Análise Preditiva de Desmatamento: Metodologia SEMMA Completa

Autor: Pedro Mambelli Fernandes
Projeto: Desafio Zetta Labs 2025

Este notebook demonstra a aplicação completa da metodologia SEMMA (Sample, Explore, Modify, Model, Assess) no projeto de análise preditiva de desmatamento. Cada etapa utiliza as funções e classes desenvolvidas nos scripts do projeto, proporcionando uma execução reprodutível e didática de todo o pipeline de machine learning.

Metodologia SEMMA

- **Sample (Amostragem):** Carregamento e resumo inicial dos dados
- **Explore (Exploração):** Análise exploratória e identificação de padrões
- **Modify (Modificação):** Pré-processamento e engenharia de features
- **Model (Modelagem):** Treinamento dos modelos preditivos
- **Assess (Avaliação):** Análise de performance e interpretação

1. Configuração do Ambiente

Primeiro, configuramos o ambiente de trabalho. Adicionamos o diretório raiz do projeto ao `sys.path` para permitir a importação dos módulos customizados localizados em `/scripts`. Em seguida, importamos as bibliotecas e os scripts necessários.

```
import os
import sys
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings('ignore')
```

```python
# Adiciona a raiz do projeto ao path para importar os scripts
module_path = os.path.abspath(os.path.join('..'))
if module_path not in sys.path:
    sys.path.append(module_path)

# Importando os scripts do projeto (metodologia SEMMA)
from scripts import sampling  # S - Sample
from scripts import explore   # E - Explore
from scripts import modify    # M - Modify
from scripts import model     # M - Model
from scripts import assess    # A - Assess

print("  Ambiente configurado com sucesso!")
print("  Módulos SEMMA importados:")
print("    sampling - Amostragem e carregamento de dados")
print("    explore - Análise exploratória")
print("    modify - Pré-processamento e feature engineering")
print("    model - Treinamento de modelos")
print("    assess - Avaliação e análise de resultados")
```

```
 Ambiente configurado com sucesso!
 Módulos SEMMA importados:
    sampling - Amostragem e carregamento de dados
    explore - Análise exploratória
    modify - Pré-processamento e feature engineering
    model - Treinamento de modelos
    assess - Avaliação e análise de resultados
```

Estrutura dos Scripts SEMMA

Cada script implementa uma etapa específica da metodologia:

- **`sampling.py`** → Carregamento inicial e geração de resumos estatísticos
- **`explore.py`** → Análise de qualidade e visualizações exploratórias

- **`modify.py`** → Limpeza, agregações e engenharia de features
- **`model.py`** → Treinamento temporal de modelos (XGBoost, Random Forest)
- **`assess.py`** → Avaliação comparativa e geração de insights

O notebook organiza cada etapa em sequência, utilizando as funções e classes desenvolvidas nos scripts para demonstrar o processo de forma clara.

2. SAMPLE - Amostragem e Carregamento dos Dados

A primeira etapa do SEMMA consiste em carregar os dados e gerar resumos iniciais para entender a estrutura de cada dataset. O script `sampling.py` automatiza esse processo para os três principais conjuntos de dados:

- **Desmatamento Anual:** Dados históricos de desmatamento por município (formato Parquet)
- **Infrações do IBAMA:** Autos de infração ambientais de 1977-2025 (múltiplos CSVs)
- **Índice de Progresso Social (IPS):** Indicadores socioeconômicos dos municípios do Pará

O script gera relatórios detalhados com estatísticas descritivas, informações sobre tipos de dados e identificação de valores ausentes.

```
print("  Executando etapa SAMPLE - Amostragem...")
print("=" * 50)

# Executar o script de sampling completo
sampling.main()

print("\n  Etapa SAMPLE concluída!")
print("  Relatórios gerados em: ../reports/")
print("    - summary_ips_municipios.txt")
print("    - summary_desmatamento_anual.txt")
print("    - summary_ibama_infracoes_consolidadas.txt")
```

```
  Executando etapa SAMPLE - Amostragem...
==================================================
Iniciando a etapa de Sampling (Amostragem)...
Relatório de resumo salvo em: C:\Users\pedro\Documents\Dev\zetta-labs-2\reports\summary_ips_
Relatório de resumo salvo em: C:\Users\pedro\Documents\Dev\zetta-labs-2\reports\summary_desma
Relatório de resumo salvo em: C:\Users\pedro\Documents\Dev\zetta-labs-2\reports\summary_ibama

Etapa de Sampling concluída.
Verifique os arquivos de resumo no diretório: reports

  Etapa SAMPLE concluída!
  Relatórios gerados em: ../reports/
    - summary_ips_municipios.txt
    - summary_desmatamento_anual.txt
    - summary_ibama_infracoes_consolidadas.txt
```

