Course: Artificial Intelligence and Machine Learning

Code: 20CS51I, WEEK - 4

SESSION – 6 **Multivariate plotting**

- Multivariate distribution plot
- Multivariate comparison plot
- Multivariate relationship plot
- Multivariate composition plot

Seaborn is an interface built on top of Matplotlib that uses short lines of code to create and style statistical plots from Pandas data frames. We will use the vehicles dataset from Kaggle that is under the Open database license. The code below imports the required libraries, sets the style, and loads the dataset.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as snssns.set_style('darkgrid')
sns.set(font_scale=1.3)cars = pd.read_csv('edited_cars.csv')
```

Multivariate distribution plot

Pair plot

A pair plot creates a grid of scatter plots to compare the distribution of pairs of numeric variables. It also features a histogram for each feature in the diagonal boxes.

Functions to use:

- sns.pairplot() — figure-level plot

The kind parameter changes the type of bivariate plots created with kind= ‘scatter’ (default), ‘kde’, ‘hist’or ‘reg’.

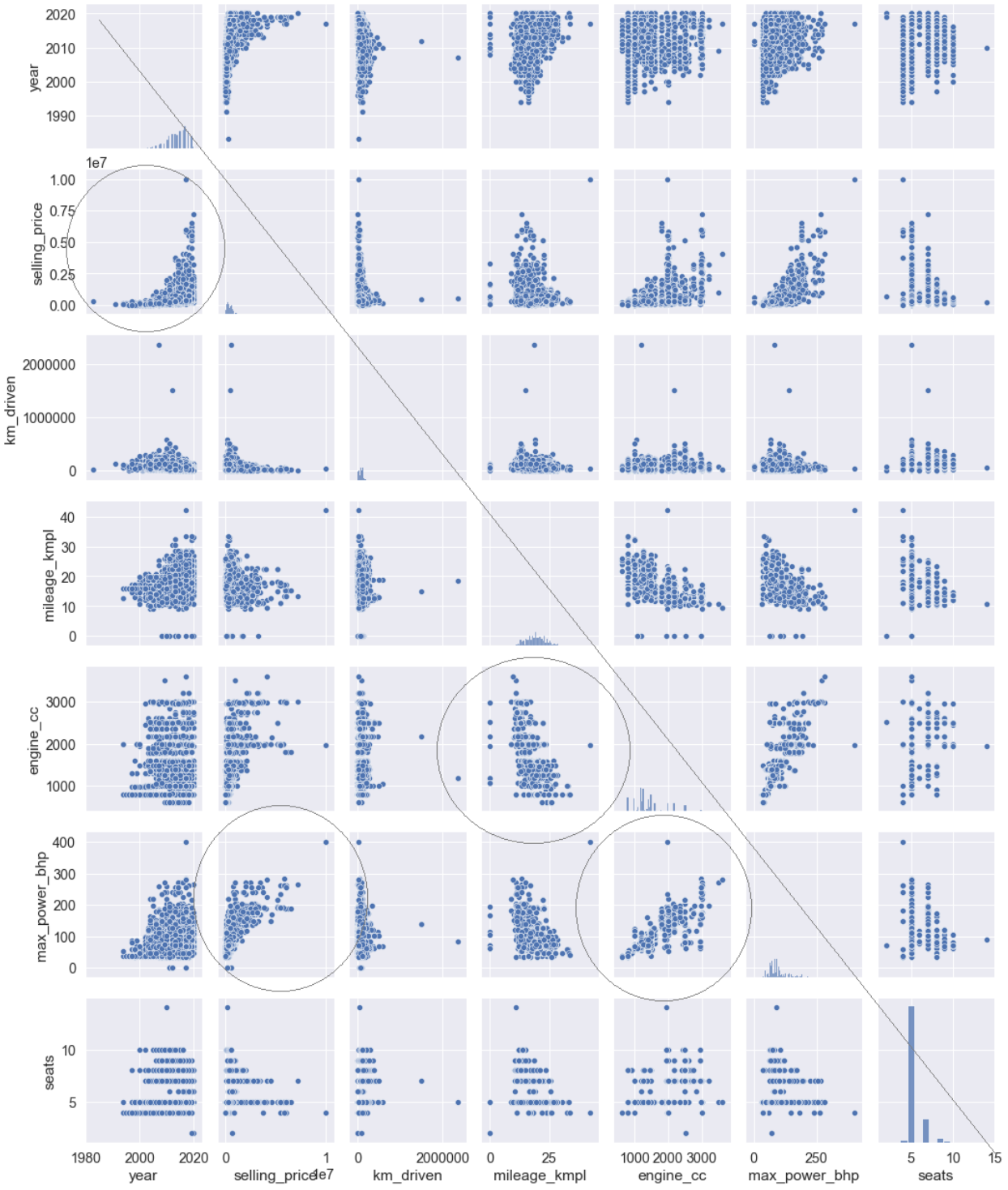
Two columns per grid (Bivariate)

```
sns.pairplot(cars);
```

What to look out for:

- Scatter plots showing either positive linear relationships (if x increases, y increases) or negative (if x increases, y decreases).
