Projekt

Najpopularniejsze języki programowania

Wizualizacja danych

Aleksandra Femin

Spis treści

1. Wprowadzenie

W dzisiejszym świecie programowania, zrozumienie trendów i zmian w popularności języków programowania ma kluczowe znaczenie dla programistów, firm technologicznych i osób zainteresowanych technologią. Ten projekt ma na celu szczegółową analizę i wizualizację danych dotyczących popularności wybranych języków programowania na przestrzeni lat. Skupimy się głównie na języku Python, ale również porównamy go z innymi popularnymi językami takimi jak Java, C/C++, PHP.

2. Cel projektu

Celem projektu jest:

1) Analiza trendów: Analiza zmian popularności poszczególnych języków programowania na przestrzeni lat pozwoli zidentyfikować rozwijające się trendy w świecie programowania. Pozwoli to lepiej zrozumieć, które języki zyskują na popularności, a które tracą.

2) Porównanie popularności: Porównanie popularności języka Python z innymi językami programowania pomoże zrozumieć jego pozycję w stosunku do konkurencji. W szczególności zbadamy, czy Python rzeczywiście rośnie w popularności w porównaniu z innymi językami, takimi jak Java czy C/C++.

3) Wizualizacja danych: Przedstawienie danych w formie czytelnych i atrakcyjnych wizualizacji ułatwi zrozumienie analizy i pozwoli na szybkie wnioskowanie na temat trendów w popularności języków programowania.

3. Opis użytych danych

Dane wykorzystane w tym projekcie pochodzą ze zbioru udostępnionego na platformie https://www.kaggle.com/. Zbiór ten zawiera informacje na temat popularności różnych języków programowania na przestrzeni lat, przy czym dane są agregowane miesięcznie.

- Plik ten zawiera 29 kolumn.
- Kolumna 1 zawiera daty.
- Kolumny 2-29 zawierają procentowy udział popularności języka w danym miesiącu.

4. Wczytywanie i przetwarzanie danych

Na początku wczytujemy dane z pliku CSV, konwertujemy kolumnę "Date" na format rok oraz grupujemy dane po latach, obliczając średnią. Poniższy kod realizuje te zadania:

```
# Importowanie bibliotek
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.graph_objects as go
from IPython.display import display

# Wczytywanie danych z pliku CSV
def load_data(file_name):
    data = pd.read_csv(file_name, skiprows=[1, 2, 3, 4, 5, 6, 211])
    data["Date"] = pd.to_datetime(data["Date"]).dt.year
    data = data.groupby("Date").mean().reset_index()
    return data

# Użycie funkcji load_data
data = load_data("Most_Popular_Programming_Languages_from_2004_to_2022.csv")

# Wyświetlenie pierwszych kilku wierszy danych
print(data.head())

# Wyświetlenie całego zestawu danych w postaci tabeli
display(data)

# Eksport danych do pliku Excel
data.to_excel("dane.xlsx", index=False)

display(data1)
```

	Date	Abap	Ada	C/C++	C#	Cobol	Dart	Delphi/Pascal	Go	Groovy	...	PHP	Python	R	Ruby	Rust	Scala	Swift	TypeScript	VBA	Visual Basic
0	2005	0.358333	0.336667	9.300833	5.850833	0.440000	0.000000	2.500000	0.000000	0.082500	...	19.684167	3.132500	0.439167	0.682500	0.135833	0.017500	0.000000	0.000000	1.547500	7.432500
1	2006	0.363333	0.252500	8.405000	6.698333	0.438333	0.000000	2.098333	0.000000	0.094167	...	20.142500	3.829167	0.535000	1.947500	0.094167	0.022500	0.000000	0.004167	1.601667	6.326667
2	2007	0.370833	0.240833	8.080000	6.900000	0.360833	0.000000	1.781667	0.000000	0.146667	...	20.166667	4.354167	0.637500	2.698333	0.088333	0.056667	0.000000	0.000833	1.649167	5.462500
3	2008	0.375000	0.253333	8.064167	7.602500	0.410833	0.000000	1.750000	0.000000	0.250833	...	19.506667	5.242500	0.830833	2.862500	0.076667	0.136667	0.000000	0.000000	1.721667	5.242500
4	2009	0.434167	0.201667	9.155833	7.272500	0.427500	0.000000	1.555000	0.000000	0.286667	...	18.945833	6.537500	1.030000	2.644167	0.095000	0.179167	0.000000	0.000000	2.027500	5.422500
5	2010	0.575833	0.187500	11.458333	6.386667	0.520833	0.000000	1.276667	0.027500	0.341667	...	17.752500	6.379167	1.197500	2.359167	0.093333	0.220000	0.000000	0.000000	2.425000	5.312500
6	2011	0.577500	0.214167	12.620000	6.300833	0.455833	0.060833	1.063333	0.025000	0.314167	...	16.380833	7.070000	1.350000	2.580000	0.060000	0.277500	0.000000	0.000000	2.067500	4.793333
7	2012	0.662500	0.238333	9.580000	8.358333	0.379167	0.124167	0.886667	0.054167	0.385833	...	15.141667	8.135833	1.659167	2.659167	0.075000	0.377500	0.000000	0.004167	2.242500	3.902500
8	2013	0.610000	0.242500	8.210000	9.760833	0.322500	0.123333	0.734167	0.100000	0.385833	...	14.088333	9.447500	1.993333	2.744167	0.084167	0.485833	0.000000	0.082500	1.709167	3.304167
9	2014	0.610000	0.300000	8.028333	9.651667	0.333333	0.166667	0.655000	0.164167	0.457500	...	12.845833	10.223333	2.305833	2.610000	0.230833	0.675833	0.801667	0.110000	1.645000	2.707500
10	2015	0.581667	0.346667	7.790833	9.307500	0.327500	0.123333	0.533333	0.235833	0.468333	...	11.725000	11.291667	2.834167	2.600833	0.205000	0.814167	2.745000	0.175000	1.549167	2.222500
11	2016	0.472500	0.370000	7.566667	9.047500	0.288333	0.115833	0.412500	0.360000	0.402500	...	10.882500	13.461667	3.347500	2.360833	0.267500	1.027500	3.424167	0.560000	1.479167	1.752500
12	2017	0.480833	0.445000	7.133333	8.279167	0.339167	0.126667	0.351667	0.605000	0.445833	...	9.320833	17.415833	3.993333	2.035833	0.365833	1.275000	3.479167	1.200833	1.441667	1.435000
13	2018	0.530000	0.339167	6.213333	7.740833	0.328333	0.162500	0.266667	0.846667	0.492500	...	7.819167	22.715833	4.092500	1.651667	0.360000	1.198333	2.805833	1.482500	1.416667	1.202500
14	2019	0.534167	0.355000	5.934167	7.290000	0.312500	0.313333	0.255833	1.135833	0.488333	...	6.719167	27.493333	3.875833	1.406667	0.560000	1.107500	2.421667	1.704167	1.330000	1.060833
15	2020	0.470833	0.460000	5.858333	6.811667	0.364167	0.506667	0.257500	1.350833	0.409167	...	6.056667	31.012500	3.901667	1.210000	0.878333	0.945000	2.290000	1.877500	1.283333	0.841667
16	2021	0.527500	0.620000	6.689167	7.034167	0.334167	0.568333	0.078333	1.414167	0.434167	...	6.262500	30.290000	3.835000	1.089167	0.871667	0.550833	1.770833	1.696667	1.259167	0.693333

