MACHINE LEARNING

Algumas estimativas apontam que em torno de 80% dos projetos de Machine Learning (ML) não são colocados em produção, não saindo da fase de protótipos e experimentações. E dos que entram em produção, muitos acabam sendo desativados, por várias razões como baixo desempenho operacional ou por não alcançarem um nível adequado de assertividade. Outros também, por acabarem não se justificando como vantagem competitiva para a empresa.

Um projeto de ML não é algo que se começa apenas a partir de uma inspiração genial. Ele precisa ter uma clara oportunidade de negócio para a empresa. Afinal serão comprometidos tempo, energia, dinheiro e talentos, daqueles escassos profissionais que conseguem realmente desenvolver soluções adequadas de ML.

Portanto, para ter sucesso em projetos dessa natureza é necessário comprometimento e engajamento dos executivos e gestores, e a criação de uma equipe de profissionais preparados. Uma equipe para desenvolver projetos de ML não é constituída apenas por cientistas de dados e engenheiros de ML. Um projeto de ML faz parte de uma solução para um problema de negócio e isso envolve integração com outros sistemas, acesso a dados validados e limpos, interfaces adequados, entendimento do negócio e assim por diante. Uma equipe multidisciplinar.

O apoio dos gestores é fundamental. Mas, para engajar executivos de negócios, é importante que eles conheçam como um projeto de ML é desenvolvido. Vamos seguir um roteiro básico de um projeto e apontar alguns pontos principais que devem ser enfatizados quando conversando com seu CEO e demais C-levels, que devem ser envolvidos nas questões de aprovação, e, claro, da alocação dos budgets.

Vamos seguir o fluxo acima. A primeira ação é identificar claramente um projeto de negócios que deve ser resolvido com ML. Não esqueça da máxima “The First Rule of Machine Learning: Start without Machine Learning”. Assim, o sistema de ML deve prover uma solução que seja mais barata ou mais funcional ou que traga um benefício de negócio palpável. Uma vez identificado essa demanda e que ML aponta ser a melhor solução, analise se os dados para essa solução existem e se estão disponíveis em volume, variedade e qualidade adequados.

Algoritmos sofisticados não darão certo se não puderem ser treinados de forma adequada. Para isso serão necessários dados. Dados são o sistema circulatório de qualquer sistema de ML. Um dos princípios fundamentais da “data science” diz “os dados e capacidade de extrair conhecimento útil a partir deles, devem ser considerados importantes ativos estratégicos”.

A carência ou falta de qualidade dos dados é que faz muitos projetos de ML naufragarem. Sem dados, os algoritmos não são treinados adequadamente e faz com que a solução proposta não seja validada. Por isso, antes de entrar em projetos de ML planeje e desenvolva uma estratégia de dados.

Uma estratégia de dados deve conter pelo menos seis componentes:

1. Aquisição e processamento: obter e processar os dados necessários para desenvolver protótipos e algoritmos. Os dados podem vir de diversas fontes, sejam internas, como sistemas da empresa ou externas. Lembre-se que muitas vezes os ERPS são uma boa fonte de dados, mas em torno deles existem muitos outros sistemas que têm muitos dados a oferecer. Como dados externos temos desde base de dados públicas, como de imagens ou as disponibilizadas por órgãos públicos ou empresas especializadas. Além disso, novos dados podem ser gerados por novos sistemas, como sensores instalados em carrinhos de supermercados ou câmeras que mostrem o trajeto dos clientes nos corredores das lojas.

2. Qualidade: desenvolver um conjunto de dados com as características apropriadas para resolver o desafio de negócios, minimizar o viés e oferecer dados de treinamento rotulados com alto grau de precisão. Eliminar o viés é importante. Se seus dados mostrarem desbalanceamento, como maior número de determinado tipo de clientes, seus algoritmos vão aprender de forma distorcida e suas respostas tenderão a amplificar o viés. Esteja ciente do viés em seus dados e modelos para tomar as ações apropriadas e minimizar seu impacto. Teste minuciosamente os modelos para garantir que as variáveis que não devem afetar as previsões não o façam. Se possível, exclua essas variáveis dos modelos. Infelizmente, evitar viés não é muito simples.

