Sumário

Introdução à inteligência artificial

Computação Natural (CN) - 1960

- Formalizada em 2004 por Leandro N. de Castro e Fernando J. Van Zuben
- Permeia três formas de implementação
    - Computação em que ideias são obtidas através da observação da natureza, que é a inspiração para o desenvolvimento de soluções de problemas complexos
        - Exemplos
            - Redes neurais artificiais (1943)
            - Computação evolutiva (1965)
            - Inteligência de enxame (1988)
            - Sistemas imunológicos artificiais (1999)
            - Entre outros ...
    - Síntese de fenômenos naturais através da computação que envolve mecanismos de computação para sintetizar comportamentos naturais, padrões e processos biológicos
        - As principais linhas de atuação são os estudos sobre a vida de organismos artificiais
            - Vida artificial (1998)
            - Geometria fractal (1982)
    - Computação com mecanismos naturais que são novos paradigmas de computação e podem resultar em computadores altamente potentes, chamados computadores naturais
        - Baseados em computação molecular, através de cadeias de DNA (1998)
        - Computação quântica (2000)
        - Entre outros

Inteligência Computacional (IC) - 1994 & Aprendizagem de Máquina - 2007

- Surgiu como uma forma de desassociar o que a IA clássica pretendia
- Algoritmos de Redes Neurais Artificiais (RNA), Computação Evolutiva (algoritmos genéticos) e Lógica de Fuzzy fizeram parte das técnicas de implementação do IC
- O Machine Learning, atualmente muito associado ao Big Data e ao Analytics, foi defendido por T. Mitchell em 1997 e surgiu dos sistemas baseados em conhecimento da IA clássica
- O grande desafio é desenvolver sistemas capazes de aprender
    - Por si mesmos, através de experiências e comportamentos passados (aprendizagem não supervisionada)
    - Por meio de entrada de mapas de dados (aprendizagem supervisionada)
    - Interagindo com o ambiente (aprendizagem por reforço), p.e. dirigindo um carro
- Para a implementação do Machine Learning, diversas técnicas estão envolvidas, do uso de estatística para auxiliar na análise e predição de dados a técnicas de mineração de dados (Data Mining), algoritmos de árvore de decisão, redes Bayesianas e processos de clustering
- O Deep Learning é uma técnica de Machine Learning eficaz e precisa para a Aprendizagem de Máquina que utiliza grandes quantidades de dados não estruturados e possibilita uma representação hierárquica das camadas de dados
- Algoritmos de RNA são utilizados no Deep Learning justamente por permitir que o aprendizado de padrões ocorra. Quaisquer soluções que envolvam reconhecimento de voz, processamento de imagem, análise de comportamento, entre outras características, são aplicações factíveis de Deep Learning

Aplicabilidade

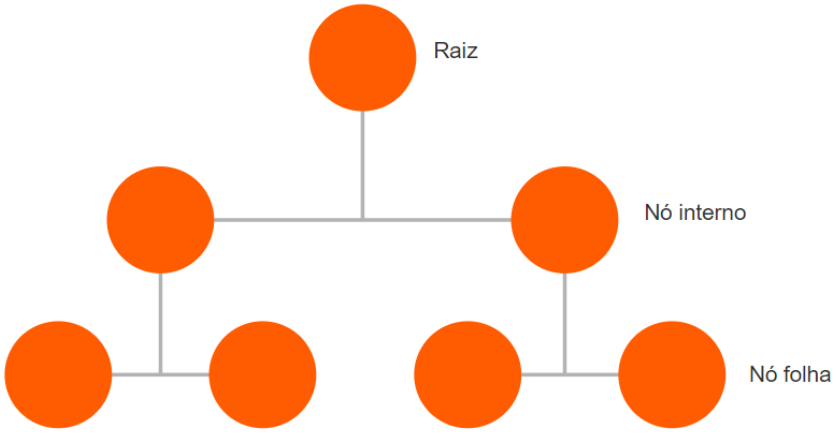
- O mais interessante do posicionamento da IA é ser altamente aplicável nas soluções de negócios
- Instituições financeiras e de cobrança têm utilizado algoritmos de detecção de fraude que analisam padrões no processamento dos dados trafegados, buscando validar, encontrar erros de informações e minerá-los em grupos segmentados. O que permite que os sistemas sejam capazes de prever "candidatos" a fraudes, tendo por base situações anteriormente detectadas e mapeadas, ou ainda simular fraudes que jamais ocorreram, mas que tenham possibilidade de serem aplicadas
- Técnicas de análise forense, arquiteturas de redes Bayesianas e algoritmos de classificação são muito utilizados na predição dos dados para detecção. Grandes montadoras têm apostado em veículos autônomos. Por meio de algoritmos de Deep Learning, é possível aperfeiçoar o reconhecimento de imagens, aumentando a segurança, além de permitir o aprendizado com a experiência realizada
- Grandes centros médicos no mundo estão cada vez mais recheados de equipamentos e sistemas inteligentes para leituras de imagens médicas. A técnica de processamento de imagens utiliza métodos de IA, tais como Redes Neurais Artificiais e Lógica Fuzzy. O Google desenvolveu a API Google Vision, baseada em Aprendizagem de Máquina e reconhecimento de padrões, sendo possível, de maneira muito simples, implementar a leitura de imagens e a exportação em texto do conteúdo da imagem
- Os cada vez mais atuantes chatbots como atendimento inteligente e automatizado são um grande canal de agilidade no atendimento sem a perda na qualidade da informação. O Poupatempo, projeto do Estado de São Paulo que oferece diversos serviços de emissão de documentos, atestados, licenciamento veicular, entre outros, implantou o Poupinha, um atendente virtual (Chatbot). Através do próprio Portal do Poupatempo ou da página do Facebook, ele tira dúvidas do usuário sobre informações das condições, prazos, valores e retirada de documentos, atendendo, em média, 5 mil usuários ao dia
- A implementação da mineração de sentimentos e de opiniões tem sido outra grande aposta da utilização de IA. As redes sociais e as interações digitais, via os diversos dispositivos móveis, têm produzido uma grande quantidade de informações descontroladas e desorganizadas, afastando a personificação do ser humano perante a sociedade, produtos e serviços. Uma reaproximação se faz necessária para um entendimento mais real dos significados intrínsecos dos posts, imagens, vídeos, conversas e likes. Classificar o grau de sentimento que o usuário dá às interações e suas opiniões implícitas ou explícitas é artefato sendo trabalhado pela é artefato sendo trabalhado pela IA em prol da personificação

Resolução de problemas por meio de busca

Problemas e soluções

- Há diversas maneiras de se resolver um problema, assim como há diversos tipos de problemas. Geralmente os problemas podem ser solucionados com um conjunto de ações que levam a um objetivo. Quando as possibilidades para solução de um problema tornam-se quase inumeráveis, o espaço para se explorar uma solução deve ser considerado com técnicas que guiem a resolução do problema, levando em conta **limitações de tempo**, **processamento** e **memória**
- Definição formal de um problema
    - **Um problema é definido em cinco componentes** (considerando um agente que realiza entregas de produtos adquiridos pela internet partindo de um centro de distribuição (CD) até chegar ao local de entrega (LE), percorrendo o caminho (C) mais barato possível, de acordo com uma função de custo (FC)
        - Um **estado inicial** (S_0) a partir do qual o agente inicia. O agente inicia no CD, que é o S_0
        - Uma descrição das **ações** (a_n) possíveis ao agente. Dado um estado s, AÇÕES consiste, por exemplo, nas vias que partem do CD
        - Um **modelo de transição** entre as ações, que defino o próximo estado S_y, dada a escolha de uma ação, estando no estado S_x. É formalizado como RESULTADO(S_x, a) = S_y. Esses três componentes, estado inicial, ações e o modelo de transição compõem o espaço de estados do problema. O **espaço de estados** pode ser representado como um **grafo** dirigido, em que os nós são os estados e as arestas são as ações. Nesse grafo, um **caminho** é uma sequência de estados conectados por ações (arestas)
        - Um teste de **objetivo**, que verifica, a cada estado, se o estado é o objetivo. Para alguns problemas, pode-se ter mais de um estado objetivo, que define a solução do problema. No exemplo dado, o **estado objetivo** é o local de entrega (LE) da mercadoria adquirida
        - Uma **função de custo do caminho**, que atribui um valor numérico a cada caminho do grafo. Na função de custo do agente entregador a quilometragem, por simplicidade, pode ser considerada como função de custo do caminho, quanto menor o caminho, menor o custo
    - No exemplo do agente entregador, além de considerar o caminho mais curto, a modelagem real deve levar em consideração outras variáveis, tais como:
        - Tempo da viagem
        - Horário de trabalho do entregador, que deve se adequar à legislação trabalhista vigente
        - Velocidade das vias
        - Condições reais do trânsito nas vias
        - Número de cruzamentos
        - Dentre muitas outras variáveis

Métodos de busca por solução



- Cada possível solução de um problema forma o que chamamos de **árvore de busca**, originado do universo de possíveis soluções. Uma árvore consiste em uma estrutura de dados não linear composta por nós (informações) organizadas hierarquicamente
- Critérios de avaliação da busca
    - Completeza
        - Capacidade do algoritmo pesquisar a árvore toda caso necessário
    - Otimalidade
        - Capacidade de encontrar a melhor solução possível
    - Complexidade de tempo
        - Tempo médio para se encontrar a solução
    - Complexidade de espaço
        - Espaço contido na árvore que será processado pelo algoritmo

