APRENDIZADO SUPERVISIONADO E MÉTODOS DE CLASSIFICAÇÃO

Bruce Williss Pires de Mendonça
Universidade Federal de São João Del-Rei
iambruc3@aluno.ufsj.edu.br

Abstract

The purpose of this study is to introduce and induce five known supervised machine learning algorithms to analyse and classify 4 different datasets and compare their performance. We will be focusing on K-Nearest Neighbors algorithm, also known as KNN, which one was implemented from scratch.

Key-words: supervised learning, machine learning, decision trees, id3 classifier, likelihood classification, naïve Bayes, Bayes classifier, svm classification, superficial neural networks classification, knn classification, k nearest neighbors, supervised learning comparison.

1 Introdução

Aprendizado de Máquina (do inglês: Machine Learning) é um subcampo da Engenharia e da Ciência da Computação que evoluiu do estudo de reconhecimento de padrões e da teoria do aprendizado computacional em inteligência artificial[1]. Em 1959, Arthur Samuel definiu aprendizado de máquina como o "campo de estudo que dá aos computadores a habilidade de aprender sem serem explicitamente programados"[2]. O aprendizado automático explora o estudo e construção de algoritmos que podem aprender de seus erros e fazer previsões sobre dados.

O aprendizado de máquina compreende realizar tarefas que são tipicamente classificadas em três grupos: aprendizado supervisionado, em que o objetivo é aprender uma regra geral que mapeia as entradas para as saídas, nesse caso é utilizado um agente externo que indica a resposta desejada para o padrão de entrada; o aprendizado não supervisionado, em que o algoritmo lida sozinho com as entradas e tem como objetivo encontrar padrões nos dados para alcançar um meio de atingir um fim e, por último, o aprendizado por reforço, em que o algoritmo – chamado de agente - interage com ambientes dinâmicos e deve desempenhar um determinado objetivo maximizando seu conhecimento e seu desempenho.

O foco do estudo a ser apresentado compreende descrever, comparar e analisar cinco diferentes algoritmos de aprendizado de máquina que terão seu foco em atuar sobre tarefas de aprendizado supervisionado à classificação. São eles: K-Nearest Neighbors (em tradução: K-Vizinhos Mais Próximos - KNN), Árvores de Decisão, Naive-Bayes, Redes Neurais Artificiais(RNAs) e Máquina de Vetor de Suporte (SVM).

Esses algoritmos serão capazes de receber como entrada, separadamente, quatro conjuntos de dados coletados em repositórios públicos (chamados de datasets), e, ao atuar sobre eles, fornecerão predições e classificações sobre as informações neles analisadas.

O objetivo principal é compreender o funcionamento e comparar a performance de cada algoritmo ao analisar cada um desses conjuntos e quantificar sua performance por meio de métricas de desempenho.

2 Fundamentos

Nesta seção os os algoritmos de aprendizado supervisionado que foram mencionados anteriormente serão melhor definidos abordando seus aspectos técnicos e também avaliando o propósito de uso de cada um. Além disso, também será importante avaliar suas vantagens e desvantagens sobre determinados usos para que se possa inferir conclusões sobre o desempenho dos algoritmos ao fim do estudo.

Um foco maior será dado ao algoritmo KNN pois este foi escolhido para ser implementado sem o auxílio de bibliotecas externas de aprendizado de máquina da linguagem Python, a qual será utilizada no estudo. Portanto, detalhes sobre fundamentos matemáticos, detalhes sobre implementação e tomadas de decisão serão abordadas com mais ênfase sobre esse algoritmo.

2.1 Indução por Árvores de Decisão

Árvores de decisão (Decision Trees) são algoritmos amplamente utilizados em problemas de inferência indutiva no aprendizado de máquina, além de ser uma das formas mais simples de aprendizado supervisionado.

Como o nome sugere, nesses algoritmos vários pontos de decisão são criados, chamados de nós da árvore. Em cada um deles o resultado da decisão tomará um caminho ou outro, criando um fluxo ao longo da árvore.

De modo geral, a estrutura de dados utilizada nesse modelo é a ávore binária, a qual possui um nó chamado nó raíz, sendo o de maior nível hierárquico da árvore, e ligações para outros elementos chamados nós filhos, que por sua vez se expandem de modo a obter os nós folha, ou terminais, que são compreendidos nos níveis hierárquicos mais baixos da árvore.

Em uma árvore de decisão, decisões são tomadas através do caminhamento a partir do nó raíz até um nó folha. O nó folha é determinado com base em cada decisão tomada no caminhamento intermediário da árvore.
Tratando-se de uma árvore binária, as decisões mencionadas são tomadas com base em perguntas com opções de resposta sim ou não. Uma resposta "sim" levará a uma próxima pergunta, e a resposta "não" levará a outra[3]. Essa série de perguntas permite a construção da árvore de decisão, partindo de um ponto comum que é o nó raíz. Várias possibilidades de caminhos diferentes, chamados de ramos, podem ser tomados na árvore, e cada um levará a um resultado diferente. Esses são os elementos compreendem a estrutura básica de uma árvore de decisão.

