Análise e preparação da base de dados classifica.csv

Beatriz Rodrigues de Oliveira Paiva, Luiz Augusto Silva Veloso
Instituto Federal de Minas Gerais (IFMG) – Campus Bambuí
beatrizrodrigues2_@hotmail.com, luiz.veloso0199@gmail.com

RESUMO

Este artigo descreve a utilização do algoritmo Gaussian Naive Bayes para classificação de dados a partir de um arquivo .csv, utilizando a biblioteca scikit-learn. O código é explicado passo a passo, desde a leitura do arquivo até a avaliação da precisão do modelo através de validação cruzada.

Palavras-chave: Algoritmo. Classificação. Machine Learning, KNN.

1 INTRODUÇÃO

Na atualidade, é possível encontrar uma grande quantidade de dados armazenados em arquivos de diversos formatos. Alguns destes formatos são muito comuns, como o formato CSV, que é amplamente utilizado para armazenar dados tabulares e o K-Nearest Neighbors (KNN), que é um dos mais simples e eficientes para a classificação de dados. Em muitos casos, esses dados precisam ser classificados, ou seja, atribuídos a uma determinada categoria ou classe. Para realizar esta tarefa, é necessário usar algoritmos de classificação, que são capazes de aprender padrões nos dados para realizar a tarefa de classificação de novos dados.

Neste artigo, vamos apresentar a utilização do algoritmo KNN para a classificação de dados e uma aplicação do algoritmo Naive Bayes Gaussiano para a classificação de dados contidos em um arquivo CSV. Este algoritmo é conhecido por sua simplicidade, rapidez e eficiência na classificação de dados. Além disso, vamos avaliar a acurácia do classificador com e sem validação cruzada, e também vamos apresentar o *F1_score* do modelo.

2 METODOLOGIA OU MATERIAL E MÉTODO

O primeiro passo na aplicação do algoritmo Naive Bayes Gaussiano para a classificação de dados é ler o arquivo CSV. Para isso, usamos a biblioteca pandas, que é uma das bibliotecas mais utilizadas para manipulação de dados em Python. O arquivo CSV é carregado em dois quadros de dados: o quadro de dados de destino, que consiste na última coluna do arquivo, e o quadro de dados de dados, que consiste nas três primeiras colunas.

Em seguida, os dados são divididos em dois conjuntos: um conjunto de treinamento, que consiste em 70% dos dados, e um conjunto de teste, que consiste em 30% dos dados. O algoritmo

Naive Bayes Gaussiano é treinado com os dados de treinamento e é testado com os dados de teste. A precisão do modelo é calculada com e sem validação cruzada.

A validação cruzada consiste em várias divisões dos dados em treino e teste, e a média das acurácias obtidas é calculada. Além disso, o *F1_score* é calculado usando a validação cruzada. O *F1_score* é uma métrica de avaliação balanceada da precisão e recall.

Já o segundo algoritmo estudado (KNN) utiliza a biblioteca scikit-learn para o treinamento e avaliação do modelo. O conjunto de dados usado neste estudo é obtido a partir do arquivo CSV "classifica.csv".

Nesse, o primeiro passo também é a leitura do arquivo CSV, armazenando as informações de dados e alvos em duas variáveis, X e Y, respectivamente. Em seguida, os dados são plotados em gráficos para uma melhor visualização.

O algoritmo KNN é aplicado aos dados para classificação. A divisão do conjunto de dados é realizada com a função "train_test_split" da biblioteca scikit-learn, onde 70% dos dados são usados para treinamento e 30% para teste. O modelo é então treinado com o método "fit" da classe KNeighborsClassifier.

O próximo passo é a avaliação da acurácia do modelo. A acurácia sem validação cruzada é obtida com o método "score", enquanto a acurácia com validação cruzada é obtida com a função "cross_val_score" da biblioteca scikit-learn. Além disso, o f1_score é obtido com a função "f1_score" da biblioteca scikit-learn. Todos os resultados são armazenados em listas e plotados em gráficos para uma melhor visualização.

