Universidade Federal do Maranhão
Ciência da Computação


Eliã Ribeiro Silva

Israel Barbosa Silva

Matheus Conceição Silva

# Relatório KNN

São Luís

2025

Universidade Federal do Maranhão
Ciência da Computação

# Relatório KNN

Este documento foi entregue à disciplina de Inteligência Artificial, que faz parte do curso de Ciência da Computação da Universidade Federal do Maranhão, com a finalidade de avaliação de desempenho.
Professor: Prof. Dr. Tiago Bonini Borchartt

São Luís
2025

# Sumário

## Resumo

Neste projeto, desenvolvemos um modelo de aprendizado de máquina para prever a sobrevivência de passageiros do Titanic, utilizando o algoritmo K-Nearest Neighbors (KNN). Inicialmente, carregamos e pré-processamos os dados de treinamento, tratando valores ausentes e transformando variáveis categóricas em numéricas. Em seguida, normalizamos os dados para garantir que todas as variáveis estivessem na mesma escala.

Após o pré-processamento, treinamos o modelo KNN com os dados disponíveis e avaliamos seu desempenho utilizando métricas como acurácia, matriz de confusão, relatório de classificação, taxa de erro, especificidade, sensibilidade, precisão e o índice de Youden. Para validar o modelo, aplicamos o mesmo processo aos dados de validação, incluindo carregamento, tratamento de valores ausentes, transformação de variáveis e normalização. As previsões foram então revertidas para a escala original e salvas em um arquivo de resultados.

Além disso, geramos visualizações para analisar a taxa de sobrevivência em relação a diferentes variáveis, como gênero, classe, tarifa paga, presença de familiares a bordo, idade, cabine, local de embarque e tamanho da família. Esses gráficos forneceram insights valiosos sobre os fatores que influenciaram a sobrevivência dos passageiros.

Em resumo, este projeto abrangeu desde o pré-processamento dos dados até a validação e análise dos resultados, demonstrando a aplicação prática do algoritmo KNN na predição de sobrevivência no contexto do Titanic.


**Palavras-chave: knn, aprendizado de máquina**

## 1 - Pré-processamento

No processo de pré-processamento do *dataset* Titanic, primeiramente os dados do arquivo original são carregados em um *DataFrame*. Nesta etapa, faz-se uma verificação inicial para identificar a presença de valores ausentes, observando que colunas como *Age*, *Cabin* e *Embarked* apresentavam dados nulos. A partir dessa avaliação, passou-se a corrigir e tratar esses valores faltantes.

Para a coluna *Age*, decidiu-se pelo preenchimento com a mediana, pois a mediana é menos sensível a valores muito altos ou muito baixos. Já para a coluna *Cabin*, foi criada a coluna *CabinDeck* contendo apenas a letra inicial (A, B, C etc.), e onde não havia registro de cabine, foi atribuído 'X'. Em seguida, como a coluna *Embarked* apresentava apenas dois registros nulos, considerou-se viável remover essas duas linhas, evitando inferências incertas sobre a categoria de embarque que não estava informada.

Além disso, foram removidas colunas que não traziam valor preditivo direto, como *Name* e *Ticket*. Esses campos, por se tratarem de dados mais específicos e textuais, poderiam introduzir ruídos ao modelo, sem benefício considerável para a acurácia. Ainda durante o pré-processamento, as variáveis categóricas passaram por transformações: para a coluna *Sex*, mapeou-se *female* para 0 e *male* para 1. As variáveis *Embarked* (S, C, Q) e *CabinDeck* (A, B, C, D, E, F, G, T, X) foram convertidas em colunas *dummies* (binárias), ampliando a capacidade do algoritmo KNN de lidar com essas categorias de forma numérica.

Por fim, realizou-se a normalização dos atributos numéricos usando o *MinMaxScaler* da biblioteca scikit-learn, que ajusta todos os valores para uma faixa padrão, como entre 0 e 1. Esse passo se mostra essencial para algoritmos baseados em distância, como o KNN, pois garante que nenhuma variável domine as outras unicamente por conta de sua escala. Com todos esses ajustes, definiu-se o conjunto X (*features* preditivas) e y (alvo *Survived*), além de se fazer a partição em treino e teste, comumente no formato 80% para treino e 20% para teste, preparando, assim, o terreno para a etapa de modelagem.

## 2 - Implementação

Após finalizar o pré-processamento, deu-se início à implementação do algoritmo KNN utilizando a biblioteca scikit-learn, por meio da classe *KNeighborsClassifier*. O primeiro passo foi determinar o melhor valor de K, ou seja, quantos vizinhos o modelo deveria considerar para efetuar a classificação. Para isso, aplicou-se a técnica de *cross validation*, que consiste em dividir o conjunto de treino em diversas partes, utilizando algumas delas para treinar e as restantes para validar, em um processo cíclico. Ao testar diferentes valores de K (como 3, 5, 7, 8 e 9) por meio da validação cruzada, observou-se que todos apresentaram desempenhos muito próximos, diferindo em torno de 1% ou 2% na acurácia. Optou-se por manter K=5 em virtude de sua praticidade e de ser um número ímpar, alinhando-se, assim, ao padrão frequentemente adotado em implementações de KNN. Desse modo, alcançou-se um equilíbrio entre a simplicidade do parâmetro e a consistência dos resultados, mantendo a acurácia em patamares equivalentes aos obtidos com outras.

