Modul Pelatihan

Data Understanding 2

Thematic Academy

Digital Talent Scholarship

Tahun 2021

Tujuan Pembelajaran

A. Tujuan Umum

Setelah mempelajari modul ini peserta latih diharapkan mampu menggunakan teknik visualisasi untuk menganalisis data

B. Tujuan Khusus

Adapun tujuan mempelajari unit kompetensi melalui modul visualisasi hingga pada akhir pelatihan diharapkan memiliki kemampuan sebagai berikut:

1. Mampu menelaah data dengan teknik visualisasi
2. Mampu menggunakan teknik visualisasi untuk menganalisis data

Latar Belakang

Unit kompetensi ini dinilai berdasarkan tingkat kemampuan peserta dalam memahami visualisasi dari teknologi AI. Adapun penilaian dilakukan dengan menggabungkan serangkaian metode untuk menilai kemampuan dan penerapan pengetahuan pendukung penting. Penilaian dilakukan dengan mengacu kepada Kriteria Unjuk Kerja (KUK) dan dilaksanakn di Tempat Uji Kompetensi (TUK), ruang simulasi atau workshop dengan cara:

1.1. Lisan
1.2. Wawancara
1.3. Tes tertulis
1.4. Metode lain yang relevan

Deskripsi Pelatihan

Tujuan utama dari modul pelatihan ini adalah untuk membantu para peserta menggunakan teknik visualisasi untuk menganalisis data

Kompetensi Dasar

A. Mampu menelaah data dengan teknik visualisasi
B. Mampu menggunakan teknik visualisasi untuk menganalisis data

Indikator Hasil Belajar

Mampu menelaah dan menganalisis data dengan teknik visualisasi

INFORMASI PELATIHAN

Akademi	Thematic Academy
Mitra Pelatihan	Kementerian Komunikasi dan Informatika
Tema Pelatihan	**Data Scientist: Artificial Intelligence untuk Dosen dan Instruktur**
Sertifikasi	● Certificate of Attainment; ● Sertifikat Kompetensi Associate Data Scientist
Persyaratan Sarana Peserta/spesifikasi device Tools/media ajar yang akan digunakan	Memiliki laptop/komputer dengan spesifikasi minimal: ● RAM minimal 2 GB (disarankan 4 GB) ● Laptop dengan 32/64-bit processor ● Laptop dengan Operating System Windows 7, 8, 10, MacOS X atau Linux ● Laptop dengan konektivitas WiFi dan memiliki Webcam ● Akses Internet Dedicated 126 kbps per peserta per perangkat ● Memiliki aplikasi Zoom ● Memiliki akun Google Colab
Aplikasi yang akan digunakan selama pelatihan	● Word Processor ● Python (numpy, pandas, scipy, matplotlib, seaborn)
Tim Penyusun	Ari Wibisono, M.Kom (Universitas Indonesia)

INFORMASI PEMBELAJARAN

Unit Kompetensi	Materi pembelajaran	Kegiatan pembelajaran	Durasi Pelatihan	Rasio Praktek:Teori	Sumber pembelajaran
Peserta mampu menelaah dan menganalisis data dengan teknik visualisasi	Data Understanding 2	Daring /Online	Live Class 2 JP LMS 4 JP @ 45 menit	70:30	LMS

Materi Pokok

Visualisasi

Sub Materi Pokok

- Visualisasi Variable
- Visualisasi Statistik
- Visualisasi Deskriptif Statistik
- Visualisasi Grouping

Materi Pelatihan Visualisasi

Visualisasi Variabel

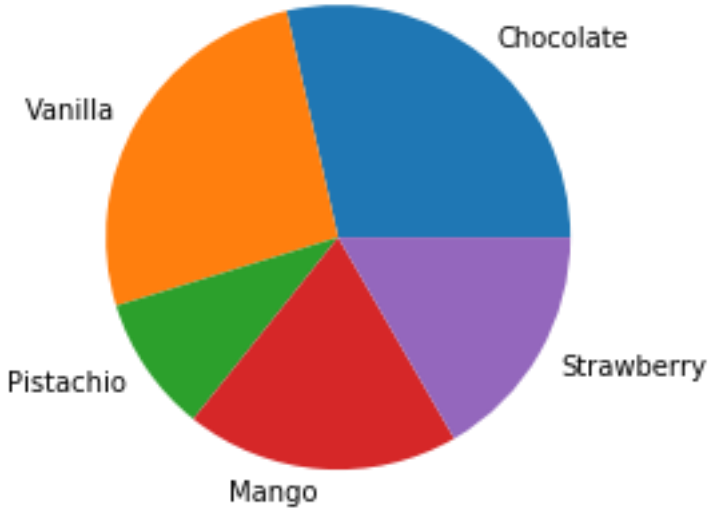
Visualisasi berperan peran penting dalam bidang machine learning dan data science. Seringkali kita perlu menyaring informasi kunci yang ditemukan dalam sejumlah data untuk menjadi bentuk yang bermakna dan mudah dicerna. Visualisasi yang baik dapat menceritakan sebuah cerita tentang data dengan cara yang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

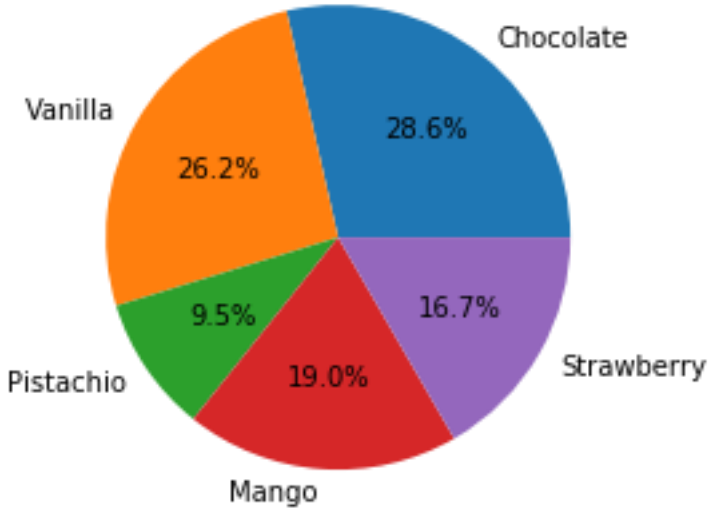
```
import matplotlib.pyplot as plt

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 kirim, 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()
```



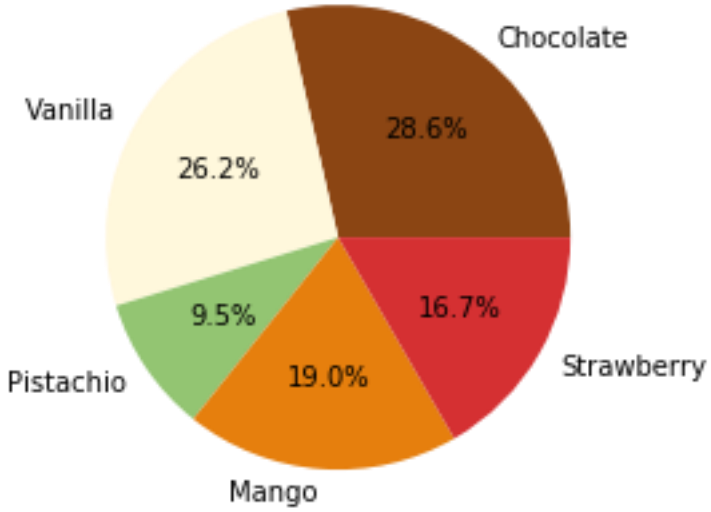
Gambar 0.2. Penambahan argument persentase

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).

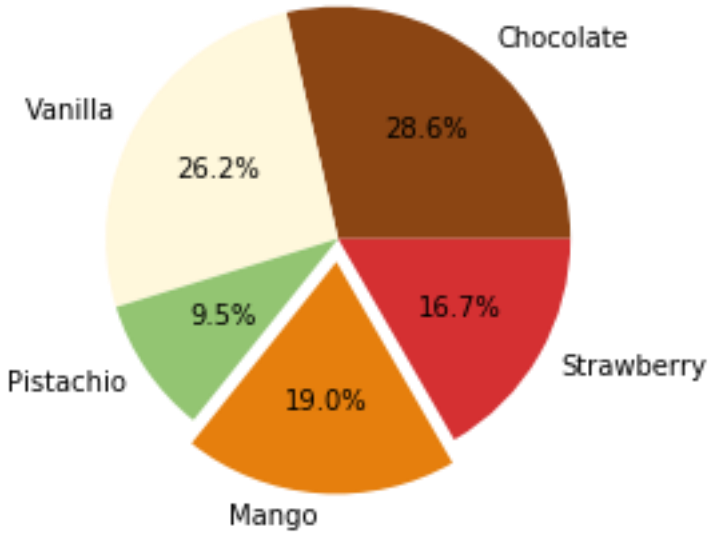
```
import matplotlib.pyplot as plt

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.

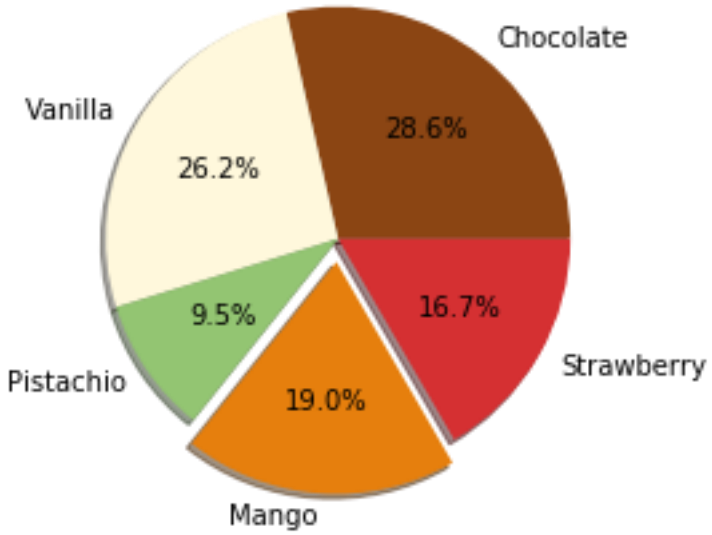
```
import matplotlib.pyplot as plt

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).

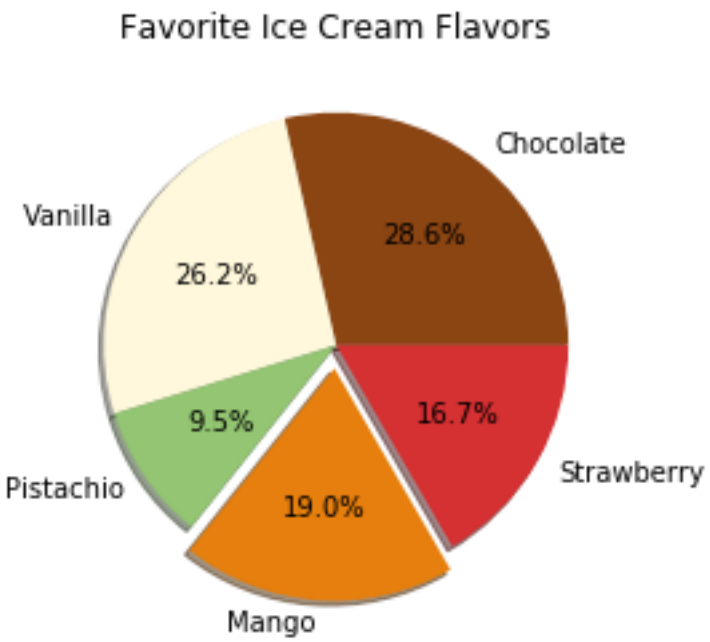
```
import matplotlib.pyplot as plt

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 menyelesaikannya, 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 survey! 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

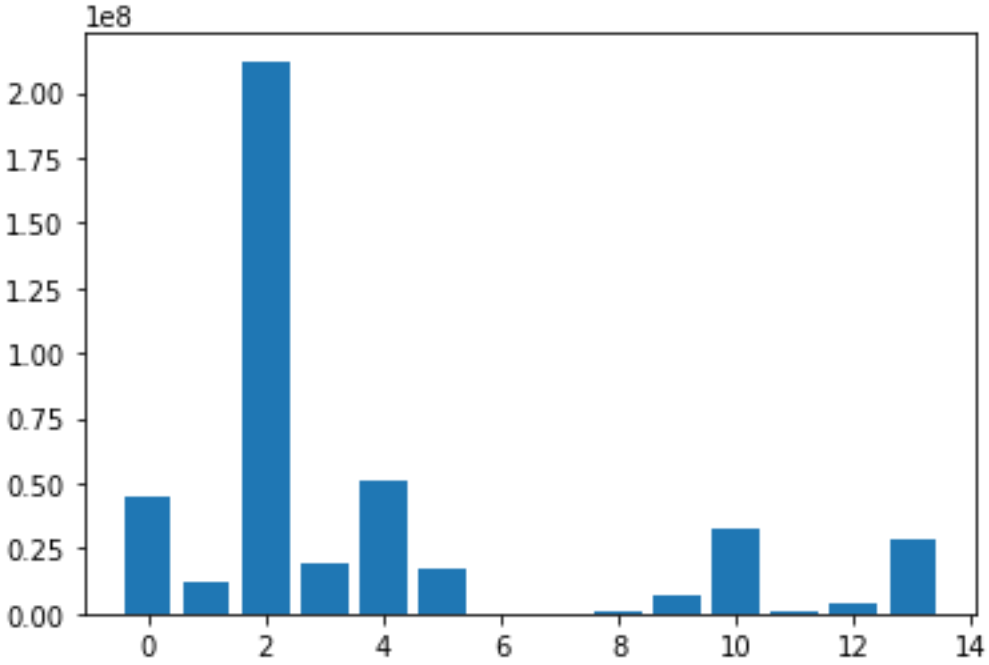
Bar Charts

Bar Chart adalah merupakan tools visualisassi 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()
```

