SPRAWOZDANIE

Zajęcia: Eksploracja i wizualizacja danych

Prowadzący: prof. dr hab. inż. Vasyl Martsenyuk

Laboratorum nr 1 Data: 26.09.2021 Temat: Ustalenia platformy Jupyter. Użycia biblioteki pandas w celu eksploracji i wizualizacji danych Wariant: 1	Piotr Rybka Informatyka II stopień, niestacjonarne, III semestr, gr. 1

Repozytorium z kodem programu:

`https://github.com/prybka82-student/eksploracja_i_wizualizacja_danych`

Polecenie

Zadanie dotyczy pobrania danych z pliku, tworzenia ramki danych, wykonania poszczególnych zadań na podstawie odpowiedniego zbioru danych:

`http://ghdx.healthdata.org/record/ihme-data/development-assistance-health-database-1990-2020`

Zadanie 1

Załadować bibliotekę `pandas`.

```
import pandas as pd
```

Zadanie 2

Utworzyć ramkę danych ze słownika.

```
dane1 = {
    "liczby": [532, 2532, 523, 2543, 235, 231, 34, 23552, 5324],
    "kategorie": ["a", "b", "c", "d", "e", "f", "g", "h", "i"]
}

df1 = pd.DataFrame(dane1)
df1
```

Zadanie 3

Zapisać ramkę danych w pliku `csv`.

```
sciezka = r"C:\Users\piotr\Downloads\dane_ze_slownika.csv"

df1.to_csv(sciezka, encoding="utf-8")
```

Zadanie 4

Utworzyć ramkę danych z listy list.

```
dane2 = [
    [532, 2532, 523, 2543, 235, 231, 34, 23552, 5324],
    ["a", "b", "c", "d", "e", "f", "g", "h", "i"]
]

df2 = pd.DataFrame(dane2)
```

Zadanie 5

Wczytać dane z pliku `csv`.

```
sciezka = r"C:\Users\piotr\Downloads\IHME_DAH_DATABASE_1990_2020_CSV_1\
    IHME_DAH_DATABASE_1990_2020_Y2021M09D22.CSV"

df = pd.read_csv(sciezka, low_memory=False, )
```

Zadanie 6

Wyświetlić pierwsze 10 wierszy ramki danych.

```
df.head(10)
```

Zadanie 7

Wyświetlić ostatnie 10 wierszy ramki danych.

```
df.tail(10)
```

Zadanie 8

Wyświetlić informacje o ramce danych.

```
df.info()
```

Zadanie 9

Przetransponować dane w ramce danych.

```
df_transposed = df.T

df_transposed.head()
```

Zadanie 10

Wyświetlić liczbę wierszy i kolumn zawartych w ramce danych.

```
rows, cols = df.shape
print(f"Kolumn: {cols}")
print(f"Wierszy: {rows}")
```

Zadanie 11

Wyodrębnić wiersze i kolumny przy użyciu nazw i indeksów.

```
df["year"]

df.year

df[["year", "source", "recipient_country"]]

df.loc[:, "year":"channel"] # wszystkie wiersze, kolumny od "year" do "
    channel" wlacznie

df.iloc[0:10, 2:4] # wiersze od pierwszego do dziesiatego, kolumny od
    trzeciej do czwartej
```

Zadanie 12

Wyświetlić podstawowe informacje statystyczne o kolumnach liczbowych (liczba wartości niepowtarzalnych, średnia, odchylenie standardowe, minimum, maksimum, wartości kwartyli).

```
liczbowe = df.select_dtypes(include='number')
liczbowe.describe()
```

Zadanie 13

Wyświetlić podstawowe informacje statystyczne o kolumnach kategoryzowanych (liczba wartości niepowtarzalnych, wartość najczęstsza, liczba wystąpień wartości najczęstszej).

```
kategoryzowane = df.select_dtypes(exclude='number')
kategoryzowane.describe()
```

Zadanie 14

Usunięcie brakujących wartości z ramki danych.

```
df.dropna(inplace=True) # inplace=True - zmodyfikowane zostana oryginalne
    dane
```

