A Mentalidade e os Elementos de uma Plataforma de Machine Learning

André Perez @ #TheDevConf 2021

Agenda

1. Apresentação
2. ML do ponto de vista da ciência
3. ML do ponto de vista da engenharia
4. Plataforma de ML
5. Q&A*

André Perez

Mercado

Engenheiro de Dados & ML

Stone Co. | Serasa Experian | Amdocs

Academia

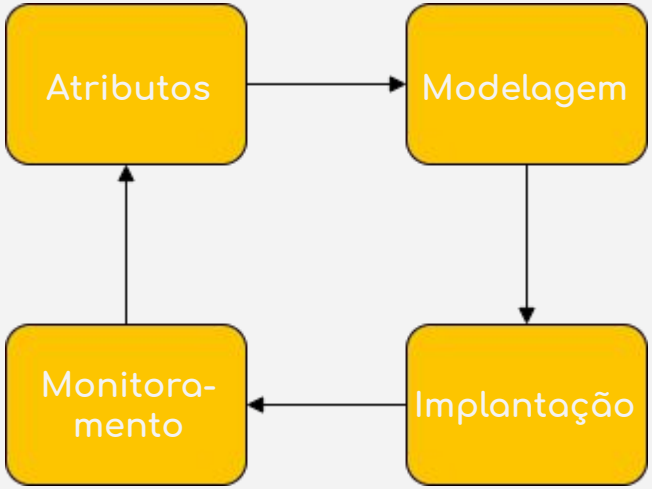
Engenheiro Eletricista @ EESC/USP

Mestrando em Ciência de Dados @ ICMC/USP

Extra

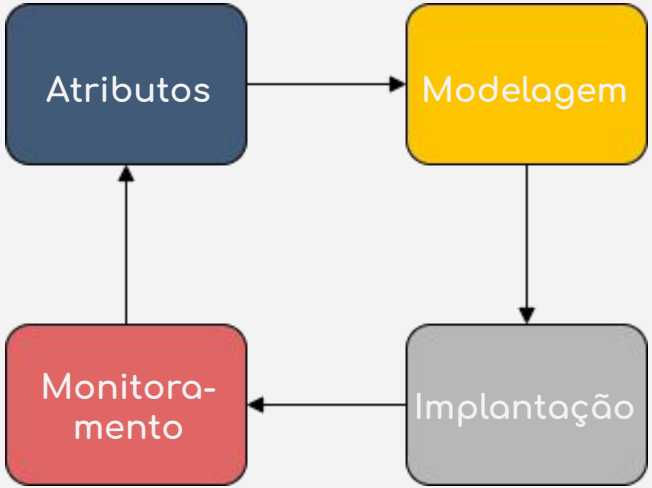
Lives, podcasts, artigos e projetos open source

