

Pós-graduação MIT em Inteligência Artificial, Machine Learning & Deep Learning

Bloco: Aprendizado de Dados em Tempo Real [23E1-23E1]

Disciplina: Engenharia de Machine Learning [23E1_3]

Docente: Felipe Fink Grael

Aluno: Winicius Botelho Faquiere



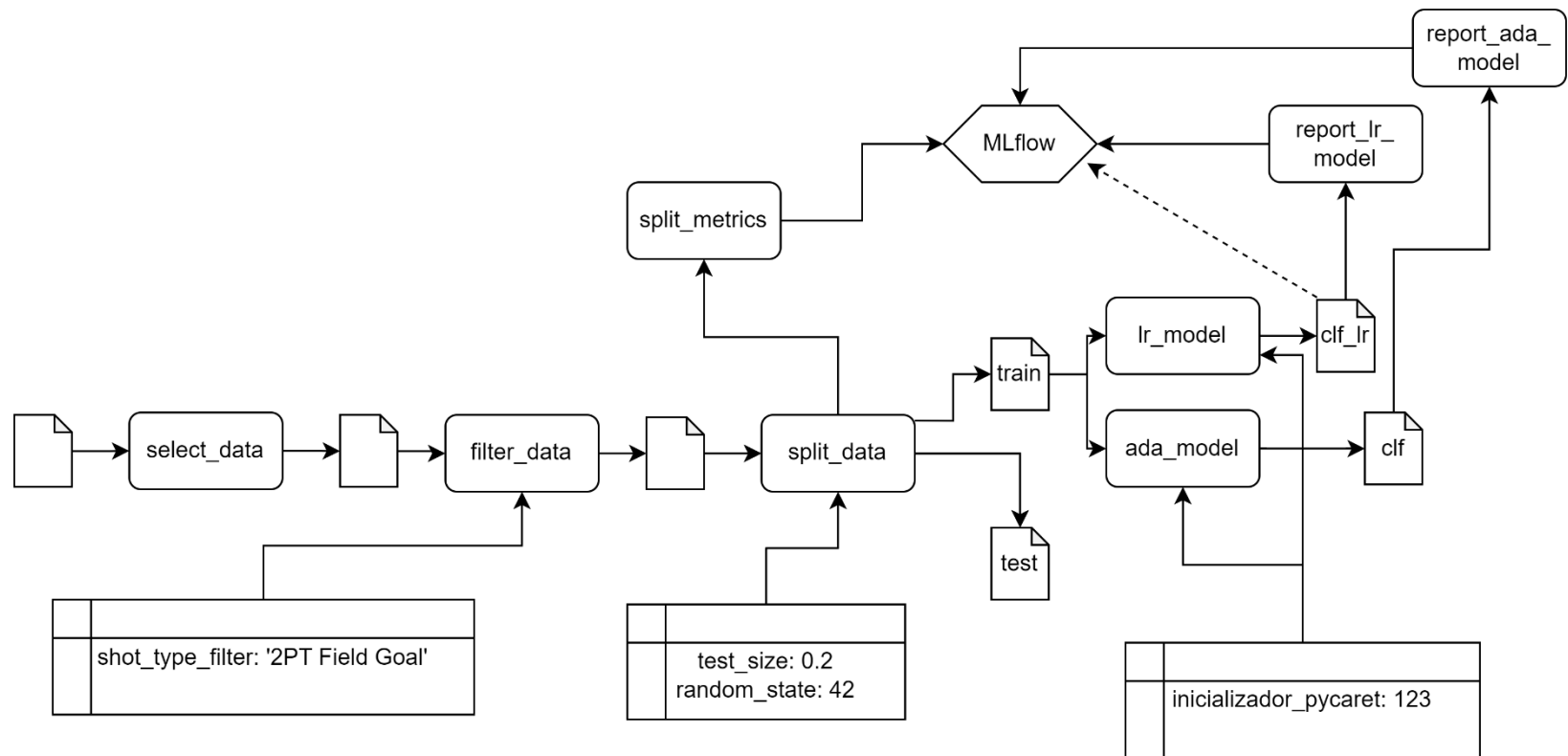
Instituto Infnet
27 anos de história



1. A solução criada nesse projeto deve ser disponibilizada em repositório git e disponibilizada em servidor de repositórios (Github (recomendado), Bitbucket ou Gitlab). O projeto deve obedecer o Framework TDSP da Microsoft. Todos os artefatos produzidos deverão conter informações referentes a esse projeto (não serão aceitos documentos vazios ou fora de contexto). Escreva o link para seu repositório.

