Fichamento dos Modelos

Regressão Logística

Algoritmo/Metodologia base:

Função Logit. Razão logarítmica das probabilidades de um evento acontecer.

Principal mudança/evolução em relação ao algoritmo base:

Introduz a função logit na saída linear da regressão para obter uma probabilidade.

Pontos Positivos:

- Simples e eficiente
- Produz probabilidade para interpretação
- Tem fácil interpretabilidade

Pontos Negativos:

- Assume independência entre as variáveis
- Sensível a outliers

Penalty	C	Max_Iter
Controla o peso das penalidades para evitar overfitting	Constante que multiplica o termo da regularização	Número máximo de iterações possíveis

Solver	Class_Weight	Tol
Algoritmo usado para a otimização do modelo	Peso para as classes, buscando seu balanceamento	Tolerância para o critério de parada. Também chamado de taxa de aprendizagem

VIÉS X VARIÂNCIA

Alto viés pode ocorrer quando assumimos erroneamente que a relação entre as variáveis independentes e a variável dependente é simples, enquanto na realidade é mais complexa. Alta variância pode ser excessivamente flexível, levando a overfitting. A técnica de regularização, como a regularização L1 (Lasso) ou L2 (Ridge), pode ser usada para controlar a complexidade do modelo e é possível ajustar os hiperparametros “penalty” e “C” para atingir o melhor ponto de viés e variância.

Regressão Linear

Algoritmo/Metodologia base:

Se baseia no conceito de encontrar a melhor linha ou hiperplano que se ajusta aos dados.

Principal mudança/evolução em relação ao algoritmo base:

Pode ser estendida para uma regressão linear múltipla.

Pontos Positivos:

- Simples e fácil de interpretar
- Rápida e eficiente para grandes conjuntos de dados
- Facilmente escalável para uma regressão múltipla

Pontos Negativos:

- Sensível a outliers
- Assume linearidade entre as variáveis dependentes e independentes
- Pode levar ao overfitting caso atribua coeficientes muito grandes para apenas uma variável

N_Jobs

Número de processadores utilizados pelo modelo

VIÉS X VARIÂNCIA

Alto viés pode ocorrer quando assumimos erroneamente que a relação entre as variáveis independentes e a variável dependente é simples, enquanto na realidade é mais complexa. Alta variância pode ser excessivamente flexível, levando a overfitting. A técnica de regularização, como a regularização L1 (Lasso) ou L2 (Ridge), pode ser usada para controlar a complexidade do modelo.

Ridge

Algoritmo/Metodologia base:

Extensão da regressão linear.

Principal mudança/evolução em relação ao algoritmo base:

Adiciona o termo de regularização L2 na regressão para evitar valores de coeficientes muito altos.

Pontos Positivos:

- Reduz a variância do modelo, diminuindo a probabilidade de overfitting
- Lida bem com multicolinearidade

Pontos Negativos:

- Reduz os coeficientes para o mais próximo de zero
- Menos eficaz se todas as variáveis independentes forem relevantes

Alpha	Tol	Max_Iter
Constante que multiplica o termo L2, controlando a força da regularização	Tolerância para o critério de parada. Também chamado de taxa de aprendizagem	Número máximo de iterações possíveis

VIÉS X VARIÂNCIA

Introduz um leve viés nos coeficientes dos parâmetros, pois o termo de regularização L2 penaliza esses coeficientes, forçando-os a serem menores. A regularização do modelo Ridge tem como principal objetivo reduzir a variância, controlando a amplitude dos coeficientes dos parâmetros. Quando os dados de treinamento têm multicolinearidade ou um número significativo de variáveis independentes, a estimativa dos coeficientes pode ser instável. A regularização Ridge ajuda a estabilizar essas estimativas, reduzindo a sensibilidade do modelo a pequenas variações nos dados de treinamento. Além disso, é possível ajustar o hiperparâmetro “alpha” para um melhor ajuste entre viés e variância.

Lasso

Algoritmo/Metodologia base:

Extensão da regressão linear.

Principal mudança/evolução em relação ao algoritmo base:

Adiciona o termo de regularização L1 na regressão, podendo levar a coeficientes com valores esparsos.

