Universidade Federal de Pernambuco - CIN

Aprendizagem de Máquina

Relatório da Lista de Exercícios 2

Feito por :
Arnaldo Rafael Morais Andrade
arma

Conteúdo

1 Objetivos

Implementar os algoritmos de redução de instância: LVQ1, LVQ2.1 e LVQ3, com variação no número de protótipos. Aplicar o classificador k-NN, com distância Euclidiana e valores de k referente ao conjunto {1,3}, tanto nos dados originais quanto nos modificados. Os experimentos devem ser realizados em dois bancos de dados distintos do repositório Promise.

2 Metodologia dos Experimentos

As implementações foram feitas utilizando a linguagem Python.

2.1 Bases de Dados

Assim como no relatório anterior, as duas bases foram retiradas do repositório Promise, sendo a primeira datatrieve.arff e a segunda cm1.arff. A base DATATRIEVE possui 9 atributos e 130 instâncias que podem ser classificadas como "0"(91.54%) ou "1"(8.46%). Já a base CM1 dispõe de 22 *features* e 498 registros, dos quais podem ter os valores "false"(90.16%) ou "true"(9.83%).

Em ambos conjuntos de dados, cada atributo foi normalizado de acordo com o método *min-max*, descrito pela equação (1).

$$x_{norm} = a * \frac{x - x_{min}}{x_{max} - x_{min}} + b \tag{1}$$

2.2 Treino

Para a fase de treino, a divisão dos conjuntos foi dada pelo método de validação cruzada *k-fold* estratificado, visto o desbalanceamento entre as classes das bases de dados. O número de *folds* escolhido foi 10.

Os parâmetros utilizados na implementação dos algoritmos LVQ foram baseados em [1], exceto o número de iterações (*Epochs*) que foi 10 ao invés de 100, por conta de seu alto custo computacional. Para a versão LVQ1, os protótipos de entrada foram retirados do conjunto de treinamento, de forma aleatória e estratificada. A saída do método anterior serviu como entrada para os demais algoritmos.

2.3 Experimentos

Os experimentos foram realizados em uma máquina virtual Ubuntu 18.04.5 LTS, com 7.8 GB de memória e 3 núcleos de CPU virtuais.

3 Resultados e Discussões

Para cada base de dados 3 figuras foram construídas. As figuras 1, 2 e 3 pertencem ao conjunto de dados DATATRIEVE, já as outras se referem à base de dados CM1.

A figura 1 agrupa as 4 diferentes estratégias de redução de instâncias em dois gráficos. No gráfico da esquerda, o eixo y, tempo de processamento em segundos, é dado pela soma do tempo de geração de protótipos, tempo de treinamento e o tempo de teste. O número de protótipos foi mantido em 10, variando apenas a quantidade de vizinhos k. A linha de cor preta, 'None' caracteriza-se pelo não uso da geração de protótipos.

