# Projeto Final de Curso – Eng. ML

Herikc Brecher
Repositório:
https://github.com/Herikc2/Engenharia-de-Machine-Learning

# Agenda do Trabalho

O aluno deve preencher essa apresentação com os resultados da sua implementação do modelo. Os códigos devem ser disponibilizados em repositório próprio, público, para inspeção.

Essa apresentação é padronizada para que os alunos possam incluir os seus resultados, com figuras, tabelas e descrições sobre o projeto de curso. Os resultados aqui descritos serão confrontados com os códigos disponibilizados.

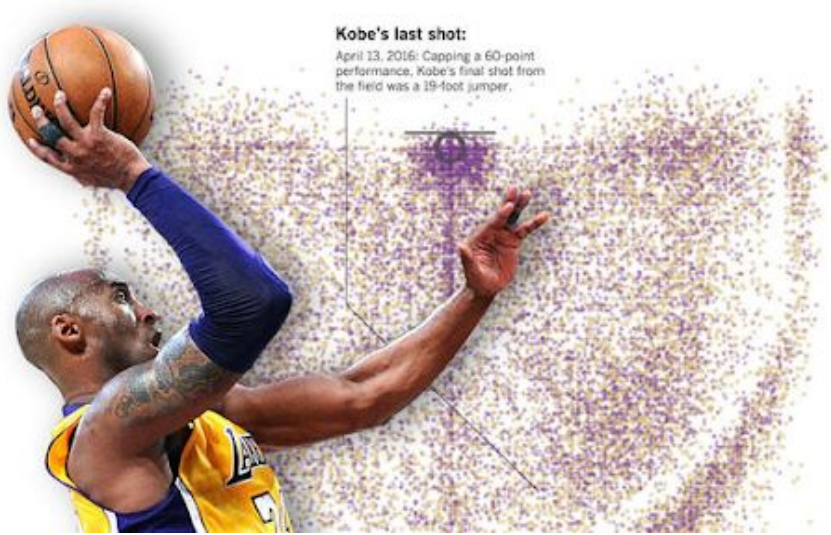
# Roteiro

- Objetivo da modelagem
- Arquitetura da solução
  - Diagrama
  - Bibliotecas
  - Artefatos e Métricas
- Pipeline de processamento dos dados
  - Descrição dos dados
  - Análise Exploratória
  - Seleção base de teste
- Pipeline de Treinamento do Modelo
  - Validação Cruzada
  - Regressão Logística
  - Árvore de Decisão
  - Seleção, finalização e registro
- Aplicação do Modelo
  - Model as a Service localmente
  - Interface para aplicação na base de produção
  - Monitoramento do modelo

# Objetivo da modelagem

Em homenagem ao jogador da NBA Kobe Bryant (falecido em 2020), foram disponibilizados os dados de 20 anos de arremessos, bem sucedidos ou não, e informações correlacionadas.

O objetivo desse estudo é aplicar técnicas de inteligência artificial para prever se um arremesso será convertido em pontos ou não.



