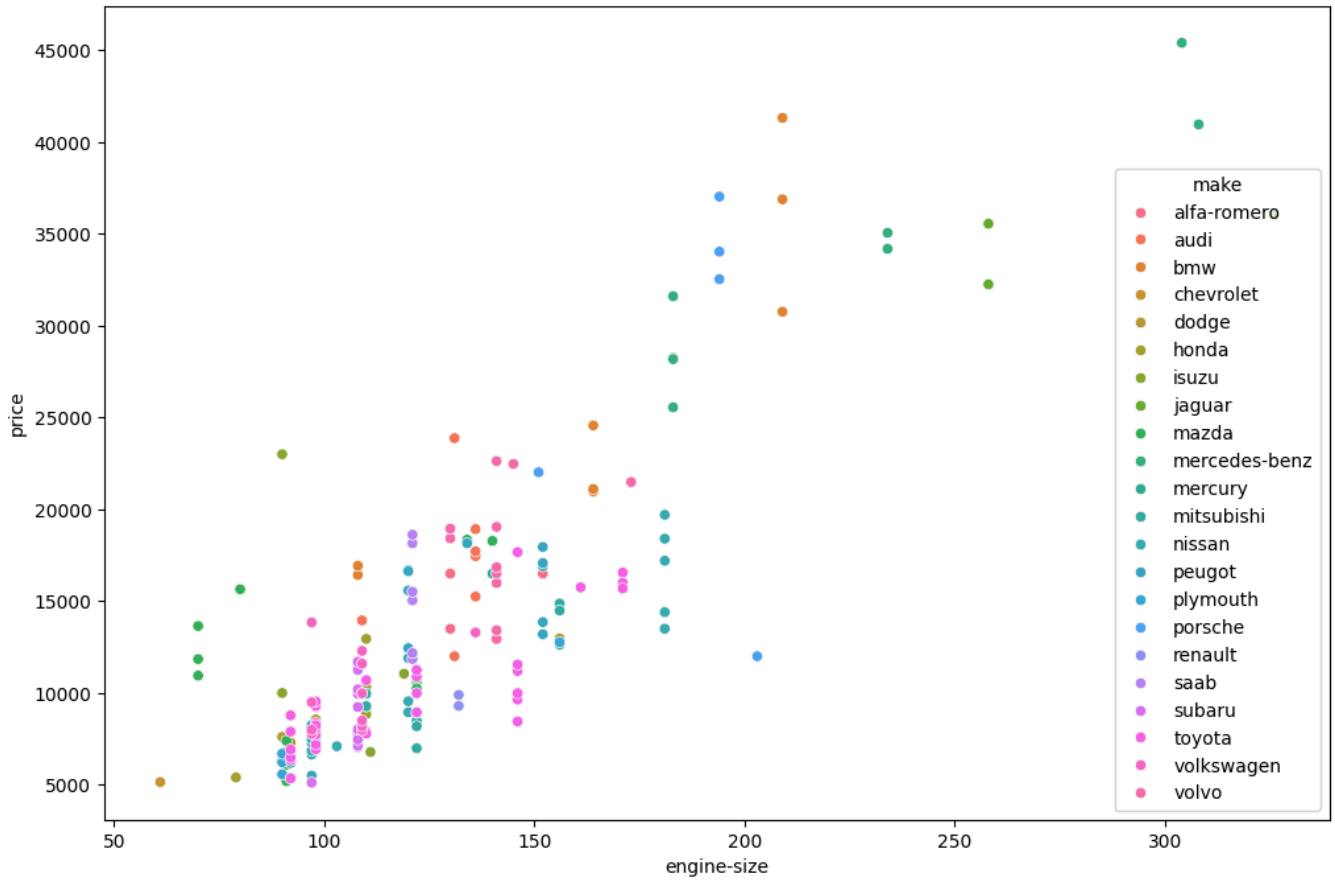

```
sns.scatterplot(data=df, x="engine-size", y="price", hue="fuel-type")
```

```
<Axes: xlabel='engine-size', ylabel='price'>
```



