Atividade 1: Perceptron

Disciplina: Redes Neurais Artificiais
Programa de Pós-Graduação em Ciência da Computação (PPGCC-UNESP)

Prof. Dr. Lucas C. Ribas

1. Implementação do Perceptron

1. **Função de Treinamento**

   - Implemente uma função para treinar o algoritmo do *Perceptron*.
   - Sua função deve receber como entrada: os dados de treinamento, os rótulos, taxa de aprendizado, número de épocas e uma semente aleatória (opcional).
   - Ao final, retorne:
     - os pesos aprendidos,
     - o número de épocas executadas,
     - e a evolução do erro ao longo das épocas. Para isso, a cada época guarde o erro (número de amostras classificadas incorretamente / total de amostras).
   - Utilize como base os pseudocódigos apresentados em aula.
   - Não utilize implementações prontas. Use somente a biblioteca `numpy`.
   - Lembre-se de incluir o bias (constante 1 ou -1) em cada vetor de entrada e também no vetor de pesos (inicializa aleatório).

2. **Função de Teste**

   - Implemente uma função de teste do *Perceptron*, que utiliza os pesos aprendidos para realizar previsões sobre novos dados.
   - Calcule as previsões e retorne a acurácia sobre os dados fornecidos.
   - Não utilize implementações prontas. Use somente a biblioteca `numpy`.
   - Lembre-se de incluir o bias (constante 1 ou -1) em cada vetor de entrada como no treinamento. O vetor peso treinado já tem o bias.

2. Experimentos

1. **Dataset #1**

   - 2 atributos. 140 amostras de treino e 60 amostras de teste.
   - Treine o perceptron utilizando:
     - 100 épocas,
     - taxa de aprendizado igual a 0.1.
   - Calcule e apresente a acurácia nos conjuntos de treino e teste.

- Mostre o gráfico da evolução do erro de treinamento em função das épocas.
- Plote:
  - os dados de treinamento com a linha que representa a fronteira de decisão aprendida.
  - os dados de teste com a mesma linha da fronteira de decisão.
- Reflita sobre os resultados e comente o que pode estar acontecendo em cada situação.

2. **Dataset #2**

   - 2 atributos. 175 amostras de treino e 75 amostras de teste.
   - Repita os mesmos passos realizados com o **Dataset 1**.

3. **Dataset #3**

   - 10 atributos. 147 amostras de treino e 63 amostras de teste.
   - Treine o perceptron com os seguintes parâmetros iniciais:
     - 100 épocas
     - taxa de aprendizado igual a 0.1.
   - Calcule e apresente a acurácia nos conjuntos de treino e teste.
   - Mostre o gráfico da evolução do erro de treinamento em função das épocas.
   - Realize novos testes, variando:
     - Taxa de aprendizado $\eta \in \{0.1,\ 0.001,\ 0.0001\}$,
     - Número de épocas $\in \{100,\ 200\}$.
   - Para cada combinação de parâmetros, execute o treinamento e teste. Mostre a acurácia e desvio padrão em cada combinação.
   - Para cada combinação de época e taxa de aprendizado mostre o gráfico com o *erro médio* $\times$ *época*. Ou seja, qual o valor do erro em cada época.
   - Reflita sobre os resultados e comente o que pode estar acontecendo em cada situação.
   - *Opcional*: um teste adicional é treinar e testar várias vezes, calcular média e desvio padrão das acurácias.

Base de dados

Para leitura das base de dados que devem ser utilizadas e estão em anexo use o código abaixo. As matrizes de atributos estão na dimensão (n^o atributos x n^o amostras), no entanto, use e implemente como achar mais conveniente.

```
df_train_loaded = pd.read_csv('train_dataset1.csv')
df_test_loaded = pd.read_csv('test_dataset1.csv')

X_train = df_train_loaded.drop('label', axis=1).values.T  # (n_features, n_amostras)
y_train = df_train_loaded['label'].values.reshape(1, -1)  # (1, n_amostras)

X_test = df_test_loaded.drop('label', axis=1).values.T
y_test = df_test_loaded['label'].values.reshape(1, -1)
```

3. Entrega

- Entregue um **notebook Jupyter (.ipynb)** contendo:
  - o código-fonte das funções desenvolvidas,
  - os resultados dos experimentos solicitados,
  - os gráficos e análises,
  - e breves explicações ou comentários sobre os resultados obtidos.
  - fique a vontade para incluir mais testes, visualizações ou variações nos experimentos. Brinque à vontade :-).
- Lembre-se: como os pesos iniciais são definidos aleatoriamente, é esperado que os resultados variem a cada execução.