Powtarzalny workflow analizy danych w Jupyter Notebook

Moduł 1 kursu Python LLM i analiza danych w notatnikach Jupyter

Materiał do samodzielnej nauki offline

Autor modułu mgr inż. Marcin Godziemba Maliszewski

ENHANCER+ Politechnika Warszawska

Jak korzystać z tego materiału

Ten moduł buduje warsztat pracy potrzebny przed rozpoczęciem szczegółowej analizy danych. Prowadzi od problemu praktycznego do notebooka, który można ponownie uruchomić, przeczytać i sprawdzić. Nie chodzi o zapamiętanie pojedynczych poleceń. Chodzi o uporządkowanie sposobu pracy tak, aby każda liczba, wykres i decyzja miały jasne pochodzenie.

Czytaj rozdziały kolejno. W trakcie pracy równolegle twórz katalog projektu oraz notebook M1_szkielet_analizy.ipynb. Kod w przykładach ma pokazać logikę workflow. Nie jest jeszcze pełnym kursem Pandas ani projektowania wykresów; te zagadnienia zostaną rozwinięte w kolejnych modułach.

Po ukończeniu modułu

- zamienisz problem praktyczny na pytanie, na które dane mogą odpowiedzieć;
- zaprojektujesz notebook jako dokument łączący opis, kod, wynik i interpretację;
- uporządkujesz projekt tak, aby dane, notebooki, wyniki i dokumentacja miały stałe miejsca;
- sprawdzisz notebook po restarcie kernela i pełnym wykonaniu od góry do dołu;
- wykonasz mini analizę od pliku CSV do obserwacji oraz ograniczenia;
- użyjesz LLM jako źródła propozycji, którą można sprawdzić i udokumentować.

Spis treści

1. Od problemu do procesu analitycznego
2. Notebook jako dokument obliczeniowy
3. Organizacja projektu i plików
4. Stan sesji i odtwarzalność
5. Pierwszy mini workflow analizy
6. Pierwsze użycie LLM w notebooku
7. Podsumowanie modułu i słownik pojęć

Rozdział 1 Od problemu do procesu analitycznego

Analiza danych zwykle zaczyna się od problemu opisanego językiem praktyki. Stwierdzenie „awarie zdarzają się za często” może być ważne, ale nie mówi jeszcze, co należy policzyć, z jakich danych skorzystać ani jak rozpoznać odpowiedź. Pierwszym zadaniem analityka jest przełożyć taki problem na pytanie, które można rozstrzygnąć za pomocą określonych danych i miar.

Od problemu do pytania

Dobre pytanie analityczne wskazuje mierzoną wielkość, grupę porównania i zakres danych. Dzięki temu ogranicza liczbę możliwych, przypadkowych działań. Pozwala też sprawdzić, czy wynik odpowiada na pytanie, a nie tylko czy kod działa bez błędu.

Sformułowanie zbyt ogólne	Pytanie gotowe do analizy
Czy z naszymi urządzeniami jest coś nie tak?	Które pięć urządzeń miało najwięcej zdarzeń awaryjnych w 12 miesiącach objętych zbiorem?
Chcemy ograniczyć zużycie energii.	Które typy urządzeń miały najwyższe średnie zużycie energii w analizowanym okresie?
Sprawdź parametry pracy urządzeń.	Czy wyższej temperaturze pracy towarzyszyły większe wartości drgań?

W przypadku przewodnim Eksploatacja urządzeń analizujemy kartotekę urządzeń, kolejne pomiary ich pracy oraz rejestr zdarzeń eksploatacyjnych. Każde źródło ma inną jednostkę obserwacji. W kartotece jeden wiersz opisuje urządzenie, w pliku pomiarowym jeden pomiar, a w rejestrze awarii jedno zdarzenie. Tę różnicę trzeba ustalić przed łączeniem plików i przed liczeniem rekordów.

