Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки

Звіт

З лабораторної роботи №1 з дисципліни
«Прикладні Задачі Машинного Навчання»

На тему:
«Введення в data science»

Виконав студент *ІП-24, Піддубний Борис Сергійович* *21.03.2024*
(шифр, прізвище, ім'я, по батькові)

Перевірив *Нестерук А. О.*
(прізвище, ім'я, по батькові)

Київ 2024

Лабораторна робота №1
«Введення в data science»

Мета:

Вміти збирати, обробляти та аналізувати дані за допомогою бібліотек мови програмування Python.

Постановка задачі лабораторної роботи №1:

При виконанні лабораторної роботи необхідно виконати наступні дії:

1) На сайті http://www.ukrstat.gov.ua/ оберати дані які для Вас є цікавими, можна використати будь-який ресурс з відкритими даними, та завантажте дані.
2) Знайти математичне сподівання, медіану, моду, дисперсію, середньоквадратичне відхилення (поясніть їх зміст)
3) Візуалізувати завантажені дані за допомогою гістограми.
4) Для цих даних проробити всі дії з пункту колекції Series і DataFrame бібліотеки pandas.
5) Виконати первинну обробку даних
6) Прочитати набір даних катастрофи «Титаніка».
   Набір даних катастрофи «Титаніка» належить до числа найпопулярніших наборів даних машинного навчання і доступний в багатьох форматах, включаючи CSV: https://vincentarelbundock.github.io/Rdatasets/datasets.html.
7) Завантажити набір даних катастрофи «Титаніка» за URL-адресою.
   Якщо у вас є URL-адресу, що представляє набір даних в форматі CSV, то ви можете завантажити його в DataFrame функцією read_csv - припустимо, з GitHub:

```
In [1]: import pandas as pd

In [2]: titanic = pd.read_csv('https://vincentarelbundock.github.io/' +
   ...:     'Rdatasets/csv/carData/TitanicSurvival.csv')
   ...:
```

8) Переглянути рядки набору даних катастрофи «Титаніка».

   Набір даних містить понад 1300 рядків, кожен рядок це інформація про одного пасажира. За даними «Вікіпедії», на борту було приблизно 1317 пасажирів, а 815 з них загинули. Для великих наборів даних при виведенні DataFrame показуються тільки перші 30 рядків, потім йде три крапки «...» і останні 30 рядків. Для економії місця перегляньте перші і останні п'ять рядків за допомогою методів head і tail колекції DataFrame. Обидва методи за замовчуванням повертають п'ять рядків, але число виведених рядків можна передати в аргументі:

```
In [3]: pd.set_option('precision', 2)  # Формат для значений с плавающей точкой

In [4]: titanic.head()
Out[4]:
                      Unnamed: 0 survived     sex    age passengerClass
0    Allen, Miss. Elisabeth Walton      yes  female  29.00            1st
1   Allison, Master. Hudson Trevor      yes    male   0.92            1st
2     Allison, Miss. Helen Loraine       no  female   2.00            1st
3  Allison, Mr. Hudson Joshua Crei       no    male  30.00            1st
4  Allison, Mrs. Hudson J C (Bessi       no  female  25.00            1st

In [5]: titanic.tail()
Out[5]:
                    Unnamed: 0 survived     sex    age  passengerClass
1304       Zabour, Miss. Hileni       no  female  14.50             3rd
1305      Zabour, Miss. Thamine       no  female    NaN             3rd
1306  Zakarian, Mr. Mapriededer       no    male  26.50             3rd
1307        Zakarian, Mr. Ortin       no    male  27.00             3rd
1308         Zimmerman, Mr. Leo       no    male  29.00             3rd
```

   Зверніть увагу: pandas регулює ширину кожного стовпця на підставі самого широкого значення в стовпці або імені стовпця (в залежності від того, яке має більшу ширину); в стовпці age рядки 1305 стоїть значення NaN - ознака відсутнього значення в наборі даних.

9) Налаштувати назви стовпців. Назва першого стовпчика в наборі даних виглядає досить дивно ('Unnamed: 0'). Цю проблему можна вирішити налаштуванням імен стовпців. Замініть 'Unnamed: 0' на 'name' і скоротіть 'passengerClass' до 'class':

```
In [6]: titanic.columns = ['name', 'survived', 'sex', 'age', 'class']

In [7]: titanic.head()
Out[7]:
```

	name	survived	sex	age	class
0	Allen, Miss. Elisabeth Walton	yes	female	29.00	1st
1	Allison, Master. Hudson Trevor	yes	male	0.92	1st
2	Allison, Miss. Helen Loraine	no	female	2.00	1st
3	Allison, Mr. Hudson Joshua Crei	no	male	30.00	1st
4	Allison, Mrs. Hudson J C (Bessi	no	female	25.00	1st

10) Провести простий аналіз даних.

