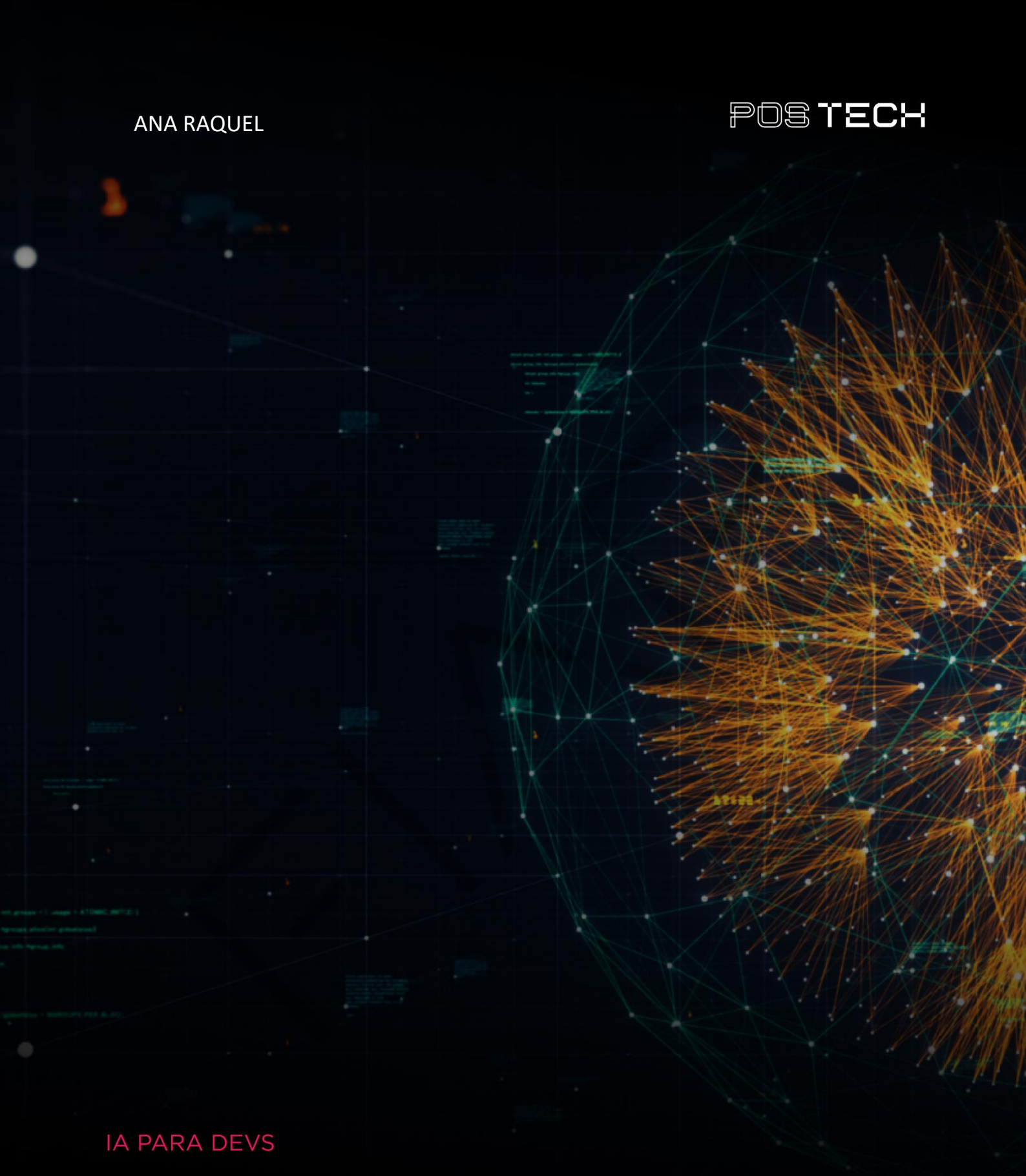
ANA RAQUEL



IA PARA DEVS

MACHINE LEARNING AVANÇADO

AULA 03

SUMÁRIO

O QUE VEM POR AÍ?

Nessa aula, você irá aprender sobre os modelos não supervisionados. Será que os modelos não supervisionados possuem o mesmo comportamento que os algoritmos supervisionados?

HANDS ON

Você aprendeu alguns poderosos algoritmos e técnicas para lidar com modelos não supervisionados. Agora, vamos aprender a como aplicar essas técnicas no Python utilizando um case de negócio?