Também foram experimentadas diferentes métricas de distância, como a Euclidiana, a Manhattan e a Minkowski, sem que se observassem variações expressivas na acurácia ou na performance global do modelo. Tal resultado se justifica, em grande parte, pelo tamanho relativamente pequeno do conjunto de dados, o que tende a mitigar as discrepâncias entre as diversas formas de calcular a similaridade entre as instâncias. Com isso, optou-se por manter a distância Euclidiana como padrão, já que ela é a mais habitual no KNN e apresentou resultados praticamente equivalentes aos das demais métricas testadas.

De posse desse valor de K, treinou-se o modelo definindo KNeighborsClassifier(n_neighbors=5) com o conjunto de treino (80% dos dados). Em seguida, o modelo foi testado com o conjunto de teste (os 20% restantes), resultando em métricas de avaliação como a matriz de confusão, o relatório de classificação (precision, recall e f1-score) e a acurácia final. Observou-se que o desempenho permaneceu consistente, com valores de acurácia próximos dos verificados durante a etapa de validação cruzada. Além disso, foi possível analisar, por meio da matriz de confusão, se o modelo cometia mais erros ao prever

sobreviventes ou não sobreviventes, auxiliando a compreender melhor o comportamento do KNN nessa tarefa de classificação.

Por fim, vale destacar que o KNN pode ser sensível à normalização dos dados e à escolha de K, justificando a relevância de testes para diferentes valores de K e de técnicas como a MinMaxScaler, que asseguram escalas comparáveis entre todos os atributos. Mesmo não sendo um método tão sofisticado quanto modelos mais complexos, o KNN se mostrou relativamente eficaz para predizer a sobrevivência de passageiros do Titanic, principalmente após todas as etapas de pré-processamento, escolha de hiperparâmetros e avaliações cuidadosas dos resultados obtidos.

**3 - Treino e Teste**

Como dito anteriormente, para efetuar o treinamento do modelo, optou-se por K=5, considerando que esse valor, ainda que não tenha apresentado grande distância em termos de desempenho para outros K testados, mostrou-se suficientemente competitivo e coerente com a prática de escolher um número ímpar. Assim, definiu-se KNeighborsClassifier(n_neighbors=5) e treinou-se o algoritmo com 80% dos dados, mantendo o padrão de dividir o conjunto em treino e teste.

Concluída essa etapa, procedeu-se à avaliação com os 20% remanescentes, calculando-se métricas como acurácia, precisão, revocação e f1-score por meio de funções como *accuracy_score, precision_score e recall_score*. Além disso, a matriz de confusão forneceu um panorama sobre quantos sobreviventes ou não sobreviventes foram corretamente ou incorretamente previstos. Em geral, a acurácia situou-se na faixa de 78%, espelhando o que já havia sido identificado durante os testes de validação cruzada e confirmando a solidez do KNN na presente tarefa.

Como forma de verificar a real influência da normalização, realizou-se ainda um teste sem normalizar os dados, o que resultou em uma acurácia em torno de 68%. Assim, constatou-se que a simples adoção do MinMaxScaler trouxe uma melhora de aproximadamente 10% no desempenho do modelo, reforçando a importância de escalar atributos quando se recorre a algoritmos baseados em distância. Essa diferença se explica pelo fato de o KNN calcular a similaridade por

métricas de distância, de modo que trabalhar com valores brutos (frequentemente em escalas distintas) pode levar o algoritmo a produzir previsões menos precisas.

Com o modelo devidamente treinado, recorreu-se a outro arquivo contendo dados distintos daqueles usados no treinamento, e que não incluía a coluna de alvo (Survived). Dessa forma, o KNN pôde gerar, para cada passageiro, uma predição indicando se ele sobreviveria ou não. O resultado foi armazenado em um CSV, contendo o ID do passageiro e o valor previsto de sobrevivência (0 ou 1). Esse arquivo foi então submetido à avaliação externa, uma vez que os verdadeiros rótulos (Survived) não faziam parte do arquivo de validação, cabendo ao professor verificar a performance final do modelo

Vale ressaltar que o KNN, em especial para bases de tamanho mais modesto, muitas vezes não exibe oscilações drásticas de desempenho ao se variar ligeiramente os parâmetros ou as métricas de distância. Na prática, o modelo conseguiu demonstrar boa estabilidade, classificando de forma bastante satisfatória a sobrevivência dos passageiros do Titanic com os ajustes realizados na etapa de pré-processamento e a escolha do K=5.

## 5 - Conclusão

Em suma, a implementação do algoritmo KNN para a predição de sobrevivência no dataset do Titanic apresentou resultados consistentes e elucidativos. O pré-processamento desempenhou papel fundamental, evidenciando que a limpeza dos dados, a transformação de variáveis categóricas em dummies e, sobretudo, a normalização foram determinantes para elevar a acurácia em cerca de 10%. Ademais, as análises de diferentes valores de K e métricas de distância, por meio da validação cruzada, mostraram que o modelo é relativamente estável em bases menores, com diferenças de desempenho pouco significativas entre as combinações testadas. Diante disso, optou-se por K=5 e distância Euclidiana, estratégia que se provou suficientemente eficaz, culminando em resultados que superaram 78% de acerto. Essa experiência reforça a importância de boas práticas de pré-processamento e ajuste de parâmetros para maximizar o potencial de algoritmos baseados em distância.

O trabalho aqui citado, está presente no link abaixo, disponível no Google Colab

KNN - Predição Titanic

## Referências

RUSSELL, Stuart; NORVIG, Peter. Inteligência artificial. 3ª ed. Rio de Janeiro: Elsevier, 2013.