Визначте наймолодшого пасажира, найстаршого, який був середній вік пасажирів та статистику по пасажирам які вижили. Відсортуйте всіх жінок з кают 1-го класу, знайдіть наймолодшу та найстаршу серед них, кількість вижевших Зверніть увагу на розбіжності в значенні count (1046) і кількості рядків даних в наборі даних (1309 - при виклику tail індекс останнього рядка дорівнював 1308). Тільки 1046 рядків даних (значення count) містили значення age. Решта результатів були відсутні і були помічені NaN, як в рядку 1305. При виконанні обчислень бібліотека pandas за замовчуванням ігнорує відсутні дані (NaN).

11) Побудувати гістограму віку пасажирів

Візуалізація - хороший спосіб ближче познайомитися з даними. Pandas містить багато вбудованих засобів візуалізації, реалізованих на базі Matplotlib. Щоб використовувати їх, спочатку включіть підтримку Matplotlib в Ipython.

Гістограма наочно показує розподіл числових даних за діапазоном

значень. Метод hist колекції DataFrame автоматично аналізує дані кожного числового стовпця і будує відповідну гістограму. Щоб переглянути гістограми по кожному числовому стовпці даних, викличте hist для своєї колекції DataFrame:

```
In [11]: histogram = titanic.hist()
```

Набір даних катастрофи «Титаніка» містить тільки один числовий стовпець даних, тому побудуйте діаграму, що покаже гістограму для розподілу вікових груп.

Виконання:

1. Для виконання лабораторної роботи було обрано дані про наявний дохід населення по регіонах України в розрахунку на одну особу за 2013-2021рр. Джерело: https://www.ukrstat.gov.ua/. Дані зображено на таблиці 1.1.

Таблиця 1.1 – Наявний дохід у розрахунку на одну особу

	Наявний дохід у розрахунку на одну особу, грн / Disposable income per capita, UAH								
	2013	2014[1]	2015[1]	2016[1]	2017[1]	2018[1]	2019[1]	2020[1]	2021[1]
Україна	**26719**	**26782**	**31803**	**37080**	**47270**	**58442**	**69140**	**74688**	**90036**
Автономна Республіка Крим	22793	…	…	…	…	…	…	…	…
області									
Вінницька	23001	23422	29637	34931	45436	55734	65503	70939	86274
Волинська	19805	20137	24980	30013	38514	46120	53990	57973	70061
Дніпропетровська	30301	32036	39142	44366	57333	74755	89042	94804	113085
Донецька	31049	26234	21346	20927	25278	33840	39843	42219	49217
Житомирська	21652	22102	27801	32979	42684	52715	62571	66651	79328
Закарпатська	17929	17358	22457	26856	33891	41418	47852	52379	60386
Запорізька	28388	30182	36277	43462	54261	65065	76062	83309	97924
Івано-Франківська	20988	20357	26540	31719	40580	48724	56514	61088	71944
Київська	27391	28443	33956	40127	50664	65623	76232	80274	98771
Кіровоградська	21671	21954	27383	32745	42227	50373	58461	64510	76623
Луганська	25590	19788	15634	13793	16416	21252	24975	27274	32223
Львівська	23138	23595	29542	35325	44981	56592	67353	73092	89441
Миколаївська	23869	23459	29342	34971	45356	55469	64700	69884	85575

Таблиця 1.1 - продовження

	Наявний дохід у розрахунку на одну особу, грн / Disposable income per capita, UAH								
області	2013	2014[1]	2015[1]	2016[1]	2017[1]	2018[1]	2019[1]	2020[1]	2021[1]
Одеська	25572	24242	32385	39132	50111	63153	75288	82007	96851
Полтавська	25371	26196	31997	37938	48663	61649	72843	78813	95770
Рівненська	21165	21781	26708	31295	40325	48184	55917	59350	70826
Сумська	23559	23938	30572	36084	45852	55829	65932	71955	87410
Тернопільська	18994	18401	24040	28195	36204	43577	50536	55776	67467
Харківська	26098	26274	32198	38197	48370	56421	66547	75923	92746
Херсонська	21724	20728	27880	32968	41695	50195	58129	63853	76532
Хмельницька	22789	22686	29292	34395	43638	50640	58934	65411	78500
Черкаська	21633	21761	26970	32327	41854	50600	59626	64852	79621
Чернівецька	19438	18476	23929	28361	36215	42762	49142	54178	64130
Чернігівська	23600	23093	28440	33231	42501	51213	59972	65815	76777
м.Київ	55842	62715	76514	92254	118208	143676	173677	182547	225321
м.Севастополь	26584	…	…	…	…	…	…	…	…

[1] Без урахування тимчасово окупованої території Автономної Республіки Крим, м. Севастополя та частини тимчасово окупованих територій у Донецькій та Луганській областях / [1] Data exclude the temporarily occupied territories of the Autonomous Republic of Crimea, the city of Sevastopol and a part of temporarily occupied territories in the Donetsk and Luhansk regions.

