UNIVERSIDADE FEDERAL DE UBERLÂNDIA

FACULDADE DE COMPUTAÇÃO

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

GULLIT DAMIÃO TEIXEIRA DE CAMPOS

Algoritmos K-NN

Uberlândia, Brasil

2024.

GULLIT DAMIÃO TEIXEIRA DE CAMPOS

Algoritmos K-NN

Segundo Projeto de conclusão da disciplina Inteligência Artificial apresentado ao Departamento da Faculdade de Computação da Universidade Federal de Uberlândia, de Uberlândia, Câmpus Santa Mônica, como parte dos requisitos para obtenção da pontuação referente a trabalhos e projeto da disciplina do curso bacharelado em Ciências de Computação.

Orientador: Prof. Dr. Jefferson Rodrigo de Souza

Uberlândia, Brasil

2024.

O principal objetivo do foi código classificar morangos em duas categorias de qualidade: boa e má. Para isso, o código utiliza o algoritmo de K-Nearest Neighbors (KNN), que é um modelo de aprendizado supervisionado. O KNN classifica os morangos com base nas suas características, como tamanho (Size), peso (Weight), acidez (Acidity) ...

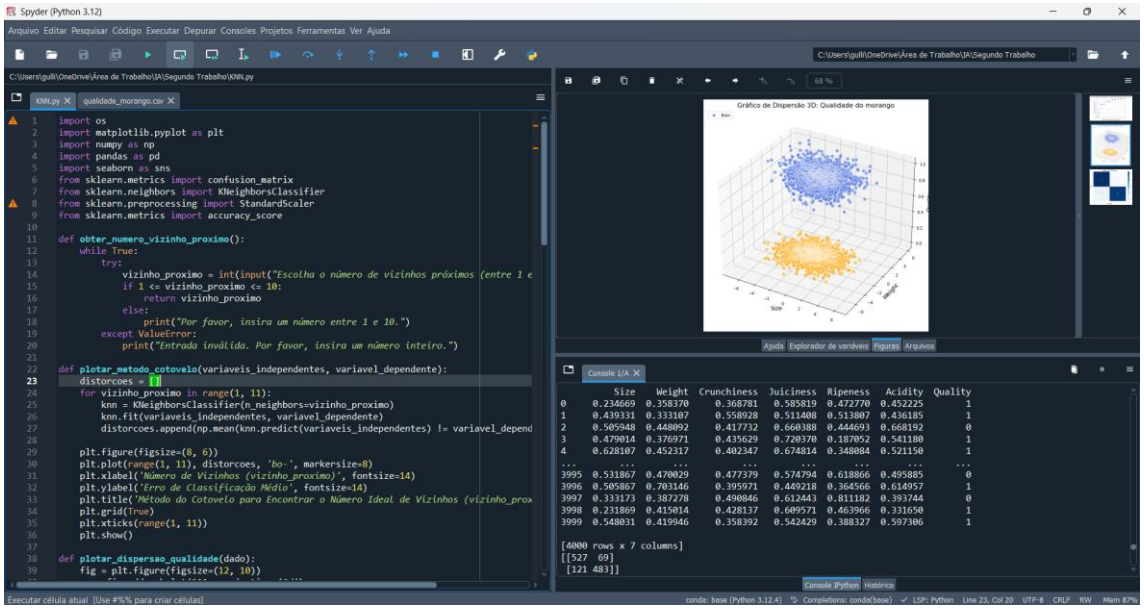
O código foi estruturado de forma a aplicar várias técnicas para garantir uma classificação precisa e eficiente como:

I. Pré-processamento dos dados: Converte os dados em um formato adequado para o KNN.
II. Busca pelo número ideal de vizinhos (k): O número de vizinhos no KNN influencia o desempenho do modelo, e a escolha correta desse valor.
III. Avaliação da importância das características: A cada iteração, verifica-se como cada característica (como peso, tamanho, acidez) contribui para a classificação, ajudando a refinar o modelo

Técnicas Utilizadas no Código

Pré-processamento de Dados a função tratar_dataset serve para preparar os dados antes de serem usados pelo algoritmo KNN. Essa função realiza duas tarefas principais:

I. Conversão de valores qualitativos para numéricos: Os valores de qualidade dos morangos, que são inicialmente representados como "good" e "bad", são convertidos para valores numéricos, 1 para "good" e 0 para "bad". Isso é necessário porque o KNN só funciona com dados numéricos.
II. Remoção de valores nulos: O tratamento de dados nulos é feito para garantir que não haja interrupções ou erros durante o processo de treinamento e classificação. Isso ajuda a evitar problemas de precisão e desempenho do modelo.
    Essas duas tarefas foram realizadas, pois o algoritmo KNN depende de dados “limpos” e numéricos para calcular as distâncias entre as instâncias.



Normalização dos Dados: A normalização foi uma técnica utilizada pensado que importante os dados cujas variáveis possuem escalas diferentes. Por exemplo, um morango pode ter um peso em gramas, enquanto sua acidez é medida em uma escala diferente.

A função de normalização aplicada ao código ajusta as características numéricas, como o peso e o tamanho dos morangos, para que todas as variáveis estejam na mesma escala. Isso evita que características com valores maiores dominem o cálculo da distância, garantindo que todas as variáveis contribuam de maneira equilibrada para a classificação.

Isso é feito por meio de uma transformação que escala os valores para um intervalo comum, geralmente entre 0 e 1.

Classificação com KNN: O K-Nearest Neighbors (KNN) é um algoritmo de classificação baseado em distâncias. Ele classifica um dado novo (neste caso, um morango) com base nas classes das instâncias mais próximas (os k vizinhos mais próximos).

A função testes_knn implementa o algoritmo KNN e, para encontrar a melhor classificação, percorre diferentes valores de "k" (número de vizinhos). O código realiza uma avaliação da taxa de erro médio para determinar qual valor de k resulta na melhor acurácia para a classificação da qualidade do morango.

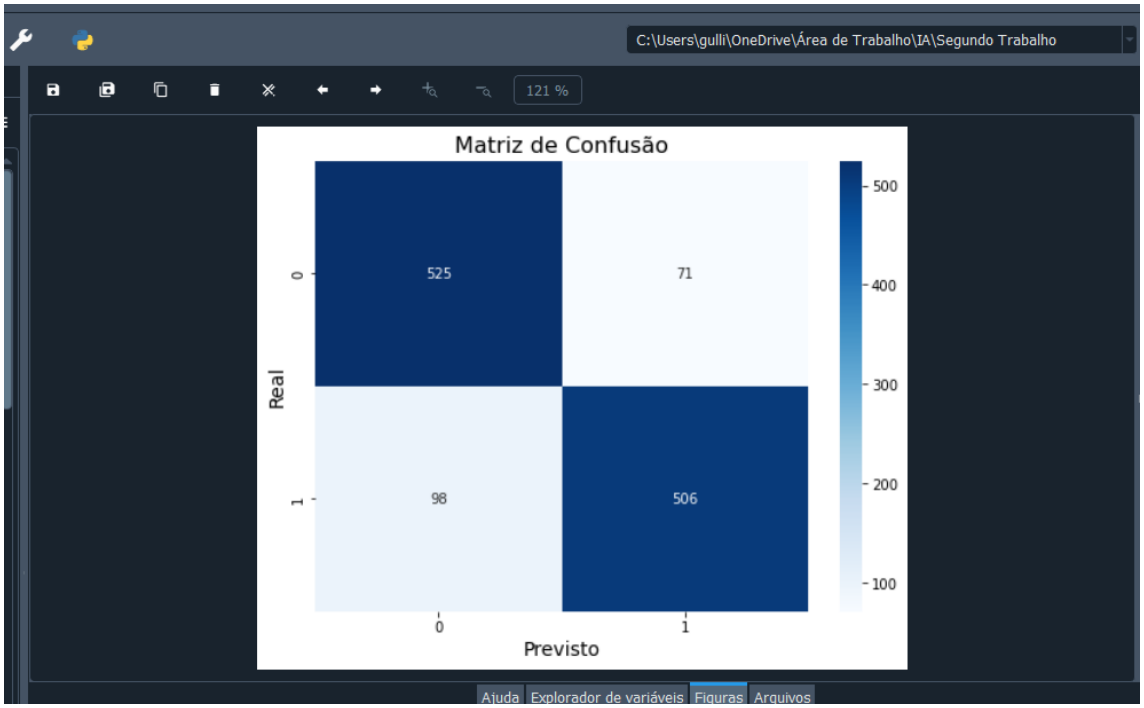
O valor de k é importante porque valores muito pequenos podem ser suscetíveis ao overfitting (ajuste excessivo aos dados de treinamento), enquanto valores muito grandes podem resultar em um modelo que perde a sensibilidade às variabilidades dos dados.

