Ganho de Informação com Processamento Paralelo

Sergio Polimante Souto

São Paulo, Brasil

Abstract

Ganho de informação (GI) mede o quanto de informação um atributo nos fornece sobre uma classe. Ou seja, atributos que dividem perfeitamente a classe fornecem informação máxima sobre a classe, e atributos que não se correlacionam com a classe não fornecem informação alguma. A medição de Ganho de Informação utiliza medidas de redução de Entropia. Essa técnica é utilizada para reduzir o conjunto de informação de um determinado banco de dados e minimizar os efeitos da maldição da dimensionalidade [1]. Sabendo o ganho de informação de cada atributo, é possível selecionar quais os melhores atributos para serem analisados sobre uma classe. Entretanto, esses cálculos apesar de simples, podem se tornar computacionalmente complicados dado a grande quantidade de informação de um banco de dados. Por isso, é importante a implementação de algoritmos que possibilitem o processamento em paralelo para otimizar o processamento do Ganho de Informação. Esse trabalho ilustra uma aplicação dessa técnica. Foi desenvolvido um algoritmo para computar o ganho de informação total dos atributos de um banco de dados. O algoritmo foi programado em Python utilizando a biblioteca pyspark. [2] [4]

Keywords: Ganho de Informação, Computação Paralela, pyspark

1. Algoritmos

O algoritmo para cálculo do ganho de informação é dividido em duas funções. A primeira, calcula a Entropia de um conjunto de dados categóricos, a segunda, calcula o Ganho de Informação de um atributo sobre uma classe. A segunda função utiliza a primeira.

1.1. Cálculo da Entropia de Shannon

O cálculo de Entropia de Shannon é feito através da equação 1 [2].

$$H = \sum_{i=0}^{N-1} -p_i \times log_2 \times p_i \quad (1)$$

O algoritmo que realiza o cálculo está exibido abaixo:

```
def Entropia (classe):
    """Calcula a Entropia de Shannon de uma distribuio de dados.
    Args:
        classe (RDD): RDD contendo conjunto de dados a ser calculado
            a entropia.
                     Valores devem ser categricos.
    Returns:
        float: valor de Entropia de Shannon calculado para o RDD.
    """
    #counts calcula paralelamente o contedo do RDD
    #como tuplas contendo (tipo, quantidade)
    counts = (classe.map(lambda x: (x, 1))
                    .reduceByKey(lambda a,b: a + b))
    # n recebe o valor total de itens do RDD
    n = classe.count()

    # probs calcula a probabilidade de cada um dos estados do RDD
    probs = counts.map(lambda x: x[1]/float(n))

    # Entropia calcula a entropia do RDD
    ## a  funo  map faz o calculo da Entropia de cada um dos estados
    ## a  funo  reduce faz o somatrio da entropia de Shannon
    entropia = (probs.map(lambda p: -p*math.log(p,2))
                     .reduce(lambda a,b: a + b))

    # retorna valor escalar refernte a entropia do RDD.
    return entropia
```

1.2. Cálculo do Ganho de Informação

O ganho de informação é uma medida que expressa o quanto de informação um atributo fornece em relação a classe. Atributos que dividem perfeitamente a classe fornecem máximo ganho de informação (1), enquanto que

atributos que se relacionam aleatoriamente em relação a classe fornecem mínimo ganho de informação (0). O ganho de informação é calculado utilizando a equação 2 [2]:

$$GI = H - \sum_{i=0}^{N-1} \frac{n_i}{n} \times H_i \tag{2}$$

Onde H é a entropia calculada para a classe, n_i é a quantidade de atributos de um determinado tipo de valor, N é o número de tipos diferentes de valores que o atributo pode assumir, n é o total de dados do atributos e H_i é a entropia calculada da classe para cada tipo específico de atributo. A classe é filtrada e dividida em grupos de classe, um para cada tipo de valor do atributo.

O código abaixo calcula o Ganho de Informação.

```
def infoGain (feature, classe, H):
    """Calcula o ganho de informao de um atributo em relao a uma
        classe.

    Args:
        feature (RDD): RDD contendo os conjuntos de dados do
            atributo a ser
                    calculado o Ganho de Informao

        classe (RDD): RDD contendo conjunto de dados da classe

        H (float): Entropia da Classe, previamente calculada.

    Returns:
        float: valor de ganho de informao ( reduo da Entropia) que
            o atributo fornece sobre a classe
    """
    # calcula paralelamente o contedo do RDD
    # como tuplas contendo (tipo, quantidade)
    feat_count = feature.map(lambda x: (x, 1))\
                        .reduceByKey(lambda a,b: a + b)\
                        .collect()
    # calcula as Entropias de um conjunto da classe dado cada um dos
        estados do atributo
    entropiasN = [Entropia(classe.zip(feature).filter(lambda x:
```

```
        x[1]==v).map(lambda x: x[0])) for v,_ in feat_count]

    # calcula a quantidade de itens no atributo
    n = classe.count()

    # calcula o ganho de informao do atributo.
    ig = H - sum([(f[1]/float(n))*p for f,p in zip(feat_count,
        entropiasN)])

    return ig
```

2. Resultados

A base de dados escolhida é um banco de dados de atraso em voôs comerciais [3]. A classe escolhida é Vôos cancelados. O objeitvo é encontrar qual dos atributos gera o maior ganho de informação.

A Entropia calculada para a classe é no valor de 0.1150, isso significa que os dados são fortemente puros. Existem 5729195 entradas com valor '0' (não cancelados), e 89884 entradas com o valor '1' (cancelados).

Tabela 1: Resultados

-	Mês	Dia do Mês	Dia da Semana	cia. Aérea
Ganho de Informação	0.0048013	0.0019261	0.0008652	0.0043092
Tempo com 8 Partições (min)	37.64	94.51	21.79	43.14
Tempo com 1 Partição (min)	36.46	90.98	21.26	41.94

Verificou-se que é obtido o melhor desempenho com 1 partição para o hardware utilizado (i7 4500 2 cores). Também verificou-se que os atributos que fornecem maior ganho de informação para os voos cancelados são o mês em que o voo acontece a **companhia aérea** responsável. Ou seja, os voos cancelados possuem maior correlação com o mês em que o voo é realizado e a companhia aérea responsável pelo voo. Há baixa correlação com o dia do mês e muito baixa correlação com o dia da semana.

3. Agradecimentos

Agradeço ao Prof. Dr. Fabricio Olivetti por ministrar o curso e me encorajar a continua-lo até o fim. Agradeço ao Prof. Dr. Ronaldo Prati, meu coorientador no mestrado, pela ajuda na realização deste projeto.

4. Referências

[1] **Dimention Reduction** Acessado em 10/05/2018. Disponível em: *https://www.knime.com/blog/seven-techniques-for-data-dimensionality-reduction*

[2] **Information Gain** Acessado em 10/05/2018. Disponível em: *https://www.kaggle.com/usdot/flight-delays*

[3] **2015 Flight Delays and Cancellations** Acessado em 10/05/2018. Disponível em: *https://www.kaggle.com/usdot/flight-delays*

[4] **Information Gain** Acessado em 10/05/2018. Disponível em: *https://courses.cs.washington.edu /courses/cse455/10au/notes/InfoGain.pdf*