Para essa aula, temos alguns notebooks para você. Acesse abaixo:

- Notebook 1
- Notebook 2

Além disso, também disponibilizamos as bases de dados, para te ajudar com os estudos e exercícios:

- Base de dados 1
- Base de dados 2
- Base de dados 3
- Base de dados 4

SAIBA MAIS

Aprendizado não supervisionado

Você aprendeu como podemos solucionar problemas com modelos supervisionados, mas como podemos definir o aprendizado não supervisionado?

O termo não supervisionado se refere a métodos estatísticos que extraem significado dos dados sem treinar um modelo em dados rotulados (dados em que o resultado de interesse é conhecido). Os modelos não supervisionados também constroem um modelo dos dados, mas não distinguem entre variáveis de características e variáveis preditoras.

O aprendizado não supervisionado pode ter diferentes objetivos possíveis. Em alguns casos, pode ser utilizado para criar uma regra preditiva, na ausência de uma resposta rotulada. Os métodos de agrupamento podem ser utilizados para identificar grupos de dados significativos. Por exemplo, podemos extrair de uma base de consumidores de um shopping, segmentação de clientes com base em comportamentos similares, criando assim grupos de clientes.

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

Figura 1 - Exemplo de base não supervisionada
Fonte: Elaborado pela autora (2024)

Perceba que nessa base de dados, não temos a variável target nos dados (variável resposta), então não conseguimos identificar um rótulo que classifique os consumidores por grupos.

Em outros casos, podemos utilizar os modelos não supervisionados para reduzir a dimensão dos dados para um conjunto de dados reduzido, tornando-o mais gerenciável de variáveis. Esse conjunto de dados reduzido poderia então ser utilizado

como entrada para um modelo preditivo, como regressão ou classificação por exemplo.

Por fim, podemos também concluir que modelos não supervisionados podem ser vistos como uma extensão da análise exploratória de dados em situações que somos confrontados com um grande volume de diferentes variáveis que se relacionam entre si.

Vamos aprender um algoritmo não supervisionado?

K-Means

O agrupamento é uma **técnica para dividir os dados em diferentes grupos, na qual os registros em cada grupo são semelhantes uns aos outros**. Os grupos podem ser usados diretamente, analisando mais a fundo ou passados como uma característica ou resultado para um modelo de regressão ou classificação. O K-means foi o primeiro método de agrupamento desenvolvido (referências originais datam de 1956, 1965 e 1967).

O algoritmo K-Means divide os dados em **k grupos** através da **minimização da soma das distâncias quadráticas** de cada registro à medida de seu grupo atribuído. Isso é chamado de soma dos quadrados dentro do grupo, ou SSE dentro do grupo. O algoritmo K-Means não garante que os grupos tenham o mesmo tamanho, mas encontra grupos que sejam melhores separados.

O algoritmo inicializa centroides (pontos centros nos dados) de forma aleatória e cria grupos de dados com base em alguma métrica de distância (a mais comum é a euclidiana, mas também podemos utilizar outras, como Manhattan) em torno desse centro inicializado, os centros são atualizados e novos pontos de dados são agrupados. Esse processo acontece até ser estabelecido algum critério de convergência que seja atendido (exemplo: número máximo de iterações, limiar mínimo de mudança nos centroides).

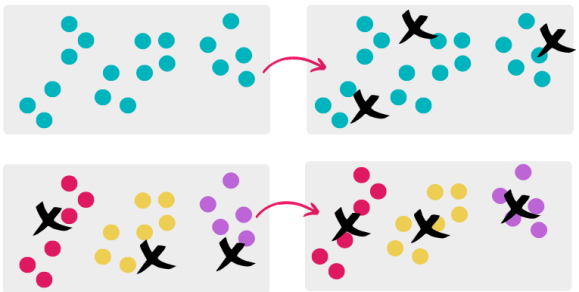


Figura 2 - Algoritmo K-Means.
Fonte: Elaborado pela autora (2024)

O hiperparâmetro do algoritmo K-Means é a definição de **k** (número de grupos a serem criados). Caso esse número não seja orientado por orientações práticas ou gerenciais pela aplicação da técnica (por exemplo, quero encontrar grupos de até 3 personas diferentes), podem surgir dúvidas de como eu posso identificar qual é o melhor número para k.