Pontos Positivos:

- Realiza seleção de variáveis ao forçar coeficientes a serem zero
- Lida bem com muitas variáveis irrelevantes

Pontos Negativos:

- Não performa bem com multicolinearidade
- Coeficientes enviesados

Alpha	Tol	Max_Iter
Constante que multiplica o termo L2, controlando a força da regularização	Tolerância para o critério de parada. Também chamado de taxa de aprendizagem	Número máximo de iterações possíveis

VIÉS X VARIÂNCIA

O modelo Lasso introduz viés ao aplicar a regularização L1, que penaliza os coeficientes dos parâmetros, forçando alguns deles a serem exatamente iguais a zero. Ao forçar alguns coeficientes a zero, o Lasso simplifica o modelo, tornando-o menos suscetível a overfitting. Assim como no modelo Ridge, o hiperparâmetro “alpha” é utilizado para atingir o melhor balanço entre viés e variância.

Elastic Net

Algoritmo/Metodologia base:

É uma combinação da ridge e lasso.

Principal mudança/evolução em relação ao algoritmo base:

Adiciona os termos de regularização L1 e L2 na regressão.

Pontos Positivos:

- Combina a capacidade de selecionar variáveis da lasso com a capacidade de lidar com multicolinearidade da ridge

Pontos Negativos:

- Pode requerer um maior custo computacional
- Ajustar os hiperparâmetros pode ser complicado

Alpha Constante que multiplica o termo L2, controlando a força da regularização	**Max_Iter** Número máximo de iterações possíveis
L1_Ratio Valor para balancear os termos de regularização. Caso 0, implica ridge e caso 1, implica lasso	**Tol** Tolerância para o critério de parada. Também chamado de taxa de aprendizagem

VIÉS X VARIÂNCIA

O Elastic Net, ao incorporar tanto a regularização L1 quanto a L2, introduz um viés que combina as características de ambas as técnicas. Assim como o Lasso, o Elastic Net pode forçar alguns coeficientes a serem exatamente zero, realizando seleção de variáveis. Ao mesmo tempo, a regularização L2 ajuda a estabilizar as estimativas dos coeficientes, reduzindo a sensibilidade a multicolinearidade. Quando alpha = 0, o Elastic Net é equivalente à regressão Ridge, e quando alpha = 1, é equivalente ao Lasso.

Árvore de Decisão

Algoritmo/Metodologia base:

Divisão recursiva binária, onde o conjunto de dados é dividido em subconjuntos a partir das características mais significativas.

Principal mudança/evolução em relação ao algoritmo base:

Pontos Positivos:

- Fácil de entender e visualizar
- Pode ser usada tanto para regressão como para classificação
- Não necessita de uma grande preparação dos dados, como normalização ou enconding

Pontos Negativos:

- Pode resultar em overfitting caso a árvore seja muito funda, não conseguindo generalizar
- Sensível a pequenas variações nos dados de entrada, podendo resultar em árvores totalmente diferentes

Max_Depth	Min_Sample_Split
Número máximo da profundidade da árvore	Número mínimo de amostras necessárias para a continuação da árvore
Criterion	**Min_Sample_Leaf**
Critério usado para medir a qualidade da divisão	Número mínimo de amostras para formar uma folha

VIÉS X VARIÂNCIA

Em árvores de decisão, alto viés pode ocorrer quando a árvore é muito rasa e simples. Isso significa que a árvore não é capaz de capturar a complexidade subjacente nos dados. Em relação à variância, uma árvore de decisão com alta variância pode ocorrer quando a árvore é muito profunda e se ajusta demais aos detalhes dos dados de treinamento. A regularização pode ser alcançada ajustando hiperparâmetros como a profundidade máxima da árvore, busando atingir a profundidade ideal.

SVM

Algoritmo/Metodologia base:

Encontrar o melhor hiperplano que separa as classes.

