UNIVERSIDADE DE SÃO PAULO - ESCOLA DE ARTES, CIÊNCIAS E HUMANIDADES

Trabalho de Mineração de Dados

Arthur Patrício Grava, Lucas Fernandes Brunialti

14 de janeiro de 2014

0 VISUALIZAÇÃO

Primeiramente, para a realização desse trabalho, foi feita uma análise dos conjuntos de dados disponibilizados para o trabalho (no caso, os dois conjuntos foram usados). Foi feita a tentativa de visualizar os conjuntos de dados como imagens em escala de preto e branco, variando o tamanho das imagens e como sinais. No entanto, não foi obtido sucesso, então, tentou-se visualizar os conjuntos de dados pela Redução de Dimensionalidade, assim, foram obtidas algumas conclusões que levaram a escolher as técnicas usadas para resolver os conjuntos de dados.

1 IMPLEMENTAÇÃO

Para a resolução dos conjuntos de dados, foram escolhidas as seguintes técnicas: PCA (*Principal Component Analysis*) para a Redução de Dimensionalidade; Redes Neurais Multicamadas para resolução do conjunto de classificação, DBSCAN (*Density-Based Spacial Clustering of Application with Noise*) para a resolução do conjunto de agrupamento. Todas as implementações foram feitas utilizando as linguagens MATLAB e Java. Todos os códigos desenvolvidos podem ser encontrados em: https://github.com/lucasbrunialti/DataMiningProject.

1.1 PCA

Para o PCA foi utilizada a linguagem MATLAB, sendo a sua implementação baseada nos slides vistos em aula para melhor compreensão do cálculo e nas notas de aula [1] para implemetação

[1] http://math.hanyang.ac.kr/hjang/NLA/lecture23.pdf

do algoritmo QR simples. O algoritmo QR foi utilizado para o cálculo dos autovalores da matriz de covariância, o qual recebe um erro, que determina o critério de parada do algoritmo, ou seja, se os autovetores encontrados na iteração i menos os encontrados na iteração $i-1$ for menor que um erro ϵ, significa que o algoritmo convergiu. Com os autovalores, foram calculados os autovetores por decomposição LU, extraindo apenas a matriz U (*Upper*) sendo resolvida de baixo para cima (canto inferior), lembrando que o sistema é indeterminado, então, o valor do último índice do autovetor é igualado a 1, para então resolver os outros índices.

1.2 Redes Neurais Multicamadas

Foi desenvolvida uma Rede Neural Multicamada *feedforward* usando como algoritmo de otimização o *(*Stochastic Batch Gradient Descent with Momentum and Adaptive Learning Rate). A implementação foi baseada nas notas de aula do cursou de *Machine Learning Coursera Stanford University*[2], no livro *Neural Networks: Acomprehensive Foundation*[3] e no artigo de um dos autores[4]

1.3 DBSCAN

O DBSCAN foi desenvolvido utilizando-se as linguagens Java, para a implementação do algoritmo em si, e MATLAB, utilizado na implementação da técnica de validação *Davies-Bouldin Index* (DB), baseando-se nos slides e no artigo de DBSCAN[5], ambos vistos em aula. Utilizou-se a heurística descrita no artigo para determinar parâmetros ótimos: raio de vizinhança (R) e número mínimo de pontos para se caracterizar um cluster($MinPoints$).

2 Pré-Processamento

2.1 Conjunto de Classificação

Para o pré-processamento do Conjunto de Dados de Classificação foi realizada uma normalização nos dados (*Whitening*) e um parâmetro α dentro da função sigmóide, colocando os dados mais próximos do centro da sigmóide. Além disso, foram realizados testes com o PCA.

2.2 Conjunto de Agrupamento

Para o conjunto de agrupamento foi utilizado o PCA, como técnica de pré-processamento, para a redução da dimensionalidade dos dados, a fim de melhor utilizar a memória disponível para a resolução do problema, além de melhorar o desempenho em nível de processamento do algoritmo, sem prejudicar a capacidade de caracterização do conjunto.

[2] https://class.coursera.org/ml-004

[3] S. Haykin, Neural Networks: A Comprehensive Foundation, 2nd Edition, Prentice-Hall, 1999

[4] APRENDIZADO POR TRANSFERÊNCIA PARA APLICACÕES ORIENTADAs À USUÁRIO: UMA EXPERIÊNCIA EM LÍNGUA DE SINAIS

[5] http://dns2.icar.cnr.it/manco/Teaching/2005/datamining/articoli/KDD-96.final.frame.pdf

3 Divisão dos Conjuntos

3.1 Conjunto de Classificação

No conjunto de classificação o conjunto foi dividido em três partes: treinamento (250 instâncias), teste (9875 instâncias) e validação (9875) instâncias.

