Analiza rozmieszczenia sklepów sieci Biedronka na terenie Polski

Miron Czech

O projekcie **2**

Gromadzenie Danych **2**

Analiza eksploracyjna danych **2**
- Czyszczenie danych 2
- Zawartość bazy danych 3
  - Gminy 3
  - Sklepy 4
  - Godziny_otwarcia 4
  - Liczba_sklepow 5
- Przykładowe zależności 5
  - Relacja dla każdej gminy z osobna 6
  - Podział ze względu na typ gminy 9
  - Podział na regularne przedziały 10
  - Autorski podział na przedziały 11

Badanie korelacji i zależności danych **13**
- Badanie histogramów 13
- Wzrost liczby sklepów wraz ze wzrostem liczby mieszkańców 17
- Wzrost stosunku liczby sklepów do liczby mieszkańców wraz ze wzrostem liczby mieszkańców 17

Wnioski i podsumowanie **18**

Dokumentacja **19**
- Communities.py 19
- Data_downloader.py 21
- Database_operations.py 25

O projekcie

Projekt przedstawia analizę zależności między ilością sklepów sieci Biedronka, a miejscem ich występowania. Do jego realizacji użyto języków python oraz R, a także SQLite do stworzenia bazy danych. Inne użyte narzędzia to PyCharm, RStudio, DB Browser for SQLite, Google Docs oraz Excel.

Pomysł na projekt zrodził się podczas jednej ze studenckich wycieczek do biedronki. W głowie autora pojawiła się wtedy pewna myśl: “Chodzę do Biedronki od lat, niezależnie od tego gdzie mieszkam. Ten sklep jest chyba wszędzie! Właśnie, czy wszędzie? Może Biedronek w miastach jest stosunkowo mało? A może to w moim rodzinnym miasteczku jest ich wyjątkowo dużo? Czy można Biedronkę określić mianem sklepu dla prowincji? A może wręcz przeciwnie, jest to sklep typowo miejski?” Odpowiedź na te pytania znajduje się w poniższym skrypcie :)

Gromadzenie Danych

Dane, na których oparty jest projekt, zostały pobrane ze strony GUS oraz strony sieci Biedronka. Ze strony GUS wykorzystano tabelę 11 z dostępnego na stronie skompresowanego folderu. Dane poddano krótkiej, wstępnej obróbce ręcznej w programie Excel, a następnie za pomocą skryptu 1 przekonwertowano je do pliku .csv. Niestety, nie udało uzyskać się danych w jednym pliku na temat wszystkich sklepów sieci Biedronka, support odsyła zainteresowanych do strony internetowej sieci. Stąd też zostały pobrane dane do projektu za pomocą skryptu 2 zapisującego je do pliku .csv. Z powodu braku odpowiednich materiałów zostało pominiętych ok. 150 sklepów znajdujących się w miejscowościach nie będących siedzibami gmin. Jest to akceptowalny brak w skali ok. 3300 sklepów w kraju.

Analiza eksploracyjna danych

Czyszczenie danych

Za całość czyszczenia danych odpowiadają skrypty napisane w pythonie oraz ręczne poprawki. Warto wymienić kolejne kroki wykonane w celu doprowadzenia danych do postaci wyjściowej:

1. Przekopiowanie arkuszy dla każdego z województw do nowych plików tak, by dało się je zapisać bez problemu jako .csv w tym:
    a. pozbycie się nagłówków i zbędnych kolumn
    b. sprowadzenie nazw województw do pojedynczych komórek (np WOJ. KUJAWSKO - POMORSKIE zostało umieszczone w 2 komórkach)
    c. usunięcie wierszy z dzielnicami miast
    d. usunięcie dopisku “M. St.“ przed “Warszawa”
2. Zapisanie plików jako .csv i uruchomienie skryptu konwertującego wszystkie pliki na jeden, wprowadzany później do bazy danych. Skrypt działa następująco:
    a. Zapisanie dla każdego wiersza z gminą niezbędnych informacji do tablicy
    b. Zapisanie do pliku .csv informacji z tablicy
3. Dodanie powstałego pliku do bazy danych za pomocą prostej metody z biblioteki pandas.

4. Pobranie danych o sklepach i zapisanie ich do pliku .csv za pomocą skryptu działającego następująco:
    a. Pobranie informacji o gminach z bazy danych
    b. Dla kolejnych gmin wyszukanie strony, pobranie pliku HTML
    c. obróbka w celu uzyskania danych
    d. zapis do plików .csv
5. Dodanie powstałych plików do bazy danych za pomocą prostej metody z biblioteki pandas.

Zawartość bazy danych

Baza danych została stworzona w SQLite z pomocą pythona. Funkcje tworzące tabele i widoki oraz dodające dane są w skrypcie 3.

Zebrano informacje o 2477 gminach. Dane przedstawiają stan z dnia 30.06.2022r. Każda z gmin jest określona następującymi parametrami: identyfikator, nazwa, typ, liczba mieszkańców, województwo. Przykładowy fragment tabeli gminy z bazy danych:

Gminy

id	nazwa	typ	liczba_ mieszkancow	wojewodztwo	powiat
201011	Bolesławiec	miasto	37355	Dolnośląskie	bolesławiecki
201022	Bolesławiec	wies	15181	Dolnośląskie	bolesławiecki
201032	Gromadka	wies	5052	Dolnośląskie	bolesławiecki
201043	Nowogrodziec	gmina miejsko-wiejska	14730	Dolnośląskie	bolesławiecki
201052	Osiecznica	wies	7297	Dolnośląskie	bolesławiecki
201062	Warta Bolesławiecka	wies	8407	Dolnośląskie	bolesławiecki
202011	Bielawa	miasto	28475	Dolnośląskie	dzierżoniowski
202021	Dzierżoniów	miasto	31256	Dolnośląskie	dzierżoniowski
202033	Pieszyce	gmina miejsko-wiejska	9026	Dolnośląskie	dzierżoniowski
202041	Piława Górna	miasto	6016	Dolnośląskie	dzierżoniowski

Zebrano również dane na temat sklepów sieci Biedronka. Dane zostały pobrane ze strony Biedronki 22.01.2023r. Informacje na temat sklepów umieszczono w 2 tabelach w bazie danych: sklepy oraz godziny_otwarcia. Poniżej umieszczono przykładowe fragmenty tych tabel:

Sklepy

id_gminy	id_sklepu	kod_pocztowy	ulica
611032	1	21-412	Zagłoby 3
611032	2	06-320	Antoniego Madalińskiego 48
401042	3	87-700	J. Słowackiego 28c
401042	4	87-700	G. Narutowicza 8A
401042	5	62-540	aleja 600-lecia 23
401042	6	66-540	Kościuszki 46
401042	7	73-108	Szczecińska 1
1020043	8	95-069	Konstantynowska 5/7
1203013	9	32-566	A. Mickiewicza 15a
1218013	10	34-120	Krakowska 83

Godziny_otwarcia

id_ sklepu	poniedziałek	wtorek	środa	czwartek	piątek	sobota	niedziela
1	06:00-22:00	06:00-22:00	06:00-22:00	06:00-22:00	06:00-22:00	06:00-22:00	09:00-20:00
2	06:00-22:00	06:00-23:30	06:00-23:30	06:00-23:30	06:00-23:30	06:00-23:30	08:00-20:00
3	06:00-23:00	06:00-23:00	06:00-23:00	06:00-23:00	06:00-23:00	06:00-23:00	07:00-21:00
4	06:00-23:00	06:00-23:00	06:00-23:00	06:00-23:00	06:00-23:00	06:00-23:00	07:00-21:00
5	06:00-23:00	06:00-23:00	06:00-23:00	06:00-23:00	06:00-23:00	06:00-23:00	08:00-21:00
6	06:00-22:00	06:00-22:00	06:00-22:00	06:00-22:00	06:00-22:00	06:00-22:00	09:00-20:00
7	06:00-22:00	06:00-22:00	06:00-22:00	06:00-22:00	06:00-22:00	06:00-22:00	08:00-21:00
8	06:00-23:00	06:00-23:00	06:00-23:00	06:00-23:00	06:00-23:00	06:00-23:00	07:00-21:00
9	05:00-23:30	05:00-23:30	05:00-23:30	05:00-23:30	05:00-23:30	05:00-23:30	06:00-21:00
10	05:00-23:30	05:00-23:30	05:00-23:30	05:00-23:30	05:00-23:30	05:00-23:30	09:00-20:00

Część danych w bazie nie została wykorzystana w projekcie (np tabela godziny_otwarcia), jednak dane te zostały zachowane w celu zwiększenia możliwości analizy.

Poza tabelami stworzono dwa widoki w celu łatwiejszego korzystania z danych na późniejszym etapie projektu. Są to liczba_sklepow i sklepy_detale, przykładowy fragment poniżej:

Liczba_sklepow

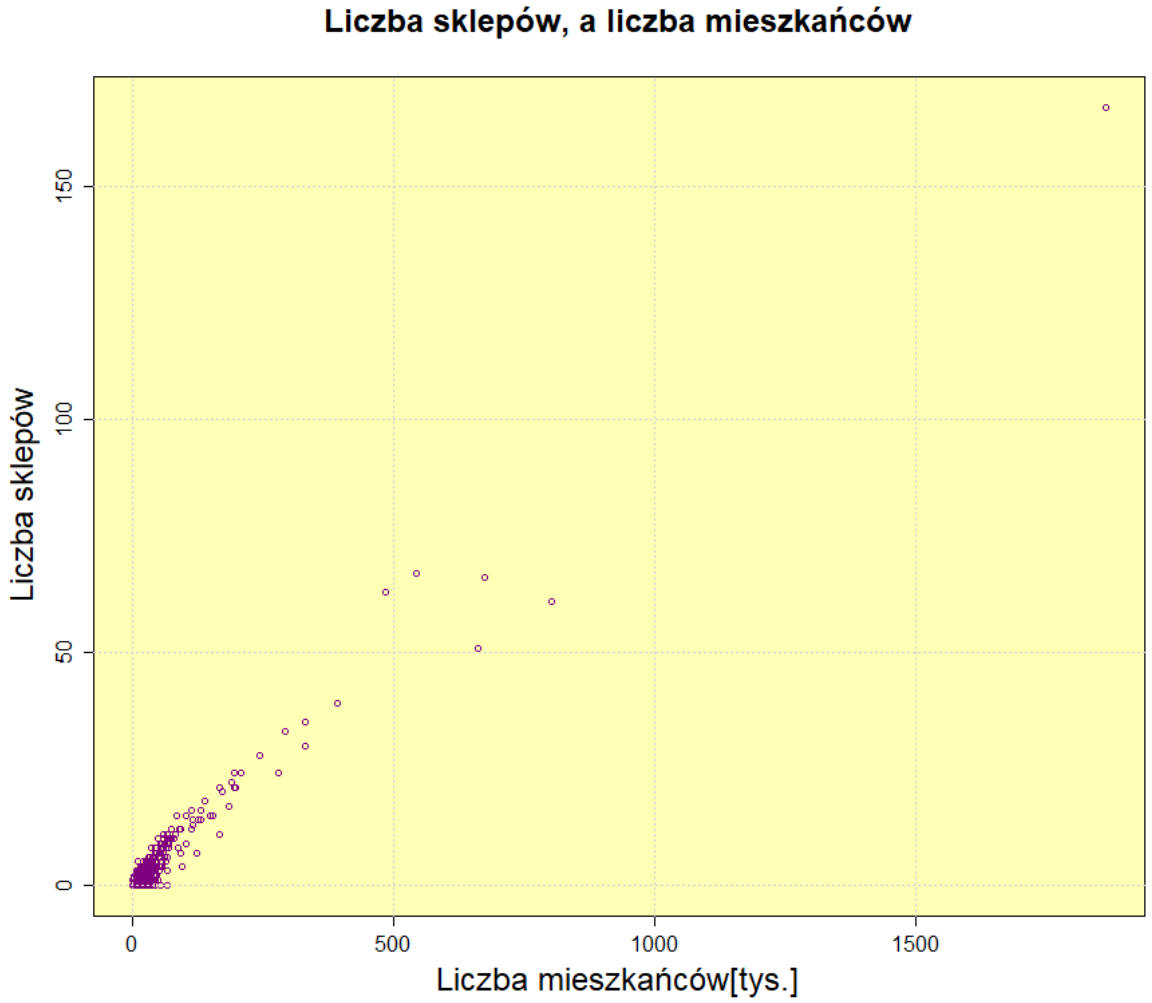
nazwa	id	wojewodztwo	powiat	typ	liczba_ mieszkancow	liczba_ sklepow
Abramów	608022	Lubelskie	lubartowski	wies	3873	0
Adamów	611032	Lubelskie	łukowski	wies	5207	2
Adamów	620012	Lubelskie	zamojski	wies	4462	0
Adamówka	1814022	Podkarpackie	przeworski	wies	4035	0
Aleksandrów	602022	Lubelskie	biłgorajski	wies	3201	0
Aleksandrów	1010012	Łódzkie	piotrkowski	wies	4280	0
Aleksandrów Kujawski	401011	Kujawsko-pomorskie	aleksandrowski	miasto	11586	0
Aleksandrów Kujawski	401042	Kujawsko-pomorskie	aleksandrowski	wies	11970	5
Aleksandrów Łódzki	1020043	Łódzkie	zgierski	gmina miejsko-wiejska	34753	1
Alwernia	1203013	Małopolskie	chrzanowski	gmina miejsko-wiejska	12397	1

Widok sklepy_detale jest zbyt szeroki, by umieszczenie go tu poprawiło czytelność bazy. Zawiera on następujące kolumny: id, nazwa, typ, liczba_mieszkancow, wojewodztwo, powiat, id_gminy, id_sklepu, kod_pocztowy, ulica, id_sklepu, poniedzialek, wtorek, sroda, czwartek, piatek, sobota, niedziela. Są to niemalże wszystkie zebrane dane na temat sklepów.

Przykładowe zależności

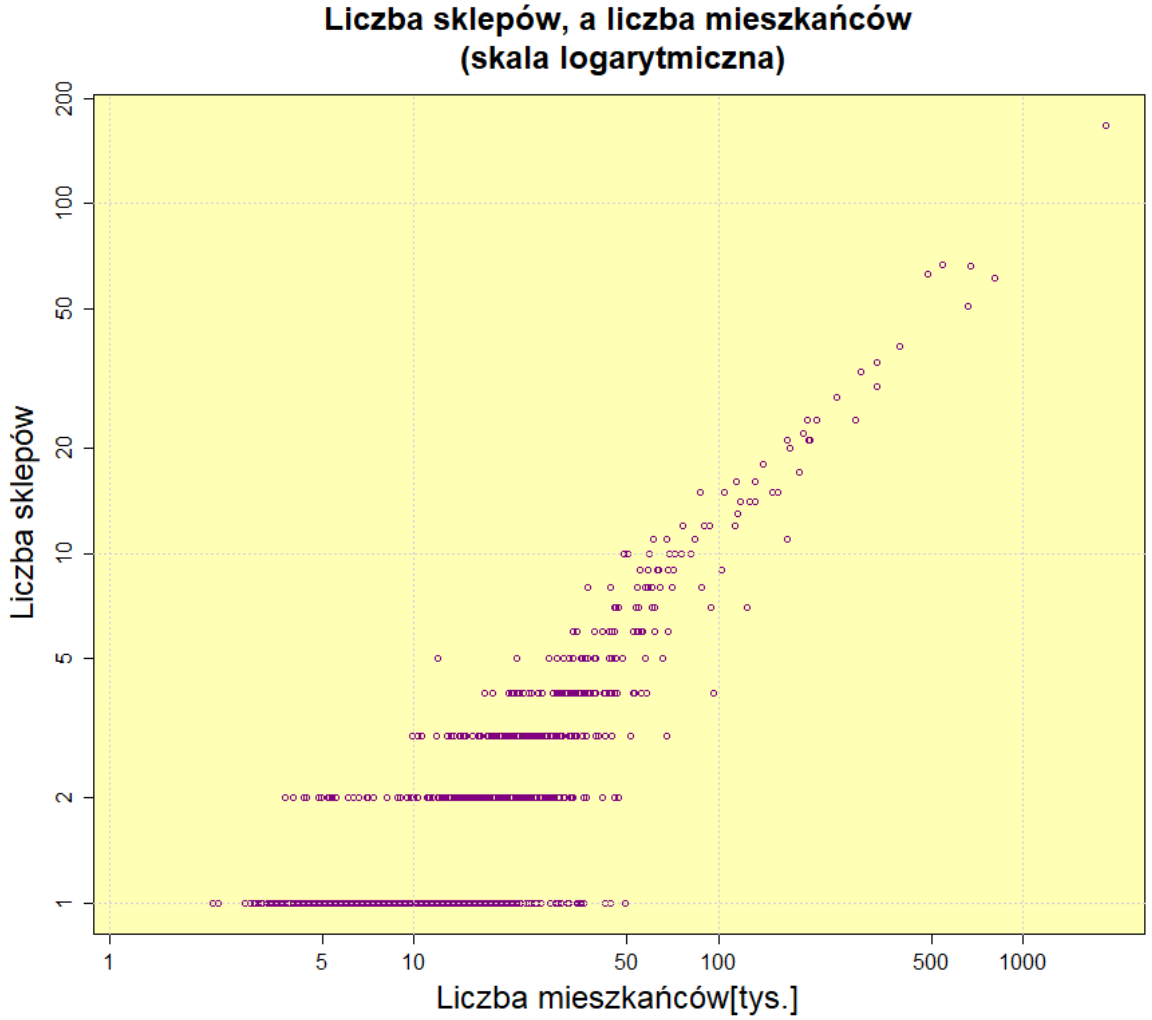
Poniżej zaprezentowano przykładowe zależności pomiędzy liczbą sklepów, a ich rozmieszczeniem. W analizie skupiono się na relacji liczby sklepów w danej gminie i jej liczby mieszkańców oraz na liczbie sklepów na mieszkańca w różnych przypadkach.

Relacja dla każdej gminy z osobna



Powyższy wykres przedstawia zależność liczby sklepów od liczby mieszkańców w każdej z gmin. Można zauważyć, że jedna z gmin zdecydowanie się wyróżnia na tle pozostałych - ma wyraźnie więcej sklepów i mieszkańców. Jak łatwo zgadnąć jest to Miasto Stołeczne Warszawa. Poza tym tylko 6 gmin (0,24% wszystkich gmin) ma na swoim terenie powyżej 50 sklepów, a tylko 5 gmin (0,20% wszystkich gmin) w kraju liczy powyżej 500 tys. mieszkańców. Rekordy te jednak nie są pomijalne przez bardzo dużą liczbę sklepów znajdujących się w takich gminach.

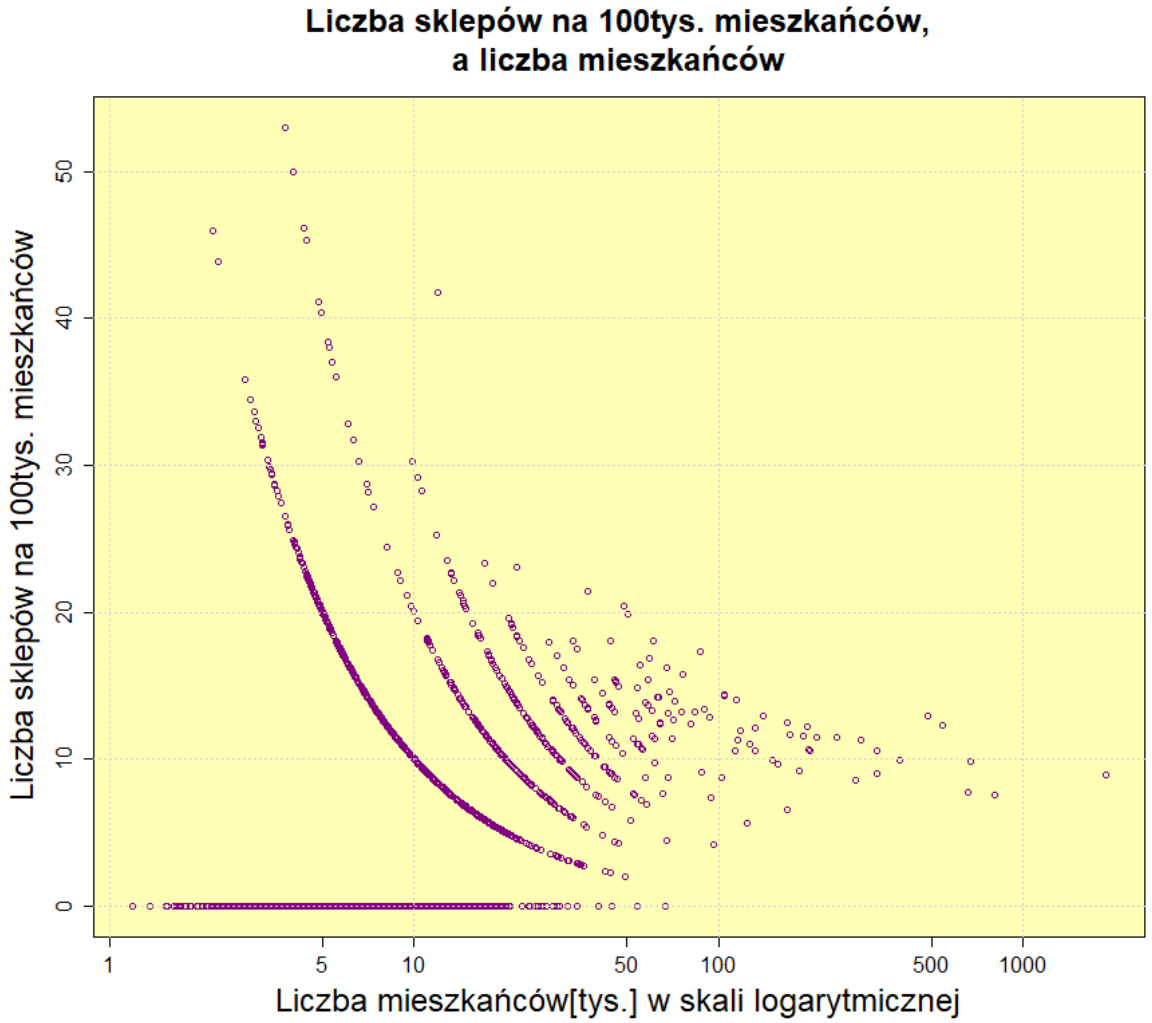
Poniżej znajduje się wykres tej samej zależności w skali logarytmicznej, w celu zwiększenia czytelności:



Wykres ten jest zdecydowanie czytelniejszy. Wciąż można zauważyć, że w Warszawie sklepów i mieszkańców jest najwięcej. Widać także, że liczba mieszkańców większości gmin nie przekracza 50 tys., a w większości z nich jest do pięciu sklepów. Na wykresie nie zaprezentowano gmin nie posiadających sklepu z powodu braku możliwości umieszczenia wartości zerowej w skali logarytmicznej.

Poniżej zaprezentowano wskaźniki położenia i rozproszenia dla liczby sklepów:
Średnia: 1.272507
Mediana: 0
Wariancja: 25.59736
Odchylenie standardowe: 5.059383
Minimum: 0
Kwartyl dolny: 0
Kwartyl górny: 1
Maksimum: 167
Rozstęp międzykwartylowy: 1
Z zerowej mediany i niskiego kwartylu górnego można wywnioskować, że w większości gmin w kraju najpewniej w ogóle nie ma sklepów sieci Biedronka.

Ostatni wykres dotyczy gęstości występowania sklepów:

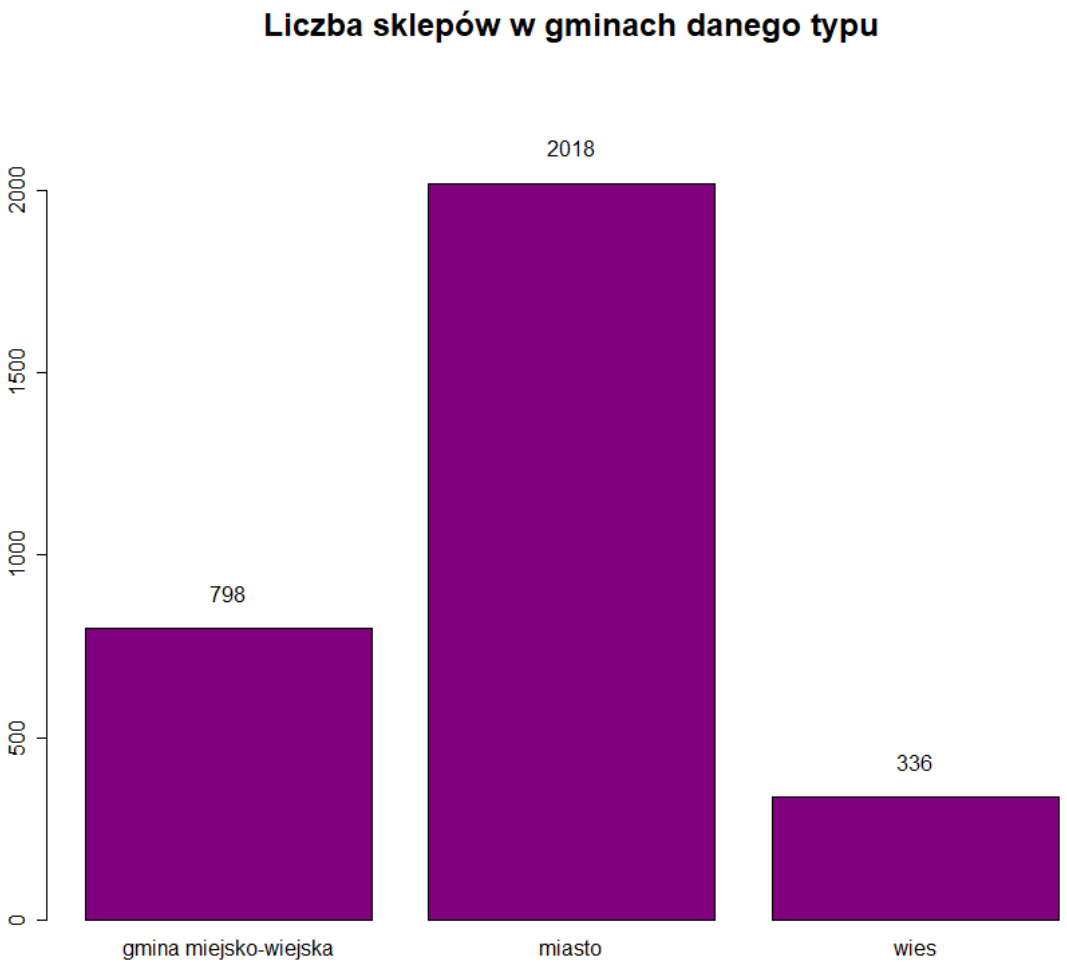


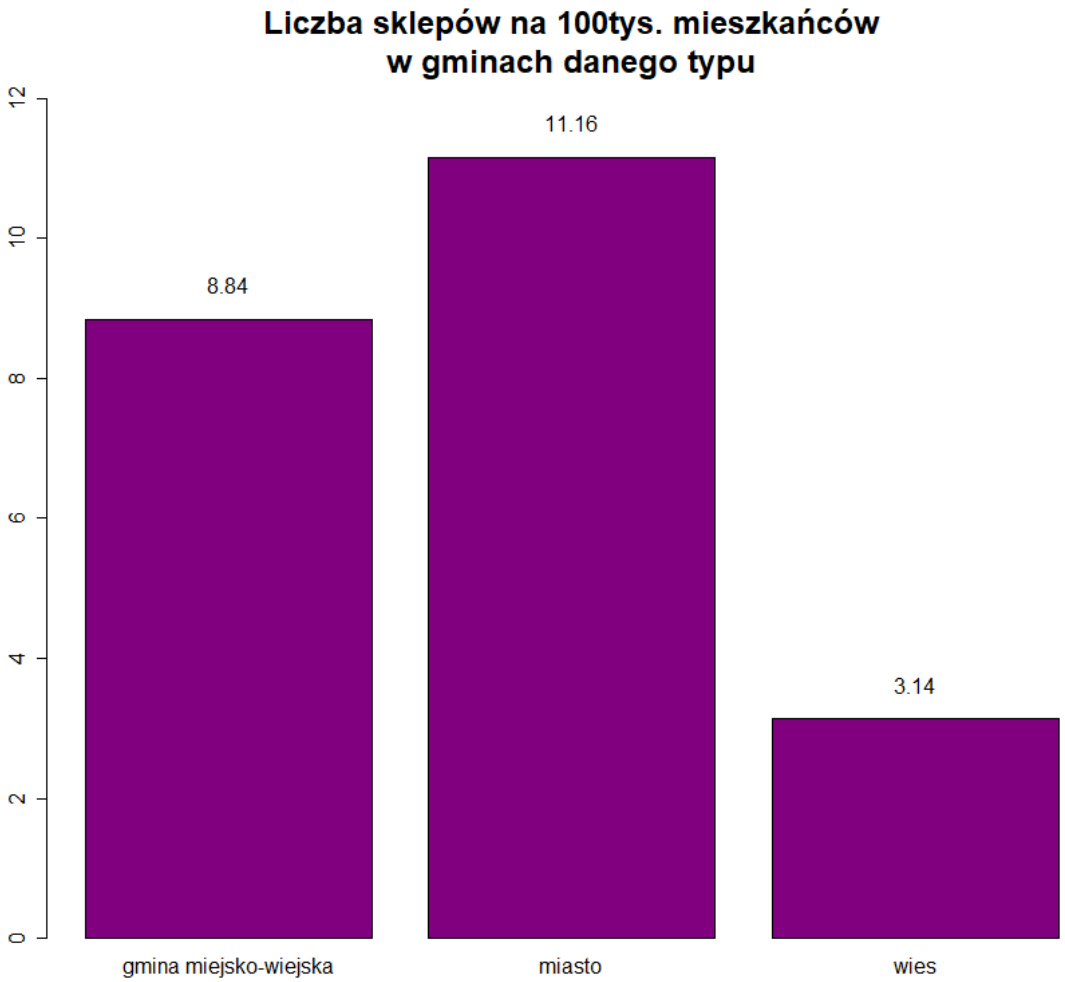
Oś x wykresu została przedstawiona w skali logarytmicznej w celu “przysunięcia” Warszawy do reszty gmin, a oś Y została przedstawiona w skali liniowej w celu stworzenia lepszych możliwości odczytu wartości. Można zauważyć, że ilość sklepów na mieszkańca potrafi się znacznie różnić w zależności od gminy. W większych miastach jest ona podobna i wynosi ok 10 sklepów na 100 tys. mieszkańców. Najwięcej sklepów na 100 tys. mieszkańców ma gmina Ustronie Morskie i liczba ta to 52,97 przy dwóch posiadanych Biedronkach.

Wskaźniki położenia i rozproszenia dla liczby sklepów na 100 tys. mieszkańców prezentują się następująco:
Średnia: 6.025687
Mediana: 0
Wariancja: 62.64838
Odchylenie standardowe: 7.915073
Minimum: 0
Kwartyl dolny: 0
Kwartyl górny: 11.44217
Maksimum: 52.96610
Rozstęp międzykwartylowy: 11.44217
Wysoka wartość wariancji potwierdza rozproszenie danych widoczne na wykresie.