Sześć etapów analizy

Etap	Pytanie pomocnicze	Rezultat etapu
Dane wejściowe	Skąd pochodzą dane i co reprezentuje wiersz?	Wczytany zbiór oraz opis jego źródła
Kontrola jakości	Czy są braki, duplikaty i wartości wymagające uwagi?	Lista rozpoznanych problemów
Przygotowanie danych	Jakie typy, formaty i kolumny są potrzebne?	Dane gotowe do zaplanowanej analizy
Analiza	Jakie miary i porównania odpowiadają na	Podsumowania, rankingi lub zależności

Etap	Pytanie pomocnicze	Rezultat etapu
	pytanie?	
Wizualizacja	Jaka forma najczytelniej pokaże wynik?	Wykres odpowiadający na pytanie
Interpretacja i raport	Co widać w wyniku i czego on nie rozstrzyga?	Obserwacja, ograniczenie i dokumentacja

Kolejność etapów porządkuje pracę, ale nie oznacza ruchu tylko w jednym kierunku. Wynik może ujawnić zdublowane rekordy, brakującą informację lub pytanie sformułowane zbyt szeroko. Powrót do wcześniejszego kroku jest normalną częścią analizy, pod warunkiem że zapisujesz powód zmiany i jej wpływ na wynik.

Plan analizy przed kodem

Zanim utworzysz pierwszą komórkę z kodem, zapisz plan. Dla pytania o urządzenia z największą liczbą awarii wybierzesz plik zdarzenia_eksploatacyjne.csv, zdefiniujesz jeden wiersz jako jedno zdarzenie, użyjesz liczby awarii jako miary głównej i sumy czasu przestoju jako miary pomocniczej. Wynikiem może być ranking pięciu urządzeń oraz wykres słupkowy.

Plan powinien zawierać również ograniczenie. Sama liczba awarii nie uwzględnia intensywności wykorzystania urządzeń. Maszyna pracująca nieprzerwanie może mieć więcej zdarzeń niż urządzenie używane sporadycznie. Takie zastrzeżenie nie unieważnia analizy, lecz wyznacza granicę jej interpretacji.

Ćwiczenie Dobór źródła danych

Dla każdego pytania wskaż plik, który jest potrzebny do odpowiedzi.

8. Dla pytania o wiek urządzeń wybierz źródło zawierające datę instalacji.
9. Dla pytania o zmianę temperatury U-07 wybierz źródło z kolejnymi pomiarami w czasie.
10. Dla pytania o największą liczbę awarii wybierz rejestr zdarzeń.
11. Dla pytania o dopuszczalny zakres drgań wybierz dokumentację opisującą pola i normy.

Sprawdź się: Wybór danych ma wynikać z pytania, a nie z samej dostępności pliku.

Najczęstsze błędy w planowaniu

- Pytanie jest zbyt ogólne i nie wskazuje miary, zakresu ani porównania.
- Kod powstaje przed planem, więc łatwo analizować to, co akurat jest proste do policzenia.
- Źródło danych i jednostka obserwacji pozostają nieopisane.
- Obliczenia są mieszane z interpretacją, przez co wniosek wykracza poza wynik.

Rozdział 2 Notebook jako dokument obliczeniowy

Plan określa, co chcesz zrobić. Notebook pokazuje, jak to zrobiłeś i co otrzymałeś. Dobry notebook nie jest zbiorem komórek uruchamianych w zapamiętanej kolejności. Jest dokumentem, który prowadzi czytelnika przez tok analizy i pozwala mu sprawdzić decyzje.

Cztery warstwy analizy

Warstwa	Pytanie	Typowy zapis
Opis	Po co wykonujemy ten krok?	Markdown z celem, założeniem lub zakresem
Obliczenie	Co i jak wykonujemy?	Komórka Code z widocznym kodem
Wynik	Co zwrócił program?	Tabela, liczba, komunikat lub wykres pod kodem
Znaczenie	Co wynika z rezultatu i czego on nie pokazuje?	Markdown z obserwacją i ograniczeniem

Nie każda sekcja musi zawierać wszystkie cztery warstwy, ale ten rytm pomaga przy właściwej części analitycznej. Kod mówi, jak działa program. Markdown objaśnia, dlaczego operacja ma zostać wykonana, jaką decyzję podejmujesz oraz co wolno powiedzieć o wyniku.

