UNIVERSIDADE FEDERAL DE LAVRAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

Aluno: Michel Alexandrino de Souza **Matrícula: 202111084**
Aluno: João Pedro Alves Carneiro Valadão **Matrícula: 202020295**

GCC128 – Inteligência Artificial **Prof. Ahmed Ali Abdalla Esmin**

Relatório Técnico – Perceptron

1. Definição e apresentação da base de dados

O conjunto de dados *Iris*, introduzido por Ronald Fisher em 1936, constitui um benchmark clássico em *machine learning*, frequentemente utilizado para tarefas de classificação supervisionada. Composto por 150 amostras igualmente distribuídas entre três espécies do gênero *Iris* (*setosa*, *versicolor* e *virginica*), o dataset apresenta quatro atributos numéricos contínuos: comprimento e largura da sépala (em cm), e comprimento e largura da pétala (em cm). Cada instância está perfeitamente categorizada, sem valores faltantes ou inconsistentes, o que o torna ideal para prototipagem de algoritmos. A relevância científica desta base reside em suas propriedades estatísticas bem comportadas e na separabilidade parcial das classes. Enquanto *Iris setosa* é linearmente separável das demais, *versicolor* e *virginica* apresentam regiões de sobreposição em seu espaço de características, criando um desafio interessante para classificadores. A distribuição balanceada (50 amostras por classe) elimina a necessidade de técnicas de reamostragem, permitindo uma análise mais direta do desempenho dos modelos supervisionados. Já o conjunto de dados *Wine*, oriundo da UCI Machine Learning Repository, é composto por 178 amostras de vinhos provenientes da região italiana de cultivação de uvas. As amostras estão divididas em três classes, cada uma representando um tipo distinto de cultivar. Cada instância é descrita por 13 atributos numéricos contínuos, como teor alcoólico, concentração de flavonoides, acidez, entre outros fatores físico-químicos. Diferentemente da base *Iris*, que possui apenas quatro dimensões, a base *Wine* apresenta uma maior complexidade dimensional, o que representa um desafio adicional para os modelos de classificação. A técnica de classificação utilizada neste trabalho foi o Perceptron Multicamadas (MLP).

O Perceptron Multicamadas (*MLPClassifier*) é um algoritmo de aprendizado supervisionado baseado em redes neurais artificiais, amplamente utilizado em tarefas de classificação. Seu objetivo é aprender uma função de mapeamento que associe vetores de

entrada a rótulos de classe, por meio de uma arquitetura composta por camadas de neurônios conectados. O funcionamento do modelo baseia-se na combinação linear dos atributos de entrada, seguida da aplicação de uma função de ativação não linear — neste caso, a função ReLU (*Rectified Linear Unit*), responsável por introduzir não linearidades ao modelo. Durante o treinamento, o algoritmo realiza a propagação dos dados da entrada até a saída, estimando a classe de cada instância, e calcula o erro entre a predição e o valor real. Esse erro é então utilizado no processo de retropropagação (*backpropagation*), que ajusta os pesos das conexões por meio da descida do gradiente estocástico. O objetivo é minimizar a função de perda, que quantifica o desempenho do modelo. O treinamento ocorre de forma iterativa até que os pesos se estabilizem, respeitando um número máximo de épocas ou até que a convergência seja atingida. Neste projeto, o Perceptron foi aplicado aos conjuntos de dados *Iris* e *Wine*, ambos com rótulos conhecidos, o que caracteriza o aprendizado supervisionado. Para garantir que todos os atributos contribuíssem igualmente ao treinamento, foi realizada a padronização dos dados com a técnica *StandardScaler*. A configuração adotada para o modelo envolveu uma única camada oculta com 100 neurônios, função de ativação ReLU, e limite de 500 iterações.

A avaliação do desempenho do Perceptron foi conduzida por meio das métricas de acurácia, precisão e revocação, permitindo observar a capacidade do modelo em generalizar padrões mesmo em conjuntos com sobreposição parcial entre as classes, como é o caso das espécies *versicolor* e *virginica* na base *Iris*. A combinação entre capacidade de modelagem não linear e aprendizagem iterativa faz do Perceptron uma alternativa robusta para problemas de classificação em múltiplas dimensões, como os encontrados no conjunto *Wine*.

2. Comparação entre as aplicações

Neste projeto, foram implementadas duas abordagens supervisionadas de classificação: o algoritmo KNN (K-Nearest Neighbors), desenvolvido manualmente sem o uso de bibliotecas específicas para o modelo, e o MLPClassifier (Multi-Layer Perceptron), utilizando a implementação da biblioteca scikit-learn. Ambas as abordagens foram aplicadas aos conjuntos de dados *Iris* e *Wine* com o objetivo de comparar o desempenho entre um classificador tradicional baseado em instâncias e uma rede neural simples. Para garantir que as diferenças de escala entre os atributos não influenciassem o treinamento da rede neural, os dados foram previamente normalizados utilizando a técnica de padronização (*StandardScaler*). A avaliação do desempenho dos classificadores foi realizada por meio das métricas acurácia, precisão e revocação, permitindo analisar a taxa de acertos globais, a

proporção de acertos entre as previsões positivas e a capacidade dos modelos em identificar corretamente as classes reais. Os resultados de classificação e as matrizes de confusão foram salvos em arquivos separados para cada conjunto de dados, facilitando a análise e comparação final dos métodos.

