Instituto Politécnico de Coimbra

Instituto Superior de Engenharia de Coimbra

Trabalho Prático

Redes Neuronais

Unidade Curricular: Conhecimento e Raciocínio

ANO LETIVO 2023/2024

Beatriz Filipa Santos Marques - 2022137934

Pedro Teixeira Amorim - 2022157609

Índice

Introdução

As redes neurais artificiais têm desempenhado um papel fundamental no avanço da inteligência artificial e no desenvolvimento de sistemas inteligentes capazes de aprender e tomar decisões a partir de dados. Entre os diversos tipos de redes neurais, as redes feedforward destacam-se pela sua simplicidade e eficácia em tarefas de classificação e regressão.

Este relatório apresenta um estudo e análise detalhados das redes neurais feedforward aplicadas à tarefa de classificação, utilizando um dataset específico relacionado à hepatite. O objetivo deste trabalho é explorar a influência de diferentes arquiteturas de redes, funções de treino e funções de ativação no desempenho da classificação, além de investigar técnicas para lidar com valores ausentes nos dados.

Dataset Utilizado

O dataset utilizado para a realização do trabalho prático foi o dataset 1 - Hepatite.

Começando pela análise do dataset, composto por 14 atributos (ID, Category, Age, Sex, ALB, ALP, ALT, AST, BIL, CHE, CHOL, CREA, GGT, PROT), no ficheiro start apresenta 10 linhas (usado para a realização das tarefas descritas em 3.4.a), assim como no ficheiro test (usado para a realização das tarefas descritas em 3.4.c), no ficheiro train apresenta 600 linhas, no entanto possui alguns valores em falta (NA) (usado para a realização das tarefas descritas em 3.3 e 3.4.b).

Preparação do Dataset

O target do dataset é o atributo Category ao qual foram necessárias fazer algumas alterações antes de ser usado no estudo e análise de desempenho de redes neuronais feedforward. Assim como o atributo Sex que também este teve de ser convertido em valores numéricos.

Para a execução desta tarefa foi usado o Excel.

Começamos por duplicar a coluna do atributo Sex, portanto a coluna D. Aplicamos à coluna duplicada a seguinte função:

=SE(D2="f";1;SE(D2="m";0;""))

Depois foi apenas necessário arrastar o preenchimento automático para todas as linhas da coluna duplicada. Com a nova coluna alterada eliminamos a anterior e substituímos pela coluna com os booleanos corretos.

Para alterar a coluna Category, respetiva coluna B, foi aplicada a mesma metodologia. Duplicámos a tabela e aplicámos a seguinte função:

=SE(B2="0=Blood Donor";0;SE(B2="0s=suspect Blood Donor";1;SE(B2="1=Hepatitis";2;
SE(B2="2=Fibrosis";3;SE(B2="3=Cirrhosis";4;SE(B2="NA";"NA";""))))))

Target

O atributo que corresponderá à saída desejada (target) da rede neuronal é a coluna Category, adaptada anteriormente. Possui alguns valores em falta (NA).

Feitas as alterações necessárias e identificado o target prosseguimos para a implementação de um sistema de raciocínio baseado em casos para preencher os atributos com valores em falta (NA).

Retrieve

A função Retrieve procura o caso mais semelhante para cada caso com valores em falta, dentro da tabela Train, usando uma medida de similaridade calculada com base na distância euclidiana entre os atributos.

