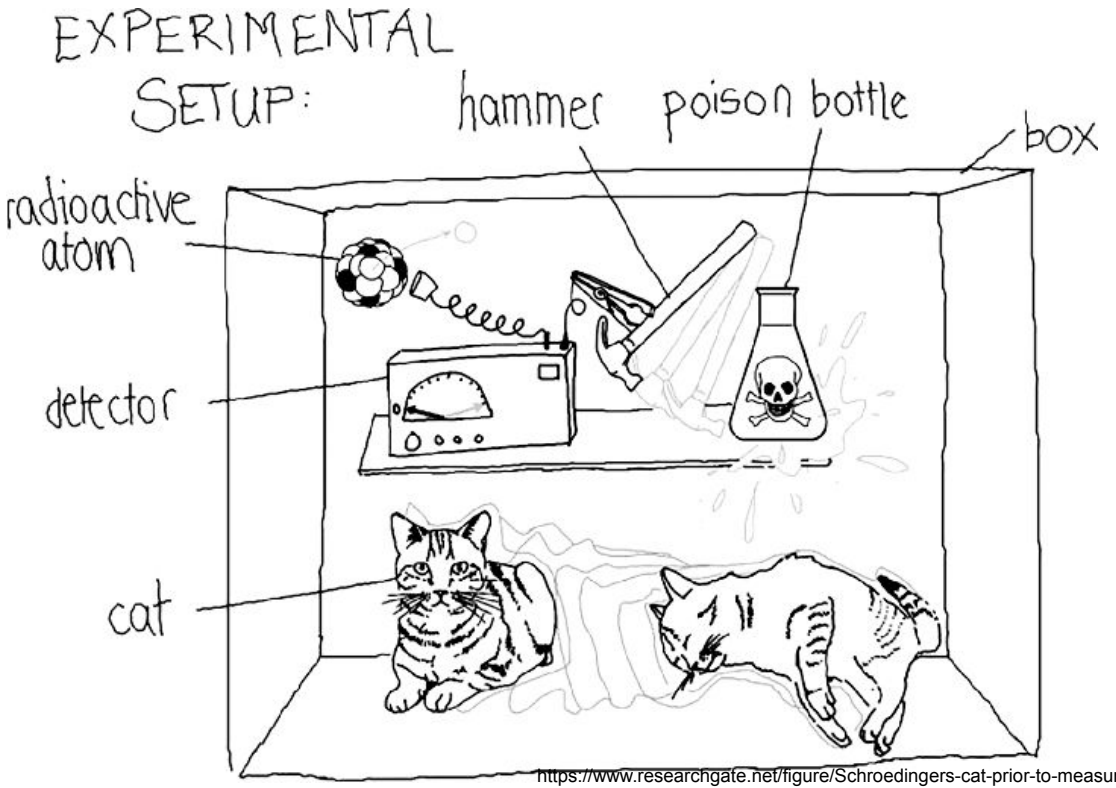
Тестування в ML-проєкті

Pytest та друзі

(Юлія Макогон, Semantrum)

Як ми знаємо, що код працює?



https://www.researchgate.net/figure/Schroedingers-cat-prior-to-measurement-the-cat-is-in-a-superposition-of-the-states-1_fig3_301703361

Як ми знаємо, що код працює?

- без багів

  ... в продукті
  ... під час розробки
  ... коли внесли зміни
  ... через півроку

- з достатньою якістю

  (тестовий корпус + метрика)

