

Visualisasi Data

Visualisasi Variabel

Visualisasi berperan peran penting dalam bidang machine learning dan data science. Seringkali kita perlu menyaring informasi kunci yang ditemukan dalam sejumlah data untuk menjadi bentuk yang bermakna dan mudah dicerna. Visualisasi yang baik dapat menceritakan sebuah cerita tentang data dengan cara yang tidak dapat dilakukan oleh sebuah klaimat.

Di modul ini kita akan mengeksplorasi beberapa teknik visualisasi yang umum. Lab ini akan menggunakan toolkit seperti [Matplotlib's Pyplot](#) dan [Seaborn](#) untuk membuat gambar informatif yang memberikan informasi dan pengetahuan mengenai dataset.

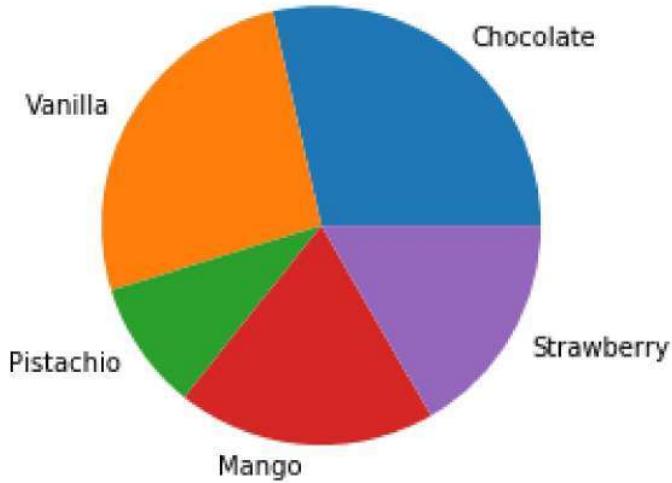
Pie Charts

Pie chart digunakan untuk menunjukkan seberapa banyak dari setiap jenis kategori dalam dataset berbanding dengan keseluruhan. Pada bagian ini kita akan membuat diagram lingkaran menggunakan kumpulan data sampel. Variabel label berisi tupel rasa es krim. Variabel voting berisi tupel voting. Data tersebut mewakili jumlah voting rase es krim favorit. Kita dapat membuat grafik menggunakan library Pyplot Matplotlib. Method plt.pie () digunakan untuk membuat interface pie chart berdasarkan data rasa es krim dan jumlah voting .

```
import matplotlib.pyplot as plt

flavors = ('Chocolate', 'Vanilla', 'Pistachio', 'Mango', 'Strawberry')
votes = (12, 11, 4, 8, 7)

plt.pie(
    votes,
    labels=flavors,
)
plt.show()
```



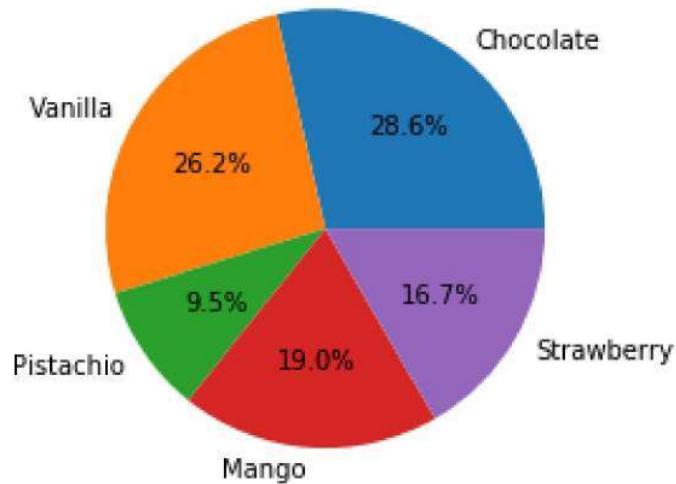
Gambar 0.1. Pie Chart perbandingan rasa es krim

Berdasarkan Gambar 0.1, kita dapat dengan mudah melihat bahwa cokelat adalah rasa yang paling populer, diikuti dengan rasa vanila. Kita dapat mengetahui hal ini dengan melihat data mentahnya. Namun, data dalam format diagram lingkaran juga dapat digunakan untuk melihat informasi lain dengan mudah, seperti fakta bahwa kombinasi cokelat dan vanila mewakili lebih dari setengah suara. Apa yang tidak kita lihat adalah persentase sebenarnya. Jika kita ingin melihat berapa persen kontribusi masing-masing rasa es krim, kita bisa menggunakan argumen autopct. Untuk nilai argumen, ada beberapa string format yang dapat digunakan untuk mengatur ketepatan tampilan data. Coba ubah nilainya menjadi `%1.0 %%` dan `%1.2f %%`. Apa yang terjadi?

```
import matplotlib.pyplot as plt
```

```
flavors = ('Chocolate', 'Vanilla', 'Pistachio', 'Mango', 'Strawberry')
votes = (12, 11, 4, 8, 7)

plt.pie(
    votes,
    labels=flavors,
    autopct='%1.1f%%',
)
plt.show()
```



Gambar 0.2. Penambahan argument persentase

Sekarang kita dapat melihat persentase kontribusi setiap rasa es krim secara keseluruhan (Gambar 0.2). Satu hal yang masih sedikit membingungkan tentang bagan ini adalah pilihan warnanya. Kita dapat merubah warna dari setiap rasa es krim Pie Chart. Matplotlib memungkinkan Anda mengubah warna yang ditampilkan pada bagan dengan memasukkan nilai warna. Anda dapat menggunakan shorcut yang telah diprogram seperti 'b' untuk biru dan 'g' untuk hijau.

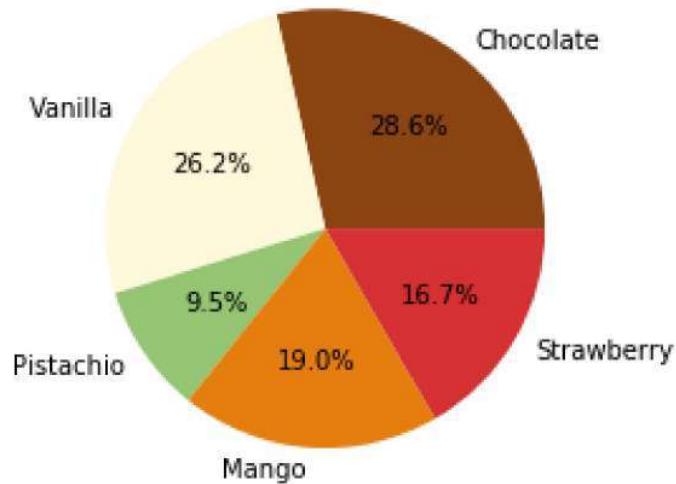
Dalam kasus ini, kita menggunakan warna html. Warna ini adalah adalah nilai enam karakter di mana dua karakter pertama mewakili jumlah warna merah, dua karakter berikutnya adalah jumlah warna hijau, dan dua karakter terakhir mewakili jumlah warna biru. Anda dapat menemukan custom warna yang lebih banyak dengan mencari kata kunci 'kode warna html'.

Pada code dibawah ini assignment warna dilakukan untuk setiap rasa (Gambar 0.3).

```
import matplotlib.pyplot as plt
```

```
flavors = ('Chocolate', 'Vanilla', 'Pistachio', 'Mango',
'Strawberry')
votes = (12, 11, 4, 8, 7)
colors = ('#8B4513', '#FFF8DC', '#93C572', '#E67F0D', '#D53032')

plt.pie(
    votes,
    labels=flavors,
    autopct='%1.1f%%',
    colors=colors,
)
plt.show()
```



Gambar 0.3. Penambahan warna pada masing-masing kelas

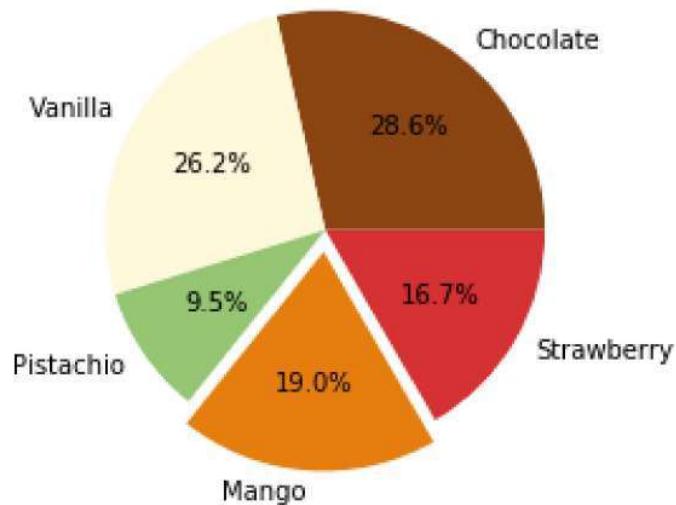
Sekarang mari kita bayangkan kita sedang mempersiapkan bagan ini untuk presentasi, dan kita ingin memanggil salah satu rasa secara khusus. Mungkin mangga baru dipasarkan, dan kita ingin melakukan highlight terhadap data mangga.

Untuk melakukan ini kita bisa menggunakan argumen explode. Ini memungkinkan kita menyetel offset untuk setiap irisan pai dari tengah. Pada contoh di bawah ini kita mendorong mangga keluar sebesar 0,1 sambil menjaga semua potongan lainnya tetap berada ditengah.

```
import matplotlib.pyplot as plt

flavors = ('Chocolate', 'Vanilla', 'Pistachio', 'Mango',
'Strawberry')
votes = (12, 11, 4, 8, 7)
colors = ('#8B4513', '#FFF8DC', '#93C572', '#E67F0D', '#D53032')
explode = (0, 0, 0, 0.1, 0)

plt.pie(
    votes,
    labels=flavors,
    autopct='%1.1f%%',
    colors=colors,
    explode=explode,
)
plt.show()
```



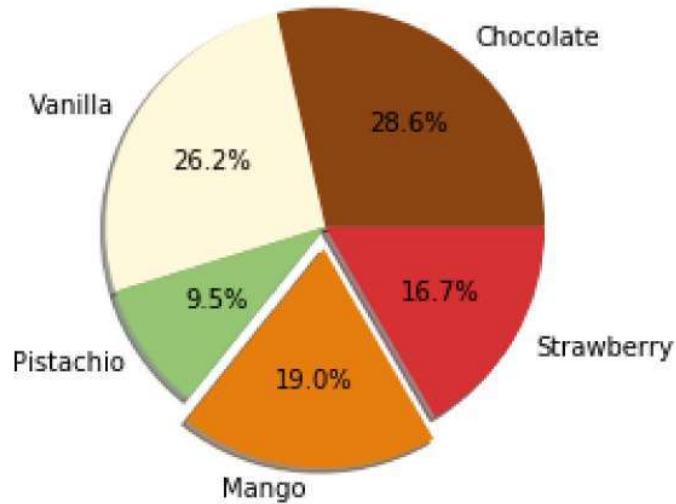
Gambar 0.4. Highlight item mango

Sekarang mangga sudah ditarik keluar sedikit dari Pie Chart, sehingga kita dapat melihat highlight dari data mangga (Gambar 0.4). Diagram lingkaran sudah terlihat cukup bagus, tetapi tampilanya sangat standard. Kita bisa memberinya sedikit tampilan tiga dimensi dengan menambahkan bayangan dengan argumen bayangan (Gambar 0.5).

```
import matplotlib.pyplot as plt
```

```
flavors = ('Chocolate', 'Vanilla', 'Pistachio', 'Mango',
'Strawberry')
votes = (12, 11, 4, 8, 7)
colors = ('#8B4513', '#FFF8DC', '#93C572', '#E67F0D', '#D53032')
explode = (0, 0, 0, 0.1, 0)

plt.pie(
    votes,
    labels=flavors,
    autopct='%1.1f%%',
    colors=colors,
    explode=explode,
    shadow=True
)
plt.show()
```



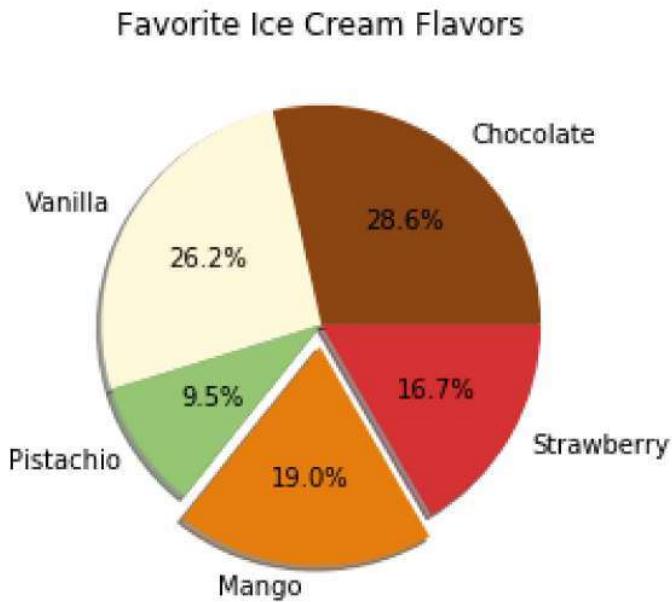
Gambar 0.5. Penambahan bayangan pada visualisasi

Untuk menyelesaiannya, kita bisa menambahkan judul menggunakan plt.title(). Perhatikan bahwa ini bukan argumen untuk plt.pie(), melainkan pemanggilan metode terpisah di plt.

```
import matplotlib.pyplot as plt

flavors = ('Chocolate', 'Vanilla', 'Pistachio', 'Mango',
           'Strawberry')
votes = (12, 11, 4, 8, 7)
colors = ('#8B4513', '#FFF8DC', '#93C572', '#E67F0D', '#D53032')
explode = (0, 0, 0, 0.1, 0)

plt.title('Favorite Ice Cream Flavors')
plt.pie(
    votes,
    labels=flavors,
    autopct='%1.1f%%',
    colors=colors,
    explode=explode,
    shadow=True
)
plt.show()
```



Gambar 0.6. Pie Chart Lengkap

Sekarang kita dapat memiliki Pie Chart (Gambar 0.6) yang menunjukkan semua rasa es krim favorit dalam sebuah survei! Ingat diagram lingkaran bagus untuk menunjukkan bagaimana distribusi kelas pada data yang berbeda (dalam hal ini, rasa es krim). Pie chart akan sangat efektif jika hanya ada beberapa kelas yang terwakili. Bayangkan jika kita memiliki 100 rasa es krim. Maka tampilan Pie Chart akan sangat penuh

Bar Charts

Bar Chart adalah merupakan tools visualisasi yang dapat digunakan untuk membandingkan data kategorikal. Mirip dengan diagram lingkaran, diagram ini dapat digunakan untuk membandingkan kategori data satu sama lain. Namun, diagram lingkaran sangat spesifik untuk melihat bagaimana satu kategori data dibandingkan dengan keseluruhan. Grafik diagram batang tidak terlalu tepat untuk hal tersebut. Selain itu, diagram batang dapat menampilkan lebih banyak kategori data daripada diagram lingkaran.