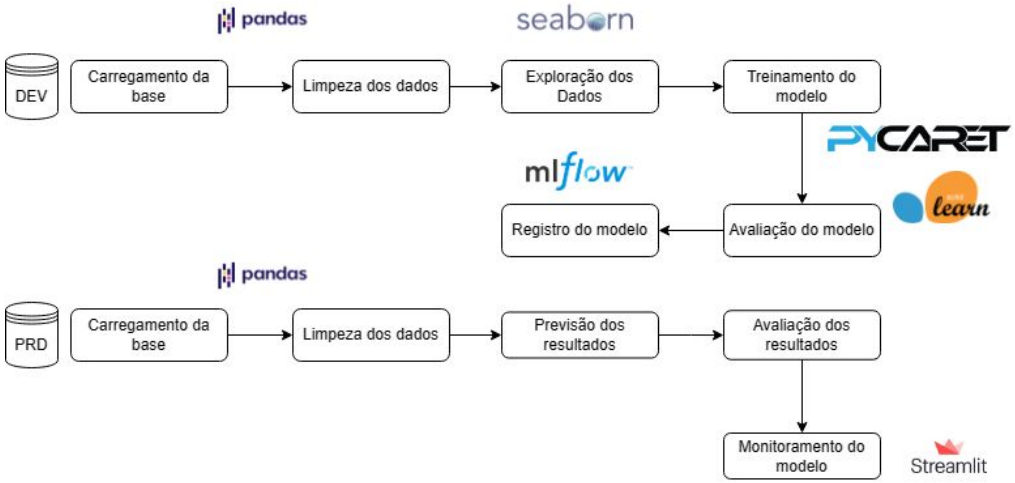
# Arquitetura da Solução

# Arquitetura da Solução

A solução foi construída em duas etapas:

No ambiente de desenvolvimento é carregado a base e realizado uma limpeza dos dados para serem possíveis de se analisar, em conjunto é realizado a análise exploratória para entender quais modelos e dados são de maior valor. Sequencialmente é realizado o treinamento e o registro do modelo.

No ambiente de produção a base é carregada e limpa para servir ao modelo produtivo, esse que será monitorado por meio de uma aplicação.



# Arquitetura da Solução

## Bibliotecas

Scikit-Learn é um conjunto de facilitadores para o treinamento, fornecendo através de uma interface simples métodos complexos de processamento de dados e treinamento de modelos, a sua maior vantagem é a facilidade em comunicar com outras ferramentas como o PyCaret. Além do mais, o Scikit-Learn permite a utilização de métricas para a saúde do modelo, essas que servem para entender quais modelos possuem melhores resultados.

O PyCaret é uma biblioteca para automação de treinamento de modelos de Machine Learning, amplamente utilizado para comparação e tunagem entre os modelos. Também auxilia na escolha do melhor modelo, podendo realizar automaticamente a comparação por métricas de saúde do modelo.

A ferramenta MLFlow é uma plataforma de monitoramento e rastreabilidade do modelo, seu foco é rastrear os artefatos, etapas e versionamento dos modelos de aprendizado de máquina gerados.

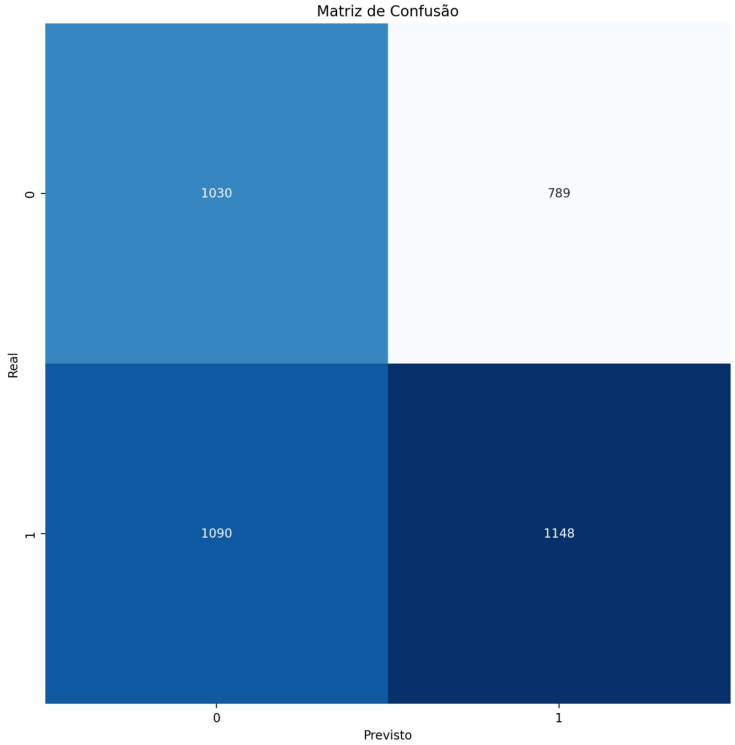
Streamlit é uma ferramenta utilizada na etapa final de visualização dos resultados, contemplando a construção de um portal para melhor monitoramento dos dados, pode ser utilizado para criar dashboards, aplicações e interfaces de comunicação com o usuário.

