**Proposta de Framework para Monitoramento Padronizado de Modelos de Machine Learning**

Gerhard Besser Neto¹; Helder Prado²

## Contextualização

A evolução da ciência de dados tem permitido avanços significativos no desenvolvimento de modelos preditivos para diferentes aplicações, como detecção de fraudes, análise de crédito e recomendação de produtos. No setor bancário, especificamente, modelos de machine learning são amplamente utilizados para prevenir fraudes financeiras, otimizando a identificação de comportamentos anômalos e reduzindo perdas. No entanto, após serem implantados em produção, esses modelos estão sujeitos a mudanças nas características dos dados e no comportamento dos usuários, o que pode levar à deterioração de sua performance. A ausência de monitoramento padronizado pode atrasar a detecção de problemas, comprometendo os resultados das análises e as decisões empresariais.

Para enfrentar esse desafio, é essencial implementar sistemas de monitoramento automatizados, que permitam avaliar continuamente o desempenho dos modelos em produção. Sistemas como esses não apenas identificam desvios nas distribuições de dados, mas também possibilitam a adaptação rápida a novos contextos, garantindo que os modelos continuem confiáveis e eficientes ao longo do tempo.

Este trabalho propõe um framework padronizado e extensível para monitoramento de modelos de machine learning em produção. A solução é projetada para ser configurada rapidamente, suportando múltiplas métricas — como Population Stability Index (PSI), Kolmogorov-Smirnov (KS) e análise de percentuais de fraudes por faixa percentil — e sendo escalável para processar grandes volumes de dados. A principal contribuição do framework é sua arquitetura modular, que facilita a implementação de monitoramentos para novos modelos de forma rápida e padronizada, permitindo futuras expansões com métricas customizadas.

O objetivo deste trabalho é propor um framework que automatize o monitoramento de modelos preditivos, com foco na padronização e na escalabilidade. A proposta será detalhada em sua arquitetura, implementação e resultados simulados, demonstrando sua viabilidade prática.

### Material e Métodos (Implementação do algoritmo)

Os códigos, scripts e exemplos de configuração utilizados neste trabalho estão disponíveis no repositório GitHub, acessível em: [gbessern/framework\_monitoramento](https://github.com/gbessern/framework_monitoramento/tree/main)

#### Arquitetura Geral do Framework

O framework proposto foi projetado para oferecer uma solução padronizada, modular e extensível para o monitoramento de modelos em produção. Ele organiza-se em duas principais estruturas de diretórios:

1. Diretório Principal: Contém os códigos padronizados responsáveis por cálculos de métricas, scripts para processamento dos dados e gerenciamento das execuções. Esses códigos são reutilizáveis para qualquer modelo monitorado.
2. Diretório Específico para Cada Modelo: Cada modelo monitorado possui seu próprio diretório, que contém:

* Dados de Entrada: Inclui a baseline (dados históricos usados como referência) e as tabelas mensais (safras) processadas.
* Arquivos de Configuração (JSON): Dois arquivos JSON que configuram o monitoramento:
* Parâmetros do Modelo: Define os scores, segmentações (aberturas), variáveis monitoradas, e outras configurações importantes.
* Parâmetros de Execução: Configuram o período de análise, inclusão da baseline e outros parâmetros operacionais.
* Resultados: Arquivos gerados após o processamento do monitoramento..
* Scripts Adicionais: Scripts dedicados à geração das tabelas mensais, baseline e inicialização do processamento.

Essa organização modular permite que o monitoramento de cada modelo seja executado de forma independente, utilizando os mesmos códigos padronizados.

