Centro de Informática UFPE
Aprendizagem de Máquina – 2015.1
Aluno: Danilo Neves Ribeiro (dnr2)

Relatório - Lista de Exercício #2

1 – Metodologia dos Experimentos

Os experimentos foram realizados utilizando as seguintes bases do UCI. Foram priorizados bases de dados de atributos numéricos já que o LVQ e o k-NN utilizaram cálculo da distância euclidiana. Segue a lista das bases com seus links:

PROBLEMA 1 -
Base Iris: https://archive.ics.uci.edu/ml/datasets/Iris
Base Transfusion: https://archive.ics.uci.edu/ml/datasets/Blood+Transfusion+Service+Center

Para implementação dos algoritmos foi utilizado a lingaugem de programação em python, juntamente com algumas bibliotecas que facilitaram a leitura dos dados, manipulação de matrizes, operações matemáticas e impressão de gráficos.

As bases de dados foram aleatoriamente embaralhadas utilizando a função shuffle da biblioteca numpy. A razão entre o número de instâncias de treinamento em relação ao total foi de 70%, já o número de instâncias para teste foi de 30%. Não foi necessário separar dados para validação.

Para a geração dos protótipos utilizando o LVQ1, LVQ2.1 e LVQ3 foram utilizados os seguintes parâmetros:

- Número de Iterações: 100.
- Taxa de Aprendizado: 0.02 (Função de ajuste parametricada pelo tempo)
- Valor w para janela do LVQ2.1 e LVQ3: 0.2
- Valor do epsilon para LVQ3: 0.1

Tais valores foram escolhidos baseados em de artigos e slides. No artigo de Kohonen [1] é sugerido que se use aproximadamente 1000 iterações (valor diminuido por causa do tempo de execução do algotimo), e uma taxa de aprendizado entre 0.01 e 0.02 (foi escolhido 0.02 já que o número de iterações foi diminuido). Ele também comenta que a taxa de aprendizado deveria decrescer monoticamente, então como o número de iterações era menor que o proposto, decidi diminuir a taxa de aprendizado linearmente. Para o LVQ1 (e consequentemente LVQ2 .1 e LVQ3) os protótipos foram escolhidos aleatoriamente, utilizando também a função shuffle. O método para escolha do padrão de treinamento que iria atualizar os protótipos foi feito seguindo o slides em [2]. Para o valor w da janela, foi seguida a recomendação em [3].

2 – Resultados

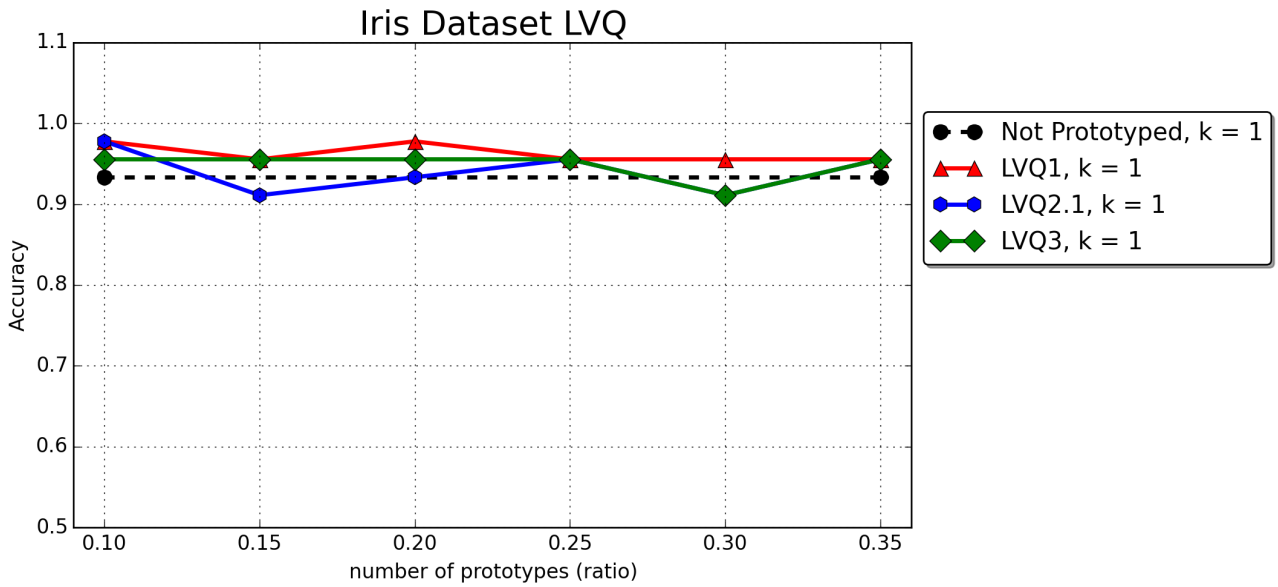
Todas as questões trataram do algoritmo k-NN (com distância euclidiana sem ponderação, ou seja, sem pesos) e da Implementação da seleção de protótipos utilizando o LVQ1, LVQ2.1 e LVQ3, variando apenas a base de dados, valores de k = {1,3}, e número de protótipos (razão ao tamanho de treinamento), que foram escolhidos entre os valores {0.10, 0.15 ,0.20, 0.25, 0.30, 0.35}. Assim será

