Universidade de São Paulo
Instituto de Matemática e Estatística
Bacharelado em Matemática Aplicada

Redes Neurais aplicadas

Eduardo Galvani Massino

Monografia Final

map 2010 — Trabalho de Formatura

Orientador: Prof. Dr. José Coelho de Pina Junior

São Paulo
Dezembro de 2020

Redes Neurais aplicadas

Eduardo Galvani Massino

Esta é a versão original da monografia
elaborada pelo candidato Eduardo
Galvani Massino, tal como
submetida à Comissão Julgadora.

Esta seção é opcional e fica numa página separada;
ela pode ser usada para uma dedicatória ou epígrafe.

Agradecimentos

Do. Or do not. There is no try.

— Mestre Yoda

Texto texto. Texto opcional.

Resumo

Eduardo Galvani Massino. **Redes Neurais aplicadas**. Monografia (Bacharelado). Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2020.

Elemento obrigatório, constituído de uma sequência de frases concisas e objetivas, em forma de texto. Deve apresentar os objetivos, métodos empregados, resultados e conclusões. O resumo deve ser redigido em parágrafo único, conter no máximo 500 palavras e ser seguido dos termos representativos do conteúdo do trabalho (palavras-chave). Deve ser precedido da referência do documento. Texto texto. Texto texto.

Palavras-chave: redes-neurais. perceptron.

Abstract

Eduardo Galvani Massino. **Redes Neurais aplicadas**. Capstone Project Report (Bachelor). Institute of Mathematics and Statistics, University of São Paulo, São Paulo, 2020.

Elemento obrigatório, elaborado com as mesmas características do resumo em língua portuguesa. De acordo com o Regimento da Pós-Graduação da USP (Artigo 99), deve ser redigido em inglês para fins de divulgação. É uma boa ideia usar o sítio www.grammarly.com na preparação de textos em inglês. Text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text. Text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text.

Keywords: neural-nets. perceptron.

Lista de Abreviaturas

CFT	Transformada contínua de Fourier (*Continuous Fourier Transform*)
DFT	Transformada discreta de Fourier (*Discrete Fourier Transform*)
EIIP	Potencial de interação elétron-íon (*Electron-Ion Interaction Potentials*)
STFT	Transformada de Fourier de tempo reduzido (*Short-Time Fourier Transform*)
ABNT	Associação Brasileira de Normas Técnicas
URL	Localizador Uniforme de Recursos (*Uniform Resource Locator*)
IME	Instituto de Matemática e Estatística
USP	Universidade de São Paulo

Lista de Símbolos

ω	Frequência angular
ψ	Função de análise *wavelet*
Ψ	Transformada de Fourier de ψ

Lista de Figuras

Lista de Tabelas

Lista de Programas

Sumário

Capítulo 1

Introdução

Algoritmos de aprendizado de máquina vem sendo utilizados para resolver um enorme gama de problemas computacionais de resolução numérica. Existem atualmente aplicações em praticamente todas as áreas do conhecimento humano, da agricultura à indústria e ao entretenimento. Mas o que é **aprendizado de máquina**? Ou melhor dizendo, o que significa dizer que o computador, ou seja, a "máquina" está *aprendendo*?

Antes da definição propriamente dita, Aurélien Géron (Géron, 2019) nos dá uma ideia geral lembrando que uma das primeiras aplicações de sucesso de aprendizado de máquina foi o filtro de *spam*, criado na década de 90. Uma das fases de seu desenvolvimento foi aquela em que os usuários marcavam que certos e-mails eram *spams* e outros não eram. Hoje em dia, raramente temos que marcar ou desmarcar e-mails, pois a maioria dos filtros já "aprenderam" a fazer seu trabalho de forma muito eficiente.

Existem muitas definições do que seja aprendizado de máquina, de modo geral uma área da ciência da computação mas que, especificamente no contexto de ciência de dados, Joel Grus (Grus, 2016) define como a criação e o uso de modelos que são aprendidos a partir dos dados. O objetivo é usar dados existentes para desenvolver modelos que possamos usar para *prever* possíveis saídas para dados novos. Exemplos, além do filtro de *spams* podem ser: prever transações de crédito fraudulentas, prever a chance de um cliente clicar em uma propaganda ou então prever qual time de futebol irá vencer o Campeonato Brasileiro.

Uma *rede neural* é um exemplo de modelo preditivo de aprendizado de máquina. Tal modelo foi criado com inspiração no funcionamento do cérebro biológico, e que, apesar de terem sido as primeiras a serem criadas, conforme descrito por David Kopec (Kopec, 2019), vem ganhando nova importância na última década, graças ao avanço computacional, uma vez que exigem muito processamento, e também porque podem ser usadas para resolver problemas de aprendizagem dos mais variados tipos.