Mari kita mulai dengan melihat diagram batang yang menunjukkan populasi setiap negara di Amerika Selatan. Untuk melakukan ini kita akan menggunakan Matplotlib. Kali ini kita akan menggunakan method bar(). bar () memiliki dua argumen yang diperlukan. Argumen pertama berisi koordinat x dari data. Karena kita ingin memplot nama negara pada sumbu x. Dalam kasus ini kita dapat menggunakan fungsi arange () NumPy untuk membuat daftar angka yang memiliki jumlah array sama. Assignment angka antara 0 dan panjang data, yang seharusnya memberi daftar bilangan bulat mulai dari 0 dan berakhir pada len (data) -1, yaitu 13 dalam contoh kasus ini. Argumen berikutnya adalah data numerik untuk dipetakan. Dalam contoh ini kita memplot data populasi..

```
import matplotlib.pyplot as plt
import numpy as np
```

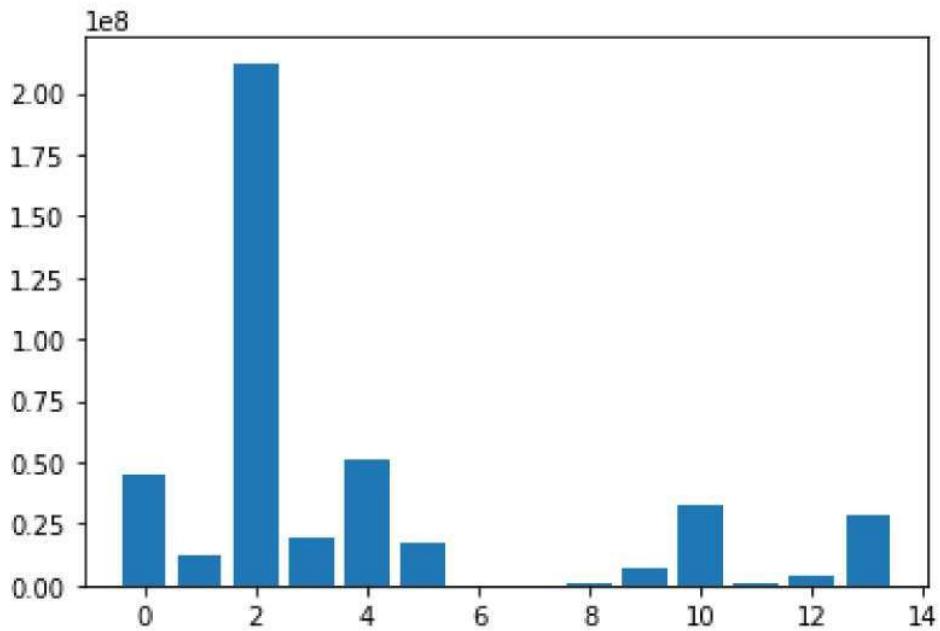
```

countries = ('Argentina', 'Bolivia', 'Brazil', 'Chile', 'Colombia',
'Ecuador',
    'Falkland Islands', 'French Guiana', 'Guyana',
'Paraguay', 'Peru',
    'Suriname', 'Uruguay', 'Venezuela')

populations = (45076704, 11626410, 212162757, 19109629, 50819826,
17579085,
    3481, 287750, 785409, 7107305, 32880332, 585169,
3470475,
    28258770)

x_coords = np.arange(len(countries))
plt.bar(x_coords, populations)
plt.show()

```



Gambar 0.7. Bar Charts Simple Visualisations

Anda dapat melihat pada Gambar 0.7 bahwa x-label tidak bermakna. Kita bisa memperbaiki ini dengan meneruskan argumen tick_label ke bar(). Karena kita memiliki label yang relatif lebar, akan berguna juga untuk memutar label sejauh 90 derajat agar lebih mudah dibaca. Lakukan panggilan metode plt.xticks (rotation = 90).

```

import matplotlib.pyplot as plt
import numpy as np

countries = ('Argentina', 'Bolivia', 'Brazil', 'Chile', 'Colombia',
'Ecuador',
    'Falkland Islands', 'French Guiana', 'Guyana',
'Paraguay', 'Peru',
    'Suriname', 'Uruguay', 'Venezuela')

```

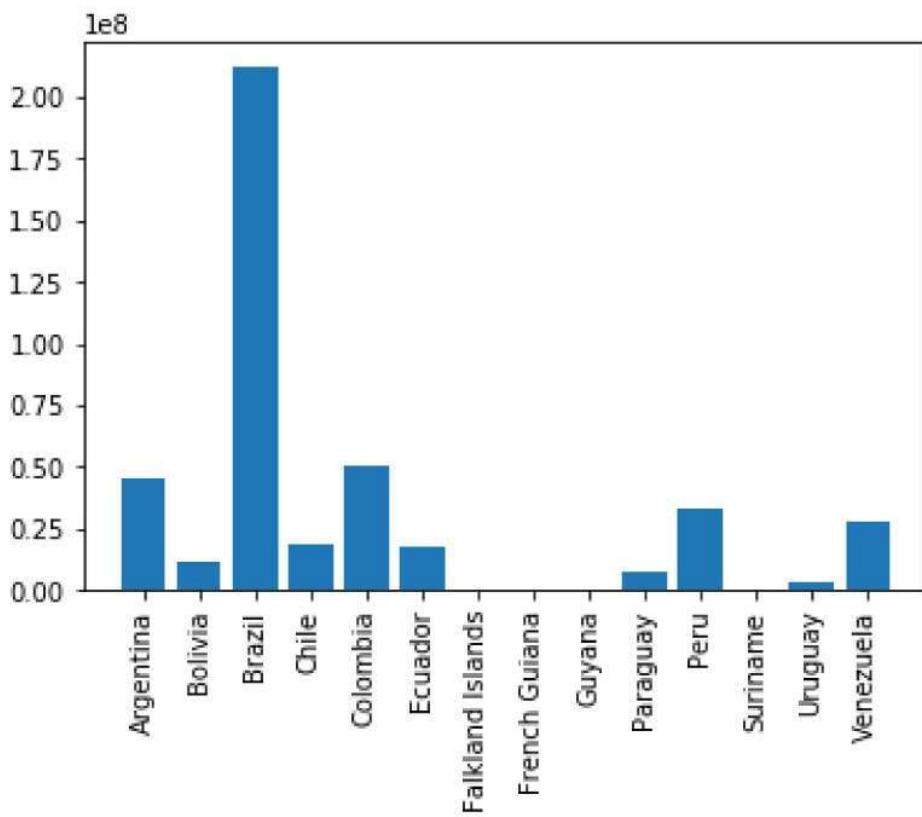
```

'Paraguay', 'Peru',
'Suriname', 'Uruguay', 'Venezuela')

populations = (45076704, 11626410, 212162757, 19109629, 50819826,
17579085,
3481, 287750, 785409, 7107305, 32880332, 585169,
3470475,
28258770)

x_coords = np.arange(len(countries))
plt.bar(x_coords, populations, tick_label=countries)
plt.xticks(rotation=90) #rotates text for x-axis labels
plt.show()

```



Gambar 0.8. Penambahan label pada bar chart

Kita dapat menambahkan label ke diagram batang (Gambar 0.8) untuk membantu membuat diagram agar lebih mudah dibaca. Pada contoh di bawah ini kita menambahkan label-y menggunakan metode ylabel () dan judul grafik menggunakan metode title () (Gambar 0.9).

```

import matplotlib.pyplot as plt
import numpy as np

countries = ('Argentina', 'Bolivia', 'Brazil', 'Chile', 'Colombia',

```

```

'Ecuador',
      'Falkland Islands', 'French Guiana', 'Guyana',
'Paraguay', 'Peru',
      'Suriname', 'Uruguay', 'Venezuela')

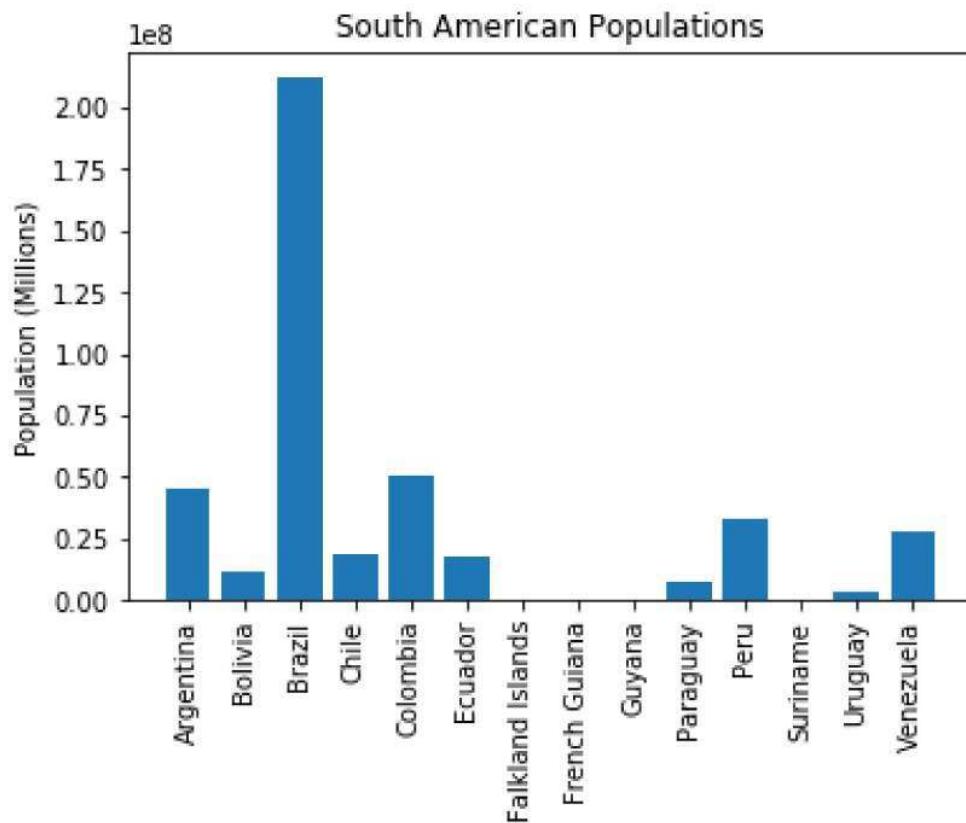
populations = (45076704, 11626410, 212162757, 19109629, 50819826,
17579085,
      3481, 287750, 785409, 7107305, 32880332, 585169,
3470475,
      28258770)

```

```

x_coords = np.arange(len(countries))
plt.bar(x_coords, populations, tick_label=countries)
plt.xticks(rotation=90)
plt.ylabel('Population (Millions)')
plt.title('South American Populations')
plt.show()

```



Gambar 0.9. Penambahan label dan title pada bar chart

Bagannya sudah terlihat cukup bagus. Tetapi bagaimana jika pertanyaan: Apa negara terpadat kedua di Amerika Selatan? Anda mungkin harus sedikit menatap Argentina dan Kolombia. Ini karena data diurutkan menurut abjad, yang bukan merupakan pengurutan

yang paling berguna untuk menjawab pertanyaan tentang data. Sayangnya Matplotlib tidak memiliki penyortiran bawaan. Sebagai gantinya, Anda dapat mengimpor Panda dan menggunakan untuk mengurutkan data (Gambar 0.10).

