Estudos em aprendizado não supervisionado

Matheus Mortatti Diamantino
RA 156740
matheusmortatti@gmail.com

José Renato Vicente
RA 155984
joserenatovi@gmail.com

I. Introdução

Este projeto teve como intuito o estudo prático de aprendizado não supervisionado, abordando diferentes técnicas de clustering, como K-means, DBSCAN e Affinity, e o método de redução de dimensionalidade PCA - Principal Component Analysis. As diferentes abordagens foram testadas e comparadas com relação à qualidade da separação dos dados de acordo com as abordagens quantitativas Silhouette Coefficient e Davies-Bouldin Index. Para isso, foi utilizado uma base de dados de mais de 13000 tweets relacionados a saúde de sites de notícias. Essa base de dados foi separada em 10000 casos de treino e 3000 casos de teste.

II. Atividades

A. Clustering

Clustering é considerado um dos problemas mais importantes de aprendizado não supervisionado. Se trata do processo de encontrar uma estrutura para organizar um conjunto de dados que ainda não foram categorizados. Pode-se dizer de forma informal que busca organizar dados em grupos cujos membros sejam similares de certa forma. Assim, um cluster é um grupo de dados, que podemos chamar de data points, que possuem uma similaridade entre si e que não possuem com os elementos de fora deste cluster.

1) Kmeans: O método Kmeans de clustering pode ser dividido em 4 etapas:

1) Definir K centroides em suas posições iniciais, sendo K o número de grupos em que deseja organizar todos os dados. No algorítimo original, as posições desses centroides são escolhidas aleatoriamente. No Kmeans++, utilizado neste trabalho, a inicialização é um pouco mais trabalhosa:
   a) O primeiro centroide é escolhido aleatoriamente
   b) Para cada data point x que ainda não é centroide, computamos a distância entre x e o centroide mais próximo, chamando essa distancia de D(x)
   c) Escolher um data point como o novo centroide, onde um data point x tem probabilidade de ser escolhido proporcional a D(x)².
   d) Repetir os passos b e c até ter escolhido k centroides
2) Para cada data point, assinalamos a ele o cluster do centroide mais próximo
3) Movemos os centroides para o centro de seus clusters
4) Repetimos os passos 2 e 3 até que os centroides parem de se mover (ou se movam muito pouco) a cada iteração

2) Affinity Propagation: Uma das grandes vantagens do Affinity Propagation é que não é necessário dar como entrada o número de clusters, como no Kmeans, sendo que o próprio algorítimo decide qual é o número ideal.

Seja x_1 a x_n o conjunto de data points e s a função que quantifica o quanto dois data points são similares, de modo que $s(x_i, x_j) > s(x_i, x_k)$ se x_i é mais similar a x_j do que a x_k.

1) Matriz de responsabilidade R possui valores r(i, k) que quantificam o quão qualificado x_k é para servir como exemplar para x_i, relativo a outros data points candidatos a serem exemplares de x_i.
2) Matriz de disponibilidade A possui valores a(i, k) que quantificam o quão apropriado seria para x_i escolher x_k como seu exemplar, relativo ao quão apropriado outros data points seriam para serem escolhidos como exemplares de x_i.

Essas duas matrizes são inicializadas com zeros e são preenchidas ao longo do algorítimo que trabalha nas seguintes etapas:

1) Ajustamos a matriz R de forma que:

$$r(i,k) \leftarrow s(i,k) - max\{a(i,k') + s(i,k')\}, \quad (1)$$

   variando k' entre todos os data points excluindo x_k
2) Ajustamos a matriz A de forma que:

$$a(i,k) \leftarrow min(0, r(k,k) + \sum max(0, r(i',k)), \quad (2)$$

   para $i \neq k$ e $i' \neq \{i, k\}$

Essas etapas são repetidas até que não haja mais modificações nas matrizes ou elas sejam muito pequenas. Os exemplares são extraídos das matrizes como aqueles cujas próprios valores nas duas matrizes somados são positivos, ou seja, $r(i,i) + a(i,i) > 0$. E para um ponto x_i, escolhemos seu exemplar tal que $r(i,k) + a(i,k)$ seja máximo.

