Principal Component Analysis - PCA

Julia Del Giudice e Sofia Iara

06 de Dezembro de 2022

0.1 Dataset

O dataset utilizado para a análise foi o Mobile Price Classification, que se trata de uma classificação de preços de celulares pelas suas especificações. Retirado do site Kaggle com 21 colunas e 2000 registros, as colunas são:

- battery power: energia total que a bateria possui. Medida em mAh.
- blue: Se tem bluetooth ou não.
- clock speed: Velocidade da execução micro-processador.
- dual sim: Se suporta dois chip ou não.
- fc: Os Mega Pixels da câmera frontal.
- four g: Se suporta 4G.
- int memory: Capacidade da memória interna em GB.
- m dep: Especificação da profundidade do celular em cm.
- mobile wt: Peso do celular.
- n cores: Números de núcleos que o processador possui.
- pc: Os Mega Pixels da câmera principal traseira.
- px height: Altura da resolução da tela.
- px width: Largura da resolução da tela.
- ram: Capacidade da memória ram em MB.
- sc h: Altura da tela do celular em cm.
- sc w: Largura da tela do celular em cm.
- talk time: Tempo de uma carga da bateria enquanto você consome.
- three g: Se suporta 3G.
- touch screen: Se suporta toque na tela.
- wifi: Se suporta wifi.
- price range: A classificação do preço, sendo 0 -¿ custo baixo; 1 -¿ custo médio; 2 -¿ custo alto; 3 -¿ custo muito alto.

0.2 Matriz de Covariância do Dataset

O primeiro passo que fizémos foi separar a coluna price range, que será nosso target, do dataset, para então calcular sua matriz de covariância:

Figura 1: Primeiro Código

```
df_mobile_prices = pd.read_csv("/kaggle/input/mobile-price-classification/train.csv")
y_price_range = df_mobile_prices['price_range']
df_mobile_prices = df_mobile_prices.drop('price_range',axis=1)
df_mobile_prices
```

Figura 2: Código da Matriz de Covariância

```
cov_matriz = df_mobile_prices.cov()
```