Aplicação do K-means no Python:

```
from sklearn.cluster import KMeans

kmeans = KMeans(n_clusters=5, random_state=7)

kmeans.fit(dados)
```

Podemos utilizar a técnica **“Elbow”** (método cotovelo ou joelho), onde temos o auxílio da soma dos erros quadráticos (SSE) em relação ao centroide para encontrar o “joelho” da curva de otimização. Com essa técnica, podemos identificar grupos que explicam a maioria da variância nos dados:

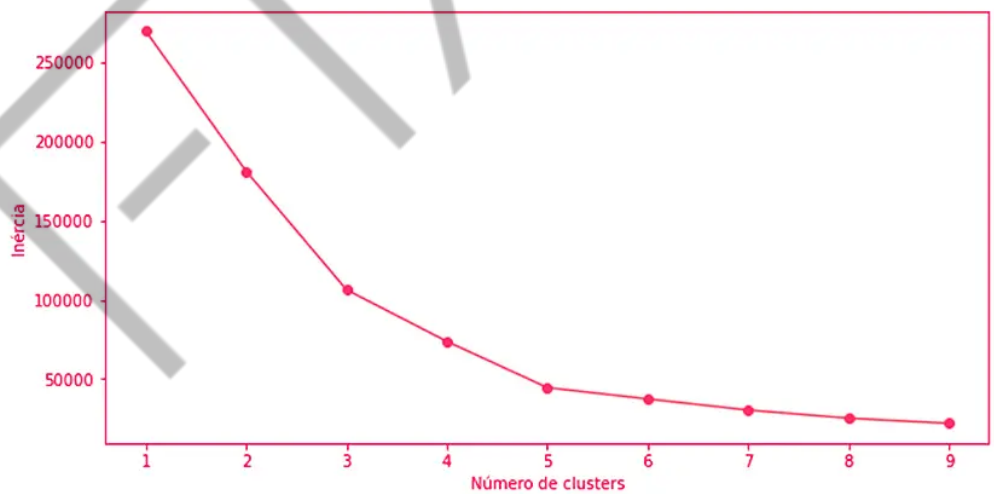


Figura 3 - Método Elbow.
Fonte: Elaborado pela autora (2024)

```
# Armazena o SSE (soma dos erros quadraticos) para cada quantidade de k
sse = []

# Roda o K-means para cada k fornecido
for i in k:
    kmeans = KMeans(n_clusters=i, random_state=0)
    kmeans.fit(dados[['Annual Income (k$)','Spending Score (1-100)']])
    sse.append(kmeans.inertia_)  #calculo do erro do k-mens (mudar o centroide dos dados)

plt.rcParams['figure.figsize'] = (10, 5)
# Plota o gráfico com a soma dos erros quadraticos
plt.plot(k, sse, '-o')
plt.xlabel(r'Número de clusters')
plt.ylabel('Inércia')
plt.show()
```

Métodos de avaliação de clusters

Em modelos supervisionados, conseguimos validar os dados utilizando uma base de teste (dados nunca visto antes) para medir a assertividade do quanto nosso modelo está acertando e errando. Mas, como podemos medir os modelos não supervisionados se não temos essa base de teste?

Em modelos não supervisionados, não temos a separação dos dados em treino e teste, isso porque nos modelos supervisionados queremos buscar uma relação entre os dados existentes e encontrar uma similaridade que possa distinguir grupos.

Dentro das métricas de desempenho de um cluster, podemos considerar dois tipos de avaliação: **interna e externa.**

Avaliação interna: Avaliamos o quão bom meu agrupamento é “por si só”.

Avaliação externa: Quão semelhante foram duas técnicas de agrupamento?

Tipo externa: Na avaliação do tipo externa, normalmente executamos dois ou mais modelos de agrupamentos para avaliar o **quão parecido estão os meus modelos**. E por que eu preciso validar se meus modelos são parecidos? Vou te apresentar um exemplo:

Vamos supor que eu executei um modelo de K-Means, e é um modelo do tipo DBSCAN para validação do tipo externo dos meus grupos de dados. Você não sabe o que seria o modelo DBSCAN? Bem, vou te dar uma introdução bem rápida apenas para que se situe!

O modelo de classificação DBSCAN (Density-Based Spatial Clustering and Application with Nose) é um algoritmo que agrupa os dados **com base em densidade (alta concentração de dados).** Muito bom para tirar ruídos (outliers). O agrupamento dos dados é calculado com base nos core (quantidade de pontos mínimos que seja igual ou maior à definição do MinPts), border (ponto de fronteira dos dados) e noise (ruído). O algoritmo DBSCAN normalmente funciona melhor em dados que possuem uma alta concentração de pontos concentrados, assim como podemos visualizar na Figura 4 – “Método Elbow”.

