

Projeto Álgebra Linear

Análise dos Componentes Principais em um Conjunto de Dados

Gabriel Luiz dos Santos Silva
João Pedro Borges Baeta

06 dezembro 2022

1 Introdução

Este trabalho contém a análise de informações correspondente ao custo de vida em quase 5000 cidades espalhadas pelo mundo. Toda a análise pode ser vista no JupyterNotebok chamado "principal-book-analysis.ipynb" no Github.

Github: <https://github.com/gabrielluizone/Principal-Component-Analysis>

Os dados foram extraídos da base de dados do site Numbeo (<https://numbeo.com>), onde se disponibiliza, por meio da contribuição dos mais de 700 mil contribuidores, o custo de vida de mais de 10 mil cidades espalhadas pelo planeta. Os dados estão disponível para download no Kaggle pelo usuário mvieira101, chamado "global-cost-of-living".

Observação 1 Ao todo, foram 54 itens utilizados para o cálculo em dólar do custo de vida nas cidades espalhadas pelo mundo. Para não haver a poluição da página, os itens podem ser visto na pasta "data" contendo o dicionário das colunas no Github.

2 Ferramentas e Metodologias Adotadas

Foram utilizadas as seguintes ferramentas e bibliotecas para análise, tratamento e cálculos dos dados sobre o custo de vida das cidades no mundo:

- JupyterNotebook (Python)
- Pandas (pd)
- NumPy (np)
- Matplotlib (plt)
- Seaborn (sns)
- Função PCA de origem do Sklearn

Para a realização do estudo, é necessário atender os requisitos para a análise dos componentes principais, para isso, o conjunto de dados deve conter somente variáveis numéricas, com exceção dos países e suas cidades. Foram removidas as linhas na qual, conforme os próprios dados, eram considerados dados de qualidade baixa e que continham dados nulos.

3 Analisando os Componentes Principais

Análise 1 No primeiro passo do estudo é necessário criar a matriz de covariância, e para isso foi necessário padronizar as variáveis pela matriz de correlação, para serem comparadas entre si. Para cada variável, é subtraído pela média e dividido pelo desvio padrão.

```
# Pegando as médias das colunas
mean = np.mean(df, axis=0)

# Padronizando os dados | A matriz subtraído pelo vetor de médias, x - média
P = df - np.tile(mean, (df.shape[0], 1))

# Dividindo pelo desvio padrão
M = P / df.std()
```

$$\frac{x - \bar{x}}{\sigma^2} \quad (1)$$

Após os cálculos e a criação das seguintes variáveis, criamos a matriz de correlação.

```
# Matriz de Correlação
Cor = M.T.dot(M) / (df.shape[0] - 1)
Cor.iloc[:3, :7]
```

	Salário Líquido Médio Mensal (Após Impostos)	Taxa de Juros de Hipoteca em Percentuais (%), Anual, por 20 Anos Taxa Fixa	1 par de jeans (Levis 501 ou similar)	1 Vestido de Verão numa Rede de Lojas (Zara, H&M, ...)	1 par de tênis de corrida Nike (gama média)	1 par de sapatos masculinos de negócios em couro	Preço por metro quadrado para comprar apartamento no centro da cidade
Salário Líquido Médio Mensal (Após Impostos)	1.00	-0.47	0.30	0.22	0.18	0.48	0.56
Taxa de Juros de Hipoteca em Percentuais (%), Anual, por 20 Anos Taxa Fixa	-0.47	1.00	-0.46	-0.04	-0.02	-0.33	-0.37
1 par de jeans (Levis 501 ou similar)	0.30	-0.46	1.00	0.44	0.44	0.65	0.41

E pela utilização da variável p , foi criada a matriz de covariância.

```
# Matriz de Covariância | Matriz Transposta
Cov = P.T.dot(P) / (df.shape[0] - 1)
Cov.iloc[:3, :8]
```