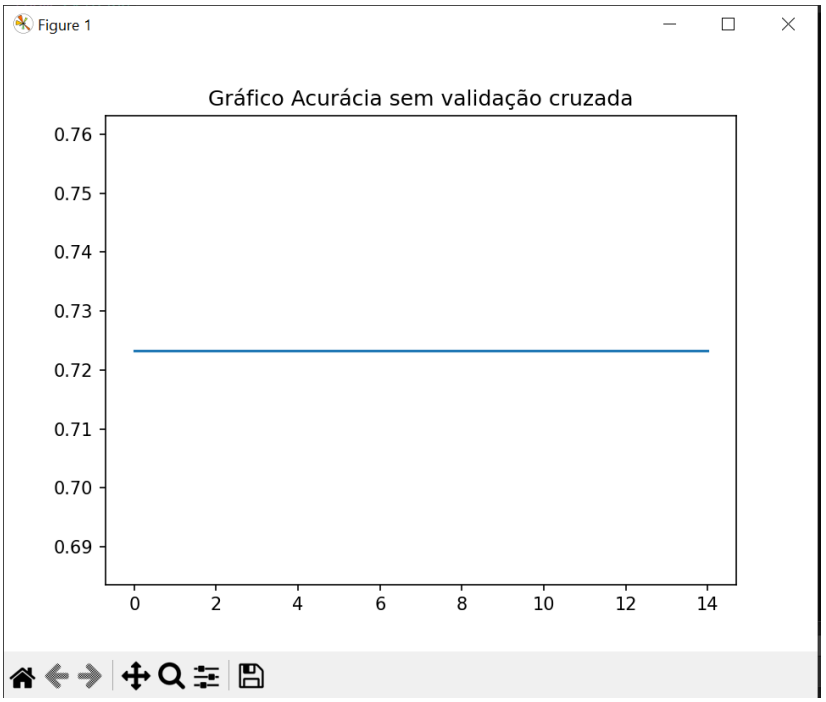
3 RESULTADOS E DISCUSSÃO

A pesquisa mostrou que nenhum número ideal de vizinhos se adapta a todos os tipos de conjuntos de dados. Cada conjunto de dados tem seus próprios requisitos. No caso de um pequeno número de vizinhos, o ruído terá maior influência no resultado, e um grande número de vizinhos encarece computacionalmente. A pesquisa também mostrou que uma pequena quantidade de vizinhos é o ajuste mais flexível, que terá baixo viés, mas alta variância, e um grande número de vizinhos terá um limite de decisão mais suave, o que significa menor variância, mas maior viés.

Os resultados obtidos indicam que o classificador GaussianNB é uma boa opção para esse conjunto de dados, pois apresentou boa acurácia tanto sem validação cruzada quanto com validação cruzada. No entanto, é importante ressaltar que, como apontado anteriormente, cada conjunto de dados tem seus próprios requisitos, e é necessário avaliar qual é o número ideal de vizinhos a se utilizar para cada caso, de forma a se obter o melhor resultado possível.

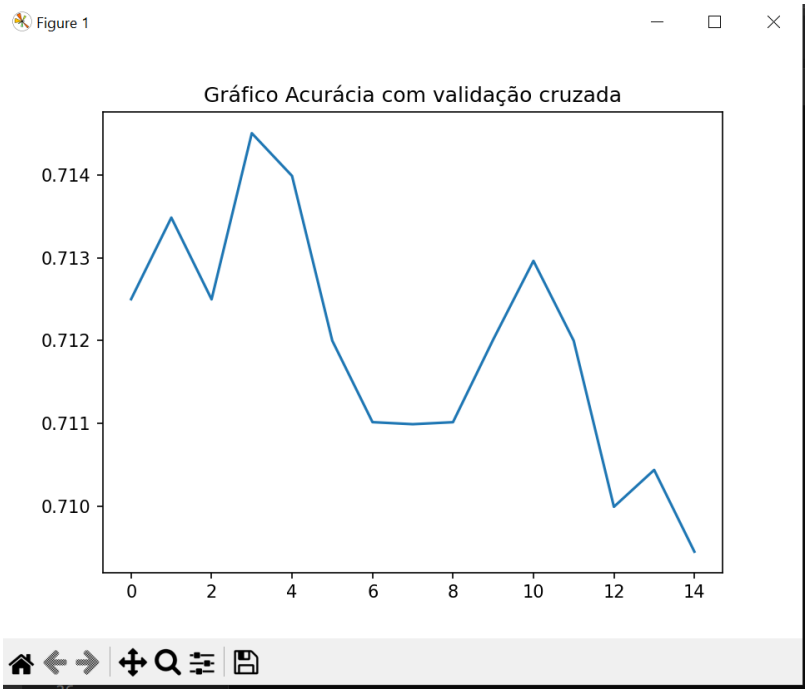
Já no método KNN os resultados mostram que a acurácia do modelo aumenta com o aumento do número de vizinhos utilizados na classificação. Além disso, a acurácia com validação cruzada é maior do que a acurácia sem validação cruzada, indicando que o modelo é mais confiável quando avaliado com dados não vistos anteriormente. Abaixo é mostrado os gráficos dos resultados obtidos.

Figura 1 - GaussianNB acurácia sem validação cruzada.