Matriz de Confusão: foi uma ferramenta fundamental para o código pois serve avaliar a eficácia do modelo de classificação. Ela mostra o desempenho do modelo, detalhando os acertos e erros na classificação dos morangos.

A função plotar_matriz_confusao gera a matriz de confusão para mostrar:

- Verdadeiros positivos (VP): Quando o modelo corretamente classifica um morango como de boa qualidade.
- Verdadeiros negativos (VN): Quando o modelo corretamente classifica um morango como de má qualidade.
- Falsos positivos (FP): Quando o modelo classifica erroneamente um morango de má qualidade como boa.
- Falsos negativos (FN): Quando o modelo classifica erroneamente um morango boa como má.

A partir dessa matriz, é possível calcular métricas como acurácia, precisão, revocação, que ajudam a entender melhor a performance do modelo.



Análise da Importância de Cada Característica:

A função executar_knn_com_exclusao_colunas permite testar o impacto de excluir cada uma das características (como tamanho, peso, acidez) na acurácia do modelo. Isso é feito para identificar quais atributos são mais relevantes para a classificação da qualidade do morango.

Ao observar o impacto de cada exclusão, é possível determinar se uma característica tem pouca influência ou se, ao contrário, é essencial para o bom desempenho do modelo. Essa análise ajuda a simplificar o modelo, caso algum atributo não contribua significativamente para a classificação.

```
Matriz normalizada:
          Size    Weight  Sweetness  Juiciness  Ripeness   Acidity  Quality
0     0.234669  0.358370   0.922484   0.585819  0.472770  0.452225        1
1     0.439331  0.333107   0.795706   0.511408  0.513807  0.436185        1
2     0.505948  0.448092   0.388567   0.660388  0.444693  0.668192        0
3     0.479014  0.376971   0.619422   0.720370  0.187052  0.541180        1
4     0.628107  0.452317   0.490589   0.674814  0.348084  0.521150        1
...        ...       ...        ...        ...       ...       ...      ...
3995  0.531867  0.470029   0.239644   0.574794  0.618866  0.495885        0
3996  0.505867  0.703146   0.504203   0.449218  0.364566  0.614957        1
3997  0.333173  0.387278   0.335661   0.612443  0.811182  0.393744        0
3998  0.231869  0.415014   0.697913   0.609571  0.463966  0.331650        1
3999  0.548031  0.419946   0.528713   0.542429  0.388327  0.597306        1

[4000 rows x 7 columns]
[[538  58]
 [ 84 520]]
Acurácia: 88.16666666666667 %
Precisão:  86.49517684887459  %
Revocação:  90.26845637583892  %
Especificidade:  86.09271523178808  %
```

