I2A2 - Genomic Analysis Challenge

Identificação

Nome: Weld Lucas Cunha
E-mail: cunha.lucas012@gmail.com

Informações Gerais

O arquivo "readcounts.xlsx" foi obtido a partir da atividade genética decorrente da transcrição de DNA->RNA de um conjunto de células de diferentes indivíduos. No total temos por volta de 65 mil genes mapeados de 45 pessoas diferentes.

O que queremos saber é quais pessoas são parentes de quais pessoas, e por extensão, quem não tem parentesco. Para efeito de análise, podemos compreender que pessoas que são parentes, possuem características genéticas similares.

Na entrega, além do código em formato jupyter notebook, deve-se montar um relatório (que pode ser derivado do próprio notebook) utilizando a metodologia CRISP-DM.

Entendimento do Problema

Para realizar este desafio é necessário aplicar alguma técnica de redução de dimensionalidade, devido ao grande número de genes (características) disponíveis; Após a redução de demensionalidade devemos agrupar os pontos (clusterização) visando encontrar os grupos com características semelhanças.

Espera-se que alguns sujeitos no conjunto de dados não tenham parentes, então, neste caso, é desejável deixá-los fora de qualquer cluster. Algoritmos de clustering como DBScan ou HDBScan são capazes de deixar algumas amostras “não agrupadas”. Definir o valor eps para o algoritmo DBScan pode ser complicado às vezes, especialmente quando os dados são desconhecidos, como neste caso. Portanto, HDBScan pode ser uma melhor opção, já que seu algoritmo é capaz de encontrar os clusters mais estáveis nos dados sem definir muitos parâmetros.

Além disso, diferentes técnicas podem ser usadas para redução de dimensionalidade, como PCA, ICA, UMAP, etc. E não há garantia de que uma técnica específica seja a melhor. Mas, espera-se que amostras semelhantes ainda sejam semelhantes no espaço de novos recursos para a maioria das técnicas aplicadas.

Entendimento dos Dados

Formato original dos dados:

	Unnamed: 0	H223	H224	H225	H226	H227	H228	H229	H230	H231	...	H261	H262	H263	H264	H265	H266	H267	H268	H269	H270
0	ENSG00000000003	0	0	0	1	0	0	0	0	1	...	0	1	0	1	0	2	0	0	1	0
1	ENSG00000000005	0	0	0	0	0	0	0	0	0	...	1	0	0	0	0	0	0	0	0	0
2	ENSG00000000419	1216	1228	1022	912	491	449	466	727	774	...	980	932	360	450	484	926	803	630	537	582
3	ENSG00000000457	189	114	110	289	186	148	169	258	145	...	117	286	137	90	105	275	101	56	87	81
4	ENSG00000000460	74	38	55	127	30	17	45	100	33	...	28	157	34	20	15	139	54	25	21	47
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
65210	ENSG00000281918	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
65211	ENSG00000281919	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
65212	ENSG00000281920	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
65213	ENSG00000281921	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
65214	ENSG00000281922	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

65215 rows × 49 columns

A planilha dos dados não está no formato adequado, a seguir mostramos uma amostra do dataframe ajustado:

	ENSG00000000003	ENSG00000000005	ENSG00000000419	ENSG00000000457	ENSG00000000460	ENSG00000000938	ENSG00000000971	ENSG00000001036	ENSG00000001084	ENSG00000001167	...	ENSG00000281913
H223	0	0	1216	189	74	31895	2	763	8963	1589	...	0
H224	0	0	1228	114	38	23361	3	712	26794	1734	...	0
H225	0	0	1022	110	55	27944	0	956	8027	1341	...	0
H226	1	0	912	289	127	41846	6	1104	716	1053	...	0
H227	0	0	491	186	30	11929	14	136	745	730	...	0
H228	0	0	449	148	17	6856	16	227	3770	663	...	0
H229	0	0	466	169	45	6756	15	217	1260	669	...	0
H230	0	0	727	258	100	7668	4	905	1480	883	...	0
H231	1	0	774	145	33	9315	1	94	75	873	...	0
H232	0	0	576	131	8	3319	7	88	288	678	...	0
H233	0	0	547	163	32	4788	3	73	81	778	...	0
H234	1	0	1111	248	99	12703	6	1413	809	1062	...	0
H235	0	0	681	98	27	32653	0	623	2248	619	...	0
H236	0	0	796	85	28	27954	0	737	8879	880	...	0
H237	1	0	799	63	26	22707	2	808	4314	736	...	0
H238	2	0	542	155	108	27262	0	876	579	690	...	0
H239	0	0	499	85	13	11143	4	141	462	486	...	0
H240	0	0	495	65	7	8381	3	271	1966	520	...	0
H241	0	0	398	77	15	7737	4	191	567	438	...	0
H242	0	0	593	131	48	10164	0	739	1305	674	...	0
H243	0	0	656	115	21	12241	0	73	51	731	...	0
H244	0	0	714	80	16	9937	1	104	365	729	...	0
H245	1	0	691	82	29	8805	0	109	87	817	...	0
H246	0	0	921	143	89	11099	0	1202	542	557	...	0
H247	0	0	881	138	42	40424	2	590	11814	1397	...	0
H248	0	0	916	98	20	27829	1	469	40184	1562	...	0