- Histograms in the diagonal boxes that show the distribution of individual features.

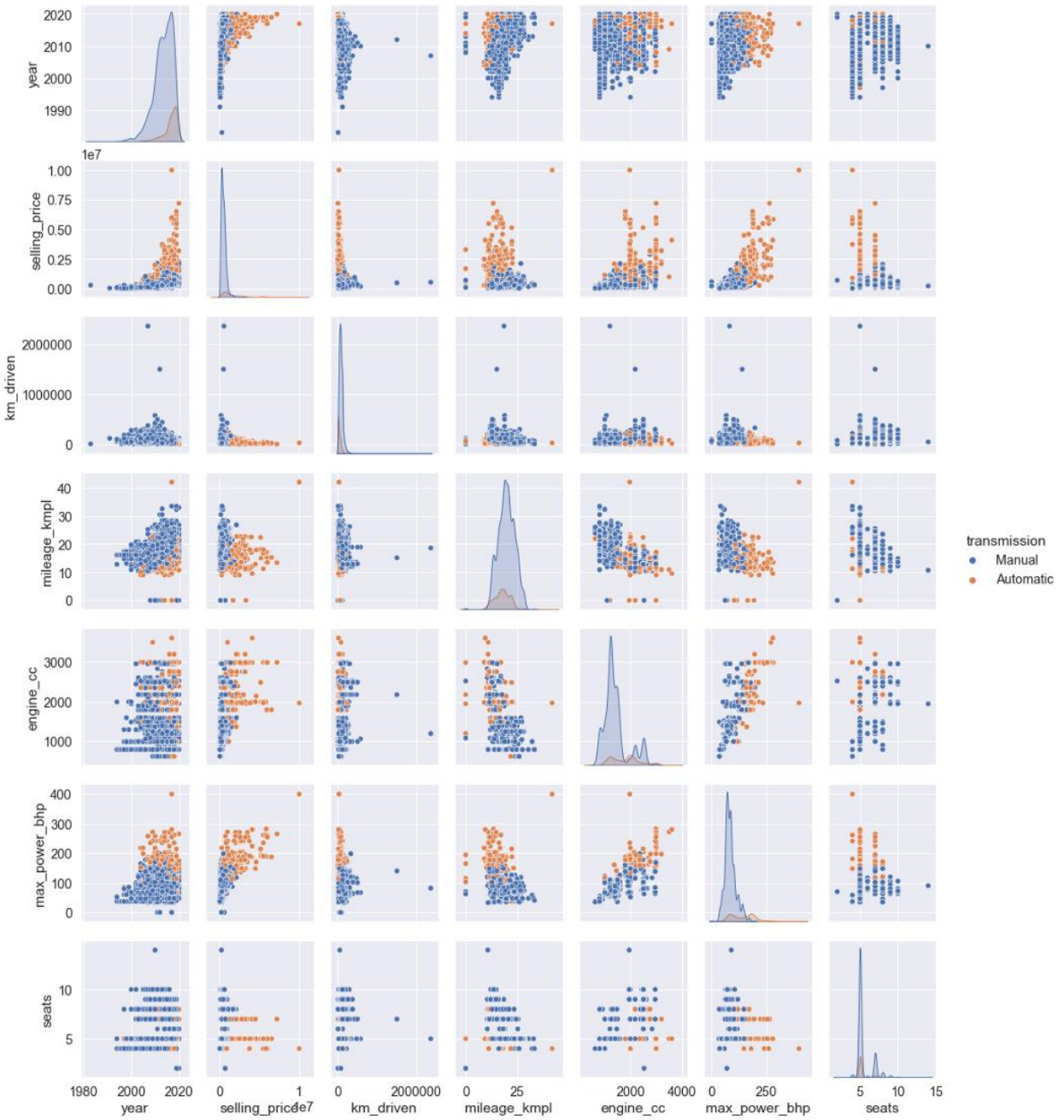
In the pair plot below, the circled plots show an apparent linear relationship. The diagonal line points out the histograms for each feature, and the pair plot's top triangle is a mirror image of the bottom.



Three columns (multivariate): two numeric and one categorical

We can add a third variable that segments the scatter plots by color using the parameter hue='cat_col'.