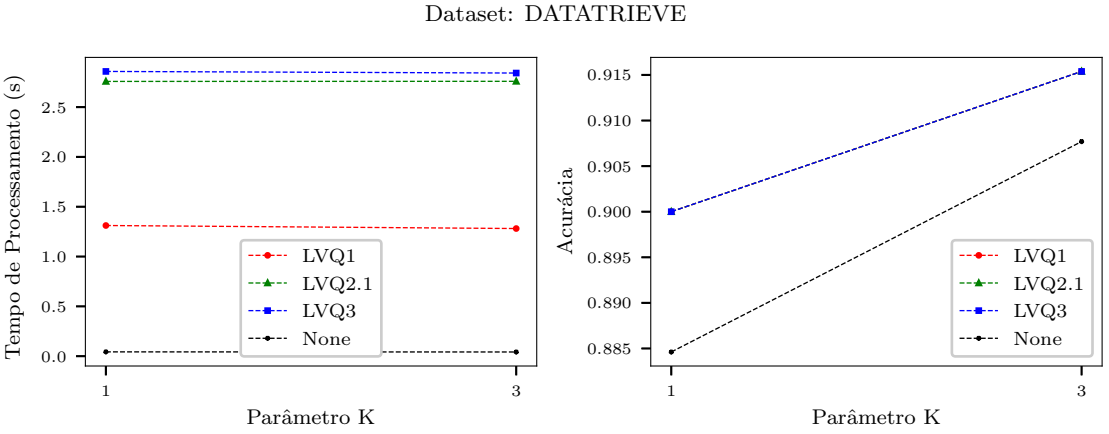

Figura 1: Tempo de processamento e acurácia - K variado, Protótipos = 10

Como os algoritmos LVQ2.1 e o LVQ3 usam o LVQ1 como entrada, era de se esperar que o tempo de processamento destes fossem superior. O gráfico da direita nos diz que apesar do número de protótipos ser baixo, os algoritmos utilizados aumentaram a acurácia média do classificador k-NN. Posteriormente discutiremos a razão disso.

A abordagem da figura 2 é semelhante, porém, desta vez fixou-se o número de vizinhos, modificando o número de protótipos gerados.

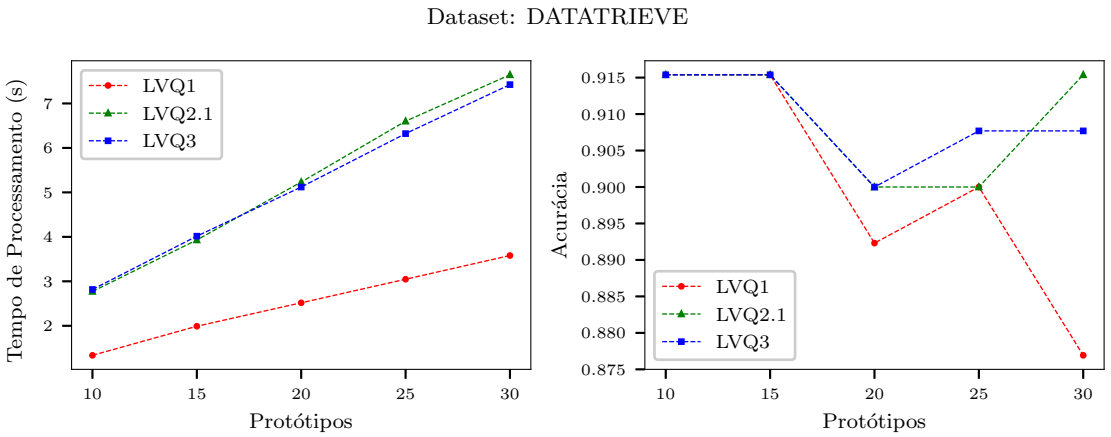

Figura 2: Tempo de processamento e acurácia - K = 3, Protótipos variados

Há uma clara relação de linearidade entre o tempo de processamento e o número de protótipos, de acordo com a imagem da esquerda. Dentre as implementações vistas, pela figura, pode-se dizer que a menos efetiva é a LVQ1. Até o número 15 de protótipos as versões se mostram equivalentes, porém ao aumentar este valor, a média das acurácias diminui.

A figura a seguir ajuda a compreender este comportamento não intuitivo. Se trata de uma matriz de confusão não normalizada para cada uma das estratégias de redução de instâncias aplicadas ao k-NN. A escolha de k foi 3 e da quantidade de protótipos foi 10. Assim como no relatório anterior, a diagonal das matrizes estão invertidas, resultando no elemento $M_{0,0}$ como o TN e o elemento $M_{1,1}$ como o TP.

Para a construção das matrizes, foi utilizado um conjunto de testes estratificado contendo 25% das instancias.

