

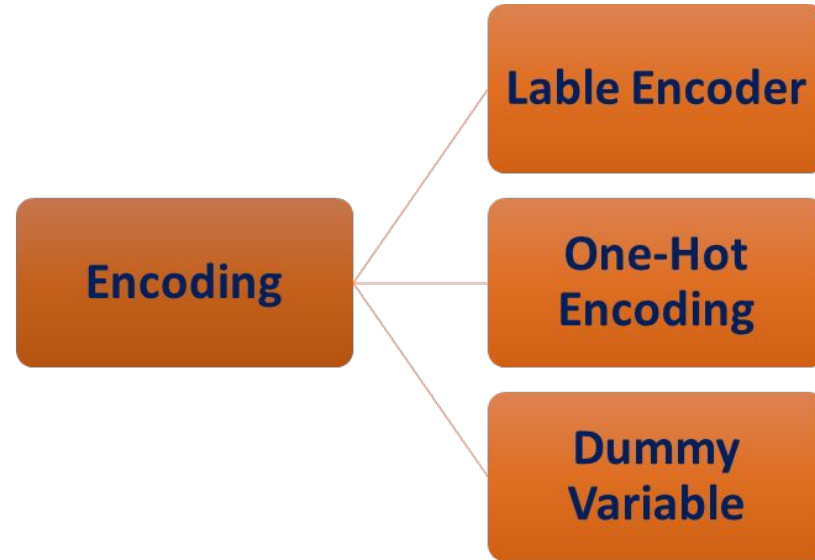
# Pré-processamento

Transformação, amostragem e balanceamento  
dos dados

Prof. Dr. Danilo S. Sanches



# Transformação dos dados



Fonte:

<https://medium.com/@swarnpriyaswarn/encoding-2nd-step-of-pre-processing-data-2d439b26a4fe>

# Transformação dos dados

## Label Encoding

- O Label Encoding é uma abordagem que atribui cada categoria a um número inteiro único;
- Útil quando as categorias não têm qualquer ordem intrínseca;
- Exemplo, colunas apenas com valores “Sim” ou “Não”, o Label Encoding atribuiria 1 “Sim” e 0 “Não”.

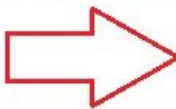
# Transformação dos dados

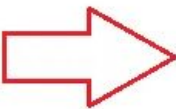
## One-Hot Encoding

- Se uma coluna categórica tiver  $k$  categorias exclusivas, a aplicação One-Hot Encoding criará novas  $k$  colunas (características).

# OneHotEncoder (Scikit-learn) vs get-dummies (Pandas)

## pd.get\_dummies()

| train_df: |          |       |        |   |       |                |             | train_features: |   |       |                |             |              |
|-----------|----------|-------|--------|---|-------|----------------|-------------|-----------------|---|-------|----------------|-------------|--------------|
|           | Call_Me? | Money | Target |   | Money | Call_Me?_Maybe | Call_Me?_No | Call_Me?_Yes    |   | Money | Call_Me?_Maybe | Call_Me?_No | Call_Me?_Yes |
| 0         | Yes      | 5     | 10     | Data Preprocessing<br> | 0     | 5              | 0           | 0               | 1 | 0     | 5              | 0           | 1            |
| 1         | No       | 3     | 4      |   | 1     | 3              | 0           | 1               | 0 | 1     | 3              | 0           | 0            |
| 2         | Maybe    | 5     | 5      |   | 2     | 5              | 1           | 0               | 0 | 2     | 5              | 1           | 0            |
| 3         | Yes      | 10    | 7      |   | 3     | 10             | 0           | 0               | 1 | 3     | 10             | 0           | 1            |
| 4         | Yes      | 9     | 9      |   | 4     | 9              | 0           | 0               | 1 | 4     | 9              | 0           | 1            |

