Resumo

A Few Useful Things to Know About Machine Learning, Domingos P. (pedrod@cs.washington.edu), Department of Computer Science and Engineering.

Aluno:Emanuel Eduardo da Silva Oliveira
Matrícula: 394271

O artigo aqui resumido condensa 12 lições sobre Aprendizagem de Máquina (ML, *Machine Learning*). Essas lições foram criadas a partir de experiências da academia e do mercado. Elas incluem dicas para se evitar armadilhas, questões relevantes e respostas para perguntas frequentes.

A lição 1 divide o processo de aprendizagem em 3 partes: **representação** (a concretização do algoritmo em alguma linguagem de programação), **avaliação** (a análise que determina se a saída do algoritmo é boa ou ruim) e **otimização** (a manutenção do algoritmo, os ajustes nas variáveis que o compõem para melhorar as saídas). Cada parte tem seus componentes: por exemplo, a aprendizagem pode ser representada por uma *Regressão Linear*, avaliada através de *Erro Quadrático* e otimizada com *Gradiente Descendente*.

A lição 2 cita o papel da **generalização** para além do conjunto de treinamento. A fé na generalização desenha o aprendizado para crer que os dados de teste e treino compartilham, mesmo que aproximadamente, algo em comum. Essa lição, também, ressalva a importância de deixar os dados de teste longe do processo de treinamento, a fim de evitar "contaminação" no classificador/regressor.

A lição 3 ressalta que os **dados, por si só, não dizem muita coisa**. Toda aprendizagem deve incluir, além dos dados, conhecimentos e premissas, a fim de obter uma boa generalização. Algumas premissas, como *exemplos similares têm classes similares*, constroem um processo indutivo, que, comparado ao processo dedutivo, requer muito menos dados de entrada para produzir bons resultados (e quanto mais dados, melhores eles são).

A lição 4 aborda um problema comum de ML: **overfitting (em suas várias facetas)**. Todo mundo que faz ML sabe sobre overfitting, mas ele acontece de diversas formas que não são imediatamente percebidas. Uma forma de entender o overfitting é decompondo-o em *bias* (a tendência de aprender constantemente a mesma coisa errada) e *variância* (a tendência de aprender coisas aleatórias independente do significado real). Alguns modelos são mais afetados por *bias*, como modelos lineares. Outros são mais afetados por *variância*, como árvores de decisão. A causa do overfitting pode ser tanto dados de treino mal classificados ou múltiplas instâncias de teste que pode considerar fatos significantes que, na verdade, não são. *Cross-validation* e *regularização* são métodos que ajudam a combater overfitting.

A lição 5 aborda a incapacidade da intuição humana perante a dados com grandes dimensões, ou **a maldição da dimensionalidade**. Generalizar em grandes dimensões se torna uma tarefa complicada e a similaridade, assim como modelos baseado em distribuições estatísticas, tão comuns em algoritmos de ML, deixa de funcionar bem em tais

situações. Felizmente, para contra-atacar tal maldição, há o que se de “benção da não-uniformidade”: na maioria das aplicações, os exemplos não estão uniformemente espalhados no espaço alto-dimensional, mas concentrados em uma parcela menor de dimensões. Então os algoritmos podem tirar vantagem de tal característica.

A lição 6 avisa que as **garantias teóricas não são tão confiáveis como se apresentam**. Mais comum delas é a quantidade de exemplos necessário para alcançar uma boa generalização. Ela falha quando fatores, como a quantidade de atributos, aumenta. Tal aumento cresce exponencialmente a quantidade de exemplos necessários para se reafirmar tal garantia. O principal papel das garantias teóricas é guiar o desenvolvimento dos algoritmos e não serem regras para aplicações práticas.

A lição 7 fala do papel decisivo da **engenharia das features**, ou *feature selection*. Se você tem features independentes entre si e se elas correlacionam com a classe, então a aprendizagem se torna fácil. Caso contrário, se a classe é uma função complexa das *features*, a aprendizagem não é tão direta. Muitos esforços de projetos em ML são direcionados para *feature selection*. É também uma das partes mais interessantes, onde a intuição, criatividade e “arte negra” são importantes. Muito do processo de tentativa e erro pode ser reduzido nessa etapa. O “santo graal” que tenta-se alcançar é a automatização dessa etapa.

A lição 8 enfatiza que **mais dados** é uma forma mais rápida e eficiente de melhorar um algoritmo, do que **tentar deixá-lo mais “esperto”**. Mas isso leva a um outro problema: escalabilidade. Uma enorme quantidade de dados está disponível hoje em dia, mas não há capacidade suficiente para processá-los. Como consequência, os classificadores mais simples passam a ser mais usados, pois levam menos tempo do que os classificadores mais complexos para classificar e, com mais dados, apresentam resultados tão bons quanto.

A lição 9 aponta que é **melhor treinar vários modelos ao invés de um só**. Um único modelo pode ser muito sensível ao contexto da aplicação. Pesquisadores entenderam que combinar variações de modelos diferentes apresenta frequentemente melhores resultados em troca de um pequeno esforço extra. Tal combinação é normalmente nomeada de *ensembles models.* Na técnica mais simples de *ensemble*, chamada de *baggin*, variações aleatórias do conjunto de teste são gerados por *resampling.* Isso funciona porque ele reduz enormemente a variância enquanto aumenta levemente o viés (bias). Há diversa outras técnicas de *ensembles models.*

A lição 10 afirma que **simplicidade não implica em acurácia.** Há diversos contra-exemplos que são contrários a intuição, que não há necessariamente uma conexão entre o número de parâmetros e a sua tendência a overfitting. Um exemplo é a função *sign(sin(ax))* pode fazer curvas arbitrariamente complexas com apenas 1 parâmetro. Há uma outra visão que liga a complexidade com o tamanho do espaço de hipóteses: espaços mais curtos permitem representações mais curtas de hipóteses. No final, conclui-se que hipóteses mais simples devem ser escolhidas porque a simplicidade é uma virtude por si só, não por conta de uma conexão hipotética com acurácia.

A lição 11 afirma que **representatividade não implica em aprendizagem**. Ou seja, só porque uma função consegue ser representada não significa que ela pode ser aprendida. Algumas representações são exponencialmente mais compactas do que outras. Como resultado, elas requerem exponencialmente menos dados para para ensinar essas funções.

Muitos algoritmos têm funcionado formando combinação linear de funções mais básicas, em diversas camadas e essa tem sido uma das principais áreas de pesquisa em ML.

A lição 12 afirma que **correlação não implica em causalidade.** ML é comumente aplicada sobre dados observacionais, onde as variáveis preditivas não estão sobre controle, diferente dos dados experimentais, onde elas estão. Por outro lado, correlação é um sinal de uma potencial conexão casual, e nós podemos usar isso para uma investigação aprofundada. A casualidade tem duas consequências práticas sobre ML. A primeira, nós gostaríamos de predizer os efeitos de nossas ações, não apenas as correlações entre as variáveis observáveis. Segundo, se você pode obter dados experimentais, então, por todos os meios, faça isso.

Concluindo, Machine Learning tem muito de sabedoria popular que pode ser muito difícil de entender, mas é crucial para o sucesso. Esse artigo, resume alguns dos tópicos mais relevantes que refletem o impacto de aprender tal sabedoria.