	UNIVERSIDADE FEDERAL DA PARAÍBA CENTRO DE INFORMÁTICA	
	Disciplina	Aprendizagem de Máquina
	Semestre	2025.1
	Professores	Bruno Jefferson de Sousa Pessoa Gilberto Farias de Sousa Filho

Mini-Projeto
Reconhecimento de Dígitos

1. Introdução

O reconhecimento de dígitos escritos a mão é um problema clássico de classificação na área de visão computacional. O problema consiste em receber uma imagem de um número escrito à mão, codificada em tons de cinza, e classificar o dígito decimal (0-9) ali contido. Para estudantes e pesquisadores das técnicas de aprendizado de máquina, o *dataset* MNIST, cujos exemplos de instâncias estão ilustrados na Figura 1, é utilizado para comparação de técnicas, competições e construções de novas soluções.

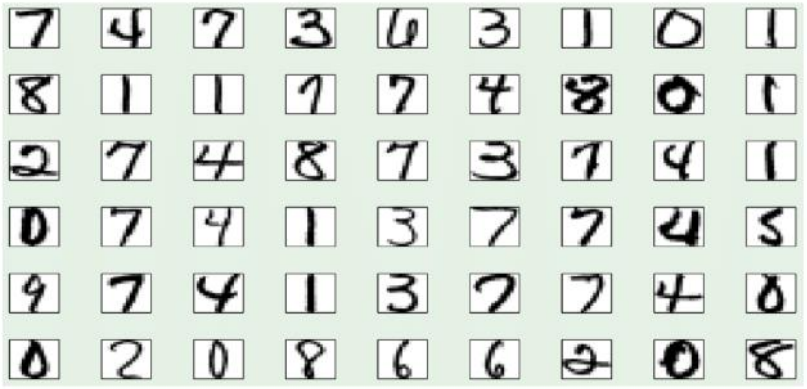

Figura 1. *Dataset* MNIST com imagens dos dígitos escritos a mão.

2. *Dataset* MNIST Adaptado

Os arquivos *train.csv* e *test.csv* contêm imagens do *dataset* MNIST, em escala de cinza, dos dígitos 0, 1, 4 e 5 escritos a mão. Cada imagem é composta por 28 linhas e 28 colunas em um total de 784 pixels. Cada pixel possui um valor associado único, que indica seu tom de cinza. Quanto mais alto é esse valor, mais escuro é o pixel. Os valores de cada pixel estão no intervalo fechado $[0, 255]$.

Os dados de entrada, (*train.csv*), possuem 785 colunas. A primeira coluna, chamada "*label*", é o dígito que foi desenhado pelo usuário. O resto das colunas contém os valores dos pixels da imagem associada.

Cada coluna de pixel, nos dados de treino, é nomeada como "*pixelx*", onde x é um inteiro no intervalo $[0, 783]$. Para localizar este pixel na imagem, suponha que decompomos x como $x = i * 28 + j$, onde i e j são inteiros no intervalor $[0, 27]$. Então o "*pixelx*" está localizado na

linha i e coluna j de uma matriz $28x28$ (indexada por zero). Por exemplo, "*pixel*31" indica o valor do *pixel* que está na quarta coluna, da esquerda pra direita, e na segunda linha.

Os dados de teste, (test.csv), possuem o mesmo formato dos dados de treinamento.

3. Descrição das atividades

Implementar três classificadores de dígitos contidos no *dataset* MNIST Adaptado, utilizando os três modelos lineares de Aprendizagem de Máquina (AM) estudados**: Perceptron, Regressão Linear e Regressão Logística**. Detalhes da implementação estão descritos a seguir.

3.1. Redução da dimensão das amostras

Para trabalharmos com modelos de AM que possuem muito pouco grau de liberdade para a construção de sua função hipótese, devemos diminuir a complexidade dos dados de entrada através da redução do número de parâmetros p das amostras de treinamento.

Como já foi dito na descrição do *dataset*, cada instância é composta por $p = 784$ parâmetros de entrada, sendo um parâmetro por pixel. Logo, há a necessidade de reduzir a quantidade de parâmetros total, a fim de atingir bons resultados na classificação das de tais imagens usando-se modelos de AM mais simples. Uma forma de reduzir consideravelmente o vetor de características é sintetizar os dados das imagens em apenas duas informações de entrada ($p = 2$) que são muito importantes na identificação de um dígito numérico: a intensidade e a simetria da imagem.

Intensidade da imagem

Como os pixel mais escuros possuem valores maiores (255 representa o preto), a intensidade de uma imagem pode ser calculada pela equação

$$I = \frac{\sum_{x=0}^{783} pixelx}{255}$$

que soma os tons de cinza de cada *pixel* e divide por 255, tentando uma aproximação da quantidade de *pixels* pretos na imagem.

