

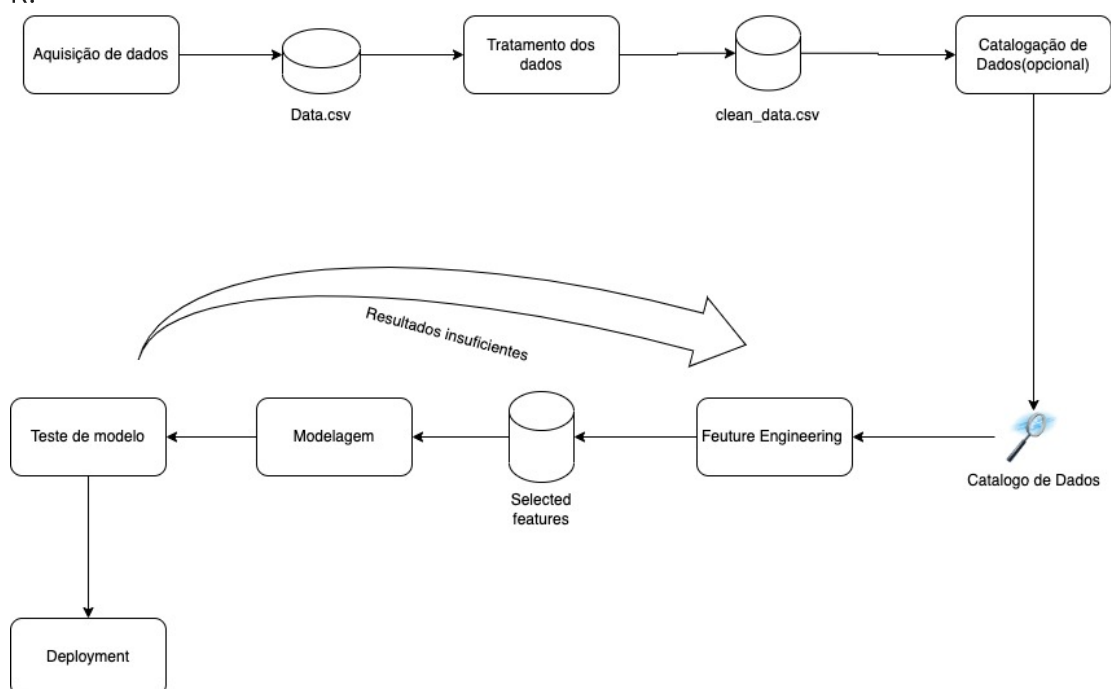
Respostas do questionário do trabalho.

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.

R: https://github.com/walterpl/infnet_IA_course/tree/main/eng-ml

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.'

R:



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

R: Quando um projeto já está em uso, eventualmente alguma atualização/ correção/bug/modificação se faz necessária, para garantir que o desenvolvimento do código futuro não afete o código presente, o mesmo é desenvolvido de forma isolada do código em uso, que é chamado de produção. O código em desenvolvimento é criado em um ambiente chamado de desenvolvimento, que normalmente é um espelho do ambiente de produção, depois de desenvolvido e testado o mesmo é entregue ao ambiente de produção, normalmente sem a percepção do usuário final.

4- Como as ferramentas Streamlit, MLFlow, PyCaret e Scikit-Learn auxiliam na construção dos pipelines descritos anteriormente? A resposta deve abranger os seguintes aspectos:

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.

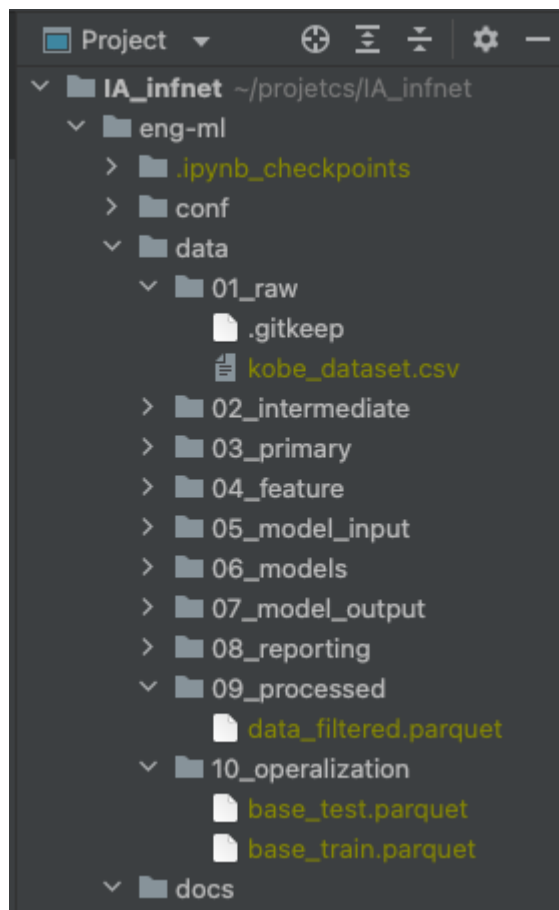
R: Os dataframes que seriam o data.csv, clean_data.parquet, train_base.parquet, test_base.parquet e o modelo best_model.pkl.