| test_df: |          |       |   |   |       |             |              | test_features: |       |             |              |
|----------|----------|-------|---|---|-------|-------------|--------------|----------------|-------|-------------|--------------|
|          | Call_Me? | Money |   |   | Money | Call_Me?_No | Call_Me?_Yes |                | Money | Call_Me?_No | Call_Me?_Yes |
| 0        | Yes      | 5     | Data Preprocessing<br> | 0 | 5     | 0           | 1            | 0              | 5     | 0           | 1            |
| 1        | Yes      | 2     |   | 1 | 2     | 0           | 1            | 1              | 2     | 0           | 1            |
| 2        | Yes      | 3     |   | 2 | 3     | 0           | 1            | 2              | 3     | 0           | 1            |
| 3        | Yes      | 6     |   | 3 | 6     | 0           | 1            | 3              | 6     | 0           | 1            |
| 4        | No       | 1     |   | 4 | 1     | 1           | 0            | 4              | 1     | 1           | 0            |

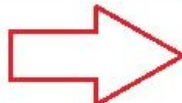
# OneHotEncoder (Scikit-learn) vs get-dummies (Pandas)

## OneHotEncoder()

train\_df:

|   | Call_Me? | Money | Target |
|---|----------|-------|--------|
| 0 | Yes      | 5     | 10     |
| 1 | No       | 3     | 4      |
| 2 | Maybe    | 5     | 5      |
| 3 | Yes      | 10    | 7      |
| 4 | Yes      | 9     | 9      |

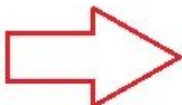
Data Preprocessing



test\_df:

|   | Call_Me? | Money |
|---|----------|-------|
| 0 | Yes      | 5     |
| 1 | Yes      | 2     |
| 2 | Yes      | 3     |
| 3 | Yes      | 6     |
| 4 | No       | 1     |

Data Preprocessing



train\_features:

|   | Money | Call_Me?_Maybe | Call_Me?_No | Call_Me?_Yes |
|---|-------|----------------|-------------|--------------|
| 0 | 5     | 0.0            | 0.0         | 1.0          |
| 1 | 3     | 0.0            | 1.0         | 0.0          |
| 2 | 5     | 1.0            | 0.0         | 0.0          |
| 3 | 10    | 0.0            | 0.0         | 1.0          |
| 4 | 9     | 0.0            | 0.0         | 1.0          |

test\_features:

|   | Money | Call_Me?_Maybe | Call_Me?_No | Call_Me?_Yes |
|---|-------|----------------|-------------|--------------|
| 0 | 5     | 0.0            | 0.0         | 1.0          |
| 1 | 2     | 0.0            | 0.0         | 1.0          |
| 2 | 3     | 0.0            | 0.0         | 1.0          |
| 3 | 6     | 0.0            | 0.0         | 1.0          |
| 4 | 1     | 0.0            | 1.0         | 0.0          |

# Técnicas de amostragem

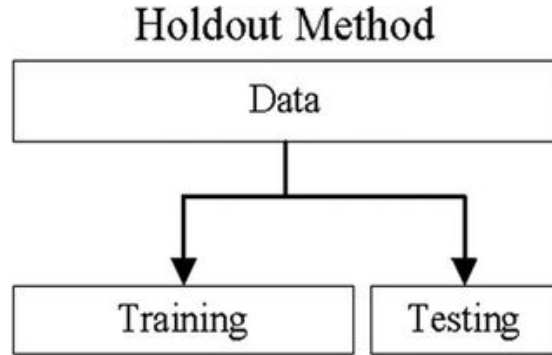
## **Técnica Holdout:**

A divisão do conjunto de dados original em treino e teste (geralmente 70% e 30%). O conjunto treino é utilizado para treinar o modelo e o conjunto teste avalia a capacidade de avaliar dados não vistos.

## **Validação Cruzada:**

Divide o conjunto de dados em várias partes e, em seguida, treina e testa o modelo em diferentes combinações dessas partes.