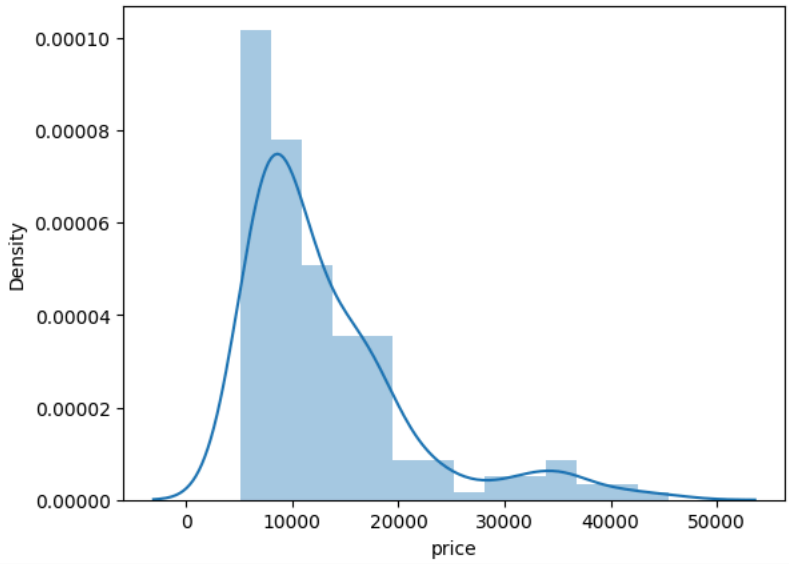

```
plt.figure(figsize=(12,8))
sns.scatterplot(data=df, x="engine-size", y="price", hue="make")
```

```
<Axes: xlabel='engine-size', ylabel='price'>
```




```
sns.distplot(df["price"])
```

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



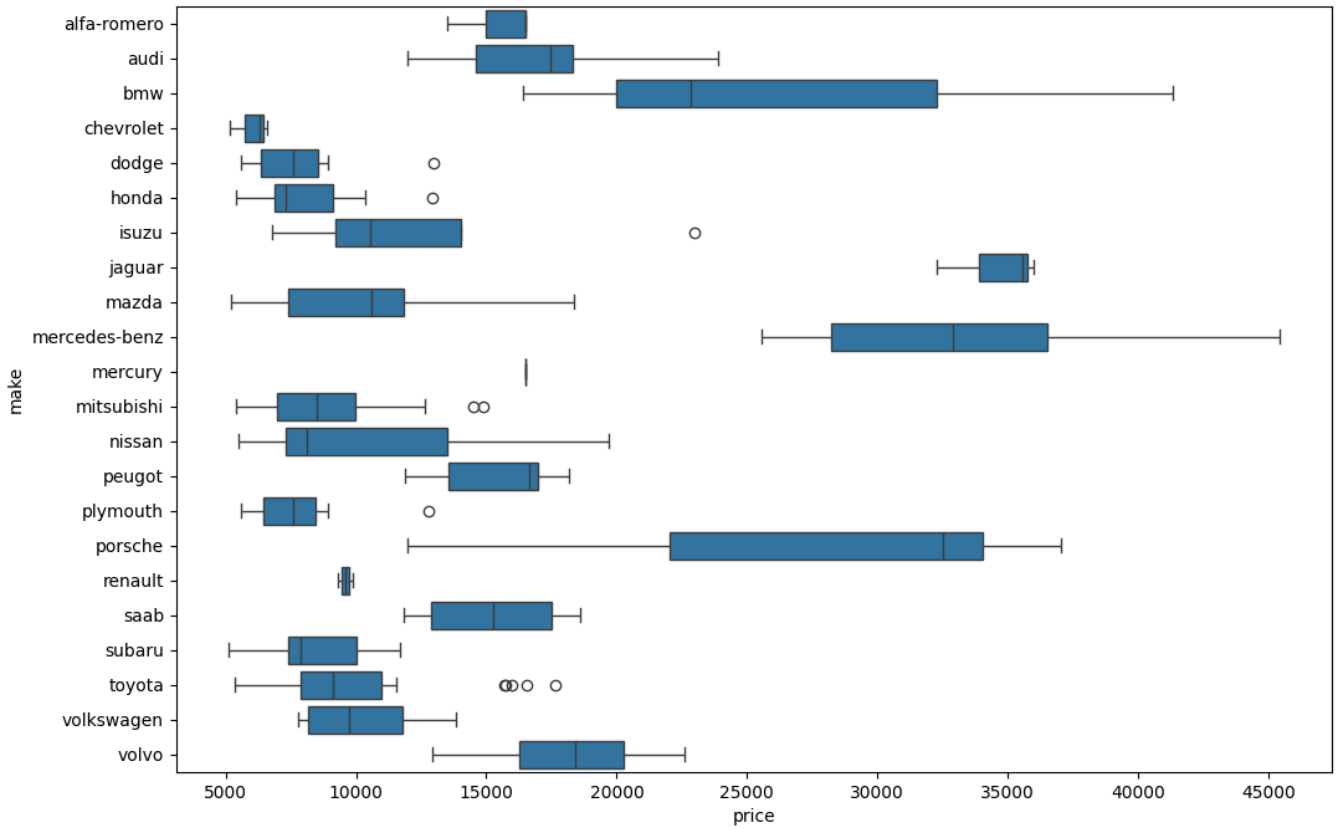

```
plt.figure(figsize=(12,8))
sns.boxplot(datat=df, x="price")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-20-a71f48c625b3> in <cell line: 0>()
      1 plt.figure(figsize=(12,8))
----> 2 sns.boxplot(datat=df, x="price")

                              5 frames
/usr/local/lib/python3.11/dist-packages/seaborn/_core/data.py in _assign_variables(self, data, variables)
    230                 else:
    231                     err += "An entry with this name does not appear in `data`."
--> 232                 raise ValueError(err)
    233
    234             else:

ValueError: Could not interpret value `price` for `x`. Value is a string, but `data` was not passed.

<Figure size 1200x800 with 0 Axes>
```