```
sns.pairplot(
   data=cars,
   aspect=.85,
   hue='transmission');
```



Multivariate Relationship Plots

Scatter plot

A scatter plot shows the relationship between two numeric features by using dots to visualize how these variables move together.

Functions to use:

- sns.scatterplot() — axes-level plot
- sns.relplot(kind='line') — figure-level

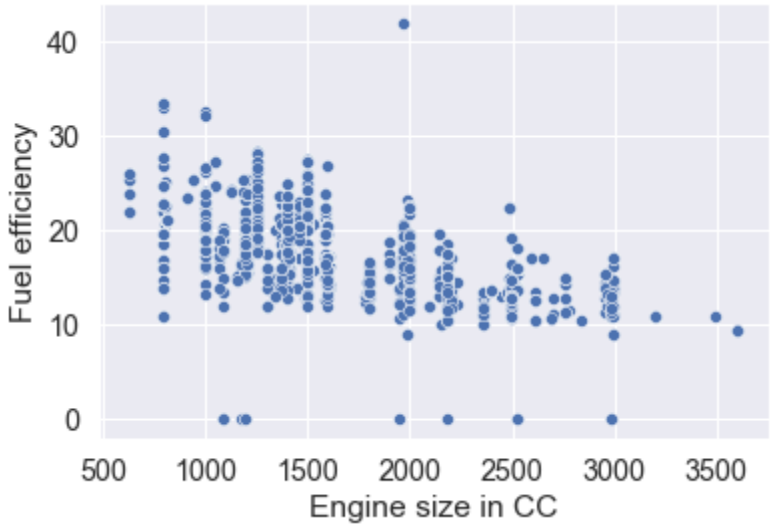
Functions with regression line;

- sns.regplot() — axes-level
- sns.lmplot() — figure-level

Two numeric columns (bivariate)

sns.scatterplot(x='num_col1', y='num_col2', data=df) — Let us visualize the engine size with the mileage (efficiency) of the vehicle.

```
sns.set(font_scale=1.3)sns.scatterplot(
   x='engine_cc',
   y='mileage_kmpl',
   data=cars)plt.xlabel(
   'Engine size in CC')
plt.ylabel(
   'Fuel efficiency')
```

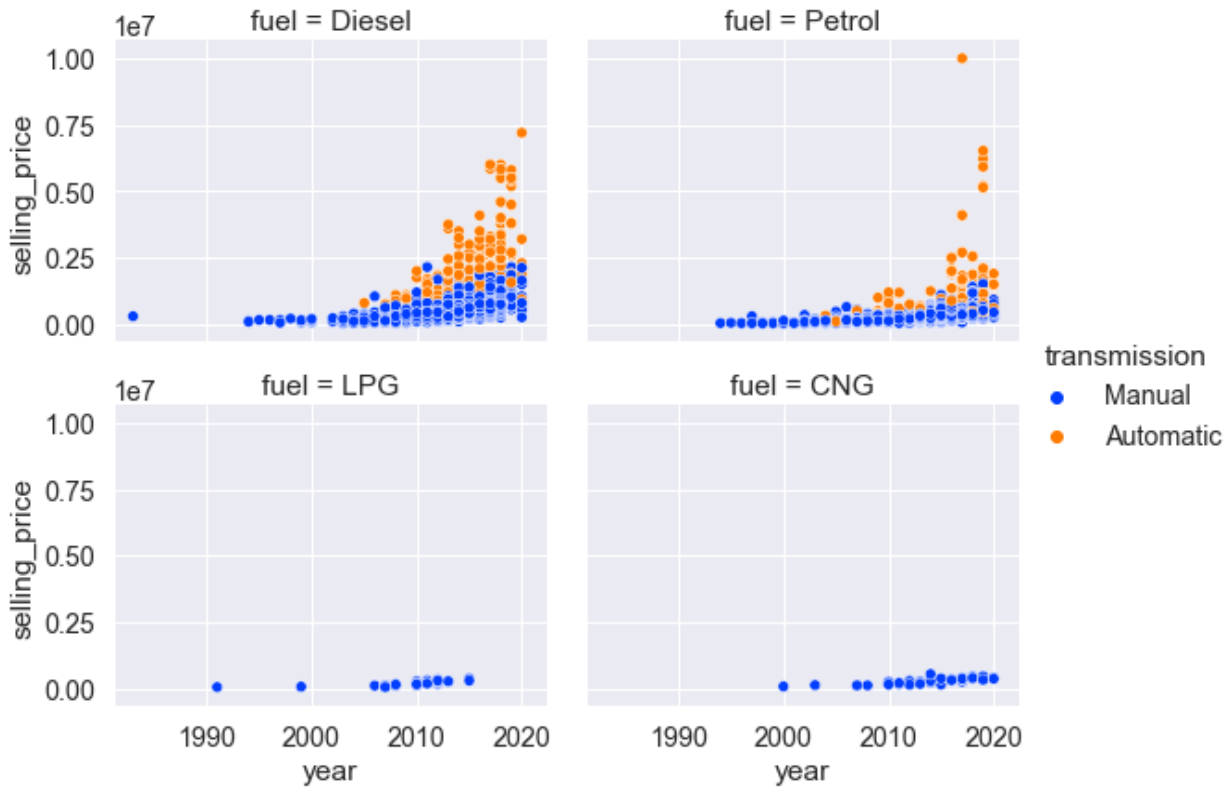


Scatter plot

Four columns: two numeric and two categorical.