A solução criada nesse projeto está disponível em: <https://github.com/wfaquiere/infnet-engenharia-machine-learning/tree/main/blackmambapredictor>. Além disso, utilizou-se no presente trabalho o framework kedro que facilita a organização e a documentação dos artefatos do projeto. O Kedro é uma ferramenta de gerenciamento de fluxo de trabalho de dados que segue uma estrutura muito semelhante à do TDSP.

2. Iremos desenvolver um preditor de arremessos usando duas abordagens (regressão e classificação) para prever se o "Black Mamba" (apelido de Kobe) acertou ou errou a cesta. Para começar o desenvolvimento, desenhe um diagrama que demonstra todas as etapas necessárias em um projeto de inteligência artificial desde a aquisição de dados, passando pela criação dos modelos, indo até a operação do modelo.



3. Descreva a importância de implementar pipelines de desenvolvimento e produção numa solução de aprendizado de máquinas.

O uso de pipelines de desenvolvimento e produção é crucial para o sucesso de uma solução de aprendizado de máquina, pois permite que as equipes de desenvolvimento e operações trabalhem juntas para criar, testar e implantar modelos de maneira eficiente. A implementação de pipelines oferece diversos benefícios, como melhoria da qualidade do modelo, redução de erros e tempo de inatividade, maior eficiência e produtividade, escalabilidade e colaboração mais estreita entre equipes. Os pipelines podem incluir etapas como aquisição e pré-processamento de dados, seleção de algoritmos, treinamento e validação de modelos, implantação em ambientes de produção, monitoramento de desempenho e atualização de modelos quando necessário.

4. Como as ferramentas Streamlit, MLFlow, PyCaret e Scikit-Learn auxiliam na construção dos pipelines descritos anteriormente? A resposta deve abranger os seguintes aspectos:
 - a. Rastreamento de experimentos;
 - b. Funções de treinamento;
 - c. Monitoramento da saúde do modelo;
 - d. Atualização de modelo;
 - e. Provisionamento (Deployment).

As ferramentas Streamlit, MLFlow, PyCaret e Scikit-Learn são úteis na construção de pipelines de aprendizado de máquina. Elas ajudam em diferentes aspectos do processo de desenvolvimento de modelos, incluindo rastreamento de experimentos, funções de treinamento, monitoramento da saúde do modelo, atualização de

modelo e provisionamento. O MLflow e o PyCaret possuem recursos de rastreamento de experimentos, o Scikit-Learn fornece uma ampla variedade de algoritmos de aprendizado de máquina pré-implementados, e o PyCaret fornece uma interface simples e intuitiva para treinar modelos. Além disso, o PyCaret é uma biblioteca de AutoML (Aprendizado de Máquina Automatizado), o que ajuda a reduzir o tempo necessário para implementar vários modelos de aprendizado de máquina. O MLflow e o PyCaret permitem monitorar a saúde do modelo e facilitam a atualização do modelo, e o Streamlit, MLFlow e PyCaret permitem a implantação do modelo em vários ambientes. O Streamlit ajuda na criação de interfaces amigáveis, como aplicativos web, de forma muito similar a uma outra biblioteca bastante conhecida chamada Shiny. É uma excelente forma de disponibilizar o modelo para o público em geral.


5. Com base no diagrama realizado na questão 2, aponte os artefatos que serão criados ao longo de um projeto. Para cada artefato, indique qual seu objetivo.

- Plano de projeto: diagrama que detalha as atividades necessárias para atender aos requisitos do projeto, incluindo as etapas de coleta de dados, treinamento do modelo e teste do modelo.
- Conjunto de dados: dataset que inclui informações sobre os arremessos de Kobe, como a distância do arremesso, posição da quadra, tempo restante, entre outros.
- Pré-processamento de dados: processo de limpeza e transformação dos dados para torná-los adequados para o treinamento do modelo. Isso pode incluir a remoção de valores ausentes, normalização de dados, transformação de dados categóricos em numéricos, entre outros.
- train_test_metrics: conjunto de treinamento e conjunto de teste, bem como o parâmetro, informando o percentual utilizado para divisão do conjunto de dados em treino teste.
- lr_metrics: métricas do modelo de regressão logística: a função custo "log loss" usando a base de teste.