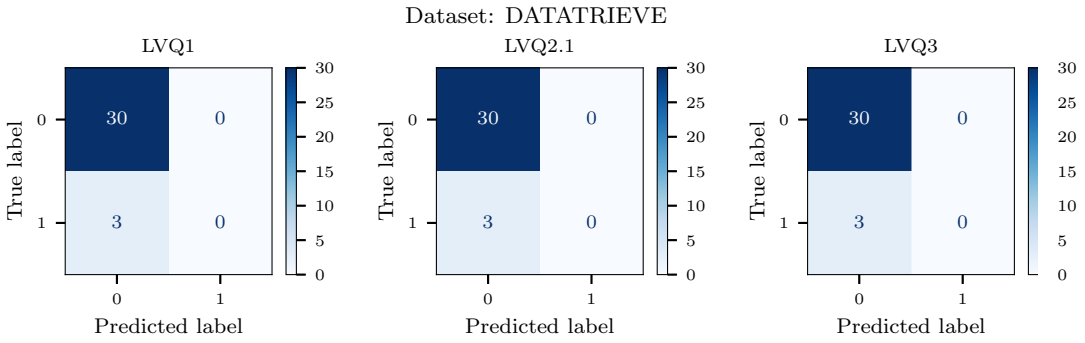

Figura 3: Matriz de confusão das 3 versões - K = 3, Protótipos = 10

Baseado nas matrizes, não há diferenças entre os 3 modelos. Para todos eles, todos os casos foram previstos como negativos, pois TP + FP = 0. Por se tratar de um problema de natureza binária e por haver um desbalanceamento entre as classes, de quase 9 ("0") para 1 ("1"), pode-se afirmar que a máquina utilizada precisa de poucos dados de treinamento, porém representativos, para atingir bons resultados.

Portanto, para este problema, um conjunto pequeno de protótipos fará o k-NN atribuir a classe "0"à todos os dados de teste.

Assim como a análise da figura 1, é mostrado a seguir, um comportamento semelhante. Apesar das versões LVQ2.1 e LVQ3 levarem mais tempo, são as que fornecem resultados mais precisos.

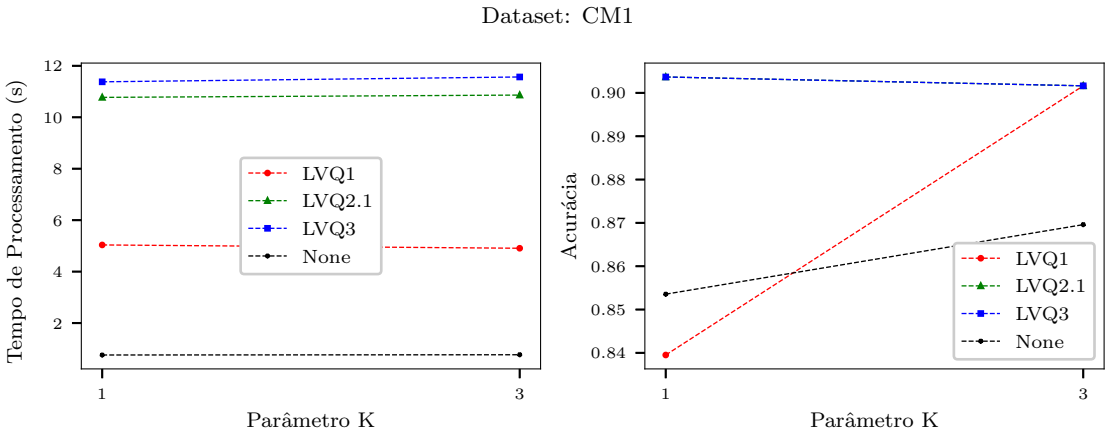

Figura 4: Tempo de processamento e acurácia - K variado, Protótipos = 10

Diferentemente da figura 1, a acurácia média resultante da não adoção do LVQ para k = 1 foi superior ao uso do LVQ1.

Nos agrupamentos da figura 5 é notório que o comportamento linear se manteve no gráfico da esquerda, apesar do tempo ter sido elevado devido à base de dados maior.