sns.relplot(x,y,data, hue='cat_col1', col='cat_col2') — the col_wrap parameter wraps columns after this width so that the subplots span multiple rows.

```
sns.relplot(
    x='year',
    y='selling_price',
    data=cars,
    palette='bright',
    height=3, aspect=1.3,
    kind='scatter',
    hue='transmission',
    col='fuel',
    col_wrap=2);
```



Relational scatterplots

Multivariate Comparison plots

Bar plot

The bar chart uses bars of different heights to compare the distribution of a numeric variable between groups of a categorical variable.

By default, bar heights are estimated using the "mean". The estimator parameter changes this aggregation function by using python's inbuilt functions such as estimator=max or len, or NumPy functions like np.max and np.median.

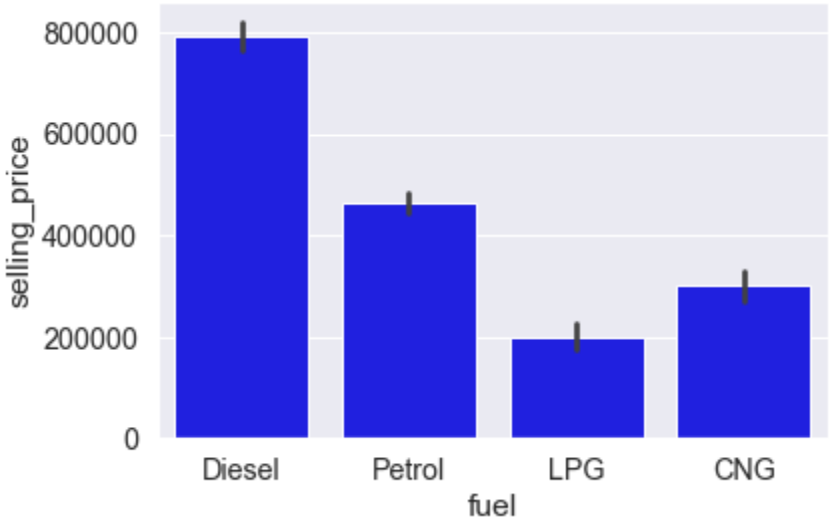
Functions to use:

- sns.barplot() — axes-level plot
- sns.catplot(kind='bar') — figure-level plot

Two columns (bivariate): numeric and categorical

sns.barplot(x='cat_col', y='num_col', data=df)

```
sns.barplot(
   x='fuel',
   y='selling_price',
   data=cars,
   color='blue',
   # estimator=sum,
   # estimator=np.median);
```

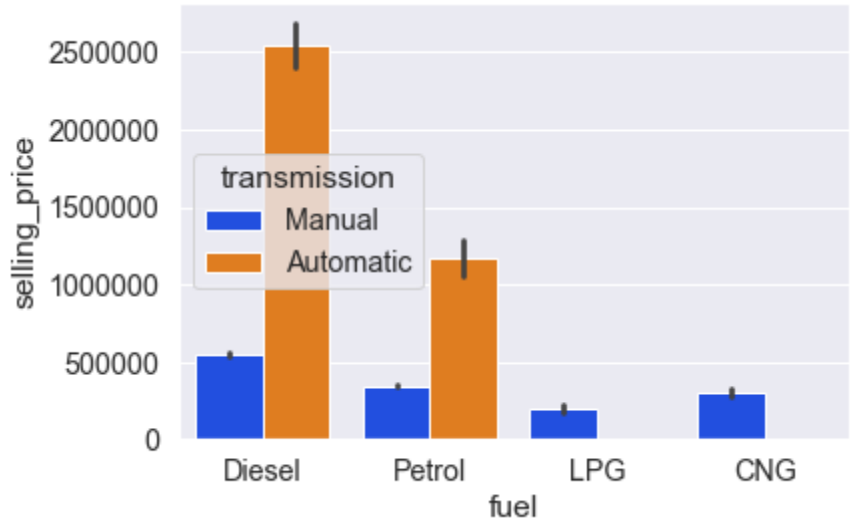


Barplot

Three columns (multivariate): two categorical and one numeric.

