Atividade Prática 04
Aprendizado Baseado em Instâncias

Universidade Tecnológica Federal do Paraná (UTFPR), campus Apucarana
Curso de Engenharia de Computação
Sistemas Inteligentes 1 - SICO7A
Prof. Dr. Rafael Gomes Mantovani

---

1 Objetivo da atividade

Objetivo desta atividade é avaliar diferentes implementações de kNN e DWNN em problemas reais, cujos dados podem ser encontrados em repositórios de dados públicos. Além disso, também deseja-se compreender melhor os comportamentos dos algoritmos.

2 Descrição

Use suas implementações de ambos os algoritmos, e encontre implementações disponíveis na literatura. Avalie o desempenho destes algoritmos em pelo menos dois datasets de sua escolha: um problema de classificação, e um problema de regressão. Execute os quatro algoritmos: seu kNN, kNN literatura, DWNN, DWNN da literatura, e avalie o desempenho dos mesmos.

Para a seleção de datasets, recomendo o uso do OpenML (https://www.openml.org), repositório público de datasets e experimentos com algoritmos de Aprendizado de Máquina. O OpenML tem mais de 3000 datasets (https://www.openml.org/search?type=data) que podem ser filtrados e usados livremente. Caso queiram usar algum outro conjunto de dados do UCI, Kaggle e afins, fiquem à vontade para explorá-los também.

3 O que fazer?

Com o dataset selecionado, execute alguns testes/experimentos para avaliar os algoritmos nos problemas escolhidos. Lembrem-se que definir o valor de k para o kNN, e a função de geração de pesos para o DWNN, são premissas para a construção de algoritmos adequados. Como trabalho, elabore um relatório técnico em .pdf que descreva as etapas realizadas:

- explicar os datasets escolhidos;

- mostrar algumas estatísticas interessantes dos problemas, e justificar porque eles foram escolhidos para investigação;
- explicar a implementação de kNN escolhida, seus hiperparâmetros, e características gerais de uso, e pacote necessário para fazê-la funcionar;
- a mesma coisa para o DWNN;
- descrever a metodologia experimental: como os dados foram divididos entre treino e teste, qual medida de desempenho foi usada, e quais tipos de investigações/análises foram realizadas;
- apresentar suas conclusões com relação aos resultados obtidos, e descrever algumas possíveis limitações/vantagens do algoritmo no problema escolhido.

Cada equipe deve entregar um único arquivo compactado contendo os seguintes itens:

- os arquivos fontes com com os códigos elaborados;
- um relatório (em PDF) apresentando análise e comentários/explicações sobre os resultados obtidos. Descreva e explique também partes relevantes do código implementado.
- O trabalho deve ser submetido pelo Moodle até o prazo final estabelecido na página do curso.

4 Algumas dicas

- kNN e DWNN funcionam com **datasets** que são formados totalmente por dados **numéricos**. Se eventualmente o dataset escolhido possuir algum atributo categórico, faça o pré-processamento que achar conveniente para deixá-lo com valores numéricos. Os algoritmos também não funcionarão se existirem valores ausentes, será necessário uma etapa de imputação de dados durante o pré-processamento;
- alguns autores sugerem também realizar a **normalização** dos atributos. Se achar interessante, realize testes com os dados originais e com os dados normalizados. Existe algum benefício em normalizar os dados no seu problema?
- Muitas vezes é mais fácil avaliar/treinar os algoritmos dividindo os dados em apenas duas partições: treino e teste. Chamamos esse processo de ***holdout***. Geralmente usamos 60, 70% dos dados no treinamento e o restante no teste. Embora seja simples, esse processo pode apresentar uma alta variância nos resultados. Isto é, partindo da premissa que selecionamos aleatoriamente quem são os exemplos de treino e teste, uma segunda amostragem seguindo a mesma estratégia poderia originar em resultados diferentes (melhores ou piores). Caso queiram explorar, isso é livre, vocês podem explorar a avaliação por **validação cruzada** (*k-fold cross validation*). Os principais *frameworks* de Aprendizado de Máquina (ver seção seguinte) já possuem funções implementadas que realizam esse processo. Na validação cruzada, os dados são dividido em k partições

similares, e todas elas são usadas ao menos uma vez como conjunto de teste. Ou seja, repetimos a criação do modelo (treinar o algoritmo) k vezes. O desempenho final do algoritmo é a média dos valores obtidos em todas as iterações;

- uma análise muito interessante que vocês podem explorar é executar o algoritmo com diferentes valores de k e diferentes medidas de similaridade (distância). O problema se resolve com mais ou menos vizinhos? O mesmo pode ser pensado com relação às distâncias: quais funções mostram um desempenho melhor? Distância euclidiana? Manhattan? Similaridade do Coseno?

- vocês podem apresentar e discutir esses resultados por meio de tabelas, mas o uso de figuras e **gráficos** é um diferencial :)

5 Links

- **mlr3** - *Machine Learning in R*: *framework* em R para implementação de soluções de Aprendizado de Máquina. Ele é bem completo, quase tudo já está implementado, o problema pode ser entender como encaixar cada peça. Link: https://mlr3.mlr-org.com;

- **scikit learner** - *Machine Learning in Python* - contrapartida do `mlr` mas em Python. https://scikit-learn.org/stable/

Referências

[1] LUGER, George F. Inteligência artificial. 6. ed. São Paulo, SP: Pearson Education do Brasil, 2013. xvii, 614 p. ISBN 9788581435503.

[2] RUSSELL, Stuart J.; NORVIG, Peter. Inteligência artificial. Rio de Janeiro, RJ: Elsevier, 2013. 988 p. ISBN 9788535237016.

[3] MARSLAND, Stephen. Machine Learning: An Algorithmic Perspective. 2nd edition. CRC Press, 2015.