UNIVERSIDADE FEDERAL DO CEARÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA E DA COMPUTAÇÃO - PPGEEC
DISCIPLINA DE RECONHECIMENTO DE PADRÕES

Aluna: Kamila Amélia Sousa Gomes
Matrícula: 516916

RELATÓRIO TRABALHO 2

Questão ELM e RBF

As questões a seguir foram implementadas no software Scilab na versão 6.1.0.

0.0.1 Implemente as redes neuraus ELM e RBF para classificar a base de dados Dermatology. Adotar hold-out com subsampling (20 execuções) e leave-one-out. Obs. Normalizar os atributos dos vetores.

Para a execução desta questão, foi utilizado banco de dados dermatologia, disponível em: https://archive.ics.uci.edu/ml/datasets/Dermatology. Ele é constituído pelo diagnóstico de doença de pele com base em informações clínicas (coletadas pelo médico no consultório) e informações hispatológicas (resultantes de uma biópsia – análise do tecido em um laboratório de patologia). O conjunto de dados possui 6 classes, 34 atributos e 366 amostras. Como mostrado na Tabela 2.

Classes	Quantidade de amostras	Patologias
1	111	Psoríase
2	60	Dermatite Seborréica
3	71	Líquen plano
4	48	Pitiríase rósea
5	48	Dermatite Crônica
6	20	Rubrar Pilar

Tabela 1 – Informações sobre o conjunto de dados dermatologia.

Na Tabela 2, não são quantificados 366 amostras, pois existem amostras com valores do tipo *Nan* (nulos), o que deixa o conjunto de dados com 358 amostras. Para isso, após a leitura da base de

dados, houve uma etapa de pré-processamento, que excluiu os valores nulos da mesma. Em seguida, a base foi permutada aleatoriamente para não haver enviesamento.

ELM

O *Extreme learning machines(ELM)* são redes neurais não-recorrentes, que se assemelham com as redes MPL. A ELM admite apenas uma única camada oculta, onde seus pesos iniciais são determinados aleatoriamente, além disso, suas funções de ativação são determinadas de forma aleatória dentro de um conjunto de funções. Este conjunto pode ser constituído, por exemplo, por funções trigonométricas, funções de base radial, tangente hiperbólica.

Inicialmente, para a averiguar a versão *Leave-one-out*, que é uma implementação de validação cruzada onde o aproximador de função é treinado em todos os dados, exceto para um ponto e uma previsão é feita para esse ponto, deve-se acessar o arquivo "elm-loo.sce". Durante seu desenvolvimento, foram utilizados a quantidade de neurônios ocultos solicitadas pelo usuário, 34 atributos e pesos aleatórios e a função ativação Sigmóide. Ao testar diferentes quantidades de neurônios têm-se:

- Para 10 neurônios uma acurácia de 78.77%.
- Para 15 neurônios uma acurácia de 88.54%.
- Para 25 neurônios uma acurácia de 92.73%.
- Para 40 neurônios uma acurácia de 95.25%.
- Para 50 neurônios uma acurácia de 96.36%.
- Para 90 neurônios uma acurácia de 96.92%.

Já para o método *Hold-Out*, que é um método simples de validação cruzada, que consiste na ideia básica de dividir o conjunto de dados de treinamento em duas partes, ou seja, treinar e testar o modelo, deve-se acessar o arquivo "elm-ho.sce". Durante seu desenvolvimento, foram utilizados 70% do conjunto de dados para treino e 30% para teste. Ademais, ttambém foram utilizados a quantidade de neurônios ocultos solicitadas pelo usuário, 34 atributos e pesos aleatórios e a função ativação Sigmóide. Além disso, para a efetuação do hold-out com 20 execuções, foi utilizado o método *grand*, método de geração de números aleatórios, que tem como argumento o tamanho da matriz desejada, a distribuição *'prm'* (permutações aleatórias) e a matriz/vetor. Inicialmente, este método também foi utilizado para permutar a base aleatoriamente. Ao testar diferentes quantidades de neurônios têm-se:

- Para 10 neurônios uma acurácia de 82.52%.
- Para 15 neurônios uma acurácia de 89.57%.
- Para 20 neurônios uma acurácia de 89.76%.
- Para 30 neurônios uma acurácia de 94.81%.
- Para 50 neurônios uma acurácia de 95.98%.