Implementação KNN

Neste projeto, o algoritmo KNN (K-Nearest Neighbors) foi implementado manualmente, sem o uso direto de bibliotecas de machine learning para a classificação. A função *predict_knn* calcula a distância euclidiana entre cada ponto de teste e todos os pontos de treino, selecionando os *k* vizinhos mais próximos para determinar a classe majoritária. O código utiliza o *numpy* para operações vetoriais e o *Counter* da biblioteca *collections* para identificar a classe mais frequente entre os vizinhos. O processo de divisão dos dados em treino e teste é realizado com o *train_test_split* do *Scikit-learn*, garantindo uma avaliação justa do modelo.

Implementação com Scikit-learn (MLPClassifier)

Para a abordagem de redes neurais, foi utilizado o *MLPClassifier* do *Scikit-learn*, que implementa um Perceptron Multicamadas (MLP). Antes do treinamento, os dados são normalizados com *StandardScaler* para melhorar a performance da rede. O modelo é treinado com uma camada oculta de 100 neurônios e até 500 iterações, utilizando a mesma divisão de treino e teste do KNN para garantir comparabilidade. O uso do *Scikit-learn* facilita a configuração e o treinamento do MLP, além de fornecer métodos otimizados para ajuste dos parâmetros e avaliação do modelo.

Comparação de Desempenho

A comparação de desempenho entre KNN e MLP foi realizada nos conjuntos de dados *Iris* e *Wine*. Os resultados mostram que ambos os algoritmos apresentam bom desempenho, mas o MLP geralmente supera o KNN, especialmente em conjuntos de dados mais complexos como o Wine. No dataset *Iris*, ambos os métodos atingiram alta acurácia, com o KNN chegando a 100% e o MLP a 96,67%. Já no dataset *Wine*, o KNN obteve 80,56% de acurácia, enquanto o MLP alcançou 97,22%, evidenciando a maior capacidade do MLP de capturar padrões mais complexos.

Comparação de Métricas

As métricas avaliadas incluem *acurácia*, *precisão* e *recall*, além da matriz de confusão para análise detalhada dos acertos e erros por classe. No *Iris*, tanto o KNN quanto o MLP

apresentaram métricas próximas de 1, indicando excelente desempenho. No *Wine*, o MLP apresentou métricas significativamente superiores ao KNN: acurácia de 97,22% contra 80,56%, precisão de 97,41% contra 80,92% e recall de 97,22% contra 80,56%. As matrizes de confusão mostram que o KNN teve maior dificuldade em distinguir entre as classes 1 e 2 do *Wine*, enquanto o MLP cometeu poucos erros.

3. Resultados e limitações

Resultados

Os resultados demonstram que ambos os algoritmos são eficazes para classificação em conjuntos de dados bem comportados como o *Iris*. No entanto, para dados mais complexos, como o *Wine*, o MLP se destaca, apresentando maior capacidade de generalização e menor taxa de erro. Os arquivos de saída gerados pelo projeto detalham as predições, métricas e matrizes de confusão, permitindo uma análise completa do desempenho dos modelos.

Limitações

A implementação manual do KNN, apesar de didática, não é otimizada para grandes volumes de dados, podendo apresentar lentidão. Além disso, o KNN é sensível à escolha de *k* e à escala dos dados, o que pode impactar negativamente seu desempenho. O MLP, por sua vez, depende de uma boa escolha de hiperparâmetros e pode sofrer com overfitting em conjuntos de dados pequenos. Outra limitação é a ausência de validação cruzada, que poderia fornecer uma avaliação mais robusta dos modelos.

Considerações Finais

O projeto permitiu explorar e comparar dois métodos clássicos de classificação supervisionada: o KNN, implementado manualmente, e o MLP, utilizando a biblioteca *Scikit-learn*. A experiência evidenciou a importância da escolha do algoritmo conforme a complexidade do problema e a natureza dos dados. Enquanto o KNN se mostrou eficiente em cenários simples, o MLP apresentou desempenho superior em situações mais desafiadoras, demonstrando maior capacidade de aprendizado.

Além disso, o projeto reforçou a relevância da análise de métricas e da matriz de confusão para uma avaliação detalhada dos modelos. A implementação prática contribuiu para o entendimento dos conceitos teóricos e das limitações de cada abordagem, destacando a necessidade de ajustes e validações cuidadosas em projetos reais de inteligência artificial.