3. Contexto: entender a procedência dos seus dados e os mapeamentos pelos quais eles passam, para que você os use e os compartilhe efetivamente em suas iniciativas de ML. É fundamental garantir que os resultados obtidos na fase de testes internos sejam mantidos quando aplicados aos dados do mundo real. Uma precisão de 95% em um teste interno será de pouco valor se a precisão cair para 20% quando o modelo entrar em produção. Teste o modelo o mais cedo possível com os dados do mundo real. A máxima é “se você não analisar os dados do mundo real o mais cedo possível, nunca obterá algo que funcione na produção”. Por isso entender o contexto é de grande importância. Tenha especialistas no negócio e no problema em si, que possam validar se os dados estão realmente sendo úteis para treinar e validar o modelo.

4. Armazenamento: armazene e estruture seus dados de forma adequada para apoiar seus objetivos em relação ao acesso, velocidade, resiliência e conformidade. Um ponto de atenção para modelos que trabalham com imagens. Armazenar e processar “dados sujos” é um dos desafios mais significativos que enfrentamos em projetos de IA. Volumes menores de dados relevantes e bem rotulados normalmente permitem melhor precisão do modelo do que grandes volumes de dados de baixa qualidade. No mundo ideal, gostaríamos de trabalhar com imagens rotuladas com 100% de precisão. Mas, na realidade, os dados geralmente não estão rotulados, ou rotulados de forma esparsa ou incorreta. Os dados rotulados por humanos muitas vezes podem ser mal rotulados. A rotulagem de dados é

frequentemente fornecida por processos de crowdsourcing e realizada por pessoas que não são especialistas. Em alguns contextos, a rotulagem também pode ser intrinsecamente subjetiva. Além disso, indivíduos que olham grandes volumes de dados podem experimentar o fenômeno da saturação visual, falta de elementos que certifiquem o tipo de objeto na imagem ou até mesmo estão vendo e rotulando objetos que não estão nela. Muita atenção a isso!

5. Aprovisionamento: otimize a acessibilidade dos dados e a implemente medidas de proteção e salvaguardas. Os dados devem ser armazenados para que os acessos sejam facilitados. Sua estratégia de armazenamento de dados afetará a usabilidade e o desempenho dos seus dados. A sua abordagem de aprovisionamento deve ser direcionada pela natureza dos seus dados, a taxa de crescimento e os requisitos de acessibilidade. De maneira geral as empresas dispõem de especialistas em bancos de dados relacionais (como MySQL ou Oracle) e usar estas tecnologias não é mais segredo. Mais recentemente, bancos de dados NoSQL (como Mongo ou Redis) tornaram-se populares porque não exigem as restrições associadas aos bancos de dados relacionais. Por conseguinte, eles são comumente associados a iniciativas de que chamamos de "big data". Considere misturar e combinar diversos formatos de dados para atender às necessidades dos seus projetos de ML. Portanto, recrute ou treine seu pessoal que trabalha com SQL em sistemas NoSQL.

6. Gerenciamento e segurança: gerencie a segurança, acesso e permissão de dados para garantir o seu uso adequado. Não esqueça de fazer um planejamento com previsão para aumento significativo no volume de dados. As soluções em nuvem permitirão que você armazene a quantidade de dados que desejar, mas equilibre o custo do armazenamento imediato e no longo prazo. Se estiver operando seu próprio hardware, você também precisará decidir se deseja arquivar parte dos dados fora do seu data center. Por exemplo, pode ser necessário manter armazenamentos de dados separados fisicamente para garantir isolamento de dados pessoais. Monitore os custos e o desempenho do sistema para que você possa agir antes que os custos se tornem proibitivos ou se esgote o espaço de armazenamento. Trate a resiliência dos seus dados como de missão crítica. Os dados são o componente mais valioso da sua estratégia de IA: se seus dados foram perdidos, você não poderá reconstruir seus modelos e perderá uma parte significativa do valor da sua empresa. Crie políticas de segurança para proteger a empresa contra incidentes e violações. Verifique se o acesso aos dados é somente para leitura. Exceto para os administradores dos dados, ninguém deve poder excluir ou alterar dados. Lembre-se que as leis de proteção à privacidade estão cada vez mais severas.

