Relatorio: Avaliacao de Modelos de Classificacao (Breast Cancer e Heart Failure)

Objetivo. Implementar em Python um pipeline de preparacao de dados e treino/avaliacao de 4 algoritmos (Naive Bayes, Decision Tree, Random Forest e kNN) em 2 datasets, medindo *accuracy* e matriz de confusao para diferentes percentagens de teste (15%, 30%, 50%).

Datasets. (1) Breast Cancer Wisconsin (Diagnostic), disponibilizado no UCI e incluido no scikit-learn via *load_breast_cancer*. (2) Heart Failure Clinical Records (299 pacientes, 12 features e alvo DEATH_EVENT), disponibilizado no UCI e replicado no Kaggle.

Metodologia. Para cada dataset, efetuou-se um *train/test split* estratificado com *random_state=42*. Em seguida, treinou-se cada algoritmo sob 4 variantes de preparacao:

• labelEncoder (apenas no alvo/labels);

• labelEncoder + standardScaler (normalizacao das features numericas);

• labelEncoder + oneHotEncoder (one-hot para colunas categoricas, quando existirem);

• labelEncoder + oneHotEncoder + standardScaler (one-hot + normalizacao).

Metricas. Para cada configuracao foi calculado: accuracy, precision (weighted), recall (weighted), f1 (weighted) e a matriz de confusao.

Resumo de Resultados (template)

Preencha/atualize as tabelas abaixo com os valores exportados pelo codigo (ficheiro *results_summary.csv*).

Breast Cancer - Precision (weighted)

Preprocessamento	Naive Bayes	Decision Tree	Random Forest	kNN
labelEncoder				
labelEncoder+standardScaler				
labelEncoder+oneHotEncoder				
labelEncoder+oneHotEncoder+standardScaler				

Heart Failure - Precision (weighted)

Preprocessamento	Naive Bayes	Decision Tree	Random Forest	kNN
labelEncoder				
labelEncoder+standardScaler				
labelEncoder+oneHotEncoder				
labelEncoder+oneHotEncoder+standardScaler				

Analise Critica (guia)

Ao comparar os algoritmos, e comum observar que: (i) **kNN** tende a beneficiar bastante de **standardScaler**, porque baseia-se em distancias; (ii) **Naive Bayes** (Gaussian) e um bom baseline e por vezes competitivo em datasets bem comportados; (iii) **Random Forest** frequentemente supera uma unica **Decision Tree**, reduzindo variancia; (iv) aumentar a percentagem de teste (15% $\rightarrow$ 50%) tende a aumentar a variabilidade dos resultados, porque o conjunto de treino fica menor. No entanto, os resultados concretos dependem do dataset e do acaso do split (daí o uso de seed).

Sugestoes de discusao:

• Identificar o melhor modelo por dataset e justificar (acuracia vs. precisao).

• Comparar estabilidade entre splits (15/30/50%).

• Discutir impacto do scaler no kNN e possiveis efeitos em outros modelos.

• Comentar tamanho do dataset (Heart Failure tem apenas 299 instancias) e risco de overfitting, sobretudo em Decision Tree.

• Sugerir melhorias: validacao cruzada, tuning de hiperparametros (k, depth, n_estimators), e analise de classes desbalanceadas.

Anexo: Codigo

O codigo entregue inclui scripts para executar os experimentos e exportar tabelas e matrizes de confusao. Consulte o README.md para instrucoes.