Ciclo de vida de um modelo de ML



Ciência de Dados

Ciclo de vida de um modelo de ML



Engenharia de Dados

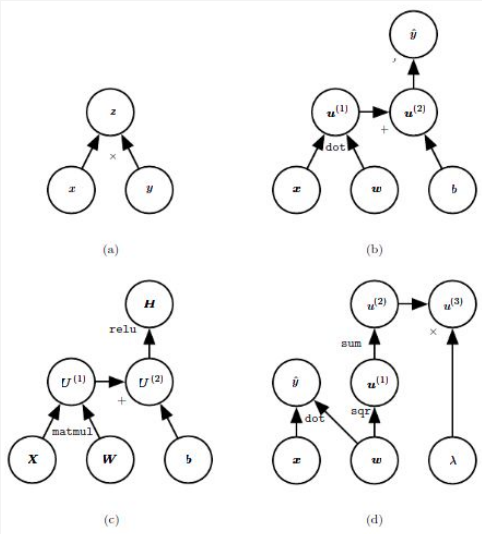
Ciência de Dados

Engenharia de Software

Inteligência de Negócios

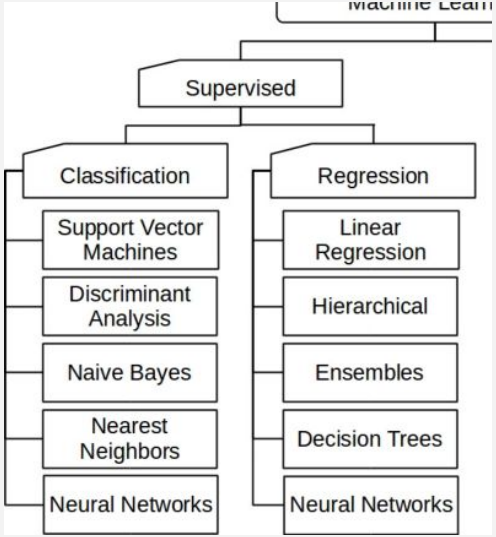
O que vem na sua cabeça quando você pensa em machine learning?

Algoritmos



Fonte

Modelos



Fonte

Aplicações

GPT-3

From Wikipedia, the free encyclopedia

Generative Pre-trained Transformer 3 (**GPT-3**) is an autoregressive language model that uses deep learning to produce human-like text. It is the third-generation language prediction model in the GPT-n series (and the successor to GPT-2) created by OpenAI, a San Francisco-based artificial intelligence research laboratory.[2] GPT-3's full version has a capacity of 175 billion machine learning parameters. GPT-3, which was introduced in May 2020, and was in beta testing as of July 2020,[3] is part of a trend in natural language processing (NLP) systems of pre-trained language representations.[1] Before the

Fonte

Programação Orientada a Objetos

generic_machine_learning_model_short.py

```python
import numpy as np


class MachineLearningModel(object):

    def __init__(self):
        self.params = None  # ml model state
        self.hyper_params = None  # ml model state

    def train(self, target: np.ndarray, features: np.ndarray, hyper_params: np.ndarray) -> None:
        # ...
        # params = f_train(target, features, hyper params)
        # ...
        self.params = params
        self.hyper_params = hyper_params

    def predict(self, features: np.ndarray) -> np.ndarray:
        # ...
        # prediction = f_prediction(features, hyper params, params) = predicted
        # ...
        return prediction
```

Atributos (estado)

Métodos

pickling_short.py

```python
import pickle

from linear_regression_model import LinearRegressionModel

if __name__ == '__main__':

    version = '1-0'
    model = LinearRegressionModel()

     # Fill state

    # target, features, hyper_params = # read data
    model.train(target=target, features=features, hyper_params=hyper_params)
    pickle.dump(model, open(f'ml-model-v{version}.pickle', 'wb'))

    # Consume state

    # unseen_features = # read data
    model = pickle.load(open(f'ml-model-v{version}.pickle', 'rb'))
    prediction = model.predict(features=unseen_features)
```

Cria/atualiza o estado

Consome o estado

Versionamento de modelos

v1.0 - Modelo: regressão linear

v1.1 - Novos dados de treino (estado)