B. Principal Component Analysis

Principal Component Analysis é um método muito utilizado para reduzir a dimensionalidade de um grupo de dados em Machine Learning, analizando o custo-benefício entre a dimensionalidade e a qualidade das informações que foram mantidas, que é quantizada pela variância dos dados. Este algoritmo possui cinco etapas distintas.

Primeiro, realizamos um pré processamento dos dados, normalizando e centralizando os dados no espaço onde elas se encontram. Então, computamos a matriz de covariância, para determinar como cada característica do dado se comporta com a variação de uma outra característica. Isso é feito através da fórmula:

$$Covariance = \frac{1}{m}\sum_{i=1}^{n}(x^{(i)})(x^{(i)})^T \tag{3}$$

, onde $x^{(i)}$ é o conjunto de dados para a característica i. Com isso, calculamos os valores dos *eigenvectors e eigenvalues* e então ordenamos os eigenvectors. Deste modo, se queremos k das n características, escolhemos os primeiros k eigenvectors. Por fim, projetamos os dados nos tais k eigenvectors para obtermos o conjunto novo de dados.

C. Silhoutte Coefficient

O Silhouette Coefficient é definido como:

$$s = \frac{b-a}{max(a,b)} \tag{4}$$

Sendo a e b dois Silhouette Scores:

1) a: A distância média entre o data point em análise e todos os outros data points do mesmo cluster
2) b: A distância média entre o data point em análise e todos os outros data points do cluster mais próximo

O Silhouette Coefficient para um grupo de data points, como é o que analisamos neste trabalho, é definido como a média de todos os Silhouette Coefficients de cada data point do grupo.

Esta métrica tem valor entre -1 e 1, sendo um valor mais próximo de -1 para clusterização ruim e mais próximo de 1 para uma boa clusterização.

D. Davies-Bouldin Index

Para calcular o index, primeiro precisamos definir s e d. Para um cluster i dentro de k clusters, tais parâmetros são definidos do seguinte modo:

1) s_i é a distância média entre cada ponto do cluster e seu centroide
2) d_{ij} é a distância entre os centroides dos clusters i e j

Agora podemos calcular R_{ij} como:

$$R_{ij} = \frac{s_i + s_j}{d_{ij}} \tag{5}$$

E, por fim, temos o cáculo da métrica:

$$DB = \frac{1}{N}\sum_{i=1}^{N} D_i \tag{6}$$

$$D_i = maxR_{ij} \tag{7}$$

para $N =$ número de clusters e $i \neq j$. Esta métrica verifica o quão longe cada cluster está dos outros e o quão perto cada ponto de dentro do cluster está de seu centroide. Verificando o valor de R_{ij}, vemos que o valor diminui se a distância entre os clusters i e j é maior, e aumenta caso as distâncias entre os pontos internos de i e j e seus centroides aumenta. Deste modo, quanto menor é o valor desta métrica, melhor a clusterização.

III. Soluções Propostas

Para este trabalho, utilizamos a biblioteca scikit-learn, que conta com a implementação dos algorimos Kmeans++, DBSCAN, Affinity Propagation e PCA, além das métricas de avaliação Silhouette e Davies-Bouldin.

Os textos dos mais de 13000 tweets foram organizados no formato word to vector (cada palavra é transformada em um vetor e os vetores são combinados dentro de um tweet) e normalizados com a função normalize da biblioteca scikit-learn [1].

IV. Experimentos e Discussão

Os experimentos foram realizados em uma máquina que possui um processador Intel Core i7-6700HQ com 4 cores rodando a 2.60GHz e 16GB de RAM, com Ubuntu 16.04.

