Matheus Facure Alves

Da Econometria ao Aprendizado de Máquina

Brasil

2017

Matheus Facure Alves

Da Econometria ao Aprendizado de Máquina

Universidade de Brasília - UnB

Faculdade de Economia, Administração e Contabilidade

Graduação

Orientador: Daniel Oliveira Cajueiro

Brasil

2017

Matheus Facure Alves
Da Econometria ao Aprendizado de Máquina/ Matheus Facure Alves. – Brasil, 2017-
53 p. : il. (algumas color.) ; 30 cm.

Orientador: Daniel Oliveira Cajueiro

Monografia – Universidade de Brasília - UnB
Faculdade de Economia, Administração e Contabilidade
Graduação, 2017.

1. Econometria. 2. Aprendizado de Máquina. 2. Redes Neurais. I. Daniel Oliveira Cajueiro. II. Universidade de Brasília. III. Faculdade de Economia, Administração e Contabilidade. IV. Da Econometria ao Aprendizado de Máquina

Resumo

Nos últimos anos, aprendizado de máquina progrediu de forma exponencial, atraindo o interesse não só de acadêmicos mas do publico geral. No entanto, ainda é tímida a aplicação do ferramental de aprendizado de máquina em questões de interesse econométrico. Assim, este trabalho visa cobrir um pouco dessa lacuna, introduzindo aprendizado de máquina na forma de uma transição suave a partir da econometria.

Palavras-chave: aprendizado de máquina. econometria. redes neurais.

Abstract

In recent years, machine learning has achieved exponential growth, attracting the interest not only from academics but also from the general public. However, there si still very little work on the application of machine learning tools in questions of econometric interest. Thus, this paper aims to cover a little of this gap, introducing machine learning in the form of a smooth transition from econometrics.

Keywords: machine learning. econometrics. neural networks.

Lista de ilustrações

Sumário

Introdução

Nos últimos dois anos, os avanços em aprendizado de máquina progrediram em escala nunca antes vista, passando de um campo de conhecimento obscuro, restrito aos poucos profissionais da área, para um assunto de interesse geral (VEJA, 2017). Além disso o interesse em adquirir conhecimento necessário para um cargo em aprendizado de máquina disparou, tornando o tópico mais demandado por estudados de universidades como o MIT (KIRSNER, 2017). Cada vez mais, firmas de tecnologia tentam atrair profissionais de inteligência artificial com salários rivalizando o de estrelas do espore (ECONOMIST, 2016). No Brasil, esse entusiasmo geral chegou apenas em 2017, numa avalanche tão arrebatadora que colocou o interesse em contratar profissionais de aprendizado de máquina acima da tarefa de entender o que de fato é essa ciência e o que se deve esperar desses profissionais.

Se por um lado há uma explosão de demanda, a oferta de pessoal capacitado nessa área continua fraca. Tendo isso em vista, este trabalho tem dois propósitos principais. O primeiro é fornecer uma introdução ao aprendizado de máquina, a nível de graduação, para aqueles que estão vindo da econometria. O segundo é esclarecer como aprendizado de máquina se diferencia da econometria e de suas técnicas estatísticas. Não temos a pretensão de fornecer conteúdo suficiente sobre aprendizado de máquina, mas apenas mostrar como as ferramentas desse campo estão mais próximas das de econometria do que se imagina a primeira vista. Esperamos com isso despertar maior interesse na intercessão entre econometria e aprendizado de máquina, uma área extremamente promissora mas ainda pouco explorada.

A primeira parte deste trabalho faz a transição da econometria para o aprendizado de máquina no âmbito teórico. Primeiro apontamos algumas limitações da abordagem econométrica clássica de modelagem e mostramos como técnicas de aprendizado de máquina podem contornar essas limitações. Também apontamos alguns desafios e problemas por resolver nas aplicações de aprendizado de máquina em conjunto com econometria. Em segundo lugar, abordamos modelos de aprendizado de representações com redes neurais profundas , ressaltando a similaridade entre esses modelos e os comumente utilizados em econometria. A segunda parte mostra como aplicar esses modelos em dados reais, num problema complexo de previsão de preços de passagens aéreas.

Parte I

Referencial Teórico

1 Da Econometria ao Aprendizado de Máquina

É de senso comum que economia, como uma ciência social, vale-se largamente de ferramentas matemáticas. Na maioria dos casos, utilizamos modelos para tentar entender um pouco da complexidade presente nas diversas interações humanas. Sob essa perspectiva, como economistas, podemos cair na tentação de ver a ciência de aprendizado de máquina apenas como uma fonte de técnicas estatísticos poderosíssimas, que podem ser prontamente utilizados nos problemas complicados demais para os nossos métodos econométricos tradicionais. Ao fazer isso, acabamos por ignorar décadas de conhecimento dedicado ao estudo de análise de dados e interações complexas, algo que seria bastante proveitoso se aliado ao conhecimento econômico. Esse conhecimento não está apenas nos modelos de aprendizado de máquina, mas também nos fundamentos teóricos (sob as formas de aprendizado estatístico e teoria da informação), na métricas e formas de avaliação e validação de modelos.

Assim, muito mais proveitoso seria integrar essas duas ciências - economia e aprendizado de máquina - não só com respeito às suas ferramentas, mas também quanto aos objetivos e objetos de estudo. Se considerarmos um ponto de vista mais amplo, podemos lembrar que aprendizado de máquina é fortemente recomendado em cenários onde a expertise humana é fraca ou ausente. E por mais que detestemos admitir isso, a maioria dos problemas de economia aplicada cai dentro dessa categoria. De fato, sabemos que estamos sujeitos a todo tipo de vieses cognitivos e ilusões de racionalidade (KAHNEMAN, 2011), que podem ser extremamente prejudiciais tanto do ponto de vista das péssimas decisões econômicas que tomamos na vida privada quanto do ponto de vista das políticas públicas, pensadas e executadas também por agentes que muitas vezes agem de maneira irracional. Seria, portanto, extremamente interessante se conseguíssemos desenhar sistemas inteligentes artificiais que não estivessem suscetíveis aos nosso deslises cognitivos. Quiçá em algum momento eles possam nos substituir nas tarefas cujo nosso desempenho é simplesmente terrível, como planejamento financeiro ou montar a política econômica de um país.

Não podemos dizer ao certo quão longe ainda estamos desse objetivo, mas aqui ele será útil para motivar a integração entre economia e aprendizado de máquina. Hoje, a maioria dos trabalhos que utilizam aprendizado de máquina em problemas de economia ainda utilizam principalmente modelos prontos (*off-the-shelf*) (ATHEY, 2016), mas esperamos que, com o tempo, econometristas modifiquem essas ferramentas de acordo com as necessidades particulares das ciências sociais.

1.1 Econometria e Aprendizado de Máquina

Em termos gerais, a econometria é uma ciência que busca extrair conhecimento econômico a partir de dados empíricos. Mais precisamente, econometria busca desenvolver métodos estatísticos para estimar relações, testar teorias e avaliar o impacto de decisões e políticas econômicas (WOOLDRIDGE, 2012, p. 1). Aprendizado de máquina, por outro lado, é a ciência de fazer com que os computadores aprendam a realizar alguma tarefa sem serem explicitamente programados para isso. Em termos mais técnicos, aprendizado de máquina é quando um computador, por meio de uma experiência E, melhora sua habilidade em uma tarefa T, de acordo com alguma métrica de performance P (MITCHELL, 1997, p. 2).

Embora essas duas definições pareçam severamente distintas, econometria e aprendizado de máquina compartilham muitos fundamentos teóricos – como o princípio de maximização da verossimilhança –, modelos matemáticos – como regressão linear e logística – e até objetivos: ambas se dedicam, em boa parte, a resolver problemas em cenários de incerteza, isto é, quando não é possível fazer afirmações sem alguma margem de erro. Assim, as diferenças entre elas são, na verdade, bastante sutis: em aprendizado de máquina estamos principalmente preocupados em gerar boas previsões na presença de restrições computacionais, enquanto que em econometria estamos mais interessados em estabelecer conclusões estatisticamente válidas, achar padrões interessantes nos dados e resumi-los de maneira informativa (**??**).

Mais ainda, diferentemente de econometria, em aprendizado de máquina não se dá muita atenção às propriedades assintóticas dos estimadores e estes quase nunca são normalmente distribuídos em torno de uma média. A enfase recai mais na minimização do erro quadrático médio, o que significa que os modelos tipicamente enfrentam um *trade-off* entre viés e variância; os parâmetros desses modelos são normalmente estimativas enviesadas por alguma técnica de regularização (puxados em direção à média ou a um valor definido a priori), afim de se obter maior sucesso preditivo. Por fim, quando as propriedades tóricas são analisadas, isso é feito sob a forma de limites no pior dos casos ou analise de risco estrutural (ATHEY; IMBENS, 2016).

Por exemplo, em econometria, nós podemos construir um modelo para estimar se a presença de computadores em sala de aula melhora o desempenho dos alunos no ENEM, de forma estatistica e economicamente significantes. Para tanto, é preciso ser extremamente rigoroso com a forma de coleta dos dados, com as hipóteses assumidas pelo modelo e com as interpretações dos resultados. Com aprendizado de máquina, por outro lado, nosso interesse seria, por exemplo, prever a nota de um aluno no ENEM, dadas as variáveis da escola onde ele estudou. No primeiro problema, estamos preocupados com inferência, isto é, entender como variáveis independentes se relacionam com variáveis dependentes. No segundo caso, estamos mais preocupados com previsão, isto é, prever uma variável de saída

a partir de variáveis preditoras (JAMES et al., 2014). Nesse caso, o rigor das inferências probabilísticas e a interpretação dos modelos é posta em segundo plano para que se possa fazer boas previsões. Note que essas distinções nem sempre são válidas e refletem mais uma terminologia usual do que uma definição precisa.

Além disso, aprendizado de máquina nem sempre usa modelos estatísticos, como nos casos de aprendizado por reforço, e estatística também se preocupa com escopos fora da área de aprendizado de máquina, como o design de experimentos (CAJUEIRO, 2015). Mas, para a manutenção da linha de raciocínio aqui desenvolvida, vamos manter essas distinções um pouco grotescas e menos precisas, de que aprendizado de máquina foca mais em previsão e econometria, em inferência causal.

Para melhor entender como aprendizado de máquina e econometria se relacionam, nas próximas sessões nós consideraremos um problema simulado e o resolveremos com uma abordagem da econometria e em seguida com uma abordagem de aprendizado de máquina. Nossa intenção aqui é fornecer uma transição da econometria ao aprendizado de máquina que seja simples e intuitiva, mostrando como as linhas que as separam podem ser facilmente cruzadas.

Para tanto, considere o problema de estimar a produção de um estudante em termos de páginas escritas. Para manter a simplicidade, nós vamos considerar que o número de páginas escritas só dependa da quantidade de horas que o estudante trabalhou. Para facilitar o entendimento, vamos utilizar dados simulados da seguinte forma (Figura 1):

$$paginas = \log(2 * horas + 1) \tag{1.1}$$

Dizemos que essa equação é a função geradora de dados. Trata-se de uma fórmula matemática que representa um processo no mundo. Como cientistas ignorantes, não podemos ter certeza de como é essa função geradora de dados, mas sabemos que ela se manifesta na realidade e podemos coletar essas manifestações na forma de dados. No entanto, para dificultar nossa tarefa, os dados que conseguimos coletar estão sempre corrompidos com alguma forma de ruído (Figura 2), tornando ainda mais nebulosa a forma como o número de páginas escritas está relacionado com as horas trabalhadas. Esses ruídos podem ser resultados de erros de sensores ou digitação, variáveis relevantes omitidas ou aleatoriedades inerentes ao processo estudado.

Nossa tarefa então será entender a relação entre horas trabalhadas e páginas produzidas a partir dos dados. Para isso, vamos utilizar tanto econometria como aprendizado de máquina e assim perceber as diferenças e semelhanças entre essas duas ciências. Uma consideração final que devemos ressaltar é que esse problema ilustrativo é bastante simples, pois só envolve duas variáveis. Na prática, isso quase nunca será o caso e sequer saberemos se os dados que dispomos são suficientes para entender a nossa variável de interesse.