Markdown i Code

Komórki Markdown przeznacz na tytuły, cel analizy, pytania, założenia, decyzje, obserwacje i ograniczenia. Komórki Code służą do wczytywania danych, przekształceń, obliczeń, kontroli i wykresów. Krótki komentarz w kodzie może wyjaśniać techniczną instrukcję, ale nie zastępuje opisu decyzji analitycznej.

```
# Eksploatacja urządzeń
## 1. Cel i pytanie
**Cel:** wskazać urządzenia z największą liczbą awarii

## 2. Dane i źródła
- liczba zdarzeń
- suma czasu przestoju
Kolumna `czas_przestoju_min`
```

Nagłówki tworzą mapę dokumentu. W całym notebooku utrzymuj jedną hierarchię, na przykład: Cel i pytanie, Dane i źródła, Kontrola jakości, Przygotowanie danych, Analiza, Wizualizacja, Wnioski i ograniczenia. Osoba otwierająca plik powinna rozumieć strukturę nawet przed przeczytaniem kodu.

Jedna komórka jeden cel

Dobra komórka kodu realizuje jedno logiczne zadanie. Jeżeli nie potrafisz opisać jej funkcji jednym zdaniem, rozważ podział. Przykładowo komórka może wybrać urządzenie o największej liczbie awarii. Nie musi zawierać jednej linii, lecz powinna mieć jedno wyraźne wejście, jedną operację i wynik, który da się sprawdzić.

```
awarie = {"U-03": 4, "U-07": 9, "U-12": 5}
lider = max(awarie, key=awarie.get)
lider, awarie[lider]
```

Wynik U-07 i liczba 9 pozwalają zapisać obserwację, że w analizowanej próbce U-07 miało najwięcej zdarzeń awaryjnych. Nie pozwalają jednak stwierdzić, że urządzenie było źle utrzymywane. Do takiego wniosku potrzebne byłyby dodatkowe dane i dowody.

Szkielet notebooka projektu

12. Cel i pytanie
13. Dane i źródła
14. Importy i konfiguracja
15. Wczytanie danych
16. Kontrola jakości
17. Przygotowanie danych
18. Analiza
19. Wizualizacja
20. Obserwacje i wnioski
21. Ograniczenia
22. Rejestr użycia LLM

W module 1 zakładasz wszystkie sekcje. Część dotycząca jakości danych rozwiniesz w module 2, analizę w Pandas w module 3, a zasady projektowania wykresów w module 4. Struktura zostaje stabilna, chociaż jej zawartość stopniowo rośnie.

Ćwiczenie Szkielet notebooka

Utwórz notebook M1_szkielet_analizy.ipynb i potraktuj go jak dokument, a nie katalog kodu.

23. Dodaj tytuł oraz pytanie analityczne z rozdziału 1.
24. Utwórz jedenaście sekcji drugiego poziomu w podanej kolejności.
25. W sekcjach rozwijanych teraz wpisz oczekiwany rezultat.
26. W pozostałych sekcjach zapisz jednym zdaniem, co pojawi się w kolejnych modułach.

Sprawdź się: Struktura powinna być zrozumiała, zanim pojawi się w niej właściwy kod.

Rozdział 3 Organizacja projektu i plików

Notebook jest centralnym dokumentem analizy, ale sam nie wystarczy do jej powtórzenia. Potrzebujesz danych wejściowych, miejsca na dane przetworzone, wyników zapisanych do plików i krótkiej dokumentacji. Wszystkie te elementy powinny znajdować się w jednym katalogu głównym projektu.

Struktura projektu

```
eksploatacja_urzadzen/
├── data/
│   ├── raw/
│   └── processed/
├── notebooks/
│   └── M1_szkielet_analizy.ipynb
├── output/
│   ├── figures/
│   └── tables/
├── docs/
│   └── slownik_danych.md
└── README.md
```

Miejsce	Przeznaczenie	Przykład
data/raw	Niezmienione dane źródłowe	pomiary_eksploatacyjne.csv
data/processed	Dane utworzone przez kod	pomiary_clean.csv
notebooks	Notebooki analityczne	M1_szkielet_analizy.ipynb
output/figures	Wykresy przeznaczone do raportu	awarie_top5.png
output/tables	Tabele i zestawienia	ranking_awarii.csv
docs	Opis danych i decyzji	slownik_danych.md

Dane źródłowe jako punkt odniesienia

Katalog data/raw traktuj jak archiwum. Kod może z niego czytać, ale nie powinien nadpisywać oryginalnych plików. Nie poprawiaj ręcznie wartości w arkuszu, nie usuwaj podejrzanych rekordów bez kryterium i nie zapisuj wyników obok źródła. Przekształcenia wykonuj kodem, a przygotowaną wersję danych zapisuj oddzielnie w data/processed.