Principal mudança/evolução em relação ao algoritmo base:

Pontos Positivos:

- Eficaz em espaços de alta dimensão
- Eficaz em casos onde o número de dimensões é maior que o de amostras
- É versátil, podendo lidar com conjunto de dados não lineares a partir da escolha do kernel

Pontos Negativos:

- Sensível à escala dos dados
- Alto custo computacional, principalmente em grandes conjuntos de dados
- Escolha do kernel e ajuste dos hiperparâmetros é crucial para o desempenho e para evitar overfitting

Kernel	Tol	C
Pode ser linear, polinomial, radial etc. A escolha é crucial para o desempenho do modelo	Tolerância para o critério de parada. Também chamado de taxa de aprendizagem	Constante que multiplica o termo da regularização

Gamma	Degree	Max_Iter
Coeficiente dos kernels não lineares	Grau do kernel polinomial	Número máximo de iterações possíveis

VIÉS X VARIÂNCIA

Em casos onde os dados são linearmente separáveis, o SVM pode ter um baixo viés, pois é capaz de se ajustar bem aos dados de treinamento. Caso a separação não seja linear, o SVM pode ter um viés mais alto, não sendo flexível o suficiente para capturar padrões complexos. A variância no SVM pode ser influenciada pela escolha do tipo de kernel e pelos hiperparâmetros associados. O trade-off entre viés e variância no SVM é frequentemente controlado pelo ajuste do hiperpaparâmetro “C”.

Naive Bayes

Algoritmo/Metodologia base:

Baseado no Teorema de Bayes, que descreve a probabilidade condicional de um evento, dado outro.

Principal mudança/evolução em relação ao algoritmo base:

Assume uma independência condicional entre as variáveis, simplificando o cálculo da probabilidade.

Pontos Positivos:

- Simples e eficiente
- Lida bem com dados esparsos
- É eficaz em conjunto de dados com muitas variáveis

Pontos Negativos:

- Assume a independência condicional, o que não é algo realista
- Sensível a outliers

Priors	Var_Smoothing
Probabilidades prévias das classes. Se especificado, os anteriores não são ajustados de acordo com os dados.	Parte da maior variação de todos os recursos que é adicionada às variações para estabilidade do cálculo.

VIÉS X VARIÂNCIA

O Naive Bayes, devido à sua natureza simplificada, geralmente apresenta um trade-off viés-variância diferente de modelos mais complexos. Ele tende a ter um viés mais alto devido à suposição de independência condicional, mas uma variância menor devido à simplicidade. A escolha entre modelos mais complexos e o Naive Bayes dependerá da natureza do problema e dos dados.

KNN

Algoritmo/Metodologia base:

A partir da vizinhança, onde um dado é classificado com base nas classes dos dados mais próximos

Principal mudança/evolução em relação ao algoritmo base:

Pontos Positivos:

- Simples de implementar e entender
- Pode lidar com conjunto de dados não lineares e complexos
- Bem eficaz em conjunto de dados pequenos ou médios

Pontos Negativos:

- Sensível à escala das variáveis, sendo necessária uma normalização dos dados
- A escolha de K é crucial, impactando diretamente o desempenho do modelo

N_Neighbors (K)	Metric	P
Quantidade de vizinhos que serão analisados para classificar o dado	Define a distância que será utilizada para realizar os cálculos	Valor do expoente utilizado por algumas distâncias

Weights	Algorithm
Determina como os vizinhos são ponderados ao fazer uma previsão para um dado	Determina o algoritmo que será utilizado na pesquisa dos vizinhos mais próximos

VIÉS X VARIÂNCIA

O KNN possui um viés elevado caso os dados possuam muitas dimensões. Já a variância pode ser mais alta quando o valor de K é pequeno. Um valor pequeno de K faz com que o modelo seja mais sensível a pequenas variações nos dados de treinamento, levando a uma maior variância. Para ajustar o viés e variância desse modelo, geralmente se ajusta o valor de K.

Bagging

Algoritmo/Metodologia base:

Método de ensemble que combina múltiplos modelos que são treinados com subconjuntos aleatórios dos dados de treino.

Principal mudança/evolução em relação ao algoritmo base:

Pontos Positivos:

- Melhora a generalização
- Aumenta a robustez e estabilidade

Pontos Negativos:

- Pode aumentar o tempo de treinamento
- Introduz uma perda na interpretabilidade

N_Estimators	Max_Samples	Max_Features
Números de estimadores que serão utilizados	Número máximo de amostras para serem usadas no treinamento de cada estimador	Número máximo de variáveis que serão usadas para treinar cada estimador

VIÉS X VARIÂNCIA

Pode reduzir o viés em comparação com modelos individuais, com cada modelo no ensemble aprendendo diferentes aspectos dos dados. Além disso, pode reduzir a variância ao treinar modelos em diferentes subconjuntos de dados e combinar suas previsões, suavizando as flutuações nos dados de treinamento, resultando em uma melhor generalização para novos dados. A escolha dos modelos base, a diversidade entre eles e o número de modelos no ensemble são aspectos importantes a serem considerados para otimizar o trade-off entre viés e variância.

