SISTEMAS INTELIGENTES

LISTA EXERCÍCIOS

2016.2

**LINK PARA O RESUMÃO:**

[**https://docs.google.com/document/d/1bN2Vxzxvui\_ouSNjBNXCR-Ew3BAyBOkMuUtLOQEQq1k/edit**](https://docs.google.com/document/d/1bN2Vxzxvui_ouSNjBNXCR-Ew3BAyBOkMuUtLOQEQq1k/edit)

**1. O que é aprendizagem de máquina? O que é necessário para poder aplicar este tipo de técnicas a um problema?**

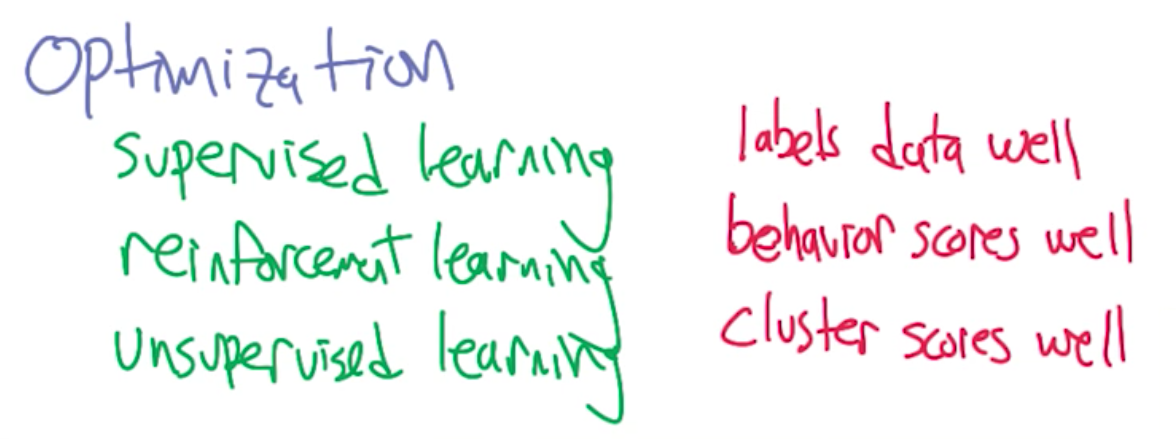
* É o campo de estudo da ciência da computação que dá aos computadores a habilidade de aprender sem serem explicitamente programados. A aprendizagem automática explora o estudo e construção de [algoritmos](https://pt.wikipedia.org/wiki/Algoritmo) que podem [aprender](https://pt.wikipedia.org/wiki/Aprendizagem) de seus erros e fazer previsões sobre [dados](https://pt.wikipedia.org/wiki/Dados). Tais algoritmos operam construindo um modelo a partir de inputs amostrais a fim de fazer previsões ou decisões guiadas pelos dados ao invés de simplesmente seguindo inflexíveis e estáticas [instruções programadas](https://pt.wikipedia.org/wiki/Programa%C3%A7%C3%A3o_de_computadores). Enquanto que na inteligência artificial existem dois tipos de [raciocínio](https://pt.wikipedia.org/wiki/Racioc%C3%ADnio) - o indutivo, que extrai regras e padrões de grandes conjuntos de dados, e o [dedutivo](https://pt.wikipedia.org/wiki/Dedu%C3%A7%C3%A3o), que faz uma análise lógica utilizada para construir argumentos, utilizando premissas/argumentos para obter uma conclusão. A conclusão torna explícito um conhecimento já existente nas premissas. - o aprendizado de máquina só se preocupa com o indutivo.
* Obviamente, é necessário ter principalmente três elementos para o aprendizado de máquina ser possível:
  + Existir um padrão
  + Não ser possível construir uma equação matemática
  + Existirem dados disponíveis
* Os componentes do aprendizado são:
  + Exemplos de treinamento (dados disponíveis por histórico)
  + conjunto de hipóteses (fórmulas candidatas e soluções possíveis)
  + função objetivo desconhecida
  + algoritmo de aprendizagem: utilizado sobre a base anterior para alcançar a hipótese final.
* Conceitos gerais do aprendizado
  + Exemplo: É uma tupla de valores de atributos.
  + Atributo: Descreve uma característica ou aspecto de um exemplo.
  + Classe: Atributo especial presente no aprendizado supervisionado. É o rótulo ou classe associada de cada exemplo.
* Classificador ou Hipótese
  + dado um conjunto de exemplos de treinamento, um indutor gera uma saída de classificador, também denominado hipótese ou descrição de conceito. A saída é dada de tal forma que, dado um novo exemplo, o classificador possa predizer com maior precisão sua possível classe associada.
* Ruído
  + Dados com imperfeições, provocadas por erros no experimento que os gerou ou por erros de digitação ao serem inseridos. Provocam classes rotuladas incorretamente.

RESPOSTA ALTERNATIVA (by: Lucas Rufino)

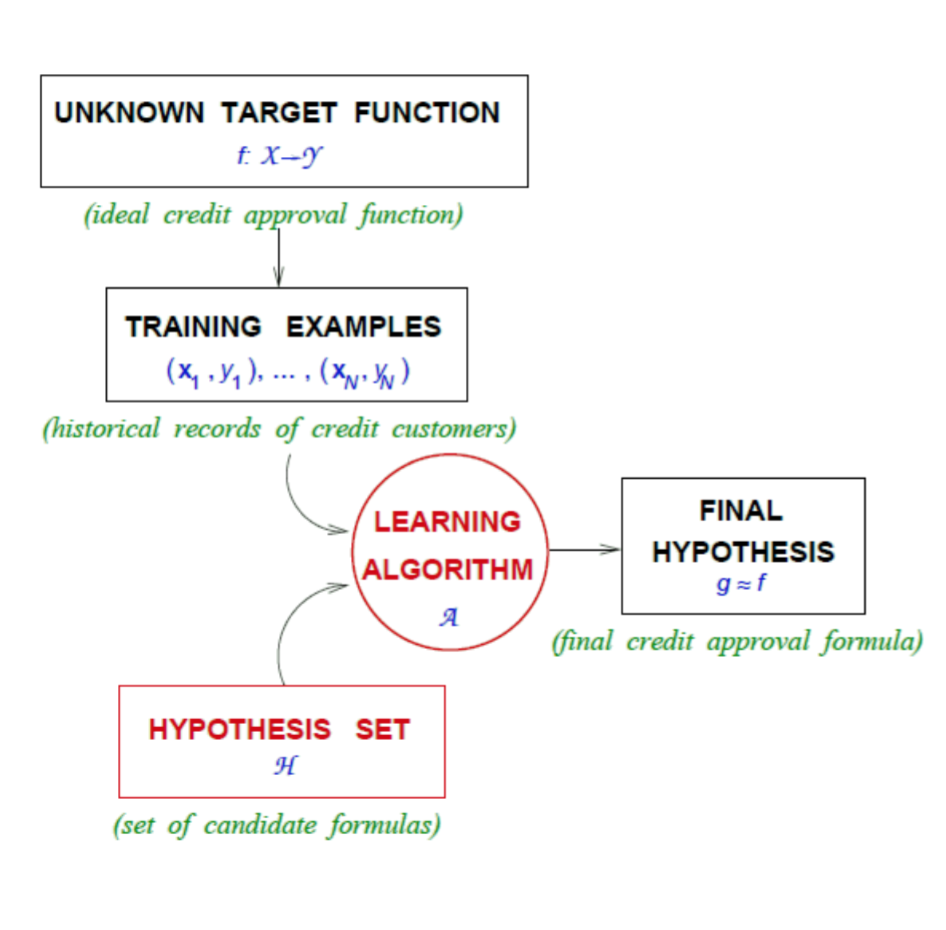
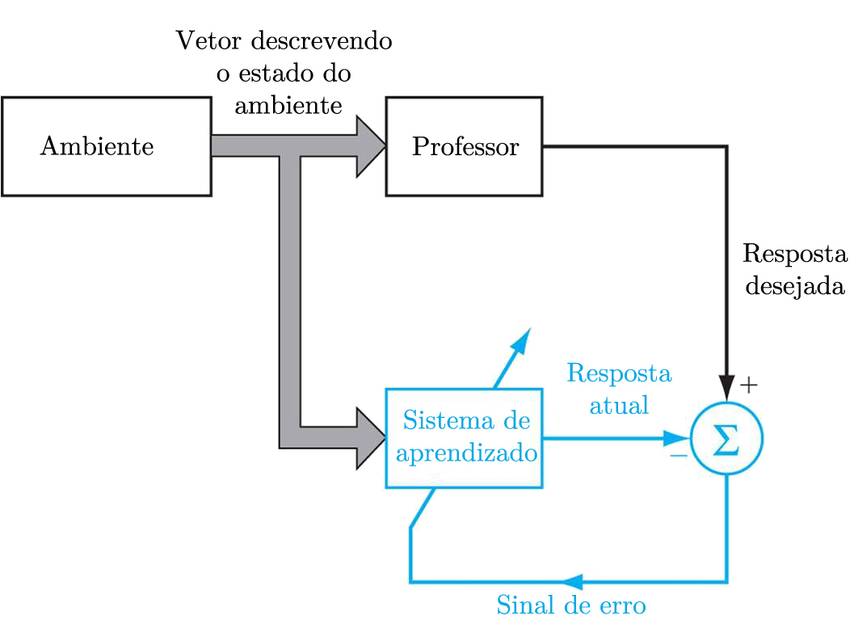
Na ciência da computação, a aprendizagem de máquina é a sub-área da inteligência artificial que busca a construção de programas de computador que melhoram seu desempenho por meio de experiência e/ou observação. Buscam resolver problemas de grande complexidade, seja em termos de número, variedade e natureza. São problemas para os quais ainda não existe solução algorítmica, mas existe conhecimento. A ideia é modelar um ser com comportamento inteligente (conhecimento, aprendizagem, iniciativa…), produzindo um sistema onde ele mesmo seja capaz de aprender o conhecimento necessário para solucioná-lo.

**2. Descreva o processo de aprendizado supervisionado e não supervisionado.**

* Supervisionado
  + Os rótulos das classes associadas aos dados de entrada são conhecidos
  + O objetivo do indutor é construir um classificador que seja capaz de determinar corretamente a classe de novos exemplos (entradas) ainda não rotulados
  + Para rótulos de classe discretos ou nominais, esse problema é chamado de classificação. Quando o rótulo das classes é contínuo, chamamos o problema de regressão.
* Não-supervisionado
  + Não há informações sobre os rótulos da entrada
  + O indutor analisa os exemplos fornecidos e tenta determinar se alguns deles podem ser agrupados de alguma maneira, formando agrupamentos ou clusters
  + Após a determinação dos agrupamentos, em geral, é necessária uma análise para determinar o que cada agrupamento significa no contexto do problema sendo resolvido, isto é, rotular os agrupamentos conforme o contexto do problema.



**3. Elabore um diagrama de blocos que ilustre o funcionamento do treinamento supervisionado.**

****

**4. O que significa mínimo local em treinamento supervisionado?**

* Quando um indutor está sendo treinado, ele possui como objetivo criar uma Região de Separação Linear, que nada mais é do que criar regiões bem separadas entre as saídas feitas a partir dos dados de treinamento. Para isso, ele deve classificar os dados de acordo com as classes, que são os tipos de dados disponíveis. Cada classe pode representar uma dimensão para o classificador.
* Para efetuar a separação das classes, o classificador deve utilizar uma Função de Mapeamento, na qual podem existir os chamados mínimos locais, que nada mais são do que, dado um intervalo da função, haverá um ponto que possuirá um valor mínimo que representa o menor erro de classificação que o classificador gera naquele dado intervalo.
* Teorema de Cover: Dependendo da quantidade de tipos de dados disponíveis, podem ser necessárias muitas dimensões para que seja possível efetuar uma separação das classes. Isto dificulta a determinação da função de mapeamento. O teorema de Cover diz que, dado um conjunto de dados de treinamento não-linearmente separáveis, podemos transformá-lo num conjunto linearmente separável projetando-o num espaço de dimensão maior, através de uma função de mapeamento.

**5. Por que existem arquiteturas distintas de redes neurais artificiais?**

* Dado que associados a cada entrada, temos um conjunto de pesos e quando a entrada supera certo limiar, a informação é propagada, e que o objetivo das redes neurais é conseguir generalização, queremos escolher um hiperplano que minimize a quantidade de erros na saída. Porém a maneira de determinação de tal hiperplano pode variar conforme o problema que estamos querendo resolver. Por isso a necessidade de diferentes arquiteturas para lidar com as diferentes classes de problemas.

**6. O que significa aprendizagem em redes neurais artificiais?**

* Significa obter generalização realizando o ajuste dos pesos da maneira correta, através da distância entre a saída desejada e saída obtida → que é o erro obtido. Tal ajuste é aplicado diretamente na função de ativação, que são propostas para conseguirmos classificar abordagens não-lineares, e permitir o tratamento de superfícies não-lineares.

**7. Quais as principais diferenças existentes entre os métodos de treinamento supervisionado e não-supervisionado.**

* No método supervisionado já temos dados com classes bem definidas de acordo com o contexto do problema, e o objetivo do indutor é somente determinar corretamente a classe de novos exemplos (entradas) ainda não rotulados. Em geral, está relacionado a problemas de reconhecimento de padrões (classificação) e de regressão (aproximação de funções). É aplicável para dados sequenciais (exemplo: reconhecimento de fala ou gesto), na forma de função que providencia um feedback contínuo da qualidade da solução até o momento atual.
* Já no método não supervisionado é necessário fazer mais trabalho: não temos informações sobre rótulos da entrada nem sobre os dados que temos na base. Logo, o indutor deverá formar agrupamentos dos dados de entrada recebidos através da similaridade com o que já existe na base, formando o que chamamos de clusters. Após isso, é necessária uma análise para determinar o significado de cada cluster para o contexto do problema abordado, e assim rotulá-los. Em geral, estão relacionados com problemas de estimação.

**8. Explique por que o Perceptron somente consegue classificar padrões cuja fronteira de separação entre as classes seja linear.**

* O Perceptron é um classificador binário e é considerado o tipo mais simples de rede neural artificial: um classificador linear. Associados a cada entrada, temos um conjunto de pesos. Quando a entrada supera certo limiar, a informação é propagada com o objetivo de conseguir generalização. Para isso, o Perceptron precisa determinar uma região de separação linear através de um hiperplano que minimize a quantidade de erros. Como o Perceptron é binário, a região de separação linear é uma reta. Para que seja possível determinar tal hiperplano, o Perceptron precisa estar lidando com um aprendizado supervisionado. Por isso a necessidade que a fronteira de separação seja linear.

A superfície de decisão num perceptron tem a forma de um hiperplano então as duas classes entre ela tem que ser linearmente separável, se não fossem não seria possível ser um hiperplano**.**

**9. Explique o que são situações de underfitting e overfitting, descrevendo- se também os meios para as suas detecções e as técnicas utilizadas para o seu contorno.**

Se a quantidade de pesos for pequena demais, pode haver *underfitting.*

A função implementada não tem complexidade suficiente para resolver o problema abordado. O ajuste dos pesos está sendo muito fraco e não está contribuindo para alcançar uma solução relevante.

Se a quantidade de pesos for grande demais, pode haver *overfitting.*

A função implementada tem complexidade demais para o problema, sendo capaz de modelar detalhes demais dos dados de treinamento. Ou seja, o ajuste dos pesos está acima do normal e causando diferenciações que não são relevantes para a solução geral do problema.

Usando taxa de aprendizado muito baixa, cada iteração faz um ajuste muito pequeno nos pesos (passo muito pequeno).

Pode precisar de muitas iterações para convergir para o ponto de mínimo desejado na superfície de busca.

Usando taxa de aprendizado muito alta, cada iteração faz um ajuste muito grande nos pesos (passo muito grande).

Pode causar oscilações em torno de um ponto de mínimo e entrar em loop.

Overfitting: Para detectar overfitting basta testarmos o conjunto de treinamento e o de teste e ver a taxa de erro de ambos. Nesta situação, a taxa de erro para o training set deverá ser muito baixa, enquanto, para o test set, deverá ser muito alta. Para resolvermos esse problema, podemos regularizar a função ou adicionar mais dados no training set, afim de variar o mesmo. Algumas maneiras de contornar overfitting é com o uso de validação cruzada, regularização ou early stopping (parada cedo).

Underfitting: A detecção de underfitting se dá da mesma forma do overfitting, porém a taxa de erro deve ser alta para ambos os casos. Para contornar esse problema podemos escolher outro algoritmo de aprendizagem ou modificar os parâmetros da função.

**10. Considere os aspectos da arquitetura neural, discorre sobre três diferenças entre redes MLP e RBF.**

* Sobre a rede RBF:
* Possuem basicamente três camadas:
  + entrada: modelada como um vetor de números reais
  + camada escondida (onde ficam as funções radiais de aproximação não-lineares)
  + saída: uma função escalar do vetor de entrada
* Para cada neurônio existe um vetor central ‘c’ e um peso associado a ele. Funções que dependem apenas da distância do vetor central são radialmente simétricas em relação a esse vetor, daí vem o nome função de base radial (RBF).
* Em sua forma mais básica todas as entradas estão conectadas a cada neurônio escondido(camada escondida). Em condições regulares, uma rede RBF com uma quantidade suficiente de neurônios escondidos pode aproximar qualquer função contínua com precisão arbitrária.

1. Redes MLP são bons classificadores quando queremos uma saída categórica para o problema abordado, pois a formação de regiões de separação lineares ocorre a partir da conjunção de várias retas. Utilizam produtos pontuais entre os pesos e as entradas do problema, validando através de funções de ativação sigmoidal. Já as redes RBF lidam com funções e valores reais utilizando Distância Euclidiana entre as entradas e pesos, que neste caso podem ser vistos como os vetores centrais, possibilitando tratar regiões e superfícies não-lineares que envolvem saídas mais complexas do que simplesmente diferenciação entre categorias.
2. Como consequência da diferença anterior, redes MLP somente lidam com treinamentos supervisionados e possuem uma função de ativação linear em todos os neurônios, enquanto as redes RBF lidam com os não-supervisionados, sendo capazes de lidar com aproximação de funções e problemas de estimação (possuem uma função de ativação não-linear)
3. O treinamento de redes RBF é feito utilizando um algoritmo dividido em duas etapas: No primeiro passo, os vetores centrais das funções RBF na camada escondida são escolhidos. Esse passo pode ser feito de várias formas, incluindo o uso de algoritmos de clusterização. Note que esse passo não é supervisionado. O segundo passo simplesmente encaixa um modelo linear com os coeficientes *Wi* para as saídas da camada escondida com respeito a alguma função objetivo. Já o treinamento de redes MLP é feito através do Backpropagation para todas as camadas e pode utilizar dois tipos de algoritmos:

Algoritmos Estáticos: Determina a arquitetura completa e retorna os pesos ideais a serem ajustados.

Algoritmos Construtivos: Baseado na qtd de entradas e saídas, vai adicionando os neurônios um de cada vez e verificando a taxa de erro

//resposta alternativa

1) MLP usa produtos de entradas com pesos, enquanto RBF usa distâncias euclidianas(entre entradas e pesos, que podem ser vistos como centros)

2) MLP usa uma função de ativação sigmoidal(ou outras funções monotônicas), enquanto RBF usa funções de ativação Gaussianas(que podem ser multivariadas).

3) MLPS podem possuir mais de uma hidden layer, enquanto RBF normalmente só usam uma.

**11. Explique o princípio de funcionamento do KNN.**

O KNN se baseia no princípio do “voto da maioria”. Para classificar um novo dado usando o K-NN, temos seu funcionamento baseado no seguinte princípio:

-> Tomamos os K vizinhos mais próximos do mesmo (normalmente usando distâncias euclidianas normalizadas)

-> Observamos a classe da maioria dos vizinhos. Essa será a classe do novo dado.

-> Caso haja empate entre as classes majoritárias, se observa a classe dos K-1 vizinhos mais próximos, se continuar havendo empate observa os K-2 vizinhos e assim por diante, até desempatar.

**12. Explique o princípio de operação do Simulated Annealing.**

O maior problema dos algoritmos gulosos de busca como o Hill Climbing é que eles ficam presos em ótimos locais. Para evitar isso, algumas técnicas como o Simulated Annealing foram criadas. Nesse algoritmo, é introduzido uma variável de temperatura T, para tentar simular um comportamento observado na termodinâmica. No algoritmo guloso, se estamos em um estado ‘X’, nós apenas consideramos ir para os ‘X’ vizinhos que possuem uma medida de desempenho maior do que X, o que nos leva a ficar presos em um ótimo local. Já no Simulated Annealing, existe uma probabilidade de escolhermos um ‘X’ que possua uma medida de desempenho pior. Essa probabilidade é proporcional ao valor da temperatura T. Ou seja, quanto mais ‘quente’, mais instável é o sistema, saltitando aleatoriamente entre os estados vizinhos. No fluxo de vida do algoritmo, começamos com um T alto, e aos poucos vamos resfriando o sistema, até que atingimos T = 0, ou seja, o algoritmo se torna apenas um guloso comum. Nesse momento, é provável que tenhamos encontrado um ótimo global.

**13. Explique o princípio de operação da Busca Tabu**

Busca Tabu é outra forma de evitar ótimos locais. Nela, o programa guarda uma memória, pequena ou grande, dos últimos N estados visitados, e o algoritmo é proibido de voltar para eles. Assim, por vezes ele é forçado a escolher caminhos que apresentem um desempenho local ruim, mas que podem salvá-lo de ficar preso em um ótimo local. Isso acontece em um vale, por exemplo, quando o guloso tende a ficar estacionado no ponto mais baixo, a Busca Tabu é obrigada a se mover, subindo o vale e saindo do mínimo local.

**14. Explique o funcionamento do processo de otimização via algoritmos genéticos.**

Os algoritmos genéticos se apoiam em conceitos biológicos e evolucionários. Cada estado é representado por um encoding, que representa o seu DNA e lista as suas características. É geralmente uma string ou array de bits. Existe também uma Fitting Function que avalia o quão saudável e promissor um estado é. Numa configuração inicial, é criado aleatoriamente um conjunto de estados, e, a cada geração, os estados mais bem avaliados são escolhidos para reproduzir, realizando crossovers entre seus encodings e gerando indivíduos cada vez mais especializados. Existe também um fator de mutação que ocorre nos estados individuais e dá uma certa estocacidade ao processo. Assim, ao longo das gerações, os indivíduos se aproximam cada vez mais do ótimo global.

**15. Compare as abordagens Simulated Annealing, Busca Tabu e Algoritmos Genéticos.**

Simulated Annealing:

Possui um fator Temperatura que influi na probabilidade do algoritmo decidir por um estado que possui uma medida de desempenho ruim localmente, na esperança de fugir de um ótimo local.

Busca Tabu:  
Guarda uma memória dos estados previamente visitados, evitando revisitá-los e forçando o algoritmo a ir por caminhos não explorados, mesmo que possuam uma medida local de desempenho pior.

Algoritmo Genético:  
O fator de mutação influi diretamente na escolha dos caminhos, aleatoriamente fugindo de ótimos locais. Além disso, os crossover’s naturalmente convergem para um ótimo global.

**16. Qual é o princípio por trás da navalha de Occam?**

O princípio por trás da navalha de Occam é o princípio do reducionismo.

**Reducionismo** é o nome dado a teorias correlatas que afirmam que objetos, fenômenos, teorias e significados complexos podem ser sempre reduzidos, a fim de explicá-los em suas partes constituintes mais simples.

De acordo com a navalha de Occam é preferível uma função simples que pode definir a maioria dos dados do que uma função completa. Se essa é sempre a melhor decisão a se tomar ainda é uma questão aberta, mas existem muitos argumentos a favor. Hipóteses mais simples tendem a ser mais facilmente construídas, ocupam menos espaço e, principalmente, parecem ter um poder maior de generalização do que as mais complexas.

**17. No contexto dos algoritmos genéticos, como o processo de busca para chegar a solução de um problema é realizada?**

Primeiro é gerada uma população de estados aleatoriamente. Depois, os estados passam por um processo incremental onde, a cada geração, os estados com a melhor função de desempenho são escolhidos para realizar um crossover, gerando filhos com características de ambos os pais e substituindo a população anterior. Esse processo é realizado ao longo de várias gerações, aliado a um fator de mutação que ocorre estocasticamente, convergindo a população cada vez mais para um ótimo global.

**18. No contexto de sistemas difusos, quais são as etapas de raciocínio envolvidas na solução de um dado problema?**

Sistemas Difusos dependem da teoria dos Conjuntos Difusos e da Lógica Difusa para serem implementados. Para entendê-los, precisamos definir alguns conceitos:

* Conjunto CRISP: Conjunto de dados tradicional. Os elementos abaixo são bem definidos
  + enumeração
  + relação bem definida
  + predicado
* Função de pertinência: função que é responsável por retornar se o elemento está contido ou não no conjunto
  + Pode ter valores graduais
    - Intervalo de transição: [0, 1]
    - Válido para conjuntos difusos
* Conjuntos difusos são uma generalização do CRISP
  + São definidas a forma de representação e a interpretação para os intervalos da função de pertinência
* a lógica difusa é um conjunto de técnicas e ferramentas para tratar incertezas da informação
  + Mais relacionado a maneiras de converter informações **QUALITATIVAS** em **quantitativas.**
  + Permitem valores verdade diferentes de 0 e 1
  + Permitem predicados mais diversificados
  + Qualificadores mais variados

Para que os sistemas difusos sejam implementados, é necessário utilizar a teoria dos conjuntos difusos (teoria base), formalizar os conceitos através da lógica difusa e implementar o sistema com o auxílio de Regras Difusas

* Regras difusas: regras do tipo se..então
  + Se <antecedente> então <consequente>
* Sistemas Difusos usam as regras difusas para raciocinar sobre os dados
  + Codificam o conhecimento de forma próxima à utilizada por especialistas
    - Regras + próximas da linguagem natural
    - Fácil manutenção
    - Simplicidade estrutural

As etapas são:  
Primeiro é feita uma fuzzificação dos dados de entrada em funções de pertinência.

Depois executam-se todas as regras de inferência aplicáveis para computar as funções de pertinência.

E, por último, é feita uma defuzzificação da saída das funções para obter seus valores “CRISP”

**19. No contexto de redes neurais artificiais, quais são as etapas de treinamento de uma rede MLP utilizando o algoritmo de treinamento Backpropagation?**

BackPropagation é um algoritmo de gradiente descendente, ou seja, utiliza informações de derivada.

Por isso, as funções de ativação devem ser contínuas e diferenciáveis (é o caso da sigmóide logística).

Objetivo:

Fazer “ajuste de pesos”, ou seja, escolher os pesos que geram as saídas mais corretas possíveis (menor erro) de forma iterativa.

Idéia geral:

A cada iteração, obter um erro cada vez menor para os dados de treinamento.

Cuidado:

Não permitir que a rede aprenda detalhes demais do conjunto de treinamento (overfitting).

Etapas de treinamento:

É recomendável que o treinamento seja interrompido quando o erro no conjunto de validação atingir um mínimo.

A partir deste ponto, supõe-se que a rede só aprenderia detalhes irrelevantes do conjunto de treinamento.

O erro para dados de treinamento seria cada vez menor, mas o erro para dados novos (validação) seria cada vez mais alto.

O algoritmo consiste basicamente das etapas:

- propagação positiva do sinal: durante este processo todos os pesos são mantidos fixos; e

- retropropagação do erro: durante este processo os pesos da rede são ajustados baseados na regra de correção de erro (distância entre saída obtida e saída desejada).

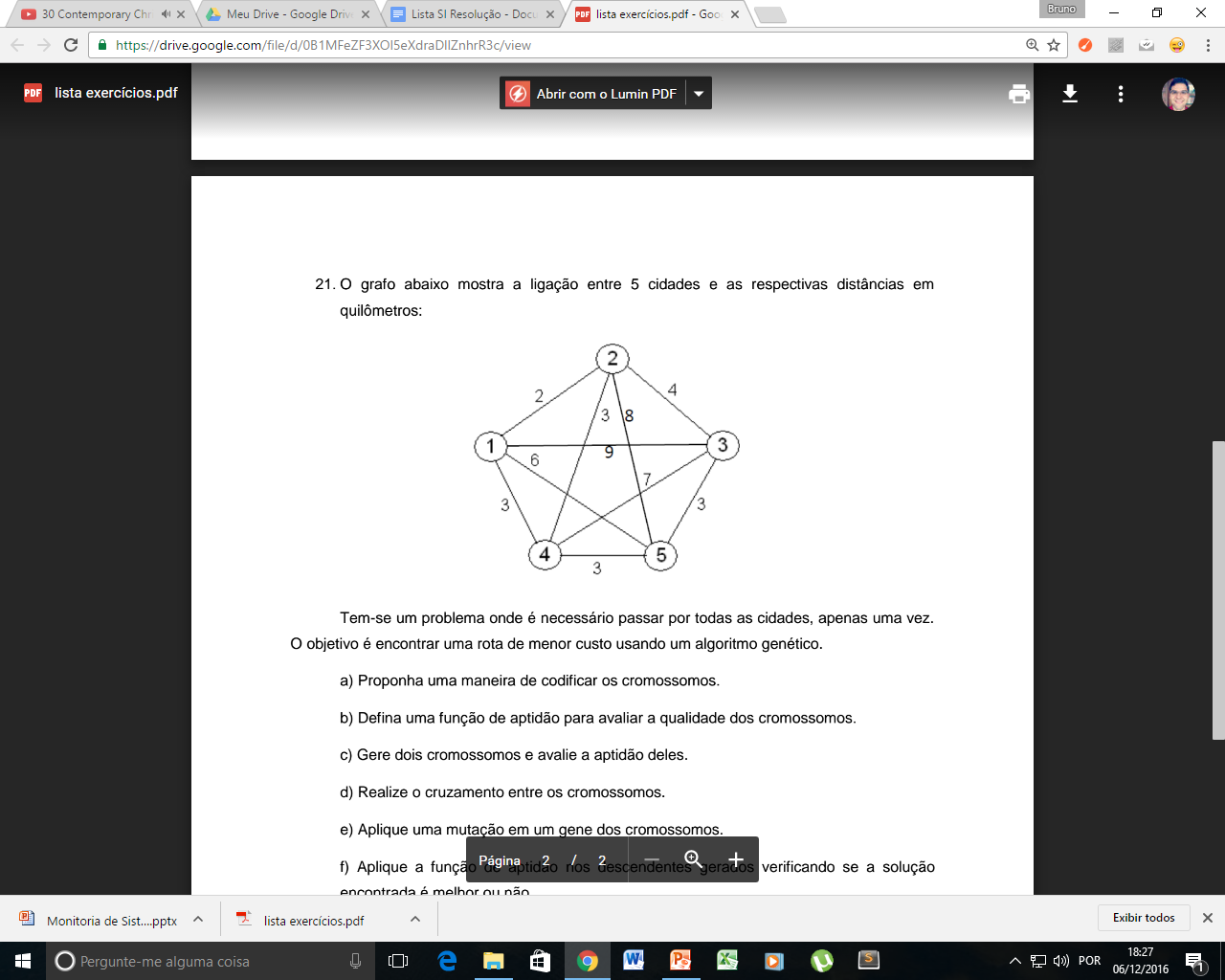
**20. Descreva o funcionamento do Algoritmo ID3 (Árvore de Decisão).**

Ele constrói a árvore em uma abordagem top-down considerando a questão: “Qual atributo é o mais importante e, portanto, deve ser colocado na raíz da árvore?”, para isso cada atributo é testado e sua capacidade para se tornar nó raíz avaliada.

Os atributos mais importantes são aqueles em que há maior ganho de informação. Para avaliar o ganho de informação, usa-se a fórmula E(S) = - p1\*log2(p1) - p2\*log2(p2) - ... - pn\*log2(pn), onde n é o número de valores possíveis, e pi é a fração de entradas com o valor i no total. O ganho de informação é maior em atributos com grandes quantidades de valores possíveis e distribuição mais uniforme das amostras para cada valor possível

Cria-se tantos nós filhos da raíz quantos valores possíveis esse atributo puder assumir (caso discreto). Repete-se o processo para cada nó filho da raíz e assim sucessivamente.

**21. O grafo abaixo mostra a ligação entre 5 cidades e as respectivas distâncias em quilômetros:**

****

**Tem-se um problema onde é necessário passar por todas as cidades, apenas uma vez.**

**O objetivo é encontrar uma rota de menor custo usando um algoritmo genético.**

**a) Proponha uma maneira de codificar os cromossomos.**

O encoding precisa representar um estado. Em um estado do TSP, nós precisamos saber quais cidades foram visitadas, e em que ordem. Todas as cidades precisam ter sido visitadas, e nenhuma pode ter sido visitada duas vezes. Portanto, o encoding pode ser apenas um array com uma permutação qualquer de [1, 2, 3, 4, 5].

**b) Defina uma função de aptidão para avaliar a qualidade dos cromossomos.**

O objetivo do TSP é encontrar a rota de menor distância. Portanto, essa será a nossa função de aptidão. Dada uma permutação de X = [1, 2, 3, 4, 5], F(X) deve retornar a soma das distâncias das arestas incidentes nos nós adjacentes.

**c) Gere dois cromossomos e avalie a aptidão deles.**

Cromossomo 1: [2, 4, 3, 1, 5] -> F([2, 4, 3, 1, 5]) = 3 + 7 + 9 + 6 = 25

Cromossomo 2: [2, 4, 1, 3, 5] -> F([2, 4, 1, 3, 5]) = 3 + 3 + 9 + 3 = 18

**d) Realize o cruzamento entre os cromossomos.**

Uma função de cruzamento tem que, dados dois cromossomos, gerar um filho com características de ambos. Porém, é preciso tomar cuidado para que o filho não fique com nenhum nó repetido, e nenhum nó faltando. Uma maneira de fazer isso é: Dados os encodings de dois cromossomos X e Y, primeiro escolha um subconjunto dos nós de X, e, depois, preencha os espaços faltando com os nós de Y que não estão em X, na mesma ordem.

Exemplo:

X = [1, 4, 3, 2, 5]

Y = [4, 3, 2, 1, 5]

Primeiro, eu escolho um subconjunto qualquer de X

Z = [\_, 4, 3, \_, \_]

Depois, preencho o que falta com os nós de Y que não estão em X

Z = [2, 4, 3, 1, 5]

**e) Aplique uma mutação em um gene dos cromossomos.**

Uma mutação precisa tomar cuidado com não incluir cidades a mais, e nem deixar nenhuma cidade faltando no cromossomo. Por isso, a mutação mais simples é apenas um swap em duas posições qualquer do cromossomo.

Seja X = [1, 2, 3, 4, 5]

Mutação(X) = [1, 2, 4, 3, 5]

**f) Aplique a função de aptidão nos descendentes gerados verificando se a solução encontrada é melhor ou não.**

Vamos escolher dois cromossomos X, Y aleatórios e aplicar um crossover neles para gerar um descendente Z.

X = [1, 4, 3, 2, 5] -> F(X) = 22

Y = [4, 3, 2, 1, 5] -> F(Y) = 20

Z = CROSS(X, Y)

Z = [2, 4, 3, 1, 5] -> F(Z) = 25

Assim, a função de aptidão do descendente não melhorou.

**22. Usando apenas o senso comum, defina algumas funções de pertinência para os seguintes conjuntos nebulosos.**

**a. n é grande**

A(x) = (valor de x - valor do menor elemento do conjunto) / (valor do maior elemento do conjunto - valor do menor elemento do conjunto)

**b. A média de uma variável aleatória é aproximadamente 5**

B(x) = {

1 - |µ(x) - 5|, se |µ(x) - 5| <= 1

0, se |µ(x) - 5| > 1

}

**c. x é muito maior que y**

C(x, y) = {

(x - y) / (valor do maior elemento do conjunto - valor do menor elemento do conjunto), se x > y

0, se x <= y

}

**d. O vento está forte**

//Considerando que vento é um vetor

d(vento) = valor de ||vento|| / valor máximo de ||vento||

**E. x está entre -3 e 2**

u(x) = {

0, se x < -4 ou x > 3

x + 4, se -4 < x < -3

1, se -3 <= x <= 2

3 - x, se 2 < x < 3

}