Uma simples análise dos dados nos mostram algumas informações importantes:

```
data.info()
```

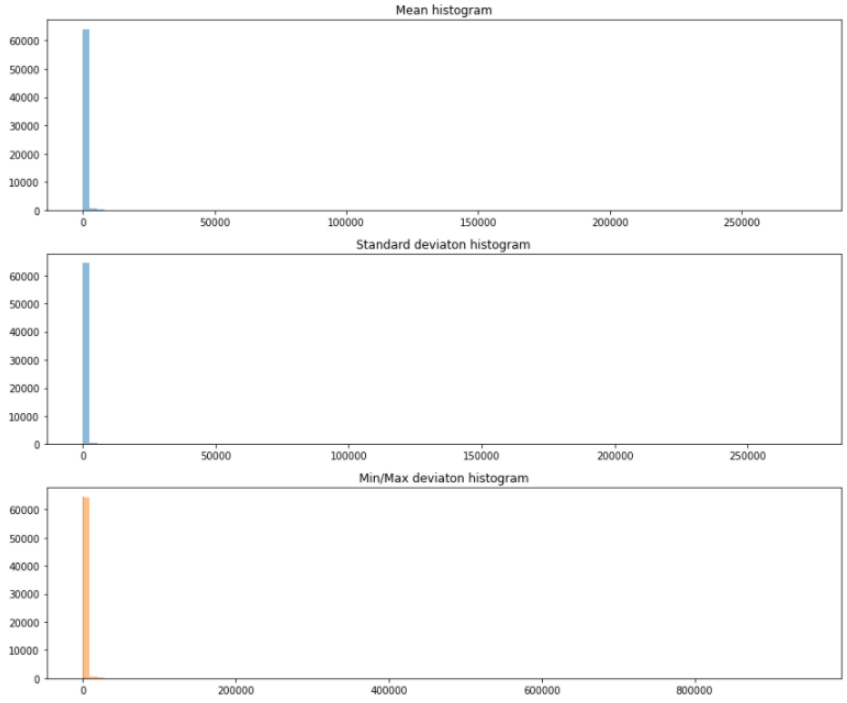
```
<class 'pandas.core.frame.DataFrame'>
Index: 48 entries, H223 to H270
Columns: 65215 entries, ENSG00000000003 to ENSG00000281922
dtypes: int64(65215)
memory usage: 23.9+ MB
```

```
data_descr = data.describe().transpose()
data_descr
```

	count	mean	std	min	25%	50%	75%	max
ENSG00000000003	48.0	0.250000	0.525924	0.0	0.00	0.0	0.0	2.0
ENSG00000000005	48.0	0.041667	0.201941	0.0	0.00	0.0	0.0	1.0
ENSG00000000419	48.0	709.125000	224.728453	318.0	540.75	686.0	884.0	1228.0
ENSG00000000457	48.0	136.395833	62.975760	56.0	90.00	114.5	157.0	289.0
ENSG00000000460	48.0	44.916667	35.518550	7.0	21.00	31.5	54.0	157.0
...	...	...	...	...	...	...	...	...
ENSG00000281918	48.0	0.041667	0.201941	0.0	0.00	0.0	0.0	1.0
ENSG00000281919	48.0	0.000000	0.000000	0.0	0.00	0.0	0.0	0.0
ENSG00000281920	48.0	0.020833	0.144338	0.0	0.00	0.0	0.0	1.0
ENSG00000281921	48.0	0.000000	0.000000	0.0	0.00	0.0	0.0	0.0
ENSG00000281922	48.0	0.000000	0.000000	0.0	0.00	0.0	0.0	0.0

65215 rows × 8 columns

O gráfico a seguir mostra um histograma com o valores médios, desvio padrão, e valores máximos/mínimos para cada variável.