1) Для завантаження даних з файлу .xlsx (excel таблиця) прописуємо наступний код:

```
In:
import pandas as pd
import os

current_dir = os.getcwd()

# Files should be located int the parent directory
file_path = os.path.join(current_dir, '..', "statistics.xlsx")
raw_df = pd.read_excel(file_path)

raw_df.head()
```

Out:

	Регіони та області	2013	2014	2015	2016	2017	2018	2019	2020	2021
0	Україна	26719	26782	31803	37080	47270	58442	69140	74688	90036
1	Автономна Республіка Крим	22793	...	...	...	...	...	...	...	...
2	Вінницька	23001	23422	29637	34931	45436	55734	65503	70939	86274
3	Волинська	19805	20137	24980	30013	38514	46120	53990	57973	70061
4	Дніпропетровська	30301	32036	39142	44366	57333	74755	89042	94804	113085

рис. 1. 1 – результат роботи коду

2. Для отримання статистичних даних по регіонам виключаємо рядок з середнім значенням по Україні та перетворюємо текст на цифри.

```
In:
# Exclude the row for 'України'
regions_df = raw_df[raw_df['Регіони та області'] != 'України']

# Select only numeric columns and convert them to numeric type
numeric_df = regions_df.iloc[:, 1:].apply(pd.to_numeric, errors='coerce')

statistics_df = pd.DataFrame({
    'mean': numeric_df.mean(),
    'median': numeric_df.median(),
    'mode': numeric_df.apply(lambda x: x.mode()[0] if len(x.mode()) == 1 else
None),
    'std_dev': numeric_df.std(),
    'variance': numeric_df.var()
})
```

```
statistics_df
```

Out:

	mean	median	mode	std_dev	variance
2013	24880.464286	23348.5	None	6890.397056	4.747757e+07
2014	24851.538462	23257.5	None	8506.480674	7.236021e+07
2015	30260.192308	28866.0	None	10594.586603	1.122453e+08
2016	35525.807692	33813.0	None	13229.936822	1.750312e+08
2017	45327.961538	43161.0	None	17079.870588	2.917220e+08
2018	55539.269231	51964.0	None	20887.641398	4.362936e+08
2019	65337.730769	61271.5	None	25518.680870	6.512031e+08
2020	70752.461538	66233.0	None	26597.268320	7.074147e+08
2021	85109.192308	79474.5	None	33182.183005	1.101057e+09

рис. 2. 1 – результат роботи коду, виведені статистичні дані

3. Для візуалізації даних про доходи регіонів на гістограмі використаємо бібліотеку matplotlib, також змінемо індекс для DataFrame для більшої зрозумілості коду та перетворимо дані в цифровий формат. Для прикладу наведу графіки за 2013 та 2015 роки.

```
In:
df = raw_df.reset_index(drop=True)
df.set_index('Регіони та області', inplace=True)
df = df.apply(pd.to_numeric, errors='coerce')

import matplotlib.pyplot as plt

def plot_year(year: int) -> None:
    plt.figure(figsize=(10, 6))
    df[year].plot(kind='bar', edgecolor='black', color='orange')
    plt.title(f'Disposable Income per Capita (Excluding Temporarily Occupied
Territories) in {year}')
    plt.xlabel('Regions')
    plt.ylabel('Value (UAH)')
    plt.xticks(rotation=45, ha='right')
    plt.tight_layout()
    plt.show()
```

```
plot_year(2013)
plot_year(2015)

Out:
```