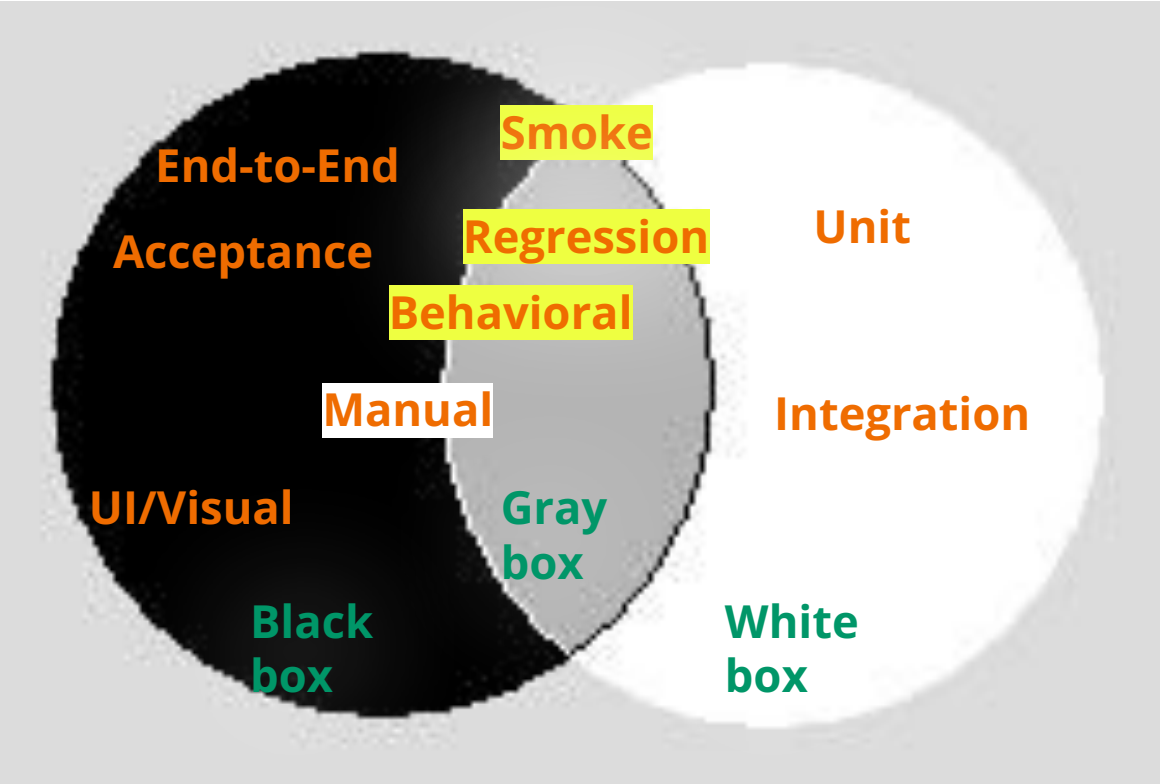
- ефективно

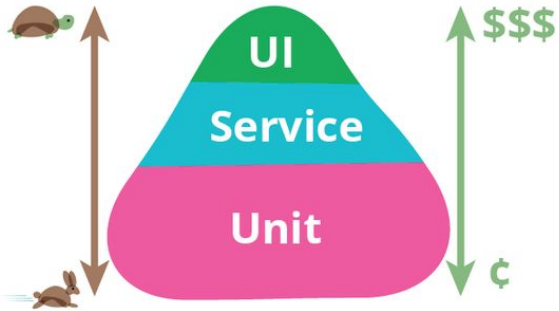
  (заміри часу, ресурсів або профайлер)

Хто тестує?

Хто?	Як?	Коли відомо про проблему?
Клієнт	Продукт	Тижні і місяці від релізу
QA або аналітик	Продукт, тестові утиліти	Тижні (або ніколи)
Розробник	Продукт, тестові утиліти, код	На етапі написання коду

Якими можуть бути тести?





