Previsão de ataques cardíacos utilizando KNN e SVM

Ewerson Vieira Nascimento
Aluno de Sistemas de Informação
Instituto Federal do Espírito Santo
Vila Velha, Brasil
ewersonv@gmail.com

José Guilherme Silva de Lima
Aluno de Sistemas de Informação
Instituto Federal do Espírito Santo
Serra, Brasil
guislima47@gmail.com

Resumo— *Neste artigo foi realizada a comparação entre algumas técnicas de classificação de dados, com o intuito de fazer previsões de ataques cardíacos em pacientes que deram entrada em hospitais, através da análise de métricas contidas em um modelo com uma base de 303 pacientes já atendidos.*
Para fazer a comparação foi feita a implementação dos algoritmos de KNN, SVM com Kernel Linear e SVM com Kernel RBF, utilizando a linguagem Python com o auxílio de algumas bibliotecas, como Pandas, Numpy e Sklearn

I. Introdução (*Heading 1*)

O setor de emergência dos hospitais recebe um grande número de pacientes frequentemente e presta o tratamento inicial para uma diversidade de doenças e enfermidades. A fim de agilizar o diagnóstico destes pacientes, o hospital fictício utilizado como exemplo neste trabalho quer usar as informações que dispõe a respeito de 303 pacientes que foram atendidos anteriormente para poder prever se um paciente irá sofrer um ataque cardíaco ou não, podendo, desta forma, prestar um serviço de mais qualidade e de menor risco para os pacientes. Para realizar esta predição, foram utilizados diferentes algoritmos de Aprendizado de Máquina para determinar aquele, dentre os utilizados, que seria o melhor para o objetivo estabelecido.

II. Referencial Teórico

Aprendizado de Máquina

Como o próprio nome já diz, Machine Learning (Aprendizado de Máquina) é uma área da computação que se baseia em técnicas que visam tornar possível a criação de algoritmos que, através de inputs (entrada de dados), constroem modelos para que consigam, sem programação explícita, fazer previsões e tomar decisões. Ou seja, estamos falando de máquinas com capacidade de aprenderem sozinhas. Esses algoritmos usam o que é chamado de raciocínio indutivo, que é a extração de regras e padrões a partir de um grande conjunto de dados, para aprender com essa conexões e com os seus próprios erros.

Existem dois tipos de categorias que são mais conhecidas para esses tipos de algorítmos:

Aprendizado Supervisionado: dados de entrada e saída de exemplo são fornecidos por um terceiro como dados de treino, para que o algoritmo aprenda com as regras e tome decisões sobre os próximos dados fornecidos.

Aprendizado não Supervisionado: nenhum dado de exemplo é fornecido ao algoritmo. Portanto, ele deve encontrar sozinho uma forma de encontrar padrões e tomar decisões.

.

Nearest Neighbors - KNN

O algoritmo Nearest Neighbors (*KNN*) é um algoritmo da categoria de aprendizado supervisionado utilizado para classificação e clusterização de objetos, ou seja, sua principal função é classificar objetos de um conjunto de dados de acordo com as suas características e agrupá-los.
É um algoritmo de classificação de dados que possui um funcionamento muito simples, onde a classificação de um elemento K é baseada nos seus elementos vizinhos, ou seja, na classe da maioria dos elementos que se encontram próximos a ele.

Para encontrar quem são os vizinhos de um elemento é necessário fazer um cálculo da distância entre ele e os demais elementos da base de dados. A distância Euclidiana é a medida mais utilizada, mas existem muitas outras, como distância de Hamming, distância de Manhattan e distância de Minkowski.

SVM

O SVM, ou Máquinas de Vetores de Suporte (support vector machines), é um conceito para definir um conjunto de algoritmos de aprendizado supervisionado, com a capacidade de analisar e classificar dados através de uma função de kernel, onde é realizado o cálculo da distância entre cada elemento de entrada e o hiperplano, que é uma linha de separação capaz de dividir, de forma equilibrada, todos os elementos de um plano em duas classes de acordo com suas características.

Para fazer a classificação, é necessário um conjunto de de dados de treinamento, onde cada elemento possui atributos e uma classe à qual pertence. Dessa forma, o algoritmo consegue construir um modelo e fazer a classificação dos novos dados inseridos.

Existem diversas funções de kernel utilizadas para esse tipo de algoritmo, mas as mais conhecidas são o Kernel Linear e o Kernel RBF.

III. METODOLOGIA

Para o desenvolvimento da pesquisa, foi utilizada a linguagem Python junto com as seguintes bibliotecas:

Pandas: Biblioteca para manipulação e análise de dados

Numpy: Biblioteca para manipulação e operação com arrays e matrizes multidimensionais

Sklearn: Biblioteca com funções específicas de aprendizado de máquina.

Com o auxílio das bibliotecas citadas, foram implementados os algoritmos de *KNN e SVM* e encontradas as matrizes de confusão, que são tabelas que mostram as frequências de classificação para cada classe do modelo. Para fazer a validação do modelo, foi utilizada uma técnica de validação cruzada que consiste basicamente na divisão do conjunto de dados em subconjuntos, onde alguns deles são utilizados como conjuntos de treino e o restante como conjuntos de teste.