3. EXPLORE - Análise Exploratória de Dados

A segunda etapa do SEMMA foca na exploração visual e estatística dos dados. O script `explore.py` realiza:

- **Filtragem geográfica:** Foco no estado do Pará (UF = 'PA')
- **Análise de qualidade:** Identificação de valores ausentes por fonte de dados
- **Visualizações:** Gráficos de barras mostrando a porcentagem de missing values
- **Validação temporal:** Verificação da cobertura temporal dos dados (2008+)

Esta etapa é crucial para identificar problemas de qualidade antes do pré-processamento.

```
print("  Executando etapa EXPLORE - Análise Exploratória...")
print("=" * 50)

# Executar o script de exploração completo
explore.main()

print("\n  Etapa EXPLORE concluída!")
print("  Gráficos de qualidade dos dados gerados em: ../reports/figures/")
print("    - missing_values_ips_pa.png")
print("    - missing_values_desmatamento_pa.png")
print("    - missing_values_ibama_pa.png")
```

```
  Executando etapa EXPLORE - Análise Exploratória...
==================================================
Iniciando a etapa de Explore (Exploração)...

[IPS] Colunas disponíveis: ['Código IBGE', 'Município', 'UF', 'Área (km²)', 'População 2022'

[IPS] Registros totais: 808. Registros para o Pará: 144
[ips_pa] Nenhum valor ausente encontrado.

[Desmatamento] Registros totais: 7304. Registros para o Pará: 2069
[desmatamento_pa] Nenhum valor ausente encontrado.

[IBAMA] Registros totais: 689189. Registros para o Pará: 67142
[ibama_pa] Gráfico de valores ausentes salvo em: C:\Users\pedro\Documents\Dev\zetta-labs-2\re

Etapa de Explore concluída.
Verifique os gráficos no diretório: C:\Users\pedro\Documents\Dev\zetta-labs-2\reports\figures

  Etapa EXPLORE concluída!
```

```
Gráficos de qualidade dos dados gerados em: ../reports/figures/
  - missing_values_ips_pa.png
  - missing_values_desmatamento_pa.png
  - missing_values_ibama_pa.png
```

4. MODIFY - Pré-processamento e Engenharia de Features

A terceira etapa do SEMMA é a mais crítica para o sucesso dos modelos. O script `modify.py` realiza:

Processamento dos Dados:

- **Carregamento inteligente:** IPS (CSV), Desmatamento (Parquet), IBAMA (múltiplos CSVs)
- **Filtragem temporal:** Dados de 2008 em diante para garantir consistência
- **Filtragem geográfica:** Foco no estado do Pará
- **Limpeza:** Padronização de códigos IBGE, tratamento de valores ausentes

Agregações Estratégicas:

- **IBAMA por município/ano:** Total de autos, valor de multas, infrações de flora
- **Interpolação de IPS:** Preenchimento temporal usando interpolação linear

Features Temporais:

- **Lags:** Desmatamento do ano anterior (`desmatamento_lag1`)
- **Médias móveis:** Tendência de 3 anos (`desmatamento_ma3`)
- **Períodos presidenciais:** Lula, Dilma, Temer, Bolsonaro (codificado)
- **Anos de transição:** Marcadores para mudanças de governo

```
print("  Executando etapa MODIFY - Pré-processamento...")
print("=" * 50)

# Executar o script de modificação completo
modify.main()

print("\n  Etapa MODIFY concluída!")
print("  Tabelas analíticas geradas em: ../reports/processed/")
print("   - analytical_base_table.parquet: Tabela completa com todas as features")
print("   - analytical_table_tabular.parquet: Para modelos tabulares (RF, XGBoost)")
```

```
print("   - analytical_table_temporal.parquet: Para modelos temporais (LSTM, GRU)")
print("   - data_summary.txt: Resumo dos dados processados")
```

```
 Executando etapa MODIFY - Pré-processamento...
==================================================
Iniciando a etapa de Modify (Modificação)...
Período de análise: 2008 em diante
Estado: PA
--------------------------------------------------
1. Carregando dados...
IPS: 144 municípios do Pará carregados
Desmatamento: 2069 registros do Pará de 2008 em diante
IBAMA: 37544 registros do Pará de 2008 em diante

2. Processando dados do IBAMA...
IBAMA agregado: 1609 registros município-ano

3. Criando tabela analítica base...
Após interpolação do IPS: 2069 registros
Após merge com IBAMA: 2069 registros

4. Adicionando features temporais...

5. Preparando encodings para ML...

6. Salvando tabelas analíticas...
Tabela analítica base salva: C:\Users\pedro\Documents\Dev\zetta-labs-2\reports\processed\anal
Tabela para modelos tabulares salva: C:\Users\pedro\Documents\Dev\zetta-labs-2\reports\proces
Tabela para modelos temporais salva: C:\Users\pedro\Documents\Dev\zetta-labs-2\reports\proces
Resumo salvo: C:\Users\pedro\Documents\Dev\zetta-labs-2\reports\processed\data_summary.txt

Etapa de Modify concluída!
Verifique os arquivos processados em: C:\Users\pedro\Documents\Dev\zetta-labs-2\reports\proce

Arquivos gerados:
- analytical_base_table.parquet: Tabela completa com todas as features
- analytical_table_tabular.parquet: Para modelos tabulares (RF, XGBoost)
- analytical_table_temporal.parquet: Para modelos temporais (LSTM, GRU)
- data_summary.txt: Resumo dos dados processados

 Etapa MODIFY concluída!
 Tabelas analíticas geradas em: ../reports/processed/
```

```
- analytical_base_table.parquet: Tabela completa com todas as features
- analytical_table_tabular.parquet: Para modelos tabulares (RF, XGBoost)
- analytical_table_temporal.parquet: Para modelos temporais (LSTM, GRU)
- data_summary.txt: Resumo dos dados processados
```

5. MODEL - Treinamento dos Modelos Preditivos

A quarta etapa do SEMMA envolve o treinamento de modelos de machine learning. A estratégia implementada é de **modelagem temporal por períodos**, onde cada modelo é treinado com dados históricos para prever o próximo ano.

Estratégia de Modelagem:

- **Períodos de treinamento:** 2013-2023 (11 modelos, um para cada ano)
- **Modelos implementados:** XGBoost e Random Forest
- **Validação:** Walk-forward temporal (sem vazamento de dados futuros)
- **Arquitetura modular:** Classes especializadas para cada algoritmo

Pipeline de Treinamento:

1. **Carregamento:** Tabela analítica processada
2. **Preparação:** Seleção de features e definição do target
3. **Treinamento:** Ajuste bayesiano dos hiperparâmetros com Optuna
4. **Predição:** Geração de predições para o ano de teste
5. **Persistência:** Salvar modelos, metadados e predições

Outputs Gerados:

- **Modelos treinados:** Arquivos `.pkl` para cada período/algoritmo
- **Metadados:** Configurações e feature importance (JSON)
- **Predições:** Valores reais vs. previstos (Parquet)
- **Métricas:** R^2, RMSE, MAE por período

```python
print("  Executando etapa MODEL - Treinamento dos Modelos...")
print("=" * 50)

# Executar o pipeline de modelagem completo
model.main()

print("\n  Etapa MODEL concluída!")
```

```
print("  Resultados de modelagem gerados em:")
print("     ../reports/models/ - Modelos treinados por período")
print("     ../reports/results/ - Métricas de performance")
print("     ../reports/checkpoints/ - Consolidação de todos os resultados")
print("\n  Para cada período (2013-2023):")
print("   - {modelo}_model.pkl: Modelo treinado")
print("   - {modelo}_metadata.json: Configurações e feature importance")
print("   - {modelo}_predictions.parquet: Predições vs. valores reais")
```

```
  Executando etapa MODEL - Treinamento dos Modelos...
==================================================
  Executando pipeline de modelagem refatorado...
  Estrutura modular:
      models/base.py - Funcionalidades comuns
      models/xgboost_model.py - XGBoost
      models/random_forest_model.py - Random Forest
      models/trainer.py - Orquestrador principal
      scripts/assess.py - Análise de resultados

Iniciando pipeline de modelagem modular...
ID da execução: 20250630_181342
============================================================
0. Verificando execuções anteriores...
Checkpoint carregado: C:\Users\pedro\Documents\Dev\zetta-labs-2\reports\checkpoints\all_resul
Carregados 22 resultados existentes
Encontrados 22 resultados de execuções anteriores
Experimentos já completados: 22

1. Carregando dados...
Dados tabulares: (2069, 21)
Dados temporais: (2069, 23)

2. Criando splits walk-forward...
Criados 11 splits walk-forward

3. Treinando modelos tabulares (11 períodos)...

Período 1/11: Treino até 2012.0, Teste 2013.0
  xgboost já treinado para este período - pulando
  random_forest já treinado para este período - pulando

Período 2/11: Treino até 2013.0, Teste 2014.0
```

```
  xgboost já treinado para este período - pulando
  random_forest já treinado para este período - pulando

Período 3/11: Treino até 2014.0, Teste 2015.0
  xgboost já treinado para este período - pulando
  random_forest já treinado para este período - pulando

Período 4/11: Treino até 2015.0, Teste 2016.0
  xgboost já treinado para este período - pulando
  random_forest já treinado para este período - pulando

Período 5/11: Treino até 2016.0, Teste 2017.0
  xgboost já treinado para este período - pulando
  random_forest já treinado para este período - pulando

Período 6/11: Treino até 2017.0, Teste 2018.0
  xgboost já treinado para este período - pulando
  random_forest já treinado para este período - pulando

Período 7/11: Treino até 2018.0, Teste 2019.0
  xgboost já treinado para este período - pulando
  random_forest já treinado para este período - pulando

Período 8/11: Treino até 2019.0, Teste 2020.0
  xgboost já treinado para este período - pulando
  random_forest já treinado para este período - pulando

Período 9/11: Treino até 2020.0, Teste 2021.0
  xgboost já treinado para este período - pulando
  random_forest já treinado para este período - pulando

Período 10/11: Treino até 2021.0, Teste 2022.0
  xgboost já treinado para este período - pulando
  random_forest já treinado para este período - pulando

Período 11/11: Treino até 2022.0, Teste 2023.0
  xgboost já treinado para este período - pulando
  random_forest já treinado para este período - pulando

5. Salvando resultados finais...

============================================================
RESUMO DOS RESULTADOS
```

```
============================================================
                  rmse              mae              r2
                  mean      std    mean     std    mean     std
model_type
random_forest  22.2485  12.9824  6.4851  2.5099  0.8607  0.0858
xgboost        20.2573   8.3364  6.5210  1.9521  0.8771  0.0758

Total de experimentos completados: 22
Modelos únicos treinados: 2
Períodos cobertos: 11

Melhor RMSE: random_forest (período 3) : 9.8141
Melhor R²: random_forest (período 10) : 0.9859

Resultados salvos em: C:\Users\pedro\Documents\Dev\zetta-labs-2\reports\results\model_results
Pipeline de modelagem concluído!

 Etapa MODEL concluída!
 Resultados de modelagem gerados em:
    ../reports/models/ - Modelos treinados por período
    ../reports/results/ - Métricas de performance
    ../reports/checkpoints/ - Consolidação de todos os resultados

 Para cada período (2013-2023):
  - {modelo}_model.pkl: Modelo treinado
  - {modelo}_metadata.json: Configurações e feature importance
  - {modelo}_predictions.parquet: Predições vs. valores reais
```

6. ASSESS - Avaliação e Análise dos Resultados

A quinta e última etapa do SEMMA é focada na avaliação objetiva dos modelos e geração de insights acionáveis. O script `assess.py` realiza uma análise abrangente da performance.