17 rows × 29 columns

Rysunek 1 Zbiór danych

5. Wizualizacja danych

5.1 Wykorzystane biblioteki

- **Matplotlib:** Jest to jedna z najpopularniejszych bibliotek do wizualizacji danych w języku Python. Oferuje szeroki zakres funkcji do tworzenia różnorodnych wykresów, takich jak wykresy liniowe, słupkowe, punktowe, histogramy itp. Matplotlib daje użytkownikowi pełną kontrolę nad wyglądem i stylem wykresów.

- **Seaborn:** Seaborn to biblioteka oparta na Matplotlib, która zapewnia interfejs wysokiego poziomu do tworzenia atrakcyjnych i informatywnych wykresów statystycznych. Jest szczególnie przydatna do eksploracyjnej analizy danych oraz prezentacji rozkładów i relacji między zmiennymi.

- **Plotly:** Plotly to narzędzie do tworzenia interaktywnych i dynamicznych wykresów. Umożliwia tworzenie interaktywnych wizualizacji danych, które można łatwo dostosowywać i eksplorować. Jest często wykorzystywane do prezentacji danych na stronach internetowych i w aplikacjach.

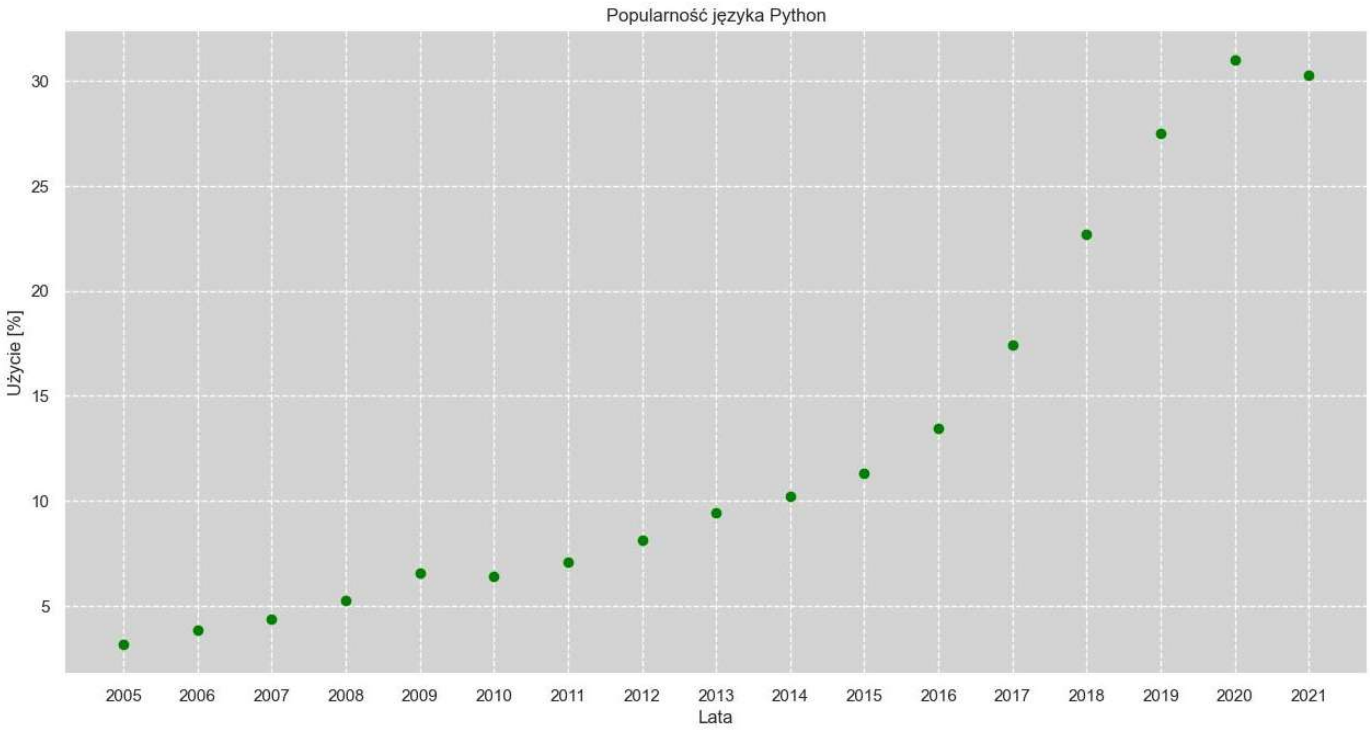
W sprawozdaniu wykorzystane są te biblioteki do generowania różnych rodzajów wykresów, takich jak wykresy punktowe, liniowe, słupkowe, histogramy i wykresy kołowe. Dzięki nim możemy lepiej zrozumieć dynamikę zmian w popularności języków programowania oraz porównać ich udziały w różnych okresach czasu. Dodatkowo, interaktywne wykresy Plotly umożliwiają bardziej zaawansowaną analizę danych poprzez eksplorację danych przy użyciu interakcji z wykresem.