# Arquitetura da Solução

| | Errou | Acertou |
|---|---|---|
| precision | 0.4858 | 0.5927 |
| recall | 0.5662 | 0.513 |
| f1-score | 0.523 | 0.5499 |
| support | 1,819 | 2,238 |

A primeira métrica de comparação é gerado pelo Scikit-Learn para comparar as diferentes métricas fornecidas, essas que auxiliam a entender a eficácia e a saúde do modelo em ambiente produtivo.

# Arquitetura da Solução




Com objetivo de comparar a previsão do modelo para cada uma das classes alvos, é utilizado a matriz de confusão a fim de entender qual a tendência do modelo.

# Processamento de Dados

# Pipeline de processamento dos dados

**Descrição dos dados**

O dataset apresenta características dos jogos e arremessos realizados por Kobe Bryant durante sua carreira.

Quantidade de linhas:  24271

Linhas com dados faltantes: 3986

Quantidade de linhas restantes:  20285

Colunas: 7

**Descrição das colunas:**

lat (float64): Latitude durante o jogo

lon (float64): Longitude durante o jogo

minutes_remaining (int64): Minutes restantes para acabar o jogo

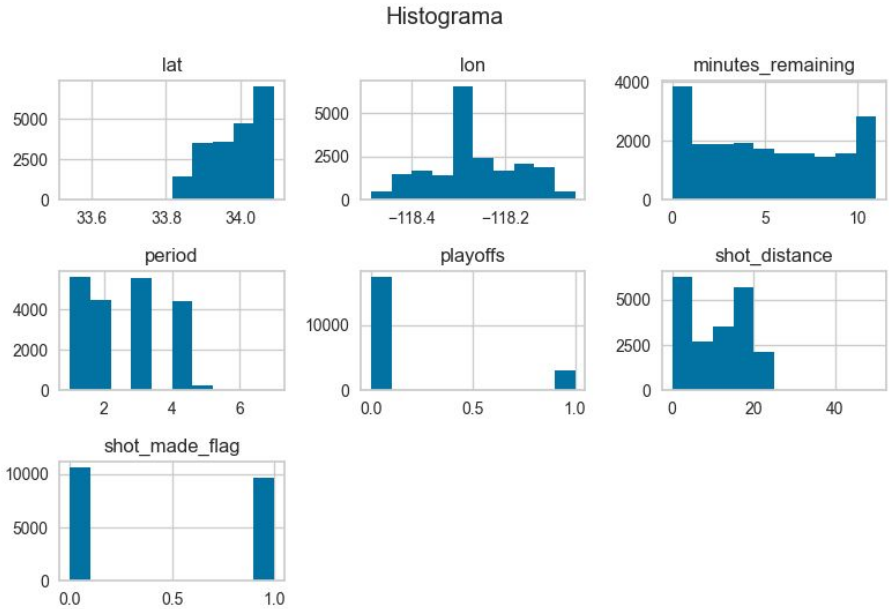
period (int64): Período do jogo

playoffs (int64): Se é um playoff

shot_distance (int64): Distância da cesta

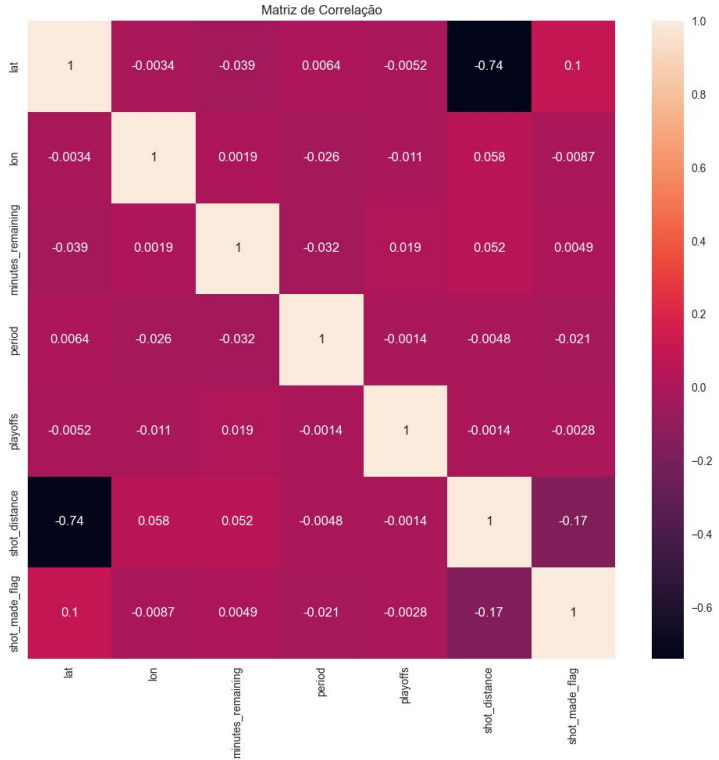
shot_made_flag (float64): Se o arremesso foi realizado ou não