***Exemplo 1**: Árvore decisão sobre "jogar ou não jogar tênis?":*

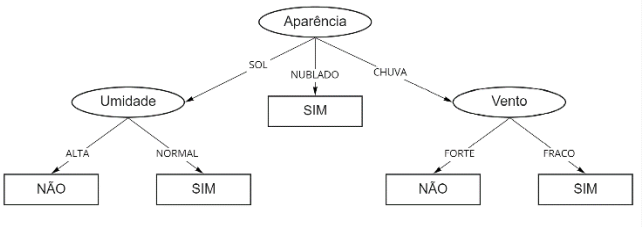

Figura 1: Jogar ou não jogar tênis com base no clima

A árvore acima é gerada com base em um dataset, que serão representados por tabelas ou listas:

Dia	*Aparência*	*Umidade*	*Vento*	*JogarTênis*
D1	*Ensolarado*	*Alta*	*Fraco*	*Não*
D2	*Ensolarado*	*Alta*	*Forte*	*Não*
D3	*Nublado*	*Alta*	*Fraco*	*Sim*
D4	*Chuvoso*	*Alta*	*Fraco*	*Sim*
D5	*Chuvoso*	*Normal*	*Fraco*	*Sim*
D6	*Chuvoso*	*Normal*	*Forte*	*Não*
D7	*Nublado*	*Normal*	*Forte*	*Sim*

Tabela 1: Dataset associado à arvore da Fig. 1

(A tabela acima é um exemplo de representação dos conjuntos de dados em que os algoritmos de classificação atuarão.)

Em suma, tem-se:
Entrada: Uma situação descrita por um conjunto de propriedades ou atributos, chamados de exemplos. Geralmente, a última coluna da tabela representa o chamado atributo classe, que é o objetivo a ser alcançado.
Saída: Uma decisão (saídas binárias: "sim" ou "não")

Árvore de decisão: Formada por um conjunto de nós de decisão, que configuram as perguntas feitas que permitem a classificação de cada caso.
Em geral, árvores de decisão são adequadas para problemas em que se têm exemplos previamente classificados por um especialista, chamados **pares atributo-valor**[4]. Ou seja, há um conjunto fixo de atributos, como no exemplo anterior o atributo "Umidade" pode assumir os valores "alta" e "normal", e também o problema é composto por regras desconhecidas que o algoritmo irá descobrir – ou aprender, propriamente dito.

Em termos mais técnicos, uma árvore pode ser representada como um conjunto de regras disjuntas do tipo se-então (*if, else, elif, etc*) onde cada regra começa na raiz da árvore e caminha para baixo, em direção às folhas. Cada nó de decisão acrescenta um teste às premissas (condições) da regra e nó folha representa a conclusão da regra. Árvores de decisão também podem ser usadas para classificar novos exemplos nunca vistos no conjunto de dados de entrada.

A situação ideal de uso é quando cada atributo pode assumir poucos valores, no entanto, as árvores de decisão também podem trabalhar com valores reais e podem ser estendidas para produzir mais de dois valores de saída além de "sim" ou "não", entretanto, tornam-se mais complexas e menos utilizadas em abordagens que buscam produzir tais valores reais como saída.

Normalmente, os dados de treinamento podem conter erros e valores de atributos faltantes, porém as árvores de decisão são robustas a erros, tanto erros nas classificações dos exemplos de treinamento, quanto erros nos valores dos atributos que descrevem estes exemplos[4].

Contudo, observa-se que a noção por trás de uma árvore de decisão é intuitiva e de simples compreensão, não sendo necessário um conhecimento estatístico aprofundado para sua interpretação. O aprendizado utilizando árvores de decisão, no mundo real, é aplicado amplamente a problemas como classificar os pacientes médicos pela doença, causa de maus funcionamentos de equipamentos, e a probabilidade de candidatos a empréstimo ficarem inadimplentes. Tais problemas, nos quais a tarefa é classificar exemplos em possíveis categorias discretas, são os chamados de problemas de classificação[4].

2.2 Naive-Bayes

Os classificadores Naive-Bayes, por sua vez, são uma família de classificadores probabilísticos. Em uma situação de aprendizado supervisionado, os Classificadores Naive Bayes são treinados de forma muito eficiente. Esses algoritmos classificam um conjunto de entradas z com base na aplicação do teorema de Bayes com a suposição Naive (ingênua) de independência entre os atributos de *z*, dada a variável de classe *t*.