Uma governança de dados é requisito essencial para que as iniciativas de ML tenham sucesso. Geralmente é um item que quase não aparece nas conversas sobre ML que vejo por aí. Fala-se muito nos algoritmos, mas pouca atenção é dada aos modelos de governança de dados. Por exemplo, a governança vai definir como tratar obsolescência dos dados. Dados antigos podem ser um desafio significativo e são uma consideração importante ao planejar sua estratégia de armazenamento. Se você estiver analisando informações que mudam rapidamente, decida quantos e quais dados históricos serão relevantes. Você pode incluir todos os dados, um volume específico de dados ou dados de uma determinada janela de tempo. Selecione uma abordagem apropriada para o problema que você está

resolvendo. Lembre-se que IA está em constante evolução e sua estratégia pode evoluir à medida que sua solução amadurece. Se você estiver correlacionando ações com o tempo, considere cuidadosamente a janela para suas séries temporais. Se você estiver prevendo níveis de estoque, alguns meses de dados não conseguirão capturar uma variação sazonal significativa. Por outro lado, para usar um algoritmo de previsão que identifica que os sinais vitais de um indivíduo na UTI estão se deteriorando, e com isso acionar uma intervenção médica, a pressão arterial deste indivíduo no mês passado não será muito relevante. Entenda se os efeitos periódicos podem impactar seu sistema e valide se seus modelos e previsões se baseiam em vários ciclos do período típico que você está modelando.

Além disso, gerenciamento e segurança de dados são componentes críticos de uma estratégia de dados. Os dados pessoais são protegidos por legislação e você deve armazená-los com segurança. Pode ser necessário criptografar dados quando armazenados, bem como ao transmiti-los entre sistemas. Pode ser vantajoso separar os dados pessoais do seu repositório de dados principal, para que você possa aplicar um nível de segurança mais alto sem afetar o acesso da sua equipe a outros dados. Registre todas as solicitações de acesso com a identidade do solicitante e os detalhes dos dados extraídos. Contrate terceiros para realizar testes de penetração para validar a segurança de seus sistemas. Se um indivíduo pedir demissão ou for demitido, revogue imediatamente o acesso a todos os sistemas confidenciais, incluindo seus dados. E valide que sua equipe está ciente da legislação e das políticas de segurança e privacidade implementadas.

Lembre-se que uma grande parte do tempo de um projeto de ML está relacionado com dados, seja na sua obtenção ou limpeza.

Agora podemos começar a fase de treinamento dos algoritmos. É uma fase importante do projeto. Um sistema de ML é um sistema de software. Para sair da ideia para a realidade deve seguir a prática de desenvolvimento ágil que já conhecemos, embora os prazos possam ser muito mais incertos que nos sistemas programáticos. Se você não tem experiência com projetos de ML terá alguma dificuldade de mensurar prazos e custos. Por exemplo, negocie claramente com os usuários qual a precisão (níveis de acerto) que os modelos devem atender. Os dados de treinamento estão rotulados e limpos, ou os dados devem ser criados? A falta de dados ou dados não rotulados e sujos pode atrasar em muitos meses o projeto! Quais prazos devem ser cumpridos? Isso pode ser muito mais desafiador do que no desenvolvimento de software tradicional, pois a melhoria de um modelo exige experimentação. Pode levar poucas semanas para se chegar a 50% de precisão, uns meses para 80%, mais alguns para 95% e muitos mais para alcançar o patamar de 98%. Até que nível você realmente precisa chegar? E, acorde também como considerar que a solução está pronta para produção. Como vemos, tem diferenças em relação aos sistemas programáticos determinísticos.

Em termos de algoritmos, identifique quais seriam as melhores opções. Uma das principais dúvidas que ouço nas aulas de ML é sobre o que é e quando usar técnicas de aprendizado de máquina supervisionadas e não supervisionadas. Estes termos implicam que são modelos diferentes. Fazendo uma metáfora, imagine alunos sendo supervisionados por um

professor que passa um objetivo ou alvo específico e mostra diversos exemplos para que eles consigam ter um direcionamento de como aprender a executar sua tarefa, para melhor atingir o alvo. No não supervisionado o professor dá exemplos, mas não indica o alvo a ser atingido. Os alunos estão livres para tirar suas conclusões a partir dos exemplos.

Estas técnicas endereçam resolução de problemas diferentes. Qual usar dependerá do objetivo do negócio. De maneira geral as técnicas supervisionadas buscam responder uma questão específica e quão mais prescritivo o caso, mais adequadas elas serão. Um exemplo simples: analisar um vídeo gerado por um drone, para identificar danos em uma cerca de proteção. Você tem dados (frames do vídeo) e um alvo específico. Já o não supervisionado o objetivo é mais aberto, como por exemplo, entender melhor segmentos de mercado.