Zadanie 15

Wybrać te wiersze ramki danych, które spełniają podany warunek dotyczący wybranej kolumny.

```
df[df["recipient_country"] == "Poland"]
```

Zadanie 16

Wybrać wiersze ramki danych, które spełniają kilka warunków jednocześnie.

```
df[(df["recipient_isocode"] == "POL") & (df["year"] >= 2005)]
```

Zadanie 17

Wybrać wiersze, które zawierają w kolumnie kategoryzowanej określone słowo.

```
df[df["source"]=="Japan"]
```

Zadanie 18

Wybrać wiersze, które nie zawierają w kolumnie kategoryzowanej podanego wyrazu.

```
selection = df[df["gbd_region"]!="Asia, East"]
```

Zadanie 19

Utworzyć kolumnę na podstawie istniejącej kolumny w ramce danych.

```
recipient_countries = df.recipient_country
```

Zadanie 20

Usunąć kolumnę z ramki danych.

```
df_copy = df
df_copy.drop(["source", "channel"], axis=1, inplace=True) # usuwa kolumny "
    source" i "channel"
```

Zadanie 21

Zmienić nazwę kolumny w ramce danych.

```
df_copy.rename(columns={"year": "rok", "recipient_isocode": "
    kod_iso_odbiorcy", "recipient_country": "kraj_odbiorcy"}, inplace=True)
```

Zadanie 22

Zapisać ramkę danych jako plik w formacie `csv`.

```
sciezka = r"C:\Users\piotr\Downloads\df_copy.csv"
df.to_csv(sciezka, encoding="utf-8")
```

Zadanie 23

Wyświetlić średnią, maksymalną i minimalną wartość danej kolumny.

```
col = df["elim_ch"]
mean = col.mean()
max_ = col.max()
min_ = col.min()

print(f"srednia: {mean}\nmaksimum: {max_}\nminimum: {min_}")
```

Zadanie 24

Podać liczbę wierszy wybranej kolumny.

```
df.rok.count()
```

Zadanie 25

Wyodrębnić wartości unikatowe w kolumnie ramki danych.

```
df_copy["kraj_odbiorcy"].unique()
```

Zadanie 26

Wyświetlić liczbę rekordów spełniajacych podany warunek.

```
df_copy[df_copy["kraj_odbiorcy"]=="Turkey"].rok.count()
```

Zadanie 27

Posortować wiersze ramki danych według wartości w wybranej kolumnie (malejąco i rosnąco).

```
df_copy.sort_values(['kraj_odbiorcy'], ascending=True).head()

df_copy.sort_values(['kraj_odbiorcy'], ascending=False).head()
```

Zadanie 28

Wyświetlić wiersze zawierające 10 największych i najmniejszych wartości z wybranej kolumny.

```
df.nlargest(10, 'elim_ch')[['rok', 'elim_ch']]

df.nsmallest(10, 'elim_ch')[['rok', 'elim_ch']]
```

Zadanie 29

Wyświetlić wiersze zawierające 10 największych wartości wybranej kolumny pod warunkiem, że wartości w innej kolumnie mają określoną wartość.

```
df[(df['kraj_odbiorcy'].isin(['Poland', 'Germany'])) & (df['rok'] == 1990)
    ].nlargest(10, 'elim_ch')
```

Zadanie 30

Pogrupować wiersze według wartości kolumny kategoryzowanej, a następnie uśrednić wartości wszystkich kolumn liczbowych.

```
df.groupby('kraj_odbiorcy').agg('mean')
```

Zadanie 31

Pogrupować wiersze wg wartości kolumny kategoryzowanej, a następnie uśrednić wartości dla wybranych kolumny, a dla innych obliczyć częstość wystąpień i medianę.

```
kraje = df.groupby('kraj_odbiorcy').agg({'gbd_location_id': ['mean'], '
    elim_ch': ['mean'], 'prelim_est': ['mean'], 'rok': ['count', 'median'],
    'wb_location_id': ['count', 'median'], 'gbd_superregion_id': ['count', '
    median']})
```

