UNINTER – CENTRO UNIVERSITÁRIO INTERNACIONAL

BACHARELADO EM MATEMÁTICA

COLODETTI, Pedro Vinicius Baptista. Bacharelando em matemática no Centro Universitário Internacional Uninter

DE LIMA, José Airton Gonçalves. Professor/Orientador no Centro Universitário Internacional Uninter

MATEMÁTICA APLICADA À INTELIGÊNCIA ARTIFICIAL: A BASE FUNDAMENTAL DO *MACHINE LEARNING*

MATEMÁTICA APLICADA À INTELIGÊNCIA ARTIFICIAL: A BASE FUNDAMENTAL DO *MACHINE LEARNING*

COLODETTI, Pedro Vinicius Baptista. Bacharelando em matemática no Centro Universitário Internacional Uninter
DE LIMA, José Airton Gonçalves. Professor/Orientador no Centro Universitário Internacional Uninter

RESUMO

Este artigo aborda a matemática utilizada nos principais modelos de *Machine Learning* utilizados hoje. Mostrando a importância da matemática neste segmento e a relevância que a matemática tem ganhado nos últimos anos devido ao crescimento exponencial dos sistemas inteligentes. O conceito de inteligência artificial e até mesmo de redes neurais artificiais, não é novo. O problema até então era o custo computacional, não tínhamos tecnologia suficientemente avançada para rodarmos redes neurais com um mínimo de complexidade, e quando nossa tecnologia avançou, nos anos 80 fomos capazes de criar sistemas inteligentes e até mesmo redes neurais, porém, de forma muito limitada, tínhamos computadores, mas com a capacidade de processamento extremamente limitada. Nas últimas duas décadas, com o surgimento de computadores de desempenho muito elevado, fomos capazes de projetar redes neurais cada vez mais complexas. Este artigo abordará modelos iniciais de inteligência artificial composta por uma rede neural artificial simples, de uma camada (*layer*) onde suas funções matemáticas de cálculo para criar o modelo em questão, tem o formato linear. Por fim, abordarei apenas os conceitos básicos de redes neurais artificiais de múltiplas camadas. Estes modelos por serem muito complexos e despenderem de uma análise matemática mais profunda e complexa, não serão abordados neste artigo, apenas comentados de forma geral.

Palavras-chave: *Machine learning*. Redes Neurais. *Deep Learning*.

1. Introdução

No fim do século 20, tivemos vários filmes de ficção científica campeões de bilheteria. Um deles, foi o filme 'Exterminador do Futuro', onde um sistema inteligente se torna auto consciente e inicia ataques massificados contra os seres humanos, os levando à beira da extinção. Naquela época, esta situação era impensável, já que sistemas inteligentes, nem de longe eram uma realidade. Trinta e sete anos depois, nós temos

sistemas inteligentes em relógios, televisores, geladeiras, carros autônomos e podemos conectar todos eles ao mesmo dispositivo. Mas como isso é possível? Como nossa tecnologia evoluiu tanto, em tão pouco tempo? Como um sistema inteligente é desenvolvido?

1.1 Objetivo Geral

Demonstrar de forma clara e objetiva como um modelo de *machine learning* é desenvolvido, apontando os conceitos matemáticos utilizados para tal, inclusive abordando de forma simples, alguns modelos de *machine learning* e seus funcionamentos. Apresentar de forma muito resumida o conceito de *deep learning* e redes neurais aritificiais.

1.2 Objetivos Específicos

- Abordar conceitos iniciais de Inteligência Artificial, evoluindo gradativamente para uma subárea específica, o *Machine Learning*, onde serão abordados conceitos teóricos e matemáticos um pouco mais detalhados.
- Demonstrar os modelos de *machine learning* mais usados hoje por grandes empresas.
- Conceituar de forma muito sucinta os conceitos básicos de *deep learning* e redes neurais artificiais, sem apresentar modelos, apenas sua forma geral.

1.3 Justificativa

Os sistemas inteligentes estão presentes em praticamente todos os setores (econômico, social, transportes, etc.) e cada vez mais, fazem parte das nossas vidas. A área de Ciência de Dados a qual pertence a Inteligência Artificial, é sem dúvidas, uma das que mais cresce e recebe investimos hoje. Entender como tudo isso funciona, está se tornando uma obrigação, dado o cenário profissional atual.

2. Metodologia

Foi realizado pesquisa documental sobre a história da inteligência artificial, bem como também, pesquisa documental de conceitos pertinentes a este artigo, já que se fez necessário uma delimitação muito grande dos conceitos abordados considerando que este

assunto é incrivelmente extenso e extrapolaria o número máximo de laudas deste artigo. Inclui-se nos conceitos pertinentes (acima citado), conceitos teóricos da matemática, conceitos de redes neurais artificias, *Deep Learning,* neurônio matemático e apresentei na forma de imagens, representações dos modelos de redes neurais artificiais, bem como também, algoritmos genéricos para se criar tais redes. A busca foi por documentos de considerável valor teórico. A ordenação do material coletado foi feita a partir de leitura minuciosa para elencar a melhor sequência dos conteúdos, respeitando a cronologia dos fatos. Os documentos foram devidamente citados.

3. **Inteligência Artificial**

A própria definição do termo I.A (inteligência artificial) é algo complexo. Existem muitas definições, algumas muito próximas e outras, que se distanciam de um senso comum. Quando se fala de I.A, logo nos vêm à cabeça, um robô, ou uma máquina humanoide inteligente, algo como um ser humano artificial, mas, na realidade, o conceito teórico e prático de I.A, abrangem um leque de possibilidades muito vasto. Podemos definir I.A como qualquer sistema que seja capaz de realizar tarefas de forma autônoma e que seja capaz de aprender novas tarefas, valendo-se (mas não apenas) de técnicas como *Machine Learning* e *Deep Learning*. Esta definição, não caracteriza uma I.A por completo, pois sua definição, é algo complexo e que evolui a cada dia. De acordo com **FIA** (2021) "Inteligência artificial é a capacidade de dispositivos eletrônicos de funcionar de maneira que lembra o pensamento humano".

A I.A. tem ganhado cada vez mais destaque, principalmente nos setores de transportes, onde empresas como a Tesla, desenvolveram veículos autônomos. O veículo em questão, dirige autonomamente, sem a necessidade que um ser humano esteja na direção. Tudo é feito pelo app ou pelo gps do veículo.

O ano de 2021, sem dúvida foi um marco para o turismo espacial. Nunca na história, ocorreram tantos lançamentos de foguetes e veículos espaciais. Em 2021 nós realizamos 144 lançamentos, dos quais 133 foram bem sucedidos[1]. A *Blue Origin* completou seu primeiro voo suborbital com tripulantes na manhã desta terça-feira, 20 de julho, data que também marca o aniversário da chegada da humanidade na Lua. A viagem ocorreu na nave

[1] ARBULU, Rafael. Em 2021, nós lançamos mais foguetes que qualquer outro ano na história. **Olhardigital**, 2022. Disponível em: <https://olhardigital.com.br/2022/01/03/ciencia-e-espaco/em-2021-nos-lancamos-mais-foguetes-que-qualquer-outro-ano-na-historia/>. Acesso em: 05 de jan. de 2022.

New Shepard e um dos integrantes da viagem foi Jeff Bezos, o fundador da Amazon e da companhia de foguetes[2]. A pilotagem da espaçonave foi feita de forma autônoma.

Existe uma diferença considerável entre a programação tradicional e a programação do *machine learning* ou do *deep learning.* Na programação tradicional, o programador insere valores de entrada e em seguida, estipula comandos para realizar uma tarefa qualquer, inclusive explicitando também como estas tarefas devem ser executadas, obtendo assim um resultado na saída deste algoritmo. No *machine learning*, o programador insere os dados de entrada e insere também o resultado desejado. O computador com as informações de entrada e de saída, cria o algoritmo que com aqueles dados de entrada, resultarão na saída esperada.

A área da programação evoluiu muito rápido, e, inevitavelmente, novas áreas a ela relacionadas apareceram. É comum associarmos a IA com a área da ciência de dados. Elas são muito próximas, mas são áreas diferentes. A confusão está no fato de que a ciência de dados, utiliza várias técnicas estatísticas que fazem parte do arcabouço da IA, mas são áreas diferentes.

A IA subdivide-se em várias áreas, as duas principais são: *Machine Learning (ML)* e *Natural Language processing (NLP)*. Por questão de escopo, este artigo não adentrará nos mares extensos do NLP onde nos ateremos ao *machine learning* e abordaremos também uma subárea do ML, o *deep learning*.

4. Machine Learning

O ML é uma ferramenta da IA onde ocorre a análise de dados para o desenvolvimento automático de modelos analíticos. Não é receita de bolo, o modelo criado não serve para qualquer situação, ele varia de acordo com suas especificações do projeto. Um modelo de ML, necessita basicamente de duas bases de dados. A primeira base é chamada de base de treino e a segunda é chamada de base de teste. A base de treino é usada pelo algoritmo para gerar aprendizado, extraindo padrões e regras comuns dos

[2] MOGNON, Mateus. Blue Origin completa primeiro voo tripulado com Jeff Bezos. **Tecmundo**, 2021. Disponível em: <https://www.tecmundo.com.br/ciencia/221402-blue-origin-completa-primeiro-voo-tripulado-jeff-bezos.htm>. Acesso em: 05 de jan. de 2022.

elementos desta base. A base de teste serve para testar o algoritmo, verificando se a classificação realizada dos elementos, está correta ou não[3].

Para que o aprendizado e a classificação atinjam níveis de assertividade muito elevados, os dados das duas bases de treino necessitam estar rotulados, ou seja, se faz necessário especificar para o sistema o que aquele dado representa. A rotulação da base de dados é intrínseca à homogeneidade dos dados, tanto estatisticamente quanto a nível de abrangência das soluções em questão. Um fator que pode se tornar um problema é, como dividir corretamente a base de dados sem que haja excesso de dados na base de treino, o que resultaria em uma carência na base de teste, ou o contrário, carência na base de treino e excesso de dados na base de teste. Não existe uma resposta para esta pergunta, tudo vai depender do tamanho da base de dados, do tipo do dado, do projeto, etc.

O volume da base de dados é muito importante. Quanto maior for a base de dados, melhor e mais eficiente será a IA desenvolvida. É possível aplicarmos diferentes técnicas às bases de dados com o intuito de melhorar ainda mais a extração do conhecimento pela IA. Duas técnicas muito conhecidas são: ***Hold-Out*** e a ***Cross-Validation***. Na técnica de *hold-out*, simplesmente divide-se a base de dados em duas partes, a base de treino e a base de teste, como vimos anteriormente. Na técnica *cross-validation*, dividimos a base de dados em 'k' partes onde uma parte é agrupada na base de treino e a outra parte é agrupada na base de teste. O algoritmo testa o *score* para então refazer o processo mudando os agrupamentos. O agrupamento que atingir o maior *score*, será a configuração a ser utilizada na etapa de aprendizado.

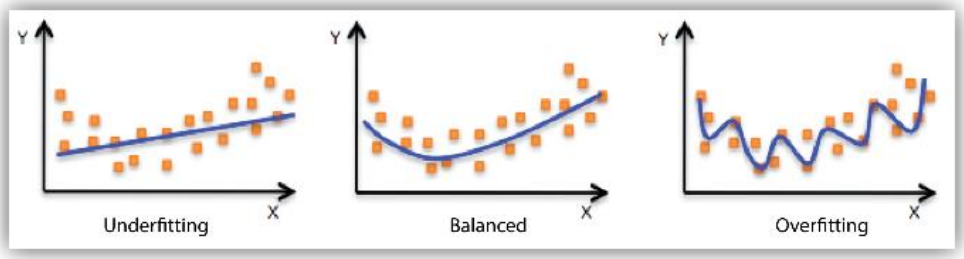
Quando o modelo classifica corretamente novas instâncias (novos dados), dizemos que o modelo generaliza bem. A **generalização** diz respeito ao nível de aprendizado de um modelo. Os modelos não possuem 100% de acurácia, chegam muito perto da perfeição, mas nunca atingem os 100%, portanto, o fator erro deve ser considerado. Um conceito básico que envolve o erro, é o conceito de **bias.** Bias ou viés, é um erro comum atribuído ao modelo, onde ocorre uma diferença entre a previsão realizada pelo modelo e o valor

[3] ALLIBHAI, Eijaz. Hold-out vs. Cross-validation in Machine Learning | by Eijaz Allibhai | Medium. Medium, 2018. Disponível em: <https://medium.com/@eijaz/holdout-vs-cross-validation-in-machine-learning-7637112d3f8f>. Acesso em: 15 de jan. de 2022.

correto daquela classificação, ou seja, o modelo classifica erroneamente o objeto em questão.

Quando o modelo apresenta falha na base de treino e consequentemente não consegue aprender o suficiente para realizar uma classificação, a este erro damos o nome ***underfitting***. O *underfitting* resulta de um alto bias. Já o erro inverso, classificamos como ***overfitting***. O *overfitting* pode ser definido como um modelo que extraiu demais as características na etapa do treino, onde a extração foi tão profunda que o modelo só é capaz de classificar os dados de treino, qualquer dado fora da amostra de treino, o sistema mostra-se incapaz de classificar. Este erro é associado à alta **variância**. A variância é um outro tipo de erro que deve ser levado em consideração na criação de um modelo de ML. O erro de variância está relacionado à sensibilidade do modelo em classificar outras bases de dados, diferentes da base de treino. A seguir, podemos ver um gráfico que mostra claramente um modelo que apresenta erro de *underfitting*, outro modelo que apresenta erro de *overfitting*, e o que seria um modelo ideal.

Figura 1 – Gráfico de tipos de erros



Fonte: Página da Amazon sobre *Machine Learning*[4]

O aprendizado dos modelos do ML não segue uma regra única, existem diferentes tipos de aprendizado e sua escolha se dará pela necessidade do projeto em questão. Cada tipo tem suas vantagens e desvantagens. Também, por motivo de escopo do artigo, abordarei aqui apenas os dois principais tipos de aprendizagem, o **supervisionado** e o **não supervisionado**.

[4] Ajuste do modelo: Subajuste versus sobreajuste - Amazon Machine Learning. **Amazon**. Disponível em: <https://docs.aws.amazon.com/pt_br/machine-learning/latest/dg/model-fit-underfitting-vs-overfitting.html>. Acesso em 06 de jan. de 2022.

No aprendizado supervisionado, ocorre interferência direta do programador. O aprendizado supervisionado é uma técnica de *machine learning* que usa base de dados rotuladas (IBM, 2021, tradução nossa)[5] que serão inseridos na entrada do algoritmo e é necessário também que seja informado ao algoritmo, qual o resultado esperado, de acordo com os dados de entrada. O sistema encontrará padrões nos dados que são específicos. Desta forma, o sistema construirá um caminho que liga os dados rotulados de entrada com os resultados desejados. Para exemplificar, podemos criar uma IA para identificar cachorros em fotos, para tal, é necessário um conjunto suficientemente grande de fotos de cachorros para que o sistema encontre padrões específicos nas fotos e assim, possa classificar uma foto nova que seja usada em análise, se há nela um cachorro ou não. Diversas técnicas são usadas como filtros nas fotos para que haja uma maior diversidade possível para base de treino. Não trataremos desta questão neste artigo.

Ainda no aprendizado supervisionado, dependendo do funcionamento da IA, podemos relacionar a dois tipos, **classificação** e **regressão**. O tipo classificação, é definido por um sistema que sua finalidade é de analisar o objeto em questão e o categorizar. O tipo regressão, o objetivo é de realizar uma previsão, ou seja, prever algo.

No aprendizado não supervisionado, não se conhece o resultado, o programador cria a base de dados não rotulados e o sistema faz todo o restante. Este tipo de aprendizado é usado quando dispomos de poucos dados sobre o problema em questão. O modelo tenta deduzir a estrutura do algoritmo a partir dos dados de entrada. Um bom exemplo deste tipo de sistema, se dá quando precisamos definir um mercado para um produto novo.

O tipo de aprendizado dependerá da base de dados. Caso a base de dados esteja pré-classificada, usa-se o aprendizado supervisionado, se não, usa-se o não supervisionado. Certamente há outros tipos de aprendizado, porém, para fins de escopo deste artigo, os dois antes citados, são suficientes.

Podemos citar como um dos pilares do *machine learning* a identificação de padrões relevantes que podem ser utilizados na tomada de decisão ou para classificar ou identificar

[5] DELUA, Julianna. Supervised vs. Unsupervised Learning: What's the Difference? **IBM**, 2021. Disponível em: <https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning>. Acesso em: 25 de jan. de 2022.

algo. Um modelo muito usado para identificação destes padrões é a técnica de **regras de associação**.

4.1 Regras de Associação

O objetivo principal da técnica regra de associação é de identificar similaridades, semelhanças, ou seja, padrões frequentes entre os elementos de uma base de dados. Ressalto aqui que quando me refiro à base de dados, pode ser tanto um conjunto de elementos ou pode ser também um banco de dados. Este tipo de modelo é utilizado com muita frequência em sistemas de recomendação, porém, são inúmeras as aplicações deste modelo[6].

Ponderemos aqui a seguinte questão: A regra de associação é uma técnica que visa encontrar padrões em uma determinada base de dados para então, de forma automatizada, criar um modelo de IA. Mas o que de fato é importante? Como o algoritmo define os padrões que tem maior relevância para o modelo? Como a técnica define dentre todos os padrões extraídos da base de dados, quais são importantes e quais são irrelevantes?

Para respondermos às perguntas anteriores, se faz necessário a explanação de alguns conceitos, o primeiro deles é o conceito de **hiperparâmetro**. Hiperparâmetro em *machine learning* é um parâmetro no qual seu valor é usado para definir as regras no processo de aprendizagem do sistema[7]. As respostas para nossas perguntas estão justamente neste conceito. A técnica de regras de associação tem como pilar fundamental na abstração de padrões relevantes para o modelo, dois hiperparâmetros, o **suporte** e a **confiança**.

O **suporte** é definido como a quantidade de eventos entre pares de elementos em uma base de dados. Este hiperparâmetro calcula percentuais de eventos similares entre pares de elementos e é calculado da seguinte forma: $sup(A \rightarrow B) = \frac{\text{Número de eventos entre A e B}}{\text{Número total de eventos}}$. Uma outra forma de denotar o suporte é por meio da probabilidade onde $sup\ (A \rightarrow B) = P(A$

[6] Data Mining de Regras de Associação. **Devmedia**. Disponível em: <https://www.devmedia.com.br/data-mining-de-regras-de-associacao/6941>. Acesso em: 12 de jan. de 2022.

[7] HYPERPARAMETER (MACHINE LEARNING). In: WIKIPEDIA, a enciclopédia livre. Flórida: Wikipedia Fundation, 2022. Disponível em: <https://en.wikipedia.org/wiki/Hyperparameter_(machine_learning)>. Acesso em: 12 de jan. de 2022.

U B). Veja que a técnica verifica a quantidade de eventos que ocorreram e eventos que ocorreram com alta frequência, muito provavelmente possui uma relevância grande.

Enquanto que o **suporte** mede a relevância da regra, ou seja, um suporte alto implica em alta relevância para a regra, a **confiança** representa dentre os eventos presentes nos itens de 'A', qual a porcentagem de eventos também presentes nos itens de 'B', ou seja, mede o grau certeza de quanto o 'A' implica em 'B'[8]. A **confiança** mede o grau de certeza da regra e é denotada da seguinte forma: $conf(A \rightarrow B) = \frac{\text{Número de eventos entre (AUB)}}{\text{Número total de eventos em A}}$.

Fica claro ante o exposto que como a técnica consegue calcular, mensurar numericamente a relevância e o grau de certeza das regras, o modelo logicamente descarta regras que apresentam um número baixo de suporte e confiança. Com os hiperparametros estipulados anteriormente, podemos construir as regras que nortearão o modelo. Uma das regras que podemos aplicar para o **suporte** é: se X & Y → Z ou { X U Y U Z} e para a **confiança**: ou { X U Y } também conter Z. O primeiro termo é conhecido como **antecedente** e o segundo termo é conhecido como **consequente**.

Existem muitos algoritmos de regras de associação, aqui será demonstrado o algoritmo *FP-Growth*. O algoritmo *Apriori* não será demonstrado neste artigo por não ser mais utilizado com frequência.

4.1.1 *FP-Growth*

O *fp-growth* é um método que gera padrões frequentes de itens sem a necessidade de gerar listas de candidatos, como ocorre no método *apriori*, o que torna o *fp-growth* muito mais eficiente, justamente por ter um custo computacional muito menor em relação ao método *apriori*[9].

O funcionamento do método *fp-growth* dá-se em duas fases, ou seja, o algoritmo percorre toda a base de dados duas vezes. Na primeira fase, o algoritmo calcula a frequência dos itens da base de dados e verifica-se também a frequência destes itens, ou seja, ele constrói a árvore de padrões frequentes (*fp-tree – frequent pattern tree*), que é

[8] Data Mining de Regras de Associação. **Devmedia**. Disponível em: <https://www.devmedia.com.br/data-mining-de-regras-de-associacao/6941>. Acesso em: 12 de jan. de 2022.
[9] Frequent Pattern Mining - Spark 3.2.0 Documentation. **Apache**. Disponível em: <https://spark.apache.org/docs/latest/ml-frequent-pattern-mining.html>. Acesso em: 13 de jan. de 2022.

uma estrutura de dados compactada em forma de árvore onde são referenciados os suportes para então o algoritmo ir para a segunda e última fase.

Na segunda fase, o algoritmo pega a *fp-tree*, ou seja, ele usa a estrutura criada na primeira fase para codificar (gerar) as transações sem a necessidade de criar conjuntos de candidatos de forma direta o que gera uma economia em custo computacional muito grande, suscitando assim, uma performance superior do algoritmo. Após as duas fases, os *itemsets* podem sem extraídos da *fp-tree*. Estes *itemsets* extraídos, são utilizados para a construção a regra a ser adotada pelo modelo.

Vejamos a seguir o modelo sendo criado na prática. Para tal, usaremos a IDE *Jupyter notebook* versão web que é gratuita acessando https://jupyter.org/. A linguagem de programação utilizada será o *python*.

Na linha 1, realizamos os *imports* da biblioteca pandas, e da biblioteca mlxtend importamos apenas a classe *TransactionEncoder*. Esta é uma classe de codificação para transações de dados que são organizados em listas. Na linha 2, criamos uma base de dados com os itens pré-estabelecidos. A linha 3 é iniciada com a criação da variável 'te' que recebe como retorno a função *TransactionEncoder()*. Em seguida, cria-se outra variável com o nome 'te_ary' onde basicamente pegam-se todas as listas e as transformam em uma única lista usando como referência o método *TransactionEncoder()*. Note que neste passo, o algoritmo calcula a frequência de cada item. Após, é criado uma terceira variável de nome 'df' que pega os dados processados e organiza em uma tabela.

Figura 2 – Primeira parte do algoritmo *FP-Growth*.

```
In [1]: import pandas as pd
        from mlxtend.preprocessing import TransactionEncoder

In [2]: dataset = [['Leite', 'Cebola', 'Ovos', 'Feijão', 'Milho', 'Iogurte'],
                   ['Coentro', 'Cebola', 'ovos', 'Feijão', 'Milho', 'Iogurte'],
                   ['Leite', 'Maçã', 'Feijão', 'Milho'],
                   ['Leite', 'Batata', 'Arroz', 'Feijão', 'Iogurte'],
                   ['Arroz', 'Cebola', 'Cebola', 'Feijão', 'Sorvete', 'Milho']]

In [3]: te = TransactionEncoder()
        te_ary = te.fit(dataset).transform(dataset)
        df = pd.DataFrame(te_ary, columns=te.columns_)
        df
```

Out[3]:

	Arroz	Batata	Cebola	Coentro	Feijão	Iogurte	Leite	Maçã	Milho	Ovos	Sorvete	ovos
0	False	False	True	False	True	True	True	False	True	True	False	False
1	False	False	True	True	True	True	False	False	True	False	False	True
2	False	False	False	False	True	False	True	True	True	False	False	False
3	True	True	False	False	True	True	True	False	False	False	False	False
4	True	False	True	False	True	False	False	False	True	False	True	False

Fonte: Autor, 2022.

Na linha 4, o algoritmo importa da classe mlxtend a função *fp-growth* e estipula como parâmetro de corte, um suporte mínimo de 0,6 e já retorna em forma de tabela, todos os itens com suporte igual ou superior a 0,6. Nesta linha são gerados os *itemsets*. Note que a primeira fase do algoritmo, engloba as linhas 1, 2 e 3 e a segunda fase do algoritmo engloba apenas a linha 4. Aqui temos o modelo de ML criado com as regras devidamente estipuladas. Vejamos a seguir, o retorno que o modelo produz de acordo com a base de dados estipulada.

Figura 3 – Segunda parte do algoritmo *FP-Growth*.

```
In [4]: from mlxtend.frequent_patterns import fpgrowth

        fpgrowth(df, min_support=0.6)
```

Out[4]:

	support	itemsets
0	1.0	(4)
1	0.8	(8)
2	0.6	(6)
3	0.6	(5)
4	0.6	(2)
5	0.8	(8, 4)
6	0.6	(4, 6)
7	0.6	(4, 5)
8	0.6	(8, 2)
9	0.6	(2, 4)
10	0.6	(8, 2, 4)

Fonte: Autor, 2022.

Nesta quinta e última linha, o algoritmo retorna de forma organizada e nos moldes de uma tabela, os dados devidamente organizados contendo os itens relevantes e os valores dos suportes respectivamente. Perceba o quão curto é o algoritmo, porém, todos os cálculos estatísticos por trás, são enormes. Foi utilizado um modelo padrão de *fp-growth* com uma base de dados muito pequena, apenas para ilustrar o modelo. Em uma situação do cotidiano, em uma empresa por exemplo, o número de itens facilmente chega na casa dos milhares e as vezes dos milhões, o que resulta em um processamento muito mais complexo.

Figura 4 - Resultado do algoritmo *FP-Growth*.

```
In [5]: fpgrowth(df, min_support=0.6, use_colnames=True)
```

Out[5]:

	support	itemsets
0	1.0	(Feijão)
1	0.8	(Milho)
2	0.6	(Leite)
3	0.6	(Iogurte)
4	0.6	(Cebola)
5	0.8	(Feijão, Milho)
6	0.6	(Feijão, Leite)
7	0.6	(Feijão, Iogurte)
8	0.6	(Milho, Cebola)
9	0.6	(Feijão, Cebola)
10	0.6	(Feijão, Milho, Cebola)

Fonte: Autor, 2022.

4.1.2 *Naïve Bayes*

A técnica de *Naïve Bayes* tem como espinha dorsal, a teoria das probabilidades[10]. É uma técnica que atua na classificação estatística tendo como direcionador o teorema de Bayes, ou seja, sua regra de atuação está calcada na probabilidade condicional. Esta técnica na prática, atua calculando as probabilidades de ocorrência de cada classe para cada valor de atributo (*feature*). Um fator importante deste modelo é que o algoritmo desconsidera correlações entre os atributos (*features*), ou seja, a probabilidade de ocorrência de uma variável não afeta outra variável. Podemos simplificar da seguinte forma: $P(X|YZ) = P(Z)$. Em alguns casos, uma variável acaba afetando outra, não sendo 100% independentes. Para resolver o problema, faz-se jus da teoria da independência condicional que se refere a um evento 'R' ser condicionalmente independente de um segundo evento 'B' dado um terceiro evento 'Y', se a distribuição de probabilidade de 'R' for independente de 'B', dado 'Y', ou seja, $P(R|B \cap Y) = P(R|B)$[11].

Este modelo exprime um bom desempenho, pois, os cálculos matemáticos não são complexos, o que resulta em um custo computacional baixo. Vejamos a seguir, quais

[10] TAMAIS, Ana Laura Moraes. Modelos de Predição | Naive Bayes | by Ana Laura Moraes Tamais | Turing Talks | Medium. **Medium**, 2019. Disponível em: <https://medium.com/turing-talks/turing-talks-16-modelo-de-predi%C3%A7%C3%A3o-naive-bayes-6a3e744e7986>. Acesso em: 15 de jan. de 2022.

[11] Independência condicional – Wikipédia, a enciclopédia livre. In: WIKIPEDIA, a enciclopédia livre. Flórida: Wikipedia Fundation, 2022. Disponível em: <https://pt.wikipedia.org/wiki/Independ%C3%AAncia_condicional>. Acesso em: 15 de janeiro de 2022.

cálculos o modelo considera na hora de efetuar a classificação. A fórmula matemática genérica para este modelo é dada por: $$P(y \mid x_1, \ldots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y)$$.

Parece complicado, mas não é, veja como a fórmula é simples e ao mesmo tempo bonita. O termo 'y' é a classe alvo que queremos descobrir, ou seja, é nesta classe que o algoritmo irá classificar o valor em questão. X_1, ..., X_n é um vetor de valores. Podemos traduzir este primeiro termo da seguinte forma: Qual a probabilidade de o evento ocorrer, dados que os atributos possuem determinados vetores de valores?

O *alfa* demonstra que os dois termos são proporcionais, ou seja, 1ºT = 2ºT. P(Y) aqui representa a probabilidade de o evento de forma geral ocorrer, ou seja, a probabilidade de ocorrência da classe. Por fim, vemos um somatório da probabilidade do atributo X_i considerando que àquela classe ocorreu. Quando o modelo finaliza os cálculos de probabilidade, o mesmo verifica qual base tem a maior probabilidade, ou seja, $P(Y|X_1, ..., X_n)$ máximo.

A técnica de *Naïve Bayes* pode ser aplicada em vários tipos de classificação, não restringindo-se apenas na binária. Uma desvantagem é que esta técnica pode não cobrir todas as soluções possíveis, ou seja, pode não conter todas as combinações possíveis de valores. Quando isso ocorre, as combinações que não foram analisadas, retornam como valor o zero, prejudicando assim, o método de classificação. Para evitar esse tipo de situação, utilizamos o método de correção de Laplace.

Há diferentes tipos de modelos *Naïve Bayes*. Será apenas mencionado cada um deles, pois, uma explicação mais detalhada resultaria em um prolongamento demasiadamente excessivo neste artigo. Vejamos o tipo e em seguida será demonstrado sua formula geral.

4.1.2.1 *Naïve Bayes Gaussiana*

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$.

4.1.2.2 *Naïve Bayes Multinomial*

$$\hat{\theta}_{yi} = \frac{N_{yi} + \alpha}{N_y + \alpha n}$$.

4.1.2.3 *Naïve Bayes Bernoulli*

$$P(x_i \mid y) = P(i \mid y)x_i + (1 - P(i \mid y))(1 - x_i)$$

Vejamos agora a implementação prática do algoritmo também utilizando o *Jupyter notebook* com a linguagem de programação *Python* e a biblioteca *Scikit Learn* que é uma das principais bibliotecas de *machine learning* em *python*.

Iniciamos importando as bibliotecas e métodos necessários. Na linha 2 o modelo lê o arquivo que contém a base de dados e na linha 3 a função retorna as 'n' linhas do *database*.

Figura 5 – Primeira parte do algoritmo *Naïve bayes*.

```
In [1]: import pandas as pd
        from sklearn.model_selection import cross_val_score

In [2]: df = pd.read_csv('Iris.csv')

In [3]: df.head()

Out[3]:
```

	sepal.length	sepal.width	petal.length	petal.width	variety
0	5.1	3.5	1.4	0.2	Setosa
1	4.9	3.0	1.4	0.2	Setosa
2	4.7	3.2	1.3	0.2	Setosa
3	4.6	3.1	1.5	0.2	Setosa
4	5.0	3.6	1.4	0.2	Setosa

Fonte: Autor, 2022.

Na linha 8, informamos ao algoritmo quais são as *features* e informamos também a posição de cada uma delas na base de dados. Neste caso, especificamos as características na coluna 'X' e a classe na coluna 'Y'. Na linha 9, importamos a biblioteca de *machine learning* e aplicamos aqui o modelo de *Naïve Bayes* Gaussiano, pois, os valores das *features* são valores contínuos. Repare que na última parte da linha 9, definimos como iremos particionar a base de dados, neste caso, é o tipo *Cross-Validation*.

Perceba que a acurácia do nosso modelo atingiu o patamar de 95.33%, ou seja, quando o modelo for classificar algum valor, ele terá uma assertividade de 95.33%.

Figura 6 – Resultado do algoritmo **Naïve** *Bayes*.

```
In [8]: x = df[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']]
        y = df['Species']

In [9]: from sklearn.naive_bayes import GaussianNB

        gnb = GaussianNB()
        cross_val_score(gnb, x, y, cv=10).mean()

Out[9]: 0.9533333333333334
```

Fonte: Autor, 2022.

Percebmos aqui, que o modelo de modo geral, fez operações com matrizes, já que o nosso *database* possui informações nas linhas e nas colunas. O modelo aplicou cálculos probabilísticos (teorema de Bayes e Independência Condional) na matriz (base de dados). Os cálculos se deram da seguinte forma, o modelo pegou todos os valores 'X_n' da classe 'Y_1' e efetuou os cálculos probabilísticos, depois, o modelo pegou os valores 'X_n' da classe 'Y_2' e repetiu os cálculos. O modelo efetua os cálculos de todos os 'X_n' em todos os 'Y_n' para então efetuar as classificações.

5. *Deep Learning:*

Classificamos o *deep learning* como uma subclasse do *machine learning*[12]. Alguns modelos de ML apresentam uma complexidade maior em relação a outros. O ML em sua base, é dotado por modelos cujos algoritmos de aprendizagem, apresentam em seu DNA, uma matemática mais simples, conforme vimos anteriormente. Porém, nem todas as situações conseguem ser resolvidas por estes modelos, fazendo-se necessário, uma complexidade matemática mais profunda, resultando em modelos de ML mais complexos.

O *deep learning* é uma subárea do *machine learning* cujos modelos de aprendizagem, apresentam uma matemática no mínimo complexa. Mas, para conferir complexidade ao modelo, se faz necessário a utilização de outra tecnologia, as **Redes Neurais Artificiais**. Este

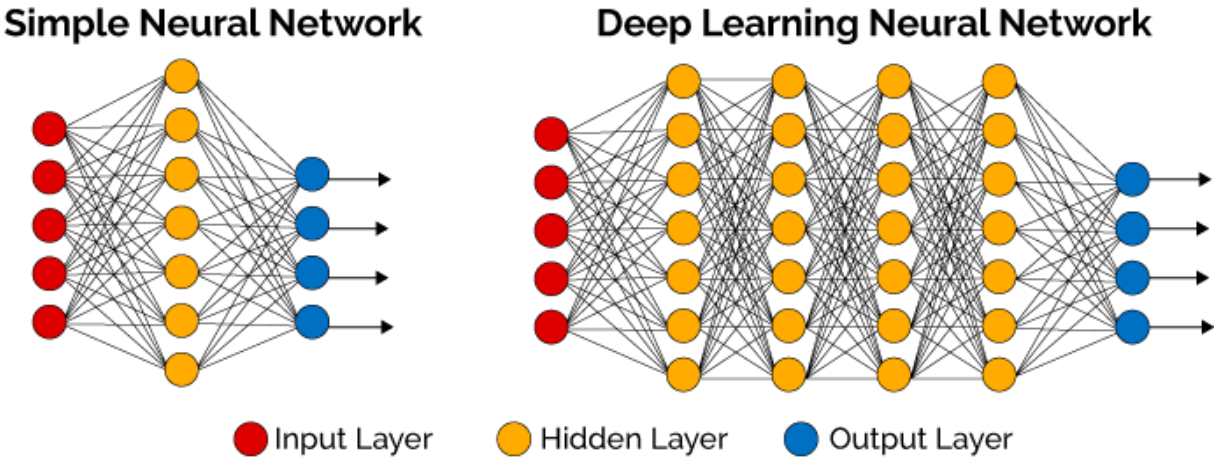
[12] Capítulo 3 - O Que São Redes Neurais Artificiais Profundas ou Deep Learning? - Deep Learning Book. **Deeplearningbook**. Disponível em: <https://www.deeplearningbook.com.br/o-que-sao-redes-neurais-artificiais-profundas/>. Acesso em: 16 de jan. de 2022.

tipo de modelo (*deep learning*), busca imitar o cérebro humano em sua essência mais fidedigna possível.

De forma direta, a diferença mais marcante entre um modelo de ML tradicional e um modelo de DL, está no fato que um modelo tradicional de ML está calcado em cálculos lineares, enquanto que um modelo de DL está calcado em várias camadas (*layers*) que se ligam entre si. Quando o modelo deixa de trabalhar de forma linear e passa a trabalhar com múltiplas camadas, o mesmo passa a imitar o funcionamento de neurônios biológicos.

Estruturalmente, um modelo de Dl recebe os dados de entrada primeira camada que produzem outros dados na saída desta camada, para então, alimentar a próxima camada com estes dados, que por sua vez, produziram novamente outros dados em sua saída. Este processo se repete até a última camada. Basicamente, as camadas superiores são alimentadas pelas camadas inferiores. A primeira camada recebe o nome de **camada de entrada**, enquanto que a última camada recebe o nome de **camada de saída**. As camadas do meio recebem o nome de **camadas ocultas**. Vejamos a seguir, como um modelo de DL funciona.

Figura 7 – Estrutura geral de uma rede neural artificial.



Fonte: Página deeplearningbook, capitulo 3.

6. Considerações finais

É fato que a tecnologia está em constante aperfeiçoamento. Novas tecnologias surgem e uma parte das que já existem, sofrem atualizações. O que vimos neste artigo, é

apenas um arcabouço superficial da I.A. O importante foi demonstrar e comentar conceitos básicos e aplicar a prática para facilitar o entendimento.

Diante de todo o exposto neste artigo, nos foi oferecido a oportunidade de conhecer um pouco sobre esta área tão fascinante e certamente multidisciplinar. Vimos fatos históricos que ajudaram a contribuir para com o desenvolvimento desta (e de muitas outras) tecnologia. Ficou bem claro que a matemática permeia a I.A, mais precisamente o Ml, tema deste artigo.

O aprendizado é constante e certamente, o acompanhará por toda a vida profissional. Este artigo demonstrou a linearidade matemática dos modelos clássicos de ML e como os modelos de DL funcionam em sua essência.

7. Referências

O EXTERMINADOR DO FUTURO. Direção: James Cameron. Produção: Gale Anne Hurd. Estados Unidos: Metro-Goldwyn-Mayer Inc, 1984. VHS (108 min).

FIA. Inteligência Artificial: o que é?, como funciona e exemplos. **FIA**, 2021. Disponível em: <https://fia.com.br/blog/inteligencia-artificial/>. Acesso em: 05 de jan. de 2022.

ARBULU, Rafael. Em 2021, nós lançamos mais foguetes que qualquer outro ano na história. **Olhardigital**, 2022. Disponível em: <https://olhardigital.com.br/2022/01/03/ciencia-e-espaco/em-2021-nos-lancamos-mais-foguetes-que-qualquer-outro-ano-na-historia/>. Acesso em: 05 de jan. de 2022.

MOGNON, Mateus. Blue Origin completa primeiro voo tripulado com Jeff Bezos. **Tecmundo**, 2021. Disponível em: <https://www.tecmundo.com.br/ciencia/221402-blue-origin-completa-primeiro-voo-tripulado-jeff-bezos.htm>. Acesso em: 05 de jan. de 2022.

ALLIBHAI, Eijaz. Hold-out vs. Cross-validation in Machine Learning | by Eijaz Allibhai | Medium. **Medium**, 2018. Disponível em: <https://medium.com/@eijaz/holdout-vs-cross-validation-in-machine-learning-7637112d3f8f>. Acesso em: 15 de jan. de 2022.

MICHEL. Overfitting e Underfitting. **3dimensões**, 2016. Disponível hem: <https://www.3dimensoes.com.br/post/overfitting-e-underfitting>. Acesso em: 06 de jan. de 2022.

Ajuste do modelo: Subajuste versus sobreajuste - Amazon Machine Learning. **Amazon**. Disponível em: <https://docs.aws.amazon.com/pt_br/machine-learning/latest/dg/model-fit-underfitting-vs-overfitting.html>. Acesso em 06 de jan. de 2022.

DELUA, Julianna. Supervised vs. Unsupervised Learning: What's the Difference?. **IBM**, 2021. Disponível em: <https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning>. Acesso em: 25 de jan. de 2022.

Data Mining de Regras de Associação. **Devmedia**. Disponível em: <https://www.devmedia.com.br/data-mining-de-regras-de-associacao/6941>. Acesso em: 12 de jan. de 2022.

HYPERPARAMETER (MACHINE LEARNING). In: WIKIPEDIA, a enciclopédia livre. Flórida: Wikipedia Fundation, 2022. Disponível em: <https://en.wikipedia.org/wiki/Hyperparameter_(machine_learning)>. Acesso em: 12 de jan. de 2022.

Frequent Pattern Mining - Spark 3.2.0 Documentation. **Apache**. Disponível em: <https://spark.apache.org/docs/latest/ml-frequent-pattern-mining.html>. Acesso em: 13 de jan. de 2022.

MANOJ, R. Joseph. FP-Growth Algorithm: Frequent Itemset Pattern | Kaggle. **Kaggle**, 2021. Disponível em: <https://www.kaggle.com/rjmanoj/fp-growth-algorithm-frequent-itemset-pattern/notebook>. Acesso em: 13 de jan. de 2022.

RASCHKA, Sebastian. Mlxtend.preprocessing – mlxtend. **Github**, 2020. Disponível em: <http://rasbt.github.io/mlxtend/api_subpackages/mlxtend.preprocessing/#transactionencoder>. Acesso em: 13 de jan. de 2022.

TAMAIS, Ana Laura Moraes. Modelos de Predição | Naive Bayes | by Ana Laura Moraes Tamais | Turing Talks | Medium. **Medium**, 2019. Disponível em: <https://medium.com/turing-talks/turing-talks-16-modelo-de-predi%C3%A7%C3%A3o-naive-bayes-6a3e744e7986>. Acesso em: 15 de jan. de 2022.

Independência condicional – Wikipédia, a enciclopédia livre. In: WIKIPEDIA, a enciclopédia livre. Flórida: Wikipedia Fundation, 2022. Disponível em: <https://pt.wikipedia.org/wiki/Independ%C3%AAncia_condicional>. Acesso em: 15 de janeiro de 2022.

Capítulo 3 - O Que São Redes Neurais Artificiais Profundas ou Deep Learning? - Deep Learning Book. **Deeplearningbook**. Disponível em: <https://www.deeplearningbook.com.br/o-que-sao-redes-neurais-artificiais-profundas/>. Acesso em: 16 de jan. de 2022.