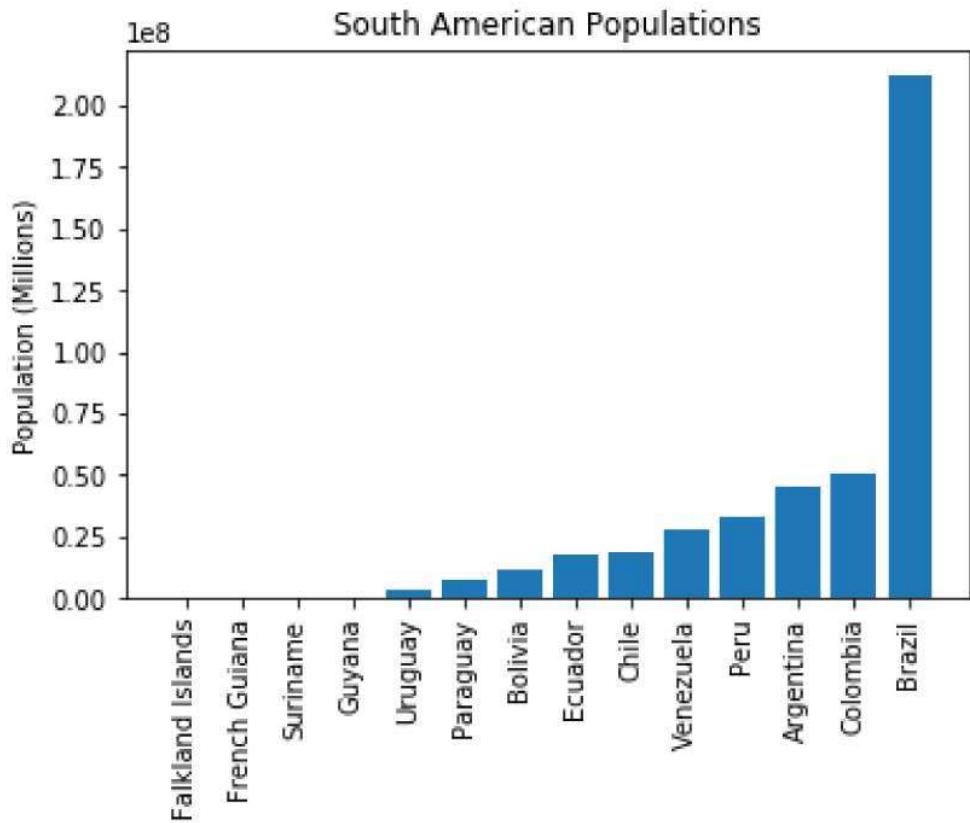
```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd

countries = ('Argentina', 'Bolivia', 'Brazil', 'Chile', 'Colombia',
'Ecuador',
    'Falkland Islands', 'French Guiana', 'Guyana',
'Paraguay', 'Peru',
    'Suriname', 'Uruguay', 'Venezuela')

populations = (45076704, 11626410, 212162757, 19109629, 50819826,
17579085,
    3481, 287750, 785409, 7107305, 32880332, 585169,
3470475,
    28258770)

df = pd.DataFrame({
    'Country': countries,
    'Population': populations,
})
df.sort_values(by='Population', inplace=True)

x_coords = np.arange(len(df))
plt.bar(x_coords, df['Population'], tick_label=df['Country'])
plt.xticks(rotation=90)
plt.ylabel('Population (Millions)')
plt.title('South American Populations')
plt.show()
```



Gambar 0.10. Pengurutan Jumlah Populasi pada Bar Chart

```
len(df)
```

```
14
```

Sekarang kita dapat dengan mudah melihat bahwa Kolombia adalah negara terbesar kedua (Gambar 0.11). Jika kita ingin memanggilnya, kita bisa meneruskan daftar warna bar ke metode bar () .

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd

countries = ('Argentina', 'Bolivia', 'Brazil', 'Chile', 'Colombia',
'Ecuador',
      'Falkland Islands', 'French Guiana', 'Guyana',
'Paraguay', 'Peru',
      'Suriname', 'Uruguay', 'Venezuela')

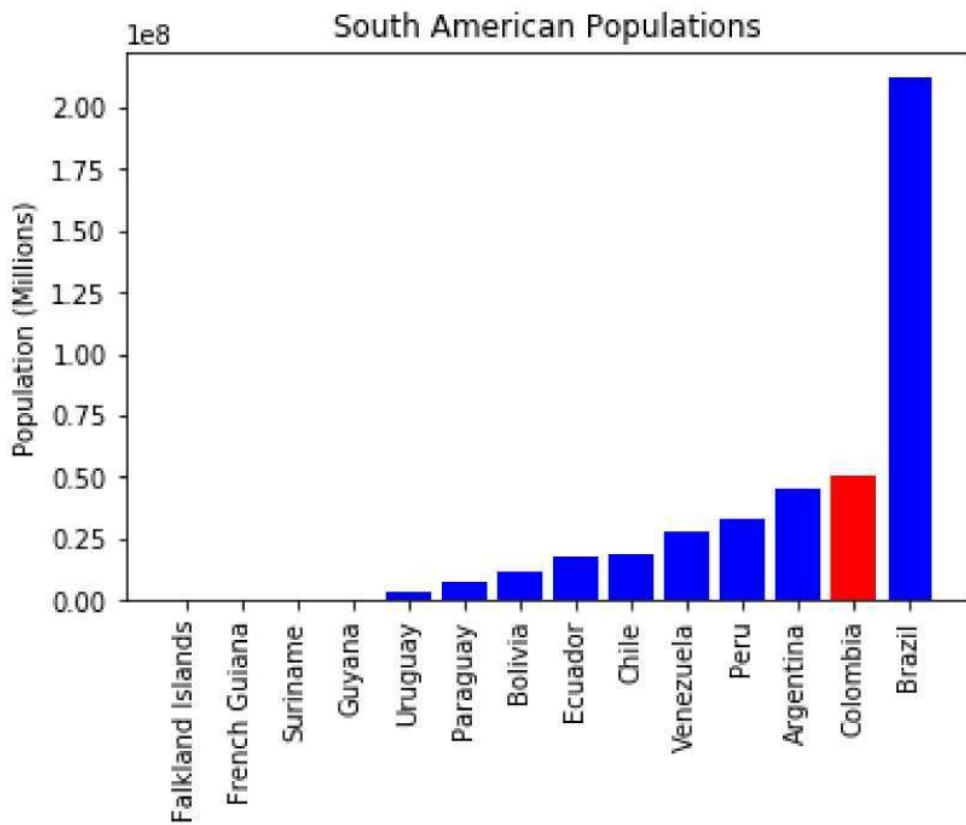
populations = (45076704, 11626410, 212162757, 19109629, 50819826,
17579085,
      3481, 287750, 785409, 7107305, 32880332, 585169,
3470475,
      28258770)
```

```

df = pd.DataFrame({
    'Country': countries,
    'Population': populations,
})
df.sort_values(by='Population', inplace=True)

x_coords = np.arange(len(df))
colors = ['#0000FF' for _ in range(len(df))]
colors[-2] = '#FF0000'
plt.bar(x_coords, df['Population'], tick_label=df['Country'],
color=colors)
plt.xticks(rotation=90)
plt.ylabel('Population (Millions)')
plt.title('South American Populations')
plt.show()

```



Gambar 0.11. Highlight Populasi Kolombia

```

colors
['#0000FF',
 '#0000FF',
 '#0000FF',
 '#0000FF',
 '#0000FF',

```

```
'#0000FF',
'#0000FF',
'#0000FF',
'#0000FF',
'#0000FF',
'#0000FF',
'#0000FF',
'#FF0000',
'#0000FF']
```

Kita juga bisa membuat grafik menjadi lebih besar menggunakan metode figure (). Kita dapat meneruskan argumen figsize = yang mewakili lebar dan tinggi gambar dalam inci (Gambar 0.12).

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd

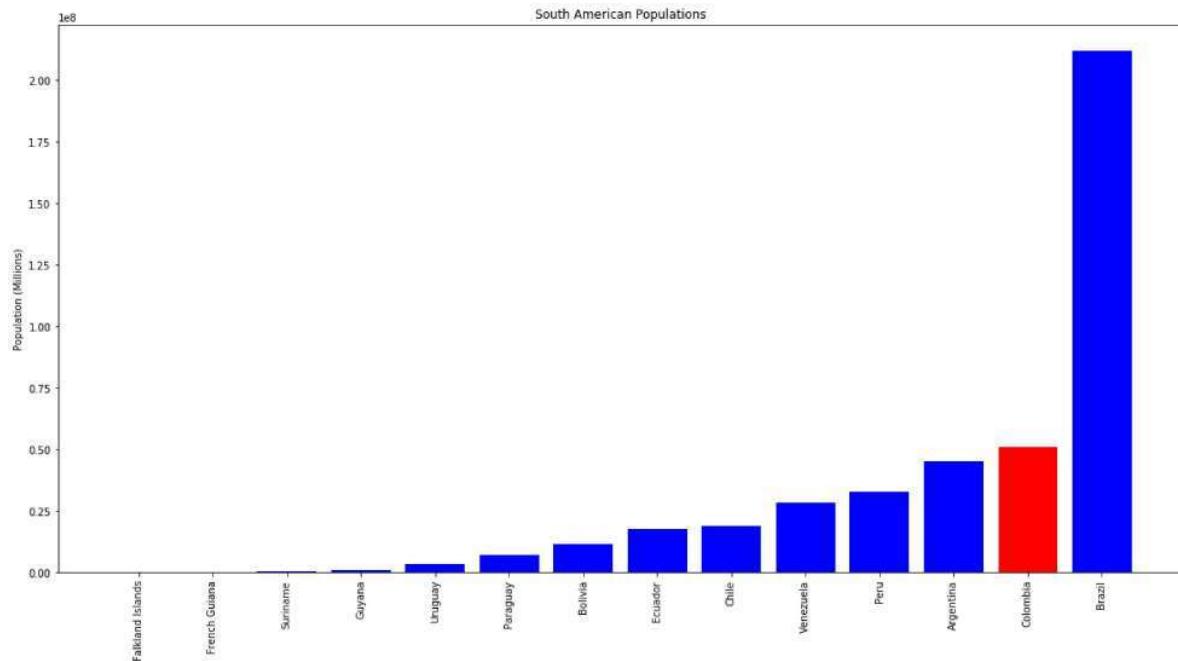
countries = ('Argentina', 'Bolivia', 'Brazil', 'Chile', 'Colombia',
'Ecuador',
      'Falkland Islands', 'French Guiana', 'Guyana',
'Paraguay', 'Peru',
      'Suriname', 'Uruguay', 'Venezuela')

populations = (45076704, 11626410, 212162757, 19109629, 50819826,
17579085,
            3481, 287750, 785409, 7107305, 32880332, 585169,
3470475,
            28258770)

df = pd.DataFrame({
    'Country': countries,
    'Population': populations,
})
df.sort_values(by='Population', inplace=True)

x_coords = np.arange(len(df))
colors = ['#0000FF' for _ in range(len(df))]
colors[-2] = '#FF0000'
plt.figure(figsize=(20,10))
plt.bar(x_coords, df['Population'], tick_label=df['Country'],
color=colors)
plt.xticks(rotation=90)
plt.ylabel('Population (Millions)')
```

```
plt.title('South American Populations')
plt.show()
```



Gambar 0.12. Contoh Hasil Bar Chart Lengkap

Line Graphs

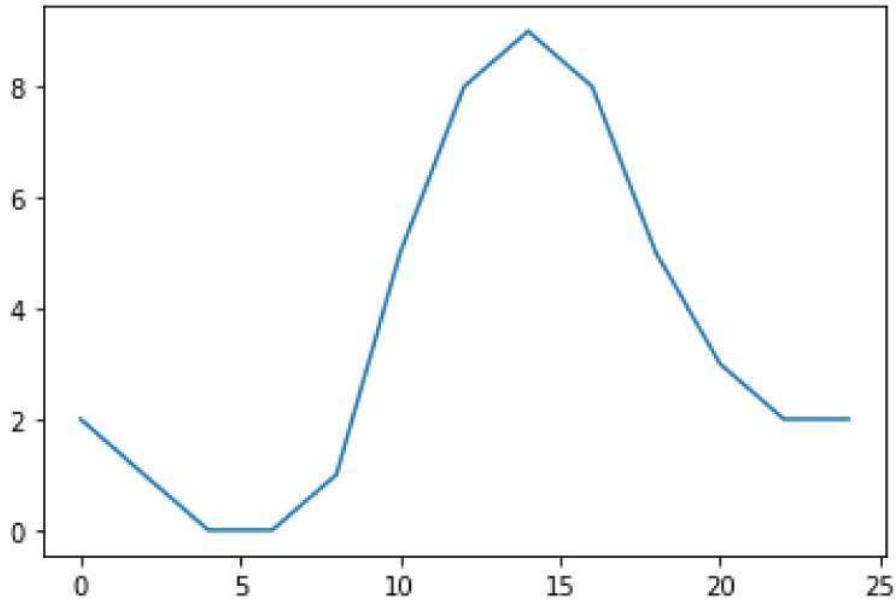
Line Graph adalah bentuk visualisasi lainnya selain diagram lingkaran dan diagram batan. Meskipun diagram lingkaran dan diagram batang berguna untuk menunjukkan bagaimana kelas data saling terkait, diagram garis lebih berguna untuk menunjukkan bagaimana kemajuan data selama beberapa periode. Misalnya, grafik garis dapat berguna dalam membuat grafik suhu dari waktu ke waktu, harga saham dari waktu ke waktu, berat menurut hari, atau metrik berkelanjutan lainnya.