Em termos práticos, o algoritmo recebe o adjetivo "naive" (ingênuo) no nome, pois ele desconsidera completamente a correlação entre os atributos (*features*). Ou seja, se determinada fruta é considerada uma "Maçã" se ela for "Vermelha", "Redonda" e possui "aproximadamente 10cm de diâmetro", o algoritmo não vai levar em consideração a correlação entre esses fatores, tratando cada um de forma independente.[7]

A lógica Bayesiana parte da suposição de que as quantidades de interesse são reguladas por distribuições de probabilidade, que são funções que descrevem a probabilidade de uma variável aleatória assumir certos valores. Com isso, decisões ótimas podem ser tomadas com base nessas probabilidades em conjunto com os dados observados.[8]

Para melhor entendimento, deve-se entender o **Teorema de Bayes**. Esse teorema estabelece a seguinte relação entre dois eventos A e B, num mesmo espaço amostral, com probabilidades P(A) e P(B), respectivamente:

$$P(A|B) = \frac{P(B|A)}{P(B)} * P(A)$$

Fórmula 1: Probabilidade condicional

A ideia é tentar calcular a probabilidade denominada como posteriori P(A|B) a partir de P(A), P(B) e P(B|A), onde cada uma representa:

- P(A|B) é chamada probabilidade condicional posterior (a posteriori) da classe (A, objetivo) dada preditor (B, atributos): A probabilidade de acontecer A dado que B é verdadeiro;
- P(A) é a probabilidade original da classe (A);
- P(B|A) é a probabilidade condicional que representa a probabilidade ocorrer o preditor (B) dado que uma classe objetivo (A) é verdadeira;
- P(B) é a probabilidade original do preditor(B).

O método pode ser usado quando os atributos que descrevem as instâncias forem condicionalmente independentes. Ou seja, o teorema de Bayes trata sobre probabilidade condicional, isto é, qual a probabilidade de o evento A ocorrer, dado o evento B. A classificação final é produzida combinando as duas fontes de informação, ou seja, a anterior e a verossimilhança, para formar uma probabilidade posterior usando a regra de Bayes.

Exemplo 2: *Funcionamento desse classificador baseando-se na fórmula 3: [7]*
"Cientistas estão trabalhando no diagnóstico de uma nova doença e realizaram testes em 100 pessoas distintas. Após coletarem a análise, descobriu-se que 20 pessoas possuíam a doença (20%) e 80 pessoas estavam saudáveis (80%), sendo que das pessoas que possuíam a doença, 90% receberam positivo no teste da doença, e 30% das pessoas que não possuíam a doença também receberam o teste positivo."

Listando esses dados de uma forma mais clara, tem-se:

- *100 pessoas realizaram o teste.*
- *20% das pessoas que realizaram o teste possuíam a doença.*
- *90% das pessoas que possuíam a doença, receberam positivo no teste.*
- *30% das pessoas que não possuíam a doença, receberam positivo no teste.*

A pergunta neste caso seria: ***Se uma nova pessoa realizar o teste e receber um resultado positivo, qual a probabilidade de ela possuir a doença?***

O algoritmo de ***Naive-Bayes*** *consiste em encontrar uma probabilidade a posteriori (possuir a doença, dado que recebeu um resultado positivo), multiplicando a probabilidade a priori (possuir a doença) pela probabilidade de "receber um resultado positivo, dado que tem a doença".*

Deve-se também computar a probabilidade a posteriori da negação (Não possuir a doença, dado que recebeu um resultado Positivo).

Ou seja:

- *P(doença|positivo) = 20% * 90%*
- *P(doença|positivo) = 0,2 * 0,9*
- *P(doença|positivo) = 0,18*
- *P(não-doença|positivo) = 80% * 30%*
- *P(não-doença|positivo) = 0,8 * 0,3*
- *P(não-doença|positivo) = 0,24*

Após isso é necessário normalizar os dados, para que a soma das duas probabilidades resulte 1 (100%).

Para isso, divide-se o resultado pela soma das duas probabilidades:

- *P(doença|positivo) = 0,18/(0,18+0,24) = 0,4285*
- *P(não doença|positivo) = 0,24/(0,18+0,24) = 0,5714*
- *0,4285 + 0,5714 =* ***0,9999.. ou aprox. 1****.*

Pode-se concluir que se o resultado do teste da nova pessoa for positivo, ela possui aproximadamente 43% (0,4285) de chance de estar doente.

Uma característica importante desse classificador é que ele necessita apenas de um pequeno número de dados de teste

para concluir classificações com uma boa precisão, não necessitando da grande parte do conjunto de dados como os demais algoritmos de classificação, portanto se sobressai a os outros métodos de classificação aqui citados e até mesmo aos algoritmos altamente sofisticados com esse propósito[11].

Além disso, é extremamente útil utilizar esse classificador para lidar com grandes volumes de dados, devido ao fato de ser rápido e altamente escalável, e é muito robusto para previsões em tempo real, ainda mais por precisar de poucos dados para realizar a classificação. No entanto, caso haja necessidade de correlacionar fatores, o algoritmo tende a falhar na predição pois, na vida real, dificilmente encontra-se recursos independentes[9]. Um problema recorrente nesse algoritmo é o fato de que se uma categoria não for capturada no conjunto de treinamento e aparecer no conjunto de dados de teste, o modelo é atribuído a probabilidade 0 (zero), o que leva ao cálculo incorreto. Este fenômeno é conhecido como 'Freqüência Zero' mas existem técnicas de suavização[12].