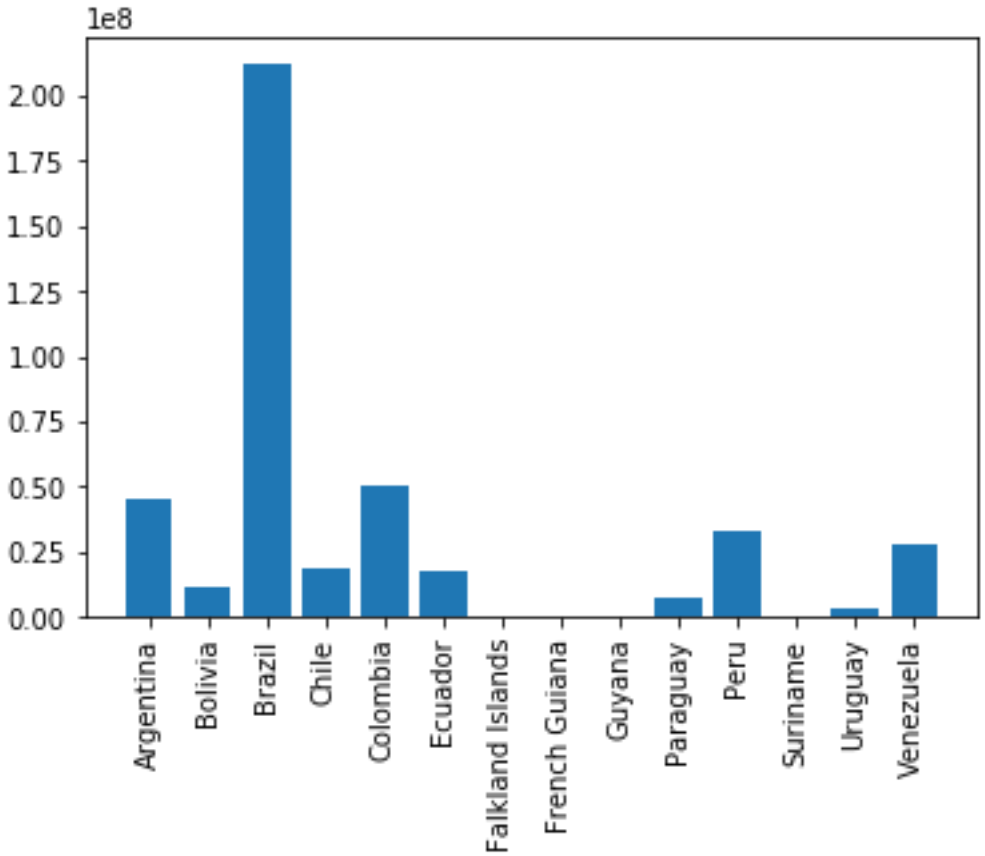


Gambar 0.7. Bar Charts Simple Visualisations

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')

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()
```

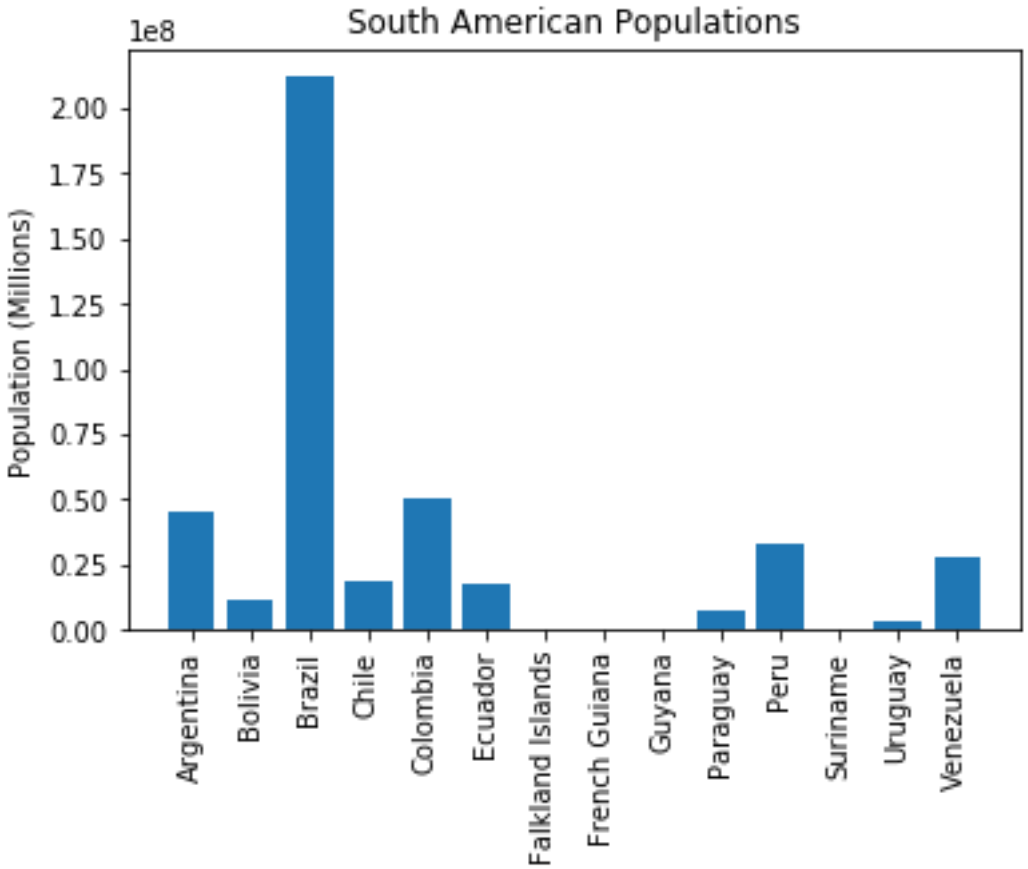


Gambar 0.8. Penambahan label pada bar chart

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()
```

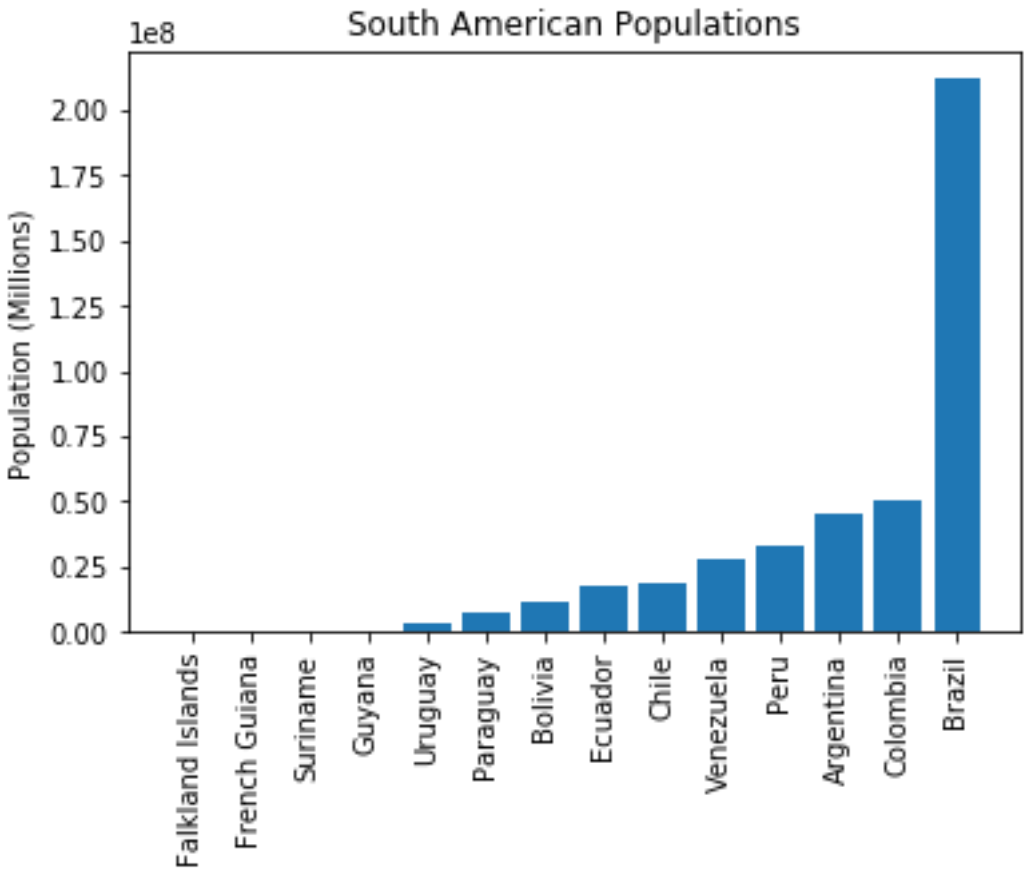


Gambar 0.9. Penambahan label dan title pada bar chart

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 menggunakannya 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()
```



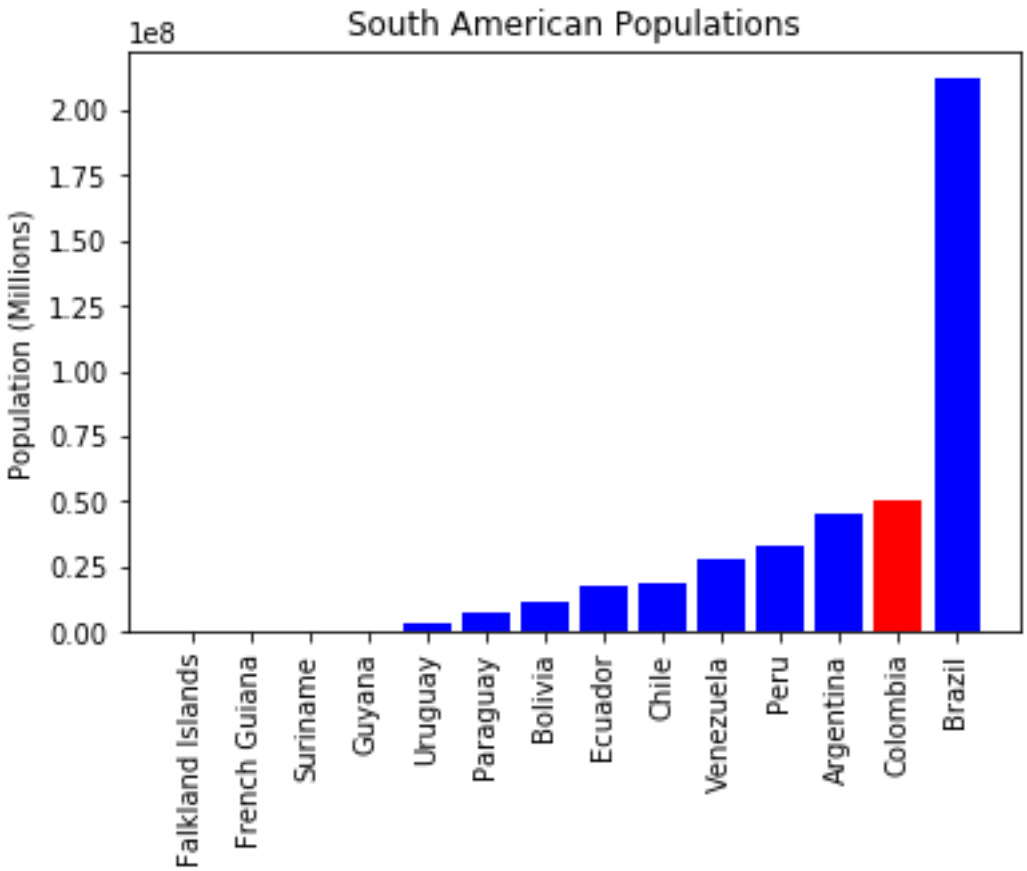
Gambar 0.10. Pengurutan Jumlah Populasi pada Bar Chart