Podział ze względu na typ gminy

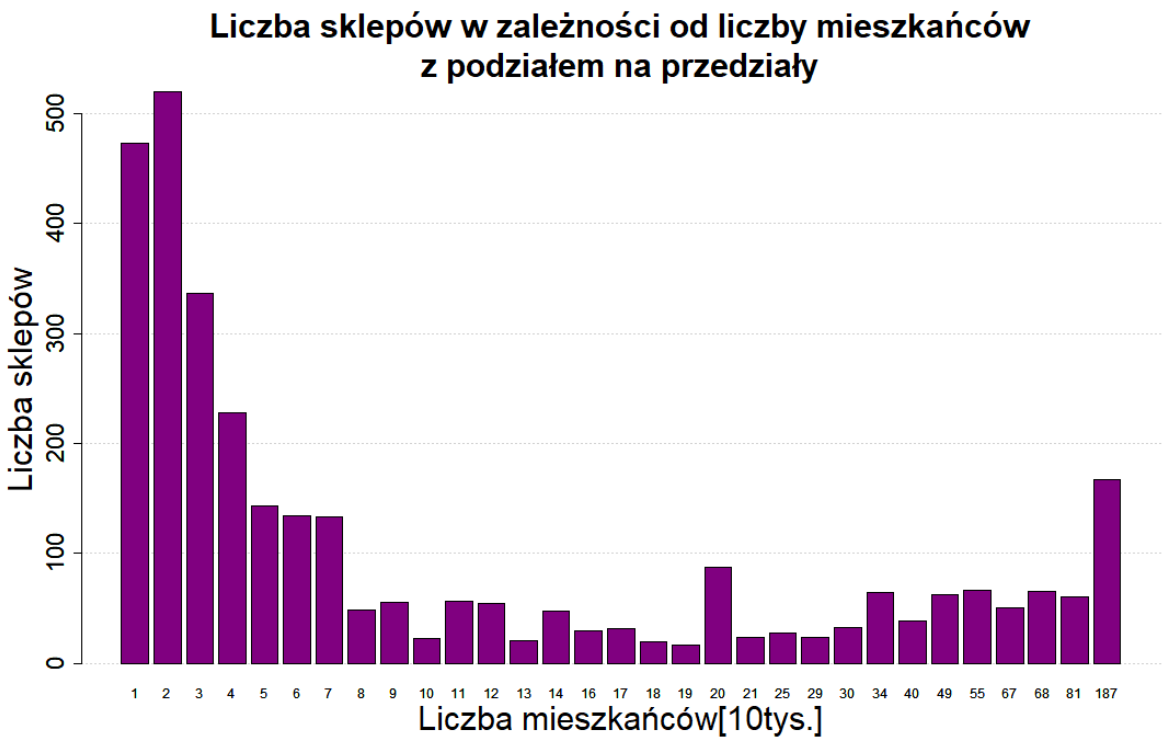
Gminy w Polsce dzielą się na gminy miejskie, miasta na prawach powiatu, gminy wiejskie oraz gminy miejsko-wiejskie. W bazie danych, dla uproszczenia, dwie pierwsze kategorie połączono w jedną. Z poniższych wykresów można wywnioskować, że najwięcej Biedronek jest w miastach oraz przypada tam ich najwięcej na jednego mieszkańca. Najmniej Biedronek jest na wsiach i tam też przypada najmniej biedronek na jednego mieszkańca. Warto dodać, że niedoszacowanie wynikające z braku danych dotknęło najbardziej gminy wiejskie i miejsko-wiejskie, więc różnice ukazane na diagramie w rzeczywistości są nieznacznie mniejsze.

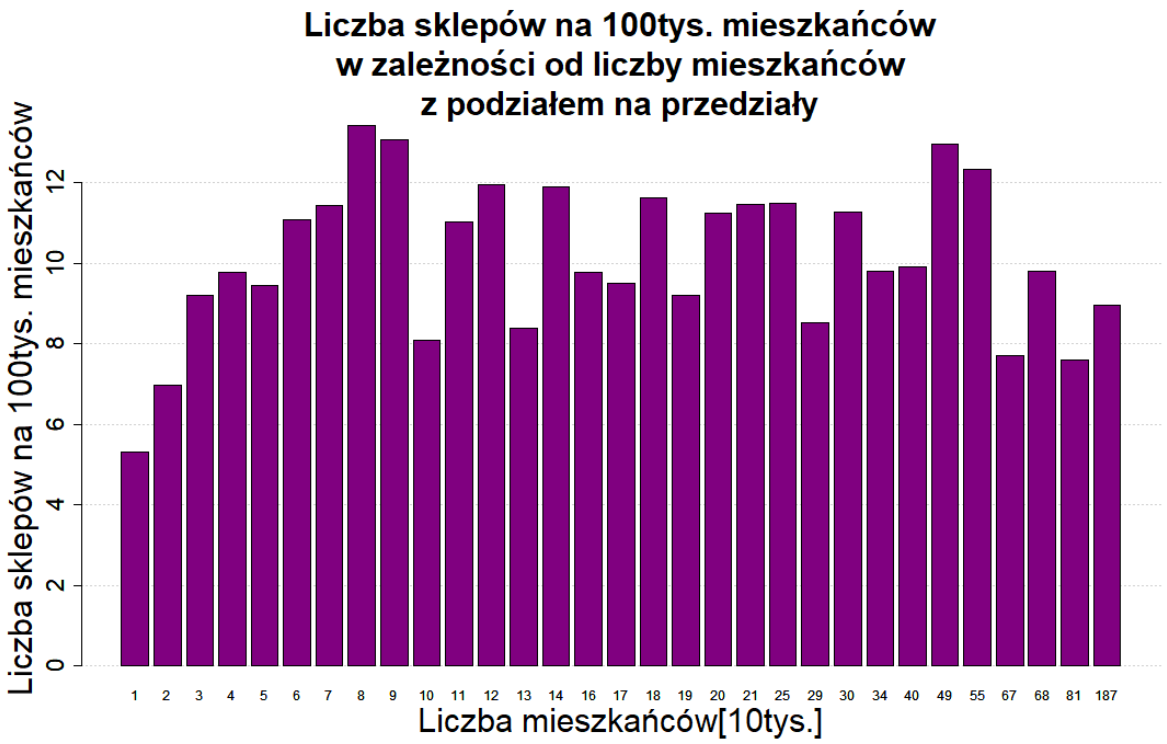




Podział na regularne przedziały

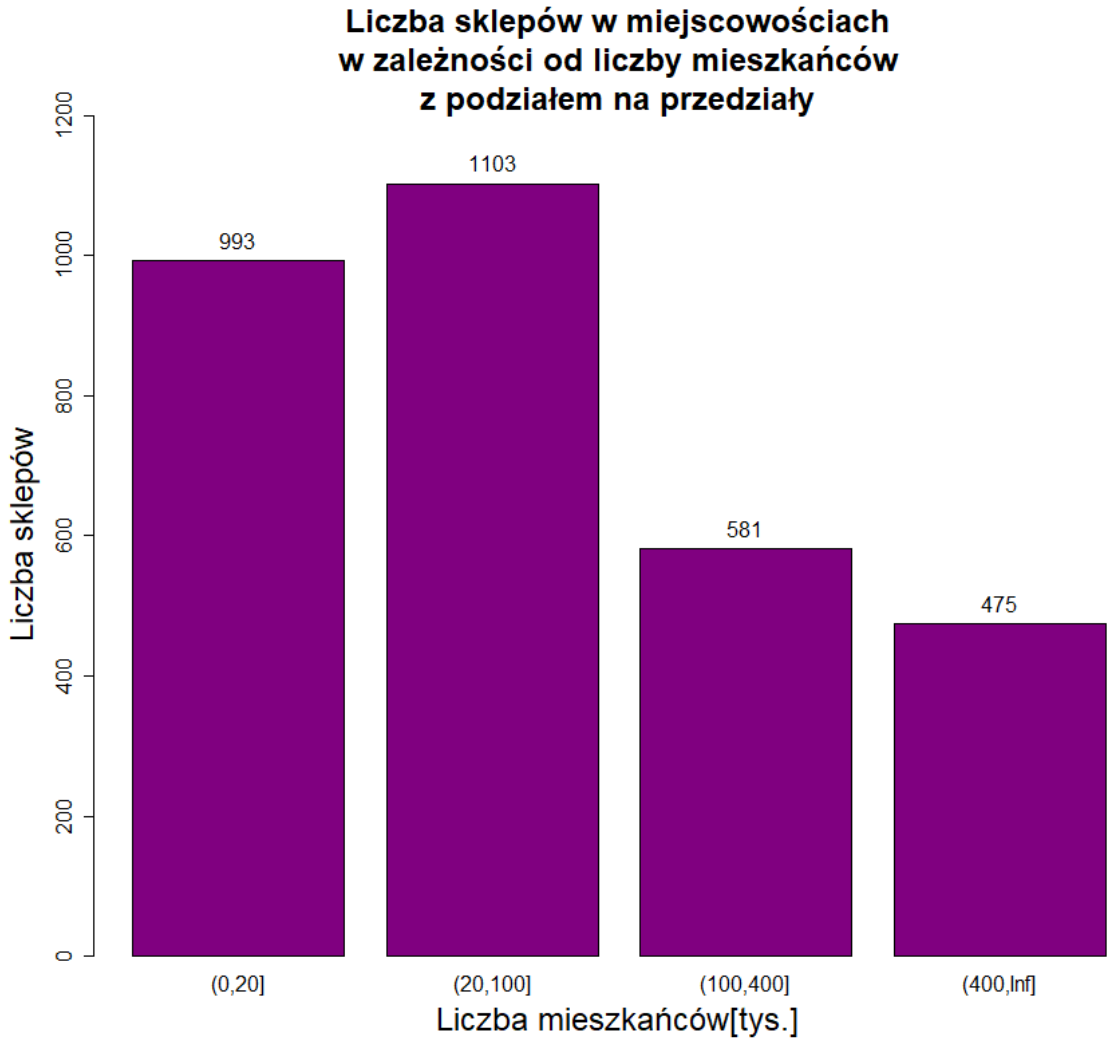
Poniższe wykresy prezentują liczbę sklepów oraz liczbę sklepów na 100 tys. mieszkańców w postaci przedziałów gromadzących gminy o liczbie mieszkańców z przedziału $(x - 10000, x]$ mieszkańców, gdzie x to liczby umieszczone pod każdym z słupków. Można zauważyć, że najwięcej sklepów jest w niewielkich gminach. Jest to spowodowane ilością tych gmin, co można wnioskować po drugim wykresie, gdzie niewielkie gminy mają najniższe wartości.