A. Escolha do Número de Clusters

Primeiramente, gostaríamos de determinar o número de clusters ótimo para o problema em questão. Testamos o algorítimo Kmeans para uma quantidade de clusters entre 10 e 100, de 10 em 10. Em seguida, testamos de 100 a 2000, de 100 em 100. Analisando os resultados das métricas de avaliação e custo, obtivemos os seguintes valores:

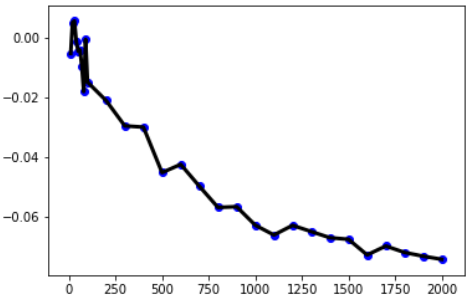

Figure 1: Silhouette Coefficient para clusterização com número de centroides entre 10 e 2000.

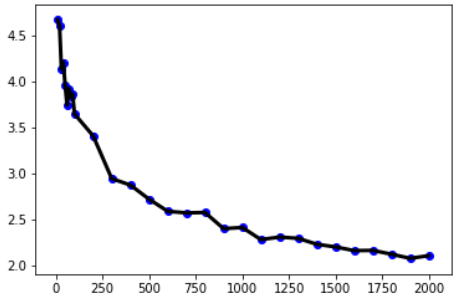

Figure 2: Davies-Bouldin Index para clusterização com número de centroides entre 10 e 2000.

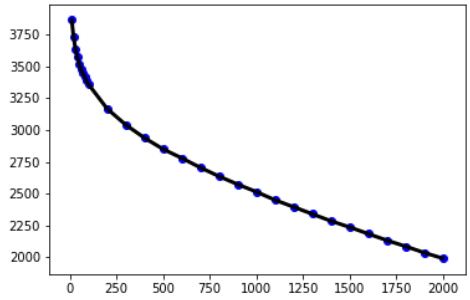

Figure 3: Custo para clusterização com número de centroides entre 10 e 2000.

Em seguida, testamos o algorimo Affinity Propagation, que nos retornou 968 como o número de clusters ideal. Os resultado das métricas Scilhouette Score e Davis-Bouldin Index para Affinity Propagation e K-Means com 100 e 1000 clusters foram:

	K-Means 100 Clusters	K-Means 1000 Clusters	Affinity Propagation (968 clusters achados)
Custo	3358.77	2514.32	-
Silhouette Score	-0.015	-0.063	-0.084
Davies-Bouldin Score	3.64	2.41	2.37

Table I: Resultados Obtidos

Como podemos ver na Figura 1, o Silhouette Score se comportou de modo não usual, variando entre 0.01 e -0.01 nos testes entre 10 a 100 clusters e tendendo a diminuir a partir de então. Isso poderia mostrar-nos que o valor dos clusters ótimo estaria entre 10 e 100, porém a Figura 2 nos mostra que o Davies-Bouldin Score mantém uma tendência a diminuir conforme aumentamos os clusters. Como esta métrica é melhor o quão menor é seu valor, ela nos diz que o número de clusters ótimo está a frente de 100. Somado ao fato de que o algoritmo Affinity Propagation nos retornou m valor de clusters próximo de 1000 e o custo na Figura 3 tende a diminuir linearmente a partir de 250 clusters, decidimos que o valor ótimo de clusters para este problema é 1000. Contudo, comparamos qualitativamente os resultados com 100 e 1000 clusters, na próxima seção.