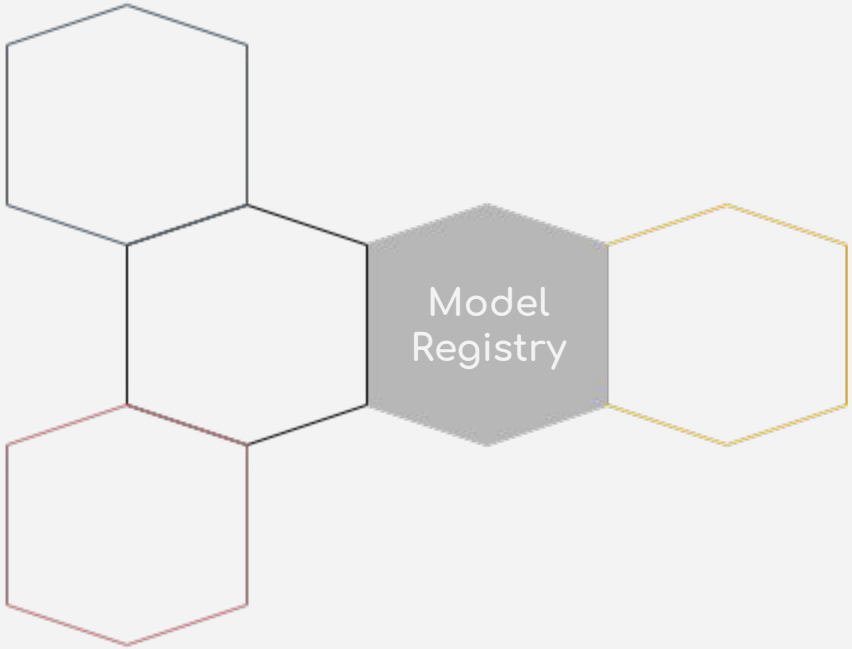
v1.2 - Novos hiper-parâmetros (estado)

v2.0 - Novo modelo: regressão logística (métodos)

v2.1 - Novos dados de treino (estado)

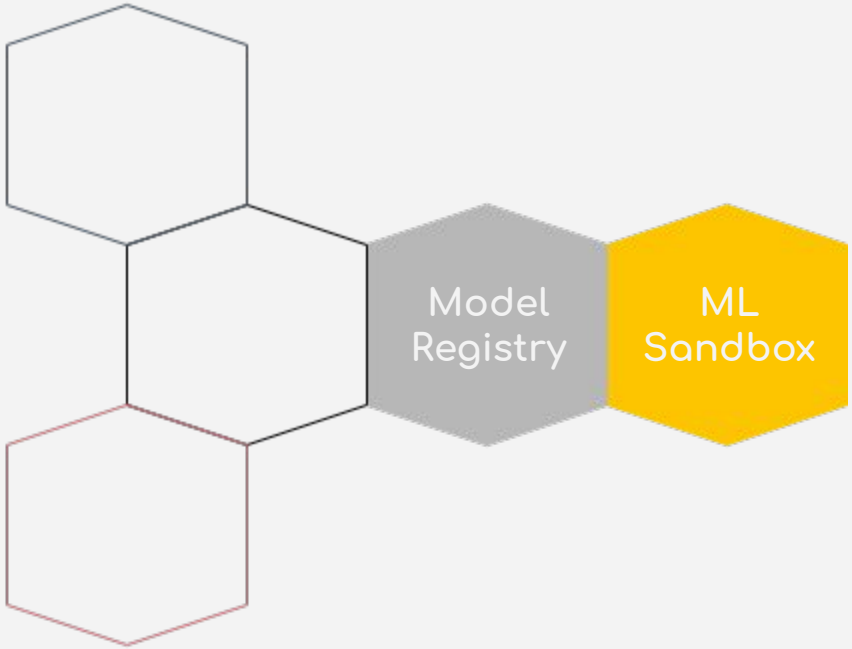
v2.2 - Novos hiper-parâmetros (estado)