Zadanie 32

Pobrać nazwy kolumn indeksu złożonego.

```
kraje.index
```

Zadanie 33

Posortować kolumnę indeksu złożonego.

```
kraje['elim_ch']['mean'].sort_values(ascending=False)
```

Zadanie 34

Stworzyć tabelę przestawną (ang. *pivot table*) na podstawie ramki danych.

```
pivot = df.pivot_table(values='elim_ch', index='kraj_odbiorcy', columns='
    rok', aggfunc='count', margins=False, dropna=True, fill_value=None)
```

Zadanie 35

Wyświetlić indeksy i kolumny tabeli przestawnej.

```
pivot.index
```

Zadanie 36

Utworzyć indeks złożony tabeli przestawnej i wyświetlić jego zawartość.

```
pivot2 = df.pivot_table(values='elim_ch', index=['gbd_region', '
    kraj_odbiorcy'], columns='rok', aggfunc='count', margins=False, dropna=
    True, fill_value=None)
```

Zadanie 37

Zaimportować moduł `pyplot` z biblioteki `matplotlib` oraz wskazać, że wykresy należy rysować bezpośrednio w zeszycie, a nie w osobnej zakładce.

```
import matplotlib.pyplot as plt

%matplotlib inline
```

Zadanie 38

Wyświetlić wykres na podstawie tabeli przestawnej.

```
pivot3 = df.pivot_table(values='elim_ch', index='rok', columns='
    gbd_superregion', aggfunc='mean', margins=False, dropna=True, fill_value
    =None)

fig = pivot3.plot(kind='line')
plt.ylabel('mean value of elim_ch')
plt.title('Mean values of elim_ch in superregions per year')
plt.rcParams["figure.figsize"] = (20,10)
```

Zadanie 39

Narysować histogram na podstawie wartości w wybranej kolumnie.

```
df_bar = df.pivot_table(values='elim_ch', index='rok', columns='
    gbd_superregion', aggfunc='sum', margins=False, fill_value=None, dropna=
    True)
df_bar.plot(kind='bar')
plt.ylabel('elim_ch')
plt.title('elim_ch by year and superregion')
```

Zadanie 40

Przedstawić sposoby łączenia ramek danych za pomocą metod `merge` i `concat`.

```
part1 = df[['rok', 'kraj_odbiorcy', 'elim_ch', 'gbd_region']]
part2 = df[['rok', 'kraj_odbiorcy', 'prelim_est', 'gbd_superregion']]

pd.merge(part1, part2, on = ['rok', 'kraj_odbiorcy'], how='inner').head()

pd.merge(part1, part2, on = [part1.index, part2.index], how='inner').head()

pd.concat([part1, part2], axis=0)

pd.concat([part1, part2], axis=0).shape

pd.concat([part1, part2], axis=1)

pd.concat([part1, part2], axis=1).shape
```

Zadanie 41

Pokazać dodawanie nowych kolumn za pomocą operacji matematycznych.

```
df['sum'] = df['elim_ch'] + df['prelim_est']
```

Zadanie 42

Przedstawić przykładowe dodawanie nowych kolumn przy użyciu wyrażenia lambda.

```
df['years_ago'] = df['rok'].apply(lambda y: 2021 - int(y))
```

Zadanie 43

Ukazać możliwości pracy z dużymi plikami przy użyciu argumentu `chunksize`.

```
sciezka = r"C:\Users\piotr\Downloads\IHME_DAH_DATABASE_1990_2020_CSV_1\
    IHME_DAH_DATABASE_1990_2020_Y2021M09D22.CSV"

chunks = pd.read_csv(sciezka, low_memory=False, chunksize=100_000)

for i, chunk in enumerate(chunks):
    print(f"\n\nChunk number {i+1}:")
    print(chunk.iloc[0:3,0:4])
```