B. Análise Qualitativa dos Resultados

Analisando os resultados qualitativamente, podemos ver que em nenhum dos métodos obteve-se bons clusters, com uma clara distinção entre eles e tópicos bem definidos. O que pudemos observar é que no pior dos casos, que são grande maioria, os dados presentes em cada cluster possuem assuntos praticamente aleatórios, sem conexão nenhuma entre si. Nos melhores casos, o que acontece é que dentro de um cluster, existem diferentes conjuntos de dados que possuem um assunto em comum, como se fossem mini-clusters dentro do cluster original. Podemos observar isso na Tabela II:

Tweet	Tópico
”Patients ’struggling to book with GP’”,	-
”Overseas nurses ’face shorter tests’”,	nursing
”Ebola: ’heavy toll’ on health staff”,	ebola
’Ebola drug given to US aid workers’,	ebola
’Heart ops backlog pledge by minister’,	heart
’Nursing: Are we facing a trade-off?’,	nursing
’Cancer waiting time targets slip’,	cancer
”Life as a nurse: ’My patients go through hell’”,	nursing
”VIDEO: ’I wish doctors had listened to me’”,	doctors
”’Action’ need over weekend doctors”,	doctors
’Doctors aim to grow ears from fat’,	doctors
’RVH major incident review announced’,	-
’VIDEO: Breast cancer surgery report due’,	cancer
”Cholesterol ’fuels’ breast cancer”,	cancer
’UK doctors to help in Philippines’	doctors

Table II: Exemplo de grupo presente na clusterização com 100 clusters

Os resultados para 1000 clusters com K-Means e para o Affinity Propagation são análogos aos discutidos até aqui e à Tabela II. Verificamos também que, para clusters maiores, existem vários clusters sem nenhum dado.

	Tamanho	Assunto: GP	Assunto: Doenças Mentais
Grupo 653	8	2	1
Grupo Mais Próximo - 41	14	1	2

Table III: Relação de assuntos entre um grupo e seu grupo mais próximo (K-Means 1000 clusters)

	Tamanho	Assunto: Câncer	Assunto: NHS	Assunto: Saúde Física / Comida
Grupo 58	7	1	1	2
Grupo Mais Próximo - 508	14	3	1	2

Table IV: Relação de assuntos entre um grupo e seu grupo mais próximo (K-Means 1000 clusters)

Se observarmos as Tabelas III e IV, que relacionam dois grupos próximos e o número de tweets que falam do mesmo assunto, podemos verificar que existe uma correlação entre alguns tweets. Este resultado era esperado, pois dois clusters muito próximos deveriam compartilhar similaridades, enquanto grupos mais longes deveriam ser o menos similares possível.

C. Aplicação de PCA

Com a escolha de 1000 clusters utilizando K-Means, aplicamos PCA no conjunto de dados e selecionamos 116 características, com uma variância total de 0.954. Obtemos então um Silhouette e Davies-Bouldin Score de -0.0725 e 1.568 respectivamente. Estes resultados são melhores que os verificados para o K-Means sem redução de dimensionalidade, o que nos pareceu estranho. Então, rodamos o algoritmo variando o número de características utilizadas e plotamos os gráficos vistos nas Figuras 4, 5 e 6.

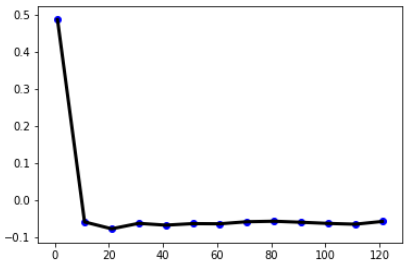

Figure 4: Silhouette Coefficient para clusterização com número de características entre 1 e 120.

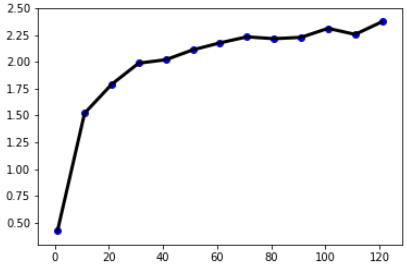

Figure 5: Davies-Bouldin Index para clusterização com número de características entre 1 e 120.