Análises Realizadas:

1. Comparação Temporal de Performance

- **Gráfico:** Evolução do R^2 e RMSE ao longo dos anos de teste
- **Insight:** Identificação de períodos onde os modelos têm melhor/pior performance
- **Output:** `model_comparison_timeline.png`

2. Importância das Features

- **Gráfico:** Top 10 features mais importantes para cada modelo
- **Análise:** Comparação entre XGBoost e Random Forest
- **Output:** `feature_importance_comparison.png`

3. Evolução Temporal: Real vs. Previsto

- **Gráfico:** Séries temporais com destaque para mudanças políticas
- **Contexto:** Períodos de suspensão do PPCDAm (2019-2022)
- **Output:** `temporal_evolution_real_vs_predicted.png`

4. Análise de Anos Críticos

- **Foco:** Performance específica em 2019 (início Bolsonaro) e 2023 (retorno Lula)
- **Insight:** Como os modelos se comportam durante mudanças políticas
- **Output:** `critical_years_analysis.png`

Dados Estruturados para Dashboard:

Todos os gráficos geram também tabelas em formato Parquet para alimentar o dashboard interativo.

```
print("  Executando etapa ASSESS - Avaliação dos Modelos...")
print("=" * 50)

# Executar a análise completa de avaliação
assess.main()

print("\n  Etapa ASSESS concluída!")
print("  Análises geradas em: ../reports/figures/")
print("     model_comparison_timeline.png: Comparação temporal R² e RMSE")
print("     feature_importance_comparison.png: Top 10 features por modelo")
print("     temporal_evolution_real_vs_predicted.png: Real vs. Previsto com contexto político
print("     critical_years_analysis.png: Performance em anos críticos (2019, 2023)")
print("\n  Dados estruturados em: ../reports/processed/")
print("     model_comparison_data.parquet")
print("     feature_importance_data.parquet")
print("     temporal_evolution_data.parquet")
print("     critical_years_data.parquet")
print("     model_summary.parquet")
```

```
  Executando etapa ASSESS - Avaliação dos Modelos...
==================================================

  ANÁLISE FOCADA DE MODELOS
========================================
  Carregando: all_results_20250629_204235.pkl
  22 resultados | 2,814 predições

  Gerando análises...
  Comparação temporal salva: model_comparison_timeline.png
  random_forest: 11 períodos processados, 18 features
  xgboost: 11 períodos processados, 18 features
  Importância das features salva: feature_importance_comparison.png
  Evolução temporal salva: temporal_evolution_real_vs_predicted.png
  Análise de anos críticos salva: critical_years_analysis.png

  RESUMO DOS MODELOS:
================================================
Random Forest:
  R² médio: 0.861 (±0.086) - Ranking: #2
  RMSE médio: 22.2 (±13.0) - Ranking: #2

Xgboost:
  R² médio: 0.877 (±0.076) - Ranking: #1
  RMSE médio: 20.3 (±8.3) - Ranking: #1


  Análise concluída!
  Gráficos: C:\Users\pedro\Documents\Dev\zetta-labs-2\reports\figures
  Dados: C:\Users\pedro\Documents\Dev\zetta-labs-2\reports\processed

  Etapa ASSESS concluída!
  Análises geradas em: ../reports/figures/
     model_comparison_timeline.png: Comparação temporal R² e RMSE
     feature_importance_comparison.png: Top 10 features por modelo
     temporal_evolution_real_vs_predicted.png: Real vs. Previsto com contexto político
     critical_years_analysis.png: Performance em anos críticos (2019, 2023)

  Dados estruturados em: ../reports/processed/
     model_comparison_data.parquet
     feature_importance_data.parquet
     temporal_evolution_data.parquet
     critical_years_data.parquet
```

```
model_summary.parquet
```

