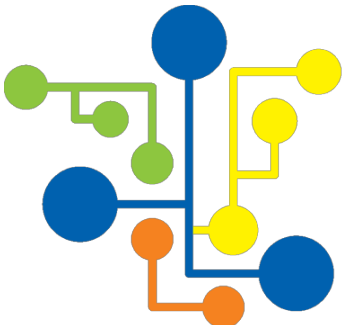


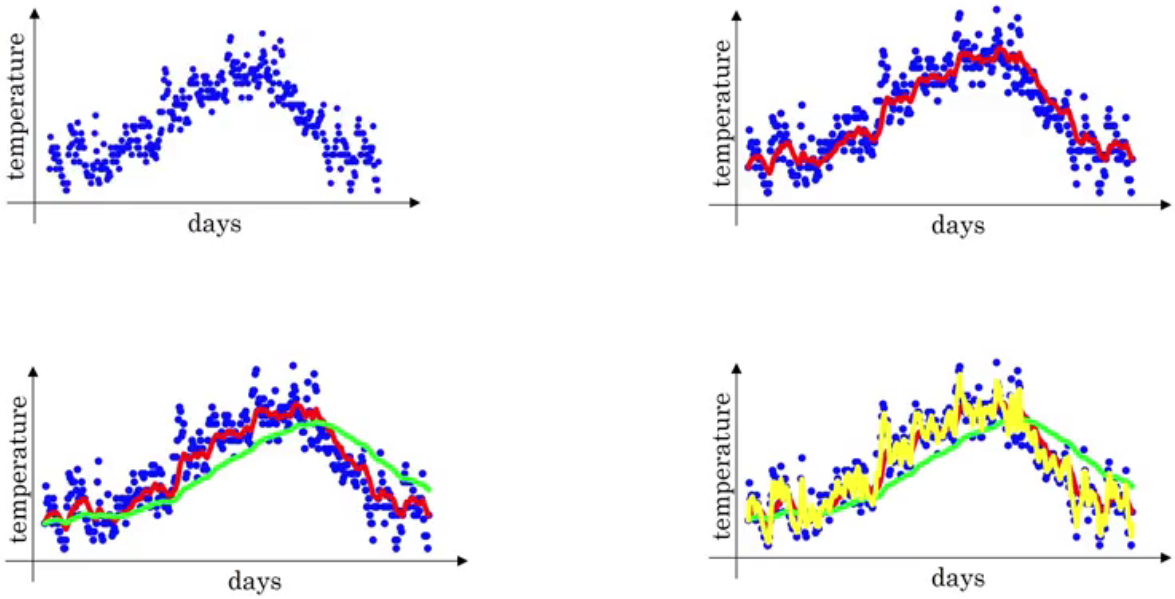
Data Science Academy

www.datascienceacademy.com.br

Matemática Para Machine Learning

Stochastic Gradient Descent Com Momentum

Para entender a dinâmica por trás das otimizações avançadas, primeiro precisamos entender o conceito de média ponderada exponencialmente (exponentially weighted average). Suponhamos que recebamos dados de temperaturas por dia de qualquer cidade específica durante todos os 365 dias do ano. Plotando, obtemos um gráfico como este no canto superior esquerdo da figura abaixo:



Agora, se quisermos calcular a temperatura média local ao longo do ano, procederemos da seguinte maneira:

```
NOTE :-
     alpha = 0.9 is randomly chosen weight.
     t(i)  is temperature at ith day.
     v(i)  is average temperature for ith day averaged over 1/(1 -
alpha) days.

v(0) = 0
v(1) = 0.9 * v(0) + 0.1 * t(1)
v(2) = 0.9 * v(1) + 0.1 * t(2)
...
v(i) = alpha * v(i-1) + (1 - alpha) * t(i)
```

A cada dia, calculamos a média ponderada das temperaturas do dia anterior e da temperatura atual do dia. A plotagem para o cálculo acima é mostrada no canto superior direito da figura anterior. Este gráfico tem uma temperatura média nos últimos 10 dias (alfa = 0,9). O gráfico no canto inferior esquerdo da figura anterior (com a linha verde) mostra os dados da média do gráfico nos últimos 50 dias (alfa = 0,98).

Um ponto importante a ser observado aqui é que, ao calcularmos a média de mais dias, o gráfico se tornará menos sensível às mudanças de temperatura. Por outro lado, se a média for menor que o número de dias, o gráfico será mais sensível às mudanças de temperatura e, consequentemente, ao comportamento distorcido.

Este aumento na latência é devido ao fato de que estamos dando mais peso às temperaturas do dia anterior do que a temperatura atual do dia.

Até aí tudo bem, mas a questão é qual a relação desse conceito com otimização? De forma bastante semelhante, ao calcular a média dos gradientes nos últimos valores, tendemos a reduzir as oscilações numa direção mais sensível e, assim, fazer convergir mais rapidamente. Isso é o que fazemos quando aplicamos *Momentum* ao método de descida estocástica do gradiente.

Na prática, os algoritmos de otimização baseados no *Momentum* são quase sempre mais rápidos que a descida do gradiente padrão. Matematicamente, teríamos isso em Python:

```
moment = 0
for i in range(iterations_count):
    param_gradients = evaluate_gradients(loss_function,
                                         data,
                                         params)
    moment = gamma * moment + param_gradients
    params += learning_rate * moment
           (where moment is building moving average of gradients.
                 gamma gives kind of friction = 0.9 or 0.99).
```

Perceba que o *Momentum* é apenas mais um hiperparâmetro do modelo, que nos ajudará a convergir mais rápido, ou seja, encontrar mais rapidamente os parâmetros ideais do modelo de aprendizado de máquina.

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/