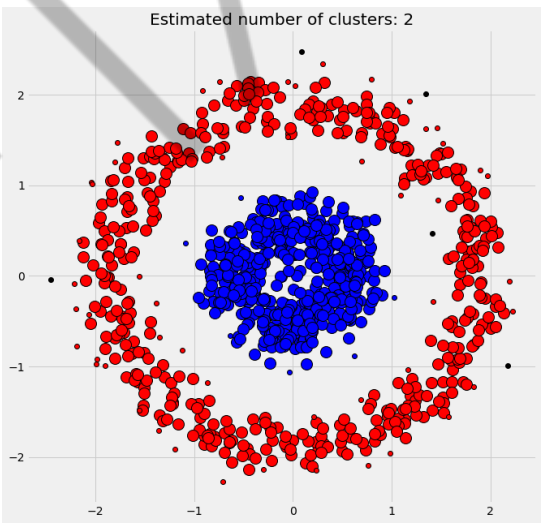


Figura 4 - Método Elbow.
Fonte: Medium (2023)

Pelo DBSCAN ter esse agrupamento por densidade, o modelo possui uma certa dificuldade em encontrar o raio de vizinhança ao tentar agrupar dados com distância média muito distinta (clusters mais densos que outros). Por isso, analisar a

forma dos dados antes de escolher o tipo de algoritmo de clusterização pode ser um bom insight para tomar a decisão da escolha do modelo. Veja uma comparação entre o modelo DBSCAN e K-Means:

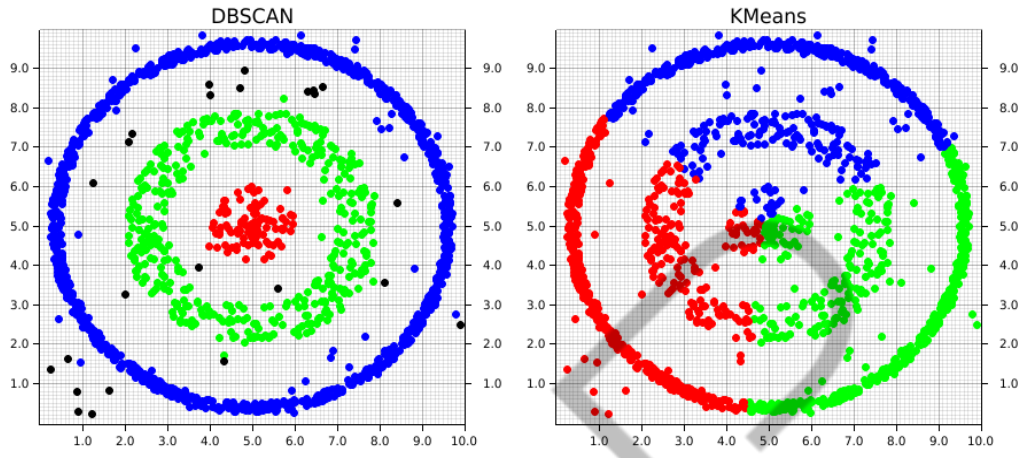


Figura 5 – DBSCAN e K-Means
Fonte: Rust Machine Learning Book (2023)

Bem, agora acredito que você saiba as características do DBSCAN! Vamos dar continuidade no exemplo.

Se eu aplicar um modelo de K-Means e comparar com o modelo de DBSCAN, sob uma base de segmentação de clientes em shoppings, eu poderia ter o seguinte resultado:

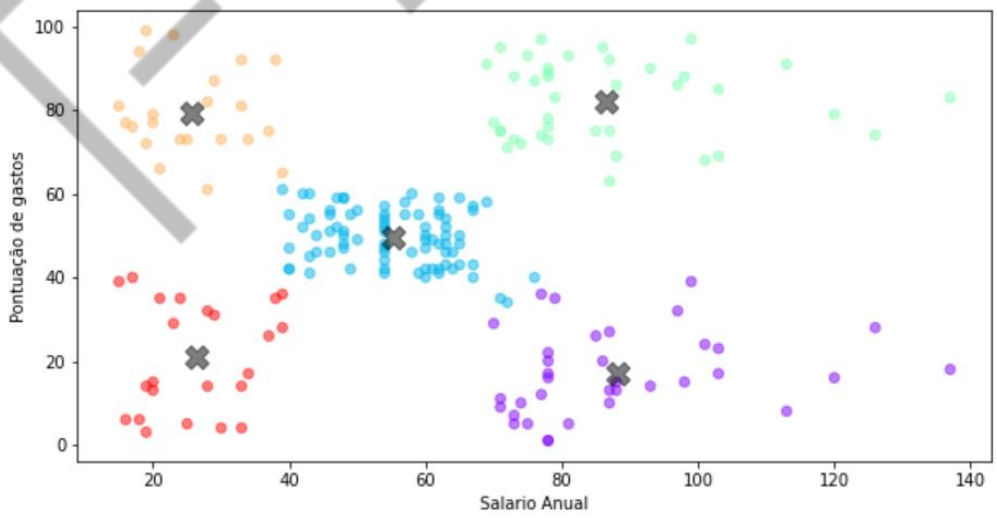


Figura 6 - K-means.
Fonte: Elaborado pela autora (2024)

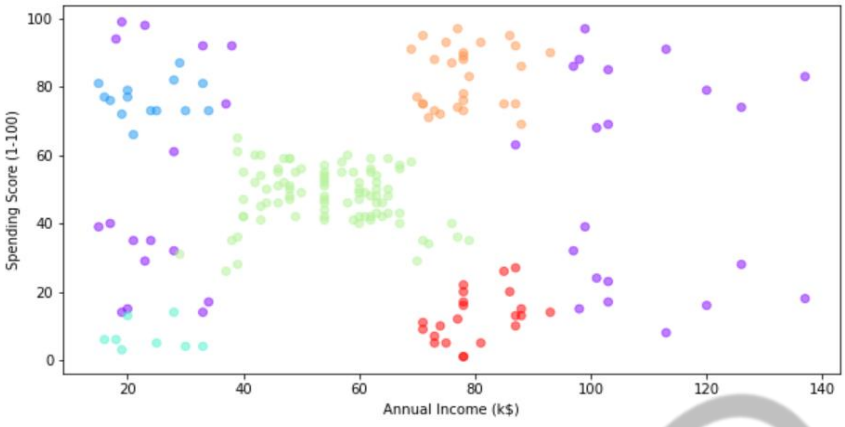


Figura 7 - DBSCAN.
Fonte: Elaborado pela autora (2024)

Comparando as duas técnicas, perceba que para o K-Means os dados foram separados entre cinco grupos (k=5), de acordo com a proximidade dos dados (distância euclidiana). No modelo do DBSCAN, esses pontos em roxo são considerados “ruídos” nos dados, ou seja, outliers. Nesse caso, o modelo identificou que esses dados não são tão densos como os demais, deixando-os de fora do agrupamento. Qual modelo agrupou melhor os dados?

Veja que, pela nossa percepção, o modelo K-Means se comportou melhor em questão de agrupamento nesse cenário. Mas como eu posso comprovar de forma estatística?

Aí que entram as métricas de validação externa! Como, por exemplo, o **adjuestd rand index,** que compara o desempenho quando são fornecidos datasets (bases de dados específicas) com labels (grupos) gerados de forma aleatória. Quando esses labels (grupos) estão muito diferentes, o valor se aproxima de 0, o que sugere um resultado negativo, ou seja, clusters não próximos. Quando temos grupos idênticos o resultado é igual a 1.

Veja o resultado do nosso exemplo no Python:

Comparação entre K-Means e DBSCAN:

```
[ ]  adjusted_rand_score(kmeans_labels,dbscan_labels)

     0.7151933782093212
```

Figura 8 - Adjusted Rand Score.
Fonte: Elaborado pela autora (2024)

Perceba que os algoritmos K-Means e DBSCAN possuem uma similaridade de 0.71, ou seja, estamos longe de 1. Isso significa que os dois modelos são distintos.

Um **outro tipo de validação que também podemos comentar é a interna**. Ainda utilizando nosso exemplo acima, podemos aplicar a métrica de Silhouette. Essa métrica mede o **formato do cluster obtido** (avalia a distância entre os centros dos clusters, nesse caso, queremos maximizar as distâncias). Valores próximos a -1, significa clusters ruins, próximo a 1, clusters bem separados.