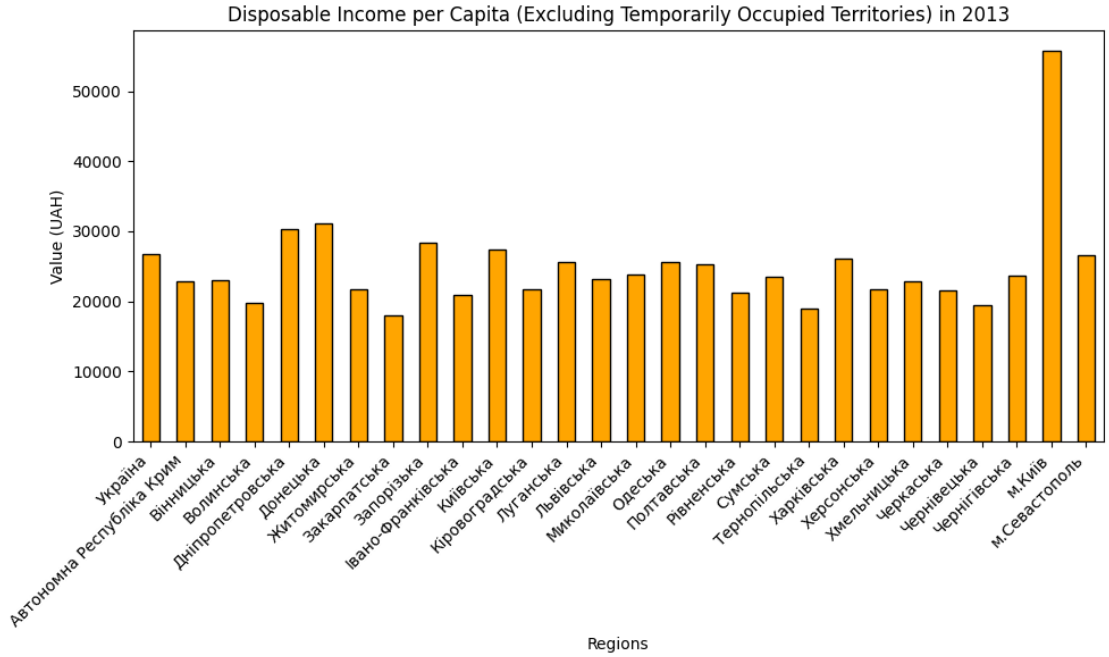

рис 3. 1 дохід регіонів України за 2013 рік

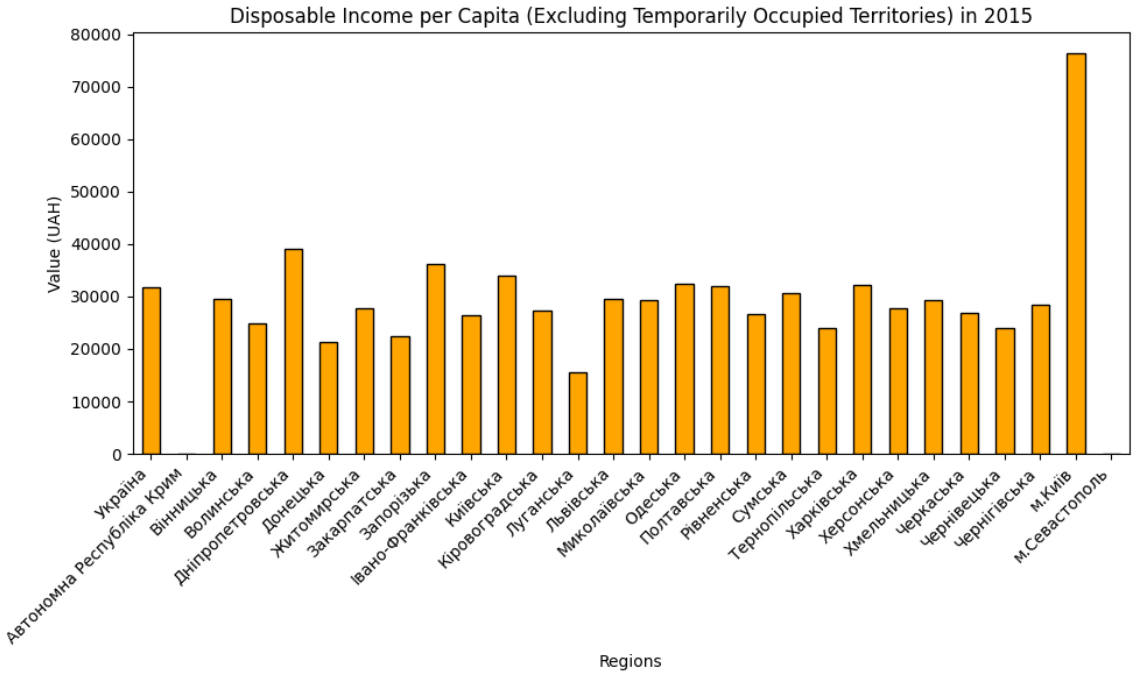

рис 3. 2 дохід регіонів України за 2015 рік

4. Колекція Series та DataFrame

Series:

1) Створення колекціх Series з базовими індексами

```
In:
series = pd.Series(raw_df[2013])
series.head()

Out:
0    26719
1    22793
2    23001
3    19805
4    30301
Name: 2013, dtype: int64
```

2) Звернення до елементів Series

```
In:
series[0]

Out:
26719
```

3) Обчислення описових статистик для Series

```
In:
series.describe()

Out:
count       28.000000
mean     24880.464286
std       6890.397056
min      17929.000000
25%      21647.250000
50%      23348.500000
75%      26219.500000
max      55842.000000
Name: 2013, dtype: float64
```

4) Створення колекції Series з нестандартними індексами

```
In:
```

```
data = {"Україна": 26719, "Київ": 55842}
new_series = pd.Series(data)
new_series

Out:
Україна    26719
Київ       55842
dtype: int64
```