Mundo Externo

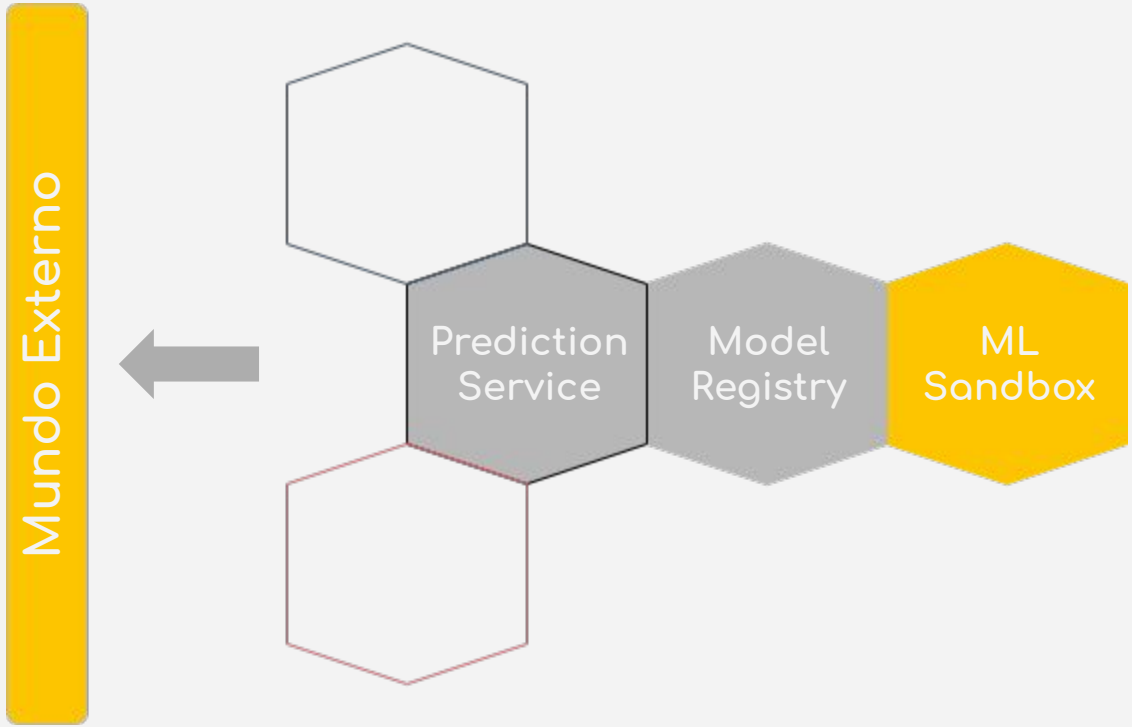


AWS S3 | GCP Cloud Storage | AWS Sagemaker | Kubeflow KFServing | PyPI | AWS ECR