Entre as possibilidades de aplicações, no geral, modelos baseados no teorema de Bayes são frequentemente aplicados em processamento de linguagem natural e diagnósticos médicos, como no exemplo visto anteriormente. Atualmente são muito populares para aplicação em filtros de spam onde é usado para categorizar textos baseado na frequência das palavras usadas, e assim pode ser usado para identificar se determinado e-mail é um spam ou sobre qual assunto se refere determinado texto, por exemplo[7].

2.3 Indução por Redes Neurais Artificiais

Redes Neurais Artificiais (RNAs) são modelos computacionais que apresentam um modelo matemático inspirado na estrutura neural de organismos inteligentes e que adquirem conhecimento através da experiência[13].

Acredita-se que a capacidade de processamento do cérebro emerge de redes de neurônios. Por essa razão, uma parte do estudo inicial no ramo da IA teve como objetivo criar **redes neurais artificiais**[14]. RNAs geralmente são apresentadas como sistemas de "neurônios interconectados, que podem computar valores de entradas", simulando o comportamento de redes neurais biológicas[16].

Os neurônios de uma RNA devem estar conectados entre si, são dispostos em camadas, e os neurônios de uma mesma camada normalmente se comportam da mesma maneira. As redes sem realimentação (feedforward) têm neurônios agrupados em camadas. O sinal percorre a rede em uma única direção, da entrada para a saída. Os neurônios da mesma camada não são conectados. [16]

Nas redes com realimentação ou recorrentes (recurrent), a saída de alguns neurônios alimentam neurônios da mesma camada (inclusive o próprio) ou de camadas anteriores. O sinal percorre a rede em duas direções, tem memória dinâmica e capacidade de representar estados em sistemas dinâmicos.

As redes neurais possuem nós ou unidades de processamento. Cada unidade possui ligações para outras unidades, nas quais recebem e enviam sinais. Cada unidade pode possuir memória local. Essas unidades são a simulação dos neurônios, recebendo e retransmitindo informações. Somam-se as entradas e se retorna uma saída, caso esta seja maior que o valor da soma.

##Uma rede neural pode ser usada para classificação ou regressão.
• Classificação booleana com saídas contínuas: – É comum ter uma única unidade de saída, com um valor acima de 0,5 interpretado com uma classe e um valor abaixo de 0,5 como a outra.
• Classificação em k categorias: – Podemos dividir o intervalo da única unidade de saída em k porções, embora seja mais comum ter k unidades de saída separadas.##

A propriedade mais importante das redes neurais é a habilidade de aprender acerca de seu ambiente e, com isso, melhorar o seu desempenho. Isto pode ser feito através de um processo iterativo de ajustes aplicados aos pesos sinápticos da rede, chamado de treinamento. O aprendizado ocorre quando a rede neural atinge uma solução generalizada para uma determinada classe de problemas.[16]

Todos os modelos de redes neurais possuem uma regra de treinamento, onde os pesos de suas conexões sinápticas são ajustados de acordo com os padrões apresentados, ou seja, a rede aprende através de exemplos provenientes de casos reais conhecidos. Deste modo, a rede neural extrai regras básicas a partr dos exemplos, onde é necessário que as regras sejam previamente conhecidas.
Dentro do paradigma de aprendizado supervisionado...

2.4 Indução por Máquinas de Vetor de Suporte

Uma máquina de vetores de suporte (do inglês: support vector machine (SVM)) é um conjunto de métodos de aprendizado supervisionado que analisam os dados e reconhecem padrões, usado para classificação e análise de regressão. O SVM padrão toma como entrada um conjunto de dados e prediz, para cada entrada dada, qual de duas possíveis classes a entrada faz parte, o que faz do SVM um classificador linear binário não probabilístico[17].

Dados um conjunto de exemplos de treinamento, cada um marcado como pertencente a uma de duas categorias, um algoritmo de treinamento do SVM constrói um modelo que atribui novos exemplos a uma categoria ou outra.