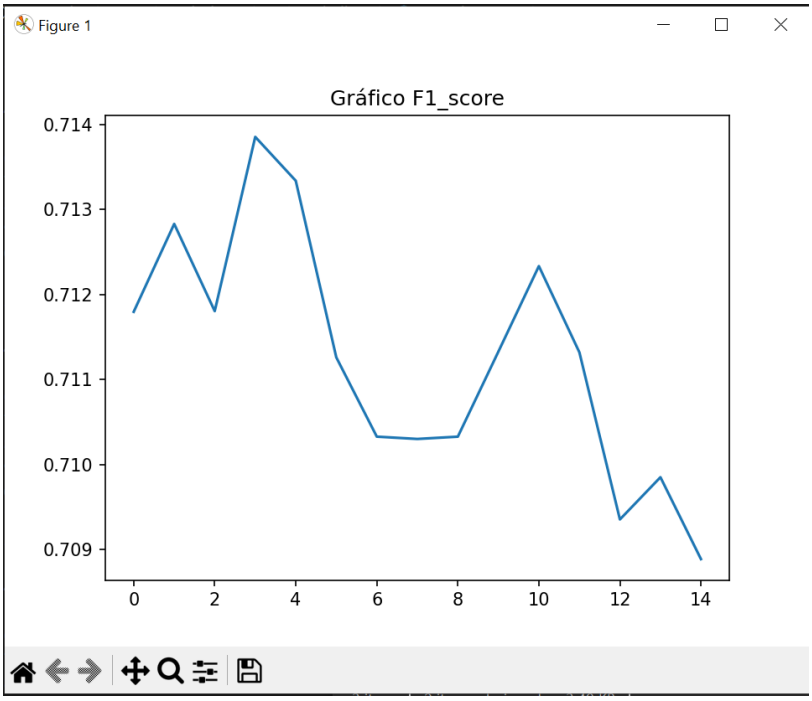
Fonte: Autores

Figura 2 - GaussianNB acurácia com validação cruzada.