3.2 Conjunto de Agrupamento

Foi extraído metade do conjunto de agrupamento - equivalente a 10.000 instâncias - para se poder utilizar a heurística de otimização dos parâmetros R e $MinPoints$ pois ao se tentar utilizar o conjunto inteiro de dados foram encontrados problemas de utilização de memória, pois a memória não foi suficiente para se criar uma matriz $M_{20000x20000}$ de números com ponto flutuante para a realização dos cálculos necessários.

4 Resultados

4.1 PCA

Com o PCA foi realizada visualização dos dados para o conjunto de classificação e agrupamento. Para o conjunto de agrupamento foi possível perceber que são dados densos e com muitos outliers, por isso, foi tomada a decisão de uma técnica que se baseia em densidade e que tem a capacidade de detectar outliers. As Figuras 4.1 e 4.1 mostram a visualização realizada.

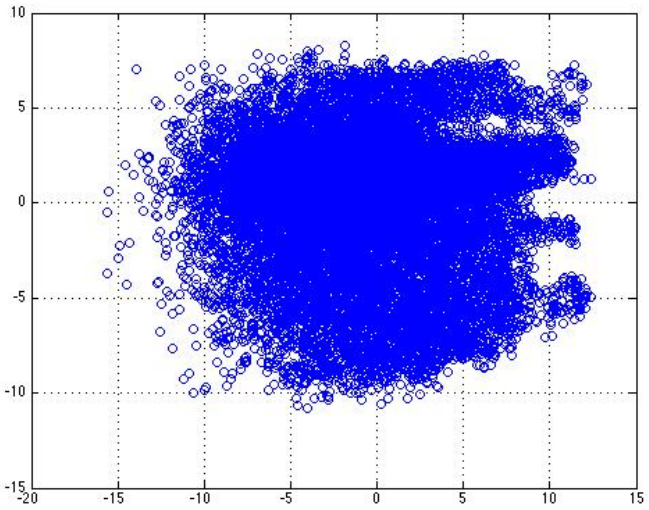

Figura 4.1: Representação em 2D do conjunto de agrupmento

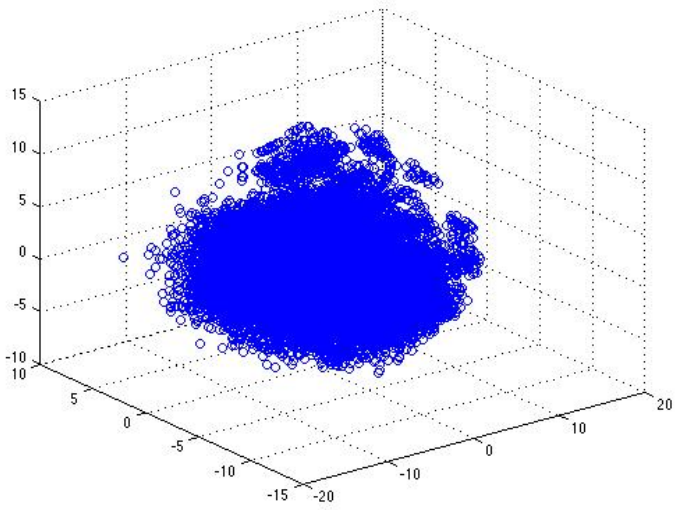

Figura 4.2: Representação do conjunto de agrupamento em 3D

A visualização com o PCA também foi utilizada nesse conjunto de dados, como mostrado na Figura 4.3, porém, não foi possível tomar alguma decisão ou inferir algo sobre a representação obtida.

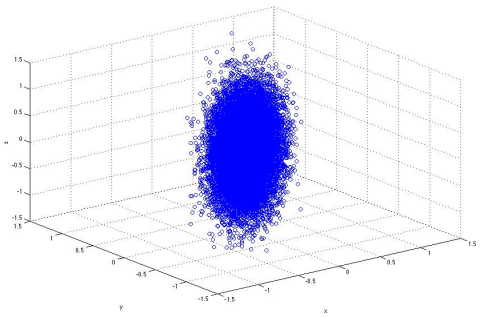

Figura 4.3: Representação em 3D do conjunto de dados de classificação

4.2 Redes Neurais Multicamadas

Com a Rede Neural Multicamada foi resolvido o conjunto de classificação. O algoritmo de otimização utilizado foi o *Stochastic Batch Gradient Descent with momentum and adaptive learning rate*. Foram realizados testes variando o número de neurônios na camada escondida e verificando a métrica F1-score da época com melhor performance no conjunto de teste, assim como mostra a Tabela 4.1.

A Tabela 4.1 apresenta a precisão e F1 score médios para diferentes configurações de neurônios na camada escondida, para o cálculo da média foram executados 5 vezes para cada configuração de neurônios na camada escondida. Os parâmetros momentum, acréscimo e decréscimo da taxa de aprendizado e taxa de aprendizado inicial foram determinadas segundo os padrões na *Toolbox NN Matlab*[6]. Como critério de parada, foram usados o número de épocas, sendo 500, que foi determinado por observação, ou seja, percebia-se que após o overfitting (em média 250 épocas), o MSE dos conjuntos de teste e validação apenas subiam.

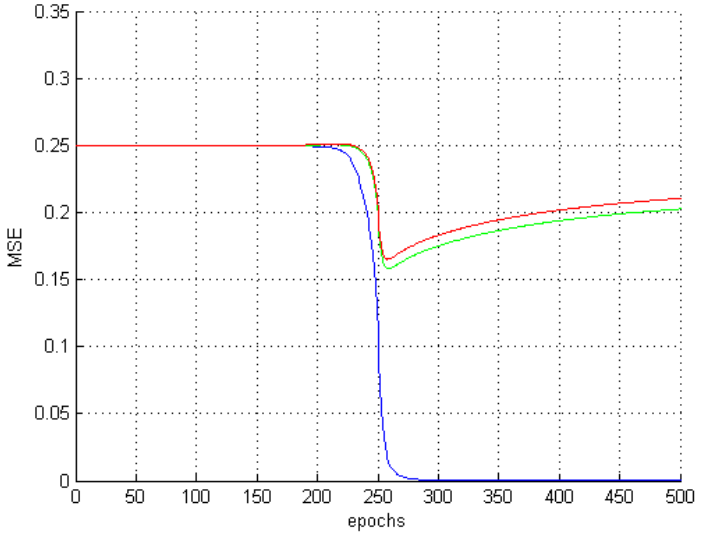

Figura 4.4: Treinamento do erro da Rede Neural com 60 neurônios.

# Neurônios na camada escondida	Precisão média	F1 score médio
10	0.7701	0.7806
60	0.7703	0.7808
110	0.7697	0.7808
160	0.7695	0.7807
210	0.7699	0.7808
260	0.7696	0.7808

Tabela 4.1: Resultado para as execuções de treinamento da Rede Neural Multicamadas.

Analisando a Tabela 4.1 é possível ver que as diferentes configurações não mudam muito a performance e o F1 score, isso ocorre por causa do overfitting, ou seja, se fossem disponibilizados mais dados para treinamento, a técnica de otimização minimizaria mais o MSE dos 3 conjuntos.

[6]http://www.mathworks.com/help/nnet/ref/traingdx.html

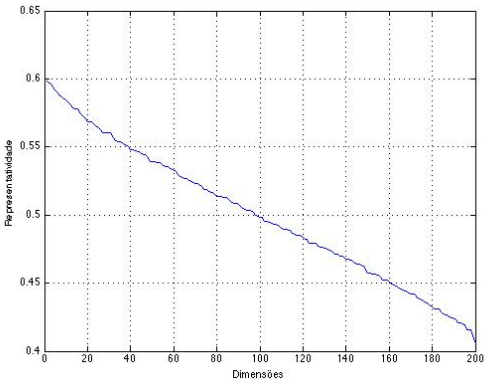

Figura 4.5: Porção de Auto Valor para cada dimensão para o Conjunto de Classificação

Uma das execuções do treinamento, com 60 neurônios, é mostrada na Figura **??**, sendo as curvas azul, vermelha e verde o *Mean Squared Error* dos conjuntos de treino, teste e validação, respectivamente. É fácil ver que a partir da época 200 o aprendizado esta sofrendo apenas *overfitting*, pois enquanto o *MSE* nos conjuntos de teste e validação aumentam, o *MSE* no conjunto de de teste diminui, aumentando a distância das curvas.
Também foi considerado o PCA para a aplicação nesse conjunto de dados, porém, pelo gráfico apresentado na Figura 4.5 foi possível perceber que o PCA não traria muitos benefícios, pois cada dimensão explica relativamente muito a variação no conjunto de dados. Mesmo assim, foi realizado um teste: com a melhor configuração dos testes apresentados na Tabela 4.1, a arquitetura com 260 neurônios (considerando a Precisão), foi usada para resolver o conjunto de dados reduzido, sendo considerado 180 dimensões (̃90% da variação). O resultado foi como esperado, foi obtido 0.7777 de precisão, e 0.7808 de F1 score, não muito diferente do que foi apresentado na Tabela 4.1.
Ainda, foi usado o processo de *Whitening* para centralização, e um parâmetro α dentro da função sigmóide, sendo possível melhor aprendizado, aproximando os valores para o centro da sigmóide nas primeiras iterações.
A curva ROC, apresentada na Figura 4.6, é utilizada para determinar o melhor limiar para ser utilizado na saída da rede. A curva também foi gerada com a melhor configuração apresentada na Tabela 4.1, assim como o teste com o PCA.
O melhor cenário é quando a curva apresenta valores altos de especificidade e sensibilidade, ou seja, não comete muitos falsos negativos e falsos positivos, respectivamente. Dependendo do conjunto de dados que se possui, é possível transitar entre os valores de sensibilidade e especificidade, de maneira que melhor atenda ao conjunto de dados, como neste caso não se possui informações sobre o conjunto de dados que ofereçam um embasamento para esta prática, o melhor limiar é aquele que apresenta uma taxa baixa de falsos positivos e uma taxa alta de verdadeiros positivos, assim, uma possível configuração que balanceia esses dois parâmetros é o limiar que apresentou taxa de falsos positivos de $\simeq 0.25$ e taxa de verdadeiros positivos de $\simeq 0.81$.

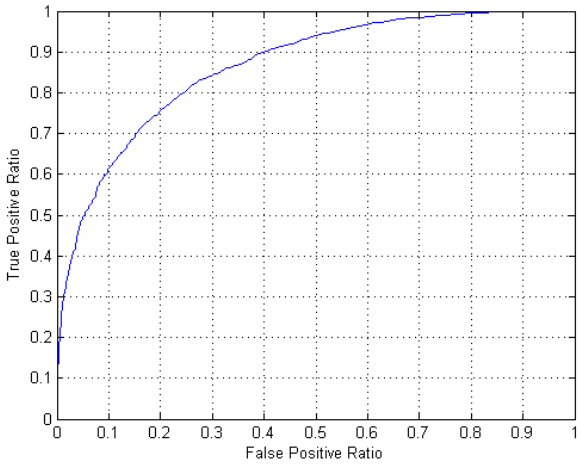

Figura 4.6: Representação da ROC seguindo a melhor arquitetura

4.3 DBSCAN

O DBSCAN foi a técnica utilizada para o agrupamento do conjunto de dados. Primeiramente foi aplicada a heurística do artigo para identificar o R e $MinPoints$ ótimos para o conjunto de dados, a partir disso foi obtido o gráfico da Figura4.7 e, de acordo com o gráfico, o melhor R para o conjunto é o de valor 4, onde, no ponto 9000 do eixo x começam a aparecer os ruídos nos dados, por estarem mais separados do conjunto de dados.

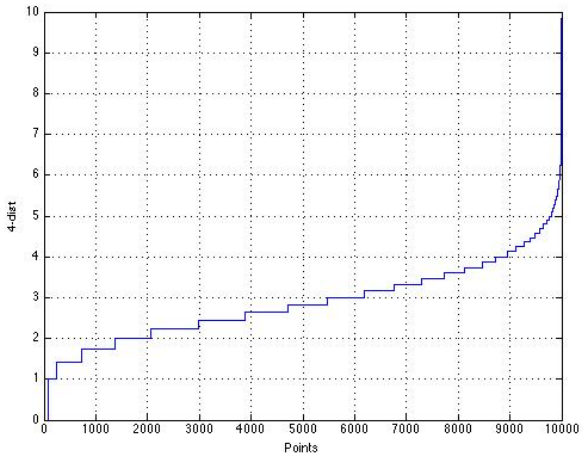

Figura 4.7: Gráfico gerado a partir da Heurística estudada no artigo de DBSCAN

16-Dimensões				10-Dimensões			
Clusters	Raio	MinPts	DB	Clusters	Raio	MinPts	DB
437	3.0	2	717.35	99	3.0	2	691.11
322	3.0	3	468.29	65	3.0	3	446.06
244	3.0	4	471.92	63	3.0	4	450.52
200	3.0	5	461.70	61	3.0	5	438.76
140	3.5	2	135.45	18	3.5	2	129.39
103	3.5	3	143.28	18	3.5	3	134.68
99	3.5	4	94.61	12	3.5	4	87.64
94	3.5	5	109.72	14	3.5	5	103.83
39	4.0	2	48.84	6	4.0	2	44.79
33	4.0	3	42.44	5	4.0	3	38.75
31	4.0	4	30.50	3	4.0	4	27.41
27	4.0	5	1.54	2	4.0	5	2.20
13	4.5	2	0.00	1	4.5	2	0.00
9	4.5	3	0.60	2	4.5	3	0.90
7	4.5	4	0.00	1	4.5	4	0.00
11	4.5	5	0.00	1	4.5	5	0.00

Tabela 4.2: Da obtido após a execução do DBSCAN com diferentes parâmetros de R e $MinPoints$.

A partir disso os parâmetros foram variados para selecionar a melhor configuração possível que represente o conjunto, como pode ser observado na Tabela4.2. Foram realizados testes em dados com 16 dimensões e com 10 dimensões, reduzidos pelo PCA implementado neste trabalho. Foi escolhido 10 dimensões para a redução pois, com esa redução, ainda se pode representar 90% dos dados, como pode ser observado na Figura4.8 e na Figura4.9. A partir desses dados e dos testes, construiu-se um gráfico representando a variação do valor de DB para os parâmetros de resultado obtido na métrica Davies-Boulin Index para todas as configurações testadas, representado na Figura4.10, onde pode-se observar que com os mesmos parâmetros os valores obtidos para a métrica são parecidos, mostrando que a redução de dimensão não alterou as condições de distância do conjunto, mesmo que a quantidade de grupos encontrada seja diferente.

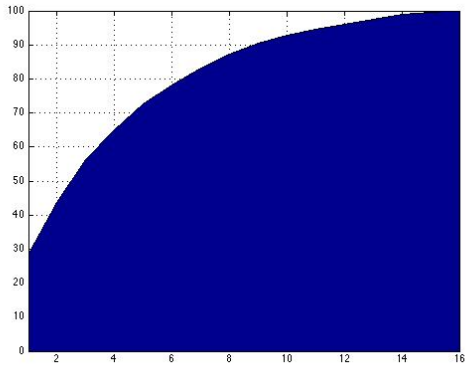

Figura 4.8: Porção de Auto Valor Acumulativo para cada dimensão [Conjunto de agrupamento]

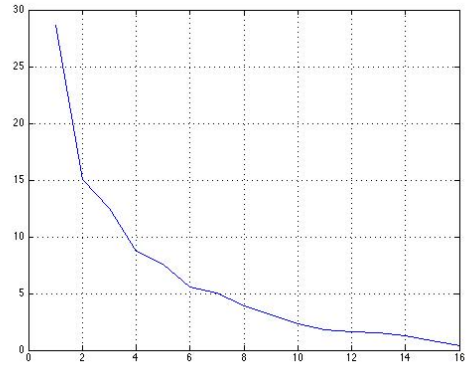

Figura 4.9: Porção de Auto Valor para cada dimensão [Conjunto de agrupamento]

A partir dos resultados e do conhecimento prévio sobre o conjunto obtido durante a especificação do trabalho - foi informado que o conjunto de agrupamento possuía uma numerosa quantidade de grupos, escolheu-se como parâmetros ótimos $R = 4.5$ e $MinPts = 5$ no conjunto original, ou seja, com 16 dimensões, pois este apresenta um índice DB baixo e com a existência de 11 grupos distintos.

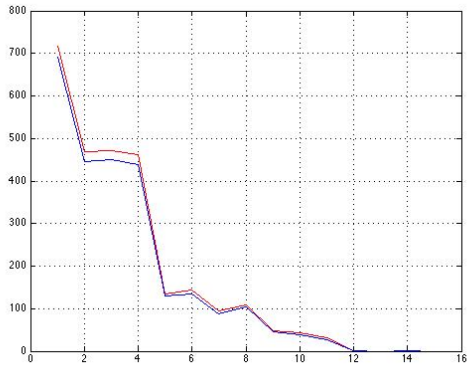

Figura 4.10: DB index para os diferentes números de clusters formados, a linha vermelha representa os dados em 16 dimensões, enquanto a azul o conjunto de 10 dimensões.