Um modelo SVM é uma representação de exemplos como pontos no espaço. Os novos exemplos são então mapeados no mesmo espaço e preditos como pertencentes a uma categoria baseados em qual o lado do espaço eles são colocados.[17]

O papel de uma SVM é encontrar uma linha de separação, chamada de **hiperplano** entre dados de duas classes. Essa linha busca maximizar a distância entre os pontos mais próximos em relação a cada uma das classes:

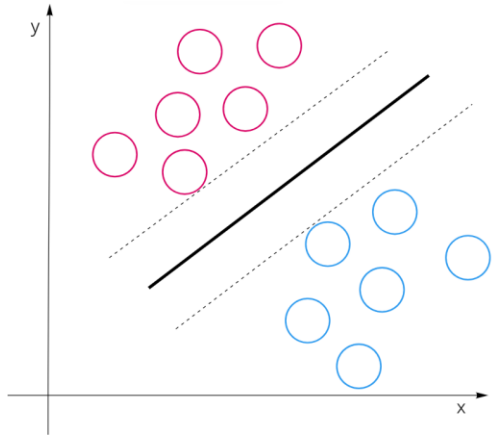

Figura 3: Hiperplano (linha central)

Essa distância entre o hiperplano e o primeiro ponto de cada classe costuma ser chamada de margem. Os pontos mais próximos ao hiperplano (representados pelas retas pontilhadas na figura) são chamados de **vetores de suporte,** dando nome ao algoritmo, pois é a partir deles que o modelo será desenvolvido matematicamente, treinado e otimizado.

O SVM coloca em primeiro lugar a classificação das classes, definindo assim cada ponto pertencente a cada uma das classes, e em seguida maximiza a margem de maneira iterativa, minimizando erros. O hiperplano com margem máxima é chamado de hiperplano ótimo, que será o objeto de busca do treinamento do classificador.
Em um conjunto de dados bidimensional, como o da figura, o hiperplano é uma reta. Em conjuntos tridimensionais, o hiperplano de fato é um plano.

O SVM funciona muito bem em domínios complicados, em que existe uma clara margem de separação, entretanto, é um modelo considerado como “caixa preta”, uma vez que sua interpretação e visualização muito complexa, principalmente em problemas de maior dimensionalidade.

É um algoritmo lento quando comparado a outros algoritmos de classificação pois não funciona bem em conjuntos de dados muito grandes, pois exige inversão de matriz aumentando a complexidade computacional, além de não funcionar bem em conjunto de dados com grande quantidade de ruídos.

Sua utilização é notória em reconhecimento de imagem, detecção facial, categorização de textos, reconhecimento de escrita manual e até mesmo tem aplicações na área biomética.

2.5 Indução pelo KNN (K-Vizinhos Mais Próximos)

Por último, não menos importante, tem-se o KNN (K-Nearest Neighbors, em tradução: K-Vizinhos Mais Próximos). É um método baseado em distâncias que considera a proximidade entre pontos de dados na realização de predições. É um dos algoritmos mais simples de supervisão de aprendizado de máquina. Geralmente é utilizado para tarefas de classificação, consistindo em classificar um dado com base na classificação de seus dados vizinhos.

O KNN armazena todos os casos possíveis e classifica novos casos com base na métrica de similaridade. O valor K é um parâmetro que referencia o número de vizinhos mais próximos para incluir no processo chamado de "votação por maioria". Em outras palavras, por uma lógica bastante simples, o dado sendo analisado é classificado com base na maior representação dos K vizinhos mais próximos.

Esse método é conhecido como um "aprendizado preguiçoso" por não aprende uma função discriminativa dos dados de treinamento, ou seja, ele não tem uma fase de treinamento como os demais, ele apenas “memoriza” os dados e prediz os vizinhos mais próximos quando é requisitado. Além de lidar melhor com conjuntos pequenos.

Pela premissa, é necessário encontrar os K vizinhos mais próximos. Para isso, o algoritmo utiliza o cálculo da **Distância Euclidiana,** que fornece a distância entre dois pontos no plano de coordenadas **x**, **y**.

$$dist(d) = \sqrt{(x2 - x1)^2 + (y2 - y1)^2}$$

Fórmula 2: Distância Euclidiana entre dois pontos no plano

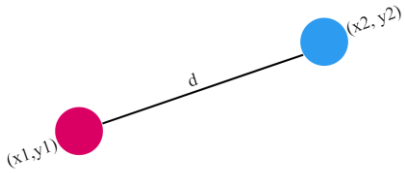

Figura 4: Representação no plano

Essa distância é calculada do ponto analisado em questão para todos os outros pontos do conjunto de dados.

Com isso, de maneira geral, o algoritmo trabalha da forma:

1) Escolhe-se um número K arbitrário (aconselhável que seja ímpar);

2) É feito o cálculo das distâncias entre duas linhas do conjunto de dados. Para dados em tabela ou em vetores, que é o caso aqui tratado, a distância euclidiana é considerada uma boa métrica. (Fórmula 2)
3) Obtendo os vizinhos mais próximos: Os vizinhos para um dado selecionado do conjunto de dados são as K instâncias (pontos de dados) mais próximas obtidas no cálculo da distância no passo anterior. Esse passo é realizado para todas as outras instâncias do conjunto de dados. Em seguida, ordena-se todas as distâncias, da menor para a maior, com base na distância do dado analisado e, por fim, seleciona-se apenas os K primeiros vizinhos de interesse;

4) Neste ponto predições sobre o dado analisado já podem ser feitas. Tratando-se de um algoritmo de classificação, neste passo será retornado a classe mais representada entre os k-vizinhos obtidos. Por exemplo, 4 (K=4) vizinhos são classificados entre *classe1* e *classe2*. Se *classe2* compõe a maioria entre os 4, democraticamente o algoritmo classifica o dado analisado como pertencente à *classe2*.