R: Todas as ferramentas auxiliam em diferentes partes durante a produção de uma solução IA, começando pelo **MLFlow**, que é uma ferramenta voltada para MLops, a qual nos permite manter o histórico de todos os modelos, seus parâmetros e seus resultados de treino, como nos permite testar as modelagens como diferentes funções de um pipeline, para garantir que todo código da solução seja não apenas compilável, mas também funcional. **Scikit-Learn** é a ferramenta que possui diversos modelos de ML include suas funções auxiliares para tratamentos de dados, e a partir dela que monta-se o modelo. **PyCaret** é a ferramenta utilizada para construir, testar e escolher o melhor modelo no momento de teste, ela utiliza os recursos do Scikit-Learn internamente para compilar os modelos e seus resultados são armazenados e catalogados no MLFlow. **StreamLit** é a porta de saída do modelo para a produção, pois é a partir dele que o modelo deixará o ambiente de desenvolvimento e será enviado ao ambiente de produção o tornando disponível via API.

5

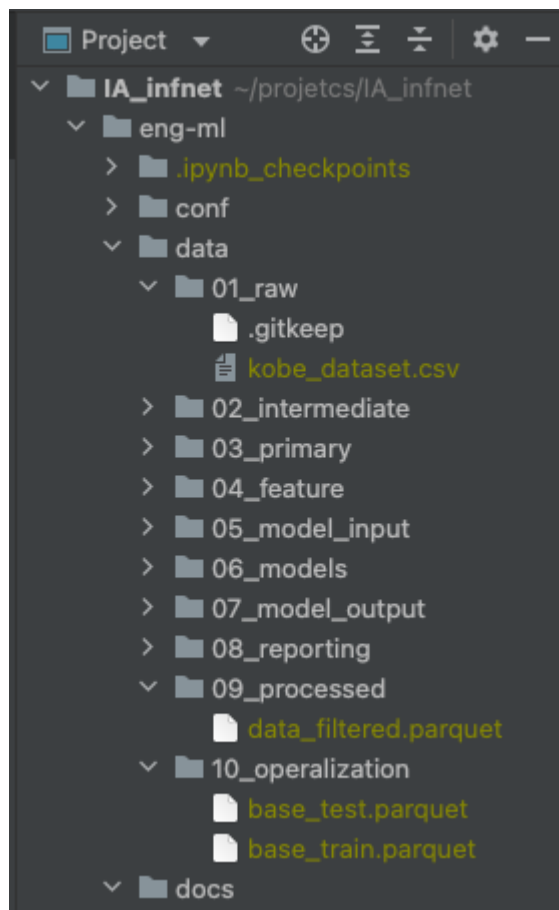
6-

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



R:

B) 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?



R:

mlflow 2.2.2 Experiments Models

eng_ml >

__default__

Run ID: 1f454fc4e2f34bfb8325484f41ac24bc

Date: 2023-09-14 14:28:10

Duration: 321ms

Status: Succeeded

> Description [Edit](#)

> Parameters (1)

> Metrics (2)

Name	Value
DF_columns_count 📈	7
DF_row_count 📈	20285

> Tags (6)

C) 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.

R: Partindo do pressuposto que a classe está desbalanceada, a separação da mesma diretamente pelo método 80|20 pode resultar em uma base de teste enviesada, onde a maioria dos dados pertencem a classe X, resultando em um modelo tendencioso a escolher a classe X como resultado, caso a base de teste seja oposta a precisão tenderá a ser baixa, o mesmo para o f1-score entretanto talvez o recall seja um pouco maior. Algumas estratégias podem ser adotadas para evitar o vies dos dados desbalanceados, o primeiro é excluir os dados da classe de maior incidência, caso o resultado final não seja uma base curta, o segundo método seria gerar valores sintéticos para a classe de menor incidência visando balancear a classe, também podemos excluir a feature caso o peso dela no modelo seja baixo ou tenha alta correlação com outra feature balanceada e por ultimo a correlação cruzada pode amenizar os efeitos da base enviesada pois dividirá a mesma em multiplas partes, efetuará multiplos treinamentos testando essas partes menores umas contra as outras.