Random Forest

Algoritmo/Metodologia base:

Extensão do Bagging.

Principal mudança/evolução em relação ao algoritmo base:

Aleatoriedade adicional na construção das árvores, treinando cada árvore em uma amostra aleatória de variáveis.

Pontos Positivos:

- Reduz overfitting e aumenta a generalização em relação às árvores individuais
- Não necessita de uma grande preparação dos dados, como normalização ou enconding

Pontos Negativos:

- Grande custo computacional
- Menos interpretável do que uma árvore de decisão individual

Max_Depth	Min_Sample_Split	Min_Sample_Leaf
Número máximo da profundidade da árvore	Número mínimo de amostras necessárias para a continuação da árvore	Número mínimo de amostras para formar uma folha

Criterion	N_Estimators	Max_Features
Critério usado para medir a qualidade da divisão	Número de árvores na floresta	Número máximo de variáveis a serem consideradas para dividir um nó

VIÉS X VARIÂNCIA

O Random Forest, ao usar múltiplas árvores de decisão, pode reduzir o viés em comparação com uma única árvore de decisão. A seleção aleatória de características para cada árvore torna as árvores mais descorrelacionadas, e a média ou a votação entre essas árvores ajuda a suavizar as previsões globais. Isso resulta em um modelo mais estável e menos suscetível a overfitting. Para alcançar o melhor equilíbrio, pode-se ajustar a profundidade das árvores e o número de árvores.

Stacking

Algoritmo/Metodologia base:

Método de ensemble que combina a saída de vários modelos de base para formar um modelo meta.

Principal mudança/evolução em relação ao algoritmo base:

Incorpora a ideia de treinar um modelo extra (meta).

Pontos Positivos:

- Aumenta a robustez e generalização
- Diminui o risco de overfitting por conta da diversidade de modelos utilizados
- Flexibilidade para usar uma variedade de modelos de base

Pontos Negativos:

- Maior custo computacional
- Ajuste cuidadoso de hiperparâmetros para bom desempenho

Estimators	Final_Estimator	CV
Lista dos estimadores base que serão empilhados juntos	Um regressor/classificador que combinará os estimadores base	Determina a estratégia de divisão da validação cruzada

VIÉS X VARIÂNCIA

O Stacking busca encontrar um equilíbrio entre viés e variância, utilizando modelos base diversificados. A escolha dos modelos base e do meta-modelo é crucial para otimizar o desempenho geral do ensemble. Modelos base que têm desempenho bom em diferentes aspectos dos dados contribuem para a diversidade e, consequentemente, para a redução da variância.

Gradient Boosting

Algoritmo/Metodologia base:

Técnica de ensemble que constrói uma sequência de modelos de aprendizado de máquina, onde cada novo modelo corrige os erros dos modelos anteriores.

Principal mudança/evolução em relação ao algoritmo base:

Treina os modelos sequencialmente, focando nos erros dos modelos anteriores.

Pontos Positivos:

- Eficaz para melhorar o resultado de modelos fracos
- Pode lidar com dados numéricos e categóricos

Pontos Negativos:

- Sensíveis a outliers e mais fáceis de levar ao overfitting
- Difícil interpretação do modelo final

Learning_Rate	N_Estimators	Max_Depth
Controla a contribuição de cada modelo à correção dos erros	Define o número de estágios de reforço a serem executados	Número máximo da profundidade de cada estimador

Criterion	Min_Sample_Split	Min_Sample_Leaf
Critério usado para medir a qualidade da divisão	Número mínimo de amostras necessárias para a continuação da árvore	Número mínimo de amostras para formar uma folha

VIÉS X VARIÂNCIA

Inicialmente, o Gradient Boosting começa com um modelo simples, geralmente uma árvore de decisão pequena. Esse modelo simples pode ter um viés mais alto, pois pode não capturar completamente a complexidade dos dados. Além disso, cada modelo complementa os pontos fracos dos modelos anteriores. A estratégia de ajustar os resíduos visa construir um modelo mais robusto e menos suscetível a overfitting. A taxa de aprendizado, o número de árvores e a profundidade das árvores são hiperparâmetros que ajudam a equilibrar o viés e variância.