Wnioski

- Załadowanie biblioteki `pandas`: `import pandas as pd`.
- Tworzenie ramki danych ze słownika: `pd.DataFrame("kol1": [], "kol2": [])`.
- Tworzenie ramki danych z listy list: `pd.DataFrame([[], [], []])`.
- Wczytanie danych do ramki danych z pliku `csv`: `pd.read_csv(sciezka, low_memory=False)`.
- Zapis danych z ramki danych do pliku `csv`: `pd.to_csv(sciezka, encoding="utf-8")`.
- Wybranie n pierwszych lub ostatnich wierszy ramki danych:
  - `df.head(n)`;
  - `df.tail(n)`;
- Transponowanie danych (zamiana miejscami kolumn i wierszy): `df.T`.
- Zestawienie informacji o ramce danych:
  - `df.info()` – spis kolumn wraz z podaniem typu i liczbą wartości niepustych (*non-null*);
  - `rows, cols = df.shape` – liczba, odpowiednio, wierszy i kolumn;
  - `df.describe()` – miary statystyczne opisujące dane;
- Sposoby wybrania kolumny danych:
  - `df['nazwa_kolumny']`;
  - `df.nazwa_kolumny`;
  - `df[["kolumna1", "kolumna2", "kolumna3"]]`;
  - `df.loc[:, "kolumna1":"kolumnaX"]` – wszystkie wiersze kolumn od 1. do x;
  - `df.iloc[:, 2:4]` – wszystkie wiersze kolumn od 3. do 4.

- Odfiltrowanie kolumn o wartościach liczbowych: `df.select_dtypes(include='number')` i kategoryzowanych: `df.select_dtypes(exclude='number')`.
- Odfiltrowanie wierszy spełniających podane warunki:
  - `df[df["nazwa kolumny"] == "wartosc"]`;
  - `df[(df["nazwa_kolumny"] == "wartosc") & (df["kolumnaX"] isin(["wart1", "wart2"])) & (df["nazwa_kolumny"] >= 100)]`;
- Usunięcie brakujących wartości z ramki danych: `df.dropna(inplace=True)`.
- Usunięcie kolumny o wskazanych nazwach: `df.drop(["kolumna1", "kolumna2"], axis=1, inplace=True)`.
- Podstawowe miary opisujące dane:
  - liczba wartości: `df.kolumna.count()`;
  - średnia arytmetyczna: `df.kolumna.mean()`;
  - maksimum: `df.kolumna.max()`;
  - minimum: `df.kolumna.min()`.
- Odfiltrowanie wartości niepowtarzalnych z kolumny: `df.kolumna.unique()`.
- Sortowanie danych wg danych we wskazanej kolumnie: `df.sort_values(["kolumna"], ascending=True)`.
- Wyświetlenie n najmniejszych i największych wartości we wskazanych kolumnach:
  - `df.nlargest(n, "kolumna")`;
  - `df.nsmallest(n, "kolumna")`.
- Grupowanie i agregowanie danych: `df.groupby("kolumna").agg('mean')`.
- Indeks pogruowanych wartości: `df.index`.
- Tabela przestawna z indeksem prostym: `pivot1 = df.pivot_table(values="wartosci", index=["kol1"], columns=żok", aggfunc=ćount", margins=False, dropna=True, fill_value=None)`
- Załadowanie modułu `pyplot`: `import maptplotlib.pyplot as plt`. Opcja rysowania wykresów bezpośrednio w zeszycie: `%matplotlib inline`.
- Wykres na podstawie tabeli przestawnej: `df.pivot_table(values="kol1", index="kol2", aggfunc="mean", margins=False, dropna=True, fill_value="none")`

- Histogram na podstawie tabeli przestawnej: `df.pivot_table(values="kol1", index="kol2", columns_gbd)`.
- Łączenie danych:
  - merge: `pd.merge(frame1, frame2), on=[part1.index, part2.person], how='inner'`
  - concat: `pd.concat(frame1, frame2), axis=0`
- Dodawanie kolumn: `df["nowaKolumna"] = df["kol1"] + df["kol2"]`.
- Podział ładowanych danych na porcje o wielkości x bajtów: `pd.read_csv(sciezka, low_memory=False, chunksize=100_000)`).