REVISÃO - Aplicações em Aprendizado de Máquina

Esta revisão está fortemente baseada na sequência de conteúdos tratados e discutidos nas videoaulas e nos textos-base que compõem o curso, e busca extrair e sintetizar os principais tópicos e conceitos apresentados ao longo do curso. Lembramos que somente a leitura deste material não é suficiente para o seu estudo. Ele deve ser usado mais fortemente como um guia para relembrá-lo(a) dos assuntos tratados. O estudo dos materiais-base/textos-base é de fundamental importância para que você possa ter um bom aproveitamento nas avaliações.

Bons estudos!

Aprendizado de Máquina
Na Semana 1, vimos que o aprendizado de máquina (AM) é um subconjunto da inteligência artificial que fornece aos sistemas a capacidade de aprender e melhorar automaticamente com a experiência, sem serem explicitamente programados. Vimos ainda que existem diversos algoritmos de AM, por exemplo redes neurais, e que a denominação de aprendizado profundo corresponde a redes neurais profundas (com muitas camadas).

Classificações do Aprendizado de Máquina
Vimos que o aprendizado pode ser classificado como:

- Aprendizado Supervisionado (Aprendizado Preditivo): com tarefas de classificação e tarefas de regressão.
- Aprendizado Não Supervisionado (Aprendizado Descritivo): com tarefas de associação, agrupamento e sumarização.

Conjunto de Dados
Antes que os algoritmos possam utilizar o conjunto de dados, muitas vezes é necessário que alguma adequação (pré-processamento) seja aplicada ao conjunto original. Dessa forma, uma análise exploratória inicial, utilizando medidas estatísticas e técnicas de visualização são importantes para a identificação de características do conjunto de dados para eventual aplicação de técnicas de pré-processamento.

Um conjunto de dados irá corresponder a um conjunto de objetos descritos por um conjunto de atributos, sendo que cada atributo está associado a uma propriedade do objeto.

Em tarefas descritivas, os dados não possuem um atributo-alvo (atributo utilizado para guiar o aprendizado, no aprendizado supervisionado), ao passo que, em tarefas preditivas, se o atributo-alvo assumir valores discretos (rótulos ou classes), tem-se um problema de classificação, ao passo que, se o atributo-alvo contém valores numéricos contínuos, tem-se um problema de regressão.

Tipos de Atributos
Os atributos podem ser qualitativos (número finito de símbolos ou nomes) ou quantitativos (normalmente valores binários, inteiros ou reais).

Em relação à escala, os atributos podem ser: nominais, ordinais, intervalares ou racionais.

Análise de Dados (Estatística Descritiva)
Uma grande quantidade de informações úteis pode ser extraída de um conjunto de dados por meio de medidas de estatística descritiva.

Algumas dessas medidas são:

- **Frequência:** busca medir a proporção de vezes que um atributo assume um dado valor em um determinado conjunto de dados;
- **Medidas de localidade:** definem pontos de referência nos dados (ex. moda, média, mediana, percentil);
- **Medidas de espalhamento:** buscam medir a dispersão dos valores (ex. intervalo, variância, desvio-padrão);
- **Medidas de distribuição:** buscam obter informações de como os valores se distribuem (e.g. obliquidade, curtose);
- **Medidas de similaridade:** buscam verificar relações de proporcionalidade e dependência entre atributos (e.g. correlação).

Análise de Dados (Visualização de Dados):

As medidas estatísticas podem ainda ser visualizadas através de ferramentas de visualização (ex. *Boxplot* para visualização de medidas de percentil, histograma para visualização de distribuições, *Scatter plot* para visualização de relação entre atributos).

Pré-Processamento
As técnicas de pré-processamento de dados são utilizadas para adequar os conteúdos dos conjuntos de dados, além de eliminar ou, ao menos, minimizar problemas encontrados em conjuntos de dados.

Algumas técnicas de pré-processamento estudadas:

- **Integração de dados:** quando se tem conjuntos provenientes de diferentes fontes;
- **Eliminação manual de atributos:** retirada de atributos pouco relevantes para o aprendizado;
- **Amostragem de dados:** visa reduzir do conjunto de dados preservando a estatística original;
- **Balanceamento de dados:** busca equilibrar o número de exemplos de cada classe no conjunto de dados;
- **Limpeza de dados:** busca corrigir ou, ainda, eliminar dados incompletos, dados inconsistentes, dados redundantes, dados com ruído.
- Transformação de dados
    - **Conversões:** simbólico-numérico, numérico-simbólico;
    - **Regularização de atributos:** visa acelerar o aprendizado;
    - **Redução de dimensionalidade:** utilizada em conjuntos de dados em que os objetos possuem um número muito elevado de atributos, como no caso de imagens (e.g. agregação, seleção de atributos).

Aprendizado Não Supervisionado (Modelos Descritivos)
No aprendizado descritivo, o algoritmo de AM não conta com nenhum elemento externo que auxilie ou guie o aprendizado (atributo-alvo). Dessa forma, os algoritmos buscarão identificar propriedades intrínsecas aos dados de entrada, de forma a realizar alguma tarefa, tais como: busca de padrões frequentes, agrupamento de objetos similares, ou ainda identificação de informações relevantes.

As tarefas de aprendizado descritivo podem ser divididas em:

- **Associação:** busca de padrões frequentes de associação entre elementos dentro do conjunto de dados;
- **Agrupamento:** busca identificar agrupamentos de dados de acordo com propriedades comuns entre os objetos;
- **Sumarização:** busca descrever os dados de uma forma simplificada (sintética).

Associação
A tarefa de associação está relacionada à mineração de conjuntos de itens frequentes, dentro do conjunto de dados.

Para a tarefa de mineração é utilizado o conceito de suporte, que corresponde ao número de vezes que um dado conjunto de itens (*itemset*), contido no conjunto de dados, aparece no conjunto de todas as transações realizadas com os itens pertencentes ao conjunto de dados. (Aqui você pode fazer o paralelo com a ideia de cesta de compras (conjunto de itens) apresentada na videoaula, e o número de compras (transações) em que aparece aquele conjunto de itens, dentre todas as compras realizadas em um estabelecimento).
A principal propriedade explorada pelos algoritmos de associação é que o suporte é monotonicamente decrescente com relação ao número de itens de um *itemset*. Ou seja, o conjunto de suporte de um conjunto de itens diminui sempre que se acrescenta um novo item. Os algoritmos de associação buscam primeiramente identificar os conjuntos de itens frequentes, e, em seguida, associar regras sob a forma $A \rightarrow B$, em que $A \cup B$ corresponde a um *itemset* frequente, de forma a determinar medidas de causalidade entre os itens de um determinado conjunto.

Algumas medidas utilizadas para avaliar as regras são:

- **Confiança:** consiste na probabilidade de ocorrer um conjunto de termos dado que ocorreu um outro conjunto;
- **Lift (coeficiente de interesse):** consiste no quociente entre a confiança e o valor esperado para a confiança;
- **Convicção**: pode ser interpretada como o quociente da frequência esperada de A ocorrer sem a ocorrência de B.

O algoritmo Apriory

O algoritmo Apriory é um dos mais populares algoritmos de associação, utilizando uma estratégia de busca em largura; em que a cada novo nível de busca (onde são formados novos *itemsets* resultantes da combinação dos *itemsets* frequentes do nível anterior) é realizada uma varredura em todos os *itemsets* formados, eliminando todos aqueles que não atendem ao critério de suporte mínimo. Esse processo é repetido até que todos os *itemsets* que atendam ao valor de suporte mínimo (*itemsets* frequentes) sejam formados. Uma vez identificados os *itemsets* frequentes, são aplicadas as regras associadas a cada *itemset*, sendo as regras então testadas para verificação de algum valor de confiança mínimo estabelecido.

Análise de Agrupamentos

As técnicas de agrupamento em conjunto de dados buscam encontrar estruturas de conjuntos de objetos similares (*clusters*) a partir de critérios que variam de acordo com o algoritmo empregado e com a definição de *cluster* utilizada. Assim, os *clusters* podem ser definidos como:

- ***Cluster bem separado:*** (critério de distância/semelhança em relação a pontos em outros *clusters*);
- ***Cluster baseado em centro*****:** (critério de distância em relação ao centro do *cluster*);
- ***Cluster contínuo ou encadeado***: (critério de vizinho mais próximo ou agrupamento transitivo);
- ***Cluster baseado em densidade*****:** (critério de densidade);
- ***Cluster baseado em similaridade*****:** (critério de similaridade entre os objetos).

Com base nas definições de *cluster*, os critérios de agrupamento utilizados pelos algoritmos podem ser divididos em três grandes categorias:

- **Compactação:** normalmente leva à formação de *clusters* esféricos ou bem separados;
- **Encadeamento ou ligação:** esse critério é bom para formas arbitrárias, mas é ruim quando há pouca separação espacial;
- **Separação Espacial:** considera apenas a distância entre *clusters*. Esse critério normalmente é utilizado em associação com os outros critérios por alguns algoritmos.

A análise de agrupamentos envolve diversas etapas que podem ser sumarizadas, como:

- pré-processamento de dados;

- determinação de medidas de similaridade/dissimilaridade (criação de matrizes de similaridade/dissimilaridade);
- formação dos agrupamentos utilizando algum algoritmo de agrupamento para identificação de possíveis estruturas de *clusters*;
- validação subjetiva (especialista)/validação objetiva (baseada em índices estatísticos);
- interpretação (rotulação dos *clusters*).

Algumas das medidas de similaridade/dissimilaridade utilizadas pelos algoritmos para formação das estruturas de *clusters* são:

- **similaridade:** cosseno, correlação de Pearson;
- **dissimilaridade:** distância de Manhattan, distância euclidiana, distância de Chebyschev, distância de Hamming.

Diferentes estruturas de *cluster* podem ser geradas dependendo do algoritmo utilizado. Assim, podemos ter:

- **algoritmos particionais:** tem como resultado uma única partição de dados (ex. algoritmo k-médias);
- **algoritmos hierárquicos:** tem como resultado uma sequência aninhada de partições (ex. algoritmos *single-link, average-link, complete-link*);
- **algoritmos baseados em densidade:** buscam formar cluster em regiões com alta densidade de objetos, podendo não formar partições (ex. algoritmo DBSCAN).

Algoritmo k-médias

O algoritmo k-médias é um algoritmo particional interativo baseado no erro quadrático. A cada iteração, o algoritmo busca formar *clusters* com base no cálculo dos centroides (calculados na iteração anterior) e na minimização da função custo (que consiste na distância ao quadrado
dos objetos em relação aos seus respectivos centroides), calculada a partir dos objetos associados a cada *cluster*. As iterações continuam, até que não haja mais migração de objetos de um *cluster* para outro entre duas iterações.

No algoritmo k-médias, o número de *clusters* "k" é um hiperparâmetro fornecido como entrada (Obs.: a determinação de "k", em muitas situações, não é uma tarefa trivial).

O algoritmo k-médias utiliza o critério de **compactação,** o que normalmente leva à formação de *clusters* esféricos ou bem separados.

Algoritmos de Agrupamento Hierárquicos

Os algoritmos hierárquicos têm como saída uma sequência de partições aninhadas, podendo ainda serem **aglomerativos** (começam com *n clusters* com um único objeto e formam uma a sequência de partições agrupando os *clusters* sucessivamente) ou **divisivos** (começam com um *cluster* contendo todos os objetos e formam a sequência de partições dividindo os *clusters* sucessivamente).

A hierarquia de partições formada pelos algoritmos hierárquicos pode ser representada por um dendograma, que consiste em uma estrutura na forma de uma árvore binária.

São exemplos de algoritmos hierárquicos aglomerativos:

- **single-link:** utiliza a distância mínima entre os objetos dos *clusters* com critério de distância mínima entre os *clusters.* Em geral, essa métrica favorece *clusters* finos e alongados de formato arbitrário;
- **complete-link:** utiliza a distância máxima entre os objetos dos *clusters* como critério de distância mínima entre os *clusters*. Em geral, essa métrica favorece *clusters* esféricos, sendo menos susceptível a ruídos e *outliers;*
- **average-link:** utiliza a distância média entre os pontos dos *clusters* com critério de distância mínima.

Como exemplo de algoritmo hierárquico divisivo, estudamos o Bicsecting k-means, baseado no algoritmo k-médias, com k=2.

Algoritmos Baseados em Densidade

Os algoritmos baseados em densidade assumem que os *clusters* são formados por regiões de alta densidade de objetos, circundadas por regiões com baixa densidade no espaço de objetos.

Os algoritmos baseados em densidade não são particionais, ou seja, nem todos os objetos precisam necessariamente pertencer a um *cluster* após o agrupamento.

Os algoritmos consideram que um objeto está em uma região de alta densidade de objetos se na sua vizinhança existem vários outros objetos. Disso resultam dois hiperparâmetros para formação dos *clusters:* o raio de vizinhança em relação ao objeto sob análise e o número mínimo de vizinhos para que seja considerada uma região densa.

Algoritmo DBSCAN

O algoritmo DBSCAN é um dos algoritmos mais populares baseados em densidade. Utilizando o raio de vizinhança e o número mínimo de vizinho como hiperparâmetros, o algoritmo DBSCAN define três tipos de objetos:

- **ponto de *core*:** um objeto é considerado um "ponto de *core*" se dentro da sua vizinhança existir um número de objetos maior ou igual ao número mínimo de vizinhos.
- **ponto de borda:** um objeto é considerado um "ponto de borda" se dentro da sua vizinhança existir um número de objetos inferior ao número mínimo de vizinhos e se esse objeto for vizinho de um "ponto de *core*".
- ***outlier*:** um objeto é considerado um "*outlier*" se ele não é nem "ponto de *core*" nem um "ponto de borda". Não irá pertencer, portanto, a nenhum *cluster*. Os pontos de *core* e os pontos de borda formam *clusters*, ao passo que os *outliers* são considerados ruído, não pertencendo a nenhum *cluster*.

Validação de Modelos Descritivos

A validação dos resultados de uma tarefa de agrupamento geralmente é baseada em índices estatísticos, que quantificam de alguma forma a qualidade de um dado agrupamento.

Existem basicamente três tipos de critério utilizados na validação de um agrupamento:

- **CRITÉRIOS RELATIVOS:** comparam diferentes resultados de agrupamento (obtidos por diferentes algoritmos) para determinar qual resultado melhor se ajusta aos dados ou ainda para determinar o valor mais apropriado para um ou mais parâmetros de um algoritmo, por exemplo o número de *clusters*.
- **CRITÉRIOS INTERNOS:** medem a qualidade (consistência) de um agrupamento com base **apenas nos dados originais** (matriz de objetos ou matriz de similaridade/dissimilaridade).
- **CRITÉRIOS EXTERNOS:** avaliam um agrupamento de acordo com alguma **estrutura conhecida** previamente (externa), como algum agrupamento desejado/conhecido para o conjunto de dados. Podem ainda refletir a intuição/conhecimento do especialista sobre a estrutura de dados.

Alguns índices utilizados no critério relativo

- **variância *intracluster*:** mede a qualidade do agrupamento em termos de compactação dos seus *clusters*.
- **índice Dunn:** utilizado para identificação de *clusters* compactos e bem separados.
- **conectividade:** mede o encadeamento dos clusters em uma partição, sendo uma medida do grau com que objetos vizinhos são agrupados em um mesmo *cluster*;
- **silhueta:** avalia a adequação de cada objeto para seu *cluster* e a qualidade de cada *cluster* individualmente.

Alguns índices utilizados no critério externo

- **índice Rand:** faz uma comparação par a par dos objetos entre duas partições para verificar se eles pertencem aos mesmos *clusters* ou a *clusters* diferentes em cada uma das partições;
- **índice Jaccard:** pode ser visto como uma variação do índice Rand. Utilizado quando o número de clusters é muito elevado;
- **ARI (índice Rand ajustado):** o índice passa a variar no intervalo [-1,1].

Algumas medidas utilizadas no critério interno

- **coeficiente de correlação cofenética:** correlação entre a matriz de distâncias (dissimilaridades) e a matriz cofenética (utilizado para validação de agrupamentos hierárquicos);
- Determinação do valor do hiperparâmetro "k" para o algoritmo k-médias, através da identificação do "joelho" da curva de erro quadrático X número de *clusters*.

Aprendizado Não Supervisionado (Modelos Preditivos)
No aprendizado preditivo é fornecido um conjunto de dados (entrada/saída) para treinamento, em que previamente é sabido qual deve ser o resultado correto de uma saída para uma dada entrada. Assim, um dos atributos dos objetos no conjunto de dados corresponde ao atributo-alvo (saída). Se o atributo-alvo contém valores numéricos contínuos, tem-se um problema de regressão (a tarefa é prever uma quantidade contínua). Quando os valores do atributo-alvo identificam classes ou categorias, então tem-se um problema de classificação (a tarefa é prever um rótulo discreto).

Os algoritmos preditivos irão induzir um modelo que representa um estimador. No caso de tarefas de classificação, o estimador é denominado classificador, em tarefas de regressão, o estimador é denominado regressor.

De maneira a induzir um modelo, os algoritmos buscam minimizar, ao longo do aprendizado, uma função custo; que representa uma medida do erro do estimador em relação aos valores corretos (valores dos atributos-alvo).

Métodos Baseados em Distância

Nesta metodologia, a premissa utilizada é a de que dados similares tendem a estar concentrados em uma mesma região no espaço de entrada. Dessa forma, algoritmos baseados em distância buscam associar a classificação de um novo objeto à classe de objetos que possuir maior semelhança (em termos de seus atributos) com o objeto a ser classificado.

Obs.: é comum o uso da distância euclidiana como medida de similaridade/dissimilaridade entre objetos.

Algoritmo k-NN

O algoritmo k-NN, baseado em “vizinhos mais próximos”, é um dos principais algoritmos baseados em distância. Ele é considerado um algoritmo “*Lazy*” (preguiçoso), pois não existe de fato um processo de aprendizado para indução de um modelo. Os objetos do conjunto de treinamento são simplesmente armazenados em memória, sendo que o processamento pesado é feito somente na etapa de inferência.
O valor "k" representa a quantidade de vizinhos mais próximos do objeto a ser classificado considerada pelo algoritmo. Esse algoritmo irá utilizar lógica majoritária para a classificação. Assim, definido um valor de k, o algoritmo irá verificar qual classe possui mais objetos próximos dentre os k objetos vizinhos verificados. Essa classe será atribuída ao objeto a ser classificado.

1-NN

O 1-NN é o caso mais simples do k-NN. Neste caso, verifica-se a qual classe pertence o

vizinho mais próximo. Essa classe é então atribuída ao objeto a ser classificado.

A superfície de decisão desenhada pelo algoritmo 1-NN é um conjunto de polítopos (e.g. poliedros, polígonos) convexos com centro em cada objeto do conjunto de treinamento. Qualquer objeto dentro da região de um polítopo estará mais próximo do objeto de treinamento

do que de qualquer outro objeto. O conjunto desses polítopos é designado diagrama de Voronoi. O diagrama de Voronoi também pode ser utilizado para definir a fronteira de decisão entre classes.

De forma geral, para um dado problema, a escolha de um valor de "k" a ser utilizado pelo algoritmo não é um problema trivial. No entanto, quando o conjunto de objetos não é muito grande, é possível utilizar "k" igual ao número de objetos do conjunto, e então calcular a contribuição ponderada de cada objeto (quanto mais distante for o objeto, menor é a sua contribuição (peso) para a decisão de classificação).

Métodos Probabilísticos
Na abordagem probabilista, os algoritmos lançam mão das relações de probabilidade entre classes e atributos para classificação de objetos.

Algoritmo Naive Bayes
O algoritmo Naive Bayes é um dos principais representantes de algoritmos baseados em métodos probabilísticos. Esse algoritmo faz uso das probabilidades: ***a priori***, ***a posteriori***, da **verossimilhança** e do uso do teorema de Bayes para indução do modelo preditivo.

- **probabilidade *a priori*:** corresponde à incerteza associada a um evento "A" antes da realização de um experimento, ou seja, é probabilidade associada a um evento em si (ocorrência de uma classe no conjunto de dados);
- **probabilidade *a posteriori*:** corresponde à probabilidade condicional que é atribuída à ocorrência de um evento (classe) depois que dados relevantes são observados (atributos);
- **verossimilhança:** corresponde à probabilidade condicional de observar um dado ou um conjunto de dados (atributos), dada a ocorrência de um evento (classe).

O algoritmo Naive Bayes, a partir da estimação da probabilidade *a priori*, da verossimilhança e da aplicação do teorema de Bayes, estima a probabilidade *a posteriori* na etapa de inferência. Esta última corresponde à probabilidade de uma determinada classe corresponder à classe de um determinado objeto (com um conjunto de atributos) no processo classificação. Assim, aplicando a regra MAP (Máximo *a Posteriori*), a classe que possuir maior probabilidade *a posteriori* para um dado objeto será selecionada para a classificação desse objeto.
Obs.: as probabilidades (verossimilhança e *a priori*) são estimadas diretamente a partir do conjunto de dados, através da abordagem frequencial (número de ocorrências do evento de interesse/número total de eventos no espaço de eventos).
O algoritmo Naive Bayes considera que todos os atributos de um determinado objeto são independentes entre si, essa premissa simplifica os cálculos realizados pelo algoritmo. Embora em muitas situações esta premissa não seja verdadeira, normalmente o algoritmo apresenta bons resultados de classificação.

Métodos Simbólicos - Árvores de Decisão
Nesta metodologia, o resultado do aprendizado é organizado em uma estrutura simbólica que permite descrever os padrões extraídos do conjunto de dados, fornecendo uma alta interpretabilidade. Um dos representantes dessa metodologia são os algoritmos de indução de árvores de decisão.

Árvores de Decisão
Árvores de decisão utilizam a estratégia “dividir para conquistar”, ou seja, um problema complexo é particionado de forma recursiva em problemas mais simples, utilizando sempre da mesma estratégia a cada novo particionamento.

Árvores de decisão remetem à ideia comum de uma árvore, sendo compostas por ramos e nós:

- **nó de decisão:** contém um teste condicional baseado no valor de um atributo. Como resultado do teste realizado no nó, podem partir dois ou mais ramos. Quando dos nós partem apenas dois ramos, a árvore é chamada de árvore binária;
- **nó-folha:** são os nós terminais da árvore e estão associados à classe final que será atribuída a um dado exemplo.

Indução da Árvore de Decisão
De forma a induzir o modelo que corresponde a uma árvore de decisão, partindo inicialmente do conjunto de dados, os algoritmos realizam sucessivos particionamentos utilizando algum **critério de particionamento**, até que algum **critério de parada** seja atingido.
Um dos principais critérios de particionamento é o ganho de informação. Assim, o algoritmo fará sucessivos particionamentos, e, a cada novo nó formado, será escolhido o atributo que maximiza o ganho de informação para realização do próximo particionamento. Na prática, isso corresponde a escolher o atributo cujo particionamento resulte em novos subconjuntos em que a entropia ponderada seja a menor possível, pois o ganho de informação corresponde à diferença entre a entropia do conjunto de exemplos e a soma ponderada da entropia dos subconjuntos gerados na partição.

A entropia pode ser entendida como o grau de mistura dos "rótulos" (classes) em um conjunto. Assim, se temos diversos exemplos com classes diferentes em um mesmo conjunto (nó), a entropia é alta, e isso não é bom para a classificação. Já na situação em que um conjunto possua predominantemente uma dada classe, ou apenas uma única classe, então a entropia é baixa, o que é desejável para a classificação. A entropia pode ser considerada como um critério de pureza de um conjunto (quanto menor a entropia maior a pureza).

Quando um nó atinge um critério de parada, então não são mais realizadas partições a partir daquele nó. A parada pode acontecer eventualmente quando se chegue a um nó que possua somente um objeto, ou ainda quando se atinja um determinado valor de pureza.

Muitas vezes, antes da etapa de inferência é realizada a poda da árvore, em que são suprimidos nós da árvore. A principal motivação para realização da poda é a melhora na generalização para a etapa da inferência. Pode-se realizar uma pré-poda quando a formação da árvore é finalizada antes que todas as partições possíveis sejam realizadas ou, ainda, um pós- poda, quando após a formação da árvore são retirados nós.

Métodos Bfaseados em Maximização de Margens

A ideia central por trás dos métodos de maximização de margens é determinar uma fronteira de separação entre diferentes classes, em um espaço de objetos, de forma a maximizar a distância de separação (margens) entre os exemplos das diferentes classes.

SVM

As SVM (*Support Vector Machine*), ou máquinas de vetores de suporte, são algoritmos baseados em maximização de margens.
Os SVMs podem ser divididos em:

- **SVM lineares de margens rígidas:** Neste caso, as classes são separáveis através de um hiperplano que define a fronteira de decisão (classificação). Dessa forma, as classes devem ser completamente separáveis, não sendo permitida a existência de nenhum exemplo (objeto) do conjunto de treinamento no interior das margens do hiperplano.
- **SVM lineares de margens suaves:** Em diversas situações práticas ocorre a presença de ruídos ou *outliers* nos conjuntos de dados, o que causaria um problema de sobreajustamento para o SVM de margens rígidas. Assim, é possível relaxar as restrições impostas ao problema de otimização utilizando o SVM com margens suaves, permitindo que alguns pontos invadam as fronteiras.
- **SVM não lineares:** Nas situações em que não é possível utilizar um hiperplano para a separação das classes, é utilizado o SVM não linear. Nesse caso, mapeia-se o conjunto de dados para um novo espaço de maior dimensão, denominado espaço de características (*feature space*), em que as classes sejam linearmente separáveis. Realiza- se então a classificação usando um SVM linear, e depois retorna-se para a dimensão original. Na prática é utilizado o que é chamado de "*kernel trick*", que consiste no uso de uma função não linear ("*kernel*"), permite operar no espaço original sem a necessidade de computar as coordenadas dos dados em um espaço de dimensão superior, reduzindo assim o esforço computacional.

Avaliação de Modelos Preditivos

A etapa de avaliação de modelos preditivos busca mais fortemente avaliar na etapa de inferência, a capacidade de generalização dos modelos induzidos. Para isso, diversas medidas/ferramentas podem ser utilizadas, dentre elas:

- **Acurácia:** corresponde à taxa de acertos do modelo (número de acertos/número total de classificações realizadas);
- **Taxa de erro:** corresponde à taxa de erros do modelo (número de erros/número total de classificações realizadas);
- **Matriz de Confusão:** relaciona de forma matricial os rótulos verdadeiros (classe à qual o exemplo pertence) com a classe predita (classificação do exemplo realizada pelo modelo). Assim, é possível analisar visualmente os acertos e os erros cometidos por um modelo na classificação.
  Através da matriz de confusão na forma binária (possui apenas duas classes: positiva e negativa) é possível extrair outras medidas:
  - **Precisão:** proporção dos exemplos que foram corretamente classificados como positivos em relação às classificações positivas.
  - **Sensibilidade (revocação):** proporção dos exemplos que foram corretamente classificados como positivos em relação a todos os exemplos que são de fato positivos.
  - **Especificidade:** proporção dos exemplos que foram corretamente classificados como negativos em relação a todos os exemplos que são de fato negativos.
- **Análise ROC:** Compara de forma gráfica a taxa de verdadeiros positivos (revocação) *versus* a taxa de falsos positivos (taxa de erro da classe negativa). Essa ferramenta permite avaliar a qualidade na predição de modelos, além de permitir a comparação no desempenho entre modelos.

Divisão dos conjuntos de treinamento e teste

Nos modelos preditivos, o conjunto de dados normalmente é dividido em: conjunto de dados de treinamento e conjunto de dados de teste. O primeiro sendo utilizado na etapa de treinamento, e o segundo para avaliação do modelo na etapa de inferência. Existem diversas técnicas e critérios para criação dos conjuntos de treinamento e teste. Algumas delas, estudadas no curso, são:

- ***Holdout*:** esta é a técnica mais simples. Consiste basicamente em dividir o conjunto de dados sem nenhum critério específico, em uma proporção p, 1-p entre o conjunto de treinamento e teste respectivamente (é comum o uso de p=2/3).
- **Validação cruzada:** o conjunto de dados é dividido em ***r*** subconjuntos aproximadamente iguais, e são formadas combinações com ***r-1*** desses subconjuntos para formar cada um dos conjuntos de treinamento. O subconjunto que não é utilizado em uma dada combinação é usado como

conjunto de teste daquele conjunto de treinamento formado. É considerada a média dos desempenhos obtidos nos resultados de classificação dos conjuntos de teste.

- **Amostragem aleatória:** nesta técnica, diversos subconjuntos de treinamento e teste (sem repetição de objetos nos subconjuntos) são formados utilizando amostragem aleatória dos exemplos do conjunto original de dados. É considerada a média dos desempenhos obtidos nos resultados de classificação dos conjuntos de teste.
- ***Bootstrap*:** é uma amostragem aleatória que permite a repetição de exemplos dentro dos conjuntos formados. Para a formação de um subconjunto de ***n*** elementos são feitas ***n*** amostragens aleatórias no conjunto de dados. Ao final da amostragem, os dados que não fizeram parte do subconjunto de treinamento formarão o conjunto de testes. O *Bootstrap* é utilizado normalmente quando se tem um conjunto de dados pequeno e em situações bastante específicas. É considerada a média dos desempenhos obtidos nos resultados de classificação dos conjuntos de teste.

Métodos Conexionistas - Redes Neurais Artificiais (RNAs)

Os métodos conexionistas buscam de certa forma imitar o cérebro humano em suas características, com unidades de processamento básico denominadas neurônios, interconectadas em uma arquitetura que normalmente apresenta várias camadas (*layers*).

Um neurônio pode ser entendido como uma unidade básica de processamento com duas etapas de processamento:

- **primeira etapa:** É realizada a soma ponderada dos valores das entradas, multiplicadas pelos pesos internos do neurônio somado a um valor de *bias*;
- **segunda etapa:** É aplicada uma função de ativação ao resultado da etapa anterior, produzindo a saída do neurônio.

A existência de várias camadas interconectadas permite a definição de regiões arbitrárias de decisão pelo modelo induzido, para fins de classificação. De uma forma geral, podemos dizer que:

- Na primeira camada, cada neurônio aprende uma função que define um hiperplano;
- Na segunda camada, cada neurônio combina um conjunto de hiperplanos definidos pela camada anterior, formando regiões convexas;
- A partir da terceira camada os neurônios combinam um conjunto de regiões convexas em regiões de decisão de formato arbitrário.

Treinamento da rede neural
Durante o treinamento de uma rede neural, os cálculos são realizados em dois sentidos: sentido entrada-saída (*forward propagation*), quando é calculada a

estimativa das saídas para os exemplos utilizados como entradas para o treinamento, e calculada a função custo, que corresponde ao erro médio das

estimações em relação aos rótulos corretos fornecidos . Em seguida, o cálculo é feito no sentido saída-entrada (*backward propagation*). Nesta etapa os pesos internos dos neurônios e os valores de *bias* são atualizados. Um dos algoritmos mais utilizados para a atualização dos parâmetros da rede é o do Gradiente Descendente, que utiliza o cálculo de derivadas parciais para a realização das atualizações.

Funções de Ativação
Existem diferentes funções de ativação que podem ser utilizadas nos neurônios, nos diferentes *layers* de uma rede neural, dentre as mais utilizadas estão:

- **sigmoid:** Normalmente utilizada no *output layer* em redes de classificação binária. Normalmente não é utilizada nos *layers* internos por apresentar *offset*, e baixa taxa de variação para valores elevados do argumento, o que pode levar a uma lentidão no processo de convergência do algoritmo durante a etapa de treinamento.
- **tanh:** Pode ser vista como uma variação da função sigmoid, não apresentando o problema de *offset*. Porém para valores elevados do argumento, também apresenta uma baixa taxa de variação.
- **Relu:** É a função normalmente utilizada nos *layers* internos, por não apresentar problemas em relação à taxa de variação em função dos valores do argumento, a velocidade do aprendizado normalmente será bem maior do que com o uso da tanh ou da sigmoid.

Redes Convolucionais - CNNs
As CNNs são fortemente empregadas em problemas que envolvem a manipulação de imagens (classificação de imagens, reconhecimento de objetos etc.). Como as imagens normalmente possuem um número muito elevado de atributos, o uso de redes neurais convencionais, utilizando apenas neurônios, torna-se proibitivo

computacionalmente. Assim, nas CNNs os dados são tratados de forma multidimensional, através do uso de matrizes.

Uma CNN normalmente é formada por diferentes tipos de *layers*:

- ***Convolutional layer*:** As matrizes de entrada do *layer* são convoluídas (matematicamente é realizada uma operação de correlação cruzada) com filtros internos do *layer*, que consistem em matrizes de menor dimensão que são deslizadas sobre as matrizes de entrada, realizando o somatório dos produtos elemento a elemento, entre os elementos dos filtros e os elementos das matrizes de entrada em cada posição. Ao resultado de cada somatório é então adicionado um valor de *bias* e aplicada uma função de ativação. Como resultado são formadas novas matrizes na saída do *layer* (Obs.: normalmente as matrizes de saída terão menor dimensão que as matrizes de entrada);
- ***Polling layer:*** Embora seja possível o uso de *averagepooling* (extração da média), normalmente é utilizado o *maxpooling*. O *maxpooling* é realizado de forma similar à convolução, só que neste caso, o filtro apenas extrai o maior valor presente (máximo) na região da matriz onde ele se encontra durante o processo de deslizamento (há redução da dimensionalidade).
- ***Fully connected layer:*** Um *fully connected layer* consiste em um *layer* de uma rede neural convencional. Normalmente, as CNNs possuem os seus últimos *layers* na forma de *fully connected layers*. O que é feito na prática é uma operação de "*flatten*" no último "volume" da CNN. Esse volume é transformado em um vetor de neurônios, constituindo o *fully connected layer*, que então poderá ser, por exemplo, seguido de um neurônio na saída, sendo uma rede de classificação binária, ou de um *softmax layer,* no caso de classificação multiclasse.
- ***Softmax layer:*** Se tivermos um problema de classificação multiclasse com *M* classes distintas para a classificação, então normalmente se usa como último *layer* um *softmax layer*. O *layer softmax* terá *M* neurônios, e a função de ativação aplicada aos neurônios pode ser vista como uma estimativa das probabilidades da presença de cada uma das classes para uma dada imagem após o processamento da rede.

Algumas operações ainda podem estar presentes nos *layers* convolucionais como forma a ajustar as dimensões das matrizes de saída. Como o uso da técnica de *stride* para diminuir as dimensões das matrizes na saída do *layer* ou o uso do *padding* para aumentar as dimensões das matrizes na saída do *layer.*

As CNNs normalmente estão associadas ao aprendizado profundo (*deep learning*), apresentando-se na forma de redes profundas, podendo chegar a ter mais de 100 *layers*. Algumas arquiteturas atuais dessas redes são:

- ***Resnet (Residual Network)*:** Utilizam um bloco básico chamado de bloco residual ou *residual block,* que possuem os chamados "*skip connections*" que atuam como um gatilho no caso de os pesos da rede começarem a tender a zero, reduzindo nesses casos a função do bloco à função identidade.
- ***InceptionNet (Inception Network):*** São baseadas em duas ideias fundamentais: o uso de filtros com dimensão 1X1 para a operação de convolução e a paralelização e posterior concatenação de diversas operações normalmente feitas de forma sequencial em uma rede convolucional.

Aplicações em Aprendizado de Máquina

Muitos dos algoritmos estudados ao longo do curso são utilizados em diferentes áreas em tarefas do dia a dia. Vimos alguns exemplos:

- **Análise de sentimento:** a análise de sentimento está associada à identificação se uma opinião é positiva, negativa ou neutra em uma determinada avaliação. Muitas vezes a opinião está na forma de texto, sendo necessária uma etapa de pré-processamento. Normalmente os textos são tratados como uma *bag-of-words*, em que palavras e símbolos que não trazem informação para o aprendizado (*stop-words*) são extraídos, e as palavras de interesse, transformadas em *tokens* que representarão atributos no conjunto de dados formado. Assim, a presença de determinados *tokens* em um texto pode fornecer indicativos de uma opinião positiva, negativa ou neutra. Exemplos de algoritmos utilizados na análise de sentimentos são: Naive Bayes, SVMs e RNAs.
- **Filtragem de mensagens indesejadas:** existem atualmente diversas mídias nas quais pode ocorrer a presença de mensagens indesejadas, na forma de: *spam*, comentários ofensivos ou ainda *fake news*. No caso dos *e-mails*, os *spams* estão normalmente associados à venda de produtos e serviços. Assim, de forma similar ao que é feito na análise de sentimento, os algoritmos podem buscar por palavras (*tokens*) associadas a conteúdos de *spam*. Já no caso de notícias falsas, embora essa forma de spam seja relativamente recente para a abordagem utilizando AM, o uso de SVM, regressão logística e a combinação de múltiplos classificadores, associado a

técnicas de normalização lexical e indexação semântica (tenta estabelecer relação entre as palavras no texto) e desambiguação têm sido promissores.

- **Sistemas de recomendação:** buscam, a partir de um conjunto de informações relacionadas aos usuários de uma plataforma (e.g. itens comprados, conteúdos consumidos), inferir sobre possíveis interesses por novos produtos/conteúdos relacionados, para assim recomendá-los. Um algoritmo estudado durante o curso e que é utilizado nos sistemas de recomendação é o k- NN. A ideia básica é definir medidas de comparação (similaridade) entre usuários para encontrar aqueles que apresentam gostos semelhantes. Com essa informação, é possível inferir os gostos de um usuário baseado no gosto dos seus vizinhos, e recomendar, por exemplo, itens já consumidos pelos vizinhos.
- **Agronegócios:** também no agronegócio diversos algoritmos de AM já foram utilizados em diversas tarefas de classificação. Por exemplo na classificação da qualidade da carne e na agricultura de precisão. São exemplos de algoritmos utilizados: SVMs e RNAs e k-médias.
- **Saúde:** cada vez mais algoritmos de AM têm sido utilizados na área da saúde, sendo que alguns deles já apresentam capacidade preditiva superior à de médicos especialistas, como no diagnóstico de câncer de mama e doenças cardíacas. Boa parte dos bons resultados nessa área tem sido conseguido como resultado da combinação de algoritmos de agrupamento e de algoritmos de classificação. Na área da Saúde, a interpretabilidade do modelo e a sensibilidade do preditor muitas vezes são características importantes no momento da escolha de um algoritmo.

Otimização de RNAs

Abaixo são apresentadas algumas estratégias utilizadas para acelerar o aprendizado em redes neurais.

Uso de *minibatchs*

O uso de *minibachts* permite tornar o processo de aprendizado mais rápido, em especial no cenário do *big data*, no qual, devido ao elevado número de exemplos, o processo de aprendizado pode se tornar lento. A ideia da técnica de *minibatch* é quebrar o conjunto original de exemplos em subconjuntos menores chamados *minibatchs*, e processá-los sequencialmente.

Quando é utilizado o Grandiente Descendente (*Gradient Descent)* no processo de otimização utilizando *minibatchs*, a estratégia é denominada *minibatch Gradient*

Descent, ou ainda *Stochastic Gradient Descent*, quando cada *minibatch* possui um único exemplo.

Variações do *Gradient Descent*
Existem variações do *Gradient Descent* que buscam acelerar o processo de aprendizado:

- ***Gradient Descent with Momentum:*** busca suavizar as oscilações nos passos do algoritmo do Gradient Descent usando médias exponenciais (média móvel) para realizar a atualização nas matrizes de pesos;
- ***RMSprop:*** de forma similiar ao *Gradient Descent with Momentum,* busca compensar os valores dos pesos no caso de as derivadas provocarem muita oscilação nesses valores;
- ***Adam Optimization Algorithm****:* talvez seja um dos mais utilizados atualmente na otimização de redes. Esse algoritmo consiste basicamente na junção do *Gradient Descent with Momentum* com o algoritmo RMSprop.

Bibliotecas e Frameworks
Existem diversos *frameworks* e bibliotecas disponíveis para desenvolvimento em aprendizado de máquina. O ambiente de desenvolvimento também pode variar (computação local ou em nuvem). No caso de processamento em nuvem, ainda é possível a contratação de diferentes modalidades de serviço, começando desde a contratação apenas de recursos de *hardware*, até a contratação de um MLaaS (*Machine Learning as a Service*). Em relação ao *hardware* utilizado em AM, também existem diferentes hardwares que podem variar grandemente em termos do poder de processamento e do consumo de energia. Além das CPUs de uso geral, as GPUs e as TPUs oferecem recursos de hardware que podem aumentar significativamente a eficiência, tanto em termos de velocidade na etapa de aprendizado quanto no consumo de energia (caso das TPUs).
Assim, a escolha do ambiente e o uso de recursos devem buscar otimizar a relação custo x benefício, em termos da natureza do problema, e da tarefa de aprendizado a ser realizada.

Alguns dos *frameworks* e bibliotecas mais utilizados atualmente são:

- **Sckit-learn:** biblioteca de aprendizado de máquina de código aberto para a linguagem de programação Python;

- **TensorFlow:** muito eficiente para computação numérica usando tensores (estruturas de dados similares aos *ndarrays* da biblioteca *NumPy* do Python) e grafos de fluxo de dados (grafos podem ser entendidos como estruturas matemáticas que modelam relações entre objetos). A sua arquitetura flexível o torna multiplataforma, podendo ser parametrizado para computação em uma ou mais CPUs ou GPUs, em *desktops,* servidores ou dispositivos móveis;
- **PyThorch:** similar ao TensorFlow, utiliza tensores e grafos de fluxo de dados podendo também ser parametrizado para computação em uma ou mais CPUs ou GPUs, em *desktops,* servidores ou dispositivos móveis;
- **Keras:** API de alto nível, consistindo em um *front-end*, podendo utilizar diferentes *frameworks* como *back-end* (por exemplo o TensorFlow). Oferece uma estrutura do tipo *plug-and-play*, sendo adequado para desenvolvedores que buscam construir treinar, avaliar seus modelos rapidamente.

Alguns *frameworks* mais conhecidos por sua utilização em aprendizado profundo são:

- **Caffe;**
- **Microsoft Cognitive Toolkit (CNTK);**
- **Theano.**

Aqui terminamos o nosso material de revisão, lembre-se de que esse material não deve ser utilizado como única fonte de estudos, e que o estudo dos materiais-base/textos-base são de fundamental importância para que você possa ter um bom aproveitamento nas avaliações.