

# Міністерство освіти і науки України Національний технічний університет України "Київський політехнічний інститут імені Ігоря Сікорського" Факультет інформатики та обчислювальної техніки Кафедра інформатики та програмної інженерії

# Лабораторна робота №6

Аналіз даних з використанням мови Python

**Тема:** Попередня обробка даних в Pandas

Варіант: 1

Виконав	Перевірив:
студент групи ІП-11:	Тимофєєва Ю. С
Панченко С. В.	

# 3MICT

1 Мета лабораторної роботи	6
2 Завдання	7
3 Виконання	8
3.1 Зчитування файлу та зміна назви стовпців	8
3.2 Знаходження проблем з даними та виконання попередньої обробки дан	ИХ
для усунення цих проблем	9
4 Висновок	14

# 1 МЕТА ЛАБОРАТОРНОЇ РОБОТИ

Ознайомитись з операціями попередньої обробки даних Pandas.

### 2 ЗАВДАННЯ

Створити програму,яка виконує наступні завдання, використовуючи файл відповідно до варіанту:

- 1. Читає файл та змінює назви стовпців.
- 2. Знаходить проблеми з даними та виконує попередню обробку даних для усунення цих проблем.

Оформити звіт. Звіт повинен містити:

- -титульний лист;
- —код програми;
- —результати виконання коду.

Продемонструвати роботу програми та відповісти на питання стосовно теоретичних відомостей та роботи програми.

Bapiaнт 1: Version 1.xlsx

### 3 ВИКОНАННЯ

### 3.1 Зчитування файлу та зміна назви стовпців

Для початку імпортуємо модуль pandas та застосуємо функцію pandas.read\_excel для зчитування .xlsx-файлів. Видалимо колонку "Unnamed: 0", оскільки індекси за замовчуванням створюються.

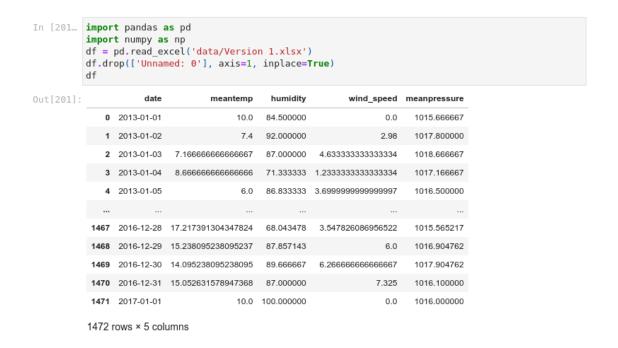


Рисунок 3.1 - Зчитування .xlsx-файлу в датафрейм

Змінимо назвустовпців.

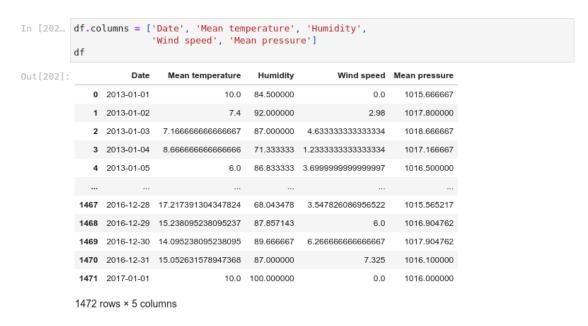


Рисунок 3.2 - Зміна назв стовпців

3.2 Знаходження проблем з даними та виконання попередньої обробки даних для усунення цих проблем

Перевіримо типи даних в стовпцях.

```
pading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js
```

Рисунок 3.3 - Типи даних

Приведемо всі типи в датафреймі в тип object.

```
In [204... df = df.astype('object')
```

Рисунок 3.4 - Приведення до типу object

Перевіримо, чи  $\epsilon$  пропущенні значення у кожному стовпці. для цього застосуємо метод іsna, який повертає для кожного стовпця вектор булевих значень, що позначають, чи елемент пустий. Далі застосуємо метод апу, для того щоб дізнатися, чи  $\epsilon$  хоча б одне пропущене значення.

```
In [205... df.isna().any()

Out[205]: Date False Mean temperature True Humidity True Wind speed True Mean pressure dtype: bool
```

Рисунок 3.5 - Інформація про присутність пустих значень у стовпцях

Розберемося з колонкою дат. При роботі я побачив цікаву деталь, що застосувавши pandas.date\_range я отримував 1462 рядки, а в колонці дат рядків 1472.

```
In [206... r = pd.date_range('2013', '2017')
print(f'{df["Date"].size} vs {len(r)}')
1472 vs 1462
```

Рисунок 3.6 - Підозра на наявність дублікатів в датах

Приберемо дублікати та перевіремо ще раз.

```
In [207... df.drop_duplicates(subset=['Date'], inplace=True)
print(f'{df["Date"].size} vs {len(r)}')
df['Date'] = r

1462 vs 1462
```

Рисунок 3.7 - Прибирання дублікатів

У стовпцях з числами можуть бути символи, що не відповідають загальному виду числа. Приберемо їх на пусті клітинки за допомогою регулярних виразів.