```
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

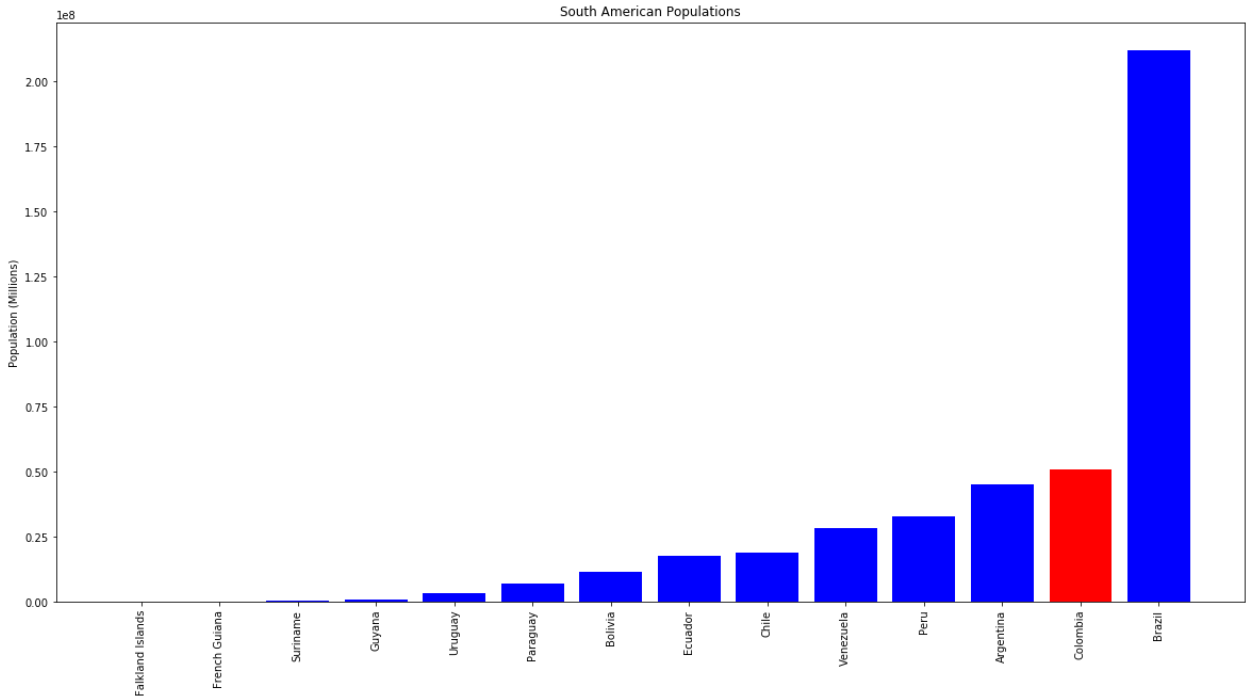
```
colors
['#0000FF',
 '#0000FF',
 '#0000FF',
 '#0000FF',
 '#0000FF',
```

```
 '#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()
```



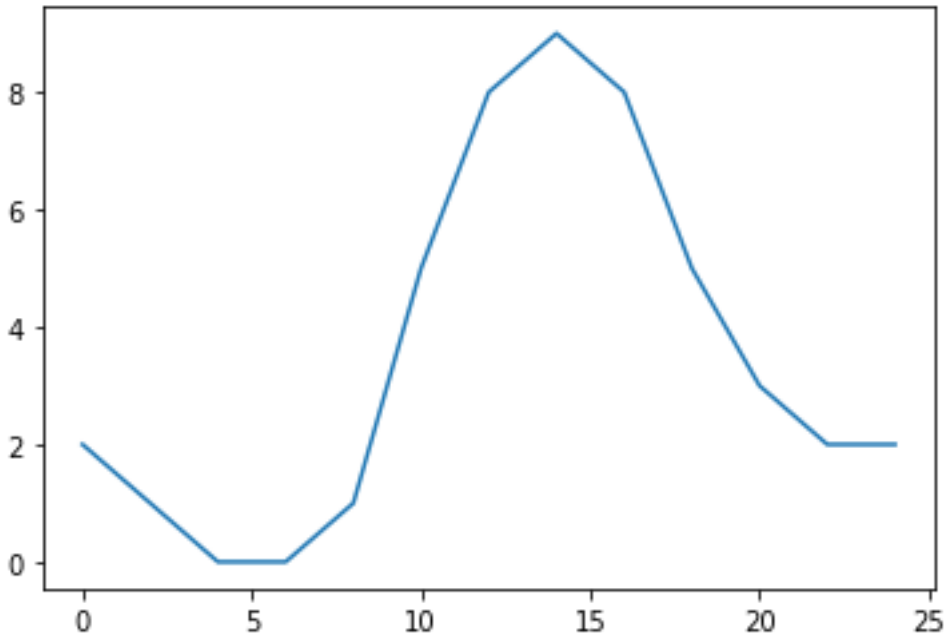
Gambar 0.12. Contoh Hasil Bar Chart Lengkap

Line Graphs

Line Graph adalah bentuk visualisasi lainya 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()
```

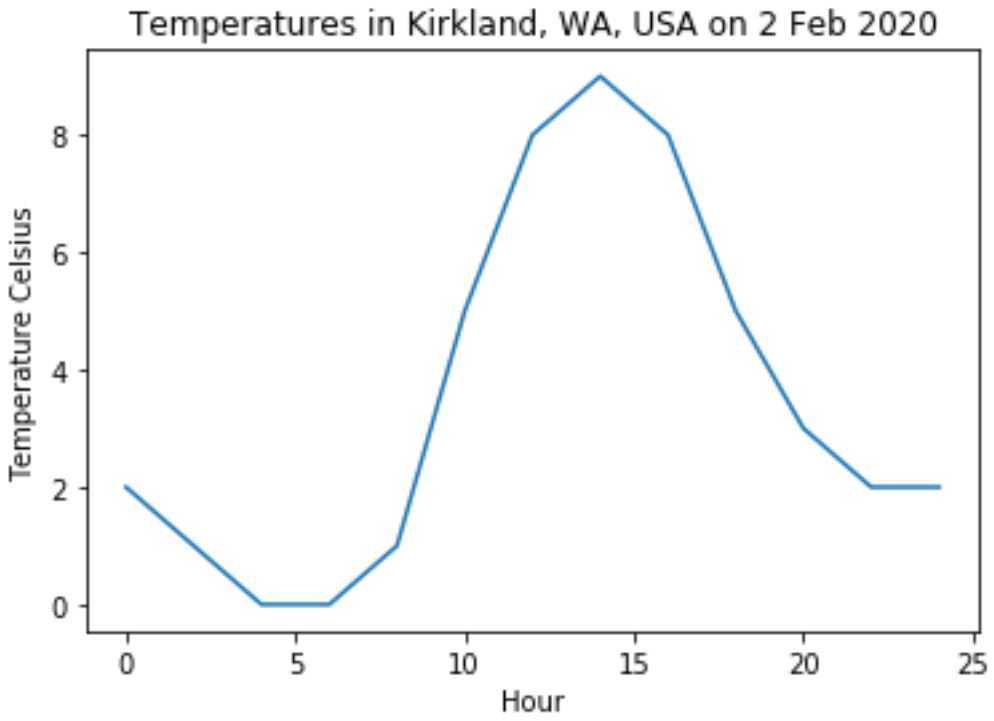


Gambar 0.13. Contoh line graph

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).

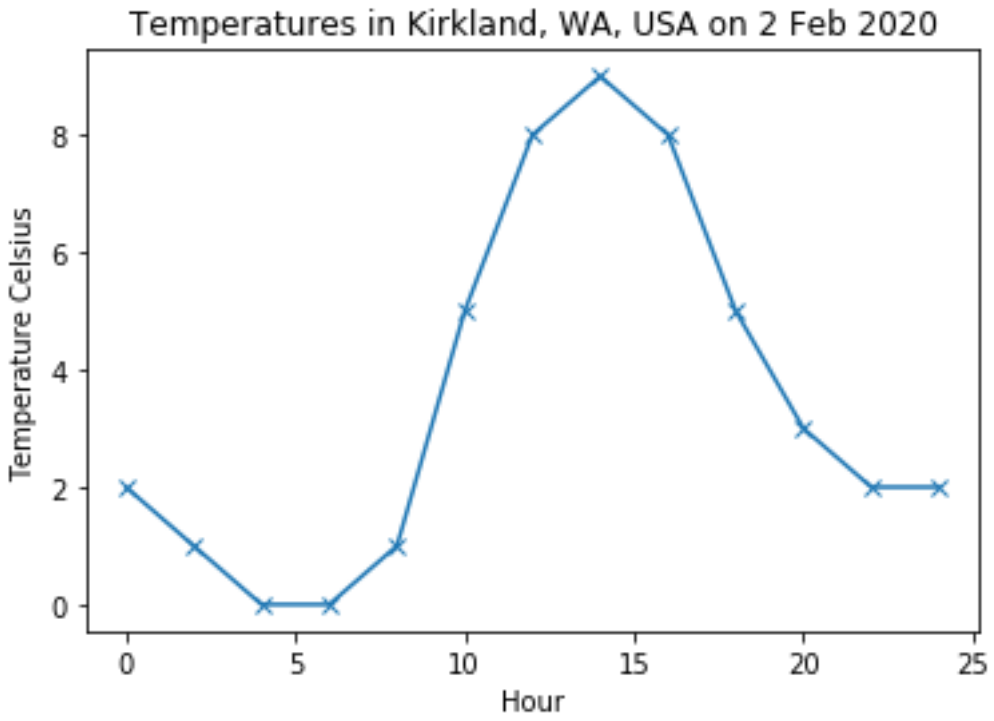
```
import matplotlib.pyplot as plt

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()
```

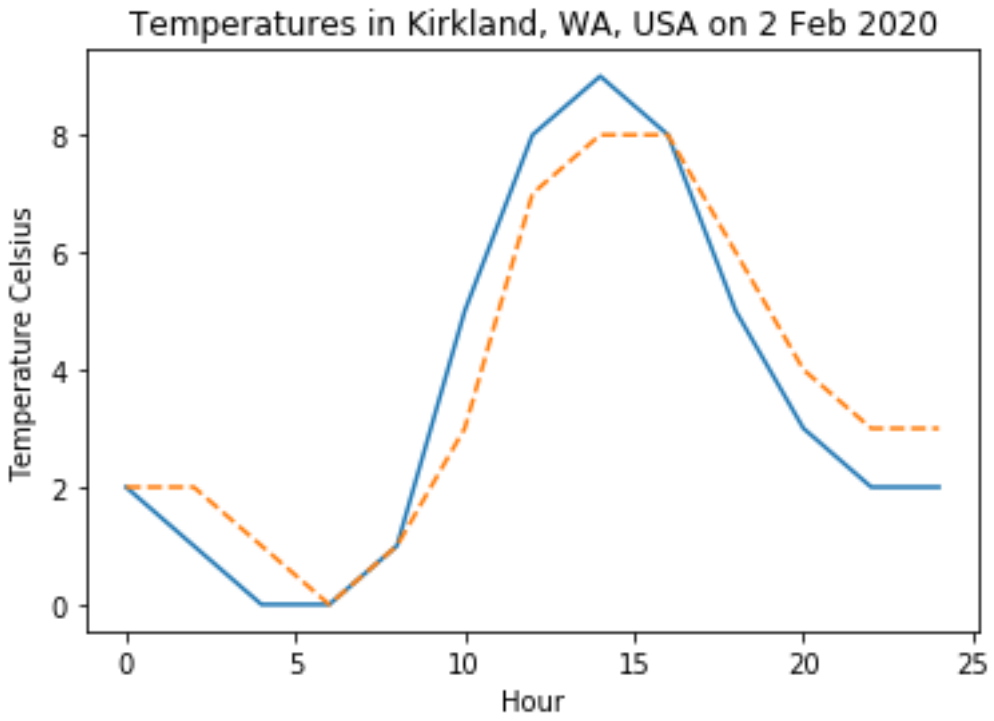


Gambar 0.15. Line Graph dengan Penanda disetiap Titik

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()
```