exposto de forma agrupada a taxa de acerto (Accuracy) de acordo com a variação do número de protótipos para cada base de dados e valores k. Nos gráficos é plotado uma linha tracejada e preta para indicar a acurácia atingida pelo k-NN sem utilização de protótipos (LVQs) e utilizando todo o conjunto de treinamento.

PROBLEMA 1 – Base Iris (K-NN utilizando k = 1)

Not prototyped = 0.9333

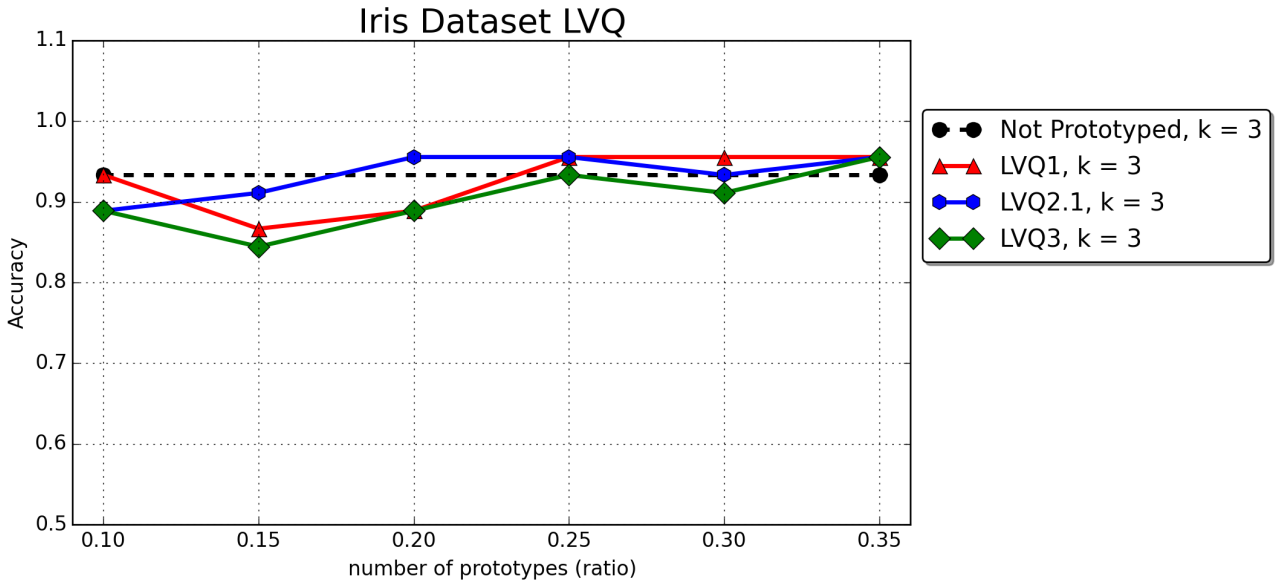
	0.1	0.15	0.2	0.25	0.3	0.35
LVQ1	0.98	0.96	0.98	0.96	0.96	0.96
LVQ2.1	0.98	0.91	0.93	0.96	0.91	0.96
LVQ3	0.96	0.96	0.96	0.96	0.91	0.96