Kita akan membuat grafik garis yang sangat sederhana di bawah ini. Data yang kita miliki adalah suhu dalam celsius dan jam dalam sehari untuk satu hari dan lokasi. Anda dapat melihat bahwa untuk membuat grafik garis kita menggunakan metode plt.plot().

```
import matplotlib.pyplot as plt
```

```
temperature_c = [2, 1, 0, 0, 1, 5, 8, 9, 8, 5, 3, 2, 2]
hour = [0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24]

plt.plot(
    hour,
    temperature_c
)
plt.show()
```



Gambar 0.13. Contoh line graph

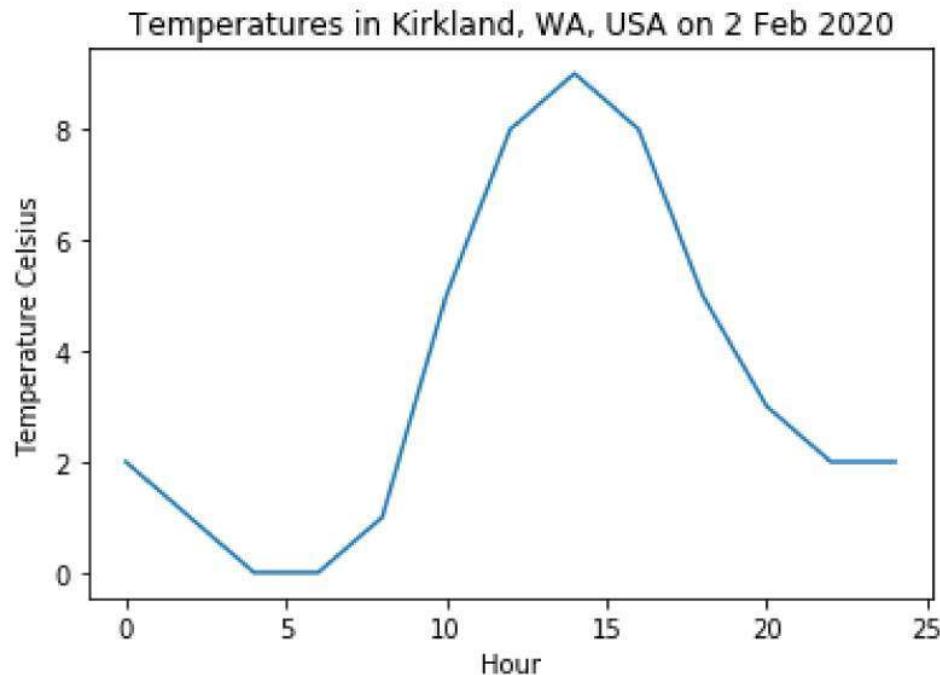
Kita dapat melihat bahwa suhu mulai sekitar 2 derajat celcius pada tengah malam, sedikit turun menjadi beku sekitar pukul 05:00, naik menjadi sekitar 9 derajat celcius pada pukul 15:00, dan kemudian turun kembali menjadi sekitar 2 derajat pada tengah malam (Gambar 0.13).

Kita juga bisa menambahkan elemen bagan standar dari title (), ylabel (), dan xlabel () (Gambar 0.14).

```
import matplotlib.pyplot as plt

temperature_c = [2, 1, 0, 0, 1, 5, 8, 9, 8, 5, 3, 2, 2]
hour = [0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24]

plt.plot(
    hour,
    temperature_c,
)
plt.title('Temperatures in Kirkland, WA, USA on 2 Feb 2020')
plt.ylabel('Temperature Celsius')
plt.xlabel('Hour')
plt.show()
```



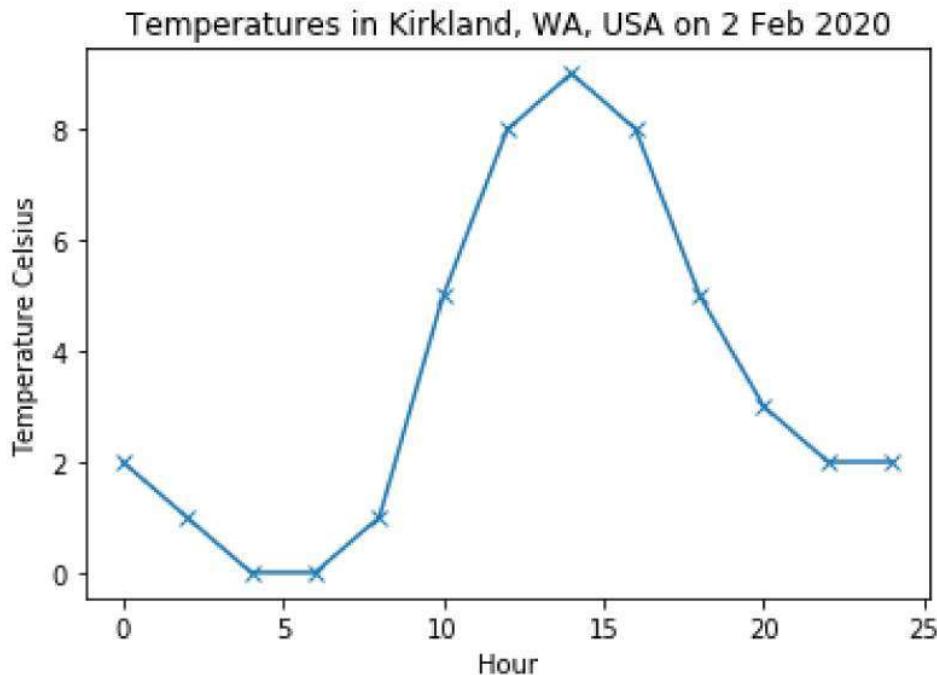
Gambar 0.14. Line Graph dengan Title dan Atribut

Kita juga dapat menambahkan penanda di setiap titik data (Gambar 0.15). Pada contoh di bawah ini kita menambahkan penanda titik pada setiap titik data menggunakan argumen marker = 'o'.

```
import matplotlib.pyplot as plt

temperature_c = [2, 1, 0, 0, 1, 5, 8, 9, 8, 5, 3, 2, 2]
hour = [0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24]

plt.plot(
    hour,
    temperature_c,
    marker='x',
)
plt.title('Temperatures in Kirkland, WA, USA on 2 Feb 2020')
plt.ylabel('Temperature Celsius')
plt.xlabel('Hour')
plt.show()
```



Gambar 0.15. Line Graph dengan Penanda disetiap Titik

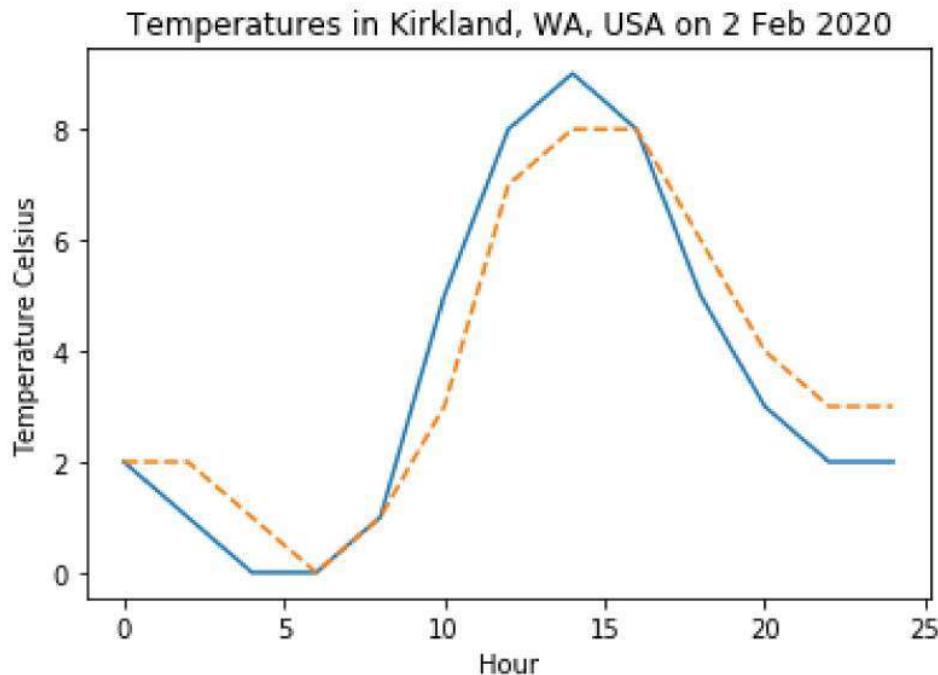
Kita bahkan dapat memiliki beberapa garis pada grafik yang sama (Gambar 0.16). Misalnya, misalnya, kita ingin mengilustrasikan nilai suhu aktual dan prediksi. Kita bisa memanggil `plot()` dua kali, sekali dengan setiap kumpulan nilai. Perhatikan bahwa dalam panggilan kedua, kita menggunakan argumen lain untuk `plot()`, `linestyle = '-'`. Hal ini menyebabkan garis prediksi terlihat seperti garis putus-putus sedangkan nilai sebenarnya tetap solid.

Anda dapat melihat dokumentasi tambahan pada [Matplotlib pyplot.plot\(\) documentation](#).

```
import matplotlib.pyplot as plt
```

```
temperature_c_actual = [2, 1, 0, 0, 1, 5, 8, 9, 8, 5, 3, 2, 2]
temperature_c_predicted = [2, 2, 1, 0, 1, 3, 7, 8, 8, 6, 4, 3, 3]
hour = [0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24]

plt.plot(hour, temperature_c_actual)
plt.plot(hour, temperature_c_predicted, linestyle='--')
plt.title('Temperatures in Kirkland, WA, USA on 2 Feb 2020')
plt.ylabel('Temperature Celsius')
plt.xlabel('Hour')
plt.show()
```



Gambar 0.16. Line Graph dengan Lebih dari satu garis

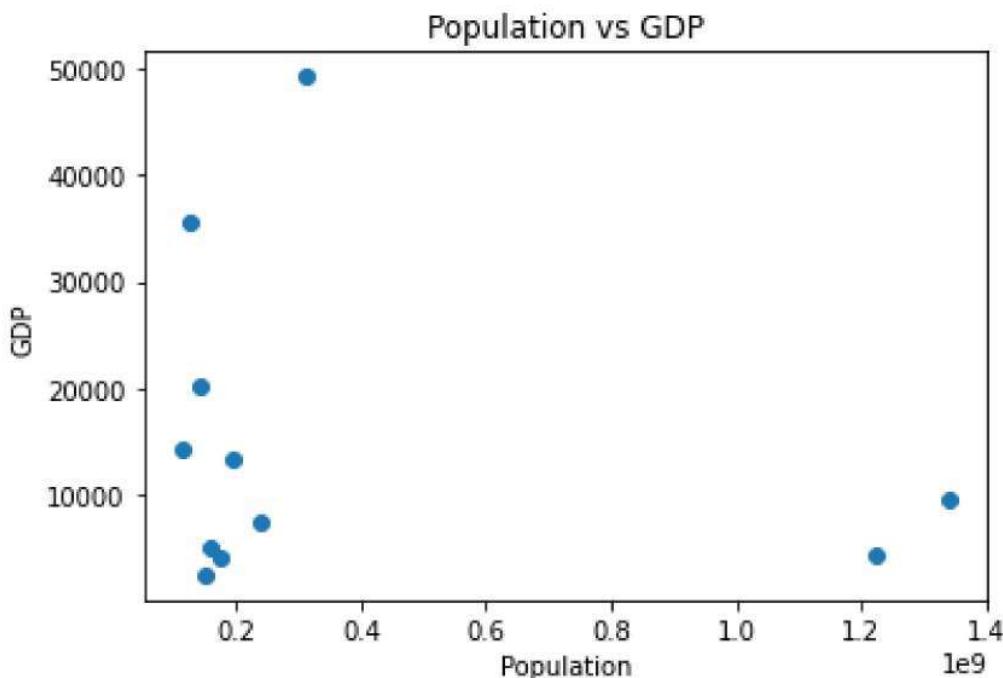
Scatter Plot

Scatter plot berfungsi baik untuk data dengan dua komponen numerik. Scatter plot dapat memberikan informasi yang berguna terutama mengenai pola atau penciran. Pada contoh di bawah ini, kita memiliki data yang terkait dengan produk domestik bruto (PDB) dan populasi untuk negara-negara dengan populasi lebih dari seratus juta. PDB adalah total nilai barang dan jasa yang dibuat / disediakan oleh suatu negara selama satu tahun. Kita kemudian menggunakan plt.scatter () untuk membuat sebaran populasi dan PDB (Gambar 0.17).

```
import matplotlib.pyplot as plt

country = ['Bangladesh', 'Brazil', 'China', 'India', 'Indonesia',
           'Japan',
           'Mexico', 'Nigeria', 'Pakistan', 'Russia', 'United
           States']
gdp = [2421, 13418, 9475, 4353, 7378, 35477, 14276, 5087, 4133,
       20255, 49267]
population = [148692131, 194946470, 1341335152, 1224614327,
              239870937,
              126535920, 113423047, 158423182, 173593383, 142958164,
              310383948]

plt.scatter(population, gdp)
plt.show()
```



Gambar 0.17. Scatter Plot data PDB dan populasi

Scatter plot sangat intuitif karena kita dapat mengumpulkan informasi tentang data kita. Kita dapat melihat bahwa ada dua pencikan populasi (pencikan PDB). Informasi ini dapat membantu kita memutuskan apakah kita perlu mengoreksi atau mengecualikan pencikan dalam analisis kita. Kita juga dapat menambahkan lebih dari satu kumpulan data ke plot (Gambar 0.18).