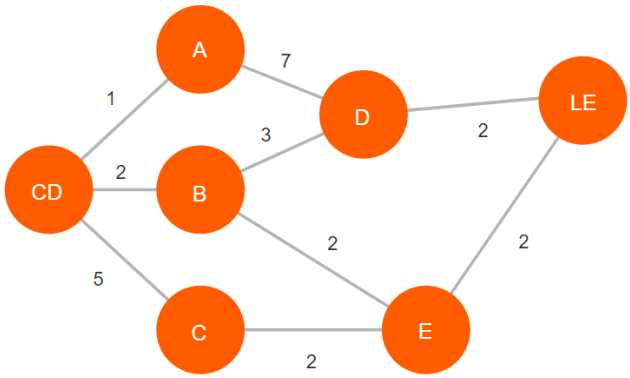
Busca sem informação

- A busca sem informação, ou busca cega, é realizada sem informação adicional sobre os estados, além da definição do problema. A estratégia é gerar novos estados e verificar se um estado objetivo é alcançado ou não. Difere na ordem em que os estados são explorados: busca em largura ou em profundidade
- Busca em largura
    - A busca em largura (BFS - *Breadth-First Search*) é uma estratégia em que o nó raiz é verificado antes dos nós filhos do nó raiz. Essa estratégia é utilizada recursivamente para os nós internos da árvore (subárvores)
- Busca em profundidade
    - Na busca em profundidade (DFS - *Depth-First Search*) busca-se o nó mais profundo a partir da raiz

Busca informada

- É assim chamada pois se tem informação, dado um conjunto de possíveis estados, de qual é mais promissor, dado o estado atual. Esse conhecimento adicional é obtido por heurísticas, ou aproximações, utilizadas durante a busca
- A cada passo utiliza-se uma **busca de melhor escolha**. A escolha de um nó a ser percorrido é feita com base na **função de avaliação** $f(n)$. Assim, dado um estado atual, a função é aplicada a cada possível expansão da solução com a finalidade de avaliar o custo. A expansão com menor custo é escolhida

- **Busca gulosa de melhor escolha**
    - A busca gulosa expande a busca para o nó que seja o mais próximo do estado objetivo considerando apenas o caminho do estado atual ao próximo estado



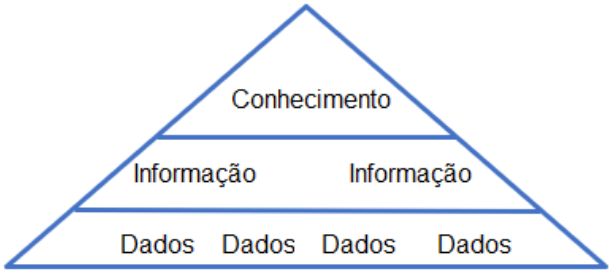
- A busca gulosa, estando no CD, considera o menor custo para a próxima expansão, isso levará ao nó A, que custa 1. A partir de A, a próxima expansão levará a D e, depois a LE. Esse caminho tem custo total de 1+7+2=10. Obviamente, esse não é o melhor caminho, que é CD-B-D-LE, com custo total de 2+3+2=7
- Nessa estratégia $f(n) = \mathrm{h}(n)$, ou seja, a função avaliação é igual à função heurística de escolher o menor custo a partir do estado atual
- **Busca A*** (minimizando custos)
    - A busca A* ("A estrela") tem como função de avaliação $f(n) = g(n) + \mathrm{h}(n)$, sendo $g(n)$ o custo para ir do nó inicial ao nó n e $\mathrm{h}(n)$ o custo estimado para ir do nó n ao nó objetivo. Essa estratégia é uma combinação de aproximações heurísticas, como a busca em largura e o famoso algoritmo de *Dijkstra*. Ele é muito utilizado em aplicativos de rotas de deslocamento entre localidades e até na resolução de quebra-cabeças. Também é muito utilizado em jogos

Representação do conhecimento e raciocínio

Agentes baseados em conhecimento

- Utilizam uma **base de conhecimento** (KB - *Knowledge Base*) armazenadora de **sentenças**, que, desaliada de um sentido puramente linguístico, representa uma **asserção** sobre o mundo. As sentenças são expressas em uma **linguagem de representação do conhecimento**. Uma sentença pode ser chamada de **axioma**, quando tal sentença é dada originalmente, isto é, sem derivação de outra sentença
    - O conhecimento em um KB é geralmente dinâmico, ou seja, novas sentenças são adicionadas e outras podem ser removidas ou modificadas. Vamos referenciar a inserção de novas sentenças pela operação TELL e a busca por uma sentença, como ASK. Essas operações podem utilizar-se de **inferências** de novas sentenças a partir das existentes no KB

Representação do conhecimento



- Antes, considere como **dados** quaisquer sequências numéricas ou alfanuméricas que não possuem um significado associado (3, 4, 5, ...). Quando esses dados são organizados e rotulados (tamanho 3, camada 4, 5 unidades, ...), passam a ser chamados de **informação**, pois significam algo. **Conhecimento** é a composição e organização de informações que permitem raciocínio por parte de um ser humano ou de uma máquina (objeto de tamanho 3, na camada 4, com 5 unidades, ...)
- Uma das formas mais antigas de representar o conhecimento de forma a permitir o raciocínio por parte de um agente inteligente é o uso da lógica. Inclusiva, diversos sistemas especialistas do início da IA tiveram a lógica como sua base de conhecimento e raciocínio
- **Lógica** é uma linguagem de propósito geral utilizada para descrever fatos (verdades) em um mundo. A lógica também possui mecanismos para operar sobre os fatos visando o raciocínio. Assim como outras linguagens, tem-se uma **sintaxe**, que define a formação das sentenças e uma **semântica**, que dá o significado às sentenças. Cada sentença é sempre avaliada em **verdade** ou **falso**
- Linguagens naturais, como o Português e o Inglês também são uma forma de representar conhecimento. Tanto é que o conhecimento deste capítulo está representado utilizando o Português. No entanto, como será percebido na continuação deste capítulo, a linguagem natural é muito complexa e tem muitos fenômenos de difícil tratamento computacional, como a ambiguidade. Portanto, vamos nos atentar a algumas formas simbólicas de representação do conhecimento cujo tratamento é mais plausível, computacionalmente

Lógica Preposicional

- É um tipo de lógica bem simples. Sua sintaxe define **sentenças atômicas**, que consistem em um **símbolo preposicional**. Aqui definiremos que as palavras que iniciam com letra maiúscula (P, Q, Humano, Máquina, etc.) são símbolos preposicionais. Cada símbolo preposicional é sempre avaliado em **verdade** ou **falso**. **Sentenças complexas** são feitas a partir de sentenças mais simples com o uso de **conectivos lógicos**: ¬(não), ^ (e), v (ou), ⇒ (implica) e ⇔ (se e somente se)

- Por exemplo, o uso do operador de negação para o símbolo P é ¬P, que significa "não P" ou "P é falso", ou "não é o caso de P". Os outros operadores dependem de uma **tabela verdade**. Considere a tabela a seguir com os operadores ¬(não), ∧ (e), ∨ (ou), ⇒ (implica, condicional) e ⇔ (se e somente se, bicondicional)

P	Q	∧	∨	⇒	⇔
V	**V**	V	V	V	V
V	**F**	F	V	F	F
F	**V**	F	V	V	F
F	**F**	F	F	V	V

  - Pela tabela verdade, a sentença $P \wedge Q$ (P e Q) assume os valores falso (F) quando pelo menos um dos símbolos preposicionais forem falsos e só assume valor verdadeiro (V) quando os dois símbolos são verdadeiros. O operador ∨ (ou), no entanto, leva a um valor verdade quando pelo menos um dos símbolos é verdade e só assume falso quando os dois são falsos. Como exercício, interprete os outros dois operadores
- Imagine um agente (um Chatbot, por exemplo) que deve responder "sim" (verdade) ou "não" (falso), para afirmações de um usuário. Ele deve avaliar perguntas como:

1. A terra é redonda?	4. A terra é quadrada **OU** a lua é redonda?
2. A terra é quadrada **E** a lua é redonda?	5. A terra **NÃO** é quadrada **E** a lua é redonda?
3. A neve é branca **SE** a água é incolor?	

  - Baseado em nosso conhecimento de mundo, as sentenças 1, 3, 4 e 5 são verdade e a sentença 2 é falsa. Embora útil em alguns cenários, para representar um conjunto de sentenças (conhecimento), sua representatividade, em cenários mais complexos, é limitada. A lógica preposicional manipula bem conhecimentos que podem ser expressos pelos conectivos lógicos entre as sentenças atômicas. Mas e para expressar coisas como "Existe algum ... tal que ..." ou " Para todo...". Como representar em lógica preposicional as sentenças: "Ser mais jovem que outrem", ou "Existe um X que é maior que todos os outros Xs"? Para isso, é utilizada uma linguagem com maior poder de expressão, a **lógica de primeira ordem**

Lógica de primeira ordem (LPO)

- Tem poder de expressão maior que a lógica preposicional. Sendo estudada por décadas, a LPO é utilizada como base para outras linguagens como o Prolog. A LPO estende lógica preposicional. Enquanto esta se ocupa da existência dos fatos, aquela se ocupa da existência de objetos e de relações entre eles
- Na LPO, as **sentenças atômicas** têm o seguinte formato: P(t_1, ..., t_n). P é chamado de predicado e tem um ou n **argumentos**. A LPO também contém **quantificadores**, a saber, $\exists x$ (para todo x) e $\forall x$ (para qualquer x). Tem-se também os **termos**, que são constantes, e, geralmente, se referem a objetos do mundo real, concretos ou abstratos. Assim, podemos expressar fatos por meio de predicados, como homem(João) ou mulher(Maria). Esses predicados podem significar que João é um homem e Maria é uma mulher

- Para expressar os quantificadores, temos que definir o conceito de variável, que será denotada como uso de uma letra latina minúscula, geralmente mais do final do alfabeto, tais como x, y, z e w. Essas variáveis podem ser pensadas como posições de memória que podem ser ocupadas ou instanciadas por valores como João e Maria. Desta forma, podemos ter:

homem(x), em que x é um homem	**pai(x, y), em que x é pai de y**
mulher(x), em que x é uma mulher	**mãe(x, y), em que x é mãe de y**

- E, dados os predicados anteriores, $\forall y$ (para qualquer y), se pai(x, y) então maisnovo(y, x). Em outras palavras, se x é pai de y, então y é mais novo que x. Para um conhecimento como: todo filho é mais novo que seu pai, definimos: $\forall y(pai(x,y) \rightarrow \exists x\big(maisnovo(y,x)\big))$
    - O predicado que admite apenas uma variável é considerado um predicado unário. Já o predicado que admite duas variáveis é binário
- Dada uma base de conhecimento expressa em LPO, algumas inferências podem ser feitas para responder a algumas questões. Considere a base de conhecimento:

pai(x,y): x é pai de y
filho(x,y): x é filho de y
irmão(x,y): x é irmão de y

    - Algumas conclusões podem ser inferidas:
        - se quisermos descobrir quem é irmão de uma dada pessoa, podemos codificar: $\forall x \forall y \forall z\big(pai(x,y) \wedge filho(z,x) \rightarrow irmão(y,z)\big)$
        - Se instanciarmos $y = Pedro$, teremos que qualquer instância de z será irmão de Pedro: $\forall x \forall y \forall z(pai(x,pedro) \wedge filho(z,x) \rightarrow irmão(pedro,z))$
- Se nossa base de conhecimento contiver:

filho(carlos, amaral)	**pai(amaral, lorena)**
filho(pedro, amaral)	**pai(amaral, pedro)**
filho(estância, amaral)	

- Concluímos que *Pedro* é irmão de *Lorena*
- Para entendermos um pouco mais sobre os quantificadores, considere: $\forall x \, \exists y \, ama(x,y)$
    - Para todo x, existe um y, tal que x, ama y, em outras palavras, todo mundo ama alguém. Porém, se quisermos dizer que existe um alguém que é amado por todo mundo, teremos que escrever $\exists y \, \forall x \, ama(x,y)$: existe um y que, para todo x, x ama y
- Um agente baseado em conhecimento pode armazenar novas sentenças em KB, utilizando a operação **TELL**. Por exemplo, $TELL(BK,\ pai(amaral, murilo))$ adiciona em KB o conhecimento de que *amaral* é pai de *murilo*, gerando um novo irmão para *pedro*

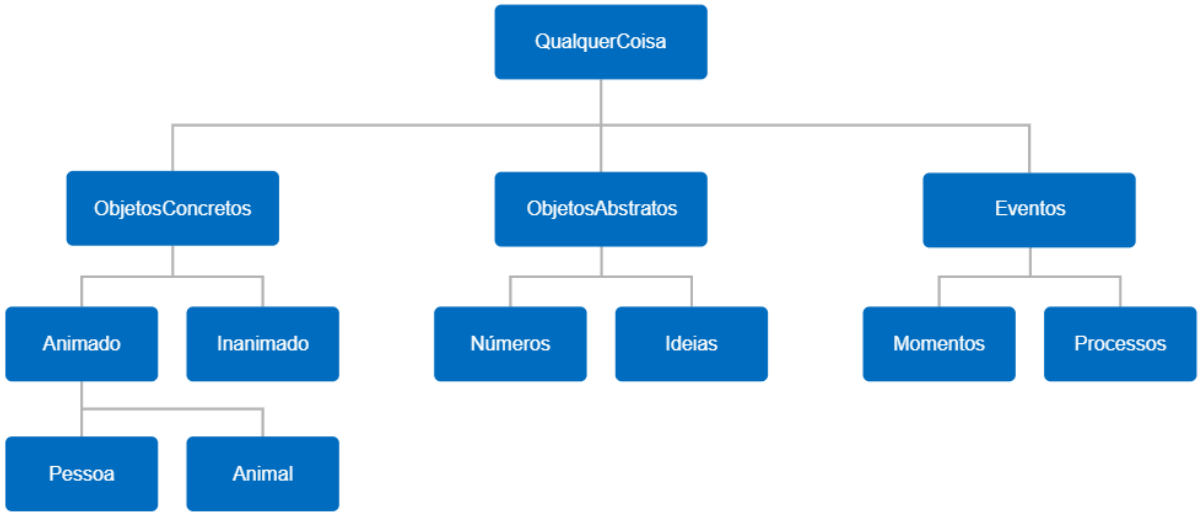
Engenharia de Conhecimento

- Foi definida a LPO como uma forma de representar um conhecimento (KB), com a finalidade de se raciocinar e resolver problemas. Para se gerar a KB, tem-se um processo chamado **engenharia de conhecimento**. Um engenheiro de conhecimento investiga o domínio do problema, aprende conceitos importantes e define uma representação formal dos objetos e relações entre esses objetos no domínio
- Nesse processo podem-se gerar bases de conhecimento para um domínio bem circunscrito ou abordagem de uma variedade maior de conhecimentos humanos. Quanto maior a variedade, maior a complexidade do KB e mais desafiadora é a engenharia de conhecimento

- As seguintes etapas são necessárias (nessa ordem):
  - **Identificar a tarefa**
    - Deve-se delinear as questões que a base de conhecimento suportará e os tipos de fatos disponíveis
  - **Agregar o conhecimento relevante**
    - Ou o engenheiro dever especialista do domínio ou fazer uma aquisição de conhecimento com um especialista no domínio. Por exemplo, para gerar uma base de conhecimento sobre diagnósticos médicos de uma dada classe de doenças, um especialista nessa classe de doenças deve ser amplamente consultado
  - **Definir um vocabulário de predicados e constantes**
    - Aqui se convertem os conhecimentos adquiridos em predicados e constantes em LPO, por exemplo. Definem-se predicados unários, binários e suas significações, por exemplo $pai(x, y)$ indica que x é pai de y ou que y é pai de x?
  - **Codificar o conhecimento geral do domínio**
    - O engenheiro de conhecimento escreve os axiomas (termos iniciais, não provindos de inferência) correspondente a todos os termos do vocabulário
    - Essa etapa permite ao especialista verificar se o conhecimento adquirido foi corretamente convertido na representação lógica. Quando erros são identificados, deve-se retornar ao passo anterior
  - **Codificar uma descrição da instância específica do problema**
    - Envolve a escrita de sentenças atômicas sobre o problema, instanciando predicados com conhecimentos adquiridos
  - **Formular consultas de inferência para obter respostas**
    - Aqui pode-se testar o funcionamento do raciocínio sobre os axiomas e fatos codificados nas etapas anteriores, inferindo novos conhecimentos, conforme nossos objetivos de utilizar a base de conhecimento. Utiliza-se todo o poder de inferência da LPO, não necessitando que se escreva um programa para cada inferência que se deseja obter
  - **Depurar a base de conhecimento**
    - Se um axioma estiver ausente da base, algumas consultas poderão não ser respondidas. Axiomas ausentes ou axiomas muito fracos podem ser identificados quando a inferência é interrompida de forma inesperada. Faz-se necessária a revisão dos passos anteriores, garantindo que as consultas desejadas sejam respondidas satisfatoriamente

Engenharia Ontológica

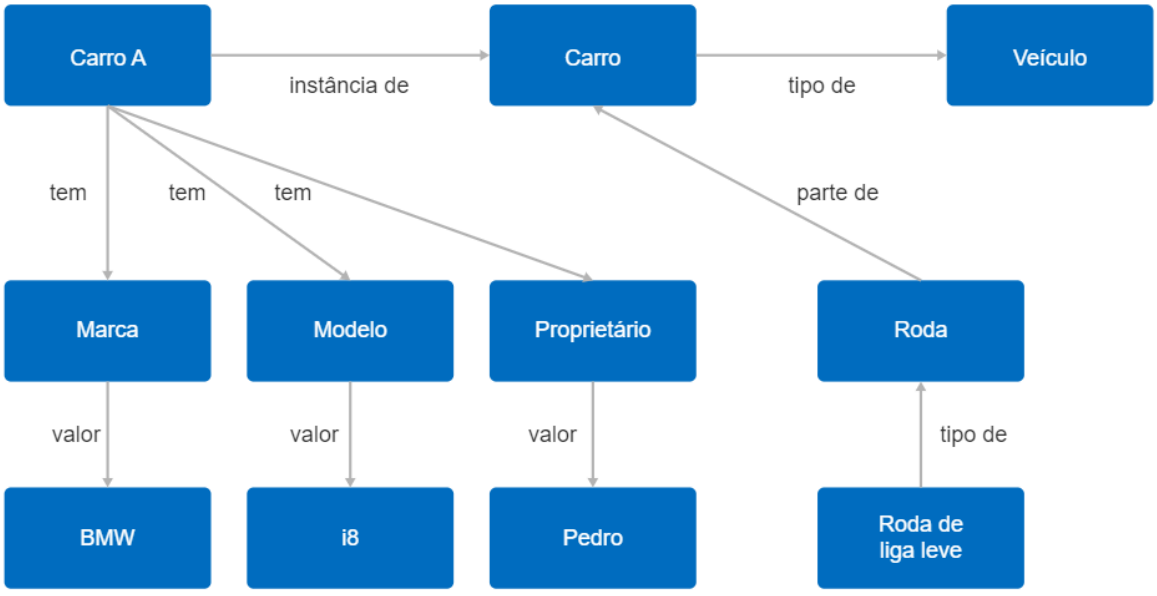
- Organiza tudo no mundo em uma hierarquia de categorias, dessa forma, são definidas categorias de objetos, substâncias e medidas, eventos e conhecimentos abstratos



- A figura ilustra a criação de uma ontologia. No nível mais superior, encontra-se "QualquerCoisa", indicando que essa ontologia pode organizar todos os conceitos do mundo real. Representar tudo, no entanto, é uma tarefa muito complexa e quase inatingível. Embora comece com "QualquerCoisa", o que se especifica de cima para baixo na ontologia pode ser específico a um determinado domínio
- A ontologia ilustrada na figura é chamada de **ontologia superior**, pois define conceitos gerais na parte superior e vai especificando os conceitos à medida em que desce na ontologia
- A definição de **categorias** é uma parte importante da engenharia ontológica, pois embora a interação no mundo real ocorra com objetos específicos, boa parte do raciocínio feito sobre conceitos ocorre no nível de categorias
    - Interage=se com um carro específico, mas um consumidor, ao procurar um carro, busca na categoria de carros e não uma instância específica de carro
- Outra definição importante a uma representação de conhecimento é a relação **parte de**. Objetos podem ser agrupados como **parte de** uma categoria de objetos. Por exemplo, temos ParteDe(roda, carro), ParteDe(motor, carro) ou ParteDe(freios, carro). Essa relação geralmente ocorre em **objetos compostos**, como o exemplo do Carro
- Outras definições devem ser levadas em consideração, como a definição de **medições**, **eventos**, **processos** e **intervalos de tempo**

Redes Semânticas

- As **categorias** são os principais blocos de construção de bases de conhecimento em grande escala. As **redes semânticas** são um sistema que permite a visualização gráfica de uma base de conhecimento e algoritmos para a dedução de propriedades de um objeto
- Existem diversas variantes de redes semânticas, mas todas podem representar objetos individuais, categorias de objetos e relações entre os objetos. A notação gráfica típica representa os objetos e categorias nomeados em retângulos ou elipses e os conecta com arcos rotulados pelas relações entre os objetos ou categorias

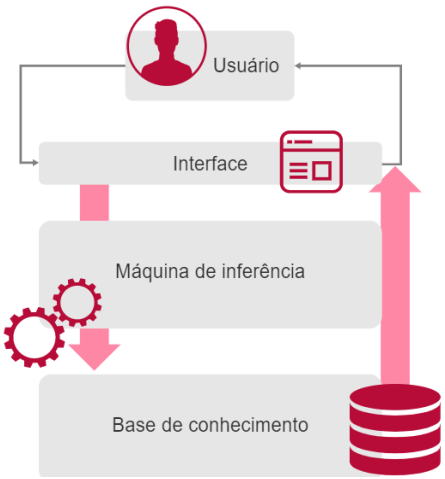


Introdução aos sistemas especialistas

Sistemas Baseados em conhecimento

- Um sistema é dito **especialista** quando seu comportamento ou decisões imitam o comportamento ou decisões de um especialista da área de atuação do sistema. Assim, um sistema especialista de diagnóstico em uma área médica baseia-se em conhecimento obtido de médicos da área. Esse sistema diagnosticará de acordo com o conhecimento que obteve dos médicos, imitando suas decisões
- A diferença entre um **sistema algorítmico** e um sistema especialista é que o sistema algorítmico armazena limitando conhecimento em formato de código (por exemplo, pelo uso de comandos condicionais). Os sistemas especialistas, no entanto, utilizam uma base de conhecimento extraída de especialistas do domínio do problema a ser resolvido. Caso seja necessário aumentar o conhecimento do domínio para melhorar o desempenho do sistema, o sistema algorítmico necessitará de mudança em seu código. Já o sistema especialista precisará apenas aumentar sua base de conhecimento, raramente necessitando alterar seu motor de inferência na base de conhecimento
- Um SE (sistema especialista) pode ser utilizado em diversos cenários com o intuito de agilizar e uniformizar soluções em organizações. Podemos citar diversos cenários:
    - Gerenciamento de decisões
    - Diagnóstico de problemas
    - Monitoramento e controle de processos
    - Seleção e classificação de opções
- Pode-se citar, também diversas áreas funcionais em que um SE pode ser utilizado:
    - Gestão financeira
    - Estratégias de marketing
    - Gerenciamento de materiais
    - Planejamento de recurso capital
    - Dentre outras

Etapas do desenvolvimento de um SISTEMA ESPECIALISTA (SE)



- A figura apresenta uma possível arquitetura para um SE. O usuário interage com o sistema através de uma interface, por meio da qual se possa obter respostas para problemas do domínio escolhido. A interface repassa as entradas fornecidas pelo usuário à máquina (ou motor) de inferência, que, por sua vez, busca na base de conhecimento a resposta mais provável para o questionamento do usuário

- Passos para a construção de um SE:
    - **Definição do problema**
        - Deve-se definir o "problema" a ser resolvido pelo SE. O escopo do problema deve ser formalizado, pois dificilmente um sistema especialista conseguirá tratar todo o domínio de um problema
    - **Aquisição do conhecimento**
        - Aqui, dada delimitação do problema, um ou mais especialistas do domínio deverão ser exaustivamente consultados para a montagem da base de conhecimento
        - É importante que o construtor do SE tenha também algum conhecimento do problema, ter um conhecimento do domínio possibilitará uma conversa mais informada com os peritos da área escolhida
    - **Representação do conhecimento**
        - Após a obtenção do conhecimento junto a especialistas, esse conhecimento deve ser adequado a uma representação do conhecimento. No capítulo anterior foram apresentados diversos formalismos para a representação do conhecimento, como a LPO (Lógica de Primeira Ordem) e redes semânticas. Esses formalismos possibilitam a criação de diversos motores de inferência, para a tomada de decisão com base no conhecimento representado
    - **Codificação**
        - Nesse passo, basicamente, deve-se codificar o sistema que, utilizando a base de conhecimento em uma dada representação, atenderá a demandas do usuário do SE. Como outro sistema computacional, uma arquitetura modular e escalável deve ser adotada para que o sistema possa crescer sem modificações muito estruturais no código
    - **Avaliação do SE**
        - Após a construção do sistema, os especialistas do domínio, junto com os construtores do SE, devem verificar a consistência das respostas obtidas automaticamente. Geralmente, separa-se um conjunto do conhecimento obtido pelos especialistas para essa etapa. Esse conjunto de teste não deve estar presente na base de conhecimento do SE. Diversas medidas de avaliação podem ser utilizadas, como precisão (P), cobertura (C) e a medida-F (uma média harmônica entre P e C)
        - Detectando-se uma baixa performance do SE, a causa deve ser identificada. Pode ser devido ao conhecimento, que é insuficiente ou está armazenado de maneira inconsistente. Ou pode ser devido à máquina de inferência (codificação) que não é poderosa o suficiente para buscar o conhecimento necessário para a reposta esperada
    - **Manutenção**
        - Por fim, o sistema deve ser bem documentado e ter uma interface de fácil utilização para os usuários do sistema. Se a base de conhecimento possibilitar a dedução de novos conhecimentos, periodicamente, o sistema deve ser avaliado para verificar se o conhecimento aderido está correto

O Motor de INFERÊNCIA

- É considerado o cérebro do SE, pois atua sobre a base de conhecimento (raciocina) em busca de uma resposta a uma questão do problema
- Como visto nos capítulos anteriores, o motor de inferência pode utilizar uma abordagem de busca até encontrar uma resposta para o usuário, quando o conhecimento está armazenado em uma árvore de decisão, por exemplo
- O raciocínio pode ser feito de maneira progressiva ou regressiva
    - No **raciocínio progressivo**, à medida que o usuário vai fornecendo evidências do problema, o sistema vai desencadeando o processo de busca na base de conhecimento até encontrar uma resposta para o usuário
    - Já no **raciocínio regressivo**, o SE escolhe, segundo alguma heurística, uma resposta/hipótese (que inclusive pode ser fornecida pelo usuário) ao problema e, numa busca reversa, verifica se a resposta é plausível, segundo o conhecimento armazenado na base de conhecimento
- Caso a representação do conhecimento tenha sido em Lógica de Primeira Ordem, o uso de uma linguagem como o Prolog traz recursos de inferência como **backtracking** (ou retrocesso) em cláusulas de LPO
    - Para exemplificarmos o uso do **backtracking**, considere uma simples base de conhecimento em LPO que representa a compatibilidade entre modelos de baterias e determinados equipamentos eletrônicos:

1	Compatível(Bateria_58PX, Filmadora1)
2	Compatível(Bateria_58PX, Celular1)
3	Compatível(Bateria_98UX, Celular1)
4	Compatível(Bateria_765, Notebook1)
5	Compatível(Bateria_58PX, Celular2)
6	Compatível(Bateria_123, Celular3)

    - Agora imagine que se deseje saber se há algum dispositivo eletrônico compatível com dois tipos de bateria: "Bateria_58PX" e "Bateria_98UX". Para isso realiza-se uma pergunta, no seguinte formato: *Compatível(Bateria_50PX, Y), Compatível(Bateria_98UX, Y)*
    - A primeira cláusula da pergunta é instanciada com o conhecimento da linha 1 e Y=Filmadora1. Agora deve-se procurar pela segunda cláusula com o valor de Y instanciado, ou seja, *Compatível(Bateria_98UX, Filmadora1)*. Essa busca não retorna valor. Nesse ponto ocorre o **backtracking**, em que se retorna ao momento em que Y foi instanciado como "Filmadora1" e desfaz essa operação
    - Agora, a primeira cláusula é instanciada com o conhecimento da linha 2 e Y=Celular1. Agora, procura-se por *Compatível(Bateria_98UX, Celular1)* e obtém-se sucesso com o conhecimento da linha 3. Assim, conclui-se que os dois modelos de bateria são compatíveis com o dispositivo "Celular1"
- Embora pareça trivial, o *backtracking* resolve problemas de busca em LPO sem que o programador tenha que codificar a busca. No entanto, em bases de conhecimento muito grandes, esse procedimento pode levar a uma explosão combinatória, pois é um procedimento inerentemente combinatório

Formas de interação com o SE

- Um sistema especialista pode ter diversas formas de interagir com o usuário, como uma interface gráfica que contém diversos elementos de escolha (formulários) para o usuário dar evidências ou sintomas, por exemplo, e obter uma resposta
- A forma mais intuitiva é o uso da linguagem natural, por meio de um diálogo. Nesse caso, o SE deve ter um bom módulo de PLN (Processamento da Linguagem Natural) para interpretar o texto do usuário e mapear o que é dito para o conhecimento armazenado na base de conhecimento
    - Considerando o exemplo apresentado anteriormente (*backtracking*), um usuário poderia utilizar a seguinte entrada em linguagem natural: m
        - "Existe algum dispositivo que seja compatível com as baterias 'Bateria_58PX' e 'Bateria_98UX'?"
    - Cabe ao módulo de interpretação textual converter a entrada nas células *Compatível(Bateria_58PX, Y), Compatível(Bateria_98UX, Y)* e retornar o valor de Y para o usuário

Tomada de DECISÃO

- Um sistema especialista deve imitar um especialista da área em que vai atuar. Portanto, dada uma entrada ao sistema ele deve **tomar a decisão** que melhor representa a decisão do especialista. Em um cenário real, um especialista ficará com certa incerteza com relação a qual decisão tomar e optará pela mais provável, isto é, optará pela decisão com maior **probabilidade** de acerto
- Levando em conta as possíveis decisões que um sistema pode tomar e suas probabilidades, tem-se um formalismo chamado de **redes de decisão** (ou diagramas de influência) muito útil à construção de sistemas especialistas
    - As **redes de decisão** são um mecanismo geral para tomadas de decisão racional. Essas redes basicamente representam:
        - O estado atual do agente
        - Suas ações possíveis a partir do estado atual
        - O estado que resultará da escolha de uma ação
        - E o ganho obtido dessa escolha

Exemplos de Sistemas Especialistas

REC_ESP

- Bronaut *et al.* (2007) apresentam o REC-ESP, um SE desenvolvido para a retomada do sistema de elétricos de potência do estado do Mato Grosso do Sul. Feito em conjunto entre a ENERSUL e a UFMS, o REC_ESP tem seu conhecimento extraído de um manual de operações da ENERSUL e do conhecimento dos operadores do sistema elétrico
- A base de conhecimento do REC_ESP consiste em um conjunto de regras que levam a sugestões sistêmicas, a partir de verificações de variáveis em pontos estratégicos do sistema elétrico (como tensão, patamar de carga etc.). Um exemplo de regra, que apresenta uma sugestão, a partir da tensão é descrito a seguir
    - Se Tensão na barra A maior que 145 kV então, tem-se Tensão elevada, o Disjuntor 01 não pode ser fechado
- Como o objetivo do SE é restabelecer as linhas de transmissão de energia mais prioritárias para o estado do MS, ele tem um sistema de busca (inferência) que analisa os caminhos entre as estações centrais e estações de borda, escolhendo as linhas mais prioritárias
- É interessante que o REC_ESP não serve apenas para guiar os operadores no restabelecimento do sistema de eletricidade, as também é utilizado para treinar novos operadores antes de ocorrer problemas na rede elétrica

Vantagens e limitações de um SE

- Vantagens
    - Armazena conhecimento de um ou mais especialistas e o torna disponível para uma grande quantidade de usuários
        - Um motor de inferência atuando nesse conhecimento agiliza o procedimento de um usuário que não demandará a assimilação das regras armazenadas, levando a uma rápida atuação no problema
    - As bases de conhecimento permitem que boa parte do conhecimento dos profissionais seja "perpetuado" e fique disponível para futuros profissionais
        - As pessoas não são eternas e, em determinados segmentos profissionais, elas mudam constantemente de empresa, levando consigo seu conhecimento profissional e do negócio
    - O SE, além de seu uso em situações reais, pode ser utilizado para treinamento de nossos operadores da área de conhecimento do SE
- Além das vantagens de um SE, vale a pena considerar cenários em que sua atuação nem sempre é possível. Um SE é útil em problemas de domínio bem delimitado com disponibilidade de especialistas, dos quais se possa extrair e construir uma base de conhecimento. Em alguns problemas, no entanto, a extração do conhecimento de especialistas não é uma tarefa trivial, além do que se dispõe de uma grande quantidade de informações já tratadas pelos especialistas. Nesses casos, outras técnicas da IA podem ser aplicadas com maior agilidade e resultados satisfatórios
- Essas técnicas consistem na aplicação de algoritmos que identificam padrões nas informações e geram modelos que resolvem outros problemas baseados nos dados de entrada do sistema, "imitando" implicitamente o comportamento dos especialistas que tratam as informações

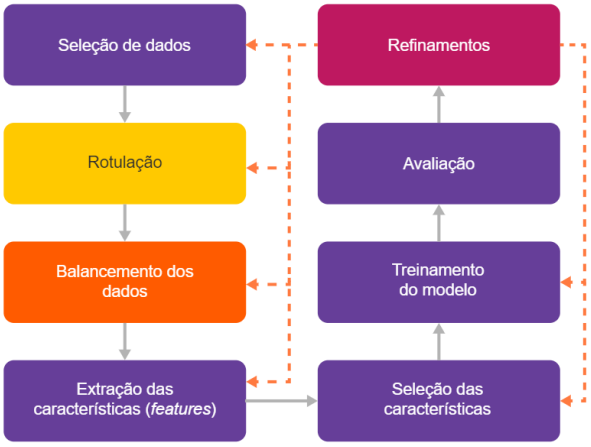
Aprendizagem de máquina

- *Diz-se que um programa de computador aprende pela **experiência E**, com respeito a algum tipo de **tarefa T** e **performance P**, se sua performance P na tarefa T melhora com a experiência E. - Mitchell, 1997*

Identificando padrões

- Em geral, quando se fala em aprendizado de máquina e seus diversos algoritmos de aprendizado, o que se objetiva é a identificação de padrões. Na realidade, poderíamos chamar aprendizado de máquina de aprendizado de padrões
- Os diversos algoritmos de aprendizado de máquina que serão apresentados utilizarão técnicas probabilísticas, matemáticas ou simbólicas (dentre outras) para identificar tais padrões e gerarão um modelo para ser utilizado em dados cuja resposta esperada não é conhecida. Esse modelo, portanto, imitará o comportamento dos humanos que definiram as respostas esperadas dos exemplos utilizados no treinamento do aprendizado automático e terão, assim, um comportamento "inteligente"

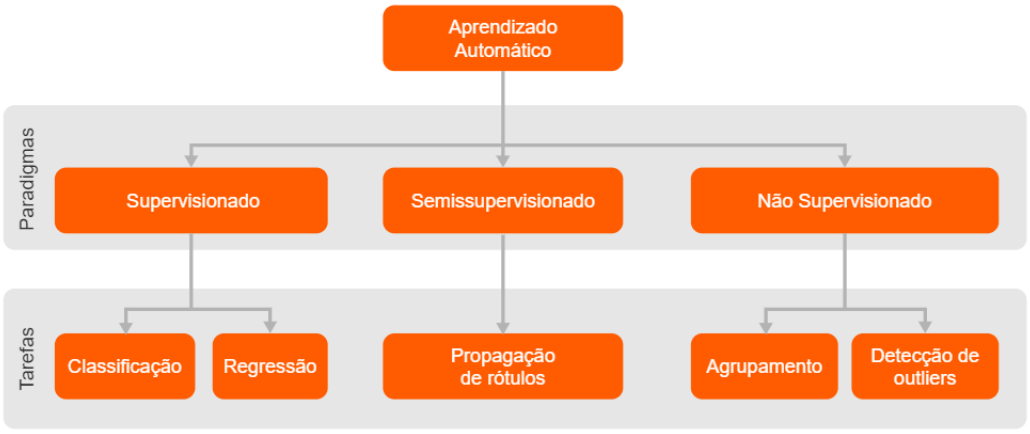
Principais etapas do Aprendizado Automático



- **Seleção dos dados**
  - É uma tarefa primordial, pois, quanto mais representativos do problema forem os dados, mais chances de se obter um modelo de aprendizado que reagirá bem a novos dados do problema
  - Se o objetivo é gerar um modelo de predição meteorológica, dados dos mais variados sensores (temperatura, pressão, velocidade do vento, etc.) devem ser reunidos do maior tempo de medição possível
- **Rotulação dos dados**
  - Consiste, basicamente, em indicar, para cada exemplo dos dados obtidos, qual a resposta esperada. O termo muito utilizado na área para a resposta esperada é **classe** (em tarefas de classificação, na taxonomia)
  - No exemplo dos dados meteorológicos, para cada conjunto de medições, deve-se dizer qual é a condição do tempo (classe) esperado, por exemplo ensolarado, chuvoso, nublado etc.
- **Balanceamento dos dados**
  - Consiste em balancear os dados obtidos para se ter quantidade suficiente de cada para se formular a resposta, balanceando, assim, a predição
  - No exemplo meteorológico, num conjunto de 10 mil exemplos, 4.9 mil são de tempo ensolarado, 5 mil de nublado e apenas 0,1 mil de chuvoso. Tem-se poucos exemplos de dados para predizer tempo chuvoso. Deve-se, portanto, balancear os dados para evitar que o

modelo aprenda e prediga mais as duas classes mais frequentes, errando muito a classe menos frequente
    - Pode ser feito por duas técnicas
        - *Oversampling*
            - Nesta técnica deve-se obter mais dados da(s) classe(s) minoritária(s), seja buscando mais dados, seja gerando dados sintéticos
        - *Undersampling*
            - Nesta técnica, exemplos das classes majoritárias devem ser dispensados, para ter proporção similar à(s) classe(s) minoritária(s)
            - Só deve ser aplicado caso o conjunto de dados tiver quantidade suficiente para dispensar dados sem perder performance no aprendizado
- **Extração de atributos**
    - Em algoritmos tradicionais de aprendizado de máquina é uma etapa essencial, pois são os atributos extraídos dos dados que permitem aos algoritmos de aprendizado gerar modelos para predizer as classes desejadas
    - No exemplo meteorológico, cada medida obtida por cada sensor pode ser considerada um atributo do problema. Nem sempre, no entanto, a extração de atributos é tão imediata. Considere o problema de detectar e-mails *spam*. Nesse caso, os atributos consistirão em pistas espalhadas pelo título e corpo do e-mail. Por exemplo, a ocorrência de determinadas palavras que indicam um *spam* (blacklist) ou palavras que indicam um e-mail confiável (whitelist) podem ser atributos dessa tarefa
    - Um outro passo, às vezes opcional, é a **seleção de atributos**. Nem sempre um atributo que escolhemos para resolver um problema é realmente importante para a solução, isso quando tal atributo não atrapalha a identificação da solução pelo algoritmo de aprendizado. Alguns algoritmos, como a árvore de decisão, tem a robustez de selecionar os atributos que melhor se adequam à solução do problema. Já outros algoritmos podem não ser tão robustos assim. A seleção de atributos é feita com base em alguma métrica, como correlação, ganho de informação ou entropia
- **Treinamento do modelo**
    - Nessa etapa, de acordo com os atributos e seus tipos, quantidade de dados, tipo de classes (discretas ou contínuas), dentre outras características, deve-se escolher o melhor algoritmo de aprendizado automático
    - Consiste na aplicação do algoritmo de treinamento nos dados que foram tratados nas etapas anteriores. Esses algoritmos, basicamente, encontrarão, para cada classe, o padrão que a indica. Posteriormente, com dados não vistos durante a etapa de treinamento, o modelo gerado poderá ser aplicado e as classes "identificadas"
    - Antes do treinamento, os dados devem ser separados em conjunto de treinamento (para efetivamente treinar o modelo) e teste. O conjunto de teste, que não foi utilizado durante o treinamento, deve ser utilizado na etapa de avaliação para a geração das métricas
- **Avaliação**
    - Nem sempre treinar um modelo é o fim da solução, uma etapa de avaliação se mostra essencial para definir uma *performance* para a solução
    - É feita utilizando-se algumas métricas como *precisão*, *cobertura*, *medida-F*, *acurácia*.

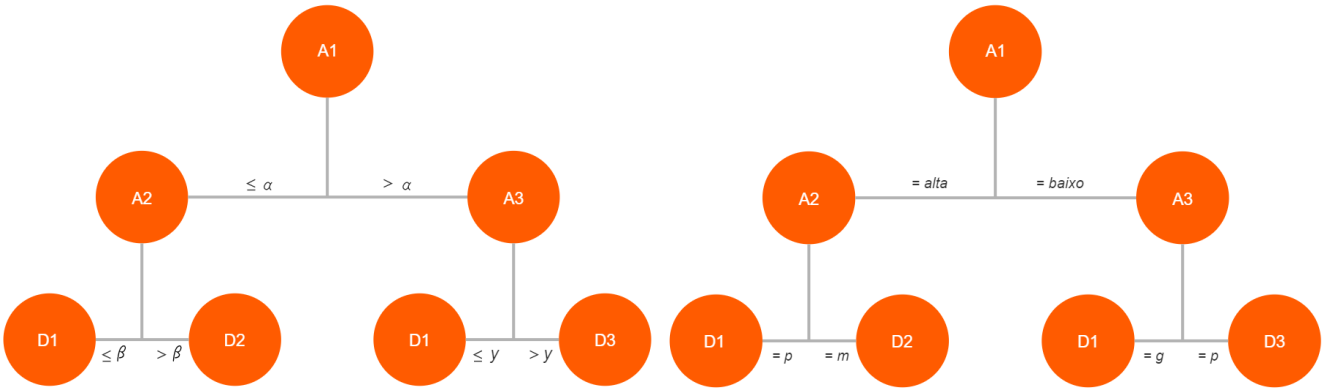
Taxonomia do aprendizado automático



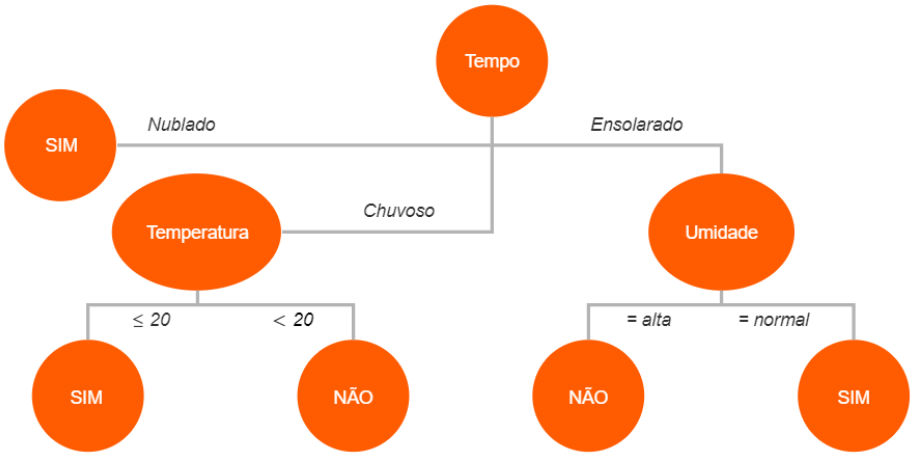
- As diversas formas de aprendizado podem ser categorizadas conforme a figura. Os três principais paradigmas de aprendizado são: **supervisionado**, **semissupervisionado** e **não supervisionado**. A grande diferença está nos dados durante o treinamento do modelo que encapsulará a "inteligência"

Paradigma de aprendizado Supervisionado

- Os dados servem como *instrutor* do algoritmo de treinamento, indicando para cada exemplo a resposta esperada
- Tem-se dois grandes grupos de tarefas: **classificação** e **regressão**. Na primeira tarefa objetiva-se, dadas as características de uma instância, definir uma classe (ou resposta esperada) dentre um número finito de possibilidades (conjunto discreto de valores). Na regressão, no entanto, esse conjunto de classes é infinito, como, por exemplo, um número real entre -1,0 e 1,0
- Principais algoritmos de Aprendizado supervisionado para a tarefa de **Classificação**
    - **Árvores de decisão**
        - Toma uma série de passos baseados nos valores de entrada até chegar a uma decisão
        - O processo de criação de uma árvore de decisão é também chamado de indução. Os algoritmos de indução objetivam a criação de Árvores que tenham a menor profundidade possível sem perder seu poder e acurácia de classificação. Isso garante, quando o conjunto de atributos é muito grande, que a decisão seja tomada com maior eficiência



- Na figura tem-se a ilustração de uma árvore de decisão que toma decisões (D1 a D3) sobre os valores de três atributos (A1 a A3)
- Para se tomar a decisão 2 (D2), o atributo A1 deve ter valor menor ou igual a α e A2 deve ser maior que β. Já a decisão 1 (D1) pode ser escolhida para qualquer valor de A1, mas A2 deve ser menor ou igual a β ou A3 deve ser menor ou igual a γ.



- A figura ilustra a árvore de decisão de sair para jogar tênis (SIM ou NÃO), considerando os atributos *aparência do tempo*, *temperatura* e *umidade*

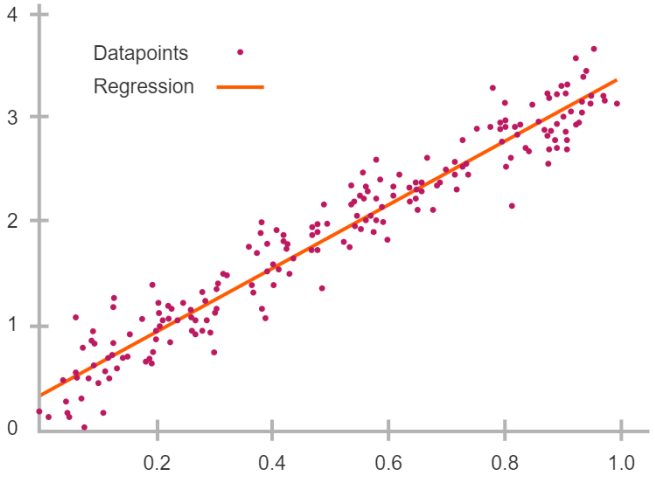
- ***Naive Bayes***

- **Support Vector Machines (SVM)**
  - É um paradigma matemático para a criação de classificadores
  - É uma das técnicas mais utilizadas em aprendizado supervisionado, por basicamente 3 razões:
  - *Uma ideia básica*
    - Generaliza bem para dados não vistos, pois define um separador de margem máxima em espaço multidimensional (cada atributo do problema define uma dimensão do espaço)
    - *Um truque hábil*
      - Embora crie uma separação linear no espaço multidimensional, pode utilizar **kernels** (truque hábil) que transformam o problema num espaço de dimensão superior. Assim, os problemas que não têm separação linear nos dados de entrada (atributos) podem ter separação linear de maior dimensão (gerado pelo kernel utilizado)
    - *Não é paramétrico* e armazena os dados de treinamento que são os mais indicativos para o separador de margem máxima

- Uma das limitações do SVM é que ele não trata diretamente de valores nominais, pois tem de gerar um espaço multidimensional, com cada atributo em um dos eixos desse espaço. Como representar valores nominais em um espaço numérico? Assim, problemas com valores nominais devem ter esses valores convertidos devidamente para valores numéricos. Há diversas técnicas de conversão, sendo a ***one-hot-encoding*** uma das mais utilizadas
- Considere o atributo tempo, que pode assumir três possíveis valores: *chuvoso*, *nublado* e *ensolarado*. Utilizando o *one-hot-encoding*, geraremos tantos atributos quantos forem os possíveis valores do atributo. Nesse caso, três. Cada um dos atributos gerados será binário, ou seja, assumirá apenas os valores 1 ou 0. Veja o exemplo em que, para representar um valor do atributo tempo utilizam-se três atributos numéricos (binários)

- Imagine um problema com 3 classes, para se decidir entre essas classes tem-se, por exemplo, duas formas de combinar classificadores binários SVM, a saber: **Um contra todos** ou **Um contra um**
- Vale salientar que a maioria das implementações SVM, em diversos pacotes de algoritmos de aprendizado de máquina, tratam automaticamente os problemas multiclasse

- Para a tarefa de **Regressão**
  - Regressão Linear
    - É uma técnica matemática que visa minimizar o erro de uma equação cuja representação geométrica se adapte a um conjunto de pontos. Esse conjunto de pontos são os exemplos do problema a ser solucionado
    - Em outras palavras, é a definição de uma equação que gera uma linha que melhor se adequa ao relacionamento entre os atributos de entrada X (dados de treinamento) e as classes Y (contínuas) do problema. Isso é feito obtendo-se coeficientes para os atributos de entrada X
      - Por exemplo, o preço de um produto Y (classe) é dado de acordo com seus atributos de entrada X (como nome, marca, modelo etc., que, por serem valores nominais, devem ser convertidos em valores numéricos)



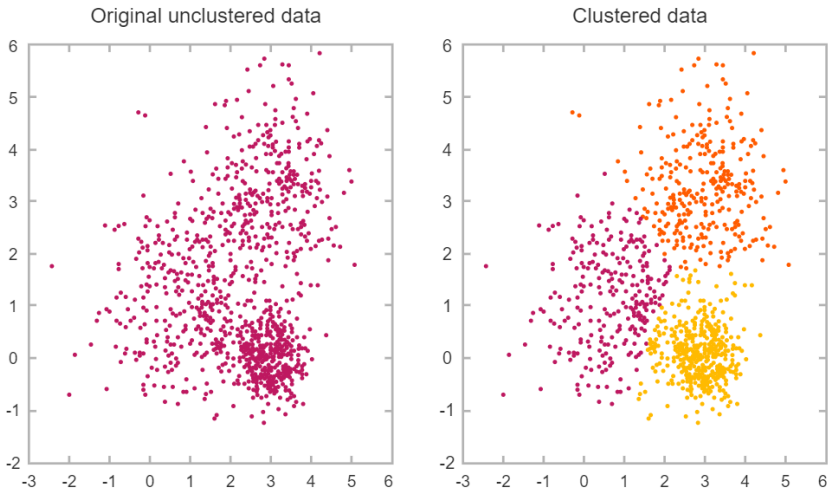
- $Y_i = \alpha + \beta X_i \Rightarrow$ desta forma, consegue-se um valor contínuo Y, de acordo com os valores de X. Essa equação define uma reta, como ilustrada na figura, em que os pontos vermelhos são as instâncias de treinamento da regressão. A reta é utilizada para, dado um valor de X, definir o valor Y (classe)

Paradigma de aprendizado Semissupervisionado

- Nessa abordagem há diversos métodos que tomam vantagem de dados anotados, mas não deixam de aproveitar também os dados que não estão anotados e, geralmente, se tem em abundância
- Quaisquer das tarefas mencionadas podem ser abordadas, iniciando tanto pela supervisão quanto pela não supervisão
- É indicada para os casos em que se tem uma parte dos dados rotulados e uma grande quantidade de dados não rotulados
- **Self-Training**
    - É um dos métodos de treinamento semissupervisionado
    - Utiliza os dados rotulados para treinar um modelo inicial. Esse modelo será, então, aplicado aos dados rotulados com a finalidade de gerar novos dados rotulados. Esses novos dados rotulados podem conter erros, obviamente, pois provêm de uma anotação (classificação) automática. Dessa forma, a escolha de um método de treinamento que indique a confiabilidade de suas classificações é muito importante, pois permite escolher apenas as anotações que forem feitas com alta confiabilidade. Os exemplos com alta confiabilidade podem se juntar aos exemplos gerados manualmente e um novo modelo será treinado com mais dados anotados

K-Means, um método de agrupamento

- É um dos métodos mais conhecidos do aprendizado semissupervisionado. Esse é um método de agrupamento que, aplicando alguma técnica de similaridade vetorial, busca identificar K grupos nas instâncias de treinamento



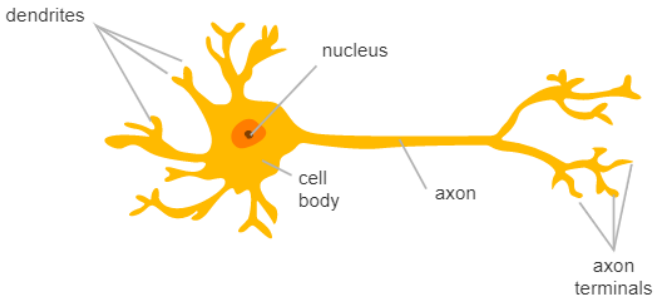
- A figura ilustra a aplicação de um algoritmo de agrupamento K-means, com K=3 (grupos roxo, amarelo e laranja à direita)
- Para definição de um grupo, o algoritmo escolhe um ponto que é chamado **centroide**. Definidos os centroides de cada grupo, um novo ponto (instância composta por atributos) é alocado no grupo a que estiver mais próximo, de acordo com a medida de similaridade escolhida
- Dentre as medidas de similaridade, podemos citar a distância **Euclidiana**, distância de **Manhattan** e distância de **Mahalanobis**
- Veja que aqui deve-se definir a quantidade de grupos (K). Há outras técnicas que não necessitam que se defina a quantidade, como os **métodos aglomerativos** ou **hierárquicos**

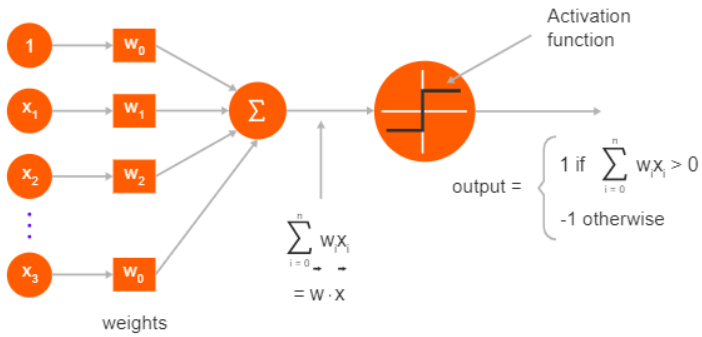
Paradigma de aprendizado Não Supervisionado

- Até agora, tratou-se de aprender a resolução de problemas quando se sabe quais são as possíveis respostas esperadas. Não podemos nos esquecer de uma classe de problemas em que não se sabe quais classes esperar, ou seja, não temos exemplos de solução do problema
    - Imagine um conjunto enorme de listas de compras realizadas em um supermercado. Será que podemos aprender alguma coisa com essa informação? Será que posso otimizar alguma coisa no supermercado, baseado no comportamento dos compradores, obtido de suas compras?
- É dito não supervisionado pois não se tem a definição ou anotação das classes do problema, mas ainda assim deseja-se aprender algo
- Podemos citar duas grandes tarefas: **associação** e **agrupamento**, dentre outras
- O algoritmo mais utilizado para esse paradigma é o das **Redes Neurais**

Redes Neurais (RN): Matematizando o biológico

- Basicamente, uma rede neural é composta por uma grande quantidade de neurônios artificiais, formalizados matematicamente. Um neurônio recebe (nos dendritos) e passa (ou não) impulsos elétricos (pelos axônios) no contato com outros neurônios. O que define se o sinal será repassado e com qual intensidade é o núcleo do neurônio





- Na formalização, cada neurônio tem n entradas (x_0 a x_n), e cada entrada é multiplicada (ponderada) por um peso w. Essas entradas ponderadas são somadas e uma função de ativação é aplicada, definindo a saída do neurônio, que será repassado para outros neurônios

- Há diversas funções de ativação. Na figura, uma função de ativação *degrau* foi utilizada (há também funções *sigmoide*, *tangente hiperbólica*, *softmax*, dentre outras). Nessa função, caso o somatório seja maior que zero, o sinal é repassado para o próximo neurônio. Caso o somatório seja menor ou igual a zero, o sinal é retido
- O aprendizado, na maioria das arquiteturas de redes neurais, é realizado no ajuste dos pesos aplicados às entradas x
    - O ajuste de pesos w é realizado no processo chamado **backpropagation**. Nesse processo, os atributos x são inseridos na rede e a resposta obtida pela rede é comparada à resposta esperada (no caso de um parendizado supervisionado). Caso a resposta seja diferente da resposta predita, os pesos são ajustados com o intuito de que a rede prediga a resposta esperada. Um dos métodos mais conhecidos e utilizados no backpropagation é conhecido como **gradiente descent** (gradiente descendente)
- Há diversas arquiteturas (formas de combinação de milhares, milhões ou bilhões de neurônios com o intuito de realizar o aprendizado automático) de redes neurais
    - Multilayer Perceptrons
        - Baseada no agrupamento de diversos Perceptrons em camadas, permite o tratamento de problemas não lineares. O Perceptron trata apenas de problemas lineares, isto é, cujas classes do problema sejam separáveis por um hiperplano no espaço de atributos
    - Redes Convolucionais
        - Uma arquitetura amplamente utilizada no tratamento de imagens
        - Os dados de entrada são tratados como matrizes numéricas tridimensionais cujas dimensões são alteradas a cada convolução (operação matemática entre duas funções para produzir uma terceira) na rede até gerar uma distribuição de probabilidades na camada de saída da rede, para tomada de uma decisão
    - Redes Recorrentes
        - Muito utilizada no tratamento de dados sequenciais, como sequência de palavras, sons e dados temporais, por exemplo
        - A predição de um dado é utilizada na predição do próximo dado
    - Long Short-Term Memory (LSTM)
        - Nessa, tem-se a introdução de unidades LSTM que ajudam a rede a aprender correlação "mais espaçadas" entre dados em uma série temporal
            - Esse espaço pode ser até mais de 1000 passos de distância na sequência dos dados
        - As LSTM são um avanço sobre as redes recorrentes para muitos problemas com longas séries temporais, como o Processamento da Linguagem Natural
    - Máquinas de Boltzmann
        - É um tipo de rede neural recorrente com uma característica mais estatística (estocástica). São úteis para resolver problemas combinatórios mais difíceis
    - Deep Belief Network
        - É uma opção para algumas limitações ao processo de aprendizado do *backpropagation*, tais como a necessidade de que os dados estejam todos rotulados (aprendizado supervisionado), tempo de aprendizado (muito demorado para redes muito profundas)
        - É uma opção de aprendizado não supervisionado em que os dados de entrada são agrupados. Os grupos aprendidos são as classes aprendidas
        - Um uso é o reconhecimento, agrupamento e geração de imagens, sequências de vídeos e dados de captura de movimento e o Processamento da Linguagem Natural
    - Deep Auto-Encoders
        - Arquitetura composta por duas redes simétricas, uma chamada de *encoder* e a outra, de *decoder*
        - São redes úteis para mapear uma representação em outra, como, por exemplo, mapear uma entrada textual em uma representação numérica

    - Generative Adversarial Network
        - É uma arquitetura bem profunda, composta por duas redes que são adversárias
        - Seu uso é no aprendizado da geração de alguma representação, seja imagem, texto ou música. Já imaginou uma rede que imite um compositor musical? Nessa arquitetura, em vez de prever uma classe Y com os dados de entrada X, ela tenta gerar os dados X para uma dada classe Y
- Essas arquiteturas podem ser combinadas para resolver os mais variados tipos de problemas, dependendo do tipo de dado a ser tratado, do tipo de predição (classes) etc.

Medidas de avaliação da aprendizagem

- Como verificar o quanto um método automático de aprendizado aprendeu ou está aprendendo? Como comparar dois modelos gerados para uma tarefa específica? Há diversas medidas de avaliação que servem para essa finalidade
- Considere Q a quantidade de instâncias disponíveis para realizar o aprendizado automático. Cada instância de Q tem uma classe y, que é a resposta esperada. Feito o aprendizado, cada instância q de Q terá uma classe predita (y_p) pelo modelo gerado. A classe predita pode ser igual (acerto) ou diferente da esperada (erro). Dadas essas definições, podemos definir as seguintes medidas de avaliação:
    - Precisão
        - $p = \frac{a}{Q_p}$
        - É o número de acertos (a) sobre a quantidade de instâncias preditas pelo modelo (Q_p)
        - Visto que nem sempre o modelo é capaz de predizer uma classe, nessa medida consideram-se apenas as respostas dadas pelo modelo gerado
    - Cobertura
        - $C = \frac{a}{Q_e}$
        - Número de acertos (a) sobre todas as predições esperadas pelo modelo (Q_e)
        - Nessa medida, diferentemente da precisão, considera-se como denominador todas as instâncias fornecidas na avaliação do modelo, mesmo que o modelo não consiga dar alguma predição para alguma delas
    - Medida-F
        - $F = \frac{2\,P\,C}{P+C}$
        - É uma média harmônica que combina precisão e cobertura. Veja que algumas tarefas podem prezar mais pela precisão que pela cobertura. Outras, o inverso
        - Para algumas tarefas, se o classificador der alguma resposta, é altamente desejável que essa resposta esteja correta. Então a precisão é mais importante que a cobertura. Para a maioria dos problemas, no entanto, é interessante um balanço entre a precisão e a cobertura, portanto, a medida-F é muito importante
    - Matriz de Confusão
        - É uma forma matricial de análise das respostas de um modelo. Na matriz de confusão, verifica-se qualitativamente o comportamento do classificador
        - Para isso, considere um modelo de classificação que distinga entre as classes *a*, *b*, *c*, e *d*. O modelo pode predizer todas as instâncias da classe *a* corretamente, mas pode predizer algumas instâncias da classe *a* como sendo da classe *b*. Esses erros são contabilizados nas medidas de P, C e F, mas ficam implícitos. Pela matriz de confusão é possível analisar essas “confusões” feitas pelo classificador

Esperada (ao lado) Predita (abaixo)	a	B	c	d
a	50	0	0	0
b	2	6	0	1
c	1	0	20	1
d	0	1	1	10

- A tabela ilustra uma matriz de confusão. Pela matriz, o modelo predisse corretamente todas as instâncias com classe esperada *a*. Já para a classe *b*, 6 instâncias estão corretas, 2 foram confundidas com a classe *a*, e 1 com a classe *d*. As da classe *c*, 20 estão corretas e 1 errada para a classe *a* e 1 errada para a classe *d*. E as da classe *d*, 10 estão corretas, 1 errada para a classe *b* e 1 errada para a classe *c*
- Os acertos estão na diagonal principal da matriz e os erros nas outras células

Representação da incerteza

Incerteza

- Diversas variáveis não podem ser observadas, por estarem além do escopo do problema que estamos tentando resolver com inteligência artificial. Assim, temos de recorrer a métodos que levem em consideração as incertezas que permeiam uma predição automática
- Imagine um sistema especialista que faça a triagem de paciente em um pronto-socorro. Sua base de conhecimento é composta de conhecimentos extraídos de especialistas em triagem e consiste basicamente de conjuntos de sintomas que indicam possíveis "quadros" dos pacientes. Um paciente que chegue ao pronto-socorro com um conjunto de sintomas C1 pode encaixar-se nos "quadros" Q1, Q2 e Q3. Por se tratar de uma triagem automática, o sistema especialista, mesmo não tendo certeza do quadro em que o paciente está, deve atribuir o paciente a um dos quadros. Então, como lidar com a incerteza? Qual o mecanismo que o sistema especialista deve adotar para escolher o mais promissor?

Lidando com a incerteza: *Probabilidades*

- Considere, por exemplo, um sistema especialista que auxilie um dentista em identificar problemas dentários. Um sintoma *dor_de_dente* pode indicar cáries, mas nem todos os pacientes estão com *dor_de_dente* por causa da cárie, pois pode ser por outras causas. Portanto, quanto menos características a serem observadas temos, menor a certeza que temos em uma previsão, ou classificação. No entanto, no exemplo anterior, suponhamos que 80% dos pacientes avaliados tinham o sintoma *dor_de_dente* devido a alguma cárie, então, lidamos com a incerteza de uma previsão baseada apenas nesse atributo com a afirmação: "O paciente com *dor_de_dente* tem cárie, com 80% de chances"
- Em problemas reais não baseamos uma afirmação em apenas uma característica, dessa forma, lançamos mão da **teoria da probabilidade** para atribuir um grau de confiança a cada afirmação feita. Esse grau de confiança, geralmente, está entre os valores 0 e 1

O Teorema de Bayes

- Em teoria da probabilidade, esse teorema descreve a probabilidade de um evento baseado na observação e quantização de conhecimentos anteriores relacionados ao evento
    - Em outras palavras, computa a probabilidade (ou grau de confiança) de uma afirmação dado um conjunto de observações
- O teorema de Bayes define o seguinte:
    - *P(H/E)* – a probabilidade de que a hipótese, ou proposição H, seja verdadeira dada a evidência E
    - *P(E/H)* – a probabilidade de que a evidência E será observada se a hipótese, ou proposição H, for verdadeira
    - *P(H|E)* – a probabilidade "a priori" de que a hipótese, ou proposição H, é verdadeira na ausência de qualquer evidência específica
- A probabilidade *P(A/B)* é dada pela lei da probabilidade total:
    - $P(A|B) = \frac{P(A|B) * P(A)}{P(B)}$
- Considere o seguinte exemplo:
    - Em um dado bairro, no caso do disparo de um alarme, deseja-se quantificar a chance de uma tentativa de roubo a uma casa. Para isso, tem-se as seguintes observações:
        - Quando há uma tentativa de roubo, em 95% dos casos, o alarme dispara;
        - Em apenas 1% dos casos, o alarme dispara por outros motivos;
        - No bairro em questão, há uma chance de 1 em 10000 (0.0001) de uma casa ser assaltada em determinado dia

- Podemos formular então:
  - $P_{(alarme|roubo)} = 0.95$, que é a probabilidade de um alarme disparar na tentativa de um roubo
  - $P_{(roubo)} = 0.0001$, probabilidade de um roubo
  - $P_{(alarme|não-roubo)} = 0.01$, probabilidade de um alarme por outros motivos
- Então:
  - $P_{(roubo|alarme)} = \frac{P_{(alarme|roubo)} * P_{(roubo)}}{P_{(alarme|não-roubo)}} = \frac{0.95*0.0001}{0.01} = 0.0095$
- Dessa forma, podemos dizer que as chances de uma tentativa de roubo, quando um alarme dispara, são de 0.95%, ou seja, menos de 1%. Isto ocorreu pois as chances de ocorrer uma tentativa de roubo no determinado bairro são muito pequenas (0.0001) em comparação com a probabilidade do alarme soar por qualquer outro motivo que não seja o roubo (0.01)

Agentes de sistemas

Processamento da Linguagem Natural (PLN)

- Visa capacitar os computadores a processar a linguagem natural, como o Português
  - Um compilador não fica na dúvida quando vê um comando, ele sabe exatamente o que fazer. Já uma frase em linguagem natural pode levar a uma ambiguidade de interpretação
    - "Eu vi o homem de binóculos.", quem estava de binóculos? "Eu" ou o "homem" que foi visto?
- É composto de diversas técnicas que possibilitam a interpretação textual de forma automática, desde o entendimento de cada palavra até porções maiores como sentenças, parágrafos e documentos
- Podemos citar diversas tarefas realizadas pelo PLN, tanto de análise, como de transformação e geração textual:
  - **Análise**
    - Etiquetação morfossintática
      - Identificação desambiguada das classes de palavras: substantivo, verbo, adjetivo etc.
    - Parsing Sintático
      - Identificação da estrutura das sentenças em sujeito, verbo, adjuntos etc.
    - Análise de Sentimentos
      - Se um trecho de texto tem um sentimento positivo, negativo ou neutro
    - Identificação de entidades
      - Identificação de locais, organizações, pessoas etc.
    - Parsing Discursivo
      - Identificação dos motivos que levaram o produtor do texto a dizer o que disse e da forma que disse
  - **Transformação**
    - Tradução automática
      - Transformação automática da língua de um texto para outra
    - Sumarização automática
      - Redução do tamanho do texto mantendo as informações mais importantes
  - **Geração textual**
    - Geralmente para colocar em linguagem natural algum conhecimento em alguma base de conhecimento, tal como é feito pelos chatbots mais avançados

Processamento de imagens

- O Processamento de Imagens processa um lote de informações (imagem) e pode detectar os limites de cada objeto presente, assim como classificar esses objetos
- Dentre as diversas tarefas do Processamento de Imagens, podemos citar:
    - Segmentação de imagens
        - Identificando os objetos presentes na imagem
    - Reconhecimento dos objetos Segmentados
        - Classificação dos objetos reconhecidos na imagem em “coisas” do mundo real
    - Reconstrução de cenas
        - Reconstrução de partes danificadas de uma ou mais imagens de um local, por exemplo
    - Restauração de imagens
        - Removendo ruídos, por exemplo