	Salário Líquido Médio Mensal (Após Impostos)	Taxa de Juros de Hipoteca em Percentuais (%), Anual, por 20 Anos Taxa Fixa	1 par de jeans (Levis 501 ou similar)	1 Vestido de Verão numa Rede de Lojas (Zara, H&M, ...)	1 par de tênis de corrida Nike (gama média)	1 par de sapatos masculinos de negócios em couro	Preço por metro quadrado para comprar apartamento no centro da cidade	Preço por metro quadrado para comprar apartamento fora do centro
Salário Líquido Médio Mensal (Após Impostos)	2571744.81	-3951.15	11516.90	3804.26	10712.63	28565.92	3084135.44	1986601.26
Taxa de Juros de Hipoteca em Percentuais (%), Anual, por 20 Anos Taxa Fixa	-3951.15	27.09	-57.62	-2.08	-3.72	-64.76	-6554.99	-3793.12
1 par de jeans (Levis 501 ou similar)	11516.90	-57.62	572.90	116.13	389.66	578.70	34103.69	21396.87

Análise 2 Segundo passo: a partir da matriz de covariância, precisamos encontrar os autovalores e os autovetores, e depois ordená-los em ordem decrescente.

```
autovalores, autovetores = np.linalg.eig(Cov)
print(f'>> 1º Autovalor\n{autovalores[0]}\n\n>> 1º Autovetor\n{autovetores[0]}\n\n')

>> 1º Autovalor
156704992.4841355

>> 1º Autovetor
[-2.08397926e-02  1.32513672e-01  3.51078693e-04 -9.76190343e-02
 -6.81281743e-01  3.34672829e-01 -5.91612806e-01 -1.94091141e-01
 2.52917560e-04  8.93638757e-02  8.45937299e-03 -1.94810055e-02
 -7.30159318e-03 -2.21763276e-03  9.39626418e-03  2.87598740e-03
 2.31609087e-03 -8.01470098e-04 -6.12762612e-03 -5.11516002e-04
 2.23951482e-03  1.59960962e-03  9.59782228e-05  1.49812371e-03
 9.71359470e-04 -5.17002195e-04  4.23960169e-04 -3.63881097e-04
 -9.29496960e-04 -3.49540089e-04 -3.74068963e-04 -1.39272086e-04
 2.63234250e-04 -2.13805283e-04 -9.37843450e-05 -9.72002014e-05
 -1.62439585e-04 -2.76878873e-05 -6.53091218e-05  1.30809241e-04
 1.44280236e-06 -1.02265119e-04  1.02238544e-04  9.66598403e-05
 7.86637026e-06  6.75224834e-05 -6.43800433e-05  9.51610580e-06
 -1.68868015e-04  6.28127419e-05 -1.26563462e-04  1.09650243e-04
 -5.75824255e-05  5.76468031e-05 -4.71408652e-05]
```

```
pd.set_option('display.max_colwidth', 256)
auto = [(np.abs(autovalores[i]), autovetores[:, i]) for i in range(len(autovalores))]
auto.sort()
auto.reverse()

# Não está mostrando o vetor completamente
pd.DataFrame(auto, columns=['Autovalor', 'Autovetor']).head()
```

	Autovalor	Autovetor
0	156704992.48	[-0.020839792579188186, -4.702344748648644e-06, -0.0007892864492249364, -0.00038901939153970885, -0.0017932379099833428, -0.0013860031636851546, -0.0807702980254654, -0.04437008039227515, -0.010121018153399, -0.007330654093069946, -0.021923432506429114, ...]
1	64003332.68	[0.13251367188311608, -0.00025247454863597013, 0.0006188921959131248, 0.00016067180454763056, 1.1835344556819536e-05, 0.0013494455425645015, 0.2676542864748914, 0.17271409139218014, 0.04878610975806069, 0.03860739692749473, 0.08968387712955818, 0.06686, ...]
2	19166313.71	[0.0003510786929327981, -1.5417301122396005e-05, 0.0001614264800449917, 0.00016155097247660014, 0.002788213264671614, 0.0012001822952824744, 0.035033641519784574, -0.021368415364061543, -0.0015784174393559563, -0.0005054418916777971, -0.000615907732254, ...]
3	11013136.30	[-0.09761903426578368, 0.00026824554405801296, -0.0015975022641870438, -0.00026309604806946073, -0.000952714997102188, -0.0025540503280186617, -0.7027062221894772, -0.588729504417273, -0.052235249022261704, -0.03788015385368349, -0.09922479460569959, ...]
4	2058246.51	[-0.6812817427655917, 0.0007806973631280912, -0.0005498346772096004, -0.0004745405148846372, -0.002017405589743997, -0.005371910849906478, -0.10900793111452299, 0.45082778170406373, -0.22051880474528757, -0.1824534997359846, -0.34747747969991644, -0.28, ...]