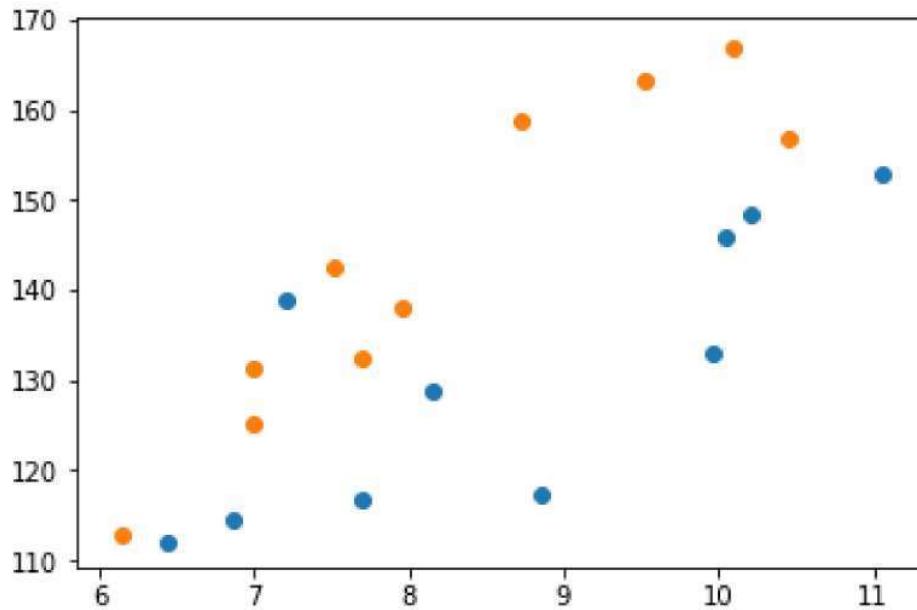
Pada contoh di bawah ini, kita memplot diameter dan berat sekumpulan lemon dan jeruk nipis agar dapat melihat apakah kita dapat menentukan polanya.

```
import matplotlib.pyplot as plt
```

```
lemon_diameter = [6.44, 6.87, 7.7, 8.85, 8.15, 9.96, 7.21, 10.04,
10.2, 11.06]
lemon_weight = [112.05, 114.58, 116.71, 117.4, 128.93,
132.93, 138.92, 145.98, 148.44, 152.81]

lime_diameter = [6.15, 7.0, 7.0, 7.69, 7.95, 7.51, 10.46, 8.72,
9.53, 10.09]
lime_weight = [112.76, 125.16, 131.36, 132.41, 138.08,
142.55, 156.86, 158.67, 163.28, 166.74]

plt.scatter(lemon_diameter, lemon_weight)
plt.scatter(lime_diameter, lime_weight)
plt.show()
```



Gambar 0.18. Perbandingan lemon dan lime

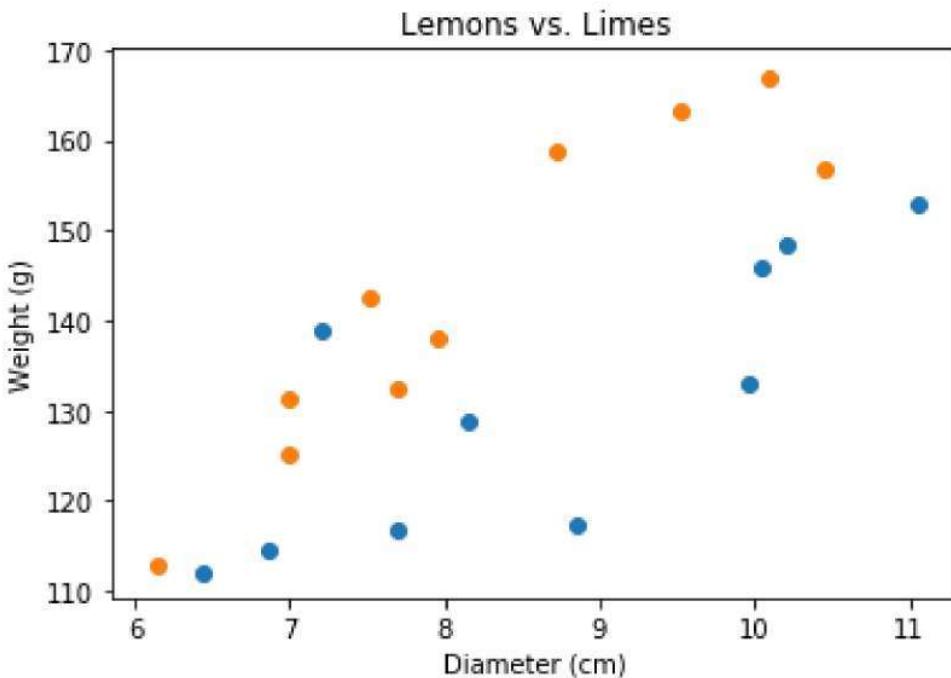
Melihat sampel kita, tidak ada pola yang sangat jelas. Namun, salah satu jenis jeruk tampaknya lebih berat dan diameternya lebih besar. Tapi yang mana? Mari kita bersihkan bagan ini sedikit. Pertama kita akan menambahkan judul menggunakan plt.title (), x-label menggunakan plt.xlabel (), dan y-label menggunakan plt.ylabel () (Gambar 0.19).

```
import matplotlib.pyplot as plt

lemon_diameter = [6.44, 6.87, 7.7, 8.85, 8.15, 9.96, 7.21, 10.04,
10.2, 11.06]
lemon_weight = [112.05, 114.58, 116.71, 117.4, 128.93,
132.93, 138.92, 145.98, 148.44, 152.81]

lime_diameter = [6.15, 7.0, 7.0, 7.69, 7.95, 7.51, 10.46, 8.72,
9.53, 10.09]
lime_weight = [112.76, 125.16, 131.36, 132.41, 138.08,
142.55, 156.86, 158.67, 163.28, 166.74]

plt.title('Lemons vs. Limes')
plt.xlabel('Diameter (cm)')
plt.ylabel('Weight (g)')
plt.scatter(lemon_diameter, lemon_weight)
plt.scatter(lime_diameter, lime_weight)
plt.show()
```



Gambar 0.19 Lemon vs lime dengan label

Sekarang kita dapat menambahkan beberapa warna dan legenda untuk membuat scatter plot sedikit lebih intuitif. Kita menambahkan warna dengan meneruskan color = argumen ke plt.scatter(). Dalam hal ini kita hanya mengatur titik lemon menjadi kuning menggunakan color = 'y' dan titik lime menjadi hijau menggunakan color = 'g'.

Untuk menambahkan legenda, kita panggil plt.legend() dan berikan daftar yang berisi label untuk setiap sebaran data (Gambar 0.20).

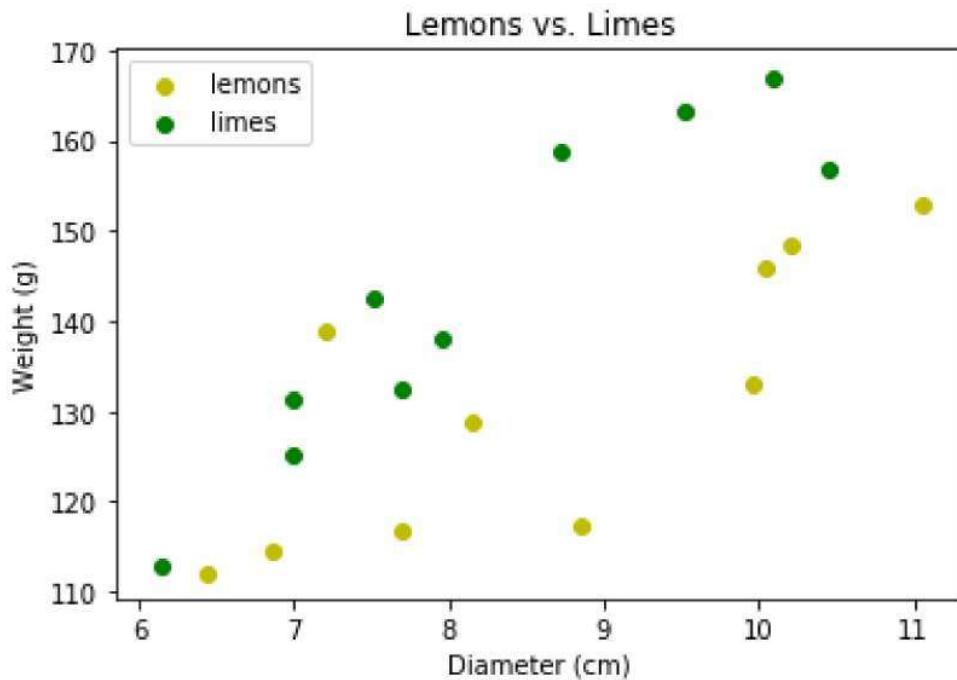
```
import matplotlib.pyplot as plt
```

```
lemon_diameter = [6.44, 6.87, 7.7, 8.85, 8.15, 9.96, 7.21, 10.04,
10.2, 11.06]
lemon_weight = [112.05, 114.58, 116.71, 117.4, 128.93,
132.93, 138.92, 145.98, 148.44, 152.81]

lime_diameter = [6.15, 7.0, 7.0, 7.69, 7.95, 7.51, 10.46, 8.72,
9.53, 10.09]
lime_weight = [112.76, 125.16, 131.36, 132.41, 138.08,
142.55, 156.86, 158.67, 163.28, 166.74]

plt.title('Lemons vs. Limes')
plt.xlabel('Diameter (cm)')
plt.ylabel('Weight (g)')
plt.scatter(lemon_diameter, lemon_weight, color='y')
plt.scatter(lime_diameter, lime_weight, color='g')
```

```
plt.legend(['lemons', 'limes'])
plt.show()
```



Gambar 0.20. Lemon vs lime perubahan warna

Sekarang kita dapat melihat lebih jelas bahwa jeruk nipis kita cenderung sedikit lebih berat dan diameternya lebih besar sedikit daripada lemon.

Heatmap

Heatmap adalah jenis visualisasi yang menggunakan kode warna untuk mewakili nilai / kepadatan relatif data di seluruh permukaan. Seringkali ini adalah bagan tabel, tetapi tidak harus terbatas pada itu. Untuk data tabular, terdapat label pada sumbu x dan y. Nilai di persimpangan label tersebut dipetakan ke warna. Warna-warna ini kemudian dapat digunakan untuk memeriksa data secara visual guna menemukan kelompok dengan nilai serupa dan mendeteksi tren dalam data.

Kita akan bekerja dengan data tentang temperatur rata-rata setiap bulan untuk 12 kota terbesar di dunia. Untuk membuat heatmap ini, kita akan menggunakan library Seaborn. Seaborn adalah library visualisasi yang dibangun di atas Matplotlib. Library ini menyediakan antarmuka tingkat yang lebih tinggi dan dapat membuat grafik yang lebih menarik. Langkah-langkah untuk melakukan visualisasi heatmap (Gambar 0.21) adalah sebagai berikut

1. Pada kode di bawah ini, pertama kita mengimpor seaborn.
2. Kita kemudian membuat daftar yang berisi nama 12 kota terbesar di dunia dan 12 bulan dalam setahun.
3. Selanjutnya kita menetapkan daftar-daftar ke variabel suhu. Setiap baris dalam daftar mewakili sebuah kota.

4. Setiap kolom adalah satu bulan. Nilai-nilai tersebut adalah suhu tinggi rata-rata untuk kota selama bulan tersebut.
5. Pada tahap akhir kita memanggil sns.heatmap () untuk membuat peta panas. Kita mengirimkan data suhu, nama kota sebagai label-y, dan singkatan bulan sebagai label-x.

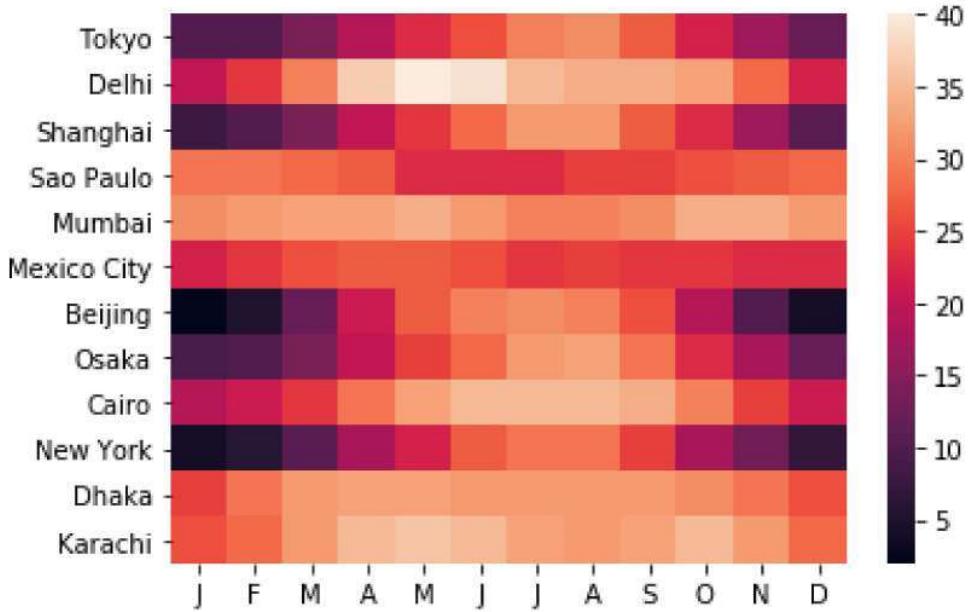
```
import seaborn as sns

cities = ['Tokyo', 'Delhi', 'Shanghai', 'Sao Paulo', 'Mumbai',
'Mexico City',
'Beijing', 'Osaka', 'Cairo', 'New York', 'Dhaka',
'Karachi']

months = ['J', 'F', 'M', 'A', 'M', 'J', 'J', 'A', 'S', 'O', 'N',
'D']

temperatures = [
[10, 10, 14, 19, 23, 26, 30, 31, 27, 22, 17, 12], # Tokyo
[20, 24, 30, 37, 40, 39, 35, 34, 34, 33, 28, 22], # Delhi
[8, 10, 14, 20, 24, 28, 32, 32, 27, 23, 17, 11], # Shanghai
[29, 29, 28, 27, 23, 23, 23, 25, 25, 26, 27, 28], # Sao Paulo
[31, 32, 33, 33, 34, 32, 30, 30, 31, 34, 34, 32], # Mumbai
[22, 24, 26, 27, 27, 26, 24, 25, 24, 24, 23, 23], # Mexico City
[2, 5, 12, 21, 27, 30, 31, 30, 26, 19, 10, 4], # Beijing
[9, 10, 14, 20, 25, 28, 32, 33, 29, 23, 18, 12], # Osaka
[19, 21, 24, 29, 33, 35, 35, 35, 34, 30, 25, 21], # Cairo
[4, 6, 11, 18, 22, 27, 29, 29, 25, 18, 13, 7], # New York
[25, 29, 32, 33, 33, 32, 32, 32, 31, 29, 26], # Dhaka
[26, 28, 32, 35, 36, 35, 33, 32, 33, 35, 32, 28], # Karachi
]

sns.heatmap(temperatures, yticklabels=cities, xticklabels=months)
<matplotlib.axes._subplots.AxesSubplot at 0x22343c81988>
```



Gambar 0.21. Heatmtp mengenai temperatur di masing-masing kota selama 12 bulan

Kita bisa melihat data di grafik yang dihasilkan. Tapi bagaimana kita menafsirkannya? Sebenarnya cukup sulit untuk memahami data. Bagian kiri dan kanan grafik mungkin berisi warna yang lebih gelap, yang memetakan ke suhu yang lebih dingin, tetapi itu pun sulit untuk ditentukan. Kita perlu mengurutkan secara manual kota-kota tersebut, dari yang terkecil ke yang terbesar. Mari kita coba ubah pengurutan berdasarkan lintang pada garis bumi (Gambar 0.22).