Pesos dos Atributos

Para a implementação da fase retrieve foi necessário atribuir diferentes pesos a cada atributo que tem influência nos casos de hepatite:

```
pesos = [5 3 4 2 5 5 5 2 1 3 4 3];
```

- Age: Peso 5. Considerando que a idade pode ter um impacto significativo no prognóstico e na resposta ao tratamento.
- Sex: Peso 3. Pode ter algum impacto, mas geralmente não é tão determinante quanto outros fatores.
- ALB (Albumina): Peso 4. A diminuição da albumina indica comprometimento da função hepática e pode influenciar na gravidade da doença.
- ALP (Fosfatase Alcalina): Peso 2. Um aumento da fosfatase alcalina pode indicar problemas hepáticos, mas pode ter menos impacto direto do que outros parâmetros.
- ALT (Alanina Aminotransferase): Peso 5. A alanina aminotransferase é uma enzima hepática importante e níveis elevados podem indicar danos no fígado.
- AST (Aspartato Aminotransferase): Peso 5. Assim como a ALT, a AST é uma enzima hepática e níveis elevados podem indicar danos no fígado.
- BIL (Bilirrubina): Peso 5. A bilirrubina é um marcador importante de função hepática e níveis elevados podem indicar problemas graves.
- CHE (Colinesterase): Peso 2. Embora a colinesterase seja produzida pelo fígado, a sua diminuição pode não ser tão específica para danos hepáticos como outros parâmetros.
- CHOL (Colesterol): Peso 1. Níveis elevados de colesterol podem estar associados a doenças hepáticas, mas geralmente têm menos importância direta na avaliação da hepatite.
- CREA (Creatinina): Peso 3. A creatinina pode indicar função renal, que pode ser afetada em casos graves de hepatite, mas pode ter menos relevância direta.

- GGT (Gama-Glutamil Transferase): Peso 4. A GGT é uma enzima hepática e níveis elevados podem indicar dano hepático.

- PROT (Proteína Total): Peso 3. A diminuição da proteína total pode indicar comprometimento da função hepática, mas pode ter menos importância direta do que outros parâmetros.

Similaridade Local

A função de similaridade local é a função distancia_euclidiana que calcula a distância euclidiana entre dois valores usando a fórmula geral, que no matlab é apresentada da seguinte mandeira:

```
function [res] = distancia_euclidiana(val1, val2)

    res = sqrt((val1 - val2)^2);

end
```

Esta função é usada para calcular a distância entre o novo caso e cada caso na biblioteca de casos para cada atributo. A distância euclidiana é uma medida comum de dissimilaridade e é calculada com a raiz quadrada da diferença ao quadrado entre os dois valores.

Similaridade Global

A função de similaridade global é a parte do código que calcula a distância geral. Esta função combina as distâncias locais (calculadas para cada atributo) em um único valor. Isso é feito calculando a média ponderada das distâncias locais, onde os pesos são dados pelo vetor pesos.

A média ponderada é então subtraída a 1 para converter a medida de dissimilaridade em uma medida de similaridade (uma vez que a similaridade é o oposto da dissimilaridade). A similaridade final varia de 0 (não similar) a 1 (muito similar).

```
distancia_geral = (distancias * pesos') / soma(pesos);
similaridade_final = 1 - distancia_geral;
```

Cbr

Este script em matlab tem como objetivo preencher os valores em falta (NA) no ficheiro Train.csv usando uma abordagem baseada em similaridade.

Fazemos a leitura do ficheiro Train.csv. Em seguida, é iniciado um ciclo for que percorre todas as linhas desta tabela para verifica-se se existem valores em falta. Caso existam, o caso com valores em falta é identificado.

De seguida chamamos a função Retrieve, que analisa o conjunto de dados e procura casos semelhantes ao caso com valores em falta. Se forem encontrados casos semelhantes, o mais parecido é selecionado e os valores em falta na linha correspondente são preenchidos com base nos valores desse caso.

