

MLops

Advanced Institute for Artificial Intelligence – AI2

<https://advancedinstitute.ai>

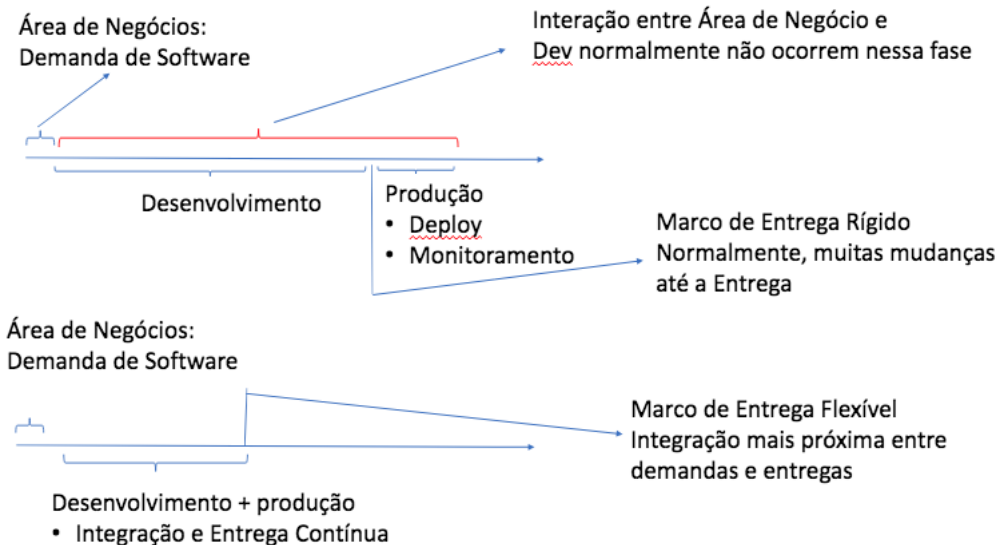
Agenda

- ☐ Integração e Entrega Contínua
- ☐ Fluxo de Trabalho de Aplicações de Aprendizagem de Máquina
- ☐ Introdução ao Modelo de Microserviços
- ☐ Automação de Fluxo de Dados (DATAOps)
- ☐ Automação de Montagem e Avaliação de Aplcações de Aprendizagem de Máquina (MLOps)

Desenvolvimento Contínuo

- Integração contínua (CI): desenvolvedor continuamente integra seu código (normalmente uma ramificação do código principal) com o código principal
 - Controle de versão
 - Build automatizado
 - Check in frequente
 - Suíte de testes abrangente
- Entrega contínua (CD): processo automatizado que permite colocar uma nova versão do código principal em produção e desfazer o processo caso ocorra problemas, de forma simples
 - Colocar em produção automaticamente
 - Automação de uso de containers
 - Integração contínua é um pré-requisito
 - Normalmente um processo semi-automático
- Montagem (Deploy) Contínua: todo código que entra na master automaticamente é colocado em produção

Desenvolvimento Contínuo



Integracao e Entrega Contínua representam uma flexibilizacao de entrega de software em marcos pré-definidos

- ❑ Maior sinergia entre demandantes de software e processo de desenvolvimento de software
- ❑ Desenvolvimento de software pode ser feito de modo mais granular
- ❑ Cada nova funcionalidade pode ser desenvolvida e implantada sem impactar o sistema como um todo
- ❑ Permite melhorar o atendimento de expectativas da área de negócios, que pode ter novas funcionalidades disponíveis em curtos espaços de tempo
- ❑ Processo auto controlado, que permite organizar o trabalho, mantendo a liberdade de cada colaborar de utilizar suas próprias ferramentas

Como alcançar tais níveis de automação?

