Relatórios Técnicos Parcial e Final - Classificação

Lucas Vieira Alves

22 de julho de 2021

Resumo

Neste trabalho foi desenvolvido a base de um método de classificação e redução de características utilizando a teoria da álgebra linear. O *dataset* aplicado a este trabalho foi o *leaf*[1], um *dataset* que consiste de 14 características de forma e textura extraídas de imagens digitais de espécimes de folhas provenientes de um total de 40 espécies de plantas diferentes com 340 amostrar. O trabalho mostra a classificação das espécies de plantas com os dados com a dimensão reduzida através da decomposição em valores singulares. Houve uma redução de 64.28% de características com 99.917% de variabilidade acumulada, resultando em 55% de acurácia utilizando o algoritmo do vizinho mais próximo para classificação das espécies de plantas.

1 Introdução

O aprendizado de máquina é algo bastante comum e difundido nos dias de hoje. São várias aplicações que existem para diversos problemas que surgiram. Os mais visados e solucionados são os problemas de classificação, ou seja, a partir de um conjunto de dados prever uma possível classe indefinida. Os classificadores mais conhecidos são Regressão Logística, KNN (k-nearest neighbors), XGBoost, SVN (Support-vector machine) e etc. Além disso, existem etapas que precede a predição, tal como a pré-processamento dos dados. Nessa etapa são realizados um conjunto de atividades que envolvem preparação, organização, e estruturação dos dados. O método de Análise de Componentes Principais (ACP) é um exemplo bastante utilizado na parte de pré-processamento que permite resumir as principais características dos dados.

[1]https://archive.ics.uci.edu/ml/datasets/leaf

Vários desses algoritmos/métodos estão implementos em bibliotecas (tais como, sklearn e keras) alto níveis permitindo uma utilização abstraindo detalhes, mas prejudicando um real entendimento dos desenvolvedores de como esses algoritmos realmente funcionam.

Portanto, esse trabalho visa implementar conceitos de álgebra linear que fundamentam vários desses algoritmos de classificação e de redução de dimensionalidade permitindo um entendimento mais profundo de como esses estratégias funcionam e são implementadas.

2 Fundamentação Teórica

2.1 Projeções Ortogonais

Quando é necessário aplicar uma função que mapeia o ponto (x, y, z) no espaço tridimensional para o ponto (x, y, 0) bidimensional é um exemplo de uma projeção ortogonal no plano-xy. Assim, uma projeção ortogonal é uma representação num hiperplano de k dimensões de um objeto que tem n dimensões, considerando k $<$ n [1].

2.2 Decomposição espectral

Os pares de autovetor v e autovalor são um conjunto de pares distintos em uma transformação linear A tal que quando a transformação Av é aplicada ao vetor diferente de zero v ele não muda sua direção e pode acabar sendo apenas escalado por um escalar λ [2].

Uma matriz simétrica $A_{n,n}$ pode ser escrita como Q^{-1}, onde Λ é a matriz diagonal dos autovalores de A e Q é uma matriz ortogonal cujas colunas são os autovetores ordenados com base nos autovalores de A.

2.3 Decomposição em valores singulares

O *singular value decomposition* (SVD) decompõe a matriz A_{mxn} em um produto de três matrizes USV^T, onde U_{mxm} (vetores singulares esquerdo) e V_{nxn} (vetores singulares direito) são matrizes ortogonais e S_{mxn} é diagonal (valores singulares). A matriz pode ser tanto matrizes quadradas quanto retangulares tendo elementos reais ou complexos.

Baseada na geometria, o SVD representa uma transformação linear indicando que toda transformação linear é uma composição de três ações fundamentais. Lendo a equação da direita para a esquerda [3]:

1. A matriz V representa uma rotação de vetores no domínio n-dimensional;

2. A matriz S representa uma dilatação ou contração linear ao longo de cada uma das direções da coordenada n. Se $m \neq n$, esta etapa também embute canonicamente (ou projeta) o domínio n-dimensional em (ou sobre) o intervalo m-dimensional;

3. A matriz U representa uma rotação ou reflexão de vetores no intervalo m-dimensional.

3 Metodologia

Para este trabalho foi utilizado o *dataset leaf* composto de 340 registros e com 15 características representando 40 especies de plantas diferentes, como apresentado na Figura 1.

Figura 1: Visão geral das especies de plantas.

Os dados fornecidos compreendem a seguinte forma (atributos 3 a 9) e textura (atributos 10 a 16) *features* (Figura 2).

1. Class (Species)
2. Specimen Number
3. Eccentricity
4. Aspect Ratio
5. Elongation
6. Solidity
7. Stochastic Convexity
8. Isoperimetric Factor
9. Maximal Indentation Depth
10. Lobedness
11. Average Intensity
12. Average Contrast
13. Smoothness
14. Third moment
15. Uniformity
16. Entropy

Figura 2: Atributos do banco de dados.

A variável dependente é apresentado pelo atributo *Class (Species)* e as variáveis independente formado pelos atributos 3-16. Vale ressaltar que o atributo *Specimen Number* foi removido do *dataset*, pois não é um dado relevante para análise.

O banco de dados pode ser baixado em um formato *comma-separated values* (CSV), facilitando a importação para uma estrutura *DataFrame* (semelhante a uma matriz, mas as suas colunas têm nomes e podem conter dados de tipo diferente, bastante utilizado em análise de dados) através da biblioteca Pandas do Python, como apresentado da Figura 3.

	Eccentricity	Aspect Ratio	Elongation	Solidity	Stochastic Convexity	Isoperimetric Factor	Maximal Indentation Depth	Lobedness	Average Intensity	Average Contrast	Smoothness	Third moment	Uniformity	Entropy
0	0.72694	1.4742	0.32396	0.98535	1.00000	0.83592	0.004657	0.003947	0.047790	0.127950	0.016108	0.005232	0.000275	1.17560
1	0.74173	1.5257	0.36116	0.98152	0.99825	0.79867	0.005242	0.005002	0.024160	0.090476	0.008119	0.002708	0.000075	0.69659
2	0.76722	1.5725	0.38998	0.97755	1.00000	0.80812	0.007457	0.010121	0.011897	0.057445	0.003289	0.000921	0.000038	0.44348
3	0.73797	1.4597	0.35376	0.97566	1.00000	0.81697	0.006877	0.008607	0.015950	0.065491	0.004271	0.001154	0.000066	0.58785
4	0.82301	1.7707	0.44462	0.97698	1.00000	0.75493	0.007428	0.010042	0.007938	0.045339	0.002051	0.000560	0.000024	0.34214

Figura 3: Visualização das 5 primeiras linhas do *DataFrame* para o banco de dados *Leaf*.