KMEANS:

```
[ ]  silhouette_score(dados[['Annual Income (k$)','Spending Score (1-100)']],kmeans_labels)

     0.553931997444648
```

DBSCAN:

```
[ ]  silhouette_score(dados[['Annual Income (k$)','Spending Score (1-100)']],dbscan_labels)

     0.36328873885121793
```

Figura 9 - Silhouette Score: K-means x DBSCAN.
Fonte: Elaborado pela autora (2024)

Analisando a validação com os dois algoritmos, podemos concluir que o K-Means separa melhor os dados do que o DBSCAN.

Desafios do aprendizado de máquina

Você está aprendendo muitas técnicas e modelos de aprendizado de máquina que podem ajudar muito a solucionar problemas da vida real, porém você deve compreender que Machine Learning não é mágica, e sim uma ferramenta. Você, como cientista de dados, pode encontrar alguns desafios ao longo da sua jornada. Vamos conversar um pouquinho sobre eles.

Podemos ter alguns impedimentos que podem ocasionar resultados ruins em nossos algoritmos, sendo eles: **algoritmos ruins ou dados ruins**.

Se tratando de **dados ruins**, podemos ter alguns fatores:

- **Quantidade insuficiente de dados:** com uma amostra de dados pequena, existirá um "ruído de amostragem", e se houver uma amostra muito grandes com dados não representativos, o método de amostragem também pode ser falho (**viés de amostragem**).
- **Dados de treinamento não representativos**: o objetivo de um bom modelo de aprendizado de máquina é generalizar bem a partir dos dados de treinamento, sendo assim, é importante buscar uma base de dados representativa. Será que a sua base de dados consegue generalizar?
- **Dados de baixa qualidade:** aqui é preciso se dedicar à limpeza dos dados, uma base não consistente pode impactar na detecção de padrões.
- **Características irrelevantes:** entra lixo, sai lixo. Atenção aos dados que entram no seu modelo! A dica aqui é a dedicação na etapa de Feature Engineering, ou técnicas de redução de dimensionalidade.

Agora, vamos falar um pouquinho sobre **algoritmos ruins**:

- **Sobreajuste nos dados (overfitting):** quando o seu modelo funciona muito bem com os dados de treinamento, mas não generaliza bem novos dados de entrada. Isso pode acontecer quando o modelo é muito complexo em relação ao ruído e quantidade. Como solução, podemos pensar em:
  - Simplificar o modelo.
  - Coletar mais dados.
  - Reduzir o ruído (exemplo, remover outliers).
  - Regularização: Chamamos de regularização quando restringimos um modelo para simplificar e reduzir o risco de reajuste dos dados. A regularização pode ajudar a generalizar melhor o modelo em novos exemplos de dados.
- **Subajuste dos dados (underfitting):** nesse caso, seu modelo ficou muito simples ao ponto de não aprender corretamente os dados. Algumas soluções:
  - Selecionar um modelo mais poderoso.
  - Feature engineering.

- Reduzir as regularizações.

- **Base de treino, teste e validação cruzada:** Para medir o erro da generalização, utilize a técnica de treinar vários hiperparâmetros utilizando o conjunto de treinamento. Selecione os hiperparâmetros que melhor se adaptam no conjunto de validação.

O QUE VOCÊ VIU NESTA AULA?

Modelos não supervisionados; K-Means e DBSCAN; métricas de validação de cluster; desafios do aprendizado de máquina.

Daqui para a frente, é importante que você replique os conhecimentos adquiridos para fortalecer mais suas bases e conhecimentos.

IMPORTANTE: não esqueça de praticar com o desafio da disciplina, para que assim você possa aprimorar os seus conhecimentos!

Você não está só nesta jornada! Te esperamos no Discord e nas *lives* com os nossos especialistas, onde você poderá tirar dúvidas, compartilhar conhecimentos e estabelecer conexões!

REFERÊNCIAS

BRUCE, Andrew. Estatística Prática para Cientistas de Dados, 1° Edição. [s.l.]: O'Reilly Media, Inc., 2019.

DOCUMENTAÇÃO SCIKIT-LEARN. **Disponível em: <https://scikit-learn.org/stable/>.** Acesso em: 11 abr 2023.

GÉRON, Aurélien. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition. [s.l.]: O'Reilly Media, Inc., 2019.

PALAVRAS-CHAVE

Palavras-Chave: K-Means. Dbscan. Modelo Não Supervisionado De Classificação. Desafios De Aprendizado De Máquina.