Czytelna nazwa pliku powinna mówić o jego zawartości lub roli. Nazwy dane_final.csv, dane_final2.csv i ostateczny.csv nie tworzą dokumentacji. Pomiary_eksploatacyjne.csv, pomiary_clean.csv i ranking_awarii.csv mówią, co zawiera plik i na jakim etapie pracy powstał.

Ścieżki względem projektu

Ścieżka bezwzględna zawiera dysk, konto użytkownika i układ katalogów jednego komputera. Po przeniesieniu projektu do innego systemu zwykle przestaje działać. Zamiast niej ustal katalog projektu raz, a wszystkie ścieżki buduj względem niego za pomocą pathlib.

```
from pathlib import Path

# Założenie: bieżący katalog roboczy to notebooks/
PROJECT_DIR = Path.cwd().parent
RAW_DIR = PROJECT_DIR / "data" / "raw"
DATA_FILE = RAW_DIR / "pomiary_eksploatacyjne.csv"

print("Plik istnieje:", DATA_FILE.exists())
```

Jeżeli plik nie zostanie znaleziony, nie zmieniaj kodu na oślep. Najpierw wyświetl bieżący katalog roboczy za pomocą Path.cwd(), potem zbudowaną ścieżkę i dopiero na końcu wynik DATA_FILE.exists(). To pozwala odróżnić błąd w założeniu o katalogu od literówki w nazwie pliku.

Minimum dokumentacji

- README.md opisuje cel projektu, strukturę katalogów, sposób uruchomienia, wymagane pliki i wyniki.
- docs/slownik_danych.md wyjaśnia znaczenie pól, jednostki, typy danych, dopuszczalne wartości oraz znane ograniczenia.
- Wykresy i tabele są zapisywane przez kod do przewidywalnych katalogów wynikowych.

Ćwiczenie Struktura projektu

Przygotuj katalog projektu Eksploatacja urządzeń.

27. Utwórz katalog główny oraz data/raw, data/processed, notebooks, output/figures, output/tables i docs.
28. Umieść otrzymane pliki wejściowe w data/raw bez zmiany ich zawartości.
29. Zapisz notebook w katalogu notebooks.
30. Dodaj README.md i słownik danych, a w notebooku zdefiniuj ścieżki z pathlib.
31. Sprawdź metodą exists(), czy plik wejściowy jest dostępny.

Sprawdź się: Gotowy projekt można spakować i przenieść jako jeden katalog.

Rozdział 4 Stan sesji i odtwarzalność

Dobrze uporządkowane pliki nie gwarantują jeszcze odtwarzalności. Notebook Jupyter działa wraz z kernelem, czyli aktywną sesją Pythona. Kernel przechowuje zmienne, importy, funkcje i inne obiekty. Notebook może wyglądać poprawnie, a mimo to działać wyłącznie dzięki pamięci pozostawionej przez wcześniejsze eksperymenty.

Dokument i sesja to dwie różne rzeczy

Plik notebooka	Pamięć kernela
Kod, Markdown, zapisane wyniki i numery wykonania	Zmienne, importy, funkcje, obiekty i stan generatora liczb losowych
Pozostaje po restarcie kernela	Jest usuwana po restarcie kernela
Może pokazywać nieaktualny wynik	Może zawierać obiekt utworzony przez niewidoczny już kod

Usunięcie komórki z notebooka nie usuwa zmiennej, którą utworzyła wcześniej wykonana wersja tej komórki. Dlatego notebook może działać dzięki informacjom, których nie ma w jego aktualnej treści. Zrestartowanie kernela usuwa tę różnicę: pozostaje tylko to, co zostanie utworzone przez widoczny kod.