A técnica de validação cruzada escolhida foi a K-Fold, com K = 5, em que o conjunto total é dividido em K subconjuntos, onde apenas um conjunto é utilizado para teste e os demais como conjuntos de treino. O processo é repetido K vezes, alternando o conjunto de teste.

Após a realização das K repetições foi criada uma matriz de confusão com os resultados acumulados, isto é, uma matriz de confusão composta pela soma de todas as matrizes de confusão obtidas durante o processo.

Frequências exibidas na matriz de confusão:

Verdadeiro Positivo (TP)*:* Classificação correta de um valor positivo .

Falso Positivo (FP): Classificação incorreta de um valor positivo.

Falso Verdadeiro (TN): Classificação incorreta de um valor verdadeiro.

Falso Negativo (FN): Classificação incorreta de um valor negativo.

Informações retiradas através da matriz de confusão:

Acurácia: Quantas classificações foram feitas corretamente.
Calculo:

$$\frac{TP + TN}{TP + FP + TN + FN}$$

No nosso trabalho, a cada divisão treino/teste obtivemos uma matriz de confusão diferente e uma acurácia relacionada à esta matriz de confusão. Portanto, para obter a média das acurácias, somamos as acurácias e dividimos pelo número de repetições realizadas, que foram 5 neste caso (k-fold com 5 divisões).

Recall: De todas as classificações que deveriam ser positivas, quantas foram feitas corretamente.
Calculo:

$$\frac{TP}{TP + FN}$$

Precisão: De todas as classificações de valor Positivo, quantas foram feitas corretamente.
Calculo:

$$\frac{TP}{TP + FP}$$

IV. RESULTADOS

Nos testes que realizamos, utilizamos a biblioteca *Pandas* para ler o arquivo .CSV que contém a base de dados necessária para implementação dos algoritmos. Para o algoritmo KNN, geramos matrizes de confusão diferentes para cada uma dos testes com diferentes números de vizinhos (*n_neighbors* utilizando o *KNeighborsClassifier* da biblioteca *sklearn*). A seguir estão listados os resultados para cada um dos algoritmos implementados.

KNN

- **Número de vizinhos igual a 3**

Média das acurácias: 49.75%

Precisão: 40.57%

Recall: 44.44%

- **Número de vizinhos igual a 5**

Média das acurácias: 49.73%

Precisão: 38.40%

Recall: 44.16%

- **Número de vizinhos igual a 7**

Média das acurácias: 49.06%

Precisão: 35.50%

Recall: 42.98%

- **Número de vizinhos igual a 9**

Média das acurácias: 50.38%

Precisão: 36.95%

Recall: 44.73%

Observando os resultados obtidos com o algoritmo KNN, temos que a maior precisão, 40.57%, foi obtida com os testes utilizando o número de vizinhos igual a 3. O teste que obteve o pior resultado foi com o número de vizinhos igual a 9, onde conseguimos uma precisão de apenas 35.50%. À medida que aumentávamos o número de vizinhos para além de 9, observamos que tanto a precisão quanto o *recall* diminuíam. Porém, o teste que obteve a maior média das acurácias e o maior *recall* foi com 9 vizinhos.

SVM Kernel Linear

- Precisão 68.84%
- Recall 76.61%

Analisando os resultados obtidos com o SVM com Kernel Linear notamos que este algoritmo foi o que obteve o melhor resultado dentre todos os testados neste trabalho. A sua precisão foi de 68.84% e o recall de 76.61%, o que significa cerca de 1,5 vez a precisão do algoritmo KNN e quase 10 vezes a precisão do Kernel RBF. Isso se deve à separação linear dos dados, em comparação ao KNN.

SVM Kernel RBF

- Precisão 7.24%
- Recall 11.23%

Os resultados dos testes com Kernel RBF nos mostram que esse algoritmo obteve o pior resultado dentre os 3 algoritmos testados neste trabalho, sendo muito inferior ao algoritmo KNN, que ficou em segundo lugar. O Kernel RBF nos trouxe a menor precisão e o menor recall.

V. Conclusão

A previsão de diagnósticos por meio de modelos estatísticos é muito utilizada na área da saúde, porém, com o avanço da tecnologia e dos algoritmos de aprendizado de máquina, o esforço para realizar estas previsões caiu drasticamente, como observado na produção deste trabalho. Com poucas linhas de código conseguimos realizar a previsão de objetivo deste trabalho com 3 diferentes algoritmos. Dentre estes, o que trouxe o melhor resultado em termos de precisão foi o SVM com Kernel Linear, mostrando que mesmo com um algoritmo simples, podemos obter resultados relativamente bons.

Referências

[1] https://medium.com/datadriveninvestor/k-fold-cross-validation-6b8518070833

[2] https://www.aionlinecourse.com/tutorial/machine-learning/k-fold-cross-validation

[3] https://stackabuse.com/k-nearest-neighbors-algorithm-in-python-and-scikit-learn/

[4] https://stackabuse.com/implementing-svm-and-kernel-svm-with-pythons-scikit-learn/

[5] https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html

[6] https://towardsdatascience.com/building-a-k-nearest-neighbors-k-nn-model-with-scikit-learn-51209555453a

[7] https://medium.com/data-hackers/entendendo-o-que-%C3%A9-matriz-de-confus%C3%A3o-com-python-114e683ec509