Depois disso, foi realizado as análise apresentadas nas seções posteriores.

4 Tarefas

4.1 Tarefas Parciais

1. Centralizar a matriz de dados original $\tilde{X}$ obtendo a matriz X.

Solução:

Devemos aplicar a equação na matriz $\tilde{X}$: $X_{ij(cm)} = \tilde{X}_{ij} - \bar{\tilde{X}}_j$

Na Tabela 1, temos a média para cada atributo.

Tabela 1: Média de cada atributo

Atributo	**Média**
Eccentricity	0.719854
Aspect Ratio	2.440210
Elongation	0.513760
Solidity	0.904158
Stochastic Convexity	0.943793
Isoperimetric Factor	0.531234
Maximal Indentation Depth	0.037345
Lobedness	0.523845
Average Intensity	0.051346
Average Contrast	0.124535
Smoothness	0.017670
Third moment	0.005928
Uniformity	0.000387
Entropy	1.162630

Assim geramos um dataset centralizado como apresentado na Figura 4.

	Eccentricity	Aspect Ratio	Elongation	Solidity	Stochastic Convexity	Isoperimetric Factor	Maximal Indentation Depth	Lobedness	Avg Intensity	Avg Contrast	Smoothness	Third moment	Uniformity	Entropy
0	0.007086	-0.96601	-0.18980	0.081192	0.056207	0.304686	-0.032688	-0.519898	-0.003556	0.003415	-0.001562	-0.000695	-0.000112	0.01297
1	0.021876	-0.91451	-0.15260	0.077362	0.054457	0.267436	-0.032102	-0.518843	-0.027186	-0.034059	-0.009550	-0.003220	-0.000312	-0.46604
2	0.047366	-0.86771	-0.12378	0.073392	0.056207	0.276886	-0.029887	-0.513724	-0.039449	-0.067090	-0.014381	-0.005007	-0.000349	-0.71915
3	0.018116	-0.98051	-0.16000	0.071502	0.056207	0.285736	-0.030468	-0.515238	-0.035396	-0.059044	-0.013399	-0.004773	-0.000321	-0.57478
4	0.103156	-0.66951	-0.06914	0.072822	0.056207	0.223696	-0.029917	-0.513803	-0.043408	-0.079196	-0.015619	-0.005368	-0.000364	-0.82049

Figura 4: Amostra de dataset centralizado

2. Dividir o banco de dados em conjunto de treinamento e conjunto de teste (proporções comumente utilizadas são 60%/40% e 70%/30%).

Solução:

Foi usado um método da biblioteca *Sklearn* usando a proporção 70%/30%:

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30)

Sendo X a matriz formada pelas colunas de variáveis independente da *dataset* (atributos) do banco de dados *leaf*. E sendo y a matriz formada pela coluna de variável dependente (*Class*) do banco de dados *leaf*.

Na Figura 5, é apresentado um exemplo de amostra de divisão realizada pela biblioteca para o X que será utilizado como treino.

	Eccentricity	Aspect Ratio	Elongation	Solidity	Stochastic Convexity	Isoperimetric Factor	Maximal Indentation Depth	Lobedness	Avg Intensity	Avg Contrast	Smoothness	Third moment	Uniformity
257	0.132987	-0.241442	0.01050	0.038634	0.055425	0.067544	0.003556	-0.223186	-0.043474	-0.080531	-0.015805	-0.005489	-0.000348
331	-0.240783	-1.198342	0.30477	-0.247316	-0.470895	-0.445574	0.060267	1.183294	0.047044	0.085240	0.024665	0.010851	0.000204
101	-0.210883	-1.083342	-0.20581	0.038304	0.036125	0.134744	0.013175	-0.067056	0.044678	0.104960	0.032569	0.019880	-0.000108
167	-0.175093	-1.258142	0.04985	-0.162516	-0.169135	-0.267146	0.044053	0.661694	0.039748	0.037270	0.007960	0.001321	0.000430
201	-0.032773	-0.846242	-0.17217	0.066604	0.037885	0.231744	-0.023826	-0.483226	-0.043974	-0.081248	-0.015869	-0.005507	-0.000340

Figura 5: Amostra X treino

3. Calcular a matriz de covariância dos dados treinamento de duas maneiras:

   - np.cov($\tilde{X}$)
   - $\frac{1}{m-1}X^T X$

Comparar os resultados. Atenção: verique se o parâmetro rowvar na função np.cov() deve ser *True* ou *False*.

Solução:

Utilizando a primeira forma de calcular a matriz de covariância foi necessário passar *True* para parâmetro *rowvar* para indicar que as colunas representam os atributos. Resultado apresentado na Figura 6.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0.040203	0.263788	0.024762	0.005836	0.006107	-0.005065	-0.001448	-0.029920	-0.002127	-0.002495	-0.000680	-2.044286e-04	-2.797594e-05	-0.036250
1	0.263788	5.993216	0.320300	-0.003711	0.019182	-0.248191	0.009047	0.254652	-0.026556	-0.039186	-0.009362	-3.165204e-03	-2.637395e-04	-0.475588
2	0.024762	0.320300	0.035419	-0.006567	-0.006001	-0.029297	0.002508	0.059403	-0.001842	-0.002460	-0.000649	-2.217594e-04	-2.412109e-05	-0.028986
3	0.005836	-0.003711	-0.006567	0.009404	0.008204	0.014564	-0.002905	-0.068639	0.000325	0.000546	0.000136	4.707357e-05	4.901473e-06	0.002323
4	0.006107	0.019182	-0.006001	0.008204	0.009724	0.012217	-0.002562	-0.062682	0.000398	0.000659	0.000160	5.876434e-05	5.177930e-06	0.004483
5	-0.005065	-0.248191	-0.029297	0.014564	0.012217	0.040610	-0.004925	-0.105410	0.000657	0.001098	0.000285	1.138083e-04	1.080978e-05	0.006597
6	-0.001448	0.009047	0.002508	-0.002905	-0.002562	-0.004925	0.001149	0.027281	-0.000081	-0.000144	-0.000021	-2.938734e-06	-1.582618e-06	-0.001464
7	-0.029920	0.254652	0.059403	-0.068639	-0.062682	-0.105410	0.027281	0.729251	-0.004993	-0.008213	-0.001790	-5.828927e-04	-5.689115e-05	-0.080601
8	-0.002127	-0.026556	-0.001842	0.000325	0.000398	0.000657	-0.000081	-0.004993	0.001393	0.001879	0.000504	1.651187e-04	1.361872e-05	0.021210
9	-0.002495	-0.039186	-0.002460	0.000546	0.000659	0.001098	-0.000144	-0.008213	0.001879	0.002795	0.000733	2.636034e-04	1.556438e-05	0.027507
10	-0.000680	-0.009362	-0.000649	0.000136	0.000160	0.000285	-0.000021	-0.001790	0.000504	0.000733	0.000201	7.342243e-05	4.058618e-06	0.007005
11	-0.000204	-0.003165	-0.000222	0.000047	0.000059	0.000114	-0.000003	-0.000583	0.000165	0.000264	0.000073	2.988359e-05	9.643493e-07	0.002084
12	-0.000028	-0.000264	-0.000024	0.000005	0.000005	0.000011	-0.000002	-0.000057	0.000014	0.000016	0.000004	9.643493e-07	2.058393e-07	0.000220
13	-0.036250	-0.475588	-0.028986	0.002323	0.004483	0.006597	-0.001464	-0.080601	0.021210	0.027507	0.007005	2.084186e-03	2.198046e-04	0.370316

Figura 6: Matriz de covariância usando a primeira forma

Para a outra forma de calcular a matriz de covariância foi aplicado apenas um produto matricial (np.dot) entre X transporto (X.T) e X e dividindo pela quantidade de colunas subtraindo 1. Resultado apresentado na Figura 7.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0.040203	0.263788	0.024762	0.005836	0.006107	-0.005065	-0.001448	-0.029920	-0.002127	-0.002495	-0.000680	-2.044286e-04	-2.797594e-05	-0.036250
1	0.263788	5.993216	0.320300	-0.003711	0.019182	-0.248191	0.009047	0.254652	-0.026556	-0.039186	-0.009362	-3.165204e-03	-2.637395e-04	-0.475588
2	0.024762	0.320300	0.035419	-0.006567	-0.006001	-0.029297	0.002508	0.059403	-0.001842	-0.002460	-0.000649	-2.217594e-04	-2.412109e-05	-0.028986
3	0.005836	-0.003711	-0.006567	0.009404	0.008204	0.014564	-0.002905	-0.068639	0.000325	0.000546	0.000136	4.707357e-05	4.901473e-06	0.002323
4	0.006107	0.019182	-0.006001	0.008204	0.009724	0.012217	-0.002562	-0.062682	0.000398	0.000659	0.000160	5.876434e-05	5.177930e-06	0.004483
5	-0.005065	-0.248191	-0.029297	0.014564	0.012217	0.040610	-0.004925	-0.105410	0.000657	0.001098	0.000285	1.138083e-04	1.080978e-05	0.006597
6	-0.001448	0.009047	0.002508	-0.002905	-0.002562	-0.004925	0.001149	0.027281	-0.000081	-0.000144	-0.000021	-2.938734e-06	-1.582618e-06	-0.001464
7	-0.029920	0.254652	0.059403	-0.068639	-0.062682	-0.105410	0.027281	0.729251	-0.004993	-0.008213	-0.001790	-5.828927e-04	-5.689115e-05	-0.080601
8	-0.002127	-0.026556	-0.001842	0.000325	0.000398	0.000657	-0.000081	-0.004993	0.001393	0.001879	0.000504	1.651187e-04	1.361872e-05	0.021210
9	-0.002495	-0.039186	-0.002460	0.000546	0.000659	0.001098	-0.000144	-0.008213	0.001879	0.002795	0.000733	2.636034e-04	1.556438e-05	0.027507
10	-0.000680	-0.009362	-0.000649	0.000136	0.000160	0.000285	-0.000021	-0.001790	0.000504	0.000733	0.000201	7.342243e-05	4.058618e-06	0.007005
11	-0.000204	-0.003165	-0.000222	0.000047	0.000059	0.000114	-0.000003	-0.000583	0.000165	0.000264	0.000073	2.988359e-05	9.643493e-07	0.002084
12	-0.000028	-0.000264	-0.000024	0.000005	0.000005	0.000011	-0.000002	-0.000057	0.000014	0.000016	0.000004	9.643493e-07	2.058393e-07	0.000220
13	-0.036250	-0.475588	-0.028986	0.002323	0.004483	0.006597	-0.001464	-0.080601	0.021210	0.027507	0.007005	2.084186e-03	2.198046e-04	0.370316

Figura 7: Matriz de covariância usando a segunda forma

Para verificar a tendência de igualdade entre as duas formas de calcular matriz de covariância, foi utilizada estratégia do máximo da diferença da matriz de covariância calculada com a primeira forma e matriz de covariância calculada com a segunda forma.
Assim, obtivemos um error de $8.881784197001252 * 10^{-15}$, ou seja, ambas forma de calcular a matriz de covariância geram uma matriz muito próxima de equivalentes, podendo assumir que são equivalentes.

4. Calcular a decomposição espectral da matriz de covariância dos dados de treinamento, isto é, $cov(X) = Q \wedge Q^T$, usando a função do NumPy:

np.linalg.eigh(cov(X)).

Solução:

Foi utilizado o método *eigh* do pacote *linalg* da biblioteca Numpy para calcular a decomposição espectral. É retornado um array de autovalores (w) e uma matriz de autovetores (Q), ambos ordenados

por posição, ou seja, o autovalor da posição 1 do array é referente ao autovetor da posição 1 da coluna da matriz.

Como é necessário ordenar em ordem decrescente, e como a método já retorna os autovalores e autovetores ordenados em ordem ascendente, basta inverte os elementos do array e da matriz.