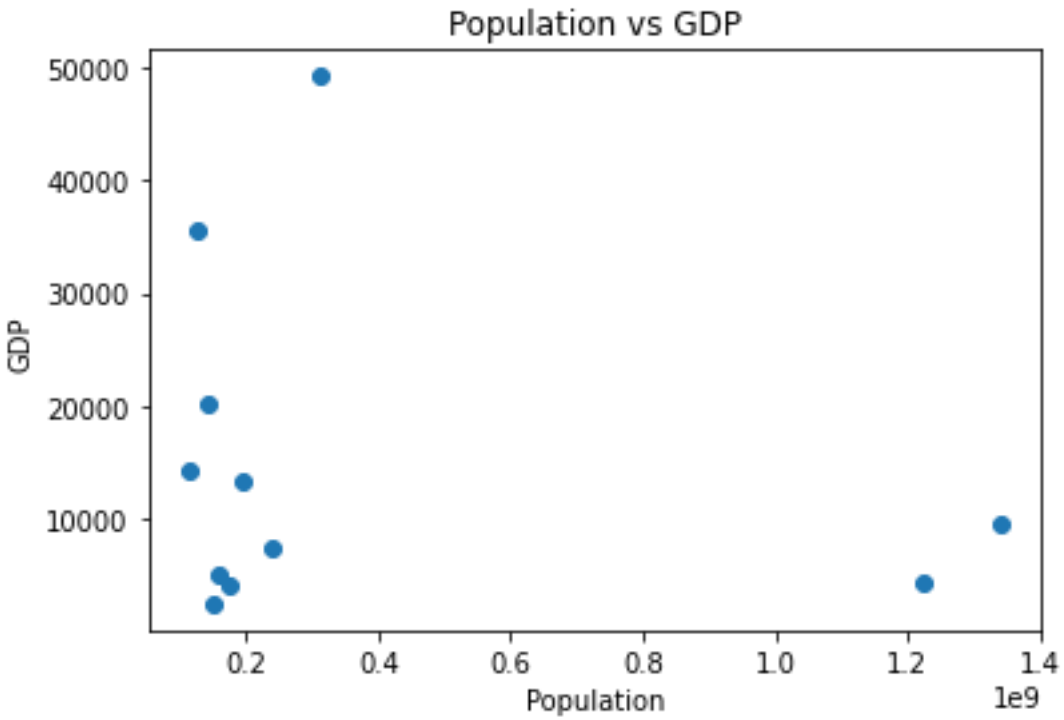


Gambar 0.16. Line Graph dengan Lebih dari satu garis

Scatter Plot

Scatter plot berfungsi baik untuk data dengan dua komponen numerik. Scatter plot dapat memberikan informasi yang berguna terutama mengenai pola atau pencilan. 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()
```

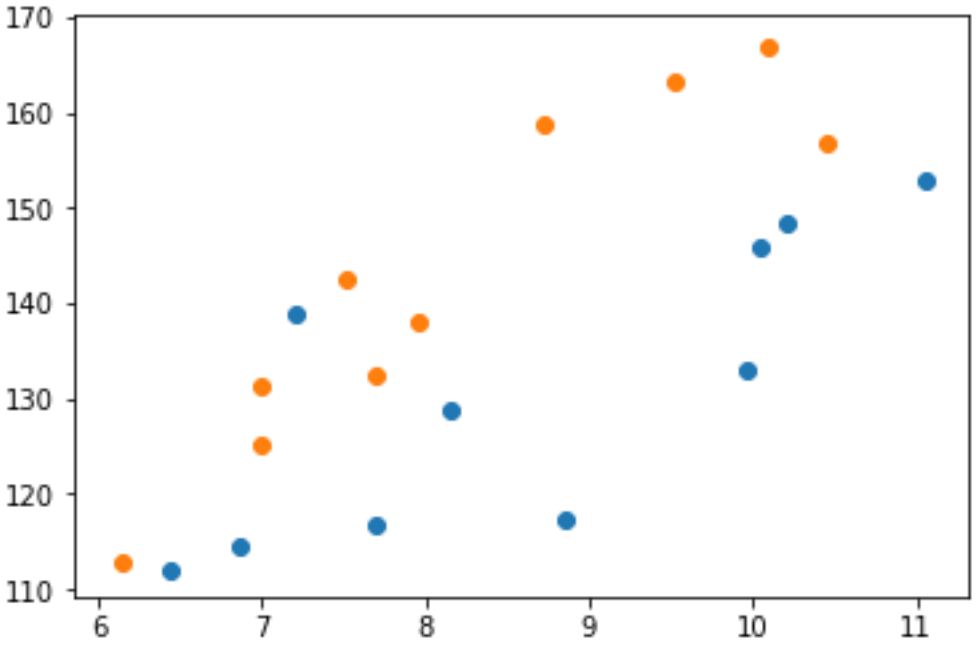


Gambar 0.17. Scatter Plot data PDB dan populasi

Scatter plot sangat intuitif karena kita dapat mengumpulkan informasi tentang data kita. Kita dapat melihat bahwa ada dua pencilan populasi (pencilan PDB). Informasi ini dapat membantu kita memutuskan apakah kita perlu mengoreksi atau mengecualikan pencilan 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.

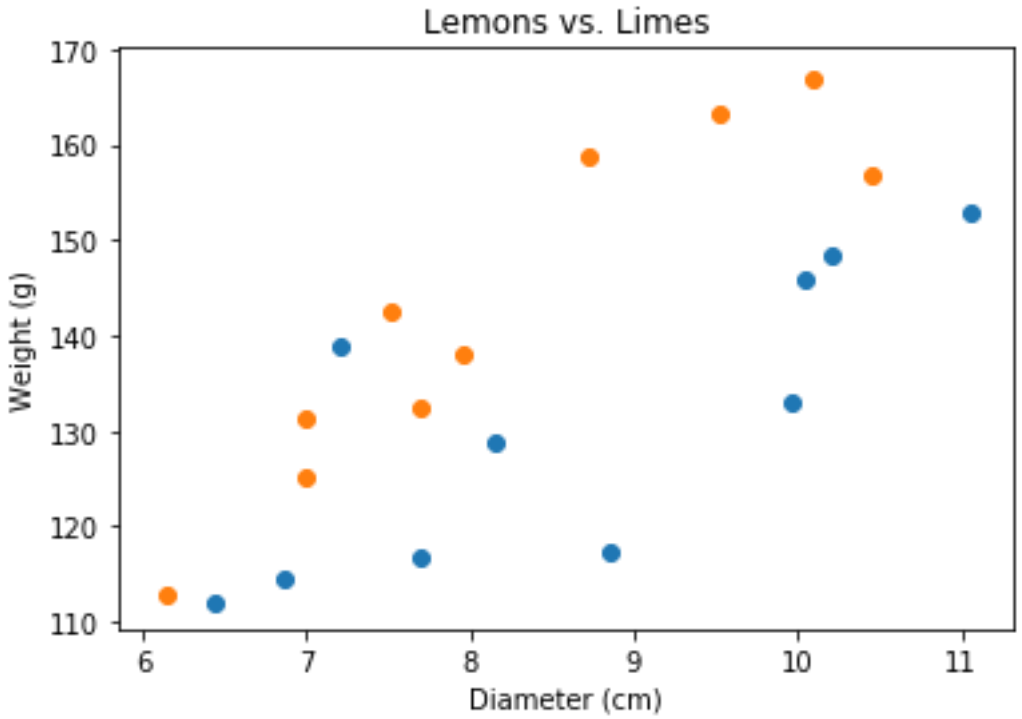
```
import matplotlib.pyplot as plt

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()
```

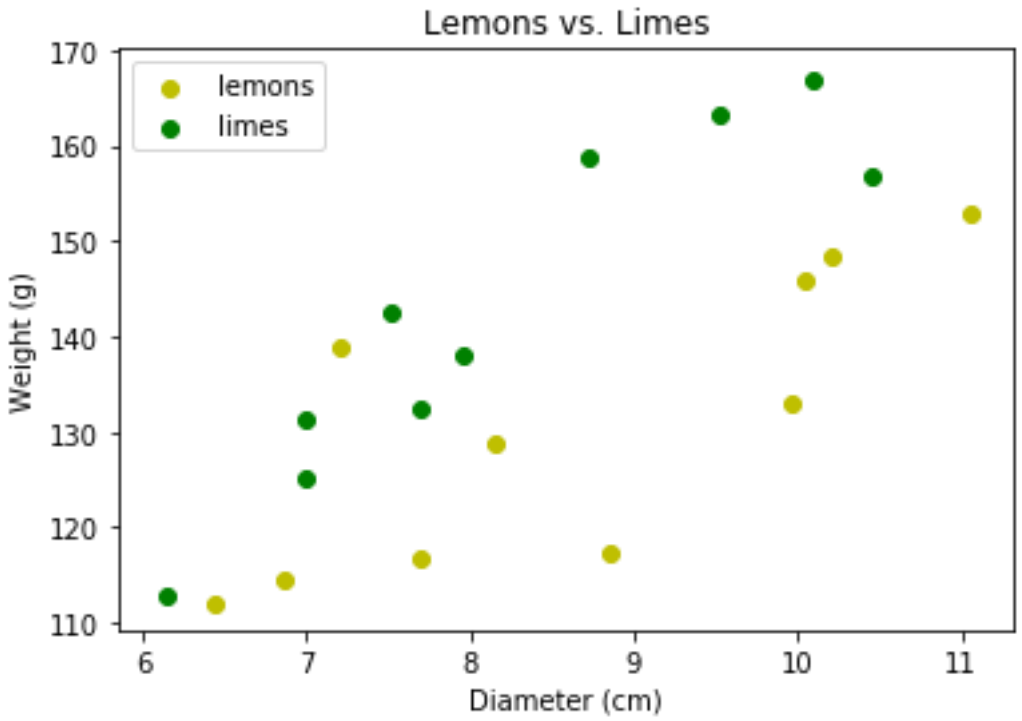


Gambar 0.19 Lemon vs lime dengan label

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

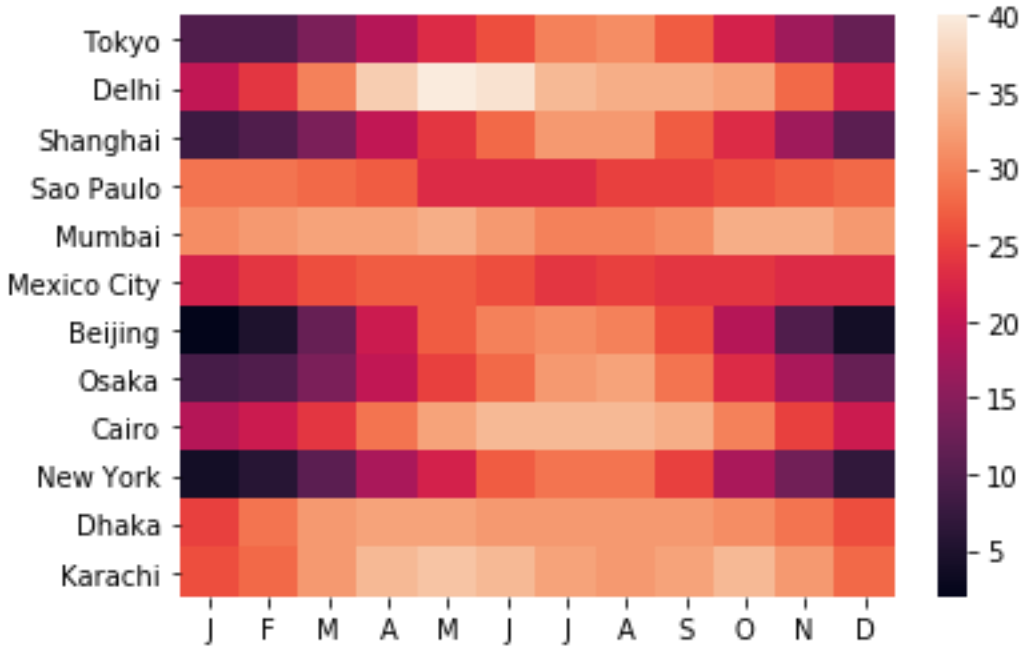
Heatmap adalah jenis visualisasi yang menggunakan kode warna untuk mewakili nilai / kepadatan relatif data di seluruh permukaan. Seringkali ini adalah bagan tabel, tetapi tidak harus terbatas pada itu. Untuk data tabular, terdapat label pada sumbu x dan y. Nilai di persimpangan label tersebut dipetakan 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, 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>
```