Como exemplos teríamos para supervisionados de classificação, usos para detecção de fraudes, identificação de imagens, diagnósticos médicos e retenção de clientes. Para supervisionados de regressão podemos citar previsão de crescimento de população, previsão do tempo e expectativa de vida. Já para não supervisionados de dimensionalidade citaríamos descoberta de novas estruturas moleculares e para não supervisionados de "clustering" ou agrupamento, os famosos sistemas de recomendação. Como as técnicas não supervisionadas são exploratórias, não tem respostas certas ou erradas. Já para os supervisionados, você tem como buscar respostas certas, que podem ser medidas como o grau de acerto de diagnósticos de imagens médicas.

Mas, além desta primeira classificação, existem grande número de algoritmos e técnicas de refinamento que vão exigir dos profissionais, conhecimentos específicos. Aí que "pega" o conhecimento matemático. O modelo tem que ser treinado, ajustado e refinado e seus resultados avaliados. Por exemplo, podemos identificar padrões e correlações que não fazem sentido. Precisamos ter confiança que não temos anomalias. Diferentemente da computação programática onde o software responde diretamente ao desenvolvedor que coloca todas as instruções em linhas de código e caso a resposta não seja correta, você depura e conserta o código, a IA interpreta dados com seus algoritmos e daí toma suas decisões. No não supervisionado, não sabemos se está certa ou errada, pois a resposta pode ser algo que nós humanos não havíamos percebido e a máquina identificou como um padrão e gerou sua decisão a partir desta constatação. A máquina pode gerar resultados surpreendentes, que nós jamais imaginaríamos.

Na fase de treinamento, além de selecionar os modelos e analisar os data sets de treinamento, tome cuidado para não criar efeitos indesejados, como underfitting e overfitting, ou inserção de vieses no sistema. No underfitting o modelo não conseguiu aprender suficiente sobre os dados. O underfitting leva à um erro elevado tanto nos dados de treino quando nos dados de teste. Overfitting é quando o modelo aprende demais sobre os dados. Nesse contexto, o modelo mostra-se adequado apenas para os dados de treino, como se o modelo tivesse "decorado" os dados de treino e não é capaz de generalizar para outros dados nunca vistos antes. Quando isso acontece, os dados de treino apresentam resultados excelentes, enquanto que a performance do modelo cai drasticamente com os dados de teste.

Outra situação é a “cauda longa”. Os algoritmos de ML não entendem realmente o mundo, pois são estruturas de software compostas por fórmulas matemáticas interconectadas. Assim, por não disporem de bom senso e nem aprenderem com a experiência de viver no mundo real, como nós humanos, erram de forma grosseira em situações que nós facilmente reconhecemos e nos saímos bem. Esses cenários numerosos e com alto grau de variabilidade são difíceis de serem criados. É o fenômeno da “cauda longa”, onde isoladamente são raros, mas em seu total são numerosos. Como é praticamente impossível registrar um volume adequado desses cenários via câmeras do mundo real, são necessárias técnicas de geração de situações sintéticas, que simulem o máximo de potenciais ocorrências possíveis.

Vamos criar um caso concreto para exemplificar o desafio do treinamento, com uma aplicação de ML para analisar imagens usadas em radiologia, para detectar uma doença como câncer de mama. Uma aplicação, que não apenas resolve problemas reais, como ajuda a salvar vidas. Mas os desafios são imensos quando saímos da ideia para a realidade. Primeiro muitos hospitais têm equipamentos antigos, e incompatíveis entre si, gerando imagens de baixa qualidade. Para o treinamento do sistema (usaremos técnicas supervisionadas, para lembrar...) precisamos que as imagens sejam rotuladas para que o algoritmo aprenda a identificar as células cancerosas. Precisamos que radiologistas anotem nas imagens onde estão as células cancerosas. Estas anotações são usadas como output enquanto as imagens são usadas como input, para que o sistema aprenda a reconhecer onde está o câncer. Precisamos de volumes muito grandes de imagens. Um desafio: além do hospital específico, temos outras fontes? Bem, onde os problemas acontecem? Os radiologistas nem sempre acertam na rotulação. Algumas estimativas apontam que os médicos têm acurácia entre 50% e 70%, dependendo do seu nível de experiência e cansaço ao analisar as imagens. Com isso a rotulação sai com erros. Dois radiologistas podem chegar a conclusões diferentes ao analisar a mesma imagem. Os hospitais usam equipamentos diferentes e as imagens refletem também hábitos dos radiologistas como posicionamento e ângulo que usam o scanner. Uma máxima antiga de TI continua valendo: “garbage-in, garbage-out”. Nenhum modelo, por mais sofisticado que seja resistirá a dados errôneos. Assim rotulações erradas irão comprometer o resultado, inutilizando o sistema. Ter dados corretos, no caso, rotulando corretamente, é como converter petróleo em gasolina. A gasolina é que vai movimentar o motor, não o petróleo.