+ Code + Markdown

```
cov_matriz
```

Figura 3: Matriz de Covariância

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	pc	px_height	px_width	ram	sc_h	sc_w	talk_time	three_g	touch_screen	wifi
battery_power	193088.359838	2.472579	4.116922	-9.194773	63.592320	3.439322	-31.923572	4.319702	28.689738	-29.885322	83.782186	2905.736870	-1595.644609	-3.112180e+02	-55.464653	-41.005692	126.075334	2.158241	-2.310961	-1.833296
blue	2.472579	0.250100	0.008741	0.008802	0.007801	0.003359	0.373669	0.000584	-0.152331	0.041373	-0.030183	-1.525223	-8.977161	1.429471e+01	-0.006221	0.001336	0.038074	-0.006446	0.002516	-0.005468
clock_speed	4.116922	0.008741	0.665863	-0.000537	-0.001537	-0.017562	0.096914	-0.003381	0.356738	-0.010686	-0.025955	-5.259133	-3.341841	3.047585e+00	-0.099970	-0.026229	-0.050970	-0.016151	0.008062	-0.009986
dual_sim	-9.194773	0.008802	-0.000537	0.250035	-0.063222	0.000796	-0.142263	-0.003193	-0.158945	-0.028209	-0.051983	-4.632342	3.088397	2.227762e+01	-0.025174	-0.036305	-0.107658	-0.002986	-0.004281	0.005686
fc	63.592320	0.007801	-0.001537	-0.063222	18.848134	-0.035922	-2.295039	-0.002243	3.629749	-0.132661	16.970829	-19.247050	-9.711403	7.110563e+01	-0.201462	-0.234004	-0.161985	0.003317	-0.032195	0.043605
four_g	3.439322	0.003359	-0.017562	0.000796	-0.035922	0.249663	0.078790	-0.000263	-0.292500	-0.033958	-0.015963	-4.265455	1.608471	3.963902e+00	0.057189	0.080550	-0.127300	0.124440	0.004188	-0.004403
int_memory	-31.923572	0.373669	0.096914	-0.142263	-2.295039	0.078790	329.266971	0.036037	-21.977567	-1.175291	-3.661448	84.080518	-65.366654	6.458695e+02	2.887692	0.927298	-0.276650	-0.072446	-0.245012	0.063456
m_dep	4.319702	0.000584	-0.003381	-0.003193	-0.002243	-0.000263	0.036037	0.083184	0.222125	-0.002312	0.045969	3.233478	2.937617	-2.951498e+00	-0.030802	-0.023104	0.026794	-0.001483	-0.000380	-0.004089
mobile_wt	28.689738	-0.152331	0.356738	-0.158945	3.629749	-0.292500	-21.977567	0.222125	1253.135567	-1.537873	4.045314	14.756486	1.373327	-9.909058e+01	-5.049343	-3.201584	1.200861	0.023398	-0.254374	-0.007247
n_cores	-29.885322	0.041373	-0.010686	-0.028209	-0.132661	-0.033958	-1.175291	-0.002312	-1.537873	5.234197	-0.016547	-6.977203	24.205785	1.208167e+01	-0.003035	0.257405	0.164357	-0.014368	0.027202	-0.011399
pc	83.782186	-0.030183	-0.025955	-0.051983	16.970829	-0.015963	-3.661448	0.045969	4.045314	-0.016547	36.775916	-49.694829	10.997543	1.906581e+02	0.126156	-0.629270	0.485661	-0.003416	-0.026513	0.016343
px_height	2905.736870	-1.525223	-5.259133	-4.632342	-19.247050	-4.265455	84.080518	3.233478	14.756486	-6.977203	-49.694829	196941.408040	97946.365509	-9.797075e+03	111.466131	83.205267	-25.812594	-5.897191	4.858605	11.500994
px_width	-1595.644609	-8.977161	-3.341841	3.088397	-9.711403	1.608471	-65.366654	2.937617	1.373327	24.205785	10.997543	97946.365509	186796.361941	1.924610e+03	39.330164	65.332778	15.869264	0.064479	-0.351972	6.552918
ram	-311.218050	14.294712	3.047585	22.277615	71.105629	3.963902	645.869530	-2.951498	-99.090582	12.081674	190.658115	-9797.075042	1924.610004	1.176644e+06	73.106269	168.113686	64.129222	7.303452	-16.521400	12.296657
sc_h	-55.464653	-0.006221	-0.099970	-0.025174	-0.201462	0.057189	2.887692	-0.030802	-5.049343	-0.003035	0.126156	111.466131	39.330164	7.310627e+01	17.751433	9.290060	-0.399071	0.021611	-0.042191	0.054632
sc_w	-41.005692	0.001336	-0.026229	-0.036305	-0.234004	0.080550	0.927298	-0.023104	-3.201584	0.257405	-0.629270	83.205267	65.332778	1.681137e+02	9.290060	18.978200	-0.543209	0.057458	0.027713	0.077170
talk_time	126.075334	0.038074	-0.050970	-0.107658	-0.161985	-0.127300	-0.276650	0.026794	1.200861	0.164357	0.485661	-25.812594	15.869264	6.412922e+01	-0.399071	-0.543209	29.854806	-0.099426	0.046990	-0.080617
three_g	2.158241	-0.006446	-0.016151	-0.002986	0.003317	0.124440	-0.072446	-0.001483	0.023398	-0.014368	-0.003416	-5.897191	0.064479	7.303452e+00	0.021611	0.057458	-0.099426	0.181709	0.002967	0.000920
touch_screen	-2.310961	0.002516	0.008062	-0.004281	-0.032195	0.004188	-0.245012	-0.000380	-0.254374	0.027202	-0.026513	4.858605	-0.351972	-1.652140e+01	-0.042191	0.027713	0.046990	0.002967	0.250116	0.002980
wifi	-1.833296	-0.005468	-0.009986	0.005686	0.043605	-0.004403	0.063456	-0.004089	-0.007247	-0.011399	0.016343	11.500994	6.552918	1.229666e+01	0.054632	0.077170	-0.080617	0.000920	0.002980	0.250076

0.3 Calculando os Autovalores e Autovetores

O terceiro passo foi calcular tanto os autovalores quanto os autovetores:

Figura 4: Código para o Cálculo dos Autovalores e Autovetores

```
autovalores, autovetores = np.linalg.eig(cov_matriz)
```