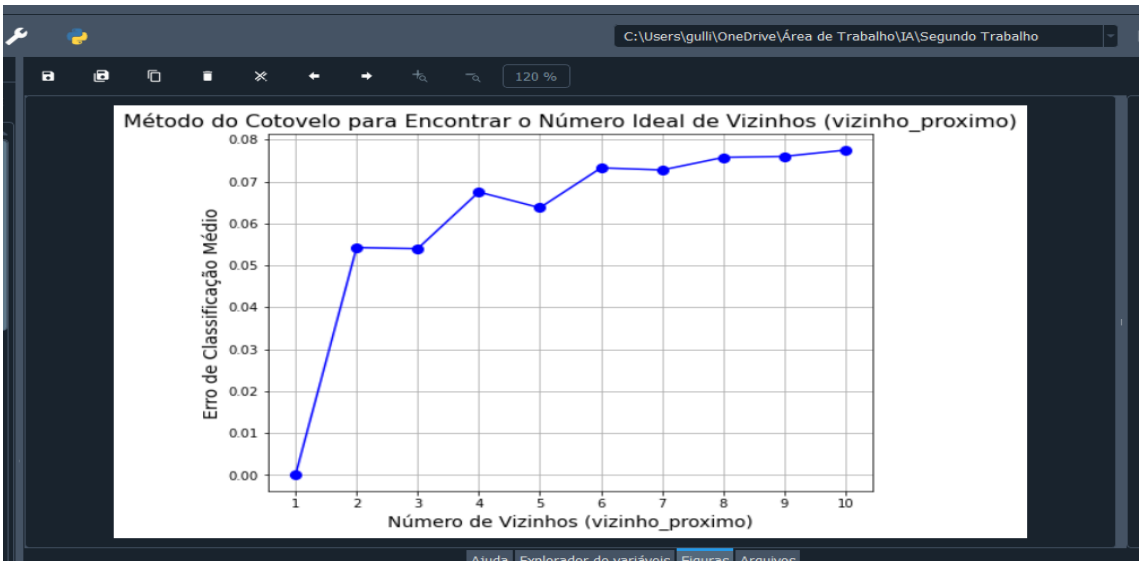
```
Excluindo a coluna: Acidity
Matriz normalizada:
          Size    Weight  Sweetness  Crunchiness  Juiciness  Ripeness  Quality
0     0.234669  0.358370   0.922484     0.368781   0.585819  0.472770        1
1     0.439331  0.333107   0.795706     0.558928   0.511408  0.513807        1
2     0.505948  0.448092   0.388567     0.417732   0.660388  0.444693        0
3     0.479014  0.376971   0.619422     0.435629   0.720370  0.187052        1
4     0.628107  0.452317   0.490589     0.402347   0.674814  0.348084        1
...        ...       ...        ...          ...        ...       ...      ...
3995  0.531867  0.470029   0.239644     0.477379   0.574794  0.618866        0
3996  0.505867  0.703146   0.504203     0.395971   0.449218  0.364566        1
3997  0.333173  0.387278   0.335661     0.490846   0.612443  0.811182        0
3998  0.231869  0.415014   0.697913     0.428137   0.609571  0.463966        1
3999  0.548031  0.419946   0.528713     0.358392   0.542429  0.388327        1

[4000 rows x 7 columns]
[[528  68]
 [106 498]]
Acurácia: 85.5 %
Precisão:  83.2807570977918  %
Revocação:  88.59060402684564  %
Especificidade:  82.45033112582782  %
```

O **Método do Cotovelo** foi uma técnica usada no código para determinar o número ideal de vizinhos (k) no KNN. A função plotar_metodo_cotovelo gera um gráfico que mostra a relação entre o número de vizinhos e o erro médio de classificação.

O "cotovelo" é o ponto onde o erro médio começa a diminuir de forma mais lenta à medida que o valor de k aumenta. Esse ponto indica o valor de k que oferece o melhor equilíbrio entre viés e variância, ajudando a evitar tanto o overfitting quanto o underfitting.

Essa técnica é útil para otimizar o valor de k e melhorar a precisão do modelo sem introduzir complexidade excessiva.



Visualização 3D da Dispersão das Características

A função plotar_dispersao_qualidade gera um gráfico tridimensional que visualiza como as características Size (tamanho), Weight (peso) e Quality (qualidade) dos morangos se relacionam.

Esse tipo de visualização foi feito para análises exploratórias dos dados, pois permite observar visualmente se existem padrões ou tendências nas características que indicam uma boa ou má qualidade.

Por exemplo, se os morangos de boa qualidade tendem a ser maiores e mais pesados, esse gráfico pode destacar essas tendências e ajudar na compreensão de como as características influenciam a classificação.

A visualização 3D foi usando para facilita a identificação de agrupamentos de dados, o que pode ser útil para ajustar o modelo ou fazer ajustes nas características utilizadas para a classificação.