PROBLEMA 1 – Base Iris (K-NN utilizando k = 3)

Not prototyped = 0.9333

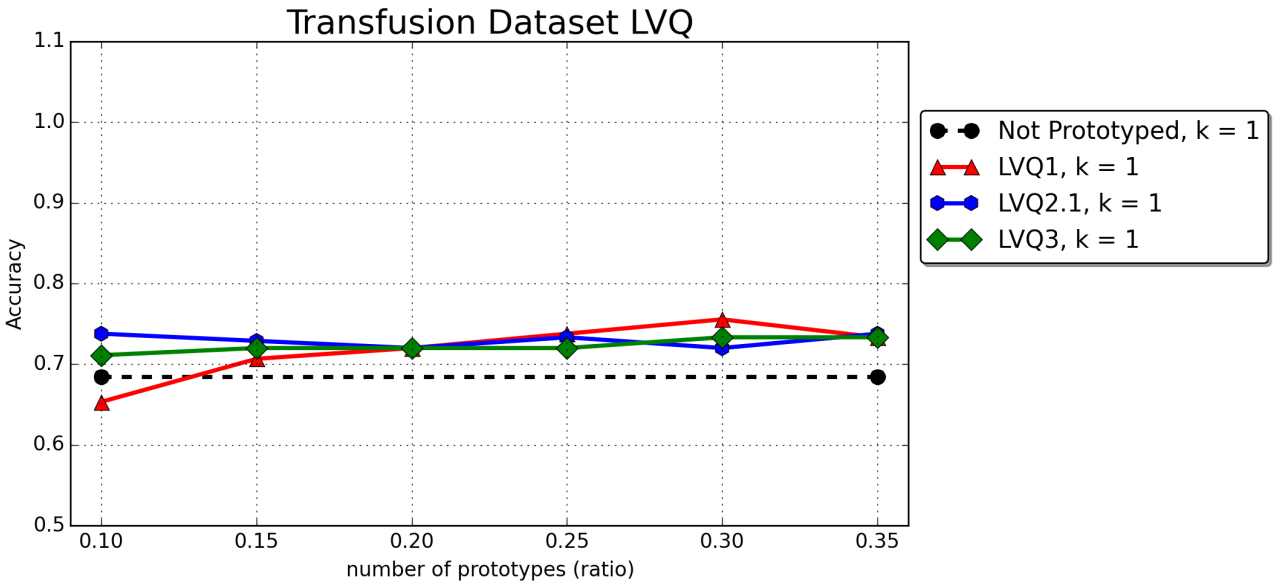
	0.1	0.15	0.2	0.25	0.3	0.35
LVQ1	0.93	0.87	0.89	0.96	0.96	0.96
LVQ2.1	0.89	0.91	0.96	0.96	0.93	0.96
LVQ3	0.89	0.84	0.89	0.93	0.91	0.96



PROBLEMA 1 – Base Transfusion (K-NN utilizando k = 1)