Atualmente existem vários tipos de redes neurais, porém este trabalho lida principalmente com aquele tipo que foi originalmente criado inspirado no cérebro, chamado de **perceptron**, e que portanto tenta imitar o comportamento dos neurônios e suas conexões, aprendendo padrões a partir de dados existentes e tentando prever o comportamento de dados novos a partir do padrão aprendido.

[Ao final a uma breve descrição do conteúdo do texto.]

Capítulo 2

Redes neurais multicamadas

Neste capítulo são apresentados alguns conceitos básicos de **aprendizado de máquina**, com foco nos algoritmos de redes neurais multicamadas, em especial o **perceptron**, cujo desenvolvimento foi inspirado nas redes neurais biológicas, ou seja, os neurônios e suas conexões no cérebro, conforme descrito por Kopec (Kopec, 2019).

Pode-se classificar as técnicas de aprendizado de várias formas, de acordo com alguma de suas características. Por exemplo, Géron (Géron, 2019) utiliza o grau de supervisão humana durante o seu funcionamento para classificá-los em aprendizado supervisionado ou não-supervisionado.

2.1 Aprendizado de máquina supervisionado

Um algoritmo de **aprendizado supervisionado** é usado quando conhecemos os rótulos dos dados que estamos utilizando para o treinamento, ou seja, temos a resposta correta da aprendizagem. Por exemplo, se estamos classificando fotos de animais, possuímos um conjunto de fotos em que já sabemos de antemão quais são de gatos, cachorros, etc.

O ato de rotular ou classificar os dados que usamos no aprendizado é o que designamos de supervisão humana. Uma vez *treinado*, o algoritmo recebe uma foto nova e então a classifica como sendo uma foto de um gato, ou cachorro, ou qualquer outra resposta daquelas que foram dadas como exemplos durante o treinamento.

Dentro do aprendizado supervisionado temos duas principais técnicas. A regressão é usada para prever valores, e a classificação é usada para prever os rótulos dos dados (que também são chamados de classes). Neste texto os termos “algoritmo” e “técnica” serão usados livremente como sinônimos, pois uma técnica de aprendizado de máquina, no contexto atual, é obviamente um algoritmo executado no computador.

Colocar exemplos...

2.2 Aprendizado não-supervisionado

Nesse tipo de aprendizado de máquina, não sabemos os rótulos dos dados que estamos lidando, assim o algoritmo poderá agrupar os dados de forma automática, por exemplo, se estiver sendo usado um algoritmo classificador.

Alguns métodos não-supervisionadas de aprendizado foram enumeradas por Géron (Géron, 2019). O **agrupamento** de dados similares, sendo essa similaridade podendo ser uma distância no espaço dos dados (inspiração geométrica), e utiliza-se algoritmos como k-Vizinhos, k-Means, k-Medians, etc. Exemplos de aplicações são agrupamento de produtos em supermercados, interesses comuns de clientes em sites de conteúdo digital, etc.

Outra técnida é a **detecção de anomalias**, cujo objetivo é ter uma descrição de como os dados considerados "normais" se parecem, e usa esse agrupamento para detectar se novos dados estariam "fora" desse padrão. Um exemplo é a detecção de fraudes.

Também pode-se citar sobre a técnica de **estimação de densidades**, que tem como objetivo a estimação da função densidade de probabilidade de um conjunto de dados gerados por algum processo aleatório.

Colocar exemplos...

2.3 Técnicas de Classificação

Sendo uma das duas técnicas principais do aprendizado supervisionado, problemas desse tipo buscam aprender com um conjunto de dados previamente rotulados, como "se parecem" os dados que pertencem às classes que queremos classificar, para que quando processarmos novos dados, o algoritmo usado possa identificar, o mais corretamente possível, as classes às quais pertecem esses dados, dentro do conjunto de classes que já definimos ao rotular os dados iniciais.

Existem vários tipos de algoritmos de classificação, dentre eles podemos mencionar: Máquina de Vetor Suporte, em inglês Support Vector Machine (SVM), Árvores de decisão, Florestas aleatórias (podendo ser entendidas como um conjunto de centenas de árvores de decisão aleatoriamente definidas) e as Redes neurais artificiais.

Colocar exemplos de aplicações...