Figura 8 e 9 é apresentada os autovalores e autovetores ordenados em ordem decrescente para o X (X de treino) do *dataset leaf* respectivamente.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	6.086171	0.755413	0.328461	0.036575	0.021228	0.002622	0.001514	0.000937	0.000682	0.000061	0.000041	0.000004	9.062929e-08	1.036916e-08

Figura 8: Auto valores da matriz X (w)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0.043795	-0.056757	0.064449	-0.709128	-0.532891	0.121320	0.412674	-0.089382	0.103590	0.005444	-0.009876	-0.002355	-0.000166	0.000007
1	0.991891	-0.065418	-0.076333	0.074075	-0.011820	-0.014180	0.014070	-0.002849	-0.003176	-0.000815	0.001173	-0.000293	0.000021	-0.000006
2	0.053782	0.060919	-0.012897	-0.605804	0.165655	-0.335476	-0.618328	0.166523	-0.273894	-0.033351	0.007235	0.002061	-0.000537	-0.000098
3	-0.001313	-0.093790	0.024036	0.047270	-0.277087	0.021390	-0.518242	-0.096532	0.794162	-0.018006	0.046809	0.007279	0.000026	0.000305
4	0.002461	-0.088088	0.009507	0.066135	-0.283725	0.782061	-0.366289	0.199495	-0.348415	0.007349	-0.015230	0.002293	0.000499	0.000078
5	-0.041976	-0.127105	0.086987	0.338512	-0.694878	-0.503434	-0.093967	0.085403	-0.320369	-0.066112	0.021129	-0.006855	-0.000004	-0.000352
6	0.001763	0.036305	-0.009242	-0.008050	0.036727	0.060731	0.026212	-0.087445	-0.015738	-0.942208	0.301850	-0.077625	0.003673	-0.001446
7	0.049604	0.968784	-0.114930	0.054380	-0.198726	0.028697	-0.031626	0.004413	0.025491	0.028425	-0.005810	0.002263	-0.000069	0.000056
8	-0.004700	-0.007892	-0.056153	0.006808	-0.015680	0.006628	-0.084945	-0.361205	-0.057378	-0.222007	-0.850809	-0.229217	-0.147284	-0.089740
9	-0.006884	-0.012159	-0.071073	-0.004602	-0.020734	0.037902	-0.153602	-0.828330	-0.224125	0.217382	0.384191	-0.195016	-0.002007	-0.004111
10	-0.001651	-0.002762	-0.018367	0.001784	-0.007544	0.007970	-0.043754	-0.246716	-0.059335	-0.089635	-0.177800	0.698425	0.588416	0.244022
11	-0.000554	-0.000863	-0.005363	0.000500	-0.002974	0.006051	-0.017053	-0.124026	-0.036167	-0.030743	0.039437	0.640279	-0.730761	-0.190983
12	-0.000047	-0.000090	-0.000581	0.000326	-0.000173	-0.000201	-0.000786	0.000547	0.001162	-0.004661	-0.024779	-0.073587	-0.313116	0.946522
13	-0.083881	-0.126484	-0.979670	-0.018985	-0.082040	-0.030335	0.029968	0.085708	0.016341	-0.001020	0.023901	0.010631	0.001715	0.001337

Figura 9: Auto vetores da matriz X (Q)

4.2 Tarefas Finais

1. Dividir o banco de dados em conjunto de treinamento e conjunto de teste (proporções comumente utilizadas são 60%/40% e 70%/30%.)

Solução:

Esta tarefa é resolvida da mesma forma que a segunda tarefa das Tarefas Parciais.

2. Calcular, para o respectivo banco de dados, a SVD da matriz de dados de treinamento centralizados X. Gerar um gráfico de número de valores singulares versus variabilidade acumulada.

Solução:

Inicialmente devemos aplicar a equação na matrizes $\tilde{X}$ de treino e de teste: $X_{ij(cm)} = \tilde{X}_{ij} - \tilde{\bar{X}}_j$.
Assim geramos a matriz de treino centralizada como apresentado na Figura 10.

	Eccentricity	Aspect Ratio	Elongation	Solidity	Stochastic Convexity	Isoperimetric Factor	Maximal Indentation Depth	Lobedness	Average Intensity	Average Contrast	Smoothness	Third moment	Uniformity	Entropy
172	0.164358	-0.068117	0.074985	0.039808	0.018537	0.000868	-0.010762	-0.363701	-0.000130	0.017307	0.002120	0.001489	-0.000086	-0.100997
38	-0.348322	-1.281817	-0.339495	0.044428	0.044857	0.239568	-0.022543	-0.439395	-0.020618	-0.038812	-0.009524	-0.003656	-0.000174	-0.062697
242	0.130808	-0.256817	0.028285	0.059178	0.044857	0.101688	-0.013977	-0.389368	0.035279	0.070237	0.018274	0.008025	0.000030	0.365303
64	0.123648	-0.611817	0.046175	-0.042172	0.015037	-0.027812	-0.004749	-0.305591	-0.018144	-0.024157	-0.007054	-0.002029	-0.000211	-0.099197
181	0.225308	1.405783	0.232845	-0.093962	-0.007773	-0.225252	0.110296	3.340629	-0.023312	-0.015543	-0.005411	-0.000557	-0.000287	-0.573547

Figura 10: Amostra da matriz de treino centralizado

Depois disso foi aplicado:

U, S, Vt =np.linalg.svd(x_de_treino_centralizado,
full_matrices=False)

para realizar a decomposição SVD reduzida de matriz x centralizada de treino (238*14), no qual o U gerado é uma matriz com colunas ortogonais (238*14) representando os vetores singulares à esquerda, S é uma matriz diagonal (14*14) representado valores singulares e Vt é a matriz ortogonal transposta dos vetores singulares à direita (14*14).

Para calcular a variabilidade acumulada é utilizada a seguinte fórmula:

$$E = \frac{\lambda_1+\lambda_2+...+\lambda_r}{\lambda_1+\lambda_2+...+\lambda_n}$$

no qual os λ‘s são os valores singulares da decomposição SVD (S), r é a dimensão do hiperplano em que os dados serão projetados e n é a quantidade de atributos da matriz de treino (hiperplano de dimensão máxima).
O código abaixo é a fórmula acima feita em python:

$$(np.sum(S[0:r] * *2)/np.sum(S * *2)) * 100$$

Foi criado um laço que testou todas as dimensionalidades possíveis para projeção a matriz de treino x, 1 dimensão até 14 dimensões, i.e, de 1 *feature* até 14 *feature*. Esse procedimento é apresentado no gráfico na Figura 11.