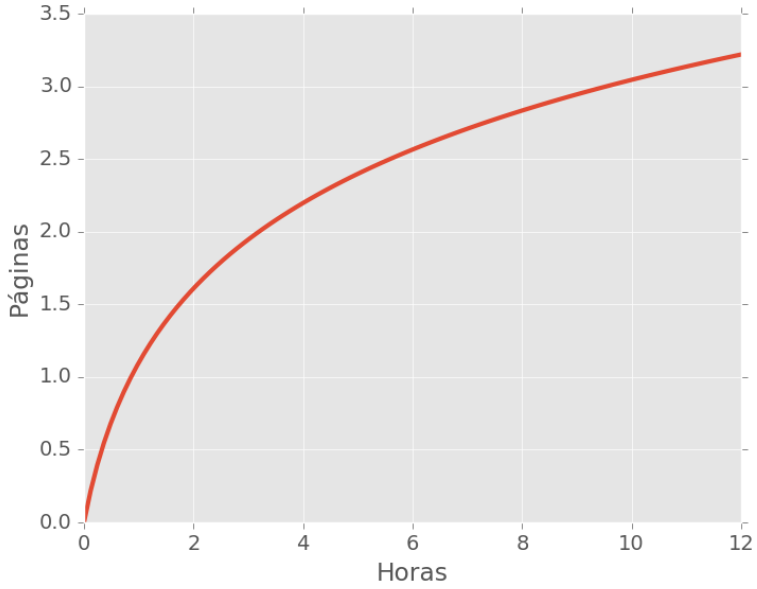

Figura 1 – Curva de páginas por horas trabalhadas criada a partir da equação 1.1. A relação expressa nesta curva mostra que conforme trabalhamos mais em um mesmo dia, nossa produtividade vai caindo.

1.2 O Modelo Econométrico Clássico: Regressão Linear

Em econometria, o modelo mais utilizado é o de regressão linear múltipla. Formalmente, seja y nossa variável dependente, seja $\boldsymbol{x}$ um vetor com os fatores que afetam y e cujo primeiro componente é 1, seja $\boldsymbol{w}$ um vetor de parâmetros com o mesmo número de dimensões de $\boldsymbol{x}$ e seja u um fator erro, o modelo de regressão linear múltipla é definido como:

$$y = \boldsymbol{x}\boldsymbol{w} + u \tag{1.2}$$

Intuitivamente, podemos pensar no modelo de regressão linear múltipla como uma soma ponderada, na qual cada variável x é ponderada de acordo com uma força w para produzir y. Podemos estimar esse tipo de modelo utilizando os dados. Nesse caso, nossa estimativa seria da forma

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{w}} \tag{1.3}$$

Na qual $\boldsymbol{X}$ é uma matriz de dados, com a primeira coluna sendo $\mathbf{1}$ e as demais representando as variáveis que afetam y. Seja o resíduo definido como $\epsilon = y - \hat{y}$, então estaríamos interessados em encontrar $\hat{\boldsymbol{w}}$ tal que soma dos resíduos quadrados fosse mínima:

$$\min_{\hat{\boldsymbol{w}}} ||\boldsymbol{\epsilon}||^2 \tag{1.4}$$

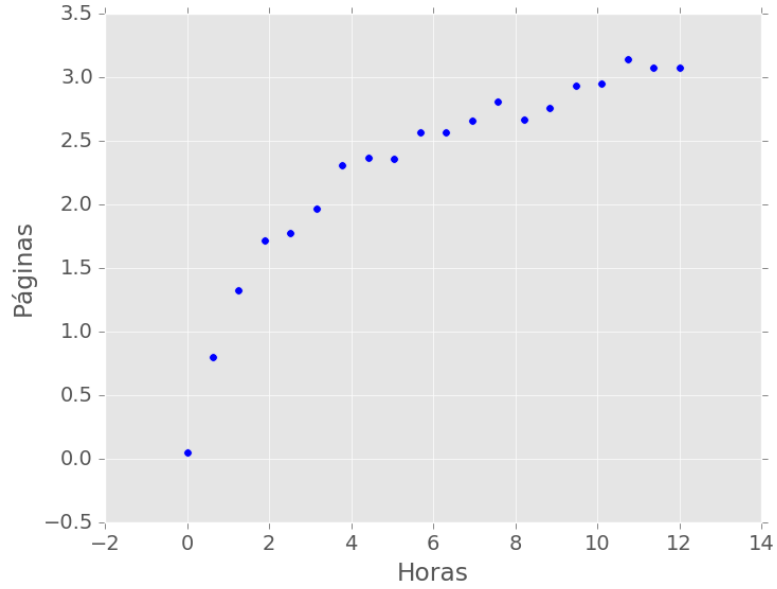

Figura 2 – Curva de páginas por hora trabalhada criada a partir da equação 1.1. A relação expressa nesta curva mostra que conforme trabalhamos mais em um mesmo dia, nossa produtividade vai caindo.

Nesse caso, estamos estabelecendo uma função custo $L(\hat{\boldsymbol{w}}) = ||\boldsymbol{\epsilon}||^2$. Do ponto de vista estatístico, minimizar essa função custo significa achar o estimador de máxima verossimilhança para $E(\boldsymbol{y}|\boldsymbol{X})$. Uma forma de achar o $\hat{\boldsymbol{w}}$ ótimo seria primeiro encontrar a derivada de $||\boldsymbol{\epsilon}||^2$ com respeito a $\hat{\boldsymbol{w}}$ e então atualizar $\hat{\boldsymbol{w}}$ na direção oposta, iterativamente. Outra forma seria resolver o problema analiticamente e chegaríamos em $\hat{\boldsymbol{w}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}\boldsymbol{y}$. De todo modo, em econometria estamos mais interessados em como interpretar o $\hat{\boldsymbol{w}}$ estimado do que em como estimá-lo.

Voltando ao nosso caso ilustrativo de páginas escritas por horas trabalhadas, nós poderíamos estimar um modelo linear a partir dos dados. Se o especificássemos como $paginas = w_0 + w_1 horas + u$, a forma estimada desse modelo seria:

$$\underset{}{\widehat{paginas}} = \underset{(0,15645)}{1,11549} + \underset{(0,02229)}{0,19639} * horas \tag{1.5}$$

Nós então podemos dividir os parâmetros estimados pelo erro padrão (embaixo, entre parenteses) para obter uma estatística t e verificar que ambas as estimativas são estatisticamente significantes, com p-valores próximos de zero. Podemos também verificar o R^2 desse modelo e obteríamos $0,8118$, indicando que mais de 80% da variação em y pode ser explicada pela equação estimada. Interpretando esses resultados, poderíamos dizer com um elevado grau de certeza que a cada uma hora a mais que este estudante trabalha, ele produz 0,196 página a mais. Finalmente, nós poderíamos colocar no gráfico a linha de melhor ajuste estimada.

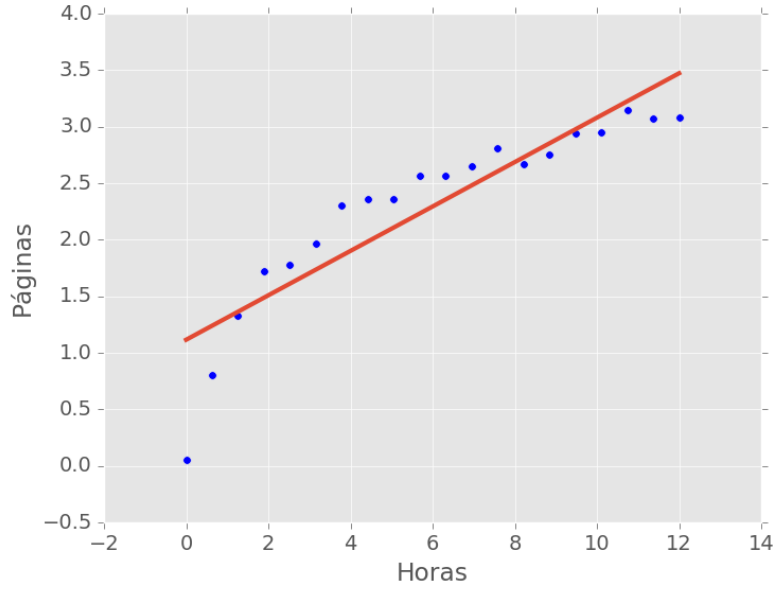

Figura 3 – Reta de melhor ajuste produzida a partir de 1.5.

Quando fazemos isso, percebemos que algo do nosso modelo está fundamentalmente errado, mesmo que todas as estatísticas apontem para ótimos resultados de previsão e estimativa. Para entender como surgiu esse erro, precisamos voltar nossa atenção a uma hipótese bastante forte assumida pelo modelo.

Ao modelar y na forma $\boldsymbol{xw}$, nós estamos assumindo que as variáveis consideradas influenciam y de forma linear. De um ponto de vista mais geral, isso é problemático, pois nas ciências humanas e sociais as relações lineares entre variáveis são extremamente raras, se não inexistentes. Muito mais comum são as relações marginais decrescentes ou crescentes, nas quais a variação de y muda para cada valor de x. Por exemplo, no nosso caso ilustrativo de páginas escritas por hora trabalhada, podemos dizer que a produtividade do estudante é marginalmente decrescente com as horas trabalhadas, pois a cada hora a mais de trabalho, a quantidade total de páginas produzidas cresce cada vez menos. A hipótese de que as variáveis afetam y de maneira linear é referida em econometria como uma das hipóteses Gauss-Markov e é fundamental para mostrar que as estimativas dos parâmetros $\boldsymbol{w}$ são não enviesadas ou consistentes. Dizemos que violações dessa hipótese são casos de má especificação da forma funcional do modelo. (WOOLDRIDGE, 2012). Infelizmente, não há maneira simples nem para detectar esse problema e nem para solucioná-lo. No nosso exemplo, estamos trabalhando com apenas uma variável, então é relativamente fácil ver que a forma funcional do nosso modelo está errada, pois podemos colocar os dados e as previsões em um gráfico, como fizemos na figura 3. No entanto, na maioria das vezes estaremos trabalhando com mais variáveis, o que impossibilitaria a visualização.

1.2.1 Modelando Não Linearidades à Mão

Para resolver esse problema, econometristas gastam uma grande quantidade de tempo desvendando a forma funcional correta do processo gerador de dados. Muitas vezes esse processo pode levar anos e envolver extensos debates acadêmicos e controvérsias, como por exemplo no caso de estimar a função de crescimento de um pais. Quando o problema é menos complexo, o que se recomenda (WOOLDRIDGE, 2012) é acrescentar variáveis que são transformações quadráticas ou logarítmicas das originais. Isso torna possível ajustar uma parábola aos dados, em vez de uma reta.

De qualquer forma a solução atual para estimar funções não lineares é desenhar à mão novas variáveis que são combinações ou transformações não lineares das originais. O grande desafio então torna-se como desenhar tais variáveis de forma a capturar bem a função geradora de dados. Note que essa abordagem é nada menos do que tentar desvendar a própria forma funcional do processo que se tenta modelar. Voltando ao nosso exemplo de páginas escritas por hora trabalhada, nós poderíamos seguir as recomendações de Wooldridge (2012) e desenhar variáveis polinomiais. Por exemplo, nós poderíamos mudar o nosso modelo de $y = b + xw + u$ para

$$y = b + w_2 x + w_2 x^2 + u \tag{1.6}$$

Assim, a forma estimada desse modelo quadrático então seria

$$\widehat{paginas} = \underset{(0,125663)}{0,563153} + \underset{(0,048539)}{0,487903} * horas - \underset{(0,003905)}{0,024292} * horas^2 \tag{1.7}$$

Novamente, todos os parâmetros estimados são significantes com um p-valor próximo de zero, o que garante que a inclusão da nova variável ($horas^2$) faz sentido. Mais ainda, agora temos $R^2 = 0,9426$, indicando que essa nova forma funcional explica mais a variação na nossa variável dependente do que o modelo anterior. Por fim, quando colocamos esse segundo modelo no gráfico (figura 4), podemos ver que ele se ajusta muito melhor aos dados do que o primeiro.

