**Relatório do Desenvolvimento de Rede Neural Multicamadas para Classificação**

**Alunos: Enzo Januario e Nicolas Ferraz**

**1. Introdução**

O objetivo deste projeto foi desenvolver uma Rede Neural Multicamadas (MLP) para realizar a tarefa de classificação, utilizando um dataset preparado especificamente para representar padrões complexos de dados. Este relatório descreve detalhadamente o dataset, os atributos selecionados, os ajustes realizados no modelo e as análises dos resultados obtidos.

**2. Descrição do Dataset**

**2.1 Fonte do Dataset**

O dataset foi gerado artificialmente utilizando a função make\_classification do scikit-learn, uma biblioteca consolidada para aprendizado de máquina. Embora seja um dataset sintético, ele foi configurado para imitar características típicas de problemas reais de classificação. Por ser gerado programaticamente, o dataset não possui uma "data de coleta", mas foi preparado especificamente para este projeto em novembro de 2024.

**2.2 Estrutura dos Dados**

O dataset contém:

* **Amostras**: 1.000 instâncias.
* **Atributos (Features)**: 20 variáveis preditoras numéricas, sendo:
  + **15 informativas**: Atributos diretamente relacionados à separação das classes.
  + **5 redundantes**: Variáveis derivadas das informativas, com correlações internas.
* **Classes**: 3 categorias distintas, representadas em codificação one-hot para treinamento da rede neural.

**2.3 Justificativa para Escolha do Dataset**

Este dataset foi escolhido por suas características:

1. **Complexidade controlada**: Permite ajustar os níveis de dificuldade para separação das classes.
2. **Diversidade de atributos**: Apresenta um número suficiente de variáveis para explorar o aprendizado da rede neural.
3. **Equilíbrio**: O dataset possui classes balanceadas, evitando a introdução de viés no modelo.

**2.4 Atributos Utilizados**

Todos os 20 atributos foram considerados no treinamento. A justificativa para isso é:

* A inclusão de atributos informativos e redundantes possibilita que a rede identifique as combinações mais relevantes para a separação das classes.
* Nenhum atributo foi descartado para garantir que o modelo avalie corretamente todas as relações e interações possíveis.

**3. Configuração e Estrutura da Rede Neural**

A rede neural desenvolvida é uma Multilayer Perceptron (MLP) configurada com três camadas ocultas, projetadas para capturar padrões complexos.

**3.1 Arquitetura da Rede**

* **Camada de entrada**: 20 neurônios (um para cada atributo).
* **Camadas ocultas**:
  + Primeira camada: 64 neurônios.
  + Segunda camada: 32 neurônios.
  + Terceira camada: 16 neurônios.
* **Camada de saída**: 3 neurônios (uma para cada classe), com ativação softmax.
* **Funções de ativação testadas**:
  + relu: Para treinar com padrões não lineares com eficiência.
  + tanh: Para avaliar sua estabilidade e desempenho comparativamente.

**4. Ajustes de Hiperparâmetros**

Para otimizar o modelo, foram realizadas variações nos seguintes parâmetros:

1. **Taxa de aprendizado (learning\_rate)**:
   * Valores testados: 0.01 e 0.001.
   * Justificativa: Avaliar a velocidade e estabilidade da convergência.
2. **Número de épocas (epochs)**:
   * Valores testados: 50 e 100.
   * Justificativa: Observar o impacto do treinamento prolongado na perda e precisão.
3. **Tamanho do lote (batch\_size)**:
   * Valores testados: 16 e 32.
   * Justificativa: Examinar como o número de amostras por iteração influencia o aprendizado.
4. **Funções de ativação**:
   * relu e tanh foram comparadas quanto à eficiência em lidar com padrões complexos.

**5. Resultados Obtidos e Análise**

**5.1 Métricas de Avaliação**

* **Perda de validação**: Indicador principal de erro durante o treinamento.
* **Precisão de validação**: Mede a capacidade de generalização do modelo.

**5.2 Observações**

**Funções de ativação:**

* relu apresentou melhores resultados em termos de estabilidade e convergência, especialmente com taxas de aprendizado mais altas.
* tanh teve um desempenho mais lento e, em alguns casos, apresentou maior perda de validação.

**Taxa de aprendizado:**

* 0.01 foi mais eficiente para convergência rápida e resultados mais estáveis.
* 0.001 exigiu mais épocas para alcançar resultados similares, mas apresentou menos oscilações.

**Tamanho do lote:**

* Lotes menores (batch\_size=16) resultaram em melhores ajustes, capturando padrões mais detalhados.
* Lotes maiores (batch\_size=32) reduziram o tempo de treinamento, mas apresentaram maior perda de validação.

**Número de épocas:**

* 100 épocas foram necessárias para alcançar convergência com a taxa de aprendizado menor (0.001).
* 50 épocas foram suficientes para taxas de aprendizado maiores (0.01), reduzindo o custo computacional.

**6. Conclusão**

O melhor modelo foi obtido com:

* **Taxa de aprendizado**: 0.01.
* **Função de ativação**: relu.
* **Batch size**: 16.
* **Número de épocas**: 100.

Essa configuração resultou em um modelo robusto com menor perda de validação e alta precisão. As variações realizadas destacaram a importância do ajuste fino dos hiperparâmetros para maximizar o desempenho da rede neural.

Este estudo demonstra que o uso de redes neurais multicamadas pode ser eficaz em problemas de classificação, desde que se considerem as interações entre dados e parâmetros do modelo. O mesmo processo pode ser replicado para datasets reais, adaptando-se às suas características específicas.