Testing pyramid

Functional vs Non-Functional (Performance, Load etc.)

https://circleci.com/blog/testing-methods-all-developers-should-know

https://martinfowler.com/articles/2021-test-shapes.html

Jupyter notebooks, python scripts, streamlit

--Потрібна людина, щоб оцінити працює чи не працює

notebook*	+для розробки та ML-експериментів +візуальні та текстові елементи +how-to & tutorials	-часто зрозумілий лише розробнику -модель та результати перемішані
main.py	+для роботи з великими пакетами даних +для тестування ефективності +одна команда - один скрипт	-результат у вигляді тексту -параметри у командному рядку або json
streamlit	+візуальний інтерфейс +зручно міняти параметри +можна показати клієнту	-більш корисний, коли API уже готовий

* див. papermill для параметризації

Unit-tests FIRST

[F]ast (бо їх запускають часто)

[I]solated ➜➜➜➜➜➜

[R]epeatable (∀ запуск = ~~random~~)

[S]elf-validating (умови та результат)

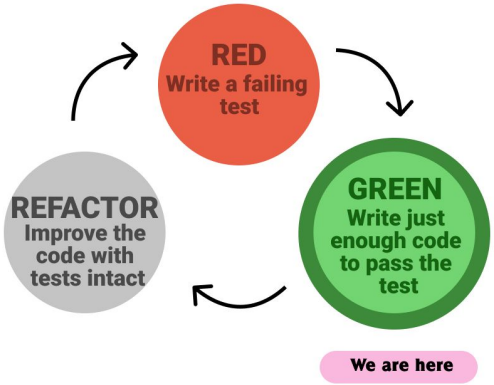
[T]imely (новий код, багфікс, регекс)

https://howtodoinjava.com/best-practices/first-principles-for-good-tests/

TDD - Test Driven Development

Як ми ~~знаємо~~, що код працює?

Як ми дізнаємось, що код працює?



Ґуґлити цих людей

Результати, пов’язані із запитами "Мартін Фаулер" і "Кент Бек"

Think-ahead Driven Development

Їсти слона шматочками (які питання вирішуємо коли пишемо код)

1. Які дані очікуємо на вхід?
2. Які дані (результат) очікуємо на виході?
3. Яку поведінку очікує користувач коду? ➜ “Червоний” тест + stub методу або класу (приклад використання нашого коду без начинки)
4. Які залежності є? ➜ Fake/ mock/ fixture - ізолюємо
5. Які обмеження та варіанти у даних та залежностей? ➜ Тести для типових і corner cases
6. Які дані використовували в експериментах (jupyter)? ➜ Тести
7. Не виходить написати мінімальний працюючий “зелений” код? ➜ Ріжемо на більш зрозумілі шматочки і знову п.1

➜ ➜ ➜ У кожного шматочка коду якомога менша кількість залежностей, один метод = одна дія