Ainda assim, podemos perceber alguns pontos fundamentalmente errados no modelo quadrático. Em primeiro lugar, a forma funcional de um polinômio de grau dois pressupõe um ponto de inversão no sinal da inclinação. Nesse caso, seria um ponto ótimo, a partir do qual mais horas trabalhadas implicariam em menos páginas produzidas. No modelo estimado, esse ponto acontece em $x = 10,0425$. Como economistas, nós ou teríamos que interpretar isso como sendo verdade, arrumando alguma explicação para produzir menos com mais horas de trabalho, ou teríamos que evadir o problema alegando poucos dados nesse ponto ótimo, tornando os resultados não confiáveis. Outro aspecto fundamentalmente errado desse modelo estimado é que ele prevê que $0,563$ páginas serão produzidas mesmo

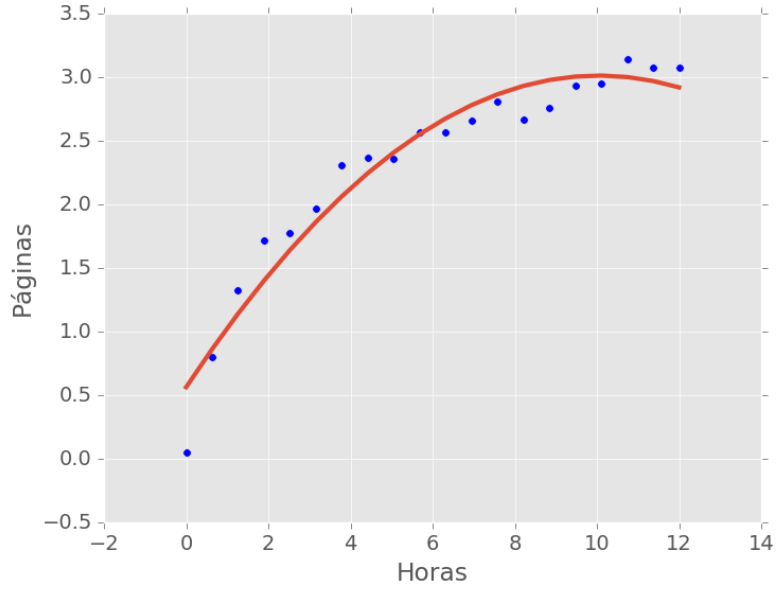

Figura 4 – Parábola de melhor ajuste produzida a partir de 1.6.

se o estudante não trabalhar nada. Aqui, nossa única saída seria alegar poucos dados no extremo inferir da curva para justificar essa previsão absurda.

De qualquer forma, ambos os problemas surgem pois não fomos capazes de identificar a forma funcional exata da função geradora de dados $paginas = \log(2 * horas + 1)$. Note que, nesse caso essa função geradora de dados é extremamente simples, mas mesmo assim é preciso desprender uma grande quantidade de raciocínio para capturá-la apenas aproximadamente. Uma grande fraqueza do método de regressão linear é assumir que a relação entre variáveis dependentes e independente seja linear; essa hipótese é demasiadamente restritiva e vamos relaxá-la com os modelos de aprendizado de máquina.

1.3 Aprendizado de Máquina Clássico

Aprendizado de máquina surgiu como um avanço no campo de inteligência artificial, permitindo que os sistemas adquirissem sua própria forma de conhecimento ao extrair padrões a partir de dados (GOODFELLOW; BENGIO; COURVILLE, 2016, p. 2). De maneira bastante informal, aprendizado de máquina pode ser entendido como um campo da estatística aplicada, com "ênfase crescente no uso de computadores para estimar estatisticamente funções complicadas e interesse decrescente em prover intervalos de confiança em torno dessas funções"[1]. Embora esse não seja o único caso em que aprendizado de máquina possa ser útil, vamos partir do recorte em que essa ciência é utilizado para resolver problemas muito complexos, nos quais a expertise humana é ausente ou fraca.

[1] Isso tem mudado nos últimos anos com o advento de métodos Bayesianos para aprendizado de máquina (KINGMA; WELLING, 2013; GAL; GHAHRAMANI, 2015; GAL, 2016)

Para melhor entender como essa ciência se diferencia e se relaciona com econometria, precisamos antes deixar claro qual é o objetivo de aprendizado de máquina, isto é, precisamos entender o que significa fazer com que a máquina aprenda. Em primeiro lugar, devemos perceber que a tarefa de aprendizado envolve descobrir uma regra geral a partir de alguns exemplos coletados na forma de dados. Do ponto de vista lógico, isso é contraditório, pois não podemos inferir regras gerais a partir de uma quantidade limitada de exemplos (GOODFELLOW; BENGIO; COURVILLE, 2016). Então tornamos nossa atenção a um problema mais simples: aprender uma regra que está aproximadamente correta, na maioria dos casos. Em termos mais técnicos, em aprendizado de máquina, queremos que nossos sistemas consigam aprender uma função que é provavelmente aproximadamente correta (VALIANT, 1984). Nesse contexto, a função estimada deve ter um erro baixo o suficiente (para estar "aproximadamente correta") mais ainda precisa que isso acontece na maioria dos casos, ou seja, não basta que desenhemos um modelo para minimizar o erro em uma base de dados específica; o que precisamos é que nosso modelo consiga erros pequenos consistentemente, mesmo em bases de dados que não foram utilizadas para estimar o modelo.

As palavra chaves que definem os objetivos de aprendizado de máquina são *representação* e *generalização*. Em primeiro lugar, os considerados melhores modelos de aprendizado de máquina têm o potencial para representar qualquer interação entre variáveis (o que não significa que, na prática, o modelo aprenderá tal relação a partir dos dados). Em termos mais técnicos, podemos dizer que os melhores modelos de aprendizado de máquina são aproximadores universais de funções. Em segundo lugar, passando agora mais para as considerações práticas, os bons modelos de aprendizado de máquina devem conseguir generalizar relações aprendidas em uma amostra de dados para cenários e aplicações reais.

Assim, podemos ver como aprendizado de máquina difere de econometria. Com exceção dos estudos em *forcasting*, em trabalhos de econometria, a maior preocupação recai em, normalmente, minimizar o erro na mesma base de dados utilizada para estimar (ou treinar) o modelo econométrico. Aprendizado de máquina, por outro lado, está sempre mais preocupado em minimizar o erro em todas as bases de dados, observadas ou não. Esse erro que queremos minimizar em aprendizado de máquina leva o nome de erro de generalização. Trata-se de um conceito puramente teórico, pois é impossível saber qual seria o erro em todos os exemplos possível. Ainda assim, aqui, daremos ênfase aos problemas cujas estimativas desse erro são relativamente simples de computar, como nos casos de aprendizado de máquina supervisionado.

1.3.1 Aprendizado Supervisionado

Aprendizado supervisionado é quando queremos prever uma variável y que depende de outras variáveis $\boldsymbol{x}$. Além disso, ambos $\boldsymbol{x}$ e y são fatores observáveis, que podem ser

coletados na forma de dados. Nesse cenário, mostramos à máquina as variáveis $\boldsymbol{x}$ e y correspondentes. Então pedimos que ela reproduza ou preveja y a partir de $\boldsymbol{x}$. A nossa esperança é que, após ser apresentada a vários exemplos de pares $(\boldsymbol{x}, y)$, a máquina consiga prever bem y a partir de observações que nunca viu, dada as variáveis $\boldsymbol{x}$ dessas novas observações. Isso é chamado de aprendizado supervisionado pois podemos traçar uma analogia com a noção de um aprendiz - o computador - sendo supervisionado por um professor - o programador - que lhe fornece vários exemplos de como realizar corretamente uma tarefa: mapear $\boldsymbol{x}$ em y.

Matematicamente, nos queremos mover de uma estimativa incondicional de y, geralmente a esperança $E[y]$, para uma estimativa condicional de y, geralmente a esperança de y dado $\boldsymbol{x}$, $E[y|\boldsymbol{x}]$. Fazemos isso usando algum modelo para estimar $f(\boldsymbol{x}) = y + \epsilon$, em que ϵ é um ruído aleatório. Quase sempre esse problema é posto explicitamente como um problema de otimização, no qual minimizamos um erro que mede a diferença entre a nossa estimativa de $f(\boldsymbol{x})$ (normalmente chamada de $\hat{y}$) e y de fato observado. Um grande problema, no entanto, é que o erro explicitamente minimizado é aquele relativo a base de dados utilizada para estimar o modelo, mas o que nos interessa é o erro relativo a uma nova base de dados - a métrica de performance P - , não vista durante o treinamento. Assim, em contraste com otimização pura, em aprendizado de máquina nós minimizamos uma função custo na esperança de que isso melhorará nossa performance preditiva de acordo com a métrica P (GOODFELLOW; BENGIO; COURVILLE, 2016), que geralmente distinta da função custo por ser mair geral.

Podemos identificar dois tipos de problemas dentro do regime de aprendizado supervisionado: regressão e classificação. Problemas de regressão são aqueles em que queremos prever um valor contínuo, como renda, peso, quantidade demandada, ângulo da direção de um carro automático ou quando acontecerá a promoção de um produto. Problemas de classificação são aqueles em que queremos prever um valor discreto, ou seja classificar um exemplo segundo uma categoria. Alguns exemplos são identificar a presença de uma doença dado os sintomas do paciente, prever se o preço da ação de uma empresa vai subir ou cair dado o histórico do mercado financeiro, identificar de que pessoa é a face em uma imagem ou classificar um livro em uma escola literária.

A maioria dos trabalhos em aprendizado de máquina são feitas sob o regime de aprendizado supervisionado, pois são bem definidos e apresentem bastante utilidade prática. Alguns dos algoritmos que podem ser usados para resolver esse tipo de problema são regressão linear, regressão logística, árvores de decisão, florestas aleatórias, máquinas de suporte vetorial, k-vizinhos mais próximos, Bayes ingênuo e redes neurais artificiais. Em suma, qualquer problema em que se busca prever uma variável y a partir de variáveis $\boldsymbol{x}$ tem o potencial para ser resolvido com aprendizado de máquina supervisionado.

1.3.1.1 Capacidade e generalização

Capacidade e generalização são as duas qualidades que gostaríamos que nossos modelos de aprendizado de máquina adquirissem: a primeira lhes da força para aprender as regularidades nos dados em que treinamos o modelo; a segunda faz com que consigam generalizar o que aprendeu para dados novos. Infelizmente essas duas forças então em polos opostos, de forma que ter mais de uma geralmente significa perder mais da outra. A seguir, vamos detalhar bem como esse *trade-off* acontece e como ponderar essas duas forças.

Considere novamente o problema de prever o número da páginas escritas dado o número de horas trabalhadas. Com os mesmos dados de antes, nós treinamos três modelos de aprendizado de máquina com capacidade diferente (figura 5). A pergunta natural que fica é então qual dos três modelos é o melhor?

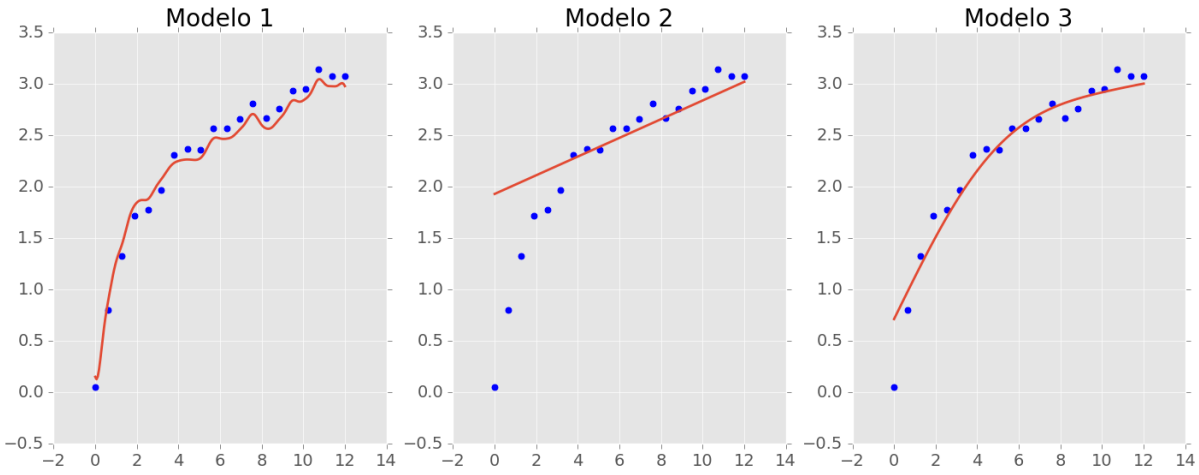

Figura 5 – Três modelos de aprendizado de máquina com capacidade diferentes. O Modelo 1 é o de maior capacidade, o Modelo 2 é o de menor e o Modelo 3, o de capacidade intermediária

O primeiro modelo tem alta capacidade e se ajusta muito bem aos dados, capturando até as pequenas variações. De fato, o ajustamento é tão bom que o R^2 é $0,98$. O segundo modelo, por sua vez, tem baixa capacidade e não consegue capturar nenhuma curvadura nos dados. Ele não consegue se ajustar muito bem aos dados, de forma que o R^2 dele é apenas $0,52$. Por fim, o último modelo tem uma capacidade intermediária. Podemos ver que ele consegue capturar alguma curvatura nos dados, mas não todas. Seu R^2 também é intermediário, sendo $0,94$.