Análise 3 Os autovalores estão representando a variabilidade dos dados, ou seja, o quão é acumulada a variância deles. A variância explicada de cada autovalor foi calcula pelo método abaixo:

```
# Soma dos Autovalores
total_sum = sum(autovalores)
print(f'Soma: {total_sum}')

# Visualização da porcentagem de variância dos dados totais que a primeira
var = [(i / total_sum) * 100 for i in sorted(autovalores, reverse=True)]
var[:4] # Pequena Amostra
```

Soma: 255233902.34962192

[61.39662131148999, 25.07634451813057, 7.509313429794129, 4.314919058342878]

	Autovalor (λ)	Variância % (fi)	Freq. Acumulada (Fi)
Componente			
PCA 1	156704992.48	61.40	61.40
PCA 2	64003332.68	25.08	86.47
PCA 3	19166313.71	7.51	93.98
PCA 4	11013136.30	4.31	98.30
PCA 5	2058246.51	0.81	99.10
...
PCA 51	0.10	0.00	100.00
PCA 52	0.16	0.00	100.00
PCA 53	0.15	0.00	100.00
PCA 54	0.12	0.00	100.00
PCA 55	0.13	0.00	100.00

Como escolhemos cerca de 93% da variância acumulada, usaremos três componentes para assim formarmos a nossa nova matriz do conjunto de dados. Logo, deve-se realizar os cálculos segundo a fórmula abaixo. Os algoritmos para os cálculos podem ser vistos no Github.

$$t_r = x \cdot w_r$$

Onde x é a nossa matriz $n \times m$ (conjunto de dados), w_r é a matriz truncada em r componentes e t é a matriz com r componentes principais.

	PCA 1	PCA 2	PCA 3
País			
India	-25008.63	-1537.78	7854.82
China	-32981.78	26214.33	3297.73
Indonesia	-38847.02	4385.68	2775.24
Philippines	-30604.36	7500.03	-2283.44
South Korea	-43407.69	19361.47	954.12
...
Canada	-40872.48	16629.89	-4483.88
Norway	-50694.56	-3168.06	-2344.55
Portugal	-37747.68	3602.78	824.73
Brazil	-29080.24	877.19	5493.99
Russia	-39169.28	3725.45	11339.59

