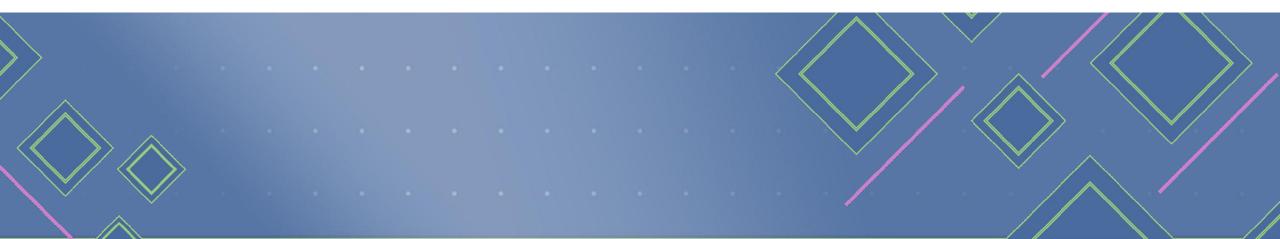


Testes Unitários para ML

Renan Santos Mendes





Testes Automatizados

Testes automatizados

O que são os testes automatizados?

- Garantir que uma alteração não gera efeito colateral;
- Deixar o código mais limpo e melhora a manutenção;
- Serve como documentação: ver quais são os cenários esperados e os tratamentos;
- Evitar trabalho manual (infinitos prints);
- Evitar bug's;
- Prover feedback para quem está desenvolvendo a aplicação.



Testes Unitários

Testes unitários

- Um teste unitário é a maneira de você testar as pequenas unidades existentes em seu código.
- Vão garantir que a aplicação esteja funcionando corretamente após alguma modificação até mesmo na lógica do código;
- Iremos utilizar o framework do Python chamado pytest;
- Existem outras ferramentas para teste: Robot, Testfy e Unittest;



Testes Unitários para ML

Testes unitários para ML

- Eles s\(\tilde{a}\)o projetados para testar partes espec\((\text{ficas do c\tilde{o}digo}\))
 do modelo;
- Os testes ajudam a garantir que o código esteja implementado corretamente e que os resultados produzidos pelo modelo sejam precisos e confiáveis.
- Podem ser usados para validar as saídas do modelo em diferentes cenários;
- Permite ter confiança na qualidade do código e nos resultados produzidos.

Testes unitários para ML

Sugestões de testes:

- Carregamento dos dados;
- Processamento de dados;
- Criação do modelo;
- Treino.





Renan Santos Mendes





O que é uma API?

- API significa Application Programming Interface.
- Define regras e protocolos para a comunicação entre softwares.
- Permite a interação padronizada entre componentes de software.
- Fornece uma interface para acessar recursos e funcionalidades de um software.
- Facilita a integração entre sistemas.
- <u>Usada para obter informações, enviar/receber dados e acessar</u>
 <u>recursos.</u>

Protocolo:

- <u>Contrato</u> entre um **provedor** e um **usuário** de informações, estabelecendo o conteúdo exigido pelo consumidor (a chamada) e o conteúdo exigido pelo produtor (a resposta);
- Não há necessidade de saber como os serviços são implementados.
- Cada serviço pode ser implementado em linguagens
 diferentes e podem se comunicar sem nenhum problema

- Existe um estilo arquitetural de aplicações chamado REST (Representational State Transfer);
- O padrão REST apresenta alguns princípios de estilo:
 - Cliente-servidor: A separação clara entre cliente e servidor
 - Stateless: Não armazena estado no servidor.
 - Interface uniforme: Uso consistente de métodos HTTP e manipulação de recursos por meio de identificadores (URLs).

- Transferência de estado representacional: As respostas do servidor contêm representações do estado atual do recurso solicitado (JSON é uma representação).
- Sistema em camadas: Possibilidade de usar uma arquitetura em camadas para escalabilidade e desempenho.
- Uma API RESTful é aquela que adere estritamente aos princípios e restrições do estilo arquitetural REST.

Principais métodos HTTP:

- GET: Método genérico para qualquer requisição que busca dados do servidor;
- POST: Método genérico para qualquer requisição que envia dados ao servidor;
- PUT: Método específico para atualização de dados no servidor;
- **DELETE**: Método específico para remoção de dados no servidor.

- Em Python existem uma grande diversidade de frameworks:
 - Flask
 - Django
 - Tornado
 - Web2py
 - Bottle
 - FastAPI



Foi lançado em 2018, e suas principais características são:

- Rápido para desenvolver: Aumenta a velocidade 200% a 300%.
- Poucos bugs: Reduz cerca de 40% de erros induzidos por humanos
- Intuitivo: Grande suporte a *IDEs. Auto-Complete* em todos os lugares.
- **Fácil**: Projetado para ser fácil de aprender e usar.
- Robusto: Tenha código pronto para produção e com documentação interativa automática.

Para se criar um API, poucos passos devem ser realizados:

- Instalação do pacote FastAPI e do web server Uvicorn;
- Criação do script:
 - Import da biblioteca;
 - Criação do app;
 - Criação das rotas;
- **Executar** localmente o script.

• Exemplo:

```
from fastapi import FastAPI

app = FastAPI()

♣ Renan Santos

@app.get("/")

async def read_root():

♠ preturn {"Hello": "World"}
```