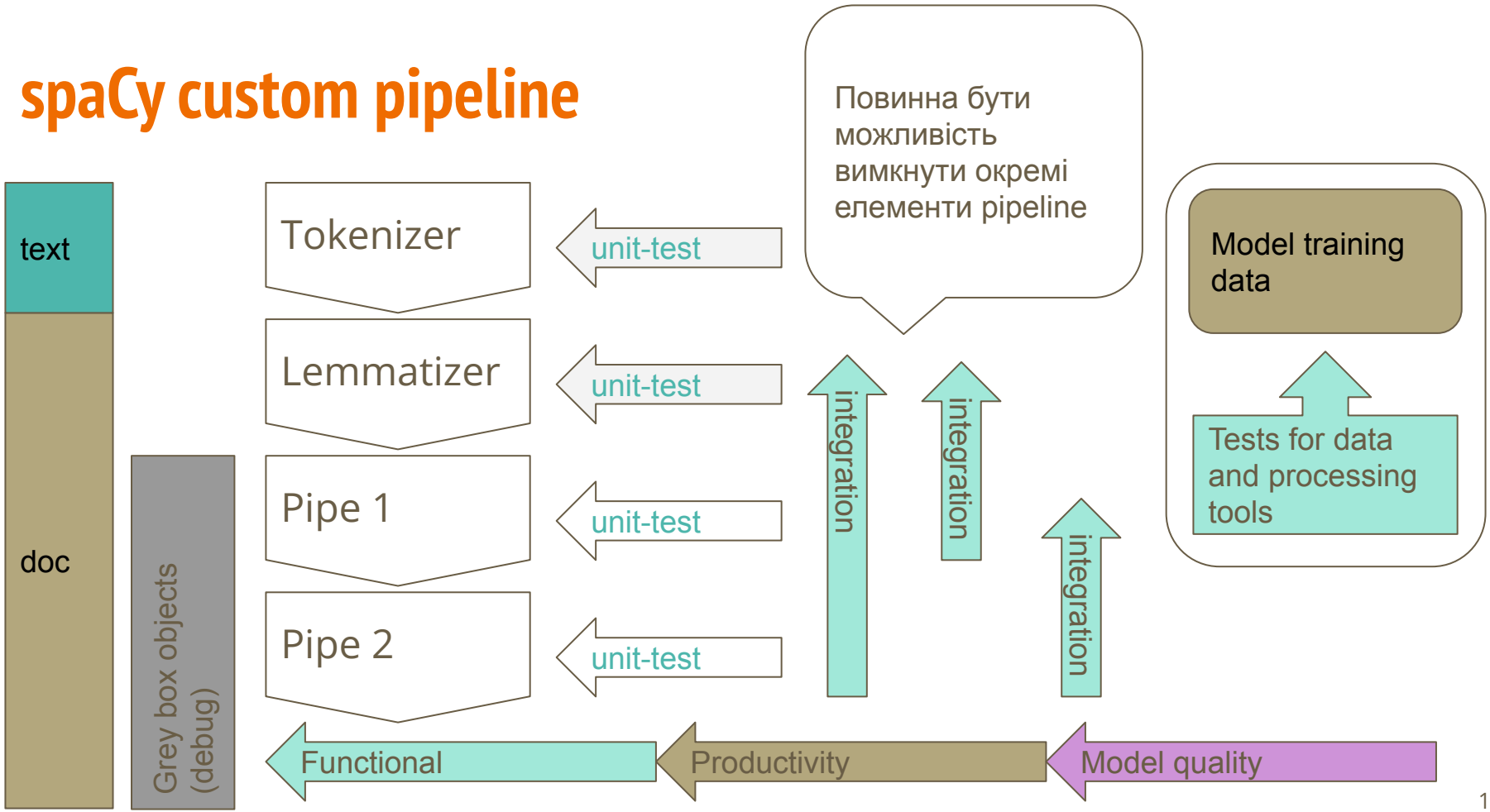
Testing ML Pipelines

- https://www.kdnuggets.com/2019/11/testing-machine-learning-pipelines.html - відносно коротка стаття про особливості тестування ML
- https://www.youtube.com/watch?v=GycRK_K0x2s - відео про pytest для sklearn
- https://www.jeremyjordan.me/testing-ml/amp/ - глибокий огляд, наприкінці всі посилання варті перегляду теж
- https://developers.google.com/machine-learning/testing-debugging/pipeline/overview - як пропонують тестувати моделі в Google

Кілька ідей

- Тестувати:
  - Валідність даних / Код для підготовки даних / Модель
  - Pre-train tests / Post-train tests (див. Jeremy Jordan)
- Не обов’язково 100% покриття тестами, але якщо код використовується більш ніж один раз, то заслуговує на тест
- Один з викликів - недетермінованість даних та результату
  - Використовувати семпли даних (відповідно до різних критеріїв, slicing)
  - Фіксувати random seed
- Pytest must have для тестування бізнес-логіки, логіки обробки даних та інтеграції
- DVC etc. дозволяють відслідковувати, чи якість моделі не погіршилася в результаті експериментів
- Behavioral tests of the model - див. CheckList

spaCy custom pipeline



Приклади з pytest

https://github.com/juliamakogon/pytest_demo