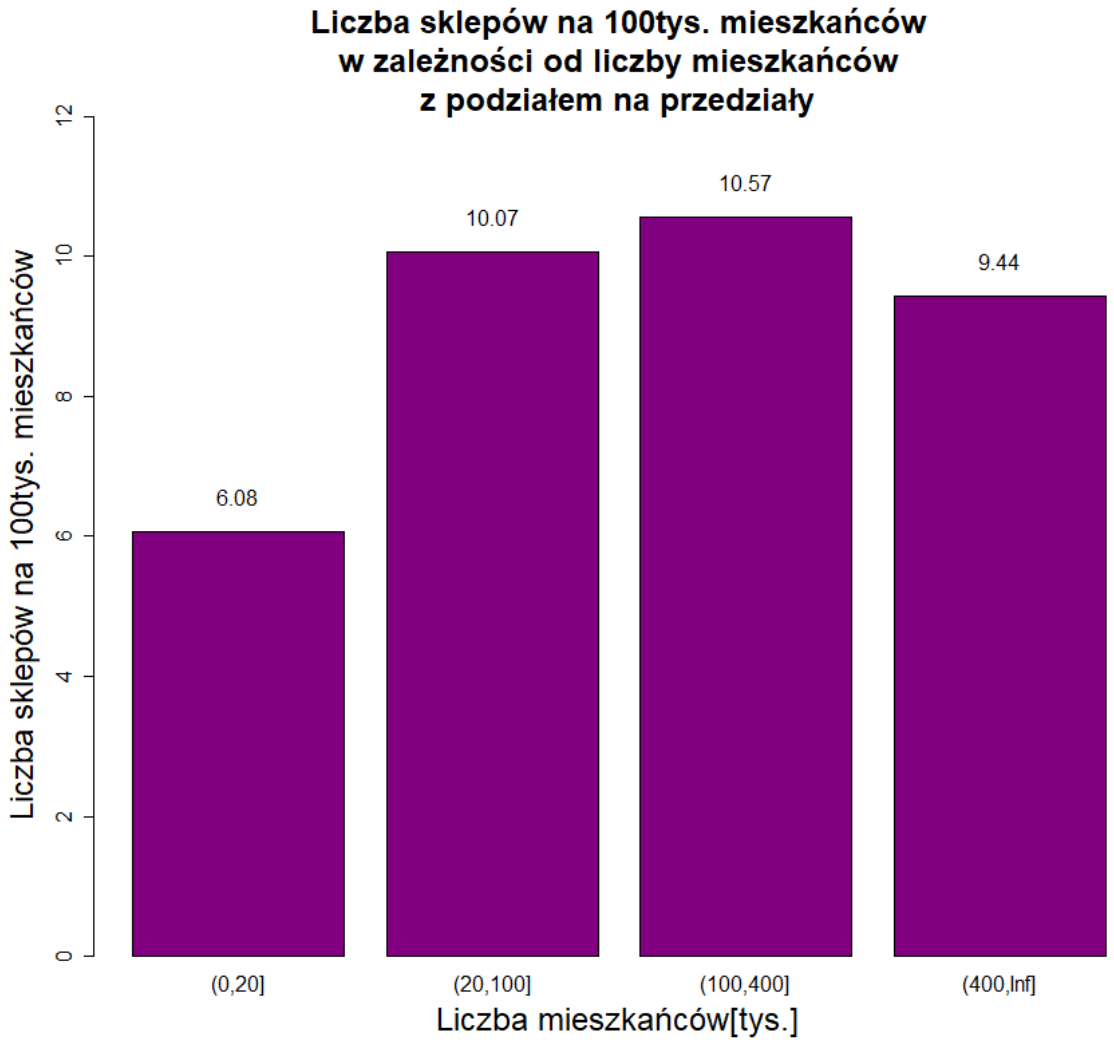

Liczba sklepów w zależności od liczby mieszkańców
z podziałem na przedziały
Liczba sklepów
0
100
200
300
400
500
1 2 3 4 5 6 7 8 9 10 11 12 13 14 16 17 18 19 20 21 25 29 30 34 40 49 55 67 68 81 187
Liczba mieszkańców[10tys.]


Liczba sklepów na 100tys. mieszkańców
w zależności od liczby mieszkańców
z podziałem na przedziały
Liczba sklepów na 100tys. mieszkańców
0
2
4
6
8
10
12
1 2 3 4 5 6 7 8 9 10 11 12 13 14 16 17 18 19 20 21 25 29 30 34 40 49 55 67 68 81 187
Liczba mieszkańców[10tys.]

Autorski podział na przedziały

Poniżej zaprezentowano jeszcze 2 wykresy symbolizujące podzielenie gmin na kilka kategorii: miasteczka i wsie, małe miasta, średnie miasta i duże miasta. Przedziały zostały dobrane wedle uznania autora i stanowią ciekawostkę dopełniającą obraz analizy.





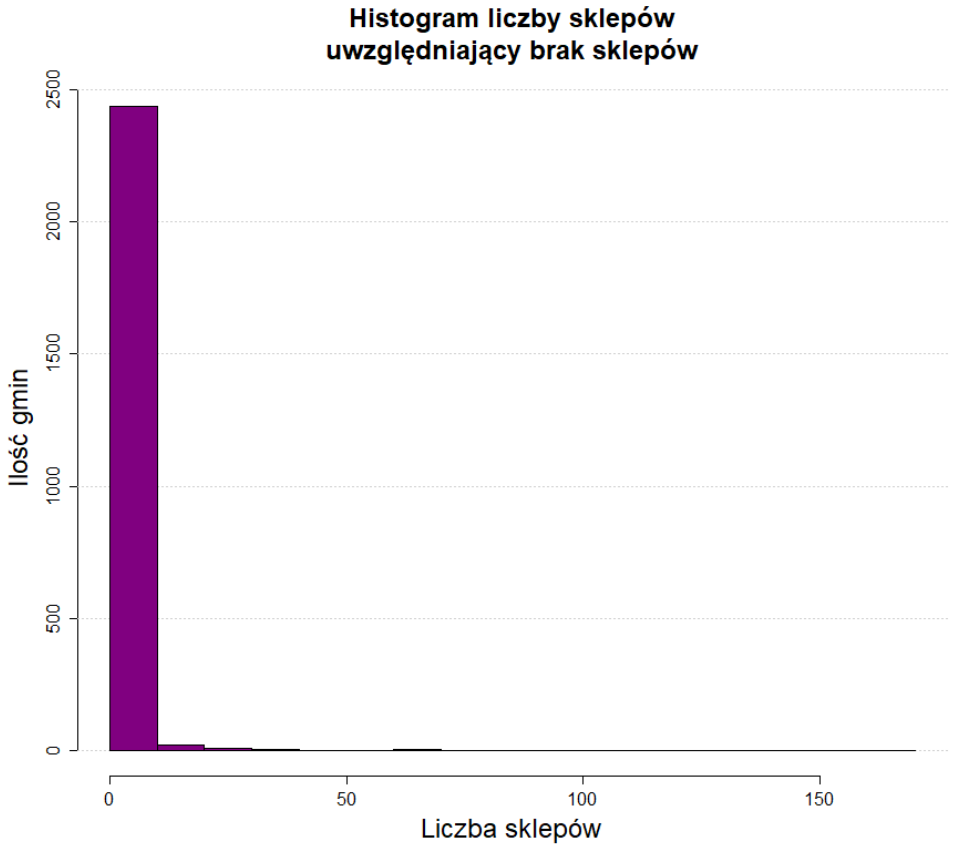
Badanie korelacji i zależności danych

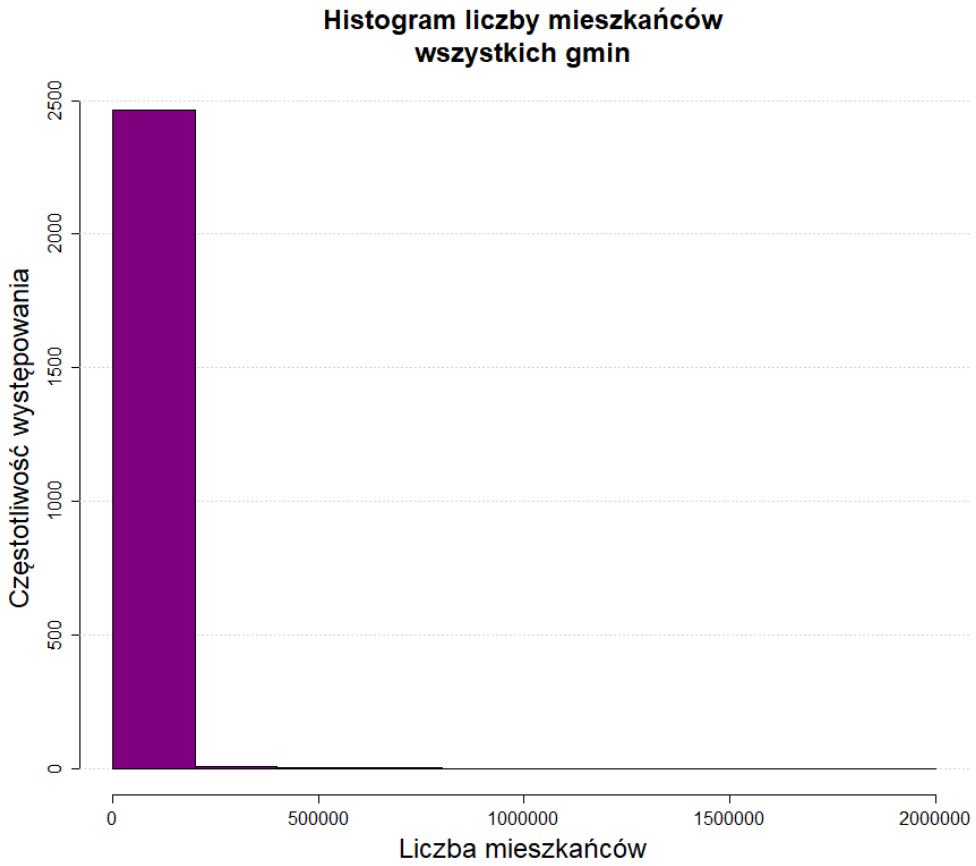
Poniżej znajduje się opracowanie na temat korelacji i zależności liczby sklepów sieci Biedronka oraz gęstości ich rozmieszczenia od liczby mieszkańców gmin w Polsce.