Sygnały ostrzegawcze

- Numery In nie rosną od góry do dołu. Pokazują historię wykonania, a nie kolejność komórek w dokumencie.
- Komórka używa zmiennej, która jest definiowana niżej. W bieżącej sesji może działać, po restarcie zgłosi NameError.
- Kod i zapisany pod nim wynik sobie przeczą. Na przykład head(3) nie może aktualnie zwracać pięciu wierszy.
- Wynik losowej operacji zmienia się między uruchomieniami bez jawnie ustalonego ziarna.

Test czystej sesji

32. Zapisz aktualną wersję notebooka.
33. Zrestartuj kernel, aby wyczyścić pamięć sesji.
34. Wykonaj wszystkie komórki od pierwszej do ostatniej.
35. Zatrzymaj się przy pierwszym błędzie i ustal jego przyczynę.
36. Sprawdź najważniejsze wyniki oraz zapisz notebook po poprawnym przebiegu.

W JupyterLab polecenie zwykle nosi nazwę Restart Kernel and Run All Cells. Notebook jest odtwarzalny w przygotowanym środowisku projektu wtedy, gdy po takim przebiegu nie wymaga dodatkowych, ręcznych działań ani wiedzy autora o kolejności komórek.

Losowość i środowisko

```
SEED = 42
probka = df.sample(n=3, random_state=SEED)

import platform, pandas as pd
print(platform.python_version())
print(pd.__version__)
```

Ustalone ziarno nie poprawia jakości analizy. Pozwala powtórzyć ten sam losowy wybór. Odtwarzalność wymaga też opisu środowiska: wersji Pythona, wybranego kernela i najważniejszych bibliotek. W projekcie zapisz je dodatkowo w requirements.txt.

Ćwiczenie Diagnostyka notebooka

Wybierz notebook, który pozornie działa, i wykonaj test czystej sesji.

37. Przed uruchomieniem obejrzyj numery wykonania i zapisane wyniki.
38. Wykonaj Restart and Run All.
39. Zapisz pierwszy błąd i nazwij jego typ: NameError, FileNotFoundError lub ModuleNotFoundError.
40. Usuń przyczynę błędu przed pierwszym miejscem użycia i powtórz test.
41. Sprawdź, czy istotne operacje losowe mają random_state oraz czy opisano wersje bibliotek.

Sprawdź się: Nie poprawiaj numerów komórek ani widocznych wyników ręcznie. Wygeneruj je ponownie.

Rozdział 5 Pierwszy mini workflow analizy

W tym rozdziale łączysz wszystkie elementy modułu w jeden niewielki, ale kompletny proces. Pytanie brzmi: Jak zmieniała się temperatura urządzenia U-07 w okresie objętym zbiorem? Korzystasz z pliku pomiary_eksploatacyjne.csv oraz trzech pól: czas, urzadzenie i temperatura.

Przebieg mini analizy

42. Wczytaj dane ze ścieżki zbudowanej względem katalogu projektu.
43. Sprawdź rozmiar, nazwy kolumn i ich typy.
44. Wykonaj minimalną kontrolę braków oraz pełnych duplikatów.
45. Wybierz potrzebne rekordy i kolumny, zamień czas na datę oraz uporządkuj dane.
46. Oblicz proste podsumowanie i pokaż wynik na wykresie.
47. Zapisz obserwację oraz ograniczenie, po czym wykonaj Restart and Run All.

Wczytanie i kontrola

```
from pathlib import Path
import pandas as pd
import matplotlib.pyplot as plt

PROJECT_DIR = Path.cwd().parent
plik = PROJECT_DIR / "data" / "raw" / "pomiary_eksploatacyjne.csv"
df = pd.read_csv(plik)

print(df.shape)
print(df.columns)
print(df.dtypes)
```

W zbiorze ćwiczeniowym znajduje się 720 wierszy i 8 kolumn. Metoda shape sprawdza rozmiar, columns pozwala potwierdzić obecność potrzebnych pól, a dtypes pokazuje interpretację ich typów przez Pandas. Kolumna temperatura jest liczbą zmiennoprzecinkową, natomiast czas po wczytaniu ma typ tekstowy, dlatego trzeba go przekształcić.

```
kolumny = ["czas", "urzadzenie", "temperatura"]
print(df[kolumny].isna().sum())
print("Pełne duplikaty:", df.duplicated().sum())
```

W danych ćwiczeniowych wymagane trzy kolumny nie mają braków, a liczba pełnych duplikatów wynosi zero. To kontrola proporcjonalna do pytania, nie pełny audyt jakości. Jeżeli problem wpłynąłby na odpowiedź, zatrzymujesz analizę, diagnozujesz go i dokumentujesz decyzję.