sns.barplot(x, y, data, hue='cat_col2')

```
sns.barplot(
   x='fuel',
   y='selling_price',
   data=cars,
   palette='bright'
   hue='transmission');
```

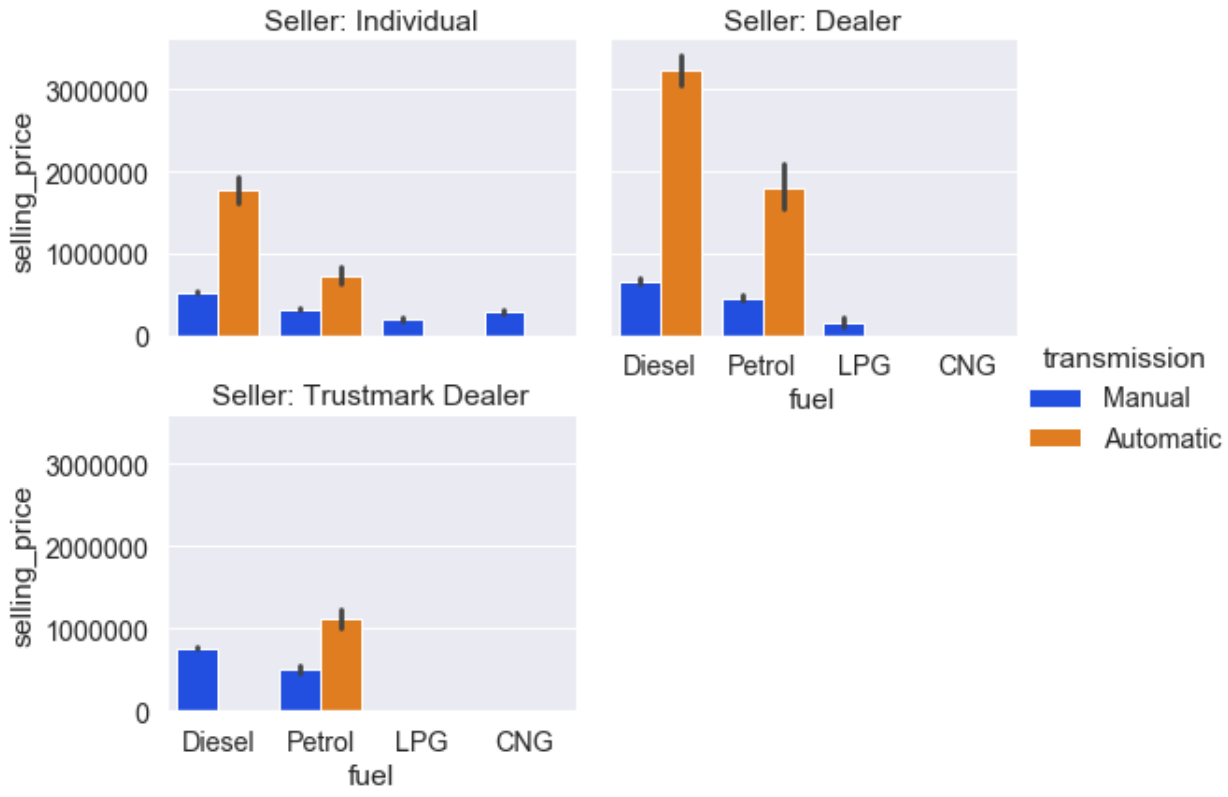


Barplot with hue

Four columns: three categorical and one numeric

sns.catplot(x, y, data, kind='bar', hue='cat_col2', col='cat_col3') — Use the col_wrap parameter to wrap columns after this width so that the subplots span multiple rows.