# Pipeline de processamento dos dados




Foi analisado o histograma com objetivo de identificar os padrões de dados existentes na base de dados, com isso foi que os dados não possuem um comportamento gaussiano de forma natural, além disso é perceptível que não existem muitos valores extremos, desclassificando a possibilidade de outliers.

# Pipeline de processamento dos dados




Foi analisado a matriz de correlação com o objetivo de entender se existem dados correlacionados e que compartilham a mesma informação. Analisando o grafico não se percebe o comportamento, além disso não existem dados com uma correlação muito forte além de “shot_distance” com “lat” e “shot_distance” com “shot_made_flag”.

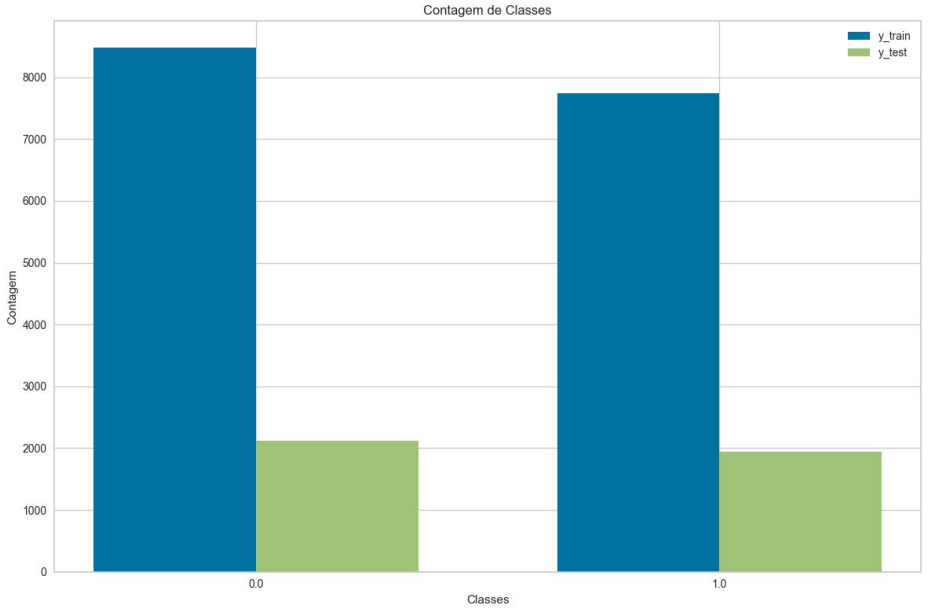
# Pipeline de processamento dos dados

**Seleção base de teste**

Os dados foram separados em conjunto de treino e teste de forma estratificada, destinando 80% dos dados para treinamento e 20% para teste.