# Técnicas de amostragem



5 K-fold Validation Method

| Data  |       |       |       |       |        |
|-------|-------|-------|-------|-------|--------|
| Test  | Train | Train | Train | Train | Fold 1 |
| Train | Test  | Train | Train | Train | Fold 2 |
| Train | Train | Test  | Train | Train | Fold 3 |
| Train | Train | Train | Test  | Train | Fold 4 |
| Train | Train | Train | Train | Test  | Fold 5 |

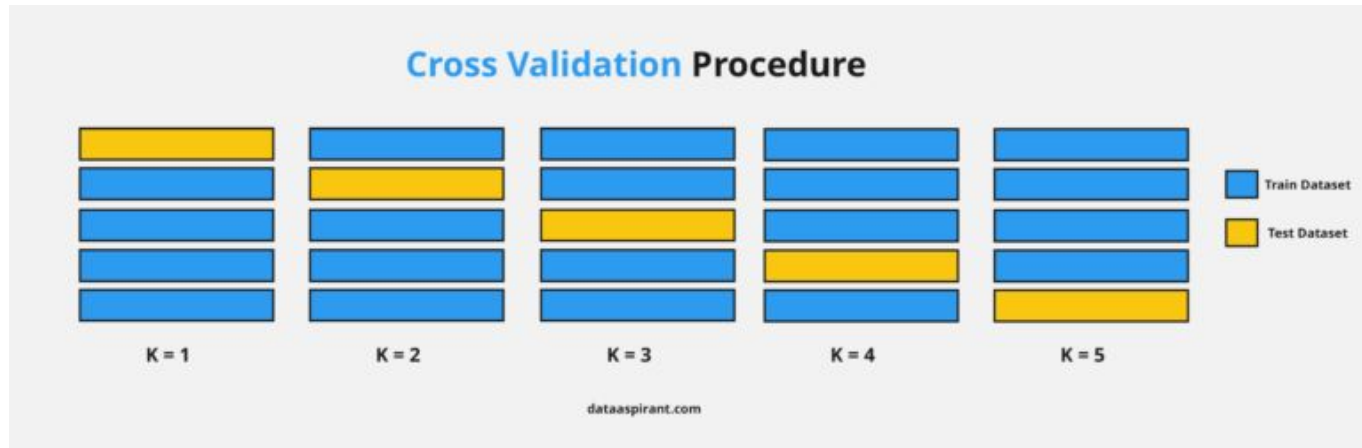
Aladwani, Faisal & Elsharkawy, Adel. Improved prediction of heavy oil viscosity at various conditions utilizing various supervised machine learning regression. ***Petroleum Science and Technology***. (2022)



# Tipos de Validação Cruzada

## K-Fold Cross-Validation:

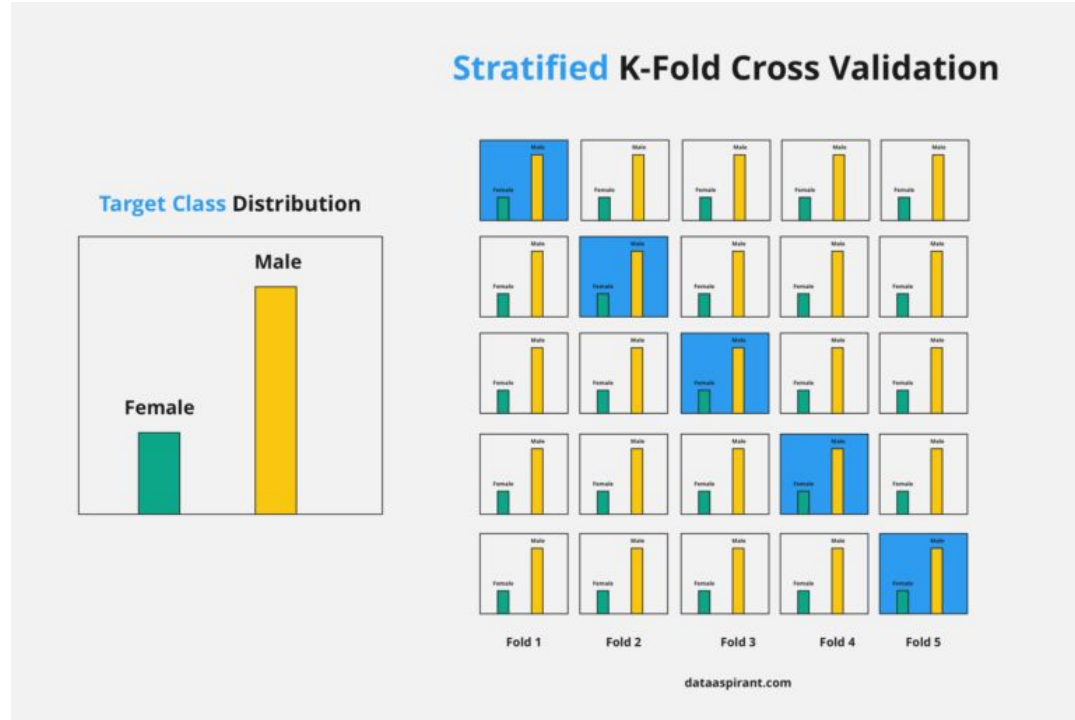
O conjunto de dados é dividido em **k** partes iguais (**folds**). O modelo é treinado em k-1 folds e testado no fold restante. Esse processo é repetido **k** vezes.