735 rows \times 3 columns

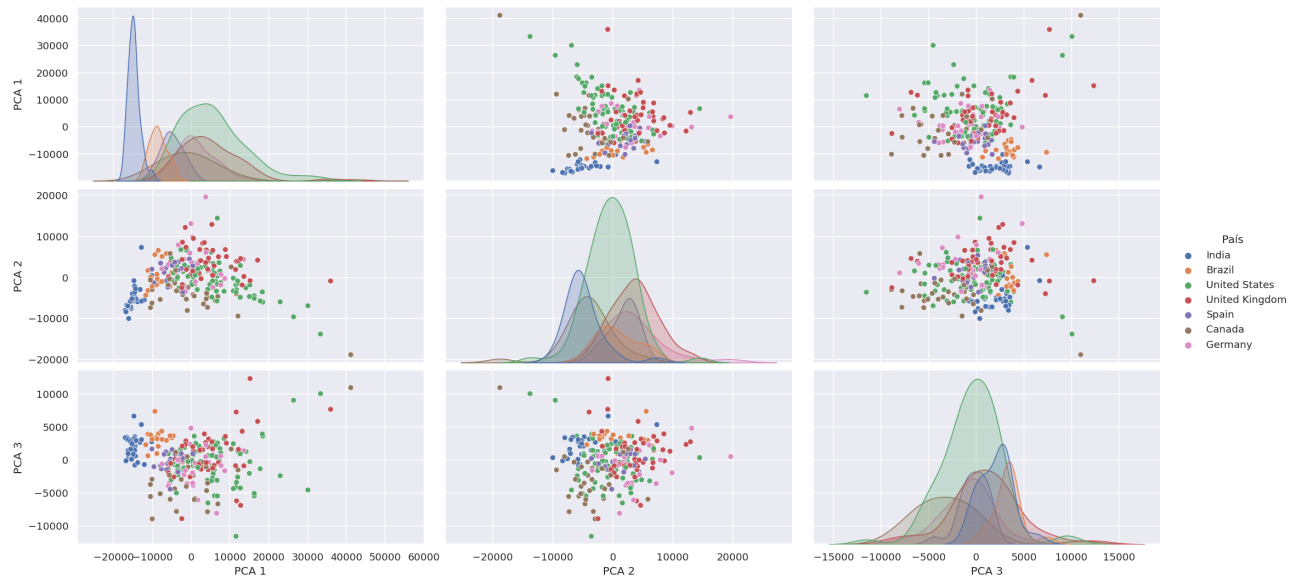
```
n_comp = pca.shape[0]
autovetores = [i[1] for i in auto]
A = autovetores

# Multiplicar a matriz pela matriz com todos os autovetores
X_ = np.dot(df, np.array(A).T)
new = pd.DataFrame(X_, columns=df.columns)
new['País'] = list(temp['País'])
new = new.set_index('País')

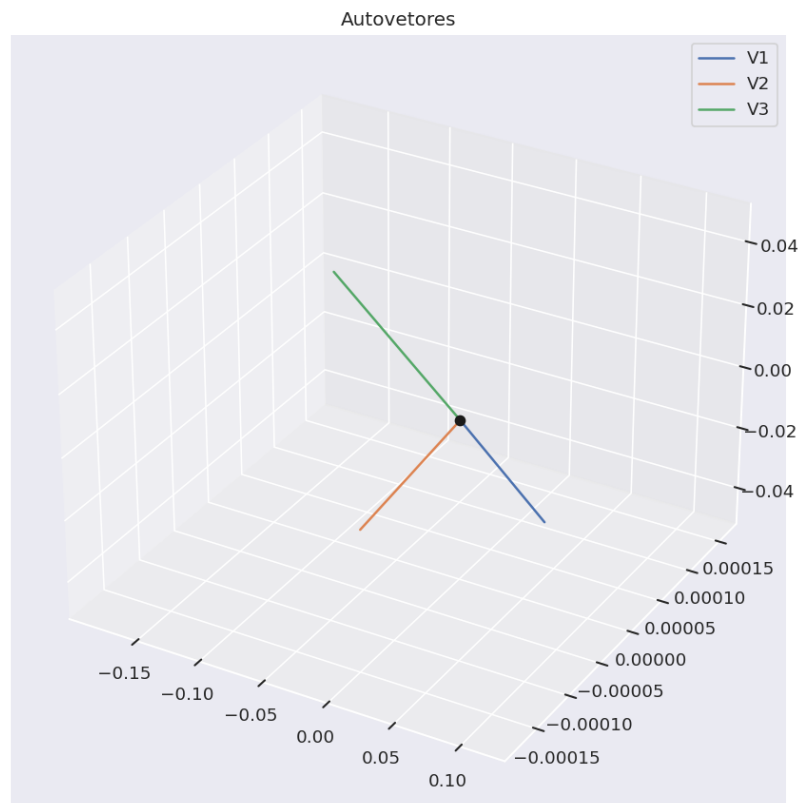
# Visualização do DataFrame de Componentes Principais
# Para deixar as coisas mais simples, colocaremos somente o país
display(new.iloc[:5, :5])
```

	Salário Líquido Médio Mensal (Após Impostos)	Taxa de Juros de Hipoteca em Percentuais (%), Anual, por 20 Anos Taxa Fixa	1 par de jeans (Levis 501 ou similar)
País			
India	-25008.63	-1537.78	7854.82
China	-32981.78	26214.33	3297.73
Indonesia	-38847.02	4385.68	2775.24
Philippines	-30604.36	7500.03	-2283.44
South Korea	-43407.69	19361.47	954.12

Análise 4 Após o processo, temos a seguinte matriz contendo os três componentes, e colocamos os países como índice para a visualização. Entretanto, para facilitar a visualização dos componentes, selecionamos alguns países para não ficar muito extenso.



Análise 5 Visualização das direções dos autovetores no 3º Plano.



4 Conclusão

Pela utilização dos componentes principais em qualquer análise, logo, temos uma visão multivariada do conjunto de dados, onde cada componente principal, é uma combinação linear das variáveis originais, e pelo primeiro componente principal, tenta explicar a variabilidade máxima do conjunto de dados, e o segundo tenta explicar o restante dela, e assim até o último. Podemos escolher a quantidade de componentes que, pela sua soma, chega perto de 100%, e pela utilização dos componentes, podemos extrair fatores importantes dos dados e produzir visualizações de dados de diversos planos, entretanto só podemos visualizar até o terceiro plano. Por essa razão a maioria das análises utiliza-se de até 3 componentes.