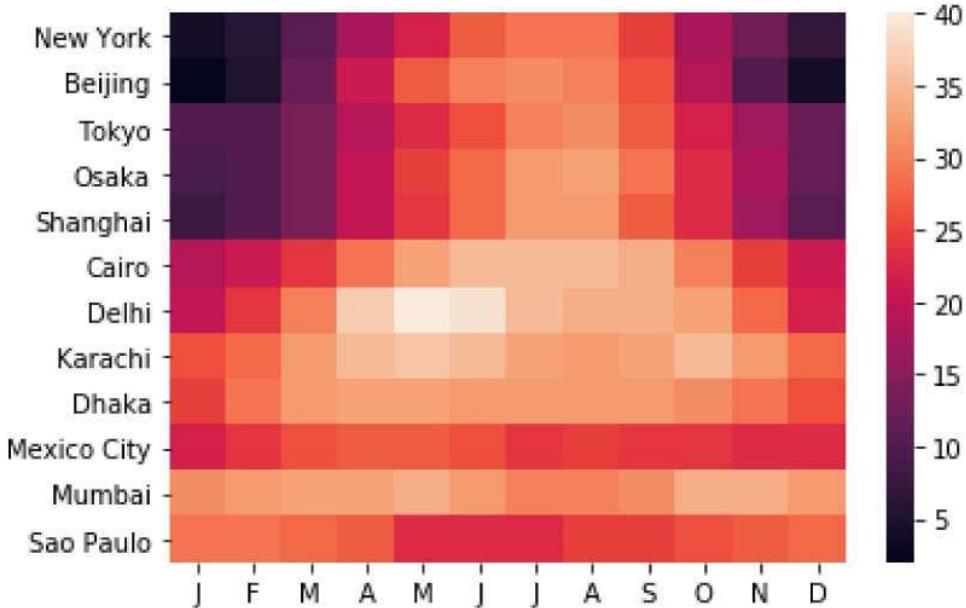
```
import seaborn as sns
```

```
cities = ['New York', 'Beijing', 'Tokyo', 'Osaka', 'Shanghai',
'Cairo', 'Delhi',
'Karachi', 'Dhaka', 'Mexico City', 'Mumbai', 'Sao Paulo']

temperatures = [
[ 4,  6, 11, 18, 22, 27, 29, 29, 25, 18, 13,  7], # New York
[ 2,  5, 12, 21, 27, 30, 31, 30, 26, 19, 10,  4], # Beijing
[10, 10, 14, 19, 23, 26, 30, 31, 27, 22, 17, 12], # Tokyo
[ 9, 10, 14, 20, 25, 28, 32, 33, 29, 23, 18, 12], # Osaka
[ 8, 10, 14, 20, 24, 28, 32, 32, 27, 23, 17, 11], # Shanghai
[19, 21, 24, 29, 33, 35, 35, 35, 34, 30, 25, 21], # Cairo
[20, 24, 30, 37, 40, 39, 35, 34, 34, 33, 28, 22], # Delhi
[26, 28, 32, 35, 36, 35, 33, 32, 33, 35, 32, 28], # Karachi
[25, 29, 32, 33, 33, 32, 32, 32, 32, 31, 29, 26], # Dhaka
[22, 24, 26, 27, 27, 26, 24, 25, 24, 24, 23, 23], # Mexico City
[31, 32, 33, 33, 34, 32, 30, 30, 31, 34, 34, 32], # Mumbai
```

```
[29, 29, 28, 27, 23, 23, 23, 25, 25, 26, 27, 28], # Sao Paulo  
]
```

```
sns.heatmap(temperatures, yticklabels=cities, xticklabels=months)  
<matplotlib.axes._subplots.AxesSubplot at 0x22345cc0a48>
```



Gambar 0.22. Heatmap temperature negara berdasarkan garis bumi (lintang)

Kita dapat melihat bahwa kota-kota di garis lintang yang lebih tinggi, lebih dingin dari bulan September hingga Maret dan suhu cenderung meningkat seiring dengan semakin mengecilnya garis lintang.

Perhatikan juga bahwa Sao Paulo terlihat lebih hangat di tengah tahun meskipun berada di belahan bumi selatan. Memang, skema warnanya sulit dibaca. Anda dapat mengubah skema warna menggunakan argumen `cmap = .cmap` = menerima daftar warna dan skema warna preset (Gambar 0.23). Anda dapat menemukan skema di [Matplotlib colormap documentation](#).

```
import seaborn as sns
```

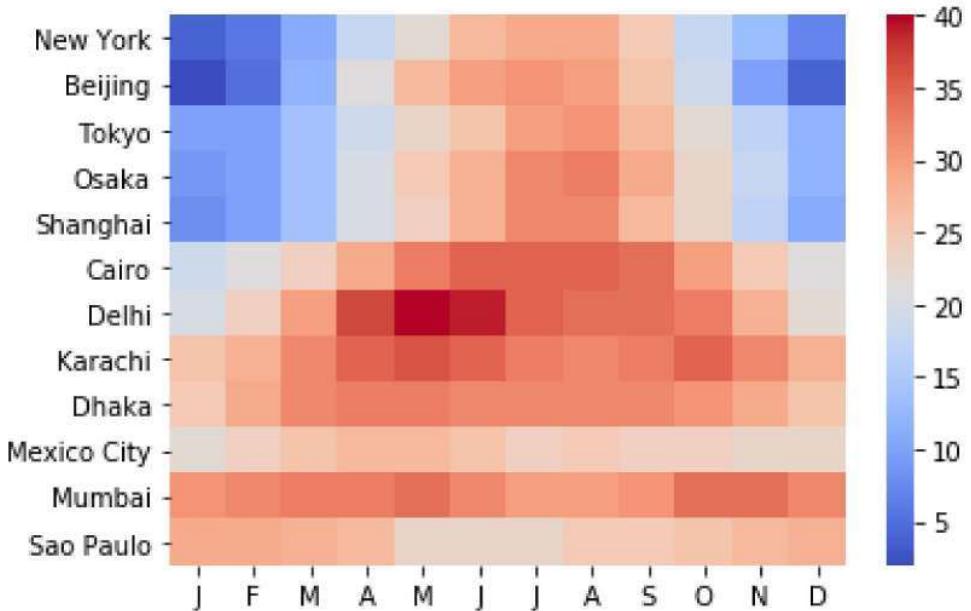
```
cities = ['New York', 'Beijing', 'Tokyo', 'Osaka', 'Shanghai',  
'Cairo', 'Delhi',  
         'Karachi', 'Dhaka', 'Mexico City', 'Mumbai', 'Sao Paulo']  
  
temperatures = [  
    [ 4,  6, 11, 18, 22, 27, 29, 29, 25, 18, 13,  7], # New York  
    [ 2,  5, 12, 21, 27, 30, 31, 30, 26, 19, 10,  4], # Beijing  
    [10, 10, 14, 19, 23, 26, 30, 31, 27, 22, 17, 12], # Tokyo  
    [ 9, 10, 14, 20, 25, 28, 32, 33, 29, 23, 18, 12], # Osaka
```

```

[ 8, 10, 14, 20, 24, 28, 32, 32, 27, 23, 17, 11], # Shanghai
[19, 21, 24, 29, 33, 35, 35, 35, 34, 30, 25, 21], # Cairo
[20, 24, 30, 37, 40, 39, 35, 34, 34, 33, 28, 22], # Delhi
[26, 28, 32, 35, 36, 35, 33, 32, 33, 35, 32, 28], # Karachi
[25, 29, 32, 33, 33, 32, 32, 32, 31, 29, 26], # Dhaka
[22, 24, 26, 27, 27, 26, 24, 25, 24, 24, 23, 23], # Mexico City
[31, 32, 33, 33, 34, 32, 30, 30, 31, 34, 34, 32], # Mumbai
[29, 29, 28, 27, 23, 23, 23, 25, 25, 26, 27, 28], # Sao Paulo
]

sns.heatmap(
    temperatures,
    yticklabels=cities,
    xticklabels=months,
    cmap='coolwarm',
)
<matplotlib.axes._subplots.AxesSubplot at 0x22345d9a3c8>

```



Gambar 0.23. Perubahan warna heatmap colormap

Visualisasi Statistik

Histogram

Histogram adalah salah satu visualisasi yang cukup penting dalam memahami distribusi pada data kita. Pandas Histogram menyediakan method yang memudahkan kita untuk membuat histogram. Plot histogram secara tradisional hanya membutuhkan satu dimensi data. Ini dimaksudkan untuk menunjukkan jumlah nilai atau kumpulan nilai secara serial. Pandas DataFrame.hist() akan mengambil DataFrame kita dan menampilkan plot histogram yang menunjukkan distribusi nilai dalam satu seri. Untuk membuat histogram

di panda, yang perlu kita lakukan adalah memberi tahu panda kolom mana yang ingin kita berikan datanya. Dalam hal ini, saya akan memberi tahu panda bahwa saya ingin melihat distribusi harga (histogram).

Parameter lain didalam histogram yang cukup menentukan banyaknya bar didalam visualisasi data kita adalah bin. Cara mudah untuk memikirkan bin adalah "berapa banyak batang yang Anda inginkan dalam bar chart?" Semakin banyak tempat sampah, semakin tinggi resolusi data Anda. Jika kita melakukan setting 2 bin sepertinya tidak terlalu memberikan informasi yang cukup, namun jika kita set menjadi 200 juga terlalu banyak. Kita bisa set sekitar 20-30 agar tampilan seimbang.

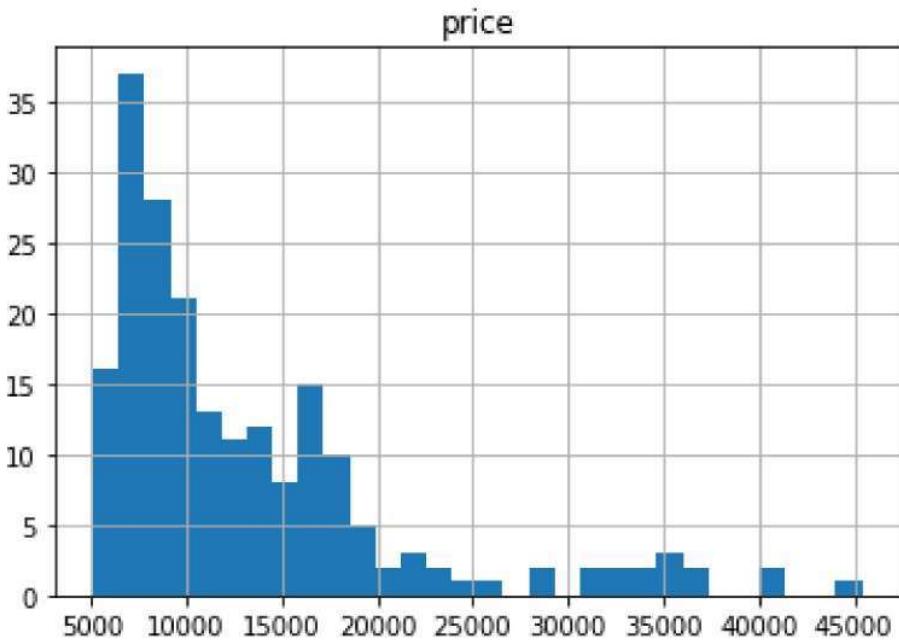
Sebagai contoh studi kasus data yang kita gunakan dalam modul ini adalah automobile.csv yang merupakan data-data spesifikasi kendaraan dari berbagai merek dan harganya. Overview data automobile dapat dilihat pada Gambar 0.24. Histogram pada data automobile dapat kita lihat pada Gambar 0.25, histogram tersebut dibentuk dengan memanggil fungsi hist().