# Tipos de Validação Cruzada

## Stratified K-Fold Cross-Validation:

Variação da K-Fold, garante que a distribuição das classes seja preservada em cada fold. Útil em conjuntos de dados desbalanceados.

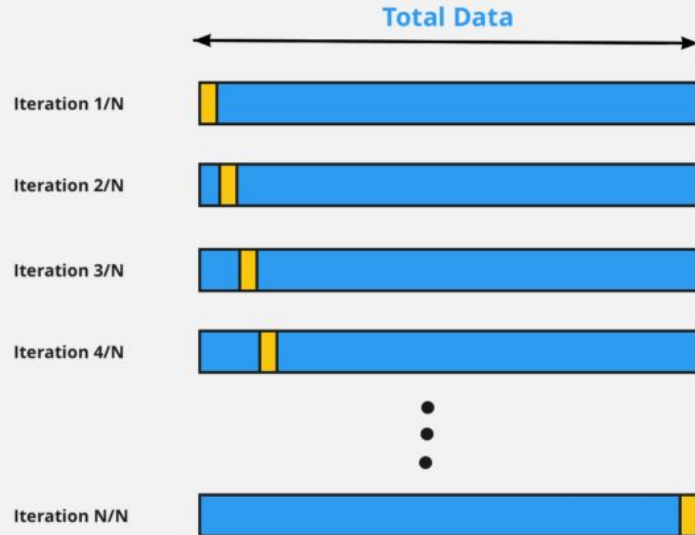


# Tipos de Validação Cruzada

**Leave-One-Out Cross-Validation (LOOCV):** Cada amostra é usada como conjunto de teste exatamente uma vez, enquanto as demais amostras são usadas para treinamento. Útil para pequeno conjunto de dados.

<https://dataaspirant.com/cross-validation/>

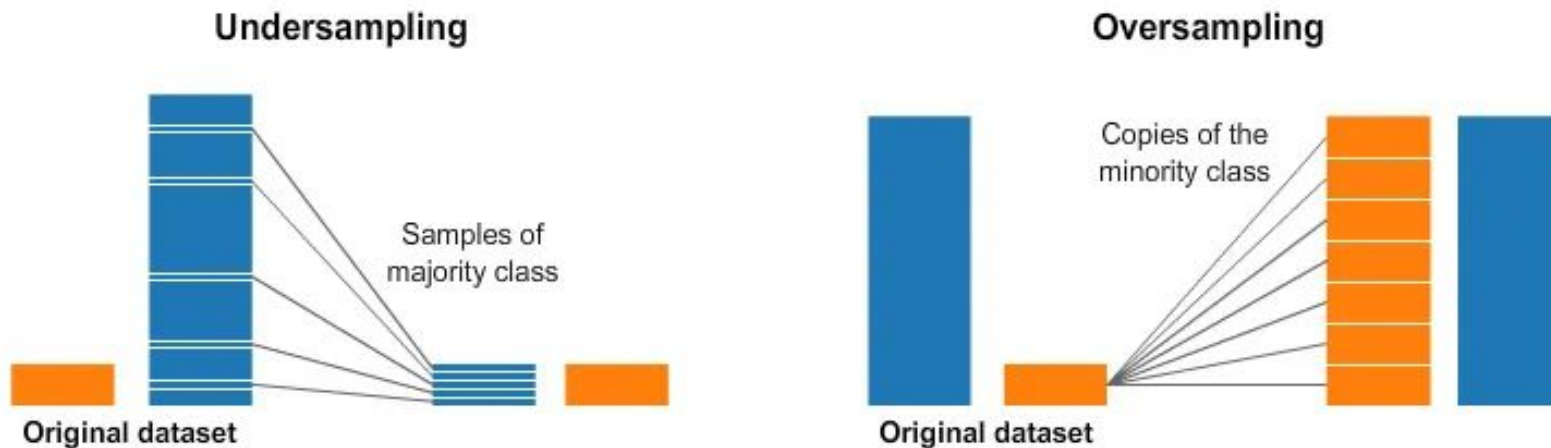
## LOOCV: Leave One Out Cross Validation