Exemplo 3: *Ação do algoritmo sobre um pequeno conjunto de dados:*

Peso	*Altura*	*Classe*
51	*167*	*Abaixo do Peso*
62	*182*	*Peso Normal*
69	*176*	*Peso Normal*
64	*173*	*Peso Normal*
56	*174*	*Abaixo do Peso*

Tabela 2: Conjunto de dados de dois atributos

Com base nesse conjunto, imagine deve-s-e classificar o set abaixo como normal ou abaixo do peso usando o KNN.

Peso	*Altura*	*Classe*
57	*170*	*?*

Tabela 3: Set de exemplo para o KNN

Para o dado set exemplo:
(x1, y1) = (57, 170)

E as distâncias (Fórmula 2) para os demais pontos do conjunto são:

Peso	*Altura*	*Classe*	*Dist. Euclidiana*
51	*167*	*Abaixo do Peso*	*6,7*
62	*182*	*Peso Normal*	*13*
69	*176*	*Peso Normal*	*13,4*
64	*173*	*Peso Normal*	*7,6*
56	*174*	*Abaixo do Peso*	*4,1*

Tabela 3: Distâncias entre o set da Tabela 3 e os demais pontos de dados do conjunto da Tabela 2.

Agora que as distâncias foram estabelecidas, o algoritmo avalia com o base no número de vizinhos próximos qual será a classificação do dado set de exemplo. Para isso, ele armazena as K menores distâncias e avalia qual classe é a maioria que corresponde a elas.

*Para o exemplo acima, quais são os vizinhos mais próximos para **K = 3**?*

*K = 3 correspondem à distâncias: **4,1**; **6,7** e **7,6.** Que em sua maioria correspondem à classe Abaixo do peso. Consequentemente, o KNN irá classificar o set de exemplo como **Abaixo do peso** também.*

```
1  inicialização:
2      Preparar conjunto de dados de entrada e saída
3      Informar o valor de k;
4  para cada nova amostra faça
5      Calcular distância para todas as amostras
6      Determinar o conjunto das k's distâncias mais próximas
7      O rótulo com mais representantes no conjunto dos k's
8      vizinhos será o escolhido
9  fim para
10 retornar: conjunto de rótulos de classificação
```

Uma das vantagens já citadas, é que o KNN não inclui período de treinamento, pois os próprios dados são um modelo que será a referência para futuras previsões e por isso é muito eficiente em termos de improvisação para uma modelagem aleatória dos dados disponíveis. Isso incorre no fato de que esse algoritmo é de fácil implementação, pois a única coisa a ser calculada é a distância entre diferentes pontos com base em dados de diferentes atributos e essa distância pode ser facilmente calculada pela distância euclidiana.

Entretanto, é muito sensível a dados ruidosos e ausentes e não funciona bem com grandes conjuntos de dados**,** pois calcular distâncias entre cada instância de dados seria muito custoso do ponto de vista computacional. Imagine que é necessário trabalhar com um conjunto de dados muito grande. Além de ser umá decisão armazenar uma grande quantidade de dados, é também algo caro do ponto de vista computacional manter o cálculo e a classificação de todos os valores.

Ademais, esse algoritmo também não funciona bem com alta dimensionalidade, pois isso complicará o processo de cálculo de distância para calcular a distância para cada dimensão. Embora a distância euclidiana seja o método mais comum usado e ensinado, nem sempre é a decisão ideal para todos os conjuntos de dados, há outras métricas conhecidas como a Distância de Manhattam, entre outras.

3 Bases de dados

Quatro conjuntos de dados (datasets) foram selecionados dos repositórios gratuitos da *UCI Machine Learning Reposi-*

tory[20] e da *Kaggle*[21] para realizar o comparativo dos algoritmos de classificação supracitados. Dos quatro conjuntos, três representam uma base relacionada ao campo da saúde/medicina e um do campo da natureza. São eles:

- **Zoo Animal Classification:** Este é um pequeno conjunto de dados que consiste em descrever animais em um zoológico e classificá-los com base em suas características[22]. Há 17 atributos e 101 instâncias. Os animais podem ser classificados como **Mamífero**, **Pássaro**, **Réptil**, **Peixe**, **Anfíbio**, **Inseto** ou **Invertebrado**

- **Heart Disease:** O objetivo é verificar se um paciente possui alguma doença cardíaca. Pode ser classificado como **sem doença** ou **com doença.** É um conjunto de dados de 1988 que consiste em quatro bancos de dados de Cleveland, Hungria, Suíça e Long Beach V[23]. Conta com 14 atributos e 303 instâncias.