5.2 Wykresy

1) Scatter Plot - Popularność języka Python w kolejnych latach

Wykres punktowy przedstawiający zmiany popularności języka Python w kolejnych latach.

```
 # Scatter Plot
    plt.figure(figsize=(12, 6))
    plt.scatter(data['Date'], data['Python'],color='green',
marker='o',zorder=2)
    plt.title("Popularność języka Python")
    plt.xlabel('Lata')
    plt.ylabel('Użycie [%]')
    plt.grid(True, linestyle='--', color='white')  # Dodanie białych linii
siatki
    plt.gca().set_facecolor('lightgray')  # Ustawienie szarego tła
    plt.xticks(sorted(data['Date'].unique()))  # Unikalne lata na osi x
    plt.show()
```

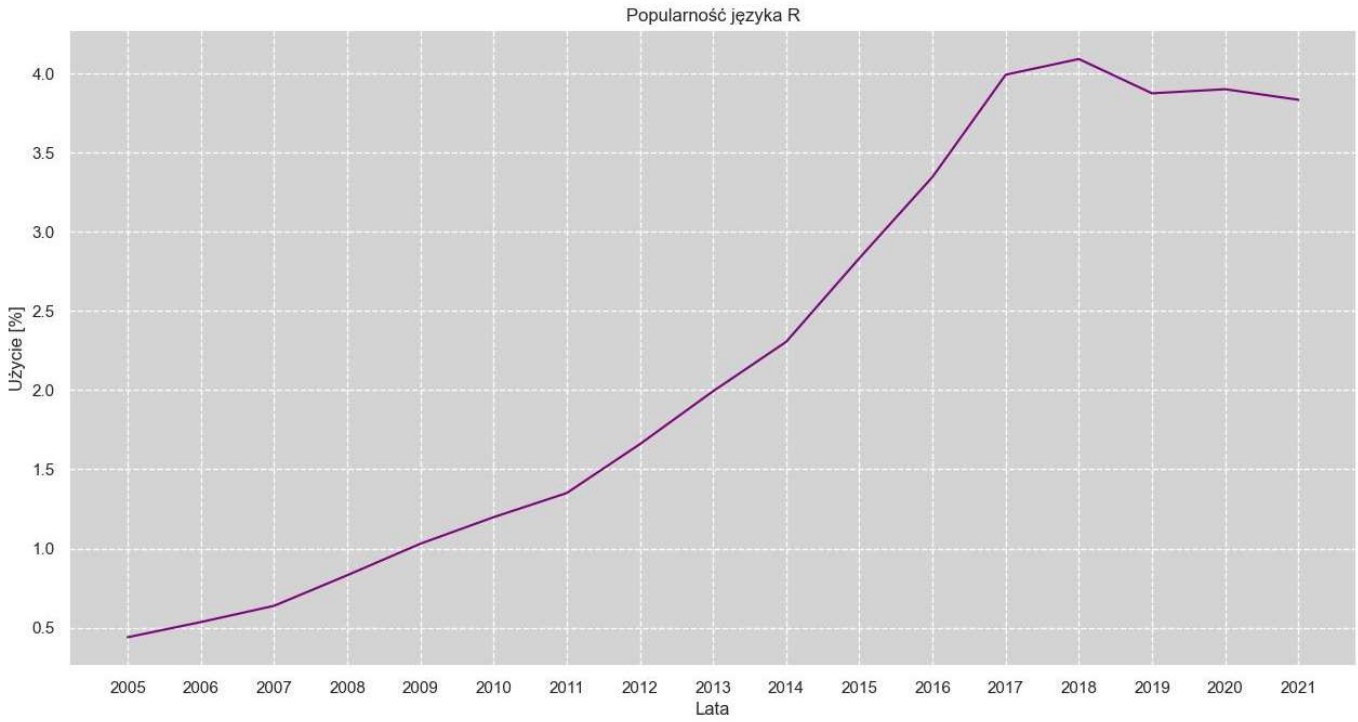


Na wykresie punktowym przedstawiającym popularność języka Python w zależności od lat, możemy zaobserwować wzrost jego używania w miarę upływu czasu. Kolor zielony symbolizuje język Python, natomiast każdy punkt reprezentuje określony rok, gdzie na osi X mamy lata, a na osi Y procentowy udział użycia języka Python w danym roku. Dzięki temu wykresowi możemy łatwo dostrzec trend rosnącej popularności Pythona w ciągu ostatnich lat. Dodatkowo, białe linie siatki oraz szare tło nadają wykresowi czytelny i estetyczny wygląd.

2) Line Chart - Popularność języka R w kolejnych latach

Wykres liniowy przedstawiający zmiany popularności języka R w kolejnych latach.

```
# Line Chart
    plt.figure(figsize=(12, 6))
    plt.plot(data['Date'], data['R'], color="purple",zorder=2)
    plt.title("Popularność języka R")
    plt.xlabel('Lata')
    plt.ylabel('Użycie [%]')
    plt.grid(True, linestyle='--', color='white')
    plt.gca().set_facecolor('lightgray')
    plt.xticks(sorted(data['Date'].unique()))
    plt.show()
```

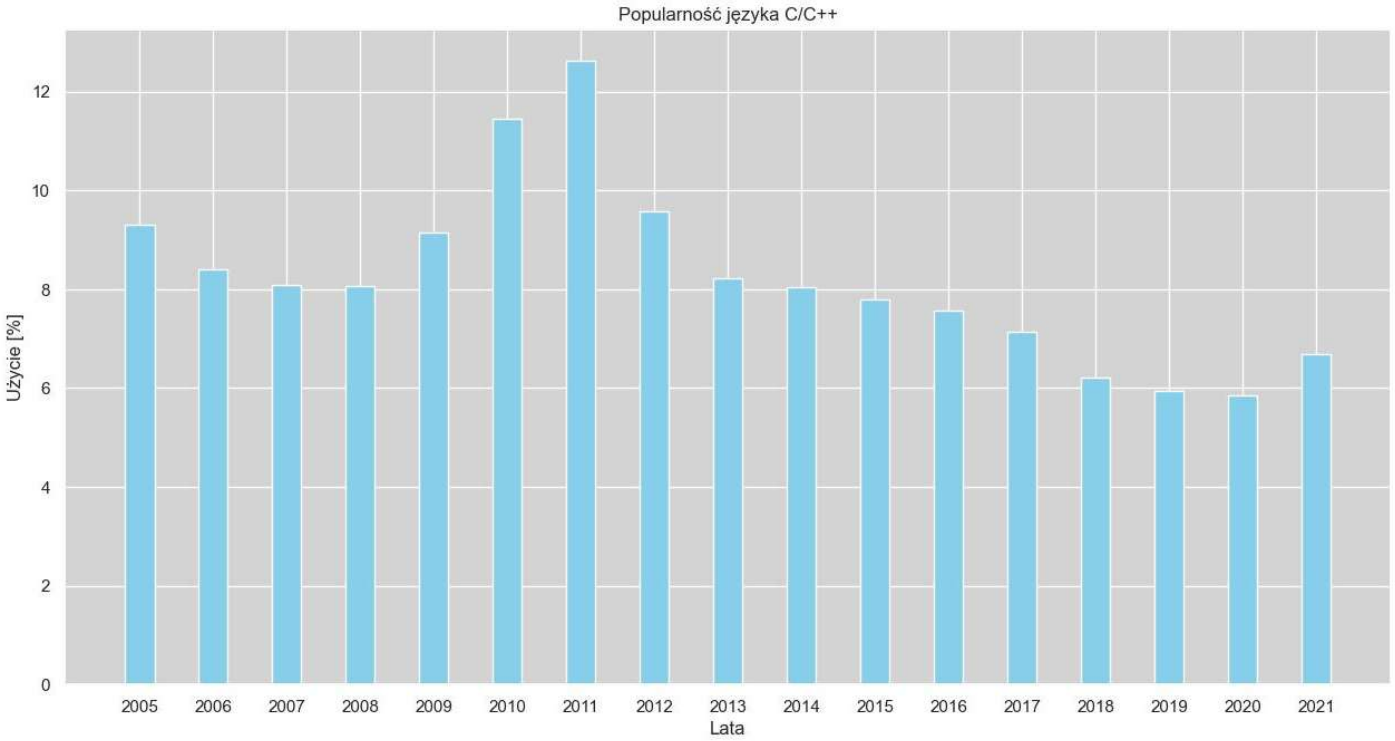


Wykres linowy pokazuje, że popularność języka R oscyluje wokół określonego poziomu w badanym okresie, zauważalny wzrost popularności.