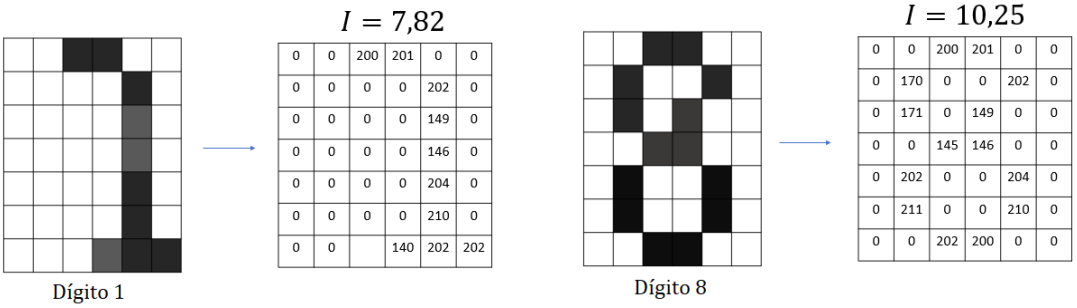

Figura 2. Comparação da intensidade I dos *pixels* entre os dígitos 1 e 8.

É possível perceber que a intensidade do dígito 1 é normalmente menor que a do dígito 8, como podemos observar na Figura 2.

Simetria da imagem

A simetria de uma imagem é computada a partir da definição de eixos de simetria. Existem duas simetrias fáceis de se computar: a vertical e horizontal. Por exemplo, na simetria vertical dividem-se as colunas da matriz de pixels em duas partes, lado direito e esquerdo, como ilustrado pelo eixo vertical da Figura 3, e computa-se a diferença dos valores dos pixels pertencentes as distintas partes.

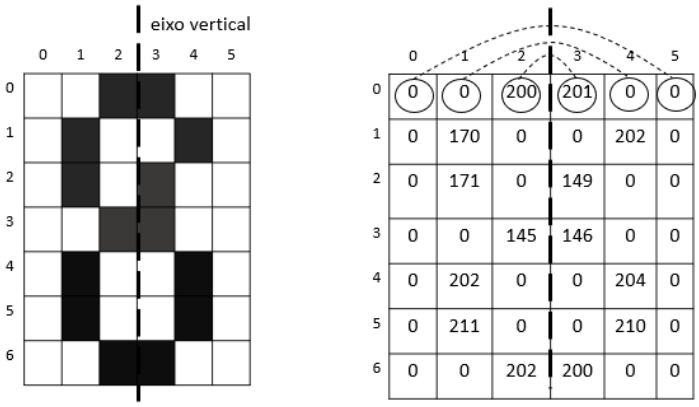

Figura 3. Eixo vertical da matriz de *pixels* do dígito 8.

A simetria vertical deve ser computada linha por linha. Em cada linha soma-se a diferença em módulo do valor do *pixel* da primeira coluna com o valor da última, da segunda com a penúltima, terceira com a antepenúltima e assim por diante.

Seja pix_{28x28} a matriz de valores dos *pixels* da imagem, então, temos a equação

$$S_v = \frac{\sum_{i=0}^{27}\sum_{j=0}^{13}\left\|pix_{i,j} - pix_{i,27-j}\right\|}{255}$$

para a simetria vertical. O valor de S_v define uma aproximação da quantidade de *pixels* pretos assimétricos. Ou seja, quando $S_v = 0$, a imagem possui simetria vertical perfeita.

A simetria horizontal é análoga a simetria vertical, sendo que o eixo horizontal divide as linhas da matriz pelo meio, criando duas partes, superior e inferior. Nesse caso, a diferença de valores entre os *pixels* é computada de cima para baixo. Logo, através de algumas adaptações da equação da simetria vertical, é possível computar o S_h e, por fim, somar ao S_v e obter o valor de simetria da imagem completa.

Nesta atividade, deve-se construir novos arquivos de treino e teste a serem chamados de *train_redu.csv* e *test_redu.csv*. Esses arquivos conterão 3 as seguintes colunas: label, intensidade e simetria.

Para cada linha do arquivo *train.csv*, deve-se extrair o valor da coluna *label*, depois os 784 valores de *pixels* da imagem das outras colunas, computar os valores de intensidade e simetria associados e registrar tais valores nas colunas “label”, “intensidade” e “simetria” como uma linha do arquivo *train_redu.csv*. O mesmo tratamento deve ser feito no arquivo *test.csv* para criar o arquivo *test_redu.csv*.

3.2. Classificação dos dígitos 1 x 5

Como no modelo *Perceptron* a classificação é binária, uma alternativa para realizar classificações multiclasse de dígitos é construir, inicialmente, uma solução que classifique apenas dois valores de dígitos: 1 e 5, por exemplo. Para isto, deve-se:

- Realizar um filtro nos dados dos arquivos *train_redu.csv* e *test_redu.csv*, deixando apenas as imagens com valores 1 ou 5 na coluna *label*, construindo as instâncias *train1x5* e *test1x5*;
- Plotar os dados de *train1x5* em um gráfico de duas dimensões (intensidade X simetria) como ilustrado na Figura 4. Dados com $label = 1$ plotar de azul e dados com $label = 5$ plotar de vermelho;

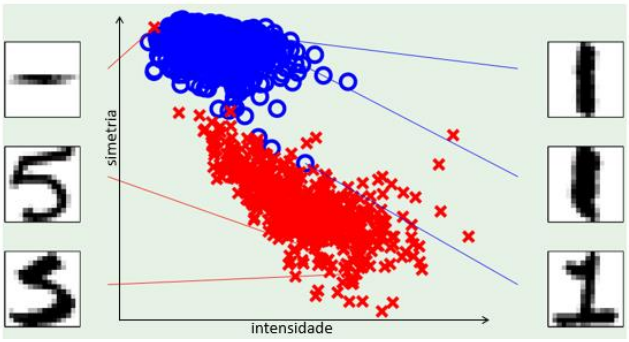

Figura 4. Dígitos 1 e 5 com seus valores de intensidade e simetria plotados.