XGBoost

Algoritmo/Metodologia base:

Implementação do Gradient Boosting, eficiente e escalável.

Principal mudança/evolução em relação ao algoritmo base:

Incorpora várias otimizações, como regularização, manipulação eficiente de dados ausentes e técnicas de pruning.

Pontos Positivos:

- Possui regularização integrada
- É flexível, pode ser usado em classificação, regressão e ranking
- Rápido e eficiente

Pontos Negativos:

- Utiliza muita memória para ser executado
- É um modelo bem complexo, sendo difícil realizar uma boa otimização

ETA (learning_rate)	Gamma	Max_Depth
Controla a contribuição de cada modelo à correção dos erros	Redução mínima de perdas necessária para fazer uma partição adicional em um nó folha da árvore.	Número máximo da profundidade de cada estimador

N_Estimators	Subsample	Colsample_Bytree
Define o número de árvores do modelo	Proporção de subamostra das instâncias de treinamento	Proporção de subamostra de colunas ao construir cada árvore

VIÉS X VARIÂNCIA

O XGBoost busca encontrar um equilíbrio entre viés e variância assim como o Gradient Boosting, mas com algumas otimizações adicionais. Hiperparâmetros como a taxa de aprendizado, profundidade máxima da árvore, ajudam a a encontrar esse equilíbrio. Junto a isso, o XGBoost incorpora regularizações e técnicas de pruning, que podem ter suas forças ajustadas.

LightGBM

Algoritmo/Metodologia base:

Implementação do Gradient Boosting, projetada para lidar com grandes conjuntos de dados e alta dimensionalidade.

Principal mudança/evolução em relação ao algoritmo base:

Adiciona seleção automática de recursos, resultando em uma aceleração do treinamento e melhor desempenho preditivo.

Pontos Positivos:

- Extremamente eficiente em termos de tempo e recursos, especialmente em grandes conjuntos de dados
- Lida bem com dados esparsos e de alta dimensão
- Pode lidar com tarefas de classificação, regressão e ranking

Pontos Negativos:

- Pode ser sensível a overfitting, especialmente com muitas árvores e parâmetros complexos
- É um modelo bem complexo, sendo difícil realizar uma boa otimização

N_Estimators	Max_Depth	Learning_Rate
Define o número de árvores do modelo	Número máximo da profundidade de cada estimador	Controla a contribuição de cada modelo à correção dos erros

Num_Leaves	Boosting_Type
Número máximo de folhas em uma árvore	Qual algoritmo de boosting que será utilizado

VIÉS X VARIÂNCIA

O LGBM busca um equilíbrio entre viés e variância, sem sacrificar a eficiência computacional. Hiperparâmetros como a taxa de aprendizado, a profundidade máxima da árvore, a força de regularização, e o número de árvores são cruciais para ajustar esse equilíbrio.

AdaBoost

Algoritmo/Metodologia base:

Algoritmo de ensemble que se baseia em treinar modelos fracos sequencialmente, com ênfase nas instâncias classificadas incorretamente pelos modelos anteriores.

Principal mudança/evolução em relação ao algoritmo base:

Atribui pesos às instâncias durante o treinamento, colocando mais foco no treino dos erros dos modelos anteriores.

Pontos Positivos:

- Eficaz em melhorar o desempenho, mesmo com modelos fracos como base
- Pode ser aplicado a uma variedade de algoritmos de aprendizado de máquina
- Lida bem com dados numéricos e categóricos

Pontos Negativos:

- Podem ser afetados por overfitting
- Sensível a outliers

N_Estimators	Learning_Rate
Define o número de árvores do modelo	Controla a contribuição de cada modelo à correção dos erros

VIÉS X VARIÂNCIA

O AdaBoost busca encontrar um equilíbrio entre viés e variância, adaptando-se à dificuldade dos exemplos no conjunto de dados. Ele ajusta dinamicamente os pesos das instâncias, dando mais importância às instâncias que são classificadas erroneamente. O ajuste do número de estimadores e da taxa de aprendizado ajudam a encontrar o equilíbrio entre viés e variância.

KMeans

Algoritmo/Metodologia base:

Algoritmo de clustering. Divide os dados em k clusters.

Principal mudança/evolução em relação ao algoritmo base:

Cada cluster dividido pelo algoritmo é representado por seu centróide, sendo a média das instâncias pertencentes ao cluster.