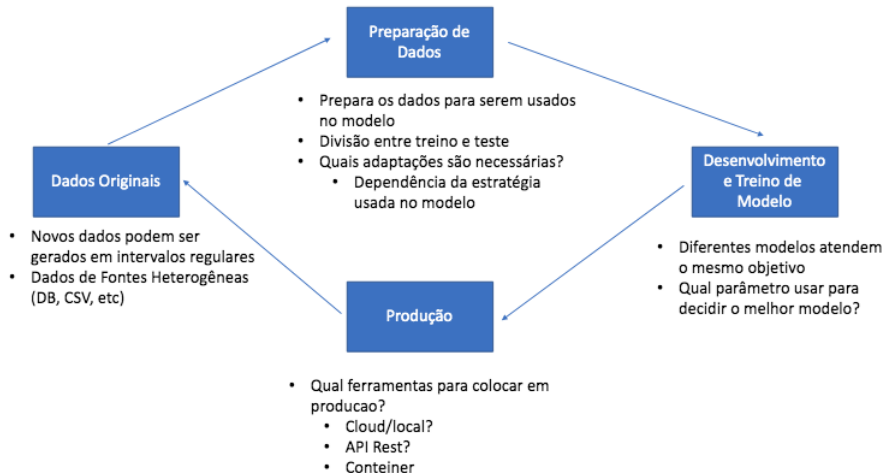
- ❑ Profissional TI dedicado a colocar sistemas diversos em produção
 - Pouco ou nenhum conhecimento da estrutura de tais sistemas
 - Desenvolvedores do sistema possuem pouco ou nenhum conhecimento quanto a montagem de tais sistemas
- ❑ Barreira entre desenvolvimento e produção
 - Dependencias normalmente desconhecidas
 - Complexidade das etapas desconhecidas
- ❑ Integração/Entrega contínua é uma forma de quebra tal barreira
 - Colocar um sistema em producao, ou uma parte dele, é um processo conhecido pelo desenvolvedor e TI
 - Processo claro, planejável e conhecido usando especificações comuns

- ❑ OPs Contração que mostra unificação de áreas com a parte operacional de colocar em produção
- ❑ DEVOPs: prática de software que unifica o desenvolvimento de software (Dev) e a operação de software (Ops)
- ❑ Tecnologias diversas como Git, Jenkins, tecnologias de teste automatizados, usando por equipes compostas por desenvolvedores e TI
- ❑ As mesmas práticas de DEVOps tem sido também se popularizado para desenvolvimento de sistemas de aprendizagem de máquinas:
 - CD4ML
 - DATAOPs
 - MLOPs

Sistemas de aprendizagem de máquina (Machine Learning), são tipos de sistemas que seguem uma estrutura clara

- ☐ Organizacao dos dados de entrada
- ☐ Desenvolvimento de um modelo
- ☐ Avaliacao do modelo
- ☐ Colocar modelo em produção
- ☐ Receber requisições para predição

Desenvolvimento Contínuo



Melhorias contínuas podem ser feitas em aplicações de Aprendizagem de Máquina considerando diferentes aspectos:

- ☐ Organizar os dados usados para treino que são gerados diariamente
- ☐ Adaptar modelos para utilização de dados complementares
- ☐ Comparação entre modelos implementados de acordo com diferentes abordagens
- ☐ Implementar modelos que abordam o problema de outras maneiras
- ☐ Adaptações quanto a forma de integração do modelo com outras aplicações
- ☐ Adequações para melhorar o desempenho do treino e predição

Ferramentas para desenvolvimento de microserviços:

- ☐ Flask biblioteca para Microserviço
- ☐ Json protocolo para troca de dados

- ❑ O framework Flask permite criar servidores web seguindo a especificação WSGI (Web Server Gateway Interface), que é um padrão para o desenvolvimento de aplicações web em Python
- ❑ A idéia desse padrão é permitir a portabilidade de uma aplicação web em python entre diferentes servidores web atuando como um middleware
- ❑ Flask provê recursos para
 - Iniciar um servidor web
 - Capturar as chamadas para diferentes funcionalidades de acordo com as URLs passadas
 - processas as requisições em um script python

Instalando flask

```
1  pip install flask
```

Desenvolvimento Contínuo

Uma aplicação mínima em Flask precisa importar a class Flask, criar uma instância da classe flask e atender ao menos uma requisição web

O Routing define um formato de URL que deve ser capturado e direciona a chamada para uma função escrita em python

Para isso vamos salvar o código a seguir em um arquivo chamado flaskapp.py

```
1  from flask import Flask
2  app = Flask(__name__)
3
4  app.route('/')
5  def hello_world():
6      return 'Hello, World!'
```

Para colocarmos o microserviço em produção é necessário executar dois passos:

Criar a variável de ambiente `FLASK_APP` e atribuir como valor o nome do arquivo criado (`flaskapp`)

```
1 export FLASK_APP=flaskapp
```

Executar o ambiente flask:

```
1 flask run
```

Isso indica que o servidor está no ar no local host e respondendo na porta TCP 5000. Por ser um servidor web, podemos realizar chamadas a ele por meio do navegador

O Código a seguir mostra como implementar uma requisição passando parâmetros

```
1  from flask import Flask
2  app = Flask(__name__)
3
4  app.route('/user/<username>')
5  def profile(username):
6      return username
```


- ❑ O parâmetro `@app.route()` define qual URL será tratada pelo servidor.
- ❑ Ao capturar essa URL o servidor direcionará a chamada a função declarada na próxima linhas após o `@app.route` que é a função `hello_world()`
- ❑ Essa função tem como objetivo retornar uma string com o valor "Hello, World!"
- ❑ Esse retorno é direcionado como resposta ao cliente (Nesse caso um navegador) que realizou a chamada ao servidor web, portanto vai mostrar na tela do navegador a mensagem : "Hello, World!"