Mesmo que tenhamos reportado o R^2, uma métrica de performance, para os três modelos, isso ainda não é suficiente para saber qual deles é melhor. Devemos lembrar que nosso objetivo é minimizar o erro de generalização, isto é, o erro em todas as bases de dados, incluindo aquelas que não foram utilizada para estimar o modelo. Para estimar

esse erro, precisamo dividir nosso dados em duas subamostras. A primeira delas leva o nome de set de treino e será utilizada para estimar (ou treinar) o modelo. A segunda delas será utilizada para avaliar o modelo, ou seja, para conseguir uma estimativa do erro de generalização, já que este será calculado em dados ainda não vistos pelo modelo. Essa subamostra levará o nome de set de teste.

Voltando ao nosso exemplo, podemos coletar mais 20 observações de pares (x, y) para servirem como set de teste. Então podemos ver como as previsões do nosso modelo nesse novo set de dados se comparam com os valores de páginas escritas realmente observado (figura 6).

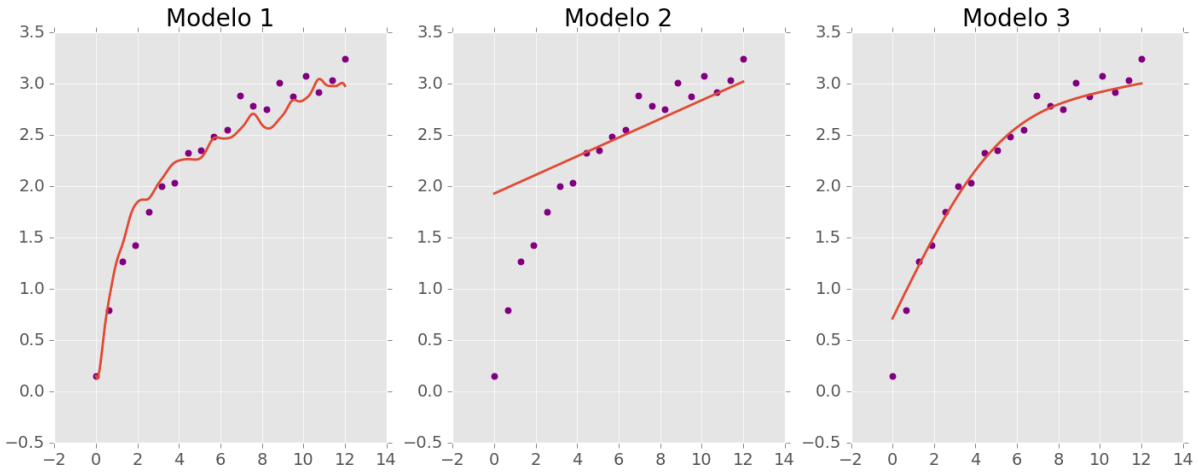

Figura 6 – As previsões dos mesmos três modelos (linha vermelha) comparadas com os dados da base de teste (não utilizados para treinamento).

Nessa nova base de dados, podemos calcular novamente o R^2 e obtemos $0,94$ para o primeiro modelo (com maior capacidade), $0,52$ para o segundo modelo (com menor capacidade) e $0,96$ para o terceiro (com capacidade intermediaria). Como essas estatísticas foram calculadas em uma base de dados não utilizada para treinamento do modelos, elas são uma estimativa melhor do erro de generalização. Assim, agora podemos afirmar que o terceiro modelo é o melhor dos três. Isso não é tão surpreendente se invocarmos o princípio da navalha de Occam, segundo o qual, dentre dois modelos de performance semelhante, o mais simples é o melhor. Em termos mais modernos, podemos dizer que esse princípio está refletido no *trade-off* entre generalização e capacidade, uma vez que que modelos com mais desta última tendem a ser mais complexos.

No nosso exemplo, nota-se que o modelo com grande capacidade é extremamente complexo, com várias curvaturas; ao passo que o modelo com capacidade intermediária é bem mais simples. Assim, essa preferência por modelos mais simples também justifica a escolha do terceiro como sendo o melhor. Mais formalmente, nós podemos decompor o erro da estimativa de máxima verossimilhança em duas fontes de erro. Seja θ o parâmetro

real de uma distribuição de dados e $\hat{\theta}$ a sua estimativa, temos que:

$$\begin{aligned}
EQM(\hat{\theta}) &= \mathbb{E}[(\hat{\theta} - \theta)^2] \\
&= \mathbb{E}[(\hat{\theta} - \mathbb{E}[\hat{\theta}] + \mathbb{E}[\hat{\theta}] - \theta)^2] \\
&= \mathbb{E}[(\hat{\theta} - \mathbb{E}[\hat{\theta}])^2] + \mathbb{E}[2(\hat{\theta} - \mathbb{E}[\hat{\theta}])(\mathbb{E}[\hat{\theta}] - \theta)] + \mathbb{E}[(\mathbb{E}[\hat{\theta}] - \theta)^2] \\
&= \mathbb{E}[(\hat{\theta} - \mathbb{E}[\hat{\theta}])^2] + 2(\mathbb{E}[\hat{\theta}] - \theta)(\mathbb{E}[\hat{\theta}] - \mathbb{E}[\hat{\theta}]) + (\mathbb{E}[\hat{\theta} - \theta])^2 \\
&= \mathbb{E}[(\hat{\theta} - \mathbb{E}[\hat{\theta}])^2] + (\mathbb{E}[\hat{\theta} - \theta])^2 \\
&= Var(\hat{\theta}) + Vies(\hat{\theta}, \theta)^2
\end{aligned}$$

Mais intuitivamente, sabemos que o viés mede o quanto a esperança de uma estimativa difere do valor real do parâmetro que se quer estimar. Pela equação acima, podemos ver que o viés contribui com ordem quadrada para o erro quadrático médio. Idealmente, as estimativas do nosso modelo serão não enviesadas, isto é, na média, serão iguais aos parâmetros da distribuição real de dados. A variância, por outro lado, mede o quanto nossas estimativas mudam de acordo com as particularidades da amostra considerada para treinar o modelo. Também gostaríamos que a variância fosse baixa, para que nossas estimativas não dependam muito do erro amostral.

Infelizmente, o *trade-off* entre capacidade e generalização se reflete formalmente em um *trade-off* entre viés e variância. O primeiro modelo do nosso exemplo tem grande capacidade, o que se reflete em uma alta variância. Isso acontece pois o modelo tem mais capacidade do que a necessária para aprender as regularidades nos dados e usa a capacidade extra para aprender ruído, presente devido ao erro de amostragem. Em aprendizado de máquina, dizemos que modelos que sofrem de alta variância estão sobre-ajustados ou em um regime de sobre-ajustamento. Nesse regime, o modelo costuma ter uma performance extremamente alta no set de treino, mas baixa em um set de dados não utilizado para treinamento.

O segundo modelo do nosso exemplo é extremamente simples e tem pouca capacidade, o que se reflete em maior erro por elevado viés. Essa baixa capacidade faz com que o modelo não aprenda as regularidades nos dados, muito menos o ruído. Por outro lado, o erro ao qual ele incorre na amostra de treinamento é similar ao erro referente aos dados de teste, o que indica que o modelo consegue generalizar o que aprendeu, mesmo que isso não seja grande coisa. Em outras palavras, se esse modelo erra, pelo menos ele erra consistentemente, independente do erro de amostragem.

Se decompormos o erro de generalização nas suas duas fontes de erro, isto é, viés e variância, um típico modelo de aprendizado de máquina terá um erro se comportando como na figura 7. Em suma, dada uma quantidade fixa de dados de treinamento, um modelo com baixa capacidade tende a sofrer mais com viés e menos com variância. A medida que a capacidade vai aumentando, o viés cai exponencialmente enquanto que a

variância aumenta linearmente. O ponto de capacidade ótima pondera viés e variância pela capacidade de tal forma que o erro de generalização seja mínimo.

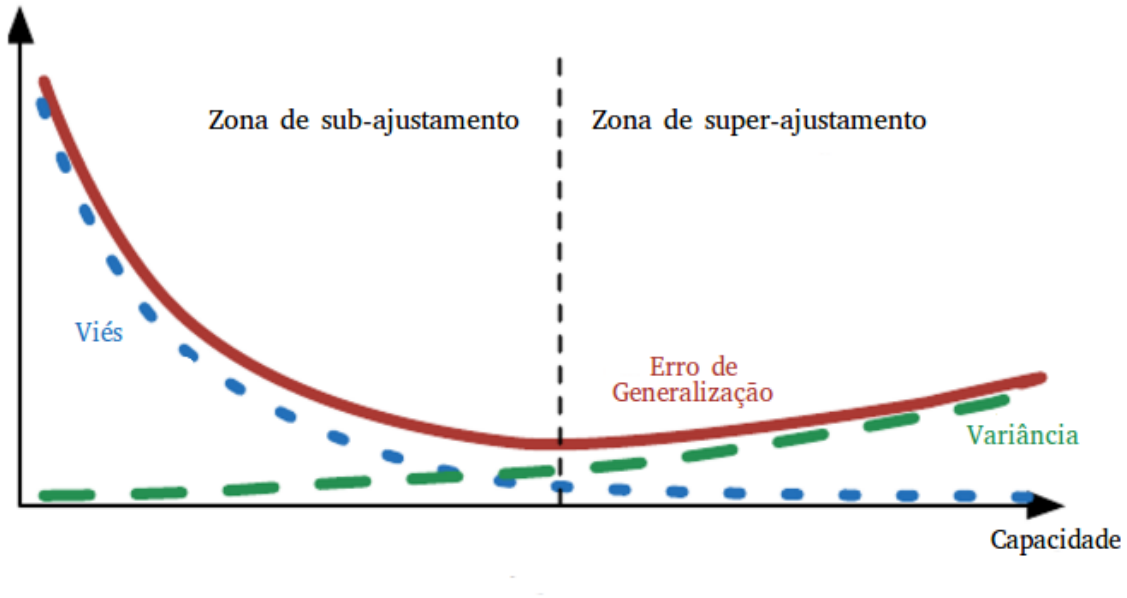

Figura 7 – imagem traduzida do livro *Deep Learning* (GOODFELLOW; BENGIO; COURVILLE, 2016)

1.3.1.2 Validação Cruzada

Como vimos acima, não podemos confiar em métricas de erro computadas nos sets de treinamento para escolher entre modelos, pois é sempre possível reduzir esse erro a zero simplesmente aumentando a capacidade do algoritmo de aprendizado. Há no entanto um método extremamente simples para selecionar o modelo com menor erro de generalização - o que realmente nos interessa. Após coletar os dados, vamos separá-los em duas subamostras. Uma delas, a sub-amostra de treino, será utilizada para estimar um modelo; a outra subamostra será utilizada para medir a performance do modelo, que será uma estimativa do erro de generalização.

Para escolher a capacidade do modelo, dividiremos a subamostra de treino mais uma vez: em um novo set de treino e em um set que chamaremos de validação. Então treinaremos o modelo no set de treino, ajustaremos a capacidade do algoritmo de aprendizado com base na performance set de validação e reportaremos uma estimativa final do erro de generalização conforme a performance no set de teste. É importante que o set de teste seja observado apenas uma única vez, na hora de reportar a estimativa de erro final. Se múltiplas tentativas forem feitas e comparadas com base no erro do set de teste, esta medida de erro não será confiável como uma estimativa do erro de generalização e o modelo provavelmente performará pior do que o esperado, quando utilizado na prática.

Qualquer tipo de exploração, ajuste ou avaliação realizada precocemente na subamostra de teste invalida a validação final do modelo pois introduz viés por *data snooping* (também conhecidos como *data dredging* e *p-hacking*)

Essa técnica de escolha de modelos leva o nome de validação cruzada. Ela pode parecer óbvia e simples, mas atualmente desconsiderada pela maioria dos trabalhos em economia. Como resultado, não é raro encontrar estudos que validem uma relação utilizando uma certa base de dados e trabalhos que invalide a mesma relação utilizando outra base de dados. A econometrista Susan Athey, que atua na área de aprendizado de máquina em economia e foi premiada com a medalha John Bates Clark pelo seu trabalho, chegou a afirmar que validação cruzada é talvez a maior contribuição que a ciência de aprendizado de máquina pôde fazer às ciências sociais.