3) Bar Chart - Popularność języka C/C++ w kolejnych latach

Wykres słupkowy przedstawiający zmiany popularności języka C/C++ w kolejnych latach.

```
# Bar Chart
    plt.figure(figsize=(12, 6))
    plt.bar(data['Date'], data['C/C++'], color ='skyblue', width=0.4,zorder=2)
    plt.title("Popularność języka C/C++")
    plt.xlabel('Lata')
    plt.ylabel('Użycie [%]')
    plt.grid(True, linestyle='-', color='white')
    plt.gca().set_facecolor('lightgray')
    plt.xticks(sorted(data['Date'].unique()))
    plt.show()
```

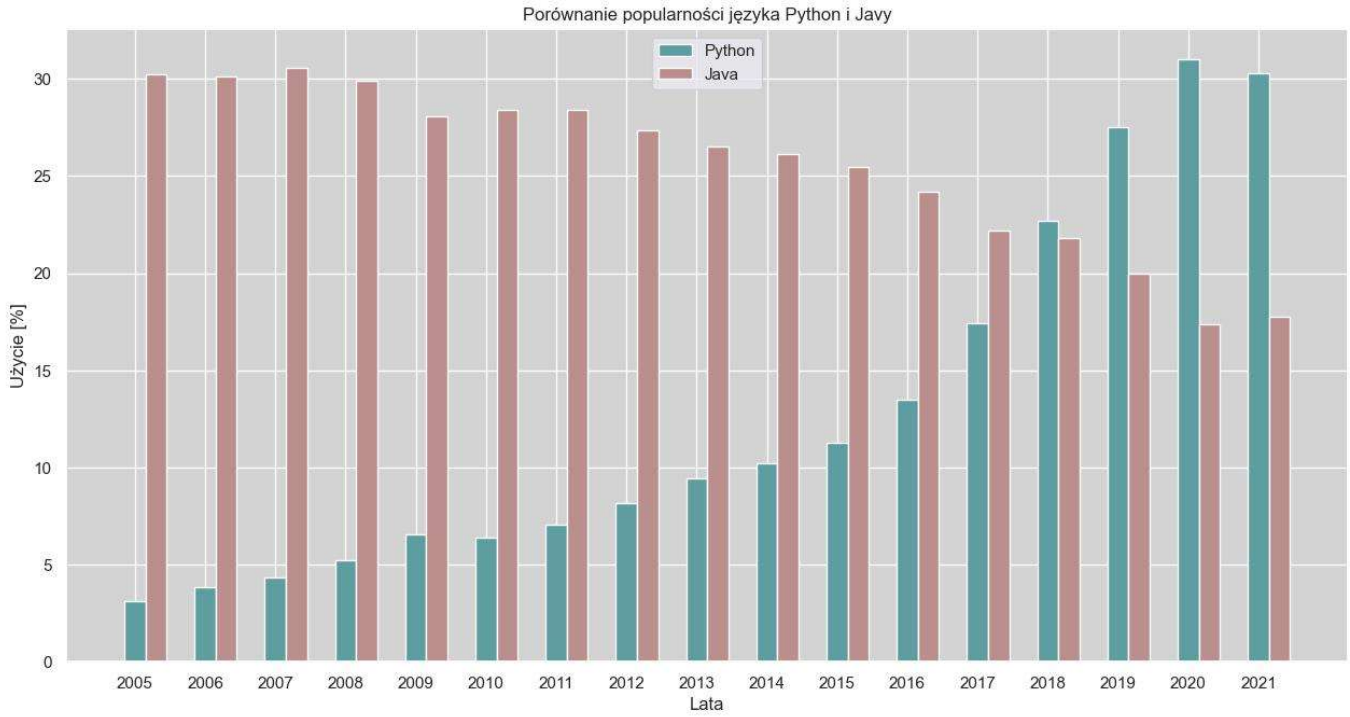


Wykres słupkowy pokazuje stabilność popularności języka C/C++ w badanym okresie.

4) Bar Chart - Porównanie Pythona i Javy w kolejnych latach

Wykres słupkowy porównujący popularność języka Python i Javy w kolejnych latach.

```
# Porównanie Pythona i Javy
    plt.figure(figsize=(12, 6))
    plt.bar(data['Date'], data['Python'], color ='cadetblue', width=0.3,
label="Python",zorder=2)
    plt.bar(data['Date'] + 0.3, data['Java'], color ='rosybrown',
width=0.3, label="Java",zorder=2)
    plt.title("Porównanie popularności języka Python i Javy")
    plt.xlabel('Lata')
    plt.ylabel('Użycie [%]')
    plt.legend()
    plt.grid(True, linestyle='-', color='white')
    plt.gca().set_facecolor('lightgray')
    plt.xticks(sorted(data['Date'].unique()))
    plt.show()
```