Ultrapassando as barreiras dos dados, chegamos a outras: humanas. Se os radiologistas olham com receio a solução de IA, o muro será quase intransponível. Eles devem ver a IA como sua auxiliar no seu estafante trabalho de analisar diariamente várias imagens. O processo tem que ser implementado aos poucos, para ganhar sua confiança. Primeiro, quando a IA identificar um possível câncer, apenas mude a prioridade da imagem na fila a ser analisada pelo radiologista, sem avisá-lo. A rotulação que ele fará ajudará a melhorar o algoritmo. É um reforço no seu treinamento. Em uma etapa posterior, alerte-o e ele prestará mais atenção à imagem, rotulando-a com mais cuidado, o que vai melhorar mais ainda o aprendizado do algoritmo. E, um dia, quando a confiança estiver elevada, passe para o radiologista apenas os casos onde sua intervenção será necessária. Lembre-se, o sistema de ML não vai substituir o médico, mas poderá ser um bom auxiliar.

Chegamos na fase de colocar o sistema em produção. Mas, ter um modelo em produção não significa que ele precisa ser visível publicamente. Pense em uma fase piloto, onde ele deve ser exposto a dados ativos, para que sua equipe possa fazer refinamentos até atender aos requisitos que serão disponibilizados para uso aberto. Com os dados ativos, você pode realizar testes mais exaustivos de execução e fornecer à sua equipe de ML feedback sobre o que está funcionando bem e o que não está. Nesse estágio, priorize o estabelecimento de um processo de liberação controlada e gradual. Você também deve monitorar o desempenho e a escalabilidade do seu sistema. Planeje ciclos contínuos de melhoria, investigue e implemente refinamentos do modelo e, alterações das interfaces. Novos modelos devem ser comprovadamente superiores aos que substituem. Teste todas as alterações antes que as atualizações sejam lançadas no ambiente de produção. Esses ciclos continuarão por toda a vida útil do sistema.

Ao planejar o projeto de implementação, valide onde seu sistema será executado: no seu data center ou na nuvem? A opção pelo seu data center geralmente é tomada quando seus dados são altamente sensíveis e você precisa mantê-los sob seu controle direto. Normalmente, isso é possível apenas para empresas que já possuem sua própria infraestrutura de hardware. Essa pode ser uma opção válida se o volume de solicitações de gerenciamento for conhecido e relativamente estável. No entanto, todo o novo hardware adicional deve ser adquirido e provisionado, o que limita a escalabilidade. A opção de usar nuvem pode ser vantajoso para a maioria das empresas. Não esqueça que você pagará para transferir e receber dados, mas vai evitar o custo da aquisição de hardware e de uma equipe para gerenciar o ambiente. Como transferir grandes volumes de dados é custoso, se você já estiver usando ambientes de nuvem como os da Amazon, Google, IBM ou Microsoft, para suas aplicações tradicionais, é mais atrativo continuar com ele para seus projetos de IA.

Para comprovação de que as novas versões de IA são eficazes e melhores que as versões anteriores, teste seu sistema de IA em vários estágios. Durante o treinamento: enquanto seu modelo está sendo treinado, teste-o constantemente com um subconjunto de dados de treinamento para validar sua precisão. Os resultados não representam totalmente o desempenho do modelo, porque os dados do teste randomizados irão influenciar o modelo. Como resultado, esse teste provavelmente exagerará a precisão do modelo. Não considere este número quando for vender a ideia para seu board!