dataaspirant.com

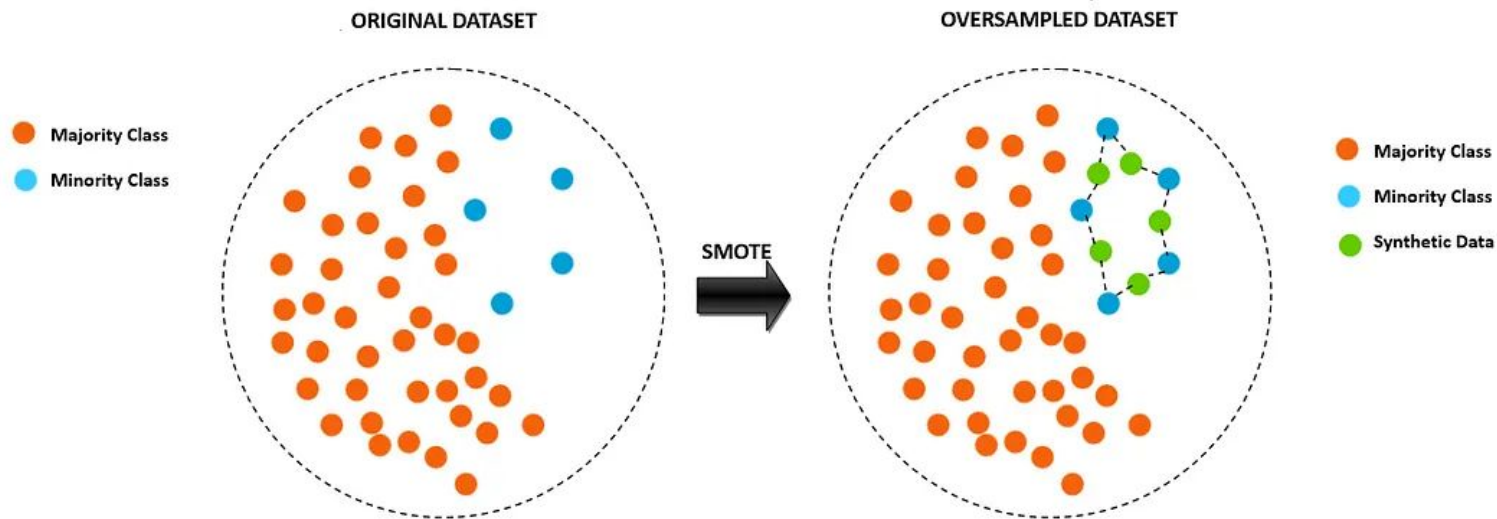
# Dados Desbalanceados

- **Sub-amostragem** (*Undersampling*): exclui exemplos da classe majoritária
- **Super-amostragem** (*Oversampling*) sintetiza novos exemplos na classe minoritária



Fonte: <https://www.kaggle.com/rafjaa/resampling-strategies-for-imbalanced-datasets>

# Dados Desbalanceados



# Dados Desbalanceados



```
from imblearn.over_sampling import SMOTE

counter = Counter(y_train)
print('Before', counter)
# oversampling the train dataset using SMOTE
smt = SMOTE()
#X_train, y_train = smt.fit_resample(X_train, y_train)
X_train_sm, y_train_sm = smt.fit_resample(X_train, y_train)

counter = Counter(y_train_sm)
print('After', counter)
```

---

Before Counter({0: 18497, 1: 4208})  
After Counter({0: 18497, 1: 18497})