Dados de treino: 16.228

Dados de teste: 4.057



# Treinamento do Modelo

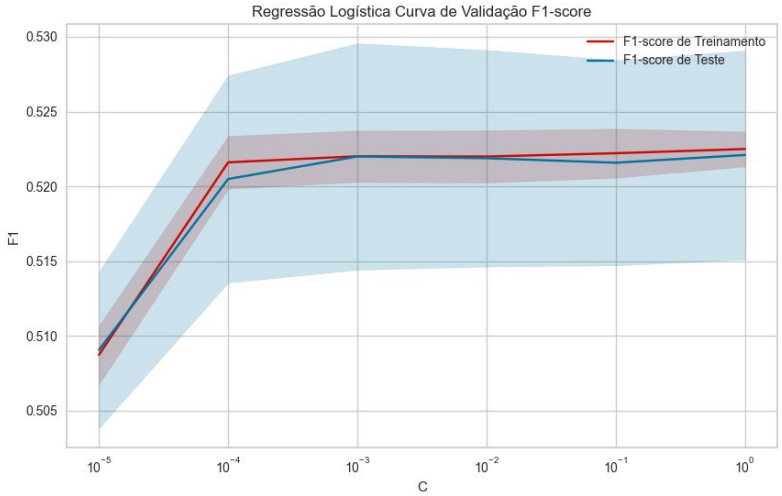
# Pipeline de Treinamento do Modelo

**Regressão Logística - Validação Cruzada**

A validação cruzada é utilizada para que o modelo consiga generalizar o máximo possível os dados, utilizando diferentes porções do conjunto de dados para o treinamento.

A curva de validação é utilizada para entender o comportamento do modelo com diferentes parâmetros quando aplicado a validação cruzada.

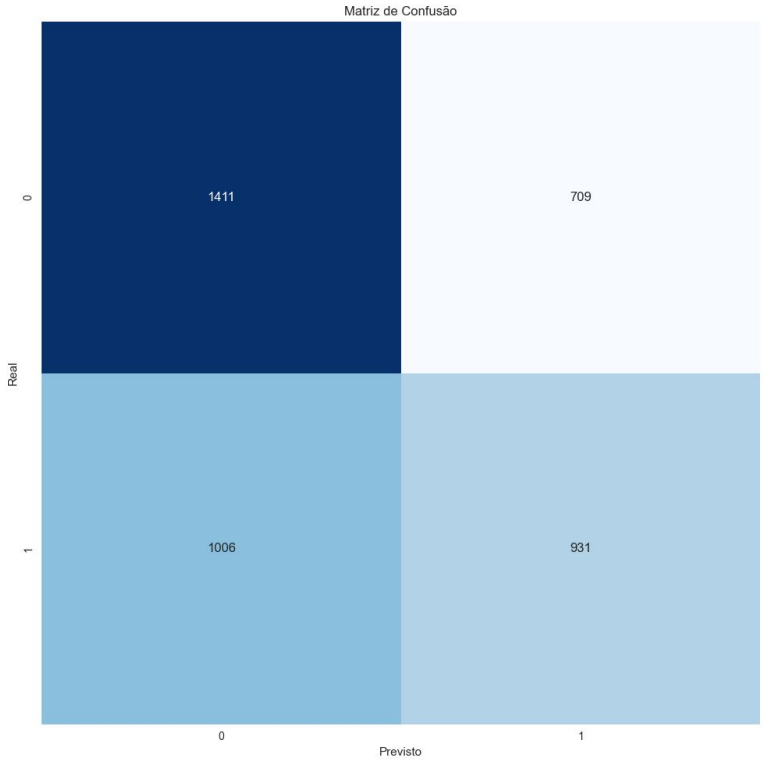
Na regressão logística seu principal uso é para comparar o parâmetro C.



# Pipeline de processamento dos dados

**Regressão Logística - Classificação**

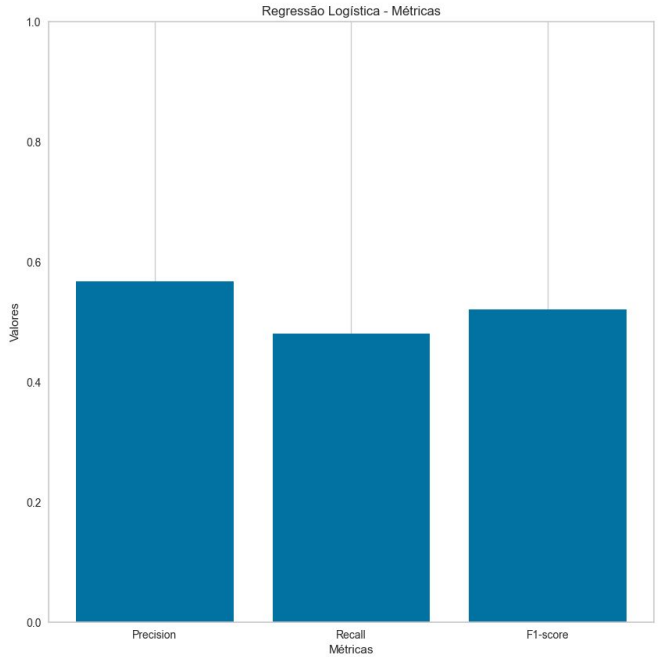
Analisando a matriz de confusão da regressão logística, o modelo apresentou resultados abaixo do esperado, prevendo muitos valores na classe ‘0’.