7. Visualização dos Resultados

Após a execução completa do pipeline, você pode examinar os resultados gerados. Aqui estão alguns exemplos de como acessar e interpretar os dados processados.

```
# Carregamento e exibição de resultados exemplo
import os
from pathlib import Path

REPORTS_DIR = Path("../reports")
PROCESSED_DIR = REPORTS_DIR / "processed"
FIGURES_DIR = REPORTS_DIR / "figures"

print(" RESULTADOS DISPONÍVEIS:")
print("=" * 40)

# Verificar se os diretórios existem
if PROCESSED_DIR.exists():
    print(" Dados processados:")
    for file in sorted(PROCESSED_DIR.glob("*.parquet")):
        size_mb = file.stat().st_size / (1024 * 1024)
        print(f"    {file.name} ({size_mb:.1f} MB)")

    print("\n Relatórios:")
    for file in sorted(PROCESSED_DIR.glob("*.txt")):
        print(f"    {file.name}")

if FIGURES_DIR.exists():
    print("\n Visualizações:")
    for file in sorted(FIGURES_DIR.glob("*.png")):
        size_kb = file.stat().st_size / 1024
        print(f"     {file.name} ({size_kb:.0f} KB)")

# Exemplo: Carregar resumo dos modelos se disponível
if (PROCESSED_DIR / "model_summary.parquet").exists():
    print("\n RESUMO DOS MODELOS:")
    print("-" * 30)
    model_summary = pd.read_parquet(PROCESSED_DIR / "model_summary.parquet")
    for _, row in model_summary.iterrows():
        model_name = row['model_type'].replace('_', ' ').title()
```

```
        print(f"{model_name}:")
        print(f"    R² médio: {row['r2_mean']:.3f} (ranking: #{row['r2_rank']})")
        print(f"    RMSE médio: {row['rmse_mean']:.1f} km² (ranking: #{row['rmse_rank']})")
        print()
else:
    print("\n  Execute primeiro o pipeline completo para ver os resultados.")
```

```
  RESULTADOS DISPONÍVEIS:
========================================
  Dados processados:
     analytical_base_table.parquet (0.1 MB)
     analytical_table_tabular.parquet (0.1 MB)
     analytical_table_temporal.parquet (0.1 MB)
     critical_years_data.parquet (0.0 MB)
     executive_summary.parquet (0.0 MB)
     feature_importance_analysis.parquet (0.0 MB)
     feature_importance_data.parquet (0.0 MB)
     model_comparison_data.parquet (0.0 MB)
     model_performance_analysis.parquet (0.0 MB)
     model_summary.parquet (0.0 MB)
     performance_statistics.parquet (0.0 MB)
     prediction_quality_metrics.parquet (0.0 MB)
     predictions_analysis_sample.parquet (0.1 MB)
     temporal_evolution_data.parquet (0.0 MB)

  Relatórios:
     data_summary.txt

  Visualizações:
      critical_years_analysis.png (118 KB)
      feature_importance_comparison.png (250 KB)
      missing_values_ibama_pa.png (102 KB)
      model_comparison_timeline.png (284 KB)
      temporal_evolution_real_vs_predicted.png (556 KB)

  RESUMO DOS MODELOS:
------------------------------
Random Forest:
    R² médio: 0.861 (ranking: #2)
    RMSE médio: 22.2 km² (ranking: #2)

Xgboost:
```

```
R² médio: 0.877 (ranking: #1)
RMSE médio: 20.3 km² (ranking: #1)
```

Conclusão

Este notebook apresenta a execução da metodologia SEMMA para análise preditiva de desmatamento, com as etapas implementadas em scripts separados (`sampling.py`, `explore.py`, `modify.py`, `model.py`, `assess.py`).

Pontos da abordagem:

- **Modularização:** Cada etapa pode ser executada isoladamente.
- **Validação temporal:** Utiliza walk-forward para evitar vazamento de dados.
- **Comparação de modelos:** XGBoost e Random Forest avaliados com as mesmas métricas.
- **Análise de variáveis:** Feature importance e gráficos para interpretação.

Próximos passos:

1. Para visualizar os principais resultados, execute `streamlit run ../dashboard.py`.
2. Os gráficos estão em `../reports/figures/`.
3. Os dados processados estão em `../reports/processed/` (formato Parquet).

Para rodar o pipeline novamente, basta executar as células do notebook em ordem.