Przygotowanie podzbioru

```
u07 = df.loc[
    df["urzadzenie"] == "U-07",
    ["czas", "urzadzenie", "temperatura"]
].copy()
u07["czas"] = pd.to_datetime(u07["czas"])
u07 = u07.sort_values("czas")
```

```
print(u07["urzadzenie"].unique())
print(len(u07))
print(u07["czas"].is_monotonic_increasing)
print(u07["temperatura"].isna().sum())
```

Metoda copy tworzy niezależną kopię wybranych danych, więc dalsze operacje nie zmieniają ramki wczytanej z pliku. Kontrole pokazują, że podzbiór zawiera tylko U-07, ma 90 rekordów, jest uporządkowany w czasie i nie zawiera braków temperatury. Każda istotna operacja na danych powinna kończyć się podobnym sprawdzeniem skutku.

Podsumowanie i wykres

```
podsumowanie = u07["temperatura"].agg(["count", "min", "mean", "max"])
print(podsumowanie)

fig, ax = plt.subplots()
ax.plot(u07["czas"], u07["temperatura"])
ax.set_xlabel("Czas")
ax.set_ylabel("Temperatura [°C]")
ax.set_title("Temperatura U-07 w czasie")
```

Dla tego podzbioru otrzymujemy 90 pomiarów, minimum 57,45 °C, średnią 61,79 °C i maksimum 66,25 °C. Te liczby opisują zakres, ale nie pokazują przebiegu w czasie. Do pytania o zmianę temperatury potrzebny jest wykres z jasnym tytułem i opisanymi osiami.

Miara	Wynik	Znaczenie
Liczba pomiarów	90	Liczba rekordów urządzenia U-07 w analizowanym okresie
Temperatura minimalna	57,45 °C	Najniższa zaobserwowana wartość
Temperatura średnia	61,79 °C	Średnia ze wszystkich 90 pomiarów
Temperatura maksymalna	66,25 °C	Najwyższa zaobserwowana wartość

Tabela 1. Podsumowanie temperatury urządzenia U-07 na podstawie danych ćwiczeniowych kursu

Wynik obserwacja ograniczenie

Wynik to liczby, tabela i wykres wygenerowane przez kod. Obserwacja może brzmieć następująco: w okresie od 1 stycznia do 14 lutego 2026 temperatura urządzenia U-07 wykazywała ogólną tendencję wzrostową z krótkookresowymi wahaniami. Średnia z pierwszych dziesięciu pomiarów wyniosła około 58.9 °C, a z ostatnich dziesięciu około 64.7 °C, czyli około 5.8 °C więcej.

Ograniczenie musi być równie widoczne jak obserwacja. Wynik dotyczy jednego urządzenia i określonego okresu. Sam wykres nie wskazuje przyczyny zmian, nie potwierdza awarii, nie

pozwala uznać wartości za niedopuszczalne bez reguły domenowej i nie może zostać uogólniony na pozostałe urządzenia.

Ćwiczenie Drugi mini workflow

Powtórz cały proces dla drgań wybranego urządzenia.

48. Sprawdź, jakie identyfikatory urządzeń rzeczywiście występują w zbiorze, i wybierz jeden z nich.
49. Odfiltruj rekordy, uporządkuj je w czasie i sprawdź kompletność kolumny drgania.
50. Oblicz count, min, mean i max, a następnie przygotuj wykres drgań względem czasu.
51. Zapisz jedną obserwację i jedno ograniczenie.
52. Wykonaj Restart and Run All, aby ponownie utworzyć wynik z pliku źródłowego.

Sprawdź się: Wartość nietypową odnotuj. W tym module nie usuwaj jej automatycznie.

Rozdział 6 Pierwsze użycie LLM w notebooku

Model językowy może przyspieszyć pracę, ale nie zastępuje kodu, danych ani oceny analitycznej. W tym module traktujesz odpowiedź LLM jako propozycję. Sprawdzasz ją w kodzie, wynikach, dokumentacji lub przez ocenę logiczną. Dopiero po tej kontroli decydujesz, czy przyjąć sugestię, zmodyfikować ją czy odrzucić.