1.3.2 Limitações e Desafios

Com aprendizado de máquina podemos superar o fardo de ter que adivinhar a forma funcional do processo gerador de dados. Podemos facilmente mapear automaticamente uma lista de variáveis $(x_1, x_2, ..., x_k)$ em uma variável dependente y que queremos prever. Mais importante, independentemente da relação entre variáveis que queremos desvendar, se tivermos dados e tempo suficientes, um modelo de aprendizado de máquina conseguirá aprendê-la. No entanto, ainda restam alguns desafios que valem a pena serem destacados.

Do ponto de vista prático, os algoritmos de aprendizado de máquina clássicos ainda são muito dependente de engenharia de variáveis. Na maioria dos casos, precisamos utilizar nosso conhecimento sobre o assunto que estamos estudando para criar novas variáveis, representativas do nosso problema em questão. Esse processo de engenharia de variáveis pode ser entendido como uma etapa ainda necessária, em que o pesquisador ou praticante de aprendizado de máquina precisa injetar alguma forma de conhecimento ou abstração ao sistema para que esse consiga ser inteligente.

Por exemplo, considere o problema de prever se a ação de uma empresa vai subir ou cair a partir de *tweets* sobre essa empresa. Nesse caso, os dados são simplesmente palavras, mas pode ser extremamente útil representar os *tweets* como um vetor de frequência de cada palavra, ou melhor ainda, um vetor de frequência de palavras no *tweet* normalizado pela frequência de palavras em todos os *tweet*. Com efeito, o que estamos fazendo nesse caso ao utilizar aprendizado de máquina é, ainda, uma etapa não automatizada de pré-processamento. A automação dessa etapa será tratada na próxima sessão, quando introduziremos o aprendizado de representações.

Um outro desafio se refere ao problema de inferência. Em economia, estamos frequentemente mais interessados em traçar relações entre variáveis independentes e dependentes do que prever a resposta desa última a partir das primeiras. Isso pode ser

bastante desafiador até mesmo em modelos de econometria um pouco mais complexos, como o de regressão logística. No momento em que deixamos o regime de linearidade, a tarefa de inferência torna-se extremamente complicada, já que é possível que a relação entre as variáveis muda para cada intervalo considerado. Em termos formais, quando estamos estudando fenômenos não lineares, o gradiente de y - nossa variável dependente - com respeito ao vetor de variáveis $\boldsymbol{x}$ pode mudar de maneira não intuitiva. Em aprendizado de máquina, como as relações aprendidas podem ser extremamente não lineares, a tarefa de inferência evolui para um desafio um tanto complicado. As derivadas (ou qualquer outro método de saber a importância de cada x para determinar y) são inacessíveis na maioria dos algoritmos de aprendizado de máquina ou são extremamente difíceis de interpretar.

2 *Deep Learning*

Apesar de seu tremendo sucesso só ter se dado recentemente, o campo de *Deep Learning* tem uma história que data desde da década de 1940. Podemos entender *Deep Learning* como inserido no contexto de modelos computacionais inspirados na dinâmica do cérebro, muito embora eles não se pretendem ser representações realísticas de funções biológicas. Esses modelos são geralmente conhecidos como **redes neurais artificiais**.

Ao longo da hisória, sua popularidade evoluiu em três ondas (GOODFELLOW; BENGIO; COURVILLE, 2016). A primeira, sob o nome de *cibernética*, teve seu ápice em 1970 com teorias de aprendizado biológico e com o desenvolvimento do *perceptron* por Rosenblatt (1958). Esses modelos eram uma versão simplificada do que hoje conhecemos como regressão logística. No entendo, devido às limitações representativas pela linearidade do modelo (MINSKY; PAPERT, 1969), os modelos biologicamente inspirados desvaneceram nos anos 80, o que ficou conhecido como o inverno da inteligência artificial. A segunda onda, sob o nome de *conectivismo*, surgiu com os avanços possibilitados pelo algoritmo de *backpropagation* (RUMELHART; HINTON; WILLIAMS, 1986). Ela durou até a metade dos anos 90, quando empreendimentos baseados em redes neurais começaram a fazer promessas não realistas na busca de investimentos (GOODFELLOW; BENGIO; COURVILLE, 2016). O não cumprimento dessas promessas concomitantemente ao surgimento de outros avanços em aprendizado de máquina novamente levaram ao um ostracismo das redes neurais que até 2007.

Até recentemente, redes neurais, embora teoricamente promissoras, eram simplesmente muito difíceis de treinar, relativamente a outras classes de modelos, como máquinas de *kernels* ou *SVMs*. Na segunda metade dos anos 2000, Hinton, Osindero e Teh (2006) mostraram como treinar redes neurais eficientemente a partir de modo semi-supervisionado. Rapidamente, grupos filiados ao CIFAR (*Canadian Institute for Advanced Research*) estenderam esse método de treinamento para vários tipos de redes neurais. Isso desencadeou a terceira onda, sob o nome de *Deep Learning*, quando redes neurais começaram a superar sistematicamente a performance de outros modelos de aprendizado de máquina (GOODFELLOW; BENGIO; COURVILLE, 2016). No momento desta escrita, essa terceira onda de popularidade continua como um ciclo virtuoso, em que avanços em modelos de *Deep Learning* levam a resolução de problemas complexos de inteligência artificial, atraindo investimentos bilionários para pesquisa, levando a novos avanços.

Hoje, a anterior dificuldade de treinar redes neurais é atribuída (GOODFELLOW; BENGIO; COURVILLE, 2016) mais às limitações tecnológicas do que problemas fundamentais do modelo ou de sua otimização. Até recentemente, simplesmente não tínhamos

dados suficiente para que redes neurais generalizassem, nem poder computacional para processá-los ou para treinar grandes modelos. Mais ainda, a relação entre os modelos computacionais e a neurociência foi largamente posta de lado. Da mesma forma que antigamente nos inspiramos em pássaros para fazer máquina voadoras, a conexão entre redes neurais e o cérebro é atualmente vista como apenas um primeiro passo no desenvolvimento de máquinas inteligentes (GÉRON, 2017).

2.1 Aprendizado de Representações

Como visto no primeiro capítulo, a performance de algoritmos de aprendizado de máquina clássicos depende fortemente da **representação** dos dados sob uma forma informativa. Esse é um fenômeno geral, que circunda uma boa parte das tarefa envolvendo inteligência. Como um exemplo, é muito mais fácil fazer aritmética com números arábicos do que com a representação numérica romana. Assim, sendo, uma grande parte do esforço em aprendizado de máquina é desprendida achando características e variáveis e representativas do fenômeno que ser quer modelar. O grande problema é que nem sempre é fácil ou possível encontrar tais representações manualmente.

Os modelos de *Deep Learning*, por outro lado, são capazes de aprender não apenas a relação de variáveis representativas com uma variável dependente, mas também conseguem aprender uma representação em si (GOODFELLOW; BENGIO; COURVILLE, 2016). Isso é feito pela extração dos fatores de variação que melhor explicam o fenômeno que se quer modelar. Por esse motivo, tarefas envolvendo *Deep Learning* também levam o nome de **aprendizado de representações**. De uma forma geral, o termo *Deep Learning* está associado à representar a realidade em camadas de hierárquicas abstração. O exemplo mais clássico disso é quando uma rede neural profunda é treinada para reconhecimento de imagens. Nesse caso, a forma original dos dados - pixeis - é extremamente bruta relativa aos conceitos de semântica visual que se quer aprender. Assim, para visão computacional, as primeiras camadas de uma rede neural mapeiam os pixeis em algo mais abstrato, como a noção de quinas e diferenças de contraste. Construindo em cima dessas representações, camadas mais ao meio da rede aprendem como abstrair dessas quinas e contrastes em contornos ou formas simples. Finalmente, camadas mais ao fim da rede neural abstraem ainda mais, mapeando das formas simples e contornos em partes de objeto com carga semântica, como a cabeça de uma pessoa, o nariz de um cachorro ou a transparência em um copo d'água (MORDVINTSEV; OLAH; TYKA, 2015).

Até agora, aprendizado de representações tem se mostrado um conceito poderoso na resolução de tarefas complexas de IA, notadamente aquelas em que nós humanos realizamos facilmente mas em que computadores têm imensa dificuldade em executar. Alguns exemplos são enxergar, ler, escrever e conversar. Por outro lado, ainda há muito

o que ser explorado para que possamos afirmar se capacidade de abstração é também é fundamental para resolver tarefas onde a expertise humana é fraca ou ausente.

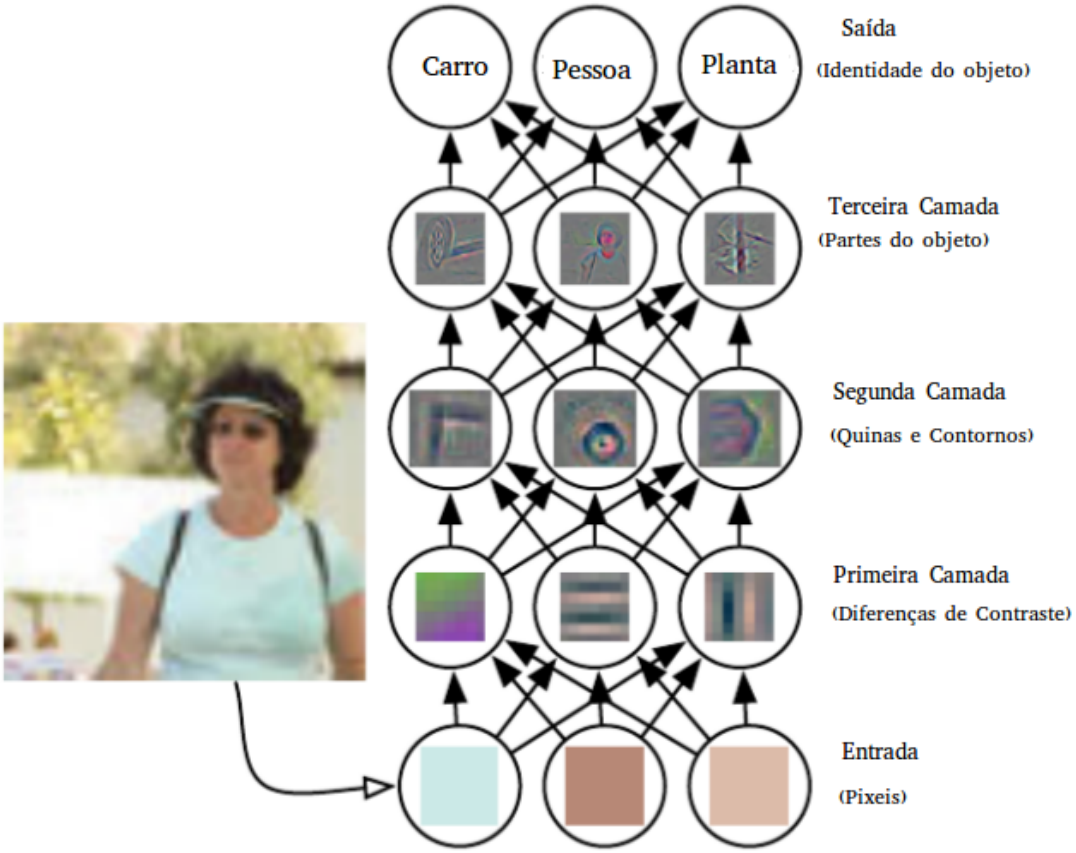

Figura 8 – Imagem traduzida do livro *Deep Learning* (GOODFELLOW; BENGIO; COURVILLE, 2016)

2.2 Modelo de Redes Neurais *Feedforward*

A maioria dos recentes desenvolvimentos em Aprendizado de Máquina se deram devido ao sucesso do modelo de redes neurais artificiais (RNAs). Esse modelo pode ser visto como uma extensão bem direta do de regressão linear. Para chegar nas RNAs, podemos partir do seguinte modelo, em notação de matriz (por simplicidade de notação, vamos omitir o vetor de erro $\boldsymbol{\epsilon}$):

$$\boldsymbol{X}\boldsymbol{w} = \boldsymbol{y}$$

$$\begin{bmatrix} 1 & x_{11} & ... & x_{1d} \\ 1 & x_{21} & ... & x_{2d} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & ... & x_{nd} \end{bmatrix} \times \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}$$