Figura 5: Autovalores do Dataset

	0
0	1.176743e+06
1	1.931781e+05
2	2.899194e+05
3	9.363010e+04
4	1.253700e+03
5	3.283495e+02
6	4.682288e+01
7	2.991329e+01
8	2.732790e+01
9	9.155218e+00
10	8.489138e+00
11	5.204934e+00
12	6.671008e-01
13	8.238436e-02
14	8.632131e-02
15	3.411977e-01
16	2.325577e-01
17	2.579842e-01
18	2.529622e-01
19	2.497154e-01

Figura 6: Autovetores do Dataset

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	0.000347	0.999442	-0.010388	-3.171326e-02	-0.000157	-0.000192	0.000562	-0.000713	0.000184	-0.000096	0.000033	0.000154	0.000019	-0.000019	0.000007	0.000028	-0.000035	0.000020	0.000028	0.000003
1	-0.000012	0.000014	0.000025	5.856317e-05	-0.000128	-0.001087	0.000378	0.001294	0.000087	0.002006	-0.001794	0.008930	-0.020914	0.005317	0.042447	-0.001467	0.562971	0.708943	-0.215245	-0.363006
2	-0.000003	0.000021	0.000021	-1.396164e-05	0.000283	-0.000323	0.000583	-0.001137	-0.003497	0.006207	0.001643	-0.002223	-0.996589	0.007212	0.001388	-0.070942	0.005930	-0.011324	0.020131	0.033084
3	-0.000019	-0.000048	0.000004	-5.306232e-05	-0.000121	0.000465	0.001722	-0.002982	-0.002654	-0.002478	0.002256	-0.006254	0.001343	0.020247	0.011500	0.028885	-0.595919	0.696633	0.302352	0.258300
4	-0.000061	0.000329	0.000067	-8.234839e-05	0.003029	0.007899	-0.515805	-0.024368	0.001125	0.370909	-0.771300	0.027479	0.000790	0.001721	-0.002327	-0.002171	-0.005478	0.001075	-0.000729	-0.000357
5	-0.000003	0.000017	0.000007	-4.465700e-05	-0.000235	-0.000234	0.000839	-0.005072	0.002387	-0.000079	0.002923	-0.007226	0.055760	-0.142333	-0.589220	-0.790185	0.022362	0.045263	0.004973	0.048919
6	-0.000548	-0.000147	-0.000068	1.190878e-03	-0.023736	-0.999553	-0.015246	0.002394	-0.007230	0.004442	0.002049	0.003536	0.000360	-0.000074	0.000283	-0.000016	-0.001235	-0.000513	0.000505	-0.000112
7	0.000003	0.000022	-0.000015	-6.542333e-07	0.000176	-0.000129	-0.000803	0.001077	-0.001096	-0.000496	0.003136	-0.000440	0.005711	0.970338	-0.239749	0.003356	0.002261	-0.007259	-0.027918	-0.008326
8	0.000084	0.000150	-0.000039	8.096726e-05	0.999694	-0.023840	0.004246	-0.001924	0.004160	-0.001390	0.000288	0.001281	0.000273	-0.000198	-0.000097	-0.000149	-0.000096	0.000155	0.000089	-0.000188
9	-0.000010	-0.000159	-0.000039	-2.275484e-04	-0.001205	0.003702	0.002183	0.005331	0.009769	0.028690	0.047837	0.998264	-0.001106	0.000037	-0.002142	-0.007026	-0.005802	-0.001324	0.004303	0.010199
10	-0.000162	0.000429	0.000091	-4.638556e-04	0.003445	0.012811	-0.856097	-0.007155	0.017934	-0.209297	0.471710	-0.014881	-0.001102	-0.001981	0.001290	0.000501	0.001623	0.001072	0.000365	-0.000227
11	0.009903	0.014308	-0.725036	6.884899e-01	-0.000061	0.000877	-0.000379	0.000398	-0.000599	0.000163	0.000200	0.000117	-0.000018	-0.000010	0.000013	-0.000057	-0.000049	0.000018	-0.000039	0.000049
12	-0.000965	-0.030147	-0.688601	-7.245119e-01	0.000016	-0.000816	0.000212	-0.000202	-0.000095	-0.000135	-0.000211	-0.000177	-0.000009	-0.000010	0.000005	0.000033	0.000046	0.000013	-0.000023	-0.000056
13	-0.999950	0.000518	-0.006520	7.505924e-03	0.000097	0.000554	0.000172	-0.000013	-0.000159	-0.000038	-0.000050	-0.000006	0.000002	0.000001	-0.000004	0.000004	-0.000004	-0.000023	-0.000008	-0.000015
14	-0.000061	-0.000285	-0.000372	5.401017e-04	-0.004152	-0.008610	0.003586	-0.160132	0.663484	-0.659812	-0.312752	0.028354	-0.006954	0.001819	0.000682	0.000085	-0.001832	0.000291	0.000758	-0.003998
15	-0.000142	-0.000216	-0.000366	1.337468e-04	-0.002629	-0.002648	0.023758	-0.178192	0.708390	0.618217	0.286699	-0.037517	0.001766	0.000216	-0.000341	0.005189	-0.001755	0.001554	-0.003542	0.002899
16	-0.000055	0.000648	0.000021	-3.502859e-04	0.000980	0.000784	-0.014267	0.970492	0.239690	0.012246	-0.015110	-0.007220	-0.001354	-0.000560	0.000164	-0.005303	-0.000948	0.001792	0.003197	0.003710
17	-0.000006	0.000011	0.000014	-4.400867e-05	0.000020	0.000213	0.000216	-0.003873	0.001280	0.002192	0.000611	-0.003295	0.048153	0.193243	0.769895	-0.602461	0.011444	-0.018089	0.031053	0.056439
18	0.000014	-0.000012	-0.000011	3.790510e-05	-0.000200	0.000752	0.000769	0.001762	0.000100	0.004600	0.004283	0.004693	-0.018509	0.002396	0.000478	-0.069020	-0.326340	-0.092393	0.327249	-0.879031
19	-0.000010	-0.000010	-0.000045	3.547184e-05	-0.000007	-0.000157	-0.000777	-0.003283	0.002273	0.002698	-0.002409	-0.002253	0.022611	0.018316	-0.025573	0.043517	0.469787	-0.031611	0.867724	0.148079