Ao longo do processo, são apresentadas as seguintes mensagens de informação:

```
>> Cbr
Caso mais semelhante com uma similaridade de 98.30%:
    ID    Category    Age    Sex    ALB    ALP    ALT    AST    BIL    CHE    CHOL    CREA    GGT    PROT
    __    ________    ___    ___    ____   ____   ____   ___    ___    ____   ____    ____    ____   ____

    12        0       33      0     36.3   78.6   23.6   22      7     8.56   5.38    78     19.4   68.7

Caso mais semelhante com uma similaridade de 98.74%:
    ID    Category    Age    Sex    ALB    ALP    ALT    AST    BIL    CHE    CHOL    CREA    GGT    PROT
    ___   ________    ___    ___    ____   ____   ____   ____   ___    _____  ____    ____    ____   ____

    105       0       41      0     44.7   74.9   25.2   20.2   6.3    10.34  4.23    74     23.7   72.1

Caso mais semelhante com uma similaridade de 98.51%:
    ID    Category    Age    Sex    ALB    ALP    ALT    AST    BIL    CHE    CHOL    CREA    GGT    PROT
    ___   ________    ___    ___    ____   ____   ____   ____   ____   ____   ____    ____    ____   ____

    359       0       38      1     41.2   61.9   19.4   22.9   10.5   7.86   3.61    85     19.5   66.6
```

…

Por fim, a tabela é atualizada e guardada num novo ficheiro TrainAdaptado.csv, mantendo o mesmo delimitador usado anteriormente.

Estudo e Análise de Redes Neuronais

Ficheiro Start

Para estudar, treinar e analisar redes neuronais feedforward começamos por usar o ficheiro Start.csv, com com uma camada de 10 neurónios.

Realizamos alguns testes alterando apenas as funções de ativação e de treino.

Utilizando as seguintes funções:

```
% Definir diferentes funções de treino
trainFcns = {'trainlm', 'traingd', 'trainbfg'};

% Definir diferentes funções de ativação
transferFcns = {'radbas', 'radbasn', 'purelin'};
```

Podemos concluir que apesar da maioria se apresentar com uma precisão de 100% alguns casos apresentam erros e menos precisão:

```
Erro: 0.475016
Precisão: 52.50%
Execution Time: 1.03 seconds

Erro: 0.000000
Precisão: 100.00%
Execution Time: 0.67 seconds

Erro: 0.000000
Precisão: 100.00%
Execution Time: 0.73 seconds

Erro: 0.019345
Precisão: 98.07%
Execution Time: 1.33 seconds

Erro: 0.275000
Precisão: 72.50%
Execution Time: 0.73 seconds

Erro: 0.000000
Precisão: 100.00%
Execution Time: 0.70 seconds
```

Por outro lado, utilizando as funções seguintes conclui-se que estas são as mais precisas e consequentemente as que apresentam menos erro:

```
% Definir diferentes funções de treino
trainFcns = {'trainlm', 'traingd', 'trainbfg'};

% Definir diferentes funções de ativação
transferFcns = {'radbas', 'radbasn', 'purelin'};
```

Resultados:

```
Erro: 0.000000
Precisão: 100.00%
Execution Time: 0.09 seconds

Erro: 0.000000
Precisão: 100.00%
Execution Time: 0.12 seconds

Erro: 0.000000
Precisão: 100.00%
Execution Time: 0.09 seconds

Erro: 0.000000
Precisão: 100.00%
Execution Time: 0.10 seconds

Erro: 0.000000
Precisão: 100.00%
Execution Time: 0.11 seconds

Erro: 0.000000
Precisão: 100.00%
Execution Time: 0.12 seconds
```

Assim é possível concluir que as funções de treino 'trainlm', 'traingd' e 'trainbfg' e as funções de ativação 'radbas', 'radbasn' e 'purelin' são as mais precisas, com uma precisão de 100%.

Ficheiro Train

Continuando o estudo, treino e análise de redes neuronais feedforward continuamos com o ficheiro *Train.csv*, com uma camada de 10 neurónios.

Realizamos alguns testes alterando apenas as funções de ativação e de treino.