5) Звернення до елементів Series з використанням нестандартних індексів

```
In:
new_series["Київ"]

Out:
55842

In:
new_series[1]

Out:
FutureWarning: Series.__getitem__ treating keys as positions is deprecated. In a future version, integer keys will always be treated as labels (consistent with DataFrame behavior). To access a value by position, use `ser.iloc[pos]`
  new_series[1]

55842
```

DataFrame:

1) Звернення до стовпців DataFrame

```
In:
df_2021 = raw_df[2021]
df_2021.head()

Out:
0     90036
1         …
2     86274
3     70061
```

```
4    113085
Name: 2021, dtype: object
```

2) Вибір рядків з використанням атрибутів `loc` і `iloc`

```
In:
df_2013_first_region = raw_df.iloc[0, 1]
df_2013_first_region

Out:
26719
```

```
In:
df_2015_ukraine = raw_df.loc[raw_df['Регіони та області'] == 'Україна', 2015]
df_2015_ukraine

Out:
0    31803
Name: 2015, dtype: object
```

```
In:
df_all_years_five_regions = raw_df.iloc[1:5]
df_all_years_five_regions

Out:
```

	Регіони та області	2013	2014	2015	2016	2017	2018	2019	2020	2021
1	Автономна Республіка Крим	22793	...	...	...	...	...	...	...	...
2	Вінницька	23001	23422	29637	34931	45436	55734	65503	70939	86274
3	Волинська	19805	20137	24980	30013	38514	46120	53990	57973	70061
4	Дніпропетровська	30301	32036	39142	44366	57333	74755	89042	94804	113085

рис. 4. 1 – результати роботи iloc

```
In:
df_two_regions = raw_df.iloc[[0, 1]]
df_two_regions

Out:
```

	Регіони та області	2013	2014	2015	2016	2017	2018	2019	2020	2021
0	Україна	26719	26782	31803	37080	47270	58442	69140	74688	90036
1	Автономна Республіка Крим	22793	...	...	...	...	...	...	...	...

рис. 4. 2 – результати роботи iloc з подвійними дужками

3) Вибір підмножин рядків і стовпців

```
In:
df_compare_two_regions_four_years = raw_df.iloc[[20, 26], :5]
df_compare_two_regions_four_years

Out:
```

	Регіони та області	2013	2014	2015	2016
20	Харківська	26098	26274	32198	38197
26	м.Київ	55842	62715	76514	92254

рис. 4. 3 – результати роботи iloc

4) Логічне індексування

```
In:
df_last_three_years = df.loc[:, 2019:2021]
df_last_three_years.head()

Out:
```

	2019	2020	2021
Регіони та області			
Україна	69140.0	74688.0	90036.0
Автономна Республіка Крим	NaN	NaN	NaN
Вінницька	65503.0	70939.0	86274.0
Волинська	53990.0	57973.0	70061.0
Дніпропетровська	89042.0	94804.0	113085.0

рис. 4. 4 – Усі доходи регіонів за останні три роки

```
In:
df_rich_regions_last_three_years = df_last_three_years[df_last_three_years >=
80000]
```

```
df_rich_regions_last_three_years.head()
```

Out:

Регіони та області	2019	2020	2021
Україна	NaN	NaN	90036.0
Автономна Республіка Крим	NaN	NaN	NaN
Вінницька	NaN	NaN	86274.0
Волинська	NaN	NaN	NaN
Дніпропетровська	89042.0	94804.0	113085.0

рис. 4. 5 – Перші 5 регіонів, показано ті, що мають або не мають доходу більше ніж 80000 за останні три роки

5) Звернення до конкретного осередку DataFrame по рядку і стовпцю

```
In:
value = raw_df.at[0, 2021]
value

Out:
90036

In:
raw_df.at[0, 2021] = 102003
raw_df.head()
```

Out:

	Регіони та області	2013	2014	2015	2016	2017	2018	2019	2020	2021
0	Україна	26719	26782	31803	37080	47270	58442	69140	74688	102003
1	Автономна Республіка Крим	22793	...	...	...	...	...	...	...	...
2	Вінницька	23001	23422	29637	34931	45436	55734	65503	70939	86274
3	Волинська	19805	20137	24980	30013	38514	46120	53990	57973	70061
4	Дніпропетровська	30301	32036	39142	44366	57333	74755	89042	94804	113085

рис. 4. 6 – Середній прибуток по Україні змінено на інший

```
In:
raw_df.at[0, 2021] = 90036
raw_df.head()
```

Out:

	Регіони та області	2013	2014	2015	2016	2017	2018	2019	2020	2021
0	Україна	26719	26782	31803	37080	47270	58442	69140	74688	90036
1	Автономна Республіка Крим	22793	...	...	...	...	...	...	...	...
2	Вінницька	23001	23422	29637	34931	45436	55734	65503	70939	86274
3	Волинська	19805	20137	24980	30013	38514	46120	53990	57973	70061
4	Дніпропетровська	30301	32036	39142	44366	57333	74755	89042	94804	113085

рис. 4. 7 – Середній прибуток по Україні змінено на початковий

6) Описова статистика

In:
```
raw_df.describe()
```

Out:

	2013
count	28.000000
mean	24880.464286
std	6890.397056
min	17929.000000
25%	21647.250000
50%	23348.500000
75%	26219.500000
max	55842.000000

рис. 4. 7 – Статистичні дані обраховані тільки для 2013р.