- ❑ REST (Representational State Transfer): Um aplicativo Web RESTful expõe informações sobre si na forma de informações sobre seus recursos. Ele também permite que o cliente execute ações nesses recursos, como criar novos recursos (por exemplo, criar um novo usuário) ou alterar os recursos existentes (por exemplo, editar uma postagem).
- ❑ Um aplicativo REST é utilizado por meio de requisições HTTP e provê respostas do tipo HTML, XML ou Json
- ❑ É como requisitar uma URL no navegador e receber uma página HTML como resposta. A diferença é que a solicitação via REST é feita a uma aplicação e não para um arquivo estático
- ❑ Json (JavaScript Object Notation): é um formato bastante flexível para determinar a forma como as aplicações se comunicam.

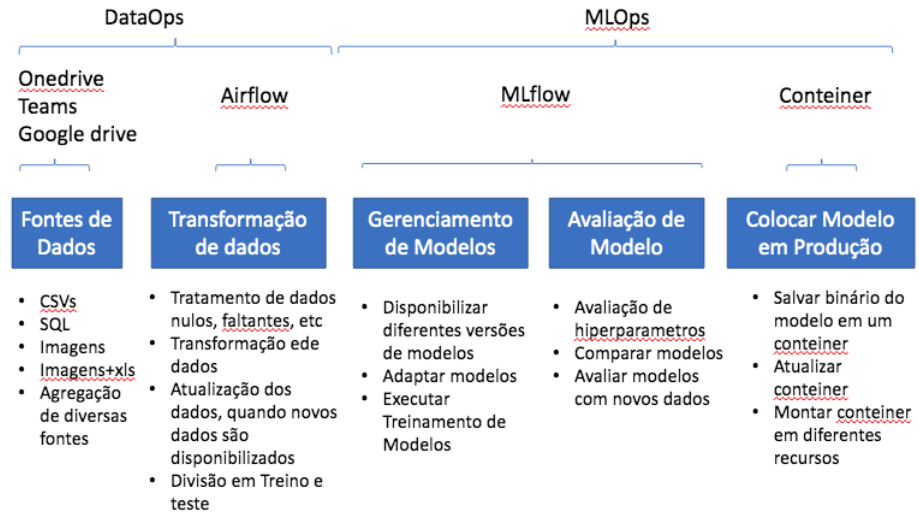
O código abaixo modifica a função de buscar a idade do cliente para retornar um objeto json

```
1
2  from flask import jsonify
3
4  app.route('/busca/<username>')
5  def searchuser(username):
6      idade=0
7      with open('clientes.txt') as f:
8          for line in f:
9              l=line.split(';')
10             if (l[0] == username):
11                 idade=l[1]
12  return jsonify(cliente= username, idade= idade)
```

O Cliente pode realizar o parser das informações enviadas usando a lib json conforme o código abaixo:

```
1  import json
2
3  def buscarCliente():
4      nome = input("Digite o nome do cliente que deseja buscar: ")
5
6      contents = urllib.request.urlopen("http://127.0.0.1:5000/busca/"+nome).
          read()
7
8      print(contents)
9      cliente_idade = json.loads(contents)
10     print(cliente_idade['cliente'])
11     print(cliente_idade['idade'])
```

Desenvolvimento Contínuo



ETL (Extract, Transform and Load) Extração Transformação e Carga é um processo para integração de dados de fontes distintas. A idéia é construir uma base de dados centralizada por meio de três passos:

- ☐ Extração dos dados de diferentes fontes
- ☐ Transformação dos dados para um formato que permita a análise conjunta dos dados
- ☐ Carga dos dados em um repositório com todas as informações em um único local

É comum utilizar ETL para diversos processos:

- ☐ integrar dados de múltiplos sistemas, e obter uma visão unificada de um processo que passa por todos esses sistemas
- ☐ integrar com dados de fontes externas
- ☐ preparar os dados para uma análise específica