Pontos Positivos:

- Simples e eficiente
- Escalável para grandes conjuntos de dados
- Bem adequado para dados onde os clusters têm formas esféricas e aproximadamente iguais em tamanho

Pontos Negativos:

- Sensível à inicialização dos centróides, pode convergir para diferentes soluções dependendo do ponto de partida
- Assume que os clusters são esféricos e isotrópicos, que nem sempre é o caso

N_Clusters	N_Init
Define o número de clusters que o algoritmo deve formar	Número de vezes que o algoritmo é executado com diferentes centróides
Algorithm	**Max_Iter**
Determina qual algoritmo será utilizado	Número máximo de iterações

VIÉS X VARIÂNCIA

O KMeans assume que os clusters têm uma forma esférica e que todos os pontos dentro de um cluster são igualmente relevantes para determinar seu centro. Essa suposição é uma simplificação significativa e pode levar a um viés quando os clusters têm formas mais complexas ou quando as densidades dos clusters não são uniformes. O KMeans pode ser sensível à inicialização dos centróides e à escolha do número de clusters K. Diferentes inicializações podem levar a soluções diferentes, e a variância do KMeans pode aumentar se os dados não estiverem naturalmente organizados em clusters esféricos.

Agglomerative Clustering

Algoritmo/Metodologia base:

Algorítimo de clustering hierárquico, formando uma estrutura de árvore.

Principal mudança/evolução em relação ao algoritmo base:

Considera cada instância como um cluster separado e, iterativamente, mescla os clusters mais próximos até que todos os dados estejam em um único cluster.

Pontos Positivos:

- Produz uma representação hierárquica dos clusters
- Não requer a especificação do número de clusters a priori
- Pode lidar com diferentes formas e tamanhos de clusters

Pontos Negativos:

- Pode ser computacionalmente intensivo para grandes conjuntos de dados
- Sensível à métrica de distância utilizada.

N_Clusters	Linkage
Define o número de clusters que o algoritmo deve formar	Define a métrica utilizada para medir a distância entre clusters durante a fase de mesclagem
Metric	**Distance_Threshold**
Métrica usada para calcular a ligação	O limite de distância de ligação ao qual os clusters não serão mesclados

VIÉS X VARIÂNCIA

O Agglomerative Clustering pode ser sensível à métrica de distância escolhida e ao critério de ligação (linkage) utilizado para medir a dissimilaridade entre clusters. Diferentes escolhas podem levar a soluções de clustering distintas. Além disso, a variância pode aumentar com o número de pontos de dados, tornando-o menos eficiente para grandes conjuntos de dados. O trade-off no Agglomerative Clustering está muitas vezes relacionado à escolha da métrica de distância e do critério de ligação.

DBScan

Algoritmo/Metodologia base:

Algoritmo de clustering baseado em densidade, projetado para identificar clusters em um espaço onde a densidade dos pontos é variável.

Principal mudança/evolução em relação ao algoritmo base:

Não requer a especificação do número de clusters a priori. Define clusters como regiões densas de pontos separadas por regiões de menor densidade.

Pontos Positivos:

- Pode identificar clusters de diferentes formas e tamanhos
- Lida bem com ruído e outliers, classificando-os como pontos de "ruído"
- Não requer a predefinição do número de clusters

Pontos Negativos:

- Pode ter dificuldade em identificar clusters de densidades semelhantes
- Não lida bem com dados de alta dimensionalidade

Eps	Min_Samples	Algorithm
Define a distância máxima entre dois pontos para serem considerados vizinhos	O número de amostras em uma vizinhança para um ponto ser considerado ponto central	Determina qual algoritmo será utilizado para encontrar vizinhos mais próximos

Metric	P
A métrica usada ao calcular a distância entre instâncias em uma matriz de recursos	Valor do expoente utilizado pela métrica Minkowski

VIÉS X VARIÂNCIA

O DBSCAN não faz muitas suposições sobre a forma ou o tamanho dos clusters, o que o torna menos propenso a viés em comparação com métodos que assumem formas específicas de clusters. A variância ainda pode ocorrer dependendo dos parâmetros escolhidos, como o raio para definir a vizinhança. Um valor muito pequeno de “eps” pode resultar em muitos clusters pequenos, enquanto um valor muito grande pode agrupar pontos distantes em um único cluster.