0.4 Selecionando os Dois Maiores Autovalores

O quarto passo foi ordenar os autovalores e autovetores do maior para o menor e calcular a variância explicada e a variância explicada cumulativa que seriam o quanto cada componente (autovetores) está representando a variabilidade do dataset.

Figura 7: Código dos Autovalores e Autovetores e da Variância Explicada

```
pares_autos = [
    (
        np.abs(autovalores[i]),
        autovetores[:,i]
    ) for i in range (len(autovalores))
]
pares_autos.sort()
pares_autos.reverse()
total = sum(autovalores)
var_exp = [
    (i/total)*100 for i in sorted(autovalores, reverse=True)
]
cum_var_exp = np.cumsum(var_exp)
```

```
df_temp = pd.DataFrame(
    {'autovalores': autovalores,
     'cum_var_exp': cum_var_exp,
     'var_esp': var_exp})
print(df_temp)
print('\nAuto-vetores')
for autovetor in [p[1] for p in pares_autos]:
    print(autovetor)
```

Figura 8: Autovalores (Sendo grifado os dois maiores) e Autovetores e da Variância Explicada

	autovalores	cum_var_exp	var_esp
0	1.176743e+06	67.043941	67.043941
1	1.931781e+05	83.561854	16.517913
2	2.899194e+05	94.568014	11.006160
3	9.363010e+04	99.902510	5.334496
4	1.253700e+03	99.973938	0.071429
5	3.283495e+02	99.992646	0.018707
6	4.682288e+01	99.995313	0.002668
7	2.991329e+01	99.997018	0.001704
8	2.732790e+01	99.998575	0.001557
9	9.155218e+00	99.999096	0.000522
10	8.489138e+00	99.999580	0.000484
11	5.204934e+00	99.999876	0.000297
12	6.671008e-01	99.999914	0.000038
13	8.238436e-02	99.999934	0.000019
14	8.632131e-02	99.999948	0.000015
15	3.411977e-01	99.999963	0.000014
16	2.325577e-01	99.999977	0.000014
17	2.579842e-01	99.999990	0.000013
18	2.529622e-01	99.999995	0.000005
19	2.497154e-01	100.000000	0.000005

Como é possível observar, com os dois maiores autovalores temos variância explicada cumulativa de 83.56%.