#### Funcionamento do Framework

O funcionamento do framework segue um fluxo bem definido, dividido em etapas:

##### Configuração Inicial

O usuário fornece dois arquivos JSON que parametrizam o monitoramento:

* Parâmetros do Modelo: Este arquivo define parâmetros específicos do modelo a ser monitorado, como variáveis monitoradas, os scores, os percentis de análise, segmentações e pastas de dados. Segue descrição dos elementos que precisam ser definidos.
  + aberturas: Segmentações (e.g., grupos, regiões) para análise independente.
  + data\_folder: Pasta onde os dados de entrada estão armazenados.
  + data\_prefix: Prefixo padrão para os arquivos de dados.
  + result\_folder: Pasta onde os resultados serão salvos.
  + variavel\_resposta: nome da coluna que indica se um evento ocorreu (e.g., fraude).
  + scores: Configurações para os scores monitorados:
    - escores\_especiais: Valores específicos no score que exigem tratamento especial.
    - percentis\_ppv: Percentis usados para análise de proporção de fraudes.
    - percentis\_dist: Percentis usados para distribuição e cálculo de PSI.
    - acumula\_maiores: Define se fraudes acumulam em faixas maiores ou menores.
* Parâmetros de Execução: este arquivo configura aspectos gerais do processo de monitoramento, como período a ser monitorado ou se as métricas da base de treino devem ser calculadas.
  + As métricas da base de treino servem como referencial para saber se o modelo está com uma performance esperada ou não. Porém, por ser composta por várias safras históricas e estáticas, as métricas de performance podem ser calculadas uma única vez, e omitidas em novas execuções, reduzindo a carga de processamento.

##### Geração de Dados

O usuário deve disponibilizar scripts para gerar as tabelas abaixo:

* Baseline: Dados históricos que servem como referência para comparação com as safras.
* Tabelas Mensais (Safras): Dados mais recentes, contendo a variável resposta, necessários para cálculos.

##### Processamento Automatizado

Após a configuração inicial, o framework processa os dados para calcular as métricas (PSI, KS e percentuais de fraudes). O processamento é realizado com PySpark, garantindo escalabilidade e eficiência para grandes volumes de dados. Além disso, é possível definir se deve ser calculado métricas para a base de treino.

Adicionalmente, o framework permite a análise segmentada dos modelos, processando métricas para diferentes grupos (e.g., tipos de clientes ou regiões geográficas). Essa funcionalidade garante uma visão detalhada do desempenho do modelo em subgrupos específicos.

#### Métricas Implementadas

O framework oferece suporte para três principais métricas: Population Stability Index (PSI), Kolmogorov-Smirnov (KS) e Análise de Percentuais de Fraudes. Abaixo é detalhado cada métrica e seu papel no monitoramento. As métricas são calculadas separadamente para cada variável e safra sendo monitorada, além de gerar as métricas para cada combinação de segmentações existentes.

#### 1. Population Stability Index (PSI)

O PSI é usado para medir a estabilidade da distribuição dos scores ao longo do tempo. Ele compara a distribuição da baseline (dados históricos) com a distribuição dos dados das safras mensais (dados mais recentes).

Essa métrica é essencial para identificar mudanças significativas na população de entrada, que podem indicar alterações no comportamento dos dados e afetar a performance do modelo.

#### 2. Kolmogorov-Smirnov (KS)

O KS é uma métrica utilizada para avaliar o poder discriminatório do modelo, ou seja, sua capacidade de separar fraudes e não fraudes. Ele calcula a maior diferença entre as taxas cumulativas de fraudes e de não fraudes ao longo dos scores.

#### 3. Análise de Percentuais de Fraudes

Essa métrica verifica a concentração de fraudes em diferentes faixas percentis do score, identificando as áreas de maior risco.

Processo de Análise:

* Divide-se a população em percentis com base no score (e.g., 10%, 20%, ..., 100%).
* Calcula-se o percentual de fraudes em cada faixa:
  + Percentual Total de Fraudes: Proporção de todas as fraudes contidas em cada faixa percentil.
  + Percentual Relativo de Fraudes: Proporção de fraudes dentro da faixa específica em relação ao total de casos na faixa.