Observa-se uma grande quantidade de valores próximos de zero, enquanto alguns poucos valores são relativamente grandes (na faixa dos milhares). Ao todo, 20537 colunas tem desvio padrão igual a zero, e portanto podem ser removidas da nossa análise.

```
zero_std_cols = list(data_descr[data_descr['std'] <= 0].index)
print(f'{len(zero_std_cols)} columns have standard deviation = 0')
```

```
20537 columns have standard deviation = 0
```

Preparação dos Dados

Na etapa de preparação dos dados, temos a eliminação das features com baixa variância (zero), e posteriormente é realizada a normalização/padronização dos dados. São realizadas 2 tipos de escalonamentos (para comparação), mas a padronização é a técnica utilizada no restante das análises.

```
X_array = data.loc[:, data.columns].values
X_array.shape
```

```
(48, 65215)
```

```
# Feature seletion:
sel = VarianceThreshold(threshold=0)
X_array_sel = sel.fit_transform(X_array)
X_array_sel.shape
```

```
(48, 44678)
```

```
# Normalizing data:
x_array_norm = MinMaxScaler().fit_transform(X_array_sel)
pd.DataFrame(x_array_norm)
```

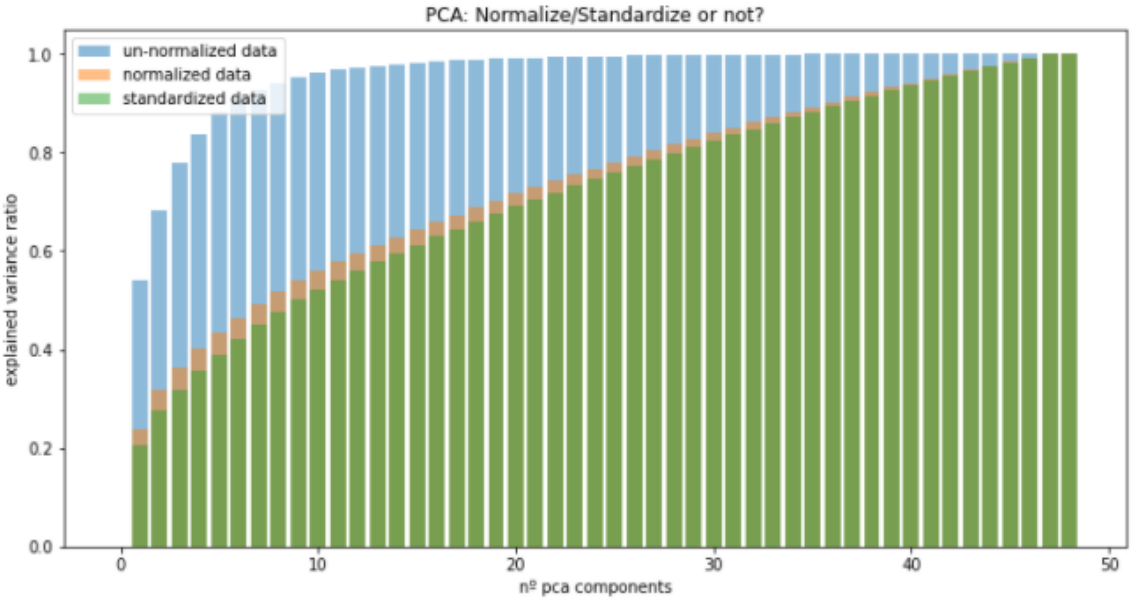
```
# Standardizing data:
x_array_std = StandardScaler().fit_transform(X_array_sel)
pd.DataFrame(x_array_std)
```

