1 - Na Gestão da Hierarquia de Geração dos Ancestrais
Meu código usa uma abordagem mais robusta baseada na estrutura de pontos, enquanto o código do seu aluno usa uma busca por substring que podem gerar erros.
O código do seu aluno usa uma lógica incorreta baseada na busca de substrings (`c2 in c1`) em vez de analisar corretamente a estrutura hierárquica com base nos pontos de separação.
Ele usa a presença de substrings para detectar ancestrais, o que pode gerar erros se os nomes não forem estritamente hierárquicos.

Geração de descendentes
O código do seu aluno: Lógica semelhante à dos ancestrais (busca por substring)
Meu código: Usa `c2.startswith(c1 + '.')` para uma verificação mais precisa

❌ **O código do seu aluno está EM DESACORDO com o artigo de Silla**
Pontos de não conformidade:
❌ Fórmula de utilidade incorreta e simplificada
❌ Lógica de geração de ancestrais falha
❌ Propagação incorreta das probabilidades a priori
❌ Ausência de propagação explícita das verossimilhanças
❌ Estrutura geral não segue as especificações de Silla

O código do seu aluno ❌ NÃO CONFORME
❌ Propagação incorreta – lógica com falhas

```python
# Soma a contagem da classe nos seus ancestrais e nela mesma
for classe in self.ancestrais[c]:
    self.prior_prob[classe] += n_instancias_classe_c +
self.alpha
```

Problema: Adiciona diretamente as contagens sem normalização apropriada

❌ Cálculo de utilidade simplificado e incorreto

```python
def calculate_usefullness(self):
    max_tree_size = max([len(value) for key, value in
self.descendentes.items()])
    usefullness = []
    for c in self.classes:
        tree_size_i = len(self.descendentes[c])
```

```
        usefullness_i = 1 - (np.log2(tree_size_i) /
max_tree_size)
        usefullness.append(usefullness_i)
    return usefullness
```

Problemas:

- Não segue a fórmula de Silla:
  `1 - (a(ci) × log2(treesize(ci))) / max`
- Falta o fator `a(ci)`
- `max_tree_size` não é `max(a(ci) × log2(treesize(ci)))`
- `tree_size` deveria ser `1 + len(descendentes)`, não apenas `len(descendentes)`

❌ Geração de ancestrais incorreta

Python
```
def gera_ancestrais(classes):
    ancestrais = {}
    for c1 in classes:
        for c2 in classes:
            if c2 in c1 and c1 not in ancestrais:  #
INCORRETO!
                ancestrais[c1] = [c2]
            elif c2 in c1 and c1 in ancestrais:
                ancestrais[c1].append(c2)
    return ancestrais
```

Problema grave: Usa `c2 in c1` (teste de substring) em vez de analisar a estrutura hierárquica corretamente

❌ Nenhuma propagação explícita das verossimilhanças
O código do seu aluno não propaga corretamente as contagens dos atributos para os ancestrais, como especificado por Silla.

O código do seu aluno faz a predição em todos os níveis da hierarquia (nós internos, modelo global), enquanto meu código faz predições apenas nas folhas (`self.classes`). É aí que está o erro.

O código do seu aluno usa modelos Naive Bayes padrão do `scikit-learn`, com uma gestão simplificada da hierarquia (baseada na presença de substrings) e uma avaliação hierárquica parcial e menos robusta.