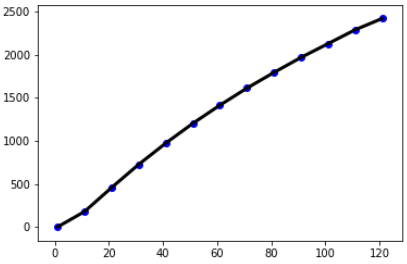

Figure 6: Custo para clusterização com número de características entre 1 e 120.

Verificamos um comportamento fora do normal quando reduzimos o número de características. Tanto o custo quanto a métrica Davies-Bouldin melhoraram significativamente com a redução da dimensionalidade, com ressalvas a Silhouette que melhorou apenas para uma característica e se manteve constante para maiores quantidades de dimensões.

D. DBSCAN - Deteccção de Outliers

Como os resultados obtidos foram abaixo do esperado, um dos possíveis motivos poderia ser a presença de muitos outliers no conjunto de dados. Então, utilizamos o algoritmo DBSCAN para detectá-los. Ao rodar o algorítimo, percebemos dois comportamentos, ao utilizar valores diferentes para a máxima distância entre dois dados para que eles fossem considerados vizinhos:

1) O número de outliers estava na ordem de 12.000 dados (existem aproximadamente 13.000 dados no conjunto).
2) O número de outliers era pequeno, porém existe um grupo que engloba um número de dados da ordem de 12.000. Com isso, concluímos que o algoritmo agrupou os outliers achados em um único grupo.

Em vista disso, podemos concluir que a qualidade dos dados utilizados era bem pequena, salvo aproximadamente 700 exemplos. Se verificarmos a qualidade dos clusters encontrados, podemos verificar um agrupamento de muito mais qualidade dos encontrados nos experimentos feitos acima, com uma tendência de agrupar tweets com hashtags iguais. Alguns exemplos foram grupos com hashtags relacionados a alergias, como #CNNAllergies e #Allergies e relacionados a exercícios, como *#getfit*.

Com o DBSCAN, obtivemos um Silhouette Score de 0.662 e Davies-Bouldin de 0.864, que são resultados muito acima do que obtivemos nos outros experimentos.

V. Conclusões

Após realizarmos todos os testes necessários, podemos dizer que obtivemos resultados abaixo do esperado. Não foram encontrados nenhum agrupamento onde todos os dados dentro de um grupo estão conectados por um assunto ou palavra específica. Por exemplo, dados sobre Ebola aparecem espalhados em diversos grupos diferentes. Isso pode ser consequência do pré processamento dos dados em vetores ter resultado em dados de assuntos diferentes estarem próximos.

Também, os experimentos realizados com redução de dimensionalidade resultaram em comportamentos fora do esperado. Esperava-se que, para uma variância de aproximadamente 0.95, os resultados das métricas fosse levemente piores, mas o algoritmo rodasse mais rapidamente. O segundo fato ocorreu, porém as métricas mostraram-se melhores quanto menos dimensões dos dados eram utilizadas.

Podemos concluir também que utilizar as métricas para escolher o número de clusters não foi um bom método para este caso. Analisando os dados que tinhamos, foi possível identificar que 1000 clusters parecia ser um número muito alto para os assuntos encontrados nos tweets. Contudo, o custo e as métricas apontavam para um número alto de clusters.

Finalmente, Desconfiando da qualidade dos dados, verificamos a existência de muitos outliers, provavelmente devido a um pré processamento de dados que resultou nos dados espalhados no espaço de forma que dados de assuntos semelhantes não se encontravam perto uns dos outros. Também, teorizamos que o pré processamento pode não ter dado o peso necessário para as palavras que conectam dois tweets. Por exemplo, a palavra *Ebola* pode não ter tido um peso grande o suficiente para colocar tweets com ela próximos.

References

[1] “scikit-learn, machine learning in python.” [Online]. Available: http://scikit-learn.org/