15

Implementando o algoritmo genérico

Até agora o nosso algoritmo soube lidar apenas com valores entre **zeros e uns**, porém, no mundo real, podemos nos deparar com valores diferentes, por exemplo, **sim e não**, **yes ou no** entre outras variações. Precisamos então implementar um algoritmo genérico que consiga lidar com qualquer tipo de valor.

Para isso, utilizaremos o `Counter` do python. Primeiro, precisamos importar o `Counter` do `collections` utilizando o código `from collections import Counter`. Então pedimos para ele contar o `Y` e devolver um objeto com todos os valores do `Y` utilizando a função `itervalues()`, então, pedimos para retornar o maior valor desse objeto utilizando a função `max()` e atribuímos o retorno para a variável `acerto_base`. Substitua o trecho `max(acerto_de_um, acerto_de_zero)`

```
taxa_de_acerto_base = 100.0 * max(acerto_de_um, acerto_de_zero) / len(Y)
```

Para apenas `acerto_base`:

```
acerto_base = max(Counter(Y).itervalues())
acerto_de_um = sum(Y)
acerto_de_zero = len(Y) - acerto_de_um
taxa_de_acerto_base = 100.0 * acerto_base / len(Y)
print("Taxa de acerto base: %f" % taxa_de_acerto_base)
```

Observe que não precisamos mais do trecho:

```
acerto_de_um = sum(Y)
acerto_de_zero = len(Y) - acerto_de_um
```

Apague esse trecho e rode o arquivo `classifica_buscas.py`, veja se está funcionando da mesma forma que antes. Cole o código final e o resultado abaixo:

Responda

INSERIR CÓDIGO		FORMATAÇÃO