Modelagem

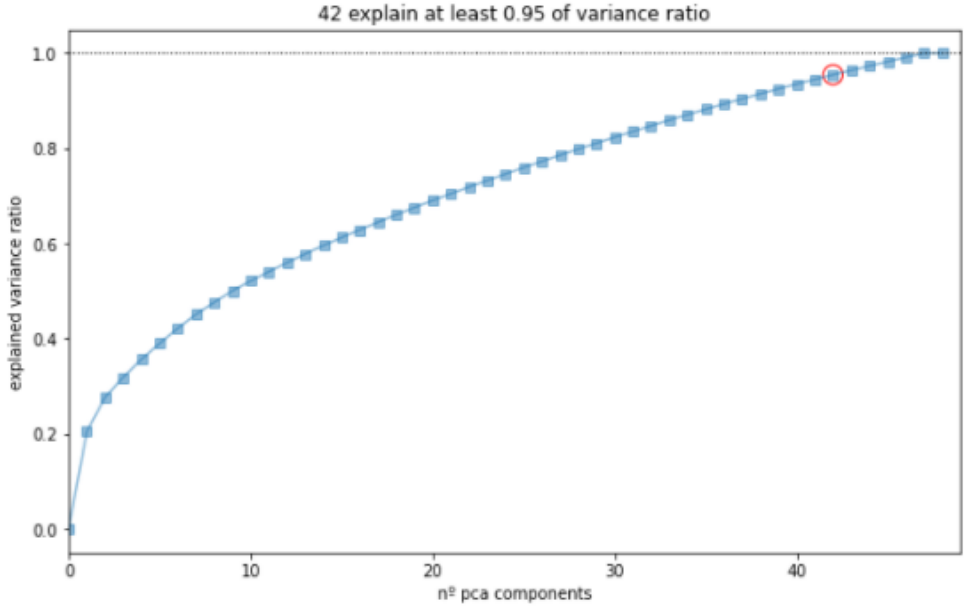
Foram realizadas três transformações do PCA (uma para cada conjuntos de dados), visando identificar quantas componentes são necessárias para explicar pelo menos 95% das características presentes nos dados.

```
n_components = 48

# PCA:
pca_obj1 = PCA(n_components=n_components)
x_pca = pca_obj1.fit_transform(X_array_sel)

pca_obj2 = PCA(n_components=n_components)
x_pca_norm = pca_obj2.fit_transform(x_array_norm)

pca_obj3 = PCA(n_components=n_components)
x_pca_strd = pca_obj3.fit_transform(x_array_std)
```

O gráfico a seguir mostra a razão de explicabilidade da variância para as três transformações.



A principal razão de normalizar/padronizar os dados é evitar que alguma feature se torne mais relevante durante a aplicação do PCA apenas devido à diferença de escala. Sendo assim, utilizando as componentes do pca aplicada aos dados padronizados, temos a seguir o gráfico que nos mostra quantas componentes são necessárias para se explicar pelo menos 95% da variância presentes nos dados originais.

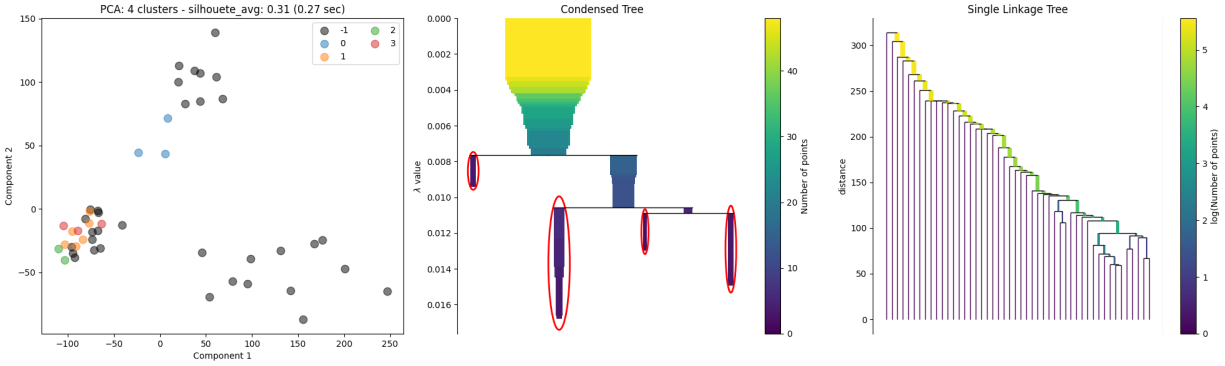


Avaliação

Após a etapa de modelagem, foi realizada a clusterização, utilizando o algoritmo HDBSCAN. Este algoritmo foi escolhido por se tratar de um algoritmo bastante poderoso, e com capacidade de trabalhar em altas dimensões: HDBSCAN, ou seja, DBSCAN hierárquico, é um poderoso algoritmo de agrupamento baseado em densidade que é: 1) indiferente à forma dos clusters, 2) não requer que o número de clusters seja especificado, 3) robusto em relação aos clusters com densidade diferente. Além disso, o HBDSCAN é muito atraente porque tem apenas um hiperparâmetro min_cluster_size, que é o número mínimo de pontos em um cluster. É relativamente rápido para grandes conjuntos de dados, detecta células periféricas e, para cada célula, relata uma probabilidade de