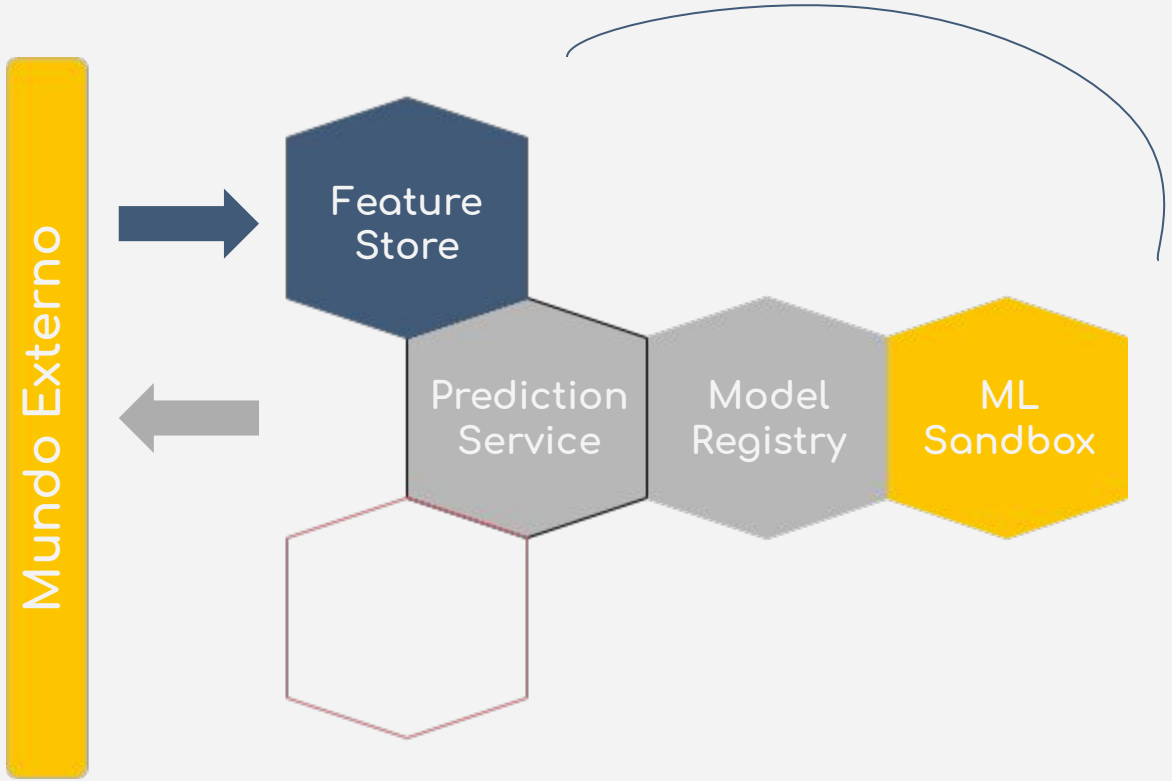
Mundo Externo



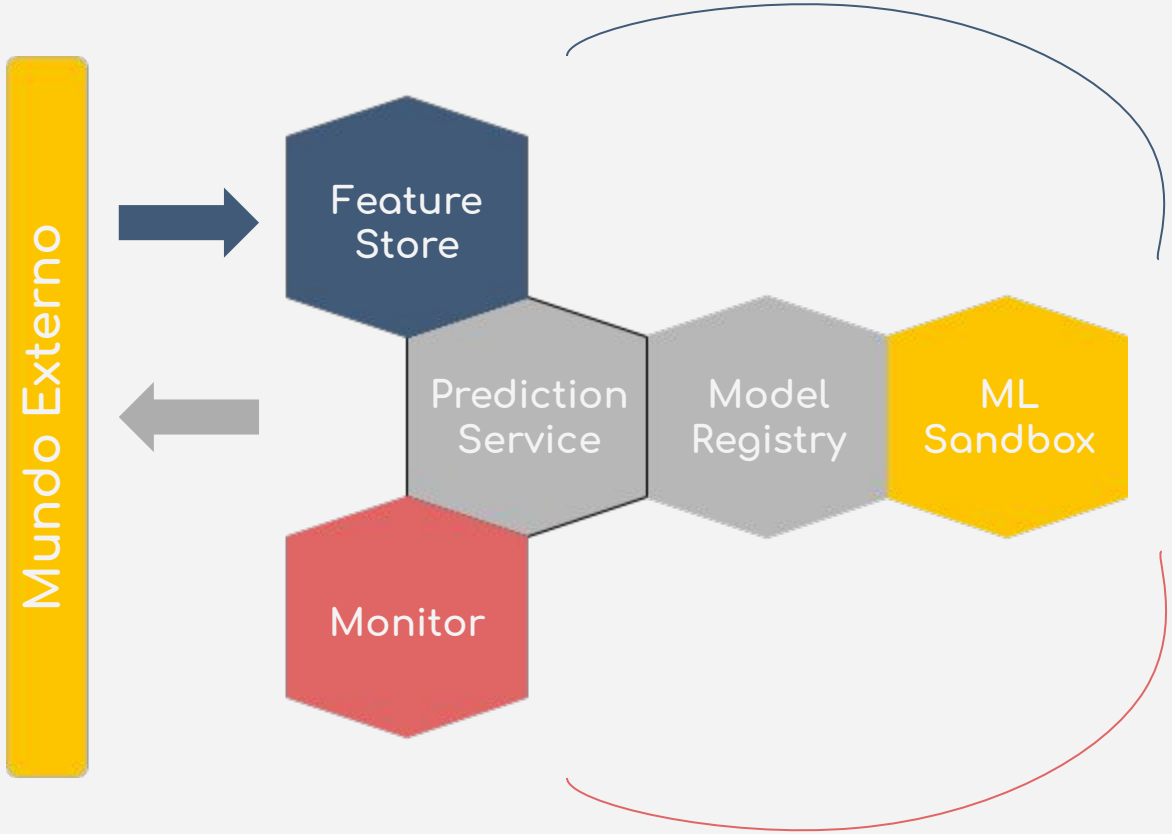
Jupyter Notebooks | AWS Sagemaker | GCP Colab | Databricks Notebooks