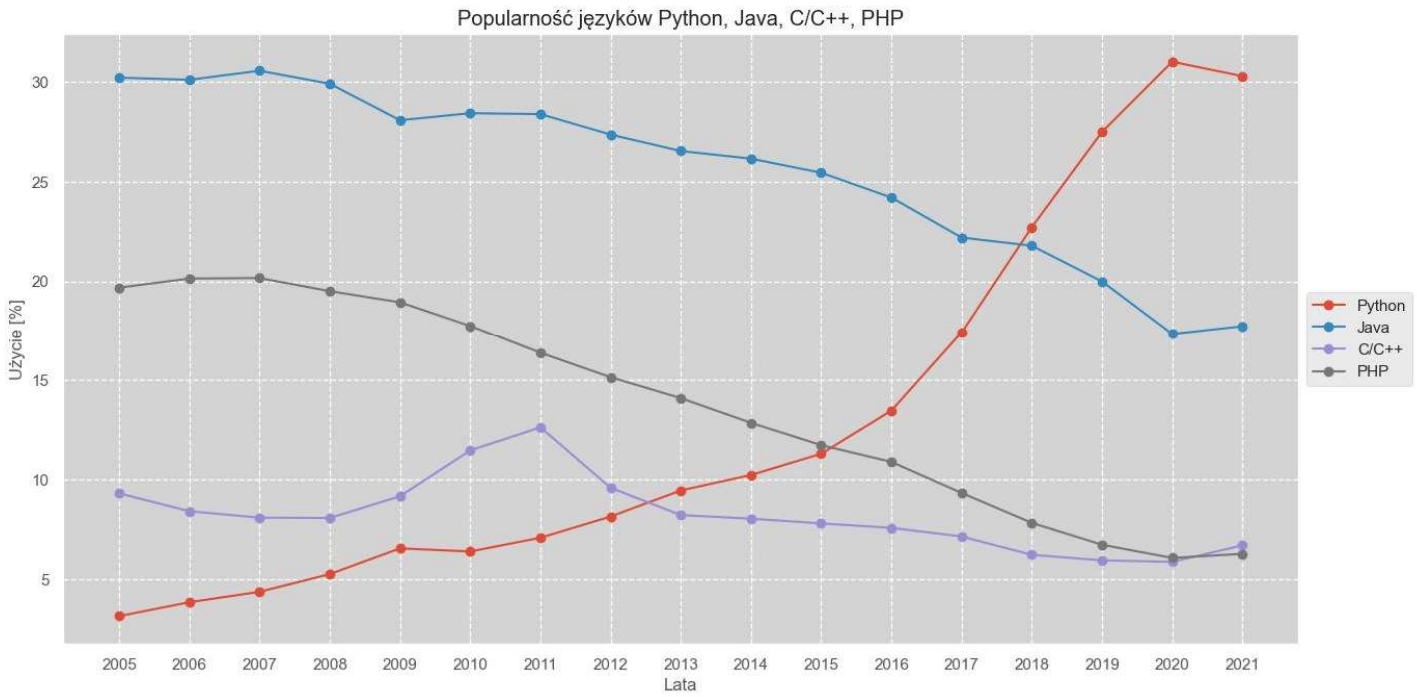
Wykres porównawczy pokazuje, że Python zyskuje na popularności w stosunku do Javy w badanym okresie.

5) Bar Chart - Popularność języków w 2021 roku

Wykres słupkowy przedstawiający udziały poszczególnych języków programowania w 2021 roku.

```
    # Line chart
    # Plotting multiple graphs

def plot_with_matplotlib(data):
    """
    Wyświetla wykres liniowy przedstawiający popularność różnych języków
programowania
    (Python, Java, C/C++, PHP) w kolejnych latach.
    """
    plt.figure(figsize=(12,6))
    plt.style.use('ggplot')

    plt.plot(data['Date'], data['Python'], marker="o", label="Python")
    plt.plot(data['Date'], data['Java'], marker="o", label="Java")
    plt.plot(data['Date'], data['C/C++'], marker="o", label="C/C++")
    plt.plot(data['Date'], data['PHP'], marker="o", label="PHP")
    plt.xticks(data['Date'])

    plt.title("Popularność języków Python, Java, C/C++, PHP")
    plt.xlabel('Lata')
    plt.ylabel('Użycie [%]')
```

```
    # Dodanie legendy z boku wykresu
    plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))

    # Dodanie szarego tła
    plt.gca().set_facecolor('lightgray')

    # Dodanie białych linii siatki
    plt.grid(True, linestyle='--', color='white')

    plt.show()
```

