Objetivo do Pré-Processamento de Dados

Antes de aplicar qualquer algoritmo de machine learning, é fundamental preparar os dados. O pré-processamento garante que:

- Os dados estejam limpos e completos
- As variáveis estejam em formatos adequados
- As escalas sejam padronizadas
- O modelo possa lidar corretamente com variáveis categóricas

---

1. Leitura e visualização inicial dos dados

```
import pandas as pd
import numpy as np

# Leitura da base de crédito
baseCredito = pd.read_csv('credit_data.csv')

# Visualização das 5 primeiras linhas
print(baseCredito.head())

# Informações gerais do DataFrame (tipos, nulos etc.)
print(baseCredito.info())

# Estatísticas descritivas básicas
print(baseCredito.describe())
```

2. Identificação de dados inconsistentes

Exemplo clássico: valores de idade negativos (não fazem sentido).

```
print(baseCredito[baseCredito['age'] < 0])
```

Correção por substituição com a média:

```
mediaIdade = baseCredito['age'][baseCredito['age'] >
0].mean()
baseCredito.loc[baseCredito['age'] < 0, 'age'] =
mediaIdade
```

3. Tratamento de dados faltantes (nulos)

Verificação de valores nulos:

```
print(baseCredito.isnull().sum())
```

Ver onde estão os nulos na coluna age, por exemplo:

```
print(baseCredito.loc[pd.isnull(baseCredito['age'])])
```

Preenchimento com a média:

```
baseCredito['age'] =
baseCredito['age'].fillna(mediaIdade)
```

Verificação final:

```
print(baseCredito.isnull().sum())
```

4. Separação entre variáveis preditoras (X) e classe (Y)

```
# Pega colunas 1, 2 e 3 como preditoras (income, age,
loan)
X_baseCredito = baseCredito.iloc[:, 1:4].values

# Pega a coluna 4 como classe (default)
Y_baseCredito = baseCredito.iloc[:, 4].values
```

5. Normalização dos dados

Escalonamento entre 0 e 1 com MinMaxScaler:

```
from sklearn.preprocessing import MinMaxScaler

normalizador = MinMaxScaler()
X_baseCredito_Normalizado =
normalizador.fit_transform(X_baseCredito)
```

6. Separação em dados de treino e teste

```
from sklearn.model_selection import train_test_split

X_baseCredito_treino, X_baseCredito_teste,
Y_baseCredito_treino, Y_baseCredito_teste =
train_test_split(
    X_baseCredito_Normalizado, Y_baseCredito,
test_size=0.2, random_state=0
)
```

7. Salvamento com pickle (opcional, mas recomendado)

```
import pickle

with open('credito.pkl', mode='wb') as f:
    pickle.dump([
        X_baseCredito_treino,
        Y_baseCredito_treino,
        X_baseCredito_teste,
        Y_baseCredito_teste
    ], f)
```

8. (Opcional) Visualizações com Seaborn/Matplotlib/Plotly

Exemplos para explorar padrões visuais:

```
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px

# Distribuição de inadimplência
sns.countplot(x=baseCredito['default'])
plt.title('Distribuição de Pagamentos')

# Histograma da idade (já tratada)
plt.hist(baseCredito['age'])
plt.title('Distribuição da Idade')

# Scatter Matrix
grafico = px.scatter_matrix(baseCredito,
dimensions=['income', 'loan', 'age'], color='default')
grafico.show()
```

Resumo das etapas seguidas na prática

1. Carregar o dataset
2. Analisar estatísticas (head(), info(), describe())
3. Identificar e corrigir dados inconsistentes
4. Tratar valores nulos
5. Separar variáveis preditoras e alvo
6. Normalizar os dados
7. Dividir em treino e teste
8. Salvar os dados com pickle
9. (Opcional) Visualizar os dados com gráficos