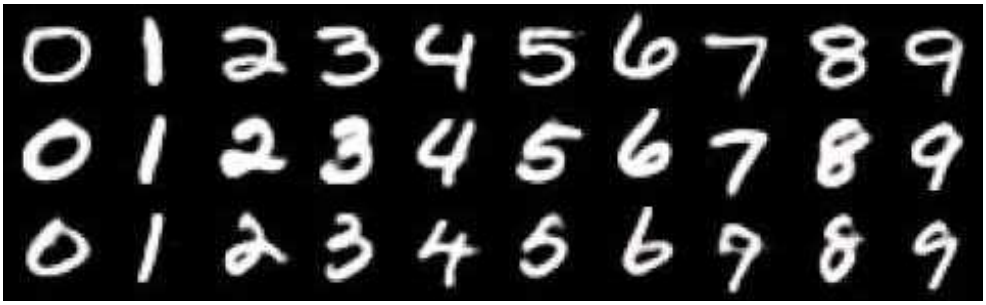
Trabalho 4
Classificação de Dígitos - MNIST

1 Descrição do Dataset

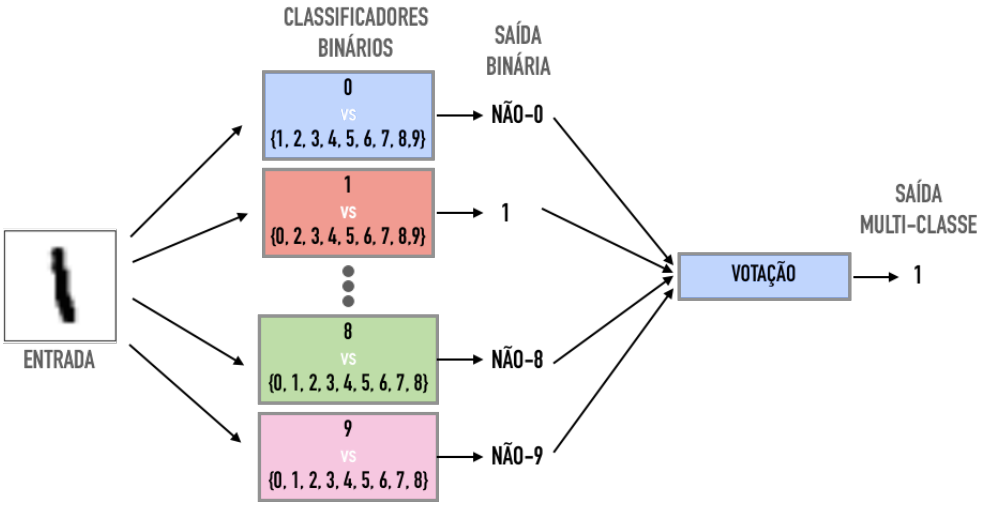
Nestre trabalho, você irá classificar qual o dígito (de 0 a 9) presente em imagens do dataset MNIST. As imagens são em tons de cinza (apenas um canal) de dimensões 28 por 28. Abaixo alguns exemplos do dataset são:



Cada imagem foi linearizada em um vetor de 784 (28×28) posições, referentes aos valores de cada pixel. A cada linha do arquivo de dados fornecido, a primeira posição é referente à classe do número (um dígito entre 0 e 9) e cada um dos elementos seguintes (nas posições 2 até 785 vetor) são referentes aos pixels da imagem ($pixel_{1,1}$, ... $pixel_{1,28}$, $pixel_{2,1}$, ..., $pixel_{2,28}$, ..., $pixel_{28,28}$) com valores entre 0 e 255.

2 Tarefas

Neste trabalho, você irá explorar uma tarefa multi-classe, no qual cada imagem deve ser classificada em uma entre as 10 classes do problema. Para isso, pedimos que você implemente o protocolo *One vs All* visto em sala. Neste protocolo, cada classificador é treinado utilizando os exemplos de uma determinada classe contra exemplos de todas as outras 9 classes. Ao todo serão 10 classificadores, cujas respostas serão combinadas (por exemplo, por votação) para determinar a classe final de um exemplo de teste. Um exemplo desse protocolo adaptado para esta Tarefa pode ser visto abaixo:



Dessa forma, pedimos que você:

1. Inspecione os dados. Quantos exemplos de cada classe você tem?
2. Divida os dados em um conjunto de treinamento e outro de validação, seguindo a proporção de 80%/20% e garantindo que as classes estejam aproximadamente balanceadas em cada conjunto. *Não se esqueça de definir o* ***seed*** $= 42$ *para que nós possamos replicar os experimentos com os mesmos grupos. Para garantir, defina o seed em múltiplos pontos do código.*
3. Para cada uma das técnicas abaixo (SVM e Redes Neurais), treine classificadores binários utilizando o protocolo *One vs All*:
   - **SVM**:
     (a) Treine SVMs com kernel RBF e lineares, fazendo um *grid search* nos valores de C e *sigma* (apenas para o RBF).
     (b) Compare a acurácia normalizada dos modelos no conjunto de treino e validação.
   - **Redes Neurais**:
     (a) Treine RNs variando o número de camadas escondidas e/ou o número de neurônios em cada camada.
     (b) Compare a acurácia normalizada dos modelos no conjunto de treino e validação.
4. Compare o melhor modelo *OvA* treinado com SVM com o melhor *OvA* obtido com Redes Neurais, reportando a acurácia normalizada no conjunto de treino e validação.
5. **IMPORTANTE:** selecione o melhor entre os modelos para ser testado no conjunto de teste. Ao final do seu script, prepare o código para ler o arquivo "*mnist_ test.csv*", fazer quaisquer transformações necessárias nos dados (por exemplo, normalização ou PCA) e predizer com o melhor modelo treinado. O seu programa deve imprimir a matriz de confusão e a acurácia normalizada (média das acurácias por classe) para o conjunto de teste.
6. Escreva um relatório de no máximo 4 páginas, compilando os experimentos realizados. Escreva também pelo menos 1 parágrafo com as conclusões tiradas na atividade;
7. *EXTRAs*:
   (a) Antes de treinar os modelos, reduza a dimensionalidade dos dados de entrada com PCA. Você deve utilizar apenas os dados de treino para encontrar as bases do PCA e, então, utilizá-las para transformar os dados de validação e teste.
   (b) Treine alguma das técnicas (SVM ou RN) com o protocolo *One Vs One* (treine um classificador para cada par de classes e combine as respostas com votação). Compare com os modelos obtidos pelo protocolo *One Vs All*.

3 Arquivos

O Moodle não suporta arquivos acima de 25 Mb, portanto, devido ao tamanho do dataset (1̃00 Mb), fizemos o upload dos arquivos em uma pasta no Google Drive que pode ser acessada em `https://goo.gl/AcZ4kT`. Os arquivos disponíveis são:

- *mnist_ trainVal.csv*: conjunto de dados para treinamento e validação;
- *mnist_ test.csv*: conjunto de dados retido pelo professor que será utilizado para avaliar as melhores soluções e computar a nota da competição;
- *aux_ tarefa04.r*: um código R com uma função auxiliar para visualizar as imagens;

4 Avaliação

O dataset foi previamente dividido aleatoriamente em dois conjuntos — treino/validação e teste — e apenas o primeiro será disponibilizado para que você implemente as suas soluções.

A avalidação consistirá da análise do relatório e do código submetidos no Moodle. As notas para esses pontos irão de 0.0 a 8.0. Os pontos restantes serão atribuídos de acordo com a acurácia normalizada que você obtiver no conjunto de teste. Iremos *rankear* todos os trabalhos pela acurácia obtida. O trabalho com a melhor acurácia (1ª posição) irá ganhar 2.0 pontos, o segundo mais bem colocado 1.9 e o terceiro 1.8. Da 4ª até a 10ª colocação receberá 1.5 pontos. Da 11ª até a 15ª ganhará 1.0 ponto e da 16ª até 25ª terá 0.5 ponto. As colocações após 25ª e também aqueles cujo código não pode ser executado (seja por erro ou por exigir modificações da parte do monitor) ficarão com 0.0.

Observações sobre a avaliação:

- Procurem deixar o código o mais *"plug and play"* possível para facilitar o teste por parte do monitor. Preparem o código de forma a treinar o melhor modelo, ler os dados de teste, fazer os pré-processamentos necessários e as predições.
- O trabalho poderá ser feito individualmente ou em duplas, podendo haver repetição das duplas a cada trabalho;
- O código e o relatório deverão ser submetidos no Moodle por **apenas um integrante da dupla**;
- Não se esqueçam de listar os nomes dos integrantes da dupla no início do relatório;
- As notas do trabalho serão divulgadas em até uma semana após o prazo da submissão;