- ada_metrics: métricas do modelo de AdaBoost: a função custo "log loss" usando a base de teste e o F1-Score.
- Modelos de classificação utilizados: modelo de aprendizado de máquina que classifica cada arremesso como "acerto" ou "erro" com base nas características dos dados.
- Implantação do modelo: processo de implementação do modelo em um ambiente de produção para uso em produção.
- Monitoramento do modelo: processo contínuo de monitoramento do modelo em produção para garantir que ele esteja funcionando conforme o esperado e para identificar quaisquer problemas ou anomalias.

6.Implemente o pipeline de processamento de dados com o mlflow, rodada (run) com o nome "PreparacaoDados": 

a. Os dados devem estar localizados em "/Data/kobe_dataset.csv" 

b. Observe que há dados faltantes na base de dados! As linhas que possuem dados faltantes devem ser desconsideradas. Você também irá filtrar os dados onde o valor de shot_type for igual à 2PT Field Goal. Ainda, para esse exercício serão apenas consideradas as colunas: i. lat, ii. Lng, iii. minutes remaining,iv. Period, v. playoffs, vi. shot_distance. 

A variável shot_made_flag será seu alvo, onde 0 indica que Kobe errou e 1 que a cesta foi realizada. O dataset resultante será armazenado na pasta "/Data/processed/data_filtered.parquet". Ainda sobre essa seleção, qual a dimensão resultante do dataset?

20285 linhas × 7 colunas

c. Separe os dados em treino (80%) e teste (20 %) usando uma escolha aleatória e estratificada. Armazene os datasets resultantes em `"/Data/operalization/base_{train|test}.parquet` . Explique como a escolha de treino e teste afetam o resultado do modelo final. Quais estratégias ajudam a minimizar os efeitos de viés de dados.

A escolha dos conjuntos de treinamento e teste pode afetar significativamente o resultado do modelo final. Quando os dados são divididos em conjuntos de treinamento e teste, o modelo é ajustado nos dados de treinamento e avaliado nos dados de teste. Se a divisão não for feita adequadamente, o modelo pode ser superajustado (*overfitting*) ou subajustado (*underfitting*), o que pode resultar em baixo desempenho do modelo em dados não vistos.

Uma estratégia comum para minimizar o viés de dados na divisão de dados em conjuntos de treinamento e teste é a validação cruzada. A validação cruzada é um método que permite que o modelo seja ajustado em diferentes conjuntos de treinamento e avaliado em diferentes conjuntos de teste, de modo que a avaliação do modelo seja mais robusta. Existem várias formas de validação cruzada, como K-fold, Leave-one-out, entre outras.

Outra estratégia é a amostragem estratificada, como mencionado anteriormente. A amostragem estratificada garante que a proporção de cada classe seja mantida em ambos os conjuntos de treinamento e teste, o que ajuda a reduzir o viés de dados.

Também é importante escolher uma proporção adequada entre os conjuntos de treinamento e teste. Em geral, um bom ponto de partida é usar uma divisão de 80/20 ou 70/30 para treinamento e teste, respectivamente, mas isso pode variar dependendo do tamanho do conjunto de dados e da complexidade do modelo.

Finalmente, a escolha do conjunto de dados de treinamento e teste deve ser feita de forma aleatória. Caso contrário, pode haver uma tendência para determinadas observações serem incluídas em um dos conjuntos, o que pode resultar em um modelo que não generaliza bem para novos dados.

Em resumo, a escolha de treinamento e teste é crítica para o desempenho do modelo final. A validação cruzada, amostragem estratificada e escolha aleatória dos conjuntos de dados de treinamento e teste são estratégias importantes para minimizar o efeito de viés de dados e obter um modelo que generalize bem para novos dados.

d. Registre os parâmetros (% teste) e métricas (tamanho de cada base) no MIFlow ✓

Vide código.

7. Implementar o pipeline de treinamento do modelo com o Miflow usando o nome "Treinamento" ✓

a. Com os dados separados para treinamento, treine um modelo com regressão logística do sklearn usando a biblioteca pyCaret. ✓

Vide código.

b. Registre a função custo "log loss" usando a base de teste ✓

Vide código.

c. Com os dados separados para treinamento, treine um modelo de classificação do sklearn usando a biblioteca pyCaret. A escolha do algoritmo de classificação é livre. Justifique sua escolha. ✓

Vide código.

d. Registre a função custo "log loss" e F1_score para esse novo modelo ✓

Vide código.