- **Breast Cancer Wisconsin Diagnosis**: Conjuntos utilizados para diagnóstico de câncer de mama em Wisconsin (EUA). Foram obtidos dos bancos de dados dos hospitais da Universidade de Wisconsin, em 1995[24]. Esse conjunto possui 10 atributos e 699 instâncias. O diagnóstico aponta se o câncer é **Benigno** ou **Maligno**.

- **Fetal Health Classification:** Conjuntos obtidos de Cardiotocogramas (CTGs), que são utilizados para avaliar a saúde fetal, permitindo a atuação de profissionais de saúde na prevenção da mortalidade infantil e materna[25]. Esse conjunto possui 21 atributos e 2.126 registros (instâncias) que foram classificados por três obstetricistas especialistas em três classes: **Normal**, **Suspeito** e **Patológico**.

Os exemplos desses conjuntos de dados são rotulados como (xi, yi), onde xi representa a entrada e yi representa a saída (classe) esperada. Os conjuntos selecionados não possuem valores faltantes ou dados que dificultem a análise (ruidosos).

A partir desses conjuntos, os classificadores deverão ser capazes de predizer classes precisamente de cada um. Todo o processo de fornecer os exemplos e gerar a classificação, parte da subdivisão de cada conjunto de dados em um conjunto de **treinamento** e um conjunto de **testes**. Foi estabelecido que todos os modelos trabalharão, de início, com a mesma divisão:

- 80% do conjunto é utilizado para treinamento;
- 20% do conjunto é utilizado para testes.

4 Análise

4.1 Validação Cruzada

Agora que os algoritmos são capazes de realizar predições, é necessário analisar sua performance. Para otimizar esse feito, uma das técnicas utilizadas é o método da Validação Cruzada, uma técnica estatística que consiste em dividir aleatoriamente o conjunto de dados em k-grupos (chamados *folds*) de tamanho aproximadamente iguais.

Para reduzir a variabilidade, várias rodadas de validação cruzada são realizadas com diferentes subconjuntos dos mesmos dados. Combinando os resultados de validação dessas rodadas vamos obtem-se uma estimativa precisa do desempenho preditivo de cada modelo.

O primeiro grupo é mantido para testes e os k-1 grupos restantes serão utilizados para treinamento[26].

Exemplo 5:

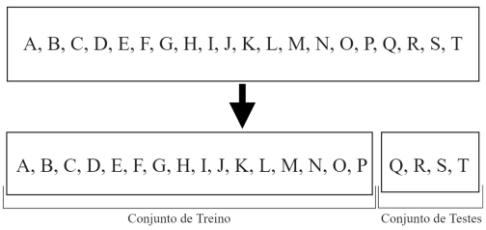

Figura 5: Particionamento de um conjunto de 20 pontos de dados em 2 subconjunto: um para treino e o outro para testes.

Esse processo é repetido k vezes e a cada vez um grupo/fold diferente é usado para validação. A medida de desempenho relatada pela validação cruzada será, então, a média dos valores calculados nas iterações[26]. Tipicamente o valor de k é 5 ou 10. No estudo utilizou-se k = 10 pois o tempo computacional é otimizado para os conjuntos de dados relativamente pequenos aqui usados e atribui-se uma característica importante que é o chamado bias reduzido, que trata-se de o algoritmo dar preferência de uma hipótese sobre outra. Todos os dados analisados são testados apenas uma vez e são usados no treinamento k-1 vezes. A variância da estimativa resultante é reduzida à medida de k aumenta.

4.2 Avaliação Estatística

4.2.1 Acurácia e Precisão de classificação

A acurácia se faz importante pois é preciso confiar nos resultados. Afinal, eles são a base de uma decisão final para alguma aplicação. Essa métrica é dada pelo cálculo da fração dos positivos verdadeiros e dos falsos positivos sobre o número total de atributos atribuídos.

Vem da forma:

$$Acc = \frac{\text{Número de predições corretas}}{\text{Total de predições feitas}}$$

Fórmula 4: Acurácia

A acurácia é a proximidade de um resultado com seu valor de referência real. Dessa forma, quanto maior a acurácia, que vai de 0 a 100%, mais próximo da referência ou valor real é o resultado encontrado. O conceito de acuracidade é muitas vezes confundido com a ideia de precisão, mas seu significado é um pouco mais complexo. A precisão é o grau de variação resultante de um conjunto de medições realizadas, como visto. Com isso, quanto mais preciso, menor é a variabilidade entre os valores encontrados[27].

Outro meio de avaliar o desempenho dos modelos é por meio da precisão, que representa a porcentagem de casos classificados corretamente. Essencialmente, permite dizer como o algoritmo performa em termos de falsos positivos. Matematicamente, para um classificador binário, é representado como precisão:

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

Fórmula 3: Precisão de um classificador

onde:

- Positivo verdadeiro, ou TP, são casos com rótulos positivos que foram classificados corretamente como positivos.
- Verdadeiro negativo, ou TN, são casos com rótulos negativos que foram classificados corretamente como negativos.
- Falso positivo, ou FP, são casos com rótulos negativos que foram classificados incorretamente como positivos.
- Falso negativo, ou FN, são casos com rótulos positivos que foram classificados incorretamente como negativos.