D) Registre os parâmetros (% teste) e métricas (tamanho de cada base) no MIFlow

eng_ml >

__default__

Run ID: 74c9ac70b31143f7b2de9b87fc065d1e

Date: 2023-0

Duration: 133ms

Status: FINISH

> Description [Edit](#)

> Parameters (1)

✓ Metrics (4)

Name	Value
test_df_columns_count 	7
test_df_rows_count 	4057
train_df_columns_count 	7
train_df_rows_count 	16228

> Tags (6)

R:

7- A e B)

eng_ml >

__default__

Run ID: 74c9ac70b31143f7b2de9b87fc065d1e

Date: 2023-1



Duration: 133ms

Status: FINIS

> Description [Edit](#)

> Parameters (1)

✓ Metrics (4)

Name	Value
test_df_columns_count 	7
test_df_rows_count 	4057
train_df_columns_count 	7
train_df_rows_count 	16228

> Tags (6)

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.

R: Continuando a utilizar a lib Pycarret, após testar todos os modelos uns contra os outros e utilizar o auto-tunning, o modelo com melhor resultado foi o Gradient Boosting classifier.

D) Registre a função custo "log loss" e F1_score para esse novo modelo.

[treino_best_model](#) >

Gradient Boosting Classifier

Run ID: f4f0102d83e64ae5a50249640b5fdadc

Date:

Duration: 260ms

Status:

> Description [Edit](#)

> Parameters (20)

✓ Metrics (9)

Name	Value
AUC 📈	0.589
Accuracy 📈	0.588
F1 📈	0.464
Kappa 📈	0.159
Log Loss 📈	6.575
MCC 📈	0.172
Prec. 📈	0.611
Recall 📈	0.373
TT 📈	0.389

R: `0.589` (F1)

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

R:

```
[97]: serialized = df_3PT.drop(columns=['shot_made_flag']).to_dict(orient='records')

[98]: %%capture
      serialized.pop('index')

[99]: data = {'dataframe_split':serialized}

[100]: import requests

[113]: data = {'dataframe_split':serialized}
      url = 'http://127.0.0.1:5000/invocations'

      result = requests.post(url,json=data)

[114]: result.status_code

[114]: 200

[112]: from json import loads

[115]: result = loads(result.text)

[117]: predicted = result['predictions']

[119]: predicted

[119]: [0.0,
      0.0,
      0.0,
      0.0,

[ ]: predicted

[122]: from sklearn.metrics import log_loss, f1_score, classification_report

[124]: print(classification_report(df_3PT.shot_made_flag,predicted))

              precision    recall  f1-score   support

    0.0         0.67      1.00      0.80       3630
    1.0         0.00      0.00      0.00       1782

 accuracy          0.34
 macro avg         0.34      0.50      0.40       5412
 weighted avg      0.45      0.67      0.54       5412

[ ]:
```

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

R: Não, o modelo não é capaz de inferir se Kobe conseguiria pontuar a partir da linha de 3 pontos. O fato ocorreu pois diversas features relacionadas a distancia e posicionamento foram omitidas do treino, pois da perspectiva do modelo a distancia,

latitude, longitude serão outliers para a cesta de 3 pontos e como todas estão acima dos threshold das de 2, o modelo definiu que Kobe não conseguiria marcar nenhum ponto.

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

R: O caso de dados com labels é mais simples, pois basta frequentemente rodar o modelo contra os novos dados, coletar as métricas e comparar com o histórico, caso um desvio acima do normal aconteça, analisar a qualidade dos dados ou reavaliar os hyperparametros do modelo se faz necessário. Para o caso onde não existe o label, a análise deve ser feita nos novos dados, os comparando com os dados históricos em busca de bruscas variações de média, covariância, correlação e etc, vale verificar também se alguma feature passou a assumir um valor antes inexistente(casos categóricos).

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

R: A preditiva segue conforme mencionado na questão anterior, testes com uma determinada frequência contra a base de dados em diferentes folds e a medida que esses dados aumentam, diminuem ou mudem, os testes e os resultados históricos desses sempre apresentaram o comportamento o modelo ao longo do tempo. O caso reativo normalmente se da por reclamação ou aviso do usuário final e para entender onde o problema ocorreu, faz-se necessário descobrir o porquê da classificação do modelo para determinada entrada, um teste de shape para verificar o peso de cada featura para a tomada de decisão e a comparação das features de input com features da base histórica ajudam a verificar a razão de determinada classificação.

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

R:

Monitoração de Saúde do Melhor Modelo de ML

Esse Pagina serve como modelo para monitoração das métricas de um modelo de ML.

Dados históricos no momento da criação são inexistentes.

Matrix de Confusão

	0	1
0	8520	2082
1	6161	3522

Tabela de Métricas

	precision	recall	f1-score	support
0.0	0.5803	0.8036	0.6740	10,602.0000
1.0	0.6285	0.3637	0.4608	9,683.0000
accuracy	0.5936	0.5936	0.5936	0.5936
macro avg	0.6044	0.5837	0.5674	20,285.0000
weighted avg	0.6033	0.5936	0.5722	20,285.0000

In []: