Terceiro Trabalho Prático

Luis Gustavo Lorgus Decker
209819
luisgustavo.decker@gmail.com

Luiz Antonio Falaguasta Barbosa
075882
lafbarbosa@gmail.com

I. Introdução

Neste trabalho, tentaremos separar os 19924 documentos oferecidos, cada um deles sendo um *post* em *Newsgroups* da Usenet. Tentamos separar estes documentos em clusters, mirando no objetivo de ter documentos semelhantes no mesmo cluster.

Para realizar esta separação utilizamos o algorítmo K-Means como algorítimo de clusterização, PCA como método redutor de dimensionalidade e a medida de silhueta como métrica avaliadora dos clusters obtidos.

II. Soluções Propostas

A. Algoritmo k-means

O algoritmo k-means desenvolvido por MacQueen é uma técnica de aprendizado não-supervisionado, ou seja, não se sabe previamente os conjuntos nos quais os dados se devidem. Assim, é necessário fazer a segmentação dos dados para descobrir a similaridade existente entre eles para poder serpará-los em grupos. O número de grupos é definido pelo usuário e deve ser fornecido como parâmetro para o algorítimo.

Sua execução inicia-se por meio da escolha de k elementos que formam os centróides; esses centróides podem ser escolhidos aleatoriamente (no scikit-learn, essa escolha é feita de maneira aleatória, com probabilidade proporcional à distância quadrática dos centros já escolhidos). Depois calcula-se a distância de cada elemento em relação a essas sementes, agrupando-se os elementos ao grupo que possuir a menor distância (maior similaridade) e recalculando o centróide do mesmo. Isso é repetido até que cada elemento faça parte de um cluster.

Feito isso, é calculada a distância média de todos os pontos dentro do cluster, muda-se a posição do centróide para o novo ponto que foi calculado, correspondendo à distância média de todos os pontos relacionados àquele centróide. Tal mudança de posição pode fazer com que mudem os pontos que pertencem àquele cluster. Isso se repete até que nenhum ponto mude mais de cluster; fato que ocorre quando os centróides não se mover mais devido a já estarem na posição central da distância entre os pontos.

B. Análise de Componentes Principais (PCA)

A análise de componentes principais (PCA) é uma das técnicas mais utilizadas como algorítimo de redução de dimensionalidade. Seu funcionamento se resume em encontrar o hiperplano que esteja mais próximo dos dados, e então projetar os mesmos neste hiperplano. O primeiro passo na tarefa de redução de dimensionalidade é encontrar o hiperplano correto para projetar os dados. Este processo se resume em encontrar o hiperplano cuja variância entre os dados se preserve ao máximo, permitindo assim uma menor perda de informação em relação a outras projeções.

O PCA identifica o eixo que preserva a maior variância no conjunto de dados. Após isso, encontra outro eixo, ortogonal ao primeiro, que preserve ao máximo o restante da variância dos dados, e repete este processo até que tenha obtido o mesmo número de vetores quanto dimensões houver nos dados.Cada um destes vetores é, ordenadamente, um componente principal.

Neste trabalho, utilizamos a técnica de *Decomposição em Valores Singulares*(SVD),já implementada no algorítimo de PCA do scikit-learn, uma técnica que é capaz de decompor uma matriz X que representa os dados de treinamento em três matrizes, $V^T, \Sigma\, e\, U$, onde V^T contém os componentes principais.

Visando reduzir a dimensionalidade, temos que escolher um número k de componentes para gerar o hiperplano em que projetaremos os dados. Fazemos esta escolha da seguinte maneira: sendo $\alpha \in [0, 1]$ o valor de porcentagem da variância que desejamos manter, escolhemos os k primeiros componentes tal que a soma da variância associada a estes componentes represente $\alpha\%$ da variância total, ou seja, partindo de $k = 1$, aumentamos k até que

$$\frac{\sum_{i=1}^{k} \Sigma_i}{\sum_{i=1}^{m} \Sigma_i} >= \alpha$$

, sendo que Σ_i é a variância associada ao iésimo componente principal, encontrada na matriz Σ retornada pelo SVD.

Encontrado um valor de k que satisfaça a restrição de α, criamos um hiperplano em $\mathbb{R}^k$ com os k primeiros componentes principais, e projetamos os dados de treino neste hiperplano. Estes dados projetados estarão sendo utilizados como entrada do algorítimo k-means ao invés dos dados originais em alguns testes.

C. Avaliação da Silhueta

O método da silhueta serve para verificar e validar a consistência de clusters de dados. É um valor que mede quão similar um dado é em relação a seu cluster comparado com outros clusters.

A silhueta varia no intervalo $[-1, 1]$, onde um valor mais alto evidencia que o objeto está bem coeso com seu próprio cluster e separado de outros clusters. Se vários objetos tem um valor alto, o que podemos verificar com a média da silhueta, então a configuração de clusterização (número de clusters) está correta. Se muitos pontos tem valores pequenos ou negativos, então a configuração de clusterização pode ter clusters demais ou muito poucos.

Tendo um conjunto de dados já clusterizados em k clusters, para cada dado x: $a(x)$ é a dissimilaridade média com todos os outros dados do mesmo cluster (quão bem ajustado x está no seu cluster), ou seja, a média da distância de x com todos os pontos do seu cluster, $b(x)$ é a menor dissimilaridade média de x com qualquer outro cluster que não o de x. Logo, a silhueta de x , $s(x)$ é definida por

$$s(x) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

III. Experimentos e Discussão

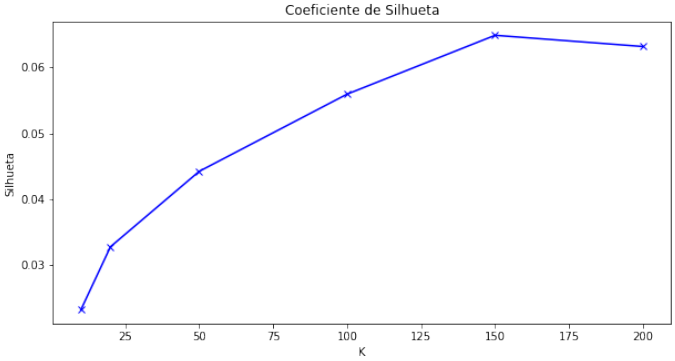
Primeiro, executamos o algorítmo de k-means nas informações provenientes do bag-of-words dos documentos. Como este algorítimo exige que o número de classes seja dado como parâmetro, testamos com vários valores visando verificar a eficácia. Para averiguar os resultados, utilizamos a informação do campo *Newsgroups* dos documentos. Estes metadados nos dão uma boa noção sobre o assunto do documento, e é esperado que documentos com assuntos semelhantes, de *Newsgroups* semelhantes ou iguais sejam agrupados juntos.

Para visualizar estes dados, na convergência do k-means analisamos os documentos pertinentes a cada cluster e então salvamos um arquivo contendo os *Newsgroups* do medóide de cada cluster e de seus 10 vizinhos mais próximos, obtidos através do algorítimo de KNN, e também a contagem de ocorrências de cada *Newsgroups* dos documentos do cluster.

Utilizamos também a média da medida de silhueta aplicada ao resultado de cada teste, visando por meio deste e da confirmação através do arquivo gerado com informações de *Newsgroups* encontrar o melhor K para nosso problema.

No nosso primeiro teste, utilizamos valores de k de 10, 20, 50, 100, 150 e 200 clusters, e então analisamos os resultados.

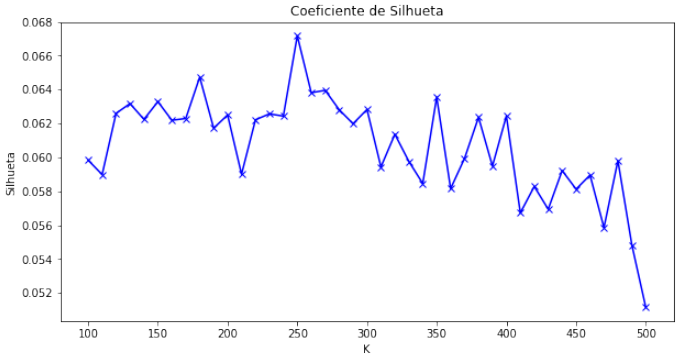
Figure 1. Resultado das medidas de silhueta no primeiro teste



Analisando os valores de silhueta obtidos, percebemos que a média de silhueta crescia com o k teve um máximo em $k = 150$, e um pequeno descréscimo após em $k = 200$. Ao analisarmos os dados provenientes dos arquivos de metadados criados, percebemos que as informações neles estavam condizentes com os valores de silhueta obtidos: grande parte dos clusters apresentava medóides e seus vizinhos com temas semelhantes, e uma maior contagem de *Newsgroups* com temas relacionados ao do medóide, porém com muitos outros *Newsgroups* não-correlacionados juntos no cluster. Isto explica o valor baixo obtido na silhueta(0.065 para $k = 150$), já que um valor de silhueta próximo de 0 aponta dados nos limites de decisão dos clusters.

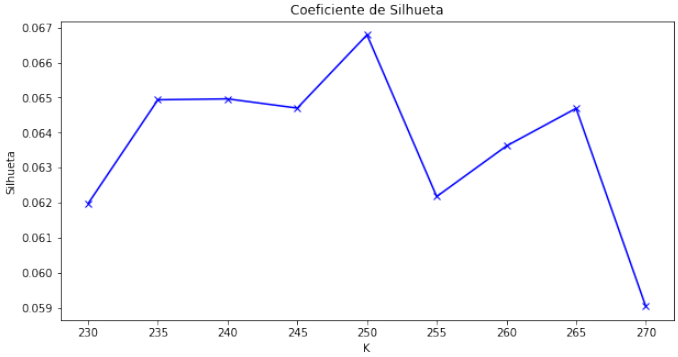
Para então refinar a busca pelo k, executamos uma segunda bateria de testes, variando o k entre 100 e 500, com passos de 10. Obtivemos um valor máximo de média de silhueta de 0.067 em $k = 250$ e um decrescimento após. Como estes testes estão mais fragmentados, não realizamos a análise manual nos clusters utilizando as informações de *Newsgroups*.

Figure 2. Resultado das medidas de silhueta no segundo teste



Visando confirmar nossos resultados e refinar ainda mais o nosso número ótimo de clusters, executamos nossa terceira bateria de testes, desta vez variando o k entre 230 e 270, com passos de 5. Novamente, obtivemos um valor máximo de média de silhueta de 0.067 em $k = 250$, que foi nosso melhor resultado até agora.

Figure 3. Resultado das medidas de silhueta no segundo teste



Após os testes usando os dados brutos, realizamos testes semelhantes utilizando o algorítmo PCA para redução de dimensionalidade dos dados. Experimentamos manter 95%, 90% e 80% da variância dos dados e analisar o resultado das

medidas de silhueta e dos arquivos de metadados para cada uma destas reduções. Para 95% obtivemos uma redução para 1601 dimensões, para 90% 1274 dimensões e para 80%, 849 dimensões

Para cada valor de variância preservada, executamos uma bateria de testes variando o valor de k entre 100 e 300, com intervalos de 50.

Mantendo 95% da variância dos dados, obtivemos um valor médio de silhueta de $0,071$ para $k = 150$. Observando o arquivo de *Newsgroups* deste resultado, observamos que a maioria dos clusters tinha um assunto central que predominava nas contagens, e outros assuntos não-relacionados, numa espécie de ruído, o que condiz com o valor de silhueta encontrado. Com 90% da variância mantida, o valor médio de silhueta foi máximo com 0.072 em $k = 250$, e com 80%, obtemos o máximo de silhueta em 0.085 com $k = 250$.