O framework ajusta os percentis de forma dinâmica para lidar com diferentes comportamentos dos scores:

* Fraudes Acumulam nas Menores Faixas: Quando scores baixos indicam maior risco, a análise considera diretamente as faixas inferiores (P ≤ X).
* Fraudes Acumulam nas Maiores Faixas: Quando scores altos indicam maior risco, os percentis são ajustados para considerar as faixas superiores (P ≥ 100 − X). Por exemplo, ao analisar 30% da população mais arriscada, o framework considera os casos em que o score está acima do percentil 70.

A análise de percentuais de fraudes é particularmente útil para priorizar ações em áreas críticas do score e monitorar a efetividade do modelo em identificar riscos. Os resultados são salvos em arquivos CSV organizados por safra.

#### Resultados e Discussão

O framework proposto foi projetado para padronizar e facilitar o monitoramento de modelos de machine learning em produção. Sua arquitetura modular e flexível permite que novos modelos sejam rapidamente configurados e monitorados, utilizando métricas amplamente reconhecidas, como PSI, KS e análise de percentuais de fraudes.

A proposta foi implementada e validada em um ambiente de testes com dados simulados, demonstrando sua capacidade de atender a cenários reais. A seguir, são apresentados os principais resultados e discussões relacionados ao funcionamento e benefícios do framework.

Para validar o framework, foram realizados testes simulados com dados representando baseline e safras mensais, configurados por meio de arquivos JSON. O processamento automatizado das métricas (PSI, KS e percentuais de fraudes) demonstrou que o framework é adaptável a diferentes modelos, eficiente no processamento de grandes volumes de dados com PySpark e confiável na detecção de mudanças na performance dos modelos.

Embora os testes utilizassem dados simulados, o processo validou a consistência e a reprodutibilidade do framework.

Os principais benefícios do framework incluem:

* Padronização: O framework estabelece um processo uniforme para monitorar modelos, o que facilita a interpretação dos resultados e a comunicação entre equipes.
* Escalabilidade: A arquitetura modular permite que novos modelos sejam adicionados com facilidade, utilizando os mesmos códigos e scripts padronizados.
* Flexibilidade: Suporte a diferentes métricas, como PSI, KS e percentuais de fraudes. Possibilidade de monitorar segmentos específicos (aberturas) para análises detalhadas. Configuração simplificada por meio de arquivos JSON, permitindo ajustes rápidos e customizações específicas para cada modelo.
* Automação: Geração automática de arquivos organizados por safra, facilitando a análise contínua. Redução do tempo necessário para implementar monitoramentos, permitindo que as equipes de dados foquem na interpretação dos resultados em vez de na configuração técnica.
* Reprodutibilidade: Os scripts padronizados garantem que o mesmo processo possa ser replicado para diferentes modelos e contextos, com garantia de consistência nos cálculos e relatórios.

Embora o framework tenha demonstrado eficiência e aplicabilidade, algumas limitações e oportunidades de melhoria foram identificadas:

* Dependência de Configurações: A configuração inicial exige que os parâmetros dos modelos sejam fornecidos com precisão. Erros ou inconsistências nos arquivos JSON podem comprometer a execução.
* Expansão de Métricas: Embora o framework suporte PSI, KS e percentuais de fraudes, ele pode ser estendido para incluir métricas adicionais, como métricas customizadas para casos específicos.

A proposta contribui para simplificar e automatizar o monitoramento, reduzindo a carga operacional sobre as equipes de dados e promovendo uma análise mais confiável e consistente. Com melhorias futuras, como a integração com sistemas em tempo real, o framework pode se tornar ainda mais poderoso e relevante em aplicações práticas.

**Considerações Finais**

Este trabalho apresentou um framework automatizado e padronizado para monitoramento de modelos de machine learning, validado com dados simulados e focado em escalabilidade, flexibilidade e facilidade de uso. Sua modularidade, organização por diretórios e configuração via arquivos JSON permitem monitoramentos consistentes e rápidos, atendendo a diferentes cenários. O framework demonstrou eficácia na identificação de mudanças na performance de modelos, facilitando intervenções e garantindo confiabilidade em produção. Entre as próximas melhorias, destaca-se a inclusão de métricas customizadas, ampliando ainda mais sua aplicabilidade em diferentes contextos.