Relatório ADO 02
Projeto Integrador V

Gabriel Lins da Silva
Luiz Kenji Coppola Yamamoto
Paulo Gonçalves Magalhães

Abril 2017

1 Introdução

A atividade consiste na implementação do algoritmo Multilayer Perceptron, um classificador supervisionado, que forma uma rede neural com peso para cada neurônio mediador - entre a entrada e a saída -, onde a saída é a classe calculada pela somatória do produto de cada dado de entrada com o peso da ligação com o neurônio, utilizando o conceito de Backpropagation, que percorre a rede neural inteira até a saída, conduzindo os dados através de neurônios, retornando um booleano 1(verdadeiro) para a classe calculada e booleanos 0(falso) para as classes restantes. O erro, que é a diferença da classe esperada com a classe obtida, é utilizado para ajustar os pesos, assim chegando cada vez mais próximo de um algoritmo

2 Desenvolvimento

2.1 Planejamento

Na parte do planejamento da rede neural, consideramos 3 estruturas de dados para representá-la de uma forma que o software controle de forma rápida e simples estes dados. Para isto, consideramos 3 estruturas possíveis de dados:

- A rede neural como a classe maior, havendo dentro dela uma matriz de neurônios e uma matriz de ligação entre os neurônios, já que consideramos a coluna na matriz de neurônio como a camada, considerando a primeira coluna como camada de input e a última coluna como camada de output;
- Grafos, utilizando os vértices como os neurônios e as arestas com a modificação do peso;
- A rede neural como a classe maior, havendo estruturas de dados diferentes dentro desta, com estruturas para camada de input, camadas ocultas, camada de output e ligações entre camadas;

Desconsideramos utilizar grafos pela grande quantidade de operações que ocorrem quando vamos tratar os pesos e os neurônios. Também desconsideramos a divisão entre 5 estruturas de dados, pois torna o código difícil de compreender e as operações lentas.

Assim, optamos por utilizar a rede neural com conjuntos de matrizes, pois as operações se tornam fáceis de controlar, e a mudança dos pesos é bem mais fácil de verificar, com a possibilidade de verificar simplesmente com a visualização da matriz de pesos.

A leitura de CSV que utilizamos neste algoritmo é diferente da utilizada no algoritmo anterior, pois verifica também os nomes das classes, facilitando a visualização do usuário, além de deixar aberto ao usuário o conjunto de dados a ser utilizado, tornando o software mais interativo.

Para melhorar mais ainda a interação, é fornecido ao usuário a opção de inserir a quantidade de camadas (de 1 a 3, como pré-definido) e a quantidade de neurônios por camada.

2.2 Cross Validation

Para a implementação do cross validation, utilizamos partições sequenciais de aproximadamente 1/10 instâncias para teste e o restante para treinamento. Definindo de uma melhor forma, temos os primeiros 1/10 dos dados na primeira execução do Backpropagation como teste, e o restante como treinamento do algoritmo. Na segunda execução do Backpropagation, temos os primeiros 1/10 e os últimos 8/10 como conjunto de treinamento e os segundos 1/10 como conjunto de teste, e assim por diante.

2.3 Backpropagation: Forward

Após a criação da estrutura da rede neural, a inicialização da matriz de pesos da rede e a execução do Cross Validation, fizemos o algoritmo de forward, utilizando valores da camada de input, passando pelas camadas ocultas, calculando a somatória do produto entre os pesos e o valor do neurônio anterior, realizando as operações de cálculo da função sigmoide e armazenando os resultados no neurônio, até chegar na camada de output, onde definimos qual a classe obtida na execução, comparando assim com a classe esperada da instância de dados e calculando o erro médio.

2.4 Backpropagation: Backward

Assim, com os erros, obtemos os novos pesos das conexões anteriores, atualizando conforme novas instâncias de dados de treinamento são inseridas no software, até convergir a um valor específico, até que a diferença entre a média do erro quadrático anterior e a média do erro quadrático atual seja menor que o limiar adotado, que é 0.01 por definição. Por meio de algumas fórmulas que contém o erro como um dos atributos do aprendizado, obtemos pesos novos para as conexões entre os neurônios, fazendo com que a saída seja mais precisa.

3 Resultados

A priori, os resultados aparentavam divergir bastante do esperado, havendo a necessidade de reformular todo o código e revisitar as explicações sobre os conceitos.

Após as melhorias do código, obtemos os seguintes resultados:

4 Conclusão

Ao realizar este trabalho, foi concluído que o algoritmo MLP é bem estruturado em termos de visualização, pois é fácil visualizar graficamente a rede neural e os conceitos atrelados ao movimento dos dados dentro dela.

Outro ponto a ser ressaltado neste algoritmo é que o aprendizado de máquina é totalmente visível quando vemos um valor convergindo para um valor específico, demonstrando que, se o algoritmo receber um input novo, terá uma probabilidade alta de definir a classe atribuida aos dados corretamente.