Todos essas técnicas podem ser usadas para classificação linear ou não-linear, no sentido em que valores eles estão classificando, assim como na forma que está sendo feita essa classificação. Se imaginamos um espaço bidimensional, um algoritmo de classificação linear irá separar as classes de dados por retas, enquanto que um classificador não-linear poderá usar outra curva qualquer para a separação. Abstraindo o espaço bidimensional para os espaços multidimensionais dos dados que são comumente analisados, podemos pensar em hiperplanos (estruturas (n−1-dimensionais de espaços n-dimensionais) para o caso dos classificadores lineares, ou subespaços quaisquer para os não-lineares.

2.4 Redes Neurais

Uma rede neural artificial é um dentre vários métodos de classificação, ou seja, de aprendizado supervisionado. De acordo com Kopec (Kopec, 2019), ele é utilizado como um classificador não-linear, e por isso pode ser utilizado para prever tipos de dados genéricos, que podem ou não ser lineares.

Kopec (Kopec, 2019) também cita que este é um dos primeiros a serem desenvolvidos ainda na década de 1950, sendo sua criação motivada pelo funcionamento do cérebro, mas que, de tempos pra cá, vem recebendo muita importância graças à evolução computacional de hardware e software desde o início dos anos 2000.

Contextualizar mais as redes neurais aqui ...

2.5 Perceptron

A rede mais simples, ou pelo menos o primeiro algoritmo a ser desenvolvido, chama-se Perceptron. Como recentemente surgiu o Perceptron de muitas camadas, a esse primeiro dá-se o nome de Perceptron de única camada, que consiste de uma camada de neurônios de entrada, uma camada oculta de neurônios usados na otimização (mais detalhes abaixo), e uma camada de saída, que contém um neurônio para cada classe pré-definida que será usada na classificação, e cada um armazena a probabilidade deste elemento da amostra pertencer a esta classe.

Um exemplo ilustrativo está na Figura 2.1. Os neurônios são representados por círculos, cada coluna de neurônios representa uma camada, nesse caso, da esquerda para a direita temos a camada de entrada, a camada oculta e a camada de saída. As linhas representam as ligações entre os neurônios, sendo que cada neurônio de uma camada está ligado a todos da camada anterior.

Vou descrever aqui as contas...

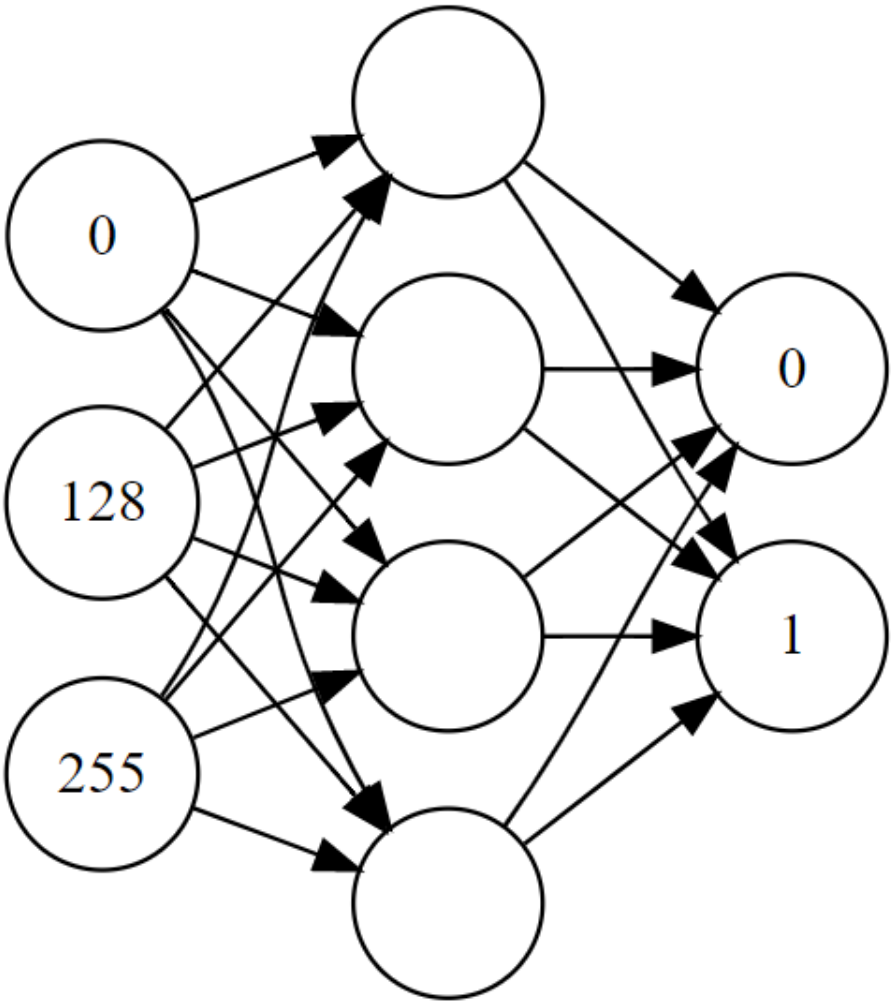

Figura 2.1: *Rede neural simples, o perceptron de única camada.*

Apêndice A

Código-Fonte e Pseudocódigo

Com a *package* listings, programas podem ser inseridos diretamente no arquivo, como feito no caso do Programa 4.1 ou importados de um arquivo externo com o comando \lstinputlisting, como no caso do Programa A.1.

