Universidade de Brasília

DEPARTAMENTO DE ESTATÍSTICA

25 de abril de 2024

Lista 2: *Dropout* e *Keras*.

Prof. Guilherme Rodrigues

Redes Neurais Profundas

Tópicos especiais em Estatística 1

(A) **As questões deverão ser respondidas em um único relatório *PDF* ou *html*, produzido usando as funcionalidades do *Rmarkdown* ou outra ferramenta equivalente.**

(B) **O aluno poderá consultar materiais relevantes disponíveis na internet, tais como livros, *blogs* e artigos.**

(C) **O trabalho é individual. Suspeitas de plágio e compartilhamento de soluções serão tratadas com rigor.**

(D) **Os códigos *R* utilizados devem ser disponibilizados na integra, seja no corpo do texto ou como anexo.**

(E) **O aluno deverá enviar o trabalho até a data especificada na plataforma Aprender 3.**

(F) **O trabalho será avaliado considerando o nível de qualidade do relatório, o que inclui a precisão das respostas, a pertinência das soluções encontradas, a formatação adotada, dentre outros aspectos correlatos.**

(G) **Escreva seu código com esmero, evitando operações redundantes, comentando os resultados e usando as melhores práticas em programação.**

Nesta lista utilizaremos o pacote computacional *Keras* (ou outro de sua preferência – *PyTorch*, *TensorFlow*, *Theano*, *H2O*, *Caffe*) para ajustar redes neurais profundas. Considere os dados e os modelos descritos na Lista 1 para responder os itens a seguir.

Questão 1)

a) Altere seu código da Lista 1 (ou, se preferir, os códigos disponibilizados como gabarito) para implementar a técnica *dropout* na camada de entrada e na camada intermediária. Use $p = 0,6$, onde p representa a probabilidade de inclusão de cada neurônio. **Atenção:** neste item, não é preciso calcular o custo da rede no conjunto de validação! A cada nova iteração do algoritmo de otimização, a rede neural corrente gera estimativas pontuais aleatórias para as observações do conunto de treinamento. Essas estimativas, por sua vez, são usadas para calcular o custo no conjunto de treinamento e atualizar os pesos da rede. Reporte o menor custo observado durante o treinamento e salve os respectivos pesos para responder os demais itens da Questão 1.

b) Considerando os pesos obtidos em a), para a primeira observação do conjunto de teste, gere 200 previsões $(\hat{y}_{1,1}, \ldots, \hat{y}_{1,200})$, uma para cada sub-rede amostrada aleatoriamente. Use as previsões para construir uma estimativa pontual e um intervalo de confiança para y_1. Veja a Figura 7.6 do livro *Deep Learning*. Note que com esse procedimento, não é preciso assumir normalidade para os erros, como fizemos na Lista 1.

c) Repita o item b) para gerar estimativas pontuais para cada observação do conjunto de testes.

d) Use a regra *weight scaling inference rule* (página 263 do livro *Deep Learning*) para gerar novas estimativas para as observações do conjunto de testes. Qual dos procedimentos (o do item c) ou o utilizado neste item) produziu melhores resultados? Considerando o tempo computacional de cada um, qual você escolheria nessa aplicação?

Observação. Perceba que com o procedimento executado nessa questão, não implementamos a técnica de *early stopping*. Para usá-la, podemos, a cada nova iteração do algoritmo *SGD*, calcular o custo no conjunto de validação usando o item d), e, então, parar o treinamento quando o custo no conjunto de validação parar de diminuir.

Questão 2)

a) Ajuste a rede neural especificada na Lista 1 usando o *Keras*. Compare com sua implementação (Lista 1, item e) quanto ao tempo computacional e ao custo obtido no conjunto de validação. Use o mesmo algoritmo de otimização (*full gradient descent*) e ponto de partida.

b) Ajuste a rede neural mais precisa (medida pelo MSE calculado sobre o conjunto de validação) que conseguir, com a arquitetura que quiser. Use todos os artifícios de regularização que desejar (*weight decay*, *Bagging*, *droupout*, *Early stopping*). Reporte a precisão obtida para essa rede no conjunto de validação.

Considerando a rede ajustada no item b), responda os itens a seguir.

c) Refaça o item h) da Lista 1 para essa nova rede. Comente os resultados.

d) Use a função de previsão do *Keras* para prever o valor da variável resposta $\hat{y} = f(x_1 = 1, x_2 = 1; \theta)$, para θ definido de acordo com a rede ajustada. (Veja o item a) da Lista 1).

e) Neste exemplo meramente didático, conhecemos a superfície que estamos estimando. Apresente, lado a lado, a Figura 1 da Lista 1 e a superfície estimada pela sua rede neural. Para tanto, basta trocar a variável `mu` pelos valores preditos pela rede. Comente os resultados.

f) Construa uma nova rede, agora ajustada sobre os valores previstos (ao invés dos valores observados de y) para cada observação dos conjuntos de treinamento e validação. Use a arquitetura mais parcimoniosa que conseguir, sem comprometer substancialmente o poder de previsão da rede (quando comparada à obtida no item 2b). Cite um possível uso para essa nova rede.