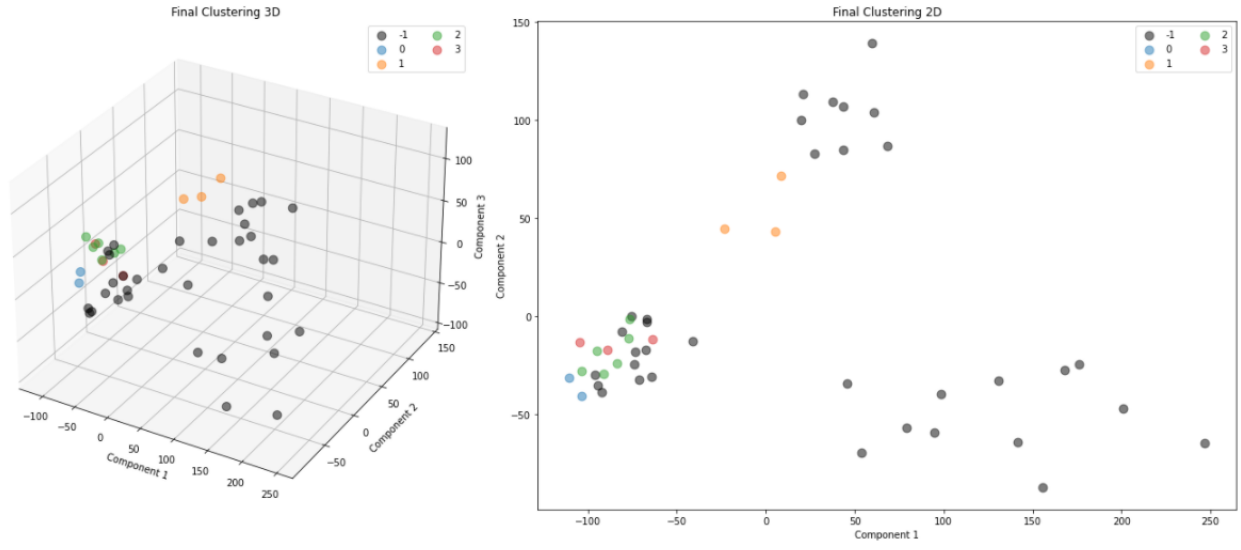
atribuição a um cluster. A fração de células com baixa probabilidade de atribuição a um cluster pode ser usada como uma função objetivo para otimizar min_cluster_size que, por sua vez, fornece o número ideal de clusters.

Durante a execução e análise dos resultados obtidos através da clusterização, observamos que com 42 componentes os resultados não estavam satisfatórios, visando obter resultados melhores, este valor foi reduzido para 30. Com esta quantidade de componentes do PCA, os resultados dos clusters se mostraram bem mais satisfatórios, ainda assim é garantida aproximadamente 80% da explicabilidade da variância. Abaixo temos o resultado da clusterização realizada pelo HDBSCAN:



Os elementes presentes em cada cluster são mostrados a seguir, (lembrando que o grupo -1 representa aqueles elementos que não fazem parte de nenhum cluster, e portanto não possuem parentesco com nenhum outro elemento da base de dados:

- -1 ['H223' 'H224' 'H225' 'H226' 'H227' 'H228' 'H229' 'H230' 'H231' 'H234' 'H238' 'H242' 'H243' 'H244' 'H245' 'H246' 'H247' 'H248' 'H249' 'H250' 'H251' 'H252' 'H253' 'H254' 'H255' 'H256' 'H257' 'H258' 'H259' 'H260' 'H261' 'H262' 'H266' 'H270']
- 0 ['H235' 'H236' 'H237']
- 1 ['H239' 'H240' 'H241' 'H263' 'H264' 'H265']
- 2 ['H232' 'H233']
- 3 ['H267' 'H268' 'H269']



Implantação

Não se aplica para este caso.