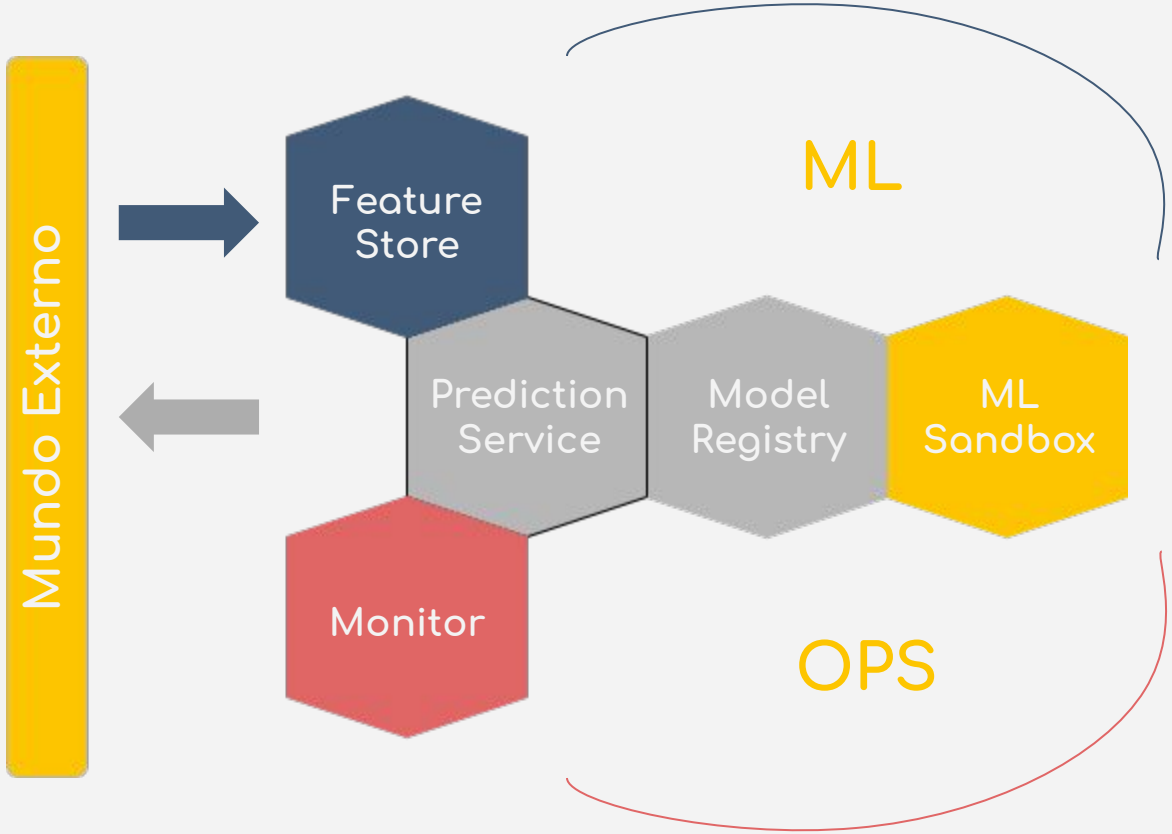
AWS Lambda + API Gateway | AWS Sagemaker | Kubeflow KFServing



Airflow | AWS S3 + AWS Athena + AWS Redshift | AWS Sagemaker | Kubeflow Feast | Tecton



AWS S3 | AWS Redshift | BigQuery | Google Data Studio | Power BI | Tableau


Mundo Externo
Feature Store
Prediction Service
Model Registry
ML Sandbox
Monitor
ML
OPS

Q&A

Engenharia de ML @
Stone Co.