# Pipeline de processamento dos dados

**Regressão Logística - Classificação**

O modelo está apresentando um recall e F1-Score abaixo do esperado, além disso a sua precisão está abaixo dos 60% o que indica baixo desempenho.

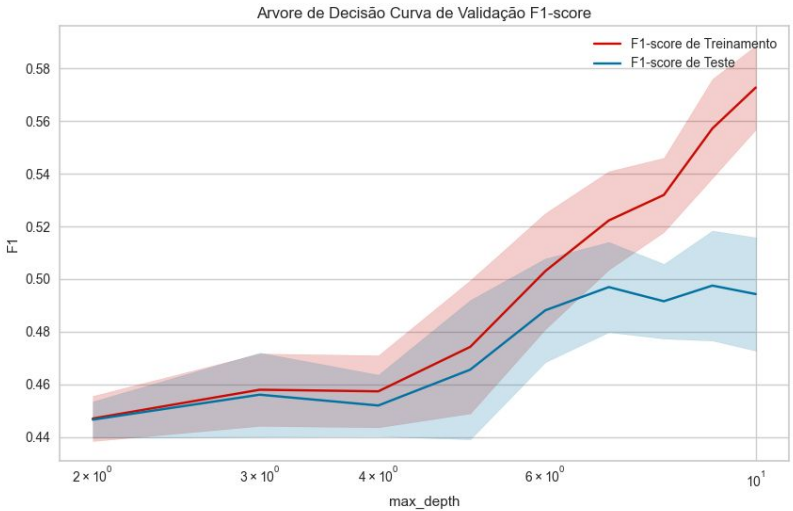


# Pipeline de Treinamento do Modelo

**Árvore de Decisão - Validação Cruzada**

A curva de aprendizado apresentou resultados superiores na base de treinamento como o esperado. É perceptível que o modelo possui um desempenho próximo na base de treino quanto na de teste até 6 e aproximadamente 8 de 'max_depth'.

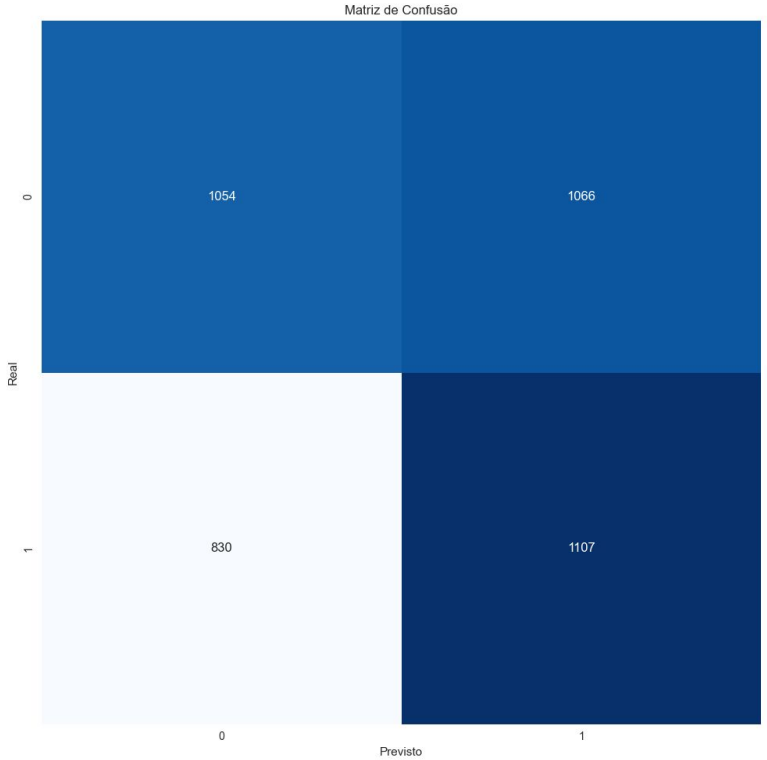
Após esses valores se caracteriza uma desconexão de treino e teste gerando o overfitting, dessa forma deve se manter valores inferiores, como 7.