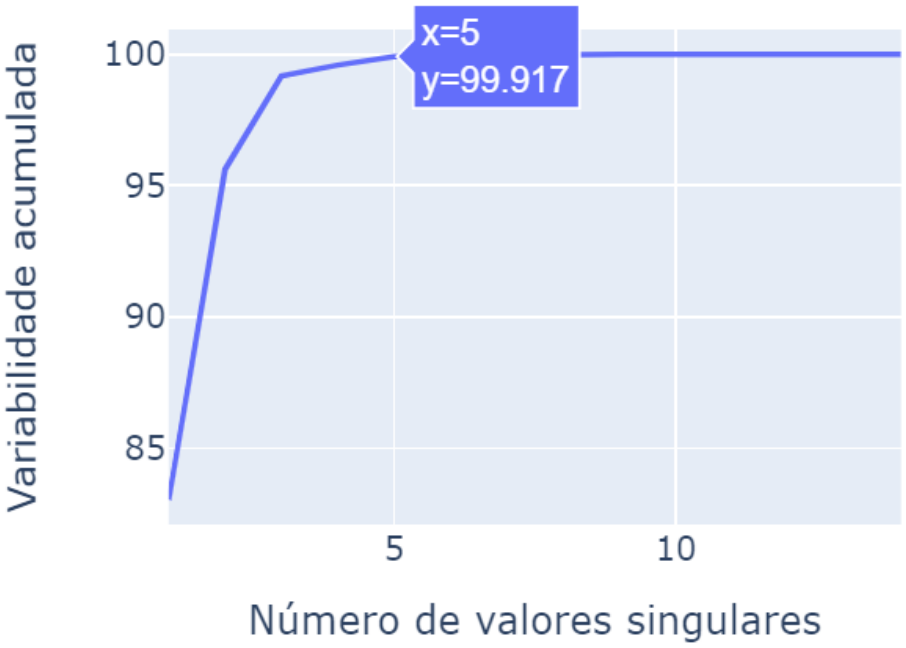

Figura 11: Variabilidade acumulada vs Número de valores singulares

Na Tabela 2 é apresentado a variabilidade acumulada para cada número de valores singulares.

Tabela 2: Número de valores singulares vs Variabilidade acumulada

Número de valores singulares	**Variabilidade acumulada**
1	83.03459183915494
2	95.61839514185483
3	99.17827716597364
4	99.5881777526391
5	99.91703200987445
6	99.95709318955667
7	99.9773525572496
8	99.98915806672541
9	99.99879145774419
10	99.99952732412878
11	99.99995158240056
12	99.99999893608009
13	99.99999988762221
14	100.0

3. Selecionar valores apropriados de variabilidade acumulada para reduzir a dimensionalidade do problema de classificação, resolvendo-o para o respectivo banco de dados.
Solução:

Foi utilizando $r = 5$ com aproximadamente 99.917% de variabilidade acumulada, mas poderia ser usada $r = 2$ pois a variabilidade acumulada já estava acima de 90% (valor geralmente adotado), comom pode ser verificado na Tabela 2. Dessa forma a redução de dimensionalidade permitiu uma redução de 64.28% *features*.
Para geração da matriz de treino projetado no hiperplano de 5 dimensões ($r = 5$), foi utilizado:

$$U[:, 0 : r] * S[0 : r]$$

E para geração da matriz de teste projetado no hiperplano de 5 dimensões ($r = 5$) foi utilizado:

$$np.dot(x\_de\_teste\_centralizado, Vt[0:r,:].T)$$

Para realizar a classificação propriamente dita, foi utilizado o algoritmo *k-Nearest Neighbors* (KNN) do *sklearn*. Para esse algoritmo, é necessário ajustar o hiperparâmetro K que indica a quantidade de vizinhos. Afim de encontrar um bom hiperparâmetro K, foi realizado um teste variando K de 1 até 50 usando a matriz dados reduzida (5 dimensões), como apresentado na Figura 12. Pode ser notado na figura que com K =4 gera a maior acurácia (aprox. 53%).

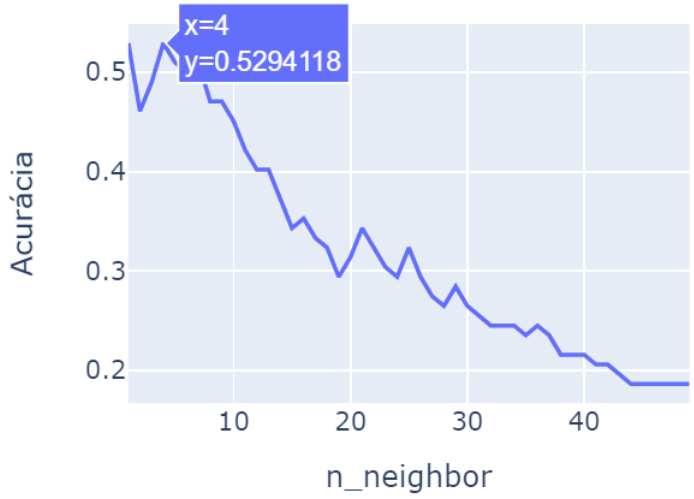

Figura 12: Acurácia vs hiperparâmetro K *(n_neighbor)*

Portando dessa forma, conseguimos uma predição da classificação das folhas com a acurácia aproximadamente de 53% quando os dados de treino reduzidos para 5 dimensões e utilizando hiperparâmetro K igual a 4.

4. Gerar um gráfico de número de valores singulares versus acurácia. Isso deve ser feito de maneira apropriada, não devendo o gráfico ser gerado em tempo superior a 24 horas.

Solução:

Usando o algoritmo KNN usando o hiperparâmetro $K = 4$ para predizer a classificação das folhas (conforme a conclusão da tarefa anterior),

foi iterado incrementado a dimensão do hiperplano de projeção sobre a matriz de treino e de teste (r) indo de 1 até 14 (número máximo de dimensão), como apresentado na Figura 13.