Figure 4. Resultado das medidas de silhueta para 95% da variância

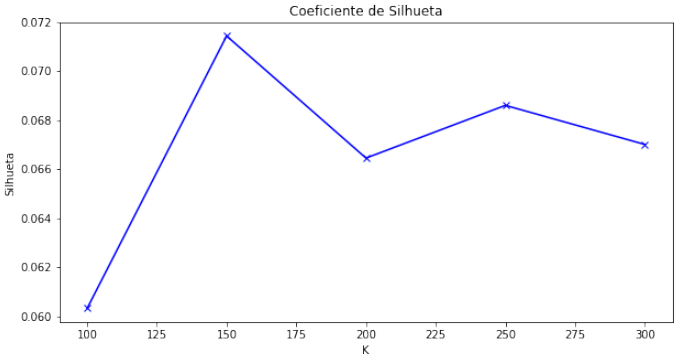


Figure 5. Resultado das medidas de silhueta para 90% da variância

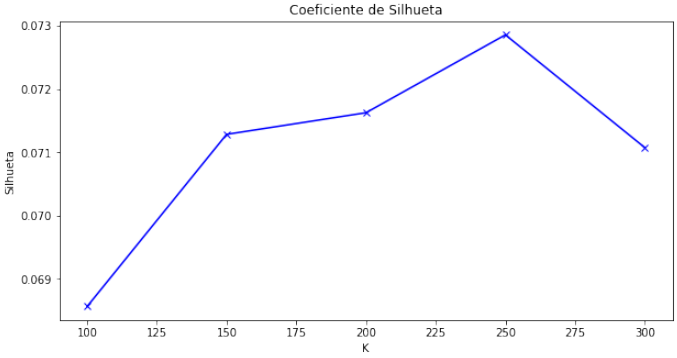
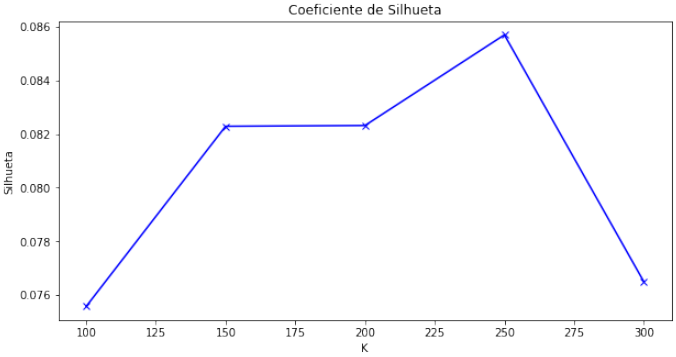


Figure 6. Resultado das medidas de silhueta para 80% da variância



IV. Conclusão e Trabalhos Futuros

Analisando os resultados obtidos sem a redução de dimensionalidade, concluímos que, apesar de alguns valores de k terem apresentado resultados ligeiramente melhores, estes resultados não são muito significativos, pois a média da silhueta de todos permanecem no intervalo $[0, 0.1]$, o que significa que os dados estão muito próximos da fronteira de decisão dos clusters.

A redução de dimensionalidade mostrou um resultado melhor do quê com os dados brutos, apesar de ainda não muito satisfatórios.

Como trabalhos futuros, temos a análise dos dados por algum algorítimo de clusterização que não necessite da escolha prévia do número de classes (como o DBSCAN) ou que seja menos sensível a alta dimensionalidade dos dados. Também, seria interessante procurar outros métodos de extrair informações dos documentos, visando maior representatividade das características as quais desejamos clusterizar estes arquivos.

References

[1] MacQueen, J. B. (1967) Some Methods for classification and Analysis of Multivariate Observations, Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, University of California Press, 1:281-297