Trzy bezpieczne role na początek

Rola	Co może zrobić model	Czego model nie rozstrzyga
Planowanie	Zaproponować kolejność kroków i kontrolę każdego z nich	Czy kroki są właściwe dla danych bez sprawdzenia
Wyjaśnianie	Opisać działanie istniejącego fragmentu kodu	Czy opis jest zgodny z kodem bez weryfikacji
Redagowanie	Przygotować roboczą obserwację na podstawie sprawdzonych wyników	Przyczyn zjawiska, awarii lub przekroczenia norm

Kontrakt dobrego promptu

- Cel: czego dokładnie potrzebujesz od modelu.
- Kontekst: jakie dane, kod lub wyniki są potrzebne do zadania.
- Ograniczenia: czego model nie może zakładać ani dopowiadać.
- Format: jak ma wyglądać odpowiedź, aby łatwo było ją sprawdzić.

```
Pracuję w Jupyter Notebook. Pytanie analityczne:
Jak zmieniała się temperatura urządzenia U-07 w okresie objętym zbiorem?
Dostępne kolumny: czas, urzadzenie, temperatura.
Zaproponuj maksymalnie sześć kroków analizy. Nie twórz wartości liczbowych
ani wniosków. Przy każdym kroku podaj sposób sprawdzenia wyniku.
```

Taki prompt prosi o plan i mechanizm kontroli, a nie o gotową analizę. Zapisuj jego treść w notebooku lub rejestrze. Dzięki temu później wiesz, jakiego zadania dotyczyła sugestia i czy odpowiedź pozostała w wyznaczonym zakresie.

Ocena sugestii krok po kroku

Sugestia	Sprawdzenie	Decyzja
Przekształć czas na typ daty	Potwierdź typ kolumny po pd.to_datetime	Przyjmij
Usuń wartości odstające przed analizą	Brak kryterium domenowego w danych i słowniku	Zmodyfikuj: sprawdź, nie usuwaj automatycznie

Sugestia	Sprawdzenie	Decyzja
Posortuj pomiary według czasu	Potrzebne do analizy przebiegu	Przyjmij
Oblicz korelację temperatury z energią	Nie odpowiada na pytanie i sięga poza kontekst	Odrzuć

Nie oceniaj całej odpowiedzi jako dobrej lub złej. Jeden fragment może być poprawny, drugi wymagać modyfikacji, a trzeci być nieuzasadniony. Ten sam standard stosuj przy wyjaśnieniu kodu. Na przykład copy tworzy kopię danych, ale nie usuwa braków, a sort_values porządkuje istniejące rekordy, lecz nie gwarantuje ciągłości pomiarów.

Granica tekstu generowanego przez model

Model może napisać płynne zdanie, które nie ma podstaw w przekazanych danych. Jeżeli otrzyma tylko podsumowanie temperatury, nie może stwierdzić systematycznego wzrostu, przekroczenia bezpiecznego poziomu ani zbliżającej się awarii. Bez wykresu nie widzi przebiegu w czasie, bez progu nie zna normy, a bez dodatkowej diagnostyki nie zna przyczyny.

Bezpieczna wersja komentarza pozostaje przy zweryfikowanych danych: średnia z ostatnich dziesięciu pomiarów była wyższa niż średnia z pierwszych dziesięciu, a wartości mieściły się w obserwowanym zakresie. Następnie jasno wskazuje ograniczenie: podsumowanie nie pozwala określić przyczyny zmian ani stwierdzić awarii.