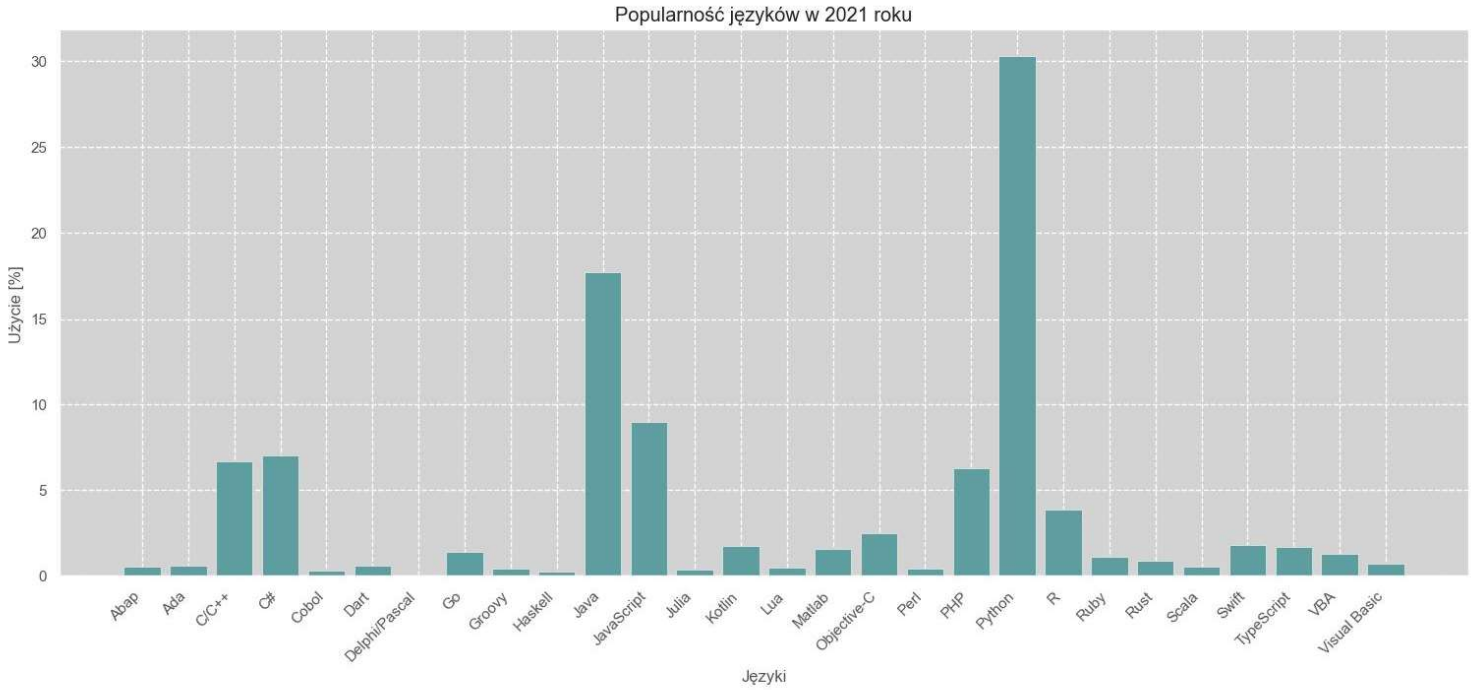


6) Wykres słupkowy popularności języków w 2021 roku

Wykres słupkowy przedstawia udział procentowy najpopularniejszych języków programowania w 2021 roku. Każdy słupek reprezentuje jeden język, a jego wysokość odpowiada jego udziałowi procentowemu. Oś X przedstawia nazwy języków, natomiast oś Y reprezentuje procentowe użycie. Wykres umożliwia szybkie porównanie popularności różnych języków w danym roku.

```
def plot_with_matplotlib(data):
    # Wykres słupkowy wszystkich języków w 2021 roku
    plt.figure(figsize=(12, 6))
    plt.bar(data.columns[1:], data[data['Date'] == 2021].values[0][1:], color ='cadetblue')

    plt.title("Popularność języków w 2021 roku")
    plt.xlabel('Języki')
    plt.ylabel('Użycie [%]')
```

```
    # Dodanie szarego tła
    plt.gca().set_facecolor('lightgray')

    # Dodanie białych linii siatki
    plt.grid(True, linestyle='--', color='white')

    plt.xticks(rotation=45, ha='right')  # Obrócenie etykiet na osi x
    plt.tight_layout()  # Dopasowanie układu, aby uniknąć przycięcia etykiet

    plt.show()
```

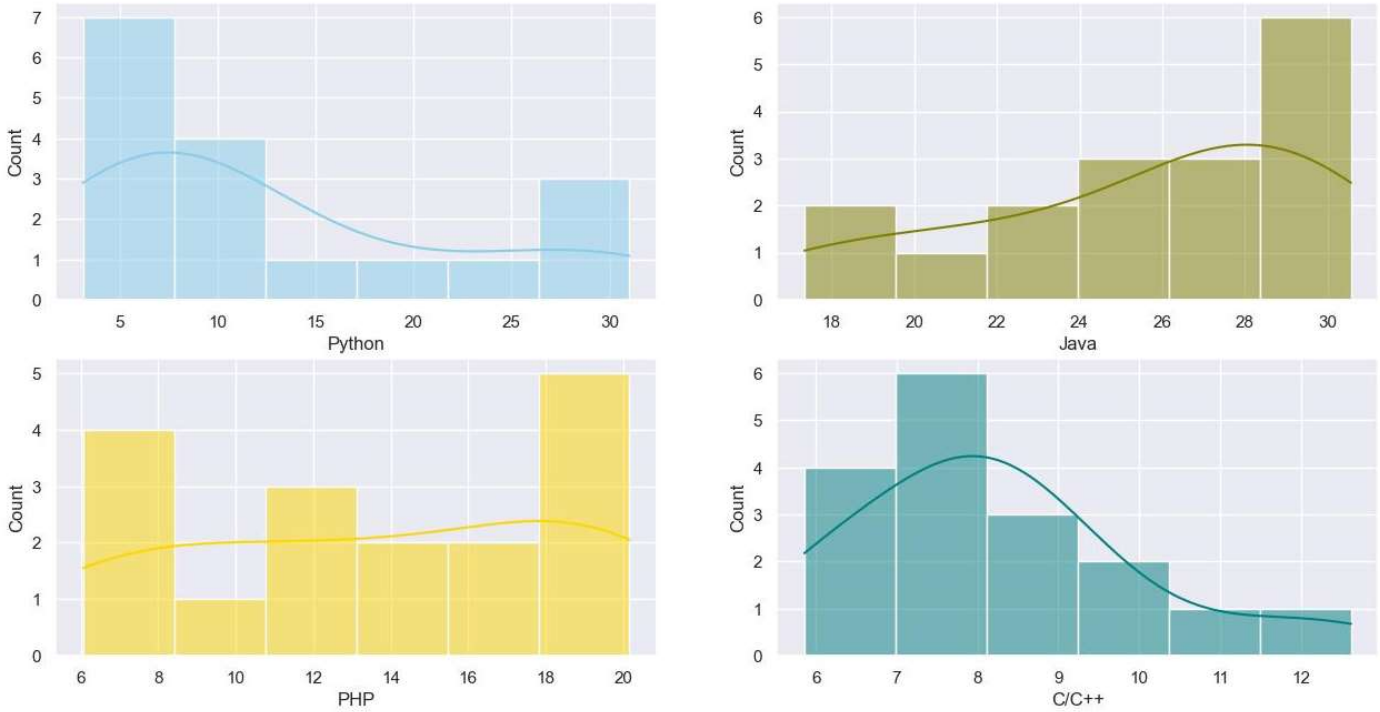


Wykres słupkowy pozwala na szybkie porównanie udziału procentowego różnych języków programowania w roku 2021, co umożliwia identyfikację najbardziej popularnych języków w danym okresie.

7) Histogram - rozkłady popularności języków Python, Java, PHP oraz C/C++

Każdy z histogramów przedstawia dystrybucję użycia danego języka programowania w zestawieniu z innymi językami. Linia krzywej gęstości (KDE) na histogramach pomaga zobaczyć ogólny kształt rozkładu danych.

```
 sns.set(style="darkgrid")
    fig, axs = plt.subplots(2, 2, figsize=(15, 15))
    sns.histplot(data=data, x="Python", kde=True, color="skyblue", ax=axs[0,
0])
    sns.histplot(data=data, x="Java", kde=True, color="olive", ax=axs[0, 1])
    sns.histplot(data=data, x="PHP", kde=True, color="gold", ax=axs[1, 0])
    sns.histplot(data=data, x="C/C++", kde=True, color="teal", ax=axs[1, 1])
    plt.show()
```

