UNIVERSIDADE FEDERAL DO CEARÁ

Reconhecimento de Padrões

João Rafael Barbosa de Araujo
18 de junho de 2019

Esse relatório contém as respostas para o segundo trabalho da disciplina de Reconhecimento de padrões. Além de um curto resumo das diferentes técnicas de validação e classificação utilizados no trabalho, assim como os resultados das diferentes técnicas aplicadas ao banco de dados *iris_log* e *twomoons* de acordo com o que foi solicitado nas questões.

1 PRIMEIRA QUESTÃO

Aplique o método Principal Component Analysis (PCA) na base de dados *iris_log.dat*. A seguir, aplique a rede neural ELM nos n primeiros componentes principais usando a estratégia de validação leave-one-out.

Apresente as acurácias obtidas para diferentes quantidades de componentes (n) e de neurônios ocultos (Q).

Análise de Componentes Principais (PCA) é uma técnica que busca identificar padrões nos dados colocando em evidência suas relações, similaridades e diferenças. A utilização do PCA aplica uma transformação linear nos dados de forma a agrupa-los de forma que primeira componente principal possui a maior variância, a segunda componente principal possui a segunda maior variância e assim por diante.

É possível utilizar PCA sem perda de informação, porém podem ser utilizadas apenas as primeiras N componentes como forma de enxugar a quantidade de dados utilizando apenas as primeiras **n** componentes mais relevantes.

ELM é uma rede neural feedfoward (i.e. sem realimentação) utilizada para classificação que possui apenas uma camada oculta com **Q** neurônios. Similar a MLP porém com um treinamento muito mais rápido, onde a atualização dos pesos é aprendida em um único passo. Para a função de ativação dos neurônio é tipicamente utilizada a tangente Hiperbólica, mas também pode ser utilizada uma função logística.

Abaixo uma tabela com os resultados obtidos para diferentes valores de **Q** e **n**.

		# de componentes PCA (n)		
		3	2	1
# de neurônios (Q)	1	63.33 %	50.67%	62.67 %
	3	82.00%	77.33%	62.67%
	5	90.67%	77.33%	62.67%
	7	93.33%	78.00%	62.67%
	9	94.00%	78.00%	62.67%
	11	93.33%	78.00%	62.67%
	13	93.33%	78.00%	62.67%
	15	94.00%	78.00%	62.67%

Tabela 1.1: Tabela de resultados variando Q e n

Podemos observar que utilizar valores pequenos de componentes PCA diminui a acurácia do algoritmo, uma vez que quanto menos componentes PCA, mais dados serão perdidos por não estarmos utilizando todos os dados.

Um outro resultado já esperado é que valores maiores de neurônios ocultos na rede ELM resulta em uma melhor classificação.

O código utilizado para gerar os resultados pode ser encontrado em *rp_pca_elm.py*.

2 SEGUNDA QUESTÃO

Usando o conjunto de dados 2-D disponível no arquivo *twomoons.dat*, trace a superfície de decisão obtida com a rede neural RBF treinada com todas as amostras.

Redes e de função de base radial (RBF), do inglês, Radial basis function network, são um rede neural artificial que utilizam funções de base radial como função de ativação dos neurônios. Ela possui 3 camadas com propósitos diferentes, sendo elas:

- A camada de entrada contém nós fonte que conectam a rede ao seu ambiente.
- A segunda camada, a única camada escondida, aplica uma transformação não-linear do espaço de entrada para o espaço escondido (alta dimensionalidade).
- A camada de saída é linear, fornecendo a resposta da rede ao padrão (sinal) de ativação aplicado na entrada.

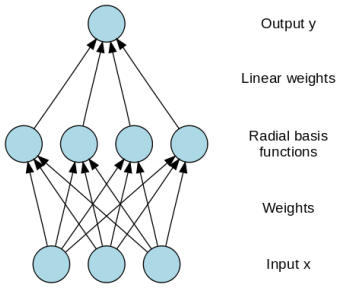

Figura 2.1: Modelo de rede RBF

Após treinar uma rede RBF com 15 neuronios, foram gerados um mapa de pontos correspondente ao universo de pontos contidos no banco de dados *twomoons*, após passar esse conjunto pela rede treinada obtivemos resultado mostrado na Figura 2.2, onde a rede neural consegue separar com sucesso as duas massas de dados.

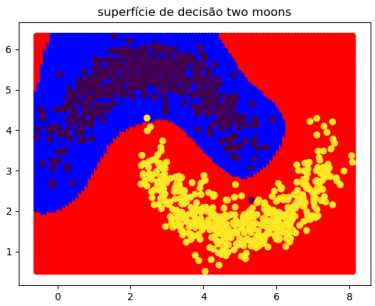

Figura 2.2: Superfície de decisão RBF

O código utilizado para gerar os resultados pode ser encontrado em *rp_rbm_elm.py*.

3 TERCEIRA QUESTÃO

Efetue o agrupamento (clustering) da base *iris_log.dat* por meio do método K-means com diferentes valores de K e apresente as larguras de silhueta correspondentes.

K-means é um algoritmo de clusterização que busca particionar um conjunto de dados em k clusters utilizando centroides inicialmente posicionados de forma aleatória no universo de dados que vão se deslocando com o objetivo de diminuir a soma das distancias dos centroides com os dados. Esse algoritmo tem como resultado a divisão dos dados em **k** clusters (um cluster para cada centroide).

Como uma medida de qualidade da separação de clusters é utilizada a medida de largura média de silhueta. Essa medida foi utilizada para avaliar a o desempenho do algoritmo k-means na base *iris_log.dat*.

Foram realizados 8 experimentos e a média dos valores de silhueta de acordo com o número de centroides está representada na Figura 3.1

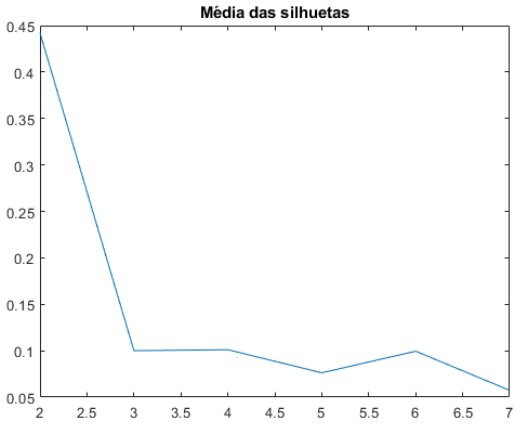

Figura 3.1: TEXTO TEXTO *leave-one-out*

O código utilizado para gerar os resultados pode ser encontrado em *rp_kmeans.py*.

4 CÓDIGO

Nessa seção será apresentada informações sobre como foram feitos os algoritmos e também os resultados desses algoritmos aplicados nas bases de dados para cada questão.

4.1 PROGRAMAÇÃO

A linguagem de programação utilizada para resolver o problema foi *Python 3.7*. Todas as técnicas utilizadas para resolver as questões foram implementadas. Foi utilizado *numpy* para

auxiliar na aplicação das técnicas discutidas na seções 1 2 e 3, e também no tratamento de dados.

5 CONCLUSÃO

Nesse relatório foram utilizados diferentes métodos de classificação e clusterização. As técnicas foram: RBF, ELM e k-means.

A aplicação das técnicas mostraram um bom desempenho dentro das questões fazendo uma boa classificação/clusterização como foi mostrado nas Sessões anteriores.

Todo o código desenvolvido para esse trabalho pode ser encontrado em `https://github.com/faellacurcio/rp`.