Na validação, reserve uma parte dos seus dados de treinamento para isso. Este conjunto de testes, conhecido como conjunto de validação, não deve ser usado para treinamento. Por conseguinte, as previsões que o seu sistema de IA faz a partir do conjunto de validação representam melhor as previsões que ele fará no mundo real. A precisão no estágio da validação geralmente é menor que a precisão obtida durante o treinamento. Cuidado para que o seu conjunto de dados represente bem os dados do mundo real, pois, caso contrário, pode gerar uma precisão superestimada. Muita atenção com os dados que podem embutir viés nos algoritmos.

Após a criação do seu modelo, teste-o com dados ativos para obter uma medida de precisão mais apropriada. De maneira geral essa precisão é menor que a obtida no teste e deve ser

refinada continuamente. Existem três medidas de “precisão” comumente usadas em sistemas de ML: recall, precisão e exatidão. Como exemplo, vamos pensar um sistema de ML que determine se uma determinada fruta é ‘boa’ ou ‘ruim’ com base em análises de imagens desta fruta.

Existem quatro resultados possíveis:

1. Verdadeiro positivo: a fruta é boa e a IA prediz ‘boa’.

2. Verdadeiro negativo: a fruta é ruim e a IA prediz ‘ruim’.

3. Falso positivo: a fruta é ruim mas a IA prediz ‘bom’.

4. Falso negativo: a fruta é boa mas a IA prediz ‘ruim’.

Para medir quão preciso é o sistema, use simultaneamente as três medidas, recall, precisão e exatidão.

Recall: Que proporção de frutas encontrei corretamente? É o número de frutas boas identificadas corretamente, dividido pelo número total de frutas boas identificadas, corretamente ou não.

Precisão: que proporção de frutas que eu disse são boas, eu acertei? É o número de frutas boas identificadas corretamente dividido pelo número total de frutas rotuladas como boas (identificadas corretamente ou não).

Acurácia: que proporção de frutas rotulei corretamente? É o número de frutas corretamente identificadas como boas ou ruins, dividido pelo número total de frutas.

Equilibrar precisão e recall pode ser difícil. À medida que você ajusta seu sistema para um recall mais alto, ou seja, menos falsos negativos-, você aumenta os falsos positivos e vice-versa. A opção por minimizar falsos negativos ou falsos positivos, dependerá do problema que você está resolvendo e do seu domínio. Se estiver desenvolvendo uma solução de marketing, convém minimizar os falsos positivos. Para evitar o constrangimento de mostrar um logotipo incorreto, é menos danoso perder algumas oportunidades de marketing. Mas, se estiver executando diagnósticos médicos, convém minimizar os falsos negativos para evitar o erro de um diagnóstico.

E finalmente, lembre-se que um sistema de ML não cessa de evoluir nunca. Infeizmente na hora que você coloca um sistema de ML em produção ele começa a degradar! Em algoritmos complexos, muitas vezes ocorre uma “falha catastrófica”, quando o sistema é continuamente exposto a data sets diferentes do qual foi treinado. Ele pode se ajustar a esse novo contexto e “esquece” o que aprendeu antes. E quando recebe um dado de entrada similar ao do antigo data set, sua assertividade é muito baixa, pois, como um algoritmo de ML não tem memória, ele “esquece” o aprendizado anterior.

Fica claro que para manter a coerência ou “inteligência” do sistema de ML, teste regularmente os resultados com dados ativos, para garantir que os resultados continuem atendendo seus critérios de aceitação. Separe budget para futuras atualizações e reciclagem, e monitore sistematicamente a degradação do desempenho. À medida que sua empresa cresce ou muda o foco, os dados (incluindo dados de séries temporais e características do produto) evoluirão e se expandirão. A reciclagem sistemática de seus sistemas deve ser um componente de sua estratégia de ML. Lembre-se de que um sistema de ML pode e deve estar sempre melhorando. Surgem novos algoritmos e as técnicas de ML que usamos hoje podem se tornar obsoletas em poucos anos.

Creio que esse resumo dá uma ideia simples de como funciona um projeto de ML. Claro que têm outros detalhes, mas de maneirageralacreditoqueessavisãosuperficial já é suficiente para esclarecer esse “monstro mágico” que muita gente acha que um sistema de ML é. Mas, no fundo é um sistema de software que usa intensamente algoritmos matemáticos.