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
path='https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-
data/CognitiveClass/DA0101EN/automobileEDA.csv'
df = pd.read_csv(path)
df.head()
```

	symboling	normalized-losses	make	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	length	width	height	curb-weight	engine-type	num-of-cylinders	engine-size	fuel-system	bore	stroke	compre
0	3	122	alfa-romero	std	two	convertible	rwd	front	88.6	0.811148	0.890278	48.8	2548	dohc	four	130	mpfi	3.47	2.68	
1	3	122	alfa-romero	std	two	convertible	rwd	front	88.6	0.811148	0.890278	48.8	2548	dohc	four	130	mpfi	3.47	2.68	
2	1	122	alfa-romero	std	two	hatchback	rwd	front	94.5	0.822681	0.909722	52.4	2823	ohcv	six	152	mpfi	2.68	3.47	
3	2	164	audi	std	four	sedan	fwd	front	99.8	0.848630	0.919444	54.3	2337	ohc	four	109	mpfi	3.19	3.40	
4	2	164	audi	std	four	sedan	4wd	front	99.4	0.848630	0.922222	54.3	2824	ohc	five	136	mpfi	3.19	3.40	

Gambar 0.24. Dataset Mobil

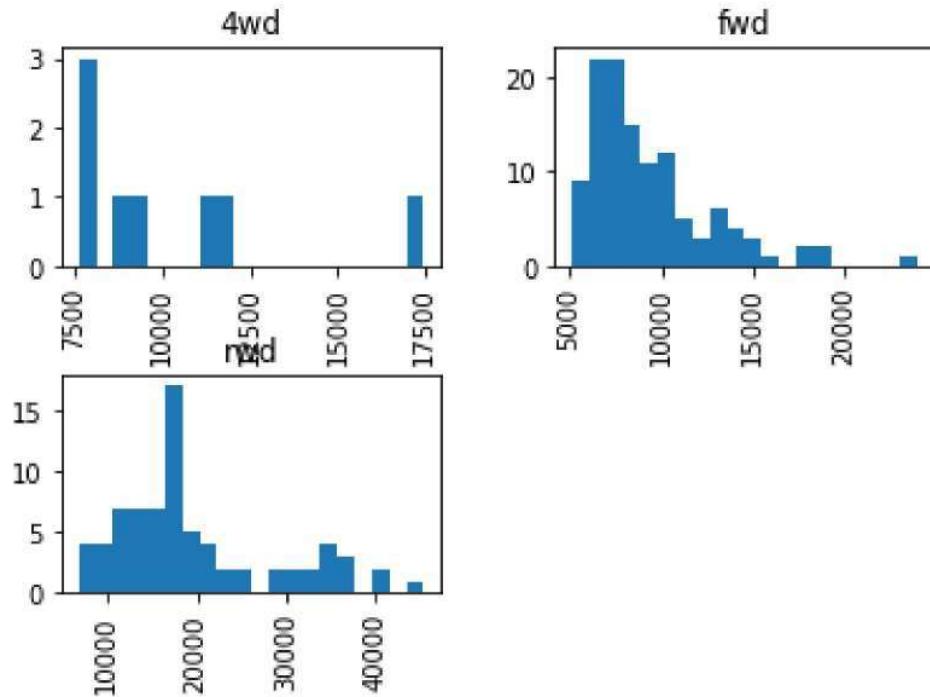
```
df.hist(column='price', bins=30);
```



Gambar 0.25. Contoh Histogram

Kita juga dapat memplot beberapa grup secara berdampingan. Di sini saya ingin melihat dua histogram, histogram price akan dikelompokkan berdasarkan roda penggerak dari kendaraan (fwd – berpenggerak roda depan, 4wd – berpenggerak 4 roda, atau rwd – penggerak belakang (Gambar 0.26).

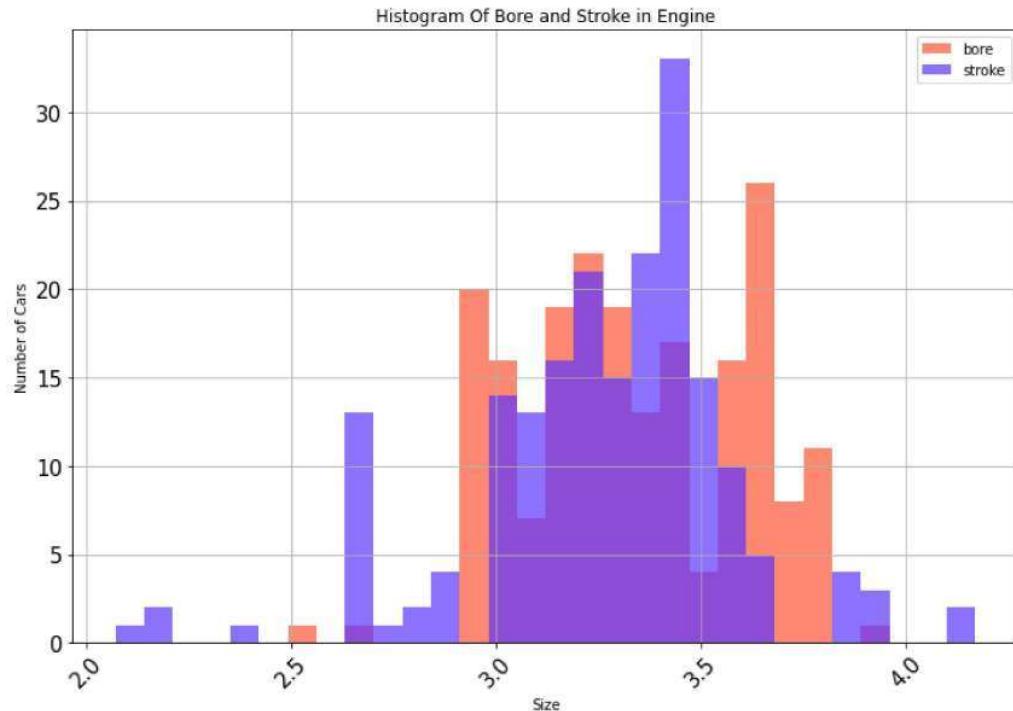
```
df.hist(column='price', by='drive-wheels', bins=20);
```



Gambar 0.26. Histogram untuk masing-masing kategori

Untuk memplot beberapa seri, kita bisa menggunakan metode df.plot(kind='hist') (Gambar 0.27).

```
df[['bore', 'stroke']].plot(kind='hist',
    alpha=0.7,
    bins=30,
    title='Histogram Of Bore and Stroke in Engine',
    rot=45,
    grid=True,
    figsize=(12,8),
    fontsize=15,
    color=['#FF5733', '#5C33FF'])
plt.xlabel('Size')
plt.ylabel("Number of Cars");
```



Gambar 0.27. Penggabungan histogram dalam satu visualisasi

Correlation and Causation

Korelasi merupakan suatu pengukuran sejauh mana nilai saling ketergantungan antar variabel. Causation merupakan hubungan antara sebab dan akibat antara dua variable. Penting untuk mengetahui perbedaan antara keduanya dan bahwa korelasi tidak mendeskripsikan sebab-akibat. Menentukan korelasi jauh lebih sederhana menentukan sebab memerlukan analisis lebih lanjut.

Korelasi Pearson

Korelasi Pearson mengukur ketergantungan linier antara dua variabel X dan Y. Koefisien yang dihasilkan adalah nilai antara -1 dan 1 inklusif, di mana:

- 1: Total korelasi linier positif.
- 0 : Tidak ada korelasi linier, kedua variabel kemungkinan besar tidak saling mempengaruhi.

- -1: Total korelasi linier negatif.

Pearson Correlation adalah metode default dari fungsi "corr". Seperti sebelumnya kita dapat menghitung Korelasi Pearson dari variabel 'int64' atau 'float64'. Terkadang kita ingin mengetahui signifikansi dari estimasi korelasi, kita dapat menggunakan p-value.

P-Value:

Berapa nilai P ini? Nilai P adalah nilai probabilitas bahwa korelasi antara kedua variabel ini signifikan secara statistik. Biasanya, kita memilih tingkat signifikansi 0,05, yang berarti bahwa kami yakin bahwa 95% korelasi antar variabel signifikan.

Dengan konvensi, ketika

- nilai p adalah $< 0,001$: kami katakan ada bukti kuat bahwa korelasinya signifikan.
- nilai p adalah $< 0,05$: terdapat bukti moderat bahwa korelasi tersebut signifikan.
- nilai p adalah $< 0,1$: ada bukti lemah bahwa korelasinya signifikan.
- nilai p adalah $> 0,1$: tidak ada bukti bahwa korelasi tersebut signifikan.

Kita dapat menggunakan library scipy untuk menghitung korelasi dan p-value

```
from scipy import stats
```

Mari kita hitung Koefisien Korelasi Pearson dan nilai-P dari 'wheel-base' dan 'price'.

```
pearson_coef, p_value = stats.pearsonr(df['wheel-base'], df['price'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a
P-value of P =", p_value)
The Pearson Correlation Coefficient is 0.584641822265508 with a P-value of P = 8.076488270733218e-20
```

Karena nilai p adalah $< 0,001$, korelasi antara wheel-base dan harga signifikan secara statistik, meskipun hubungan liniernya tidak terlalu kuat (0,588)

Mari kita hitung Koefisien Korelasi Pearson dan nilai-P dari 'horsepower' dan 'harga'.

```
pearson_coef, p_value = stats.pearsonr(df['horsepower'], df['price'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a
P-value of P =", p_value)
The Pearson Correlation Coefficient is 0.8095745670036559 with a P-value of P = 6.369057428260101e-48
```

Karena nilai p adalah $< 0,001$, korelasi antara horsepower dan harga signifikan secara statistik, dengan korelasi linear positif yang cukup kuat (~0,805)

Saat memvisualisasikan variabel individual, penting untuk terlebih dahulu memahami jenis variabel apa yang Anda hadapi (Gambar 0.28). Hal ini akan membantu kita menemukan metode visualisasi yang tepat untuk variabel tersebut.

```
# list the data types for each column
print(df.dtypes)
```

```

    ⚡ symboling           int64
normalized-losses      int64
make                   object
aspiration            object
num-of-doors          object
body-style             object
drive-wheels          object
engine-location        object
wheel-base             float64
length                 float64
width                  float64
height                 float64
curb-weight            int64
engine-type            object
num-of-cylinders       object
engine-size             int64
fuel-system            object
bore                   float64
stroke                 float64
compression-ratio      float64
horsepower              float64
peak-rpm                float64
city-mpg                int64
highway-mpg              int64
price                  float64
city-L/100km            float64
horsepower-binned       object
diesel                  int64
gas                     int64
dtype: object

```

Gambar 0.28. Type data

misalnya, kita dapat menghitung korelasi antara variabel bertipe "int64" atau "float64" menggunakan method "corr" (Gambar 0.29):

	symboling	normalized-losses	wheel-base	length	width	height	curb-weight	engine-size	bore	stroke	compression-ratio	horsepower	peak-rpm	city-mpg	highway-mpg	price
symboling	1.000000	0.466264	-0.535987	-0.365404	-0.242423	-0.550160	-0.233118	-0.110581	-0.140019	-0.008245	-0.182196	0.075819	0.279740	-0.035527	0.036233	-0.082391
normalized-losses	0.466264	1.000000	-0.056661	0.019424	0.086802	-0.373737	0.099404	0.112360	-0.029862	0.055563	-0.114713	0.217299	0.239543	-0.225016	-0.181877	0.133999
wheel-base	-0.535987	-0.056661	1.000000	0.876024	0.814507	0.590742	0.782097	0.572027	0.493244	0.158502	0.250313	0.371147	-0.360305	-0.470606	-0.543304	0.584542
length	-0.365404	0.019424	0.876024	1.000000	0.857170	0.492063	0.880665	0.685025	0.608971	0.124139	0.159733	0.579821	-0.285970	-0.665192	-0.698142	0.690628
width	-0.242423	0.086802	0.814507	0.857170	1.000000	0.316002	0.866201	0.729436	0.544885	0.188829	0.189867	0.615077	-0.245680	-0.633531	-0.680635	0.751265
height	-0.550160	-0.373737	0.590742	0.492063	0.306002	1.000000	0.307581	0.074694	0.180449	-0.062704	0.259737	-0.080727	-0.309974	-0.049800	-0.104812	0.135486
curb-weight	-0.233118	0.099404	0.782097	0.880665	0.866201	0.307581	1.000000	0.849072	0.844060	0.167562	0.156433	0.757976	-0.279361	-0.749543	-0.794889	0.834415
engine-size	-0.110581	0.112360	0.572027	0.685025	0.729436	0.074694	0.849072	1.000000	0.572609	0.209523	0.028889	0.822576	-0.256733	-0.650546	-0.679571	0.872335
bore	-0.140019	-0.029862	0.493244	0.608971	0.544885	0.180449	0.644060	0.572609	1.000000	-0.055390	0.001263	0.566936	-0.257392	-0.582027	-0.591309	0.543155
stroke	-0.008245	0.055563	0.158502	0.124139	0.188829	-0.062704	0.167562	0.209523	-0.055390	1.000000	0.187923	0.098462	-0.065713	-0.034696	-0.035201	0.082310
compression-ratio	-0.182196	-0.114713	0.250313	0.159733	0.189867	0.259737	0.156433	0.028889	0.001263	0.187923	1.000000	-0.214514	-0.435780	0.331425	0.268465	0.071107
horsepower	0.075819	0.217299	0.371147	0.579821	0.615077	-0.087027	0.757976	0.822676	0.566936	0.098462	-0.214514	1.000000	0.107885	-0.822214	-0.804575	0.809575
peak-rpm	0.279740	0.239543	-0.360305	-0.285970	-0.245680	-0.309974	-0.279361	-0.267392	-0.065713	-0.435780	0.107885	1.000000	-0.115413	-0.058598	-0.101616	
city-mpg	-0.035527	-0.225016	-0.470806	-0.665192	-0.633531	-0.049800	-0.740543	-0.650546	-0.582027	-0.034696	0.331425	-0.822214	-0.115413	1.000000	0.972044	-0.886571
highway-mpg	0.036233	-0.181877	-0.543304	-0.698142	-0.680635	-0.104812	-0.794889	-0.679571	-0.591309	-0.035201	0.268465	-0.804575	-0.058598	0.972044	1.000000	-0.704692
price	-0.082391	0.133999	0.584542	0.690628	0.751265	0.135486	0.834415	0.872335	0.543155	0.082310	0.071107	0.809575	-0.101616	-0.686571	-0.704692	1.000000
city-L/100km	0.066171	0.238567	0.476153	0.657373	0.673363	0.003811	0.785253	0.745059	0.554610	0.037300	-0.299372	0.889488	0.115830	-0.949713	-0.930028	0.789898
diesel	-0.196735	-0.101546	0.307287	0.211187	0.244356	0.281578	0.221046	0.070779	0.054458	0.241303	0.985231	-0.169053	-0.475812	0.265676	0.198690	0.110326
gas	0.196735	-0.101546	-0.307287	-0.211187	-0.244356	-0.281578	-0.221046	-0.070779	-0.054458	-0.241303	-0.985231	0.169053	0.475812	-0.265676	-0.198690	-0.110326

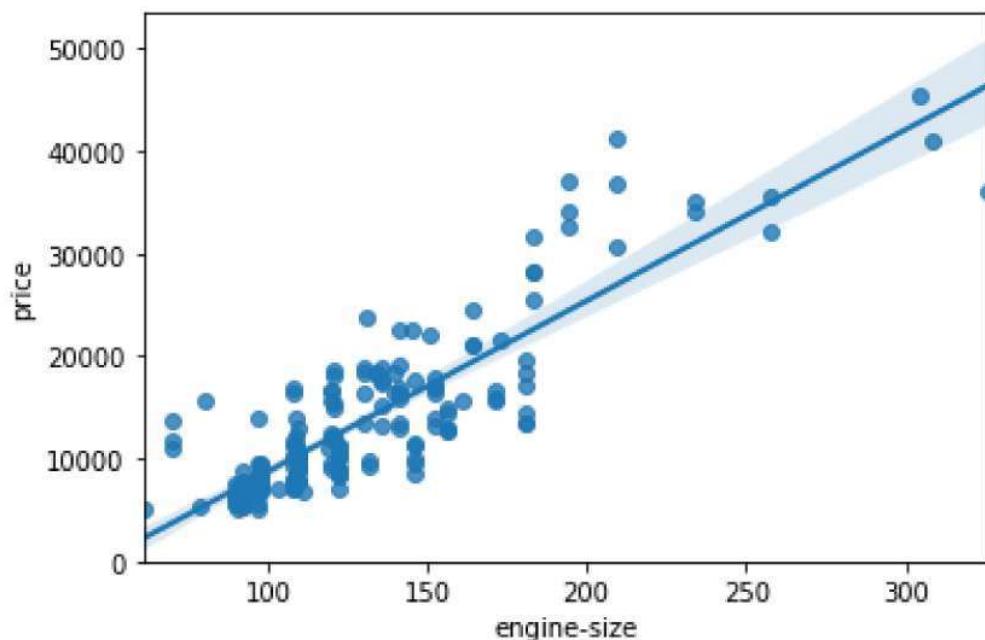
Gambar 0.29. Nilai korelasi antara varibel didalam dataset mobil

Variabel numerik kontinu adalah variabel yang mungkin berisi nilai numerik dalam rentang tertentu. Variabel numerik kontinu dapat memiliki tipe "int64" atau "float64". Cara yang bagus untuk memvisualisasikan variabel-variabel ini adalah dengan menggunakan scatterplots dengan garis-garis yang pas.

Untuk mulai memahami keterhubungan (linier) antara variabel individu dan harga. Kita dapat melakukan ini dengan menggunakan "regplot". Fungsi ini yang memplot scatterplot ditambah garis regresi yang sesuai untuk data (Gambar 0.30).

Hubungan korelasi positif kuat antara variabel