```
g = sns.catplot(
      x='fuel',
      y='selling_price',
      data=cars,
      palette='bright',
      height=3, aspect=1.3,
      kind='bar',
      hue='transmission',
      col ='seller_type',
      col_wrap=2)g.set_titles(
  'Seller: {col_name}');
```



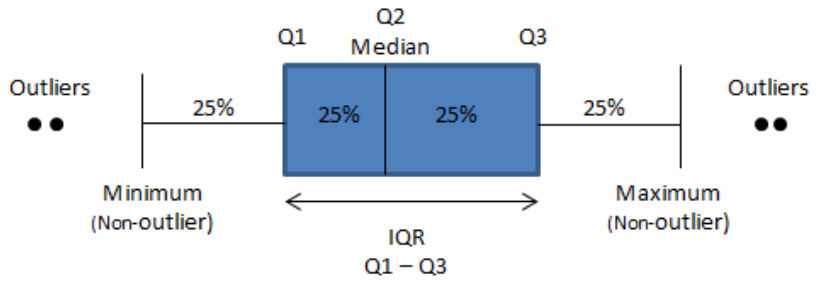
Categorical barplot

Multivariate Composition Plots

Shows proportions and percentages for groups

Box plot

A box plot visualizes the distribution between numeric and categorical variables by displaying the information about the quartiles.



From the plots, you can see the minimum value, median, maximum value, and outliers for every category class.

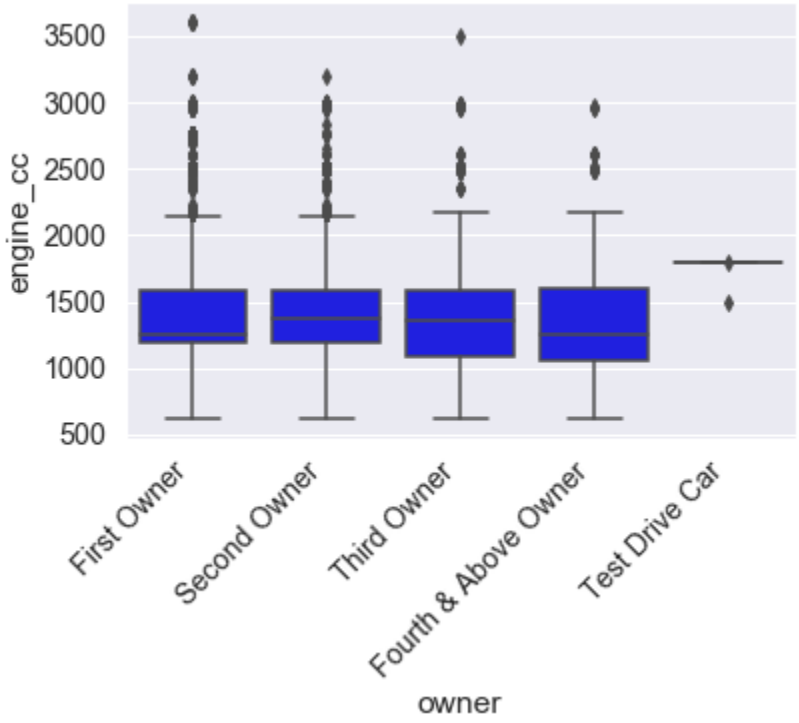
Functions to use:

- sns.boxplot() — axes-level plot
- sns.catplot(kind='box') — figure-level plot

Two columns (bivariate): one categorical and one numeric

sns.boxplot(x='cat_col', y='num_col', data=df)

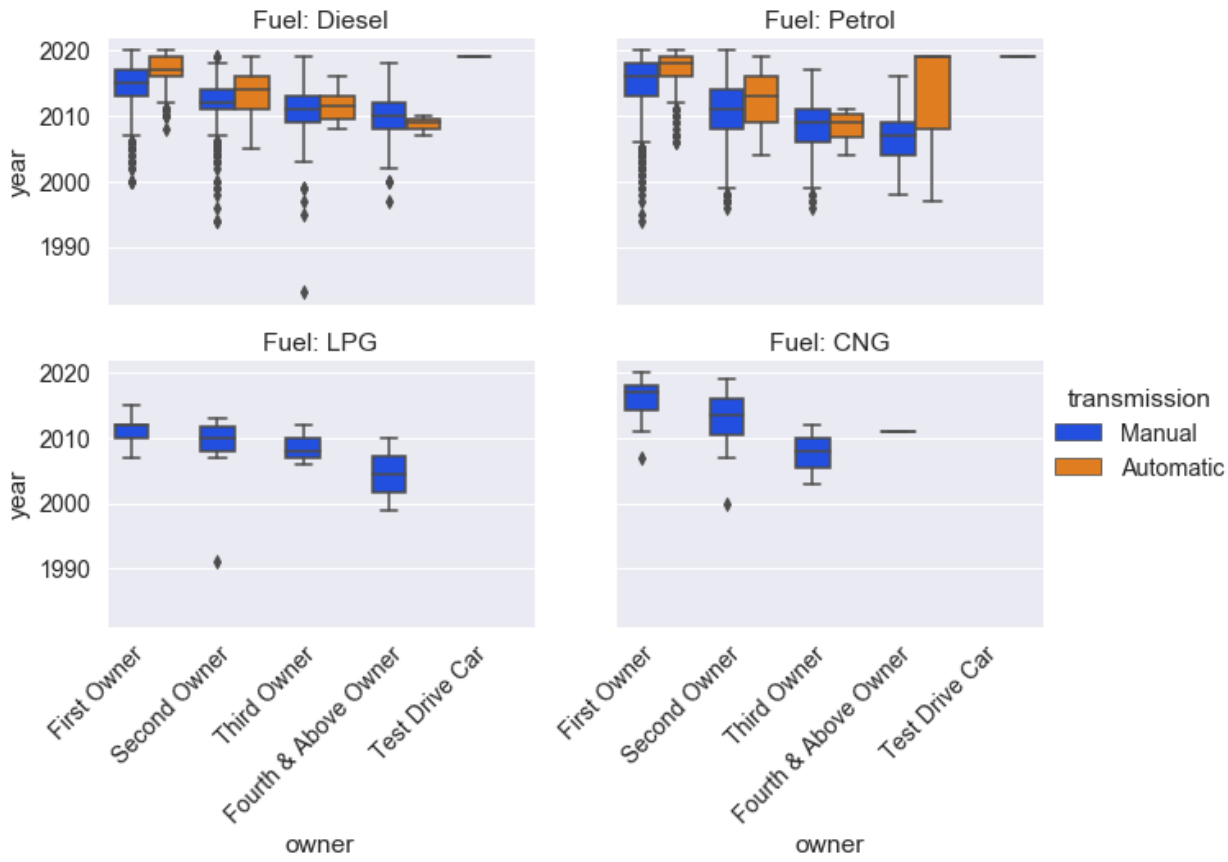
```
sns.boxplot(
   x='owner',