Podemos dizer que o modelo de regressão linear acima é uma rede neural com uma camada com um único neurônio, o vetor $\boldsymbol{w}$. Podemos adicionar mais uma camada de neurônios da seguinte forma:

$$(\boldsymbol{X}\boldsymbol{W_1})\boldsymbol{w} = \boldsymbol{y}$$

$$\begin{bmatrix} 1 & x_{11} & ... & x_{1d} \\ 1 & x_{21} & ... & x_{2d} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & ... & x_{nd} \end{bmatrix} \times \begin{bmatrix} w_{01} & w_{01} & ... & w_{0m} \\ w_{11} & w_{11} & ... & w_{1m} \\ \vdots & \vdots & \vdots & \vdots \\ w_{d1} & w_{d1} & ... & w_{dm} \end{bmatrix} \times \begin{bmatrix} w_{01} \\ w_{11} \\ \vdots \\ w_{d1} \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}$$

Com isso temos um aninhamento de multiplicações de matrizes. A multiplicação da matriz de dados por $\boldsymbol{W}$ é chamada de **primeira camada oculta** da rede. Essa operação é análoga à realizar d regressões lineares, sendo d o número de colunas em $\boldsymbol{W}$. A rede neural representada acima não é muito interessante. Na verdade, as operações lineares acima podem ser simplificadas para $\boldsymbol{X}\boldsymbol{w} = \boldsymbol{y}$, isto é, uma regressão linear. Para dar mais capacidade ao modelo, precisamos adicionar uma função não linear após cada multiplicação de matriz, a não ser a última. Essa função não linear ϕ recebe o nome de **ativação da camada oculta**.

$$\phi(\boldsymbol{X}\boldsymbol{W_1})\boldsymbol{w} = \boldsymbol{y}$$

No momento desta escrita, a não linearidade mais comum nas redes neurais é a unidade linear retificada, ou ReLU, que é dada por $\phi(z) = \max\{0, z\}$. Redes neurais como a acima tem capacidade teórica de representar qualquer função, contanto que tenha números suficientes de neurônios (colunas em $\boldsymbol{W}$). Assim, para aumentar a capacidade representativa da RNA, podemos tornar sua camada mais larga. Uma outra alternativa é manter a largura das camadas e tornar a rede mais profunda, isto é, adicionar mais camadas de neurônios.

$$\phi(\phi(\boldsymbol{X}\boldsymbol{W_1})\boldsymbol{W_2})\boldsymbol{w} = \boldsymbol{y}$$

Quando temos mais do que uma camada na rede neural, entramos no campo de *Deep Learning*. Aparte a intuição de que profundidade permite aprender abstrações

hierárquica, Montúfar et al. (2014) mostraram que a capacidade representativa da rede neural cresce exponencialmente com o número de camadas ocultas, se for utilizada uma não linearidade por partes como a ReLU. Isso porque esse tipo de ativação particiona o espaço dos dados de forma recursiva.

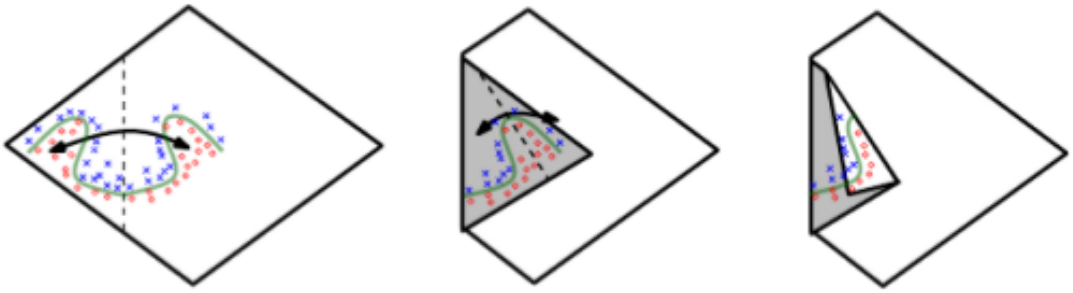

Figura 9 – Imagem de Montúfar et al. (2014),retirada do livro *Deep Learning* (GOODFELLOW; BENGIO; COURVILLE, 2016)

Esse dobramento do espaço acontece até que, no novo espaço, os dados podem ser representados por uma função linear. Note como as operações em matrizes no interior da rede porem ser representadas por $\boldsymbol{X^*} = \phi(\phi(\boldsymbol{X}\boldsymbol{W_1})\boldsymbol{W_2})$, em que $\boldsymbol{X}^*$ corresponde à forma dos dados transformada pela rede neural em algo representativo do problema que se quer modelar. Uma vez que tenhamos aprendido essa representação, a rede neural é, novamente, uma regressão linear:

$$\boldsymbol{X^*w} = \boldsymbol{y}$$

Se por um lado a profundidade da rede aumenta sua capacidade representativa, ela também torna o treinamento de redes neurais mais complicado. Diferentemente de uma modelo linear, a superfície de custo de redes neurais é não convexa e não há garantias de convergência no treinamento. Isso significa que a otimização desses modelos não pode ser feita de forma analítica, mas requer métodos iterativos de descida no custo, geralmente baseados em estimativas do gradiente. O resto deste capítulo será dedicado a mostrar como treinar redes neurais de maneira eficiente. Mas antes disso, por razões de completude, mostraremos uma alternativa à representação algébrica das redes neurais artificiais.

2.2.1 Representação Gráfica

Em muitos casos, redes neurais artificiais são representadas na forma de um **grafo computacional**. Está também é a representação que baliza a maioria dos softwares especializados em *Deep Learning*. Dessa forma, podemos abstrair parte das operações que estão sendo computadas na rede e focar mais no fluxo de dados que nela ocorre. Em alguns contextos, isso será mais conveniente do que escrever a rede neural em termos de operações algébricas. Considere a seguinte rede neural:

$$y = \phi(\boldsymbol{X}\boldsymbol{W})\boldsymbol{w}$$

$$\phi\left(\begin{bmatrix} x_{11} & x_{12} \\ \vdots & \vdots \\ x_{n1} & x_{n2} \end{bmatrix} \times \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix}\right) \times \begin{bmatrix} w_{11} \\ w_{21} \end{bmatrix} = y$$

A forma gráfica equivalente pode ser conferida na imagem 10. Cada seta está associada a um peso w_{ij}. Os nós com x_i representam a camada de entrada da rede, onde estão as variáveis dependentes. Os nós com h_i representam a camada oculta da rede, que se obtém multiplicando a primeira camada pelos respectivos pesos das conexões.

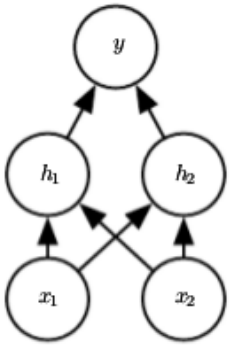

Figura 10 – Representação grafica de uma rede neural artificial (GOODFELLOW; BENGIO; COURVILLE, 2016)

2.3 Treinamento com Gradiente Descendente Modular

Redes neurais artificiais podem ser treinadas com maximização da verossimilhança. Para y contínuo, isto é, para problemas de regressão, a maximização da verossimilhança é equivalente à minimização do erro quadrático médio da amostra, se assumirmos normalidade nos resíduos

$$\mathcal{L}(\hat{\boldsymbol{W}}) = \sum_{n=1}^{N} (\boldsymbol{y} - \boldsymbol{y})^T (\boldsymbol{y} - \hat{\boldsymbol{y}})$$

onde $\hat{\boldsymbol{y}}$ é a previsão produzida pelo modelo. O treinamento então consiste em atualizar iterativamente os parâmetros $\boldsymbol{W}$ da rede, na direção de descida mais íngreme da superfície de custo. A seguinte regra de atualização dos parâmetros implementa esse procedimento:

$$\hat{\boldsymbol{W}} := \hat{\boldsymbol{W}} - \alpha \nabla(\mathcal{L})$$

isto é, atualizamos os parâmetros na direção oposta ao gradiente da função custo relativa a eles. α é um fator multiplicador do gradiente, que chamamos de taxa de aprendizagem. É um hiper-parâmetro do modelo que define um *trade-off* entre precisão e rapidez do treinamento. Quanto maior a taxa de aprendizado, maiores os passos na descida da superfície de custo, o que acelera o treinamento. Por outro lado, se a taxa de aprendizado for muito grande, o treinamento pode pular o ponto de mínimo; se α for grande de mais, o treinamento diverge. Conforme a taxa de aprendizado diminui, os passos na superfície de custo ficam menores e a busca pelo mínimo, mais refinada. Como consequência da maior precisão, o tempo de treinamento aumenta.

Para redes neurais mais simples, como as vistas até aqui, chegar na fórmula do gradiente do custo com relação aos parâmetros pode ser feito facilmente aplicando a regra da cadeia do cálculo. Por exemplo, para a rede neural com uma camada oculta a fórmula seria

$$\frac{\partial \mathcal{L}(\hat{\boldsymbol{W}})}{\partial \hat{W}} = \frac{\partial \mathcal{L}(\hat{\boldsymbol{W}})}{\partial \hat{\boldsymbol{y}}} * \frac{\partial \hat{\boldsymbol{y}}}{\partial \phi(\boldsymbol{X}\boldsymbol{W})} * \frac{\partial \phi(\boldsymbol{X}\hat{\boldsymbol{W}})}{\partial \hat{\boldsymbol{W}}}$$

Essas expansões por regra da cadeia rapidamente se tornam complicadas, além de serem particulares para cada arquitetura que se deseja construir. Felizmente é possível derivar um algoritmo recursivo que generaliza para qualquer arquitetura a computação das derivadas relevantes.

Seja z_i os dados que entram na camada i de uma rede neural, z_{i+1} os dados que saem dessa mesma camada, tal que o processamento nessa camada é dado por

$$z_{i+1} = f_i(z_i)$$

em que f_i é uma função diferenciável qualquer. Seja δ_i a derivada parcial da função custo $\mathcal{L}$ com respeito à entrada z_i da camada. O último δ é sempre 1, já que é a derivada da camada de custo com respeito à ela mesma. Os outros δ_i são então definidos de maneira recursiva.

$$\delta_i = \frac{\partial \mathcal{L}}{\partial z_i} = \frac{\partial \mathcal{L}}{\partial z_{i+1}} * \frac{\partial z_{i+1}}{\partial z_i} = \delta_{i+1} \frac{\partial z_{i+1}}{\partial z_i}$$

Em posse dos δ, podemos facilmente achar a derivada parcial do custo com respeito à qualquer parâmetro da rede neural.

$$\frac{\partial \mathcal{L}}{\partial w_i} = \frac{\partial \mathcal{L}}{\partial z_{i+1}} * \frac{\partial z_{i+1}}{\partial w_i} = \delta_{i+1} \frac{\partial z_{i+1}}{\partial w_i}$$

Note que os termos não recursivos são facilmente calculáveis a partir de f_i.

$$\frac{\partial f_i(z_i)}{\partial w_i} = \frac{\partial z_{i+1}}{\partial w_i}$$

$$\frac{\partial f_i(z_i)}{\partial z_i} = \frac{\partial z_{i+1}}{\partial z_i}$$

Na prática, cada camada é definida em termos de um processamento $f_i(z_i) = z_{i+1}$ onde a informação flui para frente na rede neural - o chamado *forward pass* - e um mecanismo $g_i(\delta_{i+1}) = \delta_i$ que propaga δs para traz - o chamado *backward pass*. Caso a camada tenha parâmetros, ela também terá um mecanismo que computa a derivada parcial do custo com respeito a esses parâmetros, usando a informação de δ_i.

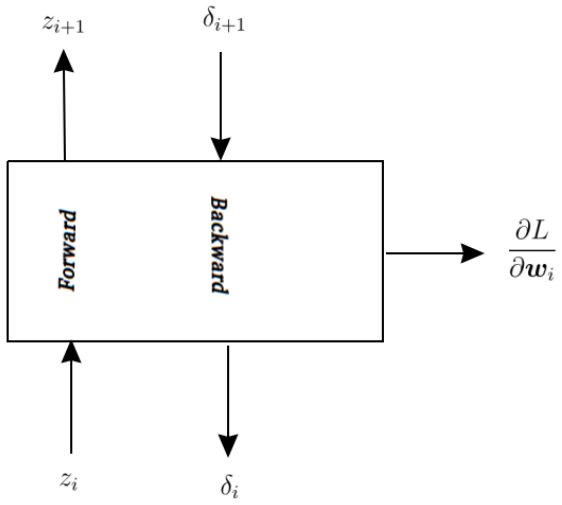

Figura 11 – Abstração de uma camada de rede neural

Isso significa que, se o software de *Deep Learning* implementar cada camada com esses dois ou três mecanismos, como normalmente é o caso, nós praticantes só precisamos nos preocupar em empilhar essas camadas de forma inteligente. Todo o cálculo de gradientes e propagação de δ acontece no fundo do programa e automaticamente. Além disso, definir uma nova camada é extremamente simples, já que basta implementá-la com os dois ou três mecanismos descritos acima para que ela seja completamente integrável ao software já existente. Essa modularidade faz com que novas descobertas no campo sobre camadas de redes neurais possam ser rapidamente compartilhadas e exploradas. Por fim, mas não menos importante, note como uma camada é simplesmente uma abstração com mecanismos de processar dados para frente, propagar derivadas para trás e, se os tiver, computar derivadas com respeito aos seus parâmetros. Isso significa que qualquer rede neural feita com essas camadas é, ela própria, uma nova camada, já que possui os três mecanismos acima definidos. Isso significa que podemos facilmente incorporar redes neurais já feitas

como camadas de novas redes neurais, o torna o conhecimento nessa área extremamente cumulativo.