Badanie histogramów

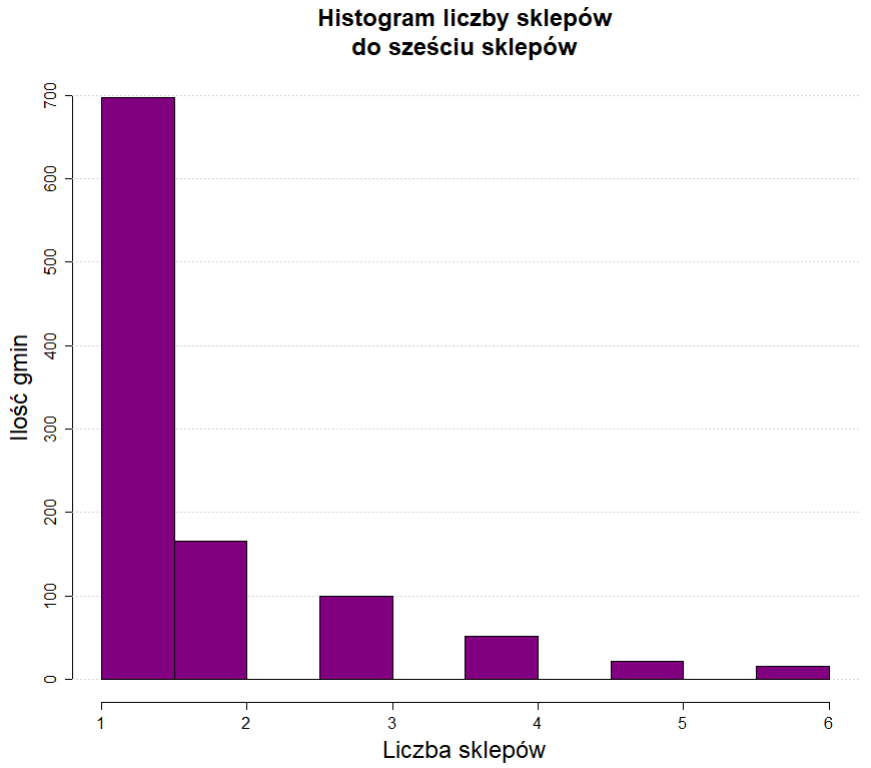
Histogramy dla obu badanych zmiennych losowych są definitywnie asymetryczne, nie przypominają rozkładu normalnego. Zdecydowana większość gmin posiada niewielką liczbę sklepów jak i niewielką liczbę mieszkańców.

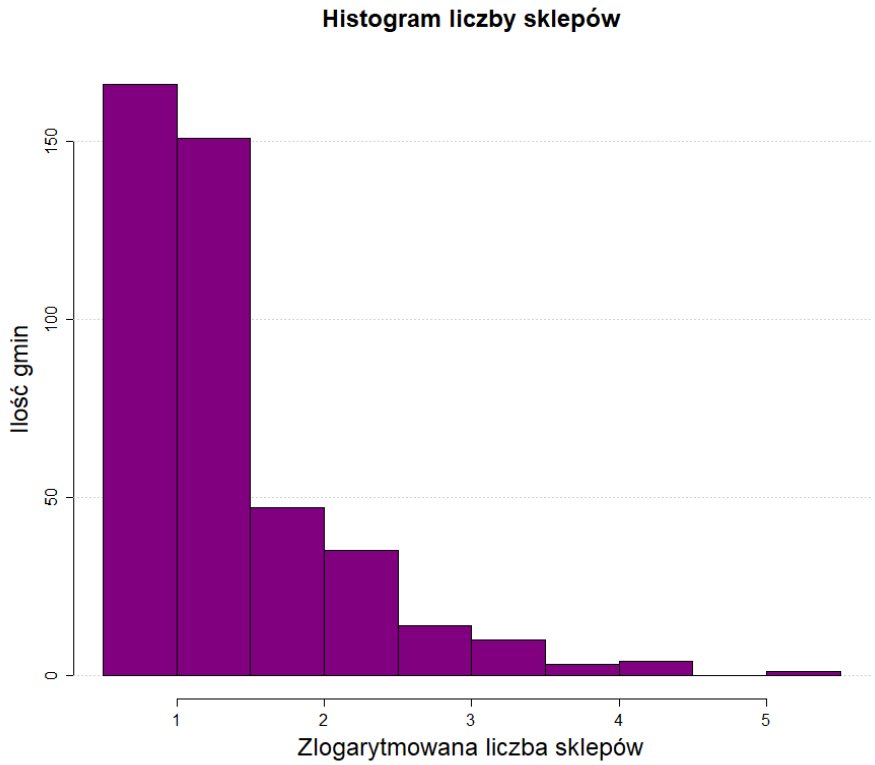
Histogramy bez skalowania i wykluczania części wartości niewiele pokazują:


Histogram liczby sklepów
uwzględniający brak sklepów
Ilość gmin
0
500
1000
1500
2000
2500
0
50
100
150
Liczba sklepów

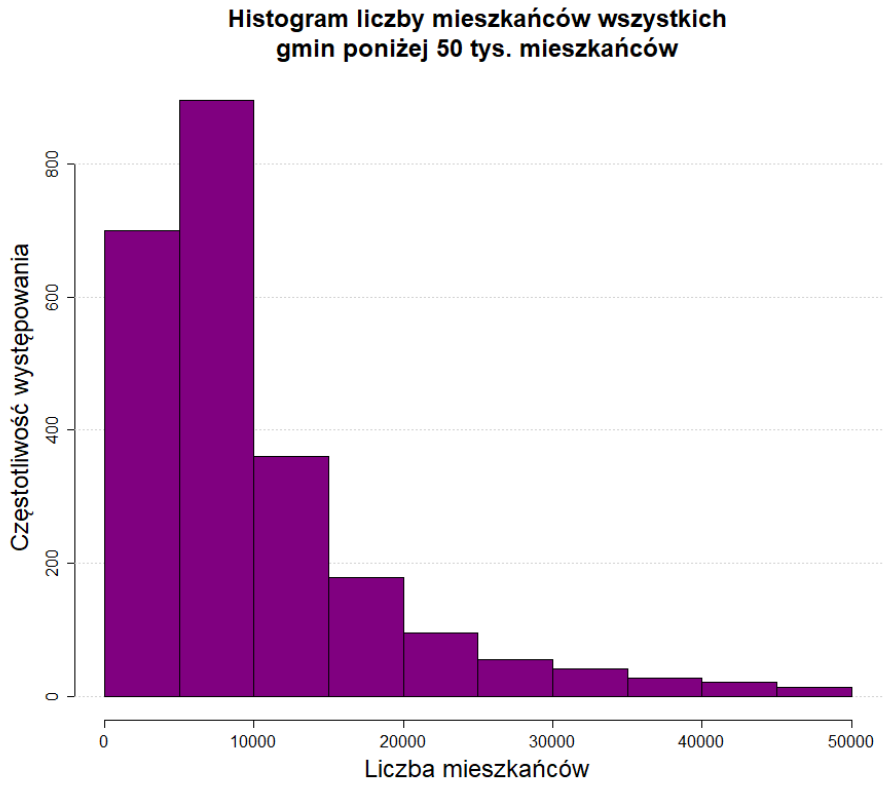

Histogram liczby mieszkańców
wszystkich gmin
Częstotliwość występowania
0
500
1000
1500
2000
2500
0
500000
1000000
1500000
2000000
Liczba mieszkańców

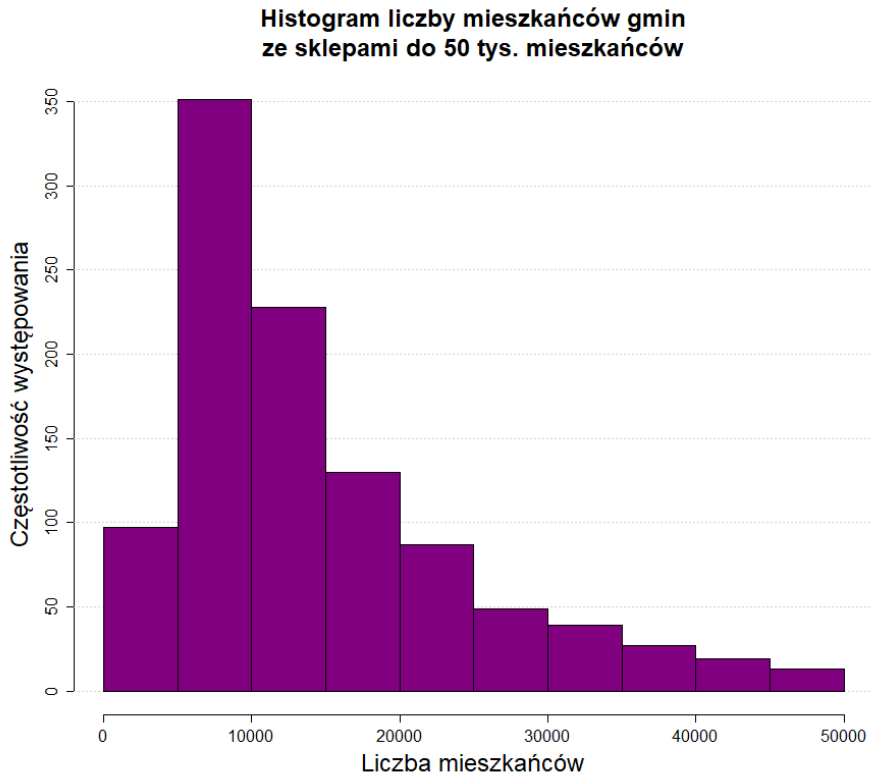
Z tego powodu poniższe histogramy liczby sklepów zostały zmodyfikowane. Pierwszy z nich uwzględnia dane, gdzie liczba sklepów jest nie większa niż 6, a drugi obrazuje zlogarytmowaną liczbę sklepów.



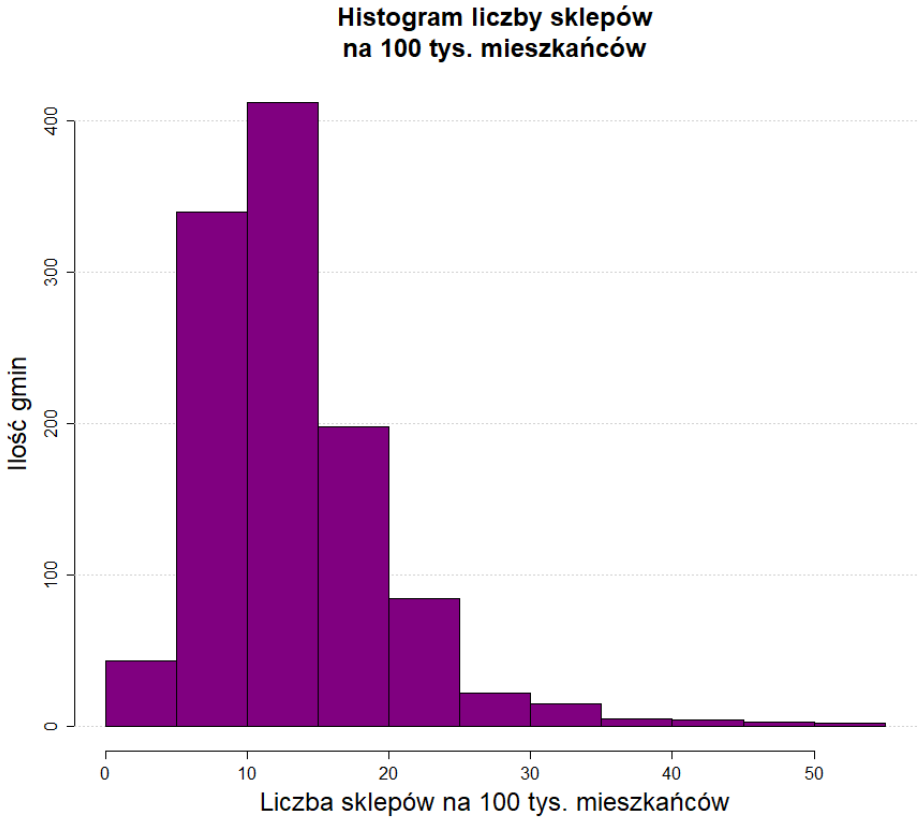


Z tych samych powodów histogramy liczby gmin również zostały zmodyfikowane. Oba zawierają dane tylko dla gmin mających mniej niż 50000 mieszkańców, przy czym pierwszy uwzględnia gminy bez sklepów, a drugi nie.