Utilizando as seguintes funções:

```
% Definir diferentes funções de treino
trainFcns = {'trainscg', 'trainlm', 'trainrp'};

% Definir diferentes funções de ativação
transferFcns = {'logsig', 'tansig', 'purelin'};
```

Podemos concluir que a função de treino '*trainlm'* apesar de ter uma execução mais demorada, produz resultados com uma precisão mais elevada, maioria a 100%.

```
Funções de Treino: trainlm e Ativação: tansig
Erro: 0.000000
Accuracy: 100.00%
Execution Time: 3.69 seconds

Funções de Treino: trainlm e Ativação: tansig
Erro: 0.000000
Accuracy: 100.00%
Execution Time: 3.77 seconds

Funções de Treino: trainlm e Ativação: tansig
Erro: 0.005591
Accuracy: 99.44%
Execution Time: 3.84 seconds

Funções de Treino: trainlm e Ativação: tansig
Erro: 0.006565
Accuracy: 99.34%
Execution Time: 3.72 seconds

Funções de Treino: trainlm e Ativação: tansig
Erro: 0.000000
Accuracy: 100.00%
Execution Time: 3.61 seconds

Funções de Treino: trainlm e Ativação: tansig
Erro: 0.000000
Accuracy: 100.00%
Execution Time: 3.66 seconds
```

```
Funções de Treino: trainscg e Ativação: logsig
Erro: 0.048601
Accuracy: 95.14%
Execution Time: 1.06 seconds

Funções de Treino: trainscg e Ativação: logsig
Erro: 0.049029
Accuracy: 95.10%
Execution Time: 1.08 seconds

Funções de Treino: trainscg e Ativação: logsig
Erro: 0.029966
Accuracy: 97.00%
Execution Time: 1.21 seconds

Funções de Treino: trainscg e Ativação: logsig
Erro: 0.041766
Accuracy: 95.82%
Execution Time: 1.06 seconds

Funções de Treino: trainscg e Ativação: logsig
Erro: 0.036215
Accuracy: 96.38%
Execution Time: 1.07 seconds

Funções de Treino: trainscg e Ativação: logsig
Erro: 0.026332
Accuracy: 97.37%
Execution Time: 1.06 seconds
```

Os piores resultados foram apresentados usando a função de ativação '*Purelin*', que apresentou uma precisão de apenas 60%.
Por outro lado com função de ativação '*logsig*' obtivemos resultados de alta precisão, 90-100%, consistentemente e com uma execução ligeira.

Ficheiro Teste

Para terminar este estudo de redes neuronais feedforward usamos o ficheiro *Test.csv*, com o objetivo de testar a precisão das melhores redes neuronais treinadas usando o ficheiro Train.

Começamos por carregar as melhores redes usando a função load.

```
% Carregar as melhores redes
load('melhores_redes.mat', 'melhores_redes');
```

De seguida simulamos a rede com os atributos do ficheiro *Test.csv*, que usamos para calcular a precisão da mesma.

```
% Simular a rede com o conjunto de dados de teste
y = melhores_redes{i}.rede(input);

% Calcular o erro da rede
erro = perform(melhores_redes{i}.rede, target, y);
```

Para calcular as métricas de acerto comparamos os valores obtidos aos valores alvo do ficheiro *Test.csv* convertendo o numero de acertos para percentagem.

```
% Calcular a métrica de acerto
acertos(i) = sum(y_class == target)/ length(target) * 100;
```

```
Métricas de acerto das melhores redes:
    50
    40
    60
```

Assim, é possível verificar que, as métricas de acerto em torno de 50% sugerem que as redes neuronais estão a conseguir classificar os exemplos de teste com uma precisão razoável.

Conclusão

Este trabalho proporcionou uma valiosa experiência no desenvolvimento e análise de modelos de redes neurais feedforward. Ao enfrentar desafios reais de pré-processamento de dados, treino de redes e avaliação de desempenho, pudemos aprofundar a nossa compreensão dos princípios subjacentes á aprendizagem da máquina e explorar a sua aplicação num contexto do mundo real.

Este trabalho demonstrou a importância da escolha adequada das funções de ativação e de treino na eficácia das redes neurais feedforward para a tarefa de classificação. Além disso, destacou a necessidade de uma preparação cuidadosa dos dados antes do treino da rede e a importância de técnicas como o Raciocínio Baseado em Casos para lidar com valores ausentes em conjuntos de dados reais.