2.3.1 Gradiente Descendente Estocástico

Otimização via gradiente descendente envolve, para cada iteração, um somatório por todos os exemplos da amostra. Isso rapidamente se torna proibitivamente ineficiente. Para lidar com isso, usamos apenas uma pequena amostra, um mini-lote de dados, para computar aproximações do gradiente a cada iteração.

Como consequência, os passos em direção ao mínimo da função custo passam a ser menos precisos e nem sempre apontarão na direção certa. No entanto, cada iteração é muito mais rápida do que se tivéssemos que considerar todos os dados para computar o gradiente. Assim, podemos realizar muito mais passos, que, mesmo sendo imprecisos, na média levam ao mínimo e de maneira muito mais rápida.

O preço a se pagar por isso é que o tamanho do mini-lote se torna outro hiperparâmetro que precisamos ajustar. Caso os dados sejam amplamente redundantes, mini-lotes com pouquíssimos dados, como 16 ou 32, bastam. Se houver mais heterogeneidade nos dados será preciso de mini-lotes maiores. É importante termos em mente que o tamanho do mini-lote é caracterizado por um *trade-off* entre realizar mais iterações em menos tempo ou realizar iterações mais precisas mas mais demoradas.

2.3.2 Adam

Adaptive Moment Estimation (Adam) (KINGMA; BA, 2014) é uma forma de acelerar a convergência da otimização por gradiente descendente. Adam armazena uma média exponencialmente decrescente dos últimos gradientes ao quadrado, $\boldsymbol{v}_t$, e uma média exponencialmente decrescente dos último gradientes, $\boldsymbol{m}_t$.

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t \\ v_t &= \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \end{aligned} \tag{2.1}$$

Intuitivamente, $\boldsymbol{m}_t$ é uma estimativa do primeiro momento dos gradientes (média) e $\boldsymbol{v}_t$ é uma estimativa do segundo momento (variância) (RUDER, 2016). A regra de atualização dos parâmetros então passa a ser

$$\hat{\boldsymbol{W}}_{t+1} = \hat{\boldsymbol{W}}_t - \alpha \frac{\boldsymbol{m}_t}{\sqrt{\boldsymbol{v}_t} + \boldsymbol{\epsilon}}$$

em que ϵ é uma pequena constante para evitar divisão por zero. Adam pode ser entendido como uma versão de gradiente descendente estocástico com taxa de aprendizado

adaptativa e momento nas descidas do gradiente. O termo $\boldsymbol{m}_t$ acelera passos em direções onde a descida é estável, enquanto que o a divisão por $\boldsymbol{v}_t$ desacelera a descida diante de fortes oscilações. Na prática, Adam acha pontos de custo baixo muito mais rapidamente do que simples gradiente descendente estocástico, além de ser menos vulnerável a ficar preso em pontos críticos ruins (mínimos locais e pontos de sela).

Parte II

Análise Empírica

3 O Problema das Passagens Aéreas

Para exemplificar o uso de aprendizado de máquina na economia, vamos aplicar um modelo de rede neural para previsão de preço de passagens aéreas. Esse problema é especialmente interessante por dispor de dados em abundância. Além disso, a evolução do preço de uma mesma passagem aéreas no tempo é extremamente complicado e definitivamente não linear.

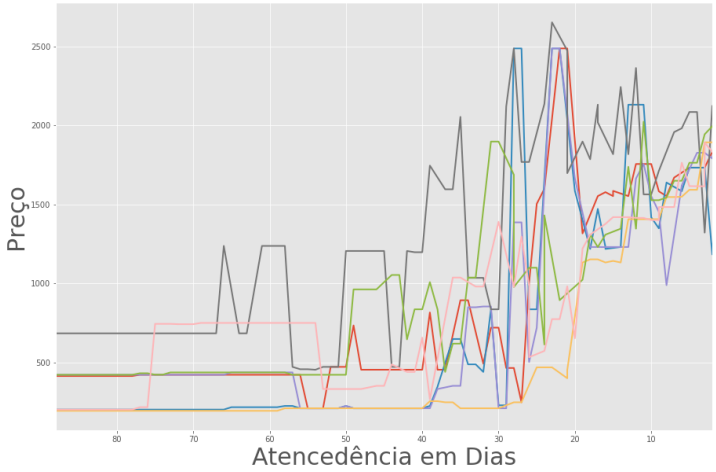

Figura 12 – Evolução de preços de algumas passagens conforme a distância entre a observação do preço e a data partida.

Uma hipótese simplista seria afirmar que os preços de uma passagens tendem a ficar estáveis e fixos por um bom tempo desde o lançamento da passagem no mercado, que acontece com um ano de antecedência da partida. Conforme a data de partida se aproxima, os preços começarão a flutuar de acordo com a demanda e escassez. Assim, a tendência é que o preço da passagem sempre suba, já que, conforme os assentos vão sendo vendidos, aumenta-se a escassez, o que aumenta o preço. Por outro lado, como o custo variável marginal de passageiros é muito baixo, caso o agente perceba uma probabilidade dos assentos não serem todos ocupados, pode haver uma brusca queda no preço da passagem.

Há mais complexidade do que isso. Por exemplo, a passagem aérea pode ter baixa elasticidade-preço para alguns clientes, por exemplo executivos que viajam pela empresa, ou extremamente alta, como para pessoas que esperam descontos de última hora para viajar. No entanto, em vez de gastar muito tempo entrando nesses detalhes, vamos simplesmente alimentar um modelo de aprendizado de máquina com dados de passagens aéreas e deixar que ele explora essas complexidades da forma mais conveniente para prever os preços.

3.1 Os Dados

Os dados consistem em eventos de passagens aéreas coletados quase diariamente durante o período de um ano (de agosto de 2016 a julho de 2017).

3.1.1 Metodologia de Coleta

Para coletar os eventos de passagens aéreas, utilizamos o API do Skyscanner (SKYSCANNER, 2016). Ele aceita como parâmetros dos pedidos o local de origem e destino da passagem que se deseja pesquisar, assim como a data de ida e de volta da viagem. O retorno do pedido é um arquivo JSON, contendo informações da passagem pesquisada naquele momento. Para mais informações sobre o API e sobre a estrutura do seu retorno, confira sua documentação.

No JSON retornado adicionamos um novo par chave-valor contendo informações sobre o momento da coleta. Posteriormente, esse JSON foi convertido em uma estrutura tabular separadas por vírgulas, que é mais conveniente para alimentar modelos de aprendizado de máquina. Devido a problemas técnicos, a coleta falhava parcialmente na maioria dos dias. Assim, pode haver lacunas de mais de um dia entre duas observações da mesma passagem aérea.

3.1.2 Variáveis

Apenas algumas informações do JSON foram mantidas na base de dados utilizada pelo modelo. As variáveis dessa base final incluem:

Preço da passagem, observado no momento da coleta do evento. Essa será a variável dependente do nosso modelo, isto é, aquela que tentaremos prever.

Nome do agente que vende a passagem aérea. Essa informação está em formato de texto. Alguns exemplos de agentes são *LATAM Airlines*, *Expedia* e *Bravofly*. Como o modelo de rede neural não aceita texto como entrada, codificamos essa variável para um inteiro único, identificador da passagem aérea. Repare que se estivéssemos usando um modelo de regressão linear, cada agente deveria ser transformado em uma coluna de variável *dummie*. Felizmente, isso não é essencial nos modelos de redes neurais, de forma que podemos assim poupar recursos computacionais e trabalhar com uma única coluna para representar esses dados categóricos.

Duração da viagem em minutos, isto é, o tempo entre a decolagem e o pouso do avião.

Nome da origem e **nome do destino** para onde é a passagem. Esta variável também é de texto e foi convertida para categorias denotadas por inteiros da mesma forma que o nome do agente.

Momento da coleta do evento da passagem. Essa variável está no formato de texto denotando período de tempo. Para ser mais específico, o formato é *yyyy-mm-dd hh:mm:ss*. Ela não será utilizada no modelo, apenas para a separação entre base de dados de treino e de teste, afinal, queremos avaliar a performance do modelo em dados futuros.

Tempo viajando em dias, isto é, o tempo entre a data da ida e da volta.

Dia da semana em que o evento da passagem foi observado.

Dia em que a o evento da passagem foi observado, representado como um inteiro, contado a partir do dia 01-01-2016. Essa variável captura tendências.

Hora em que o evento da passagem foi observado. Essa variável foi adicionada para capturar possíveis mudanças intra diárias no preço das passagens aéreas.

Dia do mês da viagem de ida.

Hora da partida, isto é, um inteiro de 0 a 23 representado a hora do dia em que sai o voo de ida.

Diferença entre a data em que o evento da passagem foi observado e o dia da viagem. Essa variável foi incluída pois acreditamos que há uma antecedência ótima para comprar a passagem, que pode ser estimada vendo a previsão do preço para a mesma passagem, mudando apenas esta variável.

3.2 Os Modelos

O primeiro modelo que consideramos é uma regressão linear simples, aplicada nas variáveis acima para prever o preço da passagem aérea. Mais formalmente, seja $\boldsymbol{x}$ um vetor [$nome\_do\_agente$, $duracao\_do\_voo$, $nome\_da\_origem$, $nome\_do\_destino$, $duracao\_da\_viagem$, $dia\_da\_semana\_da\_observacao$, $dia\_da\_observacao$, $hora\_da\_observacao$, $dia\_da\_partida$, $diferenca\_entre\_partida\_e\_observacao$, $hora\_da\_saida$], o modelo é dado por

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{w} + b$$

em que $\boldsymbol{X}$ é uma tabela de $\boldsymbol{x}$ empilhados, $\hat{\boldsymbol{y}}$ é o vetor de preços estimados das passagens aéreas e $\boldsymbol{w}$ e b são os parâmetros do modelo.

O segundo modelo considerado foi uma rede neural profunda densamente conectada. Formalmente, adotando as mesmas nomenclaturas acima definidas, o segundo modelo é dado por

$$\hat{\boldsymbol{y}} = \phi(\phi(\boldsymbol{X}\boldsymbol{W_1})\boldsymbol{W_2})\boldsymbol{w} + \boldsymbol{b}$$

em que ϕ é a função $ReLU$, dada por $f(z) = \max\{0, z\}$, $\boldsymbol{W_1}$ são os parâmetros da primeira camada oculta, $\boldsymbol{W_2}$ são os parâmetros da segunda camada oculta e $\boldsymbol{w}$ são os parâmetros da camada de saída da rede neural. Ambos $\boldsymbol{W_1}$ e $\boldsymbol{W_2}$ são matrizes com 32 colunas, indicando que cada camada oculta da rede neural tem 32 neurônios.

3.3 Métricas de Avaliação

A primeira métrica sob a qual os modelos acima serão avaliados é o coeficiente de determinação ou R^2, dado por

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=0}^{n-1}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{n-1}(y_i - \bar{y})^2}$$

Como definido acima, o R^2 varia de 1 a menos infinito, sendo 0 quando se prevê a média da variável dependente. O R^2 mede quanto da variação na variável dependente é explicada pelo modelo.

A segunda métrica utilizada será o Erro Absoluto Média, MAE, dado por

$$\text{MAE}(y, \hat{y}) = \frac{1}{n}\sum_{i=0}^{n-1}|y_i - \hat{y}_i|$$

3.4 Resultados

As métricas acima serão computadas em dois sets de dados diferentes. O primeiro consistirá nos dados coletados até junho de 2017 e será também usado para treinar os modelo. O segundo set de dados conterá observações a partir de junho de 2017 e será usado para estimar a performance de generalização dos modelos. O set de treino contará como 4.076.562 observações, enquanto que o set de teste será de 1.702.691 observações.