   y='engine_cc',
   data=cars,
   color='blue')plt.xticks(rotation=45,
        ha='right');
```



Four columns: three categorical and one numeric

sns.catplot(x, y, data, kind='box', hue='cat_col2', col=’cat_col3')

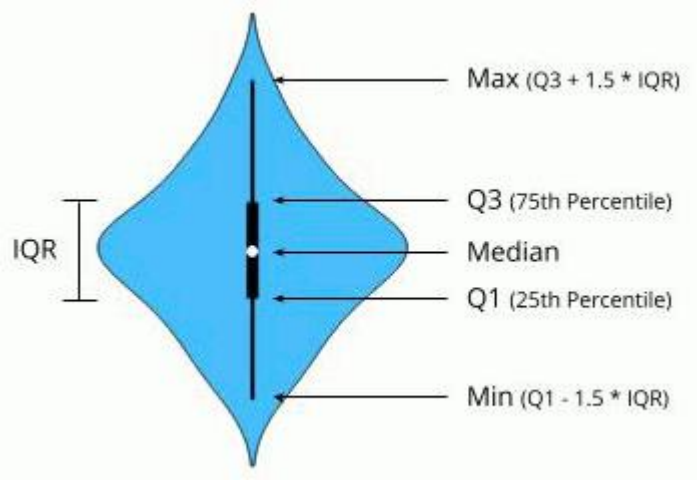
```
g = sns.catplot(
        x='owner',
        y='year',
        data=cars,
        palette='bright',
        height=3, aspect=1.5,
        kind='box',
        hue='transmission',
        col='fuel',
        col_wrap=2)g.set_titles(
    'Fuel: {col_name}');g.set_xticklabels(
    rotation=45, ha='right')
```



Categorical boxplots

Violin plot

In addition to the quartiles displayed by a box plot, a violin plot draws a Kernel density estimate curve that shows probabilities of observations at different areas.



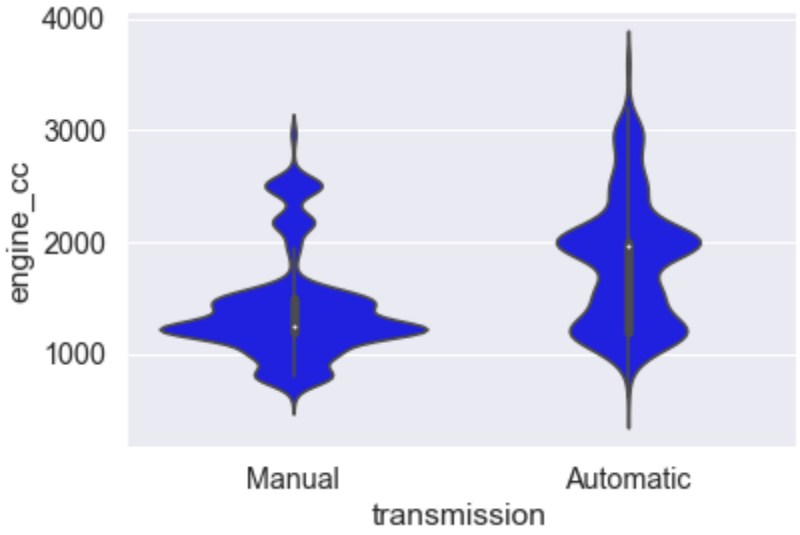
Functions to use:

- sns.violinplot() — axes-level plot
- sns.catplot(kind='violin') — figure-level plot

Two columns (bivariate): numeric and categorical.