# Pipeline de Treinamento do Modelo

**Árvore de Decisão - Classificação**

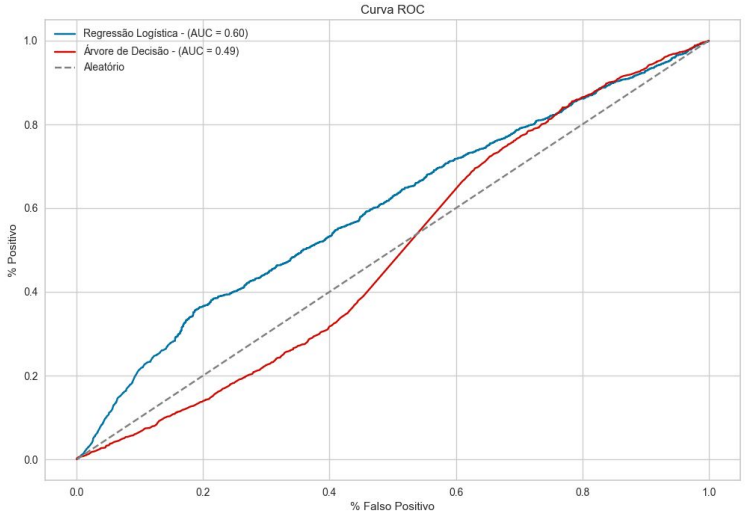
Após analisar a matriz de confusão do modelo é perceptível que o modelo está com uma baixa precisão, colocando muito dos valores na classe ‘1’, gerando muitos falsos positivos, para um melhor resultado o ideal seria analisar se as variáveis informadas são suficientes ou se o modelo está sofrendo de overfitting.



# Pipeline de Treinamento do Modelo

**Seleção, finalização e registro**

Analisando a Curva ROC nenhum dos modelos é ideal para um ambiente de produção. Entretanto a Regressão Logística se sobressai com um desempenho superior, já que a Árvore de Decisão esta beirando a aleatoriedade.