Programa A.1 Máximo divisor comum (arquivo importado).

```
1  FUNCTION euclid(a, b)  ▷ The g.c.d. of a and b
2      r ← a mod b
3      while r ≠ 0  ▷ We have the answer if r is 0
4          a ← b
5          b ← r
6          r ← a mod b
7      end
8      return b  ▷ The g.c.d. is b
9  end
```

Trechos de código curtos (menores que uma página) podem ou não ser incluídos como *floats*; trechos longos necessariamente incluem quebras de página e, portanto, não podem ser *floats*. Com *floats*, a legenda e as linhas separadoras são colocadas pelo comando \begin{program}; sem eles, utilize o ambiente programruledcaption (atenção para a colocação do comando \label{}, dentro da legenda), como no Programa A.2[1]:

Programa A.2 Máximo divisor comum.

```
1  FUNCTION euclid(a, b)  ▷ The g.c.d. of a and b
2      r ← a mod b
3      while r ≠ 0  ▷ We have the answer if r is 0
4          a ← b
5          b ← r
```

cont ⟶

[1] listings oferece alguns recursos próprios para a definição de *floats* e legendas, mas neste modelo não os utilizamos.

```
    ⟶ cont
6               r ← a mod b
7           end
8           return b   ▷ The g.c.d. is b
9       end
```

Além do suporte às várias linguagens incluídas em listings, este modelo traz uma extensão para permitir o uso de pseudocódigo, útil para a descrição de algoritmos em alto nível. Ela oferece diversos recursos:

- Comentários seguem o padrão de C++ (`//` e `/* ... */`), mas o delimitador é impresso como "▷".
- ":=", "<>", "<=", ">=" e "!=" são substituídos pelo símbolo matemático adequado.
- É possível acrescentar palavras-chave além de "if", "and" etc. com a opção "morekeywords={pchave1,pchave2}" (para um trecho de código específico) ou com o comando \lstset{morekeywords={pchave1,pchave2}} (como comando de configuração geral).
- É possível usar pequenos trechos de código, como nomes de variáveis, dentro de um parágrafo normal com \lstinline{blah}.
- "$...$" ativa o modo matemático em qualquer lugar.
- Outros comandos LaTeX funcionam apenas em comentários; fora, a linguagem simula alguns pré-definidos (\textit{}, \texttt{} etc.).
- O comando \label também funciona em comentários; a referência correspondente (\ref) indica o número da linha de código. Se quiser usá-lo numa linha sem comentários, use `///` \label{blah}; "`///`" funciona como `//`, permitindo a inserção de comandos LaTeX, mas não imprime o delimitador (▷).
- Para suspender a formatação automática, use \noparse{blah}.
- Para forçar a formatação de um texto como função, identificador, palavra-chave ou comentário, use \func{blah}, \id{blah}, \kw{blah} ou \comment{blah}.
- Palavras-chave dentro de comentários não são formatadas automaticamente; se necessário, use \func{}, \id{} etc. ou comandos LaTeX padrão.
- As palavras "Program", "Procedure" e "Function" têm formatação especial e fazem a palavra seguinte ser formatada como função. Funções em outros lugares *não* são detectadas automaticamente; use \func{}, a opção "functions={func1,func2}" ou o comando "\lstset{functions={func1,func2}}" para que elas sejam detectadas.
- Além de funções, palavras-chave, strings, comentários e identificadores, há "specialidentifiers". Você pode usá-los com \specialid{blah}, com a opção "specialidentifiers={id1,id2}" ou com o comando "\lstset{specialidentifiers={id1,id2}}".

Referências

[Géron 2019] Aurélien Géron. *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow*. 2º. O'Reilly, 2019 (citado nas pgs. 1, 3, 4).

[Grus 2016] Joel Grus. *Data Science do Zero*. 1º. O'Reilly, 2016 (citado na pg. 1).

[Kopec 2019] David Kopec. *Problemas Clássicos de Ciência da Computação com Python*. 1º. Novatec, 2019 (citado nas pgs. 1, 3, 5).

Índice Remissivo