8. Registre o modelo de classificação e o disponibilize através do MLFlow através de API. ✓

Modelo foi registrado no MLflow como ADA_MODEL tal que url = 'http://localhost:5000/invocations/ADA_MODEL'

Selecione agora os dados da base de dados original onde shot_type for igual à 3PT Field Goal (será uma nova base de dados) e através da biblioteca requests, aplique o modelo treinado. Publique uma tabela com os resultados obtidos e indique o novo log loss e f1_score. ✓

Dataset	Log Loss	F1 Score
Treinamento (2PT Field Goals)	14.765.726	0.468330
Nova Base (3PT Field Goals)	11.914.707	0.027173

a. O modelo é aderente a essa nova base? Justifique.

O modelo AdaBoost treinado na base de dados de 2PT Field Goals não parece ser aderente à nova base de dados de 3PT Field Goals. O valor de log loss reduziu em relação ao valor de treinamento (11.91 x 14.77), o que pode indicar que o modelo está tendo um desempenho melhor na nova base de dados em termos de função custo, mas o valor do F1 é muito baixo (0.03 x 0.47), sugerindo que o modelo está tendo dificuldade em classificar corretamente os exemplos na nova base de dados.

b. Descreva como podemos monitorar a saúde do modelo no cenário com e sem a disponibilidade da variável resposta para o modelo em operação.

Em um cenário em que a variável resposta está disponível, podemos monitorar a saúde do modelo por meio da análise de suas métricas de desempenho, como a função de perda log loss e o F1-Score. Essas métricas nos dão uma medida objetiva do quão bem o modelo está se saindo na tarefa de classificação.

Por exemplo, se o log loss e F1 estiverem baixos e estáveis ao longo do tempo, isso indica que o modelo está funcionando bem e é capaz de fazer previsões precisas com base nos dados disponíveis. Se houver uma queda repentina no desempenho do modelo, isso pode indicar que há um problema com a qualidade dos dados de entrada ou que o modelo precisa de ajustes.

No entanto, em um cenário em que a variável resposta não está disponível, pode ser mais difícil monitorar a saúde do modelo. Nesse caso, pode ser necessário recorrer a outras técnicas, como a validação cruzada ou a análise de sensibilidade, para avaliar a capacidade do modelo de generalização para novos dados.

c. Descreva as estratégias reativa e preditiva de retreinamento para o modelo em operação.

As estratégias de retreinamento para modelos em operação podem ser classificadas em duas categorias principais: reativa e preditiva.

A estratégia reativa de retreinamento envolve monitorar as métricas de desempenho do modelo em tempo real e acionar o processo de retreinamento quando as métricas de desempenho começam a cair abaixo de um determinado limiar. Por exemplo, se o modelo começar a fazer previsões imprecisas ou inconsistentes, a estratégia reativa pode acionar um processo de retreinamento para tentar melhorar o desempenho do modelo. Essa abordagem pode ser útil em situações em que o ambiente operacional é estável e as mudanças nos dados de entrada são relativamente previsíveis. Por outras

palavras, a estratégia reativa espera o modelo começar a errar para só então iniciar o processo de retreinamento do modelo.

Já a estratégia preditiva de retreinamento envolve a realização de retreinamento periódico do modelo com base em previsões futuras. Essa abordagem envolve o uso de técnicas de previsão para estimar quando o desempenho do modelo começará a cair e programar o retreinamento com antecedência. Por exemplo, se os dados de entrada apresentam variações sazonais, o modelo pode ser retreinado antes do início de cada temporada para garantir que esteja preparado para lidar com os novos dados. Essa abordagem pode ser útil em ambientes operacionais dinâmicos, onde as mudanças nos dados de entrada são imprevisíveis.

Ambas as estratégias podem ser complementares e podem ser usadas em conjunto para garantir que o modelo esteja sempre atualizado e pronto para lidar com novos dados. A escolha da estratégia mais adequada dependerá das características do ambiente operacional e dos requisitos do modelo.

9. Implemente um dashboard de monitoramento da operação usando Streamlit. 

Vide código.