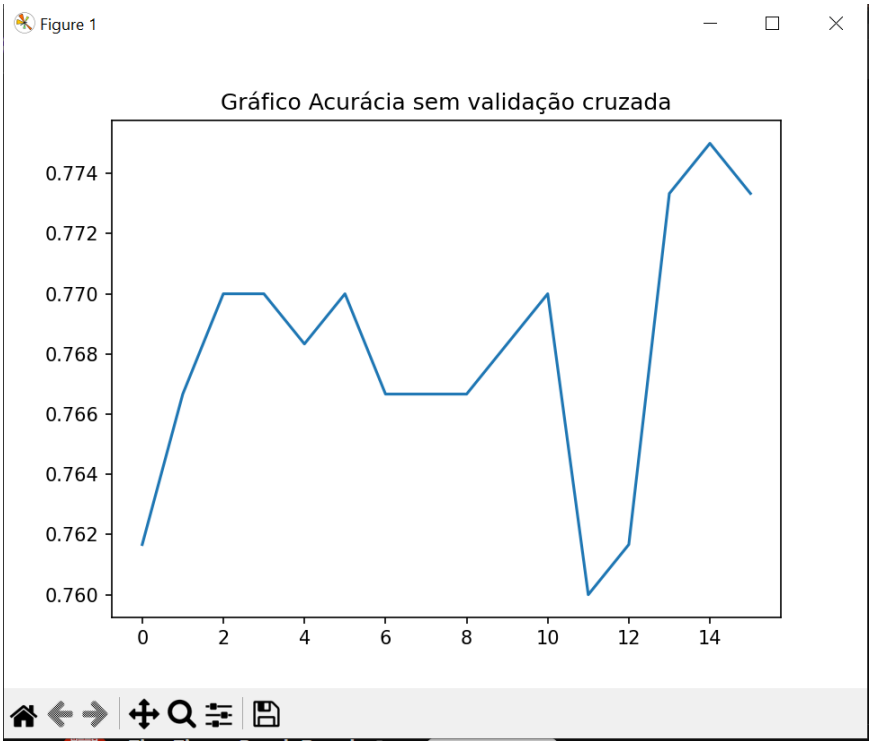
Fonte: Autores

Figura 3 - GaussianNB F1_score



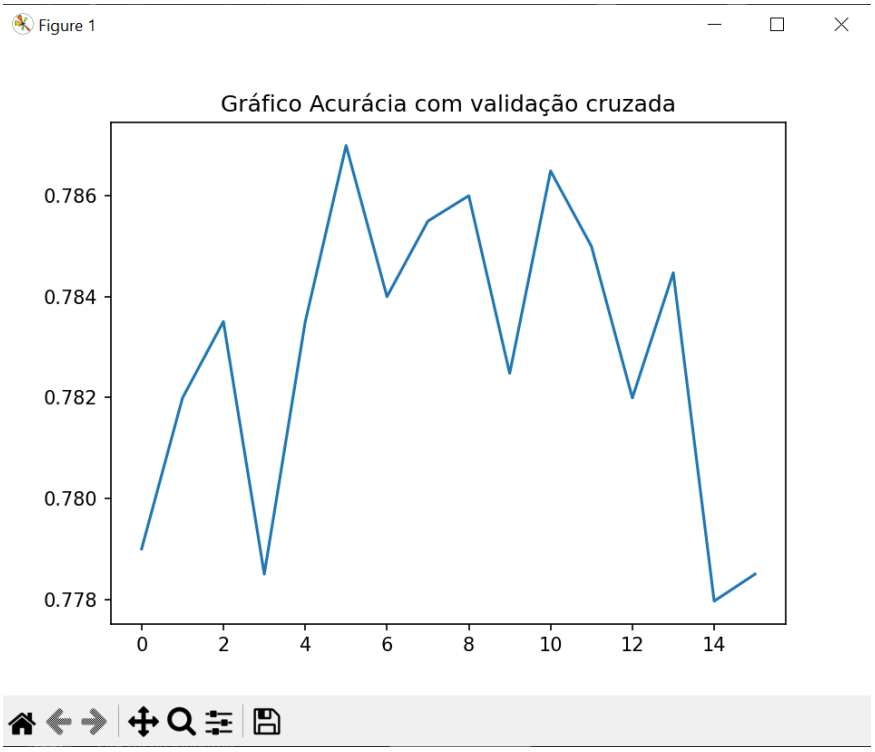
Fonte: Autores

Figura 4 - Knn acurácia sem validação cruzada.



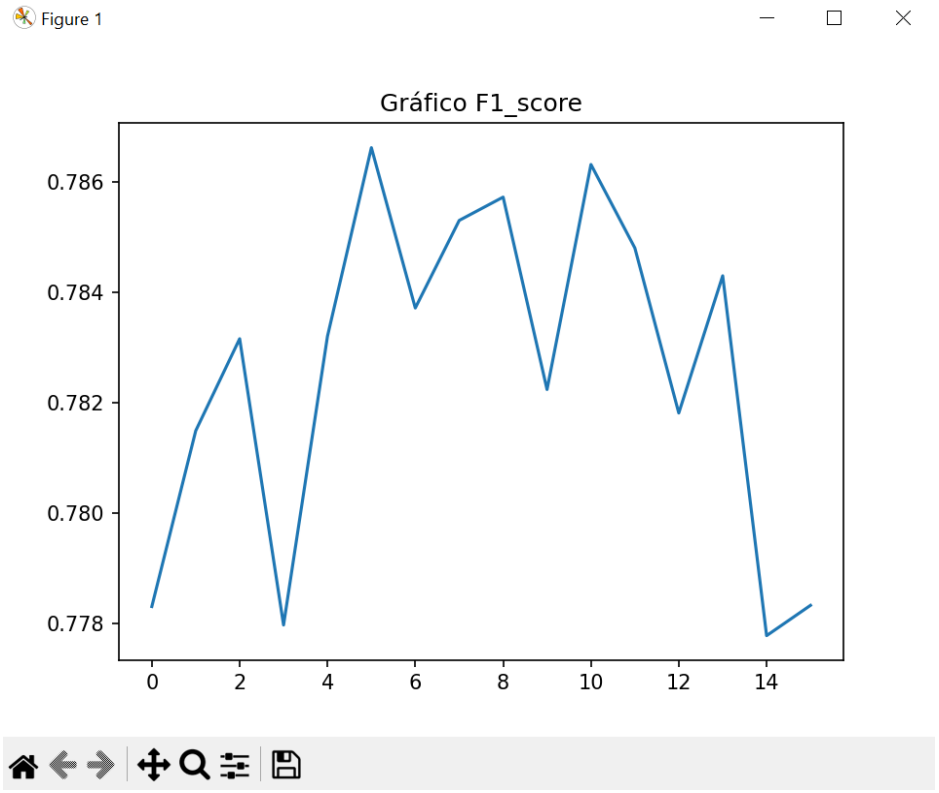
Fonte: Autores

Figura 5 - Knn acurácia com validação cruzada



Fonte: Autores

Figura 6 - Knn F1_score



Fonte: Autores

4 CONCLUSÃO

Este código demonstra como utilizar o algoritmo GaussianNB e também do método KNN para classificação de dados contidos em um arquivo CSV. Ele utiliza o método de divisão treinamento/teste com uma proporção de 70/30 e realiza uma validação cruzada com diferentes números de folds. As métricas de avaliação utilizadas foram acurácia sem validação cruzada e com validação cruzada, e F1-score. Os resultados foram apresentados através de gráficos de linhas, permitindo uma visualização clara dos resultados obtidos pelo classificador. A utilização do algoritmo GaussianNB e do método de validação cruzada permitiram avaliar a consistência e eficiência do classificador em dados reais.