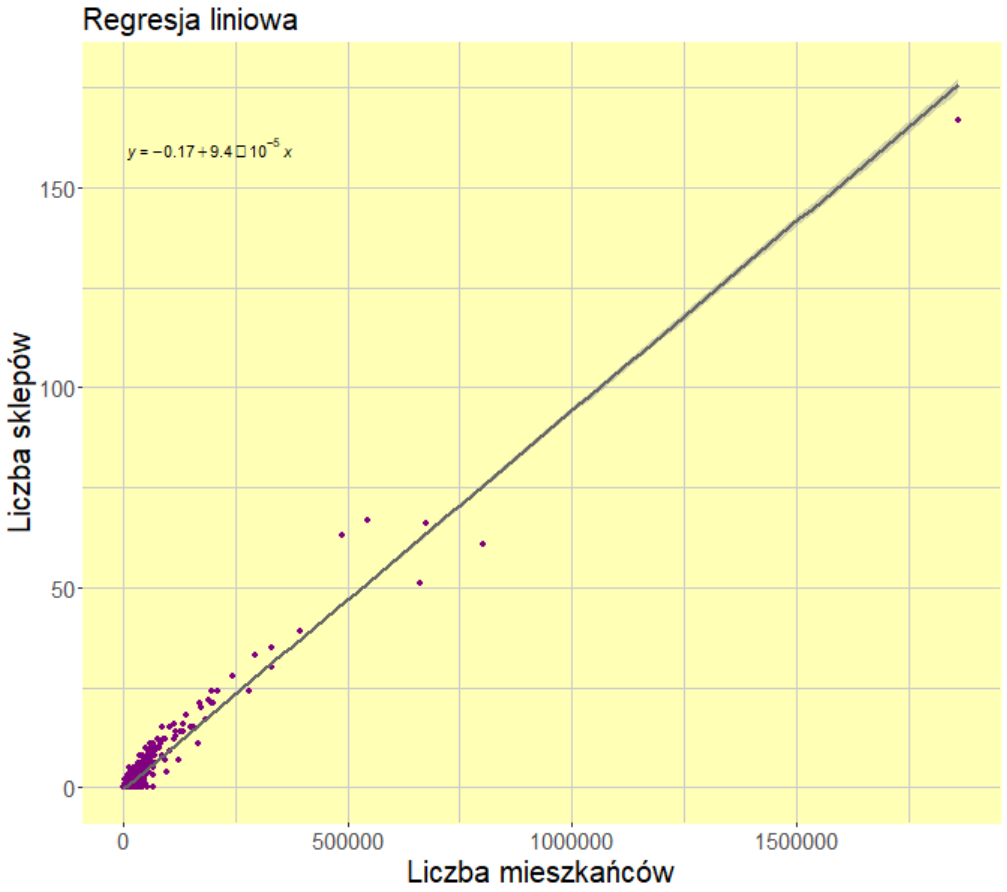


Można zauważyć, że histogramy dla obu badanych zmiennych są dość podobne.
Poniżej zamieszczono ostatni histogram liczby sklepów na 100 tys. mieszkańców. Jest on podobny do histogramu gmin mających poniżej 50000 mieszkańców.



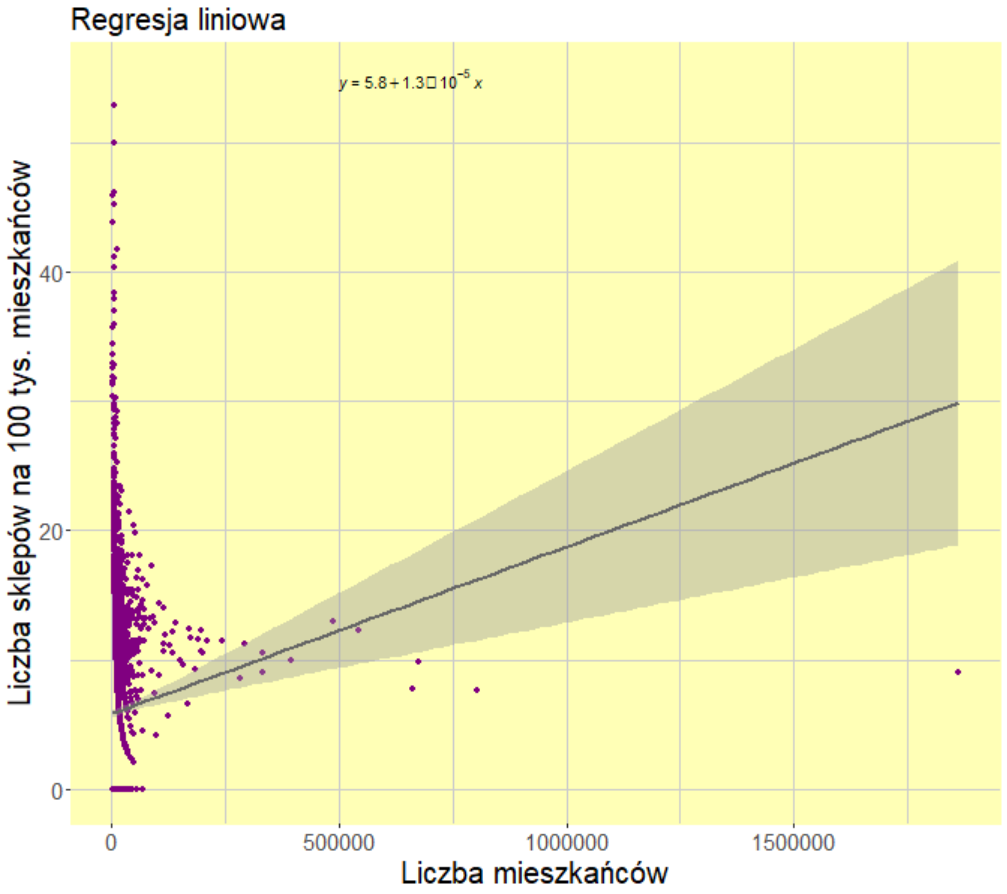
Wzrost liczby sklepów wraz ze wzrostem liczby mieszkańców

Hipoteza ta wydaje się dość intuicyjna, ale należy ją koniecznie sprawdzić przed wyciągnięciem wniosków.
Współczynnik korelacji dla danych zmiennych wynosi 0.9743298, jest on zbliżony do 1. 95% przedział ufności to 0.9722541 do 0.9762521. Te informacje wskazują na to, że zmienne są silnie ze sobą skorelowane. Poniżej przedstawiono wykres regresji liniowej:



Wzrost stosunku liczby sklepów do liczby mieszkańców wraz ze wzrostem liczby mieszkańców

Ciężko powiedzieć coś na temat tej hipotezy bez oparcia w danych. Wyglądają one następująco: współczynnik korelacji wynosi 0.08530242, a 95% przedział ufności to 0.04607287 do 0.12426927. Dane te wskazują na to, że badane zmienne nie są skorelowane. Poniżej zamieszczono prezentujący to wykres:



Wnioski i podsumowanie

Badanie odpowiedziało przecząco na większość pytań autora. Nic nie wskazuje na to aby ilość działających sklepów była zależna od liczby mieszkańców gminy. Pojawiły się za to kolejne pytania, choćby o zależności między ilością sklepów a województwem. Udało się za to potwierdzić, że wraz ze wzrostem liczby mieszkańców w danej gminie - wzrasta liczba Biedronek. Jest to istotna informacja, ponieważ nie tyczy się ona wszystkich sieci w Polsce - choćby Dino za cel obrało sobie małe i średnie miasta.

Podczas projektu udało się zgromadzić i przetworzyć dane, które nie były łatwo dostępne jak przykładowe bloki danych wykorzystywane w celach edukacyjnych. Niestety, funkcje do zbierania danych ze stron sklepów najpewniej nie nadają się do użycia w celu analizy innej sieci ze względu na różnice w budowie

stron internetowych. W przeciwieństwie do skryptów, baza danych zawiera sporo informacji, które można wykorzystać na różne sposoby.

Realizacja projektu była solidnym doświadczeniem edukacyjnym. Mnogość użytych narzędzi pozwoliła autorowi się rozwinąć i przyswoić nowe techniki obróbki danych. Narzędzia takie jak R czy SQLite są uniwersalne i umiejętność ich użytkowania jest bardzo przydatna dla informatyka. Poza tym projekt dostarczył autorowi nowych doświadczeń z zakresu statystyki i analizy danych.