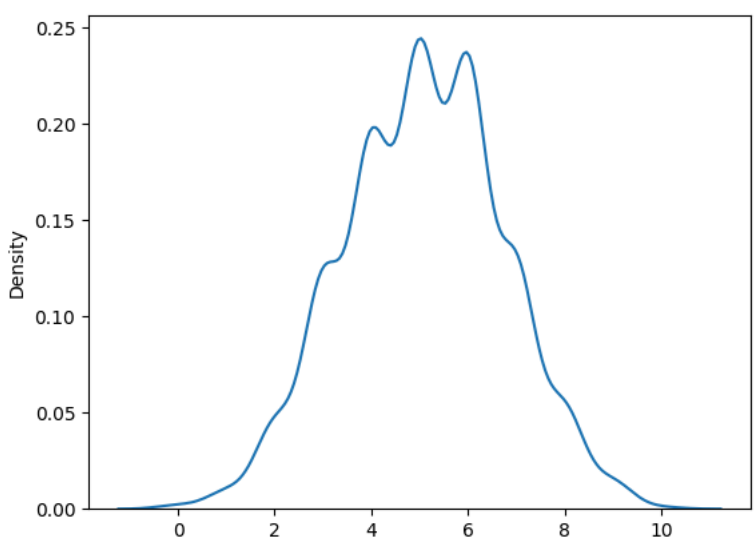
Next steps: Explain error

```
plt.figure(figsize=(12,8))
sns.boxplot(data=df, x="price", y="make")
```

```
<Axes: xlabel='price', ylabel='make'>
```

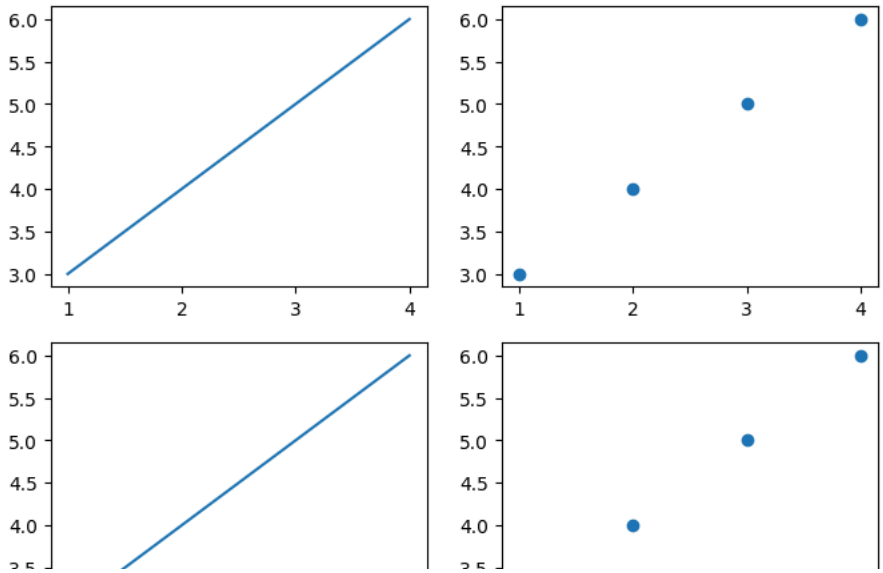



```
from numpy import random
sns.distplot(random.binomial(n=10, p=0.5, size=1000), hist=False, kde=True)
plt.show()
```




```
#subplots

x=[1,2,3,4]
y=[3,4,5,6]

fig, axes=plt.subplots(2,2, figsize=(8,6))
axes[0,0].plot(x,y)
axes[0,1].scatter(x,y)
axes[1,0].plot(x,y)
axes[1,1].scatter(x,y)
```

```
<matplotlib.collections.PathCollection at 0x7eb676507f50>
```




```
fig,axes=plt.subplots(2,2, figsize=(16,10))
sns.distplot(df["price"], ax=axes[0,0])
sns.countplot(data=df, y="make", ax=axes[0,1])
sns.scatterplot(data=df, x="width", y="price", hue="body-style", ax=axes[1,0])
sns.boxplot(data=df, x="price", ax=axes[1,1])
plt.show()
```