4 Resultados

Segue um comparativo sobre a acurácia obtida pelos algoritmos ao serem induzidos sobre os quatro conjuntos de dados.

Os algoritmos foram executados três vezes com cada conjunto devido à variâncias da acurácia. Uma média foi feita sobre as três execuções para definir a acurácia final:

- **Breast Cancer Wisconsin Diagnosis**

ID3	*GaussianNB*	*RNA*	*SVM*	*KNN*
95%	*95%*	*95%*	*96%*	*61%*

- **Fetal Health Classification**

ID3	*GaussianNB*	*RNA*	*SVM*	*KNN*
93%	*71%*	*92%*	*90%*	*68%*

- **Zoo Animal Classification**

ID3	*GaussianNB*	*RNA*	*SVM*	*KNN*
100%	*90%*	*95%*	*100%*	*28%*

- **Heart Disease**

ID3	*GaussianNB*	*RNA*	*SVM*	*KNN*
73%	*85%*	*68%*	*90%*	*54%*

4 Conclusões

Nota-se que o SVM oferece uma precisão muito alta em comparação com outros classificadores. Isso se dá por ele ser conhecido por seu truque de kernel para lidar com espaços de entrada não lineares. Eles também usam menos memória porque usam um subconjunto de pontos de treinamento na fase de decisão. O SVM funciona bem com uma clara margem de separação e com grande espaço dimensional.

Os algoritmo ID3 (Iterative Dichotomiser3 de Árvores de Decisão) vem atrás do SVM com a melhor acurácia. Notou-se que o MLP RNA (algoritmo Multilayer Perceptron de Redes Neurais) e o GaussianNB (Naive-Bayes Gaussiano) possuem uma taxa de acertos parecidas, ficando em terceiro. Em contrapartida, o KNN implementado possui uma acurácia muito inferior aos demais. Esse fato talvez tenha se dado pelo fato de não possuir uma validação cruzada robusta.

Referências

[1] https://www.britannica.com/technology/machine-learning «Definição: machine learning». Enciclopedia Britannica (em inglês)

[2] Simon, Phil (2013). Too Big to Ignore: The Business Case for Big Data. [S.l.]: Wiley. 89 páginas. ISBN 978-1-118-63817-0

https://books.google.com.br/books?id=Dn-Gdoh66sgC&pg=PA89&redir_esc=y#v=onepage&q&f=false

[4] https://www.cin.ufpe.br/~pacm/SI/ArvoreDecisaoIndutiva.pdf

[5] http://www.eletrica.ufpr.br/ufpr2/professor/36/TE808/4-ArvoresdeDecisao-AM.pdf

[6] https://medium.com/@gabriel.stankevix/algoritmos-de-aprendizado-de-maquina-uma-vis%C3%A3o-sobre-iot-1-75a6e8c45c84

[7] https://www.organicadigital.com/blog/algoritmo-de-classificacao-naive-bayes/

[8] Mitchell T. Machine Learning. WCB McGraw–

Hill, 1997. Capítulo 6

[9] https://www.datageeks.com.br/naive-bayes/

[10]http://www.eletrica.ufpr.br/ufpr2/professor/36/TE808/5-NaiveBayes-AM.pdf

(tem um exemplo bom na pagina 23)

[11] https://scikit-learn.org/stable/modules/naive_bayes.html

Observa-se que a noção por trás de uma árvore de decisão é intuitiva e de simples compreensão, não sendo necessário um conhecimento estatístico aprofundado para sua interpretação.

[12] https://www.analyticsvidhya.com/blog/2021/01/gaussian-naive-bayes-with-hyperpameter-tuning/

[13] https://sites.icmc.usp.br/andre/research/neural/

[14] slides da disciplina de inteligência artificial

[15] https://matheusfacure.github.io/2017/03/05/ann-intro/

[16] https://pt.wikipedia.org/wiki/Rede_neural_artificial

[17] https://pt.wikipedia.org/wiki/M%C3%A1quina_de_vetores_de_suporte~

[18] https://www.cin.ufpe.br/~tg/2010-2/gmoj.pdf

[19] https://www.datacamp.com/community/tutorials/svm-classification-scikit-learn-python

[20] http://archive.ics.uci.edu/ml/index.php

[21] https://www.kaggle.com/

[22] https://archive.ics.uci.edu/ml/datasets/zoo

[23] https://www.kaggle.com/ronitf/heart-disease-uci

[24] http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29

[25] https://www.kaggle.com/andrewmvd/fetal-health-classification

[26] https://scikit-learn.org/stable/modules/cross_validation.html