sns.violinplot(x='cat_col', y='num_col', data=df)

```
sns.violinplot(
    x='transmission',
    y='engine_cc',
    data=cars,
    color='blue');
```



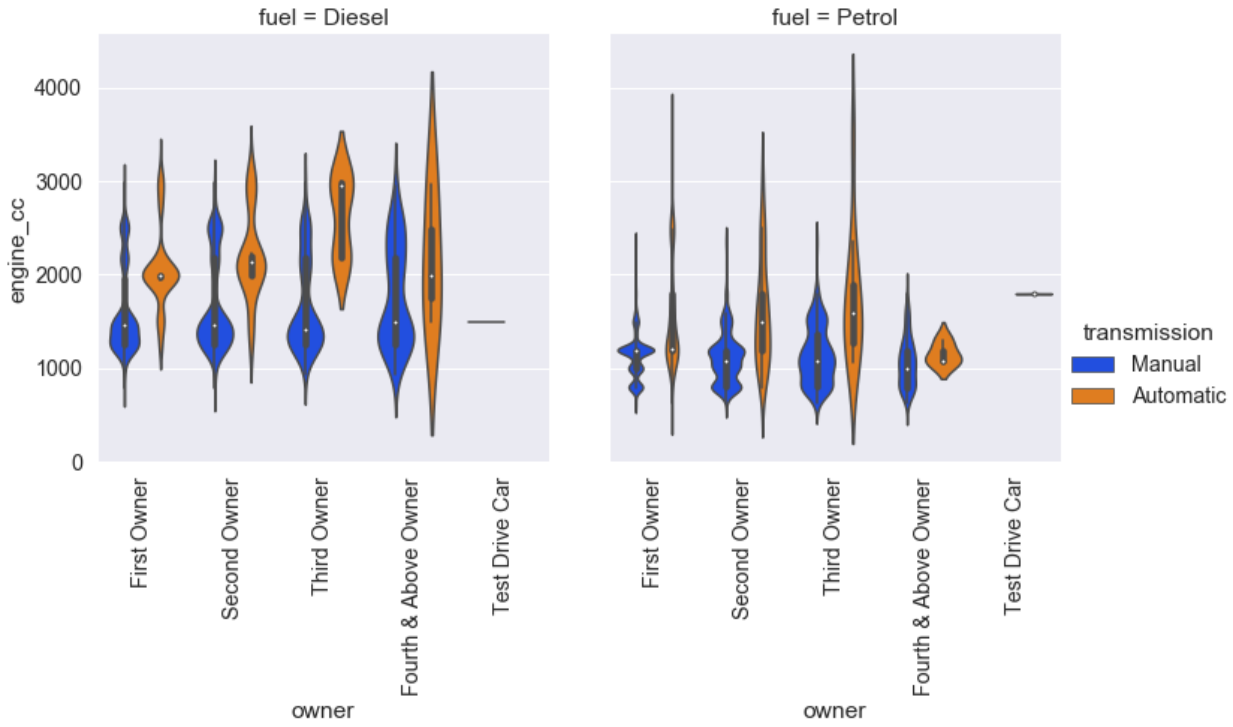
Violin plot

Four columns: three categorical and one numeric

sns.catplot(x, y, data, kind='violin', hue='cat_col2', col='cat_col3') — Here, we filter the data for only 'diesel' and 'petrol' fuel types.

```
my_df = cars[cars['fuel'].isin(['Diesel','Petrol'])]g = sns.catplot(
        x="owner",
```

```
    y="engine_cc",
    data=my_df,
    palette='bright',
    kind = 'violin',
    hue="transmission",
    col = 'fuel')g.set_xticklabels(
    rotation=90);
```



https://towardsdatascience.com/10-must-know-seaborn-functions-for-multivariate-data-analysis-in-python-7ba94847b117

https://github.com/suemnjeri/medium-articles/blob/main/Seaborn%20plot%20functions/Multivariate%20analysis%20vehicle%20dataset.ipynb