Rejestr użycia LLM

Pole	Co zapisujesz
Identyfikator i data	Na przykład LLM-001 oraz data użycia
Cel użycia	Plan mini workflow lub wyjaśnienie konkretnego kodu
Model lub narzędzie	Faktycznie użyty model i interfejs
Prompt	Skrót oraz pełną treść w komórce Markdown
Sugestia	Fragment odpowiedzi, który wpłynął na pracę
Weryfikacja	Kod, wynik, dokumentacja albo kontrola logiczna
Decyzja	Przyjęto, zmodyfikowano lub odrzucono
Uzasadnienie	Dlaczego podjęto taką decyzję

Minimalny kontekst i ochrona danych

Zwykle wystarczą nazwy i znaczenie kolumn, krótki fragment kodu, komunikat błędu, kilka przykładowych rekordów albo zweryfikowane podsumowanie. Nie przekazuj automatycznie całego zbioru. Danych osobowych, identyfikatorów osób, haseł, kluczy dostępowych i dokumentów objętych ograniczeniami nie wysyłaj bez sprawdzenia zasad przetwarzania oraz konfiguracji używanego narzędzia.

Ćwiczenie Wyjaśnij sprawdź udokumentuj

Wykonaj jeden pełny, kontrolowany cykl pracy z LLM.

53. Wybierz krótki fragment kodu z mini workflow.
54. Przygotuj prompt z celem, kontekstem, ograniczeniami i formatem odpowiedzi.
55. Poproś model o wyjaśnienie, wybierz co najmniej trzy stwierdzenia i sprawdź je w kodzie, wynikach lub dokumentacji.
56. Oznacz każde stwierdzenie jako przyjęte, zmodyfikowane albo odrzucone.
57. Dodaj wpis do rejestru użycia LLM z nazwą modelu, datą i uzasadnieniem decyzji.

Sprawdź się: Interakcja z modelem jest udokumentowana, ale nie jest ponownie uruchamiana przez notebook podczas testu odtwarzalności.

Podsumowanie modułu

Po tym module masz kompletny fundament dalszej analizy. Umiesz sformułować pytanie, zaprojektować notebook, uporządkować projekt i sprawdzić, czy wynik powstaje ponownie z danych źródłowych. Wykonałeś również pierwszy workflow oraz włączyłeś LLM w sposób, który pozostawia ślad decyzji i weryfikacji.

Triada odtwarzalności

Składnik	Warunek
Kod	Jest widoczny w notebooku i wykonuje się od góry do dołu.
Dane	Pozostają niezmienione ręcznie i są wczytywane ścieżką względem projektu.
Środowisko	Wersje Pythona i najważniejszych bibliotek są udokumentowane.

Dopiero restart kernela i pełne wykonanie notebooka pokazują, czy te trzy elementy współpracują. Pamięć autora, zapisany obraz wykresu i przekonująco brzmiące zdanie modelu nie są dowodem odtwarzalności ani poprawnej interpretacji.

Lista końcowa

☐ Pytanie wskazuje miarę, zakres danych i sposób porównania.
☐ Notebook zawiera cel, dane, kod, wynik, obserwację oraz ograniczenia.
☐ Dane źródłowe pozostają w data/raw i nie są nadpisywane.
☐ Ścieżki są budowane względem katalogu projektu.
☐ Wykres ma tytuł, opisane osie i jednostki.
☐ Każda istotna operacja na danych ma kontrolę skutku.
☐ Restart and Run All generuje wyniki od początku bez ręcznych kroków.
☐ Sugestie LLM mają zapisany prompt, weryfikację, decyzję i uzasadnienie.

Słownik modułu

Pojęcie	Znaczenie
Pytanie analityczne	Pytanie, które wskazuje, jaki wynik może stanowić odpowiedź.
Komórka Markdown	Miejsce na opis celu, założeń, decyzji, obserwacji i ograniczeń.
Stan kernela	Pamięć bieżącej sesji Pythona zawierająca między innymi zmienne i importy.
Odtwarzalność	Możliwość ponownego wykonania analizy z danych, kodu i opisanego

Pojęcie	Znaczenie
	środowiska.
Obserwacja	Stwierdzenie bezpośrednio oparte na wyniku wygenerowanym przez kod.
Ograniczenie	Informacja o tym, czego wynik nie obejmuje lub nie pozwala stwierdzić.
Rejestr użycia LLM	Zapis promptu, sugestii, weryfikacji, decyzji i uzasadnienia.

W module 2 ten warsztat wykorzystasz do dokładniejszej kontroli jakości i przygotowania danych. Zanim cokolwiek poprawisz lub usuniesz, określisz kryterium decyzji, sprawdzisz jej skutek i zachowasz dane źródłowe jako punkt odniesienia.