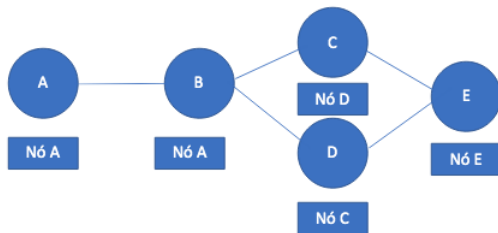
- Algoritmos de Aprendizagem de Máquina normalmente são treinados em um conjunto de dados preparado adequadamente
- É comum que a obtenção, preparação e gerenciamento dos dados seja feito por um profissional especializado nessa atividades (chamado engenheiro de dados), por ser um processo complexo e independente do desenvolvimento do modelo de aprendizagem de máquina
- Ferramentas que automatizam essa etapa são essenciais para permitir uma melhor integração entre o trabalho do engenheiro de dados e o cientista de dados
 - Essa integração é chamada de DataOps

DataOps

- ❑ uma metodologia automatizada com base no processo de desenvolvimento de algoritmos de aprendizagem de máquina
- ❑ Melhora a qualidade dos dados e reduz o tempo de análise de dados
- ❑ Basedo na metodologia ágil e no conceito de entrega contínua
- ❑ Diversas ferramentas permitem implementar esse processo
- ❑ É comum utilizarmos workflows para coordenar a execução de rotinas para preparação de dados

Workflow (DAG - Directed Acyclic Graph)

- Um workflow é um técnica para representar uma aplicação como um processo composto por tarefas e uma ordem de execucao entre das tarefas
- Cada tarefa possui uma entrada e uma saída, permitindo assim controlar o processo de maneira mais ampla
 - Monitorar cada tarefa e recuperar em caso de problemas (não necessariamente a aplicação inteira)
 - Facilitar o uso de diversos recursos computacionais para executar um workflow (Escalonamento)
 - Executar partes de uma mesma aplicação simultaneamente



- Característica do Workflow de Exemplo:
- Diversos nós executam as tarefas do workflow
- Tarefa C e D podem ser executadas simultaneamente
- Dados de Saída das tarefas C e D precisam ser copiados para o recurso que executa a tarefa E
 - Nesse caso, a rede pode trazer impacto na transferência de dados
- Quando a aplicação é definida como um workflow é possível escalonar as tarefas com base em diferentes funções objetivo:
 - Melhorar desempenho, minimizar custo, facilitar reuso com outros workflows etc

Apache Airflow é uma ferramenta aberta de DataOps

- Permitir especificar workflows capazes de extrair informação de fontes de dados e aplicar transformações diversas para gerar um dataset adequado ao cientista de dados.
- As tarefas do workflow podem ser implementadas em diversas linguagens como o Python por exemplo
- Os workflows podem ser agendados para rodar em intervalos regulares
 - Dessa forma, é possível manter um dataset relativo a um processo que ocorre todo dia sempre atualizado
- A colaboração entre o engenheiro de dados e o cientista de dados pode então ser completamente gerenciada pela ferramenta de DataOps Apache Airflow

A seguir a estrutura de um código que implementa um DAG em Airflow (Tarefas implementadas em Python)

```
1  from airflow import DAG
2  from airflow.operators.python_operator import PythonOperator
3
4  def prep_cliente():
5
6      print('task1')
7
8  def prep_cliente_perfil():
9
10     print('task2')
```

Criação do DAG

```
1
2  default_args = {
3      'owner': 'airflow',
4      'depends_on_past': False,
5      'email': ['airflow at example.com'],
6      'email_on_failure': False,
7      'email_on_retry': False,
8      'retries': 1
9  }
10
11 dag = DAG(
12     'prep_sicoob',
13     default_args=default_args,
14     description='DAG de preparacao de dados para Sicoob'
15 )
```

Referencia as tarefas

```
1
2  prep_cliente = PythonOperator(task_id='prep_cliente',
3                                python_callable=prep_cliente, dag=dag,)
4  prep_cliente_perfil = PythonOperator(task_id='prep_cliente_perfil',
5                                       python_callable=prep_cliente_perfil, dag=
6                                       dag,)
```

Sequência da execução das tarefas

```
1  prep_cliente >> prep_cliente_perfil
```

- Ao terminar o desenvolvimento do DAG é necessário armazená-lo e executá-lo no terminal
- O DAG estará disponível para ser executado a partir da interface gráfica do Airflow ou a partir do terminal

O processo de desenvolvimento de modelo de aprendizagem de máquina de modo geral se baseia em um objetivo (normalmente definido em termos de uma predição) e um conjunto de dados

- A definição do objetivo e do conjunto de dados é desafiadora, normalmente é iniciada por um processo exploratório da base
 - A partir daí o objetivo pode ser refinado
 - Interações entre engenheiro de dados e cientista de dados por serem necessárias
- Com o objetivo bem definido é possível atacar o problema usando diferentes abordagens
 - É difícil prever qual abordagem pode ter um melhor resultado sem exploração
 - Métricas de qualidade são fundamentais para comparar os modelos