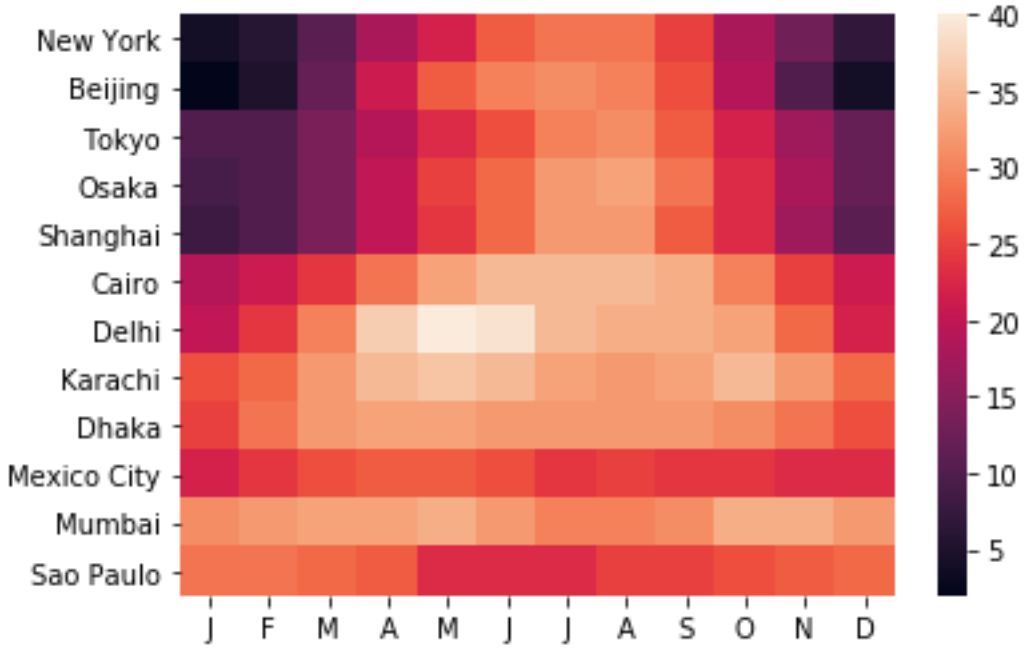


Gambar 0.21. Heatmtp mengenai temperatur di masing-masing kota selama 12 bulan

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>
```

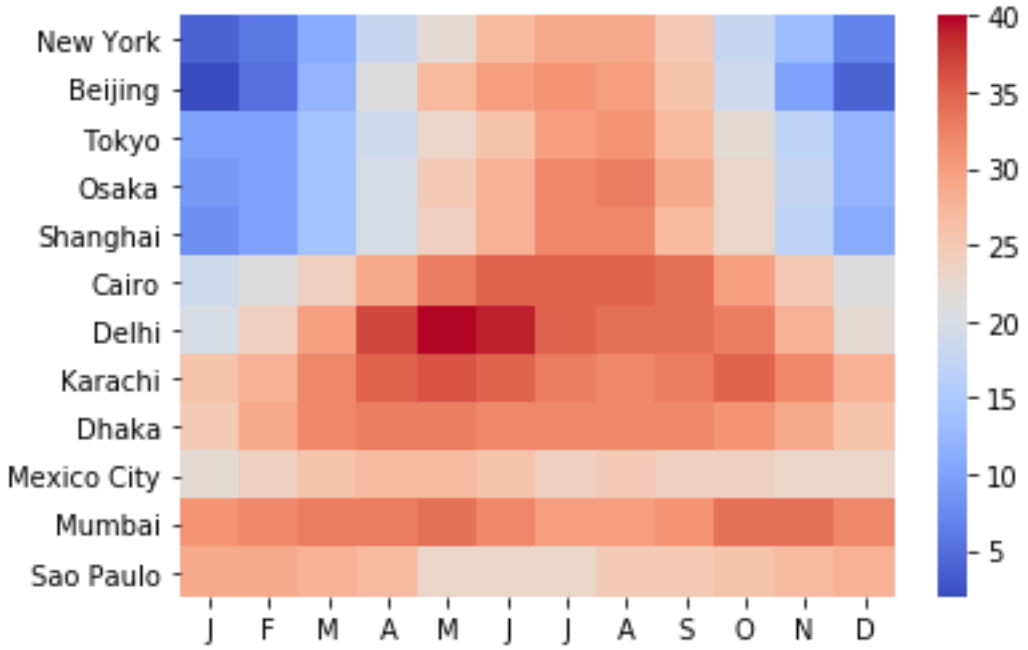


Gambar 0.22. Heatmap temperature negara berdasarkan garis bumi (lintang)

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, 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.

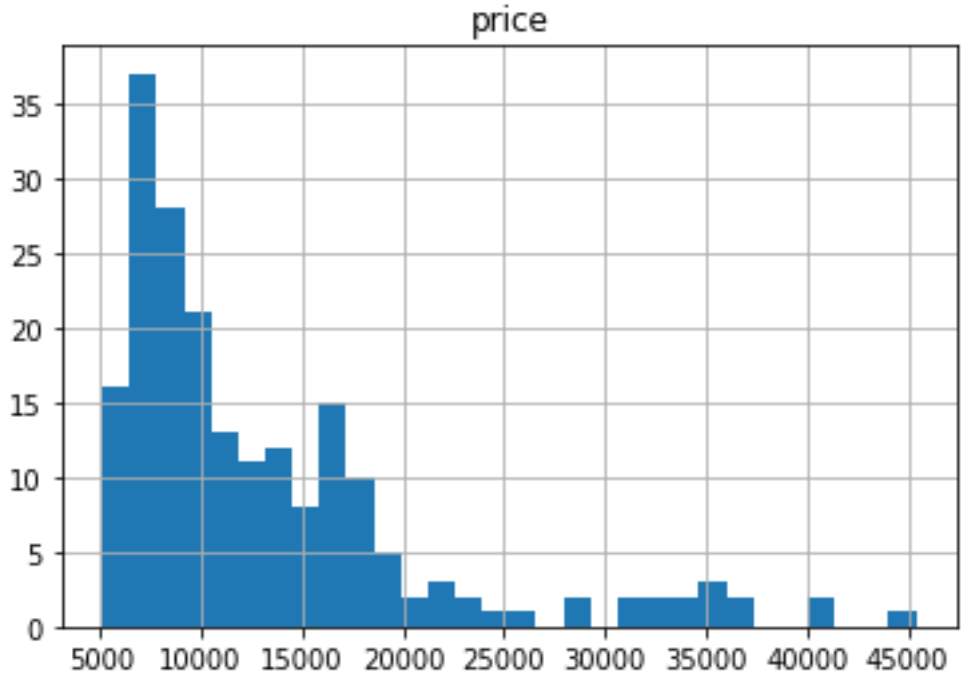
Sebagai contoh studi kasus data yang kita gunakan dalam modul ini adalah automobile.csv yang merupakan data-data spesifikasi kendaraan dari berbagai merek dan harganya. Overview data automobile dapat dilihat pada Gambar 0.24. Histogram pada data automobile dapat kita lihat pada Gambar 0.25, histogram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

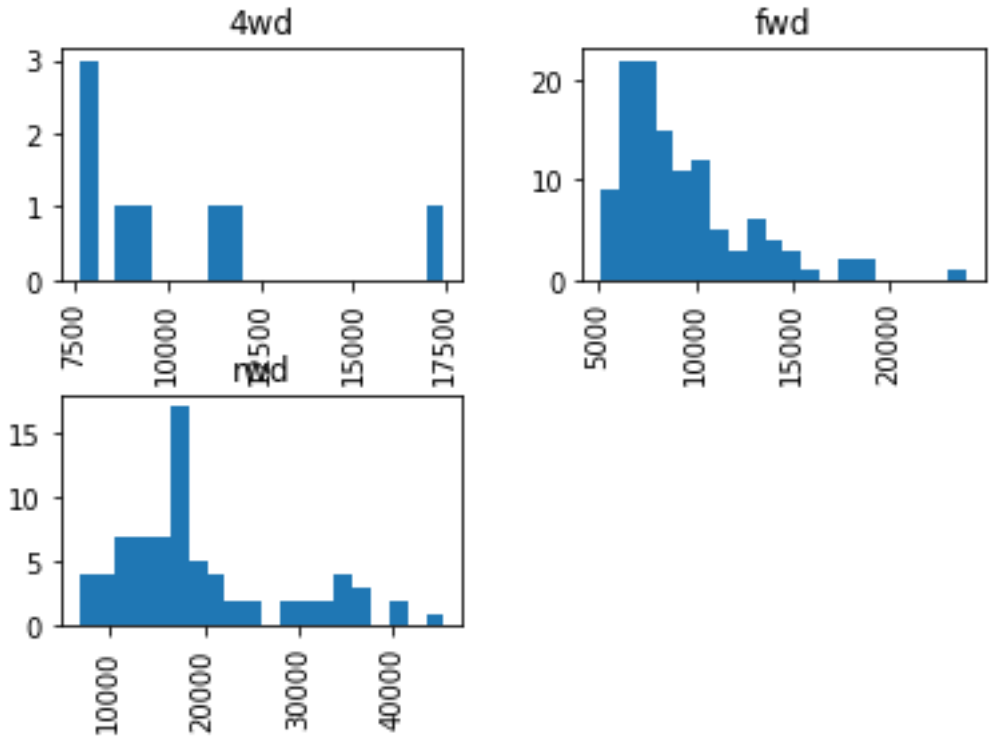
```
df.hist(column='price', bins=30);
```



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 kenderaan (fwd – berpenggerak roda depan, 4wd – berpenggerak 4 roda, atau rwd – pengerak belakang (Gambar 0.26).

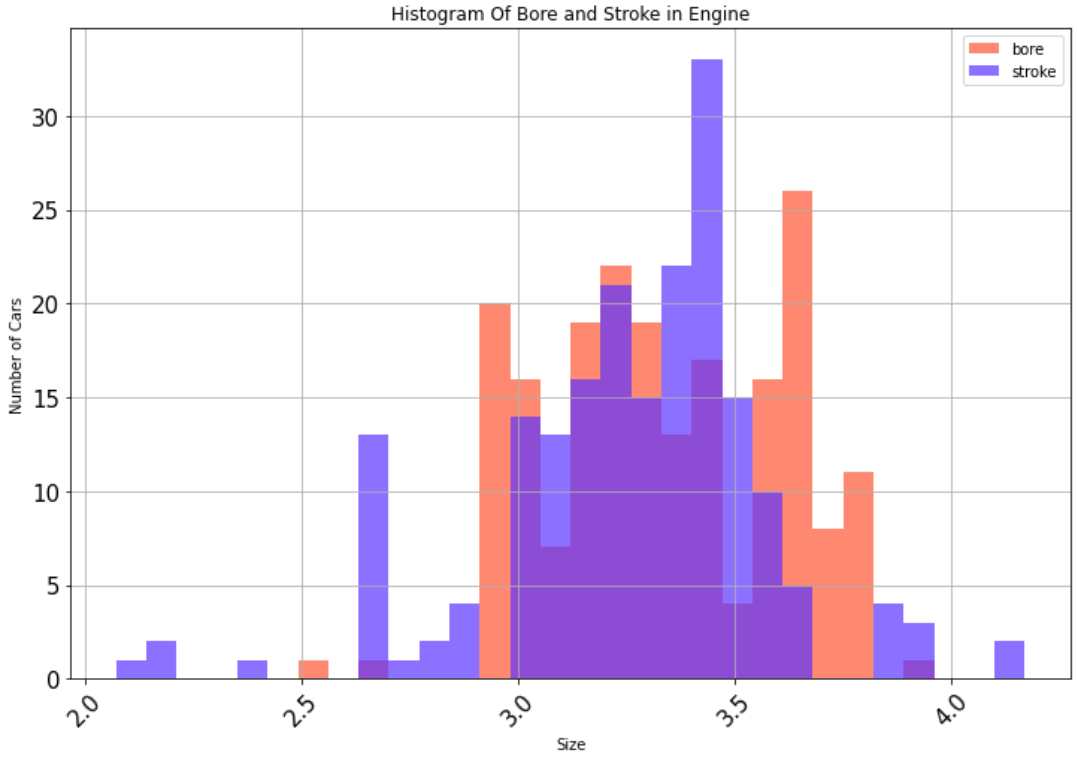
```
df.hist(column='price', by='drive-wheels', bins=20);
```



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 dan 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):

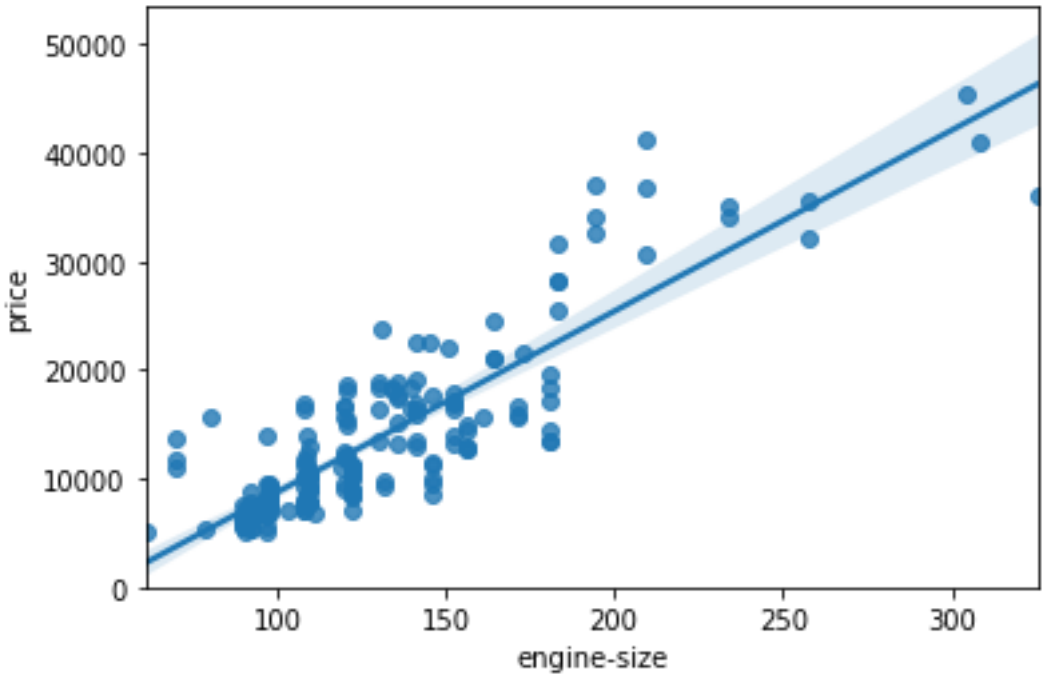
	symboling	normalized-losses	wheel-base	length	width	height	curb-weight	engine-size	bore	stroke	compression-ratio	horsepower	peak-rpm	city-mpg	highway-mpg	price
symboling	1.000000	0.466264	-0.535987	-0.365404	-0.242423	-0.550160	-0.233118	-0.110581	-0.140019	-0.008245	-0.182196	0.075819	0.279740	-0.035527	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	0.584642
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	0.306002	0.866201	0.729436	0.544885	0.188829	0.189867	0.615077	-0.245800	-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	-0.087027	-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	0.644060	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	0.822676	-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	-0.267392	-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	-0.245800	-0.309974	-0.279361	-0.256733	-0.267392	-0.065713	-0.435780	0.107885	1.000000	-0.115413	-0.058598	-0.101616
city-mpg	-0.035527	-0.225016	-0.470606	-0.665192	-0.633531	-0.049800	-0.749543	-0.650546	-0.582027	-0.034696	0.331425	-0.822214	-0.115413	1.000000	0.972044	-0.686571
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	0.584642	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	0.785353	0.745059	0.554610	0.037300	-0.299372	0.889488	0.115830	-0.949713	-0.930028	0.789898
diesel	-0.196735	-0.101546	0.307237	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	-0.307237	-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 nunerik 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 positf kuat antara variabel

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



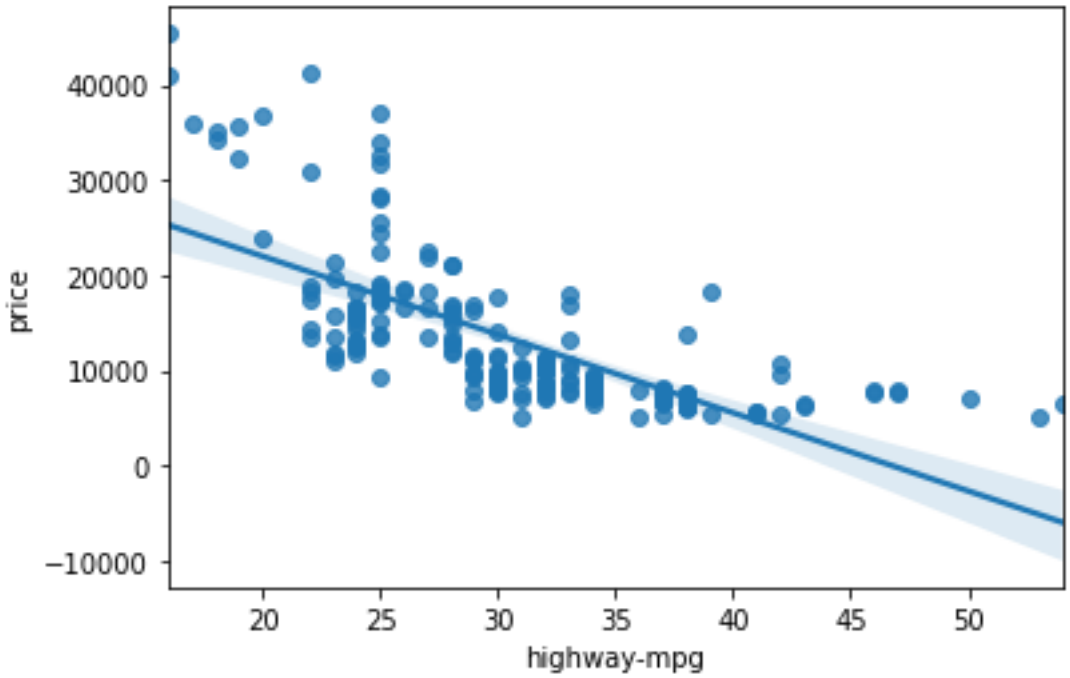
Gambar 0.30. Perbandingan korelasi antara engine-size dan price

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

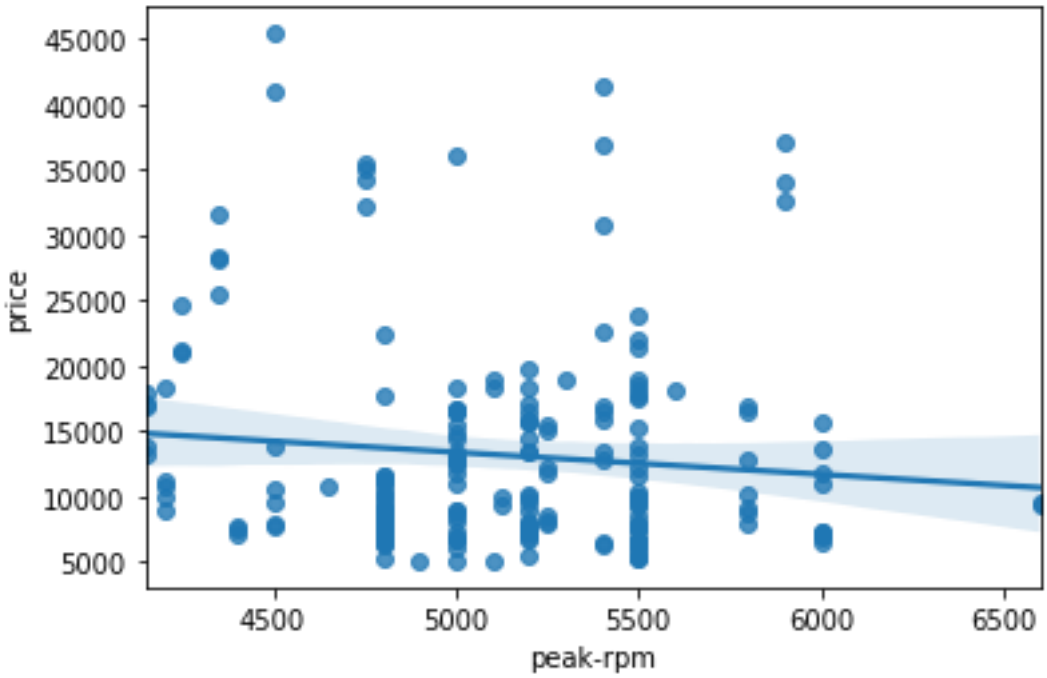
Saat highway-mpg naik, harganya mobil tersebut rendah: ini menunjukkan hubungan terbalik/negatif antara kedua variabel ini. Highway mpg berpotensi menjadi prediktor harga. Hal ini bisa dilhat sebagai korelasi kuat negative pada Gambar 0.31. Kita dapat memeriksa korelasi antara 'highway-mpg' dan 'price' 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 memperdiksi harga. Kita dapat memeriksa korelasi antara 'puncak-rpm' dan 'harga' dan melihatnya kira-kira -0,101616

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

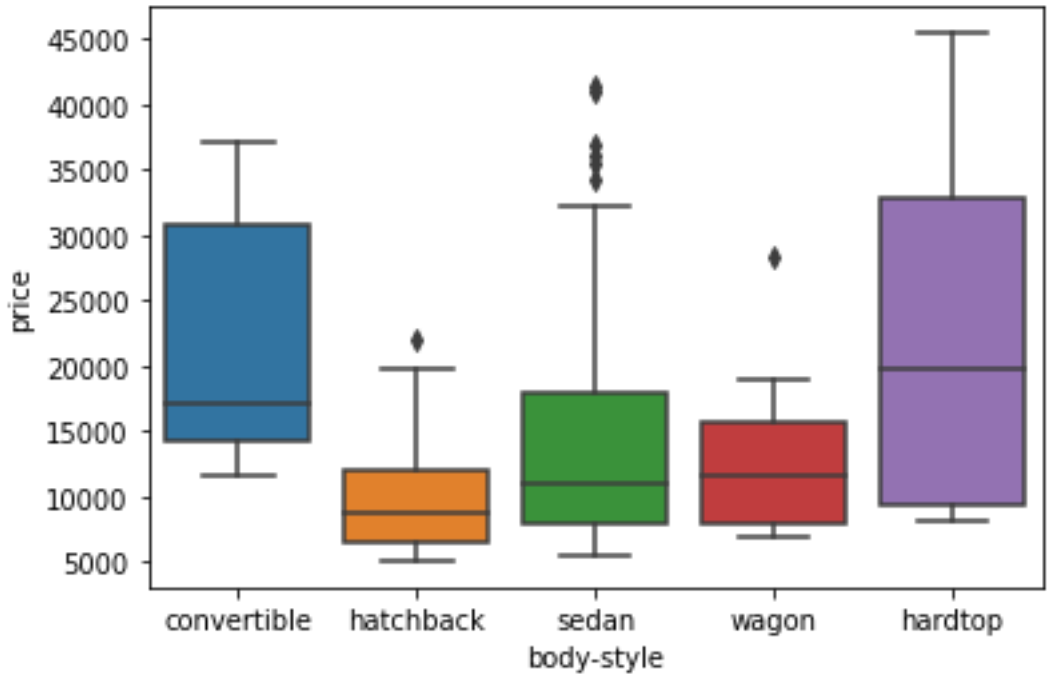
	peak-rpm	price
peak-rpm	1.000000	-0.101616
price	-0.101616	1.000000

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).

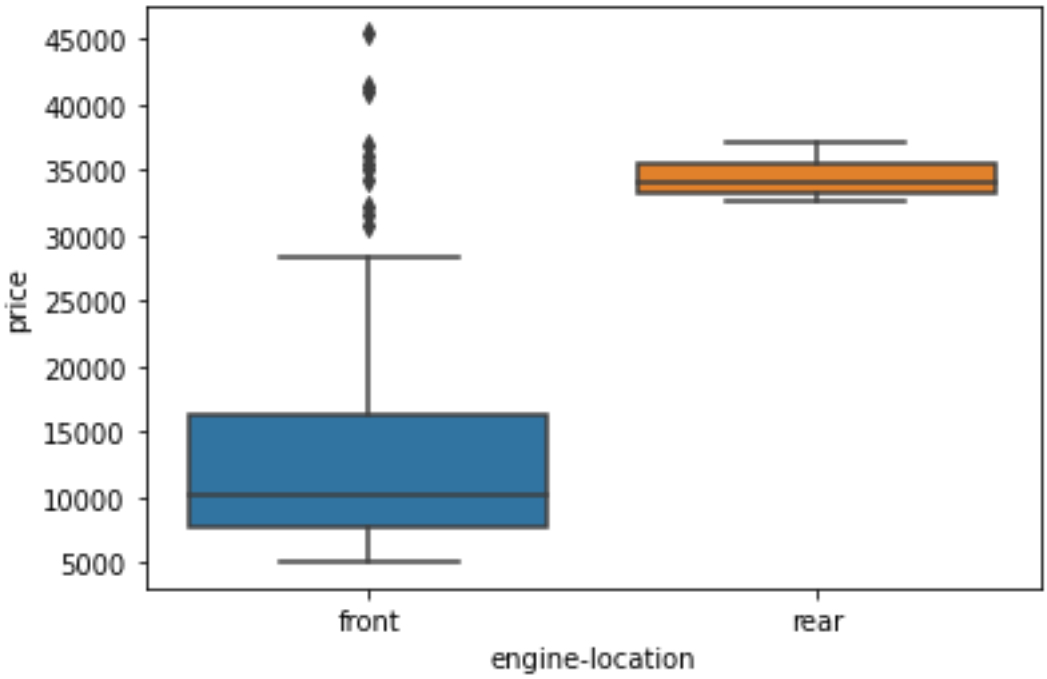
```
sns.boxplot(x="body-style", y="price", data=df)
```



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.

Visualisasi Deskriptif Statistik

Fungsi deskripsikan secara otomatis menghitung statistik dasar untuk semua variabel kontinu (Gambar 0.36). Analisis yang bisa kita dapatkan dari deskriptif statistik adalah

- Jumlah variabel
- Rata-rata
- Standard deviasi
- Nilai minimal
- IQR (Interquartile Range: 25%, 50% and 75%)
- Nilai Maximal