Пояснимо регулярний вираз. Його основу складає дана частина: "[+-]?([0-9]\*[.])?[0-9]+". Вона перевіряє рядок на те, що це або int, або float. Далі треба застосувати інверсію умови, тобто: (.(?!(умова))).

У кінці приведемо усі стовпчики до типу float64.

Рисунок 3.8 - Заміна невідповідних клітинок у стовпчиках на -1, приведення типів до float64

Заповнимо пусті значення на середнє значення для кожного стовпчика відповідно. Застосуємо метод fillna.

Рисунок 3.9 - Заміна пустих значень на середнє для кожного стовпчика

Округлимо до сотих значення. Застосуємо метод round.

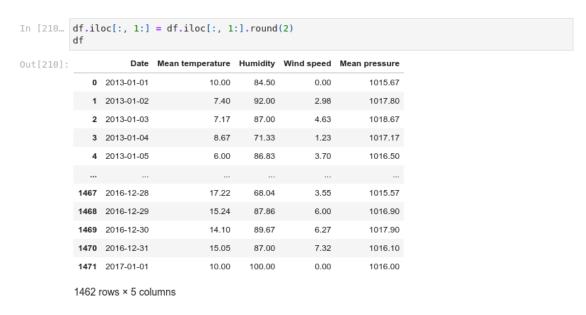


Рисунок 3.10 - Округлення чисел у датафреймі

Зробимо колонку дат індексом за допомогою методу set\_index.

In [211	<pre>df.set_index('Date', inplace=True) df</pre>							
Out[211]:		Mean temperature	Humidity	Wind speed	Mean pressure			
	Date							
	2013-01-01	10.00	84.50	0.00	1015.67			
	2013-01-02	7.40	92.00	2.98	1017.80			
	2013-01-03	7.17	87.00	4.63	1018.67			
	2013-01-04	8.67	71.33	1.23	1017.17			
	2013-01-05	6.00	86.83	3.70	1016.50			
	2016-12-28	17.22	68.04	3.55	1015.57			
	2016-12-29	15.24	87.86	6.00	1016.90			
	2016-12-30	14.10	89.67	6.27	1017.90			
	2016-12-31	15.05	87.00	7.32	1016.10			
	2017-01-01	10.00	100.00	0.00	1016.00			
	1462 rows >	4 columns						

Рисунок 3.11 - Приведення колонки дат до індексу

Експортуємо датафрейм до .xlsx-файлу за допомогою методу to\_excel.

In [214	<pre>name = 'result.xlsx' df.to_excel(name) x1 = pd.ExcelFile(name) x1.parse('Sheet1')</pre>						
Out[214]:		Date	Mean temperature	Humidity	Wind speed	Mean pressure	
	0	2013-01-01	10.00	84.50	0.00	1015.67	
	1	2013-01-02	7.40	92.00	2.98	1017.80	
	2	2013-01-03	7.17	87.00	4.63	1018.67	
	3	2013-01-04	8.67	71.33	1.23	1017.17	
	4	2013-01-05	6.00	86.83	3.70	1016.50	
	1457	2016-12-28	17.22	68.04	3.55	1015.57	
	1458	2016-12-29	15.24	87.86	6.00	1016.90	
	1459	2016-12-30	14.10	89.67	6.27	1017.90	
	1460	2016-12-31	15.05	87.00	7.32	1016.10	
	1461	2017-01-01	10.00	100.00	0.00	1016.00	
	1462 :	rows × 5 col	umns				

Рисунок 3.12 - Збереження датафрейму до .xlsx-файлу

### 4 ВИСНОВОК

Під час виконання даної лабораторної роботи я ознайомитись з операціями попередньої обробки даних Pandas.

У завданнях було прибрано непотрібний індексний стовпчик, переназвано стовпчики, прибрано дублікати зі стовпчика дат, прибрано неправильні клітинки за допомогою регулярних виразів, заповнено пусті значення середніми, приведено числові стовпчики до float64.