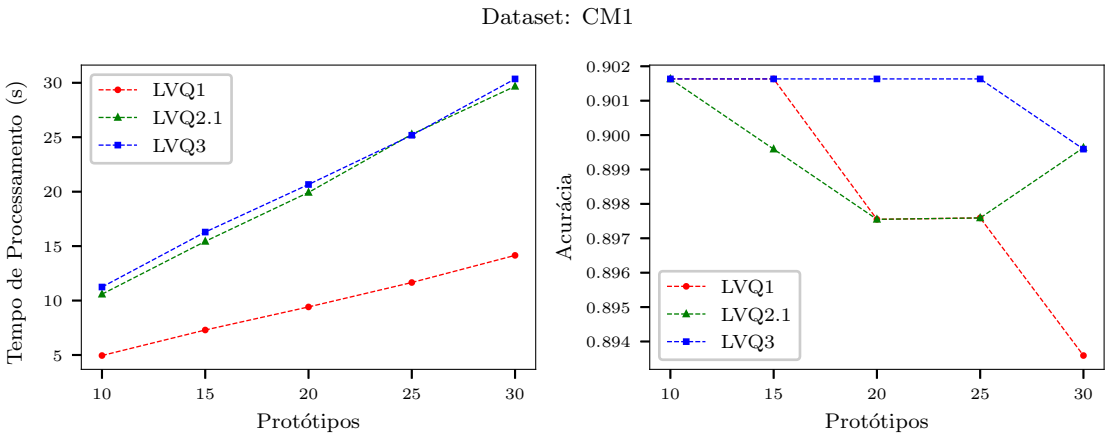

Figura 5: Tempo de processamento e acurácia - K = 3, Protótipos variados

Ainda que a implementação LVQ3 tenha sido a mais estável, a queda da acurácia média, de modo geral, também ocorreu quando o número de protótipos cresceu.

De maneira semelhante ao *dataset* anterior, neste há também um desbalanceamento entre as classes do conjunto de dados. São quase 9 ("false") para 1 ("true"), embora tenha-se mais instâncias. O que justifica a próxima figura.

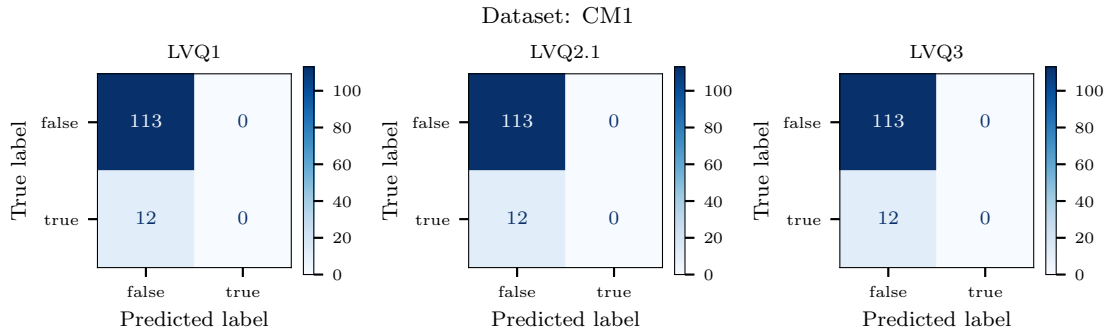

Figura 6: Matriz de confusão das 3 versões - K = 3, Protótipos = 10

Pode-se ver que o número baixo de protótipos usados para a fase de treinamento, independente da versão do LVQ, fez com que o k-NN classificasse todas as instâncias de teste como "false", obtendo assim, uma acurácia muito próxima a 90.16%, que é a proporção de registros com esta mesma classe na base de dados.

4 Conclusões

Baseado nos resultados anteriores, nota-se que as duas bases de dados são semelhantes. Portanto, a estratégia de redução de instâncias, juntamente com a escolha de um número baixo de protótipos (E.g. 10), fez com o que o classificador k-NN atingisse ótimos resultados. Vale destacar a versão LVQ3, que se mostrou superior às demais.

É importante ressaltar que ambos *datasets* se reduzem a um problema de classificação binária, ainda por cima há um alto desbalanceamento entre as classes observadas. Por isso, em um cenário do mundo real, com mais classes e mais dados, deve ser feito uma análise diferente.

Bibliografia

[1] I. Triguero, J. Derrac, S. Garcia and F. Herrera, "A Taxonomy and Experimental Study on Prototype Generation for Nearest Neighbor Classification,"in IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 42, no. 1, pp. 86-100, 2012.