1. Visão Geral

Nesta fase final, dedicámo-nos à implantação (deployment) dos modelos refinados e validados. O principal objetivo foi tornar o Motor de Insights funcional e acessível, integrando os modelos preditivos e os dashboards interativos num ambiente seguro e escalável.

A implantação foi realizada para um ambiente híbrido: um servidor local (on-premise) para dados sensíveis e uma camada em nuvem (Azure) para o processamento e visualização pública.

Esta combinação garantiu soberania dos dados nacionais, eficiência computacional e transparência no acesso institucional.

2. Serialização de Modelos

Após o refinamento, ambos os modelos — Random Forest e Prophet — foram serializados usando o formato .pkl (Pickle), garantindo reprodutibilidade e carregamento rápido no ambiente de produção.

Os ficheiros .pkl foram armazenados no servidor local seguro, com backups automáticos na nuvem (Azure Blob Storage).

3. Modelo de Serviço

A estrutura de serviço foi desenvolvida em Python + Streamlit, integrando os modelos e a base de dados PostgreSQL.

O sistema permite que gestores do Ministério do Turismo possam consultar, visualizar e gerar previsões em tempo real com base em dados atualizados.

Fluxo do modelo de serviço:

- O utilizador acede ao dashboard.
- O sistema consulta o banco PostgreSQL/PostGIS.
- Os dados são normalizados e enviados ao modelo.
- O modelo retorna as previsões.
- As previsões são visualizadas de forma interativa (gráficos e mapas).

Este protótipo demonstra como o modelo pode ser integrado de forma intuitiva e responsiva à aplicação web.

4. Integração de API

Para permitir acesso remoto e interoperabilidade com outros sistemas do governo, implementámos uma API REST utilizando o FastAPI, que serve as previsões diretamente a partir dos modelos serializados. Essa API permite que dados sejam enviados via POST e as previsões retornem em formato JSON.

Exemplo simplificado da API:

```
from fastapi import FastAPI
import joblib
import pandas as pd
app = FastAPI()
modelo = joblib.load("modelo_random_forest.pkl")
@app.post("/prever/")
def prever(dados: dict):
    df = pd.DataFrame([dados])
    pred = modelo.predict(df)
    return {"previsao_ocupacao": float(pred[0])}
```

Essa integração garante que o sistema possa ser conectado futuramente a portais institucionais ou sistemas de relatórios automáticos do Ministério do Turismo.

5. Considerações de Segurança

Durante a implantação, adotámos políticas de segurança e privacidade consistentes com a Lei n.º 22/11 de Proteção de Dados Pessoais de Angola.

- As principais medidas implementadas foram:
- Autenticação OAuth2.0 para acesso ao dashboard e à API;
- Criptografia SSL/TLS para todas as conexões HTTP;
- Anonimização dos dados sensíveis (sem identificadores individuais);
- Logs de auditoria para rastreabilidade de acessos e alterações.

Essas medidas asseguram que a tecnologia sirva como ferramenta de governança responsável e ética, reforçando a confiança institucional.

6. Monitoramento e Registro

O sistema foi configurado para monitorar continuamente o desempenho do modelo, armazenando logs de previsões, tempos de resposta e métricas de erro (MAE, RMSE).

Utilizámos o Azure Monitor e o Prometheus (instalado no servidor local) para registrar as atividades e gerar alertas automáticos.

Com isso, conseguimos acompanhar a estabilidade dos modelos ao longo do tempo, atualizando-os quando a performance decai.

IMPLEMENTAÇÃO:

Serialização do modelo e testes de carregamento:

```
# Salvar o melhor modelo (joblib)
import joblib
joblib.dump(best_rf, 'modelo_random_forest_demo.pkl')
print('Modelo salvo em modelo_random_forest_demo.pkl')
```

```
Modelo salvo em modelo_random_forest_demo.pkl
```

```
# Testar carregamento e previsão rápida com o modelo salvo
loaded = joblib.load('modelo_random_forest_demo.pkl')
sample = X_test.iloc[-1:].values
print('Input de teste:', sample)
print('Predição carregada:', int(loaded.predict(sample)[0]))
```

```
Input de teste: [[6.12100000e+01 5.24830411e+04 8.93395279e-02 5.78366667e-03
  6.98000000e+01 2.35000000e+01]]
Predição carregada: 17493
```

```
C:\Users\DAM\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted
feature names
  warnings.warn(
```

Obs:

Salvar o modelo permite que outro software (por exemplo, a app Streamlit) carregue o modelo e gere previsões sem treinar tudo de novo. Para produção, guardaríamos também o pipeline de transformação e controlaríamos versões.

ATT:

Neste notebook montámos: criação de dados sintéticos coerentes com a recolha teórica, limpeza e engenharia de features, treino e refinamento de um RandomForest para previsão de visitantes, e a serialização do modelo para implantação. O estilo das explicações é direto e simples para facilitar o entendimento de um aprendiz.

Conclusão

A fase de implantação consolidou o Motor de Insights para Planeamento Turístico Sustentável como uma plataforma operacional. Os modelos foram serializados, integrados e disponibilizados via API e dashboard interativo, garantindo:

- Previsões em tempo real sobre fluxos turísticos e ocupação;
- Transparência e acesso controlado por parte das instituições;
- Monitoramento contínuo de desempenho e segurança.

Com esta etapa, completámos o ciclo de desenvolvimento do projeto, disponibilizando uma solução funcional que alia tecnologia, sustentabilidade e governança digital ao serviço do turismo angolano.

Acreditamos que este sistema representa um passo concreto rumo a um turismo inteligente, sustentável e baseado em evidências, fortalecendo a capacidade institucional e estratégica do país.