In:
```
df.describe()
```

Out:

	2013	2014	2015	2016	2017	2018	201
count	28.000000	26.000000	26.000000	26.000000	26.000000	26.000000	26.00000
mean	24880.464286	24851.538462	30260.192308	35525.807692	45327.961538	55539.269231	65337.73076
std	6890.397056	8506.480674	10594.586603	13229.936822	17079.870588	20887.641398	25518.68087
min	17929.000000	17358.000000	15634.000000	13793.000000	16416.000000	21252.000000	24975.00000
25%	21647.250000	20986.250000	26582.000000	31401.000000	40388.750000	48319.000000	56066.25000
50%	23348.500000	23257.500000	28866.000000	33813.000000	43161.000000	51964.000000	61271.50000
75%	26219.500000	26224.500000	31948.500000	37723.500000	48095.000000	57979.500000	68693.25000
max	55842.000000	62715.000000	76514.000000	92254.000000	118208.000000	143676.000000	173677.00000

рис. 4. 8 – Статистичні дані обраховані для всіх років

7) Транспонування DataFrame з використанням атрибута T

```
In:
raw_df.T

Out:
```

	0	1	2	3	4	5	6	
Регіони та області	Україна	Автономна Республіка Крим	Вінницька	Волинська	Дніпропетровська	Донецька	Житомирська	Закарпатсь
2013	26719	22793	23001	19805	30301	31049	21652	179
2014	26782	...	23422	20137	32036	26234	22102	173
2015	31803	...	29637	24980	39142	21346	27801	224
2016	37080	...	34931	30013	44366	20927	32979	268
2017	47270	...	45436	38514	57333	25278	42684	338
2018	58442	...	55734	46120	74755	33840	52715	414
2019	69140	...	65503	53990	89042	39843	62571	478
2020	74688	...	70939	57973	94804	42219	66651	523
2021	90036	...	86274	70061	113085	49217	79328	603

10 rows × 28 columns

рис. 4. 9 – Транспоновані дані

8) Сортування рядків за індексами

In:

```
raw_df.sort_index(ascending=False).head()
```

Out:

	Регіони та області	2013	2014	2015	2016	2017	2018	2019	2020	2021
27	м.Севастополь	26584	...	...	...	...	...	...	...	...
26	м.Київ	55842	62715	76514	92254	118208	143676	173677	182547	225321
25	Чернігівська	23600	23093	28440	33231	42501	51213	59972	65815	76777
24	Чернівецька	19438	18476	23929	28361	36215	42762	49142	54178	64130
23	Черкаська	21633	21761	26970	32327	41854	50600	59626	64852	79621

рис. 4. 10 – Дані за таблиці у зворотному порядку

5. Для коректного розуміння та представлення даних потрібно виконати їхню первину обробку.

1) Встановити колонку «Регіони та області» в якості індексу

In:

```
# reset index and drop old index
df = raw_df.reset_index(drop=True)

# Sets the desired column as the index (inplace modification)
df.set_index('Регіони та області', inplace=True)

df.head()
```

Out:

	2013	2014	2015	2016	2017	2018	2019	2020	2021
Регіони та області									
Україна	26719	26782	31803	37080	47270	58442	69140	74688	90036
Автономна Республіка Крим	22793	...	...	...	...	...	...	...	...
Вінницька	23001	23422	29637	34931	45436	55734	65503	70939	86274
Волинська	19805	20137	24980	30013	38514	46120	53990	57973	70061
Дніпропетровська	30301	32036	39142	44366	57333	74755	89042	94804	113085

рис. 5. 1 – таблиці з колонкою «регіони та області» в якості індексу

2) Перетворити строки в числа або значення None для обрахунку статистичних даних.

```
In:
df = df.apply(pd.to_numeric, errors='coerce')

df.head()
```

Out:

Регіони та області	2013	2014	2015	2016	2017	2018	2019	2020	2021
Україна	26719	26782.0	31803.0	37080.0	47270.0	58442.0	69140.0	74688.0	90036.0
Автономна Республіка Крим	22793	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Вінницька	23001	23422.0	29637.0	34931.0	45436.0	55734.0	65503.0	70939.0	86274.0
Волинська	19805	20137.0	24980.0	30013.0	38514.0	46120.0	53990.0	57973.0	70061.0
Дніпропетровська	30301	32036.0	39142.0	44366.0	57333.0	74755.0	89042.0	94804.0	113085.0

рис. 5. 2 – таблиці з числовими значеннями

3) Зберігти оброблені дані в новий файл з розширенням .csv

```
df.to_csv("../processed-regions.csv", index=False)
```

рис. 5. 3 – Source Control показує новий файл у теці

6. Інформація про потерпілих на Титаніку зберігається на github pages у .csv форматі
7. Завантажимо дані за допомогою бібліотеки pandas

```
In:
import pandas as pd

titanic_df =
pd.read_csv('https://vincentarelbundock.github.io/Rdatasets/csv/carData/TitanicSurvival.csv')
```

8. Прочитаємо дані з голови та хвоста файлу

```
In:
titanic_df.head()

Out:
```

	rownames	survived	sex	age	passengerClass
0	Allen, Miss. Elisabeth Walton	yes	female	29.0000	1st
1	Allison, Master. Hudson Trevor	yes	male	0.9167	1st
2	Allison, Miss. Helen Loraine	no	female	2.0000	1st
3	Allison, Mr. Hudson Joshua Crei	no	male	30.0000	1st
4	Allison, Mrs. Hudson J C (Bessi	no	female	25.0000	1st

рис. 8. 1 – перші п’ять потерпілих зі списку

```
In:
titanic_df.tail()

Out:
```

	rownames	survived	sex	age	passengerClass
1304	Zabour, Miss. Hileni	no	female	14.5	3rd
1305	Zabour, Miss. Thamine	no	female	NaN	3rd
1306	Zakarian, Mr. Mapriededer	no	male	26.5	3rd
1307	Zakarian, Mr. Ortin	no	male	27.0	3rd
1308	Zimmerman, Mr. Leo	no	male	29.0	3rd

рис. 8. 2 –останні п’ять потерпілих зі списку

9. Змінемо назву колонок на зрозумілі

```
In:
titanic_df.columns = ['name', 'survived', 'sex', 'age', 'class']

titanic_df

Out:
```

	name	survived	sex	age	class
0	Allen, Miss. Elisabeth Walton	yes	female	29.0000	1st
1	Allison, Master. Hudson Trevor	yes	male	0.9167	1st
2	Allison, Miss. Helen Loraine	no	female	2.0000	1st
3	Allison, Mr. Hudson Joshua Crei	no	male	30.0000	1st
4	Allison, Mrs. Hudson J C (Bessi	no	female	25.0000	1st
...	...	...	...	...	...
1304	Zabour, Miss. Hileni	no	female	14.5000	3rd
1305	Zabour, Miss. Thamine	no	female	NaN	3rd
1306	Zakarian, Mr. Mapriededer	no	male	26.5000	3rd
1307	Zakarian, Mr. Ortin	no	male	27.0000	3rd
1308	Zimmerman, Mr. Leo	no	male	29.0000	3rd

рис. 9. 1 – перші та останні п’ять людей зі списку з новими колонками

10. Аналіз даних

1) Наймолодший пасажир

```
In:
youngest_passenger = titanic_df.loc[titanic_df['age'].idxmin()]
youngest_passenger

Out:

name        Dean, Miss. Elizabeth Gladys M
survived                               yes
sex                                 female
age                                 0.1667
class                                  3rd
Name: 763, dtype: object
```

2) Найстарший пасажир

```
In:
eldest_passenger = titanic_df.loc[titanic_df['age'].idxmax()]
```

```
eldest_passenger
```

```
Out:
```

```
name        Barkworth, Mr. Algernon Henry W
survived                                yes
sex                                    male
age                                    80.0
class                                   1st
Name: 14, dtype: object
```

3) Середній вік пасажирів

```
In:
average_age = titanic_df['age'].mean()
average_age
```

```
Out:
```

```
29.881134512434034
```

4) Усі жінки з першого класу відсортовані за ім’ям

```
In:
first_class_women = titanic_df[(titanic_df['sex'] == 'female') &
(titanic_df['class'] == '1st')].sort_values(by='name')
first_class_women
```

```
Out:
```

	name	survived	sex	age	class
0	Allen, Miss. Elisabeth Walton	yes	female	29.0	1st
2	Allison, Miss. Helen Loraine	no	female	2.0	1st
4	Allison, Mrs. Hudson J C (Bessi	no	female	25.0	1st
6	Andrews, Miss. Kornelia Theodos	yes	female	63.0	1st
8	Appleton, Mrs. Edward Dale (Cha	yes	female	53.0	1st
...	...	...	...	...	...
311	Wick, Mrs. George Dennick (Mary	yes	female	45.0	1st
314	Widener, Mrs. George Dunton (El	yes	female	50.0	1st
315	Willard, Miss. Constance	yes	female	21.0	1st
319	Wilson, Miss. Helen Alice	yes	female	31.0	1st
322	Young, Miss. Marie Grice	yes	female	36.0	1st