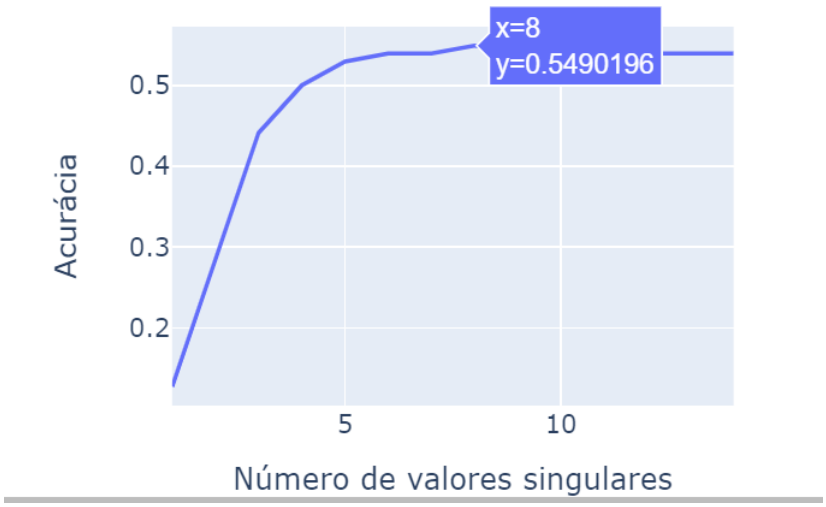

Figura 13: Acurácia vs Número de valores singulares

Podemos observar que mesmo com a redução da dimensão dos dados, é possível obter resultados melhores, como por exemplo, o resultado com 8 dimensões gerou um resultado melhor do que os hiperplanos de maiores dimensões.

5 Questões

5.1 Questões Parciais

1. Demonstrar que uma matriz Q é ortogonal (isto é, quadrada com colunas ortonormais) se, e somente se, Q^TQ = I.

Solução:

Precisamos provar que:

$$(Q^T Q)_{ij} = \begin{cases} 0 & se \quad i \neq j \\ 1 & se \quad i = j \end{cases}$$

Denotemos por q_i a i-ésima coluna de Q (e então a i-ésima linha de Q^T). Como o elemento (i,j) de $Q^T Q$ é o produto escalar da i-ésima linha de Q^T pela j-ésima coluna de Q, segue que

$$q_i q_j = \begin{cases} 0 & se \quad i \neq j \\ 1 & se \quad i = j \end{cases} \tag{1}$$

O que, pela equação (1), vale se e somente se

$$(Q^T Q)_{ij} = \begin{cases} 0 & se \quad i \neq j \\ 1 & se \quad i = j \end{cases}$$

Isso completa a demostração $\square$.

2. Verificar, com o respectivo banco de dados, que $Q^T Q \approx I$.

Resposta:

Aplicando os autovetores ordenados de forma decrescente da matriz x de treino do banco de dados *leaf*, foi possível verificar a aproximação utilizando o erro ($erro = max(Q^T Q - identidade)$, e matriz identidade com a mesma quantidade de linhas e colunas $Q^T Q$).
Foi obtido um erro de $1.5543122344752192 * 10^{-15}$, ou seja, aproximadamente zero. Assim, sendo verificada a aproximação solicitada.

3. Consideremos Z a matriz m $\times$ n em cujas linhas estão as coordenadas das amostras em relação à base de autovetores, isto é:

$$X^T = QZ^T;$$

e $\hat{Q}$ a matriz n $\times$ r, cujas colunas são a r primeiras colunas de Q. Mostrar que a matriz cujas linhas são as coordenadas das amostras em relação à base de autovetores do subespaço gerado pelos r primeiros autovetores (associados aos r maiores autovalores) é dada por:

$$\hat{Z} = \mathrm{X}\hat{Q}.$$

Solução:

Aplicando Q^T em ambos lado da primeira igualdade:

$$Q^T X^T = Q^T Q Z^T;$$

Considerando que Q^TQ = I, temos:

$$Q^T X^T = I Z^T;$$

Aplicando o conceito: matriz identidade é neutra, ou seja, qualquer matriz multiplicada pela matriz identidade terá como resultado a própria matriz, temos:

$$Q^T X^T = Z^T;$$

Aplicando a propriedade: a transposta da multiplicação de duas matrizes é igual ao produto das transpostas de cada uma delas, em ordem inversa, temos:

$$(XQ)^T = (Z)^T;$$

Removendo a transporta da igualdade temos:

$$Z = XQ;$$

Sendo demostrado isso, essa igualdade é valida para qualquer $1 \leq r \leq$ #(colunas de Q), portanto sendo válida a igualdade:

$$\hat{Z} = \mathrm{X}\hat{Q}. \ \square$$

4. Seguindo a Equação (9), a matriz dos dados projetados é definida por:

$$\hat{X}^T = \hat{Q}\hat{Z}^T.$$

Mostrar que os dados projetados são calculados pela equação

$$\hat{X} = X\hat{Q}\hat{Q}^T$$

Solução:

a Transporta em ambos o lado da igualdade para calcular a matriz dos dados projetados:

$$(\hat{X}^T)^T = (\hat{Q}\hat{Z}^T)^T.$$

Aplicando distribuindo a transporta temos:

$$\hat{X}^{TT} = \hat{Z}^{TT}\hat{Q}^T.$$

Aplicando a propriedade: a transposta de uma matriz transposta é a matriz original, temos:

$$\hat{X} = \hat{Z}\hat{Q}^T.$$

Aplicando $\hat{Z} = \mathrm{X}\hat{Q}$:

$$\hat{X} = X\hat{Q}\hat{Q}^T. \ \square$$

5. Calcular, com o respectivo banco de dados, as matrizes $\hat{Z}$ e $\hat{X}$; verificar que a matriz de projeção

$$\hat{Q}\hat{Q}^T$$

não é a matriz identidade.

Solução:

Foi calculado o $\hat{Z}$ e $\hat{X}$ referente ao banco de dados *leaf* para o X de treino com $1 \leq r \leq 14$. Sendo 14 a quantidade de colunas x de treino.
A Figura 10 apresenta $\hat{Z}$ com r = 10 ou seja, $\hat{Q}$ são as 10 primeiras colunas de Q.