- Para 90 neurônios uma acurácia de 95.04%.

RBF

O *Radial Basis Function (RBF)* é uma evolução das redes MPL. Nas redes RBF, a função de ativação de cada neurônio da camada oculta é função da distância entre seus vetores de peso e de entrada. Além disso, tradicionalmente, são redes de duas camadas, onde a primeira camada utiliza funções de ativação não lineares e a segunda camada utiliza funções de ativação lineares.

Inicialmente, para a averiguar a versão *Leave-one-out*, deve-se acessar o arquivo "rbf-loo.sce". Durante seu desenvolvimento foi feita uma normalização dos dados e uma correção na coluna de rótulos. Além disso, foram utilizadas diferentes quantidades de neurônios ocultos, no entando, o algoritmo não obteve sucesso.

Perceptron

0.0.2 Implemente um neurônio Perceptron com valores de pesos inicialmente aleatórios para separar pontos 3D em duas classes. O programa deverá solicitar ao usuário a quantidade de pontos desejada, as coordenadas x, y, z de cada ponto e a respectiva classe (1 ou 2). A seguir, ele deverá plotar os pontos (usar símbolos diferentes para cada classe) e a superfície de separação obtida.

O Perceptron é uma rede neural de camada única, conhecido por ser um classificador linear binário. Ele é usado na aprendizagem supervisionada e pode ser usado para classificar os dados de entrada fornecidos. Nele os pesos podem ser treinados para produzir um vetor alvo que deve corresponder ao vetor de entrada.

No desenvolvimento desta questão, é pedido que o usuário indique quantos pontos deseja plotar, suas coordenadas e sua classe. Em seguida, o conjunto de dados é criado através da concatenação dos elementos de cada classe. Neste caso, os valores dos pesos são escolhidos aleatoriamente, pela função rand. Dessa forma, inicia-se a preparação para a aprendizagem da rede neural perceptron. São efetuadas 10 épocas para a execução do algoritmo. Após a divisão dos rótulos e atributos, é calculada a função perceptron, dada pela multiplicação do vetor atributos pelo peso. Logo após, são calculadas a função ativação e o erro, para assim, aplicar a função perceptron. Por fim, são mostrados os resultados de entrada, saída esperada, saída real, erro e pesos utilizados.

Para a efetuação da plotagem 3d, foi utilizada a função meshgrid, disponibilizada em:

https://help.scilab.org/docs/5.5.0/pt$_BR/meshgrid.html$.

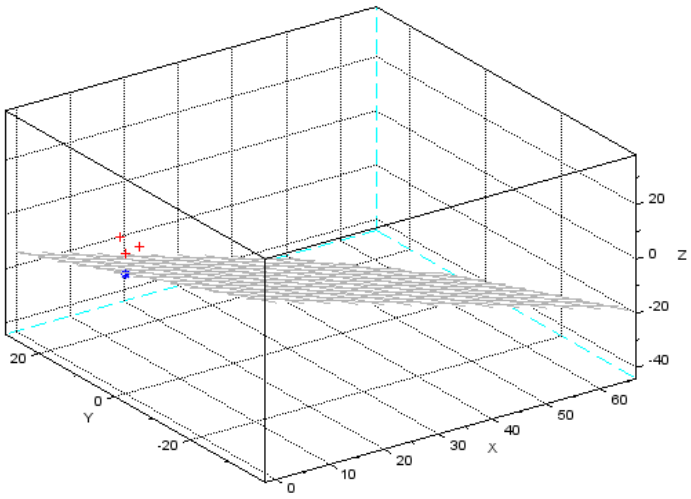
Como exemplo de execução, foram testadas alguns pontos com as seguintes coordenadas:

- Os pontos [4 2 6] na classe 1.
- Os pontos [4 7 6] na classe 1.

- Os pontos [4 2 6] na classe 1.
- Os pontos [8 97 69] na classe 0.
- Os pontos [87 79 69] na classe 0.
- Os pontos [85 69 47] na classe 0.
- Os pontos [76 98 78] na classe 0
- Os pontos [58 69 47] na classe 0
- Os pontos [2 3 4] na classe 1.

Dessa forma obteve-se como gráfico a figura abaixo.

Figura 1 – Perceptron.



Elaborado pela autora.