```
# Engine size as potential predictor variable of price  
sns.regplot(x="engine-size", y="price", data=df)  
plt.ylim(0,)
```



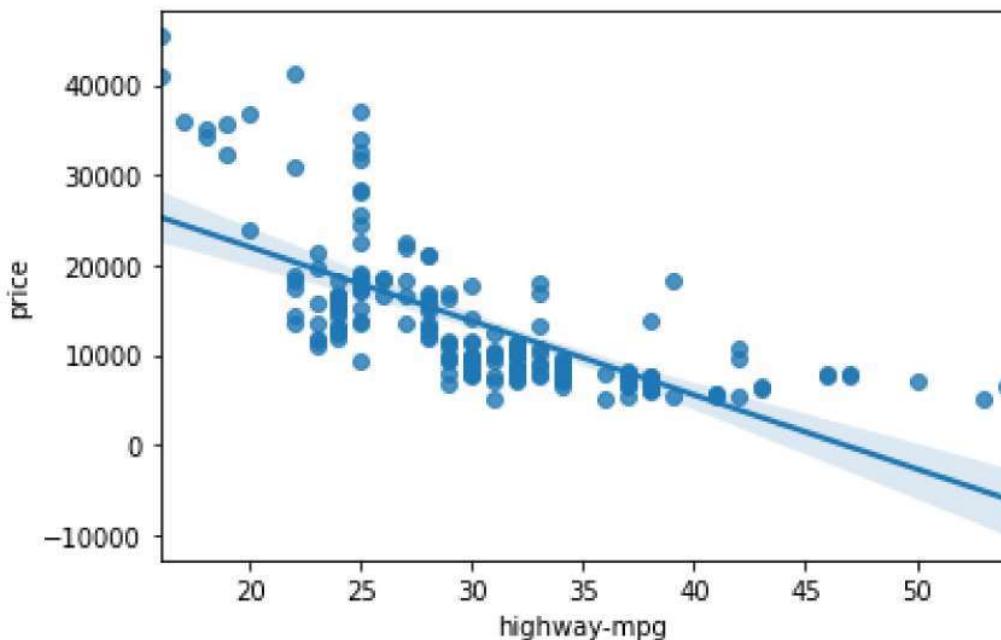
Gambar 0.30. Perbandingan korelasi antara engine-size dan price

Saat kapasitas mesin naik, harga mobil tersebut juga tinggi: ini menunjukkan hubungan linier antara kedua variabel tersebut. Ukuran mesin berpotensi menjadi prediktor harga. Kita dapat memeriksa korelasi antara engine-size dan harga sekitar 0,87

```
df[['engine-size', 'price']].corr()
```

	engine-size	price
engine-size	1.000000	0.872335
price	0.872335	1.000000

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



Gambar 0.31. Perbandingan antara variable highway-mpg dan harga

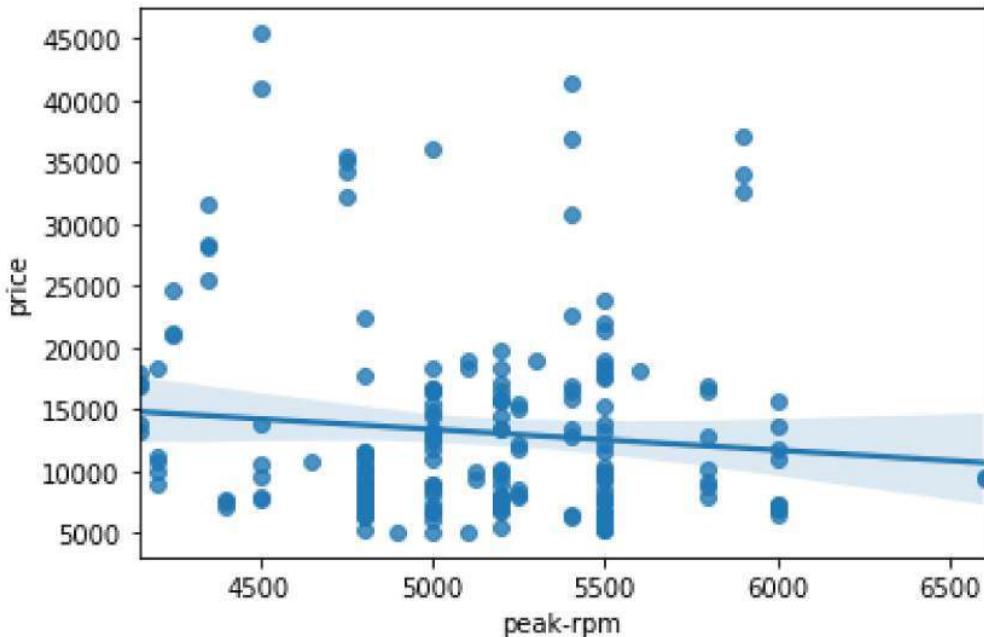
Saat highway-mpg naik, harganya mobil tersebut rendah: ini menunjukkan hubungan terbalik/negatif antara kedua variabel ini. Highway mpg berpotensi menjadi prediktor harga. Hal ini bisa dilihat sebagai korelasi kuat negatif pada Gambar 0.31. Kita dapat memeriksa korelasi antara 'highway-mpg' dan 'price' adalah -0,704

```
df[['highway-mpg', 'price']].corr()
```

	highway-mpg	price
highway-mpg	1.000000	-0.704692
price	-0.704692	1.000000

Weak Linear Relationship

```
sns.regplot(x="peak-rpm", y="price", data=df)
```



Gambar 0.32. Perbandingan antara variable peak-rpm dan harga

Peak rpm sepertinya bukan merupakan prediktor harga yang baik karena garis regresinya mendekati horizontal (Gambar 0.32). Titik-titik data sangat tersebar dan jauh dari garis pas, menunjukkan banyak variabilitas. Oleh karena itu itu bukan variabel yang dapat diandalkan untuk memprediksi harga. Kita dapat memeriksa korelasi antara 'puncak-rpm' dan 'harga' dan melihatnya kira-kira -0,101616

```
df[['peak-rpm', 'price']].corr()
```

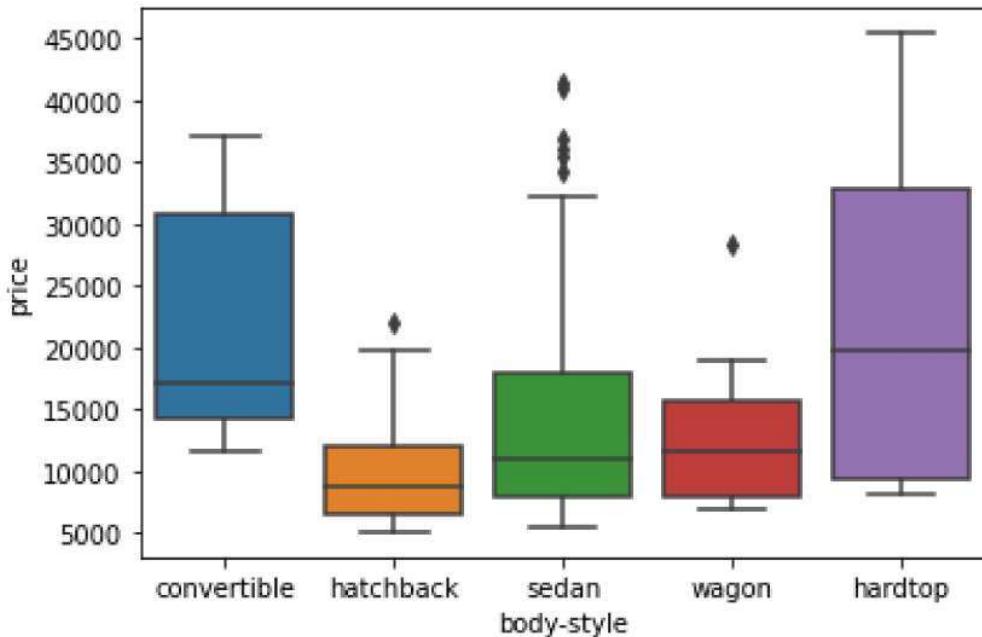
	peak-rpm	price
peak-rpm	1.000000	-0.101616
price	-0.101616	1.000000

Variabel Kategori Statistik

Variabel kategori statistic adalah variabel yang menggambarkan 'karakteristik' dari unit data, dan dipilih dari sekelompok kategori. Variabel kategori dapat memiliki tipe "objek" atau "int64". Cara yang baik untuk memvisualisasikan variabel kategori adalah dengan menggunakan boxplot.

Boxplot menggambarkan variable variable statistic seperti quartil 1, median / quartil 2, quartil 3, nilai maksimum, nilai minimum, dan outlier (Gambar 0.33).

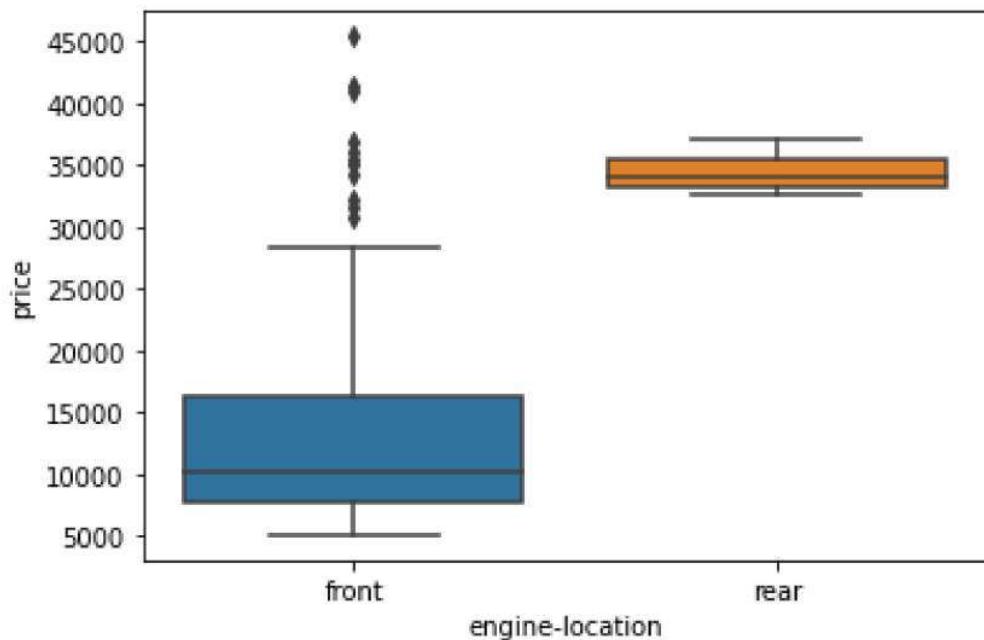
```
sns.boxplot(x="body-style", y="price", data=df)
```



Gambar 0.33. Contoh boxplot dari masing-masing jenis kendaraan

Kita melihat bahwa distribusi harga antara kategori kendaraan memiliki tumpang tindih yang signifikan, sehingga kategori tidak akan menjadi prediktor harga yang baik (Gambar 0.34). Mari kita periksa variable lokasi mesin dan harga:

```
sns.boxplot(x="engine-location", y="price", data=df)
```



Gambar 0.34. Perbandingan box-plot harga antara lokasi mesin di depan dan di belakang.