	0	1	2	3	4	5	6	7	8	9
0	-1.264130	-0.268892	-0.389557	0.200387	0.121390	0.046713	-0.019986	-0.011343	-0.019699	0.004932
1	0.117616	-0.369776	0.591253	-0.144569	-0.056957	-0.028402	-0.036964	0.027690	-0.006299	0.002342
2	-0.424367	-0.018530	1.008298	-0.048111	0.019799	0.056556	0.007153	0.013921	0.024794	-0.015998
3	-0.393218	-0.345142	0.526075	-0.097283	-0.061315	-0.020306	-0.008177	0.029169	-0.001630	0.000593
4	-0.886490	-0.114538	-0.520175	0.013049	-0.144057	0.003992	0.026305	-0.092031	-0.049778	-0.008749
...	...	...	...	...	...	...	...	...	...	...
233	0.335123	-0.474545	-0.565711	-0.236346	-0.046154	0.008200	-0.030083	-0.012467	0.016098	0.006383
234	-0.269190	-0.035028	-0.504790	-0.194566	-0.063367	0.068207	0.012510	-0.000382	0.025091	-0.005342
235	-0.704327	-0.119118	0.052653	-0.219738	0.132562	0.103944	0.093140	0.007753	-0.024441	0.008977
236	-1.288365	-0.303514	-0.317027	0.222456	0.191566	0.068412	-0.016778	0.038349	0.041705	0.009889
237	0.126905	-0.319065	-0.450978	-0.189927	-0.103918	-0.012444	-0.027805	0.008294	0.001641	-0.008153

238 rows × 10 columns

Figura 14: $\hat{Z}$ com r = 10

A Figura 11 apresenta $\hat{X}$ com r = 10 ou seja, $\hat{Q}$ são as 10 primeiras colunas de Q.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	-0.055362	-1.253879	-0.067987	0.001660	-0.003110	0.053063	-0.002228	-0.062706	0.005942	0.008703	0.002087	0.000700	0.000059	0.106036
1	0.005151	0.116662	0.006326	-0.000154	0.000289	-0.004937	0.000207	0.005834	-0.000553	-0.000810	-0.000194	-0.000065	-0.000006	-0.009866
2	-0.018585	-0.420926	-0.022823	0.000557	-0.001044	0.017813	-0.000748	-0.021050	0.001995	0.002921	0.000701	0.000235	0.000020	0.035596
3	-0.017221	-0.390030	-0.021148	0.000516	-0.000968	0.016506	-0.000693	-0.019505	0.001848	0.002707	0.000649	0.000218	0.000018	0.032983
4	-0.038824	-0.879301	-0.047677	0.001164	-0.002181	0.037211	-0.001563	-0.043973	0.004167	0.006103	0.001463	0.000491	0.000042	0.074359
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
233	0.014677	0.332406	0.018024	-0.000440	0.000825	-0.014067	0.000591	0.016623	-0.001575	-0.002307	-0.000553	-0.000186	-0.000016	-0.028110
234	-0.011789	-0.267007	-0.014478	0.000353	-0.000662	0.011299	-0.000475	-0.013353	0.001265	0.001853	0.000444	0.000149	0.000013	0.022580
235	-0.030846	-0.698615	-0.037880	0.000925	-0.001733	0.029565	-0.001242	-0.034937	0.003311	0.004849	0.001163	0.000390	0.000033	0.059079
236	-0.056424	-1.277917	-0.069291	0.001692	-0.003170	0.054080	-0.002271	-0.063908	0.006056	0.008869	0.002127	0.000714	0.000061	0.108069
237	0.005558	0.125876	0.006825	-0.000167	0.000312	-0.005327	0.000224	0.006295	-0.000597	-0.000874	-0.000209	-0.000070	-0.000006	-0.010645

238 rows × 14 columns

Figura 15: $\hat{X}$ com r = 10

Na Tabela 2 temos o erro sendo sendo o máximo da subtração da matriz e sua identidade. Pode ser notado que para $1 \leq r \leq 13$, gera um valor error "alto", indicando que $\hat{Q}\hat{Q}^T$ não é válida, exceto quando r = 14.

Tabela 3: Tabela de erro para cada r

r	**Erro**
1	0.053345671687517755
2	0.0616854817554137
3	0.071743063516429
4	0.4276588744474242
5	0.33938253443276306
6	0.2986824084755838
7	0.29678430973455944
8	0.31690841408818327
9	0.3297682605795461
10	0.2815078500155153
11	0.268653757397879
12	0.4765952526029189
13	0.18076936152177359
14	$1.1102230246251565 * 10^{-15}$

5.2 Questões Finais

1. Considerando uma matriz de dados centralizados X m×n, mostrar que as matrizes X^TX e XX^T possuem os mesmos autovalores não nulos.

Solução:

Assumindo que λ é um autovalor de X^TX com o autovetor a. Dessa forma, temos (X^TX - λI)a = 0. Agora multiplicamos ambos os lados por X, e conseguimos X(X^TX - λI)a = X * 0 = 0 $\leftrightarrow$ (XX^TX - XλI)a = 0. Agora X comuta com λI e abitemos (XX^TX - λIX)a = 0 $\leftrightarrow$ (XX^T - λI)Xa = 0. Dessa forma podemos interpretar que autovetor de XX^T é X vezes o autovetor de X^TX mantendo o mesmo autovalores.

2. Verificar numericamente a proposição anterior, calculando, para o respectivo banco de dados, os autovalores de X^TX e XX^T.

Solução:

Foi calculada a matriz de covariância para as matrizes X^TX e XX^T. Usando o algoritmo em python abaixo:

$$np.dot(aT, a)/(m-1)$$

no qual m é a quantidade de linhas da matriz X. A matriz de covariância X^TX resultou um tamanho de (14*14) e a matriz de covariância XX^T (238*238).
Depois foi realizado a decomposição espectral para cada matriz de covariância usando $np.linalg.eigh(matriz\_de\_covariância)$, ordenando os autovalores em ordem decrescente e ordenando os autovetores com base nos autovalores ordenados.
Os autovalores para matriz de covariância X^TX resultou em uma matriz diagonal (14*14) e autovalores para matriz de covariância XX^T resultou em uma matriz diagonal (238*238). Para saber se os autovalores de ambas matrizes de covariância são equivalentes foi aplicado uma função para verificar a diferença entre as duas, como apresentado abaixo em python:

mx = np.max(A-B)
mn = np.min(A-B)
diferença = max(abs(mn), abs(mx)

No qual A e B são matrizes de mesmo tamanho.
Antes de aplicar esse método de diferença, foi necessário ajustar o tamanho da matrizes para que ficassem do mesmo tamanho. Portando a matriz de autovalores de X^TX foi completada com zeros para passar a ter a mesmo tamanho da matriz de autovalores da matriz de covariância XX^T. O resultado da diferença foi $1.176836406102666e^{-14}$, podendo assumir que apresentam o mesmo autovalores para a matriz de covariância X^TX e XX^T.