- Treinar os três classificadores com os dados de *train1x5*. Construir o vetor $X = \{(intensidade, simetria)\}$ para toda imagem de *train1x5*. Atribuir o valor alvo $y = +1$ para a imagem com valor $label = 1$ e $y = -1$ para $label = 5$;
- Criar um método de predição do dígito que retorne o valor 1 quando o classificador linear classificar a saída como $y = +1$ e retorne o valor 5 quando $y = -1$;
- Construir o vetor $X = \{(intensidade, simetria)\}$ para toda imagem de *test1x5*. Atribuir o valor alvo $y = label$ para cada imagem de *test1x5*;
- Testar os três classificadores com os dados de *test1x5*;
- Plotar a reta de cada classificador sobre os dados;
- Gerar a matriz de confusão e os relatórios de eficácia de classificação de cada classificador.

3.3. Classificador de dígitos completo

Para construir um classificador para os quatro dígitos contidos na base, deve-se implementar uma estratégia conhecida como “um contra todos”. Nessa estratégia, inicialmente, escolhe-se o dígito 0 para ser a classe $y = +1$ e todas os outros dígitos (1, 4 e

5), temporariamente, definidos como a classe $y = -1$. Essa transformação está ilustrada na Figura 5.

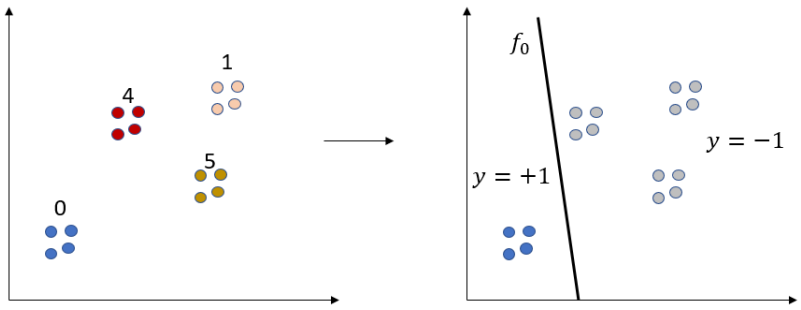

Figura 5. Transformação dos dados multiclasse em dados para classificação binário digito 0 contra todos.

A função hipótese f_0 é inferida pelos novos dados de treino e usada para classificar os dados de teste. Se f_0 classificar o novo dado como da classe $+1$, então podemos afirmar que a imagem associada é do dígito 0. Se o dado for classificado como -1, nada pode ser afirmado. O próximo passo é criar uma nova instância de treino, eliminando as instâncias com *label=0*, e construindo uma classificação binária do dígito 1 contra todos.

Ao final serão construídas três funções hipótese que juntas irão realizar a classificação multiclasse dos quatro dígitos. Seja x a imagem teste a ser classificada, classifique x com o seguinte algoritmo:

```
para os dígitos i ∈ [0,1,4]
        se f_i(x) = +1
                classifique como dígito i
        senão
                se i == 4
                        classifique como dígito 5
```

3.4. Comparação entre os classificadores

Para comparar os três classificadores, implemente a estratégia “um contra todos” para cada um dos algoritmos de classificação, construa:

1. A matriz de confusão e o relatório de eficácia de classificação contendo: acurácia, precisão, *recall* e *f1 score*;
2. Para cada classificador plotar as três retas construídas sobre os dados dos 4 dígitos.

4. Implementações avançadas

- Adotar a heurística *weight decay* para o algoritmo de regressão logística a fim de regularizar o parâmetro *lambda*.

- Implementar a estratégia de “um contra todos” definindo a ordem de teste dos dígitos que leve a melhor acurácia global. Ou seja, a ordem proposta anteriormente pode ser alterada para uma ordem arbitrária que produza uma melhor acurácia de classificação.

Instruções gerais

- O presente projeto deve ser desenvolvido em duplas.
- O projeto deverá ser enviado ao professor (bruno@ci.ufpb.br) até o dia **18/08/2025** e apresentado no dia **19/08/2025,** em horários a serem definidos posteriormente.
- O projeto deve ser implementado em python e seu código colocado em um notebook jupyter ou colab.
- Deverá ser enviado um arquivo zipado, contendo os códigos e o notebook contendo as saídas das execuções, no formato descrito a seguir:
    - AM-Projeto1-Autor1-Autor2.zip
    - Ex.: AM-Projeto1-Jose_Silva-Maria_Oliveira.zip