O Desenvolvimento de um modelo é uma atividade desafiadora e complexa, mesmo considerando que em geral parte de um modelo similar existente

- ❑ Dividir o desenvolvimento em termos de atividades menores não é trivial
- ❑ A concepção do modelo passa por etapas de colaboração em um time multi-disciplinar
- ❑ Partes desse processo são mais sistemáticas e podem ser automatizadas
 - Colocar um dado modelo em produção
 - Comparar desempenho de diferentes modelos

Desenvolvimento Contínuo

The screenshot shows the Google Colab web interface. At the top, the notebook is titled "teste-colab.ipynb" with a star icon. Below the title is a menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". A status bar indicates "Last saved at 12:23 PM". On the right side, there are icons for "Comment", "Share", a settings gear, and a user profile picture.

The main workspace contains a code cell with the text `import pandas as pd`. Above the code is a play button icon. To the right of the code cell is a toolbar with icons for undo, redo, link, comment, settings, delete, and a menu. Below the code cell, on the right, is a comment box from "Silvio Stanzani" at "12:25 PM Today" with the text "como sera feita a leitura dos dados?". A "Resolve" button is next to the comment.

On the left side of the interface, there are icons for a sidebar menu, a code editor, and a file explorer.

CD4ML (Continuous Delivery for Machine Learning) e MLOps

- Nesse cenário a integração e entrega contínua nesse processo é fundamental, pois a comunicação entre Cientistas de Dados e a equipe de operações ou produção é fundamentalmente colaborativa
- Tal colaboração precisa ser automatizada para colocar sistemas de aprendizagem de máquina em produção mais rápido e minimizar os riscos
- CD4ML é mais focada na entrega contínua enquanto que MLOps é mais focado na operacionalização dos modelos

Algumas ferramentas para CD4ML e MLOPs

- ☐ Uber Michelangelo
- ☐ FBLearner
- ☐ Tensorflow TFX

Serving systems

- Ferramentas para colocar sistemas de aprendizagem de máquina em produção são normalmente chamados de serving systems
- A idéia do serving systems é criar uma interface de microserviços que receba uma requisição para uma predição e retorne a predição
- A utilização de microserviços é essencial para permitir maior flexibilidade quanto a desempenho, segurança, montagem e integração do módulo de predição em diversas aplicações


Um exemplo de Serving System em Tensorflow é o `tensorflow_model_server`



- ❑ Modelo em Keras é salvo para o disco
- ❑ Modelo é carregado por um sistema que recebe requisições REST para fazer predições



Outras ferramentas para automatizar o processo de desenvolvimento de aplicações de aprendizagem de máquina também são disponibilizadas pelo tensorflow

O MLFlow é um exemplo de ferramenta que automatiza e facilita o controle do desenvolvimento de aplicações de aprendizagem de máquina

- O MLflow apresenta as seguintes funcionalidades:
 - Permite armazenar métricas de qualidade de diversos modelos relacionados a um mesmo experimento
 - Permite gerar o executável do modelo a partir de diversos frameworks


 **Experiments** **Models** [GitHub](#) [Docs](#)

Experiments  


Training  





Training



Experiment ID : 0 Artifact Location : file:///data/output/mlruns/0

▼ Notes 

None

Search Runs: ? State: **Active**  **Search** **Clear**

Showing 18 matching runs **Compare** **Delete** **Download CSV**     Columns

						Parameters			Metrics >		
<input type="checkbox"/>	Start Time	Run Name	User	Source	Version	alpha	k	l1_ratio	Test scor	Test_scor	acc
<input type="checkbox"/>	✓ 2020-07-08 11:41	-	silvio	 fashior	e232e3	-	-	-	-	-	0.866
<input type="checkbox"/>	✓ 2020-07-08 11:41	-	silvio	 fashior	e232e3	-	-	-	-	-	0.868

Funcionalidades do MLFLOW

- ☐ Tracking: monitorar experimentos de modelos de aprendizagem de máquina
- ☐ Project: reprodução de experimentos
- ☐ Models: automaticamente gerar um modelo que pode ser montado em diversas plataformas

Tracking

- ☐ Parâmetros do modelo
- ☐ Métricas de avaliação de modelos
- ☐ Artefatos diversos relacionados ao modelo
- ☐ Código fonte

Reprodução

- ☐ Local ou Remota
- ☐ Definição de ambiente (conda.yaml)
- ☐ Definição dos parâmetros da execução do modelo (MLProject)

Modelo

- Controla a montagem do modelo em diversas plataformas
 - Azure, Aws, datbricks, etc