Not prototyped = 0.6844

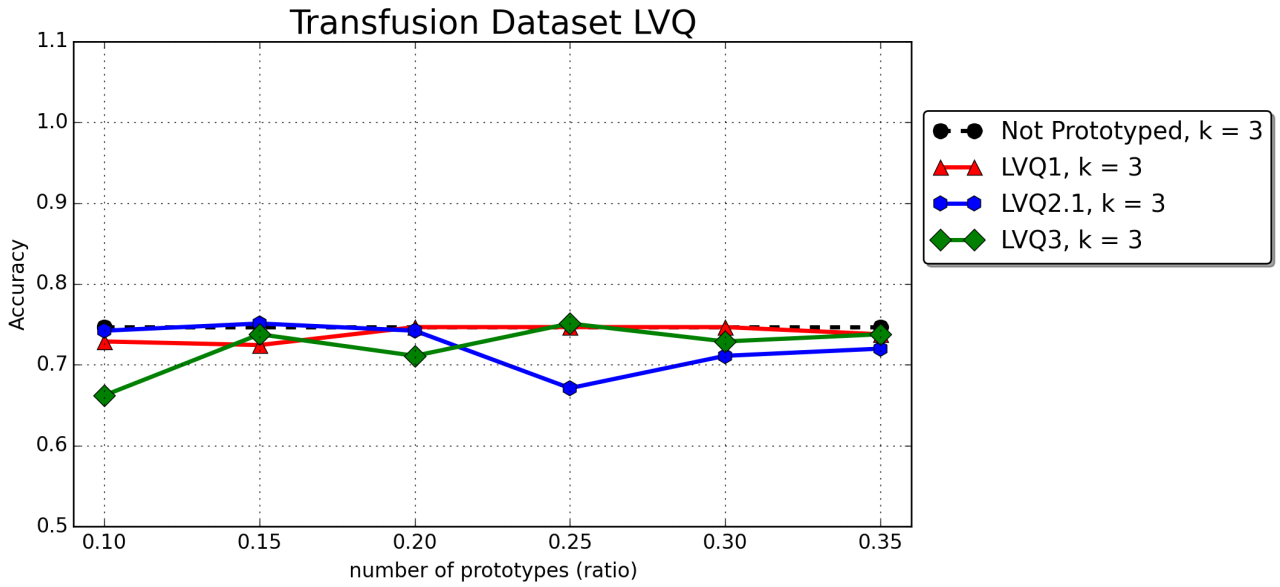
	0.1	0.15	0.2	0.25	0.3	0.35
LVQ1	0.65	0.71	0.72	0.74	0.76	0.73
LVQ2.1	0.74	0.73	0.72	0.73	0.72	0.74
LVQ3	0.71	0.72	0.72	0.72	0.73	0.73



PROBLEMA 1 – Base Transfusion (K-NN utilizando k = 3)

Not prototyped = 0.7466

	0.1	0.15	0.2	0.25	0.3	0.35
LVQ1	0.73	0.72	0.75	0.75	0.75	0.74
LVQ2.1	0.74	0.75	0.74	0.67	0.71	0.72
LVQ3	0.66	0.74	0.71	0.75	0.73	0.74



3 – Análise dos Resultados

A partir dos resultados não é possível definir claramente qual das versões do LVQ contribui mais para diminuição do erro, apesar de ser possível notar que na maioria dos *datasets* foi possível atingir um valor de acurácia tão bom ou melhor que o k-NN sem protótipos. Em geral os algortimos que utilizaram o valor k = 1 tiveram uma melhor performance que os com k = 3, talvez porque há menos instâncias de treinamento depois de gerar os protótipos do LVQ, então uma instância mais próximo significa uma melhor escolha do que considearr três valores próximos (que podem estar muito mais distantes do que o primeiro, talvez o uso de um k-NN com distância ponderada pudesse ajudar nesse aspécto) .

O número de protótipos escolhidos também não influenciou consideralvemente nos resultados, apesar que em alguns casos há alguma perda da acurácia quando o número de protótipos são pequenos (principalmente entre 10% a 20%). Isso pode ser explicado pelo fato que as bases de dados contém poucas instâncias (Base Iris por exemplo contêm 150 instâncias e 10% do conjunto de treinamento significaria apenas 10 destas instâncias) e escolher aleatoriamente os protótipos pode gerar um desbalanceamento entre o número de classes das instâncias no conjunto de protótipos escolhidos, influenciando negativamente no resultado. Talvez a utilização de outra técnica para selecionar os protítipos iniciais tivesse corrigido esse problema.

4 – Referências

[1] - Self Organizing Maps, KOHONEN, teuvo. http://www.eicstes.org/EICSTES_PDF/PAPERS/The%20Self-Organizing%20Map%20%28Kohonen%29.pdf

[2] - Learning Vector Quantization and K-Nearest Neighbor. Jia Li. http://sites.stat.psu.edu/~jiali/course/stat597e/notes2/knn.pdf

[3] - Learning Vector Quantization (LVQ) Neural Networks. http://www.mathworks.com/help/nnet/ug/learning-vector-quantization-lvq-neural-networks-1.html