3. Considerando a SVD X = USV^T, mostrar que as colunas de V são os autovetores de X^TX e que as n colunas de U são os autovetores de XX^T associados aos n maiores autovalores.

Solução:

Dada uma matriz X_{mxn}, aplicando SVD na matriz de covariância X^TX e XX^T temos:

$$(1)\ X^TX = V\Lambda^TU^T = V(\Lambda^T\Lambda)V^T$$
$$(2)\ XX^T = U\Lambda V^TV\Lambda^TU^T = U(\Lambda\Lambda^T)U^T$$

Baseado na decomposição espectral $Q\lambda Q^T$, podemos afirmar: (i) que as colunas de V (vetores no singular à direita) são autovetores de X^TX; (ii) que as colunas de U (vetores no singular à esquerda) são autovetores de XX^T.
A segunda afirmação só é valida para n autovetores associados a n maiores autovalores não nulos e para n colunas de U, pois não se tem o "controle"dos autovetores associado a autovalores nulos comumente gerando vetores ortogonais das colunas de Q diferente os vetores ortogonais das colunas de U quando índice maior que n.

4. Verificar a proposição anterior, comparando, para o respectivo banco de dados, a matriz de autovetores de X^TX e a matriz de vetores singulares direitos, isto é, V em X = USV^T. Atenção para com os sentidos dos vetores.

Solução:

Inicialmente foi gerada a matriz de covariância $X^T X$ e aplicada a fatoração espectral para obter os autovetores (Q) conforme explicado no questão 2 da seção de Questões finais.
Também foi aplicada a decomposição completa SVD conforme o código em python abaixo:

$$U, S, Vt = np.linalg.svd(x\_de\_teste\_centralizado, full\_matrices = True)$$

na matriz de X de treino resultando em uma matriz ortogonal de vetores singulares esquerdo U (238*238), uma matriz diagonal S (14*14) e uma matriz ortogonal de vetores singulares direito transposta (14*14).
Sendo aplicada a transposta em Vt, é possível obter de fato a a matriz de vetores singulares (V). Dessa forma foi aplicada a diferença entre os valores absoluto das matrizes Q e V utilizando a mesmo método da questão 3 da seção de Questões finais, resultando em uma diferença de $1.266209359584991e^{-13}$, podendo afirma que Q da decomposição espectral e V da decomposição SVD são equivalentes.

5. Comparar, para o respectivo banco de dados, a matriz Q de autovetores de XX^T e a matriz de vetores singulares esquerdos, isto é, U em X = USV^T. Justifique por que as duas matrizes são diferentes, mas as submatrizes U[:, 0 : t] e Q[:, 0 : t] são iguais (t é o índice do primeiro autovalor nulo).

Solução:

Foi gerada a matriz de covariância XX^T e aplicada a fatoração espectral para obter os autovetores (Q) ordenados e aplicada a decomposição SVD completa para obtenção de U (vetores singulares esquerdos). Comparando os valores absolutos das matrizes Q e U através da diferença foi obtido 0.9839949753541171, indicando que elas não são equivalentes.

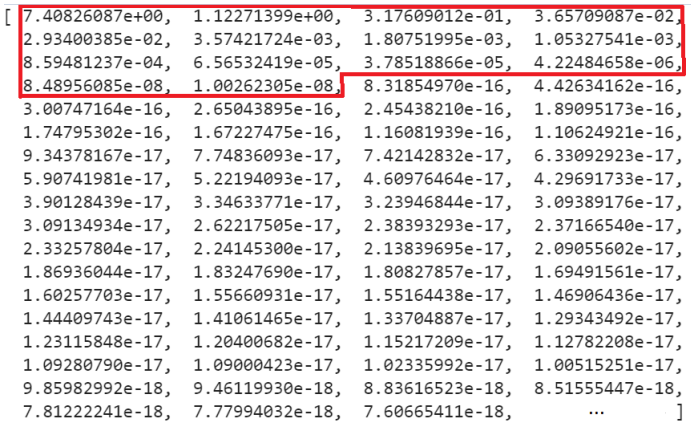

Figura 16: Autovalores decrescente de XX^T

Na Figura 16, é apresentado todos os autovalores ordenados em ordem descente matrizes de covariância de XX^T e os autovalores marcados, apresenta os autovalores não nulos, ou seja, são os 14 primeiros autovalores, pois os outros tem uma ordem de grandeza muito baixa, sendo considerados desprezíveis. Dessa forma, quando utilizado a submatrizes Q e U de dimensões 14 e realizando a diferença, é obtido $1.5328752332732787e^{-07}$, indicando que essas submatrizes são equivalentes.
Nessa situação o SVD completo, complementa as colunas da matriz U vetores ortogonais arbitrários, como apresentado na Figura 17. Da mesma forma, os autovetores associados a autovalores nulos geram vetores ortogonais arbitrários. Assim, devido aos algoritmos da decomposição espectral e decomposição SVD serem implementados diferentes, não temos o controle dos vetores ortogonais gerados das colunas de Q e de U quando os autovalores e valores singulares são nulos, resultando vetores ortogonais diferentes. Isso justifica o porquê que apenas as submatrizes $Q[:, 0:14]eU[:, 0:14]$ conseguem ser equivalentes.

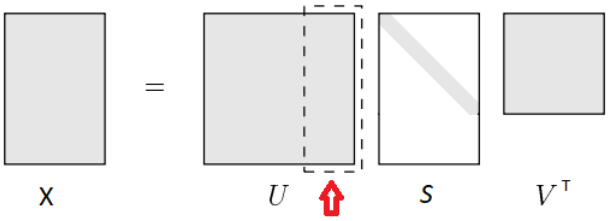

Figura 17: SVD completa

Referências

[1] K. Ozeki and T. Umeda, "An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties," *Electronics and Communications in Japan (Part I: Communications)*, vol. 67, no. 5, pp. 19–27, 1984.

[2] C. A. Callioli, H. H. Domingues, and R. C. F. Costa, *Álgebra linear e aplicações*. Atual, 2007.

[3] G. Strang, "The fundamental theorem of linear algebra," *The American Mathematical Monthly*, vol. 100, no. 9, pp. 848–855, 1993.