Dokumentacja

Wszystkie użyte pliki znajdują się w skompresowanym folderze dołączonym do tego dokumentu. Poniżej znajduje się tylko kilka najważniejszych z nich:

Communities.py

```
import os


def get_word_from_list(parts : list[str], connector : chr = " "):
    new_word = ""
    for i in range(len(parts)-1):
        new_word = new_word + parts[i] + connector
    return new_word + parts[len(parts)-1]


def get_name_and_type(chain : str):
    refactor_needed = False
    val = chain.strip().split(" ")
    type = None
    if len(val) > 1:
        refactor_needed = True
    if refactor_needed:
        starting_index = 0
        if val[0][0] == 'm' or val[0][0] == 'g':
            starting_index = 1
            if len(val[0]) == 2:
                type = 'miasto'
            elif len(val[0]) == 5:
                type = 'wies'
            else:
                type = 'gmina miejsko-wiejska'
        else:
            type = 'miasto'
        name = get_word_from_list(val[starting_index:])
    else:
        name = val[0]
```

```
        type = 'miasto'
    return name, type


def create_clear_data():
    file_names = os.listdir('Original_data/Communities')
    for i in file_names:
        # first import data from all .csv
        original_data_path = f"Original_data/Communities/{i}"
        clear_data_path = f"Clear_data/Communities/{i}"
        voyvodship = i.replace("\n", "").replace(".csv", "").strip()
        original_data_list = [[], [], []]
        with open(original_data_path, encoding='utf-8-sig') as file:
            for line in file:
                line_data = line.split(";")
                for j in range(3):
                    original_data_list[j].append(line_data[j])
        # then convert it to desired format
        clear_data_list = [[], [], [], [], [], []]
        county = None
        for j in range(len(original_data_list[0])):
            first_col = original_data_list[0][j].strip()
            if first_col[:3] == 'WOJ' or first_col[:5] == 'Miast' or
first_col[:3] == 'Cit':
                county = None
            elif first_col[:6] == 'Powiat':
                fc_list = first_col.split(" ")
                county = get_word_from_list(fc_list[1:])
            else:
                name, type = get_name_and_type(original_data_list[0][j])
                id = int(original_data_list[1][j])
                citizens = int(original_data_list[2][j])
                clear_data_list[0].append(id)
                clear_data_list[1].append(name)
                clear_data_list[2].append(type)
                clear_data_list[3].append(citizens)
                clear_data_list[4].append(voyvodship)
                if county is not None:
                    clear_data_list[5].append(county)
                else:
                    clear_data_list[5].append(name)
        # write it to all .csv
        with open(clear_data_path, "w", encoding='utf8') as file:
            for j in range(len(clear_data_list[0])):
                for k in range(len(clear_data_list)-1):
                    file.write(f"{clear_data_list[k][j]};")
                file.write(f"{clear_data_list[len(clear_data_list)-1][j]}\n")
```

```
        # for c in clear_data_list:  # - uncomment if want to see debugging
print
        #     print(c)
    # finally write it to one big .csv
    if os.path.exists("Clear_data/Communities/all_data.csv"):
        os.remove("Clear_data/Communities/all_data.csv")
    file_names = os.listdir('Clear_data/Communities')
    with open("Clear_data/Communities/all_data.csv", "w", encoding='utf8') as
w_file:
        w_file.write("id;nazwa;typ;liczba_mieszkancow;wojewodztwo;powiat\n")
        for i in file_names:
            clear_data_path = f"Clear_data/Communities/{i}"
            with open(clear_data_path, encoding='utf8') as r_file:
                for line in r_file:
                    w_file.write(line)
```

Data_downloader.py

```
import os.path
from Communities import get_word_from_list
import urllib.request
import urllib.error
import urllib.parse
import sqlite3


def convert_char(c : chr):
    utf8_letters = ['ą', 'ę', 'ć', 'ż', 'ź', 'ó', 'ł', 'ń', 'ś', 'Ć', 'Ż',
'Ź', 'Ó', 'Ł', 'Ś']
    ascii_letters = ['a', 'e', 'c', 'z', 'z', 'o', 'l', 'n', 's', 'C', 'Z',
'Z', 'O', 'L', 'S']
    for i in range(len(utf8_letters)):
        if utf8_letters[i] == c:
            return ascii_letters[i]
    return c


def convert_word(word : str):
    new_word = ""
    for i in range(len(word)):
        new_word = new_word + convert_char(word[i])
    return new_word.lower().replace(" ", "-")


def get_voyvodhip_postcode(voyvodship : str):
```

```
    postcodes = {'Mazowieckie':0, 'Warmińsko-mazurskie':1, 'Podlaskie':1,
'Lubelskie':2, 'Świętokrzyskie':2,
                 'Małopolskie':3, 'Podkarpackie':3, 'Śląskie':4,
'Opolskie':4, 'Dolnośląskie':5, 'Wielkopolskie':6,
                 'Lubuskie':6, 'Zachodniopomorskie':7, 'Pomorskie':8,
'Kujawsko-pomorskie':8, 'Łódzkie':9}
    if voyvodship in postcodes.keys():
        return postcodes.get(voyvodship)
    return None


def create_url(city : str, page_no : int):
    return
f"https://www.biedronka.pl/pl/sklepy/lista,city,{convert_word(city)},page,{p
age_no}"


def get_postcode(line : str):
    return line.strip(" ").replace('<br />', '').replace('\n', '')


def get_street(line : str):
    return line.replace('</span>', '').replace('\n', '').strip()


def get_opening_hours(line : str):
    current_opening_hours = line.replace('</span><br />', '').replace('\n',
'')
    if current_opening_hours[len(current_opening_hours)-1] == 'e':
        return 'Zamknięte'
    return current_opening_hours[-11:]


def read_and_save(community_id, shop_id : int, previous_page_first_address :
str = None):
    lines_counter = 0
    next_is_shop_address = 0
    next_are_opening_hours = 0
    postcodes = []
    addresses = []
    opening_hours = [[] for _ in range(7)]
    need_to_check_next = True
    with open("temp_data.html") as f:
        for line in f:
            if line == '\n':
                pass
            elif line == '            <ul class="shopList">\n':
                lines_counter = 499
```

```
            elif lines_counter > 1:
                lines_counter -= 1
                if line == '      <div class="pagination">\n':
                    lines_counter = 0
                    break
                elif line == '<section class="newShopSearch">':
                    lines_counter = 0
                    need_to_check_next = False
                    break
                elif next_is_shop_address == 1:
                    street = get_street(line)
                    if street == previous_page_first_address:
                        return shop_id, False, previous_page_first_address
                    addresses.append(street)
                    next_is_shop_address = 0
                elif next_is_shop_address == 2:
                    next_is_shop_address = 1
                    postcode = get_postcode(line)
                    postcodes.append(postcode)
                elif line == '                    <span
class="shopAddress">\n':
                    next_is_shop_address = 2
                elif next_are_opening_hours > 0:
                    opening_hours[7 -
next_are_opening_hours].append(get_opening_hours(line))
                    next_are_opening_hours -= 1
                elif line == '                        <b>Godziny
otwarcia:</b><br />\n':
                    next_are_opening_hours = 7
            if lines_counter == 1:
                break
    if len(addresses) == 0:
        return shop_id, False, previous_page_first_address
    all_shops_path = 'Clear_data/Shops/all_shops.csv'
    if not os.path.exists(all_shops_path):
        with open(all_shops_path, 'w', encoding='utf8') as f:
            f.write('id_gminy;id_sklepu;kod_pocztowy;ulica\n')
    opening_hours_path = 'Clear_data/Shops/all_opening_hours.csv'
    if not os.path.exists(opening_hours_path):
        with open(opening_hours_path, 'w', encoding='utf8') as f:

f.write('id_sklepu;poniedzialek;wtorek;sroda;czwartek;piatek;sobota;niedziel
a\n')
    with open(all_shops_path, 'a', encoding='utf8') as as_file:
        with open(opening_hours_path, 'a', encoding='utf8') as oh_file:
            for i in range(len(postcodes)):

as_file.write(f'{community_id};{shop_id};{postcodes[i]};{addresses[i]}\n')
```

```
                oh_list = [opening_hours[j][i] for j in range(7)]
                oh_string = get_word_from_list(oh_list, ';')
                oh_file.write(f'{shop_id};{oh_string}\n')
                shop_id += 1
    return shop_id, need_to_check_next, addresses[0]


if __name__ == '__main__':
    db = sqlite3.connect('Database/shops.db')
    cursor = db.cursor()
    cursor.execute('''
    SELECT id, nazwa, wojewodztwo FROM gminy ORDER BY nazwa,
liczba_mieszkancow DESC
    ''')
    rows = cursor.fetchall()
    db.close()
    shop_id = 1
    all_shops_path = 'Clear_data/Shops/all_shops.csv'
    if os.path.exists(all_shops_path):
        os.rename(all_shops_path, 'Clear_data/Shops/all_shops_old.csv')
    with open(all_shops_path, 'w', encoding='utf8') as f:
        f.write('id_gminy;id_sklepu;kod_pocztowy;ulica\n')
    opening_hours_path = 'Clear_data/Shops/all_opening_hours.csv'
    if os.path.exists(opening_hours_path):
        os.rename(opening_hours_path,
'Clear_data/Shops/all_opening_hours_old.csv')
    with open(opening_hours_path, 'w', encoding='utf8') as f:

f.write('id_sklepu;poniedzialek;wtorek;sroda;czwartek;piatek;sobota;niedziel
a\n')
    prev_name = None
    for r in rows:
        id = r[0]
        name = r[1]
        if name != prev_name:
            prev_name = name
            voyvodship = r[2]
            page_no = 1
            need_to_check_next = True
            first_address = None
            while need_to_check_next:
                url = create_url(name, page_no)
                print(f'Checking another url {url}')
                response = urllib.request.urlopen(url)
                web_data = response.read().decode('UTF-8')
                with open('temp_data.html', 'w') as file:
                    file.write(web_data)
```

```
                shop_id, need_to_check_next, first_address =
read_and_save(id, shop_id, first_address)
                page_no += 1

    print('finished :)')
```

Database_operations.py

```
import sqlite3
import pandas as pd
import os


def execute_all_in(path : str):
    file_names = os.listdir(path)
    files_to_exec = []
    for name in file_names:
        with open(f'{path}/{name}') as file:
            sql_script = file.read()
        files_to_exec.append(sql_script)
    db = sqlite3.connect("Database/shops.db")
    cursor = db.cursor()
    for sql_script in files_to_exec:
        cursor.executescript(sql_script)
    db.commit()
    db.close()


def create_tables():
    execute_all_in('Database/Tables')


def create_views():
    execute_all_in('Database/Views')


def insert_communities_values():
    communities = pd.read_csv("Clear_data/Communities/all_data.csv",
encoding='utf8', delimiter=';')
    db = sqlite3.connect("Database/shops.db")
    communities.to_sql('gminy', db, if_exists='append', index=False)
    db.commit()
    db.close()
```

```
def insert_shop_values():
    opening_hours = pd.read_csv("Clear_data/Shops/all_opening_hours.csv",
encoding='utf8', delimiter=';')
    shops_data = pd.read_csv("Clear_data/Shops/all_shops.csv",
encoding='utf8', delimiter=';')
    db = sqlite3.connect("Database/shops.db")
    opening_hours.to_sql('godziny_otwarcia', db, if_exists='append',
index=False)
    shops_data.to_sql('sklepy', db, if_exists='append', index=False)
    db.commit()
    db.close()


def drop_tables():
    db = sqlite3.connect("Database/shops.db")
    cursor = db.cursor()
    cursor.execute('''
        DROP TABLE gminy
    ''')
    cursor.execute('''
        DROP TABLE sklepy
    ''')
    cursor.execute('''
    DROP TABLE godziny_otwarcia
    ''')
    db.commit()
    db.close()


def drop_views():
    db = sqlite3.connect("Database/shops.db")
    cursor = db.cursor()
    cursor.execute('''
            DROP VIEW liczba_sklepow
        ''')
    cursor.execute('''
            DROP VIEW sklepy_detale
        ''')
    db.commit()
    db.close()


def recreate_base():
    drop_tables()
    create_tables()
    create_views()
    insert_communities_values()
    insert_shop_values()
```