No experimento realizado com os dois sets de dados acima, o modelo linear teve um R^2 de $0,372$ no set de treino e de $0,493$ no set de teste. O MAE no set de treino foi de R\$ $322,47$ e de R\$ $310,81$ no set de teste. O fato de que o modelo tem performance melhor no set de teste significa que os dados de teste são mais fáceis de prever do que o de treino. Além disso, isso indica que o modelo de regressão linear sofre com sub-ajustamento.

A rede neural com duas camadas ocultas de 32 neurônios cada foi treinada durante 10 épocas, com um mini-lote de 1024 e taxa de aprendizado de $0,01$. Ela obteve teve um R^2 de $0,553$ no set de treino e de $0,548$ no set de teste. O MAE no set de treino foi de R\$ $249,13$ e de R\$ 272.32 no set de teste. O modelo parece estar bem ajustado, já que a performance nos sets de treino e teste são bem similares. Além disso, em termo das

métricas aqui monitorada, fica clara a vantagem do modelo de *Deep Learning* sobre o de regressão linear.

4 Considerações Finais

Mostramos aqui como é possível partir de modelos tradicionais da econometria e chegar em modelos profundos de aprendizado de máquina de maneira bastante natural. Por meio de um exemplo ilustrativo, vimos algumas semelhanças e diferenças entre uma abordagem econométrica e uma de aprendizado de máquina. Também abordamos os principais conceitos do paradigma de aprendizado de máquina supervisionado, como capacidade e generalização e validação cruzada. Finalmente, mostramos como *Deep Learning* pode resolver o problema da forte dependência em representações informativas. Para tratar de tudo isso em algumas páginas, foi imprescindível fazer algumas simplificações severas e imprecisas. Cabe agora ressalta-las e apontar algumas fontes onde é possível encontrar informações mais aprofundadas que não foram possível incluir na breve introdução ao aprendizado de máquina que foi este trabalho.

Em primeiro lugar, devido a semelhança entre modelos de redes neurais e a clássica regressão linear utilizada em econometria, deixamos um pouco de lado alguns modelos extremamente importantes, largamente utilizados na industria e que, em muitos casos são preferidos à modelos de *Deep Learning*. Esta não é uma regra geral, mas é comum observar Máquinas de Suporte Vetorial e Processos Gaussianos obterem melhor performance do que redes reunais em regime de poucos dados (menos do que 10000 exemplos). Na indústria, métodos de Árvore de Decisão são amplamente explorados, principalmente na sua forma de mistura de especialista (*ensemble*), sob o nome de Florestas Aleatórias (*Bagging*). Além disso, no momento desta escrita, combinações de árvores na forma de *Boosting* de Gradientes são os modelos mais populares em competições online de aprendizado de máquina. Assim, recomendamos fortemente que o leitor também se familiarize com essas outras classes modelos. Algumas boas fontes para isso são a biblioteca de computação Scikit-learn (PEDREGOSA et al., 2011) - devido a sua excelente documentação -, o livro de Alpaydin (2014) e o de Géron (2017).

Em segundo lugar, nos restringimos ao campo de aprendizado de máquina supervisionado, onde estávamos mais interessados em fazer previsões de uma variável dependente a partir de variáveis independentes. Além disso, existe o campo de aprendizado de máquina não supervisionado, extremamente sólido e em rápida expansão. Nesse caso, o que se busca não é mapear uma função de x em y, mas encontrar uma representação mais conveniente para os dados. Caem nesse escopo algoritmos de redução de dimensionalidade, com Análise de Componentes Principais (PCA), Análise de Componentes Independentes (ICA), mapas auto-organizados, *word2vec* e alguns tipos de Auto-codificadores. Uma outra classe de modelos não supervisionados que tem se destacados nos últimos anos são os modelos geradores, como Mistura de Gaussianas, Auto-codificadores Variacionais, Redes

Adversárias e algumas Redes Neurais Recorrentes. Algoritmos de Clusterização, como *k-Means Clustering* e Mistura de Gaussianas, e algoritmos de extração de regras, como *Apriori* e *Eclat*, também caem no escopo não supervisionado. Além disso, a intersecção entre aprendizado supervisionado e não supervisionado - também chamada de aprendizado semi-supervisionado - tem se mostrado extremamente promissora, principalmente em tarefas como detecção de anomalias e processamento de linguagem natural.

Finalmente, por termos tratado de aprendizado de máquina numa perspectiva introdutória, grande parte do trabalho que integra economia e aprendizado de máquina, por estar na fronteira do conhecimento, ficou de fora desta monografia. Assim, vale mencionar aqui os recentes esforço em incorporar causalidade em modelos de aprendizado de máquina. Usando modelos de Florestas Aleatórias, Wager e Athey (2015) mostraram que é possível alterar o critério de partição de cada árvore para maximizar a variância do efeito de tratamento individual em cada folha, podendo assim estimar efeito de tratamento de maneira heterogênea. Na mesma linha de pesquisa, Johansson, Shalit e Sontag (2016) desenvolveram um modelo de aprendizado de representações para inferência contra-factual. Outro autor que ressalta a importância de incorporar causalidade nos modelos de aprendizado de máquina é Max Welling, onde destaca-se seu recente trabalho sobre interoperabilidade de modelos de *Deep Learning* (ZINTGRAF et al., 2017).

Reconhecemos que esta monografia tem apenas o papel de apresentar o campo de aprendizado de máquina aos praticantes ou estudantes de econometria. Ainda assim, esperamos que o que foi aqui desenvolvido, junto com as referencias acima colocadas, instigue uma integração mais frutífera entre os objetos de pesquisa da economia e o poderoso ferramental estatístico da inteligência artificial.

5 Conclusão

Modelos de aprendizado de máquina tem a flexibilidade e o poder de tornar as previsões econômicas mais precisas do que nunca antes. Neste trabalho mostramos como os modelos de *Deep Learning* são extensões simples dos modelos lineares tradicionalmente utilizados pelos economistas. Assim, argumentamos em favor da maior integração entre as ciências de Econometria e de Aprendizagem de Máquina, principalmente em aplicações de previsão.

Além disso, ressaltamos que muito ainda há de ser feito para aliar a capacidade preditiva dos modelos de aprendizado de máquina com a estatística causal desejada pelos economistas. Também mostramos que com a implementação modular do algoritmo de *backpropagation* é possível acessar facilmente as derivadas parciais da previsão $\hat{y}$ com relação às variáveis independentes $\boldsymbol{x}$. Realizar inferência nessas derivadas, contudo, é um trabalho árduo e ainda não explorado.

Por fim, realizamos uma exemplificação simples da utilização de modelos de *Deep Learning* em dados reais de passagens aéreas. Mostramos como há uma clara vantagem preditiva ao se usar um modelo de aprendizado de máquina com mais capacidade do que uma simples regressão linear.

Referências

ALPAYDIN, E. *Introduction to Machine Learning*. [S.l.]: The MIT Press, 2014. Citado na página 47.

ATHEY, S. *How will machine learning impact economics?* 2016. Disponível em: <https://www.quora.com/How-will-machine-learning-impact-economics>. Citado na página 13.

ATHEY, S.; IMBENS, G. The State of Applied Econometrics - Causality and Policy Evaluation. *ArXiv e-prints*, jul. 2016. Citado na página 14.

CAJUEIRO, D. O. *Qual é a diferença entre estatística e aprendizagem de máquinas?* 2015. Disponível em: <http://prorum.com/index.php/30/qual-diferenca-entre-estatistica-aprendizagem-de-maquinas>. Citado na página 15.

ECONOMIST, T. Million-dollar babies. *The Economist Apr. 2nd*, 2016. Disponível em: <https://www.economist.com/news/business/21695908-silicon-valley-fights-talent-universities-struggle-hold-their>. Citado na página 9.

GAL, Y. *Uncertainty in Deep Learning*. Tese (Doutorado) — University of Cambridge, 2016. Citado na página 20.

GAL, Y.; GHAHRAMANI, Z. On modern deep learning and variational inference. In: *Advances in Approximate Bayesian Inference workshop, NIPS*. [S.l.: s.n.], 2015. Citado na página 20.

GÉRON, A. *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. [S.l.]: O'Reilly Media, 2017. Citado 2 vezes nas páginas 30 e 47.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>. Citado 10 vezes nas páginas 5, 20, 21, 22, 26, 29, 30, 31, 33 e 34.

HINTON, G. E.; OSINDERO, S.; TEH, Y.-W. A fast learning algorithm for deep belief nets. *Neural Comput.*, MIT Press, Cambridge, MA, USA, v. 18, n. 7, p. 1527–1554, jul. 2006. ISSN 0899-7667. Disponível em: <http://dx.doi.org/10.1162/neco.2006.18.7.1527>. Citado na página 29.

JAMES, G. et al. *An Introduction to Statistical Learning: With Applications in R*. [S.l.]: Springer Publishing Company, Incorporated, 2014. ISBN 1461471370, 9781461471370. Citado na página 15.

JOHANSSON, F. D.; SHALIT, U.; SONTAG, D. Learning Representations for Counterfactual Inference. *ArXiv e-prints*, maio 2016. Citado na página 48.

KAHNEMAN, D. *Thinking, fast and slow*. New York: Farrar, Straus and Giroux, 2011. ISBN 9780374275631 0374275637. Disponível em: <https://www.amazon.de/

Thinking-Fast-Slow-Daniel-Kahneman/dp/0374275637/ref=wl_it_dp_o_pdT1_nS_nC?ie=UTF8&colid=151193SNGKJT9&coliid=I3OCESLZCVDFL7>. Citado na página 13.

KINGMA, D. P.; BA, J. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. Disponível em: <http://arxiv.org/abs/1412.6980>. Citado na página 37.

KINGMA, D. P.; WELLING, M. Auto-Encoding Variational Bayes. *ArXiv e-prints*, dez. 2013. Citado na página 20.

KIRSNER, S. What makes this the hottest class at mit? *Boston Globe*, 2017. Disponível em: <https://www.bostonglobe.com/business/2017/04/06/what-makes-this-hottest-class-mit/wcAVJfK8U9D64hyVSTTbqI/story.html>. Citado na página 9.

MINSKY, M.; PAPERT, S. A. *Perceptrons*. [S.l.]: MIT Press, 1969. Citado na página 29.

MITCHELL, T. M. *Machine Learning*. 1. ed. New York, NY, USA: McGraw-Hill, Inc., 1997. ISBN 0070428077, 9780070428072. Citado na página 14.

MONTÚFAR, G. et al. On the Number of Linear Regions of Deep Neural Networks. *ArXiv e-prints*, fev. 2014. Citado 2 vezes nas páginas 5 e 33.

MORDVINTSEV, A.; OLAH, C.; TYKA, M. *Inceptionism: Going Deeper into Neural Networks*. 2015. Disponível em: <https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html>. Citado na página 30.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Citado na página 47.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 1958. Citado na página 29.

RUDER, S. *An overview of gradient descent optimization algorithms*. 2016. Disponível em: <http://ruder.io/optimizing-gradient-descent/index.html#adam>. Citado na página 37.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning internal representations by error propagation. In: RUMELHART, D. E.; MCCLELLAND, J. L.; GROUP, C. P. R. (Ed.). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1*. Cambridge, MA, USA: MIT Press, 1986. cap. Learning Internal Representations by Error Propagation, p. 318–362. ISBN 0-262-68053-X. Disponível em: <http://dl.acm.org/citation.cfm?id=104279.104293>. Citado na página 29.

SKYSCANNER. *Travel APIs*. 2016. Disponível em: <https://partners.skyscanner.net/affiliates/travel-apis/>. Citado na página 42.

VALIANT, L. G. A theory of the learnable. *Commun. ACM*, ACM, New York, NY, USA, v. 27, n. 11, p. 1134–1142, nov. 1984. ISSN 0001-0782. Disponível em: <http://doi.acm.org/10.1145/1968.1972>. Citado na página 21.

VEJA. 2017. Citado na página 9.

WAGER, S.; ATHEY, S. Estimation and Inference of Heterogeneous Treatment Effects using Random Forests. *ArXiv e-prints*, out. 2015. Citado na página 48.

WOOLDRIDGE, J. M. *Introductory Econometrics : A Modern Approach*. [S.l.]: Cengage South-Western, 2012. Citado 3 vezes nas páginas 14, 18 e 19.

ZINTGRAF, L. M. et al. Visualizing deep neural network decisions: Prediction difference analysis. *Por publicar*, 2017. Citado na página 48.