рис. 10. 1 – перші та останні п’ять жінок з першого класу

5) Наймолодша жінка з першого класу

```
In:
youngest_first_class_women =
first_class_women.loc[first_class_women['age'].idxmin()]
youngest_first_class_women

Out:
name        Allison, Miss. Helen Loraine
survived                              no
sex                               female
age                                  2.0
class                                1st
Name: 2, dtype: object
```

6) Найстарша жінка з першого класу

```
In:
eldest_first_class_women =
first_class_women.loc[first_class_women['age'].idxmax()]
eldest_first_class_women

Out:
name        Cavendish, Mrs. Tyrell William
survived                                 yes
sex                                  female
age                                    76.0
```

```
class                                 1st
Name: 61, dtype: object
```

7) Кількість жінок з першого класу, що вижили

```
In:
first_class_women_survive_number =
first_class_women[first_class_women['survived'] == 'yes'].shape[0]
first_class_women_survive_number

Out:
139
```

11. Гістограма розподілу віку

```
In:
import matplotlib.pyplot as plt

titanic_df.hist(column='age', bins=40)

plt.xlabel('Age (years)')
plt.ylabel('Number of Passengers')
plt.title('Age Distribution of Titanic Passengers')

# optional
plt.grid(True)

Out:
```

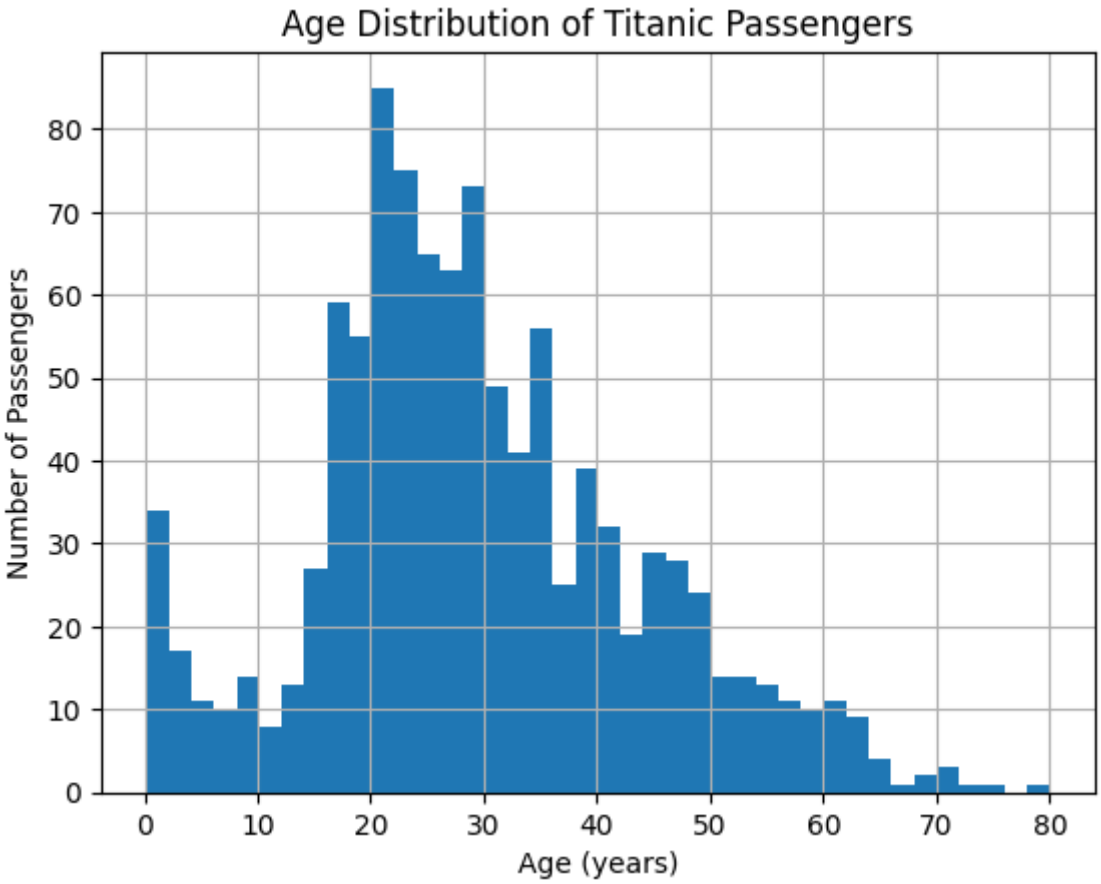

рис. 11. 1 – розподіл віку пасажирів Титаніку

Висновок:

Під час виконання лабораторної роботи я навчився вміти збирати, обробляти та аналізувати дані за допомогою програми Jupyter Notebook на основі бібліотек мови програмування Python. Вищезгадані дії були виконані на основі даних із загальновідкритих джерел. Було оброблено дані про Доходи населення по регіонах України та дані про пасажирів Титаніку.