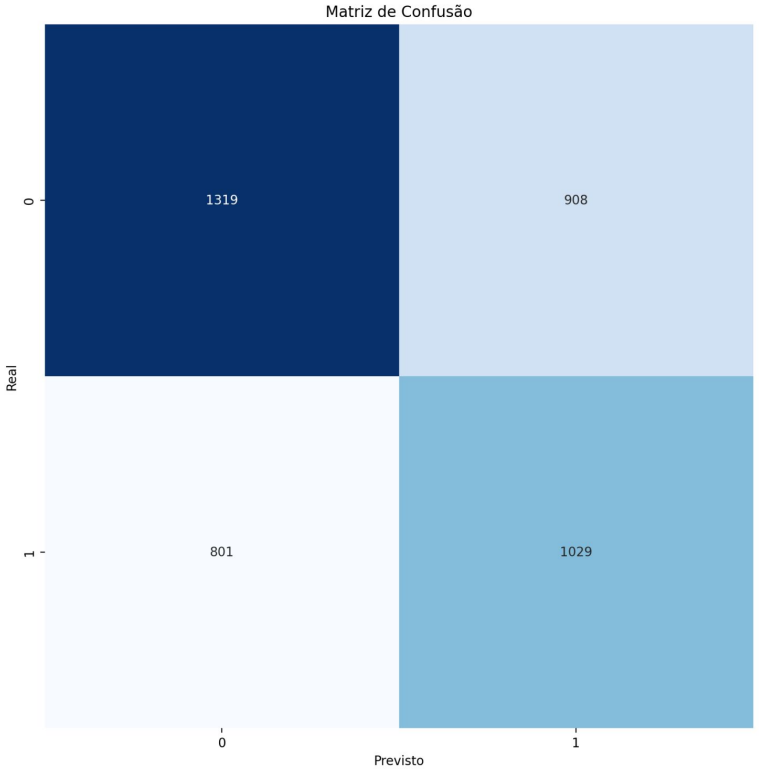
# Aplicação do Modelo

# Pipeline de Aplicação do Modelo

**Deployment**

O modelo em ambiente de produção conta com uma previsão de apenas 53% na classe 1 e 62% na classe 0.

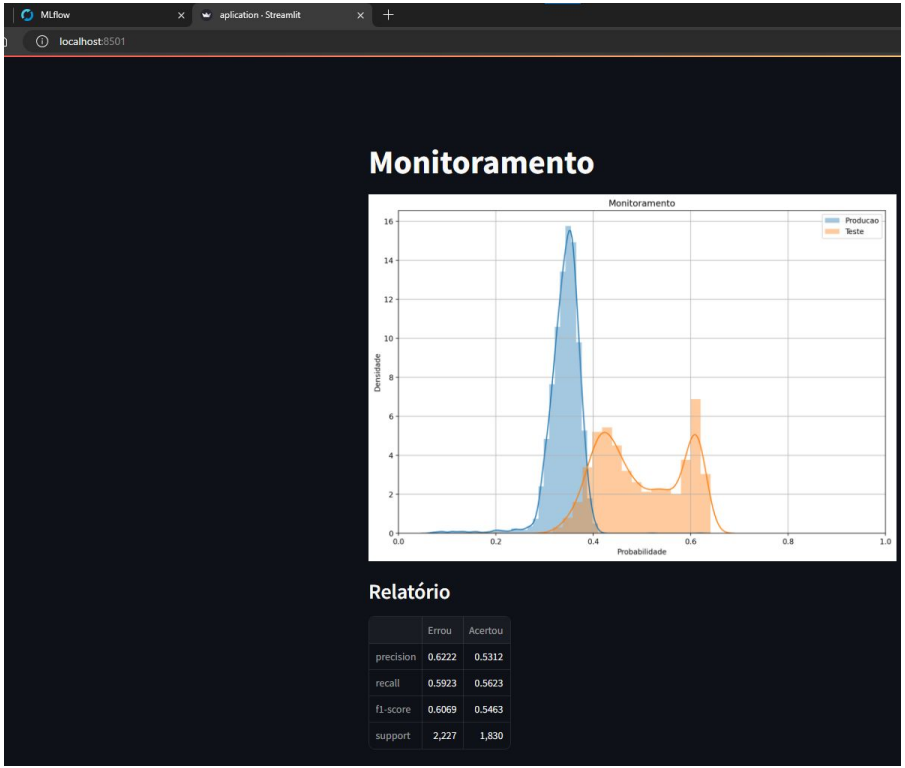
| | Errou | Acertou |
|---|---|---|
| precision | 0.6222 | 0.5312 |
| recall | 0.5923 | 0.5623 |
| f1-score | 0.6069 | 0.5463 |
| support | 2,227 | 1,830 |



# Pipeline de Aplicação do Modelo

**Interface Monitoramento**

Foi desenvolvido uma plataforma de monitoramento simples no streamlit com objetivo de servir ao modelo e acompanhar a densidade e a probabilidade na base de desenvolvimento e produção.



# Pipeline de Aplicação do Modelo

**Retreinamento**

**Estratégia reativa**:

A estratégia reativa se baseia que o modelo será treinado em situações específicas, ou seja reagir somente a situações em que se dá a necessidade de treinar novamente o modelo, como por exemplo agendamentos fixos.

**Estratégia preditiva**:

Nessa estratégia o modelo será treinado novamente com base em algum fator dinâmico, como por exemplo o seu desempenho. O modelo pode ser acompanhado em ambiente de produção por métricas e sempre que o desempenho cair abaixo de um threshold será treinado novamente.