```
df.describe()
```

	symboling	normalized-losses	wheel-base	length	width	height	curb-weight	engine-size	bore	stroke	compression-ratio	horsepower	peak-rpm	city-mpg	highway-mpg	price	city-L/100km	diesel	gas
count	201.000000	201.00000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000	197.000000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000
mean	0.840796	122.00000	98.797015	0.837102	0.915126	53.766667	2555.666667	126.875622	3.330692	3.256904	10.164279	103.405534	5117.665368	25.179104	30.686567	13207.129353	9.944145	0.099502	0.900498
std	1.254802	31.99625	6.066366	0.059213	0.029187	2.447822	517.296727	41.546834	0.268072	0.319256	4.004965	37.365700	478.113805	6.423220	6.815150	7947.066342	2.534599	0.300083	0.300083
min	-2.000000	65.00000	86.600000	0.678039	0.837500	47.800000	1488.000000	61.000000	2.540000	2.070000	7.000000	48.000000	4150.000000	13.000000	16.000000	5118.000000	4.795918	0.000000	0.000000
25%	0.000000	101.00000	94.500000	0.801538	0.890278	52.000000	2169.000000	98.000000	3.150000	3.110000	8.600000	70.000000	4800.000000	19.000000	25.000000	7775.000000	7.833333	0.000000	1.000000
50%	1.000000	122.00000	97.000000	0.832292	0.909722	54.100000	2414.000000	120.000000	3.310000	3.290000	9.000000	95.000000	5125.369458	24.000000	30.000000	10295.000000	9.791667	0.000000	1.000000
75%	2.000000	137.00000	102.400000	0.881788	0.925000	55.500000	2926.000000	141.000000	3.580000	3.410000	9.400000	116.000000	5500.000000	30.000000	34.000000	16500.000000	12.368421	0.000000	1.000000
max	3.000000	256.00000	120.900000	1.000000	1.000000	59.800000	4066.000000	326.000000	3.940000	4.170000	23.000000	262.000000	6600.000000	49.000000	54.000000	45400.000000	18.076923	1.000000	1.000000

Gambar 0.35. Hasil descriptive statistics

Pengaturan default "describe" melewatkan variabel tipe objek. Kita bisa menggunakan code ini untuk menghitung jumlah type data objek (Gambar 0.36).

```
df.describe(include=['object'])
```

	make	aspiration	num-of-doors	body-style	drive-wheels	engine-location	engine-type	num-of-cylinders	fuel-system	horsepower-binned
count	201	201	201	201	201	201	201	201	201	200
unique	22	2	2	5	3	2	6	7	8	3
top	toyota	std	four	sedan	fwd	front	ohc	four	mpfi	Low
freq	32	165	115	94	118	198	145	157	92	115

Gambar 0.36. Hasil descriptive statistic untuk type data objek

Nilai-hitungan adalah cara untuk memahami berapa banyak unit dari setiap karakteristik/variabel yang kita miliki. Kita bisa menerapkan metode "value_counts" pada kolom 'drive-wheels'. Jangan lupa metode "value_counts" hanya berfungsi pada seri Pandas, bukan Pandas Dataframes.

```
df['drive-wheels'].value_counts()
fwd    118
rwd     75
4wd      8
Name: drive-wheels, dtype: int64
```

Kita dapat mengonversi seri ke Dataframe sebagai berikut:

```
df['drive-wheels'].value_counts().to_frame()
```

	drive-wheels
fwd	118
rwd	75
4wd	8

Mari ulangi langkah di atas tetapi simpan hasilnya ke dataframe "drive_wheels_counts" dan ganti nama kolom 'drive-wheels' menjadi 'value_counts'.

```
drive_wheels_counts = df['drive-wheels'].value_counts().to_frame()
drive_wheels_counts.rename(columns={'drive-
wheels': 'value_counts'}, inplace=True)
drive_wheels_counts
```

	value_counts
fwd	118
rwd	75
4wd	8

Sekarang mari kita ganti nama indeks menjadi 'drive-wheels':

```
drive_wheels_counts.index.name = 'drive-wheels'
drive_wheels_counts
```

drive-wheels	value_counts
fwd	118
rwd	75
4wd	8

Kita dapat mengulangi proses di atas untuk variabel 'engine-location'.