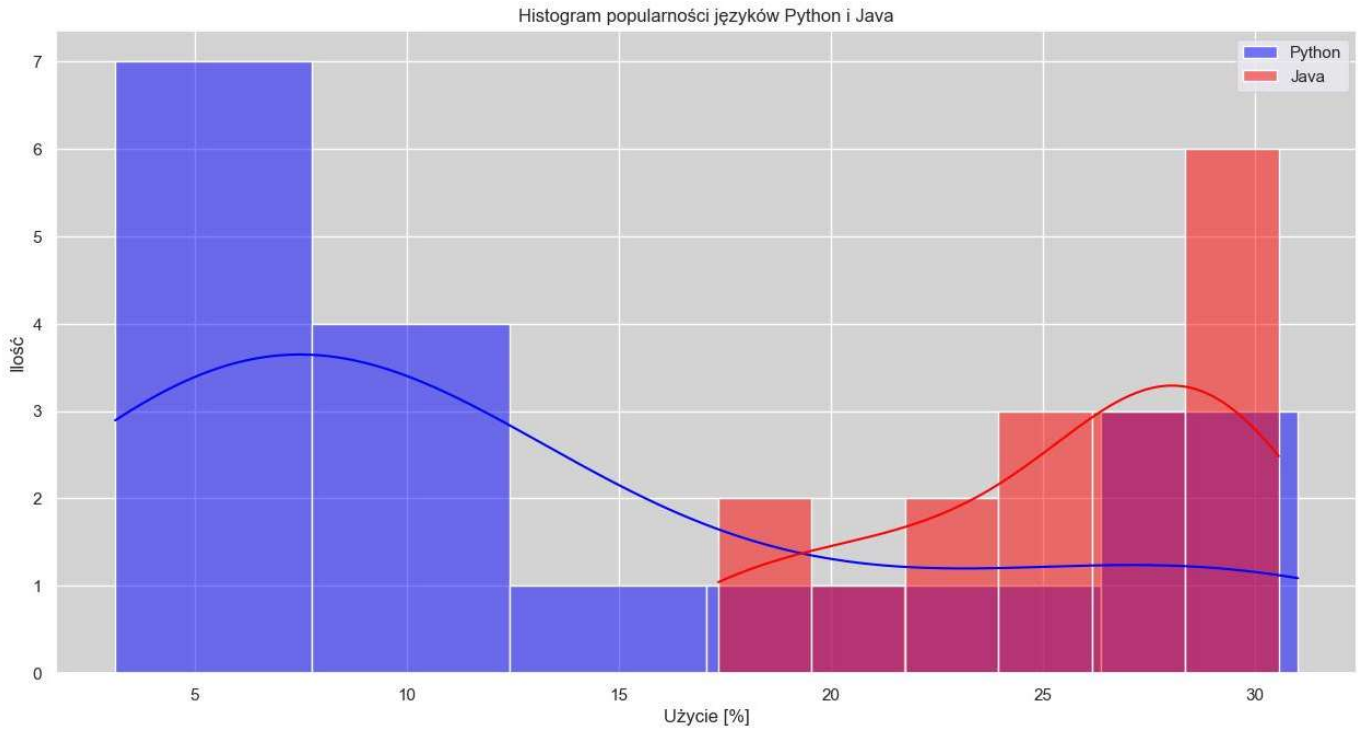


Te histogramy pozwalają na zrozumienie rozkładu popularności różnych języków programowania. Porównując histogramy dla różnych języków, można zidentyfikować różnice w ich rozkładach popularności, co może wskazywać na preferencje i tendencje wśród programistów. Dodatkowo, krzywe KDE pomagają zobaczyć estymowaną gęstość prawdopodobieństwa dla różnych wartości procentowych, co umożliwia lepsze zrozumienie kształtu rozkładu danych.

8) Histogram popularności języków Python i Java

Histogramy porównujące dystrybucję popularności języków Python i Java wśród programistów. Każdy histogram przedstawia procentowy udział użycia danego języka w projektach programistycznych. Linie wygładzone (KDE) pokazują estymowaną gęstość rozkładu, co umożliwia lepsze zrozumienie kształtu rozkładu danych.

```
plt.figure(figsize=(12, 6))
    sns.histplot(data=data, x="Python", color="blue", label="Python",
kde=True)
    sns.histplot(data=data, x="Java", color="red", label="Java", kde=True)
    plt.title("Histogram popularności języków Python i Java")
    plt.xlabel('Użycie [%]')
    plt.ylabel('Ilość')
    plt.legend()
    plt.grid(True, linestyle='-', color='white')
    plt.gca().set_facecolor('lightgray')
    plt.show()
```

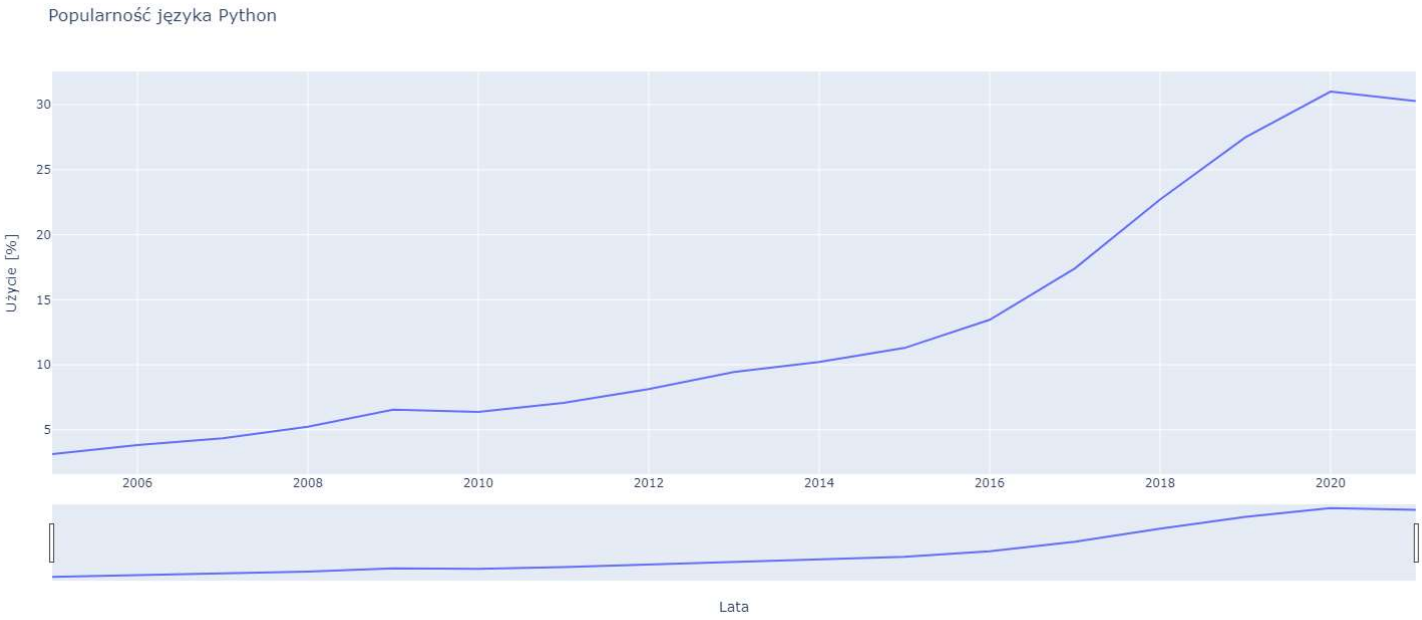


Ten wykres porównawczy ułatwia zrozumienie różnic w dystrybucji popularności języków Python i Java. Porównując oba histogramy, można szybko zobaczyć, który język ma większą lub mniejszą koncentrację w określonych zakresach procentowych. Dodatkowo, obecność krzywych KDE pozwala na ocenę kształtu rozkładu danych dla obu języków.

