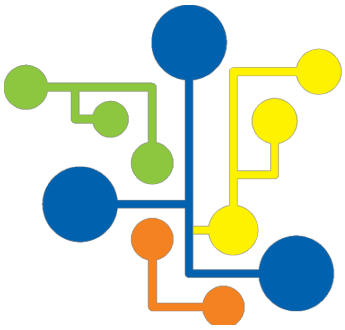

Data Science Academy

www.datascienceacademy.com.br

Matemática Para Machine Learning

Otimização AdaGrad

A ideia por trás deste algoritmo de otimização é, para cada parâmetro, armazenar a soma dos quadrados de todos os seus gradientes históricos. Essa soma é usada posteriormente para dimensionar a taxa de aprendizado.

Observe que, em contraste com as otimizações anteriores, temos diferentes taxas de aprendizado para cada parâmetro.

```
squared_gradients = 0
for i in range(iterations_count):
    param_gradients = evaluate_gradients(loss_function,
                                         data,
                                         params)
    squared_gradients += param_gradients * param_gradients
    params -= learning_rate * param_gradients/
                         (np.sqrt(squared_gradients) + 1e-8)
                           {1e-8 is to avoid divide by zero}
```

Agora a questão é como esta escala está nos ajudando quando temos um número de condição muito alto para nossa função de perda?

Para parâmetros com valores de gradiente altos, o termo ao quadrado será grande e, portanto, a divisão a longo prazo faria o gradiente acelerar lentamente nessa direção. Da mesma forma, os parâmetros com gradientes baixos produzirão termos quadrados menores e, portanto, o gradiente será acelerado mais rapidamente nessa direção.

No entanto, observe que, à medida que o gradiente é elevado a cada etapa, a estimativa de movimentação crescerá monotonicamente (em matemática, uma função monotônica é uma função entre conjuntos ordenados que preserva ou inverte a ordem dada. Esse conceito surgiu primeiro no cálculo e depois foi generalizado para o cenário mais abstrato da teoria da ordem) ao longo do tempo e, portanto, o tamanho do passo que nosso algoritmo levará para convergir para o mínimo será menor e menor.

E, de certo modo, isso é benéfico para problemas convexos, já que esperamos diminuir a velocidade para o mínimo neste caso. No entanto, o mesmo presente se torna uma maldição em caso de problemas de otimização não convexos, à medida que aumenta a chance de ficar preso em pontos de sela.

Não existe algoritmo ideal para todos os cenários e, como sempre, a escolha de um algoritmo de otimização depende do objetivo final do modelo preditivo e dos dados em mãos.

Referências:

http://cs229.stanford.edu/notes/cs229-notes1.pdf

https://stanford.edu/~rezab/classes/cme323/S15/notes/lec11.pdf

https://arxiv.org/abs/1609.04747

http://ruder.io/optimizing-gradient-descent/