```
# engine-location as variable
engine_loc_counts = df['engine-location'].value_counts().to_frame()
engine_loc_counts.rename(columns={'engine-
location': 'value_counts'}, inplace=True)
engine_loc_counts.index.name = 'engine-location'
engine_loc_counts.head(10)
```

engine-location	value_counts
front	198
rear	3

Memeriksa jumlah lokasi mesin mobil tidak akan menjadi variabel prediktor yang baik untuk harga. Karena, kita hanya punya tiga mobil dengan mesin belakang dan 198 dengan mesin di depan, hasilnya sangat tidak semimbang. Oleh karena itu, lokasi mesin bukan sebagai perdiktor yang baik untuk harga.

Grouping

Method "groupby" digunakan untuk mengelompokkan data menurut kategori yang berbeda. Data dikelompokkan berdasarkan satu atau beberapa variabel dan analisis dilakukan pada kelompok individu.

Sebagai contoh, mari kita kelompokkan berdasarkan variabel "roda penggerak". Kita melihat bahwa ada 3 kategori roda penggerak yang berbeda.

```
df['drive-wheels'].unique()
```

```
array(['rwd', 'fwd', '4wd'], dtype=object)
```

Jika kita ingin mengetahui, secara rata-rata, jenis roda penggerak mana yang paling mahal, kita dapat mengelompokkan "roda penggerak" dan kemudian membuat rata-ratanya. Kita dapat memilih kolom 'drive-wheels', 'body-style' dan 'price', lalu menetapkannya ke variabel "df_group_one".

```
df_group_one = df[['drive-wheels','body-style','price']]
```

Kami kemudian dapat menghitung harga rata-rata untuk setiap kategori data yang berbeda.

```
# grouping results
df_group_one = df_group_one.groupby(['drive-
wheels'],as_index=False).mean()
df_group_one
```

	drive-wheels	price
0	4wd	10241.000000
1	fwd	9244.779661
2	rwd	19757.613333

Dari data kita, sepertinya kendaraan roda belakang rata-rata paling mahal, sedangkan penggerak 4 roda dan roda depan harganya kurang lebih sama. Anda juga dapat mengelompokkan dengan beberapa variabel. Misalnya, mari kita kelompokkan berdasarkan 'roda penggerak' dan body-style'. Ini mengelompokkan dataframe dengan kombinasi unik 'drive-wheels' dan 'body-style'. Kita dapat menyimpan hasilnya dalam variabel 'grouped_test1'.

```
# grouping results
df_gptest = df[['drive-wheels','body-style','price']]
grouped_test1 = df_gptest.groupby(['drive-wheels','body-
style'],as_index=False).mean()
grouped_test1
```

	drive-wheels	body-style	price
0	4wd	hatchback	7603.000000
1	4wd	sedan	12647.333333
2	4wd	wagon	9095.750000
3	fwd	convertible	11595.000000
4	fwd	hardtop	8249.000000
5	fwd	hatchback	8396.387755
6	fwd	sedan	9811.800000
7	fwd	wagon	9997.333333
8	rwd	convertible	23949.600000
9	rwd	hardtop	24202.714286
10	rwd	hatchback	14337.777778
11	rwd	sedan	21711.833333
12	rwd	wagon	16994.222222

Data yang dikelompokkan ini jauh lebih mudah untuk divisualisasikan ketika dibuat menjadi tabel pivot. Tabel pivot yang mirip seperti pada spreadsheet Excel, dengan satu variabel di sepanjang kolom dan variabel lainnya di sepanjang baris. Kita dapat mengonversi kerangka data menjadi tabel pivot menggunakan metode "pivot" untuk membuat tabel pivot dari grup. Dalam hal ini, kita akan membiarkan variabel drive-wheel sebagai baris tabel, dan pivot body-style menjadi kolom tabel:

```
grouped_pivot = grouped_test1.pivot(index='drive-wheels',columns='body-style')
grouped_pivot
```

	price				
body-style	convertible	hardtop	hatchback	sedan	wagon
drive-wheels					
4wd	NaN	NaN	7603.000000	12647.333333	9095.750000
fwd	11595.0	8249.000000	8396.387755	9811.800000	9997.333333
rwd	23949.6	24202.714286	14337.777778	21711.833333	16994.222222

Seringkali, kita tidak memiliki data untuk beberapa sel pivot. Kita dapat mengisi sel yang hilang ini dengan nilai 0, tetapi nilai lain apa pun berpotensi digunakan juga. Harus disebutkan bahwa data yang hilang adalah subjek yang cukup kompleks.

```
grouped_pivot = grouped_pivot.fillna(0) #fill missing values with 0
grouped_pivot
```

	price				
body-style	convertible	hardtop	hatchback	sedan	wagon
drive-wheels					
4wd	0.0	0.000000	7603.000000	12647.333333	9095.750000
fwd	11595.0	8249.000000	8396.387755	9811.800000	9997.333333
rwd	23949.6	24202.714286	14337.777778	21711.833333	16994.222222

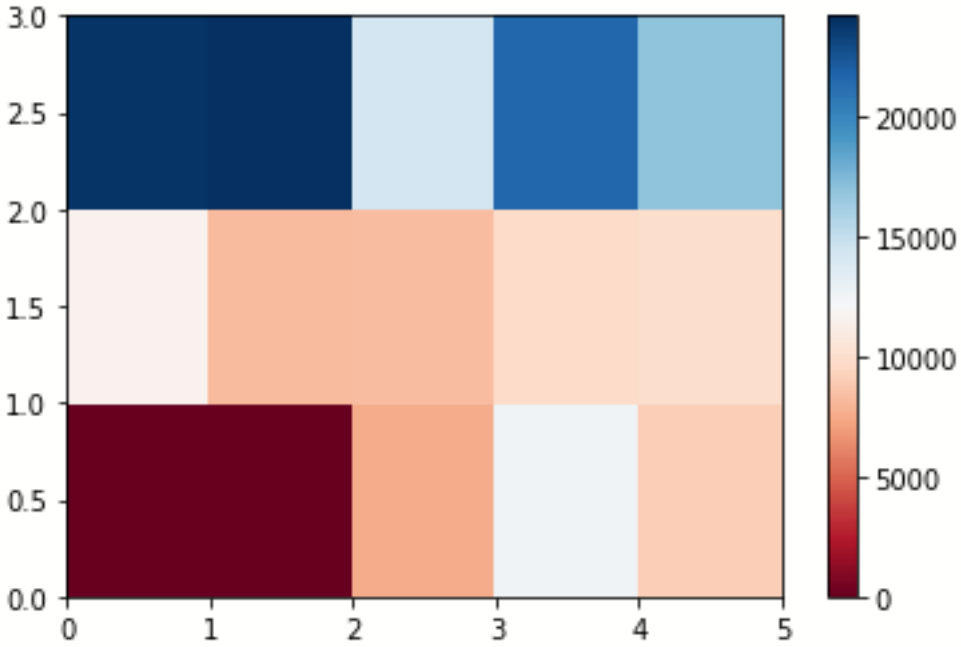
Gunakan fungsi "groupby" untuk mencari "harga" rata-rata setiap mobil berdasarkan "body-style" ?

```
# Write your code below and press Shift+Enter to execute
df_gptest2 = df[['body-style','price']]
grouped_test_bodystyle = df_gptest2.groupby(['body-
style'],as_index= False).mean()
grouped_test_bodystyle
```

	body-style	price
0	convertible	21890.500000
1	hardtop	22208.500000
2	hatchback	9957.441176
3	sedan	14459.755319
4	wagon	12371.960000

Heatmap Grouping
Hasil dari pivot dapat kita visualisasikan dalam bentuk heatmap dapat kita lihat pada Gambar 0.37

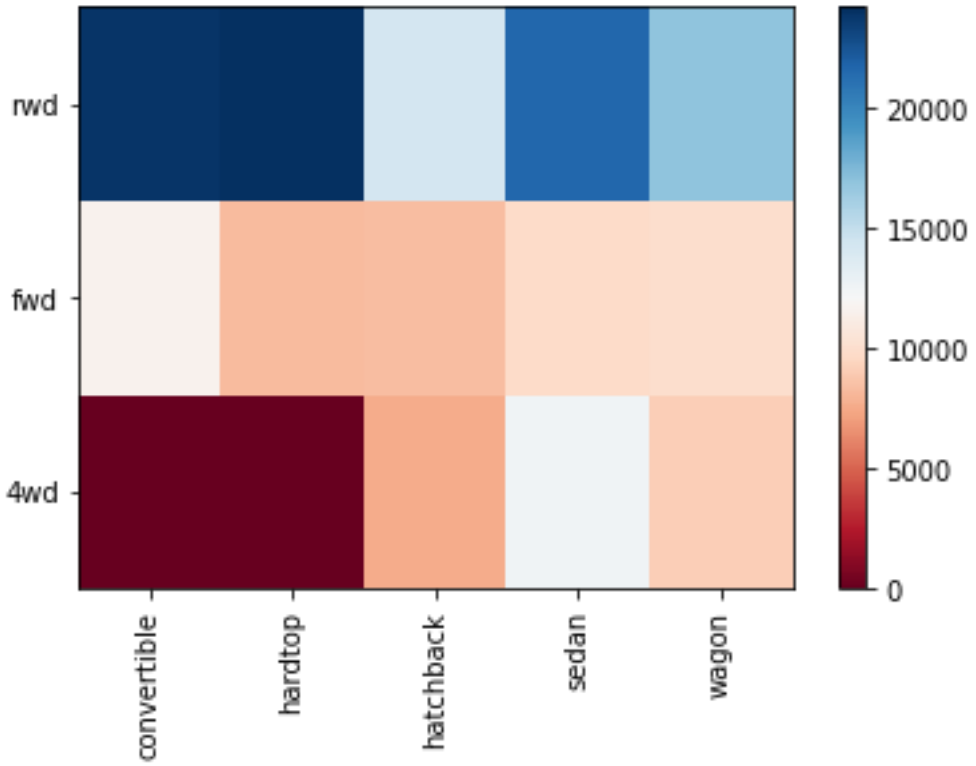
```
import matplotlib.pyplot as plt
#use the grouped results
plt.pcolor(grouped_pivot, cmap='RdBu')
plt.colorbar()
plt.show()
```



Gambar 0.37. Heatmap grouping dari data kendaraan (group by roda penggerak)

Heatmap memplot variabel target (harga) dengan variabel 'roda penggerak' dan 'body-style' disumbu vertikal dan horizontal. Hal ini memungkinkan kita untuk memvisualisasikan bagaimana harga terkait dengan 'drive-wheel' dan 'body-style'.
Label default belum menyampaikan informasi yang cukup kepada kita. Mari kita ubah label pada heatmap tersebut agar bisa memiliki informasi legend (Gambar 0.38):

```
fig, ax = plt.subplots()
im = ax.pcolor(grouped_pivot, cmap='RdBu')

#label names
row_labels = grouped_pivot.columns.levels[1]
col_labels = grouped_pivot.index

#move ticks and labels to the center
ax.set_xticks(np.arange(grouped_pivot.shape[1]) + 0.5, minor=False)
ax.set_yticks(np.arange(grouped_pivot.shape[0]) + 0.5, minor=False)

#insert labels
ax.set_xticklabels(row_labels, minor=False)
ax.set_yticklabels(col_labels, minor=False)

#rotate label if too long
plt.xticks(rotation=90)

fig.colorbar(im)
plt.show()
```



Gambar 0.38. Heatmap grouping dari data kendaraan + legend (group by roda penggerak)

Visualisasi sangat penting dalam data science, dan paket visualisasi Python memberikan kebebasan untuk dapat dikonfigurai. Pertanyaan utama yang ingin dijawab pada dataset ini, adalah "Apakah karakteristik utama yang paling berpengaruh terhadap harga mobil?".

ANOVA: Analysis of Variance

Analysis of Varians (ANOVA) adalah metode statistik yang digunakan untuk menguji apakah ada perbedaan yang signifikan antara rata-rata dua kelompok atau lebih. ANOVA mengembalikan dua parameter

F-Score: ANOVA mengasumsikan rata-rata semua kelompok adalah sama, anova akan menghitung seberapa jauh rata-rata yang sebenarnya menyimpang dari asumsi, dan melaporkannya sebagai F-Skore. Skor yang lebih besar berarti ada perbedaan yang lebih besar antara rata-rata.

P-Value: Nilai-P menunjukkan seberapa signifikan secara statistik nilai skor yang dihitung.

Jika variabel harga pada dataset mobil sangat berkorelasi dengan variabel lainya, ANOVA akan mengembalikan skor F-Score yang cukup besar dan nilai-p yang kecil.

ANOVA menganalisis perbedaan antara kelompok yang berbeda dari variabel yang sama, fungsi groupby akan berguna dalam kasus ANOVA.

Mari kita lihat apakah jenis 'roda penggerak' mempengaruhi 'harga',

```
grouped_test2=df_gptest[['drive-wheels', 'price']].groupby(['drive-wheels'])
grouped_test2.head(2)
```

	drive-wheels	price
0	rwd	13495.0
1	rwd	16500.0
3	fwd	13950.0
4	4wd	17450.0
5	fwd	15250.0
136	4wd	7603.0

```
df_gptest
```

	drive-wheels	body-style	price
0	rwd	convertible	13495.0
1	rwd	convertible	16500.0
2	rwd	hatchback	16500.0
3	fwd	sedan	13950.0
4	4wd	sedan	17450.0
...	...	...	...
196	rwd	sedan	16845.0
197	rwd	sedan	19045.0
198	rwd	sedan	21485.0
199	rwd	sedan	22470.0
200	rwd	sedan	22625.0

Kita dapat memperoleh nilai dari grup , method yang digunakan adalah "get_group".