0.5 Colunas Referentes aos Autovalores

Para partir para a plotagem das colunas correspondentes aos dois maiores autovalores, foi preciso transformar o dataset com os autovalores mencionados e

adicionar a coluna price range neste dataset transformado.

Figura 9: Transformação para o Dataset

```
n_componentes = 2
autovetores = [p[1] for p in pares_autos]
A = autovetores[0:n_componentes]
X = np.dot(df_mobile_prices,np.array(A).T)
novo_df_mobile_prices = pd.DataFrame(X, columns=['pc1','pc2'])
novo_df_mobile_prices['price_range'] = y_price_range
```

```
novo_df_mobile_prices
```

Figura 10: Dataset com as Colunas Correspondentes

	pc1	pc2	price_range
0	-2549.103138	-560.461995	1
1	-2623.490779	-2052.872594	2
2	-2591.835872	-2120.192363	2
3	-2758.328368	-2135.941407	2
4	-1399.520860	-1738.554752	1
...	...	...	...
1995	-657.409877	-2200.063364	0
1996	-2024.060277	-2050.188032	2
1997	-3049.175309	-1792.921678	3
1998	-865.768428	-726.366114	0
1999	-3914.587565	-900.262791	3

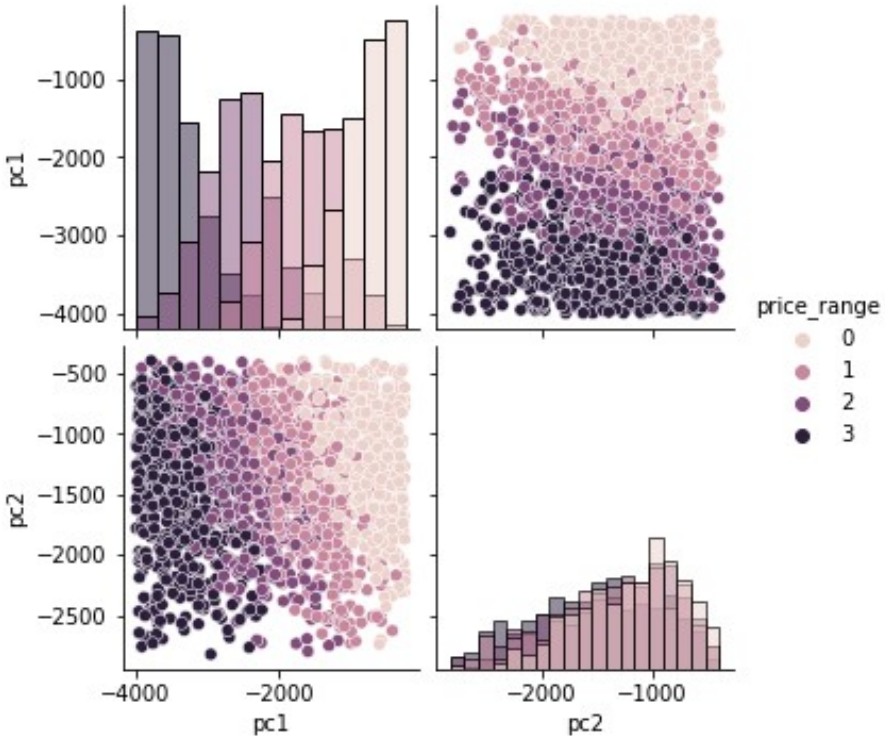
0.6 Plotagem das Colunas Correspondentes dos maiores Autovalores em Relação ao Dataset

O último passo é trazer esse dataset transformado em gráfico.

Figura 11: Código com a Plotagem para o Gráfico

```
sns.pairplot(
    novo_df_mobiles_prices, vars=['pc1','pc2'],
    hue='price_range',diag_kind="hist")
plt.show()
```

Figura 12: Gráfico das Colunas Referentes aos Dois Maiores Autovalores



0.7 Link Repositório GitHub

Para acesso ao notebook, segue o link do repositório:

- https://github.com/Juhdelg/pca_algebra