9) Wykres liniowy popularności języka Python – Plotly

Ten wykres liniowy przedstawia zmiany popularności języka Python w kolejnych latach. Linia ciągła reprezentuje procentowy udział użycia języka Python w projektach programistycznych na przestrzeni lat. Dodatkowo, użyto funkcjonalności interaktywnych elementów Plotly, takich jak suwak do zmiany zakresu osi x, co umożliwia użytkownikowi interaktywne przeglądanie danych.

```
# Wizualizacja danych za pomocą Plotly
def plot_with_plotly(data):

    x = data['Date']
    y = data['Python']
    plot = px.Figure(data=[px.Scatter(x=x, y=y, mode='lines')])
    plot.update_layout(
        title="Popularność języka Python",
        xaxis_title=r'Lata',
        yaxis_title=r'Użycie [%]',
        xaxis=dict(rangeselector=dict(buttons=list([dict(count=1, step="year",
stepmode="backward",)])),
                   rangeslider=dict(visible=True)))
    plot.show()
```



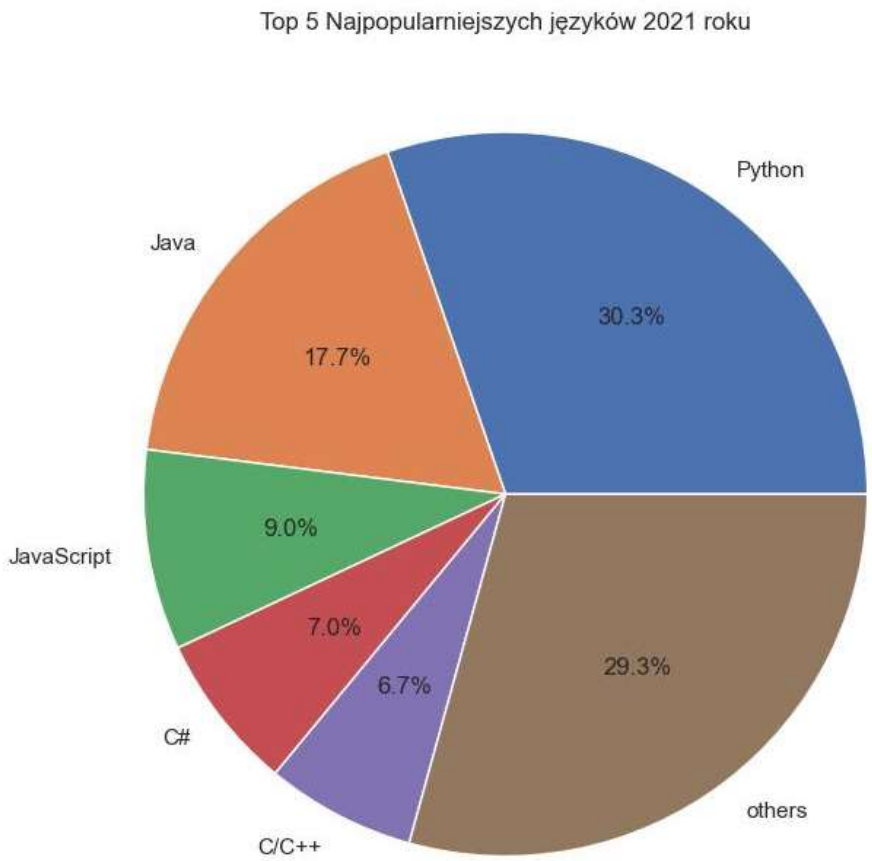
Ten interaktywny wykres pozwala na łatwe zrozumienie trendów popularności języka Python w różnych okresach czasu. Użytkownik może dostosować widok, wybierając określony zakres lat za pomocą suwaka, co pozwala na bardziej szczegółową analizę danych w wybranych okresach czasu.

10) Wykres kołowy top 5 najpopularniejszych języków w 2021 roku

Ten wykres kołowy prezentuje procentowy udział top 5 najpopularniejszych języków programowania w roku 2021. Dane zostały przetworzone, aby wybrać 5 najpopularniejszych języków oraz zsumować udziały pozostałych języków pod kategorią "others". Wykres kołowy jest czytelny i łatwy do zrozumienia, a wartości procentowe na kawałkach koła pozwalają szybko ocenić proporcje między poszczególnymi językami.

```
# Przetwarzanie danych dla roku 2021
data2 = data.set_index("Date")
data3 = data2.T
data4 = data3[2021]
data5 = data4.sort_values(ascending=False)

# Top 5 najpopularniejszych języków
data6 = data5[:5].copy()

# Tworzenie DataFrame z wszystkimi językami
data7 = pd.DataFrame({'language': data5.index, 'percentage': data5.values})

# Tworzenie DataFrame z top 5 językami
data6 = pd.DataFrame({'language': data6.index, 'percentage': data6.values})

# Dodanie kategorii 'others'
new_row = pd.DataFrame(data={
    'language': ['others'],
    'percentage': [data7['percentage'][5:].sum()]})
```

```
data8 = pd.concat([data6, new_row], ignore_index=True)

# Wyświetlenie końcowego DataFrame
display(data8)

# Wykres kołowy
data8.plot(kind='pie', y='percentage', labels=data8['language'],
autopct='%1.1f%%', figsize=(10, 10), legend=False,
           title="Top 5 Najpopularniejszych języków 2021 roku", ylabel=' ')
plt.show()
```



Ten wykres kołowy pozwala na szybkie zrozumienie, które języki programowania były najpopularniejsze w roku 2021. Dzięki wykorzystaniu kategorii "others", można również zobaczyć, jaki procent popularności zajmują inne języki programowania spoza top 5.

6. Podsumowanie

Podsumowując, wizualizacja danych w Pythonie jest niezwykle istotnym narzędziem w analizie danych i komunikacji wyników. Dzięki bogactwu bibliotek i narzędzi, jakie oferuje Python, można tworzyć wizualizacje na różnych poziomach skomplikowania, dopasowane do potrzeb konkretnego projektu. Jest to nie tylko sposób prezentacji wyników, ale także narzędzie do odkrywania wzorców, trendów i zależności w danych.

Jednakże warto pamiętać, że sama wizualizacja to tylko część procesu analizy danych. Ważne jest również odpowiednie zrozumienie danych, umiejętność interpretacji wykresów i wyciąganie wniosków na ich podstawie. Dlatego też wizualizacja danych powinna być traktowana jako środek do celu - ułatwienia zrozumienia danych i wspierania podejmowania decyzji opartych na faktach.