```
grouped_test2.get_group('4wd')['price']
```

```
4      17450.0
136     7603.0
140     9233.0
141    11259.0
144     8013.0
145    11694.0
150     7898.0
151     8778.0
Name: price, dtype: float64
```

kita dapat menggunakan fungsi 'f_oneway' di modul 'stats' untuk mendapatkan F-Score dan P-Value

```
# ANOVA
```

```
f_val, p_val = stats.f_oneway(grouped_test2.get_group('fwd')['price'],
grouped_test2.get_group('rwd')['price'], grouped_test2.get_group('4wd')
['price'])

print( "ANOVA results: F=", f_val, ", P =", p_val)
```

```
 ANOVA results: F= 67.95406500780399 , P = 3.3945443577151245e-23
```

Hasil ANOVA ini termasuk hasil yang bagus, dengan F-Score yang besar menunjukkan korelasi yang kuat dan nilai P hampir 0 menyiratkan signifikansi statistik yang hampir pasti.

Tetapi apakah ini berarti ketiga kelompok yang diuji semuanya berkorelasi tinggi?

Separately: fwd and rwd

```
f_val, p_val = stats.f_oneway(grouped_test2.get_group('fwd')['price'],
grouped_test2.get_group('rwd')['price'])

print( "ANOVA results: F=", f_val, ", P =", p_val )
```

```
 ANOVA results: F= 130.5533160959111 , P = 2.2355306355677845e-2
```

4wd and fwd

```
f_val, p_val = stats.f_oneway(grouped_test2.get_group('4wd')['price'],
grouped_test2.get_group('fwd')['price'])

print("ANOVA results: F=", f_val, ", P =", p_val)
```

```
 ANOVA results: F= 0.665465750252303 , P = 0.41620116697845666
```

Latihan: Penggunaan Visualisasi

Terdapat 5 latihan mandiri pada modul ini. Setiap latihan berisi dataset. Anda boleh menggunakan Matplotlib atau Seaborn, buat visualisasi yang memungkinkan.
Untuk setiap latihan, pilih salah satu dari visualisasi berikut: * Pie Chart * Bar Chart * Line Chart * Scatter plot * Heatmatp. Anda hanya dapat menggunakan setiap visualisasi satu kali.
Gunakan penilaian Anda untuk memilih mana yang menurut Anda terbaik untuk setiap pertanyaan. Tambahkan judul, label, kode warna, dan alat bantu visual lainnya untuk membantu pengguna menafsirkan bagan.

Latihan 1: Bitcoin Price
Kita memiliki daftar harga Bitcoin yang dicatat setiap akhir minggu (Minggu) di 2018 dan 2019. Buat visualisasi yang memungkinkan Anda menjawab pertanyaan: Tahun apa, 2018 atau 2019, yang cenderung memberikan pengembalian yang lebih baik bagi pemegang Bitcoin?

Problem

```
prices = [14292.2, 12858.9, 11467.5, 9241.1, 8559.6, 11073.5,
9704.3, 11402.3,
          8762.0, 7874.9, 8547.4, 6938.2, 6905.7, 8004.4, 8923.1,
9352.4,
          9853.5, 8459.5, 8245.1, 7361.3, 7646.6, 7515.8, 6505.8,
6167.3,
          6398.9, 6765.5, 6254.8, 7408.7, 8234.1, 7014.3, 6231.6,
6379.1,
          6734.8, 7189.6, 6184.3, 6519.0, 6729.6, 6603.9, 6596.3,
6321.7,
          6572.2, 6494.2, 6386.2, 6427.1, 5621.8, 3920.4, 4196.2,
3430.4,
          3228.7, 3964.4, 3706.8, 3785.4, 3597.2, 3677.8, 3570.9,
3502.5,
          3661.4, 3616.8, 4120.4, 3823.1, 3944.3, 4006.4, 4002.5,
4111.8,
          5046.2, 5051.8, 5290.2, 5265.9, 5830.9, 7190.3, 7262.6,
8027.4,
          8545.7, 7901.4, 8812.5, 10721.7, 11906.5, 11268.0,
11364.9, 10826.7,
          9492.1, 10815.7, 11314.5, 10218.1, 10131.0, 9594.4,
10461.1, 10337.3,
          9993.0, 8208.5, 8127.3, 8304.4, 7957.3, 9230.6, 9300.6,
8804.5,
          8497.3, 7324.1, 7546.6, 7510.9, 7080.8, 7156.2, 7321.5,
7376.8]

# Solusi ??
```

Penjelasan anda
Chart apa yang anda pilih untuk problem diatas dan mengapa anda memilih chart tersebut?
Jawaban anda.
Tahun berapa pemegang bitcoin memiliki keuntungan yang lebih banyak?
Jawaban anda.

Latihan 2: Permen
Kita memiliki sekantong permen. Terdapat lima jenis permen, masing-masing diberi nama di bawah ini. Buat diagram yang menunjukkan persentase peluang bahwa kita akan mengeluarkan permen Snickers dari kantong jika kita melakukan pengambilan acak. Sebutkan peluang memilih permen Snickers.

Problem
```
candy_names = ['Kit Kat', 'Snickers', 'Milky Way', 'Toblerone',
'Twix']
candy_counts = [52, 39, 78, 13, 78]

# jawaban anda
```

Penjelasan anda
Chart apa yang anda pilih untuk problem diatas dan mengapa anda memilih chart tersebut?
Jawaban anda
Berapa persen kemungkinan Anda akan memilih Snickers saat mengeluarkan permen dari tas secara acak?
Jawaban anda

Latihan 3: Makanan
Restoran memiliki menu makanan penutup yang terlalu besar. Mereka ingin memotong beberapa item dari menu. Untuk membuat sebagian besar pelanggan mereka senang, mereka ingin menghapus hanya tiga makanan penutup yang paling tidak populer dari menu. Kita memiliki daftar makanan penutup yang disajikan restoran, serta hitungan berapa kali makanan penutup tersebut dijual dalam seminggu terakhir. Buat visualisasi yang menunjukkan popularitas relatif dari makanan penutup. Sebutkan tiga makanan penutup yang harus disingkirkan.

Problem
```
dessert_sales = {
  'Lava Cake': 14,
  'Mousse': 5,
  'Chocolate Cake': 12,
  'Ice Cream': 19,
```

```
  'Truffles': 6,
  'Brownie': 8,
  'Chocolate Chip Cookie': 12,
  'Chocolate Pudding': 9,
  'Souffle': 10,
  'Chocolate Cheesecake': 17,
  'Chocolate Chips': 2,
  'Fudge': 9,
  'Mochi': 13,
}

# jawaban anda
```

Penjelasan anda
Chart apa yang anda pilih untuk problem diatas dan mengapa anda memilih chart tersebut?
Jawaban anda
Makanan penutup apa saja yang perlu anda sarankan untuk dikeluarkan dari menu?
Jawaban anda

Latihan 4: Penggunaan CPU
Kita memiliki penggunaan CPU rata-rata per jam untuk komputer pekerja selama seminggu. Setiap baris data mewakili satu hari dalam seminggu yang dimulai dengan Senin. Setiap kolom data adalah satu jam dalam sehari dimulai dengan 0 menjadi tengah malam.

Buat bagan yang menunjukkan penggunaan CPU selama seminggu. Anda harus dapat menjawab pertanyaan-pertanyaan berikut menggunakan bagan:
• Jam berapa pekerja biasanya makan siang?
• Apakah pekerja tersebut bekerja pada akhir pekan?
• Pada hari apa pekerja mulai bekerja pada komputer mereka pada malam hari?

Student Solution
```
cpu_usage = [
  [2, 2, 4, 2, 4, 1, 1, 4, 4, 12, 22, 23,
   45, 9, 33, 56, 23, 40, 21, 6, 6, 2, 2, 3], # Monday
  [1, 2, 3, 2, 3, 2, 3, 2, 7, 22, 45, 44,
   33, 9, 23, 19, 33, 56, 12, 2, 3, 1, 2, 2], # Tuesday
  [2, 3, 1, 2, 4, 4, 2, 2, 1, 2,  5, 31,
   54, 7, 6, 34, 68, 34, 49, 6, 6, 2, 2, 3], # Wednesday
  [1, 2, 3, 2, 4, 1, 2, 4, 1, 17, 24, 18,
   41, 3, 44, 42, 12, 36, 41, 2, 2, 4, 2, 4], # Thursday
  [4, 1, 2, 2, 3, 2, 5, 1, 2, 12, 33, 27,
   43, 8, 38, 53, 29, 45, 39, 3, 1, 1, 3, 4], # Friday
  [2, 3, 1, 2, 2, 5, 2, 8, 4, 2, 3,
   1, 5, 1, 2, 3, 2, 6, 1, 2, 2, 1, 4, 3], # Saturday
```

```
  [1, 2, 3, 1, 1, 3, 4, 2, 3, 1, 2,
   2, 5, 3, 2, 1, 4, 2, 45, 26, 33, 2, 2, 1], # Sunday
]

# Jawaban anda
```

Penjelasan anda

Chart apa yang anda pilih untuk problem diatas dan mengapa anda memilih chart tersebut?

Jawaban anda

Jam berapa pekerja biasanya makan siang?

Jawaban anda

Apakah pekerja tersebut bekerja pada akhir pekan?

Jawaban anda

Pada hari apa pekerja mulai bekerja pada komputer mereka pada malam hari?

Jawaban anda

Latihan 5: Jamur

Seorang peneliti sedang mempelajari jamur. Mereka telah menemukan cincin jamur dan memberi label koordinat. Biasanya jamur menyebar keluar dari pusat jamur awal. Dengan koordinat di bawah ini, peneliti ingin menjawab pertanyaan: Kira-kira di manakah letak pusat pertumbuhan jamur? Buat bagan yang memungkinkan peneliti memperkirakan pusat pertumbuhan.

Problem

```
x = [4.61, 5.08, 5.18, 7.82, 10.46, 7.66, 7.6, 9.32, 14.04, 9.95,
4.95, 7.23,
     5.21, 8.64, 10.08, 8.32, 12.83, 7.51, 7.82, 6.29, 0.04, 6.62,
13.16, 6.34,
     0.09, 10.04, 13.06, 9.54, 11.32, 7.12, -0.67, 10.5, 8.37, 7.24,
9.18,
     10.12, 12.29, 8.53, 11.11, 9.65, 9.42, 8.61, -0.67, 5.94, 6.49,
7.57, 3.11,
     8.7, 5.28, 8.28, 9.55, 8.33, 13.7, 6.65, 2.4, 3.54, 9.19, 7.51,
-0.68,
     8.47, 14.82, 5.31, 14.01, 8.75, -0.57, 5.35, 10.51, 3.11, -
0.26, 5.74,
     8.33, 6.5, 13.85, 9.78, 4.91, 4.19, 14.8, 10.04, 13.47, 3.28]

y = [-2.36, -3.41, 13.01, -2.91, -2.28, 12.83, 13.13, 11.94, 0.93, -
2.76, 13.31,
     -3.57, -2.33, 12.43, -1.83, 12.32, -0.42, -3.08, -2.98, 12.46,
8.34, -3.19,
     -0.47, 12.78, 2.12, -2.72, 10.64, 11.98, 12.21, 12.52, 5.53,
11.72, 12.91,
     12.56, -2.49, 12.08, -1.09, -2.89, -1.78, -2.47, 12.77, 12.41,
5.33, -3.23,
     13.45, -3.41, 12.46, 12.1, -2.56, 12.51, -2.37, 12.76, 9.69,
```

```
12.59, -1.12,
     -2.8, 12.94, -3.55, 7.33, 12.59, 2.92, 12.7, 0.5, 12.57, 6.39,
12.84,
     -1.95, 11.76, 6.82, 12.44, 13.28, -3.46, 0.7, -2.55, -2.37,
12.48, 7.26,
     -2.45, 0.31, -2.51]

# jawaban anda
```

Penjelasan anda

Chart apa yang anda pilih untuk problem diatas dan mengapa anda memilih chart tersebut?

Jawaban anda

Koordinat pusat (x,y) pusat pertumbuhan jamur berada di?

Jawaban anda

References:

Tugas Dan Proyek Pelatihan
Latihan pada Bab.7.6

Link Referensi Modul Tujuh
1. Google LLC, Google Colabs Documentation, 2020, Machine Learning- Data Visualizations 2. Pandas Histogram - DataFrame.hist() - Data Independent, https://www.dataindependent.com/pandas/pandas-histogram/ 3. Course | DA0101EN | Cognitive Class, https://courses.cognitiveclass.ai/courses/course-v1:CognitiveClass+DA0101EN+2017/course/

Link Pertanyaan Modul Tujuh

Bahan Tayang
Power Point

Link room Pelatihan dan Jadwal live sesi bersama instruktur
Zoom

Penilaian
Komposisi penilaian Tugas Data Understanding 2: Nilai 100

Target Penyelesaian Modul Tujuh
1 hari / sampai 6 JP

KOMINFO

Badan Penelitian dan Pengembangan SDM
Kementerian Komunikasi dan Informatika